Leif Ulvund

Explaining fake news

Master's thesis in Informatikk
Supervisor: Jon Atle Gulla
Co-supervisor: Yujie Xing
June 2021

Master's thesis

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

NTNU
Kunnskap for en bedre verden

Leif Ulvund

Explaining fake news

Master's thesis in Informatikk
Supervisor: Jon Atle Gulla
Co-supervisor: Yujie Xing
June 2021

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 current news ecosystem faces a significant challenge due to the amount of fake news being published. Even if a news article can be automatically detected as fake, it is still challenging to explain the difference between real and fake news. One avenue that can differ is the semantics of the text. These differences can be visualized and help create an understanding of what makes fake news fake. This thesis evaluates the use of contextualized language models on two semantic change tasks and analyses real and fake news to detect lexical semantic change between them.

To evaluate their performance, the three language models BERT, GPT-2 and XL-Net are assessed using a semantic change framework that consists of a graded and a binary change detection task, and a diachronic corpus of text. The models are evaluated on how well their ranking of which words have changed the most correlate with those of human annotators, and their accuracy in detecting words that are marked as changed. The best performing model is chosen to analyse news data for types of semantic change undergone and for which words these changes are prevalent.

The main results show that there is a large difference in how contextual language models perform on these tasks. BERT achieves a correlation of 0.646 after fine-tuning, which is slightly higher than previous usages of BERT, and higher than comparable methods used in the literature. It is also able to achieve a correlation of 0.547 pre-trained after removing the part-of-speech tag appended to every target word. GPT-2 and XLNet are able to beat the baselines, but do not perform better than comparable methods. When used to analyze news, the findings show that multiple types of semantic change are present, but also pinpoints areas where further work is important to reduce the level of noise in the data. These are areas such as removing many of the very similar sentences that are repeated often in news, and that are specific to the source that published the article.

Sammendrag

Dagens nyhetsbilde står forran en stor utfordring på grunn av mengden falske nyheter som florerer. Selv om en nyhetsartikkel kan bli automatisk detektert som falsk er det fremdeles vanskelig å forklare forskjellen mellom ekte og falske nyheter. En måte de kan skille seg på er semantikken i teksten. Disse endringene kan visualiseres og øke vår forståelse av hva som gjør falske nyheter falsk. Denne oppgaven evaluerer bruken av kontekstualiserte språkmodeller på to oppgaver om semantisk endring og analyserer ekte of falske nyheter for å detektere semantiske forskjeller mellom dem.

For å evaluere ytelsen deres blir de tre språkmodellene BERT, GPT-2 og XLNet testet ved å bruke et rammeverk for semantisk endring. Dette består av en rangerings- og en binær klassifiseringsoppgave og et diakronisk korpus av tekst. Modellene er evaluert etter hvor bra rangeringene av ord som endrer seg mest korrelerer med mennesker, og nøyaktigheten deres i å klassifisere ord som er markert som endret. Den modellen som yter best blir valgt til å analysere nyhetsdataen etter typer av semantiske endringer og for hvilke ord disse endringene er synlige.

Hovedresultatene viser at det er en stor forskjell i ytelsen til kontekstualiserte språkmodeller på disse oppgavene. BERT oppnår en korrelasjon på 0.646 etter videre trening, noe som er litt bedre enn tidligere forsøk med BERT, og høyere enn tilsvarende metoder fra litteraturen. Modellen oppnår også en korrelasjon på 0.547 uten videre trening om man fjerner ordklassetaggen tilknyttet hvert ord som testes. GPT-2 og XLNet klarer å slå grunnlinje-testene, men er ikke bedre enn tilsvarende metoder. Brukt til å analysere nyheter viser resultatene at flere typer semantiske endringer kan observeres, men påpeker også områder der fremtidig arbeid er vikig for å redusere nivået av støy i dataen. Dette innebærer å fjerne mange av de veldig like setningene som repeteres ofte i nyheter, og som er spesifikke til kilden som publiserer dem.

Preface

This master thesis was submitted to the Norwegian University of Science and Technology (NTNU), Department of Computer Science (IDI) as part of the course IT3902 - Informatics Postgraduate Thesis: Database Management and Search.

I would like to thank my supervisor Jon Atle Gulla, and my co-supervisor Yujie Xing for their invaluable guidance and feedback throughout all stages of the project. I would also like to thank my friends and family for their unwavering support through the whole course of my studies.

Contents

Figures

Tables

Acronyms

ALBERT A Lite BERT.

APD Average Pairwise Distance.

BERT Bidirectional Encoder Representations from Transformers.

CCOHA Clean Corpus of Historical American English.

GPT-2 Generative Pretrained Transformer-2.

GRU Gated Recurrent Unit.

JSD Jensen-Shannon Distance.

LSC Lexical Semantic Change.

LSTM Long Short-Term Memory.

MLM Masked Language Modelling.

NELA NELA-GT-2019.

NLP Natural Language Processing.

NSP Next Sentence Prediction.

PCA Principal Component Analysis.

POS part-of-speech.

PRT Inverted Cosine Similarity Over Word Prototypes.

RNN Recurrent Neural Network.

RoBERTa Robustly Optimized BERT Pretraining Approach.

WSI Word Sense Induction.

Chapter 1

Introduction

This chapter gives an overview of the main components of this thesis. The motivation for tackling the given problem, the goals and research questions are stated, and the approach taken and results are briefly mentioned. Following the result is an outline for the rest of the thesis.

1.1 Motivation

Around the time of the US election in 2016, the phrase fake news entered the vocabulary of the general public. The term might be new, but it encapsulates a problem that far precedes it. The need for journalistic standards arose after the rampant use of propaganda during World War I. Since then, the consumption of news has changed dramatically with the advent of broadcasting, television and the internet. The latter has greatly reduced the cost of creating news content, enabling less structured actors than traditional news sources to reach a large audience quickly. The spread of news content on social media allows people to surround themselves with stories that subscribe to their world view. Since people are more likely to believe stories they agree with [1], this facilitates a breeding ground for fake news and the sources that make them.

Exactly how prevalent fake news has become is a question with few good answers. The amount of fake news continues to increase, and its consequences can be adverse. After the outbreak of the COVID-19 pandemic, the amount of both truthful news and misinformation published that cover this topic reached such a high level that the World Health Organization warned of an infodemic [2]. [3] states that just under half of the US and UK population reported to have read fake news related to the virus, and that two thirds of them were exposed to fake news every day. [4] shows that people who are prone to believe conspiracies and that show distrust in authority figures are also prone to believe COVID-19 conspiracies.

In today's digital media ecosystem, a news article that generates high levels of

engagement gives the outlet more revenue than an article which doesn't. The more people who read the article, the more ads are being watched. Because of this, a situation arises where the path of veracious news and profitable news can diverge. Directly increasing revenue is not the only lucrative aspect of fake news though. Creating articles with false defamatory statements about political opponents can have lasting effects even if the stories are later debunked. [5] reports that readers who consume fake news are not likely to consume news correcting these claims, and that the readers who do tends to show a negative sentiment towards it, and a stronger affinity to consume more fake news afterwards.

One of the earliest approaches taken to combat fake news was by manually assessing the correctness of claims. Sites like Politifact, Snopes and Faktisk have teams of journalists and experts that work on an article to article basis. With the ever increasing amount of fake news, this leads to a gap in how fast it is created versus checked. To bridge this gap, automated systems exploiting advances in natural language processing and machine learning have started to show success.

Another important field in the research on fake news is fake news detection, which is described in more detail in section 2.1. One important component largely missing from the field of fake news research is to be able to actually explain what properties that separate real and fake news, and the observable changes between them. While work has been done on trying to explain linguistic differences in fake news [6], not much has been done to explore how the semantics of news change as one venture from real to fake news. With the explosive advancements of language models and NLP in general in the past years, this could be a valuable tool to help us better understand the concept of fake news.

1.2 goals and research questions

Semantic change can be defined as the difference that appears in the meaning of words as they are used in separate time periods or in differing domains such as political discourse or a science fiction book. Vector embeddings can be used to capture the semantics of these words, with contextual embeddings created from many of the emerging language models even being able to create unique embeddings for a word based on the context that surrounds it.

The goal of this thesis is to explain differences in real and fake news by exploring how contextualized embeddings created from different fine-tuned language models perform on two semantic change tasks, and to use the best performing one to analyze a news dataset to search for semantic differences between the two classes of news.

The research questions are shown below:

RQ1 How do different fine-tuned contextual language models perform on the task of semantic change detection?

RQ2 How does the effect of processing the evaluation corpus affect the results of these models?

RQ3 To what degree does the laws of semantic change hold for synchronous news data?

RQ4 Do fake and real news show semantic differences when comparing word senses from a contextual language model?

1.3 approach

The two main approaches to explaining fake news are either to look at news in general, to see what distinguishes real and fake news on an aggregate level, or to look at one particular article and detect which statements in the article are false. At the aggregate level, semantic properties can help both researchers in further work on fake news and readers in getting a view of what separates real and fake news.

To get an understanding of how models perform on the semantic change detection task they must be evaluated in a structured and reproducible manner. The most state-of-the-art evaluation framework for semantic change was proposed by SemEval-2020 [7]. It consists of a labeled dataset and two subtasks. This framework was used to evaluate three different contextualized language models, namely BERT, GPT-2, and XLNet using three change detection metrics. While the models all originate from the same architecture, the transformer, they have many differences resulting in individual strengths and weaknesses. Out of the 33 teams that took part in the original study, 6 of them had entries to the original SemEval contest utilizing BERT, while GPT-2 and XLNet have not been evaluated for semantic change detection. The evaluation corpus is processed in two ways that can potentially affect the performance of models producing context-based embeddings, so it was also of interest to experiment if this was the case. Firstly, the target words to detect change on are all suffixed with their part-of-speech (POS)-tag. One experiment is based on the removal of this tag. Secondly, the corpus is lemmatized. As these models are pre-trained on raw text, using the original corpus might give more insight into their abilities to solve semantic change tasks.

The best performing model from the evaluation tasks was used to conduct a set of experiments to analyse real and fake news for their semantic differences. This also constitutes a difference between most work on semantic change that focuses on diachronic change, or changes over time, as this work is on synchronous data, where the changes are observed over a domain and the time frame is held constant. The model was fine-tuned on the NELA-GT-2019 (NELA) [8] news corpus, and the embeddings it produced was extracted and used to check the strength of two laws of semantic change, and to analyse words to examine their semantic changes. The two laws are the law of conformity and the law of innovation, which

state that frequent words are less likely to change, and that polysemous words are more likely to change.

1.4 results/summary

The results of the evaluation show that the three language models perform quite differently, with BERT outperforming the other two. All models are able to beat the baselines, with the frequency difference having a correlation of -0.22 and the count vector approach with a correlation of 0.02, but only BERT is able to get statistically significant results and get higher scores than previously used models. These findings for BERT are in line with the results obtained by other researchers. Fine-tuned BERT manages to achieve a correlation of 0.646 with human annotators on the graded semantic change task, meaning that the model largely agrees with people on which words are the ones with the most and least change. This score is substantially higher than those obtained by GPT-2 and XLNet, at 0.188 and 0.285 respectively. The reason is likely a congregation of all the differences between the models, rather than one specific difference. For the binary change task, none of the models perform exceptionally well with the methods used. The baseline that always predicts no change got an accuracy of 0.568, while the best results for BERT, GPT-2 and XLNet were 0.622, 0.649 and 0.622 respectively.

When exploring the effect of the pos-tag in the evaluation corpus, the results show that removing the tag yields a significant increase to BERT when pre-trained. The pre-trained model achieves a correlation of 0.547 and preforms better than previous models in the original challenge. After fine-tuning, removal still leads to increased scores for two out of the three metrics used to measure change, while there is a small decrease for the last. Using the raw corpus to fine-tune BERT also shows increased scores for two of the metrics, with a small decrease for the last.

The analysis of the two laws of change found a weak positive to no correlation for the law of conformity, indicating that the law does not hold, and a weak to moderate positive correlation for the law of innovation, indicating that it holds to some degree. Instead of supporting these laws, the results might on the contrary support research indicating that the laws were based on bias caused by the methods used to discover them.

Finally, the analysis uncovered words with two different types of observable semantic changes, namely broadening / narrowing, and sense shifts. This shows that the methods can help explain fake news, but it also uncovered areas that are important to improve on in further research.

1.5 Thesis outline

The remaining parts of the thesis is organized as follows: Chapter 2 presents background material that serves to give the required theoretical knowledge to understand the work presented. Chapter 3 gives a brief overlook on related work in the field of fake news research and semantic change. Chapter 4 presents the data used for evaluating and analysing, and discusses the datasets briefly. Chapter 5 gives a thorough explanation of the experiments and their methodology, while chapter 6 discusses their results. Chapter 7 concludes the thesis and states relevant further work.

Chapter 2

Background

This chapter contains the theoretical background material that is used by the experiments and analysis in this work, and that explain the project in more depth. The chapter starts with an overview of material related to fake news, and follows with language models, their embeddings and the transformer architecture. Then the three specific models used in this work are described, before the material related to lexical semantic change is covered. This includes a brief overview of the field itself and the methods of interest related to this work.

2.1 Fake news

The term fake news is rather vague, and is often used as an umbrella term for a multitude of related but still distinct problems of information. This is the case for both general text and scientific literature. It is often used to describe misinformation, information that is deliberately intended to mislead. Some sources also separate the term misinformation from disinformation, where misinformation is taken to mean false information, and disinformation means false information deliberately intended to mislead [9]. Satire and hyperpartisan news are included by some, but not others. With satire, the content is false, but the intent is not to mislead, but rather to entertain. For partisan news, the content may not be false, but the intent is not to give readers a truthful view of the information. The information is often portrayed in a fashion that validates the preexisting beliefs of either the creators or the readers. In this thesis a narrower definition of fake news is used, inline with [10] [11] [12] which define fake news as news articles that are intentionally and verifiably false, and could mislead readers.

People consume news on many different platforms, and the amount of fake news circulating on each one varies. [13] shows that the majority on news consumption by the US population comes from TV programs, and that this platform has no stations creating verifiable fake news in the same fashion that online platforms do.

They find that fake news account for less than one percent of daily news consumption for all platforms combined, but that the amount is higher for online news, especially from social media. Even though this number is low in itself, the impact it has can be much higher than for truthful news, especially combined with the low level of news consumption in general. The magnitude of the fake news problem is also exacerbated not only by people believing misinformation, but also by the concept itself being used by people who oppose the so-called *mainstream media* to sow distrust in prominent journalists and established news publishers.

Fake news research is a large field, and is generally explored from four main perspectives [14]. These are knowledge-, style-, propagation- and source-based methods. The first-mentioned aims to extract knowledge from the text of the articles and compare it with facts from knowledge bases, and is often deployed by journalists. Both the assessment sites mentioned in the introduction, and ClaimBuster mentioned more in section 3.1 fall under this category. Style-based approaches try to use the linguistic and semantic information inherent in the article. While knowledge-based approaches often try to assess the veracity of specific claims, style-based approaches more often look at properties that differentiate real and fake news. Propagation-based approaches look at how articles spread as an indication of fakeness. Instead of looking at the properties of the articles, one can look at the properties of the users who read such articles and how they interact with them. News flourishing on social media provide large networks of users spreading both real and fake news. The last approach overlaps with many of the previous approaches by looking at the credibility of the content, sources and users. The work conducted in this thesis is a combination of the style- and credibility-based approaches.

2.2 Fake news explanations

There are many approaches to generating explanations for fake news, depending on for whom they are meant and what purpose they serve. Finding changes in linguistic properties can be of great interest to other researchers. As an example, knowledge about which properties contribute the most and the least to making an article fake enables researchers to only include the most salient properties as dimensions in a feature vector used to detect fake articles. They can also be helpful to readers, giving them cues that might make them more able to discern fake articles.

