Using character overlap to improve language transformation

Sander Wubben
Tilburg University
P.O. Box 90135
5000 LE Tilburg
The Netherlands
s.wubben@uvt.nl

Emiel Krahmer
Tilburg University
P.O. Box 90135
5000 LE Tilburg
The Netherlands
e.j.krahmer@uvt.nl

Antal van den Bosch
Radboud University Nijmegen
P.O. Box 9103
6500 HD Nijmegen
The Netherlands
a.vandenbosch@let.ru.nl

Abstract

Language transformation can be defined as translating between diachronically distinct language variants. We investigate the transformation of Middle Dutch into Modern Dutch by means of machine translation. We demonstrate that by using character overlap the performance of the machine translation process can be improved for this task.

1 Introduction

In this paper we aim to develop a system to paraphrase between diachronically distinct language variants. For research into history, historical linguistics and diachronic language change, historical texts are of great value. Specifically from earlier periods, texts are often the only forms of information that have been preserved. One problem that arises when studying these texts is the difference between the language the text is written in and the modern variant that the researchers who want to study the texts know and speak themselves. It takes a great deal of deciphering and interpretation to be able to grasp these texts. Our aim is to facilitate scholars such as historians who do not possess extensive knowledge of Middle Dutch who are studying medieval texts. We do this by attempting to generate literal translations of the sentences in the text into modern language. In particular we focus on the task of translating Middle Dutch to modern Dutch. The transformation between language variants, either synchronically or diachronically, can be seen as a paraphrase and a translation task, as it is often impossible to categorize two languages as either variants or different languages.

We define Middle Dutch as a collection of closely related West Germanic dialects that were spoken and written between 1150 and 1500 in the area that is now defined as the Netherlands and parts of Belgium. One of the factors that make Middle Dutch difficult to read is the fact that at the time no overarching standard language existed. Modern Dutch is defined as Dutch as spoken from 1500. The variant we investigate is contemporary Dutch. An important difference with regular paraphrasing is the amount of parallel data available. The amount of parallel data for the variant pair Middle Dutch - Modern Dutch is several orders of magnitude smaller than bilingual parallel corpora typically used in machine translation (Koehn, 2005) or monolingual parallel corpora used for paraphrase generation by machine translation (Wubben et al., 2010).

We do expect many etymologically related words to show a certain amount of character overlap between the Middle and Modern variants. An example of the data is given below, from the work 'Van den vos Reynaerde' ('About Reynard the Fox'), part of the Comburg manuscript that was written between 1380-1425. Here, the first text is the original text, the second text is a modern translation in Dutch by Walter Verniers and a translation in English is added below that for clarity.

"Doe al dat hof versamet was,
Was daer niemen, sonder die das,
Hine hadde te claghene over Reynaerde,
Den fellen metten grijsen baerde."

"Toen iedereen verzameld was,
was er niemand -behalve de das-
die niet kwam klagen over Reynaert,
die deugniet met zijn grijze baard."

"When everyone was gathered,
there was noone -except the badger-
who did not complain about Reynaert,
that rascal with his grey beard."

Proceedings of the 7th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities, pages 11–19,
Sofia, Bulgaria, August 8 2013.

We can observe that although the two Dutch texts are quite different, there is a large amount of character overlap in the words that are used. Our aim is to use this character overlap to compensate for the lower amount of data that is available. We compare three different approaches to translate Middle Dutch into Modern Dutch: a standard Phrase-Based machine translation (PBMT) approach, a PBMT approach with additional preprocessing based on Needleman-Wunsch sequence alignment, and a character bigram based PBMT approach. The PBMT approach with preprocessing identifies likely translations based on character overlap and adds them as a dictionary to improve the statistical alignment process. The PBMT approach based on character bigrams rather than translating words, transliterates character bigrams and in this way improves the transformation process. We demonstrate that these two approaches outperform standard PBMT in this task, and that the PBMT transliteration approach based on character bigrams performs best.

