Linguistic Enrichment of Historical Dutch using Deep Learning

Creten, Silke; Dekker, Peter; Vandeghinste, Vincent

Published in:
Computational Linguistics in the Netherlands Journal

Publication date:
2020

Computational Linguistics in the Netherlands Journal 10 (2020) 57–72 Submitted 10/2020; Published 12/2020

Linguistic Enrichment of Historical Dutch using Deep Learning

Silke Creten[1] SILKE.CRETEN@KULEUVEN.COM
Peter Dekker[2,3] PETER.DEKKER@AI.VUB.AC.BE
Vincent Vandeghinste[3] VINCENT.VANDEGHINSTE@IVDNT.ORG

[1] *KU Leuven, Leuven, Belgium*

[2] *Vrije Universiteit Brussel, Brussels, Belgium*

[3] *Instituut voor de Nederlandse Taal, Leiden, The Netherlands*

Abstract

This article discusses the automatic linguistic enrichment of historical Dutch corpora through the use of part-of-speech tagging and lemmatization. Such a type of enrichment facilitates linguistic research where manual annotation is unfeasible.

We built a neural network-based model using the PIE framework and performed an in-depth error analysis, in order to identify the strengths and weaknesses of each approach with respect to labeling historical data.

In order to do so, we experimented with two data sets: the *Corpus Gysseling* (13th century texts) and the *Corpus van Reenen/Mulder* (14th century texts). We used two different statistical approaches (MBT and HunPos) as baselines for our neural approach. MBT is a memory-based tagger frequently used for modern Dutch, while HunPos is an open source trigram tagger.

We present thoroughly analyzed results. In general, the neural model scores better than the two baselines, even with limited training data. Based on the error analysis, we propose several strategies for future research in order to improve the labeling of historical Dutch.

1. Introduction

This article aims to facilitate the linguistic enrichment of historical Dutch corpora through the use of automatic part-of-speech tagging and lemmatization. Part-of-speech tagging is the assignment of morpho-syntactic tags to tokens. Lemmatization, on the other hand, is the task of converting a token to its dictionary headword, which allows us to abstract away from orthographic and inflectional variation. The goal of these two classification tasks is to facilitate linguistic research by automatically annotating a large number of historical texts, since manual annotation is expensive and prone to human errors, e.g., deviations on the normalizations agreed upon. The annotated data can then be beneficial for corpus linguistics (Daelemans et al. 1996). This article contributes to research on sequence tagging of historical Dutch by creating a neural network-based model, and by analyzing the specific problems that model encounters during tagging.

As in Kestemont et al. (2010), we regard lemmatization and PoS tagging as related tasks which both involve assigning labels (PoS tags or lemmas) to words. Consequently, we used the same sequence tagger for lemmatizing as for PoS tagging. We used two baselines in default settings, namely the statistical taggers MBT (Daelemans et al. 1996) and HunPos (Megyesi 2009). Additionally, we built a neural network based model using the PIE framework created by Manjavacas et al. (2019). These taggers were tested on two fully annotated corpora from the thirteenth and fourteenth century, namely *Corpus Gysseling* (*CG*) and *Corpus van Reenen/Mulder* (*CRM*) respectively.

In recent years, several techniques have been proposed to optimize natural language processing. Approaches using deep learning have shown promising results, e.g., BERT for pre-training (Devlin et al. 2019), the Transformer for machine translation (Vaswani et al. 2017) and the Transformer-XL

(Dai et al. 2019). These recent advances motivated us to apply and evaluate a deep learning based approach for our particular task.

Sequence tagging of historical data, and more specifically historical Dutch, is more challenging than the tagging of standardized modern languages for several reasons. First of all, less data are available for historical languages. Secondly, in the thirteenth and fourteenth century, no standard Middle Dutch existed, and texts were manually copied. Scribes were not required to be consistent and there was a lot of room for individual liberties (Kestemont and Van Dalen-Oskam 2009). Consequently, it is possible that several orthographic forms of one word occur in the same text (Kerckvoorde 1993). Moreover, in this period an amalgam of regional dialects existed, without any sort of supra-regional variant (Kestemont et al. 2016). Most of the time, the scribes wrote phonetically, i.e., they wrote a word based on how they would pronounce it (Kerckvoorde 1993). Therefore, there was a strong regional variation in the spelling, even for highly frequent words. Additionally, there were preferred, more frequently used orthographic forms depending on when and, again, where the texts were written. This lack of an orthographic standard makes the task of PoS tagging and lemmatization more difficult (Kestemont et al. 2010).

However, not only the orthographic variation in Middle Dutch is much larger, compared to contemporary standardized Dutch, but also the morphological variation. An example of such variation is the existence of a case system, similar to that of Latin, in which the declension varied in accordance with the part of speech (Kerckvoorde 1993).

Apart from the morphological and orthographic variation, frequent lexical ambiguities pose a third difficulty for sequence tagging (Manjavacas et al. 2019). Lexical ambiguity can arise, for instance, when two tokens have the same form, but when they belong to different parts of speech. For example, in English, "spelling" could be a noun or the gerund of a verb ("to spell"). In Middle Dutch "dier" could be a noun ("animal") or a determiner ("the" or "this").

The remainder of this article is organized as follows: in related, we give a brief overview of similar research, in method we present the corpora we worked on and outline the methodology used in this study, in results we present results of the experiments with special attention to the errors our model encountered. Finally, conclusions concludes the paper and discusses future work.

