Automatic Normalisation of Historical Text

Alexander Robertson

Master of Science by Research
Centre for Doctoral Training in Data Science
School of Informatics
University of Edinburgh
2017

Abstract

Spelling variation in historical text negatively impacts the performance of natural language processing techniques, so normalisation is an important pre-processing step. Current methods fall some way short of perfect accuracy, often requiring large amounts of training data to be effective, and are rarely evaluated against a wide range of historical sources. This thesis evaluates three models: a Hidden Markov Model, which has not been previously used for historical text normalisation; a soft attention Neural Network model, which has previously only been evaluated on a single German dataset; and a hard attention Neural Network model, which is adapted from work on morphological inflection and applied here to historical text normalisation for the first time. Each is evaluated against multiple datasets taken from prior work on historical text normalisation. This facilitates direct comparison of this work to that existing work. The hard attention Neural Network model achieves state-of-the-art normalisation accuracy in all datasets, even when the volume of training data is restricted. This work will be of particular interest to researchers working with noisy historical data which they would like to explore using modern computational techniques.

Acknowledgements

First and foremost I am grateful to my primary supervisor, who knew when to nudge me in a sensible direction and when to just push. Without that expert guidance, this thesis would be ninety pages of trying to improve the accuracy of my Hidden Markov models.

This work was supported in part by the EPSRC Centre for Doctoral Training in Data Science, funded by the UK Engineering and Physical Sciences Research Council (grant EP/L016427/1) and the University of Edinburgh.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(Alexander Robertson)

Table of Contents

1	Introduction	1
1.1	The historical spelling variation problem	1
1.2	The elements of the problem	3
1.3	Contributions	3
1.4	Outline	4
2	Background	5
2.1	Widely applied approaches	5
2.1.1	Manual normalisation	5
2.1.2	Dictionary lookup	6
2.1.3	Rule-based transformation	7
2.2	Approaches not yet commonly applied	8
2.2.1	Statistical and Neural Machine Translation	8
2.2.2	Structural decomposition	8
2.3	Evaluation	9
2.3.1	Analysis	9
3	Historical Datasets	11
3.1	Preprocessing	12
3.2	Modern language resources and baselines	14
3.3	Baselines per language	15
3.4	Descriptive statistics	16
3.5	Edit distance between historical and modern strings	17
3.6	Predictions	20
4	Hidden Markov Models	21
4.1	Components of an HMM	21
4.2	Relating HMMs to historical spelling variation	22

4.3	Training an HMM	23
4.4	Potential issues for HMMs	23
4.4.1	Observation sequence structure	23
4.4.2	Differences between train and test observation sequences .	24
4.4.3	Model assumptions	25
4.4.4	The problem of “best path” Viterbi	26
4.5	Experiments	27
4.5.1	Training, testing and development subsets	27
4.5.2	Model outlines	27
4.5.3	Model evaluation	28
4.6	Results	29
4.6.1	Standard models	29
4.6.2	Lexical filter	30
4.6.3	Reranking	31
4.6.4	Volume of training data	33
4.7	Summary	34
5	Neural Network Models	36
5.1	Neural networks for sequence labelling	36
5.1.1	Application to historical spelling variation	37
5.1.2	Shortcomings of the encoder-decoder work	38
5.2	Drawing parallels with morphology	39
5.2.1	The hard monotonic attention model	40
5.2.2	Applying hard monotonic attention to historical spelling variation	41
5.3	Experiments	42
5.4	Results and comparisons	42
5.5	Summary	43
6	Comparison of models	45
6.1	Qualitative analysis	45
6.2	Quantitative analysis	46
6.3	Future work	48
6.4	Conclusion	50
	Bibliography	52

Chapter 1

Introduction

This work applies new technologies to an old problem in natural language processing, one that is caused by even older sources of data: the historical spelling variation problem. A variety of probabilistic and statistical models are evaluated. These are motivated by the findings of a detailed investigation of the differences between historical and modern text.

1.1 The historical spelling variation problem

Researchers in any area of the humanities have a multitude of sources at their disposal. The internet has made enormous volumes of new data available, either through creation of new resources (e.g. Twitter) or by making old ones more widely available (e.g. Google Books). Natural language processing (NLP) has aided in extracting useful information from these resources, at scale and at speed. However, the digitisation of more and more old resources presents both problems and opportunities.

