Computational Linguistics in the Netherlands Journal 10 (2020) 161–178 Submitted 10/2020; Published 12/2020

The CLIN30 Shared Task: Have-doubling in Historical Varieties of Dutch

Marijn Schraagen[1] M.P.Schraagen@uu.nl
Joanna Wall[1,2] J.H.Wall@uu.nl
Eduardo Brito[3] eduardo.alfredo.brito.chacon@iais.fraunhofer.de

[1] *Utrecht University, The Netherlands*

[2] *Meertens Institute, The Netherlands*

[3] *Fraunhofer Center for Machine Learning, Fraunhofer IAIS, Germany*

Abstract

The CLIN30 Shared Task is defined as a computational approach to classify have-doubling, which is a syntactic phenomenon combining a past participle construction with an additional participle, usually from the verb have, e.g., ‘he has had lived there’. This paper describes perfect doubling, a particular subform of the have-doubling construction, in detail, and introduces a dataset to study the phenomenon in a computational way. Different classification approaches from the Shared Task participants are discussed, and an error analysis of classification results is provided. The models reach up to nearly 80% accuracy, which is a viable starting point for further research.

1. Introduction

The field of computational linguistics has historically been used for various different research goals. One of these goals is the application of computational techniques to solve real world language-related tasks, with early examples in, e.g., machine translation (Locke and Booth 1955) and conversational agents (Weizenbaum 1966). Another goal is the study of language itself using computational methods, such as the research field known as lexicostatistics (Swadesh 1952) and the development of corpus linguistics (Kučera and Francis 1967). A third research goal is the development of computational linguistics as such, i.e., a more fundamental approach geared towards methodology rather than direct applications in the real world or in regular linguistics, for example the SemEval series (Agirre et al. 2009).

The Shared Task of the 30th Computational Linguistics in The Netherlands conference (CLIN30) is a text classification task about the syntactic construction have-doubling. The task is to develop a classifier based on a corpus of historical Dutch sentences that predicts whether a given sentence provides the context for have-doubling. The topic of the Shared Task is compatible with all three general research goals. First of all, there are various applications that could benefit from a computational analysis of have-doubling, such as modernization of historical text or relation extraction. Furthermore, regular linguistic analysis may gain insights from computational models of have-doubling, especially if such models are to some degree explainable. Finally, the dataset and objectives of the current Shared Task may be used to apply pre-existing methodologies in order to investigate how these models behave outside of the regular application domain.

This paper is organized as follows: first, the linguistic background on have-doubling will be discussed (Section 2), followed by an overview of Natural Language Processing aspects (Section 3). After the introduction, the task definition of the Shared Task and a description of the dataset are provided (Section 4). Next, the approaches and results of the task are described (Section 5), followed by an analysis of classification errors (Section 6). The paper concludes with a general overview and directions for further research (Section 7).

2. Linguistic background

Linguistically, syntactic doubling is the doubling of the same syntactic element (see e.g. Koeneman et al. (2011, p. 35ff.)). Perfect doubling is a type of syntactic doubling where the doubled element is the perfect auxiliary of a verb. This phenomenon occurs most frequently as have-doubling, with the perfect auxiliary *have* (Barbiers et al. 2008, p. 40). Formally, have-doubling constructions contain a form of the word have combined with a lexical past participle, as well as an additional, past participial form of *have.* An example from a historical variety of Dutch is given in (1). This example is found in the personal journal of the Dutch politician Willem Frederik from 1647.[1]

(1) *Graf Hendrick van den Berch heeft daer gewoont gehadt.*
Count Hendrick van den Berch *has* there *lived* *had*
Approx: Count Hendrick van den Berch lived there.

In present-day standard Dutch the perfect doubling construction is no longer present, however it remains in use in various dialects in the Netherlands and Flanders (example 2), as well as in varieties of French (3) and German (4), see e.g. Schaden (2007), Haß (2016). In Dutch a perfect tense can be constructed using either the verb *have* or the verb *be* as perfect auxiliary, and both in present-day dialectical Dutch and in historical varieties of Dutch the corresponding be-doubling construction is attested ((5) and (6)). Examples (2) – (5) are repeated from Koeneman et al. (2011), example (6) is found in Witkam (1986).

Despite the large number of language varieties in which perfect doubling is found, the occurrences are generally very low. The frequency in historical varieties of Dutch has been estimated at between 30 and 50 per 1 million words in selected time periods, and less than 10 per 1 million words overall (see (Wall 2018b, Figure 6), (Wall 2018a) and (Wall In preparation)).

(2) *Ik heb het gezegd gehad.*
I have it said had
I said it. (present-day South-eastern Dutch)

(3) *Quand j'ai eu dîné, je suis sorti.*
when I have had dined I am left
After having dined, I left. (present-day regional French)

(4) *Wia i hamkumman bin, hot mai schwesta den opfl scho gessen ghobt.*
as I home.come am has my sister the apple already eaten had
When I arrived at home, my sister had already eaten the apple. (present-day Bavarian)

(5) *Ik ben twee keer gevallen geweest.*
I am two times fallen been
I fell twice. (present-day South-eastern Dutch)

(6) *dat Mommillan niet en is gecomen geweest in handen van de Fransoisen*
that Mommillan not is come been in hands of the French
that Mommillan did not fall in the hands of the French (17th century Dutch)

2.1 Semantics

Considering the semantics of perfect doubling, in general there is little difference in meaning between a sentence with perfect doubling and the same sentence without doubling. Compare (1) and the constructed example (1a) without the additional past participle:

[1] https://www.dbnl.org/tekst/will077glor01_01/will077glor01_01_0008.php

(1) *Graf Hendrick van den Berch heeft daer gewoont gehadt.*
Count Hendrick van den Berch has there lived had
Count Hendrick van den Berch lived there.

(1a) *Graf Hendrick van den Berch heeft daer gewoont.*
Count Hendrick van den Berch has there lived
Count Hendrick van den Berch lived there. *(constructed example)*

These two sentences are almost completely interchangeable in meaning: they both express a past event. Research in linguistics has focused on this issue: if the meaning is so similar or even equivalent, then what is the reason that speakers of the language choose a perfect doubling construction over the run-of-the-mill form of the perfect?

A number of authors, e.g., Schaden (2007), Koeneman et al. (2011) distinguish two core readings of perfect doubling constructions: the anterior and superperfect. The anterior reading characterizes the use of perfect doubling as a relative past tense, i.e., it is used approximately to denote that one event is *anterior* relative to another (see Koeneman et al. (2011, p.72)). Sentence (4) is an example of this interpretation, which expresses approximately that the apple was eaten before the subject came home. The second, *superperfect* reading indicates “an action or state which is definitively complete and unlikely to recur; an action or state which took place or existed in a distant past; an action or state which occurred at an indeterminate point or points in time; an action or state which is in some way exceptional; heightened speaker involvement in the action or state on the part of the speaker” (Carruthers (1994) in Koeneman et al. (2011, p. 73)).