2 Related work

Language transformation by machine translation within a language is a task that has not been studied extensively before. Related work is the study by Xu et al. (2012). They evaluate paraphrase systems that attempt to paraphrase a specific style of writing into another style. The plays of William Shakespeare and the modern translations of these works are used in this study. They show that their models outperform baselines based on dictionaries and out-of-domain parallel text. Their work differs from our work in that they target writing in a specific literary style and we are interested in translating between diachronic variants of a language.

Work that is slightly comparable is the work by Zeldes (2007), who extrapolates correspondences in a small parallel corpus taken from the Modern and Middle Polish Bible. The correspondences are extracted using machine translation with the aim of deriving historical grammar and lexical items. A larger amount of work has been published about spelling normalization of historical texts. Baron and Rayson (2008) developed tools for research in Early Modern English. Their tool, VARD 2, finds candidate modern form replacements for spelling variants in historical texts. It makes use of a dictionary and a list of spelling rules. By plugging in other dictionaries and spelling rules, the tool can be adapted for other tasks. Kestemont et al. (2010) describe a machine learning approach to normalize the spelling in Middle Dutch Text from the 12th century. They do this by converting the historical spelling variants to single canonical (present-day) lemmas. Memory-based learning is used to learn intra-lemma spelling variation. Although these approaches normalize the text, they do not provide a translation.

More work has been done in the area of translating between closely related languages and dealing with data sparsity that occurs within these language pairs (Hajič et al., 2000; Van Huyssteen and Pilon, 2009). Koehn et al. (2003) have shown that there is a direct negative correlation between the size of the vocabulary of a language and the accuracy of the translation. Alignment models are directly affected by data sparsity. Uncommon words are more likely to be aligned incorrectly to other words or, even worse, to large segments of words (Och and Ney, 2003). Out of vocabulary (OOV) words also pose a problem in the translation process, as systems are unable to provide translations for these words. A standard heuristic is to project them into the translated sentence untranslated.

Various solutions to data sparsity have been studied, among them the use of part-of-speech tags, suffixes and word stems to normalize words (Popovic and Ney, 2004; De Gispert and Marino, 2006), the treatment of compound words in translation (Koehn and Knight, 2003), transliteration of names and named entities, and advanced models that combine transliteration and translation (Kondrak et al., 2003; Finch et al., 2012) or learn unknown words by analogical reasoning (Langlais and Patry, 2007).

Vilar et al. (2007) investigate a way to handle data sparsity in machine translation between closely related languages by translating between characters as opposed to words. The words in the parallel sentences are segmented into characters. Spaces between words are marked with a special character. The sequences of characters are then used to train a standard machine translation model and a language model with n-grams up to $n = 16$. They apply their system to the translation between the related languages Spanish and Catalan, and find that a word based system outperforms their

letter-based system. However, a combined system performs marginally better in terms of BLEU scores.

Tiedemann (2009) shows that combining character-based translation with phrase-based translation improves machine translation quality in terms of BLEU and NIST scores when translating between Swedish and Norwegian if the OOV-words are translated beforehand with the character-based model.

Nakov and Tiedemann (2012) investigate the use of character-level models in the translation between Macedonian and Bulgarian movie subtitles. Their aim is to translate between the resource poor language Macedonian to the related language Bulgarian, in order to use Bulgarian as a pivot in order to translate to other languages such as English. Their research shows that using character bigrams shows improvement over a word-based baseline.

It seems clear that character overlap can be used to improve translation quality in related languages. We therefore use character overlap in language transformation between two diachronic variants of a language.

