Acronym expansion in Dutch: Improving out-expansion performance with BERT and SBERT

submitted in partial fulfillment for the degree of master of science

Jesher Appels
12865362

master information studies
data science
faculty of science
university of amsterdam

July 07, 2022

	Internal Supervisor	**Examiner**
Title, Name	J.L.M Pereira	Prof. Dr. P.T. Groth
Affiliation	University of Amsterdam	University of Amsterdam
Email	j.p.pereira@uva.nl	p.t.groth@uva.nl

Universiteit van Amsterdam

Acronym expansion in Dutch: Improving out-expansion performance with BERT and SBERT

J.R.H Appels
University of Amsterdam
jesher.appels@student.uva.nl

ABSTRACT

Acronyms are an essential part of most languages and make it possible to merge a set of words into just a few letters. Despite their practicality in saving space in text, they also come with the disadvantage of losing context if not adequately presented. For example, when the exact meaning of an acronym is not known to the reader or associated with the wrong expansion. This type of ambiguity might happen when the expansion of an acronym is not exactly stated in the document. This is known as an out-expansion since the acronym's expansion must come from outside the document. Acronym expander systems aim to mitigate this problem by matching an acronym to its proper expansion. There is considerable research on acronym expansion methods in English, but none of these methods have been extensively evaluated in the Dutch language. In this study, we experimented with various methods such as Doc2Vec, SBERT, and BERT for acronym expansion on a human-annotated Dutch data set. We used the AcX acronym expander system to evaluate these methods and mainly focused on out-expansion acronyms. The results show that the current methods available within the AcX system are suitable for acronym expansion in Dutch and that SBERT offers a slight improvement over the initial Doc2Vec and CCV methods. The largest multilingual SBERT model that we evaluated, "paraphrase-multilingual-mpnet-base-v2", yielded an F1 score of 47,95%. This was an improvement of 0,83% compared to the second-best model, which uses Doc2Vec.

Keywords: Acronym, Expansion, Extraction, Dutch BERT, Multilingual SBERT

GitHub: https://github.com/JesherA/AcExSy

1 INTRODUCTION

In most professional and casual settings, using acronyms for common terms and names is standard practice and allows people to save space and time in their text writing. Acronyms are, therefore, an essential part of most languages, including the Dutch language. For humans and natural language processing (NLP) processes, the expansion of a particular acronym can be helpful. For example, a human better understands a given article if he/she knows the exact meaning of an acronym. Likewise, for search engines and document retrieval algorithms, acronym expansion allows the users of such systems to search for the article of interest based on an acronym and its field of use. For example, the terms "ILS Aviation" would return articles related to "instrument landing systems," whereas the terms "ILS Medical" would return articles related to Immediate Life Support.

An acronym-expansion pair consists of two parts, the acronym, sometimes also called short-form or abbreviation, and the expansion, also known as long-form. The acronym resembles the expansion in a combination of fewer letters or numbers, for example, the acronym AC and the expansion Air-Conditioning. The most straightforward way to expand acronyms is applicable when the acronym and expansion are present within the same document, which is known as in-expansion. However, one of the challenges when matching an acronym with an expansion is the absence of the expansion in the document. This is known as out-expansion since the expansion must be derived from general knowledge and context. One of the challenges with out-expansion is that one acronym can have multiple expansions. For example, An aircraft is often referred to as AC in the aviation industry. There are also end-to-end systems that aim to find both in- and out expansion acronyms in a given document. This research focuses on the end-to-end AcX system [1], which uses text representers in the out-expansion part to transform and match documents based on their vector representations. These vectors are created using algorithms such as Doc2Vec and BERT. Matching the documents is necessary since the out-expansions are retrieved from training documents that contain the complete acronym expansion pair.

There is considerable research on acronym expansion in English [2, 3, 4, 5, 6]. However, the Dutch language lags far behind since in-expansion methods such as the ones from Schwartz and Hearst [2] and Veyseh et al. [3], and out-expansion methods such as the ones from Veyseh et al. [3], Singh and Kumar [4], and Egan and Bohannon [7] have not been extensively evaluated on a Dutch data set. The lack of research in the Dutch language provides three main reasons for the relevance of this research. First, there has been no evaluation of an out-expansion and end-to-end system for expanding acronyms in the Dutch language. Second, the field lacks a solid Dutch benchmark data set that uses general topics and entire documents. Lastly, the AcX system does not

utilize Bidirectional Encoder Representations from Transformers (BERT) models for text representation purposes. The system uses these text representations to find matching documents with the same context and thus possibly the same acronyms. BERT-based models have proven to perform well in other fields of NLP [8, 9, 7] and can therefore provide additional performance in Dutch. In addition, not much is known about the performance of language-specific BERT and multilingual SBERT models, specifically for the AcX[1] end-to-end acronym expansion system that was used during this research.

This research aims to counter these gaps by creating a Dutch benchmark data set, utilizing BERT and SBERT models as text representers, and ultimately creating an end-to-end system for the Dutch language. A BERT-based acronym expander system for the Dutch language will advance the field of NLP specifically for Dutch and possibly provide new insights that can improve acronym expansion for other (non-English) West Germanic languages. It is for these reasons that the research question is formulated as follows:

To what extent can an end-to-end system for acronym expansion in the Dutch language be built upon AcX, and how does its performance for Dutch improve when using multilingual Sentence-BERT and Dutch-specific BERT models as text representer?

The following sub-questions support the main research question:

- How do the current end-to-end acronym expander methods perform on a Dutch data set, and how does this compare to the performance on the English Wikipedia data set?
- To what extent do the language-specific BERT and multilingual SBERT models attain better performance in the AcX system regarding precision, recall, and F1-score for a Dutch data set?