Note as well as perfect doubling constructions it has been argued that there is another type of have-doubling in historical varieties of Dutch with the same surface form but where an important difference compared to perfect doubling constructions is that the syntactic subject is not an agent, see e.g. Duinhoven (1997, pp. 346–348). Constructions with this broad interpretation are also found in, amongst other varieties, modern Dutch dialects, e.g., van Bree (1981). An example is shown in (7), repeated from Koeneman et al. (2011, p. 42).

(7) *Ik heb het haar geverfd gehad.*
I have the hair dyed had
My hair has been dyed. (Limburg Dutch)

In (7), the subject *have* can be interpreted as a possessor of the direct object, *het haar* ‘the hair’, rather than as the agent, i.e. the dyer of the hair.

In what follows, we will abstract away from the differences between these constructions and simply refer to both as have-doubling constructions.

2.2 Syntax

One syntactic factor which has been noted as significant for perfect doubling constructions in both modern Dutch dialects (Koeneman et al. 2011) and historical varieties of Dutch (Wall 2018a) is the word order restrictions in verb clusters in subordinate clauses. For Dutch dialects, these word order restrictions are those shown in (8), repeated from Koeneman et al. (2011, p. 44).

(8) ... dat ik de fiets

a. * *heb gehad gestolen.*
have had stolen

b. * *heb gestolen gehad.*
have stolen had

c. *gestolen gehad heb.*
stolen had have

d. *gestolen heb gehad.*
stolen have had

that I had stolen the bicycle.

(8) shows that, according to grammaticality judgements collected by and reported in Koeneman et al. (2011, pp. 41, 44), Dutch dialect speakers only allow orders in which the lexical past participle proceeds the other cluster elements (8c,8d); orders in which it follows one, (8b), or both elements, (8a) are disallowed. Similarly, based on a single-author corpus of the work of Early Modern Dutch author D. V. Coornhert, Wall (2018a, pp. 160–161) finds that the (8c,8d)-type orders overwhelmingly predominate although the (8b)-type is allowed in a minority of cases (cf. its *-status in modern Dutch dialects). Both Koeneman et al. (2011) and in turn Wall (2018a) relate this to the lexical participle having an adjectival rather than verbal categorial status (cf. Wall (2018b)). More broadly, this connects to an extensive literature on verb cluster variation in both historical (e.g., Coupé (2015), Coussé (2008)) and modern varieties of Dutch (e.g., Dros-Hendriks (2018)).

The current Shared Task formulates the analysis of have-doubling as a classification task on sentences, i.e., to predict whether a sentence provides the context for a have-doubling construction or not (see Section 4 for details). As a supplement to linguistic findings, error analysis of the classifiers as well as analysis of influential classification features may provide more insights into the linguistic properties of perfect doubling.

3. NLP perspective on have-doubling

With the advance of Digital Humanities, an increasing amount of Natural Language Processing (NLP) research is being performed on under-resourced languages and language varieties that lack large annotated corpora, lexicons, language models and tools. Research into have-doubling can be classified into this category, as both historical language varieties and present-day dialectical variants belong to the class of under-resourced languages. The current Shared Task contributes to this area by investigating how various approaches that are known to work well in a high-resource setting perform on a small amount of data. Furthermore, one of the participants investigates the usefulness of modern Parts-of-Speech (POS) tags for historical text in the context of this task (see Section 5.1). The task itself also provides an addition to the field of historical NLP given the focus on classifying a syntactic phenomenon, complementing related work that focuses on semantics (Moritz and Büchler 2017), morphosyntax (Kestemont et al. 2016), POS-tagging (van Halteren and Rem 2013, Hupkes and Bod 2016), translation and rephrasing (Pagé-Perron et al. 2017), or information retrieval (Gotscharek et al. 2011).

Possible applications include normalization of historical text into modern standard language, to allow the use of resources and tools intended for modern varieties (see for some examples Tjong Kim Sang et al. (2017), Scherrer et al. (2019), Ruzsics et al. (2019)). Given that have-doubling does not occur in present-day standard Dutch, the doubled verb needs to be removed during normalization. This presupposes that a have-doubling construction can be identified and possibly decomposed into elements, for the removal process to be successful. While identification of have-doubling based on POS-tags is relatively trivial, the lack of accurate POS-tagging tools for a certain language variety is often among the reasons for normalizing a text. Therefore, a non-trivial approach to identifying have-doubling constructions is needed. Similarly, a more language-agnostic statistical modernization approach based on example sentence pairs does not require the specific information that a historical or dialectical sentence contains have-doubling, however in order to create such sentence pairs for training it is necessary to correctly identify have-doubling constructions as such.

The identification of have-doubling constructions can also be valuable outside of a normalization context. Examples include increasing the accuracy of Information Retrieval related measures that are sensitive to the number of tokens in a sentence or n-gram overlap, such as cosine similarity or ROUGE scores (Lin 2004).

Furthermore, at the document level, the property of containing have-doubling somewhere in the text can be an interesting variable in document classification approaches, given that have-doubling is correlated with certain historical periods and a particular geographical distribution, and potentially with various other sociolinguistic dimensions of interest (Wall 2018b), (Wall In preparation).

4. Shared Task definition

The objective of the CLIN30 Shared Task is to determine whether a given sentence provides the context for a have-doubling construction. The main aspect of interest is to find out which properties of the sentence are associated to have-doubling, rather than to detect have-doubling as such. To operationalize this task, examples of have-doubling are preprocessed to strip out the defining occurrence of the past participial form of *have*. The task for the classifier is then to determine whether a sentence originally contained have-doubling or not, i.e., whether the *have*-participle has been removed from a sentence or not. As an example, sentence (1a) would be presented, for which the classifier should predict that this sentence was constructed from an actual example of have-doubling.

If such a classifier is successful, then an argument can be made that have-doubling sentences are indeed different from regular perfect tense sentences in other ways than just the seemingly redundant second participle. Analysis of features and results can provide further indications towards the nature of such differences. However, such conclusions must take possible bias in the selection of positive and negative examples into account.

4.1 Data

The data for the Shared Task consists of 1030 example sentences from historical varieties of Dutch, ranging from the 13th century to the 19th century[2]. Half of the examples (515 sentences) contain have-doubling. The other half of the data contains negative examples, i.e., perfect tense sentences without doubling.

