Disambiguating Dutch Medical Abbreviations using Language Models

Julia van Oosten
Information Systems, University of Amsterdam
Amsterdam, The Netherlands

ABSTRACT

Miscommunication between medical professionals from different fields of healthcare can cause grave consequences. This is i.a. caused by ambiguity in medical abbreviations used in clinical notes. An abbreviation can mean one thing in optometry and something else in the field of neurology. It is therefore important that these medical abbreviations are disambiguated. Natural Language Modeling and the growing amount of medical data allow for medical abbreviation disambiguation. This application has been explored in earlier research, but is not explored in the Dutch language. Therefore, in this research paper, a Dutch, medical abbreviation disambiguation text model and dataset are proposed and analyzed to answer the following questions: to what extent can the abbreviation disambiguation for Dutch medical text be improved upon through the use of language models? To answer these questions, three models were created, namely two multi-classifiers and a binary model. These models used the language model RobBERT as a basis and were trained on a specially curated dataset. Afterwards, they were compared against a Zero Rule classifier using accuracy/micro F1-scores and macro F1-scores. Results showed that the binary model improved upon the baseline greatly when testing on both an balanced and imbalanced test set.

1 INTRODUCTION

The growing amount of digital clinical data, such as clinical notes, medical literature, and electronic health records (EHR), create an opportunity for healthcare informatics to explore the insights and potential of automatic inference, data mining techniques and their application to healthcare questions [9]. Recent work focuses on mining medical text to create deep learning models that are capable of natural language processing (NLP) fit for multiple different medical tasks, such as predicting the mortality of intensive care patients using unstructured clinical notes, decision support [9], disease-risk prediction, clinical recommendation, and precision medicine [13].

Although there is potential in this field for NLP application, there are also challenges when applying these techniques to medical documentation. One of these challenges is that most electronic health records and clinical notes consist of free-form text [16]. Health practitioners prefer using free-form text as it provides them more freedom of expression opposed to predefined data structures [16]. This free-form text often contains (informal) abbreviations that can have different meanings in different medical contexts. For example, the abbreviation*v* can mean *volume* in the field of pulmonology and *vene* in the field of cardiology. A challenge here is, that when using notes from different health practitioners, confusion can occur. Wrongful interpretation of medical abbreviations has caused serious errors and deaths in the past [12]. Therefore, correctly disambiguating abbreviations is an important task. This problem of wrongful interpretation of medical abbreviations is also present in the Netherlands [18].

Another problem with creating NLP Systems in the field of healthcare is that there are few adequate, medical datasets available to the public. This is not only a problem in the English-language medical field, Dutch-language medical datasets are even more scarce. Because of this scarcity of suitable data, there is a limited offer of specifically tuned models for processing healthcare records, clinical notes, etc ... in the Dutch language [5].

Inspired by these challenges, the following research question has been set up: To what extent can the abbreviation disambiguation for Dutch medical text be improved upon through the use of language models? Additionally, the following sub-question was defined: What is an appropriate dataset for the task? To answer these questions, the method was loosely based on the state-of-the-art approach as seen in the English-language dataset and model, MeDAL [19], where the aforementioned challenges have been tackled.

The structure of this paper is as follows. In the Related Work Section, MeDAL and language models BERT and RobBERT are described. Also, an overview of applications of Dutch Medical NLP is given. Furthermore, in the method, the data collection and preparation process is explained, after which three models are presented. Next, the results are presented. In the discussion section, these results are further analyzed and the research questions are answered. Furthermore, the limitations of the research method and the possible use of proposed models are discussed. The paper ends with future work recommendations and a conclusion.

2 RELATED WORK

In this section, the MeDAL dataset is further explained, as the chosen method of this project is heavily influenced by the creation of this dataset. Next, a brief overview of current developments within the Dutch field of medical NLP is given. Furthermore, language models and in particular BERT and RobBERT are explained.

2.1 MeDAL

In the introduction, it was discussed that challenges within NLP in the medical field included data availability and ambiguity in biomedical abbreviations. This paper [19] introduces a state-of-the-art approach to solving these problems. In this paper, both a model and a dataset were curated for the use of medical abbreviation disambiguation and called **Me**dical **D**ataset for **A**bbreviation Disambiguation for Natural **L**anguage Understanding (MeDAL). The dataset consists of 14,393,619 article abstracts of PubMed articles. Reverse substitution was used to generate samples without human labeling. When doing this, terms in the text that have known abbreviations were determined and then 30% of those terms were replaced with their abbreviations. Then, mappings of abbreviations to

expansions were used [20]. To give an example of an abbreviation-expansion mapping is that *a* is an abbreviation that maps to the expansion of to the full word *arterie*. Furthermore, if an abbreviation only mapped to one expansion, i.e. the abbreviation could only point to one meaning, the mapping was discarded. Next, if an expansion mapped to multiple abbreviations, the mapping was discarded as well. In the end, 24,005 valid pairs of mappings were created consisting of 5,886 abbreviations. Each abbreviation had about 4 expansions on average.

Several models were pre-trained on the dataset, with the main model being the Electra-small discriminator [3] and BiLSTM [11] being the baseline model. To pre-train the model, the abbreviation disambiguation task was treated as a classification problem where the classes signified all possible expansions or meanings. Because of the size of the dataset, a subset of 5 million data points was used. This subset was then further split into 3 million training samples, 1 million test samples, and 1 million validation samples [19].