3 This study

In this study we investigate the task of translating from Middle Dutch to Modern Dutch. Similarly to resource poor languages, one of the problems that are apparent is the small amount of parallel Middle Dutch - Modern Dutch data that is available. To combat the data sparseness we aim to use the character overlap that exists between the Middle Dutch words and their Modern Dutch counterparts. Examples of overlap in some of the words given in the example can be viewed in Table1. We are interested in the question how we can use this overlap to improve the performance of the translation model. We consider three approaches: (A) Perform normal PBMT without any preprocessing, (B) Apply a preprocessing step in which we pinpoint words and phrases that can be aligned based on character overlap and (C) perform machine translation not to words but to character bigrams in order to make use of the character overlap.

We will first discuss the PBMT baseline, followed by the PBMT + overlap system and the character bigram PBMT transliteration system in Section 4. We then describe the experiment with human judges in Section 6, and its results in Section 7. We close with a discussion of our results in Section 8.

Middle Dutch	Modern Dutch
versamet	verzameld
was	was
niemen	niemand
die	de
das	das
claghene	klagen
over	over
Reynaerde	Reynaert
metten	met zijn
grijsen	grijze
baerde	baard

Table 1: Examples of character overlap in words from a fragment of 'Van den vos Reynaerde'

4 Language transformation Models

4.1 PBMT baseline

For our baseline we use the Moses software to train a phrase based machine translation (PBMT) model (Koehn et al., 2007). In general, a statistical machine translation model normally finds a best translation $\tilde{E}$ of a text in language F for a text in language E by combining a translation model $P(F|E)$ with a language model $P(E)$:

$$\tilde{E} = arg \max_{E \in E^*} P(F|E)P(E)$$

In phrase-based machine translation the sentence F is segmented into a sequence of I phrases during decoding. Each source phrase is then translated into a target phrase to form sentence E. Phrases may be reordered.

The GIZA++ statistical alignment package (Och and Ney, 2003) is used to perform the word alignments, which are later combined into phrase alignments in the Moses pipeline to build the language transformation model. GIZA++ implements IBM Models 1 to 5 and an HMM word alignment model to find statistically motivated alignments between words. We first tokenize our data. We then lowercase all data and use all sentences from the Modern Dutch part of the corpus to train an n-gram language model with the SRILM toolkit (Stolcke, 2002). Then we run the GIZA++ aligner using the training pairs of sentences in Middle Dutch and Modern Dutch. We

execute GIZA++ with standard settings and we optimize using minimum error rate training with BLEU scores. The Moses decoder is used to generate the translations.

4.2 PBMT with overlap-based alignment

Before using the Moses pipeline we perform a preprocessing alignment step based on character overlap. Word and phrase pairs that exhibit a large amount of character overlap are added to the parallel corpus that GIZA++ is trained on. Every time we find a phrase or word pair with large overlap it is added to the corpus. This helps bias the alignment procedure towards aligning similar words and reduces alignment errors. To perform the preprocessing step we use the Needleman-Wunsch algorithm (Needleman and Wunsch, 1970). The Needleman-Wunsch algorithm is a dynamic programming algorithm that performs a global alignment on two sequences. Sequence alignment is a method to find commonalities in two (or more) sequences of some items or characters. One often used example is the comparison of sequences of DNA to find evolutionary differences and similarities. Sequence alignment is also used in linguistics, where it is applied to finding the longest common substring or the differences or similarities between strings.

The Needleman-Wunsch algorithm is a sequence alignment algorithm that optimizes a score function to find an optimal alignment of a pair of sequences. Each possible alignment is scored according to the score function, where the alignment giving the highest similarity score is the optimal alignment of a pair of sequences. If more than one alignment yields the highest score, there are multiple optimal solutions. The algorithm uses an iterative matrix to calculate the optimal solution. All possible pairs of characters containing one character from each sequence are plotted in a 2-dimensional matrix. Then, all possible alignments between those characters can be represented by pathways through the matrix. Insertions and deletions are allowed, but can be penalized by means of a gap penalty in the alignment.

