Applying Rule-Based Normalization to Different Types of Historical Texts — An Evaluation

Marcel Bollmann, Florian Petran, and Stefanie Dipper

Department of Linguistics, Ruhr-Universität Bochum, 44780 Bochum, Germany,
{bollmann|petran|dipper}@linguistics.rub.de

Abstract. This paper deals with normalization of language data from Early New High German. We describe an unsupervised, rule-based approach which maps historical wordforms to modern wordforms. Rules are specified in the form of context-aware rewrite rules that apply to sequences of characters. They are derived from two aligned versions of the Luther bible and weighted according to their frequency. Applying the normalization rules to texts by Luther results in 91% exact matches, clearly outperforming the baseline (65%). Matches can be improved to 93% by combining the approach with a word substitution list. If applied to more diverse language data from roughly the same period, performance goes down to 43% exact matches (baseline: 35%), and to 46% using the combined method. The results show that rules derived from a highly different type of text can support normalization to a certain extent.

Keywords: historical language data, spelling normalization, rewrite rules

1 Introduction[1]

Historical language data differs from modern data in that there are no agreed-upon, standardized writing conventions. Instead, characters and symbols used by the writer of some manuscript in parts reflect impacts as different as spatial constraints or dialect influences. This often leads to inconsistent spellings, even within one text written up by one writer.[2]

The ultimate goal of our research is an automatic mapping from wordforms from Early New High German (ENHG, 14th–16th centuries) to the corresponding modern wordforms from New High German (NHG). Enriching the data with modern wordform annotations facilitates further processing, e.g. by POS taggers, and simplifies wordform queries by other users working with the data.

The approach we pursue, first described in [2], is to derive character replacement rules from a parallel corpus we built from two freely available texts:

[1] This is a revised and extended version of [1], evaluated on a larger test corpus than the original paper. The research reported here was financed by Deutsche Forschungsgemeinschaft (DFG), Grant DI 1558/4-1.

[2] Some of these characteristics in fact show up again in specific uses of modern language, such as contributions in chat rooms.

Luther's bible translation in the original ENHG version, and in a modernized NHG version. The idea is to exploit these resources to derive normalization rules that subsequently can be applied to historical texts which do not have modernized editions. Compared to a simple wordlist substitution approach, the rule-based method is able to capture generalizations and, consequently, to produce correct normalizations even for a number of wordforms which were not seen during training.

In our evaluation, we compare performance of the rule-based approach, a wordlist substitution method, and a combination of both. We apply those methods (i) to texts by Luther, and (ii) to further religious texts (which are less similar to NHG than Luther's texts). It turns out that the wordlist substitution method works suprisingly well, while a combined approach produces the best results overall.

The paper is organized as follows. In Sec. 2, we introduce our data. Sec. 3 addresses the way we derive rewrite rules from the data, while Sec. 4 deals with the application of the rules to generate modern wordforms. Sec. 5 presents the comparative evaluation, Sec. 6 discusses related work and recent developments, and Sec. 7 presents a conclusion.

2 Corpora

2.1 Luther Bible

In our approach, replacement rules are derived from a word-aligned parallel corpus. A source that provides parallel texts in many languages, including historical ones, is the bible. We retrieved two editions of the bible translated by Martin Luther from the web[3]: the original ENHG version of the 1545 bible (which has been enriched with modern punctuation), as well as a revised NHG version of it, which uses modern spelling and replaces extinct words by modern ones.

We aligned the editions on the basis of verses, sentences, and words, using the Gargantua toolkit [3] and the GIZA++ toolkit [4]. To minimize noise in our system's input, alignment pairs with a length difference of more than five characters were excluded from further processing. Since the two languages are highly similar – around 65% of the pairs align identical wordforms – a length difference of that magnitude rarely leads to meaningful alignments.

