Master's thesis

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

Pernille Andresen Klevstuen

Assisting efficient and fair grading with information retrieval and text mining techniques

Master's thesis in Computer Science
Supervisor: Trond Aalberg
February 2022

NTNU
Norwegian University of
Science and Technology

Pernille Andresen Klevstuen

Assisting efficient and fair grading with information retrieval and text mining techniques

Master's thesis in Computer Science
Supervisor: Trond Aalberg
February 2022

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 process of grading answers to exams and other tests is a manual and often time-consuming process and demands a high level of concentration over a long period of time. This can cause the process to be subject to fluctuations, and discrepancies between grades given to similar answers may occur. As a result, the grading process can be experienced as unjust for the students and tiresome for the sensors. To address this challenge, it has been proposed to use computers to automatically evaluate and grade student answers.

The research of automatic grading has been explored for decades. Researchers have focused on automatically assessing answers based on different criteria. For multiple-choice questions, the problem of automatic grading is considered solved. However, natural language answers remain a challenge. From the field of automatic grading of text-based answers, two dominating research areas have emerged. Automatic Grading of Essays (AGE) evaluates essays based on writing style and language. Automatic Short-Answer Grading (ASAG) focuses on evaluating the content of short-answer. Most studies have been conducted using answers and essays written in English. Automatic grading of answers evaluated based on content has not been tried using Norwegian answers.

The goal of this thesis is to investigate the use of information retrieval and text mining techniques to evaluate the content of Norwegian exam answers within the field of Computer Science. Answers from three earlier exams in the course IT2810 Web Development at the Norwegian University of Science and Technology are used to explore whether such techniques can contribute to an efficient and fair grading process.

To achieve this goal, a literature review related to automatic grading is conducted. Furthermore, an analysis of the exam datasets is performed to discover challenges and possibilities for using information retrieval and text mining techniques in the assessment of student texts. Three experiments are conducted, focusing on evaluating answers based on term usage. Findings from this thesis suggest that such techniques should be further investigated to ensure an efficient and fair grading process of Norwegian exams. Applications of these techniques are especially promising in terms of providing guidance to the sensor and detecting answers that may have received the wrong grade.

Sammendrag

Sensurering av eksamensbesvarelser er en manuell og ofte tidkrevende prosess som krever høy konsentrasjon over lengre tid. Prosessen kan dermed bli utsatt for feil og to i utgangspunktet like gode svar, kan bli tildelt ulik karakter. Dette kan medføre at sensurprosessen oppleves urettferdig for studenter og slitsom for sensorer. For å overkomme denne utfordringen har det blitt foreslått å bruke datamaskiner til å automatisk evaluere og sensurere eksamensbesvarelser.

Automatisk sensurering (automatic grading) har blitt forsket på i flere tiår. Forskere har fokusert på å automatisk evaluere svar basert på ulike kriterier. Automatisk sensur av flersvarsoppgaver regnes i dag som løst, derimot er automatisk sensurering av tekstsvar en vedvarende utfordring. Sensurering av tekstsvar har utviklet seg i to dominerende retninger. Automatisk sensurering av essays (Automatic Grading of Essays) tar sikte på å vurdere stiler basert på språk. Automatisk sensurering av kortsvar (Automatic Short-Answer Grading) fokuserer på å evaluere innholdet i tekstsvar. De fleste studier gjennomført innen begge feltene har brukt Engelske svar og stiler. Ingen tidligere forskning er gjort innen automatisk sensurering av norske svar hvor det innholdet i svarene evalueres.

Denne masteroppgaven har som mål å bruke teknikker fra informasjonsgjenfinning (information retrieval) og tekstdatautvikling (text mining) til å evaluere innholdet av norske eksamensbesvarelser innen feltet informatikk (Computer Science). Svar fra tre tidligere eksamener i faget IT2810 Webutvikling ved Norges teknisk-naturvitenskapelige universitet brukes for å undersøke om slike teknikker kan bidra til en effektiv og rettferdig sensurprosess.

For å nå målet vil det bli gjennomført en litteraturstudie knyttet til automatisk sensurering. Videre vil det bli utført en analyse av eksamenssettene for å avdekke utfordringer og muligheter ved å bruke informasjonsgjenfinning og tekstdatautvikling til å evaluere eksamensbesvarelser. Tre eksperimenter vil bli gjennomført. De vil forsøke å evaluere svar basert på ordbruken i besvarelsene. Funnene fra studien tyder på at det er potensiale for å benytte disse teknikkene for å sikre en effektiv og rettferdig sensurprosess av norske eksamener. Spesielt viser disse teknikkene seg lovende for å kunne gi veiledning under sensurprosessen og for å oppdage svar hvor potensiell feil karakter har blitt satt.

Preface

This Master's Thesis was written by Pernille Andresen Klevstuen in the Fall of 2021 and the beginning of 2022, as a part of the Master of Science degree in Computer Science at the Norwegian University of Science and Technology. The project was supervised by Trond Aalberg.

I would like to thank my supervisor, Trond Aalberg, for his excellent guidance throughout the work on this thesis. Furthermore, I would like to thank my parents, Trond and Guri, who have supported me through my work and my studies, and Eirik for his help and encouragement of this thesis. I would also like to mention Bedkom, Abakus, Studentersamfundet, ISFiT, my flatmates, Harry Potter, Bymarka, my siblings and, last but not least, my friends. They have made my study years the best five and a half years of my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The number of students in higher education is rapidly growing. As a consequence, the workload dedicated to assessment has increased at all educational levels. The increased workload also comes with additional costs.

In recent years, schools and universities have undergone structural changes, and computers are becoming more and more common in education. Several schools and universities have now replaced, or are in the process of replacing, handwritten assignments with digital deliveries. With the use of electronic learning also comes the need for generating electronic assessments. This has motivated research into computerised assistance in grading and assessment [1].

This thesis is a study in the field of automatic grading focusing on the application of text mining and information retrieval techniques on text-based Norwegian exams in Computer Science. This chapter introduces the scope of this thesis. Section 1.1 provides the background and motivation. The research goal and questions are presented in Section 1.2 and Section 1.3 explains how the work will be conducted. Section 1.4 lists this thesis' contributions, while Section 1.5 provides the outline for the remainder of this thesis.

1.1 Background and Motivation

The process of grading text answers to exam questions is time-consuming, requiring much effort from sensors with many answers to evaluate. The process demands a high level of concentration over a long period of time, making the process subject to fluctuations in the assessment of individual answers. Discrepancies between evaluation of similar responses might occur as sensors experience emotional factors hard to mitigate. In addition, grading exams can be challenging for many sensors and often depend on personal perception, interpretation and understanding. This can lead to answers of similar quality being graded differently.

Furthermore, the order in which answers are assessed might affect the grading as well. An average quality answer might receive a high score if graded after several

low-quality answers. Likewise, the same average answer could receive a low score if graded after several high-quality answers. Two equally good answers can also receive different scores if one is graded at the beginning of the process and the other at the end [2].

Such unfortunate outcomes can be perceived as unjust by students. Every year, multiple students deliver formal complaints asking for their exams to be reevaluated. Numbers from 2018 gathered from 10 different educational institutions in Norway showed that over 15,000 complaints were received during one semester and that these numbers were increasing [3]. A study conducted by the Norwegian newspaper VG, showed that 39% of the students who in 2020 asked for their exam to be reevaluated, received a new grade [4].

These consequences, which can be experienced as tiresome for the sensor and unjust for the students, can be reduced by utilising computerised assistance. Dating back to the early work of Page [5] during the 1960s, researchers have contributed to the field of automatic grading. While several studies have developed methods that are stated to perform relatively well, few approaches are compared using the same datasets [6]. A commonality for most of these studies is that the researchers collect their own data due to the lack of a standard corpus. Additionally, few researchers provide a detailed description of their datasets, making it hard to compare the general success of their approach. In fact, from the available dataset descriptions, it is clear that they differ in several ways; different exam topics, languages, and answer lengths have been used.

There are numerous additional issues related to automatic grading and assessment of exam answers. One of the main challenges identified in this thesis was the inconsistent vocabulary found in student answers. In an exam setting, students are under time pressure. As a consequence, answers often contain inconsistent and incorrect language. This poses a challenge and preprocessing student answers has by Burrows et al. [6] been characterised as the most difficult part of the automatic grading process. Furthermore, exam questions tend to be renewed for each exam, limiting the amount of test data that is gathered for each question and used during development of new systems. Though some questions are reused in new exams, requirements for each grade might change.

Systems that automatically predict student grades have been criticised as no solution thus far guarantees correct results. Replacing human sensors with machines that cannot deliver optimally can lead to more student complaints and a lack of trust in the system. Thus, some researchers have focused on developing semi-automatic systems. These systems do not intend to provide a grade to each answer but rather to give guidance to the sensor during the grading process.

Though the literature on automatic grading is vast, most studies are conducted on English answers, which is inadequate for other languages such as Norwegian. This highly motivates research in computerised assistance for Norwegian exam evaluation. This thesis is the first study on how to aid in a Norwegian grading process where answers are evaluated based on content. Furthermore, research conducted on automatically evaluating answers based on their content mainly deals with short answers, ranging from one phrase to a paragraph. This differs from the answers

used in this thesis, where the average answer length is 189 words. Moreover, by providing a detailed description of the datasets, the work can be further built upon in the future.

1.2 Research Goal and Questions

The goal of this thesis is to study the use of information retrieval and text mining techniques to ease the process of evaluating Norwegian exams in the field of Computer Science and if this can be done in a manner that is fair and just for the students. To meet the goal, a literature review is first conducted. Secondly, a thorough analysis of three datasets consisting of student answers is performed. The datasets are obtained from exams held at the Norwegian University of Science and Technology in the course IT2810 Web Development. Lastly, three experiments are conducted using information retrieval and text mining techniques.

The following three research questions have been formulated and are presented below:

RQ-1 *What is state-of-the-art within automatic grading of text-based answers?*

RQ-2 *What are the possibilities and challenges for applying text mining and information retrieval techniques to Norwegian text-based exam questions in Computer Science?*

RQ-3 *Can information retrieval and text mining techniques be employed in order to assist an efficient and fair grading process?*

1.3 Method

Datasets from three exams were given to explore different techniques and approaches that might help meet the research goal. This section presents the methods and approaches that will be used to answer the research questions. For each question, this section presents how the question will be approached in order to provide a satisfying answer. Thus, the section outlines how the work on this thesis will be structured.

RQ-1: What is state-of-the-art within automatic grading of text-based answers?

The first step in this thesis is to conduct a qualitative analysis of relevant research within the field of automatic grading, specifically that of automatic grading and assessment of natural language text. This qualitative analysis is presented as a literature review and is necessary to gain theoretical insight into earlier work and methods and how existing solutions handle this problem today. The literature review

will give the reader an overview of existing solutions and approaches. Addressing this research question will also help in gaining knowledge to approach the dataset analysis and determine experiments that should be examined to meet the research goal.

RQ-2: What are the possibilities and challenges for applying text mining and information retrieval techniques to Norwegian text-based exam questions in Computer Science?

The next step is to perform a quantitative and qualitative analysis of the datasets. The goal is to identify both challenges and opportunities within the field of automatic grading. The dataset analysis will look into both challenges related to the specific datasets and more general challenges that may apply to the assessment of other Norwegian exam answers. Because the author did not find any earlier work on challenges related to automatic grading and assessment of Norwegian exam sets, a thorough analysis is required. Chapter 4 will introduce both a summary of the datasets, as well as challenges and important features. Section 5.1 discusses and suggests how different techniques from information retrieval and text mining can be employed.

RQ-3: Can information retrieval and text mining techniques be employed in order to assist an efficient and fair grading process?

To approach this question, two aspects must be considered: what techniques can be tested and how can these techniques be easily employed by a sensor later on. Once the initial cleaning and analysis is conducted, experiments must be defined to be further examined. The aim will be to exploit existing off-the-shelf techniques and see if any turns out to be promising in further aiding the process. Chapter 5 will define, conduct and present the results from the experiments. Chapter 6 explores and defines different use cases where these techniques could be helpful to ensure a more efficient and fair grading process.

1.4 Contributions

No earlier work has been identified in the field of automatic grading using Norwegian exam answers evaluated based on content. Furthermore, there is not much research on automatic assessments of long answers when the focus is on evaluating content. Thus, the work conducted in this thesis contributes to further research on automatic grading and assessment of text-based exams.

The main contributions of this thesis are listed below:

1. An overview of current approaches and systems within automatic grading
2. A descriptive approach for how to clean exam datasets retrieved from Inspera

3. A detailed analysis of challenges that might occur when dealing with Norwegian exam answers, specifically those within the field of Computer Science
4. Attributes that characterise answers of different qualities that can help identify and separate student answers from one another
5. Methods for providing guidance to the sensor during the grading process
6. A new use case within the field of automatic grading
7. Methods for detecting answers with possible incorrect grade
8. A first approach of using Norwegian answers in an automatic assessment scheme where the answers are evaluated based on content

1.5 Thesis structure

The remainder of this thesis is organised as follows:

Chapter 2 introduces and describes relevant methods, technologies and theories used in related work and the work of this thesis.

Chapter 3 provides a detailed overview of related work in the field of automatic grading, including literature related to automatic grading and assessment of short-answer questions and essays. Further, the chapter also includes the benefits and challenges of automatic grading.

Chapter 4 provides an analysis of the datasets. First, a summary of the datasets is presented, giving the reader an overview of the available data. Then the dataset cleaning processes and related challenges are presented. A suggestion for how to clean Inspera datasets is provided to the reader. The chapter also includes an analysis of the grade distributions and answer lengths before challenges related to automatic grading are discussed.

Chapter 5 suggests and presents the three experiments conducted in this thesis. Different information retrieval and text mining techniques are applied to the datasets to investigate if some yield results that can indicate if they are suited for automatic grading or guidance.

Chapter 6 discusses the findings from this thesis and use cases that information retrieval and text mining techniques can be applied to, including guidance and anomaly detection. Suggestions for utilising the techniques in a grading process are also given to the reader. In the end, the research questions are revisited.

Chapter 7 concludes the thesis and presents suggestions for further work in order to improve the results in the future.

The thesis includes four appendices presenting the exam questions, some statistics about the datasets and a list of Norwegian stopwords removed from the answers.

Chapter 2

Background Theory

This chapter presents the theory used and discussed throughout this thesis. Section 2.1 gives an introduction to the fields of text mining and information retrieval. The following section, Section 2.2, describes text processing and representation techniques discussed and applied to clean and process the datasets. The process of ranking documents and an approach for measuring similarity are presented in Section 2.3. Next, machine learning and learning algorithms are explained, along with models frequently used in automatic grading. Evaluation metrics used to evaluate the performance of machine learning algorithms and ranking systems will also be presented. In the end, appropriate implementation tools and libraries used in this project are listed. The reader is explained the necessary concepts and technologies, though is expected to have some basic understanding of computer science. The background theory presented in this chapter is mainly based on the book *Modern Information Retrieval, the concepts and technology behind search* by Baeza-Yates et al. [7].

2.1 Text Mining and Information Retrieval

This section introduces some key concepts in text mining and information retrieval that will be used in the following sections and throughout the thesis.

Text Mining

Text Mining is the practice of analysing collections of textual data to capture concepts, trends and hidden relationships. The process is conducted by transforming unstructured text into a structured format to discover new insight and meaningful patterns [8].

Text mining is a vast area, typically including tasks as document classification, document clustering, and information extraction [8]. However, the unstructured data must be processed before any useful information can be retrieved. The process of transforming unstructured text into a structured format is necessary in order to

make text mining methods work [9]. In the most fundamental text mining model, a word is represented as either present or absent in a document. Once the text is processed, text mining algorithms can be applied to derive insights [8].

Text mining can be used to solve a number of problems, but the primary focus and most widely studied is that of classification and prediction. Given a sample of labeled past experiences, the objective is to assigning a correct label to new and unseen text. The concept of classifying text by assigning a label can also be extended to cases where we do not have any labels. Thus, the objective becomes to organise the data so that labels can be made afterwards. This process is referred to as clustering. An essential ingredient to both these tasks is measuring similarities between texts or documents. Measuring similarity is fundamental to most document analyses, especially information retrieval [9].

Information Retrieval

Information Retrieval (IR) is by many considered a closely related field to text mining. The general task of information retrieval is presented in Figure 2.1. The primary goal is to retrieve all documents from a collection of documents relevant to some user query whilst at the same time retrieving as few non-relevant documents as possible [7]. The query is often composed of words that can help identify the stored, relevant documents. This process can be generalised to a document matcher. Instead of inputting a query with a few words, a complete document is presented as a collection of words. The document is then matched with all documents in the collection, retrieving the documents with the best match [9].

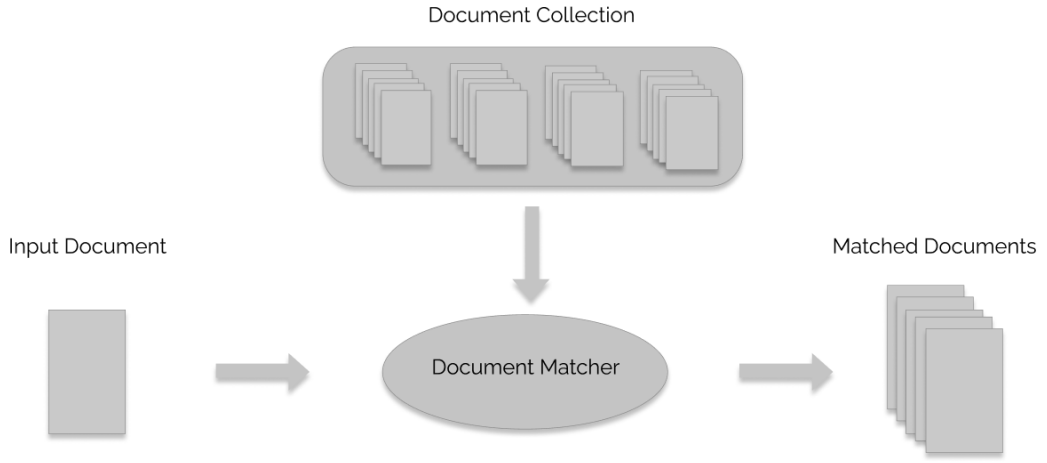


Figure 2.1: Process of retrieving matched documents.

Selecting the most relevant documents is based on calculating the similarity between the input query or document against all other documents. The similarity score is used to rank documents from most similar to least similar. The ranking is then employed in order to retrieve only the most relevant documents. However, before comparison, the query and the documents must first be transformed into a vector of

values that make measuring similarity possible. This process often includes preprocessing the words in the documents by using techniques such as *stopword removal*, *tokenization*, and *stemming* [7]. This process is further elaborated in the following section.

2.2 Document Preprocessing and Representation

Document preprocessing and representation are important steps in text mining and information retrieval and are integral to automatic grading. There are several steps that can be conducted during the preprocessing step, but among the most common are *stopword removal*, *tokenization* and *stemming* or *lemmatization* [7]. These techniques are also a vital part of what is known as *Natural Language Processing* (NLP); the process of enabling computers to understand unstructured language and interpret ambiguous words as humans can. By utilising these techniques, text becomes mathematically commutable so that computer systems and models can make sense of it [10]. This section presents text preprocessing and representation methods often used in automatic grading.

2.2.1 Textual Preprocessing

For a machine to work with natural language, the text must first be preprocessed. This is an essential step in text mining and information retrieval, especially when dealing with user-generated content such as exam answers written by students. There are several steps that can be included during preprocessing. A typical preprocessing pipeline is illustrated in Figure 2.2. This example includes some of the most common steps in the pipeline. Not every step is necessarily included every time, and often, more steps are a part of the process.

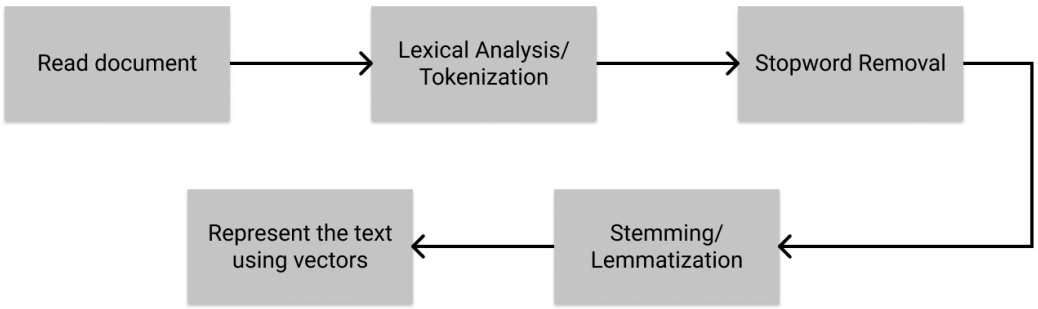


Figure 2.2: A typical preprocessing pipeline.

The process begins when a document is read. The first step is typically to perform lexical analysis. Lexical analysis is the process of converting a stream of characters, the text in the documents, into a stream of words, commonly referred to as tokens.

Tokenization can broadly be classified into three types; word, characters or subword (n-grams) tokenization. One of the most common strategies for tokenization is splitting on white space to create words. However, this can have some pitfalls. During lexical analysis, signs like punctuation, hyphens, and numbers must also be handled. To correctly separate the text into tokens, one has to consider what the token should represent. The goal is to express the text through tokens that represent the meaning as intended in the text [11].

After the text have been tokenized, stopword removal is often performed to ensure more effective queries. Many common words may have little value for retrieval purposes. Examples of such words are the Norwegian "og" (and), "men" (but), and "har" (have). Stopwords are usually frequent terms; thus, removing stopwords also has the additional benefit of reducing the size of the index structure considerably [7].

Another step in order to ensure a more correct comparison of documents and text is the process of applying stemming or lemmatization. Stemming is the raw heuristic process of removing the end of words. An example can be the word "asking". After applying stemming, the word will be reduced to its stem, or root, "ask". One of the most popular stemmers is the *Porter stemmer*, developed by Martin Porter. However, this stemmer is not applicable to the Norwegian language. The most popular Norwegian stemmer is called the *Snowball stemmer*. This stemmer is a modification of the Porter stemmer.

Stemming can be helpful when we want to conflate words to increase the number of possible query matches. However, there are also pitfalls when performing stemming. Stemming does not consider how a word is used. An example is the word "saw". "Saw" will be reduced to "saw" without considering whether the word is used as a verb or a noun. When the word is used as a verb, the canonical form of the word would be "see". To address this problem, lemmatization can be used.

Lemmatization is a similar process to stemming but aims to reduce a word to its lemma. A lemma is a words dictionary, or canonical, form. A lemmatizer is more fine-tuned than a stemmer. To create a lemmatizer, one needs a word list or lexicon that contains as many words as possible. A lemmatizer can often use context and model knowledge. By considering a words surrounding context it can easier determine the correct root [11].

Before representing the processed text numerically, more steps can optionally be included. Which additional steps to choose depends on the overall task at hand. One such step is the task of language identification. It involves trying to predict the language a text is written in. For many problems, having text expressed in the same language is essential. This can include both when querying text in an IR system and when using text as features in learning algorithms [12].

The last step is to represent the remaining tokens as numerical vectors. It is often customary to use terms as features in text classification or clustering. After the data has been represented numerically, it can be used as input to a learning algorithm or compared with other documents using similarity measures. There are several ways this can be achieved. Text representation is a crucial part of the document

preprocessing tasks and will be elaborated more thoroughly in Section 2.2.2.

2.2.2 Text Representation

In an information retrieval system, a document is often described by a set of representative keywords, called *index terms*. In its most general form, an index term can be any word contained in the document. It is, however, more common that the index terms are a group of preselected words representing either a topic or a key concept in a document. The index terms are commonly made up of the nouns contained in the documents, as these words are self-explanatory. Connectives, adverbs and adjectives are often less useful. The set of index terms in a document collection is referred to as the *Vocabulary* of the collection, which is formally expressed in Definition 2.2.1.

Definition 2.2.1. Let t be the number of index terms in the document collection and k_i be a generic index term. $V = \{k_1,...k_t\}$ is the set of all distinct index terms in the collection and is referred to as the *Vocabulary* V of the collection. The size of the vocabulary is t.

In an IR system, documents and queries are modelled through document and query representations. After the text is preprocessed and the vocabulary is defined, the text must be represented numerically. This is done by considering each term in the document representation as an independent variable or *feature*. The technique is generally referred to as *text representation* and can be formally defined as:

Definition 2.2.2. For a given set of text documents $D = \{d_i, i = 1, 2, ..., n\}$, where each d_i stands for a document, the problem of text representation is to represent each d_i of D as a point s_i in a numerical space S, where the distance/similarity between each pair of points in space S is well defined.

Text representation is one of the fundamental problems in text mining, and information retrieval [13]. Several approaches exist that allows a document to be expressed in a numerical space. The remainder of this section will present some of the most popular text representation methods.

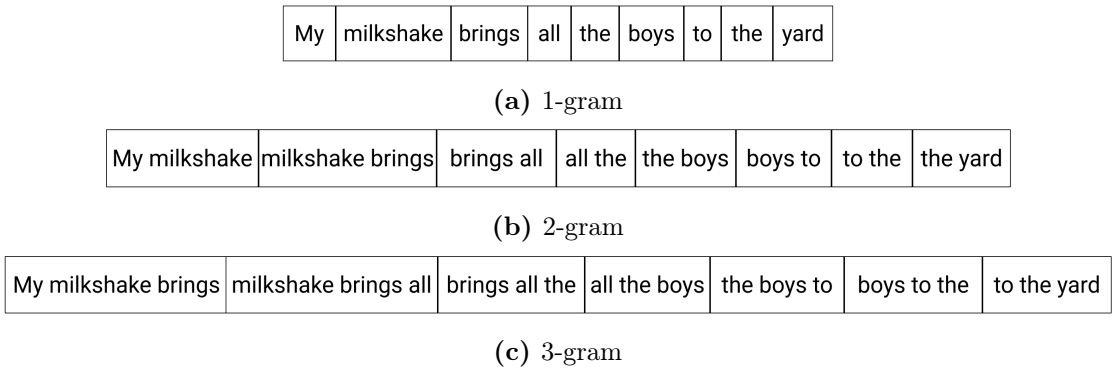
Bag of Words

Bag of Words is the easiest form of text representation. The model is based on the assumption that similar documents contain similar content. Each document text is simply represented as a bag that contains its words. There is no regard to the order of the words or their grammar. Each document can thus be regarded as a word-count vector. These counts can, for instance, be binary counts or absolute counts. Using binary counts, the text is represented by *term conjunctive components* reflecting the terms it contains. Thus, decisions can be made based on either the absence or presence of a specific word. Another approach is counting the term frequency of each term. This makes it possible to find keywords in the documents. The model also enables the application of TD-IDF, which accounts for word specificity.

N-grams

As mentioned earlier in this section, tokens need not necessarily be standalone words. They can also be consecutive words grouped together. *N-grams* is simply any sequence of n characters or n words. The value of n is set by the user and can be 1-gram (unigram), 2-gram (bigram), 3-gram (trigram) and so on. The optimal value to choose for n varies depending on the language. Bag of n-grams is a natural extension of the bag of words model.

N-grams have several areas of application. They are often used when it is important to handle spelling error corrections, likelihood for misspelt words, or predict the next word or characters in a sequence [14]. Another application for n-grams is by using them to improve retrieval in IR systems or in text clustering with the goal of finding similar documents. In some cases, a bag of n-grams can be more informative than a simple bag of words, as it manages to capture more context. For instance, by using trigrams, one would be able to capture "boys wear dresses" rather than just "dresses". However, this comes with the additional cost of producing a larger sparser feature set than if a simple bag of words model were used [15]. An example of how word n-grams can be used to represent the sentence "My milkshake brings all the boys to the yard" is presented in Figure 2.3.



(a) 1-gram

(b) 2-gram

(c) 3-gram

Figure 2.3: The figure shows the sentence "My milkshake brings all the boys to the yard" expressed as (a) Unigram/1-gram, (b) Bigram/2-gram and (c) Trigram/3-gram.

TF-IDF

Instead of simply expressing the presence of a token using binary numbers, each token can instead be assigned a weight indicating its importance in the document or relative to the document collection. According to Baeza-Yates et al. [7], *term frequency* (TF) and *inverse document frequency* (IDF) are the foundations for the most popular weighting scheme in IR, TF-IDF. TF-IDF is used to determine how important a term is in a given document. Baeza-Yates et al. defines TF and IDF as follows:

Definition 2.2.3. Term Frequency: The value, or weight, of a term k_i that occurs in a document d_j is simply proportional to the term frequency $f_{i,j}$.

However, documents are seldom of the same length. Thus, it is possible that a specific term would appear more often in a longer document than in a shorter one. This does not mean that a longer document is more relevant than a shorter one. To account for different document lengths, *sublinear tf scaling* can be used. This formula is expressed in Equation 2.1.

$$tf_{i,j} = \begin{cases} 1+\log f_{i,j} & \text{if} f_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

Term frequency still sufferers from the problem that there are little discriminating power in determining relevance. For that, inverse document frequency (IDF) is used. In simple terms, IDF is used to measure the rareness, or specificity, of a term and is formally defined in Definition 2.2.4.

Definition 2.2.4. Inverse Document Frequency: Let k_i be the term with the r-th largest document frequency, i.e., $n(r) = n_i$. Associate with the term k_i the weight IDF_i given by

$$IDF_i = log\frac{N}{n_i} \tag{2.2}$$

where IDF_i is called the inverse document frequency of term k_i and N is the number of documents in the collection.

By combining TF and IDF, we get a statistical measure for evaluating how relevant a word is to a document in a collection of documents. This is called the TF-IDF weighting scheme, which is formulated in Definition 2.2.5.

Definition 2.2.5. TF-IDF Let $w_{i,j}$ be the term weight associated with the pair (k_i,d_j). The we define

$$w_{i,j} = \begin{cases} (1 + \log f_{i,j}) \times \log\frac{N}{n_i} & \text{if} f_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

2.2.3 Representing the Term-Document Relationship

Once each document has been processed and expressed using some text representation technique, the documents must be gathered and structured for ranking algorithms or machine learning models to take advantage of it. This section presents two commonly used term-document structures.

The Term-Document Matrix

The relationship between terms and documents can be further exploited by creating a matrix structure to represent the collection of all documents and all terms in

the vocabulary. This matrix is known as the *term-document matrix*. Given two documents, d_1 and d_2 and a vocabulary $V = \{k_1, k_2, k_3\}$, the term-document matrix can be written as

$$\begin{array}{c} \\ k_1 \\ k_2 \\ k_2 \end{array} \begin{array}{c} \begin{array}{cc} d_1 & d_2 \end{array} \\ \begin{bmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \\ w_{3,1} & w_{3,2} \end{bmatrix} \end{array}$$

where each $w_{i,j}$ element represent a weight connected with each term k_i in document d_j.

Inverted Index

An *index* is a data structure built on top of a text to speed up the process of searching. The most basic and frequently used indexing technique is the *inverted index*. Inverted indexes are used for retrieval in ranking models that rank according to word frequencies, including the Vector Model, briefly described in Section 2.3.2.

An inverted index, or inverted file, is composed of two elements: the *vocabulary* and the *occurrences*. For every word in the vocabulary, the inverted index stores which document(s) the word occur in. In its most simple form, this is done using the term-document matrix described above, where the matrix consists of the number of times each word appears in each document. However, such a matrix tends to become sparse, thus requiring too much space. A better solution is for each word to associate a list of documents. The set of all those lists is called the *occurrences*.

Inverted lists are also useful when the aim is to retrieve the most relevant documents. When utilising, e.i., the vector model, the order of the inverted list must be changed to one based on decreasing frequency. The number of word occurrences can also be changed to use weights associated with each term-document pair in decreasing order. This is often called *rank-oriented* inverted indexes. Including the number of times each word occurs in each document makes it possible to calculate each document's relevance and utilise schemes such as TF-IDF.

2.3 Similarity Measures and Ranking

In order to compare how similar two documents are, we first need a measure of similarity. There are several ways to compute the similarity between documents. Having a way to express how similar two documents are is essential to IR systems and is often the most important part when working with text classification or clustering. However, most machine learning algorithms rely on the distance between documents. As similarity and distance in terms of documents are highly related, it is often customary to express the distance as a function of similarity. The higher the distance, the less similar the documents. This is expressed in Equation 2.4

$$distance(x, y) = 1 - sim(x, y) \tag{2.4}$$

where $sim(x, y)$ is any similarity measure where the similarity score is in the range $\{0, 1\}$. Very similar documents will have a similarity close to 1, thus a distance close to 0. Highly dissimilar documents will have a similarity close to 0 and a distance near 1. This section presents the cosine similarity, a common approach for measuring the similarity between documents or between a document and a query.

2.3.1 Cosine Similarity

Cosine similarity is the classical IR approach to comparing documents. The formula is calculated using documents represented as weighted term vectors and is expressed in the equation below:

$$sim(d_i, d_j) = \frac{\vec{d_i} \cdot \vec{d_j}}{|\vec{d_i}| \times |\vec{d_j}|} = \frac{\sum_{k=1}^{t} w_{k,i} \times w_{k,j}}{\sqrt{\sum_{k=1}^{t} w_{k,i}^2} \times \sqrt{\sum_{k=1}^{t} w_{k,j}^2}} \tag{2.5}$$

where $|\vec{d_i}|$ and $|\vec{d_j}|$ are the norms of the documents and $\vec{d_i} \cdot \vec{d_j}$ is the internal product of the two vectors [7]. The document vectors consist of the weighted terms in the documents. These are often computed by calculating the TF-IDF scores of the terms. The norms of the documents are used to account for the fact that documents are of variable length, and frequency information can be misleading. The weight of a word is 0 if the word does not appear in the document. Thus, only words that appear in both documents are used to compare the similarity between the two.

2.3.2 The Ranking Process

Modelling in IR is a process aimed at producing a ranking function. A *ranking function* is a function that assigns a score to a document in regards to some query or another document. This process consists of two tasks:

1. A logical framework for representing documents and queries
2. A definition of a ranking function that computes a rank for each document with regard to a query or another document

How to represent documents and queries where presented in sections 2.2.2 and 2.2.3. Ranking functions, or measures of similarity and distance, were discussed in Section 2.3.1. By combining these two elements, a system for ranking and retrieval can be created. One of the most known ranking models is the vector model.

The *vector model* works by assigning non-binary weights to index terms in documents and queries. These weights are again used to compute the degree of similarity

between all documents in the collection and the user's query. Documents are then sorted according to their degree of similarity. The degree of similarity is modelled as the correlation between the document and query vectors and can be quantified by, for instance, the cosine of the angle between these two vectors. This formula was earlier expressed in Equation 2.5. The weights in the vector model are TF-IDF weights.

2.4 Machine Learning

Machine Learning is the science of programming computers to learn from data. It can be applied to a variety of problems, but in general, machine learning is preferred in the following cases:

- Problems where existing solutions require much fine-tuning or a long list of rules. Using a machine learning algorithm can often make the code simpler or even out-perform the traditional approach
- Problems so complex that transitional solutions yield no good solution. The best machine learning techniques might find a solution
- Environments that are fluctuating. Machine Learning systems can adapt to new data.
- Getting insight into large amounts of data and complex problems.

Generally, machine learning algorithms can learn from some provided dataset of variable size by examining the data and identifying common patterns and differences [16]. The following sections present the different types of learning algorithms used in machine learning and some often used methods within the field of automatic grading.