2. Related research

Daelemans et al. (1996) performed part-of-speech tagging using their memory-based tagger MBT on the *Eindhoven corpus*, a corpus of modern Dutch. However, their research differs from ours in that they worked with contemporary data and had a smaller, less complicated tag set, as they restricted it to 13 PoS tags. Kestemont et al. (2010) also used a memory-based approach in order to lemmatize the literary texts in historical Dutch in *Corpus Gysseling*. They improved memory-based tagging by applying Levenshtein distance. Longrée and Poudat (2010) compared the performance of three sequence taggers on classical Latin texts, namely MBT, Trigrams'n'Tags (TnT) (Brants 2000), and TreeTagger (Schmid 1994, 1995). In their experiments TnT performed slightly better than MBT. HunPos is an open-source reimplementation of TnT. As Middle Dutch and Latin are both morphologically rich languages, a similar performance is expected in our experiments, i.e., HunPos is expected to outperform MBT on our data sets. Also for Middle Dutch, van Halteren and Rem (2013) created a tagger-lemmatizer with a focus on orthographic variation. They used 10-fold cross-validation on *Corpus van Reenen/Mulder* (*CRM*), and reached an overall tagging-lemmatization accuracy of about 95%.

Nowadays, sequence tagging tasks are more often tackled by means of a neural approach. Kestemont et al. (2016) looked into lemmatization using character-level convolutions, thus replacing the more traditional one-hot encoding. On *CRM*, the same corpus as van Halteren and Rem (2013) used, they reported a slightly lower lemmatization accuracy of 93.95%. This was found to be advantageous when tagging unknown tokens. Nevertheless, the authors proved that one-hot encoding remains a good approach for known tokens. Schmid (2019) introduced a morphological tagger with the spe-

cific scope of annotating historical texts. He used a character-based bidirectional LSTM (BiLSTM), which allows the model to learn bidirectional long-term dependencies. He tested his PoS tagger on *Corpus Gysseling*, among other historical corpora, tagging the words in their historical spelling, without a normalization step. He achieved 91.01% accuracy on his test set for Middle Dutch. He also used a standard encoder-decoder network for lemmatization, but that task was not tested on *CG*.

Kestemont and De Gussem (2017) researched joint learning, and created a multi-class learning environment, where lemmatization and PoS tagging can be executed at the same time. They tested their model on Latin corpora and noticed that the performance of PoS tagging tasks increased in a multi-task environment, but that the overall lemmatization results were lower. Since further research is necessary in order to explain this phenomenon, we tried multi-task sequence labeling on our corpora to verify whether we would find similar tendencies in Middle Dutch. Manjavacas et al. (2019) used the PIE framework for lemmatization of non-standard languages with joint learning. We based our experiments with the neural model on their work. However, instead of focusing on multiple data sets, we limited our research to Middle Dutch and to how we could improve the model specifically for our two data sets. We contribute to their research by creating a new neural network based model with the PIE framework, and by doing a fine-grained error analysis. The morphological and orthographic variation, as well as the lexical ambiguity previously mentioned, had an influence on our decision to operate directly on the historical spelling, instead of on modern word forms, a choice also made by Schmid (2019) and Dipper (2010, 2011), among others.

3. Methodology

3.1 Corpora

3.1.1 *Corpus Gysseling (CG)* AND *Corpus van Reenen/Mulder (CRM)*

We tested our sequence taggers on two corpora containing texts in Middle Dutch, from which we extracted two separate data sets. First, we used the *Corpus Gysseling* (*CG*), which consists of 1.5 million tokens in various heterogeneous documents from the thirteenth century. It was published between 1977 and 1987 on the initiative of Maurits Gysseling and has been annotated and digitized by the Institute for Dutch Lexicology in Leiden, now called the Dutch Language Institute (INT).[1] It was used as source material for a dictionary of Early Middle Dutch (*VMNW*) (Piotrowski 2012). The corpus consists of a collection of documents that vary in length. Consequently, each text has a different number of tokens. There are two main genres, namely literary documents and official documents. We expect the first category to be more of a problem for sequence tagging, since there is more room for individual liberties, contrary to the official documents in which we can encounter some recurrent formulae. Apart from the different genres, there is also temporal variability (with a maximum temporal distance of 100 years between two texts), and regional variability, as the corpus includes texts from thirteen main regions.

The second corpus that we used in our experiments is the *Corpus van Reenen/Mulder* (*CRM*)[2]. This corpus was compiled by Maaike Mulder and Piet van Reenen in the 1980's and it consists of 2,700 charters from the fourteenth century. Contrary to the *CG*, this corpus only consists of formal documents that treat legal affairs, consequently it contains some recurring field-specific expressions. Mostly, these formulaic expressions occur at the beginning or the end of the charter, while the middle is less fixed (van Halteren and Rem 2013). *CRM* was annotated manually, and cleaned semi-automatically with pattern matching techniques. The Adelheid 1.0 tagger-lemmatizer was

1. The corpus is available for research on the following website: `https://ivdnt.org/corpora-lexica/corpus-gysseling/`. A user-friendly interface has also been created, which allows to look for specific word forms, lemmas, and parts of speech in the corpus: `http://gysseling.corpus.taalbanknederlands.inl.nl/gysseling/page/search`
2. *CRM* is available on the following website: `http://www.diachronie.nl/corpora/crm14`

then trained on this corpus (Piotrowski 2012, Rem and van Halteren 2007). As van Halteren and Rem indicate, it contains charters from 345 places of origins. Furthermore, since *CRM* contains documents from the beginning to the end of the century, there is temporal variation.