The data originates from two different data providers. First, the Digital Library of Dutch Literature (DBNL)[3] has been used.[4] This resource contains documents from all historical time periods for written Dutch. The quality of the materials is usually very high, with virtually no transcription errors. The Shared Task dataset contains 277 positive examples and 515 negative examples from DBNL. The distribution over time for this part of the data is provided in Figure 1. Secondly, materials found using the search interface of the Nederlab project have been used, predominantly (227 examples) from the correspondence archive of the Dutch politician Anthonie Heinsius (1641-1720). All of the examples from this data source are part of the positive class, i.e., the sentences all contain have-doubling (see for further discussion of this issue Section 4.2). This resource consists of letters digitized using Optical Character Recognition (OCR). The source of the OCR is a printed edition from 1976 consisting of manual transcriptions of the original letters. Although the quality of this source is relatively high, the OCR results still contain a high degree of transcription errors. All of these documents originate from the period between 1702 and 1720, from a variety of correspondents. A few examples (11 in total) in the Nederlab part of the data originate from different sources. Note that the data curation policy and implementation of search tools and source formatting of both data providers influence the results of the data collection process, and the contents of the current dataset, while determined by the historical texts, is mediated by the context of the data provider.

[2] For the availability of the data see `https://git.science.uu.nl/-/snippets/61`.

[3] `https://www.dbnl.org`

[4] For technical implementation reasons the DBNL data collection has been performed from scratch using the DBNL website as a source. The resulting dataset is similar, although not identical, to data collected previously by Wall (see Wall (2018b) and (Wall In preparation) for details of her dataset and collection procedure as well as non-computational analyses).

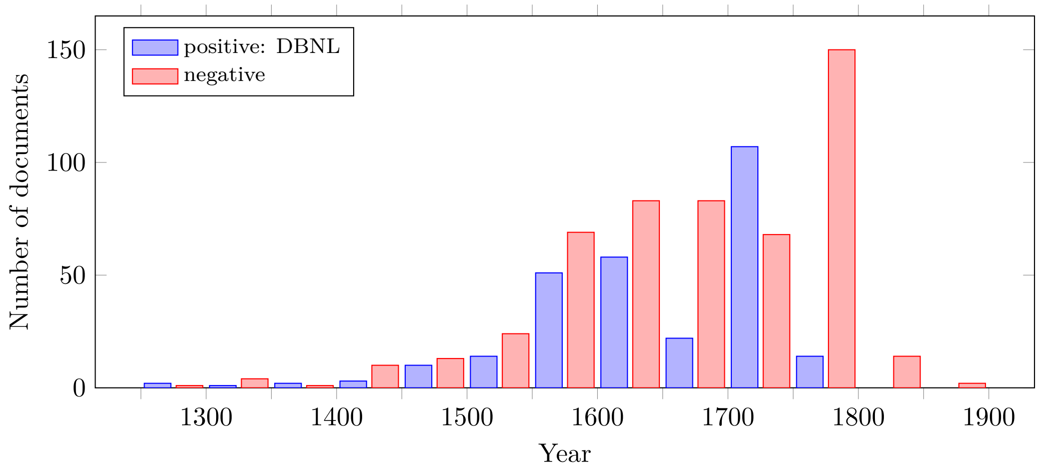

Figure 1: Distribution of publication year/year of writing

4.2 Example selection procedure

For the DBNL part of the data, using the search interface on the DBNL website all documents were retrieved that contain one of a certain group of historical forms of the past participle *had*. This includes the historical spellings *gehadt, ghehadt, gehat, ghehat* from the full timespan of DBNL, as well as all documents containing the modern form *gehad* that were originally published between 1500 and 1800. This resulted in around 20,000 documents in total. Secondly, these documents were cleaned to remove footnotes and margin notes, that often contain remarks by editors in present-day Dutch. Afterwards, all documents were POS-tagged using Frog (van den Bosch et al. 2007), which was configured with an Early Modern Dutch language model[5] that was trained on the Letters as Loot (Brieven als Buit) dataset (Rutten and van der Wal 2014).

Using the POS-tags, all phrases were selected that contain at least three verbs, of which at least two forms of the verb *have*. A sentence is defined by the language model in Frog, and may contain punctuation characters. A phrase is defined as a substring of a sentence delimited by a comma (or the start or end of the sentence). The pre-selection resulted in 6572 potential example sentences. After manual inspection[6] of all potential examples, 277 positive examples of have-doubling remained. For the negative examples (perfects without doubling) a similar procedure was implemented, with a pre-selection of all sentences with at least two verbs of which exactly one was a form of *have*. From this set, two (non-overlapping) random subsets were selected and manually checked, one to match the DBNL positive examples and one to match the Nederlab positive examples. This procedure resulted in a balanced dataset with an equal number of positive and negative examples (515 each).

It should be noted that, while the described procedure was performed exhaustively on all DBNL data, this does not imply that each and every example of have-doubling was actually found. Notably, the definition of a phrase is quite strict, and will cause some occurrences of have-doubling to be filtered out - however given the manual component in the example selection procedure it was opted to limit the number of results from pre-selection using this definition of a phrase. Moreover, the automatic POS-tagging is not 100% accurate, which means that examples may be missed containing verbs that are incorrectly tagged as a different Part-of-Speech.

[5] `https://github.com/LanguageMachines/frogdata/tree/master/config/nld-vnn`

[6] This manual inspection step has caused a small minority of examples not to confirm to the definition of have-doubling as stated (i.e., two forms of *have* with a lexical past participle complement). However, as the number of such instances appears to be small and the purpose here is a computational study, these do not detract from the overall conclusions.

The positive examples from Nederlab were collected with the Nederlab search interface[7] in Spring and Summer 2018[8]. Parallel to the DBNL data, a search was made for the modern form *gehad* and a minimally larger group of historical spelling variants to those used above (*ghehad, gehadt, ghehadt, gehat, ghehat*)[9] in texts whose metadata indicated that they had an author who died between 1701 and 1750[10]. The resulting examples were then manually sorted to include only instances of have-doubling, and exclude perfects of lexical *have.* Nederlab houses various digitalized texts. The majority of these examples originated from the *Briefwisseling van Anthonie Heinsius 1702-1720*, which contains the correspondence both from and to the important Dutch statesman in that period[11]. Taken on its own this large set of positive examples has the advantage of being homogenous for genre and time period, which may make it easier to find which other variables are behind the use of have-doubling. In addition six examples that were found using the specified query in the Nederlab interface are hosted by Delpher[12], which features digitalized copies of a range of texts held in academic institutions, and five examples are hosted by DBNL, which were kept as part of the Nederlab category instead of being grouped together with the other DBNL data.

Finally, full sentences from which the selected phrases originated have been retrieved for inclusion into the dataset. The sentence borders were manually identified for each instance. Generally this selection follows the grammatical definition of a sentence, however in the case of historical varieties of Dutch some sentences can be extremely long, in which case a suitable part of the sentence has been used. Further, when a sentence contained two or more different occurrences of have-doubling, the sentence was split to create the corresponding amount of positive examples. The average sentence length is relatively high (around 48.5 words) and many sentences contain multiple subordinate clauses and verb clusters (cf. (Burridge 1993, p. 12)). Furthermore, sentence partitioning was performed by a single annotator, which may introduce inconsistencies in the data. To check the impact of these data characteristics on classification results additional experiments have been performed using a fixed-length sentence window, as discussed at the end of the current section. For the Nederlab examples, all OCR errors in the verbs that are elements of the have-doubling construction were manually corrected, as well as a number of other OCR errors in the sentences. Most errors are examples of single character substitutions and whitespace errors, as illustrated in Figure 2. The corrections in the sentences outside of the verbs were however not performed thoroughly, and many errors remain in the data.