2.4.1 Learning Algorithms

Machine learning algorithms differ in what type of problem they try to solve, what data they input and output, and how they learn. This division leads to a categorisation of different types of learning models. The most known are *Supervised Learning, Semi-supervised Learning* and *Unsupervised Learning.*

In supervised learning, the algorithm is provided with a dataset that includes several rows with information. Each row contains data about an item and a label that can be considered a solution for that item. This dataset is commonly referred to as a training set. The algorithm builds a model based on the training set information by learning why each item is assigned that specific label. After the model is trained, the model is then fed with data about new items and tries to label each item based on what was originally learned during the training phase. Supervised learning can be

used to predict the grade of a student answer if provided sufficiently enough graded answers to learn from.

In unsupervised learning, the training data is not labelled. The algorithm has to learn patterns and similarity trends based on the input without any explicit feedback. Unsupervised learning is often used on pattern detection, text classification, anomaly detection and dimensionality reduction. Unsupervised learning algorithms and techniques used in automatic grading include *K-Means* and *Latent Semantic Analysis*.

The last type of learning algorithm is semi-supervised learning. Most semi-supervised learning algorithms are combinations of unsupervised and supervised algorithms. The model is trained by using both labelled and unlabeled data. Typically there are more unlabeled data than there are labelled data. When it is time-consuming providing labels to all kinds of data, semi-supervised learning can be a good choice [16].

2.4.2 Machine Learning Techniques in Automatic Grading

As will later be presented in Chapter 3, machine learning techniques are often used as means for solving the problem of automatic grading. Two such techniques are K-means and Latent Semantic Analysis. These are now introduced.

K-means Clustering

Given a collection D of documents, a text clustering method automatically separates these documents into K clusters according to some predefined criteria. In K-means clustering, the number K is provided as input [7].

When using K-means, each cluster is represented by a centre point, often called a centroid. Each document in the collection is assigned to the cluster with the closest centroid. The centroids are then recomputed for each cluster. The process is repeated until no centroids change. When dealing with textual clustering using k-means, the procedure is carried out as follows:

Let each document d_j be represented as a weighted term vector $\vec{d_j}$, given by

$$\vec{d_j} = (w_{1,j}, w_{2,j}, ..., w_{t,j})$$

where $w_{i,j}$ is the weight of term k_i in document d_j and t is the size of the vocabulary. Four steps are carried out:

1. **Initial Step:** Select K documents at random and assigned each of them to a distinct cluster. These documents are used as initial cluster centroids.
2. **Assignment Step:** Assign each of the N documents in the collection to the cluster with the minimum distance to the document.

3. **Update step:** Recompute the centroids of each cluster based on the document vectors assigned to it.

4. **Final step:** Repeat steps 2 and 3 until no centroid changes.

Latent Semantic Analysis

Latent Semantic Analysis (LSA) is based on singular value decomposition (SVD) and is a method for representing and extracting the meaning of words based on their context. This is done by applying statistical computations to a large corpus of text. LSA is based on the idea that the context a word appears (or not appears) in can provide a set of mutual constraints that can determine the similarity of words based on their meaning.

After preprocessing text so that it is machine-readable, the algorithm represents the words and any set of words as points in a high dimensional semantic space. The technique relies on the *distribution hypothesis*. That is, words with similar meanings frequently appear together. Several education applications employ LSA, including automatically scoring the content of answers or essays [17].

2.5 Evaluation metrics

In order to determine how well a ranking system or machine learning model has performed, the developed system or model must be evaluated. There exists numerous measures of evaluation. This section will present some evaluation metrics used and discussed in this thesis, including accuracy and error, precision and recall, and purity.

2.5.1 Accuracy and Error

Accuracy and error are to common ways of measuring how successful a given classification algorithm is. In terms of text classification, the algorithm tries to assign a class c_p for each class $c = \{c_1,...c_n\}$ to each document. *Accuracy* is then measured as the fraction of documents assigned to the correct class by the classifier. *Error* is the fraction of documents assigned to the incorrect class.

A binary classification problem has two classes, often called Positive and Negative. The task is to correctly predict the class for each sample. When a sample is correctly classified, it is often called a *True Positive* (TP) or *True Negative* (TN), depending on whether it was classified as Positive or Negative. However, when a Positive sample is *falsely* classified as Negative, this is called a *False Negative* (FN). Similarly, a Negative sample falsely classified as Positive is called a *False Positive* (FP). These definitions can be used to calculate the accuracy and error of a system.

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.6}$$

$$Error = \frac{\text{Number of incorrect predictions}}{\text{Total number of predictions}} = \frac{FP + FN}{TP + FP + TN + FN} \tag{2.7}$$

Accuracy and error can also be extended to a multiclass problem. In such a case, accuracy, $Acc(c_p)$, and error, $Err(c_p)$, is calculated for each class c_p.

Accuracy and error have some disadvantages. If the number of documents in a category is much smaller relative to the total number of documents in the document collection, these metrics might become unreliable. This disadvantage might be minimised by using precision and recall, presented in the following section.

2.5.2 Precision and Recall

In Information Retrieval (IR), precision and recall are measurements to evaluate how well a system performs when selecting relevant documents for a query. *Precision* is a measurement of the fraction of retrieved documents that is relevant for a query. *Recall* is a measurement of the fraction of relevant documents which has been retrieved. By using the same notation as presented in *Accuracy and Error*, precision and recall can be expressed as:

$$Precision = p = \frac{TP}{TP + FP} \tag{2.8}$$

$$Recall = r = \frac{TP}{TP + FN} \tag{2.9}$$

2.5.3 Purity

Purity is an external evaluation metric for evaluating clusters that can be used when applying an unsupervised clustering technique where golden standard classes are available. The gold standard is ideally produced by human judges with expert-specific knowledge. Purity quantifies the extent to which a cluster contains entities from only one class.

To compute purity, each cluster first is assigned to the most frequent class in the cluster. The accuracy per cluster is then measured by counting the number of correctly assigned documents in the cluster divided by the total number of documents in the cluster [18]. The formula used to calculate the purity score per cluster is expressed in Equation 2.10.

$$purity_i = \frac{1}{n_i} \max_{j=1}^{k} \{n_{ij}\} \tag{2.10}$$

The purity of the entire clustering scheme is defined as the weighted sum of the clusterwise purity values. This is expressed in Equation 2.11.

$$purity = \sum_{i=1}^{r} \frac{n_i}{n} purity_i = \frac{1}{n} \sum_{i=1}^{r} \max_{i=1}^{k} \{n_{ij}\} \tag{2.11}$$

where the ratio $\frac{n_i}{n}$ denotes the fraction of points in cluster C_i [19]. Perfect clustering has a purity of 1, while bad clusters have a purity score close to 0.

There are some drawbacks to using purity. In particular, purity is always 1 if there is only one document in the cluster [18].

2.6 Tools and libraries

Many open-source libraries and tools are developed to easily provide reusable and accessible components. Several such libraries and tools are created to work with textual data. Using these tools makes the job more convenient and reduces time spent developing the same techniques over and over again. As this thesis aims to see if off-the-shelf technology can be useful in a grading process, it was necessary to examine the available tools and choose a subset of them to use during the work.

For this thesis, `Python` was chosen as the primary programming language as it has extensive support for various text processing and text mining techniques, as well as IR tools and libraries. Furthermore, `pandas`[1], was chosen for handling the datasets, `NLTK`[2], `spacy`[3] and `polyglot`[4] were used in order to clean and preprocess the dataset. `Whoosh!`[5] and `scikit-learn`[6] were used during the experiments for creating index structures, retrieval models and performing clustering.

[1] https://pandas.pydata.org/
[2] https://www.nltk.org/
[3] https://spacy.io/
[4] https://polyglot.readthedocs.io/
[5] https://whoosh.readthedocs.io/
[6] https://scikit-learn.org/

Chapter 3

Related Work

The literature on automatic grading is vast, and during the last decades, there have been numerous publications on the topic. There are systems developed and research conducted on automatic grading for various question types, ranging from natural language questions, selection questions and more structured text like math questions and programming code. For some types of questions, like in the case of multiple-choice, we say that that task of automatic grading is already solved. However, there is still much work to be done in optimising the process of grading natural language questions.

Automatic evaluation of text-based assessment items, such as essays or short answers, is a complex but important research challenge. Automatic Short Answer Grading (ASAG) and Automatic Grading of Essays (AGE) has been widely explored in decades, dating back to the work of Page [5] in 1966. As technology is becoming more and more used for educational purposes, new methods and systems are increasing in popularity within both research and commercial use, especially in Massive Open Online Courses (MOOCs) [20].

In general, the task of automatically grading text items is conducted in five stages: creating a dataset, preprocessing the text, building a model, grading each answer and evaluating the model. This chapter introduces the field of automatic grading of text-based items and related work. The first section will introduce the types of questions for which systems of automatic grading are relevant before scoping down to the characteristics of natural language questions. Further, an overview of the different approaches to automatic grading will be presented, along with examples of specific solutions within each method. Section 3.3 will go deeper into the different components of automatic grading, including the datasets, textual preprocessing, response and reference-based approaches, and types of grading models. Although the dataset includes some programming code alongside the natural language, this subject will not be visited as the scope of the thesis is to focus on text. In the end, benefits and challenges of automatic grading will be discussed in Section 3.4.

3.1 Question types

Research has been conducted on automatically grading a range of different question types. In a review provided by Burrows et al. [6], a categorisation over the different types of common questions where automatic grading methods can be applied was presented. This overview is shown in Figure 3.1. In terms of depth of learning, the authors distinguished between recognition and recall when categorising the questions. Recognition refers to questions where the respondent only needs to organise or identify key information. Recall refers to questions where the respondent is required to provide original answers expressed with own words.

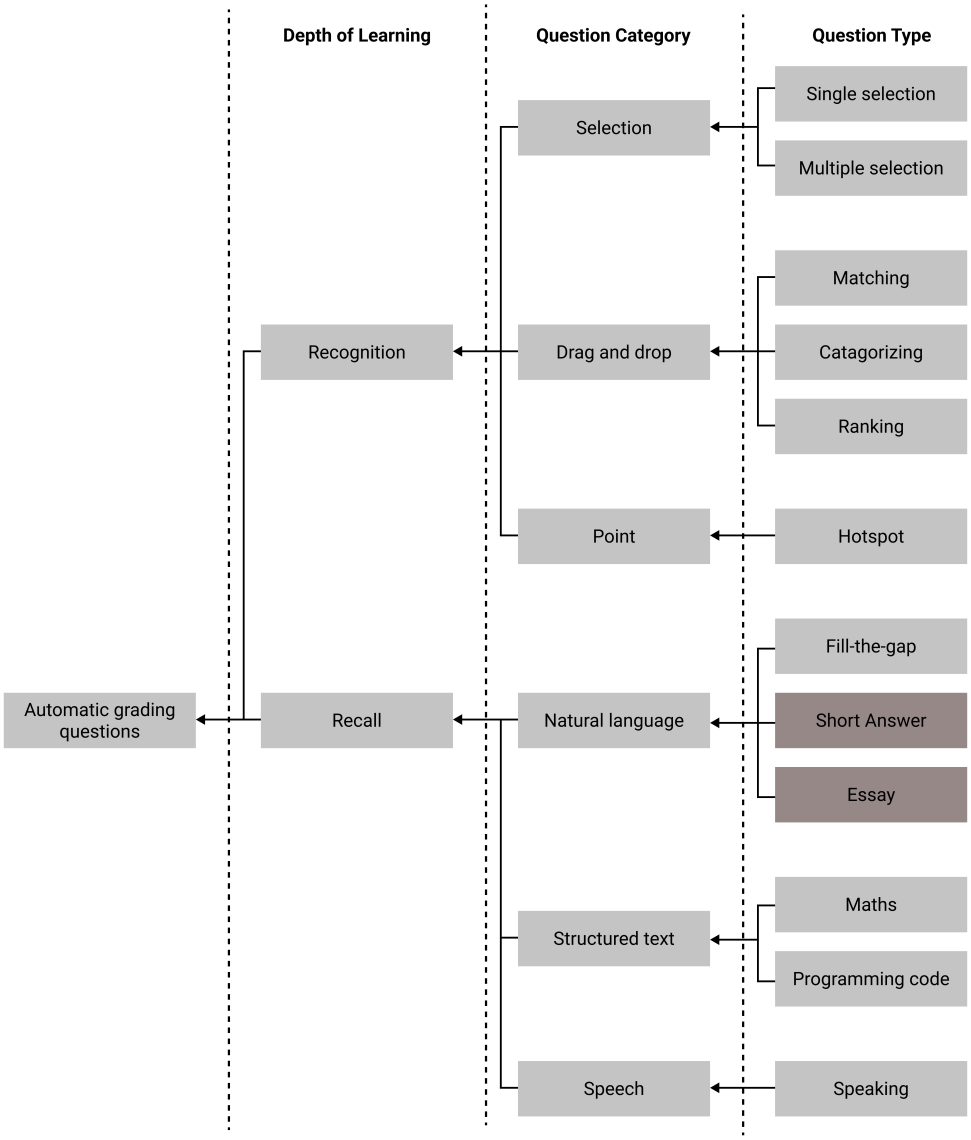


Figure 3.1: A hierarchical overview over questions in which automatic grading methods can be applied.

For recognition questions, the problem of automatic grading is considered solved as the answer is always one among a set of options. Therefore, the focus of the research is on solving automatic grading for the recall type of questions. Within this category

falls questions where answers are formulated as "semi-structured" text, like math and code questions, but also speech and questions expressed in natural language, like fill-the-gap, short answer and essay. As this thesis focuses on natural language questions and answers, the remainder of this chapter will be dedicated to introduce text-based assessment tools and methods used and explored.

Figure 3.1 shows the different question categories where automatic grading methods can be applied. However, in terms of natural language questions, a further division is needed. The difference between fill-the-gap, short answer and essay questions can be blurry. Table 3.1 distinguish the different types according to three properties: *length, focus* and *openness*.

	Question type		
Property	*Fill-the-gap*	*Short answer*	*Essay*
Length	One word to a few words	One phrase to one paragraph	Two paragraphs to several pages
Focus	Words	Content	Style
Openness	Fixed	Closed	Open

Table 3.1: Distinction between fill-the-gap, short answer and essay questions.

In terms of length, both essays and short answers require answer lengths long enough to allow for a variety of unique wordings and answers. This does not apply to fill-the-gap questions as the solutions are only one or a few words. From the literature, most research categorise short answer questions within the length of one phrase to one paragraph. Essay lengths usually range from two paragraphs to several pages.

The second key property is that of focus during grading. From the table, we find that short answer grading focuses on the content, while essay grading focuses on the style, that is, writing style and linguistics. This claim is valid for most identified work done by the author of this thesis. The claim is also backed up by The Educational Testing Service (ETS), one of the biggest contributors to the field of automatic grading. ETS have developed two systems called c-rater and e-rater. C-rater is "an automated scoring engine that has been developed to score responses to content-based short answer questions" [21], while e-rater is based on "features related to writing proficiency in student essays" [22]. Fill-the-gap questions focus on the correctness of the specific word or words.

The last property concerns the openness of the question. Short-answer questions typically tend to be objective, where the respondent answers in facts and statements. On the other hand, essays are more open-ended questions, allowing the respondent to answer subjectively using examples and opinions. For fill-the-gap questions, responses can be characterised as fixed as there is no novelty to be expressed [6].

This separation in terms of the three properties listed is not finite. There are still automatic short-answer grading systems focusing on the style rather than the content and automatic essay grading systems aiming to evaluate the content. Several exams expect students to express their knowledge in far more than one paragraph but still evaluate only the content of their answers. Likewise, not all short-answer questions need to be closed but instead can be characterised as open or semi-open-ended.

However, this division mirrors most research conducted on automatic grading.

3.2 Methods and Evolution of Automatic Grading

The distinction between question types has led to two dominating areas in the field of automatic grading, namely Automatic Short Answer Grading (ASAG) and Automatic Essay Grading (AGE). The general definition of an ASAG system is to automatically assigning a label or score to an answer given in response to a short-answer question [23]. Burrows et al. [6] defined a short-answer question as a question designed to ask for objective facts, where the answer is a free-form input in natural language. Typically, the length of the answer varies from a single phrase to a few sentences or paragraphs. The aim is to evaluate the semantic content of the answer, neglecting the linguistic errors or writing style. Most often, these questions aim to test the students' factoid knowledge.

Automatic Grading of Essays (AGE) is a research area where the goal is to automatically grade students essay without human interference. The length of the essays in AGE systems varies. The shortest are often a paragraph, while the longest could be as long as a master thesis. An AGE system takes an essay as input and outputs a score reflecting the essay's quality. Most often, this is based on the essay's style, structure, and writing [24]. From the literature, the distinction between the two is not finite, and researchers can sometimes use the names interchangeably or characterise the problems slightly differently.

In this thesis, exam answers that have been graded based on their content will be used. Thus, the remainder of this chapter will focus on methods and research with the same goal. The terms essays and answers are used interchangeably, depending on the word used by the researchers, but ultimately refers to student answers. This section will provide an overview of the different approaches used to solve the problem of automatic grading based on content and present systems and research done within each approach. As defined by Burrows et al. [6], the four dominant approaches within automatic grading are concept mapping, information extraction, corpus-based methods and machine learning. In addition to these, a semi-automatic approach, where the primary goal is not to assign each answer with a grade or label of correctness but rather aid the sensor through the grading process, will be presented in Section 3.2.5. Much work has been done on generating feedback for short-answer and essay questions. However, as this is out of the scope of this thesis, it will not be further elaborated.

3.2.1 Concept Mapping

Concept mapping refers to a method in which the student answers can be thought of as made up of several concepts. Then, the task during grading becomes to detect either the presence or absence of each concept. This way of evaluating answers puts some restrictions on the questions being asked: the appropriate questions need to

accommodate the possibility of concepts being present in the answers. An example of such a question is one that asks for a problem to be solved requiring a justification or requests several explanations [6].

Burstein (1996) [25] considered hypothesis-styled questions. The students were given a hypothetical situation and had to compose different hypotheses that could describe why the situation could have occurred. Each answer was considered an individual concept. The hypotheses were then scored as correct or incorrect using a Lexical Conceptual Structure representation in which a concept-based lexicon and concept grammar were built to represent a response set using training data. Concept grammar rules were developed by mapping concepts from the lexicon onto the concept-structure patterns. The system showed an 81% accuracy when tested on the test set.

Automatic Text Marker (2001) (ATM) was introduced by Callear et al. [26] in 2001. ATM aims to break down the student answers and the model answer (the teacher's suggested solution) into concepts of the smallest possible size and counts the number of shared concepts between the two. The student score is then calculated using assigned weights for concepts developed that are similar between the student answer and the model answer.

3.2.2 Information Extraction Methods

Information extraction describes a process in which one selectively structures and combines data either explicitly stated or implied in text. This is done by applying a set of patterns from the analysed text [27]. When applying information extraction to text assessment, the researchers are concerned with identifying facts as patterns in the answers delivered by the students. The idea is built upon the assumption that short answers include specific ideas that can be searched for and modelled. The technique in the information extraction approach is to extract structured data from unstructured sources and then represent the structured data in a format that can easier be analysed [28]. Mitchell et al. [29] specified that the reasons for choosing information extraction techniques in their system AutoMark, was that they were easy to implement, good at handling grammatical mistakes and incomplete sentences whilst not requiring accurate and complete parsing.

The task of computerised grading of free-text responses using information extraction can be framed as follows: The student answers represent the free-text that requires analysis. Each question asked represents a separate domain of the system, and the correct (or incorrect) responses for the specific question represents the concepts of interest [29].

In a review over information extraction techniques for automatic short answer grading, Hasanah et al. [30] identified seven information extraction techniques used to tackle this task. Those are presented in Table 3.2 along with developed systems utilising the various methods. Some of these systems are presented below.

IE Technique	System example
Parse Tree Matching	AutoMark[29]
Regular Expression Matching	WebLAS[31], PMatch[32]
Boolean Phrase Matching	Thomas 2003[33]
Syntactic Pattern Matching	auto-marking[34], eMax[35]
Syntactic-semantic Pattern Matching	FreeText Author[36]
Semantic Word Matching	Auto-Assessor[37]
LRS Representation Matching	CoSeC-DE[38]

Table 3.2: Techniques within information extraction used in automatic grading and systems or research using them.

AutoMark (2002) was developed by Mitchell et al. [29]. The system aimed at automatically grade short answers based on their content without penalising students for errors in spelling, typing, syntax and semantics. The system employs a grading scheme that specifies correct and incorrect answers for each question. This is done by using a syntactic-semantic template. An example of such a template is shown in Figure 3.2. The template can be expected to match a student response if it contains one of the stated subjects, verbs and prepositions. Student style errors are handled by pre-processing the answers using techniques from NLP.

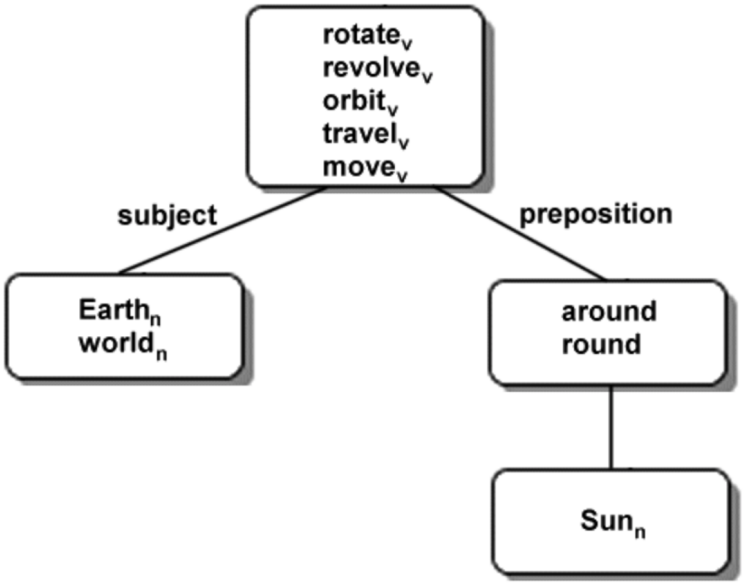


Figure 3.2: Example of a syntactic-semantic template used by AutoMark [29].

WebLAS (2002) [31] makes use of regex matching in order to grade student answers. The system is concerned not only with grading answers but also with task creation and modification. Regexes are created based on the model answer given by the teacher. After a model answer is created, it is parsed, receives POS tags[1], and important elements of the answer necessary to receive a full score are identified by the system and confirmed by the teacher. The model also uses WordNet[2] to search

[1]Part-of-speech (POS) tagging refers to categorising words in the text in correspondence with a particular part of speech, such as adverbs, verbs and nouns [39].

[2]WordNet [40] is a lexical database of semantic relations between words.

for synonym words. Once the model answer is processed, it is used to generate regular expressions for pattern matching. The regex is matched with the student answer to set a correct grade.

Thomas (2003) [33] developed a relatively simple algorithm based on matching key-phrases between student answers and a model answer. They recognised that answers could be made up of several concepts that, put together, accounted for a complete solution. Because of that, they ended up splitting the model answer into a set of alternatives expressed as a set of phrases incorporated into a Boolean expression. Boolean-AND expressions were used for phrases required to be included in an answer. Boolean-OR expressions refer to acceptable alternative expressions. These expressions were then matched with the student answers. The result showed an 86% agreement with the human grader.

3.2.3 Corpus-Based Methods

Exploiting the statistical properties of large document corpora is the general idea behind corpus-based methods [6]. Some of the most promising work done in this field includes utilising Latent Semantic Analysis (LSA), which has been shown to be suitable for document similarity comparisons in automated grading.

Apex (2003) (Assistant for Preparing EXams) is a system utilising LSA to perform this task [41]. The assessment procedure developed consists of two phases. First, an essay-prompt representative corpus is used to create the reference material. The algorithm then processes the word-by-context matrix. In the next step, the system uses human-graded essays to determine threshold similarity values for each grade category by comparing essays to the reference material. The essays to be graded is represented using a query vector and then compared to each document of the LSA representation to calculate the similarity score for the essay. The results showed a significant correlation between the human grads and the grades provided by Apex. One problem the system encountered was tackling very short student texts. The researchers discovered that if a student has submitted an answer with very few words, the content-based assessment could yield a high score, though a human assessor would have considered it low. Lemaire et al. was able to handle it by keeping track of the length of each essay, and if the essay length fell below a threshold, the numeric threshold for achieving a specific score was raised accordingly. The average length of the student answers is not presented in their paper, but they did define their threshold for answers of a short length to those containing less than 300 words.

Atenea (2004) was proposed by Alfonseca et al. [42]. The approach included using a Bleu-inspired algorithm and NLP. Bleu is a method aimed at evaluating and ranking Machine Translation systems. By using some reference translations, it calculates an n-gram precision metric. This is done by finding the percentage of n-grams from the candidate translation that appears in any of the references. Alfonseca et al. used the Bleu algorithm to automatically grade answers by considering the student answers as candidate translations and the teacher's answer as reference. The researchers worked with a dataset consisting of seven questions and 885 student answers. The average answer length varied from 44 words to 166 words depending

on the question. The authors multiplied the metric by a brevity penalty factor to account for very short answers. NLP, including stemming and removal of closed-class-words, was used in the preprocessing phase. The project concluded that their method performed better than other keyword-based scoring modules.

Klein (2011) [43] developed an automatic grading system based on LSA, designed to grade exam answers with a length of approximately one paragraph. The system used only the student's answers, and no model answers were included. Only unigrams were used in the model, though the researchers proposed including n-grams of higher order in further work. The algorithm they proposed consisted of the following steps:

1. Preprocess the text by correcting the spelling, removing stopwords and punctuation
2. Generate a term-frequency matrix using the bag-of-word model and TF-IDF weights
3. SVD is performed. The singular decomposition space is reduced by selecting the k larges singular values and correspondion singular vectors
4. A small master set M of essays are selected to be manually graded
5. When all essays not in M are comparable to at least one essay in M, they are automatically graded using a similarity function

As stated in the steps, the system relied on a human to manually grade a set of all answers before it could predict a grade for the other answers. In order to select which answers to grade manually, three methods were tested. The first method selects answers randomly until all answers, not in the master set, were comparable to at least one answer in the master set (as explained in the next paragraph). The second is based on k-means clustering. The answers are clustered based on their similarity scores. A submission is then chosen from each cluster to be manually graded. That resulted in all remaining submissions being comparable with at least one submission in the master set. The third method used an adopted Min-Max algorithm. First, an answer is chosen at random to be manually graded. That answer is then added to the master set M. The next answer determined by the algorithm is the answer which is the least similar to the closest answer already in M.

For two answers to be counted as comparable, a threshold was determined to specify how similar two answers needed to be. When all answers not in the master set were comparable to at least one answer in the master set, the manual grading process ended. Similarity was computed using a combination of two distance functions, cosine similarity and euclidean distance. When all answers not in the master set were comparable to at least one answer in the master set, the manual grading process ended. The remaining grades were graded the same as the closest submission in M, which yielded better results than using a weighted average of similar solutions in M.

The experiments showed that the Min-Max algorithm performed the best and the random method the worst.

The researchers used a lot of time configuring suitable parameters for their model and found that the parameters needed to be question-specific. With the best parameters, the system achieved over 80% agreement with the human evaluator, given that an average of 82% of the answers was first manually graded. For one question, the system failed; thus, all answers needed to be graded manually.

3.2.4 Machine Learning based Methods

Several researchers have approached the problem by building automatic grading models using statistical machine learning algorithms. The development process can be divided into two main steps: feature engineering and model building. In the first step, features are extracted from the students' answers to represent the characteristics of each answer. Throughout the literature, several features have been proposed and tested. Many of the existing machine learning methods seem to follow one of two general approaches, each which comes with its own strategy for extracting features: (1) learn a question specific grading model for each question, and (2) learn a question general grading model that can grade several questions in the same domain.

Regarding the question-specific grading model, the model relies on a set of labelled student answers. The grading mechanism thus learns directly from the students' answers and typically does not require a model answer. However, a key issue with this method is collecting sufficiently enough labelled answers for training the classifier. In a real-world situation, this approach would depend on having the teacher manually grade several assignments before a model can be built, trained and applied to the ungraded answers.

In the general model, however, features extracted must also be question general. As such, the model answer tends to become essential as it can provide the additional question specific information. Differences and similarities between the model answer and the student answer are thus the most common features in this approach. Other features include question difficulty, answer length, and student competence [44].

Automatic grading has been modelled as a supervised learning problem. In such a case, it can be viewed as either a classification task or a regression task. Classification approaches generally aim to classify the student answer as either correct, incorrect or partially correct. Regression approaches usually have to goal of assigning a specific grade to each answer [45]. Following, a few machine learning approaches will be described.

CAM (2008) (Content Assessment Module) was developed by Bailey et al. [46] in 2008. The dataset used for development consisted of English as a Second Language learner corpus of short answer reading comprehension questions, where the corpus consisted of 566 responses, each 1-3 sentences long. The corpus was divided into a training and test set. A k-nearest neighbour classifier was built based on the data in the training set. The training set consisted of 311 responses from 11 students'

responses to 47 different questions. The test set held 255 responses from 15 students answering 28 questions. The result showed an 88% accuracy in classifying the test set. The dataset they built where based on features that measured the percentage overlap of content on various linguistic levels between the students and teacher answers. In total, 13 features were included. The types over overlap included word unigrams and trigrams, noun-phrase chunks, text similarity thresholds, part-of-speech, lemma, and synonym overlaps.

Nielsen (2008) [47] used decision trees in order to classify primary school science questions with the labels "unaddressed", "different argument", "self contradicted", "contradicted" and "understood". The system consisted of syntactic features (edit distance and dependency relation type) and lexical features (part-of-speech, stem matches, and entailment probabilities). The system showed a 75.5% accuracy.

Süzen (2018) [48] developed two automatic short answer grading methods using text mining and supervised and unsupervised machine learning approaches. The first method clustered similar answers, allowing the teacher to quickly grade one answer that again propagated to answers in the same cluster. This method can also be classified as a semi-supervised approach, as described in Section 3.2.5. The second method used linear regression to assign grades with high accuracy to each student answer. The dataset consisted of 29 student answers from ten assignments and two exams. For each question, there where one model answer having an average length of one word to a sentence.

Both approaches began with applying standard text mining techniques to the corpus of student answers before measuring the similarity between the student and the model answers. Their working assumption was that the students' grades were highly dependent on the words the students used in their answers which also occur in the model answer. Before calculating the similarity score, some NLP was applied. The tasks included removal of stopwords, numbers, punctuation and stemming. They also removed words that appeared in the questions from the student answers and words that appeared in less than 90% of the corpus. The resulting corpus was made up of 20 words. Each student answer was then represented using a bag of words model and term frequency weights.

For the clustering approach, Euclidean distance (the sum of squares distance) was used to compare student and model answers. To adjust for the effect of answer length, the data was normalised using L_2-normalisation. After calculating the distance between each student answer and the model answer, student answers were clustered into groups using k-means. The number of clusters was determined using the Elbow method, a commonly used heuristic to determine the optimal number of clusters. This resulted in a total of three clusters which the researchers named *Excellent, Mixed* and *Weak* depending on the grades in the cluster. A human evaluator can then be tasked with grading one of the answers in the excellent and weak clusters. This grade then propagates to the rest of the answers in the same clusters.

The predictive model aimed to assign a grade between 0 and 5 to each question using the distances between the model answer and the student answers. The student grades were predicted using linear regression and the Hamming distance between the student and model answer. The model was trained by minimising the Mean

Square Error (MSE) of prediction. According to the authors, their model could predict marks with high accuracy.

3.2.5 Semi-Automatic Methods

The systems described in the earlier sections are all concerned with automatically grading or labelling answers with little to no human interference regarding the grading process. However, as will be discussed in Section 3.4, there are disadvantages with excluding a human evaluator from the grading process. Therefore, some solutions have been developed to make the grading process more efficient for the human evaluator by reaping the benefits of including them in the process. Semi-automatic systems and approaches aim to assist the teachers instead of replacing them.

This section is not concerned with models that can be classified as semi-supervised in terms of having the teacher grade a subset of all answers in order for the model to grade the rest. In such a case, it is the system that needs assistance from the teacher. Instead, this section presents systems intended to provide assistance to the teacher by allowing the grading process to run more efficiently. The latter is what here is called a semi-automatic method. As far as the author is aware, few systems have been developed as semi-automatic systems. However, two different approaches identified will be explained next.

The Superlative Model (2006) was developed by Jayashankar et al. [49] to address the fact that no past attempt in automated grading guaranteed a 100% accuracy. The researchers attempted to help the sensor increase efficiency during grading by developing a visual tool to partially automate text grading. The overall architecture consisted of six steps: (1) creating a corpus, (2) document preprocessing, (3) scrutinising synonyms using WordNet, (4) substituting plurals, (5) building a term-document matrix, and (6) creating two word clouds: a *relative* word cloud and a *cohesion* word cloud.

The corpus consisted of student and model answers as unstructured text. Removal of numbers, punctuation, white space and stopwords was then applied to the entire corpus as a part of the document preprocessing step. Synonym words were handled by using WordNet and measuring the cosine similarity between words before plural words were substituted using only the words root. After expressing each answer using a bag of word model and term frequency weight, the answers were represented in a term-document matrix.

The term-document matrix was used to generate two word clouds, a cohesion word cloud and a relative word cloud. The cohesion cloud included words shared between the student and model answer. The relative cloud included the words not shared. When grading student answers, evaluators where presented the answers along with the two word clouds. The researchers also conducted a case study along with teachers to evaluate the system. When asked, the teachers said that they found the word cloud helpful and easy to analyse. They were also tasked to evaluate the student answers using only the cohesion cloud. The model was found to have an average agreement of 98% with the sensor.

Powergrading (2013) was a method developed by Basu et al. [50]. The aim was to allow teachers to grade multiple responses in a single action. Their approach consisted of first developing a general classifier that later could be used to cluster similar answers. The process began by selecting a subset of questions and answers (e.g., questions 3 and 4 and all answers to those questions) and performing a manual job of grouping labelled answers in buckets based on similarity. One of the buckets consisted of answers not fitting any of the other buckets.

After the buckets of similar answers and the outlier bucket were created, the next step was to create a new dataset containing rows describing the relationship between two and two answers. This process was conducted as follow:

1. For each answer in each bucket of similar answers, pick one answer
2. For each answer picked, choose one new answer from the same bucket, one answer from one of the other buckets, and one answer from the outlier bucket
3. Create three new rows in the dataset, each containing features that describe the similarities and dissimilarities between the first answer and one of the other three
4. To each row, add a classifier that describes the distance between the two answers.

This process was repeated until all pairs of answers were compared. The new dataset constituted the training set. After building the new dataset, three different methods were tested to classify the remaining answers; logistic regression, LSA and decision trees. The classifier was trained to learn the distance between any two par of answers. The authors defined several features to describe each pair of answers. They tested each of them individually by only training the dataset using one feature at a time. The test showed that TF-IDF was the most important feature to include, but letter-based similarity was also important. However, the combination of all identified features yielded the best result.

After building the classifier, a distance matrix was created by using the classifier to calculate the pairwise distance between every answer. K-medoits clustering was used to cluster answers into groups of similar answers based on the distance matrix. The teacher could then use these clusters of answers to grade all answers in the same cluster simultaneously.