The data sets contain three main elements, namely token, PoS tag, and lemma. The PoS tag consists of a part of speech and supplementary morpho-syntactic features. The singular common noun *vrouwen* has for instance *NOU(type=common,number=sg,inflection=n)* as PoS tag, and *VROUW* as lemma. The following is an example of a phrase annotated with lemma and PoS tag information, extracted from *CG*:

ene/PD(type=art,subtype=indef,inflection=e)/EEN//**schone**/ADJ(number=sg,inflection=e)
/SCHOON//**historie**/NOU(type=common,number=sg,inflection=0)/HISTORIE

In both data sets, there are contracted forms and compounds. With contracted forms, we indicate tokens with two or more tags. These tokens are lemmatized and tagged per recognizable token. An example of a contracted form that frequently occurs in both data sets is the token *vanden*, with the PoS categories *ADP+PD* and the lemmas *VAN+DE*. We will designate the standard forms as single PoS/lemma tags, and if they are concatenated we will talk about double, triple, quadruple, or quintuple PoS/lemma tags. The Middle Dutch compounds are treated in the same manner as the contracted forms, e.g., *hůuelants* has two PoS categories (*NOU+NOU*) and two lemmas (*HOEVE+LAND*).

3.1.2 Pre-processing

As illustrated in method: tokens, the data set extracted from *CG* consists of 1,035,780 non-unique tokens after pre-processing. All tokens have been lower-cased and the punctuation has been removed, since, in historical languages, punctuation was not consistently used because of (orthographic) variation (Piotrowski 2012). The end-of-sentence information was not always present in the annotated corpus and the punctuation was not sufficient as an end-of-sentence indicator without a lot of (expensive) manual control. Instead we have opted to include the end-of-paragraph information in our *CG* data set. The corpus contained Latin tokens that had not been annotated with a lemma and/or had been labeled "other" as part of speech. As these Latin tokens were deemed irrelevant to our analysis, we removed them during pre-processing. Our training and test sets were extracted from the data set with a 80/20 split. For the neural model, a validation set was extracted from the training data with a 90/10 split. In the training and test set, full documents were included on one particular topic, as to avoid the risk of an information leak. The genre, literary or official, was also included in the data set. 88.00% of the non-unique tokens in our pre-processed data set are from official texts. In our test set, 9.60% of the tokens come from literary texts. We did not extract regional or temporal information.

	CG			***CRM***		
	full	train	test	full	train	test
Unique tokens	58,045	50,084	20,044	55,752	48,864	20,976
Unique PoS	1,715	1,563	866	815	750	497
Unique lemmas	19,173	17,402	7,761	16,512	14,866	7,381
Total tokens	1,035,780	851,118	184,662	1,078,223	864,860	213,363

Table 1: Number of unique tokens (= types), tags, and lemmas in the full data set extracted from *Corpus Gysseling* and *Corpus van Reenen/Mulder*. The distribution between training and test set is indicated. The total amount of non-unique tokens per document are mentioned in the final row.

Test set	unknown tokens	unknown tags	unknown lemmas
CG	5.78%	0.12%	1.74%
CRM	4.07%	0.04%	1.11%

Table 2: Number of non-unique unknown token, tags, and lemmas in the two test sets. "Unknown" means that the tagger did not encounter these forms during training.

	CG	***CRM***
Single tags	94.52%	97.38%
Double tags	5.29%	2.60%
Triple tags	0.19%	0.01%
Quadruple tags	0.01%	0.00%
Quintuple tags	0.00%	N/A

Table 3: Non-unique contracted (lemma and PoS) tags in the two data sets. Percentage = n tags in set / all tags in set. In *CG*, there were fifteen quintuple tags (none of which were included in the test set). In *CRM*, there was only one quadruple tag, and there were no quintuple tags.

CRM has a similar number of non-unique tokens (cf. method: tokens). Again, we removed the punctuation and we lower-cased all tokens. Contrary to *CG*, the end-of-sentence information was consistently indicated in the corpus and thus included in our data set. In this case, there were no Latin tokens. As with the *CG* data set, we have opted for a 80/20 split to create the training and test set. Contrary to the *CG* data set, we did not divide the corpus based on genre, since it only contains official documents.

Both data sets have a similar number of tokens. The amount of non-unique unknown tokens, PoS tags, and lemmas are given in method: unknown. Unknown tokens are tokens which occur in the test set, but not in the training set. We found more unknown non-unique tokens in *CG* than in *CRM*, which could influence the final results of our taggers.

From method: contracted, we can infer that most of the tags are not contracted. The *CRM* data set, though still large, contains far fewer PoS tags and lemma tags (cf. method: tokens). In addition, there are fewer contracted forms. Also, *CRM* has a more rigid structure, with only charters, contrary to *CG*, which has more variation in genre. Because of this, we expect to perform better on this corpus.

PoS:	***CG***			***CRM***		
subcategories	full	train	test	full	train	test
NOU	29	29	24	22	22	20
PD	64	64	55	56	56	55
VRB	58	57	46	59	59	47
ADJ	18	18	13	14	14	12
ADV	55	50	44	37	35	29
ADP	10	10	10	6	6	5
NUM	23	23	20	19	19	18
CON	25	24	19	15	14	9

Table 4: Distribution of non-contracted tags for PoS tags in both data sets. The number of subcategories in the training and test set are also indicated.

The different parts of speech the tagger can encounter in *CG* and *CRM* are shown in method: PoS cat. We notice that the two data sets have a very similar distribution with respect to the PoS categories, even though the *CRM* data set generally contains fewer subcategories. We define subcategories as tags with the same part of speech, but with different morpho-syntactic information, e.g., type (common/proper) or number (plural/singular).

