Eirik Plahte
Christian Riksvold

Quality Assurance of Exam Grading using Norwegian BERT Models

Master's thesis in Informatics/Computer Science
Supervisor: Trond Aalberg
June 2023

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Eirik Plahte
Christian Riksvold

Quality Assurance of Exam Grading using Norwegian BERT Models

Master's thesis in Informatics/Computer Science
Supervisor: Trond Aalberg
June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

ABSTRACT

The exam grading process needs optimization in tandem with improvements in technology. With Artificial Intelligence (AI) technology increasing in viability and availability, automatic grading in the school system is on the horizon. Fully automatic grading for answers in longer text form is however not quite achievable yet, with the main concern being the precision of the results. AI technology is excellent at many natural language processing tasks, but it is not devoid of fallibility. The school system can not afford to corrupt the work of students by using a tool with even the smallest possibilities of errors.

This thesis proposes a method that does not automatically grade the submissions, but rather supports the examiner in the process of grading. The method is implemented in an application called *Transformer-based Grade Revision Tool*, or TGRT. After the grading is done, TGRT will analyze the submissions along with their grades and flag instances where it believes the examiner has made an error. Test subjects have admitted to being affected by time of day, mood and other personal factors when grading papers, which can lead to inconsistencies in strictness and thoroughness. TGRT will limit the amount of mistakes like this by comparing the contents of the submissions by means of AI and natural language processing and warn the examiner if the grades do not reflect the contents.

Three main aspects of this process are explored in this thesis. The first is how to make the tool compare the answers most accurately. The technology chosen is transformers and BERT, which are AI models. Because of the black box property of transformers, it is hard to know why answers are considered to be similar. The second aspect is therefore how to explain this to the examiner. This needs to be as intuitive and easy to understand as possible for the tool to be useful in a real setting within a non-substantial time. This leads into the final aspect, revolving around how such a tool is perceived by the users.

The findings show that TGRT is useful at identifying errors in the grading process, but the means of explaining the similarity has not met the desired goals. The test subjects mostly resorted to reading the students' full answers instead of using the supporting features, which would lead to an increase in time needed for grading. The tool was however sporadically excellent at pointing out mistakes the examiners agreed with, and the test subjects were very positive to the prospect of using TGRT in the grading process.

SAMMENDRAG

Sensureringsprosessen ved skoleeksamen trenger optimalisering i takt med forbedring av moderne teknologi. Med tanke på opptrappingen av kvaliteten til kunstig intelligens, er automatisk karaktersetting i skolesystemet like rundt hjørnet. Helautomatisk vurdering er imidlertid ikke helt oppnåelig ennå, hvor hovedproblemet er nøyaktigheten av resultatene. Kunstig intelligens (KI) er utmerket på mange språkbehandlingsoppgaver, men det er ikke ufeilbarlig. Skolesystemet kan ikke risikere at elevenes arbeid kan bli urettferdig vurdert ved å bruke et verktøy med selv de minste muligheter for feil.

Denne masteroppgaven presenterer en metode som ikke automatisk retter eksamensinnleveringer, men som heller støtter sensoren i karaktersettingen. Metoden er implementert i applikasjonen *Transformer-based Grade Revision Tool*, eller TGRT. Etter at karakteren er satt, vil TGRT analysere innleveringene sammen med karakterene deres og markere tilfeller der det mener sensoren har gjort en feil. Testpersoner har innrømmet å være påvirket av tiden på dagen, humør og andre personlige faktorer ved sensurering, som kan føre til uregelmessig grad av strenghet og grundighet. Det foreslåtte verktøyet vil begrense antall slike feil ved å sammenligne innholdet i innleveringene ved hjelp av kunstig intelligens og språkbehandling og advare sensor dersom karakterene ikke gjenspeiler innholdet.

Tre hovedaspekter ved denne prosessen blir utforsket i denne masteroppgaven. Det første er hvordan verktøyet kan sammenligne svarene mest nøyaktig. Teknologien som blir brukt er transformatorer og BERT, som er KI-modeller. På grunn av black box-egenskapen til transformatorer, er det vanskelig å vite hvorfor noen svar regnes for å være like. Det andre aspektet er derfor hvordan man kan forklare til sensoren hvorfor verktøyet anbefaler det den gjør. Dette må være så intuitivt og enkelt å forstå som mulig for at verktøyet skal være nyttig i en reell setting og uten å koste sensor for mye ekstra tid. Dette leder inn til det siste aspektet, som handler om hvordan sensorene oppfatter TGRT.

Funnene viser at TGRT er nyttig for å identifisere feil i karakterprosessen, men hjelpemidlene som skulle forklare likheten har ikke oppfylt ambisjonene. Testbrukerne endte for det meste opp med å lese studentenes fullstendige svar i stedet for å bruke sammenligningsfunksjonene som i praksis ville ført til en økning i tiden som trengs for karaktersetting. TGRT var imidlertid sporadisk utmerket til å påpeke feil sensorene sa seg enige i, og de var veldig positive til bruken av TGRT i karaktersettingsprosessen.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AES Automated Essay Scoring

AGI Artificial General Intelligence

AI Artificial Intelligence

API Application Programming Interface

BERT Bidirectional Encoder Representations from Transformers

C2SA Cross2Self-attention

CoLDE Contrastive Long Document Encoder

ELMo Embeddings from Language Models

GloVe Global Vectors for Word Representation

IDF Inverse Document Frequency

IR Information Retrieval

LSTM Long Short Term Memory

MLM Masked Language Model

MSE Mean Squared Error

NER Named Entity Recognition

NLP Natural Language Processing

nltk Natural Language Toolkit

NSP Next Sentence Prediction

NTNU Norwegian University of Science and Technology

NorLM Norwegian Language Model

PCC Pearson Correlation Coefficient

QWK Quadratic Weighted Kappa

RAKE Rapid Automatic Keyword Extraction

RNN Recurrent Neural Network

SBERT Sentence-Bidirectional Encoder Representations from Transformers

SGCC Similarity-Grade Correlation Coefficient

SSH Secure Shell

TF Term Frequency

TF-IDF Term Frequency - Inverse Document Frequency

TGRT Transformer-based Grade Revision Tool

VPN Virtual Private Network

WSD Word Sense Disambiguation

CHAPTER ONE

INTRODUCTION

1.1 Motivation

Grading exams is an arduous and time-consuming task. Certain types of tasks (such as multiple choice questions) can be automatically graded, but descriptive text answers are a different matter. In several academic fields, such as in the humanities and social sciences, answers often contain reflective discussion, and there may be multiple equally valid ways to approach a question. The lack of one correct answer makes it difficult if not impossible to create a tool for automatically grading such answers in a fair and satisfactory way. There are however other ways that AI can aid in the grading process.

When an examiner is grading a batch of exams, there are several factors that might affect their judgement. How tired they are, their mood, the quality of the last few exams they graded and how long their grading session has been might all affect the judgement of the examiner, even if only subconsciously. In the case of several examiners splitting a submission set between them, they might have different criteria for what makes a good grade. This can lead to unfair results, where some exams might be graded lower than they deserve, and vice versa. This problem was observed in a study conducted in 2018, in which several instructors graded the same set of solutions to a programming task. The results found that the grades given were very inconsistent from one instructor to the other [1]. Moreover, data from 2020 uncovered that 40 % of students' appeals against grades across ten Norwegian universities and colleges resulted in a different grade [2]. This is further proof that mistakes often occur in the grading process. Because of the reasons outlined above, there is a need for an automatic method of quality assurance that can warn the examiner when they likely are about to give an unfair grade.

1.2 Project description

The goal of this thesis is to examine the possibility of verifying the fairness of grades by comparing the contents of every answer to each other. The transformer based language model Bidirectional Encoder Representations from Transformers (BERT), which was developed by Google and published in 2018 [3], was chosen to extract the contents of the answers. This model has been proven to achieve state-of-the-art results in a number of different Natural Language Processing (NLP) tasks, and has rendered previous models such as Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM) somewhat obsolete. There are several Norwegian variants of BERT, which were tested in their ability to create accurate embeddings of Norwegian text answers. After creating the embeddings, cosine similarity was used to ascertain a numeric value of how similar answers are.

Transformer-based models naturally contain a *black box* property, meaning that an output is given without any indications of how it was calculated. In practice, an examiner needs to understand why an answer is graded unfairly in order to change it. One way to accomplish this is to read both answers again and examine them juxtaposed, but this alone could lead to a great increase in the time needed for grading. To assist the examiner in the examination of the similar answers, several methods that demonstrate *how* they are similar were also tested. This includes extractive methods like Term Frequency - Inverse Document Frequency (TF-IDF) and Rapid Automatic Keyword Extraction (RAKE) that highlight important terms used in the texts, and generative methods that generate summaries, keywords and themes based on the texts. These methods could help the examiner understand how the answers are similar, but also how they differ. An answer with a higher grade might include some crucial details that the one with the lower grade does not have, which could indicate that the two answers deserve different grades after all.

Based on the presented goals, the following research questions were decided upon:

- **RQ1: How can the similarities between exam answers be calculated?**
- **RQ2: What features can be extracted for assisting the examiners in analyzing similar answers?**
- **RQ3: How will a tool for detecting inconsistencies in grading be perceived by examiners?**

In order to answer these questions, a tool called Transformer-based Grade Revision Tool (TGRT) was developed. It consists of a server where most of the heavy calculations are made, and a client that displays the results to a user. The resulting answer pairs with the highest similarity are sorted into recommendations. A recommendation lists the answers that are deemed most similar to the answer in question, and proposes either an increase or decrease in grade based on the grades of the most similar answers. The examiner can then consider the recommendations, examine the reasoning behind them, and decide on a final grade.

Two types of tests were performed in this thesis. First, qualitative tests of the different Norwegian BERT models were performed in order to judge their ability to rank texts according to similarity, and to observe how the similarity scores are affected by changes to the base text, such as spelling errors, synonyms, adding irrelevant information or excluding certain parts of the text. These tests were used as the basis for deciding which BERT model to integrate into TGRT. After the development of TGRT was completed, two qualitative user tests were conducted in order to observe how the tool works in practice with members of the target group. This tested both the degree in which the test subjects agreed with the recommendations made by the tool, and the usefulness of the supporting features. The test subjects also provided their overall impressions of the tool and the concept behind it.

1.3 Contributions

Automatic grading has been a widely researched field for several years, but the use of AI to assist in the grading process by detecting inconsistent grading for similar answers has not been explored thus far. Text similarity calculation is a thriving area in research, but this is mainly done for English texts. Little research has been done with Norwegian AI models, such as the different variants of BERT that are explored in this thesis.

The contributions of this thesis are therefore as follows:

1. A review of the existing literature regarding automatic grading and the calculation of text similarity.
2. A proposal of assisting exam grading by detecting instances of inconsistent grading using BERT.
3. An analysis of how well the different Norwegian BERT rank texts by similarity.
4. A proposal for extractive and generative methods to be used in order to understand similarities between exam answers.
5. TGRT, a tool for assisting exam grading that implements the methods described above.

CHAPTER

TWO

THEORY

To fully understand the methods and technologies used in this project, it is important to know how the theory behind them works. Not every method and technology described in this chapter was used in this project, but they all appear with some frequency in the related work. This chapter is an expanded version of the theory chapter from the report written in conjunction with last year's preparatory project [4].

2.1 Information Retrieval

Information Retrieval (IR) is a big field in computer science, which serves as a basis for many of the concepts used in this thesis. It is relevant because ranking the similarities between an exam answer and the rest of the answers in its set can be thought of in the context of an IR system, where the compared answer is a query, and the rest of the answers are the document collection.

IR can be defined as the process of finding documents of an unstructured nature that satisfies an information need from within large collections [5]. That the data is unstructured means that it is not arranged according to a preset data model or schema, unlike for example the data that is stored in relational databases. Unstructured data is stored in its native format and is not processed until it's being used. It can be stored in a variety of file formats, but it is usually text.

IR is based on having a query and retrieving a set of documents from a collection. The documents in the retrieved set are considered to be relevant to the terms found in the query. The query is often composed of words that can help identify the relevant documents, but it can also consist of an entire document. When trying to find documents that are the most similar to one specific document, the latter approach is used. Most IR methods support ranking of the results based on their relevance to the query. In order to rank the documents, similarity scores are calculated between the input query and each of the documents.

As well as ranking of retrieved documents, IR also allows for partial matching if none of the documents in the collection contain any exact matches with the query. Aside from retrieving relevant documents, IR also covers further processing of a set of retrieved documents, such as with clustering, which is the task of coming up with a good grouping of the documents of a set based on their contents.

2.2 Natural Language Processing

NLP is a subfield of AI that centers around helping computers better process, interpret, and understand human language and speech patterns [6]. NLP is relevant to this thesis because several syntactic NLP tasks will be used for text preprocessing, while semantic NLP tasks are relevant for conveying information to the examiner about the content of the answers.

NLP uses algorithms to identify and interpret natural language rules so unstructured language data can be processed in a way the computer can actually understand, thereby attempting to bridge the computer-human speech gap. The algorithms used by NLP utilize both the syntax and the semantics of language in order to process the data, extract meaning from it, and provide a response [6]. Some syntactic techniques used by NLP include the following:

- Lemmatization: the process of grouping together the inflected forms of a word so that they can be analyzed as a single item, identified by the word's lemma, or dictionary form (for example, the lemma of the words "breaks", "broke", "broken", and "breaking" is "break").
- Word segmentation, or tokenization: dividing a large piece of continuous text into tokens (i.e. words). In English and many other languages, this is a trivial task, since all words are separated by white spaces [7].
- Morphological segmentation: breaks down words into smaller units called morphemes. Morphemes express a direct meaning and cannot be further separated into smaller parts. The word "unsegmented" can be separated into three morphemes: "un", "segment" and "ed".
- Part of speech tagging: identifying the part of speech for each word (as in whether it is a noun, verb, adjective etc.).
- Stemming: cutting down words to their word stem (for example, "running" and "runs" are reduced to simply "run").
- Parsing: analysis of the grammar of a given phrase or sentence, usually by constructing a parse tree, which represents the syntactic structure of a given sentence.

Some semantic techniques utilized by NLP algorithms include the following:

- Named Entity Recognition (NER): determining which tokens in a text correspond to proper names, and what the type of each such name is (i.e. whether it is a person, place, organization, etc.).
- Word Sense Disambiguation (WSD): involves giving meaning to a word based on context. Many words have several entirely different meanings,

such as the word "well", which is both a synonym of "good", and a word for a deep hole dug to obtain water, among other things. The correct meaning is induced by analyzing the context, i.e. the rest of the sentence.

- Terminology extraction: automatically extracting key terms from a corpus.
- Sentiment analysis: analyzing text data to identify and extract the meaning or intent of the text.

Some examples of common NLP tasks include automatic summarization, speech recognition, text-to-speech, autocompletion, machine translation and chatbots.

2.3 Keyword extraction

Keyword extraction is an NLP task whose goal is to extract words or phrases from a text that are indicative of this text's subject [8]. This is relevant for the supporting features in TGRT, which show the similarities and differences between two text answers in a way that is more intuitive to the examiner than only the results from BERT. Two methods for keyword extraction were implemented: TF-IDF and RAKE. The former is based on the vector space model and is a specific way of weighting the vectors. The vector space model is a core concept in IR that is also important in order to understand cosine similarity, which will be expanded on in section 2.4. RAKE is a keyword extraction algorithm that selects key phrases (meaning consecutive series of words) that represent the text's content.

2.3.1 TF-IDF

One of the most common ways of categorizing documents or texts based on their contents is with the vector space model. In the vector space model, text is represented by a vector of terms. Every term in the vocabulary then becomes an independent dimension in a high dimensional vector space [9]. If a term is found in a document, it gets a non-zero value in the document vector along the dimension corresponding to the term.

Combining the vectors results in a matrix where each row represents a document and each column represents a word. In the boolean matrix model, which is one of the simplest IR models, the cells simply indicate if the word is present in the document or not. A step further would be to represent the documents in a Term Frequency (TF) matrix where the amount of times a word is present in a document is stored instead of a boolean value for whether it is present or not. The vectors usually get normalized so that the proportions are easier to understand.

$$TF(t) = \frac{n_t}{max(n)} \tag{2.1}$$

The TF of term t is given by n amount of times the term is used in a document (n_t) divided by the maximum number of times the term is used in any document *(max(n)) [10].*

Another model is for Document Frequency (DF) where instead of looking at how many times a term is used in a document, it counts how many documents a term is present in. This results in a variable that represents how frequently a term is used across the documents. With a high DF, the term is likely not unique to the document. Because it is desired to have a low DF, the Inverse Document Frequency (IDF) is often used instead, where the higher the value, the fewer documents the term is present in.

$$IDF(t) = log(\frac{N}{(DF(t) + 1)}) \tag{2.2}$$

The IDF of term t is given by N amount of documents divided by the document frequency of term t plus one, so as not to divide by zero when a term is not present in the collection of documents. Log is used to prevent the numbers from reaching extreme heights when the document collection is very large [11].

The IDF and the TF can be multiplied together to form a TF-IDF score. This value represents both how often a term is used in a document and how frequent the term is used in the other documents in a collection.

$$\textit{TF-IDF}(t) = TF(t) \cdot IDF(t) \tag{2.3}$$

The formula for calculating TF-IDF for term t.

In many NLP tasks, it is important to remove stop words. Stop words are terms that are used frequently in text, but carry little meaning [12]. Examples are words like "the", "it", "not" and "a". They are removed from the calculations in order to not increase the data size and processing time with terms that can't represent the document well. In TF-IDF, stop words are removed so that common filler words will not get a high score and muddle the results.

2.3.2 RAKE

RAKE is a domain independent keyword extraction algorithm which tries to determine key phrases in a body of text by analyzing the frequency of word appearance and its co-occurrence with other words in the text [13]. The following short text will be used an example to explain how the algorithm works:

Keyword extraction is a subfield of text mining. It is very useful.

The algorithm starts by removing the stop words from the input text. Then the text is split at the stop word positions and punctuation characters. Thus, the words that occur consecutively without any stop words or punctuation characters between them are taken as candidate keywords. These candidate keywords are shown below, with the stop words that separate them grayed out:

Keyword extraction is a subfield of text mining. It is very useful.

Next, the frequency of all the individual words in the candidate keywords are calculated.

Word	keyword	extraction	subfield	text	mining	useful
Word frequency: *freq(w)*	1	1	1	1	1	1

Table 2.1: Word frequencies for every word in the text

Similarly, the word co-occurrence count for every individual word is calculated. This means that for every word, the amount of times that this word occurs in conjunction with every other word as part of a candidate keyword is calculated. This amount is called the degree of that word. This metric identifies words that often occur in longer candidate keywords.

Word	keyword	extraction	subfield	text	mining	useful
keyword	1	1	0	0	0	0
extraction	1	1	0	0	0	0
subfield	0	0	1	0	0	0
text	0	0	0	1	1	0
mining	0	0	0	1	1	0
useful	0	0	0	0	0	1
degree: *deg(w)*	$1+1=2$	$1+1=2$	1	$1+1=2$	$1+1=2$	1

Table 2.2: Degrees for every word in the text

Subsequently, a final score for each word is calculated by dividing the degree by the frequency. This score is higher for words that occur more in longer candidate keywords than individually.

Word	keyword	extraction	subfield	text	mining	useful
Score $= \frac{deg(w)}{freq(w)}$	$\frac{2}{1}=2$	$\frac{2}{1}=2$	$\frac{1}{1}=1$	$\frac{2}{1}=2$	$\frac{2}{1}=2$	$\frac{1}{1}=1$

Table 2.3: Scores for every word in the text

After each word has a score, the scores for the candidate keywords are calculated by summing the scores for their member words together. The higher the score, the more useful of a phrase it is considered to be. Finally, the keywords are sorted in descending order of their score value.

Keyword	Score	Explanation
keyword extraction	4	score(keyword) + score(extraction) = 2 + 2 = 4
text mining	4	score(text) + score(mining) = 2 + 2 = 4
subfield	1	score(subfield) = 1
useful	1	score(subfield) = 1

Table 2.4: Final ranking of the candidate keywords

2.4 Similarity functions

A key objective with this thesis is to inquire into how similarities between exam answers can be calculated. In order to calculate a numerical value that represents the degree of similarity between two texts, a similarity function is needed. One such function is cosine similarity, which is based on the vector space model. This function is compared to Jaccard's similarity, which is another prevalent similarity function in IR.

2.4.1 Cosine similarity

As mentioned in 2.1, IR methods usually rank the retrieved documents by relevance to the query. This is done by using a similarity function that calculates similarity scores for the input query and each of the documents. One such similarity function is cosine similarity. Cosine similarity uses the vector space model, and TF-IDF is often used as the weighting method for each term in the vector space.

After having represented the documents and the query in the vector space model, the cosine similarity function can be used to calculate similarity scores. The cosine similarity between a document d and query q is calculated as follows:

$$cos(d,q) = \frac{d \cdot q}{|d|\ |q|} = \frac{\sum_{i=1}^{N} w_i(d)\ w_i(q)}{\sqrt{\sum_{i=1}^{N} w_i(d)^2}\sqrt{\sum_{i=1}^{N} w_i(q)^2}} \tag{2.4}$$

N is the length of the vectors, while $w_i(d)$ and $w_i(q)$ are the i-th values in the vectors for the document and query, respectively. w corresponds to the weight given to a word.

The numerator contains the dot product of the vectors for the document d_j and the query q, while the denominator contains the lengths of the vectors multiplied by each other. To put it in simpler terms, the formula calculates the cosine of the angle between the two vectors. The cosine is used because it has the property of being 1.0 for identical vectors and 0.0 for orthogonal vectors. By calculating similarity scores between the query and each of the documents, the documents can be ranked in order of relevance. Cosine similarity can be used in other contexts than finding the similarity between a query and a document; it is applicable as a measure of similarity between two equal-length vectors in any situation.

2.4.2 Jaccard similarity

Another similarity measure is Jaccard similarity. Just like cosine similarity, it outputs a number between 0 and 1. A Jaccard coefficient of 1 indicates that the documents are identical, whereas a Jaccard coefficient of 0 indicates that there are no tokens (i.e. words) in common between the two documents. Jaccard similarity between document A and B is defined as follows:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{2.5}$$

The Jaccard similarity between two documents A and B is defined as the intersection of the two documents divided by their union.

Simply put, it is defined as the division of the number of tokens common to both documents by the total number of tokens in both documents [14]. There are some notable issues with using Jaccard similarity. Firstly, it does not consider term frequency. Secondly, Jaccard does not account for the fact that rare terms in a collection are more informative than frequent terms. These issues are resolved by using a TF-IDF score for weighting each term in the collection. Jaccard similarity can however be useful in cases where duplicate words do not matter.

2.5 Machine Learning

Machine learning is a subfield within AI that is dedicated to understanding and developing methods that let machines "learn" [15]. This is mainly done by building models based on training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning models are used in several different NLP tasks. One of these models is BERT, which was implemented in TGRT for creating embeddings for exam answers. In order to provide some context for this model, there are several significant precursors that need to be explained, namely RNN, LSTM, and the transformer model.

2.5.1 Recurrent Neural Network

One challenge with neural networks is to provide context for what the model tries to understand. Looking at a sentence, the model can understand the word "King", but not ascertain if the king is an actual king of a kingdom or if it is a chess piece. An RNN is a version of the traditional neural networks that lets the model memorize some of the earlier context. It works like other neural networks, but has a memory state that goes in a loop. This way it can retain information from earlier work.

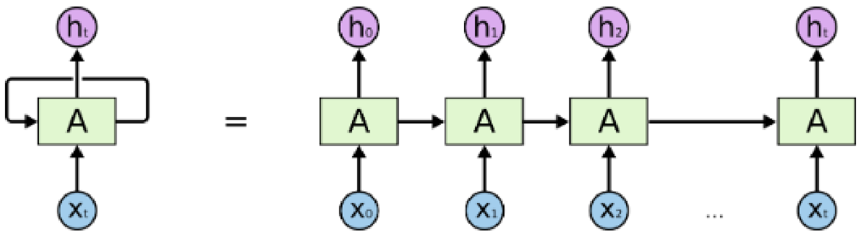


Figure 2.1: An illustration of an RNN [16]. X represents the input term, while h represents the output encoding. A is the cell where the calculations are made. The cell outputs both the encoding result and a hidden weight that retains a bit of memory for the next operation.

A problem with RNN however, is that it has a very short term memory. If the gap between the context for the word and the word itself becomes very large, an RNN struggles to remember it. The bigger the gap, the more it fails to connect the information.

2.5.2 LSTM

An LSTM is a version of RNN, but has more complex calculations inside the cell and also outputs two vectors for the next iteration: the main memory state and the hidden state. The hidden state is the same as the output of the last iteration.

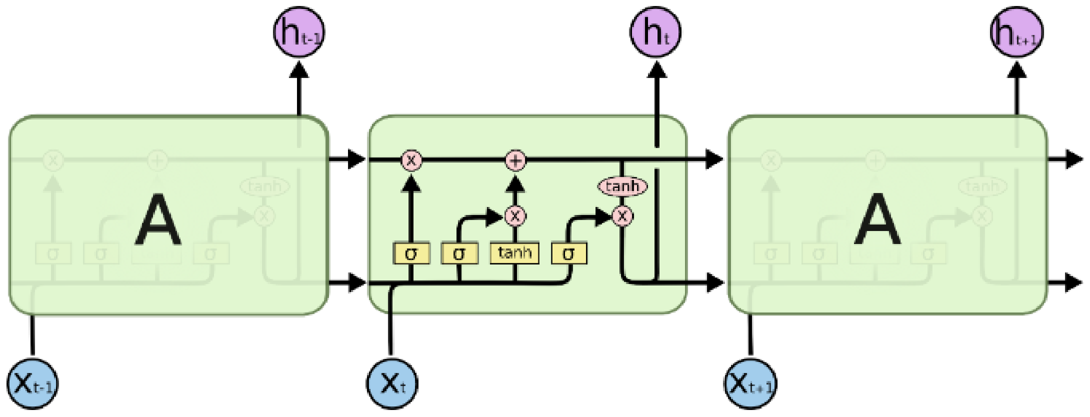


Figure 2.2: A representation of an LSTM [16]. X represents the input term, while h represents the output encoding. A is the cell where the calculations are made. The two states are represented as arrows moving between the cells. In practice, there is only one cell that updates its weights, but the figure displays cells next to each other to illustrate the concept.

2.5.3 Transformers

Transformers are another way of performing text analysis and operations. It uses an attention mechanism to understand the relationship of text context. They were introduced in 2017 and have become increasingly popular since then [17].

A transformer is structured into two main sections, encoder and decoder. One of the main calculations present in both encoder and decoder is the multi-head attention mechanism. The attention mechanism works a lot like AI in computer vision, which only pays attention to the most important parts of the input to save on computational effort. A transformer creates weights for each word in a sentence to decide which are most significant. This is done by processing the input through many transformer layers in the encoder section.

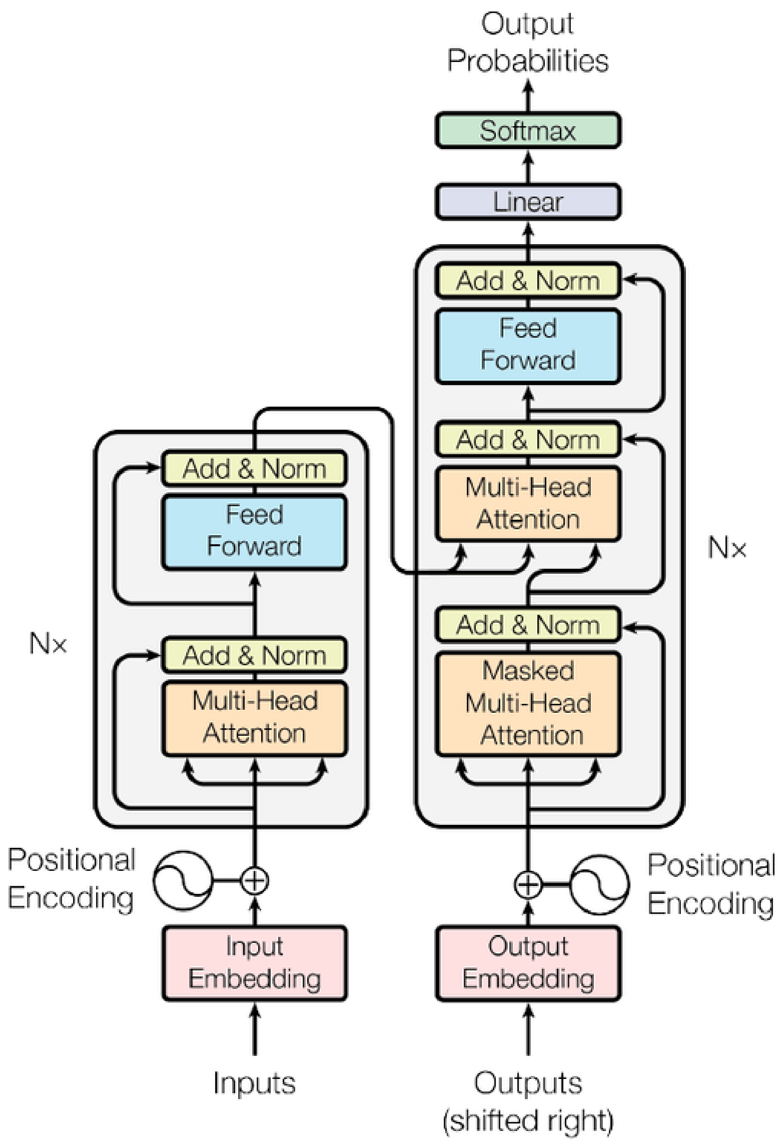
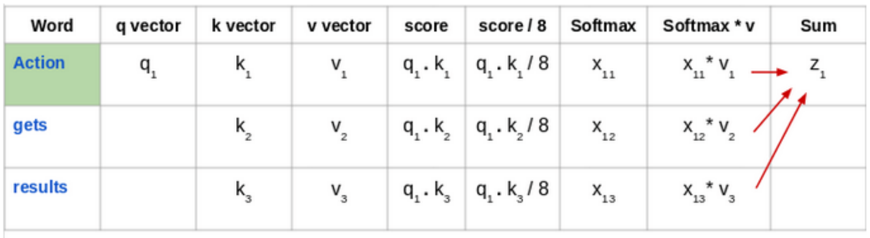


Figure 2.3: An overview of a typical transformer structure [18].

The input of both the encoder and the decoder are represented by a form of embedding. This embedding loses the positional information of the input sequence, hence the injection of the positional encoding vector after the embedding in figure 2.3 [18]. The word embedding inputted to an encoder is transformed into three vectors: queries, keys and values. They are calculated by "[...] multiplication of the word embedding against three matrices with learned weights" [18]. The query is the current word multiplied by a weight matrix, while the keys and values vectors are like the key-value pairs in hash maps multiplied by the same weight matrix. Keys are used to index the values, while the values are the information in the input words [19]. To calculate the attention score for a word, the dot product of the word's query vector is calculated with the key vectors of all the words. The result is then divided by the root of the dimension of the key vector and then normalized with the softmax function. Lastly, this is then multiplied with the value vectors of all the words and summed to a final score [20]. This represents the attention score of the word by how important it is to the other words in the sentence. It is given by the following formula [19]:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{Q \cdot K}{\sqrt{d_K}})V \tag{2.6}$$

Attention is calculated by using the vectors query Q, key K and value V. d_k *represents the dimension length of the key vector.*



Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum
Action	q_1	k_1	v_1	$q_1 \cdot k_1$	$q_1 \cdot k_1 / 8$	x_{11}	$x_{11} * v_1$	z_1
gets		k_2	v_2	$q_1 \cdot k_2$	$q_1 \cdot k_2 / 8$	x_{12}	$x_{12} * v_2$	
results		k_3	v_3	$q_1 \cdot k_3$	$q_1 \cdot k_3 / 8$	x_{13}	$x_{13} * v_3$	

Figure 2.4: An example of how attention is calculated. [18]. "Action" is the current term that calculates its attention to the other terms in a sentence. The dot product of query vector q with each key vector is calculated and softmaxed. Lastly it is multiplied with the value vector and added together for a sum of z which represents the attention of "Action" with regard to the sentence "Action gets results".

In a transformer, the attention scores are calculated many times independently of each other in parallel, hence multi-head attention. The outputs of the attention calculations are then concatenated and normalized. The encoder transforms the input sequence x $(x_1, ..., x_n)$, into a sequence of representations z $(z_1, ..., z_n)$. The decoder generates an output sequence from z, y $(y_1, ..., y_n)$. This is done in three parts: a masked multi-head attention mechanism where the output from each step of the decoders is given the same attention mechanism as the encoders, but has the section of words after the current word masked. This is because the model should not look at what is coming up in a sentence for context, only what has come before it. The next part is a multi-head attention mechanism that merges the output from part one and the output of the encoders. Lastly, the decoder post-processes the output with a fully connected feed-forward network [19]. The final output is softmaxed to select the final probabilities of words.

2.5.4 BERT

BERT is an open source NLP model that broke records for how well it handled language-based tasks when it came out in 2018 [17]. It uses a similar architecture as a transformer, but only uses the encoder stack. A unique function of BERT is that it is not only directional like transformers, but bidirectional, meaning it looks for context both backwards and forwards.

Before BERT is used, it is pre-trained on a large corpus to learn basic language similarities and connect words that have bindings. It can then be fine-tuned to be especially competent in one field of knowledge. The pre-training makes

BERT suitable for several use-cases: text summarization, question answering, text similarity, next-sentence prediction, feature extraction, masked word prediction and more [21].

When training a language model, it can be difficult to set prediction goals. A usual method is to make it predict the last word in a sentence, but that makes it only take into account the context behind it and limits the potential for learning [22]. This is why BERT is trained with primarily two strategies. The first is Masked Language Model (MLM), where a randomly chosen word is masked out with a [MASK] token and the model then will predict what word is masked. The other is Next Sentence Prediction (NSP) where two sentences are passed in, and the model will predict if they logically should be adjacent.

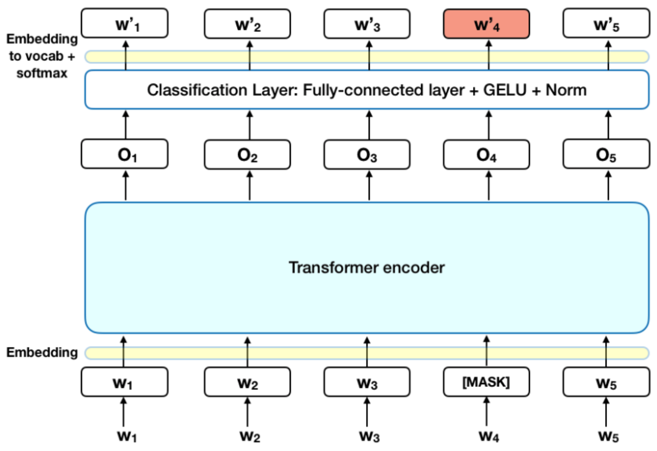

W'1
W'2
W'3
W'4
W'5
Embedding to vocab + softmax
Classification Layer: Fully-connected layer + GELU + Norm
O1
O2
O3
O4
O5
Transformer encoder
Embedding
W1
W2
W3
[MASK]
W5
W1
W2
W3
W4
W5

Figure 2.5: An illustration of an MLM process [22]. Word tokens are inputted to the encoder where w_4 is masked. The classification layer then predicts what word should be in place of the masked token with the outputted term w'_4.

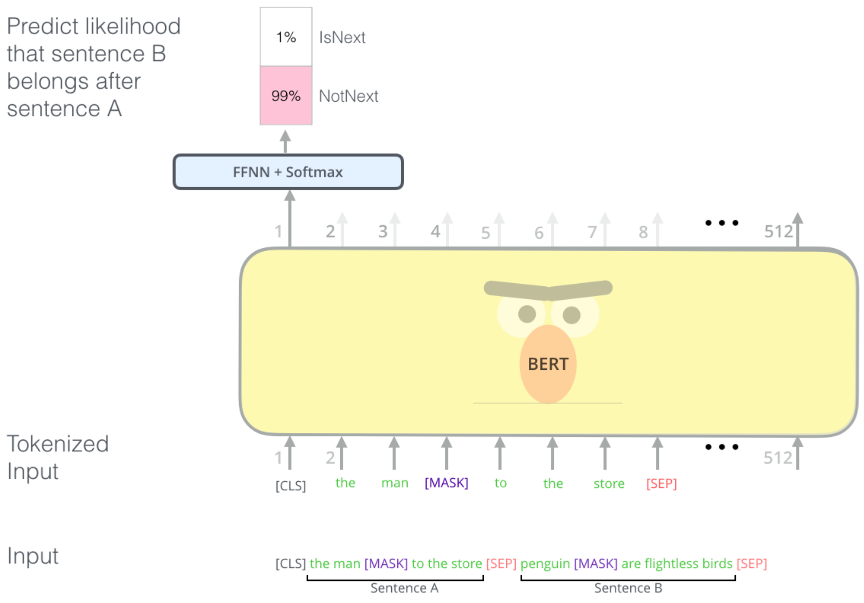

Predict likelihood that sentence B belongs after sentence A
1%
IsNext
99%
NotNext
FFNN + Softmax
1
2
3
4
5
6
7
8
512
BERT
Tokenized Input
[CLS]
the
man
[MASK]
to
the
store
[SEP]
Input
[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]
Sentence A
Sentence B

Figure 2.6: An example of an NSP process [21]. Two sentences A and B are inputted with a separator token ([SEP]) between them. The outputted encoding of the classification token ([CLS]) is processed by a feed forward neural network and softmaxed to determine if the two sentences are in succession or not.

The input of BERT is structured into a sequence of tokens, where each word has its representation. The start of an input sequence has a classifier token [CLS] and the end of each sentence has a separator token [SEP]. In the output of MLM, the item in place of the masked token is the predicted word in embedded form, while in NSP the classifier will tell if they are connected. The output of the encoder stack is passed through a classification layer, where the prediction will be made. In case of NSP, there are two possible answers for the classifier to make; either IsNext or NotNext. In the case of MLM, there are a lot of possibilities. The masked word could be anything in a given dictionary.

2.5.5 Siamese Networks and SBERT

Siamese neural networks are networks that contain multiple subnetworks. The subnetworks are structured identically with the same weights. They are usually utilized to calculate the similarity between two items. They are trained by inputting triplets of items; one base item, a second that is defined as similar to the base and a third that has very little similarity to the base item. The two pairs of base item with similar item, and base item with dissimilar item have their similarity scores calculated. The output is used for calculating a loss score for updating the weights in both models simultaneously [23].

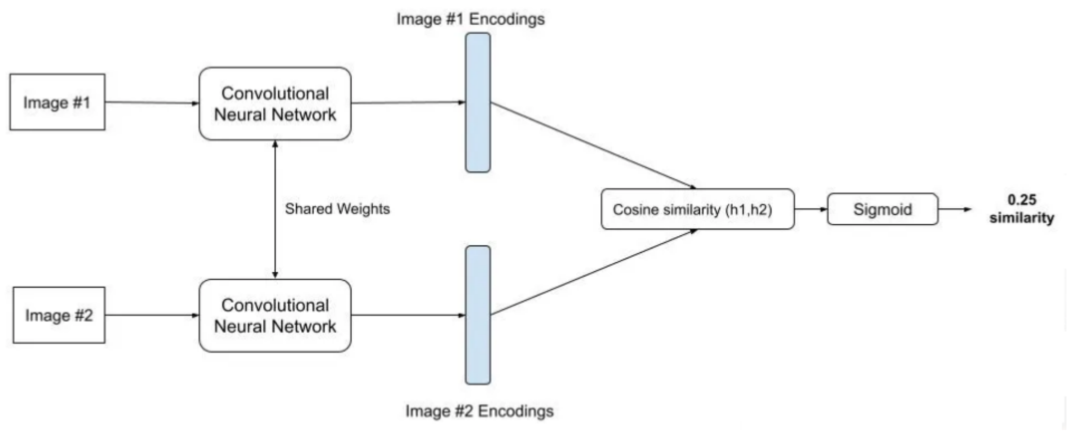


Figure 2.7: An example of a Siamese Network with two subnetworks measuring image similarity. [23]

Sentence-Bidirectional Encoder Representations from Transformers (SBERT) (also known as Siamese BERT) was presented in paper published in 2019 [24], and uses a Siamese architecture with BERT as parallel models. The output of BERT is pooled by a mean pooling operation to derive a fixed sized sentence embedding. This method calculates the average values of all output vectors, but it is also possible to only use the classification token [CLS] or calculating a max-over-time of the output vectors. The most optimal is however to use the mean method [24]. It can be used for several purposes like large-scale semantic similarity comparison, clustering and information retrieval via semantic search.

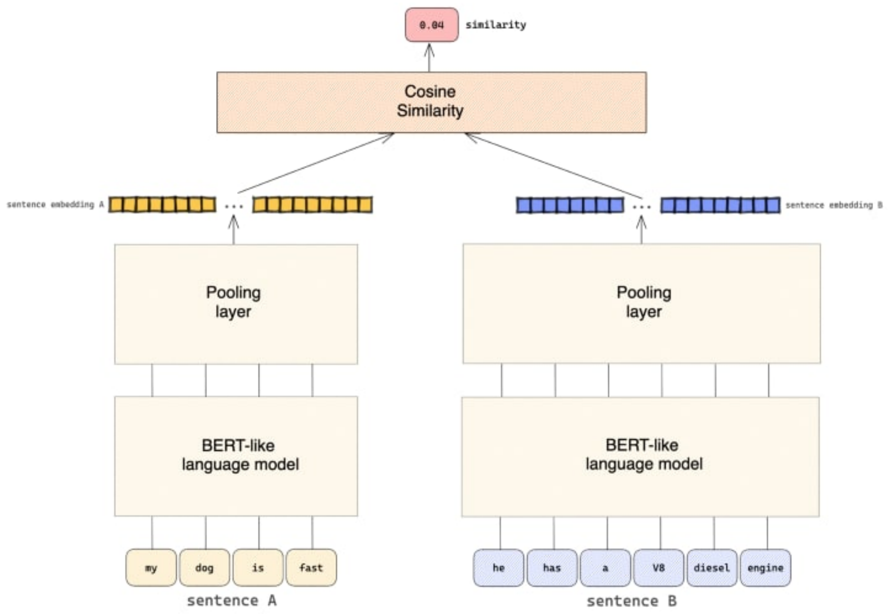


Figure 2.8: An illustration of how SBERT uses a Siamese architecture with pooling layers. [25]

2.5.6 Models pre-trained in Norwegian

Norwegian Language Model (NorLM) is an initiative for developing large scale language models for Norwegian. It is a collaboration between the SANT project (Sentiment Analysis for Norwegian Text), NLPL (The Nordic Language Processing Laboratory) and EOSC-Nordic (European Open Science Cloud) coordinated by the Language Technology Group (LTG) at the University of Oslo [26]. They developed a version of BERT that was pre-trained on a Norwegian corpus. NorBERT was released in 2021 and was trained on Norwegian Wikipedia articles both in "Bokmål" and "Nynorsk", along with a news article corpus. In 2022, they released NorBERT 2 with a much higher word count in training data and a vocabulary of 50 000 compared to NorBERT 1's 30 000 [27].

The National Library of Norway has via the project "NoTraM - Norwegian Transformer Model" developed two BERT variants pre-trained on the Norwegian language [28]. Their models NB-BERT-Base and NB-BERT-Large were trained to work on all types of Norwegian, both old and modern texts and texts where foreign languages like English are incorporated into it. To accommodate for the mixed language support, the models build upon the multilingual version of BERT, mBERT. In addition to mBERTs pre-training, the models got trained on the Colossal Norwegian Corpus containing 18 billion words approximately. Their results show better or on par results in most categories when compared to mBERT and NorBERT 1 [28].

As explained in subsection 2.5.3, a transformer's encoder is built with many layers stacked on top of each other. BERT-Base has 12 layers and BERT-Large has 24. As shown in Figure 2.9. BERT-Large also has more attention heads, parameters or weights and hidden layers than the base version. BERT-Base has 12 attention heads, 110 million parameters and 768 hidden layers, while BERT-Large has 16 attention heads, 340 million parameters and 1024 hidden layers [29]. The structure of the large and base variants extend to NB-BERT-Base and NB-BERT-Large as well, which are tested in this thesis.

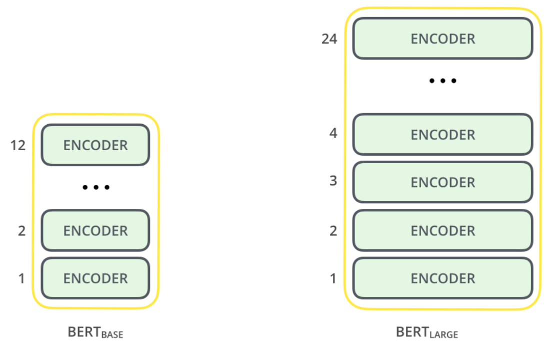


Figure 2.9: Encoder layers stacked on top of each other in BERT-Base and BERT-Large [29].

Another offshoot of NB-BERT-Base is their SBERT model NB-SBERT-Base [30] (hereafter referred to as NB-SBERT). It is a SentenceTransformers model trained on the Norwegian Multi-Genre Natural Language Inference (MNLI) [31] which is a collection of 433 000 pairs of sentences that includes textual entailment information. The Norwegian MNLI is translated into Norwegian using Google Translate. The dataset includes a section that is suited for training SBERT models with triplets of base-entailment-contradiction. It also contains sections that mix both Norwegian and English. The SBERT model maps sentences and paragraphs into a 768 dimensional dense vector space. Because of the MNLI dataset it is trained on, it should calculate that English and Norwegian sentence pairs with similar meaning has a high similarity value.

2.6 Evaluation measures

Evaluation measures are ways of assessing how well a system performs. It can both be used to describe methods for evaluating IR systems and machine learning models. The measures described below were all used in the relevant literature further examined in chapter 3.

In the context of IR, evaluation measures are ways of assessing how well search results satisfy the users' query intents. In order to evaluate this, several measures that take the relevance of the retrieved documents into account are used. Although the relevance of a document might be thought of as measurable on a scale (some relevant documents are less relevant than others), for simplicity's sake it is either categorized as relevant or non-relevant. Relevance is assessed relative to an information need, not a query [5]. It does not matter if a document contains all the words of the given query if the user's information need is still unfulfilled. Three of the most common measures that are used to evaluate IR systems are precision, recall and f-score.

2.6.1 Precision and recall

Precision is defined as the fraction of relevant documents among the retrieved documents, which is given by the following formula:

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|} \tag{2.7}$$

Recall is defined as the fraction of the relevant documents that are successfully retrieved. It is given by the following formula:

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|} \tag{2.8}$$

Precision and recall trade off against one another; for example, by retrieving every single document in a collection, the recall will be 1 (i.e. the highest possible value), while the precision will likely be very low. Whether to prioritize precision or recall depends on the circumstances of the user and the system. Search engines for web browsers typically aim for a high precision; ideally every result on the first page should be relevant, but the average web surfer has no interest in finding *every* relevant document. On the other hand, people searching their hard disks typically want a high recall; they want to find which ever file(s) they're looking for, and as long as that is achieved it does not matter if they see some non-relevant files as well. Other contexts in which recall is usually prioritized include legal and intelligence systems [5].

2.6.2 F-score

F-score is a measure that trades off precision P versus recall R. It is a weighted harmonic mean of P and R, and is given by the following formula [5]:

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R} \quad \text{where} \quad \beta^2 = \frac{1-\alpha}{\alpha} \tag{2.9}$$

α and β are weights that are used to emphasize either precision or recall. Here α is between 0 and 1, whereas β is between 0 and infinity. The default balanced F-score equally weights precision and recall, which means that $\alpha = 1/2$ and $\beta = 1$. When $\beta = 1$ (which is very common), the formula on the right is simplified to the following:

$$F = \frac{2PR}{P+R} \tag{2.10}$$

However, using an even weighting like the one shown above is not the only option. Values of $\beta < 1$ emphasize precision, whereas values of $\beta > 1$ emphasize recall. The reason for using harmonic mean rather than arithmetic mean (i.e. adding P and R together and dividing by 2) is that it would lead to some highly misleading results. As mentioned previously, if you retrieve every document in the collection, the recall will be 1 ($R = 1$), while the precision will likely be very low ($P \approx 0$), leading to an arithmetic mean of $\frac{1+0}{2} = 0.5$, which is misleading given that the system just retrieved every document.

2.6.3 Pearson correlation coefficient

While the measures described above are used for evaluating IR systems where documents are labeled as either relevant or non-relevant, other measures are used to evaluate different systems of relevance. One such type of system would be a machine learning model that labels every document in a set, and is evaluated by how well its labelling corresponds to that of a human. A concrete example of such a system is an automatic grader. An automatic grader analyzes the contents of every exam in an exam set and gives each of them a grade. The quality of this model can then be evaluated by calculating how the grades given by the model correspond to those given by the examiner for the course. One way to calculate this is to use the Pearson Correlation Coefficient (PCC).

The PCC is a parametric statistic that measures the linear relationship between two sets of variables [32]. It is defined as the ratio between the covariance of two variables and the product of their standard deviations. Thus, it can be thought of as a normalized measurement of the covariance, where the result always has a value between -1 and 1. The measure only reflects a linear correlation of variables, and ignores other types of relationships and correlations. The PCC for a pair of variables X and Y is defined as follows:

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{2.11}$$

The PCC ρ for variables X and Y is given by the covariance cov of X and Y divided by the product of the standard deviations σ for X and Y.

A high PCC implies that there is a strong correlation between the two variables. In the automatic grading example described above, a high PCC would mean that there is a strong correlation between the grades given by the AI model and the grades given by the human examiner, implying that the model is good at grading exams in a similar fashion to humans. Another measure that can be used to evaluate the automatic grader is the Mean Squared Error (MSE).

2.6.4 Mean squared error

The MSE is a measure used to calculate the amount of error in statistical models. It is defined as the average squared difference between the values observed in a statistical study and the values predicted by a model [33]. The MSE is given by the following formula:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.12}$$

The MSE for a set of predictions and their corresponding observations is given by the sum of the squares of all errors divided by n amount of observations, where an *error is defined as the difference between the observed value Y and the predicted value $\hat{Y}$.*

The reason for squaring the differences is to prevent a situation in which the MSE would be 0 despite there being significant errors. Since some observations will be higher than the prediction (meaning a positive difference), and some observations will be lower (meaning a negative difference), there is a possibility that the differences add up to 0, which would be highly misleading. An MSE of 0 *should* imply that no errors are present, which is not the case in this instance. By squaring the differences, this problem is avoided, as it prevents any negative values from being included in the sum.

In the automatic grading example described in subsection 2.6.3, the grades given by the AI model can be thought of as predictions, while the grades given by the human examiner can be thought of as observations. A low MSE would thus imply that the AI model is good at grading exams in a similar fashion to humans.

2.7 Summarization methods

Text summarization is the process of converting a longer text into a shorter summary. The possibility of implementing a summarizer in TGRT was considered from the start of the project because it would let the examiner get a quick overview of the content of answers, and could thus spare them significant time. After inquiring into different types of summarizers, it was decided that OpenAI's Application Programming Interface (API) would be used to generate abstractive summaries for the answers. While this approach abstracts away the actual generation of the summaries, having a grasp on the theory behind it can still be useful.

There are two main types of summarization: extractive and abstractive/generative. Extractive summarization detects the most crucial sentences and extracts them from the text. The resulting sentences become a summary. Abstractive or generative summarization is much more advanced, as it generates the summary from scratch. It needs to identify the most representative parts of the text and retell the meaning in as few words and sentences as possible. An abstractive summarizer takes a lot of computing time and can be very inefficient for long texts, but gets better and better with new technology [34].

2.8 Edit distance

The edit distance or Levenshtein distance is a measure of how similar two words or strings are to each other [35]. This has several applications in NLP, most prominently the detection of spelling errors, which was something that was considered to be implemented in TGRT. Edit distance measures how many single-character operations are needed to transform one string into another string. There are three base methods of transformation that have their own cost adding to the edit distance score: insertion, deletion and substitution. The cost for addition and deletion is usually one point while substitution is two. This is because substitution can be broken down into a deletion and an addition.

An example could be to transform "Squirrel" into "Quarrel". The operations needed are a deletion of the "s" and a substitution of the "i" with an "a". The edit distance in this example is 3.

CHAPTER

THREE

RELATED WORK

The aim of this project is to explore how text analysis methods can be applied for aiding exam grading by detecting instances of grading discrepancies for similar answers. No previous research on that exact problem could be found, but the related problem of automatically grading text exam answers has been researched extensively. Moreover, the field of text similarity, which is very relevant for this thesis, has also been widely researched. In order to start exploring the problem for the thesis, it was therefore necessary to get familiar with the existing literature and previous research that had been done in these fields. This included both reading the literature on how AI has been utilized in the aiding of exam grading (chiefly automatic grading), and the technology and methods that can be used for the purpose of this thesis, such as text pre-processing, similarity functions and transformers. This chapter is structured so that each subchapter explains the research that has been done in a certain relevant topic. Finally, there will be a discussion which will summarize the findings and identify key takeaways. This chapter is an expanded version of the literature review chapter from the report written in conjunction with last year's preparatory project [4].

3.1 Automatic grading

The automatic grading of exams has been a thriving area in research for several years. For the purpose of this thesis, only automatic grading of exams with text answers is relevant, as the automatic grading of multiple choice exams is a trivial task that has been widely implemented in the school system already. Automatic grading systems for text answers generally use a similarity function to compare the students' answers to the teacher's ideal answer. Several state-of-the-art NLP methods have been used for the text embeddings that form the basis for comparing answers, including LSTM and BERT, along with variants of them.

3.1.1 Using Siamese Manhattan LSTM for grading

Bahel and Thomas [36] propose a text analysis based automated approach for automatic evaluation of descriptive exam answers. They present an architecture where the text similarity model is based on Siamese Manhattan LSTM (MaLSTM).

The first step in their proposed system was for the examiners to input the questions and their corresponding evaluation factors. The evaluation factors were chosen based on a survey where professors were asked for important factors they consider while evaluating an answer. The input required to evaluate the surveyed factors were as follows:

- Ideal answer: teachers inputted an ideal answer to the question that would be compared to the students' answers.
- Number of words in the answer: usually the examiner expects an approximate number of words for each answer.
- Required keywords: central words that the examiner compulsorily expects in the answers.
- Total marks allotted for each question.

The above-mentioned parameters along with the students' answers are then loaded into the NLP model to calculate a total score. The scoring factors of the model are as follows:

- Size of the answer: accounts for 5 % of the total score, and is calculated based on a perfect size x and thresholds for percentages removed from this size.
- Language of the answer: evaluates the answer's language, grammar and sentence structuring, and gives it a score. This accounts for 5 % of the total score.
- Presence of necessary keywords: checks if the answer contains the examiner's listed important keywords, or words with a similar meaning to these. This accounts for 10 % of the total score.
- Similarity index of submitted answer and ideal answer: the examiner's ideal answer is compared to the submitted answer, and a similarity index is calculated. The model used for this is the Siamese Manhattan LSTM algorithm (MaLSTM). The two pieces of text are first converted into individual vectors of numbers through encoding. These vectors are then passed over two LSTM subnetworks. The semantic meaning of the texts are compared in a hidden network and the similarity index is given as output. The similarity index accounts for 80 % of the total score.
- Copying index: finally, an index for how much of the text is copied from other sources is calculated. This has no impact on the total score, but if the value is over a certain threshold, the teacher is notified.

In order to test the performance of the model, the authors created a question sheet and had four students answer the questions. Scores for the answers were then calculated by the authors' new model, as well as with other methods, like cosine similarity, Jaccard's similarity and an RNN. The answers were also manually graded by three professors. The authors' model gave scores that were very similar to those that the professors gave, and the MSE was the second lowest of all the tested models, after Jaccard's similarity.

3.1.2 Using different variants of BERT for grading

Ye and Manoharan [37] developed three automatic graders based on three different language models. These three models all used different versions of BERT. Each task on the exam(s) being tested has a specimen answer that is provided by the instructor. The models generate sentence embeddings representing the meaning of each sentence. A neural network is used to compare two sentences and discern whether they have the same meaning or not. It takes the sentence embeddings of the two sentences as inputs, and outputs a score or a probability representing the level of similarity of the two sentences. There are two ways of judging whether or not two sentences have the same meaning. The first is to use a binary classifier, which outputs a binary value that represents either that the sentences have the same meaning, or that they do not, based on the sentence embeddings. The other approach is to give a rating on how similar the meanings of the two sentences are. In this study, the latter approach was used.

The models differ in which variant of BERT they use. One of the models uses the standard BERT, whereas the two others use RoBERTa and DeBERTa, respectively. RoBERTa uses a dynamic masking pattern instead of BERT's static masking. It removes BERT's next-sentence training objective, which allows it to improve on the masked language modelling objective. It also uses 10 times more training data than BERT. DeBERTa, on the other hand, encodes the content and position information of a token in two separate vectors instead of just one. While it predicts the masked tokens, it takes the absolute positions of the tokens into account.

The models were trained on three different datasets. In order to judge the correlation between the similarity rating predicted by the automatic grader and the ground truth rating given by humans, the PCC is used. The convention is that if the PCC is between 0.5 and 1, the correlation between the predicted rating and the ground truth is strong. The authors tested the models for both the PCC and the MSE. The results showed that in both of these regards, the RoBERTa model did not perform well compared to the two others, and that the DeBERTa model performed slightly better than the standard BERT model. However, all the three models obtained a PCC of over 0.5, indicating that they generate similarity ratings that correlate to a human.

3.1.3 Calculating the robustness of three AES models

Wangkriangkri et al. [38] compared three different Automated Essay Scoring (AES) models, each of them using different text embedding methods for automatically scoring essays. One of the models was based on BERT, while the other two were based on Global Vectors for Word Representation (GloVe) and Embeddings from Language Models (ELMo), respectively. GloVe is a word embedding method which encodes the ratio of co-occurrence probabilities between pairs of words. A limitation of word embedding methods is that a word's representation is the same in all contexts. With context embedding, this limitation is addressed by encoding text on a case-by-case basis and giving a unique representation to every text. ELMo is an example of such a context embedding. It encodes texts using two unidirectional LSTM models, whereas BERT encodes texts using a bidirectional transformer model.

A focus of this study was to test the three models' ability to detect adversarial input. Adversarial input is input that is made to trick the system, and does not fulfill the criteria of the given task(s). Examples of adversarial inputs include well-written paragraphs repeated many times or a set of well-written sentences randomly permuted. The models' ability to detect these inputs is an indicator of their robustness. In order to test this, the authors corrupted every essay in the test set in seven different ways, creating seven altered test sets and keeping the original set. Some of the ways in which the essays were corrupted include removing articles or conjunctions, reversing the sentence order, removing certain sentences and swapping words in the same sentence randomly. The authors then developed a formula for calculating the robustness of a model. Robustness was defined as the fractional difference between the total number of corrupted essays scored higher than their unaltered counterpart and vice versa. The model with the highest average robustness of all essay prompts and altered sets was designated as the most preferred model for the task.

The results showed that the BERT-based model achieved the highest robustness, particularly a variant with frozen weights in the embedding mechanism, although the standard variant came in second. Furthermore, the models had higher robustness on sentence-level alterations than word-level alterations. The highest robustness was achieved when the longest sentence in the essay was removed, which suggests that the longest sentence contributes the most to the essay. This is also suggested by the attention weights for the longer sentences. The ELMo-based model achieved the lowest robustness. The authors attribute this to the model's predictions changing less because the changes in lower-layer features are too subtle. The cosine similarities between the lowest-layer feature sentence vectors for the word-level altered and unaltered sets were significantly lower for the ELMo model than the GloVe model. This suggests that the alterations affected GloVe's embedding more than ELMo's.

The authors did not only test the robustness of the model. They also tested how much the models' predictions agreed with human ratings, using Quadratic Weighted Kappa (QWK) as a metric [39]. All the models achieved state-of-the-art QWK, which is on par with human performance. However, the authors discovered that by freezing the weights in the embedding layer in the GloVe-based and BERT-based model, the QWK decreased, while the robustness increased. The authors attribute this trade-off to the fact that the models' embedding layers were mostly trained on well-written passages, and thus were more sensitive to errors present in the dataset. After fine-tuning, the models got accustomed to poor writing, and were thus less sensitive to errors. They gained a higher QWK, but were less able to differentiate altered essays from non-altered ones.

3.2 Similarity measures

Calculating similarity scores is a fundamental aspect of the thesis for this project, which is encapsulated in RQ 1. The problem of calculating similarities between text exam answers can be generalized to encompass similarity calculations for any type of text. Therefore, it was decided to review the literature for text similarity in general. As will be seen in the descriptions of two articles below, variations of BERT and older models like LSTM and RNN are prevalent in the field.

3.2.1 Long-form document matching with CoLDE

Jha et al. [40] identify three main challenges when finding similarities between longer texts:

- The presence of different contexts for a specific word throughout the document.
- Small sections of text that is contextually similar between two documents, but dissimilar text in the other parts.
- The generalizing nature of a single global similarity measure does not capture the diverseness of the document's content.

To tackle these challenges, they introduce Contrastive Long Document Encoder (CoLDE), a transformer-based framework that addresses these challenges and tries to capture similarity at three different levels. Firstly, they aim for high-level similarity scores between a pair of documents. Secondly, they calculate similarity scores between different sections within and across documents. Finally, they calculate similarity scores between different chunks in the same document and across other documents. CoLDE divides a long document into different sections and uses unique positional embeddings in order to capture the document for additional interpretability. The authors define interpretability as "the weighted similarity scores between different sections and different chunks of text in a document". They use a combination of contrastive loss and a multi-headed attention layer for different text chunks as the means to calculate this weighted similarity score.

CoLDE consists of three main components: data augmentation, data encoding and the contrastive loss function. First the document is divided into two different sections. To overcome BERT's limitation of only 512 tokens per single feed-forward pass, the authors divide each document section into multiple chunks consisting of non-overlapping 512 tokens. These input section chunks are enhanced by unique positional embeddings to encode the long document structure.

For the encoder part, the augmented input chunks consisting of 512 tokens are given to BERT as input. For chunks smaller than 512 tokens, zero-padding is performed. The encoded BERT chunk embeddings are then given to a bidirectional-LSTM (Bi-LSTM) layer for aggregation. This layer uses a unique 'multi-headed chunkwise attention' component. It is a self-attention layer that computes attention between the BERT embeddings of different chunks consisting of 512 tokens, both within and across sections. The chunk that plays the most important part in computing the similarity score is the one with the highest attention weight with regard to a query chunk. The multiheaded chunkwise attention between sections is calculated by treating each chunk in a section as its own query. A Bi-LSTM layer then sequentially receives the encoded BERT output of different chunks, in addition to the multiheaded chunkwise attention. This is done in order to aggregate the segmented chunk representations so that an intermediate high dimensional section representation can be obtained. Furthermore, this intermediate representation is given to a projection layer for reducing the dimensionality, resulting in a final section-level representation. This is used to compute the contrastive loss between sections across query and target documents. Finally, in order to provide document-level similarity scores, similarity scores between different sections, and to ensure that documents belonging to the same class are closer to each other in the latent embedding space, supervised contrastive loss is used.

The authors tested the model on three different datasets and compared the results to several other models, including one that used RNNs, one that used convolutional neural networks and another BERT-based model. They reported precision, recall, f-score and accuracy for each model, and found that CoLDE had the highest f-score and accuracy out of them all. The authors attribute this to their novel mechanism of exploiting long documents' structure in combination with the contrastive learning framework. The authors also found that CoLDE's performance steadily increases when the length of the documents increase. Finally, they performed an ablation study where they tested the performance of the model without certain layers or steps. One version of the model did not include the splitting of documents into sections and without the positional embeddings in the input documents. Instead, the entire long document was split into chunks consisting of 512 tokens and given to the data encoder module. By using the same contrastive loss function, the authors observed a significant drop in the performance for document matching across all three datasets. Also, to overcome BERT's limitation of only 512 tokens at a time, a Bi-LSTM layer is used for aggregation. The authors observed a performance drop when not including this layer.

3.2.2 C2SA for Biomedical Semantic Text Similarity

Li et al. [41] proposed the Cross2Self-attention (C2SA) mechanism, which is composed of self-attention within a single sequence and cross self-attention between sequences. This was integrated with bidirectional RNNs, creating the C2SA-biRNN model. This model was taken as the downstream model of BERT for evaluating semantic text similarity between sentence pairs in biomedical literature. For a given sentence pair, the model calculates a similarity score for these sentences using the softmax activation function on the output of the bi-RNN layer.

A significant challenge in sequence learning is the problem of long-dependency. This is the problem when there is a significant gap between the relevant information of a sentence, and the point where this information is needed. As an example, if an AI is fed the incomplete sentence "I grew up in France, and I speak fluent

", and is tasked with predicting the next word (which should be "French"), the most recent information suggests that the next word should be a language. Yet, in order to narrow this down, the context of the word "France" from further back is needed. Long-dependency is particularly a problem in syntactically complex and long sentences, such as are often found in biomedical literature. This problem is addressed with BERT's self-attention mechanism. However, in order to obtain mutual semantic information between sentences, cross-attention is more appropriate. The authors therefore propose the C2SA mechanism, which combines advantages of both self-attention and cross-attention.

The C2SA-biRNN achieved state-of-the-art performance in terms of PCC. The authors attribute this performance to BERT partly solving the long-dependency problem that exists in biomedical literature, with BERT representing abstract and deep semantic features precisely, and the C2SA mechanism enhancing the semantic representation between sentences.

3.3 Transformers

Transformers are a widely used tool in NLP tasks. The transformer architecture has several advantages over older models such as RNN. One advantage is that it has the potential to understand the relationship between sequential elements that are far from each other. Another one is that the input data does not need to be processed sequentially, meaning that it allows for parallelization. This means that it can process and train more data in less time. An application of transformers that is relevant for this thesis is the detection of similarities between texts and explaining these similarities. One such project is described below.

3.3.1 BTI

In a recent article, Malkiel et al. [42] present BTI, which uses a pre-trained BERT model to infer unlabeled paragraph similarities, and then produces interpretable explanations for the similarity of two textual paragraphs. The first step in the chain of operations is to propagate two paragraphs through a pre-trained BERT model, which yields contextual paragraph representations. Then it calculates a similarity score to measure the affinity between the two paragraphs, with for example cosine similarity. Then the model calculates gradient maps for the first paragraph's embeddings with regard to the similarity to the second paragraph. These gradient maps are then scaled by multiplying them with the corresponding activation maps and summing them across the feature dimensions. This results in a saliency score for each token in the first paragraph. These token saliency scores are then aggregated to words, which produces word scores. Next, the same procedure is performed again, but with the first and second paragraphs reversed. This produces word scores for the second paragraph, calculated with regard to the similarity with the first. Finally, word pairs are matched from both paragraphs and scored by the importance scores corresponding with its elements and the similarity score corresponding with the pair. The algorithm then detects and retrieves the most important word-pairs as explanations. The similarity between paragraph-pairs can thus be interpreted by highlighting and matching important words from every element. The highlighted words in both elements should then decide the semantic similarity of the two paragraphs in a manner that correlates with how a human would perceive it.

In order to test the efficacy of BTI, the authors conducted an experiment where they tested BTI against both several other alternative methods and ablation variants, and compared the results to the results of mean opinion scoring done by five human judges. The same test set, being comprised of 100 samples, was ranked for all variants. Scoring was performed blindly, and the set of samples was shuffled randomly. The sample interpretations were then ranked on a scale from 1 to 5. The results showed that BTI performed better than the other alternatives, implying a better correlation with human perception. The results of the ablation variants indicated the importance of using the gradients on the embedding layer, and emphasized the importance of the multiplication between gradient and activations.

3.4 Discussion of findings

A set of goals was presented in the introduction of this chapter, namely to research both how AI models hitherto have been used in the grading process, and the various different technologies that could potentially can be applied in this thesis. These technologies could be relevant for either of the two research questions introduced in section 1.2, meaning either the calculation of similarities or the extraction of features for assisting the examiner. The main findings are summarized and discussed below.

The studies performed in [41, 38, 37, 42, 40] all use BERT, either for automatic grading, or for finding similarities between texts. In all cases except for [37], the BERT-based model outperformed the other models that were tested. In the case of [37], three different variants of BERT were tested, and DeBERTa, which encodes the content and position information of a token in two separate vectors instead of just one, performed slightly better than the standard variant. A reason why BERT consistently performs so well might be because it takes the context of words into account. The output for each input token is influenced by all the input tokens, and each token impacts the outputs of all other tokens. As mentioned in section 2.5, BERT is also bidirectional; it looks for context both left-to-right and right-to-left. BERT's pre-trained representations reduce the need for many heavily-engineered task-specific architectures, and it manages to even outperform many such architectures [3], something that is reflected in several of the articles referenced above.

Three of the cited articles ([36, 37, 38]) concern themselves with automatic grading of text answers, which is not exactly what this thesis is concerned with. Yet, they still provide for some useful takeaways. Bahel and Thomas [36] calculated a similarity index between the examiner's provided ideal answer and the students' answers with the Siamese Manhattan LSTM algorithm. This is not entirely unlike what this thesis is concerned with; instead of calculating the similarity between an ideal answer and a student's answer, the tool made for this thesis compares students' answers directly. The scores given by Bahel and Thomas' model (largely based on the similarity index) were very similar to what was given by teachers manually grading the tasks. However, LSTM-based models have certain issues that are largely solved by BERT-based models; words are passed in sequentially, making the learning process slow, and its lack of bidirectionality makes it suboptimal to capture the true meaning of words. It can also be noted that the MaLSTM model only achieved the second lowest MSE out of all the four models that were compared in the study, scoring slightly lower than Jaccard's similarity, which is a very simple model.

The CoLDE model (which is based on BERT) by Jha et al. [40] managed to tackle several different challenges associated with finding similarities between longer texts by calculating similarities at section and chunk level, in addition to document level. They also found that the performance of this model increased with the document length, which proved to be useful for this project, as the tool developed for the thesis is aimed at long exam answers. A unique method for handling answers that exceeded BERT's limit of 512 tokens was developed for this tool, which will be explained in subsection 4.1.2. The C2SA-biRNN model proposed by Li et al. [41] enhanced BERT's self-attention mechanism by combining it with cross-attention, which is more appropriate for obtaining mutual semantic information between sentences. This could have been useful for a version of the tool developed for this thesis that is more specialized towards similarity between sentences, since this thesis is more focused on document-level similarity than sentence-level similarity.

Malkiel et al.'s BTI model used a pre-trained BERT model to infer unlabeled paragraph similarities and retrieved the most important word-pairs from each paragraph as explanations [42]. The highlighted words should dictate the semantic similarity between the paragraphs in a way that humans can understand. Something akin to this ended up being implemented in the tool produced for this thesis, where TF-IDF terms that two exam answers have in common are displayed in order to show the examiner exactly what the similarities between two answers are. This will be explained in detail in section 4.3.

CHAPTER

FOUR

METHODS

This thesis proposes a set of methods for performing quality assurance for the grading process of exams. The methods should help point out mistakes and biases that can occur in the grading process. Examiners are humans, and humans make mistakes. As automatic grading is not fully acquirable yet, the proposed methods need a collaboration between humans and AI. They both have to assist each other in order to reach the most fair final grades.

To accomplish this collaboration, a tool needed to be developed which implemented the proposed methods for supporting the grading process. The tool must be built to uphold the following criteria:

- It must be able to read and format the submissions to an exam in a large dataset
- It must inhabit a comparison algorithm to compare the answers' contents as accurately as possible
- It must contain methods for extracting features from the texts that can be used for examining similarities and differences
- It must be easy to use and display the results to the examiner in a well-structured manner

The proposed tool, TGRT, is meant to be used after the initial grading process is finished and the answers have received their grades. Norwegian University of Science and Technology (NTNU)'s online exam coordinator Inspera has implemented features for giving each answer a graded score. TGRT depends on this prerequisite. If an answer's content is to be compared to the others, it needs a graded score. There cannot be recommendations for a change in grade if the answer has received no grade prior. TGRT will then process the dataset and output recommendations for the answers it decides should have their grades changed. Each recommendation includes reasoning for why the answer should be regraded, by listing every similar answer within a given threshold of similarity score. The list of similar answers will convey a tendency of being graded either higher or lower than the compared answer, indicating which way its grade should be adjusted. Every similar answer displays supportive comparative features that attempt to explain the similarity for the examiner.

4.1 Comparison algorithm

The basis for the project was to compare students' answers with one another. To do this, a comparison method was to be implemented. Research in the topic of measuring text similarity shows an increased utilization of the BERT models after its release in 2018 [17] as analyzed in the related work section (chapter 3). Further reasoning as to why it was ultimately chosen as the base for the comparison algorithm for this thesis is explained in section 3.4.

4.1.1 BERT models

BERT has been used in several projects relating to text similarity, as evident by the related works. It was however unclear if it worked well enough with the Norwegian language to be considered optimal for this project. After some research, several models that were trained on large corpora in Norwegian were discovered. These pre-trained models are described in detail in section 2.5. The models implemented and tested in this thesis were NorBERT version 1 and 2, NB-BERT-Base and NB-SBERT. NB-BERT-Large were initially also in this list, but it was decided that it would be unfit for the product. The reason for this is discussed at length in subsection 4.1.3. In order to find out how well each of the models fit the purpose of this thesis, they all needed to be tested on appropriate datasets.

Herein lies a significant problem encountered during this project. There was no definitive way of measuring the accuracy of the output from BERT. The text answers were often several hundred words long and included technical terms the researches did not deem themselves capable of fully understanding. It would have needed an unreasonably long time to thread through all answers and create scores between them that measured their similarity. Towards the end, the compared answers' similarity would be tested by letting examiners analyze the results during user tests, but by that time the product needed a comparison method that could reasonably well measure the difference between texts. One of the solutions to this problem was to create custom datasets which were constructed with knowledge in advance of what answers would be similar. This way it was known in advance what answers the algorithm should deem similar. The constructed datasets are discussed further in section 4.2.

4.1.2 Algorithm setup

The setup of the comparison algorithm was structured to easily change what pre-trained BERT model would be used. The main script for testing the models runs all the different pre-trained models considered for this project on a specified dataset. In the final product, it would have taken far too long to compute outputs for every model, so the testing would determine which model would give the best results. This needed to be decided before user tests could be performed.

The script starts by importing the tokenizer and model from the specified pre-trained model in the transformers' library. It processes all the answers to one question at a time by tokenizing them and adding them to input IDs and attention mask arrays. The answers are then formatted into a single tensor and passed on to the model. The resulting output is a list of embeddings that represent each answer in number form. Cosine similarity is then calculated between every item in the embeddings list. The resulting matrix contains a score that notes how similar two answers are to each other.

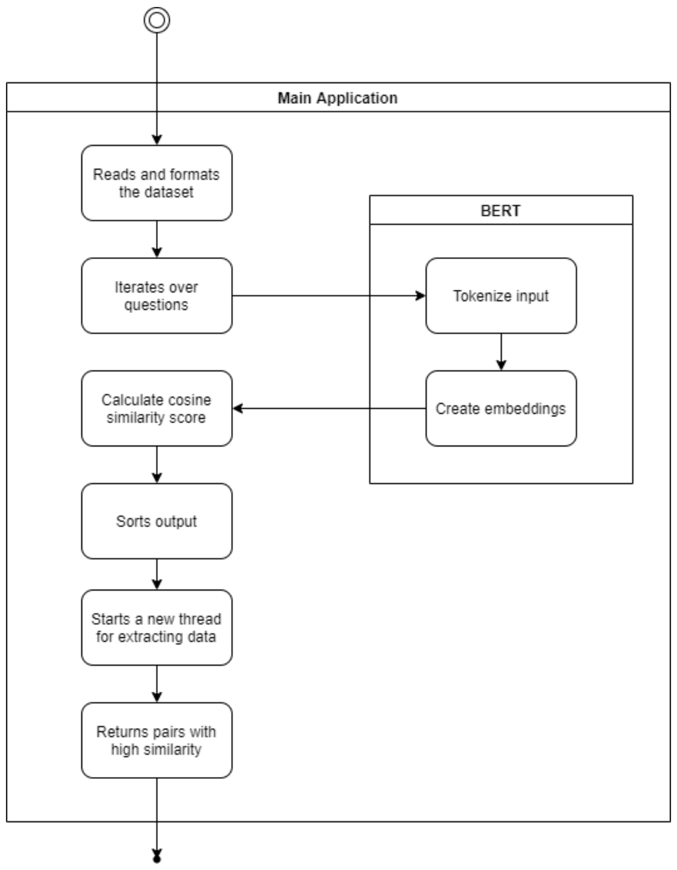

Main Application
Reads and formats the dataset
Iterates over questions
Calculate cosine similarity score
Sorts output
Starts a new thread for extracting data
Returns pairs with high similarity
BERT
Tokenize input
Create embeddings

Figure 4.1: How BERT is integrated into the application.

One problem with the setup was that BERT only accepted a length of 512 tokens as input. In this case a token is a word. If any of the answers were longer than that, the model would only process the first 512 tokens and ignore the rest. As this would be the case quite often with long text answers, this length limit would not hold for purposes of this thesis. The solution to this problem was to split the text into chunks of a maximum of 512 tokens, similarly to how Jha et al. [40] describe their experience with the same problem. The chunks are then sent to the comparison algorithm and calculated into embeddings. The outputted embeddings are added together and averaged. This might "flatten" the embeddings in some cases and prevent them from swaying too far in one direction or the other, but at least it should include all the parts of the text in the embedding.

4.1.3 Larger models

BERT has a maximum input of 512 tokens. The more tokens, the longer the algorithm takes to calculate the results. During testing, the calculations were often so heavy that the computer got an out of memory error. To mitigate this, it was proposed to connect to NTNU's server and host the back end of the code there. This resulted in fewer out of memory errors and shorter processing time. The processing time was still too long, however, which is why it was settled to limit the maximum amount of tokens for input to BERT to 128. This means that the parts of the text after the initial 128 words would be cut and not sent as input, but because of the method of inputting longer texts described in subsection 4.1.2 this would not be a problem. The answers would be split into chunks of 128 tokens and passed in separately to BERT, and combined again after being processed into embeddings.

The elongated processing time was first encountered when testing NB-BERT-Large. As explained in subsection 2.5.6, the large version is much bigger with more attention heads, encoder layers, hidden layers and parameters. This should have resulted in better comparisons of the texts, which might still be the case. Early on, NB-BERT-Large was included as a potential candidate for the comparison algorithm. It was immediately apparent however that this model used a lot more processing power, which resulted in the program crashing again due to out of memory errors. The proposed solution of splitting the input to BERT into smaller chunks worked fine with the smaller custom datasets first used. It ran without errors, but it was still very slow and took three times as long as the other models.

The out of memory errors emerged again when testing on a dataset with dozens of participants. This was also a reason for hosting the program on NTNU's local servers in the first place. Processing time on the server was a lot quicker when running the other models, and it even ran NB-BERT-Large. The first run of the large model started in the middle of the day, and kept running for several hours after the work day was complete. The next morning, the results had arrived. The results were on par with the other models, but it does not really matter how good they were. A processing time of several hours is unacceptable for a tool like this. It is a balancing act to make the tool as adept as possible at comparing the answers, while still running in a reasonable time. If the final product uses half a day to process, it can not be expected to be used professionally. Because of this, it was decided not to perform any further testing on NB-BERT-Large in this thesis.

4.2 Datasets

Privacy is a big concern emerging when using students' answers in research. Several departments, faculties and examiners at NTNU were approached to inquire for data to use in this research. Text answers were needed in order to test the pre-trained models, ideally with their scores included. Many were hesitant to reply, and only one person reached out to support the research with a couple of very small datasets with about fifteen submissions each. In this case, the students had given their permission for the answers to be used in research. This was of course the best way to collect data, but in most exams this is not asked of the students. The two datasets collected were exams in a course about religion, and were great for testing the tool as they contained longer text answers. They did however not contain the score or grade each answer received, which made analyzing the results harder. In any case, the answers helped with developing a comparison algorithm early on by having texts to compare and analyze. Even if the scores were unknown, the contents of the top results could be manually analyzed for similarity.

Later on, the supervisor for this thesis provided a dataset of exam answers from a web development course. It consisted of three substantially larger exam sets with up to several hundred answers. The big advantage of these answers was that they all had their scores attached. This meant that the program could be tested on exams with many participants, while also having access to a number based scoring system that could confirm if the results were somewhat accurate. In a sense, the graded scores would not help determine if the answers were similar or not as the tool was supposed to find mistakes in grades, but it would serve as a way of verifying the results from the comparison algorithm. If two answers with a high outputted similarity score from the algorithm had the same grade, this would to a certain extent verify the results. The downside to this dataset was that the subject matter was software engineering, which meant that many technical terms and characters would be included in the answers. Programming languages with special characters and words borrowed from English might not be the best test material to begin with. As the BERT models for this project were trained on large corpora of Norwegian words, they most surely would not understand the technical meanings behind many of the terms used. This made the testing less robust, but at least some of the questions contained text answers where the meaning could be understood and derived from context.

4.2.1 Format

The web development exam dataset was also in a format that is standard for exam answers exported from Inspera. Each exam set was contained in one JSON file with a logically built up structure of all the students' answers and scores. The code that reads and formats the datasets for further processing would be based on this structure. The structure is the same across all subjects, courses and exams in NTNU. The value corresponding to the keyword "ext_inspera_manualScore" would for instance always be the score an answer received. Another positive was that the file was easy to export from Inspera's database, and a corresponding file for a given exam would therefore be available to the examiner.

The dataset reader was based on the standard format from Inspera, but there were many prerequisites that needed to be met before the exams could be exported with this structure. The exam needed to be held via Inspera and the answers must have been written directly into it and not by means of handwritten texts or other files submitted as attachments. To account for other file formats would be very difficult, as the setup for each exam is often wildly different. The two other exam datasets collected during testing were in PDF form, and to successfully extract the raw text, it needed a specific formatting tool that split the full text on prompts that were unique to that exam. This would not be feasible for a final product that would be compatible with any type of exam.

Because the comparison method was based on a Norwegian model, the answers also needed to be in Norwegian. All these prerequisites became a problem when contacting test subjects. Many examiners had some of the prerequisites fulfilled, but few had all. Many courses are taught in English at NTNU, which means most of the students answer the exam in English as well. Calculations for math questions and figures were also not possible for the reader to understand, so applicants that had datasets where calculations were needed to understand the full extent of the answer were also turned down.

The number of fitting candidates that matched all prerequisites ended up being relatively small. In order to expand the amount of applicants, compatibility for other types of questions were implemented. This included questions that are often present in digital exams, like multiple choice questions. TGRT is not able to read multiple choice questions by design, as that would not be very interesting to compare. The purpose of multiple choice questions is to easily and automatically be able to grade them, and Inspera already has a separate automatic grader for these types of questions. It was therefore decided to filter out multiple choice questions when importing the datasets. This would be done by only including answers where the key "ext_inspera_manualScores" was present (since these are manually graded and multiple choice questions are automatically graded).

As will be further expanded on in subsection 5.2.2, there was a problem with this approach. There would sometimes be answers with the "ext_inspera_manualScores" key that were actually multiple choice, since the answer text started with "simpleChoice" followed by a string of numbers. In order to exclude these answers, other conditional checks had to be added. The conditions for considering a question to be valid were decided to be that the question has a list of answers larger than zero, and the answers do not start with "simpleChoice".

A problem that occurred when processing the JSON files before giving them to BERT as input was that there were many special characters that had to be accounted for. Characters from the Nordic languages, such as "æ", "ø" and "å", appeared in the JSON files as their HTML entity codes, namely "&#230;", "&#248;", and "&#229;", respectively. The same was the case with a wide range of other special characters, such as "&", """ (quotation marks) and "=". Additionally, many HTML tags were present, corresponding to the structure of the answers. These included for example "<br />" (for line breaks) and "<p>" (at the start of paragraphs). Including these HTML tags in the input to BERT would only confuse it, so they would have to be removed before this step. To solve the problems associated with these special characters, a step was introduced at the end of the processing where every answer is iterated through. In this step, the HTML entity codes and tags were replaced with their appropriate characters (or in some cases, white spaces or empty strings).

4.2.2 Custom datasets

There were problems with all the collected datasets so far, which made it difficult to test all the aspects needed. Two custom datasets were made in order to perform the tests the collected datasets were not compatible with. TGRT compares answers for one question at a time and makes no connections between questions, so the constructed datasets only needed one question with several answers to it. The question for the first dataset ended up being: "Explain the two main types of friction (static and kinetic)". Some answers could then only focus on the static friction, and others only on the kinetic friction.

The dataset was structured into four main sub categories: "Good", "Static", "Kinetic" and "Irrelevant". "Good" was the label for the answers that were considered well written and would get a higher graded score. "Static" and "Kinetic" labels specified answers that focused on only one part of the answer, that being only static or kinetic friction. "Irrelevant" specified answers with no correlation to the question and consisted of texts that answered what "fiction" and "faction" was. All the categories (except for irrelevant) had variants in them labeled "synonyms", "typos", "long" and "short" along with the "normal" version. This enabled for instance testing for how BERT would handle typos, different lengths of answers and missing core concepts present in some answers and not in others. The synonym answers had many key words of the text replaced by words with the same meaning, the typo answers had many key words spelled incorrectly, the long answers had longer texts containing sections with unnecessary examples and extra details not needed to answer the question, and the short answers were a bit too concise to contain all the key information. The "Good, normal" category has more entries than others to have several hits that should get a high similarity score when comparing to another answer of the same category.

The other custom dataset consisted of descriptions of Quentin Tarantino movies rather than answers to a question. From a pool of four movies to choose from (Inglourious Basterds [43], Django Unchained [44], Pulp Fiction [45] and Reservoir Dogs [46]), every "answer" contained a combination of two of the descriptions. To further distance the vocabulary and style of writing from answer to answer, a couple of descriptions were written by fellow students that volunteered. In addition, a few of the descriptions were generated by ChatGPT, and some were summaries found on the internet. The resulting dataset contained eighteen entries where each configuration of a movie pair had three variants. This means that every movie is present in half the entries, but a specific combination of two movies is only present in three of eighteen entries.

Many entries in the second dataset had the movie title in the description, which could potentially make it a lot easier for the comparison algorithm to distinguish between them. This led to the creation of a third dataset which is the same as the second, but with the titles removed and substituted by "The first movie..." for instance.

The Tarantino dataset had another variant with each of the descriptions split into two entries. This way, each of the four movies had nine different descriptions. The dataset was created with the dual description entries, but by splitting the descriptions into separate entries they could be used for testing the ability of the models to compare pairs of text where each text had only one theme. This dataset used the descriptions without the movie titles included.

4.3 Supporting comparative features

The supporting comparative features are important in communicating the similarities between answers in other ways than just a numerical similarity score. The background sections of this thesis have examined several features that might work well for TGRT. In cases where answers are relatively short, it might be enough to only show the raw text for each answer. For longer texts, however, there should be some representative properties that give an overview of the longer text without having to read the entire answer. This thesis suggests a set of supporting features in order to achieve this. The selected features are two extractive features (TF-IDF and RAKE) and three generative features from the OpenAI API. The features can be helpful on their own, but additional functions using the features have also been implemented to assist in comparing the answers.

4.3.1 TF-IDF

TF-IDF was the first supporting feature to be implemented. As described in subsection 2.3.1, TF-IDF is intended to reflect how important a word is to a document in a corpus. It does not explicitly explain to the user why two answers are deemed to be similar. TF-IDF terms from two similar answers does not in itself convey how those answers are similar, but they could with some additional calculations. Instead of showing the TF-IDF terms from the two answers separately, TGRT shows the terms that appeared in both answers along with their TF-IDF scores.

When the similarity scores for a question have been computed on the server, the results are sent to the client while a new thread starts where the extractive supporting features are calculated. The algorithm starts by loading in the stop words list. The list has been generated by adding words manually. Some words were included on the basis of common sense, some through testing, and some by the recommendations made by the model implemented. After running, the TfidfVectorizer suggests terms that maybe should be included in the stop words list. The stop words are words to not take into account when calculating the most unique terms for the text. They are removed mostly for optimization purposes, but also because they can be given a high TF-IDF score by mistake.

The stop words are inputted to the TfidfVectorizer from the sklearn library along with a function for tokenizing the text answers. This will split each text into a separate token for each term. This process removes all special characters like punctuation and commas and creates a list of all the terms. The function also includes a lemmatization function from the spaCy library. A model called "nb_core_news_lg" [47] is imported and loaded that compacts the terms into their lemmatized form, as explained in section 2.2. This model is the largest of the spaCy models trained on the Norwegian language.

The vectorizer can enable TF-IDF scores for either single or multiple terms. It was originally set to one and two n-grams, meaning it would calculate scores for single terms and double terms, but was for optimization purposes ultimately changed to only work with single terms. The process of computing the extractive supportive features is the most time-consuming, so the decisions leading to most optimization were most often chosen. The results from the vectorizer is a matrix with TF-IDF scores for each term in each answer or document. This matrix does not contain the actual term, but only a numeric representation. The term in word form is then extracted and added to the matrix. This is the basis for the TF-IDF matrix.

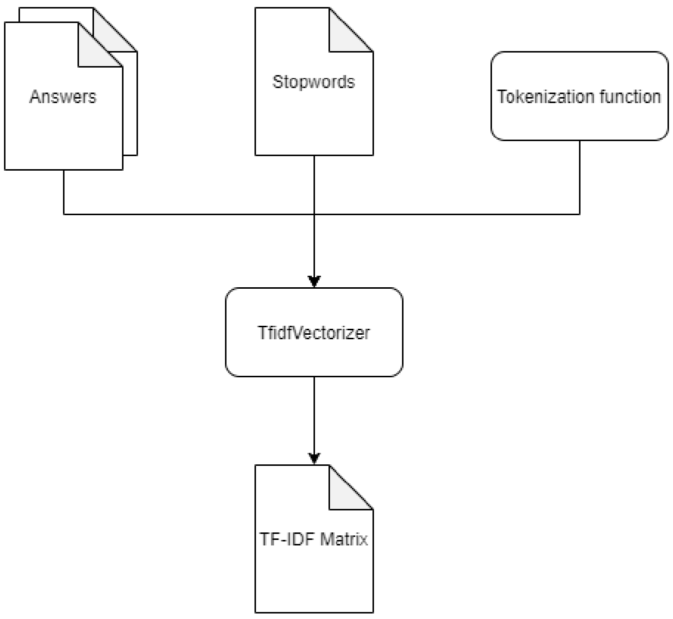
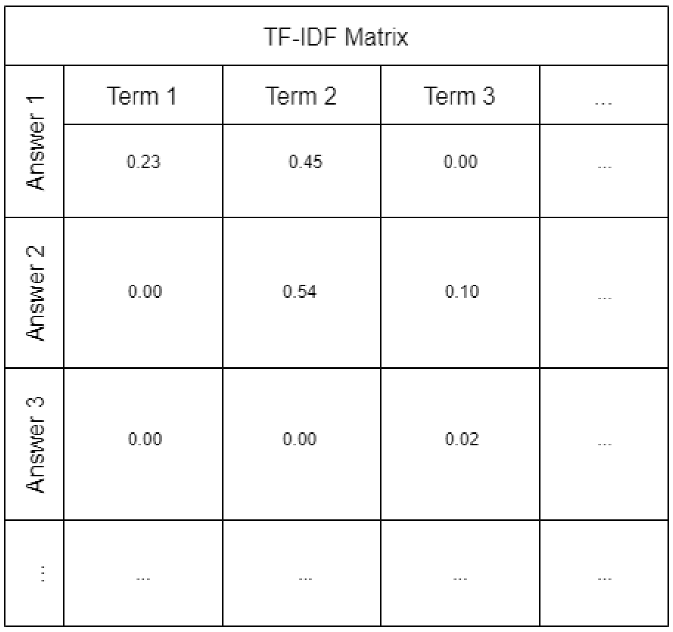


Figure 4.2: Setup of TF-IDF vectorizer.



TF-IDF Matrix

	Term 1	Term 2	Term 3	...
Answer 1	0.23	0.45	0.00	...
Answer 2	0.00	0.54	0.10	...
Answer 3	0.00	0.00	0.02	...
⋮	...	...	...	...

Figure 4.3: The structure of the resulting TF-IDF matrix, where each element in the list contains the TF-IDF score for a term defined by column in an answer defined by row.

The matrix is then used in collaboration with the similarity scores calculated by the similarity algorithm. For each pair of answers, the matrix is used as a lookup to see which terms are present in both text answers. If a term is used in an answer, the TF-IDF score in the corresponding element in the matrix is larger than zero. Additionally, since people use different words to describe the same concepts, a system to look up synonyms was implemented. If a term is present in one text, but not the other, a lookup is performed in a separate file containing many Norwegian synonyms. If the term has at least one synonym present in the list, all the synonyms gets matched against the terms in the other text. This way, words like "forsiktig" and "beskjeden" will match each other and get flagged as the same term. The words may not always match exactly with the context and may carry a different meaning, which is why TGRT displays the terms separated by a slash instead of just the original term when this occurs.

If a term is present in both answers, a comparative score for this term is calculated, which denotes how significant this term is overall across both answers. The term score for a term that is present in both answer a and b is calculated as follows:

$$\text{Term score}(a,b) = \frac{(\text{TF-IDF}_a + \text{TF-IDF}_b)}{2} \cdot (1 - |\text{TF-IDF}_a - \text{TF-IDF}_b|) \quad (4.1)$$

The average TF-IDF score of the term in both answers is calculated. Doing only this would result in terms that could represent the first answer very well and the second answer less, getting ranked higher than it should. The average score is therefore multiplied by the difference in TF-IDF score subtracted from 1. Since the TF-IDF score is always between 0.0 and 1.0, this multiplication factor is also between 0.0 and 1.0. This way matches where the TF-IDF scores are very different are punished and matches where the scores are closer are rewarded. The matched terms and their corresponding comparative scores are then sorted by the latter and sent to the client. Thus the TF-IDF terms that appear in both answers are displayed in order of relative significance.

4.3.2 RAKE

In addition to showing why two answers are considered similar, TGRT should also show how two answers are distinct, so that the user can see if one answer is markedly better than the other. This would be done by extracting keywords from each answer in a similar fashion to TF-IDF. By seeing keywords from each answer side by side, the user might have an easier time investigating differences in quality between the two answers. As TF-IDF in this implementation only uses single terms as input, it was decided to use RAKE as the keyword extraction method. RAKE processes the text and finds the most unique and descriptive parts of the sentences instead of just single words. This was implemented by cloning a GitHub repository [48], which was based on the original RAKE algorithm that was proposed by Rose et al. [13]. Only the rake.py file was used, which contains a class with all the functions necessary to extract candidate keywords. The code had to be edited slightly to fit into the pipeline, but was mostly untouched.

In order to instantiate an object of the RAKE class, the path to a file containing stop words is needed. The inputted path leads to the same stop word file as the one used for the TF-IDF calculations. When an object of the RAKE class is instantiated, a regex pattern is built that contains every stop word in the specified file. The actual extraction of candidate keywords is done in the class method run(). This method receives the text as input, and splits the text into phrases using punctuation characters as the delimiter. Candidate keywords are then generated by splitting the phrases when stop words are found, and adding the words between them to a list. Then the word scores, and finally the keyword candidate scores, are calculated as described in subsection 2.3.2. The list of candidate keywords is then sorted by score and returned. An instance of the RAKE class is initialized right after the TF-IDF matrix is created. The run() method is executed, and the resulting list of candidate keywords is sliced so that it only includes the 25 uppermost items, so as not to overwhelm the user with many irrelevant phrases.

As explained in subsection 2.3.2, the phrases it chooses are consecutive series of words that are found between stop words and/or punctuation. Therefore, the user can see at least part of the context in which the phrase it is used. In some instances, both the text before and after a stop word in a sentence is crucial for understanding the meaning of that sentence. Therefore, the resulting phrase deemed as important by RAKE is not always understandable when viewed in isolation. However, whether this is an issue or not depends on the sentence. As will be expanded upon more in subsection 4.5.1, the user can click on any phrase in the list in order to highlight the occurrences of that phrase in the chosen answer, which makes it easy for the user to understand the context. The phrases that are picked are ranked by their scores, whereby a higher score implies higher importance.

The implementation of TF-IDF does not extract multiple consecutive words, meaning that RAKE is better at explaining context for the terms. Precisely because TF-IDF only extracts single words, it has an advantage in that it is possible to use it to find common keywords between two texts. This is a lot harder with RAKE, since it is highly unlikely that two different texts have the same sequence of multiple words. However, the meaning behind different sequences could still be similar and therefore also display similarities between the answers.

4.3.3 Summarizers

One of the most important supporting features were always a summarizer. This would let the user quickly read through a summary of each answer in order to get a good overview, and enable them to distinguish between the two answers. This is especially useful for longer text answers, which are the primary target exams for this thesis. One problem with using summarizers as a guide for judging exam answers is that they naturally leave out many details, and the details might be what separates a great answer from a good answer. However, as long as the user utilizes the summarizer as a supplement to the other functionalities, and does not rely on it alone in order to decide whether two answers deserve different grades or not, it can be of good use.

As explained in section 2.7, there are two main types of summarizers: extractive and abstractive. Extractive summaries consist of sentences that are already present in the original text, whereas abstractive summaries consist of newly generated sentences that are supposed to convey the meaning of the original text as succinctly as possible. Extractive summarizers were the easiest to implement, as there were several ready-made implementations to choose from. Several of them were tested in the development phase, including the summarizers created by Natural Language Toolkit (nltk) [49] and Summa [50], which are both based on the TextRank algorithm [51]. A problem with both of those summarizers is that the sentences they pick from the original text are not ordered according to when they appear in that text, leading to disjointed summaries where a sentence from the end of the original text appears in the beginning of the summary without any context. The reason this happens is that the TextRank algorithm only picks the sentences it deems to be most important, and constructs the summary by concatenating these sentences in order of importance.

A more general problem with extractive summarizers is that even if the sentences are in order, they might leave out crucial information that is not deemed important by the algorithm, that would otherwise have been retained if an abstractive summarizer had been used instead. This is because abstractive summarizers are based on deep learning, and thus gain a deeper understanding of the context of each sentence. They can therefore summarize the content of three sentences in one new sentence, rather than either including or excluding each of the three sentences, like extractive summarizers do. Abstractive summarizers by nature of summaries in general also leave out information, but can to a bigger degree insert information into one concise sentence than extractive summarizers. Abstractive summarizers are also trained to mimic how humans might summarize a text, and thus produce summaries that are more readable by humans. Because of these advantages, it was decided to implement an abstractive summarizer rather than an extractive one, which lead to exploration of OpenAI's API.

4.3.4 OpenAI

OpenAI is an American AI company that was founded in 2015 [52]. They conduct research on AI with the declared intention of ensuring that Artificial General Intelligence (AGI) benefits all of humanity [53]. AGI refers here to a hypothetical intelligent agent that can learn any intellectual task that human beings can perform. OpenAI has developed a wide range of products based on reinforcement learning and deep learning models, including video game bots and speech recognition tools. The product that OpenAI is most known for, however, is ChatGPT [54], which is a chatbot that was released in late 2022. ChatGPT garnered widespread attention in the media for its ability to provide detailed and articulate responses to questions spanning a wide array of topics. The researchers tested ChatGPT outside the confines of this project, and came up with the idea of using it to generate summaries. In order to integrate an OpenAI model with TGRT however, it was necessary to connect to an API.

OpenAI hosts an API that can be implemented in applications for a small fee for traffic. The API will generate answers to prompts sent in. This was utilized in order to generate summaries for the exam answers. After sending prompts that asked it to summarize certain texts, it generated short abstractive summaries that captured the meaning of the texts better than any of the summarizers previously tested. The API has several models to choose from that can perform the calculations needed. The model that was decided on to use for the requests was text-davinci-003 [55], because of its ability to perform any language task with better quality, longer output, and more consistent instruction-following than most of the other OpenAI models.

In addition to generating summaries, themes and keywords were also generated from the answers. The themes prompt proved to be useful, since the model generated new words for encompassing the main themes, which could help to get an overview of the content of an answer. The keyword prompt produced results in the same vein as TF-IDF and RAKE by selecting words that appeared in the original texts. These were not ordered in a particular way. It was decided that all the three types of requests (summary, themes and keywords) would be used in the TGRT.

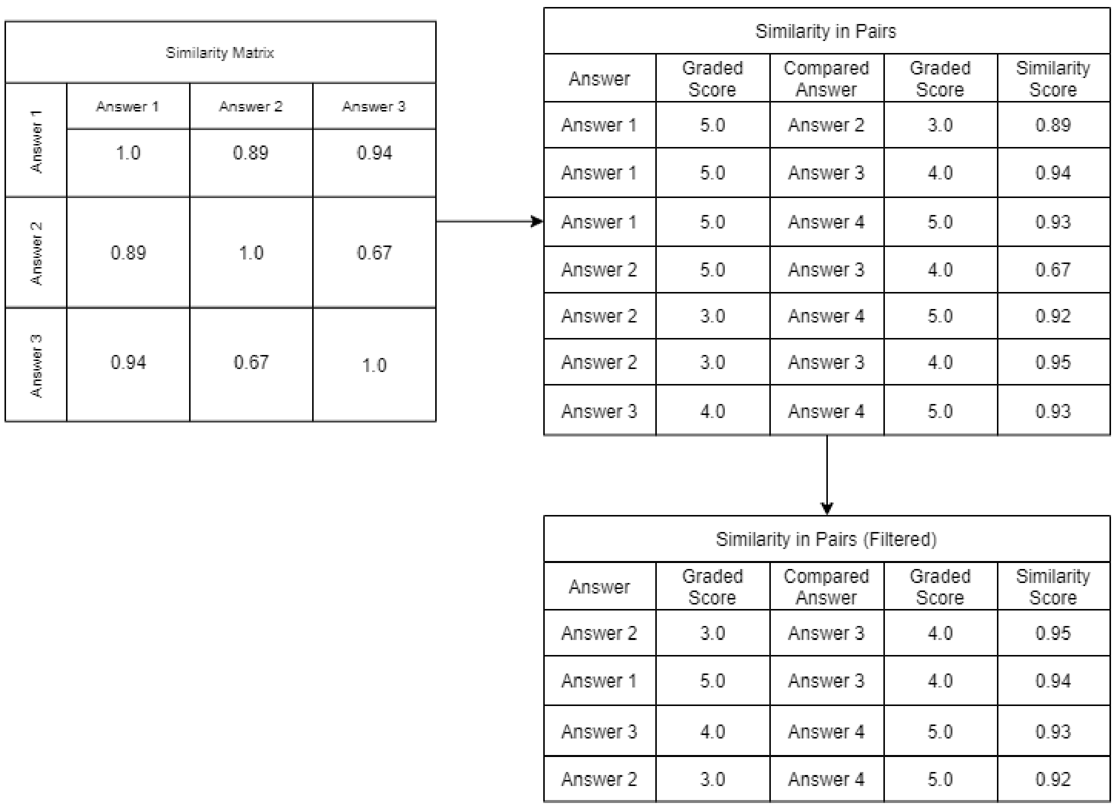
The calls to OpenAI were first implemented on the server, but were soon moved to run directly from the client. The API only accepts a few requests before it charges money for every call after that. The initial plan was to run calls to the API to generate summaries, themes and keywords for all the answers, but this would go beyond the limit of free requests very quickly. As some answers further down the list of comparisons would probably never be examined, it would be unnecessary to generate the supporting features for all answers. By moving the API calls to the client and only calling them when the user selects an answer to examine, the supporting features are generated only when the user needs them. The results are then saved client side, to prevent the requests from running multiple times if the user jumps from answer to answer in the examination process. It would have been possible to send the requests from the server, but that would have been an unnecessary extra step and resulted in an increase in latency.

When the user has selected a recommendation along with a compared answer, the requests are sent. First the themes are generated, then the summary and lastly the keywords. This was done in succession to not overwhelm the OpenAI API as there is a limit to how fast they accept calls. The requests are built by encasing the answer in quotation marks and concatenating it to a prompt that asks for the AI to generate the supporting features. The prompts are written in Norwegian and is the only thing telling the API what to return. There are no settings to let it know what is wanted, which means that it needs to analyze and understand that the inputted text answer is in Norwegian and return a text written in the same language. There is no guarantee it will always understand that, but there were no instances during testing where it misunderstood the prompt.

Because of the generative nature of the OpenAI API, the resulting features do not have a defined format. This means for instance it could return keywords in the form of a bullet point list, a numbered list or only listed as a sentence separated by commas. The most common result received was a numbered list. The features were inserted into a hash map that kept track of which answer and question they corresponded to. The results are then displayed in a table to the user.

4.4 Sorting algorithm

An important aspect of the product is how the pairs of similar answers are sorted. At first, the idea was that all pairs would be sorted by a combination of how similar the answers are, and how big the discrepancy in scores is, so that the biggest purported "mistakes" in the examination rank the highest. Upon further consideration, it was decided that rather than just displaying a list of pairs of answers, the answers could be sorted so that each of them would have a list of other answers that are considered to be similar to it. Thus, the user can focus on one answer at a time, rather than skipping around between different answer pairs and risking forgetting their thoughts on a particular answer before encountering it again. By comparing the answers that are similar to a certain answer, it would also be possible for the system to give a recommendation as to whether that answer should be scored higher or lower than it currently is. This also made it possible to sort the recommendations in a way that displayed the recommendations that were most certain at the top, and the recommendations with less certainty further down. Thus, the user would be able to start with the most severe alleged mistakes in the grading before moving on to the less severe instances.



Similarity Matrix

	Answer 1	Answer 2	Answer 3
Answer 1	1.0	0.89	0.94
Answer 2	0.89	1.0	0.67
Answer 3	0.94	0.67	1.0

Similarity in Pairs

Answer	Graded Score	Compared Answer	Graded Score	Similarity Score
Answer 1	5.0	Answer 2	3.0	0.89
Answer 1	5.0	Answer 3	4.0	0.94
Answer 1	5.0	Answer 4	5.0	0.93
Answer 2	5.0	Answer 3	4.0	0.67
Answer 2	3.0	Answer 4	5.0	0.92
Answer 2	3.0	Answer 3	4.0	0.95
Answer 3	4.0	Answer 4	5.0	0.93

Similarity in Pairs (Filtered)

Answer	Graded Score	Compared Answer	Graded Score	Similarity Score
Answer 2	3.0	Answer 3	4.0	0.95
Answer 1	5.0	Answer 3	4.0	0.94
Answer 3	4.0	Answer 4	5.0	0.93
Answer 2	3.0	Answer 4	5.0	0.92

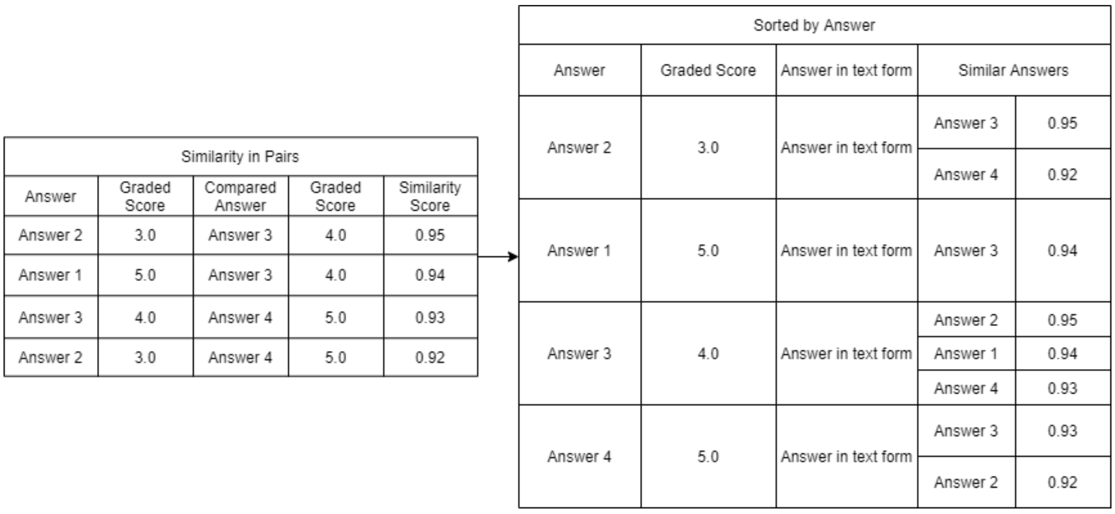
Figure 4.4: The process of sorting the initial similarity matrix received from BERT into a pairwise list of similar answers. The list is then filtered by the specified criteria.

The sorting process starts right after the calculations for the similarity scores are completed. First, the pairs of answers are all sorted by the similarity scores in descending order. Then all the pairs in the list are iterated through in order to filter out all pairs that do not fit the criteria for inclusion. These criteria are that the answers must have received different grades (if not reexamining them would be pointless), and that the similarity score between them must be above a certain threshold (so that answers that are not sufficiently similar are excluded).

For each pair of answers, the difference between their graded score is calculated and divided by the maximum possible graded score for normalization purposes. The algorithm makes sure the difference is bigger than zero, as if it is equal to zero, the two answers received the same graded score and is therefore not necessary to examine. The server also only passes on the answer pairs where the similarity score is above a certain threshold. Answers to a specific question often contain a lot of the same information and can be quite similar at a base level. The algorithm should only keep the answer pairs which contents are significantly more similar than only the base level of similarity that is shared across most answers. Because of this, and based on the numbers emerging during testing, the threshold was set to 0.92.

If these conditions are fulfilled, the pair is kept in the list along with the corresponding similarity score and the difference in graded score variable. Duplicate entries are removed in order to prevent the pairs appearing in the list twice, but in switched positions. This list is then sorted by similarity score and sent from the server to the client (more on this in section 4.5).

After the client has received the list described above, the process of sorting the pairs so that each answer has its own list of comparisons begins. A dictionary for storing the data is created, where the key is the index of the answer and the value is an object containing the text answer itself, the score it was given by the examiner and a list of comparisons with similar answers. Each entry in this list is an object containing the index of the comparison answer and the similarity score between the two answers. The algorithm for structuring this data starts by iterating through the list of similarity pairs from the server. Each answer is added to the dictionary if it is not already present there. Then both of the answers are added to each other's list of comparison answers.



Similarity in Pairs

Answer	Graded Score	Compared Answer	Graded Score	Similarity Score
Answer 2	3.0	Answer 3	4.0	0.95
Answer 1	5.0	Answer 3	4.0	0.94
Answer 3	4.0	Answer 4	5.0	0.93
Answer 2	3.0	Answer 4	5.0	0.92

Sorted by Answer

Answer	Graded Score	Answer in text form	Similar Answers	
Answer 2	3.0	Answer in text form	Answer 3	0.95
			Answer 4	0.92
Answer 1	5.0	Answer in text form	Answer 3	0.94
Answer 3	4.0	Answer in text form	Answer 2	0.95
			Answer 1	0.94
			Answer 4	0.93
Answer 4	5.0	Answer in text form	Answer 3	0.93
			Answer 2	0.92

Figure 4.5: Data structure of the client side list of answers and their sublists of similar answers.

Finally, a certainty score for every answer is calculated. The certainty score is a measure of how certain the system is that the answer deserves a different score than it was originally given by the examiner. The certainty score for the recommendation for change in answer a, where Sim denotes the similarity score and g is the graded score for a specified answer, is calculated as:

$$\text{Certainty score}_a = \left| \frac{1}{|\text{comparisons}|} \cdot \sum_{b=1}^{|\text{comparisons}|} \begin{cases} \mathrm{g}_a > \mathrm{g}_b, & -\mathrm{Sim}_{ab} \\ \mathrm{g}_a < \mathrm{g}_b, & +\mathrm{Sim}_{ab} \end{cases} \right|$$

If the graded score of answer a is higher than that of the current comparison answer, the variable is decremented by the similarity score of the two answers, whereas if the comparison answer has been given the higher score, the variable is incremented by the similarity score. The sum of these increments and decrements results in a value that states how certain a recommendation is. If it is positive, it implies that the answer deserves a higher score, and the opposite implies that the answer deserves a lower score. The value is divided by the amount of comparisons to normalize the output between 0.0 and 1.0. To format the value to only tell how certain the system is of a recommendation, the absolute value is calculated.

For example, the more compared answers that have a lower grade than a, the more certain the system is that the grade should be lowered. If there are about as many instances of the comparison answer having a higher and a lower graded score than the original answer, the certainty score will hover around zero, depending on the different similarity scores. This indicates less certainty regarding whether the original answer deserves a different score.

The final step in the sorting process is to sort the order the answers are displayed in. A recommendation containing more compared answers with grades leaning towards the same direction compared to the original answers grade, should be higher up the list of recommendations. This means that if an answers' comparison answers tend to be graded higher than itself, it should be ranked higher than an answer where the comparison answers are more varied in their graded scores. The recommendations could have been sorted by certainty score, but this has the unfortunate side effect of ranking recommendations with very few similar answers listed as high in the list. An answer with several similar answers with pretty high similarity scores should be ranked higher than an answer with only a few good comparisons. As the certainty score is averaged, it performs poorly in sorting with this in mind. The sorting score is therefore calculated as follows:

$$\text{Sorting score}_a = \sum_{b=1}^{|\text{comparisons}|} \begin{cases} \mathrm{g}_a > \mathrm{g}_b, & -1 \\ \mathrm{g}_a < \mathrm{g}_b, & +1 \end{cases}$$

The recommendations are sorted by calculating a sorting score for each, in a similar fashion to the certainty score. The only difference is that the score is incremented and decremented by one instead of the similarity score, as the sorting should only take into account the amount of comparisons and not the score. The list of recommendations is sorted by sorting score.

If two recommendations have the same sorting score, the recommendation with the highest certainty score receives precedence over the other. This algorithm also filters out answers that have less than three comparison answers, and answers that have a certainty score of less than 0.75 (1.0 is the highest possible value). This filtering is done so as not to clutter the website with answers that either have very few similar answers or cannot be said with sufficient certainty to deserve either a higher or lower score. If the completion of the filtering process results in no If there are no recommendations that survive the filtering process for a given question, the website will display a message explaining this.

4.5 Application setup

Although the main focus of this thesis is concerned with finding similar answers and explaining their similarities, it was necessary to create a graphical user interface so that the product could be properly tested by people in the target group. It was decided to build TGRT with a client-server architecture, where the client takes the form of a website. In short, the user uploads a file containing the exam answers to the website, which is sent to the server. The server then calculates similarity scores for the answers and sends back the relevant pairs, followed by the supporting features corresponding to the answer pairs. An explanation of how the different components of the application interact with each other can be seen in Figure 4.6.

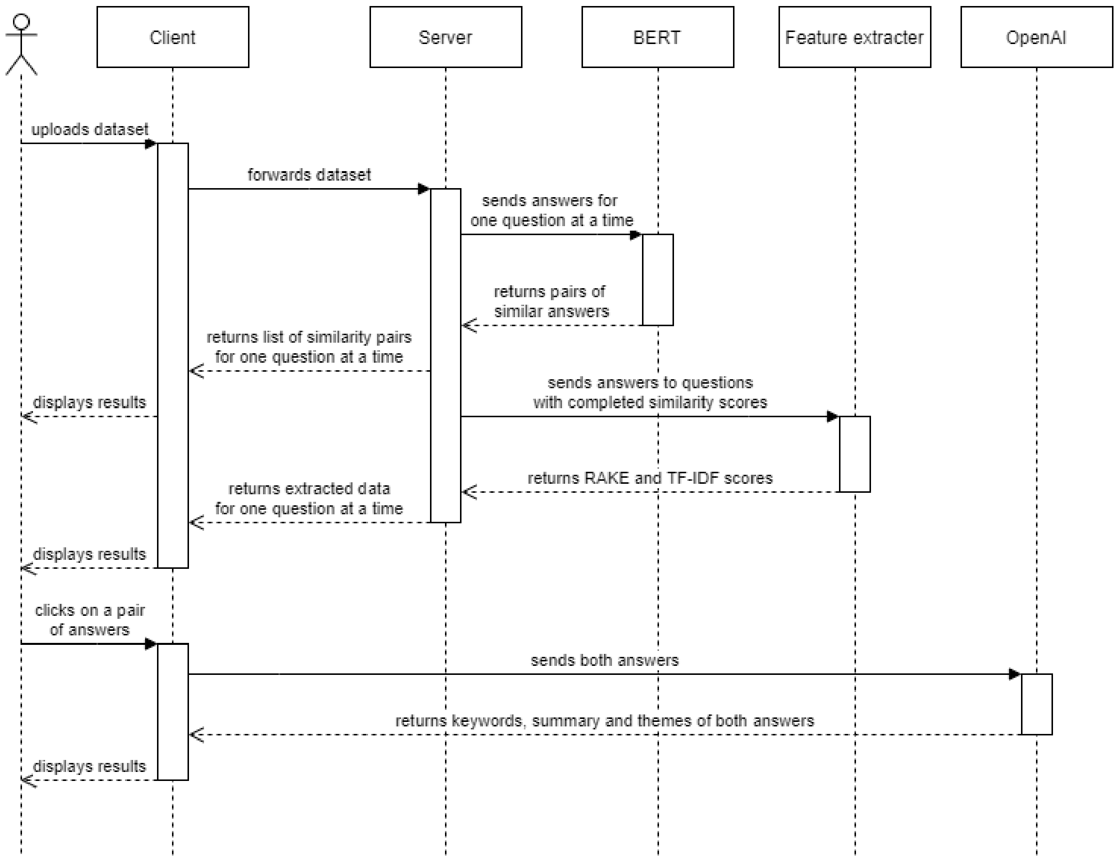

Client
Server
BERT
Feature extracter
OpenAI
uploads dataset
forwards dataset
sends answers for one question at a time
returns pairs of similar answers
returns list of similarity pairs for one question at a time
displays results
sends answers to questions with completed similarity scores
returns RAKE and TF-IDF scores
returns extracted data for one question at a time
displays results
clicks on a pair of answers
sends both answers
returns keywords, summary and themes of both answers
displays results

Figure 4.6: Sequence diagram of the application layout

4.5.1 Client

The website was created using React.js [56]. React is a JavaScript library used for creating user interfaces that was released by Meta (formerly Facebook) in 2013. It enables developers to create reusable components for single-page applications. The website consists of an upload page and the main page. The upload page is used simply for uploading the dataset of exam answers, while all the main functionality of the product is present on the main page. Aside from the feature of uploading a file, the upload page also has a check mark for controlling whether or not the OpenAI requests will be sent. This is just for testing purposes, to prevent too many calls to the API during development. As a free trial of the OpenAI API is used, where every request sent consumes part of the free tokens included in this free trial. The OpenAI check mark is supposed to be checked, when not in a development stage of the process. The upload page can be seen in Figure 4.7.

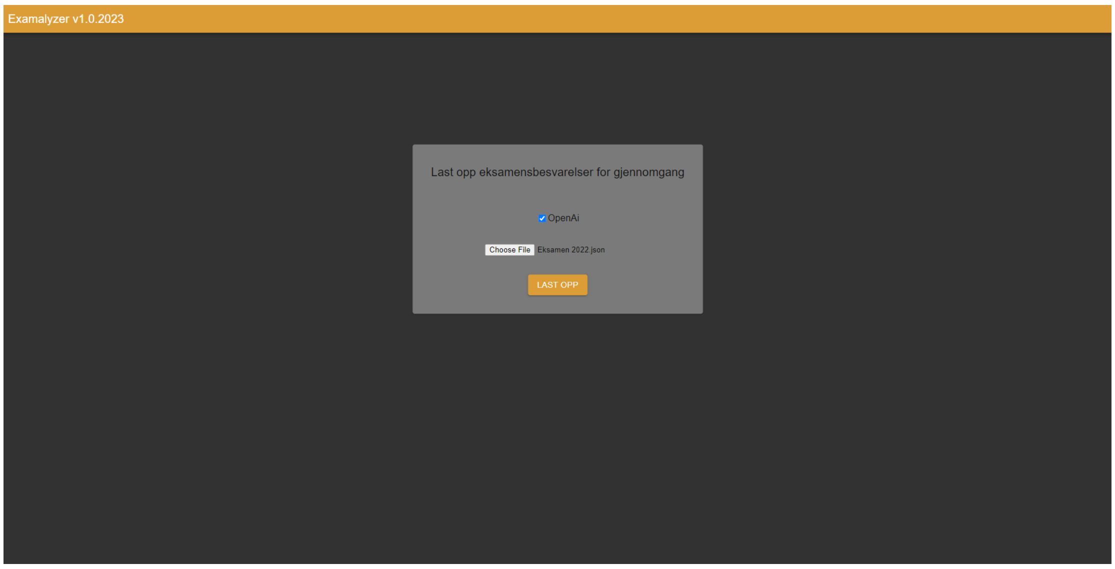


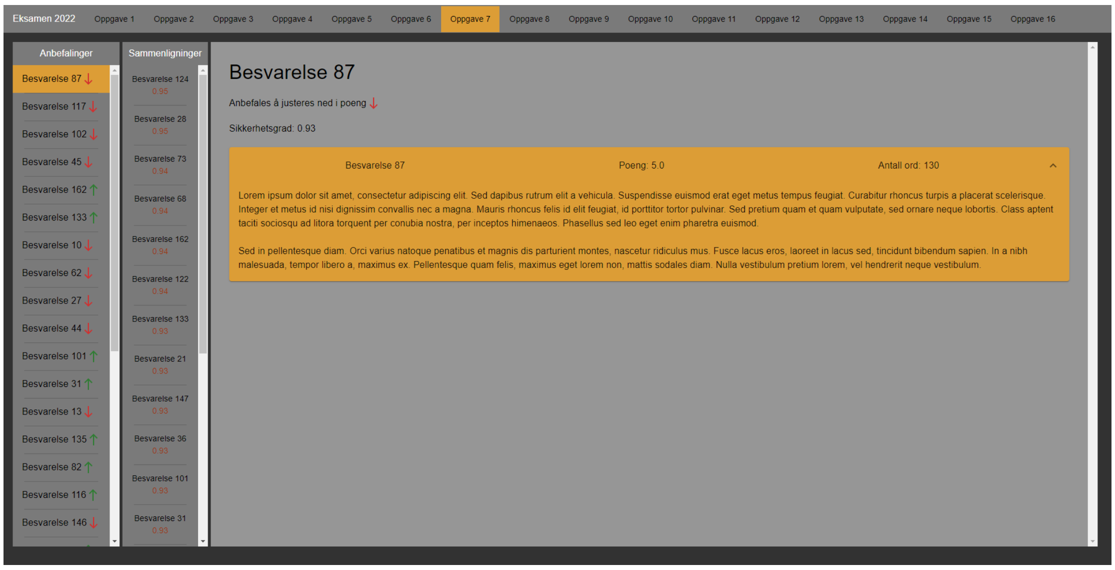
Figure 4.7: The upload page of the website

After the user has clicked the upload button, they are taken to the main page of the website. This page displays the relevant recommendations as they are consecutively received from the server. The first request to the server sends the uploaded file, and it receives back an ID (more on this in subsection 4.5.2), and the amount of relevant questions on the exam (multiple choice questions are excluded, since they are irrelevant for the purposes of the product). The amount of questions is used for initializing the bar containing all questions at the top of the page (see Figure 4.8). The client then starts polling the server for results every ten seconds until recommendations for all questions are received. The recommendations are sent one by one in consecutive order.

The user can change the selected question in the bar at the top of the page, while the list of recommendations for the selected question is displayed at the far left of the page, in the order determined by the sorting algorithm explained in section 4.4. If the results from the server contain no recommendations for a specified question, a message displays this in place of the list of recommendations. The items in the list display the index of the answer along with a colored arrow pointing up or down if the answer should get an increased or decreased grade respectively. A green colored arrow implies that the answer deserves an increase in graded score, while a red arrow implies a decrease.

When selecting a recommendation in the list, that answer's list of compared answers is displayed on the right. In the main section of the screen, the user can see an overview of the selected answer, including its text, length, graded score, certainty score, and whether the system thinks it should be scored higher or lower. When selecting one of the compared answers, the user can see data about both answers side by side. The aforementioned data is displayed for both answers, as well as the TF-IDF terms that appear in both answers with their corresponding scores, RAKE terms and scores for both answers, and the OpenAI data (themes, summaries and keywords for both answers). The TF-IDF and RAKE data is polled from the server, whereas the OpenAI data is retrieved from the OpenAI API when two answers have been selected (see Figure 4.6).

Once the OpenAI data for an answer has been retrieved from the API, it will not try to fetch new data for the same answer even if it is part of a pair with another combination of answers. When changing the question selected at the top of the screen, the corresponding TF-IDF and RAKE terms are also retrieved from the server if they have not already been received. The TF-IDF and RAKE terms have additional functionality to examine the terms in the answer text. When selecting a term in either the TF-IDF or the RAKE table, the terms will be highlighted in the text representation of both answers, making it easy to compare the two with regard to the term and see the context the terms are used in. This does not always work optimally, as the terms listed are in lemmatized form, as described in section 2.2. This leads to some words not matching properly in the answer text, as the terms' suffix and form might have been changed.



Besvarelse 87

Anbefales å justeres ned i poeng ↓

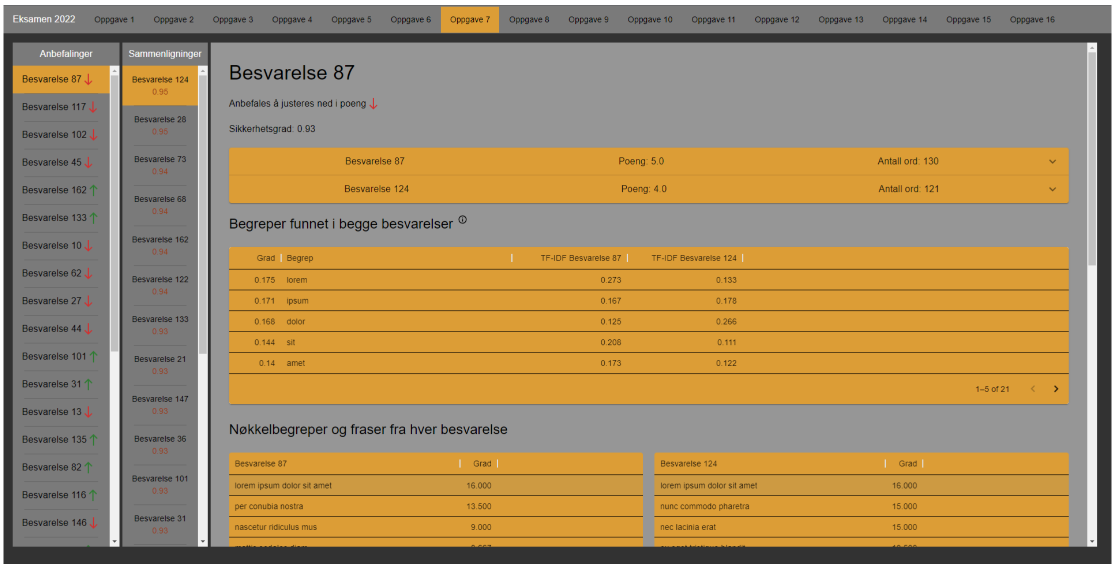
Sikkerhetsgrad: 0.93

Besvarelse 87 Poeng: 5.0 Antall ord: 130

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed dapibus rutrum elit a vehicula. Suspendisse euismod erat eget metus tempus feugiat. Curabitur rhoncus turpis a placerat scelerisque. Integer et metus id nisi dignissim convallis nec a magna. Mauris rhoncus felis id elit feugiat, id porttitor tortor pulvinar. Sed pretium quam et quam vulputate, sed ornare neque lobortis. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Phasellus sed leo eget enim pharetra euismod.

Sed in pellentesque diam. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Fusce lacus eros, laoreet in lacus sed, tincidunt bibendum sapien. In a nibh malesuada, tempor libero a, maximus ex. Pellentesque quam felis, maximus eget lorem non, mattis sodales diam. Nulla vestibulum pretium lorem, vel hendrerit neque vestibulum.

Figure 4.8: The main page of the website, with a question and an answer selected. Due to privacy concerns, the displayed answer consists of generated placeholder text.



Besvarelse 87

Anbefales å justeres ned i poeng

Sikkerhetsgrad: 0.93

Besvarelse 87	Poeng: 5.0	Antall ord: 130
Besvarelse 124	Poeng: 4.0	Antall ord: 121

Begreper funnet i begge besvarelser

Grad	Begrep	TF-IDF Besvarelse 87	TF-IDF Besvarelse 124
0.175	lorem	0.273	0.133
0.171	ipsum	0.167	0.178
0.168	dolor	0.125	0.266
0.144	sit	0.208	0.111
0.14	amet	0.173	0.122

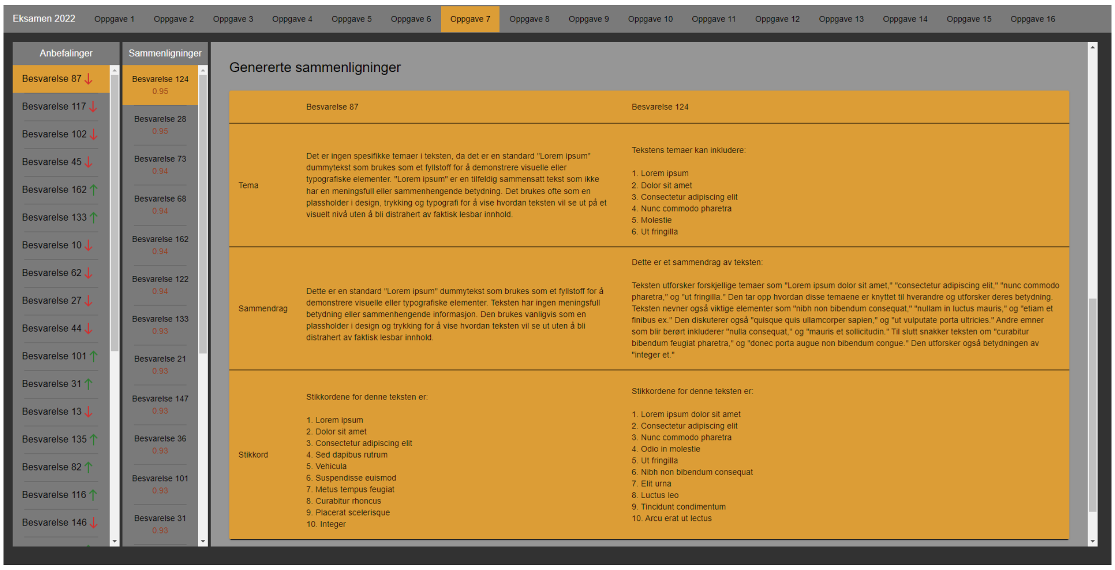
1–5 of 21

Nøkkelbegreper og fraser fra hver besvarelse

Besvarelse 87	Grad
lorem ipsum dolor sit amet	16.000
per conubia nostra	13.500
nascetur ridiculus mus	9.000

Besvarelse 124	Grad
lorem ipsum dolor sit amet	16.000
nunc commodo pharetra	15.000
nec lacinia erat	15.000

Figure 4.9: The main page of the website, with a comparison answer selected. TF-IDF and RAKE terms/scores can be seen in the bottom half of the page.



Genererte sammenligninger

	Besvarelse 87	Besvarelse 124
Tema	Det er ingen spesifikke temaer i teksten, da det er en standard "Lorem ipsum" dummytekst som brukes som et fyllstoff for å demonstrere visuelle eller typografiske elementer. "Lorem ipsum" er en tilfeldig sammensatt tekst som ikke har en meningsfull eller sammenhengende betydning. Det brukes ofte som en plassholder i design, trykking og typografi for å vise hvordan teksten vil se ut på et visuelt nivå uten å bli distrahert av faktisk lesbar innhold.	Tekstens temaer kan inkludere: 1. Lorem ipsum 2. Dolor sit amet 3. Consectetur adipiscing elit 4. Nunc commodo pharetra 5. Molestie 6. Ut fringilla
Sammendrag	Dette er en standard "Lorem ipsum" dummytekst som brukes som et fyllstoff for å demonstrere visuelle eller typografiske elementer. Teksten har ingen meningsfull betydning eller sammenhengende informasjon. Den brukes vanligvis som en plassholder i design og trykking for å vise hvordan teksten vil se ut uten å bli distrahert av faktisk lesbar innhold.	Dette er et sammendrag av teksten: Teksten utforsker forskjellige temaer som "Lorem ipsum dolor sit amet," "consectetur adipiscing elit," "nunc commodo pharetra," og "ut fringilla." Den tar opp hvordan disse temaene er knyttet til hverandre og utforsker deres betydning. Teksten nevner også viktige elementer som "nibh non bibendum consequat," "nullam in luctus mauris," og "etiam et finibus ex." Den diskuterer også "quisque quis ullamcorper sapien," og "ut vulputate porta ultricies." Andre emner som blir berørt inkluderer "nulla consequat," og "mauris et sollicitudin." Til slutt snakker teksten om "curabitur bibendum feugiat pharetra," og "donec porta augue non bibendum congue." Den utforsker også betydningen av "integer et."
Stikkord	Stikkordene for denne teksten er: 1. Lorem ipsum 2. Dolor sit amet 3. Consectetur adipiscing elit 4. Sed dapibus rutrum 5. Vehicula 6. Suspendisse euismod 7. Metus tempus feugiat 8. Curabitur rhoncus 9. Placerat scelerisque 10. Integer	Stikkordene for denne teksten er: 1. Lorem ipsum dolor sit amet 2. Consectetur adipiscing elit 3. Nunc commodo pharetra 4. Odio in molestie 5. Ut fringilla 6. Nibh non bibendum consequat 7. Elit urna 8. Luctus leo 9. Tincidunt condimentum 10. Arcu erat ut lectus

Figure 4.10: The main page of the website, with a comparison answer selected. Here the data generated by OpenAI can be seen.

4.5.2 Server

The server uses the web framework Flask [57], which is written in Python. Flask is considered a micro framework because it does not require particular tools or libraries. Thus, it is quick and easy to set up. The server hosts three main endpoints available for the client. The upload path is where the server receives the initial request containing the exam dataset from the client. The server then generates a unique ID as a random string of letters and numbers with a length of five. This ID makes it easy to keep track of the data on the server, as it quickly can become confusing since the processing is performed in multiple threads. The ID is also used as a key for the client to fetch the correct temporarily saved data from the server. The server responds with the generated ID and a number representing the amount of relevant questions on the exam. This number is the result from an analysis of all the questions in the exam and what types of answers it has. The filtering process is described in length in subsection 4.2.1.

The server then starts a thread that performs the main calculations on the dataset. The algorithm for this is further described in subsection 4.1.2, but simply put, it extracts pairs of similar answers from the dataset. The resulting pairs of similar answers are then sorted as described in section 4.4, that is, in descending order according to similarity score, and filtering out pairs with the same graded scores, as well as pairs with similarity scores below the threshold. This data is then temporarily saved on the server, one question at a time, so that it is easily accessible when the client starts polling.

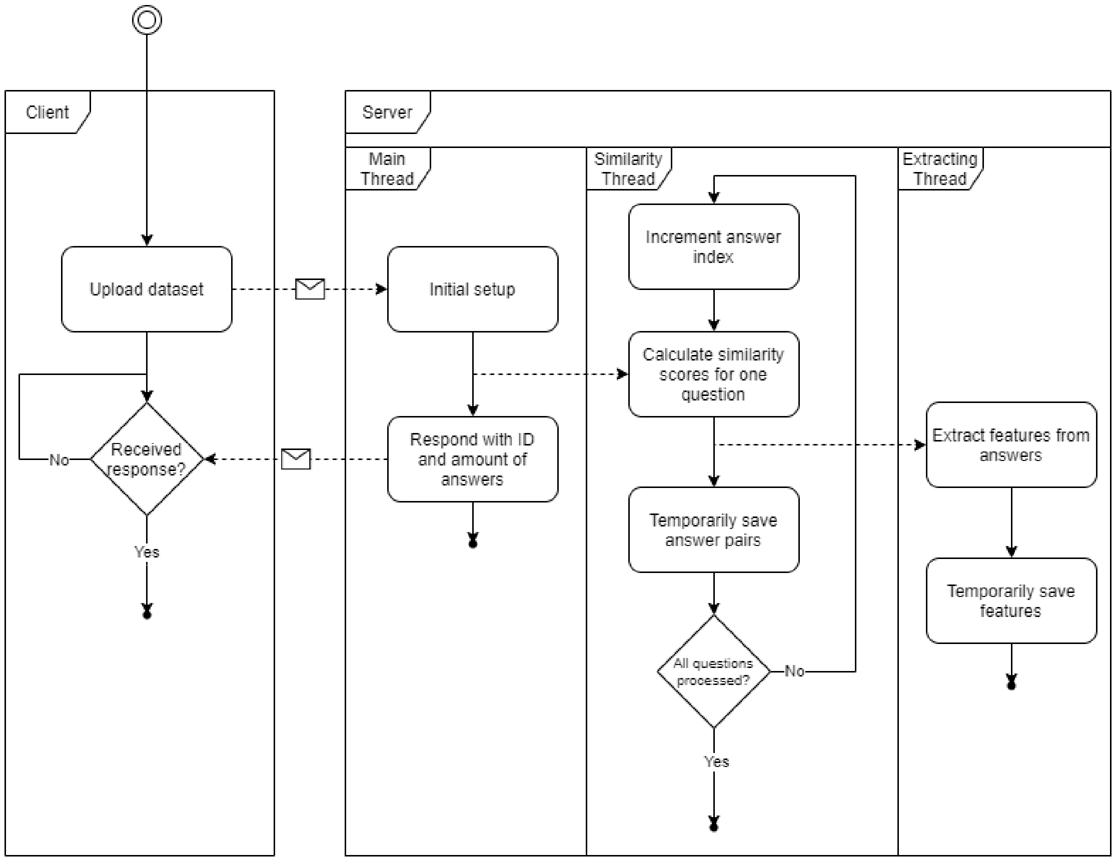

Client
Server
Main Thread
Similarity Thread
Extracting Thread
Upload dataset
Initial setup
Increment answer index
Calculate similarity scores for one question
Extract features from answers
Received response?
No
Yes
Respond with ID and amount of answers
Temporarily save answer pairs
Temporarily save features
All questions processed?
No
Yes

Figure 4.11: The main process on server that runs after the initial request with the dataset is received

When the similarity data for one question is calculated, sorted and saved, another thread is started for extracting the TF-IDF and RAKE data for that question and saving it on the server. The extracted features are also temporarily saved in a separate file on the server. When the client has received the response saying that the process has started with the generated ID and the number of valid questions, it moves on to the polling process.

The two other endpoints of the server are for polling for answer pairs and supporting features. As the client has received the amount of valid questions, it knows how many sets of results it is supposed to receive. Polling for both the answer pairs and the supporting features are initialized at the same time, and the two processes are very similar (Figure 4.12). Every ten seconds, the client sends a request for the results. If the results are ready, they are sent back to the client. If they are not ready, the server responds with the message "False", indicating that it is not ready. Every time results for a question are received, either answer pairs or supporting features, they are immediately displayed on the website. The client then iterates the question it requests and continues polling every ten seconds as long as there are questions whose data has not been received.

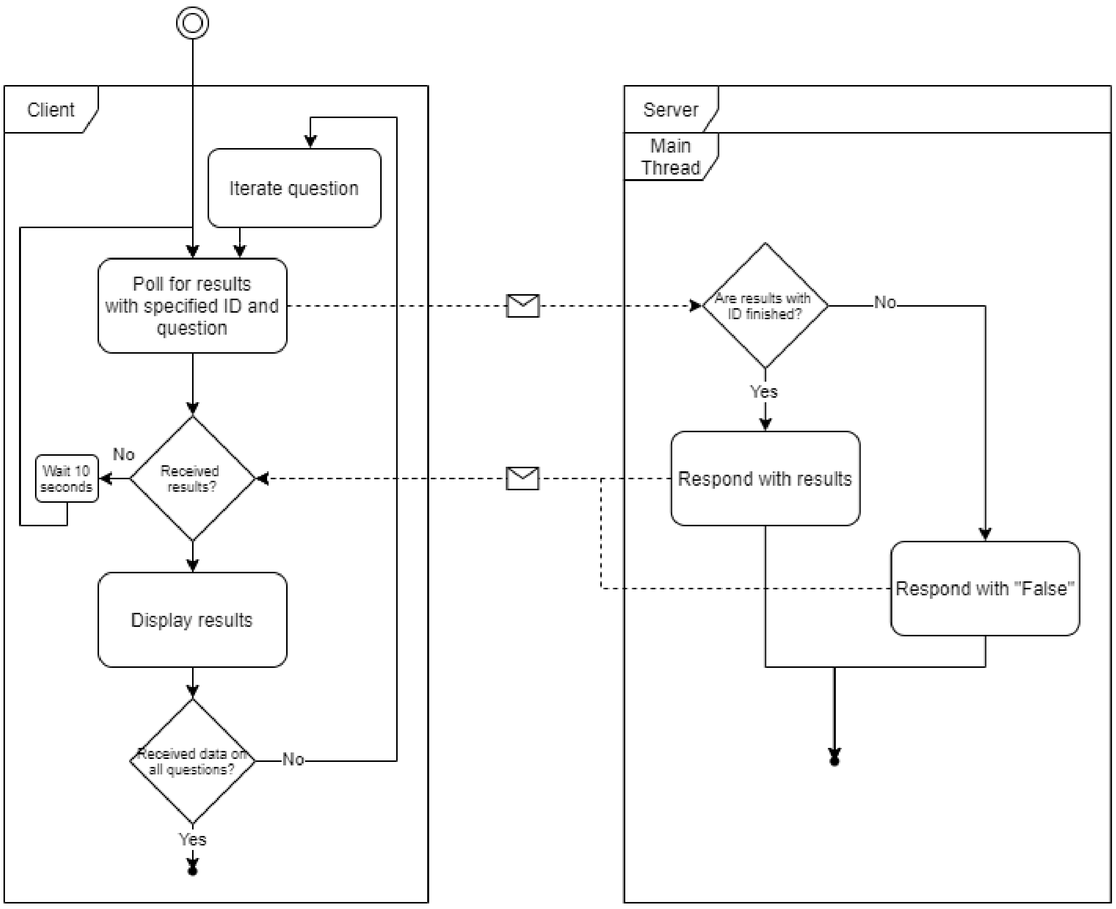


Figure 4.12: The polling process between client and server to fetch both answer pairs and the extracted features. They are two separate processes, but the flow is exactly the same.

The server is hosted on an NTNU server installed on campus. It was initially hosted on the researchers' own computers, but the heavy computations took too long to process. The server is connected via a Secure Shell (SSH) and can be accessed anywhere where NTNU's internet is available.

4.6 Considered additions

Several functions and extensions to TGRT were discussed during development. Some of them were decided against for different reasons that might not be immediately evident. At first glance, they seem like expansions to the functionality of TGRT that would make it perform better and be more helpful in assisting the examiners. There are however downsides that emerged as the project evolved.

4.6.1 Same grade with different content

When the project started, the innovative method had a much broader definition. It was always supposed to double-check grading and give warnings if the grades were unfair, but it evolved over time into a more specific use case. This was to give warnings when the contents of the answers are similar, but the grades are different. Early on, there were also ambitions of doing the opposite, namely warning the examiner when the grades are similar, but the content is dissimilar. This proved to be a bad idea on several grounds.

The main problem with this idea is that the requirements for achieving a grade are not always set in stone. In some courses, there might be straight forward questions with straight forward answers. This is however not always the case. If a question asks for several aspects of a topic and the answers contain explanations for different aspects, the answers become more complicated to compare. Even in a simple case where each aspect is weighted equally, this would be a problem. If TGRT was programmed to warn the examiner when the contents are different and the grades are not, this example would result in many recommendations with faulty logic. The graded score could be correct, but since the answers do not contain the same information, the tool would recommend they get a different grade.

Even though TGRT was not programmed to detect dissimilar answers with the same grade, it should in theory flag them either way. In the aforementioned example, the grades were correct even though the content was not similar. In cases like this, it is not desired for TGRT to flag the answer as wrong. Another example would be that an answer was given a higher graded score than it deserved because the answer did not contain the expected aspects of the questions needed for that grade. In this case it is not desired for TGRT to flag the answer as too high, but even though it is compared as very dissimilar to the other answers on that grade level, it will not be flagged as an error simply because they have the same grade. However, it will also be compared to the answers with different grades, and when it is compared to be more similar to answers on a lower grade level, TGRT will recommend that it is lowered. This means that even if it specifically does not flag answers with similar grades and different content for that reason alone, they are flagged in the cases where they are more similar to answers on a different grade level.

4.6.2 Recommending specific grade shifts

The results from TGRT gives the examiner a recommendation to either increase or decrease an answers' graded score. A natural extension to this is to recommend a specific amount of points to shift the score. If an answer has received a score of 5.0 and has a high similarity score with many answers that got a graded score of 3.0, the tool could then recommend the examiner to drop the score specifically by two points. To calculate the difference in graded score between the answer and the average score of the most similar answers is not difficult and could easily be implemented. This was decided against primarily because of the implications this entails.

TGRT is not perfect. As will be clear from the tests of the algorithm (subsection 5.1.1) to the user tests (section 5.2), there are always outliers and occasions where it judges incorrectly. By only recommending that the answer should increase or decrease by an unspecified amount, there is an acknowledgement that the tool is not an automatic grader. It only gives the examiner a warning that an answer might have been graded slightly wrong and an indication of which direction on the scale it could be adjusted. If the tool stated exactly what grade it recommends for an answer, it would indicate that the recommendation is objectively correct. This could lead to examiners blindly trusting the recommendations and just accepting what grade is suggested. Instead, TGRT wants to invite the examiner to analyze and understand what might the discrepancies might be, so that they can draw their own conclusions as to what the final graded score should be.

4.6.3 Spelling errors

Spelling errors are an omnipresent phenomenon in exam answers, as in every other written medium. This might have posed a challenge to the model used in our product, as it creates embedding vectors for each token in the input text so that similar words have similar numbers in their embedding vectors. The presence of spelling errors has the potential to confuse the model by obscuring words and making them unrecognizable, so that certain embedding vectors turn out a lot different from how they would be if the words were spelled correctly. This does not only affect the embedding vectors either, as the TF-IDF matrix would also look different. Instead of the same word occupying one column in the matrix, the differently spelled versions of the same word would be found in separate columns, and the scores given would not reflect the words' actual frequency in the answers being compared. It was therefore decided to investigate options to mitigate the effects that spelling errors might have on the results generated by the different components of the product.

A way to do this was to try to correct as many spelling errors as possible before comparing the answers. This was tested by trying out the Python library SymSpell [58]. SymSpell uses the edit distance algorithm described in section 2.8 to find potential candidates for word replacements. The principle is that if a word is spelled incorrectly, the edit distance between the incorrectly spelled word and the correctly spelled word is probably very low (meaning that there likely is not more than one or two letters that are different in both spellings). By analyzing the dictionary for words within a small edit distance from the misspelled word, the correctly spelled version of the word might be found, and can then replace the spelling error in the text.

