Programming Project: Historical Text Normalization with Synthetic Data

Benjamin Suter
Institut für Informatik
Universität Zürich
benjamin.suter@uzh.ch

Abstract

This paper describes the method used to normalize Early New High German letters collected by the Zurich protestant reformer Heinrich Bullinger (1504–1575). This project is a part of the *Bullinger Digital* research project at the University of Zurich.

The normalization models are trained on purely synthetic parallel data. The method requires a list of possible sub-string correspondences between the source and target side, but no manually annotated parallel samples. The method is not limited to a specific language or historical period and can be applied to any text normalization task.

The normalization models described here are neural sequence-to-sequence models (transformers). They can make use of the full sentence context when normalizing a specific token, which improves the quality for tokens that are ambiguous in isolation. On a manually prepared test set of 1201 tokens, the main model has a word error rate of 14.2%, and a lemma error rate of 3.7%.

1 Introduction

Text normalization is the task of mapping inconsistently spelled texts to a standardized form. In case of historical text normalization, the target for each historical word is typically the closest corresponding word in the modern standard language.

The present project has the goal to normalize Early New High German (ENHG) letters collected by Heinrich Bullinger (1504–1575), a collaborator and successor of the Zurich Protestant reformer Huldrych Zwingli (1484–1531). Bullinger was not only a prolific letter writer, but he also collected more than 10 000 letters that he received (and some that he wrote himself). The letters are written mostly in Medieval Latin, but partly also in ENHG. This project focuses on the ENHG part of the corpus.

ENHG is the predecessor of New High German (NHG). While the two variants of German are relatively close to each other linguistically, ENHG spelling differs greatly from the NHG orthography, and due to the lack of a standardized orthography in ENHG, there is a lot of spelling variation, and every writer uses his own idiosyncratic spelling conventions.

Not all differences between ENHG and NHG are due to spelling, though. Some differences in writing do reflect actual differences in pronunciation. This is true in particular for vowels. For instance, some vowels have not yet consistently undergone the Early New High German diphthongization; e.g., ENHG *lüdt* [lyˑt] vs. NHG *Leute* [ˈlɔɪ̯tə] (en. *people*). In addition, there are some morphological suffixes that differ between the two language variants; e.g., ENHG 3rd pl. *wöllind* vs. NHG *wollen* (en. *[they] want*).

Normalization not only facilitates readability for human readers, it moreover greatly improves searchability of words and phrases in a text corpus, as well as downstream processing with standard NLP tools.

The normalization task is defined here as a strict one-to-one mapping of words to their modern spelling. The goal is explicitly not to translate ENHG texts into contemporary German. This means that syntax remains unchanged, and outdated words are not mapped to contemporary synonyms. This text normalization project is part of the *Bullinger Digital* research project at the University of Zurich[1]. The models and training scripts are available on GitHub.[2]

[1] http://www.bullinger-digital.ch/
[2] https://github.com/besou/text_normalization

2 Method

2.1 Model

The normalization model is a standard transformer model (Vaswani et al. 2017), trained with the `fairseq` framework (Ott et al. 2019). Both encoder and decoder consist of 6 layers with an embedding dimension of 512, a feed-forward network embedding dimension of 2048, and 8 attention heads. ReLU is used as the activation function, and dropout is used with a rate of 0.3. The model has 44 707 840 trainable parameters and shared embeddings.

The model is trained for a maximum of 10 epochs with optional early stopping. It uses Adam as the optimizer and label-smoothed cross entropy with a label smoothing rate of 0.1. It has a learning rate of 0.001, inverse square root as a learning rate scheduler, and parameter updates after every 16 batches.

The raw text data is subword-encoded with a `sentencepiece` (Kudo and Richardson 2018) unigram model with a relatively small vocabulary size of 1000 types.

2.2 Dataset

A challenge for many text normalization task is that manual preparation of a sufficently large parallel corpus that is suitable for the task at hand is time-consuming and expensive. This project goes in a different direction and does completely without manual sample annotations.