3.2 Taggers

3.2.1 Baselines: MBT and HunPos

MBT (*Memory-Based Tagger*) is a statistical, memory-based tagger, introduced by Daelemans et al. (1996).[3] We decided to use this model as a baseline in our evaluation because it has been a popular sequence tagger for modern Dutch (Kestemont and De Gussem 2017). In a memory-based approach, training examples are being stored, each with its preceding and following context and its label. When the trained model encounters a new token during tagging, it searches for the most similar stored example, and assigns the same label to the test example. It is therefore a form of "inductive learning from examples" (Daelemans et al. 1996). MBT automatically extracts a lexicon from the training set. During tagging, every token will be looked up in this lexicon, and based on its presence there, it will be treated as known or unknown. Unknown tokens will be classified based on their context and their form. The morphological information taken into account concerns mostly the suffixes (Daelemans et al. 1996). MBT, and memory-based learning algorithms in general, have the advantage that they can handle sparse data, with an implicit similarity-based smoothing scheme, and that they implement automatic feature-weighting. They are lazy learners, since memory storage is done without abstraction, therefore there is no rule induction in this part (Zavrel and Daelemans 1999). During tagging, a similarity-based classification will take place. There are several options for parameter tuning, for which we refer to the reference guide accompanying the software by Daelemans et al. (2010). In this article, however, we opted for the default parameters: The tagger generator creates a list with the 100 most frequent words in the corpus. For the known words, the memory-based learning algorithm is IGTREE, consequently features are sorted according to information gain and classification happens through decision-tree traversal. For unknown words, IB1 with the overlap metric with gain ratio feature weighting is used, and the number of nearest neighbors is set to 1.

The other baseline used on our two data sets is HunPos[4]. This is an open source trigram tagger, created in 2007. It is a reimplementation of the closed source TnT tagger, and they are both taggers based on Markov models (Brants 2000). The earliest statistical approaches already used Hidden Markov Models (HMM), and had the aim to build probabilistic models of tag transition sequences in sentences (Zavrel and Daelemans 1999). HunPos takes contextualized lexical probabilities into account, and more precisely the information of current and previous tag. It makes use of the end-of-sentence information. It has the major advantage that the training/tagging cycle is a lot faster than in most statistical models (Halácsy et al. 2007). Morphological information is also integrated, which is supposed to increase the precision during the tagging of unknown and unseen tokens. When the tagger encounters an unseen token, it generates all possible labels and then assigns weights to it using a suffix guessing algorithm (Halácsy et al. 2007). It is a command line tool, with only a few possibilities of parameter tuning, which are described in Megyesi (2009) and Brants (2000). As for MBT, we have opted for the default parameters. For training, this means that the probability of a tag, as well as the probability of a token, is estimated based on the previous 2 tags, and that the rareness parameter is set at 10. Also, the length of the longest suffix to be considered when the algorithm estimates an unseen word's tag distribution is 10. Finally, during tagging, HunPos keeps 10 tags with the highest probabilities.

3. Code available online: `https://languagemachines.github.io/mbt/`.
4. HunPos is available on this website: `https://code.google.com/archive/p/hunpos/downloads`

3.2.2 Neural approach

We built a neural network-based model for sequence labeling, with an encoder-decoder architecture, using the PIE framework (Manjavacas et al. 2019). Besides facilitating this task, the PIE framework also allows for the joint learning of sequence labeling tasks of variation-rich languages (in this case Middle Dutch), or in other words, it is possible to combine lemmatization and part-of-speech tagging.

The underlying structure of PIE is hierarchical, features are extracted from the character-level up to the sentence-level. For instance, for each input token, a sentence-level feature vector is extracted and then used to predict certain target tasks (Manjavacas et al. 2019). In our case, we tested the performance of the model on PoS tagging and lemmatization, separately as well as jointly.

The PIE framework is highly configurable and user-friendly, as it allows for a lot of parameter tweaking. By means of this framework, we carried out the parameter tuning on the development set and created a neural model that allowed us to achieve good results on the two test sets. Keeping the differences between the tag sets in mind, we decided to build different models for each data set to achieve optimal performance based on the development set.

For PoS tagging we used PIE to build a one-layered model, with a linear decoder. It is a Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) as a cell type. LSTM is a "recurrent network architecture in conjunction with an appropriate gradient-based learning algorithm", which is designed to overcome the error back-flow problems of RNN (Hochreiter and Schmidhuber 1997). The hidden layer size was 300. The main difference between the present model and that of Manjavacas et al. (2019) is that we opted for a word-level approach, and not a character-level approach, as this approach yielded better results during preliminary experiments on our development set. We also enlarged the default maximum sentence length to fifty words. The batch size of our model was 32. Manjavacas et al. (2019) proved that for lemmatization, it is possible to get better results by using joint Language Model loss, especially on ambiguous data sets. However, on our *CG* data set, with PoS tagging as the target task, we found better results without LM-loss. It was included for our *CRM* data set, with 0.2 as weight. Pretraining embeddings using word2vec did not improve the *CG* model either, but it did improve the results on the validation set of *CRM*. Our drop-out rate was 0.5 and the value of our learning rate was 0.75. As an optimizer we used Adam (Kingma and Ba 2015).

For lemmatization, we built an identical model for the two data sets. We made, similarly to our approach for PoS tagging, a one-layered RNN with LSTMs, but with a hidden layer size of 200. Again, we approached this task on a word-level, as this yielded better results during parameter tuning on the development set. In this case, we did include the autoregressive loss (with 0.3 as weight and factor 0.5). The batch size was 64. For the lemmatizer, we did get better results by pretraining the embeddings with word2vec. Our learning rate is the same as for our PoS model, but the drop-out rate was 0.4. As in our PoS model, we used Adam as an optimizer.

For our combined model with the *CRM* data set, we obtained better results by making PoS tagging our target task. With the *CG* data set, lemmatizing was our target task. We used similar settings to those of the PoS model of *CRM*, LM-loss with 0.2 as weight included. However, pretraining did not improve the performance of the model.