To assess the quality of the resulting word pairs, a small sample of 1,000 pairs of aligned non-identical wordforms was manually inspected by a student assistant. Three types of mappings can be distinguished: The large majority (61%) consists of correct, obvious mappings (such as *vnd* – *und* 'and'). 30% are word pairs that differ with respect to inflection or affixing (e.g. *truncken* – *trunkenen* 'drunken', *oben* – *obenan* 'on top'). The remaining pairs consist of historically unrelated words.

For deriving replacement rules, obvious mappings are the perfect input. The second type of mappings is still useful, to a certain extent: correct rules can be

[3] `http://www.sermon-online.de`

derived from the roots of the words; mapping of differing inflection and affixes, however, should probably not be learned. The last type (which occurs rather rarely) represents less useful input for our learning approach.

The resulting corpus consists of 550,000 aligned pairs of words or phrases. We randomly picked 20% of the alignment pairs for a development corpus, which was used for the development of the rule extraction and application processes described below. Another 40% were afterwards used to extract the replacement rules for the following experiments (= training corpus), and a final 20% were picked for an evaluation corpus. We held back another 20% for additional future evaluations.

The Luther bible serves us both as a source for deriving replacement rules (from the training portion) as well as a text to apply the rules and map historical wordforms to modern ones (the evaluation portion). Evaluation results from applying the rules are presented in Sec. 5.

2.2 Anselm Corpus

In addition to Luther's texts, we evaluate the rules on other texts which show considerably more variation than Luther's bible. The data consists of different manuscripts of the text "Interrogatio Sancti Anselmi de Passione Domini" ('Questions by Saint Anselm about the Lord's Passion'). The text contains a collection of questions posed by Saint Anselm of Canterbury and answered by Saint Mary. In the 14th–16th centuries, this text has been written up in various German dialects (from Upper, Central, and Low German), and transformed into long and short prose and lyric versions. In total, there are more than 50 manuscripts and prints, which makes the text an exceptionally broadly-documented resource.

These manuscripts are difficult to use for deriving normalization rules, since they are rather short (about 8,000 tokens on average for the entire Anselm corpus), and also there is no modernized version available. While we experimented on semi-automatic normalization with small amounts of manually annotated training data later [5], the study reported here concerns the use of large amounts of training data from available sources. We therefore decided to try to apply the rules derived from the Luther bible to the Anselm corpus and evaluate their performance on a set of manually normalized (i.e. modernized) data. We selected four manuscripts from different major dialectal regions to capture a broad range of graphematic variation: Alemannic (ALEM), Bavarian (BAV), West Central German (WCG), and East Central German (ECG). With an average length of 6,200, they are slightly smaller than the Anselm average, but since they are carefully annotated with gold standard normalizations, they make a good sample set. The normalizations were independently done by two student assistants and manually revised by a historical linguist. For a detailed discussion of the treatment of special writing conventions, refer to [5].

Differences between the dialects most prominently concern wordforms (e.g. *chind* – *kínt* – *kynt* 'child', *geezzen* – *geſzen* – *geſſen* 'eaten') as well as word order. Figure 1 shows the normalization of an exemplary passage from the ALEM text.

ENHG:	do	min	kint	hatt	geſſen	das	Iung	maſz	mit	ſinen	Iungren
NHG:	da	mein	kind	hatte	gegessen	das	junge	mahl	mit	seinen	jüngern
	as	my	child	had	eaten	the	young	meal	with	his	disciples

Fig. 1. Passage from the ALEM text and a word-by-word normalization (NHG)

3 Normalization Rules

We use a modified algorithm for computing Levenshtein distance which not only calculates the numerical edit distance, but also outputs the respective edit operations (substitution, insertion, and deletion of single characters) that map the strings to each other. Moreover, the record of edit operations was enriched with information about the immediate context of the edited character. Ex. (1) shows two sample edit operations, using the notation of phonological rewrite rules.

(1) a. $\varepsilon \rightarrow$ `h / j _ r`
   (‘h’ is inserted between ‘j’ and ‘r’)

  b. `v` $\rightarrow$ `u / # _ n`
   (‘v’ is replaced by ‘u’ between the left word boundary (‘#’) and ‘n’)

Determining the context for these edit operations is not straightforward, because applying one rule can change the context for other rules. Since the Levenshtein implementation applies the rules from left to right, we use the (new) target word for the left context and the (unaltered) source word for the right context.