The first step is to initialize the matrix and fill in the gap scores in the top row and leftmost column. In our case we heuristically set the values of the scores to +1 for matches, -2 for mismatches and -1 for gaps after evaluating on our development set. After initialization, we can label the cells in the matrix $C(i,j)$ where $i = 1,2,...,N$ and $j = 1,2,...,M$, the score of any cell $C(i,j)$ is then the maximum of:

$$\begin{aligned} q_{diag} &= C(i-1,j-1) + s(i,j) \\ q_{down} &= C(i-1,j) + g \\ q_{right} &= C(i,j-1) + g \end{aligned}$$

Here, $s(i,j)$ is the substitution score for letters i and j, and g is the gap penalty score. If i and j match, the substitution score is in fact the matching score. The table is filled this way recursively, filling each cell with the maximum score of the three possible options (diagonally, down and right). After this is done, an optimal path can be found by performing a traceback, starting in the lower right corner of the table and ending in the upper left corner, by visiting cells horizontally, vertically or diagonally, but only those cells with the highest score. After this process we end up with an alignment.

We use the Needleman-Wunsch algorithm to find an optimal alignment of the Middle Dutch - Modern Dutch sentence pairs. We regard each line as a sentence. In case of rhyming text, a frequent phenomenon in Middle Dutch text, lines are usually parts of sentences. We then consider each line a string, and we try to align as many characters and whitespaces to their equivalents in the parallel line. We split the aligned sentences in each position where two whitespaces align and we consider the resulting aligned words or phrases as alignments. For each aligned word or phrase pair we calculate the Jaccard coefficient and if that is equal or higher than a threshold we add the aligned words or phrases to the training material. We heuristically set this threshold to 0.5. By using this method we already can find many-to-one and one-to-many alignments. In this way we help the GIZA++ alignment process by biasing it towards aligning words and phrases that show overlap. Table 2 illustrates this process for two lines.

4.3 Character bigram transliteration

Another somewhat novel approach we propose for Language Transformation is Character-based transliteration. To circumvent the problem of

Middle Dutch:	hine	hadde+	++te	claghene	over	Reynaerde	,
Modern Dutch:	di+e	++niet	kwam	klag+en+	over	Reynaer+t	,
Jaccard	0.4	0.14	0	0.63	1	0.70	1

Middle Dutch:	+den	fe++llen	met++ten	grijsen	baerde	.
Modern Dutch:	die+	deugniet	met zijn	grijze+	baard+	.
Jaccard	0.50	0.09	0.50	0.71	0.8	1

Table 2: Alignment of lines with Jaccard scores for the aligned phrases. A + indicates a gap introduced by the Needleman Wunsch alignment.

OOV-words and use the benefits of character overlap more directly in the MT system, we build a translation model based on character bigrams, similar to (Nakov and Tiedemann, 2012). Where they use this approach to translate between closely related languages, we use it to translate between diachronic variants of a language. The sentences in the parallel corpus are broken into character bigrams, with a special character representing whitespaces. These bigrams are used to train the translation model and the language model. An example of the segmentation process is displayed in Table 3. We train an SRILM language model on the character bigrams and model sequences of up to 10 bigrams. We then run the standard Moses pipeline, using GIZA++ with standard settings to generate the phrase-table and we use the Moses decoder to decode the bigram sequences. A number of sample entries are shown in Table 4. As a final step, we recombine the bigrams into words. The different sizes of the Phrase-table for the different approaches can be observed in Table 5.

original	segmented
Hine	#H Hi in ne e#
hadde	#h ha ad dd de e#
te	#t te e#
claghene	#c cl la ag gh he en ne e#
over	#o ov ve er r#
Reynaerde	#R Re ey yn na ae er rd de e#
,	#, ,#

Table 3: Input and output of the character bigram segmenter

5 Data Set

Our training data consists of various Middle Dutch literary works with their modern Dutch translation. A breakdown of the different works is in Table 6. All works are from the Digital Library of Dutch Literature[1]. "Middle Dutch" is a very broad definition. It encompasses all Dutch language spoken and written between 1150 and 1500 in the Netherlands and parts of Belgium. Works stemming from different centuries, regions and writers can differ greatly in their orthography and spelling conventions. No variant of Dutch was considered standard or the norm; Middle Dutch can be considered a collection of related lects (regiolects, dialects). This only adds to the problem of data