1.1 Remaining sections

The rest of this paper is structured in the following way. Section 2 contains the related literature and describes the existing state of acronym expander systems, the latest data sets used in this field, the usage of BERT-Models and SBERT models, and finally, the present state of Dutch BERT models. Section 3 contains the methodology and describes how the data was gathered, the different models used, and the affiliated evaluation methods. Subsequently, Section 4 contains the result of the various experiments, and Section 5 holds the discussion, where the performance is evaluated and summarised together with the shortcomings of this research. Lastly, Section 6 provides the conclusion and recommendations for future works.

2 RELATED WORKS

The interest in acronym expansion has increased in recent years, both in improving the already accurate rule-based strategies as well as in applying machine learning (ML) methods. Schwartz and Hearst' [2] rules-based method is one of the most cited in the field and forms the foundation for many other approaches, including MadDog [3]. MadDog's approach improves the candidate expansions using a series of rules where each one produces better fitting expansions than the previous one. For example, the "character match" and "Initial Capitals" rules. Character match evaluates if the initials in the expansion could form the acronym and initial capitals tries to match the possible capital letters of the expansion to the acronym. In addition, there is also a multilingual technique by Glas et al.[5] that uses word embedding to measure similarity between candidate acronyms and extensions. All three methods, Schwartz and Hearst [2], MadDog[3], and Glas et al.[5], have a precision ranging between 95% and 98% and recall between 50% and 85% on English data sets.

As for out-expansion, the main focus has been finding an efficient way to retrieve expansions from other sources, for example, other documents. Charbonnier and Wartena [6] focus on the scientific domain and use Word2Vec [10] to create context vectors from the acronyms and the words in the expansions. They then use cosine similarity to match these context vectors based on their similarity. Furthermore, Maddog [3] utilizes a deep sequential model as a sentence encoder, followed by a feed-forward network to match the input sentence with the most likely expansion. Egan and Bohannon [7] use BERT [9] to create sentence embeddings. Singh and Kumar experimented with different BERT models on the SciAD [7] data set. The performance of these models ranges between 89,8% for BERT-uncased, and 92,9% for SciBERT [11] uncased with a two-stage training regime. They also tested an ensemble method that achieved a macro-F1-score of 93,2%. Egan and Bohannon created a model using transformer-based language for sentence embeddings and a linear projection for each BIO (Beginning, Inside, and Outside) tag. They compared its results against different BERT models. In the end, their ensemble method yielded an F1 score of 91,22% on the SciAD data set.

Currently Maddog [3] and AcX [1] are the only two who have a publicly available end-to-end system. However, the creators of AcX argue that their system provides a framework that "easily" allows one to plug in for different in- and out-expansion methods and different data sets.

2.1 The AcX system

AcX is short for Acronym eXpansion [1], and the creators of this system aim to create an open-source end-to-end acronym expander. The system seeks to improve the current state of acronym expansion techniques and allows others to experiment with different methods that could lead to new technologies.

The end-to-end system works in two phases: the in-expansion phase followed by the out-expansion phase. The in-expansion part of the system is responsible for finding all acronyms and their possible expansion inside a given document. This part of the system utilizes rule-based algorithms like those from Veyseh et al.[3], and Schwartz and Hearst [2] to find acronyms. In the end-to-end system, this is done first over training documents to create a database with all the in-expansion pairs and the corresponding documents in which they appear.

The out-expansion phase of the system aims to find the expansions that are not present inside a given document. In order to do so, the system first encodes a set of train documents into their vector representation using various methods (Doc2Vec, BERT, SBERT). These text representations are coupled to the in-expander results, allowing the system to know which document contains which acronym and expansion. During the evaluation, the system tries to match the out-expansion acronyms to the correct expansion using a so-called predictor, which calculates the similarity between documents to find the correct acronym-expansion pair.

2.2 Data sets for acronym expansion

There are various commonly used in-expansion data sets, such as BioText, a data set provided by Schwartz and Hearst [2] consisting of acronyms from Biomedical papers, and BIOADI provided by Kuo and Lingtexts[12]. SciAI provided by Veyseh et al. [13] is another English in-expansion data set consisting of sentences from arXiv papers. As for out-expansion, common data sets are SciAD [13] (there is a revised version by Edgar [7]), which is an out-expansion version of SciAI and CSWiki (Computer Science Wikipedia) [14] which consists of Wikipedia articles from the Computer Science domain. Also, Van Oosten [15] used Dutch medical sentences and paragraphs and generated labeled data with reverse substitution. As for end-to-end acronym expansion, AcX [1] is the only one that has a human-annotated data set where the data consists of English Wikipedia articles.

2.3 BERT

Bidirectional Encoder Representations from Transformers, better known as BERT, were first introduced by Devlin et al. [9] in 2018 and have revolutionized the field of NLP.

BERT utilizes deep bidirectional representations to create general language representations, and these have three main advantages. 1) The initial use of unsupervised training methods, 2) the ability to work bidirectionally, and 3) most importantly, they can be fine-tuned for specific tasks by changing the in- and/or output layer(s). Training BERT models, therefore, consist of two distinct phases. First, the model is pre-trained in an unsupervised manner and uses unlabelled data. After that, the model is fine-tuned for a specific task using task-specific data. Hence, the architecture of BERT makes it possible to use a pre-trained model that has been trained on large amounts of data and improve its performance on specific tasks by fine-tuning that same model. In addition, it is not necessary to train for many iterations in the fine-tuning phase. In the original paper, the authors recommend fine-tuning for about three epochs [9]. The original BERT model was pre-trained on the BooksCorpus data set and the bodies of English Wikipedia articles. This initial training was done with two unsupervised tasks: Masked Language Modelling (MLM) and Next Sentence Prediction (NSP). The fine-tuning was done by altering the input and output layers for a specific task, meaning that the original creators could fine-tune their initial model for 11 specific benchmark tasks.