In order to test SymSpell a prerequisite was thus a Norwegian dictionary where lookups for incorrectly spelled words could be done. SymSpell requires a specific format for the dictionary to be used, namely one column for the words themselves, and one column for the corresponding frequencies of the words. The frequencies are calculated by counting the appearances of every word in the corpus that the dictionary is based on. After some searching, such a dictionary was found [59], which was posted online by the Faculty of Humanities at the University of Bergen, and was purported to contain the most common words in the Norwegian language. This dictionary, which is based on an unknown corpus, consisted of 462 000 words sorted by their frequency. It contained word forms as well as conjugations for word classes. With this dictionary in place, SymSpell was ready to be tested.

When testing, a decision had to be made on how to detect which words were to be considered misspelled in the first place. One option was to simply replace every word, no matter if it is misspelled or not, with the word of the highest frequency within the set edit distance. The suggested words for replacements do not include the word being compared itself. This turned out to not be a good option, as that led to correctly spelled words being replaced by similar words within the set edit distance. Another way was to amend the method described above, but only replacing the word if the frequency of the most frequent suggested word is higher than the frequency of the original word (if the original word is present in the dictionary at all). While this did mitigate the problem of correctly spelled words being replaced somewhat, it did not solve it entirely. For example, it would lead to the word "at" (the eleventh most frequent word in the dictionary) being replaced by "av" (the fifth most frequent word). The option that turned out to be best at replacing *only* incorrectly spelled words was to only look for replacement words if the original word is not present in the dictionary at all. Thus, the only way for correctly spelled words to be replaced is if the word is obscure enough to not be among the 462 000 words in the dictionary, and at least one word within the edit distance *is* in the dictionary. This is a rare occurrence, as the edit distance was set to 1. In SymSpell's case, that means that no more than one letter could be wrong, as substitution counts as one operation.

There was however a problem with that approach. Since the dictionary likely is based on a large corpus where every word in all the documents are included without any qualifications, it contains many spelling errors that appeared in the documents (although these have very low frequencies). For example, it includes spellings like "strker", "strmet", "polti" and "politker", which are all missing a letter. The result of this is that if an answer contains any of the spelling errors that are included in the dictionary, they will not be replaced, leading to the usefulness of spell checking being somewhat reduced. Granted, there are many typical spelling errors that do not appear in the dictionary, so the problem might not be that significant. Moreover, when testing answers with spelling errors on the different BERT models, it was discovered that their similarity scores were mostly very high when comparing to their correctly spelled counterparts. As shown in Table 6.2, they all scored higher than 0.85, and most of them higher than 0.90. This indicates that the BERT models are able to derive meaning from incorrectly spelled words by looking at the context of the sentence the word is being used in. Because of this, and the problem of incorrectly spelled words being present in the dictionary, it was decided not to implement the spell checker in the product, as it would likely not have a significant effect on the results.

CHAPTER

FIVE

TESTS

Two types of tests were performed in order to answer the research questions. The first type was for testing the different BERT models and how well they suited the task of creating accurate embeddings for Norwegian text. This included some tests for measuring similarity between texts where a desired result was known, and some that tested different aspects of how the models reacted to textual differences. The friction dataset had many variants of the same texts that included for example spelling errors or word swaps with synonyms. In a similar fashion to Wangkriangkri et al. [38], the tests on the variants would in some sense test the robustness of the models and grant a wider knowledge of what aspects each model emphasizes.

The other type of tests were user tests. Performing user tests would encompass all the research questions in that they tested the chosen BERT model, how useful the selected supporting features were, and how real examiners would respond to TGRT and what might be interpreted as its "corrections" of their work.

5.1 BERT models tests

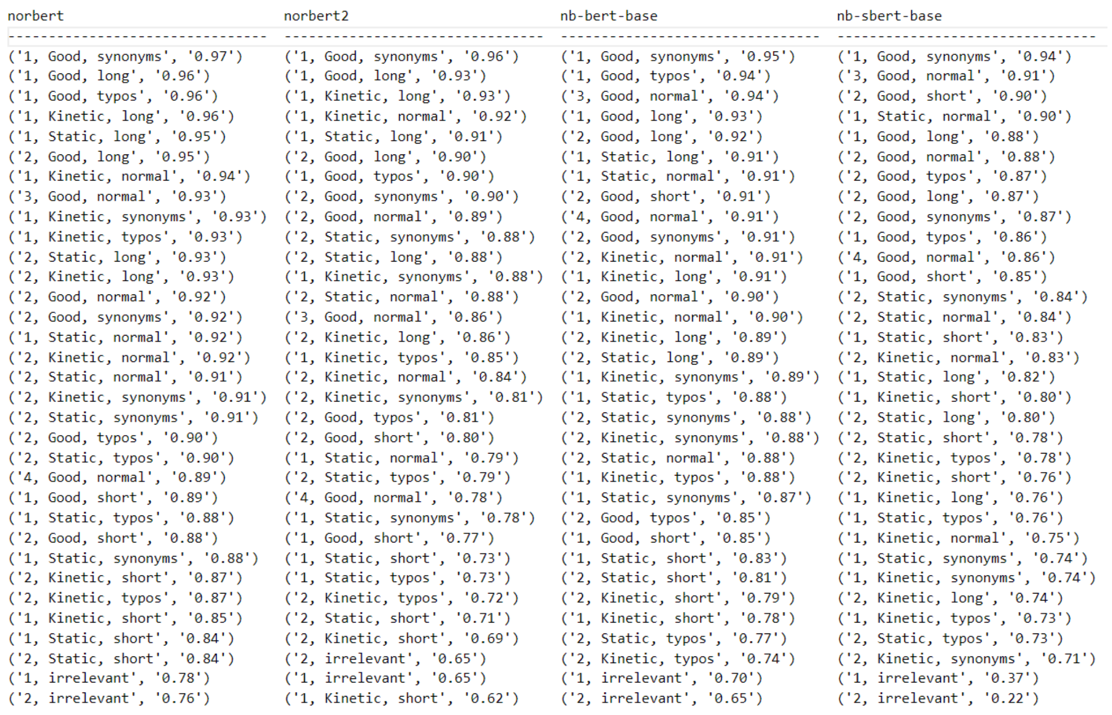
To test how well the models were at measuring similarity between pairs of texts, the custom datasets explained in subsection 4.2.2 were used. The tests were performed by choosing a specific base answer in the dataset, comparing it to every other answer in the dataset, and outputting the similarity scores between them. A custom file writer was made that output the result from all BERT versions, with each resulting item consisting of an answers' category variation and its corresponding similarity score with respect to the answer being compared. The category variation determines what category the answer has, either determining what movies it describes (in the Tarantino datasets) or what variations of the four different written answers it consists of (in the friction dataset). The base answer is not present in the resulting tables, as it would have gotten the top similarity score of 1.0. For the friction dataset, two tests with different answers specified were performed. In both tests, the answer specified to be calculated similarity against was one of the answers labeled "Good, normal".

Two separate tests for the Tarantino datasets with dual descriptions were also performed (with and without titles). The texts to be compared to were specified to be the same across the two datasets to see if there was any difference in results. For the first test a text containing descriptions of the movies Inglourious Basterds and Django Unchained (shortened to "IB & DU") was chosen, and for the second test the chosen combination was Inglourious Basterds and Pulp Fiction (shortened to "IB & PF").

The final comparison algorithm test was performed on the custom Tarantino dataset with split entries as explained in subsection 4.2.2. The descriptions, numbering 36 in total, were compared to each other. The optimal result for any model is to have all the descriptions for the same movie at the top of the list of comparisons. The tests compared the entries labeled "IB 2" and "DU 2" against the other entries.

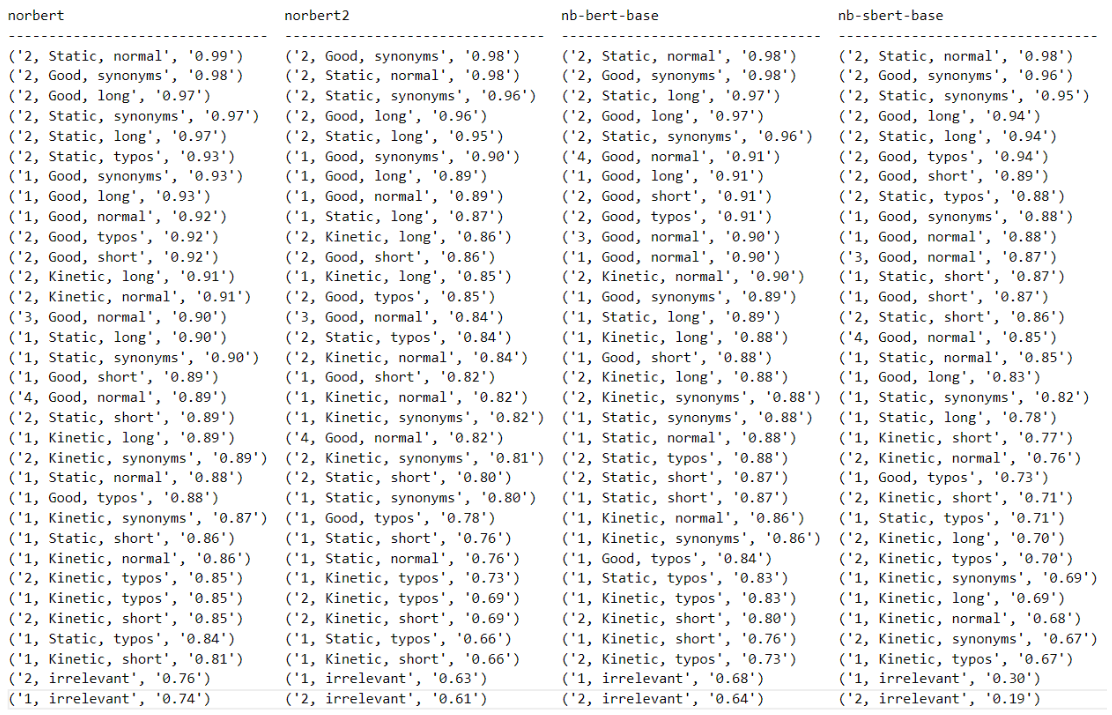
5.1.1 Friction dataset results

The following figures are outputs from the tests of the comparison models. Each column shows results for the specified model. Entries contain the category and the similarity score to the answer being compared. The entries are sorted by similarity score in descending order.



```
norbert                            norbert2                           nb-bert-base                       nb-sbert-base
---------------------------------  ---------------------------------  ---------------------------------  ---------------------------------
('1, Good, synonyms', '0.97')      ('1, Good, synonyms', '0.96')      ('1, Good, synonyms', '0.95')      ('1, Good, synonyms', '0.94')
('1, Good, long', '0.96')          ('1, Good, long', '0.93')          ('1, Good, typos', '0.94')         ('3, Good, normal', '0.91')
('1, Good, typos', '0.96')         ('1, Kinetic, long', '0.93')       ('3, Good, normal', '0.94')        ('2, Good, short', '0.90')
('1, Kinetic, long', '0.96')       ('1, Kinetic, normal', '0.92')     ('1, Good, long', '0.93')          ('1, Static, normal', '0.90')
('1, Static, long', '0.95')        ('1, Static, long', '0.91')        ('2, Good, long', '0.92')          ('1, Good, long', '0.88')
('2, Good, long', '0.95')          ('2, Good, long', '0.90')          ('1, Static, long', '0.91')        ('2, Good, normal', '0.88')
('1, Kinetic, normal', '0.94')     ('1, Good, typos', '0.90')         ('1, Static, normal', '0.91')      ('2, Good, typos', '0.87')
('3, Good, normal', '0.93')        ('2, Good, synonyms', '0.90')      ('2, Good, short', '0.91')         ('2, Good, long', '0.87')
('1, Kinetic, synonyms', '0.93')   ('2, Good, normal', '0.89')        ('4, Good, normal', '0.91')        ('2, Good, synonyms', '0.87')
('1, Kinetic, typos', '0.93')      ('2, Static, synonyms', '0.88')    ('2, Good, synonyms', '0.91')      ('1, Good, typos', '0.86')
('2, Static, long', '0.93')        ('2, Static, long', '0.88')        ('2, Kinetic, normal', '0.91')     ('4, Good, normal', '0.86')
('2, Kinetic, long', '0.93')       ('1, Kinetic, synonyms', '0.88')   ('1, Kinetic, long', '0.91')       ('1, Good, short', '0.85')
('2, Good, normal', '0.92')        ('2, Static, normal', '0.88')      ('2, Good, normal', '0.90')        ('2, Static, synonyms', '0.84')
('2, Good, synonyms', '0.92')      ('3, Good, normal', '0.86')        ('1, Kinetic, normal', '0.90')     ('2, Static, normal', '0.84')
('1, Static, normal', '0.92')      ('2, Kinetic, long', '0.86')       ('2, Kinetic, long', '0.89')       ('1, Static, short', '0.83')
('2, Kinetic, normal', '0.92')     ('1, Kinetic, typos', '0.85')      ('2, Static, long', '0.89')        ('2, Kinetic, normal', '0.83')
('2, Static, normal', '0.91')      ('2, Kinetic, normal', '0.84')     ('1, Kinetic, synonyms', '0.89')   ('1, Static, long', '0.82')
('2, Kinetic, synonyms', '0.91')   ('2, Kinetic, synonyms', '0.81')   ('1, Static, typos', '0.88')       ('1, Kinetic, short', '0.80')
('2, Static, synonyms', '0.91')    ('2, Good, typos', '0.81')         ('2, Static, synonyms', '0.88')    ('2, Static, long', '0.80')
('2, Good, typos', '0.90')         ('2, Good, short', '0.80')         ('2, Kinetic, synonyms', '0.88')   ('2, Static, short', '0.78')
('2, Static, typos', '0.90')       ('1, Static, normal', '0.79')      ('2, Static, normal', '0.88')      ('2, Kinetic, typos', '0.78')
('4, Good, normal', '0.89')        ('2, Static, typos', '0.79')       ('1, Kinetic, typos', '0.88')      ('2, Kinetic, short', '0.76')
('1, Good, short', '0.89')         ('4, Good, normal', '0.78')        ('1, Static, synonyms', '0.87')    ('1, Kinetic, long', '0.76')
('1, Static, typos', '0.88')       ('1, Static, synonyms', '0.78')    ('2, Good, typos', '0.85')         ('1, Static, typos', '0.76')
('2, Good, short', '0.88')         ('1, Good, short', '0.77')         ('1, Good, short', '0.85')         ('1, Kinetic, normal', '0.75')
('1, Static, synonyms', '0.88')    ('1, Static, short', '0.73')       ('1, Static, short', '0.83')       ('1, Static, synonyms', '0.74')
('2, Kinetic, short', '0.87')      ('1, Static, typos', '0.73')       ('2, Static, short', '0.81')       ('1, Kinetic, synonyms', '0.74')
('2, Kinetic, typos', '0.87')      ('2, Kinetic, typos', '0.72')      ('2, Kinetic, short', '0.79')      ('2, Kinetic, long', '0.74')
('1, Kinetic, short', '0.85')      ('2, Static, short', '0.71')       ('1, Kinetic, short', '0.78')      ('1, Kinetic, typos', '0.73')
('1, Static, short', '0.84')       ('2, Kinetic, short', '0.69')      ('2, Static, typos', '0.77')       ('2, Static, typos', '0.73')
('2, Static, short', '0.84')       ('2, irrelevant', '0.65')          ('2, Kinetic, typos', '0.74')      ('2, Kinetic, synonyms', '0.71')
('1, irrelevant', '0.78')          ('1, irrelevant', '0.65')          ('1, irrelevant', '0.70')          ('1, irrelevant', '0.37')
('2, irrelevant', '0.76')          ('1, Kinetic, short', '0.62')      ('2, irrelevant', '0.65')          ('2, irrelevant', '0.22')
```

Figure 5.1: Results of comparison test with the friction dataset where the answer being compared is "1, Good, normal" as explained in section 5.1.

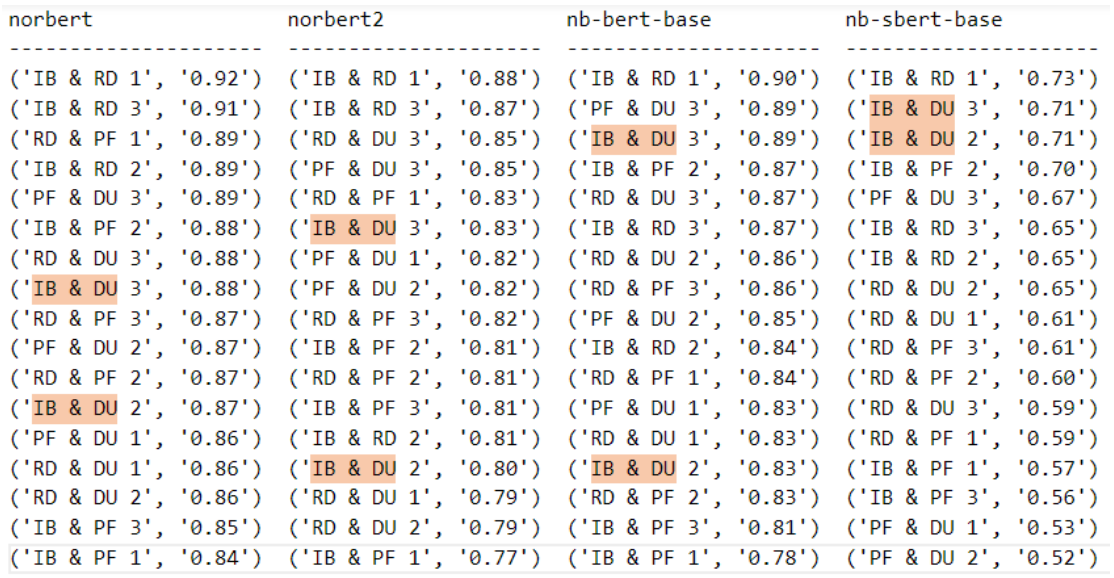


```
norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('2, Static, normal', '0.99')     ('2, Good, synonyms', '0.98')     ('2, Static, normal', '0.98')     ('2, Static, normal', '0.98')
('2, Good, synonyms', '0.98')     ('2, Static, normal', '0.98')     ('2, Good, synonyms', '0.98')     ('2, Good, synonyms', '0.96')
('2, Good, long', '0.97')         ('2, Static, synonyms', '0.96')   ('2, Static, long', '0.97')       ('2, Static, synonyms', '0.95')
('2, Static, synonyms', '0.97')   ('2, Good, long', '0.96')         ('2, Good, long', '0.97')         ('2, Good, long', '0.94')
('2, Static, long', '0.97')       ('2, Static, long', '0.95')       ('2, Static, synonyms', '0.96')   ('2, Static, long', '0.94')
('2, Static, typos', '0.93')      ('1, Good, synonyms', '0.90')     ('4, Good, normal', '0.91')       ('2, Good, typos', '0.94')
('1, Good, synonyms', '0.93')     ('1, Good, long', '0.89')         ('1, Good, long', '0.91')         ('2, Good, short', '0.89')
('1, Good, long', '0.93')         ('1, Good, normal', '0.89')       ('2, Good, short', '0.91')        ('2, Static, typos', '0.88')
('1, Good, normal', '0.92')       ('1, Static, long', '0.87')       ('2, Good, typos', '0.91')        ('1, Good, synonyms', '0.88')
('2, Good, typos', '0.92')        ('2, Kinetic, long', '0.86')      ('3, Good, normal', '0.90')       ('1, Good, normal', '0.88')
('2, Good, short', '0.92')        ('2, Good, short', '0.86')        ('1, Good, normal', '0.90')       ('3, Good, normal', '0.87')
('2, Kinetic, long', '0.91')      ('1, Kinetic, long', '0.85')      ('2, Kinetic, normal', '0.90')    ('1, Static, short', '0.87')
('2, Kinetic, normal', '0.91')    ('2, Good, typos', '0.85')        ('1, Good, synonyms', '0.89')     ('1, Good, short', '0.87')
('3, Good, normal', '0.90')       ('3, Good, normal', '0.84')       ('1, Static, long', '0.89')       ('2, Static, short', '0.86')
('1, Static, long', '0.90')       ('2, Static, typos', '0.84')      ('1, Kinetic, long', '0.88')      ('4, Good, normal', '0.85')
('1, Static, synonyms', '0.90')   ('2, Kinetic, normal', '0.84')    ('1, Good, short', '0.88')        ('1, Static, normal', '0.85')
('1, Good, short', '0.89')        ('1, Good, short', '0.82')        ('2, Kinetic, long', '0.88')      ('1, Good, long', '0.83')
('4, Good, normal', '0.89')       ('1, Kinetic, normal', '0.82')    ('2, Kinetic, synonyms', '0.88')  ('1, Static, synonyms', '0.82')
('2, Static, short', '0.89')      ('1, Kinetic, synonyms', '0.82')  ('1, Static, synonyms', '0.88')   ('1, Static, long', '0.78')
('1, Kinetic, long', '0.89')      ('4, Good, normal', '0.82')       ('1, Static, normal', '0.88')     ('1, Kinetic, short', '0.77')
('2, Kinetic, synonyms', '0.89')  ('2, Kinetic, synonyms', '0.81')  ('2, Static, typos', '0.88')      ('2, Kinetic, normal', '0.76')
('1, Static, normal', '0.88')     ('2, Static, short', '0.80')      ('2, Static, short', '0.87')      ('1, Good, typos', '0.73')
('1, Good, typos', '0.88')        ('1, Static, synonyms', '0.80')   ('1, Static, short', '0.87')      ('2, Kinetic, short', '0.71')
('1, Kinetic, synonyms', '0.87')  ('1, Good, typos', '0.78')        ('1, Kinetic, normal', '0.86')    ('1, Static, typos', '0.71')
('1, Static, short', '0.86')      ('1, Static, short', '0.76')      ('1, Kinetic, synonyms', '0.86')  ('2, Kinetic, long', '0.70')
('1, Kinetic, normal', '0.86')    ('1, Static, normal', '0.76')     ('1, Good, typos', '0.84')        ('2, Kinetic, typos', '0.70')
('2, Kinetic, typos', '0.85')     ('1, Kinetic, typos', '0.73')     ('1, Static, typos', '0.83')      ('1, Kinetic, synonyms', '0.69')
('1, Kinetic, typos', '0.85')     ('2, Kinetic, typos', '0.69')     ('1, Kinetic, typos', '0.83')     ('1, Kinetic, long', '0.69')
('2, Kinetic, short', '0.85')     ('2, Kinetic, short', '0.69')     ('2, Kinetic, short', '0.80')     ('1, Kinetic, normal', '0.68')
('1, Static, typos', '0.84')      ('1, Static, typos', '0.66')      ('1, Kinetic, short', '0.76')     ('2, Kinetic, synonyms', '0.67')
('1, Kinetic, short', '0.81')     ('1, Kinetic, short', '0.66')     ('2, Kinetic, typos', '0.73')     ('1, Kinetic, typos', '0.67')
('2, irrelevant', '0.76')         ('1, irrelevant', '0.63')         ('1, irrelevant', '0.68')         ('1, irrelevant', '0.30')
('1, irrelevant', '0.74')         ('2, irrelevant', '0.61')         ('2, irrelevant', '0.64')         ('2, irrelevant', '0.19')
```

Figure 5.2: Results of comparison test with the friction dataset where the answer being compared is "2, Good, normal" as explained in section 5.1.

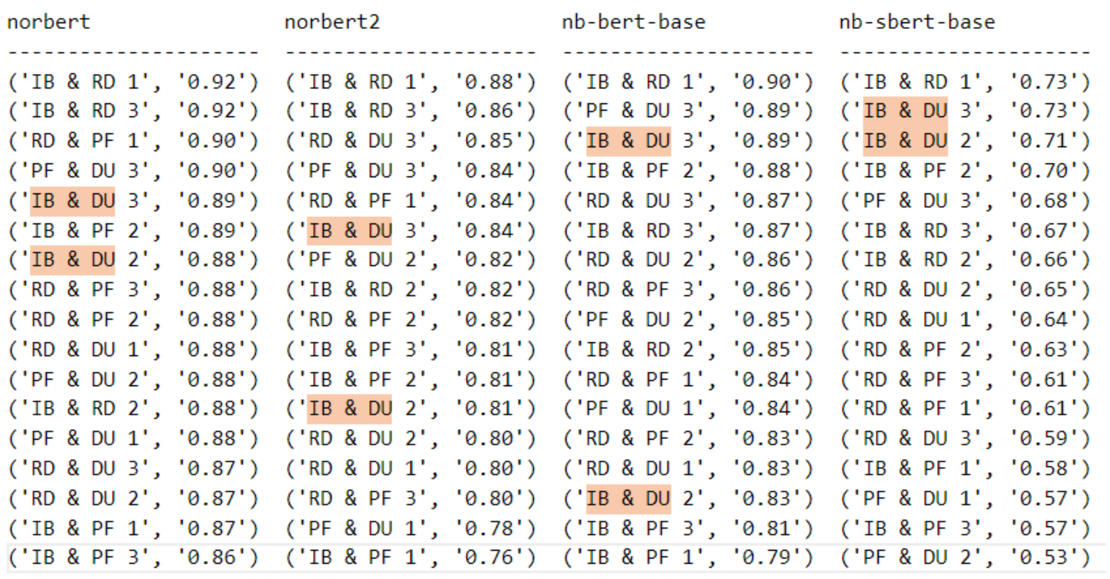
5.1.2 Tarantino datasets results

In the following figures, the category used as the compared entry is highlighted to show where the most similar items thematically are listed.



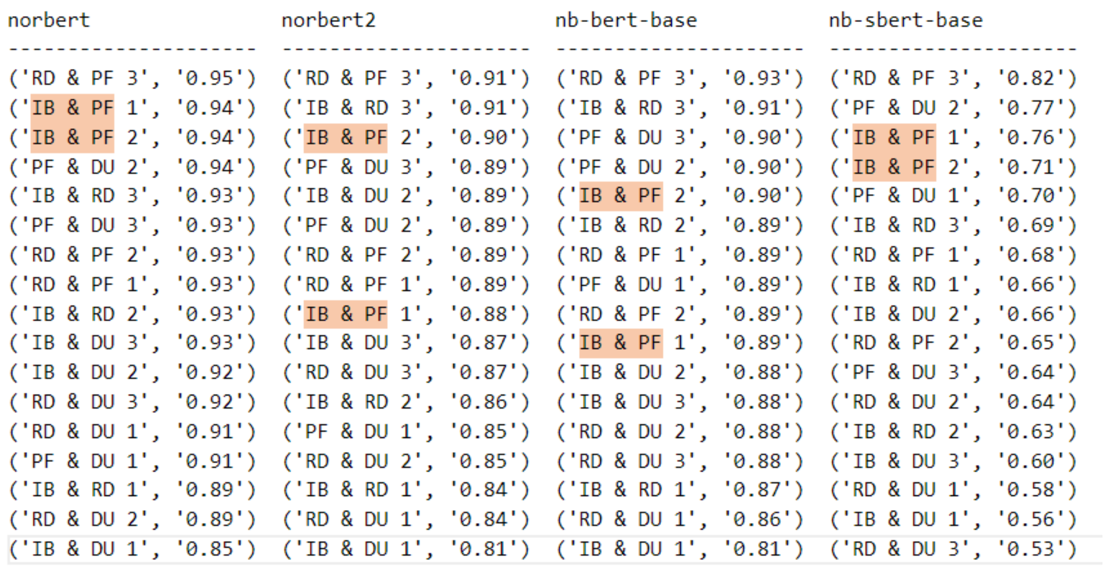
```
norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('IB & RD 1', '0.92')    ('IB & RD 1', '0.88')    ('IB & RD 1', '0.90')    ('IB & RD 1', '0.73')
('IB & RD 3', '0.91')    ('IB & RD 3', '0.87')    ('PF & DU 3', '0.89')    ('IB & DU 3', '0.71')
('RD & PF 1', '0.89')    ('RD & DU 3', '0.85')    ('IB & DU 3', '0.89')    ('IB & DU 2', '0.71')
('IB & RD 2', '0.89')    ('PF & DU 3', '0.85')    ('IB & PF 2', '0.87')    ('IB & PF 2', '0.70')
('PF & DU 3', '0.89')    ('RD & PF 1', '0.83')    ('RD & DU 3', '0.87')    ('PF & DU 3', '0.67')
('IB & PF 2', '0.88')    ('IB & DU 3', '0.83')    ('IB & RD 3', '0.87')    ('IB & RD 3', '0.65')
('RD & DU 3', '0.88')    ('PF & DU 1', '0.82')    ('RD & DU 2', '0.86')    ('IB & RD 2', '0.65')
('IB & DU 3', '0.88')    ('PF & DU 2', '0.82')    ('RD & PF 3', '0.86')    ('RD & DU 2', '0.65')
('RD & PF 3', '0.87')    ('RD & PF 3', '0.82')    ('PF & DU 2', '0.85')    ('RD & DU 1', '0.61')
('PF & DU 2', '0.87')    ('IB & PF 2', '0.81')    ('IB & RD 2', '0.84')    ('RD & PF 3', '0.61')
('RD & PF 2', '0.87')    ('RD & PF 2', '0.81')    ('RD & PF 1', '0.84')    ('RD & PF 2', '0.60')
('IB & DU 2', '0.87')    ('IB & PF 3', '0.81')    ('PF & DU 1', '0.83')    ('RD & DU 3', '0.59')
('PF & DU 1', '0.86')    ('IB & RD 2', '0.81')    ('RD & DU 1', '0.83')    ('RD & PF 1', '0.59')
('RD & DU 1', '0.86')    ('IB & DU 2', '0.80')    ('IB & DU 2', '0.83')    ('IB & PF 1', '0.57')
('RD & DU 2', '0.86')    ('RD & DU 1', '0.79')    ('RD & PF 2', '0.83')    ('IB & PF 3', '0.56')
('IB & PF 3', '0.85')    ('RD & DU 2', '0.79')    ('IB & PF 3', '0.81')    ('PF & DU 1', '0.53')
('IB & PF 1', '0.84')    ('IB & PF 1', '0.77')    ('IB & PF 1', '0.78')    ('PF & DU 2', '0.52')
```

Figure 5.3: Results of a comparison test with the Tarantino dataset, where the text being compared is "IB & DU 1" (Inglourious Basterds and Django Unchained) as explained in section 5.1.



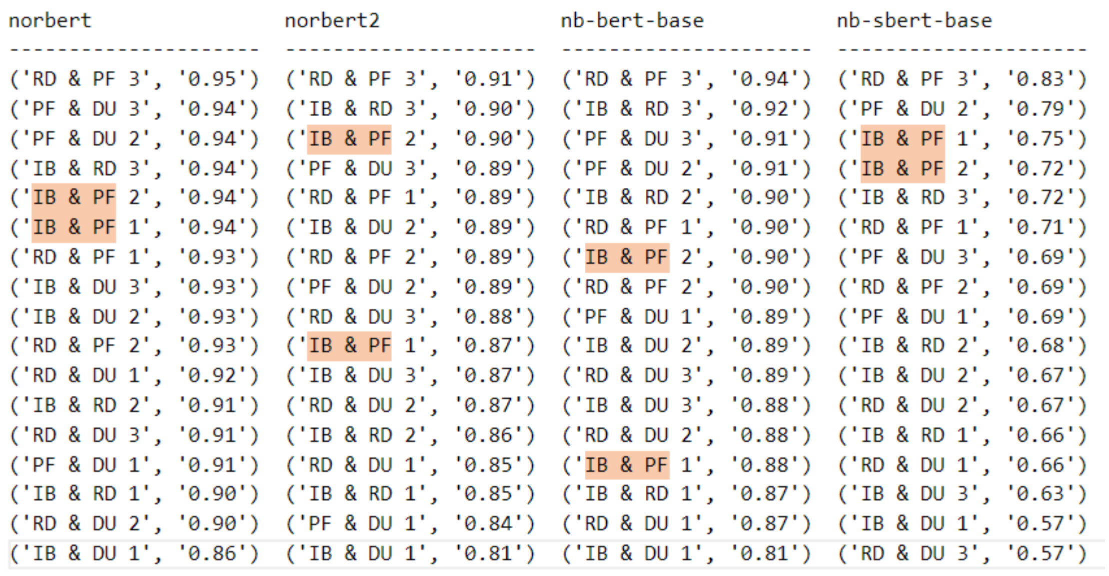
```
norbert                 norbert2                nb-bert-base            nb-sbert-base
--------------------    --------------------    --------------------    --------------------
('IB & RD 1', '0.92')   ('IB & RD 1', '0.88')   ('IB & RD 1', '0.90')   ('IB & RD 1', '0.73')
('IB & RD 3', '0.92')   ('IB & RD 3', '0.86')   ('PF & DU 3', '0.89')   ('IB & DU 3', '0.73')
('RD & PF 1', '0.90')   ('RD & DU 3', '0.85')   ('IB & DU 3', '0.89')   ('IB & DU 2', '0.71')
('PF & DU 3', '0.90')   ('PF & DU 3', '0.84')   ('IB & PF 2', '0.88')   ('IB & PF 2', '0.70')
('IB & DU 3', '0.89')   ('RD & PF 1', '0.84')   ('RD & DU 3', '0.87')   ('PF & DU 3', '0.68')
('IB & PF 2', '0.89')   ('IB & DU 3', '0.84')   ('IB & RD 3', '0.87')   ('IB & RD 3', '0.67')
('IB & DU 2', '0.88')   ('PF & DU 2', '0.82')   ('RD & DU 2', '0.86')   ('IB & RD 2', '0.66')
('RD & PF 3', '0.88')   ('IB & RD 2', '0.82')   ('RD & PF 3', '0.86')   ('RD & DU 2', '0.65')
('RD & PF 2', '0.88')   ('RD & PF 2', '0.82')   ('PF & DU 2', '0.85')   ('RD & DU 1', '0.64')
('RD & DU 1', '0.88')   ('IB & PF 3', '0.81')   ('IB & RD 2', '0.85')   ('RD & PF 2', '0.63')
('PF & DU 2', '0.88')   ('IB & PF 2', '0.81')   ('RD & PF 1', '0.84')   ('RD & PF 3', '0.61')
('IB & RD 2', '0.88')   ('IB & DU 2', '0.81')   ('PF & DU 1', '0.84')   ('RD & PF 1', '0.61')
('PF & DU 1', '0.88')   ('RD & DU 2', '0.80')   ('RD & PF 2', '0.83')   ('RD & DU 3', '0.59')
('RD & DU 3', '0.87')   ('RD & DU 1', '0.80')   ('RD & DU 1', '0.83')   ('IB & PF 1', '0.58')
('RD & DU 2', '0.87')   ('RD & PF 3', '0.80')   ('IB & DU 2', '0.83')   ('PF & DU 1', '0.57')
('IB & PF 1', '0.87')   ('PF & DU 1', '0.78')   ('IB & PF 3', '0.81')   ('IB & PF 3', '0.57')
('IB & PF 3', '0.86')   ('IB & PF 1', '0.76')   ('IB & PF 1', '0.79')   ('PF & DU 2', '0.53')
```

Figure 5.4: Results of a comparison test with the Tarantino dataset without titles, where the text being compared is "IB & DU 1" (Inglourious Basterds and Django Unchained) as explained in section 5.1.



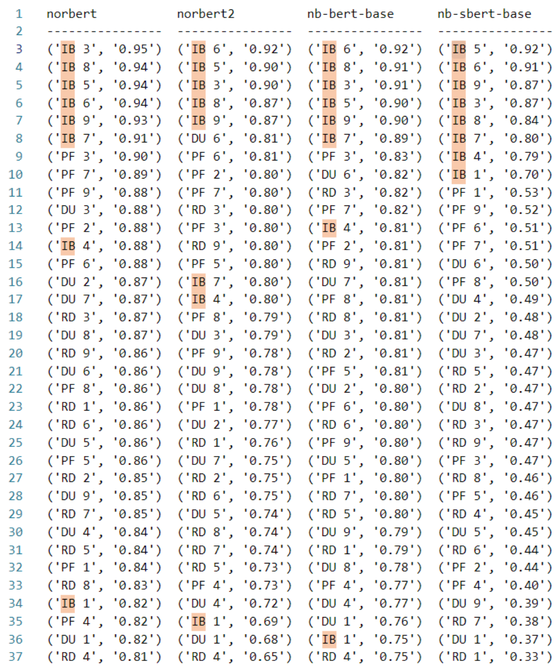
```
norbert                 norbert2                nb-bert-base            nb-sbert-base
--------------------    --------------------    --------------------    --------------------
('RD & PF 3', '0.95')   ('RD & PF 3', '0.91')   ('RD & PF 3', '0.93')   ('RD & PF 3', '0.82')
('IB & PF 1', '0.94')   ('IB & RD 3', '0.91')   ('IB & RD 3', '0.91')   ('PF & DU 2', '0.77')
('IB & PF 2', '0.94')   ('IB & PF 2', '0.90')   ('PF & DU 3', '0.90')   ('IB & PF 1', '0.76')
('PF & DU 2', '0.94')   ('PF & DU 3', '0.89')   ('PF & DU 2', '0.90')   ('IB & PF 2', '0.71')
('IB & RD 3', '0.93')   ('IB & DU 2', '0.89')   ('IB & PF 2', '0.90')   ('PF & DU 1', '0.70')
('PF & DU 3', '0.93')   ('PF & DU 2', '0.89')   ('IB & RD 2', '0.89')   ('IB & RD 3', '0.69')
('RD & PF 2', '0.93')   ('RD & PF 2', '0.89')   ('RD & PF 1', '0.89')   ('RD & PF 1', '0.68')
('RD & PF 1', '0.93')   ('RD & PF 1', '0.89')   ('PF & DU 1', '0.89')   ('IB & RD 1', '0.66')
('IB & RD 2', '0.93')   ('IB & PF 1', '0.88')   ('RD & PF 2', '0.89')   ('IB & DU 2', '0.66')
('IB & DU 3', '0.93')   ('IB & DU 3', '0.87')   ('IB & PF 1', '0.89')   ('RD & PF 2', '0.65')
('IB & DU 2', '0.92')   ('RD & DU 3', '0.87')   ('IB & DU 2', '0.88')   ('PF & DU 3', '0.64')
('RD & DU 3', '0.92')   ('IB & RD 2', '0.86')   ('IB & DU 3', '0.88')   ('RD & DU 2', '0.64')
('RD & DU 1', '0.91')   ('PF & DU 1', '0.85')   ('RD & DU 2', '0.88')   ('IB & RD 2', '0.63')
('PF & DU 1', '0.91')   ('RD & DU 2', '0.85')   ('RD & DU 3', '0.88')   ('IB & DU 3', '0.60')
('IB & RD 1', '0.89')   ('IB & RD 1', '0.84')   ('IB & RD 1', '0.87')   ('RD & DU 1', '0.58')
('RD & DU 2', '0.89')   ('RD & DU 1', '0.84')   ('RD & DU 1', '0.86')   ('IB & DU 1', '0.56')
('IB & DU 1', '0.85')   ('IB & DU 1', '0.81')   ('IB & DU 1', '0.81')   ('RD & DU 3', '0.53')
```

Figure 5.5: Results of a comparison test with the Tarantino dataset, where the text being compared is "IB & PF 3" (Inglourious Basterds and Pulp Fiction) as explained in section 5.1.



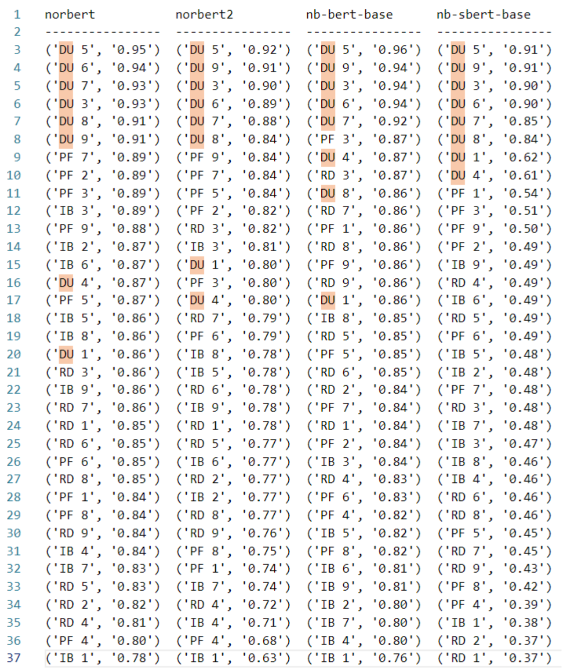
```
norbert                   norbert2                  nb-bert-base              nb-sbert-base
---------------------     ---------------------     ---------------------     ---------------------
('RD & PF 3', '0.95')     ('RD & PF 3', '0.91')     ('RD & PF 3', '0.94')     ('RD & PF 3', '0.83')
('PF & DU 3', '0.94')     ('IB & RD 3', '0.90')     ('IB & RD 3', '0.92')     ('PF & DU 2', '0.79')
('PF & DU 2', '0.94')     ('IB & PF 2', '0.90')     ('PF & DU 3', '0.91')     ('IB & PF 1', '0.75')
('IB & RD 3', '0.94')     ('PF & DU 3', '0.89')     ('PF & DU 2', '0.91')     ('IB & PF 2', '0.72')
('IB & PF 2', '0.94')     ('RD & PF 1', '0.89')     ('IB & RD 2', '0.90')     ('IB & RD 3', '0.72')
('IB & PF 1', '0.94')     ('IB & DU 2', '0.89')     ('RD & PF 1', '0.90')     ('RD & PF 1', '0.71')
('RD & PF 1', '0.93')     ('RD & PF 2', '0.89')     ('IB & PF 2', '0.90')     ('PF & DU 3', '0.69')
('IB & DU 3', '0.93')     ('PF & DU 2', '0.89')     ('RD & PF 2', '0.90')     ('RD & PF 2', '0.69')
('IB & DU 2', '0.93')     ('RD & DU 3', '0.88')     ('PF & DU 1', '0.89')     ('PF & DU 1', '0.69')
('RD & PF 2', '0.93')     ('IB & PF 1', '0.87')     ('IB & DU 2', '0.89')     ('IB & RD 2', '0.68')
('RD & DU 1', '0.92')     ('IB & DU 3', '0.87')     ('RD & DU 3', '0.89')     ('IB & DU 2', '0.67')
('IB & RD 2', '0.91')     ('RD & DU 2', '0.87')     ('IB & DU 3', '0.88')     ('RD & DU 2', '0.67')
('RD & DU 3', '0.91')     ('IB & RD 2', '0.86')     ('RD & DU 2', '0.88')     ('IB & RD 1', '0.66')
('PF & DU 1', '0.91')     ('RD & DU 1', '0.85')     ('IB & PF 1', '0.88')     ('RD & DU 1', '0.66')
('IB & RD 1', '0.90')     ('IB & RD 1', '0.85')     ('IB & RD 1', '0.87')     ('IB & DU 3', '0.63')
('RD & DU 2', '0.90')     ('PF & DU 1', '0.84')     ('RD & DU 1', '0.87')     ('IB & DU 1', '0.57')
('IB & DU 1', '0.86')     ('IB & DU 1', '0.81')     ('IB & DU 1', '0.81')     ('RD & DU 3', '0.57')
```

Figure 5.6: Results of a comparison test with the Tarantino dataset without titles, where the text being compared is "IB & PF 3" (Inglourious Basterds and Pulp Fiction) as explained in section 5.1.



```
norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('IB 3', '0.95')    ('IB 6', '0.92')    ('IB 6', '0.92')    ('IB 5', '0.92')
('IB 8', '0.94')    ('IB 5', '0.90')    ('IB 8', '0.91')    ('IB 6', '0.91')
('IB 5', '0.94')    ('IB 3', '0.90')    ('IB 3', '0.91')    ('IB 9', '0.87')
('IB 6', '0.94')    ('IB 8', '0.87')    ('IB 5', '0.90')    ('IB 3', '0.87')
('IB 9', '0.93')    ('IB 9', '0.87')    ('IB 9', '0.90')    ('IB 8', '0.84')
('IB 7', '0.91')    ('DU 6', '0.81')    ('IB 7', '0.89')    ('IB 7', '0.80')
('PF 3', '0.90')    ('PF 6', '0.81')    ('PF 3', '0.83')    ('IB 4', '0.79')
('PF 7', '0.89')    ('PF 2', '0.80')    ('DU 6', '0.82')    ('IB 1', '0.70')
('PF 9', '0.88')    ('PF 7', '0.80')    ('RD 3', '0.82')    ('PF 1', '0.53')
('DU 3', '0.88')    ('RD 3', '0.80')    ('PF 7', '0.82')    ('PF 9', '0.52')
('PF 2', '0.88')    ('PF 3', '0.80')    ('IB 4', '0.81')    ('PF 6', '0.51')
('IB 4', '0.88')    ('RD 9', '0.80')    ('PF 2', '0.81')    ('PF 7', '0.51')
('PF 6', '0.88')    ('PF 5', '0.80')    ('RD 9', '0.81')    ('DU 6', '0.50')
('DU 2', '0.87')    ('IB 7', '0.80')    ('DU 7', '0.81')    ('PF 8', '0.50')
('DU 7', '0.87')    ('IB 4', '0.80')    ('PF 8', '0.81')    ('DU 4', '0.49')
('RD 3', '0.87')    ('PF 8', '0.79')    ('RD 8', '0.81')    ('DU 2', '0.48')
('DU 8', '0.87')    ('DU 3', '0.79')    ('DU 3', '0.81')    ('DU 7', '0.48')
('RD 9', '0.86')    ('PF 9', '0.78')    ('RD 2', '0.81')    ('DU 3', '0.47')
('DU 6', '0.86')    ('DU 9', '0.78')    ('PF 5', '0.81')    ('RD 5', '0.47')
('PF 8', '0.86')    ('DU 8', '0.78')    ('DU 2', '0.80')    ('RD 2', '0.47')
('RD 1', '0.86')    ('PF 1', '0.78')    ('PF 6', '0.80')    ('DU 8', '0.47')
('RD 6', '0.86')    ('DU 2', '0.77')    ('RD 6', '0.80')    ('RD 3', '0.47')
('DU 5', '0.86')    ('RD 1', '0.76')    ('PF 9', '0.80')    ('RD 9', '0.47')
('PF 5', '0.86')    ('DU 7', '0.75')    ('DU 5', '0.80')    ('PF 3', '0.47')
('RD 2', '0.85')    ('RD 2', '0.75')    ('PF 1', '0.80')    ('RD 8', '0.46')
('DU 9', '0.85')    ('RD 6', '0.75')    ('RD 7', '0.80')    ('PF 5', '0.46')
('RD 7', '0.85')    ('DU 5', '0.74')    ('RD 5', '0.80')    ('RD 4', '0.45')
('DU 4', '0.84')    ('RD 8', '0.74')    ('DU 9', '0.79')    ('DU 5', '0.45')
('RD 5', '0.84')    ('RD 7', '0.74')    ('RD 1', '0.79')    ('RD 6', '0.44')
('PF 1', '0.84')    ('RD 5', '0.73')    ('DU 8', '0.78')    ('PF 2', '0.44')
('RD 8', '0.83')    ('PF 4', '0.73')    ('PF 4', '0.77')    ('PF 4', '0.40')
('IB 1', '0.82')    ('DU 4', '0.72')    ('DU 4', '0.77')    ('DU 9', '0.39')
('PF 4', '0.82')    ('IB 1', '0.69')    ('DU 1', '0.76')    ('RD 7', '0.38')
('DU 1', '0.82')    ('DU 1', '0.68')    ('IB 1', '0.75')    ('DU 1', '0.37')
('RD 4', '0.81')    ('RD 4', '0.65')    ('RD 4', '0.75')    ('RD 1', '0.33')
```

Figure 5.7: Results of a comparison test between the single movie descriptions from the Tarantino movies, as explained in section 5.1. The description compared to is "IB 2".



```
norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('DU 5', '0.95')    ('DU 5', '0.92')    ('DU 5', '0.96')    ('DU 5', '0.91')
('DU 6', '0.94')    ('DU 9', '0.91')    ('DU 9', '0.94')    ('DU 9', '0.91')
('DU 7', '0.93')    ('DU 3', '0.90')    ('DU 3', '0.94')    ('DU 3', '0.90')
('DU 3', '0.93')    ('DU 6', '0.89')    ('DU 6', '0.94')    ('DU 6', '0.90')
('DU 8', '0.91')    ('DU 7', '0.88')    ('DU 7', '0.92')    ('DU 7', '0.85')
('DU 9', '0.91')    ('DU 8', '0.84')    ('PF 3', '0.87')    ('DU 8', '0.84')
('PF 7', '0.89')    ('PF 9', '0.84')    ('DU 4', '0.87')    ('DU 1', '0.62')
('PF 2', '0.89')    ('PF 7', '0.84')    ('RD 3', '0.87')    ('DU 4', '0.61')
('PF 3', '0.89')    ('PF 5', '0.84')    ('DU 8', '0.86')    ('PF 1', '0.54')
('IB 3', '0.89')    ('PF 2', '0.82')    ('RD 7', '0.86')    ('PF 3', '0.51')
('PF 9', '0.88')    ('RD 3', '0.82')    ('PF 1', '0.86')    ('PF 9', '0.50')
('IB 2', '0.87')    ('IB 3', '0.81')    ('RD 8', '0.86')    ('PF 2', '0.49')
('IB 6', '0.87')    ('DU 1', '0.80')    ('PF 9', '0.86')    ('IB 9', '0.49')
('DU 4', '0.87')    ('PF 3', '0.80')    ('RD 9', '0.86')    ('RD 4', '0.49')
('PF 5', '0.87')    ('DU 4', '0.80')    ('DU 1', '0.86')    ('IB 6', '0.49')
('IB 5', '0.86')    ('RD 7', '0.79')    ('IB 8', '0.85')    ('RD 5', '0.49')
('IB 8', '0.86')    ('PF 6', '0.79')    ('RD 5', '0.85')    ('PF 6', '0.49')
('DU 1', '0.86')    ('IB 8', '0.78')    ('PF 5', '0.85')    ('IB 5', '0.48')
('RD 3', '0.86')    ('IB 5', '0.78')    ('RD 6', '0.85')    ('IB 2', '0.48')
('IB 9', '0.86')    ('RD 6', '0.78')    ('RD 2', '0.84')    ('PF 7', '0.48')
('RD 7', '0.86')    ('IB 9', '0.78')    ('PF 7', '0.84')    ('RD 3', '0.48')
('RD 1', '0.85')    ('RD 1', '0.78')    ('RD 1', '0.84')    ('IB 7', '0.48')
('RD 6', '0.85')    ('RD 5', '0.77')    ('PF 2', '0.84')    ('IB 3', '0.47')
('PF 6', '0.85')    ('IB 6', '0.77')    ('IB 3', '0.84')    ('IB 8', '0.46')
('RD 8', '0.85')    ('RD 2', '0.77')    ('RD 4', '0.83')    ('IB 4', '0.46')
('PF 1', '0.84')    ('IB 2', '0.77')    ('PF 6', '0.83')    ('RD 6', '0.46')
('PF 8', '0.84')    ('RD 8', '0.77')    ('PF 4', '0.82')    ('RD 8', '0.46')
('RD 9', '0.84')    ('RD 9', '0.76')    ('IB 5', '0.82')    ('PF 5', '0.45')
('IB 4', '0.84')    ('PF 8', '0.75')    ('PF 8', '0.82')    ('RD 7', '0.45')
('IB 7', '0.83')    ('PF 1', '0.74')    ('IB 6', '0.81')    ('RD 9', '0.43')
('RD 5', '0.83')    ('IB 7', '0.74')    ('IB 9', '0.81')    ('PF 8', '0.42')
('RD 2', '0.82')    ('RD 4', '0.72')    ('IB 2', '0.80')    ('PF 4', '0.39')
('RD 4', '0.81')    ('IB 4', '0.71')    ('IB 7', '0.80')    ('IB 1', '0.38')
('PF 4', '0.80')    ('PF 4', '0.68')    ('IB 4', '0.80')    ('RD 2', '0.37')
('IB 1', '0.78')    ('IB 1', '0.63')    ('IB 1', '0.76')    ('RD 1', '0.37')
```

Figure 5.8: Results of a comparison test between the single movie descriptions from the Tarantino movies, as explained in section 5.1. The description compared to is "DU 2".

5.1.3 Tests on graded datasets

The web development dataset obtained contained a graded score for each answer. The graded score was usually between 0.0 and 5.0. This allowed tests of the comparison algorithm based on the graded scores. For the sake of this test, it is assumed that the higher similarity score given between two answers, the lower the difference in grade should be. There are four main outcomes that the similarity score and difference in grade could have a tendency towards:

1. High similarity score and low difference in grade
2. High similarity score and high difference in grade
3. Low similarity score and low difference in grade
4. Low similarity score and high difference in grade

1. and 4. should be rewarded as outcomes. If the similarity score is high, the answers contain much of the same content and should have a similar grade. If the similarity score is low, the answers contain content that is very different and should therefore have a higher difference in grade. Similarly, 2. and 3. should be punished as they display the opposite outcomes. The proposed measurement Similarity-Grade Correlation Coefficient (SGCC) calculates how well the similarity scores correspond with the graded scores. It is calculated as one numerical value between 0.0 and 1.0 by the following formula:

$$\text{SGCC} = \left| \frac{\text{Difference in graded score}}{\text{Max graded score}} - \text{Similarity score} \right| \tag{5.1}$$

The SGCC between two answers is calculated by subtracting the similarity score they share from the normalized difference in their graded score and measuring the absolute value of the result.

As displayed in the results in subsection 5.1.1, the different BERT models often vary in their scale. NorBERT 1 often gives a higher similarity score on average than the others and NB-SBERT often gives lower, but more spread out scores. To test the SGCC on the models without the differences in scale, a normalized similarity score is calculated by the following formula:

$$\text{Normalized sim score} = \frac{\text{Sim score} - \text{Min sim score}}{\text{Max sim score} - \text{Min sim score}} \tag{5.2}$$

The normalized similarity score is calculated by subtracting the minimum similarity score given by the BERT model used from both the similarity score in question and the max similarity score, followed by calculating the ratio between the similarity score in question and the maximum similarity score possible.

The normalized similarity score replaces the similarity score in Equation 5.1 and results in a normalized SGCC, which conveys the accuracy of the similarity score given. SGCC is meant to determine how well a model works, and the calculations were therefore made between every answer to every question in the exam dataset. The following formula shows how the SGCC were accumulated:

$$\text{Accumulated SGCC} = \sum_{n=1}^{|questions|} \sum_{m=1}^{|answers|} \sum_{l=1}^{|comparisons|} \text{SGCC}_{n_{m_l}} \tag{5.3}$$

The accumulated SGCC was calculated by adding up all coefficients for the similarity scores between answers for every question.

After the accumulated SGCC is calculated, it is also averaged by dividing by the amount of coefficients added together. In addition, since three exam datasets from the web development course were collected, the SGCC was calculated for each of them and then averaged. The results from these tests are shown in Table 5.1.

Acc. SGCC	NorBERT 1	NorBERT 2	NB-BERT-Base	NB-SBERT-Base
Natural order	0.17	0.15	0.15	0.14
Normalized	0.16	0.15	0.15	0.13

Table 5.1: Accumulated SGCC for both the natural order of every model and their normalized variant.

5.2 User tests

In order to judge the legitimacy of TGRT, it needed to be tested on examiners with a set of previously graded exams from one of their courses. Many examiners and lecturers were contacted and invited to test the tool. This ended up being the biggest challenge of the project. During development, it was a struggle to collect datasets for testing the comparison algorithm, and it would be an even bigger struggle to persuade people to test it. The research questions concerned what model would be the best to use for comparing exam answers, how to display to the examiner why the model deemed the answers to be similar, and how the examiners themselves would perceive such a tool. The user tests would in essence test all of them. If the test subjects disagreed with the recommendations because the answers differed in content, then the comparison algorithm would have failed at its task. If the test subjects struggled to understand why the algorithm recommended what it did, the supporting features implemented would have failed in communicating the nuances of each answer and their similarity. The overall response of the test subjects would answer the third research question. Would they find the tool helpful or insulting in pointing out mistakes in their grading?

The user tests started with an introduction to the project, what had been developed and how the test would be conducted. Surveys were held both before and after the main testing of TGRT. The intention of the first survey was to get an introduction of the user, with their professional information, background on the exam they were going to use for the test and their experience with AI. An important part of the survey was to get an impression of their preconceived notions on how useful they believed the tool could be, to contrast with their perceptions after the test. The questions were as follows:

1. Who are you and what courses have you graded earlier?
2. From which course are the exam answers that are about to be tested?
3. What do you think of the concept of the product, and does it seem like something you could use?
4. Do you think this tool is suitable for censoring answers in your field? Are there any other fields of study where you can imagine it could be of greater use?
5. Do you have any thoughts about the use of AI in censorship in the school system?
6. Do you have any experiences with the use of AI in the past?

While one of the researchers held the survey, the other set up the server and website for the upcoming test. The website was hosted on NTNU's internet, so the test subject could navigate to the hosted IP address to connect. The following assignment was given to the test subject: "Upload the answers and find out if any of them should have their grades adjusted up or down". The assignment itself was very simple, but it would test the tool's capabilities and encompass all the different aspects of it. Follow-up questions in the survey held after the main test was concluded would also allow the test subject to elaborate on their experience. The tests were meant to be very informal, meaning that the test subject discussed and commented as the test went on. The main purpose of the test was not to test the capabilities of the user interface and if the system was intuitive on its own, but rather to test the algorithms' judgement and ability to convey why it outputs the results it does. Because of this, the researchers did not shy away from explaining the interface and helping the test subject understand the layout if any confusion would occur. During the practical test, one of the researchers communicated with the test subject while the other took notes and kept a lookout on the server status.

The second survey was meant to question their experience of the tool and how it affected their views on assisted grading. The questions were as follows:

1. What is your immediate reaction after using the product?
2. Was it easy to get an overview of the recommended grade changes?
3. Do you agree with any of the recommended grade changes?
   (a) How did you experience discovering discrepancies between the scores of two answers deemed to be similar?
4. Were there any recommendations you completely disagreed with?
   (a) Do you understand why the tool recommended what it did, even if you did not agree?
   (b) Does this lessen your overall impression of the product?
   (c) Do you have less faith in what the tool recommends because of this?
5. Which of the supporting features did you find most helpful in understanding the similarities?
6. Were there any that you did not find very helpful?
7. Are there any supporting features you miss that could make the similarities easier to understand?
8. Is there anything else you are missing that would have made the process easier or more useful?
9. Is this a tool you would consider using?
10. Did your perception of the idea of using AI in the censorship process change?
11. Do you see any limitations with this type of tool?
12. Do you have any other feedback?

The tests were designed with the ability to perform them anywhere on NTNU's campus (they could also have been performed elsewhere by the means of a Virtual Private Network (VPN)). The tests were performed physically in the test subjects' office or office space in roughly an hours' time.

5.2.1 First user test results

The first user test was conducted with a lecturer from NTNU's Faculty of Medicine and Health Sciences. Prior to the test, she had a positive inclination towards the concept, as she thought it was difficult to be sure whether or not she was grading her exams fairly throughout the entire examination process. She was also positively inclined towards the adoption of AI tools in the aiding of exam grading.

The execution of the test did not go quite as planned, since an error in the program prevented TF-IDF and RAKE terms from being retrieved and displayed. Thus, the only supporting features she had at her disposal were the OpenAI data. Nonetheless, she was quite satisfied with the product. She thought that the structuring and displaying of the recommendations were intuitive. Out of the five recommendations for adjusting the grade made by the product that she examined in detail, she agreed with three of them. While she did disagree with two of the recommendations, she said that it was understandable that the AI did not understand all the nuances of the answers, and that this did not diminish her overall impression of the product. One of the recommendations she disagreed with had a comparison answer that was overall quite similar, but contained small differences that made the entire answer better at answering the question. For the second of the recommendations she disagreed with she did not understand why the answers were considered to be similar.

As for the supporting features, she found the summaries generated by OpenAI very useful. However, she did not find the OpenAI-generated themes and keywords quite as useful. She remarked that the themes tended to just replicate the main points contained in the assignment text, while the keywords were missing context, making it difficult to assess whether the text answered the question in a satisfactory way based on the keywords alone.

She did have a few suggestions for improvements of the product. The main one was that she thought it would be useful to integrate the sensor's proposed solution into the website, so that one could easily compare the students' answers to this. Also, when examining a specific comparison answer, she became interested in reading the comparison answers of that answer. However, she did not find this answer in the leftmost bar where the answers that are recommended to be regraded are found. Finally, she remarked that some exam questions have multiple subquestions and thus multiple answers, and these answers are contained in the same text string. Having several different answers in the same string could potentially confuse the AI, as there is no way for the system to discern exactly when one answer ends and the next begins. She conceded this, and suggested that if TGRT were to be put to use, the exams would have to be structured differently.

5.2.2 Second user test results

The second user test was performed with a research fellow at NTNU's Department of Computer Science. Because of unforeseen problems that occurred during the execution of the test, it had to be postponed twice, so that it wasn't until the third meeting that a proper test was actually done. Before attempting to test the website, he said that he thought it seemed like a useful tool and that there are many subconscious biases which might affect an examiner when grading an exam. He was unsure if his exams were optimal for testing the tool, as 60 % of them consisted of programming tasks. However, he was interested in seeing if TGRT could compare and find similarities between different pieces of code. The exams also contained several questions where the answers were shorter texts, which would probably be more useful for the purposes of this test, as TGRT was developed with text in mind. When asked about his opinions on the use of AI in the grading of exams, he said that he was positively inclined towards tools that can make suggestions, but that there should always be a human element behind the grading.

When starting the test in the first meeting, there were immediate problems in the back end that prevented any data from being sent back to the client. These problems were all related to the processing of the JSON file containing the dataset of exam submissions. The dataset that was tested on included several multiple choice questions, which meant that when the server was processing the data, it looked for certain keys in the JSON file that did not exist for those questions. This lead to an error, and the whole processing stopped. There were several other issues as well. If a student had submitted a blank exam, their list of questions was empty, which meant that the server tried to access a non-existent element in the list. In addition, some unanswered questions consisted of an undefined variable instead of an empty string, which was problematic because the code only filtered out answers with empty strings at that time. There were attempts to fix some of the problems on the spot, but every fixed issue lead to another one arise. The test subject tried writing a quick script for filtering out the undesirable answers before sending the data to the server, but to no avail. As the code needed a bigger rewrite, it was decided to schedule a second test at a later date when the problems were fixed.

During the second test, no errors occurred when processing the data, and the output from the server was displayed on the website. However, the answers from the first question were quite strange; they all started with "simpleChoice" followed by a seemingly random string of characters. It turned out that these were the answers to a multiple choice question, and that the reason the answers were processed was that during the grading process the examiner noticed that several of the choices were correct answers to the question. Thus, they had to go back and manually grade the answers that were answered correctly by the new standards. This led to these answers being registered as manually scored, and were thus interpreted as text answers by the code.

Another issue that occurred was that many of the answers that were displayed as having been given zero points had not actually been graded at all. It was previously assumed that when an answer had not received a manual score, it meant that the answer was graded with a score of zero (which was often the case). Therefore, TGRT was programmed to automatically assign it with a score of zero. For some unknown reason, quite a lot of answers had not been given a score by the examiner, and were thus shown as having zero points. This lead to some confusion, as there were several instances of pairs of answers with very similar content where one of them had ten points and the other zero. The test subject remarked that the answer with zero points deserved to be scored higher, and it seemed that the product was very good at pointing out errors in the grading process. It was not until after this had happened a few times that it was discovered that the answers with zero points had not actually been given zero points at all.

Yet another issue with the dataset was that there often was a mismatch between the question number in the JSON file and the actual correct question number. This led to many answers showing up under the wrong question tab, and answers to different questions were compared. This is a fault of the dataset and how it is exported by Inspera. As far as what could be tested despite all the aforementioned problems, it was evident that the NB-SBERT model was quite adept at finding similar code answers (as was the case with the comparisons with the ungraded answers described above). The test subject also found a relevant comparison between an answer that was given 10 points, and another that was given 8 points, where he remarked that both answers could have been given 10 points. However, because of all the problems that occurred, it was decided to perform yet another test after ironing out the new issues that appeared with the dataset processor.

The third and final test went a lot smoother than the first two. The processing and displaying of data worked without any issues, and there were no instances of ungraded answers that were automatically given 0 points. However, there still were several multiple choice questions that had not been filtered out. It is hard to understand fully why they were retrieved without testing further, but it could be because of leading white space in the multiple choice answer text, which the dataset reader did not account for. This time the questions were however correctly sorted by the question number and were therefore easy to dismiss. The test subject agreed with some recommendations and disagreed with others. He found several instances of identical or nearly identical answers to the same question (for a programming task) where the answers were graded differently.

Because the exam mainly consisted of programming tasks and most of the answers were quite short, the test subject found limited use in the supporting features. He found that he could often get a better overview of the answers by skimming them than by reading the keywords extracted by the TF-IDF, RAKE and OpenAI components. He did however find the summaries generated by OpenAI somewhat useful, since they were surprisingly good at conveying the semantic meaning of the code in the answers. He specified that the summaries should not be trusted blindly, since they are generative and thus consist of different words and sentences than the original answers. He also acknowledged that the supporting features would be more useful for longer answers.

Overall, he found that TGRT was good at pointing out which direction the grades should be adjusted, but there were a few cases where he thought that one of the similarly compared answers should be adjusted in the opposite direction instead of the recommended change. Although he disagreed with several of the recommendations, he understood why the model considered them to be similar. He said that this did not diminish his overall impression of the product, but that it probably would if he did not understand how AI models work. He emphasized that if the product was to be put to use, it should come with a disclaimer that specifies that this is not an automatic grading tool, and that no recommendations should be trusted blindly.

When asked about how intuitive he thought the website was, he responded that most things were well explained and easy to understand except the similarity and certainty score. It was not clear from the beginning on what the numbers meant, but he said he gained a better understanding as the test went on. He had several suggestions for improvement of the user interface, such as navigating between answers using the arrow keys, and toggling between stacking two answers on top of each other and having them side by side. While he acknowledged that he knew that the tool was not meant to be used for programming tasks, he suggested that if it was to be expanded for this use, it should include syntax highlighting and color schemes for code snippets that are either unique or found in both answers. Overall, he was pleased with the product and stated that it confirmed his view that AI-assisted grading will become increasingly prominent in the future.

CHAPTER

SIX

DISCUSSION

The proposed TGRT has been tested in primarily two different ways. The similarity algorithm has been tested with several different Norwegian BERT models, and both this algorithm and the validity of the methods included in TGRT have been tested with user tests. The tests were held to answer the research questions introduced in section 1.2:

- **RQ1: How can the similarities between exam answers be calculated?**
- **RQ2: What features can be extracted for assisting the examiners in analyzing similar answers?**
- **RQ3: How will a tool for detecting inconsistencies in grading be perceived by examiners?**

6.1 RQ1 Comparison algorithm

RQ1 has mostly been researched by the similarity algorithm tests. How do the different models compete when accounting for different aspects of texts that are found in real answer sets? Will texts of similar lengths tend to be rated similar? How will spelling mistakes influence the results? How accurately will the models rank texts with varying degrees of thematic similarity? The results from each model analyzed with respect to the different aspects will enlighten what model succeeds above the rest at the given task. The models tested in this manner are NB-BERT-Base, NB-SBERT-Base, NorBERT 1 and NorBERT 2 as explained in section 4.1.

6.1.1 Friction dataset

By testing on the friction dataset, many aspects of the comparison algorithm were tested. The variants made it possible to see how the models interpreted and compared texts of different length, spelling errors and texts where parts of the answer were missing. Both the tests use an answer which was answered correctly and with correct spelling as comparison base.

Looking at the first test results (Figure 5.1), the NB-SBERT model has all the answers marked as "Good" on top, except for an outlier with one marked "Static" also listed together with them. The outlier might be an unwanted effect caused by the limitations of the dataset. In order to test the aspects previously mentioned as separately as possible, the different answers were made to be literal variants of the original answer. This means that the static version is just a trimmed down version of the original answer with the kinetic portions edited out. This means that the answer marked "Static" with a high similarity score for NB-SBERT contains a lot of the exact same text as the base answer being compared. This is also one reason why all the answers except for the irrelevant answers have a high similarity score. Along with containing information about the same subject, many of the answers contain the same language and words, meaning they naturally would be close in similarity score. The tests on this dataset will not display how a comparative tool calculates the similarity score between answers naturally answered to a question, but rather show how the different aspects in the variants influence the comparison algorithm in positive or negative ways.

The next sections will analyze the distribution of the many aspects tested by the friction dataset. Most of the aspects tested do not necessarily have an expected outcome that should be met in order for the comparison tool to be accurate. If the model deems texts that have too much non-relevant information to be not that similar to an optimal answer, it is not necessarily wrong, but rather enlightens an aspect of how the algorithm works. The tables show similarity scores for the two main tests that were performed. The first test had answer "1, Good, normal" as the base answer to compare to, while the second used "2, Good, normal".

6.1.1.1 Synonyms variant

Test	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
Test 1	0.97	0.96	0.95	0.94
Test 2	0.98	0.98	0.98	0.96

Table 6.1: Similarity scores of the synonyms variant of the answers labeled "Good, normal".

In the first test, all the models have the same answer on the top of their list, namely "1, Good, synonyms". The synonyms variants should be further up the list as they should contain the same contents as the original text, only written in other words. The first test uses the answer "1, Good, normal" as base comparative answer, and naturally has its synonym answer at the top. Further along the list one can also see that the second answer "2, Good, normal" is often quite similar in score to its synonyms variant as well. It is not always at the top of the list compared to the base comparative answer, but that is because the contents of the second answer is not completely equal to the first. The same will be true for the synonyms variant of the second answer.

In the second test the same general distribution is apparent, but the list is inhabited more by other variants in the top sections of the list. When looking at the scores, however, the same general result is present where the synonyms variant is ranked close to its normal counterpart.

6.1.1.2 Typos variant

Test	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
Test 1	0.96	0.90	0.94	0.86
Test 2	0.92	0.85	0.91	0.94

Table 6.2: Similarity scores of the typos variant of the answers labeled "Good, normal".

For the "typos" variants of the answers, words like "Friksjon" were changed to "Friskjon", removing the comprehensive meaning of the word. The models are pre-trained on the Norwegian language and should therefore register two texts with similar structure, but words spelled differently as dissimilar. The results show something different, however. The typo variants generally score quite high in similarity, with the lowest being a 0.85. This would suggest that the models account for spelling errors and can derive meaning from words by looking at the context of the sentence as a whole. This would lead to the deprioritization of including spelling correction as a part of the algorithm, which was something that was considered early on. This is discussed more in subsection 4.6.3.

6.1.1.3 Long variant

Test	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
Test 1	0.96	0.93	0.93	0.88
Test 2	0.97	0.96	0.97	0.94

Table 6.3: Similarity scores of the long variant of the answers labeled "Good, normal".

The long variants generally have a pretty high similarity score. The long variants are just elongated versions of the original texts and contain all the original words. The difference is that they have additional examples and complementary details not really needed to answer the question. The results show that the models extract the meaning despite all the extra details added and compare quite preferably to the original.

6.1.1.4 Short variant

Test	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
Test 1	0.89	0.77	0.85	0.85
Test 2	0.92	0.86	0.91	0.89

Table 6.4: Similarity scores of the short variant of the answers labeled "Good, normal".

The results for the shorter answers were a bit more sporadic. The second version of NorBERT especially struggles to determine the similarity between the original and the shortened version, but all the models generally rank this variant lower than the others. The shortened texts are often rewritten from the original answer to compress it to as few words as possible, meaning the sentence structure is different from the original answers. The models should examine the context and contents of the texts, and the results should display an understanding of the material. The shortened answers are however a bit too short and do not explain the concepts of friction as well as the other answers. This is reflected in the similarity scores. The models only had a little drop in similarity scores when extra information was added in the long variants, but when information was retracted, the results showed a noticeable drop in similarity.

6.1.1.5 Static and Kinetic variants

Test	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
Test 1	0.92	0.79	0.91	0.90
Test 2	0.91	0.98	0.98	0.98

Table 6.5: Similarity scores of the static variant of the answers labeled "Good, normal".

Test	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
Test 1	0.94	0.92	0.90	0.75
Test 2	0.91	0.84	0.90	0.76

Table 6.6: Similarity scores of the kinetic variant of the answers labeled "Good, normal".

Similarly to the short variants, the scores of the static and kinetic variants are preferred to have a noticeable drop in similarity, as they too are trimmed down versions of the original answer with parts of the answer missing. The results are however a bit mixed. The results for the kinetic variants shows that NB-SBERT is able to understand that a good chunk of context is missing, whereas the results for the static variants show a high similarity across the board for all the models except the second version of NorBERT. The first test outputs similarity scores between 0.90 to 0.92 for all models apart from NorBERT 2 which gives 0.79. The second test is similar with 0.98 in similarity score for all, apart from NorBERT 1 which gives 0.91. These results could be because of how the answers were written. The original answer was split into two parts of static and kinetic, with a larger portion dedicated to explaining static friction than kinetic friction. Because of this, it makes sense that the tests for static variants outputs high similarity between itself and the original answer. The kinetic versions have varied results, but lower similarity scores compared to the static versions.

6.1.2 Irrelevant variant

Answer	Test	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
1	Test 1	0.78	0.65	0.70	0.37
	Test 2	0.74	0.63	0.68	0.30
2	Test 1	0.76	0.65	0.65	0.22
	Test 2	0.76	0.61	0.64	0.19

Table 6.7: Similarity scores of the irrelevant variants of the answers labeled "Good, normal".

The irrelevant variants are the only ones that should definitely be as low in similarity score as possible, since the contents have no correlation to the base answer. Across the tests, the irrelevant answers were very low on the results list, but NorBERT 1, NorBERT 2 and NB-BERT-Base still graded them fairly high with scores between 0.78 and 0.61. NorBERT 2 even ranks a shortened kinetic friction answer as lower than both irrelevant answers, but NorBERT 1 stands out for ranking the irrelevant answers highest when examining similarity scores. NB-SBERT is the only one that outputs significantly lower scores, which conveys that the texts are mostly dissimilar, with scores between 0.37 and 0.19.

6.1.3 Good variants

Answer	Test	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
1	Test 1	-	-	-	-
	Test 2	0.92	0.89	0.90	0.88
2	Test 1	0.92	0.89	0.90	0.88
	Test 2	-	-	-	-
3	Test 1	0.93	0.86	0.94	0.91
	Test 2	0.90	0.84	0.90	0.87
4	Test 1	0.89	0.78	0.91	0.86
	Test 2	0.89	0.82	0.91	0.85

Table 6.8: Similarity scores of good normal variants to the answers labeled "Good, normal". (Test 1 for answer 1 and test 2 for answer 2 were removed, as the answers were compared to themselves, which would have been 1.0)

The variants marked "Good, normal" are supposed to be well written and answer the question most accurately. Because of this, they should preferably have a high similarity score compared to many of the other variants. The resulting similarity scores are however a bit mixed. Ranging from 0.78 to 0.94, the results are not exemplary, with many other variants ranked above them. This might be caused by the other variants often containing many similar words and using a similar sentence structure to the compared answer, while the other good normal answers are answered independently of each other.

The tests on the friction dataset were not mainly meant to test the capabilities of the comparison algorithm to compare similar texts, but rather to test and analyze how the algorithm ranks different aspects like length of text, synonyms and spelling mistakes. The four answers that answered the question well were quite different and did not strive to be similar to each other. They were written to answer the question properly, but there are many ways to answer a question well. As will be discussed in subsection 4.6.1, to get a good graded score, you do not need to write the exact same content. This is possibly why the results for the good answers are as indecisive.

Looking at the category "Good, normal", no decisive trends were found, but when looking at the variants marked as "Good" in contrast to "Kinetic" and "Static", there is somewhat of a pattern. As seen in Figure 5.1 NorBERT, NorBERT 2 and NB-BERT-Base, the answers marked as "Good" with all of their variants have a tendency to float to the top of the list, but sporadically spread further down as well. NB-SBERT on the other hand, has all answers marked as "Good" concentrated at the top of their list. On the second test, NB-SBERT has the good answers more spread, but are generally further up than down. In this case, NorBERT version 1 and NB-BERT-Base have a more concentrated section of the good variants further up the list.

6.1.4 Tarantino dataset

The Tarantino dataset was created for the purpose of finding out how good the different BERT variants were at finding similarities between texts with a somewhat wider thematic range than the friction dataset. As explained in subsection 4.2.2, each text contained a short description of two Quentin Tarantino films (out of a pool of four specific films), with three texts for each specific combination of two films. Ideally, when comparing a text with all the others, the two texts that have the highest similarity score should be the two other texts in the dataset containing descriptions of the same two films.

The results of comparing the text "IB & DU 1" (short for Inglourious Basterds and Django Unchained) using all four BERT variants can be seen in Figure 5.3. Ideally, "IB & DU 2" and "IB & DU 3" should be ranked the highest. As can be seen from the table, the model that ranks these two answer the highest is NB-SBERT by a large margin (second and third place). All the other models rank "IB & DU 2" well within the lower half of their list, while "IB & DU 3" ranges from being ranked third (NB-BERT-BASE) and eighth (NorBERT 1). Interestingly, every model ranks the same text as number one, namely "IB & RD 1" (short for Inglourious Basterds and Reservoir Dogs), which is most likely because of similar descriptions of Inglourious Basterds contained in both texts.

As mentioned in subsection 4.2.2, a dataset that was identical to the original

Tarantino dataset was created, except that all instances of the films' names were replaced. This was supposed to challenge the models by making it harder for them to distinguish between the different descriptions. As can be seen in Figure 5.4, this actually caused NorBERT 1 and NorBERT 2 to rank the relevant texts slightly higher than they did when the names were present, whereas NB-BERT-BASE ranked "IB & DU 2" one place lower than before, and NB-SBERT ranked the two relevant texts the same as with the names present. All the models ranked "IB & RD 1" the highest in this instance as well, with the exact same similarity scores as those in Figure 5.3. The reason for the similarity scores being the same in the two cases is that neither of the texts ("IB & DU 1" and "IB & RD 1") actually use the films' titles at all in the original dataset, so there was no difference when comparing the two versions of the text. As for why "IB & RD 1" is ranked higher than the two texts that are actually more similar to "IB & DU 1" in its themes ("IB & DU 2" and "IB & DU 3"), that is rather difficult to assess. Because of the "black box" nature of the models it is hard to say for certain what makes all of them consider these two answers to be the most similar, and by just reading the answers on their own it was hard to find any obvious similarities.

Figure 5.5 and Figure 5.6 display the results when comparing "IB & PF 3" to every other text, the former containing film titles, and the latter without. When using film titles, NorBERT 1 ranks the relevant texts the highest of all the models (second and third), while NB-SBERT is just below (third and fourth). Without film titles, NB-SBERT has the same ranking, whereas NorBERT 1's ranking of the two texts drops to fifth and sixth place. NorBERT 2 and NB-BERT-BASE performed a lot worse than the other two models in these tests.

Table 6.9 contains an overview of the rankings of the relevant texts for every model and every test, as well as the average ranking for every model. NB-SBERT's average ranking of relevant answers is 3, which is significantly better than the model with the second highest average ranking, which is NorBERT 1 with 6.

Base	Compared	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
IB & DU 1	IB & DU 2	12	14	14	3
	IB & DU 3	8	6	3	2
IB & DU 1	IB & DU 2	7	12	15	3
	IB & DU 3	5	6	3	2
IB & PF 3	IB & PF 1	2	9	10	3
	IB & PF 2	3	3	5	4
IB & PF 3	IB & PF 1	6	10	14	3
	IB & PF 2	5	3	7	4
Average ranking		6	7.88	8.88	3

Table 6.9: Rankings of the two most relevant texts by each model.

Another way to evaluate the results is to look at how the different models rank the texts that thematically have the least in common with the text being compared. For the first test, this would be the "RD & PF" texts, and for the second test, this would be "RD & DU". Ideally, these texts would be ranked the lowest for "IB & DU" and "IB & PF", respectively, since they describe entirely different films than those. The average rankings of the least relevant texts for each model in Figure 5.3 were calculated, and it was found that NB-BERT-BASE and NB-SBERT gave them the same average ranking of 11.33, compared to NorBERT 1's average of 7.66 and NorBERT 2's average of 8.33. A table containing these rankings, as well as the corresponding ones for the other three tests can be seen in Table 6.10. Here it is apparent that NB-SBERT gives a lower overall average ranking to the least thematically similar texts across all four test, implying that it is better at discarding the least similar texts. NB-BERT-BASE is also very close with an overall average ranking of 12.41 compared to NB-SBERT's 12.83.

Answer	Dataset	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
RD & PF	With titles	7.66	8.33	11.33	11.33
RD & PF	Without titles	6.66	9.66	10.66	11
RD & DU	With titles	13.66	13.66	14.33	14.66
RD & DU	Without titles	13.33	11.66	13.33	14.33
Overall average ranking		10.33	10.83	12.41	12.83

Table 6.10: Average rankings of all answers with the least thematic similarity to the text being compared.

By analyzing how good the models are at ranking the relevant texts compared to the less relevant text, it becomes apparent how adept they are at retrieving as little irrelevant data as possible. However, there will always be a threshold for how high the similarity score between a pair of texts has to be in order for the texts to be considered sufficiently similar. In TGRT, this threshold was set to be 0.92. As can be seen from the above cited figures, NB-SBERT gives consistently lower similarity scores than the other models, with its highest score being 0.83 and its lowest score being 0.52. All the other models range between 0.95 as the highest score and 0.76 as the lowest score. This means that if the Tarantino dataset were to be used on TGRT with NB-SBERT as the model, none of the pairs from these tests would be displayed. All the other models except NorBERT 2 would return at least some pairs with a similarity score above 0.92.

In the tests for the friction dataset, however (subsection 6.1.1), NB-SBERT gave the highest ranked texts similarity scores more in line with the other models. Several of the texts were given similarity scores over 0.90. This makes sense when you consider that the texts being compared in those tests by and large were a lot more similar to each other than the texts in the Tarantino dataset. The former texts consisted of different variations of mostly two distinct base texts, while the latter consisted of texts that were all distinctly formulated and structured.

From all tests of the Tarantino dataset shown above, it is clear that NB-SBERT has the highest range in terms of similarity scores. Its highest range between the top and bottom rankings is 0.29 and its average range across all four tests is 0.24, compared to NB-BERT-BASE's highest range of 0.13 (which is the highest out of all the other three models across all four tests). The fact that NB-SBERT has a higher range of similarity scores indicates that it is better at distinguishing between the different texts.

6.1.5 Tarantino dataset with single entries

The tests for the datasets with one movie description per entry had a very clear tendency in the results. In both the rankings displayed in Figure 5.7 and Figure 5.8, NB-SBERT is clearly the better option. Both tests show that the descriptions for the same movie are ranked together at the top of the list. This is a general pattern for the other models as well, but there seems to always be one or two descriptions that the models struggle to understand. Both "DU 1" and "IB 1" are quite short and leave out many details from the plot. In the second test, "DU 1" is further down the list, but still barely in the top half of it, but in the first test, "IB 1" is almost at the bottom of the rankings for all models except for NB-SBERT.

Answer	NorBERT 1	NorBERT 2	NB-BERT	NB-SBERT
IB 1	0.82	0.69	0.75	0.70
IB 3	0.95	0.90	0.91	0.87
IB 4	0.88	0.80	0.81	0.79
IB 5	0.94	0.90	0.90	0.92
IB 6	0.94	0.92	0.92	0.91
IB 7	0.91	0.80	0.89	0.80
IB 8	0.94	0.87	0.91	0.84
IB 9	0.93	0.87	0.90	0.87
Average	0.91375	0.84375	0.87375	0.8375

Table 6.11: Similarity scores for the same movie as the one compared to ("IB 2").

NB-SBERT understands that the descriptions labeled with the same movie concern the same plot. This is also evident by the similarity score. When only looking at the similarity scores for the similarly labeled entries that should be at the top (Table 6.11), there is no big difference between the models. NorBERT 1 even has a higher average ranking for the entries, and NB-SBERT has the lowest average similarity score. However, when looking at all the descriptions ranked in Figure 5.7 the impression changes. NorBERT 1 generally gives a higher similarity score to all the entries, with the lowest being 0.81, and all the models except for NB-SBERT struggle to differentiate between the descriptions for different movies. The relevant entries are generally at the top, but the differences in score between them and the non-relevant entries are not substantial. NB-BERT-Base has given "IB 7" 0.91 and "PF 3" 0.90 with only 0.01 in score difference. NorBERT 2 and NB-BERT-Base both have a drop of 0.06 from the clustered section of relevant entries and the rest. In addition, NorBERT 1, NorBERT 2 and NB-BERT-Base have single relevant entries spread further down the rankings. NB-SBERT on the other hand, has a noticeable drop in similarity score quality with a decrease of 0.17 from "IB 1" to "PF 1". All the entries that describe movies other than Inglourious Basterds are given similarity scores with 0.17 or more in difference to the entries labeled "IB".

This is a pattern that repeats when comparing to all the movie descriptions. Although NB-SBERT in some cases does not have a clustered section at the top containing all the relevant entries, it always performs better than the other models in differentiating the relevant and non-relevant entries.

6.1.6 Web development dataset

The web development dataset was important in development as test data that would function as an example of data that TGRT could actually receive in a practical setting. It would also enable tests of the comparison algorithm that measured the accuracy of the system by comparing the similarity scores to the graded scores. If the similarity scores were high when two answers had a similar grade, it can be counted as a positive towards the model. The accumulated SGCC was calculated to measure this for each model (displayed in Table 5.1). The higher the SGCC, the better the system is performing according to the graded scores.

The results were very similar across the models, with a range of 0.14 - 0.17 for the natural order and 0.13 - 0.16 with normalized similarity scores. According to the results, NorBERT 1 stands out from the rest in both tests, NorBERT 2 and NB-BERT-Base were in the middle with the exact same score for both natural order and normalized, while NB-SBERT has the lower coefficients. The results should not be trusted as a definite measurement of how well the models work, however.

SGCC measures how accurately the similarity scores match the graded scores from the examiner. This ties in with the main purpose of TGRT, which is to find errors in the graded scores. The SGCC can be completely wrong if the graded scores have many errors. This does not necessarily mean that the dataset used includes many errors, but there could be a small amount of errors that skew the results from this test. In addition to this, the calculations of SGCC do not take into account the problems described in subsection 4.6.1, namely that a similar grade does not necessarily mean the content of the answer is the same. Answers can contain different aspects of what is being asked in a question and receive the same points for it even though they have little in common content-wise. Because of these problems, and the fact that the SGCC's are close to equal across the models, this test was not considered as important in the decision of the comparison model.

6.1.7 Deciding on a model

By analyzing the tests described above, it was possible to discern how the models differed from each other and what strengths and weaknesses they had. Thus, a firm basis for deciding which particular model to implement in the project was established. From the discussion provided in this section, it is clear that NB-SBERT stands out in several ways. From the tests on the friction dataset, it was apparent that it tended to rank the texts labeled "Good, normal" higher than NorBERT 1 and NorBERT 2, and about as equally high as NB-BERT-BASE. This is positive because these texts are all semantically similar, but have a different structure and sentences. This thesis is concerned with detecting texts that convey the same meaning, but not necessarily with the same sentences.

NB-SBERT gave very low similarity scores for the texts in the friction dataset that were totally irrelevant. For the Tarantino datasets with dual descriptions, NB-SBERT both ranked the least relevant texts lower on average and gave them lower similarity score than the other models, although NB-BERT-BASE was very close in this regard. NB-SBERT gives a much higher range of similarity scores overall, implying a better ability to differentiate the texts from each other.

When it comes to ranking the most relevant texts, NB-SBERT stood out dramatically from the other models when testing the Tarantino datasets. For the datasets with dual descriptions, the average ranking of relevant answers for NB-SBERT was 3, compared to NorBERT 1's 6, which had the second highest ranking. This implies that it is better at finding texts with high semantic similarity. The tests for single movie descriptions also reveal that NB-SBERT excels by ranking most of the relevant entries on top of their list across all tests.

Another aspect which stood out in NB-SBERT's favor was that for the friction dataset, the texts with the "kinetic" label (i.e. those texts that described kinetic energy and not static) received markedly lower similarity scores than those given to the same texts by the other models. In other words, it was better than the other models at noticing that a significant portion of the content was missing. One thing that does not reflect as well on NB-SBERT is that it gave significantly lower similarity scores for the Tarantino dataset, with none of them exceeding the current threshold of 0.92. However, this might not be a problem, as the texts only had the same theme, and not necessarily similar content, which would be enough for TGRT to give a warning. Because of all the advantages of NB-SBERT, it was chosen as the model to compare answers.

6.1.8 Comparison algorithm in user tests

Before the user tests were conducted, it was already decided on which model to use based on the performance of the similarity algorithm tests. The user tests can therefore work as a validation for the chosen model. The results could not be matched against results using other models, as only the one model was used for the tests, but the examiners' approval could verify if the chosen model was adequate for the purpose of this project.

During the first test, the test subject examined five recommendations. Three of them she agreed with, which could be counted as positives towards the comparison algorithm. For the first of the remaining recommendations, she both disagreed with it and did not understand why the machine would deem them to be similar. If anyone could understand why the two answers were considered similar based on subject matter, it would be an expert in the field. Her not knowing why the recommendation was made can only be read as the algorithm having made a mistake. Exactly why the recommendation was made is hard to know because of the black box nature of transformers. For the other recommendation she disagreed with, she understood why the machine recommended as it did, as there was only a difference in the level of details that made one answer better than the other. Differences like this are hard to grasp by transformers, as they condense the meaning of the text into embeddings. Details that do not change the core meaning of the text can get lost. This is part of the reason the tool is meant as a supplementary tool and not as an automatic grader.

During the second test, the test subject found both some recommendations he disagreed with and some he agreed with. As mentioned in section 5.2, most of the questions on the exam being tested were programming tasks. Because of the nature of programming, there were several cases where answers that were very syntactically similar (and were thus considered to be similar by the algorithm) would produce very different results if the code was to be executed. A very simple change in one line of code could cause the entire program to stop working, meaning that two very similar answers could be given very different scores. This is different from normal text answers, where a slight change in one sentence is unlikely to change the meaning of the entire answer. Because of this distinction, normal text answers are more appropriate for the purposes of the tool. It also does not help that the BERT model is trained on a Norwegian corpus and is not trained to understand code.

Despite these false positives, there were several instances where TGRT recommended grade adjustments for programming tasks that the test subject agreed with. There were in fact several cases of nearly identical code snippets producing the same output that were identified as similar by the tool, where each answer had been graded differently. This is exactly the kind of situation that TGRT is meant to uncover, and it indicates that a tool for uncovering grade discrepancies between similar coding answers is feasible. Exploring the programming angle more thoroughly is however outside the scope of this project. Even still, despite the shortcomings associated with using the tool for programming tasks, the test subject stated that he found it useful and would have liked to use it in the grading process.

6.2 RQ2 Supporting features

In order to explain to the examiner why answers were given a high similarity score, features from each text needed to be generated and displayed. In this thesis, supporting features have been selected, and tested by experienced examiners. Are the features helpful in any way? In most cases, the examiner using TGRT has graded the submissions and therefore has previous knowledge of the contents. The supporting features are supposed to give a quick overview of the two compared answers. In this regard, the features should optimally save time for the examiner, as they should not need to read the entire answer again. The time saved increases the longer the text answers submitted are. The supporting features should also give the examiner a quick way of comparing the answers with more data than the similarity score alone. Ultimately, the supporting features should decrease time investment needed to both get an overview of the content and make it easier to compare the answers. These were the two main considerations examined during the user tests. The selected features implemented were unique terms from TF-IDF and RAKE along with generated summaries, themes and keywords from the OpenAI API.

6.2.1 First user test

Before the first test, a session was held where all warnings produced by the code were fixed. In the process of this, a conditional statement in the code for the supporting features TF-IDF and RAKE broke and resulted in those features not being displayed on the website. This was an easily avoidable mistake that should have been caught in testing right before the test, but even without TF-IDF and RAKE the test turned out to be successful. The test subject was very engaged and eager to try out the tool, and gave helpful feedback of her experience.

As TF-IDF and RAKE were unavailable for this test, the test subject could only comment on the supporting features from OpenAI. The summaries were helpful in that she could read them faster than scanning through the entire text answer, which was the main point of the supporting features. Interestingly, she found the keywords and themes less useful. As she comments, the theme for the answers often turn out to be a compacted version of the question in the exam. The keywords lacked context from her perspective and were therefore not useful, TF-IDF and RAKE would most likely also would not be as useful to her. The missing TF-IDF terms are core terms that are present in both answers. They also lack context, just like the keywords from OpenAI. RAKE could have been more useful, as it often contains more than just one word, with it extracting the most unique sequence of words. This is only speculation, as the extractive supporting features were not available for testing.

Both in the first test and the second, the test subjects have found instances where instead of the answer recommended for regrading, they found the answer it is compared to as worthy of changing. The user has in several cases then skimmed through the list of recommendations and in most cases not found it present there. It is missing because of how TGRT sorts and limits the list of recommendations, as explained in section 4.4. The reason for this decision was to not overwhelm the user by giving a disproportionately long list, and also not muddy TGRT's credibility by giving too many recommendations with too low similarity and few comparisons. The initial thoughts of the test subjects have been that the answers compared are assumed to be in the recommendations list. This could be subject to change in order to give the user enough of an overview of all the answers, while still marking them as not recommended to regrade.

Several sub-answers to sub-questions is a problem worthy of discussion. The sub-answers could be inputted into one or several input boxes during the exam. To support the case of several questions answered in only one box would be a big expansion to TGRT. To understand the structure of the answer, a separate analyzer built on AI could read it and grasp if it contains several answers based on markings like "a, b, c..." or "i, ii, iii...", but this is beyond this project's scope. The dataset reader does however combine sub-answers inputted into several text boxes into one text. To rather read these as separate answers is an expansion that is very attainable, but would either have needed a bit of a redesign of the website structure or letting the sub-questions have their own tab on the questions bar at the top of the page.

Another suggestion she had was to let the examiner compare the answers to a proposed solution to the exam. This means running the proposed solution through the model and comparing them to the submitted answers. It is not clear whether this is a good suggestion. The purpose of TGRT is to compare the contents of the answers and make notice if the content is similar and grades are not. If the proposed solution to the exam is included as a document in the algorithm, it would only serve the purpose to set the bar of the top graded answers. If answers have a high similarity score to the proposed solution, they might deserve the maximum score. In other situations, when examining less similar answers, it is less clear how useful this approach would be. It might be more helpful to the examiner to keep the proposed solution on their side while examining the results to manually compare the answers.

6.2.2 Second user test

All the supporting features were available for the second user test. As mentioned in section 5.2, the test subject stated that he found little use in the TF-IDF and RAKE terms, as well as the themes and keywords generated by OpenAI. This had a lot to do with the fact that the exam consisted mostly of programming tasks, where there is little to be gained by looking at words or lines of code in isolation. One might also expect that an AI-generated summary of a code snippet would not prove to be useful, but the test subject was surprised to see that the OpenAI-generated summaries were quite adept at describing what the code did. He did however emphasize that such summaries should not be trusted unconditionally, and that he needed to read the answers themselves in order to judge whether or not a grade adjustment was needed.

The test subject suggested a couple of new features that could be implemented in a tool that was targeted specifically towards programming tasks. These were syntax highlighting and color schemes for diffing between two answers. The former is a good idea, since it greatly increases the readability of the code, making it simpler to find the part of the code that is relevant and allowing for the discovery of syntax errors. Color schemes for diffing also seems like a useful feature, as it would allow the examiner to see which parts of the answers that are identical and which parts are different. This will make it even easier for the examiner to focus on the relevant parts of the answers, which will further speed up the process. While these are all useful features for a tool aimed at programming exams, this is well beyond the scope of this project.

6.3 RQ3 User perception

The third research question relates to how users perceive the innovative methods proposed by this thesis. Are examiners open to the idea of an AI quality-checking their work? And even if an examiner is positive to the concept, how will they experience it when the tool finds a mistake in their grading? It is important that the examiner, being the target user, is positive to the tool. Will they find it helpful in the grading process? The only way to test this was with practical user tests.

6.3.1 First user test

The test subject was initially positive to the concept of AI assisted exam grading. She had experienced on a first-hand basis how hard it is to stay objective and grade the submissions on a level playing field. Factors like time of day, day of the week, how well-fed she was, or other personal reasons could very easily affect how she scored the answers. She could for instance grade harsher based on small mistakes after a long session of grading many submissions than she would at the start of the session. This was of course a tendency she fought, and she would try to stop herself from being biased by factors that should not influence the grading process. She was very open to the idea of tools for supporting the grading process with the help of AI being put to use. Students could use AI tools to gain an advantage, and it was only natural that the examiners could also have a beneficial relationship with AI technology.

She found the test to be interesting and found TGRT intuitive and simple to use. As stated in section 5.2, the researchers gave an introduction to how the structure of the website worked and gave clarifications along the way to help the test subject understand how to test the methods most optimally, as the intuitiveness of the website was not under scrutiny. Regardless of explanations in advance, she found the recommendations well-structured and said it was easy to get an overview of the data.

The test unearthed a few answers that were recommended for a different grade than initially given, which she agreed with. She commented that she did not find the experience degrading, nor did she experience any other negative feelings. She saw it as human to make mistakes, and only found it interesting that the methods uncovered them. TGRT made a few mistakes as well. She understood how the algorithm was unable to understand the finer details of the content in one of the cases, but not in the other. She did not find this disconcerting, nor did it sour the overall impression of TGRT. It was understandable that the AI could not pick up on every detail, which is a reason as to why the methods are only meant to assist in the grading process and not to automatically adjust the grades.

In hindsight, the test subject might have not fully understood the setup and structure of the recommendations. She mostly looked at one of the comparisons given to the recommended changed answer and made conclusions based on that alone. There could have been more clarity in the explanation of the website and the flow of the process. The comparisons are there to substantiate the recommendation, and if only the top one is examined the recommendation could still be very valid, but it might be missing some support in order to be deemed valid.

6.3.2 Second user test

Prior to the test, the test subject had a positive inclination towards the use of AI in assisting exam grading and liked the concept of TGRT. He was concerned with how unconscious biases might affect how exams are graded, and was acutely aware of the need that TGRT was made to fulfill. This is similar to what the first test subject stated and concerns how the examiner as a human is affected by external factors. He was very excited to examine how potent and accurate the recommendations would be.

The test subject was very open to tools that can suggest changes and give helpful tips on things that can be improved along the grading process, but stressed that there needs to be a human element behind the grading. He had earlier attempted to let AI tools answer exam questions to test out their functionality. The tools were great at answering shorter concise questions, but when the questions became more complex, the AI struggled to answer within the correct context. If AI is to be used in any way in the grading process, a human must oversee the work performed.

As the exam set had been graded by other examiners, the test subject's reactions to uncovering grade discrepancies for similar answers were not as strong as they might have been if he had graded them. Nonetheless, he found it quite concerning to discover these discrepancies, though he did not find it surprising. He appreciated that TGRT suggested either an increase or decrease in graded score. The indication for a change in grade was helpful, even when he concluded that it was one of the answers compared to the recommended answer that should be graded differently, and not the recommended answer itself.

Many of the questions were programming tasks, which meant that answers contained only a program without any normal sentences or other context. The test subject blames this fact for some of the mistakes TGRT made in comparing answers. Without the context of what the programs are meant to do, it is hard for the algorithm to understand what is a well written answer and not. One question asked for what the output of a program would be, and specifically the outputted order of a list of words. It is very unlikely that TGRT can compare these answers justly, as it often dismisses the order of sentences and rather looks at the text as a whole. Regardless, the test subject was impressed at how well the comparison worked without context.

The recommendations the test subject disagreed with did not diminish his impression of TGRT, nor did it lessen the trust he had in the recommendations. This was because he did not have absolute faith in the recommendations to begin with. He emphasized the importance of the tool as a supportive measure in grading. If TGRT was meant as an automatic grader, an absolute trust in each recommendation would have been a lot more necessary. He was however aware from the start that this process would only measure discrepancies in the graded scores for answers with similar content and suggest changes based on them. The discrepancies that were discovered were noteworthy, and he emphasized that TGRT was great at extracting issues that needed to be examined. He did however also mention that if he had been an arbitrary examiner without his technical background, he would probably be more disheartened by the errors. A final product would benefit from a disclaimer stating that the recommendations are not inherently the correct solution, but only an indicator for a possible discrepancy that requires further examination.

The test subject was already convinced that AI-assisted grading would be the future of the grading process, and did not change his opinion during the test. He stated that it was great to see that AI could assist in this manner, and thatTGRT is a tool he would have liked to use when grading exams.

CHAPTER

SEVEN

CONCLUSIONS

7.1 Summary

This thesis has proposed methods for aiding examiners in the grading process. The methods have been compacted into a tool that uses an algorithm for comparing all the exam answers' contents to each other. If any answer has content similar to multiple answers in a higher or lower grade level, the tool would warn the examiner about this. The application was structured with a server that calculated the similarities and a client that displayed the findings to the user. It was tested by two examiners using previously graded exams. The three research questions presented in the introduction were:

- **RQ1: How can the similarities between exam answers be calculated?**
- **RQ2: What features can be extracted for assisting the examiners in analyzing similar answers?**
- **RQ3: How will a tool for detecting inconsistencies in grading be perceived by examiners?**

The chosen method of comparing the answers was to use BERT and to test different models that were pre-trained on Norwegian corpora. As discussed in subsection 6.1.7, the model that stood out the most with regard to accurately comparing the texts was NB-SBERT. The absence of predetermined similarity between answers in the datasets made it hard to determine which model compared them most accurately. However, a conclusion could still be reached after performing the tests described in section 5.1. These tests examined specific aspects like length and spelling mistakes, calculated similarity to answers that had the same grade, and calculated similarity between answers with the same theme or parts of the same theme. The results showed that the NB-SBERT model stood out from the rest of the selection of models. NB-SBERT was therefore used as the selected model for the user tests, and it performed well enough that the test subjects found several mistakes in their own grading. How well the other models might have fared in the same user tests is difficult to ascertain, but as the calculation method gave recommendations the test subjects agreed with, it can be concluded to work adequately for this thesis' purpose.

It was harder to conclude on the assistive properties of the supporting comparative features that were implemented. Their purpose was to make it easier for the examiner to know how the answers differ from and resemble each other without having to examine the full texts again. The user tests demonstrated that the supporting features had varying amounts of usefulness. The tests were not conclusive with regard to supportive features, as technical problems occurred, and they were not really helpful in examining code. The test subjects still agreed on the supporting features' value to the process, especially the summaries generated by OpenAI, as they helped to gain an overview of the answers quickly. Overall, the tool managed to successfully uncover mistakes that were made, but the time investment needed was greater than hoped. Optimizations and better ways of giving the examiner an overview quicker could be researched further.

With hardships in gathering examiners for testing the innovative methods implemented, the perception of examiners is not fully tested. The two tests conducted were held with examiners that were already positive to the idea of AI-assisted grading. The tests did not dissuade them from that view, even if TGRT made some mistakes the test subjects did not agree with. They were aware of the concept behind TGRT of extracting compared answers with a discrepancy in grades for answers with similar content. The user interface might indicate a bit too boldly that the recommendations given are correct, and should be marked with a disclaimer saying the recommendations are only indicators letting the user know there is something worth examining. The test subjects did not experience the process of finding errors in their grading as degrading or humiliating, but rather found it interesting and concerning that mistakes slipped through the system. They were both positively inclined to using the methods implemented in TGRT to aid them in the grading process.

7.2 Further work

There are many expansions to this project that could enhance the tool, which were deemed unfeasible for this thesis. As the comparison method was designed to work as broadly as possible, the model was not trained any further than what was already pre-trained on the base model. In theory, the models should understand the subject matter better if it is pre-trained on the subject in question. It is even possible to split the model into separate models that are each trained on different general subjects. Because there already was a very specific set of prerequisites necessary for the exams to work with the tool, leading to a considerably condensed user group to test on, it was undesirable to condense it any further by only focusing on a single subject.

In the same vein, the model could reward the use of certain terms and words per question, as was done in the study conducted by Bahel and Thomas [36]. This way, the similarity score between two answers could be boosted if they both contained the same key term. This would of course need to be researched further and fine-tuned to not only reward the student by mentioning the term, but to make sure the term is explained and understood in context.

This ties into another functionality that could expand the product. To understand what the student means by specific terms, it is possible to use opinion mining and sentiment analysis (as described in section 2.2). This entails extracting the writer's opinion of the subject matter, often to tell if it is negatively or positively skewed. The possibility of including this in the project was not fully explored because the need for it is not immediately evident. Most exams have little subjective and opinion based answers, but strive to be objective and fact based. It might however be useful to analyze the motivation behind key terms, or the opinions in the text could be interesting in edge cases where the student is supposed to be more subjective.

A way to expand the target group and have the tool be useful to more people would be to also account for the English language. Many of the examiners that were contacted who responded positively had to turn the offer of testing down because all their exams were held in English. A little under half of courses taught at NTNU are taught in English [60]. It was decided to focus on the Norwegian language and using Norwegian models to measure textual similarity because it is not researched nearly as much as the English models and would pose an interesting challenge for the thesis. A possible problem with exams held in English is that the submissions are not always written in English. Some students may choose to submit their answers in Norwegian, which would complicate the comparison with other students. Aside from the added process of detecting what language an answer is written in, many problems rise that need more research and testing. Should the dataset be split into an English set and a Norwegian set? Should one of them be translated? Maybe the BERT models can compare across languages? Whatever the solutions to these problems are, if the tool was to be developed further, support for English exams should be a high priority.

One of the other reasons why examiners turned down the test was because their exams were held either physically or were submitted as files in other formats like PDF. The first two datasets collected were in PDF format and needed to be manually formatted to work with TGRT. To create a dataset reader to be compatible with every conceivable way an exam could be structured would be close to impossible. It should however be possible to create an AI program that scans the file and extracts only the answer the student has written and excludes all the other data like page number and question text. In the same way, AI is capable of understanding handwritten text and converting it into digital text. The problem with these methods is that there is no guarantee that the conversion to digitally formatted text is absolutely correct. The AI program could misinterpret a word and completely change the meaning of the answer, which would be problematic in the school system. There should be no possibility of distorting the work of students to harm their grades.

A final thing that would improve the experience of the examiners would be to integrate a tool like TGRT into Inspera. If an examiner wants to use TGRT they first need to grade the exams in Inspera, then request a JSON file from the department administration, upload the file to TGRT, consider the recommendations made and decide on whether to regrade the answers, and go back to Inspera and change the grades. This is a cumbersome process that could easily be simplified by integrating it into Inspera. Then the examiner would not have to worry about obtaining and handling any files, and they would not have to access any external websites. There could simply be a button in Inspera to be pressed after giving the initial grades, which would start the calculations and lead to a new page where the recommendations for regrading would be displayed. Here the grades could be changed directly, as opposed to having to keep track of which answers from which exams need to be regraded. All of this would lead to a much more seamless user experience, but this is well beyond the scope of this project.

7.3 Limitations

There are several aspects that prevented this project from reaching its full potential. In some regards, this is because of the design decisions that were made, but also because of the nature of the project and limited time and resources. A core intention behind the methods implemented was to design them to be as inclusive as possible. This meant that the methods should not be optimized to fit exams for specific courses, for instance. TGRT was not designed in order to please one demographic while discouraging others. There are several reasons why this decision was made. The main reason was to keep the target group eligible for testing and dataset applicants as large as possible. Another reason was to show that the methods work on a large scale and not just single branches of academia. The implementation is meant as a prototype to prove that the concept works in practice, with a target group of exams that is as broad as possible.

If the project was to be researched further and developed into an extension for Inspera, it would be important to show that it can work well without any targeted pre-training. Because TGRT is designed to work broadly across many domains, it would in theory be helpful to most text answer exams without a lot of extra work to make it compatible.

AI is applicable in a wide range of fields, but it is not without limits. This thesis builds on the notion that automatic grading is not quite ripe for longer text answers, and instead proposes methods that support the examiner. This has to be thoroughly communicated to the expected user of TGRT. The recommendations made are not entirely fallible, and it might even be too presumptuous to call them recommendations. To call them recommendations indicates that TGRT has the correct answer and urges the examiner to change the given grade. The recommendation might not be accurate, as the user tests have proved with test subjects' disapproval of a number of recommendations given. A recommendation only indicates that there is a pattern of discrepancies between the grades of similar answers, with a tendency for the grades of the compared answers to be either higher or lower than the answer recommended to change. Whether this means the answer should be graded differently or not, is entirely up to the examiner.

A key aspect of the thesis that made the project enticing was that the similarity models and comparison methods were supposed to work on Norwegian texts. There has been little research on similarity in the Norwegian language using transformer models, and with several options of BERT models that had been trained on the Norwegian language, this was an intriguing prospect of the project. To only work with Norwegian exams and texts did however limit the amount of potential test subjects and datasets available for testing. This decision, and all the other prerequisites that emerged when designing TGRT around the data format from Inspera (explained in detail in subsection 4.2.1), led to a very small group of eligible exams and examiners that were able to test the system. Only having two user tests limited the ability to assert the robustness of TGRT, but the tests conducted proved that the concept has merit.

Privacy issues were prevalent during the project. Contacts at NTNU were concerned about the privacy of the students that have written the exam answers being tested. It was very important during testing that the researchers had no insight in the texts themselves, and only got excerpts from verbal communication with the test subjects. Development of the dataset reader would have been a lot quicker if several datasets were fully available for use. The test subjects were the only ones that had full control of the data used in testing, and they only use the data for examination purposes. The data temporarily saved on the server was deleted right after each test was completed. The answers were anonymous and only contained a candidate number connected to Inspera which got filtered out in the reading process. If the supporting features from OpenAI were used, the answers were sent to a web based service not under the control of the researchers. The data is assumed to be temporarily saved, but if the tool is to be further developed, a deal would have to be made with OpenAI to ensure that the privacy concerns of the students are upheld.

There are many of the potential expansions for the project considered in section 7.2 that could have been implemented if there was more time. However, this is a master's thesis, and to implement and test too many aspects of a subject can muddle the thesis as a whole. The subject of reading and formatting text from handwritten documents for the purpose of use in TGRT, for instance, would require a large expansion of the thesis and might fit better in a separate thesis dedicated to that subject.

7.4 In defense of viability

There are many prerequisites that need to be fulfilled in order for an exam to be eligible for the tool:

- It must be written in Norwegian
- It must be written digitally into Inspera and not via other file formats
- It must be mostly understandable without the inclusion of formulas, figures and images

Considering all this, one could question the legitimacy of the tool in the education system. If the number of tests in this project is indicative of the number of previous exams eligible for testing the tool, how can it be useful in a real life setting with today's exam system? A crucial part of the situation is that the test subjects brought exams that were conducted in the past few years. As digital exams become more commonplace, Inspera will likely become more useful as a tool for submitting exams directly, leading to more exams becoming eligible. The language problem will not solve itself, but the tool can be expanded to include an option to select what language the exam is held in. A lot more testing would be needed to find the best model to use for English exams, but it is entirely feasible.

The only problem not accounted for is exams where the answers are dependent upon figures and formulas for context. It is possible that AI technology could eventually read the figures and format it reliably enough to be a fair representation of the students' work. The exams where this is needed, however, are in most cases not amongst the target group for this tool. The target exams of the tool are those with answers consisting of longer texts rather than a lot of calculations and figures. It is also possible to use the tool for exams that include both figures/formulas and longer texts, and exclude the questions that contain figures or formulas. In any case, the tool should most likely see an increase in potential exams eligible in the near future, or can feasibly be expanded to encompass a much larger target group.

REFERENCES

[1] Ibrahim Albluwi. “A Closer Look at the Differences Between Graders in Introductory Computer Science Exams”. In: *IEEE Transactions on Education* 61.3 (2018), pp. 253–260. DOI: 10.1109/TE.2018.2805706.

[2] Maren Olava Ask Hütt. “Så sannsynlig er det å få bedre karakter om du klager”. In: *VG* (Oct. 7, 2021). URL: https://www.vg.no/nyheter/innenriks/i/v5wPRj/saa-sannsynlig-er-det-aa-faa-bedre-karakter-om-du-klager (visited on 06/06/2023).

[3] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[4] Eirik Plahte and Christian Riksvold. “Initial report for computer assisted exam grading”. Report written in preparation for this project. 2022.

[5] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[6] Michelle Wilson. *What is Natural Language Processing (NLP) and How is It Used Today?* URL: https://www.hp.com/us-en/shop/tech-takes/what-is-natural-language-processing (visited on 11/23/2022).

[7] Haoda Huang and Benyu Zhang. “Text Segmentation”. In: *Encyclopedia of Database Systems*. Ed. by LING LIU and M. TAMER ÖZSU. Boston, MA: Springer US, 2009, pp. 3072–3075. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_421. URL: https://doi.org/10.1007/978-0-387-39940-9_421.

[8] Tadashi Nomoto. “Keyword Extraction: A Modern Perspective”. In: *SN Computer Science* 4.1 (2022), p. 92.

[9] Amit Singhal and I. Google. “Modern Information Retrieval: A Brief Overview”. In: *IEEE Data Engineering Bulletin* 24 (Jan. 2001).

[10] Anirudha Simha. Oct. 6, 2021. URL: https://www.capitalone.com/tech/machine-learning/understanding-tf-idf/ (visited on 11/24/2022).

[11] Yassine Hamdaoui. *TF(Term Frequency)-IDF(Inverse Document Frequency) from scratch in python.* Dec. 10, 2019. URL: https://towardsdatascience.com/tf-term-frequency-idf-inverse-document-frequency-from-scratch-in-python-6c2b61b78558 (visited on 11/24/2022).

[12] Pradyoth SP. *Decoding the Key Concepts of Stop Words, BOW, TF, and IDF in NLP.* URL: https://medium.com/@sppradyoth/stop-words-bow-tf-and-idf-7fa2898ed03e (visited on 05/02/2023).

[13] Stuart Rose et al. “Automatic Keyword Extraction from Individual Documents”. In: *Text Mining: Applications and Theory*. John Wiley & Sons, Ltd, 2010. Chap. 1, pp. 1–20. ISBN: 9780470689646. DOI: https://doi.org/10.1002/9780470689646.ch1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470689646.ch1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470689646.ch1.

[14] Suphakit Niwattanakul et al. “Using of Jaccard Coefficient for Keywords Similarity”. In: Mar. 2013.

[15] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997.

[16] Christopher Olah. *Understanding LSTM Networks*. Aug. 27, 2015. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (visited on 11/17/2022).

[17] Britney Muller. *BERT 101 - State Of The Art NLP Model Explained*. Mar. 2, 2022. URL: https://huggingface.co/blog/bert-101 (visited on 11/22/2022).

[18] Aayush Srivastava. *What Are Transformers In NLP And It’s Advantages*. Aug. 8, 2022. URL: https://blog.knoldus.com/what-are-transformers-in-nlp-and-its-advantages/ (visited on 11/22/2022).

[19] Ria Kulshrestha. *Transformers*. May 29, 2020. URL: https://towardsdatascience.com/transformers-89034557de14 (visited on 11/22/2022).

[20] Jesse Vig. *Deconstructing BERT, Part 2: Visualizing the Inner Workings of Attention*. Jan. 7, 2019. URL: https://towardsdatascience.com/deconstructing-bert-part-2-visualizing-the-inner-workings-of-attention-60a16d86b5c1 (visited on 05/09/2023).

[21] Jay Alammar. *The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)*. URL: http://jalammar.github.io/illustrated-bert/ (visited on 11/22/2022).

[22] Rani Horev. *BERT Explained: State of the art language model for NLP*. URL: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270 (visited on 11/23/2022).

[23] Saketh Kotamraju. *An Intuitive Explanation of Sentence-BERT*. June 23, 2022. URL: https://towardsdatascience.com/an-intuitive-explanation-of-sentence-bert-1984d144a868 (visited on 03/13/2023).

[24] Nils Reimers and Iryna Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. 2019. arXiv: 1908.10084 [cs.CL].

[25] Amit Kayal. *My journey into Sentence Transformer*. Oct. 16, 2022. URL: https://dev.to/aws-builders/my-journey-into-sentence-transformer-1b7m (visited on 03/13/2023).

[26] *Say hi to NorBERT!* Jan. 14, 2021. URL: https://www.mn.uio.no/ifi/english/research/projects/sant/news/norlm.html (visited on 11/23/2022).

[27] *Vectors/norlm/norbert*. URL: http://wiki.nlpl.eu/Vectors/norlm/norbert (visited on 11/23/2022).

[28] Per E Kummervold et al. “Operationalizing a National Digital Library: The Case for a Norwegian Transformer Model”. In: *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*. Reykjavik, Iceland (Online): Linköping University Electronic Press, Sweden, 2021, pp. 20–29. URL: https://aclanthology.org/2021.nodalida-main.3.

[29] Zuhaib Akhtar. *BERT base vs BERT large.* URL: https://iq.opengenus.org/bert-base-vs-bert-large/ (visited on 04/25/2023).

[30] URL: https://huggingface.co/NbAiLab/nb-sbert-base (visited on 03/13/2023).

[31] URL: https://huggingface.co/datasets/NbAiLab/mnli-norwegian (visited on 04/17/2023).

[32] Dandan Chen and Carolyn J. Anderson. "Categorical data analysis". In: *International Encyclopedia of Education (Fourth Edition)*. Ed. by Robert J Tierney, Fazal Rizvi, and Kadriye Ercikan. Fourth Edition. Oxford: Elsevier, 2023, pp. 575–582. ISBN: 978-0-12-818629-9. DOI: https://doi.org/10.1016/B978-0-12-818630-5.10070-3. URL: https://www.sciencedirect.com/science/article/pii/B9780128186305100703.

[33] Ken Stewart. *mean squared error.* 2023. URL: https://www.britannica.com/science/mean-squared-error (visited on 05/23/2023).

[34] Shrivarsheni. *Text Summarization Approaches for NLP – Practical Guide with Generative Examples.* Oct. 24, 2020. URL: https://www.machinelearningplus.com/nlp/text-summarization-approaches-nlp-example/ (visited on 11/17/2022).

[35] Dan Jurafsky. *Minimum Edit Distance.* Jan. 17, 2012. URL: web.stanford.edu/class/cs124/lec/med.pdf (visited on 11/24/2022).

[36] Vedant Bahel and Achamma Thomas. "Text similarity analysis for evaluation of descriptive answers". In: *CoRR* abs/2105.02935 (2021). arXiv: 2105.02935. URL: https://arxiv.org/abs/2105.02935.

[37] Xinfeng Ye and Sathiamoorthy Manoharan. "Performance Comparison of Automated Essay Graders Based on Various Language Models". In: *2021 IEEE International Conference on Computing (ICOCO).* 2021, pp. 152–157. DOI: 10.1109/ICOCO53166.2021.9673585.

[38] Phakawat Wangkriangkri et al. "A Comparative Study of Pretrained Language Models for Automated Essay Scoring with Adversarial Inputs". In: *2020 IEEE REGION 10 CONFERENCE (TENCON).* 2020, pp. 875–880. DOI: 10.1109/TENCON50793.2020.9293930.

[39] Carlo Lepelaars. *Understanding The Metric: Quadratic Weighted Kappa.* Nov. 23, 2019. URL: https://www.kaggle.com/code/carlolepelaars/understanding-the-metric-quadratic-weighted-kappa (visited on 06/08/2023).

[40] Akshita Jha et al. "Supervised Contrastive Learning for Interpretable Long Document Comparison". In: *CoRR* abs/2108.09190 (2021). arXiv: 2108.09190. URL: https://arxiv.org/abs/2108.09190.

[41] Zhengguang Li et al. "Cross2Self-attentive Bidirectional Recurrent Neural Network with BERT for Biomedical Semantic Text Similarity". In: *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).* 2020, pp. 1051–1054. DOI: 10.1109/BIBM49941.2020.9313452.

[42] Itzik Malkiel et al. "Interpreting BERT-Based Text Similarity via Activation and Saliency Maps". In: *Proceedings of the ACM Web Conference 2022.* WWW '22. Virtual Event, Lyon, France: Association for Computing Machinery, 2022, pp. 3259–3268. ISBN: 9781450390965. DOI: 10.1145/3485447.3512045. URL: https://doi.org/10.1145/3485447.3512045.

[43] *Inglourious Basterds.* 2009.

[44] *Django Unchained*. 2012.

[45] *Pulp Fiction*. 1994.

[46] *Reservoir Dogs*. 1992.

[47] URL: https://spacy.io/models/nb (visited on 05/01/2023).

[48] *RAKE*. Mar. 2, 2018. URL: https://github.com/zelandiya/RAKE-tutorial (visited on 05/30/2023).

[49] *Natural Language Toolkit*. Jan. 2, 2023. URL: https://www.nltk.org/ (visited on 05/30/2023).

[50] *summa 1.2.0*. Jan. 16, 2019. URL: https://pypi.org/project/summa/ (visited on 05/30/2023).

[51] Rada Mihalcea and Paul Tarau. "TextRank: Bringing Order into Text". In: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 404–411. URL: https://aclanthology.org/W04-3252.

[52] Greg Brockman and Ilya Sutskever. *Introducing OpenAI*. Dec. 11, 2015. URL: https://openai.com/blog/introducing-openai (visited on 06/08/2023).

[53] *OpenAI Charter*. Apr. 9, 2018. URL: https://openai.com/charter (visited on 05/30/2023).

[54] *Introducing ChatGPT*. Nov. 30, 2022. URL: https://openai.com/blog/chatgpt (visited on 05/30/2023).

[55] *Models*. URL: https://platform.openai.com/docs/models (visited on 05/30/2023).

[56] *React*. URL: https://react.dev/ (visited on 05/30/2023).

[57] *Welcome to Flask*. URL: https://flask.palletsprojects.com/en/2.3.x/ (visited on 05/30/2023).

[58] *symspellpy*. URL: https://symspellpy.readthedocs.io/en/latest/index.html (visited on 05/30/2023).

[59] Knut Hofland. Oct. 11, 1998. URL: http://korpus.uib.no/humfak/nta/ (visited on 05/30/2023).

[60] URL: https://www.ntnu.edu/studies/coursesearch#semester=2022&gjovik=false&trondheim=false&alesund=false&faculty=-1&institute=-1&multimedia=false&english=false&phd=false&open=false&courseAutumn=false&courseSpring=false&courseSummer=false&pageNo=1&season=spring&sortOrder=ascTitle (visited on 04/25/2023).

APPENDIX

A - CODE EXCERPTS

A1 - Text formatter

```
def get_valid_questions_list(file):
    full_set = file["ext_inspera_candidates"]
    questions = {}
    for student in full_set:
        for question in student["result"]["
           ext_inspera_questions"]:
            if "ext_inspera_manualScores" in question:
                if "ext_inspera_candidateResponses" in
                   question and len(question["
                   ext_inspera_candidateResponses"]) > 0
                   and not ((type(question["
                   ext_inspera_candidateResponses"][0]["
                   ext_inspera_response"]) == 'str' or type
                   (question["
                   ext_inspera_candidateResponses"][0]["
                   ext_inspera_response"]) == 'string') and
                    question["
                   ext_inspera_candidateResponses"][0]["
                   ext_inspera_response"].startswith("
                   simpleChoice_")):
                    questions[question["
                       ext_inspera_questionId"]] = True
            elif question["ext_inspera_questionId"] not in
               questions:
                questions[question["ext_inspera_questionId"
                   ]] = False
    print(questions)
    return questions

def parse_file(file):
    answers = {}
    full_set = file["ext_inspera_candidates"]
    valid_questions_list = get_valid_questions_list(file)
    pre_scores = {}
```

```
for student in full_set:
    question_list = student["result"]["
       ext_inspera_questions"]
    if len(question_list) == 0:
        continue
    for current_answer in question_list:
        if not valid_questions_list[current_answer["
           ext_inspera_questionId"]] or not ("
           ext_inspera_manualScores" in current_answer
           and len(current_answer["
           ext_inspera_manualScores"]) > 0):
            continue
        answer = ""
        if "ext_inspera_candidateResponses" in
           current_answer and len(current_answer["
           ext_inspera_candidateResponses"]) > 0:
            part_answers = [part_answer["
               ext_inspera_response"] for part_answer
               in current_answer["
               ext_inspera_candidateResponses"] if
               isinstance(part_answer["
               ext_inspera_response"], str)]
            answer = "\n".join([part_answer for
               part_answer in part_answers])

        if current_answer["ext_inspera_questionId"] not
            in answers:
            answers[current_answer["
               ext_inspera_questionId"]] = []
        answers[current_answer["ext_inspera_questionId"
           ]].append(answer)

        if current_answer["ext_inspera_questionId"] not
            in pre_scores:
            pre_scores[current_answer["
               ext_inspera_questionId"]] = []

        scores = [float(score["ext_inspera_manualScore"
           ]) for score in current_answer["
           ext_inspera_manualScores"]]
        pre_scores[current_answer["
           ext_inspera_questionId"]].append(sum(scores)
           )
```

A2 - Server main

```
import os
import string
from flask import Flask, flash, request, redirect, url_for,
    jsonify
from flask.logging import default_handler
import logging
from flask_cors import CORS
from bert import BERT
from tfidf import compare_tfidf, tfidf_matrix
from utils import sort_by_pre_score
from RAKE.rake import Rake
import json
from readers.data_set_reader import
   get_valid_questions_list, parse_file
import threading
import random
from sbert_test import encode
from colors import CGREEN, CEND
import os
import json
import codecs

UPLOAD_FOLDER = '/'
ALLOWED_EXTENSIONS = {'json'}

default_handler.setFormatter(logging.Formatter(CGREEN + "%(
   message)s" + CEND))

app = Flask(__name__)
CORS(app)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER


def tuple_to_string(tuple):
    return "("+str(tuple[0])+", "+str(tuple[1])+")"


def generate_id(length):
    return ''.join(random.choice(string.ascii_letters) for
       i in range(length))

def post_thread_runner(docs, scores, id, question):
    tables = {
            "results": []
        }

    if len(scores) > 0:
        print("post_thread; init")
        tfidf_matrix_list = tfidf_matrix(docs)
        rake = Rake(stop_words_path="stopwords.txt")
        j = codecs.open(os.curdir+"/norwegian-synonyms.json
            ", "r", "utf-8")
        synonyms = json.load(j)

        print("post_thread;␣file␣read␣complete")

        for score in scores:
            tfidf_comparison = compare_tfidf(
               tfidf_matrix_list, synonyms, score[0], score
               [1])

            comparison = []
            for el in tfidf_comparison:
                comparison.append({
                    "term": el[0],
                    "score": el[1],
                    "tf_idf_alpha": el[2],
                    "tf_idf_beta": el[3]
                })
            rake_terms_alpha = rake.run(docs[score[0]])
               [:25]
            rake_terms_beta = rake.run(docs[score[1]])[:25]
            tables["results"].append({
                "alpha_index": score[0],
                "beta_index": score[1],
                "metrics": comparison,
                "rake_terms_alpha": rake_terms_alpha,
                "rake_terms_beta": rake_terms_beta
            })

    res = json.dumps(tables, indent=4)

    if not os.path.exists("results/" + id):
        os.makedirs("results/" + id)

    with open("results/" + id + "/" + str(question + 1) + "
       _metrics.json", "w") as file:
        file.write(res)
        file.close()

    print("metrics␣for␣question␣" + str(question) + "␣
       finished")


def main_thread_runner(posted_data, id):
    full_set, full_pre_scores = parse_file(posted_data)
    for (i, question_number) in enumerate(full_set):
        docs = full_set[question_number]
        pre_scores = full_pre_scores[question_number]
        print(f'{len(docs)=}')
        print(f'{len(pre_scores)=}')
        matrices = [BERT(docs, "NbAiLab/nb-sbert-base")]
        scores = sort_by_pre_score(matrices[0], pre_scores)
        tables = {
            "results": []
        }
        for score in scores:
            result = {
                "alpha": docs[score[0]],
                "beta": docs[score[1]],
                "alpha_index": str(score[0]),
                "beta_index": str(score[1]),
                "score_diff": str(score[2]),
                "similarity": str(score[3]),
                "alpha_score": str(pre_scores[score[0]]),
                "beta_score": str(pre_scores[score[1]]),
                "tf_idf_terms": []
            }
            tables["results"].append(result)
        res = json.dumps(tables, indent=4)

        if not os.path.exists("results/" + id):
            os.makedirs("results/" + id)

        with open("results/" + id + "/" + str(i + 1) + ".
           json", "w") as file:
            file.write(res)
            file.close()

        thread = threading.Thread(target=post_thread_runner
           , args=(
            docs, scores, id, i))
        thread.start()

        print(str(question_number) + "/" + str(len(docs)) +
            "␣finished")


@app.route("/", methods=['GET'])
def test():
    return "Hello␣world"


@app.route('/upload', methods=['POST'])
```

```
def process_data():
    posted_file = request.files['file']
    posted_data = json.load(posted_file)
    valid_questions = list(get_valid_questions_list(
       posted_data).keys())
    #valid_questions = [1, 2]
    id = generate_id(5)
    thread = threading.Thread(target=main_thread_runner,
       args=(
        posted_data, id))
    thread.start()
    response = jsonify({'id': id, "valid_questions":
       valid_questions})
    response.headers.add('Access-Control-Allow-Origin', '*'
       )
    return response


@app.route("/poll", methods=['GET'])
def poll():
    id = request.args.get('id', default=1, type=str)
    question = request.args.get('question', default=0, type
       =int)
    path = "results/" + id + "/" + str(question) + ".json"
    res = "false"
    if os.path.exists(path):
        with open(path, "r") as file:
            res = file.read()
            file.close()
    response = jsonify({'res': res})
    response.headers.add('Access-Control-Allow-Origin', '*'
       )
    return res

@app.route("/poll_metrics", methods=['GET'])
def poll_metrics():
    id = request.args.get('id', default=1, type=str)
    question = request.args.get('question', default=0, type
       =int)
    path = "results/" + id + "/" + str(question) + "
       _metrics.json"
    res = "false"
    if os.path.exists(path):
        with open(path, "r") as file:
            res = file.read()
            file.close()
    response = jsonify({'res': res})
    response.headers.add('Access-Control-Allow-Origin', '*'
        )
    return res
```

A3 - BERT setup

```
from sklearn.metrics.pairwise import cosine_similarity
from transformers import AutoTokenizer, AutoModel
import torch
from progress_printer import print_progress
from transformers import logging
def warn(*args, **kwargs):
    pass
import warnings
warnings.warn = warn
logging.set_verbosity_error()

def getModel(model_string):
    return AutoTokenizer.from_pretrained(model_string),
       AutoModel.from_pretrained(model_string)

def BERT(sentences, model_string):
    print_progress(0, model_string)
    tokenizer, model = getModel(model_string)

    chunksize = 128

    indexes = []
    # initialize dictionary to store tokenized sentences
    tokens = {'input_ids': [], 'attention_mask': []}
    print_progress(0.1, model_string)
    for sentence in sentences:
        # encode each sentence and append to dictionary
        new_tokens = tokenizer.encode_plus(sentence,
           add_special_tokens=False,
                                return_tensors='pt')

        input_id_chunks = list(new_tokens['input_ids'][0].
           split(chunksize - 2))
        mask_chunks = list(new_tokens['attention_mask'][0].
           split(chunksize - 2))

        indexes.append(len(input_id_chunks))

        for i in range(len(input_id_chunks)):
            # add CLS and SEP tokens to input IDs
            input_id_chunks[i] = torch.cat([
                torch.tensor([101]), input_id_chunks[i],
                torch.tensor([102])
            ])
            # add attention tokens to attention mask
            mask_chunks[i] = torch.cat([
                torch.tensor([1]), mask_chunks[i], torch.
                   tensor([1])
            ])

            # padding
            pad_len = chunksize - input_id_chunks[i].shape
                [0]
            if pad_len > 0:
                input_id_chunks[i] = torch.cat([
                    input_id_chunks[i], torch.Tensor([0] *
                       pad_len)
                ])
                mask_chunks[i] = torch.cat([
                    mask_chunks[i], torch.Tensor([0] *
                       pad_len)
                ])

        tokens['input_ids'].extend(input_id_chunks)
        tokens['attention_mask'].extend(mask_chunks)

    print_progress(0.2, model_string)

    # reformat list of tensors into single tensor
    tokens['input_ids'] = torch.stack(tokens['input_ids']).
       long()
    tokens['attention_mask'] = torch.stack(tokens['
       attention_mask']).int()

    print_progress(0.3, model_string)

    outputs = model(**tokens)

    print_progress(0.4, model_string)

    embeddings = outputs.last_hidden_state
    attention_mask = tokens['attention_mask']

    print_progress(0.5, model_string)

    mask = attention_mask.unsqueeze(-1).expand(embeddings.
       size()).float()
    masked_embeddings = embeddings * mask

    print_progress(0.7, model_string)
    summed = torch.sum(masked_embeddings, 1)
    summed_mask = torch.clamp(mask.sum(1), min=1e-9)
    mean_pooled = summed / summed_mask

    print_progress(0.8, model_string)

    # calculate
    adjusted_mean_pooled = []
    temp = 0
    for i, amnt in enumerate(indexes):
        res = torch.zeros(mean_pooled.shape[1])
        for j in range(amnt):
            res = torch.add(res, mean_pooled[i+temp], alpha
                =(1/amnt))
            if j > 0:
                temp += 1
        adjusted_mean_pooled.append(res)
    adjusted_mean_pooled = torch.stack(adjusted_mean_pooled
        )

    # convert from PyTorch tensor to numpy array
    adjusted_mean_pooled = adjusted_mean_pooled.detach().
        numpy()

    print_progress(0.9, model_string)

    sims = cosine_similarity(
        adjusted_mean_pooled,
        adjusted_mean_pooled

    )

    print_progress(1, model_string)

    return sims
```

A4 - File manager

```
import pickle
import os
import json
import codecs
import settings
from tabulate import tabulate

# write list to binary file
```

```
def write_list(a_list, file_ext):
    # store list in binary file so 'wb' mode
    with open("results/" + settings.data_set + file_ext, '
       wb') as fp:
        pickle.dump(a_list, fp)

# Read list to memory
def read_list(file_ext):
    # for reading also binary mode is important
    with open("results/" + settings.data_set + file_ext, '
       rb') as fp:
        n_list = pickle.load(fp)
        return n_list


def tuple_to_string(tuple):
    return "("+str(tuple[0])+",␣"+str(tuple[1])+")"

def write_res_all(scores):
    categories = [doc["category"] for doc in settings.
       raw_data if "category" in doc]
    inspera_datasets = ["inspera_friction", "
       inspera_tarantino", "inspera_tarantino_no_names", "
       inspera_tarantino_split", "
       inspera_tarantino_split_no_names"]
    if len(categories) == 0 and settings.data_set in
       inspera_datasets:
        dataset = "_".join(settings.data_set.split("_")
           [1:])
        j = codecs.open(os.curdir + "/readers/
           custom_datasets/" + str(dataset) + ".json", "r",
            "utf-8")
        docs = json.load(j)
        categories = [doc["category"] for doc in docs]

    """ prev_el = None
    print("-------" + categories[settings.alpha_answer] +
       "-----------")
    for el in [(categories[s[0]], s[1]) for s in scores
       [3]]:
        if categories[settings.alpha_answer].split(" ")[0]
           == "PF":
            break
        print(el)
        if el[0].split(" ")[0] == categories[settings.
           alpha_answer].split(" ")[0]:
            prev_el = el
        elif prev_el is not None:
            if abs(el[1] - prev_el[1]) > settings.max_drop:
                settings.max_drop = abs(el[1] - prev_el[1])
                settings.cat = categories[settings.
                    alpha_answer] + " - " + str(settings.
                    alpha_answer) + " - " +  el[0] + " - " +
                     prev_el[0] + " - " + str(el[1]) + " - "
                     + str(prev_el[1])
            break """


    tables = []
    for i in range(len(scores[0])):
        tables.append([(categories[s[0]] if len(categories)
            > 0 else s[0], f'{s[1]:4.2f}') for s in [score[
           i] for score in scores]])
    t = tabulate(tables, headers=['norbert', 'norbert2', '
       nb-bert-base', 'nb-sbert-base'])

    with open("results/results.txt", "w") as file:
        file.write("dataset:␣" + settings.data_set + "\
           nanswer␣compared␣to␣" + categories[settings.
           alpha_answer] + "\n\n")
        file.close()

    with open("results/results.txt", "a") as file:
        file.write(t)
        file.close()

def write_res_all_with_prescores(scores, pre_scores):
    import warnings

    with warnings.catch_warnings():
        warnings.filterwarnings('error')
        variance = []
        variance_norm = []
        for score in scores:
            max = 0
            min = 1
            max_grade = 0
            for s in score:
                if s[1] > max:
                    max = s[1]
                if s[1] < min:
                    min = s[1]
                if pre_scores[s[0]] > max_grade:
                    max_grade = pre_scores[s[0]]

            v = 0
            v_norm = 0
            try:
                for s in score:
                    """ ratio = 1
                    if pre_scores[settings.alpha_answer] !=
                        pre_scores[s[0]]:
                        ratio = 1 / abs(pre_scores[s[0]] -
                            pre_scores[settings.alpha_answer
                            ])
                    v += s[1] * ratio """
                    diff = abs(pre_scores[settings.
                        alpha_answer] - pre_scores[s[0]]) /
                        max_grade
                    v += abs(diff - s[1])
                    v_norm += abs(diff - ((s[1] - min) / (
                        max - min)))
                    settings.variance_counter += 1
            except:
                print("Error:")
                print(settings.raw_data[settings.
                    alpha_answer])

            variance.append(v)
            variance_norm.append(v_norm)

        for (i, var) in enumerate(variance):
            settings.variance[i] += var

        for (i, var) in enumerate(variance_norm):
            settings.variance_normalized[i] += var

def write_res_by_diff(scores):
    categories = [doc["category"] for doc in settings.
        raw_data if "category" in doc]

    tables = []
    for i, score in enumerate(scores):
        tables.append((categories[score[0]] if len(
            categories) > 0 else score[0], categories[score
            [1]] if len(categories) > 0 else score[1], score
            [2], score[3], settings.pre_scores[score[0]],
            settings.pre_scores[score[1]]))
    t = tabulate(tables)

    with open("results/results_by_score.txt", "w") as file:
        file.write(t)
        file.close()
```

```
def write_tfidf_matrix(tfidf_matrix):
    res = ""
    for (i) in range(len(tfidf_matrix)):
        res += tuple_to_string(tfidf_matrix[i]) + "\n"

    with open("results/" + settings.data_set + "
       _tfidf_matrix.txt", "w") as file:
        file.write(res)
        file.close()
```

A5 - TF-IDF setup

```
from sklearn.feature_extraction.text import TfidfVectorizer
from nltk.tokenize import word_tokenize
from sklearn.metrics.pairwise import cosine_similarity
import spacy
import settings

def warn(*args, **kwargs):
    pass
import warnings
warnings.warn = warn

nlp = spacy.load("nb_core_news_lg")

def tokenize(text):
    tokens = [word for word in word_tokenize(text) if len(
       word) > 1]
    lemmatizations = [nlp(token)[:] for token in tokens]
    return [lem.lemma_ for lem in lemmatizations]

def calculate(docs):
    stopwords = []
    with open("stopwords.txt", "r", encoding="utf-8") as
       file:
        file_lines = file.read()
        stopwords = file_lines.split("\n")
        file.close()

    tfidf_vectorizer = TfidfVectorizer(use_idf=True,
       stop_words=stopwords, tokenizer=tokenize,
       ngram_range=(1, 1))  # type: ignore
    tfidf_matrix = tfidf_vectorizer.fit_transform(docs)

    return tfidf_vectorizer, tfidf_matrix

def term_matrix(tfidf_vectorizer, tfidf_matrix):
    feature_names = tfidf_vectorizer.get_feature_names_out
       ()
    res = [(word, [0] * tfidf_matrix.shape[0]) for word in
       feature_names]

    for doc in range(tfidf_matrix.shape[0]):
        feature_index = tfidf_matrix[doc,:].nonzero()[1]
        tfidf_scores = zip(feature_index, [tfidf_matrix[doc
           , x] for x in feature_index])
        for (i, s) in tfidf_scores:
            res[i][1][doc] = s

    return res

# returns similarity score based on tfidf
def tfidf(docs):
    _, tfidf_matrix = calculate(docs)

    # measure similarity of all articles with cosine
       similarity
    cosine_sim = cosine_similarity(tfidf_matrix,
       tfidf_matrix)

    return cosine_sim

# returns tfidf_matrix with terms used and tfidf score of
   each term in each document
def tfidf_matrix(docs):
    tfidf_vectorizer, tfidf_matrix = calculate(docs)

    return term_matrix(tfidf_vectorizer, tfidf_matrix)

def get_tfidf_terms(tfidf_matrix_list, alpha_answer):
    res = [(tup[0], tup[1][alpha_answer]) for tup in
       tfidf_matrix_list if tup[1][alpha_answer] > 0]
    return sorted(res, key=lambda x: x[1], reverse=True)

def comparative_score(alpha_score, beta_score):
    diff = 1 - abs(alpha_score - beta_score)
    avg = (alpha_score + beta_score) / 2
    return diff * avg

def compare_synonyms(synonyms, alpha_term, beta_terms):
    if alpha_term[0] in synonyms:
        for term in synonyms[alpha_term[0]]:
            if term in [z[0] for z in beta_terms]:
                beta_term = [z for z in beta_terms if z[0]
                   == term][0]
                return (alpha_term[0] + "/" + term,
                   comparative_score(alpha_term[1],
                   beta_term[1]), alpha_term[1], beta_term
                   [1])
    return None

# compares two tfidf scores with a set formula
def compare_tfidf(tfidf_matrix_list, synonyms, alpha_answer
   =-1, beta_answer=-1):
    if alpha_answer == -1 and beta_answer == -1:
        alpha_answer = settings.alpha_answer
        beta_answer = settings.beta_answer
    alpha = get_tfidf_terms(tfidf_matrix_list, alpha_answer
       )
    beta = get_tfidf_terms(tfidf_matrix_list, beta_answer)
    res = []
    for alpha_term in alpha:
        if alpha_term[0] in [z[0] for z in beta]:
            beta_term = [z for z in beta if z[0] ==
               alpha_term[0]][0]
            res.append((alpha_term[0], comparative_score(
               alpha_term[1], beta_term[1]), alpha_term[1],
                beta_term[1]))
        else:
            synonym_res = compare_synonyms(synonyms,
               alpha_term, beta)
            if synonym_res != None:
                res.append(synonym_res)
    return sorted(res, key=lambda x: x[1], reverse=True)
```

A6 - Server side sorting algorithm

```
import settings

def sort_by_pre_score(matrix, pre_scores=[]):
    scores = []
    for i in range(len(matrix)):
        scores.append(extract_similar_answers(matrix, i))
    if len(pre_scores) < 1:
        pre_scores = settings.pre_scores
    max_score = max(pre_scores)
    print(max_score)
    res = []
    for i, score in enumerate(scores):
        for j, s in score:
            score_diff = abs(pre_scores[i] - pre_scores[j])
                / max_score
            if s > 0.92 and score_diff > 0 and (j, i,
               score_diff, s) not in res:
                res.append((i, j, score_diff, s))
    res = sorted(res, key=lambda x: x[3], reverse=True)
    return res

def extract_similar_answers(res, answer=-1):
    if answer == -1:
        answer = settings.alpha_answer
    sim_scores = list(enumerate(res[answer]))
    # sorting based on match score
    sim_scores = sorted(sim_scores, key=lambda x: x[1],
       reverse=True)
    sim_scores = sim_scores[1:]

    return sim_scores
```

A7 - Client side sorting algorithm

```
export const structureData = (res) => {
    let data = new Map();
    res.results.forEach((el) => {
        if (!data.has(el.alpha_index)) {
            data.set(el.alpha_index, { text: el.alpha,
               score: el.alpha_score, comparisons: [] });
        }
        if (!data.has(el.beta_index)) {
            data.set(el.beta_index, { text: el.beta, score:
                el.beta_score, comparisons: [] });
        }
        data.get(el.alpha_index).comparisons.push({index:
           el.beta_index, sim: el.similarity});
        data.get(el.beta_index).comparisons.push({index: el
           .alpha_index, sim: el.similarity});
    })
    let highestProposalScore = 0;
    data.forEach((val, key )=> {
        const proposalScore = calculateProposalScore(val,
           data);
        highestProposalScore = proposalScore >
           highestProposalScore ? proposalScore :
           highestProposalScore;
        data.set(key, {...val, proposal: proposalScore});
    })
    return data;
}
```

```
export const sortAnswers = (data) => {
    const res = [];
    data.forEach((value, key) => {
        res.push({key: key, value: value});
    });
    return res.sort((a,b) => {
        let aCounter = 0;
        let bCounter = 0;
        a.value.comparisons.forEach((el) => {
            aCounter += (data.get(el.index).score < a.value
                .score ? -1 : 1);
        })
        b.value.comparisons.forEach((el) => {
            bCounter += (data.get(el.index).score < b.value
                .score ? -1 : 1);
        })
        return Math.abs(bCounter) - Math.abs(aCounter) ||
           Math.abs(b.value.proposal) - Math.abs(a.value.
           proposal);
    }).filter(el => el.value.comparisons.length > 2 && Math
       .abs(el.value.proposal) > 0.75).map((el) => el.key);
}

export const calculateProposalScore = (alpha, data) => {
    let res = 0;
    alpha.comparisons.forEach(comp => {
        const beta = data.get(comp.index);
        Number(alpha.score) > Number(beta.score) ? res -=
           Number(comp.sim) : res += Number(comp.sim);
    });
    return alpha.comparisons.length > 0 ? res / alpha.
       comparisons.length : res;
}
```

A8 - Client side polling

```
import { useSetRecoilState } from "recoil";
import { amountOfQuestionsState, idState, metricsState,
   resultsState } from "./state";
import { structureData } from "./utils";

export const usePoller = () => {
    const setId = useSetRecoilState(idState);
    const setData = useSetRecoilState(resultsState);
    const setMetrics = useSetRecoilState(metricsState);
    const setAmountOfQuestions = useSetRecoilState(
       amountOfQuestionsState);
```

```
const poller = async (data) => {
    let url = "http://129.241.106.76:5000/upload";
    await fetch(url, {
        method: "POST",
        body: data,
    }).then((response) => {
        response.json().then((res) => {
            console.log(res);
            setId(res.id);
            let id = res.id;
            setAmountOfQuestions(res.valid_questions.
               length)
            let question = 1;
            let interval = setInterval(async () => {
                let url =
                    "http://129.241.106.76:5000/poll?id
                       =" +
                    id +
                    "&question=" +
                    question;
                await fetch(url, {
                    method: "GET",
                }).then((response) => {
                    response.json().then((res) => {
                        console.log(res);
                        if (res !== false) {
                            setData((arr) => [...arr,
                               structureData(res)]);
                            question += 1;
                        }
                    });
                });
                if (question > res.valid_questions.
                   length) {
                    clearInterval(interval);
                }
            }, 10000);

            let metricsQuestion = 1;
            let metricsInterval = setInterval(async ()
               => {
                let url = "http://129.241.106.76:5000/
                   poll_metrics?id=" + id + "&question=
                   " + metricsQuestion;
                await fetch(url, { method: "GET", }).
                   then((response) => {
                    response.json().then((res) => {
                            console.log(res);
                            if (res !== false) {
                                setMetrics((arr) => new Map
                                   (arr.set(metricsQuestion
                                   , res)));
                                metricsQuestion += 1;
                            }
                        });
                    });
                    if (metricsQuestion > res.
                       valid_questions.length) {
                        clearInterval(metricsInterval);
                    }
                }, 10000);
            });
        });
    };
    return poller;
};

export const useOnlyPoller = () => {
    const setId = useSetRecoilState(idState);
    const setData = useSetRecoilState(resultsState);
    const setMetrics = useSetRecoilState(metricsState);
    const setAmountOfQuestions = useSetRecoilState(
       amountOfQuestionsState);

    const poller = async (id) => {
        console.log(id);
        setId(id);
        let question = 1;
        let interval = setInterval(async () => {
            let url =
                "http://129.241.106.76:5000/poll?id=" +
                id +
                "&question=" +
                question;
            await fetch(url, {
                method: "GET",
            }).then((response) => {
                response.json().then((res) => {
                    console.log(res);
                    if (res !== false) {
                        setData((arr) => [...arr,
                           structureData(res)]);
                        question += 1;
                    }else{
                        clearInterval(interval);
                }
            });
        });
    }, 1000);
    setAmountOfQuestions(question -1);

    let metricsQuestion = 1;
    let metricsInterval = setInterval(async () => {
        let url = "http://129.241.106.76:5000/
          poll_metrics?id=" + id + "&question=" +
          metricsQuestion;
        await fetch(url, { method: "GET", }).then((
          response) => {
            response.json().then((res) => {
                console.log(res);
                if (res !== false) {
                    setMetrics((arr) => new Map(arr.set
                      (metricsQuestion, res)));
                    metricsQuestion += 1;
                }else{
                    clearInterval(metricsInterval);
                }
            });
        });
    }, 1000);
};

return poller;
};
```

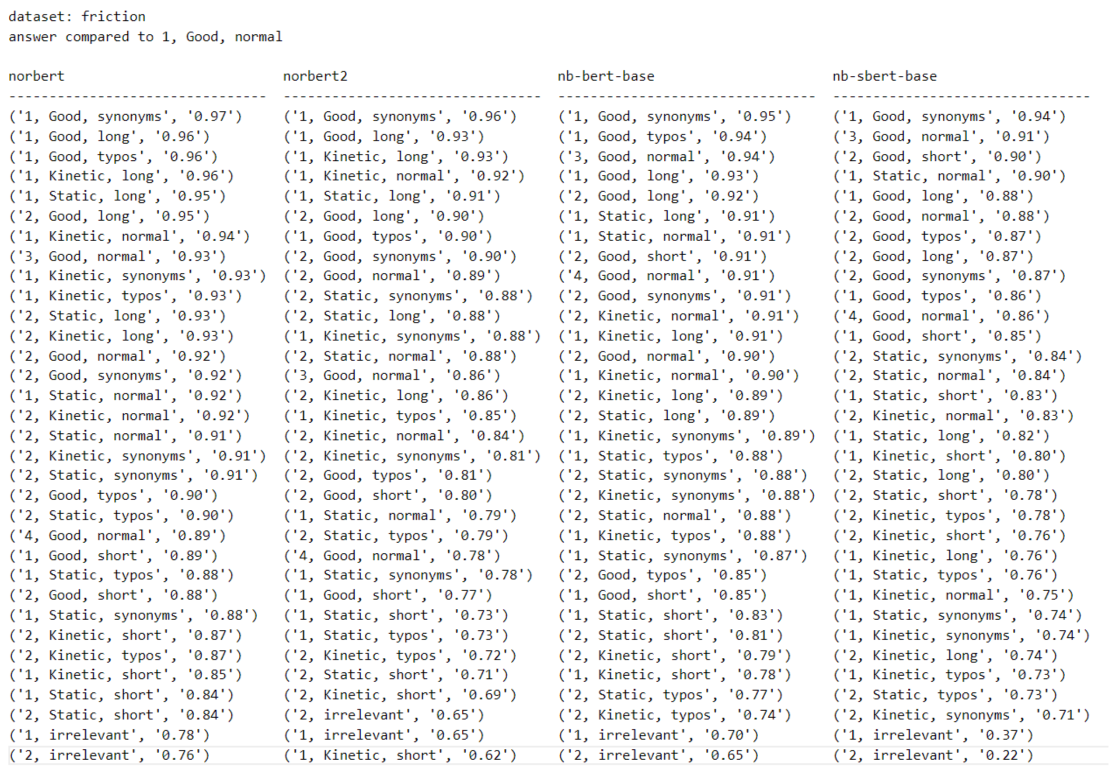
A9 - OpenAI poller

```
export const useThemes = () => {
  const themes = async (answer) => {
    const requestOptions = {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
        Authorization:
          "Bearer␣{API_KEY}",
      },
      body: JSON.stringify({
        prompt: 'Hva␣er␣temaene␣for␣denne␣teksten:␣"' +
           answer + '"',
        temperature: 0.6,
        max_tokens: 500,
      }),
    };
    const response = await fetch(
      "https://api.openai.com/v1/engines/text-davinci-003/
         completions",
      requestOptions
    );
    const data = await response.json();

    return data.choices[0].text;
  };
  return themes;
};

export const useKeywords = () => {
  const keywords = async (answer) => {
    const requestOptions = {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
        Authorization:
          "Bearer␣{API_KEY}",
      },
      body: JSON.stringify({
        prompt: 'Nevn␣stikkordene␣for␣denne␣teksten:␣"' +
           answer + '"',
        temperature: 0.6,
        max_tokens: 500,
      }),
    };
    const response = await fetch(
        "https://api.openai.com/v1/engines/text-davinci
           -003/completions",
        requestOptions
      );
      const data = await response.json();

      return data.choices[0].text;
  };

  return keywords;
};

export const useSummary = () => {
  const summary = async (answer) => {
    const requestOptions = {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
          Authorization:
            "Bearer␣{API_KEY}",
        },
        body: JSON.stringify({
          prompt: 'Skriv␣et␣sammendrag␣av␣denne␣teksten:␣"' +
              answer + '"',
          temperature: 0.6,
          max_tokens: 500,
        }),
      };
      const response = await fetch(
          "https://api.openai.com/v1/engines/text-davinci
             -003/completions",
          requestOptions
        );
        const data = await response.json();

        return data.choices[0].text;
    };

  return summary;
};
```

B - RESULTS FROM BERT MODELS TESTS

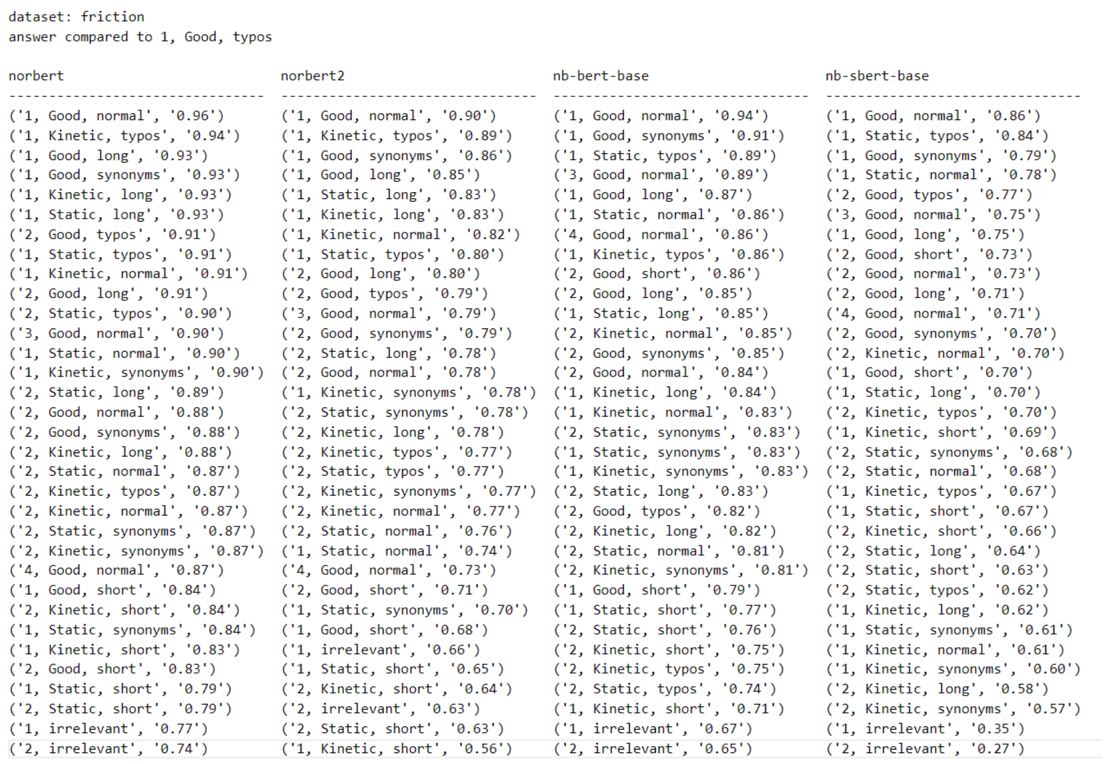
B1 - Friction dataset

All results from the tests performed on the friction dataset.