One particular problem is variation. Whilst modern texts can vary in terms of content, style and purpose, historical texts exhibit variation at other levels. Language is not static and its usage changes over time. Syntactic change results in variation in word order at the sentence level. Semantic change shrinks or grows the number of senses per lexical item. Morphological change creates and removes affixes, resulting in variation at the morpheme level. Of special importance to NLP is the issue of orthographic variation. The notion that every lexical item has a fixed orthographic representation is not one that exists at every stage of a language's development, as illustrated for the modern word form *bishopric* in

figure 1.1.

Figure 1.1 – Examples of variation in historical English

Spelling variation raises two issues when attempting to apply NLP techniques to historical texts. Consider the task of part-of-speech tagging. First, models pre-trained on modern text will perform poorly when used with historical text due to the especially large number of unseen vocabulary items. The Paston letter above is a clear example of this. Second, models trained on historical text will similarly result in many items tagged as unknown, but much of the statistical information extracted from the data will simply be incorrect. When there are many orthographic forms for a particular word, it is no longer possible to calculate simple statistics such as word frequency without also knowing how forms map to words. Historical texts, therefore, appear to have much larger vocabularies than is actually the case.

It must be noted that historical spelling variation is not the same as modern spelling errors. An author in the 15th century was not a clumsy typist; the modern day concept of standard spelling simply did not exist. But spelling variation was not wild and unconstrained. The written word met communicative needs and served as a representative encoding of spoken language. Even today we can read the letters of Paston and Cromwell and know what is meant by *seyd* and *adversarie*. It is exactly this which makes historical spelling variation a fascinating topic for research: knowing that people both today can and five hundred years ago

could make these connections between what is written and what is meant, how can we train machines to do the same? If we achieve this goal, we can leverage NLP for the benefit of scholars in other areas and extract useful information from historical resources as easily as from modern ones.

1.2 The elements of the problem

With regard to NLP, the historical spelling variation problem is comprised of two elements. The first is identification. Given a suitably tokenised historical text, how can we know which tokens are orthographic variants of each other? The second is normalisation. Given a list of known orthographic variants, how can we map these to their fixed modern equivalents? Normalisation may seem dependent upon identification but in practise they can be decoupled. Either every token is treated as a variant and an attempt is made to normalise it, or the input is assumed to have already been suitably filtered so as to leave only variants. The normalisation process is the focus of this thesis.

1.3 Contributions

This thesis builds upon existing theoretical and practical work on historical text normalisation by:

- undertaking a thorough investigation of the relation between historical and modern texts (chapter 3);
- using the results of that investigation to construct a well-motivated Hidden Markov model and evaluate it against multiple datasets (chapter 4);
- evaluating an existing neural network model against those same datasets for the first time (section 5.1);
- adapting and evaluating a neural network model recently used in morphological inflection, directly motivated by the relation between historical and modern text (section 5.2);

This final, well-motivated model outperforms all others by as much as 4%. This represents the current state of the art in historical text normalisation. Performance is maintained even when trained on as little as 50% of the available

training data, and remains competitive with as little as 10%. This holds true even for languages which have traditionally been difficult to normalise, such as Icelandic, and for small datasets, such as the Swedish corpus used in this work. These findings are bolstered by evaluating models on exactly the same datasets as many other methods reported in the literature.

1.4 Outline

Following an overview of popular approaches to historical text normalisation, I describe in some detail the datasets that will be used to evaluate the models I build. This makes clear what work the normalisation task actually requires a machine to do. Hidden Markov and neural network models are evaluated in turn and then compared. I conclude with a discussion of possible directions in which future work may head.

Chapter 2

Background

This chapter is based on an essay written for the Topics in Natural Language Processing¹ course.

The approaches to historical text normalisation described here are separated into two classes. The first (rule-based and dictionary-based systems) are commonplace, having been used extensively in real world applications. They are implemented in a variety of software packages such as VARD2² (Baron and Rayson, 2008) and Norma³ (Bollmann, 2012), which allow users to automatically normalise historical texts. The second class are more recently developed techniques, relying on a variety of statistical approaches, which are yet to be incorporated into such tools.