Identity rules In addition to canonical replacement rules, our rule induction algorithm also produces identity rules, i.e. rewrite rules that map a character to itself. Identity rules reflect the fact that words can remain unaltered when mapped to their modernized forms, and many words are modified by few characters only. Identity rules and actual rewrite rules are intended to compete with each other during the process of rewriting: whenever multiple rules are applicable at the same position within a word, the rule that is ranked higher (according to a ranking system described below) “wins” and is applied. If an identity rule is ranked higher than a non-identity rule, this results in the character in question not being modified.

Sequence-based rules In a second step, we merge non-identity rules, resulting in replacement rules that operate on *sequences* of characters. Whenever a series of edits occurs at the same target or source position, we assume that this is actually an insertion or deletion of a *sequence* of characters, such as an affix. Whenever edits occur at adjacent positions, we assume that it is a substitution of a character sequence by another. By merging substitutions with adjacent deletions/additions, we account for character sequences of variable length on each side of the rule.[4] As an example, see Ex. (2), mapping *jrem* to *ihrem* 'their' ('s' is for substitution, '=' means identity mapping).

(2)

Input	*j*	*r*	*e*	*m*
Operations	s	=	=	=
Output	*ih*	*r*	*e*	*m*

Epsilon identity rules In the system developed so far, insertion rules are rather difficult to handle. In generating modern wordforms by means of the rewrite rules (see Sec. 4), they tend to apply in an uncontrollable way, garbling the words in the process. This is due to the fact that the conditions for the application of insertion rules are less specific: while substitutions and deletions require the left hand side (LHS) of the rule *and* the context to match in the source word, insertions are constrained by their two context characters only, since the LHS of the rule is empty. At word boundaries, the problem gets even worse, since only one context side is specified here. Furthermore, substitution and deletion rules compete against the identity rules for their LHS, which restricts their application—but no similar competitor exists so far to curb the application of insertion rules. We therefore introduced the concept of epsilon identity rules:

(3) $\varepsilon \rightarrow \varepsilon$ / e _ n

These rules are taken to mean that no insertion should be performed in the respective context. Just as the identity rules described above are supposed to compete with actual rewrite rules, epsilon identity rules compete with actual insertion rules to determine whether or not an insertion should take place. That way, they restrict the actual insertion of characters. Derivation of these rules is relatively straightforward: after all replacement rules for an alignment pair have been generated, an epsilon identity rule is created for each position where no insertion has taken place.

Ranking of the rules Applying the rule induction algorithm to the development corpus yielded about 1.1 million rule instances (training corpus: 2.2 million) of about 6,500 different types (training: 7,902). These were sorted and ranked according to their frequency. Table 1 lists sample instances of rules. Not surprisingly, none of the top-ranked rules actually modifies the input word. Rank 6 is taken by the first rule that maps some real character rather than ε to itself (identity rules, '='). Rank 20 features the most frequently-seen substitution rule ('s'), rank 176 the first deletion ('–'), rank 239 the first insertion rule ('+'). The bottom part of the table lists frequent sequence-based rules. The rule ranked 605th shows that the algorithm can also produce 1:N mappings, i.e. rules which map one input word to several output words. It inserts a whitespace followed by 'd' in a certain context, which applies in mappings such as *soltu* – *sollst du* 'should you', where *soltu* represents a contraction of verb and pronoun.

[4] Identity rules are excluded from the merging process. Otherwise, merging would result in mappings of entire words instead of character sequences, basically identical to a word substitution list.