To evaluate the model, the model was fine-tuned on two downstream tasks. Consequently, the results of the disambiguation problem were not given in the paper. The first evaluation task was mortality prediction, where ICU patient notes were used to predict the mortality of a patient at the end of a hospital stay. The second evaluation task was diagnosis prediction, where medical notes were used to predict the diagnosis of a patient. These evaluation tasks were performed by models that were pre-trained on the MeDAL dataset, after which the pre-trained weights were used to initialize models for training on the downstream tasks. These pre-trained models were then compared to models trained from scratch [19].

Both for the diagnosis prediction and the mortality prediction, the pre-trained models performed better than the models made from scratch. For the diagnosis prediction, the difference between a pre-trained and a model made from scratch was more significant with a model performance increase by more than 70%. The results show that pre-training using MeDAL offers an advantage in medical NLP [19].

2.2 Natural Language Processing of Dutch Medical Text

The research on natural language processing on Dutch medical text is still rather limited. In this section, multiple papers pertaining to Dutch medical NLP applications are summarized.

A study from 2012 [5] compiled an inventory of available tools for Dutch medical text NLP applications. This was done with the intent to map Dutch phrases to SNOMED-CT [8] concepts. SNOMED-CT is an advanced terminology and coding system for eHealth.

In this research paper [5], it is stated that most NLP tools and terminological systems cater to the English language. Consequently, some of the presented NLP tools were usable for both Dutch and English, whereas some tools were only suitable for Dutch. The categorized NLP tools were suitable for different components in the NLP pipeline. Furthermore, some of these components were language-independent and could therefore be executed with a wide variation of tools.

A more recent study [2] showed a NLP approach to gaining insights from free-text patient experiences. For this, 38,664 survey responses on patient experience were categorized into topics, n-grams after which a sentiment score was calculated. Then, an impact indicator was created. For this, the frequency and sentiment score were combined to indicate which topics need improvement. For the sentiment score, the sentiment analysis tools from the pattern.nl module were used. In addition to sentiment analysis tools, this module offers a multitude of tools for Dutch sentence analysis such as a part-of-speech tagger and tools for verb conjugation.

Another application of NLP on Dutch medical text is the de-identification or anonymization of notes or other private medical text files (therapeutic diaries, blog posts..). A study from 2019 [17] tackled this problem by converting Dutch Wikipedia biographies into autobiographies, and then using the Dutch linguistic processing program Frog for the de-identification of these autobiographies.

A study from 2020 [14] reported a rule-based NLP algorithm for classifying Dutch free-text radiology reports. This algorithm aimed to improve the structure of the reports and exposing the information they contain. This way, further analysis is facilitated. For this model, PyContextNLP, spaCy and regular expressions were used. Both of these libraries cater to the Dutch language.

Lastly, a study from 2014 [1] proposed the ContextD algorithm and a Dutch clinical corpus. This algorithm was created to identify contextual properties of medical terms in a Dutch clinical corpus. The clinical corpus was used in the method of this research (see 3.1.2) and is therefore not elaborated upon in this section. The tools used for this algorithm were the Apache OpenNLP library, a custom UML Dutch term list, and was based on the English-language ConText algorithm as proposed in earlier research[10].

These papers all show a wide variety of Dutch medical NLP applications. However, the research direction of this paper seems to be relatively unique in Dutch and can therefore contribute knowledge to the field of Dutch Medical NLP.

2.3 Transformer Models and Pre-trained Language Models

Pre-trained language models have dominated recent works in the domain of natural language processing, outperforming other language models in a wide range of NLP tasks [6]. When pre-training a model, a model is trained on a specific task, learning an embedding representation for the input words. Then, these same word embeddings can be used as input for another task, creating a pre-trained model[1]. In this section, multiple large-scale transformer models[2] are introduced. The key innovation of transformer models is the use of self-attention layers. Summarized, the essence of self-attention learning lies within comparing a certain item against other items within a given sentence, revealing their relevance in the current context.

It is important to note that relying on these transformer models (see Section 3.2) for the embeddings of input words creates a pre-trained model.

[1] https://web.stanford.edu/ jurafsky/slp3/7.pdf
[2] https://web.stanford.edu/ jurafsky/slp3/9.pdf

2.4 BERT

In 2018, a new language representation model called BERT was introduced. The abbreviation BERT stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers [7]. BERT is unique because it looks at the text in a bi-directional matter. Another feature that makes BERT unique is that it only needs one additional output layer to fine-tune the model and thus enabling the model to perform a certain NLP task. This study uses one of the strategies for applying pre-trained language models to specific tasks, namely fine-tuning. The fine-tuning approach uses the pre-trained model parameters and fine-tunes these on the downstream task.

BERT is created in two steps: by pre-training and fine-tuning. During the pre-train stage, the model is trained on a raw corpus of text originating from BooksCorpus and English Wikipedia. The pre-training is done using two unsupervised tasks, Masked LM and Next Sentence Prediction (NSP). In masked LM, a percentage of the input tokens is masked and the model has to guess which word is underneath. The other task, Next Sentence Prediction, is based on understanding the relationship between two sentences. In this task, the model reviews two sentences (A and B) and has to predict if one follows the other. 50% of these B sentences are not the next sentence. Next, during the fine-tuning stage, these pre-trained parameters are initialized and all the model parameters are fine-tuned on a specific task. BERT was fine-tuned on seven NLP tasks and showed promising results when compared to baseline models. [7]

2.4.1 *RobBERT.*

In this study [6], the pre-trained language model RoBERTa, a more robust variant of the well-known BERT was, was used as a basis for the Dutch pre-trained language model, RobBERT. A multi-lingual BERT-model (mBERT) already existed, trained on a corpus in 104 different languages, however, models trained on data of one specific language are known to perform better than their multi-lingual counterpart [6].