4. Results

4.1 General Findings

For our part-of-speech tagging task on the data set extracted from *Corpus Gysseling*, we achieved a total accuracy of 87.23% using MBT with the default settings. HunPos, also with the default settings, reached 89.08% accuracy. With the neural approach, using the PIE framework, we found the highest accuracy, namely 91.83%. In results1: CG PoS the performance on contracted forms is illustrated. We notice that single tags were, as expected, overall better predicted by the taggers. Secondly, in general, the neural network approach obtained better results. This is especially true for the double tags, as there was a clear increase in performance compared to the other two taggers (HunPos:

CG: contracted	MBT	HunPos	PIE-FW
Single PoS tags	87.83%	89.78%	92.39%
Double PoS tags	77.81%	78.09%	83.27%
Triple PoS tags	39.33%	39.61%	41.85%
Quadruple Pos tags	0.00%	0.00%	0.00%
Overall	**87.23%**	**89.08%**	**91.83%**

Table 5: Accuracy of the three models on *Corpus Gysseling* and comparison of the performance on the contracted PoS forms.

CRM: contracted	MBT	HunPos	PIE-FW
Single PoS tags	92.00%	93.14%	95.03%
Double PoS tags	85.60%	85.49%	87.72%
Triple PoS tags	23.81%	23.81%	19.05%
Quadruple PoS tags	0.00%	0.00%	0.00%
Overall	**91.83%**	**92.93%**	**94.82%**

Table 6: Accuracy of the three models on *Corpus van Reenen/Mulder* and comparison of the performance on the contracted PoS forms.

-5.18%, MBT: -5.46%). All three taggers failed to correctly predict any of the 18 quadruple tags in the test set. This is not surprising, as there were only 76 of these forms included in the training set.

The results for the PoS tagging task on the *CRM* corpus are listed in results1: CRM PoS. With MBT, we reached a total accuracy of 91.83%. With HunPos, with the default settings, 92.93%. With the neural approach, using the PIE framework, we achieved an accuracy of 94.82%, and thus the best performance. If we look at the contracted forms, we notice that the neural model reached a substantially lower accuracy (-4.76%) than MBT or HunPos on the rare triple PoS tags.

MBT, HunPos, and the neural model performed better on the second data set, with entries from the fourteenth century. We achieved the greatest increase in performance with MBT (+4.60%), and the lowest increase with our neural model (+2.99%). However, for our neural PoS tagger we used different parameters for each of the two corpora, which made them difficult to compare.

There are several possible causes for this increase in performance. Since both data sets have approximately the same number of tokens, we do not consider this aspect to be an influential factor. A possible reason for a higher accuracy seems to be the fact that the tag set of *CRM* is smaller (cf. method: tokens), and that it contains fewer contracted forms (cf. method: contracted). This would also explain why the accuracy increased the most with a memory-based learning approach. If there are fewer varied examples in memory, the tagger can more accurately identify the most similar token and attribute the correct tag to the test example. The downside of the smaller tag set is that the *CRM* data set contains fewer contracted triple forms, and, consequently, that on this specific task the performance of the taggers decreases drastically. However, this does not really influence the overall accuracy, since these triple forms only concern a small part of the test set. Furthermore, the higher frequency of these forms can facilitate the learning process. Another explanation we have mentioned earlier could be that *CG* has more variation in genre, as it also includes literary texts, whereas *CRM* only contains official documents.

Now, we will briefly look at the performance of the lemmatization task on *CG*. All taggers reached a better overall accuracy on this sequence labeling task. With MBT we reached 89.08% accuracy, with HunPos 90.24% and with the neural model 92.23%.

As illustrated in results1: CG lemma, we encountered the same tendencies as for PoS tagging, but each tagger performed slightly better on lemmatization. This was mainly caused by a higher

CG: LEM	MBT	HunPos	PIE-FW
Single LEM tags	89.77%	90.92%	92.90%
Double LEM tags	78.08%	79.33%	81.64%
Triple LEM tags	38.48%	40.17%	41.29%
Quadruple LEM tags	0.00%	0.00%	0.00%
Overall LEM	**89.08%**	**90.24%**	**92.23%**

Table 7: Accuracy of the three models on *Corpus Gysseling* for lemmatization and comparison of the performance on the contracted forms.

CRM: LEM	MBT	HunPos	PIE-FW
Single LEM	92.79%	93.51%	94.93%
Double LEM	87.56%	87.83%	88.23%
Triple LEM	19.05%	19.05%	28.57%
Quadruple LEM	0.00%	0.00%	0.00%
Overall LEM	**92.64%**	**93.35%**	**94.74%**

Table 8: Accuracy of the three models on *Corpus van Reenen/Mulder* for lemmatization and comparison of the performance on the contracted forms.

accuracy on single lemma tags, which make up the biggest part of our data set. However, the neural model performed with a slight decrease in accuracy while tagging double and triple tags, and so did HunPos for double tags. As mentioned earlier, in our neural modal we adopted a word-level approach. This affected the results on unknown tokens. On lemmatization with *CG* our model achieved an accuracy of 51.15%. On PoS tagging, the predictions were more accurate (69.44%). Since there were less than 5% unknown tags, this did not have a large influence on the final accuracy score.

In results1: CRM lemma we see a similar distribution for the lemmatization of *CRM*. With MBT we reached a total accuracy of 92.64%, with HunPos 93.35% and with the neural approach 94.74%. With HunPos and MBT, the overall performance was higher for lemmatization than for PoS tagging. However, the model built with the PIE framework achieved a slightly lower accuracy on this task, even though it performed better on the triple tags than the other two models. Moreover, it outperformed the PoS tagging task on these contracted forms. The accuracy of our neural model on unknown tokens while lemmatizing was 52.85%. For the PoS tagging task, this accuracy increased to 79.59%.

If we compare the results for the lemmatization of the two data sets, we notice the same tendencies as for part-of-speech tagging, though the difference in accuracy is smaller (*CRM*: MBT: +3.56%, Hunpos: + 3.11% and PIE-FW: +2.51%). In this case, all three taggers used the same parameters on both data sets.

