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

Marcel Bollmann, Florian Petran, Stefanie Dipper

Ruhr-University Bochum
Department of Linguistics, 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 42% exact matches (baseline: 32%), but is higher than using a wordlist. The results show that rules derived from a highly different type of text can support normalization to a certain extent.

Keywords: historical language data, 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 Bollmann et al. (2011), is to derive character replacement rules from a parallel corpus we built from two freely available texts: 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). The rules are shown to be better than the wordlist for the latter category, 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 Sec. 7 presents a conclusion.

2. The 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.

Alignment We aligned the editions on the basis of verses, sentences, and words, using the Gargantua toolkit (Braune and Fraser, 2010) and the GIZA++ toolkit (Och and Ney, 2003). 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.

Some corpus statistics 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.

[1]We would like to thank the anonymous reviewers for their helpful comments. 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.

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

BAV: Do meín chind híet geezzen· mít ſeínen Jungern· vor ſeíner marter daz íungíſt mal.
as my child had eaten with his disciples before his martyrdom the youngest meal

WCG: Da myn kínt hatte geſzen mit ſynen jungern daz jungſte maſze
as my child had eaten with his disciples the youngest meal

ECG: do myn lybes kynt das obent brot hatte geſſen· myt ſynen iungeren
as my dear child the evening meal had eaten with his disciples

ALEM: Do mín kínt hatt geſſen das Iung maſz mít ſínen Iungren vor ſiner marter
as my child had eaten the young meal with his disciples before his martyrdom

'As my (dear) child had eaten the Last Supper together with his disciples (before his martyrdom)'

Figure 1: Sample passage from different Anselm texts

ENHG: Do meín chind híet geezzen· mít ſeínen Jungern· vor ſeíner marter daz íungíſt mal.
NHG: da mein kind hatte gegessen mit seinen jüngern vor seiner marter das jüngste mahl

Figure 2: Passage from the BAV text and a word-by-word normalization

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 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.

Procedure 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.

2.2. Anselm Corpus

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). Performance results from applying the rules are presented in Sec. 5.

In addition, we evaluate the rules on other types of data, 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 40 manuscripts and prints, which makes the text an exceptionally broadly-documented resource.

These manuscripts cannot be sensibly used for deriving normalization rules for two reasons. First, no modernized version is readily available. Second, the manuscripts are rather small (average length: 8,000 tokens). We therefore decided to try to apply the rules derived from the Luther bible to the Anselm corpus and evaluate their performance on a small subset of manually normalized data. In contrast to the Luther texts, extinct wordforms have been marked in the annotation. As our goal is to mainly eliminate variations in spelling and orthography, and not full-fledged lexical substitution of extinct wordforms, we excluded those wordforms for the purpose of this study. We selected a manuscript from the Bavarian region (BAV) consisting of 8,206 tokens as our main text. Additionally, we considered short fragments of three manuscripts (899 tokens in total) from different dialectal regions: West Central German (WCG), East Central German (ECG), and Alemannic (ALEM).

Figure 1 shows the same passage in the four different manuscripts we selected. Differences most prominently concern wordforms (e.g. *chind – kínt – kynt* 'child', *geezzen – geſzen – geſſen* 'eaten') as well as word order. Figure 2 shows a possible normalization of the passage from the BAV text.

All Anselm manuscripts were pre-processed: punctuation was deleted and special writing conventions, such as abbreviations or diacritic marks, were converted to plain alphabetic characters. For example, a superscribed horizontal bar that denotes a generic nasal ('Nasalstrich', written as '‾') was transformed to 'n', except in the very common abbreviation *vn̄* 'and', which was transformed to *vnd*. The pre-processing is basically identical to that in Dipper and Schrader (2008).

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 the majority of words 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, which look like this:

(3) ε → ε / 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.

	Rank	Frequency	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 1: Sample rankings and rules

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. Generating Normalized Wordforms

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

[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.

	Original		Generated
	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	✓

Table 2: Example mappings from the Luther bible and the Anselm texts. '✓' marks correctly generated wordforms, '*' marks incorrect ones. Wordforms marked '†' are useful normalizations with wrong inflection, currently counted as incorrect.

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.[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 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, shown in Table 4. This is the baseline for our evaluation; any normalizing process that results in less than that number of matches has likely done more harm than good, and it would be better to leave all words unchanged. Full evaluation results are shown in Table 3.