3.3 Component Analysis

There are other ways of categorising the different approaches taken to solve the problem of automatic grading than the four mentioned in Section 3.2. This section discuss other components that are part of automatic grading. These components can either be mapped directly to belong to one of the mentioned approaches or can be shared across systems.

3.3.1 Datasets

The first step in building an automatic grading system is gathering data and creating a dataset to analyse. These datasets usually consist of multiple student answers, where all or some already have received one or several grades. If several grades are given, it is often due to having each answer graded by two or more teachers. Some datasets include one or several model answers, normally provided by one or several teachers. Throughout the literature, the topic in these datasets seems to vary. Common topics include biology, maths, science, operating systems, history, computer science and more. Most datasets contain student answers written in English, though there has also been conducted studies using other languages such as Turkish or Indonesian [51, 52]. The average answer lengths vary greatly between the different systems; most answers are no longer than one paragraph.

Gathering the dataset can be considered one of the primary challenges in ASAG and AGE. Across the literature, there is little overlap in the different datasets used. It varies how detailed the researchers describe their datasets; few provide a detailed description, and some do not describe their datasets at all. That means that there is no standard dataset tested among the different systems. Because of the lack of a common benchmark, measuring the methods against one another is challenging. One reason for this limitation is the lack of available datasets. Burrows et al. [6] explained this by referring to the fact that many academics used datasets from their own universities.

There have been some attempts in making the research into automatic grading more comparable by providing open datasets and arranging competitions. In these competitions, researchers compete against each other using a shared dataset. Some of these datasets are not limited to answers on one topic but contain answers from various topics such as arts and science. One company hosting such an event was Kaggle [53].

3.3.2 Natural Language Processing and Features

Natural Language Processing (NLP) is a branch within computer science and artificial intelligence aiming to give computers the ability to understand text in the same way as humans [10]. At the highest level, the NLP techniques used in automatic grading belong to one of two categories: linguistic processing techniques or statistical processing techniques. In the first category are techniques that perform textual manipulation, while the latter focus on extracting features from the text. This section will present both, starting with the linguistic techniques.

In their review of 35 ASAG systems, Burrows et al. [6] identified 17 different linguistic processing techniques used in processing student answers. Figure 3.3 shows an overview of the identified NLP techniques and their frequency. The figure only represents the number of times each technique was mentioned in a paper. As not all researchers describe their processing techniques in detail, the frequency count might be somewhat misleading. The linguistic techniques identified can roughly be

divided into five categories: lexical, morphological, semantic, syntactic and surface features. A taxonomy representing these techniques can be seen in Figure 3.4.

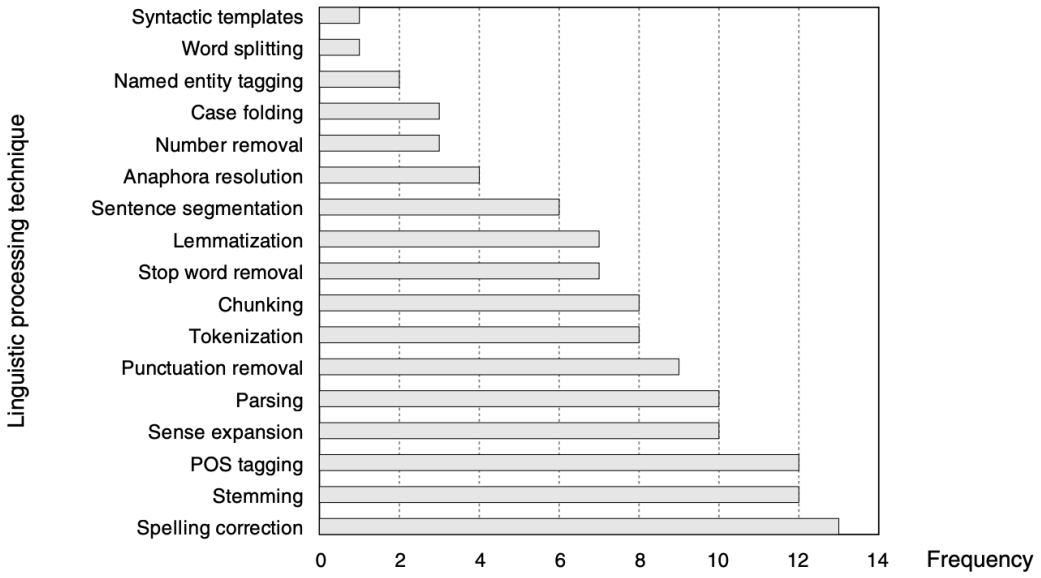


Figure 3.3: Frequency of use of different linguistic processing techniques [6].

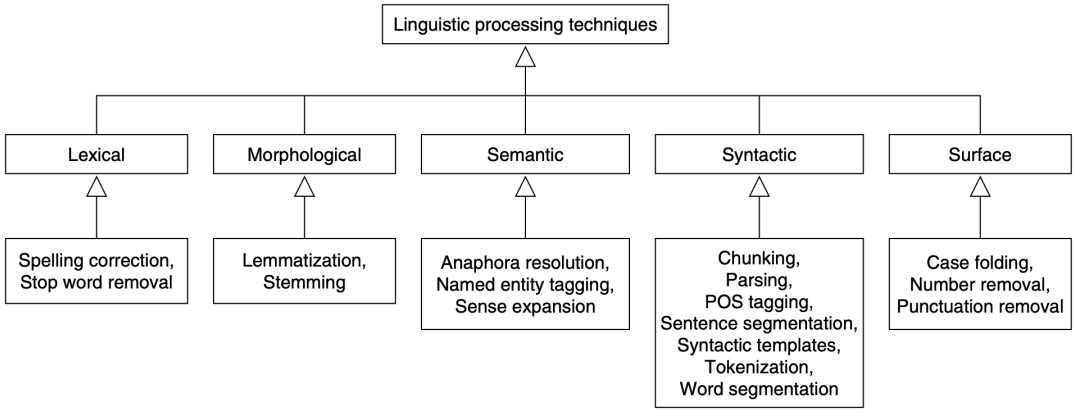


Figure 3.4: A taxonomy over different linguistic processing techniques [6].

Statistical techniques result in features that typically apply to systems utilising machine learning. Extracting features from the text is typically known as feature engineering. Many of the features that can be extracted from text can roughly be divided into the same five categories as the linguistic techniques: lexical, morphological, semantic, syntactic and surface features. In the lexical features, one can find stopword overlap, spelling errors and bag-of-words. Morphological features include stem matches, while an example of a semantic feature is LSA. POS tags, dependency parse tree features and verb occurrences belong to the syntactic category, while character count, word count, sentence count, word length and punctuation are examples of features that belong to the surface category.

There are also other statistical features that fall outside of these five categories. These include the use of n-grams at different levels (character, word or similar),

information retrieval features like term frequency, cosine and F1, textual similarity using edit distance, and longest common sub-sequence, to mention a few [6].

The traditional ASAG and AGE systems rely on features strategically designed to evaluate and grade answers. Of such, the performance of the systems are tightly bound to the quality of the features [54]. The majority of available NLP tools today are developed to handle the English language. It is worth noticing that though most ASAG and AGE systems are developed to target English answers, there is research conducted on automatic grading of text for other languages. It is often stated that performing NLP and extracting features is the most challenging part of the process.

3.3.3 Response-based vs Reference-based Approaches

One way to classify the different approaches to computerised assessment is whether the method is response-based, reference-based or a hybrid of the two. Reference-based approaches focus on comparing student answers with a model answer. Response-based approaches are not concerned with a model answer but instead compare the student answers with each other. A third way is a hybrid approach, in which a combination of the above-mentioned approaches is used. Below are some characteristics regarding the response and reference-based approaches and relevant features for each of them [55].

Reference-Based Approaches

As short-answer questions typically have one or several clear, correct answers, many systems and researchers have focused on developing a reference-based system. In reference-based systems, the answers provided by the students are compared to a model answer or answers, usually provided by the teacher, to assess the answers. Thus, the features extracted are based on the similarities and dissimilarities between the students' answers and the model answer(s). Some also compare the student answers to selected phrases collected from textbook materials used in the curriculum.

Reference-based features are commonly embodied by similarity measurements that aim to capture various aspects of similarity between the student answers and the model answer(s): content, structure and style [56]. One way to capture content-similarity between the model answer(s) and the student answers is by using the Bleu algorithm. This was done by Alfonseca et al. [42] in developing Atenea. Other reference-based features focus on comparing the stylistic or syntactic aspects between the answers [57].

Response-Based Approaches

Another way to assess student answers is by comparing all answers with each other without using a model answer. This approach is called a response-based approach. The assumption is that answers given the same label or grade share similar char-

acteristics. Features applicable to the reference-based approach can also apply to a response-based approach. An example is the bag-of-words model. In a reference-based approach, the bag-of-word model can be used to calculate the cosine similarity between the student answer and the model answer. In a response-based approach, it can be used to generate a vector space representation using all student answers, which again can be used, for instance, to train a classification model [57].

Response-based features also include both similarity, syntactic, semantic and lexical features. An example of a similarity feature was presented by Klein et al. [43] when they used LSA trained only on student answers to calculate the distance between labelled and non-labelled student answers. Statistical language modelling is also common in response-based approaches. One example is the use of character and word n-grams. Mantecon et al. [58] proposed a solution where multiple classifiers were trained using a sparse vector space model containing TF-IDF values calculated from unigrams, bigrams and trigrams. Lexical and syntactic features in a response-based approach include the length of responses, number of sentences, and POS-tags.

3.3.4 Grading Models

In Section 3.2, the approaches taken to design automatic grading systems were divided into four different methods: concept mapping, information extraction, corpus-based and machine learning. However, by inspecting the grading models used within the various methods, it is possible to find common elements that allow for higher classification of grading models. By looking at the research conducted within concept mapping and information extraction, most models seemed to be based on entailment or pattern matching. On the other hand, the corpus-based and machine learning methods frequently treated the calculated scores as individual features. This has led to a higher-order organisation of the methods, namely "rule-based" and "statistical" grading models. The rule-based method include concept mapping and information extraction methods. The statistical method include the corpus-based and machine learning methods [6].

There are benefits and disadvantages to both the rule-based and the statistical approaches. The trad-offs can be classified in two dimensions; whether or not they can handle unseen questions and domain and whether they apply to repeated assessment or not. Put in other words, there are two key properties: generalisation and repetition. As rule-based methods can make additional investment in specific solutions when benefits can be realised several times, they can be considered more suitable for repetitive assessment. On the other hand, statistical approaches are deemed more appropriate when handling new and unseen questions and domains.

As of now, there seems to be more recent work done on statistical approaches compared to rule-based approaches. Burrows et al. [6] deemed the ultimate goal of assessing text-based items automatically as developing an accurate system able to inhabit both the properties of generalisation and repetition. However, whether the rule-based or the statistical methods are strong enough to tackle both challenges are yet unclear.

3.4 Benefits and Concerns of Automatic Grading

There has been work conducted on the topic of automatic grading for several decades. It is clear that being able to perfect the art could bring along several benefits for both students and teachers. In a review conducted by Hahn et al. [59], 125 studies published between 2016-2020 was examined in order to identify the effects of automatic scoring and feedback in educational settings. Their study identified three main benefits affecting educational goals. These were:

- Reduction in bias and increase consistency of the grading
- Instructors to focus on other activities instead of grading
- Ability to provide education to a larger number of students at the same time

Automated grading systems have also been subject to much controversy. Common criticism of such systems focuses on the machines' ability to interpret meaning, correctness of content and quality of argumentation. Because machines cannot truly read, understand and interpret meaning, it is a chance that the systems can be insensitive to features that humans are better at detecting and penalising, such as lack of coherence or repetition. Many systems are also developed by letting machines mimic human graders. Human graders can also be unreliable and prone to subjective opinions. Thus, systems trained using grades given by human evaluators can also be affected [60].

A survey conducted by Patil et al. [61], highlighted some concerns regarding the existing methods and technologies. The drawbacks observed included:

- No standard dataset of same size is maintained across all developers
- Instead of improving on one method or approach for grading, various approaches and methodologies for grading are used, having answers vary in length
- Inconsistency in rating systems due to erroneous judgment
- Lack of faith in the system and process

In addition, utilising only partially perfected systems can result in students being assigned incorrect grades. Thus, there is still a long way to go for putting these systems to use at today's schools and universities.

Chapter 4

Dataset Analysis

This thesis examines three datasets containing answers and grading information from three exams in the course IT2810, Web Development, during the years 2018, 2019 and 2020 at the Norwegian University of Science and Technology. Henceforward, these three datasets will be referred to as E18, E19 and E20, respectively. This chapter will describe the datasets and their content. Section 4.1 provide a summary and overview of the datasets. Furthermore, the data cleansing process and exploration phase necessary in order to perform further analysis and experiments will be explained in Section 4.2. Challenges related to this phase and how they were handled are also included. Section 4.3 explores two important features in the dataset; the grade distribution and the answer length. At last, Section 4.4 focuses on what challenges are related to the three datasets in terms of automatic assessment.

4.1 Datasets Summary

All three datasets are from exams held in the same course, namely IT2810 Web Development, from 2018, 2019 and 2020. The course is a third-year course at the Norwegian University of Science and Technology (NTNU) and covers technology and methods used to develop web-based solutions. The curriculum includes learning standards, formats, languages and architectures used in web applications and services [62]. During the given years, the same professor was employed as course coordinator and had sole responsibility for grading all exams. Hence, all answers received one grade from the same sensor. The instruction language is Norwegian, thus, the answers are *mostly* written in Norwegian. As there exist two official written languages in Norway, *Norwegian Bokmål* and *Norwegian Nynorsk*, the students were free to answer in either of the two languages. Some students also chose to answer in English.

The three datasets each have a different number of questions; 16 (E18), 8 (E19) and 3 (E20). During the 2018 exam, 164 students participated, while at the 2019 and 2020 exams, the numbers were 185 and 220 students, respectively. A high-level overview of the characteristics describing the three datasets can be found in Table

4.1. The table also includes the minimum, maximum and median word count of the exam answers for each exam.

All exams made use of the digital platform Inspera, an online assessment solution used by NTNU and other universities. The questions asked were meant to test the student's knowledge gathered throughout a semester of learning. Most questions were asked in natural language, where answers were expected to be given in natural language as well. However, many students also included some programming code to exemplify their answers, or simply function names, function calls, variable declarations or HTML tags. Some questions explicitly asked to use examples, while others did not. One question, question 8 in E18, was a pure programming question. This question was excluded from further examination during the cleaning phase described in Section 4.2. There has already been conducted work on automatic assessment of programming code by other researchers [63, 64], and this falls outside the scope of this thesis. An overview of all questions asked can be found in Appendix A.

Each question was given a grade ranging from 0 to 5, 0 being the lowest and 5 being the highest. Students do not receive their grades for individual questions; instead, they are given the overall grade calculated by adding together the scores for each answer multiplied by a weight specifying the importance of the question. In these exams, all questions were equally important, and the weights were set to 1. The distribution of question grades for each exam can be found in Table 4.1. The table also includes the overall average question grade.

Not every student chose to answer each question. Some left answers blank, while other students had answers like *This was hard...*, *I don't know*, simply unrelated fill-words or longer explanations of why they were unable to answer the question. These answers were chosen not the be excluded from the dataset as such answers are often common during exams, and thus the dataset remained more realistic.

The average number of words used in the answers differed for each question. The shortest answer contained no words and was simply an empty string, while the longest was 1049 words. The average answer length increased for each year. In 2018 the average answer length was 79 words per answer, while in 2020, this value got as high as 357 words per answer. By comparing the number of questions in each exam with the average answer length, it is notable that the average number of words used in the answers increases as the number of questions is reduced. The mean, median, minimum and maximum number of words for individual questions can be found in tables B.1, B.2 and B.3 in Appendix B together with the distribution of grades for each question. It is worth noticing that the word count (number of words in an answer) was calculated after the initial data cleaning process described in Section 4.2.

As mentioned, answers written in three different languages were present: Norwegian Bokmål, Norwegian Nynorsk and English. In addition, several English load words were included in almost every answer. These words were mostly what can be considered terminology words. That is, words that are typical for the course or topic. A decision was made only to include answers primarily written in Norwegian Bokmål, which accounted for the majority of the answers. Section 4.2 provides an explanation of how the process of identifying those answers was carried out. Henceforward,

a distinction is made between two datasets:

1. **Original DS**: Containing all answers to all questions, in all languages.
2. **Bokmål DS:** Contains only Norwegian Bokmål answers, excluding the pure programming question and associated answers and a few other answers explained in Section 4.2.

For the tables and statistics presented in this chapter, all data describes Bokmål DS unless otherwise explicitly stated. Bokmål DS is also the only dataset that will be used for experiments during this thesis.

Property	**E20**	**E19**	**E18**
Students participating	220	185	164
Number of Questions	3	8	16
Answers Original DS	660	1480	2624
Answers Bokmål DS	639	1454	2385
Mean Grade	3.56	3.44	3.60
Answers graded 0	2	43	151
Answers graded 1	5	68	72
Answers graded 2	40	256	191
Answers graded 3	217	284	649
Answers graded 4	336	439	431
Answers graded 5	39	364	891
Mean W.C.	357	131	79
Median W.C.	333	111	66
Min W.C.	1	1	0
Max W.C.	1049	621	460

Table 4.1: High-level overview over the three datasets. W.C. is an abbreviation for word count, the number of words in an answer. The mean grade, number of answers per grade and the word count are numbers from Bokmål DS.

4.2 Data Cleaning and Exploration

This section describes the data exploration phase and cleansing process. Several challenges or obstacles were detected during the exploration phase, and some immediately affected the cleansing process. Table 4.2 provides a summary of all detected challenges related to the cleansing phase and which exam(s) they applied to. Challenges affecting experiments and strategies related to automatic assessment are described in Section 4.4. At the end of the section, a proposed pipeline for how to handle future cleansing processes of Inspera exam datasets is suggested.

During the cleansing phase, the data was prepared for further analysis and experiments. The original datasets in their raw format contained 33 attributes describing the exam, answers, scores and other relevant exam information and was written

Id	Challenge	Dataset
CH1	Text formatting by Inspera vs programming code	E20, E19, E18
CH2	Use of tables in answers	E20, E19, E18
CH3	Use of images in exam answer	E19, E18
CH4	Use of citations	E20
CH5	Not answered questions	E20, E19, E18
CH6	Referring to previously answered questions	E20, E19, E18
CH7	Presence of special characters	E20, E19, E18
CH8	Pure program-code questions	E18
CH9	Different languages	E20, E19, E18

Table 4.2: All identified challenges, tagged with an ID and which dataset the challenge related to.

in a nested JSON format. The first step of the process thus included normalising the data and selecting the relevant attributes. The normalisation step was conducted by utilising the force of pandas dataframes[1]. During the normalisation process, it was also noticed that all student answers were encoded using HTML Character Encoding. Thus, all answers needed to be decoded before further exploration and cleansing were conducted. This was easily done using the python `html`-package.

All exams were digital exams answered using the Inspera platform. The interface that the students interact with is displayed in Figure 4.1. As one can see from the image, the students are allowed to format their text by making the text bold, creating lists, inserting equations, creating titles and so on. Thus, when the answers were stored, they were stored together with markup language and HTML tags describing the text's metadata. In principle, this can be easily handled by identifying and removing all HTML text and markup. However, as this exam covered the topic of Web Development, HTML tags were also used by multiple students as part of their answers. There was no distinction in the raw data between what was created by Inspera and what was written by the students. Thus, simply identifying the HTML tags and removing them would also remove parts of the students' answers. There exists no openly available overview over which HTML tags Inspera uses. This challenge is listed in Table 4.2 as CH1.

The strategy to deal with this challenge was to simply remove all markup and HTML in the text, including the HTML written by the students. This was chosen as the text was overflowing with Inspera-generated HTML and markup, and identifying the once written by the students would need to be done manually for each answer. An assumption made was that most questions were expected to be answered in natural language, and thus the HTML and markup used by the students had less effect on the final grade than the rest of the written words. The identification and removal process was conducted by using regular expressions, patterns used to match character combinations in strings. All HTML tags and markup was replaced with a space character.

[1]Pandas dataframe is a 2-dimensional labelled data structure with columns of potentially different types that allow for easy manipulation of the data inside the dataframe [65].

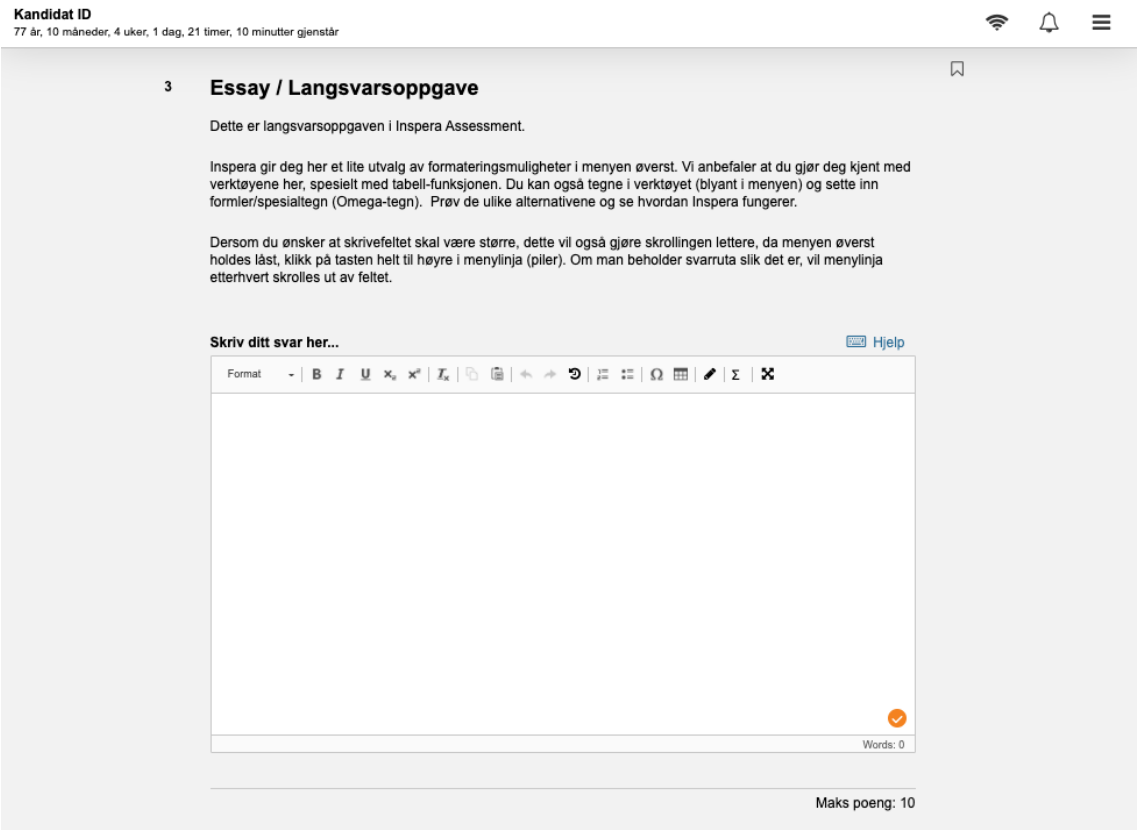


Figure 4.1: Interface to Inspera, a online assessment solution used by NTNU [66].

The allowance of formation of student answers also led to more complex cases: the students were allowed to structure their answers using tables (CH2) and include images in their answers (CH3). As in the above case, keeping the text in the tables and removing only the HTML tags can easily be done using regular expressions. However, due to how tables are structured in HTML, removing the tags would result in a text structure where the connection between which cell-text that belongs to the different columns would be lost. If utilising a simple bag of words model using individual words as tokens, this would not be a problem as bag of words does not consider the word ordering. However, if using n-grams of longer lengths or more advanced strategies, unrelated words might be grouped together and create word relations that does not exist. Nevertheless, as only one answer in E20, nine answers in E19 and seven answers in E18 included tables, this was handled by utilising the same strategy as for other HTML tags. Other options should be explored if the use of tables is more extensive.

Two answers in E19 and eleven answers in E18 included images. These answers were handled in one of two ways. First, all images were identified and removed using regex. The affected answers were tagged with an `image_removed` attribute set to `True`. All other answers were tagged with a `image_removed` attribute set to False. Then, the number of words in the image answers were counted and compared to the average word count of all other answers belonging to the same question, received the same grade and had an `image_removed` attribute was set to False. The second was chosen as the word count was later found to correlate with the answer grade, as is described in Section 4.3.2. If the answer length of the image answer was

drastically lower than the average word count, the entire answer was removed; if not, the answer was kept though the image was not. The assumption was that if the answer length was much lower than the average answer length, the grade was most likely given based on a combination of both the text and the image. If the answer length was somewhat equal, the assumption was that the image did not affect the grade significantly.

During the exam of 2020, the world was affected by the covid pandemic. This impacted the conduction of the exam. To decrease contamination in the society, the exam was altered to be taken as a home exam. This impacted the answers given during the E20 exam. Challenge CH4 is a direct result of this. Because the exam was no longer conducted in a regulated environment, the students could use all available aids, including the Internet and PowerPoint slides from the course curriculum.

As utilising all sources and aids was possible during the 2020 exam, the students were asked to list the sources they had used in their answers. This introduced a new challenge in the text cleaning process (CH4): many of the answers contained text like "*Source: lecturers and http://some-url.com*". Though this information can be useful during a manual grading process, it is probably not helpful in order to compare answers and thus needs to be removed.

One obstacle concerning the removal of sources and URLs was that the students were not required to write their sources in any particular format. Thus, sources were both listed at the end of the text, in the middle, right after they were used, in the beginning, or by using Vancouver-style citations[2]. Some students simply stated that they had not used any sources at all. In addition, the word "source" (Norwegian: *kilde* or *kjelde*) has in both Norwegian and English several meanings and was used both to list sources and as a descriptive word in the middle of the text. Identifying what was what, was challenging but was ultimately conducted by creating numerous regex's that were able to handle different cases.

Furthermore, the words "http" and "https" were not only used as part of a source URL. As the topic of the exam was web development, the words also described HTTP verbs or were used to exemplify API-calls, e.g., `https://example.api/user/<id>/`. By using regex matching and checking the length of the matched strings, most source listings, words describing the sources used and URLs were successfully removed. However, there might still be some source references not caught, and some text removed that should have remained in the text as they referred to API calls. A manual check was conducted to ensure that most cases were handled correctly. This was not a problem in the 2018 and 2019 exams as this was before covid, and the students did not have access to any additional aids. As there is a growing debate on whether to continue the practice of home exams in the future, the format of citations in student answers should be discussed if automatic grading practises should be part of the grading process.

Not all students chose to answer every question (CH5). In an ideal dataset, these answers would just contain an empty string or a None-value. This was not usually the case with the answers to these exams. While some students left the questions

[2]Vancouver is a numbered referencing style commonly used in medicine and science [67]

they could not answer blank, many chose to write a short text describing that or why they did not know the answer to the question. These answers varied in length; some only contained a single word, while other students used over 40 words in describing that they did not know the solution.

A decision had to be made if these answers should be included or excluded from the dataset. If all unanswered questions had been left blank, the choice would be to exclude these answers, and they can easily be identified and automatically graded as failed by using a simple *if*-statement in a computer program. However, because so many chose to write some small unrelated text instead, the answers were chosen to be included as all such answers cannot be as easily identified automatically. However, it is worth mentioning that most such answers fall below a given threshold in terms of the number of words included in the text. This can be exploited by a sensor in order to simultaneously grade low-quality answers, as is proposed in Section 6.2.

Some students chose to refer to earlier answered questions in their answers (CH6). Such cases occurred in all three datasets, though it only applied to a small number of answers. As there was no simple way of automatically identifying all such answers, they were left as-is. However, such answers can likely deviate from other answers unrelated to what approach is used to evaluate the the answers.

Special characters were another encountered obstacle (CH7). Usually, the removal of special characters is a part of the lexical analysis conducted during document preprocessing. However, due to the exam topic, special characters showed to be more extensive in this dataset than in regular text. Thus, special characters were handled in the initial data cleansing process. In addition to signs like "." (dot), "," (comma), "-" (hyphens) and "(...)" (parenthesis), signs were also used in example programming code, and numbers and hyphens were used to indicate list structures. The solution became to replace all special characters with a space character (" "). Any multiple proceeding space characters were then reduced to a single space character.

As mentioned in Section 4.1, question 8 in E18 was a pure programming question. It asked the students to write the code for a React-component that included an H1-element with the text "Hello World", no natural language was expected (CH8). Such questions are outside the scope of this thesis. As a result, question 8 in E18 was dropped from the dataset.

Another challenge, briefly described in Section 4.1, was that students had chosen to answer questions in different written languages (CH9). Though most answers were written in Norwegian Bokmål, two other languages were detected: English and Norwegian Nynorsk. That means that in a simple word comparison strategy, three words with the same meaning would not be automatically counted as equal. An example is the English word "source", which in Norwegian Bokmål is written as "kilde" and in Norwegian Nynorsk is written as "kjelde". There are many more examples that could be mentioned here. In addition, due to the exam topic, questions primarily answered in Norwegian also included several English loanwords. To handle this difficulty, the following strategy was utilised:

First, the library `polyglot` was imported as a python package. Polyglot is a natural

language pipeline that supports massive multilingual applications [68]. The library is able to analyse written text and detect in what language the text is written. If the text contains snippets from different languages, the detector can find the three most used languages in the text as well as the most probable languages used in the text. It also includes the confidence level for each language. Because of this, and because polyglot can detect both English, Norwegian Bokmål and Norwegian Nynorsk, it was chosen as the preferred tool for performing language detection. After the detector was run, each answer was tagged with the most probable language. Some were also tagged with a second and third most likely language. An overview of how many answers were classified as most likely to be written in what language is presented in Table 4.3. For each language (no: Norwegian Bokmål, nn: Norwegian Nynorsk and en: English), the table displays how many answers were tagged as the most likely, second-most likely or third most likely to be written in that language.

Exam	**No** 1^{st}	**No** 2^{nd}	**No** 3^{rd}	**Nn** 1^{st}	**Nn** 2^{nd}	**Nn** 3^{rd}	**En** 1^{st}	**En** 2^{nd}	**En** 3^{rd}
E20	636	0	1	13	0	0	11	117	1
E19	1354	35	0	12	3	0	112	220	7
E20	2242	53	0	84	2	0	133	414	3

Table 4.3: Answer tagged with either Norwegian Bokmål (No), Norwegian Nynorsk (Nn) or English (En) as first, second or third most probable language for all answers in Original DS.

After conducting a quick manual inspection, it was clear that the detector did not perform perfectly, particularly on shorter answers. It was especially problematic that although most answers primarily were written in Norwegian Bokmål, many included several English loanwords. Thus, a higher number of answers were marked as English than what was the reality.

Two more steps were added to the process of separating answers written in different languages. First, if an answer was marked with English as the most probable language, but polyglot also detected Norwegian Bokmål or Norwegian Nynorsk in the text, the answer was considered written in its second-most probable language (Bokmål/Nynorsk). Secondly, an assumption was made that the students answered all questions in the same language. Thus, for each answer written by the same student, a check was made to see which language was used on most of that student's answers. The language most answers were classified as written in was considered the primary language for all answers by that student.

The above strategy is not guaranteed to be 100% correct, and some answers might have been treated as written in another language than what was the case. However, this approach should ensure that most questions are classified correctly.

After all answers were classified, a new dataset, *Bokmål DS*, was created containing only answers classified with Norwegian Bokmål as the most probable language. English answers were excluded as the scope of this thesis was to investigate Norwegian exams answers. In addition, a choice was made to exclude answers written in Nynorsk.

The entire data cleansing pipeline can be summed up in the following steps:

1. Normalise the dataset, select relevant columns and drop irrelevant columns
2. Decode HTML Charset encoding
3. Remove all HTML-tags and markup language except those describing images
4. Identify answers including images and drop answers where the image is suspected of having an impact on the grade
5. (If home exam) Remove all sources and citations
6. Replace all special characters with a space (" ") character and reduce multiple proceeding space characters to a single space character
7. Remove any pure programming-code questions
8. Mark all answers with the language they are written in
9. Keep only answers written in Norwegian Bokmål

4.3 Feature Analysis

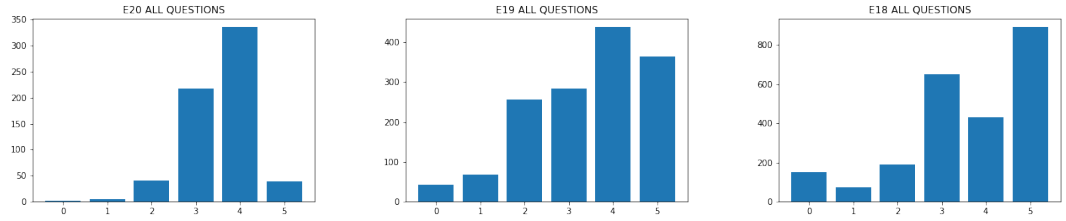
An important part of the dataset analysis is to look at the different features present in the dataset. Some features might help characterise or differentiate different answers, while others might affect the choice of which methods or approaches to examine. The information found can also help explain the results of the approaches tried later. This section will look into the grade distribution and the answer lengths for the three exams.

4.3.1 Grade Distribution

The grade distribution was briefly mentioned in Section 4.1. This section will go more in detail into the different distributions found both in the overall exams and per question. The section will examine what types of distributions are present, what type of patterns might be found, and what implications that might have for further experiments.

The overall grade distribution for E18, E19 and E20 are presented in Figure 4.2. From the plots, it is clear that the grade distributions are *skewed right*. That means that the higher grades are more common than the lower grades. In E18, the most common grade is 5, while in E19 and E20, 4 is the most commonly received grade. The grades 0 and 1 are fairly uncommon in all three datasets. This means that the datasets are unbalanced, that is, the target variable has more observations for some specific classes than for others.

Dealing with an unbalanced dataset can have several implications on various experiments related to automatic assessment. One example is in trying to predict a correct grade using supervised machine learning, which was found in Chapter 3 to

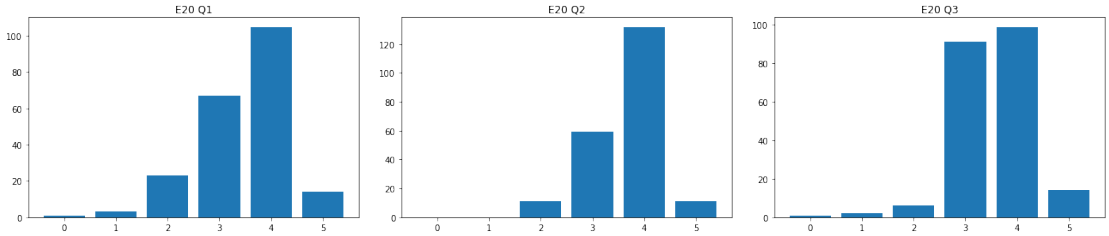


(a) Grade distribution in E20 **(b)** Grade distribution in E19 **(c)** Grade distribution in E18

Figure 4.2: Grade distribution for all three exams, based on the scores given to individual questions.

be a fairly common approach. During such an approach, the dataset is usually split into a test and a training set. The computer is given access to the class labels in the training set and tries to learn a pattern that will help it predict the correct class of the answers in the test set. If the computer sees more of some classes than others, it can be prone to be more biased towards those specific classes because it will be harder for it to understand the underlying pattern that allows the model to distinguish between characteristics of the different classes [69].