Instead, a synthetic source-side (here: ENHG) corpus is generated based on an arbitrary target-side (here: NHG) corpus. This is done by means of a list of possible substitutions on the level of short character sequences. Source sequences are then generated from target sequences by applying randomly selected substitutions from the list. A toy example can be found in Table 1.

target-side	source-side substitutions
b	*p, bp, pp*
r	*rr, rh*
ei	*i, e, ey, ay, eih, ej*
t	*d, dt, tt, th*

Table 1: Three examples for possible substitutions of target-side *b*, *r*, *ei*, *t*. Given the target-side word *bereit* (en. *ready*), some possibe source-side variants that may be generated are: *berrejtt*, *bereydt*, *perayth*.

It is often the case that for the same letter or letter group different spelling variations exist depending on the position inside a word or syllable. For instance, while gemination of consonants (e.g., *t* → *tt*) appears frequently at the end of a syllable, it is rare at the beginning of a syllable. Therefore, in the present project, the target-side corpus is first syllable-tokenized with `pyphen`[3], and then different substitutions are applied to the onset, nucleus, and coda of each syllable separately.

Different from manual annotation of samples from the historical text domain, the approach presented here is based solely on an arbitrary corpus containing sentences in the target language. The resulting dataset does not contain any real sample from the historical source texts. However, if such data is available, it can simply be added to the synthetic parallel corpus.

While no manual annotation of samples is needed, the creation of a list of spelling correspondences still requires some degree of observation and/or intuition about what spellings may occur in the attested historical texts (here: ENHG). Nevertheless, the approach has some advantages over manually annotated data.

Most notably, the size and content of the resulting parallel corpus are completely independent from the list of spelling correspondences between the source and target languages. Once the substitution table is sufficiently developed, any suitable monolingual data can be used to create an arbitrarily large and arbitrarily domain-specific parallel corpus with no additional effort. In addition, an existing table of correspondence may be reused with some modifications to quickly generate parallel data for a new text normalization project.

An important observation is that overgeneration of non-attested spellings does not harm the model. It is enough to ensure that (ideally) all actually attested spellings occur in the source corpus as possible spelling variants.

In order to normalize ENHG texts, the target-side corpus can be any collection of text in standard German, but ideally the corpus contains texts from a similar domain. The corpus used for the normalization of the Bullinger letters consists of two parts: The major part is the subcorpus of the *Deutsches Textarchiv* of the period 1900–1999[4] with a size of 83.9 MB. This corpus was chosen be-

[3] `https://pyphen.org/`

[4] `https://www.deutschestextarchiv.de/download`

NHG	Dabei hege ich keinerlei Zweifel an Deiner Umgänglichkeit und Freundlichkeit .
synth	dabei hege ich keinerrly zweyfel an deinner umgängligkeit und freuntlichkeith .
NHG	Ach wäre doch jemand , der dieses Problem behandelt !
synth	ach wäry duuch jemaundd , der dieses probleemb pehandehlt !

Table 2: Two examples of synthetically generated source sequences.

cause it contains novels and letters from a period that is already NHG, but still somewhat conservative in style. The smaller part of the corpus is a project-internal collection of translations and summaries (regests) of Bullinger letters. It has a size of only 5.4 MB, but is helpful because it contains data from the relevant domain.

The full corpus of 89.4 MB was slightly preprocessed and cleaned. For instance, the *Deutsches Text Archiv* subcorpus contains some outdated spellings like *th* instead of contemporary *t* that were still common in German spelling in the early 20th century (e.g. *Theil* → *Teil*, en. *part*). This was cleaned using regular expressions. In addition, the corpus was tokenized using the NLTK tokenizer (Loper and Bird 2002).

Table 2 shows two examples of synthetically generated source sequences that are part of the training set. As a kind of data augmentation, each sample from the original corpus is used five times with different random substitutions during the creation of the synthetic parallel corpus. Since capitalization is used rather inconsistently in ENHG, the source-side corpus is lowercase-only. The normalization model thus includes prediction of capital letters as part of the normalization.