A third class consisting of neural networks is not included here, but is instead presented alongside experimental work in chapter 5. The rationale for the separation is that these neural models will be extended and adapted as part of this thesis.

2.1 Widely applied approaches

2.1.1 Manual normalisation

For decades following the introduction of electronic corpora, the only way to address historical spelling variation was to manually check each word individually — the same situation as before electronic corpora. The datasets used in this work

¹<http://www.inf.ed.ac.uk/teaching/courses/tnlp/>

²<http://ucrel.lancs.ac.uk/var2/about/>

³<https://www.linguistics.rub.de/comphist/resources/norma/>

were created this way. Though skilled annotators can achieve very high accuracy of normalisation, they are not likely to be available in significant numbers. This is especially true when working with source documents which require special training to read. And despite the potential for high accuracy, errors are unavoidable in practice.

There is also the issue that normalisation is not a process with a single defined outcome. Eisenstein (2013), working with noisy social media text, points out that normalisation decisions often seem “little more amenable to automated parsing and information extraction than the original text” because there is a tendency to both not go far enough (e.g. not expanding *wtf* when it is used to abbreviate syntactic constituents as in “wtf is the matter with you?”) as well as to go too far (should *bro* really be normalised to *brother*?). The same situation is found in historical texts, where the reason for normalisation dictates its scope; vocabulary and syntax may also end up normalised. An example is the Queen Elizabeth I Corpus, a collection of Elizabeth’s correspondence. Evans (2011) describes some issues facing researchers in historical sociolinguistics, in particular the decisions that must be made regarding the normalisation process.

Detailed examples of problems with manual normalisation are found in chapter 3, where a corpus of English is closely examined.

2.1.2 Dictionary lookup

A manually normalised text can be used to bootstrap the normalisation of other texts. A correspondence dictionary can be extracted, mapping historical word forms to modern ones. This can then be applied to a new text, saving time and effort. It should be noted that dictionary lookup is also commonly applied when normalising modern text that displays a significant degree of idiolectal and sociolinguistic variation, but techniques there are much more sophisticated and often unsupervised — a good example of this contrast is found in Han et al. (2012).

Such dictionaries are generally not transferable to many other texts, due to differences in genre and era: a dictionary created out of Old English sagas is unlikely to be of much use in normalising Early Modern English medical texts. An example of this is found in Rocio et al. (2003), who used a general dictionary of medieval Portuguese to pre-process text from that era, after which it was

syntactically parsed with greater accuracy.

2.1.3 Rule-based transformation

The majority of historical word forms seem to share many similarities with their modern counterparts. Often only a single letter is added or subtracted. Changes occur in predictable locations, commonly at the end of a word. Consonants are often doubled. Rule-based transformation methods attempt to capture the regularity of these similarities and apply them in the style of the rewrite rules which have been used to describe phonological processes.

These rules can be taken at face value from scholarly work on historical spelling. Works like Fisher (1977) catalogue rules such as $u \rightarrow v/\#_n$. Such a rule replaces historical *u* with modern *v* when it appears at the start of a word before *n*. Or they can be extracted automatically from annotated data, where pairs of equivalent historical/modern word forms are available. Bollmann et al. (2011) used the Levenshtein algorithm (Levenshtein, 1966) to determine the minimum number of edit operations (deletions, substitutions, insertions) required to transform each historical word form into its modern equivalent. The context of these operations (i.e. the characters to the left and right in the historical word) were also recorded. Edit operations and context taken together constitute a rewrite rule. Each rule is assigned a probability (its frequency out of all rules) and when all available rules are applied to an historical word form, all possible outputs are scored as the product of the probability of the rules involved, normalised by the length of the input. To prevent over-generation of normalisation candidates, the list of outputs can be restricted to those in a list of words deemed acceptable, such as a modern lexicon.

An extension of the above, tested on Swedish by Pettersson et al. (2013a), takes an unsupervised approach. Historical words are pairwise compared with the words in a modern lexicon. Modern words within a predetermined Levenshtein distance are used as candidates for extracting rewrite rules. Furthermore, the individual edit operations extracted by the Levenshtein algorithm are weighted by a factor equal to the number of times the left hand side of the rule was not changed, divided by the number of all rules with the same left hand side. A similar unsupervised method for learning the actual character edit weights, as opposed to edit rule weights, is found in Hauser and Schulz (2007). For a thorough evaluation

of alternative methods for aligning strings, focusing on the Levenshtein algorithm but also looking at Pair Hidden Markov Models, see Wieling et al. (2009).