As mentioned in Section 4.1, even though the dataset has an equal number of positive and negative examples, all negative examples were selected from documents obtained through the DBNL data provider, whereas the positive examples were selected from DBNL as well as Nederlab. In order to check for possible bias caused by this variable, additional experiments have been performed on the DBNL set only (i.e., training without Nederlab positive examples and a smaller set of negative examples matching the size of the DBNL positive set). Of course the amount of training data is reduced significantly by this experiment, which may also influence the results.

Another variable of interest is sentence length, which will be discussed in Section 6.2.1. The variation in length is very large, ranging from just a few words to over 100 words in the same sentence. To control for this variable an additional experiment was performed in which the input

[7] `https://www.nederlab.nl/onderzoeksportaal/?action=verkennen`

[8] Note the date of the searches is relevant as both Nederlab and the collections therein have since undergone updates and may as a result return different results.

[9] However, all examples in the resultant dataset involve one of the spelling variants *gehad*, *gehadt* and *ghehadt*, and as such this dataset differs little from the DBNL dataset. Indeed, the vast majority of examples in both datasets contain either the modern form *gehad* or the historical variant *gehadt*, hence the spelling variant of *have* does not constitute a significant difference between the datasets.

[10] The reasons for the confinement of these examples to this period was largely practical: as noted in Section 2.1 the overall frequency of *have*-doubling is very low, which also made it difficult to construct a suitable dataset for the task. As such, one of the authors made available a large number of examples from previous work conducted on this phenomenon to supplement the above DBNL data, which fell only within this period.

[11] `http://resources.huygens.knaw.nl/briefwisselingheinsius`

[12] `https://www.delpher.nl`

Haar **Ho. Mo.** genoegzaam alleenig **nvt** consideratie voor de dry hoge magten van de **Quadrup -le** Alliantie **gedehbereert** hebben om de accessie tot dezelve, dewijle **door** geene **traetaten** zijn verbonden [...] ende **naderhanl** dat **diversehe resolution** om tot de gemelte Quadruple Alliantie **Ie aeeederen.**
Haar **Ho.Mo.** genoegzaam alleenig **uyt** consideratie voor de dry hoge magten van de **Quadruple** Alliantie **gedelibereert** hebben om de accessie tot dezelve, dewijle **door** geene **tractaten** zijn verbonden [...] ende **naderhant** dat **diversche resolutiën** om tot de gemelte Quadruple Alliantie **te accederen,**

Figure 2: Example of manual OCR correction: C. Hop to Heinsius, September 15th, 1719. Corrections indicated in bold.

selected phrase dye ten tijde van haer eerste beroepinge zeer weynich tydts hyer **gewoont hebben gehadt**
full sentence Alsmen dan daertegen naedenckt, hoe dat nu lange iaren achtervolcht is den voet om in den kerckenraedt veel uytheemschen te gebruycken, ia zelfs degeene, dye ten tijde van haer eerste beroepinge zeer weynich tydts hyer **gewoont hebben gehadt.**
window om in den kerckenraedt veel uytheemschen te gebruycken, ia zelfs degeene, dye ten tijde van haer eerste beroepinge zeer weynich tydts hyer **gewoont hebben gehadt**

Figure 3: Example of sentence selection: C.P. Hooft, *Memoriën en Adviezen*, 1611. Have-doubling indicated in bold.

sentences were cut off to a window of 25 word tokens. The window was centered around the lexical participle, with 12 tokens on either side of the participle token. In case the participle occurred near the start or end of the sentence the window was shifted accordingly (i.e., if the participle occurred within 12 tokens from the start then the window consisted of the first 25 tokens of the sentence, and if the participle occurred within 12 tokens from the end then the window consisted of the final 25 tokens). If the full sentence was less then 25 tokens then the window consisted of the full sentence. An example of the selection procedure is shown in Figure 3.

5. Approaches and results

Two teams participated in the Shared Task, one from the Fraunhofer Center for Machine Learning (Fraunhofer IAIS, Germany) and one from Utrecht University (The Netherlands). The participants were given a dataset consisting of all sentences[13], the labels, and metadata (source url, author, year, verbs cluster). However, the task prescribed that the classification must be performed using only the sentences and the labels, with the metadata to be used for analysis of the classification results. POS and other syntactic information was not provided to the participants.

The Fraunhofer team used the RatVec framework (Brito et al. 2019) to apply kernel PCA (Principal Component Analysis) using a POS-based similarity function. The Utrecht team used various machine learning algorithms with bag-of-words features, including experiments using word embeddings for dimension reduction. The performance of the different approaches is measured as classification accuracy, i.e., the ratio of correct predictions to all test examples, computed using 10-fold cross-validation. The training and test examples in each fold are selected randomly.

[13] The Fraunhofer team used a dataset of 1032 examples that contained a duplicate positive entry and an additional corresponding negative entry. For the experiments of the Utrecht team these two entries were removed.

5.1 Details of the Fraunhofer approach

The Fraunhofer approach treats each sentence as a sequence of POS-tags that are modeled via the RatVec framework (Brito et al. 2019), which generates a low-dimensional vector representation for each sequence by aplying kernel PCA and trains a k-nearest neighbors classifier. Given a list of n sequences S and a kernel function f, training the representation learning model consists of the following steps:

1. Compute a kernel matrix K by evaluating f, the similarity function on all sequence pairs
$$K_{ij} = f(s_i, s_j)\ \forall s_i, s_j \in S \tag{1}$$
2. Diagonalize K and select the d eigenvectors $v_1, \ldots, v_d$, where $d \leq n$ belonging to the largest eigenvalues.
3. Construct a projection matrix P by dividing the selected eigenvectors by their respective eigenvalues
$$P = \left[\frac{v_1}{\lambda_1}, \ldots, \frac{v_d}{\lambda_d}\right]. \tag{2}$$

Any new sequence t not belonging to the training dataset S can get a vector representation by evaluating f on t against all sequences in S. The product of the resulting kernelized distance vector with the projection matrix P constitutes the d-dimensional representation r_t of t:

$$r_t = P^\top f(t, S) \tag{3}$$

For this Shared Task in particular, S consisted of coarse POS-sequences related to a set of randomly selected sentences from the subcorpora j, k, l, m, and n of the Corpus Gesproken Nederlands (CGN) (van Eynde et al. 2000), where the number of sentences varied from 2000 to 8000 depending on the training dataset. The kernel function f was the composition of the 2-spectrum kernel (Leslie et al. 2002) with either a RBF kernel or a polynomial kernel[14]. The shared task sentences were converted to POS sequences[15] by means of the Frog parser (van den Bosch et al. 2007). In order to improve POS-tagging, the historical sentences were translated to modern Dutch with the tool provided for the CLIN27 Shared Task (Tjong Kim Sang et al. 2017). Then, the POS-tag sequences were transformed to low dimensional vectors via the trained RatVec model, which constitutes the training dataset for a k-nearest neighbors classifier. As such, these experiments allowed us to assess the usefulness of modern POS tags for historical text. Preliminary experiments have been performed using historical data POS-tagged with a historical language model, instead of the CGN. These experiments however resulted in low accuracy scores, therefore these experiments have not been developed further. An overview of the best performing classifiers of this approach is displayed in Table 1.

