Semantic Word Error Rate: A Metric Based on Semantic Distance

Espen Stokke

Samia Touileb, MediaFutures
Lubos Steskal, TV2

Department of Information Science and Media Studies
University of Bergen

May 31, 2023

Scientific environment

This study is carried out at the Department of Information Science and Media Studies, University of Bergen. Supervision is conducted as a collaboration between MediaFutures Research Centre for Responsible Media Technology & Innovation and TV2, with supervisors from both affiliations.

Acknowledgements

I would like to express my sincerest gratitude to my supervisors, Samia Touileb and Lubos Steskal, for their guidance, mentorship, and detailed feedback throughout the course of this thesis.

I would also like to thank my parents, Karen and Kristian, for their constant support during my time studying.

I am also thankful to Katrine, who helped me in staying focused and hydrated through the final weeks.

Espen James Rodriguez Stokke
Bergen, Norway, 31.05.2023

Abstract

As the Norwegian broadcasting company TV2 wishes to develop new tools for evaluating automatically generated transcripts, we explore the development and evaluation of a new metric for evaluation based on the semantic distance between transcripts. The is achieved through utilizing pre-trained Norwegian BERT models to represent words as word embeddings, and computing the cosine distance between them. The metric is evaluated on the Norwegian Parliamentary Speech Corpus, which provides audio files along with their professionally written transcripts. The audio files are run through a speech to text service, producing automatically generated transcripts that will be evaluated by the metric up against the professional transcripts. In addition, the metric is evaluated by comparing it to human judgement of semantic distance, which is captured by gathering data through an online survey. The results indicate that our metric manages to capture the human opinion of semantic distance to some degree, and can function as a useful metric for evaluating transcripts. The results also show that choice of BERT model can significantly effect the performance of the metric.

Contents

List of Figures

Chapter 1

Introduction

The proliferation of automatic speech recognition (ASR) systems in recent years has contributed significantly to improving access to various forms of content for individuals around the world. Broadcast media, in particular, have embraced these technologies as means to create a more inclusive environment, notably for the deaf and hard-of-hearing community, by automatically creating accurate subtitles for live broadcasts. One such example is the Norwegian broadcasting company TV2, which is keen on enhancing the accessibility and quality of their content.

The utilization of ASR systems in broadcasting presents unique challenges that are different from more controlled use-cases like personal assistant devices. In broadcasting, these systems must accurately transcribe a wide array of speech, often with complex topics, multiple speakers, diverse accents, and background noise. Subtitles that reflect these complexities accurately are not just a matter of convenience but essential for inclusive communication.

As a major broadcasting network, TV2 acknowledges the necessity of integrating ASR systems into their workflow. The utilization of automatically generated transcripts has the potential to significantly enhance the accessibility of their content, particularly for live broadcasts, by enabling the instantaneous creation of subtitles. However, the implementation of such systems demands a system in place for monitoring the level of quality of the generated transcripts, in order to ensure that these transcripts accurately represent and reflect their content.

1.1 Problem Statement

To ensure that the transcripts provided by their speech to text systems are of acceptable quality, TV2 wishes to further develop tools to monitor the performance of automatically generated transcripts. This is where the current thesis aims to contribute by creating a metric that measures the quality of a generated ASR output. The metric we wish to develop measures the semantic distance between transcripts and their gold standard.

Traditional metrics for evaluating transcripts, such as Word Error Rate (WER), are based on measuring the syntactic similarity between transcripts. However, such metrics often fail to capture the semantic congruity. Transcripts might be syntactically different but semantically identical, yet be penalized by traditional metrics.

For example, the sentences “It was a good day” and “Today was a great evening” convey a very similar meaning, but a metric based on syntactic evaluation would penalize the sentences for not being a verbatim match. This highlights the need for an alternate metric to aid in providing a fuller understanding of the quality of transcripts produced from ASR systems.

1.2 Objectives

The main objective of this thesis is to explore the feasibility of developing a metric that penalizes transcripts for their semantic distance, rather than penalizing syntactical distance. This will be conducted by utilized state of the art pre-trained Norwegian language models. Large-scale transformer models are pre-trained on huge amounts of data and achieve state-of-the-art performance in a wide range of natural language processing tasks. Further, the objective is to measure the metrics ability to successfully capture semantic distance between transcripts.

1.3 Contribution

Due to the small amount of Norwegian speakers, the state of the field of natural language processing is naturally not as advanced for the Norwegian language as for other larger languages like English. Metrics for semantic distance currently exist for the English language, and this thesis aims to provide a first analysis of how such metrics can be applied to the Norwegian language by developing a similar metric that uses BERT models pre-trained on Norwegian data.

Additionally, in an attempt to capture the human judgement of semantic distance, we create a survey collecting feedback from people on how distant they perceive sentences to be semantically, providing us with data that depicts human evaluation on a small set of the sentences.

1.4 Research Questions

This thesis aims to explore and answer the following research questions:

- **RQ1:** How does semantic-WER compare to other existing metrics for evaluating transcripts?
- **RQ2:** How can semantic-WER be evaluated and validated against human judgments of semantic similarity or relatedness?
- **RQ3:** How does the performance of semantic-WER compare when using different models for semantic representation?

1.5 Thesis outline

The work of this thesis will be presented in the following chapters:

Chapter 2: Background – provides a brief overview of relevant concepts and technology in the field of natural language processing, ASR, as well as related work within measuring semantics.

Chapter 3: Data – describes the dataset the metric will be tested on.

Chapter 4: Methods – presents the steps included in processing the data and implementing the metric.

Chapter 5: Evaluation – explains the approaches involved for evaluating the performance of the metric.

Chapter 6: Results – presents the results produced by the evaluation process.

Chapter 7: Discussion and Conclusion – discusses the implications of the results as well as the strengths, weaknesses, and limitations of the metric. Finally, we conclude and discuss possible future works.

Chapter 2

Background

2.1 Natural Language Processing

Natural Language Processing (NLP) is a subfield of Artificial Intelligence (AI) that focuses on enabling computers to understand, interpret, and generate human language. NLP combines techniques from computer science, linguistics, and statistics to process and analyze large volumes of natural language data.

The ultimate goal of NLP is to enable computers to understand natural language in the same way that humans do, so that they can interact with humans in a more intuitive and natural way. This involves teaching computers to recognize patterns and structures in language, such as grammar, syntax, and semantics. NLP is used in a wide range of applications, including machine translation, sentiment analysis, text classification, question answering, and speech recognition.

One of the main challenges in NLP is the complexity of human language. Language is highly contextual, meaning that the same word or phrase can have different meanings depending on the context in which it is used. Language is also ambiguous, with multiple interpretations and nuances that can be difficult for computers to understand. In addition, language is constantly evolving, with new words and expressions emerging all the time.

Despite these challenges, NLP has made significant progress in recent years, thanks to advances in machine learning and deep learning techniques. Today, NLP is used in a wide range of industries, including healthcare, finance, and e-commerce, to automate and streamline tasks that would otherwise require human intervention. As the volume of natural language data continues to grow, NLP is poised to become even more important in the years ahead.

2.1.1 Regular Expressions

Regular expressions, also known as regex, are a powerful tool used to describe patterns in text. They are a sequence of characters that define a search pattern, which can be used to match and manipulate text. Regular expressions are commonly used in programming languages, text editors, and command-line tools for tasks such as search and replace, data validation, and text parsing.

A regular expression is made up of literal characters, metacharacters, and quantifiers. Literal characters represent themselves and match exact characters in the text. Metacharacters, on the other hand, have a special meaning and can match more than one character or a certain type of character. For example, the "." (dot) metacharacter matches any single character, while the ""̣ metacharacter matches any digit. Quantifiers specify how many times a character or a pattern should be matched, such as the "*" quantifier which means "zero or more times".

Regular expressions provide a flexible and efficient way to search and manipulate text. They can be used to find and extract specific information from a text document, such as email addresses or phone numbers. They can also be used to validate user input on a web form, ensuring that the data entered by the user follows a certain pattern or format. With regular expressions, complex text processing tasks can be accomplished quickly and easily, making them a valuable tool for anyone working with text data.

2.1.2 Stop Words

Stop words, also known as noise words or function words, are common words in a language that hold little semantic value and are often used for structural or grammatical purposes *Manning et al.* (2008). Examples of stop words in English include "a", "an", "the", "and", "in", "is", "of", "for", and "with" among others. These words are considered "stop" words because they can often be filtered or "stopped" from text processing algorithms without significantly impacting the analysis or interpretation of the text.

In certain text processing tasks, such as keyword extraction, text classification, and document clustering, stop words can be considered as "noise" that might hinder the performance of the algorithms *Rajaraman et al.* (2014). Since stop words are common and occur frequently in text, they can negatively impact the efficiency and accuracy of these tasks. By removing stop words, the algorithms can focus on the more meaningful and content-rich words that contribute to the overall semantic meaning of the text. This filtering process can improve the performance and accuracy of NLP models, as well as reduce computational complexity and storage requirements.

Various techniques and resources exist to identify and remove stop words from a given text. These include pre-defined stop word lists, which are available for many languages and can be easily integrated into text processing pipelines *Manning et al.* (2008). Additionally, more advanced approaches, such as using statistical measures like term frequency-inverse document frequency (TF-IDF), can help to determine the importance of a word in a document or a collection of documents and assist in filtering out stop words *Ramos* (2003).

In conclusion, stop words are common words with little semantic value that play a vital role in the structure and context of a language. Although they are essential for certain NLP tasks, their removal can be beneficial in others where the focus is on extracting meaningful information and reducing noise in the data. Understanding stop words and their role in NLP is crucial for effectively measuring semantic distances between words, as it enables the development of more accurate and efficient text analysis algorithms.

2.1.3 Language Models

One of the fundamental concepts in NLP is the notion of a language model. Language models are probabilistic frameworks for predicting sequences of words. They assign probabilities to sequences of words, which can be used to determine the likelihood of a given sentence or to generate text (*Goodfellow et al.*, 2016). Early versions of language models were predominantly n-gram models, which based their predictions on the prior n words in a sequence (*Jurafsky and Martin*, 2008). However, these models struggled to capture the long-term dependencies and complexities of human language.

A significant advancement came with the introduction of recurrent neural networks (RNNs) and long short-term memory (LSTM) networks (*Hochreiter and Schmidhuber*, 1997). These deep learning models introduced the capability to remember and make use of information from past inputs, allowing for the modeling of long-term dependencies in text.

Recently, a transformative shift has occurred in the language modeling landscape with the development of Transformer-based models (*Vaswani et al.*, 2017). These models use a mechanism called self-attention, enabling them to focus on different parts of the input when producing an output, providing an effective way to handle long-range dependencies.

Googles Bidirectional Encoder Representations from Transformers (BERT) (*Devlin et al.*, 2019) and OpenAIs Generative Pretrained Transformer (GPT) series, including GPT-4 (*Brown et al.*, 2020; *Radford et al.*, 2018), are among the most popular and powerful models based on the Transformer architecture. These models are pretrained on vast amounts of text and can generate human-like text, understand the sentiment of a sentence, answer questions, and much more.

However, despite these advances, creating language models that understand and generate human language with high accuracy, efficiency, and human-like nuance remains a challenging problem. It continues to be an active area of research with great potential for advancement and impact.

2.1.4 Word Embeddings

Word embeddings are a type of natural language processing (NLP) technique that represents words as dense vectors in a high-dimensional space. Each dimension in the vector represents a different feature or attribute of the word, such as its meaning, context, or relationship to other words. Word embeddings are used to capture the semantic and syntactic relationships between words, and are widely used in NLP tasks such as text classification, sentiment analysis, and machine translation.

One of the most popular methods for creating word embeddings is the Word2Vec algorithm, which was introduced by *Mikolov et al.* (2013). Word2Vec is based on a neural network architecture that learns to predict the context in which words occur in a large corpus of text. The resulting word embeddings capture the similarities and differences between words based on their co-occurrence patterns in the text.

Another popular method for creating word embeddings is the GloVe algorithm, which was introduced by *Pennington et al.* (2014). GloVe is based on a matrix factorization approach that uses global word co-occurrence statistics to learn the word embeddings. Like Word2Vec, GloVe embeddings capture the semantic and syntactic relationships between words, and have been shown to be effective in a wide range of NLP tasks.

Word embeddings have many practical applications in NLP, including improving the accuracy of text classification and sentiment analysis models, and enhancing the performance of machine translation systems. They have also been used to analyze the semantic and syntactic structure of natural language, and to study how language evolves over time.

In recent years, researchers have developed more advanced methods for creating word embeddings, such as contextualized embeddings and multilingual embeddings. These embeddings capture more nuanced relationships between words by taking into account the context in which they occur, and are able to handle multiple languages simultaneously.

Contextualized Embeddings

Contextualized embeddings have become an important area of research in natural language processing (NLP) in recent years. Traditional word embeddings, such as word2vec and GloVe, represent each word in a fixed and context-independent manner, which can limit their effectiveness in tasks that require understanding of the nuances of language (*Mikolov et al.*, 2013; *Pennington et al.*, 2014).

In contrast, contextualized embeddings take into account the context in which a word appears and generate a different embedding for the same word depending on its context. This allows them to capture more nuanced meanings and relationships between words, making them particularly useful for tasks such as text classification, sentiment analysis, and machine translation (*Devlin et al.*, 2019; *Peters et al.*, 2018).

One popular approach to generating contextualized embeddings is through the use of transformer models, such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer). These models are pre-trained on large corpora of text data and are able to generate high-quality embeddings for individual words, as well as for entire sentences or paragraphs (*Devlin et al.*, 2019; *Radford et al.*, 2018).

BERT, for example, uses a masked language modeling task during pre-training, where it is trained to predict the masked words in a given sentence. This allows it to capture the context-dependent relationships between words and generate embeddings that are specific to the context in which they appear. Similarly, GPT is trained on a language modeling task, where it is trained to predict the next word in a given sentence based on the preceding context (*Radford et al.*, 2018).

Contextualized embeddings have been shown to outperform traditional word embeddings in a variety of NLP tasks, such as question answering, sentiment analysis, and named entity recognition (*Devlin et al.*, 2019; *Peters et al.*, 2018).

Multilingual Embeddings

Multilingual embeddings are a type of contextualized embeddings that have gained significant attention in recent years due to the increasing need for natural language processing (NLP) systems that can handle multiple languages. These embeddings are designed to be applicable across multiple languages, enabling the same model to be used for tasks in multiple languages without the need for language-specific training.

One of the early approaches to multilingual embeddings was to learn a shared embedding space for multiple languages. For instance, the polyglot embedding method (*Al-Rfou' et al.*, 2013) used a shared embedding space to represent words across multiple languages. However, these methods did not always result in the best embeddings for each individual language.

Recent approaches have utilized pre-trained transformer models, such as mBERT (multilingual BERT) and XLM-RoBERTa (cross-lingual language model RoBERTa) to learn multilingual contextualized embeddings. These models are pre-trained on large datasets of text in multiple languages and can generate high-quality contextualized embeddings for words in each of these languages (*Conneau et al.*, 2020; *Devlin et al.*, 2019).

mBERT, for example, was pre-trained on a large corpus of text in 104 languages and can generate high-quality contextualized embeddings for each of these languages. It has been shown to perform well on a variety of cross-lingual tasks, such as cross-lingual sentiment analysis and machine translation (*Pires et al.*, 2019; *Wu and Dredze*, 2019). Similarly, XLM-RoBERTa is pre-trained on a large corpus of text in 100 languages and has achieved state-of-the-art performance on many cross-lingual tasks *Conneau et al.* (2020).

Multilingual embeddings have the potential to greatly improve the performance of NLP systems for multilingual applications, particularly in low-resource languages where training data may be limited.

2.1.5 Transformer Networks

Transformer networks, first introduced by Vaswani et al.(*Vaswani et al.*, 2017), have revolutionized the field of natural language processing (NLP) and have since become the backbone of state-of-the-art language models like BERT (*Devlin et al.*, 2019), GPT (*Radford et al.*, 2018), and T5 (Raffel et al., 2019). The key innovation behind transformers is their ability to effectively model long-range dependencies in sequences, which is achieved by utilizing self-attention mechanisms. This section provides an overview of the transformer architecture, its key components, and the motivation behind its development.

The development of transformer networks can be traced back to the limitations of previous architectures, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs). While these models achieved considerable success in natural language processing tasks, their sequential nature made it difficult to process long-range dependencies and parallelize training (*Vaswani et al.*, 2017).

The transformer model introduced a new approach called the self-attention mechanism, which allowed it to learn and represent long-range dependencies more effectively. In this mechanism, the model computes a weighted sum of all input tokens, with the weights determined by the relevance of each token to the others (*Vaswani et al.*, 2017). This attention-based method enables the model to capture relationships between words in a sequence, regardless of their distance.

A transformer network consists of an encoder and a decoder, both built using multiple layers of self-attention and feed-forward layers. The encoder takes an input sequence, applies self-attention, and generates a high-level representation of the input. The decoder uses this representation to generate an output sequence, applying self-attention within its layers and attending to the encoder's outputs (*Vaswani et al.*, 2017).

Since the transformer model does not inherently capture the order of input tokens, *Vaswani et al.* (2017) introduced positional encoding to provide information about the positions of tokens in a sequence. These encodings are added to the input embeddings before being fed into the model, allowing the network to learn and utilize positional information effectively.

Large-scale transformer models are pre-trained on massive amounts of text data using unsupervised learning techniques. These models are typically fine-tuned on smaller, task-specific datasets to achieve state-of-the-art performance in a wide range of natural language processing tasks (*Devlin et al.*, 2019; *Radford et al.*, 2018).

2.1.6 BERT

Bidirectional Encoder Representations from Transformers (BERT) is a powerful language model introduced by *Devlin et al.* (2019). BERT is based on the Transformer architecture and is trained on a large corpus of text data using a self-supervised learning approach, which allows it to capture the context and meaning of words in a sentence more effectively than traditional language models.

One of the key features of BERT is its ability to capture the bidirectional context of words in a sentence. Unlike traditional language models that process text in a left-to-right or right-to-left manner, BERT processes text in both directions simultaneously, allowing it to capture the full context of each word in a sentence. This results in more accurate predictions of the meaning of words in context and has led to significant improvements in a wide range of NLP tasks, such as question answering, text classification, and named entity recognition.

BERT has also been designed to handle multiple languages and can be trained on large datasets in different languages. This makes it a valuable tool for researchers and developers working on multilingual NLP applications.

Since its introduction, BERT has been widely adopted in the NLP community and has inspired the development of several variants and extensions, such as RoBERTa, ALBERT, and ELECTRA. These models build on the success of BERT by incorporating additional training techniques, such as larger datasets and different training objectives, to further improve performance on various NLP tasks.