Articles from news assessment sites like Politifact is a type of fake news explanation directed at the readers of news articles. These give a detailed explanation about the truthfulness of important segments of specific articles. The readers get a self-contained explanation of what is truthful and what is not regarding a contemporary topic or event. A backlog of annotated articles are also of interest to researchers creating automated approaches to these types of explanations, allowing them to employ machine learning techniques to train neural models, or simply

retrieve previously annotated statements as additional explanations for new fact checks.

Similarly to linguistic properties, semantic ones are of interest to researchers and can be useful as features for downstream tasks. [15] used semantic changes as features for the tasks of document classification and contrastive viewpoint summarization. They can also be of use to journalists at assessment sites in helping to visualize the differences. If an article contains a false claim, knowing how keywords in this claim differ in the two classes of news can be a helpful piece of information in an explanation.

2.3 Language models

A language model can in its simplest form be described as a probability distribution over sequences of tokens. Given a sequence of words, a language model tries to predict the most probable word to appear next in that sequence. Such a model is useful for solving many different tasks ranging from text generation, summarization or machine translation. For humans, completing a sentence like "*In the morning I drink ...*" is an easy task. It is intuitive that a word like coffee or water would be a good fit, while snow or car would not. The computer on the other hand, does not have the luxury of this intuition, and must rely on computing the probabilities for all possible sequences to give an answer.

The size of the corpus of sequences must be very large to facilitate good estimates of the probabilities, but even then, there are bound to be sequences that are not captured. The fact that a sequence has not been written before is not an indication that it will never be done in the future, and a language model should be able to model this. To overcome both the problem of unseen sequences and the computational complexity, models could approximate the sequences. This is the intuition behind one of the earliest class of language models, the N-gram models. Given a word w, the probability of w being the next word in a sequence s_1^k is approximated by the probability of w being the next word in s_{k-n}^k, the n last words of the sequence of length k. When n is one, the model is a unigram, when it is two, it is a bigram and so forth. The longer the sequence, the more context the model has when making its predictions. Increasing the size of the N-gram will yield results that are increasingly coherent. Unfortunately, with too high values of N the problems of sparse observations and complexity also returns.

2.3.1 vector semantics

The way N-grams represent words is merely a frequency distribution over a corpus. The words themselves do not carry any significance to the model, so it has no notion of the relationships between the words. It does not know that there is a similarity between words like coffee and water, or the sentiment of the words good or bad. To capture properties like these, words are represented as real val-

ued vectors. Using vector semantics, words can be embedded as a point in a vector space. Words with similar meanings will be closer to each other in the vector space than words that are not.

The earliest approach to vector semantics was the use of co-occurrence matrices to count how often words occurred as neighbors to each other. A word is then represented as a vector with the length of the vocabulary. The values are often weighted with methods such as tf-idf, where terms occurring often are given a higher value, but those occurring in many of the documents are penalised, as they are less discriminating. The resulting vectors suffer from the curse of dimensionality, as they are long and sparse. In the case of co-occurrence matrices, many words will never show up as neighbors of each other. Each such dimension contributes to bringing the words closer together in the vector space, lessening the effect of the actually present words.

Newer approaches use methods that achieve a much lower vector space dimensionality. One such model is skip-gram with negative sampling, often referred to as word2vec [16]. It is a neural network architecture that uses unsupervised learning on running text to train a classifier. The weights of the classifier corresponds to the word embeddings. The result of this is a set of vectors that are short and dense.

Using dense vectors as word representations allows language models to become much more efficient and produce better results compared to their sparse counterparts. This turned training language models based on neural networks into a reality. Recurrent Neural Network (RNN) models based on Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) became the new state of the art for most NLP tasks.

2.4 Transformer architecture

Many modern neural language models are based on the encoder-decoder architecture. As the name implies, this architecture consists of two distinct elements. The encoder takes an input sequence and produces a vector that embeds the context of this sequence. The context vector becomes the input to the decoder, which autoregressively computes an output sequence. The computation happening in both the encoder and decoder was originally done through RNNs, but due to its sequential nature this would lead to the context vector being more conditioned on the latter part of the input sequence than the former. This makes it more difficult for the model to make predictions with dependencies to the start of long sequences. To alleviate this, the attention mechanism which will be explained shortly was introduced. In 2017, Vaswani, et al. [17] showed that one could create encoder-decoder models based only on attention mechanisms. They called the resulting architecture the transformer. The transformer is the current state of the art in neural language models [17] [18] [19], and the basis for all the models used in

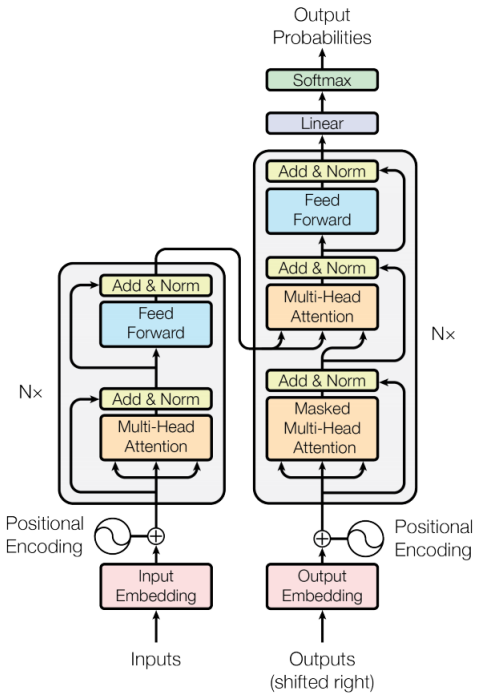


Figure 2.1: The transformer architecture. Source: [17]

this thesis. The architecture is visualized in figure 2.1.

2.4.1 Positional encoding

The first thing the transformer must do is to create a vector representation of all input tokens so that they can be used for computation. The weights are learned as the model is trained. The encoder and decoder shares the same weight matrix, as does the linear classifier at the end of the decoder stack, where the decoder output is turned into the token prediction. One thing that is missing from these vectors is the positional information about where in a sequence a token occurs, and the relations between the positions of all the other tokens. This information is included by the use of sine and cosine encodings that are added to the input embeddings.

2.4.2 Attention mechanism

The attention mechanism is the central part of the transformer architecture. It is what allows the network to maintain a far longer memory than recurrent neural network approaches like LSTM or GRU networks. This is achieved by the fact that the attention mechanism can use the whole sequence of input in the computation of an output token, and learn dependencies between all tokens in the

sequence. The input is transformed into three separate token-token matrices, Q, K, and V, using fully connected linear layers. Q can be thought of as a matrix of query vectors, and K and V as key-value pairs belonging to the output. A matrix multiplication, QK, produces a score matrix that is essentially taking the dot product of each query vector with each key. The scores represent how much attention each token has on every other token. To reduce the exploding gradient problem, the scores are downscaled by the square root of d_k, the dimensionality of the query and key vectors. After this, the scores are softmaxed to create a probability distribution between 0 and 1. The final scores now represent attention weights that can be multiplied with V, thus resulting in embeddings for the whole input sequence where each embedding is conditioned differently on all the other tokens in the sequence.

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

The sequential nature of recurrent neural networks is one of the main drawbacks that the attention mechanism helps alleviate. By splitting Q, K and V into h heads, and projecting each head into different vector spaces, the model is able to learn different features from each vector space in parallel.

The transformer contains three attention layers that each works slightly differently. In the encoder stack, the first input matrices come from the input sequence. The output this encoder produces becomes the input to the next encoder in the stack. In the decoder there are two attention layers. The first one takes the produced output token of the last run through the decoder stack as input and produces tokens in an auto-regressive fashion. During training, the whole output sequence is known at the start, so a matrix of all positional embeddings are given. To stop the decoder from conditioning on words that have not been generated yet, a mask is applied to the scaled attention scores before calculating the softmax. The result is a lower triangular matrix that allows for efficiently calculating the output sequence.

2.4.3 Residual layers

After each attention and feed forward layer, a residual layer adds the output embedding of the previous layer to the output of the current layer. This allows the input to flow through parts of the network without being transformed by the non-linear activation functions in the other layers. The intuition here is that the network does not need to focus on learning the representation of the input, and can thus focus on the difference, or residual, between the input and the final output. Residual layers also helps diminish the effect of the vanishing gradient problem. In deep neural networks this problem arises when training the network with gradient based learning and backpropagation. The gradient used to update the weights of the network become smaller and smaller the further it is propagated. The earlier layers of the network can effectively stop learning due to the small gradient.

Residual blocks helps alleviate this by allowing for larger gradients in the earlier layers. After the residual layer, the output is layer normalized, which helps reduce training time.

2.4.4 Feed forward networks

Feed forward networks are the most fundamental artificial neural networks. They consist of an input layer, N hidden layers, and an output layer. The layers are made up of fully connected processing units called neurons. The neurons have weights and biases that are trained to activate differing parts of the input, and the result of this activation is propagated further through the network.

$$a = f(Wx + b)$$

This equation shows the computation of one layer in a feed forward network. The input vector x is multiplied with a weight matrix W and summed with a vector of biases, b. The result goes through a non-linear activation function element-wise, and produces the input to the next layer.

In the transformer architecture, the goal of the feed forward layers is to transform the output of the attention layers into vectors that are better suited as input to the next layer in the model. Since the attention layers are multiheaded and the results of each of the h heads are concatenated to create the output, the vector essentially consists of h distinct sections that each has learned to attend to different features of the input. By running this vector through a feed forward network, the result is an embedding where the context is encoded in the whole vector, not in distinct parts of it.

2.5 Transformer-based language models

2.5.1 BERT

Bidirectional Encoder Representations from Transformers (BERT) is a pre-trained language model proposed by Devlin et al. [18] based on the transformer architecture. As mentioned previously, the transformer architecture is an encoder-decoder network. In contrast to this, BERT is only made up of encoder blocks. The authors proposed two models already pre-trained on large amounts of text that generalize exceedingly well to downstream tasks, needing only a small network that can be fine-tuned inexpensively. The models also allow for feature extraction, and thus produce word representations.

To tokenize text, BERT uses wordPiece tokenization [20]. With this scheme, the list of tokens is made up of a vocabulary of common words, but also includes sub-word tokens. This gives the model a way of handling words that are out of vocabulary. For instance the word *playing* could be split into the wordPiece tokens *play* and *##ing*. *##* denotes that the token is a continuation of the preceding

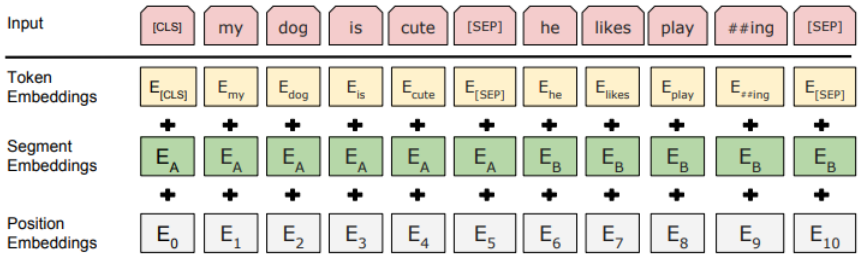


Figure 2.2: BERT embeddings. Source: [18]

token. BERT also makes use of two special tokens *[CLS]* and *[SEP]*. The former is a representation of the input sequence that is useful for classification tasks, while the latter is used to separate different sequences given to the model as input.

BERT is trained using two different semi-supervised tasks, masked language modelling and next sentence prediction. In masked language modelling, 15% of the wordPiece tokens in the corpus are substituted by a special *[MASK]* token, a random token or the same token can be kept. The *[MASK]* token is applied 80% of the time, while a random token or keeping the same one is chosen 10% respectively. The model must use the context provided by the surrounding sequence to predict the word most likely to be masked.

In next sentence prediction, the model is given two or more sequences separated by the *[SEP]* token and tasked with finding the likelihood that the latter sequence follows the former. Half of the sequences given to the model are ones that follow each other, while the rest are randomly chosen from the corpus. In addition to the positional encodings seen previously in the transformer, BERT also includes sequence encodings that help the model learn to distinguish sequences. The components that make up the final embeddings are shown in figure 2.2

2.5.2 GPT-2

Generative Pretrained Transformer-2 (GPT-2) [19] is a decoder-based language model, and as such it has the autoregressive property. This means that the model produces one token for each run, and that this token is appended to the input sequence when predicting the next output. Due to this, GPT-2 is exceedingly good at generating natural language text.

Since GPT-2 has no encoder blocks, the second attention layer is also removed. Recall from the transformer architecture that the second attention layer in the decoder combined the query and key matrices produced by the last layer of the encoder with the value matrix of the first decoder attention layer. The decoder blocks in GPT-2 are thus made up of a masked self-attention layer and a feed forward neural network.

To represent tokens, GPT-2 uses byte-pair encodings. It is originally a compression

algorithm, but is also useful in creating a vocabulary given a text corpus. At first, the vocabulary consists of all the symbols present in the corpus. The two tokens that combine to create the token with the highest frequency count is added to the vocabulary. This process is repeated until a given size of the vocabulary is reached. The token representations are learned using the training objective causal language modelling. Given a sequence of tokens, predict the most likely token to follow.

2.5.3 XLNet

XLNet is an autoregressive language model that aims to also incorporate the autoencoding property of encoder based models like BERT to be able to utilize bidirectional contexts. [21]. It is based on the Transformer-XL architecture [22].

The introduction of the original transformer architecture helped alleviate one of the large drawbacks with RNN based methods, namely its sequential nature hindering parallel execution. Their replacement with attention greatly reduces training time, but the $\mathcal{O}(n^2)$ time complexity still makes it infeasible to train on large contexts. Transformer-XL reintroduces recurrence, but on the sequence level instead of token level. The model keeps the hidden states from previous sequences and uses them to condition the current state, allowing dependencies between the sequences to be made and thus increases the effective context without increasing the max sequence length. For the model to be able to distinguish between the sequences, relative positioning is used instead of absolute. Recall that the original transformer architecture injected positional encodings into the token embedding at the start of each run through the model. Transformer-XL instead adds the relative position for each word dynamically during the attention calculation.

In addition to recurrence, XLNet also proposes a novel training objective called permutation language modelling. It is the usage of this objective that allows XLNet to incorporate bidirectional contexts while keeping the autoregressive property. In traditional autoregressive models, the objective has been to predict the i^{th} token x_i given the preceding tokens $x_{<i}$ in the sequence: $P(x_i) = P(x_i|x_{<i})$. The idea behind permutation language modelling is to sample a set of permuted orderings of the input sequence, and to autoregressively predict each token given the preceding tokens and their original relative position. As an example, a sequence with 4 tokens where [1, 2, 3, 4] denotes the indices of the tokens in their original positions can be permuted to [3, 4, 1, 2]. Given this permutation, the 3^{rd} token has no prior tokens to base its prediction on. The 4^{th} token can be predicted given the 3^{rd}, the 1^{st} given the 3^{rd} and 4^{th}, while the 2^{nd} has knowledge of all other tokens in the sequence. Instead of inputting the permuted sequences into the model, the attention mask filters out what tokens each token can use in its attention calculation. The mask for the given example is shown below.

$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

As can be seen from the empty diagonal, the token to be predicted can not attend to itself. While this is a necessity, it would be beneficial to include the positional information about the token in the prediction. This is achieved by the use of two-stream self attention. Instead of having one vector for each token that includes both content and positional information, these are separated into two vectors where one contains both content and positional information, while the other only contains the position.

2.6 Lexical semantic change detection

Lexical semantic change detection is the task of detecting words that change their meaning across time or domains. Given a corpus of text from the 1800s and one from the early 2000s, there will be many words present in both that have not changed syntactically, but that will have very different meanings in the texts they appear in. A word like *web* is likely not used to describe a digital network used for communication in an older text, while it is quite likely in a newer one. In this example, the word has gained a word sense, a new meaning. Gaining a sense, or a sense becoming much more prominent are two examples of a semantic change where the meaning of a word broadens. In contrast to this, word narrowing happens when a word loses a sense, or the sense becomes much less prominent. If a word starts to take on a completely other meaning than it has previously had, this is considered a semantic shift.