Concerning the performance on unknown tokens, we notice that the neural models had more difficulties with lemmatization than with PoS tagging. This is not surprising since the lemma sets of both data sets were much larger than their tag sets, and the tagger had already encountered most of the PoS tags during training. Moreover, for PoS tagging our model performed better on *CRM* than on *CG*. Again, this is probably due to the fact that the *CRM* tag set was far less complicated than the tag set of *CG*.

CG: PoS	MBT	HunPos	PIE-FW
NOU	87.20%	89.13%	91.82%
PD	88.39%	90.41%	91.88%
VRB	84.96%	87.77%	91.65%
ADJ	83.90%	86.59%	88.94%
ADV	73.46%	78.94%	83.36%
ADP	96.87%	96.77%	97.86%
NUM	89.07%	90.34%	94.60%
CON	92.57%	93.53%	96.04%
Overall	**87.83%**	**89.78%**	**92.39%**

Table 9: Accuracy for the three taggers on the *Corpus Gysseling* test set, with the specific goal of comparing the accuracy on each part of speech. We only considered the single tags (contracted forms were excluded).

CRM: PoS	MBT	HunPos	PIE-FW
NOU	90.24%	91.55%	93.41%
PD	92.72%	93.75%	95.48%
VRB	88.52%	90.32%	93.35%
ADJ	90.84%	91.81%	93.96%
ADV	84.57%	87.35%	91.42%
ADP	98.30%	98.37%	99.07%
NUM	96.33%	96.32%	97.76%
CON	96.65%	97.44%	98.22%
Overall	**92.00%**	**93.14%**	**95.03%**

Table 10: Accuracy for the three taggers on the *Corpus van Reenen/Mulder* test set, with the specific goal of comparing the accuracy on each part of speech. We only considered the single tags (contracted forms were excluded).

4.2 Error Analysis

In this section, we will look further into the difficulties the models encountered and the categories where the three taggers performed better or worse.

EA: CG PoS cat and EA: CRM PoS cat show the performance on single tags in more detail for both data sets, by evaluating the prediction for each part of speech separately. The taggers performed best on adpositions (ADP), conjunctions (CON), and numerals (NUM). The lowest accuracy was reached on adverbs (ADV). We presuppose that this is caused by the large number of morpho-syntactic subcategories, as illustrated in method: PoS cat. One could counter this hypothesis by pointing out the fact that on verbs (VRB) and pronouns/determiners (PD) a higher accuracy was reached than on adverbs, even though VRB and PD contained more subcategories. However, when we look at the full data set, including the non-unique tokens, we notice that the models were trained on double the number of VRB tags and triple the number of PD tags.

If we compare the two data sets, we notice that the three taggers performed better on all categories in the *CRM* data set. However, the difference was more apparent for the adverb category. Again, we believe that this is due to the fact that adverbs in *CRM* had less subcategories than in *CG* (cf. method: PoS cat).

To verify our hypothesis, we evaluated our trained taggers on the main parts of speech, without any morpho-syntactic values, thus on the so-called 'short' forms. The detailed results of this

	CG	*CRM*
PIE PoS	91.83%	94.82%
PIE LEM	92.23%	94.74%
PIE-MT PoS	91.71%	**94.89%**
PIE-MT LEM	**92.86%**	95.26%

Table 11: A comparison of the results we obtain for both tasks (PoS and LEM), without multi-task sequence tagging and with (-MT). The accuracy on the target task is printed in bold.

experiment are listed in appendix: CG short and appendix: CRM short in the Appendix (5). After the subcategories were removed during the evaluation, the performance improved substantially for all three taggers (on *CG*: MBT: +6.67%, HunPos: +6.00%, PIE-FW: +4.53%). As expected, the performance on the adverb category increased the most, e.g., with +7.72% on *CG* using the neural model (cf. appendix: CG cat short and appendix: CRM cat short).

For CG, we looked at the performance of our neural tagger based on text genre. On official texts, our PoS tagger scored 7.83% higher than on literary texts. The same tendency was noted in our predictions for the lemmatization task, with a 6.30% higher accuracy on official texts compared to literary texts.

Finally, in EA: MT, we compare the accuracy of the neural model on the *CG* and *CRM* data sets in a multi-task environment to those achieved for each task separately. As mentioned earlier (cf. 3.2.2), our target task in the combined model for *CG* was lemmatization. The performance increased on this specific task, but there was a slight decrease in accuracy for PoS tagging. The performance on more complex triple tags slightly decreased for the lemmatization task, and substantially improved on the PoS tagging task. For *CRM*, the target task was PoS tagging. The performance on the *CRM* data set increased only slightly on the target task. However, on lemmatization it increased with +0.52%. On *CRM*, the performance on triple tags did not improve.

5. Conclusion

We developed a novel neural model for historical linguistic enrichment, and rigorously evaluated this model on PoS tagging and lemmatization of historical Dutch, using two existing sequence taggers as baselines. The adequate automated annotation of historical Dutch corpora can facilitate further research. In general, we obtained better results with our a neural approach, even when there were not many training data, e.g., for the lemmatization of triple tags. Furthermore, we also noted that, as for classical Latin data (Longrée and Poudat 2010), a HMM-based model (HunPos) performed better on Middle Dutch than a memory-based model (MBT). In general, our model performed better on the lemmatization task than on the PoS tagging task. Concerning the results on *CRM*, our accuracy of almost 95% on both sequence tagging tasks was similar to the result of the tagger-lemmatizer of van Halteren and Rem (2013), who expanded the lexicon by including orthographic variations, and then used 10-fold cross-validation.

By evaluating our models, and by giving special attention to the difficulties the model encountered, we noticed that, next to sparse data, the task of attributing a part-of-speech tag to a token was rendered more difficult by the complicated PoS tag set, which included subcategories with inflectional information. The complicated tag set was a consequence of the morphological variation of Middle Dutch. A first solution would be to work with a simplified tag set, although the full tag information might contribute to the disambiguation. Another solution would be to predict this morphological and inflectional information separately, perhaps by adding the lemma and PoS category as separate features to execute the task.

