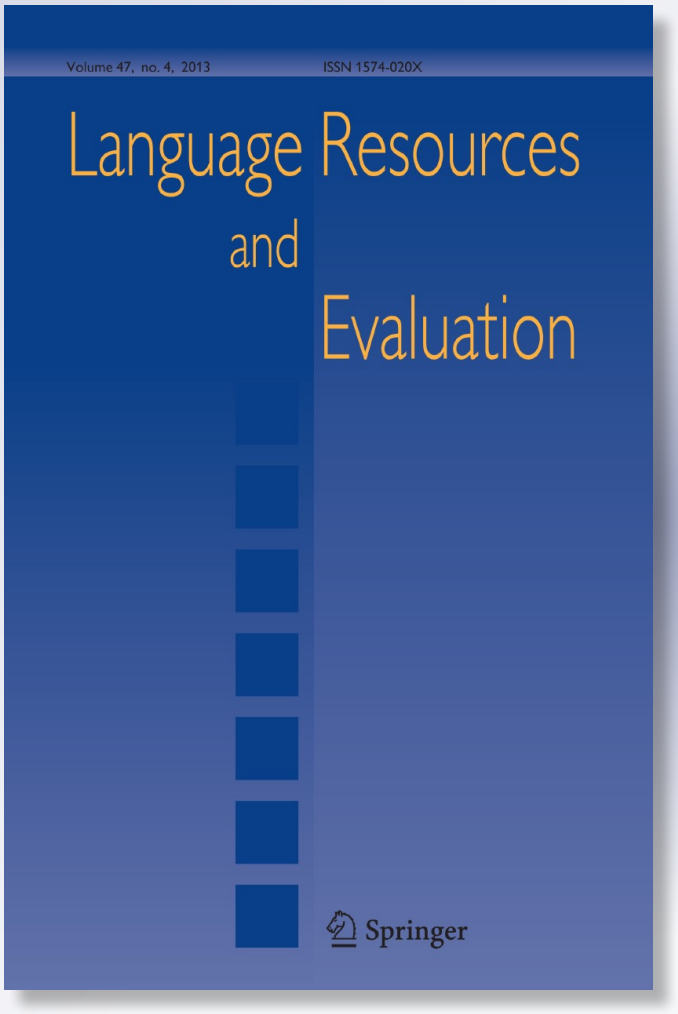
Dealing with orthographic variation in a tagger-lemmatizer for fourteenth century Dutch charters

Hans van Halteren & Margit Rem

Language Resources and Evaluation

ISSN 1574-020X
Volume 47
Number 4

Lang Resources & Evaluation (2013)
47:1233-1259
DOI 10.1007/s10579-013-9236-1



Springer

Lang Resources & Evaluation (2013) 47:1233–1259
DOI 10.1007/s10579-013-9236-1

ORIGINAL PAPER

Dealing with orthographic variation in a tagger-lemmatizer for fourteenth century Dutch charters

Hans van Halteren · Margit Rem

Published online: 29 May 2013

Abstract In this paper we describe a tagger-lemmatizer for fourteenth century Dutch charters (as found in the corpus van Reenen/Mulder), with a special focus on the treatment of the extensive orthographic variation in this material. We show that despite the difficulties caused by the variation, we are still able to reach about 95 % accuracy in a tenfold cross-validation experiment for both tagging and lemmatization. We can deal effectively with the variation in tokenization (as applied by the authors) by pre-normalization (*retokenization*). For variation in spelling, however, we choose to expand our lexicon with predicted spelling variants. For those forms which can also not be found in this expanded lexicon, we first derive the word class and subsequently search for the most similar lexicon word. Interestingly, our techniques for recognizing spelling variants turn out to be vital for lemmatization accuracy, but much less important for tagging accuracy.

Keywords POS tagging · Lemmatization · Spelling variation · Orthographic variation · Historical text · Middle Dutch

1 Introduction

Word class tagging is currently assumed to be a well-understood task, for which increases in the state of the art are measured in tenths, even hundredths of percents on benchmark datasets. Lemmatization is considered a bit more challenging, but is

H. van Halteren (✉)
CLS—Department of Linguistics/CLST, Radboud University Nijmegen, Nijmegen, The Netherlands
e-mail: hvh@let.ru.nl

M. Rem
CLS—Department of Dutch, Radboud University Nijmegen, Nijmegen, The Netherlands
e-mail: M.Rem@let.ru.nl

also rather well dealt with. These observations, however, apply only to these tasks in general. For specific types of text material, tagging and lemmatization are still tasks whose problems are far from solved. In this paper, we address one of those types of text, fourteenth century Dutch.

In the fourteenth century, Dutch orthography was far from standardized, not only at the level of spelling, but also at the level of punctuation and spacing. During the Middle Ages, a group of dialects made up the Dutch language without any dialect prevailing as the dominant one. The Latin alphabet was used to try to put these dialects into writing. This resulted in many spelling variants, regional variants and clitic combinations. A massive increase in societal and interregional interaction, at the end of the Middle Ages, created the need for a more standardized language. For Dutch and its dialects, this led to fundamental changes in grammar and usage during the fifteenth and sixteenth century, and a gradual standardization during the sixteenth and seventeenth century.

Currently, there is a new drive toward digitalization of historical Dutch texts, not only for historical linguists, but also e.g. for historians. More variation in the user group implies that not all users are actually well-versed in reading the historical variants of Dutch and it would be useful to enrich the material with annotation layers, at least lemma and POS tag layers, enabling a more efficient access for everybody. As an example, assume that a researcher would like to investigate the development of prices during the fourteenth century; some of the relevant texts could e.g. be located with the word *penning* ("coin"), but without annotation our researcher would have to guess at least the 56 different spelling variants that we see in the 1,055 occurrences in our corpus. But how, then, should such annotation layers be created? Preferably mostly automatic, but for Dutch text from the fourteenth century, we are not aware of any available existing tagger-lemmatizers. There used to be the LIMA system developed at the Free University of Amsterdam for semi-automatic tagging of (among others) Middle Dutch (Huber 1988, 1989), but this was unfortunately lost during computer upgrades (van Reenen, personal communication). Furthermore, tagging with LIMA still needed a substantial amount of manual work. The work in the late 1980s, however, did produce a reasonably sizable annotated corpus (*Corpus of fourteenth century Dutch* also known as *Corpus van Reenen/Mulder, CRM;* Rem 2003*, Chapter* 2), which can be used as the basis for the creation of an automatic tagger-lemmatizer. This of course means that the tagging and lemmatization guidelines of that time have to be largely followed. However, this is not a major problem as these guidelines were created by historical linguists specializing in the Dutch of the target period and the resulting material is still the de facto standard. Still, as we will see below, some details could be handled better, such as the lemmatization of proper names. However, we decided to leave such details as they are and withhold adaptations until later, possibly as a part of a unification of the tagging and lemmatization for the whole history of Dutch, which can be investigated as soon as sufficient material for all periods becomes available, which is now being realized in the *Nederlab* project (see http://www.nederlab.nl).

For the task at hand, there is a fundamental choice as to how the processing pipeline is set up. For historical text, a very popular approach is a preprocessing step which normalizes the orthography, after which more or less standard NLP tools are applied;

the probably best known example of a normalizer is the VARD (Baron and Rayson 2008). Treatment of variants can also be integrated into the NLP tool itself. This means that more information becomes available to help process the variety, but also that any NLP tools cannot be off-the-shelf and will need adjustment or will even have to be created from scratch. A theoretically possible third option, applying an NLP tool and then correcting its output for spelling variants, does not appear very promising. The VARD approach was designed for English and appears to be rather successful there. However, a significant amount of manual work is sometimes still needed in normalizing a text. Also, the texts under investigation already show a reasonable degree of standardization: less than 40 % of the tokens are variants even for the oldest studied period, fifteenth century, with a slight peak towards 50 % around 1500 (Baron et al. 2009; in comparison, a similar measurement on the CRM shows 75 % variants) and it is unknown how the performance is affected when going to texts with more extreme variation. Similar results are achieved for other languages where the VARD or other normalizers have been used. Reynaert et al. (2012) apply VARD2 and TICCL (Reynaert 2005) to Portuguese letters. Again, very impressive overall accuracies of 94.7 % (VARD) and 93.5 % (TICCL2) are reported, and tests per period showed that VARD's accuracy on the ten sixteenth and seventeenth century letter was hardly lower than the overall one (Hendrickx and Marquilhas 2011). However, as we see that only about 25 % of the tokens in these ten letters needed normalization, we must conclude that the Portuguese from that period is also already reasonably stabilized. For German, however, we can see what happens beyond the thresholds of the Modern periods (Bollmann 2012): a combination of normalizers managed to reach a normalization accuracy of 93.5 % of the tokens on the 1545 Luther bible, but only 73.6 % on Anselm texts (various versions from fourteenth to sixteenth century; Bollmann et al. 2011). Since our texts are more like such older texts, since we recognize the amount of manual work involved, and since we have the luxury of a lemma-annotated corpus, we decided to opt for the second approach and build a special purpose system, learning not only POS statistics but also orthographic variation statistics from training material and extrapolate to other potential word form variants.

In this paper, we describe in some detail how we approached the creation of a tagger-lemmatizer based on the CRM. The detail will focus mostly on our treatment of the orthographic variation in the text material, this being the most influential difference with the text material usually discussed in the tagging literature. Section 2 describes the corpus with specific attention for the orthographic variation in that corpus. Section 3 describes the tagger-lemmatizer as a whole and Sect. 4 gives a detailed description of our strategy to deal with the extreme orthographic variation. Section 5 lists the results of tenfold cross-validation experiments with the charters in the corpus. Section 6 briefly discusses some related work and Sect. 7 gives our conclusions and plans for future work.