For pre-training, the same method as used for RoBERTa was followed. Two versions were trained, one where the corpus was replaced for a Dutch corpus, RobBERT v1, and one where both the corpus and the tokenizer were adapted to Dutch, named RobBERT v2. The Dutch corpus that was used, consisted of the Dutch Section of the OSCAR-corpus [15]. This corpus had a size of 39GB, with 126 million lines including over 6.6 billion words. Other Dutch pre-trained language models, such as BERTje and BERT-NL used significantly smaller corpora. For RobBERT v2, the default byte pair encoding (BPE) tokenizer was swapped out for a Dutch tokenizer. This tokenizer was constructed with the aid of the same Dutch OSCAR-Section and the same BPE *algorithm* as used in RoBERTa. Afterward, the model was trained using the masked language model (MLM) task [6]. Note: BERT is also trained on the Next Sentence Prediction Task, which was omitted when training RobBERT.

Lastly, for evaluation, RobBERT performed multiple downstream, Dutch language tasks. One of these tasks was sentiment analysis, which shows how well the model performs on text classification. Also, *Die/Dat* disambiguation was tested for text classification. For both of these tasks, RobBERT v2 outperformed other Dutch pre-trained language models BERTje, BERT-NL, the baseline model, mBERT, and its variant RobBERT v1 significantly. To test RobBERT's token tagging capabilities, part-of-speech (POS) tagging and named entity recognition (NER) tasks were performed. Again, RobBERT v2 outperformed the previously mentioned models. The results show that pre-trained models on a single language outperform multi-lingual models and that using an adapted tokenizer is of great value in creating these models [6]. As seen in Section 2.1, the pre-trained model ensures a certain advantage, meaning that using RobBERT as a basis for the biomedical abbreviation disambiguation task could prove to be useful.

3 METHOD

To be able to disambiguate medical abbreviations, relevant data had to be collected and processed into a certain format. In this section, the data collection, pre-processing, and the creation of the model is elaborated upon.

3.1 Data Collection and Preparation

3.1.1 *Biomedical Abbreviations Dataset.*

First, a dataset of biomedical abbreviations had to be sourced. It was important that the abbreviation dataset contained abbreviations from different disciplines in medicine, as the goal was to find abbreviations mapping to different expansions. The reasoning behind this was that overlapping abbreviations are less likely to occur within the same discipline. The dataset was created by combining abbreviation datasets from three different sources, namely *Optometristen Vereniging Nederland*, the Academic Hospital of Brussels (*UZ Brussel*) and *Medischeafkorting.nl*. The first source contained, as the name of the association already suggests, optometry-related abbreviations, whereas the latter two sources contained a broad selection of abbreviations from different disciplines within medicine.

The datasets (CSV-files) from each source were cleaned, meaning that rows containing NaNs were removed and that the abbreviations and their expansions were lower-cased. The three datasets dataset that resulted after cleaning and formatting, were combined to produce one biomedical abbreviation dataset. Furthermore, rows with both the same value for the abbreviation as for the expansion were removed. It is important to note that those rows were seen as removable duplicates, whereas duplicates when only considering the abbreviation column, were necessary to be able to research the task of biomedical abbreviation disambiguation. Therefore, the final step of creating a useful biomedical abbreviation dataset for this task was to extract abbreviations with two or more meanings. The final dataset consisted of 2 columns containing 655 rows of abbreviations and their expansions.

3.1.2 *Medical Corpus.*

A biomedical text corpus was created. Previous to this, data from two sources were combined, namely, from Europe PubMed Central (Europe PMC) [4] and EMC Dutch Clinical Corpus [1]. The latter dataset contains sensitive information, which, as a result, required a signed License of Agreement. This corpus, created by researchers of the Department of Medical Informatics at Erasmus Medical Center, contains entries from general practitioners, specialists' letters, radiology reports, and discharge letters.

To create the medical corpus, each entry from the EMC Dutch Clinical Corpus was added to a dataset. Furthermore, some noise

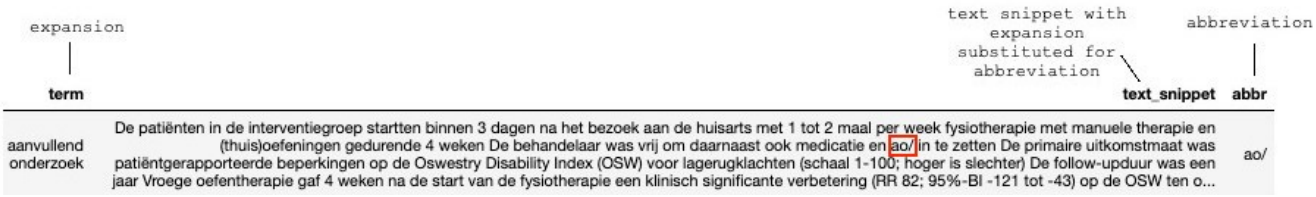

Figure 1: The Data

was removed from the notes using regex. The resulting dataset contained one column with in each row an individual entry. Another approach was used to add the second dataset using EuropePMC. Using the API the organization provides[3], open-access, Dutch, biomedical articles were downloaded as XML-files. Using BeautifulSoup[4], the text body of each article was extracted. Furthermore, the full article bodies were then split into snippets with a max. character amount of 1000 and added to the same medical corpus dataset containing the EMC Dutch Clinical Corpus entries. The final corpus consisted of 10021 medical text snippets.

3.1.3 *The Combined Dataset.*

As described in the MeDAL paper [19], reverse substitution was used to create a final dataset containing all the elements to facilitate the training and testing of the models. To do this, both the medical corpus and the biomedical abbreviations dataset were used. This new dataset contained 4326 text snippets from the medical corpus in which the expansions from the biomedical abbreviations dataset were replaced with their corresponding abbreviation. Two new columns were added containing the abbreviation and the expansion. An overview of the final combined dataset is given in Figure 1.