Systems used to computationally detect semantic change are generally comprised of three main components: a model used to create semantic word or sense representations, a vector space alignment technique, and a change detection metric. Creation of word representations relevant for this thesis is described in section 2.3.1, while alignment and change detection metrics are described below.

2.6.1 Vector space alignment

Due to the stochastic nature of neural language models, comparing embeddings created from distinct models becomes problematic. The embeddings are invariant under rotation, meaning their distances to each other are similar in both vector spaces, but their location in the space is not, barring comparison. To make the comparisons meaningful, several techniques have been used. One of the most prominent being Orthogonal Procrustes [23], which uses singular value decomposition to optimize a set of weights that are applied to one of the embedding sets.

Having to train several models and then aligning their vector spaces is a tedious process, and some methods aim to only train one model and forego the alignment step. This can be achieved by training a model iteratively on each time step or domain, and extracting the word representations before each consecutive iteration.

With the advent of contextualized language models, the possibility of extracting distinct usages based on the context of a sequence allowed for the creating of usage matrices for each time step or domain, and at the same time the model could be trained on all available data simultaneously. This latter method is the one used in this thesis.

2.6.2 Change detection

Using the language models, we are able to generate embeddings for each usage of each word in the corpora. By collecting all usages from each corpus into separate usage matrices, a word w from corpus one can be represented by its usage matrix U_w^{C1}, and the same word can be represented by its usages from corpus two by U_w^{C2}. These matrices becomes the basis for calculating the changes for each word.

The first two of the following methods were proposed by [24] and utilize the cosine distance to measure change. The last method is proposed by [25] and is a cluster based approach.

Cosine similarity

Cosine similarity is a measure of how similar two vectors are. It is defined as the cosine of the angle between two vectors in a shared vector space, and can be measured as the dot product of the normalized vectors.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{X} \cdot \mathbf{Y}}{\|\mathbf{X}\|\|\mathbf{Y}\|} = \frac{\sum_{i=1}^{n} X_i Y_i}{\sqrt{\sum_{i=1}^{n} X_i^2}\sqrt{\sum_{i=1}^{n} Y_i^2}}$$

Inverted cosine similarity over word prototypes

The first method used is the Inverted Cosine Similarity Over Word Prototypes (PRT). The change undergone by a word is found by taking the mean of the usage matrices to create one embedding representation for each corpus. The inverted cosine similarity between these two vectors is then used to calculate the final change value.

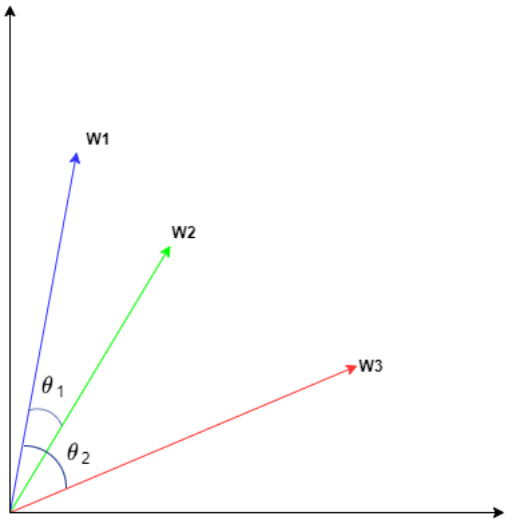


Figure 2.3: Three words in a vector space. The words W1 and W2 are closer together than W1 and W3, and thus have a smaller angle between them.

$$PRT(U_w^{C1}, U_w^{C2}) = \frac{1}{d(\frac{\Sigma_{x_i \in U_w^{C1}} x_i}{N_w^{C1}}, \frac{\Sigma_{x_j \in U_w^{C2}} x_j}{N_w^{C2}})}$$

d is the cosine similarity and N_w^{C1} and N_w^{C2} are the number of occurrences for word w in corpus *C1* and *C2*.

Average pairwise distance

The second method is the Average Pairwise Distance (APD). Change is measured by calculating the cosine distance between each pair of usage embeddings from the two corpora and taking the mean of these values as the final distance.

$$APD(U_w^{C1}, U_w^{C2}) = \frac{1}{N_w^{C1} \cdot N_w^{C2}} \sum_{x_i \in U_w^{C1}, x_j \in U_w^{C2}} d(x_i, x_j)$$

d is the cosine distance and N_w^{C1} and N_w^{C2} are the number of occurrences for word w in corpus *C1* and *C2*.

Clustered usage representations

The last change detection method relies on clustering the usage representations using K-means and calculating the Jensen-Shannon Distance to detect the frequency distribution divergence of cluster usages between the corpora. K-means and JSD are discussed in the two following sections.

2.6.3 K-means clustering

K-means is a clustering algorithm that aims to separate the points of a vector space into K distinct clusters by minimizing the intra-cluster sum of squares. The algorithm starts by randomly assigning K points in the vector space as centroids for the clusters. These points are not necessarily included in the dataset. Each data-point is then assigned to the cluster that has the closest centroid. When each data-point is assigned a cluster, the centroids are updated to the mean of the points currently part of the cluster. The latter two steps are repeated until no data-points change cluster, or the number of points is below a threshold. The intra-cluster sum of squares objective is given by:

$$\sum_{i=0}^{n} \underset{\mu_j \in C}{min}(||x_i - \mu_j||^2)$$

Where n is the number of data-points in a cluster and μ_j is the centroid for cluster j in the set of clusters C.

A more sophisticated method of choosing the location of the initial clusters than random selection is through the use of K-means++. Random selection is prone to selecting centroids that make the algorithm converge in a local minima. K-means++ helps overcoming this by choosing centroids by a probability proportional to the distance between the other centroids. The first point is chosen arbitrarily from the data-points, and the distance between this point an all other points are computed and used to choose the probability of any point being chosen as the next centroid. This process is repeated using the closest centroid to compute the distance with until K centroids are chosen.

2.6.4 Jensen-Shannon distance

The Jensen-Shannon distance is the square root of the Jensen-Shannon divergence, which is used to measure the divergence between two probability distributions for a variable, and is itself based on the Kullback-Leibler divergence.

$$KL(P||Q) = \sum_{x \in X} P(x) log(\frac{P(x)}{Q(x)})$$

P and Q are two probability distributions over the same variable, and X is the set of discrete values of the variable. If $P(x)$ is high while $Q(x)$ is low or vice versa, the divergence is high. Kullback-Leibler is an asymmetric measure and can not be used as a distance metric. It is asymmetric since $KL(P||Q) \neq KL(Q||P)$. For measures requiring symmetry, Jensen-Shannon distance is often used instead.

$$JSD(P,Q) = \sqrt{\frac{KL(P||M) + KL(Q||M)}{2}}$$

M is defined as the point-wise mean of the values in P and Q, $M = \frac{P+Q}{2}$. The Jensen-Shannon distance is thus given by the square root of the normalized Kullback-Leibler divergence of P from M, and Q from M.

2.6.5 Principal component analysis

Principal component analysis is a technique used to discover the axes that contain the most of the variance, called the principal components, in a dataset. These components are useful as a dimensionality reduction technique, mapping high dimensional vectors from the feature vector space to the potentially lower dimensional principal component space.

In a dataset where each observation consists of many different attributes, there is a possibility that some of the attributes are correlated, resulting in redundancy. Given an $m * n$ data-matrix X, where m is the number of observations and n is the number of attributes, the covariance between every attribute is given by the matrix multiplication of the transposed data-matrix by itself, $X^T X$. This matrix can be decomposed into matrices containing its eigenvectors and eigenvalues as column vectors, which represent the direction and importance of the axes that explain the most of the variance in the data. Since all eigenvectors are orthogonal, there is no correlation between them, and by multiplying X by the matrix of eigenvectors sorted by their eigenvalues, the data-points undergo a change of basis. In the principal component space, the latter attributes of the data-points correspond to dimensions containing diminishing amounts of information, and all data-points are combinations of the original data-points. By removing these latter attributes, the maximum amount of information can be kept while reducing the dimensionality of the data.

2.6.6 Evaluation metrics

Spearman's rank correlation coefficient

The Spearman's rank correlation coefficient, ρ, measures the size and direction of the relationship between the ranks of two lists of values. The Spearman correlation between two variables is high when the observations that achieve a high rank for one variable also achieve a high rank for the other, and low when the high-ranking observations receives low ranks for the other variable. The coefficient ranges in values from 1 to -1, where 1 is a perfect positive correlation, -1 a perfect negative correlation, and 0 means no correlation. Figure 2.4 plots an example of both a strong positive and negative correlation.

Accuracy

Accuracy is a measure that can be used to evaluate how many instances a classification method has correctly labeled out of all observations, and is given by the formula:

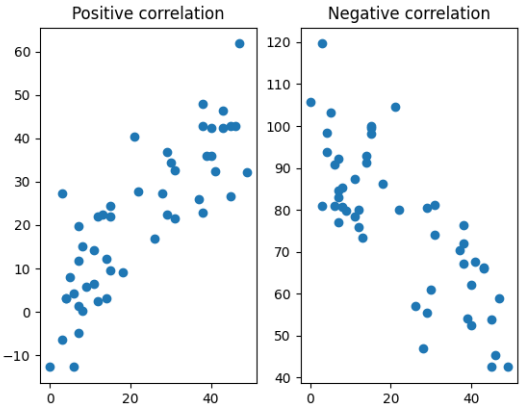


Figure 2.4: The plot on the left shows a positive correlation where $\rho = 0.83$. The right plot shows a negative correlation where $\rho = -0.87$.

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn}$$

Where *tp* and *tn* refer to the number of true positive and true negative classifications respectively, and *fp* and *fn* refer to the false positive and false negative classifications. This can be visualized in a confusion matrix. In a confusion matrix, as the one shown in table 2.1 below, the rows represent the predicted labels and the columns represent the ground truth label. The main diagonal thus shows how many correct classifications were made while the minor diagonal shows the erroneous predictions.

Table 2.1: An example confusion matrix with 15 correct classifications and 7 misclassifications, resulting in an accuracy of 0.68

		Ground truth	
		Change	No change
Predictions	Change	7	4
	No change	3	8

Chapter 3

Related work

Both the field of lexical semantic change and fake news research has seen a large increase in work done in the past few years. This chapter serves as an introduction to some of the most prominent papers and research projects in these disciplines that relate to the work in this thesis.

3.1 Fake news

As mentioned in the introduction, not much work has been done to explain what properties of fake and real news differ. This is especially so for semantic properties, where no papers applying lexical semantic change on fake news could be found. [6] does a thorough analysis of the differences of linguistic properties of news. They use three datasets containing real, fake and satire news. They found that fake news at the aggregate level is much less complex than real news, and is more similar to satire. Fake news tend to put much more information in the title of the article and use more proper nouns. The fake articles themselves are often short and have less punctuation and stop words. They also found that the most prominent features can be used to improve detection of fake news and satire.

ClaimBuster [26] [27] is an automated fact-checking tool that combines methods from machine learning, NLP and database querying to find and fact-check important claims from political discourse at an article or speech level. The system can be set to monitor TV programs and extract closed captions from them. These sentences are given a score on how likely they are to contain information worth fact-checking. The model is trained using supervised learning on text from previous general elections and is labeled by humans. The sentences worth checking are first compared to sentences already annotated by fact-checking sites. The system then formulates questions based on the claim and queries knowledge banks like wolfram alpha. It also sends the claim as a query to google, and aggregates matching claims with their surrounding sentences from the top results for extra data.

3.2 Language models

Since the introduction of the transformer architecture, new, pre-trained language models are emerging constantly, and furthering the state-of-the-art on many NLP tasks. Two advancements proposed to the original BERT model are found in Facebook's Robustly Optimized BERT Pretraining Approach (RoBERTa) [28] and Google's A Lite BERT (ALBERT) [29]. The researchers behind RoBERTa discovered that BERT was significantly undertrained, and by increasing the size of the pre-training dataset, training for more iterations, and only training the model on the masked language modeling task, they were able to greatly increase performance. ALBERT, on the other hand does not first and foremost aim to increase performance as compared to BERT, but rather to reduce the size and training time needed to achieve similar results. This was achieved by integrating factorized embedding parameterization and cross-layer parameter sharing. The resulting model has 18 times fewer parameters than the largest version of BERT, and can be trained 1.7 times faster.

In June 2020, OpenAi published a paper detailing a new language model, GPT-3 [30], with the aim of increasing few-shot learning performance, and thus reducing the need for labeled task specific datasets. It uses a similar architecture as GPT-2, but utilizes sparse attention. The number of parameters in the largest version of the model is increased from 1.5 billion in GPT-2 to 175 billion. Due to the sheer size of the model and the fact that the source code is not publicly available, applying this model is not a feasible approach for many researchers yet.

[31] proposes to treat all NLP problems as text-to-text problems solvable with one model, T5. The task to be completed is fed to the model together with the input text, and the model is trained to generate the correct output for the task.

ELECTRA [32] is an autoencoding model similar to BERT. The paper points out the inefficiencies of the masked language modeling objective BERT uses by only being able to learn from the 15% of the sequence that is masked out. They propose an objective where some tokens are replaces by a similar alternative, and the model is tasked with predicting if each token in the input was replaced or not. They find that the proposed training objective is more effective that masked language modeling, and that the model is able to create word representations that perform better on the GLUE benchmark than BERT, given the same amount of data, computational power and model size.

3.3 Lexical semantic change detection

In recent years, the field of Lexical Semantic Change (LSC) detection has seen a large increase in research activity. Many influential papers survey the field and tie up the current state of the art. [33] [34] [35]. They show a promising evolution of the field, but also that it is far from mature. Most of the work is conducted

on English, with few datasets available for other languages. There are also not a consensus on standardised datasets and tasks, making comparisons of different works challenging and imprecise.

3.3.1 Evaluation tasks and data

One of the open problems in researching LSC is having standardized tasks and evaluation data to compare different approaches with. In August 2020, the Semeval task on unsupervised lexical semantic change detection was published [7]. It contains diachronic datasets for English, German, Latin and Swedish annotated manually. The data relevant for this thesis is the English dataset, which is the Clean Corpus of Historical American English (CCOHA) [36]. The dataset is described in more detail in chapter 4.

The two subtasks in the Semeval challenge focus on finding semantic changes in two corpora from different time frames. The first task is binary change. If a word has gained or lost a sense between the corpora, classify it as changed. The other task is graded change. Rank all words by their amount of change. Words that have changed a lot receive a high rank, while the ones with the least change receive a low rank.

The top performing system on the graded subtask [37] utilize static word embeddings created from skip-gram with negative sampling [16]. Embeddings are created for each corpus independently and aligned using orthogonal Procrustes [23]. They use euclidean distance as a change metric. The team also submitted a model that utilized BERT to create the embeddings, but this model ranked 68^{th} out of the 186 systems submitted in total.

The systems most closely related to this work are the entries by the UiO-UvA team [24]. Focusing on the graded change subtask, they create contextualized word embeddings using two different language models, ELMo [38] and BERT. They use three change metrics: average pairwise cosine distance, inverted cosine similarity over word prototypes and the Jensen Shannon divergence on clusters made using affinity propagation. This thesis follows these change metrics, and they are described in greater detail in section 2.6. Their findings show that ELMo slightly outperform BERT, and that the metrics based on cosine distance performed better than the cluster based approach. They also noted that the performance of the cosine metrics correlate with the distribution of the score values for the different languages. Their contribution ranked 10^{th} in the contest, but in the post-evaluation phase they achieve higher scores than the original best performing entry.

DIACR-ITa [39] is a LSC detection challenge for diachronic Italian text. [40] utilized BERT embeddings and average pairwise cosine distance in their entry. They evaluated their model on the English dataset from the semeval-2020 challenge and receive good results, but find that these results do not carry over to Italian.

They hypothesize that the results might be affected by the quality of the underlying BERT model used for the two languages.