By adding more training data, we would be able to improve our results, but unfortunately data in Middle Dutch is rather sparse. Another, and more realistic option, would be to adapt some decisions made during pre-processing. For instance, in *CG*, there were Latin tokens that had not been annotated with a lemma, and that had received "other" as tag. We decided to remove these tokens from our data set. However, some Middle Dutch words in the corpus were surrounded by Latin words. HunPos, as a trigram tagger, takes the context of a token into account, but after removing the Latin, the context of these tokens was processed differently. That would have been less of a problem if the end-of-sentence information had been correctly extracted from the corpus, but, as we explained in related, doing this would necessitate more manual labour. A possible solution to this issue would be to keep the Latin tokens in the data set, but to exclude them from evaluation, or to replace them by a marker <LATIN>. Another option would be to remove texts with more tokens in Latin than in Middle Dutch from the data set, using a threshold.

Another possibility to improve our results would be the integration of a lexicon-based lemmatization approach, in which a computational lexicon (containing a mapping between lemmas and inflections) would be used as an extra input to a neural model. However, because of the particular nature of Middle Dutch, this approach would have supplementary difficulties with correctly tagging out-of-vocabulary words, and with orthographic variation (Kestemont and De Gussem 2017). Therefore, a normalization step would have to be added, e.g., using Levenshtein distance (Kestemont et al. 2010). As previously mentioned, according to Manjavacas et al. (2019), normalization is not feasible for historical languages. However, van Halteren and Rem (2013) showed that by taking orthographic variation into account and by expanding the lexicon, they reached higher accuracy, with minimal increase in ambiguity.

During our evaluation, we noticed that the performance for lemmatization and part-of-speech tagging on official text seemed higher than on literary texts. We believe that, for both tasks, this result is influenced by the more rigid structure of official documents, as we discussed in intro. However, as our data set was not equally balanced based on genre, more research is needed to verify our hypothesis.

With this research article, we have shown that neural networks can be used as a powerful technique to automate the linguistic enrichment of historical Dutch corpora, if used in combination with human knowledge of factors like genre and historical variation. We hope that our findings will facilitate future linguistic research on historical corpora.

Acknowledgements

Part of this research was done as an internship at the Instituut voor de Nederlandse Taal by Silke Creten, who is currently working as a PhD student at KU Leuven. Peter Dekker was supported by funding from the Flemish Government under the *Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen.* programme. Supervision of the project was done by Vincent Vandeghinste, senior researcher at the Instituut voor de Nederlandse Taal and part-time researcher at Leuven.AI and the Centre for Computational Linguistics of KU Leuven.

We want to thank the anonymous reviewers for their valuable suggestions that have helped greatly improve the present article.

References

Brants, Thorsten (2000), TnT: A statistical part-of-speech tagger, *Proceedings of the Sixth Conference on Applied Natural Language Processing*, Association for Computational Linguistics, Seattle, p. 224–231. https://doi.org/10.3115/974147.974178.

Daelemans, Walter, Jakub Zavrel, and Peter Berck (1996), *Part-of-speech tagging of Dutch with MBT, a memory-based tagger generator*, Werkgemeenschap Informatiewetenschap, Delft,

pp. 33–40. http://www.cnts.ua.ac.be/papers/1996/dzb96.pdf.

Daelemans, Walter, Jakub Zavrel, Antal Van den Bosch, and Ko Van der Sloot (2010), *MBT: Memory-Based Tagger, version 3.2, Reference Guide*, Induction of Linguistic Knowledge, Tilburg.

Dai, Zihang, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov (2019), Transformer-XL: Attentive language models beyond a fixed-length context, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Florence, pp. 2978–2988. https://www.aclweb.org/anthology/P19-1285.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019), BERT: Pre-training of deep bidirectional transformers for language understanding, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Vol. 1, Association for Computational Linguistics, Minneapolis, pp. 4171–4186. https://www.aclweb.org/anthology/N19-1423.

Dipper, Stefanie (2010), POS-tagging of historical language data: First experiments, *in* Pinkal, Manfred, Ines Rehbein, Sabine Schulte im Walde, and Angelika Storrer, editors, *Proceedings of the 10th Conference on Natural Language Processing, KONVENS*, Saarland University Press, pp. 117–121.

Dipper, Stefanie (2011), Morphological and part-of-speech tagging of historical language data: A comparison, *Journal for Language Technology and Computational Linguistics* **26**, pp. 25–37, Gesellschaft für Sprachtechnologie und Computerlinguistik.

Halácsy, Péter, András Kornai, and Csaba Oravecz (2007), Hunpos - an open source trigram tagger, *in* Zaenen, Annie and Antal van den Bosch, editors, *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Prague.

Hochreiter, Sepp and Jürgen Schmidhuber (1997), Long Short-Term Memory, *Neural Computation* **9** (8), pp. 1735–1780, MIT Press. https://doi.org/10.1162/neco.1997.9.8.1735.

Kerckvoorde, Colette M. Van (1993), *An Introduction to Middle Dutch*, Mouton de Gruyter, Berlin, New York.

Kestemont, Mike and Jeroen De Gussem (2017), Integrated sequence tagging for Medieval Latin using deep representation learning, *Journal of Data Mining & Digital Humanities*, CNRS, INRA, INRIA. https://jdmdh.episciences.org/3835.

Kestemont, Mike and Karina Van Dalen-Oskam (2009), Predicting the past: memory-based copyist and author discrimination in medieval epics, *Proceedings of the twenty-first Benelux conference on artificial intelligence*, Eindhoven, pp. 121–128.