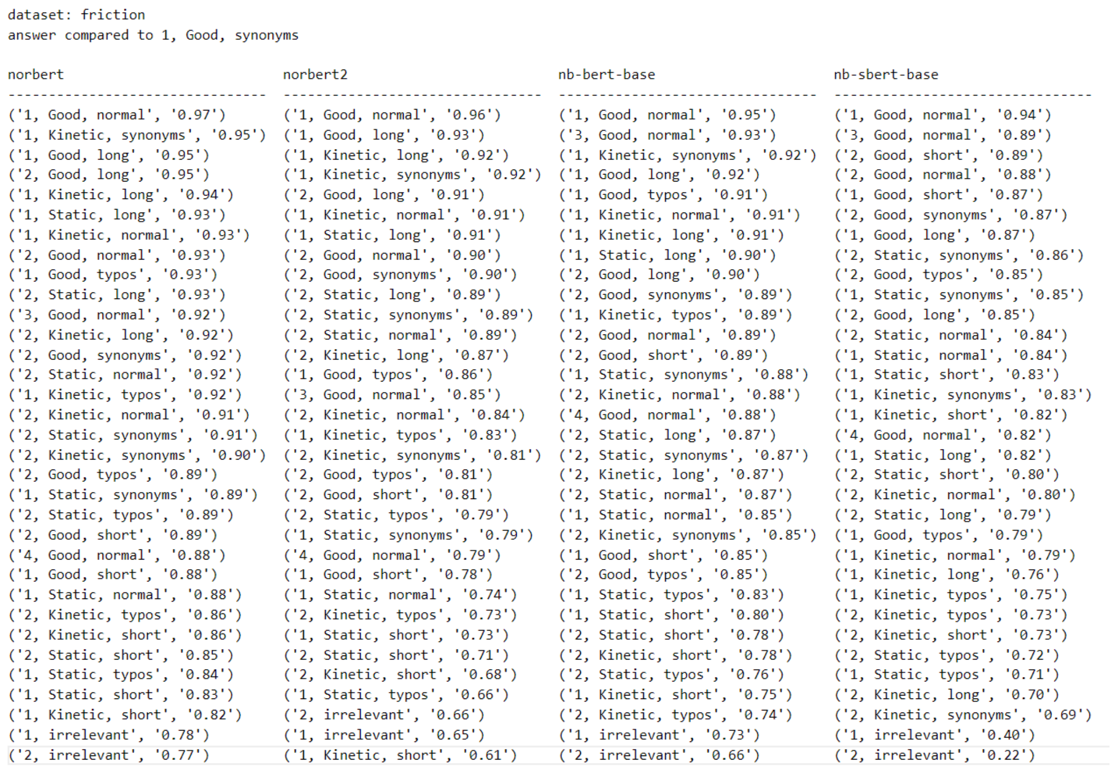
```
dataset: friction
answer compared to 1, Good, normal

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
--------------------------------   --------------------------------   --------------------------------   --------------------------------
('1, Good, synonyms', '0.97')      ('1, Good, synonyms', '0.96')      ('1, Good, synonyms', '0.95')      ('1, Good, synonyms', '0.94')
('1, Good, long', '0.96')          ('1, Good, long', '0.93')          ('1, Good, typos', '0.94')         ('3, Good, normal', '0.91')
('1, Good, typos', '0.96')         ('1, Kinetic, long', '0.93')       ('3, Good, normal', '0.94')        ('2, Good, short', '0.90')
('1, Kinetic, long', '0.96')       ('1, Kinetic, normal', '0.92')     ('1, Good, long', '0.93')          ('1, Static, normal', '0.90')
('1, Static, long', '0.95')        ('1, Static, long', '0.91')        ('2, Good, long', '0.92')          ('1, Good, long', '0.88')
('2, Good, long', '0.95')          ('2, Good, long', '0.90')          ('1, Static, long', '0.91')        ('2, Good, normal', '0.88')
('1, Kinetic, normal', '0.94')     ('1, Good, typos', '0.90')         ('1, Static, normal', '0.91')      ('2, Good, typos', '0.87')
('3, Good, normal', '0.93')        ('2, Good, synonyms', '0.90')      ('2, Good, short', '0.91')         ('2, Good, long', '0.87')
('1, Kinetic, synonyms', '0.93')   ('2, Good, normal', '0.89')        ('4, Good, normal', '0.91')        ('2, Good, synonyms', '0.87')
('1, Kinetic, typos', '0.93')      ('2, Static, synonyms', '0.88')    ('2, Good, synonyms', '0.91')      ('1, Good, typos', '0.86')
('2, Static, long', '0.93')        ('2, Static, long', '0.88')        ('2, Kinetic, normal', '0.91')     ('4, Good, normal', '0.86')
('2, Kinetic, long', '0.93')       ('1, Kinetic, synonyms', '0.88')   ('1, Kinetic, long', '0.91')       ('1, Good, short', '0.85')
('2, Good, normal', '0.92')        ('2, Static, normal', '0.88')      ('2, Good, normal', '0.90')        ('2, Static, synonyms', '0.84')
('2, Good, synonyms', '0.92')      ('3, Good, normal', '0.86')        ('1, Kinetic, normal', '0.90')     ('2, Static, normal', '0.84')
('1, Static, normal', '0.92')      ('2, Kinetic, long', '0.86')       ('2, Kinetic, long', '0.89')       ('1, Static, short', '0.83')
('2, Kinetic, normal', '0.92')     ('1, Kinetic, typos', '0.85')      ('2, Static, long', '0.89')        ('2, Kinetic, normal', '0.83')
('2, Static, normal', '0.91')      ('2, Kinetic, normal', '0.84')     ('1, Kinetic, synonyms', '0.89')   ('1, Static, long', '0.82')
('2, Kinetic, synonyms', '0.91')   ('2, Kinetic, synonyms', '0.81')   ('1, Static, typos', '0.88')       ('1, Kinetic, short', '0.80')
('2, Static, synonyms', '0.91')    ('2, Good, typos', '0.81')         ('2, Static, synonyms', '0.88')    ('2, Static, long', '0.80')
('2, Good, typos', '0.90')         ('2, Good, short', '0.80')         ('2, Kinetic, synonyms', '0.88')   ('2, Static, short', '0.78')
('2, Static, typos', '0.90')       ('1, Static, normal', '0.79')      ('2, Static, normal', '0.88')      ('2, Kinetic, typos', '0.78')
('4, Good, normal', '0.89')        ('2, Static, typos', '0.79')       ('1, Kinetic, typos', '0.88')      ('2, Kinetic, short', '0.76')
('1, Good, short', '0.89')         ('4, Good, normal', '0.78')        ('1, Static, synonyms', '0.87')    ('1, Kinetic, long', '0.76')
('1, Static, typos', '0.88')       ('1, Static, synonyms', '0.78')    ('2, Good, typos', '0.85')         ('1, Static, typos', '0.76')
('2, Good, short', '0.88')         ('1, Good, short', '0.77')         ('1, Good, short', '0.85')         ('1, Kinetic, normal', '0.75')
('1, Static, synonyms', '0.88')    ('1, Static, short', '0.73')       ('1, Static, short', '0.83')       ('1, Static, synonyms', '0.74')
('2, Kinetic, short', '0.87')      ('1, Static, typos', '0.73')       ('2, Static, short', '0.81')       ('1, Kinetic, synonyms', '0.74')
('2, Kinetic, typos', '0.87')      ('2, Kinetic, typos', '0.72')      ('2, Kinetic, short', '0.79')      ('2, Kinetic, long', '0.74')
('1, Kinetic, short', '0.85')      ('2, Static, short', '0.71')       ('1, Kinetic, short', '0.78')      ('1, Kinetic, typos', '0.73')
('1, Static, short', '0.84')       ('2, Kinetic, short', '0.69')      ('2, Static, typos', '0.77')       ('2, Static, typos', '0.73')
('2, Static, short', '0.84')       ('2, irrelevant', '0.65')          ('2, Kinetic, typos', '0.74')      ('2, Kinetic, synonyms', '0.71')
('1, irrelevant', '0.78')          ('1, irrelevant', '0.65')          ('1, irrelevant', '0.70')          ('1, irrelevant', '0.37')
('2, irrelevant', '0.76')          ('1, Kinetic, short', '0.62')      ('2, irrelevant', '0.65')          ('2, irrelevant', '0.22')
```

Figure .1: The results from comparison to answer labeled "1, Good, normal".



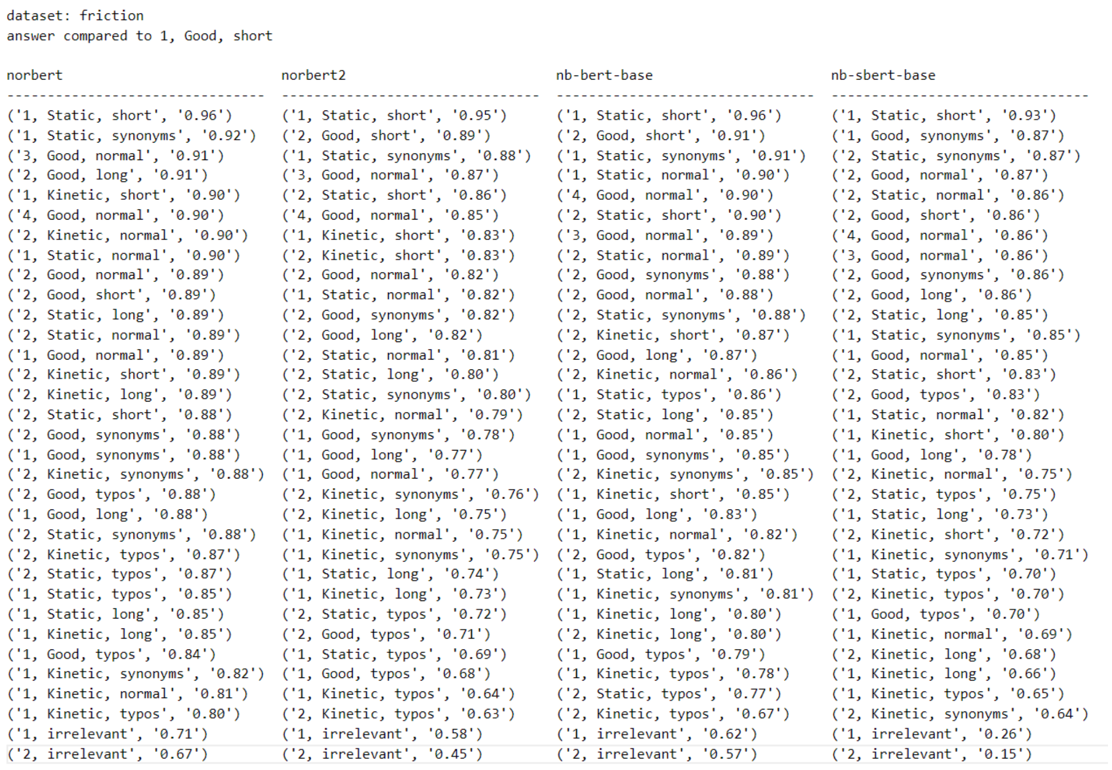
```
dataset: friction
answer compared to 1, Good, typos

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
--------------------------------   --------------------------------   --------------------------------   --------------------------------
('1, Good, normal', '0.96')        ('1, Good, normal', '0.90')        ('1, Good, normal', '0.94')        ('1, Good, normal', '0.86')
('1, Kinetic, typos', '0.94')      ('1, Kinetic, typos', '0.89')      ('1, Good, synonyms', '0.91')      ('1, Static, typos', '0.84')
('1, Good, long', '0.93')          ('1, Good, synonyms', '0.86')      ('1, Static, typos', '0.89')       ('1, Good, synonyms', '0.79')
('1, Good, synonyms', '0.93')      ('1, Good, long', '0.85')          ('3, Good, normal', '0.89')        ('1, Static, normal', '0.78')
('1, Kinetic, long', '0.93')       ('1, Static, long', '0.83')        ('1, Good, long', '0.87')          ('2, Good, typos', '0.77')
('1, Static, long', '0.93')        ('1, Kinetic, long', '0.83')       ('1, Static, normal', '0.86')      ('3, Good, normal', '0.75')
('2, Good, typos', '0.91')         ('1, Kinetic, normal', '0.82')     ('4, Good, normal', '0.86')        ('1, Good, long', '0.75')
('1, Static, typos', '0.91')       ('1, Static, typos', '0.80')       ('1, Kinetic, typos', '0.86')      ('2, Good, short', '0.73')
('1, Kinetic, normal', '0.91')     ('2, Good, long', '0.80')          ('2, Good, short', '0.86')         ('2, Good, normal', '0.73')
('2, Good, long', '0.91')          ('2, Good, typos', '0.79')         ('2, Good, long', '0.85')          ('2, Good, long', '0.71')
('2, Static, typos', '0.90')       ('3, Good, normal', '0.79')        ('1, Static, long', '0.85')        ('4, Good, normal', '0.71')
('3, Good, normal', '0.90')        ('2, Good, synonyms', '0.79')      ('2, Kinetic, normal', '0.85')     ('2, Good, synonyms', '0.70')
('1, Static, normal', '0.90')      ('2, Static, long', '0.78')        ('2, Good, synonyms', '0.85')      ('2, Kinetic, normal', '0.70')
('1, Kinetic, synonyms', '0.90')   ('2, Good, normal', '0.78')        ('2, Good, normal', '0.84')        ('1, Good, short', '0.70')
('2, Static, long', '0.89')        ('1, Kinetic, synonyms', '0.78')   ('1, Kinetic, long', '0.84')       ('1, Static, long', '0.70')
('2, Good, normal', '0.88')        ('2, Static, synonyms', '0.78')    ('1, Kinetic, normal', '0.83')     ('2, Kinetic, typos', '0.70')
('2, Good, synonyms', '0.88')      ('2, Kinetic, long', '0.78')       ('2, Static, synonyms', '0.83')    ('1, Kinetic, short', '0.69')
('2, Kinetic, long', '0.88')       ('2, Kinetic, typos', '0.77')      ('1, Static, synonyms', '0.83')    ('2, Static, synonyms', '0.68')
('2, Static, normal', '0.87')      ('2, Static, typos', '0.77')       ('1, Kinetic, synonyms', '0.83')   ('2, Static, normal', '0.68')
('2, Kinetic, typos', '0.87')      ('2, Kinetic, synonyms', '0.77')   ('2, Static, long', '0.83')        ('1, Kinetic, typos', '0.67')
('2, Kinetic, normal', '0.87')     ('2, Kinetic, normal', '0.77')     ('2, Good, typos', '0.82')         ('1, Static, short', '0.67')
('2, Static, synonyms', '0.87')    ('2, Static, normal', '0.76')      ('2, Kinetic, long', '0.82')       ('2, Kinetic, short', '0.66')
('2, Kinetic, synonyms', '0.87')   ('1, Static, normal', '0.74')      ('2, Static, normal', '0.81')      ('2, Static, long', '0.64')
('4, Good, normal', '0.87')        ('4, Good, normal', '0.73')        ('2, Kinetic, synonyms', '0.81')   ('2, Static, short', '0.63')
('1, Good, short', '0.84')         ('2, Good, short', '0.71')         ('1, Good, short', '0.79')         ('2, Static, typos', '0.62')
('2, Kinetic, short', '0.84')      ('1, Static, synonyms', '0.70')    ('1, Static, short', '0.77')       ('1, Kinetic, long', '0.62')
('1, Static, synonyms', '0.84')    ('1, Good, short', '0.68')         ('2, Static, short', '0.76')       ('1, Static, synonyms', '0.61')
('1, Kinetic, short', '0.83')      ('1, irrelevant', '0.66')          ('2, Kinetic, short', '0.75')      ('1, Kinetic, normal', '0.61')
('2, Good, short', '0.83')         ('1, Static, short', '0.65')       ('2, Kinetic, typos', '0.75')      ('1, Kinetic, synonyms', '0.60')
('1, Static, short', '0.79')       ('2, Kinetic, short', '0.64')      ('2, Static, typos', '0.74')       ('2, Kinetic, long', '0.58')
('2, Static, short', '0.79')       ('2, irrelevant', '0.63')          ('1, Kinetic, short', '0.71')      ('2, Kinetic, synonyms', '0.57')
('1, irrelevant', '0.77')          ('2, Static, short', '0.63')       ('1, irrelevant', '0.67')          ('1, irrelevant', '0.35')
('2, irrelevant', '0.74')          ('1, Kinetic, short', '0.56')      ('2, irrelevant', '0.65')          ('2, irrelevant', '0.27')
```

Figure .2: The results from comparison to answer labeled "1, Good, typos".



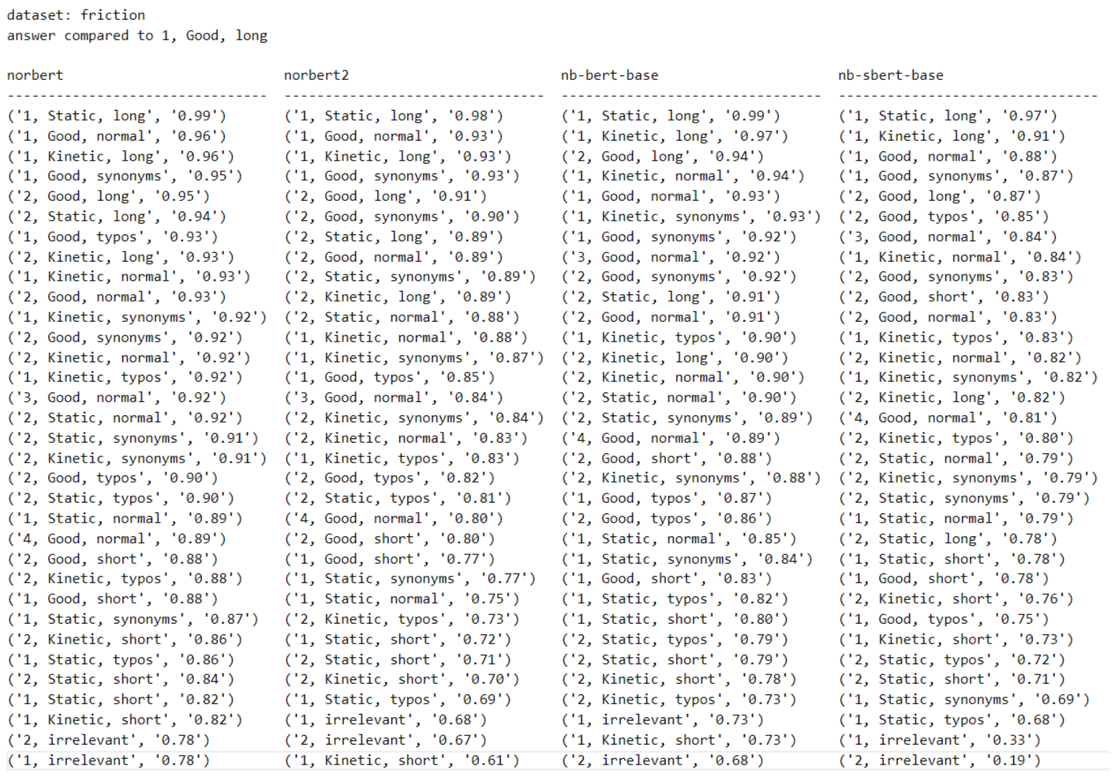
```
dataset: friction
answer compared to 1, Good, synonyms

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
--------------------------------   --------------------------------   --------------------------------   --------------------------------
('1, Good, normal', '0.97')        ('1, Good, normal', '0.96')        ('1, Good, normal', '0.95')        ('1, Good, normal', '0.94')
('1, Kinetic, synonyms', '0.95')   ('1, Good, long', '0.93')          ('3, Good, normal', '0.93')        ('3, Good, normal', '0.89')
('1, Good, long', '0.95')          ('1, Kinetic, long', '0.92')       ('1, Kinetic, synonyms', '0.92')   ('2, Good, short', '0.89')
('2, Good, long', '0.95')          ('1, Kinetic, synonyms', '0.92')   ('1, Good, long', '0.92')          ('2, Good, normal', '0.88')
('1, Kinetic, long', '0.94')       ('2, Good, long', '0.91')          ('1, Good, typos', '0.91')         ('1, Good, short', '0.87')
('1, Static, long', '0.93')        ('1, Kinetic, normal', '0.91')     ('1, Kinetic, normal', '0.91')     ('2, Good, synonyms', '0.87')
('1, Kinetic, normal', '0.93')     ('1, Static, long', '0.91')        ('1, Kinetic, long', '0.91')       ('1, Good, long', '0.87')
('2, Good, normal', '0.93')        ('2, Good, normal', '0.90')        ('1, Static, long', '0.90')        ('2, Static, synonyms', '0.86')
('1, Good, typos', '0.93')         ('2, Good, synonyms', '0.90')      ('2, Good, long', '0.90')          ('2, Good, typos', '0.85')
('2, Static, long', '0.93')        ('2, Static, long', '0.89')        ('2, Good, synonyms', '0.89')      ('1, Static, synonyms', '0.85')
('3, Good, normal', '0.92')        ('2, Static, synonyms', '0.89')    ('1, Kinetic, typos', '0.89')      ('2, Good, long', '0.85')
('2, Kinetic, long', '0.92')       ('2, Static, normal', '0.89')      ('2, Good, normal', '0.89')        ('2, Static, normal', '0.84')
('2, Good, synonyms', '0.92')      ('2, Kinetic, long', '0.87')       ('2, Good, short', '0.89')         ('1, Static, normal', '0.84')
('2, Static, normal', '0.92')      ('1, Good, typos', '0.86')         ('1, Static, synonyms', '0.88')    ('1, Static, short', '0.83')
('1, Kinetic, typos', '0.92')      ('3, Good, normal', '0.85')        ('2, Kinetic, normal', '0.88')     ('1, Kinetic, synonyms', '0.83')
('2, Kinetic, normal', '0.91')     ('2, Kinetic, normal', '0.84')     ('4, Good, normal', '0.88')        ('1, Kinetic, short', '0.82')
('2, Static, synonyms', '0.91')    ('1, Kinetic, typos', '0.83')      ('2, Static, long', '0.87')        ('4, Good, normal', '0.82')
('2, Kinetic, synonyms', '0.90')   ('2, Kinetic, synonyms', '0.81')   ('2, Static, synonyms', '0.87')    ('1, Static, long', '0.82')
('2, Good, typos', '0.89')         ('2, Good, typos', '0.81')         ('2, Kinetic, long', '0.87')       ('2, Static, short', '0.80')
('1, Static, synonyms', '0.89')    ('2, Good, short', '0.81')         ('2, Static, normal', '0.87')      ('2, Kinetic, normal', '0.80')
('2, Static, typos', '0.89')       ('2, Static, typos', '0.79')       ('1, Static, normal', '0.85')      ('2, Static, long', '0.79')
('2, Good, short', '0.89')         ('1, Static, synonyms', '0.79')    ('2, Kinetic, synonyms', '0.85')   ('1, Good, typos', '0.79')
('4, Good, normal', '0.88')        ('4, Good, normal', '0.79')        ('1, Good, short', '0.85')         ('1, Kinetic, normal', '0.79')
('1, Good, short', '0.88')         ('1, Good, short', '0.78')         ('2, Good, typos', '0.85')         ('1, Kinetic, long', '0.76')
('1, Static, normal', '0.88')      ('1, Static, normal', '0.74')      ('1, Static, typos', '0.83')       ('1, Kinetic, typos', '0.75')
('2, Kinetic, typos', '0.86')      ('2, Kinetic, typos', '0.73')      ('1, Static, short', '0.80')       ('2, Kinetic, typos', '0.73')
('2, Kinetic, short', '0.86')      ('1, Static, short', '0.73')       ('2, Static, short', '0.78')       ('2, Kinetic, short', '0.73')
('2, Static, short', '0.85')       ('2, Static, short', '0.71')       ('2, Kinetic, short', '0.78')      ('2, Static, typos', '0.72')
('1, Static, typos', '0.84')       ('2, Kinetic, short', '0.68')      ('2, Static, typos', '0.76')       ('1, Static, typos', '0.71')
('1, Static, short', '0.83')       ('1, Static, typos', '0.66')       ('1, Kinetic, short', '0.75')      ('2, Kinetic, long', '0.70')
('1, Kinetic, short', '0.82')      ('2, irrelevant', '0.66')          ('2, Kinetic, typos', '0.74')      ('2, Kinetic, synonyms', '0.69')
('1, irrelevant', '0.78')          ('1, irrelevant', '0.65')          ('1, irrelevant', '0.73')          ('1, irrelevant', '0.40')
('2, irrelevant', '0.77')          ('1, Kinetic, short', '0.61')      ('2, irrelevant', '0.66')          ('2, irrelevant', '0.22')
```

Figure .3: The results from comparison to answer labeled "1, Good, synonyms".



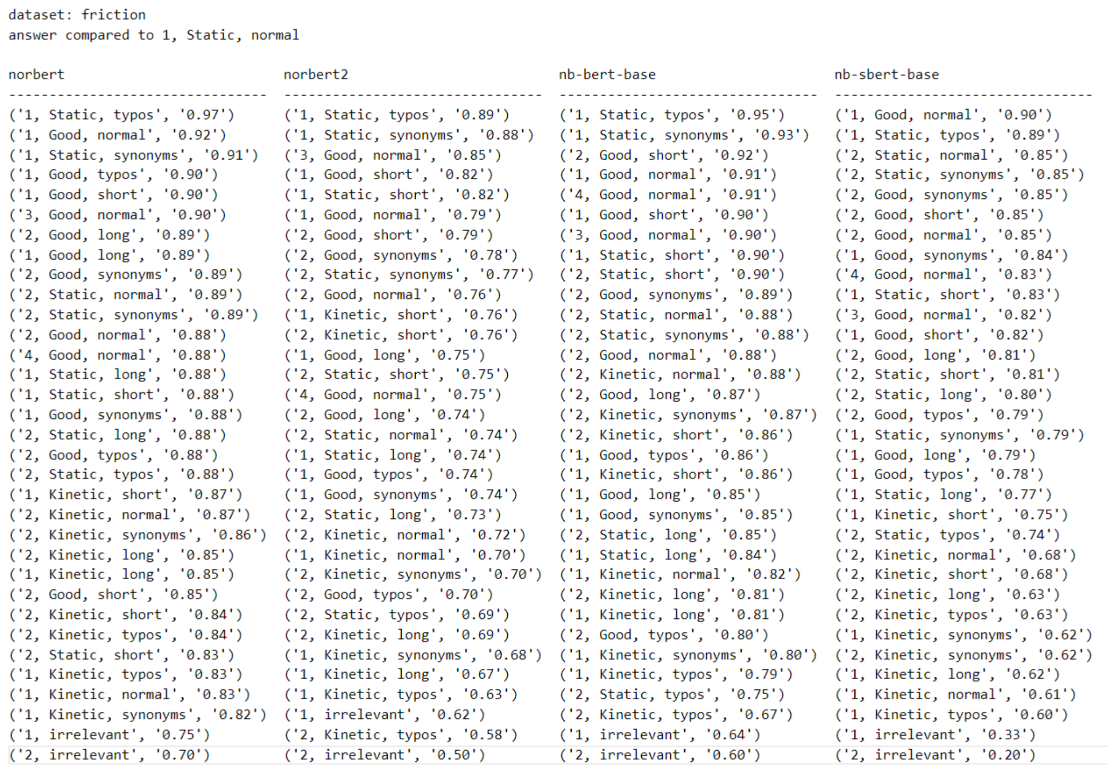
```
dataset: friction
answer compared to 1, Good, short

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Static, short', '0.96')      ('1, Static, short', '0.95')      ('1, Static, short', '0.96')      ('1, Static, short', '0.93')
('1, Static, synonyms', '0.92')   ('2, Good, short', '0.89')        ('2, Good, short', '0.91')        ('1, Good, synonyms', '0.87')
('3, Good, normal', '0.91')       ('1, Static, synonyms', '0.88')   ('1, Static, synonyms', '0.91')   ('2, Static, synonyms', '0.87')
('2, Good, long', '0.91')         ('3, Good, normal', '0.87')       ('1, Static, normal', '0.90')     ('2, Good, normal', '0.87')
('1, Kinetic, short', '0.90')     ('2, Static, short', '0.86')      ('4, Good, normal', '0.90')       ('2, Static, normal', '0.86')
('4, Good, normal', '0.90')       ('4, Good, normal', '0.85')       ('2, Static, short', '0.90')      ('2, Good, short', '0.86')
('2, Kinetic, normal', '0.90')    ('1, Kinetic, short', '0.83')     ('3, Good, normal', '0.89')       ('4, Good, normal', '0.86')
('1, Static, normal', '0.90')     ('2, Kinetic, short', '0.83')     ('2, Static, normal', '0.89')     ('3, Good, normal', '0.86')
('2, Good, normal', '0.89')       ('2, Good, normal', '0.82')       ('2, Good, synonyms', '0.88')     ('2, Good, synonyms', '0.86')
('2, Good, short', '0.89')        ('1, Static, normal', '0.82')     ('2, Good, normal', '0.88')       ('2, Good, long', '0.86')
('2, Static, long', '0.89')       ('2, Good, synonyms', '0.82')     ('2, Static, synonyms', '0.88')   ('2, Static, long', '0.85')
('2, Static, normal', '0.89')     ('2, Good, long', '0.82')         ('2, Kinetic, short', '0.87')     ('1, Static, synonyms', '0.85')
('1, Good, normal', '0.89')       ('2, Static, normal', '0.81')     ('2, Good, long', '0.87')         ('1, Good, normal', '0.85')
('2, Kinetic, short', '0.89')     ('2, Static, long', '0.80')       ('2, Kinetic, normal', '0.86')    ('2, Static, short', '0.83')
('2, Kinetic, long', '0.89')      ('2, Static, synonyms', '0.80')   ('1, Static, typos', '0.86')      ('2, Good, typos', '0.83')
('2, Static, short', '0.88')      ('2, Kinetic, normal', '0.79')    ('2, Static, long', '0.85')       ('1, Static, normal', '0.82')
('2, Good, synonyms', '0.88')     ('1, Good, synonyms', '0.78')     ('1, Good, normal', '0.85')       ('1, Kinetic, short', '0.80')
('1, Good, synonyms', '0.88')     ('1, Good, long', '0.77')         ('1, Good, synonyms', '0.85')     ('1, Good, long', '0.78')
('2, Kinetic, synonyms', '0.88')  ('1, Good, normal', '0.77')       ('2, Kinetic, synonyms', '0.85')  ('2, Kinetic, normal', '0.75')
('2, Good, typos', '0.88')        ('2, Kinetic, synonyms', '0.76')  ('1, Kinetic, short', '0.85')     ('2, Static, typos', '0.75')
('1, Good, long', '0.88')         ('2, Kinetic, long', '0.75')      ('1, Good, long', '0.83')         ('1, Static, long', '0.73')
('2, Static, synonyms', '0.88')   ('1, Kinetic, normal', '0.75')    ('1, Kinetic, normal', '0.82')    ('2, Kinetic, short', '0.72')
('2, Kinetic, typos', '0.87')     ('1, Kinetic, synonyms', '0.75')  ('2, Good, typos', '0.82')        ('1, Kinetic, synonyms', '0.71')
('2, Static, typos', '0.87')      ('1, Static, long', '0.74')       ('1, Static, long', '0.81')       ('1, Static, typos', '0.70')
('1, Static, typos', '0.85')      ('1, Kinetic, long', '0.73')      ('1, Kinetic, synonyms', '0.81')  ('2, Kinetic, typos', '0.70')
('1, Static, long', '0.85')       ('2, Static, typos', '0.72')      ('1, Kinetic, long', '0.80')      ('1, Good, typos', '0.70')
('1, Kinetic, long', '0.85')      ('2, Good, typos', '0.71')        ('2, Kinetic, long', '0.80')      ('1, Kinetic, normal', '0.69')
('1, Good, typos', '0.84')        ('1, Static, typos', '0.69')      ('1, Good, typos', '0.79')        ('2, Kinetic, long', '0.68')
('1, Kinetic, synonyms', '0.82')  ('1, Good, typos', '0.68')        ('1, Kinetic, typos', '0.78')     ('1, Kinetic, long', '0.66')
('1, Kinetic, normal', '0.81')    ('1, Kinetic, typos', '0.64')     ('2, Static, typos', '0.77')      ('1, Kinetic, typos', '0.65')
('1, Kinetic, typos', '0.80')     ('2, Kinetic, typos', '0.63')     ('2, Kinetic, typos', '0.67')     ('2, Kinetic, synonyms', '0.64')
('1, irrelevant', '0.71')         ('1, irrelevant', '0.58')         ('1, irrelevant', '0.62')         ('1, irrelevant', '0.26')
('2, irrelevant', '0.67')         ('2, irrelevant', '0.45')         ('2, irrelevant', '0.57')         ('2, irrelevant', '0.15')
```

Figure .4: The results from comparison to answer labeled "1, Good, short".



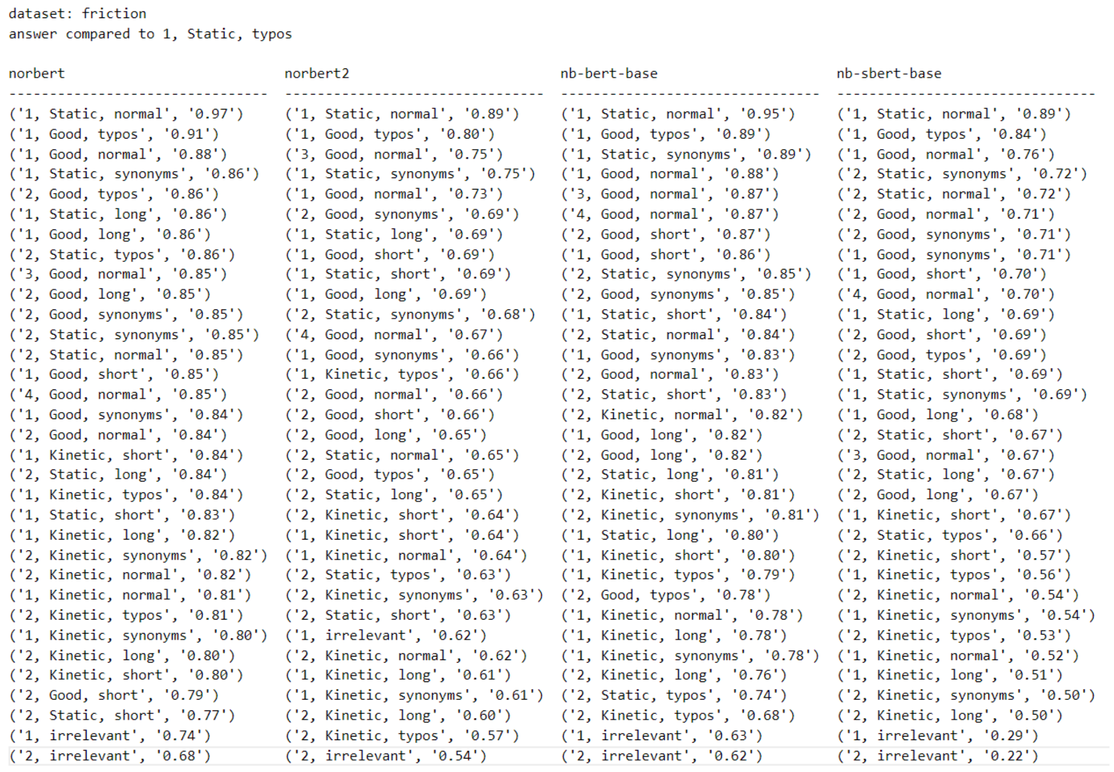
```
dataset: friction
answer compared to 1, Good, long

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Static, long', '0.99')       ('1, Static, long', '0.98')       ('1, Static, long', '0.99')       ('1, Static, long', '0.97')
('1, Good, normal', '0.96')       ('1, Good, normal', '0.93')       ('1, Kinetic, long', '0.97')      ('1, Kinetic, long', '0.91')
('1, Kinetic, long', '0.96')      ('1, Kinetic, long', '0.93')      ('2, Good, long', '0.94')         ('1, Good, normal', '0.88')
('1, Good, synonyms', '0.95')     ('1, Good, synonyms', '0.93')     ('1, Kinetic, normal', '0.94')    ('1, Good, synonyms', '0.87')
('2, Good, long', '0.95')         ('2, Good, long', '0.91')         ('1, Good, normal', '0.93')       ('2, Good, long', '0.87')
('2, Static, long', '0.94')       ('2, Good, synonyms', '0.90')     ('1, Kinetic, synonyms', '0.93')  ('2, Good, typos', '0.85')
('1, Good, typos', '0.93')        ('2, Static, long', '0.89')       ('1, Good, synonyms', '0.92')     ('3, Good, normal', '0.84')
('2, Kinetic, long', '0.93')      ('2, Good, normal', '0.89')       ('3, Good, normal', '0.92')       ('1, Kinetic, normal', '0.84')
('1, Kinetic, normal', '0.93')    ('2, Static, synonyms', '0.89')   ('2, Good, synonyms', '0.92')     ('2, Good, synonyms', '0.83')
('2, Good, normal', '0.93')       ('2, Kinetic, long', '0.89')      ('2, Static, long', '0.91')       ('2, Good, short', '0.83')
('1, Kinetic, synonyms', '0.92')  ('2, Static, normal', '0.88')     ('2, Good, normal', '0.91')       ('2, Good, normal', '0.83')
('2, Good, synonyms', '0.92')     ('1, Kinetic, normal', '0.88')    ('1, Kinetic, typos', '0.90')     ('1, Kinetic, typos', '0.83')
('2, Kinetic, normal', '0.92')    ('1, Kinetic, synonyms', '0.87')  ('2, Kinetic, long', '0.90')      ('2, Kinetic, normal', '0.82')
('1, Kinetic, typos', '0.92')     ('1, Good, typos', '0.85')        ('2, Kinetic, normal', '0.90')    ('1, Kinetic, synonyms', '0.82')
('3, Good, normal', '0.92')       ('3, Good, normal', '0.84')       ('2, Static, normal', '0.90')     ('2, Kinetic, long', '0.82')
('2, Static, normal', '0.92')     ('2, Kinetic, synonyms', '0.84')  ('2, Static, synonyms', '0.89')   ('4, Good, normal', '0.81')
('2, Static, synonyms', '0.91')   ('2, Kinetic, normal', '0.83')    ('4, Good, normal', '0.89')       ('2, Kinetic, typos', '0.80')
('2, Kinetic, synonyms', '0.91')  ('1, Kinetic, typos', '0.83')     ('2, Good, short', '0.88')        ('2, Static, normal', '0.79')
('2, Good, typos', '0.90')        ('2, Good, typos', '0.82')        ('2, Kinetic, synonyms', '0.88')  ('2, Kinetic, synonyms', '0.79')
('2, Static, typos', '0.90')      ('2, Static, typos', '0.81')      ('1, Good, typos', '0.87')        ('2, Static, synonyms', '0.79')
('1, Static, normal', '0.89')     ('4, Good, normal', '0.80')       ('2, Good, typos', '0.86')        ('1, Static, normal', '0.79')
('4, Good, normal', '0.89')       ('2, Good, short', '0.80')        ('1, Static, normal', '0.85')     ('2, Static, long', '0.78')
('2, Good, short', '0.88')        ('1, Good, short', '0.77')        ('1, Static, synonyms', '0.84')   ('1, Static, short', '0.78')
('2, Kinetic, typos', '0.88')     ('1, Static, synonyms', '0.77')   ('1, Good, short', '0.83')        ('1, Good, short', '0.78')
('1, Good, short', '0.88')        ('1, Static, normal', '0.75')     ('1, Static, typos', '0.82')      ('2, Kinetic, short', '0.76')
('1, Static, synonyms', '0.87')   ('2, Kinetic, typos', '0.73')     ('1, Static, short', '0.80')      ('1, Good, typos', '0.75')
('2, Kinetic, short', '0.86')     ('1, Static, short', '0.72')      ('2, Static, typos', '0.79')      ('1, Kinetic, short', '0.73')
('1, Static, typos', '0.86')      ('2, Static, short', '0.71')      ('2, Static, short', '0.79')      ('2, Static, typos', '0.72')
('2, Static, short', '0.84')      ('2, Kinetic, short', '0.70')     ('2, Kinetic, short', '0.78')     ('2, Static, short', '0.71')
('1, Static, short', '0.82')      ('1, Static, typos', '0.69')      ('2, Kinetic, typos', '0.73')     ('1, Static, synonyms', '0.69')
('1, Kinetic, short', '0.82')     ('1, irrelevant', '0.68')         ('1, irrelevant', '0.73')         ('1, Static, typos', '0.68')
('2, irrelevant', '0.78')         ('2, irrelevant', '0.67')         ('1, Kinetic, short', '0.73')     ('1, irrelevant', '0.33')
('1, irrelevant', '0.78')         ('1, Kinetic, short', '0.61')     ('2, irrelevant', '0.68')         ('2, irrelevant', '0.19')
```

Figure .5: The results from comparison to answer labeled "1, Good, long".



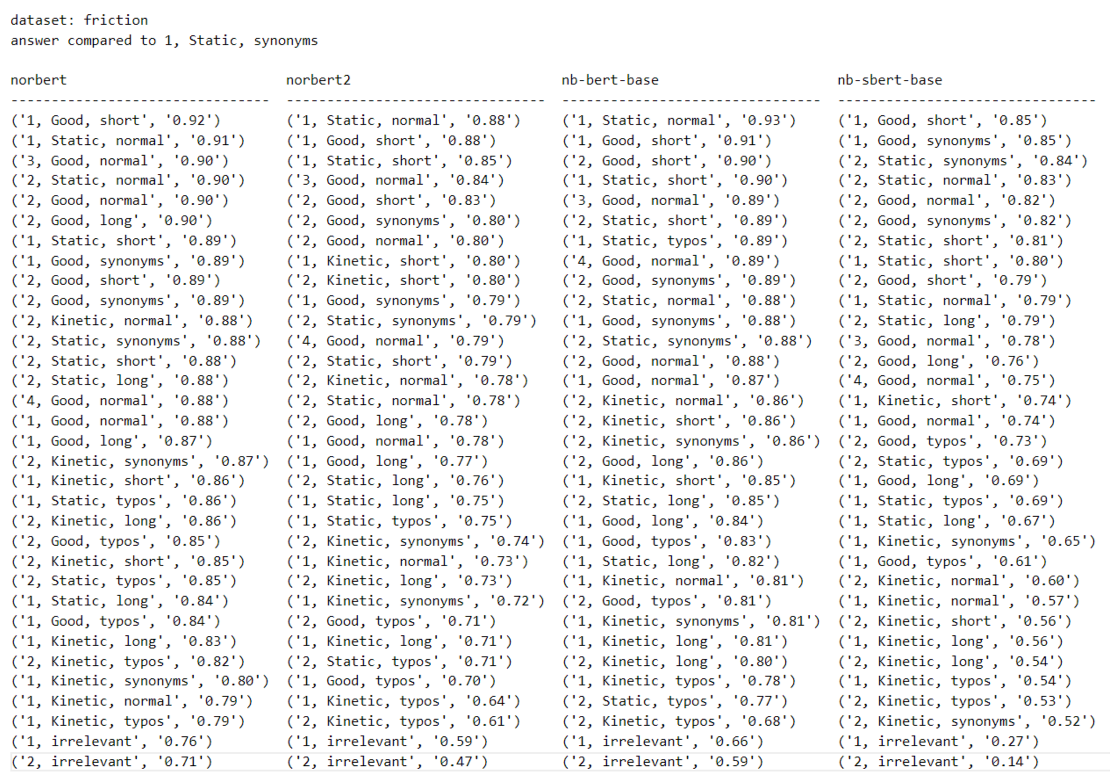
```
dataset: friction
answer compared to 1, Static, normal

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Static, typos', '0.97')      ('1, Static, typos', '0.89')      ('1, Static, typos', '0.95')      ('1, Good, normal', '0.90')
('1, Good, normal', '0.92')       ('1, Static, synonyms', '0.88')   ('1, Static, synonyms', '0.93')   ('1, Static, typos', '0.89')
('1, Static, synonyms', '0.91')   ('3, Good, normal', '0.85')       ('2, Good, short', '0.92')        ('2, Static, normal', '0.85')
('1, Good, typos', '0.90')        ('1, Good, short', '0.82')        ('1, Good, normal', '0.91')       ('2, Static, synonyms', '0.85')
('1, Good, short', '0.90')        ('1, Static, short', '0.82')      ('4, Good, normal', '0.91')       ('2, Good, synonyms', '0.85')
('3, Good, normal', '0.90')       ('1, Good, normal', '0.79')       ('1, Good, short', '0.90')        ('2, Good, short', '0.85')
('2, Good, long', '0.89')         ('2, Good, short', '0.79')        ('3, Good, normal', '0.90')       ('2, Good, normal', '0.85')
('1, Good, long', '0.89')         ('2, Good, synonyms', '0.78')     ('1, Static, short', '0.90')      ('1, Good, synonyms', '0.84')
('2, Good, synonyms', '0.89')     ('2, Static, synonyms', '0.77')   ('2, Static, short', '0.90')      ('4, Good, normal', '0.83')
('2, Static, normal', '0.89')     ('2, Good, normal', '0.76')       ('2, Good, synonyms', '0.89')     ('1, Static, short', '0.83')
('2, Static, synonyms', '0.89')   ('1, Kinetic, short', '0.76')     ('2, Static, normal', '0.88')     ('3, Good, normal', '0.82')
('2, Good, normal', '0.88')       ('2, Kinetic, short', '0.76')     ('2, Static, synonyms', '0.88')   ('1, Good, short', '0.82')
('4, Good, normal', '0.88')       ('1, Good, long', '0.75')         ('2, Good, normal', '0.88')       ('2, Good, long', '0.81')
('1, Static, long', '0.88')       ('2, Static, short', '0.75')      ('2, Kinetic, normal', '0.88')    ('2, Static, short', '0.81')
('1, Static, short', '0.88')      ('4, Good, normal', '0.75')       ('2, Good, long', '0.87')         ('2, Static, long', '0.80')
('1, Good, synonyms', '0.88')     ('2, Good, long', '0.74')         ('2, Kinetic, synonyms', '0.87')  ('2, Good, typos', '0.79')
('2, Static, long', '0.88')       ('2, Static, normal', '0.74')     ('2, Kinetic, short', '0.86')     ('1, Static, synonyms', '0.79')
('2, Good, typos', '0.88')        ('1, Static, long', '0.74')       ('1, Good, typos', '0.86')        ('1, Good, long', '0.79')
('2, Static, typos', '0.88')      ('1, Good, typos', '0.74')        ('1, Kinetic, short', '0.86')     ('1, Good, typos', '0.78')
('1, Kinetic, short', '0.87')     ('1, Good, synonyms', '0.74')     ('1, Good, long', '0.85')         ('1, Static, long', '0.77')
('2, Kinetic, normal', '0.87')    ('2, Static, long', '0.73')       ('1, Good, synonyms', '0.85')     ('1, Kinetic, short', '0.75')
('2, Kinetic, synonyms', '0.86')  ('2, Kinetic, normal', '0.72')    ('2, Static, long', '0.85')       ('2, Static, typos', '0.74')
('2, Kinetic, long', '0.85')      ('1, Kinetic, normal', '0.70')    ('1, Static, long', '0.84')       ('2, Kinetic, normal', '0.68')
('1, Kinetic, long', '0.85')      ('2, Kinetic, synonyms', '0.70')  ('1, Kinetic, normal', '0.82')    ('2, Kinetic, short', '0.68')
('2, Good, short', '0.85')        ('2, Good, typos', '0.70')        ('2, Kinetic, long', '0.81')      ('2, Kinetic, long', '0.63')
('2, Kinetic, short', '0.84')     ('2, Static, typos', '0.69')      ('1, Kinetic, long', '0.81')      ('2, Kinetic, typos', '0.63')
('2, Kinetic, typos', '0.84')     ('2, Kinetic, long', '0.69')      ('2, Good, typos', '0.80')        ('1, Kinetic, synonyms', '0.62')
('2, Static, short', '0.83')      ('1, Kinetic, synonyms', '0.68')  ('1, Kinetic, synonyms', '0.80')  ('2, Kinetic, synonyms', '0.62')
('1, Kinetic, typos', '0.83')     ('1, Kinetic, long', '0.67')      ('1, Kinetic, typos', '0.79')     ('1, Kinetic, long', '0.62')
('1, Kinetic, normal', '0.83')    ('1, Kinetic, typos', '0.63')     ('2, Static, typos', '0.75')      ('1, Kinetic, normal', '0.61')
('1, Kinetic, synonyms', '0.82')  ('1, irrelevant', '0.62')         ('2, Kinetic, typos', '0.67')     ('1, Kinetic, typos', '0.60')
('1, irrelevant', '0.75')         ('2, Kinetic, typos', '0.58')     ('1, irrelevant', '0.64')         ('1, irrelevant', '0.33')
('2, irrelevant', '0.70')         ('2, irrelevant', '0.50')         ('2, irrelevant', '0.60')         ('2, irrelevant', '0.20')
```

Figure .6: The results from comparison to answer labeled "1, Static, normal".



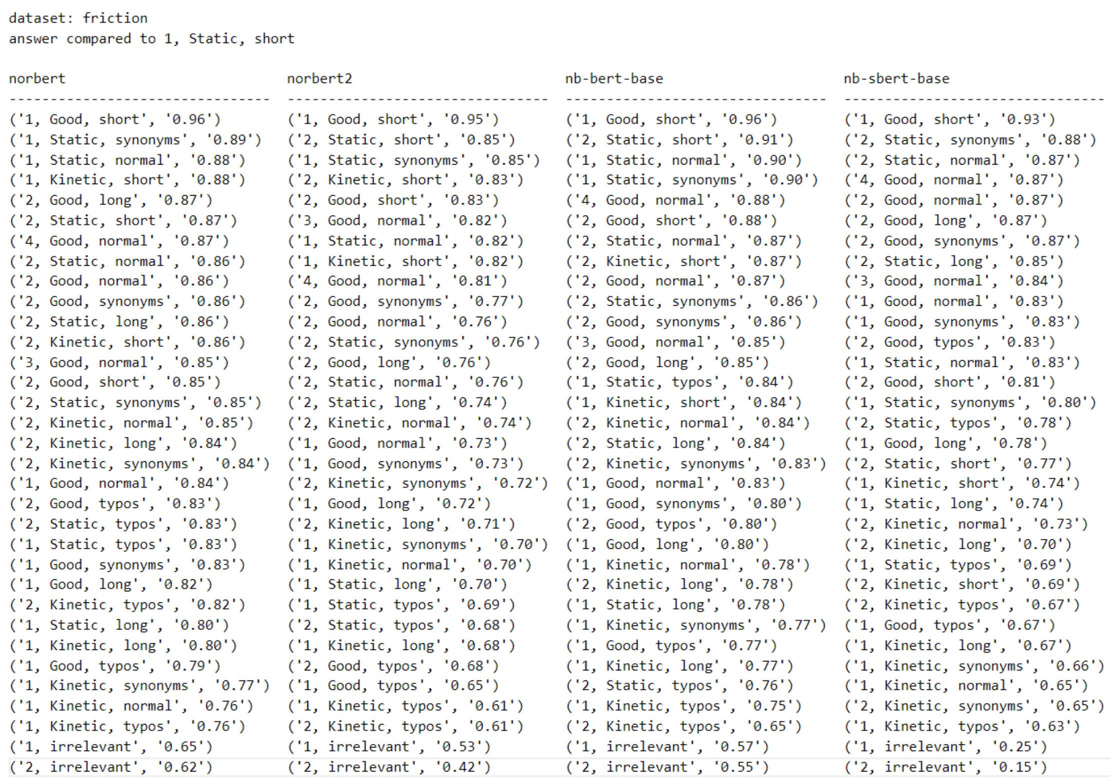
```
dataset: friction
answer compared to 1, Static, typos

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Static, normal', '0.97')     ('1, Static, normal', '0.89')     ('1, Static, normal', '0.95')     ('1, Static, normal', '0.89')
('1, Good, typos', '0.91')        ('1, Good, typos', '0.80')        ('1, Good, typos', '0.89')        ('1, Good, typos', '0.84')
('1, Good, normal', '0.88')       ('3, Good, normal', '0.75')       ('1, Static, synonyms', '0.89')   ('1, Good, normal', '0.76')
('1, Static, synonyms', '0.86')   ('1, Static, synonyms', '0.75')   ('1, Good, normal', '0.88')       ('2, Static, synonyms', '0.72')
('2, Good, typos', '0.86')        ('1, Good, normal', '0.73')       ('3, Good, normal', '0.87')       ('2, Static, normal', '0.72')
('1, Static, long', '0.86')       ('2, Good, synonyms', '0.69')     ('4, Good, normal', '0.87')       ('2, Good, normal', '0.71')
('1, Good, long', '0.86')         ('1, Static, long', '0.69')       ('2, Good, short', '0.87')        ('2, Good, synonyms', '0.71')
('2, Static, typos', '0.86')      ('1, Good, short', '0.69')        ('1, Good, short', '0.86')        ('1, Good, synonyms', '0.71')
('3, Good, normal', '0.85')       ('1, Static, short', '0.69')      ('2, Static, synonyms', '0.85')   ('1, Good, short', '0.70')
('2, Good, long', '0.85')         ('1, Good, long', '0.69')         ('2, Good, synonyms', '0.85')     ('4, Good, normal', '0.70')
('2, Good, synonyms', '0.85')     ('2, Static, synonyms', '0.68')   ('1, Static, short', '0.84')      ('1, Static, long', '0.69')
('2, Static, synonyms', '0.85')   ('4, Good, normal', '0.67')       ('2, Static, normal', '0.84')     ('2, Good, short', '0.69')
('2, Static, normal', '0.85')     ('1, Good, synonyms', '0.66')     ('1, Good, synonyms', '0.83')     ('2, Good, typos', '0.69')
('1, Good, short', '0.85')        ('1, Kinetic, typos', '0.66')     ('2, Good, normal', '0.83')       ('1, Static, short', '0.69')
('4, Good, normal', '0.85')       ('2, Good, normal', '0.66')       ('2, Static, short', '0.83')      ('1, Static, synonyms', '0.69')
('1, Good, synonyms', '0.84')     ('2, Good, short', '0.66')        ('2, Kinetic, normal', '0.82')    ('1, Good, long', '0.68')
('2, Good, normal', '0.84')       ('2, Good, long', '0.65')         ('1, Good, long', '0.82')         ('2, Static, short', '0.67')
('1, Kinetic, short', '0.84')     ('2, Static, normal', '0.65')     ('2, Good, long', '0.82')         ('3, Good, normal', '0.67')
('2, Static, long', '0.84')       ('2, Good, typos', '0.65')        ('2, Static, long', '0.81')       ('2, Static, long', '0.67')
('1, Kinetic, typos', '0.84')     ('2, Static, long', '0.65')       ('2, Kinetic, short', '0.81')     ('2, Good, long', '0.67')
('1, Static, short', '0.83')      ('2, Kinetic, short', '0.64')     ('2, Kinetic, synonyms', '0.81')  ('1, Kinetic, short', '0.67')
('1, Kinetic, long', '0.82')      ('1, Kinetic, short', '0.64')     ('1, Static, long', '0.80')       ('2, Static, typos', '0.66')
('2, Kinetic, synonyms', '0.82')  ('1, Kinetic, normal', '0.64')    ('1, Kinetic, short', '0.80')     ('2, Kinetic, short', '0.57')
('2, Kinetic, normal', '0.82')    ('2, Static, typos', '0.63')      ('1, Kinetic, typos', '0.79')     ('1, Kinetic, typos', '0.56')
('1, Kinetic, normal', '0.81')    ('2, Kinetic, synonyms', '0.63')  ('2, Good, typos', '0.78')        ('2, Kinetic, normal', '0.54')
('2, Kinetic, typos', '0.81')     ('2, Static, short', '0.63')      ('1, Kinetic, normal', '0.78')    ('1, Kinetic, synonyms', '0.54')
('1, Kinetic, synonyms', '0.80')  ('1, irrelevant', '0.62')         ('1, Kinetic, long', '0.78')      ('2, Kinetic, typos', '0.53')
('2, Kinetic, long', '0.80')      ('2, Kinetic, normal', '0.62')    ('1, Kinetic, synonyms', '0.78')  ('1, Kinetic, normal', '0.52')
('2, Kinetic, short', '0.80')     ('1, Kinetic, long', '0.61')      ('2, Kinetic, long', '0.76')      ('1, Kinetic, long', '0.51')
('2, Good, short', '0.79')        ('1, Kinetic, synonyms', '0.61')  ('2, Static, typos', '0.74')      ('2, Kinetic, synonyms', '0.50')
('2, Static, short', '0.77')      ('2, Kinetic, long', '0.60')      ('2, Kinetic, typos', '0.68')     ('2, Kinetic, long', '0.50')
('1, irrelevant', '0.74')         ('2, Kinetic, typos', '0.57')     ('1, irrelevant', '0.63')         ('1, irrelevant', '0.29')
('2, irrelevant', '0.68')         ('2, irrelevant', '0.54')         ('2, irrelevant', '0.62')         ('2, irrelevant', '0.22')
```

Figure .7: The results from comparison to answer labeled "1, Static, typos".



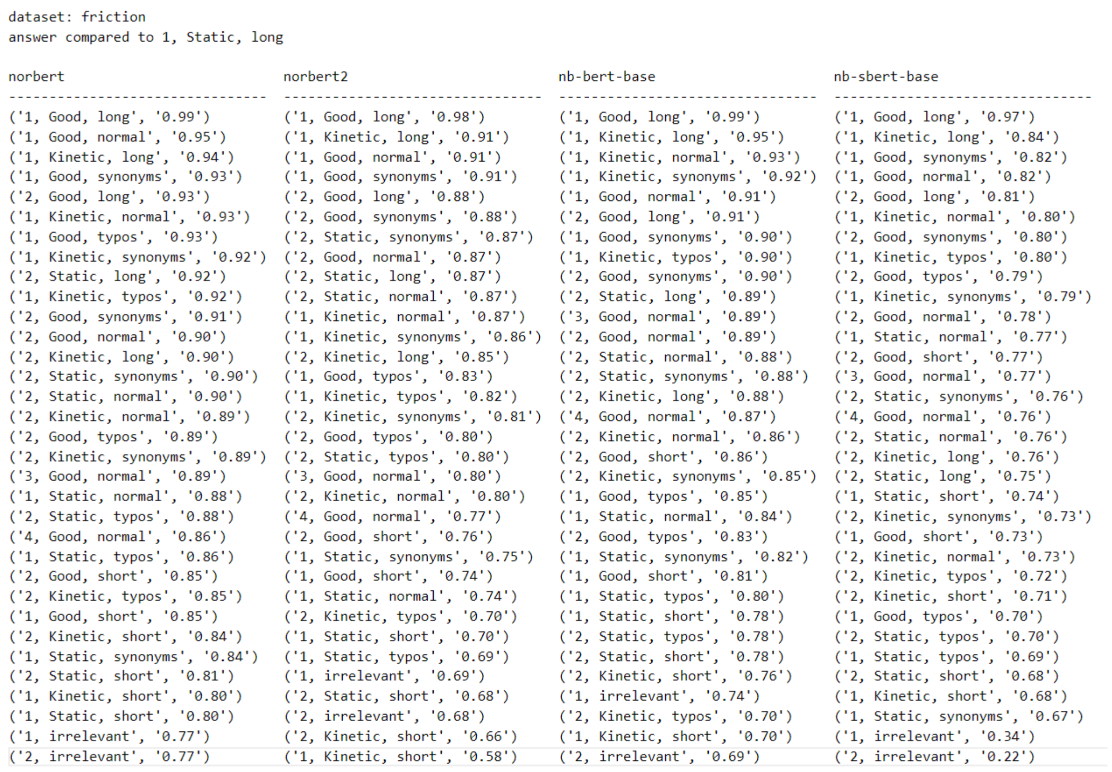
```
dataset: friction
answer compared to 1, Static, synonyms

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
--------------------------------   --------------------------------   --------------------------------   --------------------------------
('1, Good, short', '0.92')         ('1, Static, normal', '0.88')      ('1, Static, normal', '0.93')      ('1, Good, short', '0.85')
('1, Static, normal', '0.91')      ('1, Good, short', '0.88')         ('1, Good, short', '0.91')         ('1, Good, synonyms', '0.85')
('3, Good, normal', '0.90')        ('1, Static, short', '0.85')       ('2, Good, short', '0.90')         ('2, Static, synonyms', '0.84')
('2, Static, normal', '0.90')      ('3, Good, normal', '0.84')        ('1, Static, short', '0.90')       ('2, Static, normal', '0.83')
('2, Good, normal', '0.90')        ('2, Good, short', '0.83')         ('3, Good, normal', '0.89')        ('2, Good, normal', '0.82')
('2, Good, long', '0.90')          ('2, Good, synonyms', '0.80')      ('2, Static, short', '0.89')       ('2, Good, synonyms', '0.82')
('1, Static, short', '0.89')       ('2, Good, normal', '0.80')        ('1, Static, typos', '0.89')       ('2, Static, short', '0.81')
('1, Good, synonyms', '0.89')      ('1, Kinetic, short', '0.80')      ('4, Good, normal', '0.89')        ('1, Static, short', '0.80')
('2, Good, short', '0.89')         ('2, Kinetic, short', '0.80')      ('2, Good, synonyms', '0.89')      ('2, Good, short', '0.79')
('2, Good, synonyms', '0.89')      ('1, Good, synonyms', '0.79')      ('2, Static, normal', '0.88')      ('1, Static, normal', '0.79')
('2, Kinetic, normal', '0.88')     ('2, Static, synonyms', '0.79')    ('1, Good, synonyms', '0.88')      ('2, Static, long', '0.79')
('2, Static, synonyms', '0.88')    ('4, Good, normal', '0.79')        ('2, Static, synonyms', '0.88')    ('3, Good, normal', '0.78')
('2, Static, short', '0.88')       ('2, Static, short', '0.79')       ('2, Good, normal', '0.88')        ('2, Good, long', '0.76')
('2, Static, long', '0.88')        ('2, Kinetic, normal', '0.78')     ('1, Good, normal', '0.87')        ('4, Good, normal', '0.75')
('4, Good, normal', '0.88')        ('2, Static, normal', '0.78')      ('2, Kinetic, normal', '0.86')     ('1, Kinetic, short', '0.74')
('1, Good, normal', '0.88')        ('2, Good, long', '0.78')          ('2, Kinetic, short', '0.86')      ('1, Good, normal', '0.74')
('1, Good, long', '0.87')          ('1, Good, normal', '0.78')        ('2, Kinetic, synonyms', '0.86')   ('2, Good, typos', '0.73')
('2, Kinetic, synonyms', '0.87')   ('1, Good, long', '0.77')          ('2, Good, long', '0.86')          ('2, Static, typos', '0.69')
('1, Kinetic, short', '0.86')      ('2, Static, long', '0.76')        ('1, Kinetic, short', '0.85')      ('1, Good, long', '0.69')
('1, Static, typos', '0.86')       ('1, Static, long', '0.75')        ('2, Static, long', '0.85')        ('1, Static, typos', '0.69')
('2, Kinetic, long', '0.86')       ('1, Static, typos', '0.75')       ('1, Good, long', '0.84')          ('1, Static, long', '0.67')
('2, Good, typos', '0.85')         ('2, Kinetic, synonyms', '0.74')   ('1, Good, typos', '0.83')         ('1, Kinetic, synonyms', '0.65')
('2, Kinetic, short', '0.85')      ('1, Kinetic, normal', '0.73')     ('1, Static, long', '0.82')        ('1, Good, typos', '0.61')
('2, Static, typos', '0.85')       ('2, Kinetic, long', '0.73')       ('1, Kinetic, normal', '0.81')     ('2, Kinetic, normal', '0.60')
('1, Static, long', '0.84')        ('1, Kinetic, synonyms', '0.72')   ('2, Good, typos', '0.81')         ('1, Kinetic, normal', '0.57')
('1, Good, typos', '0.84')         ('2, Good, typos', '0.71')         ('1, Kinetic, synonyms', '0.81')   ('2, Kinetic, short', '0.56')
('1, Kinetic, long', '0.83')       ('1, Kinetic, long', '0.71')       ('1, Kinetic, long', '0.81')       ('1, Kinetic, long', '0.56')
('2, Kinetic, typos', '0.82')      ('2, Static, typos', '0.71')       ('2, Kinetic, long', '0.80')       ('2, Kinetic, long', '0.54')
('1, Kinetic, synonyms', '0.80')   ('1, Good, typos', '0.70')         ('1, Kinetic, typos', '0.78')      ('1, Kinetic, typos', '0.54')
('1, Kinetic, normal', '0.79')     ('1, Kinetic, typos', '0.64')      ('2, Static, typos', '0.77')       ('2, Kinetic, typos', '0.53')
('1, Kinetic, typos', '0.79')      ('2, Kinetic, typos', '0.61')      ('2, Kinetic, typos', '0.68')      ('2, Kinetic, synonyms', '0.52')
('1, irrelevant', '0.76')          ('1, irrelevant', '0.59')          ('1, irrelevant', '0.66')          ('1, irrelevant', '0.27')
('2, irrelevant', '0.71')          ('2, irrelevant', '0.47')          ('2, irrelevant', '0.59')          ('2, irrelevant', '0.14')
```

Figure .8: The results from comparison to answer labeled "1, Static, synonyms".



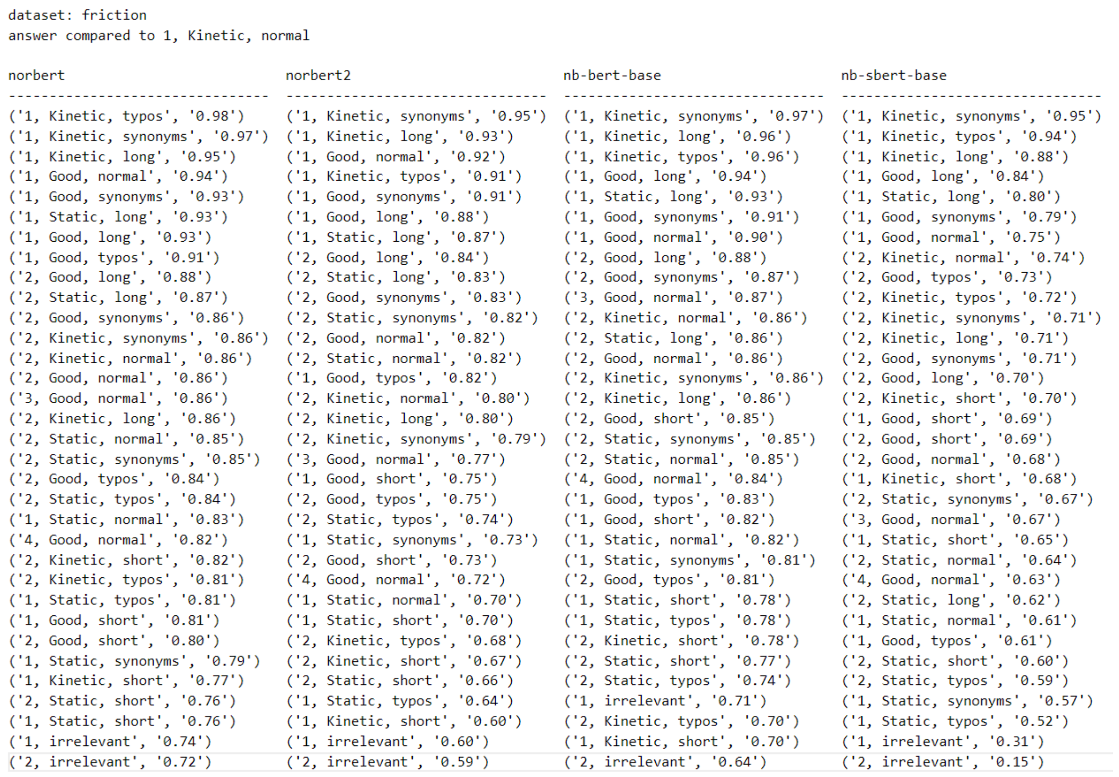
```
dataset: friction
answer compared to 1, Static, short

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
--------------------------------   --------------------------------   --------------------------------   --------------------------------
('1, Good, short', '0.96')         ('1, Good, short', '0.95')         ('1, Good, short', '0.96')         ('1, Good, short', '0.93')
('1, Static, synonyms', '0.89')    ('2, Static, short', '0.85')       ('2, Static, short', '0.91')       ('2, Static, synonyms', '0.88')
('1, Static, normal', '0.88')      ('1, Static, synonyms', '0.85')    ('1, Static, normal', '0.90')      ('2, Static, normal', '0.87')
('1, Kinetic, short', '0.88')      ('2, Kinetic, short', '0.83')      ('1, Static, synonyms', '0.90')    ('4, Good, normal', '0.87')
('2, Good, long', '0.87')          ('2, Good, short', '0.83')         ('4, Good, normal', '0.88')        ('2, Good, normal', '0.87')
('2, Static, short', '0.87')       ('3, Good, normal', '0.82')        ('2, Good, short', '0.88')         ('2, Good, long', '0.87')
('4, Good, normal', '0.87')        ('1, Static, normal', '0.82')      ('2, Static, normal', '0.87')      ('2, Good, synonyms', '0.87')
('2, Static, normal', '0.86')      ('1, Kinetic, short', '0.82')      ('2, Kinetic, short', '0.87')      ('2, Static, long', '0.85')
('2, Good, normal', '0.86')        ('4, Good, normal', '0.81')        ('2, Good, normal', '0.87')        ('3, Good, normal', '0.84')
('2, Good, synonyms', '0.86')      ('2, Good, synonyms', '0.77')      ('2, Static, synonyms', '0.86')    ('1, Good, normal', '0.83')
('2, Static, long', '0.86')        ('2, Good, normal', '0.76')        ('2, Good, synonyms', '0.86')      ('1, Good, synonyms', '0.83')
('2, Kinetic, short', '0.86')      ('2, Static, synonyms', '0.76')    ('3, Good, normal', '0.85')        ('2, Good, typos', '0.83')
('3, Good, normal', '0.85')        ('2, Good, long', '0.76')          ('2, Good, long', '0.85')          ('1, Static, normal', '0.83')
('2, Good, short', '0.85')         ('2, Static, normal', '0.76')      ('1, Static, typos', '0.84')       ('2, Good, short', '0.81')
('2, Static, synonyms', '0.85')    ('2, Static, long', '0.74')        ('1, Kinetic, short', '0.84')      ('1, Static, synonyms', '0.80')
('2, Kinetic, normal', '0.85')     ('2, Kinetic, normal', '0.74')     ('2, Kinetic, normal', '0.84')     ('2, Static, typos', '0.78')
('2, Kinetic, long', '0.84')       ('1, Good, normal', '0.73')        ('2, Static, long', '0.84')        ('1, Good, long', '0.78')
('2, Kinetic, synonyms', '0.84')   ('1, Good, synonyms', '0.73')      ('2, Kinetic, synonyms', '0.83')   ('2, Static, short', '0.77')
('1, Good, normal', '0.84')        ('2, Kinetic, synonyms', '0.72')   ('1, Good, normal', '0.83')        ('1, Kinetic, short', '0.74')
('2, Good, typos', '0.83')         ('1, Good, long', '0.72')          ('1, Good, synonyms', '0.80')      ('1, Static, long', '0.74')
('2, Static, typos', '0.83')       ('2, Kinetic, long', '0.71')       ('2, Good, typos', '0.80')         ('2, Kinetic, normal', '0.73')
('1, Static, typos', '0.83')       ('1, Kinetic, synonyms', '0.70')   ('1, Good, long', '0.80')          ('2, Kinetic, long', '0.70')
('1, Good, synonyms', '0.83')      ('1, Kinetic, normal', '0.70')     ('1, Kinetic, normal', '0.78')     ('1, Static, typos', '0.69')
('1, Good, long', '0.82')          ('1, Static, long', '0.70')        ('2, Kinetic, long', '0.78')       ('2, Kinetic, short', '0.69')
('2, Kinetic, typos', '0.82')      ('1, Static, typos', '0.69')       ('1, Static, long', '0.78')        ('2, Kinetic, typos', '0.67')
('1, Static, long', '0.80')        ('2, Static, typos', '0.68')       ('1, Kinetic, synonyms', '0.77')   ('1, Good, typos', '0.67')
('1, Kinetic, long', '0.80')       ('1, Kinetic, long', '0.68')       ('1, Good, typos', '0.77')         ('1, Kinetic, long', '0.67')
('1, Good, typos', '0.79')         ('2, Good, typos', '0.68')         ('1, Kinetic, long', '0.77')       ('1, Kinetic, synonyms', '0.66')
('1, Kinetic, synonyms', '0.77')   ('1, Good, typos', '0.65')         ('2, Static, typos', '0.76')       ('1, Kinetic, normal', '0.65')
('1, Kinetic, normal', '0.76')     ('1, Kinetic, typos', '0.61')      ('1, Kinetic, typos', '0.75')      ('2, Kinetic, synonyms', '0.65')
('1, Kinetic, typos', '0.76')      ('2, Kinetic, typos', '0.61')      ('2, Kinetic, typos', '0.65')      ('1, Kinetic, typos', '0.63')
('1, irrelevant', '0.65')          ('1, irrelevant', '0.53')          ('1, irrelevant', '0.57')          ('1, irrelevant', '0.25')
('2, irrelevant', '0.62')          ('2, irrelevant', '0.42')          ('2, irrelevant', '0.55')          ('2, irrelevant', '0.15')
```

Figure .9: The results from comparison to answer labeled "1, Static, short".



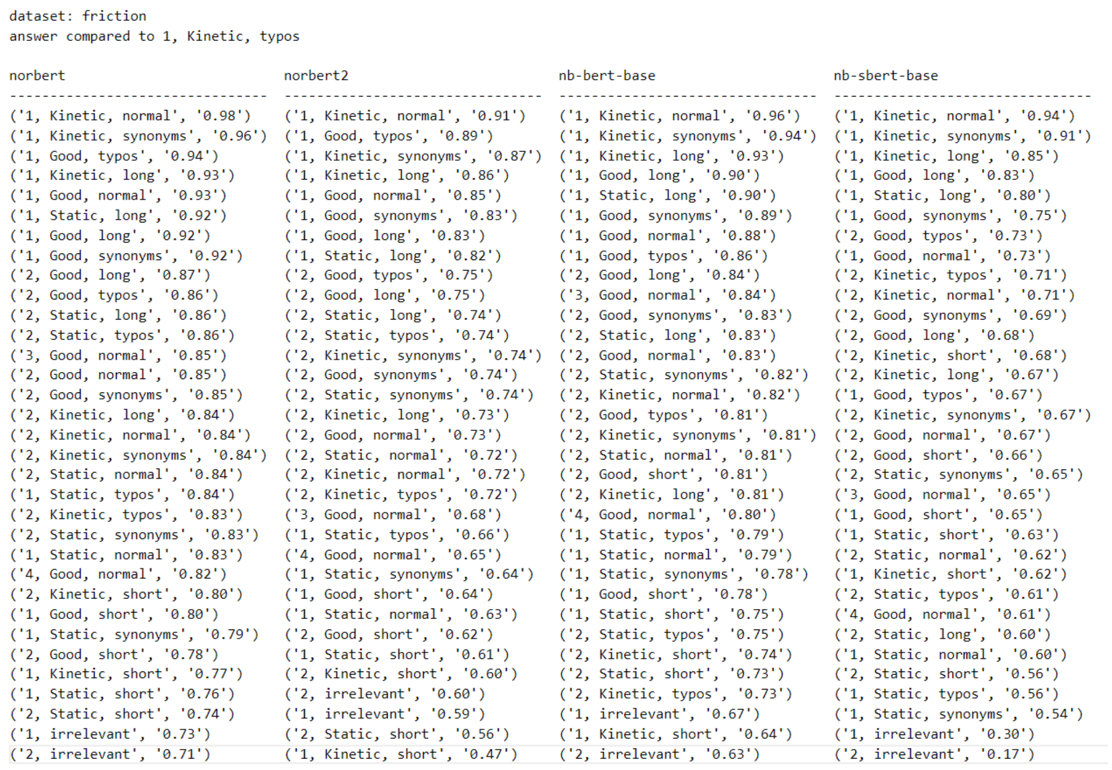
```
dataset: friction
answer compared to 1, Static, long

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Good, long', '0.99')         ('1, Good, long', '0.98')         ('1, Good, long', '0.99')         ('1, Good, long', '0.97')
('1, Good, normal', '0.95')       ('1, Kinetic, long', '0.91')      ('1, Kinetic, long', '0.95')      ('1, Kinetic, long', '0.84')
('1, Kinetic, long', '0.94')      ('1, Good, normal', '0.91')       ('1, Kinetic, normal', '0.93')    ('1, Good, synonyms', '0.82')
('1, Good, synonyms', '0.93')     ('1, Good, synonyms', '0.91')     ('1, Kinetic, synonyms', '0.92')  ('1, Good, normal', '0.82')
('2, Good, long', '0.93')         ('2, Good, long', '0.88')         ('1, Good, normal', '0.91')       ('2, Good, long', '0.81')
('1, Kinetic, normal', '0.93')    ('2, Good, synonyms', '0.88')     ('2, Good, long', '0.91')         ('1, Kinetic, normal', '0.80')
('1, Good, typos', '0.93')        ('2, Static, synonyms', '0.87')   ('1, Good, synonyms', '0.90')     ('2, Good, synonyms', '0.80')
('1, Kinetic, synonyms', '0.92')  ('2, Good, normal', '0.87')       ('1, Kinetic, typos', '0.90')     ('1, Kinetic, typos', '0.80')
('2, Static, long', '0.92')       ('2, Static, long', '0.87')       ('2, Good, synonyms', '0.90')     ('2, Good, typos', '0.79')
('1, Kinetic, typos', '0.92')     ('2, Static, normal', '0.87')     ('2, Static, long', '0.89')       ('1, Kinetic, synonyms', '0.79')
('2, Good, synonyms', '0.91')     ('1, Kinetic, normal', '0.87')    ('3, Good, normal', '0.89')       ('2, Good, normal', '0.78')
('2, Good, normal', '0.90')       ('1, Kinetic, synonyms', '0.86')  ('2, Good, normal', '0.89')       ('1, Static, normal', '0.77')
('2, Kinetic, long', '0.90')      ('2, Kinetic, long', '0.85')      ('2, Static, normal', '0.88')     ('2, Good, short', '0.77')
('2, Static, synonyms', '0.90')   ('1, Good, typos', '0.83')        ('2, Static, synonyms', '0.88')   ('3, Good, normal', '0.77')
('2, Static, normal', '0.90')     ('1, Kinetic, typos', '0.82')     ('2, Kinetic, long', '0.88')      ('2, Static, synonyms', '0.76')
('2, Kinetic, normal', '0.89')    ('2, Kinetic, synonyms', '0.81')  ('4, Good, normal', '0.87')       ('4, Good, normal', '0.76')
('2, Good, typos', '0.89')        ('2, Good, typos', '0.80')        ('2, Kinetic, normal', '0.86')    ('2, Static, normal', '0.76')
('2, Kinetic, synonyms', '0.89')  ('2, Static, typos', '0.80')      ('2, Good, short', '0.86')        ('2, Kinetic, long', '0.76')
('3, Good, normal', '0.89')       ('3, Good, normal', '0.80')       ('2, Kinetic, synonyms', '0.85')  ('2, Static, long', '0.75')
('1, Static, normal', '0.88')     ('2, Kinetic, normal', '0.80')    ('1, Good, typos', '0.85')        ('1, Static, short', '0.74')
('2, Static, typos', '0.88')      ('4, Good, normal', '0.77')       ('1, Static, normal', '0.84')     ('2, Kinetic, synonyms', '0.73')
('4, Good, normal', '0.86')       ('2, Good, short', '0.76')        ('2, Good, typos', '0.83')        ('1, Good, short', '0.73')
('1, Static, typos', '0.86')      ('1, Static, synonyms', '0.75')   ('1, Static, synonyms', '0.82')   ('2, Kinetic, normal', '0.73')
('2, Good, short', '0.85')        ('1, Good, short', '0.74')        ('1, Good, short', '0.81')        ('2, Kinetic, typos', '0.72')
('2, Kinetic, typos', '0.85')     ('1, Static, normal', '0.74')     ('1, Static, typos', '0.80')      ('2, Kinetic, short', '0.71')
('1, Good, short', '0.85')        ('2, Kinetic, typos', '0.70')     ('1, Static, short', '0.78')      ('1, Good, typos', '0.70')
('2, Kinetic, short', '0.84')     ('1, Static, short', '0.70')      ('2, Static, typos', '0.78')      ('2, Static, typos', '0.70')
('1, Static, synonyms', '0.84')   ('1, Static, typos', '0.69')      ('2, Static, short', '0.78')      ('1, Static, typos', '0.69')
('2, Static, short', '0.81')      ('1, irrelevant', '0.69')         ('2, Kinetic, short', '0.76')     ('2, Static, short', '0.68')
('1, Kinetic, short', '0.80')     ('2, Static, short', '0.68')      ('1, irrelevant', '0.74')         ('1, Kinetic, short', '0.68')
('1, Static, short', '0.80')      ('2, irrelevant', '0.68')         ('2, Kinetic, typos', '0.70')     ('1, Static, synonyms', '0.67')
('1, irrelevant', '0.77')         ('2, Kinetic, short', '0.66')     ('1, Kinetic, short', '0.70')     ('1, irrelevant', '0.34')
('2, irrelevant', '0.77')         ('1, Kinetic, short', '0.58')     ('2, irrelevant', '0.69')         ('2, irrelevant', '0.22')
```

Figure .10: The results from comparison to answer labeled "1, Static, long".



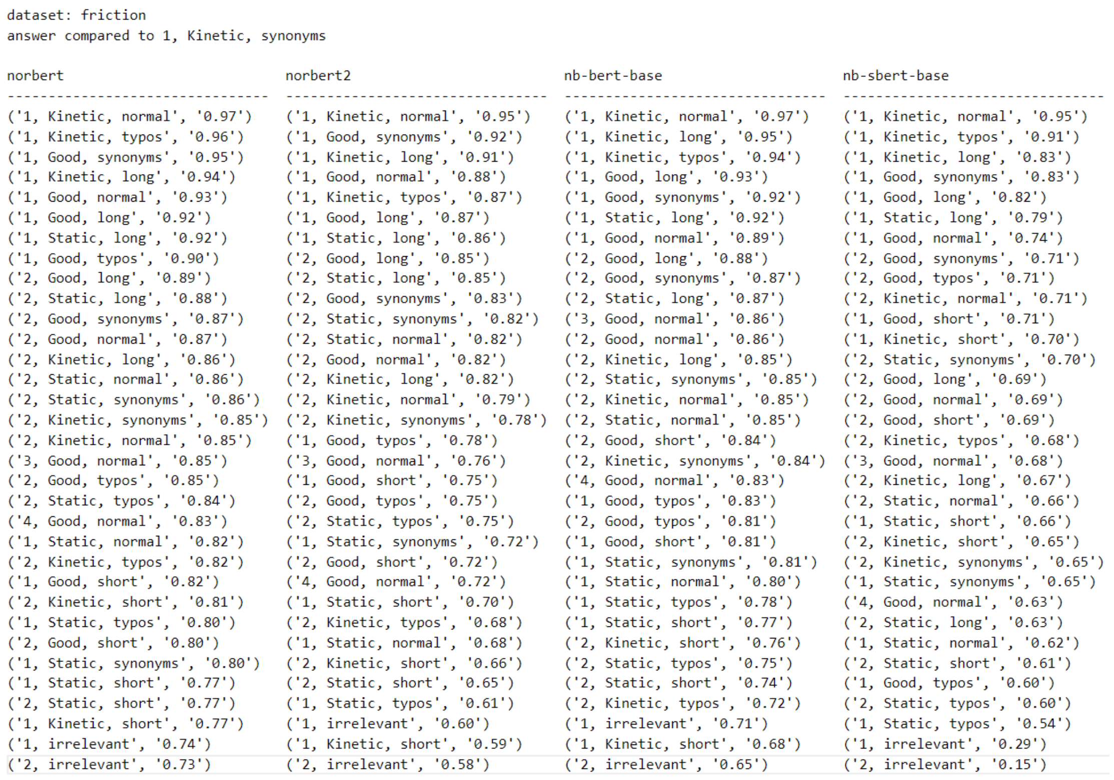
```
dataset: friction
answer compared to 1, Kinetic, normal

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Kinetic, typos', '0.98')     ('1, Kinetic, synonyms', '0.95')  ('1, Kinetic, synonyms', '0.97')  ('1, Kinetic, synonyms', '0.95')
('1, Kinetic, synonyms', '0.97')  ('1, Kinetic, long', '0.93')      ('1, Kinetic, long', '0.96')      ('1, Kinetic, typos', '0.94')
('1, Kinetic, long', '0.95')      ('1, Good, normal', '0.92')       ('1, Kinetic, typos', '0.96')     ('1, Kinetic, long', '0.88')
('1, Good, normal', '0.94')       ('1, Kinetic, typos', '0.91')     ('1, Good, long', '0.94')         ('1, Good, long', '0.84')
('1, Good, synonyms', '0.93')     ('1, Good, synonyms', '0.91')     ('1, Static, long', '0.93')       ('1, Static, long', '0.80')
('1, Static, long', '0.93')       ('1, Good, long', '0.88')         ('1, Good, synonyms', '0.91')     ('1, Good, synonyms', '0.79')
('1, Good, long', '0.93')         ('1, Static, long', '0.87')       ('1, Good, normal', '0.90')       ('1, Good, normal', '0.75')
('1, Good, typos', '0.91')        ('2, Good, long', '0.84')         ('2, Good, long', '0.88')         ('2, Kinetic, normal', '0.74')
('2, Good, long', '0.88')         ('2, Static, long', '0.83')       ('2, Good, synonyms', '0.87')     ('2, Good, typos', '0.73')
('2, Static, long', '0.87')       ('2, Good, synonyms', '0.83')     ('3, Good, normal', '0.87')       ('2, Kinetic, typos', '0.72')
('2, Good, synonyms', '0.86')     ('2, Static, synonyms', '0.82')   ('2, Kinetic, normal', '0.86')    ('2, Kinetic, synonyms', '0.71')
('2, Kinetic, synonyms', '0.86')  ('2, Good, normal', '0.82')       ('2, Static, long', '0.86')       ('2, Kinetic, long', '0.71')
('2, Kinetic, normal', '0.86')    ('2, Static, normal', '0.82')     ('2, Good, normal', '0.86')       ('2, Good, synonyms', '0.71')
('2, Good, normal', '0.86')       ('1, Good, typos', '0.82')        ('2, Kinetic, synonyms', '0.86')  ('2, Good, long', '0.70')
('3, Good, normal', '0.86')       ('2, Kinetic, normal', '0.80')    ('2, Kinetic, long', '0.86')      ('2, Kinetic, short', '0.70')
('2, Kinetic, long', '0.86')      ('2, Kinetic, long', '0.80')      ('2, Good, short', '0.85')        ('1, Good, short', '0.69')
('2, Static, normal', '0.85')     ('2, Kinetic, synonyms', '0.79')  ('2, Static, synonyms', '0.85')   ('2, Good, short', '0.69')
('2, Static, synonyms', '0.85')   ('3, Good, normal', '0.77')       ('2, Static, normal', '0.85')     ('2, Good, normal', '0.68')
('2, Good, typos', '0.84')        ('1, Good, short', '0.75')        ('4, Good, normal', '0.84')       ('1, Kinetic, short', '0.68')
('2, Static, typos', '0.84')      ('2, Good, typos', '0.75')        ('1, Good, typos', '0.83')        ('2, Static, synonyms', '0.67')
('1, Static, normal', '0.83')     ('2, Static, typos', '0.74')      ('1, Good, short', '0.82')        ('3, Good, normal', '0.67')
('4, Good, normal', '0.82')       ('1, Static, synonyms', '0.73')   ('1, Static, normal', '0.82')     ('1, Static, short', '0.65')
('2, Kinetic, short', '0.82')     ('2, Good, short', '0.73')        ('1, Static, synonyms', '0.81')   ('2, Static, normal', '0.64')
('2, Kinetic, typos', '0.81')     ('4, Good, normal', '0.72')       ('2, Good, typos', '0.81')        ('4, Good, normal', '0.63')
('1, Static, typos', '0.81')      ('1, Static, normal', '0.70')     ('1, Static, short', '0.78')      ('2, Static, long', '0.62')
('1, Good, short', '0.81')        ('1, Static, short', '0.70')      ('1, Static, typos', '0.78')      ('1, Static, normal', '0.61')
('2, Good, short', '0.80')        ('2, Kinetic, typos', '0.68')     ('2, Kinetic, short', '0.78')     ('1, Good, typos', '0.61')
('1, Static, synonyms', '0.79')   ('2, Kinetic, short', '0.67')     ('2, Static, short', '0.77')      ('2, Static, short', '0.60')
('1, Kinetic, short', '0.77')     ('2, Static, short', '0.66')      ('2, Static, typos', '0.74')      ('2, Static, typos', '0.59')
('2, Static, short', '0.76')      ('1, Static, typos', '0.64')      ('1, irrelevant', '0.71')         ('1, Static, synonyms', '0.57')
('1, Static, short', '0.76')      ('1, Kinetic, short', '0.60')     ('2, Kinetic, typos', '0.70')     ('1, Static, typos', '0.52')
('1, irrelevant', '0.74')         ('1, irrelevant', '0.60')         ('1, Kinetic, short', '0.70')     ('1, irrelevant', '0.31')
('2, irrelevant', '0.72')         ('2, irrelevant', '0.59')         ('2, irrelevant', '0.64')         ('2, irrelevant', '0.15')
```

Figure .11: The results from comparison to answer labeled "1, Kinetic, normal".



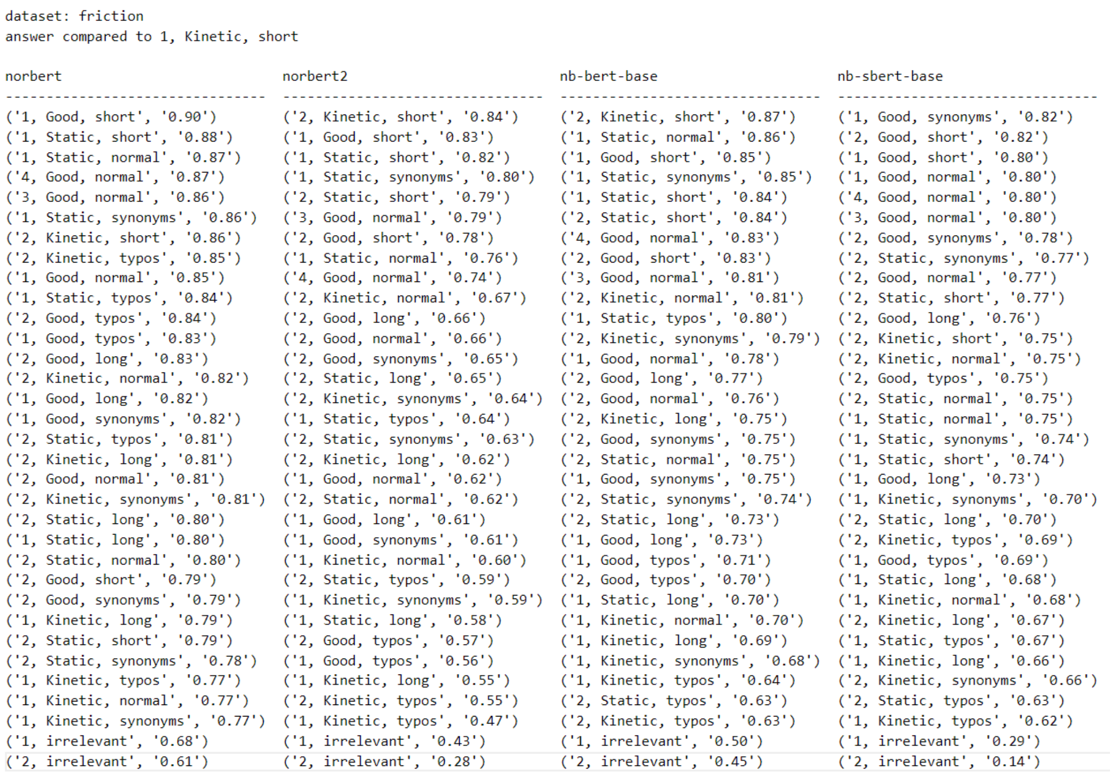
```
dataset: friction
answer compared to 1, Kinetic, typos

norbert                             norbert2                            nb-bert-base                        nb-sbert-base
--------------------------------    --------------------------------    --------------------------------    --------------------------------
('1, Kinetic, normal', '0.98')      ('1, Kinetic, normal', '0.91')      ('1, Kinetic, normal', '0.96')      ('1, Kinetic, normal', '0.94')
('1, Kinetic, synonyms', '0.96')    ('1, Good, typos', '0.89')          ('1, Kinetic, synonyms', '0.94')    ('1, Kinetic, synonyms', '0.91')
('1, Good, typos', '0.94')          ('1, Kinetic, synonyms', '0.87')    ('1, Kinetic, long', '0.93')        ('1, Kinetic, long', '0.85')
('1, Kinetic, long', '0.93')        ('1, Kinetic, long', '0.86')        ('1, Good, long', '0.90')           ('1, Good, long', '0.83')
('1, Good, normal', '0.93')         ('1, Good, normal', '0.85')         ('1, Static, long', '0.90')         ('1, Static, long', '0.80')
('1, Static, long', '0.92')         ('1, Good, synonyms', '0.83')       ('1, Good, synonyms', '0.89')       ('1, Good, synonyms', '0.75')
('1, Good, long', '0.92')           ('1, Good, long', '0.83')           ('1, Good, normal', '0.88')         ('2, Good, typos', '0.73')
('1, Good, synonyms', '0.92')       ('1, Static, long', '0.82')         ('1, Good, typos', '0.86')          ('1, Good, normal', '0.73')
('2, Good, long', '0.87')           ('2, Good, typos', '0.75')          ('2, Good, long', '0.84')           ('2, Kinetic, typos', '0.71')
('2, Good, typos', '0.86')          ('2, Good, long', '0.75')           ('3, Good, normal', '0.84')         ('2, Kinetic, normal', '0.71')
('2, Static, long', '0.86')         ('2, Static, long', '0.74')         ('2, Good, synonyms', '0.83')       ('2, Good, synonyms', '0.69')
('2, Static, typos', '0.86')        ('2, Static, typos', '0.74')        ('2, Static, long', '0.83')         ('2, Good, long', '0.68')
('3, Good, normal', '0.85')         ('2, Kinetic, synonyms', '0.74')    ('2, Good, normal', '0.83')         ('2, Kinetic, short', '0.68')
('2, Good, normal', '0.85')         ('2, Good, synonyms', '0.74')       ('2, Static, synonyms', '0.82')     ('2, Kinetic, long', '0.67')
('2, Good, synonyms', '0.85')       ('2, Static, synonyms', '0.74')     ('2, Kinetic, normal', '0.82')      ('1, Good, typos', '0.67')
('2, Kinetic, long', '0.84')        ('2, Kinetic, long', '0.73')        ('2, Good, typos', '0.81')          ('2, Kinetic, synonyms', '0.67')
('2, Kinetic, normal', '0.84')      ('2, Good, normal', '0.73')         ('2, Kinetic, synonyms', '0.81')    ('2, Good, normal', '0.67')
('2, Kinetic, synonyms', '0.84')    ('2, Static, normal', '0.72')       ('2, Static, normal', '0.81')       ('2, Good, short', '0.66')
('2, Static, normal', '0.84')       ('2, Kinetic, normal', '0.72')      ('2, Good, short', '0.81')          ('2, Static, synonyms', '0.65')
('1, Static, typos', '0.84')        ('2, Kinetic, typos', '0.72')       ('2, Kinetic, long', '0.81')        ('3, Good, normal', '0.65')
('2, Kinetic, typos', '0.83')       ('3, Good, normal', '0.68')         ('4, Good, normal', '0.80')         ('1, Good, short', '0.65')
('2, Static, synonyms', '0.83')     ('1, Static, typos', '0.66')        ('1, Static, typos', '0.79')        ('1, Static, short', '0.63')
('1, Static, normal', '0.83')       ('4, Good, normal', '0.65')         ('1, Static, normal', '0.79')       ('2, Static, normal', '0.62')
('4, Good, normal', '0.82')         ('1, Static, synonyms', '0.64')     ('1, Static, synonyms', '0.78')     ('1, Kinetic, short', '0.62')
('2, Kinetic, short', '0.80')       ('1, Good, short', '0.64')          ('1, Good, short', '0.78')          ('2, Static, typos', '0.61')
('1, Good, short', '0.80')          ('1, Static, normal', '0.63')       ('1, Static, short', '0.75')        ('4, Good, normal', '0.61')
('1, Static, synonyms', '0.79')     ('2, Good, short', '0.62')          ('2, Static, typos', '0.75')        ('2, Static, long', '0.60')
('2, Good, short', '0.78')          ('1, Static, short', '0.61')        ('2, Kinetic, short', '0.74')       ('1, Static, normal', '0.60')
('1, Kinetic, short', '0.77')       ('2, Kinetic, short', '0.60')       ('2, Static, short', '0.73')        ('2, Static, short', '0.56')
('1, Static, short', '0.76')        ('2, irrelevant', '0.60')           ('2, Kinetic, typos', '0.73')       ('1, Static, typos', '0.56')
('2, Static, short', '0.74')        ('1, irrelevant', '0.59')           ('1, irrelevant', '0.67')           ('1, Static, synonyms', '0.54')
('1, irrelevant', '0.73')           ('2, Static, short', '0.56')        ('1, Kinetic, short', '0.64')       ('1, irrelevant', '0.30')
('2, irrelevant', '0.71')           ('1, Kinetic, short', '0.47')       ('2, irrelevant', '0.63')           ('2, irrelevant', '0.17')
```

Figure .12: The results from comparison to answer labeled "1, Kinetic, typos".



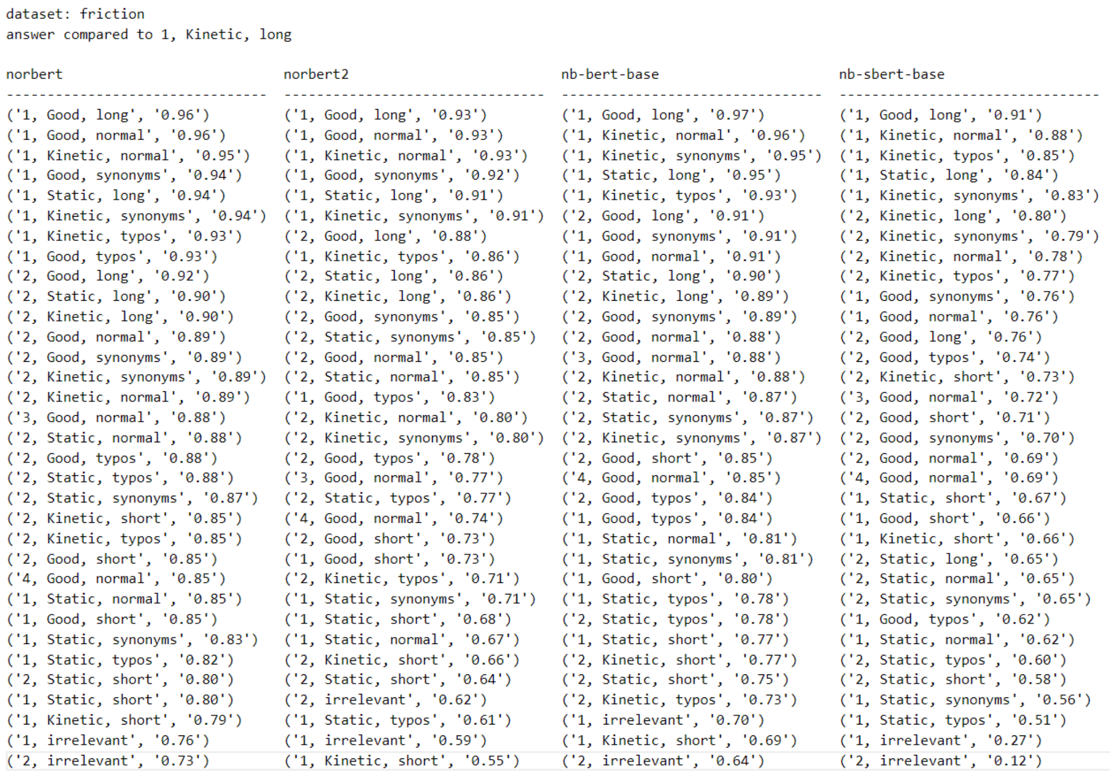
```
dataset: friction
answer compared to 1, Kinetic, synonyms

norbert                             norbert2                            nb-bert-base                        nb-sbert-base
--------------------------------    --------------------------------    --------------------------------    --------------------------------
('1, Kinetic, normal', '0.97')      ('1, Kinetic, normal', '0.95')      ('1, Kinetic, normal', '0.97')      ('1, Kinetic, normal', '0.95')
('1, Kinetic, typos', '0.96')       ('1, Good, synonyms', '0.92')       ('1, Kinetic, long', '0.95')        ('1, Kinetic, typos', '0.91')
('1, Good, synonyms', '0.95')       ('1, Kinetic, long', '0.91')        ('1, Kinetic, typos', '0.94')       ('1, Kinetic, long', '0.83')
('1, Kinetic, long', '0.94')        ('1, Good, normal', '0.88')         ('1, Good, long', '0.93')           ('1, Good, synonyms', '0.83')
('1, Good, normal', '0.93')         ('1, Kinetic, typos', '0.87')       ('1, Good, synonyms', '0.92')       ('1, Good, long', '0.82')
('1, Good, long', '0.92')           ('1, Good, long', '0.87')           ('1, Static, long', '0.92')         ('1, Static, long', '0.79')
('1, Static, long', '0.92')         ('1, Static, long', '0.86')         ('1, Good, normal', '0.89')         ('1, Good, normal', '0.74')
('1, Good, typos', '0.90')          ('2, Good, long', '0.85')           ('2, Good, long', '0.88')           ('2, Good, synonyms', '0.71')
('2, Good, long', '0.89')           ('2, Static, long', '0.85')         ('2, Good, synonyms', '0.87')       ('2, Good, typos', '0.71')
('2, Static, long', '0.88')         ('2, Good, synonyms', '0.83')       ('2, Static, long', '0.87')         ('2, Kinetic, normal', '0.71')
('2, Good, synonyms', '0.87')       ('2, Static, synonyms', '0.82')     ('3, Good, normal', '0.86')         ('1, Good, short', '0.71')
('2, Good, normal', '0.87')         ('2, Static, normal', '0.82')       ('2, Good, normal', '0.86')         ('1, Kinetic, short', '0.70')
('2, Kinetic, long', '0.86')        ('2, Good, normal', '0.82')         ('2, Kinetic, long', '0.85')        ('2, Static, synonyms', '0.70')
('2, Static, normal', '0.86')       ('2, Kinetic, long', '0.82')        ('2, Static, synonyms', '0.85')     ('2, Good, long', '0.69')
('2, Static, synonyms', '0.86')     ('2, Kinetic, normal', '0.79')      ('2, Kinetic, normal', '0.85')      ('2, Good, normal', '0.69')
('2, Kinetic, synonyms', '0.85')    ('2, Kinetic, synonyms', '0.78')    ('2, Static, normal', '0.85')       ('2, Good, short', '0.69')
('2, Kinetic, normal', '0.85')      ('1, Good, typos', '0.78')          ('2, Good, short', '0.84')          ('2, Kinetic, typos', '0.68')
('3, Good, normal', '0.85')         ('3, Good, normal', '0.76')         ('2, Kinetic, synonyms', '0.84')    ('3, Good, normal', '0.68')
('2, Good, typos', '0.85')          ('1, Good, short', '0.75')          ('4, Good, normal', '0.83')         ('2, Kinetic, long', '0.67')
('2, Static, typos', '0.84')        ('2, Good, typos', '0.75')          ('1, Good, typos', '0.83')          ('2, Static, normal', '0.66')
('4, Good, normal', '0.83')         ('2, Static, typos', '0.75')        ('2, Good, typos', '0.81')          ('1, Static, short', '0.66')
('1, Static, normal', '0.82')       ('1, Static, synonyms', '0.72')     ('1, Good, short', '0.81')          ('2, Kinetic, short', '0.65')
('2, Kinetic, typos', '0.82')       ('2, Good, short', '0.72')          ('1, Static, synonyms', '0.81')     ('2, Kinetic, synonyms', '0.65')
('1, Good, short', '0.82')          ('4, Good, normal', '0.72')         ('1, Static, normal', '0.80')       ('1, Static, synonyms', '0.65')
('2, Kinetic, short', '0.81')       ('1, Static, short', '0.70')        ('1, Static, typos', '0.78')        ('4, Good, normal', '0.63')
('1, Static, typos', '0.80')        ('2, Kinetic, typos', '0.68')       ('1, Static, short', '0.77')        ('2, Static, long', '0.63')
('2, Good, short', '0.80')          ('1, Static, normal', '0.68')       ('2, Kinetic, short', '0.76')       ('1, Static, normal', '0.62')
('1, Static, synonyms', '0.80')     ('2, Kinetic, short', '0.66')       ('2, Static, typos', '0.75')        ('2, Static, short', '0.61')
('1, Static, short', '0.77')        ('2, Static, short', '0.65')        ('2, Static, short', '0.74')        ('1, Good, typos', '0.60')
('2, Static, short', '0.77')        ('1, Static, typos', '0.61')        ('2, Kinetic, typos', '0.72')       ('2, Static, typos', '0.60')
('1, Kinetic, short', '0.77')       ('1, irrelevant', '0.60')           ('1, irrelevant', '0.71')           ('1, Static, typos', '0.54')
('1, irrelevant', '0.74')           ('1, Kinetic, short', '0.59')       ('1, Kinetic, short', '0.68')       ('1, irrelevant', '0.29')
('2, irrelevant', '0.73')           ('2, irrelevant', '0.58')           ('2, irrelevant', '0.65')           ('2, irrelevant', '0.15')
```

Figure .13: The results from comparison to answer labeled "1, Kinetic, synonyms".



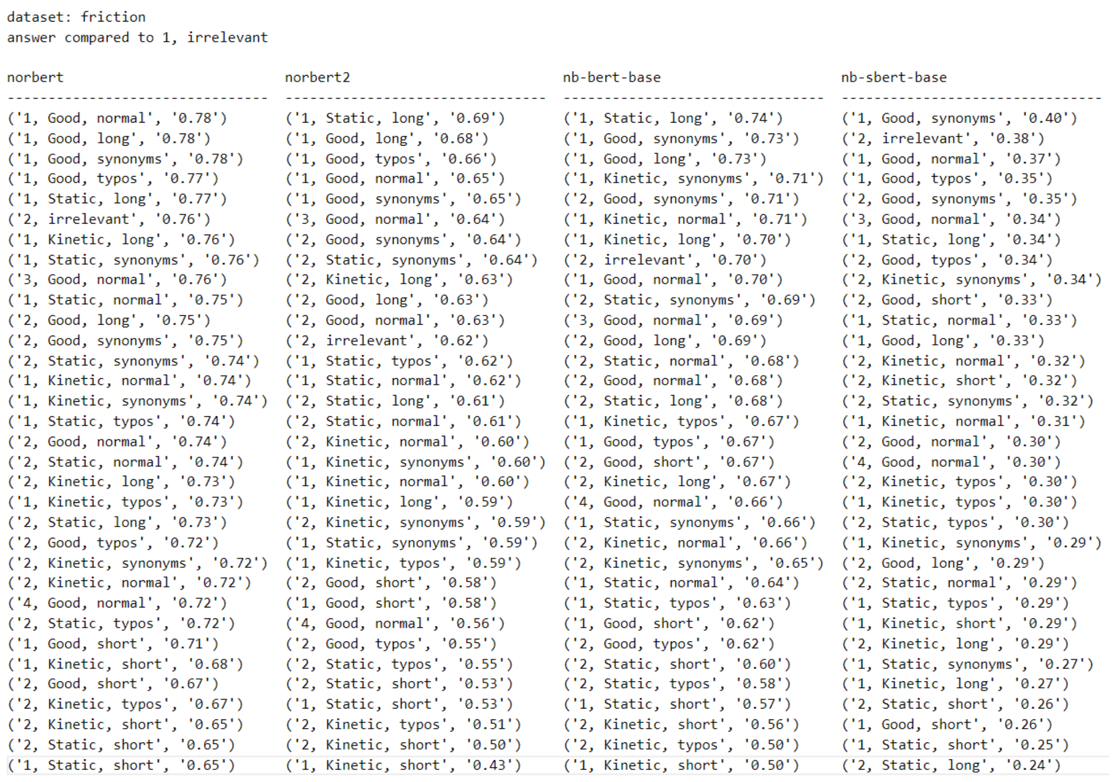
```
dataset: friction
answer compared to 1, Kinetic, short

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Good, short', '0.90')        ('2, Kinetic, short', '0.84')     ('2, Kinetic, short', '0.87')     ('1, Good, synonyms', '0.82')
('1, Static, short', '0.88')      ('1, Good, short', '0.83')        ('1, Static, normal', '0.86')     ('2, Good, short', '0.82')
('1, Static, normal', '0.87')     ('1, Static, short', '0.82')      ('1, Good, short', '0.85')        ('1, Good, short', '0.80')
('4, Good, normal', '0.87')       ('1, Static, synonyms', '0.80')   ('1, Static, synonyms', '0.85')   ('1, Good, normal', '0.80')
('3, Good, normal', '0.86')       ('2, Static, short', '0.79')      ('1, Static, short', '0.84')      ('4, Good, normal', '0.80')
('1, Static, synonyms', '0.86')   ('3, Good, normal', '0.79')       ('2, Static, short', '0.84')      ('3, Good, normal', '0.80')
('2, Kinetic, short', '0.86')     ('2, Good, short', '0.78')        ('4, Good, normal', '0.83')       ('2, Good, synonyms', '0.78')
('2, Kinetic, typos', '0.85')     ('1, Static, normal', '0.76')     ('2, Good, short', '0.83')        ('2, Static, synonyms', '0.77')
('1, Good, normal', '0.85')       ('4, Good, normal', '0.74')       ('3, Good, normal', '0.81')       ('2, Good, normal', '0.77')
('1, Static, typos', '0.84')      ('2, Kinetic, normal', '0.67')    ('2, Kinetic, normal', '0.81')    ('2, Static, short', '0.77')
('2, Good, typos', '0.84')        ('2, Good, long', '0.66')         ('1, Static, typos', '0.80')      ('2, Good, long', '0.76')
('1, Good, typos', '0.83')        ('2, Good, normal', '0.66')       ('2, Kinetic, synonyms', '0.79')  ('2, Kinetic, short', '0.75')
('2, Good, long', '0.83')         ('2, Good, synonyms', '0.65')     ('1, Good, normal', '0.78')       ('2, Kinetic, normal', '0.75')
('2, Kinetic, normal', '0.82')    ('2, Static, long', '0.65')       ('2, Good, long', '0.77')         ('2, Good, typos', '0.75')
('1, Good, long', '0.82')         ('2, Kinetic, synonyms', '0.64')  ('2, Good, normal', '0.76')       ('2, Static, normal', '0.75')
('1, Good, synonyms', '0.82')     ('1, Static, typos', '0.64')      ('2, Kinetic, long', '0.75')      ('1, Static, normal', '0.75')
('2, Static, typos', '0.81')      ('2, Static, synonyms', '0.63')   ('2, Good, synonyms', '0.75')     ('1, Static, synonyms', '0.74')
('2, Kinetic, long', '0.81')      ('2, Kinetic, long', '0.62')      ('2, Static, normal', '0.75')     ('1, Static, short', '0.74')
('2, Good, normal', '0.81')       ('1, Good, normal', '0.62')       ('1, Good, synonyms', '0.75')     ('1, Good, long', '0.73')
('2, Kinetic, synonyms', '0.81')  ('2, Static, normal', '0.62')     ('2, Static, synonyms', '0.74')   ('1, Kinetic, synonyms', '0.70')
('2, Static, long', '0.80')       ('1, Good, long', '0.61')         ('2, Static, long', '0.73')       ('2, Static, long', '0.70')
('1, Static, long', '0.80')       ('1, Good, synonyms', '0.61')     ('1, Good, long', '0.73')         ('2, Kinetic, typos', '0.69')
('2, Static, normal', '0.80')     ('1, Kinetic, normal', '0.60')    ('1, Good, typos', '0.71')        ('1, Good, typos', '0.69')
('2, Good, short', '0.79')        ('2, Static, typos', '0.59')      ('2, Good, typos', '0.70')        ('1, Static, long', '0.68')
('2, Good, synonyms', '0.79')     ('1, Kinetic, synonyms', '0.59')  ('1, Static, long', '0.70')       ('1, Kinetic, normal', '0.68')
('1, Kinetic, long', '0.79')      ('1, Static, long', '0.58')       ('1, Kinetic, normal', '0.70')    ('2, Kinetic, long', '0.67')
('2, Static, short', '0.79')      ('2, Good, typos', '0.57')        ('1, Kinetic, long', '0.69')      ('1, Static, typos', '0.67')
('2, Static, synonyms', '0.78')   ('1, Good, typos', '0.56')        ('1, Kinetic, synonyms', '0.68')  ('1, Kinetic, long', '0.66')
('1, Kinetic, typos', '0.77')     ('1, Kinetic, long', '0.55')      ('1, Kinetic, typos', '0.64')     ('2, Kinetic, synonyms', '0.66')
('1, Kinetic, normal', '0.77')    ('2, Kinetic, typos', '0.55')     ('2, Static, typos', '0.63')      ('2, Static, typos', '0.63')
('1, Kinetic, synonyms', '0.77')  ('1, Kinetic, typos', '0.47')     ('2, Kinetic, typos', '0.63')     ('1, Kinetic, typos', '0.62')
('1, irrelevant', '0.68')         ('1, irrelevant', '0.43')         ('1, irrelevant', '0.50')         ('1, irrelevant', '0.29')
('2, irrelevant', '0.61')         ('2, irrelevant', '0.28')         ('2, irrelevant', '0.45')         ('2, irrelevant', '0.14')
```

Figure .14: The results from comparison to answer labeled "1, Kinetic, short".



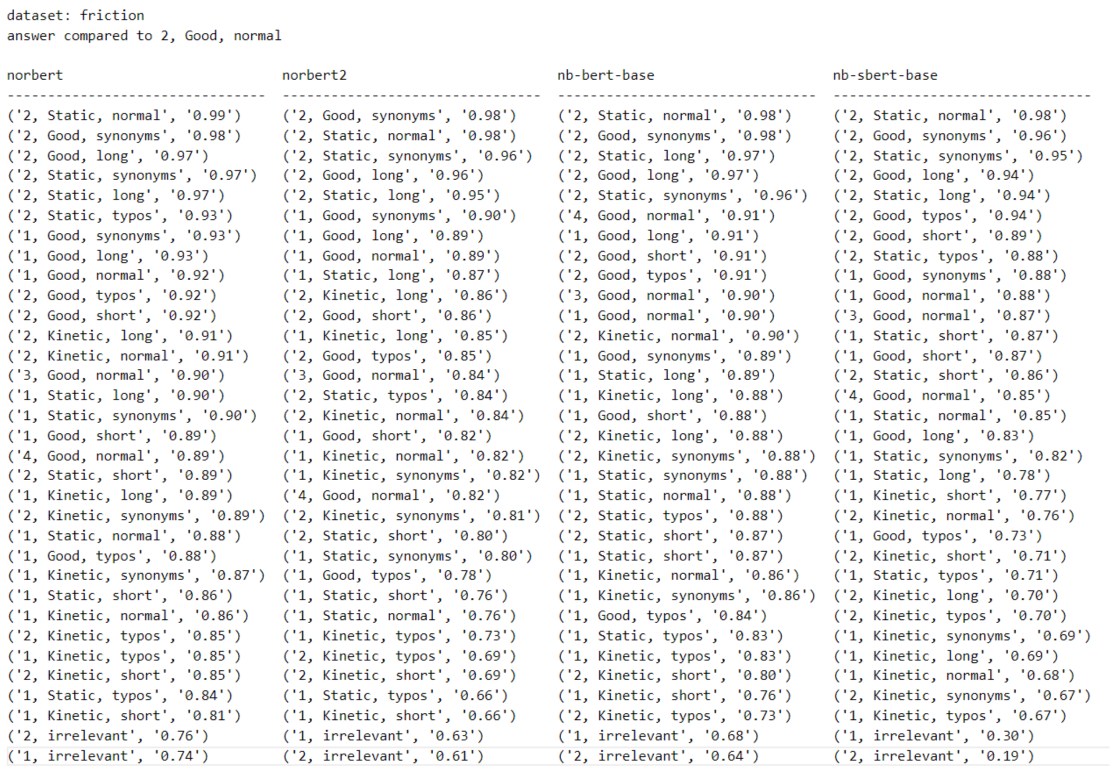
```
dataset: friction
answer compared to 1, Kinetic, long

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Good, long', '0.96')         ('1, Good, long', '0.93')         ('1, Good, long', '0.97')         ('1, Good, long', '0.91')
('1, Good, normal', '0.96')       ('1, Good, normal', '0.93')       ('1, Kinetic, normal', '0.96')    ('1, Kinetic, normal', '0.88')
('1, Kinetic, normal', '0.95')    ('1, Kinetic, normal', '0.93')    ('1, Kinetic, synonyms', '0.95')  ('1, Kinetic, typos', '0.85')
('1, Good, synonyms', '0.94')     ('1, Good, synonyms', '0.92')     ('1, Static, long', '0.95')       ('1, Static, long', '0.84')
('1, Static, long', '0.94')       ('1, Static, long', '0.91')       ('1, Kinetic, typos', '0.93')     ('1, Kinetic, synonyms', '0.83')
('1, Kinetic, synonyms', '0.94')  ('1, Kinetic, synonyms', '0.91')  ('2, Good, long', '0.91')         ('2, Kinetic, long', '0.80')
('1, Kinetic, typos', '0.93')     ('2, Good, long', '0.88')         ('1, Good, synonyms', '0.91')     ('2, Kinetic, synonyms', '0.79')
('1, Good, typos', '0.93')        ('1, Kinetic, typos', '0.86')     ('1, Good, normal', '0.91')       ('2, Kinetic, normal', '0.78')
('2, Good, long', '0.92')         ('2, Static, long', '0.86')       ('2, Static, long', '0.90')       ('2, Kinetic, typos', '0.77')
('2, Static, long', '0.90')       ('2, Kinetic, long', '0.86')      ('2, Kinetic, long', '0.89')      ('1, Good, synonyms', '0.76')
('2, Kinetic, long', '0.90')      ('2, Good, synonyms', '0.85')     ('2, Good, synonyms', '0.89')     ('1, Good, normal', '0.76')
('2, Good, normal', '0.89')       ('2, Static, synonyms', '0.85')   ('2, Good, normal', '0.88')       ('2, Good, long', '0.76')
('2, Good, synonyms', '0.89')     ('2, Good, normal', '0.85')       ('3, Good, normal', '0.88')       ('2, Good, typos', '0.74')
('2, Kinetic, synonyms', '0.89')  ('2, Static, normal', '0.85')     ('2, Kinetic, normal', '0.88')    ('2, Kinetic, short', '0.73')
('2, Kinetic, normal', '0.89')    ('1, Good, typos', '0.83')        ('2, Static, normal', '0.87')     ('3, Good, normal', '0.72')
('3, Good, normal', '0.88')       ('2, Kinetic, normal', '0.80')    ('2, Static, synonyms', '0.87')   ('2, Good, short', '0.71')
('2, Static, normal', '0.88')     ('2, Kinetic, synonyms', '0.80')  ('2, Kinetic, synonyms', '0.87')  ('2, Good, synonyms', '0.70')
('2, Good, typos', '0.88')        ('2, Good, typos', '0.78')        ('2, Good, short', '0.85')        ('2, Good, normal', '0.69')
('2, Static, typos', '0.88')      ('3, Good, normal', '0.77')       ('4, Good, normal', '0.85')       ('4, Good, normal', '0.69')
('2, Static, synonyms', '0.87')   ('2, Static, typos', '0.77')      ('2, Good, typos', '0.84')        ('1, Static, short', '0.67')
('2, Kinetic, short', '0.85')     ('4, Good, normal', '0.74')       ('1, Good, typos', '0.84')        ('1, Good, short', '0.66')
('2, Kinetic, typos', '0.85')     ('2, Good, short', '0.73')        ('1, Static, normal', '0.81')     ('1, Kinetic, short', '0.66')
('2, Good, short', '0.85')        ('1, Good, short', '0.73')        ('1, Static, synonyms', '0.81')   ('2, Static, long', '0.65')
('4, Good, normal', '0.85')       ('2, Kinetic, typos', '0.71')     ('1, Good, short', '0.80')        ('2, Static, normal', '0.65')
('1, Static, normal', '0.85')     ('1, Static, synonyms', '0.71')   ('1, Static, typos', '0.78')      ('2, Static, synonyms', '0.65')
('1, Good, short', '0.85')        ('1, Static, short', '0.68')      ('2, Static, typos', '0.78')      ('1, Good, typos', '0.62')
('1, Static, synonyms', '0.83')   ('1, Static, normal', '0.67')     ('1, Static, short', '0.77')      ('1, Static, normal', '0.62')
('1, Static, typos', '0.82')      ('2, Kinetic, short', '0.66')     ('2, Kinetic, short', '0.77')     ('2, Static, typos', '0.60')
('2, Static, short', '0.80')      ('2, Static, short', '0.64')      ('2, Static, short', '0.75')      ('2, Static, short', '0.58')
('1, Static, short', '0.80')      ('2, irrelevant', '0.62')         ('2, Kinetic, typos', '0.73')     ('1, Static, synonyms', '0.56')
('1, Kinetic, short', '0.79')     ('1, Static, typos', '0.61')      ('1, irrelevant', '0.70')         ('1, Static, typos', '0.51')
('1, irrelevant', '0.76')         ('1, irrelevant', '0.59')         ('1, Kinetic, short', '0.69')     ('1, irrelevant', '0.27')
('2, irrelevant', '0.73')         ('1, Kinetic, short', '0.55')     ('2, irrelevant', '0.64')         ('2, irrelevant', '0.12')
```

Figure .15: The results from comparison to answer labeled "1, Kinetic, long".



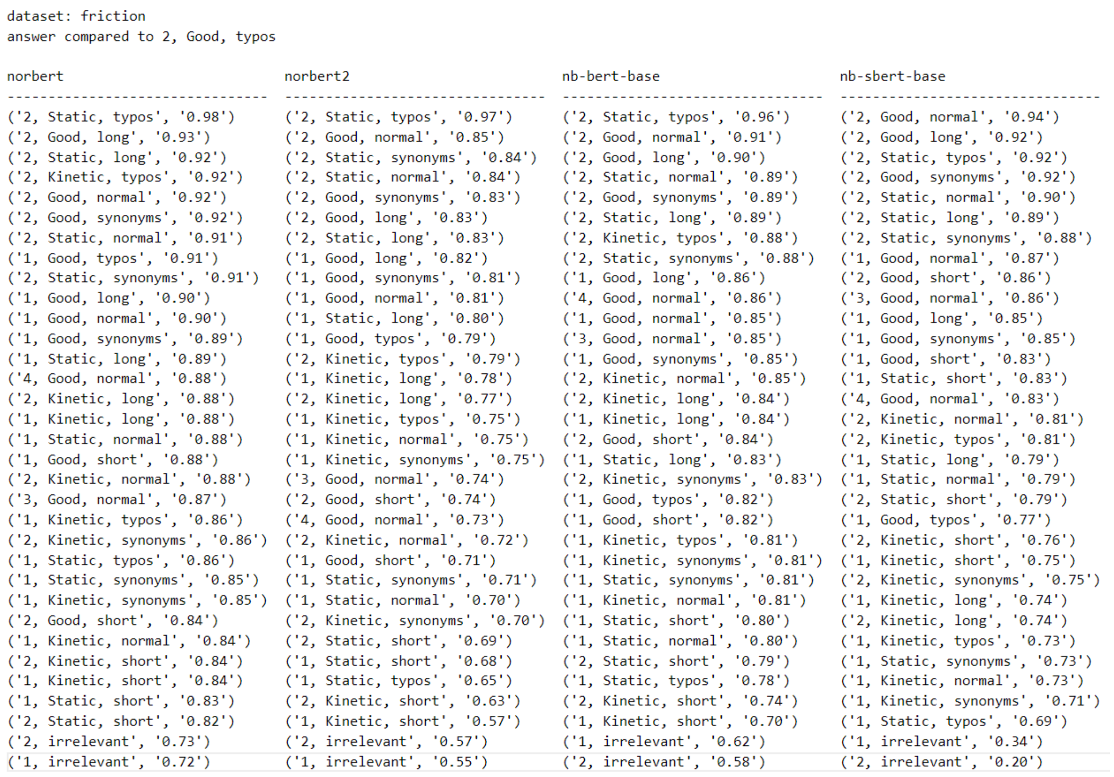
```
dataset: friction
answer compared to 1, irrelevant

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
--------------------------------   --------------------------------   --------------------------------   --------------------------------
('1, Good, normal', '0.78')        ('1, Static, long', '0.69')        ('1, Static, long', '0.74')        ('1, Good, synonyms', '0.40')
('1, Good, long', '0.78')          ('1, Good, long', '0.68')          ('1, Good, synonyms', '0.73')      ('2, irrelevant', '0.38')
('1, Good, synonyms', '0.78')      ('1, Good, typos', '0.66')         ('1, Good, long', '0.73')          ('1, Good, normal', '0.37')
('1, Good, typos', '0.77')         ('1, Good, normal', '0.65')        ('1, Kinetic, synonyms', '0.71')   ('1, Good, typos', '0.35')
('1, Static, long', '0.77')        ('1, Good, synonyms', '0.65')      ('2, Good, synonyms', '0.71')      ('2, Good, synonyms', '0.35')
('2, irrelevant', '0.76')          ('3, Good, normal', '0.64')        ('1, Kinetic, normal', '0.71')     ('3, Good, normal', '0.34')
('1, Kinetic, long', '0.76')       ('2, Good, synonyms', '0.64')      ('1, Kinetic, long', '0.70')       ('1, Static, long', '0.34')
('1, Static, synonyms', '0.76')    ('2, Static, synonyms', '0.64')    ('2, irrelevant', '0.70')          ('2, Good, typos', '0.34')
('3, Good, normal', '0.76')        ('2, Kinetic, long', '0.63')       ('1, Good, normal', '0.70')        ('2, Kinetic, synonyms', '0.34')
('1, Static, normal', '0.75')      ('2, Good, long', '0.63')          ('2, Static, synonyms', '0.69')    ('2, Good, short', '0.33')
('2, Good, long', '0.75')          ('2, Good, normal', '0.63')        ('3, Good, normal', '0.69')        ('1, Static, normal', '0.33')
('2, Good, synonyms', '0.75')      ('2, irrelevant', '0.62')          ('2, Good, long', '0.69')          ('1, Good, long', '0.33')
('2, Static, synonyms', '0.74')    ('1, Static, typos', '0.62')       ('2, Static, normal', '0.68')      ('2, Kinetic, normal', '0.32')
('1, Kinetic, normal', '0.74')     ('1, Static, normal', '0.62')      ('2, Good, normal', '0.68')        ('2, Kinetic, short', '0.32')
('1, Kinetic, synonyms', '0.74')   ('2, Static, long', '0.61')        ('2, Static, long', '0.68')        ('2, Static, synonyms', '0.32')
('1, Static, typos', '0.74')       ('2, Static, normal', '0.61')      ('1, Kinetic, typos', '0.67')      ('1, Kinetic, normal', '0.31')
('2, Good, normal', '0.74')        ('2, Kinetic, normal', '0.60')     ('1, Good, typos', '0.67')         ('2, Good, normal', '0.30')
('2, Static, normal', '0.74')      ('1, Kinetic, synonyms', '0.60')   ('2, Good, short', '0.67')         ('4, Good, normal', '0.30')
('2, Kinetic, long', '0.73')       ('1, Kinetic, normal', '0.60')     ('2, Kinetic, long', '0.67')       ('2, Kinetic, typos', '0.30')
('1, Kinetic, typos', '0.73')      ('1, Kinetic, long', '0.59')       ('4, Good, normal', '0.66')        ('1, Kinetic, typos', '0.30')
('2, Static, long', '0.73')        ('2, Kinetic, synonyms', '0.59')   ('1, Static, synonyms', '0.66')    ('2, Static, typos', '0.30')
('2, Good, typos', '0.72')         ('1, Static, synonyms', '0.59')    ('2, Kinetic, normal', '0.66')     ('1, Kinetic, synonyms', '0.29')
('2, Kinetic, synonyms', '0.72')   ('1, Kinetic, typos', '0.59')      ('2, Kinetic, synonyms', '0.65')   ('2, Good, long', '0.29')
('2, Kinetic, normal', '0.72')     ('2, Good, short', '0.58')         ('1, Static, normal', '0.64')      ('2, Static, normal', '0.29')
('4, Good, normal', '0.72')        ('1, Good, short', '0.58')         ('1, Static, typos', '0.63')       ('1, Static, typos', '0.29')
('2, Static, typos', '0.72')       ('4, Good, normal', '0.56')        ('1, Good, short', '0.62')         ('1, Kinetic, short', '0.29')
('1, Good, short', '0.71')         ('2, Good, typos', '0.55')         ('2, Good, typos', '0.62')         ('2, Kinetic, long', '0.29')
('1, Kinetic, short', '0.68')      ('2, Static, typos', '0.55')       ('2, Static, short', '0.60')       ('1, Static, synonyms', '0.27')
('2, Good, short', '0.67')         ('2, Static, short', '0.53')       ('2, Static, typos', '0.58')       ('1, Kinetic, long', '0.27')
('2, Kinetic, typos', '0.67')      ('1, Static, short', '0.53')       ('1, Static, short', '0.57')       ('2, Static, short', '0.26')
('2, Kinetic, short', '0.65')      ('2, Kinetic, typos', '0.51')      ('2, Kinetic, short', '0.56')      ('1, Good, short', '0.26')
('2, Static, short', '0.65')       ('2, Kinetic, short', '0.50')      ('2, Kinetic, typos', '0.50')      ('1, Static, short', '0.25')
('1, Static, short', '0.65')       ('1, Kinetic, short', '0.43')      ('1, Kinetic, short', '0.50')      ('2, Static, long', '0.24')
```

Figure .16: The results from comparison to answer labeled "1, irrelevant".



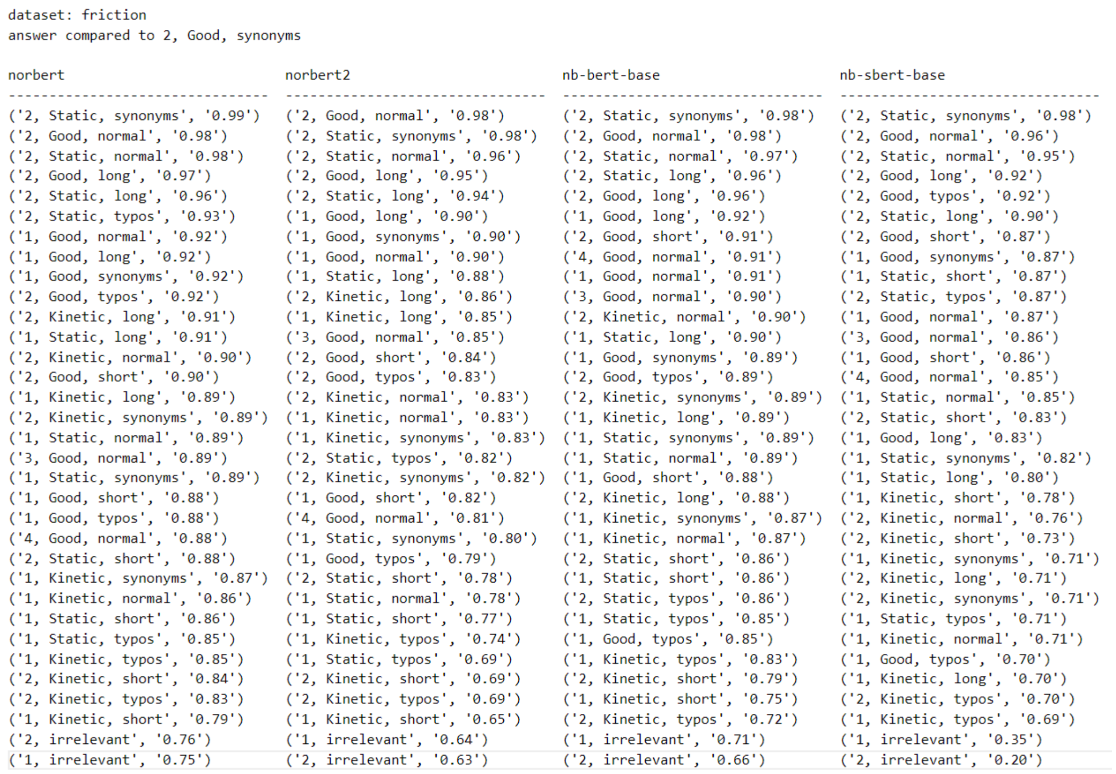
```
dataset: friction
answer compared to 2, Good, normal

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
--------------------------------   --------------------------------   --------------------------------   --------------------------------
('2, Static, normal', '0.99')      ('2, Good, synonyms', '0.98')      ('2, Static, normal', '0.98')      ('2, Static, normal', '0.98')
('2, Good, synonyms', '0.98')      ('2, Static, normal', '0.98')      ('2, Good, synonyms', '0.98')      ('2, Good, synonyms', '0.96')
('2, Good, long', '0.97')          ('2, Static, synonyms', '0.96')    ('2, Static, long', '0.97')        ('2, Static, synonyms', '0.95')
('2, Static, synonyms', '0.97')    ('2, Good, long', '0.96')          ('2, Good, long', '0.97')          ('2, Good, long', '0.94')
('2, Static, long', '0.97')        ('2, Static, long', '0.95')        ('2, Static, synonyms', '0.96')    ('2, Static, long', '0.94')
('2, Static, typos', '0.93')       ('1, Good, synonyms', '0.90')      ('4, Good, normal', '0.91')        ('2, Good, typos', '0.94')
('1, Good, synonyms', '0.93')      ('1, Good, long', '0.89')          ('1, Good, long', '0.91')          ('2, Good, short', '0.89')
('1, Good, long', '0.93')          ('1, Good, normal', '0.89')        ('2, Good, short', '0.91')         ('2, Static, typos', '0.88')
('1, Good, normal', '0.92')        ('1, Static, long', '0.87')        ('2, Good, typos', '0.91')         ('1, Good, synonyms', '0.88')
('2, Good, typos', '0.92')         ('2, Kinetic, long', '0.86')       ('3, Good, normal', '0.90')        ('1, Good, normal', '0.88')
('2, Good, short', '0.92')         ('2, Good, short', '0.86')         ('1, Good, normal', '0.90')        ('3, Good, normal', '0.87')
('2, Kinetic, long', '0.91')       ('1, Kinetic, long', '0.85')       ('2, Kinetic, normal', '0.90')     ('1, Static, short', '0.87')
('2, Kinetic, normal', '0.91')     ('2, Good, typos', '0.85')         ('1, Good, synonyms', '0.89')      ('1, Good, short', '0.87')
('3, Good, normal', '0.90')        ('3, Good, normal', '0.84')        ('1, Static, long', '0.89')        ('2, Static, short', '0.86')
('1, Static, long', '0.90')        ('2, Static, typos', '0.84')       ('1, Kinetic, long', '0.88')       ('4, Good, normal', '0.85')
('1, Static, synonyms', '0.90')    ('2, Kinetic, normal', '0.84')     ('1, Good, short', '0.88')         ('1, Static, normal', '0.85')
('1, Good, short', '0.89')         ('1, Good, short', '0.82')         ('2, Kinetic, long', '0.88')       ('1, Good, long', '0.83')
('4, Good, normal', '0.89')        ('1, Kinetic, normal', '0.82')     ('2, Kinetic, synonyms', '0.88')   ('1, Static, synonyms', '0.82')
('2, Static, short', '0.89')       ('1, Kinetic, synonyms', '0.82')   ('1, Static, synonyms', '0.88')    ('1, Static, long', '0.78')
('1, Kinetic, long', '0.89')       ('4, Good, normal', '0.82')        ('1, Static, normal', '0.88')      ('1, Kinetic, short', '0.77')
('2, Kinetic, synonyms', '0.89')   ('2, Kinetic, synonyms', '0.81')   ('2, Static, typos', '0.88')       ('2, Kinetic, normal', '0.76')
('1, Static, normal', '0.88')      ('2, Static, short', '0.80')       ('2, Static, short', '0.87')       ('1, Good, typos', '0.73')
('1, Good, typos', '0.88')         ('1, Static, synonyms', '0.80')    ('1, Static, short', '0.87')       ('2, Kinetic, short', '0.71')
('1, Kinetic, synonyms', '0.87')   ('1, Good, typos', '0.78')         ('1, Kinetic, normal', '0.86')     ('1, Static, typos', '0.71')
('1, Static, short', '0.86')       ('1, Static, short', '0.76')       ('1, Kinetic, synonyms', '0.86')   ('2, Kinetic, long', '0.70')
('1, Kinetic, normal', '0.86')     ('1, Static, normal', '0.76')      ('1, Good, typos', '0.84')         ('2, Kinetic, typos', '0.70')
('2, Kinetic, typos', '0.85')      ('1, Kinetic, typos', '0.73')      ('1, Static, typos', '0.83')       ('1, Kinetic, synonyms', '0.69')
('1, Kinetic, typos', '0.85')      ('2, Kinetic, typos', '0.69')      ('1, Kinetic, typos', '0.83')      ('1, Kinetic, long', '0.69')
('2, Kinetic, short', '0.85')      ('2, Kinetic, short', '0.69')      ('2, Kinetic, short', '0.80')      ('1, Kinetic, normal', '0.68')
('1, Static, typos', '0.84')       ('1, Static, typos', '0.66')       ('1, Kinetic, short', '0.76')      ('2, Kinetic, synonyms', '0.67')
('1, Kinetic, short', '0.81')      ('1, Kinetic, short', '0.66')      ('2, Kinetic, typos', '0.73')      ('1, Kinetic, typos', '0.67')
('2, irrelevant', '0.76')          ('1, irrelevant', '0.63')          ('1, irrelevant', '0.68')          ('1, irrelevant', '0.30')
('1, irrelevant', '0.74')          ('2, irrelevant', '0.61')          ('2, irrelevant', '0.64')          ('2, irrelevant', '0.19')
```

Figure .17: The results from comparison to answer labeled "2, Good, normal".