Table 1. Sample rankings and rules

	Rank	Freq.	Rule
=	1	24,867	ε → ε / n _ #
=	2	18,213	ε → ε / e _ r
=	3	18,200	ε → ε / e _ n
=	4	17,772	ε → ε / # _ d
=	5	14,871	ε → ε / r _ #
=	6	14,853	n → n / e _ #
s	20	8,448	v → u / # _ n
−	176	1,288	f → ε / u _ f
+	239	932	ε → l / o _ l
	156	1,443	j → ih / # _ r
	272	796	j → ih / # _ n
	329	601	j → ih / # _ m
	605	263	ε → ␣d / t _ u
	879	142	ss → ß / o _ e

Table 2. Example mappings from the Luther bible and the Anselm texts.

	Original		**Auto**
	Old	Modernized	
Bible	jrem	ihrem	✓
	vmbher	umher	✓
	soltu	sollst du	✓
	gemachen	gemächern	†gemächer
Anselm	etleich	etliche	✓
	gessen	gegessen	✓
	vnse	unsere	†unser
	vrouwe	frau	*vrouwe
	zitt	zeit	*zittern
Both	vnd	und	✓

4 Generating Normalized Wordforms

Automatically normalizing ENHG texts is done on a word-by-word basis, i.e. the input of the normalizing process is always a single word form. Words are processed from left to right; for each position within a word, applicable edit rules are determined. As with the rule extraction process, the left context is matched against the output already generated up to that point, while the right context is always matched against the input word. If a rule is applied, its right-hand side is appended to the output string, and the next character from the input word is processed. The process continues until the end of the word has been reached.

Rules with sequences of characters on the left-hand side (LHS) are applicable at the position of their first LHS character. In that case, if the rule is applied, processing continues with the next character that is not part of the LHS. If there is no applicable rule for a given position in a word, the character at that position is left unchanged and processing continues with the next character.

Epsilon rules Epsilon rules, as explained in Sec. 3, are supposed to signify whether or not an insertion between two characters should be performed. In order for this to work, only one epsilon rule can be applied between any two given characters, or otherwise multiple—possibly infinite—insertions could take place. This is achieved by treating epsilon like an ordinary letter with regard to the LHS of rules, and preprocessing words so that exactly one epsilon is placed between each character and at word boundaries. For example, the input word *jrem* 'their' is converted internally to the following form:

(4) # ε j ε r ε e ε m ε #

Now, whenever an epsilon rule is applied, the read/write head moves to the next character, so that no other epsilon rule can be applied at the same time in the same position. Note that this does not generally prevent the insertion of multiple characters, as those are merged into sequences during rule extraction. Of course, epsilon characters have to be ignored for all purposes except for the LHS of replacement rules; in particular, they do never contribute to rule contexts or prevent the recognition of character sequences.

Selecting an output variant For each character and its context within a word, there will usually be a number of applicable rules to choose from. Therefore, several output variants can usually be generated from one input word. As our aim is to generate exactly one (modernized) form for each input word, a decision has to be made about which variant to choose. To this end, each generated variant is assigned a probability score.[5] The probability of a replacement rule is defined as its frequency divided by the sum of all rule frequencies. Word probability is calculated from the probabilities of the rules that were used to generate it; for this, we use the weighted harmonic mean, with the length of the LHS as weights. If the LHS contains a sequence, length is counted including additional epsilons between each character. This way, all variants generated from the same input word have the same total weight, regardless of whether sequence-based rules were used or not.

Additionally, to block the generation of nonsense words, all generated variants are checked against a dictionary. From all variants that are covered by the dictionary, the one with the highest probability score is then selected as the output form. If no variant can be generated in this way, the input word is left unchanged. In the evaluation presented below, we used all wordforms from the modernized Luther bible as our dictionary. This skews the results slightly in our favor when normalizing the 1545 bible text. Using a dictionary with wordforms from current newspaper texts, however, turned out problematic, since abbreviations, typos, etc., result in too many false positives with the dictionary lookup, and the vocabulary of newspaper texts differs considerably from religious texts.