The distributions per question in all three exams can be found in figures 4.3, 4.4 and 4.5 for the questions in E20, E19 and E18 respectively. Many of the distributions found for the individual questions in the datasets follow the same pattern as found in the overall grade distributions. That is, there tend to be more answers receiving a high grade than a low grade.

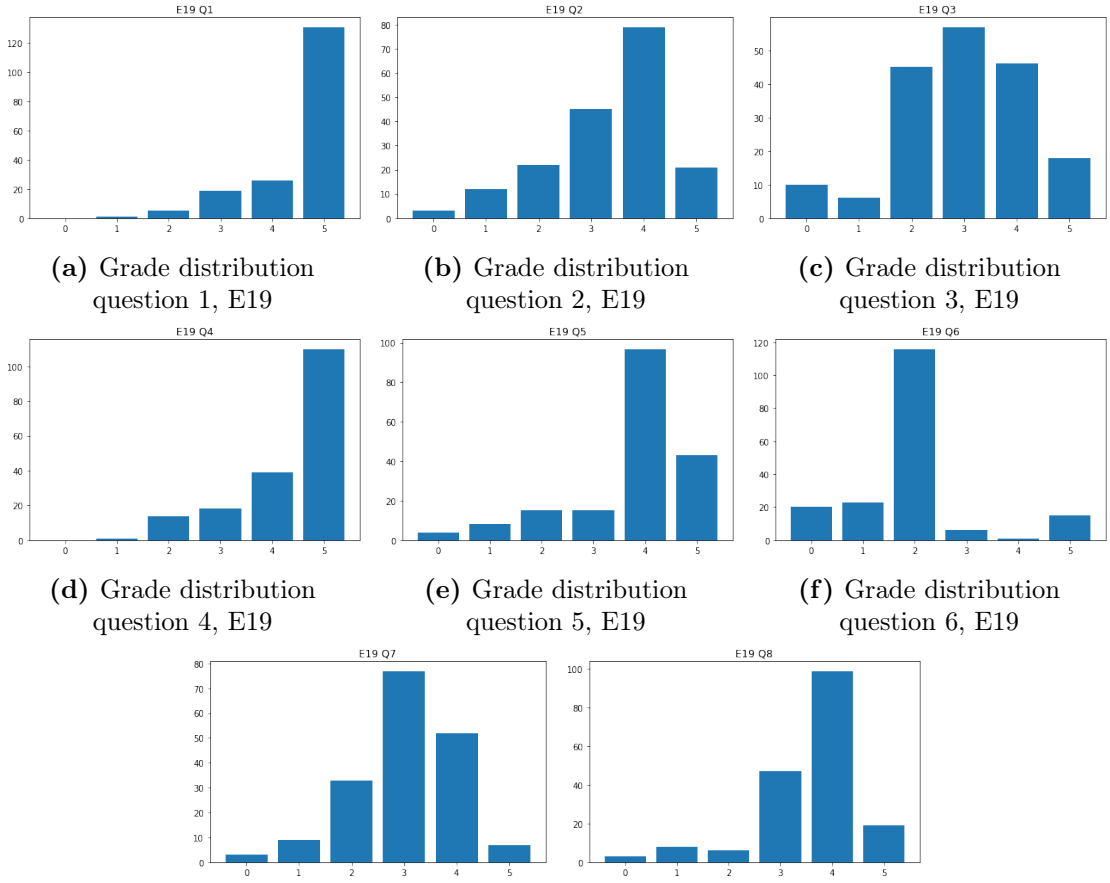


(a) Grade distribution question 1, E20 **(b)** Grade distribution question 2, E20 **(c)** Grade distribution question 3, E20

Figure 4.3: Grade distribution for each question in E20.

However, there are other distributions present as well. Several of the questions seem to have only one dominating grade. Questions 1 and 4 in E19 and 11, 14 and 16 in E18 are dominated by answers that have received the highest grade. In question 6 in E19, most students have received grade 2, while in question 4 in E18, most answers were graded 3. Question 2 in E18 has a distribution resembling something between a uniform and a multimodal distribution (several peaks are present). Question 10 in E18 has multiple answers that received either the highest or the lowest grade, but few answers that received any of the grades in between.

The presence of the various grade distributions can be challenging for several reasons. In machine learning, most models assume that the test and train set are independent and identically distributed. Thus, when splitting a dataset, it is often common to ensure that the two have similar distributions [70]. When using already graded exam



(a) Grade distribution question 1, E19

(b) Grade distribution question 2, E19

(c) Grade distribution question 3, E19

(d) Grade distribution question 4, E19

(e) Grade distribution question 5, E19

(f) Grade distribution question 6, E19

(g) Grade distribution question 7, E19

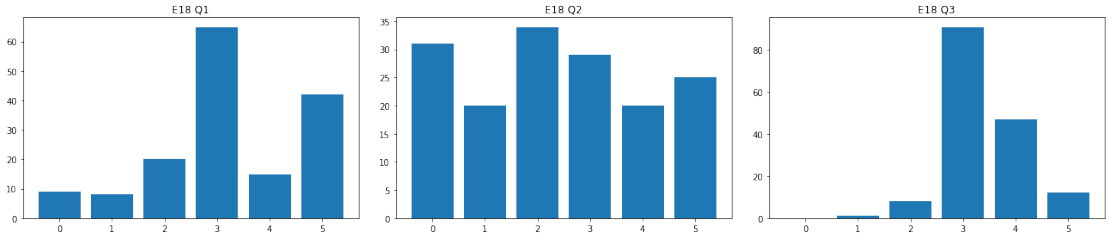
(h) Grade distribution question 8, E19

Figure 4.4: Grade distribution for each question in E19.

answers to build a machine learning model, it is no problem to split the answers into a test and training set with equal distributions. However, there is no guarantee that if using the same model on new exam questions, as is done in question general models, that the underlying distribution is remotely similar to the one used during training. By inspecting the grade distributions in these three exam answers, it is clear that expecting the same distribution for all questions would be unwise.

Furthermore, evaluation metrics can easily be affected by a skewed distribution. As discussed in Section 2.5.1, accuracy and error can become unreliable measures when the number of documents in one category is much smaller relative to the total number of documents in the document collection.

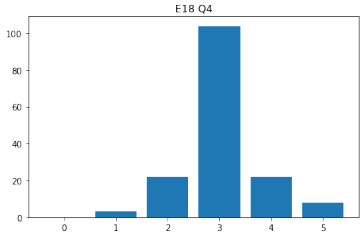
To summarise, a distribution favouring high grades seems most common, though it does not always apply. Furthermore, though these distributions were found in these three datasets, it does not conform to a ground truth for how future distributions will look. A new exam can lead to new distributions. Thus, there might be a need to develop methods more fitted to specific questions rather than to have a general model that should work optimally for all questions when attempting grade prediction.



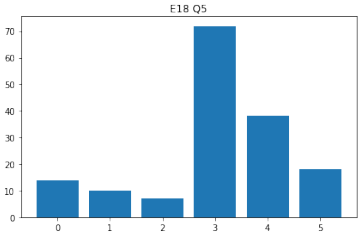
(a) Grade distribution question 1, E18

(b) Grade distribution question 2, E18

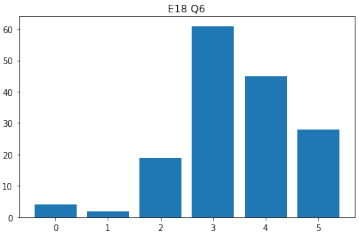
(c) Grade distribution question 3, E18



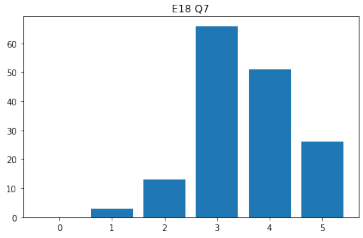
(d) Grade distribution question 4, E18



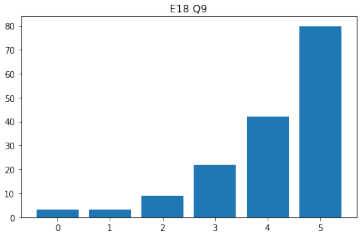
(e) Grade distribution question 5, E18



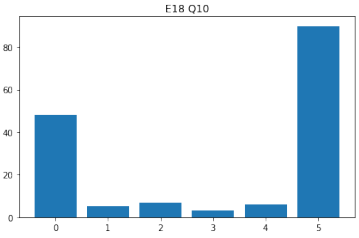
(f) Grade distribution question 6, E18



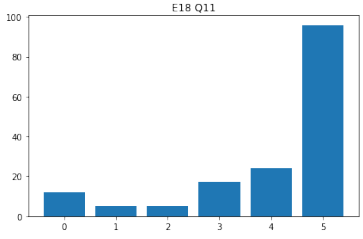
(g) Grade distribution question 7, E18



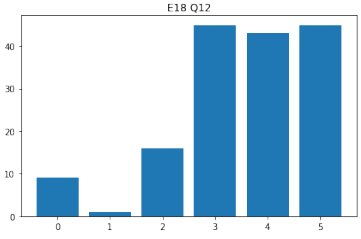
(h) Grade distribution question 9, E18



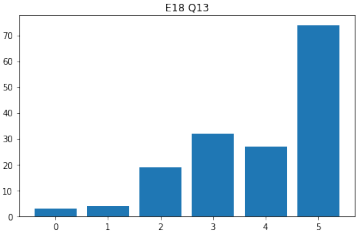
(i) Grade distribution question 10, E18



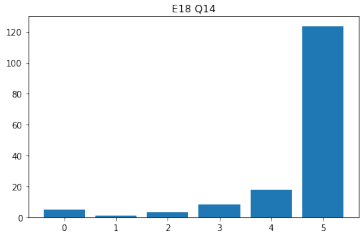
(j) Grade distribution question 11, E18



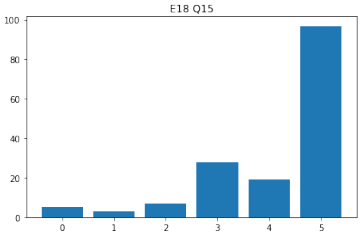
(k) Grade distribution question 12, E18



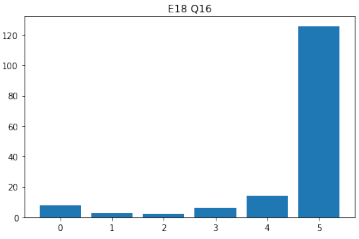
(l) Grade distribution question 13, E18



(m) Grade distribution question 14, E18



(n) Grade distribution question 14, E18



(o) Grade distribution question 16, E18

Figure 4.5: Grade distribution for each question in E18.

4.3.2 Answer Length

A common feature often mentioned in the literature is answer length and its relation to grades. This section will conduct an analysis by comparing the word count to the grades given by the sensor. The analysis was conducted after the completion of the data cleansing process. The following shows the results from this analysis for each of the three exams. The *Bokmål DS* was used in all analyses. For each of the three datasets, the word count is displayed using four different statistical measures; mean, median, minimum and maximum word count. All measures are rounded up to the nearest whole number. The measures are calculated from three different points of view:

1. Word count per grade
2. Word count per question
3. Word count per grade for each question

Tables 4.4, 4.5 and 4.6 shows the word count measured in mean, median, minimum and maximum for each grade in the three datasets. Tables B.1, B.2 and B.3 in Appendix B holds the result of the number of words per question for E20, E19 and E18 respectively using the same measurements. Furthermore, Appendix C shows the word count for each grade in each answer given to every question. The tables also includes the number of answers receiving each grade. The tables are organised such that the data in each row is connected to either the question or the grade listed in the first column. The following abbreviations are used in the column names:

- **Q**: Number of questions
- **A**: Number of answers for that specific question or receiving that specific grade
- **W.C.**: Word count
- ***Grade* {0-5}**: How many students received that specific grade

Table 4.4 shows how the answer length relates to the grades in E20. By inspecting the numbers, one can see that for each grade, both the mean, median and minimum word count increases as the grade increases, and the gap between the measures for each grade is relatively big. The maximum word count increases for grades 1-4, though it is slightly lower for answers graded 5 than those graded 4. However, this pattern can indicate that the word count positively correlates with the grades. The longer the answer tends to be, the higher the grade.

A similar pattern is present for the answers in E19, but it is not as prominent as in E20. By inspecting Table 4.5, one can see that the mean word count is increasing with the grades 0-3, and then again from grade 4 to grade 5. Answers graded 4 has an average word count slightly lower than answers graded 3. However, the median word count is constantly increasing. By looking at the maximum word count, the

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	2	1	1	1	1
1	5	158	112	41	399
2	40	188	184	88	484
3	217	299	267	110	963
4	336	404	370	133	1049
5	39	484	459	263	1036

Table 4.4: Statistics describing answers receiving the different grades in Bokmål DS E20.

longest answer is, in fact, graded 3. This might indicate some outliers in terms of answer lengths resulting in an overall high average answer length for answers graded 3. Grades from 2-5 all have answers longer than 500 words, which is way above the mean word count. Contrary to E20, all minimum answer lengths remain fairly low, though the minimum answer length for grade 5 is still higher than for the other questions. The biggest difference in answer lengths is found when grouping grades 0-2 and 3-5 and comparing the mean and median measurements.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	43	25	9	1	144
1	68	62	37	9	299
2	256	93	71	13	501
3	284	150	122	12	621
4	439	142	130	15	595
5	364	153	138	25	609

Table 4.5: Statistics describing answers receiving the different grades in Bokmål DS E19.

The word counts found in E18 also share similar characteristics with E20. This is presented in Table 4.6. The mean, median and maximum word count for an answer is always greater than the word count of answers graded one grade lower. Again, this might indicate a positive correlation between the number of words in an answer and the grade. Nevertheless, as one can see from the minimum word count, there are still very short answers receiving the highest grade.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	151	31	26	0	134
1	72	38	31	2	159
2	191	56	49	7	191
3	649	79	67	6	264
4	431	87	74	10	355
5	891	93	78	6	460

Table 4.6: Statistics describing answers receiving the different grades in Bokmål DS E18.

Tables 4.4, 4.5 and 4.6 indicates that the number of words used in an answer might be positively correlate with the grade. However, these tables do not differentiate between the different questions. By inspecting the word count for each individual question and how it might relate to the distribution of grades, one can get a more

clear indication of whether this conclusion can be drawn. Appendix C displays the same tables for each question in each dataset. The tables show that the same pattern found in the above tables is still present in many of the questions.

Based on the pattern found in the tables, the correlation between the answer lengths and the answers grades were calculated. The result showed that there was, in fact, a positive correlation between the number of words and the grades. The correlation coefficient between length and grades for each question is presented in Table 4.7. The overall correlation in each dataset is also shown.

	Dataset		
Question	*E20*	*E19*	*E18*
All questions	**0.46**	**0.29**	**0.30**
1	0.52	0.02	0.19
2	0.53	0.48	0.29
3	0.37	0.66	0.26
4	-	0.19	0.35
5	-	0.43	0.46
6	-	0.33	0.38
7	-	0.20	0.28
8	-	0.41	-
9	-	-	0.40
10	-	-	0.29
11	-	-	0.18
12	-	-	0.51
13	-	-	0.56
14	-	-	0.39
15	-	-	0.43
16	-	-	0.29

Table 4.7: Correlation between the word count and the grades for each question in each dataset, including the overall correlation when not looking at any specific question.

The correlation coefficient measures the strength of the linear relationship between two variables. 0 indicates no linear relationship, +1 a perfect positive linear relationship and -1 a perfect negative linear relationship. According to Ratner [71], a value between 0 and 0.3 indicates a weak positive relationship, a value between 0.3 and 0.7 indicates a moderate positive linear relationship, and between 0.7 and 1.0, there is a strong positive relationship.

The values in Table 4.7 indicates a moderate positive linear relationship between the word count and the grade in E20 and E18, and a weak positive relationship between the word count and the grades in E19. The correlation coefficient for individual questions varies, but it remains positive in all cases. For some of the questions, the coefficient is higher or equal to 0.4. This applies to questions 1 and 2 in E20, 2, 3, 5 and 8 in E19 and 5, 9, 12, 13 and 15 in E18. Thus, from the table, one can conclude that the answer length can be an important feature for classifying and separating answers of different grades.

4.4 Challenges Related to Automatic Grading

Section 4.2 listed challenges related to cleaning the dataset. This section focuses on potential challenges related to performing automatic grading and similar activities using the three described exams. The answers in the datasets will be compared to those typically used in the literature of automatic grading. This thesis aims to explore opportunities that can help in the grading process of future exams, rather than perfecting a model that works optimally on the given answers. The three datasets can be viewed as somewhat representative of future exam questions and answers in the course. That is, the questions and answers are most likely renewed for each exam, but the challenges found and mentioned can often be generalised to apply to future exams as well. Some challenges mentioned in this section are identical to those listed in Table 4.2, but are now discussed in terms of how they might affect future experiments. Identified challenges are presented in Table 4.8.

Id	Challenge
CH1	Norwegian answers
CH2	Presence of different languages
CH3	Multilingual answers
CH4	Evaluating performance for each question
CH5	Less data when evaluating
CH6	Longer and more varied answer lengths
CH7	No clear correct model answer(s)
CH8	Different question types
CH9	New questions each exam
CH10	Data format
CH11	Mixture of natural language and programming code
CH12	Presence of spelling mistakes
CH13	Skewed and multiple grade distributions
CH14	No benchmark

Table 4.8: All identified challenges, tagged with an ID.

The first challenge detected, CH1, regards the primary language used in the datasets; Norwegian. During the literature study, it was found that most research was conducted using English datasets. Some studies were also found on other languages, like Indonesian, Turkish and Arabic [51, 52, 72]. However, as far as this author is aware, there has not yet been any studies on automatically grading the content of Norwegian exam answers. That means that there is still uncertain how well earlier described methods apply to Norwegian answers. Several approaches described in Chapter 3 uses online resources only developed for the English language.

Another challenge is the presence of different languages (CH2). As mentioned in Section 4.2, answers were written in one of three different languages; Norwegian Bokmål, Norwegian Nynorsk or English. That implies that no specific language model can be applied to the entire dataset. Furthermore, comparing answers of different languages using a bag-of-words model would not be possible without first translating the text or specifying synonym words between the languages. A Nor-

wegian sensor with basic knowledge of English will have no problem reading and understanding all three languages, but a computer will not be able to compare two texts where one is written in Norwegian and another in English without extensive preprocessing first applied. The literature gave no example of an automatic grading model or system able to handle answers written in different languages. As mentioned, this challenge has been addressed in this thesis by only including answers written in Norwegian Bokmål. However, in a real-life situation, answers might still be written using different languages.

In addition to having different answers written in different languages, many of the answers used a mixture of Norwegian and English words (CH3). After filtering out all English answers, there were still several answers marked by polyglot as having English as most likely language and Norwegian as the second most likely language, or Norwegian as most likely language and English as the second most likely language. In practice, this means that polyglot cannot establish with certainty what language the text is written in as it contains words written in more than one language. It is also worth mentioning that this does not mean that these answers were the only ones containing a mixture of Norwegian and English words, just that the presence of both languages was severe enough compared to the text length to make polyglot uncertain in the choice of determining the language. Statistics showing the number of answers not characterised as only Norwegian are presented in Table 4.9. The table displays how many answers were marked as English as the most likely and second most likely language and how big percent of the total number of answers this applied to. The author has found no references to researchers that have worked with bi-lingual answers.

Exam	# **En** 1^{st} **lang**	% **En** 1^{st} **lang**	# **En** 2^{nd} **lang**	% **En** 2^{nd} **lang**
E20	0	0.0%	117	18.3%
E19	70	4.9%	225	17.8%
E18	87	3.7%	406	17.2%

Table 4.9: The table shows how many answers (in number and percentage of total answers) for each exam that had answers marked with English as either most likely or second most likely language.

Three datasets were given to analyse and experiment with. However, as the questions are not connected and evolve around different topics, one might have to consider each question as a separate domain. This is the case of the question specific models discussed in Chapter 3. That means that when analysing how well a given method has performed, the method must be measured for each question, not each exam (CH4). This complicates model building and analysis, as it has to be performed a total of 26 times if tested on each question. An approach that works well for one specific question might not perform as well for another. Similarly, attributes that might be important in grading one specific question can be irrelevant for another question. By only investigating the performance of models for a single question, the data available for each model also becomes considerably smaller; each question only has between 155 and 212 answers (CH5). Though there are examples of the same question being asked in more than one exam, merging all answers to those questions will most likely not be a good solution as what is considered a sufficiently

good enough solution to a question might vary for each exam.

As defined in Section 3.1, most ASAG systems deal with answers having an answer length of one phrase to one paragraph when the task is to evaluate the content of the answer. AGE systems usually deal with answers that range from two paragraphs to several pages. However, these systems focus on the writing style, not the content of the text. As presented in Table 4.1, the answer lengths in these exams are considerably longer, with E20 having an average of over 350 words per answer (E20). In addition, the answer length for answers to the same question also varies considerably more. The longest answer in question 1 in E20 is 1049 words while the shortest is only one word. This is listed as challenge CH6 in Table 4.8.

Few of the identified systems and models handled answers of this length, the exceptions being two corpus-based methods, Apex and Atenea, mentioned in Section 3.2.3. That means that there is great uncertainty about how well most identified work would adapt to datasets such as the three used in this thesis. Furthermore, several identified systems rely on a reference-based approach where the student answers are compared to one or several model answers. This relies on questions having a clear, correct answer or answers, which is not the case for many of the questions asked on these exams (CH7). Though the course coordinator has suggested solutions for some answers, it was also stated that those solutions were only suggestions, and other answers could be regarded as equally valid. In addition, many of the suggested solutions were not actual answers rather a list of what the answers were expected to contain. Thus, using a response-based approach is more suitable for the experiments in this thesis.

There are different types of questions being asked during the exams. Several properties can be used to characterise them. Table 4.10 classifies the different questions according to which properties they fit. Some questions fit more than one property and are thus listed several times. Different types of questions might respond differently to different approaches or methods within automatic grading (CH8). For instance, questions that expect the student to use examples when explaining their answer open up for a variety of different answers, where two completely different answers might receive the same grade. Though all questions asked in the three exams fits at least one of the properties listed here, new exams contain new questions that might be characterised differently (CH9).

Property	**E20**	**E19**	**E18**
Expects some programming code	-	2, 6	4, 5, 10, 13
Expects to use example	-	6	4, 5, 13
Expects mentioning concrete concepts	1	1, 8	7, 13
Open-ended questions	-	7	15
Comparison and discussion questions	2, 3	3	3
Explain single concept	-	2, 6	2, 5, 6, 10, 11, 12, 14, 16
Explain multiple concepts	-	4, 5	1, 2, 3, 4, 9

Table 4.10: All questions from the three exams classified according to the defined properties. Some questions inhabit several properties.

Though few of the questions can be considered completely closed, some questions are

on the complete opposite end of the scale. The questions classified as open-ended in this setting refer to questions where very different answers might have received the same grade as the question opens up for a variety of allowed answers. Two questions fall within this category. Question 7 in E19 asks the students to list *up to* five good advises for developing a search application given some requirements and argue why their advises are important. Question 15 in E18 lets the students choose if they want to explain *either* REST or GraphQL[3]. As these questions open up for various answers, a response-based approach might fail to separate answers by their quality as the terms used in the different answers might vary too much.

Another challenge, addressed in Section 4.2, is that the dataset is formatted by Inspera (CH10). A solution for addressing the Inspera formatted answers was provided in the same section. However, the procedure is not able to preserve all student-written text. Thus, some parts of the answers are lost, which again can affect the results found in the experiments. Compared to the description provided by other researchers, there has been no mentioning of datasets affected by the same amount of noise in the data.

As the course IT2810 is concerned with web development, many answers include some programming code. In fact, most answers include example programming code, whether or not it was asked for in the question (CH11). In the literature, research focusing on automatic grading of programming code or automatic grading of natural language answers were both identified, but no work on evaluating answers including a mixture of both, where discovered. To the author's knowledge, handling such answers is this far an unexplored topic. Though focusing on this aspect is outside the scope of this thesis, it does not hinder the results from being affected by programming code and syntax.

When inspecting the student answers, several answers contained spelling errors. An assumption is that the text in these datasets contains more than an average amount of spelling mistakes (CH12). This is due to the exams being graded based on the content in the answers rather than correct language and linguistics. As the students are under time-pressure, many might not bother to check their language for spelling mistakes and errors. Furthermore, during data exploration, many examples were found of students who had wrongly used compound words. One such example was the word "komponenthierarki". Some students wrote the word as an open compound (spelt as two words, e.g., "komponent hierarki"), closed compounds (joined to form a single word, e.g., "komponenthierarki"), or hyphenated compound (e.g., "komponent-hierarki"). If utilising a bag of words model without addressing this problem, the three variations would be measured to have a similarity equal to 0 (or distance equal to 1) if using cosine similarity. However, a human sensor would count all three variations as equal.

CH13 is concerned with the overall skewed grade distributions and the presence of multiple different grade distributions when viewing the grades given answers for each individual question. Challenges related to grade distributions has been discussed in Section 4.3.1.

[3]REST is an application programming interface. GraphQL is a query language for application programming interfaces

To conclude, several challenges have been detected during the initial dataset analysis. Many of these challenges have not been previously addressed in the literature. In Section 3.1, a distinction was made between typically answers used in AGE and ASAG questions. Based on the identified challenges, it is clear that the questions often bear more similarities with questions use in AGE then in ASAG. Many are considerably longer then one paragraph. Sometimes the question specifically request the students to include examples. However, the students often include them without it being requested. Other questions asks the student to discuss and reflect. As a result, there are no identified benchmarks for measuring the performance of the methods examined against earlier work (CH14).

Chapter 5

Experiments

This chapter examines the use of techniques within information retrieval and text mining to assist an efficient and fair grading process. The focus will be on examining techniques that uses terms as features and investigate whether or not these can be used to distinguish between quality and low-quality answers. The goal will be to examine how different available, off-the-shelf techniques can be utilised to automatically assessment Norwegian exam answers in the topic Computer Science.

Section 5.1 presents the selected experiments and how each of them will be evaluated or analysed. The dataset preprocessing procedure, including operations as lexical analysis, stopword removal, lemmatization and stemming, will be explained in Section 5.2. The three selected experiments will be presented in sections 5.3, 5.4 and 5.5. The first experiment is an analysis of the most common words used in the answers and whether or not important terminology can be identified amongst those. The goal is to investigate if the language used in answers of different quality differs in terms of terminology words and if these words can be used to differentiate between good and poor quality answers. The second experiment will rank answers based on a query created from words included in the student answers to see if the grade hierarchy is represented in the ranking order. The last experiment tests a commonly used clustering algorithm within automatic grading, k-means, to see how answers of different grades are clustered and if any patterns can be detected in the clusters. The experiments are not conducted using every question in the three datasets. Rather, a few questions have been selected to be examined and tested for each experiment.

5.1 Selecting Experiments

Throughout the literature, a variety of different methods were identified that have contributed to the work of automatic grading. These methods have aimed to automatically grade or label answers or aid the sensor through the grading process. The goal of this thesis is to look into how off-the-shelf information retrieval and text mining techniques can be employed to address this topic. With this in mind, three experiments have been identified to be examined in depth.

5.1.1 Suggestion 1: Identifying Terminology

As shown in Section 3.2, research on automatic grading is often concerned with answer terms as they have been shown useful for identifying similar answers. According to the course coordinator, exam answers, in the field of Computer Science, are often graded based on whether or not the students mention the correct terminology related to the question. Thus, a hypothesis is that the higher the answer is rated, the more terminology is included in the answer. This laid the foundation for the following experiment: *Is it possible to identify question-related terminology, and can it be used to distinguish answers of different quality*? In order to investigate this, the following four questions were formulated to be examined in detail:

1 *How much terminology can be found amongst the most common words in the best graded answers?*

2 *Are the most commonly used words in the best answers less common in lower graded answers?*

3 *Are the most common words in the best answers representative words for what should be included in an answer?*

4 *Is it possible to retrieve the same terms found in the best answers when applying the same technique to all answers?*

This experiment distinguishes between highly graded answers (the best answers) and lower graded answers. Answers that belong to the category of highly graded answers have received the grade 5 or 4. Answers characterised as lower graded answers are answers that received either 0, 1 or 2.

In cooperation with the course coordinator, a few questions were chosen to examine in more detail. These were questions where including terminology was expected in order to receive a high grade. In Table 4.10, they are listed under the property *Expects mentioning concrete concepts*, and evolves around *State Management* or *Responsive Design*. The questions are also listed in tables A.1, A.2 and A.3 in Appendix A. In addition to these five questions, a sixth question not bearing the same characteristics was chosen, question 2 from E18. The question was chosen as a counterpart to the other questions to see if the results differed when choosing a question with other characteristics. The question also has a significantly different grade distribution from the other five which was why that particular question was picked. Question 2 from E18 can be found in Table A.3 in Appendix A.

5.1.2 Suggestion 2: Ranking and retrieval

At the core of information retrieval lies ranking and retrieval. The general ranking process was earlier explained in Section 2.3.2. Ranking and retrieval consist of two main tasks: a logical framework for representing documents and queries and a ranking function that computes a rank for each document based on some given query

[7]. Before applying the ranking function, the documents must first be indexed. Afterwards, a ranking algorithm is used to determine how well a given document matches a user-specified query. The documents that are deemed most similar to the query are then retrieved. How many documents to retrieve can be specified by the system. An evaluation metric is used to determine how well the system performs.

This analogy can also be used in an automatic assessment scheme. In Chapter 3, several examples were identified where systems and researchers chose to measure the similarity between the student answers and a model answer to determine which grade to assign to the answers. In those cases, the researchers had access to or constructed one or several correct model answer(s). However, as most such model answers are either unavailable and unsuited to the given exam questions, this experiment aims to extract important words from the answers to form a query for which the answers will be ranked against. This will be done on a per-question level. In order to perform indexing, ranking and retrieval, a search engine library, `Whoosh!`, will be used. The library handles both text indexing and allows for searching the index.

As only terms found in the student answers are used to rank the answers, the solution can be considered a response-based approach. However, the solution also builds on elements from a reference-based approach. The query can be considered a reference for which all student answers are compared against. Thus, the approach can be seen as a hybrid of the two.

Some questions have been selected in order to analyse and evaluate the experiment. However, the experiment can easily be replicated for all questions. The answers were picked based on some predefined requirements:

R1 At least one question from each exam

R2 A variety of different answer lengths

R3 Questions with different properties

R4 Different grade distributions

R5 Questions where terminology is expected to be defining for high grades

The chosen questions and how they relate to the selected requirements are shown in Table 5.1. W.C. refers to the average word count in the question. H.G. and L.G. are how many of the answers were graded as either 4 or 5, or 0, 1 or 2, respectively.

To meet R2, questions with a variety of different average word lengths where chosen. The average word length range from 390 words at the highest and 127 words at the lowest. Furthermore, in terms of R3 and R5, question 1 in E20 and question 13 in E18 are both characterised with the property *expects mentioning concrete concepts* in Table 4.10. Thus, terminology is expected to be defining for answers that received a high grade, as described in Section 5.1.1. In addition, question 13 is also characterised by the two properties *expects some programming code* and *expects to use example*. Question 2 in E20 and 3 in E19 are both *comparison and discussion* questions. In addition, question 7 from E19 was chosen, which has been

Q	E	Properties	W.C	H.G.	L.G.
Q1	E20	Expects mentioning of concrete concepts	390	119	27
Q2	E20	Comparison and discussion	355	143	11
Q3	E19	Comparison and discussion	127	64	61
Q7	E19	Open-ended question	238	59	45
Q13	E18	Expects mentioning of concrete concepts Expects to use example Expects some programming code	150	101	26

Table 5.1: Questions chosen for the ranking and retrieval experiment and how they related to the selected requirements.

characterised as an open-ended question. In the question, the students are asked to list up to five good pieces of advice for designing a search engine and argue why those pieces of advice are impotent. This question drastically differs from the other questions, as it opens up for more varied answers and thus a brother vocabulary, where answers receiving high grades might look very dissimilar. The results for question 7 in E19 are expected to be drastically worse than for the other questions. However, it is still an interesting question to investigate as the results can indicate of the solution is suitable for all questions types or not. Together, these questions represent five of the seven properties listed in Table 4.10.

As in the first experiment, presented in Section 5.1.1, this experiment also distinguishes between highly graded answers, or the best answers, and lower graded answers. Answers that belong to the category as highly graded answers have received the grade 5 or 4. Answers characterised as lower graded answers are answers that received either 0, 1 or 2. Thus, when selecting questions based on different grade distributions, the evaluation was made based on the distribution of high and low grades. This will be further explained when the evaluation of the experiment is discussed.

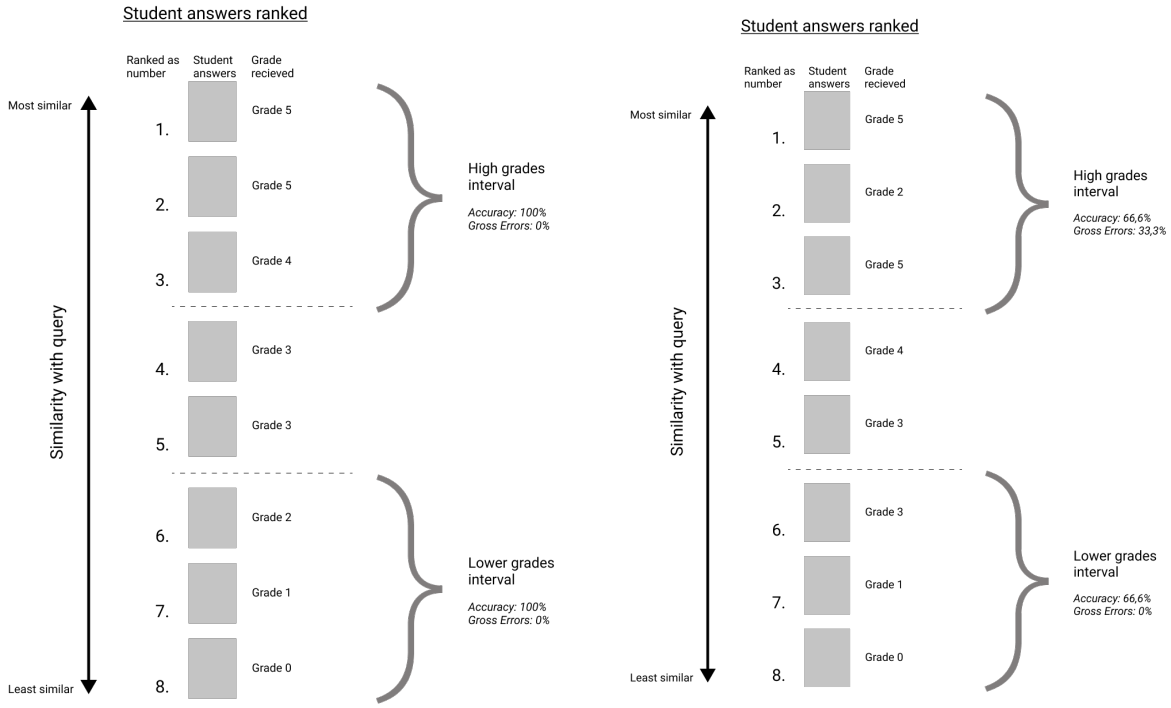
To meet R4, it was chosen to include some questions where the majority of the answers received a high grade and some questions where the grades were more uniformly distributed between low and high grades. Question 1 and 2 in E20 and 13 in E18 all have a grade distribution where the vast majority of the students received a high grade and a smaller number of students who received a low grade. For questions 3 and 7 in E19, the grades are more equally distributed in terms of low and high grades. Among all the questions picked, at least one from each dataset has been chosen; thus, all requirements were met.