It is important to consider that the combined dataset is unbalanced. For some abbreviations, one expansion or term occurred predominantly. For other abbreviations, only one expansion was found in the medical corpus. Figure 2 shows that although the curated abbreviations have multiple expansions, often only one unique abbreviation-expansion pair was found in the text. How this class imbalance is dealt with, is further elaborated upon in Section 3.3.

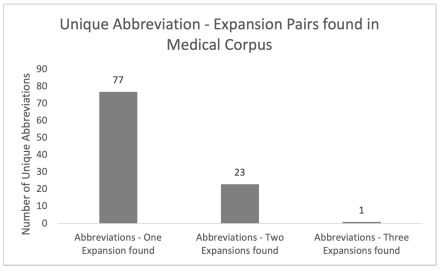

Figure 2: The Occurrence of Unique Abbreviation-Expansion Pairs

[3]https://europepmc.org/RestfulWebService
[4]https://www.crummy.com/software/BeautifulSoup/

3.2 Models

In this section, the set-up of multiple models is described. First, the baseline model is elaborated upon. Next, the multi-class classifier model and its two settings are explained. In the last section, the binary classifier is introduced. It is important to note that for these models, the Simple Transformers library[5] was used. This Natural Language Processing library is designed to facilitate the usage of transformers and is built on the Hugging Face Transformers library[6]. Another important note: All the models were trained using RobBERT-base (v1)[7] despite RobBERT-base (v2)'s existence (see Section 2.4.1). This was done because RobBERT-base (v2) was not yet available in the Hugging Face Transformers library at the moment of creating the models.

3.2.1 *Most Frequent Class Baseline.*

The baseline model was set up as follows: for each abbreviation that needed disambiguation, the most prevalent expansion was chosen. Meaning that in a perfectly balanced dataset where each abbreviation has two expansions, the accuracy of the model should be 50%. This type of baseline model is called the Most Frequent Class Baseline[8] and is suitable for unbalanced problems. To determine the most prevalent classes or expansions for each abbreviation, the whole dataset was considered.

3.2.2 *Multi-Class Classifier.*

A model that was explored in this research, is the multi-class classifier model. The chosen language model for this classifier was RobBERT-base (v1). RobBERT-base has a similar structure to BERT-base, sporting 12 self-attention layers with 12 heads [6]. As for hyper-parameters, mostly default parameters were used. The batch size for training was 16 instead of the default value of 32. This was done to preserve GPU, as this model was trained on a personal computer. The learning rate was kept at default value. Furthermore, the number of epochs was 5, this allowed for a manageable training time and prevented over-fitting. The full hyper-parameter overview can be seen in Table 1. For the training of the model, the combined dataset as described in Section 3.1.3 was divided into train-test split of 70/30.

[5]https://simpletransformers.ai/
[6]https://huggingface.co/
[7]https://huggingface.co/pdelobelle/robBERT-base
[8]https://web.stanford.edu/ jurafsky/slp3/8.pdf

	Multi-Classification Model	Binary Classifier Model
User Input	Abbreviation + Bio-Medical Sentence/Text	Abbreviation + Bio-Medical Sentence/Text
Model Input	**Abbreviation** + Bio-Medical Sentence/Text	**Expansion** + Bio-Medical Sentence/Text
Model Output	Multi-Class Label (Expansion)	Binary Label
Label Range	0-95	0 or 1
User Output	Expansion	Expansion

Table 2: Comparison between Multi-Classification Model and Binary Classification: User and Model View

Hyper-Parameter	Value
Batch Size	16
Learning Rate	4e-5
Epochs	5
Hidden Layers	12
Attention Heads	12

Table 1: Hyper-parameters used for the Multi-Class Classifier (RobBERT)

Lastly, for each abbreviation and the given context or text snippet, a label or class was given to the model. These labels corresponded with the full expansions, so for N full expansions, there were N number of labels. The total number of expansions and thus labels/classes was 95.

Setting 1: Not showing all the Abbreviations to the Model during training. Two configurations of this multi-classification model were tested in this research, both pertaining to the splitting of the training data. This first setting used the same train-test split of 70/30. Using this configuration, the model was trained on 3028 data samples.

Setting 2: Showing all the Abbreviations to the Model beforehand. In the second configuration, the model sees all the abbreviations during training, the samples are stratified, and consequently, all 95 expansions and thus all abbreviations are seen during training. Using this configuration, the model was trained on 3007 data samples. The slight difference between this number of data samples and that of setting 1 is caused by having to remove some samples to be able to stratify.

3.2.3 *Binary Classifier.*

Hyper-Parameter	Value
Batch Size	32
Learning Rate	4e-5
Epochs	5
Hidden Layers	12
Attention Heads	12

Table 3: Hyper-parameters used for the Binary Classifier (RobBERT)

The second approach to the classification problem was creating a binary classifier. This classifier consists of a RobBERT (v1) model and additional code which permits the different formatting steps as described further in this section. In Table 3, it can be seen that the hyper-parameters have not changed much compared to those of the multi-classification model, except that the batch size was 32 instead of 16. The binary model was trained on 6923 data samples, as each row in the training data, as seen in Section 3.2.2, had to be expanded into multiple rows representing the binary problem.