2 Orthographic variation in the corpus of fourteenth century Dutch

In this section we describe the corpus material we are using in this paper (Sect. 2.1) and illustrate the orthographic variation that we find (Sect. 2.2).

2.1 Corpus of fourteenth century Dutch

The *Corpus of fourteenth century Dutch* (*Corpus van Reenen/Mulder, CRM*) was originally created by M.J. Mulder and P.Th.van Reenen and consists of 2,700 charters from 345 places of origin. Charters are a distinct type of formal documents, and typically make a grant of land or record a privilege. They treat, subject to certain procedural requirements, Legal Affairs and are intended to serve as evidence thereof or to make them public. The nature of charters leads to the presence of specific formulaic expressions. However, these expressions usually differ per location and they also show a clear development during the fourteenth century. Furthermore, the formulae tend to be present mostly at the beginning and end of the charters, with the middle allowing for much more freedom.

The CRM was designed to be representative of the local language use of Middle Dutch and to be suitable for all types of linguistic research, including dialectal, morphological and phonological research. In order to be eligible for inclusion in the corpus, a charter had to be original and the date of creation should be present in the charter itself. Furthermore (Rem 2003), the parties named in the document had to be from the same place or area, and none of them should be of a high social status. The assumption was that a charter was more representative of a genuinely local variant of language use if the participants are all local and of relatively humble stature. After all, individuals of high social status are in a position to recruit staff from different areas, and can be expected to have many supra-regional contacts which might promote the use of non-local variants. Recently, the CRM has been extended, by Chris De Wulf, Margit Rem and others, with approximately 1,000 further Flemish and Brabantic charters, thus providing an even better coverage of the entire Middle Dutch area. For the experiments described in this paper, however, we used only the original CRM as the newer additions had at that time not yet been annotated with tag and lemma information.

All charters were transcribed diplomatically, i.e. it was attempted to retain all features of the original manuscript, such as spacing and punctuation, rather than to interpret the text and replace these features according to the conventions valid at the time of transcription. An example charter and transcription are shown in Figs. 1 and 2. Token separation, spelling, capitalization and punctuation were all left intact. The mixed use of *u/v/w* and *i/j* was not normalized. Abbreviations were resolved, but marked in such a way that the original form could be reproduced. In some cases, spacing was adjusted, merging or splitting tokens, sometimes according to project guidelines as to which tokens constituted whole words but sometimes also according to the opinion of the transcriber; again the original form was kept recoverable. Unreadable or badly readable passages were put between square brackets. After transcription, the texts were extensively manually annotated. Supported by the LIMA tool (Huber 1988, 1989), part-of-speech tags and lemmas were added (for more detail, see Sect. 3.1 below). Furthermore, the parts of separated verb forms or pronominal adverbs were marked as such. For part of the corpus, the most important aspects of the syntactic structure were also provided. In this paper, we only address the annotation with a Tag-Lemma Pair (henceforth called TLP). During annotation, although the process did contain correction by a

senior researcher, there was no measurement of inter-annotator agreement, nor has any systematic measurement of the quality of the annotation been undertaken afterwards. Still, although the experiments with our system will lead us to identify some types of problems, our impression is that, overall, the annotation is not worse than that of other annotated corpora, with an error rate of only a few percent.

2.2 Orthographic variation in the CRM

In the corpus we see both extensive variation in the spacing applied by the authors and in the spelling they use for the various words. We do not address punctuation here, as any treatment would have to be based on some kind of analysis of the syntactic and rhythmic properties of the text, seeing that Greidanus (1926) observes that periods and capitalization do not indicate sentence separation, but depends on where the author judges there to be some kind of split in thought or rhythm. For statistics on thirteenth century prose, we refer the reader to Gerritsen (1990).

As for spacing, it was decided to let the transcribers adjust spacing (leaving access to the original) in order to separate the perceived individual words as described in the transcription guidelines. Such adjustments are limited to the spacing itself; no adjustments are made for clitic forms which are in principle recognizable as such, but where one or more parts have mutated during cliticization so that some kind of component reconstruction would be necessary. In the original manuscripts, there are about 770,000 points where spacing is observed. In about 1 % of these, the transcribers decided remove the spacing: adjacent tokens were merged into a single word. Mostly, this concerned the merging of two tokens, but there were also about 1,200 3-part merges and even 4 4-part merges. A frequent example of two tokens that were merged by the transcribers is the sequence COMPASS DIRECTION + *zijde*, e.g. *noordzijde* ("north side"). If we consider all possible compass directions, we find 1,108 occurrences, of which 742 (67 %) were written in the manuscripts as one token and 266 (33 %) as two tokens. On the other hand, the transcribers also inserted an additional 1.5 % spacings by splitting about 12,000

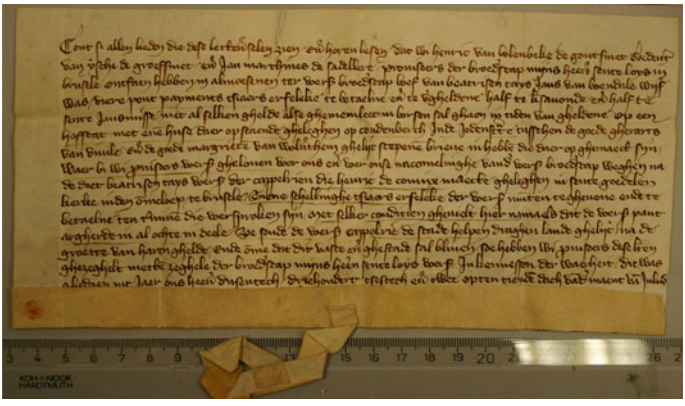


Fig. 1 An example charter from the Corpus of fourteenth Century Dutch

Cont si allen lieden die dese lett(er)en selen zien / En(de) horen lesen Dat wi henric van bolenbeke de goutsmet Goedeu(er)t
van ysche de groefsmet / en(de) jan marthines de sadellere / Prouisoers der broed(er)scap mijns he(er)en sente loys in
brusele Ontfaen hebben in almoesenen ter voers(eider) broed(er)scap boef van beatrisen tays Jans van boendale wijf
was / viere pont payments tsiaers erfeleke te betaelne en(de) te v(er)gheldene half te k(er)sauonde / en(de) half te
sente Jansmisse / met al selken ghelde alse ghemeinleec in borsen sal ghaen in tiden van gheldene / Op een
hofstat met ene(n) huse daer opstaende gheleghen op coudenberch Jnde Jodenst(ra)te tusschen de goede gherarts
van vinile En(de) de goede margriete(n) van wolu(er)them ghelijc scepene(n) brieue in hebbe(n) die daer op ghemaect sijn /
waer bi wi P(ro)uisoers voers(eit) ghelouen voer ons en(de) voer onse nacomelinghe vand(er) voers(eider) broed(er)scap weghen na
de doet beatrisen tays voers(eit) Der cappelrien die henric de coninx maecte gheleghen in sente goedelen
kerke inden o(m)meloep / te brusele / Tiene schellinghe tsiaers erfeleke der voers(eider) mu(n)ten te gheuene ende te
betaelne ten t(er)mine(n) die voersproken sijn / Met selker conditien gheuielt hier namaels dat de voers(eide) pant
argherde in al ochte in deele / Soe soude de voers(eide) Cappelrie de scade helpen draghen lande ghelijc / na de

Fig. 2 Transcription of the charter shown in Fig. 1

tokens. A frequent example here is the sequence Adposition-Article. We find this sequence 30,547 times in the corpus. In 7,795 of the cases, we are clearly dealing with an enclitic form in which one or both parts have been mutated, e.g. *te der* ("at the") written as *ter* and *met den* ("with the") written as *metten*, so that they have to be considered as a single token. Of the other 22,752 cases, where there are no mutations but where the component words are simply concatenated, 17,482 (77 %) are written as two separate tokens and 5,270 (23 %) as a single token. For the annotation of the corpus, it was decided to view this sequence as two tokens, so that 5,270 separators were inserted by the transcribers.

A much more complex situation is presented by the variation in spelling, as most words provide several positions where variation is possible. As an example, Table 1 shows the word forms which have been assigned the lemma *gelijk* ("similarly") and the tag adverb, a combination which is present 691 times in the corpus, with 24 different word forms. The variation is even greater for proper names, where the corpus designers decided to group all forms under the standardized name forms listed by Van der Schaar (1992). The resulting extensive perceived variation is witnessed by the name *Gerard*, whose 1,119 occurrences (in uninflected condition) show 66 different forms, listed in Table 2.

As can be expected, the number of observed variants increases as the length of the word goes up (since there is more potential for variation) and as the frequency of the word goes up (since more variation can be observed). In our corpus, we examined the word forms associated with the about 14,000 TLPs occurring more than once.[1] We included the tag in order to isolate actual spelling variation and to avoid counting variation due to various inflectional forms. We then grouped these TLPs into length and frequency bands. As for length, we grouped the TLPs by taking the integral part of the $\log_2$ of the average length of the corresponding word forms, but included the ten TLPs with an average word length lower than 2, which should therefore in principle be assigned to band 0, into length band 1. This lead to 4 bands with band 1 containing 134 TLPs consisting on average of less than 4 characters and band 4 of 940 TLPs consisting on average of 16 up to 21.8 [the plural noun *heiligegeestmeesters* ("administrators of the Poor Relief Fund")]. As for frequency, we grouped the TLPs by taking the integral part of the $\log_2$ of the frequency, with a maximum of 10, as the TLPs with a higher frequency than 2^{11}

[1] We excluded hapaxes as they obviously cannot show spelling variation. Interestingly, the 760,000 words of the corpus include only 11,000 hapax TLPs.