Evaluating the Experiment

Evaluating how well the experiment works poses a new challenge. Few existing evaluation metrics are well suited for the proposed experiment. Typically, document retrieval systems are evaluated using precision and recall, as described in Section 2.5.2. However, these systems usually only retrieve a subset of all documents and display them to users. However, in this experiment, all documents are retrieved and ranked. Thus, a new and more tailored metric of evaluation is called for. The

evaluation metrics presented are based on accuracy and error, described in Section 2.5.1, but adapted to fit the experiment and to account for the grade distributions.

Optimally, the student answers would be ranked according to their grades. That means that all answers graded 5 would be ranked first, then all answers graded 4, and in the end, all answers graded 0 would be found. An example of such a ranking is displayed in Figure 5.1a. In the example, a question with eight student answers are ranked according to a query. The better the grade, the higher rank the answer received, thus the more similar it was to the query. For this rank to be achieved, it requires that the system works perfectly and that the grades determined by the sensor are "correct". The evaluation metric constructed is based on the optimal rank.



(a) Optimal ranking

(b) Suboptimal ranking

Figure 5.1: Example illustration showing the high grades and lower grads interval and corresponding accuracy and gross error values. Figure 5.1a show an optimal ranking of 8 graded student answers. Figure 5.1b show a suboptimal ranking of 8 graded student answers.

In Figure 5.1a one can see that two intervals are defined, *High Grade Interval* and *Lower Grades Interval*. A formal definition of the high grade interval is expressed in Definition 5.1.1. A formal definition of the lower grades interval can be found in Definition 5.1.2.

Definition 5.1.1. For a given set of student answers $SA = \{sa_i, i = 1, 2, ..., n\}$, where each sa_i stands for a student answer, and each student answer has received a grade from the set $G = \{g_i, i = 0, 1, ..., 5\}$, the student answers sa_i that belong to the high grade interval is the first k ranked student answers, where k is the number of student answers that received a grade g_i by a human evaluator, where $g_i \in G' = \{g_i, i = 4, 5\}$

Definition 5.1.2. For a given set of student answers $SA = \{sa_i, i = 1, 2, ..., n\}$, where each sa_i stands for a student answer, and each student answer has received a grade from the set $G = \{g_i, i = 0, 1, ..., 5\}$, the student answers sa_i that belong to the lower grades interval is the last k ranked student answers, where k is the number of student answers that received a grade g_i by a human evaluator, where $g_i \in G' = \{g_i, i = 0, 1, 2\}$

Within each of the intervals, the accuracy and *gross* error is calculated. The equations for calculating accuracy and gross errors in the high grad interval can be found in equations 5.1 and 5.2 The equations for calculating accuracy and gross errors in the lower grades interval can be found in equations 5.3 and 5.4.

$$Accuracy_{HG} = \frac{Number\ of\ answers\ graded\ 5\ or\ 4\ in\ high\ grade\ interval}{Total\ number\ of\ answers\ graded\ 5\ or\ 4} \tag{5.1}$$

$$GrossError_{HG} = \frac{Number\ of\ answers\ graded\ 2,\ 1\ or\ 0\ in\ high\ grade\ interval}{Total\ number\ of\ answers\ graded\ 5\ or\ 4} \tag{5.2}$$

$$Accuracy_{LG} = \frac{Number\ of\ answers\ graded\ 2,\ 1\ or\ 0\ in\ lower\ grades\ interval}{Total\ number\ of\ answers\ graded\ 2,\ 1,\ 0} \tag{5.3}$$

$$GrossError_{LG} = \frac{Number\ of\ answers\ graded\ 5\ or\ 4in\ lower\ grades\ interval}{Total\ number\ of\ answers\ graded\ 2,\ 1,\ 0} \tag{5.4}$$

Figure 5.1b shows an example where the answers were not optimally ranked. In the example, the accuracy of the high grades interval is 66.6%, as only two out of the three answers graded 4 or above were found in the interval. The gross error of the high grad interval is 33.3% as one of the answers found in the interval would have been part of to the lower grads interval in an optimal ranking. Similarly, for the lower grades interval, the accuracy is 66.6% as two out of three answers that should be ranked inside the interval were identified. However, the gross error is still 0% as the third answer in this interval received the grade 3, and thus does not belong to any of the two intervals.

The aim of the evaluation is not to accomplish the optimal ranking as shown in Figure 5.1a, but to investigate how well the ranking algorithm manages to distinguish between quality and low-quality answers. Because of that, answers graded 3 are not considered in the experiment as they are often counted as "average" answers in terms of quality.

5.1.3 Suggestion 3: Clustering

Clustering is a common technique within text mining and has been employed by some researchers within the field of automatic grading. This experiment will create

clusters of similar answers based on their content. The goal is to investigate how answers of different grades are represented in the different clusters, that is, whether answers of similar grade are grouped together or if the clusters consist of answers with all grades.

The answers will be clustered based on their similarity (distance) to all other answers. In Chapter 3 two different clustering approaches were presented. Suzen et al. [48] used a reference-based approach where answers where clustered based on their similarity with the model answer. Klein et al. [43] used a response-based approach where student answers were clustered based on their similarity to all other answers. Due to the lack of sufficient model answers, a response-based clustering approach will be used in this experiment.

A benefit of choosing an unsupervised learning algorithm over a supervised one is that it is more suited for the real-world situation. During a grading process, there will be no access to a ground truth. If shown to achieve good results, the algorithm can easily be applied to new exams.

As in the two previously explained experiments, a few questions have been chosen for the clustering experiment. One question has been selected from each exam: question 1 from E20, question 5 from E19 and question 13 from E18.

Evaluating the Clusters

Clustering is an unsupervised machine learning algorithm. That is, it aims to discover natural groupings in data. Most unsupervised machine learning experiments have no access to what the "correct" outcome of the algorithm should be, making them hard to evaluate. That is why evaluating clustering algorithms often are seen as a challenging task [16]. However, in this case, the student grades can be used to conduct an external evaluation.

In external evaluation, clustering results are evaluated based on data not included in the clustering, such as known class labels, or in this case, student grades. Several external evaluation measures exist. For this experiment, the purity measure, described in Section 2.5.3 will be used as part of the evaluation. Furthermore, the clusters will also be evaluated through a quantitative analysis where the most important features for some selected clusters will be chosen to examine in-depth.

5.2 Preparing the Dataset

Before the experiments were conducted, the exam answers were first processed using text preprocessing techniques. The following four techniques were used:

1. Lexical Analysis
2. Elimination of Stopwords

3. Lemmatization
4. Stemming

5.2.1 Lexical Analysis

Lexical Analysis is a collective term that incorporates several procedures, but the objective is to handle digits, hyphens, punctuation marks, and casing of letters to create tokens. Parts of the lexical analysis has already been conducted and is described in Section 4.2. This included handling digits, hyphens, punctuation marks and other special characters. At this point, one more step was included, namely handling letter casing (lower and upper case). All uppercased letters were changed to lower case. By not transforming all uppercased letters to lower case, words like "spell" and "Spell" would not be considered equal by a string comparison algorithm.

5.2.2 Stopword Removal

The second step included the removal of stopwords. These words seldom bring valuable information in terms of describing the content of a text. To remove the stopwords, two iterations were conducted. First, a trained language model created by `spacy` was applied to analyse the answers. The language model had previously been trained on large Norwegian news corpora. This makes the model able to recognise different types of words, like stopwords. By utilising the model, most stopwords were removed. However, the language model was unable to recognise all stopwords. Thus, additional stopwords were removed by creating a list over known stopwords and then removing all words that appeared in that list. The list was created by combining two open-sourced lists containing Norwegian Bokmål and Nynorsk words and some added by the author to cover common abbreviations. The list can be found in Appendix D.

5.2.3 Lemmatization and Stemming

It is often normal to conduct either lemmatization or stemming when performing document preprocessing. In this project, both techniques were utilised. Neither the stemmer nor the lemmatizer were able to perform optimally alone; however, by utilising both techniques, the results improved.

Lemmatization was performed first. Like in the case of stopwords, the lemmatization was carried out by letting the language model infer the root of all words. Spacy's language models assign base forms to tokens using rules based on part-of-speech tags or look-up tables [73]. However, as the language model was trained on a Norwegian news corpus, many English written words did not have an identified root. Thus, several of the words were left untouched. This could have resulted in having the same word written with different endings not counted as equal. This was addressed by using stemming.

After lemmatization, stemming was conducted to ensure that most words would end up with the same stem. In order to apply stemming, the `Snowball stemmer` from the `NLTK` library was used. The stemmer is currently supporting 16 languages, including Norwegian. When applying the NLTK Snowball stemmer, it applies one main rule called R1:

> *R1 is the region after the first non-vowel following a vowel, or is the null region at the end of the word if there is no such non-vowel. But then R1 is adjusted so the region before it contains at least three letters.* [74]

The stemmer is unable to infer the language of a word. This meant that words words written in English would be processed using the same rule. As a result, several words were reduced to a stem that did not match any word in a dictionary. An example was the words "states" and "state", which both were reduced to "stat". However, this approach is still preferred as it would allow the two words to be comparable by a computer. A downside to this approach is that words of different meanings could have the same stem, though the original words are different.

When performing lemmatization and stemming, a dictionary holding the root or the stem of the words as a key and all original words as values was created to allow for look-up.

5.3 Experiment 1: Identifying Terminology

The overall goal of this experiment is to see if it is possible to identify question-related terminology and if that can be used to distinguish answers of different quality. This is done by examining the most common terms used in the student answers. Furthermore, the presence and frequency of terminology words amongst differently graded answers will be examined. Words defined as terminology relate to both the course topic, computer science, and the subject the question evolves around. In Section 5.1.1, six questions were identified that would be examined in this experiment.

5.3.1 Approach

To extract the most common terms from the answers, the answers were first indexed. The inverted indexes were created by using the library `Whoosh!`. Whoosh! allows a user to specify a schema. The schema specifies the fields of documents in an index and is the set of all possible fields in a document [75]. By passing the student answers to a field within the schema, Whoosh can index the text using a text analyser. Because all necessary document preprocessing had already been conducted, the `SimpleAnalyzer` was used. The analyser creates tokens by splitting on the space character. This means that each word becomes a token.

The answers to each question were divided into three groups: one containing all answers receiving 4 or 5, one containing all answers that received 0, 1 or 2 and

the last containing all answers. Then, the answers in each group were indexed by Whoosh!. In total, 18 inverted indexes were created, three for the answers to each question. Afterwards, the indexes were used to retrieve the most commonly used words. In this setting, the most commonly used words are not the words with the overall highest word frequency but the words mentioned in the most answers. Because the words were both lemmatized and stemmed, the extracted terms were, in some cases, hard to interpret. To make the words more understandable for the reader, the words were replaced with the shortest word found in the stemmed-lemmatized dictionary created in Section 5.2.3 when indexing the dictionary using the retrieved word.

5.3.2 Results and Analysis

The results from the experiment will be analysed through a qualitative analysis based on the identified words. Throughout this section, the questions formulated in Section 5.1.1 will be revisited in order. The results will be shown and discussed.

Q1: How much terminology can be found amongst the most common words in the best graded answers?

Using the method described above for the selected questions, the thirty most frequent words were extracted from the highly graded answers. These words are for each question, shown in Table 5.2. The words are sorted in decreasing order of frequency. Words considered terminology are marked in bold. This was decided through a manual inspection of the list. The column T shows how many of the top 30 words are considered terminology.

From the table, we can see that the majority of the top 30 words can be considered terminology words. On average, 21.5 out of the 30 most common words are considered to be terminology words. Question 2 from E18 does not distinguish from the other questions. Question 8 from E19 has less terminology words than the others, but most words are still considered terminology words.

An interesting fact is that some of the words that are not considered terminology appears in several of the top 30 lists, including "endre", "stort", and "små". That these words appear in the top 30 list of several different questions might indicate that they should have been removed during the elimination of stopwords as they probably do not help in distinguishing the different answers. Furthermore, in question 2 in E18 both the words "funksjon" and "function" are listed. These words mean the same, though one is written in Norwegian and the other in English.

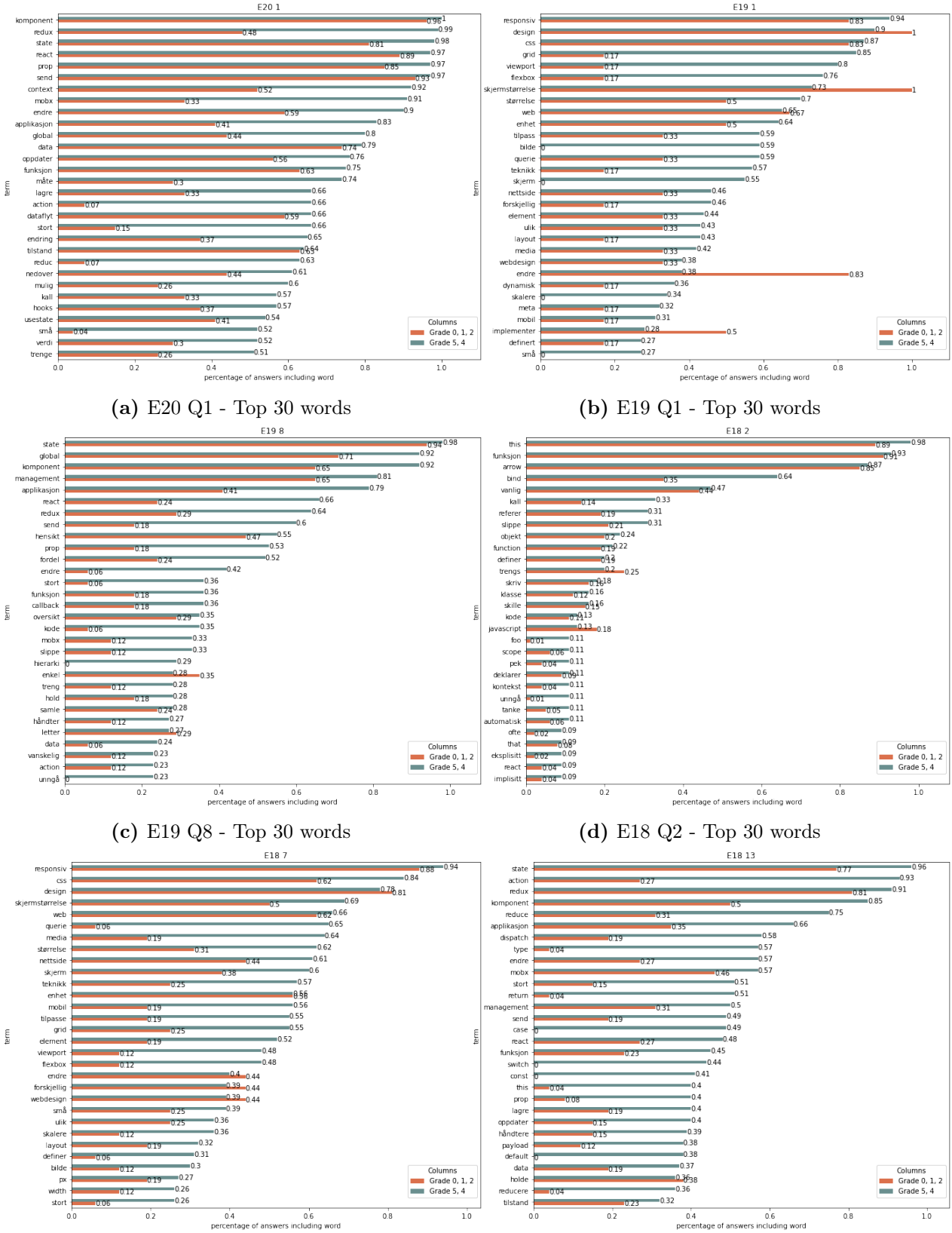
Question	Top 30 terms	T
E20 Q1	**komponent**, **redux**, **state**, **react**, **prop**, **send**, **context**, **mobx**, endre, **applikasjon**, **global**, **data**, **oppdater**, **funksjon**, måte, **lagre**, **action**, **dataflyt**, stort, endring, **tilstand**, **reduc**, nedover, mulig, **kall**, **hooks**, **usestate**, **verdi**, små, trenge	22/30
E19 Q1	**responsiv**, **design**, **css**, **grid**, **viewport**, **flexbox**, **skjermstørrelse**, størrelse, **web**, **enhet**, **tilpass**, **bilde**, **querie**, teknikk, **skjerm**, **nettside**, forskjellig, **element**, ulik, **layout**, **media**, **webdesign**, endre, **dynamisk**, **skalere**, **meta**, **mobil**, **implementer**, definert, små	23/30
E19 Q8	**state**, **global**, **komponent**, **management**, **applikasjon**, **react**, **redux**, **send**, hensikt, **prop**, fordel, endre, **funksjon**, stort, **callback**, oversikt, **kode**, **mobx**, slippe, **hierarki**, enkel, treng, hold, samle, håndter, letter, **data**, **action**, vanskelig, unngå	16/30
E18 Q2	**this**, **funksjon**, **arrow**, **bind**, vanlig, **kall**, **referer**, slippe, **objekt**, **function**, **definer**, trengs, skriv, **klasse**, skille, **kode**, **javascript**, **foo**, **scope**, **pek**, **deklarer**, **kontekst**, unngå, tanke, **automatisk**, **that**, **eksplisitt**, **react**, ofte, **implisitt**	22/30
E18 Q7	**responsiv**, **css**, **design**, **skjermstørrelse**, **web**, **querie**, **media**, størrelse, **nettside**, **skjerm**, teknikk, **enhet**, **mobil**, tilpasse, **grid**, **element**, **viewport**, **flexbox**, endre, forskjellig, **webdesign**, små, ulik, **skalere**, **layout**, definer, **bilde**, **px**, **width**, stort	21/30
E18 Q13	**state**, **action**, **redux**, **komponent**, **reduce**, **applikasjon**, **dispatch**, **type**, endre, **mobx**, stort, **return**, **management**, send, **case**, **react**, **funksjon**, **switch**, **const**, **this**, **prop**, **lagre**, **oppdater**, håndtere, **payload**, **default**, **data**, **reducere**, holde, **tilstand**	25/30

Table 5.2: 30 most frequent words from answers graded 5 and 4. Bold words are considered terminology words.

Q2: Are the most commonly used words in the best answers less common in lower graded answers?

The lists in 5.2 does not consider how many answers the words occurred in. Figure 5.2 shows the top 30 words used in the higher graded answers. For each word, the graphs show many of those answers contains that word measured in percentage, and how many answers graded 0, 1 or 2 contains the same word.

There are several noteworthy aspects of the graphs in Figure 5.2. First, one can see that the most common words in highly graded answers also appear in most lower graded answers. Furthermore, for almost all questions, the top 30 words appear in a larger percentage of the higher graded answers than the lower. The words "grid", "viewport" and "flexbox" which are the fourth, fifth and sixth most common words in E19 question 1, appear in 85%, 80% and 76% of the high graded answers, respectively, while only 17% of answers with a low grade contains these words. An assumption can thus be that these words are important in terms of separating answers from each other. This will be further elaborated when addressing Q3.



(a) E20 Q1 - Top 30 words

(b) E19 Q1 - Top 30 words

(c) E19 Q8 - Top 30 words

(d) E18 Q2 - Top 30 words

(e) E18 Q7 - Top 30 words

(f) E20 Q13 - Top 30 words

Figure 5.2: The frequency, measured in percentage, of common words in answers graded 5 or 4 are shown in blue. Their frequencies in answers graded 0, 1 or 2 is shown in green.

How many answers these top 30 words appear in varies from question to question. For question 1 in E20, the $30th$ most frequent word is present in 52% of the highly graded answers. For questions 1 and 8 in E19, the $30th$ most frequent word is only present in 28% and 23% of the answers. For questions 7 and 13 in E18, the number is 27% and 32%, respectively. The reason why the most frequent words are in a larger proportion of the answers for question 1 in E20 can be related to that the average number of words for each answer is considerably longer in the E20 dataset than in the other two. Out of the 30 most common words, the top 15 (E19 Q1), 11 (E19 Q8), 16 (E18 Q7), and 13 (E18 Q13) terms are present in more than 50 % of the highest graded answers. When looking at the lower graded answers, only 13 (E20 Q1), 10 (E19 Q1), 5 (E19 Q8), 6 (E18 Q7) and 3 (E19 Q13) out of the words are included in at least 50% of the answers.

The graph describing question 2 in E18 paints a slightly different picture. The percentage of answers containing the top 30 most common words from the highly graded answers are often as common in the lower graded answers as in the higher. Another interesting fact found by analysing the graph for question 2 in E18 is that only the top four terms are included in over 50% of the answers. This can indicate that there are other factors than simply the individual words mentioned in the answer that can account for the answer grade. However, it could also point to a large number of synonym words or spelling errors.

Q3: Are the most common words in the best answers representative words for what should be included in an answer?

In answering Q1 and Q2, it was found that the majority of the most common terms in highly graded answers can be considered terminology words. In addition, these words are found to appear in highly graded answers more often than in the ones that received a low grade. To determine if these words can be used to separate high and low-quality answers, another aspect to look into is whether the most common words are helpful features and representative for expected answers. There are several ways to examine if the words bear these qualities. To answer Q3, it will be looked into how the thirty most common words relate to a suggested solution provided by the teacher. Only some of the questions are compared with the model answer. This has to do with the availability of model answers. Furthermore, the words in the top 30 lists will also be compared to the words used in the question.

The teachers suggested solution for question 1 in E19 is given below. When comparing the words in the solution to the terms in the top 30 list for question 1 in E19, several terms are present in the solution, either directly or as synonym words. Given that the model answer had been processed using the same techniques as described in Section 5.2, the words that would map directly are highlighted. Synonym words are underlined.

> Websider som har dynamisk design og kan tilpasse layout mm til forskjellige enheter/skjermstørrelser.
>
> Teknikker:
>
> - Viewport (i meta-taggen) for å definere størrelse på det synlige vinduet
> - CSS mediaqueries
> - Velge bilder av forskjellig størrelse tilpasset enheten eller dynamisk størrelse på bildet
> - Bruk av dynamisk layout med css grid eller flexbox
> - Eller bruke rammeverk som støtter responsiv design (og kanskje er basert på det over...)

16 of the top 30 words are directly mentioned in the model answer. In addition, many of the other words can be regarded as synonyms of words found in the model answer. This is the case of the word "websider" which also can be referred to as "nettside", or "forskjellig" and "ulik". The word "mediaqueries" used in the model answer is not present in the top 30 list but, the two words "media" and "querie" are both found in the list. This can result from the fact that several students wrote the word as an open compound; "media queries". In addition, the word "endre" can be seen as a synonym to "tilpasse" if used in the correct context. The same goes for "vinduet" and "skjerm". A computer cannot automatically infer these connections. However, it is still clear that the words found in the top 30 list for question 1 in E19 are representative words for what should be included in a good answer.

An interesting question to compare with its model answer is question 13 in E18. The question asks the students to explain state management and give examples of its implementation using code. The model answer is given below. Words directly included in the top 30 list are highlighted, and synonym words are underlined. Bold words are words where parts of the word are listed in the top 30 list.

> Applikasjoner med litt mer omfattende state vil typisk bli vanskelig å vedlikeholde, ha unødvendige mye callbacks, state flyttes oppover til felles **foreldrekomponent** og havner ikke der det er logisk å ha den etc. State management løsning som redux og mobx lar deg samle state en plass og gir deg mekanismene for å endre og lese state der du trenger det i **komponenthierarkiet**. Bak redux ligger det en veldefinert best-practise design med bruk av actions og reducers, mens mobx har fokus på enkel syntaks og skjuler mye av det som skjer.
>
> Her forventer vi litt kode som viser at du faktisk har kodet selv eller i det minste lest og forstått løsningen som er brukt i prosjektet.

In this answer, only 9 of the words from the top 30 list are directly mentioned in the model answer. Some words are mentioned using synonym words, either expressed in Norwegian or English. This is the case of the words "state", which in Norwegian is called "tilstand", and "send", "oppdater" and "lagre", which in this case could have replaced the words "flyttes", "endre" and "samle" respectively. Furthermore, the word "komponent" is part of the top 30 list but not a part of the model

answer. However, the words "komponenthierarkiet" and "foreldrekomponent" are both included in the answer. As discovered, wrongly written compound words are a common problem in the student answers.

The question also asks the students to create examples using code. The model answer does not provide an example but states that it is expected. By inspecting the 30 most common words, one sees that several programming syntax words are part of the list. This includes "dispatch", "type", "return", "case", "react", "switch", "const", "this", "prop", "reduce", and "default". These words are all fairly common in the best-graded answers and fairly uncommon in answers with lower grades.

A provided model answer for question 2 in E19 is given below. Directly mentioned words and synonym words from the top 30 list is marked in the same fashion.

> Arrow-funksjoner har ikke egen this (eller binding av), men bruker this fra koden som arrow-funksjonen er skrevet inn i (leksikalsk scoping). Dette er forskjellig fra «vanlige» funksjoner hvor this bestemmes i kjøretid eller bindes eksplisitt i koden.

As for question 13 in E18, only 9 of the top 30 words are present in the model answer. Five of those are the five most common terms in the highest graded answers. There can be several reasons to why so few of the most common words are a part of the suggested solution. The average answer length might be a factor. The average answer length for answers to question 13 is only 40 words. The model answer has only 39 words. Furthermore, though only nine words were found in the model answer, most words in the model answer that would not be filtered away during stopword removal were found amongst the thirty most common words.

From comparing the thirty most common words from questions 1 in E19 and 2 and 13 from E18, it is clear that the common words are representative of words that should be included in a good answer. However, there are many synonym words amongst the most common words. This can be a challenge if measuring the similarity between answers.

Another interesting aspect to look at is how the most common terms relate to the words mentioned in the asked question. Some of the words found in the top 30 lists are a part of the asked question. For each question and the corresponding top 30 list, the following words are found in both the question and the list:

- **E20 Q1**: *tilstand, state, dataflyt, react, applikasjon*
- **E19 Q1**: *responsiv, web, design, teknikk, implementere*
- **E19 Q8**: *global, state, management, hensikt, fordel, react, applikasjon*
- **E18 Q2**: *arrow, funksjon, javascript, this*
- **E18 Q7**: *responsiv web, design, webdesign, teknikk*
- **E18 Q13**: *state, management, redux, mobx*

Some of the identified words are among the most common in both the higher and lower graded answers. "State" and "react" from E20 Q1, "responsiv" and "design" from E19 Q1, "global" and "state" from E19 Q8, "arrow", "funksjon" and "this" in E18 Q2, "responsiv" and "design" from E18 Q7 and "state" and "redux" from E18 Q13 are all in over 70% of the answers graded either high and low.

In some automatic grading systems, researchers have chosen to exclude all terms present in the question [48]. This is because they consider the words to be a poor feature in terms of separating different answers from each other, as they are naturally suspected to be included in most answers. However, removing the terms mentioned in the question can remove valuable information. That the words are not mentioned might indicate that the student has not addressed parts of the question. Thus, by removing such words from student answers, two answers with a low degree of similarity might become more similar if the words are excluded.

However, one cannot automatically conclude that those parts of the question are not addressed just because the terms are not included. Instead of directly answering the question by referring to the mentioned concept, it is often customary to use pronouns to refer back to the question. Other times, the question does not need to be referenced in order for the sensor to understand which parts of the question the student is referring to. Furthermore, when asked to explain a given concept, it is the words describing the concept, not the concept itself, that determines the grade. Thus, when computing the similarity between answers, it might be beneficial in, most cases, to remove terms included in the question from the answers.

Another interesting aspect about this list of question keywords can be found by looking at question 7 in E18. The question includes the words "web", "design" and "webdesign" listed as three separate words. When looking at the frequency of usage for the higher graded answers, "design" is used in 90% of the answers, web is used in 65% of the answers and "webdesign" in 38% of the answers. The words "web" and "design" can in many cases be used separate from each other, e.g., "responsivt design" or "the web". However, there is a possibility that students have chosen to write "webdesign" by splitting the word into two words; "web design". In such a case, "web design" and "webdesign" would not be considered equal when using a bag of words model, though they refer to the same concept. This might also be the case in other words, as the earlier identified "komponenthierarki". Using domain knowledge, such discrepancies can easily be fixed before running any text mining technique. However, it can be much harder to detect and handle in an unknown domain.

A related issue can be found by inspecting the words in questions 8 in E19 and 13 in E18. Both include the two related words "state" and "management". In some cases, these two words might be used together to describe one aspect, namely "state management". However, the word "state" can also be used in different contexts. From figures 5.2c and 5.2f, "state" is included slightly more often then "management". This can be handled by using word n-grams. That would allow the algorithm to consider "state management" as one token and "updating state" as another.

Q4: Is it possible to retrieve the same terms found in the best answers when applying the same technique to all answers?

Accessing the most common words from the best answers are easily done after the grading process is completed. However, when presented with new and ungraded answers, one does not know which answers are considered the best answers. Thus, it would be interesting to see if the most common terms in the higher graded answers remains common when applying the same technique to the entire dataset. Table 5.3 shows the top 30 most common words in all answers to the given questions, sorted in order of frequency. The numbers in the columns *Top 10*, *Top 20* and *Top 30* refers to how many of the first 10, 20 and 30 words in this list are the same as the top 10, 20 and 30 most common words in the highly graded answers.

As one can see from Table 5.3, many of the same words found in Table 5.2 appears when extracting the most common words from all answers. Though the lists do not overlap entirely in their most common words, most words are shared by both lists. For all questions, at least 9 out of the 10 first words are the same, and both share at least 80% of the first 20 words.

As expected, when looking at the 30 most common words for question 2 in E18, there are fewer shared words amongst the most common words in all answers and highly graded answers. This can be a result of two factors. First, as mentioned earlier, the average answer length is considerably shorter for this question than for the other questions. Secondly, the grade distribution for question 2 in E18 stands out from the other questions. While most of the answers to the other questions are dominated by answers that have received 5 or 4, the grade distribution is more uniform for question 2 in E18. However, despite these facts, the lists still overlap by 70% when looking at all 30 words and 85% when comparing the first 20 words.

It is also interesting that the words for question 2 in E18 have the same word listed three times, though written in three different ways. The word "function" is included with three different spellings; "function", "funksjon" and "funskjon". The second word is the same word written in Norwegian, while the latter is a misspelt variant of the second word.

5.3.3 Conclusion

The answer to the question "*Is it possible to identify question-related terminology, and can it be used to distinguish answers of different quality?*" is complex. As mentioned, over half of the most common words, both in the highest graded answers and in all answers, are question-related terminology words. These words would need to be selected manually or by incorporating domain knowledge into the solution. Furthermore, the top 30 words extracted from the best answers were found to be representative for the words in the provided model answers. It was also shown that these words were more common in the best answers than the lower graded answers. The same words could often be found when extracting the most common words form all answers. This indicates that the terms might perform well at distinguishing

Question	Top 30 terms	Top 10	Top 20	Top 30
E20 Q1	komponent, state, react, prop, send, redux, endre, context, mobx, data, funksjon, applikasjon, global, oppdater, måte, dataflyt, tilstand, lagre, stort, usestate, mulig, action, endring, nedover, hooks, kall, reduc, håndter, verdi, management	9/10	18/20	28/30
E19 Q1	responsiv, design, css, grid, skjermstørrelse, viewport, flexbox, størrelse, web, enhet, querie, tilpass, teknikk, skjerm, bilde, nettside, forskjellig, media, layout, ulik, element, endre, webdesign, skalere, dynamisk, mobil, meta, implementer, små, definert	10/10	19/20	30/30
E19 Q8	state, global, komponent, management, applikasjon, react, redux, hensikt, send, fordel, prop, endre, callback, oversikt, funksjon, stort, mobx, slippe, hold, kode, samle, enkel, letter, treng, hierarki, håndter, data, vanskelig, hent, oppdatere	9/10	19/20	28/30
E18 Q2	this, funksjon, arrow, bind, vanlig, slippe, referer, trengs, objekt, function, definer, kall, skriv, javascript, skille, variabel, klasse, kode, deklarer, måte, that, scope, arrowfunksjon, return, komponent, konstruktør, pek, funskjon, const, es	9/10	17/20	21/30
E18 Q7	responsiv, design, css, web, skjermstørrelse, skjerm, nettside, media, querie, enhet, mobil, størrelse, teknikk, tilpasse, element, grid, forskjellig, viewport, endre, webdesign, små, flexbox, ulik, skalere, layout, px, pc, implementer, definer, funger	9/10	19/20	27/30
E18 Q13	state, redux, action, komponent, reduce, applikasjon, mobx, endre, management, dispatch, stort, type, send, react, holde, funksjon, return, lagre, håndtere, data, oppdater, case, måte, endring, payload, prop, tilstand, this, all, switch	9/10	16/20	27/30

Table 5.3: 30 most frequent words from all answers.

between answers of different qualities, which can come in handy during a manual grading process. Section 6.2 suggests a solution where this can be taken advantage.

The experiment also uncovered challenges with using terms to distinguish answers of different qualities by using terms. The most significant identified challenge was the extensive occurrence of synonym words. Most words have several synonym words. In this case, those words can be written in both Norwegian and English. Synonym words cannot automatically be identified. Thus, handling them requires the use of new techniques. Furthermore, as mentioned in Section 4.4, occurrences of both spelling mistakes and variations of compound words were found amongst the most

commonly used words. Optimally, these words should be handled and corrected before applying automatic grading methods. However, such operations can often become a complex process, especially when dealing with the Norwegian language, as there is less research within natural language processing languages like English.

5.4 Experiment 2: Ranking and Retrieval

This experiment aims to rank all answers to a question using a ranking function that calculates the similarity between each answer and a query. The query will be constructed using terms extracted from the answers. The aim is to investigate how well the ranking is able to separate high and lower graded answers. Five questions have been chosen to use in the experiment, questions 1 and 2 from E20, questions 3 and 7 from E19 and question 13 from E18. Section 5.4.1 describes the approach taken to conduct the ranking. Section 5.4.2 presents the results from the ranking given the five questions. The conclusion to the experiment is given in Section 5.4.3.

5.4.1 Approach

To perform the ranking and retrieval experiment, a search engine was constructed using the Python library `Whoosh!`, which allows for indexing and searching through documents, or in this case, student answers. The search is conducted by providing a query to the system and the system ranks the student answers by calculating the similarity between the provided query and the answers. The answers are ranked according to similarity. The more similar the answer is to the query, the higher is the rank it receives. Some answers might receive a similarity score of 0. These answers are still retrieved but receive the lowest rank.

The experiment will test two different approaches for creating the query. The first approach creates a query using common words found in answers graded 5 or 4. The other approach is to construct a query using terms found in all answers. The terms will be combined using the *"OR"*-operator. The ranking will then be evaluated as described by Section 5.1.3. When the query is constructed using the terms from the highest graded answers, the results are expected to be better than in the latter case, as the query most likely will favour the answers graded 5 or 4. As shown in the first experiment, common words extracted from highly graded answers are often less common in lower graded answers. However, it can provide a benchmark for using a query constructed from the entire vocabulary.