Kestemont, Mike, Guy de Pauw, Renske van Nie, and Walter Daelemans (2016), Lemmatization for variation-rich languages using deep learning, *Digital Scholarship in the Humanities* **32** (4), pp. 797–815, Oxford University Press. https://doi.org/10.1093/llc/fqw034.

Kestemont, Mike, Walter Daelemans, and Guy De Pauw (2010), Weigh your words—memory-based lemmatization for Middle Dutch, *Literary and Linguistic Computing* **25** (3), pp. 287–301, Oxford University Press. https://doi.org/10.1093/llc/fqq011.

Kingma, Diederik P. and Jimmy Ba (2015), Adam: A method for stochastic optimization, *in* Bengio, Yoshua and Yann LeCun, editors, *3rd International Conference on Learning Representations*. http://arxiv.org/abs/1412.6980.

Longrée, Dominique and Céline Poudat (2010), New ways of lemmatizing and tagging classical and post-classical Latin: the LATLEM project of the LASLA, *in* Anreiter, Peter and Manfred Kienpointner, editors, *Proceedings of the 15th International Colloquium on Latin Linguistics*, Innsbruck, pp. 683–694.

Manjavacas, Enrique, Ákos Kádár, and Mike Kestemont (2019), Improving lemmatization of non-standard languages with joint learning, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Vol. 1, Association for Computational Linguistics, Minneapolis, pp. 1493–1503. https://www.aclweb.org/anthology/N19-1153.

Megyesi, Beáta (2009), The open source tagger HunPos for Swedish, *Proceedings of the 17th Nordic Conference of Computational Linguistics*, Northern European Association for Language Technology, Odense, pp. 239–241. https://www.aclweb.org/anthology/W09-4636.

Piotrowski, Michael (2012), *Natural language processing for historical texts*, Vol. 5 of *Synthesis digital library of engineering and computer science*, Morgan & Claypool, San Rafael.

Rem, Margit and Hans van Halteren (2007), *Tagging and lemmatization manual for the corpus van Reenen-Mulder and the Adelheid 1.0 Tagger-Lemmatizer*, Radboud University, Nijmegen.

Schmid, Helmut (1994), Probabilistic part-of-speech tagging using decision trees, *Proceedings of the International Conference on New Methods in Language Processing*, Association for Computational Linguistics, Manchester.

Schmid, Helmut (1995), Improvements in part-of-speech tagging with an application to German, *Proceedings of the ACL SIGDAT-Workshop*, Association for Computational Linguistics, Dublin, pp. 47–50.

Schmid, Helmut (2019), Deep learning-based morphological taggers and lemmatizers for annotating historical texts, *Proceedings of Digital Access to Cultural Heritage*, Association for Computing Machinery, New York, pp. 133–137. https://doi.org/10.1145/3322905.3322915.

van Halteren, Hans and Margit Rem (2013), Dealing with orthographic variation in a tagger-lemmatizer for fourteenth century Dutch charters, *Language Resources and Evaluation* **47** (4), pp. 1233–1259, Springer, New York. https://doi.org/10.1007/s10579-013-9236-1.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin (2017), Attention is all you need, *in* Guyon, I., U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, Curran Associates, Inc., pp. 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Zavrel, Jakub and Walter Daelemans (1999), Recent advances in memory-based part-of-speech tagging, *in* de Lingüística Aplicada, Centro, editor, *Actas de VI Simposio Internacional de Comunicacion Social*, Ediciones Editoriales Oriente, Santiago de Cuba, pp. 590–597.

Appendix

CG: Short PoS	MBT	HunPos	PIE-FW
Single tags short	94.66%	95.90%	97.16%
Double tags short	81.76%	81.89%	87.44%
Triple tags short	41.85%	42.70%	45.22%
Quadruple tags short	0.00%	0.00%	0.00%
Overall	**93.90%**	**95.08%**	**96.36%**

Table 12: Results for PoS tagging on the contracted forms in the *CG* test set. Only the main parts of speech are taken into account (left column), the subcategories are ignored (morpho-syntactic values).

CRM: Short PoS	MBT	HunPos	PIE-FW
Single tags short	96.33%	97.25%	98.14%
Double tags short	88.43%	88.14%	90.71%
Triple tags short	23.81%	23.81%	19.05%
Quadruple tags short	0.00%	0.00%	0.00%
Overall	**96.11%**	**97.00%**	**97.93%**

Table 13: Results for PoS tagging on the contracted forms in the *CRM* test set. Only the main parts of speech are taken into account (left column), the subcategories are ignored (morpho-syntactic values).

CG: Short PoS	MBT	HunPos	PIE-FW
NOU	96.64%	97.87%	98.17%
PD	95.11%	96.30%	97.11%
VRB	93.41%	95.23%	97.03%
ADJ	89.55%	91.07%	92.95%
ADV	82.90%	86.67%	91.08%
ADP	98.07%	98.07%	99.15%
NUM	94.95%	95.25%	96.95%
CON	95.04%	95.96%	97.59%
Overalll	**94.66%**	**95.90%**	**97.16%**

Table 14: Results for PoS tagging on the short version of the single tags in the *CG* test set. Only the main parts of speech are taken into account (left column), the subcategories are ignored (morpho-syntactic values).

CRM: Short PoS	MBT	HunPos	PIE-FW
NOU	97.44%	98.30%	98.63%
PD	97.65%	98.08%	98.89%
VRB	94.88%	96.44%	97.70%
ADJ	93.00%	94.25%	96.05%
ADV	87.06%	90.23%	93.24%
ADP	98.57%	98.83%	99.33%
NUM	97.31%	97.58%	98.81%
CON	97.82%	98.35%	98.79%
Overall	**96.33%**	**97.25%**	**98.14%**

Table 15: Results for PoS tagging on the single tags in the *CRM* test set. Only the main parts of speech are taken into account (left column), the subcategories are ignored (morpho-syntactic values)