Exploring Norma for Historical Text Normalization

Tova Erbén
LT2314 Language Technology Resources
Gothenburg University
guserbto@student.gu.se

Abstract

This work aims to explore the Norma tool (Bollmann, 2012) for historical text normalization using substitution lists. The material used in this endeavour is the MAÞiR resource from Språkbanken, containing annotated excerpts from five different texts in Old Swedish (ca 1225-1526). The results from this experiment are less than good, which can partially be explained by the choice of source data for the lookup table, as well as the lack of other complementary methods used in the normalization process. Future work aims to dive deeper into Norma and other normalization tools.

1 Introduction

Historical texts that were written before any spelling standardization had taken place in their language of medium tend to contain a lot of spelling variation. This becomes problematic when you want to parse them for further processing or research them in a corpus. Luckily, there exists several different methods for normalizing such text, i.e. mapping the variants to a standard form.

Bollmann (2019) gives an overview and comparison of the following methods: 1) substitution lists, 2) rule-based methods, 3) distance-based methods, 4) statistical machine translation, and 5) neural machine translation.

Substitution lists (also known as lookup-based normalization) are the simplest out of these methods and consists of putting together a list of different variants for different words. Variants that don't make it to the list cannot be handled. The normalization tools VARD and Norma contain such substitution list components.

Rule-based methods instead try to pick up on regularities in the variations, such as the letter *v* often being written like *u*. Bollmann (2012) describe the rule-based re-write rules as operating on one or more characters and only taking the immediate character context into consideration. If more than one rule is applicable, then you choose the one with the highest frequency in the training data.

The distance-based methods look at the number of edits needed to go from a variant spelling to a standard form, often using the Levenshtein distance. Adesam et al. (2012) combine the distance- and rule-based approach by using the Levenshtein distance to derive rewrite rules for Old Swedish, which perform better when compared to manually compiled rewrite rules from dictionaries.

The statistical machine translation-based methods look to optimize the probability for a spelling variant *v* being of standard form *w* by looking, for instance, at character-based statistics (cSMT). Pettersson et al. (2013) manage to increase the normalization accuracy for Swedish from 64.6 to 92.3 % by applying cSMT to historical texts from the time period 1527–1812.

A recent data-driven approach for spelling normalization is neural machine translation (NMT), commonly using encoder-decoder models with LSTM units (Bollmann, 2019). However, when Bollmann (2019) compares some character-based NMT models to the previously mentioned methods, he notes that these models need more data in order to perform well. Instead, he recommends using the Norma tool in the "combined" setting (more about this later) for languages with little training data, and the character-based statistical machine translation tool cSMTiser otherwise.

In this work, I will experiment with normalization on Old Swedish texts using substitution lists and the Norma tool (Bollmann, 2012). All the code and generated data can be found on GitHub.[1]

[1] https://github.com/datatjej/langtechres-project

2 Material and Software

2.1 MAÞiR

The material that I will be using in this study comes from the MAÞiR resource at Språkbanken, consisting of annotated excerpts from five different Old Swedish texts:

1. *Abota*
    - ca 1448
    - 540 tokens
2. *Avgl*
    - Elder Westrogothic law
    - ca 1220/1280
    - 1228 tokens
3. *Moses-b*
    - Paraphrases of the five books of Moses
    - ca 1330/1526
    - 7049 tokens
4. *Ogl-a*
    - Ostrogothic law
    - ca 1290/1350
    - 22090 tokens
5. *Tungulus*
    - 1457
    - 2814 tokens

These texts have been annotated word-by-word in XML format with information about syntactic relations, part of speech tags, morphology, lemma and head. This part of the resource is called MaÞiR Trees. The other one, MaÞiR Words, contains 10 TSV ('tab-separated values') files with lemmata information extracted from a reference dictionary for Medieval Swedish by Söderwall (1891–1918) (Sdw).

Half of the files have four columns with information about lemma (which may contain spaces, and in the case of the verb file, a '+' sign separating the verb from a separable participle, e.g. EX), pseudolemma (containing only the verbal head in the case of verbs with separable participles), coarse-grained POS tag in accordance with Sdw, and finally a fine-grained POS tag. The other five files contain an additional column with parenthetical form, which show inflectional and spelling-related variation, also taken from Sdw.