5.2 Details of the Utrecht approach

In the Utrecht approach, three existing machine learning algorithms were applied to have-doubling classification. First, a Multinomial Naive Bayes classifier was used as implemented in the Python library Scikit-learn. Multinomial Naive Bayes is often used in text classification, either using word frequency or using tf-idf vectors. In the current experiments, the Naive Bayes classifier was used to demonstrate the performance of a simple machine learning algorithm, therefore a basic word

[14] The tested hyperparameter combinations including kernel choice, dimensionality of the produced vectors, and k-value for the k-nearest neighbors classifiers can be found in the open-sourced repository: `https://github.com/ebritoc/clin30_ratvec`

[15] Although POS-tags were used while constructing the dataset, the final set given to participants did not include the tags. Therefore, any approaches for which POS-tags were used needed to re-tag the data.

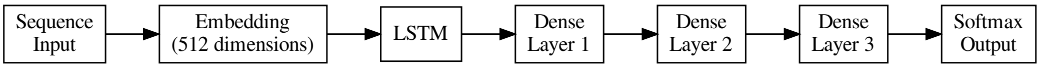

Figure 4: Layout of LSTM network

Approach	*Full data*	*DBNL*	*Full data,*	*DBNL,*
			window	*window*
RatVec	0.67	0.61	0.64	0.63
Naive Bayes	0.77	0.71	0.75	0.75
Logistic Regression	**0.79**	0.73	0.78	0.75
LSTM, pre-trained embeddings, fixed	0.65	0.61	0.66	0.62
LSTM, pre-trained embeddings, adaptive	0.71	0.64	0.70	0.67
LSTM, on-the-fly embeddings	0.73	0.65	0.75	0.67

Table 1: Overview of classification accuracy

frequency count for each example sentence was used as input to the classifier. Tokenization of the data was performed by the Scikit library, which converts the input to lower case and uses a pattern consisting of 2 or more alphanumeric characters to represent tokens, with punctuation characters used as token separator. This creates a vocabulary of around 10 thousand words on which the term frequency matrix was based. The classifier itself has very few parameters, which have been left to Scikit defaults.

A second set of cross-validation experiments was performed using a Logistic Regression classifier from the Scikit library. The same tokenization was used. The classifier uses the Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (lbfgs) as the solver and a multinomial distribution over the labels[16]. The input consisted of a bag-of-words representation of the sentences, similar to the Naive Bayes experiments.

The third set of experiments used a Long Short Term Memory network (LSTM) as implemented in the Python library Keras with TensorFlow backend. An LSTM is a recurrent neural network that is particularly suited for sequential data such as natural language sentences. Moreover, this approach provides a straightforward way of incorporating word embeddings, by adding an embedding layer on top of the input (see Figure 4). In the experiments the softmax activation function, the categorical cross-entropy loss function[17] and the Adaptive Moment Estimation optimizer (Adam) were used.

Two different sets of word embeddings were used for the LSTM experiments. The first set consisted of pre-trained Word2Vec embeddings generated from a 4.5 million word corpus of Early Modern Dutch (1600–1750) obtained from DBNL. With this set two experiments were performed, in the first experiment the embeddings were fixed, while in the second experiment the pre-trained embeddings were adapted during training. The second set consisted of on-the-fly embeddings based on the input data which were trained together with the classifier.

The results of the experiments are listed in Table 1.

[16] Other parameters have been tested, such as a liblinear solver and the one-versus all class distribution. These settings did not significantly influence the performance of the Logistic Regression classifier.

[17] During initial experiments, categorical cross-entropy surprisingly outperformed a binary loss function.

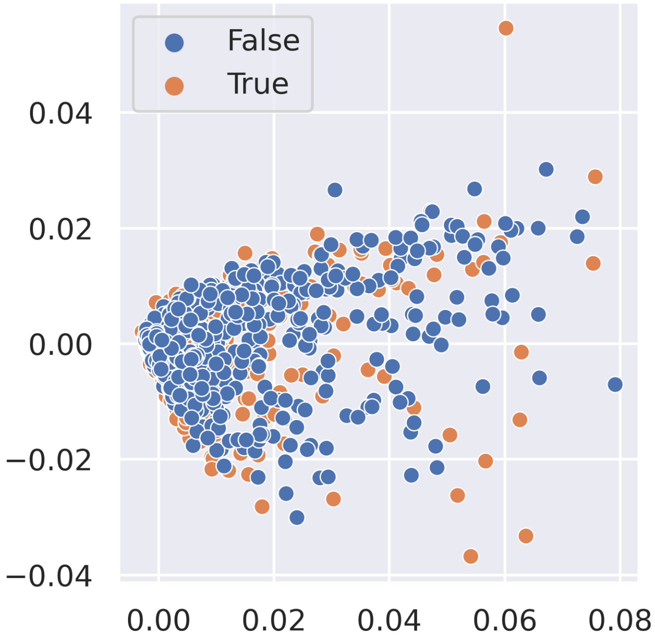

Figure 5: Sentences represented by their two first principal components obtained via the RatVec approach. The positive class refers to the sentences containing have-doubling.

6. Error analysis

The error analysis presented in this section is performed on the results of the experiments with the full dataset. The results regarding influence of bias in the data as derived from the additional experiments are discussed in Section 7.

6.1 Kernel Principal Components Analysis

The vector representations derived from the RatVec approach allow the visualization of the represented sentences: similar sentences (according to the applied similarity function) are mapped to vectors close to each other in their vector space. Figure 5 shows the two first principal components for the RatVec classifier colored according to their class. This visualization provides some insights into the performance of the classifier. By checking the represented sentences, the first principal component (x-axis) seems to be correlated with sentence length. In fact, a Pearson's correlation of -0.62 (calculated with the sentence length measured as number of tokens) validates this observation (shorter sentences tend to appear rather in the right-hand side of Figure 5). This can be explained by the applied 2-spectrum kernel. It indirectly makes sentences “distinguishable” by their number of tokens since the longer the sentences are, the more likely they will have bigrams in common with other sequences. Table 2 shows that sentences containing have-doubling (positive class) are on average longer than the negative examples, which explains that sentence length is a discriminative factor. The combination of both principal components shows a correlation with the data source:

Class	*Mean*	*Standard deviation*
positive (+)	205	114
negative (–)	146	73
Total	176	100

Table 2: Statistics on number of tokens per sentence

most of the positive outliers in the bottom left of the graph are found in the Nederlab part of the data while positive DBNL examples can be found in the area to the top right of the main cloud.