BERT and its variants have achieved state-of-the-art performance on many benchmark datasets and are rapidly becoming the standard baseline for many NLP tasks. As the field of NLP continues to evolve, it is likely that BERT and its derivatives will continue to play an important role in developing more advanced and accurate language models.

NB-BERT

NB-BERT, one of the best performing language models for the Norwegian language, was created by the AI lab at the National Library of Norway (*Kummervold et al.*, 2021). It has outperformed mBert, a multilingual model, on numerous NLP tasks for both Bokmål and Nynorsk, including named entity recognition, part of speech tagging and sentence classification.

NB-BERT is trained on 109.1 GB of text consisting comprising a set of 18,438M words after deduplication. The model is initated from the pre-trained mBERT weights. mBERT was trained on 104 languages, including both Bokmål and Nynorsk Norwegian (*Devlin et al.*, 2019).

NorBERT 1 & NorBERT2

NorLM, an ongoing community iniative, introduced both NorELMo and NorBERT1, based on ELMo and BERT respectively, which are the first large-scale monolingual language models for Norwegian (*Kutuzov et al.*, 2021). NorBERT1 is trained specifically for the Norwegian language and can be used for the same purposes as the original BERT. The vocabulary size of the model is 30,000, which consists of almost exclusively Norwegian words.

The model is trained on the Norsk Aviskorpus (NAK)[1], which is a collection of Norwegian news texts in both Bokmål and Nynorsk, consisting of 1.7 billion words. The model is also trained on data from both Bokmål and Nynorsk Wikipedia, consisting of 160 million and 40 million words respectively. In total, the training corpus comprises about two billion word tokens in 203 million sentences.

NorBert1 is trained from scratch for Norwegian and utilizes a training setup built on the work conducted by the creators of FinBERT (*Virtanen et al.*, 2019). It features a custom WordPiece vocabulary, containing a much better coverage of Norwegian words than NB-BERT, which uses the same vocabulary as mBERT.

NorBERT2 is trained on the non-copyrighted part of the Norwegian Colossal Corpus[2] and a random sample of the Norwegian part of the C4 web-crawled corpus[3], com-

[1]https://www.nb.no/sprakbanken/ressurskatalog/oai-nb-no-sbr-4/
[2]https://huggingface.co/datasets/NbAiLab/NCC
[3]https://aclanthology.org/2021.naacl-main.41/

prising of 5 billion words and 9.5 billion words respectively. In total, NorBERT2's training corpora consists of about 15 billion word tokens in about 1 billion sentences, containing both Bokmål and Nynorsk text.

S-BERT

S-BERT (Sentence-BERT) is a modification of the BERT architecture that is specifically designed for sentence embeddings (*Reimers and Gurevych*, 2019). Sentence embeddings are a way of representing the meaning of a sentence in a vector form, which can be useful for a variety of natural language processing tasks, such as text classification, information retrieval, and similarity matching.

S-BERT works by fine-tuning a pre-trained BERT model on a sentence-level objective, rather than a word-level objective as in the original BERT model (*Reimers and Gurevych*, 2019). This allows the model to learn more effective sentence-level representations that capture the semantic meaning of the sentence, rather than just the individual words. S-BERT achieves this by using a siamese network architecture, where two copies of the pre-trained BERT model are used to encode two sentences, and the similarity between the encoded vectors is used as a proxy for the semantic similarity of the sentences.

S-BERT has been shown to achieve state-of-the-art performance on a range of sentence-level natural language processing tasks, such as semantic textual similarity, paraphrase detection, and text classification (*Reimers and Gurevych*, 2019). It has also been used in a variety of practical applications, such as search engines, chatbots, and recommendation systems.

Overall, S-BERT represents a significant advance in the field of sentence embeddings and has the potential to improve the performance of many natural language processing applications.

XLM-RoBERTA

XLM-RoBERTa is a powerful cross-lingual language model that was developed by Facebook AI (*Conneau et al.*, 2020). It is based on the RoBERTa architecture, which is itself a variant of the BERT model, and has been pre-trained on a massive amount of text data from over 100 languages. This allows it to understand and generate text in multiple languages, making it particularly useful for cross-lingual natural language processing tasks.

The XLM-RoBERTa model achieved state-of-the-art results on a number of benchmark datasets for multilingual language modeling, including the XNLI cross-lingual classification task and the XTREME benchmark for cross-lingual understanding (*Con-*

neau et al., 2020). It has also been used in a range of natural language processing applications, including machine translation, sentiment analysis, and question answering.

XLM-RoBERTa's superior performance can be attributed to several key factors, including its large-scale pre-training on diverse text sources, its use of dynamic masking during pre-training, and its fine-tuning on downstream tasks using a simple and effective training strategy (*Conneau et al.*, 2020). Overall, XLM-RoBERTa represents a significant advance in the field of cross-lingual language modeling and has the potential to revolutionize the way we approach multilingual natural language processing tasks.

2.2 Automatic Speech Recognition

Automatic Speech Recognition (ASR) is the technology which allows for communication between human beings and information processing devices through speech, rather than the more traditional approaches such as using a keyboard (*Arora and Singh*, 2012). Speech is the fastest and most widely means of communication among humans, however, for it to be useful in the context of machine interaction, the machine must understand speech with a high accuracy.

Several parameters contribute to affecting the accuracy of ASR systems, some of them being word recognition, vocabulary, environment, noise, and word pronunciation. In addition to these parameters, ASR systems face several difficulties such as having to understand dialects, and the lack of contextual information such as a speaker's body language.

2.2.1 End-to-End Speech Recognition

Previously, automatic speech recognition was largely based on Hidden Markov Models, where HMM-GMM became the mainstream acoustic model. This changed after Li Deng and Hinton proposed applying neural networks and deep learning to the field, which resulted in an upsurge in research of speech technology.

As developing speech recognition systems may be complicated, end-to-end speech recognition has been proposed as a solution. End-to-end is a system where a sequence of input acoustic features are directly mapped to a sequence of words. The system is trained to optimize criteria relevant to our final evaluation metric.In contrast, conventional ASR often includes separately trained acoustic, pronunciation and language model components.

Deep Learning has brought significant advancements to the field, leading to the development of end-to-end speech recognition systems. These models learn to map the raw audio input to the corresponding textual transcription, circumventing the need for explicit intermediate representations (*Hinton et al.*, 2012).

The two main models of end-to-end speech recognition are Connectionist Temporal Classification (CTC) (*Graves et al.*, 2006) and the Attention model. Unfortunately, E2E speech recognition with CTC still relies on language models to get better results.

Later, the advent of sequence-to-sequence models, initially applied to machine translation tasks, introduced an encoder-decoder framework to the speech recognition problem (*Sutskever et al.*, 2014). The encoder processes the input sequence, which is audio features, and the decoder generates the output sequence , which is transcribed text.

Recently, Transformer-based models, with their self-attention mechanism, have also been successfully applied to speech recognition tasks, providing an effective way to handle long-range dependencies in the audio input (*Vaswani et al.*, 2017).

Despite these advances, creating end-to-end speech recognition systems that can handle variations in accent, noise, speaker characteristics, and different languages with high accuracy remains a challenging problem. It continues to be an active area of research with significant potential for practical applications.

2.3 Correction of ASR

Even though ASR has progressed to the point where it can be applied commercially, high error rates in some speech recognition domains remain a major barrier to widespread adoption of the speech technology, especially for the applications of continuous large vocabulary speech recognition. The continuing presence of errors in ASR has increased the need for alternate ways to detect and correct such errors automatically.

In recent years the use of E2E models has emerged, outperforming the conventional models built on Hidden Markov Models. E2E models are easier and faster to implement, train and deploy, in contrast to prior models that require training several independent components.

To improve the accuracy of speech recognition, independently trained language models have been used on ASR models to re-score the list of n-best hypotheses. This introduces a tradeoff between speed and performance where simple N-gram models will re-score quickly, while more advanced models like transformers will perform better*Hrinchuk et al.* (2020).

Language model re-scoring can be effective, however, it may not have much effect if the ground truth word is wrongly given a low score by the ASR model. Additionally, traditional left-to-right models may accumulate errors as a result of an early wrong word, as the error will affect the scores of all succeeding words. This paper aims to address these issues by training a transformer model that corrects errors by translating erroneous outputs to correct language.

For the baseline ASR model, Jasper is used, which is a deep convolutional E2E model. For re-scoring the hypotheses, two models are used, a 6-gram KenLM and a Transformer-XL model. For correction of the outputs, a transformer model is used, with two different options for initializing weight, the first being random initialization, and the second being using the weights of BERT, a pre-trained model.

The best performing model was the model initialized with pre-trained BERT weights for both encoder and decoder, which was able to successfully correct errors in the ASR outputs where a mistakenly generated context would negatively affect traditional left-to-right language models.

2.4 Metrics

To measure the accuracy and quality of the output of ASR systems, some metrics and tools are needed. These require access to the correct transcripts to assess the quality of the generated ones.

2.4.1 WER

For ASR evaluation, the Word Error Rate (WER) is the most popular metric. WER is derived from the Levenshtein distance, which is also called the edit distance (*Mccowan et al.*, 2004). The edit distance from one string to another is determined by the number of insertions, deletions and substitutions that are required to transform one string into the other. The WER is calculated by taking the edit distance of a reference word sequence and a generated transcript and normalizing it by dividing by the length of the reference word sequence. The formula for WER is defined as such:

$$WER = \frac{S + D + I}{N_r} \tag{2.1}$$

The reason for normalizing the edit distance is so that the metric can be compared across different systems with different tasks in mind.

Even though WER is the most popular metric for evaluating ASR systems, it has its disadvantages and shortcomings. The first one being that it is not a true percentage due to not having an upper bound (*Errattahi et al.*, 2018). This results in the metric not necessarily providing information on how good a system performs by itself, but rather how it compares to another.

However, the WER is still effective for cases of speech recognition, such as dictation, where the error can be corrected by typing. For other tasks it is necessary to consider alternative evaluation methods.

While it is mostly quoted as the Word Error Rate, it can also be quoted as the word recognition rate, which is defined as such:

$$WRR = 1 - WER \tag{2.2}$$

Another measure used, often for word recognition systems, is the Word Correct Rate. The WCR does not take insertion errors into consideration and is defined as:

$$WCR = \frac{H}{N_r} \tag{2.3}$$

For the equation, note that $H = N_r - (S + D)$.

2.4.2 Cosine Distance

Cosine distance is a widely used metric in natural language processing (NLP) to measure the similarity between two vectors. It has been employed in many NLP applications such as text classification, information retrieval, and clustering tasks(*Li*, 2013).

Cosine distance calculates the cosine of the angle between two vectors, and returns a value between 0 and 1, where 0 indicates no similarity and 1 indicates perfect similarity. In the context of NLP, cosine distance is often used to calculate the similarity between two text documents represented as vectors of word counts or word embeddings (*Manning et al.*, 2008; *Salton et al.*, 1975).

For example, if we have two text documents represented as vectors x and y, where each element in the vector represents the frequency of a particular word in the document, we can calculate the cosine distance between the two vectors using the following formula:

$$CosineDistance(x,y) = \frac{(xy)}{(||x||||y||)} \tag{2.4}$$

Here, the dot product of x and y is divided by the product of their magnitudes, which results in a value between -1 and 1. The resulting value represents the cosine of the angle between the two vectors, and the cosine distance is equal to 1 minus the cosine similarity.

One advantage of using cosine distance is that it is insensitive to the magnitude of the vectors, and only takes into account the direction of the vectors. This means that even if two vectors have different lengths, they can still be considered similar if they have the same direction. Additionally, cosine distance is computationally efficient, making it a popular choice for large-scale NLP tasks (*Goldberg*, 2015).

2.5 Semantic WER

Despite significant advances within E2E ASR systems, the ways of evaluating the quality of speech recognition systems remain largely unchanged with Word Error Rate still being the standard metric for evaluation. WER is calculated by total error count normalized by the reference length, where the total error count consists of the sum of substitutions, insertions, and deletions. A limitation of WER is that both content words and function words are equally important, which is not ideal. WER fails to capture semantic differences between hypotheses, making WER less ideal for evaluating ASR correctness.

To improve the usability of WER, *Roy* (2021). proposes an alternative evaluation metric called Semantic-WER which aims to build upon WER to include the semantic weight of the words in a sentence. SWER introduces weights to substitution, deletion and insertion. The substitution weight has four cases:

- 1, if the reference word belongs to the set of named entities and sentiment words.
- The character error rate of the reference word and hypothesis word if the reference word belongs to the set of spelled out entities.
- 1 if the cosine similarity of the embeddings of the reference and hypothesis words is less than 0.6 and the reference word does not belong to the set of named entities and sentiment words.
- 0 if the cosine similarity of the embeddings of the reference and hypothesis words

is greater than 0.6 and the reference word does not belong to the set of named entities and sentiment words.

This results in the WER not penalizing semantically similar words, so that for a reference word go and a hypothesis word goes a substitution penalty is not applied. Similar rules are applied to deletion of words.

To evaluate the results of the SWER, it was compared to the WER and the Human WER, which is a score created by humans subjectively scoring sentences. The results show that the SWER was closer in score to the HWER than the WER, meaning that SWERs scoring may be more accurate in scoring sentences according to how humans perceive them.

Chapter 3

Data

For this thesis the Norwegian Parliamentary Speech Corpus (NPSC)[1] was used. This dataset consists of audio files from the Norwegian parliament, as well as their transcriptions which have been transcribed by trained professionals. An automatic speech recognition system was run on the audio files to generate automatic transcriptions, which then could be compared and evaluated up against the gold standard transcriptions.

3.1 Norwegian Parliamentary Speech Corpus

The Norwegian Parliamentary was created in 2019-2021 by the Norwegian Language Bank at the National Library of Norway. The corpus contains recordings of speech from inside the Norwegian parliament, as well as the corresponding orthographic transcriptions.

The transcriptions are written in both of Norway's written languages, Norwegian Bokmål and Norwegian Nynorsk. For each sentence the original written language is annotated. Additionally, metadata is included for all speakers which contains data on the speaker such as their name, spoken dialect, place of birth, date of birth, and gender. For all speakers, information on their dialect is included. However, since the other data is extracted from WikiData, some fields have null values for some speakers.

The transcriptions are performed manually by linguists and philologists who have undergone appropriate training. Then the manual transcriptions are checked to ensure that they are accurate and consistent.

The corpus was primarily created with the intention of being an open-source dataset utilized for development of automatic speech recognition. There are several reasons for the Norwegian parliament being chosen as the source of the data, one being that the speakers are well known individuals, making details on most of the speakers' back-

[1] https://www.nb.no/sprakbanken/en/resource-catalogue/oai-nb-no-sbr-58/

grounds simple to acquire from the public domain. Another reason is that the parliament consists of speakers from all regions of Norway, which results in a corpus containing a wide variety of Norwegian dialects.

The audio files make up a total duration of 140 hours, 20 minutes, and 16 seconds, which is shortened to approximately 126 hours when pauses are excluded. For meetings with a duration longer than six hours, the audio is cut at approximately 6 hours and 10 minutes, which results in some files having sentences end mid-sentence at the end.

For the files containing the metadata and the transcriptions, the data is formatted in JavaScript Object Notation (JSON), encoded in UTF-8.

The corpus was transcribed with some guiding principles, which consist of consistency, standardized orthography, faithful rendering of speech, and flagging of non-standard speech. To ensure the data set adheres to these principles, the transcribers followed a set of common guidelines.

Additionally, a set of conventions have been followed for certain aspects of the data set. The first convention being that each full sentence shall be a segment in the transcription. The second convention is that non-standard language should be annotated as such. Considering the fact that there are many dialects, yet only two written languages, there exists a fair amount of words which deviate orally from their written version. These words are flagged in the data set as having non-standard spelling. The final convention regards hesitations, interruptions and inaudible speech. Hesitations are flagged with a *special_status*, where its value corresponds to either *HESITATION*, *INTERRUPTED*, *INAUDIBLE*, or *OVERLAPPING*, where *OVERLAPPING* is used for when multiple speakers are speaking at once.

The transcribers have written the transcriptions to be of the "spoken domain", which means that the transcriptions should be as faithful to the speech as ortography allows. Specifically, numbers, years, and dates are written with letters as they are pronounced, e.g. the number 100 000 is written as *hundre tusen*. Additionally, abbreviations are not used unless the speaker specifically pronounces the abbreviation.

For each sentence exists a corresponding normalized sentence where numbers, years, dates, and percentages are written as digits, and standard abbreviations are used where it is appropriate.

Seeing as the data set only contains approximately 12% Nynorsk, the creators of the data set ran a script using machine translation to convert all Bokmål sentences to Nynorsk. Since they then already had the script in place, they applied the same process, but from Nynorsk to Bokmål. This results in the data set containing machine translations to the opposite written language for all sentences. However, the machine translated sentences do not have the same guarantee of quality as the manually transcribed ones. Finally, the corpus contains some English sentences, but none of these

	Duration (h)	Nynorsk (%)	Female (%)	Sentence length	West. (%)	East. (%)	Sout. (%)	Nort. (%)	Trønd. (%)
Train	100.3	12.8	37.7	18.7	26.5	46.1	6.5	11.9	9.1
Test	12.3	13	39.9	18	35	44.4	4.6	9.4	6.5
Eval	13.1	12.7	41.8	18	32.2	43.6	7.6	7.7	8.8
Total	125.7	12.8	38.3	18.6	27.9	45.6	6.4	11.2	8.8

Table 3.1: Data split statistics.

are translated and are contained in the data set in their original form.

For this thesis, the sentences without normalization will be used, in their original written language, meaning the data has not been affected by machine translation. The reason for this is to provide the best gold standard transcriptions for comparison with the automatically generated transcriptions later on.

Each folder in the data set is named after the date of the meeting, and contains files named after the date and time of the audio recording in the format *yyyymmdd-hhmmss*. A data split based on the folder names has been defined in advance. The data split aims to approximately split the data into a train-test-evaluation split of 80-10-10 based on audio duration, where pauses are excluded. The reason the split is based on folders is because a folder a contains one meeting day, and making the splits contain full days is beneficial for Natural Language Processing tasks where context in the text matters. The full overview of which split each folder belongs to can be seen in Table A.1

The data was split with the goal of each split maintaining an equal distribution of Nynorsk speakers, female speakers, and sentence length. The splits vary 4.1% in the percentage of female speakers, where the lowest percentage is 37.7 and the largest is 41.8. For Nynorsk, the test set contains the largest percentage, with 13% Nynorsk speakers. While the smallest percentage is found in the evaluation set, with 12.7%. This makes a difference of 0.3%, meaning the distribution of Nynorsk speakers across splits is evenly distributed. Finally, for average sentence length, the splits differ from 18 to 18.6 words per sentence, meaning the data splits are very similar in average length.

The creators of the data set did not take distribution of dialects into consideration upon splitting the set, however, upon investigation the distributions of dialects are fairly reasonable in the resulting splits. A complete overview of statistics for the splits can be seen in Table 3.1.