2.4 Dutch BERT Models

There are currently several different BERT-based models that are pre-trained explicitly on a Dutch corpus. By pre-training in Dutch, these models have learned its structure and word usage, which, in turn, should lead to better performing models. Specifically when compared to BERT models trained in multiple languages (multilingual). The two most notable Dutch BERT models are RobBERT [16] and BERTje [17].

The creators of RobBERT [16] saw the potential performance gains that a Dutch-specific model would bring over multilingual models. The original version, RobBERT-v1 is trained on the Dutch part of the OSCAR data set [18], which has a size of 39 GB and contains 6.6 billion words. The second version, RobBERT-v2 is trained on the same data set, but this time with the introduction of a Dutch tokenizer, which has improved its performance. Furthermore, both versions are pre-trained with the Robustly Optimized BERT pre-training Approach (RoBERTa) [19]. This training method, in essence, uses more training data, more epochs, and larger batch sizes.

BERTje, the other well-known Dutch BERT model, was an improvement over RobBERT-v1 [17]. The creators of BERTje do not indicate which data was used for pre-training, but Delobelle et al. [16] indicate that they used 12GB of data containing Wikipedia articles.

As for the performance of these models, RobBERT-v2 seems to perform better in most NLP tasks, as shown in Table 1. However, it is essential to note that for this research,

the primary usage of BERT is creating document embeddings, and a model's overall performance is a reasonable indicator of the quality of its embeddings since we normally only fine-tune the out-put layer of the model.

Model	RobBERT-V2	BERTje
	Accuracy	
Sentiment analysis	95,10%	93,00%
Finetuning on 10K examples	97,82%	93,10%
Part-of-Speech Tagging	96,40%	96,30%
Named Entity Recognition	89,08%	88,03%

Table 1: A comparison of the performance on different tasks between RobBERT-v2 and BERTje. RobBERT-V2 performs better in all the tasks

Furthermore, a recent study utilized Dutch BERT models for acronym expansion purposes. Van Oosten [15] trained the RobBERT-v1 model for binary and multi-class classification tasks. The binary classification aimed to determine whether or not an expansion belongs in a particular sentence or paragraph. In the multi-class classification setting, the expansions were used as labels, and the aim was to predict which of the 96 labels (expansions) best fit a given sentence or paragraph. Van Oosten's approach differs from that of other studies in the field, in particular the way in which the correct expansion is determined. In addition, the method looks at a limited number of expansions and requires the acronyms and expansions to be known in advance. This makes it not generalizable to other disciplines and the Dutch language.

2.5 Sentence-BERT

BERT has many advantages, but its original architecture is not perfect. For instance, the creators of sentence-BERT (SBERT) [20] indicate BERT's shortcomings with regard to semantic textual similarity (STS). STS aims to decide if two texts have a similar meaning. They state that BERT is not optimized for STS applications, which makes the algorithm computationally inefficient, leading to long execution times for this specific task. SBERT significantly reduces the time required for STS (from 65 hours to 5 seconds for a comparable task) by adding an additional abstraction level and encoding the semantic meaning of the entire sentence. Since SBERT uses the same transformer-framework as BERT, the pre-training and fine-tuning principles still apply.

Literature is absent of Dutch-specific SBERT models, and the model referred to on Stackoverflow (sent_bert_base _cased) has no described methods. However, there are multiple multilingual models which have been trained in a variety of languages, including Dutch. This research focuses in particular on; paraphrase-multilingual-MiniLM-L12-v2 (miniLM-SBERT) and paraphrase-multilingual-mpnet-base-v2 (ml-mpnet-SBERT), both created by the original SBERT research group [21].

2.5.1 miniLM-SBERT. MiniLM-SBERT is a multilingual SBERT model trained in over 50 languages [21]. It is a distilled model of Multilingual-MiniLM-L12-H384, which the Microsoft research group created. The model has a 384-dimensional dense vector space and a max sentence length of 128 tokens making it a relatively small transformer network, hence the name MiniLM. However, despite its name, its performance is not significantly lower on tasks such as sentence embeddings and semantic search. Also, according to the sentence-transformers website, the MiniLM models are "5 times faster and still offers good quality" [22] compared to all-mpnet-base-v2, which is the best performing SBERT during writing.

2.5.2 ml-mpnet-SBERT. Ml-mpnet-SBERT is a multilingual SBERT model trained in over 100 languages [21]. Its transformer network is based on xlm-roberta-base, which is pre-trained on 2.5TB of filtered CommonCrawl data containing 100 languages. The model has a 768-dimensional dense vector space and a maximum sentence length of 128 tokens. It is about twice as larges as miniLM-SBERT but offers better performance. Specifically, for sentence embedding tasks, the performance was 1,58% better, and for semantic search tasks, the performance was 2,49% better compared to miniLM-SBERT.

3 METHODOLOGY

The following section describes the methods applied during this study. It describes the annotation process and its resulting data set, the evaluation metrics, experimental setup with the models that were utilized.

3.1 Wikipedia data set

There is currently no high-quality human-annotated data set for the acronym expansion in Dutch. Van Oosten [15] created a Dutch data set for out-expansion with medical sentences and paragraphs. We did not consider using this data since it focuses on a specific domain, which does not fit this study's aim since we want our approach to be generalizable to the Dutch language. Furthermore, Van Oosten's data was created using reverse substitution, which requires one to know the acronyms of interest beforehand.

In order to fulfill the need for a high-quality annotated data set, we gathered 301 articles from the Dutch Wikipedia website that would be used as an evaluation set in the AcX system. Whereas other researchers [23, 24, 15] use individual sentences or paragraphs in which acronyms are used, we decided to use the entire document similar to [14, 25]. Using the entire document allows the initial methods (Doc2Vec

and CCV) and BERT-based models to capture the context in which an acronym is used, improving out-expansion performance. The articles are manually selected from different fields since many Dutch Wikipedia articles do not include acronyms. Also, with the aim to keep the data set diverse and general in terms of document length and lexical diversity. This should ensure that any future breakthroughs in the field of acronym-expanders are to some extent generalizable to the entire Dutch language.