Table 2 lists sample mappings from both the Luther and the Anselm corpus. '✓' marks correctly generated wordforms, '*' marks incorrect ones. Wordforms marked '†' are useful normalizations with wrong inflection, currently counted as incorrect. [6]

Wordlist substitution An alternative approach to normalization is the idea of using bilingual wordlists. In order to compare the rule-based method to a pure wordlist-based approach, we created a wordlist from the Luther training corpus. The wordlist maps each historical wordform to the modern wordform that it is most often aligned with. Using this method, normalized wordforms are generated

[5] As an alternative method, we tried always selecting the rule with the highest frequency, which we found to be an inferior approach during evaluation. For details, see [2].

[6] Note that we ignore capitalization for the time being.

by substituting an old wordform with its modern counterpart as specified in the wordlist. If a word is not in the list, it is left unchanged.

Finally, we also tried a combination of both approaches. Here, the wordlist substitution is always tried first, with the rule-based method being applied only if the historical wordform is not found in the wordlist.

5 Evaluation

For evaluation, we generated normalized forms of all historical wordforms and compared them to their modernized counterparts. For each text, there is a number of words which do not differ at all between old and modernized versions (column "Original" in Table 3). This is the baseline for our evaluation; any normalizing process that results in less than that number of matches has done more harm than good, and it would be better to leave all words unchanged. Full evaluation results are shown in Table 3.

Table 3. Identical tokens before (= "Original") and after normalization

		Tokens	**Original**	**Normalization**		
				Rules	Wordlist	Both
Luther	All	109,972	64.71%	91.00%	91.98%	92.93%
	Unknowns	2,911	40.88%	76.88%	40.88%	76.88%
Anselm	ALEM	8,201	39.06%	48.81%	49.02%	51.35%
	BAV	4,630	47.00%	57.84%	59.46%	60.74%
	WCG	7,409	33.84%	41.64%	43.53%	44.28%
	ECG	4,705	23.80%	32.88%	30.07%	35.24%
	Harmonic mean	–	35.42%	42.55%	42.58%	45.68%

5.1 Luther Bible

Before normalization, the ratio of identical tokens in the historical and the modernized text of the Luther bible is 64.71%, i.e., only a third of all wordforms even differ at all. Table 3 shows that our method increases that match ratio to 91.00%, which is a significant increase from the baseline. Our normalization approach is not only successful in changing historical forms to modern ones, but also in correctly leaving most of the word forms (about 99%) unchanged that do not need to be changed, as a separate evaluation showed. The pure wordlist substitution method achieves 91.98%, which is slightly—but significantly ($p < 0.005$)—better.

In order to evaluate the overlap between both methods, and to better assess the bias of using the same type of text for rule training, rule application, and deriving the dictionary, a separate evaluation was done on word pairs that were not seen during training ('Unknowns'). The wordlist-based approach cannot normalize any of those wordforms, as they have not been previously learned. The rule-based method, on the other hand, still achieves a considerable increase in matching pairs. Consequently, combining both methods yields the best results (92.93%) for the Luther text.

5.2 Anselm Corpus

The ratio of identical tokens before normalization (= the baseline) in the Anselm texts is only 35.42% on average. This is an indicator that these texts differ significantly more from New High German than Luther's bible—at least in spelling. Consequently, the mean match ratio after normalization is also considerably lower (42.55%). While this is far from optimal, the increase is still notable, and both methods outperform the baseline for every text. Judging from the normalized wordforms, at least some of the normalization rules learned from the Luther bible can be successfully applied to Anselm texts. This becomes apparent from non-trivial word pairs such as *liff* – *lief* 'ran', which are correctly normalized even though they do not appear in the training corpus, so the wordlist does not cover them.

In all texts except ECG, the wordlist approach performs better than the rule-based approach, by a rather small margin (between 0.21–1.89 percentage points). The harmonic means of the ratios of correct (i.e. identical) tokens show only negligible difference between the rule-based and the wordlist approach. The combination of both methods, again, yields the best results, outperforming the individual methods by 0.75–5.17 percentage points. This indicates that the approaches complete each other to a certain extent, when one succeeds where the other fails.