Regarding the size of the data set, it contains a total of 1 152 471 words, which divided by 267 speakers yields an average of 4316.37 words per speaker. These words make up 61 862 sentences, resulting in an average of 231.69 sentences per speaker. The average length of a sentence is 18.7 words.

As mentioned, the data set includes 267 different speakers, where *Eastern Norway* is the most prominent dialect, and *Southern Norway* is the least. For one speaker the dialect is marked as unclear. A complete overview of dialect distribution can be seen

	Number of speakers	Sentences	Words
Eastern	116	28 184	487 215
Western	70	17 068	345 091
Northern	33	7 015	131 800
Southern	19	5 463	81 799
Unclear	1	18	49

Table 3.2: Dialect distribution.

	Number of speakers	Sentences	Words
Male	154	37 720	712 163
Female	111	24 039	438 787
None	2	103	1 521

Table 3.3: Gender distribution.

in Table 3.2. Eastern Norwegian speakers make up a total of 28 184 sentences, while the least represented dialect, Southern Norwegian, only makes up 4 114 sentences. The second most used dialect is Western Norwegian with 17 068, which is a reasonably large amount compared to the lesser used dialects. A distribution of sentences per dialect can be seen in Table 3.2.

Considering the large difference in represenation between the most used and least used dialects, any experimental result based on this data set may contain a certain bias towards the over represented groups.

The gender distribution in the data set is to some extent even with 111 female speakers making up 24 039 sentences containing a total of 438 787 words, while there are 154 male speakers who make up 37 720 sentences for a total of 712 163 words. As seen in Table 3.3, there is a significantly larger amount of data from male speakers, however, the amount of data from female speakers should make up a large enough amount for experiments to be conducted without yielding a too significant bias. There are two speakers with unidentified genders, which could be a result of the audio transcribed being inaudible due to e.g. microphone issues, or perhaps the speakers not fitting in to the binary gender categories.

Seeing as this data set only groups dialects into *Eastern Norway*, *Southern Norway*, *Western Norway*, *Northern Norway*, and *Trøndelag*, it may be interesting to look at where the speakers are born, since e.g. Eastern Norway contains many more dialects than just one. An overview of which counties the speakers are born in can be seen in Table 3.4. It is worth noting that since this particular data is fetched from WikiData, its value is missing for a significant amount of speakers.

There are several factors that contribute to this dataset being selected for this project.

	Number of speakers
Agder	11
Innlandet	15
Møre og Romsdal	15
Nordland	11
Oslo	22
Rogaland	15
Troms og Finnmark	17
Trøndelag	20
Vestfold og Telemark	13
Vestland	29
Viken	36
None	63

Table 3.4: Speaker County Distribution.

Firstly, it is a considerably large dataset, containing speakers with different dialects and genders, which ultimately provides a diverse representation of the Norwegian language. Secondly, the dataset is manually annotated by professionals, which provides a guarantee of quality of the data. Finally, the dataset is publicly available and free to use, allowing other researchers to more easily replicate the experiments in this project.

3.2 Automatically Generated Transcripts

To produce the automatically generated transcripts needed for this project's experiments, the audio files included in the NPSC were run through a third party commercial service which provides speech-to-text conversion. The NPSC dataset contains one audio file per folder, meaning the resulting transcription will contain the same segment of sentences as the folder's respective JSON-file.

The files output from the speech-to-text system are .txt files containing all the transcribed text for its respective audio file in a single long document. The started of the document is denoted by some metadata providing information on the file's name, the provider of the transcript, i.e. the third party that generated it, and finally the date and time the transcript was produced.

After the first three lines of metadata, the main contents of transcriptions begin. The transcripts are divided on a sentence level, where each sentence is marked with a timestamp describing what time in the audio file the sentence corresponds to. When no speech is detected for a certain amount of time in the audio, the transcripts are marked with a timestamp and white space where the sentence usually would have been. These timestamps are seemingly added in intervals of circa 10 seconds.

As the transcripts from the NPSC dataset are stored as JSON objects and the automatically generated transcripts are stored in .txt files, some data processing tasks must be performed to get the transcripts in the same format. A thorough explaination of these steps are later described in section 4.1.

Chapter 4

Methods

The aim of this project is to create an alternate metric for evaluating speech-to-text systems that takes semantics into context. The commonly used Word Error Rate (WER) suffers from certain limitations, such as all errors weighing equally.

The metric, which we will call Semantic-WER, attempts to extend the WER by incorporating the semantic context of words. This is done by representing each word as an embedding and calculating the cosine distance between the two embeddings. Additionally, the metric is extended with several optional parameters allowing for more flexibility in its use.

The code for this project is uploaded to a public repository on GitHub[1].

4.1 Data Processing

As mentioned in chapter 3, this project utilizes an open-source dataset from the Norwegian Parliament containing audio files of speech as well as their professionally written respective transcriptions. These professionally written transcripts will be referred to as the “reference text” or as the “gold label transcripts” as of now. The audio files from the dataset are run through a 3rd party commercial speech-to-text service providing us with automatically generated transcriptions. These will be referred to as the automatically generated transcripts, or simply just transcripts..

Seeing as the transcripts provided by the third party commercial service are formatted differently from the NPSC data, some pre-processing and alignment must be done to facilitate for sentence-level comparison.

First, the generated transcripts need to be cleaned of certain elements, such as the metadata on the first lines, as well as the timestamps above sentences. This is done using regular expressions that find and remove timestamps. Additionally, regular ex-

[1]https://github.com/sfimediafutures/MA_Espen-James-Rodriguez-Stokke

pressions are used to remove unnecessary white spaces, shaping the document as one long paragraph.

To facilitate for alignment of the gold label sentences with the generated ones, we merge all the sentences from the NPSC dataset's JSON-file into one big document to match the shape of the generated transcripts.

As the manually annotated data from the NPSC dataset includes special tokens for hesitation and stuttering, these must be removed since they will prevent the transcripts from aligning properly. These special tokens include tokens for coughing which is represented as <*qq*>, as well as inaudible speech which is marked as <*INAUDIBLE*> Since these special tokens are marked with angle brackets, they can easily be removed with regular expressions.

Once both transcript documents have been cleaned, they are tokenized using regular expressions to divide the documents into lists of words and punctuation. Now that both transcripts are cleaned from unnecessary tokens and have equal shape, they can be aligned.

Once the two transcripts are both represented as two lists of strings, they can be aligned using the Needleman-Wunsch algorithm(*Likic*, 2008), which is an algorithm from bio-informatics that uses dynamic programming to align protein or nucleotide sequences. It provides an optimal alignment based on a scoring system that assigns values to matches, mismatches, and gaps. It obtains the resulting output by inserting gaps, which in our case is represented by an empty string.

The output of the algorithm was slightly modified for this use case to make the output be a list of lists-data structure, where each list contains two elements. The first element is the token from the generated transcript, and the second element is the token from the gold label transcript. If either of the transcripts is missing a token, the element will be an empty string.

Both transcripts are then cleaned of special tokens and white spaces, and ultimately reduced to two large documents of purely sentences. Then the documents are aligned using the Needleman-Wunsch algorithm, and output to a list of lists, where each list contains an automatically transcribed word at index 0, and the gold label word at index 1.

As the transcriptions had to be divided into single words for the sake of alignment, they must now be regrouped as sentences as this project aims to compare texts on a sentence-level. Seeing as we wish to evaluate the quality of the transcripts generated by the ASR system, we let the generated transcripts decide when a sentence comes to an end.

To merge the words into sentences, we iterate over our list of lists containing words, and whenever a token in the set of {`. ! ?`} is found, the preceding sequence of tokens is appended to a new list. This ultimately results in a new, similar list of lists where instead of containing transcript words and gold label words, the nested lists contain transcript sentences and gold label sentences.

To summarize, we have taken textual data in different formats, cleaned and pre-processed them to be of the same shape, aligned them, and finally segmented the words into sentences to facilitate sentence-level comparison. A diagram illustrating an overview of the entire process can be seen in Figure 4.1.

4.2 Implementation

For our metric, two main approaches were initially attempted. The first approach involved loading both a BERT-tokenizer and a BERT-model from HuggingFace[2] and then encoding sentences using the tokenizer. The input-IDs returned from the encoding are then fed into the BERT-model which will provide us with word embeddings. However, with this approach the BERT-embeddings contain special tokens which will have to be accounted for.

For the sake of simplicity, a second approach was explored where HuggingFace's pipeline function is utilized. By providing the pipeline function with the task of feature extraction and the name of the desired BERT-model, it will return a pipeline object that can take in a sentence and input and return a list of embeddings as output. As opposed to the first proposed approach, this method allows for retrieving embeddings in a single step, which makes for a more simplistic approach.

The metric itself is implemented as a function that takes two sentences represented as lists of words as input, as well as HuggingFace pipeline object. The idea behind taking the pipeline as input is so to ensure that the Semantic-WER function is model agnostic, making it a simple matter to change which BERT model is used for creating the word embeddings. This allows for experimenting with different models and comparing how they perform on the dataset.

In our dataset of post-processed sentences, all sentences are of the same length. However, the Semantic-WER function should allow for sentences of different length for several reasons. One of them being that speech-to-text systems may not pick up on certain words, or divide compound words into multiple words, which will result in a transcript that is different in length from the gold label text.

To deal with sentences of different lengths, our semantic-WER function calls upon another function to equalize the length of the lists. This helper function essentially takes the shortest of the lists and right-pads it, which means it adds empty strings to it

[2] https://huggingface.co/

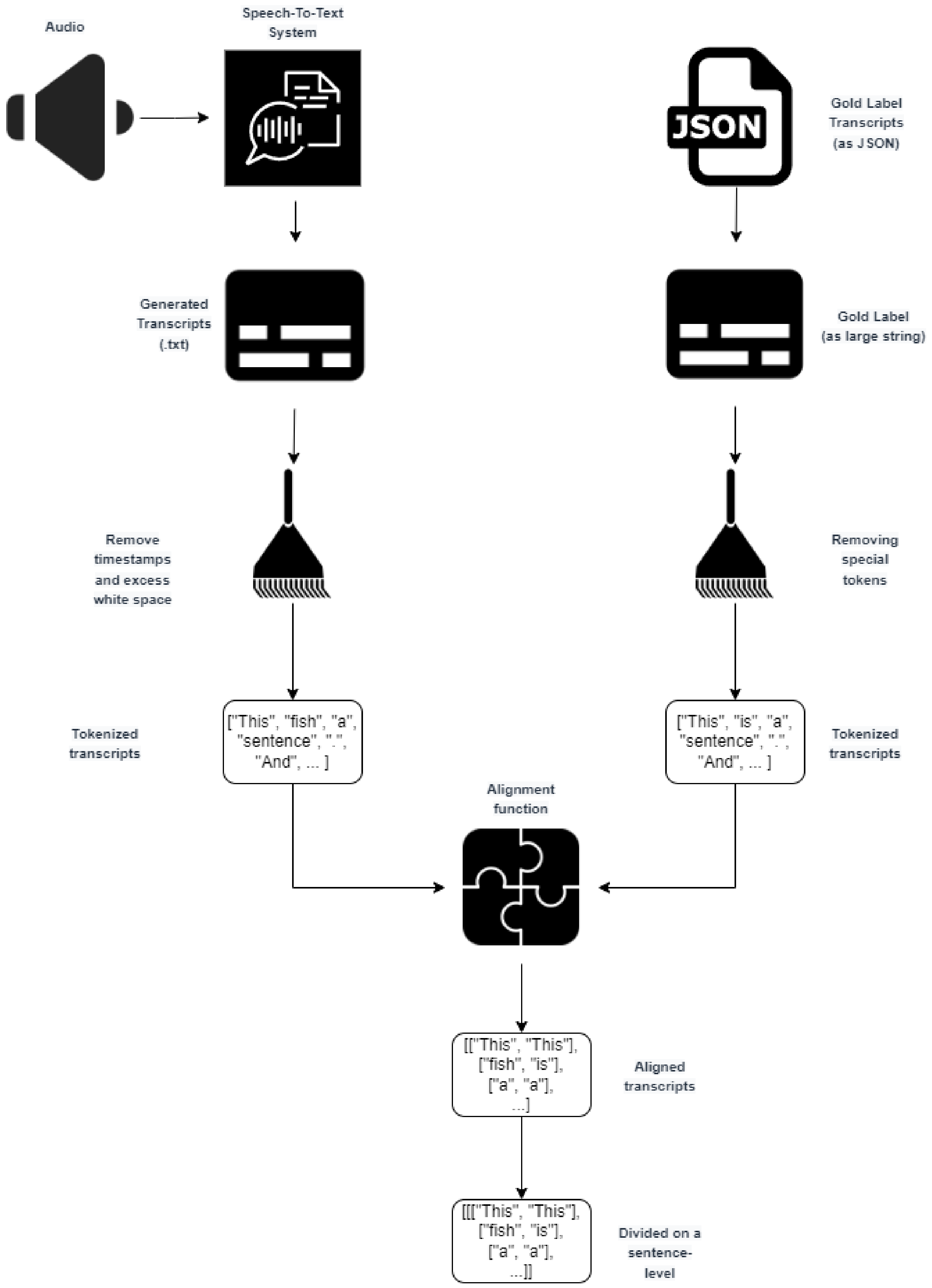

Audio
Speech-To-Text System
Gold Label Transcripts (as JSON)
JSON
Generated Transcripts (.txt)
Gold Label (as large string)
Remove timestamps and excess white space
Removing special tokens
Tokenized transcripts
["This", "fish", "a", "sentence", ".", "And", ...]
["This", "is", "a", "sentence", ".", "And", ...]
Tokenized transcripts
Alignment function
[["This", "This"], ["fish", "is"], ["a", "a"], ...]
Aligned transcripts
[[["This", "This"], ["fish", "is"], ["a", "a"], ...]]
Divided on a sentence-level

Figure 4.1: A diagram of the data pre-processing pipeline.

to acquire the same length as of the longer one. An example of the input and output of the right-pad function can be seen in Figure 4.2

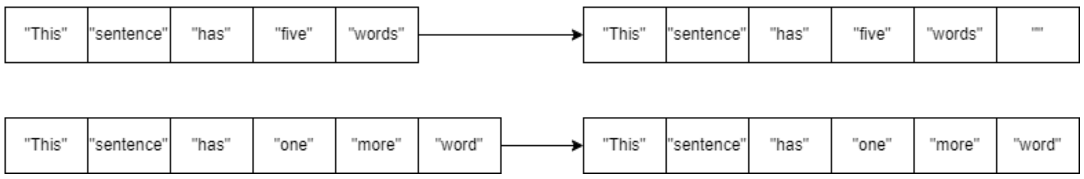


Figure 4.2: An example of input and output of right-pad function, showing how sentences are equalized in length

Optimally, the function would locate where in the sentence the missing word is and append an empty string at that location. This would help prevent the sentences from becoming unaligned. When the sentences are unaligned, the wrong words will be compared against each other adding additional semantic distance in the final score. However, correcting this potential error by detecting missing words is outside of the scope for this project and is left for future work. An alternate approach using S-BERT is explored later on in Section 4.2.1

The calculation of the Semantic-WER score is quite simple in itself, as it aims to build upon how Word Error Rate is calculated. First and foremost, the sentences are converted to lists of word embeddings using the model given as input. Then the lists of embeddings are iterated over simultaneously and cosine distance between each pair of embeddings is calculated using SciPy's[3] function for cosine distance and stored in a list. Finally, the list of distances is summed together and divided by the length of the first sentence. It is worth noting that this is the length of the sentence after padding has potentially been added, making it arbitrary if the first or second sentence is chosen as they are bound to be of the same length.

$$SWER = \frac{1}{n_1}\sum_{i=1}^{n_1} \mathrm{dist}(s_1[i], s_2[i]) \tag{4.1}$$

The equation for the Semantic-WER can be seen in equation 4.1. Note that n1 represents the length of the automatically generated transcript, while s1 and s2 represent the transcript and gold label sentence respectively, represented as embeddings. A flowchart illustrating the logic inside the Semantic Word Error Rate function can be seen in Figure 4.3.

As the NPSC dataset is divided into several batches of data, we initially test the function on a single batch of data and its speech-to-text generated transcript. Post-processing our data batch contains 553 sentences. Calculating the Word Error Rate on our post-processed data is a simple process, since with our list of lists representation

[3]https://docs.scipy.org/doc/scipy/index.html

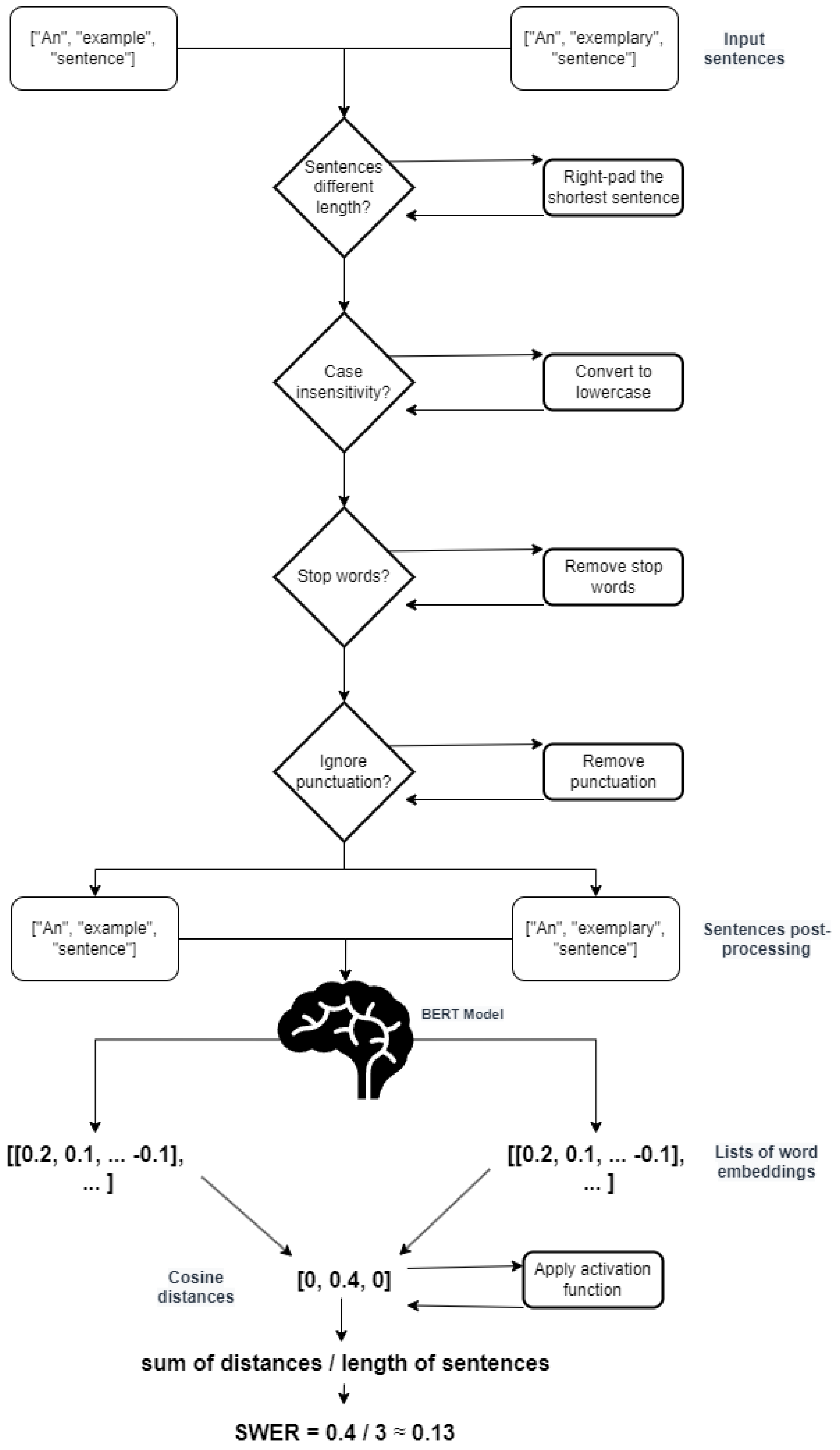

["An", "example", "sentence"]
["An", "exemplary", "sentence"]
Input sentences
Sentences different length?
Right-pad the shortest sentence
Case insensitivity?
Convert to lowercase
Stop words?
Remove stop words
Ignore punctuation?
Remove punctuation
["An", "example", "sentence"]
["An", "exemplary", "sentence"]
Sentences post-processing
BERT Model
[[0.2, 0.1, ... -0.1], ...]
[[0.2, 0.1, ... -0.1], ...]
Lists of word embeddings
Cosine distances
[0, 0.4, 0]
Apply activation function
sum of distances / length of sentences
SWER = 0.4 / 3 ≈ 0.13

Figure 4.3: A flow chart illustrating the logic of the Semantic Word Error Rate function

if the nested list contains two different words, then that will count as one error. This means we can iterate over our list, count errors, then finally divide by the length of the outer list. The Word Error Rate will later be used as an important baseline for comparison with our Semantic-WER.