Figure 1 shows a histogram with the number of tokens in each document within the corpus. These documents were first annotated, and then the same documents were used for the evaluation process. Since BERT has its own tokenizer, adding additional processing steps is not necessary. Figure 1 also shows a small number of relatively large and small documents within the corpus. Large documents help to enhance generalizability since they may contain many different acronyms. However, these longer documents might be bothersome for the annotators as it takes a long time to annotate, and this could lead to mistakes. Contrariwise, short documents are easy to annotate but might not provide enough context to use out-expansion properly.

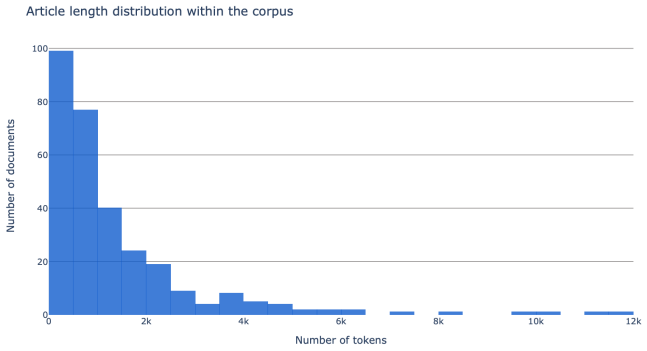

Figure 1: The length distribution of the articles that where used for annotation and evaluation

Another way to express the document divergence is the lexical diversity described by Malvern et al. [26]. This measure indicates that using a wide range of vocabulary is often associated with more complex text, so the higher the lexical diversity, the higher the difficulty level. As for this Dutch Wikipedia data set, the average lexical diversity score is 0,44, which is associated with a difficult or undergraduate reading level. The English end-to-end Wikipedia data compares with an average lexical diversity score of 0,45.

3.2 Annotation system

Figure 2 shows a user-flow diagram of the annotation system, which demonstrates the web pages with which the user interacts. The website was created in collaboration with three other students, and the candidate of this paper created the initial design of the system's front end. In the first step, the user registers with his/her email and selects a specific language to annotate since the pages that follow are language-specific.

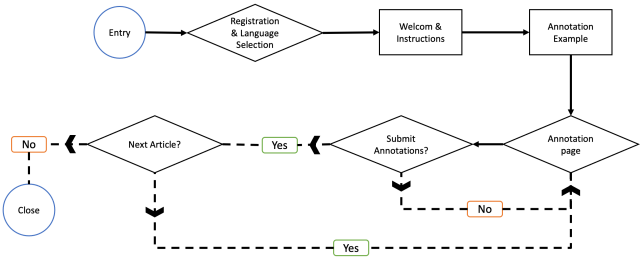

Figure 2: A graphical overview of the annotation system showing the process the users has to go though in order to annotate a article.

The following two pages contain instructions and examples to ensure that the annotator has a clear sense of how the annotations must be documented. The annotation page itself has three main fields: one for the text to be annotated, another for the in-expansions, and another for the out expansions. In addition, there is also a drop-down field where the user should provide the language of the expansion. After the user has submitted the annotations, he/she can decide to either continue annotating another article or stop annotating and leave the website. The users generally consist of other students with a similar background (information studies, computer science, etc.) and family members. Every document is annotated by two different users. The annotations are saved per user per document in .json files containing acronyms-expansion pairs and language. Having the articles annotated by two different users provides the first form of data validity. If there are any differences between the two annotations, the document in question was checked and corrected accordingly by the candidate of this paper and another graduate student. Finally, the user annotations and Wikipedia articles are combined into .pickle files which serve as input for the AcX system. BERT-based models use the raw text as input, so there is no need for additional pre-processing such as stop-word and special character removal.

3.3 Final annotated data set

The final data set used for the AcX system contains 94 annotated Dutch Wikipedia articles, annotated by 11 different annotators and checked by two graduate students. 63 of the 94 annotated articles were checked and cleaned by the candidate of this paper. The code for cleaning and transforming the data into a format that the AcX system can use was also created by the candidate. Figure 3 shows an overview of the data characteristics. The data set contains 1421 acronym

expansion pairs, of which 697 unique pairs. 827 (58%) of the expansions were out-expansion pairs, and 594 (42%) were in-expansion. As for the language in which the acronyms originated, 50% of the acronyms were Dutch, 40% were English, and 10% were from another language or not submitted (none of above and other language). The large amount of English acronyms highlights the extensive use of English words in the Dutch language. However, this also leads to ambiguity since some of these words are generally known by their Dutch expansion. An example of this is the acronym DNA which is a commonly used acronym in Dutch. However, the Dutch translation is "Desoxyribonucleïnezuur" and not Deoxyribonucleic acid which forms the acronym DNA.

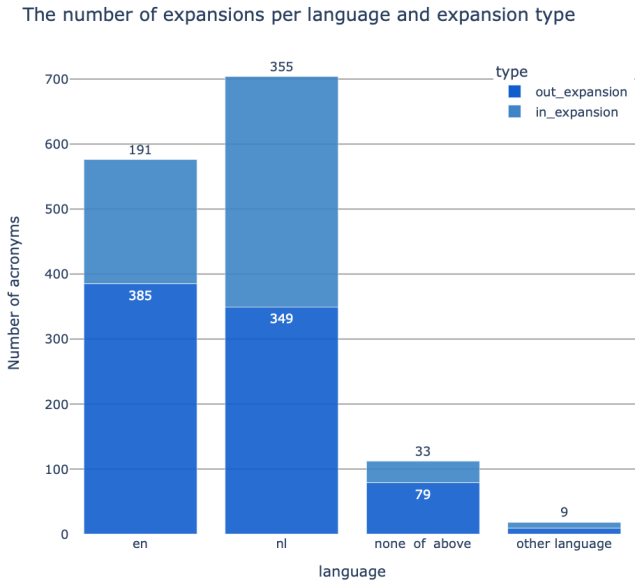

Figure 3: The key numbers of the evaluation data set showing that most acronyms-expansion pairs are Dutch, but that there are also a lot of English pairs.