3.3.2 synchronous data

Most of the research on LSC is performed in a diachronic setting, while comparatively little work is done on synchronous data. [41] evaluate multiple approaches on both diachronic and synchronous data. The synchronous task is to discover meaning shifts from general to domain specific text and is evaluated using the German SURel dataset [42]. Their findings show that the evaluation metrics and methods applied to diachronic data are also applicable to synchronous data.

The work presented in [15] is closely related to the work in this thesis. They detect semantic shifts between two different viewpoints covering the same domain, political discourse. They use speeches from members of the British House of Commons to construct two semantic spaces, one from members of the Conservative party and the other from members of the Labour party. The speeches are all from the Thatcher era (1979 - 1990). A set of 24 target words were created to use for evaluation. The words that were chosen were all central to many controversies of that time. Target words were annotated by political scientists who were given a target word and two lists of related terms of the target word. One list was created from usages from the Conservative party and the other from the Labour party. The annotators were asked to identify which list belonged to which party.

The vectors in the semantic space are created using Word2vec. To detect changes they experiment with three measures: The first is to align the semantic spaces by projecting words from one space to the other, and then measure the distance between the same words. The second approach uses a graph where words are nodes and the edges between them are similarities. They then measure the similarity between neighbors. The final approach is a combination of these two.

Their findings show that their proposed methods are able to find semantic changes between viewpoints, even for words that are stable over time, and that the laws of semantic change proposed by [23] hold for synchronous data. These laws state that frequent words are less likely to change, and that polysemous words are more likely to change.

A literature review could not uncover any works where LSC has been used in an effort to explain fake news. The closest work was the aforementioned paper on changes in political viewpoints. The similarities between changes in political viewpoints and news are that they both only cover a specific domain in the two corpora that are compared, and that this domain is overlapping, as news contains a lot of political discourse. Even with this overlap, an important difference is found in the formality of the language used. As text collected from speeches is much less structured than an article covering the same speech.

3.3.3 Word sense induction

Word Sense Induction (WSI) is the problem of clustering usages of a word according to the meaning inherent in each usage of the word, and these clusters can be used to detect LSC. As mentioned previously, [24] uses affinity propagation, which is a message passing based clustering algorithm. With this algorithm, one does not need to specify the number of clusters beforehand, which is a benefit in WSI, as the number of senses for a word can not be known at the time of the clustering. Their approach is based on [25], but this paper utilizes K-means to cluster. As K-means require the number of clusters to be known, they conduct multiple runs of the clustering and choose the run with the number of clusters that result in the best silhouette score.

3.4 Datasets

The main datasets used in this thesis will be presented in chapter 4, but there are more datasets that can be valuable towards explaining fake news.

The ClaimBuster dataset [43] contains 23 533 human annotated statements from US presidential debates. The statements are annotated based on how likely they are to be check-worthy, factual but unimportant or non-factual. This dataset can be helpful for systems that use snippets of text from the articles in their explanations.

CREDBANK [44] and Some Like It Hoax [45] are two engagement driven datasets that contain text from tweets and Facebook posts respectively. CREDBANK contains more than 60 million tweets annotated by their credibility. Some Like It Hoax comprises 15 500 Facebook posts and likes from over 900 000 users, and aims to identify misinformation based on the users who like posts from scientific or fake scientific sources. Engagement driven data from social media outlets allows the includion of user information and network data, which can be valuable explanation tools.

NELA-gt-2018 [46] is the predecessor to the dataset used in this thesis. It is made up of over 700 000 news articles from 194 sources. The articles are sources between February and November of 2018, and they are annotated by the veracity of their source outlet using labels from 8 different reliability assessment sites.

Chapter 4

Data

This thesis used data from two different datasets, which are both described in this chapter. The first dataset is used to conduct a quantitative evaluation of the methods used in this thesis, while the second is used to make a qualitative analysis of news data.

4.1 Clean Corpus of Historical American English (CCOHA)

CCOHA [36] is a processed version of the Corpus of Historical American English [47]. The original corpus is made up of 400 million words from over 100 000 texts written in the period between 1810s to the 2000s. For each decade, the texts are balanced by genre, sub-genre and domains. The genres range from newspapers, magazines, fiction and non-fiction books. The text is available both as a lemmatized and part-of-speech tagged version, and as raw text.

4.1.1 Shorcomings of the original dataset

Due to copyright issues, 10 tokens are replaced by the '@' token every 200 tokens in every text. This introduces a number of problems. The first is the loss of 5% of the corpus, but it also reduces the quality of the context around a word. For tasks where one is dependent on the collocates of a target word, a sequence of '@' tokens are likely to impair the results. One also introduces the possibility of losing the punctuation that indicates the end of a sentence, resulting in new sentences that are likely semantically invalid.

The second limitation is the malformed tokens included in the corpus. Some tokens are not separated properly, or they can be artefacts from the process of gathering the text. These tokens then result in malformed or inconsistent lemmas and part-of-speech tags. Lastly, the corpus also includes HTML Characters not present in the original text.

As a result of the cleaning, the new corpus has increased in number of tokens by over 25 million, but reduced the number of non-word- and invalid tokens. The number of different lemma types is increased by nearly three times.

4.1.2 Semeval challenge

CCOHA is the dataset included as the English corpus in the semeval challenge. The organizers split the corpus into two diachronic corpora, one with text ranging from 1810-1860, referred to as C1, and the other with text from 1960-2010, referred to as C2. They then annotated a list of 37 target words by first collecting a list of 100-200 words with meaning change by searching through historical dictionaries. The words from this list was then filtered by collecting 50 sequences containing each word from both corpora, and asserting that the changes was present in the corpora. For stable words, the same procedure was taken, but the words were also filtered to make sure that the POS-tags and frequencies of these words were balanced with the changing words.

To create the labels for the binary task, the words were simply assigned a 1 or 0 during the collection of the target words. 0 represents stable words and 1 represents unstable words. For the graded change task, the target words were annotated using an extended version of the DURel framework [48]. Native speakers were asked to judge how related two usage pairs from different time frames are to each other on a scale of 1 to 4.

4.1.3 sequence properties

There are a total of 507 336 sequences in the whole dataset, where 253 644 are from C1 and 253 692 from C2. Not all of these include usages of any of the target words, however. In C1, there are 25 955 sequences with at least one target word, and in C2 there are 30 059. The average sequence length, standard deviation, min and max lengths for C1 and C2 are shown in table 4.1 and how the length increases for the whole dataset is visualized in figure 4.1. The length is given by the number of tokens. In the setting of contextualized language models, these sequences are considered short. Sequence length is an important property for such models, as shorter sequences give less context and might be disparate from the pre-training data.

Table 4.1: Statistics on the sequence length in the CCOHA dataset.

	C1	C2
Average	37.1	24.7
Standard deviance	23.6	14.9
Min	5	5
Max	186	182

Figure 4.1: The number of tokens in the sequences in the CCOHA dataset.

The average frequency count of the target words is 701 and 812 in C1 and C2 respectively, with a standard deviance of 1181 and 1024. As shown in figure 4.2 there are six target words with much higher frequency counts than the rest. The word *head* has the highest combined frequency, with 7582 usages, while *stab* has the lowest with 208.

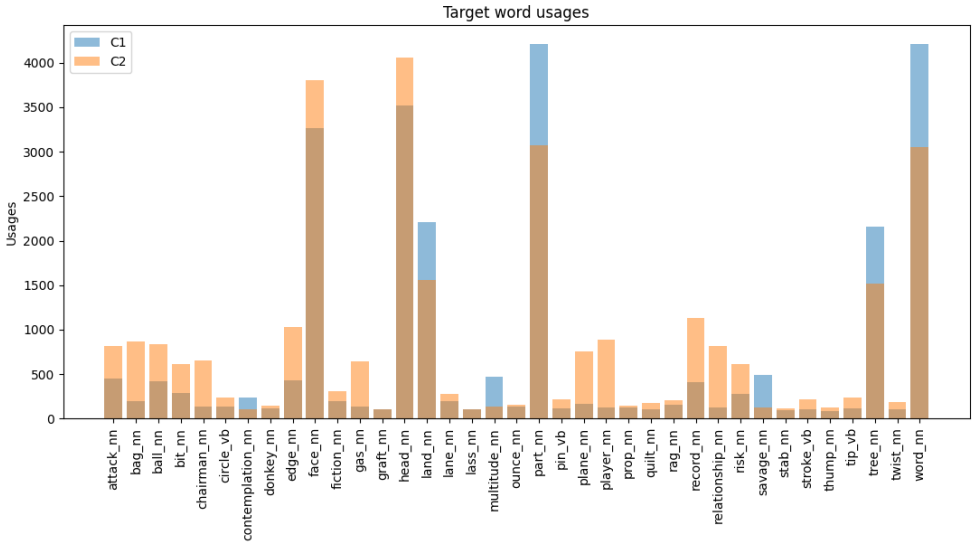


Figure 4.2: The frequency of each target word in C1 and C2.

4.2 NELA-GT-2019

NELA-GT-2019 [8] is a dataset of news articles from 260 news outlets collected over the year 2019. It is the successor of the NELA-GT-2018 dataset [46], with

a new time-frame, more data and updated ground-truth labels. The new dataset contains over 1.12 million articles from reliable, unreliable, and mixed sources.

4.2.1 Data format

The dataset was released in two formats; an SQLite database and as a set of JSON-files. This thesis uses the JSON-files. Each file correspond to a source and consists of a list of objects that represent an article. The properties of each article is shown in table 4.2. All articles are written in English, with an exception being the articles from the source *Der Spiegel* which are all in German. [8] states that sources might be based in other countries, but that all articles are written in English, so this might be an erroneous inclusion.

Table 4.2: Properties of an article in the NELA-GT-2019 dataset.

Property	**Description**
id	An id unique to each article
date	Publication date in YYYY-MM-DD format
source	Name of the source the article is from
title	Title of the article
content	Body text of the article
author	Name of article writer
url	Url linking to the article
published	Publication time as formatted by the source
published_utc	Publication time in utc as unix time stamp
collection_utc	Collection time in utc as unix time stamp

The ground-truth labels are included as a CSV-file. The labels are based on attributes of the source, meaning that all articles from one source are considered to have the same label. Veracity attributes are collected from seven different assessment cites, and an aggregated reliability label is created based on these. Not all sources have labels from every assessment cite, and 55 of the sources have no labels at all. Sources marked as satire have no aggregated label.

All reliable news sources are prone to unintentionally publishing misinformation, and not all articles published from an unreliable source necessarily contain false information. For the former case, the reliability of the sources the publisher itself operates with on a story might not be reliable, or new information that invalidates previous claims might come to light. For the latter case, publishing some truthful articles can put the source under the guise as reliable, and make readers more likely to believe the false articles as well. This opens up the possibility that not all articles in the dataset used in this work have content that match its reliability label. Since the labels themselves are based on the content created by the sources, the occurrence of such articles is assumed to be low. The work conducted in this thesis is also operating on the aggregate level rather than on an article to article

Table 4.3: Properties of sequences per target word

	C1	C2
Average	3079.3	3226.8
Standard deviance	1502.5	1495.9
Min	1000	1000
25^{th} percentile	1645.5	1779
50^{th} percentile	2782	3016
75^{th} percentile	5000	5000
Max	5000	5000

Table 4.4: Properties of sequence length

	C1	C2
Average	27.4	32.7
Standard deviance	14.9	25.7
Min	4	4
25^{th} percentile	18	19
50^{th} percentile	26	28
75^{th} percentile	34	39
Max	1309	1309

basis, so this is deemed an acceptable artifact of the dataset. As a further measure, articles labeled as mixed veracity are not included in the used data.

4.2.2 Properties

82 of the sources are labeled as reliable, 50 as unreliable, 50 as mixed, and the remaining 78 sources have no aggregated label. The dataset is very unbalanced, as over 40% of the data comes from reliable articles while only around 12% are from unreliable ones. Due to this and the sheer size of the dataset, this thesis does not use all the available data. The details on the process of downsampling are described in section 5.3.1, and the process of obtaining the vocabulary of target words in section 5.3.2. The rest of the present section is concerned with the data after processing.

Similarly to section 4.1, the corpus of reliable sequences will be referred to as C1, and the corpus of unreliable sequences as C2. Table 4.3 shows some statistical properties of the number of usages for the different target words, and image 4.3 shows how the sequence lengths increase. While most of the sequences are short, there are also some very long ones. When comparing to the CCOHA dataset, the target words all have a much higher number of usages each, as is to be expected since the downsampled dataset is over 12 times larger, and the target words were chosen specifically to have at least 1000 usages in both C1 and C2. From table 4.4 we can also see that the average length of the sequences is comparable to CCOHA, even with the few long sequences.

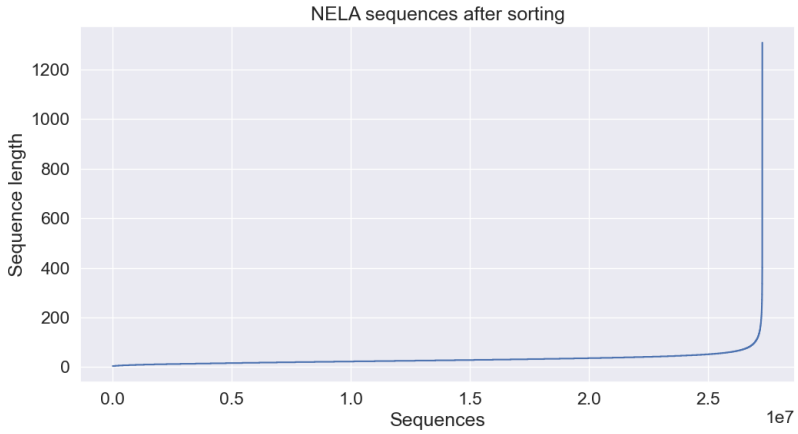


Figure 4.3: The number of tokens in the sequences in the NELA-GT-2019 dataset.

Chapter 5

Method

This chapter gives an overview of the methods used to conduct the experiments and analysis in this work. The chapter is divided into three main sections. The first is an outline of the tools used for the implementation, the second is concerned with the methods used to answer the two first research questions on evaluating contextual language models for semantic change, while the third is concerned with the the two last questions on semantic change in news data.

5.1 Libraries

Python has been the programming language used to implement all parts of the experiments. It is a language with a large collection of robust data science libraries. The ones used in this work are listed below.

Numpy[1]	Numpy is a library that simplifies working with multi-dimensional arrays and matrices. It adds performant data structures and multiple methods to conduct mathematical operations on them.
Scikit-learn[2]	Scikit-learn is a library that implements many prominent machine learning and data analysis algorithms for processing of data, experimenting and evaluation.
nltk[3]	nltk, or the natural language toolkit is a library for processing text in natural languages, like English or Norwegian. It includes many datasets and tools for tokenization, stemming and many other often used text processing steps.
Scipy[4]	Scipy is a library for scientific and mathematical computation in python. Similarly to scikit-learn it contains methods useful for data analysis.
Pytorch[5]	Pytorch is a deep learning library for creating neural networks and performing tensor operations utilizing GPUs.
Huggingface Transformers[6]	Huggingface Transformers contains pytorch implementations of many popular transformer-based neural network architectures.
Huggingface Datasets[7]	Huggingface Datasets contains tools to process datasets to the format suitable for Huggingface transformer models.
matplotlib[8]	Matplotlib is a library for creating plots and visualizations.
seaborn[9]	Seaborn is a visualization library that extends the capabilites of matplotlib.

[1] `https://numpy.org/`
[2] `https://scikit-learn.org/`
[3] `https://www.nltk.org/`
[4] `https://www.scipy.org/`
[5] `https://pytorch.org/`
[6] `https://huggingface.co/transformers/`
[7] `https://huggingface.co/docs/datasets/`
[8] `https://matplotlib.org/`
[9] `https://seaborn.pydata.org/`

5.2 Evaluation of contextual language models

5.2.1 Data processing

The first step in conducting these experiments was to convert the text data to the format required by the models for fine-tuning and feature extraction. Huggingface Transformer models require input to be formatted in a very specific way, and thus offer tokenizers specific to each model. These tokenizers take a text sequence or a batch of sequences as input and produce the inputs expected by the respective model. The first step in this process is to load the data into a Huggingface Datasets *Dataset* object. This requires the input to be formatted as a line-by-line text file, i.e that each sequence has its own line in the text file, which was the provided format of the CCOHA dataset. The next step is the actual tokenization of the sequence. The sequence is split up into a list of the tokens known to the model, which are then converted into integers representing the tokens.