6.2 Logistic Regression

From the Utrecht experiments, the Logistic Regression classifier showed the best performance, therefore the following error analysis was performed for this classifier. While analysis for Naive Bayes has not been performed to the same level of detail, various observations for Logistic Regression are also applicable to the Naive Bayes results, for example occurrences of irrealis in false positives, or the general difficulty in classifying short sentences.

6.2.1 Sentence length

Error analysis of the Logistic Regression experiments shows a difference in sentence length of around 5 words on average between correctly classified and incorrectly classified sentences. Figure 6 (left) shows the length differences for a typical run of the Logistic Regression algorithm. These results indicate that very short sentences (< 5 words) are difficult to classify. The figure also shows that the distribution of errors extends to longer sentences as well, however the distribution of correctly classified sentences extends a bit further, including sentences of over 75 words in length. This indicates that, while shorter sentences are indeed somewhat more difficult, the difference in the average is mainly caused by the performance differences for longer sentences. Still, the average sentence length is very high. Most sentences contain multiple subordinate clauses, of which only one clause is part of a have-doubling construction. However, the current state of the art in parsing historical varieties of Dutch is not yet sufficiently accurate to reliably identify relevant subordinate clauses for use in classification. Note that the additional experiments on using a window of 25 words centered on the lexical past participle showed as well that long sentences have only limited impact on classification accuracy over the datasets in general.

6.2.2 Publication year

Another possible source of bias is publication year, given that this variable is unevenly distribution over the classes (see Figure 1 and further discussion in Section 4.2). To evaluate the impact of this variable, an analysis was performed to check errors grouped by publication year, as shown in Figure 6 (right). The figure shows that this variable is unlikely to be a source of bias, given that correct and incorrect classifications are found in the full time period of the corpus.

6.2.3 Class error distribution and data source

The errors are approximately equally distributed over the two classes. However, the data source has a large effect on classification. Out of the false negative examples (i.e., the original sentence contained have-doubling but the classifier predicted that the example was a sentence without have-doubling) almost all examples originate from the DBNL set, while the Nederlab set does not contribute a significant amount of false negatives. This bias can be explained by the fact that all negative examples are extracted from DBNL data, while Nederlab data only contributed to positive examples. Therefore, when an example is more similar to DBNL, it has a higher probability to be negative,

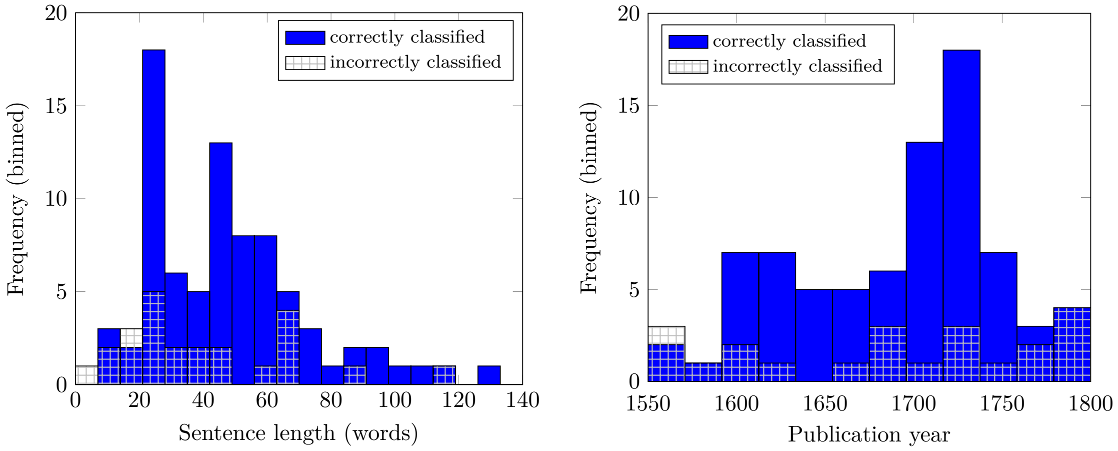

Figure 6: Logistic Regression results by sentence length (left) and publication year (right) for a typical run.

and vice versa. The two data sources show a large difference in vocabulary and spelling, because Nederlab contains OCR errors while DBNL does not, and because the time span of DBNL is very large while Nederlab data originates mostly from the first half of the 18th century. The influence of the data sources is examined in further experiments, as mentioned in Section 4.2 and discussed further in Section 7.

6.2.4 Analysis of individual sentences

Detailed inspection of individual misclassified sentences shows a number of potential causes for error. Sentences (9) and (10) show false positives[18]. Sentence (9) shows a modal construction using three verbs, which is not considered an example of have-doubling although it shares some similarities (notably a present tense form of *have* combined with a past participle *had*). However, in this case the complement of *gehadt* is not another past participle, but the noun phrase *de Papieren van Butkens* (the papers of Butkens). Note that, because this is not a form of have-doubling, the participle *gehadt* is present in the example, whereas in the training set used to construct the classifier the word *gehadt* and any of its variants has been stripped from have-doubling constructions. The presence of this token therefore is unlikely to have influenced the decision of the classifier, however the similarities in the context may have been sufficient to result in a false positive classification. Indeed a test in which this sentence is presented to the classifier with and without *gehadt* shows that in both cases the sentence is classified as (false) positive. Another factor which might make (9) be misidentified as have-doubling is that it is an irrealis, as shown by the presence of the auxiliary *zoude* 'would'. Have-doubling constructions are thought to be particularly frequent in the irrealis (e.g. (Kern 1912, p. 290)).

The source of Sentence (10) is a song book from 1745 taken from DBNL. This is relatively late compared to true positive examples of have-doubling from DBNL (see Figure 1). The modern spelling of the participle *gedaan* (done) contrasts with the spelling *gedaen* from earlier sources. The modern spelling is relatively more prevalent in the Nederlab part of the data (*gedaan* constitutes 45% of occurrences of *gedaen/gedaan*) vs. the DBNL part (28% *gedaan*). Because the Nederlab part provided only positive examples, the classifier may be biased towards classifying modern spelling as a case of have-doubling, although this effect is partly counterbalanced by the fact that a significant

[18] For presentation purposes only the relevant phrases from the full sentences are shown.

part of the negative examples (from DBNL) is also modern[19]. In order to prevent (or at least discover) this kind of false positive error it will therefore be useful to take such domain knowledge into account, for example by postprocessing classification results using publication year as a factor. Note that while this particular error may be an example of dataset bias, in general the source of the examples does not appear to be a significant source of bias, as discussed in Section 7.