Figure 4 shows a histogram with the number of documents per number of acronyms in bins of 10. The figure shows that most documents contain between 0 to 10 acronyms, with the number of documents decreasing rapidly as the number of acronyms increases.

Furthermore, we calculated the Inter-Annotator Agreement (IAA) among our annotators using Krippendorff's alpha [27] which provides an indication for the clarity of the instructions and reproducibility of the annotation tasks. Alpha ranges between 0 and 1, where a number close to one is associated with better matching annotation-expansion pairs between the two annotators. Krippendorff's alpha was calculated using the 'simpledorff' package [28] and returned an alpha of 0,73 for in-expansion pairs and 0,53 for out-expansion

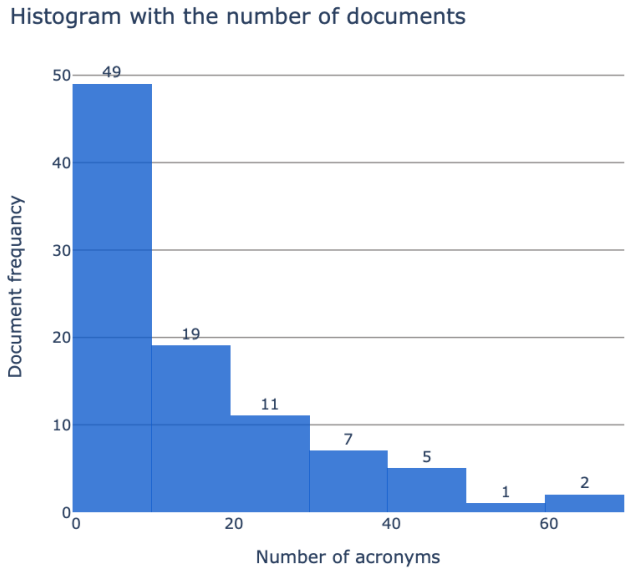

Figure 4: The distribution of the number of documents based on the number of acronyms. Most documents have between zero and ten acronyms, but there are also two with more than 60 acronyms.

pairs. Note that this is calculated over the initial annotations before the two graduate students reviewed the annotation.

It was not possible to have all 301 Wikipedia articles annotated, partly due to the short duration of the research. This is a shortcoming, but even with 94 high-quality annotated articles, the results would be generalizable to a more extensive data set with the same characteristics. The final Dutch data set, which contains the Wikipedia article, acronyms, and expansions, is available on the project repository. The individual .json files cannot be shared since they contain the personal email addresses of the annotators.

3.4 Evaluation metrics

We evaluated different models with a fixed set of metrics during this research. Based on the literature [3, 29, 30, 1], we chose to use precision, recall, and F1 calculated over the acronym expansion pairs. Precision measures how many detected acronym expansion pairs are truly correct. It is calculated by dividing the true acronym expansion pairs by the total number of pairs found. The recall measures how many correct acronym expansion pairs were detected. It divides true acronym expansion pairs by the total number of pairs in the data set. The F1-score represents the harmonic mean of precision and recall and allows one to compare the performance of two or more models. The AcX system calculates these evaluation metrics across multiple categories. For example, the system provides the evaluation metrics

specifically for the in-expansion acronyms or specifically for the out-expansions. As for this study, we are only interested in the results of the end-to-end system, particularly for all acronym-expansion pairs, since this contains the results for both in- and out-expansion.

3.5 Experimental setup

The results of the different models on the English data set [1] provide a starting point for the Dutch experiments. However, one should not expect the results to be the same, partly because the Dutch data set is smaller than its English counterpart. The sections that follow describe the various methods of the AcX system in chronological order. The aim of these experiments is to answer two main questions: are the current methods within the AcX system capable of adequately expanding acronyms in the Dutch language, and to what extent do BERT and SBERT improve upon the current methods?

3.5.1 In-expansion methods. The rule-based methods provided by Schwartz and Hearst [2], and MadDog [3] were the best-performing methods on the English Wikipedia data set within the AcX system. We choose to determine the best in-expander first since the BERT and SBERT models take up to 35 hours to encode train documents. The in-expanders partly influence the out-expansions' performance because they supply the acronym expansion pairs for all training documents. The best in-expander was determined by running the system 42 times and averaging its performance. We removed the intermediate files to ensure that the models have to find the acronyms without prior knowledge. Some initial testing showed that Maddog had slight fluctuations in its performance even though it is a rule-based method. It is noteworthy that the AcX system offers the possibility to alter the priority of the MadDog rules but has not been considered since this research focuses on the out-expansion part of acronym expansion.

3.5.2 Text representers. The AcX system offers various text representers, and the best-performing methods on the different English data sets are Classical Context Vectors (CCV), and Doc2Vec [1]. This research extends the range of text representers with one language-specific BERT model called RobBERT-v2 and two multilingual SBERT called paraphrase-multilingual-mpnet-base-v2 (ml-mpnet-SBERT) and paraphrase-multilingual-MiniLM-L12-v2 (miniLM-SBERT).

CCVs are fixed-length vector representations mainly used for document retrieval and word sense disambiguation [31]. The vectors contain so-called glosses, which can be considered as a bag-of-word where each word has a corresponding word vector. These vectors are then averaged to get a Context Vector [32]. CCVs can recognize the resemblance of words without the need for labeled data and are suitable for finding similar sentences and words with a specific meaning in a set of documents. The main disadvantage of CCVs is their inability to capture language-specific syntax and logic.