Table 1 Word forms found when the lemma *gelijk* ("similarly") is used as an adverb

ghelijc	373	gheliich	10	ghelic	3	gheljjch	1
gheliic	86	gelijch	9	gelic	2	ghelljc	1
gelijc	64	gelike	9	geliic	2	ghlijc	1
ghelike	54	gheliken	9	ghelijck	2	gilycs	1
ghelijch	33	gelyck	4	ghelijcke	1	like	1
ghelyc	19	ghelich	4	ghelijct	1	yegelike	1

Table 2 Word forms associated with the proper name *Gerard*

gheriit	121	gert	12	gerijd	4	gherut	2
gherijt	111	gerat	11	gheeraert	4	garret	1
gherart	84	gerit	10	gheredt	4	ger	1
gheret	70	gheerd	10	gheryd	4	geraed	1
gherit	70	geraert	8	gherrijd	3	gerairt	1
gherijd	58	gerd	8	ghierart	3	gerard	1
gerart	56	gheryt	8	gered	2	gerid	1
gheert	55	gerijt	7	gereet	2	geriit	1
gheriid	54	gheeraerd	7	geyrart	2	geryt	1
gheraerd	47	gheerard	7	gheeraerdt	2	gheerlec	1
gherd	47	gherret	7	gheeraet	2	gheraird	1
ghert	46	geerd	6	gheerit	2	gherrid	1
ghered	37	gherid	6	gher	2	gherud	1
gheraet	24	geraet	5	gherairt	2	gherydijn	1
gheeraed	19	geret	5	gherardt	2	gierkijn	1
gherard	18	gheraed	5	gherat	2		
gheraert	16	geert	4	gherrijt	2		

occur far to little to assign them separate bands. This led to 10 groups with group 1 containing 3,695 TLPs with 2 or 3 occurrences, up to group 10 containing 158 TLPs with 1,024 or more occurrences. For all TLPs, we then measured the spelling variation in two ways. The associated number of different forms shows the overall range of options and the entropy (Shannon 1948) for the choice of the form shows whether these options are taken more or less in equal amounts or whether there are one or more dominant options. On average, a TLP type has 3.12 spellings and an entropy of 0.83, which shows that there are indeed more dominant and less dominant forms as equal probabilities for all forms would lead to an entropy of about 1.6. The average variation in the different band combinations can be seen in Figs. 3 and 4. Note that, as these graphs show averages over given length and frequency bands, the extremes appear lower than they in fact are. The highest numbers of forms are found for the adverbs *erfelijk* ("hereditarily", 97 forms), *jaarlijks* ("each year", 89 forms) and *tegenwoordig* ("these days", 78 forms). The highest three entropies are found for again the adverb *erfelijk* (4.8) and also again *tegenwoordig*, but now as an adjective inflected to its *–n* and *–e* forms (5.1 and 5.8). It was unexpected that the extremes are found for adjectives rather than proper

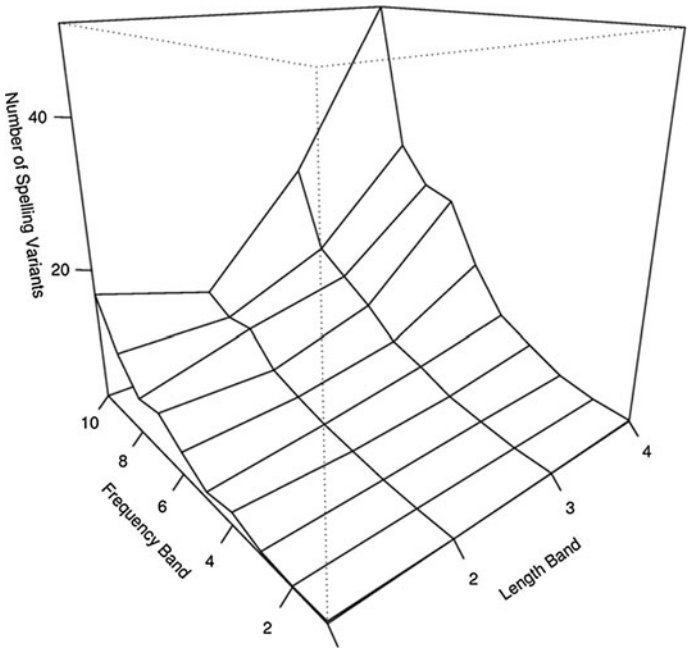


Fig. 3 The number of spelling variants as a function of the length and frequency of the TLP

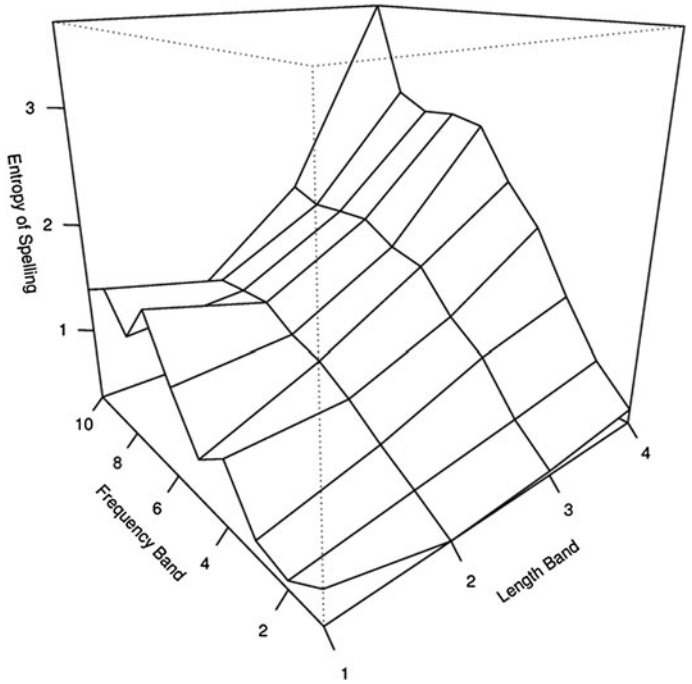


Fig. 4 The entropy of the spelling variation as a function of the length and frequency of the TLP

names, but on closer examination it turns out that the forms also include a lot of morphological variants which are not marked as such in the tag. Looking at the averaged graphs, we see that, with the exception of some irregularities, e.g. the ones caused by low numbers of TLPs per cell in length band 1, the number of variants indeed appears to be mostly correlated to both length and frequency (Fig. 3). However, if we look at entropy (Fig. 4), it would seem that the general trend is that entropy is highest in the medium frequency bands. In the highest frequency bands, there may be more forms, but most of these turn out to be rare. An example is the finite verb form *doen* ("do"). It has 8 variant spellings, but one of these (*doen* itself) is used in 798 of its 819 occurrences.

3 Tagger-lemmatizer architecture

In this section, we describe our tagger-lemmatizer as a whole. We start by describing the nature of the annotation (Sect. 3.1). Then we describe the pipeline architecture (Sect. 3.2) and its components: tokenization (Sect. 3.3), TLP suggestion (Sect. 3.4) and contextual disambiguation (Sect. 3.5).

3.1 Word class tags and lemmas

We transformed the numerical tags in the original annotation of the corpus to a mnemonic format for reasons of readability, using an attribute notation as exemplified by V(fin,past,lex) (a ***V***erb in a ***fin***ite form, namely ***past*** tense, and being a ***lex***ical (rather than an auxiliary) verb). The part before the parentheses indicates the part of speech (see Table 3). The parts between the parentheses indicate various more detailed subgroupings. In all, we observe 184 different tags in the corpus. For the moment, this also limits the number of tags that the tagger will output, as it is automatically trained on the corpus. However, we should note that the annotation also distinguishes clitic forms, e.g. *tsconuents* is tagged as representing a combination of *te dat convent* ("at that convent") and receives the tag Adp()+Art(def,forms)+N(sing,forms). In the corpus, we encounter 16,099 clitics, showing 436 different tag combinations.[2]

Each token is also associated with a lemma. In general, this is the modern form of the lemma, e.g. *si* → *zij* ("she"). When no modern form exists, a pseudo-modern form was created, e.g. *eeuwelike* → *eeuwelijk* ("eternal"), *verlyen* → *verlijden* ("confess"), *manesse* → *manis* ("summons"). For proper names, a normalized form is used, based on the list given by Van der Schaar (1992). For clitic forms, the same

[2] The tagset used here is very similar to the Decoy tagset (Van Eynde 2005), which is the current de facto standard for modern Dutch, which is only to be expected as both tagsets are in principle based on the Dutch grammatical tradition. However, there are also differences. Obviously, the whole clitic apparatus is absent from the Decoy tagset. Another clear difference in the tags is the fact that we do not attempt to mark case, but merely the form of the word ending. Other differences will mostly be in the application of the tags rather than the tags themselves, as some constructs were still in flux in the fourteenth century, e.g. the development of articles from demonstrative pronouns which we will address in footnote 9.

Table 3 Main part-of-speech tags and their observed frequencies in the CRM

POS	Long name	Observed by itself	Observed with clitic	Total
N	Noun	225,850	3,302	229,152
Adp	Adposition	92,401	8,622	101,023
V	Verb	95,083	1,446	96,529
Pron	Pronoun	75,740	3,805	79,545
Conj	Conjunction	71,773	985	72,758
Art	Article	46,301	11,821	58,122
Adj	Adjective	51,840	1,185	53,025
Num	Numeral	33,514	231	33,745
Adv	Adverb	30,967	309	31,276
PronAdv	Pronominal adverb	15,132	557	15,689
Misc	Miscellaneous (e.g. Latin words)	2,727	4	2,731
Punc	Punctuation	51,033		51,033

strategy is followed as for tags, and the clitic form is assigned a lemma combination, e.g. the abovementioned *tsconuents* is assigned lemma *te+dat+convent*.[3] In the corpus, we find 12,510 simple lemmas and 853 clitic combinations.