Before indexing and searching can be achieved, some choices needed to be made in terms of query construction, creation of the vocabulary, and a ranking function. These are now explained.

Creating the Vocabulary

The vocabulary is an essential component of the collection as it identifies all index terms [7]. After conducting document preprocessing, as described in Section 5.2, one more step was added to the process. As discovered and discussed in Section 5.3.2, terms present in the question are often poor discriminators. For that reason, those words were excluded from the vocabulary. The vocabulary then consisted of the remaining distinct words.

In Section 2.2.2, two text-representation strategies were presented: a simple bag-of-words model consisting of distinctive terms and n-grams, where n consecutive words are grouped together. Both approaches were tested in the experiment, but as using only unigrams showed slightly better results in all cases, only the results from this approach are presented.

Ranking Function

There are several ranking functions used in IR models. One of the most used ranking functions for comparing student answers to a model answer is the *cosine similarity* [43, 49, 57]. Cosine similarity is also the ranking function used in the vector model briefly described in Section 2.3.2. Thus, the cosine similarity was an obvious choice.

As mentioned in Section 2.3.1, cosine similarity measures the similarity between a query and an answer by calculating the cosine of the angle between the two. For this to be achieved, the query and the student answers must first be represented numerically using vectors. This process was handled by the `Whoosh!` library. TF-IDF weights were used to represent each term in the vocabulary.

Selecting the Optimal Query

There are several ways to construct a suitable query. Several queries have been tested in order to find an optimal query. In all cases, the queries were created by selecting terms from the most common terms. That is, the terms used in most student answers. The extracted terms were then combined using an OR-operator. The queries were created by selecting words from two different vocabularies:

Query type 1: From answers graded 5 or 4

Query type 2: From all answers

The answers are first ranked by using terms found in answers graded 5 or 4 and then by using terms found in all answers. The results from these two approaches will later be compared and discussed.

The query length was another parameter to set. Two approaches were discussed:

1. Fixed query length

2. Terms only included in a minimum and maximum number of the answers

An example of the latter would be to extract terms part of minimum 50% of the answers, but no more than 80%. The benefit of this approach would be that it is more prone to account for the average word length of the answers. Selecting a fixed query length of 30 terms made up by the 30 most often included answer words could result in selecting terms that very few answers contain. This was illustrated in 5.2d where the 30th most common word only was included in 9% of the highest graded answers and 4% of the lowest graded answers. For a longer query, the percentage would probably be much lower. However, the second strategy would result in a large variety of query lengths, as how many terms are shared by, for instance, 50-80% of the answers varies greatly for each question. Thus, to ensure consistency throughout the experiment, the first approach was chosen. The selected fixed lengths for the queries that were tested were 20, 30, 40, 50 and 60 terms.

5.4.2 Results

This section shows the results from the experiment for each of the five selected questions. For each question, a table displays the accuracy and gross error within the high grade and lower grades interval, using both the query constructed by terms from the higher graded answer and all answers.

The tables are structured as follows. The left side of the table shows the result after calculating the accuracy and the gross error using a query constructed by terms found in answers graded 5 or 4. The right side shows the result when using a query constructed from terms found in all answers. The accuracy and gross error from both the high grade interval and the lower grades interval are shown. The following abbreviations are used in the tables:

- **Q.L.**: The length of the query
- **H.G. Interval**: High Grade Interval
- **L.G. Interval**: Lower Grades Interval
- **Acc**: Accuracy in the given interval
- **Gross**: Gross Error in the given interval
- **Tot A.**: Total answers that belong inside the interval

When looking at the high grade interval, total answers refer to how many answers received 5 or 4. For the lower grade interval it is how many answers received 0, 1 or 2. The results from the ranking are for each question described below. The best results are marked in bold.

Question 1 E20

	Query: Grades {5,4}				Query: All grades			
	H.G. Interval		L.G. Interval		H.G. Interval		L.G. Interval	
Q.L.	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**
20	0.773	**0.008**	0.556	**0.037**	0.781	**0.008**	0.556	0.037
30	0.79	**0.008**	0.519	0.074	**0.782**	**0.008**	0.519	0.111
40	0.782	**0.008**	**0.63**	**0.037**	**0.782**	**0.008**	**0.63**	0.037
50	**0.798**	**0.008**	0.593	**0.037**	0.773	0.0168	0.593	**0**
60	0.782	0.017	0.556	**0.037**	0.773	0.017	0.519	0.074
Tot A.	119		27		119		27	

Table 5.4: Results for question 1 in E20. Accuracy and Gross Error using a query constructed from answers graded 4 or 5 is shown to the left. Accuracy and Gross Error using a query constructed from all answers is shown to the right.

The results for question 1 in E20 is shown in Table 5.4. The earlier assumption was that the queries constructed by using terms found in answers graded 4 or 5 would yield the best results. However, this only turned out to be true for the accuracy in the high grad interval. The query constructed using terms extracted from all answers yielded equally good or better results in all other cases. Using the query constructed from all answers, the query of length 50 was able to produce zero gross errors in the lower grades interval.

The solution provided overall good results in the high grade interval for both query types, with the highest accuracy for query type 1 being 0.798 and the lowest 0.773. For query type 2, the best accuracy is 0.782 and the lowest 0.773. That means that at its best 95 and 93 out of the 119 answers graded 4 or 5 were inside the high grade interval for query types 1 and 2, respectively. For both query types, the best results showed that only one answer graded either 2, 1 or 0 were placed inside the high grade interval. The rest were answers graded 3.

The accuracy scores within the lower grades interval are considerably lower than in the high grade interval, a reduction of 16,8% and 15,2% for the best results using query types 1 and 2, respectively. However, the gross error remains low in both cases.

The fact that there is not much difference between using a query constructed from the terms found in the best answers and terms found in all answers is beneficial as it enables creating a solution where ranking and retrieval can be performed before even grading a single answer. However, it is worth noticing that since over half of the answers were graded 5 or 4, the terms in query type 2 might still be dominated by terms that, to a large degree, can be found in these answers.

Question 2 E20

	Query: Grades {5,4}				Query: All grades			
	H.G. Interval		L.G. Interval		H.G. Interval		L.G. Interval	
Q.L.	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**
20	0.818	0.014	0.455	0.364	0.811	0.014	0.455	0.364
30	0.804	0.014	0.545	0.273	0.804	**0.007**	0.636	0.182
40	0.825	**0.007**	0.636	0.182	0.797	**0.007**	0.636	**0.091**
50	0.825	**0.007**	**0.727**	**0.091**	0.811	**0.007**	**0.727**	**0.091**
60	**0.832**	**0.007**	**0.727**	**0.091**	**0.818**	**0.007**	0.636	**0.091**
Tot A.	143		11		143		11	

Table 5.5: Results for question 2 in E20. Accuracy and Gross Error using a query constructed from answers graded 4 or 5 is shown to the left. Accuracy and Gross Error using a query constructed from all answers is shown to the right.

The results for question 2 in E20 is shown in Table 5.5. From the table, it is clear that the longer queries give better results, especially for the lower grades interval. In fact, for both the accuracy and the gross error within the low grades intervals, the scores improve drastically as the query length increases to 50 and 60 terms. For the higher grade interval, the best accuracy and gross error score are found using a query length of 60, though using 50 terms produces almost the same results. In terms of the two query types, the results are often equally good, though query type 1 shows a slightly better result for accuracy in the high grade interval.

Compared to the score for question 1 in E20, the results are better or equally good 6 out of 8 times. However, in terms of grade distribution, question 2 in E20 is even more unbalanced than question 1 in favour of higher grades. A total of 143 answers were graded 4 or 5, while 70 students received a lower grade. The best results are given when the query consists of 50 words.

Question 3 E19

	Query: Grades {5,4}				Query: All grades			
	H.G. Interval		L.G. Interval		H.G. Interval		L.G. Interval	
Q.L.	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**
20	**0.703**	0.031	0.721	**0.033**	**0.672**	0.063	0.721	0.049
30	**0.703**	0.031	0.787	**0.033**	0.656	0.078	0.770	0.049
40	0.672	0.062	0.82	0.049	**0.672**	0.078	**0.787**	0.067
50	**0.703**	**0.016**	**0.836**	**0.033**	**0.672**	**0.031**	**0.787**	**0.033**
60	**0.703**	**0.016**	0.77	**0.033**	0.656	**0.031**	**0.787**	**0.033**
Tot A.	64		61		64		61	

Table 5.6: Results for question 3 in E19. Accuracy and Gross Error using a query constructed from answers graded 4 or 5 is shown to the left. Accuracy and Gross Error using a query constructed from all answers is shown to the right.

Table 5.6 shows the results for question 3 in E19. As opposed to the above question, question 3 in E19 is almost completely balanced in terms of high and low grades. A fact that is better reflected by comparing the results from the two query types: query type 1 are able to produce overall better results than query type 2. In addition, the accuracy scores are, in most cases, somewhat lower than for questions 1 and 2 in E20.

Despite the more balanced grade distribution, the gross error remains low for both query types in both intervals. Both query types manage to only incorrectly rank one answer graded 4 or 5 in the lower grades interval. In the high grade interval, only one answer graded 2 or lower is found when using query type 1 and two answers when using query type 2.

Question 7 E19

	Query: Grades {5,4}				**Query: All grades**			
	H.G. Interval		L.G. Interval		H.G. Interval		L.G. Interval	
Q.L.	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**
20	**0.525**	**0.102**	**0.489**	0.156	**0.458**	0.153	**0.533**	0.222
30	0.475	0.119	0.467	**0.111**	**0.458**	**0.119**	0.422	**0.156**
40	0.508	0.119	**0.489**	**0.111**	**0.458**	**0.119**	0.489	**0.156**
50	0.475	**0.102**	0.467	**0.111**	0.407	**0.119**	0.422	0.2
60	0.475	**0.102**	0.467	0.178	0.407	0.153	0.378	0.244
Tot A.	59		45		59		45	

Table 5.7: Results for question 7 in E19. Accuracy and Gross Error using a query constructed from answers graded 4 or 5 is shown to the left. Accuracy and Gross Error using a query constructed from all answers is shown to the right.

As can be seen in Table 5.6, the results for question 7 in E19 are considerably worse than for all other questions. Two factors might have affected the low scores. First, the grade distribution is more balanced than for questions 1 and 2 in E20 and 13 in E18. However, question 3 in E19 had a higher occurrence of lower graded answers but was still able to achieve fairly better results. Another factor that might have contributed is the question type.

Question 7 in E19 was classified by Table 4.10 as an *open-ended* question. When the selection of questions was made, described in Section 5.1.3, an assumption was that due to the nature of the question, the results would be considerably worse than for the other questions. The results might indicate that the assumption can be confirmed, though other explanations cannot be ruled out.

Question 13 E18

	Query: Grades {5,4}				Query: All grades			
	H.G. Interval		L.G. Interval		H.G. Interval		L.G. Interval	
Q.L.	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**	**Acc**	**Gross**
20	0.851	**0**	0.654	**0**	0.822	0.02	0.615	0.115
30	**0.861**	0.01	0.654	**0**	0.851	**0.01**	**0.692**	**0**
40	0.842	0.02	0.692	**0**	0.842	0.03	**0.692**	**0**
50	0.851	0.02	**0.731**	**0**	**0.861**	0.02	**0.692**	0.038
60	0.851	0.01	**0.731**	**0**	0.851	0.02	**0.692**	0.038
Tot A.	101		26		101		26	

Table 5.8: Results for question 13 in E18. Accuracy and Gross Error using a query constructed from answers graded 4 or 5 is shown to the left. Accuracy and Gross Error using a query constructed from all answers is shown to the right.

The results for question 13 in E18, presented in Table 5.8, shows the experiments overall best results for accuracy in the high grade interval, independently of what query type is used. In addition, both query types are able to provide a gross error of 0% in the lower grades interval. Query type 1 also produces zero gross errors in the high grade interval.

As in the case of questions 1 and 2 in E20, most answers to the question have received a high grade. Thus, the terms in query type 2 might still be dominated by terms used in these answers.

5.4.3 Conclusion

To sum up the results and conclude, the results using query type 1 will first be examined before looking at query type 2. In the end, the overall results are presented.

Query type 1

When using terms extracted from the highest graded answers, the best results showed that 4 out of 5 questions were able to rank above 70% of the highest graded answers in the high grade interval. Two questions also achieved an accuracy above 80%. The best queries managed to produce a gross error below 1,6 % for the same questions. That means, only one or no answers graded 0, 1 or 2 were ranked as part of the high grades interval.

The accuracy for the lower grades interval was overall somewhat lower. Though the results for question 3 in E19 showed higher accuracy (83.6%) for the lower grades interval compared to the higher grades interval, all other questions had a reduced accuracy. However, the gross error remained low. In several cases, the query that was able to perform the best accuracy in the interval was also the query that produced the lowest gross error in the same interval, though there were some exceptions.

The open-ended question did, as expected, perform considerably worse than the other question. The best-achieved accuracy was only 52,5% in the high grade interval and below 50% in the lower grades interval. At its best, however, only six answers graded 0, 1 or 2 were placed in the high grades interval, and six answers graded 4 or above were ranked as part of the lower grades interval using query type 1.

Query type 2

Using query type 2, the results were often slightly worse than when using query type 1. However, the queries actually produced equally good or better results in a few cases. In terms of accuracy, 3 out of the 5 questions accurately ranked over 70% of answers graded 4 or 5 inside the high grade interval, two of these had an accuracy above 80%. Question 3 in E19 had an accuracy of 67,2% in the high grades interval. For the same questions, only one or two answers graded 2 or below were ranked inside the high grade interval. The accuracy in the lower grades interval were somewhat lower. Two questions had an accuracy of over 70%. The best accuracy for question 13 in E18 was 69,2%, while for question 1 in E20, the best-achieved accuracy was 63%. As for the open-ended question, the results were considerably worse.

Regarding gross errors in both the high and lower grades interval, questions 1 and 2 from E20, question 3 from E19 and question 13 from E18, no more than a maximum of two answers were incorrectly ranked inside the wrong interval. For the open-ended question, seven answers graded 4 or higher were ranked as part of the lower grades interval. In the higher grades interval, there were also seven identified answers ranked 2 or lower.

Overall performance

Overall, the ranking and retrieval managed to show that separating high and low quality answers is possible. Though the accuracy in the two intervals varied, accuracy up to 86,1% was achieved at best in the high grades interval using either query type 1 or 2 and 83,6% and 78,7% in the lower grades interval using query type 1 and 2 respectively. In all cases, the gross error in the two intervals remained low.

Which query length was the most optimal varied depending on the question and the evaluation metric. Some questions had their accuracy increased by over 10% from one query length to another. The most significant individual difference was found for question 2 in E20. Increasing the query length from 20 to 50 words increased the accuracy in the lower grades interval by 27,2% and lowered the gross error by 27,3%. This makes it clear that a question-specific model could be more suited for the solution to perform sufficiently enough.

5.5 Experiment 3: Clustering similar answer

The final experiment tests a commonly used unsupervised clustering algorithm identified through the literature study, k-means. The goal is to cluster together similar answers to see if the results can yield value during a grading process. The algorithm determines the clusters based on the distance between each individual answer. As mentioned, three questions were chosen to examine in-depth, question 1 in E20, question 5 in E19 and question 13 in E18. Section 5.5.1 explains the set-up and approach in order to calculate the clusters for the three questions. Section 5.5.2 shows and evaluates the results from the clustering algorithm as described earlier in Section 5.1.3. Section 5.5.3 provides the conclusion to the experiment.

5.5.1 Approach

In order to cluster together similar answers, the Python library `scikit-learn` is used. scikit-learn provides simple and efficient tools for predictive data analysis and can let users represent answers as numerical vectors and perform clustering. The clustering is conducted by first representing the answers in a term-document matrix using some text-representation strategy. The term-document matrix is then transformed into a distance matrix by first calculating the similarity between each answer in the matrix and then using the similarity score to find the distance. A distance matrix is simply a matrix where the rows and columns correspond to a document; the cells represent the distance between two documents.

During the experiments, Latent Semantic Analysis (LSA) was experimented with to create a more dense matrix, as it has previously been shown to provide good results within the field of automatic grading. However, after clustering, the results showed that the majority of answers were grouped together in one cluster; the rest mostly ended up in clusters by themselves. Better results were found when not using LSA. Thus, only the above-described approach for clustering will be presented.

Before applying k-means, some choices needed to be made regarding answer representation and algorithmic parameters.

Representing the Answers

In order to represent each answer, n-grams of length 1 (unigrams), 2 (bigrams) and 3 (trigrams) were used. However, the use of n-grams leads to very sparse matrices, so the vocabulary was reduced to keep a more dense matrix. Suzen et al. [48] suggested a solution where words that appeared in less than 90% of the corpus were removed. This strategy was incorporated in the clustering procedure. N-grams appearing in 10% or less of the answers were filtered out from each answer.

The construction and removal of n-grams were conducted using scikit-learn's `TfidfVectorizer`. The function constructed the n-grams, removed those appearing in 10% or less of the answers, and represented the remaining n-grams in a matrix of

weighted features. The function is flexible and allows the user to choose to use either TF-IDF, TF-weights or simply binary number expressing that a feature is either present or absent. The TF-weights can be sublinear or not. All four approaches were experimented with, but TF-weights using sublinear scaling showed the most promising results.

The resulting term-document matrix were then used to create the distance matrix. The weights were first used to calculate the cosine similarity between the answers. The distance is simply $1 - sim(d_i, d_j)$ where sim is the cosine similarity and d_i and d_j are two documents. The formula for the cosine similarity is expressed in Equation 2.5.

Selecting k Clusters

Before running the k-means algorithm, some user-specified parameters must first be provided. These include the number of times the k-means algorithm will be run with different centroid seeds, the number of iterations for a single run and the tolerance for declaring convergence. However, the most critical parameter to specify is the number of clusters to use.

In the choice of k cluster, the most important aspect is choosing a number that creates clusters of value during a grading process. Several cluster sizes were tested. It was seen that when choosing very low cluster sizes it resulted in clusters with no separation of grades. Very high cluster sizes resulted in several clusters with only one answer. In the end, a cluster size of 40 was chosen. The resulting clusters are shown in Section 5.5.2.

5.5.2 Results and Analysis

Figures 5.3, 5.4 and 5.5 shows how answers of different grades were clustered for the three chosen questions. For each graph, the number of answers in the cluster is shown on the y-axis, and the bottom labels represent the cluster number. The colours in the graph represent different grades. Thus, a graph with dark green and a light green colour only contains answers that are graded 5 or 4. The proportion of the column marked in a specific colour represents the number of answers with that grade inside the cluster.

For each created cluster, a purity score is calculated as well as the purity for all clusters to a question using equations 2.10 and 2.11. These scores are presented in Table 5.9. As can be seen from the table, seven, six and twelve out of the 40 clusters for questions 1, 5 and 13, respectively, have received a purity score of 1, meaning that the clusters only consists of answers with the same grade. Question 13 has the highest purity score of 0.70. The scores for the other questions are only slightly lower: question 1 and question 5 has a purity score of 0.62 and 0.65, respectively. There are five clusters in question 13, and four clusters in questions 1 and 5 with a purity score above 0.7 (and below 1). However, most clusters have a purity score below 0.7, and several are also lower than 0.5. Overall, few clusters have received a

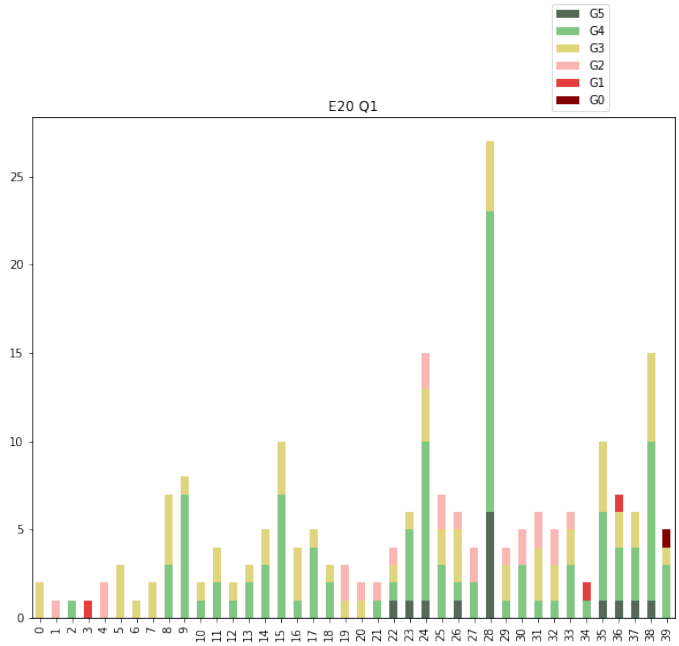


Figure 5.3: Question 1 E20: The x-axis shows number of student answer in each cluster. The bottom labels represent the cluster number. The colors represent different grades as shown by the legend.

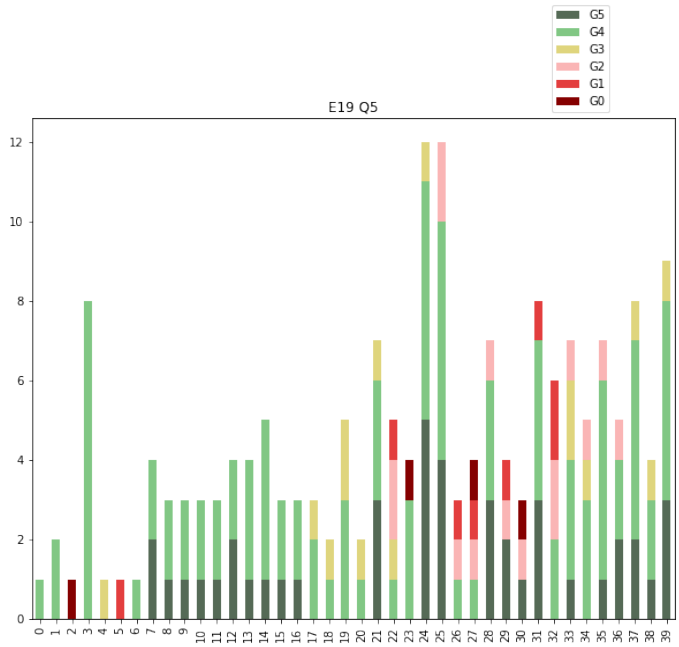


Figure 5.4: Question 5 E19: The x-axis shows number of student answer in each cluster. The bottom labels represent the cluster number. The colors represent different grades as shown by the legend.

high purity score, meaning that the clustering algorithm performs poorly in creating clusters containing equally graded answers.

The purity score only tells how well a clustering algorithm constructs clusters containing data elements of the same class. To get a better understanding of how the

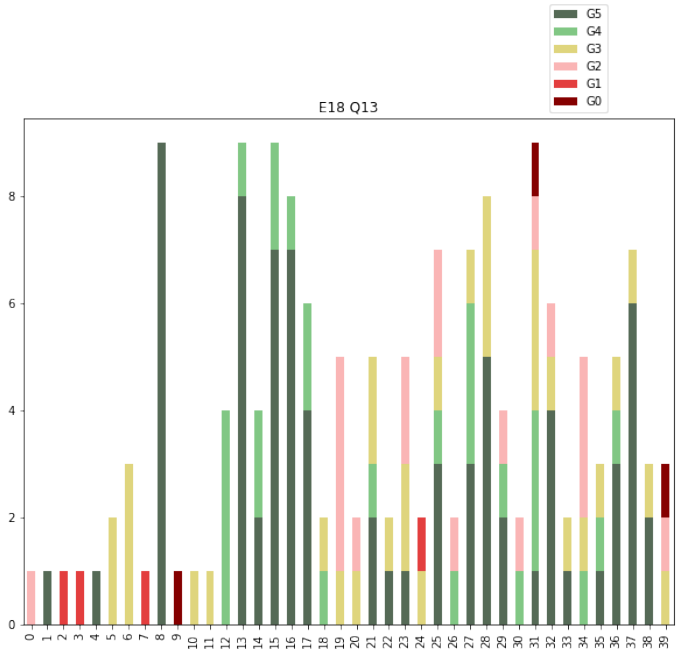


Figure 5.5: Question 13 E18: The x-axis shows number of student answer in each cluster. The bottom labels represent the cluster number. The colors represent different grades as shown by the legend.

algorithm performed, the results must also be compered to the information found in the graphs. Out of the clusters that received a purity score of 1, several only consist of one answer. Question 13, which had the highest number of clusters with a purity score of 1 and the highest overall purity score, was the question with the most clusters consisting of only one answer. Furthermore, out of the 13 clusters with a purity score above 0.7 (and below 1), all contained answers from only two grades. Nine out of those are clusters where the two grades that are represented are consecutive grades in the grading scale, i.e., most 5s and a few 4s or most 4s and a few 3s. This is interesting, as it might indicate an "error" in the grading. That is, the clustering algorithm may have managed to cluster together answers that "deserved" the same grade, though the human grader evaluated them differently.

There are other ways to analyse the results without relying on a purity score. In fact, by simply viewing the graphs, some interesting patterns can be detected. First, the clustering algorithm for both question 5 in E19 and question 13 in E18 has identified several clusters containing only answers with high grades. That is, clusters with answers only graded 4 or above. Question 1, which has very few answers graded 5, has several clusters of answers graded 3 and 4. In all three clusters, there are also several clusters with answers graded 3 or 4 or answers graded 5, 4 or 3. Secondly, though some answers graded low has ended up in a cluster by themselves, in most cases, these answers are placed in clusters containing answers from all grades.

This might indicate that the higher graded answers contain words and a language that separates them from questions of lower grades. Furthermore, lower graded answers need not necessarily be similar for them to be graded equally. Students receiving a low grade often might be partially correct concerning some aspects of

Cluster	Purety		
	Q1 E20	*Q5 E19*	*Q13 E18*
Overall purity	**0.65**	**0.62**	**0.70**
0	1.00	1.00	1.00
1	1.00	1.00	1.00
2	1.00	1.00	1.00
3	1.00	1.00	1.00
4	1.00	1.00	1.00
5	1.00	1.00	1.00
6	1.00	1.00	1.00
7	1.00	0.50	1.00
8	0.57	0.67	1.00
9	0.88	0.67	1.00
10	0.50	0.67	1.00
11	0.50	0.67	1.00
12	0.50	0.50	1.00
13	0.67	0.75	0.89
14	0.60	0.80	0.50
15	0.70	0.67	0.78
16	0.75	0.67	0.88
17	0.80	0.67	0.67
18	0.67	0.50	0.50
19	0.67	0.60	0.80
20	0.50	0.50	0.50
21	0.50	0.43	0.40
22	0.25	0.40	0.50
23	0.67	0.75	0.40
24	0.60	0.50	0.50
25	0.43	0.50	0.43
26	0.50	0.33	0.50
27	0.50	0.25	0.43
28	0.63	0.43	0.62
29	0.50	0.50	0.50
30	0.60	0.33	0.50
31	0.50	0.50	0.33
32	0.40	0.33	0.67
33	0.50	0.43	0.50
34	0.50	0.60	0.60
35	0.50	0.71	0.33
36	0.43	0.40	0.60
37	0.50	0.62	0.86
38	0.60	0.50	0.67
39	0.60	0.56	0.33

Table 5.9: Purity calculated for each individual cluster for the three question. The purity for the entire clustering algorithm is shown in the top row.

the questions, but off on other parts. These parts need not necessarily always be the same for all students. Thus, lower graded answers could be further apart from each other and instead be more similar to the higher graded answers.

scikit-learn makes it possible to extract the most defining features for each cluster. In this case, that means the most important n-grams. To further investigate the clusters, the most important n-grams from a few chosen clusters were retrieved and

examined. For question 13, clusters 8, 31 and 39 are chosen. Cluster 8 is amongst the largest clusters and only contains answers graded 5. Cluster 31 has equally many student answers inside, but these have received almost all types of grades. Cluster 39 contains answers graded 3, 2 and 0. For questions 1 and 5, two clusters each are chosen. Cluster 7 from question 5 and cluster 28 from question 1 are both dominated with high grades. Cluster 22 from question 5 have answers where the grades stretch from 4 to 0, and cluster 19 from question 1 has only answers graded 2 or 3. The most important n-grams for these clusters are shown in tables 5.10, 5.11 and 5.12. Each new n-gram is separated with a comma. The terms in the n-grams are replaced with a word in the stemmed-lemmatized dictionary to allow for more readable terms.

C	30 most defining features
19	data, verdi, lagre, oppdater, endre, måte, endring, nytt, bygd, altså, setstate, context, komponent komponent, mulig, føre, knapp, vis, forskjelen, tilgjenglig, lag, hooks, funksjonalitet, useeffect, app, kjør, funksjon, komponenttre, spesifikt, ofte, global
28	mobx, context, stort, oppdater, action, endre, funksjon, data, måte, global, hooks, håndter, reduc, usestate, enkle, endring, kode, lagre, små, trenge, mulig, redux mobx, sette, funksjonell, dersom, funksjonell komponent, bibliotek, tilgang, enkelt, setstate

Table 5.10: Question 1 E20: 30 most defining features (n-grams) for clusters 19 and 28.

C	30 most defining features
7	scope, block, funksjon, global, endre, block scope, funksjon scope, variablen, global scope, verdi, tilgjenglig, derimot, dersom, bruk, let, endre verdi, reassigne, function scope, function, bety, scoped, blokk, blokk scope, definert, konstant endre, scope funksjon scope, konstant, scope funksjon, mulig, immutable
22	endre, bruk, kode, global, funksjon, altså, definert, scope, function, bety, block, block scope, blokk, blokk scope, tilgjenglig, derimot, dersom, scoped, endre verdi, function scope, reassigne, scope funksjon scope, funksjon scope, variablen, global scope, immutable, innenfor, scope funksjon, konstant, konstant endre

Table 5.11: Question 5 E19: 30 most defining features (n-grams) for clusters 7 and 22.

Comparing the n-grams in the three clusters for question 13 shows that nearly all n-grams found for cluster 8 are terminology words. The proportion of terminology words is reduced in cluster 31 and again for cluster 39, which is mostly dominated by generic words. That means that the level of terminology words drop as the clusters are more and more consumed by lower graded answers. The same pattern is found in the two chosen clusters for questions 1 and 5, though the distinction is not as prominent.

Another noteworthy aspect is that most words part of n-grams of length 2 or 3 are also present as unigrams. This might indicate that n-grams of length 2 and 3 either are redundant or that unigrams should have been excluded. This is particularly

C	30 most defining features
8	action, reduce, import, const, prop, from, js, komponent, return, type, default, data, component, export, app, dispatch, mapstatetoprops, this, this prop, cas, stort, switch, react, class, applikasjon, export default, connect, håndtere, createstore, all
31	action, reduce, dispatch, applikasjon, endring, stort, komponent, dispatch action, data, endre, payload, definer, from, all, lagre, tilstand, baser, import, action type, type, kall, add, samle, holde, gjerne, tar, altså, informasjon, løse, send
39	holde, variabel, endring, styr, holde styr, måte, observable, verdi, mulig, forskjellig, vanskelig, letter, dele, skje, applikasjon, fort, endre, trengs, app, enkelt, gjøres, stort, gjerne, data, action action, extend, export default, action dispatch, action type, function

Table 5.12: Question 13 E18: 30 most defining features (n-grams) for clusters 8 31 and 39.

notable when looking at the most important n-grams in cluster 22 for question 5. The unigrams "scope", "function", "funksjon", "block" and "blokk" are all used to form various n-grams. In addition, this cluster particularly shows the challenge of synonyms and bi-lingual answers.

5.5.3 Conclusion

To conclude, the k-means algorithm cannot be relied on to successfully separate answers based on their (pre-given) grades. Though some clusters seemed to contain only answers given the same grade, most consisted of various grades. However, the algorithm showed better results in clustering together answers with only high grades. The lower grades are usually separated and grouped together with all other grades. However, by inspecting the thirty most defining features, or n-grams, the results indicate that clusters containing high grades also contained more terminology than clusters containing a mixture of grades, which again contained more terminology than clusters with low grades. As has been earlier noted in Chapter 4 and Section 5.3.2, the presence of synonym words and bilingual answers seem to be a consistent challenge.

Chapter 6

Discussion

The first part of this chapter discusses different aspects related to the datasets and the experiments presented in the earlier chapter. The second part presents and discusses use cases and solutions where the experiments might be applicable to aid in a grading process. In the last section, the research questions are revisited.

6.1 Discussion

During this thesis, a literature review was carried out, and relevant theory was presented. A thorough analysis of the datasets containing questions and answers from three exams were conducted. During the analysis, the data cleaning process and encountered challenges in this phase were described. Some features, including the grade distribution and answer length, were also discussed. In the end, challenges regarding the task at hand were presented. After the dataset analysis, three experiments were executed. They aimed to investigate if text mining and information retrieval techniques could be used to assist in an efficient and fair grading process. Specifically, the first experiment examined if it was possible to identify question-related terminology and whether it could be used to distinguish answers of different quality. The second experiment tested a ranking and retrieval method for separating high and low graded answers based on common terms. The last experiment was a response-based approach aiming to see how answers of equal grades were clustered. This section will discuss different aspects related to the datasets and the experiments.

6.1.1 Datasets and Text-Processing

The datasets differed in several aspects from those found to be used in earlier work. The biggest difference was that the student answers were primarily written in Norwegian. No earlier work has been done within automatic grading of Norwegian exams. Furthermore, most student answers contained a mixture of Norwegian and English terms. No identified work have handled dealing with multiple languages in

the same answer. Thus, in terms of language, there is no benchmark to compare the work against. This thesis chose to only work with answers primarily written in Norwegian Bokmål, though answers written in both Norwegian Nynorsk and English were present. As the experiments tested in this thesis rely on answer terms, it also means that they would work poorly if answers written in other languages were present. In a real-world grading process, such answers would need to be filtered out before applying the techniques used in this thesis. Section 4.2 proposed a solution for how this could be solved. However, solutions able to handle these answers as well would be preferable.

Another identified difference between this dataset and most research conducted was the answer length. Few identified researchers have experimented on answers of this length. The exception being the two corpus-based approaches, Apex [41] and Atenea [42] described in Section 3.2.3. Both were reference-based approaches where the student answers were compared to some reference answer or material. However, the lack of reference material for the student answers used in this thesis called for other approaches. Time could have been spent on creating model answers, but this was not prioritised. For the future, comparing student answers to one or several model answers can be interesting to investigate further.

Neither of the experiments performed any additional natural language processing but those described in Section 5.2. This thesis was interested in discovering the potential for using text mining and information retrieval techniques within the domain of automatic grading for Norwegian answers in the field of Computer Science. During the data exploration, and later during the experiments, several challenges were detected regarding the large degree of synonym words, compound words, spelling errors and a mixture of words from different languages. However, since the goal did not involve creating an optimal model for prediction or guidance, it was not a priority to use additional time on correcting the many problems discovered related to either of these challenges. Furthermore, though the dataset consisted of a mixture of natural language and programming code, all text was handled as natural language. It is possible that if the programming code were either excluded, evaluated separately or processed in other ways, this could have led to better and more optimal solutions. Regardless, as handling programming code was outside the scope and identifying all code would be a time-consuming job, code syntax was dealt with as treating it as any other words in the texts.