Doc2Vec transforms textual input into their vector representation and is widely used for sentence and document similarity tasks in which they often perform well. These vectors contain representations of a group of words taken collectively as a single unit. Doc2Vec takes word order into account and generalizes well on long documents. The AcX system uses the unlabeled training data to create document vectors. The training methods can be influenced by adjusting hyper-parameters such as the number of epochs, learning algorithm [CBOW or Skip-gram], and window size. We have experimented with various combinations during this study.

RobBERT-v2 is a BERT model trained explicitly for the Dutch language and is currently the best-performing model (see 2.4). We are interested in the embedding the model creates on our training data and, therefore, did not see the need to fine-tune the model downstream. We expect the model to perform well since it has been trained solely on Dutch data. The model was initiated with mean pooling to get one fixed sized sentence vector. Also, we used a batch size of 150, embedding normalization set to false and the length of the longest document as input size with zero padding.

MiniLM-SBERT is a multilingual SBERT model trained in over 50 languages [21]. The model has a 384-dimensional dense vector space and a maximum sentence length of 128 tokens. We chose this model because it is popular and the second-best multilingual SBERT-model, according to the Hugging face and sentence-transformer websites. The default maximum sentence length was used to split the input documents into smaller parts. This allows us to use the information of whole documents while keeping the dimensions of the input layer of the model at a reasonable size.

Ml-mpnet-SBERT is a multilingual SBERT model trained in over 100 languages [21]. The model has a 768-dimensional dense vector space and a maximum sentence length of 128 tokens. We chose this model because it is the best performing multilingual SBERT model, according to the sentence-transformer website. As with the smaller MiniLM-SBERT model, the default maximum sentence length was used to split the input documents into smaller parts.

3.5.3 Predictors. Cosine similarity (cossim) and support vector machines (SVM) are the best performing predictors on the English data set. Cossim determines the similarity between different documents by calculating the cosine between the different document vectors. The aim is to find the most similar document that contains the out-expansion of a particular acronym. SVMs, on the other hand, find the best fitting expansion by calculating the probability that a certain document belongs to a particular expansion. The probability is

calculated using a "one-vs-all" method that uses non-binary classification where the classes correspond to the expansions [1]. The performance of these predictors depends on the text representer with which they are paired. Hence, we experimented with different combinations and some of these combinations are shown in Table 3.

4 RESULTS

The results of the experiments are presented in the following tables. Table 2 shows the results of the in-expansion methods, Table 3 the results of Doc2Vec and CCV as text representer and Table 4 shows the results of BERT and SBERT as text representer. One can conclude from these tables that current methods in the AcX system can expand acronyms in Dutch and that the largest SBERT model brings a marginal improvement over the Doc2Vec and CCV.

4.1 In-expansion results

The in-expansion part of the AcX system can be seen as self-contained and was therefore evaluated separately. The performance of the two evaluated in-expanders, Schwartz and Hearst and Maddog, are shown in Table 2 and illustrates that the methods' performance are similar when looking at the F1 score. Maddog performed worse with an average F1 score of 61,87%, with their algorithm having minor deviations of around 0,5%. However, its precision is 8,8% higher than that of Schwartz and Hearts, but it loses this advantage in terms of recall. Alternatively, the Schwartz and Hearts' algorithm had an F1-score of 62,73% and proved to be more consistent with no deviations between the different experiments. As a result, the BERT and SBERT models were all evaluated with Schwartz and Hearts as in-expander since this method performs better on the Dutch Wikipedia data set.

In-expansion type	P	R	F1
Maddog	**89,87**%	47,18%	61,87%
Schwartz & Hearts	81,05%	**51,16**%	**62,73**%

Table 2: The comparison between the two in-expander types shows that Schwartz and Hearts' rules slightly outperform Maddog in terms of F1 by 0.86% on the Dutch data set.

4.2 Results of the Doc2Vec and CCV

The results of the Doc2Vec and CCV models are shown in Table 3. The various experiments done with these methods have shown that the combination of Schwartz and Hearst, SVM, and Doc2Vec leads to the best F1 score of 47,12%. The definitive parameters that led to this performance were: epochs =100, learning algorithm = CBOW, vector size =100, Window size = 5. These parameters also led to the best results on the English data set. We experimented with a range of epochs from 80 to 130 with strides of 10, Skip-gram as a learning algorithm, and large window sizes, but none of these methods yielded better results. The overall performance with Doc2Vec as text representer is better than their CCV counterparts, with Doc2Vec performing better in all comparable situations. CCV as a text representer yielded, on average, a 2,53% lower F1 score. Furthermore, the best-performing model is 1,14% better than the Maddog variant with the same out-expansion parameters.

4.3 Results of BERT and SBERT

The results of the different BERT and SBERT models are presented in Table 4. All of the experiments with BERT and SBERT have been performed with Schwartz and Hearst as in-expansion method. Table 4 shows that ml-mpnet-SBERT had the highest performance of all models with an F1 score of 47,95%, a precision of 54,45%, and a recall of 42,83%. The F1 score of this model is 0,83% better than the best-performing initial model (SVM Doc2Vec) and the smaller MiniLM-SBERT model. Surprisingly, the MiniLM-SBERT model has the same precision and recall as the SVM Doc2Vec model. Furthermore, SVM as a predictor provides the best results for all transformer models, whereas the language-specific RobBERT-v2 model has the worst performance in all three-evaluation metrics.

5 DISCUSSION

The results show that the rule-based methods of Schwartz and Hearst provide the highest in-expansion F1 score of 62,73%. This score is only marginally higher than that of Maddog, which has an F1 score of 64,87%. When comparing the precision and recall of both methods, one can see that Maddog has a higher precision with an average of 89,87% compared to 81,05% for Schwartz and Hearst. The opposite is true for the recall, where Maddog reached an average of 47,18%, compared to 51,16% for Schwartz and Hearst, and this explains why Schwartz and Hearst's F1 score is slightly higher. On the English data set, Maddog is the better performing in-expander. It is unclear if this is caused by the linguistic characteristics of the Dutch language or by the larger amount of evaluation data. Regardless, Maddog's additional set of rules causes the method to find fewer acronym expansion pairs, leading to a lower recall. The results of the end-to-end experiments show that ml-mpnet-SBERT as text representer and SVM as predictor provide the best acronym-expansion results in the Dutch language. This method had an F1 score of 47.95%. The larger network behind ml-mpnet-SBERT enables the model to generate document vectors that, combined with SVM as a predictor, lead to the best results. However, the F1 is only marginally better than miniLM-SBERT and the model that uses SVM as predictor and Doc2Vec as text representer.