3.2 System pipeline

Our tagger-lemmatizer system is built using the pipeline architecture that is rather standard for such systems. First there is a tokenization module, which has the task to recognize the tokens within the character stream. How this is done is described in Sect. 3.3. The tokenization module does not perform sentence separation; seeing the almost total lack of systematic punctuation connected to sentence boundaries, we do not distinguish sentences but simply process the whole charter at once. After the tokens have been proposed, each token is associated with a list of potential TLPs, as described in Sect. 3.4. Each potential TLP is also assigned a number indicating its likelihood. After this, a disambiguation module, described in Sect. 3.5, combines the lexical likelihoods with the contextual likelihoods for each token and assigns an overall likelihood to each TLP.

A distinguishing feature of our system in relation to other tagging systems, however, is that it is designed to do two more passes through the whole pipeline. After a first pass, it goes back to tokenization and re-estimates the token recognition, now also using the information about the likelihoods of the TLPs provided by the first passage (Sect. 3.3). In these first two passes, the system uses its standard lexicon, which contains spelling variants that have been observed in the training data or have been extrapolated from the training data. After the second pass, this already expanded lexicon is expanded yet again by adding best guesses for all

[3] In the tagger-lemmatizer, such clitics are viewed as atomic units and are not split in any way for processing.

unknown word forms in the tagged text, using the contextual TLP likelihood in all their observed occurrences (Sect. 3.4). As can be expected, these additional passes increase processing time, especially the lexicon re-expansion. We therefore evaluate the effect of these additions in Sects. 5.2 (re-estimating tokens) and 5.3 (re-expanding lexicon).

3.3 Tokenization module

The tokenization module attempts to identify the tokens in the text. Punctuation tokens are split off from any word tokens they might be attached to. We will not model punctuation explicitly, but as it does appear to represent some kind of structure (Gerritsen 1990), we do keep punctuation tokens in our data so that they can aid in contextual disambiguation.

The word tokens are identified with a pure machine learning classification approach, using two different models. The basic model only uses character level information about a potential token separation point and can be used on unprocessed text. The re-estimation model also uses information about the probable tags and lemmas next to potential separation points and can therefore only be used after an initial round of tagging and lemmatization. For both models, we use the WPDV machine learning system (van Halteren 2000a), as it has two special properties which allow us better control over the learning process. First of all, it allows set-valued features, i.e. the same feature can have multiple values in the same case (Cohen 1996).[4] Secondly, the system allows that weights are attached in the input to specific feature name-feature value combinations inside each individual case, which will be used in addition to the weights that the system has determined on the basis of the training data per feature (or rather combination of features). We will show below how these properties are used in our tokenization models.

In order to decide which tokens there are in the text, we have to determine for each position between two non-spacing characters (which might or might not be separated in the manuscript by white-space) whether this position separates two tokens or whether it is inside a token.[5] The basic model attempts to do this purely on the basis of the context in terms of character strings. As an example, take the string *ende op gedregen haren* ("and transferred (to) their (…)"), where we have to determine whether the space between *op* and *gedregen* separates two tokens or not. In this string, it does not and *opgedregen* should be considered as a single token. For WPDV, we prepare three feature names, `ap` (actually present), `cl` (left context), and `cr` (right context). The first indicates whether or not the manuscript actually shows

[4] In machine learning classification, data sets consist of *cases*, each of which belongs to a *class*, which is to be estimated by the classifier, and furthermore contains a number of *features*, representations of properties that might be useful for the classifier in its task. E.g. in POS-tagging, the class would be a part of speech and the features would include the current token, its potential parts of speech and the part of speech of the previous token. In most classification systems, each feature is assigned a name or placed in a specific column and can contain only one value.

[5] Note that the hyphen may, already in fourteenth century Dutch, indicate a word break over a line boundary and is therefore ambiguous between spacing and non-spacing character. We will not go into in detail about this.

separation, which in this string it does.[6] The second and third only differ in direction, but are otherwise the same. Both are set-valued, including values for distance to the next spacing (e.g. `cl=#L#2` and `cr=#L#8`), full string up to the next spacing (e.g. `cl=#W#op` and `cr=#W#gedregen`), and 1–5 adjacent non-spacing characters, but adding markers for spacing (e.g. for left context `cl=#1#p`, `cl=#2#op`, `cl=#3#e+op`, `cl=#4#de+op`, `cl=#5#nde+op`). For the various lengths, we add weights of 2^{length}, and 16 for the full string up to the spacing. In this way, we indicate that matching on more context should be weighted stronger. WPDV does not only consider individual features, but also feature combinations and will therefore consider e.g. combinations of `ap` and `cr`; however, only features having different names are being combined, so not e.g. different values for `cr`. We can also specify frequency thresholds. We have decided to use only feature combinations that occur at least 3 times in the training material, which leads to about 13 million active feature combinations used for the classification. Our choice of threshold was mostly pragmatic, as lowering it to 2 would lead to a data model that is larger than the working memory of our 4 Gb machines, and we did not measure to which degree this choice of threshold has influenced the accuracy of the process.

For the re-estimation model, only those potential separation points where the basic model yielded a best probability of less than 0.9 are being considered. For these, we now replace the character-level features by token-level features, based on the token and tag-lemma information suggested by the previous pass through the pipeline. The context between the potential separation point and the next token separation according to the previous run is described by two features `fl` (focus to the left) and `fr` (focus to the right). Each of these is set-valued, including various length substrings, again weighting more extensive substrings with higher weights. Apart from these, we use the feature `cl` (left context) and `cr` (right context), each set-valued, listing all possible uni-, bi- and trigrams of tags and/or lemmas, weighted with their probabilities as provided by the previous run. Finally, we again use the feature `ap` to indicate whether or not there is actual spacing in the manuscript. Building the model on only those cases where the basic model is not confident allows us to set the frequency threshold to only 2 for feature combinations in the re-estimation model. This leads to about 9 million active feature combinations taken into account by the system. The effect of this re-estimation is described in Sect. 5.2.

3.4 Potential tag-lemma pair proposal module

The next step in the process is the suggestion of TLPs that could be associated with each token. Here too, we start by following standard techniques. Known word forms are simply looked up in a lexicon, the difference with most other systems being that we have expanded our lexicon so that it also contains forms which have not been observed in the training material but can be expected on the basis of observed spelling variation (for details see Sect. 4). Each suggested TLP is also given a

[6] As explained in Sect. 2.2, the transcribers of the corpus have sometimes reinterpreted the orthography, which means that we have two opinions on token separation. We use the transcribers' opinion as the target of the classification, but use the actual tokenization in the manuscript as a feature.

numerical likelihood indication, which can be used by the contextual disambiguation module (Sect. 3.5).

Unknown word forms are provided with a tag by way of a machine learning algorithm, as are word forms which have a virtual frequency (see Sect. 4.4) of less than 10 in the lexicon. In the latter case, the newly suggested tags are merged with the ones derived from the lexicon, using a weighting scheme which gives less value to the machine learner suggestions as the virtual frequency in the lexicon increases. The TLP's suggested by this algorithm do not contain a proper lemma; instead, the lemma position in the TLP is filled with a marker (currently "???") to indicate this. We again use the WPDV system as machine learner. We learn a model on the basis of all open class TLPs. The features for this model are exemplified in Fig. 5. A standard feature for such models is the word-final string (`Ie`), again set-valued with lengths 1–6 and weighted with 2^{length} (`P1` to `P32`). The length of the word is also taken into account, using length bands 1, 2–3, 4–5, 6–10, 11–20 and 21+ (`Ilen`). The remaining features are connected to Dutch morphology. First of all, Dutch has separable verbs where a particle can be split off for some finite forms but is attached at the beginning of the word in others. On the basis of the split off occurrences, we are able to learn an inventory of particles, so that we can check in unknown words whether or not they start with a particle. Secondly, past participle formation in Dutch is generally done with prefixation of *ge–* (or spelling variants), but with an infix *–ge–* placed after the particle for separable verbs. As there are also some other recognizable prefixes, we add features for the first 1–3 characters of the word (`Ib`) and the first 2–3 characters following a recognized particle (`Ibp`). Both these features are again set-valued and weighted for length. We set a frequency threshold of 2, leading to about 400,000 active feature combinations.

After a full pass through the tagging pipeline, we have suggestions for all unknown word forms in their context(s). At this point, our system allows a further lexicon expansion. Every TLP that is suggested by the unknown word form algorithm, and that is given a probability higher than 0.1, is examined. First, an exception rule is applied, for word forms which are tagged as numbers and look like Roman numerals, as for these a similarly spelled word may well represent a different number. For all other word forms, the system finds the closest word form in the lexicon which can be associated with the assigned tag. This lemma is then added to the lexicon, with an estimated probability dependent on the sum of all estimated probabilities in the texts currently being tagged. Furthermore, the probabilities for known word forms are also updated in the lexicon dependent on their estimated contextual probabilities in the texts. The resulting updated lexicon is then used in a new pass through the tagging pipeline. The effects of this re-expansion are described in Sect. 5.3.

```
Iprt=1 Ilen=6-10 Ib=w P1 Ib=wt P2 Ibp=gh P2 Ibp=ghe P4
Ie=n P1 Ie=en P2 Ie=men P4 Ie=omen P8 Ie=nomen P16 Ie=enomen P32
```

Fig. 5 The case for the word form *wtghenomen* ("taken out") for the machine learning module providing potential tags for unknown word forms

3.5 Contextual disambiguation module

After all tokens have been associated with a list of potential TLPs and their likelihoods, the system adjusts these likelihoods on the basis of the context. For contextual disambiguation, we did not have to develop new methods, but could reuse existing tools. As we knew that better results can be achieved with combinations of systems rather than individual ones (van Halteren et al. 2001), we opted to combine our own tagger generator system (WPDVit) with two other popular systems: TnT and SVMTool. We use all systems in several modes, and combine their outputs into a final result.