#d da at t#	en n# #d da aa ar
#d da at t#	et t# #s st
#d da at t#	et t# #s
#d da at t#	ie et t# #s st
#d da at t#	ie et t# #s
#d da at t#	la an n#
#d da at t#	le et t#
#d da at t#	n# #d da aa ar ro
#d da at t#	n# #d da aa ar
#d da at t#	n#
#d da at t#	rd da at t#
#d da at ts s#	#d da at t#
#d da at ts si i#	#h he eb bb be en
#d da at ts	#d da at t#
#d da at ts	#w wa at t#
#d da at tt tu	#w wa at t# #j

Table 4: Example entries from the character bigram Phrase-table, without scores.

system	lines
PBMT	20,092
PBMT + overlap	27,885
character bigram transliteration	93,594

Table 5: Phrase-table sizes of the different models

[1] http://www.dbnl.org

sparsity. Our test set consists of a selection of sentences from the Middle Dutch work Beatrijs, a Maria legend written around 1374 by an anonymous author.

source text	lines	date of origin
Van den vos Reynaerde	3428	around 1260
Sint Brandaan	2312	12th century
Gruuthuuse gedichten	224	around 1400
't Prieel van Trojen	104	13th century
Various poems	42	12th-14th cent.

Table 6: Middle Dutch works in the training set

Middle Dutch	Si seide: 'Ic vergheeft u dan. Ghi sijt mijn troest voer alle man
Modern Dutch	Ze zei: 'ik vergeef het je dan. Je bent voor mij de enige man
PBMT	Ze zei : ' Ik vergheeft u dan . Gij ze alles in mijn enige voor al man
PBMT + Overlap	Ze zei : ' Ik vergheeft u dan . dat ze mijn troest voor al man
Char. Bigram PBMT	Ze zeide : ' Ik vergeeft u dan . Gij zijt mijn troost voor alle man
Middle Dutch	Dat si daer snachts mochte bliven. 'Ic mocht u qualijc verdriven,'
Modern Dutch	omdat ze nu niet verder kon reizen. 'Ik kan u echt de deur niet wijzen,'
PBMT '	dat ze daar snachts kon bliven . Ik kon u qualijc verdriven , '
PBMT + Overlap	dat ze daar s nachts kon blijven . ' Ik kon u qualijc verdriven , '
Char. Bigram PBMT	dat zij daar snachts mocht blijven . ' Ik mocht u kwalijk verdrijven ,

Table 7: Example output

6 Experiment

In order to evaluate the systems, we ran an experiment to collect human rankings of the output of the systems. We also performed automatic evaluation.

6.1 Materials

Because of the nature of our data, in which sentences often span multiple lines, it is hard to evaluate the output on the level of separate lines. We therefore choose to evaluate pairs of lines. We randomly choose a line, and check if it is part of a sensible sentence that can be understood without more context. If that is the case, we select it to include in our test set. In this way we select 25 pairs of lines. We evaluate the translations produced by the three different systems for these sentences. Examples of the selected sentences and the generated corresponding output are displayed in Table 7.

6.2 Participants

The participants in this evaluation study were 22 volunteers. All participants were native speakers of Dutch, and participated through an online interface. All participants were adults, and 12 were male and 10 female. In addition to the 22 participants, one expert in the field of Middle Dutch also performed the experiment, in order to be able to compare the judgements of the laymen and the expert.