In-exp	Out-exp predictor	Representer	P	R	F1
Schwartz & Hearst	Cossim	Doc2Vec	52,07%	40,95%	45,85%
		CCV	49,20%	38,69%	43,31%
	SVM	Doc2Vec	53,51%	**42,09%**	**47.12%**
		CCV	50,32%	39,57%	44,30%
MadDog	Cossim	Doc2Vec	60,74%	35,18%	44,55%
		CCV	57,70%	33,42%	42,32%
	SVM	Doc2Vec	**62,69%**	36,31%	45,98%
		CCV	59,22%	34,30%	43,44%

Table 3: The results show that a combination of Schwartz and Hearst, SVM, and Doc2Vec leads to the highest F1 score of 47,12%. Furthermore, Doc2Vec outperforms CCV in all comparable combinations.

Representer	Predictor	P	R	F1
MiniLM-SBERT	Cossim	50,80%	39,95%	44,73%
	SVM	53,51%	42,09%	47,12%
ml-mpnet-SBERT	Cossim	51,91%	40,61%	45,57%
	SVM	**54,45%**	**42,83%**	**47,95%**
RobBERT-v2	Cossim	49,42%	38,95%	43,60%
	SVM	51,87%	40,82%	45,69%

Table 4: The best-performing text representer is ml-mpnet-SBERT, with an F1 score of 47,95%. This is only marginally higher than MiniLM-SBERT and the Doc2Vec, each having an F1 of 47,12%. RobBERT-v2 did not perform as expected, only achieving an F1 score of 45,69%.

5.1 Limitations

Overall, the results show that the current methods offered by the AcX system make it possible to properly find and expand acronyms in Dutch Wikipedia articles. Table 5 shows the results of the Doc2Vec and SVM model on both the English [1] and Dutch Wikipedia data set, as well as the performance of ml-mpnet-SBERT. One can see that there is a performance gap between the two languages.

There are two likely reasons for this performance difference. The first reason relates to the size of the Wikipedia data used for training. The English data set has 6.512.725 articles, while the Dutch data set contains 2.092.636 articles. This difference in the amount of training data influences the out-expansion performance. The second reason for the performance difference might have to do with a large number of English acronyms and expansions (576 pairs) in the Dutch evaluation data set and presumably in the overall language. Some of these English acronyms cannot be matched with a Dutch article containing its full expansion, which negatively affects the performance.

Language	Model	P	R	F1
English	Doc2Vec + SVM	**59,38%**	**48,46%**	**53,37%**
Dutch	Doc2Vec + SVM	53,51%	42,09%	47,12%
	SBERT + SVM	54,45%	42,83%	47,95%

Table 5: The Doc2Vec SVM model within the AcX system perform better on English with the performance gap being 6.25%. The reason for this probably lies in the amount of training data and the high number of English acronyms in the Dutch language.

6 CONCLUSION

There is a considerable amount of research on acronym expansion in English, but little is known about the applicability and performance of these methods in the Dutch language.

During this research, we examined to what extent the current acronym expansion methods are applicable in the Dutch language. The results show that the current methods are indeed applicable to the Dutch language, but there is still a performance void of 5,42% when comparing the best Dutch and English models. Furthermore, BERT-based models have proven to perform well in other fields of NLP [8, 9, 7]. We, therefore, experimented with BERT and SBERT models to explore their possible performance gains as text representers in an end-to-end acronym expander. The results show that the best performing text representer on the Dutch data set was ml-mpnet-SBERT, which yielded an F1 score that was 0,83% higher than the second-best model that uses Doc2Vec. Overall, this research has partially closed the knowledge gap in Dutch acronym expander systems. For instance, there is now a high-quality data set for evaluating acronym expansion systems that can be used for in- and out-expansion purposes. In addition, more is now known about the performance of in-expansion, out-expansion, and end-to-end acronym expansion systems. The annotated evaluation set is smaller than its English counterpart, with 94 annotated Wikipedia articles, but it contains a diverse range of articles, acronyms, and expansions.

Future studies can close the knowledge gap by focusing on two main topics. First is the influence of adding English training data in the training phase since the Dutch language uses a variety of English acronyms. Second, we could not evaluate the performance of an SBERT model, specifically fine-tuned for acronym expansion. This is a shortcoming in our research but provides an opportunity for others in the near future.

REFERENCES

[1] João L M Pereira et al. "AcX: system, techniques, and experiments for Acronym eXpansion [Extended Version]". In: *PVLDB 2022 [In press]* (2022).

[2] Ariel S Schwartz and Marti A Hearst. *A simple algorithm for identifying abbreviation definitions in biomedical text*. Tech. rep. 2002. URL: www.worldscientific.com.

[3] Amir Pouran Ben Veyseh et al. "MadDog: A Web-based System for Acronym Identification and Disambiguation". In: *CoRR* (Jan. 2021). URL: http://arxiv.org/abs/2101.09893.

[4] Aadarsh Singh and Priyanshu Kumar. "SciDr at SDU-2020 : IDEAS-Identifying and Disambiguating Everyday Acronyms for Scientific Domain". In: *CoRR* (2021), pp. 451–462.

[5] Michael R Glass, Faisal Mahbub Chowdhury, and Alfio M Gliozzo. *Language Independent Acquisition of Abbreviations*. Tech. rep.