SVMtool (Giménez and Márquez 2004) is one of the best tagger generators currently available, using Support Vector Machine technology. It allows for a number of different strategies, of which we use modes 0 (one-pass, default), 1 (two-pass), 2 (robust against unknown words) and 4 (very robust against unknown words), leaving all other settings at their default value. In addition we apply each mode going both left-to-right and right-to-left through the text, leading to a total of 8 SVMtool suggestions. As SVMtool yields only tags with likelihoods and no lemmas, the likelihoods for the suggestions are merged into those for all TLPs with the corresponding tags.

TnT (Tags 'n Trigrams; Brants 2000) is one of the most used HMM-based tagger generators. Seeing that HMM tagging has proved useful in the past and differs considerably from SVM-based tagging, we judged that including it in our combination would be fruitful. We use TnT with its default settings, but do again run it both left-to-right and right-to-left, leading to 2 TnT suggestions. TnT too only yields tags and no lemmas, and output is used in the same way as that from SVMtool.[7]

Our final addition to the combination is an experimental system using the WDPV machine learning system, called the WPDV iterative tagger (WPDVit). It has an architecture similar to an earlier WPDV-based tagger (van Halteren 2000b), but is expanded in that the contextual disambiguation is used iteratively. As already mentioned above, the WPDV system allows each feature in a specific case to be weighted individually by the user. In WPDVit, we use this property to weight each TLP in the context with its likelihood as yielded in the previous iteration, being the lexical likelihoods on the first iteration and contextual likelihoods in later iterations. E.g. if we are disambiguating *geet* in *geet telken geldens tide* ("goes at each pay period") and the first tagging suggested a probability of 0.54 for *telken* being a clitic form with tag Adp()+Num(indef,formn) and lemma te+elk and a probability of 0.81 for *geldens* having tag V(participle,pres,forms) and lemma gelden, then we would add a feature for the right context being Adp()+Num(indef,formn)++ V(participle, pres,forms) with a weight of $0.54 \times 0.81 = 0.44$, which helps identify *geet* as the third person singular finite form of the verb *gaan* ("go") rather than its closest competitor, a form of the proper noun *Gerhard*. Pilot experiments showed that, for the fourteenth century data, iteration was only useful at the early stages and we kept only the outputs of the first and second iteration. The direction in the tagging

[7] Both SVMtool and TnT work on the original word forms and do not make use of the TLPs suggested by our own system.

process is not relevant for WPDVit. SVMtool and TnT proceed through the sentence word by word and use the choices made for each token in the disambiguation of the next token, which means that the resulting likelihoods will depend on the direction in which the sentence is tagged. WPDVit on the other hand considers the undisambiguated tags of the context in both directions and is therefore not influenced by a chosen tagging direction. As a result, there are only two WPDVit TLP suggestions, one for the output of each iteration.

After running the individual disambiguation systems, we have 14 suggestions as to the likelihood of each TLP. In principle, one could now train a new classifier which uses the 14 opinions as features for its classification (*stacked classification*). However, this would be rather computationally expensive. For the current system, we have therefore decided to just give each opinion equal weight and simply average the suggested likelihoods. To reduce the number of TLPs that are taken into account, though, which is especially important for modules like WPDVit, we remove all TLPs with a likelihood of less than 0.0001 at specific points in the pipeline.

4 Lexicon expansion

Given the enormous amount of spelling variation, it is unlikely that all spelling variants will be observed in training material comprising a mere 700,000 words. We therefore expand our lexicon by including variants of observed word forms which are likely given the also observed spelling variation. This process is completely data driven and not supported by manually created rule systems such as e.g. VARD's modified SoundEx and letter replacement rules (Baron and Rayson 2008). Although especially phonetic modeling appears attractive, we do not expect it very likely to be successful, as the spelling variation in our texts is mostly linked to the sometimes extreme dialectal pronunciation variation. Furthermore, the exact pronunciation of most dialects is not known, which seriously hampers the desired modeling. We first determine the cost of variation at the character level (Sect. 4.1), which is then used to align all word forms per TLP into variation grids showing the variation options for that TLP (Sect. 4.2). We then examine all variation grids in order to generalize to a set of likely variation rules which are more widely applicable (Sect. 4.3). Finally, we generate expected variants from known word forms by applying what we learned about variation for those forms and in general (Sect. 4.4).

4.1 Character variation cost

For the alignment of the various word forms for the same TLP, we build upon the Levenshtein (1965) algorithm with differentiated edit operation costs. As a result, we need to estimate costs for all possible character insertions, deletions and substitutions. Since we are working with two observed forms rather than a standard form and a variant, every word form comparison is symmetric, and the costs for insertion and deletion of the same character are equal. Furthermore, given the nature

of our data, we also derive separate costs for doubling/undoubling (which are cheaper than the corresponding insertion/deletion). Finally, as *ch* and *ph* at first sight appeared to act as single characters in variation, we transformed them both internally to a single character (viz. < and >). Later observation showed that *ch* turned out to alternate more with *c* than with the phonologically related *g*, but this was unproblematic as h-deletion is common elsewhere as well. For *ph*, we saw the opposite, in that alternation with *f* is more common than with *p*.

In order to estimate operation costs, we examine all word form pairs for the same TLP which are only one edit operation apart. For all observed edit operations, we start with a cost of one. Then, for every word pair where the operation is observed, the cost is reduced by a constant C times the current cost. Note that the optimal C is obviously corpus dependent; in the work reported here, C is set to 0.003. For future work, we are considering to switch to a more corpus independent method, e.g. Mutual Information-based (Wieling et al. 2009).

We show examples of the resulting costs in Table 4. Vowels are clearly strongly subject to variation, as is the presence of the h. Consonants are more stable, but here too we find pairs such as d-t which are almost completely interchangeable. At the higher cost end, we find very unlikely changes, such as insertion of b and substitution of b into z, which apparently do sometimes occur in our data. Their cost remains at almost 1.

All unobserved insertion/deletion and doubling/undoubling operations are given cost 1. Unobserved substitutions are set to 2 if the two characters are compatible, i.e. both are vowels, both are consonants or both are digits. For measuring compatibility, the characters *v* and *w* are counted as both vowel and consonant since they are freely interchanged with *u* and *uu*, all *u*'s themselves are transformed at the start of processing to *v*, and the character *j* is system-internally treated as *i*, always a vowel. If the two characters are not compatible, substitution is blocked with a cost of 1,000,000.

4.2 Multi-variant alignment

We now proceed to building a variation grid for all word forms observed with the same TLP (see example in Table 5). For each TLP, we first find the two most similar forms, according to the Levenshtein algorithm with edit costs as described above, and align them. From then on, as long as there are more forms to be merged into the grid, we each time identify which form is most similar to the whole current grid. When comparing a character in the new form with a column of the grid, the edit cost is primarily determined by the row yielding the lowest cost, but slightly adjusted by the row yielding the highest cost.[8] If at any time, the overall cost for the new form is higher than 2.25, a number chosen as appropriate on the basis of human inspection of the grid formation process, we assume that the TLP in question actually has more than one different orthogonal form and start a new grid. This happens for about

[8] There is only space here for a general description and not for the various heuristics employed, such as penalization of an insertion which hinders a substitution of neighboring characters. For further detail, please contact the author.

Table 4 Selected data-driven edit operation costs for determining Levenshtein distance

	Insertion/ deletion	Doubling/ undoubling		Substitution
e	.017	.004	e ↔ i	.050
h	.085	.085	i ↔ y	.086
n	.459	.339	d ↔ t	.235
r	.769	.661	c ↔ k	.598
m	.956	.849	b ↔ p	.969
b	.979	.949	b ↔ z	.997

0.7 % of the TLPs, most often names, e.g. *Ambrosius* is found both as *ambrosivs* and as *brosikens*.

4.3 Rule derivation

After we have built the grids, we can examine which edit operations occur in the same character context in more TLPs. For each edit operation in each TLP grid, including the identity operation, we generate all contexts with 1–3 characters left and right and list them together with the operation in question and the observed frequencies of the source and target form, resulting in a kind of protorules (now no longer symmetric as to insertion/deletion). We continue with those protorules which are observed at least 10 times and assign a probability for the given operation in the given context by dividing the frequency of the source form by the frequency of the target form, thus merging the token-dependent protorules into probabilistic character-context-dependent rules. If the item is observed for less than 25 TLPs, this probability is reduced. We keep all rules with a probability higher than 0.05, but filter out those that are redundant. i.e. those where a more general context provides the same or higher probability. Table 6 exemplifies the output by showing the most probable operations of the various types (for a specific training set). The operations shown all have probabilities higher than 0.5 and transform a less likely variant into a more likely one; the reverse operation may also be present but with much lower probability, e.g. s__c__o → < has probability 0.11.