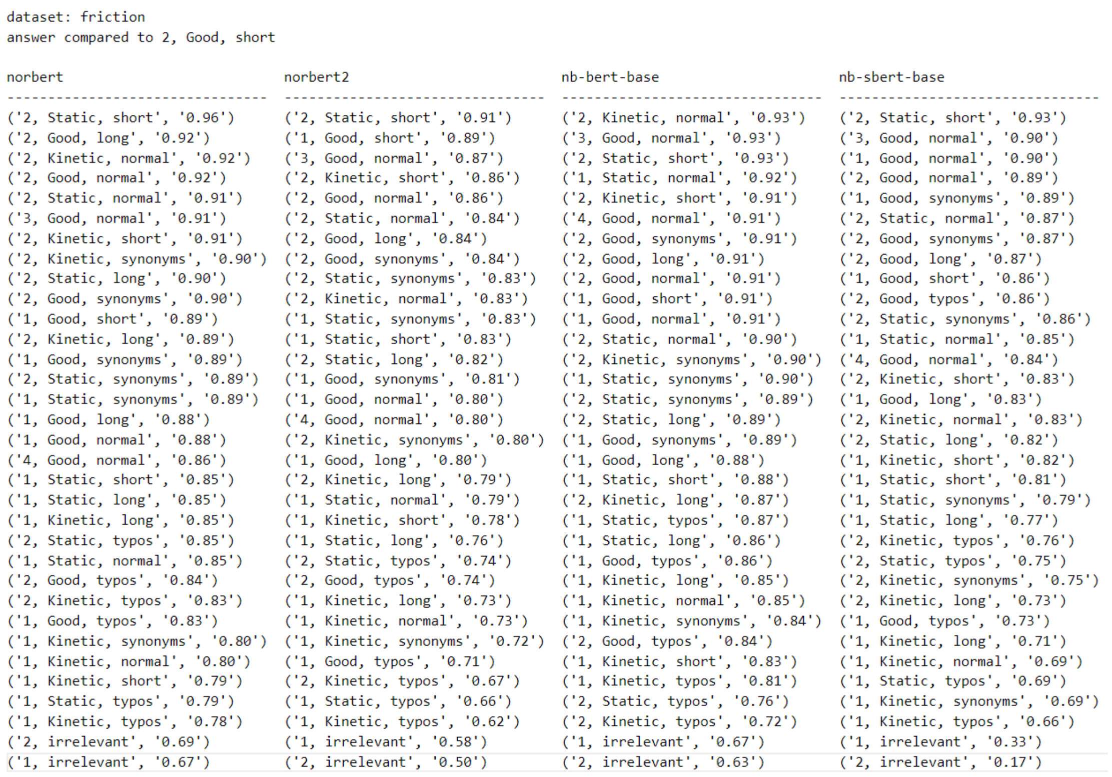
```
dataset: friction
answer compared to 2, Good, typos

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
---------------------------------  ---------------------------------  ---------------------------------  ---------------------------------
('2, Static, typos', '0.98')       ('2, Static, typos', '0.97')       ('2, Static, typos', '0.96')       ('2, Good, normal', '0.94')
('2, Good, long', '0.93')          ('2, Good, normal', '0.85')        ('2, Good, normal', '0.91')        ('2, Good, long', '0.92')
('2, Static, long', '0.92')        ('2, Static, synonyms', '0.84')    ('2, Good, long', '0.90')          ('2, Static, typos', '0.92')
('2, Kinetic, typos', '0.92')      ('2, Static, normal', '0.84')      ('2, Static, normal', '0.89')      ('2, Good, synonyms', '0.92')
('2, Good, normal', '0.92')        ('2, Good, synonyms', '0.83')      ('2, Good, synonyms', '0.89')      ('2, Static, normal', '0.90')
('2, Good, synonyms', '0.92')      ('2, Good, long', '0.83')          ('2, Static, long', '0.89')        ('2, Static, long', '0.89')
('2, Static, normal', '0.91')      ('2, Static, long', '0.83')        ('2, Kinetic, typos', '0.88')      ('2, Static, synonyms', '0.88')
('1, Good, typos', '0.91')         ('1, Good, long', '0.82')          ('2, Static, synonyms', '0.88')    ('1, Good, normal', '0.87')
('2, Static, synonyms', '0.91')    ('1, Good, synonyms', '0.81')      ('1, Good, long', '0.86')          ('2, Good, short', '0.86')
('1, Good, long', '0.90')          ('1, Good, normal', '0.81')        ('4, Good, normal', '0.86')        ('3, Good, normal', '0.86')
('1, Good, normal', '0.90')        ('1, Static, long', '0.80')        ('1, Good, normal', '0.85')        ('1, Good, long', '0.85')
('1, Good, synonyms', '0.89')      ('1, Good, typos', '0.79')         ('3, Good, normal', '0.85')        ('1, Good, synonyms', '0.85')
('1, Static, long', '0.89')        ('2, Kinetic, typos', '0.79')      ('1, Good, synonyms', '0.85')      ('1, Good, short', '0.83')
('4, Good, normal', '0.88')        ('1, Kinetic, long', '0.78')       ('2, Kinetic, normal', '0.85')     ('1, Static, short', '0.83')
('2, Kinetic, long', '0.88')       ('2, Kinetic, long', '0.77')       ('2, Kinetic, long', '0.84')       ('4, Good, normal', '0.83')
('1, Kinetic, long', '0.88')       ('1, Kinetic, typos', '0.75')      ('1, Kinetic, long', '0.84')       ('2, Kinetic, normal', '0.81')
('1, Static, normal', '0.88')      ('1, Kinetic, normal', '0.75')     ('2, Good, short', '0.84')         ('2, Kinetic, typos', '0.81')
('1, Good, short', '0.88')         ('1, Kinetic, synonyms', '0.75')   ('1, Static, long', '0.83')        ('1, Static, long', '0.79')
('2, Kinetic, normal', '0.88')     ('3, Good, normal', '0.74')        ('2, Kinetic, synonyms', '0.83')   ('1, Static, normal', '0.79')
('3, Good, normal', '0.87')        ('2, Good, short', '0.74')         ('1, Good, typos', '0.82')         ('2, Static, short', '0.79')
('1, Kinetic, typos', '0.86')      ('4, Good, normal', '0.73')        ('1, Good, short', '0.82')         ('1, Good, typos', '0.77')
('2, Kinetic, synonyms', '0.86')   ('2, Kinetic, normal', '0.72')     ('1, Kinetic, typos', '0.81')      ('2, Kinetic, short', '0.76')
('1, Static, typos', '0.86')       ('1, Good, short', '0.71')         ('1, Kinetic, synonyms', '0.81')   ('1, Kinetic, short', '0.75')
('1, Static, synonyms', '0.85')    ('1, Static, synonyms', '0.71')    ('1, Static, synonyms', '0.81')    ('2, Kinetic, synonyms', '0.75')
('1, Kinetic, synonyms', '0.85')   ('1, Static, normal', '0.70')      ('1, Kinetic, normal', '0.81')     ('1, Kinetic, long', '0.74')
('2, Good, short', '0.84')         ('2, Kinetic, synonyms', '0.70')   ('1, Static, short', '0.80')       ('2, Kinetic, long', '0.74')
('1, Kinetic, normal', '0.84')     ('2, Static, short', '0.69')       ('1, Static, normal', '0.80')      ('1, Kinetic, typos', '0.73')
('2, Kinetic, short', '0.84')      ('1, Static, short', '0.68')       ('2, Static, short', '0.79')       ('1, Static, synonyms', '0.73')
('1, Kinetic, short', '0.84')      ('1, Static, typos', '0.65')       ('1, Static, typos', '0.78')       ('1, Kinetic, normal', '0.73')
('1, Static, short', '0.83')       ('2, Kinetic, short', '0.63')      ('2, Kinetic, short', '0.74')      ('1, Kinetic, synonyms', '0.71')
('2, Static, short', '0.82')       ('1, Kinetic, short', '0.57')      ('1, Kinetic, short', '0.70')      ('1, Static, typos', '0.69')
('2, irrelevant', '0.73')          ('2, irrelevant', '0.57')          ('1, irrelevant', '0.62')          ('1, irrelevant', '0.34')
('1, irrelevant', '0.72')          ('1, irrelevant', '0.55')          ('2, irrelevant', '0.58')          ('2, irrelevant', '0.20')
```

Figure .18: The results from comparison to answer labeled "2, Good, typos".



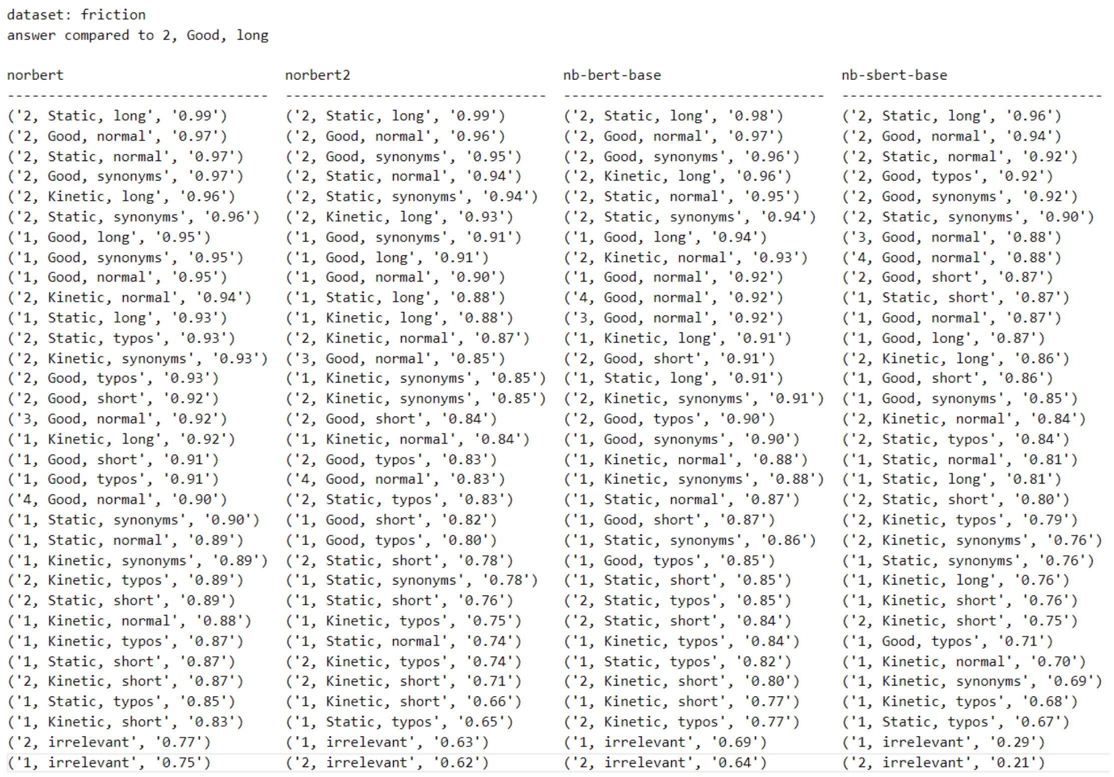
```
dataset: friction
answer compared to 2, Good, synonyms

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
---------------------------------  ---------------------------------  ---------------------------------  ---------------------------------
('2, Static, synonyms', '0.99')    ('2, Good, normal', '0.98')        ('2, Static, synonyms', '0.98')    ('2, Static, synonyms', '0.98')
('2, Good, normal', '0.98')        ('2, Static, synonyms', '0.98')    ('2, Good, normal', '0.98')        ('2, Good, normal', '0.96')
('2, Static, normal', '0.98')      ('2, Static, normal', '0.96')      ('2, Static, normal', '0.97')      ('2, Static, normal', '0.95')
('2, Good, long', '0.97')          ('2, Good, long', '0.95')          ('2, Static, long', '0.96')        ('2, Good, long', '0.92')
('2, Static, long', '0.96')        ('2, Static, long', '0.94')        ('2, Good, long', '0.96')          ('2, Good, typos', '0.92')
('2, Static, typos', '0.93')       ('1, Good, long', '0.90')          ('1, Good, long', '0.92')          ('2, Static, long', '0.90')
('1, Good, normal', '0.92')        ('1, Good, synonyms', '0.90')      ('2, Good, short', '0.91')         ('2, Good, short', '0.87')
('1, Good, long', '0.92')          ('1, Good, normal', '0.90')        ('4, Good, normal', '0.91')        ('1, Good, synonyms', '0.87')
('1, Good, synonyms', '0.92')      ('1, Static, long', '0.88')        ('1, Good, normal', '0.91')        ('1, Static, short', '0.87')
('2, Good, typos', '0.92')         ('2, Kinetic, long', '0.86')       ('3, Good, normal', '0.90')        ('2, Static, typos', '0.87')
('2, Kinetic, long', '0.91')       ('1, Kinetic, long', '0.85')       ('2, Kinetic, normal', '0.90')     ('1, Good, normal', '0.87')
('1, Static, long', '0.91')        ('3, Good, normal', '0.85')        ('1, Static, long', '0.90')        ('3, Good, normal', '0.86')
('2, Kinetic, normal', '0.90')     ('2, Good, short', '0.84')         ('1, Good, synonyms', '0.89')      ('1, Good, short', '0.86')
('2, Good, short', '0.90')         ('2, Good, typos', '0.83')         ('2, Good, typos', '0.89')         ('4, Good, normal', '0.85')
('1, Kinetic, long', '0.89')       ('2, Kinetic, normal', '0.83')     ('2, Kinetic, synonyms', '0.89')   ('1, Static, normal', '0.85')
('2, Kinetic, synonyms', '0.89')   ('1, Kinetic, normal', '0.83')     ('1, Kinetic, long', '0.89')       ('2, Static, short', '0.83')
('1, Static, normal', '0.89')      ('1, Kinetic, synonyms', '0.83')   ('1, Static, synonyms', '0.89')    ('1, Good, long', '0.83')
('3, Good, normal', '0.89')        ('2, Static, typos', '0.82')       ('1, Static, normal', '0.89')      ('1, Static, synonyms', '0.82')
('1, Static, synonyms', '0.89')    ('2, Kinetic, synonyms', '0.82')   ('1, Good, short', '0.88')         ('1, Static, long', '0.80')
('1, Good, short', '0.88')         ('1, Good, short', '0.82')         ('2, Kinetic, long', '0.88')       ('1, Kinetic, short', '0.78')
('1, Good, typos', '0.88')         ('4, Good, normal', '0.81')        ('1, Kinetic, synonyms', '0.87')   ('2, Kinetic, normal', '0.76')
('4, Good, normal', '0.88')        ('1, Static, synonyms', '0.80')    ('1, Kinetic, normal', '0.87')     ('2, Kinetic, short', '0.73')
('2, Static, short', '0.88')       ('1, Good, typos', '0.79')         ('2, Static, short', '0.86')       ('1, Kinetic, synonyms', '0.71')
('1, Kinetic, synonyms', '0.87')   ('2, Static, short', '0.78')       ('1, Static, short', '0.86')       ('2, Kinetic, long', '0.71')
('1, Kinetic, normal', '0.86')     ('1, Static, normal', '0.78')      ('2, Static, typos', '0.86')       ('2, Kinetic, synonyms', '0.71')
('1, Static, short', '0.86')       ('1, Static, short', '0.77')       ('1, Static, typos', '0.85')       ('1, Static, typos', '0.71')
('1, Static, typos', '0.85')       ('1, Kinetic, typos', '0.74')      ('1, Good, typos', '0.85')         ('1, Kinetic, normal', '0.71')
('1, Kinetic, typos', '0.85')      ('1, Static, typos', '0.69')       ('1, Kinetic, typos', '0.83')      ('1, Good, typos', '0.70')
('2, Kinetic, short', '0.84')      ('2, Kinetic, short', '0.69')      ('2, Kinetic, short', '0.79')      ('1, Kinetic, long', '0.70')
('2, Kinetic, typos', '0.83')      ('2, Kinetic, typos', '0.69')      ('1, Kinetic, short', '0.75')      ('2, Kinetic, typos', '0.70')
('1, Kinetic, short', '0.79')      ('1, Kinetic, short', '0.65')      ('2, Kinetic, typos', '0.72')      ('1, Kinetic, typos', '0.69')
('2, irrelevant', '0.76')          ('1, irrelevant', '0.64')          ('1, irrelevant', '0.71')          ('1, irrelevant', '0.35')
('1, irrelevant', '0.75')          ('2, irrelevant', '0.63')          ('2, irrelevant', '0.66')          ('2, irrelevant', '0.20')
```

Figure .19: The results from comparison to answer labeled "2, Good, synonyms".



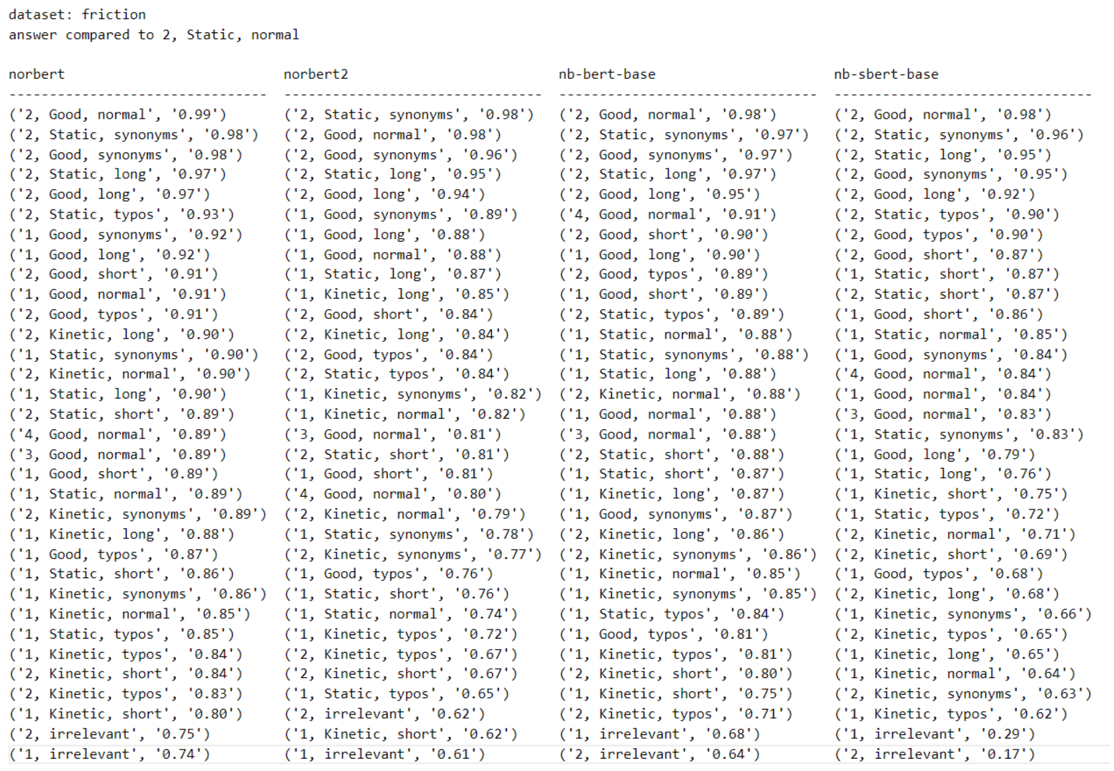
```
dataset: friction
answer compared to 2, Good, short

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('2, Static, short', '0.96')      ('2, Static, short', '0.91')      ('2, Kinetic, normal', '0.93')    ('2, Static, short', '0.93')
('2, Good, long', '0.92')         ('1, Good, short', '0.89')        ('3, Good, normal', '0.93')       ('3, Good, normal', '0.90')
('2, Kinetic, normal', '0.92')    ('3, Good, normal', '0.87')       ('2, Static, short', '0.93')      ('1, Good, normal', '0.90')
('2, Good, normal', '0.92')       ('2, Kinetic, short', '0.86')     ('1, Static, normal', '0.92')     ('2, Good, normal', '0.89')
('2, Static, normal', '0.91')     ('2, Good, normal', '0.86')       ('2, Kinetic, short', '0.91')     ('1, Good, synonyms', '0.89')
('3, Good, normal', '0.91')       ('2, Static, normal', '0.84')     ('4, Good, normal', '0.91')       ('2, Static, normal', '0.87')
('2, Kinetic, short', '0.91')     ('2, Good, long', '0.84')         ('2, Good, synonyms', '0.91')     ('2, Good, synonyms', '0.87')
('2, Kinetic, synonyms', '0.90')  ('2, Good, synonyms', '0.84')     ('2, Good, long', '0.91')         ('2, Good, long', '0.87')
('2, Static, long', '0.90')       ('2, Static, synonyms', '0.83')   ('2, Good, normal', '0.91')       ('1, Good, short', '0.86')
('2, Good, synonyms', '0.90')     ('2, Kinetic, normal', '0.83')    ('1, Good, short', '0.91')        ('2, Good, typos', '0.86')
('1, Good, short', '0.89')        ('1, Static, synonyms', '0.83')   ('1, Good, normal', '0.91')       ('2, Static, synonyms', '0.86')
('2, Kinetic, long', '0.89')      ('1, Static, short', '0.83')      ('2, Static, normal', '0.90')     ('1, Static, normal', '0.85')
('1, Good, synonyms', '0.89')     ('2, Static, long', '0.82')       ('2, Kinetic, synonyms', '0.90')  ('4, Good, normal', '0.84')
('2, Static, synonyms', '0.89')   ('1, Good, synonyms', '0.81')     ('1, Static, synonyms', '0.90')   ('2, Kinetic, short', '0.83')
('1, Static, synonyms', '0.89')   ('1, Good, normal', '0.80')       ('2, Static, synonyms', '0.89')   ('1, Good, long', '0.83')
('1, Good, long', '0.88')         ('4, Good, normal', '0.80')       ('2, Static, long', '0.89')       ('2, Kinetic, normal', '0.83')
('1, Good, normal', '0.88')       ('2, Kinetic, synonyms', '0.80')  ('1, Good, synonyms', '0.89')     ('2, Static, long', '0.82')
('4, Good, normal', '0.86')       ('1, Good, long', '0.80')         ('1, Good, long', '0.88')         ('1, Kinetic, short', '0.82')
('1, Static, short', '0.85')      ('2, Kinetic, long', '0.79')      ('1, Static, short', '0.88')      ('1, Static, short', '0.81')
('1, Static, long', '0.85')       ('1, Static, normal', '0.79')     ('2, Kinetic, long', '0.87')      ('1, Static, synonyms', '0.79')
('1, Kinetic, long', '0.85')      ('1, Kinetic, short', '0.78')     ('1, Static, typos', '0.87')      ('1, Static, long', '0.77')
('2, Static, typos', '0.85')      ('1, Static, long', '0.76')       ('1, Static, long', '0.86')       ('2, Kinetic, typos', '0.76')
('1, Static, normal', '0.85')     ('2, Static, typos', '0.74')      ('1, Good, typos', '0.86')        ('2, Static, typos', '0.75')
('2, Good, typos', '0.84')        ('2, Good, typos', '0.74')        ('1, Kinetic, long', '0.85')      ('2, Kinetic, synonyms', '0.75')
('2, Kinetic, typos', '0.83')     ('1, Kinetic, long', '0.73')      ('1, Kinetic, normal', '0.85')    ('2, Kinetic, long', '0.73')
('1, Good, typos', '0.83')        ('1, Kinetic, normal', '0.73')    ('1, Kinetic, synonyms', '0.84')  ('1, Good, typos', '0.73')
('1, Kinetic, synonyms', '0.80')  ('1, Kinetic, synonyms', '0.72')  ('2, Good, typos', '0.84')        ('1, Kinetic, long', '0.71')
('1, Kinetic, normal', '0.80')    ('1, Good, typos', '0.71')        ('1, Kinetic, short', '0.83')     ('1, Kinetic, normal', '0.69')
('1, Kinetic, short', '0.79')     ('2, Kinetic, typos', '0.67')     ('1, Kinetic, typos', '0.81')     ('1, Static, typos', '0.69')
('1, Static, typos', '0.79')      ('1, Static, typos', '0.66')      ('2, Static, typos', '0.76')      ('1, Kinetic, synonyms', '0.69')
('1, Kinetic, typos', '0.78')     ('1, Kinetic, typos', '0.62')     ('2, Kinetic, typos', '0.72')     ('1, Kinetic, typos', '0.66')
('2, irrelevant', '0.69')         ('1, irrelevant', '0.58')         ('1, irrelevant', '0.67')         ('1, irrelevant', '0.33')
('1, irrelevant', '0.67')         ('2, irrelevant', '0.50')         ('2, irrelevant', '0.63')         ('2, irrelevant', '0.17')
```

Figure .20: The results from comparison to answer labeled "2, Good, short".



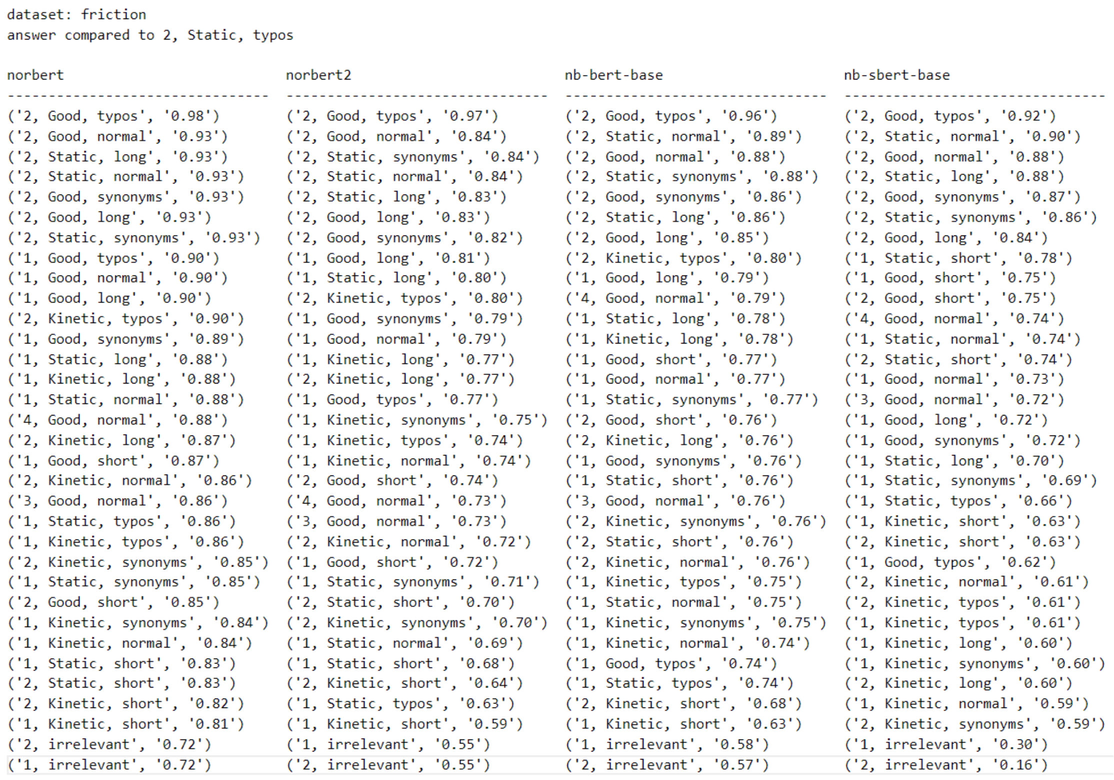
```
dataset: friction
answer compared to 2, Good, long

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('2, Static, long', '0.99')       ('2, Static, long', '0.99')       ('2, Static, long', '0.98')       ('2, Static, long', '0.96')
('2, Good, normal', '0.97')       ('2, Good, normal', '0.96')       ('2, Good, normal', '0.97')       ('2, Good, normal', '0.94')
('2, Static, normal', '0.97')     ('2, Good, synonyms', '0.95')     ('2, Good, synonyms', '0.96')     ('2, Static, normal', '0.92')
('2, Good, synonyms', '0.97')     ('2, Static, normal', '0.94')     ('2, Kinetic, long', '0.96')      ('2, Good, typos', '0.92')
('2, Kinetic, long', '0.96')      ('2, Static, synonyms', '0.94')   ('2, Static, normal', '0.95')     ('2, Good, synonyms', '0.92')
('2, Static, synonyms', '0.96')   ('2, Kinetic, long', '0.93')      ('2, Static, synonyms', '0.94')   ('2, Static, synonyms', '0.90')
('1, Good, long', '0.95')         ('1, Good, synonyms', '0.91')     ('1, Good, long', '0.94')         ('3, Good, normal', '0.88')
('1, Good, synonyms', '0.95')     ('1, Good, long', '0.91')         ('2, Kinetic, normal', '0.93')    ('4, Good, normal', '0.88')
('1, Good, normal', '0.95')       ('1, Good, normal', '0.90')       ('1, Good, normal', '0.92')       ('2, Good, short', '0.87')
('2, Kinetic, normal', '0.94')    ('1, Static, long', '0.88')       ('4, Good, normal', '0.92')       ('1, Static, short', '0.87')
('1, Static, long', '0.93')       ('1, Kinetic, long', '0.88')      ('3, Good, normal', '0.92')       ('1, Good, normal', '0.87')
('2, Static, typos', '0.93')      ('2, Kinetic, normal', '0.87')    ('1, Kinetic, long', '0.91')      ('1, Good, long', '0.87')
('2, Kinetic, synonyms', '0.93')  ('3, Good, normal', '0.85')       ('2, Good, short', '0.91')        ('2, Kinetic, long', '0.86')
('2, Good, typos', '0.93')        ('1, Kinetic, synonyms', '0.85')  ('1, Static, long', '0.91')       ('1, Good, short', '0.86')
('2, Good, short', '0.92')        ('2, Kinetic, synonyms', '0.85')  ('2, Kinetic, synonyms', '0.91')  ('1, Good, synonyms', '0.85')
('3, Good, normal', '0.92')       ('2, Good, short', '0.84')        ('2, Good, typos', '0.90')        ('2, Kinetic, normal', '0.84')
('1, Kinetic, long', '0.92')      ('1, Kinetic, normal', '0.84')    ('1, Good, synonyms', '0.90')     ('2, Static, typos', '0.84')
('1, Good, short', '0.91')        ('2, Good, typos', '0.83')        ('1, Kinetic, normal', '0.88')    ('1, Static, normal', '0.81')
('1, Good, typos', '0.91')        ('4, Good, normal', '0.83')       ('1, Kinetic, synonyms', '0.88')  ('1, Static, long', '0.81')
('4, Good, normal', '0.90')       ('2, Static, typos', '0.83')      ('1, Static, normal', '0.87')     ('2, Static, short', '0.80')
('1, Static, synonyms', '0.90')   ('1, Good, short', '0.82')        ('1, Good, short', '0.87')        ('2, Kinetic, typos', '0.79')
('1, Static, normal', '0.89')     ('1, Good, typos', '0.80')        ('1, Static, synonyms', '0.86')   ('2, Kinetic, synonyms', '0.76')
('1, Kinetic, synonyms', '0.89')  ('2, Static, short', '0.78')      ('1, Good, typos', '0.85')        ('1, Static, synonyms', '0.76')
('2, Kinetic, typos', '0.89')     ('1, Static, synonyms', '0.78')   ('1, Static, short', '0.85')      ('1, Kinetic, long', '0.76')
('2, Static, short', '0.89')      ('1, Static, short', '0.76')      ('2, Static, typos', '0.85')      ('1, Kinetic, short', '0.76')
('1, Kinetic, normal', '0.88')    ('1, Kinetic, typos', '0.75')     ('2, Static, short', '0.84')      ('2, Kinetic, short', '0.75')
('1, Kinetic, typos', '0.87')     ('1, Static, normal', '0.74')     ('1, Kinetic, typos', '0.84')     ('1, Good, typos', '0.71')
('1, Static, short', '0.87')      ('2, Kinetic, typos', '0.74')     ('1, Static, typos', '0.82')      ('1, Kinetic, normal', '0.70')
('2, Kinetic, short', '0.87')     ('2, Kinetic, short', '0.71')     ('2, Kinetic, short', '0.80')     ('1, Kinetic, synonyms', '0.69')
('1, Static, typos', '0.85')      ('1, Kinetic, short', '0.66')     ('1, Kinetic, short', '0.77')     ('1, Kinetic, typos', '0.68')
('1, Kinetic, short', '0.83')     ('1, Static, typos', '0.65')      ('2, Kinetic, typos', '0.77')     ('1, Static, typos', '0.67')
('2, irrelevant', '0.77')         ('1, irrelevant', '0.63')         ('1, irrelevant', '0.69')         ('1, irrelevant', '0.29')
('1, irrelevant', '0.75')         ('2, irrelevant', '0.62')         ('2, irrelevant', '0.64')         ('2, irrelevant', '0.21')
```

Figure .21: The results from comparison to answer labeled "2, Good, long".



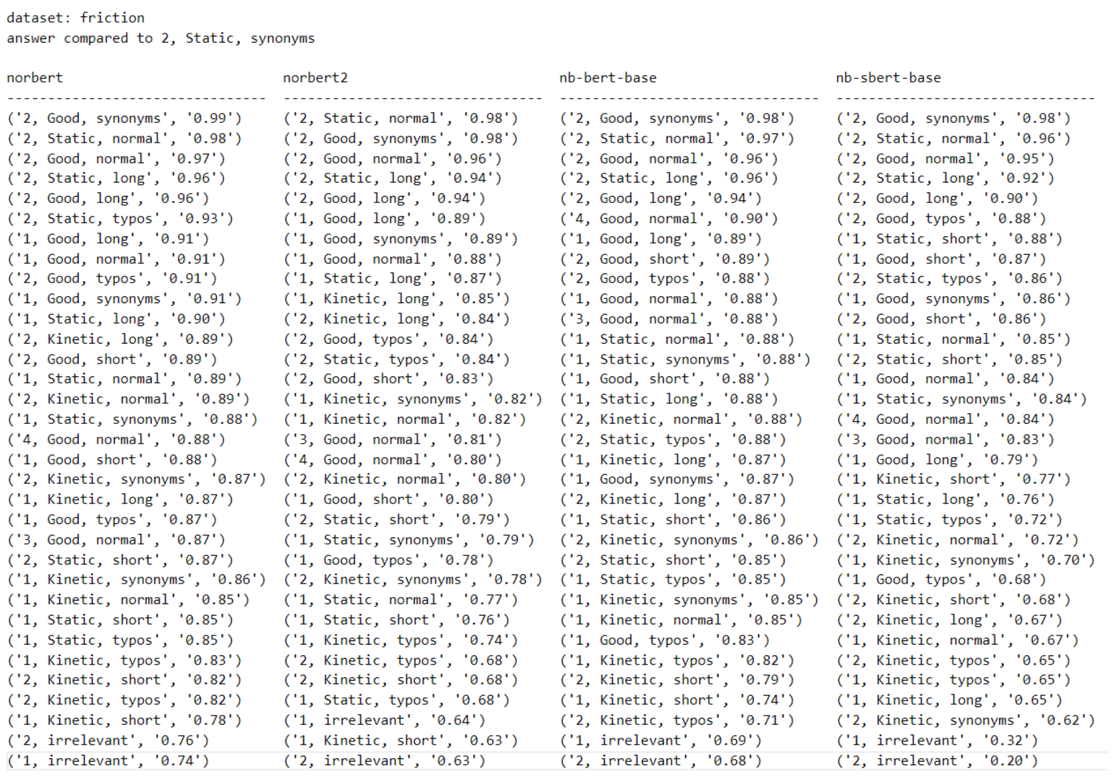
```
dataset: friction
answer compared to 2, Static, normal

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
---------------------------------  ---------------------------------  ---------------------------------  ---------------------------------
('2, Good, normal', '0.99')        ('2, Static, synonyms', '0.98')    ('2, Good, normal', '0.98')        ('2, Good, normal', '0.98')
('2, Static, synonyms', '0.98')    ('2, Good, normal', '0.98')        ('2, Static, synonyms', '0.97')    ('2, Static, synonyms', '0.96')
('2, Good, synonyms', '0.98')      ('2, Good, synonyms', '0.96')      ('2, Good, synonyms', '0.97')      ('2, Static, long', '0.95')
('2, Static, long', '0.97')        ('2, Static, long', '0.95')        ('2, Static, long', '0.97')        ('2, Good, synonyms', '0.95')
('2, Good, long', '0.97')          ('2, Good, long', '0.94')          ('2, Good, long', '0.95')          ('2, Good, long', '0.92')
('2, Static, typos', '0.93')       ('1, Good, synonyms', '0.89')      ('4, Good, normal', '0.91')        ('2, Static, typos', '0.90')
('1, Good, synonyms', '0.92')      ('1, Good, long', '0.88')          ('2, Good, short', '0.90')         ('2, Good, typos', '0.90')
('1, Good, long', '0.92')          ('1, Good, normal', '0.88')        ('1, Good, long', '0.90')          ('2, Good, short', '0.87')
('2, Good, short', '0.91')         ('1, Static, long', '0.87')        ('2, Good, typos', '0.89')         ('1, Static, short', '0.87')
('1, Good, normal', '0.91')        ('1, Kinetic, long', '0.85')       ('1, Good, short', '0.89')         ('2, Static, short', '0.87')
('2, Good, typos', '0.91')         ('2, Good, short', '0.84')         ('2, Static, typos', '0.89')       ('1, Good, short', '0.86')
('2, Kinetic, long', '0.90')       ('2, Kinetic, long', '0.84')       ('1, Static, normal', '0.88')      ('1, Static, normal', '0.85')
('1, Static, synonyms', '0.90')    ('2, Good, typos', '0.84')         ('1, Static, synonyms', '0.88')    ('1, Good, synonyms', '0.84')
('2, Kinetic, normal', '0.90')     ('2, Static, typos', '0.84')       ('1, Static, long', '0.88')        ('4, Good, normal', '0.84')
('1, Static, long', '0.90')        ('1, Kinetic, synonyms', '0.82')   ('2, Kinetic, normal', '0.88')     ('1, Good, normal', '0.84')
('2, Static, short', '0.89')       ('1, Kinetic, normal', '0.82')     ('1, Good, normal', '0.88')        ('3, Good, normal', '0.83')
('4, Good, normal', '0.89')        ('3, Good, normal', '0.81')        ('3, Good, normal', '0.88')        ('1, Static, synonyms', '0.83')
('3, Good, normal', '0.89')        ('2, Static, short', '0.81')       ('2, Static, short', '0.88')       ('1, Good, long', '0.79')
('1, Good, short', '0.89')         ('1, Good, short', '0.81')         ('1, Static, short', '0.87')       ('1, Static, long', '0.76')
('1, Static, normal', '0.89')      ('4, Good, normal', '0.80')        ('1, Kinetic, long', '0.87')       ('1, Kinetic, short', '0.75')
('2, Kinetic, synonyms', '0.89')   ('2, Kinetic, normal', '0.79')     ('1, Good, synonyms', '0.87')      ('1, Static, typos', '0.72')
('1, Kinetic, long', '0.88')       ('1, Static, synonyms', '0.78')    ('2, Kinetic, long', '0.86')       ('2, Kinetic, normal', '0.71')
('1, Good, typos', '0.87')         ('2, Kinetic, synonyms', '0.77')   ('2, Kinetic, synonyms', '0.86')   ('2, Kinetic, short', '0.69')
('1, Static, short', '0.86')       ('1, Good, typos', '0.76')         ('1, Kinetic, normal', '0.85')     ('1, Good, typos', '0.68')
('1, Kinetic, synonyms', '0.86')   ('1, Static, short', '0.76')       ('1, Kinetic, synonyms', '0.85')   ('2, Kinetic, long', '0.68')
('1, Kinetic, normal', '0.85')     ('1, Static, normal', '0.74')      ('1, Static, typos', '0.84')       ('1, Kinetic, synonyms', '0.66')
('1, Static, typos', '0.85')       ('1, Kinetic, typos', '0.72')      ('1, Good, typos', '0.81')         ('2, Kinetic, typos', '0.65')
('1, Kinetic, typos', '0.84')      ('2, Kinetic, typos', '0.67')      ('1, Kinetic, typos', '0.81')      ('1, Kinetic, long', '0.65')
('2, Kinetic, short', '0.84')      ('2, Kinetic, short', '0.67')      ('2, Kinetic, short', '0.80')      ('1, Kinetic, normal', '0.64')
('2, Kinetic, typos', '0.83')      ('1, Static, typos', '0.65')       ('1, Kinetic, short', '0.75')      ('2, Kinetic, synonyms', '0.63')
('1, Kinetic, short', '0.80')      ('2, irrelevant', '0.62')          ('2, Kinetic, typos', '0.71')      ('1, Kinetic, typos', '0.62')
('2, irrelevant', '0.75')          ('1, Kinetic, short', '0.62')      ('1, irrelevant', '0.68')          ('1, irrelevant', '0.29')
('1, irrelevant', '0.74')          ('1, irrelevant', '0.61')          ('2, irrelevant', '0.64')          ('2, irrelevant', '0.17')
```

Figure .22: The results from comparison to answer labeled "2, Static, normal".



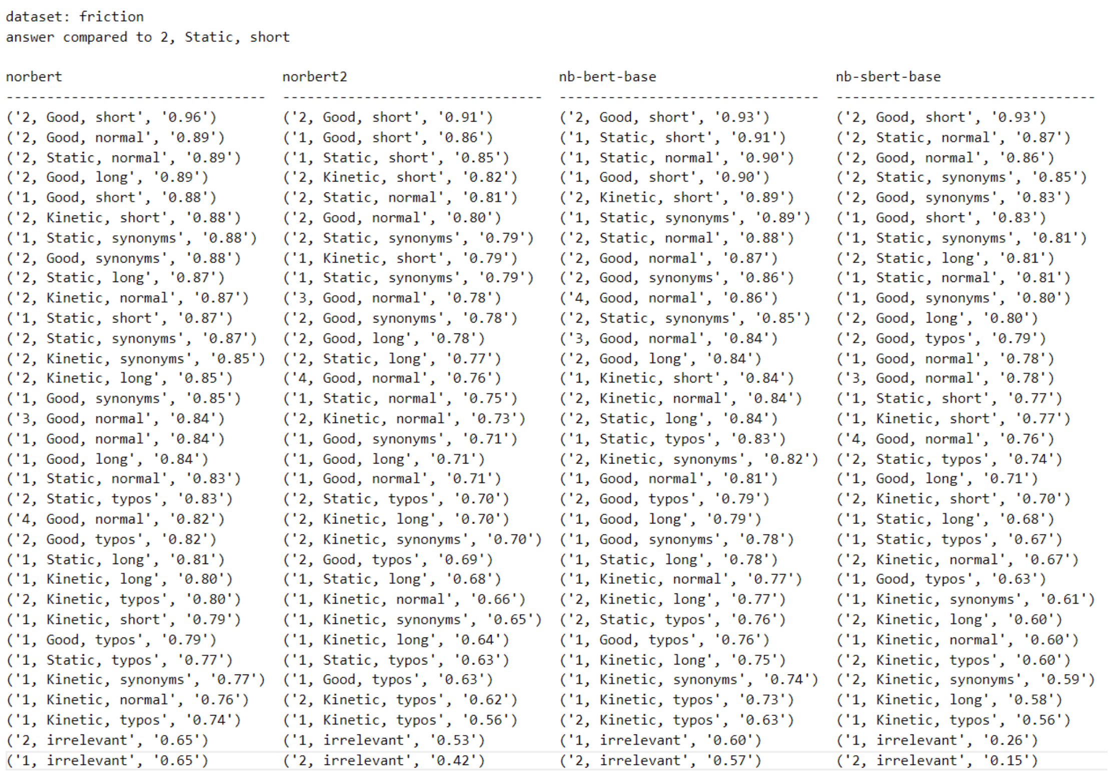
```
dataset: friction
answer compared to 2, Static, typos

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
---------------------------------  ---------------------------------  ---------------------------------  ---------------------------------
('2, Good, typos', '0.98')         ('2, Good, typos', '0.97')         ('2, Good, typos', '0.96')         ('2, Good, typos', '0.92')
('2, Good, normal', '0.93')        ('2, Good, normal', '0.84')        ('2, Static, normal', '0.89')      ('2, Static, normal', '0.90')
('2, Static, long', '0.93')        ('2, Static, synonyms', '0.84')    ('2, Good, normal', '0.88')        ('2, Good, normal', '0.88')
('2, Static, normal', '0.93')      ('2, Static, normal', '0.84')      ('2, Static, synonyms', '0.88')    ('2, Static, long', '0.88')
('2, Good, synonyms', '0.93')      ('2, Static, long', '0.83')        ('2, Good, synonyms', '0.86')      ('2, Good, synonyms', '0.87')
('2, Good, long', '0.93')          ('2, Good, long', '0.83')          ('2, Static, long', '0.86')        ('2, Static, synonyms', '0.86')
('2, Static, synonyms', '0.93')    ('2, Good, synonyms', '0.82')      ('2, Good, long', '0.85')          ('2, Good, long', '0.84')
('1, Good, typos', '0.90')         ('1, Good, long', '0.81')          ('2, Kinetic, typos', '0.80')      ('1, Static, short', '0.78')
('1, Good, normal', '0.90')        ('1, Static, long', '0.80')        ('1, Good, long', '0.79')          ('1, Good, short', '0.75')
('1, Good, long', '0.90')          ('2, Kinetic, typos', '0.80')      ('4, Good, normal', '0.79')        ('2, Good, short', '0.75')
('2, Kinetic, typos', '0.90')      ('1, Good, synonyms', '0.79')      ('1, Static, long', '0.78')        ('4, Good, normal', '0.74')
('1, Good, synonyms', '0.89')      ('1, Good, normal', '0.79')        ('1, Kinetic, long', '0.78')       ('1, Static, normal', '0.74')
('1, Static, long', '0.88')        ('1, Kinetic, long', '0.77')       ('1, Good, short', '0.77')         ('2, Static, short', '0.74')
('1, Kinetic, long', '0.88')       ('2, Kinetic, long', '0.77')       ('1, Good, normal', '0.77')        ('1, Good, normal', '0.73')
('1, Static, normal', '0.88')      ('1, Good, typos', '0.77')         ('1, Static, synonyms', '0.77')    ('3, Good, normal', '0.72')
('4, Good, normal', '0.88')        ('1, Kinetic, synonyms', '0.75')   ('2, Good, short', '0.76')         ('1, Good, long', '0.72')
('2, Kinetic, long', '0.87')       ('1, Kinetic, typos', '0.74')      ('2, Kinetic, long', '0.76')       ('1, Good, synonyms', '0.72')
('1, Good, short', '0.87')         ('1, Kinetic, normal', '0.74')     ('1, Good, synonyms', '0.76')      ('1, Static, long', '0.70')
('2, Kinetic, normal', '0.86')     ('2, Good, short', '0.74')         ('1, Static, short', '0.76')       ('1, Static, synonyms', '0.69')
('3, Good, normal', '0.86')        ('4, Good, normal', '0.73')        ('3, Good, normal', '0.76')        ('1, Static, typos', '0.66')
('1, Static, typos', '0.86')       ('3, Good, normal', '0.73')        ('2, Kinetic, synonyms', '0.76')   ('1, Kinetic, short', '0.63')
('1, Kinetic, typos', '0.86')      ('2, Kinetic, normal', '0.72')     ('2, Static, short', '0.76')       ('2, Kinetic, short', '0.63')
('2, Kinetic, synonyms', '0.85')   ('1, Good, short', '0.72')         ('2, Kinetic, normal', '0.76')     ('1, Good, typos', '0.62')
('1, Static, synonyms', '0.85')    ('1, Static, synonyms', '0.71')    ('1, Kinetic, typos', '0.75')      ('2, Kinetic, normal', '0.61')
('2, Good, short', '0.85')         ('2, Static, short', '0.70')       ('1, Static, normal', '0.75')      ('2, Kinetic, typos', '0.61')
('1, Kinetic, synonyms', '0.84')   ('2, Kinetic, synonyms', '0.70')   ('1, Kinetic, synonyms', '0.75')   ('1, Kinetic, typos', '0.61')
('1, Kinetic, normal', '0.84')     ('1, Static, normal', '0.69')      ('1, Kinetic, normal', '0.74')     ('1, Kinetic, long', '0.60')
('1, Static, short', '0.83')       ('1, Static, short', '0.68')       ('1, Good, typos', '0.74')         ('1, Kinetic, synonyms', '0.60')
('2, Static, short', '0.83')       ('2, Kinetic, short', '0.64')      ('1, Static, typos', '0.74')       ('2, Kinetic, long', '0.60')
('2, Kinetic, short', '0.82')      ('1, Static, typos', '0.63')       ('2, Kinetic, short', '0.68')      ('1, Kinetic, normal', '0.59')
('1, Kinetic, short', '0.81')      ('1, Kinetic, short', '0.59')      ('1, Kinetic, short', '0.63')      ('2, Kinetic, synonyms', '0.59')
('2, irrelevant', '0.72')          ('1, irrelevant', '0.55')          ('1, irrelevant', '0.58')          ('1, irrelevant', '0.30')
('1, irrelevant', '0.72')          ('2, irrelevant', '0.55')          ('2, irrelevant', '0.57')          ('2, irrelevant', '0.16')
```

Figure .23: The results from comparison to answer labeled "2, Static, typos".



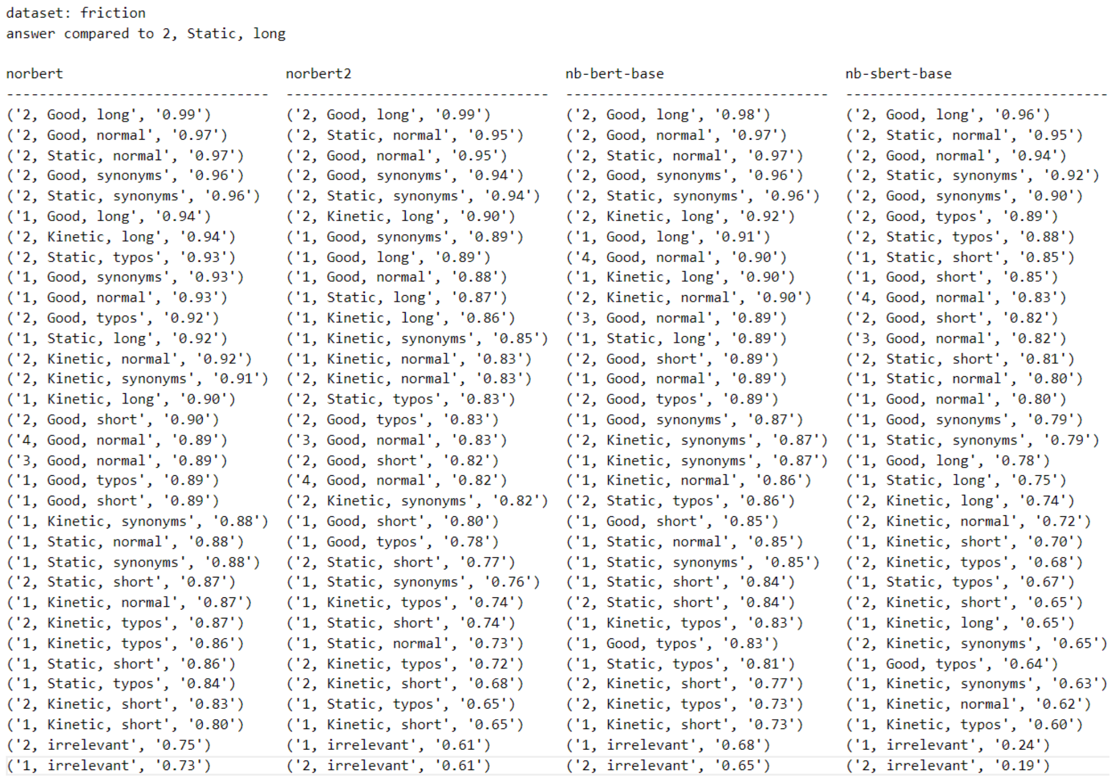
```
dataset: friction
answer compared to 2, Static, synonyms

norbert                           norbert2                          nb-bert-base                      nb-sbert-base

--------------------------------  --------------------------------  --------------------------------  --------------------------------
('2, Good, synonyms', '0.99')     ('2, Static, normal', '0.98')     ('2, Good, synonyms', '0.98')     ('2, Good, synonyms', '0.98')
('2, Static, normal', '0.98')     ('2, Good, synonyms', '0.98')     ('2, Static, normal', '0.97')     ('2, Static, normal', '0.96')
('2, Good, normal', '0.97')       ('2, Good, normal', '0.96')       ('2, Good, normal', '0.96')       ('2, Good, normal', '0.95')
('2, Static, long', '0.96')       ('2, Static, long', '0.94')       ('2, Static, long', '0.96')       ('2, Static, long', '0.92')
('2, Good, long', '0.96')         ('2, Good, long', '0.94')         ('2, Good, long', '0.94')         ('2, Good, long', '0.90')
('2, Static, typos', '0.93')      ('1, Good, long', '0.89')         ('4, Good, normal', '0.90')       ('2, Good, typos', '0.88')
('1, Good, long', '0.91')         ('1, Good, synonyms', '0.89')     ('1, Good, long', '0.89')         ('1, Static, short', '0.88')
('1, Good, normal', '0.91')       ('1, Good, normal', '0.88')       ('2, Good, short', '0.89')        ('1, Good, short', '0.87')
('2, Good, typos', '0.91')        ('1, Static, long', '0.87')       ('2, Good, typos', '0.88')        ('2, Static, typos', '0.86')
('1, Good, synonyms', '0.91')     ('1, Kinetic, long', '0.85')      ('1, Good, normal', '0.88')       ('1, Good, synonyms', '0.86')
('1, Static, long', '0.90')       ('2, Kinetic, long', '0.84')      ('3, Good, normal', '0.88')       ('2, Good, short', '0.86')
('2, Kinetic, long', '0.89')      ('2, Good, typos', '0.84')        ('1, Static, normal', '0.88')     ('1, Static, normal', '0.85')
('2, Good, short', '0.89')        ('2, Static, typos', '0.84')      ('1, Static, synonyms', '0.88')   ('2, Static, short', '0.85')
('1, Static, normal', '0.89')     ('2, Good, short', '0.83')        ('1, Good, short', '0.88')        ('1, Good, normal', '0.84')
('2, Kinetic, normal', '0.89')    ('1, Kinetic, synonyms', '0.82')  ('1, Static, long', '0.88')       ('1, Static, synonyms', '0.84')
('1, Static, synonyms', '0.88')   ('1, Kinetic, normal', '0.82')    ('2, Kinetic, normal', '0.88')    ('4, Good, normal', '0.84')
('4, Good, normal', '0.88')       ('3, Good, normal', '0.81')       ('2, Static, typos', '0.88')      ('3, Good, normal', '0.83')
('1, Good, short', '0.88')        ('4, Good, normal', '0.80')       ('1, Kinetic, long', '0.87')      ('1, Good, long', '0.79')
('2, Kinetic, synonyms', '0.87')  ('2, Kinetic, normal', '0.80')    ('1, Good, synonyms', '0.87')     ('1, Kinetic, short', '0.77')
('1, Kinetic, long', '0.87')      ('1, Good, short', '0.80')        ('2, Kinetic, long', '0.87')      ('1, Static, long', '0.76')
('1, Good, typos', '0.87')        ('2, Static, short', '0.79')      ('1, Static, short', '0.86')      ('1, Static, typos', '0.72')
('3, Good, normal', '0.87')       ('1, Static, synonyms', '0.79')   ('2, Kinetic, synonyms', '0.86')  ('2, Kinetic, normal', '0.72')
('2, Static, short', '0.87')      ('1, Good, typos', '0.78')        ('2, Static, short', '0.85')      ('1, Kinetic, synonyms', '0.70')
('1, Kinetic, synonyms', '0.86')  ('2, Kinetic, synonyms', '0.78')  ('1, Static, typos', '0.85')      ('1, Good, typos', '0.68')
('1, Kinetic, normal', '0.85')    ('1, Static, normal', '0.77')     ('1, Kinetic, synonyms', '0.85')  ('2, Kinetic, short', '0.68')
('1, Static, short', '0.85')      ('1, Static, short', '0.76')      ('1, Kinetic, normal', '0.85')    ('2, Kinetic, long', '0.67')
('1, Static, typos', '0.85')      ('1, Kinetic, typos', '0.74')     ('1, Good, typos', '0.83')        ('1, Kinetic, normal', '0.67')
('1, Kinetic, typos', '0.83')     ('2, Kinetic, typos', '0.68')     ('1, Kinetic, typos', '0.82')     ('2, Kinetic, typos', '0.65')
('2, Kinetic, short', '0.82')     ('2, Kinetic, short', '0.68')     ('2, Kinetic, short', '0.79')     ('1, Kinetic, typos', '0.65')
('2, Kinetic, typos', '0.82')     ('1, Static, typos', '0.68')      ('1, Kinetic, short', '0.74')     ('1, Kinetic, long', '0.65')
('1, Kinetic, short', '0.78')     ('1, irrelevant', '0.64')         ('2, Kinetic, typos', '0.71')     ('2, Kinetic, synonyms', '0.62')
('2, irrelevant', '0.76')         ('1, Kinetic, short', '0.63')     ('1, irrelevant', '0.69')         ('1, irrelevant', '0.32')
('1, irrelevant', '0.74')         ('2, irrelevant', '0.63')         ('2, irrelevant', '0.68')         ('2, irrelevant', '0.20')
```

Figure .24: The results from comparison to answer labeled "2, Static, synonyms".



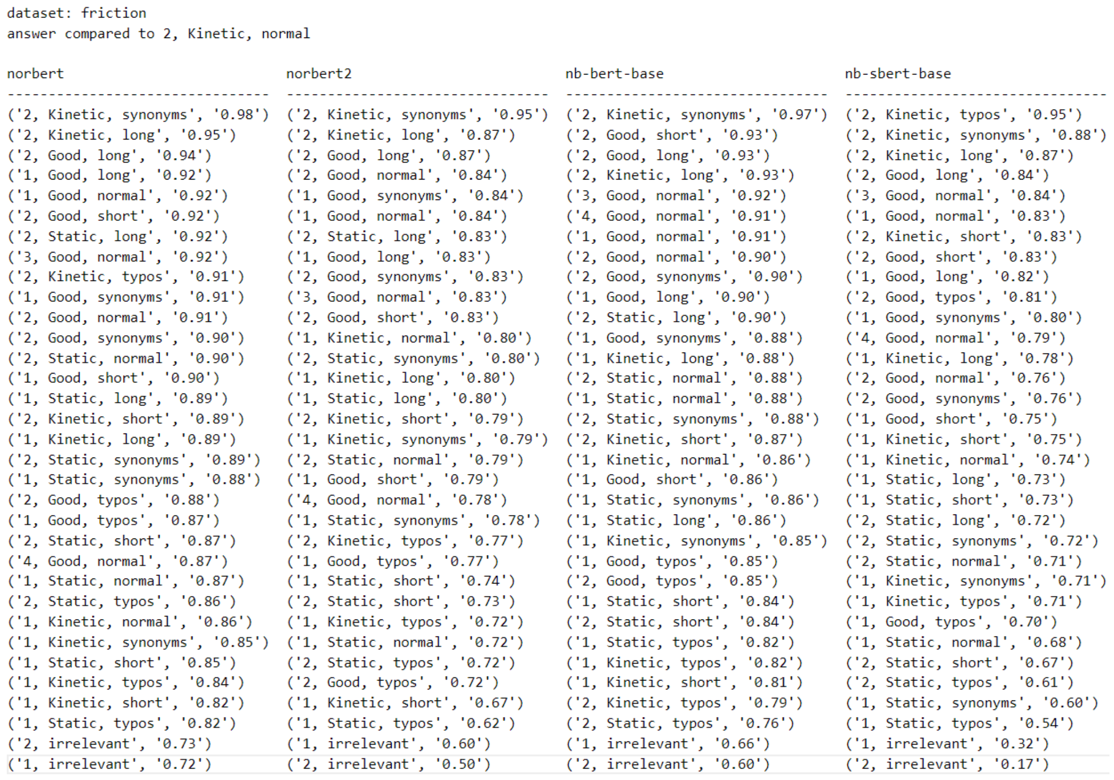
```
dataset: friction
answer compared to 2, Static, short

norbert                           norbert2                          nb-bert-base                      nb-sbert-base

--------------------------------  --------------------------------  --------------------------------  --------------------------------
('2, Good, short', '0.96')        ('2, Good, short', '0.91')        ('2, Good, short', '0.93')        ('2, Good, short', '0.93')
('2, Good, normal', '0.89')       ('1, Good, short', '0.86')        ('1, Static, short', '0.91')      ('2, Static, normal', '0.87')
('2, Static, normal', '0.89')     ('1, Static, short', '0.85')      ('1, Static, normal', '0.90')     ('2, Good, normal', '0.86')
('2, Good, long', '0.89')         ('2, Kinetic, short', '0.82')     ('1, Good, short', '0.90')        ('2, Static, synonyms', '0.85')
('1, Good, short', '0.88')        ('2, Static, normal', '0.81')     ('2, Kinetic, short', '0.89')     ('2, Good, synonyms', '0.83')
('2, Kinetic, short', '0.88')     ('2, Good, normal', '0.80')       ('1, Static, synonyms', '0.89')   ('1, Good, short', '0.83')
('1, Static, synonyms', '0.88')   ('2, Static, synonyms', '0.79')   ('2, Static, normal', '0.88')     ('1, Static, synonyms', '0.81')
('2, Good, synonyms', '0.88')     ('1, Kinetic, short', '0.79')     ('2, Good, normal', '0.87')       ('2, Static, long', '0.81')
('2, Static, long', '0.87')       ('1, Static, synonyms', '0.79')   ('2, Good, synonyms', '0.86')     ('1, Static, normal', '0.81')
('2, Kinetic, normal', '0.87')    ('3, Good, normal', '0.78')       ('4, Good, normal', '0.86')       ('1, Good, synonyms', '0.80')
('1, Static, short', '0.87')      ('2, Good, synonyms', '0.78')     ('2, Static, synonyms', '0.85')   ('2, Good, long', '0.80')
('2, Static, synonyms', '0.87')   ('2, Good, long', '0.78')         ('3, Good, normal', '0.84')       ('2, Good, typos', '0.79')
('2, Kinetic, synonyms', '0.85')  ('2, Static, long', '0.77')       ('2, Good, long', '0.84')         ('1, Good, normal', '0.78')
('2, Kinetic, long', '0.85')      ('4, Good, normal', '0.76')       ('1, Kinetic, short', '0.84')     ('3, Good, normal', '0.78')
('1, Good, synonyms', '0.85')     ('1, Static, normal', '0.75')     ('2, Kinetic, normal', '0.84')    ('1, Static, short', '0.77')
('3, Good, normal', '0.84')       ('2, Kinetic, normal', '0.73')    ('2, Static, long', '0.84')       ('1, Kinetic, short', '0.77')
('1, Good, normal', '0.84')       ('1, Good, synonyms', '0.71')     ('1, Static, typos', '0.83')      ('4, Good, normal', '0.76')
('1, Good, long', '0.84')         ('1, Good, long', '0.71')         ('2, Kinetic, synonyms', '0.82')  ('2, Static, typos', '0.74')
('1, Static, normal', '0.83')     ('1, Good, normal', '0.71')       ('1, Good, normal', '0.81')       ('1, Good, long', '0.71')
('2, Static, typos', '0.83')      ('2, Static, typos', '0.70')      ('2, Good, typos', '0.79')        ('2, Kinetic, short', '0.70')
('4, Good, normal', '0.82')       ('2, Kinetic, long', '0.70')      ('1, Good, long', '0.79')         ('1, Static, long', '0.68')
('2, Good, typos', '0.82')        ('2, Kinetic, synonyms', '0.70')  ('1, Good, synonyms', '0.78')     ('1, Static, typos', '0.67')
('1, Static, long', '0.81')       ('2, Good, typos', '0.69')        ('1, Static, long', '0.78')       ('2, Kinetic, normal', '0.67')
('1, Kinetic, long', '0.80')      ('1, Static, long', '0.68')       ('1, Kinetic, normal', '0.77')    ('1, Good, typos', '0.63')
('2, Kinetic, typos', '0.80')     ('1, Kinetic, normal', '0.66')    ('2, Kinetic, long', '0.77')      ('1, Kinetic, synonyms', '0.61')
('1, Kinetic, short', '0.79')     ('1, Kinetic, synonyms', '0.65')  ('2, Static, typos', '0.76')      ('2, Kinetic, long', '0.60')
('1, Good, typos', '0.79')        ('1, Kinetic, long', '0.64')      ('1, Good, typos', '0.76')        ('1, Kinetic, normal', '0.60')
('1, Static, typos', '0.77')      ('1, Static, typos', '0.63')      ('1, Kinetic, long', '0.75')      ('2, Kinetic, typos', '0.60')
('1, Kinetic, synonyms', '0.77')  ('1, Good, typos', '0.63')        ('1, Kinetic, synonyms', '0.74')  ('2, Kinetic, synonyms', '0.59')
('1, Kinetic, normal', '0.76')    ('2, Kinetic, typos', '0.62')     ('1, Kinetic, typos', '0.73')     ('1, Kinetic, long', '0.58')
('1, Kinetic, typos', '0.74')     ('1, Kinetic, typos', '0.56')     ('2, Kinetic, typos', '0.63')     ('1, Kinetic, typos', '0.56')
('2, irrelevant', '0.65')         ('1, irrelevant', '0.53')         ('1, irrelevant', '0.60')         ('1, irrelevant', '0.26')
('1, irrelevant', '0.65')         ('2, irrelevant', '0.42')         ('2, irrelevant', '0.57')         ('2, irrelevant', '0.15')
```

Figure .25: The results from comparison to answer labeled "2, Static, short".



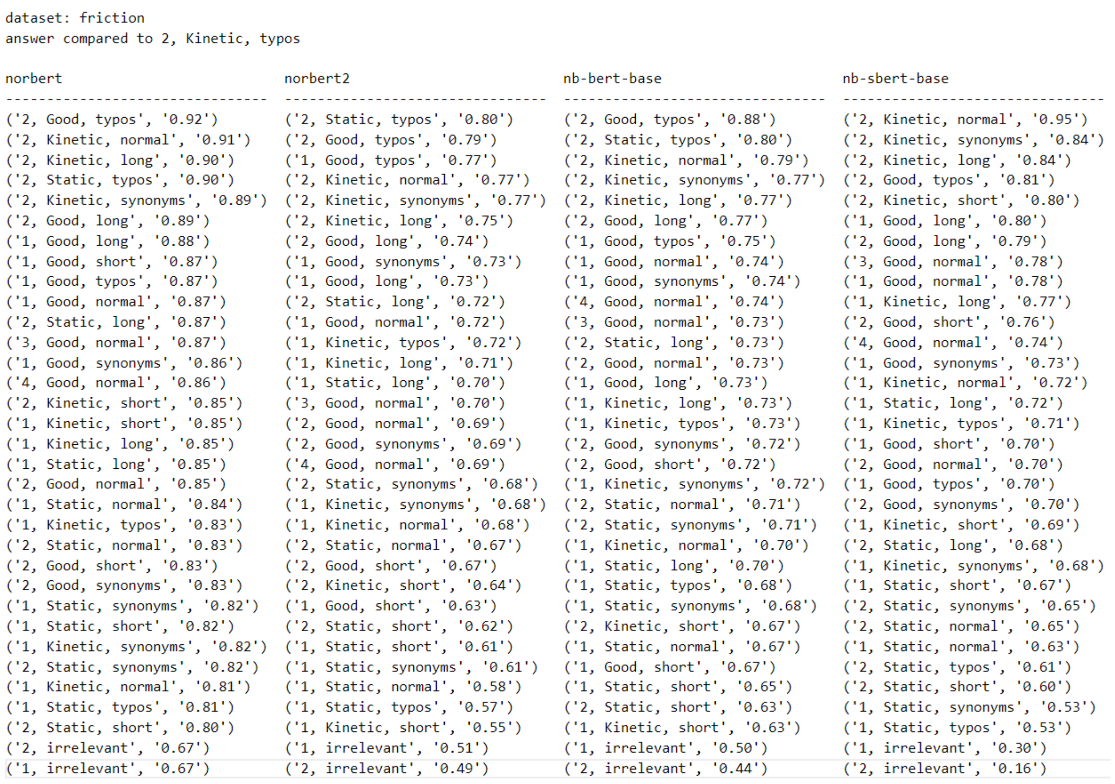
```
dataset: friction
answer compared to 2, Static, long

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('2, Good, long', '0.99')         ('2, Good, long', '0.99')         ('2, Good, long', '0.98')         ('2, Good, long', '0.96')
('2, Good, normal', '0.97')       ('2, Static, normal', '0.95')     ('2, Good, normal', '0.97')       ('2, Static, normal', '0.95')
('2, Static, normal', '0.97')     ('2, Good, normal', '0.95')       ('2, Static, normal', '0.97')     ('2, Good, normal', '0.94')
('2, Good, synonyms', '0.96')     ('2, Good, synonyms', '0.94')     ('2, Good, synonyms', '0.96')     ('2, Static, synonyms', '0.92')
('2, Static, synonyms', '0.96')   ('2, Static, synonyms', '0.94')   ('2, Static, synonyms', '0.96')   ('2, Good, synonyms', '0.90')
('1, Good, long', '0.94')         ('2, Kinetic, long', '0.90')      ('2, Kinetic, long', '0.92')      ('2, Good, typos', '0.89')
('2, Kinetic, long', '0.94')      ('1, Good, synonyms', '0.89')     ('1, Good, long', '0.91')         ('2, Static, typos', '0.88')
('2, Static, typos', '0.93')      ('1, Good, long', '0.89')         ('4, Good, normal', '0.90')       ('1, Static, short', '0.85')
('1, Good, synonyms', '0.93')     ('1, Good, normal', '0.88')       ('1, Kinetic, long', '0.90')      ('1, Good, short', '0.85')
('1, Good, normal', '0.93')       ('1, Static, long', '0.87')       ('2, Kinetic, normal', '0.90')    ('4, Good, normal', '0.83')
('2, Good, typos', '0.92')        ('1, Kinetic, long', '0.86')      ('3, Good, normal', '0.89')       ('2, Good, short', '0.82')
('1, Static, long', '0.92')       ('1, Kinetic, synonyms', '0.85')  ('1, Static, long', '0.89')       ('3, Good, normal', '0.82')
('2, Kinetic, normal', '0.92')    ('1, Kinetic, normal', '0.83')    ('2, Good, short', '0.89')        ('2, Static, short', '0.81')
('2, Kinetic, synonyms', '0.91')  ('2, Kinetic, normal', '0.83')    ('1, Good, normal', '0.89')       ('1, Static, normal', '0.80')
('1, Kinetic, long', '0.90')      ('2, Static, typos', '0.83')      ('2, Good, typos', '0.89')        ('1, Good, normal', '0.80')
('2, Good, short', '0.90')        ('2, Good, typos', '0.83')        ('1, Good, synonyms', '0.87')     ('1, Good, synonyms', '0.79')
('4, Good, normal', '0.89')       ('3, Good, normal', '0.83')       ('2, Kinetic, synonyms', '0.87')  ('1, Static, synonyms', '0.79')
('3, Good, normal', '0.89')       ('2, Good, short', '0.82')        ('1, Kinetic, synonyms', '0.87')  ('1, Good, long', '0.78')
('1, Good, typos', '0.89')        ('4, Good, normal', '0.82')       ('1, Kinetic, normal', '0.86')    ('1, Static, long', '0.75')
('1, Good, short', '0.89')        ('2, Kinetic, synonyms', '0.82')  ('2, Static, typos', '0.86')      ('2, Kinetic, long', '0.74')
('1, Kinetic, synonyms', '0.88')  ('1, Good, short', '0.80')        ('1, Good, short', '0.85')        ('2, Kinetic, normal', '0.72')
('1, Static, normal', '0.88')     ('1, Good, typos', '0.78')        ('1, Static, normal', '0.85')     ('1, Kinetic, short', '0.70')
('1, Static, synonyms', '0.88')   ('2, Static, short', '0.77')      ('1, Static, synonyms', '0.85')   ('2, Kinetic, typos', '0.68')
('2, Static, short', '0.87')      ('1, Static, synonyms', '0.76')   ('1, Static, short', '0.84')      ('1, Static, typos', '0.67')
('1, Kinetic, normal', '0.87')    ('1, Kinetic, typos', '0.74')     ('2, Static, short', '0.84')      ('2, Kinetic, short', '0.65')
('2, Kinetic, typos', '0.87')     ('1, Static, short', '0.74')      ('1, Kinetic, typos', '0.83')     ('1, Kinetic, long', '0.65')
('1, Kinetic, typos', '0.86')     ('1, Static, normal', '0.73')     ('1, Good, typos', '0.83')        ('2, Kinetic, synonyms', '0.65')
('1, Static, short', '0.86')      ('2, Kinetic, typos', '0.72')     ('1, Static, typos', '0.81')      ('1, Good, typos', '0.64')
('1, Static, typos', '0.84')      ('2, Kinetic, short', '0.68')     ('2, Kinetic, short', '0.77')     ('1, Kinetic, synonyms', '0.63')
('2, Kinetic, short', '0.83')     ('1, Static, typos', '0.65')      ('2, Kinetic, typos', '0.73')     ('1, Kinetic, normal', '0.62')
('1, Kinetic, short', '0.80')     ('1, Kinetic, short', '0.65')     ('1, Kinetic, short', '0.73')     ('1, Kinetic, typos', '0.60')
('2, irrelevant', '0.75')         ('1, irrelevant', '0.61')         ('1, irrelevant', '0.68')         ('1, irrelevant', '0.24')
('1, irrelevant', '0.73')         ('2, irrelevant', '0.61')         ('2, irrelevant', '0.65')         ('2, irrelevant', '0.19')
```

Figure .26: The results from comparison to answer labeled "2, Static, long".



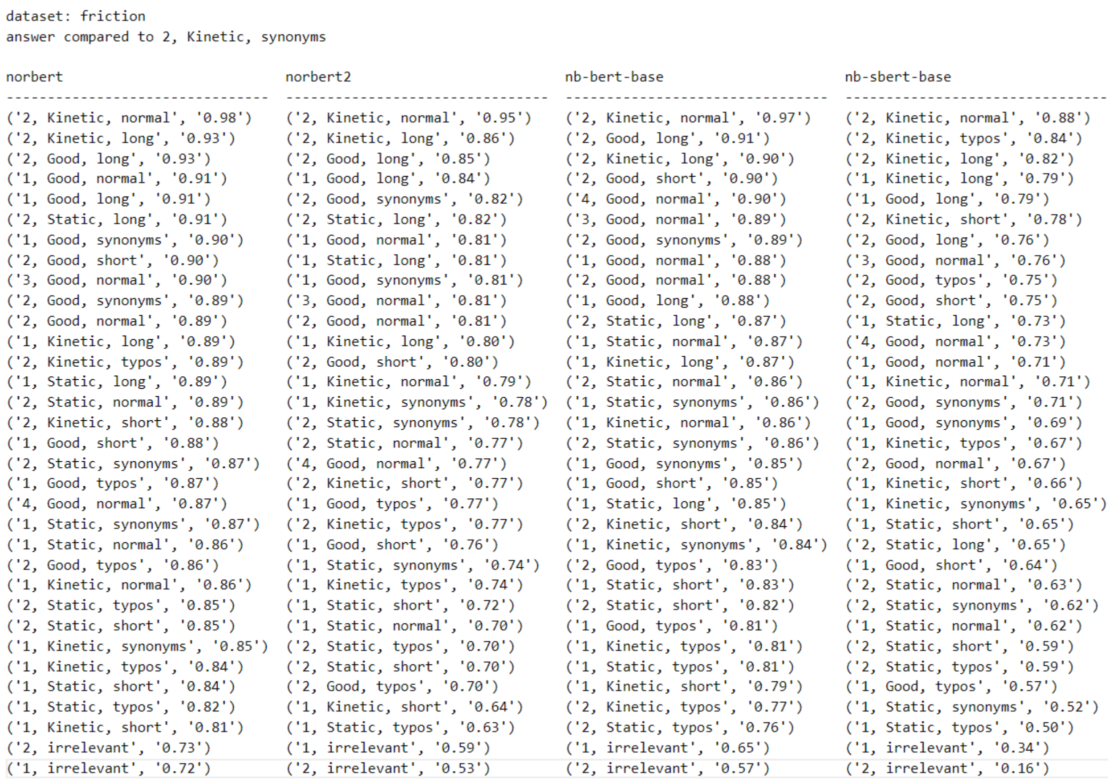
```
dataset: friction
answer compared to 2, Kinetic, normal

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('2, Kinetic, synonyms', '0.98')  ('2, Kinetic, synonyms', '0.95')  ('2, Kinetic, synonyms', '0.97')  ('2, Kinetic, typos', '0.95')
('2, Kinetic, long', '0.95')      ('2, Kinetic, long', '0.87')      ('2, Good, short', '0.93')        ('2, Kinetic, synonyms', '0.88')
('2, Good, long', '0.94')         ('2, Good, long', '0.87')         ('2, Good, long', '0.93')         ('2, Kinetic, long', '0.87')
('1, Good, long', '0.92')         ('2, Good, normal', '0.84')       ('2, Kinetic, long', '0.93')      ('2, Good, long', '0.84')
('1, Good, normal', '0.92')       ('1, Good, synonyms', '0.84')     ('3, Good, normal', '0.92')       ('3, Good, normal', '0.84')
('2, Good, short', '0.92')        ('1, Good, normal', '0.84')       ('4, Good, normal', '0.91')       ('1, Good, normal', '0.83')
('2, Static, long', '0.92')       ('2, Static, long', '0.83')       ('1, Good, normal', '0.91')       ('2, Kinetic, short', '0.83')
('3, Good, normal', '0.92')       ('1, Good, long', '0.83')         ('2, Good, normal', '0.90')       ('2, Good, short', '0.83')
('2, Kinetic, typos', '0.91')     ('2, Good, synonyms', '0.83')     ('2, Good, synonyms', '0.90')     ('1, Good, long', '0.82')
('1, Good, synonyms', '0.91')     ('3, Good, normal', '0.83')       ('1, Good, long', '0.90')         ('2, Good, typos', '0.81')
('2, Good, normal', '0.91')       ('2, Good, short', '0.83')        ('2, Static, long', '0.90')       ('1, Good, synonyms', '0.80')
('2, Good, synonyms', '0.90')     ('1, Kinetic, normal', '0.80')    ('1, Good, synonyms', '0.88')     ('4, Good, normal', '0.79')
('2, Static, normal', '0.90')     ('2, Static, synonyms', '0.80')   ('1, Kinetic, long', '0.88')      ('1, Kinetic, long', '0.78')
('1, Good, short', '0.90')        ('1, Kinetic, long', '0.80')      ('2, Static, normal', '0.88')     ('2, Good, normal', '0.76')
('1, Static, long', '0.89')       ('1, Static, long', '0.80')       ('1, Static, normal', '0.88')     ('2, Good, synonyms', '0.76')
('2, Kinetic, short', '0.89')     ('2, Kinetic, short', '0.79')     ('2, Static, synonyms', '0.88')   ('1, Good, short', '0.75')
('1, Kinetic, long', '0.89')      ('1, Kinetic, synonyms', '0.79')  ('2, Kinetic, short', '0.87')     ('1, Kinetic, short', '0.75')
('2, Static, synonyms', '0.89')   ('2, Static, normal', '0.79')     ('1, Kinetic, normal', '0.86')    ('1, Kinetic, normal', '0.74')
('1, Static, synonyms', '0.88')   ('1, Good, short', '0.79')        ('1, Good, short', '0.86')        ('1, Static, long', '0.73')
('2, Good, typos', '0.88')        ('4, Good, normal', '0.78')       ('1, Static, synonyms', '0.86')   ('1, Static, short', '0.73')
('1, Good, typos', '0.87')        ('1, Static, synonyms', '0.78')   ('1, Static, long', '0.86')       ('2, Static, long', '0.72')
('2, Static, short', '0.87')      ('2, Kinetic, typos', '0.77')     ('1, Kinetic, synonyms', '0.85')  ('2, Static, synonyms', '0.72')
('4, Good, normal', '0.87')       ('1, Good, typos', '0.77')        ('1, Good, typos', '0.85')        ('2, Static, normal', '0.71')
('1, Static, normal', '0.87')     ('1, Static, short', '0.74')      ('2, Good, typos', '0.85')        ('1, Kinetic, synonyms', '0.71')
('2, Static, typos', '0.86')      ('2, Static, short', '0.73')      ('1, Static, short', '0.84')      ('1, Kinetic, typos', '0.71')
('1, Kinetic, normal', '0.86')    ('1, Kinetic, typos', '0.72')     ('2, Static, short', '0.84')      ('1, Good, typos', '0.70')
('1, Kinetic, synonyms', '0.85')  ('1, Static, normal', '0.72')     ('1, Static, typos', '0.82')      ('1, Static, normal', '0.68')
('1, Static, short', '0.85')      ('2, Static, typos', '0.72')      ('1, Kinetic, typos', '0.82')     ('2, Static, short', '0.67')
('1, Kinetic, typos', '0.84')     ('2, Good, typos', '0.72')        ('1, Kinetic, short', '0.81')     ('2, Static, typos', '0.61')
('1, Kinetic, short', '0.82')     ('1, Kinetic, short', '0.67')     ('2, Kinetic, typos', '0.79')     ('1, Static, synonyms', '0.60')
('1, Static, typos', '0.82')      ('1, Static, typos', '0.62')      ('2, Static, typos', '0.76')      ('1, Static, typos', '0.54')
('2, irrelevant', '0.73')         ('1, irrelevant', '0.60')         ('1, irrelevant', '0.66')         ('1, irrelevant', '0.32')
('1, irrelevant', '0.72')         ('2, irrelevant', '0.50')         ('2, irrelevant', '0.60')         ('2, irrelevant', '0.17')
```

Figure .27: The results from comparison to answer labeled "2, Kinetic, normal".



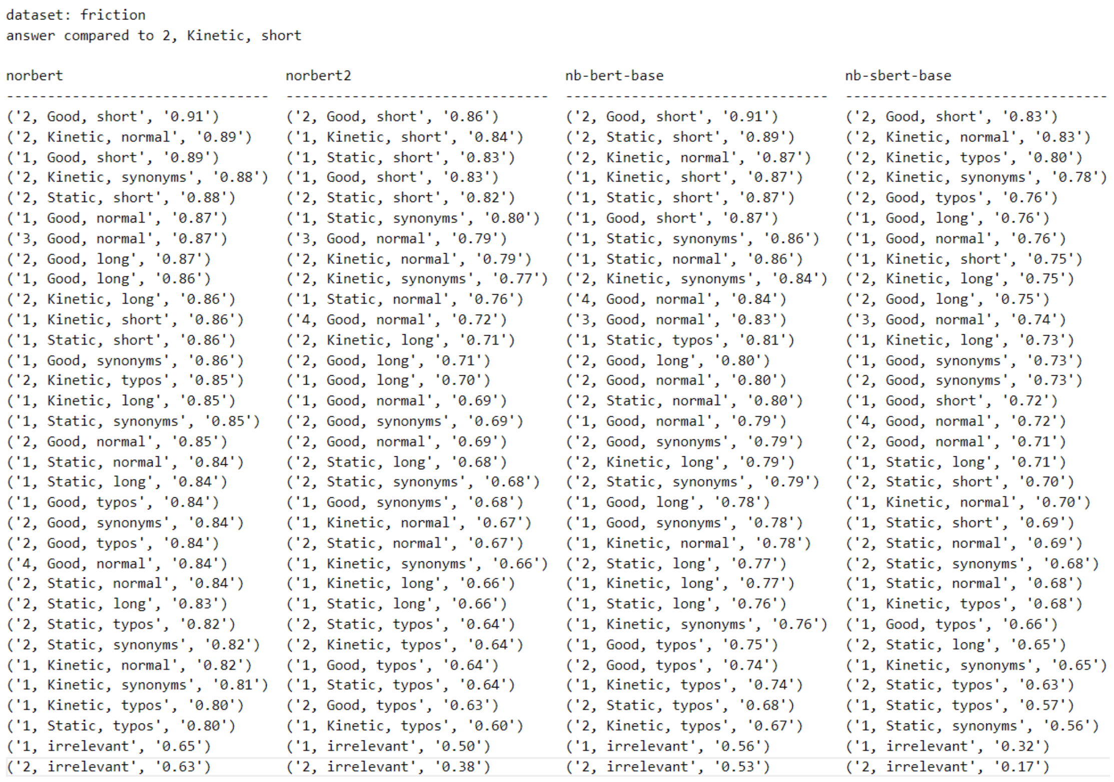
```
dataset: friction
answer compared to 2, Kinetic, typos

norbert                             norbert2                            nb-bert-base                        nb-sbert-base
--------------------------------    --------------------------------    --------------------------------    --------------------------------
('2, Good, typos', '0.92')          ('2, Static, typos', '0.80')        ('2, Good, typos', '0.88')          ('2, Kinetic, normal', '0.95')
('2, Kinetic, normal', '0.91')      ('2, Good, typos', '0.79')          ('2, Static, typos', '0.80')        ('2, Kinetic, synonyms', '0.84')
('2, Kinetic, long', '0.90')        ('1, Good, typos', '0.77')          ('2, Kinetic, normal', '0.79')      ('2, Kinetic, long', '0.84')
('2, Static, typos', '0.90')        ('2, Kinetic, normal', '0.77')      ('2, Kinetic, synonyms', '0.77')    ('2, Good, typos', '0.81')
('2, Kinetic, synonyms', '0.89')    ('2, Kinetic, synonyms', '0.77')    ('2, Kinetic, long', '0.77')        ('2, Kinetic, short', '0.80')
('2, Good, long', '0.89')           ('2, Kinetic, long', '0.75')        ('2, Good, long', '0.77')           ('1, Good, long', '0.80')
('1, Good, long', '0.88')           ('2, Good, long', '0.74')           ('1, Good, typos', '0.75')          ('2, Good, long', '0.79')
('1, Good, short', '0.87')          ('1, Good, synonyms', '0.73')       ('1, Good, normal', '0.74')         ('3, Good, normal', '0.78')
('1, Good, typos', '0.87')          ('1, Good, long', '0.73')           ('1, Good, synonyms', '0.74')       ('1, Good, normal', '0.78')
('1, Good, normal', '0.87')         ('2, Static, long', '0.72')         ('4, Good, normal', '0.74')         ('1, Kinetic, long', '0.77')
('2, Static, long', '0.87')         ('1, Good, normal', '0.72')         ('3, Good, normal', '0.73')         ('2, Good, short', '0.76')
('3, Good, normal', '0.87')         ('1, Kinetic, typos', '0.72')       ('2, Static, long', '0.73')         ('4, Good, normal', '0.74')
('1, Good, synonyms', '0.86')       ('1, Kinetic, long', '0.71')        ('2, Good, normal', '0.73')         ('1, Good, synonyms', '0.73')
('4, Good, normal', '0.86')         ('1, Static, long', '0.70')         ('1, Good, long', '0.73')           ('1, Kinetic, normal', '0.72')
('2, Kinetic, short', '0.85')       ('3, Good, normal', '0.70')         ('1, Kinetic, long', '0.73')        ('1, Static, long', '0.72')
('1, Kinetic, short', '0.85')       ('2, Good, normal', '0.69')         ('1, Kinetic, typos', '0.73')       ('1, Kinetic, typos', '0.71')
('1, Kinetic, long', '0.85')        ('2, Good, synonyms', '0.69')       ('2, Good, synonyms', '0.72')       ('1, Good, short', '0.70')
('1, Static, long', '0.85')         ('4, Good, normal', '0.69')         ('2, Good, short', '0.72')          ('2, Good, normal', '0.70')
('2, Good, normal', '0.85')         ('2, Static, synonyms', '0.68')     ('1, Kinetic, synonyms', '0.72')    ('1, Good, typos', '0.70')
('1, Static, normal', '0.84')       ('1, Kinetic, synonyms', '0.68')    ('2, Static, normal', '0.71')       ('2, Good, synonyms', '0.70')
('1, Kinetic, typos', '0.83')       ('1, Kinetic, normal', '0.68')      ('2, Static, synonyms', '0.71')     ('1, Kinetic, short', '0.69')
('2, Static, normal', '0.83')       ('2, Static, normal', '0.67')       ('1, Kinetic, normal', '0.70')      ('2, Static, long', '0.68')
('2, Good, short', '0.83')          ('2, Good, short', '0.67')          ('1, Static, long', '0.70')         ('1, Kinetic, synonyms', '0.68')
('2, Good, synonyms', '0.83')       ('2, Kinetic, short', '0.64')       ('1, Static, typos', '0.68')        ('1, Static, short', '0.67')
('1, Static, synonyms', '0.82')     ('1, Good, short', '0.63')          ('1, Static, synonyms', '0.68')     ('2, Static, synonyms', '0.65')
('1, Static, short', '0.82')        ('2, Static, short', '0.62')        ('2, Kinetic, short', '0.67')       ('2, Static, normal', '0.65')
('1, Kinetic, synonyms', '0.82')    ('1, Static, short', '0.61')        ('1, Static, normal', '0.67')       ('1, Static, normal', '0.63')
('2, Static, synonyms', '0.82')     ('1, Static, synonyms', '0.61')     ('1, Good, short', '0.67')          ('2, Static, typos', '0.61')
('1, Kinetic, normal', '0.81')      ('1, Static, normal', '0.58')       ('1, Static, short', '0.65')        ('2, Static, short', '0.60')
('1, Static, typos', '0.81')        ('1, Static, typos', '0.57')        ('2, Static, short', '0.63')        ('1, Static, synonyms', '0.53')
('2, Static, short', '0.80')        ('1, Kinetic, short', '0.55')       ('1, Kinetic, short', '0.63')       ('1, Static, typos', '0.53')
('2, irrelevant', '0.67')           ('1, irrelevant', '0.51')           ('1, irrelevant', '0.50')           ('1, irrelevant', '0.30')
('1, irrelevant', '0.67')           ('2, irrelevant', '0.49')           ('2, irrelevant', '0.44')           ('2, irrelevant', '0.16')
```

Figure .28: The results from comparison to answer labeled "2, Kinetic, typos".



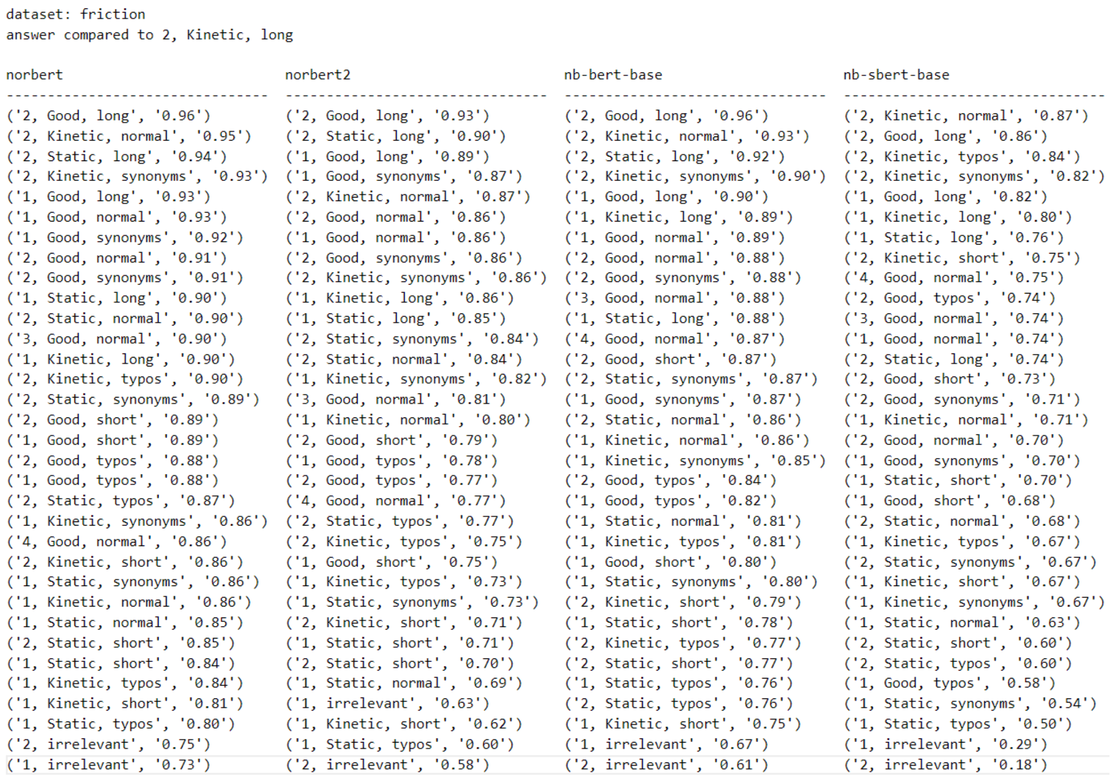
```
dataset: friction
answer compared to 2, Kinetic, synonyms

norbert                             norbert2                            nb-bert-base                        nb-sbert-base
--------------------------------    --------------------------------    --------------------------------    --------------------------------
('2, Kinetic, normal', '0.98')      ('2, Kinetic, normal', '0.95')      ('2, Kinetic, normal', '0.97')      ('2, Kinetic, normal', '0.88')
('2, Kinetic, long', '0.93')        ('2, Kinetic, long', '0.86')        ('2, Good, long', '0.91')           ('2, Kinetic, typos', '0.84')
('2, Good, long', '0.93')           ('2, Good, long', '0.85')           ('2, Kinetic, long', '0.90')        ('2, Kinetic, long', '0.82')
('1, Good, normal', '0.91')         ('1, Good, long', '0.84')           ('2, Good, short', '0.90')          ('1, Kinetic, long', '0.79')
('1, Good, long', '0.91')           ('2, Good, synonyms', '0.82')       ('4, Good, normal', '0.90')         ('1, Good, long', '0.79')
('2, Static, long', '0.91')         ('2, Static, long', '0.82')         ('3, Good, normal', '0.89')         ('2, Kinetic, short', '0.78')
('1, Good, synonyms', '0.90')       ('1, Good, normal', '0.81')         ('2, Good, synonyms', '0.89')       ('2, Good, long', '0.76')
('2, Good, short', '0.90')          ('1, Static, long', '0.81')         ('1, Good, normal', '0.88')         ('3, Good, normal', '0.76')
('3, Good, normal', '0.90')         ('1, Good, synonyms', '0.81')       ('2, Good, normal', '0.88')         ('2, Good, typos', '0.75')
('2, Good, synonyms', '0.89')       ('3, Good, normal', '0.81')         ('1, Good, long', '0.88')           ('2, Good, short', '0.75')
('2, Good, normal', '0.89')         ('2, Good, normal', '0.81')         ('2, Static, long', '0.87')         ('1, Static, long', '0.73')
('1, Kinetic, long', '0.89')        ('1, Kinetic, long', '0.80')        ('1, Static, normal', '0.87')       ('4, Good, normal', '0.73')
('2, Kinetic, typos', '0.89')       ('2, Good, short', '0.80')          ('1, Kinetic, long', '0.87')        ('1, Good, normal', '0.71')
('1, Static, long', '0.89')         ('1, Kinetic, normal', '0.79')      ('2, Static, normal', '0.86')       ('1, Kinetic, normal', '0.71')
('2, Static, normal', '0.89')       ('1, Kinetic, synonyms', '0.78')    ('1, Static, synonyms', '0.86')     ('2, Good, synonyms', '0.71')
('2, Kinetic, short', '0.88')       ('2, Static, synonyms', '0.78')     ('1, Kinetic, normal', '0.86')      ('1, Good, synonyms', '0.69')
('1, Good, short', '0.88')          ('2, Static, normal', '0.77')       ('2, Static, synonyms', '0.86')     ('1, Kinetic, typos', '0.67')
('2, Static, synonyms', '0.87')     ('4, Good, normal', '0.77')         ('1, Good, synonyms', '0.85')       ('2, Good, normal', '0.67')
('1, Good, typos', '0.87')          ('2, Kinetic, short', '0.77')       ('1, Good, short', '0.85')          ('1, Kinetic, short', '0.66')
('4, Good, normal', '0.87')         ('1, Good, typos', '0.77')          ('1, Static, long', '0.85')         ('1, Kinetic, synonyms', '0.65')
('1, Static, synonyms', '0.87')     ('2, Kinetic, typos', '0.77')       ('2, Kinetic, short', '0.84')       ('1, Static, short', '0.65')
('1, Static, normal', '0.86')       ('1, Good, short', '0.76')          ('1, Kinetic, synonyms', '0.84')    ('2, Static, long', '0.65')
('2, Good, typos', '0.86')          ('1, Static, synonyms', '0.74')     ('2, Good, typos', '0.83')          ('1, Good, short', '0.64')
('1, Kinetic, normal', '0.86')      ('1, Kinetic, typos', '0.74')       ('1, Static, short', '0.83')        ('2, Static, normal', '0.63')
('2, Static, typos', '0.85')        ('1, Static, short', '0.72')        ('2, Static, short', '0.82')        ('2, Static, synonyms', '0.62')
('2, Static, short', '0.85')        ('1, Static, normal', '0.70')       ('1, Good, typos', '0.81')          ('1, Static, normal', '0.62')
('1, Kinetic, synonyms', '0.85')    ('2, Static, typos', '0.70')        ('1, Kinetic, typos', '0.81')       ('2, Static, short', '0.59')
('1, Kinetic, typos', '0.84')       ('2, Static, short', '0.70')        ('1, Static, typos', '0.81')        ('2, Static, typos', '0.59')
('1, Static, short', '0.84')        ('2, Good, typos', '0.70')          ('1, Kinetic, short', '0.79')       ('1, Good, typos', '0.57')
('1, Static, typos', '0.82')        ('1, Kinetic, short', '0.64')       ('2, Kinetic, typos', '0.77')       ('1, Static, synonyms', '0.52')
('1, Kinetic, short', '0.81')       ('1, Static, typos', '0.63')        ('2, Static, typos', '0.76')        ('1, Static, typos', '0.50')
('2, irrelevant', '0.73')           ('1, irrelevant', '0.59')           ('1, irrelevant', '0.65')           ('1, irrelevant', '0.34')
('1, irrelevant', '0.72')           ('2, irrelevant', '0.53')           ('2, irrelevant', '0.57')           ('2, irrelevant', '0.16')
```

Figure .29: The results from comparison to answer labeled "2, Kinetic, synonyms".



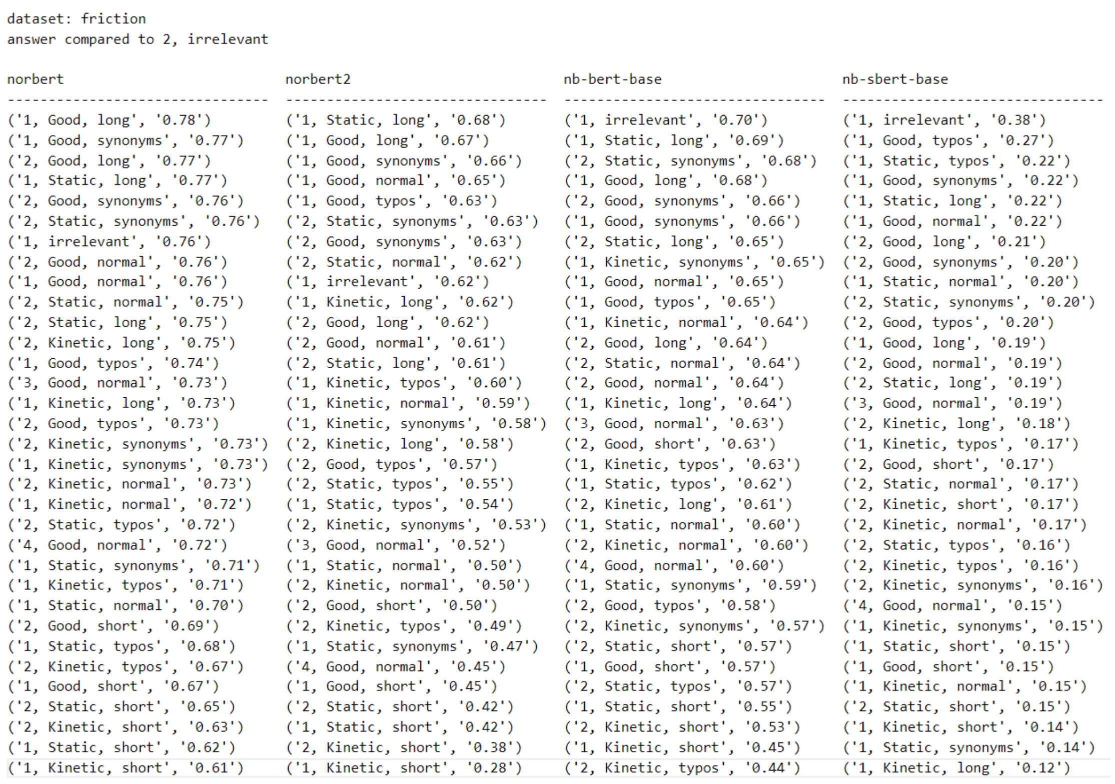
```
dataset: friction
answer compared to 2, Kinetic, short

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
---------------------------------  ---------------------------------  ---------------------------------  ---------------------------------
('2, Good, short', '0.91')         ('2, Good, short', '0.86')         ('2, Good, short', '0.91')         ('2, Good, short', '0.83')
('2, Kinetic, normal', '0.89')     ('1, Kinetic, short', '0.84')      ('2, Static, short', '0.89')       ('2, Kinetic, normal', '0.83')
('1, Good, short', '0.89')         ('1, Static, short', '0.83')       ('2, Kinetic, normal', '0.87')     ('2, Kinetic, typos', '0.80')
('2, Kinetic, synonyms', '0.88')   ('1, Good, short', '0.83')         ('1, Kinetic, short', '0.87')      ('2, Kinetic, synonyms', '0.78')
('2, Static, short', '0.88')       ('2, Static, short', '0.82')       ('1, Static, short', '0.87')       ('2, Good, typos', '0.76')
('1, Good, normal', '0.87')        ('1, Static, synonyms', '0.80')    ('1, Good, short', '0.87')         ('1, Good, long', '0.76')
('3, Good, normal', '0.87')        ('3, Good, normal', '0.79')        ('1, Static, synonyms', '0.86')    ('1, Good, normal', '0.76')
('2, Good, long', '0.87')          ('2, Kinetic, normal', '0.79')     ('1, Static, normal', '0.86')      ('1, Kinetic, short', '0.75')
('1, Good, long', '0.86')          ('2, Kinetic, synonyms', '0.77')   ('2, Kinetic, synonyms', '0.84')   ('2, Kinetic, long', '0.75')
('2, Kinetic, long', '0.86')       ('1, Static, normal', '0.76')      ('4, Good, normal', '0.84')        ('2, Good, long', '0.75')
('1, Kinetic, short', '0.86')      ('4, Good, normal', '0.72')        ('3, Good, normal', '0.83')        ('3, Good, normal', '0.74')
('1, Static, short', '0.86')       ('2, Kinetic, long', '0.71')       ('1, Static, typos', '0.81')       ('1, Kinetic, long', '0.73')
('1, Good, synonyms', '0.86')      ('2, Good, long', '0.71')          ('2, Good, long', '0.80')          ('1, Good, synonyms', '0.73')
('2, Kinetic, typos', '0.85')      ('1, Good, long', '0.70')          ('2, Good, normal', '0.80')        ('2, Good, synonyms', '0.73')
('1, Kinetic, long', '0.85')       ('1, Good, normal', '0.69')        ('2, Static, normal', '0.80')      ('1, Good, short', '0.72')
('1, Static, synonyms', '0.85')    ('2, Good, synonyms', '0.69')      ('1, Good, normal', '0.79')        ('4, Good, normal', '0.72')
('2, Good, normal', '0.85')        ('2, Good, normal', '0.69')        ('2, Good, synonyms', '0.79')      ('2, Good, normal', '0.71')
('1, Static, normal', '0.84')      ('2, Static, long', '0.68')        ('2, Kinetic, long', '0.79')       ('1, Static, long', '0.71')
('1, Static, long', '0.84')        ('2, Static, synonyms', '0.68')    ('2, Static, synonyms', '0.79')    ('2, Static, short', '0.70')
('1, Good, typos', '0.84')         ('1, Good, synonyms', '0.68')      ('1, Good, long', '0.78')          ('1, Kinetic, normal', '0.70')
('2, Good, synonyms', '0.84')      ('1, Kinetic, normal', '0.67')     ('1, Good, synonyms', '0.78')      ('1, Static, short', '0.69')
('2, Good, typos', '0.84')         ('2, Static, normal', '0.67')      ('1, Kinetic, normal', '0.78')     ('2, Static, normal', '0.69')
('4, Good, normal', '0.84')        ('1, Kinetic, synonyms', '0.66')   ('2, Static, long', '0.77')        ('2, Static, synonyms', '0.68')
('2, Static, normal', '0.84')      ('1, Kinetic, long', '0.66')       ('1, Kinetic, long', '0.77')       ('1, Static, normal', '0.68')
('2, Static, long', '0.83')        ('1, Static, long', '0.66')        ('1, Static, long', '0.76')        ('1, Kinetic, typos', '0.68')
('2, Static, typos', '0.82')       ('2, Static, typos', '0.64')       ('1, Kinetic, synonyms', '0.76')   ('1, Good, typos', '0.66')
('2, Static, synonyms', '0.82')    ('2, Kinetic, typos', '0.64')      ('1, Good, typos', '0.75')         ('2, Static, long', '0.65')
('1, Kinetic, normal', '0.82')     ('1, Good, typos', '0.64')         ('2, Good, typos', '0.74')         ('1, Kinetic, synonyms', '0.65')
('1, Kinetic, synonyms', '0.81')   ('1, Static, typos', '0.64')       ('1, Kinetic, typos', '0.74')      ('2, Static, typos', '0.63')
('1, Kinetic, typos', '0.80')      ('2, Good, typos', '0.63')         ('2, Static, typos', '0.68')       ('1, Static, typos', '0.57')
('1, Static, typos', '0.80')       ('1, Kinetic, typos', '0.60')      ('2, Kinetic, typos', '0.67')      ('1, Static, synonyms', '0.56')
('1, irrelevant', '0.65')          ('1, irrelevant', '0.50')          ('1, irrelevant', '0.56')          ('1, irrelevant', '0.32')
('2, irrelevant', '0.63')          ('2, irrelevant', '0.38')          ('2, irrelevant', '0.53')          ('2, irrelevant', '0.17')
```

Figure .30: The results from comparison to answer labeled "2, Kinetic, short".



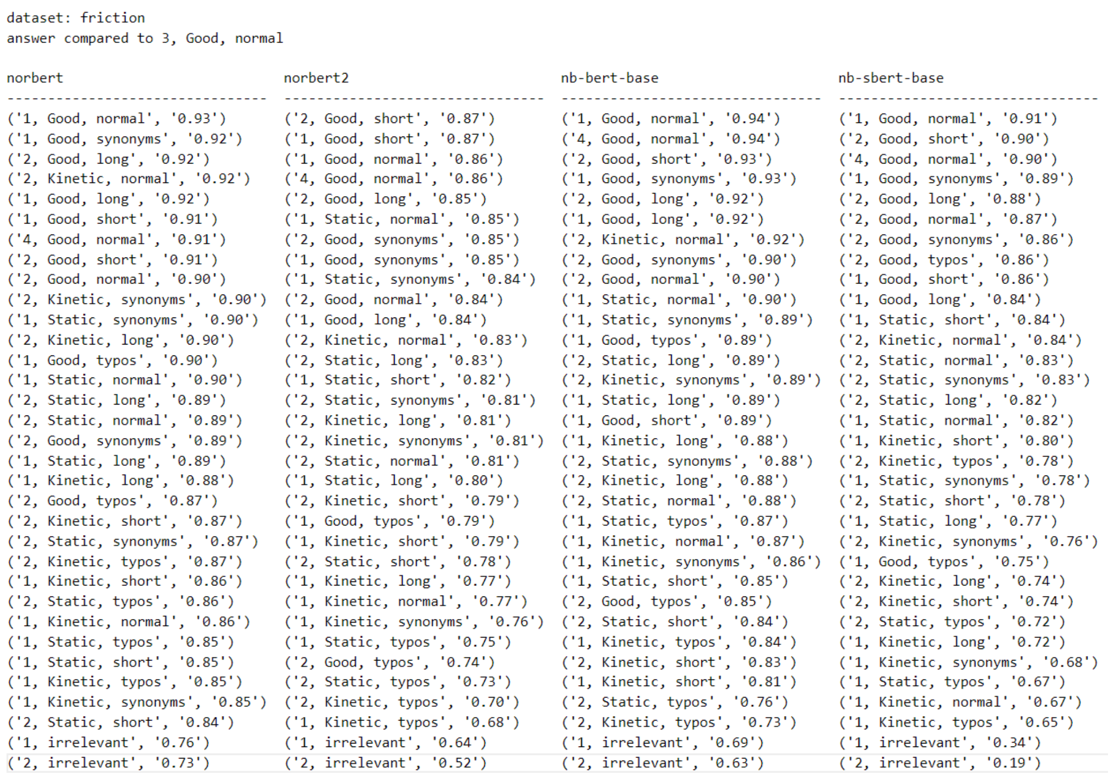
```
dataset: friction
answer compared to 2, Kinetic, long

norbert                            norbert2                           nb-bert-base                       nb-sbert-base
---------------------------------  ---------------------------------  ---------------------------------  ---------------------------------
('2, Good, long', '0.96')          ('2, Good, long', '0.93')          ('2, Good, long', '0.96')          ('2, Kinetic, normal', '0.87')
('2, Kinetic, normal', '0.95')     ('2, Static, long', '0.90')        ('2, Kinetic, normal', '0.93')     ('2, Good, long', '0.86')
('2, Static, long', '0.94')        ('1, Good, long', '0.89')          ('2, Static, long', '0.92')        ('2, Kinetic, typos', '0.84')
('2, Kinetic, synonyms', '0.93')   ('1, Good, synonyms', '0.87')      ('2, Kinetic, synonyms', '0.90')   ('2, Kinetic, synonyms', '0.82')
('1, Good, long', '0.93')          ('2, Kinetic, normal', '0.87')     ('1, Good, long', '0.90')          ('1, Good, long', '0.82')
('1, Good, normal', '0.93')        ('2, Good, normal', '0.86')        ('1, Kinetic, long', '0.89')       ('1, Kinetic, long', '0.80')
('1, Good, synonyms', '0.92')      ('1, Good, normal', '0.86')        ('1, Good, normal', '0.89')        ('1, Static, long', '0.76')
('2, Good, normal', '0.91')        ('2, Good, synonyms', '0.86')      ('2, Good, normal', '0.88')        ('2, Kinetic, short', '0.75')
('2, Good, synonyms', '0.91')      ('2, Kinetic, synonyms', '0.86')   ('2, Good, synonyms', '0.88')      ('4, Good, normal', '0.75')
('1, Static, long', '0.90')        ('1, Kinetic, long', '0.86')       ('3, Good, normal', '0.88')        ('2, Good, typos', '0.74')
('2, Static, normal', '0.90')      ('1, Static, long', '0.85')        ('1, Static, long', '0.88')        ('3, Good, normal', '0.74')
('3, Good, normal', '0.90')        ('2, Static, synonyms', '0.84')    ('4, Good, normal', '0.87')        ('1, Good, normal', '0.74')
('1, Kinetic, long', '0.90')       ('2, Static, normal', '0.84')      ('2, Good, short', '0.87')         ('2, Static, long', '0.74')
('2, Kinetic, typos', '0.90')      ('1, Kinetic, synonyms', '0.82')   ('2, Static, synonyms', '0.87')    ('2, Good, short', '0.73')
('2, Static, synonyms', '0.89')    ('3, Good, normal', '0.81')        ('1, Good, synonyms', '0.87')      ('2, Good, synonyms', '0.71')
('2, Good, short', '0.89')         ('1, Kinetic, normal', '0.80')     ('2, Static, normal', '0.86')      ('1, Kinetic, normal', '0.71')
('1, Good, short', '0.89')         ('2, Good, short', '0.79')         ('1, Kinetic, normal', '0.86')     ('2, Good, normal', '0.70')
('2, Good, typos', '0.88')         ('1, Good, typos', '0.78')         ('1, Kinetic, synonyms', '0.85')   ('1, Good, synonyms', '0.70')
('1, Good, typos', '0.88')         ('2, Good, typos', '0.77')         ('2, Good, typos', '0.84')         ('1, Static, short', '0.70')
('2, Static, typos', '0.87')       ('4, Good, normal', '0.77')        ('1, Good, typos', '0.82')         ('1, Good, short', '0.68')
('1, Kinetic, synonyms', '0.86')   ('2, Static, typos', '0.77')       ('1, Static, normal', '0.81')      ('2, Static, normal', '0.68')
('4, Good, normal', '0.86')        ('2, Kinetic, typos', '0.75')      ('1, Kinetic, typos', '0.81')      ('1, Kinetic, typos', '0.67')
('2, Kinetic, short', '0.86')      ('1, Good, short', '0.75')         ('1, Good, short', '0.80')         ('2, Static, synonyms', '0.67')
('1, Static, synonyms', '0.86')    ('1, Kinetic, typos', '0.73')      ('1, Static, synonyms', '0.80')    ('1, Kinetic, short', '0.67')
('1, Kinetic, normal', '0.86')     ('1, Static, synonyms', '0.73')    ('2, Kinetic, short', '0.79')      ('1, Kinetic, synonyms', '0.67')
('1, Static, normal', '0.85')      ('2, Kinetic, short', '0.71')      ('1, Static, short', '0.78')       ('1, Static, normal', '0.63')
('2, Static, short', '0.85')       ('1, Static, short', '0.71')       ('2, Kinetic, typos', '0.77')      ('2, Static, short', '0.60')
('1, Static, short', '0.84')       ('2, Static, short', '0.70')       ('2, Static, short', '0.77')       ('2, Static, typos', '0.60')
('1, Kinetic, typos', '0.84')      ('1, Static, normal', '0.69')      ('1, Static, typos', '0.76')       ('1, Good, typos', '0.58')
('1, Kinetic, short', '0.81')      ('1, irrelevant', '0.63')          ('2, Static, typos', '0.76')       ('1, Static, synonyms', '0.54')
('1, Static, typos', '0.80')       ('1, Kinetic, short', '0.62')      ('1, Kinetic, short', '0.75')      ('1, Static, typos', '0.50')
('2, irrelevant', '0.75')          ('1, Static, typos', '0.60')       ('1, irrelevant', '0.67')          ('1, irrelevant', '0.29')
('1, irrelevant', '0.73')          ('2, irrelevant', '0.58')          ('2, irrelevant', '0.61')          ('2, irrelevant', '0.18')
```

Figure .31: The results from comparison to answer labeled "2, Kinetic, long".



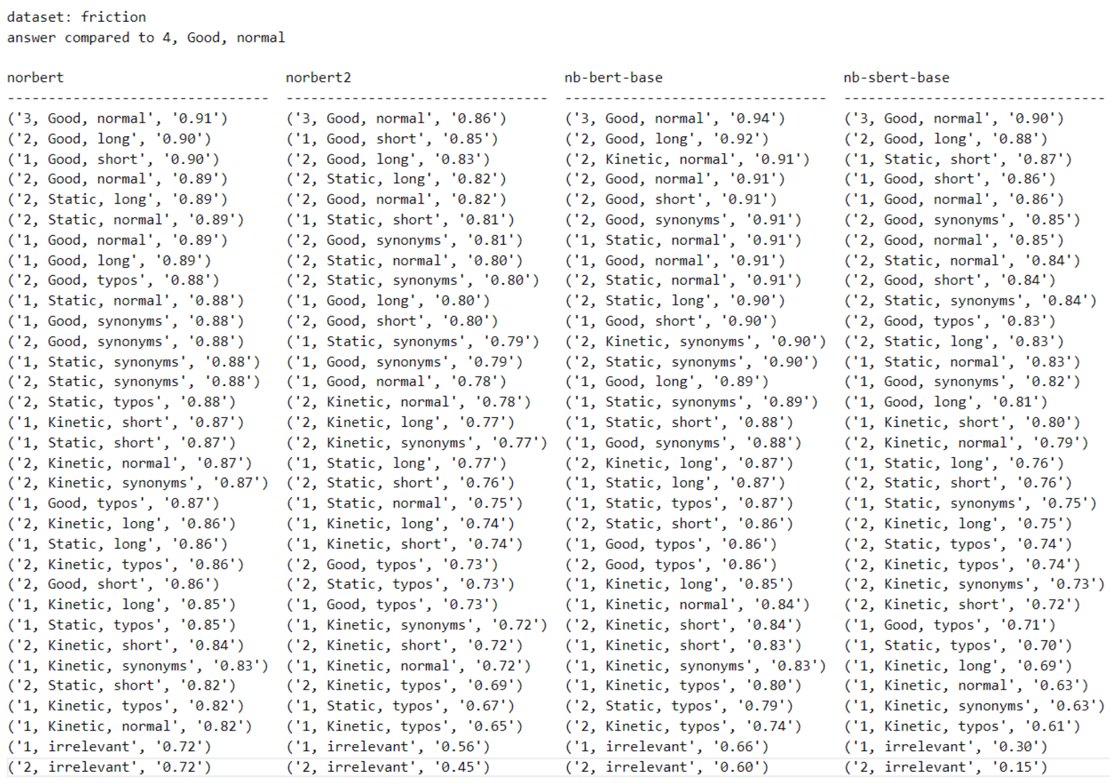
```
dataset: friction
answer compared to 2, irrelevant

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Good, long', '0.78')         ('1, Static, long', '0.68')       ('1, irrelevant', '0.70')         ('1, irrelevant', '0.38')
('1, Good, synonyms', '0.77')     ('1, Good, long', '0.67')         ('1, Static, long', '0.69')       ('1, Good, typos', '0.27')
('2, Good, long', '0.77')         ('1, Good, synonyms', '0.66')     ('2, Static, synonyms', '0.68')   ('1, Static, typos', '0.22')
('1, Static, long', '0.77')       ('1, Good, normal', '0.65')       ('1, Good, long', '0.68')         ('1, Good, synonyms', '0.22')
('2, Good, synonyms', '0.76')     ('1, Good, typos', '0.63')        ('2, Good, synonyms', '0.66')     ('1, Static, long', '0.22')
('2, Static, synonyms', '0.76')   ('2, Static, synonyms', '0.63')   ('1, Good, synonyms', '0.66')     ('1, Good, normal', '0.22')
('1, irrelevant', '0.76')         ('2, Good, synonyms', '0.63')     ('2, Static, long', '0.65')       ('2, Good, long', '0.21')
('2, Good, normal', '0.76')       ('2, Static, normal', '0.62')     ('1, Kinetic, synonyms', '0.65')  ('2, Good, synonyms', '0.20')
('1, Good, normal', '0.76')       ('1, irrelevant', '0.62')         ('1, Good, normal', '0.65')       ('1, Static, normal', '0.20')
('2, Static, normal', '0.75')     ('1, Kinetic, long', '0.62')      ('1, Good, typos', '0.65')        ('2, Static, synonyms', '0.20')
('2, Static, long', '0.75')       ('2, Good, long', '0.62')         ('1, Kinetic, normal', '0.64')    ('2, Good, typos', '0.20')
('2, Kinetic, long', '0.75')      ('2, Good, normal', '0.61')       ('2, Good, long', '0.64')         ('1, Good, long', '0.19')
('1, Good, typos', '0.74')        ('2, Static, long', '0.61')       ('2, Static, normal', '0.64')     ('2, Good, normal', '0.19')
('3, Good, normal', '0.73')       ('1, Kinetic, typos', '0.60')     ('2, Good, normal', '0.64')       ('2, Static, long', '0.19')
('1, Kinetic, long', '0.73')      ('1, Kinetic, normal', '0.59')    ('1, Kinetic, long', '0.64')      ('3, Good, normal', '0.19')
('2, Good, typos', '0.73')        ('1, Kinetic, synonyms', '0.58')  ('3, Good, normal', '0.63')       ('2, Kinetic, long', '0.18')
('2, Kinetic, synonyms', '0.73')  ('2, Kinetic, long', '0.58')      ('2, Good, short', '0.63')        ('1, Kinetic, typos', '0.17')
('1, Kinetic, synonyms', '0.73')  ('2, Good, typos', '0.57')        ('1, Kinetic, typos', '0.63')     ('2, Good, short', '0.17')
('2, Kinetic, normal', '0.73')    ('2, Static, typos', '0.55')      ('1, Static, typos', '0.62')      ('2, Static, normal', '0.17')
('1, Kinetic, normal', '0.72')    ('1, Static, typos', '0.54')      ('2, Kinetic, long', '0.61')      ('2, Kinetic, short', '0.17')
('2, Static, typos', '0.72')      ('2, Kinetic, synonyms', '0.53')  ('1, Static, normal', '0.60')     ('2, Kinetic, normal', '0.17')
('4, Good, normal', '0.72')       ('3, Good, normal', '0.52')       ('2, Kinetic, normal', '0.60')    ('2, Static, typos', '0.16')
('1, Static, synonyms', '0.71')   ('1, Static, normal', '0.50')     ('4, Good, normal', '0.60')       ('2, Kinetic, typos', '0.16')
('1, Kinetic, typos', '0.71')     ('2, Kinetic, normal', '0.50')    ('1, Static, synonyms', '0.59')   ('2, Kinetic, synonyms', '0.16')
('1, Static, normal', '0.70')     ('2, Good, short', '0.50')        ('2, Good, typos', '0.58')        ('4, Good, normal', '0.15')
('2, Good, short', '0.69')        ('2, Kinetic, typos', '0.49')     ('2, Kinetic, synonyms', '0.57')  ('1, Kinetic, synonyms', '0.15')
('1, Static, typos', '0.68')      ('1, Static, synonyms', '0.47')   ('2, Static, short', '0.57')      ('1, Static, short', '0.15')
('2, Kinetic, typos', '0.67')     ('4, Good, normal', '0.45')       ('1, Good, short', '0.57')        ('1, Good, short', '0.15')
('1, Good, short', '0.67')        ('1, Good, short', '0.45')        ('2, Static, typos', '0.57')      ('1, Kinetic, normal', '0.15')
('2, Static, short', '0.65')      ('2, Static, short', '0.42')      ('1, Static, short', '0.55')      ('2, Static, short', '0.15')
('2, Kinetic, short', '0.63')     ('1, Static, short', '0.42')      ('2, Kinetic, short', '0.53')     ('1, Kinetic, short', '0.14')
('1, Static, short', '0.62')      ('2, Kinetic, short', '0.38')     ('1, Kinetic, short', '0.45')     ('1, Static, synonyms', '0.14')
('1, Kinetic, short', '0.61')     ('1, Kinetic, short', '0.28')     ('2, Kinetic, typos', '0.44')     ('1, Kinetic, long', '0.12')
```

Figure .32: The results from comparison to answer labeled "2, irrelevant".



```
dataset: friction
answer compared to 3, Good, normal

norbert                           norbert2                          nb-bert-base                      nb-sbert-base
--------------------------------  --------------------------------  --------------------------------  --------------------------------
('1, Good, normal', '0.93')       ('2, Good, short', '0.87')        ('1, Good, normal', '0.94')       ('1, Good, normal', '0.91')
('1, Good, synonyms', '0.92')     ('1, Good, short', '0.87')        ('4, Good, normal', '0.94')       ('2, Good, short', '0.90')
('2, Good, long', '0.92')         ('1, Good, normal', '0.86')       ('2, Good, short', '0.93')        ('4, Good, normal', '0.90')
('2, Kinetic, normal', '0.92')    ('4, Good, normal', '0.86')       ('1, Good, synonyms', '0.93')     ('1, Good, synonyms', '0.89')
('1, Good, long', '0.92')         ('2, Good, long', '0.85')         ('2, Good, long', '0.92')         ('2, Good, long', '0.88')
('1, Good, short', '0.91')        ('1, Static, normal', '0.85')     ('1, Good, long', '0.92')         ('2, Good, normal', '0.87')
('4, Good, normal', '0.91')       ('2, Good, synonyms', '0.85')     ('2, Kinetic, normal', '0.92')    ('2, Good, synonyms', '0.86')
('2, Good, short', '0.91')        ('1, Good, synonyms', '0.85')     ('2, Good, synonyms', '0.90')     ('2, Good, typos', '0.86')
('2, Good, normal', '0.90')       ('1, Static, synonyms', '0.84')   ('2, Good, normal', '0.90')       ('1, Good, short', '0.86')
('2, Kinetic, synonyms', '0.90')  ('2, Good, normal', '0.84')       ('1, Static, normal', '0.90')     ('1, Good, long', '0.84')
('1, Static, synonyms', '0.90')   ('1, Good, long', '0.84')         ('1, Static, synonyms', '0.89')   ('1, Static, short', '0.84')
('2, Kinetic, long', '0.90')      ('2, Kinetic, normal', '0.83')    ('1, Good, typos', '0.89')        ('2, Kinetic, normal', '0.84')
('1, Good, typos', '0.90')        ('2, Static, long', '0.83')       ('2, Static, long', '0.89')       ('2, Static, normal', '0.83')
('1, Static, normal', '0.90')     ('1, Static, short', '0.82')      ('2, Kinetic, synonyms', '0.89')  ('2, Static, synonyms', '0.83')
('2, Static, long', '0.89')       ('2, Static, synonyms', '0.81')   ('1, Static, long', '0.89')       ('2, Static, long', '0.82')
('2, Static, normal', '0.89')     ('2, Kinetic, long', '0.81')      ('1, Good, short', '0.89')        ('1, Static, normal', '0.82')
('2, Good, synonyms', '0.89')     ('2, Kinetic, synonyms', '0.81')  ('1, Kinetic, long', '0.88')      ('1, Kinetic, short', '0.80')
('1, Static, long', '0.89')       ('2, Static, normal', '0.81')     ('2, Static, synonyms', '0.88')   ('2, Kinetic, typos', '0.78')
('1, Kinetic, long', '0.88')      ('1, Static, long', '0.80')       ('2, Kinetic, long', '0.88')      ('1, Static, synonyms', '0.78')
('2, Good, typos', '0.87')        ('2, Kinetic, short', '0.79')     ('2, Static, normal', '0.88')     ('2, Static, short', '0.78')
('2, Kinetic, short', '0.87')     ('1, Good, typos', '0.79')        ('1, Static, typos', '0.87')      ('1, Static, long', '0.77')
('2, Static, synonyms', '0.87')   ('1, Kinetic, short', '0.79')     ('1, Kinetic, normal', '0.87')    ('2, Kinetic, synonyms', '0.76')
('2, Kinetic, typos', '0.87')     ('2, Static, short', '0.78')      ('1, Kinetic, synonyms', '0.86')  ('1, Good, typos', '0.75')
('1, Kinetic, short', '0.86')     ('1, Kinetic, long', '0.77')      ('1, Static, short', '0.85')      ('2, Kinetic, long', '0.74')
('2, Static, typos', '0.86')      ('1, Kinetic, normal', '0.77')    ('2, Good, typos', '0.85')        ('2, Kinetic, short', '0.74')
('1, Kinetic, normal', '0.86')    ('1, Kinetic, synonyms', '0.76')  ('2, Static, short', '0.84')      ('2, Static, typos', '0.72')
('1, Static, typos', '0.85')      ('1, Static, typos', '0.75')      ('1, Kinetic, typos', '0.84')     ('1, Kinetic, long', '0.72')
('1, Static, short', '0.85')      ('2, Good, typos', '0.74')        ('2, Kinetic, short', '0.83')     ('1, Kinetic, synonyms', '0.68')
('1, Kinetic, typos', '0.85')     ('2, Static, typos', '0.73')      ('1, Kinetic, short', '0.81')     ('1, Static, typos', '0.67')
('1, Kinetic, synonyms', '0.85')  ('2, Kinetic, typos', '0.70')     ('2, Static, typos', '0.76')      ('1, Kinetic, normal', '0.67')
('2, Static, short', '0.84')      ('1, Kinetic, typos', '0.68')     ('2, Kinetic, typos', '0.73')     ('1, Kinetic, typos', '0.65')
('1, irrelevant', '0.76')         ('1, irrelevant', '0.64')         ('1, irrelevant', '0.69')         ('1, irrelevant', '0.34')
('2, irrelevant', '0.73')         ('2, irrelevant', '0.52')         ('2, irrelevant', '0.63')         ('2, irrelevant', '0.19')
```

Figure .33: The results from comparison to answer labeled "3, Good, normal".

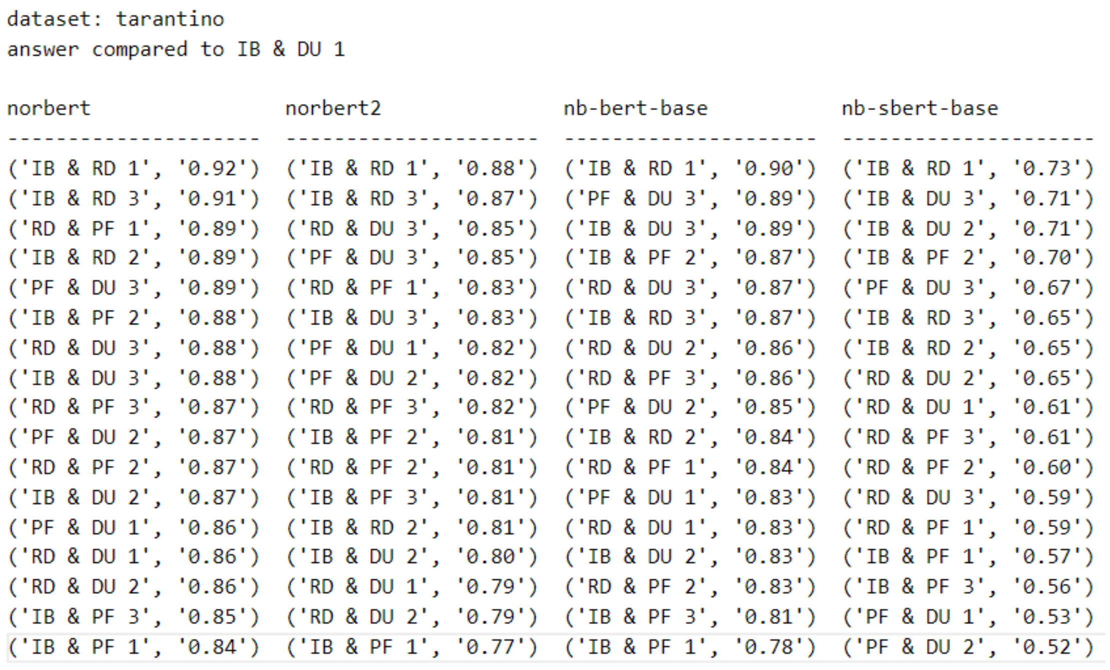


```
dataset: friction
answer compared to 4, Good, normal

norbert                            norbert2                           nb-bert-base                        nb-sbert-base
--------------------------------   --------------------------------   --------------------------------    --------------------------------
('3, Good, normal', '0.91')        ('3, Good, normal', '0.86')        ('3, Good, normal', '0.94')         ('3, Good, normal', '0.90')
('2, Good, long', '0.90')          ('1, Good, short', '0.85')         ('2, Good, long', '0.92')           ('2, Good, long', '0.88')
('1, Good, short', '0.90')         ('2, Good, long', '0.83')          ('2, Kinetic, normal', '0.91')      ('1, Static, short', '0.87')
('2, Good, normal', '0.89')        ('2, Static, long', '0.82')        ('2, Good, normal', '0.91')         ('1, Good, short', '0.86')
('2, Static, long', '0.89')        ('2, Good, normal', '0.82')        ('2, Good, short', '0.91')          ('1, Good, normal', '0.86')
('2, Static, normal', '0.89')      ('1, Static, short', '0.81')       ('2, Good, synonyms', '0.91')       ('2, Good, synonyms', '0.85')
('1, Good, normal', '0.89')        ('2, Good, synonyms', '0.81')      ('1, Static, normal', '0.91')       ('2, Good, normal', '0.85')
('1, Good, long', '0.89')          ('2, Static, normal', '0.80')      ('1, Good, normal', '0.91')         ('2, Static, normal', '0.84')
('2, Good, typos', '0.88')         ('2, Static, synonyms', '0.80')    ('2, Static, normal', '0.91')       ('2, Good, short', '0.84')
('1, Static, normal', '0.88')      ('1, Good, long', '0.80')          ('2, Static, long', '0.90')         ('2, Static, synonyms', '0.84')
('1, Good, synonyms', '0.88')      ('2, Good, short', '0.80')         ('1, Good, short', '0.90')          ('2, Good, typos', '0.83')
('2, Good, synonyms', '0.88')      ('1, Static, synonyms', '0.79')    ('2, Kinetic, synonyms', '0.90')    ('2, Static, long', '0.83')
('1, Static, synonyms', '0.88')    ('1, Good, synonyms', '0.79')      ('2, Static, synonyms', '0.90')     ('1, Static, normal', '0.83')
('2, Static, synonyms', '0.88')    ('1, Good, normal', '0.78')        ('1, Good, long', '0.89')           ('1, Good, synonyms', '0.82')
('2, Static, typos', '0.88')       ('2, Kinetic, normal', '0.78')     ('1, Static, synonyms', '0.89')     ('1, Good, long', '0.81')
('1, Kinetic, short', '0.87')      ('2, Kinetic, long', '0.77')       ('1, Static, short', '0.88')        ('1, Kinetic, short', '0.80')
('1, Static, short', '0.87')       ('2, Kinetic, synonyms', '0.77')   ('1, Good, synonyms', '0.88')       ('2, Kinetic, normal', '0.79')
('2, Kinetic, normal', '0.87')     ('1, Static, long', '0.77')        ('2, Kinetic, long', '0.87')        ('1, Static, long', '0.76')
('2, Kinetic, synonyms', '0.87')   ('2, Static, short', '0.76')       ('1, Static, long', '0.87')         ('2, Static, short', '0.76')
('1, Good, typos', '0.87')         ('1, Static, normal', '0.75')      ('1, Static, typos', '0.87')        ('1, Static, synonyms', '0.75')
('2, Kinetic, long', '0.86')       ('1, Kinetic, long', '0.74')       ('2, Static, short', '0.86')        ('2, Kinetic, long', '0.75')
('1, Static, long', '0.86')        ('1, Kinetic, short', '0.74')      ('1, Good, typos', '0.86')          ('2, Static, typos', '0.74')
('2, Kinetic, typos', '0.86')      ('2, Good, typos', '0.73')         ('2, Good, typos', '0.86')          ('2, Kinetic, typos', '0.74')
('2, Good, short', '0.86')         ('2, Static, typos', '0.73')       ('1, Kinetic, long', '0.85')        ('2, Kinetic, synonyms', '0.73')
('1, Kinetic, long', '0.85')       ('1, Good, typos', '0.73')         ('1, Kinetic, normal', '0.84')      ('2, Kinetic, short', '0.72')
('1, Static, typos', '0.85')       ('1, Kinetic, synonyms', '0.72')   ('2, Kinetic, short', '0.84')       ('1, Good, typos', '0.71')
('2, Kinetic, short', '0.84')      ('2, Kinetic, short', '0.72')      ('1, Kinetic, short', '0.83')       ('1, Static, typos', '0.70')
('1, Kinetic, synonyms', '0.83')   ('1, Kinetic, normal', '0.72')     ('1, Kinetic, synonyms', '0.83')    ('1, Kinetic, long', '0.69')
('2, Static, short', '0.82')       ('2, Kinetic, typos', '0.69')      ('1, Kinetic, typos', '0.80')       ('1, Kinetic, normal', '0.63')
('1, Kinetic, typos', '0.82')      ('1, Static, typos', '0.67')       ('2, Static, typos', '0.79')        ('1, Kinetic, synonyms', '0.63')
('1, Kinetic, normal', '0.82')     ('1, Kinetic, typos', '0.65')      ('2, Kinetic, typos', '0.74')       ('1, Kinetic, typos', '0.61')
('1, irrelevant', '0.72')          ('1, irrelevant', '0.56')          ('1, irrelevant', '0.66')           ('1, irrelevant', '0.30')
('2, irrelevant', '0.72')          ('2, irrelevant', '0.45')          ('2, irrelevant', '0.60')           ('2, irrelevant', '0.15')
```

Figure .34: The results from comparison to answer labeled "4, Good, normal".

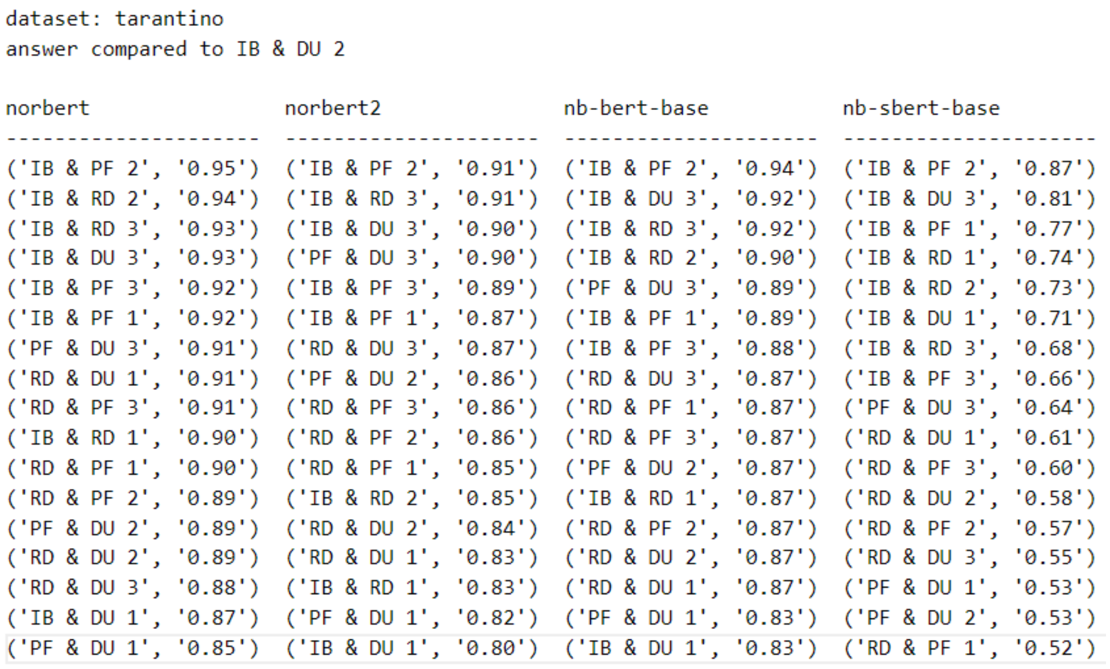
B2 - Tarantino original dataset

All results from the tests performed on the Tarantino dataset with two movie descriptions per entry and titles included.



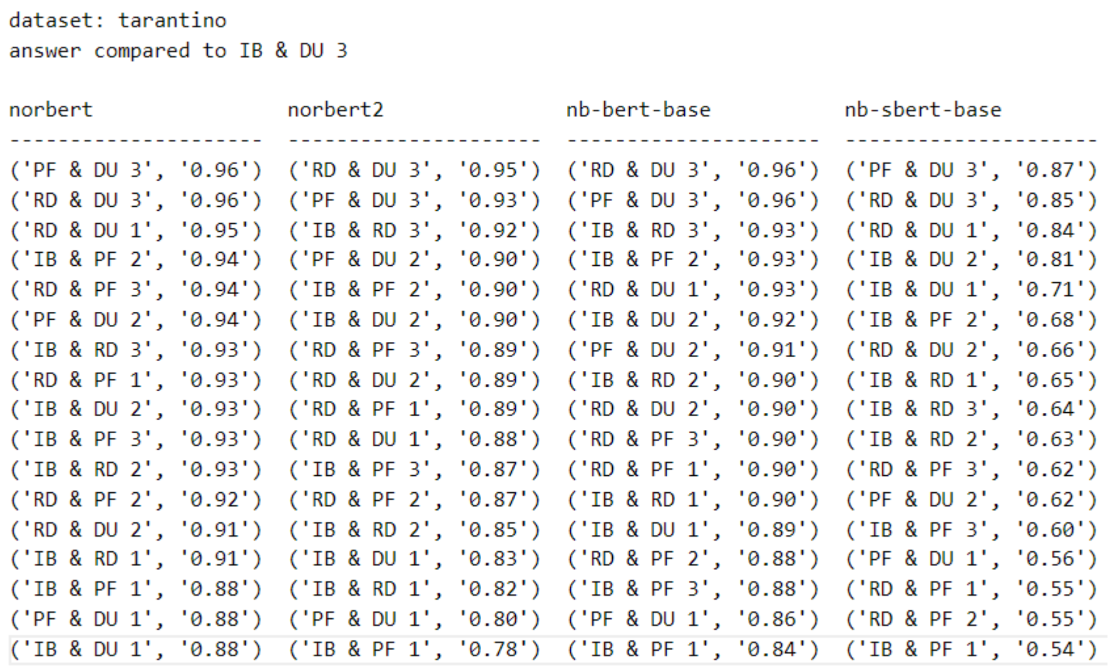
```
dataset: tarantino
answer compared to IB & DU 1

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & RD 1', '0.92')   ('IB & RD 1', '0.88')   ('IB & RD 1', '0.90')   ('IB & RD 1', '0.73')
('IB & RD 3', '0.91')   ('IB & RD 3', '0.87')   ('PF & DU 3', '0.89')   ('IB & DU 3', '0.71')
('RD & PF 1', '0.89')   ('RD & DU 3', '0.85')   ('IB & DU 3', '0.89')   ('IB & DU 2', '0.71')
('IB & RD 2', '0.89')   ('PF & DU 3', '0.85')   ('IB & PF 2', '0.87')   ('IB & PF 2', '0.70')
('PF & DU 3', '0.89')   ('RD & PF 1', '0.83')   ('RD & DU 3', '0.87')   ('PF & DU 3', '0.67')
('IB & PF 2', '0.88')   ('IB & DU 3', '0.83')   ('IB & RD 3', '0.87')   ('IB & RD 3', '0.65')
('RD & DU 3', '0.88')   ('PF & DU 1', '0.82')   ('RD & DU 2', '0.86')   ('IB & RD 2', '0.65')
('IB & DU 3', '0.88')   ('PF & DU 2', '0.82')   ('RD & PF 3', '0.86')   ('RD & DU 2', '0.65')
('RD & PF 3', '0.87')   ('RD & PF 3', '0.82')   ('PF & DU 2', '0.85')   ('RD & DU 1', '0.61')
('PF & DU 2', '0.87')   ('IB & PF 2', '0.81')   ('IB & RD 2', '0.84')   ('RD & PF 3', '0.61')
('RD & PF 2', '0.87')   ('RD & PF 2', '0.81')   ('RD & PF 1', '0.84')   ('RD & PF 2', '0.60')
('IB & DU 2', '0.87')   ('IB & PF 3', '0.81')   ('PF & DU 1', '0.83')   ('RD & DU 3', '0.59')
('PF & DU 1', '0.86')   ('IB & RD 2', '0.81')   ('RD & DU 1', '0.83')   ('RD & PF 1', '0.59')
('RD & DU 1', '0.86')   ('IB & DU 2', '0.80')   ('IB & DU 2', '0.83')   ('IB & PF 1', '0.57')
('RD & DU 2', '0.86')   ('RD & DU 1', '0.79')   ('RD & PF 2', '0.83')   ('IB & PF 3', '0.56')
('IB & PF 3', '0.85')   ('RD & DU 2', '0.79')   ('IB & PF 3', '0.81')   ('PF & DU 1', '0.53')
('IB & PF 1', '0.84')   ('IB & PF 1', '0.77')   ('IB & PF 1', '0.78')   ('PF & DU 2', '0.52')
```

Figure .35: The results from comparison to entry labeled "IB & DU 1".



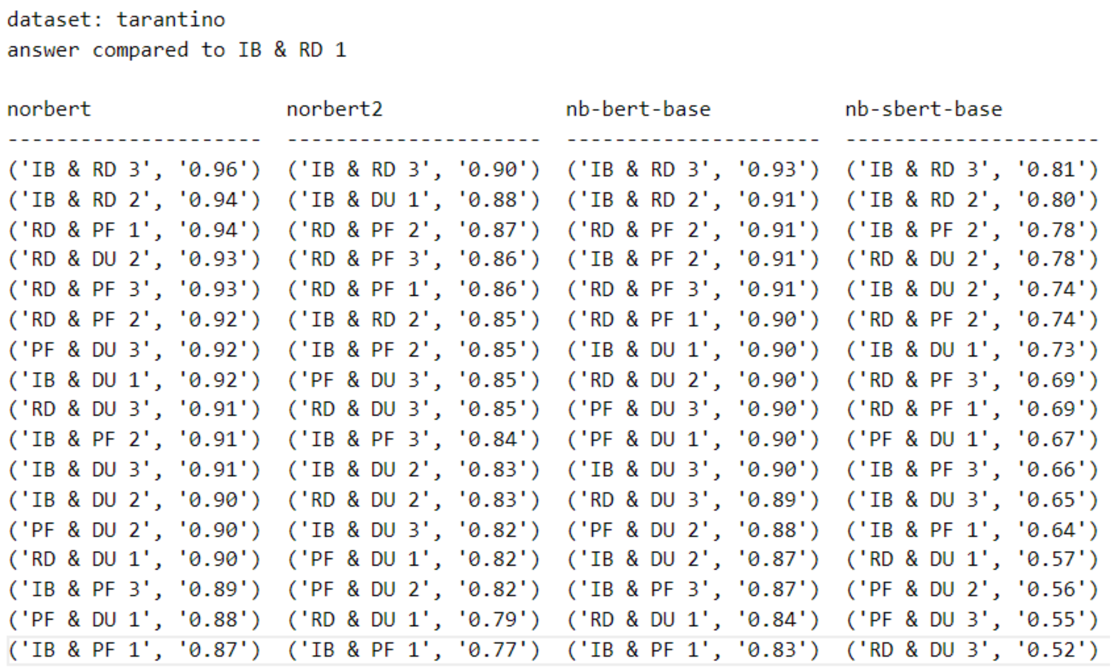
```
dataset: tarantino
answer compared to IB & DU 2

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & PF 2', '0.95')   ('IB & PF 2', '0.91')   ('IB & PF 2', '0.94')   ('IB & PF 2', '0.87')
('IB & RD 2', '0.94')   ('IB & RD 3', '0.91')   ('IB & DU 3', '0.92')   ('IB & DU 3', '0.81')
('IB & RD 3', '0.93')   ('IB & DU 3', '0.90')   ('IB & RD 3', '0.92')   ('IB & PF 1', '0.77')
('IB & DU 3', '0.93')   ('PF & DU 3', '0.90')   ('IB & RD 2', '0.90')   ('IB & RD 1', '0.74')
('IB & PF 3', '0.92')   ('IB & PF 3', '0.89')   ('PF & DU 3', '0.89')   ('IB & RD 2', '0.73')
('IB & PF 1', '0.92')   ('IB & PF 1', '0.87')   ('IB & PF 1', '0.89')   ('IB & DU 1', '0.71')
('PF & DU 3', '0.91')   ('RD & DU 3', '0.87')   ('IB & PF 3', '0.88')   ('IB & RD 3', '0.68')
('RD & DU 1', '0.91')   ('PF & DU 2', '0.86')   ('RD & DU 3', '0.87')   ('IB & PF 3', '0.66')
('RD & PF 3', '0.91')   ('RD & PF 3', '0.86')   ('RD & PF 1', '0.87')   ('PF & DU 3', '0.64')
('IB & RD 1', '0.90')   ('RD & PF 2', '0.86')   ('RD & PF 3', '0.87')   ('RD & DU 1', '0.61')
('RD & PF 1', '0.90')   ('RD & PF 1', '0.85')   ('PF & DU 2', '0.87')   ('RD & PF 3', '0.60')
('RD & PF 2', '0.89')   ('IB & RD 2', '0.85')   ('IB & RD 1', '0.87')   ('RD & DU 2', '0.58')
('PF & DU 2', '0.89')   ('RD & DU 2', '0.84')   ('RD & PF 2', '0.87')   ('RD & PF 2', '0.57')
('RD & DU 2', '0.89')   ('RD & DU 1', '0.83')   ('RD & DU 2', '0.87')   ('RD & DU 3', '0.55')
('RD & DU 3', '0.88')   ('IB & RD 1', '0.83')   ('RD & DU 1', '0.87')   ('PF & DU 1', '0.53')
('IB & DU 1', '0.87')   ('PF & DU 1', '0.82')   ('PF & DU 1', '0.83')   ('PF & DU 2', '0.53')
('PF & DU 1', '0.85')   ('IB & DU 1', '0.80')   ('IB & DU 1', '0.83')   ('RD & PF 1', '0.52')
```

Figure .36: The results from comparison to entry labeled "IB & DU 2".