The model is trained on full sequences in order to make use of the full context. All samples in the training set have an equal number of tokens on the source and target side, which allows for a simple one-to-one alignment between the original sequence and the normalized sequence. However, the model is not guaranteed to produce an equal number of tokens. Therefore, a second model was trained on a dataset which contains a single token with a fixed context window on the source side, and a single output token on the target side. This model can be applied to a sequence word by word and therefore guarantees a one-to-one alignment at the cost of less context. For the normalization of the Bullinger model, this second model is used as a fallback model in cases where the main model fails to produce a token sequence of the same length as the input.

3 Results

On a manually prepared test set of 1201 tokens, the normalization model achieves a word error rate of 14.2%. This is comparable in quality to other state-of-the-art models for historical text normalization (Makarov and Clematide 2020). Table 3 shows two example outputs of the main model.

Most errors are caused by minor spelling variations (capitalization or no capitalization, word-final unstressed *e* present or missing, etc.). A manual inspection showed that that the lemma error rate (i.e. the rate of tokens that are normalized to another lemma than the one in the reference normalization) is only 3.7%. An example of a lemma error is ENHG *[doch ist by uns der] bruch*, which the model normalizes to *Bruch* (en. *breaking*) instead of *Brauch* (en. *custom*).

Although the main model does not guarantee a one-to-one alignment between the input and the output, the model has a strong bias to favor output sequences with the same number of tokens as the input. In a test set of 2333 samples, the main model produced a correct one-to-one alignment in 95.5% of all cases. It may be noted that this is not trivial since the input is subword-segmented, and the number of subword tokens can differ between the input and an output with the correct number of (full-word) tokens.

There are two main causes for the main model to fail to produce a one-to-one alignment: The most common one are overlong input sequences. In this case, the model simply stops the output sequence too early, and the last part of the sequence is completely missing. The other cause of failure is when two or more adjacent words look similar, which accidentally leads the model to skip one of them. An example of the latter is ENHG *also wil ich üch ouch thun […]*, for which the model produces *also will ich Euch tun*, skipping the token *ouch* (*auch*, en. *too*) that directly follows the similarly looking tokens *ich üch*. In these cases, the distance in length between the input and the output sequence is typically exactly one.

ENHG	Dann er gar ein eerlicher , wolhabender iüngling ist und noch vast iung .
hyp	Dann er gar ein ehrlicher , wohlhabender Jüngling ist und noch fast jung .
ref	Denn er gar ein ehrlicher , wohlhabender Jüngling ist und noch fast jung .
ENHG	Mitt dem verschwund das kind , das der wächter nütt wusst , wohin es kommen were .
hyp	Mit dem Verschwinden das Kind , das der Wächter nichts wusste , wohin es kommen wäre .
ref	Mit dem verschwand das Kind , dass der Wächter nicht wusste , wohin es gekommen wäre .

Table 3: Two examples of model output and reference normalization.

4 Conclusion

This paper described the method used to normalize Early New High German letters collected by the Zurich protestant reformer Heinrich Bullinger (1504–1575) as part of the *Bullinger Digital* research project at the University of Zurich.

The model used in this project is a transformer model trained on full sentences, which allows the model to make use of the full context when normalizing each word.

No manually annotated parallel data is used in the training data. Instead, synthetic parallel data is generated based on a monolingual corpus in standard language and a table with possible spelling correspondences between the source and the target language.

On a manually prepared test set of 1201 tokens, the normalization model achieves a word error rate of 14.2%, which is comparable in quality to other state-of-the-art models for historical text normalization. Most erroneous tokens do still contain the correct lemma. Wrong lemmas are produced in only 3.7% of all tokens.

There are two directions for future work. First, it would be worthwhile to apply the proposed method of using synthetic parallel data to previously used historical data and compare the results with other text normalization methods. Secondly, it might be interesting to explore possibilities to automate the generation of a table with spelling correspondences between the source and target languages, as this is the most laborious and critical task in the presented approach.

References

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Edward Loper and Steven Bird. 2002. NLTK: The natural language toolkit. In *Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*.

Peter Makarov and Simon Clematide. 2020. Semi-supervised contextual historical text normalization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*.