4.4 Variant generation

After deriving generalized rules, we turn back to the grids to expand them with expected but unobserved variant forms. We always start with a form which is actually observed, and then allow a number of operations on it which are based on the current TLP grid and which number at most the square root of the length of the form. These operations consist of choosing a column in the grid and replacing the character that the source form has there (possibly none) by a different character in the same column in another row (again possibly none). For each change, the probability of the target word form is adjusted by the probability of the edit operation. After the grid-based operations, we allow the application of a single general rule, but now reduce the probability not only by multiplying with the

Table 5 Multiple spelling variant alignment grid, for the word *geboorte* ("birth"), with the observed count in the rightmost column

G	h	e	b	o	e	r	t	e	14
G		e	b	v	e	r	d	e	11
			b	o	e	r	t	e	7
G	h	e	b	o		r	t	e	3
G		e	b	o		r	t	e	3
G		e	b	v		r	t	e	2
G	h	e	b	v	e	r	t	e	1
G	h	e	b	v	e	r	d	e	1
G		e	b	o	i	r	d	e	1
			b	o		r	t	e	1

Table 6 Most probable generalized spelling variation rules

Substitution		Deletion		Insertion	
#s__ < __e → c	.73	eg__h__#	.90	#g__ __el → h	.76
t__z__# → s	.71	f__f__#	.87	#g__ __es → h	.75
an__d__# → t	.70	t__h_en	.85	g__ __el → h	.71
l__d__# → t	.70	en__n__e#	.78	dag__ __e → h	.70
s__ < __o → c	.68	ike__n__#	.78	#g__ __e → h	.66

The notation should be read as follows. Each rule has a left hand side and a right hand side, separated by an arrow. The left hand side consists of a left context, a focus (possibly empty) and a right context, separated by double underscores. A rule shows that the focus may be replaced by the right hand side when in the presence of both left and right context, with the probability indicated. As to the characters within the rules, remember that < represents the character combination ch; and the # indicates the beginning or end of the token

probability of the rule but also by an additional multiplication with 0.4 for using an operation not observed for this TLP. We then sum the probabilities of each target form over all its derivations, and where present also add in the originally observed count for this form. We now include those target forms in the lexicon which have a final probability of 0.0001 and which do not include character trigrams that were not observed in the training material. In order to be able to provide a frequency count to the lexicon, we transform the probabilities to virtual frequencies in such a way that the frequencies for all spelling variants for a TLP add up to the actual total frequency of this TLP. As an example, Table 7 shows some observed and virtual frequencies of the noun *geboorte* ("birth"), for which the original grid is shown in Table 5.

4.5 Effects of expansion on the lexicon

The lexicon expansion as explained above increases the number of lexicon entries (i.e. word form—TLP pairs) from about 50,000 to about 1,250,000. The number of

TLPs obviously remains the same, at about 23,000, but the number of distinct word forms goes up from about 40,000 to about 1,100,000.

If we look at the number of word forms per TLP (Fig. 6), which has an average of about 2 in the original lexicon and about 50 in the expanded one, we see that the number of expanded forms does not depend primarily on the number of forms observed in the corpus. In a typical run of the system, i.e. fold3 from the cross-validation described below, we see that the TLP with the most variation is the plural form of the noun *heiligegeestmeester* ("administrator of the Poor Relief Fund"), which goes from 11 observed variant forms to 8,437 variant forms in the expanded lexicon.

The expansion of the lexicon of course also increases the ambiguity of the word forms. However, as we see in Fig. 7, this increase is limited. Again using fold3, 980,606 (91 %) of the 1,076,916 newly introduced variants are only associated with one TLP and 28,400 (82 %) of the 34,511 originally unambiguous word forms remain unambiguous. The average ambiguity goes up from 1.22 TLPs to 1.72 TLPs per word form, a factor of 1.36. We will see in Sect. 5.3 whether this increase in ambiguity can be dealt with to such a degree that lexicon expansion is beneficial rather than detrimental.

5 Experimental results

In order to test our system, we have performed a tenfold cross-validation on the CRM, each time taking 90 % of the charters as training material and 10 % as test material. Manually determined settings, such as the constant C mentioned in Sect. 4.1, were kept constant, but all models were trained purely on the training material for a specific fold. The system output was evaluated against the original annotation in the corpus, accepting this as a gold standard for annotation of fourteenth century Dutch charters. We first give overall results on system performance (Sect. 5.1) and then continue with a focused evaluation of the effects of retokenization (Sect. 5.2) and lexicon expansion (Sect. 5.3).

5.1 Overall results

The results for the tagger in its full form over the whole corpus are shown in Table 8. With scores approaching 95 % for the 1-best output, the system is not yet

Table 7 Selected forms for the noun *geboorte* ("birth") in the expanded lexicon

Token	Observed	Virtual frequency
gheboerte	14	7.73
gebverte	2	1.94
ghebvrde	0	0.41
boerde	0	0.14
heboeirte	0	0.0005

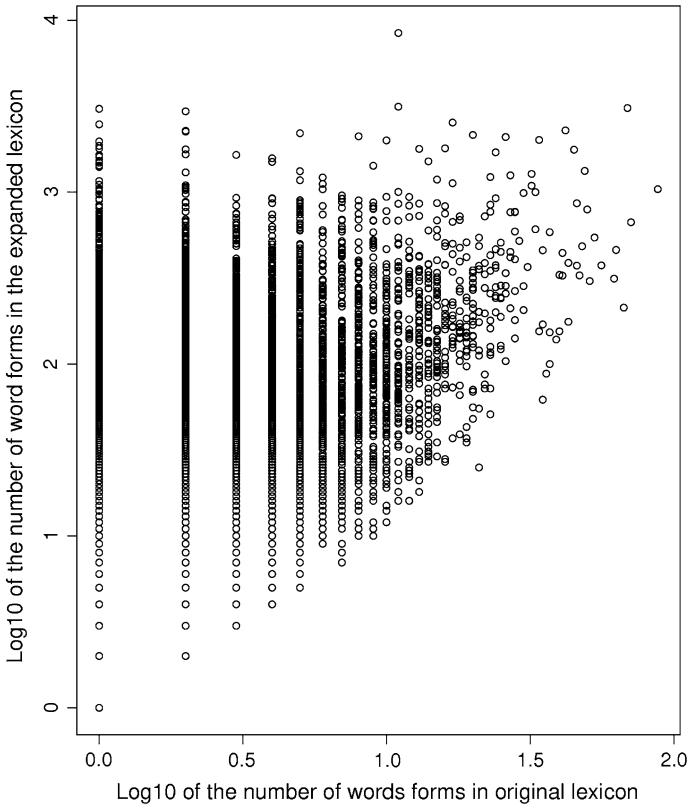


Fig. 6 Number of word forms per TLP in the expanded lexicon as a function of the number of word forms in the original lexicon (both in log scale)

near the results reported for taggers of modern languages, but this is only to be expected. Given the nature of the material, the success rate is actually quite high. Most of the errors for the 1-best output are disambiguation errors, but both lexicon (especially where it concerns lemmas) and tokenization also provide room for improvement. As can be expected, there is also a substantial difference in quality between the charters. There are even charters (up to 172 tokens) that are tagged and lemmatized completely correctly. At the other end of the spectrum, we find charters in which only around 80 % of the tags (and/or lemmas) are correct. Sometimes, such low quality is caused by the nature of the charter, e.g. having many unreadable words or addressing a subject matter different from that of the (administrative) bulk of the material, such as a furniture inventory.

Error analysis has shown that many of the errors can be easily explained, but not easily repaired. An example at the tag level is the suffix *–n*, e.g. in *joncvrouwen* ("damsel/damsels"). Our tagset does not indicate the function of a suffix, but only marks the form, here the *–n*. However, for e.g. feminine nouns, the *–n* suffix is ambiguous between singular (genitive or dative) and plural, and the singular-plural distinction is made in the tagset. A similar problem is presented by *–s*, which is

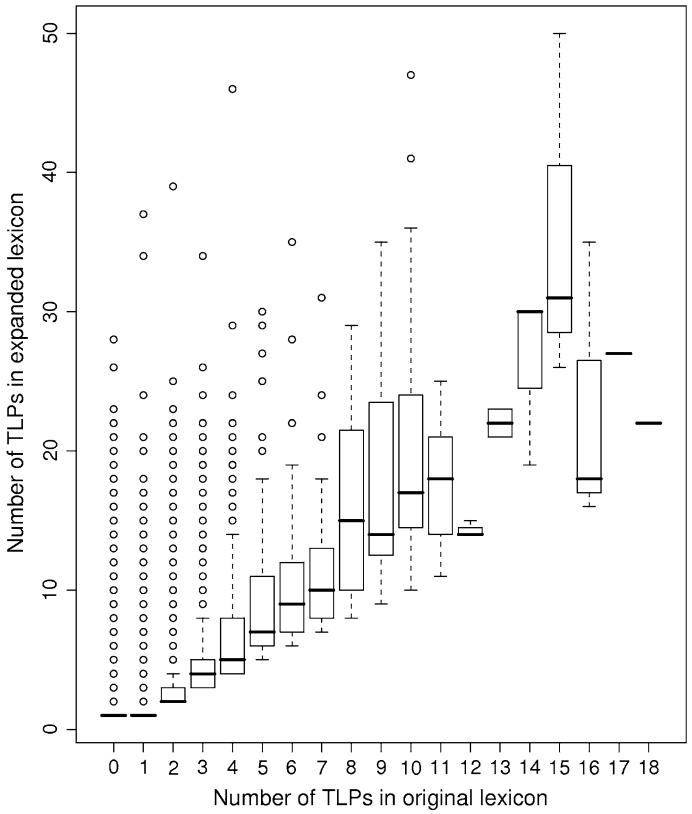


Fig. 7 Box plot of word form ambiguity in the expanded lexicon as a function of the ambiguity in the original

ambiguous between singular genitive (masculine or neutral) and plural. As correct disambiguation depends on actually understanding the meaning of the text, the system is distinctly challenged, which explains the large number of singular-plural confusions with these two endings: 3,815 (11 % of all tag errors). At the lemma level, the most common confusions are between *de* ("the") and *dat* ("that") with 3,599 errors (10.3 % of all lemma errors)[9] and between the lemmas *haar* ("her") and *hun* ("their") which correspond to the singular and plural interpretations of forms of *hare* with 1,617 errors (4.6 %). In both cases, a deeper understanding of the text is needed in order to make the desired distinction.