Each model has a max sequence length which define the length of the input it can process. If sequences are too long they get truncated, and if they have differing lengths, padding is applied. To make sure that padding tokens are not included in the attention calculation, the tokenizer also produces a binary attention mask with zeroes for the padding and ones for the input. The last output of the tokenizer is a list with positional information called token type ids. If an input sequence contains multiple separate sentences, each token from one sentence will have the same token type id, which is different from the other sentence. For these experiments, the input was batched and padded to the length of the longest sequence in the batch. Since no sequences were longer than the maximum sequence length of the model, no truncation was performed.

For the unlemmatized version of the corpus, some additional processing was necessary. This is due to the fact that the target words were suffixed with a part-of-speech tag that is not present in this corpus, and that the word-forms the target words appear as in the unlemmatized sequences are recognized as distinct tokens by the models. For example, the word-form used in the first sequence for the target word attack_nn is attacked. The part-of-speech tag was thus simply removed from each word. The second solution was to lemmatize all occurrences of the target word while keeping all other tokens in their original form. This was done using the *WordNetLemmatizer* from nltk. The words were also tagged with their part-of-speech tag to make the lemmatizer able to correctly convert all word-forms, but the tag was not kept in the final sequences.

5.2.2 Fine-tuning

Most of the language models published now are pre-trained models. The authors who create them train them on large amounts of data, and release them for scientists to use. The time and computational power needed to pre-train these models is prohibitively large for anyone except large firms and research labs, but fine-

tuning these models afterwards on a smaller dataset is a much more feasible task. The models can either be used directly as they are, but fine-tuning them allows the models to adapt to the domain specific data and task they are to be used for.

The goal of all three models in this work is language modelling, and are thus fine-tuned on the same objective as when they were pre-trained. One exception is for BERT which was pre-trained on both the MLM and NSP tasks, but is only fine-tuned on MLM. This is because the latter objective is shown to not have much impact on the performance of the model [28]. Huggingface Transformers offers several models depending on the training objective. These models can also be instantiated with different pre-trained weights depending on the size of the underlying network and the amount of training data used. The ones used in this work are shown in table 5.1. Note that BERT is pre-trained using uncased text, while the text for GPT-2 and XLNet is cased. Uncased text is preferable since the case of the words are removed in the feature extraction, so all target words are treated equally. Versions of GPT-2 and XLNet trained on uncased text is not available.

Table 5.1: shows the pre-trained models and the Huggingface initialization parameters used for each of them.

Model	Class	Weights
BERT	BertForMaskedLm	bert-base-uncased
GPT 2	GPT2LMHeadModel	gpt2
XLNet	XLNetLMHeadModel	xlnet-base-cased

Most hyperparameter choices were left at default values, except from some changes that better suit fine-tuning, and to take computational limitations into consideration. The changes done was to reduce the learning rate to $2e-5$, include 500 warmup steps at the start of the tuning, and for BERT, include a weight decay of 0.01. The learning rate is reduced so as to not skew the weights too far from their pre-trained values. The learning rate used was proposed as a good choice for fine-tuning by the original BERT authors [18]. The warmup steps are included to reduce the learning rate for the first few sequences the model sees. This is because the data used for the fine-tuning likely differs a lot from the one used for pre-training, and to make the model not update its weights as much in this transition period. The weight decay included for BERT reduces the chance of overfitting the model.

In addition to this, experimentation was done on the combination of maximum sequence length and batch size, which showed that a sequence length of 128 and a batch size of 32 performed best on the semantic change evaluation given the available memory during this work.

The huggingface transformer library also offers a set of data collators to process the model input for its specific learning objective. In this work, the *DataColla-*

torForLanguageModelling was used to mask out 15% of the token ids before they were given to BERT. The *DataCollatorForPermutationLanguageModelling* was used to permute the attention mask for XLNet.

5.2.3 Feature extraction

Word representations can be extracted using the fine-tuned models. At each layer in the models, different embeddings of all input tokens are created. The lower layers often capture linguistic properties, while the last layers capture semantics. Often used extraction techniques are to take the last layer, or to average or concatenate all or the four last layers. In this work, the last layer is used. The embeddings for the input tokens that represent the target word is then retrieved from this layer. A single target word can be represented by multiple tokens if this word is not known to the model. If this is the case, the embeddings are averaged together. All embeddings have 768 dimensions.

While the model is fine-tuned on the whole dataset, not all of the sequences contain any of the target words. These sequences were therefore not used during feature extraction. The maximum sequence length was also set back to the default of 512, since the memory requirement for feature extraction is less than fine-tuning due to the model not having to store weight gradients. The final result is two numpy matrices for each target word, that contains all usage representations from the corpus of text from the 1800s and 1900s respectively.

5.2.4 change detection

The theory behind the methods discussed in this section is presented in section 2.6.2. The current section discusses their implementation details.

For the graded semantic change detection, all calculations are done using the usage matrices for a word as input. To calculate the PRT score, the matrices are averaged to produce one 768-dimensional vector each. Then, the cosine similarity of these two vectors is found using the cosine similarity implementation from scikit-learn. The resulting score is then the inverse of this similarity value. For APD, scipy is used to calculate the cosine distance between all embeddings, and these values are then averaged together.

To produce the K-means clusters, the two matrices are combined. This is to ensure that the clusters obtained for one corpus are the same as the one obtained for the other. If this is not the case, the clusters are not comparable. The combined usage matrix is standardized to remove the average and scale each value to unit variance using the *StandardScaler* from scikit-learn. The usage matrix is then clustered nine times, with the number of clusters ranging from two to ten, to discover the number of clusters that result in the lowest silhouette score. Both the clustering and silhouette score implementations come from scikit-learn. The clustering results in a label for each word usage denoting which cluster it belongs

to. This information is used to create a frequency distribution for how many times each word was used in the different clusters for each corpus. In other words, how frequently a word was used in a specific sense in both the older and newer texts. The Jensen-Shannon distance implementation from scipy is used on the frequency distributions to create the final change score. An approach to clustering using affinity propagation was also experimented with, but due to the many hyperparameter choices the number of clusters for a word could vary widely, and the results were not performant.

For the binary change detection, three methods were used. The two first were to use the average change values from APD and PRT respectively as a threshold. Words with a change score under the threshold were classified as not changed, while those on or above were classified as changed. The third approach utilizes the frequency distributions from the clustering and two thresholds k and n. One high and one low. If the frequency of one sense is less than k in one corpus and higher than n in the other, this is regarded as losing or gaining a word sense, and the word is classified as changed.

5.2.5 POS-tag removal

In the original semeval challenge, all top performing models utilized static embeddings, while contextualized models generally performed worse. The experiment described in this section and the next therefore aim to examine the effect on the performance of contextualized models based on processing of the dataset. This experiment is concerned with the effect of the inclusion of a POS tag in the vocabulary of target words and the dataset itself. For both the pre-trained and fine-tuned BERT models, the following configurations were tried:

- Include all POS tags.
- Remove POS tags from target word vocabulary
- Revome POS tags from both vocabulary and dataset

5.2.6 Graded semantic change on raw text corpus

Due to the fact that contextualized language models are pre-trained on text that is minimally processed, this experiment is concerned with the effect of fine-tuning and extracting features from a version of the text that is not lemmatized. Two different approaches are used:

- Fine-tuning on the completely raw text
- Fine-tuning on the raw text with target words lemmatized

5.2.7 Evaluation

Two metrics are used to evaluate the change scores. Spearman's rank correlation is used on the graded change scores obtained in the graded change detection task

for both the lemmatized and raw corpus, and for the POS-tag removal scores, while accuracy is used on the binary change detection scores. The Spearman's rank correlation is implemented using scipy.

5.3 Analysis of news data

5.3.1 Data processing

As mentioned is chapter 4, the NELA dataset was formatted as 260 JSON files - one for each source, and a CSV-file with labels for each source. one of these sources, Der Spiegel, contains German text data which is of no use for these experiments, and was subsequently removed. The remaining files where then loaded using the provided source code[10] and converted into two line-by-line text files based on their aggregated reliability label. The JSON files contains a list of each article gathered for that source, with their content and some metadata. All text in an article is stored as one string, and had to be split into sequences. This was done by first splitting the articles into paragraphs and then using the *sent_tokenizer* function from nltk to produce the standalone sequences. This tokenizer is an unsupervised model trained on english text data, and might thus make mistakes or handle ambiguous sentences differently than the author intended. A sequence extracted from the text might therefore be shorter or longer than the original corresponding sentence from the text.

Due to the exceedingly large size of the dataset, its size had to be reduced to make fine-tuning and feature extraction feasible for the scope of this thesis. The whole dataset consists of 33.1 million sequences, which would take roughly seven days of continuous computation only to fine-tune. The processing steps conducted were as follows:

1. Remove all sequences from sources labeled as mixed veracity or that are unlabeled.
2. Remove sequences of three or less words.
3. Downsample reliable sequences until there are equally many reliable and unreliable sequences.

Even though mixed veracity and unlabeled sequences could have been used to train the model during fine-tuning, they would be more troublesome to incorporate during feature extraction. More experimentation would have to be conducted to assert whether they provide a beneficial effect for the experiments in this work, which is one of the reasons why they have been ruled out altogether. Short sequences are removed as they contain very little context around the target words, so longer sequences are preferred. At last, there is a large imbalance between the number of reliable and unreliable sequences, as one would expect to be the case in

[10] `https://github.com/mgruppi/nela-gt-2019`

real life as well. The number of unreliable sequences is still very substantial, and by randomly downsampling the reliable sequences to this size, the final dataset becomes aptly sized for this work. The resulting number of sequences can be seen in table 5.2.

Table 5.2: The number of sequences for both reliable and unreliable news after reducing the size of the dataset.

	Original number	Remove short sequences	Downsampled
Reliable sequences	11 958 999	11 333 167	2 894 822
Unreliable sequences	3 030 559	2 894 822	2 894 822

5.3.2 Target word selection

A vocabulary of 4327 words was created to be used as target words for the analysis, as using all words encountered would not be feasible. Doing so would include many words that are not likely to be of great interest in explaining news veracity. An example of such words are stop words, words that do not carry much factual information, but are present in most sentences. There would also be many words that are used so infrequently that there are insufficient usages to create trustworthy change scores for them.

The vocabulary was created by using the *freqDist* object from nltk to count the occurrences of each word. Before any processing there were 789 393 distinct words over both corpora. This number is far larger than the actual number of proper distinct words in the text, but contains a lot of misspellings, errors from the tokenization process, and strings of punctuation. What constitutes a distinct word is also not definite. Differing word forms, like *said* or *saying* both stem from the word *say* and can either be counted separately or as two instances of the base word. In this work, differing word forms are counted as separate distinct words. The first step to reduce this number was stop word removal. Stop words were removed using the stop word collection from nltk, pythons inbuilt list of punctuation, and a self defined list of punctuation, numbers and single letter words that occurred often in the text. This removed a total of 221 words. Words with few usages were filtered out, where a threshold of 1000 usages in each corpus was chosen as the minimum usages needed. Including words with fewer usages could lead to unreliable results for these words while also requiring more more time to run the feature extraction and clustering operations. This step removed a substantial amount of words, resulting in a list of 4327 words. Since the most frequently used words have especially many occurrences, a higher threshold of 5000 was chosen to reduce the computational requirement further. For the words with usages over the threshold, 5000 sequences were randomly sampled for the feature extraction and clustering operations.

5.3.3 Fine-tuning

Fine-tuning on the NELA dataset was done similarly to the above mentioned dataset, and with the same hyperparameters. The combination of reliable and unreliable data was used. The computations were run on the IDUN GPU cluster for high performance computing.

5.3.4 Feature extraction

The feature extraction was done similarly to the above mentioned dataset but with one exception. In this dataset, some of the sequences are above the maximum sequence length of the model. For these words, a context window of length 512 was used to truncate the sequence around the target word. Depending on the placement of the target word in the sequence, the truncated sequence will retain as much context on both sides as possible.

5.3.5 change detection

For change detection, APD and Jensen-Shannon distance on K-means clusters was used. An overview of the whole linguistic pipeline is shown in figure 5.1

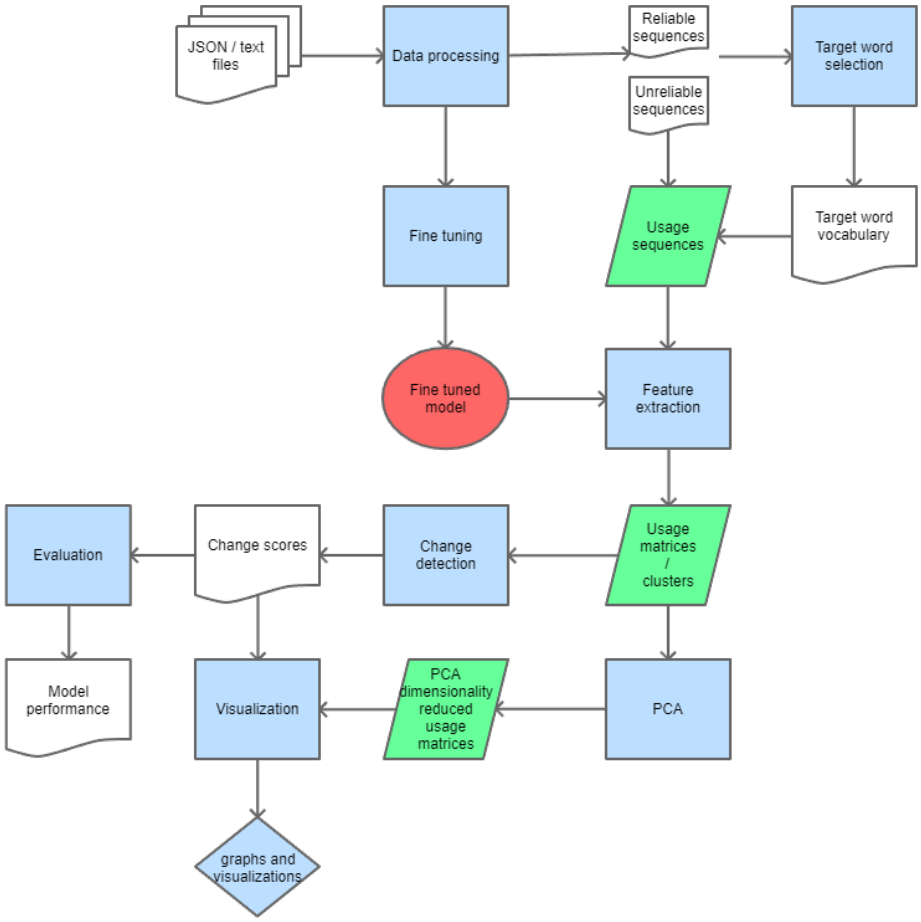


Figure 5.1: A visualization of the whole linguistic pipeline.

Chapter 6

Results and Discussion

This chapter presents the results of the experiments and analysis conducted in this thesis, as described in chapter 5. Similarly to the aforementioned chapter, this chapter is split into two main parts, where the first is concerned with the evaluation of the language models, and the second about the analysis of news data. Each of these sections are again divided into subsections for each experiment that contain the results obtained, and a discussion and comparisons to relevant literature.

6.1 Evaluation of the models

This section presents the results of the experiments that aim to answer the two first research questions.

6.1.1 Baselines

A baseline is an effective tool to compare a proposed method against standardized or much less complex approaches to see if the new method performs better, or if its complexity is justifiable. The authors of [7] propose three baselines for the binary change detection task and two for the graded task. The graded change baselines are the frequency difference and count vectors with cosine distance. The binary task also includes a majority class prediction which always predicts no change.