As it is of great interest to discover how different models perform in different settings, all experiments are conducted using three different models, NorBERT[4], NorBERT2[5], and NB-BERT[6]. First and foremost, we explore the feasibility of including and excluding case sensitivity, stop words, and punctuation.

Secondly, we explore using the large multi-lingual model XLM-RoBERTa, as well as S-BERT, which can compute embeddings on a sentence-level for more than 100 languages. To limit the scope of this thesis, this is done without the implementation of our parameters.

As mentioned in chapter 3, the dataset is divided into several folders, where each folder represents one day of speech in the Norwegian Parliament. To get an idea of how our SWER-function initially performs, we run it once on a single folder for each of our selected Norwegian BERT models.

4.2.1 Additional Models

Mainly the experimentation focuses on Norwegian BERT models, as the transcriptions in the data set are in Norwegian. However, it is valuable to also test the metric when it is given multilingual models to compare the results. This way we can to some degree compare how the Norwegian BERT models perform in comparison to the multilingual ones. Additionally, one of the multilingual models we will use functions on a sentence level, which may deem different results as our other models operate on a word-level.

XLM-RoBERTa

As our main models are trained specifically for the Norwegian language, it is of interest to compare their results to a multilingual model. For that reason, we experiment with how the SWER metric performs given the multilingal model XLM-RoBERTa.

Due to the SWER-function treating BERT models as a plug-in component, this is simply achieved by switching which model we provide to the function with XLM-RoBERTa.

[4]https://huggingface.co/ltg/norbert
[5]https://huggingface.co/ltg/norbert2
[6]https://huggingface.co/NbAiLab/nb-bert-basehttps://huggingface.co/ltg/norbert

S-BERT

S-BERT, which converts entire sentences to embeddings, can prove to be a useful tool for comparison of the semantics of sentences. As our other language models take sentences and produce embeddings word for word, S-BERT will produce an embedding that represents the entire sentence. This may be beneficial, seeing as we will be comparing entire sentences at once, instead of comparing sentences word by word.

S-BERT provides a wide range of pre-trained models, including “distiluse-base-multilingual-cased-v2”, which works for 50 languages, one of them being Norwegian.

Although models specifically trained for Norwegian may perform better in certain tasks, S-BERT may perform better in our case since it manages to represent entire sentences at once. This solves an underlying problem with our implementation of SWER, which is that it assumes that the sentences are perfectly aligned. If the sentences are not aligned, our function for calculating SWER will be comparing the wrong words against each other. There also exists scenarios where the speech-to-text system generating transcripts misses one or more words, resulting in the generated transcripts eventually becoming misaligned somewhere in a sentence. This exact behaviour can be difficult to correct, making the approach of representing entire sentences as embeddings more feasible.

Since S-BERT works on entire sentences instead of single words, we need a different implementation than the one we have in our SWER function. However, this implementation is easily achieved by simply converting the sentences with S-BERT, and then computing the cosine distance between two sentence embeddings.

4.3 Parameters

Now that the main functionality behind computing the metric has been implemented, we extend it with several parameters to tweak its performance. These parameters are added to explore how they alter the behaviour of the metric, as well as to provide the user with flexibility. For the sake of experimentation, we can observe the performance of the metric compared to the standard WER after tweaking the parameters, which may provide insight into when our metric performs best. In addition, the parameters provide the user of the metric with the flexibility of choosing what the metric should emphasize.

The parameters consist of the possibility of including or excluding stop words, punctuation, case sensitivity, as well as applying an activation function to the cosine distance between word embeddings. These parameters will hopefully contribute to allowing the user to emphasize what they consider important in the weighting of semantics in sentences.

4.3.1 Case Sensitivity

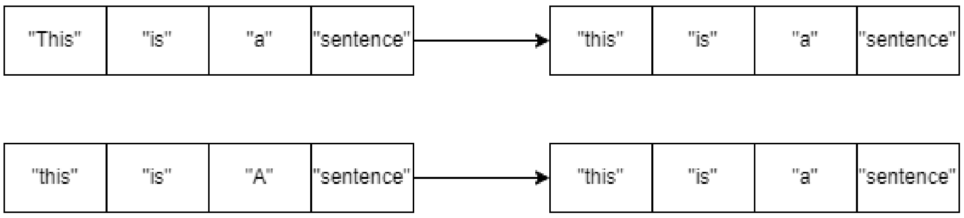


Figure 4.4: An example of how sentences are transformed when case insensitivity is toggled.

As the automatically generated transcripts contain a notable amount of capitalization errors, it is of interest to test the effect of making the SWER-function case insensitive. Therefore, we have added an optional parameter of case insensitivity that will convert all words in the sentences to lowercase when applied. An example of this can be seen in Figure 4.4.

The reasoning behind the implementation of this parameter is that words in the Norwegian language will rarely change meaning purely as a result of different capitalization. Some named entities may consist of an acronym that changes meaning when converted to lowercase. An example of this is "BIFF", which which is the name of the Bergen International Film Festival. When converted to lowercase, BIFF can not be differentiated from the Norwegian word for beef, "biff". Therefore, there are situations where case sensitivity should be taken into consideration to preserve the semantics of the sentences to its fullest extent.

However, in most cases the meaning of words remain the same independent of case, and upon investigation it can be observed that BERT-embeddings of the same word with different capitalization have a non-trivial cosine distance between them. Therefore it is of interest to add the possibility of opting out of case sensitivity, which should make the metric more focused on the pure semantics of sentences. Adding back case sensitivity can be valuable when one wishes for the metric to be stricter in the evaluation of the speech-to-text systems. As mentioned earlier, there are cases where capitalization may change the meaning of a word, so it should be an ultimate goal to keep the automatically generated transcripts unaltered.

4.3.2 Stop Words

To make the metric even more focused on the pure semantics of sentences, the option for removing stop words from sentences was implemented as well. As described in subsection 2.1.2, stop words are commonly used words in a language that are considered to have little semantic meaning and are often filtered out during natural language processing tasks, such as text analysis or search engines. These words include articles (e.g., "the," "a"), prepositions (e.g., "in," "on"), conjunctions (e.g., "and," "but"), and other frequently occurring words (e.g., "is," "are"). Removing stop words from a data set is often done when conducting sentiment analysis as the stop words do not contain any significant emotional weight(*Rosenberg*, 2014). For sentiment analysis this may often improve the results.

However, the value of removing stop words in the context of our metric is highly debatable. Even though stop words may not carry emotional weight or contain a large amount of semantic meaning by themselves, they do contribute to the grammar of sentences. For example, both the Norwegian words for “before” and “after” are included in the list of stop words. These words are opposite from each other and should result in a significant semantic distance in the metric. However, if stop words are removed then these two words appearing at the same index of two sentences being compared would not add any semantic distance to the final score of the metric. Arguably, this works against its purpose, but is included as a parameter anyway for the sake of experimentation.

The idea behind including the possibility for removing stop words is that the metric then will focus only on the more semantically heavy words. It may make the metric less useful for comparing entire sentences, but could still prove to be valuable for evaluating whether a speech-to-text system has transcribed the most important words accurately.

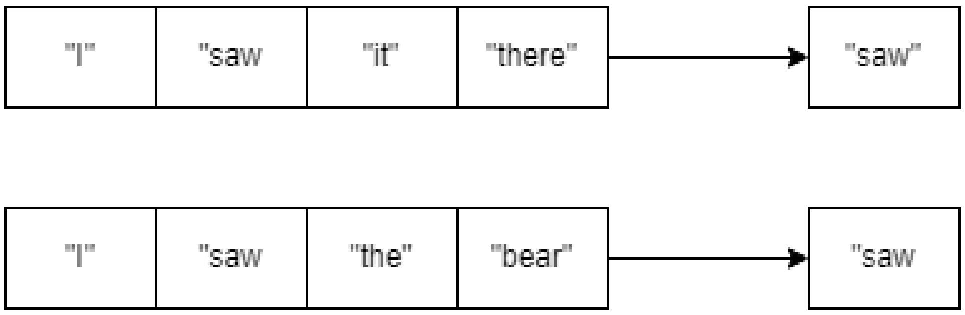


Figure 4.5: An example of two sentences after removal of stop words.

When it comes to the implementation of the stop word-parameter, the function iterates over the words in both the gold label sentence and the transcript sentence and checks if the word from either of the sentences exists in the set of stop words. If a stop word is found, it is removed from the sentence. Additionally, the word at the corresponding index from the other sentence is also removed. This builds on the assumption that if a word in a sentence is a stop word, then its respective word in the other sentence is also a stop word. An example where this may cause significant information loss can be seen in Figure 4.5, where the word “bear” is removed since its corresponding word is “there”, which is considered a stop word.

This assumption results in two potential errors. The first being that if a speech-to-text system incorrectly transcribes a word as one that is included in the list of stop words, the correct word is removed from the reference text even if it is not necessarily a stop word. The other potential error is that if the sentences are slightly misaligned, then several words may wrongly be removed. If an automatically transcribed sentence is misaligned by one word, and it has a stop word at *e.g.* index 2 that is supposed to be at index 3, then both words at index 2 and 3 will be removed from the transcripts and reference sentences. This may lead to situations where a large amount of words are wrongly removed which ultimately can lead to artificially low SWER scores.

As there are no official lists of stop words, it is important that the SWER function is flexible to what list of stop words it operates with. Therefore, the list of stop words is provided to the function as an argument. This helps make the parameter for stop words more flexible and can in turn make the function less prone to errors as mentioned above. Since the user provides the stop words, it gives the user the ability to filter out certain words, which could be useful in scenarios where the user knows that there are certain words that should be ignored.

4.3.3 Punctuation

Upon investigation, it appears that the cosine distance between the embeddings of punctuation symbols and the embedding of an empty string is rather large. As a result of this, missing punctuation in the automatically generated transcripts will often affect the final score quite significantly. For that reason, the option to exclude punctuation from the calculation of the score was implemented. How sentences are transformed by punctuation removal can be seen in Figure 4.6.

The idea behind this is similar to the one about removing stop words. However, excluding punctuation is arguably more sensible as it more rarely affects the meaning of a sentence. There are certain scenarios where punctuation can change the meaning of a sentence, such as the sentence "Lets eat, grandma", which drastically changes meaning upon removing the comma. Nevertheless, given context, most sentences are usually interpreted similarly regardless of punctuation.

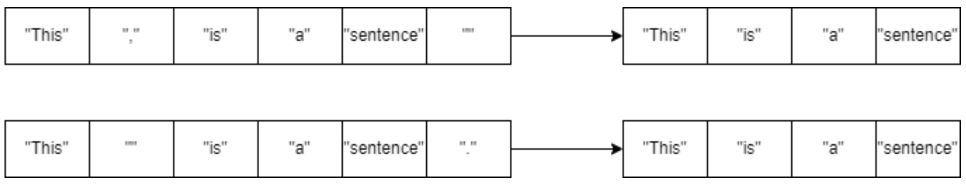


Figure 4.6: An example of how sentences are transformed punctuation is removed.

One reason for disregarding punctuation is that spoken natural language is significantly more free flowing than written language, and sentences in spoken language will often contain a lot more pauses, interruptions, stuttering, and similar. This makes it a difficult task to accurately punctuate sentences when transcribing spoken language.

Ultimately, punctuation still has an impact on the semantics of sentences and as mentioned earlier may drastically change the meaning of sentences in some scenarios. Therefore, the exclusion of punctuation is an optional parameter, as punctuation should be included if one wants to evaluate the quality of transcriptions in a strict manner.

4.3.4 Applying an Activation Function

Seeing as two words, no matter how similar, will have some cosine distance between their embeddings as long as they are not the same word, it is of interest to add the possibility of applying an activation function. This will allow the user of the metric to alter the values of the cosine distances before the final metric is computed. For example, if one wishes to disregard cosine distances below a value of 0.1, this can be achieved through supplying the metric with an activation function of own choice. Therefore, the possibility of providing the SWER function with an activation function was added as a parameter. Which activation function is to be used is up to the user to decide, as different activation functions may have different strengths and weaknesses.

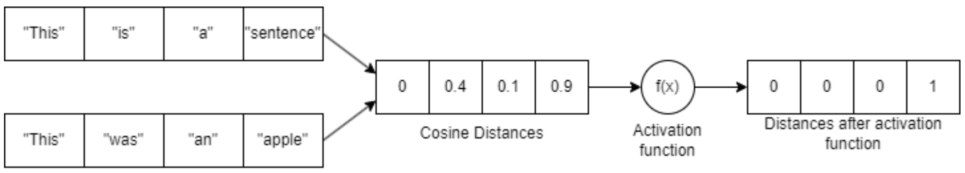


Figure 4.7: An example of how the cosine distances between two sentences can be transformed with an activation function. For this example, binary step is used.

Considering implementation, the SWER function will still function as usual by converting words to word embeddings and calculating the cosine distance. However, before calculating the final score, the function will run each individual cosine distance through the provided activation function and replace the cosine distance with the value returned by the activation function. Finally, the SWER score is computed as usual.

To test the effect of including an activation function, the simple, yet effective, binary step function was implemented and tested. The binary step function returns 1 or 0 depending on if the value given is above or below a certain threshold. The binary step function was here coded in a way such that the threshold can be altered, but defaults to a threshold of 0.5. The equation for the function can be seen in Equation 4.2.

$$f(x) = \begin{cases} 1, & \text{if } x \geq 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

The intuition behind applying the binary step to the cosine distances is that small values for cosine distance will be reduced to 0, ultimately making the metric not punish words that are considered “close enough” in meaning. On the other hand, cosine distances equal to or greater than the threshold of 0.5 will be increased to 1, which means all words that are considered “far enough” from each other in meaning will be punished more heavily.

4.4 Severity analysis

As of now the SWER function acts as a kind of “black box” where sentences are fed into the function and a score is returned without much explanation as to how or why the score is what it is. As the words are converted to word embeddings based on pre-trained models, there are some results that may appear peculiar and unexplainable for humans. Additionally, the score provided by SWER is dependent on the model provided, which makes the metric subjective in a sense.

In contrast, the regular Word Error Rate metric is calculated by following a very specific set of steps, which will always return the same score for two sentences. SWER will also always return the same score for two sentences, but as mentioned earlier, that requires that the same BERT model is used.

In an attempt to make the SWER metric more explainable, a parameter called `print_severity` was added, which will print the words in the two sentences that result in the largest cosine distances. To achieve this, SWER was extended with two optional parameters, one being `print_severity` which is by default set to `False`. The other parameter being `n_severity` which defines the amount of severe words one wants to display. By default, *n_severe* is set to 3, meaning the three most distant words will be printed if `print_severity` is set to `True`.

Printing the most severe words will aid in providing insight into why the score is as it is, especially in situations where the score is considerably high. In these scenarios it can be valuable to observe which words are critically different.

Chapter 5

Evaluation

As the implementation of the semantic word error rate is completed, it is of great interest to evaluate how the metric performs in different scenarios. To evaluate the performance of a metric itself may prove to be a difficult task, which is why qualitative and quantitative approaches are combined in an effort to produce a full image of the metric's strengths and weaknesses.

It is worthy to note that the semantic word error rate does not aim to replace the currently used metrics, but rather function as an additional tool used when assessing the quality of ASR transcripts. An example of when combining different metrics may be useful, is when a sentence has a low word error rate, while having a high semantic word error rate. This could indicate that even though the transcript contains few errors, those errors are quite severe. On the contrary, a high word error rate and a low semantic word error rate could indicate the presence of several less severe errors.

As these examples illustrate situations where our metric functions best in combination with others, this chapter attempts to explore how the metric performs on its own. This includes methods such as testing the metric on random sentences, as well as some artificially constructed sentences. Additionally, our metric will be tested for how much it correlates with word error rate, and humans' perceptions of the semantic distances between sentences. This is achieved by gathering data from a survey where respondents manually rate the semantic distances between sentence pairs.

5.1 Semantic Word Error Rate Scores

As mentioned in chapter 4, the function that calculates the semantic word error rate takes a BERT model as input, making it easy to experiment with how different BERT models perform. For most tasks, experimentation will be performed with the three current main BERT models, NorBERT1, NorBERT2, and NB-BERT. When the metric is tested without any of the optional parameters, the results will also be compared against how the metric performs using S-BERT and XLM-RoBERTa.

The experiments are conducted on a smaller batch of the NPSC dataset, namely one folder of the NPSC dataset, which contains one day of recordings from the Norwegian parliament. This limitation of data is an unfortunate consequence of the fact that the transcription service the recordings were run through took a considerable amount of time to run due to the length of the recordings, as well as the service frequently crashing and interrupting transcribing, ultimately rendering the produced data useless.