Sentence (11) is an example of false negative prediction. Sentence (11) contains the participle *getemoigneert*, which is a French loan word that occurs only two times in the entire dataset. Therefore, it is unlikely that the classifier can identify this word as a participle and therefore as part of a have-doubling construction. A pattern-based POS-tagging approach, or for example using subword units, may prove useful to improve performance for such low-frequency vocabulary items (see Section 7 for further discussion).

(9) *Aangaande de Papieren van Butkens, die ouwendijk zoude gehadt hebben*
Regarding the papers of Butkens, that ouwendijk should had have
Regarding the papers of Butkens, that Ouwendijk supposedly had had

(10) *ook eenige daar je een Wys op zult moeten maken, zo als ik gedaan heb*
also some there you a melody on should must make, so as I done have
Also some to which you have to make a melody, as I have done

(11) *dat sijn ongenoege aan dien commissaris hadde getemoigneert gehadt*
that his disapproval to the commissioner had communicated had.
that he had communicated his disapproval to the commissioner

Note however that the analysis presented in this section is predicated on a syntactic interpretation of the patterns and generalizations learned by the Logistic Regression classifier. While such an interpretation is potentially valid, the possibility should be taken into account that the classifier operates on a different level entirely, and an apparent correlation with syntactic patterns is coincidental. Further feature and error analysis is needed to establish the validity of the current interpretation.

7. Discussion and future work

The results of the experiments show that simple models such as Naive Bayes and Logistic Regression perform better than more complex models such as neural networks and principal components analysis on the task of have-doubling prediction on the Shared Task dataset. For the LSTM networks this can be explained partly by the word embedding approach. The pre-trained embeddings performed the worst of all classifiers, which is likely to be caused by the high out-of-vocabulary rate of this set of embeddings (68%). A more extensive set of embeddings may improve the performance of this approach. Alternatively, subword embeddings (Bojanowski et al. 2017) and approaches incorporating byte-pair encoding (Devlin et al. 2019) may be able to overcome the issue of out-of-vocabulary words, although for the current dataset these approaches are not expected to result in large performance improvements given the small amount of data. Allowing the pre-trained embeddings to be adapted during training did improve performance, but compared to the fully on-the-fly trained embeddings the pre-trained embeddings did not offer any additional accuracy. While the embeddings that were trained on-the-fly on the Shared Task data showed the highest performance of the LSTM models (0.73 accuracy), this model was still significantly worse than the simple models. This is likely to be caused by the small size of the dataset, which does not provide sufficient context for computing accurate

[19] A test that presented two versions of this sentence to the classifier with different spelling of *gedaan* did not show the assumed bias towards the modern spelling. However, the other words in the sentence are also spelled in a modern way, which may be responsible for the prediction error. No further spelling changes were tested, in order to avoid creating a fully artificial example.

word embeddings. Furthermore, even though the results obtained with 10-fold cross-validation were stable in repeated experiments, the individual folds did show considerable variation. This provides additional indications that the dataset is too small to allow for training a robust LSTM model. The results of the separate experiments for the DBNL part of the data reinforce this explanation, with consistently lower scores for the smaller dataset.

As noted in Section 4.2 the selection of negative examples and the distribution of sentence length and time period may influence classifier behavior, i.e., the data characteristics may cause the classifier to recognize the biased variables instead of the actual have-doubling constructions. However, the results of additional experiments that controlled for these variables indicate that the classification algorithms are robust to the amount of bias on the data. The experiment on only DBNL data showed some performance drop for each classifier, which is expected when the amount of training data is reduced by 50%, but the performance remains clearly above chance levels by a wide margin. In the condition of equal sentence length all models regain some performance for the DBNL set. This is expected given that the window is generally shorter than the original sentence and more focused on the have-doubling context with a much reduced presence of unrelated other verbs and clauses. More importantly, however, it indicates that sentence length is not used as a spurious proxy for have-doubling classification. Interestingly, the window effect is much less pronounced for the full dataset, with only improvement for two LSTM variants and a marginally decreased performance for the other models. The differences are however rather minor, so these results do not point towards a bias effect for sentence length. Also for the RatVec approach, while some sentence length and dataset bias was observed in the error analysis, the results of the additional experiments show that this method still has discriminative power for have-doubling constructions when the bias variables are controlled.

Considering the performance differences between the Utrecht approach and the RatVec system, the Principal Component Analysis approach in the RatVec system was trained on the CGN, which is rather different from the language used in the historical, written sources of the Shared Task. Training on a different, more closely related corpus (such as Letters as Loot (Rutten and van der Wal 2014)) is likely to result in improved performance.

None of the proposed models provide straightforward ways to explain the predictions of the classifier, limiting the possibilities of gaining linguistic insights from the models. However, feature and error analysis has shown a possible linguistic approach of analyzing the examples of have-doubling based on classifier results. In future work post-hoc explainability (such as Štrumbelj and Kononenko (2014)) or intrinsically explainable models such as decision trees or bayesian networks could be investigated to implement this aspect as envisaged by the Shared Task organizers.

As have-doubling is highly infrequent in corpora, this may make it less suitable as a linguistic phenomenon to be analysed by current computational linguistic means. However, despite the lack of data the best model in the current experiments reaches up to 80% accuracy. The CLIN30 Shared Task has therefore provided a first step in analyzing have-doubling computationally, which may help further research to address the final 20%.

Acknowledgement

Eduardo Brito was supported in parts by the Fraunhofer Research Center for Machine Learning (RCML) within the Fraunhofer Cluster of Excellence Cognitive Internet Technologies (CCIT) and by the Competence Center for Machine Learning Rhine Ruhr (ML2R) which is funded by the Federal Ministry of Education and Research of Germany (grant no. 01—S18038B). We gratefully acknowledge this support.

Marijn Schraagen and Joanna Wall were supported by the Language Dynamics in the Dutch Golden Age research projected, financed by the Netherlands Organisation for Scientific Research (NWO) (Grant 360-78-02), and Joanna Wall was additionally supported by the Meertens Institute. In addition, we would like to thank Feike Dietz and Marjo van Koppen for their input.

References

Agirre, Eneko, Lluís Màrquez, and Richard Wicentowski (2009), *Computational Semantic Analysis of Language: SemEval-2007 and Beyond*, Vol. 43 of *Language Resources and Evaluation*, Springer.