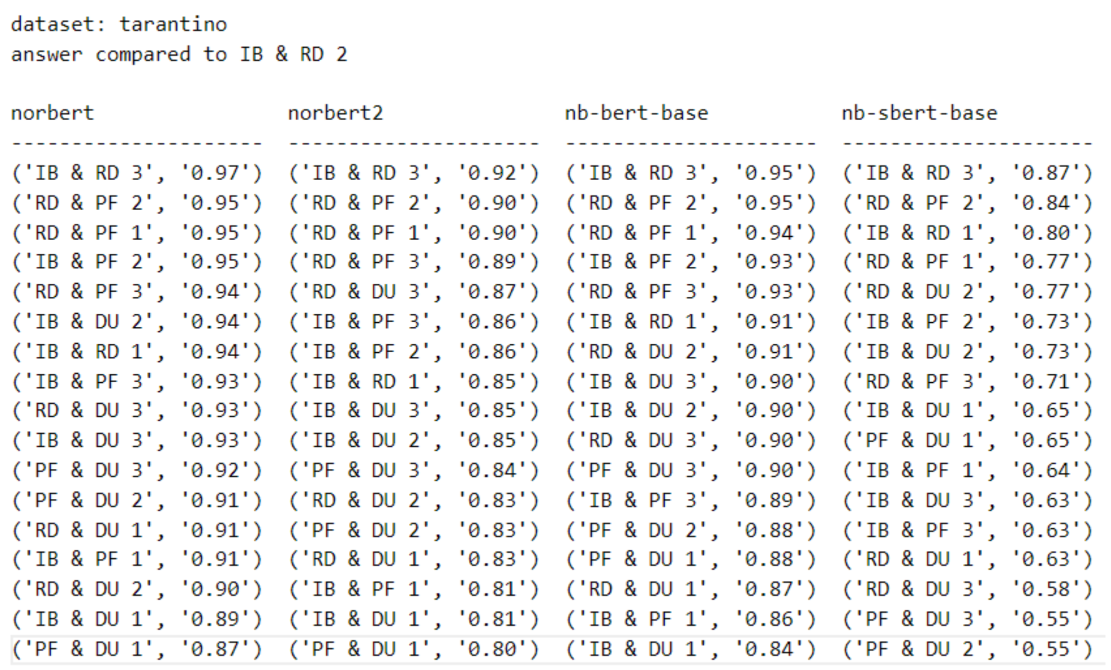
```
dataset: tarantino
answer compared to IB & DU 3

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('PF & DU 3', '0.96')   ('RD & DU 3', '0.95')   ('RD & DU 3', '0.96')   ('PF & DU 3', '0.87')
('RD & DU 3', '0.96')   ('PF & DU 3', '0.93')   ('PF & DU 3', '0.96')   ('RD & DU 3', '0.85')
('RD & DU 1', '0.95')   ('IB & RD 3', '0.92')   ('IB & RD 3', '0.93')   ('RD & DU 1', '0.84')
('IB & PF 2', '0.94')   ('PF & DU 2', '0.90')   ('IB & PF 2', '0.93')   ('IB & DU 2', '0.81')
('RD & PF 3', '0.94')   ('IB & PF 2', '0.90')   ('RD & DU 1', '0.93')   ('IB & DU 1', '0.71')
('PF & DU 2', '0.94')   ('IB & DU 2', '0.90')   ('IB & DU 2', '0.92')   ('IB & PF 2', '0.68')
('IB & RD 3', '0.93')   ('RD & PF 3', '0.89')   ('PF & DU 2', '0.91')   ('RD & DU 2', '0.66')
('RD & PF 1', '0.93')   ('RD & DU 2', '0.89')   ('IB & RD 2', '0.90')   ('IB & RD 1', '0.65')
('IB & DU 2', '0.93')   ('RD & PF 1', '0.89')   ('RD & DU 2', '0.90')   ('IB & RD 3', '0.64')
('IB & PF 3', '0.93')   ('RD & DU 1', '0.88')   ('RD & PF 3', '0.90')   ('IB & RD 2', '0.63')
('IB & RD 2', '0.93')   ('IB & PF 3', '0.87')   ('RD & PF 1', '0.90')   ('RD & PF 3', '0.62')
('RD & PF 2', '0.92')   ('RD & PF 2', '0.87')   ('IB & RD 1', '0.90')   ('PF & DU 2', '0.62')
('RD & DU 2', '0.91')   ('IB & RD 2', '0.85')   ('IB & DU 1', '0.89')   ('IB & PF 3', '0.60')
('IB & RD 1', '0.91')   ('IB & DU 1', '0.83')   ('RD & PF 2', '0.88')   ('PF & DU 1', '0.56')
('IB & PF 1', '0.88')   ('IB & RD 1', '0.82')   ('IB & PF 3', '0.88')   ('RD & PF 1', '0.55')
('PF & DU 1', '0.88')   ('PF & DU 1', '0.80')   ('PF & DU 1', '0.86')   ('RD & PF 2', '0.55')
('IB & DU 1', '0.88')   ('IB & PF 1', '0.78')   ('IB & PF 1', '0.84')   ('IB & PF 1', '0.54')
```

Figure .37: The results from comparison to entry labeled "IB & DU 3".



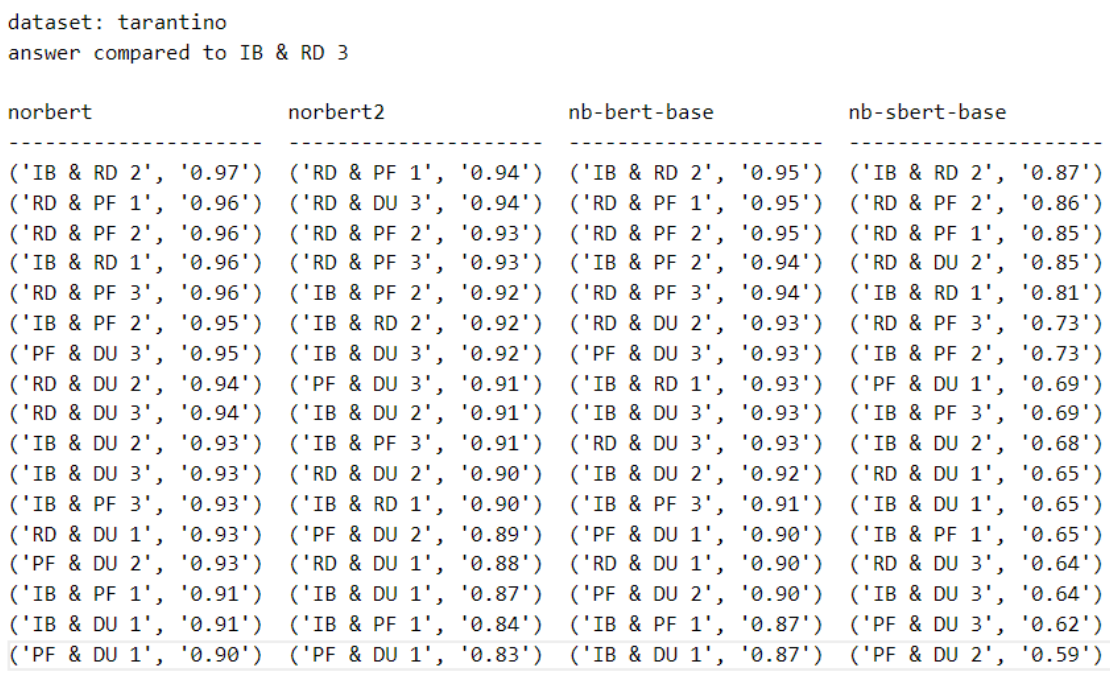
```
dataset: tarantino
answer compared to IB & RD 1

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & RD 3', '0.96')   ('IB & RD 3', '0.90')   ('IB & RD 3', '0.93')   ('IB & RD 3', '0.81')
('IB & RD 2', '0.94')   ('IB & DU 1', '0.88')   ('IB & RD 2', '0.91')   ('IB & RD 2', '0.80')
('RD & PF 1', '0.94')   ('RD & PF 2', '0.87')   ('RD & PF 2', '0.91')   ('IB & PF 2', '0.78')
('RD & DU 2', '0.93')   ('RD & PF 3', '0.86')   ('IB & PF 2', '0.91')   ('RD & DU 2', '0.78')
('RD & PF 3', '0.93')   ('RD & PF 1', '0.86')   ('RD & PF 3', '0.91')   ('IB & DU 2', '0.74')
('RD & PF 2', '0.92')   ('IB & RD 2', '0.85')   ('RD & PF 1', '0.90')   ('RD & PF 2', '0.74')
('PF & DU 3', '0.92')   ('IB & PF 2', '0.85')   ('IB & DU 1', '0.90')   ('IB & DU 1', '0.73')
('IB & DU 1', '0.92')   ('PF & DU 3', '0.85')   ('RD & DU 2', '0.90')   ('RD & PF 3', '0.69')
('RD & DU 3', '0.91')   ('RD & DU 3', '0.85')   ('PF & DU 3', '0.90')   ('RD & PF 1', '0.69')
('IB & PF 2', '0.91')   ('IB & PF 3', '0.84')   ('PF & DU 1', '0.90')   ('PF & DU 1', '0.67')
('IB & DU 3', '0.91')   ('IB & DU 2', '0.83')   ('IB & DU 3', '0.90')   ('IB & PF 3', '0.66')
('IB & DU 2', '0.90')   ('RD & DU 2', '0.83')   ('RD & DU 3', '0.89')   ('IB & DU 3', '0.65')
('PF & DU 2', '0.90')   ('IB & DU 3', '0.82')   ('PF & DU 2', '0.88')   ('IB & PF 1', '0.64')
('RD & DU 1', '0.90')   ('PF & DU 1', '0.82')   ('IB & DU 2', '0.87')   ('RD & DU 1', '0.57')
('IB & PF 3', '0.89')   ('PF & DU 2', '0.82')   ('IB & PF 3', '0.87')   ('PF & DU 2', '0.56')
('PF & DU 1', '0.88')   ('RD & DU 1', '0.79')   ('RD & DU 1', '0.84')   ('PF & DU 3', '0.55')
('IB & PF 1', '0.87')   ('IB & PF 1', '0.77')   ('IB & PF 1', '0.83')   ('RD & DU 3', '0.52')
```

Figure .38: The results from comparison to entry labeled "IB & RD 1".



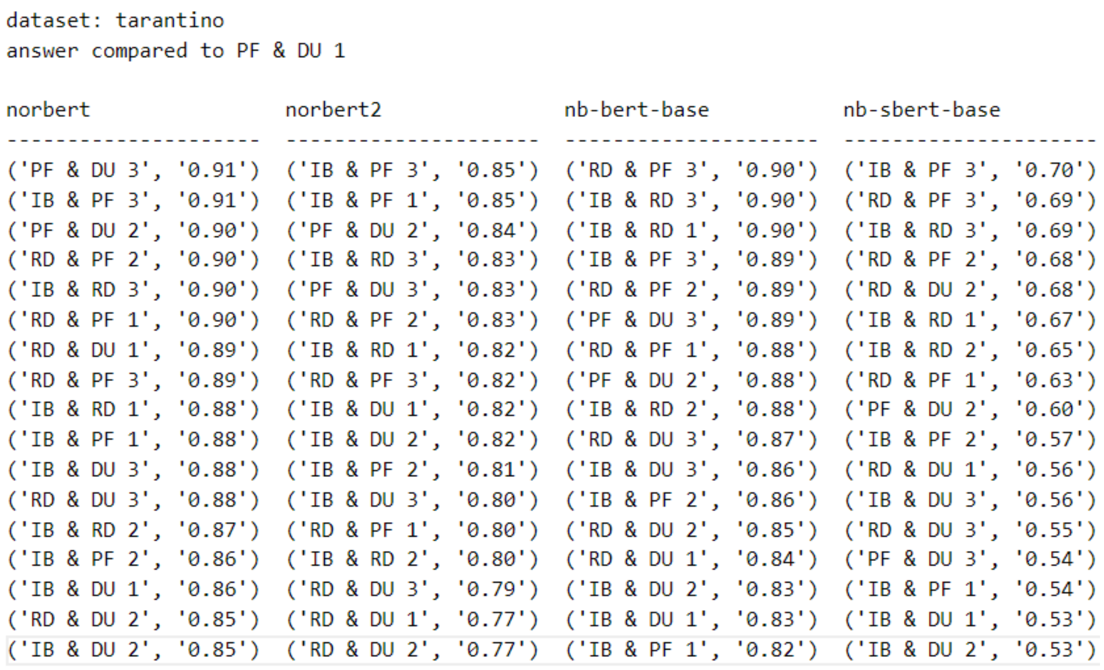
```
dataset: tarantino
answer compared to IB & RD 2

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('IB & RD 3', '0.97')    ('IB & RD 3', '0.92')    ('IB & RD 3', '0.95')    ('IB & RD 3', '0.87')
('RD & PF 2', '0.95')    ('RD & PF 2', '0.90')    ('RD & PF 2', '0.95')    ('RD & PF 2', '0.84')
('RD & PF 1', '0.95')    ('RD & PF 1', '0.90')    ('RD & PF 1', '0.94')    ('IB & RD 1', '0.80')
('IB & PF 2', '0.95')    ('RD & PF 3', '0.89')    ('IB & PF 2', '0.93')    ('RD & PF 1', '0.77')
('RD & PF 3', '0.94')    ('RD & DU 3', '0.87')    ('RD & PF 3', '0.93')    ('RD & DU 2', '0.77')
('IB & DU 2', '0.94')    ('IB & PF 3', '0.86')    ('IB & RD 1', '0.91')    ('IB & PF 2', '0.73')
('IB & RD 1', '0.94')    ('IB & PF 2', '0.86')    ('RD & DU 2', '0.91')    ('IB & DU 2', '0.73')
('IB & PF 3', '0.93')    ('IB & RD 1', '0.85')    ('IB & DU 3', '0.90')    ('RD & PF 3', '0.71')
('RD & DU 3', '0.93')    ('IB & DU 3', '0.85')    ('IB & DU 2', '0.90')    ('IB & DU 1', '0.65')
('IB & DU 3', '0.93')    ('IB & DU 2', '0.85')    ('RD & DU 3', '0.90')    ('PF & DU 1', '0.65')
('PF & DU 3', '0.92')    ('PF & DU 3', '0.84')    ('PF & DU 3', '0.90')    ('IB & PF 1', '0.64')
('PF & DU 2', '0.91')    ('RD & DU 2', '0.83')    ('IB & PF 3', '0.89')    ('IB & DU 3', '0.63')
('RD & DU 1', '0.91')    ('PF & DU 2', '0.83')    ('PF & DU 2', '0.88')    ('IB & PF 3', '0.63')
('IB & PF 1', '0.91')    ('RD & DU 1', '0.83')    ('PF & DU 1', '0.88')    ('RD & DU 1', '0.63')
('RD & DU 2', '0.90')    ('IB & PF 1', '0.81')    ('RD & DU 1', '0.87')    ('RD & DU 3', '0.58')
('IB & DU 1', '0.89')    ('IB & DU 1', '0.81')    ('IB & PF 1', '0.86')    ('PF & DU 3', '0.55')
('PF & DU 1', '0.87')    ('PF & DU 1', '0.80')    ('IB & DU 1', '0.84')    ('PF & DU 2', '0.55')
```

Figure .39: The results from comparison to entry labeled "IB & RD 2".



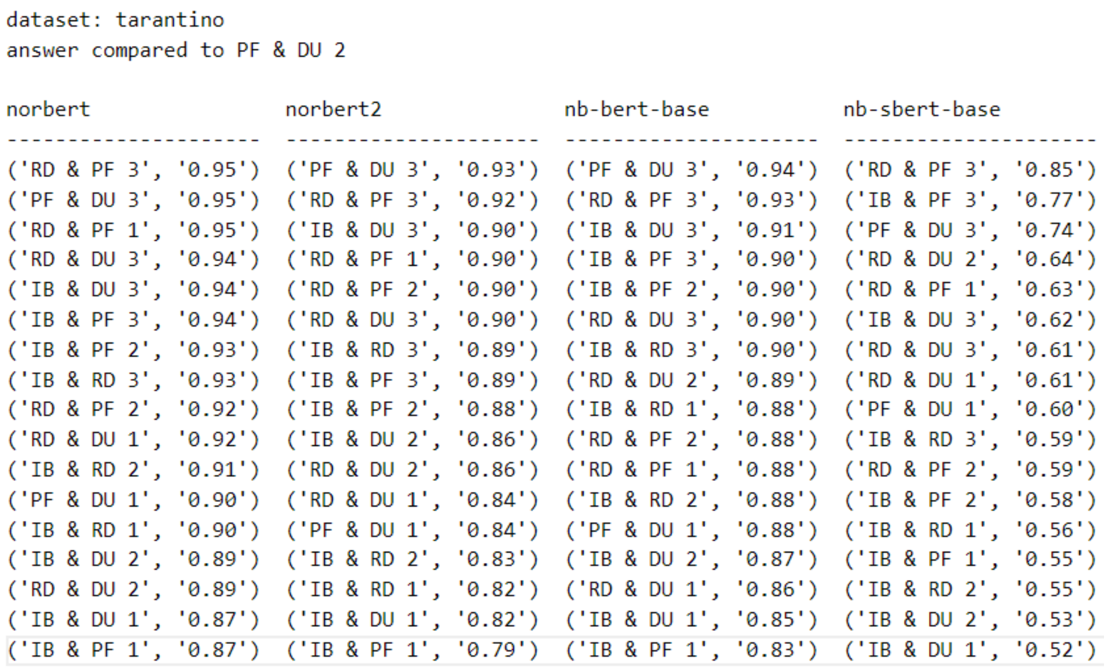
```
dataset: tarantino
answer compared to IB & RD 3

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('IB & RD 2', '0.97')    ('RD & PF 1', '0.94')    ('IB & RD 2', '0.95')    ('IB & RD 2', '0.87')
('RD & PF 1', '0.96')    ('RD & DU 3', '0.94')    ('RD & PF 1', '0.95')    ('RD & PF 2', '0.86')
('RD & PF 2', '0.96')    ('RD & PF 2', '0.93')    ('RD & PF 2', '0.95')    ('RD & PF 1', '0.85')
('IB & RD 1', '0.96')    ('RD & PF 3', '0.93')    ('IB & PF 2', '0.94')    ('RD & DU 2', '0.85')
('RD & PF 3', '0.96')    ('IB & PF 2', '0.92')    ('RD & PF 3', '0.94')    ('IB & RD 1', '0.81')
('IB & PF 2', '0.95')    ('IB & RD 2', '0.92')    ('RD & DU 2', '0.93')    ('RD & PF 3', '0.73')
('PF & DU 3', '0.95')    ('IB & DU 3', '0.92')    ('PF & DU 3', '0.93')    ('IB & PF 2', '0.73')
('RD & DU 2', '0.94')    ('PF & DU 3', '0.91')    ('IB & RD 1', '0.93')    ('PF & DU 1', '0.69')
('RD & DU 3', '0.94')    ('IB & DU 2', '0.91')    ('IB & DU 3', '0.93')    ('IB & PF 3', '0.69')
('IB & DU 2', '0.93')    ('IB & PF 3', '0.91')    ('RD & DU 3', '0.93')    ('IB & DU 2', '0.68')
('IB & DU 3', '0.93')    ('RD & DU 2', '0.90')    ('IB & DU 2', '0.92')    ('RD & DU 1', '0.65')
('IB & PF 3', '0.93')    ('IB & RD 1', '0.90')    ('IB & PF 3', '0.91')    ('IB & DU 1', '0.65')
('RD & DU 1', '0.93')    ('PF & DU 2', '0.89')    ('PF & DU 1', '0.90')    ('IB & PF 1', '0.65')
('PF & DU 2', '0.93')    ('RD & DU 1', '0.88')    ('RD & DU 1', '0.90')    ('RD & DU 3', '0.64')
('IB & PF 1', '0.91')    ('IB & DU 1', '0.87')    ('PF & DU 2', '0.90')    ('IB & DU 3', '0.64')
('IB & DU 1', '0.91')    ('IB & PF 1', '0.84')    ('IB & PF 1', '0.87')    ('PF & DU 3', '0.62')
('PF & DU 1', '0.90')    ('PF & DU 1', '0.83')    ('IB & DU 1', '0.87')    ('PF & DU 2', '0.59')
```

Figure .40: The results from comparison to entry labeled "IB & RD 3".



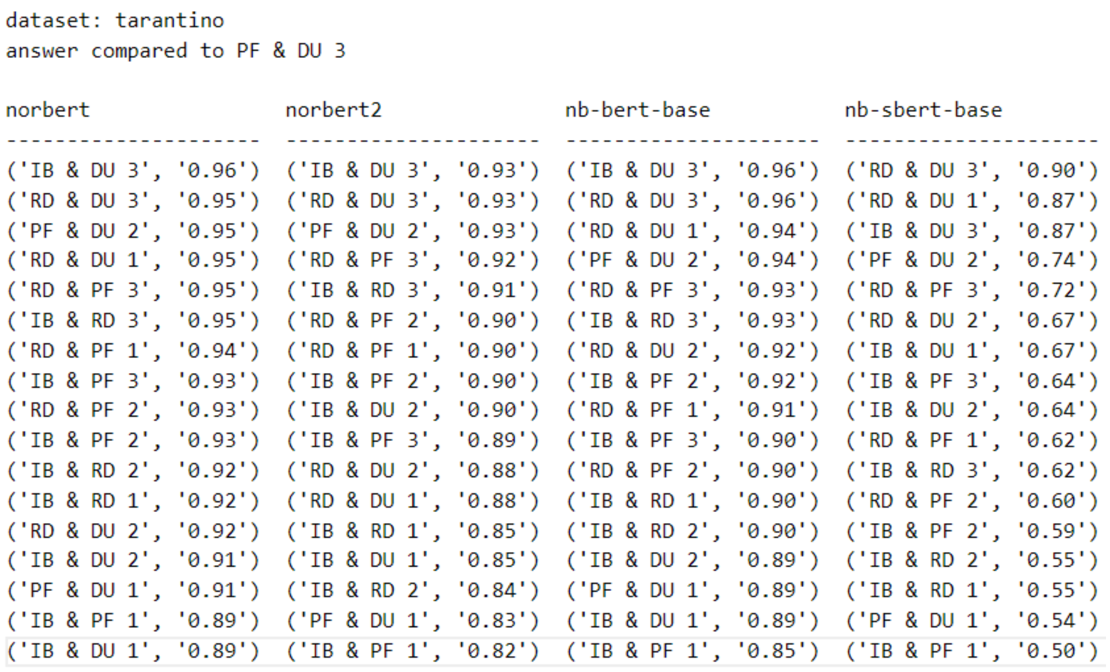
```
dataset: tarantino
answer compared to PF & DU 1

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('PF & DU 3', '0.91')   ('IB & PF 3', '0.85')   ('RD & PF 3', '0.90')   ('IB & PF 3', '0.70')
('IB & PF 3', '0.91')   ('IB & PF 1', '0.85')   ('IB & RD 3', '0.90')   ('RD & PF 3', '0.69')
('PF & DU 2', '0.90')   ('PF & DU 2', '0.84')   ('IB & RD 1', '0.90')   ('IB & RD 3', '0.69')
('RD & PF 2', '0.90')   ('IB & RD 3', '0.83')   ('IB & PF 3', '0.89')   ('RD & PF 2', '0.68')
('IB & RD 3', '0.90')   ('PF & DU 3', '0.83')   ('RD & PF 2', '0.89')   ('RD & DU 2', '0.68')
('RD & PF 1', '0.90')   ('RD & PF 2', '0.83')   ('PF & DU 3', '0.89')   ('IB & RD 1', '0.67')
('RD & DU 1', '0.89')   ('IB & RD 1', '0.82')   ('RD & PF 1', '0.88')   ('IB & RD 2', '0.65')
('RD & PF 3', '0.89')   ('RD & PF 3', '0.82')   ('PF & DU 2', '0.88')   ('RD & PF 1', '0.63')
('IB & RD 1', '0.88')   ('IB & DU 1', '0.82')   ('IB & RD 2', '0.88')   ('PF & DU 2', '0.60')
('IB & PF 1', '0.88')   ('IB & DU 2', '0.82')   ('RD & DU 3', '0.87')   ('IB & PF 2', '0.57')
('IB & DU 3', '0.88')   ('IB & PF 2', '0.81')   ('IB & DU 3', '0.86')   ('RD & DU 1', '0.56')
('RD & DU 3', '0.88')   ('IB & DU 3', '0.80')   ('IB & PF 2', '0.86')   ('IB & DU 3', '0.56')
('IB & RD 2', '0.87')   ('RD & PF 1', '0.80')   ('RD & DU 2', '0.85')   ('RD & DU 3', '0.55')
('IB & PF 2', '0.86')   ('IB & RD 2', '0.80')   ('RD & DU 1', '0.84')   ('PF & DU 3', '0.54')
('IB & DU 1', '0.86')   ('RD & DU 3', '0.79')   ('IB & DU 2', '0.83')   ('IB & PF 1', '0.54')
('RD & DU 2', '0.85')   ('RD & DU 1', '0.77')   ('IB & DU 1', '0.83')   ('IB & DU 1', '0.53')
('IB & DU 2', '0.85')   ('RD & DU 2', '0.77')   ('IB & PF 1', '0.82')   ('IB & DU 2', '0.53')
```

Figure .41: The results from comparison to entry labeled "PF & DU 1".



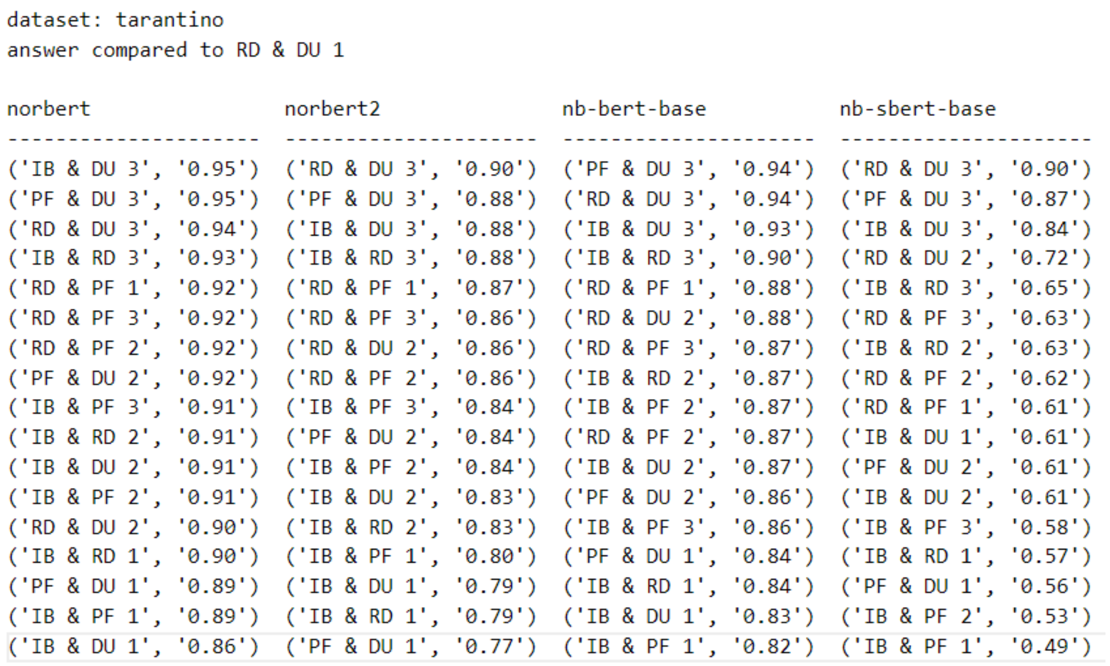
```
dataset: tarantino
answer compared to PF & DU 2

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('RD & PF 3', '0.95')   ('PF & DU 3', '0.93')   ('PF & DU 3', '0.94')   ('RD & PF 3', '0.85')
('PF & DU 3', '0.95')   ('RD & PF 3', '0.92')   ('RD & PF 3', '0.93')   ('IB & PF 3', '0.77')
('RD & PF 1', '0.95')   ('IB & DU 3', '0.90')   ('IB & DU 3', '0.91')   ('PF & DU 3', '0.74')
('RD & DU 3', '0.94')   ('RD & PF 1', '0.90')   ('IB & PF 3', '0.90')   ('RD & DU 2', '0.64')
('IB & DU 3', '0.94')   ('RD & PF 2', '0.90')   ('IB & PF 2', '0.90')   ('RD & PF 1', '0.63')
('IB & PF 3', '0.94')   ('RD & DU 3', '0.90')   ('RD & DU 3', '0.90')   ('IB & DU 3', '0.62')
('IB & PF 2', '0.93')   ('IB & RD 3', '0.89')   ('IB & RD 3', '0.90')   ('RD & DU 3', '0.61')
('IB & RD 3', '0.93')   ('IB & PF 3', '0.89')   ('RD & DU 2', '0.89')   ('RD & DU 1', '0.61')
('RD & PF 2', '0.92')   ('IB & PF 2', '0.88')   ('IB & RD 1', '0.88')   ('PF & DU 1', '0.60')
('RD & DU 1', '0.92')   ('IB & DU 2', '0.86')   ('RD & PF 2', '0.88')   ('IB & RD 3', '0.59')
('IB & RD 2', '0.91')   ('RD & DU 2', '0.86')   ('RD & PF 1', '0.88')   ('RD & PF 2', '0.59')
('PF & DU 1', '0.90')   ('RD & DU 1', '0.84')   ('IB & RD 2', '0.88')   ('IB & PF 2', '0.58')
('IB & RD 1', '0.90')   ('PF & DU 1', '0.84')   ('PF & DU 1', '0.88')   ('IB & RD 1', '0.56')
('IB & DU 2', '0.89')   ('IB & RD 2', '0.83')   ('IB & DU 2', '0.87')   ('IB & PF 1', '0.55')
('RD & DU 2', '0.89')   ('IB & RD 1', '0.82')   ('RD & DU 1', '0.86')   ('IB & RD 2', '0.55')
('IB & DU 1', '0.87')   ('IB & DU 1', '0.82')   ('IB & DU 1', '0.85')   ('IB & DU 2', '0.53')
('IB & PF 1', '0.87')   ('IB & PF 1', '0.79')   ('IB & PF 1', '0.83')   ('IB & DU 1', '0.52')
```

Figure .42: The results from comparison to entry labeled "PF & DU 2".



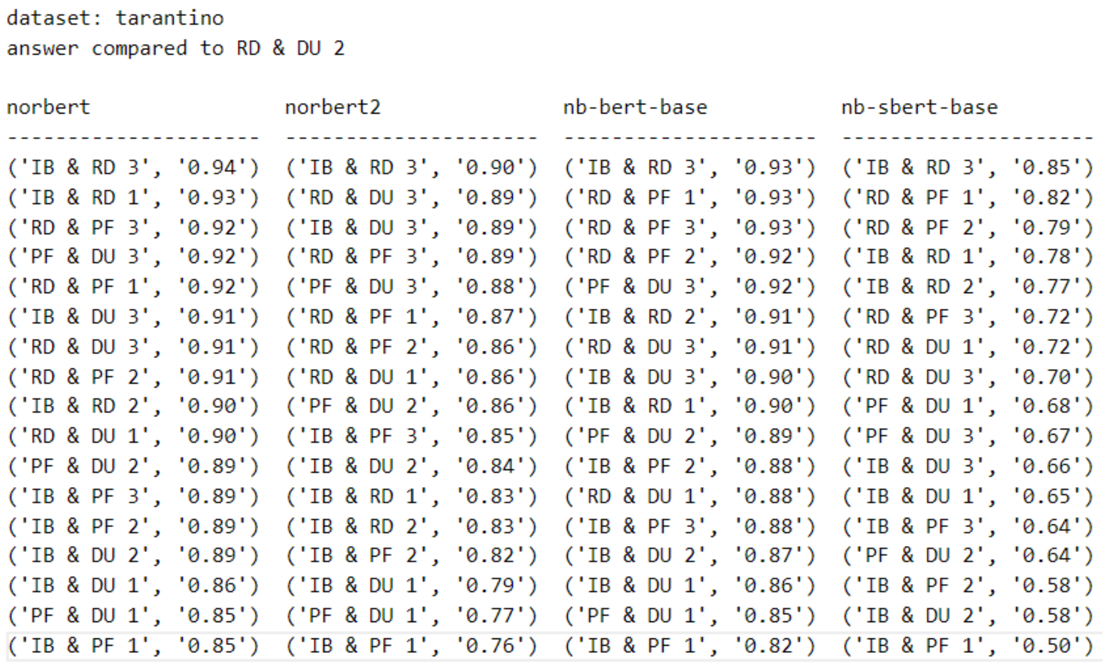
```
dataset: tarantino
answer compared to PF & DU 3

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('IB & DU 3', '0.96')    ('IB & DU 3', '0.93')    ('IB & DU 3', '0.96')    ('RD & DU 3', '0.90')
('RD & DU 3', '0.95')    ('RD & DU 3', '0.93')    ('RD & DU 3', '0.96')    ('RD & DU 1', '0.87')
('PF & DU 2', '0.95')    ('PF & DU 2', '0.93')    ('RD & DU 1', '0.94')    ('IB & DU 3', '0.87')
('RD & DU 1', '0.95')    ('RD & PF 3', '0.92')    ('PF & DU 2', '0.94')    ('PF & DU 2', '0.74')
('RD & PF 3', '0.95')    ('IB & RD 3', '0.91')    ('RD & PF 3', '0.93')    ('RD & PF 3', '0.72')
('IB & RD 3', '0.95')    ('RD & PF 2', '0.90')    ('IB & RD 3', '0.93')    ('RD & DU 2', '0.67')
('RD & PF 1', '0.94')    ('RD & PF 1', '0.90')    ('RD & DU 2', '0.92')    ('IB & DU 1', '0.67')
('IB & PF 3', '0.93')    ('IB & PF 2', '0.90')    ('IB & PF 2', '0.92')    ('IB & PF 3', '0.64')
('RD & PF 2', '0.93')    ('IB & DU 2', '0.90')    ('RD & PF 1', '0.91')    ('IB & DU 2', '0.64')
('IB & PF 2', '0.93')    ('IB & PF 3', '0.89')    ('IB & PF 3', '0.90')    ('RD & PF 1', '0.62')
('IB & RD 2', '0.92')    ('RD & DU 2', '0.88')    ('RD & PF 2', '0.90')    ('IB & RD 3', '0.62')
('IB & RD 1', '0.92')    ('RD & DU 1', '0.88')    ('IB & RD 1', '0.90')    ('RD & PF 2', '0.60')
('RD & DU 2', '0.92')    ('IB & RD 1', '0.85')    ('IB & RD 2', '0.90')    ('IB & PF 2', '0.59')
('IB & DU 2', '0.91')    ('IB & DU 1', '0.85')    ('IB & DU 2', '0.89')    ('IB & RD 2', '0.55')
('PF & DU 1', '0.91')    ('IB & RD 2', '0.84')    ('PF & DU 1', '0.89')    ('IB & RD 1', '0.55')
('IB & PF 1', '0.89')    ('PF & DU 1', '0.83')    ('IB & DU 1', '0.89')    ('PF & DU 1', '0.54')
('IB & DU 1', '0.89')    ('IB & PF 1', '0.82')    ('IB & PF 1', '0.85')    ('IB & PF 1', '0.50')
```

Figure .43: The results from comparison to entry labeled "PF & DU 3".



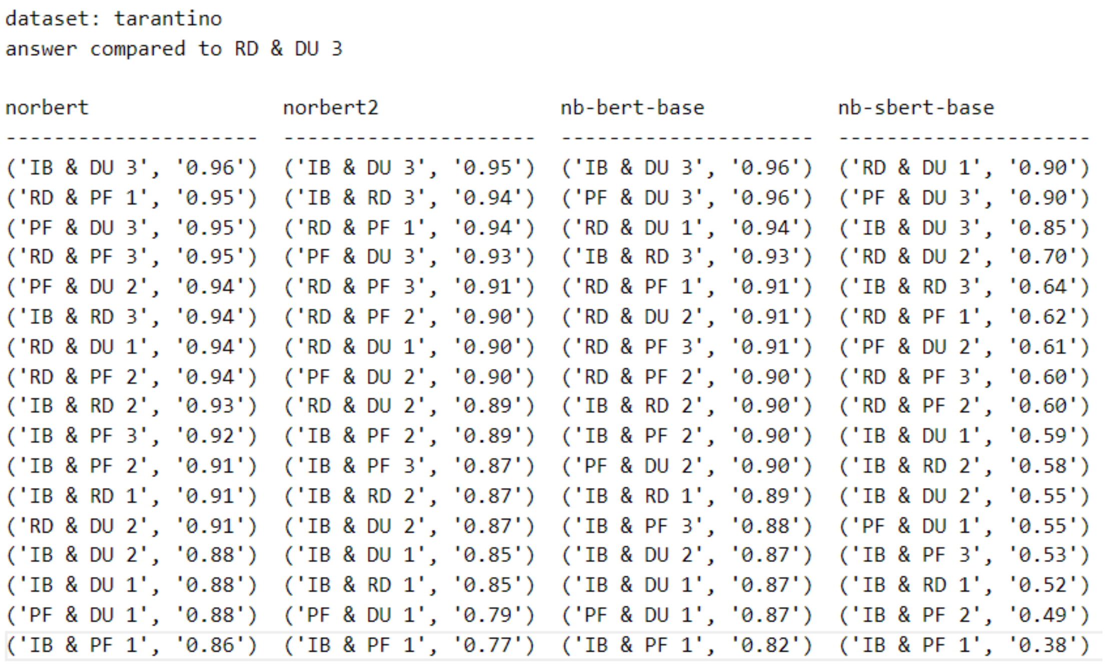
```
dataset: tarantino
answer compared to RD & DU 1

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('IB & DU 3', '0.95')    ('RD & DU 3', '0.90')    ('PF & DU 3', '0.94')    ('RD & DU 3', '0.90')
('PF & DU 3', '0.95')    ('PF & DU 3', '0.88')    ('RD & DU 3', '0.94')    ('PF & DU 3', '0.87')
('RD & DU 3', '0.94')    ('IB & DU 3', '0.88')    ('IB & DU 3', '0.93')    ('IB & DU 3', '0.84')
('IB & RD 3', '0.93')    ('IB & RD 3', '0.88')    ('IB & RD 3', '0.90')    ('RD & DU 2', '0.72')
('RD & PF 1', '0.92')    ('RD & PF 1', '0.87')    ('RD & PF 1', '0.88')    ('IB & RD 3', '0.65')
('RD & PF 3', '0.92')    ('RD & PF 3', '0.86')    ('RD & DU 2', '0.88')    ('RD & PF 3', '0.63')
('RD & PF 2', '0.92')    ('RD & DU 2', '0.86')    ('RD & PF 3', '0.87')    ('IB & RD 2', '0.63')
('PF & DU 2', '0.92')    ('RD & PF 2', '0.86')    ('IB & RD 2', '0.87')    ('RD & PF 2', '0.62')
('IB & PF 3', '0.91')    ('IB & PF 3', '0.84')    ('IB & PF 2', '0.87')    ('RD & PF 1', '0.61')
('IB & RD 2', '0.91')    ('PF & DU 2', '0.84')    ('RD & PF 2', '0.87')    ('IB & DU 1', '0.61')
('IB & DU 2', '0.91')    ('IB & PF 2', '0.84')    ('IB & DU 2', '0.87')    ('PF & DU 2', '0.61')
('IB & PF 2', '0.91')    ('IB & DU 2', '0.83')    ('PF & DU 2', '0.86')    ('IB & DU 2', '0.61')
('RD & DU 2', '0.90')    ('IB & RD 2', '0.83')    ('IB & PF 3', '0.86')    ('IB & PF 3', '0.58')
('IB & RD 1', '0.90')    ('IB & PF 1', '0.80')    ('PF & DU 1', '0.84')    ('IB & RD 1', '0.57')
('PF & DU 1', '0.89')    ('IB & DU 1', '0.79')    ('IB & RD 1', '0.84')    ('PF & DU 1', '0.56')
('IB & PF 1', '0.89')    ('IB & RD 1', '0.79')    ('IB & DU 1', '0.83')    ('IB & PF 2', '0.53')
('IB & DU 1', '0.86')    ('PF & DU 1', '0.77')    ('IB & PF 1', '0.82')    ('IB & PF 1', '0.49')
```

Figure .44: The results from comparison to entry labeled "RD & DU 1".



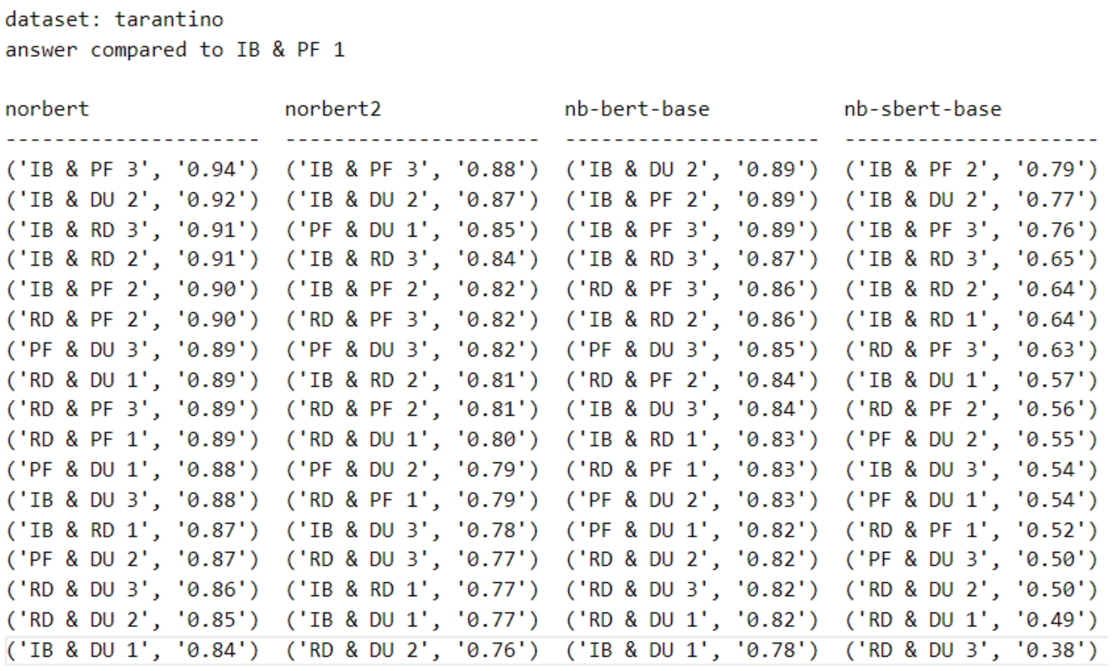
```
dataset: tarantino
answer compared to RD & DU 2

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & RD 3', '0.94')   ('IB & RD 3', '0.90')   ('IB & RD 3', '0.93')   ('IB & RD 3', '0.85')
('IB & RD 1', '0.93')   ('RD & DU 3', '0.89')   ('RD & PF 1', '0.93')   ('RD & PF 1', '0.82')
('RD & PF 3', '0.92')   ('IB & DU 3', '0.89')   ('RD & PF 3', '0.93')   ('RD & PF 2', '0.79')
('PF & DU 3', '0.92')   ('RD & PF 3', '0.89')   ('RD & PF 2', '0.92')   ('IB & RD 1', '0.78')
('RD & PF 1', '0.92')   ('PF & DU 3', '0.88')   ('PF & DU 3', '0.92')   ('IB & RD 2', '0.77')
('IB & DU 3', '0.91')   ('RD & PF 1', '0.87')   ('IB & RD 2', '0.91')   ('RD & PF 3', '0.72')
('RD & DU 3', '0.91')   ('RD & PF 2', '0.86')   ('RD & DU 3', '0.91')   ('RD & DU 1', '0.72')
('RD & PF 2', '0.91')   ('RD & DU 1', '0.86')   ('IB & DU 3', '0.90')   ('RD & DU 3', '0.70')
('IB & RD 2', '0.90')   ('PF & DU 2', '0.86')   ('IB & RD 1', '0.90')   ('PF & DU 1', '0.68')
('RD & DU 1', '0.90')   ('IB & PF 3', '0.85')   ('PF & DU 2', '0.89')   ('PF & DU 3', '0.67')
('PF & DU 2', '0.89')   ('IB & DU 2', '0.84')   ('IB & PF 2', '0.88')   ('IB & DU 3', '0.66')
('IB & PF 3', '0.89')   ('IB & RD 1', '0.83')   ('RD & DU 1', '0.88')   ('IB & DU 1', '0.65')
('IB & PF 2', '0.89')   ('IB & RD 2', '0.83')   ('IB & PF 3', '0.88')   ('IB & PF 3', '0.64')
('IB & DU 2', '0.89')   ('IB & PF 2', '0.82')   ('IB & DU 2', '0.87')   ('PF & DU 2', '0.64')
('IB & DU 1', '0.86')   ('IB & DU 1', '0.79')   ('IB & DU 1', '0.86')   ('IB & PF 2', '0.58')
('PF & DU 1', '0.85')   ('PF & DU 1', '0.77')   ('PF & DU 1', '0.85')   ('IB & DU 2', '0.58')
('IB & PF 1', '0.85')   ('IB & PF 1', '0.76')   ('IB & PF 1', '0.82')   ('IB & PF 1', '0.50')
```

Figure .45: The results from comparison to entry labeled "RD & DU 2".



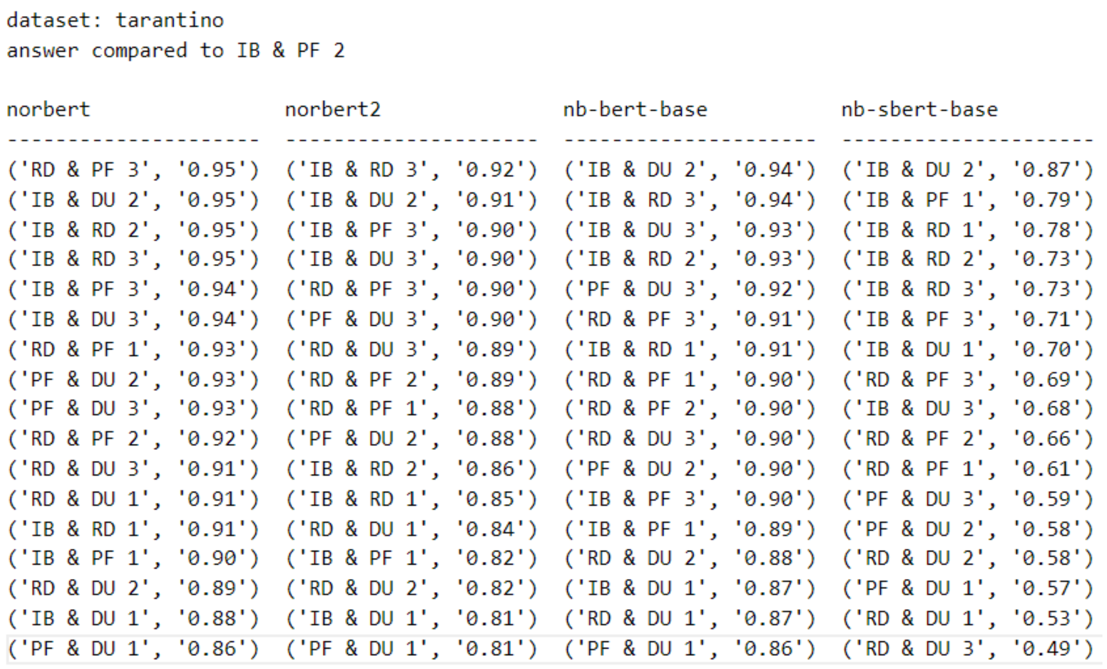
```
dataset: tarantino
answer compared to RD & DU 3

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & DU 3', '0.96')   ('IB & DU 3', '0.95')   ('IB & DU 3', '0.96')   ('RD & DU 1', '0.90')
('RD & PF 1', '0.95')   ('IB & RD 3', '0.94')   ('PF & DU 3', '0.96')   ('PF & DU 3', '0.90')
('PF & DU 3', '0.95')   ('RD & PF 1', '0.94')   ('RD & DU 1', '0.94')   ('IB & DU 3', '0.85')
('RD & PF 3', '0.95')   ('PF & DU 3', '0.93')   ('IB & RD 3', '0.93')   ('RD & DU 2', '0.70')
('PF & DU 2', '0.94')   ('RD & PF 3', '0.91')   ('RD & PF 1', '0.91')   ('IB & RD 3', '0.64')
('IB & RD 3', '0.94')   ('RD & PF 2', '0.90')   ('RD & DU 2', '0.91')   ('RD & PF 1', '0.62')
('RD & DU 1', '0.94')   ('RD & DU 1', '0.90')   ('RD & PF 3', '0.91')   ('PF & DU 2', '0.61')
('RD & PF 2', '0.94')   ('PF & DU 2', '0.90')   ('RD & PF 2', '0.90')   ('RD & PF 3', '0.60')
('IB & RD 2', '0.93')   ('RD & DU 2', '0.89')   ('IB & RD 2', '0.90')   ('RD & PF 2', '0.60')
('IB & PF 3', '0.92')   ('IB & PF 2', '0.89')   ('IB & PF 2', '0.90')   ('IB & DU 1', '0.59')
('IB & PF 2', '0.91')   ('IB & PF 3', '0.87')   ('PF & DU 2', '0.90')   ('IB & RD 2', '0.58')
('IB & RD 1', '0.91')   ('IB & RD 2', '0.87')   ('IB & RD 1', '0.89')   ('IB & DU 2', '0.55')
('RD & DU 2', '0.91')   ('IB & DU 2', '0.87')   ('IB & PF 3', '0.88')   ('PF & DU 1', '0.55')
('IB & DU 2', '0.88')   ('IB & DU 1', '0.85')   ('IB & DU 2', '0.87')   ('IB & PF 3', '0.53')
('IB & DU 1', '0.88')   ('IB & RD 1', '0.85')   ('IB & DU 1', '0.87')   ('IB & RD 1', '0.52')
('PF & DU 1', '0.88')   ('PF & DU 1', '0.79')   ('PF & DU 1', '0.87')   ('IB & PF 2', '0.49')
('IB & PF 1', '0.86')   ('IB & PF 1', '0.77')   ('IB & PF 1', '0.82')   ('IB & PF 1', '0.38')
```

Figure .46: The results from comparison to entry labeled "RD & DU 3".



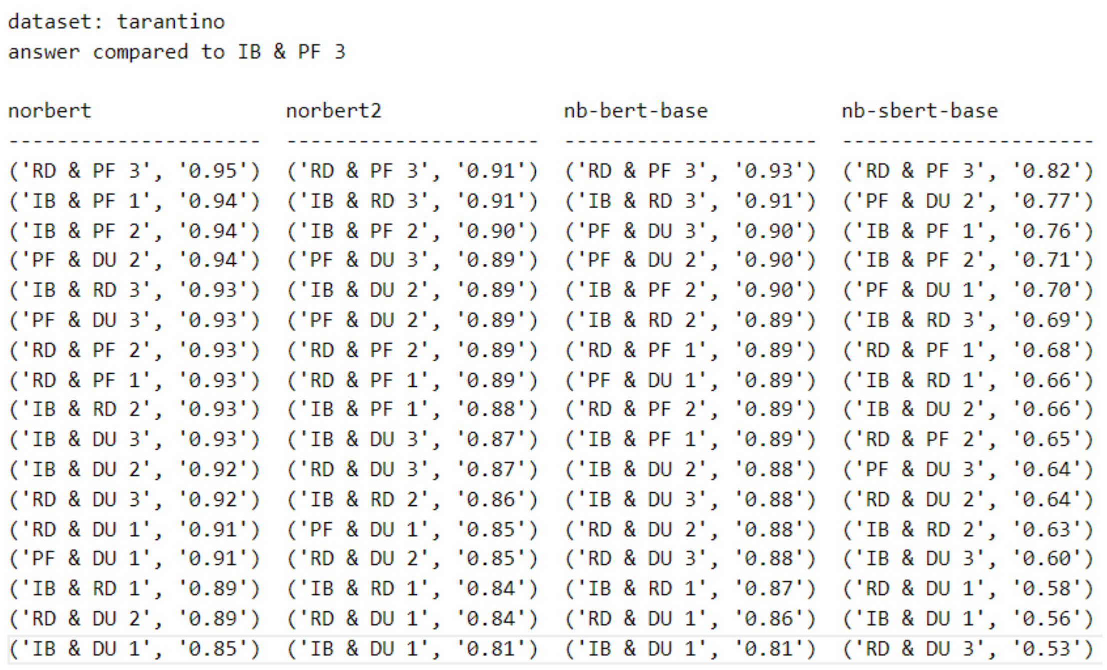
```
dataset: tarantino
answer compared to IB & PF 1

norbert                norbert2               nb-bert-base           nb-sbert-base
---------------------  ---------------------  ---------------------  ---------------------
('IB & PF 3', '0.94')  ('IB & PF 3', '0.88')  ('IB & DU 2', '0.89')  ('IB & PF 2', '0.79')
('IB & DU 2', '0.92')  ('IB & DU 2', '0.87')  ('IB & PF 2', '0.89')  ('IB & DU 2', '0.77')
('IB & RD 3', '0.91')  ('PF & DU 1', '0.85')  ('IB & PF 3', '0.89')  ('IB & PF 3', '0.76')
('IB & RD 2', '0.91')  ('IB & RD 3', '0.84')  ('IB & RD 3', '0.87')  ('IB & RD 3', '0.65')
('IB & PF 2', '0.90')  ('IB & PF 2', '0.82')  ('RD & PF 3', '0.86')  ('IB & RD 2', '0.64')
('RD & PF 2', '0.90')  ('RD & PF 3', '0.82')  ('IB & RD 2', '0.86')  ('IB & RD 1', '0.64')
('PF & DU 3', '0.89')  ('PF & DU 3', '0.82')  ('PF & DU 3', '0.85')  ('RD & PF 3', '0.63')
('RD & DU 1', '0.89')  ('IB & RD 2', '0.81')  ('RD & PF 2', '0.84')  ('IB & DU 1', '0.57')
('RD & PF 3', '0.89')  ('RD & PF 2', '0.81')  ('IB & DU 3', '0.84')  ('RD & PF 2', '0.56')
('RD & PF 1', '0.89')  ('RD & DU 1', '0.80')  ('IB & RD 1', '0.83')  ('PF & DU 2', '0.55')
('PF & DU 1', '0.88')  ('PF & DU 2', '0.79')  ('RD & PF 1', '0.83')  ('IB & DU 3', '0.54')
('IB & DU 3', '0.88')  ('RD & PF 1', '0.79')  ('PF & DU 2', '0.83')  ('PF & DU 1', '0.54')
('IB & RD 1', '0.87')  ('IB & DU 3', '0.78')  ('PF & DU 1', '0.82')  ('RD & PF 1', '0.52')
('PF & DU 2', '0.87')  ('RD & DU 3', '0.77')  ('RD & DU 2', '0.82')  ('PF & DU 3', '0.50')
('RD & DU 3', '0.86')  ('IB & RD 1', '0.77')  ('RD & DU 3', '0.82')  ('RD & DU 2', '0.50')
('RD & DU 2', '0.85')  ('IB & DU 1', '0.77')  ('RD & DU 1', '0.82')  ('RD & DU 1', '0.49')
('IB & DU 1', '0.84')  ('RD & DU 2', '0.76')  ('IB & DU 1', '0.78')  ('RD & DU 3', '0.38')
```

Figure .47: The results from comparison to entry labeled "IB & PF 1".



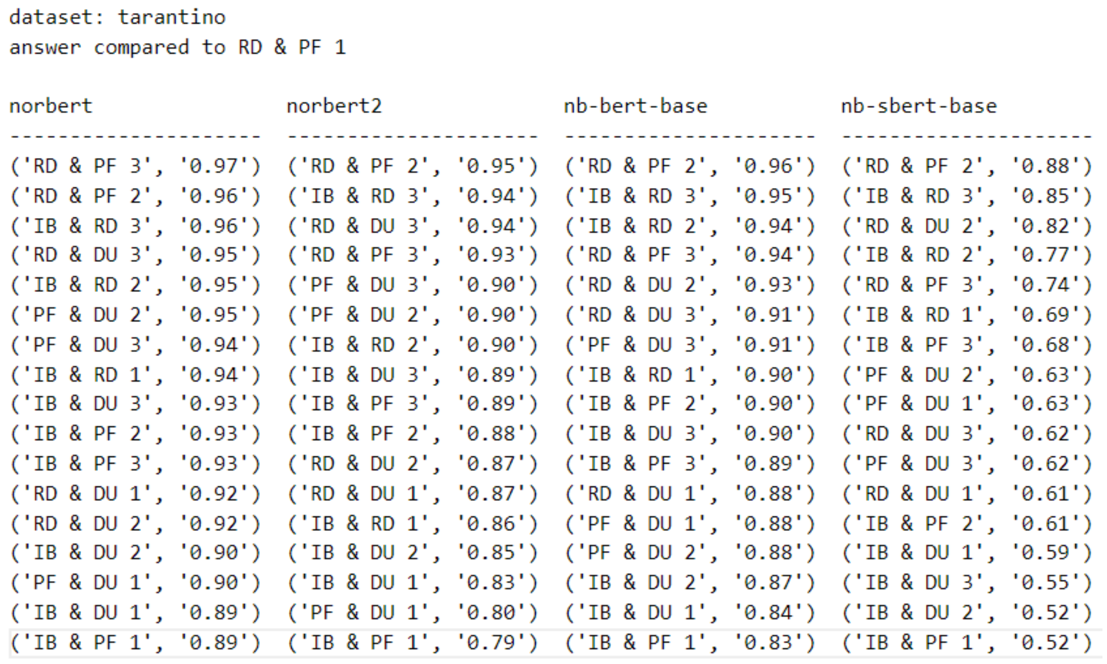
```
dataset: tarantino
answer compared to IB & PF 2

norbert                norbert2               nb-bert-base           nb-sbert-base
---------------------  ---------------------  ---------------------  ---------------------
('RD & PF 3', '0.95')  ('IB & RD 3', '0.92')  ('IB & DU 2', '0.94')  ('IB & DU 2', '0.87')
('IB & DU 2', '0.95')  ('IB & DU 2', '0.91')  ('IB & RD 3', '0.94')  ('IB & PF 1', '0.79')
('IB & RD 2', '0.95')  ('IB & PF 3', '0.90')  ('IB & DU 3', '0.93')  ('IB & RD 1', '0.78')
('IB & RD 3', '0.95')  ('IB & DU 3', '0.90')  ('IB & RD 2', '0.93')  ('IB & RD 2', '0.73')
('IB & PF 3', '0.94')  ('RD & PF 3', '0.90')  ('PF & DU 3', '0.92')  ('IB & RD 3', '0.73')
('IB & DU 3', '0.94')  ('PF & DU 3', '0.90')  ('RD & PF 3', '0.91')  ('IB & PF 3', '0.71')
('RD & PF 1', '0.93')  ('RD & DU 3', '0.89')  ('IB & RD 1', '0.91')  ('IB & DU 1', '0.70')
('PF & DU 2', '0.93')  ('RD & PF 2', '0.89')  ('RD & PF 1', '0.90')  ('RD & PF 3', '0.69')
('PF & DU 3', '0.93')  ('RD & PF 1', '0.88')  ('RD & PF 2', '0.90')  ('IB & DU 3', '0.68')
('RD & PF 2', '0.92')  ('PF & DU 2', '0.88')  ('RD & DU 3', '0.90')  ('RD & PF 2', '0.66')
('RD & DU 3', '0.91')  ('IB & RD 2', '0.86')  ('PF & DU 2', '0.90')  ('RD & PF 1', '0.61')
('RD & DU 1', '0.91')  ('IB & RD 1', '0.85')  ('IB & PF 3', '0.90')  ('PF & DU 3', '0.59')
('IB & RD 1', '0.91')  ('RD & DU 1', '0.84')  ('IB & PF 1', '0.89')  ('PF & DU 2', '0.58')
('IB & PF 1', '0.90')  ('IB & PF 1', '0.82')  ('RD & DU 2', '0.88')  ('RD & DU 2', '0.58')
('RD & DU 2', '0.89')  ('RD & DU 2', '0.82')  ('IB & DU 1', '0.87')  ('PF & DU 1', '0.57')
('IB & DU 1', '0.88')  ('IB & DU 1', '0.81')  ('RD & DU 1', '0.87')  ('RD & DU 1', '0.53')
('PF & DU 1', '0.86')  ('PF & DU 1', '0.81')  ('PF & DU 1', '0.86')  ('RD & DU 3', '0.49')
```

Figure .48: The results from comparison to entry labeled "IB & PF 2".



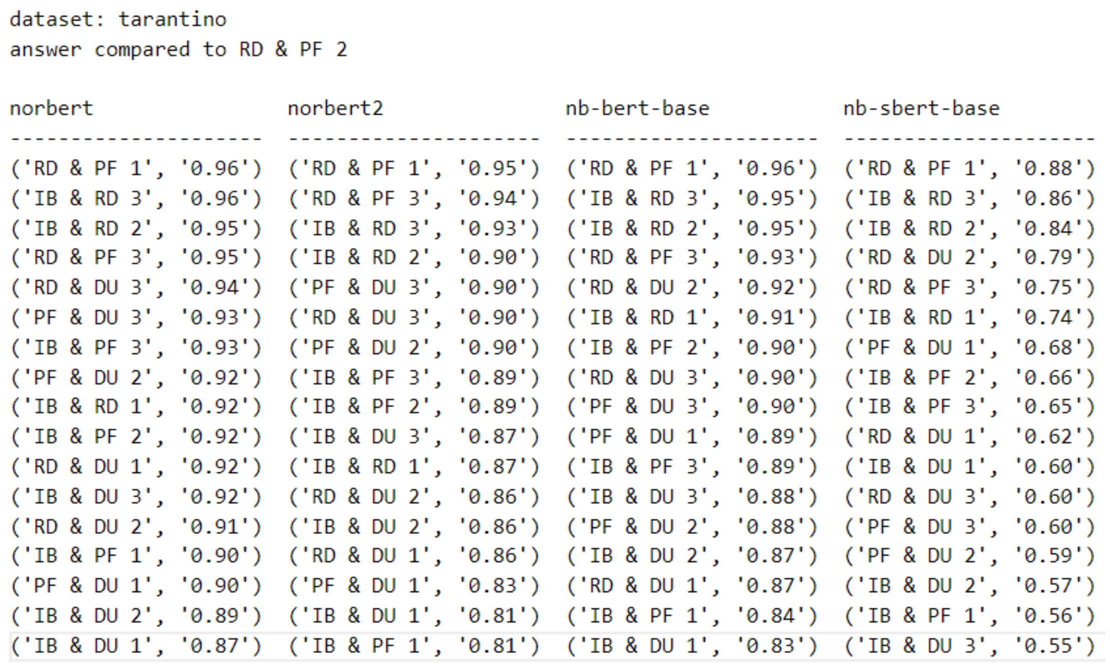
```
dataset: tarantino
answer compared to IB & PF 3

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('RD & PF 3', '0.95')    ('RD & PF 3', '0.91')    ('RD & PF 3', '0.93')    ('RD & PF 3', '0.82')
('IB & PF 1', '0.94')    ('IB & RD 3', '0.91')    ('IB & RD 3', '0.91')    ('PF & DU 2', '0.77')
('IB & PF 2', '0.94')    ('IB & PF 2', '0.90')    ('PF & DU 3', '0.90')    ('IB & PF 1', '0.76')
('PF & DU 2', '0.94')    ('PF & DU 3', '0.89')    ('PF & DU 2', '0.90')    ('IB & PF 2', '0.71')
('IB & RD 3', '0.93')    ('IB & DU 2', '0.89')    ('IB & PF 2', '0.90')    ('PF & DU 1', '0.70')
('PF & DU 3', '0.93')    ('PF & DU 2', '0.89')    ('IB & RD 2', '0.89')    ('IB & RD 3', '0.69')
('RD & PF 2', '0.93')    ('RD & PF 2', '0.89')    ('RD & PF 1', '0.89')    ('RD & PF 1', '0.68')
('RD & PF 1', '0.93')    ('RD & PF 1', '0.89')    ('PF & DU 1', '0.89')    ('IB & RD 1', '0.66')
('IB & RD 2', '0.93')    ('IB & PF 1', '0.88')    ('RD & PF 2', '0.89')    ('IB & DU 2', '0.66')
('IB & DU 3', '0.93')    ('IB & DU 3', '0.87')    ('IB & PF 1', '0.89')    ('RD & PF 2', '0.65')
('IB & DU 2', '0.92')    ('RD & DU 3', '0.87')    ('IB & DU 2', '0.88')    ('PF & DU 3', '0.64')
('RD & DU 3', '0.92')    ('IB & RD 2', '0.86')    ('IB & DU 3', '0.88')    ('RD & DU 2', '0.64')
('RD & DU 1', '0.91')    ('PF & DU 1', '0.85')    ('RD & DU 2', '0.88')    ('IB & RD 2', '0.63')
('PF & DU 1', '0.91')    ('RD & DU 2', '0.85')    ('RD & DU 3', '0.88')    ('IB & DU 3', '0.60')
('IB & RD 1', '0.89')    ('IB & RD 1', '0.84')    ('IB & RD 1', '0.87')    ('RD & DU 1', '0.58')
('RD & DU 2', '0.89')    ('RD & DU 1', '0.84')    ('RD & DU 1', '0.86')    ('IB & DU 1', '0.56')
('IB & DU 1', '0.85')    ('IB & DU 1', '0.81')    ('IB & DU 1', '0.81')    ('RD & DU 3', '0.53')
```

Figure .49: The results from comparison to entry labeled "IB & PF 3".



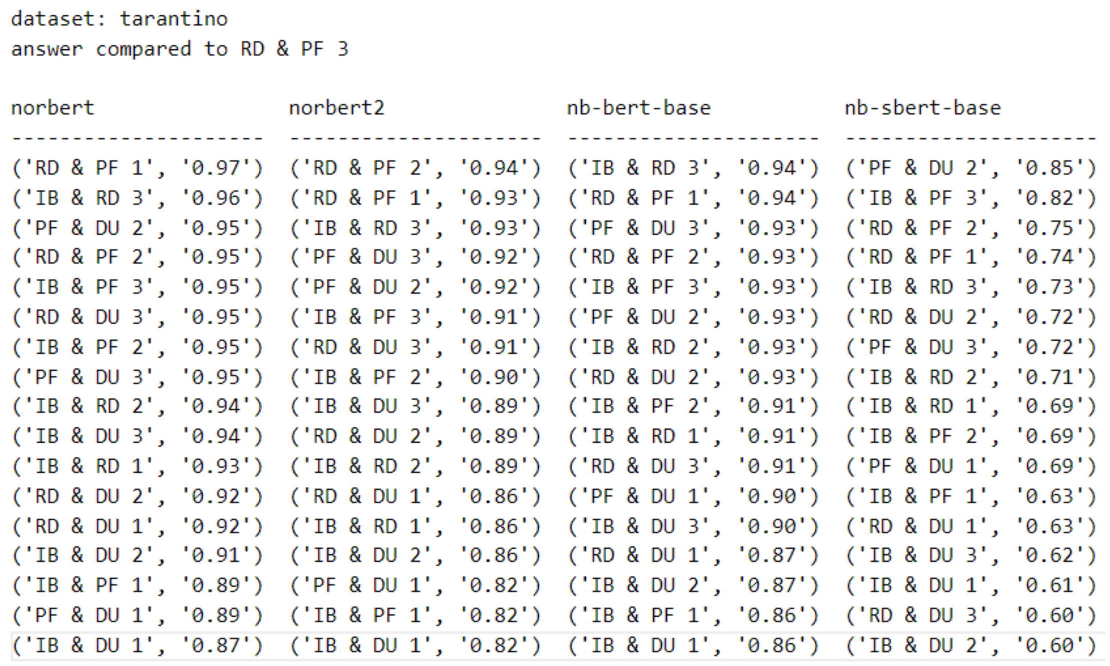
```
dataset: tarantino
answer compared to RD & PF 1

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('RD & PF 3', '0.97')    ('RD & PF 2', '0.95')    ('RD & PF 2', '0.96')    ('RD & PF 2', '0.88')
('RD & PF 2', '0.96')    ('IB & RD 3', '0.94')    ('IB & RD 3', '0.95')    ('IB & RD 3', '0.85')
('IB & RD 3', '0.96')    ('RD & DU 3', '0.94')    ('IB & RD 2', '0.94')    ('RD & DU 2', '0.82')
('RD & DU 3', '0.95')    ('RD & PF 3', '0.93')    ('RD & PF 3', '0.94')    ('IB & RD 2', '0.77')
('IB & RD 2', '0.95')    ('PF & DU 3', '0.90')    ('RD & DU 2', '0.93')    ('RD & PF 3', '0.74')
('PF & DU 2', '0.95')    ('PF & DU 2', '0.90')    ('RD & DU 3', '0.91')    ('IB & RD 1', '0.69')
('PF & DU 3', '0.94')    ('IB & RD 2', '0.90')    ('PF & DU 3', '0.91')    ('IB & PF 3', '0.68')
('IB & RD 1', '0.94')    ('IB & DU 3', '0.89')    ('IB & RD 1', '0.90')    ('PF & DU 2', '0.63')
('IB & DU 3', '0.93')    ('IB & PF 3', '0.89')    ('IB & PF 2', '0.90')    ('PF & DU 1', '0.63')
('IB & PF 2', '0.93')    ('IB & PF 2', '0.88')    ('IB & DU 3', '0.90')    ('RD & DU 3', '0.62')
('IB & PF 3', '0.93')    ('RD & DU 2', '0.87')    ('IB & PF 3', '0.89')    ('PF & DU 3', '0.62')
('RD & DU 1', '0.92')    ('RD & DU 1', '0.87')    ('RD & DU 1', '0.88')    ('RD & DU 1', '0.61')
('RD & DU 2', '0.92')    ('IB & RD 1', '0.86')    ('PF & DU 1', '0.88')    ('IB & PF 2', '0.61')
('IB & DU 2', '0.90')    ('IB & DU 2', '0.85')    ('PF & DU 2', '0.88')    ('IB & DU 1', '0.59')
('PF & DU 1', '0.90')    ('IB & DU 1', '0.83')    ('IB & DU 2', '0.87')    ('IB & DU 3', '0.55')
('IB & DU 1', '0.89')    ('PF & DU 1', '0.80')    ('IB & DU 1', '0.84')    ('IB & DU 2', '0.52')
('IB & PF 1', '0.89')    ('IB & PF 1', '0.79')    ('IB & PF 1', '0.83')    ('IB & PF 1', '0.52')
```

Figure .50: The results from comparison to entry labeled "RD & PF 1".



```
dataset: tarantino
answer compared to RD & PF 2

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('RD & PF 1', '0.96')   ('RD & PF 1', '0.95')   ('RD & PF 1', '0.96')   ('RD & PF 1', '0.88')
('IB & RD 3', '0.96')   ('RD & PF 3', '0.94')   ('IB & RD 3', '0.95')   ('IB & RD 3', '0.86')
('IB & RD 2', '0.95')   ('IB & RD 3', '0.93')   ('IB & RD 2', '0.95')   ('IB & RD 2', '0.84')
('RD & PF 3', '0.95')   ('IB & RD 2', '0.90')   ('RD & PF 3', '0.93')   ('RD & DU 2', '0.79')
('RD & DU 3', '0.94')   ('PF & DU 3', '0.90')   ('RD & DU 2', '0.92')   ('RD & PF 3', '0.75')
('PF & DU 3', '0.93')   ('RD & DU 3', '0.90')   ('IB & RD 1', '0.91')   ('IB & RD 1', '0.74')
('IB & PF 3', '0.93')   ('PF & DU 2', '0.90')   ('IB & PF 2', '0.90')   ('PF & DU 1', '0.68')
('PF & DU 2', '0.92')   ('IB & PF 3', '0.89')   ('RD & DU 3', '0.90')   ('IB & PF 2', '0.66')
('IB & RD 1', '0.92')   ('IB & PF 2', '0.89')   ('PF & DU 3', '0.90')   ('IB & PF 3', '0.65')
('IB & PF 2', '0.92')   ('IB & DU 3', '0.87')   ('PF & DU 1', '0.89')   ('RD & DU 1', '0.62')
('RD & DU 1', '0.92')   ('IB & RD 1', '0.87')   ('IB & PF 3', '0.89')   ('IB & DU 1', '0.60')
('IB & DU 3', '0.92')   ('RD & DU 2', '0.86')   ('IB & DU 3', '0.88')   ('RD & DU 3', '0.60')
('RD & DU 2', '0.91')   ('IB & DU 2', '0.86')   ('PF & DU 2', '0.88')   ('PF & DU 3', '0.60')
('IB & PF 1', '0.90')   ('RD & DU 1', '0.86')   ('IB & DU 2', '0.87')   ('PF & DU 2', '0.59')
('PF & DU 1', '0.90')   ('PF & DU 1', '0.83')   ('RD & DU 1', '0.87')   ('IB & DU 2', '0.57')
('IB & DU 2', '0.89')   ('IB & DU 1', '0.81')   ('IB & PF 1', '0.84')   ('IB & PF 1', '0.56')
('IB & DU 1', '0.87')   ('IB & PF 1', '0.81')   ('IB & DU 1', '0.83')   ('IB & DU 3', '0.55')
```

Figure .51: The results from comparison to entry labeled "RD & PF 2".

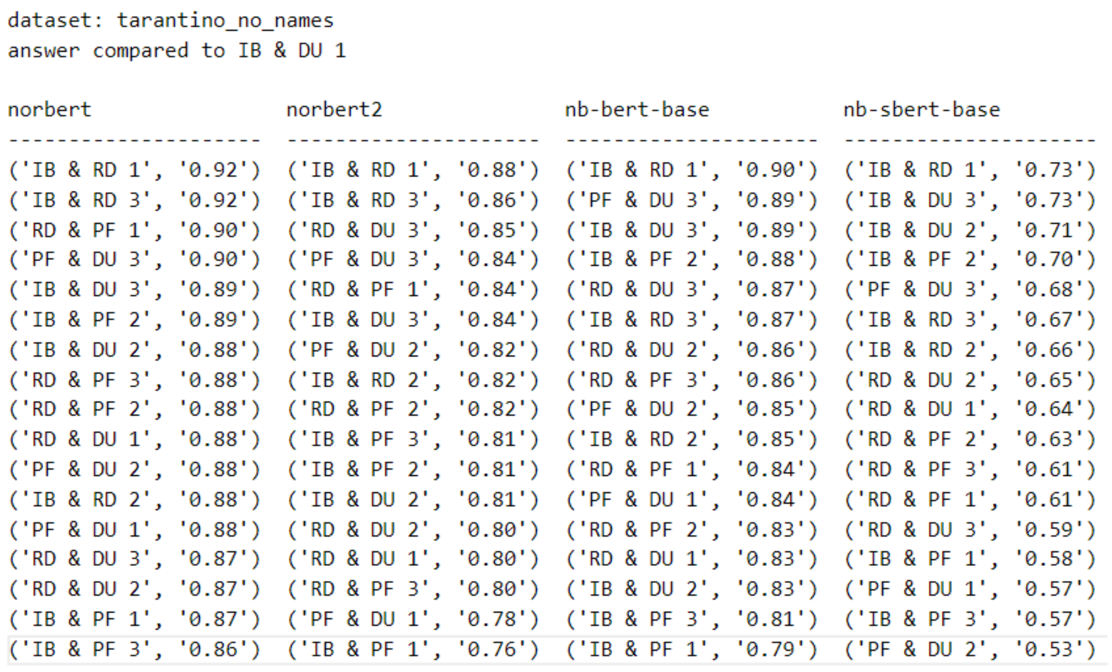


```
dataset: tarantino
answer compared to RD & PF 3

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('RD & PF 1', '0.97')   ('RD & PF 2', '0.94')   ('IB & RD 3', '0.94')   ('PF & DU 2', '0.85')
('IB & RD 3', '0.96')   ('RD & PF 1', '0.93')   ('RD & PF 1', '0.94')   ('IB & PF 3', '0.82')
('PF & DU 2', '0.95')   ('IB & RD 3', '0.93')   ('PF & DU 3', '0.93')   ('RD & PF 2', '0.75')
('RD & PF 2', '0.95')   ('PF & DU 3', '0.92')   ('RD & PF 2', '0.93')   ('RD & PF 1', '0.74')
('IB & PF 3', '0.95')   ('PF & DU 2', '0.92')   ('IB & PF 3', '0.93')   ('IB & RD 3', '0.73')
('RD & DU 3', '0.95')   ('IB & PF 3', '0.91')   ('PF & DU 2', '0.93')   ('RD & DU 2', '0.72')
('IB & PF 2', '0.95')   ('RD & DU 3', '0.91')   ('IB & RD 2', '0.93')   ('PF & DU 3', '0.72')
('PF & DU 3', '0.95')   ('IB & PF 2', '0.90')   ('RD & DU 2', '0.93')   ('IB & RD 2', '0.71')
('IB & RD 2', '0.94')   ('IB & DU 3', '0.89')   ('IB & PF 2', '0.91')   ('IB & RD 1', '0.69')
('IB & DU 3', '0.94')   ('RD & DU 2', '0.89')   ('IB & RD 1', '0.91')   ('IB & PF 2', '0.69')
('IB & RD 1', '0.93')   ('IB & RD 2', '0.89')   ('RD & DU 3', '0.91')   ('PF & DU 1', '0.69')
('RD & DU 2', '0.92')   ('RD & DU 1', '0.86')   ('PF & DU 1', '0.90')   ('IB & PF 1', '0.63')
('RD & DU 1', '0.92')   ('IB & RD 1', '0.86')   ('IB & DU 3', '0.90')   ('RD & DU 1', '0.63')
('IB & DU 2', '0.91')   ('IB & DU 2', '0.86')   ('RD & DU 1', '0.87')   ('IB & DU 3', '0.62')
('IB & PF 1', '0.89')   ('PF & DU 1', '0.82')   ('IB & DU 2', '0.87')   ('IB & DU 1', '0.61')
('PF & DU 1', '0.89')   ('IB & PF 1', '0.82')   ('IB & PF 1', '0.86')   ('RD & DU 3', '0.60')
('IB & DU 1', '0.87')   ('IB & DU 1', '0.82')   ('IB & DU 1', '0.86')   ('IB & DU 2', '0.60')
```

Figure .52: The results from comparison to entry labeled "RD & PF 3".

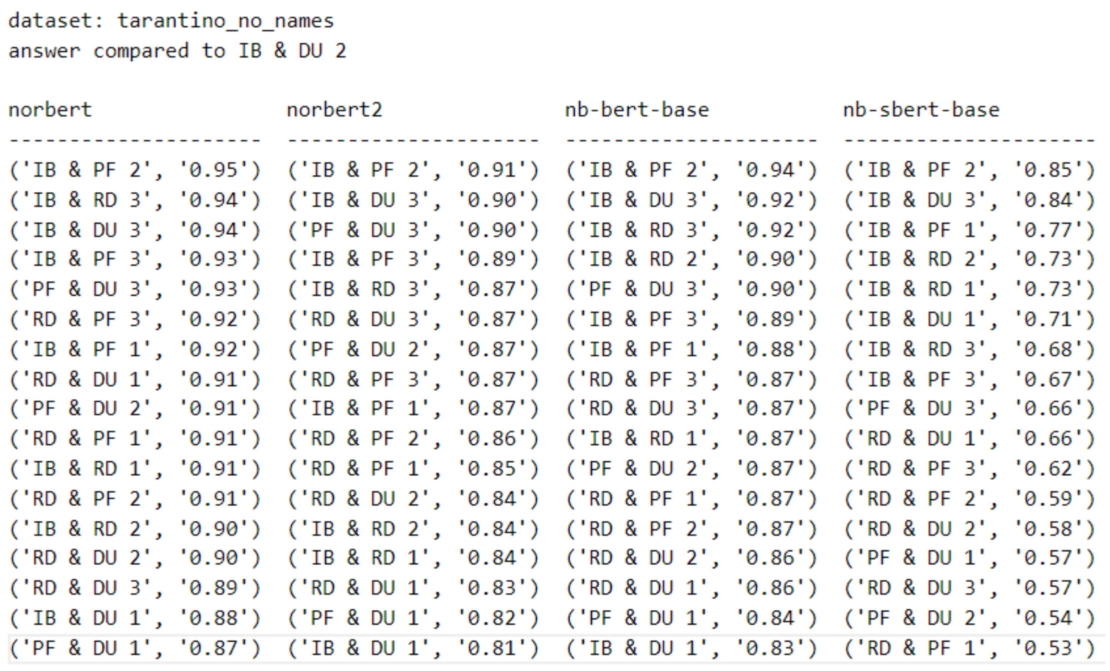
B3 - Tarantino without titles dataset

All results from the tests performed on the Tarantino dataset with two movie descriptions per entry and titles removed.



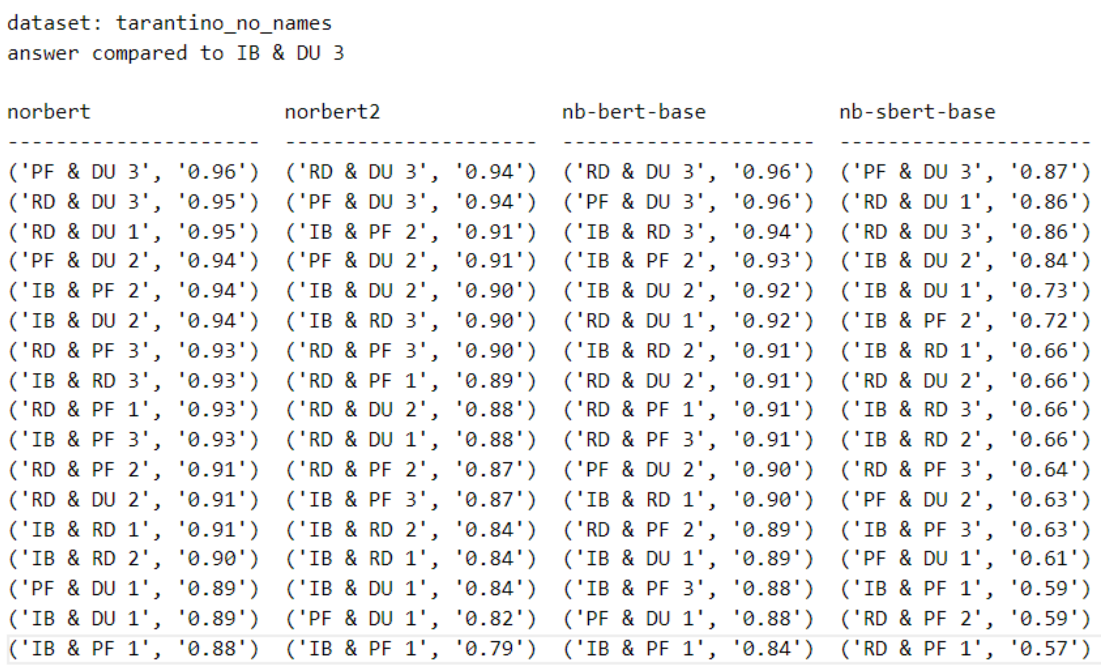
```
dataset: tarantino_no_names
answer compared to IB & DU 1

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & RD 1', '0.92')   ('IB & RD 1', '0.88')   ('IB & RD 1', '0.90')   ('IB & RD 1', '0.73')
('IB & RD 3', '0.92')   ('IB & RD 3', '0.86')   ('PF & DU 3', '0.89')   ('IB & DU 3', '0.73')
('RD & PF 1', '0.90')   ('RD & DU 3', '0.85')   ('IB & DU 3', '0.89')   ('IB & DU 2', '0.71')
('PF & DU 3', '0.90')   ('PF & DU 3', '0.84')   ('IB & PF 2', '0.88')   ('IB & PF 2', '0.70')
('IB & DU 3', '0.89')   ('RD & PF 1', '0.84')   ('RD & DU 3', '0.87')   ('PF & DU 3', '0.68')
('IB & PF 2', '0.89')   ('IB & DU 3', '0.84')   ('IB & RD 3', '0.87')   ('IB & RD 3', '0.67')
('IB & DU 2', '0.88')   ('PF & DU 2', '0.82')   ('RD & DU 2', '0.86')   ('IB & RD 2', '0.66')
('RD & PF 3', '0.88')   ('IB & RD 2', '0.82')   ('RD & PF 3', '0.86')   ('RD & DU 2', '0.65')
('RD & PF 2', '0.88')   ('RD & PF 2', '0.82')   ('PF & DU 2', '0.85')   ('RD & DU 1', '0.64')
('RD & DU 1', '0.88')   ('IB & PF 3', '0.81')   ('IB & RD 2', '0.85')   ('RD & PF 2', '0.63')
('PF & DU 2', '0.88')   ('IB & PF 2', '0.81')   ('RD & PF 1', '0.84')   ('RD & PF 3', '0.61')
('IB & RD 2', '0.88')   ('IB & DU 2', '0.81')   ('PF & DU 1', '0.84')   ('RD & PF 1', '0.61')
('PF & DU 1', '0.88')   ('RD & DU 2', '0.80')   ('RD & PF 2', '0.83')   ('RD & DU 3', '0.59')
('RD & DU 3', '0.87')   ('RD & DU 1', '0.80')   ('RD & DU 1', '0.83')   ('IB & PF 1', '0.58')
('RD & DU 2', '0.87')   ('RD & PF 3', '0.80')   ('IB & DU 2', '0.83')   ('PF & DU 1', '0.57')
('IB & PF 1', '0.87')   ('PF & DU 1', '0.78')   ('IB & PF 3', '0.81')   ('IB & PF 3', '0.57')
('IB & PF 3', '0.86')   ('IB & PF 1', '0.76')   ('IB & PF 1', '0.79')   ('PF & DU 2', '0.53')
```

Figure .53: The results from comparison to entry labeled "IB & DU 1".



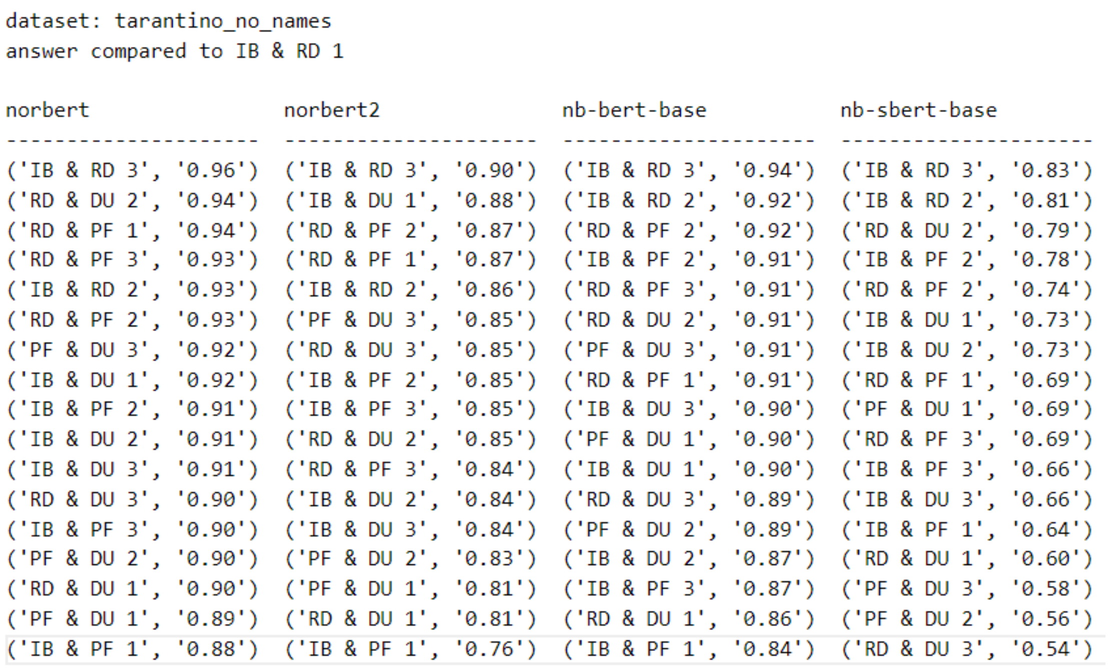
```
dataset: tarantino_no_names
answer compared to IB & DU 2

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & PF 2', '0.95')   ('IB & PF 2', '0.91')   ('IB & PF 2', '0.94')   ('IB & PF 2', '0.85')
('IB & RD 3', '0.94')   ('IB & DU 3', '0.90')   ('IB & DU 3', '0.92')   ('IB & DU 3', '0.84')
('IB & DU 3', '0.94')   ('PF & DU 3', '0.90')   ('IB & RD 3', '0.92')   ('IB & PF 1', '0.77')
('IB & PF 3', '0.93')   ('IB & PF 3', '0.89')   ('IB & RD 2', '0.90')   ('IB & RD 2', '0.73')
('PF & DU 3', '0.93')   ('IB & RD 3', '0.87')   ('PF & DU 3', '0.90')   ('IB & RD 1', '0.73')
('RD & PF 3', '0.92')   ('RD & DU 3', '0.87')   ('IB & PF 3', '0.89')   ('IB & DU 1', '0.71')
('IB & PF 1', '0.92')   ('PF & DU 2', '0.87')   ('IB & PF 1', '0.88')   ('IB & RD 3', '0.68')
('RD & DU 1', '0.91')   ('RD & PF 3', '0.87')   ('RD & PF 3', '0.87')   ('IB & PF 3', '0.67')
('PF & DU 2', '0.91')   ('IB & PF 1', '0.87')   ('RD & DU 3', '0.87')   ('PF & DU 3', '0.66')
('RD & PF 1', '0.91')   ('RD & PF 2', '0.86')   ('IB & RD 1', '0.87')   ('RD & DU 1', '0.66')
('IB & RD 1', '0.91')   ('RD & PF 1', '0.85')   ('PF & DU 2', '0.87')   ('RD & PF 3', '0.62')
('RD & PF 2', '0.91')   ('RD & DU 2', '0.84')   ('RD & PF 1', '0.87')   ('RD & PF 2', '0.59')
('IB & RD 2', '0.90')   ('IB & RD 2', '0.84')   ('RD & PF 2', '0.87')   ('RD & DU 2', '0.58')
('RD & DU 2', '0.90')   ('IB & RD 1', '0.84')   ('RD & DU 2', '0.86')   ('PF & DU 1', '0.57')
('RD & DU 3', '0.89')   ('RD & DU 1', '0.83')   ('RD & DU 1', '0.86')   ('RD & DU 3', '0.57')
('IB & DU 1', '0.88')   ('PF & DU 1', '0.82')   ('PF & DU 1', '0.84')   ('PF & DU 2', '0.54')
('PF & DU 1', '0.87')   ('IB & DU 1', '0.81')   ('IB & DU 1', '0.83')   ('RD & PF 1', '0.53')
```

Figure .54: The results from comparison to entry labeled "IB & DU 2".



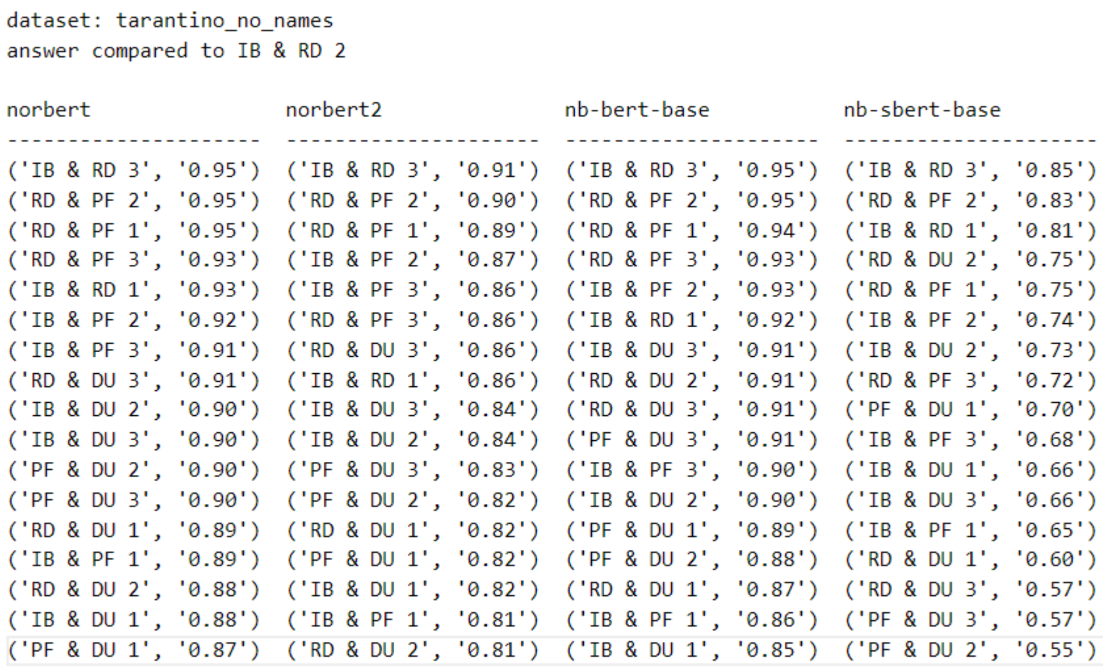
```
dataset: tarantino_no_names
answer compared to IB & DU 3

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('PF & DU 3', '0.96')    ('RD & DU 3', '0.94')    ('RD & DU 3', '0.96')    ('PF & DU 3', '0.87')
('RD & DU 3', '0.95')    ('PF & DU 3', '0.94')    ('PF & DU 3', '0.96')    ('RD & DU 1', '0.86')
('RD & DU 1', '0.95')    ('IB & PF 2', '0.91')    ('IB & RD 3', '0.94')    ('RD & DU 3', '0.86')
('PF & DU 2', '0.94')    ('PF & DU 2', '0.91')    ('IB & PF 2', '0.93')    ('IB & DU 2', '0.84')
('IB & PF 2', '0.94')    ('IB & DU 2', '0.90')    ('IB & DU 2', '0.92')    ('IB & DU 1', '0.73')
('IB & DU 2', '0.94')    ('IB & RD 3', '0.90')    ('RD & DU 1', '0.92')    ('IB & PF 2', '0.72')
('RD & PF 3', '0.93')    ('RD & PF 3', '0.90')    ('IB & RD 2', '0.91')    ('IB & RD 1', '0.66')
('IB & RD 3', '0.93')    ('RD & PF 1', '0.89')    ('RD & DU 2', '0.91')    ('RD & DU 2', '0.66')
('RD & PF 1', '0.93')    ('RD & DU 2', '0.88')    ('RD & PF 1', '0.91')    ('IB & RD 3', '0.66')
('IB & PF 3', '0.93')    ('RD & DU 1', '0.88')    ('RD & PF 3', '0.91')    ('IB & RD 2', '0.66')
('RD & PF 2', '0.91')    ('RD & PF 2', '0.87')    ('PF & DU 2', '0.90')    ('RD & PF 3', '0.64')
('RD & DU 2', '0.91')    ('IB & PF 3', '0.87')    ('IB & RD 1', '0.90')    ('PF & DU 2', '0.63')
('IB & RD 1', '0.91')    ('IB & RD 2', '0.84')    ('RD & PF 2', '0.89')    ('IB & PF 3', '0.63')
('IB & RD 2', '0.90')    ('IB & RD 1', '0.84')    ('IB & DU 1', '0.89')    ('PF & DU 1', '0.61')
('PF & DU 1', '0.89')    ('IB & DU 1', '0.84')    ('IB & PF 3', '0.88')    ('IB & PF 1', '0.59')
('IB & DU 1', '0.89')    ('PF & DU 1', '0.82')    ('PF & DU 1', '0.88')    ('RD & PF 2', '0.59')
('IB & PF 1', '0.88')    ('IB & PF 1', '0.79')    ('IB & PF 1', '0.84')    ('RD & PF 1', '0.57')
```

Figure .55: The results from comparison to entry labeled "IB & DU 3".



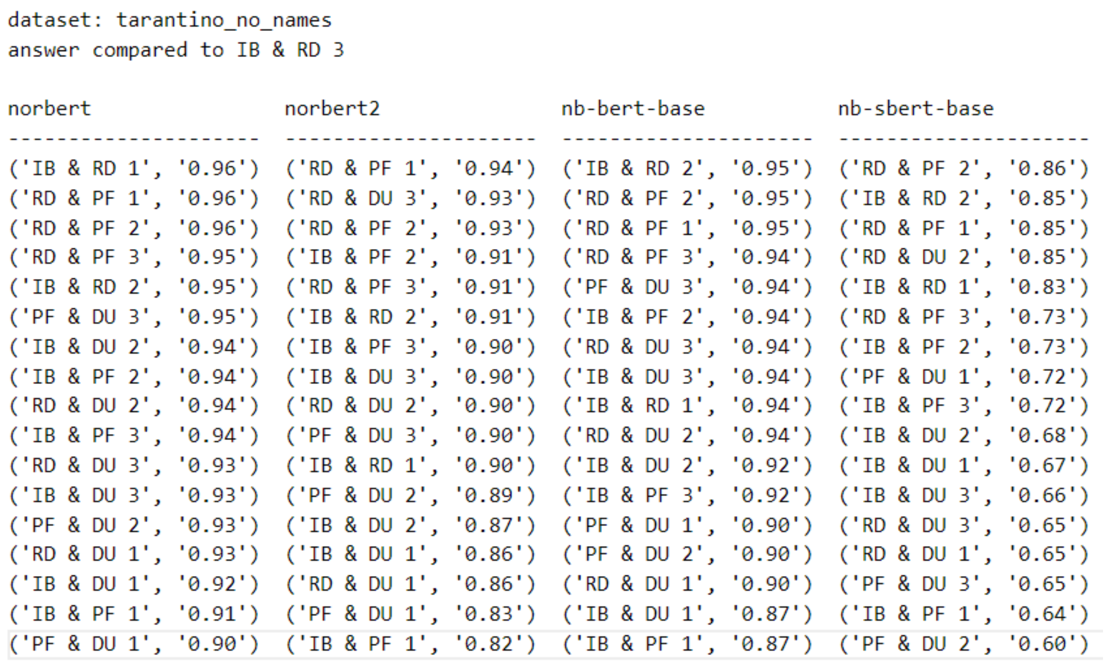
```
dataset: tarantino_no_names
answer compared to IB & RD 1

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('IB & RD 3', '0.96')    ('IB & RD 3', '0.90')    ('IB & RD 3', '0.94')    ('IB & RD 3', '0.83')
('RD & DU 2', '0.94')    ('IB & DU 1', '0.88')    ('IB & RD 2', '0.92')    ('IB & RD 2', '0.81')
('RD & PF 1', '0.94')    ('RD & PF 2', '0.87')    ('RD & PF 2', '0.92')    ('RD & DU 2', '0.79')
('RD & PF 3', '0.93')    ('RD & PF 1', '0.87')    ('IB & PF 2', '0.91')    ('IB & PF 2', '0.78')
('IB & RD 2', '0.93')    ('IB & RD 2', '0.86')    ('RD & PF 3', '0.91')    ('RD & PF 2', '0.74')
('RD & PF 2', '0.93')    ('PF & DU 3', '0.85')    ('RD & DU 2', '0.91')    ('IB & DU 1', '0.73')
('PF & DU 3', '0.92')    ('RD & DU 3', '0.85')    ('PF & DU 3', '0.91')    ('IB & DU 2', '0.73')
('IB & DU 1', '0.92')    ('IB & PF 2', '0.85')    ('RD & PF 1', '0.91')    ('RD & PF 1', '0.69')
('IB & PF 2', '0.91')    ('IB & PF 3', '0.85')    ('IB & DU 3', '0.90')    ('PF & DU 1', '0.69')
('IB & DU 2', '0.91')    ('RD & DU 2', '0.85')    ('PF & DU 1', '0.90')    ('RD & PF 3', '0.69')
('IB & DU 3', '0.91')    ('RD & PF 3', '0.84')    ('IB & DU 1', '0.90')    ('IB & PF 3', '0.66')
('RD & DU 3', '0.90')    ('IB & DU 2', '0.84')    ('RD & DU 3', '0.89')    ('IB & DU 3', '0.66')
('IB & PF 3', '0.90')    ('IB & DU 3', '0.84')    ('PF & DU 2', '0.89')    ('IB & PF 1', '0.64')
('PF & DU 2', '0.90')    ('PF & DU 2', '0.83')    ('IB & DU 2', '0.87')    ('RD & DU 1', '0.60')
('RD & DU 1', '0.90')    ('PF & DU 1', '0.81')    ('IB & PF 3', '0.87')    ('PF & DU 3', '0.58')
('PF & DU 1', '0.89')    ('RD & DU 1', '0.81')    ('RD & DU 1', '0.86')    ('PF & DU 2', '0.56')
('IB & PF 1', '0.88')    ('IB & PF 1', '0.76')    ('IB & PF 1', '0.84')    ('RD & DU 3', '0.54')
```

Figure .56: The results from comparison to entry labeled "IB & RD 1".



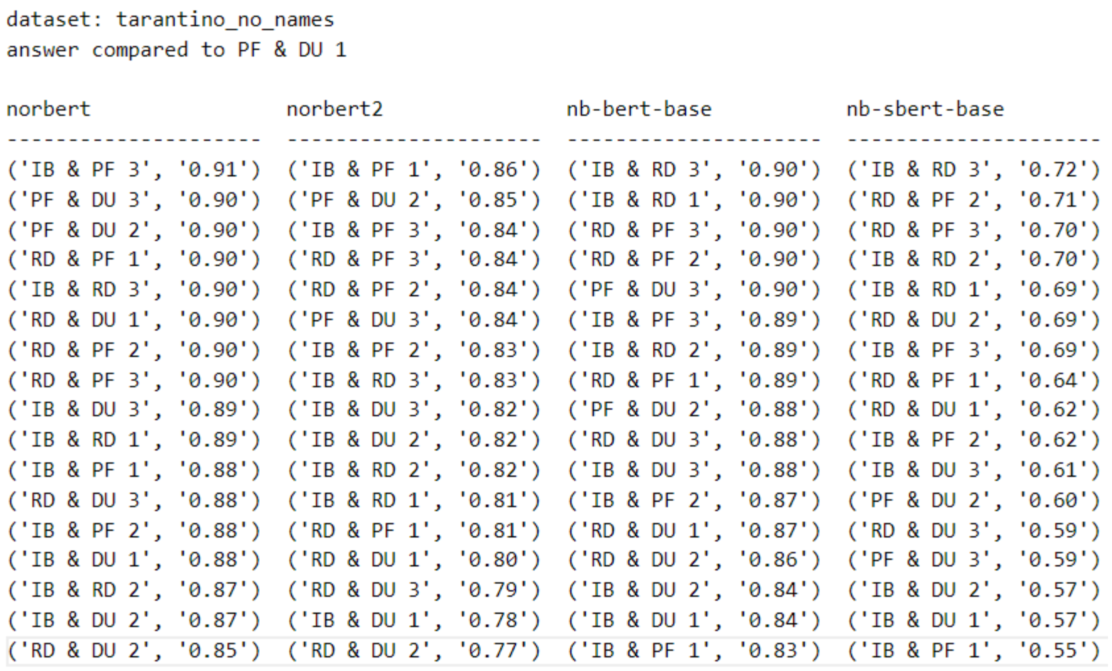
```
dataset: tarantino_no_names
answer compared to IB & RD 2

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & RD 3', '0.95')   ('IB & RD 3', '0.91')   ('IB & RD 3', '0.95')   ('IB & RD 3', '0.85')
('RD & PF 2', '0.95')   ('RD & PF 2', '0.90')   ('RD & PF 2', '0.95')   ('RD & PF 2', '0.83')
('RD & PF 1', '0.95')   ('RD & PF 1', '0.89')   ('RD & PF 1', '0.94')   ('IB & RD 1', '0.81')
('RD & PF 3', '0.93')   ('IB & PF 2', '0.87')   ('RD & PF 3', '0.93')   ('RD & DU 2', '0.75')
('IB & RD 1', '0.93')   ('IB & PF 3', '0.86')   ('IB & PF 2', '0.93')   ('RD & PF 1', '0.75')
('IB & PF 2', '0.92')   ('RD & PF 3', '0.86')   ('IB & RD 1', '0.92')   ('IB & PF 2', '0.74')
('IB & PF 3', '0.91')   ('RD & DU 3', '0.86')   ('IB & DU 3', '0.91')   ('IB & DU 2', '0.73')
('RD & DU 3', '0.91')   ('IB & RD 1', '0.86')   ('RD & DU 2', '0.91')   ('RD & PF 3', '0.72')
('IB & DU 2', '0.90')   ('IB & DU 3', '0.84')   ('RD & DU 3', '0.91')   ('PF & DU 1', '0.70')
('IB & DU 3', '0.90')   ('IB & DU 2', '0.84')   ('PF & DU 3', '0.91')   ('IB & PF 3', '0.68')
('PF & DU 2', '0.90')   ('PF & DU 3', '0.83')   ('IB & PF 3', '0.90')   ('IB & DU 1', '0.66')
('PF & DU 3', '0.90')   ('PF & DU 2', '0.82')   ('IB & DU 2', '0.90')   ('IB & DU 3', '0.66')
('RD & DU 1', '0.89')   ('RD & DU 1', '0.82')   ('PF & DU 1', '0.89')   ('IB & PF 1', '0.65')
('IB & PF 1', '0.89')   ('PF & DU 1', '0.82')   ('PF & DU 2', '0.88')   ('RD & DU 1', '0.60')
('RD & DU 2', '0.88')   ('IB & DU 1', '0.82')   ('RD & DU 1', '0.87')   ('RD & DU 3', '0.57')
('IB & DU 1', '0.88')   ('IB & PF 1', '0.81')   ('IB & PF 1', '0.86')   ('PF & DU 3', '0.57')
('PF & DU 1', '0.87')   ('RD & DU 2', '0.81')   ('IB & DU 1', '0.85')   ('PF & DU 2', '0.55')
```

Figure .57: The results from comparison to entry labeled "IB & RD 2".



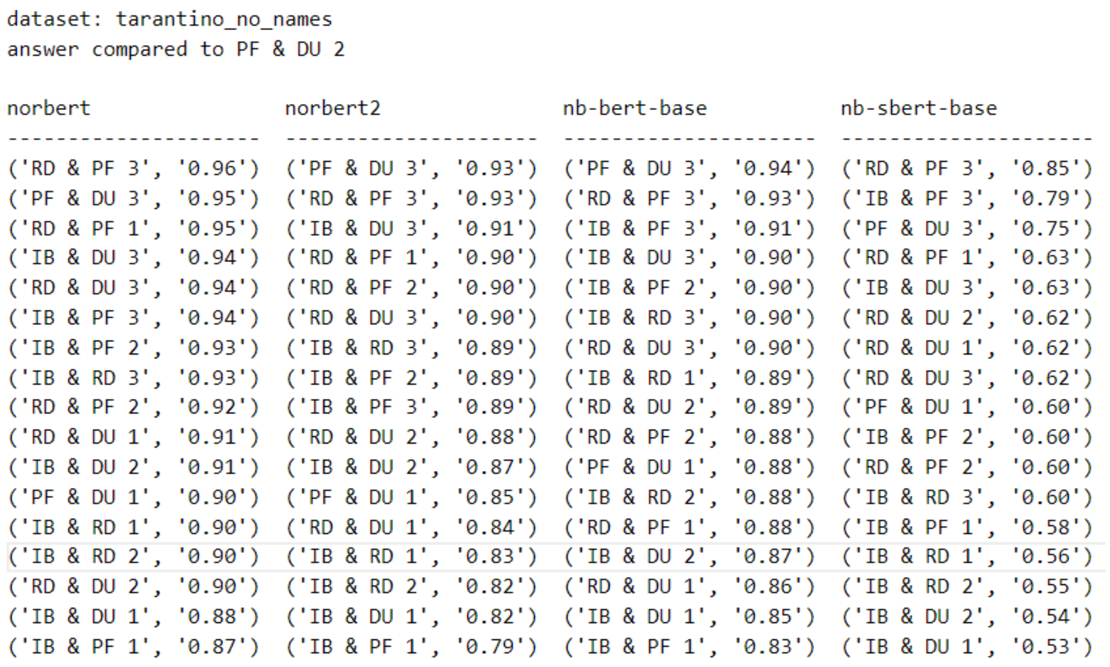
```
dataset: tarantino_no_names
answer compared to IB & RD 3

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & RD 1', '0.96')   ('RD & PF 1', '0.94')   ('IB & RD 2', '0.95')   ('RD & PF 2', '0.86')
('RD & PF 1', '0.96')   ('RD & DU 3', '0.93')   ('RD & PF 2', '0.95')   ('IB & RD 2', '0.85')
('RD & PF 2', '0.96')   ('RD & PF 2', '0.93')   ('RD & PF 1', '0.95')   ('RD & PF 1', '0.85')
('RD & PF 3', '0.95')   ('IB & PF 2', '0.91')   ('RD & PF 3', '0.94')   ('RD & DU 2', '0.85')
('IB & RD 2', '0.95')   ('RD & PF 3', '0.91')   ('PF & DU 3', '0.94')   ('IB & RD 1', '0.83')
('PF & DU 3', '0.95')   ('IB & RD 2', '0.91')   ('IB & PF 2', '0.94')   ('RD & PF 3', '0.73')
('IB & DU 2', '0.94')   ('IB & PF 3', '0.90')   ('RD & DU 3', '0.94')   ('IB & PF 2', '0.73')
('IB & PF 2', '0.94')   ('IB & DU 3', '0.90')   ('IB & DU 3', '0.94')   ('PF & DU 1', '0.72')
('RD & DU 2', '0.94')   ('RD & DU 2', '0.90')   ('IB & RD 1', '0.94')   ('IB & PF 3', '0.72')
('IB & PF 3', '0.94')   ('PF & DU 3', '0.90')   ('RD & DU 2', '0.94')   ('IB & DU 2', '0.68')
('RD & DU 3', '0.93')   ('IB & RD 1', '0.90')   ('IB & DU 2', '0.92')   ('IB & DU 1', '0.67')
('IB & DU 3', '0.93')   ('PF & DU 2', '0.89')   ('IB & PF 3', '0.92')   ('IB & DU 3', '0.66')
('PF & DU 2', '0.93')   ('IB & DU 2', '0.87')   ('PF & DU 1', '0.90')   ('RD & DU 3', '0.65')
('RD & DU 1', '0.93')   ('IB & DU 1', '0.86')   ('PF & DU 2', '0.90')   ('RD & DU 1', '0.65')
('IB & DU 1', '0.92')   ('RD & DU 1', '0.86')   ('RD & DU 1', '0.90')   ('PF & DU 3', '0.65')
('IB & PF 1', '0.91')   ('PF & DU 1', '0.83')   ('IB & DU 1', '0.87')   ('IB & PF 1', '0.64')
('PF & DU 1', '0.90')   ('IB & PF 1', '0.82')   ('IB & PF 1', '0.87')   ('PF & DU 2', '0.60')
```

Figure .58: The results from comparison to entry labeled "IB & RD 3".



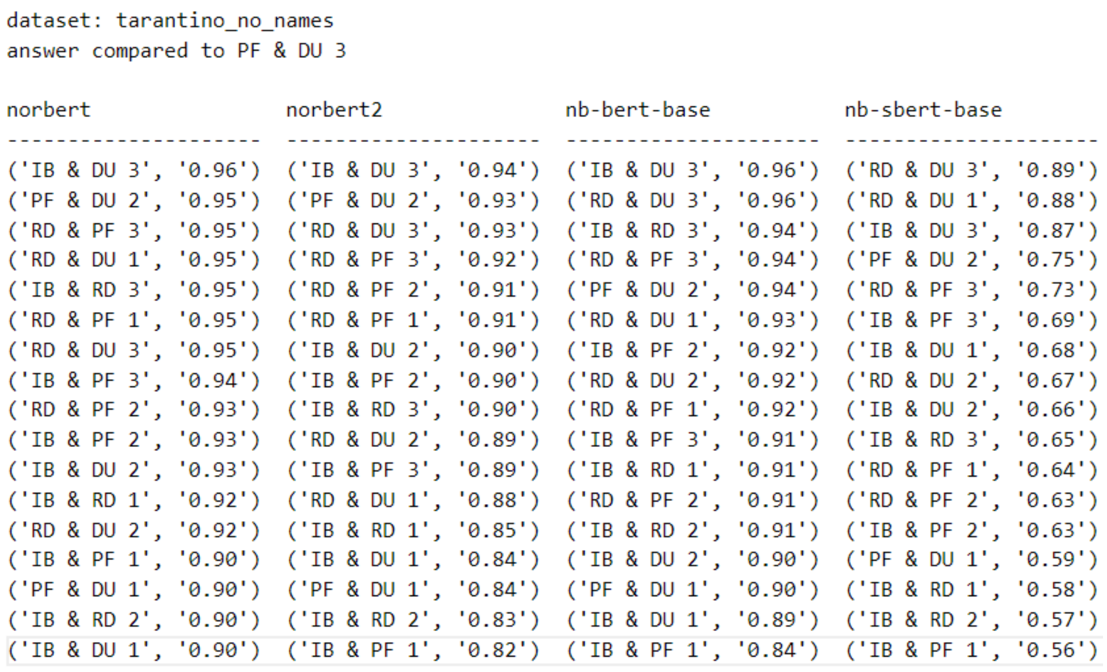
```
dataset: tarantino_no_names
answer compared to PF & DU 1

norbert                norbert2               nb-bert-base           nb-sbert-base
---------------------  ---------------------  ---------------------  ---------------------
('IB & PF 3', '0.91')  ('IB & PF 1', '0.86')  ('IB & RD 3', '0.90')  ('IB & RD 3', '0.72')
('PF & DU 3', '0.90')  ('PF & DU 2', '0.85')  ('IB & RD 1', '0.90')  ('RD & PF 2', '0.71')
('PF & DU 2', '0.90')  ('IB & PF 3', '0.84')  ('RD & PF 3', '0.90')  ('RD & PF 3', '0.70')
('RD & PF 1', '0.90')  ('RD & PF 3', '0.84')  ('RD & PF 2', '0.90')  ('IB & RD 2', '0.70')
('IB & RD 3', '0.90')  ('RD & PF 2', '0.84')  ('PF & DU 3', '0.90')  ('IB & RD 1', '0.69')
('RD & DU 1', '0.90')  ('PF & DU 3', '0.84')  ('IB & PF 3', '0.89')  ('RD & DU 2', '0.69')
('RD & PF 2', '0.90')  ('IB & PF 2', '0.83')  ('IB & RD 2', '0.89')  ('IB & PF 3', '0.69')
('RD & PF 3', '0.90')  ('IB & RD 3', '0.83')  ('RD & PF 1', '0.89')  ('RD & PF 1', '0.64')
('IB & DU 3', '0.89')  ('IB & DU 3', '0.82')  ('PF & DU 2', '0.88')  ('RD & DU 1', '0.62')
('IB & RD 1', '0.89')  ('IB & DU 2', '0.82')  ('RD & DU 3', '0.88')  ('IB & PF 2', '0.62')
('IB & PF 1', '0.88')  ('IB & RD 2', '0.82')  ('IB & DU 3', '0.88')  ('IB & DU 3', '0.61')
('RD & DU 3', '0.88')  ('IB & RD 1', '0.81')  ('IB & PF 2', '0.87')  ('PF & DU 2', '0.60')
('IB & PF 2', '0.88')  ('RD & PF 1', '0.81')  ('RD & DU 1', '0.87')  ('RD & DU 3', '0.59')
('IB & DU 1', '0.88')  ('RD & DU 1', '0.80')  ('RD & DU 2', '0.86')  ('PF & DU 3', '0.59')
('IB & RD 2', '0.87')  ('RD & DU 3', '0.79')  ('IB & DU 2', '0.84')  ('IB & DU 2', '0.57')
('IB & DU 2', '0.87')  ('IB & DU 1', '0.78')  ('IB & DU 1', '0.84')  ('IB & DU 1', '0.57')
('RD & DU 2', '0.85')  ('RD & DU 2', '0.77')  ('IB & PF 1', '0.83')  ('IB & PF 1', '0.55')
```

Figure .59: The results from comparison to entry labeled "PF & DU 1".



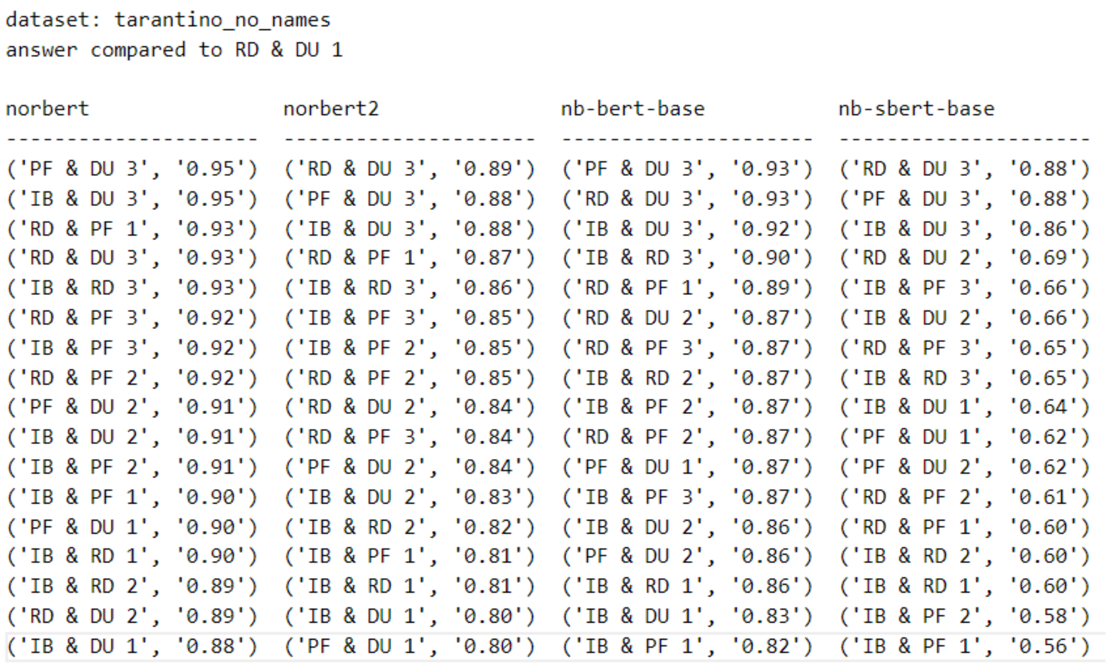
```
dataset: tarantino_no_names
answer compared to PF & DU 2

norbert                norbert2               nb-bert-base           nb-sbert-base
---------------------  ---------------------  ---------------------  ---------------------
('RD & PF 3', '0.96')  ('PF & DU 3', '0.93')  ('PF & DU 3', '0.94')  ('RD & PF 3', '0.85')
('PF & DU 3', '0.95')  ('RD & PF 3', '0.93')  ('RD & PF 3', '0.93')  ('IB & PF 3', '0.79')
('RD & PF 1', '0.95')  ('IB & DU 3', '0.91')  ('IB & PF 3', '0.91')  ('PF & DU 3', '0.75')
('IB & DU 3', '0.94')  ('RD & PF 1', '0.90')  ('IB & DU 3', '0.90')  ('RD & PF 1', '0.63')
('RD & DU 3', '0.94')  ('RD & PF 2', '0.90')  ('IB & PF 2', '0.90')  ('IB & DU 3', '0.63')
('IB & PF 3', '0.94')  ('RD & DU 3', '0.90')  ('IB & RD 3', '0.90')  ('RD & DU 2', '0.62')
('IB & PF 2', '0.93')  ('IB & RD 3', '0.89')  ('RD & DU 3', '0.90')  ('RD & DU 1', '0.62')
('IB & RD 3', '0.93')  ('IB & PF 2', '0.89')  ('IB & RD 1', '0.89')  ('RD & DU 3', '0.62')
('RD & PF 2', '0.92')  ('IB & PF 3', '0.89')  ('RD & DU 2', '0.89')  ('PF & DU 1', '0.60')
('RD & DU 1', '0.91')  ('RD & DU 2', '0.88')  ('RD & PF 2', '0.88')  ('IB & PF 2', '0.60')
('IB & DU 2', '0.91')  ('IB & DU 2', '0.87')  ('PF & DU 1', '0.88')  ('RD & PF 2', '0.60')
('PF & DU 1', '0.90')  ('PF & DU 1', '0.85')  ('IB & RD 2', '0.88')  ('IB & RD 3', '0.60')
('IB & RD 1', '0.90')  ('RD & DU 1', '0.84')  ('RD & PF 1', '0.88')  ('IB & PF 1', '0.58')
('IB & RD 2', '0.90')  ('IB & RD 1', '0.83')  ('IB & DU 2', '0.87')  ('IB & RD 1', '0.56')
('RD & DU 2', '0.90')  ('IB & RD 2', '0.82')  ('RD & DU 1', '0.86')  ('IB & RD 2', '0.55')
('IB & DU 1', '0.88')  ('IB & DU 1', '0.82')  ('IB & DU 1', '0.85')  ('IB & DU 2', '0.54')
('IB & PF 1', '0.87')  ('IB & PF 1', '0.79')  ('IB & PF 1', '0.83')  ('IB & DU 1', '0.53')
```

Figure .60: The results from comparison to entry labeled "PF & DU 2".



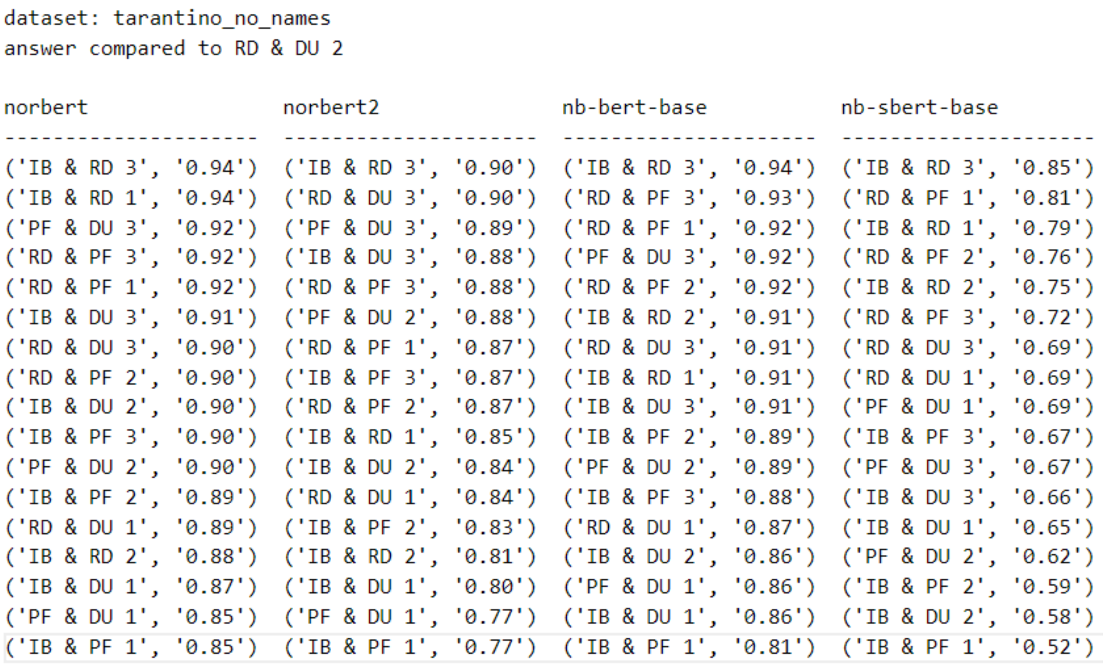
```
dataset: tarantino_no_names
answer compared to PF & DU 3

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('IB & DU 3', '0.96')    ('IB & DU 3', '0.94')    ('IB & DU 3', '0.96')    ('RD & DU 3', '0.89')
('PF & DU 2', '0.95')    ('PF & DU 2', '0.93')    ('RD & DU 3', '0.96')    ('RD & DU 1', '0.88')
('RD & PF 3', '0.95')    ('RD & DU 3', '0.93')    ('IB & RD 3', '0.94')    ('IB & DU 3', '0.87')
('RD & DU 1', '0.95')    ('RD & PF 3', '0.92')    ('RD & PF 3', '0.94')    ('PF & DU 2', '0.75')
('IB & RD 3', '0.95')    ('RD & PF 2', '0.91')    ('PF & DU 2', '0.94')    ('RD & PF 3', '0.73')
('RD & PF 1', '0.95')    ('RD & PF 1', '0.91')    ('RD & DU 1', '0.93')    ('IB & PF 3', '0.69')
('RD & DU 3', '0.95')    ('IB & DU 2', '0.90')    ('IB & PF 2', '0.92')    ('IB & DU 1', '0.68')
('IB & PF 3', '0.94')    ('IB & PF 2', '0.90')    ('RD & DU 2', '0.92')    ('RD & DU 2', '0.67')
('RD & PF 2', '0.93')    ('IB & RD 3', '0.90')    ('RD & PF 1', '0.92')    ('IB & DU 2', '0.66')
('IB & PF 2', '0.93')    ('RD & DU 2', '0.89')    ('IB & PF 3', '0.91')    ('IB & RD 3', '0.65')
('IB & DU 2', '0.93')    ('IB & PF 3', '0.89')    ('IB & RD 1', '0.91')    ('RD & PF 1', '0.64')
('IB & RD 1', '0.92')    ('RD & DU 1', '0.88')    ('RD & PF 2', '0.91')    ('RD & PF 2', '0.63')
('RD & DU 2', '0.92')    ('IB & RD 1', '0.85')    ('IB & RD 2', '0.91')    ('IB & PF 2', '0.63')
('IB & PF 1', '0.90')    ('IB & DU 1', '0.84')    ('IB & DU 2', '0.90')    ('PF & DU 1', '0.59')
('PF & DU 1', '0.90')    ('PF & DU 1', '0.84')    ('PF & DU 1', '0.90')    ('IB & RD 1', '0.58')
('IB & RD 2', '0.90')    ('IB & RD 2', '0.83')    ('IB & DU 1', '0.89')    ('IB & RD 2', '0.57')
('IB & DU 1', '0.90')    ('IB & PF 1', '0.82')    ('IB & PF 1', '0.84')    ('IB & PF 1', '0.56')
```

Figure .61: The results from comparison to entry labeled "PF & DU 3".



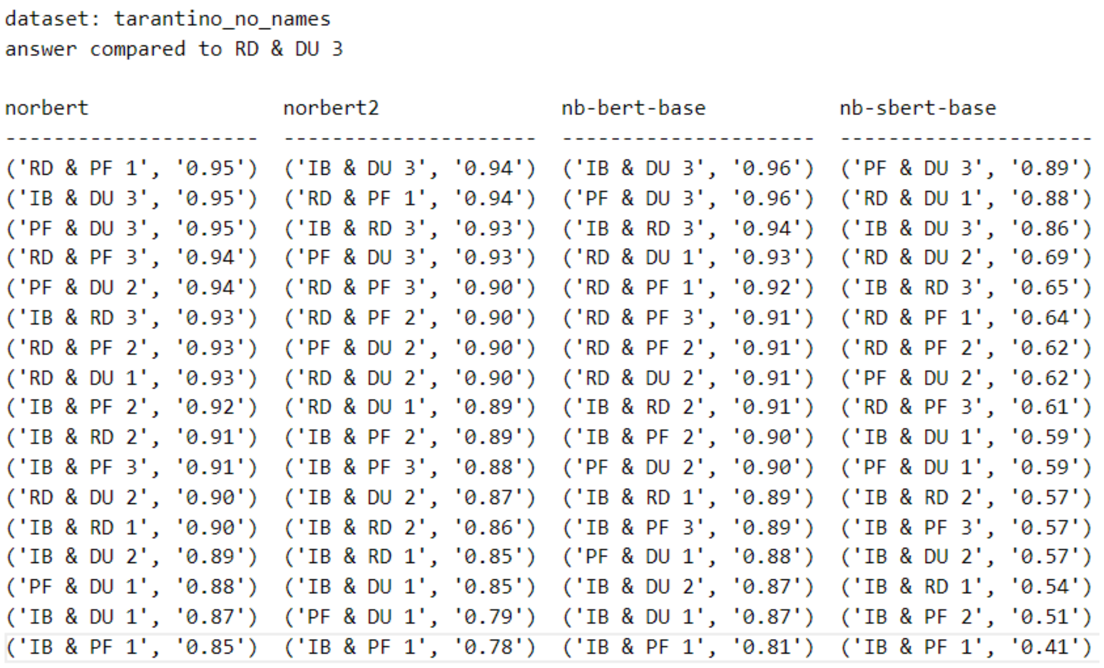
```
dataset: tarantino_no_names
answer compared to RD & DU 1

norbert                  norbert2                 nb-bert-base             nb-sbert-base
---------------------    ---------------------    ---------------------    ---------------------
('PF & DU 3', '0.95')    ('RD & DU 3', '0.89')    ('PF & DU 3', '0.93')    ('RD & DU 3', '0.88')
('IB & DU 3', '0.95')    ('PF & DU 3', '0.88')    ('RD & DU 3', '0.93')    ('PF & DU 3', '0.88')
('RD & PF 1', '0.93')    ('IB & DU 3', '0.88')    ('IB & DU 3', '0.92')    ('IB & DU 3', '0.86')
('RD & DU 3', '0.93')    ('RD & PF 1', '0.87')    ('IB & RD 3', '0.90')    ('RD & DU 2', '0.69')
('IB & RD 3', '0.93')    ('IB & RD 3', '0.86')    ('RD & PF 1', '0.89')    ('IB & PF 3', '0.66')
('RD & PF 3', '0.92')    ('IB & PF 3', '0.85')    ('RD & DU 2', '0.87')    ('IB & DU 2', '0.66')
('IB & PF 3', '0.92')    ('IB & PF 2', '0.85')    ('RD & PF 3', '0.87')    ('RD & PF 3', '0.65')
('RD & PF 2', '0.92')    ('RD & PF 2', '0.85')    ('IB & RD 2', '0.87')    ('IB & RD 3', '0.65')
('PF & DU 2', '0.91')    ('RD & DU 2', '0.84')    ('IB & PF 2', '0.87')    ('IB & DU 1', '0.64')
('IB & DU 2', '0.91')    ('RD & PF 3', '0.84')    ('RD & PF 2', '0.87')    ('PF & DU 1', '0.62')
('IB & PF 2', '0.91')    ('PF & DU 2', '0.84')    ('PF & DU 1', '0.87')    ('PF & DU 2', '0.62')
('IB & PF 1', '0.90')    ('IB & DU 2', '0.83')    ('IB & PF 3', '0.87')    ('RD & PF 2', '0.61')
('PF & DU 1', '0.90')    ('IB & RD 2', '0.82')    ('IB & DU 2', '0.86')    ('RD & PF 1', '0.60')
('IB & RD 1', '0.90')    ('IB & PF 1', '0.81')    ('PF & DU 2', '0.86')    ('IB & RD 2', '0.60')
('IB & RD 2', '0.89')    ('IB & RD 1', '0.81')    ('IB & RD 1', '0.86')    ('IB & RD 1', '0.60')
('RD & DU 2', '0.89')    ('IB & DU 1', '0.80')    ('IB & DU 1', '0.83')    ('IB & PF 2', '0.58')
('IB & DU 1', '0.88')    ('PF & DU 1', '0.80')    ('IB & PF 1', '0.82')    ('IB & PF 1', '0.56')
```

Figure .62: The results from comparison to entry labeled "RD & DU 1".



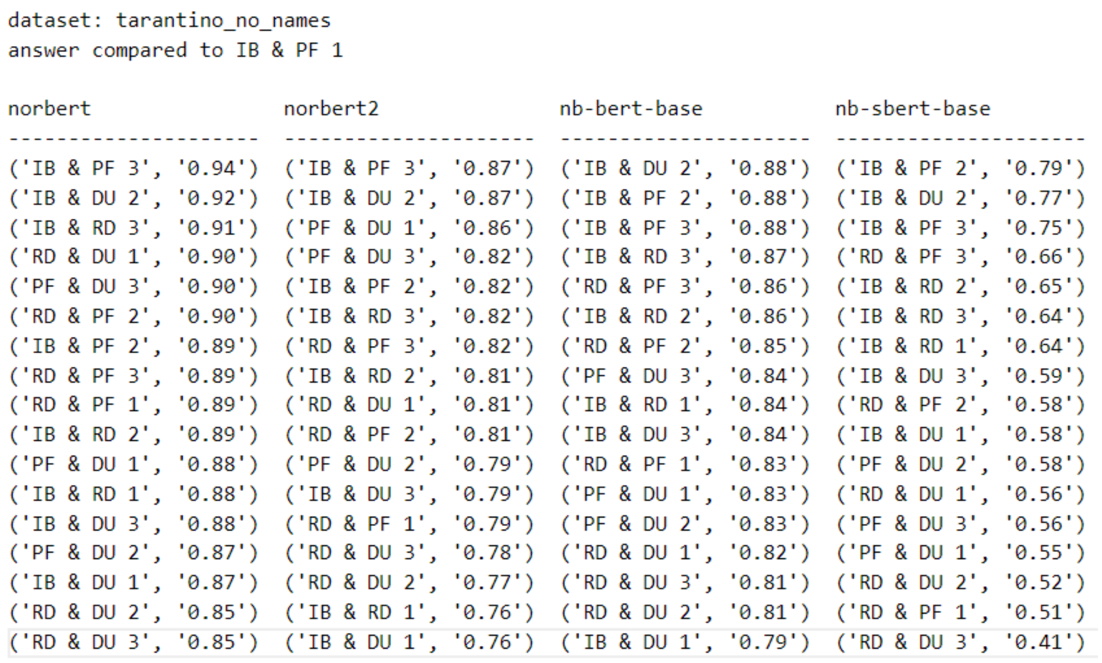
```
dataset: tarantino_no_names
answer compared to RD & DU 2

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & RD 3', '0.94')   ('IB & RD 3', '0.90')   ('IB & RD 3', '0.94')   ('IB & RD 3', '0.85')
('IB & RD 1', '0.94')   ('RD & DU 3', '0.90')   ('RD & PF 3', '0.93')   ('RD & PF 1', '0.81')
('PF & DU 3', '0.92')   ('PF & DU 3', '0.89')   ('RD & PF 1', '0.92')   ('IB & RD 1', '0.79')
('RD & PF 3', '0.92')   ('IB & DU 3', '0.88')   ('PF & DU 3', '0.92')   ('RD & PF 2', '0.76')
('RD & PF 1', '0.92')   ('RD & PF 3', '0.88')   ('RD & PF 2', '0.92')   ('IB & RD 2', '0.75')
('IB & DU 3', '0.91')   ('PF & DU 2', '0.88')   ('IB & RD 2', '0.91')   ('RD & PF 3', '0.72')
('RD & DU 3', '0.90')   ('RD & PF 1', '0.87')   ('RD & DU 3', '0.91')   ('RD & DU 3', '0.69')
('RD & PF 2', '0.90')   ('IB & PF 3', '0.87')   ('IB & RD 1', '0.91')   ('RD & DU 1', '0.69')
('IB & DU 2', '0.90')   ('RD & PF 2', '0.87')   ('IB & DU 3', '0.91')   ('PF & DU 1', '0.69')
('IB & PF 3', '0.90')   ('IB & RD 1', '0.85')   ('IB & PF 2', '0.89')   ('IB & PF 3', '0.67')
('PF & DU 2', '0.90')   ('IB & DU 2', '0.84')   ('PF & DU 2', '0.89')   ('PF & DU 3', '0.67')
('IB & PF 2', '0.89')   ('RD & DU 1', '0.84')   ('IB & PF 3', '0.88')   ('IB & DU 3', '0.66')
('RD & DU 1', '0.89')   ('IB & PF 2', '0.83')   ('RD & DU 1', '0.87')   ('IB & DU 1', '0.65')
('IB & RD 2', '0.88')   ('IB & RD 2', '0.81')   ('IB & DU 2', '0.86')   ('PF & DU 2', '0.62')
('IB & DU 1', '0.87')   ('IB & DU 1', '0.80')   ('PF & DU 1', '0.86')   ('IB & PF 2', '0.59')
('PF & DU 1', '0.85')   ('PF & DU 1', '0.77')   ('IB & DU 1', '0.86')   ('IB & DU 2', '0.58')
('IB & PF 1', '0.85')   ('IB & PF 1', '0.77')   ('IB & PF 1', '0.81')   ('IB & PF 1', '0.52')
```

Figure .63: The results from comparison to entry labeled "RD & DU 2".



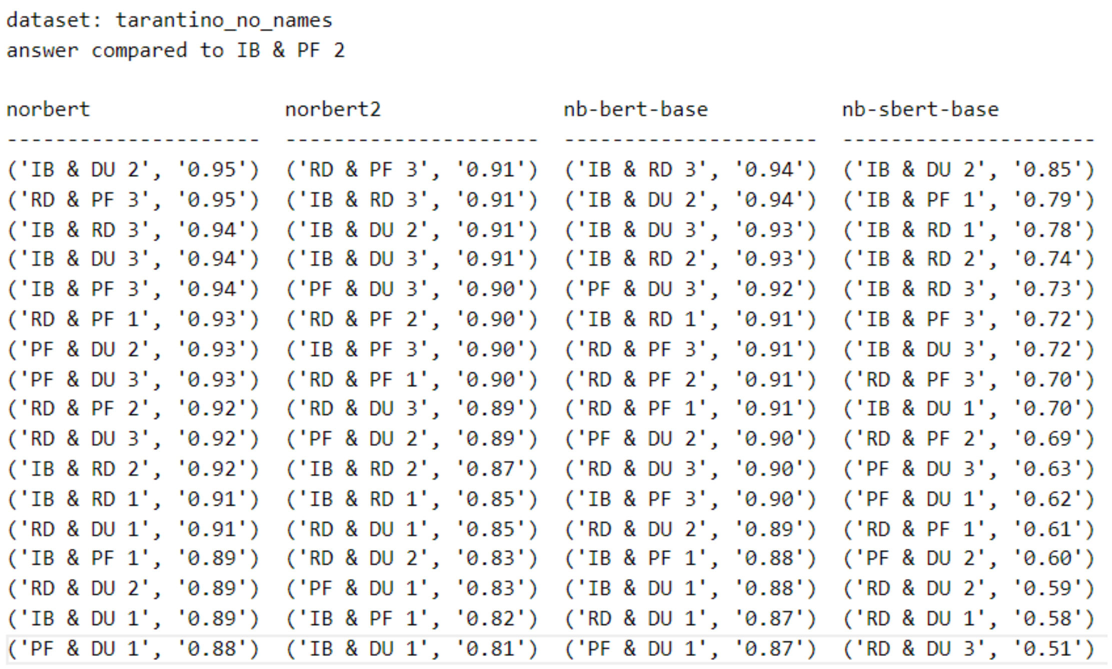
```
dataset: tarantino_no_names
answer compared to RD & DU 3

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('RD & PF 1', '0.95')   ('IB & DU 3', '0.94')   ('IB & DU 3', '0.96')   ('PF & DU 3', '0.89')
('IB & DU 3', '0.95')   ('RD & PF 1', '0.94')   ('PF & DU 3', '0.96')   ('RD & DU 1', '0.88')
('PF & DU 3', '0.95')   ('IB & RD 3', '0.93')   ('IB & RD 3', '0.94')   ('IB & DU 3', '0.86')
('RD & PF 3', '0.94')   ('PF & DU 3', '0.93')   ('RD & DU 1', '0.93')   ('RD & DU 2', '0.69')
('PF & DU 2', '0.94')   ('RD & PF 3', '0.90')   ('RD & PF 1', '0.92')   ('IB & RD 3', '0.65')
('IB & RD 3', '0.93')   ('RD & PF 2', '0.90')   ('RD & PF 3', '0.91')   ('RD & PF 1', '0.64')
('RD & PF 2', '0.93')   ('PF & DU 2', '0.90')   ('RD & PF 2', '0.91')   ('RD & PF 2', '0.62')
('RD & DU 1', '0.93')   ('RD & DU 2', '0.90')   ('RD & DU 2', '0.91')   ('PF & DU 2', '0.62')
('IB & PF 2', '0.92')   ('RD & DU 1', '0.89')   ('IB & RD 2', '0.91')   ('RD & PF 3', '0.61')
('IB & RD 2', '0.91')   ('IB & PF 2', '0.89')   ('IB & PF 2', '0.90')   ('IB & DU 1', '0.59')
('IB & PF 3', '0.91')   ('IB & PF 3', '0.88')   ('PF & DU 2', '0.90')   ('PF & DU 1', '0.59')
('RD & DU 2', '0.90')   ('IB & DU 2', '0.87')   ('IB & RD 1', '0.89')   ('IB & RD 2', '0.57')
('IB & RD 1', '0.90')   ('IB & RD 2', '0.86')   ('IB & PF 3', '0.89')   ('IB & PF 3', '0.57')
('IB & DU 2', '0.89')   ('IB & RD 1', '0.85')   ('PF & DU 1', '0.88')   ('IB & DU 2', '0.57')
('PF & DU 1', '0.88')   ('IB & DU 1', '0.85')   ('IB & DU 2', '0.87')   ('IB & RD 1', '0.54')
('IB & DU 1', '0.87')   ('PF & DU 1', '0.79')   ('IB & DU 1', '0.87')   ('IB & PF 2', '0.51')
('IB & PF 1', '0.85')   ('IB & PF 1', '0.78')   ('IB & PF 1', '0.81')   ('IB & PF 1', '0.41')
```

Figure .64: The results from comparison to entry labeled "RD & DU 3".



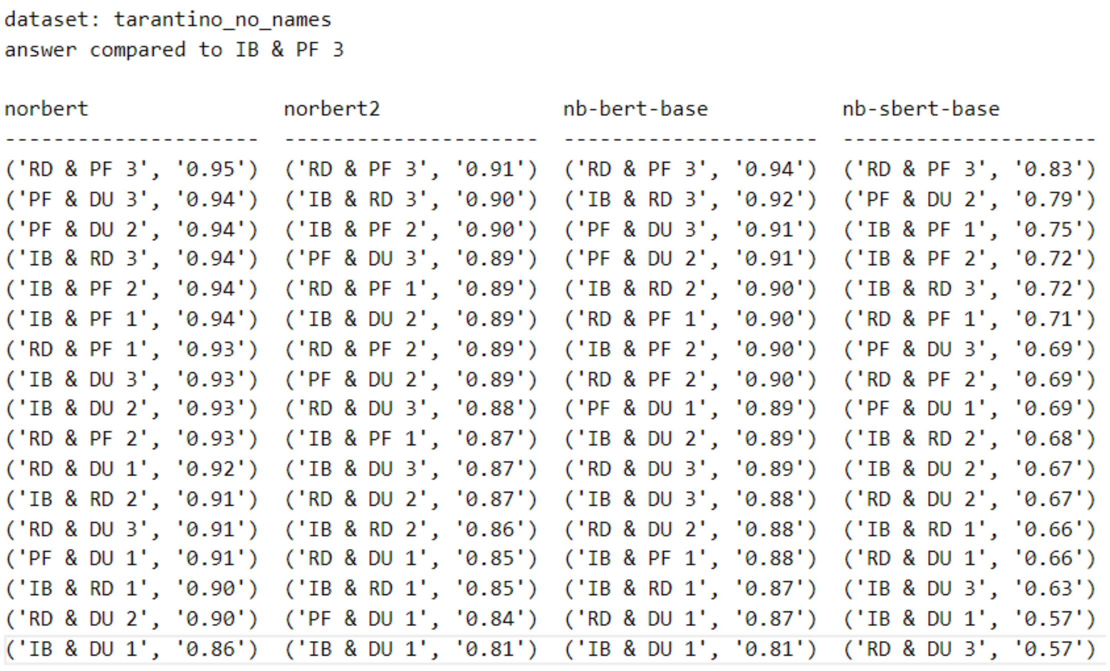
```
dataset: tarantino_no_names
answer compared to IB & PF 1

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & PF 3', '0.94')   ('IB & PF 3', '0.87')   ('IB & DU 2', '0.88')   ('IB & PF 2', '0.79')
('IB & DU 2', '0.92')   ('IB & DU 2', '0.87')   ('IB & PF 2', '0.88')   ('IB & DU 2', '0.77')
('IB & RD 3', '0.91')   ('PF & DU 1', '0.86')   ('IB & PF 3', '0.88')   ('IB & PF 3', '0.75')
('RD & DU 1', '0.90')   ('PF & DU 3', '0.82')   ('IB & RD 3', '0.87')   ('RD & PF 3', '0.66')
('PF & DU 3', '0.90')   ('IB & PF 2', '0.82')   ('RD & PF 3', '0.86')   ('IB & RD 2', '0.65')
('RD & PF 2', '0.90')   ('IB & RD 3', '0.82')   ('IB & RD 2', '0.86')   ('IB & RD 3', '0.64')
('IB & PF 2', '0.89')   ('RD & PF 3', '0.82')   ('RD & PF 2', '0.85')   ('IB & RD 1', '0.64')
('RD & PF 3', '0.89')   ('IB & RD 2', '0.81')   ('PF & DU 3', '0.84')   ('IB & DU 3', '0.59')
('RD & PF 1', '0.89')   ('RD & DU 1', '0.81')   ('IB & RD 1', '0.84')   ('RD & PF 2', '0.58')
('IB & RD 2', '0.89')   ('RD & PF 2', '0.81')   ('IB & DU 3', '0.84')   ('IB & DU 1', '0.58')
('PF & DU 1', '0.88')   ('PF & DU 2', '0.79')   ('RD & PF 1', '0.83')   ('PF & DU 2', '0.58')
('IB & RD 1', '0.88')   ('IB & DU 3', '0.79')   ('PF & DU 1', '0.83')   ('RD & DU 1', '0.56')
('IB & DU 3', '0.88')   ('RD & PF 1', '0.79')   ('PF & DU 2', '0.83')   ('PF & DU 3', '0.56')
('PF & DU 2', '0.87')   ('RD & DU 3', '0.78')   ('RD & DU 1', '0.82')   ('PF & DU 1', '0.55')
('IB & DU 1', '0.87')   ('RD & DU 2', '0.77')   ('RD & DU 3', '0.81')   ('RD & DU 2', '0.52')
('RD & DU 2', '0.85')   ('IB & RD 1', '0.76')   ('RD & DU 2', '0.81')   ('RD & PF 1', '0.51')
('RD & DU 3', '0.85')   ('IB & DU 1', '0.76')   ('IB & DU 1', '0.79')   ('RD & DU 3', '0.41')
```

Figure .65: The results from comparison to entry labeled "IB & PF 1".