6.3 Procedure

The participants were asked to rank three different automatic literal translations of Middle Dutch text. For reference, they were also shown a modern (often not literal) translation of the text by Dutch author Willem Wilmink. The order of items to judge was randomized for each participant, as well as the order of the output of the systems for each sentence. The criterium for ranking was the extent to which the sentences could be deciphered and understood. The participants were asked to always provide a ranking and were not allowed to assign the same rank to multiple sentences. In this way, each participant provided 25 rankings where each pair of sentences received a distinct rank. The pair with rank 1 is considered best and the pair with 3 is considered worst.

system	mean rank	95 % c. i.
PBMT	2.44 (0.03)	2.38 - 2.51
PBMT + Overlap	2.00 (0.03)	1.94 - 2.06
char. bigram PBMT	1.56 (0.03)	1.50 - 1.62

Table 8: Mean scores assigned by human subjects, with the standard error between brackets and the lower and upper bound of the 95 % confidence interval

7 Results

7.1 Human judgements

In this section we report on results of the experiment with judges ranking the output of the systems. To test for significance of the difference in the ranking of the different systems we ran repeated measures analyses of variance with system (PBMT, PBMT + Overlap, character bigram MT) as the independent variable, and the ranking of the output as the dependent variable. Mauchly's test for sphericity was used to test for homogeneity of

PBMT	PBMT + overlap	char. bigram PBMT	X^2
2.05	2.59	1.36	16.636**
2.77	1.82	1.41	21.545**
2.50	1.27	2.23	18.273**
1.95	1.45	2.59	14.273**
2.18	2.36	1.45	10.182**
2.45	2.00	1.55	9.091*
2.91	1.77	1.32	29.545**
2.18	2.27	1.55	6.903*
2.14	2.00	1.86	0.818
2.27	1.73	2.00	3.273
2.68	1.68	1.64	15.364**
2.82	1.95	1.23	27.909**
2.68	2.09	1.23	23.545**
1.95	2.55	1.50	12.091**
2.77	1.86	1.36	22.455**
2.32	2.23	1.45	9.909**
2.86	1.91	1.23	29.727**
2.18	1.09	2.73	30.545**
2.05	2.09	1.86	0.636
2.73	2.18	1.09	30.545**
2.41	2.27	1.32	15.545**
2.68	2.18	1.14	27.364**
1.82	2.95	1.23	33.909**
2.73	1.95	1.32	21.909**
2.91	1.77	1.32	29.545**

Table 9: Results of the Friedman test on each of the 25 sentences. Results marked * are significant at $p < 0.05$ and results marked ** are significant at $p < 0.01$

variance, but was not significant, so no corrections had to be applied. Planned pairwise comparisons were made with the Bonferroni method. The mean ranking can be found in Table 8 together with the standard deviation and 95 % confidence interval. We find that participants ranked the three systems differently, $F(2, 42) = 135, 604, p < .001, \eta_p^2 = .866$. All pairwise comparisons are significant at $p < .001$. The character bigram model receives the best mean rank (1.56), then the PBMT + Overlap system (2.00) and the standard PBMT system is ranked lowest (2.44). We used a Friedman test to detect differences across multiple rankings. We ran the test on each of the 25 K-related samples, and found that for 13 sentences the ranking provided by the test subjects was equal to the mean ranking: the PBMT system ranked lowest, then the PBMT + Overlap system and the character bigram system scored highest for each of these cases at $p < .005$. These results are detailed in Table 9. When comparing the judgements of the participants with the judgements of an expert, we find a significant medium Pearson correlation of .65 ($p < .001$) between the judgements of the expert and the mean of the judgements of the participants. This indicates that the judgements of the laymen are indeed useful.