5.1. Luther Bible

Before normalization, the ratio of identical tokens in the historical and the modernized text of the Luther bible is about 65%, i.e., only a third of all wordforms even differ at all. Table 3 shows that our method increases that match ratio to 91%, 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 (ca. 99%) unchanged that do not need to be changed, as a separate evaluation showed. The pure wordlist substitution method achieves 92%, 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

[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 Bollmann et al. (2011).

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

		Total	Identical tokens					
			Rule-based		Wordlist-based		Combined	
Luther bible	All	109,972	100,074	91.00%	101,154	91.98%	102,202	92.93%
	Unknowns	2,911	2,238	76.88%	1,190	40.88%	2,238	76.88%
Anselm texts	BAV	8,206	3,916	47.72%	3,723	45.37%	3,942	48.04%
	WCG	367	195	53.13%	189	51.50%	198	53.95%
	ECG	214	61	28.50%	55	25.70%	60	28.04%
	ALEM	285	142	49.82%	135	47.37%	143	50.18%
	Harmonic mean	–	–	42.14%	–	39.42%	–	42.13%

Table 3: Identical tokens after normalization

		Total	Identical tokens	
Luther bible	All	109,972	71,163	64.71%
	Unknowns	2,911	1,190	40.88%
Anselm texts	BAV	8,206	2,896	35.29%
	WCG	367	153	41.69%
	ECG	214	47	21.96%
	ALEM	285	105	36.84%
	H. mean	–	–	32.00%

Table 4: Identical tokens before normalization (Baseline)

	Total	No rule found	
BAV	8,206	802	9.77%
WCG	367	70	19.07%
ECG	214	55	25.70%
ALEM	285	47	16.49%
Harmonic mean	–	–	15.73%

Table 5: Number of tokens with character sequences for which no applicable rule has been learned

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 (ca. 93%) for the Luther text.

5.2. Anselm Corpus

With the Anselm BAV text, the ratio of identical tokens before normalization is only 35% (= the baseline). This is an indicator that the Anselm texts differ significantly more from New High German than Luther's bible—at least in spelling. Consequently, the match ratio after normalization is also considerably lower (about 48%). While this is far from optimal, the increase is still notable. 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 *etleich – etliche* 'several', which are correctly normalized even though they do not appear in the training corpus, so the wordlist does not cover them. In fact the results of the wordlist-based approach are worse here (45%). Even though the difference is small, it is still significant ($p < 0.005$), showing that our method actually works better with this type of text.

The other, smaller Anselm fragments show a similar tendency, with the rule-based approach always being better than the wordlist. However, possibly due to their small size, the difference cannot shown to be significant here. The same is true for the difference between the rule-based and the combined approach, which cannot be proven to be significantly better even for the larger BAV text.

When assessing the results, one should bear in mind that we currently only count identical tokens. 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 *unsere*. 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 Anselm text, but only very rarely in the Luther bible: the appropriate rule (5) was learned exactly once from the training corpus.

(5) `v → 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 shows the worst performance in our evaluation (cf. Table 3).

To further assess the problem of unseen character-context combinations, we identified the number of tokens for each text for which this problem occured. Table 5 shows the number of tokens for which not a single normalized variant can be generated without, at some point during the process, encountering a situation with no applicable character edit rules. Dictionary restrictions were not considered this time. Again, the ECG text stands out as being the most problematic one. However, the evaluation also suggests that this aspect is not the only deciding factor for low performance: while BAV is by far the least problematic text in this regard, both WCG and ALEM achieve a better match ratio after normalization.

A curious fact is that the WCG text performs best, but on manual inspection does not look all that good. It seems that the good results are primarily achieved from the normalization of function words. It is debatable to which extent this is desirable. Function words are necessarily the most

frequent words in a text, and also the ones that structure the text logically. On the other hand, they are also a closed class of words, which might account for the fact that using a word substitution list achieves similar results. While the examples in Table 2 show that the rule-based method is capable of normalizing non-trivial content words, further research is needed to assess the proportion of function words in the correctly normalized word pairs.

6. Related Work

Baron et al. (2009) present two tools for normalization of historical wordforms. VARD consists of a lexicon and user-defined replacement rules, and offers an interface to edit and correct automatically normalized wordforms. DICER[7] is a tool that derives weighted context-sensitive character edit rules from normalized texts. The algorithm that creates these rules is not described in their paper, though.