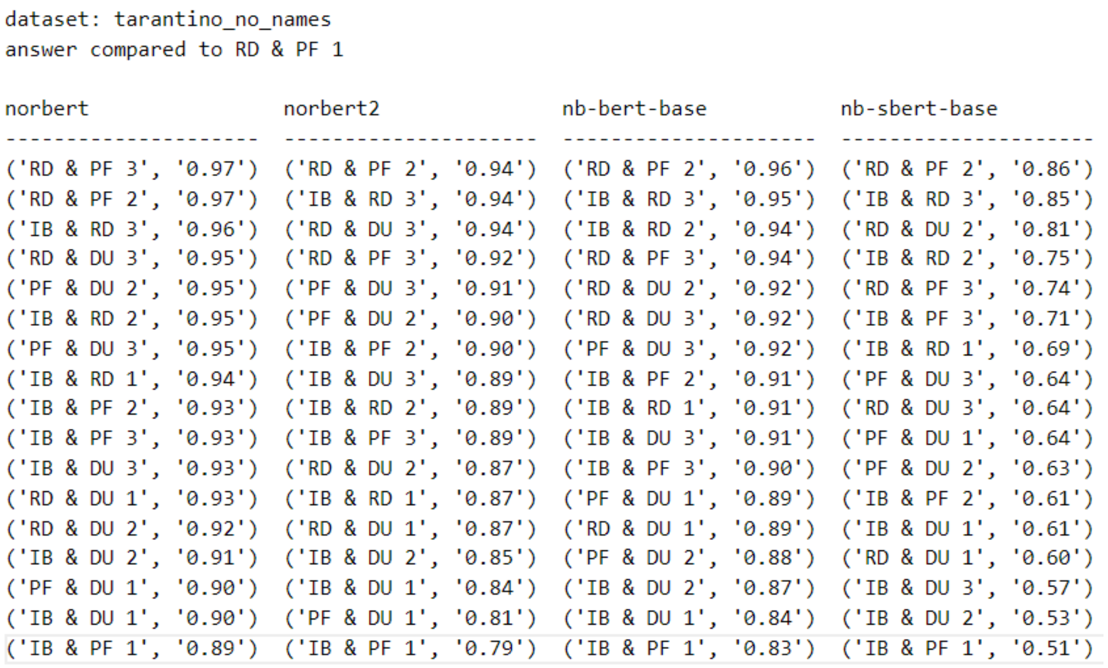
```
dataset: tarantino_no_names
answer compared to IB & PF 2

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('IB & DU 2', '0.95')   ('RD & PF 3', '0.91')   ('IB & RD 3', '0.94')   ('IB & DU 2', '0.85')
('RD & PF 3', '0.95')   ('IB & RD 3', '0.91')   ('IB & DU 2', '0.94')   ('IB & PF 1', '0.79')
('IB & RD 3', '0.94')   ('IB & DU 2', '0.91')   ('IB & DU 3', '0.93')   ('IB & RD 1', '0.78')
('IB & DU 3', '0.94')   ('IB & DU 3', '0.91')   ('IB & RD 2', '0.93')   ('IB & RD 2', '0.74')
('IB & PF 3', '0.94')   ('PF & DU 3', '0.90')   ('PF & DU 3', '0.92')   ('IB & RD 3', '0.73')
('RD & PF 1', '0.93')   ('RD & PF 2', '0.90')   ('IB & RD 1', '0.91')   ('IB & PF 3', '0.72')
('PF & DU 2', '0.93')   ('IB & PF 3', '0.90')   ('RD & PF 3', '0.91')   ('IB & DU 3', '0.72')
('PF & DU 3', '0.93')   ('RD & PF 1', '0.90')   ('RD & PF 2', '0.91')   ('RD & PF 3', '0.70')
('RD & PF 2', '0.92')   ('RD & DU 3', '0.89')   ('RD & PF 1', '0.91')   ('IB & DU 1', '0.70')
('RD & DU 3', '0.92')   ('PF & DU 2', '0.89')   ('PF & DU 2', '0.90')   ('RD & PF 2', '0.69')
('IB & RD 2', '0.92')   ('IB & RD 2', '0.87')   ('RD & DU 3', '0.90')   ('PF & DU 3', '0.63')
('IB & RD 1', '0.91')   ('IB & RD 1', '0.85')   ('IB & PF 3', '0.90')   ('PF & DU 1', '0.62')
('RD & DU 1', '0.91')   ('RD & DU 1', '0.85')   ('RD & DU 2', '0.89')   ('RD & PF 1', '0.61')
('IB & PF 1', '0.89')   ('RD & DU 2', '0.83')   ('IB & PF 1', '0.88')   ('PF & DU 2', '0.60')
('RD & DU 2', '0.89')   ('PF & DU 1', '0.83')   ('IB & DU 1', '0.88')   ('RD & DU 2', '0.59')
('IB & DU 1', '0.89')   ('IB & PF 1', '0.82')   ('RD & DU 1', '0.87')   ('RD & DU 1', '0.58')
('PF & DU 1', '0.88')   ('IB & DU 1', '0.81')   ('PF & DU 1', '0.87')   ('RD & DU 3', '0.51')
```

Figure .66: The results from comparison to entry labeled "IB & PF 2".



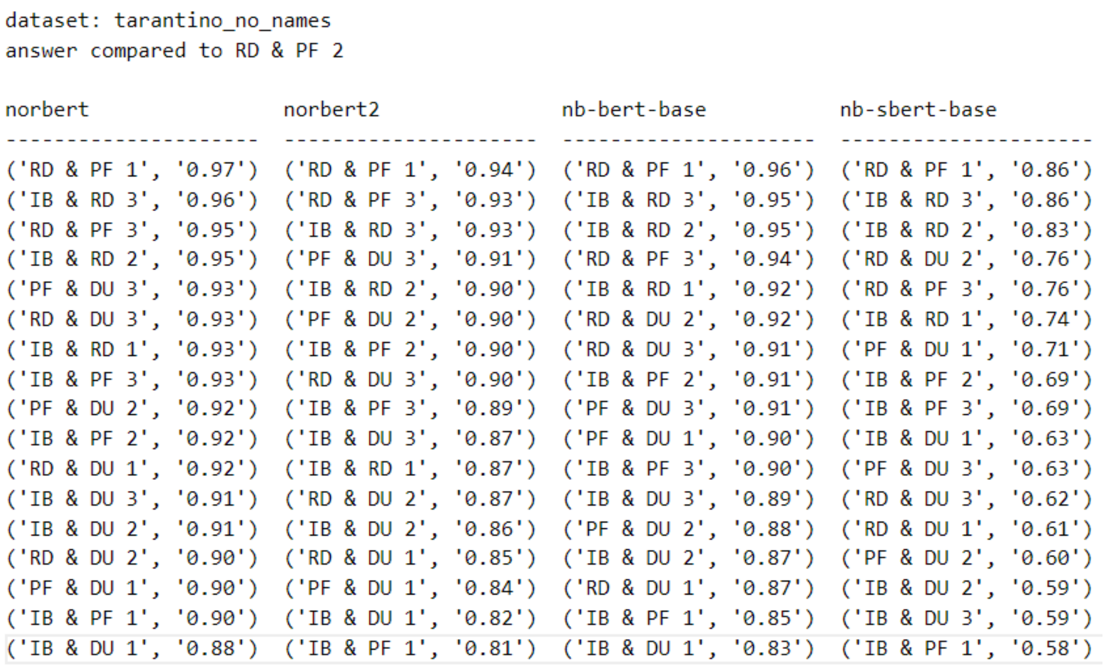
```
dataset: tarantino_no_names
answer compared to IB & PF 3

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('RD & PF 3', '0.95')   ('RD & PF 3', '0.91')   ('RD & PF 3', '0.94')   ('RD & PF 3', '0.83')
('PF & DU 3', '0.94')   ('IB & RD 3', '0.90')   ('IB & RD 3', '0.92')   ('PF & DU 2', '0.79')
('PF & DU 2', '0.94')   ('IB & PF 2', '0.90')   ('PF & DU 3', '0.91')   ('IB & PF 1', '0.75')
('IB & RD 3', '0.94')   ('PF & DU 3', '0.89')   ('PF & DU 2', '0.91')   ('IB & PF 2', '0.72')
('IB & PF 2', '0.94')   ('RD & PF 1', '0.89')   ('IB & RD 2', '0.90')   ('IB & RD 3', '0.72')
('IB & PF 1', '0.94')   ('IB & DU 2', '0.89')   ('RD & PF 1', '0.90')   ('RD & PF 1', '0.71')
('RD & PF 1', '0.93')   ('RD & PF 2', '0.89')   ('IB & PF 2', '0.90')   ('PF & DU 3', '0.69')
('IB & DU 3', '0.93')   ('PF & DU 2', '0.89')   ('RD & PF 2', '0.90')   ('RD & PF 2', '0.69')
('IB & DU 2', '0.93')   ('RD & DU 3', '0.88')   ('PF & DU 1', '0.89')   ('PF & DU 1', '0.69')
('RD & PF 2', '0.93')   ('IB & PF 1', '0.87')   ('IB & DU 2', '0.89')   ('IB & RD 2', '0.68')
('RD & DU 1', '0.92')   ('IB & DU 3', '0.87')   ('RD & DU 3', '0.89')   ('IB & DU 2', '0.67')
('IB & RD 2', '0.91')   ('RD & DU 2', '0.87')   ('IB & DU 3', '0.88')   ('RD & DU 2', '0.67')
('RD & DU 3', '0.91')   ('IB & RD 2', '0.86')   ('RD & DU 2', '0.88')   ('IB & RD 1', '0.66')
('PF & DU 1', '0.91')   ('RD & DU 1', '0.85')   ('IB & PF 1', '0.88')   ('RD & DU 1', '0.66')
('IB & RD 1', '0.90')   ('IB & RD 1', '0.85')   ('IB & RD 1', '0.87')   ('IB & DU 3', '0.63')
('RD & DU 2', '0.90')   ('PF & DU 1', '0.84')   ('RD & DU 1', '0.87')   ('IB & DU 1', '0.57')
('IB & DU 1', '0.86')   ('IB & DU 1', '0.81')   ('IB & DU 1', '0.81')   ('RD & DU 3', '0.57')
```

Figure .67: The results from comparison to entry labeled "IB & PF 3".



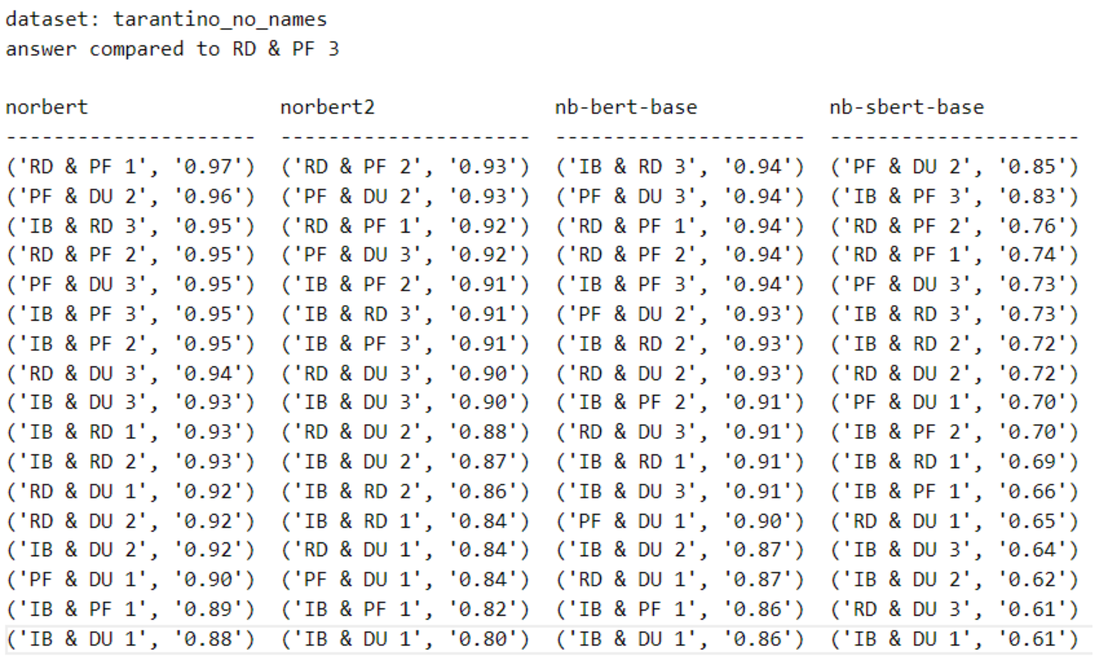
```
dataset: tarantino_no_names
answer compared to RD & PF 1

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('RD & PF 3', '0.97')   ('RD & PF 2', '0.94')   ('RD & PF 2', '0.96')   ('RD & PF 2', '0.86')
('RD & PF 2', '0.97')   ('IB & RD 3', '0.94')   ('IB & RD 3', '0.95')   ('IB & RD 3', '0.85')
('IB & RD 3', '0.96')   ('RD & DU 3', '0.94')   ('IB & RD 2', '0.94')   ('RD & DU 2', '0.81')
('RD & DU 3', '0.95')   ('RD & PF 3', '0.92')   ('RD & PF 3', '0.94')   ('IB & RD 2', '0.75')
('PF & DU 2', '0.95')   ('PF & DU 3', '0.91')   ('RD & DU 2', '0.92')   ('RD & PF 3', '0.74')
('IB & RD 2', '0.95')   ('PF & DU 2', '0.90')   ('RD & DU 3', '0.92')   ('IB & PF 3', '0.71')
('PF & DU 3', '0.95')   ('IB & PF 2', '0.90')   ('PF & DU 3', '0.92')   ('IB & RD 1', '0.69')
('IB & RD 1', '0.94')   ('IB & DU 3', '0.89')   ('IB & PF 2', '0.91')   ('PF & DU 3', '0.64')
('IB & PF 2', '0.93')   ('IB & RD 2', '0.89')   ('IB & RD 1', '0.91')   ('RD & DU 3', '0.64')
('IB & PF 3', '0.93')   ('IB & PF 3', '0.89')   ('IB & DU 3', '0.91')   ('PF & DU 1', '0.64')
('IB & DU 3', '0.93')   ('RD & DU 2', '0.87')   ('IB & PF 3', '0.90')   ('PF & DU 2', '0.63')
('RD & DU 1', '0.93')   ('IB & RD 1', '0.87')   ('PF & DU 1', '0.89')   ('IB & PF 2', '0.61')
('RD & DU 2', '0.92')   ('RD & DU 1', '0.87')   ('RD & DU 1', '0.89')   ('IB & DU 1', '0.61')
('IB & DU 2', '0.91')   ('IB & DU 2', '0.85')   ('PF & DU 2', '0.88')   ('RD & DU 1', '0.60')
('PF & DU 1', '0.90')   ('IB & DU 1', '0.84')   ('IB & DU 2', '0.87')   ('IB & DU 3', '0.57')
('IB & DU 1', '0.90')   ('PF & DU 1', '0.81')   ('IB & DU 1', '0.84')   ('IB & DU 2', '0.53')
('IB & PF 1', '0.89')   ('IB & PF 1', '0.79')   ('IB & PF 1', '0.83')   ('IB & PF 1', '0.51')
```

Figure .68: The results from comparison to entry labeled "RD & PF 1".



```
dataset: tarantino_no_names
answer compared to RD & PF 2

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('RD & PF 1', '0.97')   ('RD & PF 1', '0.94')   ('RD & PF 1', '0.96')   ('RD & PF 1', '0.86')
('IB & RD 3', '0.96')   ('RD & PF 3', '0.93')   ('IB & RD 3', '0.95')   ('IB & RD 3', '0.86')
('RD & PF 3', '0.95')   ('IB & RD 3', '0.93')   ('IB & RD 2', '0.95')   ('IB & RD 2', '0.83')
('IB & RD 2', '0.95')   ('PF & DU 3', '0.91')   ('RD & PF 3', '0.94')   ('RD & DU 2', '0.76')
('PF & DU 3', '0.93')   ('IB & RD 2', '0.90')   ('IB & RD 1', '0.92')   ('RD & PF 3', '0.76')
('RD & DU 3', '0.93')   ('PF & DU 2', '0.90')   ('RD & DU 2', '0.92')   ('IB & RD 1', '0.74')
('IB & RD 1', '0.93')   ('IB & PF 2', '0.90')   ('RD & DU 3', '0.91')   ('PF & DU 1', '0.71')
('IB & PF 3', '0.93')   ('RD & DU 3', '0.90')   ('IB & PF 2', '0.91')   ('IB & PF 2', '0.69')
('PF & DU 2', '0.92')   ('IB & PF 3', '0.89')   ('PF & DU 3', '0.91')   ('IB & PF 3', '0.69')
('IB & PF 2', '0.92')   ('IB & DU 3', '0.87')   ('PF & DU 1', '0.90')   ('IB & DU 1', '0.63')
('RD & DU 1', '0.92')   ('IB & RD 1', '0.87')   ('IB & PF 3', '0.90')   ('PF & DU 3', '0.63')
('IB & DU 3', '0.91')   ('RD & DU 2', '0.87')   ('IB & DU 3', '0.89')   ('RD & DU 3', '0.62')
('IB & DU 2', '0.91')   ('IB & DU 2', '0.86')   ('PF & DU 2', '0.88')   ('RD & DU 1', '0.61')
('RD & DU 2', '0.90')   ('RD & DU 1', '0.85')   ('IB & DU 2', '0.87')   ('PF & DU 2', '0.60')
('PF & DU 1', '0.90')   ('PF & DU 1', '0.84')   ('RD & DU 1', '0.87')   ('IB & DU 2', '0.59')
('IB & PF 1', '0.90')   ('IB & DU 1', '0.82')   ('IB & PF 1', '0.85')   ('IB & DU 3', '0.59')
('IB & DU 1', '0.88')   ('IB & PF 1', '0.81')   ('IB & DU 1', '0.83')   ('IB & PF 1', '0.58')
```

Figure .69: The results from comparison to entry labeled "RD & PF 2".

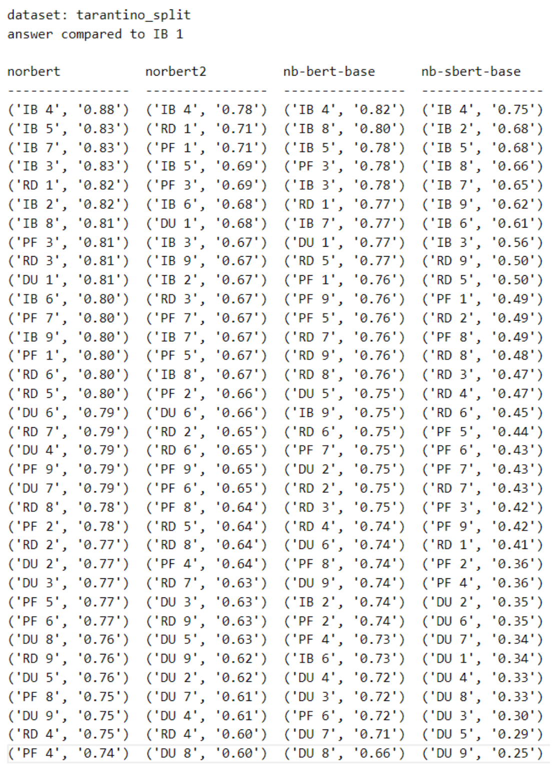


```
dataset: tarantino_no_names
answer compared to RD & PF 3

norbert                 norbert2                nb-bert-base            nb-sbert-base
---------------------   ---------------------   ---------------------   ---------------------
('RD & PF 1', '0.97')   ('RD & PF 2', '0.93')   ('IB & RD 3', '0.94')   ('PF & DU 2', '0.85')
('PF & DU 2', '0.96')   ('PF & DU 2', '0.93')   ('PF & DU 3', '0.94')   ('IB & PF 3', '0.83')
('IB & RD 3', '0.95')   ('RD & PF 1', '0.92')   ('RD & PF 1', '0.94')   ('RD & PF 2', '0.76')
('RD & PF 2', '0.95')   ('PF & DU 3', '0.92')   ('RD & PF 2', '0.94')   ('RD & PF 1', '0.74')
('PF & DU 3', '0.95')   ('IB & PF 2', '0.91')   ('IB & PF 3', '0.94')   ('PF & DU 3', '0.73')
('IB & PF 3', '0.95')   ('IB & RD 3', '0.91')   ('PF & DU 2', '0.93')   ('IB & RD 3', '0.73')
('IB & PF 2', '0.95')   ('IB & PF 3', '0.91')   ('IB & RD 2', '0.93')   ('IB & RD 2', '0.72')
('RD & DU 3', '0.94')   ('RD & DU 3', '0.90')   ('RD & DU 2', '0.93')   ('RD & DU 2', '0.72')
('IB & DU 3', '0.93')   ('IB & DU 3', '0.90')   ('IB & PF 2', '0.91')   ('PF & DU 1', '0.70')
('IB & RD 1', '0.93')   ('RD & DU 2', '0.88')   ('RD & DU 3', '0.91')   ('IB & PF 2', '0.70')
('IB & RD 2', '0.93')   ('IB & DU 2', '0.87')   ('IB & RD 1', '0.91')   ('IB & RD 1', '0.69')
('RD & DU 1', '0.92')   ('IB & RD 2', '0.86')   ('IB & DU 3', '0.91')   ('IB & PF 1', '0.66')
('RD & DU 2', '0.92')   ('IB & RD 1', '0.84')   ('PF & DU 1', '0.90')   ('RD & DU 1', '0.65')
('IB & DU 2', '0.92')   ('RD & DU 1', '0.84')   ('IB & DU 2', '0.87')   ('IB & DU 3', '0.64')
('PF & DU 1', '0.90')   ('PF & DU 1', '0.84')   ('RD & DU 1', '0.87')   ('IB & DU 2', '0.62')
('IB & PF 1', '0.89')   ('IB & PF 1', '0.82')   ('IB & PF 1', '0.86')   ('RD & DU 3', '0.61')
('IB & DU 1', '0.88')   ('IB & DU 1', '0.80')   ('IB & DU 1', '0.86')   ('IB & DU 1', '0.61')
```

Figure .70: The results from comparison to entry labeled "RD & PF 3".

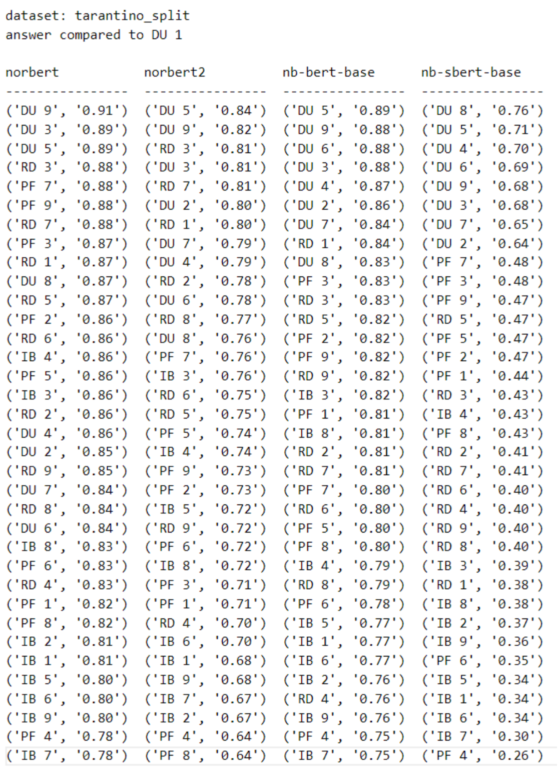
B4 - Tarantino single descriptions dataset

All results from the tests performed on the Tarantino dataset with one movie descriptions per entry and titles included.



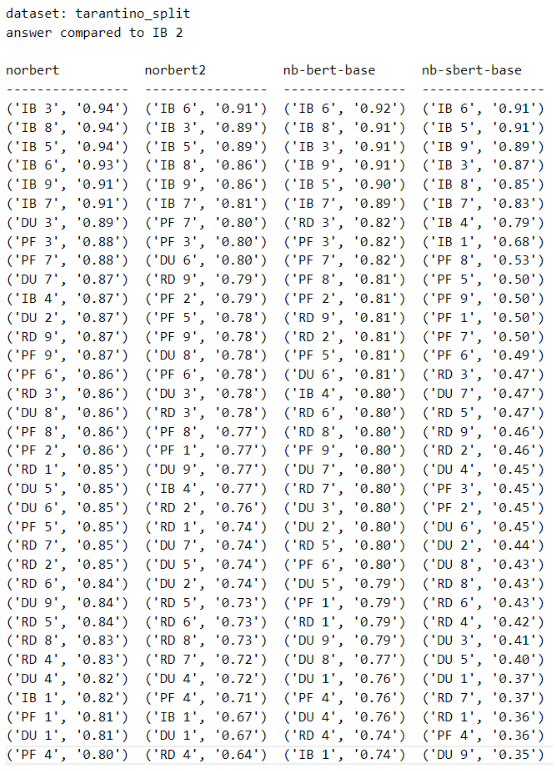
```
dataset: tarantino_split
answer compared to IB 1

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('IB 4', '0.88')   ('IB 4', '0.78')   ('IB 4', '0.82')   ('IB 4', '0.75')
('IB 5', '0.83')   ('RD 1', '0.71')   ('IB 8', '0.80')   ('IB 2', '0.68')
('IB 7', '0.83')   ('PF 1', '0.71')   ('IB 5', '0.78')   ('IB 5', '0.68')
('IB 3', '0.83')   ('IB 5', '0.69')   ('PF 3', '0.78')   ('IB 8', '0.66')
('RD 1', '0.82')   ('PF 3', '0.69')   ('IB 3', '0.78')   ('IB 7', '0.65')
('IB 2', '0.82')   ('IB 6', '0.68')   ('RD 1', '0.77')   ('IB 9', '0.62')
('IB 8', '0.81')   ('DU 1', '0.68')   ('IB 7', '0.77')   ('IB 6', '0.61')
('PF 3', '0.81')   ('IB 3', '0.67')   ('DU 1', '0.77')   ('IB 3', '0.56')
('RD 3', '0.81')   ('IB 9', '0.67')   ('RD 5', '0.77')   ('RD 9', '0.50')
('DU 1', '0.81')   ('IB 2', '0.67')   ('PF 1', '0.76')   ('RD 5', '0.50')
('IB 6', '0.80')   ('RD 3', '0.67')   ('PF 9', '0.76')   ('PF 1', '0.49')
('PF 7', '0.80')   ('PF 7', '0.67')   ('PF 5', '0.76')   ('RD 2', '0.49')
('IB 9', '0.80')   ('IB 7', '0.67')   ('RD 7', '0.76')   ('PF 8', '0.49')
('PF 1', '0.80')   ('PF 5', '0.67')   ('RD 9', '0.76')   ('RD 8', '0.48')
('RD 6', '0.80')   ('IB 8', '0.67')   ('RD 8', '0.76')   ('RD 3', '0.47')
('RD 5', '0.80')   ('PF 2', '0.66')   ('DU 5', '0.75')   ('RD 4', '0.47')
('DU 6', '0.79')   ('DU 6', '0.66')   ('IB 9', '0.75')   ('RD 6', '0.45')
('RD 7', '0.79')   ('RD 2', '0.65')   ('RD 6', '0.75')   ('PF 5', '0.44')
('DU 4', '0.79')   ('RD 6', '0.65')   ('PF 7', '0.75')   ('PF 6', '0.43')
('PF 9', '0.79')   ('PF 9', '0.65')   ('DU 2', '0.75')   ('PF 7', '0.43')
('DU 7', '0.79')   ('PF 6', '0.65')   ('RD 2', '0.75')   ('RD 7', '0.43')
('RD 8', '0.78')   ('PF 8', '0.64')   ('RD 3', '0.75')   ('PF 3', '0.42')
('PF 2', '0.78')   ('RD 5', '0.64')   ('RD 4', '0.74')   ('PF 9', '0.42')
('RD 2', '0.77')   ('RD 8', '0.64')   ('DU 6', '0.74')   ('RD 1', '0.41')
('DU 2', '0.77')   ('PF 4', '0.64')   ('PF 8', '0.74')   ('PF 2', '0.36')
('DU 3', '0.77')   ('RD 7', '0.63')   ('DU 9', '0.74')   ('PF 4', '0.36')
('PF 5', '0.77')   ('DU 3', '0.63')   ('IB 2', '0.74')   ('DU 2', '0.35')
('PF 6', '0.77')   ('RD 9', '0.63')   ('PF 2', '0.74')   ('DU 6', '0.35')
('DU 8', '0.76')   ('DU 5', '0.63')   ('PF 4', '0.73')   ('DU 7', '0.34')
('RD 9', '0.76')   ('DU 9', '0.62')   ('IB 6', '0.73')   ('DU 1', '0.34')
('DU 5', '0.76')   ('DU 2', '0.62')   ('DU 4', '0.72')   ('DU 4', '0.33')
('PF 8', '0.75')   ('DU 7', '0.61')   ('DU 3', '0.72')   ('DU 8', '0.33')
('DU 9', '0.75')   ('DU 4', '0.61')   ('PF 6', '0.72')   ('DU 3', '0.30')
('RD 4', '0.75')   ('RD 4', '0.60')   ('DU 7', '0.71')   ('DU 5', '0.29')
('PF 4', '0.74')   ('DU 8', '0.60')   ('DU 8', '0.66')   ('DU 9', '0.25')
```

Figure .71: The results from comparison to entry labeled "IB 1".



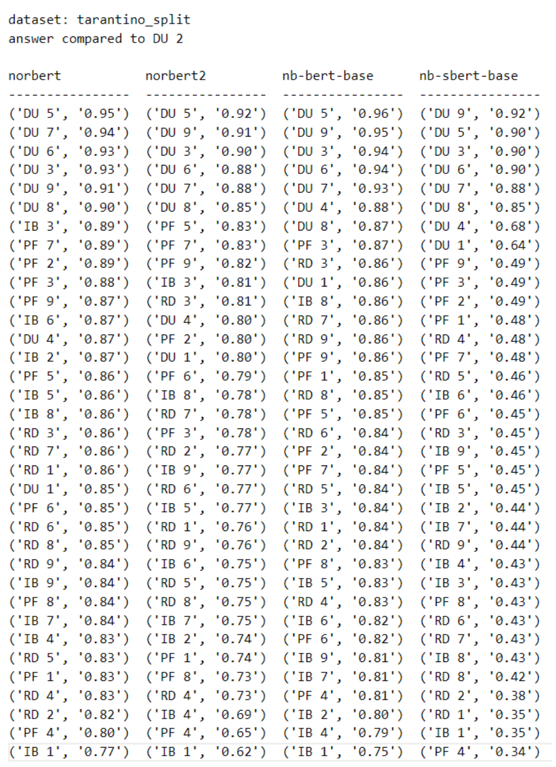
```
dataset: tarantino_split
answer compared to DU 1

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 9', '0.91')   ('DU 5', '0.84')   ('DU 5', '0.89')   ('DU 8', '0.76')
('DU 3', '0.89')   ('DU 9', '0.82')   ('DU 9', '0.88')   ('DU 5', '0.71')
('DU 5', '0.89')   ('RD 3', '0.81')   ('DU 6', '0.88')   ('DU 4', '0.70')
('RD 3', '0.88')   ('DU 3', '0.81')   ('DU 3', '0.88')   ('DU 6', '0.69')
('PF 7', '0.88')   ('RD 7', '0.81')   ('DU 4', '0.87')   ('DU 9', '0.68')
('PF 9', '0.88')   ('DU 2', '0.80')   ('DU 2', '0.86')   ('DU 3', '0.68')
('RD 7', '0.88')   ('RD 1', '0.80')   ('DU 7', '0.84')   ('DU 7', '0.65')
('PF 3', '0.87')   ('DU 7', '0.79')   ('RD 1', '0.84')   ('DU 2', '0.64')
('RD 1', '0.87')   ('DU 4', '0.79')   ('DU 8', '0.83')   ('PF 7', '0.48')
('DU 8', '0.87')   ('RD 2', '0.78')   ('PF 3', '0.83')   ('PF 3', '0.48')
('RD 5', '0.87')   ('DU 6', '0.78')   ('RD 3', '0.83')   ('PF 9', '0.47')
('PF 2', '0.86')   ('RD 8', '0.77')   ('RD 5', '0.82')   ('RD 5', '0.47')
('RD 6', '0.86')   ('DU 8', '0.76')   ('PF 2', '0.82')   ('PF 5', '0.47')
('IB 4', '0.86')   ('PF 7', '0.76')   ('PF 9', '0.82')   ('PF 2', '0.47')
('PF 5', '0.86')   ('IB 3', '0.76')   ('RD 9', '0.82')   ('PF 1', '0.44')
('IB 3', '0.86')   ('RD 6', '0.75')   ('IB 3', '0.82')   ('RD 3', '0.43')
('RD 2', '0.86')   ('RD 5', '0.75')   ('PF 1', '0.81')   ('IB 4', '0.43')
('DU 4', '0.86')   ('PF 5', '0.74')   ('IB 8', '0.81')   ('PF 8', '0.43')
('DU 2', '0.85')   ('IB 4', '0.74')   ('RD 2', '0.81')   ('RD 2', '0.41')
('RD 9', '0.85')   ('PF 9', '0.73')   ('RD 7', '0.81')   ('RD 7', '0.41')
('DU 7', '0.84')   ('PF 2', '0.73')   ('PF 7', '0.80')   ('RD 6', '0.40')
('RD 8', '0.84')   ('IB 5', '0.72')   ('RD 6', '0.80')   ('RD 4', '0.40')
('DU 6', '0.84')   ('RD 9', '0.72')   ('PF 5', '0.80')   ('RD 9', '0.40')
('IB 8', '0.83')   ('PF 6', '0.72')   ('PF 8', '0.80')   ('RD 8', '0.40')
('PF 6', '0.83')   ('IB 8', '0.72')   ('IB 4', '0.79')   ('IB 3', '0.39')
('RD 4', '0.83')   ('PF 3', '0.71')   ('RD 8', '0.79')   ('RD 1', '0.38')
('PF 1', '0.82')   ('PF 1', '0.71')   ('PF 6', '0.78')   ('IB 8', '0.38')
('PF 8', '0.82')   ('RD 4', '0.70')   ('IB 5', '0.77')   ('IB 2', '0.37')
('IB 2', '0.81')   ('IB 6', '0.70')   ('IB 1', '0.77')   ('IB 9', '0.36')
('IB 1', '0.81')   ('IB 1', '0.68')   ('IB 6', '0.77')   ('PF 6', '0.35')
('IB 5', '0.80')   ('IB 9', '0.68')   ('IB 2', '0.76')   ('IB 5', '0.34')
('IB 6', '0.80')   ('IB 7', '0.67')   ('RD 4', '0.76')   ('IB 1', '0.34')
('IB 9', '0.80')   ('IB 2', '0.67')   ('IB 9', '0.76')   ('IB 6', '0.34')
('PF 4', '0.78')   ('PF 4', '0.64')   ('PF 4', '0.75')   ('IB 7', '0.30')
('IB 7', '0.78')   ('PF 8', '0.64')   ('IB 7', '0.75')   ('PF 4', '0.26')
```

Figure .72: The results from comparison to entry labeled "DU 1".



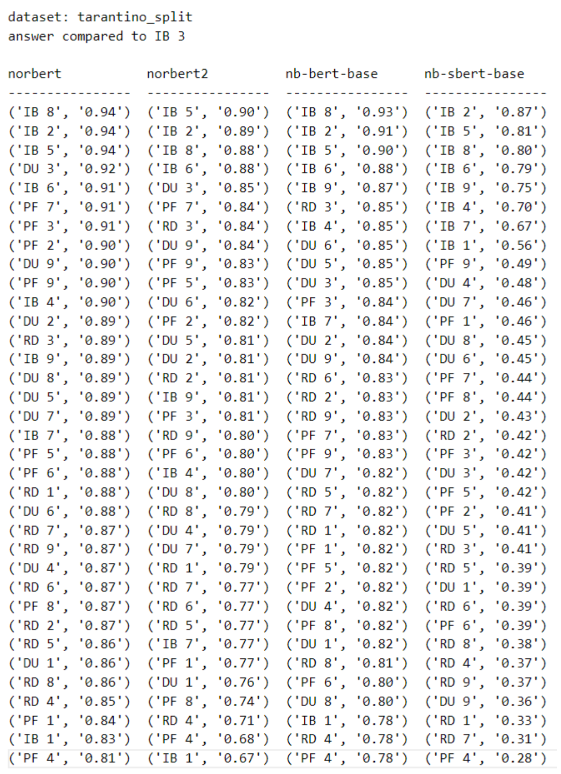
```
dataset: tarantino_split
answer compared to IB 2

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('IB 3', '0.94')   ('IB 6', '0.91')   ('IB 6', '0.92')   ('IB 6', '0.91')
('IB 8', '0.94')   ('IB 3', '0.89')   ('IB 8', '0.91')   ('IB 5', '0.91')
('IB 5', '0.94')   ('IB 5', '0.89')   ('IB 3', '0.91')   ('IB 9', '0.89')
('IB 6', '0.93')   ('IB 8', '0.86')   ('IB 9', '0.91')   ('IB 3', '0.87')
('IB 9', '0.91')   ('IB 9', '0.86')   ('IB 5', '0.90')   ('IB 8', '0.85')
('IB 7', '0.91')   ('IB 7', '0.81')   ('IB 7', '0.89')   ('IB 7', '0.83')
('DU 3', '0.89')   ('PF 7', '0.80')   ('RD 3', '0.82')   ('IB 4', '0.79')
('PF 3', '0.88')   ('PF 3', '0.80')   ('PF 3', '0.82')   ('IB 1', '0.68')
('PF 7', '0.88')   ('DU 6', '0.80')   ('PF 7', '0.82')   ('PF 8', '0.53')
('DU 7', '0.87')   ('RD 9', '0.79')   ('PF 8', '0.81')   ('PF 5', '0.50')
('IB 4', '0.87')   ('PF 2', '0.79')   ('PF 2', '0.81')   ('PF 9', '0.50')
('DU 2', '0.87')   ('PF 5', '0.78')   ('RD 9', '0.81')   ('PF 1', '0.50')
('RD 9', '0.87')   ('PF 9', '0.78')   ('RD 2', '0.81')   ('PF 7', '0.50')
('PF 9', '0.87')   ('DU 8', '0.78')   ('PF 5', '0.81')   ('PF 6', '0.49')
('PF 6', '0.86')   ('PF 6', '0.78')   ('DU 6', '0.81')   ('RD 3', '0.47')
('RD 3', '0.86')   ('DU 3', '0.78')   ('IB 4', '0.80')   ('DU 7', '0.47')
('DU 8', '0.86')   ('RD 3', '0.78')   ('RD 6', '0.80')   ('RD 5', '0.47')
('PF 8', '0.86')   ('PF 8', '0.77')   ('RD 8', '0.80')   ('RD 9', '0.46')
('PF 2', '0.86')   ('PF 1', '0.77')   ('PF 9', '0.80')   ('RD 2', '0.46')
('RD 1', '0.85')   ('DU 9', '0.77')   ('DU 7', '0.80')   ('DU 4', '0.45')
('DU 5', '0.85')   ('IB 4', '0.77')   ('RD 7', '0.80')   ('PF 3', '0.45')
('DU 6', '0.85')   ('RD 2', '0.76')   ('DU 3', '0.80')   ('PF 2', '0.45')
('PF 5', '0.85')   ('RD 1', '0.74')   ('DU 2', '0.80')   ('DU 6', '0.45')
('RD 7', '0.85')   ('DU 7', '0.74')   ('RD 5', '0.80')   ('DU 2', '0.44')
('RD 2', '0.85')   ('DU 5', '0.74')   ('PF 6', '0.80')   ('DU 8', '0.43')
('RD 6', '0.84')   ('DU 2', '0.74')   ('DU 5', '0.79')   ('RD 8', '0.43')
('DU 9', '0.84')   ('RD 5', '0.73')   ('PF 1', '0.79')   ('RD 6', '0.43')
('RD 5', '0.84')   ('RD 6', '0.73')   ('RD 1', '0.79')   ('RD 4', '0.42')
('RD 8', '0.83')   ('RD 8', '0.73')   ('DU 9', '0.79')   ('DU 3', '0.41')
('RD 4', '0.83')   ('RD 7', '0.72')   ('DU 8', '0.77')   ('DU 5', '0.40')
('DU 4', '0.82')   ('DU 4', '0.72')   ('DU 1', '0.76')   ('DU 1', '0.37')
('IB 1', '0.82')   ('PF 4', '0.71')   ('PF 4', '0.76')   ('RD 7', '0.37')
('PF 1', '0.81')   ('IB 1', '0.67')   ('DU 4', '0.76')   ('RD 1', '0.36')
('DU 1', '0.81')   ('DU 1', '0.67')   ('RD 4', '0.74')   ('PF 4', '0.36')
('PF 4', '0.80')   ('RD 4', '0.64')   ('IB 1', '0.74')   ('DU 9', '0.35')
```

Figure .73: The results from comparison to entry labeled "IB 2".



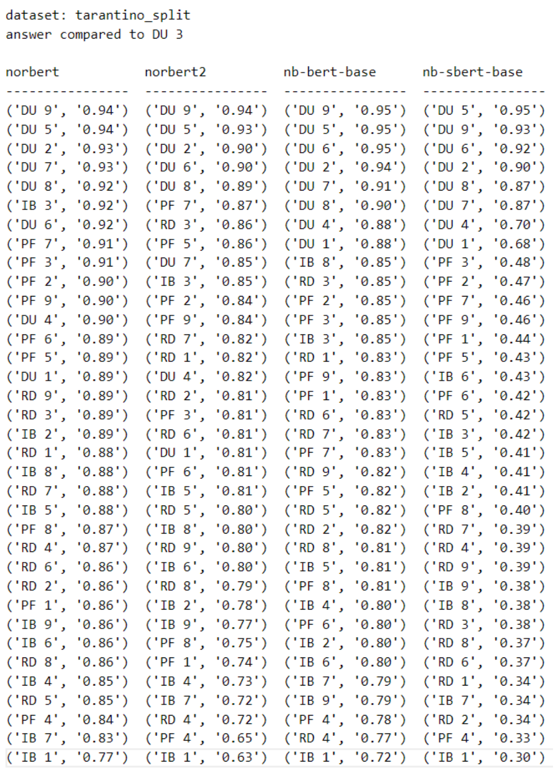
```
dataset: tarantino_split
answer compared to DU 2

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('DU 5', '0.95')    ('DU 5', '0.92')    ('DU 5', '0.96')    ('DU 9', '0.92')
('DU 7', '0.94')    ('DU 9', '0.91')    ('DU 9', '0.95')    ('DU 5', '0.90')
('DU 6', '0.93')    ('DU 3', '0.90')    ('DU 3', '0.94')    ('DU 3', '0.90')
('DU 3', '0.93')    ('DU 6', '0.88')    ('DU 6', '0.94')    ('DU 6', '0.90')
('DU 9', '0.91')    ('DU 7', '0.88')    ('DU 7', '0.93')    ('DU 7', '0.88')
('DU 8', '0.90')    ('DU 8', '0.85')    ('DU 4', '0.88')    ('DU 8', '0.85')
('IB 3', '0.89')    ('PF 5', '0.83')    ('DU 8', '0.87')    ('DU 4', '0.68')
('PF 7', '0.89')    ('PF 7', '0.83')    ('PF 3', '0.87')    ('DU 1', '0.64')
('PF 2', '0.89')    ('PF 9', '0.82')    ('RD 3', '0.86')    ('PF 9', '0.49')
('PF 3', '0.88')    ('IB 3', '0.81')    ('DU 1', '0.86')    ('PF 3', '0.49')
('PF 9', '0.87')    ('RD 3', '0.81')    ('IB 8', '0.86')    ('PF 2', '0.49')
('IB 6', '0.87')    ('DU 4', '0.80')    ('RD 7', '0.86')    ('PF 1', '0.48')
('DU 4', '0.87')    ('PF 2', '0.80')    ('RD 9', '0.86')    ('RD 4', '0.48')
('IB 2', '0.87')    ('DU 1', '0.80')    ('PF 9', '0.86')    ('PF 7', '0.48')
('PF 5', '0.86')    ('PF 6', '0.79')    ('PF 1', '0.85')    ('RD 5', '0.46')
('IB 5', '0.86')    ('IB 8', '0.78')    ('RD 8', '0.85')    ('IB 6', '0.46')
('IB 8', '0.86')    ('RD 7', '0.78')    ('PF 5', '0.85')    ('PF 6', '0.45')
('RD 3', '0.86')    ('PF 3', '0.78')    ('RD 6', '0.84')    ('RD 3', '0.45')
('RD 7', '0.86')    ('RD 2', '0.77')    ('PF 2', '0.84')    ('IB 9', '0.45')
('RD 1', '0.86')    ('IB 9', '0.77')    ('PF 7', '0.84')    ('PF 5', '0.45')
('DU 1', '0.85')    ('RD 6', '0.77')    ('RD 5', '0.84')    ('IB 5', '0.45')
('PF 6', '0.85')    ('IB 5', '0.77')    ('IB 3', '0.84')    ('IB 2', '0.44')
('RD 6', '0.85')    ('RD 1', '0.76')    ('RD 1', '0.84')    ('IB 7', '0.44')
('RD 8', '0.85')    ('RD 9', '0.76')    ('RD 2', '0.84')    ('RD 9', '0.44')
('RD 9', '0.84')    ('IB 6', '0.75')    ('PF 8', '0.83')    ('IB 4', '0.43')
('IB 9', '0.84')    ('RD 5', '0.75')    ('IB 5', '0.83')    ('IB 3', '0.43')
('PF 8', '0.84')    ('RD 8', '0.75')    ('RD 4', '0.83')    ('PF 8', '0.43')
('IB 7', '0.84')    ('IB 7', '0.75')    ('IB 6', '0.82')    ('RD 6', '0.43')
('IB 4', '0.83')    ('IB 2', '0.74')    ('PF 6', '0.82')    ('RD 7', '0.43')
('RD 5', '0.83')    ('PF 1', '0.74')    ('IB 9', '0.81')    ('IB 8', '0.43')
('PF 1', '0.83')    ('PF 8', '0.73')    ('IB 7', '0.81')    ('RD 8', '0.42')
('RD 4', '0.83')    ('RD 4', '0.73')    ('PF 4', '0.81')    ('RD 2', '0.38')
('RD 2', '0.82')    ('IB 4', '0.69')    ('IB 2', '0.80')    ('RD 1', '0.35')
('PF 4', '0.80')    ('PF 4', '0.65')    ('IB 4', '0.79')    ('IB 1', '0.35')
('IB 1', '0.77')    ('IB 1', '0.62')    ('IB 1', '0.75')    ('PF 4', '0.34')
```

Figure .74: The results from comparison to entry labeled "DU 2".



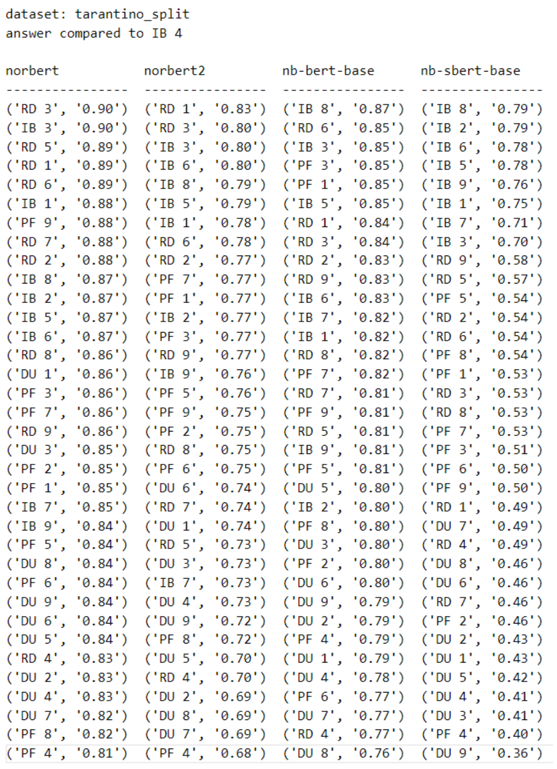
```
dataset: tarantino_split
answer compared to IB 3

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('IB 8', '0.94')    ('IB 5', '0.90')    ('IB 8', '0.93')    ('IB 2', '0.87')
('IB 2', '0.94')    ('IB 2', '0.89')    ('IB 2', '0.91')    ('IB 5', '0.81')
('IB 5', '0.94')    ('IB 8', '0.88')    ('IB 5', '0.90')    ('IB 8', '0.80')
('DU 3', '0.92')    ('IB 6', '0.88')    ('IB 6', '0.88')    ('IB 6', '0.79')
('IB 6', '0.91')    ('DU 3', '0.85')    ('IB 9', '0.87')    ('IB 9', '0.75')
('PF 7', '0.91')    ('PF 7', '0.84')    ('RD 3', '0.85')    ('IB 4', '0.70')
('PF 3', '0.91')    ('RD 3', '0.84')    ('IB 4', '0.85')    ('IB 7', '0.67')
('PF 2', '0.90')    ('DU 9', '0.84')    ('DU 6', '0.85')    ('IB 1', '0.56')
('DU 9', '0.90')    ('PF 9', '0.83')    ('DU 5', '0.85')    ('PF 9', '0.49')
('PF 9', '0.90')    ('PF 5', '0.83')    ('DU 3', '0.85')    ('DU 4', '0.48')
('IB 4', '0.90')    ('DU 6', '0.82')    ('PF 3', '0.84')    ('DU 7', '0.46')
('DU 2', '0.89')    ('PF 2', '0.82')    ('IB 7', '0.84')    ('PF 1', '0.46')
('RD 3', '0.89')    ('DU 5', '0.81')    ('DU 2', '0.84')    ('DU 8', '0.45')
('IB 9', '0.89')    ('DU 2', '0.81')    ('DU 9', '0.84')    ('DU 6', '0.45')
('DU 8', '0.89')    ('RD 2', '0.81')    ('RD 6', '0.83')    ('PF 7', '0.44')
('DU 5', '0.89')    ('IB 9', '0.81')    ('RD 2', '0.83')    ('PF 8', '0.44')
('DU 7', '0.89')    ('PF 3', '0.81')    ('RD 9', '0.83')    ('DU 2', '0.43')
('IB 7', '0.88')    ('RD 9', '0.80')    ('PF 7', '0.83')    ('RD 2', '0.42')
('PF 5', '0.88')    ('PF 6', '0.80')    ('PF 9', '0.83')    ('PF 3', '0.42')
('PF 6', '0.88')    ('IB 4', '0.80')    ('DU 7', '0.82')    ('DU 3', '0.42')
('RD 1', '0.88')    ('DU 8', '0.80')    ('RD 5', '0.82')    ('PF 5', '0.42')
('DU 6', '0.88')    ('RD 8', '0.79')    ('RD 7', '0.82')    ('PF 2', '0.41')
('RD 7', '0.87')    ('DU 4', '0.79')    ('RD 1', '0.82')    ('DU 5', '0.41')
('RD 9', '0.87')    ('DU 7', '0.79')    ('PF 1', '0.82')    ('RD 3', '0.41')
('DU 4', '0.87')    ('RD 1', '0.79')    ('PF 5', '0.82')    ('RD 5', '0.39')
('RD 6', '0.87')    ('RD 7', '0.77')    ('PF 2', '0.82')    ('DU 1', '0.39')
('PF 8', '0.87')    ('RD 6', '0.77')    ('DU 4', '0.82')    ('RD 6', '0.39')
('RD 2', '0.87')    ('RD 5', '0.77')    ('PF 8', '0.82')    ('PF 6', '0.39')
('RD 5', '0.86')    ('IB 7', '0.77')    ('DU 1', '0.82')    ('RD 8', '0.38')
('DU 1', '0.86')    ('PF 1', '0.77')    ('RD 8', '0.81')    ('RD 4', '0.37')
('RD 8', '0.86')    ('DU 1', '0.76')    ('PF 6', '0.80')    ('RD 9', '0.37')
('RD 4', '0.85')    ('PF 8', '0.74')    ('DU 8', '0.80')    ('DU 9', '0.36')
('PF 1', '0.84')    ('RD 4', '0.71')    ('IB 1', '0.78')    ('RD 1', '0.33')
('IB 1', '0.83')    ('PF 4', '0.68')    ('RD 4', '0.78')    ('RD 7', '0.31')
('PF 4', '0.81')    ('IB 1', '0.67')    ('PF 4', '0.78')    ('PF 4', '0.28')
```

Figure .75: The results from comparison to entry labeled "IB 3".



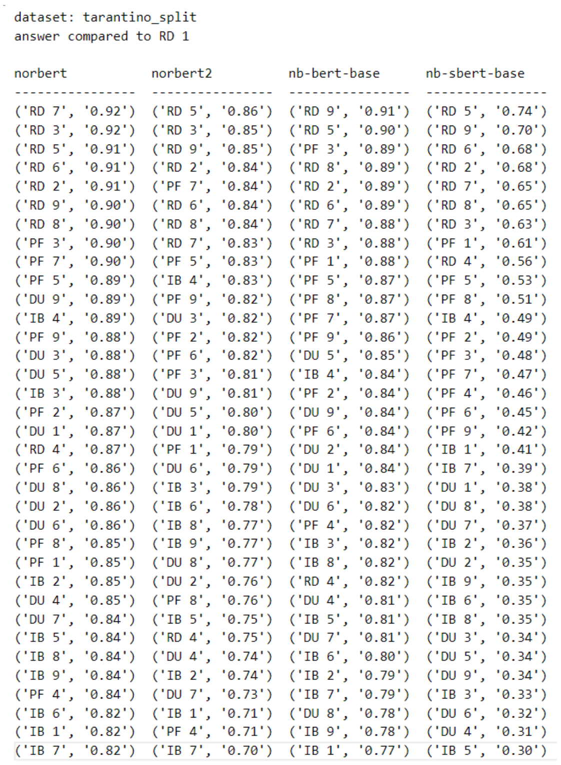
```
dataset: tarantino_split
answer compared to DU 3

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 9', '0.94')   ('DU 9', '0.94')   ('DU 9', '0.95')   ('DU 5', '0.95')
('DU 5', '0.94')   ('DU 5', '0.93')   ('DU 5', '0.95')   ('DU 9', '0.93')
('DU 2', '0.93')   ('DU 2', '0.90')   ('DU 6', '0.95')   ('DU 6', '0.92')
('DU 7', '0.93')   ('DU 6', '0.90')   ('DU 2', '0.94')   ('DU 2', '0.90')
('DU 8', '0.92')   ('DU 8', '0.89')   ('DU 7', '0.91')   ('DU 8', '0.87')
('IB 3', '0.92')   ('PF 7', '0.87')   ('DU 8', '0.90')   ('DU 7', '0.87')
('DU 6', '0.92')   ('RD 3', '0.86')   ('DU 4', '0.88')   ('DU 4', '0.70')
('PF 7', '0.91')   ('PF 5', '0.86')   ('DU 1', '0.88')   ('DU 1', '0.68')
('PF 3', '0.91')   ('DU 7', '0.85')   ('IB 8', '0.85')   ('PF 3', '0.48')
('PF 2', '0.90')   ('IB 3', '0.85')   ('RD 3', '0.85')   ('PF 2', '0.47')
('PF 9', '0.90')   ('PF 2', '0.84')   ('PF 2', '0.85')   ('PF 7', '0.46')
('DU 4', '0.90')   ('PF 9', '0.84')   ('PF 3', '0.85')   ('PF 9', '0.46')
('PF 6', '0.89')   ('RD 7', '0.82')   ('IB 3', '0.85')   ('PF 1', '0.44')
('PF 5', '0.89')   ('RD 1', '0.82')   ('RD 1', '0.83')   ('PF 5', '0.43')
('DU 1', '0.89')   ('DU 4', '0.82')   ('PF 9', '0.83')   ('IB 6', '0.43')
('RD 9', '0.89')   ('RD 2', '0.81')   ('PF 1', '0.83')   ('PF 6', '0.42')
('RD 3', '0.89')   ('PF 3', '0.81')   ('RD 6', '0.83')   ('RD 5', '0.42')
('IB 2', '0.89')   ('RD 6', '0.81')   ('RD 7', '0.83')   ('IB 3', '0.42')
('RD 1', '0.88')   ('DU 1', '0.81')   ('PF 7', '0.83')   ('IB 5', '0.41')
('IB 8', '0.88')   ('PF 6', '0.81')   ('RD 9', '0.82')   ('IB 4', '0.41')
('RD 7', '0.88')   ('IB 5', '0.81')   ('PF 5', '0.82')   ('IB 2', '0.41')
('IB 5', '0.88')   ('RD 5', '0.80')   ('RD 5', '0.82')   ('PF 8', '0.40')
('PF 8', '0.87')   ('IB 8', '0.80')   ('RD 2', '0.82')   ('RD 7', '0.39')
('RD 4', '0.87')   ('RD 9', '0.80')   ('RD 8', '0.81')   ('RD 4', '0.39')
('RD 6', '0.86')   ('IB 6', '0.80')   ('IB 5', '0.81')   ('RD 9', '0.39')
('RD 2', '0.86')   ('RD 8', '0.79')   ('PF 8', '0.81')   ('IB 9', '0.38')
('PF 1', '0.86')   ('IB 2', '0.78')   ('IB 4', '0.80')   ('IB 8', '0.38')
('IB 9', '0.86')   ('IB 9', '0.77')   ('PF 6', '0.80')   ('RD 3', '0.38')
('IB 6', '0.86')   ('PF 8', '0.75')   ('IB 2', '0.80')   ('RD 8', '0.37')
('RD 8', '0.86')   ('PF 1', '0.74')   ('IB 6', '0.80')   ('RD 6', '0.37')
('IB 4', '0.85')   ('IB 4', '0.73')   ('IB 7', '0.79')   ('RD 1', '0.34')
('RD 5', '0.85')   ('IB 7', '0.72')   ('IB 9', '0.79')   ('IB 7', '0.34')
('PF 4', '0.84')   ('RD 4', '0.72')   ('PF 4', '0.78')   ('RD 2', '0.34')
('IB 7', '0.83')   ('PF 4', '0.65')   ('RD 4', '0.77')   ('PF 4', '0.33')
('IB 1', '0.77')   ('IB 1', '0.63')   ('IB 1', '0.72')   ('IB 1', '0.30')
```

Figure .76: The results from comparison to entry labeled "DU 3".



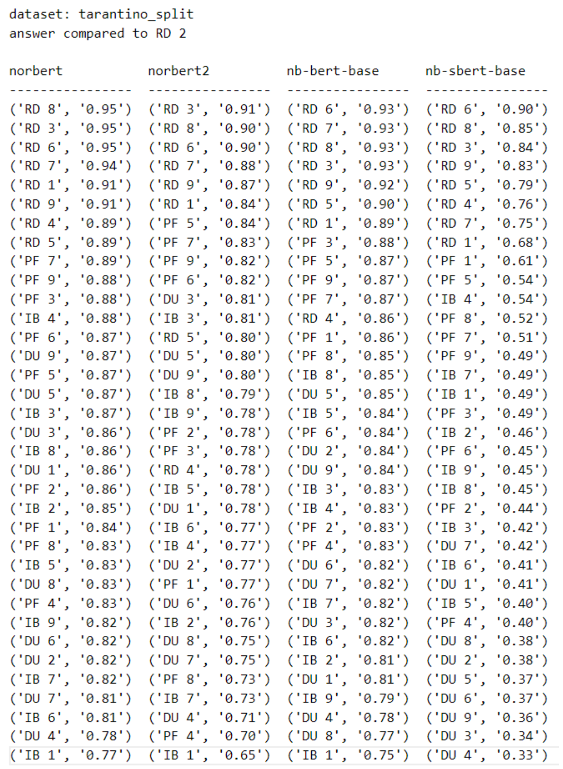
```
dataset: tarantino_split
answer compared to IB 4

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('RD 3', '0.90')   ('RD 1', '0.83')   ('IB 8', '0.87')   ('IB 8', '0.79')
('IB 3', '0.90')   ('RD 3', '0.80')   ('RD 6', '0.85')   ('IB 2', '0.79')
('RD 5', '0.89')   ('IB 3', '0.80')   ('IB 3', '0.85')   ('IB 6', '0.78')
('RD 1', '0.89')   ('IB 6', '0.80')   ('PF 3', '0.85')   ('IB 5', '0.78')
('RD 6', '0.89')   ('IB 8', '0.79')   ('PF 1', '0.85')   ('IB 9', '0.76')
('IB 1', '0.88')   ('IB 5', '0.79')   ('IB 5', '0.85')   ('IB 1', '0.75')
('PF 9', '0.88')   ('IB 1', '0.78')   ('RD 1', '0.84')   ('IB 7', '0.71')
('RD 7', '0.88')   ('RD 6', '0.78')   ('RD 3', '0.84')   ('IB 3', '0.70')
('RD 2', '0.88')   ('RD 2', '0.77')   ('RD 2', '0.83')   ('RD 9', '0.58')
('IB 8', '0.87')   ('PF 7', '0.77')   ('RD 9', '0.83')   ('RD 5', '0.57')
('IB 2', '0.87')   ('PF 1', '0.77')   ('IB 6', '0.83')   ('PF 5', '0.54')
('IB 5', '0.87')   ('IB 2', '0.77')   ('IB 7', '0.82')   ('RD 2', '0.54')
('IB 6', '0.87')   ('PF 3', '0.77')   ('IB 1', '0.82')   ('RD 6', '0.54')
('RD 8', '0.86')   ('RD 9', '0.77')   ('RD 8', '0.82')   ('PF 8', '0.54')
('DU 1', '0.86')   ('IB 9', '0.76')   ('PF 7', '0.82')   ('PF 1', '0.53')
('PF 3', '0.86')   ('PF 5', '0.76')   ('RD 7', '0.81')   ('RD 3', '0.53')
('PF 7', '0.86')   ('PF 9', '0.75')   ('PF 9', '0.81')   ('RD 8', '0.53')
('RD 9', '0.86')   ('PF 2', '0.75')   ('RD 5', '0.81')   ('PF 7', '0.53')
('DU 3', '0.85')   ('RD 8', '0.75')   ('IB 9', '0.81')   ('PF 3', '0.51')
('PF 2', '0.85')   ('PF 6', '0.75')   ('PF 5', '0.81')   ('PF 6', '0.50')
('PF 1', '0.85')   ('DU 6', '0.74')   ('DU 5', '0.80')   ('PF 9', '0.50')
('IB 7', '0.85')   ('RD 7', '0.74')   ('IB 2', '0.80')   ('RD 1', '0.49')
('IB 9', '0.84')   ('DU 1', '0.74')   ('PF 8', '0.80')   ('DU 7', '0.49')
('PF 5', '0.84')   ('RD 5', '0.73')   ('DU 3', '0.80')   ('RD 4', '0.49')
('DU 8', '0.84')   ('DU 3', '0.73')   ('PF 2', '0.80')   ('DU 8', '0.46')
('PF 6', '0.84')   ('IB 7', '0.73')   ('DU 6', '0.80')   ('DU 6', '0.46')
('DU 9', '0.84')   ('DU 4', '0.73')   ('DU 9', '0.79')   ('RD 7', '0.46')
('DU 6', '0.84')   ('DU 9', '0.72')   ('DU 2', '0.79')   ('PF 2', '0.46')
('DU 5', '0.84')   ('PF 8', '0.72')   ('PF 4', '0.79')   ('DU 2', '0.43')
('RD 4', '0.83')   ('DU 5', '0.70')   ('DU 1', '0.79')   ('DU 1', '0.43')
('DU 2', '0.83')   ('RD 4', '0.70')   ('DU 4', '0.78')   ('DU 5', '0.42')
('DU 4', '0.83')   ('DU 2', '0.69')   ('PF 6', '0.77')   ('DU 4', '0.41')
('DU 7', '0.82')   ('DU 8', '0.69')   ('DU 7', '0.77')   ('DU 3', '0.41')
('PF 8', '0.82')   ('DU 7', '0.69')   ('RD 4', '0.77')   ('PF 4', '0.40')
('PF 4', '0.81')   ('PF 4', '0.68')   ('DU 8', '0.76')   ('DU 9', '0.36')
```

Figure .77: The results from comparison to entry labeled "IB 4".



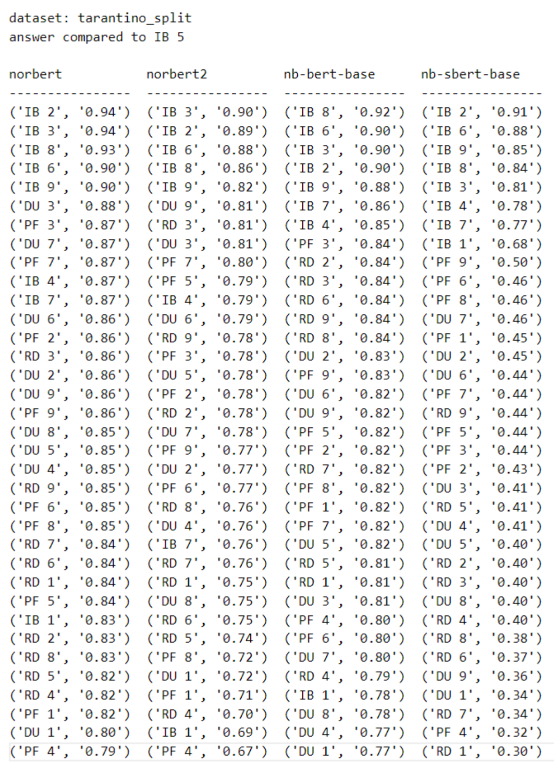
```
dataset: tarantino_split
answer compared to RD 1

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('RD 7', '0.92')    ('RD 5', '0.86')    ('RD 9', '0.91')    ('RD 5', '0.74')
('RD 3', '0.92')    ('RD 3', '0.85')    ('RD 5', '0.90')    ('RD 9', '0.70')
('RD 5', '0.91')    ('RD 9', '0.85')    ('PF 3', '0.89')    ('RD 6', '0.68')
('RD 6', '0.91')    ('RD 2', '0.84')    ('RD 8', '0.89')    ('RD 2', '0.68')
('RD 2', '0.91')    ('PF 7', '0.84')    ('RD 2', '0.89')    ('RD 7', '0.65')
('RD 9', '0.90')    ('RD 6', '0.84')    ('RD 6', '0.89')    ('RD 8', '0.65')
('RD 8', '0.90')    ('RD 8', '0.84')    ('RD 7', '0.88')    ('RD 3', '0.63')
('PF 3', '0.90')    ('RD 7', '0.83')    ('RD 3', '0.88')    ('PF 1', '0.61')
('PF 7', '0.90')    ('PF 5', '0.83')    ('PF 1', '0.88')    ('RD 4', '0.56')
('PF 5', '0.89')    ('IB 4', '0.83')    ('PF 5', '0.87')    ('PF 5', '0.53')
('DU 9', '0.89')    ('PF 9', '0.82')    ('PF 8', '0.87')    ('PF 8', '0.51')
('IB 4', '0.89')    ('DU 3', '0.82')    ('PF 7', '0.87')    ('IB 4', '0.49')
('PF 9', '0.88')    ('PF 2', '0.82')    ('PF 9', '0.86')    ('PF 2', '0.49')
('DU 3', '0.88')    ('PF 6', '0.82')    ('DU 5', '0.85')    ('PF 3', '0.48')
('DU 5', '0.88')    ('PF 3', '0.81')    ('IB 4', '0.84')    ('PF 7', '0.47')
('IB 3', '0.88')    ('DU 9', '0.81')    ('PF 2', '0.84')    ('PF 4', '0.46')
('PF 2', '0.87')    ('DU 5', '0.80')    ('DU 9', '0.84')    ('PF 6', '0.45')
('DU 1', '0.87')    ('DU 1', '0.80')    ('PF 6', '0.84')    ('PF 9', '0.42')
('RD 4', '0.87')    ('PF 1', '0.79')    ('DU 2', '0.84')    ('IB 1', '0.41')
('PF 6', '0.86')    ('DU 6', '0.79')    ('DU 1', '0.84')    ('IB 7', '0.39')
('DU 8', '0.86')    ('IB 3', '0.79')    ('DU 3', '0.83')    ('DU 1', '0.38')
('DU 2', '0.86')    ('IB 6', '0.78')    ('DU 6', '0.82')    ('DU 8', '0.38')
('DU 6', '0.86')    ('IB 8', '0.77')    ('PF 4', '0.82')    ('DU 7', '0.37')
('PF 8', '0.85')    ('IB 9', '0.77')    ('IB 3', '0.82')    ('IB 2', '0.36')
('PF 1', '0.85')    ('DU 8', '0.77')    ('IB 8', '0.82')    ('DU 2', '0.35')
('IB 2', '0.85')    ('DU 2', '0.76')    ('RD 4', '0.82')    ('IB 9', '0.35')
('DU 4', '0.85')    ('PF 8', '0.76')    ('DU 4', '0.81')    ('IB 6', '0.35')
('DU 7', '0.84')    ('IB 5', '0.75')    ('IB 5', '0.81')    ('IB 8', '0.35')
('IB 5', '0.84')    ('RD 4', '0.75')    ('DU 7', '0.81')    ('DU 3', '0.34')
('IB 8', '0.84')    ('DU 4', '0.74')    ('IB 6', '0.80')    ('DU 5', '0.34')
('IB 9', '0.84')    ('IB 2', '0.74')    ('IB 2', '0.79')    ('DU 9', '0.34')
('PF 4', '0.84')    ('DU 7', '0.73')    ('IB 7', '0.79')    ('IB 3', '0.33')
('IB 6', '0.82')    ('IB 1', '0.71')    ('DU 8', '0.78')    ('DU 6', '0.32')
('IB 1', '0.82')    ('PF 4', '0.71')    ('IB 9', '0.78')    ('DU 4', '0.31')
('IB 7', '0.82')    ('IB 7', '0.70')    ('IB 1', '0.77')    ('IB 5', '0.30')
```

Figure .78: The results from comparison to entry labeled "RD 1".



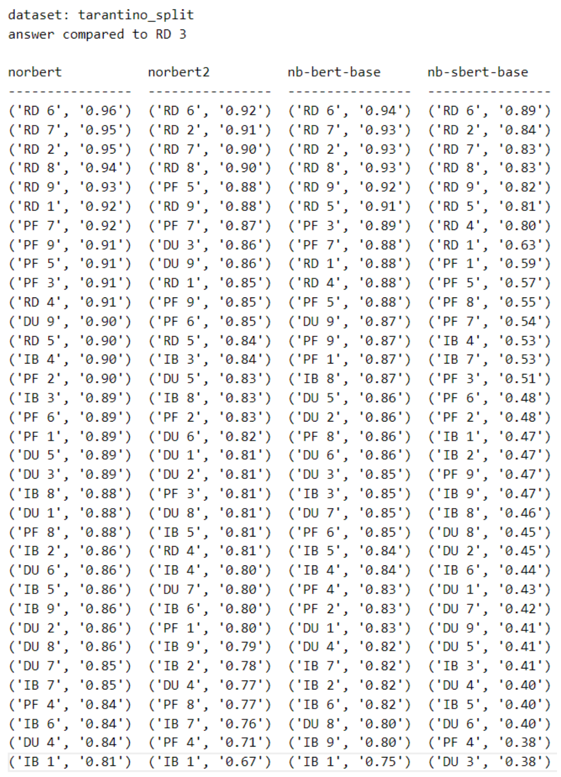
```
dataset: tarantino_split
answer compared to RD 2

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('RD 8', '0.95')    ('RD 3', '0.91')    ('RD 6', '0.93')    ('RD 6', '0.90')
('RD 3', '0.95')    ('RD 8', '0.90')    ('RD 7', '0.93')    ('RD 8', '0.85')
('RD 6', '0.95')    ('RD 6', '0.90')    ('RD 8', '0.93')    ('RD 3', '0.84')
('RD 7', '0.94')    ('RD 7', '0.88')    ('RD 3', '0.93')    ('RD 9', '0.83')
('RD 1', '0.91')    ('RD 9', '0.87')    ('RD 9', '0.92')    ('RD 5', '0.79')
('RD 9', '0.91')    ('RD 1', '0.84')    ('RD 5', '0.90')    ('RD 4', '0.76')
('RD 4', '0.89')    ('PF 5', '0.84')    ('RD 1', '0.89')    ('RD 7', '0.75')
('RD 5', '0.89')    ('PF 7', '0.83')    ('PF 3', '0.88')    ('RD 1', '0.68')
('PF 7', '0.89')    ('PF 9', '0.82')    ('PF 5', '0.87')    ('PF 1', '0.61')
('PF 9', '0.88')    ('PF 6', '0.82')    ('PF 9', '0.87')    ('PF 5', '0.54')
('PF 3', '0.88')    ('DU 3', '0.81')    ('PF 7', '0.87')    ('IB 4', '0.54')
('IB 4', '0.88')    ('IB 3', '0.81')    ('RD 4', '0.86')    ('PF 8', '0.52')
('PF 6', '0.87')    ('RD 5', '0.80')    ('PF 1', '0.86')    ('PF 7', '0.51')
('DU 9', '0.87')    ('DU 5', '0.80')    ('PF 8', '0.85')    ('PF 9', '0.49')
('PF 5', '0.87')    ('DU 9', '0.80')    ('IB 8', '0.85')    ('IB 7', '0.49')
('DU 5', '0.87')    ('IB 8', '0.79')    ('DU 5', '0.85')    ('IB 1', '0.49')
('IB 3', '0.87')    ('IB 9', '0.78')    ('IB 5', '0.84')    ('PF 3', '0.49')
('DU 3', '0.86')    ('PF 2', '0.78')    ('PF 6', '0.84')    ('IB 2', '0.46')
('IB 8', '0.86')    ('PF 3', '0.78')    ('DU 2', '0.84')    ('PF 6', '0.45')
('DU 1', '0.86')    ('RD 4', '0.78')    ('DU 9', '0.84')    ('IB 9', '0.45')
('PF 2', '0.86')    ('IB 5', '0.78')    ('IB 3', '0.83')    ('IB 8', '0.45')
('IB 2', '0.85')    ('DU 1', '0.78')    ('IB 4', '0.83')    ('PF 2', '0.44')
('PF 1', '0.84')    ('IB 6', '0.77')    ('PF 2', '0.83')    ('IB 3', '0.42')
('PF 8', '0.83')    ('IB 4', '0.77')    ('PF 4', '0.83')    ('DU 7', '0.42')
('IB 5', '0.83')    ('DU 2', '0.77')    ('DU 6', '0.82')    ('IB 6', '0.41')
('DU 8', '0.83')    ('PF 1', '0.77')    ('DU 7', '0.82')    ('DU 1', '0.41')
('PF 4', '0.83')    ('DU 6', '0.76')    ('IB 7', '0.82')    ('IB 5', '0.40')
('IB 9', '0.82')    ('IB 2', '0.76')    ('DU 3', '0.82')    ('PF 4', '0.40')
('DU 6', '0.82')    ('DU 8', '0.75')    ('IB 6', '0.82')    ('DU 8', '0.38')
('DU 2', '0.82')    ('DU 7', '0.75')    ('IB 2', '0.81')    ('DU 2', '0.38')
('IB 7', '0.82')    ('PF 8', '0.73')    ('DU 1', '0.81')    ('DU 5', '0.37')
('DU 7', '0.81')    ('IB 7', '0.73')    ('IB 9', '0.79')    ('DU 6', '0.37')
('IB 6', '0.81')    ('DU 4', '0.71')    ('DU 4', '0.78')    ('DU 9', '0.36')
('DU 4', '0.78')    ('PF 4', '0.70')    ('DU 8', '0.77')    ('DU 3', '0.34')
('IB 1', '0.77')    ('IB 1', '0.65')    ('IB 1', '0.75')    ('DU 4', '0.33')
```

Figure .79: The results from comparison to entry labeled "RD 2".



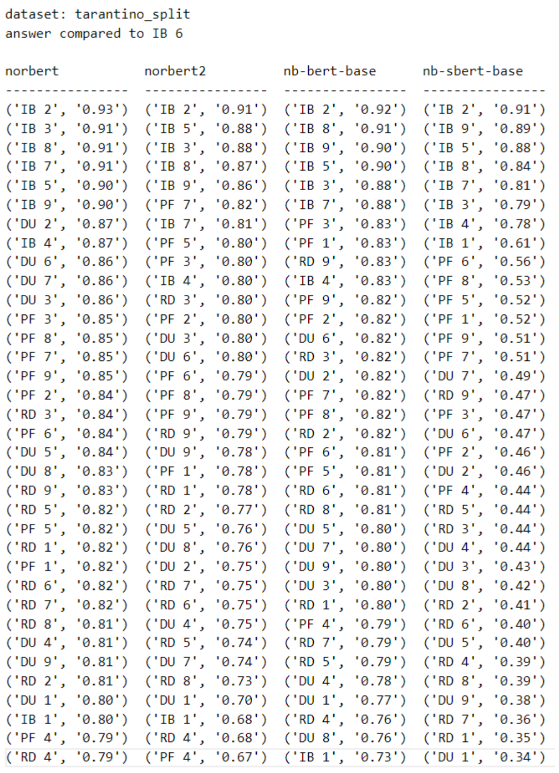
```
dataset: tarantino_split
answer compared to IB 5

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('IB 2', '0.94')   ('IB 3', '0.90')   ('IB 8', '0.92')   ('IB 2', '0.91')
('IB 3', '0.94')   ('IB 2', '0.89')   ('IB 6', '0.90')   ('IB 6', '0.88')
('IB 8', '0.93')   ('IB 6', '0.88')   ('IB 3', '0.90')   ('IB 9', '0.85')
('IB 6', '0.90')   ('IB 8', '0.86')   ('IB 2', '0.90')   ('IB 8', '0.84')
('IB 9', '0.90')   ('IB 9', '0.82')   ('IB 9', '0.88')   ('IB 3', '0.81')
('DU 3', '0.88')   ('DU 9', '0.81')   ('IB 7', '0.86')   ('IB 4', '0.78')
('PF 3', '0.87')   ('RD 3', '0.81')   ('IB 4', '0.85')   ('IB 7', '0.77')
('DU 7', '0.87')   ('DU 3', '0.81')   ('PF 3', '0.84')   ('IB 1', '0.68')
('PF 7', '0.87')   ('PF 7', '0.80')   ('RD 2', '0.84')   ('PF 9', '0.50')
('IB 4', '0.87')   ('PF 5', '0.79')   ('RD 3', '0.84')   ('PF 6', '0.46')
('IB 7', '0.87')   ('IB 4', '0.79')   ('RD 6', '0.84')   ('PF 8', '0.46')
('DU 6', '0.86')   ('DU 6', '0.79')   ('RD 9', '0.84')   ('DU 7', '0.46')
('PF 2', '0.86')   ('RD 9', '0.78')   ('RD 8', '0.84')   ('PF 1', '0.45')
('RD 3', '0.86')   ('PF 3', '0.78')   ('DU 2', '0.83')   ('DU 2', '0.45')
('DU 2', '0.86')   ('DU 5', '0.78')   ('PF 9', '0.83')   ('DU 6', '0.44')
('DU 9', '0.86')   ('PF 2', '0.78')   ('DU 6', '0.82')   ('PF 7', '0.44')
('PF 9', '0.86')   ('RD 2', '0.78')   ('DU 9', '0.82')   ('RD 9', '0.44')
('DU 8', '0.85')   ('DU 7', '0.78')   ('PF 5', '0.82')   ('PF 5', '0.44')
('DU 5', '0.85')   ('PF 9', '0.77')   ('PF 2', '0.82')   ('PF 3', '0.44')
('DU 4', '0.85')   ('DU 2', '0.77')   ('RD 7', '0.82')   ('PF 2', '0.43')
('RD 9', '0.85')   ('PF 6', '0.77')   ('PF 8', '0.82')   ('DU 3', '0.41')
('PF 6', '0.85')   ('RD 8', '0.76')   ('PF 1', '0.82')   ('RD 5', '0.41')
('PF 8', '0.85')   ('DU 4', '0.76')   ('PF 7', '0.82')   ('DU 4', '0.41')
('RD 7', '0.84')   ('IB 7', '0.76')   ('DU 5', '0.82')   ('DU 5', '0.40')
('RD 6', '0.84')   ('RD 7', '0.76')   ('RD 5', '0.81')   ('RD 2', '0.40')
('RD 1', '0.84')   ('RD 1', '0.75')   ('RD 1', '0.81')   ('RD 3', '0.40')
('PF 5', '0.84')   ('DU 8', '0.75')   ('DU 3', '0.81')   ('DU 8', '0.40')
('IB 1', '0.83')   ('RD 6', '0.75')   ('PF 4', '0.80')   ('RD 4', '0.40')
('RD 2', '0.83')   ('RD 5', '0.74')   ('PF 6', '0.80')   ('RD 8', '0.38')
('RD 8', '0.83')   ('PF 8', '0.72')   ('DU 7', '0.80')   ('RD 6', '0.37')
('RD 5', '0.82')   ('DU 1', '0.72')   ('RD 4', '0.79')   ('DU 9', '0.36')
('RD 4', '0.82')   ('PF 1', '0.71')   ('IB 1', '0.78')   ('DU 1', '0.34')
('PF 1', '0.82')   ('RD 4', '0.70')   ('DU 8', '0.78')   ('RD 7', '0.34')
('DU 1', '0.80')   ('IB 1', '0.69')   ('DU 4', '0.77')   ('PF 4', '0.32')
('PF 4', '0.79')   ('PF 4', '0.67')   ('DU 1', '0.77')   ('RD 1', '0.30')
```

Figure .80: The results from comparison to entry labeled "IB 5".



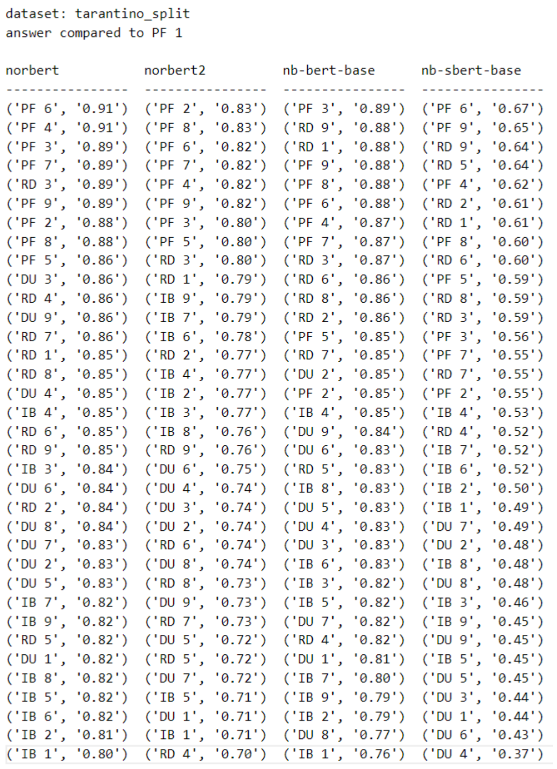
```
dataset: tarantino_split
answer compared to RD 3

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('RD 6', '0.96')   ('RD 6', '0.92')   ('RD 6', '0.94')   ('RD 6', '0.89')
('RD 7', '0.95')   ('RD 2', '0.91')   ('RD 7', '0.93')   ('RD 2', '0.84')
('RD 2', '0.95')   ('RD 7', '0.90')   ('RD 2', '0.93')   ('RD 7', '0.83')
('RD 8', '0.94')   ('RD 8', '0.90')   ('RD 8', '0.93')   ('RD 8', '0.83')
('RD 9', '0.93')   ('PF 5', '0.88')   ('RD 9', '0.92')   ('RD 9', '0.82')
('RD 1', '0.92')   ('RD 9', '0.88')   ('RD 5', '0.91')   ('RD 5', '0.81')
('PF 7', '0.92')   ('PF 7', '0.87')   ('PF 3', '0.89')   ('RD 4', '0.80')
('PF 9', '0.91')   ('DU 3', '0.86')   ('PF 7', '0.88')   ('RD 1', '0.63')
('PF 5', '0.91')   ('DU 9', '0.86')   ('RD 1', '0.88')   ('PF 1', '0.59')
('PF 3', '0.91')   ('RD 1', '0.85')   ('RD 4', '0.88')   ('PF 5', '0.57')
('RD 4', '0.91')   ('PF 9', '0.85')   ('PF 5', '0.88')   ('PF 8', '0.55')
('DU 9', '0.90')   ('PF 6', '0.85')   ('DU 9', '0.87')   ('PF 7', '0.54')
('RD 5', '0.90')   ('RD 5', '0.84')   ('PF 9', '0.87')   ('IB 4', '0.53')
('IB 4', '0.90')   ('IB 3', '0.84')   ('PF 1', '0.87')   ('IB 7', '0.53')
('PF 2', '0.90')   ('DU 5', '0.83')   ('IB 8', '0.87')   ('PF 3', '0.51')
('IB 3', '0.89')   ('IB 8', '0.83')   ('DU 5', '0.86')   ('PF 6', '0.48')
('PF 6', '0.89')   ('PF 2', '0.83')   ('DU 2', '0.86')   ('PF 2', '0.48')
('PF 1', '0.89')   ('DU 6', '0.82')   ('PF 8', '0.86')   ('IB 1', '0.47')
('DU 5', '0.89')   ('DU 1', '0.81')   ('DU 6', '0.86')   ('IB 2', '0.47')
('DU 3', '0.89')   ('DU 2', '0.81')   ('DU 3', '0.85')   ('PF 9', '0.47')
('IB 8', '0.88')   ('PF 3', '0.81')   ('IB 3', '0.85')   ('IB 9', '0.47')
('DU 1', '0.88')   ('DU 8', '0.81')   ('DU 7', '0.85')   ('IB 8', '0.46')
('PF 8', '0.88')   ('IB 5', '0.81')   ('PF 6', '0.85')   ('DU 8', '0.45')
('IB 2', '0.86')   ('RD 4', '0.81')   ('IB 5', '0.84')   ('DU 2', '0.45')
('DU 6', '0.86')   ('IB 4', '0.80')   ('IB 4', '0.84')   ('IB 6', '0.44')
('IB 5', '0.86')   ('DU 7', '0.80')   ('PF 4', '0.83')   ('DU 1', '0.43')
('IB 9', '0.86')   ('IB 6', '0.80')   ('PF 2', '0.83')   ('DU 7', '0.42')
('DU 2', '0.86')   ('PF 1', '0.80')   ('DU 1', '0.83')   ('DU 9', '0.41')
('DU 8', '0.86')   ('IB 9', '0.79')   ('DU 4', '0.82')   ('DU 5', '0.41')
('DU 7', '0.85')   ('IB 2', '0.78')   ('IB 7', '0.82')   ('IB 3', '0.41')
('IB 7', '0.85')   ('DU 4', '0.77')   ('IB 2', '0.82')   ('DU 4', '0.40')
('PF 4', '0.84')   ('PF 8', '0.77')   ('IB 6', '0.82')   ('IB 5', '0.40')
('IB 6', '0.84')   ('IB 7', '0.76')   ('DU 8', '0.80')   ('DU 6', '0.40')
('DU 4', '0.84')   ('PF 4', '0.71')   ('IB 9', '0.80')   ('PF 4', '0.38')
('IB 1', '0.81')   ('IB 1', '0.67')   ('IB 1', '0.75')   ('DU 3', '0.38')
```

Figure .81: The results from comparison to entry labeled "RD 3".



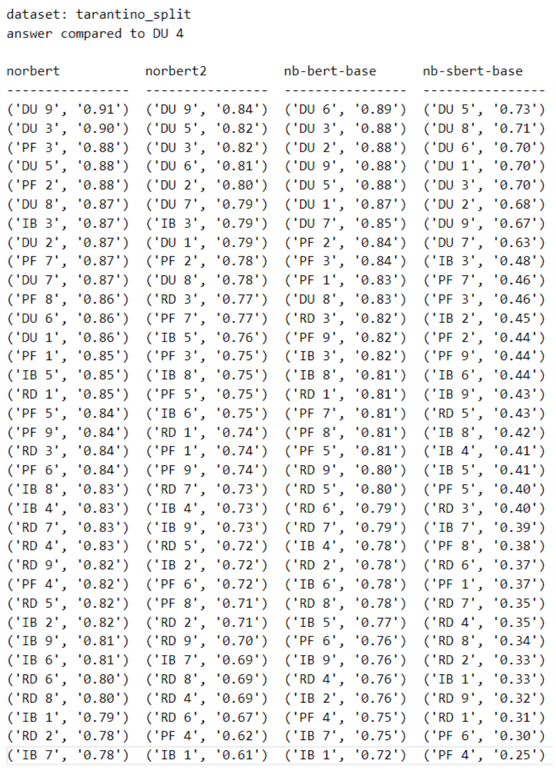
```
dataset: tarantino_split
answer compared to IB 6

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('IB 2', '0.93')  ('IB 2', '0.91')  ('IB 2', '0.92')  ('IB 2', '0.91')
('IB 3', '0.91')  ('IB 5', '0.88')  ('IB 8', '0.91')  ('IB 9', '0.89')
('IB 8', '0.91')  ('IB 3', '0.88')  ('IB 9', '0.90')  ('IB 5', '0.88')
('IB 7', '0.91')  ('IB 8', '0.87')  ('IB 5', '0.90')  ('IB 8', '0.84')
('IB 5', '0.90')  ('IB 9', '0.86')  ('IB 3', '0.88')  ('IB 7', '0.81')
('IB 9', '0.90')  ('PF 7', '0.82')  ('IB 7', '0.88')  ('IB 3', '0.79')
('DU 2', '0.87')  ('IB 7', '0.81')  ('PF 3', '0.83')  ('IB 4', '0.78')
('IB 4', '0.87')  ('PF 5', '0.80')  ('PF 1', '0.83')  ('IB 1', '0.61')
('DU 6', '0.86')  ('PF 3', '0.80')  ('RD 9', '0.83')  ('PF 6', '0.56')
('DU 7', '0.86')  ('IB 4', '0.80')  ('IB 4', '0.83')  ('PF 8', '0.53')
('DU 3', '0.86')  ('RD 3', '0.80')  ('PF 9', '0.82')  ('PF 5', '0.52')
('PF 3', '0.85')  ('PF 2', '0.80')  ('PF 2', '0.82')  ('PF 1', '0.52')
('PF 8', '0.85')  ('DU 3', '0.80')  ('DU 6', '0.82')  ('PF 9', '0.51')
('PF 7', '0.85')  ('DU 6', '0.80')  ('RD 3', '0.82')  ('PF 7', '0.51')
('PF 9', '0.85')  ('PF 6', '0.79')  ('DU 2', '0.82')  ('DU 7', '0.49')
('PF 2', '0.84')  ('PF 8', '0.79')  ('PF 7', '0.82')  ('RD 9', '0.47')
('RD 3', '0.84')  ('PF 9', '0.79')  ('PF 8', '0.82')  ('PF 3', '0.47')
('PF 6', '0.84')  ('RD 9', '0.79')  ('RD 2', '0.82')  ('DU 6', '0.47')
('DU 5', '0.84')  ('DU 9', '0.78')  ('PF 6', '0.81')  ('PF 2', '0.46')
('DU 8', '0.83')  ('PF 1', '0.78')  ('PF 5', '0.81')  ('DU 2', '0.46')
('RD 9', '0.83')  ('RD 1', '0.78')  ('RD 6', '0.81')  ('PF 4', '0.44')
('RD 5', '0.82')  ('RD 2', '0.77')  ('RD 8', '0.81')  ('RD 5', '0.44')
('PF 5', '0.82')  ('DU 5', '0.76')  ('DU 5', '0.80')  ('RD 3', '0.44')
('RD 1', '0.82')  ('DU 8', '0.76')  ('DU 7', '0.80')  ('DU 4', '0.44')
('PF 1', '0.82')  ('DU 2', '0.75')  ('DU 9', '0.80')  ('DU 3', '0.43')
('RD 6', '0.82')  ('RD 7', '0.75')  ('DU 3', '0.80')  ('DU 8', '0.42')
('RD 7', '0.82')  ('RD 6', '0.75')  ('RD 1', '0.80')  ('RD 2', '0.41')
('RD 8', '0.81')  ('DU 4', '0.75')  ('PF 4', '0.79')  ('RD 6', '0.40')
('DU 4', '0.81')  ('RD 5', '0.74')  ('RD 7', '0.79')  ('DU 5', '0.40')
('DU 9', '0.81')  ('DU 7', '0.74')  ('RD 5', '0.79')  ('RD 4', '0.39')
('RD 2', '0.81')  ('RD 8', '0.73')  ('DU 4', '0.78')  ('RD 8', '0.39')
('DU 1', '0.80')  ('DU 1', '0.70')  ('DU 1', '0.77')  ('DU 9', '0.38')
('IB 1', '0.80')  ('IB 1', '0.68')  ('RD 4', '0.76')  ('RD 7', '0.36')
('PF 4', '0.79')  ('RD 4', '0.68')  ('DU 8', '0.76')  ('RD 1', '0.35')
('RD 4', '0.79')  ('PF 4', '0.67')  ('IB 1', '0.73')  ('DU 1', '0.34')
```

Figure .82: The results from comparison to entry labeled "IB 6".



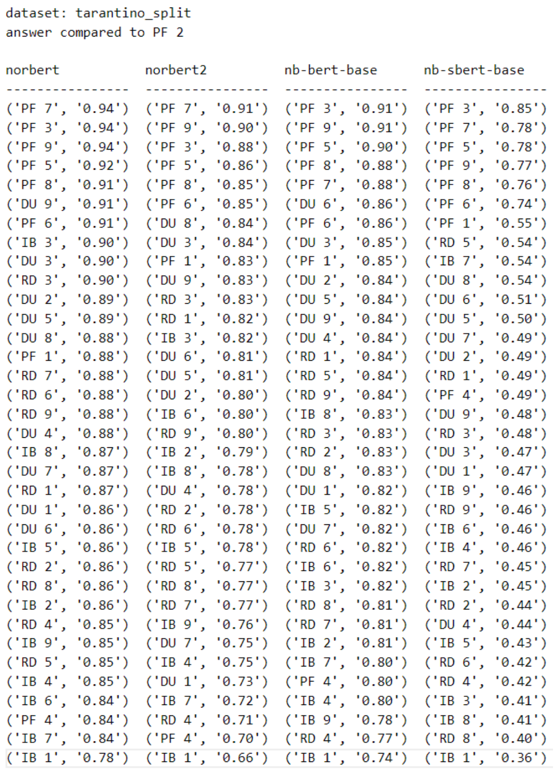
```
dataset: tarantino_split
answer compared to PF 1

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('PF 6', '0.91')  ('PF 2', '0.83')  ('PF 3', '0.89')  ('PF 6', '0.67')
('PF 4', '0.91')  ('PF 8', '0.83')  ('RD 9', '0.88')  ('PF 9', '0.65')
('PF 3', '0.89')  ('PF 6', '0.82')  ('RD 1', '0.88')  ('RD 9', '0.64')
('PF 7', '0.89')  ('PF 7', '0.82')  ('PF 9', '0.88')  ('RD 5', '0.64')
('RD 3', '0.89')  ('PF 4', '0.82')  ('PF 8', '0.88')  ('PF 4', '0.62')
('PF 9', '0.89')  ('PF 9', '0.82')  ('PF 6', '0.88')  ('RD 2', '0.61')
('PF 2', '0.88')  ('PF 3', '0.80')  ('PF 4', '0.87')  ('RD 1', '0.61')
('PF 8', '0.88')  ('PF 5', '0.80')  ('PF 7', '0.87')  ('PF 8', '0.60')
('PF 5', '0.86')  ('RD 3', '0.80')  ('RD 3', '0.87')  ('RD 6', '0.60')
('DU 3', '0.86')  ('RD 1', '0.79')  ('RD 6', '0.86')  ('PF 5', '0.59')
('RD 4', '0.86')  ('IB 9', '0.79')  ('RD 8', '0.86')  ('RD 8', '0.59')
('DU 9', '0.86')  ('IB 7', '0.79')  ('RD 2', '0.86')  ('RD 3', '0.59')
('RD 7', '0.86')  ('IB 6', '0.78')  ('PF 5', '0.85')  ('PF 3', '0.56')
('RD 1', '0.85')  ('RD 2', '0.77')  ('RD 7', '0.85')  ('PF 7', '0.55')
('RD 8', '0.85')  ('IB 4', '0.77')  ('DU 2', '0.85')  ('RD 7', '0.55')
('DU 4', '0.85')  ('IB 2', '0.77')  ('PF 2', '0.85')  ('PF 2', '0.55')
('IB 4', '0.85')  ('IB 3', '0.77')  ('IB 4', '0.85')  ('IB 4', '0.53')
('RD 6', '0.85')  ('IB 8', '0.76')  ('DU 9', '0.84')  ('RD 4', '0.52')
('RD 9', '0.85')  ('RD 9', '0.76')  ('DU 6', '0.83')  ('IB 7', '0.52')
('IB 3', '0.84')  ('DU 6', '0.75')  ('RD 5', '0.83')  ('IB 6', '0.52')
('DU 6', '0.84')  ('DU 4', '0.74')  ('IB 8', '0.83')  ('IB 2', '0.50')
('RD 2', '0.84')  ('DU 3', '0.74')  ('DU 5', '0.83')  ('IB 1', '0.49')
('DU 8', '0.84')  ('DU 2', '0.74')  ('DU 4', '0.83')  ('DU 7', '0.49')
('DU 7', '0.83')  ('RD 6', '0.74')  ('DU 3', '0.83')  ('DU 2', '0.48')
('DU 2', '0.83')  ('DU 8', '0.74')  ('IB 6', '0.83')  ('IB 8', '0.48')
('DU 5', '0.83')  ('RD 8', '0.73')  ('IB 3', '0.82')  ('DU 8', '0.48')
('IB 7', '0.82')  ('DU 9', '0.73')  ('IB 5', '0.82')  ('IB 3', '0.46')
('IB 9', '0.82')  ('RD 7', '0.73')  ('DU 7', '0.82')  ('IB 9', '0.45')
('RD 5', '0.82')  ('DU 5', '0.72')  ('RD 4', '0.82')  ('DU 9', '0.45')
('DU 1', '0.82')  ('RD 5', '0.72')  ('DU 1', '0.81')  ('IB 5', '0.45')
('IB 8', '0.82')  ('DU 7', '0.72')  ('IB 7', '0.80')  ('DU 5', '0.45')
('IB 5', '0.82')  ('IB 5', '0.71')  ('IB 9', '0.79')  ('DU 3', '0.44')
('IB 6', '0.82')  ('DU 1', '0.71')  ('IB 2', '0.79')  ('DU 1', '0.44')
('IB 2', '0.81')  ('IB 1', '0.71')  ('DU 8', '0.77')  ('DU 6', '0.43')
('IB 1', '0.80')  ('RD 4', '0.70')  ('IB 1', '0.76')  ('DU 4', '0.37')
```

Figure .83: The results from comparison to entry labeled "PF 1".



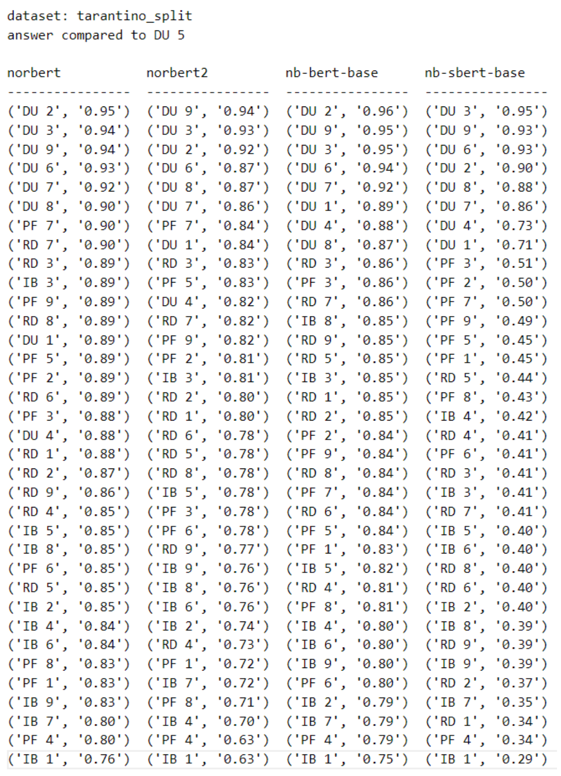
```
dataset: tarantino_split
answer compared to DU 4

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 9', '0.91')   ('DU 9', '0.84')   ('DU 6', '0.89')   ('DU 5', '0.73')
('DU 3', '0.90')   ('DU 5', '0.82')   ('DU 3', '0.88')   ('DU 8', '0.71')
('PF 3', '0.88')   ('DU 3', '0.82')   ('DU 2', '0.88')   ('DU 6', '0.70')
('DU 5', '0.88')   ('DU 6', '0.81')   ('DU 9', '0.88')   ('DU 1', '0.70')
('PF 2', '0.88')   ('DU 2', '0.80')   ('DU 5', '0.88')   ('DU 3', '0.70')
('DU 8', '0.87')   ('DU 7', '0.79')   ('DU 1', '0.87')   ('DU 2', '0.68')
('IB 3', '0.87')   ('IB 3', '0.79')   ('DU 7', '0.85')   ('DU 9', '0.67')
('DU 2', '0.87')   ('DU 1', '0.79')   ('PF 2', '0.84')   ('DU 7', '0.63')
('PF 7', '0.87')   ('PF 2', '0.78')   ('PF 3', '0.84')   ('IB 3', '0.48')
('DU 7', '0.87')   ('DU 8', '0.78')   ('PF 1', '0.83')   ('PF 7', '0.46')
('PF 8', '0.86')   ('RD 3', '0.77')   ('DU 8', '0.83')   ('PF 3', '0.46')
('DU 6', '0.86')   ('PF 7', '0.77')   ('RD 3', '0.82')   ('IB 2', '0.45')
('DU 1', '0.86')   ('IB 5', '0.76')   ('PF 9', '0.82')   ('PF 2', '0.44')
('PF 1', '0.85')   ('PF 3', '0.75')   ('IB 3', '0.82')   ('PF 9', '0.44')
('IB 5', '0.85')   ('IB 8', '0.75')   ('IB 8', '0.81')   ('IB 6', '0.44')
('RD 1', '0.85')   ('PF 5', '0.75')   ('RD 1', '0.81')   ('IB 9', '0.43')
('PF 5', '0.84')   ('IB 6', '0.75')   ('PF 7', '0.81')   ('RD 5', '0.43')
('PF 9', '0.84')   ('RD 1', '0.74')   ('PF 8', '0.81')   ('IB 8', '0.42')
('RD 3', '0.84')   ('PF 1', '0.74')   ('PF 5', '0.81')   ('IB 4', '0.41')
('PF 6', '0.84')   ('PF 9', '0.74')   ('RD 9', '0.80')   ('IB 5', '0.41')
('IB 8', '0.83')   ('RD 7', '0.73')   ('RD 5', '0.80')   ('PF 5', '0.40')
('IB 4', '0.83')   ('IB 4', '0.73')   ('RD 6', '0.79')   ('RD 3', '0.40')
('RD 7', '0.83')   ('IB 9', '0.73')   ('RD 7', '0.79')   ('IB 7', '0.39')
('RD 4', '0.83')   ('RD 5', '0.72')   ('IB 4', '0.78')   ('PF 8', '0.38')
('RD 9', '0.82')   ('IB 2', '0.72')   ('RD 2', '0.78')   ('RD 6', '0.37')
('PF 4', '0.82')   ('PF 6', '0.72')   ('IB 6', '0.78')   ('PF 1', '0.37')
('RD 5', '0.82')   ('PF 8', '0.71')   ('RD 8', '0.78')   ('RD 7', '0.35')
('IB 2', '0.82')   ('RD 2', '0.71')   ('IB 5', '0.77')   ('RD 4', '0.35')
('IB 9', '0.81')   ('RD 9', '0.70')   ('PF 6', '0.76')   ('RD 8', '0.34')
('IB 6', '0.81')   ('IB 7', '0.69')   ('IB 9', '0.76')   ('RD 2', '0.33')
('RD 6', '0.80')   ('RD 8', '0.69')   ('RD 4', '0.76')   ('IB 1', '0.33')
('RD 8', '0.80')   ('RD 4', '0.69')   ('IB 2', '0.76')   ('RD 9', '0.32')
('IB 1', '0.79')   ('RD 6', '0.67')   ('PF 4', '0.75')   ('RD 1', '0.31')
('RD 2', '0.78')   ('PF 4', '0.62')   ('IB 7', '0.75')   ('PF 6', '0.30')
('IB 7', '0.78')   ('IB 1', '0.61')   ('IB 1', '0.72')   ('PF 4', '0.25')
```

Figure .84: The results from comparison to entry labeled "DU 4".



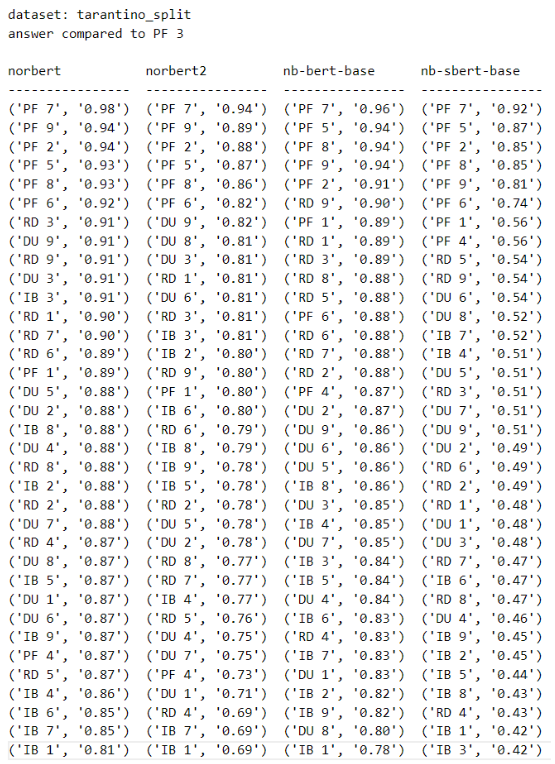
```
dataset: tarantino_split
answer compared to PF 2

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('PF 7', '0.94')   ('PF 7', '0.91')   ('PF 3', '0.91')   ('PF 3', '0.85')
('PF 3', '0.94')   ('PF 9', '0.90')   ('PF 9', '0.91')   ('PF 7', '0.78')
('PF 9', '0.94')   ('PF 3', '0.88')   ('PF 5', '0.90')   ('PF 5', '0.78')
('PF 5', '0.92')   ('PF 5', '0.86')   ('PF 8', '0.88')   ('PF 9', '0.77')
('PF 8', '0.91')   ('PF 8', '0.85')   ('PF 7', '0.88')   ('PF 8', '0.76')
('DU 9', '0.91')   ('PF 6', '0.85')   ('DU 6', '0.86')   ('PF 6', '0.74')
('PF 6', '0.91')   ('DU 8', '0.84')   ('PF 6', '0.86')   ('PF 1', '0.55')
('IB 3', '0.90')   ('DU 3', '0.84')   ('DU 3', '0.85')   ('RD 5', '0.54')
('DU 3', '0.90')   ('PF 1', '0.83')   ('PF 1', '0.85')   ('IB 7', '0.54')
('RD 3', '0.90')   ('DU 9', '0.83')   ('DU 2', '0.84')   ('DU 8', '0.54')
('DU 2', '0.89')   ('RD 3', '0.83')   ('DU 5', '0.84')   ('DU 6', '0.51')
('DU 5', '0.89')   ('RD 1', '0.82')   ('DU 9', '0.84')   ('DU 5', '0.50')
('DU 8', '0.88')   ('IB 3', '0.82')   ('DU 4', '0.84')   ('DU 7', '0.49')
('PF 1', '0.88')   ('DU 6', '0.81')   ('RD 1', '0.84')   ('DU 2', '0.49')
('RD 7', '0.88')   ('DU 5', '0.81')   ('RD 5', '0.84')   ('RD 1', '0.49')
('RD 6', '0.88')   ('DU 2', '0.80')   ('RD 9', '0.84')   ('PF 4', '0.49')
('RD 9', '0.88')   ('IB 6', '0.80')   ('IB 8', '0.83')   ('DU 9', '0.48')
('DU 4', '0.88')   ('RD 9', '0.80')   ('RD 3', '0.83')   ('RD 3', '0.48')
('IB 8', '0.87')   ('IB 2', '0.79')   ('RD 2', '0.83')   ('DU 3', '0.47')
('DU 7', '0.87')   ('IB 8', '0.78')   ('DU 8', '0.83')   ('DU 1', '0.47')
('RD 1', '0.87')   ('DU 4', '0.78')   ('DU 1', '0.82')   ('IB 9', '0.46')
('DU 1', '0.86')   ('RD 2', '0.78')   ('IB 5', '0.82')   ('RD 9', '0.46')
('DU 6', '0.86')   ('RD 6', '0.78')   ('DU 7', '0.82')   ('IB 6', '0.46')
('IB 5', '0.86')   ('IB 5', '0.78')   ('RD 6', '0.82')   ('IB 4', '0.46')
('RD 2', '0.86')   ('RD 5', '0.77')   ('IB 6', '0.82')   ('RD 7', '0.45')
('RD 8', '0.86')   ('RD 8', '0.77')   ('IB 3', '0.82')   ('IB 2', '0.45')
('IB 2', '0.86')   ('RD 7', '0.77')   ('RD 8', '0.81')   ('RD 2', '0.44')
('RD 4', '0.85')   ('IB 9', '0.76')   ('RD 7', '0.81')   ('DU 4', '0.44')
('IB 9', '0.85')   ('DU 7', '0.75')   ('IB 2', '0.81')   ('IB 5', '0.43')
('RD 5', '0.85')   ('IB 4', '0.75')   ('IB 7', '0.80')   ('RD 6', '0.42')
('IB 4', '0.85')   ('DU 1', '0.73')   ('PF 4', '0.80')   ('RD 4', '0.42')
('IB 6', '0.84')   ('IB 7', '0.72')   ('IB 4', '0.80')   ('IB 3', '0.41')
('PF 4', '0.84')   ('RD 4', '0.71')   ('IB 9', '0.78')   ('IB 8', '0.41')
('IB 7', '0.84')   ('PF 4', '0.70')   ('RD 4', '0.77')   ('RD 8', '0.40')
('IB 1', '0.78')   ('IB 1', '0.66')   ('IB 1', '0.74')   ('IB 1', '0.36')
```

Figure .85: The results from comparison to entry labeled "PF 2".



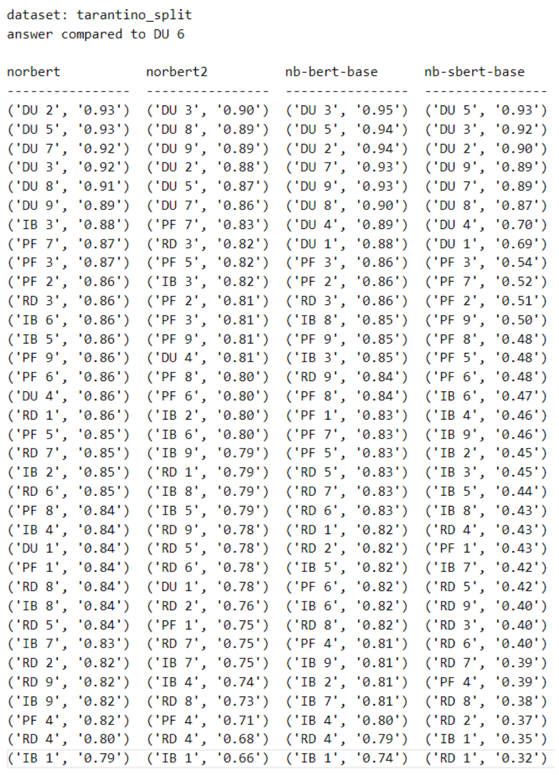
```
dataset: tarantino_split
answer compared to DU 5

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 2', '0.95')   ('DU 9', '0.94')   ('DU 2', '0.96')   ('DU 3', '0.95')
('DU 3', '0.94')   ('DU 3', '0.93')   ('DU 9', '0.95')   ('DU 9', '0.93')
('DU 9', '0.94')   ('DU 2', '0.92')   ('DU 3', '0.95')   ('DU 6', '0.93')
('DU 6', '0.93')   ('DU 6', '0.87')   ('DU 6', '0.94')   ('DU 2', '0.90')
('DU 7', '0.92')   ('DU 8', '0.87')   ('DU 7', '0.92')   ('DU 8', '0.88')
('DU 8', '0.90')   ('DU 7', '0.86')   ('DU 1', '0.89')   ('DU 7', '0.86')
('PF 7', '0.90')   ('PF 7', '0.84')   ('DU 4', '0.88')   ('DU 4', '0.73')
('RD 7', '0.90')   ('DU 1', '0.84')   ('DU 8', '0.87')   ('DU 1', '0.71')
('RD 3', '0.89')   ('RD 3', '0.83')   ('RD 3', '0.86')   ('PF 3', '0.51')
('IB 3', '0.89')   ('PF 5', '0.83')   ('PF 3', '0.86')   ('PF 2', '0.50')
('PF 9', '0.89')   ('DU 4', '0.82')   ('RD 7', '0.86')   ('PF 7', '0.50')
('RD 8', '0.89')   ('RD 7', '0.82')   ('IB 8', '0.85')   ('PF 9', '0.49')
('DU 1', '0.89')   ('PF 9', '0.82')   ('RD 9', '0.85')   ('PF 5', '0.45')
('PF 5', '0.89')   ('PF 2', '0.81')   ('RD 5', '0.85')   ('PF 1', '0.45')
('PF 2', '0.89')   ('IB 3', '0.81')   ('IB 3', '0.85')   ('RD 5', '0.44')
('RD 6', '0.89')   ('RD 2', '0.80')   ('RD 1', '0.85')   ('PF 8', '0.43')
('PF 3', '0.88')   ('RD 1', '0.80')   ('RD 2', '0.85')   ('IB 4', '0.42')
('DU 4', '0.88')   ('RD 6', '0.78')   ('PF 2', '0.84')   ('RD 4', '0.41')
('RD 1', '0.88')   ('RD 5', '0.78')   ('PF 9', '0.84')   ('PF 6', '0.41')
('RD 2', '0.87')   ('RD 8', '0.78')   ('RD 8', '0.84')   ('RD 3', '0.41')
('RD 9', '0.86')   ('IB 5', '0.78')   ('PF 7', '0.84')   ('IB 3', '0.41')
('RD 4', '0.85')   ('PF 3', '0.78')   ('RD 6', '0.84')   ('RD 7', '0.41')
('IB 5', '0.85')   ('PF 6', '0.78')   ('PF 5', '0.84')   ('IB 5', '0.40')
('IB 8', '0.85')   ('RD 9', '0.77')   ('PF 1', '0.83')   ('IB 6', '0.40')
('PF 6', '0.85')   ('IB 9', '0.76')   ('IB 5', '0.82')   ('RD 8', '0.40')
('RD 5', '0.85')   ('IB 8', '0.76')   ('RD 4', '0.81')   ('RD 6', '0.40')
('IB 2', '0.85')   ('IB 6', '0.76')   ('PF 8', '0.81')   ('IB 2', '0.40')
('IB 4', '0.84')   ('IB 2', '0.74')   ('IB 4', '0.80')   ('IB 8', '0.39')
('IB 6', '0.84')   ('RD 4', '0.73')   ('IB 6', '0.80')   ('RD 9', '0.39')
('PF 8', '0.83')   ('PF 1', '0.72')   ('IB 9', '0.80')   ('IB 9', '0.39')
('PF 1', '0.83')   ('IB 7', '0.72')   ('PF 6', '0.80')   ('RD 2', '0.37')
('IB 9', '0.83')   ('PF 8', '0.71')   ('IB 2', '0.79')   ('IB 7', '0.35')
('IB 7', '0.80')   ('IB 4', '0.70')   ('IB 7', '0.79')   ('RD 1', '0.34')
('PF 4', '0.80')   ('PF 4', '0.63')   ('PF 4', '0.79')   ('PF 4', '0.34')
('IB 1', '0.76')   ('IB 1', '0.63')   ('IB 1', '0.75')   ('IB 1', '0.29')
```

Figure .86: The results from comparison to entry labeled "DU 5".



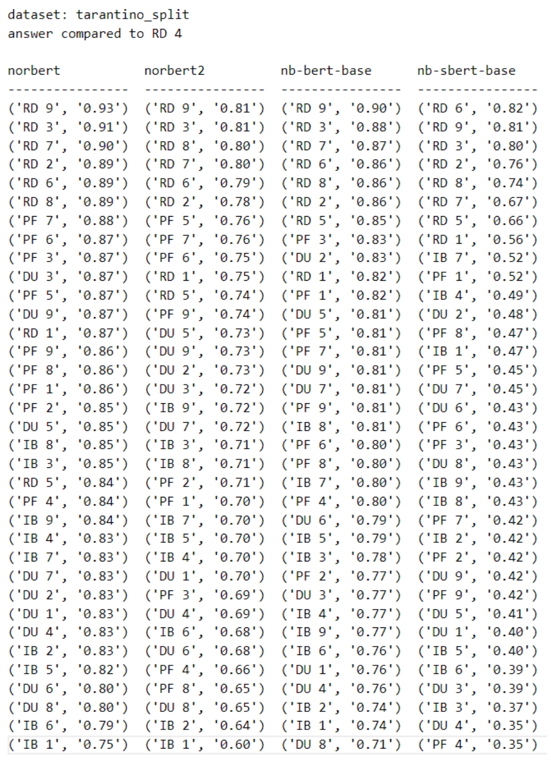
```
dataset: tarantino_split
answer compared to PF 3

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('PF 7', '0.98')   ('PF 7', '0.94')   ('PF 7', '0.96')   ('PF 7', '0.92')
('PF 9', '0.94')   ('PF 9', '0.89')   ('PF 5', '0.94')   ('PF 5', '0.87')
('PF 2', '0.94')   ('PF 2', '0.88')   ('PF 8', '0.94')   ('PF 2', '0.85')
('PF 5', '0.93')   ('PF 5', '0.87')   ('PF 9', '0.94')   ('PF 8', '0.85')
('PF 8', '0.93')   ('PF 8', '0.86')   ('PF 2', '0.91')   ('PF 9', '0.81')
('PF 6', '0.92')   ('PF 6', '0.82')   ('RD 9', '0.90')   ('PF 6', '0.74')
('RD 3', '0.91')   ('DU 9', '0.82')   ('PF 1', '0.89')   ('PF 1', '0.56')
('DU 9', '0.91')   ('DU 8', '0.81')   ('RD 1', '0.89')   ('PF 4', '0.56')
('RD 9', '0.91')   ('DU 3', '0.81')   ('RD 3', '0.89')   ('RD 5', '0.54')
('DU 3', '0.91')   ('RD 1', '0.81')   ('RD 8', '0.88')   ('RD 9', '0.54')
('IB 3', '0.91')   ('DU 6', '0.81')   ('RD 5', '0.88')   ('DU 6', '0.54')
('RD 1', '0.90')   ('RD 3', '0.81')   ('PF 6', '0.88')   ('DU 8', '0.52')
('RD 7', '0.90')   ('IB 3', '0.81')   ('RD 6', '0.88')   ('IB 7', '0.52')
('RD 6', '0.89')   ('IB 2', '0.80')   ('RD 7', '0.88')   ('IB 4', '0.51')
('PF 1', '0.89')   ('RD 9', '0.80')   ('RD 2', '0.88')   ('DU 5', '0.51')
('DU 5', '0.88')   ('PF 1', '0.80')   ('PF 4', '0.87')   ('RD 3', '0.51')
('DU 2', '0.88')   ('IB 6', '0.80')   ('DU 2', '0.87')   ('DU 7', '0.51')
('IB 8', '0.88')   ('RD 6', '0.79')   ('DU 9', '0.86')   ('DU 9', '0.51')
('DU 4', '0.88')   ('IB 8', '0.79')   ('DU 6', '0.86')   ('DU 2', '0.49')
('RD 8', '0.88')   ('IB 9', '0.78')   ('DU 5', '0.86')   ('RD 6', '0.49')
('IB 2', '0.88')   ('IB 5', '0.78')   ('IB 8', '0.86')   ('RD 2', '0.49')
('RD 2', '0.88')   ('RD 2', '0.78')   ('DU 3', '0.85')   ('RD 1', '0.48')
('DU 7', '0.88')   ('DU 5', '0.78')   ('IB 4', '0.85')   ('DU 1', '0.48')
('RD 4', '0.87')   ('DU 2', '0.78')   ('DU 7', '0.85')   ('DU 3', '0.48')
('DU 8', '0.87')   ('RD 8', '0.77')   ('IB 3', '0.84')   ('RD 7', '0.47')
('IB 5', '0.87')   ('RD 7', '0.77')   ('IB 5', '0.84')   ('IB 6', '0.47')
('DU 1', '0.87')   ('IB 4', '0.77')   ('DU 4', '0.84')   ('RD 8', '0.47')
('DU 6', '0.87')   ('RD 5', '0.76')   ('IB 6', '0.83')   ('DU 4', '0.46')
('IB 9', '0.87')   ('DU 4', '0.75')   ('RD 4', '0.83')   ('IB 9', '0.45')
('PF 4', '0.87')   ('DU 7', '0.75')   ('IB 7', '0.83')   ('IB 2', '0.45')
('RD 5', '0.87')   ('PF 4', '0.73')   ('DU 1', '0.83')   ('IB 5', '0.44')
('IB 4', '0.86')   ('DU 1', '0.71')   ('IB 2', '0.82')   ('IB 8', '0.43')
('IB 6', '0.85')   ('RD 4', '0.69')   ('IB 9', '0.82')   ('RD 4', '0.43')
('IB 7', '0.85')   ('IB 7', '0.69')   ('DU 8', '0.80')   ('IB 1', '0.42')
('IB 1', '0.81')   ('IB 1', '0.69')   ('IB 1', '0.78')   ('IB 3', '0.42')
```

Figure .87: The results from comparison to entry labeled "PF 3".



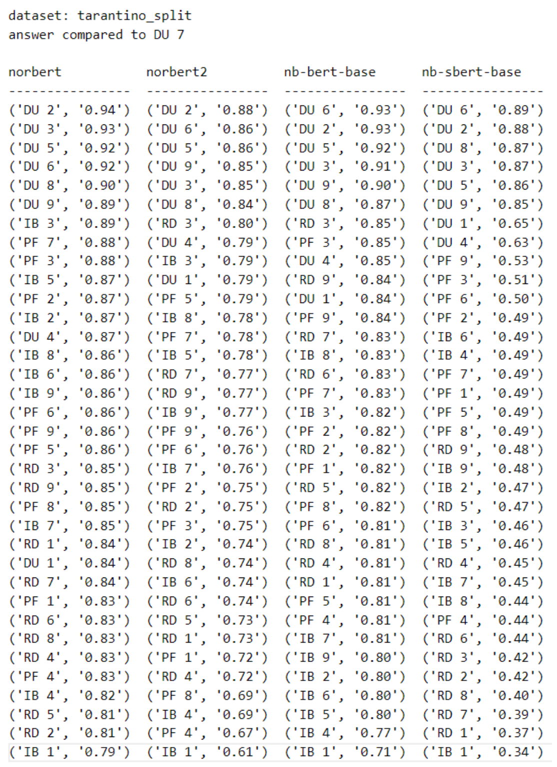
```
dataset: tarantino_split
answer compared to DU 6

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 2', '0.93')   ('DU 3', '0.90')   ('DU 3', '0.95')   ('DU 5', '0.93')
('DU 5', '0.93')   ('DU 8', '0.89')   ('DU 5', '0.94')   ('DU 3', '0.92')
('DU 7', '0.92')   ('DU 9', '0.89')   ('DU 2', '0.94')   ('DU 2', '0.90')
('DU 3', '0.92')   ('DU 2', '0.88')   ('DU 7', '0.93')   ('DU 9', '0.89')
('DU 8', '0.91')   ('DU 5', '0.87')   ('DU 9', '0.93')   ('DU 7', '0.89')
('DU 9', '0.89')   ('DU 7', '0.86')   ('DU 8', '0.90')   ('DU 8', '0.87')
('IB 3', '0.88')   ('PF 7', '0.83')   ('DU 4', '0.89')   ('DU 4', '0.70')
('PF 7', '0.87')   ('RD 3', '0.82')   ('DU 1', '0.88')   ('DU 1', '0.69')
('PF 3', '0.87')   ('PF 5', '0.82')   ('PF 3', '0.86')   ('PF 3', '0.54')
('PF 2', '0.86')   ('IB 3', '0.82')   ('PF 2', '0.86')   ('PF 7', '0.52')
('RD 3', '0.86')   ('PF 2', '0.81')   ('RD 3', '0.86')   ('PF 2', '0.51')
('IB 6', '0.86')   ('PF 3', '0.81')   ('IB 8', '0.85')   ('PF 9', '0.50')
('IB 5', '0.86')   ('PF 9', '0.81')   ('PF 9', '0.85')   ('PF 8', '0.48')
('PF 9', '0.86')   ('DU 4', '0.81')   ('IB 3', '0.85')   ('PF 5', '0.48')
('PF 6', '0.86')   ('PF 8', '0.80')   ('RD 9', '0.84')   ('PF 6', '0.48')
('DU 4', '0.86')   ('PF 6', '0.80')   ('PF 8', '0.84')   ('IB 6', '0.47')
('RD 1', '0.86')   ('IB 2', '0.80')   ('PF 1', '0.83')   ('IB 4', '0.46')
('PF 5', '0.85')   ('IB 6', '0.80')   ('PF 7', '0.83')   ('IB 9', '0.46')
('RD 7', '0.85')   ('IB 9', '0.79')   ('PF 5', '0.83')   ('IB 2', '0.45')
('IB 2', '0.85')   ('RD 1', '0.79')   ('RD 5', '0.83')   ('IB 3', '0.45')
('RD 6', '0.85')   ('IB 8', '0.79')   ('RD 7', '0.83')   ('IB 5', '0.44')
('PF 8', '0.84')   ('IB 5', '0.79')   ('RD 6', '0.83')   ('IB 8', '0.43')
('IB 4', '0.84')   ('RD 9', '0.78')   ('RD 1', '0.82')   ('RD 4', '0.43')
('DU 1', '0.84')   ('RD 5', '0.78')   ('RD 2', '0.82')   ('PF 1', '0.43')
('PF 1', '0.84')   ('RD 6', '0.78')   ('IB 5', '0.82')   ('IB 7', '0.42')
('RD 8', '0.84')   ('DU 1', '0.78')   ('PF 6', '0.82')   ('RD 5', '0.42')
('IB 8', '0.84')   ('RD 2', '0.76')   ('IB 6', '0.82')   ('RD 9', '0.40')
('RD 5', '0.84')   ('PF 1', '0.75')   ('RD 8', '0.82')   ('RD 3', '0.40')
('IB 7', '0.83')   ('RD 7', '0.75')   ('PF 4', '0.81')   ('RD 6', '0.40')
('RD 2', '0.82')   ('IB 7', '0.75')   ('IB 9', '0.81')   ('RD 7', '0.39')
('RD 9', '0.82')   ('IB 4', '0.74')   ('IB 2', '0.81')   ('PF 4', '0.39')
('IB 9', '0.82')   ('RD 8', '0.73')   ('IB 7', '0.81')   ('RD 8', '0.38')
('PF 4', '0.82')   ('PF 4', '0.71')   ('IB 4', '0.80')   ('RD 2', '0.37')
('RD 4', '0.80')   ('RD 4', '0.68')   ('RD 4', '0.79')   ('IB 1', '0.35')
('IB 1', '0.79')   ('IB 1', '0.66')   ('IB 1', '0.74')   ('RD 1', '0.32')
```

Figure .88: The results from comparison to entry labeled "DU 6".



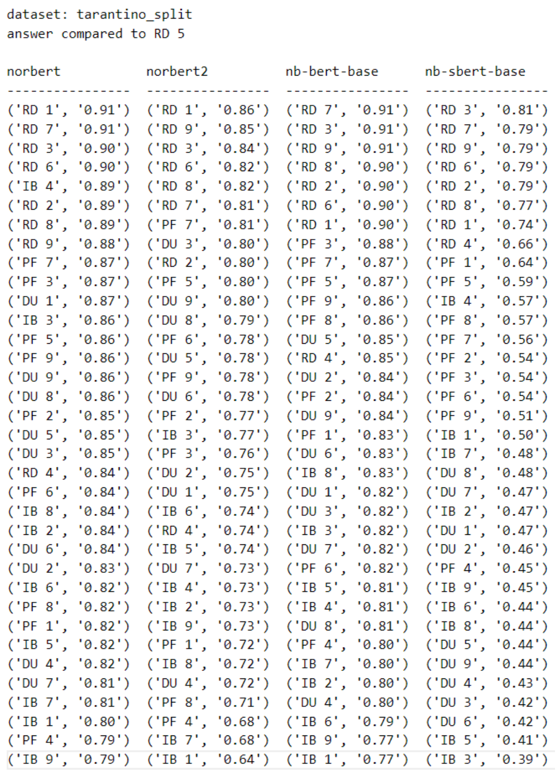
```
dataset: tarantino_split
answer compared to RD 4

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('RD 9', '0.93')   ('RD 9', '0.81')   ('RD 9', '0.90')   ('RD 6', '0.82')
('RD 3', '0.91')   ('RD 3', '0.81')   ('RD 3', '0.88')   ('RD 9', '0.81')
('RD 7', '0.90')   ('RD 8', '0.80')   ('RD 7', '0.87')   ('RD 3', '0.80')
('RD 2', '0.89')   ('RD 7', '0.80')   ('RD 6', '0.86')   ('RD 2', '0.76')
('RD 6', '0.89')   ('RD 6', '0.79')   ('RD 8', '0.86')   ('RD 8', '0.74')
('RD 8', '0.89')   ('RD 2', '0.78')   ('RD 2', '0.86')   ('RD 7', '0.67')
('PF 7', '0.88')   ('PF 5', '0.76')   ('RD 5', '0.85')   ('RD 5', '0.66')
('PF 6', '0.87')   ('PF 7', '0.76')   ('PF 3', '0.83')   ('RD 1', '0.56')
('PF 3', '0.87')   ('PF 6', '0.75')   ('DU 2', '0.83')   ('IB 7', '0.52')
('DU 3', '0.87')   ('RD 1', '0.75')   ('RD 1', '0.82')   ('PF 1', '0.52')
('PF 5', '0.87')   ('RD 5', '0.74')   ('PF 1', '0.82')   ('IB 4', '0.49')
('DU 9', '0.87')   ('PF 9', '0.74')   ('DU 5', '0.81')   ('DU 2', '0.48')
('RD 1', '0.87')   ('DU 5', '0.73')   ('PF 5', '0.81')   ('PF 8', '0.47')
('PF 9', '0.86')   ('DU 9', '0.73')   ('PF 7', '0.81')   ('IB 1', '0.47')
('PF 8', '0.86')   ('DU 2', '0.73')   ('DU 9', '0.81')   ('PF 5', '0.45')
('PF 1', '0.86')   ('DU 3', '0.72')   ('DU 7', '0.81')   ('DU 7', '0.45')
('PF 2', '0.85')   ('IB 9', '0.72')   ('PF 9', '0.81')   ('DU 6', '0.43')
('DU 5', '0.85')   ('DU 7', '0.72')   ('IB 8', '0.81')   ('PF 6', '0.43')
('IB 8', '0.85')   ('IB 3', '0.71')   ('PF 6', '0.80')   ('PF 3', '0.43')
('IB 3', '0.85')   ('IB 8', '0.71')   ('PF 8', '0.80')   ('DU 8', '0.43')
('RD 5', '0.84')   ('PF 2', '0.71')   ('IB 7', '0.80')   ('IB 9', '0.43')
('PF 4', '0.84')   ('PF 1', '0.70')   ('PF 4', '0.80')   ('IB 8', '0.43')
('IB 9', '0.84')   ('IB 7', '0.70')   ('DU 6', '0.79')   ('PF 7', '0.42')
('IB 4', '0.83')   ('IB 5', '0.70')   ('IB 5', '0.79')   ('IB 2', '0.42')
('IB 7', '0.83')   ('IB 4', '0.70')   ('IB 3', '0.78')   ('PF 2', '0.42')
('DU 7', '0.83')   ('DU 1', '0.70')   ('PF 2', '0.77')   ('DU 9', '0.42')
('DU 2', '0.83')   ('PF 3', '0.69')   ('DU 3', '0.77')   ('PF 9', '0.42')
('DU 1', '0.83')   ('DU 4', '0.69')   ('IB 4', '0.77')   ('DU 5', '0.41')
('DU 4', '0.83')   ('IB 6', '0.68')   ('IB 9', '0.77')   ('DU 1', '0.40')
('IB 2', '0.83')   ('DU 6', '0.68')   ('IB 6', '0.76')   ('IB 5', '0.40')
('IB 5', '0.82')   ('PF 4', '0.66')   ('DU 1', '0.76')   ('IB 6', '0.39')
('DU 6', '0.80')   ('PF 8', '0.65')   ('DU 4', '0.76')   ('DU 3', '0.39')
('DU 8', '0.80')   ('DU 8', '0.65')   ('IB 2', '0.74')   ('IB 3', '0.37')
('IB 6', '0.79')   ('IB 2', '0.64')   ('IB 1', '0.74')   ('DU 4', '0.35')
('IB 1', '0.75')   ('IB 1', '0.60')   ('DU 8', '0.71')   ('PF 4', '0.35')
```

Figure .89: The results from comparison to entry labeled "RD 4".



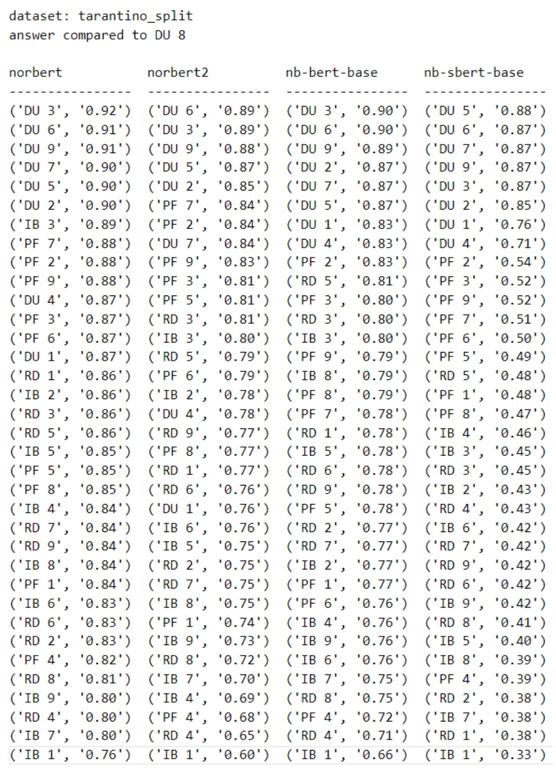
```
dataset: tarantino_split
answer compared to DU 7

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('DU 2', '0.94')    ('DU 2', '0.88')    ('DU 6', '0.93')    ('DU 6', '0.89')
('DU 3', '0.93')    ('DU 6', '0.86')    ('DU 2', '0.93')    ('DU 2', '0.88')
('DU 5', '0.92')    ('DU 5', '0.86')    ('DU 5', '0.92')    ('DU 8', '0.87')
('DU 6', '0.92')    ('DU 9', '0.85')    ('DU 3', '0.91')    ('DU 3', '0.87')
('DU 8', '0.90')    ('DU 3', '0.85')    ('DU 9', '0.90')    ('DU 5', '0.86')
('DU 9', '0.89')    ('DU 8', '0.84')    ('DU 8', '0.87')    ('DU 9', '0.85')
('IB 3', '0.89')    ('RD 3', '0.80')    ('RD 3', '0.85')    ('DU 1', '0.65')
('PF 7', '0.88')    ('DU 4', '0.79')    ('PF 3', '0.85')    ('DU 4', '0.63')
('PF 3', '0.88')    ('IB 3', '0.79')    ('DU 4', '0.85')    ('PF 9', '0.53')
('IB 5', '0.87')    ('DU 1', '0.79')    ('RD 9', '0.84')    ('PF 3', '0.51')
('PF 2', '0.87')    ('PF 5', '0.79')    ('DU 1', '0.84')    ('PF 6', '0.50')
('IB 2', '0.87')    ('IB 8', '0.78')    ('PF 9', '0.84')    ('PF 2', '0.49')
('DU 4', '0.87')    ('PF 7', '0.78')    ('RD 7', '0.83')    ('IB 6', '0.49')
('IB 8', '0.86')    ('IB 5', '0.78')    ('IB 8', '0.83')    ('IB 4', '0.49')
('IB 6', '0.86')    ('RD 7', '0.77')    ('RD 6', '0.83')    ('PF 7', '0.49')
('IB 9', '0.86')    ('RD 9', '0.77')    ('PF 7', '0.83')    ('PF 1', '0.49')
('PF 6', '0.86')    ('IB 9', '0.77')    ('IB 3', '0.82')    ('PF 5', '0.49')
('PF 9', '0.86')    ('PF 9', '0.76')    ('PF 2', '0.82')    ('PF 8', '0.49')
('PF 5', '0.86')    ('PF 6', '0.76')    ('RD 2', '0.82')    ('RD 9', '0.48')
('RD 3', '0.85')    ('IB 7', '0.76')    ('PF 1', '0.82')    ('IB 9', '0.48')
('RD 9', '0.85')    ('PF 2', '0.75')    ('RD 5', '0.82')    ('IB 2', '0.47')
('PF 8', '0.85')    ('RD 2', '0.75')    ('PF 8', '0.82')    ('RD 5', '0.47')
('IB 7', '0.85')    ('PF 3', '0.75')    ('PF 6', '0.81')    ('IB 3', '0.46')
('RD 1', '0.84')    ('IB 2', '0.74')    ('RD 8', '0.81')    ('IB 5', '0.46')
('DU 1', '0.84')    ('RD 8', '0.74')    ('RD 4', '0.81')    ('RD 4', '0.45')
('RD 7', '0.84')    ('IB 6', '0.74')    ('RD 1', '0.81')    ('IB 7', '0.45')
('PF 1', '0.83')    ('RD 6', '0.74')    ('PF 5', '0.81')    ('IB 8', '0.44')
('RD 6', '0.83')    ('RD 5', '0.73')    ('PF 4', '0.81')    ('PF 4', '0.44')
('RD 8', '0.83')    ('RD 1', '0.73')    ('IB 7', '0.81')    ('RD 6', '0.44')
('RD 4', '0.83')    ('PF 1', '0.72')    ('IB 9', '0.80')    ('RD 3', '0.42')
('PF 4', '0.83')    ('RD 4', '0.72')    ('IB 2', '0.80')    ('RD 2', '0.42')
('IB 4', '0.82')    ('PF 8', '0.69')    ('IB 6', '0.80')    ('RD 8', '0.40')
('RD 5', '0.81')    ('IB 4', '0.69')    ('IB 5', '0.80')    ('RD 7', '0.39')
('RD 2', '0.81')    ('PF 4', '0.67')    ('IB 4', '0.77')    ('RD 1', '0.37')
('IB 1', '0.79')    ('IB 1', '0.61')    ('IB 1', '0.71')    ('IB 1', '0.34')
```

Figure .90: The results from comparison to entry labeled "DU 7".



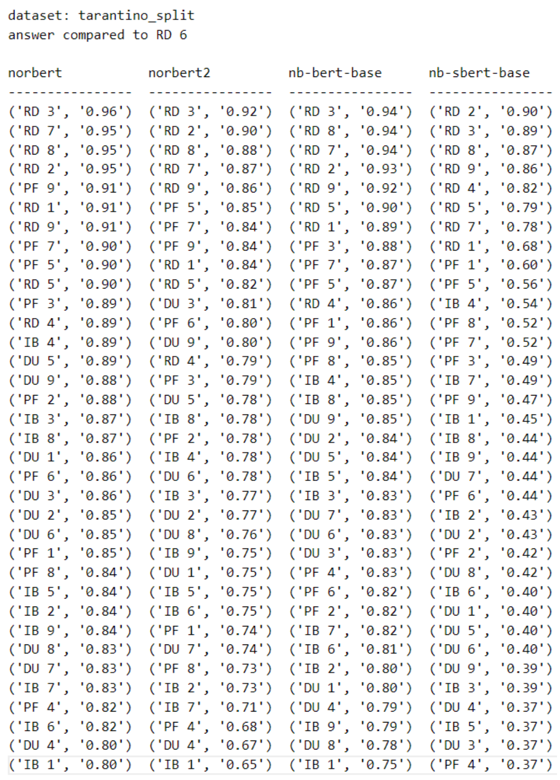
```
dataset: tarantino_split
answer compared to RD 5

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('RD 1', '0.91')    ('RD 1', '0.86')    ('RD 7', '0.91')    ('RD 3', '0.81')
('RD 7', '0.91')    ('RD 9', '0.85')    ('RD 3', '0.91')    ('RD 7', '0.79')
('RD 3', '0.90')    ('RD 3', '0.84')    ('RD 9', '0.91')    ('RD 9', '0.79')
('RD 6', '0.90')    ('RD 6', '0.82')    ('RD 8', '0.90')    ('RD 6', '0.79')
('IB 4', '0.89')    ('RD 8', '0.82')    ('RD 2', '0.90')    ('RD 2', '0.79')
('RD 2', '0.89')    ('RD 7', '0.81')    ('RD 6', '0.90')    ('RD 8', '0.77')
('RD 8', '0.89')    ('PF 7', '0.81')    ('RD 1', '0.90')    ('RD 1', '0.74')
('RD 9', '0.88')    ('DU 3', '0.80')    ('PF 3', '0.88')    ('RD 4', '0.66')
('PF 7', '0.87')    ('RD 2', '0.80')    ('PF 7', '0.87')    ('PF 1', '0.64')
('PF 3', '0.87')    ('PF 5', '0.80')    ('PF 5', '0.87')    ('PF 5', '0.59')
('DU 1', '0.87')    ('DU 9', '0.80')    ('PF 9', '0.86')    ('IB 4', '0.57')
('IB 3', '0.86')    ('DU 8', '0.79')    ('PF 8', '0.86')    ('PF 8', '0.57')
('PF 5', '0.86')    ('PF 6', '0.78')    ('DU 5', '0.85')    ('PF 7', '0.56')
('PF 9', '0.86')    ('DU 5', '0.78')    ('RD 4', '0.85')    ('PF 2', '0.54')
('DU 9', '0.86')    ('PF 9', '0.78')    ('DU 2', '0.84')    ('PF 3', '0.54')
('DU 8', '0.86')    ('DU 6', '0.78')    ('PF 2', '0.84')    ('PF 6', '0.54')
('PF 2', '0.85')    ('PF 2', '0.77')    ('DU 9', '0.84')    ('PF 9', '0.51')
('DU 5', '0.85')    ('IB 3', '0.77')    ('PF 1', '0.83')    ('IB 1', '0.50')
('DU 3', '0.85')    ('PF 3', '0.76')    ('DU 6', '0.83')    ('IB 7', '0.48')
('RD 4', '0.84')    ('DU 2', '0.75')    ('IB 8', '0.83')    ('DU 8', '0.48')
('PF 6', '0.84')    ('DU 1', '0.75')    ('DU 1', '0.82')    ('DU 7', '0.47')
('IB 8', '0.84')    ('IB 6', '0.74')    ('DU 3', '0.82')    ('IB 2', '0.47')
('IB 2', '0.84')    ('RD 4', '0.74')    ('IB 3', '0.82')    ('DU 1', '0.47')
('DU 6', '0.84')    ('IB 5', '0.74')    ('DU 7', '0.82')    ('DU 2', '0.46')
('DU 2', '0.83')    ('DU 7', '0.73')    ('PF 6', '0.82')    ('PF 4', '0.45')
('IB 6', '0.82')    ('IB 4', '0.73')    ('IB 5', '0.81')    ('IB 9', '0.45')
('PF 8', '0.82')    ('IB 2', '0.73')    ('IB 4', '0.81')    ('IB 6', '0.44')
('PF 1', '0.82')    ('IB 9', '0.73')    ('DU 8', '0.81')    ('IB 8', '0.44')
('IB 5', '0.82')    ('PF 1', '0.72')    ('PF 4', '0.80')    ('DU 5', '0.44')
('DU 4', '0.82')    ('IB 8', '0.72')    ('IB 7', '0.80')    ('DU 9', '0.44')
('DU 7', '0.81')    ('DU 4', '0.72')    ('IB 2', '0.80')    ('DU 4', '0.43')
('IB 7', '0.81')    ('PF 8', '0.71')    ('DU 4', '0.80')    ('DU 3', '0.42')
('IB 1', '0.80')    ('PF 4', '0.68')    ('IB 6', '0.79')    ('DU 6', '0.42')
('PF 4', '0.79')    ('IB 7', '0.68')    ('IB 9', '0.77')    ('IB 5', '0.41')
('IB 9', '0.79')    ('IB 1', '0.64')    ('IB 1', '0.77')    ('IB 3', '0.39')
```

Figure .91: The results from comparison to entry labeled "RD 5".