The data used[1] contains 553 sentences. This sample size is rather small, however, the data fortunately consists of a balanced mix of sentences transcribed both correctly and incorrectly providing us with a sufficient range of quality of transcribed sentences to experiment with. All sentences are stored in a list ordered chronologically.

For each of the three Norwegian BERT models, as well as S-BERT and XLM-RoBERTa, the semantic word error rate (SWER) function is run on all 553 sentences with the default parameters. For every model, the score for each sentence is stored in a python dictionary and then saved as a `JSON` file. The scores are ordered, meaning the index of a score belongs to the sentence at the corresponding index in the list of sentences.

5.1.1 Parameters

The process of computing and saving scores is performed several times with different parameters. The first parameter tested is setting the *case insensitivity* parameter to `True`, which provides us scores that allow us to observe the effects of ignoring case.

The next set of scores is produced using the *activation function* parameter while preserving case sensitivity for the sentences. As the SWER function takes the activation function it uses as input, we must provide it with a function of our choice. For this set of experiments, we provide the SWER metric with the binary step function. As mentioned in subsection 4.3.4 and seen in Equation 4.2, the binary step function returns the value `1` if the input is above or equal to some given threshold, and `0` if the input is below the threshold. The reasoning behind choosing binary step for activation function is that the intuition behind the function is understandable and its behaviour is explainable. Although other activation functions may produce interesting results, the results produced may be more complicated to explain.

Further, the sentences are run through the SWER function with the option for *removing punctuation* toggled and then finally, the SWER scores are computed with the option for *removing stop words* as well. When it comes to stop words, there is no universally agreed upon list of Norwegian stop words, which is why the SWER function

[1] From folder “20170207”.

takes the list of stop words to be used as input. For now, a list of Norwegian stop words has been somewhat arbitrarily selected, after some initial manual analysis. The list used is fetched from a public repository on GitHub [2].

Now the SWER scores have been computed with every parameter tested individually, facilitating the observation of the effect of each parameter on the scores on its own.

5.1.2 Combined parameters

Since all parameters have now been tested individually, it is of interest to test how they work in combination. Testing all possible combinations of parameter options would quickly become an extremely computationally expensive and time consuming task, therefore a certain set of parameters have to be selected to be tried in combination. For this project, two combinations of parameters have been selected for experimentation.

The first set of parameters to be tested together is case sensitivity set to `True`, in combination with the exclusion of punctuation. The intuition behind this combination is to test the performance of the SWER metric with a pure focus on the textual content of the sentences. The pros and cons of ignoring case sensitivity and punctuation are discussed in section 4.3.

The second set of parameters to be tried in combination is the same as the previous, but extended to include the use of an activation function. The activation function applied here is the binary step function with a threshold of 0.5. The idea is similar to the previous combination of parameters where we focus on the textual contents of the sentences, however cosine distances between words that are lower than 0.5 will be ignored, and those above 0.5 will be increased to 1. In essence, this means the SWER metric will look at the words in the sentences and count the ones that are significantly different in meaning, while ignoring punctuation and case.

5.2 Grouping by Severity

When evaluating the results of testing the metric on all our data, it can be useful to have the results grouped by score, allowing us to easily explore which sentences have produced a high score, which have produced a medium high score, and which have resulted in a low SWER score. Therefore, a simple function that groups sentence based on their SWER scores was implemented.

[2]https://github.com/Alir3z4/stop-words

Group	SWER range
Low	< 0.15
Medium	> 0.15 & ≤ 0.30
High	> 0.30

Table 5.1: Default thresholds for severity grouping function.

The function takes a list of sentence pairs and their SWER scores as input, as well as an optional parameter of thresholds, which is an array-like object containing two thresholds. The first threshold is the "low" threshold, meaning any sentence with a SWER score below this number will be placed in the group of sentences with a low score. The second threshold is the "high" threshold, meaning any sentence with a score above this is placed in the group of sentences with a high score. All the sentences with a score between the two thresholds is placed in the medium group. If no thresholds are provided to the function, it defaults to a low-threshold of 0.15, and a high-threshold of 0.3. These exact thresholds were chosen to divide the sentences in the dataset in such a manner that the resulting medium and high groups are roughly equal in size.

The implementation of the function itself is quite straightforward as it simply iterates over the list of sentence pairs and the list of SWER scores simultaneously and appends them to different lists based on whether their scores are above or below the thresholds. This implementation builds on the assumption that the indexes of the sentence list and the scores list match. An alternate approach would be to call the SWER function inside the grouping function, requiring only the sentence pairs and thresholds as input. However, this would in most cases result in additional unnecessary computation, as we already have computed the scores for the sentences.

The function returns three lists representing our groups. Each list contains tuples, where each tuple contains two elements. The first element being the sentence, and the second one being its SWER score. When the sentences are divided into groups of severity for all parameters and models, the groups are exported to a JSON file for later use.

5.3 Qualitative Sentence Comparison

So far, the evaluation of the metric has been largely based on testing the entire set of data and averaging the scores. To get a more comprehensive understanding of the metric's behaviour, qualitative research should be conducted on certain sentences.

To aid us in investigating certain sentences, a function called "*test_swer*" was created. This function takes a list of sentence pairs as input, as well as a BERT model.

For each sentence pair it prints out both sentences, its SWER with default parameters, its SWER with binary step, its case insensitive SWER, and its SWER with punctuation excluded. Additionally, the WER and case insensitive WER for each sentence is printed. If a list of stop words is provided to *test_swer*, it will also print out the SWER for the sentences when the words from that list are excluded.

This function is meant to serve as a tool for quickly providing a report on a set of sentences. As SWER is intended to be a metric used in addition to the traditional word error rate, it is of great use to see their results side by side.

5.3.1 Comparing Random Sentences

As with the regular word error rate, SWER will provide a score of 0 if two sentences are identical. However, since our metric is based on cosine distance between word embeddings, the metric itself will very rarely reach a score of `1`, as there will in most cases be some similarity between the word embeddings. This means that completely unrelated sentences that share no words in common whatsoever will often result in a score lower than `1`.

So far, the metric has been tried on sentences that are supposed to be similar. However, it is also important that the metric produces a high score for sentences that are completely unrelated. Therefore, a function that selects random sentences and compares them was developed. The function, called "*test_n_random_sents*", takes a number *n* and a BERT model as input. It then selects *n* pairs of random sentences from the list of all sentences and uses our previously mentioned "*test_swer*" function.

This allows us to observe the word error rate and the SWER score with different parameters for two arbitrarily selected sentences. The intention behind this is to provide better insight into how the scores relate to each other. An example of two random sentences selected by the function can be seen here:

Random Transcript 1:
Den gir en unik innsikt i det å både være foreldre.

Random Transcript 2:
men problemet er volum og kapasitet.

For each Norwegian BERT model we have, we test 1 random sentence pairs, resulting in 3 sentence pairs being tested. Due to the nature of randomness, sentences that are somewhat similar or related may be included, but 3 sentence pairs in total should produce a meaningful depiction of how the metric evaluates unrelated sentences.

5.3.2 Artificially Produced Sentences

In addition to the comparison of random sentences, a set of 3 sentence pairs was artificially produced. These sentences were produced to test how the metrics perform on a specific set of sentences that explore different edge cases, as well as regular cases.

In similarity with the idea of testing random sentences, the intention here is to challenge the metric, explore its limitations, and potentially highlight some strengths as well. An example of a sentence pair in our mock data can be seen here:

Mock sentene 1:
Det var virkelig en bra dag.

Mock sentence 2:
Det var ikke en bra dag.

This example aims to highlight a weakness with the metric, as it will likely result in a fairly low SWER. However, the sentences are quite clearly far apart semantically as the sentences mean the opposite from each other. This behavior can often be attributed to antonyms being used similarly in very similar contexts.

5.4 Detecting the Most Severe Words

When the SWER metric produces high scores, it is of interest to inspect what contributed to such a magnitude of score. Using the parameter described in section 4.4, we can feed sentences to the SWER function and inspect what words contributed the most to the final score. This way we can develop a better understanding of why the sentences scored as high as they did.

Now it is fortunate that we have grouped our sentences earlier by severity, as we can take the sentences previously grouped as having a high score and test these for most severe words. This will help us in understanding what the metric considers semantically distant. For a select set of sentences, we print the three words that contribute most to the final error rate.

It is also interesting to view what contributes to the final score on the sentences we have grouped as “low” or “medium” severity. However, it is quite likely that the greatest contributors in semantic distance between these sentences are things such as punctuation and wrongly capitalised words.

5.5 Correlation

To further develop an understanding of the SWER metrics behaviour, we calculate its correlation with two other metrics, the commonly used word error rate, as well as the human word error rate, which is a metric we create based on human opinion of semantic distance between sentences.

There are several different correlation coefficients we may use, such as Pearson's correlation coefficient, Spearman's rank correlation coefficient, and Kendall's rank correlation coefficient (*Kendall*, 1938; *Pearson*, 1895; *Spearman*, 1904). Which metric is most appropriate depends on the distribution and nature of the data. Some things to take into consideration is if the relationship between the variables are linear or not, whether the data contains outliers, and finally, the sample size of the data.

For our data, the relationship between the variables should be non-linear, as the intention of the SWER metric is to provide an alternate measure to the commonly used WER. The range of our SWER metric is from `0` to `1`, which means we do not have to deal with the presence of outliers. Finally, our sample size is considerably small. These factors combined should be considered when choosing the correlation metric to be used.

5.5.1 Word Error Rate Correlation

To gain some insight into how our SWER metric relates to the regularly used word error rate, we calculate the correlation between the two metrics' scores on our dataset. It is important to note that the point here is not to obtain perfect correlation between the two metrics, seeing as they measure two separate things.

The correlation is measured by computing the WER and SWER scores for our sentences, and then calculating the Kendall rank correlation coefficient(*Abdi*, 2007) between the scores using SciPy's kendalltau function. The Kendall rank correlation coefficient was chosen as our correlation metric as it works well with small sample sizes. The Kendall rank correlation coefficient returns a number between `-1` and `+1`, where `+1` indicates a perfect positive association and `-1` indicates a perfect negative association. A positive relationship between two variables means as one increases, the other increases as well, while a negative relationship means as one variable increases, the other decreases.

The correlation between the SWER and WER scores is calculated once for each BERT model, and for each BERT model the correlation is calculated once for each parameter or parameter combination.

5.5.2 Human Word Error Rate

As our metric attempts to represent how far apart sentences are in meaning, it should in some way be evaluated against the opinions of humans. Inspired by the work done in section 2.5, a survey was created using Microsoft Forms to produce some data of the human perception of semantic distances between the sentences in our dataset.

The survey consists of 29 pairs of sentences, where the respondents were presented with one sentence pair at a time and asked to score them on how they perceived them to be semantically similar. Each sentence pair consists of one sentence from the NPSC dataset and its corresponding sentence from the automatically generated transcripts. For each sentence pair the respondent must select a score ranging from 1 to 5 stars, where 1 star indicates the sentences are completely unrelated in meaning and 5 stars means the sentences are identical in meaning. An image of the survey can be seen in Figure 5.1.

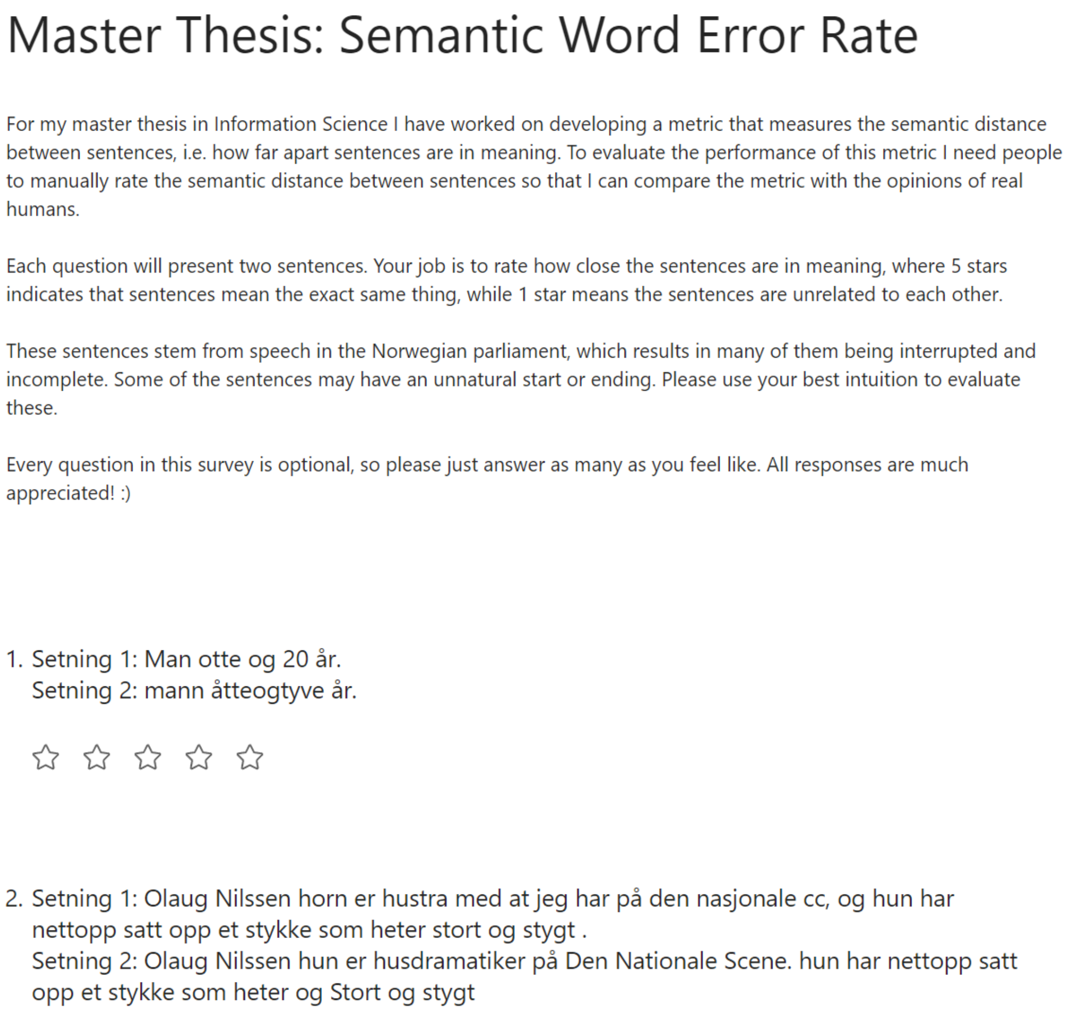
The sentence pairs in the survey are presented in random order, and the user may answer as many of them as they prefer. The respondents are asked to use their best intuition for scoring the sentence pairs, meaning the scores are prone to subjectivity as the respondents perception of the scale may differ.

The survey received a total of 40 responses, averaging 20.4 sentence pairs rated sentence pairs per response, making up a total of 817 ratings. Due to the sentences being presented in random order, all sentence pairs received ratings. The amount of ratings receiever per sentence pair can be seen in Table 5.2. An overview of which sentence pairs belong to which IDs can be found in section A.2.

As the data has been collected, the average score for each sentence pair is manually fetched and stored in a JSON file. The survey scores are loaded into a Python notebook with the semantic word error rates for the corresponding sentences.

Since the survey scores range from 1 to 5, they are all multiplied by 0.2, to alter their range to be from 0 to 1, making the scores' range match the range of SWER. Furthermore, we must address the fact that the survey scores currently measure the percentage of similarity, while SWER measures the distance. Therefore we must calculate the complement of the survey scores, so that they too measure the distance. This is done by taking each score and subtracting it from 1, which essentially inverts the metric from being a "human word correct rate" to a human word error rate. The equation for converting the survey scores to human word error rate can be seen here:

$$HWER = 1 - (\text{SurveyScore} \times 0.2)$$


Master Thesis: Semantic Word Error Rate

For my master thesis in Information Science I have worked on developing a metric that measures the semantic distance between sentences, i.e. how far apart sentences are in meaning. To evaluate the performance of this metric I need people to manually rate the semantic distance between sentences so that I can compare the metric with the opinions of real humans.

Each question will present two sentences. Your job is to rate how close the sentences are in meaning, where 5 stars indicates that sentences mean the exact same thing, while 1 star means the sentences are unrelated to each other.

These sentences stem from speech in the Norwegian parliament, which results in many of them being interrupted and incomplete. Some of the sentences may have an unnatural start or ending. Please use your best intuition to evaluate these.

Every question in this survey is optional, so please just answer as many as you feel like. All responses are much appreciated! :)

1. Setning 1: Man otte og 20 år.
   Setning 2: mann åtteogtyve år.

   ☆ ☆ ☆ ☆ ☆

2. Setning 1: Olaug Nilssen horn er hustra med at jeg har på den nasjonale cc, og hun har nettopp satt opp et stykke som heter stort og stygt .
   Setning 2: Olaug Nilssen hun er husdramatiker på Den Nationale Scene. hun har nettopp satt opp et stykke som heter og Stort og stygt

Figure 5.1: Screenshot of the Human Word Error Rate survey

Sentence ID	No. of Responses
1	25
2	29
3	30
4	29
5	25
6	28
7	27
8	27
9	29
10	28
11	29
12	25
13	26
14	26
15	26
16	29
17	27
18	24
19	30
20	27
21	27
22	26
23	30
24	26
25	25
26	27
27	26
28	28
29	29

Table 5.2: Number of responses each sentence pair received

Continuing, each sentence pair's SWER for each Norwegian BERT model is compared to its HWER. Additionally, the Kendall rank correlation coefficient between the SWER score and the human error rate score is calculated for the 29 sentences. However, due to the small sample size, the correlation between the two metrics may not provide much insight. Therefore it may be more insightful to investigate the resulting scores on a sentence level, than simply looking at a correlation coefficient.

Previously with the regular WER, the correlation between the SWER scores and the Word Error Rate was calculated for all combinations of parameters and BERT models. For Human Word Error Rate this will not be done. Instead the correlation will just be computed between Human Word Error Rate and SWER with default parameters. This is because the default parameters of our metric can be considered as the strictest evaluation of semantic distance, which is what we want to compare the human opinion with.

Chapter 6

Results

In the previous chapters, we presented the dataset, the methods for data processing, as well as the methodology behind our proposed metric called Semantic Word Error Rate, which is a metric that aims to quantify the semantic distance between two texts. Additionally, chapter 5 presented our approach for evaluating the performance of the metric. This chapter aims to present and analyse the results obtained from utilising this metric, as well as discuss the implications that follow.

These results are derived from a series of experiments designed to test the effectiveness and reliability of SWER in a variety of scenarios, as well as compare its performance to the results of the already existing WER metric. The SWER metric is also compared to human opinions of semantic distance through data gathered by a survey.