The frequency difference baseline is found by calculating the frequency of each target word in the two corpora, and dividing by the total frequency and using the absolute value as a change score. The count vector baseline learns vector representations for the corpora and aligns them. The cosine distance is then used to create the final change score. The scores for these methods for the English corpus are shown in table 6.1

Table 6.1: The baseline scores reported in [7]. The graded scores use Spearman's rank correlation and the binary scores use accuracy.

Method	Graded change	Binary change
Frequency	-0.217	0.432
Count	0.022	0.595
Majority	-	0.568

While the evaluation is done using diachronic data, the data in the analysis is synchronous. Even with this discrepancy, the tasks solved are similar, and as stated in 3.3.2, previous research has showed that methods proposed on diachronic data are also applicable in a synchronous setting. [41]

6.1.2 Graded lexical semantic change

Table 6.2: The Spearman's rank correlation of the three pre-trained models. Bold numbers represent the best score for each model. None of the results are statistically significant.

	APD	PRT	JSD
BERT	**0.288**	0.105	-0.042
GPT-2	0.094	0.224	**0.254**
XLNet	-0.205	0.081	**0.292**

Table 6.2 shows the results of the three models on each of the three change detection metrics before any fine-tuning was conducted. We can see that the scores each model obtains for the metrics varies quite a lot, and that the performance of the models also vary a lot when looking at the same metric. BERT is able to achieve a weak correlation when combined with APD, but close to no correlation with the other metrics. GPT-2 shows almost no correlation using APD, but a weak correlation for PRT and JSD. XLNet obtains the highest score over all model-metric combinations when combined with JSD, but also the lowest score by a large margin when combined with APD. This score indicates a weak negative correlation, or in other words that the model is more likely to grade words with little real change as those with the most change.

Table 6.3: The Spearman's rank correlation of the three fine-tuned models. Bold numbers represent the best score for each model. The results marked with * are statistically significant.

	APD	PRT	JSD
BERT	**0.646***	0.171	0.35*
GPT-2	0.075	0.05	**0.188**
XLNet	0.259	0.174	**0.285**

Table 6.3 shows the graded semantic change scores for the three models after fine-

tuning. BERT has shown a large increase in correlation for both APD and JSD, and a slight increase for PRT. Both the APD and JSD scores are statistically significant ($p < 0.05$). Surprisingly, GPT-2 has shown a decrease for all metrics after fine-tuning. The decrease is most notable for PRT. XLNet improves substantially from a weak negative to a weak positive correlation for APD, but shows a slight decrease for JSD. How the models and metrics compare can be visualized in figures 6.1 and 6.2.

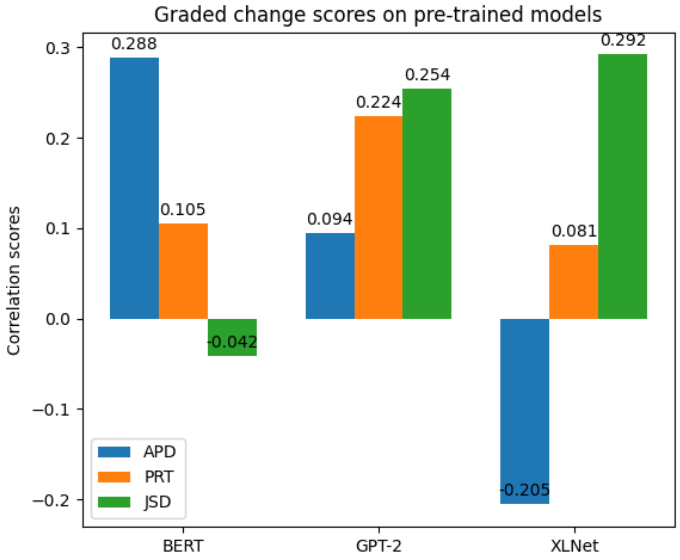


Figure 6.1: Bar charts showing the performance of the pre-trained models on graded change.

When comparing these scores to the baselines, we see that all models are able to perform better than these methods. This holds even for the pre-trained models. The results also shows that it is not always the case however, and that the choice of change detection metric has a large impact on the score. When comparing the scores to the top performing models from [7] instead of the baselines, only the fine-tuned BERT model paired with APD or JSD achieve comparabe results. Similarly to the results from [24], the BERT model obtains the highest score of all the models.

The reason why BERT performs better than the other models is likely caused by both architectural choices and the learning objectives used. GPT-2 likely performs worse because of its autoregressive nature combined with its distribution estimation training objective that causes it to not have access to the latter part of the sequences in its attention calculations. XLNet is also autoregressive, but achieves bidirectionality due to its learning objective. This model might suffer because of the transformer-XL architecture being under-utilized due to the short sequence

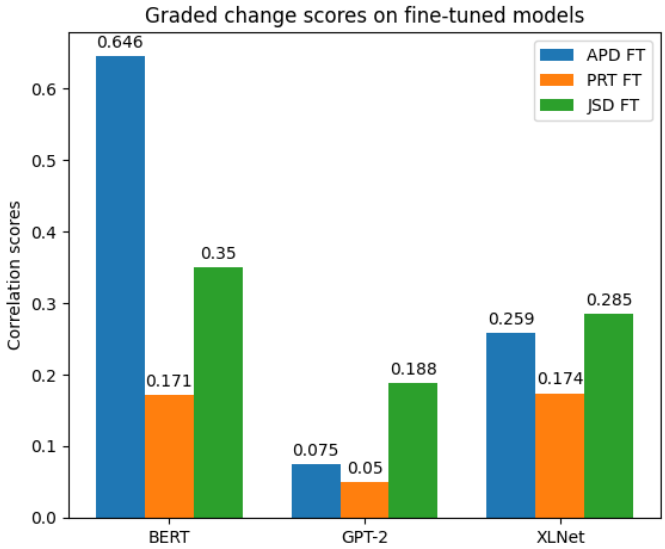


Figure 6.2: Bar charts showing the performance of the fine-tuned models on graded change.

length in the corpora.

6.1.3 Binary lexical semantic change

Table 6.4: The accuracy of the pre-trained models on the binary change task.

	APD	PRT	HLT K-means
BERT	0.568	0.568	**0.622**
GPT-2	0.459	0.541	**0.649**
XLNet	0.486	0.513	**0.568**

Table 6.4 and 6.5 shows the accuracy scores obtained on the binary change task. We see that the scores obtained are much closer to each other than for the graded change. This is the case for both the pre-trained and fine-tuned models. For all models there is an increase in performance after fine-tuning, except when measuring high-low thresholding of K-means cluster values where there is a slight decrease for BERT and GPT-2, and no change for XLNet. Since the change values used for the binary task are based on the graded values, one might expect that the models who perform well on the graded change also perform well on this task, but the results do not indicate this. One Explanation for this might be that using the average thresholded change values is not a metric that is well suited to capture binary semantic change.

When looking at the confusion matrices for the three models paired with the met-

Table 6.5: The accuracy of the fine-tuned models on the binary change task.

	APD	PRT	HLT K-means
BERT	**0.622**	**0.622**	0.514
GPT-2	0.622	**0.649**	0.541
XLNet	0.568	**0.622**	0.568

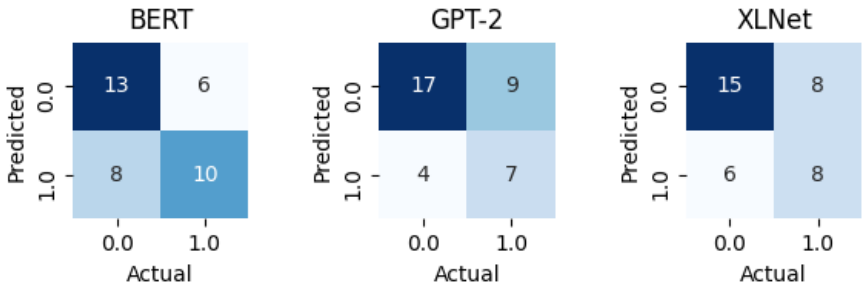


Figure 6.3: The confusions matrices for the models with their best performing binary change metric. 0 represents no change and 1 represents change.

ric they performed best with, as shown in figure 6.3, we see that BERT makes a more balanced number of predictions for the two classes, while the other two are much more prone to predicting words as not changed. In real text, change is a slow process and most words undergo very little change. One might expect models which predict more words as not changed to perform better. In the evaluation dataset on the other hand, the two classes are more balanced. 16 of the target words are marked as changed and the remaining 21 are marked as not changed. Even though the models achieve similar scores, these findings show that they are obtained by differing strategies, which might have implications for their performance given other datasets where the distribution of gold scores differ.

None of the models managed to predict the words *edge*, *land*, and *thump*. The two first words were mentioned in [7] as two of the three most difficult words to

predict for the binary change detection task for English in the original challenge. They could not find a pattern regarding the frequency or polysemy of these words. It might be that the changes these words have undergone are difficult to pick up, but the types of changes undergone by each word are not mentioned in the paper. An analysis of these words showed that *edge* is a narrowing, but did not manage to categorize the changes for the other two.

The models do not show any problems with the two verbs *circle* and *stroke*, but the last verbs, *pin* and *tip* are more challenging. They do also not have problems with the word *contemplation* which is the only word split into multiple tokens. Since so few of the words are verbs or split into multiple tokens, it is not possible to say if these properties have an effect on how easy these words are to classify.

6.1.4 POS-tag removal during feature extraction

Due to the exceedingly much higher scores obtained for BERT compared to the two other models, only the scores for BERT will be shown in this and the following section.

Table 6.6: The effect of POS-tag removal on graded LSC on the lemmatized corpus using pre-trained BERT

	APD	PRT	JSD
No removal	0.288	0.105	-0.042
Remove from target word	0.189	0.156	0.13
Remove from target word and sequences	**0.547***	**0.273**	**0.258**

Table 6.7: The effect of POS-tag removal on graded LSC on the lemmatized corpus using fine-tuned BERT

	APD	PRT	JSD
No removal	**0.646***	0.171	0.35*
Remove from target word	0.244	0.136	0.252
Remove from target word and sequences	0.554*	**0.211**	**0.442***

Table 6.6 and 6.7 shows the results of removing the POS-tag from the target words and the sequences. For pre-trained BERT, we see that the correlation is increased for all metrics when removing the tag from both target words and sequences. For APD though, there is a decrease when only removing it from the target word. For the fine-tuned model however, removal leads to a decrease for APD.

A reason for the increase we observe for the pre-trained model can be the fact that the model is not pre-trained on data that contains such information. The fact that a POS-tag follows the target word in every sequence seen, might result in the attention scores the model produces between the target words to weigh too

heavily on the tag, and thus being able to capture less semantic properties between the other tokens.

One might expect the score to increase when removing the tag from the target words, as this results in less tokens being averaged together to create the final word representation. This was not the case when using APD, which might be a further indication that the model gives the target word high attention to the tag. When the tag is removed completely, however, the scores increase drastically. This is especially noticeable for APD, where the results are even statistically significant. When comparing this to the results presented in [7] we see that the pre-trained model is able to achieve scores surpassing every entry for the English corpus. This is a good indication that the model possesses the ability to find lexical semantic changes without being trained on the specific corpora to be analysed, and that the masked language modelling learning objective is capable of teaching this task.

After the model has been fine-tuned we see that the model performs a lot better with the tag included than before, and combined with APD achieves the highest score of any of the combinations. The results are also higher than those reported by [24], which proposed the methodology used to conduct this experiment. A reason for this discrepancy might be due to the choice of seed used during feature extraction, or due to stochastic initialization of the masked language head used to fine-tune. There might also have been pre-processing steps not mentioned in the paper.

6.1.5 Graded lexical semantic change on raw corpus

Table 6.8: The scores on graded LSC from using fine-tuned BERT on the raw corpus.

	APD	PRT	JSD
Fine-tune on raw	0.587*	**0.281**	**0.505***
Fine-tune on target words lemmatized	**0.61***	0.258	0.491*

We see from table 6.8 that the performance of BERT increases significantly for both PRT and JSD for both experiments, but surprisingly this is not the case for APD. In the first experiment, the model is fine-tuned on the raw dataset without any form of processing. This could lead to many target words being used in word forms that the model tokenizer recognizes as a different token, which would result in the embedding for the target word not being updated nearly as frequently as if the target word was lemmatized. This could again lead to the embeddings being conditioned on fewer differing contexts that it can use to distinguish usages. This would not explain why the APD score decreases while the other increases, though. An intrinsic evaluation of the models would thus be of interest, but is left as further work.

When we then look at the model fine-tuned with lemmatized target words, we see an increase for APD but a decrease for PRT and JSD when compared to the completely raw corpus. The scores for the two latter metrics are still better than their scores for the lemmatized corpus, while the former is still worse. This might indicate that PRT and JSD are more sensitive to the effect of lemmatization.

6.2 Analysis of news data

This section presents the results of the experiments that aim to answer the two final research questions. The model used for the analysis is the best performing model from the evaluation phase, namely BERT paired with APD and JSD.

6.2.1 Laws of semantic change

As mentioned in chapter 3, there are two proposed laws of semantic change. The law of conformity states that words with high frequency are less likely to undergo a semantic change, and the law of innovation states that highly polysemous words are more likely to change. Figure 6.4 and 6.5 plots the two laws. The dark line in both figures indicates that the data show a positive correlation for both laws.

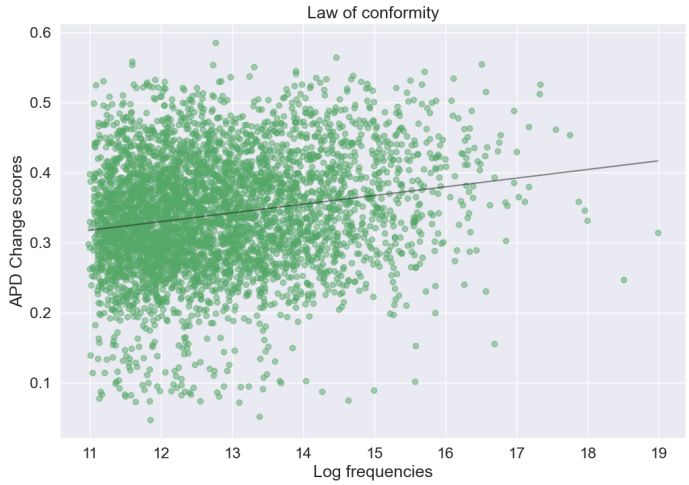


Figure 6.4: The target words plotted with their log transformed frequencies and APD change scores.

Table 6.9 shows the Spearman's rank correlation scores obtained for BERT measured with both APD and JSD. For the law of conformity to hold, there must be a strong negative correlation. The table shows that when measured using APD there is a weak positive correlation, and no correlation is observed when using JSD. This indicates that the law of conformity does not hold for synchronous news

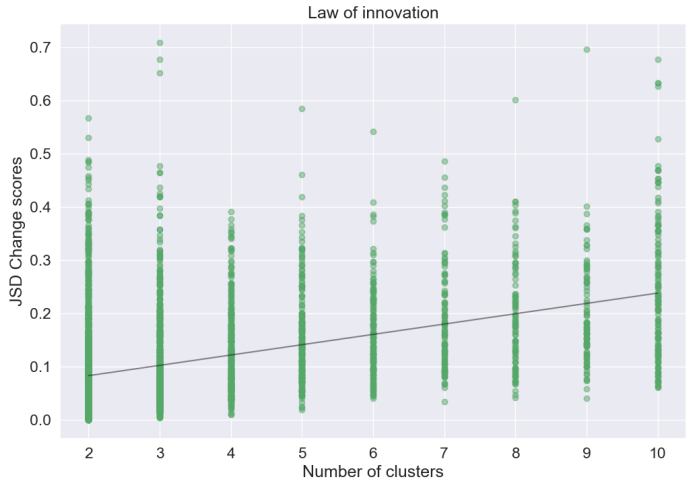


Figure 6.5: The target words plotted with their number of clusters and JSD change scores.

data. There is also a weak to moderate correlation observed for the law of innovation, indicating that the law holds for synchronous news data, but to a lesser extent than for diachronic data.