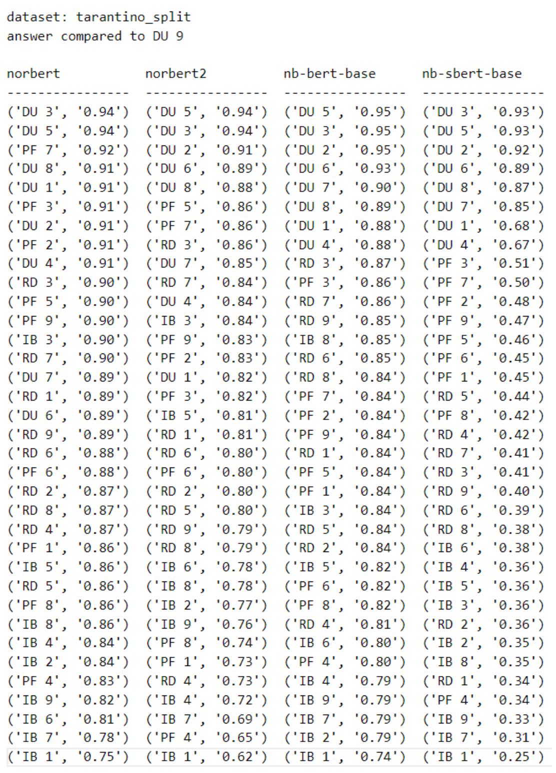
```
dataset: tarantino_split
answer compared to DU 8

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 3', '0.92')   ('DU 6', '0.89')   ('DU 3', '0.90')   ('DU 5', '0.88')
('DU 6', '0.91')   ('DU 3', '0.89')   ('DU 6', '0.90')   ('DU 6', '0.87')
('DU 9', '0.91')   ('DU 9', '0.88')   ('DU 9', '0.89')   ('DU 7', '0.87')
('DU 7', '0.90')   ('DU 5', '0.87')   ('DU 2', '0.87')   ('DU 9', '0.87')
('DU 5', '0.90')   ('DU 2', '0.85')   ('DU 7', '0.87')   ('DU 3', '0.87')
('DU 2', '0.90')   ('PF 7', '0.84')   ('DU 5', '0.87')   ('DU 2', '0.85')
('IB 3', '0.89')   ('PF 2', '0.84')   ('DU 1', '0.83')   ('DU 1', '0.76')
('PF 7', '0.88')   ('DU 7', '0.84')   ('DU 4', '0.83')   ('DU 4', '0.71')
('PF 2', '0.88')   ('PF 9', '0.83')   ('PF 2', '0.83')   ('PF 2', '0.54')
('PF 9', '0.88')   ('PF 3', '0.81')   ('RD 5', '0.81')   ('PF 3', '0.52')
('DU 4', '0.87')   ('PF 5', '0.81')   ('PF 3', '0.80')   ('PF 9', '0.52')
('PF 3', '0.87')   ('RD 3', '0.81')   ('RD 3', '0.80')   ('PF 7', '0.51')
('PF 6', '0.87')   ('IB 3', '0.80')   ('IB 3', '0.80')   ('PF 6', '0.50')
('DU 1', '0.87')   ('RD 5', '0.79')   ('PF 9', '0.79')   ('PF 5', '0.49')
('RD 1', '0.86')   ('PF 6', '0.79')   ('IB 8', '0.79')   ('RD 5', '0.48')
('IB 2', '0.86')   ('IB 2', '0.78')   ('PF 8', '0.79')   ('PF 1', '0.48')
('RD 3', '0.86')   ('DU 4', '0.78')   ('PF 7', '0.78')   ('PF 8', '0.47')
('RD 5', '0.86')   ('RD 9', '0.77')   ('RD 1', '0.78')   ('IB 4', '0.46')
('IB 5', '0.85')   ('PF 8', '0.77')   ('IB 5', '0.78')   ('IB 3', '0.45')
('PF 5', '0.85')   ('RD 1', '0.77')   ('RD 6', '0.78')   ('RD 3', '0.45')
('PF 8', '0.85')   ('RD 6', '0.76')   ('RD 9', '0.78')   ('IB 2', '0.43')
('IB 4', '0.84')   ('DU 1', '0.76')   ('PF 5', '0.78')   ('RD 4', '0.43')
('RD 7', '0.84')   ('IB 6', '0.76')   ('RD 2', '0.77')   ('IB 6', '0.42')
('RD 9', '0.84')   ('IB 5', '0.75')   ('RD 7', '0.77')   ('RD 7', '0.42')
('IB 8', '0.84')   ('RD 2', '0.75')   ('IB 2', '0.77')   ('RD 9', '0.42')
('PF 1', '0.84')   ('RD 7', '0.75')   ('PF 1', '0.77')   ('RD 6', '0.42')
('IB 6', '0.83')   ('IB 8', '0.75')   ('PF 6', '0.76')   ('IB 9', '0.42')
('RD 6', '0.83')   ('PF 1', '0.74')   ('IB 4', '0.76')   ('RD 8', '0.41')
('RD 2', '0.83')   ('IB 9', '0.73')   ('IB 9', '0.76')   ('IB 5', '0.40')
('PF 4', '0.82')   ('RD 8', '0.72')   ('IB 6', '0.76')   ('IB 8', '0.39')
('RD 8', '0.81')   ('IB 7', '0.70')   ('IB 7', '0.75')   ('PF 4', '0.39')
('IB 9', '0.80')   ('IB 4', '0.69')   ('RD 8', '0.75')   ('RD 2', '0.38')
('RD 4', '0.80')   ('PF 4', '0.68')   ('PF 4', '0.72')   ('IB 7', '0.38')
('IB 7', '0.80')   ('RD 4', '0.65')   ('RD 4', '0.71')   ('RD 1', '0.38')
('IB 1', '0.76')   ('IB 1', '0.60')   ('IB 1', '0.66')   ('IB 1', '0.33')
```

Figure .92: The results from comparison to entry labeled "DU 8".



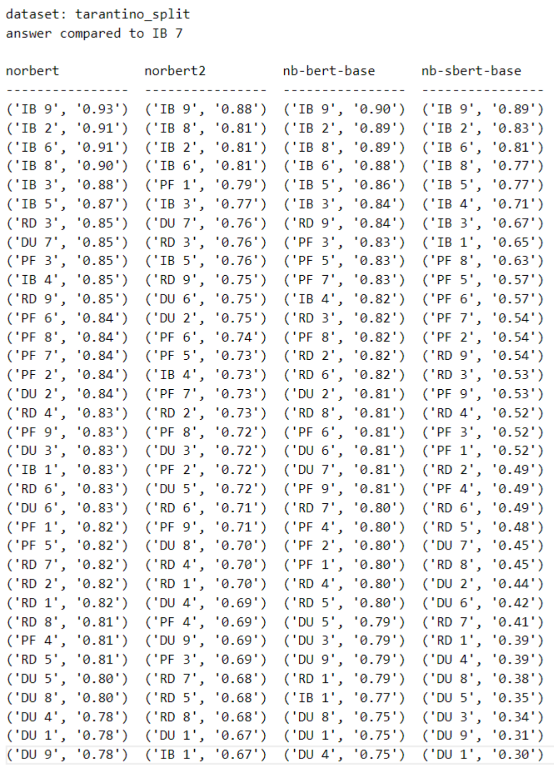
```
dataset: tarantino_split
answer compared to RD 6

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('RD 3', '0.96')   ('RD 3', '0.92')   ('RD 3', '0.94')   ('RD 2', '0.90')
('RD 7', '0.95')   ('RD 2', '0.90')   ('RD 8', '0.94')   ('RD 3', '0.89')
('RD 8', '0.95')   ('RD 8', '0.88')   ('RD 7', '0.94')   ('RD 8', '0.87')
('RD 2', '0.95')   ('RD 7', '0.87')   ('RD 2', '0.93')   ('RD 9', '0.86')
('PF 9', '0.91')   ('RD 9', '0.86')   ('RD 9', '0.92')   ('RD 4', '0.82')
('RD 1', '0.91')   ('PF 5', '0.85')   ('RD 5', '0.90')   ('RD 5', '0.79')
('RD 9', '0.91')   ('PF 7', '0.84')   ('RD 1', '0.89')   ('RD 7', '0.78')
('PF 7', '0.90')   ('PF 9', '0.84')   ('PF 3', '0.88')   ('RD 1', '0.68')
('PF 5', '0.90')   ('RD 1', '0.84')   ('PF 7', '0.87')   ('PF 1', '0.60')
('RD 5', '0.90')   ('RD 5', '0.82')   ('PF 5', '0.87')   ('PF 5', '0.56')
('PF 3', '0.89')   ('DU 3', '0.81')   ('RD 4', '0.86')   ('IB 4', '0.54')
('RD 4', '0.89')   ('PF 6', '0.80')   ('PF 1', '0.86')   ('PF 8', '0.52')
('IB 4', '0.89')   ('DU 9', '0.80')   ('PF 9', '0.86')   ('PF 7', '0.52')
('DU 5', '0.89')   ('RD 4', '0.79')   ('PF 8', '0.85')   ('PF 3', '0.49')
('DU 9', '0.88')   ('PF 3', '0.79')   ('IB 4', '0.85')   ('IB 7', '0.49')
('PF 2', '0.88')   ('DU 5', '0.78')   ('IB 8', '0.85')   ('PF 9', '0.47')
('IB 3', '0.87')   ('IB 8', '0.78')   ('DU 9', '0.85')   ('IB 1', '0.45')
('IB 8', '0.87')   ('PF 2', '0.78')   ('DU 2', '0.84')   ('IB 8', '0.44')
('DU 1', '0.86')   ('IB 4', '0.78')   ('DU 5', '0.84')   ('IB 9', '0.44')
('PF 6', '0.86')   ('DU 6', '0.78')   ('IB 5', '0.84')   ('DU 7', '0.44')
('DU 3', '0.86')   ('IB 3', '0.77')   ('IB 3', '0.83')   ('PF 6', '0.44')
('DU 2', '0.85')   ('DU 2', '0.77')   ('DU 7', '0.83')   ('IB 2', '0.43')
('DU 6', '0.85')   ('DU 8', '0.76')   ('DU 6', '0.83')   ('DU 2', '0.43')
('PF 1', '0.85')   ('IB 9', '0.75')   ('DU 3', '0.83')   ('PF 2', '0.42')
('PF 8', '0.84')   ('DU 1', '0.75')   ('PF 4', '0.83')   ('DU 8', '0.42')
('IB 5', '0.84')   ('IB 5', '0.75')   ('PF 6', '0.82')   ('IB 6', '0.40')
('IB 2', '0.84')   ('IB 6', '0.75')   ('PF 2', '0.82')   ('DU 1', '0.40')
('IB 9', '0.84')   ('PF 1', '0.74')   ('IB 7', '0.82')   ('DU 5', '0.40')
('DU 8', '0.83')   ('DU 7', '0.74')   ('IB 6', '0.81')   ('DU 6', '0.40')
('DU 7', '0.83')   ('PF 8', '0.73')   ('IB 2', '0.80')   ('DU 9', '0.39')
('IB 7', '0.83')   ('IB 2', '0.73')   ('DU 1', '0.80')   ('IB 3', '0.39')
('PF 4', '0.82')   ('IB 7', '0.71')   ('DU 4', '0.79')   ('DU 4', '0.37')
('IB 6', '0.82')   ('PF 4', '0.68')   ('IB 9', '0.79')   ('IB 5', '0.37')
('DU 4', '0.80')   ('DU 4', '0.67')   ('DU 8', '0.78')   ('DU 3', '0.37')
('IB 1', '0.80')   ('IB 1', '0.65')   ('IB 1', '0.75')   ('PF 4', '0.37')
```

Figure .93: The results from comparison to entry labeled "RD 6".



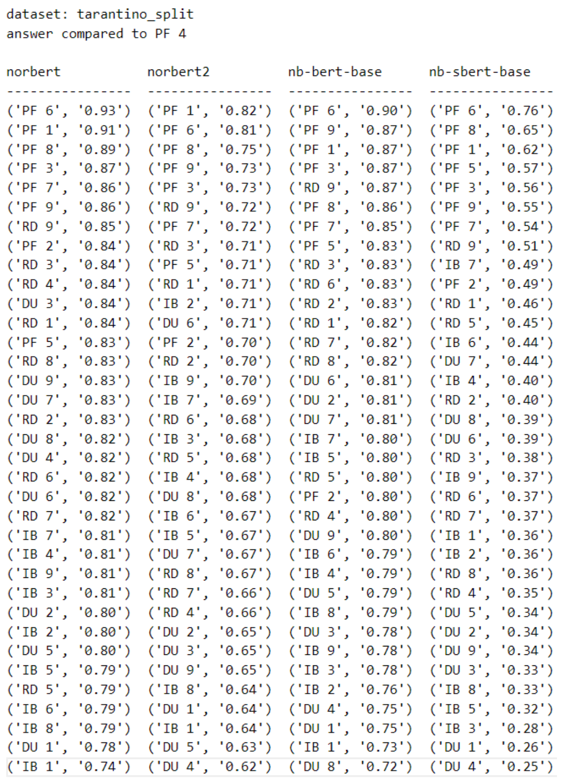
```
dataset: tarantino_split
answer compared to DU 9

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 3', '0.94')   ('DU 5', '0.94')   ('DU 5', '0.95')   ('DU 3', '0.93')
('DU 5', '0.94')   ('DU 3', '0.94')   ('DU 3', '0.95')   ('DU 5', '0.93')
('PF 7', '0.92')   ('DU 2', '0.91')   ('DU 2', '0.95')   ('DU 2', '0.92')
('DU 8', '0.91')   ('DU 6', '0.89')   ('DU 6', '0.93')   ('DU 6', '0.89')
('DU 1', '0.91')   ('DU 8', '0.88')   ('DU 7', '0.90')   ('DU 8', '0.87')
('PF 3', '0.91')   ('PF 5', '0.86')   ('DU 8', '0.89')   ('DU 7', '0.85')
('DU 2', '0.91')   ('PF 7', '0.86')   ('DU 1', '0.88')   ('DU 1', '0.68')
('PF 2', '0.91')   ('RD 3', '0.86')   ('DU 4', '0.88')   ('DU 4', '0.67')
('DU 4', '0.91')   ('DU 7', '0.85')   ('RD 3', '0.87')   ('PF 3', '0.51')
('RD 3', '0.90')   ('RD 7', '0.84')   ('PF 3', '0.86')   ('PF 7', '0.50')
('PF 5', '0.90')   ('DU 4', '0.84')   ('RD 7', '0.86')   ('PF 2', '0.48')
('PF 9', '0.90')   ('IB 3', '0.84')   ('RD 9', '0.85')   ('PF 9', '0.47')
('IB 3', '0.90')   ('PF 9', '0.83')   ('IB 8', '0.85')   ('PF 5', '0.46')
('RD 7', '0.90')   ('PF 2', '0.83')   ('RD 6', '0.85')   ('PF 6', '0.45')
('DU 7', '0.89')   ('DU 1', '0.82')   ('RD 8', '0.84')   ('PF 1', '0.45')
('RD 1', '0.89')   ('PF 3', '0.82')   ('PF 7', '0.84')   ('RD 5', '0.44')
('DU 6', '0.89')   ('IB 5', '0.81')   ('PF 2', '0.84')   ('PF 8', '0.42')
('RD 9', '0.89')   ('RD 1', '0.81')   ('PF 9', '0.84')   ('RD 4', '0.42')
('RD 6', '0.88')   ('RD 6', '0.80')   ('RD 1', '0.84')   ('RD 7', '0.41')
('PF 6', '0.88')   ('PF 6', '0.80')   ('PF 5', '0.84')   ('RD 3', '0.41')
('RD 2', '0.87')   ('RD 2', '0.80')   ('PF 1', '0.84')   ('RD 9', '0.40')
('RD 8', '0.87')   ('RD 5', '0.80')   ('IB 3', '0.84')   ('RD 6', '0.39')
('RD 4', '0.87')   ('RD 9', '0.79')   ('RD 5', '0.84')   ('RD 8', '0.38')
('PF 1', '0.86')   ('RD 8', '0.79')   ('RD 2', '0.84')   ('IB 6', '0.38')
('IB 5', '0.86')   ('IB 6', '0.78')   ('IB 5', '0.82')   ('IB 4', '0.36')
('RD 5', '0.86')   ('IB 8', '0.78')   ('PF 6', '0.82')   ('IB 5', '0.36')
('PF 8', '0.86')   ('IB 2', '0.77')   ('PF 8', '0.82')   ('IB 3', '0.36')
('IB 8', '0.86')   ('IB 9', '0.76')   ('RD 4', '0.81')   ('RD 2', '0.36')
('IB 4', '0.84')   ('PF 8', '0.74')   ('IB 6', '0.80')   ('IB 2', '0.35')
('IB 2', '0.84')   ('PF 1', '0.73')   ('PF 4', '0.80')   ('IB 8', '0.35')
('PF 4', '0.83')   ('RD 4', '0.73')   ('IB 4', '0.79')   ('RD 1', '0.34')
('IB 9', '0.82')   ('IB 4', '0.72')   ('IB 9', '0.79')   ('PF 4', '0.34')
('IB 6', '0.81')   ('IB 7', '0.69')   ('IB 7', '0.79')   ('IB 9', '0.33')
('IB 7', '0.78')   ('PF 4', '0.65')   ('IB 2', '0.79')   ('IB 7', '0.31')
('IB 1', '0.75')   ('IB 1', '0.62')   ('IB 1', '0.74')   ('IB 1', '0.25')
```

Figure .94: The results from comparison to entry labeled "DU 9".



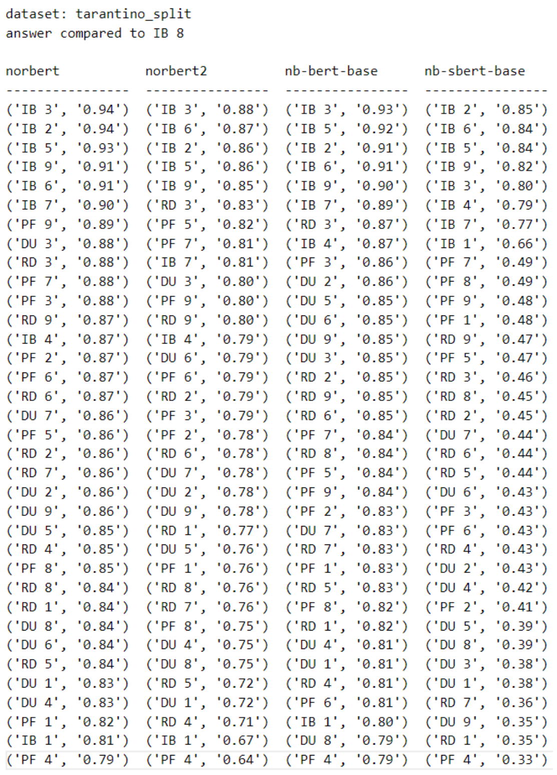
```
dataset: tarantino_split
answer compared to IB 7

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('IB 9', '0.93')   ('IB 9', '0.88')   ('IB 9', '0.90')   ('IB 9', '0.89')
('IB 2', '0.91')   ('IB 8', '0.81')   ('IB 2', '0.89')   ('IB 2', '0.83')
('IB 6', '0.91')   ('IB 2', '0.81')   ('IB 8', '0.89')   ('IB 6', '0.81')
('IB 8', '0.90')   ('IB 6', '0.81')   ('IB 6', '0.88')   ('IB 8', '0.77')
('IB 3', '0.88')   ('PF 1', '0.79')   ('IB 5', '0.86')   ('IB 5', '0.77')
('IB 5', '0.87')   ('IB 3', '0.77')   ('IB 3', '0.84')   ('IB 4', '0.71')
('RD 3', '0.85')   ('DU 7', '0.76')   ('RD 9', '0.84')   ('IB 3', '0.67')
('DU 7', '0.85')   ('RD 3', '0.76')   ('PF 3', '0.83')   ('IB 1', '0.65')
('PF 3', '0.85')   ('IB 5', '0.76')   ('PF 5', '0.83')   ('PF 8', '0.63')
('IB 4', '0.85')   ('RD 9', '0.75')   ('PF 7', '0.83')   ('PF 5', '0.57')
('RD 9', '0.85')   ('DU 6', '0.75')   ('IB 4', '0.82')   ('PF 6', '0.57')
('PF 6', '0.84')   ('DU 2', '0.75')   ('RD 3', '0.82')   ('PF 7', '0.54')
('PF 8', '0.84')   ('PF 6', '0.74')   ('PF 8', '0.82')   ('PF 2', '0.54')
('PF 7', '0.84')   ('PF 5', '0.73')   ('RD 2', '0.82')   ('RD 9', '0.54')
('PF 2', '0.84')   ('IB 4', '0.73')   ('RD 6', '0.82')   ('RD 3', '0.53')
('DU 2', '0.84')   ('PF 7', '0.73')   ('DU 2', '0.81')   ('PF 9', '0.53')
('RD 4', '0.83')   ('RD 2', '0.73')   ('RD 8', '0.81')   ('RD 4', '0.52')
('PF 9', '0.83')   ('PF 8', '0.72')   ('PF 6', '0.81')   ('PF 3', '0.52')
('DU 3', '0.83')   ('DU 3', '0.72')   ('DU 6', '0.81')   ('PF 1', '0.52')
('IB 1', '0.83')   ('PF 2', '0.72')   ('DU 7', '0.81')   ('RD 2', '0.49')
('RD 6', '0.83')   ('DU 5', '0.72')   ('PF 9', '0.81')   ('PF 4', '0.49')
('DU 6', '0.83')   ('RD 6', '0.71')   ('RD 7', '0.80')   ('RD 6', '0.49')
('PF 1', '0.82')   ('PF 9', '0.71')   ('PF 4', '0.80')   ('RD 5', '0.48')
('PF 5', '0.82')   ('DU 8', '0.70')   ('PF 2', '0.80')   ('DU 7', '0.45')
('RD 7', '0.82')   ('RD 4', '0.70')   ('PF 1', '0.80')   ('RD 8', '0.45')
('RD 2', '0.82')   ('RD 1', '0.70')   ('RD 4', '0.80')   ('DU 2', '0.44')
('RD 1', '0.82')   ('DU 4', '0.69')   ('RD 5', '0.80')   ('DU 6', '0.42')
('RD 8', '0.81')   ('PF 4', '0.69')   ('DU 5', '0.79')   ('RD 7', '0.41')
('PF 4', '0.81')   ('DU 9', '0.69')   ('DU 3', '0.79')   ('RD 1', '0.39')
('RD 5', '0.81')   ('PF 3', '0.69')   ('DU 9', '0.79')   ('DU 4', '0.39')
('DU 5', '0.80')   ('RD 7', '0.68')   ('RD 1', '0.79')   ('DU 8', '0.38')
('DU 8', '0.80')   ('RD 5', '0.68')   ('IB 1', '0.77')   ('DU 5', '0.35')
('DU 4', '0.78')   ('RD 8', '0.68')   ('DU 8', '0.75')   ('DU 3', '0.34')
('DU 1', '0.78')   ('DU 1', '0.67')   ('DU 1', '0.75')   ('DU 9', '0.31')
('DU 9', '0.78')   ('IB 1', '0.67')   ('DU 4', '0.75')   ('DU 1', '0.30')
```

Figure .95: The results from comparison to entry labeled "IB 7".



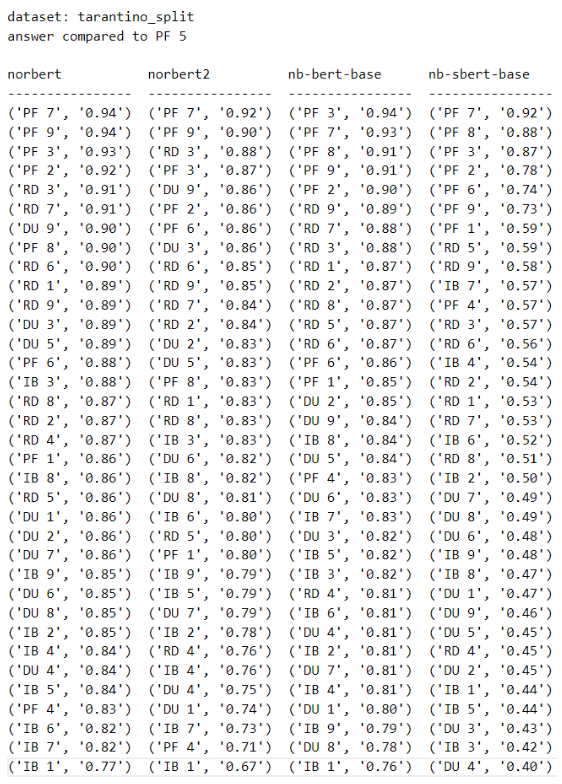
```
dataset: tarantino_split
answer compared to PF 4

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('PF 6', '0.93')   ('PF 1', '0.82')   ('PF 6', '0.90')   ('PF 6', '0.76')
('PF 1', '0.91')   ('PF 6', '0.81')   ('PF 9', '0.87')   ('PF 8', '0.65')
('PF 8', '0.89')   ('PF 8', '0.75')   ('PF 1', '0.87')   ('PF 1', '0.62')
('PF 3', '0.87')   ('PF 9', '0.73')   ('PF 3', '0.87')   ('PF 5', '0.57')
('PF 7', '0.86')   ('PF 3', '0.73')   ('RD 9', '0.87')   ('PF 3', '0.56')
('PF 9', '0.86')   ('RD 9', '0.72')   ('PF 8', '0.86')   ('PF 9', '0.55')
('RD 9', '0.85')   ('PF 7', '0.72')   ('PF 7', '0.85')   ('PF 7', '0.54')
('PF 2', '0.84')   ('RD 3', '0.71')   ('PF 5', '0.83')   ('RD 9', '0.51')
('RD 3', '0.84')   ('PF 5', '0.71')   ('RD 3', '0.83')   ('IB 7', '0.49')
('RD 4', '0.84')   ('RD 1', '0.71')   ('RD 6', '0.83')   ('PF 2', '0.49')
('DU 3', '0.84')   ('IB 2', '0.71')   ('RD 2', '0.83')   ('RD 1', '0.46')
('RD 1', '0.84')   ('DU 6', '0.71')   ('RD 1', '0.82')   ('RD 5', '0.45')
('PF 5', '0.83')   ('PF 2', '0.70')   ('RD 7', '0.82')   ('IB 6', '0.44')
('RD 8', '0.83')   ('RD 2', '0.70')   ('RD 8', '0.82')   ('DU 7', '0.44')
('DU 9', '0.83')   ('IB 9', '0.70')   ('DU 6', '0.81')   ('IB 4', '0.40')
('DU 7', '0.83')   ('IB 7', '0.69')   ('DU 2', '0.81')   ('RD 2', '0.40')
('RD 2', '0.83')   ('RD 6', '0.68')   ('DU 7', '0.81')   ('DU 8', '0.39')
('DU 8', '0.82')   ('IB 3', '0.68')   ('IB 7', '0.80')   ('DU 6', '0.39')
('DU 4', '0.82')   ('RD 5', '0.68')   ('IB 5', '0.80')   ('RD 3', '0.38')
('RD 6', '0.82')   ('IB 4', '0.68')   ('RD 5', '0.80')   ('IB 9', '0.37')
('DU 6', '0.82')   ('DU 8', '0.68')   ('PF 2', '0.80')   ('RD 6', '0.37')
('RD 7', '0.82')   ('IB 6', '0.67')   ('RD 4', '0.80')   ('RD 7', '0.37')
('IB 7', '0.81')   ('IB 5', '0.67')   ('DU 9', '0.80')   ('IB 1', '0.36')
('IB 4', '0.81')   ('DU 7', '0.67')   ('IB 6', '0.79')   ('IB 2', '0.36')
('IB 9', '0.81')   ('RD 8', '0.67')   ('IB 4', '0.79')   ('RD 8', '0.36')
('IB 3', '0.81')   ('RD 7', '0.66')   ('DU 5', '0.79')   ('RD 4', '0.35')
('DU 2', '0.80')   ('RD 4', '0.66')   ('IB 8', '0.79')   ('DU 5', '0.34')
('IB 2', '0.80')   ('DU 2', '0.65')   ('DU 3', '0.78')   ('DU 2', '0.34')
('DU 5', '0.80')   ('DU 3', '0.65')   ('IB 9', '0.78')   ('DU 9', '0.34')
('IB 5', '0.79')   ('DU 9', '0.65')   ('IB 3', '0.78')   ('DU 3', '0.33')
('RD 5', '0.79')   ('IB 8', '0.64')   ('IB 2', '0.76')   ('IB 8', '0.33')
('IB 6', '0.79')   ('DU 1', '0.64')   ('DU 4', '0.75')   ('IB 5', '0.32')
('IB 8', '0.79')   ('IB 1', '0.64')   ('DU 1', '0.75')   ('IB 3', '0.28')
('DU 1', '0.78')   ('DU 5', '0.63')   ('IB 1', '0.73')   ('DU 1', '0.26')
('IB 1', '0.74')   ('DU 4', '0.62')   ('DU 8', '0.72')   ('DU 4', '0.25')
```

Figure .96: The results from comparison to entry labeled "PF 4".



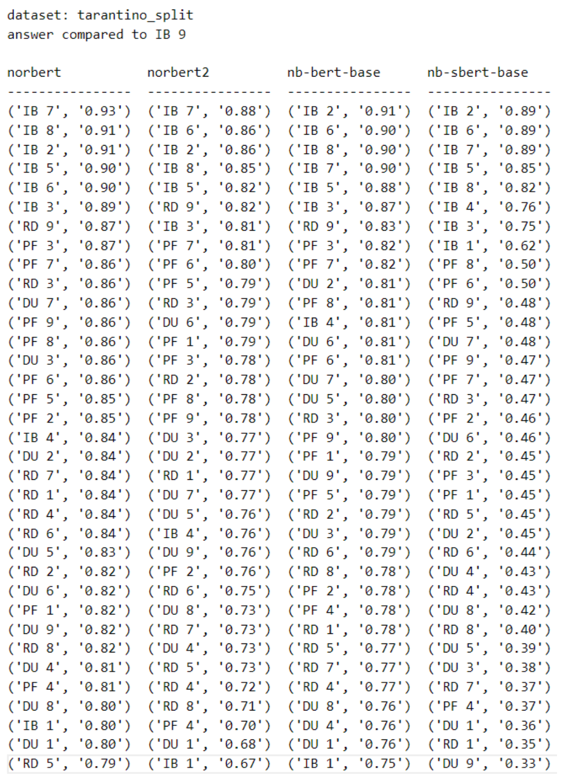
```
dataset: tarantino_split
answer compared to IB 8

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('IB 3', '0.94')   ('IB 3', '0.88')   ('IB 3', '0.93')   ('IB 2', '0.85')
('IB 2', '0.94')   ('IB 6', '0.87')   ('IB 5', '0.92')   ('IB 6', '0.84')
('IB 5', '0.93')   ('IB 2', '0.86')   ('IB 2', '0.91')   ('IB 5', '0.84')
('IB 9', '0.91')   ('IB 5', '0.86')   ('IB 6', '0.91')   ('IB 9', '0.82')
('IB 6', '0.91')   ('IB 9', '0.85')   ('IB 9', '0.90')   ('IB 3', '0.80')
('IB 7', '0.90')   ('RD 3', '0.83')   ('IB 7', '0.89')   ('IB 4', '0.79')
('PF 9', '0.89')   ('PF 5', '0.82')   ('RD 3', '0.87')   ('IB 7', '0.77')
('DU 3', '0.88')   ('PF 7', '0.81')   ('IB 4', '0.87')   ('IB 1', '0.66')
('RD 3', '0.88')   ('IB 7', '0.81')   ('PF 3', '0.86')   ('PF 7', '0.49')
('PF 7', '0.88')   ('DU 3', '0.80')   ('DU 2', '0.86')   ('PF 8', '0.49')
('PF 3', '0.88')   ('PF 9', '0.80')   ('DU 5', '0.85')   ('PF 9', '0.48')
('RD 9', '0.87')   ('RD 9', '0.80')   ('DU 6', '0.85')   ('PF 1', '0.48')
('IB 4', '0.87')   ('IB 4', '0.79')   ('DU 9', '0.85')   ('RD 9', '0.47')
('PF 2', '0.87')   ('DU 6', '0.79')   ('DU 3', '0.85')   ('PF 5', '0.47')
('PF 6', '0.87')   ('PF 6', '0.79')   ('RD 2', '0.85')   ('RD 3', '0.46')
('RD 6', '0.87')   ('RD 2', '0.79')   ('RD 9', '0.85')   ('RD 8', '0.45')
('DU 7', '0.86')   ('PF 3', '0.79')   ('RD 6', '0.85')   ('RD 2', '0.45')
('PF 5', '0.86')   ('PF 2', '0.78')   ('PF 7', '0.84')   ('DU 7', '0.44')
('RD 2', '0.86')   ('RD 6', '0.78')   ('RD 8', '0.84')   ('RD 6', '0.44')
('RD 7', '0.86')   ('DU 7', '0.78')   ('PF 5', '0.84')   ('RD 5', '0.44')
('DU 2', '0.86')   ('DU 2', '0.78')   ('PF 9', '0.84')   ('DU 6', '0.43')
('DU 9', '0.86')   ('DU 9', '0.78')   ('PF 2', '0.83')   ('PF 3', '0.43')
('DU 5', '0.85')   ('RD 1', '0.77')   ('DU 7', '0.83')   ('PF 6', '0.43')
('RD 4', '0.85')   ('DU 5', '0.76')   ('RD 7', '0.83')   ('RD 4', '0.43')
('PF 8', '0.85')   ('PF 1', '0.76')   ('PF 1', '0.83')   ('DU 2', '0.43')
('RD 8', '0.84')   ('RD 8', '0.76')   ('RD 5', '0.83')   ('DU 4', '0.42')
('RD 1', '0.84')   ('RD 7', '0.76')   ('PF 8', '0.82')   ('PF 2', '0.41')
('DU 8', '0.84')   ('PF 8', '0.75')   ('RD 1', '0.82')   ('DU 5', '0.39')
('DU 6', '0.84')   ('DU 4', '0.75')   ('DU 4', '0.81')   ('DU 8', '0.39')
('RD 5', '0.84')   ('DU 8', '0.75')   ('DU 1', '0.81')   ('DU 3', '0.38')
('DU 1', '0.83')   ('RD 5', '0.72')   ('RD 4', '0.81')   ('DU 1', '0.38')
('DU 4', '0.83')   ('DU 1', '0.72')   ('PF 6', '0.81')   ('RD 7', '0.36')
('PF 1', '0.82')   ('RD 4', '0.71')   ('IB 1', '0.80')   ('DU 9', '0.35')
('IB 1', '0.81')   ('IB 1', '0.67')   ('DU 8', '0.79')   ('RD 1', '0.35')
('PF 4', '0.79')   ('PF 4', '0.64')   ('PF 4', '0.79')   ('PF 4', '0.33')
```

Figure .97: The results from comparison to entry labeled "IB 8".



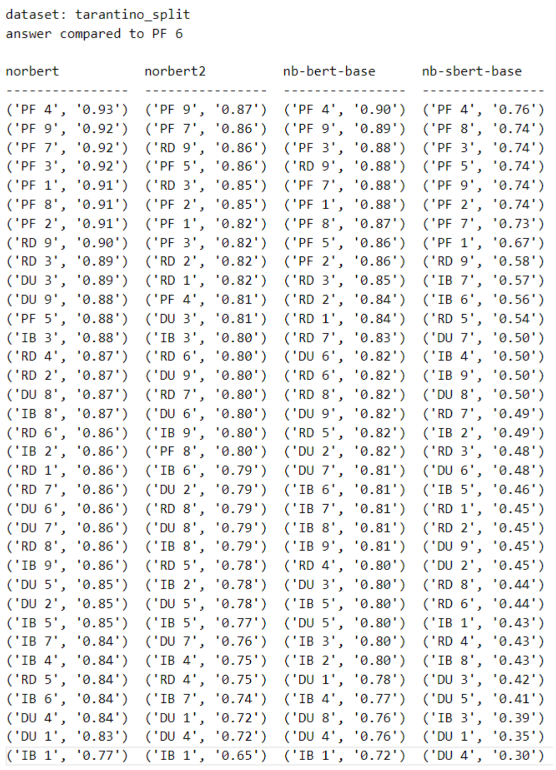
```
dataset: tarantino_split
answer compared to PF 5

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('PF 7', '0.94')   ('PF 7', '0.92')   ('PF 3', '0.94')   ('PF 7', '0.92')
('PF 9', '0.94')   ('PF 9', '0.90')   ('PF 7', '0.93')   ('PF 8', '0.88')
('PF 3', '0.93')   ('RD 3', '0.88')   ('PF 8', '0.91')   ('PF 3', '0.87')
('PF 2', '0.92')   ('PF 3', '0.87')   ('PF 9', '0.91')   ('PF 2', '0.78')
('RD 3', '0.91')   ('DU 9', '0.86')   ('PF 2', '0.90')   ('PF 6', '0.74')
('RD 7', '0.91')   ('PF 2', '0.86')   ('RD 9', '0.89')   ('PF 9', '0.73')
('DU 9', '0.90')   ('PF 6', '0.86')   ('RD 7', '0.88')   ('PF 1', '0.59')
('PF 8', '0.90')   ('DU 3', '0.86')   ('RD 3', '0.88')   ('RD 5', '0.59')
('RD 6', '0.90')   ('RD 6', '0.85')   ('RD 1', '0.87')   ('RD 9', '0.58')
('RD 1', '0.89')   ('RD 9', '0.85')   ('RD 2', '0.87')   ('IB 7', '0.57')
('RD 9', '0.89')   ('RD 7', '0.84')   ('RD 8', '0.87')   ('PF 4', '0.57')
('DU 3', '0.89')   ('RD 2', '0.84')   ('RD 5', '0.87')   ('RD 3', '0.57')
('DU 5', '0.89')   ('DU 2', '0.83')   ('RD 6', '0.87')   ('RD 6', '0.56')
('PF 6', '0.88')   ('DU 5', '0.83')   ('PF 6', '0.86')   ('IB 4', '0.54')
('IB 3', '0.88')   ('PF 8', '0.83')   ('PF 1', '0.85')   ('RD 2', '0.54')
('RD 8', '0.87')   ('RD 1', '0.83')   ('DU 2', '0.85')   ('RD 1', '0.53')
('RD 2', '0.87')   ('RD 8', '0.83')   ('DU 9', '0.84')   ('RD 7', '0.53')
('RD 4', '0.87')   ('IB 3', '0.83')   ('IB 8', '0.84')   ('IB 6', '0.52')
('PF 1', '0.86')   ('DU 6', '0.82')   ('DU 5', '0.84')   ('RD 8', '0.51')
('IB 8', '0.86')   ('IB 8', '0.82')   ('PF 4', '0.83')   ('IB 2', '0.50')
('RD 5', '0.86')   ('DU 8', '0.81')   ('DU 6', '0.83')   ('DU 7', '0.49')
('DU 1', '0.86')   ('IB 6', '0.80')   ('IB 7', '0.83')   ('DU 8', '0.49')
('DU 2', '0.86')   ('RD 5', '0.80')   ('DU 3', '0.82')   ('DU 6', '0.48')
('DU 7', '0.86')   ('PF 1', '0.80')   ('IB 5', '0.82')   ('IB 9', '0.48')
('IB 9', '0.85')   ('IB 9', '0.79')   ('IB 3', '0.82')   ('IB 8', '0.47')
('DU 6', '0.85')   ('IB 5', '0.79')   ('RD 4', '0.81')   ('DU 1', '0.47')
('DU 8', '0.85')   ('DU 7', '0.79')   ('IB 6', '0.81')   ('DU 9', '0.46')
('IB 2', '0.85')   ('IB 2', '0.78')   ('DU 4', '0.81')   ('DU 5', '0.45')
('IB 4', '0.84')   ('RD 4', '0.76')   ('IB 2', '0.81')   ('RD 4', '0.45')
('DU 4', '0.84')   ('IB 4', '0.76')   ('DU 7', '0.81')   ('DU 2', '0.45')
('IB 5', '0.84')   ('DU 4', '0.75')   ('IB 4', '0.81')   ('IB 1', '0.44')
('PF 4', '0.83')   ('DU 1', '0.74')   ('DU 1', '0.80')   ('IB 5', '0.44')
('IB 6', '0.82')   ('IB 7', '0.73')   ('IB 9', '0.79')   ('DU 3', '0.43')
('IB 7', '0.82')   ('PF 4', '0.71')   ('DU 8', '0.78')   ('IB 3', '0.42')
('IB 1', '0.77')   ('IB 1', '0.67')   ('IB 1', '0.76')   ('DU 4', '0.40')
```

Figure .98: The results from comparison to entry labeled "PF 5".



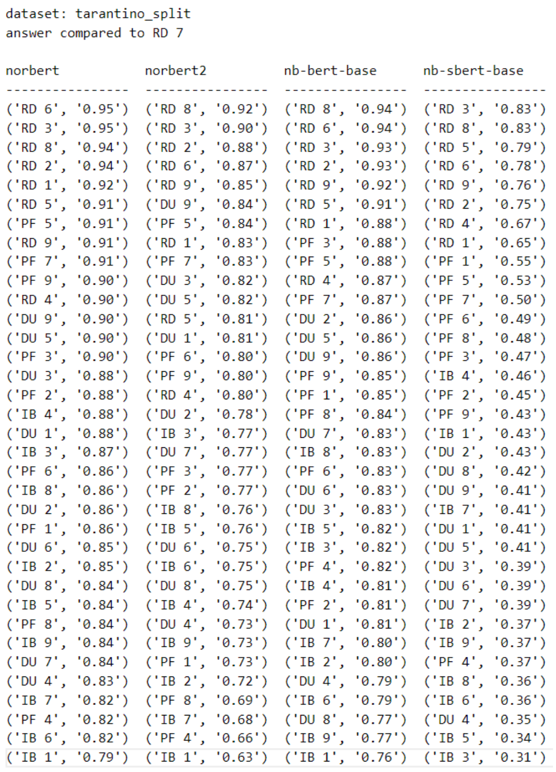
```
dataset: tarantino_split
answer compared to IB 9

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('IB 7', '0.93')   ('IB 7', '0.88')   ('IB 2', '0.91')   ('IB 2', '0.89')
('IB 8', '0.91')   ('IB 6', '0.86')   ('IB 6', '0.90')   ('IB 6', '0.89')
('IB 2', '0.91')   ('IB 2', '0.86')   ('IB 8', '0.90')   ('IB 7', '0.89')
('IB 5', '0.90')   ('IB 8', '0.85')   ('IB 7', '0.90')   ('IB 5', '0.85')
('IB 6', '0.90')   ('IB 5', '0.82')   ('IB 5', '0.88')   ('IB 8', '0.82')
('IB 3', '0.89')   ('RD 9', '0.82')   ('IB 3', '0.87')   ('IB 4', '0.76')
('RD 9', '0.87')   ('IB 3', '0.81')   ('RD 9', '0.83')   ('IB 3', '0.75')
('PF 3', '0.87')   ('PF 7', '0.81')   ('PF 3', '0.82')   ('IB 1', '0.62')
('PF 7', '0.86')   ('PF 6', '0.80')   ('PF 7', '0.82')   ('PF 8', '0.50')
('RD 3', '0.86')   ('PF 5', '0.79')   ('DU 2', '0.81')   ('PF 6', '0.50')
('DU 7', '0.86')   ('RD 3', '0.79')   ('PF 8', '0.81')   ('RD 9', '0.48')
('PF 9', '0.86')   ('DU 6', '0.79')   ('IB 4', '0.81')   ('PF 5', '0.48')
('PF 8', '0.86')   ('PF 1', '0.79')   ('DU 6', '0.81')   ('DU 7', '0.48')
('DU 3', '0.86')   ('PF 3', '0.78')   ('PF 6', '0.81')   ('PF 9', '0.47')
('PF 6', '0.86')   ('RD 2', '0.78')   ('DU 7', '0.80')   ('PF 7', '0.47')
('PF 5', '0.85')   ('PF 8', '0.78')   ('DU 5', '0.80')   ('RD 3', '0.47')
('PF 2', '0.85')   ('PF 9', '0.78')   ('RD 3', '0.80')   ('PF 2', '0.46')
('IB 4', '0.84')   ('DU 3', '0.77')   ('PF 9', '0.80')   ('DU 6', '0.46')
('DU 2', '0.84')   ('DU 2', '0.77')   ('PF 1', '0.79')   ('RD 2', '0.45')
('RD 7', '0.84')   ('RD 1', '0.77')   ('DU 9', '0.79')   ('PF 3', '0.45')
('RD 1', '0.84')   ('DU 7', '0.77')   ('PF 5', '0.79')   ('PF 1', '0.45')
('RD 4', '0.84')   ('DU 5', '0.76')   ('RD 2', '0.79')   ('RD 5', '0.45')
('RD 6', '0.84')   ('IB 4', '0.76')   ('DU 3', '0.79')   ('DU 2', '0.45')
('DU 5', '0.83')   ('DU 9', '0.76')   ('RD 6', '0.79')   ('RD 6', '0.44')
('RD 2', '0.82')   ('PF 2', '0.76')   ('RD 8', '0.78')   ('DU 4', '0.43')
('DU 6', '0.82')   ('RD 6', '0.75')   ('PF 2', '0.78')   ('RD 4', '0.43')
('PF 1', '0.82')   ('DU 8', '0.73')   ('PF 4', '0.78')   ('DU 8', '0.42')
('DU 9', '0.82')   ('RD 7', '0.73')   ('RD 1', '0.78')   ('RD 8', '0.40')
('RD 8', '0.82')   ('DU 4', '0.73')   ('RD 5', '0.77')   ('DU 5', '0.39')
('DU 4', '0.81')   ('RD 5', '0.73')   ('RD 7', '0.77')   ('DU 3', '0.38')
('PF 4', '0.81')   ('RD 4', '0.72')   ('RD 4', '0.77')   ('RD 7', '0.37')
('DU 8', '0.80')   ('RD 8', '0.71')   ('DU 8', '0.76')   ('PF 4', '0.37')
('IB 1', '0.80')   ('PF 4', '0.70')   ('DU 4', '0.76')   ('DU 1', '0.36')
('DU 1', '0.80')   ('DU 1', '0.68')   ('DU 1', '0.76')   ('RD 1', '0.35')
('RD 5', '0.79')   ('IB 1', '0.67')   ('IB 1', '0.75')   ('DU 9', '0.33')
```

Figure .99: The results from comparison to entry labeled "IB 9".



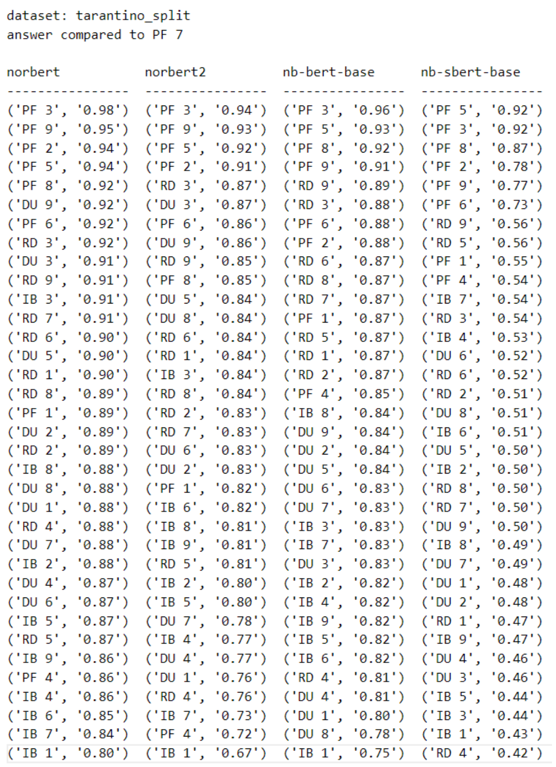
```
dataset: tarantino_split
answer compared to PF 6

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('PF 4', '0.93')    ('PF 9', '0.87')    ('PF 4', '0.90')    ('PF 4', '0.76')
('PF 9', '0.92')    ('PF 7', '0.86')    ('PF 9', '0.89')    ('PF 8', '0.74')
('PF 7', '0.92')    ('RD 9', '0.86')    ('PF 3', '0.88')    ('PF 3', '0.74')
('PF 3', '0.92')    ('PF 5', '0.86')    ('RD 9', '0.88')    ('PF 5', '0.74')
('PF 1', '0.91')    ('RD 3', '0.85')    ('PF 7', '0.88')    ('PF 9', '0.74')
('PF 8', '0.91')    ('PF 2', '0.85')    ('PF 1', '0.88')    ('PF 2', '0.74')
('PF 2', '0.91')    ('PF 1', '0.82')    ('PF 8', '0.87')    ('PF 7', '0.73')
('RD 9', '0.90')    ('PF 3', '0.82')    ('PF 5', '0.86')    ('PF 1', '0.67')
('RD 3', '0.89')    ('RD 2', '0.82')    ('PF 2', '0.86')    ('RD 9', '0.58')
('DU 3', '0.89')    ('RD 1', '0.82')    ('RD 3', '0.85')    ('IB 7', '0.57')
('DU 9', '0.88')    ('PF 4', '0.81')    ('RD 2', '0.84')    ('IB 6', '0.56')
('PF 5', '0.88')    ('DU 3', '0.81')    ('RD 1', '0.84')    ('RD 5', '0.54')
('IB 3', '0.88')    ('IB 3', '0.80')    ('RD 7', '0.83')    ('DU 7', '0.50')
('RD 4', '0.87')    ('RD 6', '0.80')    ('DU 6', '0.82')    ('IB 4', '0.50')
('RD 2', '0.87')    ('DU 9', '0.80')    ('RD 6', '0.82')    ('IB 9', '0.50')
('DU 8', '0.87')    ('RD 7', '0.80')    ('RD 8', '0.82')    ('DU 8', '0.50')
('IB 8', '0.87')    ('DU 6', '0.80')    ('DU 9', '0.82')    ('RD 7', '0.49')
('RD 6', '0.86')    ('IB 9', '0.80')    ('RD 5', '0.82')    ('IB 2', '0.49')
('IB 2', '0.86')    ('PF 8', '0.80')    ('DU 2', '0.82')    ('RD 3', '0.48')
('RD 1', '0.86')    ('IB 6', '0.79')    ('DU 7', '0.81')    ('DU 6', '0.48')
('RD 7', '0.86')    ('DU 2', '0.79')    ('IB 6', '0.81')    ('IB 5', '0.46')
('DU 6', '0.86')    ('RD 8', '0.79')    ('IB 7', '0.81')    ('RD 1', '0.45')
('DU 7', '0.86')    ('DU 8', '0.79')    ('IB 8', '0.81')    ('RD 2', '0.45')
('RD 8', '0.86')    ('IB 8', '0.79')    ('IB 9', '0.81')    ('DU 9', '0.45')
('IB 9', '0.86')    ('RD 5', '0.78')    ('RD 4', '0.80')    ('DU 2', '0.45')
('DU 5', '0.85')    ('IB 2', '0.78')    ('DU 3', '0.80')    ('RD 8', '0.44')
('DU 2', '0.85')    ('DU 5', '0.78')    ('IB 5', '0.80')    ('RD 6', '0.44')
('IB 5', '0.85')    ('IB 5', '0.77')    ('DU 5', '0.80')    ('IB 1', '0.43')
('IB 7', '0.84')    ('DU 7', '0.76')    ('IB 3', '0.80')    ('RD 4', '0.43')
('IB 4', '0.84')    ('IB 4', '0.75')    ('IB 2', '0.80')    ('IB 8', '0.43')
('RD 5', '0.84')    ('RD 4', '0.75')    ('DU 1', '0.78')    ('DU 3', '0.42')
('IB 6', '0.84')    ('IB 7', '0.74')    ('IB 4', '0.77')    ('DU 5', '0.41')
('DU 4', '0.84')    ('DU 1', '0.72')    ('DU 8', '0.76')    ('IB 3', '0.39')
('DU 1', '0.83')    ('DU 4', '0.72')    ('DU 4', '0.76')    ('DU 1', '0.35')
('IB 1', '0.77')    ('IB 1', '0.65')    ('IB 1', '0.72')    ('DU 4', '0.30')
```

Figure .100: The results from comparison to entry labeled "PF 6".



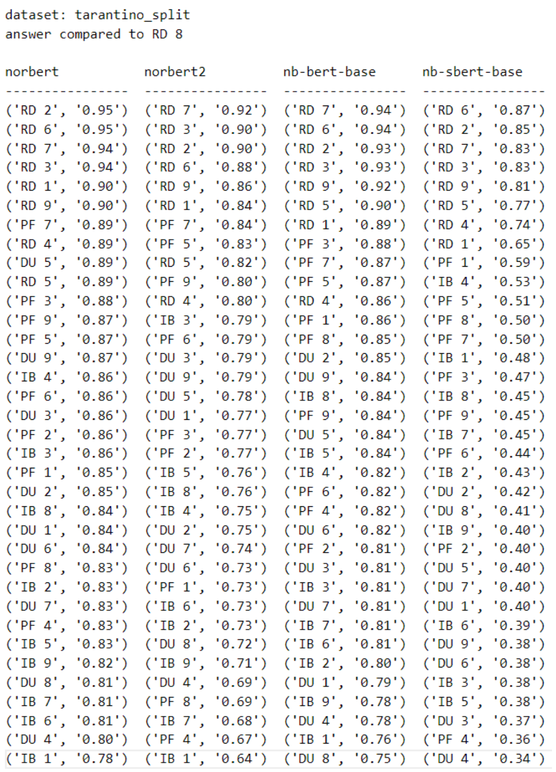
```
dataset: tarantino_split
answer compared to RD 7

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('RD 6', '0.95')    ('RD 8', '0.92')    ('RD 8', '0.94')    ('RD 3', '0.83')
('RD 3', '0.95')    ('RD 3', '0.90')    ('RD 6', '0.94')    ('RD 8', '0.83')
('RD 8', '0.94')    ('RD 2', '0.88')    ('RD 3', '0.93')    ('RD 5', '0.79')
('RD 2', '0.94')    ('RD 6', '0.87')    ('RD 2', '0.93')    ('RD 6', '0.78')
('RD 1', '0.92')    ('RD 9', '0.85')    ('RD 9', '0.92')    ('RD 9', '0.76')
('RD 5', '0.91')    ('DU 9', '0.84')    ('RD 5', '0.91')    ('RD 2', '0.75')
('PF 5', '0.91')    ('PF 5', '0.84')    ('RD 1', '0.88')    ('RD 4', '0.67')
('RD 9', '0.91')    ('RD 1', '0.83')    ('PF 3', '0.88')    ('RD 1', '0.65')
('PF 7', '0.91')    ('PF 7', '0.83')    ('PF 5', '0.88')    ('PF 1', '0.55')
('PF 9', '0.90')    ('DU 3', '0.82')    ('RD 4', '0.87')    ('PF 5', '0.53')
('RD 4', '0.90')    ('DU 5', '0.82')    ('PF 7', '0.87')    ('PF 7', '0.50')
('DU 9', '0.90')    ('RD 5', '0.81')    ('DU 2', '0.86')    ('PF 6', '0.49')
('DU 5', '0.90')    ('DU 1', '0.81')    ('DU 5', '0.86')    ('PF 8', '0.48')
('PF 3', '0.90')    ('PF 6', '0.80')    ('DU 9', '0.86')    ('PF 3', '0.47')
('DU 3', '0.88')    ('PF 9', '0.80')    ('PF 9', '0.85')    ('IB 4', '0.46')
('PF 2', '0.88')    ('RD 4', '0.80')    ('PF 1', '0.85')    ('PF 2', '0.45')
('IB 4', '0.88')    ('DU 2', '0.78')    ('PF 8', '0.84')    ('PF 9', '0.43')
('DU 1', '0.88')    ('IB 3', '0.77')    ('DU 7', '0.83')    ('IB 1', '0.43')
('IB 3', '0.87')    ('DU 7', '0.77')    ('IB 8', '0.83')    ('DU 2', '0.43')
('PF 6', '0.86')    ('PF 3', '0.77')    ('PF 6', '0.83')    ('DU 8', '0.42')
('IB 8', '0.86')    ('PF 2', '0.77')    ('DU 6', '0.83')    ('DU 9', '0.41')
('DU 2', '0.86')    ('IB 8', '0.76')    ('DU 3', '0.83')    ('IB 7', '0.41')
('PF 1', '0.86')    ('IB 5', '0.76')    ('IB 5', '0.82')    ('DU 1', '0.41')
('DU 6', '0.85')    ('DU 6', '0.75')    ('IB 3', '0.82')    ('DU 5', '0.41')
('IB 2', '0.85')    ('IB 6', '0.75')    ('PF 4', '0.82')    ('DU 3', '0.39')
('DU 8', '0.84')    ('DU 8', '0.75')    ('IB 4', '0.81')    ('DU 6', '0.39')
('IB 5', '0.84')    ('IB 4', '0.74')    ('PF 2', '0.81')    ('DU 7', '0.39')
('PF 8', '0.84')    ('DU 4', '0.73')    ('DU 1', '0.81')    ('IB 2', '0.37')
('IB 9', '0.84')    ('IB 9', '0.73')    ('IB 7', '0.80')    ('IB 9', '0.37')
('DU 7', '0.84')    ('PF 1', '0.73')    ('IB 2', '0.80')    ('PF 4', '0.37')
('DU 4', '0.83')    ('IB 2', '0.72')    ('DU 4', '0.79')    ('IB 8', '0.36')
('IB 7', '0.82')    ('PF 8', '0.69')    ('IB 6', '0.79')    ('IB 6', '0.36')
('PF 4', '0.82')    ('IB 7', '0.68')    ('DU 8', '0.77')    ('DU 4', '0.35')
('IB 6', '0.82')    ('PF 4', '0.66')    ('IB 9', '0.77')    ('IB 5', '0.34')
('IB 1', '0.79')    ('IB 1', '0.63')    ('IB 1', '0.76')    ('IB 3', '0.31')
```

Figure .101: The results from comparison to entry labeled "RD 7".



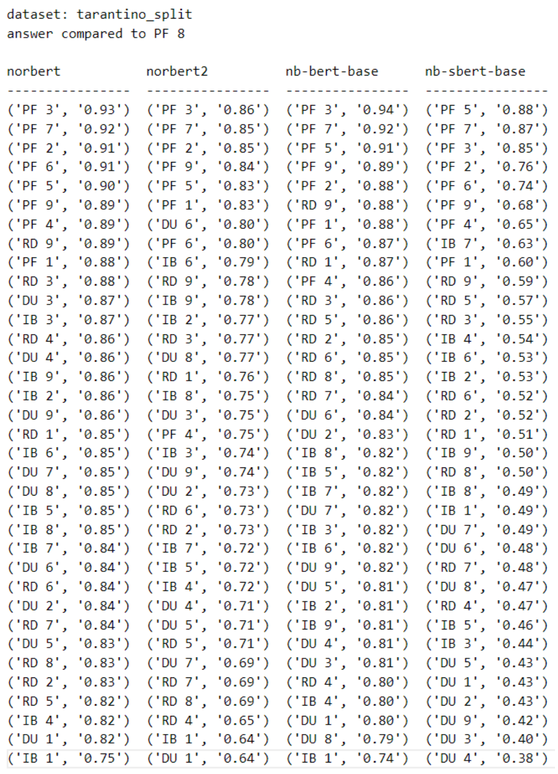
```
dataset: tarantino_split
answer compared to PF 7

norbert           norbert2          nb-bert-base      nb-sbert-base
---------------   ---------------   ---------------   ---------------
('PF 3', '0.98')  ('PF 3', '0.94')  ('PF 3', '0.96')  ('PF 5', '0.92')
('PF 9', '0.95')  ('PF 9', '0.93')  ('PF 5', '0.93')  ('PF 3', '0.92')
('PF 2', '0.94')  ('PF 5', '0.92')  ('PF 8', '0.92')  ('PF 8', '0.87')
('PF 5', '0.94')  ('PF 2', '0.91')  ('PF 9', '0.91')  ('PF 2', '0.78')
('PF 8', '0.92')  ('RD 3', '0.87')  ('RD 9', '0.89')  ('PF 9', '0.77')
('DU 9', '0.92')  ('DU 3', '0.87')  ('RD 3', '0.88')  ('PF 6', '0.73')
('PF 6', '0.92')  ('PF 6', '0.86')  ('PF 6', '0.88')  ('RD 9', '0.56')
('RD 3', '0.92')  ('DU 9', '0.86')  ('PF 2', '0.88')  ('RD 5', '0.56')
('DU 3', '0.91')  ('RD 9', '0.85')  ('RD 6', '0.87')  ('PF 1', '0.55')
('RD 9', '0.91')  ('PF 8', '0.85')  ('RD 8', '0.87')  ('PF 4', '0.54')
('IB 3', '0.91')  ('DU 5', '0.84')  ('RD 7', '0.87')  ('IB 7', '0.54')
('RD 7', '0.91')  ('DU 8', '0.84')  ('PF 1', '0.87')  ('RD 3', '0.54')
('RD 6', '0.90')  ('RD 6', '0.84')  ('RD 5', '0.87')  ('IB 4', '0.53')
('DU 5', '0.90')  ('RD 1', '0.84')  ('RD 1', '0.87')  ('DU 6', '0.52')
('RD 1', '0.90')  ('IB 3', '0.84')  ('RD 2', '0.87')  ('RD 6', '0.52')
('RD 8', '0.89')  ('RD 8', '0.84')  ('PF 4', '0.85')  ('RD 2', '0.51')
('PF 1', '0.89')  ('RD 2', '0.83')  ('IB 8', '0.84')  ('DU 8', '0.51')
('DU 2', '0.89')  ('RD 7', '0.83')  ('DU 9', '0.84')  ('IB 6', '0.51')
('RD 2', '0.89')  ('DU 6', '0.83')  ('DU 2', '0.84')  ('DU 5', '0.50')
('IB 8', '0.88')  ('DU 2', '0.83')  ('DU 5', '0.84')  ('IB 2', '0.50')
('DU 8', '0.88')  ('PF 1', '0.82')  ('DU 6', '0.83')  ('RD 8', '0.50')
('DU 1', '0.88')  ('IB 6', '0.82')  ('DU 7', '0.83')  ('RD 7', '0.50')
('RD 4', '0.88')  ('IB 8', '0.81')  ('IB 3', '0.83')  ('DU 9', '0.50')
('DU 7', '0.88')  ('IB 9', '0.81')  ('IB 7', '0.83')  ('IB 8', '0.49')
('IB 2', '0.88')  ('RD 5', '0.81')  ('DU 3', '0.83')  ('DU 7', '0.49')
('DU 4', '0.87')  ('IB 2', '0.80')  ('IB 2', '0.82')  ('DU 1', '0.48')
('DU 6', '0.87')  ('IB 5', '0.80')  ('IB 4', '0.82')  ('DU 2', '0.48')
('IB 5', '0.87')  ('DU 7', '0.78')  ('IB 9', '0.82')  ('RD 1', '0.47')
('RD 5', '0.87')  ('IB 4', '0.77')  ('IB 5', '0.82')  ('IB 9', '0.47')
('IB 9', '0.86')  ('DU 4', '0.77')  ('IB 6', '0.82')  ('DU 4', '0.46')
('PF 4', '0.86')  ('DU 1', '0.76')  ('RD 4', '0.81')  ('DU 3', '0.46')
('IB 4', '0.86')  ('RD 4', '0.76')  ('DU 4', '0.81')  ('IB 5', '0.44')
('IB 6', '0.85')  ('IB 7', '0.73')  ('DU 1', '0.80')  ('IB 3', '0.44')
('IB 7', '0.84')  ('PF 4', '0.72')  ('DU 8', '0.78')  ('IB 1', '0.43')
('IB 1', '0.80')  ('IB 1', '0.67')  ('IB 1', '0.75')  ('RD 4', '0.42')
```

Figure .102: The results from comparison to entry labeled "PF 7".



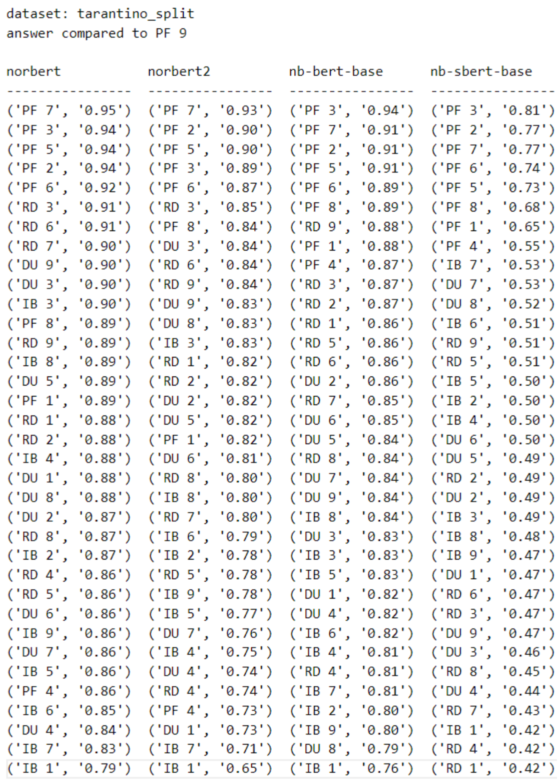
```
dataset: tarantino_split
answer compared to RD 8

norbert           norbert2          nb-bert-base      nb-sbert-base
---------------   ---------------   ---------------   ---------------
('RD 2', '0.95')  ('RD 7', '0.92')  ('RD 7', '0.94')  ('RD 6', '0.87')
('RD 6', '0.95')  ('RD 3', '0.90')  ('RD 6', '0.94')  ('RD 2', '0.85')
('RD 7', '0.94')  ('RD 2', '0.90')  ('RD 2', '0.93')  ('RD 7', '0.83')
('RD 3', '0.94')  ('RD 6', '0.88')  ('RD 3', '0.93')  ('RD 3', '0.83')
('RD 1', '0.90')  ('RD 9', '0.86')  ('RD 9', '0.92')  ('RD 9', '0.81')
('RD 9', '0.90')  ('RD 1', '0.84')  ('RD 5', '0.90')  ('RD 5', '0.77')
('PF 7', '0.89')  ('PF 7', '0.84')  ('RD 1', '0.89')  ('RD 4', '0.74')
('RD 4', '0.89')  ('PF 5', '0.83')  ('PF 3', '0.88')  ('RD 1', '0.65')
('DU 5', '0.89')  ('RD 5', '0.82')  ('PF 7', '0.87')  ('PF 1', '0.59')
('RD 5', '0.89')  ('PF 9', '0.80')  ('PF 5', '0.87')  ('IB 4', '0.53')
('PF 3', '0.88')  ('RD 4', '0.80')  ('RD 4', '0.86')  ('PF 5', '0.51')
('PF 9', '0.87')  ('IB 3', '0.79')  ('PF 1', '0.86')  ('PF 8', '0.50')
('PF 5', '0.87')  ('PF 6', '0.79')  ('PF 8', '0.85')  ('PF 7', '0.50')
('DU 9', '0.87')  ('DU 3', '0.79')  ('DU 2', '0.85')  ('IB 1', '0.48')
('IB 4', '0.86')  ('DU 9', '0.79')  ('DU 9', '0.84')  ('PF 3', '0.47')
('PF 6', '0.86')  ('DU 5', '0.78')  ('IB 8', '0.84')  ('IB 8', '0.45')
('DU 3', '0.86')  ('DU 1', '0.77')  ('PF 9', '0.84')  ('PF 9', '0.45')
('PF 2', '0.86')  ('PF 3', '0.77')  ('DU 5', '0.84')  ('IB 7', '0.45')
('IB 3', '0.86')  ('PF 2', '0.77')  ('IB 5', '0.84')  ('PF 6', '0.44')
('PF 1', '0.85')  ('IB 5', '0.76')  ('IB 4', '0.82')  ('IB 2', '0.43')
('DU 2', '0.85')  ('IB 8', '0.76')  ('PF 6', '0.82')  ('DU 2', '0.42')
('IB 8', '0.84')  ('IB 4', '0.75')  ('PF 4', '0.82')  ('DU 8', '0.41')
('DU 1', '0.84')  ('DU 2', '0.75')  ('DU 6', '0.82')  ('IB 9', '0.40')
('DU 6', '0.84')  ('DU 7', '0.74')  ('PF 2', '0.81')  ('PF 2', '0.40')
('PF 8', '0.83')  ('DU 6', '0.73')  ('DU 3', '0.81')  ('DU 5', '0.40')
('IB 2', '0.83')  ('PF 1', '0.73')  ('IB 3', '0.81')  ('DU 7', '0.40')
('DU 7', '0.83')  ('IB 6', '0.73')  ('DU 7', '0.81')  ('DU 1', '0.40')
('PF 4', '0.83')  ('IB 2', '0.73')  ('IB 7', '0.81')  ('IB 6', '0.39')
('IB 5', '0.83')  ('DU 8', '0.72')  ('IB 6', '0.81')  ('DU 9', '0.38')
('IB 9', '0.82')  ('IB 9', '0.71')  ('IB 2', '0.80')  ('DU 6', '0.38')
('DU 8', '0.81')  ('DU 4', '0.69')  ('DU 1', '0.79')  ('IB 3', '0.38')
('IB 7', '0.81')  ('PF 8', '0.69')  ('IB 9', '0.78')  ('IB 5', '0.38')
('IB 6', '0.81')  ('IB 7', '0.68')  ('DU 4', '0.78')  ('DU 3', '0.37')
('DU 4', '0.80')  ('PF 4', '0.67')  ('IB 1', '0.76')  ('PF 4', '0.36')
('IB 1', '0.78')  ('IB 1', '0.64')  ('DU 8', '0.75')  ('DU 4', '0.34')
```

Figure .103: The results from comparison to entry labeled "RD 8".



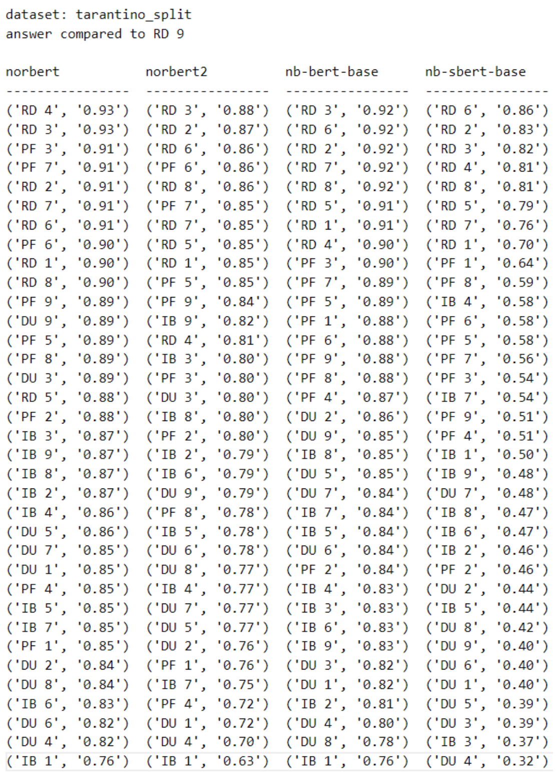
```
dataset: tarantino_split
answer compared to PF 8

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('PF 3', '0.93')    ('PF 3', '0.86')    ('PF 3', '0.94')    ('PF 5', '0.88')
('PF 7', '0.92')    ('PF 7', '0.85')    ('PF 7', '0.92')    ('PF 7', '0.87')
('PF 2', '0.91')    ('PF 2', '0.85')    ('PF 5', '0.91')    ('PF 3', '0.85')
('PF 6', '0.91')    ('PF 9', '0.84')    ('PF 9', '0.89')    ('PF 2', '0.76')
('PF 5', '0.90')    ('PF 5', '0.83')    ('PF 2', '0.88')    ('PF 6', '0.74')
('PF 9', '0.89')    ('PF 1', '0.83')    ('RD 9', '0.88')    ('PF 9', '0.68')
('PF 4', '0.89')    ('DU 6', '0.80')    ('PF 1', '0.88')    ('PF 4', '0.65')
('RD 9', '0.89')    ('PF 6', '0.80')    ('PF 6', '0.87')    ('IB 7', '0.63')
('PF 1', '0.88')    ('IB 6', '0.79')    ('RD 1', '0.87')    ('PF 1', '0.60')
('RD 3', '0.88')    ('RD 9', '0.78')    ('PF 4', '0.86')    ('RD 9', '0.59')
('DU 3', '0.87')    ('IB 9', '0.78')    ('RD 3', '0.86')    ('RD 5', '0.57')
('IB 3', '0.87')    ('IB 2', '0.77')    ('RD 5', '0.86')    ('RD 3', '0.55')
('RD 4', '0.86')    ('RD 3', '0.77')    ('RD 2', '0.85')    ('IB 4', '0.54')
('DU 4', '0.86')    ('DU 8', '0.77')    ('RD 6', '0.85')    ('IB 6', '0.53')
('IB 9', '0.86')    ('RD 1', '0.76')    ('RD 8', '0.85')    ('IB 2', '0.53')
('IB 2', '0.86')    ('IB 8', '0.75')    ('RD 7', '0.84')    ('RD 6', '0.52')
('DU 9', '0.86')    ('DU 3', '0.75')    ('DU 6', '0.84')    ('RD 2', '0.52')
('RD 1', '0.85')    ('PF 4', '0.75')    ('DU 2', '0.83')    ('RD 1', '0.51')
('IB 6', '0.85')    ('IB 3', '0.74')    ('IB 8', '0.82')    ('IB 9', '0.50')
('DU 7', '0.85')    ('DU 9', '0.74')    ('IB 5', '0.82')    ('RD 8', '0.50')
('DU 8', '0.85')    ('DU 2', '0.73')    ('IB 7', '0.82')    ('IB 8', '0.49')
('IB 5', '0.85')    ('RD 6', '0.73')    ('DU 7', '0.82')    ('IB 1', '0.49')
('IB 8', '0.85')    ('RD 2', '0.73')    ('IB 3', '0.82')    ('DU 7', '0.49')
('IB 7', '0.84')    ('IB 7', '0.72')    ('IB 6', '0.82')    ('DU 6', '0.48')
('DU 6', '0.84')    ('IB 5', '0.72')    ('DU 9', '0.82')    ('RD 7', '0.48')
('RD 6', '0.84')    ('IB 4', '0.72')    ('DU 5', '0.81')    ('DU 8', '0.47')
('DU 2', '0.84')    ('DU 4', '0.71')    ('IB 2', '0.81')    ('RD 4', '0.47')
('RD 7', '0.84')    ('DU 5', '0.71')    ('IB 9', '0.81')    ('IB 5', '0.46')
('DU 5', '0.83')    ('RD 5', '0.71')    ('DU 4', '0.81')    ('IB 3', '0.44')
('RD 8', '0.83')    ('DU 7', '0.69')    ('DU 3', '0.81')    ('DU 5', '0.43')
('RD 2', '0.83')    ('RD 7', '0.69')    ('RD 4', '0.80')    ('DU 1', '0.43')
('RD 5', '0.82')    ('RD 8', '0.69')    ('IB 4', '0.80')    ('DU 2', '0.43')
('IB 4', '0.82')    ('RD 4', '0.65')    ('DU 1', '0.80')    ('DU 9', '0.42')
('DU 1', '0.82')    ('IB 1', '0.64')    ('DU 8', '0.79')    ('DU 3', '0.40')
('IB 1', '0.75')    ('DU 1', '0.64')    ('IB 1', '0.74')    ('DU 4', '0.38')
```

Figure .104: The results from comparison to entry labeled "PF 8".



```
dataset: tarantino_split
answer compared to PF 9

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('PF 7', '0.95')    ('PF 7', '0.93')    ('PF 3', '0.94')    ('PF 3', '0.81')
('PF 3', '0.94')    ('PF 2', '0.90')    ('PF 7', '0.91')    ('PF 2', '0.77')
('PF 5', '0.94')    ('PF 5', '0.90')    ('PF 2', '0.91')    ('PF 7', '0.77')
('PF 2', '0.94')    ('PF 3', '0.89')    ('PF 5', '0.91')    ('PF 6', '0.74')
('PF 6', '0.92')    ('PF 6', '0.87')    ('PF 6', '0.89')    ('PF 5', '0.73')
('RD 3', '0.91')    ('RD 3', '0.85')    ('PF 8', '0.89')    ('PF 8', '0.68')
('RD 6', '0.91')    ('PF 8', '0.84')    ('RD 9', '0.88')    ('PF 1', '0.65')
('RD 7', '0.90')    ('DU 3', '0.84')    ('PF 1', '0.88')    ('PF 4', '0.55')
('DU 9', '0.90')    ('RD 6', '0.84')    ('PF 4', '0.87')    ('IB 7', '0.53')
('DU 3', '0.90')    ('RD 9', '0.84')    ('RD 3', '0.87')    ('DU 7', '0.53')
('IB 3', '0.90')    ('DU 9', '0.83')    ('RD 2', '0.87')    ('DU 8', '0.52')
('PF 8', '0.89')    ('DU 8', '0.83')    ('RD 1', '0.86')    ('IB 6', '0.51')
('RD 9', '0.89')    ('IB 3', '0.83')    ('RD 5', '0.86')    ('RD 9', '0.51')
('IB 8', '0.89')    ('RD 1', '0.82')    ('RD 6', '0.86')    ('RD 5', '0.51')
('DU 5', '0.89')    ('RD 2', '0.82')    ('DU 2', '0.86')    ('IB 5', '0.50')
('PF 1', '0.89')    ('DU 2', '0.82')    ('RD 7', '0.85')    ('IB 2', '0.50')
('RD 1', '0.88')    ('DU 5', '0.82')    ('DU 6', '0.85')    ('IB 4', '0.50')
('RD 2', '0.88')    ('PF 1', '0.82')    ('DU 5', '0.84')    ('DU 6', '0.50')
('IB 4', '0.88')    ('DU 6', '0.81')    ('RD 8', '0.84')    ('DU 5', '0.49')
('DU 1', '0.88')    ('RD 8', '0.80')    ('DU 7', '0.84')    ('RD 2', '0.49')
('DU 8', '0.88')    ('IB 8', '0.80')    ('DU 9', '0.84')    ('DU 2', '0.49')
('DU 2', '0.87')    ('RD 7', '0.80')    ('IB 8', '0.84')    ('IB 3', '0.49')
('RD 8', '0.87')    ('IB 6', '0.79')    ('DU 3', '0.83')    ('IB 8', '0.48')
('IB 2', '0.87')    ('IB 2', '0.78')    ('IB 3', '0.83')    ('IB 9', '0.47')
('RD 4', '0.86')    ('RD 5', '0.78')    ('IB 5', '0.83')    ('DU 1', '0.47')
('RD 5', '0.86')    ('IB 9', '0.78')    ('DU 1', '0.82')    ('RD 6', '0.47')
('DU 6', '0.86')    ('IB 5', '0.77')    ('DU 4', '0.82')    ('RD 3', '0.47')
('IB 9', '0.86')    ('DU 7', '0.76')    ('IB 6', '0.82')    ('DU 9', '0.47')
('DU 7', '0.86')    ('IB 4', '0.75')    ('IB 4', '0.81')    ('DU 3', '0.46')
('IB 5', '0.86')    ('DU 4', '0.74')    ('RD 4', '0.81')    ('RD 8', '0.45')
('PF 4', '0.86')    ('RD 4', '0.74')    ('IB 7', '0.81')    ('DU 4', '0.44')
('IB 6', '0.85')    ('PF 4', '0.73')    ('IB 2', '0.80')    ('RD 7', '0.43')
('DU 4', '0.84')    ('DU 1', '0.73')    ('IB 9', '0.80')    ('IB 1', '0.42')
('IB 7', '0.83')    ('IB 7', '0.71')    ('DU 8', '0.79')    ('RD 4', '0.42')
('IB 1', '0.79')    ('IB 1', '0.65')    ('IB 1', '0.76')    ('RD 1', '0.42')
```

Figure .105: The results from comparison to entry labeled "PF 9".

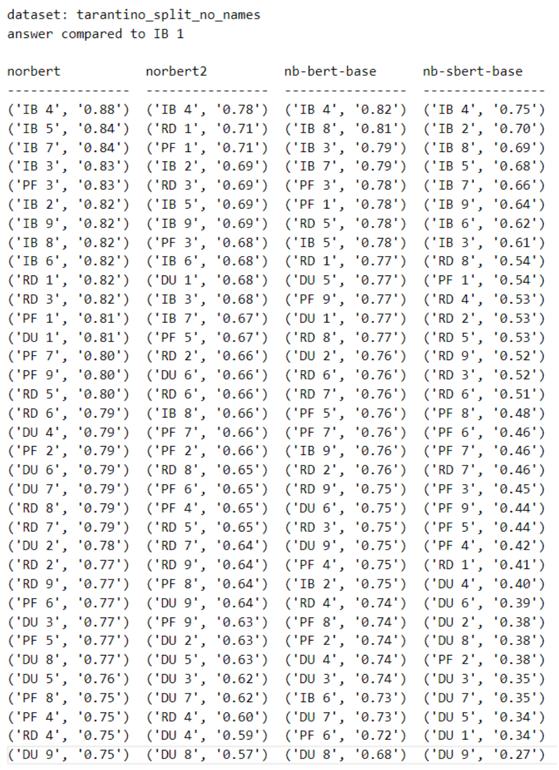


```
dataset: tarantino_split
answer compared to RD 9

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('RD 4', '0.93')   ('RD 3', '0.88')   ('RD 3', '0.92')   ('RD 6', '0.86')
('RD 3', '0.93')   ('RD 2', '0.87')   ('RD 6', '0.92')   ('RD 2', '0.83')
('PF 3', '0.91')   ('RD 6', '0.86')   ('RD 2', '0.92')   ('RD 3', '0.82')
('PF 7', '0.91')   ('PF 6', '0.86')   ('RD 7', '0.92')   ('RD 4', '0.81')
('RD 2', '0.91')   ('RD 8', '0.86')   ('RD 8', '0.92')   ('RD 8', '0.81')
('RD 7', '0.91')   ('PF 7', '0.85')   ('RD 5', '0.91')   ('RD 5', '0.79')
('RD 6', '0.91')   ('RD 7', '0.85')   ('RD 1', '0.91')   ('RD 7', '0.76')
('PF 6', '0.90')   ('RD 5', '0.85')   ('RD 4', '0.90')   ('RD 1', '0.70')
('RD 1', '0.90')   ('RD 1', '0.85')   ('PF 3', '0.90')   ('PF 1', '0.64')
('RD 8', '0.90')   ('PF 5', '0.85')   ('PF 7', '0.89')   ('PF 8', '0.59')
('PF 9', '0.89')   ('PF 9', '0.84')   ('PF 5', '0.89')   ('IB 4', '0.58')
('DU 9', '0.89')   ('IB 9', '0.82')   ('PF 1', '0.88')   ('PF 6', '0.58')
('PF 5', '0.89')   ('RD 4', '0.81')   ('PF 6', '0.88')   ('PF 5', '0.58')
('PF 8', '0.89')   ('IB 3', '0.80')   ('PF 9', '0.88')   ('PF 7', '0.56')
('DU 3', '0.89')   ('PF 3', '0.80')   ('PF 8', '0.88')   ('PF 3', '0.54')
('RD 5', '0.88')   ('DU 3', '0.80')   ('PF 4', '0.87')   ('IB 7', '0.54')
('PF 2', '0.88')   ('IB 8', '0.80')   ('DU 2', '0.86')   ('PF 9', '0.51')
('IB 3', '0.87')   ('PF 2', '0.80')   ('DU 9', '0.85')   ('PF 4', '0.51')
('IB 9', '0.87')   ('IB 2', '0.79')   ('IB 8', '0.85')   ('IB 1', '0.50')
('IB 8', '0.87')   ('IB 6', '0.79')   ('DU 5', '0.85')   ('IB 9', '0.48')
('IB 2', '0.87')   ('DU 9', '0.79')   ('DU 7', '0.84')   ('DU 7', '0.48')
('IB 4', '0.86')   ('PF 8', '0.78')   ('IB 7', '0.84')   ('IB 8', '0.47')
('DU 5', '0.86')   ('IB 5', '0.78')   ('IB 5', '0.84')   ('IB 6', '0.47')
('DU 7', '0.85')   ('DU 6', '0.78')   ('DU 6', '0.84')   ('IB 2', '0.46')
('DU 1', '0.85')   ('DU 8', '0.77')   ('PF 2', '0.84')   ('PF 2', '0.46')
('PF 4', '0.85')   ('IB 4', '0.77')   ('IB 4', '0.83')   ('DU 2', '0.44')
('IB 5', '0.85')   ('DU 7', '0.77')   ('IB 3', '0.83')   ('IB 5', '0.44')
('IB 7', '0.85')   ('DU 5', '0.77')   ('IB 6', '0.83')   ('DU 8', '0.42')
('PF 1', '0.85')   ('DU 2', '0.76')   ('IB 9', '0.83')   ('DU 9', '0.40')
('DU 2', '0.84')   ('PF 1', '0.76')   ('DU 3', '0.82')   ('DU 6', '0.40')
('DU 8', '0.84')   ('IB 7', '0.75')   ('DU 1', '0.82')   ('DU 1', '0.40')
('IB 6', '0.83')   ('PF 4', '0.72')   ('IB 2', '0.81')   ('DU 5', '0.39')
('DU 6', '0.82')   ('DU 1', '0.72')   ('DU 4', '0.80')   ('DU 3', '0.39')
('DU 4', '0.82')   ('DU 4', '0.70')   ('DU 8', '0.78')   ('IB 3', '0.37')
('IB 1', '0.76')   ('IB 1', '0.63')   ('IB 1', '0.76')   ('DU 4', '0.32')
```

Figure .106: The results from comparison to entry labeled "RD 9".

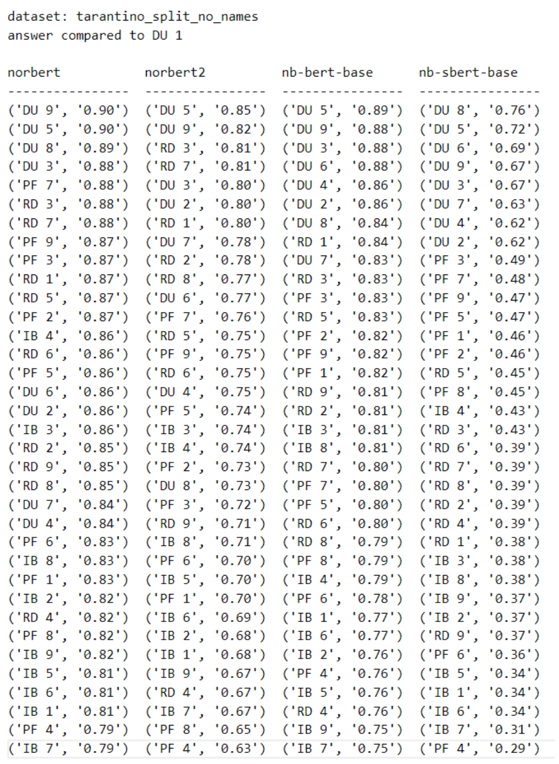
B5 - Tarantino single descriptions dataset without titles

All results from the tests performed on the Tarantino dataset with one movie descriptions per entry and titles removed.



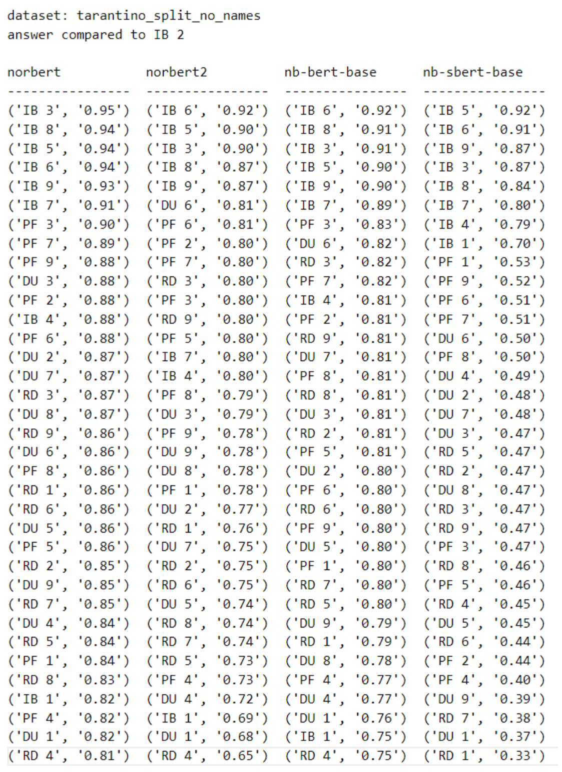
```
dataset: tarantino_split_no_names
answer compared to IB 1

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('IB 4', '0.88')  ('IB 4', '0.78')  ('IB 4', '0.82')  ('IB 4', '0.75')
('IB 5', '0.84')  ('RD 1', '0.71')  ('IB 8', '0.81')  ('IB 2', '0.70')
('IB 7', '0.84')  ('PF 1', '0.71')  ('IB 3', '0.79')  ('IB 8', '0.69')
('IB 3', '0.83')  ('IB 2', '0.69')  ('IB 7', '0.79')  ('IB 5', '0.68')
('PF 3', '0.83')  ('RD 3', '0.69')  ('PF 3', '0.78')  ('IB 7', '0.66')
('IB 2', '0.82')  ('IB 5', '0.69')  ('PF 1', '0.78')  ('IB 9', '0.64')
('IB 9', '0.82')  ('IB 9', '0.69')  ('RD 5', '0.78')  ('IB 6', '0.62')
('IB 8', '0.82')  ('PF 3', '0.68')  ('IB 5', '0.78')  ('IB 3', '0.61')
('IB 6', '0.82')  ('IB 6', '0.68')  ('RD 1', '0.77')  ('RD 8', '0.54')
('RD 1', '0.82')  ('DU 1', '0.68')  ('DU 5', '0.77')  ('PF 1', '0.54')
('RD 3', '0.82')  ('IB 3', '0.68')  ('PF 9', '0.77')  ('RD 4', '0.53')
('PF 1', '0.81')  ('IB 7', '0.67')  ('DU 1', '0.77')  ('RD 2', '0.53')
('DU 1', '0.81')  ('PF 5', '0.67')  ('RD 8', '0.77')  ('RD 5', '0.53')
('PF 7', '0.80')  ('RD 2', '0.66')  ('DU 2', '0.76')  ('RD 9', '0.52')
('PF 9', '0.80')  ('DU 6', '0.66')  ('RD 6', '0.76')  ('RD 3', '0.52')
('RD 5', '0.80')  ('RD 6', '0.66')  ('RD 7', '0.76')  ('RD 6', '0.51')
('RD 6', '0.79')  ('IB 8', '0.66')  ('PF 5', '0.76')  ('PF 8', '0.48')
('DU 4', '0.79')  ('PF 7', '0.66')  ('PF 7', '0.76')  ('PF 6', '0.46')
('PF 2', '0.79')  ('PF 2', '0.66')  ('IB 9', '0.76')  ('PF 7', '0.46')
('DU 6', '0.79')  ('RD 8', '0.65')  ('RD 2', '0.76')  ('RD 7', '0.46')
('DU 7', '0.79')  ('PF 6', '0.65')  ('RD 9', '0.75')  ('PF 3', '0.45')
('RD 8', '0.79')  ('PF 4', '0.65')  ('DU 6', '0.75')  ('PF 9', '0.44')
('RD 7', '0.79')  ('RD 5', '0.65')  ('RD 3', '0.75')  ('PF 5', '0.44')
('DU 2', '0.78')  ('RD 7', '0.64')  ('DU 9', '0.75')  ('PF 4', '0.42')
('RD 2', '0.77')  ('RD 9', '0.64')  ('PF 4', '0.75')  ('RD 1', '0.41')
('RD 9', '0.77')  ('PF 8', '0.64')  ('IB 2', '0.75')  ('DU 4', '0.40')
('PF 6', '0.77')  ('DU 9', '0.64')  ('RD 4', '0.74')  ('DU 6', '0.39')
('DU 3', '0.77')  ('PF 9', '0.63')  ('PF 8', '0.74')  ('DU 2', '0.38')
('PF 5', '0.77')  ('DU 2', '0.63')  ('PF 2', '0.74')  ('DU 8', '0.38')
('DU 8', '0.77')  ('DU 5', '0.63')  ('DU 4', '0.74')  ('PF 2', '0.38')
('DU 5', '0.76')  ('DU 3', '0.62')  ('DU 3', '0.74')  ('DU 3', '0.35')
('PF 8', '0.75')  ('DU 7', '0.62')  ('IB 6', '0.73')  ('DU 7', '0.35')
('PF 4', '0.75')  ('RD 4', '0.60')  ('DU 7', '0.73')  ('DU 5', '0.34')
('RD 4', '0.75')  ('DU 4', '0.59')  ('PF 6', '0.72')  ('DU 1', '0.34')
('DU 9', '0.75')  ('DU 8', '0.57')  ('DU 8', '0.68')  ('DU 9', '0.27')
```

Figure .107: The results from comparison to entry labeled "IB 1".



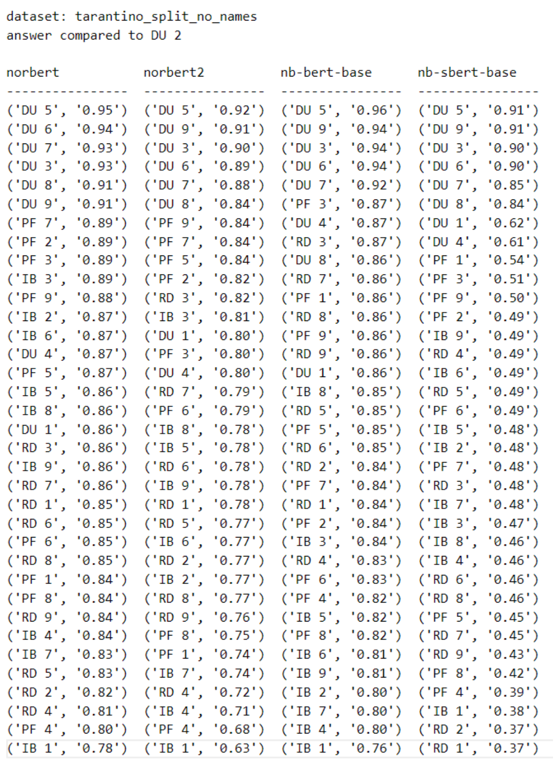
```
dataset: tarantino_split_no_names
answer compared to DU 1

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 9', '0.90')   ('DU 5', '0.85')   ('DU 5', '0.89')   ('DU 8', '0.76')
('DU 5', '0.90')   ('DU 9', '0.82')   ('DU 9', '0.88')   ('DU 5', '0.72')
('DU 8', '0.89')   ('RD 3', '0.81')   ('DU 3', '0.88')   ('DU 6', '0.69')
('DU 3', '0.88')   ('RD 7', '0.81')   ('DU 6', '0.88')   ('DU 9', '0.67')
('PF 7', '0.88')   ('DU 3', '0.80')   ('DU 4', '0.86')   ('DU 3', '0.67')
('RD 3', '0.88')   ('DU 2', '0.80')   ('DU 2', '0.86')   ('DU 7', '0.63')
('RD 7', '0.88')   ('RD 1', '0.80')   ('DU 8', '0.84')   ('DU 4', '0.62')
('PF 9', '0.87')   ('DU 7', '0.78')   ('RD 1', '0.84')   ('DU 2', '0.62')
('PF 3', '0.87')   ('RD 2', '0.78')   ('DU 7', '0.83')   ('PF 3', '0.49')
('RD 1', '0.87')   ('RD 8', '0.77')   ('RD 3', '0.83')   ('PF 7', '0.48')
('RD 5', '0.87')   ('DU 6', '0.77')   ('PF 3', '0.83')   ('PF 9', '0.47')
('PF 2', '0.87')   ('PF 7', '0.76')   ('RD 5', '0.83')   ('PF 5', '0.47')
('IB 4', '0.86')   ('RD 5', '0.75')   ('PF 2', '0.82')   ('PF 1', '0.46')
('RD 6', '0.86')   ('PF 9', '0.75')   ('PF 9', '0.82')   ('PF 2', '0.46')
('PF 5', '0.86')   ('RD 6', '0.75')   ('PF 1', '0.82')   ('RD 5', '0.45')
('DU 6', '0.86')   ('DU 4', '0.75')   ('RD 9', '0.81')   ('PF 8', '0.45')
('DU 2', '0.86')   ('PF 5', '0.74')   ('RD 2', '0.81')   ('IB 4', '0.43')
('IB 3', '0.86')   ('IB 3', '0.74')   ('IB 3', '0.81')   ('RD 3', '0.43')
('RD 2', '0.85')   ('IB 4', '0.74')   ('IB 8', '0.81')   ('RD 6', '0.39')
('RD 9', '0.85')   ('PF 2', '0.73')   ('RD 7', '0.80')   ('RD 7', '0.39')
('RD 8', '0.85')   ('DU 8', '0.73')   ('PF 7', '0.80')   ('RD 8', '0.39')
('DU 7', '0.84')   ('PF 3', '0.72')   ('PF 5', '0.80')   ('RD 2', '0.39')
('DU 4', '0.84')   ('RD 9', '0.71')   ('RD 6', '0.80')   ('RD 4', '0.39')
('PF 6', '0.83')   ('IB 8', '0.71')   ('RD 8', '0.79')   ('RD 1', '0.38')
('IB 8', '0.83')   ('PF 6', '0.70')   ('PF 8', '0.79')   ('IB 3', '0.38')
('PF 1', '0.83')   ('IB 5', '0.70')   ('IB 4', '0.79')   ('IB 8', '0.38')
('IB 2', '0.82')   ('PF 1', '0.70')   ('PF 6', '0.78')   ('IB 9', '0.37')
('RD 4', '0.82')   ('IB 6', '0.69')   ('IB 1', '0.77')   ('IB 2', '0.37')
('PF 8', '0.82')   ('IB 2', '0.68')   ('IB 6', '0.77')   ('RD 9', '0.37')
('IB 9', '0.82')   ('IB 1', '0.68')   ('IB 2', '0.76')   ('PF 6', '0.36')
('IB 5', '0.81')   ('IB 9', '0.67')   ('PF 4', '0.76')   ('IB 5', '0.34')
('IB 6', '0.81')   ('RD 4', '0.67')   ('IB 5', '0.76')   ('IB 1', '0.34')
('IB 1', '0.81')   ('IB 7', '0.67')   ('RD 4', '0.76')   ('IB 6', '0.34')
('PF 4', '0.79')   ('PF 8', '0.65')   ('IB 9', '0.75')   ('IB 7', '0.31')
('IB 7', '0.79')   ('PF 4', '0.63')   ('IB 7', '0.75')   ('PF 4', '0.29')
```

Figure .108: The results from comparison to entry labeled "DU 1".



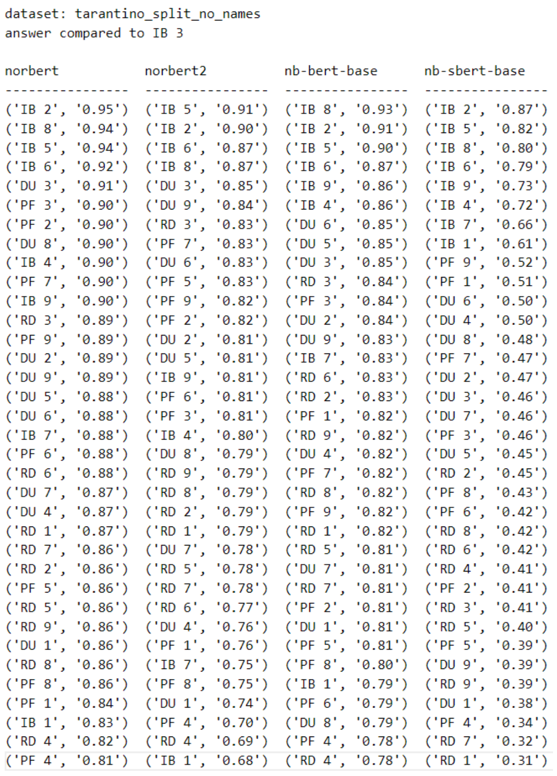
```
dataset: tarantino_split_no_names
answer compared to IB 2

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('IB 3', '0.95')   ('IB 6', '0.92')   ('IB 6', '0.92')   ('IB 5', '0.92')
('IB 8', '0.94')   ('IB 5', '0.90')   ('IB 8', '0.91')   ('IB 6', '0.91')
('IB 5', '0.94')   ('IB 3', '0.90')   ('IB 3', '0.91')   ('IB 9', '0.87')
('IB 6', '0.94')   ('IB 8', '0.87')   ('IB 5', '0.90')   ('IB 3', '0.87')
('IB 9', '0.93')   ('IB 9', '0.87')   ('IB 9', '0.90')   ('IB 8', '0.84')
('IB 7', '0.91')   ('DU 6', '0.81')   ('IB 7', '0.89')   ('IB 7', '0.80')
('PF 3', '0.90')   ('PF 6', '0.81')   ('PF 3', '0.83')   ('IB 4', '0.79')
('PF 7', '0.89')   ('PF 2', '0.80')   ('DU 6', '0.82')   ('IB 1', '0.70')
('PF 9', '0.88')   ('PF 7', '0.80')   ('RD 3', '0.82')   ('PF 1', '0.53')
('DU 3', '0.88')   ('RD 3', '0.80')   ('PF 7', '0.82')   ('PF 9', '0.52')
('PF 2', '0.88')   ('PF 3', '0.80')   ('IB 4', '0.81')   ('PF 6', '0.51')
('IB 4', '0.88')   ('RD 9', '0.80')   ('PF 2', '0.81')   ('PF 7', '0.51')
('PF 6', '0.88')   ('PF 5', '0.80')   ('RD 9', '0.81')   ('DU 6', '0.50')
('DU 2', '0.87')   ('IB 7', '0.80')   ('DU 7', '0.81')   ('PF 8', '0.50')
('DU 7', '0.87')   ('IB 4', '0.80')   ('PF 8', '0.81')   ('DU 4', '0.49')
('RD 3', '0.87')   ('PF 8', '0.79')   ('RD 8', '0.81')   ('DU 2', '0.48')
('DU 8', '0.87')   ('DU 3', '0.79')   ('DU 3', '0.81')   ('DU 7', '0.48')
('RD 9', '0.86')   ('PF 9', '0.78')   ('RD 2', '0.81')   ('DU 3', '0.47')
('DU 6', '0.86')   ('DU 9', '0.78')   ('PF 5', '0.81')   ('RD 5', '0.47')
('PF 8', '0.86')   ('DU 8', '0.78')   ('DU 2', '0.80')   ('RD 2', '0.47')
('RD 1', '0.86')   ('PF 1', '0.78')   ('PF 6', '0.80')   ('DU 8', '0.47')
('RD 6', '0.86')   ('DU 2', '0.77')   ('RD 6', '0.80')   ('RD 3', '0.47')
('DU 5', '0.86')   ('RD 1', '0.76')   ('PF 9', '0.80')   ('RD 9', '0.47')
('PF 5', '0.86')   ('DU 7', '0.75')   ('DU 5', '0.80')   ('PF 3', '0.47')
('RD 2', '0.85')   ('RD 2', '0.75')   ('PF 1', '0.80')   ('RD 8', '0.46')
('DU 9', '0.85')   ('RD 6', '0.75')   ('RD 7', '0.80')   ('PF 5', '0.46')
('RD 7', '0.85')   ('DU 5', '0.74')   ('RD 5', '0.80')   ('RD 4', '0.45')
('DU 4', '0.84')   ('RD 8', '0.74')   ('DU 9', '0.79')   ('DU 5', '0.45')
('RD 5', '0.84')   ('RD 7', '0.74')   ('RD 1', '0.79')   ('RD 6', '0.44')
('PF 1', '0.84')   ('RD 5', '0.73')   ('DU 8', '0.78')   ('PF 2', '0.44')
('RD 8', '0.83')   ('PF 4', '0.73')   ('PF 4', '0.77')   ('PF 4', '0.40')
('IB 1', '0.82')   ('DU 4', '0.72')   ('DU 4', '0.77')   ('DU 9', '0.39')
('PF 4', '0.82')   ('IB 1', '0.69')   ('DU 1', '0.76')   ('RD 7', '0.38')
('DU 1', '0.82')   ('DU 1', '0.68')   ('IB 1', '0.75')   ('DU 1', '0.37')
('RD 4', '0.81')   ('RD 4', '0.65')   ('RD 4', '0.75')   ('RD 1', '0.33')
```

Figure .109: The results from comparison to entry labeled "IB 2".



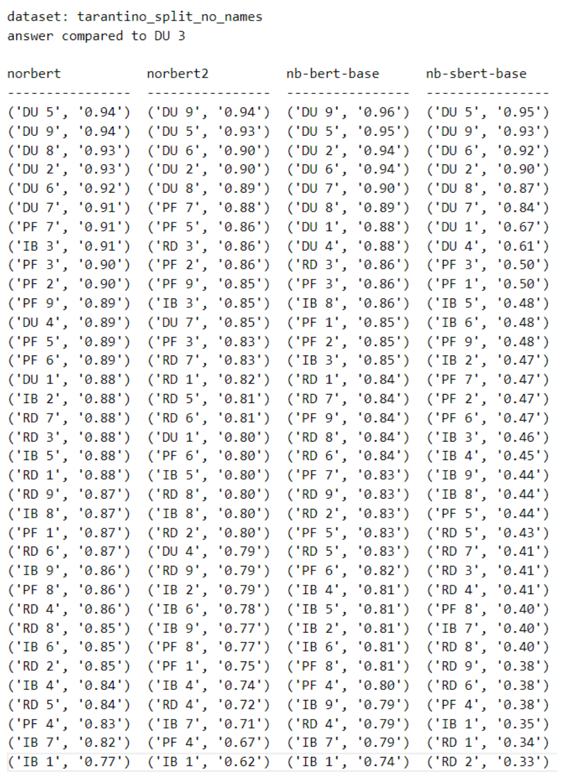
```
dataset: tarantino_split_no_names
answer compared to DU 2

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 5', '0.95')   ('DU 5', '0.92')   ('DU 5', '0.96')   ('DU 5', '0.91')
('DU 6', '0.94')   ('DU 9', '0.91')   ('DU 9', '0.94')   ('DU 9', '0.91')
('DU 7', '0.93')   ('DU 3', '0.90')   ('DU 3', '0.94')   ('DU 3', '0.90')
('DU 3', '0.93')   ('DU 6', '0.89')   ('DU 6', '0.94')   ('DU 6', '0.90')
('DU 8', '0.91')   ('DU 7', '0.88')   ('DU 7', '0.92')   ('DU 7', '0.85')
('DU 9', '0.91')   ('DU 8', '0.84')   ('PF 3', '0.87')   ('DU 8', '0.84')
('PF 7', '0.89')   ('PF 9', '0.84')   ('DU 4', '0.87')   ('DU 1', '0.62')
('PF 2', '0.89')   ('PF 7', '0.84')   ('RD 3', '0.87')   ('DU 4', '0.61')
('PF 3', '0.89')   ('PF 5', '0.84')   ('DU 8', '0.86')   ('PF 1', '0.54')
('IB 3', '0.89')   ('PF 2', '0.82')   ('RD 7', '0.86')   ('PF 3', '0.51')
('PF 9', '0.88')   ('RD 3', '0.82')   ('PF 1', '0.86')   ('PF 9', '0.50')
('IB 2', '0.87')   ('IB 3', '0.81')   ('RD 8', '0.86')   ('PF 2', '0.49')
('IB 6', '0.87')   ('DU 1', '0.80')   ('PF 9', '0.86')   ('IB 9', '0.49')
('DU 4', '0.87')   ('PF 3', '0.80')   ('RD 9', '0.86')   ('RD 4', '0.49')
('PF 5', '0.87')   ('DU 4', '0.80')   ('DU 1', '0.86')   ('IB 6', '0.49')
('IB 5', '0.86')   ('RD 7', '0.79')   ('IB 8', '0.85')   ('RD 5', '0.49')
('IB 8', '0.86')   ('PF 6', '0.79')   ('RD 5', '0.85')   ('PF 6', '0.49')
('DU 1', '0.86')   ('IB 8', '0.78')   ('PF 5', '0.85')   ('IB 5', '0.48')
('RD 3', '0.86')   ('IB 5', '0.78')   ('RD 6', '0.85')   ('IB 2', '0.48')
('IB 9', '0.86')   ('RD 6', '0.78')   ('RD 2', '0.84')   ('PF 7', '0.48')
('RD 7', '0.86')   ('IB 9', '0.78')   ('PF 7', '0.84')   ('RD 3', '0.48')
('RD 1', '0.85')   ('RD 1', '0.78')   ('RD 1', '0.84')   ('IB 7', '0.48')
('RD 6', '0.85')   ('RD 5', '0.77')   ('PF 2', '0.84')   ('IB 3', '0.47')
('PF 6', '0.85')   ('IB 6', '0.77')   ('IB 3', '0.84')   ('IB 8', '0.46')
('RD 8', '0.85')   ('RD 2', '0.77')   ('RD 4', '0.83')   ('IB 4', '0.46')
('PF 1', '0.84')   ('IB 2', '0.77')   ('PF 6', '0.83')   ('RD 6', '0.46')
('PF 8', '0.84')   ('RD 8', '0.77')   ('PF 4', '0.82')   ('RD 8', '0.46')
('RD 9', '0.84')   ('RD 9', '0.76')   ('IB 5', '0.82')   ('PF 5', '0.45')
('IB 4', '0.84')   ('PF 8', '0.75')   ('PF 8', '0.82')   ('RD 7', '0.45')
('IB 7', '0.83')   ('PF 1', '0.74')   ('IB 6', '0.81')   ('RD 9', '0.43')
('RD 5', '0.83')   ('IB 7', '0.74')   ('IB 9', '0.81')   ('PF 8', '0.42')
('RD 2', '0.82')   ('RD 4', '0.72')   ('IB 2', '0.80')   ('PF 4', '0.39')
('RD 4', '0.81')   ('IB 4', '0.71')   ('IB 7', '0.80')   ('IB 1', '0.38')
('PF 4', '0.80')   ('PF 4', '0.68')   ('IB 4', '0.80')   ('RD 2', '0.37')
('IB 1', '0.78')   ('IB 1', '0.63')   ('IB 1', '0.76')   ('RD 1', '0.37')
```

Figure .110: The results from comparison to entry labeled "DU 2".



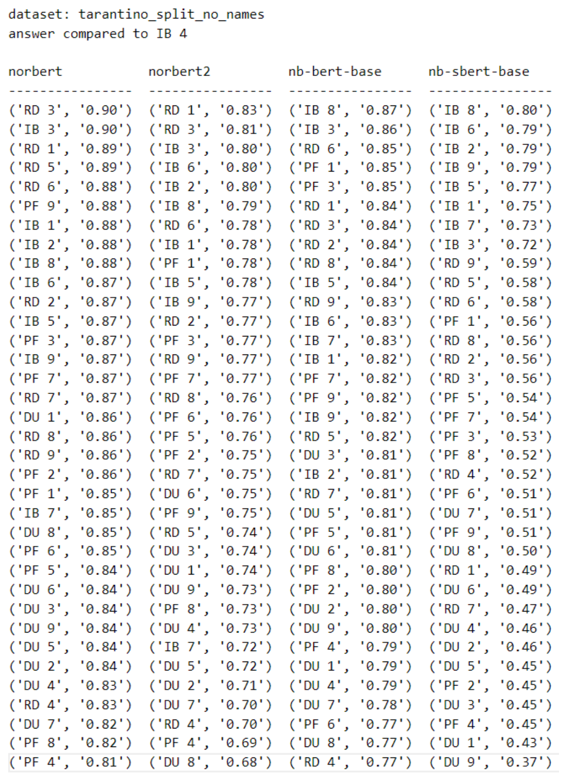
```
dataset: tarantino_split_no_names
answer compared to IB 3

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('IB 2', '0.95')   ('IB 5', '0.91')   ('IB 8', '0.93')   ('IB 2', '0.87')
('IB 8', '0.94')   ('IB 2', '0.90')   ('IB 2', '0.91')   ('IB 5', '0.82')
('IB 5', '0.94')   ('IB 6', '0.87')   ('IB 5', '0.90')   ('IB 8', '0.80')
('IB 6', '0.92')   ('IB 8', '0.87')   ('IB 6', '0.87')   ('IB 6', '0.79')
('DU 3', '0.91')   ('DU 3', '0.85')   ('IB 9', '0.86')   ('IB 9', '0.73')
('PF 3', '0.90')   ('DU 9', '0.84')   ('IB 4', '0.86')   ('IB 4', '0.72')
('PF 2', '0.90')   ('RD 3', '0.83')   ('DU 6', '0.85')   ('IB 7', '0.66')
('DU 8', '0.90')   ('PF 7', '0.83')   ('DU 5', '0.85')   ('IB 1', '0.61')
('IB 4', '0.90')   ('DU 6', '0.83')   ('DU 3', '0.85')   ('PF 9', '0.52')
('PF 7', '0.90')   ('PF 5', '0.83')   ('RD 3', '0.84')   ('PF 1', '0.51')
('IB 9', '0.90')   ('PF 9', '0.82')   ('PF 3', '0.84')   ('DU 6', '0.50')
('RD 3', '0.89')   ('PF 2', '0.82')   ('DU 2', '0.84')   ('DU 4', '0.50')
('PF 9', '0.89')   ('DU 2', '0.81')   ('DU 9', '0.83')   ('DU 8', '0.48')
('DU 2', '0.89')   ('DU 5', '0.81')   ('IB 7', '0.83')   ('PF 7', '0.47')
('DU 9', '0.89')   ('IB 9', '0.81')   ('RD 6', '0.83')   ('DU 2', '0.47')
('DU 5', '0.88')   ('PF 6', '0.81')   ('RD 2', '0.83')   ('DU 3', '0.46')
('DU 6', '0.88')   ('PF 3', '0.81')   ('PF 1', '0.82')   ('DU 7', '0.46')
('IB 7', '0.88')   ('IB 4', '0.80')   ('RD 9', '0.82')   ('PF 3', '0.46')
('PF 6', '0.88')   ('DU 8', '0.79')   ('DU 4', '0.82')   ('DU 5', '0.45')
('RD 6', '0.88')   ('RD 9', '0.79')   ('PF 7', '0.82')   ('RD 2', '0.45')
('DU 7', '0.87')   ('RD 8', '0.79')   ('RD 8', '0.82')   ('PF 8', '0.43')
('DU 4', '0.87')   ('RD 2', '0.79')   ('PF 9', '0.82')   ('PF 6', '0.42')
('RD 1', '0.87')   ('RD 1', '0.79')   ('RD 1', '0.82')   ('RD 8', '0.42')
('RD 7', '0.86')   ('DU 7', '0.78')   ('RD 5', '0.81')   ('RD 6', '0.42')
('RD 2', '0.86')   ('RD 5', '0.78')   ('DU 7', '0.81')   ('RD 4', '0.41')
('PF 5', '0.86')   ('RD 7', '0.78')   ('RD 7', '0.81')   ('PF 2', '0.41')
('RD 5', '0.86')   ('RD 6', '0.77')   ('PF 2', '0.81')   ('RD 3', '0.41')
('RD 9', '0.86')   ('DU 4', '0.76')   ('DU 1', '0.81')   ('RD 5', '0.40')
('DU 1', '0.86')   ('PF 1', '0.76')   ('PF 5', '0.81')   ('PF 5', '0.39')
('RD 8', '0.86')   ('IB 7', '0.75')   ('PF 8', '0.80')   ('DU 9', '0.39')
('PF 8', '0.86')   ('PF 8', '0.75')   ('IB 1', '0.79')   ('RD 9', '0.39')
('PF 1', '0.84')   ('DU 1', '0.74')   ('PF 6', '0.79')   ('DU 1', '0.38')
('IB 1', '0.83')   ('PF 4', '0.70')   ('DU 8', '0.79')   ('PF 4', '0.34')
('RD 4', '0.82')   ('RD 4', '0.69')   ('PF 4', '0.78')   ('RD 7', '0.32')
('PF 4', '0.81')   ('IB 1', '0.68')   ('RD 4', '0.78')   ('RD 1', '0.31')
```

Figure .111: The results from comparison to entry labeled "IB 3".



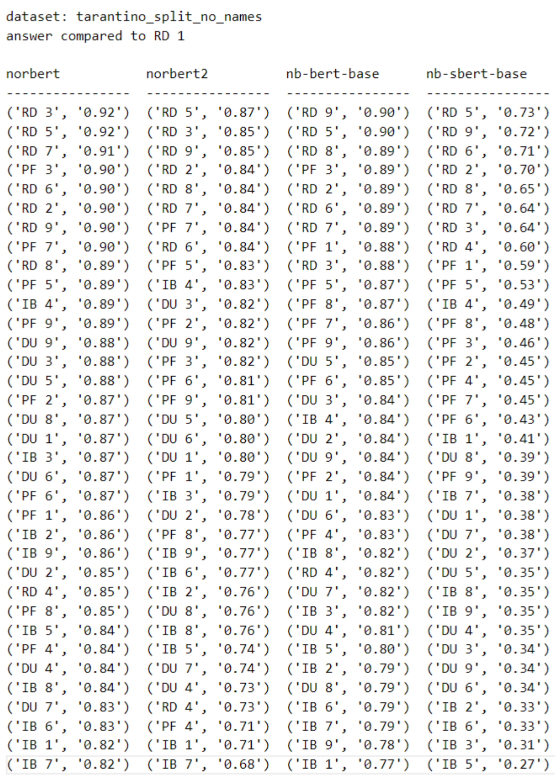
```
dataset: tarantino_split_no_names
answer compared to DU 3

norbert           norbert2          nb-bert-base      nb-sbert-base
---------------   ---------------   ---------------   ---------------
('DU 5', '0.94')  ('DU 9', '0.94')  ('DU 9', '0.96')  ('DU 5', '0.95')
('DU 9', '0.94')  ('DU 5', '0.93')  ('DU 5', '0.95')  ('DU 9', '0.93')
('DU 8', '0.93')  ('DU 6', '0.90')  ('DU 2', '0.94')  ('DU 6', '0.92')
('DU 2', '0.93')  ('DU 2', '0.90')  ('DU 6', '0.94')  ('DU 2', '0.90')
('DU 6', '0.92')  ('DU 8', '0.89')  ('DU 7', '0.90')  ('DU 8', '0.87')
('DU 7', '0.91')  ('PF 7', '0.88')  ('DU 8', '0.89')  ('DU 7', '0.84')
('PF 7', '0.91')  ('PF 5', '0.86')  ('DU 1', '0.88')  ('DU 1', '0.67')
('IB 3', '0.91')  ('RD 3', '0.86')  ('DU 4', '0.88')  ('DU 4', '0.61')
('PF 3', '0.90')  ('PF 2', '0.86')  ('RD 3', '0.86')  ('PF 3', '0.50')
('PF 2', '0.90')  ('PF 9', '0.85')  ('PF 3', '0.86')  ('PF 1', '0.50')
('PF 9', '0.89')  ('IB 3', '0.85')  ('IB 8', '0.86')  ('IB 5', '0.48')
('DU 4', '0.89')  ('DU 7', '0.85')  ('PF 1', '0.85')  ('IB 6', '0.48')
('PF 5', '0.89')  ('PF 3', '0.83')  ('PF 2', '0.85')  ('PF 9', '0.48')
('PF 6', '0.89')  ('RD 7', '0.83')  ('IB 3', '0.85')  ('IB 2', '0.47')
('DU 1', '0.88')  ('RD 1', '0.82')  ('RD 1', '0.84')  ('PF 7', '0.47')
('IB 2', '0.88')  ('RD 5', '0.81')  ('RD 7', '0.84')  ('PF 2', '0.47')
('RD 7', '0.88')  ('RD 6', '0.81')  ('PF 9', '0.84')  ('PF 6', '0.47')
('RD 3', '0.88')  ('DU 1', '0.80')  ('RD 8', '0.84')  ('IB 3', '0.46')
('IB 5', '0.88')  ('PF 6', '0.80')  ('RD 6', '0.84')  ('IB 4', '0.45')
('RD 1', '0.88')  ('IB 5', '0.80')  ('PF 7', '0.83')  ('IB 9', '0.44')
('RD 9', '0.87')  ('RD 8', '0.80')  ('RD 9', '0.83')  ('IB 8', '0.44')
('IB 8', '0.87')  ('IB 8', '0.80')  ('RD 2', '0.83')  ('PF 5', '0.44')
('PF 1', '0.87')  ('RD 2', '0.80')  ('PF 5', '0.83')  ('RD 5', '0.43')
('RD 6', '0.87')  ('DU 4', '0.79')  ('RD 5', '0.83')  ('RD 7', '0.41')
('IB 9', '0.86')  ('RD 9', '0.79')  ('PF 6', '0.82')  ('RD 3', '0.41')
('PF 8', '0.86')  ('IB 2', '0.79')  ('IB 4', '0.81')  ('RD 4', '0.41')
('RD 4', '0.86')  ('IB 6', '0.78')  ('IB 5', '0.81')  ('PF 8', '0.40')
('RD 8', '0.85')  ('IB 9', '0.77')  ('IB 2', '0.81')  ('IB 7', '0.40')
('IB 6', '0.85')  ('PF 8', '0.77')  ('IB 6', '0.81')  ('RD 8', '0.40')
('RD 2', '0.85')  ('PF 1', '0.75')  ('PF 8', '0.81')  ('RD 9', '0.38')
('IB 4', '0.84')  ('IB 4', '0.74')  ('PF 4', '0.80')  ('RD 6', '0.38')
('RD 5', '0.84')  ('RD 4', '0.72')  ('IB 9', '0.79')  ('PF 4', '0.38')
('PF 4', '0.83')  ('IB 7', '0.71')  ('RD 4', '0.79')  ('IB 1', '0.35')
('IB 7', '0.82')  ('PF 4', '0.67')  ('IB 7', '0.79')  ('RD 1', '0.34')
('IB 1', '0.77')  ('IB 1', '0.62')  ('IB 1', '0.74')  ('RD 2', '0.33')
```

Figure .112: The results from comparison to entry labeled "DU 3".



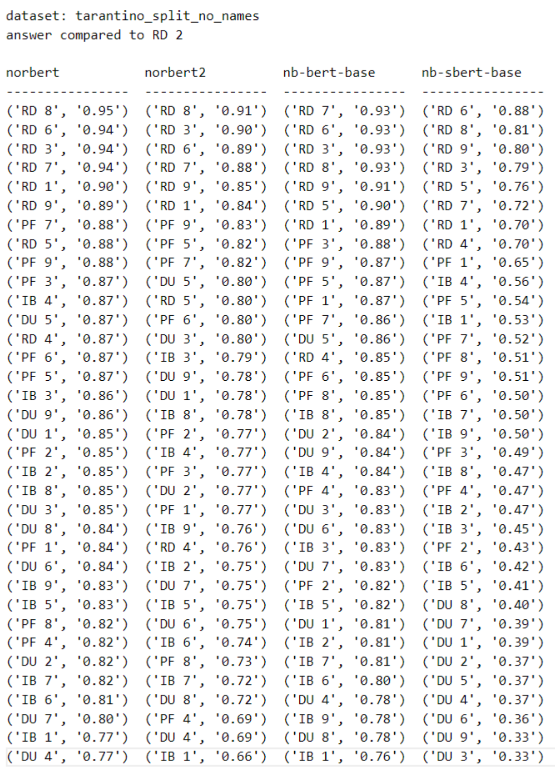
```
dataset: tarantino_split_no_names
answer compared to IB 4

norbert           norbert2          nb-bert-base      nb-sbert-base
---------------   ---------------   ---------------   ---------------
('RD 3', '0.90')  ('RD 1', '0.83')  ('IB 8', '0.87')  ('IB 8', '0.80')
('IB 3', '0.90')  ('RD 3', '0.81')  ('IB 3', '0.86')  ('IB 6', '0.79')
('RD 1', '0.89')  ('IB 3', '0.80')  ('RD 6', '0.85')  ('IB 2', '0.79')
('RD 5', '0.89')  ('IB 6', '0.80')  ('PF 1', '0.85')  ('IB 9', '0.79')
('RD 6', '0.88')  ('IB 2', '0.80')  ('PF 3', '0.85')  ('IB 5', '0.77')
('PF 9', '0.88')  ('IB 8', '0.79')  ('RD 1', '0.84')  ('IB 1', '0.75')
('IB 1', '0.88')  ('RD 6', '0.78')  ('RD 3', '0.84')  ('IB 7', '0.73')
('IB 2', '0.88')  ('IB 1', '0.78')  ('RD 2', '0.84')  ('IB 3', '0.72')
('IB 8', '0.88')  ('PF 1', '0.78')  ('RD 8', '0.84')  ('RD 9', '0.59')
('IB 6', '0.87')  ('IB 5', '0.78')  ('IB 5', '0.84')  ('RD 5', '0.58')
('RD 2', '0.87')  ('IB 9', '0.77')  ('RD 9', '0.83')  ('RD 6', '0.58')
('IB 5', '0.87')  ('RD 2', '0.77')  ('IB 6', '0.83')  ('PF 1', '0.56')
('PF 3', '0.87')  ('PF 3', '0.77')  ('IB 7', '0.83')  ('RD 8', '0.56')
('IB 9', '0.87')  ('RD 9', '0.77')  ('IB 1', '0.82')  ('RD 2', '0.56')
('PF 7', '0.87')  ('PF 7', '0.77')  ('PF 7', '0.82')  ('RD 3', '0.56')
('RD 7', '0.87')  ('RD 8', '0.76')  ('PF 9', '0.82')  ('PF 5', '0.54')
('DU 1', '0.86')  ('PF 6', '0.76')  ('IB 9', '0.82')  ('PF 7', '0.54')
('RD 8', '0.86')  ('PF 5', '0.76')  ('RD 5', '0.82')  ('PF 3', '0.53')
('RD 9', '0.86')  ('PF 2', '0.75')  ('DU 3', '0.81')  ('PF 8', '0.52')
('PF 2', '0.86')  ('RD 7', '0.75')  ('IB 2', '0.81')  ('RD 4', '0.52')
('PF 1', '0.85')  ('DU 6', '0.75')  ('RD 7', '0.81')  ('PF 6', '0.51')
('IB 7', '0.85')  ('PF 9', '0.75')  ('DU 5', '0.81')  ('DU 7', '0.51')
('DU 8', '0.85')  ('RD 5', '0.74')  ('PF 5', '0.81')  ('PF 9', '0.51')
('PF 6', '0.85')  ('DU 3', '0.74')  ('DU 6', '0.81')  ('DU 8', '0.50')
('PF 5', '0.84')  ('DU 1', '0.74')  ('PF 8', '0.80')  ('RD 1', '0.49')
('DU 6', '0.84')  ('DU 9', '0.73')  ('PF 2', '0.80')  ('DU 6', '0.49')
('DU 3', '0.84')  ('PF 8', '0.73')  ('DU 2', '0.80')  ('RD 7', '0.47')
('DU 9', '0.84')  ('DU 4', '0.73')  ('DU 9', '0.80')  ('DU 4', '0.46')
('DU 5', '0.84')  ('IB 7', '0.72')  ('PF 4', '0.79')  ('DU 2', '0.46')
('DU 2', '0.84')  ('DU 5', '0.72')  ('DU 1', '0.79')  ('DU 5', '0.45')
('DU 4', '0.83')  ('DU 2', '0.71')  ('DU 4', '0.79')  ('PF 2', '0.45')
('RD 4', '0.83')  ('DU 7', '0.70')  ('DU 7', '0.78')  ('DU 3', '0.45')
('DU 7', '0.82')  ('RD 4', '0.70')  ('PF 6', '0.77')  ('PF 4', '0.45')
('PF 8', '0.82')  ('PF 4', '0.69')  ('DU 8', '0.77')  ('DU 1', '0.43')
('PF 4', '0.81')  ('DU 8', '0.68')  ('RD 4', '0.77')  ('DU 9', '0.37')
```

Figure .113: The results from comparison to entry labeled "IB 4".



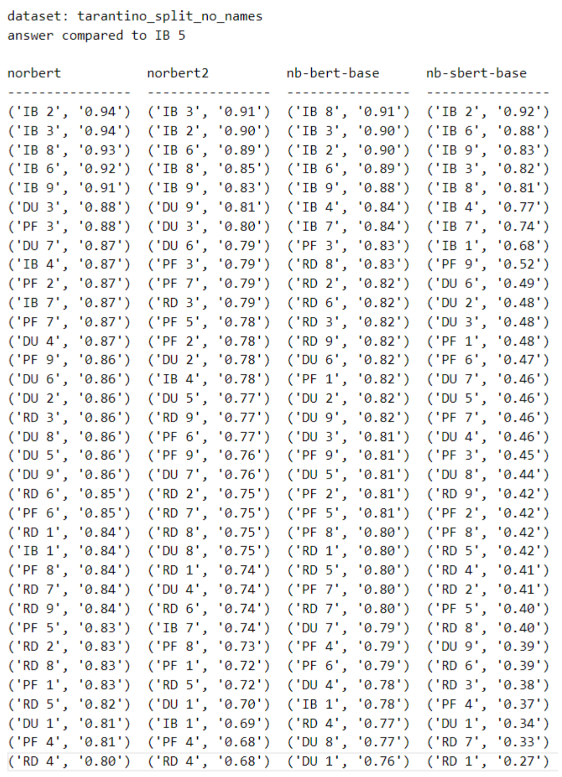
```
dataset: tarantino_split_no_names
answer compared to RD 1

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('RD 3', '0.92')  ('RD 5', '0.87')  ('RD 9', '0.90')  ('RD 5', '0.73')
('RD 5', '0.92')  ('RD 3', '0.85')  ('RD 5', '0.90')  ('RD 9', '0.72')
('RD 7', '0.91')  ('RD 9', '0.85')  ('RD 8', '0.89')  ('RD 6', '0.71')
('PF 3', '0.90')  ('RD 2', '0.84')  ('PF 3', '0.89')  ('RD 2', '0.70')
('RD 6', '0.90')  ('RD 8', '0.84')  ('RD 2', '0.89')  ('RD 8', '0.65')
('RD 2', '0.90')  ('RD 7', '0.84')  ('RD 6', '0.89')  ('RD 7', '0.64')
('RD 9', '0.90')  ('PF 7', '0.84')  ('RD 7', '0.89')  ('RD 3', '0.64')
('PF 7', '0.90')  ('RD 6', '0.84')  ('PF 1', '0.88')  ('RD 4', '0.60')
('RD 8', '0.89')  ('PF 5', '0.83')  ('RD 3', '0.88')  ('PF 1', '0.59')
('PF 5', '0.89')  ('IB 4', '0.83')  ('PF 5', '0.87')  ('PF 5', '0.53')
('IB 4', '0.89')  ('DU 3', '0.82')  ('PF 8', '0.87')  ('IB 4', '0.49')
('PF 9', '0.89')  ('PF 2', '0.82')  ('PF 7', '0.86')  ('PF 8', '0.48')
('DU 9', '0.88')  ('DU 9', '0.82')  ('PF 9', '0.86')  ('PF 3', '0.46')
('DU 3', '0.88')  ('PF 3', '0.82')  ('DU 5', '0.85')  ('PF 2', '0.45')
('DU 5', '0.88')  ('PF 6', '0.81')  ('PF 6', '0.85')  ('PF 4', '0.45')
('PF 2', '0.87')  ('PF 9', '0.81')  ('DU 3', '0.84')  ('PF 7', '0.45')
('DU 8', '0.87')  ('DU 5', '0.80')  ('IB 4', '0.84')  ('PF 6', '0.43')
('DU 1', '0.87')  ('DU 6', '0.80')  ('DU 2', '0.84')  ('IB 1', '0.41')
('IB 3', '0.87')  ('DU 1', '0.80')  ('DU 9', '0.84')  ('DU 8', '0.39')
('DU 6', '0.87')  ('PF 1', '0.79')  ('PF 2', '0.84')  ('PF 9', '0.39')
('PF 6', '0.87')  ('IB 3', '0.79')  ('DU 1', '0.84')  ('IB 7', '0.38')
('PF 1', '0.86')  ('DU 2', '0.78')  ('DU 6', '0.83')  ('DU 1', '0.38')
('IB 2', '0.86')  ('PF 8', '0.77')  ('PF 4', '0.83')  ('DU 7', '0.38')
('IB 9', '0.86')  ('IB 9', '0.77')  ('IB 8', '0.82')  ('DU 2', '0.37')
('DU 2', '0.85')  ('IB 6', '0.77')  ('RD 4', '0.82')  ('DU 5', '0.35')
('RD 4', '0.85')  ('IB 2', '0.76')  ('DU 7', '0.82')  ('IB 8', '0.35')
('PF 8', '0.85')  ('DU 8', '0.76')  ('IB 3', '0.82')  ('IB 9', '0.35')
('IB 5', '0.84')  ('IB 8', '0.76')  ('DU 4', '0.81')  ('DU 4', '0.35')
('PF 4', '0.84')  ('IB 5', '0.74')  ('IB 5', '0.80')  ('DU 3', '0.34')
('DU 4', '0.84')  ('DU 7', '0.74')  ('IB 2', '0.79')  ('DU 9', '0.34')
('IB 8', '0.84')  ('DU 4', '0.73')  ('DU 8', '0.79')  ('DU 6', '0.34')
('DU 7', '0.83')  ('RD 4', '0.73')  ('IB 6', '0.79')  ('IB 2', '0.33')
('IB 6', '0.83')  ('PF 4', '0.71')  ('IB 7', '0.79')  ('IB 6', '0.33')
('IB 1', '0.82')  ('IB 1', '0.71')  ('IB 9', '0.78')  ('IB 3', '0.31')
('IB 7', '0.82')  ('IB 7', '0.68')  ('IB 1', '0.77')  ('IB 5', '0.27')
```

Figure .114: The results from comparison to entry labeled "RD 1".



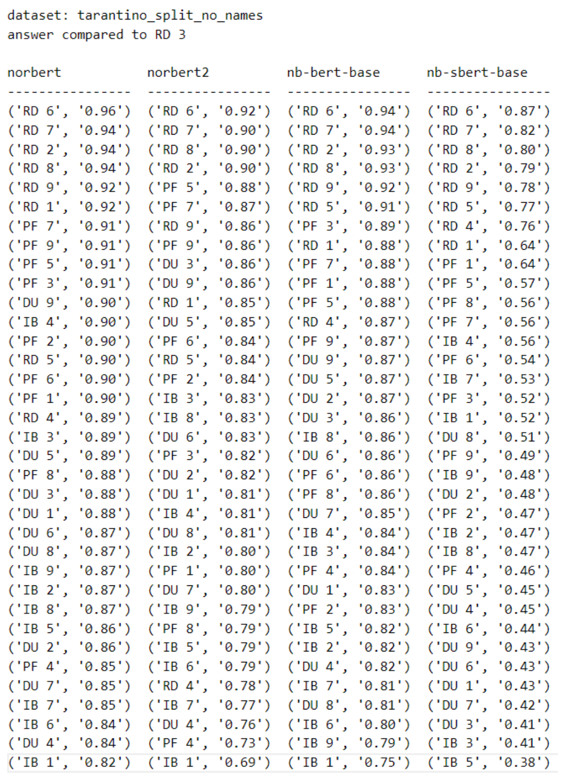
```
dataset: tarantino_split_no_names
answer compared to RD 2

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('RD 8', '0.95')  ('RD 8', '0.91')  ('RD 7', '0.93')  ('RD 6', '0.88')
('RD 6', '0.94')  ('RD 3', '0.90')  ('RD 6', '0.93')  ('RD 8', '0.81')
('RD 3', '0.94')  ('RD 6', '0.89')  ('RD 3', '0.93')  ('RD 9', '0.80')
('RD 7', '0.94')  ('RD 7', '0.88')  ('RD 8', '0.93')  ('RD 3', '0.79')
('RD 1', '0.90')  ('RD 9', '0.85')  ('RD 9', '0.91')  ('RD 5', '0.76')
('RD 9', '0.89')  ('RD 1', '0.84')  ('RD 5', '0.90')  ('RD 7', '0.72')
('PF 7', '0.88')  ('PF 9', '0.83')  ('RD 1', '0.89')  ('RD 1', '0.70')
('RD 5', '0.88')  ('PF 5', '0.82')  ('PF 3', '0.88')  ('RD 4', '0.70')
('PF 9', '0.88')  ('PF 7', '0.82')  ('PF 9', '0.87')  ('PF 1', '0.65')
('PF 3', '0.87')  ('DU 5', '0.80')  ('PF 5', '0.87')  ('IB 4', '0.56')
('IB 4', '0.87')  ('RD 5', '0.80')  ('PF 1', '0.87')  ('PF 5', '0.54')
('DU 5', '0.87')  ('PF 6', '0.80')  ('PF 7', '0.86')  ('IB 1', '0.53')
('RD 4', '0.87')  ('DU 3', '0.80')  ('DU 5', '0.86')  ('PF 7', '0.52')
('PF 6', '0.87')  ('IB 3', '0.79')  ('RD 4', '0.85')  ('PF 8', '0.51')
('PF 5', '0.87')  ('DU 9', '0.78')  ('PF 6', '0.85')  ('PF 9', '0.51')
('IB 3', '0.86')  ('DU 1', '0.78')  ('PF 8', '0.85')  ('PF 6', '0.50')
('DU 9', '0.86')  ('IB 8', '0.78')  ('IB 8', '0.85')  ('IB 7', '0.50')
('DU 1', '0.85')  ('PF 2', '0.77')  ('DU 2', '0.84')  ('IB 9', '0.50')
('PF 2', '0.85')  ('IB 4', '0.77')  ('DU 9', '0.84')  ('PF 3', '0.49')
('IB 2', '0.85')  ('PF 3', '0.77')  ('IB 4', '0.84')  ('IB 8', '0.47')
('IB 8', '0.85')  ('DU 2', '0.77')  ('PF 4', '0.83')  ('PF 4', '0.47')
('DU 3', '0.85')  ('PF 1', '0.77')  ('DU 3', '0.83')  ('IB 2', '0.47')
('DU 8', '0.84')  ('IB 9', '0.76')  ('DU 6', '0.83')  ('IB 3', '0.45')
('PF 1', '0.84')  ('RD 4', '0.76')  ('IB 3', '0.83')  ('PF 2', '0.43')
('DU 6', '0.84')  ('IB 2', '0.75')  ('DU 7', '0.83')  ('IB 6', '0.42')
('IB 9', '0.83')  ('DU 7', '0.75')  ('PF 2', '0.82')  ('IB 5', '0.41')
('IB 5', '0.83')  ('IB 5', '0.75')  ('IB 5', '0.82')  ('DU 8', '0.40')
('PF 8', '0.82')  ('DU 6', '0.75')  ('DU 1', '0.81')  ('DU 7', '0.39')
('PF 4', '0.82')  ('IB 6', '0.74')  ('IB 2', '0.81')  ('DU 1', '0.39')
('DU 2', '0.82')  ('PF 8', '0.73')  ('IB 7', '0.81')  ('DU 2', '0.37')
('IB 7', '0.82')  ('IB 7', '0.72')  ('IB 6', '0.80')  ('DU 5', '0.37')
('IB 6', '0.81')  ('DU 8', '0.72')  ('DU 4', '0.78')  ('DU 4', '0.37')
('DU 7', '0.80')  ('PF 4', '0.69')  ('IB 9', '0.78')  ('DU 6', '0.36')
('IB 1', '0.77')  ('DU 4', '0.69')  ('DU 8', '0.78')  ('DU 9', '0.33')
('DU 4', '0.77')  ('IB 1', '0.66')  ('IB 1', '0.76')  ('DU 3', '0.33')
```

Figure .115: The results from comparison to entry labeled "RD 2".



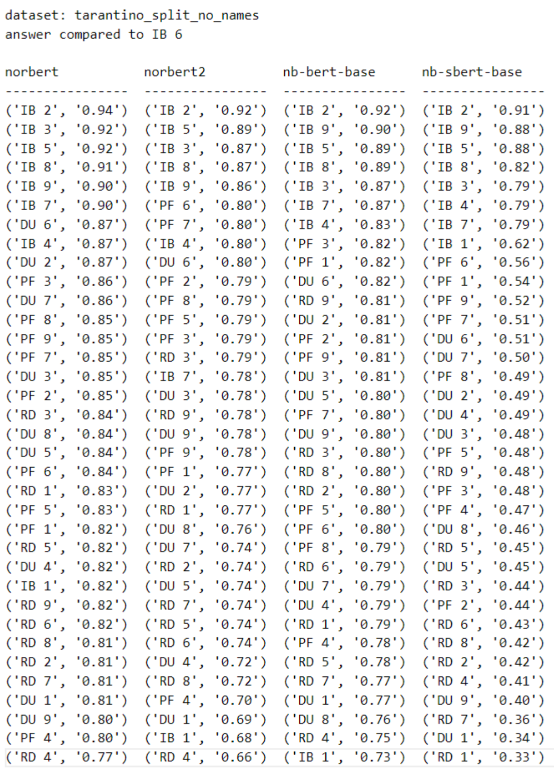
```
dataset: tarantino_split_no_names
answer compared to IB 5

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('IB 2', '0.94')  ('IB 3', '0.91')  ('IB 8', '0.91')  ('IB 2', '0.92')
('IB 3', '0.94')  ('IB 2', '0.90')  ('IB 3', '0.90')  ('IB 6', '0.88')
('IB 8', '0.93')  ('IB 6', '0.89')  ('IB 2', '0.90')  ('IB 9', '0.83')
('IB 6', '0.92')  ('IB 8', '0.85')  ('IB 6', '0.89')  ('IB 3', '0.82')
('IB 9', '0.91')  ('IB 9', '0.83')  ('IB 9', '0.88')  ('IB 8', '0.81')
('DU 3', '0.88')  ('DU 9', '0.81')  ('IB 4', '0.84')  ('IB 4', '0.77')
('PF 3', '0.88')  ('DU 3', '0.80')  ('IB 7', '0.84')  ('IB 7', '0.74')
('DU 7', '0.87')  ('DU 6', '0.79')  ('PF 3', '0.83')  ('IB 1', '0.68')
('IB 4', '0.87')  ('PF 3', '0.79')  ('RD 8', '0.83')  ('PF 9', '0.52')
('PF 2', '0.87')  ('PF 7', '0.79')  ('RD 2', '0.82')  ('DU 6', '0.49')
('IB 7', '0.87')  ('RD 3', '0.79')  ('RD 6', '0.82')  ('DU 2', '0.48')
('PF 7', '0.87')  ('PF 5', '0.78')  ('RD 3', '0.82')  ('DU 3', '0.48')
('DU 4', '0.87')  ('PF 2', '0.78')  ('RD 9', '0.82')  ('PF 1', '0.48')
('PF 9', '0.86')  ('DU 2', '0.78')  ('DU 6', '0.82')  ('PF 6', '0.47')
('DU 6', '0.86')  ('IB 4', '0.78')  ('PF 1', '0.82')  ('DU 7', '0.46')
('DU 2', '0.86')  ('DU 5', '0.77')  ('DU 2', '0.82')  ('DU 5', '0.46')
('RD 3', '0.86')  ('RD 9', '0.77')  ('DU 9', '0.82')  ('PF 7', '0.46')
('DU 8', '0.86')  ('PF 6', '0.77')  ('DU 3', '0.81')  ('DU 4', '0.46')
('DU 5', '0.86')  ('PF 9', '0.76')  ('PF 9', '0.81')  ('PF 3', '0.45')
('DU 9', '0.86')  ('DU 7', '0.76')  ('DU 5', '0.81')  ('DU 8', '0.44')
('RD 6', '0.85')  ('RD 2', '0.75')  ('PF 2', '0.81')  ('RD 9', '0.42')
('PF 6', '0.85')  ('RD 7', '0.75')  ('PF 5', '0.81')  ('PF 2', '0.42')
('RD 1', '0.84')  ('RD 8', '0.75')  ('PF 8', '0.80')  ('PF 8', '0.42')
('IB 1', '0.84')  ('DU 8', '0.75')  ('RD 1', '0.80')  ('RD 5', '0.42')
('PF 8', '0.84')  ('RD 1', '0.74')  ('RD 5', '0.80')  ('RD 4', '0.41')
('RD 7', '0.84')  ('DU 4', '0.74')  ('PF 7', '0.80')  ('RD 2', '0.41')
('RD 9', '0.84')  ('RD 6', '0.74')  ('RD 7', '0.80')  ('PF 5', '0.40')
('PF 5', '0.83')  ('IB 7', '0.74')  ('DU 7', '0.79')  ('RD 8', '0.40')
('RD 2', '0.83')  ('PF 8', '0.73')  ('PF 4', '0.79')  ('DU 9', '0.39')
('RD 8', '0.83')  ('PF 1', '0.72')  ('PF 6', '0.79')  ('RD 6', '0.39')
('PF 1', '0.83')  ('RD 5', '0.72')  ('DU 4', '0.78')  ('RD 3', '0.38')
('RD 5', '0.82')  ('DU 1', '0.70')  ('IB 1', '0.78')  ('PF 4', '0.37')
('DU 1', '0.81')  ('IB 1', '0.69')  ('RD 4', '0.77')  ('DU 1', '0.34')
('PF 4', '0.81')  ('PF 4', '0.68')  ('DU 8', '0.77')  ('RD 7', '0.33')
('RD 4', '0.80')  ('RD 4', '0.68')  ('DU 1', '0.76')  ('RD 1', '0.27')
```

Figure .116: The results from comparison to entry labeled "IB 5".