Through a combination of quantitative analysis and qualitative observations, we seek to demonstrate the utility of SWER in the area of evaluating the quality of speech-to-text systems. The presentation of these results intend to facilitate a comprehensive understanding of the metrics' capabilities, its potential applications, and how these are effected by our implemented parameters.

6.1 Model Performance

After computing the SWER for all 553 sentence pairs with three Norwegian BERT models as well as two multilingual models, the average semantic word error rates were calculated and can be seen in Table 6.1. As mentioned in subsection 4.2.1, the multilingual S-BERT model converts entire sentences to embeddings. In contrast, all other models used here produce embeddings on a word level. The error rates here are computed with the default SWER parameters, meaning the texts are not manipulated in any manner other than the words being converted to word embeddings.

For the sake of comparison, the average word error rate for all sentence pairs is included, as well as the average case insensitive word error rate.

	SWER
NorBERT1	8.39%
NorBERT2	11.02%
NB-BERT	12.78%
S-BERT	6.54%
XLM-RoBERTa	1.01%
Regular WER	22.74%
Case insensitive WER	14.82%

Table 6.1: Average BERT model SWER & WER on our dataset.

The model with the lowest average SWER for all sentences is XLM-RoBERTA by a great amount with an average error rate of 1.01%. The model with the highest average is NB-BERT, with an average semantic word error rate of 12.78%. The two other Norwegian models are relatively close in performance as NorBERT1 & NorBERT2 have SWERs of 8.39% and 11.02% respectively. S-BERT achieved a SWER of 6.54%, which is significantly higher than the other multilingual model XLM-RoBERTA, while also a considerable amount lower than the Norwegian models.

In comparison, the WER for the same set of sentences is 22.74%, which is significantly higher than all the models' SWER. This could be an early indication that SWER punishes errors less harshly resulting in lower error rates in general. However, it is also worth noting the case insensitive WER for these sentences is 14.82%, meaning a lot of the errors punished by the regular WER comes from errors such as capitalisation. Still, all models return an average SWER lower than the case insensitive WER.

One model that stands out particularly, is XLM-RoBERTa which produced an average error rate that is remarkably lower than all the others. There may be several factors contributing to this, one being that too many words are out of vocabulary for this multilingual model, resulting in words with erroneously low cosine distances between them. The other multilingual model, S-BERT produces a more reasonable average SWER. However, this may be due to its nature of creating sentence embeddings. This aspect will be revisited in subsubsection 7.1.3.

6.1.1 Effect of Parameters

Now that the behaviour of our metric using its default parameters has been established with all our BERT models, the effect of applying the different parameters may be explored. Note that the multilingual models are not included in these further experiments. All the parameters make the metric ignore certain errors in some way, meaning that these can be viewed as different approaches to making the metric less strict. Additionally, two slightly different combinations of parameters were tested to observe the

	NorBERT1	NorBERT2	NB-BERT
Default	8.39%	11.02%	12.78%
Case Insensitive	5.64%	8.71%	10.73%
No Punctuation	7.30%	7.68%	9.29%
With Stop Words	9.29%	13.65%	15.96%
With Binary Step	3.58%	12.13%	12.13%

Table 6.2: Parameter effect on SWER.

resulting SWER scores when parameters are combined.

The average SWER for each parameter and each BERT model has been computed for our 553 sentence pairs, and the resulting error rates can be seen in Table 6.2.

Case Insensitivity

The first parameter was a natural inclusion as the metric aims to focus on the semantics of the textual content. As argued in subsection 4.3.1, capitalisation will rarely effect the semantics of words, except for some cases such as named entities where the capitalisation changes what the word refers to or means.

Upon investigation, it can be observed that with NorBERT1 the cosine distance between the word embedding for "Dessverre" and the word embedding for "dessverre" is 0.32, which is considerably large given the semantic difference between the two, which is arguably none. Therefore it is sensible for a metric that focuses on the semantics to provide an option for case insensitivity.

As one may see in Table 6.2, when case insensitivity is applied, all models' average SWER decrease by roughly 2%. The reasoning behind this notable decrease can likely be attributed to the amount of capitalisation errors generated by the speech-to-text system. These capitalisation errors may in turn be caused by errors in punctuation, which is a likely error as speech-to-text systems often struggle with properly identifying when sentences end.

Punctuation Exclusion

The option for excluding punctuation from the texts when computing the metric is quite similar in motivation as the option for case insensitivity. As mentioned, speech-to-text systems may often struggle with identifying the ends of sentences, nonetheless where commas should be placed.

Between the word embedding for "," and the word embedding for a blank string, there is a cosine distance of 0.34, meaning missing commas are noticeably punished within the metric. Other punctuation symbols result in similar cosine distances, which become even larger if the symbols are compared to words. However, it is arguable whether this behaviour is unwanted. As argued in subsection 4.3.3, the meaning of a sentence can in fact be altered drastically by its punctuation.

The results in Table 6.2 show that the average SWER for all models are reduced by roughly 2% to 3.5%, meaning that this parameter contributes significantly to making the metric behave less strictly. Whether this behaviour is desired depends on the user's opinion on the importance of punctuation, as this parameter makes a trade-off between weighing grammatical accuracy and focusing purely on the semantic content of the words in the sentences.

Stop Words

Whether removing stop words is sensible for a metric focused on evaluating transcripts is debatable seeing as the input text is manipulated and altered considerably when doing so. This was discussed further in subsection 4.3.2, where it is demonstrated how words that are usually included in the lists of stop words may alter the meaning of sentences significantly. However, the experimenting with this parameter produce some interesting results.

As seen in Table 6.2, the average SWER increases for all models when enabling the parameter for removing stop words. This may seem counter-intuitive as removing words should result in less errors, however, what is happening is that for each word that is removed, the length we divide the sum of cosine distances by is reduced by 1. This results in errors from words that are not considered to be stop words will weigh more heavily in the final score, as a consequence of the sentence length the sum of distances is divided by is reduced by 1 for each stop word pair removed from the transcript.

Even though the average SWER for all models increase, it would not necessarily be correct to state that this parameter makes the metric more strict, as words are removed from the input making the metric ignore certain grammatical aspects. Nonetheless, applying this parameter may have altered the metric to be a better indicator of the presence of severe errors in the transcript, as the focus is shifted towards the words that weigh more heavily.

Another thing worth noting, is that with the stop words parameter, NorBERT2 and NB-BERT experience a noticeably larger increase in average SWER than NorBERT1. This could stem from several reasons, one perhaps being that NorBERT2 and NB-BERT produce lower cosine distances between words that do not carry much semantic meaning, i.e. the words we have considered to be stop words.

Binary Step

Our final parameter is the possibility of adding an activation function, which we experimented with by applying the binary step function, a function that increases values to 1 or decreases them to 0, depending on whether they exceed the given threshold. Applying other activation functions could yield interesting results, however, such experimentation was outside the scope of this project.

Looking at the performances of the different models in Table 6.2, one can see that applying the binary step function to the models alters the result quite differently between them. The first thing to stand out is that the activation function greatly reduces the average SWER for NorBERT1, which could be interpreted as NorBERT1 rarely producing embeddings with a cosine distance equal to or greater than 0.5, making most cosine distances being reduced to 0 within the SWER function.

On the contrary, NorBERT2 experiences an increase slightly above 1% on average SWER. This means that, generally, values are being increased more than decreased by the binary step function. Our final model, NB-BERT, experiences a slight decrease of 0.65%, which may be interpreted as the binary step function not making a significant impact on the final average error rate.

To further develop an understanding of how the binary step function behaves inside the SWER function, some test sentences were run through SWER with the binary step function. An interesting example can be seen here:

Automatically generated transcript:
Det opereres med ulike tall på forekomst av autisme, men cirka 1 % av befolkningen har autisme.

Gold label:
det opereres med ulike tall på forekomst av autisme men cirka én prosent av befolkningen har autisme.

This automatically generated transcript has 4 differences from the gold label, which divided by 19 tokens results in a word error rate of 21.05%. The four differences in the transcripts are the capitalisation of “Det”, the presence of a comma, “1” as a number instead of “én”, and the “%” symbol instead of the corresponding word for the symbol.

The semantic word error rate for this sentence pair is 6.11%, which is an arguably more representative error rate for these sentences than the significantly higher word error rate. However, one may even argue that the semantic word error rate for these equal to or close to 0, seeing as the sentences are arguably identical in meaning.

	Parameters
Combination 1	Case insensitive, no punctuation
Combination 2	Case insensitive, no punctuation, binary step

Table 6.3: Parameter combinations.

This is where the binary step function produces promising results. As all the word embeddings for all the differences in the transcripts produce cosine distances below 0.5, they are reduced to 0 by the binary step function, ultimately producing a SWER of 0.00%. Though, it is important to emphasise that this is simply one sentence-pair and this behaviour may not always serve as optimally across other sentences in the dataset.

An interesting result is the fact that the binary step function also worked as an additional way to deal with capitalisation and punctuation errors without removing punctuation or converting the transcripts to lowercase. This could especially work as a solution for the named entity issue mentioned in subsection 4.3.1, where our example word "BIFF" changes meaning entirely when converted to lowercase.

As we have seen that capitalisation errors and missing punctuation usually results in a cosine distance of roughly 0.3, we could attempt to prune these out by ignoring cosine distances below a threshold such as e.g. 0.35. However, increasing values above this threshold to 1 would result in many cosine distances being increased greatly. Therefore, it could be an idea to implement an activation function inspired by the Rectified Linear Unit (ReLU) activation function. A potential function such as this can be seen here:

$$f(x) = \begin{cases} 0 & \text{if } x < 0.35 \\ x & \text{if } x \geq 0.35 \end{cases}$$

For this function, if a value is below 0.35, it is reduced to 0. Otherwise if the value is above that threshold, the value remains unchanged. This is purely an example of an activation function that could be interesting to experiment, and there are likely numerous other activation functions that can produce more promising results.

Combined Parameters

As the results of applying the parameters individually have been presented, it is of interest to look into how they performed in combination. For this, two combinations of parameters were crafted, which we will name "Combination 1" and "Combination 2". Their parameters can be seen in Table 6.3. The results of the parameter combinations are presented in Table 6.4.

	NorBERT1	NorBERT2	NB-BERT
Combination 1	3.95%	4.94%	6.76%
Combination 2	2.40%	6.08%	6.46%

Table 6.4: Combined parameters average SWER.

For the first combination of case insensitivity and exclusion of punctuation, the average SWER is drastically reduced for all models. A motivation for this exact combination of parameters was that a punctuation error will often lead to a following capitalisation error, meaning these errors can ultimately effect the final score significantly.

The error rates from this combination can be viewed as representing the semantic distance purely between the words of the sentences. This combination could be beneficial in scenarios where one wishes to evaluate sentences in a less strict manner, making the metric aim its focus on the words contained in the sentences.

The second combination is the same as the first one, but extended with applying the binary step function. As we previously explored in subsubsection 6.1.1, the binary step function already contributes to removing punctuation and capitalisation errors from the final score, which in turn makes it interesting to see how binary step effects the average SWER when capitalisation and punctuation already has been accounted for.

The error rate for combination 2 in Table 6.4 show that for NorBERT1 the average SWER evidently becomes quite low with an average of 2.40%, which is considerably lower than NorBERT1's result for combination 1. For this combination of parameters and this model it appears that most sources contributing to the metric have been filtered out, which is not desirable for a metric intended for evaluation. In this case, the metric has arguably become too lenient.

For the two remaining models, the addition of the binary step function exhibits behaviour quite similar to when it was introduced by itself, as it increases the average SWER slightly for NorBERT2 and decreases the average SWER slightly for NB-BERT. This combination of parameters seems to give more promising results with NorBERT2 and NB-BERT than for NorBERT1, however, it is still arguable that the combination of parameters generally makes the metric too lenient.

6.2 Correlation

For all BERT models and all parameters, including the two parameter combinations, the SWER scores' correlation with the regular word error rate of the same sentences has been calculated. In addition, the correlation with case insensitive WER was calculated. Finally, in an attempt to uncover how well SWER as a metric matches the human opinion of semantic distance, the SWER scores' correlation with HWER, a metric based on survey data, is calculated. The correlation metric used is the Kendall rank correlation coefficient, which returns a value from `-1` to `+1`, where `-1` indicates a perfect negative relationship and `+1` indicates a perfect positive relationship.

6.2.1 Word Error Rate

The first metric SWER is compared to is the word error rate. As mentioned in section 5.5, a perfect positive relationship between the two metrics is not desired, as the motivation behind SWER is to provide an alternate metric to WER that contributes with different insights on the quality of transcripts. A perfect negative relationship is not necessarily desired either, as we want the metrics to act independent from one another to some degree.

Case Sensitive

From computing the correlation between SWER and WER, the results generally indicate a quite strong correlation between the two metrics. The correlation between SWER and case sensitive WER for all Norwegian BERT models and parameters can be seen in Table 6.5.

The results vary, with the lowest correlation with WER stemming from NorBERT1 with combination 2 of parameters applied. This model and parameter combination resulted in a correlation of +0.25, which indicates a slight correlation between the two metrics. On the contrary, NorBERT1 with default parameters for SWER ended up with a correlation of +0.78, indicating a significantly strong relationship with the WER metric.

Ranging from +0.25 to +0.78, it is apparent that SWER generally correlates well with WER. A positive correlation between the two metrics can be interpreted as the semantic distance of sentences increases as a consequence of the amount of incorrect words in a sentence. Sensibly, the more incorrect words in a transcript, the further the semantics of the transcript shift.

Looking at the choice of parameters, default SWER has a notably high correlation with WER for all models. This result is expected to some degree as the metrics are calculated similarly. It is when parameters are applied that the correlation decreases. As SWER attempts to provide distinct insight into the quality of transcripts, these parameters might contribute to providing a different perspective when evaluating transcripts.

	NorBERT1	NorBERT2	NB-BERT
Default	+0.78	+0.73	+0.72
Case Insensitive	+0.59	+0.60	+0.60
No Punctuation	+0.61	+0.59	+0.59
Stop Words	+0.59	+0.57	+0.57
Binary Step	+0.28	+0.59	+0.56
Combination 1	+0.41	+0.41	+0.41
Combination 2	+0.25	+0.36	+0.37

Table 6.5: SWER correlation with WER.

	NorBERT1	NorBERT2	NB-BERT
Default	+0.62	+0.72	+0.76
Case Insensitive	+0.87	+0.85	+0.85
No Punctuation	+0.41	+0.45	+0.50
Stop Words	+0.55	+0.61	+0.64
Binary Step	+0.26	+0.73	+0.73
Combination 1	+0.62	+0.62	+0.62
Combination 2	+0.37	+0.51	+0.53

Table 6.6: SWER correlation with case insensitive WER.

Case Insensitive

As we alter the WER metric to be case insensitive, the correlation with SWER shifts in value as well. The most noticeable difference is that case insensitive SWER carries a remarkably high correlation with case insensitive WER. It is expected that the metrics correlate better when both are case insensitive, however, a correlation of +0.87 for case insensitive NorBERT1 indicates that the metrics ultimately offer the same insight.

The correlation between default SWER and WER experiences an unsurprising decrease as case insensitivity is applied to WER.

Generally, an increase for most parameters and models can be observed as a result of ignoring case for WER. This may be an indication of WER punishing capitalisation errors heavily.

6.2.2 Human Word Error Rate

As described in subsection 5.5.2, a survey was created containing 29 sentence pairs in attempt to capture the human opinion of semantic distance between the sentence pairs. The survey received 40 responses, where each respondent evaluated an average of 20.4 sentence pairs, resulting in a total of 817 ratings.

The scores from the survey results were processed to match the shape of our SWER

	Correlation
NorBERT1	+0.34
NorBERT2	+0.36
NB-BERT	+0.36

Table 6.7: SWER correlation with human word error rate.

metric, then the semantic word error rate was computed for the sentences for each of our Norwegian BERT models. The human word error rate, along the semantic word error rates for the first 10 sentence pairs can be viewed in Table 6.8. The remaining sentences' HWER and SWER can be found in section A.2 Note that the semantic word error rate has been computed with the default parameters, meaning the punctuation is not ignored and the sentences are case sensitive. To

An observation from the survey results is that as little as one word can strongly effect people's interpretation of a sentence's meaning. This is often quite valid, as a sentence's meaning can often change entirely because of a single word, an example being the presence of the word “not” which can negate a sentence, potentially altering it to mean something entirely opposite.

An example sentence pair from the survey that highlights the strength of human interpretation and the weakness of a computational approach can be seen here:

Automatically generated transcript:
Det er ikke søtt og møtet hevet.

Gold label:
det er ikke sett. og møtet

This sentence pair resulted in a human word error rate of 66.4%, meaning people found the sentences to be highly unrelated in meaning. With SWER, this sentence pair scored 20.95% with NorBERT1, 24.87% with NorBERT2, and 36.54% with NB-BERT. Even though these error rates are considerably high, they are significantly lower than human word error rate of 66.4%.

This exact sentence pair may function as an argument against our metrics' word-for-word approach in counting semantic distance, as humans rather evaluate the semantic distance between sentences by comparing them on sentence-level, instead of comparing one word at a time. This sentence pair would also achieve a lower SWER with punctuation excluded, as the absence of a full stop in the automatically generated transcript causes a one-off error, where the wrong words are being compared to each other after it.

For our 29 selected sentences, the correlation between their SWER and HWER can be seen in Table 6.7. The results present a slight correlation between SWER and HWER for all three BERT models, ranging from +0.34 to +0.36. This suggest a moderately positive correlation between metrics, indicating a discernible relationship between the values of the two. However, a correlation below +0.50 does not indicate a remarkably strong correlation between the metrics.

It is crucial to emphasise that these exact results are based on a small sample size. The insights derived from this correlation coefficient would likely be more valuable if the survey data consisted of more sentence pairs and more responses.

6.3 Sentence Analysis

So far, the metric has been evaluated with quantitative approaches, depicting the metric's performance in different scenarios using our implemented parameters, as well as to what degree it correlates with other metrics. Further, we delve deeper into the metric, attempting to gain a deeper understanding of its performance by running a small set of sentences through the SWER function and investigating the results more closely.

Firstly, we divide our set of 553 sentence pairs into three groups. The three groups represent sentences with a low SWER, sentences with a medium SWER, and sentences with a high SWER. The low SWER group consists of sentences below a 15% SWER, while the medium group consists of sentences above 15% and below or equal to 30%. Finally, the high group contains all sentences above 30%.

The sentences were divided into groups for all different parameters, including the parameter combinations, using all three BERT models. The resulting number of sentences per group can be seen in Table 6.9. This table aims to provide a better picture of the SWER distribution. For instance, one can observe that combination 2 of parameters results in a significantly low amount of sentences with a SWER above 30% with only 6 sentences in that group.