2.2 Approaches not yet commonly applied

2.2.1 Statistical and Neural Machine Translation

Viewing the normalisation of historical spelling variation as a translation task, Pettersson et al. (2013b) used an off-the-shelf statistical machine translation (SMT) package to process parallel historical/modern texts, in either Icelandic or Swedish, just as one would process a pair of French and German documents. The SMT approach models $P(\text{modern} \mid \text{historical})$ by splitting it up into the product of $P(\text{modern})$ and $P(\text{historical} \mid \text{modern})$. The first of these is estimated from the parallel text, with each historical/modern pair aligned at the character level, and the second from a source of modern text, using the Moses⁴ package. A similar character-based SMT approach was taken by Scherrer and Erjavec (2013) for Slovene.

A neural network-based version of machine translation, Neural Machine Translation (NMT), was taken by Korchagina (2017), using a system based on convolutional neural networks outlined in Lee et al. (2016). The focus was on historical German and Swiss German texts.

2.2.2 Structural decomposition

The REBELS (regularities-based embeddings of language structures) system (Mitantkin et al., 2014) modifies the translation model of SMT. Pairs of historical/modern word pairs are recursively decomposed into hierarchical subunits, which are then mapped between each other. For example, one possible level of decomposition of (knoweth, knows) will map *kn* to *kn* and *oweth* to *ows*. This is based on the assumption that it is by “distinctive infixes” (Sariev et al., 2014) that historical words are transformed into their modern counterparts. In the learning stage of the REBELS process, statistics are gathered over which historical infixes match historical ones. In the search stage, the most common infixes (relative to the previously gathered statistics) found in an historical word are used to find a matching hierarchy of modern infixes and the modern word from which they

⁴<http://www.statmt.org/moses/>

were generated. Supervised and unsupervised variants of REBELS differ in how the word pairs for the learning stage are generated. In the unsupervised case, candidate modern analogues are approximated by minimising the Levenshtein distance between each historical word form and those in a modern lexicon.

2.3 Evaluation

	LC-ICAMET (English)	GerManC (German)	IcePaHC (Icelandic)	GaW (Swedish)	LemmData + GerManC (Swiss, German)	Depositions (English)	IMP (Slovene)
Baseline	75.8	90.4	47.3	57.9	Not given	75.6	48.3
Rule-based ¹	82.9	87.3	67.3	79.4			
Dictionary lookup ¹	91.7	94.6	81.7	86.2			
Rule-based + dictionary ¹	92.9	95.1	84.6	90.8			
SMT	94.3 ¹	96.6 ¹	71.8 ¹	92.9 ¹	76.0 ²		81.7 ³
NMT ²					81.0		
REBELS (supervised) ⁴						94.0	
REBELS (unsupervised) ⁴						84.8	

Table 2.1 – Normalisation accuracy (%) of methods described. Best-performing model highlighted where comparison is possible

1=Pettersson (2016); 2=Korchagina (2017); 3=Scherrer and Erjavec (2013); 4=Mitankin et al. (2014)

The relevant literature for each method above reports intrinsic evaluation such as word accuracy/error rates. Table 2.1 summarises the accuracy reported in the above work. The variety of datasets used makes direct comparison difficult even when the language is notionally the same.

2.3.1 Analysis

Published work in historical text normalisation is narrowly focused on achieving high accuracy results, with little consideration for the practical issues at the core of the historical spelling variation problem; in particular, the lack of annotated data for training models. Models are generally trained using as much data as possible, with no analysis of model performance when less annotated data is available. This may be due to space limitations — larger works such as the PhD thesis of Pettersson (2016) and the MA thesis of Bollmann (2013) do contain such analyses. Knowing how models perform when training data is scarce is

of practical importance to the task of supervised historical text normalisation, since being able to compare methods on both accuracy and how much annotated data is required gives a better idea of which methods are likely to be adopted in real-world normalisation situations.