Table 6.9: The Spearman's correlation scores for both laws of semantic change.

	APD	JSD
Law of conformity	0.175*	-0.027
Law of innovation	0.263*	0.472*

These laws were first proposed for diachronic data, but as mentioned in section 3.3.2 it was later also shown by [15] that the law of conformity strongly holds for synchronous data and that the law of innovation weakly holds. The findings presented here thus agree with previous work on the latter law, but differ on the former. There are multiple potential reasons for the disparity between the results shown in this thesis and those of previous work.

In contrast to this work, [15] does not create one shared vector space for both corpora. Instead they train two separate spaces and subsequently aligns them. The approach that achieves the highest correlation for the law of conformity utilizes anchor words to carry out the alignment. The anchor words chosen are stop words and very frequent words. The assumption is that the anchor words are similar in both spaces and can be used as stationary points to align the rest of the words. This approach will thus give many frequent words little to no change, and the correlation found for the law of conformity will increase as well. Another possible reason for the results obtained is due to noise in the corpora used in this thesis

resulting in artificially high change scores for many of the words found to have the largest amount of change. This is covered more in depth in section 6.2.2.

An important point to note is that these laws are somewhat controversial in the literature. [49] states that the count vector based method PPMI used to create the word representations to propose the two laws is likely to cause a frequency bias. They also hypothesize that the same holds for Skip-gram with negative-sampling, which is the model-architecture used by [15]. It is not shown if the same holds for word representations created through other models, such as the transformer architecture BERT utilizes. [49] also state that polysemy has a positive correlation with frequency, and that some of the correlation observed for the law of innovation is caused by this. The findings in this work may indicate that the dense word representations created by BERT is not as sensitive to frequency bias as count vector based approaches, and corroborates [49] that the effect of frequency on semantic change may not be as high as previously reported. There is also found to be a correlation of only 0.1 between polysemy and frequency in this work, which can further point towards the model not being biased by frequency. Together with the observed reduced effect of frequency, this might explain the weaker correlation of the law of innovation.

6.2.2 Analysis of changes in word senses

This section shows the results of an analysis of the word senses created through the use of K-means clustering. The words evaluated are those with very high or low change scores using both APD and JSD, and a selection of words that show relevant properties for the analysis. The aim was to see if the model was able to create interpretable senses and if it was possible to observe semantic changes between these senses in real and fake news.

interpretability of senses

The first criteria that needs to be fulfilled to be able to observe meaningful semantic changes is that the senses carry a well defined meaning and that they are distinct from the other senses. The senses should also have meaning that are attributed to the actual word, and not noise from the sequences they are used in.

A word that showcases how interpretable and how well the model captures diverse meanings is the word *cross*. Cross is ranked as a word which has undergone a large change, being ranked as the 6^{th} and 176^{th} most changed word using APD and JSD respectively. *Cross* is found to have 6 senses, where all are very interpretable and distinct. They are visualized in figure 6.6, and their meanings are shown in table 6.10.

As seen in table 6.10, the first sense is made up of verb usages. It captures both the literal meaning of crossing a physical distance, and the figurative meaning of

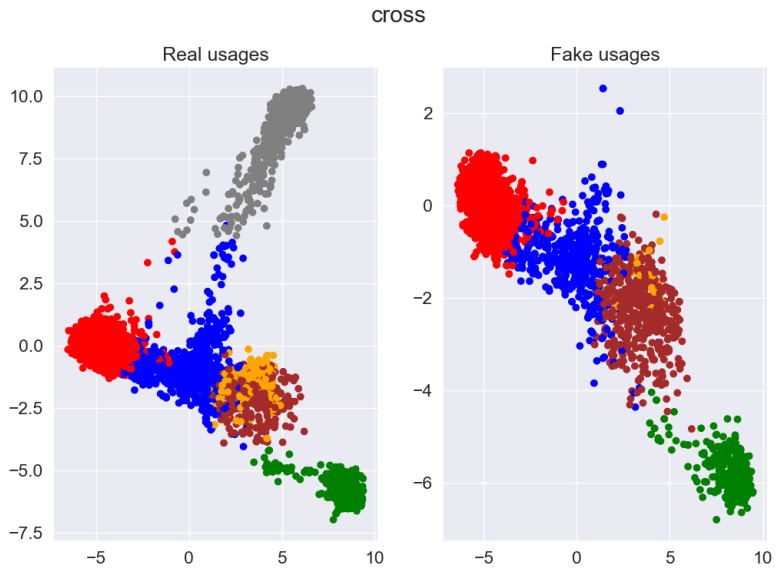


Figure 6.6: The clusters representing real and fake word senses for the word Cross. The figure shows distinct senses, and also one sense that is prevalent in real news that is not found in fake news.

Table 6.10: The meaning of the 6 senses of the word *Cross*.

Color representation	meaning
Red	Verb, "cross the border", "cross the line"
Green	The red cross organization
Blue	Cross as a surname
Orange	Cross as part of a place name
Grey	Sports
Brown	A religious cross

going too far. The three next senses are all names, but names that are used very differently. This shows that the model can distinguish between usages related to places, people and even organizations. While most of the senses have close to no disambiguity in their meaning, the sense of *Cross* being a surname also includes some usages in the context of cross examination. The reason for this might be that the sequences are often phrased as one person being being cross examined by someone else, so the word is often used in conjunction with peoples names, thus making this sense closer in meaning than the others.

Figure 6.6 gives the impression that the two senses for cross as a religious symbol and a surname are interpolated. This is however likely a side effect of PCA dimensionality reduction used to visualize the senses, and is likely not present in the high dimensional data, as inspection showed that these clusters were very

distinct and did not show any sign of overlap in meaning. The figure also shows clearly that the sense related to sports is very prominent in real news, but does not exist at all in fake news. This is a recurring observation for many words and is discussed in the following section.

Artefacts of semantic change in news data

The analysis showed that there are some artefacts present in the results caused by the clustering procedure and noise in the data. These are described in this section.

The first problem comes from the clustering procedure with K-means and silhouette score. As mentioned in section 5.2.4, the clustering procedure for each word is repeated nine times with K-values ranging from two to ten. This is to find the number of clusters that result in the highest silhouette score for the data. The problem arises for the words that one would expect to only have one meaning, which would result in there only being one cluster. This is the case since the silhouette score is not defined when all data-points belong to the same cluster, as it relies on the average distance to the closest cluster a given data-point is not itself part of. The result is that some words with one clearly distinct cluster are separated into two. An example is the name *Trump*. For both real and fake news there is no discernible difference in the meaning of the two clusters.

Another artefact arises due to many news sources repeating similar, but not the exact same sequences in many of their articles, which results in the dataset containing many almost duplicates. These sequences often get clustered together as their own word sense. These senses are then very likely to only be present in one of the corpora, as they are only from one specific source which is either labeled as real of fake news. This results in many words showing a high degree of broadening or narrowing, with little semantic change actually taking place. The sequences are often similar to the form *Subscribe from just 15p a day*, so words like *subscribe*, *blog*, and *newsletter* used to reference the source or the article itself achieve artificially high change scores.

As mentioned in the previous section, many words have a sense relating to sports present in real news that is not observed in fake news. While this can indicate a real difference between the two classes of news, it is also difficult to determine if this is the case, or if it is a property of the sources that publish the news rather than the content itself. It is possible that the dataset happens to include unreliable sources that simply do not focus on sports, while this not being the case for unreliable sources in general.

Broadening and narrowing

Broadening and narrowing happens when a new sense appears or disappears, or when a sense becomes more or less prominent. The latter is the case for the word

hillary, with a sense becoming more prominent in fake news.

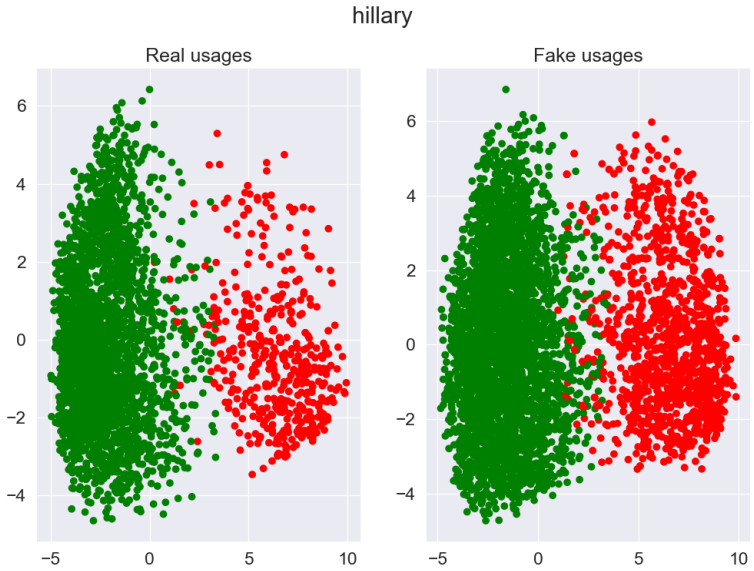


Figure 6.7: The clusters representing reliable and fake word senses for the word hillary.

Figure 6.7 shows the two word senses present for the word *hillary*. The clusters themselves are very general and encompass many varying usages. Observing a distinction between the clusters was not possible, except for one region of the red cluster in figure 6.7. The regions of interest are the topmost right parts of the plots in the figure. This region is concerned almost exclusively with usages of the word related to emails, scandals and fbi or wikileaks investigations. While the whole cluster marked in red is used more frequently in fake news with 451 usages from reliable sources as compared to 1355 from unreliable sources, the increased frequency is especially noticeable for this region. This results in a weak, but noticeably broader usage of the sense in fake news.

Sense shifts

Figure 6.8 shows that the word *chelsea* has three word senses. The cluster marked in red represents Chelsea as the football club, green represents a first name, and blue represents Chelsea Manning specifically. The narrowing observed for sports related senses was discussed in section 6.2.2. The sense of interest in this section is the usage of the word as a first name. Observing the usages from reliable news sources shows that the name is used very generally to write about anyone named Chelsea. In the cases where the sequence refers to Chelsea Clinton, they become more distinctly placed in the vector space, forming the leftmost green spike seen in figure 6.8. When comparing this sense to the fake usages, the part of the cluster

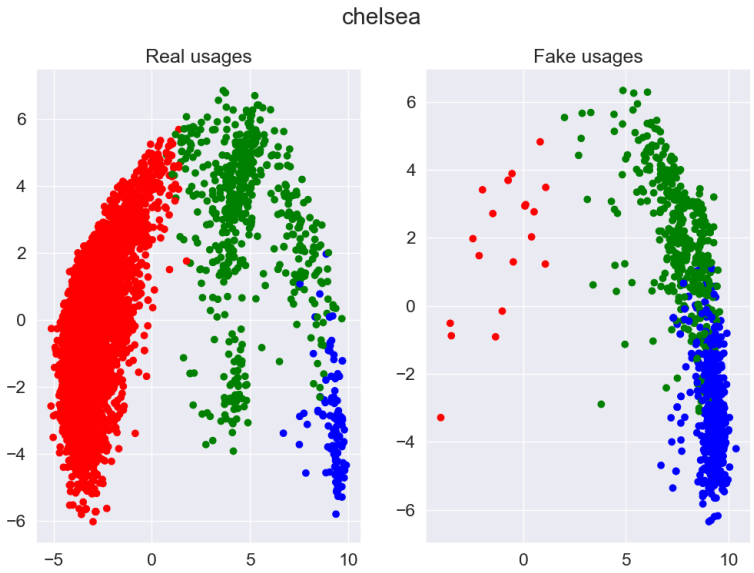


Figure 6.8: The clusters representing reliable and fake word senses for the word chelsea. The figure shows that this word has undergone both a narrowing and a sense shift.

containing general usages is almost completely gone. Only the usages that refer specifically to Chelsea Clinton remain, and one can also observe that there are many more of these usages as compared to reliable sources.

Comparisons between APD and JSD scores

The change values obtained from APD and JSD are relatively different, with the two lists having a Spearman's correlation of 0.347. At the top of the list of changed words for APD are many of the words that have a sense related to sports that is far away from the other senses in the vector space, while the top for JSD consists of words with multiple almost duplicate sequences clustered into new senses. The APD metric is sensitive to the number of usages that is more prominent in either of the corpora, and that result in a high cosine distance to the other points. The presence of these points skews the average distance farther from the average distance observed for the other corpus. For JSD on the other hand, the relative distance between the points are of no importance. It is sensitive to the size of the frequency divergence of the clusters, and the number of clusters with a high divergence. Thus since words often only have one sense related to sports they are not as likely to result in an equally large change when measured using JSD as for APD. The words with almost duplicates are likely to have more than one of these clusters, resulting in high JSD scores.

6.2.3 Reflection on changes in content and wording

An important distinction to make is that of changes detected because of differences in content versus changes caused by differences in wording. The word *belt*, shown in figure 6.9 showcases that the model is able to distinguish different usages of a word that share the same underlying meaning, while still keeping them in the same cluster. The clusters marked in green in the figure contains usages of the word related to the belt and road initiative, which is a name name given to a large Chinese infrastructure and economic development project. The cluster is clearly separated into two, with the smallest being nearly nonexistent in real news. This resembles a semantic shift, but is in reality caused by a change in wording. The smaller cluster represents a rephrasing from the belt and road initiative, to one belt, one road initiative. While the meaning itself is the same between wordings, this is still an interesting observation, as the different wordings might give readers distinct connotations that can lead to amelioration or pejoration, the change where words become more or less positive.

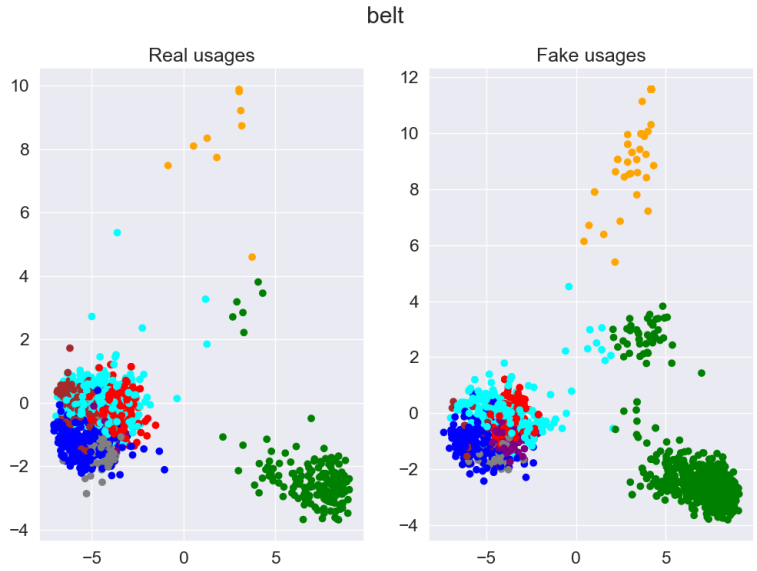


Figure 6.9: The cluster marked in green for the word belt representing the belt and road initiative which seems to have undergone a sense shift, but in reality constitutes a change in wording.

There is a possibility that changes in wording that result in slightly different senses helps elevate the visibility of other changes caused by a change of content, but it can also cancel them out, leaving real changes harder to observe. By increasing the context around the words to sequences with multiple sentences or full paragraphs, this effect might become less noticeable, since the contexts are more likely to gravitate towards fewer but larger common themes.

Chapter 7

Conclusion

This section concludes the work done in this thesis with some final remarks on the research questions and further work needed to advance research on both fake news and lexical semantic change.

7.1 Discussion of research questions

RQ1 *How do different fine-tuned contextual language models perform on the task of semantic change detection?*
This thesis has shown that contextual language models are able to achieve good scores on the task of semantic change detection, but that the models achieve very different results, and that the choice of model and change detection metric plays a large role on the results.