6.3.1 Word Severity

To investigate the greatest contributors to sentences' final semantic word error rate, we print the three most severe words for three sentences in each of our sentence groups. This is done with NorBERT1 using the default parameters. The results can be seen in Table 6.10, Table 6.11, and Table 6.12. For each sentence pair, the highest contributor to the SWER is displayed in the column labelled "1. word", while the second and third highest contributors are seen in the "2. word" and "3. word" columns respectively. If

Transcript	Gold Label	HWER	Nor1	Nor2	NB
Man otte og 20 år.	mann åtteogtyve år.	49.6%	35.94%	55.64%	61.72%
Olaug Nilssen horn er hustra med at jeg har på den nasjonale cc, og hun har nettopp satt opp et stykke som heter stort og stygt.	Olaug Nilssen hun er husdramatiker på Den Nationale Scene. hun har nettopp satt opp et stykke som heter og Stort og stygt	37.2%	30.83%	38.03%	48.61%
I dag jobber hun 20 %, er 80 % sykmeldt, og hun er ikke ulykkelig, men hun er sliten .	i dag jobber hun tjue prosent er åtti prosent sykemeldt. hun er ikke ulykkelig men hun er sliten.	12.6%	33.18%	43.74%	53.67%
Kanskje hvert 10.	kanskje hvert tiende	4.8%	29.53%	33.82%	44.52%
Har vi gitt forstår slakk til kommunene?	har vi gitt for stor slack til kommunene	57.6%	29.71%	33.82%	38.75%
Det er ikke søtt og møtet hevet.	det er ikke sett. og møtet	66.4%	20.95%	24.87%	36.54%
Er at jeg har en hypotese og den hypotesen gruer på .	er at jeg har en hypotese. og den hypotesen går ut på	55.6%	22.69%	26.37%	30.82%
Det semi der det fungerer, så tar spesialisthelsetjenesten en aktiv rolle veiledningsrolle.	det ser vi. der det fungerer så tar spesialisthelsetjeneste en aktiv rolle veiledningsrolle.	51.8%	40.82%	46.38%	56.82%
Elektroniske fagprosedyre for tidlig og intensiv behandling basert på atferdsanalyse det som nå har fått navnet eibi og dette vil og bli publisert på helsebiblioteket i våren tjuesytten, og det er jo et felles.	elektronisk fagprosedyre for tidlig og intensiv behandling basert på atferdsanalyse det som nå har fått navnet EIBI. og dette vil òg bli publisert på Helsebiblioteket våren tjuesytten og det er jo et felles	17.2%	17.98%	22.84%	27.49%
Men det kommer i en dialog ved hjelp av pasient og brukerombudene som nå og har et ansvar innenfor dette området.	ned men komme i en dialog ved hjelp av pasient - og brukerombudene som nå òg har et ansvar inn forbi dette området her.	46.4%	25.13%	31.97%	38.75%

Table 6.8: Results for the first 10 sentences from human word error rate (HWER) survey. Where Nor1, Nor2, and NB stand for respectively NorBERT1, NorBERT2, and NB-BERT.

Parameter	**Model**	**< 15%**	**15% < & ≤30%**	**> 30%**
Default	NorBERT1	489	41	23
	NorBERT2	455	67	31
	NB-BERT	419	95	39
Case insensitive	NorBERT1	512	21	20
	NorBERT2	482	48	23
	NB-BERT	451	74	28
No punctuation	NorBERT1	503	27	23
	NorBERT2	506	25	22
	NB-BERT	483	42	28
Stop words	NorBERT1	455	63	35
	NorBERT2	383	116	54
	NB-BERT	344	140	69
Binary step	NorBERT1	516	27	10
	NorBERT2	424	90	39
	NB-BERT	421	89	43
Combination 1	NorBERT1	526	7	20
	NorBERT2	522	12	19
	NB-BERT	505	22	26
Combination 2	NorBERT1	533	14	6
	NorBERT2	512	19	22
	NB-BERT	503	27	23

Table 6.9: Number of sentences in certain SWER ranges.

Transcript	Gold Label	1. word	2. word	3. word
Sønnen vår er språkløs.	sønnen vår er språkløs	"Sønnen" - "sønnen"		
Kommunen vår har ingen kunnskap om alibi.	kommunen vår har ingen kunnskap om EIBI tidligere	"alibi" - "EIBI"	"." - "tidligere"	"Kommunen" - "kommunen"
Hvem skal ha ansvaret?	hvem skal ha ansvaret.	"?" - "."	"Hvem" - "hvem"	

Table 6.10: Most severe word pairs for three low SWER sentences with NorBERT1 and default parameters.

Transcript	Gold Label	1. word	2. word	3. word
Og noen steder?	og noen steder	"?" - ""	"Og" - "og"	
Stor ruiter vær så god president.	de Ruiter vær så god. president	"ruiter" - "Ruiter"	"Stor" - "de"	"" - "."
Og så dårlige holdninger.	også dårlige holdninger	"Og" - ""	"så" - "også"	

Table 6.11: Most severe word pairs for three medium SWER sentences with NorBERT1 and default parameters.

there are fewer than 3 errors for a pair of sentences, the row may contain some blank cells.

Since there were so few sentences for NorBERT1 with combination 2 of parameters in the high SWER group, the three most severe words for all six of its sentences were printed. This is to investigate what separates these high SWER sentence pairs from the rest for this combination. The results are displayed in Table 6.13.

For the most severe errors for sentences from the low SWER group with NorBERT1 in Table 6.10, the results correspond well with our expectations. Most of the contributors to the SWER scores in general seem to be capitalisation errors and erroneous punctuation. For the second sentence, the error producing the highest cosine distance is the comparison of the word “alibi” with the named entity “EIBI”, which is a desired behaviour as these two words refer to very different things. The second highest contributor to the error rate is the comparison of a full stop with the word “tidligere”. The comparison of these two tokens occurs as a consequence of the ASR system missing the word “tidligere”, and it is reasonable that it is the second highest contributor to the SWER of the sentences.

Transcript	Gold Label	1. word	2. word	3. word
Det er ruiter.	de Ruiter	"." - "Ruiter"	"ruiter" - "de"	"Det" - ""
Man otte og 20 år.	mann åtteog-tyve år.	"20" - "åtteog-tyve"	"otte" - ""	"Man" - ""
Gutt, 17 år.	gutt sytten år	"17" - "sytten"	"," - "gutt"	"Gutt" - ""

Table 6.12: Most severe word pairs for three high SWER sentences with NorBERT1 and default parameters.

Looking at the medium SWER sentences in Table 6.11, the most severe words are quite similar to the previously mentioned cases, as capitalisation and punctuation errors seem to be a recurring theme. However, something to take notice of is the third sentence pair. In this case the ASR system has transcribed the word “også” as two separate words, “og” and “så”. This word separation ultimately results in producing two errors, and since the sentence is relatively short, the SWER ends up being considerably high. Arguably, the SWER for this sentence pair should be much lower as the sentences share the exact same meaning.

For our final group with default parameters, seen in Table 6.12, the high SWER group, we witness some different cases of behaviour. For the first sentence pair, the ASR system has mistaken “de”, which is a part of a surname, as “Det er”. This is a quite considerable error, and due to the short length of the sentence it produces a high SWER which can be considered reasonable. The next sentence is also short, which means less errors are needed to produce a high SWER. For this sentence pair, the ASR system makes a significant error in parsing the number “åtteogtyve”, dividing it into three words, “otte”, “og”, and “tyve”. The word “otte” is considerably distant from the number “åtte” so it can be argued that the metric performed well in evaluating this transcript. The final sentence pair in this group is identical in meaning, yet resulted in a high SWER. This is partly due to the presence of a comma causing some slight misalignment. However, the biggest contributor to the error rate appears to be the word pair of “17” and “sytten”, which is clearly undesirable behaviour as “sytten” is just the number 17 spelled out in written form. The high cosine distance between the word embeddings for these two tokens underline the importance of model selection, as the cosine distance between these should be remarkably lower.

As combination 2 of parameters with NorBERT1 only resulted in 6 sentences with a SWER above 30%, we investigate these more closely to see why they achieved a noticeably higher SWER than the others. The results are displayed in Table 6.13. The two first sentence pairs are the same as in Table 6.12. The remaining ones have clearly produced a high SWER as a consequence of short sentence length and faulty alignment

Transcript	Gold Label	1. word	2. word	3. word
Man otte og 20 år.	mann åtteog-tyve år.	"20" - "åtteog-tyve"	"otte" - ""	"Man" - ""
Gutt, 17 år.	gutt sytten år	"17" - "sytten"	"Gutt" - ""	
Anette drang-sholt, som er leder i autisme-foren.	Annette Drang-sholt som er leder i Autisme-foren	"drangsholt" - "Annette"	"autismeforen" - ""	"Annette" - ""
Utvalget fores-lår 8.	Utvalet foreslår	""Utvalget" - "utvalet"	"8" - ""	
Anders.	Andersen	"Anders" - ""		
Ja.	hevet	"Ja" - ""		

Table 6.13: Most severe word pairs for high SWER sentences with combination 2 of parameters and NorBERT1 as model

.

of the transcripts. This strongly indicates that this choice of model and parameters makes the metric too lenient, generally producing too low error rates.

6.3.2 Sentence Comparison

With the intentions of exploring the strengths and weaknesses of the semantic word error rate, we conduct a series of experiments on a small selection of sentences. First we test the results SWER produce when evaluating completely unrelated sentences. Further, we investigate the semantic word error rates of some artificially produced sentences to potentially highlight some limitations and strengths of the metric.

Random Sentences

To test the strictness of SWER as a metric, we calculate the SWER of two random sentences once for each BERT model. This is done to develop a better understanding of what value the metric returns for sentences that are seemingly unrelated. Additionally, the sentences are tested with binary step applied to see whether the function will contribute to increasing or decreasing the error rate.

NorBERT1 Random Transcript:
Sentence 1: Den gir en unik innsikt i det å både være foreldre.
Sentence 2: men problemet er volum og kapasitet.

The first pair of random sentences is a clear example of two sentences that should ideally result in a high SWER as they are evidently unrelated and semantically distant. The word error rate for these sentences is 100%, meaning no tokens are the same. On the contrary, the SWER for the sentences is 33.02%. For two completely unrelated sentences, this error rate is arguably too low. With binary step applied as an activation function, the error rate is reduced 8.34%, which is undeniably an undesired behaviour in this case.

NorBERT2 Random Transcript:
Sentence 1: statsråd Høie vær så god.
Sentence 2: Der oppstår det fryktelig mye konflikter ofte.

This next example also produced two strongly unrelated sentences, with a word error rate of 100%. This sentence pair produces a SWER of 44.37%, which is a slight improvement from the previous sentence pair. However, this still indicates that the metric is too lenient when evaluating highly dissimilar transcripts. With binary step applied, the SWER experiences a notable reduction to 37.50%.

NB-BERT Random Transcript:
Sentence 1: Habiliteringstjenesten forteller at brukere.
Sentence 2:Det opereres med ulike tall på forekomst av autisme, men cirka 1 % av befolkningen har autisme.

Our final example consists of yet another sentence pair that results in a word error rate of 100%. The SWER for these sentences is 80.90%, which is significantly better than the previous examples considering the semantic word error rate should be high for unrelated sentences. With binary step applied, the SWER increases to 94.73%.

These three random sentence pairs combined suggest that the SWER may be too lenient, producing error rates lower than desired for sentences that are critically different. A potential fix to this could be an alternate activation function, similar to what we discussed in subsubsection 6.1.1, which punishes cosine distances more heavily.

The results from this experiment make a strong case for why the SWER should not be considered as a replacement for the traditional WER, but rather as a supplement.

Artificially Produced Sentences

Continuing, we test SWER on our set of three artificially produced sentences. These sentences were crafted with the intention of exploring edge case behaviour of the metric. The first sentence pair in our mock data aims to highlight the consequence of a negation being mistaken for another word, heavily altering the semantics of the sentence.

These artificially produced sentences are tested with SWER using default settings and NorBERT2 as choice of model. The choice of model is somewhat arbitrary. However, so far, NorBERT2 has seem to produce fairly balanced results, making it a viable choice for this short series of experiments.

Artificial Sentence Pair 1:
Sentence 1: Det var virkelig en bra dag.
Sentence 2: Det var ikke en bra dag.

This sentence pair results in a SWER of 4.35% which is due to only one word being dissimilar in the sentences. However, the desired behaviour would be a far higher SWER, seeing as the two sentences have quite the opposite meaning from each other.

The next sentence pair is intended to find out what the SWER is for two sentences that are worded quite differently but essentially share the same meaning. The sentences are intended to consist of a large amount of synonyms sharing the same index, to see what the semantic word error becomes when the sentences consist of syntactically different, yet semantically close, words.

Artificial Sentence Pair 2:
Sentence 1: Kjøretøyet suste unna i høy fart.
Sentence 2: Bilen kjørte vekk i enorm hastighet.

These two sentences have a word error rate of 71.42%, meaning almost all the tokens are different from each other. The SWER for the sentences is 20.12% which is much less than the WER, meaning SWER successfully captures some similarity between them. It is worth noting that with binary step applied, the SWER is 0.00%. However, as observed with the randomly selected sentences, the binary step may seem to often reduce cosine distances more than desired.

The final mock sentence-pair intends to depict a one-off error, where due to the absence of a comma, the wrong words are being compared for a large part of the sentences.

Artificial Sentence Pair 3:
Sentence 1: På fredag , så dro jeg ikke på jobb fordi jeg var syk.
Sentence 2: På fredag så dro jeg ikke på jobb fordi jeg var syk.

After the first two tokens, one sentence is shifted as a consequence of the presence of a comma. This causes the word “dro” to be compared to “jeg”, and “jeg” to “ikke”, etc. The SWER for this sentence pair is 30.23%, which would desirably be lower as the sentences basically have identical meanings. This example sentence-pair makes a good argument for why a sentence-level approach may be better than our current word-for-word comparison.

6.4 Summary

This chapter has presented average semantic word error rates when using different BERT models, as well as the effect of including various combinations of parameters. Additionally, to gain a deeper understanding of the metrics’ behaviour, we computed its correlation with regular word error rate, case insensitive word error rate, and our survey based human word error. Furthermore, and to investigate the results closer, we then grouped the sentences by their semantic word error rate for all model and parameter combinations and looked into what words contribute to increasing the error rate. Finally, we test the metric on a set a of random unrelated sentences, as well as some artificially produced sentences, to explore the metrics’ behaviours in some scenarios one could consider as edge cases.

Chapter 7

Discussion and Conclusion

Throughout this thesis, the development and evaluation of the Semantic Word Error Rate (SWER) as a tool for evaluating automatic Speech Recognition (ASR) systems have been at the forefront. The objective was to extend upon the existing Word Error Rate (WER) metric to implement the aspect of taking semantic information into consideration.

This chapter aims to discuss the implications of the results, including the effects of the parameters, the limitations of the metrics, and some potential areas of improvement. Finally, the thesis is concluded by summarizing how we adressed the research questions we set out to explore.

7.1 Discussion

The results show that average semantic word error rates for our dataset varies greatly dependent on choice of model and parameters. Fortunately, the metric offers flexibility over these choices, as the choice of BERT model is provided to the metric, while parameters are optional.

Additionally, when analysing the results on sentences closely, some weaknesses of the metric became clear, as well as its limitations which have effected the outcomes of the results.

7.1.1 Difference in Model Performance

As the metric builds upon the cosine distances between the word embeddings produced by the provided BERT model, selection of BERT model can effect SWER's output greatly. On a general level, it is apparent that NorBERT1 produces the lowest semantic word error rates, while NB-BERT tends to produce the highest.

Both multilingual models we tested, XLM-RoBERTa and S-BERT, produced significantly lower error rates in average, where XLM-RoBERTa resulted in a remarkably low average SWER for the dataset. It is important to underline that the goal of the metric is to accurately evaluate transcripts, not achieve a low error rate, meaning these results should not necessarily be interpreted as positive. However, they contribute to highlighting the importance of model selection when using the metric.

7.1.2 Parameters

For the sake of both experimentation and increasing the usability of the metric, a set of parameters were implemented. The three first parameters, which are stop word removal, punctuation exclusion, and case insensitivity, all have in common that they manipulate the textual input before the metric is computed. These parameters are not strictly necessary from a user point of view, as the user can always simply manipulate the textual input on their own before applying the metric. However, the inclusion of these parameters is interesting from an experimental point of view, as they allow us to observe how they ultimately alter the output of the metric.

When enabling case insensitivity or punctuation exclusion, the average SWER will naturally decline, as the parameters relax the strictness of the metric. On the contrary, removing stop words from the transcript tends to increase the SWER for transcripts. This occurs as a result of the fact that for every word pair removed from the transcripts, the length the sum of the cosine distances is divided by is reduced by one. An alternate implementation that may be explored, is making the metric divide the sum of the cosine distances by the original sentence length, i.e. before the removal of stop words.

A strength of the stop word parameter, is the fact that the list of stop words must be supplied by the user. This effectively makes the parameter function as a filter, where the user may supply a list of words they wish the metric to disregard.

The final parameter, the option for including an activation function, enables the user to alter the values of the cosine distances before the final computation of the metric occurs. The experimentation of applying the binary step as activation function did not produce any remarkable results, however, the flexibility the parameter offers to the user can still be of great value as other activation functions may produce more promising results.

7.1.3 Weaknesses

As mentioned, the qualitative sentence analysis underlined some current weaknesses of the semantic word error rate. There are two main weaknesses, that would improve the metric significantly if addressed. The first one being that the metric seems to be too lenient in its error rates, and the second one being the aspect of it comparing one pair of words at a time.

Lenience

The first issue with the metric, is that it rarely will approach an error rate of 100%. For a sentence pair to achieve a SWER this high, every word pair's embeddings produce a cosine distance of 1.00 between them, which is highly unlikely. To compensate for this, implementing a weighting of cosine distances could potentially improve the range of the metric. This could alternatively be achieved through the activation function parameter by supplying a function that increases the cosine distances.

An observation to be made when looking at the results of human word error rate on the sentences in Table 6.8, is that the respondents may punish sentences heavily for a single error if they consider that error to drastically alter the semantics of the sentence. A good example for this is the fifth sentence pair in Table 6.8, where the presence of the word "slakk" in place of "slack" results in a HWER of 57.6%. If the SWER metric wishes to strive towards matching human opinion of semantic distance, this type of strictness should somehow be reflected in the metric.

Word for Word Comparison

Another weakness of the metric lies in the implementation, as sentences are evaluated by comparing word pair for word pair. This implementation builds on the assumption that corresponding words exist at the same index in the sentence pairs. This assumption makes the metric quite error prone, as transcripts may often be misaligned to some degree as we experienced with our data. Additionally, ASR systems may often miss a word, or a comma for example. This may cause the transcriptions that are being compared to eventually become misaligned by one token, causing the wrong word embeddings to be compared to each other.