A commonality is the lack of investigation into what the spelling variation problem actually is, in terms of the empirical differences between historical and modern word forms. By sidestepping this question to varying degrees, the models employed are justifiable only on the grounds of their results. By leaving unstated their assumptions about what spelling variation actually constitutes, prior work justifies trying anything in the hope of achieving reportable levels of performance, rather than critically designing models which can reasonably be expected to address the problem at hand. SMT and NMT in particular, as they have been applied to historical spelling variation, have been lifted from machine translation with little in the way of introspection as to how the new task differs to the old — the investigative focus is entirely on determining which software settings achieve the best results. Principled models, which fully state the problem they are tasked with solving and how they are suited to dealing with particular aspects of that problem, are surely preferable.

In order to address this issue, the following chapter of this thesis closely examines the historical datasets that will be used in all experiments. I closely examine the word-level differences between historical texts and their modern counterparts, highlighting what work a model must do to perform normalisation.

Chapter 3

Historical Datasets

Four datasets are used in this work, each covering a different language. Three of these (German, Icelandic and Swedish) were created as part of Pettersson (2016) and are used here with no changes. The fourth dataset, English, was derived specifically for this work from the Letter Corpus component of the Innsbruck Corpus of Machine-Readable Texts (LC-ICAMET) (Markus, 1993). Details of each are given in Table 3.1.

Language	Time span	Genre	Tokens	Types
English	15th–18th century	Correspondence	178,094	26,229
German	17th–19th century	Multiple	38,651	9,833
Icelandic	15th century	Sagas, religious texts	61,717	14,942
Swedish	16th–19th century	Court records, church documents	29,119	10,724

Table 3.1 – Details of dataset sources

The dataset for each language consists of a list of tuples. The first item is an historical word form, the second is its manually annotated modern equivalent. No metadata for the historical words is available except for the English dataset, for which a variety of further information is available. This includes the year and place of writing as well as details (e.g. gender, class, education) of the author and the recipient.

3.1 Preprocessing

The Pettersson texts were provided in a convenient tabular format, with each line containing a historical word and its normalised modern form. No preprocessing was necessary.

LC-ICAMET contains 468 texts, manually normalised by a variety of people between 1992 and 1997. The corpus is provided as an interlinear gloss, with one line of historical text followed by a matching line of normalised text. Converting this to a tabular format was not straightforward: 7% of lines did not have the same number of words, meaning it was not possible to simply split each sentence on whitespace. These lines had to be manually inspected and corrected. This process revealed other issues with the corpus, with examples given in Table 3.2.

And forasmoche as, in þe name of Almighty god and in oure	Splitting of historical words
And for as much as, in the name of Almighty God and in our	
Mr Parr, I have received your letter, and I	Expansion of contractions
Mister Parr, I have received your letter, and I	
litill encresse. Never the lesse , as I have wrytyn to the Lorde	Concatenation of historical words
little increase. Nevertheless , as I have written to the Lord	
assercion be comers betwene of your gode desires, enclinyng	Deletion of historical words
assertion by comers between your good desires, inclining	
closing of thees, tidings of trouthe ben sent hider that	Insertion of modern words (e.g. auxiliary verbs)
closing of these, tidings of truth have been sent hider that	
and Marschall of France forth with have leyd siege	Insertion of emendations and other explanatory items
and Marshal of France forthwith/*immediately have laid siege	
much in al this tyme as oon balanger to revive their	
much in all this time as one balinger/*small ship to revive their	
And Sir, as for þe vj cowpull of haberndens , the which ye wryte ffore,	Lexical changes
And Sir, as for the 6 couple of *cod , the which you write for,	
on that was wyth me callid Roberd Lovegold, brasere, and threte	Syntactic and morphological changes
one that was with me called Roberd Lovegold, brazier, and threatened	
Prince, of þat þat your Lordly clemence so benignly vouchep sauf ,	
Prince, of that that your Lordly clemence so benignly vouchsaves ,	

Table 3.2 – Normalisation issues in the LC-ICAMET corpus

These issues pose several problems. Not only do they make it difficult to extract historical-modern word pairs, they take a degree of interpretative liberty with the source text. This results in poor training examples — how can *cod* be in any way considered a spelling variation of *haberndens*? Worse, there is no consistency in the application of these annotations. During the manual processing

of the corpus I took the opportunity to address the inconsistencies in Table 3.2 as well as the following:

- Word order differences between historical and modern texts were not corrected;
- Historical morphemes were not changed, e.g. *-th* to *-s*;
- Modern morphemes were not added where they could be considered missing in historical words;
- Archaic words which had been transliterated, e.g. *chirurgion* to *surgeon*, were reverted to their original form;
- Historical compounds were not split;
- Modern compounds were not used to represent multiple historical words;

My general approach was one of “leave it alone”. Where a normalisation candidate was unclear, no normalisation was performed, and any instances where the LC-ICAMET normalisation was seen to be interpretative beyond the orthographic level was undone. Many errors in the original normalisation were also corrected, such as *to* being used instead of *two*, *log* instead of *lodge*, *husbond* instead of *husband*. Examples of the differences between the historical source, the original LC-ICAMET normalisation and my revised normalisation are presented in Table 3.3.

All text was converted to lower case. Non-alphabetic characters within words were removed. 506 instances of the letter thorn, *þ*, were replaced with *th*. Foreign words, mainly Latin, were removed. Elements where either the historical or modern item used a non-lexical representation (counts, money, times and dates all commonly use a mix of Arabic or Roman numerals) were removed. These do not fall within the scope of a project dealing with lexical spelling variation.¹ The result was a tab-separated list of historical-modern word pairs.

¹For a thorough consideration of the normalisation of such elements, see Sproat and Jaitly (2016)

Historical	man that suffreth and helpeth it to be doon. Wherfor
Original	man that suffers and helps it to be done. Wherefore
Revised	man that suffereth and helpeth it to be done. Wherefore

Historical	after thys greuous compleynt, as is before seid, maed
Original	after this grievous complaint, as was before said, made
Revised	after this grievous complaint, as is before said, made

Historical	whiche hath be the Maier is grete laboure the grete part of all this
Original	which has been the Mayor's great labour the great part of all this
Revised	which hath be the Mayor is great labour the great part of all this

Historical	that he wend that he had be, the which worde is to hym right
Original	that he thought that he had been, the which word is to him right
Revised	that he wend that he had be, the which word is to him right

Table 3.3 – Examples of differences between historical text, the original LC-ICAMET normalisation and my revised normalisation

3.2 Modern language resources and baselines

For each dataset, a modern lexicon was created. This is later used to determine certain baseline figures and also in some elements of the experiments. For the English dataset, the standard UNIX dictionary² was used. For the other datasets, I used the same resources as Pettersson (2016):

- the Parole Corpus for German (Teubert, 2003)
- a database of modern inflectional forms (Bjarnadóttir, 2012) plus items appearing more than one hundred times in the Tagged Icelandic Corpus of Contemporary Icelandic Texts (Helgadóttir et al., 2012)
- version 2 of the Swedish Associative Thesaurus (Borin et al., 2010)

The lexicons differ markedly in size but this is partly due to the linguistic differences of each language. German has a high level of compounding. Swedish has gendered adjectives as well as multiple paradigms for declension of definite/indefinite singular and plural nouns. Icelandic is a highly inflected language with three genders, four distinct noun cases, and all nouns, pronouns and adjectives decline for both case and number. By comparison, English only distinguishes

²Located at `/usr/share/dict/words`

Language	Modern lexicon size
English	71,935
German	488,414
Icelandic	2,864,675
Swedish	736,147

Table 3.4 – Sizes of the modern lexicon resources used (unique items)

verbal inflection in one tense for the third person, has only one inflected plural form and uses the same grapheme, *-s*, to represent this morpheme for both.

3.3 Baselines per language

The standard approach in the literature to setting a baseline is to calculate the percentage of historical tokens in the testing data which already match their modern equivalent. This captures how similar the historical and modern texts are. A normalisation model can potentially achieve an accuracy lower than this baseline. This baseline focuses on the accuracy of the model with respect to the testing data.

Language	Baseline 1	Baseline 2	Historical in lexicon but needs modernising
English	77.377	78.393	2.75
German	90.427	86.08	3.55
Icelandic	47.343	32.647	7.11
Swedish	57.898	43.99	6.01

Table 3.5 – Baselines for the development set of each language (%). Baseline 1 compares the historical text to its gold standard. Baseline 2 compares the historical text to a modern lexicon.