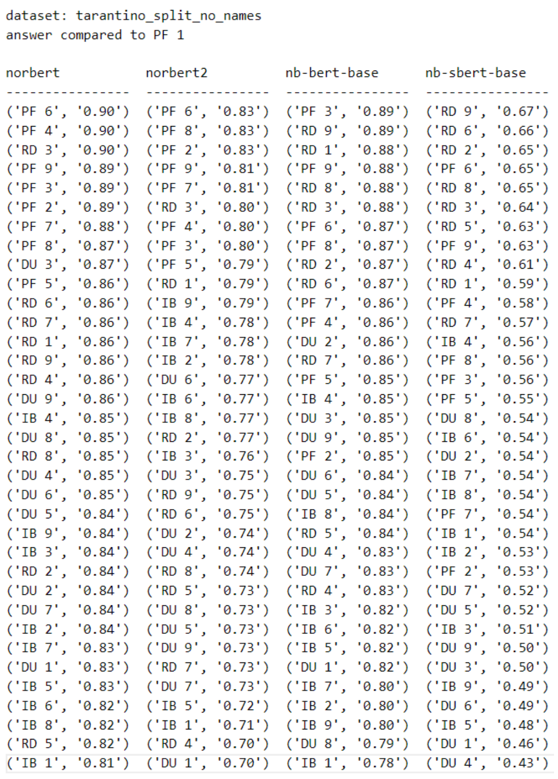
```
dataset: tarantino_split_no_names
answer compared to RD 3

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('RD 6', '0.96')  ('RD 6', '0.92')  ('RD 6', '0.94')  ('RD 6', '0.87')
('RD 7', '0.94')  ('RD 7', '0.90')  ('RD 7', '0.94')  ('RD 7', '0.82')
('RD 2', '0.94')  ('RD 8', '0.90')  ('RD 2', '0.93')  ('RD 8', '0.80')
('RD 8', '0.94')  ('RD 2', '0.90')  ('RD 8', '0.93')  ('RD 2', '0.79')
('RD 9', '0.92')  ('PF 5', '0.88')  ('RD 9', '0.92')  ('RD 9', '0.78')
('RD 1', '0.92')  ('PF 7', '0.87')  ('RD 5', '0.91')  ('RD 5', '0.77')
('PF 7', '0.91')  ('RD 9', '0.86')  ('PF 3', '0.89')  ('RD 4', '0.76')
('PF 9', '0.91')  ('PF 9', '0.86')  ('RD 1', '0.88')  ('RD 1', '0.64')
('PF 5', '0.91')  ('DU 3', '0.86')  ('PF 7', '0.88')  ('PF 1', '0.64')
('PF 3', '0.91')  ('DU 9', '0.86')  ('PF 1', '0.88')  ('PF 5', '0.57')
('DU 9', '0.90')  ('RD 1', '0.85')  ('PF 5', '0.88')  ('PF 8', '0.56')
('IB 4', '0.90')  ('DU 5', '0.85')  ('RD 4', '0.87')  ('PF 7', '0.56')
('PF 2', '0.90')  ('PF 6', '0.84')  ('PF 9', '0.87')  ('IB 4', '0.56')
('RD 5', '0.90')  ('RD 5', '0.84')  ('DU 9', '0.87')  ('PF 6', '0.54')
('PF 6', '0.90')  ('PF 2', '0.84')  ('DU 5', '0.87')  ('IB 7', '0.53')
('PF 1', '0.90')  ('IB 3', '0.83')  ('DU 2', '0.87')  ('PF 3', '0.52')
('RD 4', '0.89')  ('IB 8', '0.83')  ('DU 3', '0.86')  ('IB 1', '0.52')
('IB 3', '0.89')  ('DU 6', '0.83')  ('IB 8', '0.86')  ('DU 8', '0.51')
('DU 5', '0.89')  ('PF 3', '0.82')  ('DU 6', '0.86')  ('PF 9', '0.49')
('PF 8', '0.88')  ('DU 2', '0.82')  ('PF 6', '0.86')  ('IB 9', '0.48')
('DU 3', '0.88')  ('DU 1', '0.81')  ('PF 8', '0.86')  ('DU 2', '0.48')
('DU 1', '0.88')  ('IB 4', '0.81')  ('DU 7', '0.85')  ('PF 2', '0.47')
('DU 6', '0.87')  ('DU 8', '0.81')  ('IB 4', '0.84')  ('IB 2', '0.47')
('DU 8', '0.87')  ('IB 2', '0.80')  ('IB 3', '0.84')  ('IB 8', '0.47')
('IB 9', '0.87')  ('PF 1', '0.80')  ('PF 4', '0.84')  ('PF 4', '0.46')
('IB 2', '0.87')  ('DU 7', '0.80')  ('DU 1', '0.83')  ('DU 5', '0.45')
('IB 8', '0.87')  ('IB 9', '0.79')  ('PF 2', '0.83')  ('DU 4', '0.45')
('IB 5', '0.86')  ('PF 8', '0.79')  ('IB 5', '0.82')  ('IB 6', '0.44')
('DU 2', '0.86')  ('IB 5', '0.79')  ('IB 2', '0.82')  ('DU 9', '0.43')
('PF 4', '0.85')  ('IB 6', '0.79')  ('DU 4', '0.82')  ('DU 6', '0.43')
('DU 7', '0.85')  ('RD 4', '0.78')  ('IB 7', '0.81')  ('DU 1', '0.43')
('IB 7', '0.85')  ('IB 7', '0.77')  ('DU 8', '0.81')  ('DU 7', '0.42')
('IB 6', '0.84')  ('DU 4', '0.76')  ('IB 6', '0.80')  ('DU 3', '0.41')
('DU 4', '0.84')  ('PF 4', '0.73')  ('IB 9', '0.79')  ('IB 3', '0.41')
('IB 1', '0.82')  ('IB 1', '0.69')  ('IB 1', '0.75')  ('IB 5', '0.38')
```

Figure .117: The results from comparison to entry labeled "RD 3".



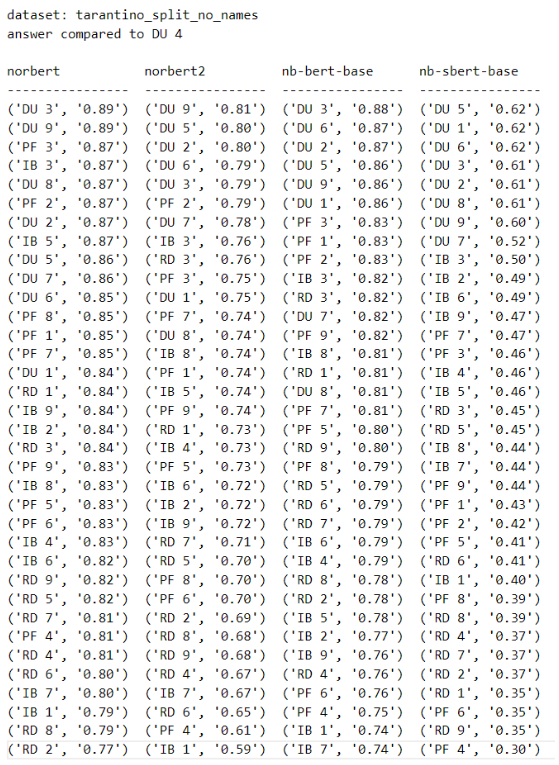
```
dataset: tarantino_split_no_names
answer compared to IB 6

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('IB 2', '0.94')   ('IB 2', '0.92')   ('IB 2', '0.92')   ('IB 2', '0.91')
('IB 3', '0.92')   ('IB 5', '0.89')   ('IB 9', '0.90')   ('IB 9', '0.88')
('IB 5', '0.92')   ('IB 3', '0.87')   ('IB 5', '0.89')   ('IB 5', '0.88')
('IB 8', '0.91')   ('IB 8', '0.87')   ('IB 8', '0.89')   ('IB 8', '0.82')
('IB 9', '0.90')   ('IB 9', '0.86')   ('IB 3', '0.87')   ('IB 3', '0.79')
('IB 7', '0.90')   ('PF 6', '0.80')   ('IB 7', '0.87')   ('IB 4', '0.79')
('DU 6', '0.87')   ('PF 7', '0.80')   ('IB 4', '0.83')   ('IB 7', '0.79')
('IB 4', '0.87')   ('IB 4', '0.80')   ('PF 3', '0.82')   ('IB 1', '0.62')
('DU 2', '0.87')   ('DU 6', '0.80')   ('PF 1', '0.82')   ('PF 6', '0.56')
('PF 3', '0.86')   ('PF 2', '0.79')   ('DU 6', '0.82')   ('PF 1', '0.54')
('DU 7', '0.86')   ('PF 8', '0.79')   ('RD 9', '0.81')   ('PF 9', '0.52')
('PF 8', '0.85')   ('PF 5', '0.79')   ('DU 2', '0.81')   ('PF 7', '0.51')
('PF 9', '0.85')   ('PF 3', '0.79')   ('PF 2', '0.81')   ('DU 6', '0.51')
('PF 7', '0.85')   ('RD 3', '0.79')   ('PF 9', '0.81')   ('DU 7', '0.50')
('DU 3', '0.85')   ('IB 7', '0.78')   ('DU 3', '0.81')   ('PF 8', '0.49')
('PF 2', '0.85')   ('DU 3', '0.78')   ('DU 5', '0.80')   ('DU 2', '0.49')
('RD 3', '0.84')   ('RD 9', '0.78')   ('PF 7', '0.80')   ('DU 4', '0.49')
('DU 8', '0.84')   ('DU 9', '0.78')   ('DU 9', '0.80')   ('DU 3', '0.48')
('DU 5', '0.84')   ('PF 9', '0.78')   ('RD 3', '0.80')   ('PF 5', '0.48')
('PF 6', '0.84')   ('PF 1', '0.77')   ('RD 8', '0.80')   ('RD 9', '0.48')
('RD 1', '0.83')   ('DU 2', '0.77')   ('RD 2', '0.80')   ('PF 3', '0.48')
('PF 5', '0.83')   ('RD 1', '0.77')   ('PF 5', '0.80')   ('PF 4', '0.47')
('PF 1', '0.82')   ('DU 8', '0.76')   ('PF 6', '0.80')   ('DU 8', '0.46')
('RD 5', '0.82')   ('DU 7', '0.74')   ('PF 8', '0.79')   ('RD 5', '0.45')
('DU 4', '0.82')   ('RD 2', '0.74')   ('RD 6', '0.79')   ('DU 5', '0.45')
('IB 1', '0.82')   ('DU 5', '0.74')   ('DU 7', '0.79')   ('RD 3', '0.44')
('RD 9', '0.82')   ('RD 7', '0.74')   ('DU 4', '0.79')   ('PF 2', '0.44')
('RD 6', '0.82')   ('RD 5', '0.74')   ('RD 1', '0.79')   ('RD 6', '0.43')
('RD 8', '0.81')   ('RD 6', '0.74')   ('PF 4', '0.78')   ('RD 8', '0.42')
('RD 2', '0.81')   ('DU 4', '0.72')   ('RD 5', '0.78')   ('RD 2', '0.42')
('RD 7', '0.81')   ('RD 8', '0.72')   ('RD 7', '0.77')   ('RD 4', '0.41')
('DU 1', '0.81')   ('PF 4', '0.70')   ('DU 1', '0.77')   ('DU 9', '0.40')
('DU 9', '0.80')   ('DU 1', '0.69')   ('DU 8', '0.76')   ('RD 7', '0.36')
('PF 4', '0.80')   ('IB 1', '0.68')   ('RD 4', '0.75')   ('DU 1', '0.34')
('RD 4', '0.77')   ('RD 4', '0.66')   ('IB 1', '0.73')   ('RD 1', '0.33')
```

Figure .118: The results from comparison to entry labeled "IB 6".



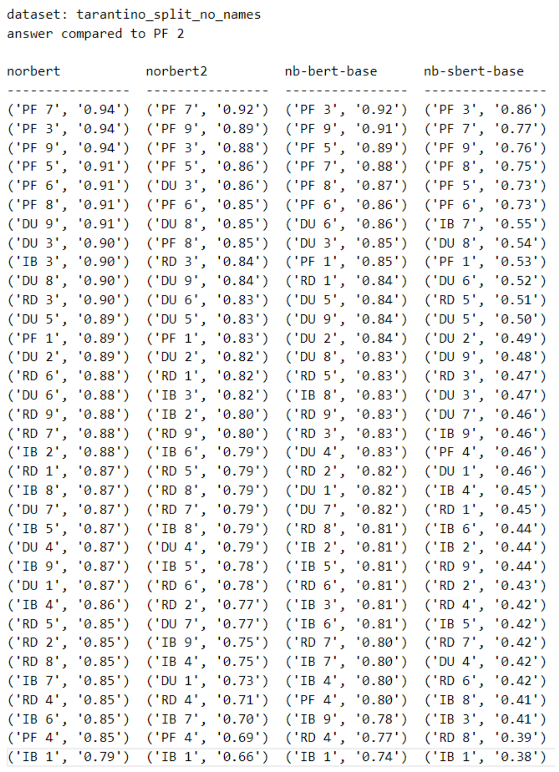
```
dataset: tarantino_split_no_names
answer compared to PF 1

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('PF 6', '0.90')   ('PF 6', '0.83')   ('PF 3', '0.89')   ('RD 9', '0.67')
('PF 4', '0.90')   ('PF 8', '0.83')   ('RD 9', '0.89')   ('RD 6', '0.66')
('RD 3', '0.90')   ('PF 2', '0.83')   ('RD 1', '0.88')   ('RD 2', '0.65')
('PF 9', '0.89')   ('PF 9', '0.81')   ('PF 9', '0.88')   ('PF 6', '0.65')
('PF 3', '0.89')   ('PF 7', '0.81')   ('RD 8', '0.88')   ('RD 8', '0.65')
('PF 2', '0.89')   ('RD 3', '0.80')   ('RD 3', '0.88')   ('RD 3', '0.64')
('PF 7', '0.88')   ('PF 4', '0.80')   ('PF 6', '0.87')   ('RD 5', '0.63')
('PF 8', '0.87')   ('PF 3', '0.80')   ('PF 8', '0.87')   ('PF 9', '0.63')
('DU 3', '0.87')   ('PF 5', '0.79')   ('RD 2', '0.87')   ('RD 4', '0.61')
('PF 5', '0.86')   ('RD 1', '0.79')   ('RD 6', '0.87')   ('RD 1', '0.59')
('RD 6', '0.86')   ('IB 9', '0.79')   ('PF 7', '0.86')   ('PF 4', '0.58')
('RD 7', '0.86')   ('IB 4', '0.78')   ('PF 4', '0.86')   ('RD 7', '0.57')
('RD 1', '0.86')   ('IB 7', '0.78')   ('DU 2', '0.86')   ('IB 4', '0.56')
('RD 9', '0.86')   ('IB 2', '0.78')   ('RD 7', '0.86')   ('PF 8', '0.56')
('RD 4', '0.86')   ('DU 6', '0.77')   ('PF 5', '0.85')   ('PF 3', '0.56')
('DU 9', '0.86')   ('IB 6', '0.77')   ('IB 4', '0.85')   ('PF 5', '0.55')
('IB 4', '0.85')   ('IB 8', '0.77')   ('DU 3', '0.85')   ('DU 8', '0.54')
('DU 8', '0.85')   ('RD 2', '0.77')   ('DU 9', '0.85')   ('IB 6', '0.54')
('RD 8', '0.85')   ('IB 3', '0.76')   ('PF 2', '0.85')   ('DU 2', '0.54')
('DU 4', '0.85')   ('DU 3', '0.75')   ('DU 6', '0.84')   ('IB 7', '0.54')
('DU 6', '0.85')   ('RD 9', '0.75')   ('DU 5', '0.84')   ('IB 8', '0.54')
('DU 5', '0.84')   ('RD 6', '0.75')   ('IB 8', '0.84')   ('PF 7', '0.54')
('IB 9', '0.84')   ('DU 2', '0.74')   ('RD 5', '0.84')   ('IB 1', '0.54')
('IB 3', '0.84')   ('DU 4', '0.74')   ('DU 4', '0.83')   ('IB 2', '0.53')
('RD 2', '0.84')   ('RD 8', '0.74')   ('DU 7', '0.83')   ('PF 2', '0.53')
('DU 2', '0.84')   ('RD 5', '0.73')   ('RD 4', '0.83')   ('DU 7', '0.52')
('DU 7', '0.84')   ('DU 8', '0.73')   ('IB 3', '0.82')   ('DU 5', '0.52')
('IB 2', '0.84')   ('DU 5', '0.73')   ('IB 6', '0.82')   ('IB 3', '0.51')
('IB 7', '0.83')   ('DU 9', '0.73')   ('IB 5', '0.82')   ('DU 9', '0.50')
('DU 1', '0.83')   ('RD 7', '0.73')   ('DU 1', '0.82')   ('DU 3', '0.50')
('IB 5', '0.83')   ('DU 7', '0.73')   ('IB 7', '0.80')   ('IB 9', '0.49')
('IB 6', '0.82')   ('IB 5', '0.72')   ('IB 2', '0.80')   ('DU 6', '0.49')
('IB 8', '0.82')   ('IB 1', '0.71')   ('IB 9', '0.80')   ('IB 5', '0.48')
('RD 5', '0.82')   ('RD 4', '0.70')   ('DU 8', '0.79')   ('DU 1', '0.46')
('IB 1', '0.81')   ('DU 1', '0.70')   ('IB 1', '0.78')   ('DU 4', '0.43')
```

Figure .119: The results from comparison to entry labeled "PF 1".



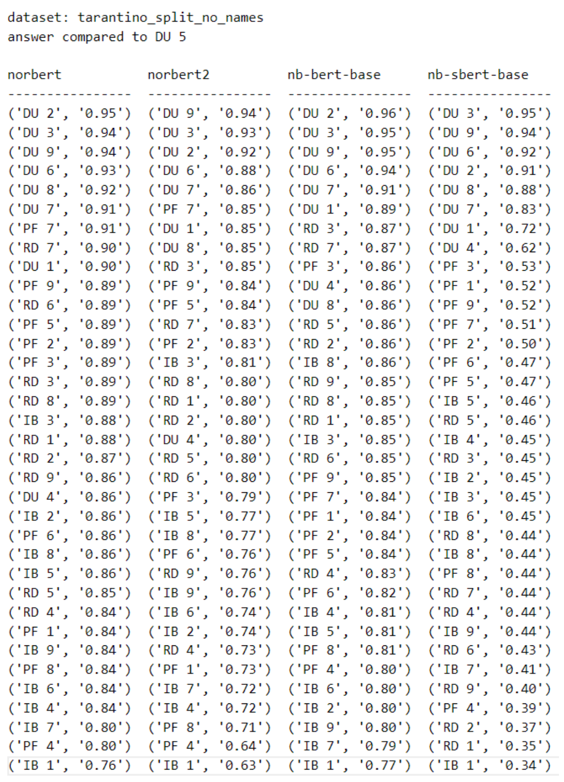
```
dataset: tarantino_split_no_names
answer compared to DU 4

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('DU 3', '0.89')   ('DU 9', '0.81')   ('DU 3', '0.88')   ('DU 5', '0.62')
('DU 9', '0.89')   ('DU 5', '0.80')   ('DU 6', '0.87')   ('DU 1', '0.62')
('PF 3', '0.87')   ('DU 2', '0.80')   ('DU 2', '0.87')   ('DU 6', '0.62')
('IB 3', '0.87')   ('DU 6', '0.79')   ('DU 5', '0.86')   ('DU 3', '0.61')
('DU 8', '0.87')   ('DU 3', '0.79')   ('DU 9', '0.86')   ('DU 2', '0.61')
('PF 2', '0.87')   ('PF 2', '0.79')   ('DU 1', '0.86')   ('DU 8', '0.61')
('DU 2', '0.87')   ('DU 7', '0.78')   ('PF 3', '0.83')   ('DU 9', '0.60')
('IB 5', '0.87')   ('IB 3', '0.76')   ('PF 1', '0.83')   ('DU 7', '0.52')
('DU 5', '0.86')   ('RD 3', '0.76')   ('PF 2', '0.83')   ('IB 3', '0.50')
('DU 7', '0.86')   ('PF 3', '0.75')   ('IB 3', '0.82')   ('IB 2', '0.49')
('DU 6', '0.85')   ('DU 1', '0.75')   ('RD 3', '0.82')   ('IB 6', '0.49')
('PF 8', '0.85')   ('PF 7', '0.74')   ('DU 7', '0.82')   ('IB 9', '0.47')
('PF 1', '0.85')   ('DU 8', '0.74')   ('PF 9', '0.82')   ('PF 7', '0.47')
('PF 7', '0.85')   ('IB 8', '0.74')   ('IB 8', '0.81')   ('PF 3', '0.46')
('DU 1', '0.84')   ('PF 1', '0.74')   ('RD 1', '0.81')   ('IB 4', '0.46')
('RD 1', '0.84')   ('IB 5', '0.74')   ('DU 8', '0.81')   ('IB 5', '0.46')
('IB 9', '0.84')   ('PF 9', '0.74')   ('PF 7', '0.81')   ('RD 3', '0.45')
('IB 2', '0.84')   ('RD 1', '0.73')   ('PF 5', '0.80')   ('RD 5', '0.45')
('RD 3', '0.84')   ('IB 4', '0.73')   ('RD 9', '0.80')   ('IB 8', '0.44')
('PF 9', '0.83')   ('PF 5', '0.73')   ('PF 8', '0.79')   ('IB 7', '0.44')
('IB 8', '0.83')   ('IB 6', '0.72')   ('RD 5', '0.79')   ('PF 9', '0.44')
('PF 5', '0.83')   ('IB 2', '0.72')   ('RD 6', '0.79')   ('PF 1', '0.43')
('PF 6', '0.83')   ('IB 9', '0.72')   ('RD 7', '0.79')   ('PF 2', '0.42')
('IB 4', '0.83')   ('RD 7', '0.71')   ('IB 6', '0.79')   ('PF 5', '0.41')
('IB 6', '0.82')   ('RD 5', '0.70')   ('IB 4', '0.79')   ('RD 6', '0.41')
('RD 9', '0.82')   ('PF 8', '0.70')   ('RD 8', '0.78')   ('IB 1', '0.40')
('RD 5', '0.82')   ('PF 6', '0.70')   ('RD 2', '0.78')   ('PF 8', '0.39')
('RD 7', '0.81')   ('RD 2', '0.69')   ('IB 5', '0.78')   ('RD 8', '0.39')
('PF 4', '0.81')   ('RD 8', '0.68')   ('IB 2', '0.77')   ('RD 4', '0.37')
('RD 4', '0.81')   ('RD 9', '0.68')   ('IB 9', '0.76')   ('RD 7', '0.37')
('RD 6', '0.80')   ('RD 4', '0.67')   ('RD 4', '0.76')   ('RD 2', '0.37')
('IB 7', '0.80')   ('IB 7', '0.67')   ('PF 6', '0.76')   ('RD 1', '0.35')
('IB 1', '0.79')   ('RD 6', '0.65')   ('PF 4', '0.75')   ('PF 6', '0.35')
('RD 8', '0.79')   ('PF 4', '0.61')   ('IB 1', '0.74')   ('RD 9', '0.35')
('RD 2', '0.77')   ('IB 1', '0.59')   ('IB 7', '0.74')   ('PF 4', '0.30')
```

Figure .120: The results from comparison to entry labeled "DU 4".



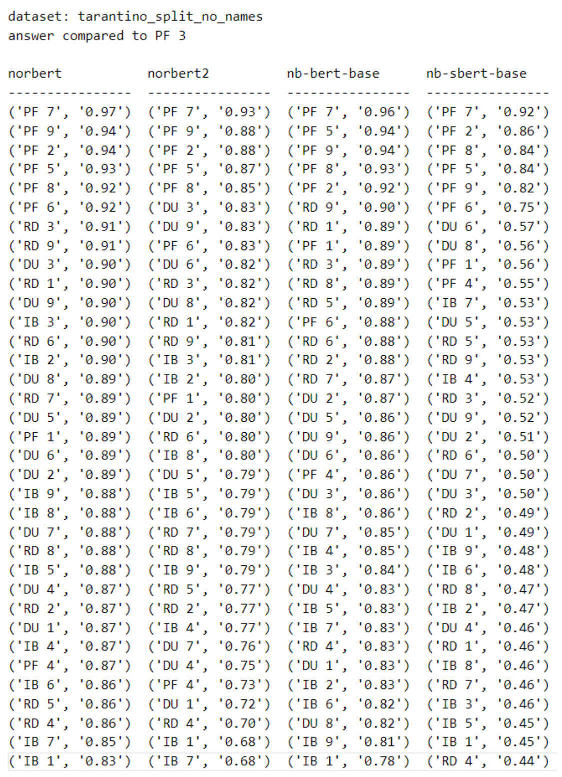
```
dataset: tarantino_split_no_names
answer compared to PF 2

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('PF 7', '0.94')   ('PF 7', '0.92')   ('PF 3', '0.92')   ('PF 3', '0.86')
('PF 3', '0.94')   ('PF 9', '0.89')   ('PF 9', '0.91')   ('PF 7', '0.77')
('PF 9', '0.94')   ('PF 3', '0.88')   ('PF 5', '0.89')   ('PF 9', '0.76')
('PF 5', '0.91')   ('PF 5', '0.86')   ('PF 7', '0.88')   ('PF 8', '0.75')
('PF 6', '0.91')   ('DU 3', '0.86')   ('PF 8', '0.87')   ('PF 5', '0.73')
('PF 8', '0.91')   ('PF 6', '0.85')   ('PF 6', '0.86')   ('PF 6', '0.73')
('DU 9', '0.91')   ('DU 8', '0.85')   ('DU 6', '0.86')   ('IB 7', '0.55')
('DU 3', '0.90')   ('PF 8', '0.85')   ('DU 3', '0.85')   ('DU 8', '0.54')
('IB 3', '0.90')   ('RD 3', '0.84')   ('PF 1', '0.85')   ('PF 1', '0.53')
('DU 8', '0.90')   ('DU 9', '0.84')   ('RD 1', '0.84')   ('DU 6', '0.52')
('RD 3', '0.90')   ('DU 6', '0.83')   ('DU 5', '0.84')   ('RD 5', '0.51')
('DU 5', '0.89')   ('DU 5', '0.83')   ('DU 9', '0.84')   ('DU 5', '0.50')
('PF 1', '0.89')   ('PF 1', '0.83')   ('DU 2', '0.84')   ('DU 2', '0.49')
('DU 2', '0.89')   ('DU 2', '0.82')   ('DU 8', '0.83')   ('DU 9', '0.48')
('RD 6', '0.88')   ('RD 1', '0.82')   ('RD 5', '0.83')   ('RD 3', '0.47')
('DU 6', '0.88')   ('IB 3', '0.82')   ('IB 8', '0.83')   ('DU 3', '0.47')
('RD 9', '0.88')   ('IB 2', '0.80')   ('RD 9', '0.83')   ('DU 7', '0.46')
('RD 7', '0.88')   ('RD 9', '0.80')   ('RD 3', '0.83')   ('IB 9', '0.46')
('IB 2', '0.88')   ('IB 6', '0.79')   ('DU 4', '0.83')   ('PF 4', '0.46')
('RD 1', '0.87')   ('RD 5', '0.79')   ('RD 2', '0.82')   ('DU 1', '0.46')
('IB 8', '0.87')   ('RD 8', '0.79')   ('DU 1', '0.82')   ('IB 4', '0.45')
('DU 7', '0.87')   ('RD 7', '0.79')   ('DU 7', '0.82')   ('RD 1', '0.45')
('IB 5', '0.87')   ('IB 8', '0.79')   ('RD 8', '0.81')   ('IB 6', '0.44')
('DU 4', '0.87')   ('DU 4', '0.79')   ('IB 2', '0.81')   ('IB 2', '0.44')
('IB 9', '0.87')   ('IB 5', '0.78')   ('IB 5', '0.81')   ('RD 9', '0.44')
('DU 1', '0.87')   ('RD 6', '0.78')   ('RD 6', '0.81')   ('RD 2', '0.43')
('IB 4', '0.86')   ('RD 2', '0.77')   ('IB 3', '0.81')   ('RD 4', '0.42')
('RD 5', '0.85')   ('DU 7', '0.77')   ('IB 6', '0.81')   ('IB 5', '0.42')
('RD 2', '0.85')   ('IB 9', '0.75')   ('RD 7', '0.80')   ('RD 7', '0.42')
('RD 8', '0.85')   ('IB 4', '0.75')   ('IB 7', '0.80')   ('DU 4', '0.42')
('IB 7', '0.85')   ('DU 1', '0.73')   ('IB 4', '0.80')   ('RD 6', '0.42')
('RD 4', '0.85')   ('RD 4', '0.71')   ('PF 4', '0.80')   ('IB 8', '0.41')
('IB 6', '0.85')   ('IB 7', '0.70')   ('IB 9', '0.78')   ('IB 3', '0.41')
('PF 4', '0.85')   ('PF 4', '0.69')   ('RD 4', '0.77')   ('RD 8', '0.39')
('IB 1', '0.79')   ('IB 1', '0.66')   ('IB 1', '0.74')   ('IB 1', '0.38')
```

Figure .121: The results from comparison to entry labeled "PF 2".



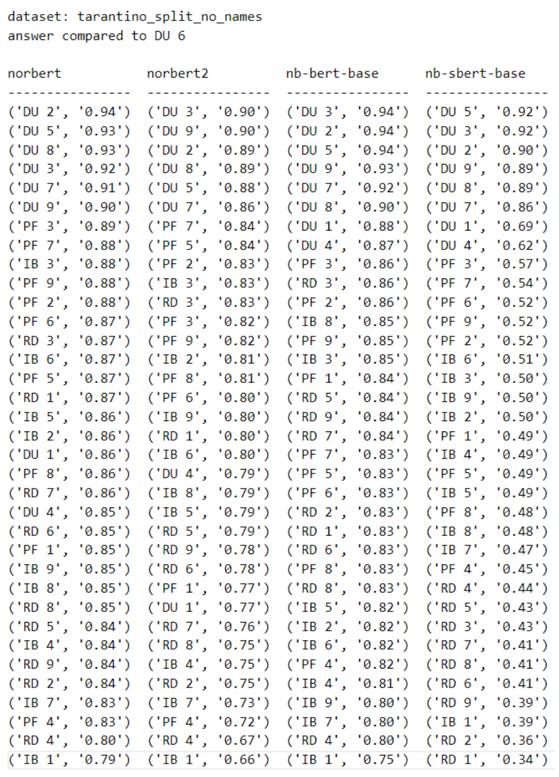
```
dataset: tarantino_split_no_names
answer compared to DU 5

norbert            norbert2           nb-bert-base       nb-sbert-base
---------------    ---------------    ---------------    ---------------
('DU 2', '0.95')   ('DU 9', '0.94')   ('DU 2', '0.96')   ('DU 3', '0.95')
('DU 3', '0.94')   ('DU 3', '0.93')   ('DU 3', '0.95')   ('DU 9', '0.94')
('DU 9', '0.94')   ('DU 2', '0.92')   ('DU 9', '0.95')   ('DU 6', '0.92')
('DU 6', '0.93')   ('DU 6', '0.88')   ('DU 6', '0.94')   ('DU 2', '0.91')
('DU 8', '0.92')   ('DU 7', '0.86')   ('DU 7', '0.91')   ('DU 8', '0.88')
('DU 7', '0.91')   ('PF 7', '0.85')   ('DU 1', '0.89')   ('DU 7', '0.83')
('PF 7', '0.91')   ('DU 1', '0.85')   ('RD 3', '0.87')   ('DU 1', '0.72')
('RD 7', '0.90')   ('DU 8', '0.85')   ('RD 7', '0.87')   ('DU 4', '0.62')
('DU 1', '0.90')   ('RD 3', '0.85')   ('PF 3', '0.86')   ('PF 3', '0.53')
('PF 9', '0.89')   ('PF 9', '0.84')   ('DU 4', '0.86')   ('PF 1', '0.52')
('RD 6', '0.89')   ('PF 5', '0.84')   ('DU 8', '0.86')   ('PF 9', '0.52')
('PF 5', '0.89')   ('RD 7', '0.83')   ('RD 5', '0.86')   ('PF 7', '0.51')
('PF 2', '0.89')   ('PF 2', '0.83')   ('RD 2', '0.86')   ('PF 2', '0.50')
('PF 3', '0.89')   ('IB 3', '0.81')   ('IB 8', '0.86')   ('PF 6', '0.47')
('RD 3', '0.89')   ('RD 8', '0.80')   ('RD 9', '0.85')   ('PF 5', '0.47')
('RD 8', '0.89')   ('RD 1', '0.80')   ('RD 8', '0.85')   ('IB 5', '0.46')
('IB 3', '0.88')   ('RD 2', '0.80')   ('RD 1', '0.85')   ('RD 5', '0.46')
('RD 1', '0.88')   ('DU 4', '0.80')   ('IB 3', '0.85')   ('IB 4', '0.45')
('RD 2', '0.87')   ('RD 5', '0.80')   ('RD 6', '0.85')   ('RD 3', '0.45')
('RD 9', '0.86')   ('RD 6', '0.80')   ('PF 9', '0.85')   ('IB 2', '0.45')
('DU 4', '0.86')   ('PF 3', '0.79')   ('PF 7', '0.84')   ('IB 3', '0.45')
('IB 2', '0.86')   ('IB 5', '0.77')   ('PF 1', '0.84')   ('IB 6', '0.45')
('PF 6', '0.86')   ('IB 8', '0.77')   ('PF 2', '0.84')   ('RD 8', '0.44')
('IB 8', '0.86')   ('PF 6', '0.76')   ('PF 5', '0.84')   ('IB 8', '0.44')
('IB 5', '0.86')   ('RD 9', '0.76')   ('RD 4', '0.83')   ('PF 8', '0.44')
('RD 5', '0.85')   ('IB 9', '0.76')   ('PF 6', '0.82')   ('RD 7', '0.44')
('RD 4', '0.84')   ('IB 6', '0.74')   ('IB 4', '0.81')   ('RD 4', '0.44')
('PF 1', '0.84')   ('IB 2', '0.74')   ('IB 5', '0.81')   ('IB 9', '0.44')
('IB 9', '0.84')   ('RD 4', '0.73')   ('PF 8', '0.81')   ('RD 6', '0.43')
('PF 8', '0.84')   ('PF 1', '0.73')   ('PF 4', '0.80')   ('IB 7', '0.41')
('IB 6', '0.84')   ('IB 7', '0.72')   ('IB 6', '0.80')   ('RD 9', '0.40')
('IB 4', '0.84')   ('IB 4', '0.72')   ('IB 2', '0.80')   ('PF 4', '0.39')
('IB 7', '0.80')   ('PF 8', '0.71')   ('IB 9', '0.80')   ('RD 2', '0.37')
('PF 4', '0.80')   ('PF 4', '0.64')   ('IB 7', '0.79')   ('RD 1', '0.35')
('IB 1', '0.76')   ('IB 1', '0.63')   ('IB 1', '0.77')   ('IB 1', '0.34')
```

Figure .122: The results from comparison to entry labeled "DU 5".



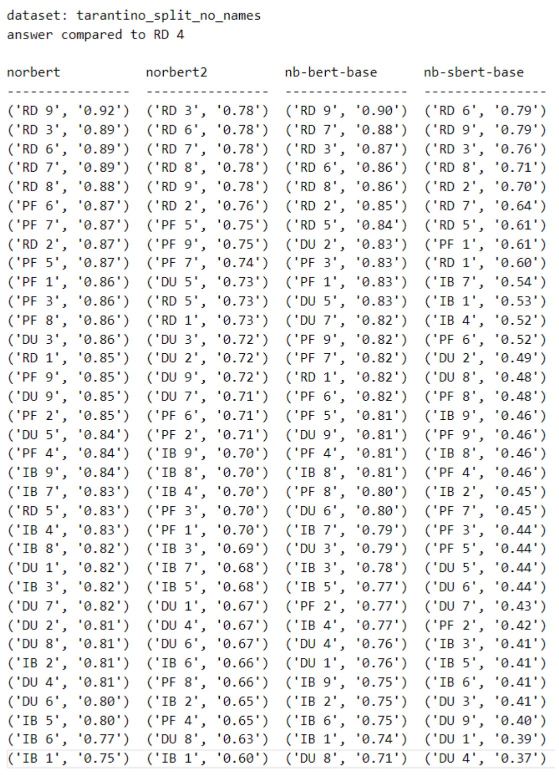
```
dataset: tarantino_split_no_names
answer compared to PF 3

norbert            norbert2           nb-bert-base       nb-sbert-base
---------------    ---------------    ---------------    ---------------
('PF 7', '0.97')   ('PF 7', '0.93')   ('PF 7', '0.96')   ('PF 7', '0.92')
('PF 9', '0.94')   ('PF 9', '0.88')   ('PF 5', '0.94')   ('PF 2', '0.86')
('PF 2', '0.94')   ('PF 2', '0.88')   ('PF 9', '0.94')   ('PF 8', '0.84')
('PF 5', '0.93')   ('PF 5', '0.87')   ('PF 8', '0.93')   ('PF 5', '0.84')
('PF 8', '0.92')   ('PF 8', '0.85')   ('PF 2', '0.92')   ('PF 9', '0.82')
('PF 6', '0.92')   ('DU 3', '0.83')   ('RD 9', '0.90')   ('PF 6', '0.75')
('RD 3', '0.91')   ('DU 9', '0.83')   ('RD 1', '0.89')   ('DU 6', '0.57')
('RD 9', '0.91')   ('PF 6', '0.83')   ('PF 1', '0.89')   ('DU 8', '0.56')
('DU 3', '0.90')   ('DU 6', '0.82')   ('RD 3', '0.89')   ('PF 1', '0.56')
('RD 1', '0.90')   ('RD 3', '0.82')   ('RD 8', '0.89')   ('PF 4', '0.55')
('DU 9', '0.90')   ('DU 8', '0.82')   ('RD 5', '0.89')   ('IB 7', '0.53')
('IB 3', '0.90')   ('RD 1', '0.82')   ('PF 6', '0.88')   ('DU 5', '0.53')
('RD 6', '0.90')   ('RD 9', '0.81')   ('RD 6', '0.88')   ('RD 5', '0.53')
('IB 2', '0.90')   ('IB 3', '0.81')   ('RD 2', '0.88')   ('RD 9', '0.53')
('DU 8', '0.89')   ('IB 2', '0.80')   ('RD 7', '0.87')   ('IB 4', '0.53')
('RD 7', '0.89')   ('PF 1', '0.80')   ('DU 2', '0.87')   ('RD 3', '0.52')
('DU 5', '0.89')   ('DU 2', '0.80')   ('DU 5', '0.86')   ('DU 9', '0.52')
('PF 1', '0.89')   ('RD 6', '0.80')   ('DU 9', '0.86')   ('DU 2', '0.51')
('DU 6', '0.89')   ('IB 8', '0.80')   ('DU 6', '0.86')   ('RD 6', '0.50')
('DU 2', '0.89')   ('DU 5', '0.79')   ('PF 4', '0.86')   ('DU 7', '0.50')
('IB 9', '0.88')   ('IB 5', '0.79')   ('DU 3', '0.86')   ('DU 3', '0.50')
('IB 8', '0.88')   ('IB 6', '0.79')   ('IB 8', '0.86')   ('RD 2', '0.49')
('DU 7', '0.88')   ('RD 7', '0.79')   ('DU 7', '0.85')   ('DU 1', '0.49')
('RD 8', '0.88')   ('RD 8', '0.79')   ('IB 4', '0.85')   ('IB 9', '0.48')
('IB 5', '0.88')   ('IB 9', '0.79')   ('IB 3', '0.84')   ('IB 6', '0.48')
('DU 4', '0.87')   ('RD 5', '0.77')   ('DU 4', '0.83')   ('RD 8', '0.47')
('RD 2', '0.87')   ('RD 2', '0.77')   ('IB 5', '0.83')   ('IB 2', '0.47')
('DU 1', '0.87')   ('IB 4', '0.77')   ('IB 7', '0.83')   ('DU 4', '0.46')
('IB 4', '0.87')   ('DU 7', '0.76')   ('RD 4', '0.83')   ('RD 1', '0.46')
('PF 4', '0.87')   ('DU 4', '0.75')   ('DU 1', '0.83')   ('IB 8', '0.46')
('IB 6', '0.86')   ('PF 4', '0.73')   ('IB 2', '0.83')   ('RD 7', '0.46')
('RD 5', '0.86')   ('DU 1', '0.72')   ('IB 6', '0.82')   ('IB 3', '0.46')
('RD 4', '0.86')   ('RD 4', '0.70')   ('DU 8', '0.82')   ('IB 5', '0.45')
('IB 7', '0.85')   ('IB 1', '0.68')   ('IB 9', '0.81')   ('IB 1', '0.45')
('IB 1', '0.83')   ('IB 7', '0.68')   ('IB 1', '0.78')   ('RD 4', '0.44')
```

Figure .123: The results from comparison to entry labeled "PF 3".



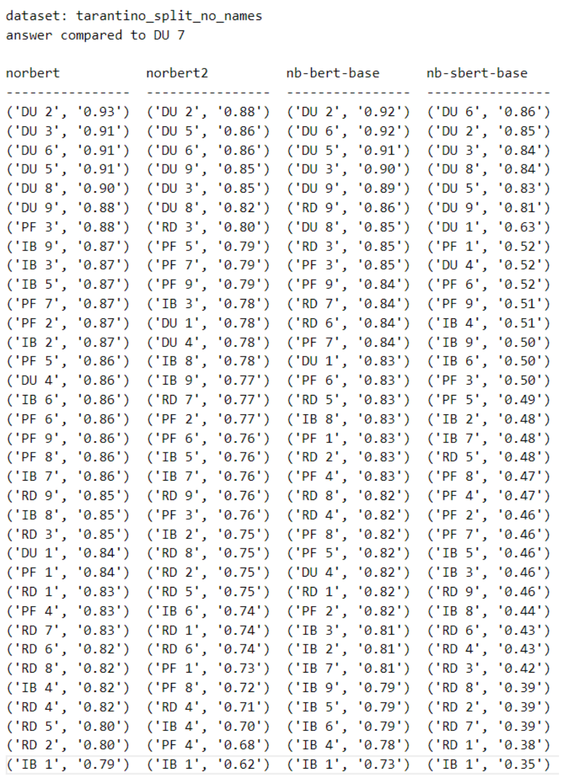
```
dataset: tarantino_split_no_names
answer compared to DU 6

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('DU 2', '0.94')  ('DU 3', '0.90')  ('DU 3', '0.94')  ('DU 5', '0.92')
('DU 5', '0.93')  ('DU 9', '0.90')  ('DU 2', '0.94')  ('DU 3', '0.92')
('DU 8', '0.93')  ('DU 2', '0.89')  ('DU 5', '0.94')  ('DU 2', '0.90')
('DU 3', '0.92')  ('DU 8', '0.89')  ('DU 9', '0.93')  ('DU 9', '0.89')
('DU 7', '0.91')  ('DU 5', '0.88')  ('DU 7', '0.92')  ('DU 8', '0.89')
('DU 9', '0.90')  ('DU 7', '0.86')  ('DU 8', '0.90')  ('DU 7', '0.86')
('PF 3', '0.89')  ('PF 7', '0.84')  ('DU 1', '0.88')  ('DU 1', '0.69')
('PF 7', '0.88')  ('PF 5', '0.84')  ('DU 4', '0.87')  ('DU 4', '0.62')
('IB 3', '0.88')  ('PF 2', '0.83')  ('PF 3', '0.86')  ('PF 3', '0.57')
('PF 9', '0.88')  ('IB 3', '0.83')  ('RD 3', '0.86')  ('PF 7', '0.54')
('PF 2', '0.88')  ('RD 3', '0.83')  ('PF 2', '0.86')  ('PF 6', '0.52')
('PF 6', '0.87')  ('PF 3', '0.82')  ('IB 8', '0.85')  ('PF 9', '0.52')
('RD 3', '0.87')  ('PF 9', '0.82')  ('PF 9', '0.85')  ('PF 2', '0.52')
('IB 6', '0.87')  ('IB 2', '0.81')  ('IB 3', '0.85')  ('IB 6', '0.51')
('PF 5', '0.87')  ('PF 8', '0.81')  ('PF 1', '0.84')  ('IB 3', '0.50')
('RD 1', '0.87')  ('PF 6', '0.80')  ('RD 5', '0.84')  ('IB 9', '0.50')
('IB 5', '0.86')  ('IB 9', '0.80')  ('RD 9', '0.84')  ('IB 2', '0.50')
('IB 2', '0.86')  ('RD 1', '0.80')  ('RD 7', '0.84')  ('PF 1', '0.49')
('DU 1', '0.86')  ('IB 6', '0.80')  ('PF 7', '0.83')  ('IB 4', '0.49')
('PF 8', '0.86')  ('DU 4', '0.79')  ('PF 5', '0.83')  ('PF 5', '0.49')
('RD 7', '0.86')  ('IB 8', '0.79')  ('PF 6', '0.83')  ('IB 5', '0.49')
('DU 4', '0.85')  ('IB 5', '0.79')  ('RD 2', '0.83')  ('PF 8', '0.48')
('RD 6', '0.85')  ('RD 5', '0.79')  ('RD 1', '0.83')  ('IB 8', '0.48')
('PF 1', '0.85')  ('RD 9', '0.78')  ('RD 6', '0.83')  ('IB 7', '0.47')
('IB 9', '0.85')  ('RD 6', '0.78')  ('PF 8', '0.83')  ('PF 4', '0.45')
('IB 8', '0.85')  ('PF 1', '0.77')  ('RD 8', '0.83')  ('RD 4', '0.44')
('RD 8', '0.85')  ('DU 1', '0.77')  ('IB 5', '0.82')  ('RD 5', '0.43')
('RD 5', '0.84')  ('RD 7', '0.76')  ('IB 2', '0.82')  ('RD 3', '0.43')
('IB 4', '0.84')  ('RD 8', '0.75')  ('IB 6', '0.82')  ('RD 7', '0.41')
('RD 9', '0.84')  ('IB 4', '0.75')  ('PF 4', '0.82')  ('RD 8', '0.41')
('RD 2', '0.84')  ('RD 2', '0.75')  ('IB 4', '0.81')  ('RD 6', '0.41')
('IB 7', '0.83')  ('IB 7', '0.73')  ('IB 9', '0.80')  ('RD 9', '0.39')
('PF 4', '0.83')  ('PF 4', '0.72')  ('IB 7', '0.80')  ('IB 1', '0.39')
('RD 4', '0.80')  ('RD 4', '0.67')  ('RD 4', '0.80')  ('RD 2', '0.36')
('IB 1', '0.79')  ('IB 1', '0.66')  ('IB 1', '0.75')  ('RD 1', '0.34')
```

Figure .124: The results from comparison to entry labeled "DU 6".



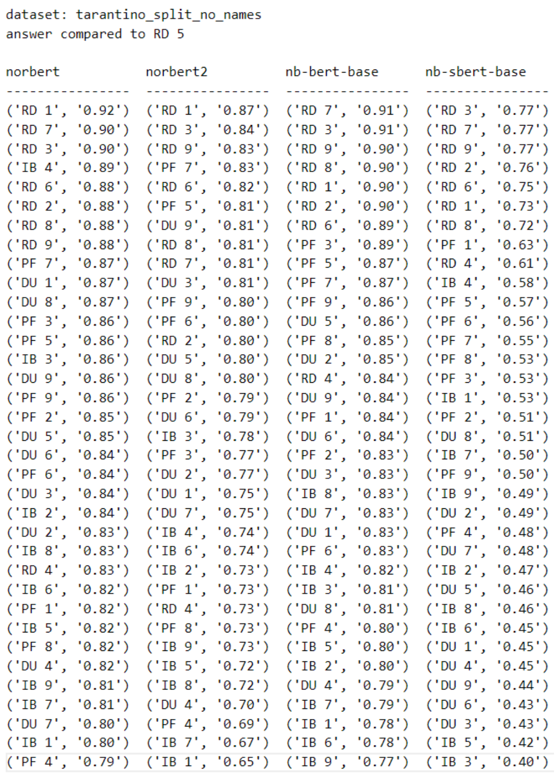
```
dataset: tarantino_split_no_names
answer compared to RD 4

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('RD 9', '0.92')  ('RD 3', '0.78')  ('RD 9', '0.90')  ('RD 6', '0.79')
('RD 3', '0.89')  ('RD 6', '0.78')  ('RD 7', '0.88')  ('RD 9', '0.79')
('RD 6', '0.89')  ('RD 7', '0.78')  ('RD 3', '0.87')  ('RD 3', '0.76')
('RD 7', '0.89')  ('RD 8', '0.78')  ('RD 6', '0.86')  ('RD 8', '0.71')
('RD 8', '0.88')  ('RD 9', '0.78')  ('RD 8', '0.86')  ('RD 2', '0.70')
('PF 6', '0.87')  ('RD 2', '0.76')  ('RD 2', '0.85')  ('RD 7', '0.64')
('PF 7', '0.87')  ('PF 5', '0.75')  ('RD 5', '0.84')  ('RD 5', '0.61')
('RD 2', '0.87')  ('PF 9', '0.75')  ('DU 2', '0.83')  ('PF 1', '0.61')
('PF 5', '0.87')  ('PF 7', '0.74')  ('PF 3', '0.83')  ('RD 1', '0.60')
('PF 1', '0.86')  ('DU 5', '0.73')  ('PF 1', '0.83')  ('IB 7', '0.54')
('PF 3', '0.86')  ('RD 5', '0.73')  ('DU 5', '0.83')  ('IB 1', '0.53')
('PF 8', '0.86')  ('RD 1', '0.73')  ('DU 7', '0.82')  ('IB 4', '0.52')
('DU 3', '0.86')  ('DU 3', '0.72')  ('PF 9', '0.82')  ('PF 6', '0.52')
('RD 1', '0.85')  ('DU 2', '0.72')  ('PF 7', '0.82')  ('DU 2', '0.49')
('PF 9', '0.85')  ('DU 9', '0.72')  ('RD 1', '0.82')  ('DU 8', '0.48')
('DU 9', '0.85')  ('DU 7', '0.71')  ('PF 6', '0.82')  ('PF 8', '0.48')
('PF 2', '0.85')  ('PF 6', '0.71')  ('PF 5', '0.81')  ('IB 9', '0.46')
('DU 5', '0.84')  ('PF 2', '0.71')  ('DU 9', '0.81')  ('PF 9', '0.46')
('PF 4', '0.84')  ('IB 9', '0.70')  ('PF 4', '0.81')  ('IB 8', '0.46')
('IB 9', '0.84')  ('IB 8', '0.70')  ('IB 8', '0.81')  ('PF 4', '0.46')
('IB 7', '0.83')  ('IB 4', '0.70')  ('PF 8', '0.80')  ('IB 2', '0.45')
('RD 5', '0.83')  ('PF 3', '0.70')  ('DU 6', '0.80')  ('PF 7', '0.45')
('IB 4', '0.83')  ('PF 1', '0.70')  ('IB 7', '0.79')  ('PF 3', '0.44')
('IB 8', '0.82')  ('IB 3', '0.69')  ('DU 3', '0.79')  ('PF 5', '0.44')
('DU 1', '0.82')  ('IB 7', '0.68')  ('IB 3', '0.78')  ('DU 5', '0.44')
('IB 3', '0.82')  ('IB 5', '0.68')  ('IB 5', '0.77')  ('DU 6', '0.44')
('DU 7', '0.82')  ('DU 1', '0.67')  ('PF 2', '0.77')  ('DU 7', '0.43')
('DU 2', '0.81')  ('DU 4', '0.67')  ('IB 4', '0.77')  ('PF 2', '0.42')
('DU 8', '0.81')  ('DU 6', '0.67')  ('DU 4', '0.76')  ('IB 3', '0.41')
('IB 2', '0.81')  ('IB 6', '0.66')  ('DU 1', '0.76')  ('IB 5', '0.41')
('DU 4', '0.81')  ('PF 8', '0.66')  ('IB 9', '0.75')  ('IB 6', '0.41')
('DU 6', '0.80')  ('IB 2', '0.65')  ('IB 2', '0.75')  ('DU 3', '0.41')
('IB 5', '0.80')  ('PF 4', '0.65')  ('IB 6', '0.75')  ('DU 9', '0.40')
('IB 6', '0.77')  ('DU 8', '0.63')  ('IB 1', '0.74')  ('DU 1', '0.39')
('IB 1', '0.75')  ('IB 1', '0.60')  ('DU 8', '0.71')  ('DU 4', '0.37')
```

Figure .125: The results from comparison to entry labeled "RD 4".



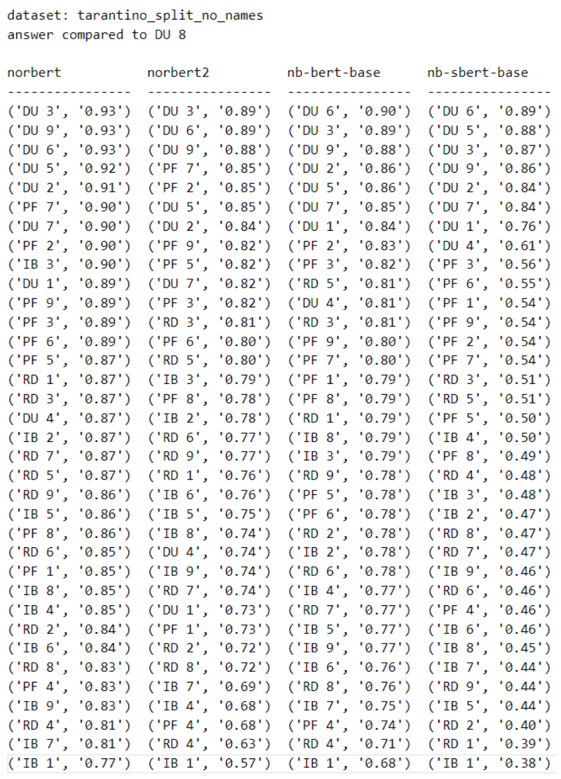
```
dataset: tarantino_split_no_names
answer compared to DU 7

norbert           norbert2          nb-bert-base      nb-sbert-base
---------------   ---------------   ---------------   ---------------
('DU 2', '0.93')  ('DU 2', '0.88')  ('DU 2', '0.92')  ('DU 6', '0.86')
('DU 3', '0.91')  ('DU 5', '0.86')  ('DU 6', '0.92')  ('DU 2', '0.85')
('DU 6', '0.91')  ('DU 6', '0.86')  ('DU 5', '0.91')  ('DU 3', '0.84')
('DU 5', '0.91')  ('DU 9', '0.85')  ('DU 3', '0.90')  ('DU 8', '0.84')
('DU 8', '0.90')  ('DU 3', '0.85')  ('DU 9', '0.89')  ('DU 5', '0.83')
('DU 9', '0.88')  ('DU 8', '0.82')  ('RD 9', '0.86')  ('DU 9', '0.81')
('PF 3', '0.88')  ('RD 3', '0.80')  ('DU 8', '0.85')  ('DU 1', '0.63')
('IB 9', '0.87')  ('PF 5', '0.79')  ('RD 3', '0.85')  ('PF 1', '0.52')
('IB 3', '0.87')  ('PF 7', '0.79')  ('PF 3', '0.85')  ('DU 4', '0.52')
('IB 5', '0.87')  ('PF 9', '0.79')  ('PF 9', '0.84')  ('PF 6', '0.52')
('PF 7', '0.87')  ('IB 3', '0.78')  ('RD 7', '0.84')  ('PF 9', '0.51')
('PF 2', '0.87')  ('DU 1', '0.78')  ('RD 6', '0.84')  ('IB 4', '0.51')
('IB 2', '0.87')  ('DU 4', '0.78')  ('PF 7', '0.84')  ('IB 9', '0.50')
('PF 5', '0.86')  ('IB 8', '0.78')  ('DU 1', '0.83')  ('IB 6', '0.50')
('DU 4', '0.86')  ('IB 9', '0.77')  ('PF 6', '0.83')  ('PF 3', '0.50')
('IB 6', '0.86')  ('RD 7', '0.77')  ('RD 5', '0.83')  ('PF 5', '0.49')
('PF 6', '0.86')  ('PF 2', '0.77')  ('IB 8', '0.83')  ('IB 2', '0.48')
('PF 9', '0.86')  ('PF 6', '0.76')  ('PF 1', '0.83')  ('IB 7', '0.48')
('PF 8', '0.86')  ('IB 5', '0.76')  ('RD 2', '0.83')  ('RD 5', '0.48')
('IB 7', '0.86')  ('IB 7', '0.76')  ('PF 4', '0.83')  ('PF 8', '0.47')
('RD 9', '0.85')  ('RD 9', '0.76')  ('RD 8', '0.82')  ('PF 4', '0.47')
('IB 8', '0.85')  ('PF 3', '0.76')  ('RD 4', '0.82')  ('PF 2', '0.46')
('RD 3', '0.85')  ('IB 2', '0.75')  ('PF 8', '0.82')  ('PF 7', '0.46')
('DU 1', '0.84')  ('RD 8', '0.75')  ('PF 5', '0.82')  ('IB 5', '0.46')
('PF 1', '0.84')  ('RD 2', '0.75')  ('DU 4', '0.82')  ('IB 3', '0.46')
('RD 1', '0.83')  ('RD 5', '0.75')  ('RD 1', '0.82')  ('RD 9', '0.46')
('PF 4', '0.83')  ('IB 6', '0.74')  ('PF 2', '0.82')  ('IB 8', '0.44')
('RD 7', '0.83')  ('RD 1', '0.74')  ('IB 3', '0.81')  ('RD 6', '0.43')
('RD 6', '0.82')  ('RD 6', '0.74')  ('IB 2', '0.81')  ('RD 4', '0.43')
('RD 8', '0.82')  ('PF 1', '0.73')  ('IB 7', '0.81')  ('RD 3', '0.42')
('IB 4', '0.82')  ('PF 8', '0.72')  ('IB 9', '0.79')  ('RD 8', '0.39')
('RD 4', '0.82')  ('RD 4', '0.71')  ('IB 5', '0.79')  ('RD 2', '0.39')
('RD 5', '0.80')  ('IB 4', '0.70')  ('IB 6', '0.79')  ('RD 7', '0.39')
('RD 2', '0.80')  ('PF 4', '0.68')  ('IB 4', '0.78')  ('RD 1', '0.38')
('IB 1', '0.79')  ('IB 1', '0.62')  ('IB 1', '0.73')  ('IB 1', '0.35')
```

Figure .126: The results from comparison to entry labeled "DU 7".



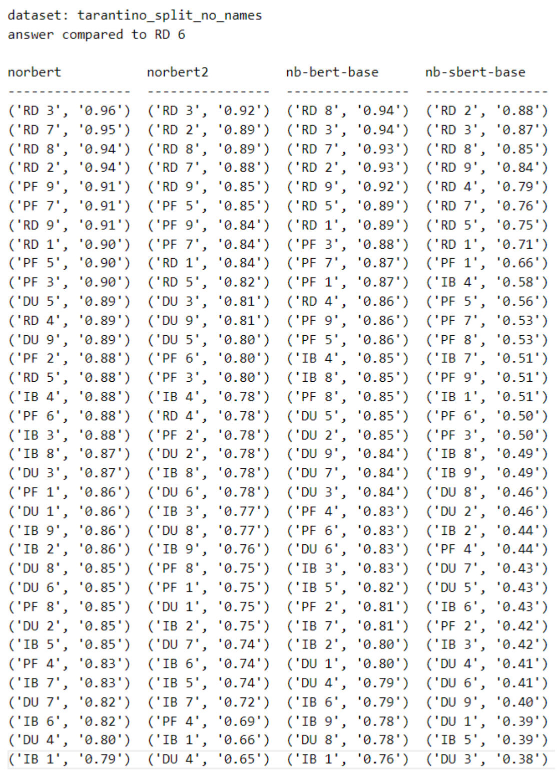
```
dataset: tarantino_split_no_names
answer compared to RD 5

norbert           norbert2          nb-bert-base      nb-sbert-base
---------------   ---------------   ---------------   ---------------
('RD 1', '0.92')  ('RD 1', '0.87')  ('RD 7', '0.91')  ('RD 3', '0.77')
('RD 7', '0.90')  ('RD 3', '0.84')  ('RD 3', '0.91')  ('RD 7', '0.77')
('RD 3', '0.90')  ('RD 9', '0.83')  ('RD 9', '0.90')  ('RD 9', '0.77')
('IB 4', '0.89')  ('PF 7', '0.83')  ('RD 8', '0.90')  ('RD 2', '0.76')
('RD 6', '0.88')  ('RD 6', '0.82')  ('RD 1', '0.90')  ('RD 6', '0.75')
('RD 2', '0.88')  ('PF 5', '0.81')  ('RD 2', '0.90')  ('RD 1', '0.73')
('RD 8', '0.88')  ('DU 9', '0.81')  ('RD 6', '0.89')  ('RD 8', '0.72')
('RD 9', '0.88')  ('RD 8', '0.81')  ('PF 3', '0.89')  ('PF 1', '0.63')
('PF 7', '0.87')  ('RD 7', '0.81')  ('PF 5', '0.87')  ('RD 4', '0.61')
('DU 1', '0.87')  ('DU 3', '0.81')  ('PF 7', '0.87')  ('IB 4', '0.58')
('DU 8', '0.87')  ('PF 9', '0.80')  ('PF 9', '0.86')  ('PF 5', '0.57')
('PF 3', '0.86')  ('PF 6', '0.80')  ('DU 5', '0.86')  ('PF 6', '0.56')
('PF 5', '0.86')  ('RD 2', '0.80')  ('PF 8', '0.85')  ('PF 7', '0.55')
('IB 3', '0.86')  ('DU 5', '0.80')  ('DU 2', '0.85')  ('PF 8', '0.53')
('DU 9', '0.86')  ('DU 8', '0.80')  ('RD 4', '0.84')  ('PF 3', '0.53')
('PF 9', '0.86')  ('PF 2', '0.79')  ('DU 9', '0.84')  ('IB 1', '0.53')
('PF 2', '0.85')  ('DU 6', '0.79')  ('PF 1', '0.84')  ('PF 2', '0.51')
('DU 5', '0.85')  ('IB 3', '0.78')  ('DU 6', '0.84')  ('DU 8', '0.51')
('DU 6', '0.84')  ('PF 3', '0.77')  ('PF 2', '0.83')  ('IB 7', '0.50')
('PF 6', '0.84')  ('DU 2', '0.77')  ('DU 3', '0.83')  ('PF 9', '0.50')
('DU 3', '0.84')  ('DU 1', '0.75')  ('IB 8', '0.83')  ('IB 9', '0.49')
('IB 2', '0.84')  ('DU 7', '0.75')  ('DU 7', '0.83')  ('DU 2', '0.49')
('DU 2', '0.83')  ('IB 4', '0.74')  ('DU 1', '0.83')  ('PF 4', '0.48')
('IB 8', '0.83')  ('IB 6', '0.74')  ('PF 6', '0.83')  ('DU 7', '0.48')
('RD 4', '0.83')  ('IB 2', '0.73')  ('IB 4', '0.82')  ('IB 2', '0.47')
('IB 6', '0.82')  ('PF 1', '0.73')  ('IB 3', '0.81')  ('DU 5', '0.46')
('PF 1', '0.82')  ('RD 4', '0.73')  ('DU 8', '0.81')  ('IB 8', '0.46')
('IB 5', '0.82')  ('PF 8', '0.73')  ('PF 4', '0.80')  ('IB 6', '0.45')
('PF 8', '0.82')  ('IB 9', '0.73')  ('IB 5', '0.80')  ('DU 1', '0.45')
('DU 4', '0.82')  ('IB 5', '0.72')  ('IB 2', '0.80')  ('DU 4', '0.45')
('IB 9', '0.81')  ('IB 8', '0.72')  ('DU 4', '0.79')  ('DU 9', '0.44')
('IB 7', '0.81')  ('DU 4', '0.70')  ('IB 7', '0.79')  ('DU 6', '0.43')
('DU 7', '0.80')  ('PF 4', '0.69')  ('IB 1', '0.78')  ('DU 3', '0.43')
('IB 1', '0.80')  ('IB 7', '0.67')  ('IB 6', '0.78')  ('IB 5', '0.42')
('PF 4', '0.79')  ('IB 1', '0.65')  ('IB 9', '0.77')  ('IB 3', '0.40')
```

Figure .127: The results from comparison to entry labeled "RD 5".



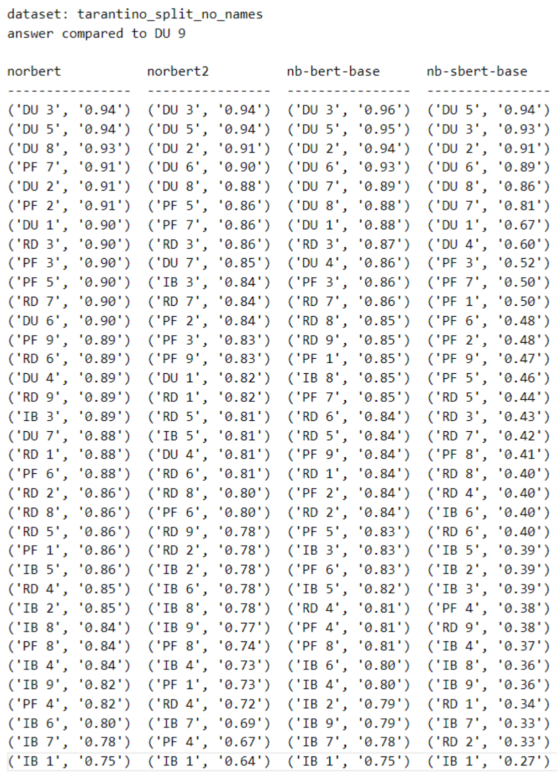
```
dataset: tarantino_split_no_names
answer compared to DU 8

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('DU 3', '0.93')    ('DU 3', '0.89')    ('DU 6', '0.90')    ('DU 6', '0.89')
('DU 9', '0.93')    ('DU 6', '0.89')    ('DU 3', '0.89')    ('DU 5', '0.88')
('DU 6', '0.93')    ('DU 9', '0.88')    ('DU 9', '0.88')    ('DU 3', '0.87')
('DU 5', '0.92')    ('PF 7', '0.85')    ('DU 2', '0.86')    ('DU 9', '0.86')
('DU 2', '0.91')    ('PF 2', '0.85')    ('DU 5', '0.86')    ('DU 2', '0.84')
('PF 7', '0.90')    ('DU 5', '0.85')    ('DU 7', '0.85')    ('DU 7', '0.84')
('DU 7', '0.90')    ('DU 2', '0.84')    ('DU 1', '0.84')    ('DU 1', '0.76')
('PF 2', '0.90')    ('PF 9', '0.82')    ('PF 2', '0.83')    ('DU 4', '0.61')
('IB 3', '0.90')    ('PF 5', '0.82')    ('PF 3', '0.82')    ('PF 3', '0.56')
('DU 1', '0.89')    ('DU 7', '0.82')    ('RD 5', '0.81')    ('PF 6', '0.55')
('PF 9', '0.89')    ('PF 3', '0.82')    ('DU 4', '0.81')    ('PF 1', '0.54')
('PF 3', '0.89')    ('RD 3', '0.81')    ('RD 3', '0.81')    ('PF 9', '0.54')
('PF 6', '0.89')    ('PF 6', '0.80')    ('PF 9', '0.80')    ('PF 2', '0.54')
('PF 5', '0.87')    ('RD 5', '0.80')    ('PF 7', '0.80')    ('PF 7', '0.54')
('RD 1', '0.87')    ('IB 3', '0.79')    ('PF 1', '0.79')    ('RD 3', '0.51')
('RD 3', '0.87')    ('PF 8', '0.78')    ('PF 8', '0.79')    ('RD 5', '0.51')
('DU 4', '0.87')    ('IB 2', '0.78')    ('RD 1', '0.79')    ('PF 5', '0.50')
('IB 2', '0.87')    ('RD 6', '0.77')    ('IB 8', '0.79')    ('IB 4', '0.50')
('RD 7', '0.87')    ('RD 9', '0.77')    ('IB 3', '0.79')    ('PF 8', '0.49')
('RD 5', '0.87')    ('RD 1', '0.76')    ('RD 9', '0.78')    ('RD 4', '0.48')
('RD 9', '0.86')    ('IB 6', '0.76')    ('PF 5', '0.78')    ('IB 3', '0.48')
('IB 5', '0.86')    ('IB 5', '0.75')    ('PF 6', '0.78')    ('IB 2', '0.47')
('PF 8', '0.86')    ('IB 8', '0.74')    ('RD 2', '0.78')    ('RD 8', '0.47')
('RD 6', '0.85')    ('DU 4', '0.74')    ('IB 2', '0.78')    ('RD 7', '0.47')
('PF 1', '0.85')    ('IB 9', '0.74')    ('RD 6', '0.78')    ('IB 9', '0.46')
('IB 8', '0.85')    ('RD 7', '0.74')    ('IB 4', '0.77')    ('RD 6', '0.46')
('IB 4', '0.85')    ('DU 1', '0.73')    ('RD 7', '0.77')    ('PF 4', '0.46')
('RD 2', '0.84')    ('PF 1', '0.73')    ('IB 5', '0.77')    ('IB 6', '0.46')
('IB 6', '0.84')    ('RD 2', '0.72')    ('IB 9', '0.77')    ('IB 8', '0.45')
('RD 8', '0.83')    ('RD 8', '0.72')    ('IB 6', '0.76')    ('IB 7', '0.44')
('PF 4', '0.83')    ('IB 7', '0.69')    ('RD 8', '0.76')    ('RD 9', '0.44')
('IB 9', '0.83')    ('IB 4', '0.68')    ('IB 7', '0.75')    ('IB 5', '0.44')
('RD 4', '0.81')    ('PF 4', '0.68')    ('PF 4', '0.74')    ('RD 2', '0.40')
('IB 7', '0.81')    ('RD 4', '0.63')    ('RD 4', '0.71')    ('RD 1', '0.39')
('IB 1', '0.77')    ('IB 1', '0.57')    ('IB 1', '0.68')    ('IB 1', '0.38')
```

Figure .128: The results from comparison to entry labeled "DU 8".



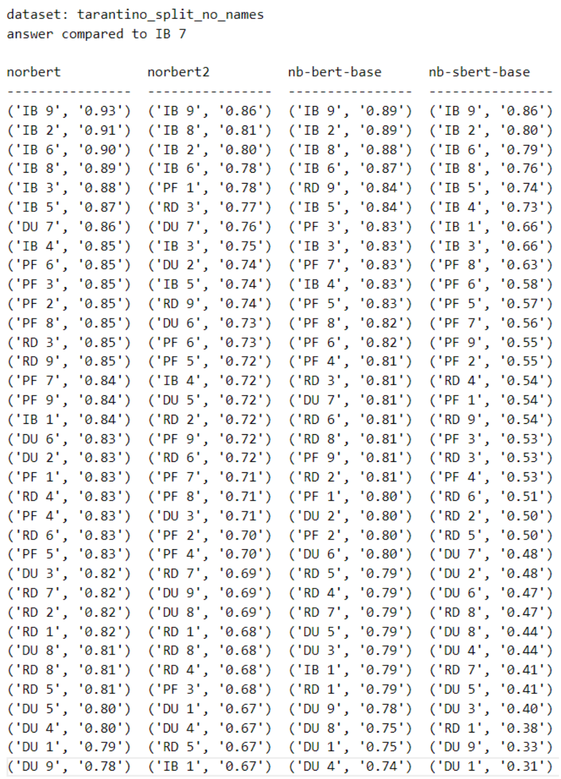
```
dataset: tarantino_split_no_names
answer compared to RD 6

norbert             norbert2            nb-bert-base        nb-sbert-base
----------------    ----------------    ----------------    ----------------
('RD 3', '0.96')    ('RD 3', '0.92')    ('RD 8', '0.94')    ('RD 2', '0.88')
('RD 7', '0.95')    ('RD 2', '0.89')    ('RD 3', '0.94')    ('RD 3', '0.87')
('RD 8', '0.94')    ('RD 8', '0.89')    ('RD 7', '0.93')    ('RD 8', '0.85')
('RD 2', '0.94')    ('RD 7', '0.88')    ('RD 2', '0.93')    ('RD 9', '0.84')
('PF 9', '0.91')    ('RD 9', '0.85')    ('RD 9', '0.92')    ('RD 4', '0.79')
('PF 7', '0.91')    ('PF 5', '0.85')    ('RD 5', '0.89')    ('RD 7', '0.76')
('RD 9', '0.91')    ('PF 9', '0.84')    ('RD 1', '0.89')    ('RD 5', '0.75')
('RD 1', '0.90')    ('PF 7', '0.84')    ('PF 3', '0.88')    ('RD 1', '0.71')
('PF 5', '0.90')    ('RD 1', '0.84')    ('PF 7', '0.87')    ('PF 1', '0.66')
('PF 3', '0.90')    ('RD 5', '0.82')    ('PF 1', '0.87')    ('IB 4', '0.58')
('DU 5', '0.89')    ('DU 3', '0.81')    ('RD 4', '0.86')    ('PF 5', '0.56')
('RD 4', '0.89')    ('DU 9', '0.81')    ('PF 9', '0.86')    ('PF 7', '0.53')
('DU 9', '0.89')    ('DU 5', '0.80')    ('PF 5', '0.86')    ('PF 8', '0.53')
('PF 2', '0.88')    ('PF 6', '0.80')    ('IB 4', '0.85')    ('IB 7', '0.51')
('RD 5', '0.88')    ('PF 3', '0.80')    ('IB 8', '0.85')    ('PF 9', '0.51')
('IB 4', '0.88')    ('IB 4', '0.78')    ('PF 8', '0.85')    ('IB 1', '0.51')
('PF 6', '0.88')    ('RD 4', '0.78')    ('DU 5', '0.85')    ('PF 6', '0.50')
('IB 3', '0.88')    ('PF 2', '0.78')    ('DU 2', '0.85')    ('PF 3', '0.50')
('IB 8', '0.87')    ('DU 2', '0.78')    ('DU 9', '0.84')    ('IB 8', '0.49')
('DU 3', '0.87')    ('IB 8', '0.78')    ('DU 7', '0.84')    ('IB 9', '0.49')
('PF 1', '0.86')    ('DU 6', '0.78')    ('DU 3', '0.84')    ('DU 8', '0.46')
('DU 1', '0.86')    ('IB 3', '0.77')    ('PF 4', '0.83')    ('DU 2', '0.46')
('IB 9', '0.86')    ('DU 8', '0.77')    ('PF 6', '0.83')    ('IB 2', '0.44')
('IB 2', '0.86')    ('IB 9', '0.76')    ('DU 6', '0.83')    ('PF 4', '0.44')
('DU 8', '0.85')    ('PF 8', '0.75')    ('IB 3', '0.83')    ('DU 7', '0.43')
('DU 6', '0.85')    ('PF 1', '0.75')    ('IB 5', '0.82')    ('DU 5', '0.43')
('PF 8', '0.85')    ('DU 1', '0.75')    ('PF 2', '0.81')    ('IB 6', '0.43')
('DU 2', '0.85')    ('IB 2', '0.75')    ('IB 7', '0.81')    ('PF 2', '0.42')
('IB 5', '0.85')    ('DU 7', '0.74')    ('IB 2', '0.80')    ('IB 3', '0.42')
('PF 4', '0.83')    ('IB 6', '0.74')    ('DU 1', '0.80')    ('DU 4', '0.41')
('IB 7', '0.83')    ('IB 5', '0.74')    ('DU 4', '0.79')    ('DU 6', '0.41')
('DU 7', '0.82')    ('IB 7', '0.72')    ('IB 6', '0.79')    ('DU 9', '0.40')
('IB 6', '0.82')    ('PF 4', '0.69')    ('IB 9', '0.78')    ('DU 1', '0.39')
('DU 4', '0.80')    ('IB 1', '0.66')    ('DU 8', '0.78')    ('IB 5', '0.39')
('IB 1', '0.79')    ('DU 4', '0.65')    ('IB 1', '0.76')    ('DU 3', '0.38')
```

Figure .129: The results from comparison to entry labeled "RD 6".



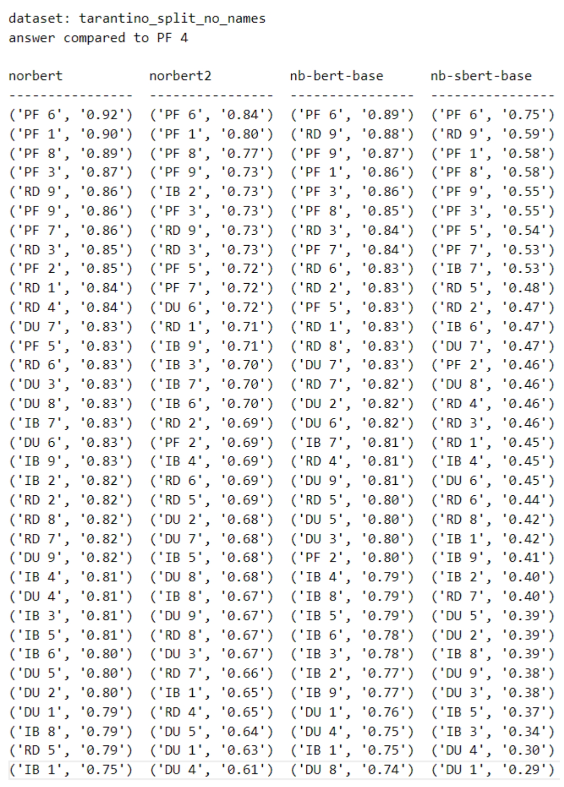
```
dataset: tarantino_split_no_names
answer compared to DU 9

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('DU 3', '0.94')  ('DU 3', '0.94')  ('DU 3', '0.96')  ('DU 5', '0.94')
('DU 5', '0.94')  ('DU 5', '0.94')  ('DU 5', '0.95')  ('DU 3', '0.93')
('DU 8', '0.93')  ('DU 2', '0.91')  ('DU 2', '0.94')  ('DU 2', '0.91')
('PF 7', '0.91')  ('DU 6', '0.90')  ('DU 6', '0.93')  ('DU 6', '0.89')
('DU 2', '0.91')  ('DU 8', '0.88')  ('DU 7', '0.89')  ('DU 8', '0.86')
('PF 2', '0.91')  ('PF 5', '0.86')  ('DU 8', '0.88')  ('DU 7', '0.81')
('DU 1', '0.90')  ('PF 7', '0.86')  ('DU 1', '0.88')  ('DU 1', '0.67')
('RD 3', '0.90')  ('RD 3', '0.86')  ('RD 3', '0.87')  ('DU 4', '0.60')
('PF 3', '0.90')  ('DU 7', '0.85')  ('DU 4', '0.86')  ('PF 3', '0.52')
('PF 5', '0.90')  ('IB 3', '0.84')  ('PF 3', '0.86')  ('PF 7', '0.50')
('RD 7', '0.90')  ('RD 7', '0.84')  ('RD 7', '0.86')  ('PF 1', '0.50')
('DU 6', '0.90')  ('PF 2', '0.84')  ('RD 8', '0.85')  ('PF 6', '0.48')
('PF 9', '0.89')  ('PF 3', '0.83')  ('RD 9', '0.85')  ('PF 2', '0.48')
('RD 6', '0.89')  ('PF 9', '0.83')  ('PF 1', '0.85')  ('PF 9', '0.47')
('DU 4', '0.89')  ('DU 1', '0.82')  ('IB 8', '0.85')  ('PF 5', '0.46')
('RD 9', '0.89')  ('RD 1', '0.82')  ('PF 7', '0.85')  ('RD 5', '0.44')
('IB 3', '0.89')  ('RD 5', '0.81')  ('RD 6', '0.84')  ('RD 3', '0.43')
('DU 7', '0.88')  ('IB 5', '0.81')  ('RD 5', '0.84')  ('RD 7', '0.42')
('RD 1', '0.88')  ('DU 4', '0.81')  ('PF 9', '0.84')  ('PF 8', '0.41')
('PF 6', '0.88')  ('RD 6', '0.81')  ('RD 1', '0.84')  ('RD 8', '0.40')
('RD 2', '0.86')  ('RD 8', '0.80')  ('PF 2', '0.84')  ('RD 4', '0.40')
('RD 8', '0.86')  ('PF 6', '0.80')  ('RD 2', '0.84')  ('IB 6', '0.40')
('RD 5', '0.86')  ('RD 9', '0.78')  ('PF 5', '0.83')  ('RD 6', '0.40')
('PF 1', '0.86')  ('RD 2', '0.78')  ('IB 3', '0.83')  ('IB 5', '0.39')
('IB 5', '0.86')  ('IB 2', '0.78')  ('PF 6', '0.83')  ('IB 2', '0.39')
('RD 4', '0.85')  ('IB 6', '0.78')  ('IB 5', '0.82')  ('IB 3', '0.39')
('IB 2', '0.85')  ('IB 8', '0.78')  ('RD 4', '0.81')  ('PF 4', '0.38')
('IB 8', '0.84')  ('IB 9', '0.77')  ('PF 4', '0.81')  ('RD 9', '0.38')
('PF 8', '0.84')  ('PF 8', '0.74')  ('PF 8', '0.81')  ('IB 4', '0.37')
('IB 4', '0.84')  ('IB 4', '0.73')  ('IB 6', '0.80')  ('IB 8', '0.36')
('IB 9', '0.82')  ('PF 1', '0.73')  ('IB 4', '0.80')  ('IB 9', '0.36')
('PF 4', '0.82')  ('RD 4', '0.72')  ('IB 2', '0.79')  ('RD 1', '0.34')
('IB 6', '0.80')  ('IB 7', '0.69')  ('IB 9', '0.79')  ('IB 7', '0.33')
('IB 7', '0.78')  ('PF 4', '0.67')  ('IB 7', '0.78')  ('RD 2', '0.33')
('IB 1', '0.75')  ('IB 1', '0.64')  ('IB 1', '0.75')  ('IB 1', '0.27')
```

Figure .130: The results from comparison to entry labeled "DU 9".



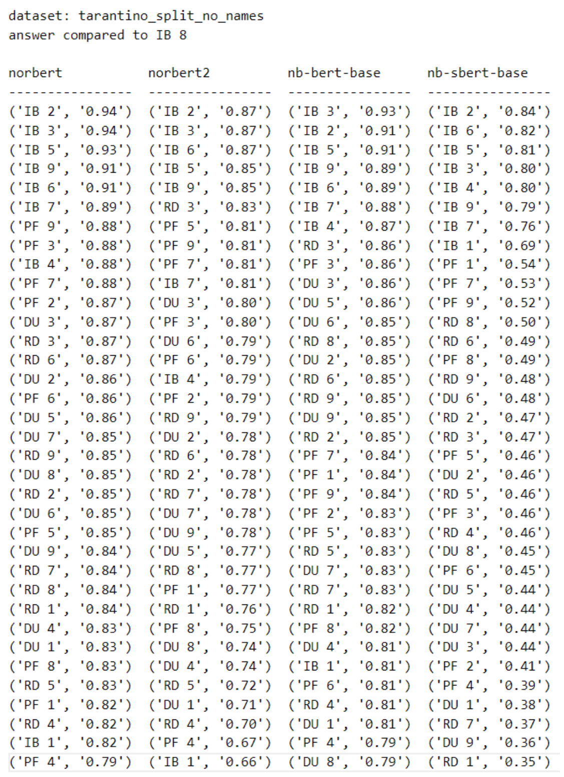
```
dataset: tarantino_split_no_names
answer compared to IB 7

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('IB 9', '0.93')  ('IB 9', '0.86')  ('IB 9', '0.89')  ('IB 9', '0.86')
('IB 2', '0.91')  ('IB 8', '0.81')  ('IB 2', '0.89')  ('IB 2', '0.80')
('IB 6', '0.90')  ('IB 2', '0.80')  ('IB 8', '0.88')  ('IB 6', '0.79')
('IB 8', '0.89')  ('IB 6', '0.78')  ('IB 6', '0.87')  ('IB 8', '0.76')
('IB 3', '0.88')  ('PF 1', '0.78')  ('RD 9', '0.84')  ('IB 5', '0.74')
('IB 5', '0.87')  ('RD 3', '0.77')  ('IB 5', '0.84')  ('IB 4', '0.73')
('DU 7', '0.86')  ('DU 7', '0.76')  ('PF 3', '0.83')  ('IB 1', '0.66')
('IB 4', '0.85')  ('IB 3', '0.75')  ('IB 3', '0.83')  ('IB 3', '0.66')
('PF 6', '0.85')  ('DU 2', '0.74')  ('PF 7', '0.83')  ('PF 8', '0.63')
('PF 3', '0.85')  ('IB 5', '0.74')  ('IB 4', '0.83')  ('PF 6', '0.58')
('PF 2', '0.85')  ('RD 9', '0.74')  ('PF 5', '0.83')  ('PF 5', '0.57')
('PF 8', '0.85')  ('DU 6', '0.73')  ('PF 8', '0.82')  ('PF 7', '0.56')
('RD 3', '0.85')  ('PF 6', '0.73')  ('PF 6', '0.82')  ('PF 9', '0.55')
('RD 9', '0.85')  ('PF 5', '0.72')  ('PF 4', '0.81')  ('PF 2', '0.55')
('PF 7', '0.84')  ('IB 4', '0.72')  ('RD 3', '0.81')  ('RD 4', '0.54')
('PF 9', '0.84')  ('DU 5', '0.72')  ('DU 7', '0.81')  ('PF 1', '0.54')
('IB 1', '0.84')  ('RD 2', '0.72')  ('RD 6', '0.81')  ('RD 9', '0.54')
('DU 6', '0.83')  ('PF 9', '0.72')  ('RD 8', '0.81')  ('PF 3', '0.53')
('DU 2', '0.83')  ('RD 6', '0.72')  ('PF 9', '0.81')  ('RD 3', '0.53')
('PF 1', '0.83')  ('PF 7', '0.71')  ('RD 2', '0.81')  ('PF 4', '0.53')
('RD 4', '0.83')  ('PF 8', '0.71')  ('PF 1', '0.80')  ('RD 6', '0.51')
('PF 4', '0.83')  ('DU 3', '0.71')  ('DU 2', '0.80')  ('RD 2', '0.50')
('RD 6', '0.83')  ('PF 2', '0.70')  ('PF 2', '0.80')  ('RD 5', '0.50')
('PF 5', '0.83')  ('PF 4', '0.70')  ('DU 6', '0.80')  ('DU 7', '0.48')
('DU 3', '0.82')  ('RD 7', '0.69')  ('RD 5', '0.79')  ('DU 2', '0.48')
('RD 7', '0.82')  ('DU 9', '0.69')  ('RD 4', '0.79')  ('DU 6', '0.47')
('RD 2', '0.82')  ('DU 8', '0.69')  ('RD 7', '0.79')  ('RD 8', '0.47')
('RD 1', '0.82')  ('RD 1', '0.68')  ('DU 5', '0.79')  ('DU 8', '0.44')
('DU 8', '0.81')  ('RD 8', '0.68')  ('DU 3', '0.79')  ('DU 4', '0.44')
('RD 8', '0.81')  ('RD 4', '0.68')  ('IB 1', '0.79')  ('RD 7', '0.41')
('RD 5', '0.81')  ('PF 3', '0.68')  ('RD 1', '0.79')  ('DU 5', '0.41')
('DU 5', '0.80')  ('DU 1', '0.67')  ('DU 9', '0.78')  ('DU 3', '0.40')
('DU 4', '0.80')  ('DU 4', '0.67')  ('DU 8', '0.75')  ('RD 1', '0.38')
('DU 1', '0.79')  ('RD 5', '0.67')  ('DU 1', '0.75')  ('DU 9', '0.33')
('DU 9', '0.78')  ('IB 1', '0.67')  ('DU 4', '0.74')  ('DU 1', '0.31')
```

Figure .131: The results from comparison to entry labeled "IB 7".



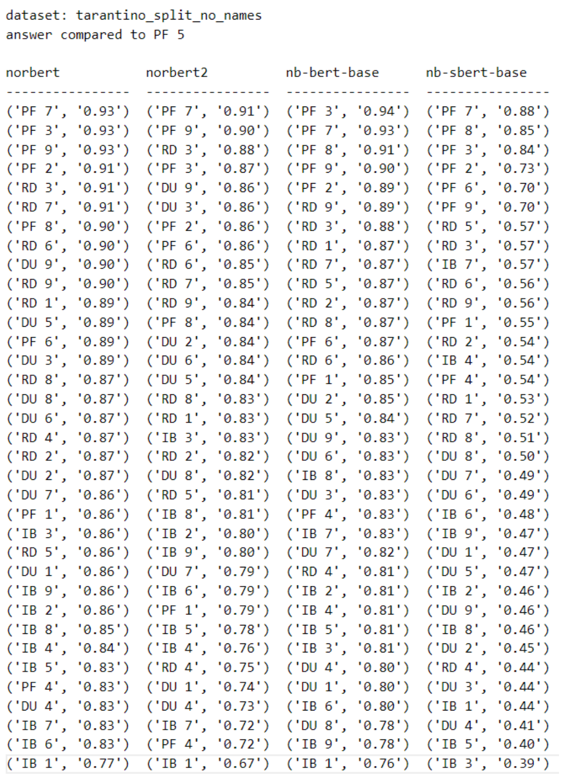
```
dataset: tarantino_split_no_names
answer compared to PF 4

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('PF 6', '0.92')  ('PF 6', '0.84')  ('PF 6', '0.89')  ('PF 6', '0.75')
('PF 1', '0.90')  ('PF 1', '0.80')  ('RD 9', '0.88')  ('RD 9', '0.59')
('PF 8', '0.89')  ('PF 8', '0.77')  ('PF 9', '0.87')  ('PF 1', '0.58')
('PF 3', '0.87')  ('PF 9', '0.73')  ('PF 1', '0.86')  ('PF 8', '0.58')
('RD 9', '0.86')  ('IB 2', '0.73')  ('PF 3', '0.86')  ('PF 9', '0.55')
('PF 9', '0.86')  ('PF 3', '0.73')  ('PF 8', '0.85')  ('PF 3', '0.55')
('PF 7', '0.86')  ('RD 9', '0.73')  ('RD 3', '0.84')  ('PF 5', '0.54')
('RD 3', '0.85')  ('RD 3', '0.73')  ('PF 7', '0.84')  ('PF 7', '0.53')
('PF 2', '0.85')  ('PF 5', '0.72')  ('RD 6', '0.83')  ('IB 7', '0.53')
('RD 1', '0.84')  ('PF 7', '0.72')  ('RD 2', '0.83')  ('RD 5', '0.48')
('RD 4', '0.84')  ('DU 6', '0.72')  ('PF 5', '0.83')  ('RD 2', '0.47')
('DU 7', '0.83')  ('RD 1', '0.71')  ('RD 1', '0.83')  ('IB 6', '0.47')
('PF 5', '0.83')  ('IB 9', '0.71')  ('RD 8', '0.83')  ('DU 7', '0.47')
('RD 6', '0.83')  ('IB 3', '0.70')  ('DU 7', '0.83')  ('PF 2', '0.46')
('DU 3', '0.83')  ('IB 7', '0.70')  ('RD 7', '0.82')  ('DU 8', '0.46')
('DU 8', '0.83')  ('IB 6', '0.70')  ('DU 2', '0.82')  ('RD 4', '0.46')
('IB 7', '0.83')  ('RD 2', '0.69')  ('DU 6', '0.82')  ('RD 3', '0.46')
('DU 6', '0.83')  ('PF 2', '0.69')  ('IB 7', '0.81')  ('RD 1', '0.45')
('IB 9', '0.83')  ('IB 4', '0.69')  ('RD 4', '0.81')  ('IB 4', '0.45')
('IB 2', '0.82')  ('RD 6', '0.69')  ('DU 9', '0.81')  ('DU 6', '0.45')
('RD 2', '0.82')  ('RD 5', '0.69')  ('RD 5', '0.80')  ('RD 6', '0.44')
('RD 8', '0.82')  ('DU 2', '0.68')  ('DU 5', '0.80')  ('RD 8', '0.42')
('RD 7', '0.82')  ('DU 7', '0.68')  ('DU 3', '0.80')  ('IB 1', '0.42')
('DU 9', '0.82')  ('IB 5', '0.68')  ('PF 2', '0.80')  ('IB 9', '0.41')
('IB 4', '0.81')  ('DU 8', '0.68')  ('IB 4', '0.79')  ('IB 2', '0.40')
('DU 4', '0.81')  ('IB 8', '0.67')  ('IB 8', '0.79')  ('RD 7', '0.40')
('IB 3', '0.81')  ('DU 9', '0.67')  ('IB 5', '0.79')  ('DU 5', '0.39')
('IB 5', '0.81')  ('RD 8', '0.67')  ('IB 6', '0.78')  ('DU 2', '0.39')
('IB 6', '0.80')  ('DU 3', '0.67')  ('IB 3', '0.78')  ('IB 8', '0.39')
('DU 5', '0.80')  ('RD 7', '0.66')  ('IB 2', '0.77')  ('DU 9', '0.38')
('DU 2', '0.80')  ('IB 1', '0.65')  ('IB 9', '0.77')  ('DU 3', '0.38')
('DU 1', '0.79')  ('RD 4', '0.65')  ('DU 1', '0.76')  ('IB 5', '0.37')
('IB 8', '0.79')  ('DU 5', '0.64')  ('DU 4', '0.75')  ('IB 3', '0.34')
('RD 5', '0.79')  ('DU 1', '0.63')  ('IB 1', '0.75')  ('DU 4', '0.30')
('IB 1', '0.75')  ('DU 4', '0.61')  ('DU 8', '0.74')  ('DU 1', '0.29')
```

Figure .132: The results from comparison to entry labeled "PF 4".



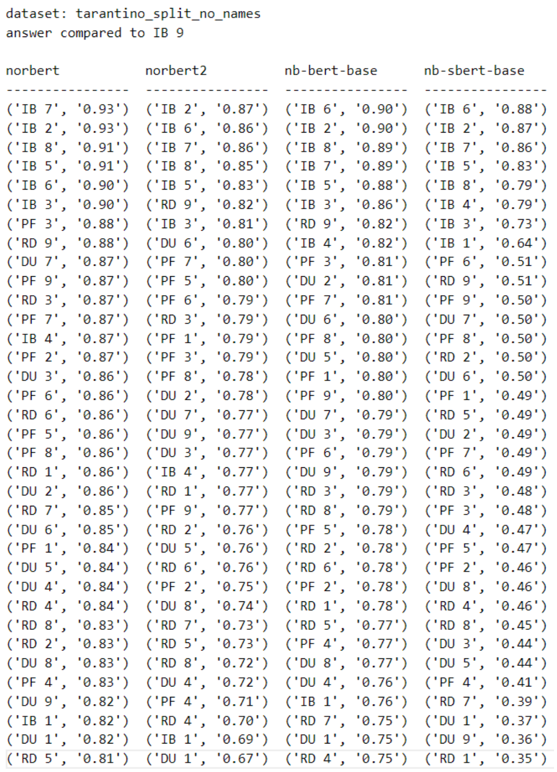
```
dataset: tarantino_split_no_names
answer compared to IB 8

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('IB 2', '0.94')  ('IB 2', '0.87')  ('IB 3', '0.93')  ('IB 2', '0.84')
('IB 3', '0.94')  ('IB 3', '0.87')  ('IB 2', '0.91')  ('IB 6', '0.82')
('IB 5', '0.93')  ('IB 6', '0.87')  ('IB 5', '0.91')  ('IB 5', '0.81')
('IB 9', '0.91')  ('IB 5', '0.85')  ('IB 9', '0.89')  ('IB 3', '0.80')
('IB 6', '0.91')  ('IB 9', '0.85')  ('IB 6', '0.89')  ('IB 4', '0.80')
('IB 7', '0.89')  ('RD 3', '0.83')  ('IB 7', '0.88')  ('IB 9', '0.79')
('PF 9', '0.88')  ('PF 5', '0.81')  ('IB 4', '0.87')  ('IB 7', '0.76')
('PF 3', '0.88')  ('PF 9', '0.81')  ('RD 3', '0.86')  ('IB 1', '0.69')
('IB 4', '0.88')  ('PF 7', '0.81')  ('PF 3', '0.86')  ('PF 1', '0.54')
('PF 7', '0.88')  ('IB 7', '0.81')  ('DU 3', '0.86')  ('PF 7', '0.53')
('PF 2', '0.87')  ('DU 3', '0.80')  ('DU 5', '0.86')  ('PF 9', '0.52')
('DU 3', '0.87')  ('PF 3', '0.80')  ('DU 6', '0.85')  ('RD 8', '0.50')
('RD 3', '0.87')  ('DU 6', '0.79')  ('RD 8', '0.85')  ('RD 6', '0.49')
('RD 6', '0.87')  ('PF 6', '0.79')  ('DU 2', '0.85')  ('PF 8', '0.49')
('DU 2', '0.86')  ('IB 4', '0.79')  ('RD 6', '0.85')  ('RD 9', '0.48')
('PF 6', '0.86')  ('PF 2', '0.79')  ('RD 9', '0.85')  ('DU 6', '0.48')
('DU 5', '0.86')  ('RD 9', '0.79')  ('DU 9', '0.85')  ('RD 2', '0.47')
('DU 7', '0.85')  ('DU 2', '0.78')  ('RD 2', '0.85')  ('RD 3', '0.47')
('RD 9', '0.85')  ('RD 6', '0.78')  ('PF 7', '0.84')  ('PF 5', '0.46')
('DU 8', '0.85')  ('RD 2', '0.78')  ('PF 1', '0.84')  ('DU 2', '0.46')
('RD 2', '0.85')  ('RD 7', '0.78')  ('PF 9', '0.84')  ('RD 5', '0.46')
('DU 6', '0.85')  ('DU 7', '0.78')  ('PF 2', '0.83')  ('PF 3', '0.46')
('PF 5', '0.85')  ('DU 9', '0.78')  ('PF 5', '0.83')  ('RD 4', '0.46')
('DU 9', '0.84')  ('DU 5', '0.77')  ('RD 5', '0.83')  ('DU 8', '0.45')
('RD 7', '0.84')  ('RD 8', '0.77')  ('DU 7', '0.83')  ('PF 6', '0.45')
('RD 8', '0.84')  ('PF 1', '0.77')  ('RD 7', '0.83')  ('DU 5', '0.44')
('RD 1', '0.84')  ('RD 1', '0.76')  ('RD 1', '0.82')  ('DU 4', '0.44')
('DU 4', '0.83')  ('PF 8', '0.75')  ('PF 8', '0.82')  ('DU 7', '0.44')
('DU 1', '0.83')  ('DU 8', '0.74')  ('DU 4', '0.81')  ('DU 3', '0.44')
('PF 8', '0.83')  ('DU 4', '0.74')  ('IB 1', '0.81')  ('PF 2', '0.41')
('RD 5', '0.83')  ('RD 5', '0.72')  ('PF 6', '0.81')  ('PF 4', '0.39')
('PF 1', '0.82')  ('DU 1', '0.71')  ('RD 4', '0.81')  ('DU 1', '0.38')
('RD 4', '0.82')  ('RD 4', '0.70')  ('DU 1', '0.81')  ('RD 7', '0.37')
('IB 1', '0.82')  ('PF 4', '0.67')  ('PF 4', '0.79')  ('DU 9', '0.36')
('PF 4', '0.79')  ('IB 1', '0.66')  ('DU 8', '0.79')  ('RD 1', '0.35')
```

Figure .133: The results from comparison to entry labeled "IB 8".



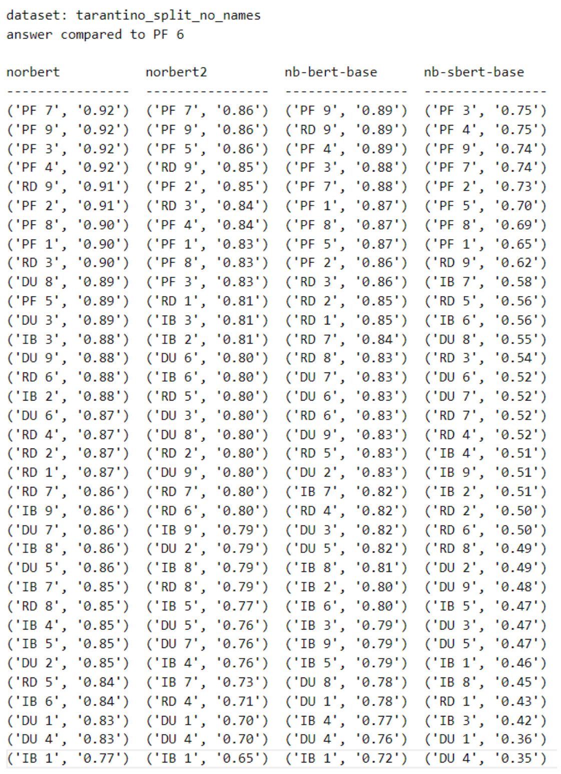
```
dataset: tarantino_split_no_names
answer compared to PF 5

norbert              norbert2             nb-bert-base         nb-sbert-base
---------------      ---------------      ---------------      ---------------
('PF 7', '0.93')     ('PF 7', '0.91')     ('PF 3', '0.94')     ('PF 7', '0.88')
('PF 3', '0.93')     ('PF 9', '0.90')     ('PF 7', '0.93')     ('PF 8', '0.85')
('PF 9', '0.93')     ('RD 3', '0.88')     ('PF 8', '0.91')     ('PF 3', '0.84')
('PF 2', '0.91')     ('PF 3', '0.87')     ('PF 9', '0.90')     ('PF 2', '0.73')
('RD 3', '0.91')     ('DU 9', '0.86')     ('PF 2', '0.89')     ('PF 6', '0.70')
('RD 7', '0.91')     ('DU 3', '0.86')     ('RD 9', '0.89')     ('PF 9', '0.70')
('PF 8', '0.90')     ('PF 2', '0.86')     ('RD 3', '0.88')     ('RD 5', '0.57')
('RD 6', '0.90')     ('PF 6', '0.86')     ('RD 1', '0.87')     ('RD 3', '0.57')
('DU 9', '0.90')     ('RD 6', '0.85')     ('RD 7', '0.87')     ('IB 7', '0.57')
('RD 9', '0.90')     ('RD 7', '0.85')     ('RD 5', '0.87')     ('RD 6', '0.56')
('RD 1', '0.89')     ('RD 9', '0.84')     ('RD 2', '0.87')     ('RD 9', '0.56')
('DU 5', '0.89')     ('PF 8', '0.84')     ('RD 8', '0.87')     ('PF 1', '0.55')
('PF 6', '0.89')     ('DU 2', '0.84')     ('PF 6', '0.87')     ('RD 2', '0.54')
('DU 3', '0.89')     ('DU 6', '0.84')     ('RD 6', '0.86')     ('IB 4', '0.54')
('RD 8', '0.87')     ('DU 5', '0.84')     ('PF 1', '0.85')     ('PF 4', '0.54')
('DU 8', '0.87')     ('RD 8', '0.83')     ('DU 2', '0.85')     ('RD 1', '0.53')
('DU 6', '0.87')     ('RD 1', '0.83')     ('DU 5', '0.84')     ('RD 7', '0.52')
('RD 4', '0.87')     ('IB 3', '0.83')     ('DU 9', '0.83')     ('RD 8', '0.51')
('RD 2', '0.87')     ('RD 2', '0.82')     ('DU 6', '0.83')     ('DU 8', '0.50')
('DU 2', '0.87')     ('DU 8', '0.82')     ('IB 8', '0.83')     ('DU 7', '0.49')
('DU 7', '0.86')     ('RD 5', '0.81')     ('DU 3', '0.83')     ('DU 6', '0.49')
('PF 1', '0.86')     ('IB 8', '0.81')     ('PF 4', '0.83')     ('IB 6', '0.48')
('IB 3', '0.86')     ('IB 2', '0.80')     ('IB 7', '0.83')     ('IB 9', '0.47')
('RD 5', '0.86')     ('IB 9', '0.80')     ('DU 7', '0.82')     ('DU 1', '0.47')
('DU 1', '0.86')     ('DU 7', '0.79')     ('RD 4', '0.81')     ('DU 5', '0.47')
('IB 9', '0.86')     ('IB 6', '0.79')     ('IB 2', '0.81')     ('IB 2', '0.46')
('IB 2', '0.86')     ('PF 1', '0.79')     ('IB 4', '0.81')     ('DU 9', '0.46')
('IB 8', '0.85')     ('IB 5', '0.78')     ('IB 5', '0.81')     ('IB 8', '0.46')
('IB 4', '0.84')     ('IB 4', '0.76')     ('IB 3', '0.81')     ('DU 2', '0.45')
('IB 5', '0.83')     ('RD 4', '0.75')     ('DU 4', '0.80')     ('RD 4', '0.44')
('PF 4', '0.83')     ('DU 1', '0.74')     ('DU 1', '0.80')     ('DU 3', '0.44')
('DU 4', '0.83')     ('DU 4', '0.73')     ('IB 6', '0.80')     ('IB 1', '0.44')
('IB 7', '0.83')     ('IB 7', '0.72')     ('DU 8', '0.78')     ('DU 4', '0.41')
('IB 6', '0.83')     ('PF 4', '0.72')     ('IB 9', '0.78')     ('IB 5', '0.40')
('IB 1', '0.77')     ('IB 1', '0.67')     ('IB 1', '0.76')     ('IB 3', '0.39')
```

Figure .134: The results from comparison to entry labeled "PF 5".



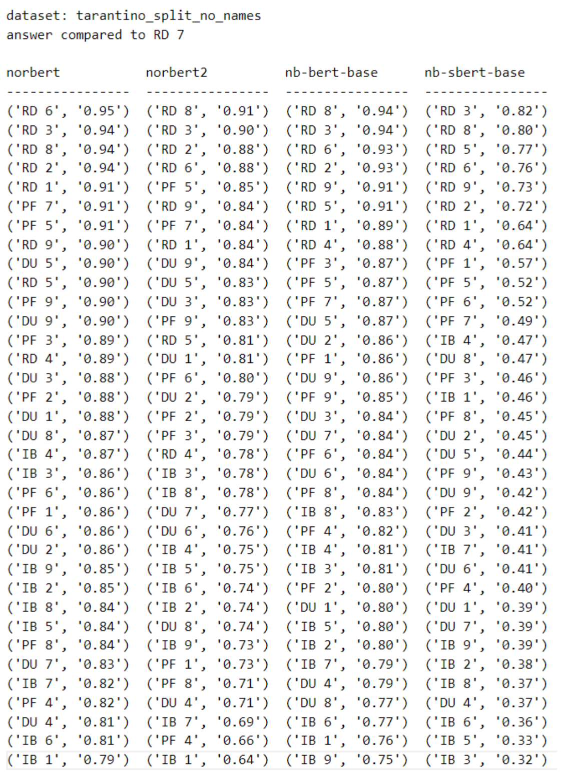
```
dataset: tarantino_split_no_names
answer compared to IB 9

norbert              norbert2             nb-bert-base         nb-sbert-base
---------------      ---------------      ---------------      ---------------
('IB 7', '0.93')     ('IB 2', '0.87')     ('IB 6', '0.90')     ('IB 6', '0.88')
('IB 2', '0.93')     ('IB 6', '0.86')     ('IB 2', '0.90')     ('IB 2', '0.87')
('IB 8', '0.91')     ('IB 7', '0.86')     ('IB 8', '0.89')     ('IB 7', '0.86')
('IB 5', '0.91')     ('IB 8', '0.85')     ('IB 7', '0.89')     ('IB 5', '0.83')
('IB 6', '0.90')     ('IB 5', '0.83')     ('IB 5', '0.88')     ('IB 8', '0.79')
('IB 3', '0.90')     ('RD 9', '0.82')     ('IB 3', '0.86')     ('IB 4', '0.79')
('PF 3', '0.88')     ('IB 3', '0.81')     ('RD 9', '0.82')     ('IB 3', '0.73')
('RD 9', '0.88')     ('DU 6', '0.80')     ('IB 4', '0.82')     ('IB 1', '0.64')
('DU 7', '0.87')     ('PF 7', '0.80')     ('PF 3', '0.81')     ('PF 6', '0.51')
('PF 9', '0.87')     ('PF 5', '0.80')     ('DU 2', '0.81')     ('RD 9', '0.51')
('RD 3', '0.87')     ('PF 6', '0.79')     ('PF 7', '0.81')     ('PF 9', '0.50')
('PF 7', '0.87')     ('RD 3', '0.79')     ('DU 6', '0.80')     ('DU 7', '0.50')
('IB 4', '0.87')     ('PF 1', '0.79')     ('PF 8', '0.80')     ('PF 8', '0.50')
('PF 2', '0.87')     ('PF 3', '0.79')     ('DU 5', '0.80')     ('RD 2', '0.50')
('DU 3', '0.86')     ('PF 8', '0.78')     ('PF 1', '0.80')     ('DU 6', '0.50')
('PF 6', '0.86')     ('DU 2', '0.78')     ('PF 9', '0.80')     ('PF 1', '0.49')
('RD 6', '0.86')     ('DU 7', '0.77')     ('DU 7', '0.79')     ('RD 5', '0.49')
('PF 5', '0.86')     ('DU 9', '0.77')     ('DU 3', '0.79')     ('DU 2', '0.49')
('PF 8', '0.86')     ('DU 3', '0.77')     ('PF 6', '0.79')     ('PF 7', '0.49')
('RD 1', '0.86')     ('IB 4', '0.77')     ('DU 9', '0.79')     ('RD 6', '0.49')
('DU 2', '0.86')     ('RD 1', '0.77')     ('RD 3', '0.79')     ('RD 3', '0.48')
('RD 7', '0.85')     ('PF 9', '0.77')     ('RD 8', '0.79')     ('PF 3', '0.48')
('DU 6', '0.85')     ('RD 2', '0.76')     ('PF 5', '0.78')     ('DU 4', '0.47')
('PF 1', '0.84')     ('DU 5', '0.76')     ('RD 2', '0.78')     ('PF 5', '0.47')
('DU 5', '0.84')     ('RD 6', '0.76')     ('RD 6', '0.78')     ('PF 2', '0.46')
('DU 4', '0.84')     ('PF 2', '0.75')     ('PF 2', '0.78')     ('DU 8', '0.46')
('RD 4', '0.84')     ('DU 8', '0.74')     ('RD 1', '0.78')     ('RD 4', '0.46')
('RD 8', '0.83')     ('RD 7', '0.73')     ('RD 5', '0.77')     ('RD 8', '0.45')
('RD 2', '0.83')     ('RD 5', '0.73')     ('PF 4', '0.77')     ('DU 3', '0.44')
('DU 8', '0.83')     ('RD 8', '0.72')     ('DU 8', '0.77')     ('DU 5', '0.44')
('PF 4', '0.83')     ('DU 4', '0.72')     ('DU 4', '0.76')     ('PF 4', '0.41')
('DU 9', '0.82')     ('PF 4', '0.71')     ('IB 1', '0.76')     ('RD 7', '0.39')
('IB 1', '0.82')     ('RD 4', '0.70')     ('RD 7', '0.75')     ('DU 1', '0.37')
('DU 1', '0.82')     ('IB 1', '0.69')     ('DU 1', '0.75')     ('DU 9', '0.36')
('RD 5', '0.81')     ('DU 1', '0.67')     ('RD 4', '0.75')     ('RD 1', '0.35')
```

Figure .135: The results from comparison to entry labeled "IB 9".



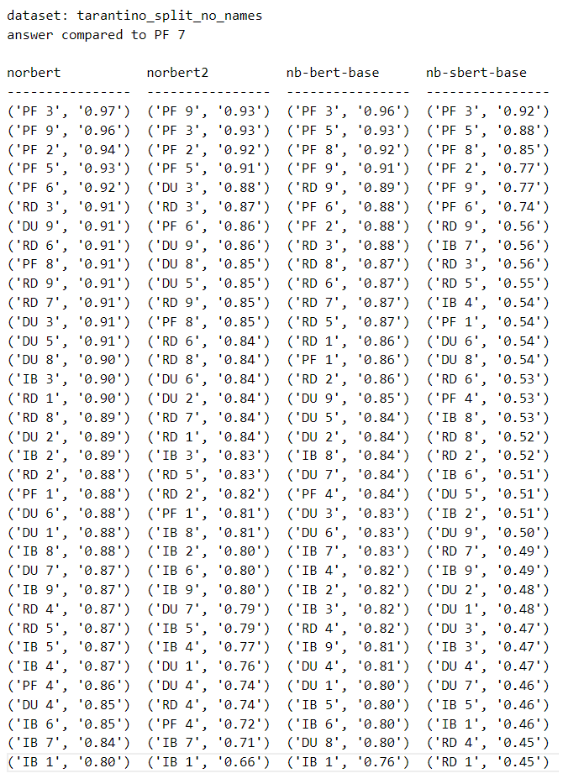
```
dataset: tarantino_split_no_names
answer compared to PF 6

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('PF 7', '0.92')   ('PF 7', '0.86')   ('PF 9', '0.89')   ('PF 3', '0.75')
('PF 9', '0.92')   ('PF 9', '0.86')   ('RD 9', '0.89')   ('PF 4', '0.75')
('PF 3', '0.92')   ('PF 5', '0.86')   ('PF 4', '0.89')   ('PF 9', '0.74')
('PF 4', '0.92')   ('RD 9', '0.85')   ('PF 3', '0.88')   ('PF 7', '0.74')
('RD 9', '0.91')   ('PF 2', '0.85')   ('PF 7', '0.88')   ('PF 2', '0.73')
('PF 2', '0.91')   ('RD 3', '0.84')   ('PF 1', '0.87')   ('PF 5', '0.70')
('PF 8', '0.90')   ('PF 4', '0.84')   ('PF 8', '0.87')   ('PF 8', '0.69')
('PF 1', '0.90')   ('PF 1', '0.83')   ('PF 5', '0.87')   ('PF 1', '0.65')
('RD 3', '0.90')   ('PF 8', '0.83')   ('PF 2', '0.86')   ('RD 9', '0.62')
('DU 8', '0.89')   ('PF 3', '0.83')   ('RD 3', '0.86')   ('IB 7', '0.58')
('PF 5', '0.89')   ('RD 1', '0.81')   ('RD 2', '0.85')   ('RD 5', '0.56')
('DU 3', '0.89')   ('IB 3', '0.81')   ('RD 1', '0.85')   ('IB 6', '0.56')
('IB 3', '0.88')   ('IB 2', '0.81')   ('RD 7', '0.84')   ('DU 8', '0.55')
('DU 9', '0.88')   ('DU 6', '0.80')   ('RD 8', '0.83')   ('RD 3', '0.54')
('RD 6', '0.88')   ('IB 6', '0.80')   ('DU 7', '0.83')   ('DU 6', '0.52')
('IB 2', '0.88')   ('RD 5', '0.80')   ('DU 6', '0.83')   ('DU 7', '0.52')
('DU 6', '0.87')   ('DU 3', '0.80')   ('RD 6', '0.83')   ('RD 7', '0.52')
('RD 4', '0.87')   ('DU 8', '0.80')   ('DU 9', '0.83')   ('RD 4', '0.52')
('RD 2', '0.87')   ('RD 2', '0.80')   ('RD 5', '0.83')   ('IB 4', '0.51')
('RD 1', '0.87')   ('DU 9', '0.80')   ('DU 2', '0.83')   ('IB 9', '0.51')
('RD 7', '0.86')   ('RD 7', '0.80')   ('IB 7', '0.82')   ('IB 2', '0.51')
('IB 9', '0.86')   ('RD 6', '0.80')   ('RD 4', '0.82')   ('RD 2', '0.50')
('DU 7', '0.86')   ('IB 9', '0.79')   ('DU 3', '0.82')   ('RD 6', '0.50')
('IB 8', '0.86')   ('DU 2', '0.79')   ('DU 5', '0.82')   ('RD 8', '0.49')
('DU 5', '0.86')   ('IB 8', '0.79')   ('IB 8', '0.81')   ('DU 2', '0.49')
('IB 7', '0.85')   ('RD 8', '0.79')   ('IB 2', '0.80')   ('DU 9', '0.48')
('RD 8', '0.85')   ('IB 5', '0.77')   ('IB 6', '0.80')   ('IB 5', '0.47')
('IB 4', '0.85')   ('DU 5', '0.76')   ('IB 3', '0.79')   ('DU 3', '0.47')
('IB 5', '0.85')   ('DU 7', '0.76')   ('IB 9', '0.79')   ('DU 5', '0.47')
('DU 2', '0.85')   ('IB 4', '0.76')   ('IB 5', '0.79')   ('IB 1', '0.46')
('RD 5', '0.84')   ('IB 7', '0.73')   ('DU 8', '0.78')   ('IB 8', '0.45')
('IB 6', '0.84')   ('RD 4', '0.71')   ('DU 1', '0.78')   ('RD 1', '0.43')
('DU 1', '0.83')   ('DU 1', '0.70')   ('IB 4', '0.77')   ('IB 3', '0.42')
('DU 4', '0.83')   ('DU 4', '0.70')   ('DU 4', '0.76')   ('DU 1', '0.36')
('IB 1', '0.77')   ('IB 1', '0.65')   ('IB 1', '0.72')   ('DU 4', '0.35')
```

Figure .136: The results from comparison to entry labeled "PF 6".



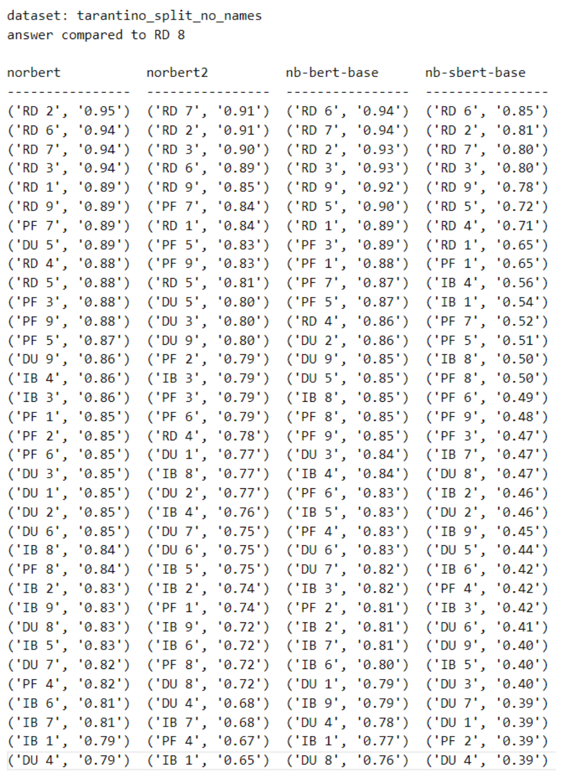
```
dataset: tarantino_split_no_names
answer compared to RD 7

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('RD 6', '0.95')   ('RD 8', '0.91')   ('RD 8', '0.94')   ('RD 3', '0.82')
('RD 3', '0.94')   ('RD 3', '0.90')   ('RD 3', '0.94')   ('RD 8', '0.80')
('RD 8', '0.94')   ('RD 2', '0.88')   ('RD 6', '0.93')   ('RD 5', '0.77')
('RD 2', '0.94')   ('RD 6', '0.88')   ('RD 2', '0.93')   ('RD 6', '0.76')
('RD 1', '0.91')   ('PF 5', '0.85')   ('RD 9', '0.91')   ('RD 9', '0.73')
('PF 7', '0.91')   ('RD 9', '0.84')   ('RD 5', '0.91')   ('RD 2', '0.72')
('PF 5', '0.91')   ('PF 7', '0.84')   ('RD 1', '0.89')   ('RD 1', '0.64')
('RD 9', '0.90')   ('RD 1', '0.84')   ('RD 4', '0.88')   ('RD 4', '0.64')
('DU 5', '0.90')   ('DU 9', '0.84')   ('PF 3', '0.87')   ('PF 1', '0.57')
('RD 5', '0.90')   ('DU 5', '0.83')   ('PF 5', '0.87')   ('PF 5', '0.52')
('PF 9', '0.90')   ('DU 3', '0.83')   ('PF 7', '0.87')   ('PF 6', '0.52')
('DU 9', '0.90')   ('PF 9', '0.83')   ('DU 5', '0.87')   ('PF 7', '0.49')
('PF 3', '0.89')   ('RD 5', '0.81')   ('DU 2', '0.86')   ('IB 4', '0.47')
('RD 4', '0.89')   ('DU 1', '0.81')   ('PF 1', '0.86')   ('DU 8', '0.47')
('DU 3', '0.88')   ('PF 6', '0.80')   ('DU 9', '0.86')   ('PF 3', '0.46')
('PF 2', '0.88')   ('DU 2', '0.79')   ('PF 9', '0.85')   ('IB 1', '0.46')
('DU 1', '0.88')   ('PF 2', '0.79')   ('DU 3', '0.84')   ('PF 8', '0.45')
('DU 8', '0.87')   ('PF 3', '0.79')   ('DU 7', '0.84')   ('DU 2', '0.45')
('IB 4', '0.87')   ('RD 4', '0.78')   ('PF 6', '0.84')   ('DU 5', '0.44')
('IB 3', '0.86')   ('IB 3', '0.78')   ('DU 6', '0.84')   ('PF 9', '0.43')
('PF 6', '0.86')   ('IB 8', '0.78')   ('PF 8', '0.84')   ('DU 9', '0.42')
('PF 1', '0.86')   ('DU 7', '0.77')   ('IB 8', '0.83')   ('PF 2', '0.42')
('DU 6', '0.86')   ('DU 6', '0.76')   ('PF 4', '0.82')   ('DU 3', '0.41')
('DU 2', '0.86')   ('IB 4', '0.75')   ('IB 4', '0.81')   ('IB 7', '0.41')
('IB 9', '0.85')   ('IB 5', '0.75')   ('IB 3', '0.81')   ('DU 6', '0.41')
('IB 2', '0.85')   ('IB 6', '0.74')   ('PF 2', '0.80')   ('PF 4', '0.40')
('IB 8', '0.84')   ('IB 2', '0.74')   ('DU 1', '0.80')   ('DU 1', '0.39')
('IB 5', '0.84')   ('DU 8', '0.74')   ('IB 5', '0.80')   ('DU 7', '0.39')
('PF 8', '0.84')   ('IB 9', '0.73')   ('IB 2', '0.80')   ('IB 9', '0.39')
('DU 7', '0.83')   ('PF 1', '0.73')   ('IB 7', '0.79')   ('IB 2', '0.38')
('IB 7', '0.82')   ('PF 8', '0.71')   ('DU 4', '0.79')   ('IB 8', '0.37')
('PF 4', '0.82')   ('DU 4', '0.71')   ('DU 8', '0.77')   ('DU 4', '0.37')
('DU 4', '0.81')   ('IB 7', '0.69')   ('IB 6', '0.77')   ('IB 6', '0.36')
('IB 6', '0.81')   ('PF 4', '0.66')   ('IB 1', '0.76')   ('IB 5', '0.33')
('IB 1', '0.79')   ('IB 1', '0.64')   ('IB 9', '0.75')   ('IB 3', '0.32')
```

Figure .137: The results from comparison to entry labeled "RD 7".



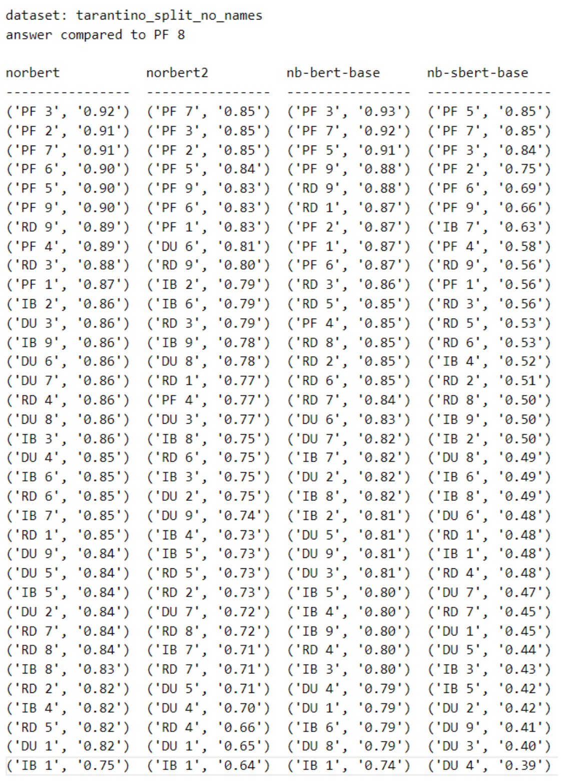
```
dataset: tarantino_split_no_names
answer compared to PF 7

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('PF 3', '0.97')  ('PF 9', '0.93')  ('PF 3', '0.96')  ('PF 3', '0.92')
('PF 9', '0.96')  ('PF 3', '0.93')  ('PF 5', '0.93')  ('PF 5', '0.88')
('PF 2', '0.94')  ('PF 2', '0.92')  ('PF 8', '0.92')  ('PF 8', '0.85')
('PF 5', '0.93')  ('PF 5', '0.91')  ('PF 9', '0.91')  ('PF 2', '0.77')
('PF 6', '0.92')  ('DU 3', '0.88')  ('RD 9', '0.89')  ('PF 9', '0.77')
('RD 3', '0.91')  ('RD 3', '0.87')  ('PF 6', '0.88')  ('PF 6', '0.74')
('DU 9', '0.91')  ('PF 6', '0.86')  ('PF 2', '0.88')  ('RD 9', '0.56')
('RD 6', '0.91')  ('DU 9', '0.86')  ('RD 3', '0.88')  ('IB 7', '0.56')
('PF 8', '0.91')  ('DU 8', '0.85')  ('RD 8', '0.87')  ('RD 3', '0.56')
('RD 9', '0.91')  ('DU 5', '0.85')  ('RD 6', '0.87')  ('RD 5', '0.55')
('RD 7', '0.91')  ('RD 9', '0.85')  ('RD 7', '0.87')  ('IB 4', '0.54')
('DU 3', '0.91')  ('PF 8', '0.85')  ('RD 5', '0.87')  ('PF 1', '0.54')
('DU 5', '0.91')  ('RD 6', '0.84')  ('RD 1', '0.86')  ('DU 6', '0.54')
('DU 8', '0.90')  ('RD 8', '0.84')  ('PF 1', '0.86')  ('DU 8', '0.54')
('IB 3', '0.90')  ('DU 6', '0.84')  ('RD 2', '0.86')  ('RD 6', '0.53')
('RD 1', '0.90')  ('DU 2', '0.84')  ('DU 9', '0.85')  ('PF 4', '0.53')
('RD 8', '0.89')  ('RD 7', '0.84')  ('DU 5', '0.84')  ('IB 8', '0.53')
('DU 2', '0.89')  ('RD 1', '0.84')  ('DU 2', '0.84')  ('RD 8', '0.52')
('IB 2', '0.89')  ('IB 3', '0.83')  ('IB 8', '0.84')  ('RD 2', '0.52')
('RD 2', '0.88')  ('RD 5', '0.83')  ('DU 7', '0.84')  ('IB 6', '0.51')
('PF 1', '0.88')  ('RD 2', '0.82')  ('PF 4', '0.84')  ('DU 5', '0.51')
('DU 6', '0.88')  ('PF 1', '0.81')  ('DU 3', '0.83')  ('IB 2', '0.51')
('DU 1', '0.88')  ('IB 8', '0.81')  ('DU 6', '0.83')  ('DU 9', '0.50')
('IB 8', '0.88')  ('IB 2', '0.80')  ('IB 7', '0.83')  ('RD 7', '0.49')
('DU 7', '0.87')  ('IB 6', '0.80')  ('IB 4', '0.82')  ('IB 9', '0.49')
('IB 9', '0.87')  ('IB 9', '0.80')  ('IB 2', '0.82')  ('DU 2', '0.48')
('RD 4', '0.87')  ('DU 7', '0.79')  ('IB 3', '0.82')  ('DU 1', '0.48')
('RD 5', '0.87')  ('IB 5', '0.79')  ('RD 4', '0.82')  ('DU 3', '0.47')
('IB 5', '0.87')  ('IB 4', '0.77')  ('IB 9', '0.81')  ('IB 3', '0.47')
('IB 4', '0.87')  ('DU 1', '0.76')  ('DU 4', '0.81')  ('DU 4', '0.47')
('PF 4', '0.86')  ('DU 4', '0.74')  ('DU 1', '0.80')  ('DU 7', '0.46')
('DU 4', '0.85')  ('RD 4', '0.74')  ('IB 5', '0.80')  ('IB 5', '0.46')
('IB 6', '0.85')  ('PF 4', '0.72')  ('IB 6', '0.80')  ('IB 1', '0.46')
('IB 7', '0.84')  ('IB 7', '0.71')  ('DU 8', '0.80')  ('RD 4', '0.45')
('IB 1', '0.80')  ('IB 1', '0.66')  ('IB 1', '0.76')  ('RD 1', '0.45')
```

Figure .138: The results from comparison to entry labeled "PF 7".



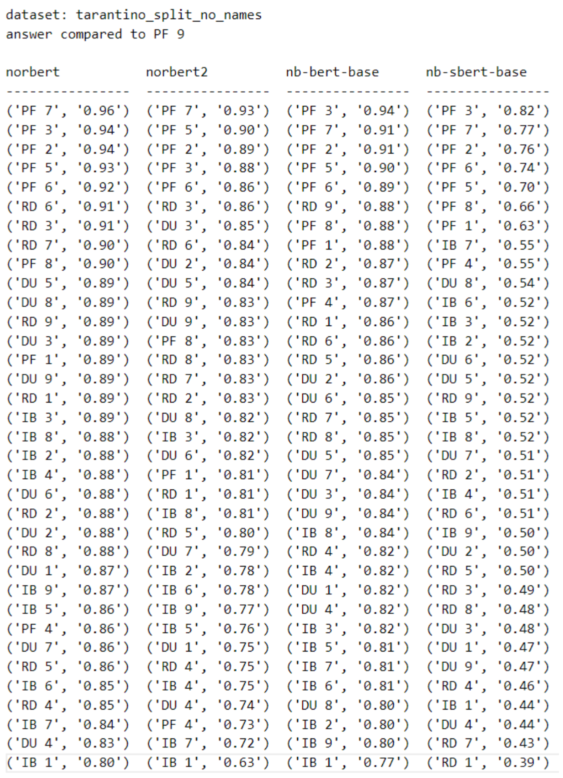
```
dataset: tarantino_split_no_names
answer compared to RD 8

norbert           norbert2          nb-bert-base      nb-sbert-base
----------------  ----------------  ----------------  ----------------
('RD 2', '0.95')  ('RD 7', '0.91')  ('RD 6', '0.94')  ('RD 6', '0.85')
('RD 6', '0.94')  ('RD 2', '0.91')  ('RD 7', '0.94')  ('RD 2', '0.81')
('RD 7', '0.94')  ('RD 3', '0.90')  ('RD 2', '0.93')  ('RD 7', '0.80')
('RD 3', '0.94')  ('RD 6', '0.89')  ('RD 3', '0.93')  ('RD 3', '0.80')
('RD 1', '0.89')  ('RD 9', '0.85')  ('RD 9', '0.92')  ('RD 9', '0.78')
('RD 9', '0.89')  ('PF 7', '0.84')  ('RD 5', '0.90')  ('RD 5', '0.72')
('PF 7', '0.89')  ('RD 1', '0.84')  ('RD 1', '0.89')  ('RD 4', '0.71')
('DU 5', '0.89')  ('PF 5', '0.83')  ('PF 3', '0.89')  ('RD 1', '0.65')
('RD 4', '0.88')  ('PF 9', '0.83')  ('PF 1', '0.88')  ('PF 1', '0.65')
('RD 5', '0.88')  ('RD 5', '0.81')  ('PF 7', '0.87')  ('IB 4', '0.56')
('PF 3', '0.88')  ('DU 5', '0.80')  ('PF 5', '0.87')  ('IB 1', '0.54')
('PF 9', '0.88')  ('DU 3', '0.80')  ('RD 4', '0.86')  ('PF 7', '0.52')
('PF 5', '0.87')  ('DU 9', '0.80')  ('DU 2', '0.86')  ('PF 5', '0.51')
('DU 9', '0.86')  ('PF 2', '0.79')  ('DU 9', '0.85')  ('IB 8', '0.50')
('IB 4', '0.86')  ('IB 3', '0.79')  ('DU 5', '0.85')  ('PF 8', '0.50')
('IB 3', '0.86')  ('PF 3', '0.79')  ('IB 8', '0.85')  ('PF 6', '0.49')
('PF 1', '0.85')  ('PF 6', '0.79')  ('PF 8', '0.85')  ('PF 9', '0.48')
('PF 2', '0.85')  ('RD 4', '0.78')  ('PF 9', '0.85')  ('PF 3', '0.47')
('PF 6', '0.85')  ('DU 1', '0.77')  ('DU 3', '0.84')  ('IB 7', '0.47')
('DU 3', '0.85')  ('IB 8', '0.77')  ('IB 4', '0.84')  ('DU 8', '0.47')
('DU 1', '0.85')  ('DU 2', '0.77')  ('PF 6', '0.83')  ('IB 2', '0.46')
('DU 2', '0.85')  ('IB 4', '0.76')  ('IB 5', '0.83')  ('DU 2', '0.46')
('DU 6', '0.85')  ('DU 7', '0.75')  ('PF 4', '0.83')  ('IB 9', '0.45')
('IB 8', '0.84')  ('DU 6', '0.75')  ('DU 6', '0.83')  ('DU 5', '0.44')
('PF 8', '0.84')  ('IB 5', '0.75')  ('DU 7', '0.82')  ('IB 6', '0.42')
('IB 2', '0.83')  ('IB 2', '0.74')  ('IB 3', '0.82')  ('PF 4', '0.42')
('IB 9', '0.83')  ('PF 1', '0.74')  ('PF 2', '0.81')  ('IB 3', '0.42')
('DU 8', '0.83')  ('IB 9', '0.72')  ('IB 2', '0.81')  ('DU 6', '0.41')
('IB 5', '0.83')  ('IB 6', '0.72')  ('IB 7', '0.81')  ('DU 9', '0.40')
('DU 7', '0.82')  ('PF 8', '0.72')  ('IB 6', '0.80')  ('IB 5', '0.40')
('PF 4', '0.82')  ('DU 8', '0.72')  ('DU 1', '0.79')  ('DU 3', '0.40')
('IB 6', '0.81')  ('DU 4', '0.68')  ('IB 9', '0.79')  ('DU 7', '0.39')
('IB 7', '0.81')  ('IB 7', '0.68')  ('DU 4', '0.78')  ('DU 1', '0.39')
('IB 1', '0.79')  ('PF 4', '0.67')  ('IB 1', '0.77')  ('PF 2', '0.39')
('DU 4', '0.79')  ('IB 1', '0.65')  ('DU 8', '0.76')  ('DU 4', '0.39')
```

Figure .139: The results from comparison to entry labeled "RD 8".



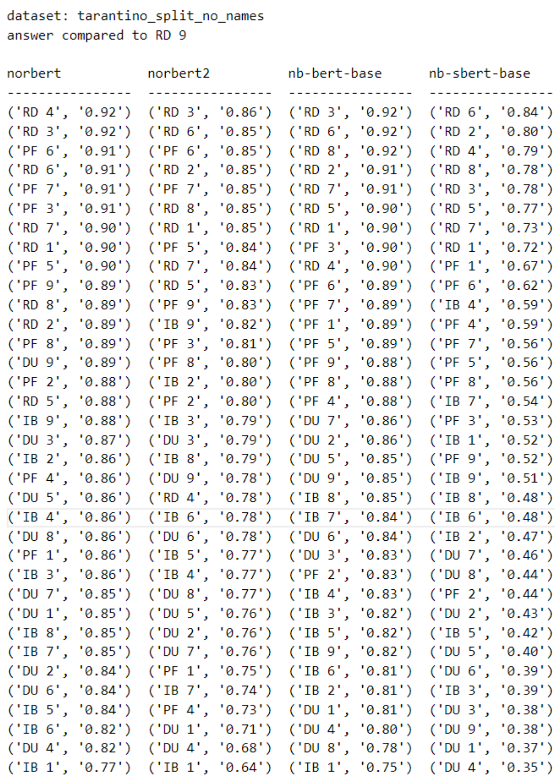
```
dataset: tarantino_split_no_names
answer compared to PF 8

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('PF 3', '0.92')   ('PF 7', '0.85')   ('PF 3', '0.93')   ('PF 5', '0.85')
('PF 2', '0.91')   ('PF 3', '0.85')   ('PF 7', '0.92')   ('PF 7', '0.85')
('PF 7', '0.91')   ('PF 2', '0.85')   ('PF 5', '0.91')   ('PF 3', '0.84')
('PF 6', '0.90')   ('PF 5', '0.84')   ('PF 9', '0.88')   ('PF 2', '0.75')
('PF 5', '0.90')   ('PF 9', '0.83')   ('RD 9', '0.88')   ('PF 6', '0.69')
('PF 9', '0.90')   ('PF 6', '0.83')   ('RD 1', '0.87')   ('PF 9', '0.66')
('RD 9', '0.89')   ('PF 1', '0.83')   ('PF 2', '0.87')   ('IB 7', '0.63')
('PF 4', '0.89')   ('DU 6', '0.81')   ('PF 1', '0.87')   ('PF 4', '0.58')
('RD 3', '0.88')   ('RD 9', '0.80')   ('PF 6', '0.87')   ('RD 9', '0.56')
('PF 1', '0.87')   ('IB 2', '0.79')   ('RD 3', '0.86')   ('PF 1', '0.56')
('IB 2', '0.86')   ('IB 6', '0.79')   ('RD 5', '0.85')   ('RD 3', '0.56')
('DU 3', '0.86')   ('RD 3', '0.79')   ('PF 4', '0.85')   ('RD 5', '0.53')
('IB 9', '0.86')   ('IB 9', '0.78')   ('RD 8', '0.85')   ('RD 6', '0.53')
('DU 6', '0.86')   ('DU 8', '0.78')   ('RD 2', '0.85')   ('IB 4', '0.52')
('DU 7', '0.86')   ('RD 1', '0.77')   ('RD 6', '0.85')   ('RD 2', '0.51')
('RD 4', '0.86')   ('PF 4', '0.77')   ('RD 7', '0.84')   ('RD 8', '0.50')
('DU 8', '0.86')   ('DU 3', '0.77')   ('DU 6', '0.83')   ('IB 9', '0.50')
('IB 3', '0.86')   ('IB 8', '0.75')   ('DU 7', '0.82')   ('IB 2', '0.50')
('DU 4', '0.85')   ('RD 6', '0.75')   ('IB 7', '0.82')   ('DU 8', '0.49')
('IB 6', '0.85')   ('IB 3', '0.75')   ('DU 2', '0.82')   ('IB 6', '0.49')
('RD 6', '0.85')   ('DU 2', '0.75')   ('IB 8', '0.82')   ('IB 8', '0.49')
('IB 7', '0.85')   ('DU 9', '0.74')   ('IB 2', '0.81')   ('DU 6', '0.48')
('RD 1', '0.85')   ('IB 4', '0.73')   ('DU 5', '0.81')   ('RD 1', '0.48')
('DU 9', '0.84')   ('IB 5', '0.73')   ('DU 9', '0.81')   ('IB 1', '0.48')
('DU 5', '0.84')   ('RD 5', '0.73')   ('DU 3', '0.81')   ('RD 4', '0.48')
('IB 5', '0.84')   ('RD 2', '0.73')   ('IB 5', '0.80')   ('DU 7', '0.47')
('DU 2', '0.84')   ('DU 7', '0.72')   ('IB 4', '0.80')   ('RD 7', '0.45')
('RD 7', '0.84')   ('RD 8', '0.72')   ('IB 9', '0.80')   ('DU 1', '0.45')
('RD 8', '0.84')   ('IB 7', '0.71')   ('RD 4', '0.80')   ('DU 5', '0.44')
('IB 8', '0.83')   ('RD 7', '0.71')   ('IB 3', '0.80')   ('IB 3', '0.43')
('RD 2', '0.82')   ('DU 5', '0.71')   ('DU 4', '0.79')   ('IB 5', '0.42')
('IB 4', '0.82')   ('DU 4', '0.70')   ('DU 1', '0.79')   ('DU 2', '0.42')
('RD 5', '0.82')   ('RD 4', '0.66')   ('IB 6', '0.79')   ('DU 9', '0.41')
('DU 1', '0.82')   ('DU 1', '0.65')   ('DU 8', '0.79')   ('DU 3', '0.40')
('IB 1', '0.75')   ('IB 1', '0.64')   ('IB 1', '0.74')   ('DU 4', '0.39')
```

Figure .140: The results from comparison to entry labeled "PF 8".



```
dataset: tarantino_split_no_names
answer compared to PF 9

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('PF 7', '0.96')   ('PF 7', '0.93')   ('PF 3', '0.94')   ('PF 3', '0.82')
('PF 3', '0.94')   ('PF 5', '0.90')   ('PF 7', '0.91')   ('PF 7', '0.77')
('PF 2', '0.94')   ('PF 2', '0.89')   ('PF 2', '0.91')   ('PF 2', '0.76')
('PF 5', '0.93')   ('PF 3', '0.88')   ('PF 5', '0.90')   ('PF 6', '0.74')
('PF 6', '0.92')   ('PF 6', '0.86')   ('PF 6', '0.89')   ('PF 5', '0.70')
('RD 6', '0.91')   ('RD 3', '0.86')   ('RD 9', '0.88')   ('PF 8', '0.66')
('RD 3', '0.91')   ('DU 3', '0.85')   ('PF 8', '0.88')   ('PF 1', '0.63')
('RD 7', '0.90')   ('RD 6', '0.84')   ('PF 1', '0.88')   ('IB 7', '0.55')
('PF 8', '0.90')   ('DU 2', '0.84')   ('RD 2', '0.87')   ('PF 4', '0.55')
('DU 5', '0.89')   ('DU 5', '0.84')   ('RD 3', '0.87')   ('DU 8', '0.54')
('DU 8', '0.89')   ('RD 9', '0.83')   ('PF 4', '0.87')   ('IB 6', '0.52')
('RD 9', '0.89')   ('DU 9', '0.83')   ('RD 1', '0.86')   ('IB 3', '0.52')
('DU 3', '0.89')   ('PF 8', '0.83')   ('RD 6', '0.86')   ('IB 2', '0.52')
('PF 1', '0.89')   ('RD 8', '0.83')   ('RD 5', '0.86')   ('DU 6', '0.52')
('DU 9', '0.89')   ('RD 7', '0.83')   ('DU 2', '0.86')   ('DU 5', '0.52')
('RD 1', '0.89')   ('RD 2', '0.83')   ('DU 6', '0.85')   ('RD 9', '0.52')
('IB 3', '0.89')   ('DU 8', '0.82')   ('RD 7', '0.85')   ('IB 5', '0.52')
('IB 8', '0.88')   ('IB 3', '0.82')   ('RD 8', '0.85')   ('IB 8', '0.52')
('IB 2', '0.88')   ('DU 6', '0.82')   ('DU 5', '0.85')   ('DU 7', '0.51')
('IB 4', '0.88')   ('PF 1', '0.81')   ('DU 7', '0.84')   ('RD 2', '0.51')
('DU 6', '0.88')   ('RD 1', '0.81')   ('DU 3', '0.84')   ('IB 4', '0.51')
('RD 2', '0.88')   ('IB 8', '0.81')   ('DU 9', '0.84')   ('RD 6', '0.51')
('DU 2', '0.88')   ('RD 5', '0.80')   ('IB 8', '0.84')   ('IB 9', '0.50')
('RD 8', '0.88')   ('DU 7', '0.79')   ('RD 4', '0.82')   ('DU 2', '0.50')
('DU 1', '0.87')   ('IB 2', '0.78')   ('IB 4', '0.82')   ('RD 5', '0.50')
('IB 9', '0.87')   ('IB 6', '0.78')   ('DU 1', '0.82')   ('RD 3', '0.49')
('IB 5', '0.86')   ('IB 9', '0.77')   ('DU 4', '0.82')   ('RD 8', '0.48')
('PF 4', '0.86')   ('IB 5', '0.76')   ('IB 3', '0.82')   ('DU 3', '0.48')
('DU 7', '0.86')   ('DU 1', '0.75')   ('IB 5', '0.81')   ('DU 1', '0.47')
('RD 5', '0.86')   ('RD 4', '0.75')   ('IB 7', '0.81')   ('DU 9', '0.47')
('IB 6', '0.85')   ('IB 4', '0.75')   ('IB 6', '0.81')   ('RD 4', '0.46')
('RD 4', '0.85')   ('DU 4', '0.74')   ('DU 8', '0.80')   ('IB 1', '0.44')
('IB 7', '0.84')   ('PF 4', '0.73')   ('IB 2', '0.80')   ('DU 4', '0.44')
('DU 4', '0.83')   ('IB 7', '0.72')   ('IB 9', '0.80')   ('RD 7', '0.43')
('IB 1', '0.80')   ('IB 1', '0.63')   ('IB 1', '0.77')   ('RD 1', '0.39')
```

Figure .141: The results from comparison to entry labeled "PF 9".



```
dataset: tarantino_split_no_names
answer compared to RD 9

norbert            norbert2           nb-bert-base       nb-sbert-base
----------------   ----------------   ----------------   ----------------
('RD 4', '0.92')   ('RD 3', '0.86')   ('RD 3', '0.92')   ('RD 6', '0.84')
('RD 3', '0.92')   ('RD 6', '0.85')   ('RD 6', '0.92')   ('RD 2', '0.80')
('PF 6', '0.91')   ('PF 6', '0.85')   ('RD 8', '0.92')   ('RD 4', '0.79')
('RD 6', '0.91')   ('RD 2', '0.85')   ('RD 2', '0.91')   ('RD 8', '0.78')
('PF 7', '0.91')   ('PF 7', '0.85')   ('RD 7', '0.91')   ('RD 3', '0.78')
('PF 3', '0.91')   ('RD 8', '0.85')   ('RD 5', '0.90')   ('RD 5', '0.77')
('RD 7', '0.90')   ('RD 1', '0.85')   ('RD 1', '0.90')   ('RD 7', '0.73')
('RD 1', '0.90')   ('PF 5', '0.84')   ('PF 3', '0.90')   ('RD 1', '0.72')
('PF 5', '0.90')   ('RD 7', '0.84')   ('RD 4', '0.90')   ('PF 1', '0.67')
('PF 9', '0.89')   ('RD 5', '0.83')   ('PF 6', '0.89')   ('PF 6', '0.62')
('RD 8', '0.89')   ('PF 9', '0.83')   ('PF 7', '0.89')   ('IB 4', '0.59')
('RD 2', '0.89')   ('IB 9', '0.82')   ('PF 1', '0.89')   ('PF 4', '0.59')
('PF 8', '0.89')   ('PF 3', '0.81')   ('PF 5', '0.89')   ('PF 7', '0.56')
('DU 9', '0.89')   ('PF 8', '0.80')   ('PF 9', '0.88')   ('PF 5', '0.56')
('PF 2', '0.88')   ('IB 2', '0.80')   ('PF 8', '0.88')   ('PF 8', '0.56')
('RD 5', '0.88')   ('PF 2', '0.80')   ('PF 4', '0.88')   ('IB 7', '0.54')
('IB 9', '0.88')   ('IB 3', '0.79')   ('DU 7', '0.86')   ('PF 3', '0.53')
('DU 3', '0.87')   ('DU 3', '0.79')   ('DU 2', '0.86')   ('IB 1', '0.52')
('IB 2', '0.86')   ('IB 8', '0.79')   ('DU 5', '0.85')   ('PF 9', '0.52')
('PF 4', '0.86')   ('DU 9', '0.78')   ('DU 9', '0.85')   ('IB 9', '0.51')
('DU 5', '0.86')   ('RD 4', '0.78')   ('IB 8', '0.85')   ('IB 8', '0.48')
('IB 4', '0.86')   ('IB 6', '0.78')   ('IB 7', '0.84')   ('IB 6', '0.48')
('DU 8', '0.86')   ('DU 6', '0.78')   ('DU 6', '0.84')   ('IB 2', '0.47')
('PF 1', '0.86')   ('IB 5', '0.77')   ('DU 3', '0.83')   ('DU 7', '0.46')
('IB 3', '0.86')   ('IB 4', '0.77')   ('PF 2', '0.83')   ('DU 8', '0.44')
('DU 7', '0.85')   ('DU 8', '0.77')   ('IB 4', '0.83')   ('PF 2', '0.44')
('DU 1', '0.85')   ('DU 5', '0.76')   ('IB 3', '0.82')   ('DU 2', '0.43')
('IB 8', '0.85')   ('DU 2', '0.76')   ('IB 5', '0.82')   ('IB 5', '0.42')
('IB 7', '0.85')   ('DU 7', '0.76')   ('IB 9', '0.82')   ('DU 5', '0.40')
('DU 2', '0.84')   ('PF 1', '0.75')   ('IB 6', '0.81')   ('DU 6', '0.39')
('DU 6', '0.84')   ('IB 7', '0.74')   ('IB 2', '0.81')   ('IB 3', '0.39')
('IB 5', '0.84')   ('PF 4', '0.73')   ('DU 1', '0.81')   ('DU 3', '0.38')
('IB 6', '0.82')   ('DU 1', '0.71')   ('DU 4', '0.80')   ('DU 9', '0.38')
('DU 4', '0.82')   ('DU 4', '0.68')   ('DU 8', '0.78')   ('DU 1', '0.37')
('IB 1', '0.77')   ('IB 1', '0.64')   ('IB 1', '0.75')   ('DU 4', '0.35')
```

Figure .142: The results from comparison to entry labeled "RD 9".