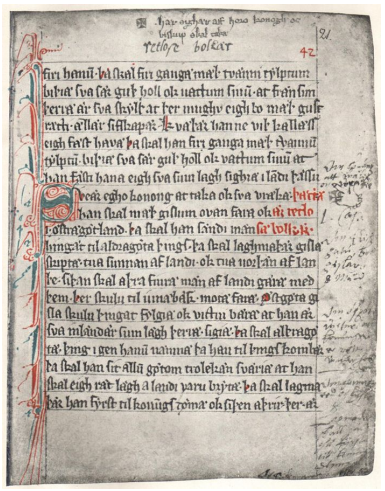

Figure 1: Page from the Elder Westrogothic law.

2.2 Norma

Norma[2] (Bollmann, 2012) is an automatic normalization tool that contains functionality for rule-based, distance-based and lookup up-based normalization, as well as the option of running all these three in combination. When running the program, you have to provide it with a configuration file that specifies which normalizer(s) to use, and in which order.

When the program reads the first historical input word to be normalized (given in an input text file with one word per line), it proceeds to find a normalized candidate through the first normalizer specified in the configuration file. If the candidate word is above a certain confidence score threshold (between 0 and 1) defined by the user, it moves on to the next step of validating the candidate against the correct word form and starting the training function which takes the historical spelling and the standardized word form as input and adjusts the parameters of the normalizer. If the first normalizer does not yield a high enough confidence score, it moves on to the next normalizer. If none of the specified normalizers give a high enough score, the word remains unchanged.

Norma has three modes: interactive mode, training mode and batch mode. The interactive mode is supposed to allow the user to confirm or correct the normalization candidate provided by Norma themself, but is currently unavailable in the latest release of Norma due to technical issues. The batch mode is the default mode which takes the configuration file and text file of words to normalize as input and outputs a list of normalized words, one token per

[2] https://github.com/comphist/norma

line together with a confidence score. The training mode allows you to provide manually normalized data in the form of a list with two tokens per line: historical form and the normalized form.

According to the user guide[3], "most normalizers" require a target lexicon that defines valid output strings in order to avoid generating non-existing words. This, naturally, does not seem to be the case for the Mapper normalizer, as it only normalizes words to forms that exist in the substitution list.

3 Method

For preprocessing the annotated MAÞiR files I wrote a Python script that takes the path to the XML files as argument and then loops them through in order to extract tokens and their lemma. These are saved in two files:

1. `mathir_tokens.txt`, which lists all tokens (including duplicates) from all files in lower case and one token per line:

   hær
   sigx
   aff
   abotum
   allum
   ...

2. `mathir_train.txt`, which lists all tokens and their corresponding lemma, one tab-separated token-lemma pair per line:

hær	här
sigx	sighia
aff	af
abotum	abbote
allum	alder
...	...

This latter file is only used for calculating the baseline accuracy, i.e. the number of tokens that already match their corresponding lemma.

As basis for a subsitution list, I instead use five of the files in MAÞiR Words which contain parenthetical form or variant spelling of the dictionary lemmas. Therefore I wrote another script that takes the MAÞiR Words directory as input and loops through these files, saving the parenthetical form and corresponding lemma in another text file called `sdw_train.txt`:

...	...
lach	lagh
lagha	lagh
laghä	lagh
lak	lagh
laugh	lagh
...	...

When running this file in train mode in Norma, a parameter file called `sdw_train.Mapper.mapfile` is automatically generated. In those cases where the training data is not coming from dictionary entries (like here), but instead generated from real data like the `mathir_train.txt` file, this file will contain frequencies of each variant-lemma pair, which will then help Norma determine which lemma to choose when doing the lookup.

For normalizing the tokens, I simply ran the `mathir_tokens.txt` file in batch mode and saved the output to a text file by adding `>norma_output.txt` to the Norma command.

For evaluating the output file, I wrote a script that takes the token file and output file as arguments and counts the number of matches between these two documents. The accuracy is calculated by simply dividing the total number of matches with the total number of tokens.

4 Results

Normalizer	Accuracy
Baseline (unchanged)	27.30 %
Norma (lookup)	34.14 %

Table 1: Accuracy for baseline (leaving all tokens unchanged) and lookup-based normalization in Norma.

The lookup-based normalization in Norma yielded a lemmatization accuracy of 34.14 %, which can be compared to the baseline of 27.30 %. I would have wanted to try out the weighted Levenshtein distance-based normalization in Norma as well, but unfortunately the command for generating the necessary target dictionary files was not working in Docker.See Table 2 for some examples of successful and unsuccessful lookup matches.