On the other hand, our examination also shows that quite a few of the discrepancies between system output and corpus are in fact caused by errors in the

[9] The Modern Dutch definite articles originated from the unstressed demonstrative pronouns. In Middle Dutch, the latter can still occur both as demonstrative pronouns and as articles. They have been annotated as articles during the manual annotation the CRM unless the context clearly shows that they are meant to be read as demonstrative pronouns. As the assigned lemmas are the modern ones, the choice entails a difference in lemma.

Table 8 Accuracy measurements for the system as a whole

	Tag (%)	Lemma (%)
Token recognized	99.20	99.20
Gold standard annotation proposed	98.75	97.28
Gold standard annotation selected	94.97	94.88

corpus annotation, about 30 % of the tag errors and about 10 % of the lemma errors, which means that the quality might actually be higher than our measurements indicate. Furthermore, about 30 % of the lemma errors concern proper names. Here the system suffers from the predictable absence of many proper names from the test manuscripts in the training manuscripts, so that 2,960 tokens tagged as proper name do not get a lemma, accounting for 8.5 % of the lemma errors. But it also suffers from inconsistency in the choice of lemmas for proper names. As an extreme example, of the 1,119 occurrences of the name Gerard (Table 2), only 160 are actually lemmatized as *Gerard*, the others being marked as *Gerhard* (634) or *Gerrit* (325). For this name, the indexing in Van der Schaar (1992) could be clearer, leading to confusion with the annotators. The system copies some of these errors, but the remaining discrepancies account for 419 lemma errors (1.2 %).

5.2 Token separation results

As described in Sect. 3.3, we do not follow the token separation as observed in the actual charters, but attempt to provide our own tokenization, using at first a simple character-based model and in a later phase a partly word-based model. The question, then, is whether these tokenization attempts are worth the effort. We measured the quality of the output after the first and second pass through the system pipeline, i.e. the passes using the expanded and not the re-expanded lexicon. In addition, we ran a single pass with the tokenization module switched off, and hence using tokenization as given by the layout in the charter. From Table 9 it is clear that the initial tokenization is essential, as it improves both tag and lemma accuracy by about 3 %. However, the later word-based re-estimate yields an improvement of a mere 0.1 %, and would seem to be superfluous.

5.3 Lexicon expansion results

The same question as in 5.2 can be asked about lexicon expansion and re-expansion. Here we measured the quality by counting the errors after the second and third pass through the pipeline, i.e. the passes using re-estimated tokenization, and in addition after two passes with the expanded lexicon replaced by a lexicon which only contains word forms observed in the training material. The results are given in Table 10. We see that the importance of the lexicon expansion is fully dependent on the goal of the annotation. If only tags are needed, the expansions hardly bring profit. It would seem that the machine learning model for tagging unknown words is

Table 9 Accuracy for various levels of retokenization

	Token (%)	Tag (%)	Lemma (%)
As written	96.05	91.73	90.90
First estimate	99.11	94.85	93.88
Re-estimate	99.20	94.94	93.96

Table 10 Accuracy for various levels of lexicon expansion

	Gold standard tag proposed (%)	Gold standard tag selected (%)	Gold standard lemma proposed (%)	Gold standard lemma selected (%)
Known forms only	98.66	94.91	95.18	93.11
Expanded lexicon	98.79	94.94	96.23	93.96
With test token adaptation	98.75	94.97	97.28	94.88

well equipped to provide the proper tags for spelling variants. For lemmas, however, each lexicon expansion yields an accuracy improvement of almost 1 %. And although 1 % does not sound impressive, one should consider that the step from 93.11 to 93.96 % is a 12.3 % reduction of errors and from 93.96 to 94.88 % a reduction of 15.2 %. In other words, both expansions are certainly worth the effort.

6 Related work

The spelling of late Medieval Dutch is currently being studied actively by several researchers. The general trends in spelling and phonological variation were already reported earlier (e.g. Berteloot 1984), but the availability of machine readable corpora (see e.g. Mooijaart (1992), who used the Corpus Gysseling) and especially the development of machine learning techniques allow us to enhance these reports with much more detailed information.

Mike Kestemont and colleagues at the University of Antwerp have been looking into spelling variation in thirteenth century Dutch literary texts (taken from the Corpus Gysseling; Gysseling 1977–1981), combining Levenshtein-based distances with k-nn techniques (Kestemont et al. 2010). Although this work shows many similarities with ours, there are also many dissimilarities, too many to make the results directly comparable. Instead, we are planning to exchange test material in the near future and apply each method to each data set. Martin Reynaert of Tilburg University has been working on spelling correction in general, using statistics on common forms and apparent spelling variants in large corpora (Reynaert 2005). Although his methods do work on historical material, he has not yet attempted to process Dutch texts from before the eighteenth century (Reynaert, personal communication). The Institute for Dutch Lexicography (INL) has also worked on a tool and a procedure to derive spelling variation rules from example material and a

spelling-variation-aware lemmatizer in the context of the IMPACT project (Depuydt and de Does 2009; Depuydt, personal communication). However, the lexicon that they developed covers the sixteenth century and later, this being the language period covered by the Woordenboek der Nederlandse Taal (WNT; de Vries et al. 1956).

For German, where the situation was very similar in Middle High German (eleventh–fourteenth centuries) to that described here for Dutch, there is also active research on automatic canonicalization. Examples are the work of Pilz et al. (2006, 2008), Jurish (2010), Schnurrenberger (2010), Bollmann et al. (2011) and Bollmann (2012). Apart from Jurish, who also investigates phonologically based rewrite rules (both manually constructed and automatically derived from a lexicon), the general approach is using similarity measures to find matching known forms. However, only Schurrenberger clearly focuses on text as old as ours. The others all target (often much) younger texts. Given the differences in language, period and text type, we will not go into a detailed comparison of the results. Automatic tagging of Middle High German is also being attempted, but this work is still in its infancy (Dipper 2011; Hinrichs and Zastrow 2012). For English, the language for which NLP work is usually the most advanced, spelling variant research is mostly being done on younger texts (Archer et al. 2003; Pilz et al. 2008; Baron et al. 2011). For older material, spelling variation does not appear to be addressed at all, e.g. the Penn-Helsinki Corpus of Middle English (2nd Edition) is annotated extensively, but (in the tradition of Penn-parsed corpora) not for lemma (Santorini 2010).

Further related efforts include attempts to recognize the region, period or individual author/scribe of a text. Although the current paper attempts to deal with the spelling problem in a generalized fashion, it is very likely that spelling remains much more constant if we only consider texts from a single region, period or author/scribe, which means that specialized spelling variation models could lead to better results. Now the scribe can mostly be recognized, as long as the original manuscript is still available, by way of an examination of the handwriting (e.g. Burgers 1995; Dijkhof 1997; Van Camp 2011). In another paper (van Halteren and Rem, to be submitted), we show that the handwriting-based classification can be reproduced almost perfectly, when sufficient training material is present, on the basis of spelling. This implies that indeed spelling is much more constant within the writings of a specific person. Region can also be recognized to a large degree on the basis of spelling (Rem 2003; Van Uytvank 2007), but period so far defies high quality recognition (Van Uytvank 2007). However, both region and period do not necessarily have to be determined on the basis of the form of the text, but can often be derived in a different manner, either extra-linguistically or from actual mentions in the text.

There is also extensive current work on orthographic variation outside the historical context, especially in relation to the new media such as SMS (cf. e.g. Tagg 2009) and Twitter (cf. e.g. van Halteren and Oostdijk 2012), and in relation to spelling variation due to the transliteration of proper names (cf. e.g. Hsu and Chen 2010). However, as the nature of such variation tends to differ from that in historical texts, we prefer to refrain from a detailed comparison here.

7 Conclusion and future work

We have implemented a tagger-lemmatizer system for fourteenth century Dutch which is enhanced in relation to general state-of-the-art systems in that it attempts to compensate for the extreme orthographic variation in the target text. The results can be deemed good, reaching an accuracy of around 95 % for both tags and lemmas. We have seen that it is not advisable to use the spacing in the manuscripts, but instead the tokens as proposed by a machine learning model. However, a simple character-based model appears to be sufficient for this task. For lemmatization, our lexicon expansion, based on observed variation and extrapolated spelling variation rules, greatly improved the quality. For tagging, however, a machine learning model for unknown words proved equally capable.

Analyses of the output show several areas where there is room for further improvement. The simplest change from the point of view of the system developer is a further clean-up of the training material. In our analysis, we noticed errors and inconsistencies in the corpus van Reenen-Mulder which negatively influence both training and evaluation. However, the amount of work involved in rechecking the annotation of such a corpus should not underestimated, and implementing such an endeavor is likely to prove difficult. However, the system itself can also be improved. Lemmatization of proper names, e.g. can be improved by importing name lists from other sources, e.g. the Repertorium voor Eigennamen in Middelnederlandse Literaire Teksten (See http://cf.hum.uva.nl/dsp/scriptamanent/remlt/remltindex.htm) which is currently under construction. And tagging problems like the distinction between plural and inflected singular (Sect. 5.1) could possibly be addressed by a rule-based postprocessing module. If rules cannot be formulated, then we should in this case consider simplifying the tagset and no longer attempt to make distinctions which are in principle based on world knowledge.

Although some of our future work will obviously address improving system quality for the current material, we are also very much looking outward. We have already started processing texts from the same period and domain but from other regions, and the next steps will gradually move towards other domains and periods. We hope that such a gradual expansion, coupled with a limited level of focused manual correction, will allow us to make the system useful for a much wider range of historical texts. And once a sufficient amount of text has become available, we can investigate if further improvements are possible by first recognizing what type of text is being provided and subsequently processing this with specialized models for that type of text. Obviously, this strategy is only viable if sufficient material becomes available with corrected annotations, too much material for one group to produce. However, as part of the NWO-Clarin NL project (see http://www.clarin.nl), we have made a streamlined version of the system described in this paper available for general use through the European Clarin infrastructure (http://www.clarin.eu) under the name Adelheid (see http://adelheid.ruhosting.nl for details about its use). This means that more researchers can now start to process and correct their own texts and thus help realize a proper tagging and lemmatization of all historical Dutch in the near future.