Table 4. Number of tokens with character sequences for which no applicable rule has been learned

	Tokens	**Tokens w/o rule**	
ALEM	8,201	2,108	25.70%
BAV	4,630	723	15.62%
WCG	7,409	2,735	36.91%
ECG	4,705	2,174	46.21%
Harmonic mean	–	–	25.61%

ECG is the only text where the rule-based approach outperforms the wordlist approach. From the four Anselm manuscripts, this text seems most dissimilar to the Luther text as far as spelling is concerned, cf. Table 4. Table 4 shows the number of tokens for which not a single normalized variant can be generated because, at some point during the process, a situation arises where no character edit rule can be applied. This means the respective token contains at least one character trigram that was not seen in the training data. ECG has the highest amount of such tokens: almost half of its tokens are concerned (46.21%).

By comparison of Tables 3 and 4, we observe a strong correlation between the amount of applicable rules from training data, and overall accuracy. BAV and ALEM, the texts with the highest accuracy after normalization, have the lowest amounts of unseen contexts. ECG, the text with the lowest accuracy, has the highest amount. These findings indicate that close relatedness of training and evaluation data is more important than large amounts of training data (this is also confirmed by [5]).

Note that we currently only count identical tokens as correct normalization. Some words produce sensible normalizations, but do not match their aligned form because of a difference in inflection: *vnse* 'our' is normalized to the masculine/neutral form *unser*, but aligned with the feminine form *unsere*.

Another difference concerns tokenization. As shown in Sec. 2, the rewrite rules for Luther include modernization of tokenization by inserting of spaces. The Anselm texts on the other hand have already been modernized with regard to token boundaries before the manual normalization takes place. This means that the rule-based approach sometimes incorrectly introduces token boundaries, e.g. modernizing *allerlibste* 'dearest' to *aller liebste*. In these cases, the normalization is still useful, but does not count as a match in our evaluation, thereby downgrading the results.

However, in many other cases, the source words contain spelling characteristics which do not occur in the bible text and therefore could not be learned. A typical example would be *vrouwe*, which should normalize to *frau* 'woman'. The spelling *v* for *f* seems to occur quite frequently in the ECG text, but only very rarely in the Luther bible: the appropriate rule (5) was learned exactly once from the training corpus.

(5) `v` $\rightarrow$ `f / # _ r`

Similarly, there are no rules which would transform *ou* into *au*, as this spelling does not occur at all in Luther. Again, this spelling is a characteristic feature of the ECG text, which could be one of the reasons why this text performs worst in our evaluation. Note, however, that ECG shows the strongest synergy between the rule-based and the wordlist approaches.

6 Related Work and Recent Developments

Research in NLP for historical texts is very active, and spelling normalization as described in our paper is still state-of-the-art for the processing of this type of data [6].

Since the original publication of our paper, we expanded upon the idea of combining normalization methods by creating the Norma tool [7], which implements wordlists, rule-based normalization, and a third method based on a modified Levenshtein distance. More importantly, though, we moved away from using the Luther bible as training data for our normalization methods, and use small fragments of in-domain training data instead. We also started to investigate the usefulness of normalization for part-of-speech tagging [8].

Another tool for spelling normalization is VARD [9], which implements its own normalization algorithms specifically targeted at Early Modern English, and offers an interface for editing and correcting the automatically normalized wordforms. While it can theoretically be adapted to other languages, we found it to perform worse than our Norma tool in a later experiment [7].

Rule-based normalization techniques similar, albeit not identical, to ours have been used for historical Dutch [10] and Swedish [11]. A project working on texts from Old Spanish [12] uses edit transducers to model the underlying rewrite rules. Work by Jurish [13] on historical German from 1780–1880 is still notable for including token context in the normalization process, contrary to most other approaches which only operate on wordform types. A different, yet promising recent approach is the application of statistical machine translation techniques on a character level [14, 15].

7 Conclusion

We showed that using only unsupervised learning, a minimum of knowledge engineering, and freely available resources, it is possible to map historical wordforms to their modern counterparts with a high success rate. With Luther's bible, the rule-based method performs far better than the baseline, and a combination of this method and the wordlist-based method was shown to produce significantly better results than one method alone.