[3] https://github.com/comphist/norma/blob/master/doc/UserGuide.md

Matches		Failed Matches		
input	**output**	**input**	**output**	**correct**
...		...		
oc	ok	wt	wt	ut
uili	vilia	ær	ær	vara
scal	skula	thet	tea	þän
med	mäþ	æter	æter	äta
þessi	þänne	þa	þiggia	þa
engelin	ängeli	bætri	bætri	bätre
bönder	bonde	sigher	sigha	sighia
wordo	varþa	wigyæ	wigyæ	vighia
kyrkiu	kirkia	guðfæþur	guðfæþur	guþfaþir
landbor	landboe	for budhit	for budhit	forbiuþa
himmerike	himirike	wtan widh	wtan widh	utan viþer
...		...		

Table 2: Some successful and unsuccessful lookups.

5 Discussion

The accuracy for the lookup-based normalization may seem very low at 34.14 %, but the results are not surprising given the source of the lookup table. The Sdw dictionary entries and their associated parenthetical forms will often not include inflected or conjugated forms, and if they do (as in the case of some verbs and nouns) it's purely by chance. Some of these by-chance successful normalizations include plural forms such *landbor* (normalized to *landboe*) and conjugated verbs like *wordo* (normalized to *varþa*).

Most of the successful normalization can be explained by the fact that 27.30 % of the tokens in the input data are already identical to their lemma forms and that Norma does not modify tokens that are missing in the lookup table. This number would probably increase somewhat with some simple rule-based procedures like changing the letter *w* to *u* (compare *wt* and *ut* in Table 2) and *æ* to *ä* or *e*.

However, this does not mean that lookup-based normalization is useless - quite the opposite. Bollmann (2019) shows that lookup-based normalization can reach accuracy scores of more than 90 % for historical texts in in English, Spanish and Slovenian. Bollmann's comparison also illustrates that Norma in "combined" mode – using rule-based, distance-based and lookup-based normalization in combination – yields better results than any of these tools used in isolation.

6 Conclusion and further work

This exploratory study opens up many doors for future research. It would be interesting to see how rule-based and distance-based normalization in Norma would affect the accuracy, and I hope to try that once I figure out the technical issue I had with generating the necessary target lexicons.

The accuracy might be low, but for me, this project highly contributed to the expected learning outcome of the course by familiarizing me with two different language technology resources: Norma and MAÞiR. The process of figuring out how to install Norma was time-consuming and frustrating, but it lead me to discover Docker, a platform that allows you to install programs like Norma without worrying about dependencies, since all the softwares needed get packed in so called containers.

The installation of Docker was also time-consuming and included removing my current Windows Subsystem for Linux (WSL 1) installation to upgrade to the 2nd version and losing all previously installed packages. Once that was done, though, Norma could easily be installed using a single terminal command. And I recently noticed that cSMTiser, the software for statistical machine translation that I originally wanted to use for this project (but gave up on because of the equally tedious installation process) is also available via Docker.

On a finishing note, this project also introduced me to report writing in LaTeX and Overleaf, and I don't think I will ever go back to Word or Google Drive documents.

References

Yvonne Adesam, Malin Ahlberg, and Gerlof Bouma. 2012. bokstaffua, bokstaffwa, bokstafwa, bokstaua, bokstawa... Towards lexical link-up for a corpus of Old Swedish. In *Proceedings of KONVENS 2012 (LThist 2012 workshop)*, pages 365–369.

Marcel Bollmann. 2012. (Semi-)Automatic Normalization of Historical Texts using Distance Measures and the Norma tool. In *Proceedings of the Second Workshop on Annotation of Corpora for Research in the Humanities (ACRH-2)*, pages 3–14, Lisbon, Portugal.

Marcel Bollmann. 2019. A Large-Scale Comparison of Historical Text Normalization Systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3885–3898, Minneapolis, Minnesota. Association for Computational Linguistics.

Eva Pettersson, Beáta Megyesi, and Jörg Tiedemann. 2013. An SMT Approach to Automatic Annotation of Historical Text. In *Proceedings of the Workshop on Computational Historical Linguistics at NODALIDA 2013*, volume NEALT Proceedings Series 18, pages 54–69. Linköping Electronic Conference Proceedings 87.

Knut Söderwall. 1891–1918. *Ordbok öfver svenska medeltids-språket*, volume Series 1, Svenska skrifter 27. Svenska fornskriftsällskapet.