A potential approach to addressing this issue is by using S-BERT which possesses the ability to compute sentence embeddings. The application of S-BERT was explored to a small degree in subsection 4.2.1. However, using S-BERT in combination with our implemented parameters was left out of the scope of this thesis, meaning the viability of this is currently left unexplored.

7.1.4 Limitations

Along the course of this thesis, some limitations were met, effecting the final results to some degree. These limitations are presented here, as well as improvements that could be made to potentially overcome these.

Data Limitations

A quite severe limitation that arose for this project, was the lack of automatically generated transcripts, that occurred as a result of errors on behalf of the 3rd party speech to text provider. Due to sheer size of the audio files, generating the transcripts was a time consuming process, which often ended up being interrupted, ultimately hindering our gathering of data. Fortunately, one audio file was able to be transcribed, providing us with enough data to produce some meaningful results.

Another limitation, though less severe, is the fact that the transcripts were divided into sentences dependent on the placement of punctuation in the automatically generated transcripts during the data pre-processing. As a consequence of this, whenever the ASR system misplaces punctuation in the automatic transcripts, the corresponding gold label sentence will be unnaturally split. An arguably better approach would be to divide the transcripts into sentences based on the punctuation in the gold label transcripts, as the quality of these transcripts are assured and represent what the automatic transcripts strive to match in quality.

The final limitation within the domain of our data, is the amount of data collected by our survey for the human word error rate metric. Optimally, we would collect more survey data from more respondents on a broader span of sentences. However, due to a limited time span, only data from 40 respondents was collected. Fortunately, the resulting sample size is still of considerable size, enabling the representation of human opinion to some degree.

Alignment

In our pre-processing of the data, we attempt to align two considerably large transcripts at once. The outcome is quite satisfactory, yet it is evident for some sentence pairs that they are misaligned. For these sentence pairs we then experience the issues mentioned in subsubsection 7.1.3, where the wrong words are being compared.

To address this, other approaches to aligning transcripts could be explored. For example, in the function for SWER, after equalising the lengths of the sentences, an alignment function could be called to apply alignment on a sentence level, in contrast to applying it to our entire dataset at once.

7.2 Conclusion

This thesis has sought to address a set of research questions related to the development and evaluation of the semantic word error rate, which we will address now.

- **RQ1:** How does semantic-WER compare to other existing metrics for evaluating transcripts?
- **RQ2:** How can semantic-WER be evaluated and validated against human judgements of semantic similarity or relatedness?
- **RQ3:** How does the performance of semantic-WER compare when using different models for semantic representation?

For the first research question, we explored the correlation between SWER and the traditional WER. The results can be seen in Table 6.5, indicate a strong correlation between the two metrics, which is likely a result of the fact that the implementation of SWER is heavily built upon WER's. To compare the two, they still represent different aspects of quality when evaluating transcripts as WER has a syntactical focus, while SWER has a semantic focus.

Ideally, these metrics should be used in combination to depict different perspectives of the transcripts' quality. For example, if a sentence pair results in a low WER, but a high SWER, it may be an indication of the presence of a few, yet critical errors. On the contrary, a high WER combined with a low SWER may indicate the presence of several trivial errors.

The second research question was explored by the creation of a survey, collecting ratings from humans on how close they perceived sentences to be semantically. Though these opinions are undeniably due to subjectivity, by aggregating them and making them match the shape of SWER with some processing, we are able to compare the results of our metric with the judgement of humans. Ultimately, the results show that there is some correlation between SWER and the results of human judgement.

For our final research question, we have explored how the resulting semantic word error rates differ when different BERT models are applied. The results show that the error rates may differ significantly dependent on model choice, where some models may generally produce lower cosine distances, while others generally will produce higher ones. Therefore, it is of importance to make a conscious decision in choice of BERT model when utilising the SWER metric.

7.3 Future Work

As this thesis has built a foundation for the semantic word error rate, further work remains to improve the applicability of the metric.

The first aspect of the metric with a significant potential for improvement is addressing the issue with the word-for-word comparison mentioned in subsubsection 7.1.3. This could be achieved by utilising a sentence-level approach, such as S-BERT, or perhaps exploring more advanced approaches for sentence alignment.

A natural next step for the metric would be to develop an interface for it. As of now, the function for the metric exists inside a Python Notebook, which is not optimal for usability. A clear improvement would be to develop a command line interface where the user *e.g.* selects a BERT model and provides sentences as input and receives the SWER as output. This command line tool could also benefit from integrating other evaluation metrics to create a comprehensive evaluation framework for ASR systems.

Appendix A

Appendix

A.1 NPSC Train-Test-Evaluation split

Table A.1 contains an overview of which folders belong to which data split when dividing the dataset into a train-test-evaluation split.

A.2 HWER

In Table A.2, Table A.3 and Table A.4, are all sentences from the Human Word Error Rate survey, along with their sentence ID, HWER, and SWERs for each of the three Norwegian BERT models.

Split	**Folder names**
Train	20170110, 20170208, 20170215, 20170216, 20170222, 20170314, 20170322, 20170323, 20170403, 20170405, 20170419, 20170426, 20170503, 20170510, 20170516, 20170613, 20170615, 20171007, 20171012, 20171018, 20171024, 20171208, 20171211, 20180316, 20180321, 20180404, 20180410, 20180411, 20180601, 20180613, 20180615
Test	20171219, 20180530, 20171122, 20170207
Evaluation	20180611, 20180201, 20170209, 20180307, 20180109

Table A.1: Distribution of train-test-evaluation data in NPSC dataset.

ID	Transcript	Gold Label	HWER	Nor1	Nor2	NB
1	Man otte og 20 år.	mann åtteogtyve år.	49.6%	35.94%	55.64%	61.72%
2	Olaug Nilssen horn er hustra med at jeg har på den nasjonale cc, og hun har nettopp satt opp et stykke som heter stort og stygt.	Olaug Nilssen hun er husdramatiker på Den Nationale Scene. hun har nettopp satt opp et stykke som heter og Stort og stygt	37.2%	30.83%	38.03%	48.61%
3	I dag jobber hun 20 %, er 80 % sykmeldt, og hun er ikke ulykkelig, men hun er sliten.	i dag jobber hun tjue prosent er åtti prosent sykemeldt. hun er ikke ulykkelig men hun er sliten.	12.6%	33.18%	43.74%	53.67%
4	Kanskje hvert 10.	kanskje hvert tiende	4.8%	29.53%	33.82%	44.52%
5	Har vi gitt forstår slakk til kommunene?	har vi gitt for stor slack til kommunene	57.6%	29.71%	33.82%	38.75%
6	Det er ikke søtt og møtet hevet.	det er ikke sett. og møtet	66.4%	20.95%	24.87%	36.54%
7	Er at jeg har en hypotese og den hypotesen gruer på.	er at jeg har en hypotese. og den hypotesen går ut på	55.6%	22.69%	26.37%	30.82%
8	Det semi der det fungerer, så tar spesialisthelsetjenesten en aktiv rolle veiledningsrolle.	det ser vi. der det fungerer så tar spesialisthelsetjeneste en aktiv rolle veiledningsrolle.	51.8%	40.82%	46.38%	56.82%
9	Elektroniske fagprosedyre for tidlig og intensiv behandling basert på atferdsanalyse det som nå har fått navnet eibi og dette vil og bli publisert på helsebiblioteket i våren tjuesytten, og det er jo et felles.	elektronisk fagprosedyre for tidlig og intensiv behandling basert på atferdsanalyse det som nå har fått navnet EIBI. og dette vil òg bli publisert på Helsebiblioteket våren tjuesytten og det er jo et felles	17.2%	17.98%	22.84%	27.49%

Table A.2: Results for sentences 1 to 9 from human word error rate (HWER) survey.

ID	Transcript	Gold Label	HWER	Nor1	Nor2	NB
10	Men det kommer i en dialog ved hjelp av pasient og brukerombudene som nå og har et ansvar innenfor dette området.	ned men komme i en dialog ved hjelp av pasient - og brukerombudene som nå òg har et ansvar inn forbi dette området her.	46.4%	25.13%	31.97%	38.75%
11	Lyset i tunnelen er nettopp at vi har denne kompetansen.	lyset i tunnelen er nettopp at vi har denne kompetansen	98.0%	7.59%	8.48%	9.16%
12	Der oppfølging av våre innbyggere med autisme fungerer preges denne høy kompetanse fra spesialisthelsetjenesten og kommunene.	der oppfølging av våre innbyggere med autisme fungerer preges den av høy kompetanse fra spesialisthelsetjenesten og kommunene	81.6%	18.42%	23.16%	28.07%
13	Hun sier jeg er voksen.	hun sier. jeg er voksen.	63.0%	26.25%	37.41%	46.38%
14	Læreren min lytter til meg og la meg tenke , selv om jeg tenker seint.	læreren min lytter til meg og lar meg tenke selv selv om jeg tenker seint.	80.0%	5.9%	6.71%	7.76%
15	Jeg ønsker meg en jobb.	jeg ønsker meg en jobb.	97.6%	5.81%	7.74%	3.19%
16	Jeg har hatt 17 forskjellige arbeidsgivere , kommer i konflikt og forstår dårlig kollegaene mine.	jeg har hatt sytten forskjellige arbeidsgivere kommer i konflikt og forstår dårlig kollegaene mine.	81.4%	29.14%	37.08%	43.4%
17	Jeg har reist fra 3 samboer og 5 barn.	jeg har reist fra tre samboere og fem barn.	85.20%	13.08%	11.62%	9.84%
18	Kommunen vår har ingen kunnskap om alibi.	kommunen vår har ingen kunnskap om EIBI tidligere	29.2%	14.67%	15.02%	17.24%

Table A.3: Results for sentences 10 to 18 from human word error rate (HWER) survey.

ID	Transcript	Gold Label	HWER	Nor1	Nor2	NB
19	Vi vurderer om vi var rime og opp familien si opp jobber og flytte til en plass vi kan få hjelp , og det vises vilje til at det blir gitt tjenester som vil gi ham mulighet til utvikling og språk.	vi vurderer om vi må rive opp familien si opp jobber og flytte til en plass vi kan få hjelp og det vises vilje til at det blir gitt tjenester som vil gi han mulighet til utvikling og språk.	56.6%	4.7%	6.75%	7.91%
20	Og det er stor variasjon i hvordan mennesker med autisme fungerer i hverdagslivet.	og det er stor variasjon i hvordan mennesker med autisme fungerer i hverdagslivet.	97.8%	3.26%	2.42%	2.75%
21	Det skjer allerede en positiv utvikling i noen kommuner.	det skjer allerede en positiv utvikling i noen kommuner.	97.0%	3.48%	2.43%	2.23%
22	Vi er nødt til å lære mer av de som har funnet gode modeller og løsninger fra mennesker med sammensatte behov.	vi er nødt å lære mer av de som har funnet gode modeller og løsninger for mennesker med sammensatte behov.	84.6%	5.15%	5.7%	7.08%
23	Denne debatten har vi hatt med jevne mellomrom de siste årene , uten at vi har kommet helt i mål.	denne debatten har vi hatt med jevne mellomrom de siste årene uten at vi har kommet helt i mål.	96.0%	2.85%	4.85%	4.78%
24	Savner nok au eit system for å sikre kvalitet i tjenestene.	jeg savner nok au et system for å sikre kvalitet i tjenestene.	87.6%	31.39%	41.16%	58.55%
25	Bare å understreke et par poeng.	jeg bare understreke et par poeng.	64.0%	9.57%	10.01%	14.31%
26	De er det bare å bruke, så jeg er glad for det.	de er det bare å bruke. så er jeg òg glad for det	77.0%	21.02%	24.97%	31.17%

Table A.4: Results for sentences 19 to 26 from human word error rate (HWER) survey.

ID	Transcript	Gold Label	HWER	Nor1	Nor2	NB
27	Sak 2 er referat og det foreligger ikke referat, og dermed er Dagens kart ferdigbehandlet.	sak to er referat og det foreligger ikke referat og dermed er dagens kart ferdigbehandlet.	89.2%	5.6%	5.59%	6.7%
28	Berre 25 % av utviklingshemma i yrkesaktiv alder er i jobb.	berre femogtjue prosent av utviklingshemma i yrkesaktiv alder er i jobb.	94.2%	8.19%	10.31%	10.61%
29	Er pappa til en gutt på 6 år med autisme.	jeg er pappa til en gutt på seks år med autisme.	81.4%	38.19%	46.57%	61.21%

Table A.5: Results for sentences 27 to 29 from human word error rate (HWER) survey.

Bibliography

Abdi, H. (2007), The kendall rank correlation coefficient, *Encyclopedia of Measurement and Statistics. Sage, Thousand Oaks, CA*, pp. 508–510. 5.5.1

Al-Rfou', R., B. Perozzi, and S. Skiena (2013), Polyglot: Distributed word representations for multilingual NLP, in *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pp. 183–192, Association for Computational Linguistics, Sofia, Bulgaria. 2.1.4

Arora, S., and R. Singh (2012), Automatic speech recognition: A review, *International Journal of Computer Applications*, *60*, 34–44, doi:10.5120/9722-4190. 2.2

Brown, T. B., B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei (2020), Language models are few-shot learners. 2.1.3

Conneau, A., K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov (2020), Unsupervised cross-lingual representation learning at scale. 2.1.4, 2.1.6

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2019), Bert: Pre-training of deep bidirectional transformers for language understanding. 2.1.3, 2.1.4, 2.1.4, 2.1.5, 2.1.6, 2.1.6

Errattahi, R., A. El Hannani, and H. Ouahmane (2018), Automatic speech recognition errors detection and correction: A review, *Procedia Computer Science*, *128*, 32–37, doi:https://doi.org/10.1016/j.procs.2018.03.005, 1st International Conference on Natural Language and Speech Processing. 2.4.1

Goldberg, Y. (2015), A primer on neural network models for natural language processing. 2.4.2

Goodfellow, I., Y. Bengio, and A. Courville (2016), *Deep learning*, MIT press. 2.1.3

Graves, A., S. Fernández, F. Gomez, and J. Schmidhuber (2006), Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks, pp. 369–376, doi:10.1145/1143844.1143891. 2.2.1

Hinton, G., L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury (2012), Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, *IEEE Signal Processing Magazine*, *29*(6), 82–97, doi:10.1109/MSP.2012.2205597. 2.2.1

Hochreiter, S., and J. Schmidhuber (1997), Long short-term memory, *Neural computation*, *9*, 1735–80, doi:10.1162/neco.1997.9.8.1735. 2.1.3

Hrinchuk, O., M. Popova, and B. Ginsburg (2020), Correction of automatic speech recognition with transformer sequence-to-sequence model, pp. 7074–7078, doi:10.1109/ICASSP40776.2020.9053051. 2.3

Jurafsky, D., and J. Martin (2008), *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, vol. 2. 2.1.3

Kendall, M. G. (1938), A new measure of rank correlation, *Biometrika*, *30*(1/2), 81–93. 5.5

Kummervold, P. E., J. De la Rosa, F. Wetjen, and S. A. Brygfjeld (2021), Operationalizing a national digital library: The case for a Norwegian transformer model, in *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pp. 20–29, Linköping University Electronic Press, Sweden, Reykjavik, Iceland (Online). 2.1.6

Kutuzov, A., J. Barnes, E. Velldal, L. Øvrelid, and S. Oepen (2021), Large-scale contextualised language modelling for Norwegian, in *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pp. 30–40, Linköping University Electronic Press, Sweden, Reykjavik, Iceland (Online). 2.1.6

Li, B. (2013), Distance weighted cosine similarity measure for text classification, doi:10.1007/978-3-642-41278-3_74. 2.4.2

Likic, V. (2008), The needleman-wunsch algorithm for sequence alignment, *Lecture given at the 7th Melbourne Bioinformatics Course, Bi021 Molecular Science and Biotechnology Institute, University of Melbourne*, pp. 1–46. 4.1

Manning, C. D., P. Raghavan, and H. Schütze (2008), *Introduction to Information Retrieval*, Cambridge University Press. 2.1.2, 2.4.2

Mccowan, I., D. Moore, J. Dines, D. Gatica-Perez, M. Flynn, P. Wellner, and H. Bourlard (2004), On the use of information retrieval measures for speech recognition evaluation. 2.4.1

Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013), Efficient estimation of word representations in vector space, *Proceedings of Workshop at ICLR*, *2013*. 2.1.4, 2.1.4

Pearson, K. (1895), Note on regression and inheritance in the case of two parents, *Proceedings of the Royal Society of London*, *58*, 240–242. 5.5

Pennington, J., R. Socher, and C. Manning (2014), GloVe: Global vectors for word representation, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Association for Computational Linguistics, Doha, Qatar, doi:10.3115/v1/D14-1162. 2.1.4, 2.1.4

Peters, M., M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer (2018), Deep contextualized word representations. 2.1.4

Pires, T., E. Schlinger, and D. Garrette (2019), How multilingual is multilingual BERT?, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4996–5001, Association for Computational Linguistics, Florence, Italy, doi:10.18653/v1/P19-1493. 2.1.4

Radford, A., K. Narasimhan, T. Salimans, and I. Sutskever (2018), Improving language understanding by generative pre-training. 2.1.3, 2.1.4, 2.1.5

Rajaraman, A., J. Leskovec, and J. Ullman (2014), *Mining of Massive Datasets*, doi: 10.1017/CBO9781139058452. 2.1.2

Ramos, J. (2003), Using tf-idf to determine word relevance in document queries. 2.1.2

Reimers, N., and I. Gurevych (2019), Sentence-bert: Sentence embeddings using siamese bert-networks. 2.1.6

Rosenberg, D. (2014), Stop, words, *Representations*, *127*(1), 83–92. 4.3.2

Roy, S. (2021), Semantic-wer: A unified metric for the evaluation of asr transcript for end usability. 2.5

Salton, G., A. Wong, and C. S. Yang (1975), A vector space model for automatic indexing, *Commun. ACM*, *18*(11), 613620, doi:10.1145/361219.361220. 2.4.2

Spearman, C. (1904), The proof and measurement of association between two things, *The American Journal of Psychology*, *15*(1), 72–101. 5.5

Sutskever, I., O. Vinyals, and Q. V. Le (2014), Sequence to sequence learning with neural networks. 2.2.1

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin (2017), Attention is all you need, *Advances in neural information processing systems*, *30*. 2.1.3, 2.1.5, 2.2.1

Virtanen, A., J. Kanerva, R. Ilo, J. Luoma, J. Luotolahti, T. Salakoski, F. Ginter, and S. Pyysalo (2019), Multilingual is not enough: Bert for finnish. 2.1.6

Wu, S., and M. Dredze (2019), Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 833–844, Association for Computational Linguistics, Hong Kong, China, doi:10.18653/v1/D19-1077. 2.1.4