BERT performs well on graded change detection when paired with APD or JSD, achieving Spearman's rank correlations of 0.646 and 0.35 respectively after fine-tuning. GPT-2 and XLNet on the other hand do not achieve similar results. While XLNet is still able to perform better than many of the entries reported in [7] for the English corpora, GPT-2 receives the lowest scores with 0.188 for JSD after fine-tuning.

RQ2 *How does the effect of processing the evaluation corpus affect the results of these models?*
This thesis has shown that the inclusion of a POS-tag suffix to the target words and corpus has a substantial detrimental effect on the performance of the pre-trained BERT model. After fine-tuning, the tag results in an increase in performance when measured using APD while the performance is increased when removing the tag for both PRT and JSD.

Fine-tuning BERT using the raw corpus instead of the lemmatized one shows

a slight decrease in performance for APD, but a substantial increase for PRT and JSD. This shows that contextual language models have the potential to be negatively impacted by the processing applied to the evaluation corpus.

RQ3 *To what degree does the laws of semantic change hold for synchronous news data?*
This thesis found that the law of conformity does not hold for synchronous news data. On the contrary, the results show a slight positive correlation between word frequency and semantic change instead of the opposite that previous research has found for both synchronous and diachronic data. The results also shows a weak positive correlation between polysemy and semantic change, which to a lesser extent supports the results from previous research on the law of innovation. The results might instead support research that the laws are a result of bias in the methods used to formulate them.

RQ4 *Do fake and real news show semantic differences when comparing word senses from a contextual language model?*
This thesis shows that word senses created from BERT uncovers observable semantic differences between real and fake news. Both narrowing and broadening, and sense shifts were observed. This work also shows some issues present when finding semantic changes in domain specific synchronous data that are not as prevalent in general diachronic data. This is namely dissimilarities in the topics covered by the domain in the corpora, as observed with the words with a sense related to sports, and the considerable number of repeating sequences forming their own senses. This shows that the methods are capable, but that there is still more work to do to get precise results.

The results also show that changes in wording can resemble semantic shifts in the content of the words, which might be alleviated by increasing the context of the sequences to multiple sentences or even paragraphs.

7.2 Further work

The results presented in this thesis are promising, and there are multiple areas to explore for improvements and additions that will help further the performance of contextual language models on semantic change tasks and their applicability to explaining fake news. Some of these are mentioned in this section.

7.2.1 Language models

As the results of the models vary greatly, choosing a good model is paramount. Further work should explore the performance of more models, and dive deeper into what aspects of them that has the largest impact on their results. The learning objective, pre-training corpus and architecture type are some of the aspects

of interest. When it comes to exploring more models, since BERT is shown to perform well, RoBERTa and ELECTRA are two interesting models, as RoBERTa is more robustly pre-trained than BERT, and ELECTRA has a similar architecture, but proposes a new learning objective.

7.2.2 Change detection metrics

In the same way that the performance on semantic change varies greatly between models, the same is true between change detection metrics. Having a good understanding of the properties of semantic change that is best captured by each metric will help in deciding for which tasks and data a given metric is most applicable. For the task of binary change detection, methods not reliant on thresholding of graded values should be explored.

7.2.3 Processing duplicate sequences

One of the largest differences between domain specific news and general data is the presence of many similar sequences being repeated. Having these removed is likely to result in much less noisy results. Finding a good way of achieving this that is not too computationally complex is an important part of the future work on domain specific semantic change.

7.2.4 Restrict the news domain

Another way to reduce the noise present in the results is to restrict the data to one specific topic within the news domain. This could be for example politics or culture. This ensures that both corpora are not too skewed from each other based on the topics that the sources included in the dataset report on. At the time of this writing, a large dataset of news labeled with their veracity and that includes topic labels does not exist. Thus, it is also of interest to make such a dataset available.

7.2.5 More languages

Extending the analysis to more languages opens up the opportunity to learn a lot about fake news. More language models are trained on other languages than English, and multilingual models keep performing better. As fake news datasets for these languages becomes available a large area of further work opens up.

7.2.6 More semantic properties

While the literature has not yet firmly solidified the types of semantic change, there are more properties to look at than those covered in this work. One of interest is the property of amelioration or pejoration. By incorporating sentiment

analysis, further work could examine words that are used more or less favourably in fake news.

Bibliography

[1] M. H. Ribeiro, P. H. Calais, V. A. F. Almeida, and W. M. Jr, ""everything i disagree with is #fakenews": Correlating political polarization and spread of misinformation," 2017. arXiv: 1706.05924 [cs.SI].

[2] J. Zarocostas, "How to fight an infodemic," *The lancet*, vol. 395, no. 10225, p. 676, 2020.

[3] S. van der Linden, J. Roozenbeek, and J. Compton, "Inoculating against fake news about covid-19," *Frontiers in Psychology*, vol. 11, p. 2928, 2020.

[4] J. E. Uscinski, A. M. Enders, C. Klofstad, M. Seelig, J. Funchion, C. Everett, S. Wuchty, K. Premaratne, and M. Murthi, "Why do people believe covid-19 conspiracy theories?" *Harvard Kennedy School Misinformation Review*, vol. 1, no. 3, 2020.

[5] F. Zollo, A. Bessi, M. Del Vicario, A. Scala, G. Caldarelli, L. Shekhtman, S. Havlin, and W. Quattrociocchi, "Debunking in a world of tribes," *PloS one*, vol. 12, no. 7, e0181821, 2017.

[6] B. D. Horne and S. Adali, "This just in: Fake news packs a lot in title, uses simpler, repetitive content in text body, more similar to satire than real news," *arXiv preprint arXiv:1703.09398*, 2017.

[7] D. Schlechtweg, B. McGillivray, S. Hengchen, H. Dubossarsky, and N. Tahmasebi, "Semeval-2020 task 1: Unsupervised lexical semantic change detection," *arXiv preprint arXiv:2007.11464*, 2020.

[8] M. Gruppi, B. D. Horne, and S. Adalı, "Nela-gt-2019: A large multi-labelled news dataset for the study of misinformation in news articles," *arXiv preprint arXiv:2003.08444*, 2020.

[9] D. M. Lazer, M. A. Baum, Y. Benkler, A. J. Berinsky, K. M. Greenhill, F. Menczer, M. J. Metzger, B. Nyhan, G. Pennycook, D. Rothschild, *et al.*, "The science of fake news," *Science*, vol. 359, no. 6380, pp. 1094–1096, 2018.

[10] Ö. Özgöbek and J. A. Gulla, "Towards an understanding of fake news," in *CEUR Workshop Proceedings*, 2017.

[11] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD explorations newsletter*, vol. 19, no. 1, pp. 22–36, 2017.

[12] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *Journal of economic perspectives*, vol. 31, no. 2, pp. 211–36, 2017.

[13] J. Allen, B. Howland, M. Mobius, D. Rothschild, and D. J. Watts, "Evaluating the fake news problem at the scale of the information ecosystem," *Science Advances*, vol. 6, no. 14, eaay3539, 2020.

[14] X. Zhou and R. Zafarani, "Fake news: A survey of research, detection methods, and opportunities," *arXiv preprint arXiv:1812.00315*, 2018.

[15] H. Azarbonyad, M. Dehghani, K. Beelen, A. Arkut, M. Marx, and J. Kamps, "Words are malleable: Computing semantic shifts in political and media discourse," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 1509–1518.

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[19] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[20] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[21] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Advances in neural information processing systems*, 2019, pp. 5753–5763.

[22] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," *arXiv preprint arXiv:1901.02860*, 2019.

[23] W. L. Hamilton, J. Leskovec, and D. Jurafsky, "Diachronic word embeddings reveal statistical laws of semantic change," *arXiv preprint arXiv:1605.09096*, 2016.

[24] A. Kutuzov and M. Giulianelli, "Uio-uva at semeval-2020 task 1: Contextualised embeddings for lexical semantic change detection," *arXiv preprint arXiv:2005.00050*, 2020.

[25] M. Giulianelli, M. Del Tredici, and R. Fernández, "Analysing lexical semantic change with contextualised word representations," *arXiv preprint arXiv:2004.14118*, 2020.

[26] N. Hassan, F. Arslan, C. Li, and M. Tremayne, "Toward automated fact-checking: Detecting check-worthy factual claims by claimbuster," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1803–1812.

[27] N. Hassan, G. Zhang, F. Arslan, J. Caraballo, D. Jimenez, S. Gawsane, S. Hasan, M. Joseph, A. Kulkarni, A. K. Nayak, *et al.*, "Claimbuster: The first-ever end-to-end fact-checking system," *Proceedings of the VLDB Endowment*, vol. 10, no. 12, pp. 1945–1948, 2017.

[28] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[29] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[30] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[31] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *arXiv preprint arXiv:1910.10683*, 2019.

[32] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," *arXiv preprint arXiv:2003.10555*, 2020.

[33] N. Tahmasebi, L. Borin, and A. Jatowt, "Survey of computational approaches to lexical semantic change," *arXiv preprint arXiv:1811.06278*, 2018.

[34] X. Tang, "A state-of-the-art of semantic change computation," *arXiv preprint arXiv:1801.09872*, 2018.

[35] A. Kutuzov, L. Øvrelid, T. Szymanski, and E. Velldal, "Diachronic word embeddings and semantic shifts: A survey," *arXiv preprint arXiv:1806.03537*, 2018.

[36] R. Alatrash, D. Schlechtweg, J. Kuhn, and S. S. im Walde, "Ccoha: Clean corpus of historical american english," in *Proceedings of The 12th Language Resources and Evaluation Conference*, 2020, pp. 6958–6966.

[37] M. Pömsl and R. Lyapin, "Circe at semeval-2020 task 1: Ensembling context-free and context-dependent word representations," *arXiv preprint arXiv:2005.06602*, 2020.

[38] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[39] P. Basile, A. Caputo, T. Caselli, P. Cassotti, and R. Varvara, "Diacr-ita@ evalita2020: Overview of the evalita2020 diachronic lexical semantics (diacr-ita) task," *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020), Online. CEUR. org*, 2020.

[40] S. Laicher, G. Baldissin, E. Castañeda, D. Schlechtweg, and S. S. i. Walde, "Cl-ims@ diacr-ita: Volente o nolente: Bert does not outperform sgns on semantic change detection," *arXiv preprint arXiv:2011.07247*, 2020.

[41] D. Schlechtweg, A. Hätty, M. Del Tredici, and S. S. i. Walde, "A wind of change: Detecting and evaluating lexical semantic change across times and domains," *arXiv preprint arXiv:1906.02979*, 2019.

[42] A. Hätty, D. Schlechtweg, and S. S. im Walde, "Surel: A gold standard for incorporating meaning shifts into term extraction," in *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (* SEM 2019)*, 2019, pp. 1–8.

[43] F. Arslan, N. Hassan, C. Li, and M. Tremayne, "A benchmark dataset of check-worthy factual claims," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 14, 2020, pp. 821–829.

[44] T. Mitra and E. Gilbert, "Credbank: A large-scale social media corpus with associated credibility annotations.," in *ICWSM*, 2015, pp. 258–267.

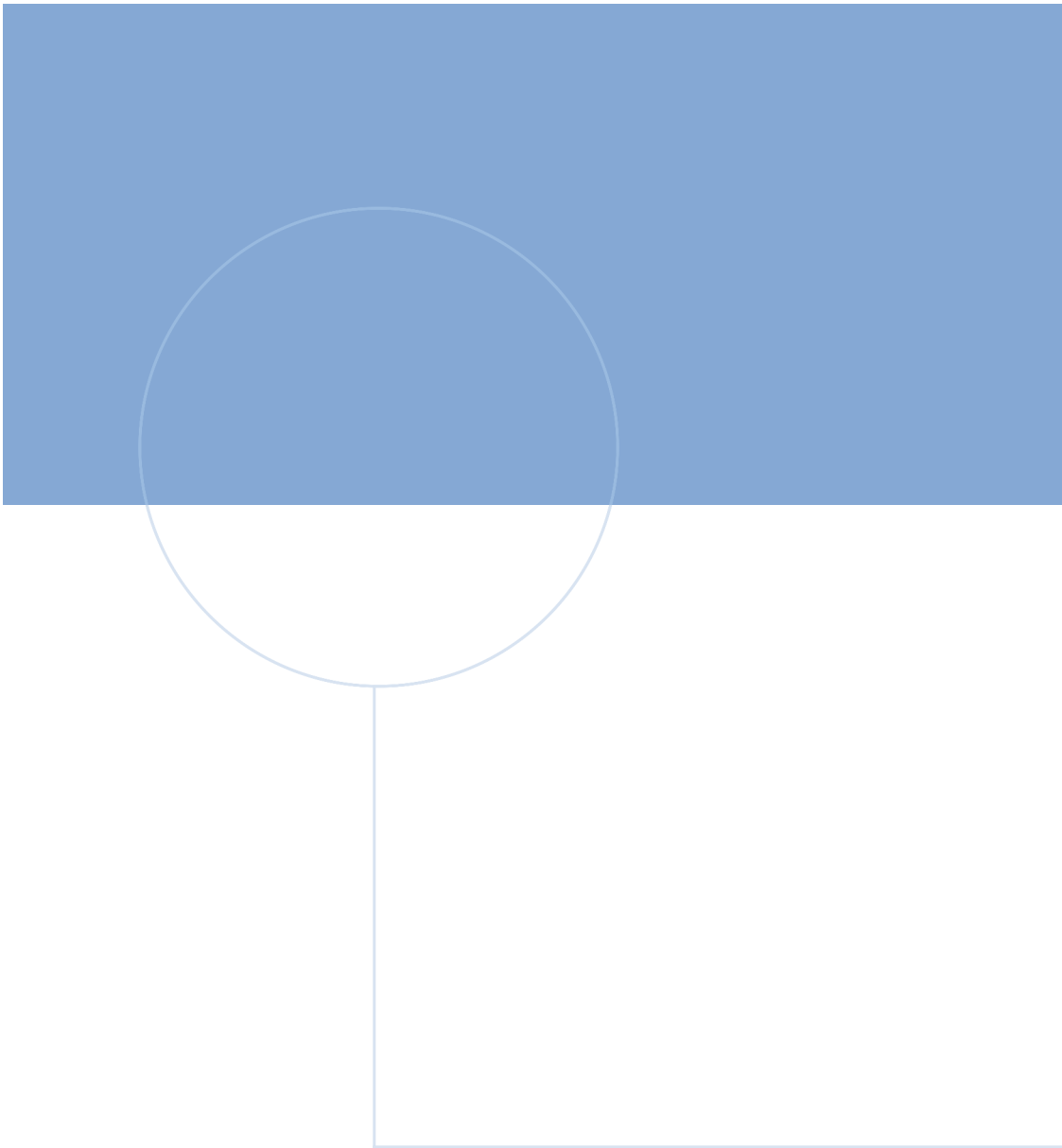
[45] E. Tacchini, G. Ballarin, M. L. Della Vedova, S. Moret, and L. de Alfaro, "Some like it hoax: Automated fake news detection in social networks," *arXiv preprint arXiv:1704.07506*, 2017.

[46] J. Nørregaard, B. D. Horne, and S. Adalı, "Nela-gt-2018: A large multi-labelled news dataset for the study of misinformation in news articles," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 13, 2019, pp. 630–638.

[47] M. Davies, "Expanding horizons in historical linguistics with the 400-million word corpus of historical american english," *Corpora*, vol. 7, no. 2, pp. 121–157, 2012.

[48] D. Schlechtweg, S. S. i. Walde, and S. Eckmann, "Diachronic usage relatedness (durel): A framework for the annotation of lexical semantic change," *arXiv preprint arXiv:1804.06517*, 2018.

[49] H. Dubossarsky, D. Weinshall, and E. Grossman, "Outta control: Laws of semantic change and inherent biases in word representation models," in *Proceedings of the 2017 conference on empirical methods in natural language processing*, 2017, pp. 1136–1145.



NTNU

Kunnskap for en bedre verden