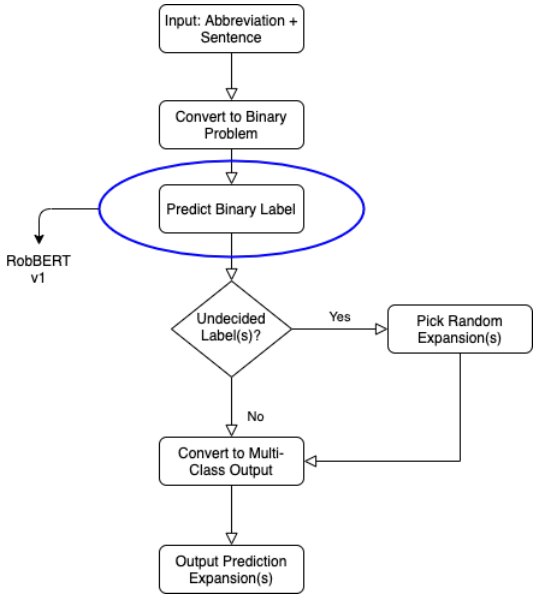

Figure 3: The Structure of the Custom Binary Classifier

Similar to the multi-class classifier, this binary classifier classifies into the same 95 classes as presented in multi-classification model (see Section 3.2.2), but presented the data to RobBERT-base v1 in a different manner. In the binary classifier, the user inputs an abbreviation and a context, either a sentence or a short note/text, similar to the multi-classification model. The model, however, was then presented each possible expansion *for that abbreviation* within the given context or sentence and was asked to predict a binary label, i.e. the input data was then converted to a binary problem. For each of these options, a 0 or 1 was predicted, where 1 means the model deemed this expansion to be right for the sentence and vice versa. In the ideal situation, the model only classifies the right expansion for the abbreviation as 1. However, in the case of indecisiveness, either by classifying more than one expansion-sentence pair as right or by having only zeros predicted by the model, the model picked an

	Multi-Classifier - Setting 1	Multi-Classifier - Setting 2	Binary Classifier
Unbalanced Test Dataset	649 samples	645 samples	1484 samples
Balanced Test Dataset	17 samples	17 samples	44 samples

Table 4: Number of Testing Samples for the different Models and Configurations

expansion randomly. The model then converted these outputs back to multi-class labels. This was done to be able to compare accuracy scores to the multi-classification model. The exact structure of this custom model can be viewed in Section 3. For a comparison between the configurations of both models, see Table 2.

3.3 Dealing with Class Imbalance

After training, the models (see Section 3.2) were tasked with predicting either multi-class or binary labels. However, since the combined dataset was unbalanced (see Section 3.1.3 and Figure 2), it is important to check its impact on the models' performance. To test if the models were actually disambiguating and not simply predicting the majority class, predictions were made on both the unbalanced and a balanced dataset.

The unbalanced test set was created from the combined dataset, containing 15% of that data, containing 649 samples for setting 1 of the multi-class classifier, 645 samples for setting 2 of the multi-class classifier, and 1484 samples when expanded to the binary classification model. The balanced test set could be described as an extraction of the unbalanced test set. It is filtered so that it only contains abbreviation-sentence pairs for which two or more expansions or classes have been found in the medical corpus. Each of these abbreviation-expansion pair occurs once in the dataset. This means that the Most Frequent Class Baseline baseline classifier should achieve around 50% in accuracy scores, as most abbreviations have two meanings, exceptions aside. As a result, this balanced dataset contained 17 samples that expanded to 44 samples to cater to the binary classifier. For the final overview, see Table 4.

4 RESULTS

The results of the model are presented in Table 5 and Table 6. For the evaluation metrics, accuracy/micro F1-score and the macro F1-scores were decided upon. Accuracy is the proportion of predictions that were rightfully classified. The macro-averaged F1-score is the mean of the macro-averaged recall and precision scores whereas the micro-averaged F1-score is the mean of the micro-averaged recall and precision scores. The F1-score metric is useful in assessing the quality of a model as it is seen as the harmonic mean between precision and recall. The difference between macro and micro F1-scores is that in the macro F1-score, each label or class is given equal importance regardless of how prevalent samples of the classes are in the dataset. Therefore, the macro-f1 score is expected to be low for models that only successfully predict the most common classes without rightfully predicting the rare classes. The micro F1-score, however, assigns more weight to more prevalent classes in the dataset. In multi-classification problems, the micro F1-score is the same as the accuracy score. Because of the variety in class sizes, i.e. the imbalance of the classes in the used dataset, the macro F1-score is important to consider when looking at the performance on the unbalanced dataset. The closer the F1-score is to 1, the more desirable the model is for the selected problem.

4.1 The Accuracy/Micro F1-scores

As presented in Table 5, the accuracy/micro F1-scores were rather high for all models on the unbalanced dataset. This is a logical consequence of that imbalance. The reasoning behind this was that certain classes dominate the dataset, which caused more focus on them during training and testing. When considering the balanced dataset, however, the accuracy for all models dropped. Accuracy-wise, the baseline performed the least well on the balanced dataset with a score of 0.45, whereas setting 2 of the multi-class classifier performed the least well (0.96) when regarding the unbalanced dataset. The binary classifier outperformed all models on both the unbalanced and the balanced datasets with scores of 0.99 and 0.82 respectfully.

Accuracy/Micro F1-score	Unbalanced Dataset	Balanced Dataset
Multi-Classifier - Setting 1	0.97	0.55
Multi-Classifier - Setting 2	0.96	0.55
Binary Classifier	0.99	0.82
Baseline Classifier	0.97	0.45

Table 5: Accuracy Scores for the different Models and Configurations

4.2 The Macro F1-scores

As presented in Table 6, the different models resulted in a diverse range of macro F1-scores. For both the unbalanced dataset and the balanced dataset, the Binary Classifier performed best with scores of 0.89 and 0.78 respectfully. The multi-class classifiers performed the least well on the unbalanced dataset with both scoring a value of 0.68 for the macro F1-score. However, for the balanced dataset, the baseline classifier performed the least well with a score of 0.28.