7.2 Automatic judgements

In order to attempt to measure the quality of the transformations made by the different systems automatically, we measured NIST scores by comparing the output of the systems to the reference translation. We do realize that the reference translation is in fact a literary interpretation and not a literal translation, making automatic assessment harder. Having said that, we still hope to find some effect by using these automatic measures. We only report NIST scores here, because BLEU turned out to be very uninformative. In many cases sentences would receive a BLEU score of 0. Mauchly's test for sphericity was used to test for homogeneity of variance for the NIST scores, and was not significant. We ran a repeated measures test with planned pairwise comparisons made with the Bonferroni method. We found that the NIST scores for the different systems differed significantly ($F(2, 48) = 6.404, p < .005, \eta_p^2 = .211$). The average NIST scores with standard error and the lower and upper bound of the 95 % confidence interval can be seen in Table 10. The character bigram transliteration model scores highest with 2.43, followed by the PBMT + Overlap model with a score of 2.30 and finally the MT model scores lowest with a NIST score of 1.95. We find that the scores for the PBMT model differ significantly from the PBMT + Overlap model ($p < .01$) and the character bigram PBMT model ($p < .05$), but the scores for the PBMT + Overlap and the character bigram PBMT model do not differ significantly. When we compare the automatic scores to the human assigned ranks we find no significant Pearson correlation.

8 Conclusion

In this paper we have described two modifications of the standard PBMT framework to improve the transformation of Middle Dutch to Modern Dutch

system	mean NIST	95 % c. i.
PBMT	1.96 (0.18)	1.58 - 2.33
PBMT + overlap	2.30 (0.21)	1.87 - 2.72
char. bigram PBMT	2.43 (0.20)	2.01 - 2.84

Table 10: Mean NIST scores, with the standard error between brackets and the lower and upper bound of the 95 % confidence interval

by using character overlap in the two variants. We described one approach that helps the alignment process by adding words that exhibit a certain amount of character overlap to the parallel data. We also described another approach that translates sequences of character bigrams instead of words. Reviewing the results we conclude that the use of character overlap between diachronic variants of a language is beneficial in the translation process. More specifically, the model that uses character bigrams in translation instead of words is ranked best. Also ranked significantly better than a standard Phrase Based machine translation approach is the approach using the Needleman-Wunsch algorithm to align sentences and identify words or phrases that exhibit a significant amount of character overlap to help the GIZA++ statistical alignment process towards aligning the correct words and phrases. We have seen that one issue that is encountered when considering the task of language transformation from Middle Dutch to Modern Dutch is data sparseness. The number of lines used to train on amounts to a few thousand, and not millions as is more common in SMT. It is therefore crucial to use the inherent character overlap in this task to compensate for the lack of data and to make more informed decisions. The character bigram approach is able to generate a translation for out of vocabulary words, which is also a solution to the data sparseness problem. One area where the character bigram model often fails, is translating Middle Dutch words into Modern Dutch words that are significantly different. One example can be seen in Table 7, where 'mocht' is translated by the PBMT and PBMT + Overlap systems to 'kon' and left the same by the character bigram transliteration model. This is probably due to the fact that 'mocht' still exists in Dutch, but is not as common as 'kon' (meaning 'could'). another issue to consider is the fact that the character bigram model learns character mappings that are occurring trough out the language. One example is the disappearing silent 'h' after a 'g'. This often leads to transliterated words of which the spelling is only partially correct. Apparently the human judges rate these 'half words' higher than completely wrong words, but automatic measures such as NIST are insensitive to this.

We have also reported the NIST scores for the output of the standard PBMT approach and the two proposed variants. We see that the NIST scores show a similar patterns as the human judgements: the PBMT + Overlap and character bigram PBMT systems both achieve significantly higher NIST scores than the normal PBMT system. However, the PBMT + Overlap and character bigram PBMT models do not differ significantly in scores. It is interesting that we find no significant correlation between human and automatic judgements, leading us to believe that automatic judgements are not viable in this particular scenario. This is perhaps due to the fact that the reference translations the automatic measures rely on are in this case not literal translations but rather more loosely translated literary interpretations of the source text in modern Dutch. The fact that both versions are written on rhyme only worsens this problem, as the author of the Modern Dutch version is often very creative.