Jurish (2010) compares different methods to normalize German wordforms from 1780–1880. The methods include mappings based on phonetic representations and manually created rewrite rules. The highest F-score (99.4%) is achieved by an HMM (Hidden Markov Model) that selects one of the candidates proposed by the other methods.

Research on normalizing historical language data has also been done in the field of Information Retrieval, applied to historical documents. They address the task reverse to ours: mapping modern wordforms to historical (or dialectal) wordforms. Rules are derived from word pairs that are generated by means of a spellchecker (Ernst-Gerlach and Fuhr, 2006) or a dictionary (Hauser and Schulz, 2007).

Our approach is similar to Ernst-Gerlach and Fuhr (2006) and Hauser and Schulz (2007) in that we derive replacement rules of character sequences from aligned pairs. The algorithms to learn rules differ, though. Ernst-Gerlach and Fuhr (2006) specify recursive definitions that take into account rewrite sequences and contexts to varying extents; rules can refer to characters or to the underspecified classes 'vowel' and 'consonant'. Hauser and Schulz (2007) extract n-gram sequences of varying size from the aligned wordforms, and learn n-gram mapping rules. Furthermore, our approach differs from theirs in that our training pairs stem from aligned corpora.

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, though in this case, it is significantly better than using wordlist substitution alone. It turns out that substantial variations in spelling, as they occur in the Anselm texts, are currently problematic for the rule-based approach, as it cannot normalize character-context sequences that have not been previously learned. However, a more fine-grained evaluation is required to assess the amount of partially correct normalizations, which are also useful but currently counted as total misses.

The Anselm data seems to suggest that we include more sophisticated similarity measures into our system, such as phonetic or even graphemic similarity. This could cover common substitutions like *tz* by *z*, which are phonetically similar in German, or *u* by *v*, which is a common graphemic variant 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.

Another issue would be to replace our simple heuristic of sequence determination by the use of an association measure such as the log-likelihood ratio or the Fisher-Yates test to determine which rules are merged to sequences. Association measures could be further used to determine the significance of the association between a rule and its context, and potentially abstracting the rules from their specific contexts.

[7] http://corpora.lancs.ac.uk/dicer/

References

Baron, A., Rayson, P., and Archer, D. (2009). Automatic standardization of spelling for historical text mining. In: *Proceedings of Digital Humanities 2009*, Maryland, USA.

Bollmann, M., Petran, F., and Dipper, S. (2011). Rule-Based Normalization of Historical Texts. In: *Proceedings of the International Workshop on Language Technologies for Digital Humanities and Cultural Heritage*, pp. 34–42, Hissar, Bulgaria.

Braune, F. and Fraser, A. (2010). Improved unsupervised sentence alignment for symmetrical and asymmetrical parallel corpora. In: *Proceedings of the 23rd International Conference on Computational Linguistics (COLING), Poster Volume*, pp. 81–89, Beijing, China.

Dipper, S. and Schrader, B. (2008). Computing distance and relatedness for medieval text variants from german. In: Storrer, A., Geyken, A., Siebert, A., and Würzner, K.M. (Eds.), *Text Resources and Lexical Knowledge. Selected Papers from the 9th Conference on Natural Language Processing (KONVENS-08)*, pp. 39–51. Mouton de Gruyter, Berlin.

Ernst-Gerlach, A. and Fuhr, N. (2006). Generating search term variants for text collections with historic spellings. In: Lalmas, M., MacFarlane, A., Rüger, S., Tombros, A., Tsikrika, T., and Yavlinsky, A. (Eds.), *Advances in Information Retrieval. Proceedings of the 28th European Conference on IR Research (ECIR 2006)*, pp. 49–60, Berlin. Springer.

Hauser, A.W. and Schulz, K.U. (2007). Unsupervised learning of edit distance weights for retrieving historical spelling variations. In: *Proceedings of the First Workshop on Finite-State Techniques and Approximate Search*, pp. 1–6, Borovets, Bulgaria.

Jurish, B. (2010). More than words: Using token context to improve canonicalization of historical German. *Journal for Language Technology and Computational Linguistics*, 25 (1), pp. 23–39.

Och, F.J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29 (1), pp. 19–51.