Macro F1-score	Unbalanced Dataset	Balanced Dataset
Multi-Classifier - Setting 1	0.68	0.41
Multi-Classifier - Setting 2	0.68	0.41
Binary Classifier	0.89	0.78
Baseline Classifier	0.81	0.28

Table 6: Macro F1-Scores for the different Models and Configurations

5 DISCUSSION

5.1 Analysis of Results

In Section 4, the results of the different configurations and models were presented. In this section, these results are analyzed. The results showed that on the unbalanced dataset, all the classifiers performed well according to the accuracy metric (see Table 5). However, when looking at the balanced dataset, this performance dropped. As mentioned in Section 4 the high performance on the unbalanced dataset is a logical consequence of the skewed distribution of the classes within the dataset. It is important to consider that the baseline had an advantage over the multi-class classifier. Namely, the baseline classifies into the most prevalent class *for the given abbreviation*, whereas the multi-class classifier did not "know" which classes belong to which abbreviation and had to pick between 95 classes. This is reflected in the results, where the baseline classifier was shown to perform better or equally as well as the two multi-class classifiers when looking at the accuracy (see Table 5). When considering the macro F1-scores for the unbalanced dataset, the baseline also outperformed the multi-class classifier.

However, the advantage of the baseline is diminished slightly when testing on the balanced dataset, because picking the most prevalent class is then no longer an effective method. This is reflected in the accuracy scores in Section 4, Table 5 where it was shown that in that when using the balanced dataset, the multi-class classifiers do outperformed the baseline model. When considering the macro F1-scores for the balanced dataset (see Table 6), the same pattern could be discerned. The baseline model performed worse than the multi-class classifiers. The most probable explanation is that the baseline model was worse in classifying into more rare classes than the multi-class classifiers.

Lastly, the binary classifier had the highest accuracy score and macro F1-score in both datasets (see Table 5 and 6). Showing that it can perform high on both unbalanced and balanced data, thus is able to process rare classes as well. This is plausible, as it overcomes the main disadvantage the multi-class classifier had. Namely, not knowing which abbreviations correspond to which set of classes. The way the binary classifier worked, is that for each abbreviation, *only* the corresponding classes or expansions were presented to model (see Figure 3 and Section 3.2.3). This was done in the form of a binary problem. The binary model, therefore, "knew" which set of classes are options for the given abbreviation, just like the baseline model did.

Another factor in the high scores for this model could be the way the problem was presented to the model. For the multi-class classifiers, the models regarded the abbreviation and the context, whereas the binary model looked at full expansions and the context. It could be that the binary model performed better because it derived more semantic context from the full expansions instead of the abbreviations.

Lastly, it can be concluded from both the macro F1-scores and the accuracy scores corresponding to the multi-class classifiers (see Table 5 and 6), that showing all the abbreviations to the model during training did not make a great difference within the borders of this research.

5.2 Error Analysis of Binary Classifier

5.3 Research Questions

In this research, the following questions were asked: "To what extent can abbreviation disambiguation for Dutch medical text be improved upon through the use of language models?" and "What is an appropriate dataset for the task?". In this section, these two questions are answered.

To what extent can abbreviation disambiguation for Dutch medical text be improved upon through the use of language models?
For this question, the performance of the proposed models was compared to a proposed baseline model. As seen in the results and described in the analysis of these results (see Section 4 and 5.1), it can be concluded that when using a balanced dataset, both the multi-class classifier models and the binary model outperformed the baseline. On the unbalanced dataset, only the binary classifier improved over the baseline. The binary classifier also performs the best overall. Therefore, the research can be answered positively, the use of language models did improve remarkably when compared to a baseline, with the best model performing 50% better on a balanced dataset.

What is an appropriate dataset for the task?
As seen in Section 3.1, an appropriate dataset was made for the task. Roughly following the method for the MeDAL dataset, as described in Section 2.1, three types of data were necessary. Namely, a biomedical corpus, biomedical abbreviations, and a dataset in which expansions in this biomedical corpus dataset were substituted for their abbreviations. Both of these datasets were curated and have proven to successfully aid in completing the problem. However, the dataset cannot be published publicly because of the private nature of the information it contains.

5.4 Limitations

5.4.1 Models and Data.

After conducting the research, multiple areas of improvement have been identified. In this section, these are elaborated upon.

The first possible improvement lies within the collection of suitable data for the problem. The gathering of Dutch medical data proved to be difficult due to multiple reasons. First, most scientific articles are published in English, so finding Dutch text that could be used in the substitution dataset (see the text snippet in Figure 1) was not an easy task. Finding the abbreviations was a challenge as well because of this exact reason. Another reason why finding biomedical contexts was difficult, is the private nature of clinical notes. This hampers the creation of a database containing this type of sensible information. Luckily, such a database was found and permission was given to be able to use it in context of this research. However, because of the scarcity of the data, the quality assessment of the data was less strict than it would have been in the case of an abundance of data. One particular future improvement is that there could be a better match between the medicine fields of the medical corpus and the abbreviations. Both datasets covered multiple medicine fields and there was enough overlap between the abbreviation-expansion tuples and the expansions in the medical corpus to be able to tackle the problem. However, this overlap could have been increased if the abbreviations and the medical texts were

all from the same medical fields. The expectation is that this would have created more training data and thus better models. Furthermore, not all sources of data for the biomedical abbreviations were scientifically peer-reviewed. This means that the validity of the biomedical abbreviations cannot be proved.