[6] Jean Charbonnier and Christian Wartena. "Using Word Embeddings for Unsupervised Acronym Disambiguation". In: (Aug. 2018), pp. 2610–2619. URL: https://aclanthology.org/C18-1221.

[7] Nicholas Egan and John Bohannon. "Primer AI's Systems for Acronym Identification and Disambiguation". In: *CoRR* abs/2012.08013 (2020). arXiv: 2012.08013. URL: https://arxiv.org/abs/2012.08013.

[8] Matej Ulčar and Marko Robnik-Šikonja. "FinEst BERT and CroSloEngual BERT". In: *Text, Speech, and Dialogue*. Ed. by Petr Sojka et al. Cham: Springer International Publishing, 2020, pp. 104–111.

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

[10] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. DOI: 10.48550/ARXIV.1301.3781. URL: https://arxiv.org/abs/1301.3781.

[11] Iz Beltagy, Kyle Lo, and Arman Cohan. "SciBERT: A Pretrained Language Model for Scientific Text". In: *CoRR* (2019). DOI: 10.48550/ARXIV.1903.10676. URL: https://arxiv.org/abs/1903.10676.

[12] Cheng Ju Kuo et al. "BIOADI: A machine learning approach to identifying abbreviations and definitions in biological literature". In: *BMC Bioinformatics* 10.SUPPL. 15 (Dec. 2009). ISSN: 14712105. DOI: 10.1186/1471-2105-10-S15-S7.

[13] Amir Pouran Ben Veyseh et al. "What Does This Acronym Mean? Introducing a New Dataset for Acronym Identification and Disambiguation". In: *CoRR* abs/2010.14678 (2020), pp. 1 –17. arXiv: 2010.14678. URL: https://arxiv.org/abs/2010.14678.

[14] Aditya Thakker, Suhail Barot, and Sudhir Bagul. "Acronym Disambiguation: A Domain Independent Approach". In: (Nov. 2017). URL: http://arxiv.org/abs/1711.09271.

[15] Julia Van Oosten. *Disambiguating Dutch Medical Abbreviations using Language Models*. URL: https://scripties.uba.uva.nl/search?id=722742.

[16] Pieter Delobelle, Thomas Winters, and Bettina Berendt. "RobBERT: a Dutch RoBERTa-based Language Model". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, Nov. 2020, pp. 3255–3265. DOI: 10.18653/v1/2020.findings-emnlp.292. URL: https://www.aclweb.org/anthology/2020.findings-emnlp.292.

[17] Wietse de Vries et al. *BERTje: A Dutch BERT Model*. arXiv:1912.09582. Dec. 2019. URL: http://arxiv.org/abs/1912.09582.

[18] Julien Abadji et al. "Towards a Cleaner Document-Oriented Multilingual Crawled Corpus". In: *arXiv e-prints*, arXiv:2201.06642 (Jan. 2022), arXiv:2201.06642. arXiv: 2201.06642 [cs.CL].

[19] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *CoRR* abs/1907.11692 (2019). arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692.

[20] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: http://arxiv.org/abs/1908.10084.

[21] Nils Reimers and Iryna Gurevych. "Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation". In: (Nov. 2020). URL: https://arxiv.org/abs/2004.09813.

[22] Nils Reimers. *SentenceTransformers*. 2022. URL: https://www.sbert.net/docs/pretrained_models.html#multi-lingual-models.

[23] Amir Pouran Ben Veyseh et al. "What Does This Acronym Mean? Introducing a New Dataset for Acronym Identification and Disambiguation". In: *CoRR* abs/2010.14678 (2020), pp. 1 –17. arXiv: 2010.14678. URL: https://arxiv.org/abs/2010.14678.

[24] Cheng Ju Kuo et al. "BIOADI: A machine learning approach to identifying abbreviations and definitions in biological literature". In: *BMC Bioinformatics* 10.SUPPL. 15 (Dec. 2009). ISSN: 14712105. DOI: 10.1186/1471-2105-10-S15-S7.

[25] Antonio J. Jimeno-Yepes, Bridget T. McInnes, and Alan R. Aronson. "Exploiting MeSH indexing in MEDLINE to generate a data set for word sense disambiguation". In: *BMC Bioinformatics* 12 (June 2011). ISSN: 14712105. DOI: 10.1186/1471-2105-12-223.

[26] David Malvern et al. *Lexical Diversity and Language Development*. Palgrave Macmillan, a division of Macmillan Publishers Limited 2004: Palgrave Macmillan London, 2004. DOI: https://doi.org/10.1057/9780230511804.

[27] Gareth James et al. *An introduction to statistical learning: With applications in R*. Springer, 2021.

[28] Tal Perry. "Simpledorff - Krippendorff's Alpha". In: LIGHTTAG, 2020. URL: https://www.lighttag.io/blog/krippendorffs-alpha/.

[29] Kayla Jacobs, Alon Itai, and Shuly Wintner. "Acronyms: identification, expansion and disambiguation". In: *Annals of Mathematics and Artificial Intelligence* 88.5-6 (June 2020), pp. 517–532. ISSN: 15737470. DOI: 10.1007/s10472-018-9608-8.

[30] Sungrim Moon, Bridget McInnes, and Genevieve B. Melton. "Challenges and practical approaches with word sense disambiguation of acronyms and abbreviations in the clinical domain". In: *Healthcare Informatics Research* 21.1 (Jan. 2015), pp. 35–42. ISSN: 2093369X. DOI: 10.4258/hir.2015.21.1.35.

[31] Stephen I. Gallant. "Context Vectors: A Step Toward a "Grand Unified Representation"". In: *Hybrid Neural Systems*. Ed. by Stefan Wermter and Ron Sun. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 204–210.

[32] Siddharth Patwardhan and Ted Pedersen. "Using WordNet-based Context Vectors to Estimate the Semantic Relatedness of Concepts". In: 2006.