6.1.2 Results and Evaluation

Overall, the results showed that separating answers of different grades is a challenging task not easily conducted. Nevertheless, it was observed during the experiments that techniques from the domains of information retrieval and text mining could yield value in terms of separating high and low-quality answers. The terminology analysis in the first experiment showed that higher-graded answers tended to include more terminology words than lower graded answers. The ranking experiment was able to, in most cases, rank the majority of the highly graded answers in the top while at the same time excluding most of the lower graded answers from the highest

ranks. Furthermore, though the clustering approach was overall unsuccessful in separating answers of different grades, several clusters did consist of only highly graded answers. A significant drawback to all these methods is that the questions were often dominated by answers that received high grades from the sensor.

That grouping together answers of equal grades is a challenging task was specifically prominent in the clustering experiment. Out of the identified clusters that did not consist of a single item, few contained only equally graded answers. Some researchers have been more successful in using a clustering approach. Suzen et al. [48] used 29 student answers and grouped them into three clusters called excellent, mixed and weak using the k-means algorithm. The excellent and weak clusters contained almost entirely of answers with the highest and lowest grades, respectively. The mixed clusters contained answers with all types of grades. The experiment used a reference-based clustering approach. Basu et al. [50] used a response-based approach, though their dataset consisted of data elements containing various features describing similarity elements between pre-determined equal and unequal answers. The clusters produced for the three questions used in this thesis showed that there were some clusters that contained all high grades or grades from 3 and above. However, the lower graded answers were rarely grouped together but instead placed within clusters consisting of answers with all kinds of grades. There might be several reasons why other researchers have been more successful in clustering similar answers. The answer length, the presence of synonyms, spelling errors, compound words, text-representation strategy, choice of term weights, human grading errors or the general approach are some factors that might have contributed, though further research might be needed in order to pinpoint the exact cause. Nevertheless, that so few answers of lower grades were able to be clustered together can, as mentioned in Section 5.5.2, indicate that lower graded answers have less in common with each other than higher-graded answers in terms of language.

The two first experiments indicated that both terms and ranking could be used to separate high and low graded answers. A drawback for both these experiments and the clustering experiment was that they were only performed for a subset of all available questions. Out of the 26 questions provided, only ten were used in the experiments. Thus, the results only speak to how well the approaches work for these specific questions. Other questions might have produced different results that could have been either significantly better or worse. Nonetheless, questions with all properties (as classified in Table 4.10), from all exams, with different answer lengths and grade distributions were chosen and can thus be seen as representative for the remaining questions in the datasets. As such, they give an indication of how the experiments perform for other questions as well.

The evaluation metrics used in experiments two and three is another aspect that needs to be discussed. Ranking and retrieval are usually evaluated using precision and recall. The metrics captures how well a system performs when selecting relevant documents for a query. However, IR-systems usually only retrieve a subset of all documents in the collection, where each of them is evaluated to be either relevant or irrelevant for a query. After the student answers were compared to a query, all answers were retrieved and ranked to investigate if the ranking could mirror the grades. This called for another method of evaluation more fitted to the problem.

The solution became to create two new evaluation metrics based on accuracy and error. Accuracy and error have some drawbacks. They can become unreliable when the number of documents in a category is much smaller relative to the number of documents in the document collection, as was the case of some questions used in the experiment. Specifically, there were a larger number of high grades compared to lower grades. For instance, different questions produced different gross errors, but in reality, the number of answers incorrectly ranked were the same. Furthermore, the evaluation metrics is as of now unable to summarise what query length provides the overall best results. The evaluation metrics in the clustering experiment suffered from its own drawback. Some clusters had a purity score of 1, which contributed to an increased overall purity score. However, some of these clusters consisted of only a single answer.

It is worth noticing that the evaluation metrics used in both experiments only speak to the relative agreement with the human grader. As was discussed in Section 1.1, there can exist discrepancies between ratings of similar responses. Humans might experience emotional factors during a grading process that can have consequences for answer grades. Furthermore, each year, several students complain about their grades. Many experiences that their grade, as a consequence, is altered. Both the ranking and retrieval experiment and the clustering experiment might have discovered patterns that a human sensor have neglected, which again can have affected the scores of the experiments.

6.1.3 Improving the Experiments

As identified earlier in the thesis, the presences of synonym words remain a challenge, as thus the bilingual answers, spelling errors and compound words. Time was not prioritised to fix these challenges. Nonetheless, being able to handle them might yield more trustworthy results. For instance, the ranking experiment aimed at ranking answers based on the answers similarity to a query constructed from the most common terms. For some questions, the results were promising, as they showed relatively high accuracy and low gross errors. An important question to ask is: why did some of the highly graded answers receive a rank significantly lower than others? There might be several explanations to the question. A possible solution is that the experiments detected errors in the grading. However, as for now, one cannot rule out that it might be because these answers have used other synonym words, English words where other answers have used Norwegian (or opposite), that they have a more considerable degree of spelling errors, or have written the compound words in other ways than the majority of other highly graded answers. Being able to solve these problems would result in a more reliable system. Thus investigating why some answers graded highly received a low rank would become an easier task. These challenges were also present in the first experiment. When comparing the retrieved words to suggested solutions (model answer), a larger degree of overlap would have been possible if using a thesaurus or correcting misspelt words.

As was discovered in Section 4.3.2, the answer length is correlated with the answer grades. Both Alfonseca et al. [42] and Lemaire et al. [41] suggested incorporating a

penalty for answers that had an answer length shorter than some threshold. It might be worth investigating if including such an element in the ranking and retrieval can provide better results. However, as was shown in tables 4.5 and 4.6, shorter answer does in some cases achieve high grades. However, by inspecting the tables in Appendix C and Appendix B more closely, one can see that the questions where highly graded answers had a very low minimum word count, the average word count for the question was also considerably low. Thus, this solution might yield the best results for those questions with an average answer length above some threshold.

During the clustering experiment, there were generally many parameters that needed to be decided. As discussed, other researchers have shown to be more successful in separating answers of equal grades. There might be several reasons why the k-means algorithm performed as it did. However, in general, there were many choices to be made in terms of text-representation strategy, term weighting, and vocabulary size. There might be other approaches that yield better solutions, and thus more time should be spent on examining different options. When inspecting the most important features, the terms used in n-grams of length 2 and 3 were usually present as unigrams, which might have made them redundant. It is worth noticing that different questions might respond well to different approaches. Being able to determine why is a step in the right direction.

6.2 Use-Cases

Aiding the sensor through the process of grading answers to exam questions have the possibility to both reduce time spent on grading and ensure that the grades are given in a matter that is thought of as fair by the students. An important question to ask is if and *how* the experiments can be used during a grading process. By reviewing the literature, two use-cases were identified as to what computers might do in order to achieve a more efficient grading process:

1. Automatically predicting grades
2. Providing guidance during the grading process

I would like to add a third use-case that computers might be beneficial for: detecting answers that might have received the wrong grade, or detecting *anomalies*. Detecting anomalies or errors after the grading process has been conducted is closely related to "providing guidance during the grading process". However, earlier work has only focused on providing means for efficient grading and has not been focused on evaluating the fairness of those grades afterwards. Being able to aid the professor by examining answers that he or she might have graded incorrectly could lead to a fairer grading process, which would be beneficial for the students. In terms of the three use-cases, how can the experiments and knowledge gathered during this thesis contribute?

For the first use-case, grade prediction, none of the experiments work fairly well. The most promising experiment is that of ranking and retrieval, which aims to rank

answers according to their grades. The challenge of using ranking and retrieval for grade prediction is that the grade distribution is, naturally, unknown before grading. Thus, where to make the cut and say, all first k answers should be graded 5, then the next i answers should be graded 4 and so on, cannot be determined even if the system can compute a perfect rank. It might be possible to do a more thorough analysis of the calculated similarity scores in order to set a threshold for the different grade scores.

However, there are ways that the experiments can be helpful for both guidance and anomaly detection. This section will discuss how different off-the-shelf techniques examined in this thesis can be applied to assist a more fair and efficient grading process. The benefit of these techniques is that they can be easily implemented and used. Different approaches for providing guidance is presented in Section 6.2.1. Techniques that can be used for anomaly detection are discussed in Section 6.2.2.

6.2.1 Guidance

This section presents and discusses how the techniques used in this thesis can be utilised in schemes that aid the sensor by providing guidance during the grading process.

Automatically Highlighting Important Terms

Many questions are formulated so that the students will be graded on whether or not they have been able to identify the most important question-related terminology. Thus, when grading a student answer, the sensor is concerned with whether or not these terms are included in the answer. The first experiment detected that terminology words are often more profound in higher graded answers than in lower graded answers. The experiment also showed that the most common terms in the highly graded answers also were common terms in the entire set. Furthermore, these words were often considered important to provide a good solution. This can be utilised in a scheme where important words in the student answers are highlighted before they are presented to the sensor. In that way, the sensor can easily see if essential terms and concepts are mentioned in an answer. In order to achieve this goal, the following steps are necessary:

1. Dataset creation
2. Dataset cleaning
3. Text-processing and creation of stemmed-lemmatized dictionary
4. Indexing all answers to a question
5. Retrieving n most common terms
6. Presenting n most common terms to sensor

7. Sensor picks k most promising terms (and can choose to add new terms)
8. Chosen terms are highlighted in the question answers by comparing the terms to the selected terms and the stemmed-lemmatized dictionary.
9. The highlighted answers are presented to the sensor

These steps will now be explained more in-depth before the benefits and challenges of using the solution are discussed.

After the students have completed their exam and the dataset is created (step 1), the answers are cleaned (step 2) as described in Section 4.2. Text preprocessing is then applied to all student answers as described in Section 5.2 (step 3). During the stemming and lemmatization step, the stem (root) is stored as a key in a dictionary, and all variants of the original stem appearing in the student texts are stored together with the stem. There are several reasons why this should be done. First, the Norwegian stemmers are still not optimal, and often a larger part of the suffix is removed than what should have been. By storing all used variants of the stemmed words, the original words are still accessible and can replace a retrieved word to make it more readable for the sensor. This has been done both in Section 5.3 and 5.5 in order to make the retrieved words more readable. Secondly, highlighting the chosen words in the student answers would allow for an easier process if there exists a mapping between the stemmed words and the original words. A dictionary would need to be created for the answers to each question.

Once the dataset has been cleaned and preprocessed, an inverted index must be constructed in order to extract the most common terms (step 4). As the most common terms are extracted from the answers to each question, an inverted index is created for each question in the dataset. How many words to extract can easily be decided by either the sensor or the system (step 5). After the most common words have been extracted, the words can be presented to the sensor in a list (step 6), allowing the sensor the choose the words of interest (step 7). This solution can also be enriched by allowing the sensor to specify own words, not appearing in the system constructed list (step 8). The student answers are then iterated over, and if the chosen words are a part of an answer, they are highlighted. The answers are then presented during the grading process (step 9).

This tool is meant to reduce the burden on sensors and allow for a speedy and fair evaluation. By highlighting the most relevant terminology, the sensor can easily tell if the students have been able to include the relevant course terminology, which could reduce time spent on grading individual answers and make it more evident if the student has been able to answer the question properly.

Furthermore, the tool is easy to implement, it requires a short amount of time to compute the necessary steps, and most can be run as a background job without interference from a human. The same approach can be applied to all answers in the dataset and does not need to be tailored to any particular question to yield the intended results. Because the tools needed to implement the solution is openly available and can be used as off-the-shelf techniques, they can also be easily integrated with existing assessment solutions such as Inspera.

There are, however, some challenges and drawbacks related to the approach. This strategy works well for questions where mentioning the correct terms is of essence. However, not all questions are formulated in such a way. Some questions are more open-ended, allowing for a variety of different answers where multiple answers can be considered as being of high quality though only a small amount of the same words are included. This can be solved by allowing the sensor to choose which questions the technique should be applied to. Furthermore, the solution only focuses on the occurrence of specific words, not the context they are mentioned in.

Another drawback is that the solution would only work for questions written in the same language. Norwegian Nynorsk and English answers would need to be filtered out as these answers can contain the same words as identified during the extraction process but written in different ways. This is also the case for students that have used words synonymously to those chosen or simply written the word wrong. Handling such limitations would include utilising more natural language process. Furthermore, the process must be conducted once for each question. Meaning that the sensor would need to choose important terms for each question. This does consume extra time, but hopefully, the time spent during grading can be reduced overall.

Ranking and Retrieval

Ranking and retrieval can be helpful during the grading process. As briefly discussed during the introduction in Section 1.1, the order the sensor examines and grades answers might affect the individual score answers receives. Though the ranking experiment was not able to provide a perfect division of high and low grades, the solution can still be useful during a grading process. By first examining a proportion of the highest and lowest-ranked answers, the sensor will likely get a good overview of the best and worse answers to a question. This might contribute to a more balanced and accurate grading process. Conducting such a selection of answers would need to be done using words extracted from the entire vocabulary (query type 2), as what is considered the best results is still unknown. It could also be possible to let the sensor choose which words should be part of the query by first extracting the words and then letting the sensor choose the query terms from the identified terms and possibly add new ones.

As was seen when examining the results of the ranking and retrieval experiment in Section 5.4.2, the approach works better for some questions than others. The open-ended question showed considerably worse results than the other questions indicating that the solution might not be optimal for all questions. Further investigation of ranking and retrieval might be necessary to provide even better results for all questions. However, it should be possible to use a solution where the sensor chooses what questions the tool should be used on. This solution can also easily be combined with automatic highlighting of important terms.

Clustering Analysis

Clustering might also be beneficial for guidance. If clustering before the answer grades have been decided, clusters can be displayed together with the most important terms. By inspecting the terms, it might be easier for the sensor to discover related answers with interesting wordings that can be chosen for grading. As was seen when analysing the clusters, clusters with highly graded answers showed to have a high degree of terminology words, while those with almost only low grades contained more generic words.

Other Approaches

Another approach that can increase the efficiency of the grading process is by utilising what has been learned from examining answer lengths. In Section 4.3.2 it was found that the answer length has a positive correlation with the grade. In addition, Section 4.2 found that by filtering answers based on number of words below some threshold, unanswered questions and answers where the students had only answered using some fill-words were easily identified. By extracting all answers with an answer length below some user-defined threshold, many low-quality answers can be easily retrieved, grouped and graded in almost a single click.

Providing answer statistics and additional information about answers to the sensor might also be beneficial during grading. Some core statistics have been identified through analysis and experiments that can easily be retrieved and shown during grading. Furthermore, the techniques used during this thesis can be used to provide additional information to the sensor. The suggestions for relevant answer statistics and information is shown in the list below. Each suggested in then explained in detail.

- Answer word count compared to average word count
- The answers rank after compared to some query
- The grade of the most similar answer already graded
- The answer most similar to this answer that is not already graded

As mentioned, a positive correlation has been found between each answers word count and the belonging grade. That is, as the word count increased, so did the grade. For some questions, like question 4 in E19, the correlation coefficient was as high as 0.66. As the answer length thus seems to be a good indicator of the grade, the attribute can be considered important to include amongst the statistics presented to the sensor.

Another way to utilise document ranking is by simply presenting each answers rank after queried and retrieved. Because the best answers are not identified before after the grading process is conducted, the query words must be taken from the entire vocabulary. However, the experiment showed that even when using the most

common words from the entire vocabulary, the ranks did not differ much in terms of accuracy and gross error than when the query was constructed using terms extracted from answers graded 4 or 5. The rank would not be able to say if the answer deserves a high or low grade but can tell the sensor how close the answers are to the specific query relative to other answers, which again might indicate if the answer is among those of high or low quality.

Similar answers often receive the same grade. This can be incorporated into the system to ensure a more efficient and fair process. Using a similarity measure, calculating the similarity to answers both already graded and not yet graded can easily be achieved by first expressing the answers in the vector space. Thus, the sensor can see which grade was given to the most similar answer, and get a suggestion for which answer that should be graded next. An important factor to consider is that there will be few answers that have been graded in the beginning. Similarly, some answers can be fairly unique, and thus even the most similar answer might still be very different. Thus, similar answers should only be shown when their distance is below some set threshold. Determining a threshold for how close two answers must be for them to be comparable (similar) was also proposed by Klein et al. [43].

To make use of these additional approaches, it is important to communicate both the advantages and drawbacks to the sensor. Though the answer word length has been shown to be an important indicator of the grade, it is no exact science. As was discussed in Section 6.1, there are some very short answers that have received a high grade. There are also very long answers that have received a low grade. Furthermore, a high rank does not necessarily mean that the answer should receive a high grade. In the ranking experiment, it was seen that lower graded answers also received high ranks, though the majority of the answers that were ranked high were answers that also were graded with a high grade. However, including the statistics and additional information is something that has the possibility of being easily included in existing assessment tools. Figure 6.1 shows how this additional information can be view through Inspera's graphical user-interface.

6.2.2 Anomaly Detection

Every year, several students at NTNU send formal complaints regarding their received exam grade. An article written by the Norwegian newspaper VG, discovered by analysing data from the ten biggest universities in Norway that 39% of all student complaints resulted in an altered grade [4]. Thus, a system that can help cross-check the initial grades set by the sensor might benefit both the students and the sensor. Both the ranking experiment and the clustering experiment can be used to detect such answers.

Ranking and retrieval have the potential for being useful for anomaly detection. After each answer has received a grade, words can be extracted from the best answers to create a query. By now, the grade distribution is known, and it can be easier to see which answers graded highly has been ranked outside of the high grade interval or which answers graded with a low grade has been ranked as part of the interval. There are, of course, many reasons why these answers have received a rank deviating

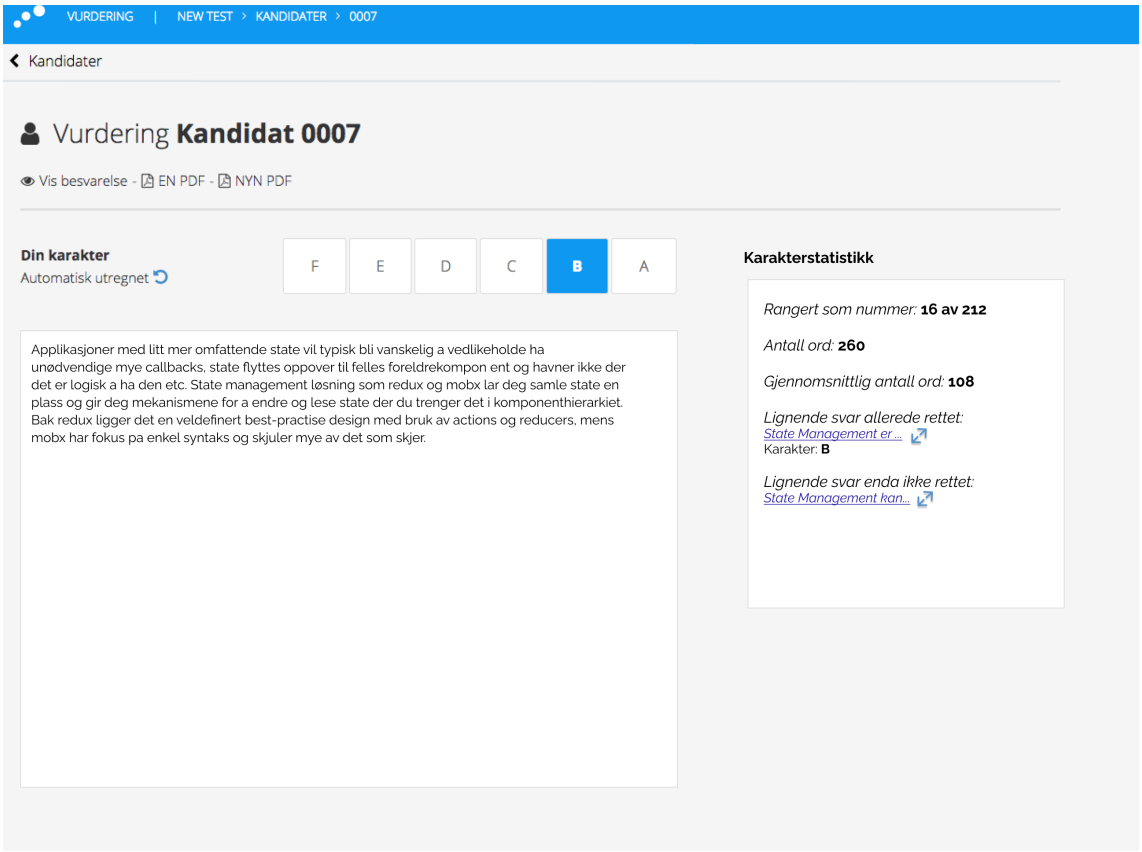


Figure 6.1: A suggestion for how additional information and statistics can be provided to the sensor through Inspera's user interface.

from the grade, but by examining them, it is possible that the sensor discoverers that he or she has either been too harsh or too mild when the grade was first decided. By solving the synonym problem and correcting spelling errors, the solution will become more reliable.

Furthermore, after the grading process is conducted, it is possible to see how answers of different grades have been distributed amongst the different clusters. The sensor might discover some answers part of clusters consisting of answers of very dissimilar grades worth reevaluating. This might open up for identifying wrongly graded answers.

These methods cannot tell the sensor that some answers have been graded wrong. That decision must be taken by the sensor. However, they can suggest some answers that might be worth examining once more. If these solutions are able to detect anomalies is not yet clear, but investigating if they can provide the desired results is worth examining further as they both have the possibility of ensuring a more fair grading process.

6.3 Revisiting the Research Questions

The goal of this thesis is to study the use of information retrieval and text mining techniques to assist the process of evaluating Norwegian exams in the field of Computer Science and if this can be done in a manner that is fair and just for the students. Three research questions were formulated to meet this goal. These are addressed in this section.

Research question 1 What is state-of-the-art within automatic grading of text-based answers?

By conducting a literature study of the work conducted on automatic grading and assessment of text-based answers, it was clear that there has been much work done within the field the past decades. Broadly, two sub-fields were identified: Automatic Short-Answer Grading (ASAG) and Automatic Grading of Essays (AGE). The two fields differ in how they approach the grading process. While AGE systems grade answers based on their writing style, ASAG systems try to evaluate the content of the answers. Often, the systems also differ in terms of average answer lengths. AGE systems usually deal with answers having an average length of two paragraphs to several pages, while ASAG systems deal with answers where the length ranges from one phrase to one paragraph. As the answers dealt with in this thesis were to be evaluated based on content, it became natural to investigate ASAG systems further.

Four dominating approaches were found within automatic grading of text-based answers: concept mapping, information extraction, corpus-based and machine learning approaches. Out of the four approaches, the most recent work has been done within machine learning. In addition, some researchers have focused on developing systems aimed to aid the sensor through the grading process. These approaches were characterised as semi-automatic methods, as they did not aim to provide a specific grade for each answer. In all, two use-cases were detected within the field of automatic grading: automatically predicting grades and using computers to guide the sensor through the grading process.

There are also other ways to distinguish between the different systems. One way is to classify the approaches as either response or reference-based. Reference-based approaches relied on a model answer that the student answers are compared against, often using a feature or features based on the similarity between a student answer and a model answer. Response-based approaches do not rely on a model answer but instead compare student answers with each other. The assumption is that answers graded equally share similar characteristics. There also exists approaches that are a combination of the two.

It is hard to conclude what exactly can be considered state-of-the-art. There have been some attempts to make datasets more open and available and competitions have been arranged where researchers and data scientists compete to create the best solutions. However, in most cases, few methods have been tested on several exam sets within different types of courses and most datasets are not openly available. The focus of automatic grading has been on developing new approaches fitted to

whatever dataset is accessible to the researcher, rather than improving existing work to optimise an overall good solution. As a result, few methods are comparable.

Research question 2 What are the possibilities and challenges for applying text mining and information retrieval techniques to Norwegian text-based exam questions in Computer Science?

There are different ways to approach this question. Simply applying information retrieval and text mining techniques does not pose a big challenge once some data cleaning has been performed. However, for the techniques to yield any value in terms of automatic grading and assessment schemes, several challenges must first be addressed. Challenges within automatic grading and assessment have been discussed thought this thesis, both in terms of general challenges and challenges specific to this dataset. Section 4.4 provides a list of all detected challenges during the dataset analysis. Several challenges detected in this section can also be generalised. Challenges were also detected during the experiments and by reviewing the literature.

The first obvious obstacle is the language. No earlier work within automatic grading of answers based on content has been identified where Norwegian answers are examined. Thus, when approaching this thesis, there existed no overview over linguistic challenges or approaches that have been shown to work better than others. Norwegian is a language that allows for a more flexible word ordering, has richer morphology and more distinctive linguistic characteristics than other languages like English. This can be further complicated by a large degree of spelling errors and wrongly written compound words. Moreover, common terminology found in Computer Science often has poorly translated alternatives. As a result, student answers related to this topic often include English terminology words, making the answers bilingual. However, the students are free to choose whether they want to write the terminology words in English or Norwegian. This thesis showed that the vocabulary often consisted of the same words written in both languages (e.g., "state" and "tilstand").

Another challenge regarding Norwegian answers is that Norway has two principal written forms, Norwegian Bokmål and Norwegian Nynorsk. This thesis chose to exclude answers written in Nynorsk; however, using techniques that can handle both languages would be beneficial. Some researchers have suggested simply using a translator to apply their approach to other languages [42]. However, as discussed, answers from the field of Computer Science are often bilingual as they may contain terminology written in English. A simple translation scheme can potentially provide worse results than using the answers as-is, especially when the student answers also contain linguistic errors.

Computer Science is the study of computers and computing along with theoretical and *practical* applications [76]. This thesis focused on using answer terms in the experiments. However, some exam questions requested the students to provide code to exemplify their answers. Furthermore, questions that did not require students to use any programming code still contained code snippets. The experiments in this thesis still treated all words, including programming syntax, as natural language.

As discussed, this might have introduced some noise in the data that could have affected the approaches. That some programming code is a part of a student answer when the topic is Computer Science is not unlikely, and it might be worth spending time on further investigating how to handle such discrepancies in the answers.

Answer length is another challenge. Most researchers have focused on answers of short lengths. That can make the answers more comparable, as it leads to more restricted word choices and reduces the number of possible synonym words. Thus, when dealing with longer answers, researchers should bear in mind that when using terms as features, the vocabulary can consist of multiple words of the same meaning. More flexible word choices result in increased distance between answers though the semantic meaning can be the same.

Specific for the datasets used in this thesis were that the exam answers were retrieved from Inspera. Inspera was used by the students during the exam. The platform allows students to format their texts, include images and use tables. This created considerable noise in the data that needed to be cleaned before applying information retrieval and text mining techniques. It did not make the process any easier that the exam topic was Computer Science. Several students used the same HTML-tags that Inspera uses for formatting as part of their answers. It is not unlikely that more and more Universities will make use of similar solutions in the future. Thus, it is essential for further work to have a good solution for cleaning the data while at the same time preserving the students' answers.

Despite these challenges, it is still possible to apply text mining and information retrieval techniques to Norwegian text-based exam questions in Computer Science, and that such techniques can yield value in a grading process. During the experiments, techniques such as indexation, querying, computing similarities using cosine similarity, weighing schemes such as term frequency and inverse document frequency, and clustering algorithms were applied to the student answers. The answers were first preprocessed using lexical analysis, stopword removal, stemming and lemmatization. In the future, more natural language processing should be performed to address the earlier mentioned challenges.

Due to the non-existing research on Norwegian answers, it could not be decided with certainty which methods or approaches should be examined. The first experiment was based on knowledge gathered during the literature review and conversations with the course coordinator regarding how exams within Computer Science often are graded. The second experiment utilised the core of Information Retrieval in an effort to rank student answers to see if the grade order was mirrored in the rank. Lastly, the third experiment used one of the main text mining techniques, clustering, to see how the student grades were reflected in the clusters and if it was possible to distinguish clusters containing answers of different grades. The unsupervised approach was used as it does not rely on any grading knowledge, making it more valuable during a grading process if successfully implemented. Other approaches can also be tested, such as predicative text mining to determine answer grades.

Research question 3 Can information retrieval and text mining techniques be employed in order to assist an efficient and fair grading process?

There are two aspects of this question to address. Can information retrieval and text mining techniques be used to assist an *efficient* grading process, and can they be used to assist a *fair* grading process. To answer this research question, three experiments were conducted. The experiments examined how these techniques can be used to distinguish answers based on their quality. The conducted experiments and their results were presented in Chapter 5. Several aspects of these techniques have been considered and discussed earlier in this chapter to determine if the methods can be useful within the field of automatic grading. Furthermore, Section 6.2 proposed applications for how information retrieval and text mining techniques can be used to achieve an efficient and fair grading process, both in terms of providing guidance and anomaly detection.

The experiments showed that the techniques could vary in how successful they were, depending on question properties. In the ranking and retrieval experiment, the open-ended question achieved considerably worse results than the other questions. Thus, there might be a need for more research on how the techniques adapt to different question types. Furthermore, the grade distribution also affects how well different methods perform. It was seen that questions that had a considerably larger amount of highly graded answers often achieved better results than questions that had a more balanced grade distribution. However, use-cases for guidance and anomaly detection based on the experiments were proposed. The guidance systems can work without access to the grade distribution and still provide value with little human interference. This makes them more suited for handling an actual grading process compared to question-specific models building on supervised machine learning. Furthermore, to assist a fair grading process, both clustering and ranking should be examined more in-depth.

Nevertheless, the approaches face several issues when used on student answers. As discussed, exam answers, specifically those concerning the field of Computer Science, are often bilingual. Furthermore, as the answer length increases, more synonyms and spelling errors are found. This makes it harder to compare answers based on included terms. Thus, there are still many challenges and possible improvements that should be addressed. However, overall, the methods' achievements suggest that with further research, fine-tuning, and perhaps stronger natural language processing, it is possible to utilise information retrieval and text mining techniques in an actual grading process of Norwegian exams in Computer Science.

Chapter 7

Conclusion and Future Work

The goal of this thesis was to study the use of information retrieval and text mining techniques to ease the process of evaluating Norwegian exams in Computer Science and to see if this can be done in a manner that is fair and just for the students. This chapter provides a conclusion to the work. Furthermore, the contributions to the field of automatic grading will be summarised and presented. The chapter ends with a description of possible improvements and other ideas related to how automatic grading and assessment of Norwegian exams can be achieved in the future.

7.1 Conclusion

There has been much research on automatic grading and assessment of student answers, dating back to the 1960s [5]. However, most research has been conducted on English answers, and no earlier work has been identified where answers written in Norwegian are evaluated based on content. This thesis can be seen as a first study of how a grading process of Norwegian content-assessed exam answers can be conducted in an efficient and fair manner by utilising techniques within information retrieval and text mining.

In order to meet the goal, three research questions were formulated. These were addressed in Section 6.3. To answer these questions, a literature review of research related to automatic grading was conducted. During the literature study, it was discovered that there had been several attempts to solve the problem of automatic grading. However, as few methods have been tested on several exams, containing questions from different courses and of different lengths, there are few methods that are comparable. In addition, few datasets have been described in detail or made accessible to other researchers. A thorough analysis of the three exam datasets was conducted. The dataset analysis discovered several challenges in terms of applying text mining and information retrieval techniques in automatic grading and assessment schemes. The biggest challenge concerned the language used in the answers. Norwegian answers have thus far not been used in research on automatic grading, where answers are evaluated based on content. Hence, it could not be decided with certainty which methods or approaches would yield the most value to

examine. Furthermore, common terminology found in Computer Science often has poorly translated alternatives. As a result, some students chose to write these words in English, while others used the Norwegian equivalent. The long answer lengths also resulted in a more diverse language, where several synonym words and spelling errors were present. However, it was also detected that the answer length positively correlated with the grade, indicating that longer answers often receive better grades.

Three experiments were performed. These used some of the most known and available techniques within information retrieval and text mining, as the goal was to investigate if techniques from these domains could yield value in a grading process. The results from the experiments showed that both information retrieval and text mining can be useful during grading. Specifically, it was discovered that terminology words included in student answers can be good discriminators for separating answers of high and low quality. As was discussed in Chapter 6, applications of these techniques are especially promising in terms of providing guidance to the sensor and detecting answers that may have received the wrong grade. Nevertheless, more time should be spent on further investigating these techniques in terms of automatic grading, guidance, and anomaly detection for Norwegian exams as it can reduce the burden of the sensor and benefit the students by providing a fair and just grading process.

7.2 Contributions

This thesis contributes to the area of automatic grading of exam answers by encouraging further exploration of information retrieval and text mining techniques in order to assist an efficient and fair grading process. The study is the first that uses Norwegian text-answers, graded based on content. An extensive literature review of related work and research was conducted and is summarised in Chapter 3. The overview can be used as a starting point for future research within the field.

Furthermore, this thesis has contributed with a descriptive approach for how to clean exam datasets retrieved from Inspera. NTNU and many other universities uses the exam platform Inspera Assessment for all deliveries that receives a grade. For further work on analysing exams delivered through the platform, the approach suggested in Section 4.2 can be used as a first step in the process. Chapter 4 also highlights challenges related to the dataset and automatic grading. Though some might be specific to this dataset, many apply to other exam answers, specifically those within the field of Computer Science. There can be several ways to handle such challenges, but they are still important to review when dealing with new exam answers. It has also been detected that longer answers and more terminology often characterise answers that receive good grades.

The overall goal of this thesis was to investigate if techniques within information retrieval and text mining can be used to ease the process of evaluating Norwegian exams within the field of Computer Science in a manner that can be perceived as more fair towards the students and more efficient for a sensor. Though the tested techniques might not perform sufficiently enough to allow for usage today,

the results are promising and indicate that the research should be further extended in the future. Nevertheless, solutions for how the methods can be enabled has been suggested. Section 6.1.3 provides an overview over suggested improvements that might make them more applicable in the future. Both ways to guide the sensor through the grading process and approaches that might aid in detecting answers that has received the wrong grade were provided. As far as the author is aware, there has not been any research on the latter. Still, every year multiple students in Norway complain about the grades they receive on an exam, and many of them end up receiving a new grade [3, 4]. The following section proposes suggestions for further work and methods that might contribute to achieving better results.

7.3 Future Work

Though there have been several studies focusing on aiding the grading process, it is still challenging to achieve satisfactory results. There is also a lack of research in trying to aid the sensor in the grading process instead of replacing him or her with automatic grading systems. This thesis has provided a first study focused on examining if techniques within information retrieval and text mining can provide a fair and efficient grading process of Norwegian exam answers in Computer Science. This section provides suggestions for how the research conducted in this thesis can be further extended and improved. In addition, new ideas for potential research that may be beneficial for automatic grading are presented.

Further testing on other datasets and more questions

The study showed some promising results. However, the methods have only been tested on a subset of the questions from the three datasets. In order to get a better understanding of how well the approaches work and what challenges there are, experiments should be conducted on more answers and answers from other datasets. The approaches have not been tailored to work specifically for Computer Science questions, making it possible to test questions and answers from other fields using the same techniques.

More extensive text-preprocessing

Languages differ in how challenging they are to preprocess. There exist many resources on how to process English written text, and it can be regarded as relatively easy to process compared to other languages. More preprocessing is needed when studying a language with more flexible word ordering, richer morphology, and distinctive linguistic characteristics. There has been far less research within natural language processing using the Norwegian language. This thesis only used lexical analysis, stopword removal, stemming and lemmatization. That means that many challenges have not yet been addressed. A big challenge discovered through the dataset analysis and the experiments is the presence of spelling mistakes, synonym words

and different variations of compound words. It would be beneficial to examine ways to handle these challenges as it might lead to more reliable and better-performing solutions. It can be useful to investigate using a language model such as BERT[1] as it can capture the context of a word and allow for the identification of related words in the vector space. This might aid in building a thesaurus for better handling the synonymy challenge.