Another aspect of the models that can be improved upon, is that they were trained on an unbalanced dataset. Were the model to be trained on a balanced dataset, it would have been easier for the model to predict the less frequent classes. Also, the baseline would have been a lot weaker on the test set. However, because of the data-hungry nature of these models, this was avoided. Furthermore, the current results do show in what frequency some of the abbreviations/expansions may occur in real life, so it is still useful to know how unbalanced data affects the performance of these models in practice.

Something that could be improved upon regarding the binary model is the way the model picks a class when no decision could be made by RobBERT v1 (see the structure of the binary classifier in Figure 3). In that case, a random expansion is picked in the case of indecisiveness. However, this could be done more specifically, for example by comparing the raw output scores for each binary problem and picking the highest score out of the options, even when the model itself classifies all options as 0 or multiple options as 1.

Furthermore, the performance of all the models was tested on a rather small, balanced test set (see Table 4), which means the results and especially the evaluation metrics values can be expected to turn out differently and more representative when a bigger test set is used.

Finally, there is one more possible improvement in the set-up of the models, namely the way the substitutions have been handled. In the MeDAL paper (see Section 2.1), a certain percentage of expansions were substituted for their abbreviation, whereas in the set-up proposed in this research all the expansions were substituted. The performance of the proposed models could have been increased if the same method was used. However, because of computing power and memory limitations, this has not been tested in this paper.

5.4.2 Possible Use of Models and Ethics.

In the introduction (see Section 1) it was explained that wrongful interpretation of abbreviations in medical notes between medical professionals can have grave consequences. Therefore, the use of a Dutch biomedical disambiguation model could save lives. Consequently, this Section is dedicated to a brief description of how, and in which form, the proposed models can be used in the medical field and what the implications are.

All the models could, in the future, be used in practice, but only if presented in a form that is user-friendly for a medical professional. The medical professional can input a clinical note or sentence and an abbreviation and both the multi-class classifier and the binary classifier will return an expansion. However, for this application to be ethical, the decision-making processes of the models should be made transparent. Medical professionals should be able to know how the model obtains a certain expansion. Furthermore, blindly accepting the results without critical reflection on this decision-making process is not feasible. Also, the models should present how certain some outcomes are. For example, the binary model, even with the improved upon process of dealing with indecisiveness, should present that the presented output is not as certain as some other outputs may be.

Lastly, another important consideration when using these models is the responsibility when something goes wrong. The application of this model in the field should come with clear rules about where the responsibility lies in the case of malfunctioning of the model.

In addition to the possible usage of the proposed models, the curated dataset could also be used on a broader scale, for example for large-scale analysis of medical notes. The dataset, created for the abbreviation disambiguation task, could be used to pre-train models for different natural language understanding tasks. In this research, a pre-trained model (RobBERT v1) was fine-tuned on this dataset for disambiguation, however, this disambiguation task can also be used to pre-train a model for other medical NLP applications. As seen in MeDAL [19], the resulting pre-trained model can then be fine-tuned on different useful downstream tasks, such as diagnosis prediction and mortality prediction.

5.5 Future Work

In this section, it is assessed what work still needs to be done to advance this field of research.

Starting with the most common research problem, namely, to find more and better data. Gathering more specific biomedical abbreviations from different medical fields and gathering a more elaborate medical corpus will most likely improve the models.

Furthermore, different set-ups and configurations should be explored to see what works best for this research problem. In the analysis (see Section 5.1), it was concluded that showing all the abbreviations to the model during training barely impacted the results. However, multiple factors could have contributed to this, such as the data quality and the small, balanced test size. Further research can show what the impacts of this set-up can be in better circumstances. Also, testing the impact of different levels or probabilities for expansion substitutions, as mentioned in both Section 2.1 and Section 5.4.1 could make for interesting results.

Next, it could be interesting to see how models, pre-trained on the dataset/the abbreviation disambiguation task perform as input for other downstream tasks. In the MeDAL paper, as explained in Section 2.1, it was shown that models, pre-trained on the language disambiguation task, were useful input for other tasks. Due to time constraints, this was not tested during this research.

Additionally, research must be done to discover the actual prevalence of the different classes. This because, in Section 5.4.1, it was concluded that training and testing on unbalanced data may not be entirely useless. However, it is not easy to discern at this stage if this imbalance is a reflection of the real world or if this is due to the scarcity of data.

Lastly, research should be done on the ethics of using these types of models in the medical field. Interesting research topics could be the responsibility question: what exactly should be included in the output of these models and how the model must present its results to the user? This was also briefly discussed in Section 5.4.2.

6 CONCLUSION

In this paper, a Dutch medical abbreviation disambiguation model was created. This was done because of the lack of such a model in the Dutch language, because of the societal relevance of attempting to lessen communication issues between medical professionals, but most importantly, to act as a basis to help natural language models better understand medical free-form text. The model may also be suited as input for other medical NLP tasks. The following two research questions were solved in this research: to what extent can the abbreviation disambiguation for Dutch medical text be improved upon through the use of language models? and what is an appropriate dataset for the task? To be able to answer these questions, a database suitable for the problem was to be curated and different models had to be created and tested. For the former aspect, a suitable dataset was successfully built despite the scarcity of Dutch medical data. This dataset, containing text snippets in which expansions were substituted for their corresponding abbreviation, was the answer to the second research question. Furthermore, four models were proposed: two differently set up multi-class classifiers, a binary model, and a baseline model. The binary model was made so that it could predict multi-class labels and could therefore be compared to the multi-class classifiers and the baseline model.