Barbiers, Sjef, Johan van der Auwera, Hans Bennis, Eefje Boek, Gunther De Vogelaer, and Margreet van der Ham (2008), *Syntactic Atlas of the Dutch dialects*, Amsterdam University Press.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2017), Enriching word vectors with subword information, *Transactions of the Association for Computational Linguistics* **5**, pp. 135–146.

van den Bosch, Antal, Bertjan Busser, Sander Canisius, and Walter Daelemans (2007), An efficient memory-based morphosyntactic tagger and parser for Dutch, *Selected Papers of the 17th Computational Linguistics in the Netherlands Meeting*, Netherlands Graduate School of Linguistics, pp. 99–114.

van Bree, Cor (1981), *Hebben-constructies en datiefconstructies binnen het Nederlandse taalgebied. Een taalgeografisch onderzoek*, PhD thesis, Leiden University.

Brito, Eduardo, Bogdan Georgiev, Daniel Domingo-Fernández, Charles Tapley Hoyt, and Christian Bauckhage (2019), RatVec: A general approach for low-dimensional distributed vector representations via rational kernels, *Proceedings of LWDA 2019*, CEUR-WS, pp. 74–78.

Burridge, Kate (1993), *Syntactic change in Germanic*, John Benjamins.

Carruthers, Janice (1994), The passé surcomposé régional: Towards a definition of its function in contemporary spoken French, *Journal of French Language Studies* **4** (2), pp. 171–190, Cambridge University Press.

Coupé, Griet (2015), *The historical development of Dutch verb clusters*, LOT Publications.

Coussé, Evie (2008), *Motivaties voor volgordevariatie: een diachrone studie van werkwoordvolgorde in het Nederlands*, PhD thesis, Ghent University.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019), BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805v2.

Dros-Hendriks, Lotte (2018), *Not another book on verb raising*, LOT Publications.

Duinhoven, Antonius (1997), *Middel-Nederlandse syntaxis: synchroon en diachroon 2. De werkwoordgroep*, Martinus Nijhoff.

van Eynde, Frank, Jakub Zavrel, and Walter Daelemans (2000), Part of speech tagging and lemmatisation for the Spoken Dutch Corpus, *Proceedings of LREC 2000*, ELRA, pp. 1427–1434.

Gotscharek, Annette, Ulrich Reffle, Christoph Ringlstetter, Klaus Schulz, and Andreas Neumann (2011), Towards information retrieval on historical document collections: the role of matching procedures and special lexica, *International Journal on Document Analysis and Recognition* **11**, pp. 159–171, Springer.

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**, pp. 1233–1259, Springer.

Haß, Norman (2016), Doppelte Zeitformen im Deutschen und im Französischen, *Beiträge zur germanistischen Sprachwissenschaft*, Vol. 24, Helmut Buske Verlag.

Hupkes, Dieuwke and Rens Bod (2016), Pos-tagging of historical Dutch, *Proceedings of LREC 2016*, ELRA, pp. 77–82.

Kern, Johan (1912), *De met het participium praeteriti omschreven werkwoordsvormen in 't Nederlands*, Johannes Müller.

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* pp. 1–19, Oxford University Press.

Koeneman, Olaf, Marika Lekakou, and Sjef Barbiers (2011), Perfect doubling, *Linguistic Variation* **11** (1), pp. 35–75, John Benjamins.

Kučera, Henry and Nelson Francis (1967), *Computational Analysis of Present-Day American English*, Brown University Press.

Leslie, Christina, Eleazar Eskin, and William Stafford Noble (2002), The spectrum kernel: a string kernel for SVM protein classification, *Proceedings of the 7th Pacific Symposium on Biocomputing*, World Scientific, pp. 566–575.

Lin, Chin-Yew (2004), ROUGE: a package for automatic evaluation of summaries, *Proceedings of the Workshop on Text Summarization 2004*, Association for Computational Linguistics, pp. 74–81.

Locke, William and Donald Booth, editors (1955), *Machine Translation of Languages. Fourteen Essays.*, MIT Press.

Moritz, Maria and Marco Büchler (2017), Ambiguity in semantically related word substitutions:an investigation in historical bible translations, *Proceedings of the NoDaLiDa 2017 Workshop on Processing Historical Language*, ACL, pp. 18–23.

Pagé-Perron, Émilie, Maria Sukhareva, Ilya Khait, and Christian Chiarcos (2017), Machine translation and automated analysisof the sumerian language, *Proceedings of the SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, ACL, pp. 10–16.

Rutten, Gijsbert and Marijke van der Wal (2014), *Letters as Loot: A sociolinguistic approach to seventeenth- and eighteenth-century Dutch*, Vol. 2 of *Advances in Historical Sociolinguistics*, John Benjamins Publishing Company.

Ruzsics, Tatiana, Massimo Lusetti, Anne Göhring, Tanja Samardžić, and Elisabeth Stark (2019), Neural text normalization with adapted decoding and POS features, *Natural Language Engineering* **25**, pp. 585–605, Cambridge University Press.

Schaden, Gerhard (2007), *La sémantique du Parfait. Étude des "temps composés" dans un choix de langues germaniques et romanes.*, PhD thesis, University of Paris 8.

Scherrer, Yves, Tanja Samardžić, and Elvira Glaser (2019), Digitising Swiss German: how to process and study a polycentric spoken language, *Language Resources and Evaluation* **53**, pp. 735–769, Springer.

Štrumbelj, Erik and Igor Kononenko (2014), Explaining prediction models and individual predictions with feature contributions, *Knowledge and Information Systems* **41**, pp. 647–665, Springer.

Swadesh, Morris (1952), Lexico-statistic dating of prehistoric ethnic contacts: with special reference to North American Indians and Eskimos, *Proceedings of the American Philosophical Society* **96** (4), pp. 452–463, American Philosophical Society.

Tjong Kim Sang, Erik, Marcel Bollmann, Remko Boschker, Francisco Casacuberta, Feike Dietz, Stefanie Dipper, Migual Domingo, Rob van der Goot, Marjo van Koppen, Nikola Ljubešić, Robert Östling, Florian Petran, Eva Pettersson, Yves Scherrer, Marijn Schraagen, Leen Sevens, Jörg Tiedemann, Tom Vanallemeersch, and Kalliopi Zervanou (2017), The CLIN27 Shared Task: Translating historical text to contemporary language for improving automatic linguistic annotation, *Computational Linguistics in The Netherlands Journal* pp. 53–64.

Wall, Joanna (2018a), Have-doubling constructions in historical and modern Dutch, *Linguistics in the Netherlands* **35** (1), pp. 155–172, Algemene Vereniging voor Taalwetenschap.

Wall, Joanna (2018b), *Seeing double: the HAVE puzzle*, Master's thesis, Utrecht University.

Wall, Joanna (In preparation), Extralinguistic properties of have-doubling in historical varieties of Dutch. Submitted for publication in Nederlandse Taalkunde.

Weizenbaum, Joseph (1966), ELIZA–a computer program for the study of natural language communication between man and machine, *Communications of the ACM*, Association for Computing Machinery.

Witkam, Paula, editor (1986), *Briefwisseling van Hugo Grotius. Deel 12*, Martinus Nijhoff.