Furthermore, as was seen during dataset analysis, many answers contained examples and programming code. This can affect the calculated similarity scores. Even if most parts of two answers are similar, using different code snippets and examples can lead to an increased distance though a human evaluator would consider the answers almost equal. Being able to identify and handle examples and programming code part of natural language answers can provide a more sophisticated solution and should be further investigated.

Handling Multiple Written Languages

Another challenge detected was that the answers in the dataset were written in different languages: Norwegian Bokmål, Norwegian Nynorsk and English. This thesis chose to only look at answers written in Bokmål. For the future, it can be interesting to test how exam answers in all languages can be part of the same solution as both are commonly found amongst exam answers. To achieve better results, more research would have to be conducted on how to handle these challenges. Pre-trained word embeddings are often used to determine that two words are of the same meaning. However, there are no available resources identified by the author where embeddings are trained on both Norwegian Bokmål and Nynorsk that can also identify English loan words, specifically those typical to Computer Science exam answers. If existed; these can be used in order to determine, for instance, that the three words "kilde", "kjelde" and "source" have the same meaning.

Comparing With Model Answers

The approaches chosen in this thesis can be characterised as response-based approaches in that they do not rely on a model answer. This decision was made due to the lack of sufficiently good enough model answers. Most identified research seems to be focused on using a reference-based approach. In the future, more research should be conducted using model answers for grading Norwegian exams if such answers are available. For some questions, where the question opens up for more diverse answers and examples, several model answers should be used. It would then be possible to use similar techniques as presented in the experiments to analyse the exam answers.

[1] https://github.com/NBAiLab/notram

Implementing suggested solutions

Some suggested solutions for how to utilise the findings from this thesis to achieve an efficient and fair grading process were presented in Chapter 6. These solutions have not been tested. It will be interesting to implement and test how these solutions can work in practice and if they can provide the desired results during a grading process. Specifically, solutions for anomaly detection would need to be investigated in a case study together with sensors to see if the solutions indicate anomalies in the grading and if these methods can identify them.

Statistical Approach

This thesis focused on using information retrieval and text mining techniques. The literature study classified the general approaches to automatic grading within four categories. In more recent years, there has been a growing increase in research into approaches based on machine learning. In such a case, statistical features are often extracted from the answers to use in a machine learning model. The features used in these approaches are often based on similarities between the answers and the model answer(s). For further research, investigating statistical features in Norwegian exam answers that might aid in automatically predicting a grade for each answer and utilising them in a machine learning approach can be beneficial. From the dataset analysis, the answer length was shown to be an important feature and can be included in a statistical approach.

Bibliography

[1] Bhuvnesh Chaturvedi and Rohini Basak. 'Automatic Short Answer Grading Using Corpus-Based Semantic Similarity Measurements'. In: *Progress in Advanced Computing and Intelligent Engineering - Proceedings of ICACIE 2019, Volume 2*. Vol. 1199. Advances in Intelligent Systems and Computing. Springer, 2019, pp. 266–281.

[2] Nuno Escudeiro, Paula Escudeiro and Augusto Cruz. 'Semi-Automatic Grading of Students' Answers Written in Free Text'. In: *Electronic Journal of e-Learning* 9 (2011).

[3] Eva Tønnessen and Jonas Hartford Sundquist. *Eksamen: Litt vekst i antall klager*. https://khrono.no/eksamen-eksamensklage-klage/eksamen-litt-vekst-i-antall-klager/253099. Accessed: 2022-01-17.

[4] Maren Olave Ask Hütt. *Så sannsynlig er det å få bedre karakter om du klager*. https://www.vg.no/nyheter/innenriks/i/v5wPRj/saa-sannsynlig-er-det-aa-faa-bedre-karakter-om-du-klager. Accessed: 2022-01-05.

[5] E. B. Page. 'The Imminence of... Grading Essays by Computer'. In: *The Phi Delta Kappan* 47.5 (1966), pp. 238–243.

[6] Steven Burrows, Iryna Gurevych and Benno Stein. 'The Eras and Trends of Automatic Short Answer Grading'. In: *Int. J. Artif. Intell. Educ.* 25.1 (2015), pp. 60–117.

[7] Ricardo Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval - the concepts and technology behind search, Second edition*. Pearson Education Ltd., Harlow, England, 2011. ISBN: 978-0-321-41691-9. URL: http://www.mir2ed.org/.

[8] IBM Butt Education. *Text Mining*. https://www.ibm.com/cloud/learn/text-mining. Accessed: 2021-11-15.

[9] Sholom M. Weiss, Nitin Indurkhya and Tong Zhang. *Fundamentals of Predictive Text Mining*. Vol. 41. Texts in Computer Science. Springer, 2010. ISBN: 978-1-84996-225-4.

[10] IBM. *Natural Language Processing (NLP)*. https://www.ibm.com/cloud/learn/natural-language-processing. Accessed: 2021-09-21.

[11] Bo Bjerke-Lindstrøm. *Teaching NLTK Norwegian*. 2017.

[12] Conor O'Sullivan. *Deep Neural Network Language Identification*. https://towardsdatascience.com/deep-neural-network-language-identification-ae1c158f6a7d. Accessed: 2022-01-18.

[13] Jun Yan. 'Text Representation'. In: *Encyclopedia of Database Systems*. Ed. by LING LIU and M. TAMER ÖZSU. Boston, MA: Springer US, 2009, pp. 3069–3072. ISBN: 978-0-387-39940-9.

[14] Daniel Jurafsky and James H. Martin. *N-gram Language Models*. https://web.stanford.edu/~jurafsky/slp3/3.pdf. Accessed: 2022-01-28.

[15] University of Cincinnati. *Creating text features with bag-of-words, n-grams, parts-of-speach and more*. http://uc-r.github.io/creating-text-features. Accessed: 2022-01-28.

[16] Aurelien Geron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 2nd. O'Reilly Media, Inc., 2019. ISBN: 1492032646.

[17] Tom Landauer and Walter Kintsch. *What is LSA?* http://lsa.colorado.edu/whatis.html. Accessed: 2022-01-25.

[18] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008. ISBN: 978-0-521-86571-5. URL: https://nlp.stanford.edu/IR-book/pdf/irbookprint.pdf.

[19] Mohammed J. Zaki and Jr Wagner Meira. *Data Mining and Machine Learning: Fundamental Concepts and Algorithms*. 2nd. Cambridge University Press, 2020. ISBN: 978-1108473989.

[20] O. M. Ventista. 'Self-assessment in Massive Open Online Courses'. In: *E-Learning and Digital Media* 15.4 (2018), pp. 165–175.

[21] Claudia Leacock and Martin Chodorow. 'C-rater: Automated Scoring of Short-Answer Questions'. In: *Comput. Humanit.* 37.4 (2003), pp. 389–405.

[22] ETS. *About the e-rater® Scoring Engine*. https://www.ets.org/erater/about. Accessed: 2021-09-21.

[23] A. Horbach. 'Analyzing Short-Answer Questions and their Automatic Scoring'. PhD thesis. Saarland University, Oct. 2019.

[24] Kaveh Taghipour and Hwee Tou Ng. 'A Neural Approach to Automated Essay Scoring'. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. The Association for Computational Linguistics, 2016, pp. 1882–1891.

[25] J. Burstein, S. Wolff, C. Lu and R. Kaplan. 'Using Lexical Semantic Techniques to Classify Free-Responses'. In: *Breadth and Depth of Semantic Lexicons*. 1996.

[26] David Callear, Jennifer Jerrams-Smith and Victor Soh. 'CAA of Short Non-MCQ Answers'. In: 2001.

[27] J. Cowie and Y. Wilks. 'Information extraction'. In: *Handbook of Natural Language Processing* (), pp. 241–260.

[28] Shourya Roy, Y. Narahari and Om D. Deshmukh. 'A Perspective on Computer Assisted Assessment Techniques for Short Free-Text Answers'. In: *Computer Assisted Assessment. Research into E-Assessment - 18th International Conference, CAA 2015, Zeist, The Netherlands, June 22-23, 2015. Proceedings*. Vol. 571. Communications in Computer and Information Science. Springer, pp. 96–109.

[29] Tom Mitchell, Terry Russell, Peter Broomhead and Nicola Aldridge. 'Towards robust computerised marking of free-text responses'. In: (2002).

[30] U. Hasanah, A. E. Permanasari, S. S. Kusumawardani and F. S. Pribadi. 'A review of an information extraction technique approach for automatic short answer grading'. In: *2016 1st International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*. 2016, pp. 192–196.

[31] Lyle F. Bachman, Nathan Carr, Greg Kamei, Mikyung Kim, Michael J. Pan, Chris Salvador and Yasuyo Sawaki. 'A Reliable Approach to Automatic Assessment of Short Answer Free Responses'. In: *19th International Conference on Computational Linguistics, COLING 2002, Howard International House and Academia Sinica, Taipei, Taiwan, August 24 - September 1, 2002*. 2002.

[32] S. E. Jordan. 'Short-answer e-assessment questions: five years on'. In: 2012.

[33] P. Thomas. 'The evaluation of electronic marking of examinations'. In: *Proceedings of the 8th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, ITiCSE 2003, Thessaloniki, Greece, June 30 - July 2, 2003*. ACM, 2003, pp. 50–54.

[34] J. Sukkarieh S. Pulman N. Raikes. 'Auto-marking: Using computational linguistics to score short, free text responses'. In: (2003).

[35] Dezsö Sima, Balázs Schmuck, Sándor Szöllosi and Árpád Miklós. 'Intelligent Short Text Assessment in eMax'. In: *Towards Intelligent Engineering and Information Technology*. Vol. 243. Studies in Computational Intelligence. Springer, 2009, pp. 435–445.

[36] Sally E. Jordan and Tom Mitchell. 'e-Assessment for learning? The potential of short-answer free-text questions with tailored feedback'. In: *Br. J. Educ. Technol.* 40.2 (2009), pp. 371–385.

[37] Laurie Ane Cutrone, Maiga Chang and Kinshuk. 'Auto-Assessor: Computerized Assessment System for Marking Student's Short-Answers Automatically'. In: *2011 IEEE International Conference on Technology for Education, T4E 2011, Chennai, Tamil Nadu, India, July 14-16, 2011*. IEEE Computer Society, 2011, pp. 81–88.

[38] Michael Hahn and Detmar Meurers. 'Evaluating the Meaning of Answers to Reading Comprehension Questions: A Semantics-Based Approach'. In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, BEA@NAACL-HLT 2012, June 7, 2012, Montréal, Canada*. The Association for Computer Linguistics, 2012, pp. 326–336.

[39] Kurtis Pykes. *Part Of Speech Tagging for Beginners*. https://towardsdatascience.com/part-of-speech-tagging-for-beginners-3a0754b2ebba. Accessed: 2022-01-18.

[40] Princeton University. *WordNet*. https://wordnet.princeton.edu/. Accessed: 2021-11-22.

[41] Benoit Lemaire and Philippe Dessus All My Papers Are At pdessus.fr. 'A System To Assess The Semantic Content Of Student Essays'. In: *Journal of Educational Computing Research* 24 (Oct. 2003).

[42] Enrique Alfonseca and Diana Pérez. 'Automatic Assessment of Open Ended Questions with a Bleu-Inspired Algorithm and Shallow NLP'. In: *Advances in Natural Language Processing, 4th International Conference, EsTAL 2004, Alicante, Spain, October 20-22, 2004, Proceedings*. Vol. 3230. Lecture Notes in Computer Science. Springer, 2004, pp. 25–35.

[43] Richard Klein, Angelo Kyrilov and Mayya Tokman. 'Automated assessment of short free-text responses in computer science using latent semantic analysis'. In: *Proceedings of the 16th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, ITiCSE 2011, Darmstadt, Germany, June 27-29, 2011*. ACM, 2011, pp. 158–162.

[44] Lishan Zhang, Yuwei Huang, Xi Yang, Shengquan Yu and Fuzhen Zhuang. 'An automatic short-answer grading model for semi-open-ended questions'. In: *Interactive Learning Environments* (2019).

[45] Sridevi Bonthu, S. Rama Sree and Munaga H. M. Krishna Prasad. 'Automated Short Answer Grading Using Deep Learning: A Survey'. In: *Machine Learning and Knowledge Extraction - 5th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2021, Virtual Event, August 17-20, 2021, Proceedings*. Vol. 12844. Lecture Notes in Computer Science. Springer, 2021, pp. 61–78.

[46] Stacey Bailey and Detmar Meurers. 'Diagnosing Meaning Errors in Short Answers to Reading Comprehension Questions'. In: EANL '08. Association for Computational Linguistics, 2008, pp. 107–115.

[47] Rodney D. Nielsen, Wayne H. Ward and James H. Martin. 'Learning to Assess Low-Level Conceptual Understanding'. In: *Proceedings of the Twenty-First International Florida Artificial Intelligence Research Society Conference, May 15-17, 2008, Coconut Grove, Florida, USA*. AAAI Press, 2008, pp. 427–432.

[48] Neslihan Suzen, Alexander N. Gorban, Jeremy Levesley and Evgeny M. Mirkes. 'Automatic Short Answer Grading and Feedback Using Text Mining Methods'. In: *CoRR* abs/1807.10543 (2018). arXiv: 1807.10543. URL: http://arxiv.org/abs/1807.10543.

[49] Shailaja Jayashankar and R. Sridaran. 'Superlative model using word butt for short answers evaluation in eLearning'. In: *Educ. Inf. Technol.* 22.5 (2017), pp. 2383–2402.

[50] Sumit Basu, Chuck Jacobs and Lucy Vanderwende. 'Powergrading: a Clustering Approach to Amplify Human Effort for Short Answer Grading'. In: *Trans. Assoc. Comput. Linguistics* 1 (2013), pp. 391–402.

[51] Akif Yılmaz Ince Ebru Kutlu. 'Web-Based Turkish Automatic Short-Answer Grading System'. In: *Natural Language Processing Research* 1 (2021).

[52] Anak Agung Putri Ratna, Prima Dewi Purnamasari and Boma Anantasatya Adhi. 'Simple-O, An Automated Essay Grading System for Indonesian Language Using the LSA Method with Multi-Level Keywords'. In: 2015.

[53] Kaggle. *The Hewlett Foundation: Short Answer Scoring*. https://www.kaggle.com/c/asap-sas. Accessed: 2022-01-28.

[54] Torsten Zesch, Michael Wojatzki and Dirk Scholten-Akoun. 'Task-Independent Features for Automated Essay Grading'. In: *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*. The Association for Computer Linguistics, 2015, pp. 224–232.

[55] Keisuke Sakaguchi, Michael Heilman and Nitin Madnani. 'Effective Feature Integration for Automated Short Answer Scoring'. In: *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*. The Association for Computational Linguistics, 2015, pp. 1049–1054.

[56] Daniel Bär, Torsten Zesch and Iryna Gurevych. 'A Reflective View on Text Similarity'. In: *Recent Advances in Natural Language Processing, RANLP 2011, 12-14 September, 2011, Hissar, Bulgaria*. RANLP 2011 Organising Committee, 2011, pp. 515–520.

[57] Jesus Gerardo Alvarado Mantecon. 'Towards the Automatic Classification of Student Answers to Open-ended Questions'. MA thesis. Canada: University of Ottawa, 2019.

[58] Jesus Gerardo Alvarado Mantecon, Hadi Abdi Ghavidel, Amal Zouaq, Jelena Jovanovic and Jenny McDonald. 'A Comparison of Features for the Automatic Labeling of Student answers to Open-ended Questions'. In: *Proceedings of the 11th International Conference on Educational Data Mining, EDM 2018, Buffalo, NY, USA, July 15-18, 2018*. International Educational Data Mining Society (IEDMS), 2018.

[59] Marcelo Guerra Hahn, Silvia Margarita Baldiris Navarro, Luis de la Fuente Valentın and Daniel Burgos. 'A Systematic Review of the Effects of Automatic Scoring and Automatic Feedback in Educational Settings'. In: *IEEE Access* 9 (2021), pp. 108190–108198.

[60] Denise Whitelock and Duygu Bektik. 'Progress and Challenges for Automated Scoring and Feedback Systems for Large-Scale Assessments'. In: 2018.

[61] Rudragouda G. Patil and Syed Zakir Ali. 'Approaches For Automation In Assisting Evaluator For Grading Of Answer Scripts: A Survey'. In: *2018 4th International Conference on Computing Communication and Automation (ICCCA)*. 2018, pp. 1–6.

[62] NTNU. *IT2810 - Web Development*. https://www.ntnu.edu/studies/courses/IT2810#tab=omEmnet. Accessed: 2021-09-21.

[63] Petri Ihantola, Tuukka Ahoniemi, Ville Karavirta and Otto Seppälä. 'Review of recent systems for automatic assessment of programming assignments'. In: *10th Koli Calling International Conference on Computing Education Research, Koli Calling '10, Koli, Finland, October 28-31, 2010*. ACM, 2010, pp. 86–93.

[64] Kirsti Ala-Mutka. 'A Survey of Automated Assessment Approaches for Programming Assignments'. In: *Comput. Sci. Educ.* 15.2 (2005), pp. 83–102.

[65] Pandas. *Intro to data structures*. https://pandas.pydata.org/pandas-docs/stable/user_guide/dsintro.html. Accessed: 2022-01-19.

[66] Inspera. *Demo home exams and submissions*. https://ntnu.inspera.no/. Accessed: 2022-01-31.

[67] Monash University. *Citing and referencing: Vancouver*. https://guides.lib.monash.edu/citing-referencing/vancouver. Accessed: 2022-01-19.

[68] R. Al-Rfou. *Welcome to polyglot's documentation!* https://polyglot.readthedocs.io/en/latest/. Accessed: 2021-10-12.

[69] German Lahera. *Unbalanced Datasets What To Do About Them*. https://medium.com/strands-tech-corner/unbalanced-datasets-what-to-do-144e0552d9cd. Accessed: 2021-12-02.

[70] Sundaresh Chandran. *Significance of I.I.D in Machine Learning*. https://medium.datadriveninvestor.com/significance-of-i-i-d-in-machine-learning-281da0d0cbef. Accessed: 2021-12-18.

[71] B Ratner. 'The correlation coefficient: Its values range between +1/1, or do they?' In: *J Target Meas Anal Mark* 17 (17), pp. 139–142.

[72] Maram F. Al-Jouie and Aqil M. Azmi. 'Automated Evaluation of School Children Essays in Arabic'. In: *ACLING*. 2017.

[73] Spacy. *Lemmatizer*. https://spacy.io/api/lemmatizer. Accessed: 2022-01-13.

[74] NLTK. *Source code for nltk.stem.snowball*. https://www.nltk.org/_modules/nltk/stem/snowball.html. Accessed: 2022-01-13.

[75] Matt Chaput. *Whoosh! - Release notes*. https://whoosh.readthedocs.io/en/latest/releases/index.html. Accessed: 2022-01-13.

[76] A. Tucker and Geneva G. Belford. *computer science*. https://www.britannica.com/science/computer-science. Accessed: 2022-01-31.

Appendix A

Exam Questions

The three tables below lists the exam questions for E20 (Table 1), E19 (Table 2), E18 (Table 3). As the text is copied from the original exam the text is written in either Norwegian Bokmål or Norwegian Nynorsk.

Q#	Question Description
1	Gjer reie for mekanismar og teknikkar som blir brukt for tilstand (state) og dataflyt i React- applikasjoner.
2	Recoil er eit Facebook open source bibliotek for React. Bruk dokumentasjonen på web for å få eit innblikk i dette. Forklar kort kva dette er og samanlikn med tilsvarande løysingar som vi har på lista over læringsmål, og diskuter mulige fordeler og ulemper ved dette biblioteket og bruken av det i utvikling.
3	Diskuter vesentlege forskjellar mellom REST API og Graphql.

Table A.1: Exam questions from E20.

Q#	Question Description
1	1. Hva er responsiv web design? 2. Nevn forskjellige teknikker som brukes for å implementere responsiv webdesign.
2	Hvordan implementeres interaktivitet på "figurnivå" i henholdsvis SVG/DOM og med Canvas API'et? Med interaktivitet på figurnivå menes at bruker skal kunne klikke på enkeltelementer og utføre handliger på disse (f.eks. flytte en sirkel, endre farge på et rektangel etc)
3	REST og GraphQL er to forskjellige løsninger for klient-server kommunikasjon i web- applikasjoner. Beskriv og diskuter disse kort (legg vekt på å sammenligne).
4	Forklar kort hva følgende former for testing er (og hva de brukes til å teste). • Cross-browser testing • Enhetstesting • Snapshot-testing • End-to-end testing
5	Hva er de to viktige forskjellene mellom en variabel som er deklarert med const i Javascript ES6 og en variabel som er deklarert med var (pre ES6)?
6	Forklar hva funksjonen groupBy gjør og gi et eksempel på bruk gitt cars-variabelen? Vi er ute etter overordnet funksjonalitet og bruk – og ikke detaljene i kallet til reduce*.
7	Du skal være med å utvikle en søkeapplikasjon for en samling av vitenskapelige artikler. Det skal være mulig å søke på forfatter, emne, tidsskrift, år, tittelord og databasen inneholder omtrent 1 million artikler. Det skal være mulig å filtrere og sortere resultatsettet som returneres fra et søk. Lag en punktliste med opp til 5 gode råd for design og arkitekturen til systemet og argumenter etterpå kort for hvorfor disse er viktige.
8	Hva er hensikten og fordelene med global state management i React applikasjoner?

Table A.2: Exam questions from E19.
* Appended to this question was an image containing some JaveScript code.

Q#	Question Description
1	Forklar kort hvilket scope som gjelder for variabler som er deklarert med nøkkelordet var og hvilket scope gjelder for variabler deklarert med nøkkelordet let?
2	Forklar kort hvordan arrow-funksjoner skiller seg fra vanlige funksjoner i Javascript, med tanke pa this
3	Forklar kort hva CSS-grid er og CSS-flexbox er. Beskriv hvilket problem/behov de løser og hva som skiller disse to løsningene.
4	SVG og HTML5 Canvas kan begge brukes til a lage interaktiv grafikk pa websider. Forklar kort hva begge er og gi et eksempel på (og argument) for en anvendelse hvor SVG er godt egnet og en anvendelse hvor HTML5 Canvas er godt egnet.
5	Hva er selector-mekanismen i jQuery. Gi et eksempel og en kort forklaring.
6	Hva kjennetegner en SPA (Single Page Application)?
7	Hva er responsiv web design? Nevn forskjellige teknikker som brukes for å implementere responsiv webdesign.
8	Lag en React-komponent kalt HelloWorld for et H1 element med teksten "Hello World!".
9	Forklar kort hva props og state er i React
10	Forklar hvordan du må implementere dataflyt oppover i et React komponenthierarki.
11	Hvilken funksjonalitet tilbys gjennom HTML5 Web storage api'et (og det tilsvarende AsyncStorage api'et i React native)?
12	Beskriv hva som typisk kan gjenbrukes og hva som typisk ikke kan gjenbrukes hvis du skal gjøre om en React for web applikasjon til React native.
13	Hva er og hvorfor bruker vi state management som Redux og Mobx? Gi eksempel på hvordan disse brukes i implementasjonen (dvs. vis litt kode).
14	Forklar hva snapshot-testing er?
15	Forklar kort hva REST er eller hva GraphQL er (velg en av disse). Vis eksempel.
16	Forklar hva end to end testing er?

Table A.3: Exam questions from E18.

Appendix B

Word Count per Question and Grade Distribution

The tables are organized such that the data in each row is connected to the question listed in the first column. The following abbreviations are used in the column names:

- **Q**: Number of questions
- **A**: Number of answers for that specific question or receiving that specific grade
- **W.C.**: Word count
- ***Grade* {0-5}**: How many students received that specific grade

Q	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.	0	1	2	3	4	5
1	213	390	363	1	1049	1	3	23	67	105	14
2	213	355	343	88	978	0	0	11	59	132	11
3	213	324	305	1	773	1	2	6	91	99	14

Table B.1: Statistics describing answers to each question in Bokmål DS E20.

Q	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.	0	1	2	3	4	5
1	182	106	95	30	342	0	1	5	19	26	131
2	182	130	122	15	473	3	12	22	45	79	21
3	182	127	113	1	595	10	6	45	57	46	18
4	182	199	190	69	474	0	1	14	18	39	110
5	182	78	64	11	270	4	8	15	15	97	43
6	181	58	55	1	182	20	23	116	6	1	15
7	181	238	232	1	621	3	9	33	77	52	7
8	182	110	101	1	364	3	8	6	47	99	19

Table B.2: Statistics describing answers to each question in Bokmål DS E19.

Q	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.	0	1	2	3	4	5
1	159	49	47	1	141	9	8	20	65	15	42
2	159	40	36	1	131	31	20	34	29	20	25
3	159	94	88	23	249	0	1	8	91	47	12
4	159	125	123	32	276	0	3	22	104	22	8
5	159	53	51	0	126	14	10	7	72	38	18
6	159	51	46	2	181	4	2	19	61	45	28
7	159	106	99	22	275	0	3	13	66	51	26
8	-	-	-	-	-	-	-	-	-	-	-
9	159	80	74	10	207	3	3	9	22	42	80
10	159	57	55	1	200	48	5	7	3	6	90
11	159	62	52	1	219	12	5	5	17	24	96
12	159	75	63	1	273	9	1	16	45	43	45
13	159	150	142	1	460	3	4	19	32	27	74
14	159	62	58	1	157	5	1	3	8	18	124
15	159	110	94	1	401	5	3	7	28	19	97
16	159	78	68	1	293	8	3	2	6	14	126

Table B.3: Statistics describing answers to each question in Bokmål DS E18.

Appendix C

Word Count per Grade per Question

The tables are organized such that the data in each row is connected to the grade listed in the first column. The following abbreviations are used in the column names:

- **Q**: Number of questions
- **A**: Number of answers for that specific question or receiving that specific grade
- **W.C.**: Word count
- ***Grade* {0-5}**: How many students received that specific grade

Word Length and Grade Distribution in E20

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	1	1	1	1	1
1	3	117	112	69	171
2	23	208	209	106	484
3	67	334	300	139	963
4	105	459	411	192	1049
5	14	529	514	327	1036

Table C.1: Statistics describing answers to question 1 receiving the different grades in Bokmål DS E20.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	0	-	-	-	-
1	0	-	-	-	-
2	11	170	144	88	384
3	59	270	245	110	621
4	132	398	380	182	978
5	11	488	503	300	793

Table C.2: Statistics describing answers to question 2 receiving the different grades in Bokmål DS E20.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	1	1	1	1	1
1	2	220	220	41	399
2	6	145	123	91	224
3	91	292	267	112	629
4	99	354	327	133	773
5	14	437	393	263	689

Table C.3: Statistics describing answers to question 3 receiving the different grades in Bokmål DS E20.

Word Length and Grade Distribution in E19

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	0	-	-	-	-
1	1	112	112	112	112
2	5	68	74	38	88
3	19	109	114	32	232
4	26	115	114	30	206
5	131	105	91	33	342

Table C.4: Statistics describing answers to question 1 receiving the different grades in Bokmål DS E19.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	3	66	72	45	81
1	12	51	52	15	94
2	22	111	98	37	213
3	45	108	98	37	273
4	79	150	136	67	302
5	21	177	162	52	473

Table C.5: Statistics describing answers to question 2 receiving the different grades in Bokmål DS E19.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	10	4	1	1	15
1	6	23	22	11	35
2	45	77	74	26	192
3	57	123	116	38	270
4	46	176	161	63	595
5	18	243	235	99	416

Table C.6: Statistics describing answers to question 3 receiving the different grades in Bokmål DS E19.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	0	-	-	-	-
1	1	90	90	90	90
2	14	162	151	69	290
3	18	202	152	99	438
4	39	184	167	95	401
5	110	210	207	88	474

Table C.7: Statistics describing answers to question 4 receiving the different grades in Bokmål DS E19.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	4	63	47	13	144
1	8	28	29	11	55
2	15	43	41	13	90
3	15	53	50	12	135
4	97	76	66	15	208
5	43	115	116	34	270

Table C.8: Statistics describing answers to question 5 receiving the different grades in Bokmål DS E19.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	20	24	11	1	126
1	23	36	36	9	92
2	116	64	58	23	182
3	6	99	81	38	179
4	1	81	81	81	81
5	15	76	62	25	164

Table C.9: Statistics describing answers to question 6 receiving the different grades in Bokmål DS E19.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	3	1	1	1	1
1	9	197	235	32	299
2	33	207	205	52	501
3	77	256	239	65	621
4	52	239	230	82	484
5	7	343	318	214	609

Table C.10: Statistics describing answers to question 7 receiving the different grades in Bokmål DS E19.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	3	40	25	1	93
1	8	55	40	18	157
2	6	79	72	53	131
3	47	87	82	26	194
4	99	126	124	44	364
5	19	131	114	64	237

Table C.11: Statistics describing answers to question 8 receiving the different grades in Bokmål DS E19.

Word Length and Grade Distribution in E18

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	9	30	37	1	58
1	8	31	28	14	49
2	20	54	59	9	138
3	65	47	45	10	141
4	15	53	48	18	95
5	42	55	49	6	135

Table C.12: Statistics describing answers to question 1 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	31	27	28	1	50
1	20	34	29	13	87
2	34	42	37	13	113
3	29	47	41	16	131
4	20	41	36	10	97
5	25	51	46	18	108

Table C.13: Statistics describing answers to question 2 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	0	-	-	-	-
1	1	45	45	45	45
2	8	65	64	31	99
3	91	92	88	23	207
4	47	94	89	42	229
5	12	128	120	43	249

Table C.14: Statistics describing answers to question 3 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	0	-	-	-	-
1	3	66	70	49	78
2	22	78	71	32	172
3	104	132	133	52	261
4	22	149	141	69	276
5	8	128	119	34	229

Table C.15: Statistics describing answers to question 4 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	14	18	12	0	54
1	10	28	30	9	54
2	7	53	49	31	83
3	72	54	52	11	126
4	38	65	67	26	126
5	18	64	62	31	117

Table C.16: Statistics describing answers to question 5 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	4	9	7	2	19
1	2	27	27	6	47
2	19	34	33	7	61
3	61	48	39	6	173
4	45	55	54	14	109
5	28	69	59	25	181

Table C.17: Statistics describing answers to question 6 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	0	-	-	-	-
1	3	59	31	22	123
2	13	75	61	37	140
3	66	100	90	41	249
4	51	119	117	29	275
5	26	119	106	51	254

Table C.18: Statistics describing answers to question 7 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	3	21	19	11	33
1	3	33	16	13	71
2	9	57	44	10	143
3	22	65	57	22	110
4	42	78	69	19	201
5	80	92	89	23	207

Table C.19: Statistics describing answers to question 9 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	48	45	37	1	134
1	5	28	28	2	71
2	7	53	49	22	84
3	3	32	27	15	55
4	6	50	47	20	87
5	90	67	63	10	200

Table C.20: Statistics describing answers to question 10 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	12	27	16	1	95
1	5	29	30	22	39
2	5	70	72	34	115
3	17	65	48	14	167
4	24	61	61	12	168
5	96	67	57	13	219

Table C.21: Statistics describing answers to question 11 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	9	13	9	1	35
1	1	65	65	65	65
2	16	49	46	11	118
3	45	54	51	12	110
4	43	89	80	30	240
5	45	105	91	33	273

Table C.22: Statistics describing answers to question 12 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	3	17	16	1	33
1	4	91	90	24	159
2	19	61	61	25	100
3	32	125	127	37	262
4	27	162	150	66	355
5	74	189	177	68	460

Table C.23: Statistics describing answers to question 13 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	5	15	12	1	39
1	1	17	17	17	17
2	3	32	34	26	37
3	8	49	41	15	122
4	18	53	54	26	125
5	124	67	66	18	157

Table C.24: Statistics describing answers to question 14 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	5	16	17	1	32
1	3	32	31	16	48
2	7	75	69	19	191
3	28	72	68	19	264
4	19	109	100	34	273
5	97	130	115	44	401

Table C.25: Statistics describing answers to question 15 receiving the different grades in Bokmål DS E18.

Grade	A	Mean W.C.	Median W.C.	Min W.C.	Max W.C.
0	8	48	48	1	112
1	3	32	36	18	43
2	2	45	45	42	47
3	6	45	44	26	66
4	14	51	53	15	95
5	129	87	78	10	293

Table C.26: Statistics describing answers to question 16 receiving the different grades in Bokmål DS E18.

Appendix D

Norwegian Stopwords

- a
- å
- æ
- alle
- alle
- andre
- at
- av
- b
- både
- båe
- bare
- begge
- ble
- blei
- bli
- blir
- blitt
- bort
- bra
- bruke
- c
- ca
- d
- da
- då
- de
- deg
- dei
- deim
- deira
- deires
- dem
- den
- denne
- der
- dere
- deres
- deretter
- det
- dette
- di
- din
- disse
- ditt
- du
- dvs
- dykk
- dykkar
- e
- eg
- ei
- ein
- eit
- eitt
- eks
- eksempel
- eksempelvis
- eller
- ellers
- en
- én
- èn
- ene
- éne
- eneste

- enhver
- enn
- ennå
- ens
- er
- et
- etc
- ett
- etter
- evt
- ex
- exempel
- f
- få
- folk
- for
- før
- fordi
- foreksempel
- forsøke
- først
- fra
- g
- gå
- gjor
- gjør
- gjord
- gjorde
- god
- h
- ha
- hadde
- han
- hans
- har
- hennar
- henne
- hennes
- her
- hhv
- hjå
- ho
- hoe
- honom
- hoss
- hossen
- hun
- hun
- hva
- hvem
- hver
- hvilke
- hvilken
- hvis
- hvor
- hvordan
- hvorfor
- i
- i
- ift
- ikke
- ikkje
- ingen
- ingi
- inkje
- inn
- innen
- inni
- j
- ja
- jeg
- jo
- k
- kan
- kom
- korleis
- korso
- kun
- kunne
- kva
- kvar
- kvarhelst
- kven
- kvi
- kvifor
- l
- la
- lage
- lang
- lik
- like
- likevel
- m
- må
- makt
- man

- mange
- måtte
- me
- med
- medan
- meg
- meget
- mellom
- men
- mens
- mer
- mest
- mi
- min
- mine
- mitt
- mm
- mot
- mtp
- mye
- mykje
- n
- nå
- når
- når
- navn
- ned
- nei
- no
- noe
- noen
- noka
- noko
- nokon
- nokor
- nokre
- ny
- o
- ø
- of
- og
- òg
- også
- ok
- okei
- ol
- om
- opp
- oss
- osv
- over
- p
- på
- part
- pga
- punkt
- q
- r
- rett
- riktig
- s
- så
- samme
- sånn
- sant
- seg
- selv
- si
- sia
- sidan
- siden
- sin
- sine
- sist
- sitt
- sjøl
- skal
- skulle
- slik
- slutt
- so
- som
- somme
- somt
- start
- stille
- t
- tid
- til
- tilbake
- u
- um
- under
- upp
- ut
- uten

- v
- være
- vært
- var
- vår
- vår
- vart
- varte
- ved
- vere
- verte
- vi
- vil
- ville
- vite
- vore
- være
- vors
- vort
- vørt
- vs
- w
- x
- y
- z

Pernille Andresen Klevstuen

Assisting efficient and fair grading

NTNU
Norwegian University of
Science and Technology