However, because the data was unbalanced, a balanced dataset had to be extracted. This was done to be able to discern to what extent each model could actually disambiguate instead of simply predicting the dominant class for each abbreviation. Analysis of the results showed that the binary model outperformed all other models on both the balanced and unbalanced datasets. Furthermore, the multi-class classifiers showed higher accuracy and macro F1-scores on the balanced datasets, but performed equally or even worse on the unbalanced dataset according to these evaluation metrics.

This research showed that correctly disambiguating Dutch medical abbreviations comes with certain challenges, both technical and ethical. However, this research produced a useful dataset which, in further research, could be used as input for pre-training. In addition to the dataset, a binary model that is capable of disambiguating medical abbreviations was created. Overall, this research helps the advancement of field of Dutch medical NLP and creates room for future work.

REFERENCES

[1] Zubair Afzal, Ewoud Pons, Ning Kang, Miriam CJM Sturkenboom, Martijn J Schuemie, and Jan A Kors. 2014. ContextD: an algorithm to identify contextual properties of medical terms in a Dutch clinical corpus. *BMC bioinformatics* 15, 1 (2014), 1–12.

[2] Simone A Cammel, Marit S De Vos, Daphne van Soest, Kristina M Hettne, Fred Boer, Ewout W Steyerberg, and Hileen Boosman. 2020. How to automatically turn patient experience free-text responses into actionable insights: a natural language programming (NLP) approach. *BMC Medical Informatics and Decision Making* 20 (2020), 1–10.

[3] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555* (2020).

[4] The Europe PMC Consortium. 2014. Europe PMC: a full-text literature database for the life sciences and platform for innovation. *Nucleic Acids Research* 43, D1 (11 2014), D1042–D1048. https://doi.org/10.1093/nar/gku1061 arXiv:https://academic.oup.com/nar/article-pdf/43/D1/D1042/7316204/gku1061.pdf

[5] Ronald Cornet, Armand Van Eldik, and Nicolette De Keizer. 2012. Inventory of tools for Dutch clinical language processing.. In *MIE*. 245–249.

[6] Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. Robbert: a dutch roberta-based language model. *arXiv preprint arXiv:2001.06286* (2020).

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[8] Kevin Donnelly et al. 2006. SNOMED-CT: The advanced terminology and coding system for eHealth. *Studies in health technology and informatics* 121 (2006), 279.

[9] Paulina Grnarova, Florian Schmidt, Stephanie L Hyland, and Carsten Eickhoff. 2016. Neural document embeddings for intensive care patient mortality prediction. *arXiv preprint arXiv:1612.00467* (2016).

[10] Henk Harkema, John N Dowling, Tyler Thornblade, and Wendy W Chapman. 2009. ConText: an algorithm for determining negation, experiencer, and temporal status from clinical reports. *Journal of biomedical informatics* 42, 5 (2009), 839–851.

[11] Yanming Huang, Y. Jiang, Touhidul Hasan, Q. Jiang, and Chao Li. 2018. A Topic BiLSTM Model for Sentiment Classification. In *Proceedings of the 2nd International Conference on Innovation in Artificial Intelligence* (Shanghai, China) *(ICIAI '18)*. Association for Computing Machinery, New York, NY, USA, 143–147. https://doi.org/10.1145/3194206.3194240

[12] Ivy Fenton Kuhn. 2007. Abbreviations and acronyms in healthcare: when shorter isn't sweeter. *Pediatric nursing* 33, 5 (2007).

[13] Yue Li, Pratheeksha Nair, Xing Han Lu, Zhi Wen, Yuening Wang, Amir Ardalan Kalantari Dehaghi, Yan Miao, Weiqi Liu, Tamas Ordog, Joanna M Biernacka, et al. 2020. Inferring multimodal latent topics from electronic health records. *Nature communications* 11, 1 (2020), 1–17.

[14] J Martijn Nobel, Sander Puts, Frans CH Bakers, Simon GF Robben, and André LAJ Dekker. 2020. Natural Language Processing in Dutch Free Text Radiology Reports: Challenges in a Small Language Area Staging Pulmonary Oncology. *Journal of digital imaging* (2020), 1–7.

[15] Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A Monolingual Approach to Contextualized Word Embeddings for Mid-Resource Languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 1703–1714. https://www.aclweb.org/anthology/2020.acl-main.156

[16] S Trent Rosenbloom, Joshua C Denny, Hua Xu, Nancy Lorenzi, William W Stead, and Kevin B Johnson. 2011. Data from clinical notes: a perspective on the tension between structure and flexible documentation. *Journal of the American Medical Informatics Association* 18, 2 (01 2011), 181–186. https://doi.org/10.1136/jamia.2010.007237 arXiv:https://academic.oup.com/jamia/article-pdf/18/2/181/6035310/18-2-181.pdf

[17] Erik Tjong Kim Sang, Ben de Vries, Wouter Smink, Bernard Veldkamp, Gerben Westerhof, and Anneke Sools. 2019. De-identification of Dutch Medical Text. In *2nd Healthcare Text Analytics Conference (HealTAC2019). Cardiff, Wales, UK.*

[18] Eric Topol. 2019. Deep medicine. *How Artificial Intelligence Can Make Healthcare Human Again. Basic Books* (2019).

[19] Zhi Wen, Xing Han Lu, and Siva Reddy. 2020. MeDAL: Medical Abbreviation Disambiguation Dataset for Natural Language Understanding Pretraining. *arXiv preprint arXiv:2012.13978* (2020).

[20] Wei Zhou, Vetle I Torvik, and Neil R Smalheiser. 2006. ADAM: another database of abbreviations in MEDLINE. *Bioinformatics* 22, 22 (2006), 2813–2818.