References

Archer, D., McEnery, T., Rayson, P., & Hardie, A. (2003). Developing an automated semantic analysis system for Early Modern English. In *Proceedings of the Corpus Linguistics 2003 Conference. UCREL Technical Paper Number 16. UCREL, Lancaster University.*

Baron, A., & Rayson, P. (2008). VARD 2: A tool for dealing with spelling variation in historical corpora. In *Proceedings of the Postgraduate Conference in Corpus Linguistics*. Birmingham, UK: Aston University, May 22, 2008.

Baron, A., Rayson, P., & Archer, D. (2009). Word frequency and key word statistics in corpus linguistics. *Anglistik, 20*(1), 41–67.

Baron, A., Rayson, P., & Archer, D. (2011). Quantifying early modern English spelling variation: Change over time and genre. Presented at Conference on New Methods in Historical Corpora, Manchester, United Kingdom, 29–30 April 2011.

Berteloot, A. (1984). *Bijdrage tot een klankatlas van het dertiende-eeuwse Middelnederlands.* Koninklijke Academie voor Nederlandse Taal- en Letterkunde, Gent.

Bollmann, M. (2012). (Semi-) Automatic normalization of historical texts using distance measures and the norma tool. In *Proceedings of the Second Workshop on Annotation of Corpora for Research in the Humanities (ACRH-2).* Lisbon, Portugal, November 29, 2012.

Bollmann, M., Petran, F., & Dipper, S. (2011). Rule-based normalization of historical texts. In *Proceedings of the RANLP Workshop on Language Technologies for Digital Humanities and Cultural Heritage*. Hissar, Bulgaria, pp. 34–42.

Brants, T. (2000). TnT—a statistical part-of-speech tagger. In *Proceedings of the Sixth Applied Natural Language Processing Conference.* ANLP-2000, pp. 224–231.

Burgers, J. W. J. (1995). *De paleografie van de documentaire bronnen in Holland en Zeeland in de dertiende eeuw*, *3 dln*. Leuven: Schrift en Schriftdragers in de Nederlanden in de Middeleeuwen.

Cohen, W. W. (1996). Learning trees and rules with set-valued features. *AAAI/IAAI, 1*, 709–716.

de Vries, M., te Winkel, L. A., et al. (1956). *Woordenboek der Nederlandsche Taal.* Delen I-XXIX.'s-Gravenhage/Leiden etc.: M. Nijhoff/A.W. Sijthoff etc., 1882–1998. Supplement I.'s-Gravenhage/ Leiden etc.: M. Nijhoff/A.W. Sijthoff etc. Supplements parts I-III.'s-Gravenhage: Sdu Uitgevers.

Depuydt, K., & de Does, J. (2009). Computational tools and lexica to improve access to text. In Beijk, E., Colman, L., et al. (Eds.), *Fons Verborum, F. Feestbundel voor Prof. Dr. A.M.F.J. (Fons) Moerdijk, aangeboden door vrienden en collega's bij zijn afscheid van het INL.* Leiden/Amsterdam, pp. 187–199.

Dijkhof, E. C. (1997). *Het oorkondewezen van enige kloosters en steden in Holland en Zeeland*. PhD Thesis. University of Amsterdam.

Dipper, S. (2011). Morphological and part-of-speech tagging of historical language data: A comparison. In *Journal for Language Technology and Computational Linguistics, Special Issue, 26*(2), 25–37. (= *Proceedings of the TLT-Workshop on Annotation of Corpora for Research in the Humanities, 2012*).

Gerritsen, M. (1990). The relationship between punctuation and syntax in Middle Dutch. In Fisiak, J. (Ed.), *Historical Linguistics and Philology. Trends in Linguistics. Studies and Monographs 46*. Berlin: Mouton de Gruyter.

Giménez, J., & Márquez, L. (2004). SVMTool: A general POS tagger generator based on Support Vector Machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04)*. Lisbon, Portugal.

Greidanus, J. (1926). *Beginselen en ontwikkeling van de interpunctie, in't biezonder in de Nederlanden*. Zeist: Vonk & Co.

Gysseling, M. (1977–1981). *Corpus van Middelnederlandse teksten (tot en met het jaar 1300).* M.m.v. Willy Pijnenburg. Nijhoff,'s-Gravenhage.

Hendrickx, I., & Marquilhas, R. (2011). From old texts to modern spellings: An experiment in automatic normalisation. *Journal for Language Technology and Computational Linguistics (JLCL), 26*(2), 65–76.

Hinrichs, E., & Zastrow, T. (2012). Linguistic annotations for a diachronic corpus of German. *Linguistic Issues in Language Technology, 7*(7).

Hsu, C. C., & Chen, C. H. (2010). Mining synonymous transliterations from the World Wide Web. *ACM Transactions on Asian Language Information Processing (TALIP), 9*(1), 1–28.

Huber, O. (1988). Het kodeerprogramma L.I.M.A. In van Reenen-Stein, K. H., van Reenen, P. Th., & Dees, A. (eds.), *Corpus Gebaseerde Woordanalyse (VWF VULET 88/9), Jaarboek 1987-1988*, pp. 61–70.

Huber, O. (1989). *The construction of lexically analysed text corpora on the computer*. Amsterdam: Vrije Universiteit.

Jurish, B. (2010). Comparing canonicalizations of historical German text. In *Proceedings of the 11th Meeting of SIGMORPHON*, Uppsala, Sweden, July 15, 2010.

Kestemont, M., Daelemans, W., & De Pauw, G. (2010). Weight your words—memory based lemmatization for Middle Dutch. *Literary and Linguistic Computing, 25*(3), 287–301.

Левенштейн, В. И. (1965). Двоичные коды с исправлением выпадений, вставок и замещений символов. *Доклады Академий Наук СССР, 163*(4), 845–848. Appeared in English as: Levenshtein, V.I. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady, 10*, 707–710.

Mooijaart, M. A. (1992). *Atlas van Vroegmiddelnederlandse taalvarianten,* Led, Utrecht 1992. PhD Thesis. University of Leiden.

Pilz, T., Ernst-Gerlach, A., Kempken, S., Rayson, P., & Archer, D. (2008). The identification of spelling variants in English and German historical texts: manual or automatic? *Literary and Linguistic Computing, 23*(1), 65–72.

Pilz, T., Luther, W., Ammon, U., & Fuhr, N. (2006). Rule-based search in text databases with nonstandard orthography. *Literary and Linguistic Computing, 21*, 179–186.

Rem, M. (2003). *De taal van de klerken uit de Hollandse grafelijke kanselarij (1300-1340). Naar een lokaliseringsprocedure voor het veertiende-eeuws Middelnederlands*. PhD Thesis, Vrije Universiteit, Amsterdam.

Reynaert, M. (2005). *Text-Induced Spelling Correction*. PhD Thesis. University of Tilburg.

Reynaert, M., Hendrickx, I., & Marquilhas, R. (2012). Historical spelling normalization. A comparison of two statistical methods: TICCL and VARD2. In *Proceedings of the Second Workshop on Annotation of Corpora for Research in the Humanities (ACRH-2)*. Lisbon, Portugal, November 29, 2012.

Santorini, B. (2010). *Annotation manual for the Penn Historical Corpora and the PCEEC*. http://www.ling.upenn.edu/hist-corpora/annotation/index.html.

Schnurrenberger, M. (2010). *Methods for graphemic normalization of unstandardized written language from Middle High German Corpora*. Master thesis, Ruhr University Bochum.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal, 27*(3), 379–423.

Tagg, C. (2009). *A corpus linguistic study of SMS text messaging*. PhD thesis. University of Birmingham.

van Camp, V. (2011). *De oorkonden en de kanselarij van de graven van Henegouwen, Holland en Zeeland.* Schriftelijke communicatie tijdens een personele unie: Henegouwen, 1280-1345, 2 dln. Hilversum: Verloren.

van der Schaar, J. (1992). *Woordenboek van voornamen*. Utrecht: Het Spectrum.

Van Eynde, F. (2005) Part of speech tagging en lemmatisering van het D-Coi corpus. Centrum voor Computerlinguïstiek, Leuven. http://www.ccl.kuleuven.be/Papers/DCOIpos.pdf.

van Halteren, H. (2000a). A default first order family weight determination procedure for WPDV models. In Proceedings of the 2nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning, pp. 119–122.

van Halteren, H. (2000b). The detection of inconsistency in manually tagged text. In Proceedings of the LINC2000.

van Halteren, H., Daelemans, W., & Zavrel, J. (2001). Improving accuracy in word class tagging through the combination of machine learning systems. *Computational Linguistics, 27*(2), 199–229.

van Halteren, H., & Oostdijk, N. (2012). Towards identifying normal forms for various word form spellings on Twitter. *CLIN Journal, 2*, 2–22.

Van Uytvank, D. (2007). Orthography-based dating and localisation of Middle Dutch charters. MA Thesis, Radboud University Nijmegen.

Wieling, M., Prokić, J., & Nerbonne, J. (2009). Evaluating the pairwise string alignment of pronunciations. In Borin, L., & Lendvai, P. (Eds.), Proceedings Language Technology and Resources for Cultural Heritage, Social Sciences, Humanities, and Education (LaTeCH—SHELT&R 2009) Workshop at the 12th Meeting of the European Chapter of the Association for Computational Linguistics. Athens, March 30, 2009, pp. 26–34