A second, more holistic, approach is motivated by jointly considering the identification and normalisation components of the historical spelling variation problem. Since the aim of normalisation is to make historical tokens “modern”, then the focus should be on exactly those tokens which need modernising. A

simple identification method for finding historical tokens in need of normalisation is to search for them in a modern lexicon. The baseline then becomes the percentage of historical words found in the modern lexicon. Model performance cannot fall below this baseline, since only the tokens not found in the lexicon will be processed further. This is better motivated given the problem outlined in section 1.2: it addresses the identification issue (albeit in a shallow way) whilst mirroring how an historical text would be normalised in practice. If the aim is to reduce the number of unknown tokens, then it is towards precisely these tokens that attention should be directed. By comparison, we would consider a spell-checker that makes suggestions for *every* word in a text to be over-zealous. Of course, it may be the case that an historical word is found in a modern lexicon but should still be normalised.³ This is the case in as many as 7.11% of tokens in the datasets used here.

I will use the first baseline, as this will aid comparison to other work in this area whilst keeping the identification and normalisation tasks separate. The second is reported here to give an impression of the difference that even a simplistic approach to identification can have on the normalisation task.

3.4 Descriptive statistics

The models instantiated at the core of this thesis are sequence-labelling models. More specifically, these take in an historical string and return a (potentially) modified version. To present clear criteria of what such models must achieve, it will be helpful to examine in detail the differences between the historical words and their modern equivalents. The following analyses are over the entirety of the unique historical-modern word pairs available for each language.

For each historical-modern word pair, the average word lengths and difference in those lengths is shown for each data set in Table 3.6. On average, German has longer strings but fewer differences in length between historical-modern pairs. Swedish shows the greatest variance in length difference. Across all languages, historical strings tend to be longer than their modern equivalents.

However, it is more informative to look at the between-pair lengths rather than aggregate data. This is shown in Figure 3.1, where pairs are classified into three groups and shown as ratios relative to each other: those where strings are

³The excellent hypothetical example of historical *byte* and *bite* was pointed out to me

	English	German	Icelandic	Swedish
Historical word length	6.920 (2.268)	7.954 (2.775)	6.347 (2.169)	7.472 (2.740)
Modern word length	6.776 (2.278)	7.902 (2.771)	6.243 (2.141)	7.103 (2.728)
Difference	0.437 (0.597)	0.164 (0.421)	0.212 (0.451)	0.494 (0.742)

Table 3.6 – String length statistics [mean, (standard deviation)] per historical-modern word pair.

of equal length, those where the historical string is longer and those where the the historical string is shorter. In all languages, strings of equal length are the most common but there is a significant difference in the relative ratio of the three groups.

Figure 3.1 – Frequency comparison of pairs of historical-modern words, according to length difference

3.5 Edit distance between historical and modern strings

Differences in string length are an informative measure because they hint at how much “work” must be done to transform one into the other. What counts as work in the string edit literature (Wagner and Fischer, 1974) is generally edit operations

such as deletions, insertions, matches and substitutions. It will be useful to get a more precise view of what is required to transform a historical string into its modern equivalent beyond simple character counts. Simply comparing string length would suggest that *abcd* is more similar to *efgh* than to *abc* or *abcde*. I now investigate in more detail the differences between historical and modern strings.

A standard method for doing so is the Levenshtein algorithm, which calculates the minimum number of edit operations between two strings. This was used by many of the works examined in chapter 2. Looking at it now in more detail, Equation 3.1 shows a recursive method for implementing the algorithm, where $I_{(h_i \neq m_j)}$ is the identity function and is equal to 0 when two substrings are identical and 1 otherwise. Using dynamic programming to avoid recomputing common subcomponents of the procedure, it is possible to determine the minimum number of edit operations required to transform one string into the other. In addition, the minimal sequence of edit operations employed can be retrieved from the table by storing suitable backtraces for all operations.

$$\text{lev}_{h,m}(i, j) = \begin{cases} \max(i, j) & \text{if } \min(i, j) = 0, \\ \min \begin{cases} \text{lev}_{h,m}(i-1, j) + 1 \\ \text{lev}_{h,m}(i, j-1) + 1 \\ \text{lev}_{h,m}(i-1, j-1) + I_{(h_i \neq m_j)} \end{cases} & \text{otherwise.} \end{cases} \quad (3.1)$$

With the above established, it is possible to further quantify the work that must be done to transform an historical string into its modern equivalent. By aligning each historical-modern word pair with the Levenshtein algorithm, a memoised table is created, as described. The path through this table which minimises Equation 3.1 can be used to determine the optimal sequence of edit operations. Examples of the result of this process are shown in Figure 3.2.