We think techniques such as the ones described here can be of great benefit to laymen wishing to investigate works that are not written in contemporary language, resulting in improved access to these older works. Our character bigram transliteration model may also play some role as a computational model in the study of the evolution of orthography in language variants, as it often will generate words that are strictly speaking not correct, but do resemble Modern Dutch in some way. Automatic evaluation is another topic for future work. It would be interesting to see if an automatic measure operating on the character level correlates better with human judgements.

References

Alistair Baron and Paul Rayson. 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.

A. De Gispert and J. B. Marino. 2006. Linguistic knowledge in statistical phrase-based word alignment. *Natural Language Engineering*, 12(1):91–108.

Andrew Finch, Paul Dixon, and Eiichiro Sumita. 2012. Rescoring a phrase-based machine transliteration system with recurrent neural network language models. In *Proceedings of the 4th Named Entity Workshop (NEWS) 2012*, pages 47–51, Jeju, Korea, July. Association for Computational Linguistics.

Jan Hajič, Jan Hric, and Vladislav Kuboň. 2000. Machine translation of very close languages. In *Proceedings of the sixth conference on Applied natural language processing*, pages 7–12. Association for Computational Linguistics.

Mike Kestemont, Walter Daelemans, and Guy De Pauw. 2010. Weigh your words—memory-based lemmatization for Middle Dutch. *Literary and Linguistic Computing*, 25(3):287–301, September.

Philipp Koehn and Kevin Knight. 2003. Feature-rich statistical translation of noun phrases. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 311–318, Morristown, NJ, USA. Association for Computational Linguistics.

Philip Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris C. Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL*. The Association for Computer Linguistics.

P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5.

Grzegorz Kondrak, Daniel Marcu, and Kevin Knight. 2003. Cognates can improve statistical translation models. In *HLT-NAACL*.

Philippe Langlais and Alexandre Patry. 2007. Translating unknown words by analogical learning. In *EMNLP-CoNLL*, pages 877–886. ACL.

Preslav Nakov and Jörg Tiedemann. 2012. Combining word-level and character-level models for machine translation between closely-related languages. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 301–305, Jeju Island, Korea, July. Association for Computational Linguistics.

S. B. Needleman and C. D. Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, March.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Maja Popovic and Hermann Ney. 2004. Towards the use of word stems and suffixes for statistical machine translation. In *LREC*. European Language Resources Association.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *In Proc. Int. Conf. on Spoken Language Processing*, pages 901–904, Denver, Colorado.

Jörg Tiedemann. 2009. Character-based PSMT for closely related languages. In Lluís Marqués and Harold Somers, editors, *Proceedings of 13th Annual Conference of the European Association for Machine Translation (EAMT'09)*, pages 12 – 19, Barcelona, Spain, May.

Gerhard B Van Huyssteen and Suléne Pilon. 2009. Rule-based conversion of closely-related languages: a dutch-to-afrikaans convertor. *20th Annual Symposium of the Pattern Recognition Association of South Africa*, December.

David Vilar, Jan-Thorsten Peter, and Hermann Ney. 2007. Can we translate letters? In *Second Workshop on Statistical Machine Translation*, pages 33–39, Prague, Czech Republic, jun. Association for Computational Linguistics.

Sander Wubben, Antal van den Bosch, and Emiel Krahmer. 2010. Paraphrase generation as monolingual translation: Data and evaluation. In B. Mac Namee J. Kelleher and I. van der Sluis, editors, *Proceedings of the 10th International Workshop on Natural Language Generation (INLG 2010)*, pages 203–207, Dublin.

Wei Xu, Alan Ritter, Bill Dolan, Ralph Grishman, and Colin Cherry. 2012. Paraphrasing for style. In *COLING*, pages 2899–2914.

Amir Zeldes. 2007. Machine translation between language stages: Extracting historical grammar from a parallel diachronic corpus of Polish. In Matthew Davies, Paul Rayson, Susan Hunston, and Pernilla Danielsson, editors, *Proceedings of the Corpus Linguistics Conference CL2007*. University of Birmingham.