Applying the rule-based method to a different type of text results in rather low performance. It turns out that substantial variations in spelling, as they occur in the Anselm texts, are problematic for the rule-based approach, as it cannot normalize character-context sequences that have not been previously learned. However, the combination of wordlists and rules shows that the coverage of both methods is not the same, and including rule-based normalization is still useful.

The Anselm data seems to suggest that abstracting the rules from their specific contexts could be beneficial in some cases, e.g., to cover common substitutions like *tz* by *z* or *u* by *v*, which are common variants in ENHG. This way, the rules derived from the Luther bible could be generalized and would apply to a wider range of character-context sequences. In later work [7], we included a distance-based normalization approach that tries to address this issue.

Furthermore, our later research [7] showed that using small amounts of in-domain training data (as opposed to only using the Luther bible for training) improves performance dramatically. Also, normalization using a combination of methods always performed better with the rule-based method than without it. Therefore, even though the initial scores reported here are rather low, the rule-based normalization approach continues to be an important part of our normalization method.

References

1. Bollmann, M., Petran, F., Dipper, S.: Applying rule-based normalization to different types of historical texts — an evaluation. In: Proceedings of the 5th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics, Poznan, Poland (2011)
2. Bollmann, M., Petran, F., Dipper, S.: Rule-Based Normalization of Historical Texts. In: Proceedings of the International Workshop on Language Technologies for Digital Humanities and Cultural Heritage, Hissar, Bulgaria (2011) 34–42
3. Braune, F., Fraser, A.: Improved unsupervised sentence alignment for symmetrical and asymmetrical parallel corpora. In: Proceedings of the 23rd International Conference on Computational Linguistics (COLING), Poster Volume, Beijing, China (2010) 81–89
4. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Computational Linguistics **29**(1) (2003) 19–51
5. Bollmann, M., Dipper, S., Krasselt, J., Petran, F.: Manual and semi-automatic normalization of historical spelling. Case studies from Early New High German. In: Proceedings of the First International Workshop on Language Technology for Historical Text(s), Vienna, Austria (2012)
6. Piotrowski, M.: Natural Language Processing for Historical Texts. Number 17 in Synthesis Lectures on Human Language Technologies. Morgan & Claypool, San Rafael, CA (2012)
7. Bollmann, M.: (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 (2012)
8. Bollmann, M.: POS tagging for historical texts with sparse training data. In: Proceedings of the 7th Linguistic Annotation Workshop and Interoperability in Discourse, Sofia, Bulgaria (2013) 11–18
9. Baron, A., Rayson, P., Archer, D.: Automatic standardization of spelling for historical text mining. In: Proceedings of Digital Humanities 2009, Maryland, USA (2009)
10. van Halteren, H., Rem, M.: Dealing with orthographic variation in a tagger-lemmatizer for fourteenth century Dutch charters. Language Resources and Evaluation (2013)
11. Adesam, Y., Ahlberg, M., Bouma, G.: *bokstaffua, bokstaffwa, bokstafwa, bokstaua, bokstawa...* Towards lexical link-up for a corpus of Old Swedish. In: Proceedings of KONVENS 2012 (LThist 2012 workshop), Vienna, Austria (2012) 365–369
12. Porta, J., Sancho, J.L., Gómez, J.: Edit transducers for spelling variation in Old Spanish. In: Proceedings of the NODALIDA Workshop on Computational Historical Linguistics, Oslo, Norway (2013)
13. Jurish, B.: More than words: Using token context to improve canonicalization of historical German. Journal for Language Technology and Computational Linguistics **25**(1) (2010) 23–39
14. Pettersson, E., Megyesi, B., Tiedemann, J.: An SMT approach to automatic annotation of historical text. In: Proceedings of the NODALIDA Workshop on Computational Historical Linguistics, Oslo, Norway (2013)
15. Scherrer, Y., Erjavec, T.: Modernizing historical Slovene words with character-based SMT. In: Proceedings of the 4th Biennial Workshop on Balto-Slavic Natural Language Processing. (2013)