Table 3.7 shows statistics on the operations per word, for each language, with mean and standard mean error. This provides a clear picture of the nature of the differences, at a character level, between the historical and modern forms for each language. Matches are the most common operation across all datasets but German stands out as having very few edit operations per string in general, suggesting very little variation between the historical and modern texts. This result aligns well with the baselines in Table 3.5, where 90% of the historical

Figure 3.2 – Example alignments where $|h| = |m|$, $|h| > |m|$, $|h| < |m|$

German text is already identical to the modern.

The “work” required to transform historical strings into modern ones can be characterised as follows. First, the majority of word-pairs require no changes to be made between characters since the historical character already matches the modern. Second, the datasets vary in both the overall volume and per word-pair average of the different types of edit operations but in general substitutions are most common, followed by deletions then insertions. The exception to this is the German dataset, as already noted.

A system which aims to transform historical strings into modern equivalents should therefore be able to not only perform such edit operations but also be sufficiently constrained so as not to *over-apply* them, given that the prevalent operation is to make no change. Indeed, this is exactly what the baselines discussed previously in section 3.3 would do — use “match” for every operation. However, edits do constitute a significant proportion (in most datasets ranging from 15-20%) and therefore being able to perform these accurately presents an opportunity to improve over baseline accuracy by a large margin.

	English			German			Icelandic			Swedish		
	μ	σ	count	μ	σ	count	μ	σ	count	μ	σ	count
Match	6.027	(2.403)	119527	7.749	(2.800)	59980	5.261	(2.182)	61127	6.510	(2.694)	57913
Delete	0.319	(0.553)	6318	0.110	(0.346)	849	0.168	(0.405)	1954	0.446	(0.733)	3964
Insert	0.175	(0.416)	3465	0.058	(0.270)	451	0.063	(0.252)	735	0.077	(0.293)	689
Substitute	0.574	(0.860)	11393	0.096	(0.330)	742	0.918	(0.927)	10671	0.516	(0.768)	4591
All edits	1.068	(1.112)	21176	0.264	(0.584)	2042	1.150	(1.059)	13360	1.039	(1.185)	9244

Table 3.7 – Levenshtein statistics over historical-modern word pairs. Mean and standard mean error are per word pair. Counts are over all word pairs.

3.6 Predictions

Given the quantification of the differences between the historical and modern words in each language, I predict the following.

- Normalisation models will perform best on German, since it has the least amount of variation overall.
- Icelandic will see the worst performance, due to the higher number of edits overall. Furthermore, the majority of these are substitutions which I expect to be especially difficult since they require targeting the correct item for replacement *and* choosing the correct substitute.
- Despite some similarity in terms of operations per word pair, the larger size of the English dataset will result in better performance than for Swedish.

Chapter 4

Hidden Markov Models

Having established a clear picture of the properties of the data, I now turn to the details of a probabilistic model and how it may, or may not, be suited to the task of normalising that data.

Hidden Markov Models (HMMs) are commonly used for sequence labelling tasks in bioinformatics and speech recognition. Normalising historical word forms can be seen as a sequence labelling task: for each character in the historical word, we want to find the corresponding modern character. However, they have so far been overlooked in historical spelling variation research in favour of the models outlined in chapter 2. A goal of this thesis, therefore, is to understand how HMMs compare with existing approaches to historical spelling normalisation.

4.1 Components of an HMM

An HMM consists of the following five components:

- the hidden states in the model;
- the emissions that can be observed when in each hidden state;
- the probability of transitioning to a particular state given the model's current state, $P(s_i|s_{i-1})$;
- the probability of emitting each of the observations available in each state, $P(o_i|s_i)$;
- the probability of the model beginning in each of the hidden states, $P(s_i|i)$.

These can be jointly represented by a transition matrix, T , of size $t \times t$, an emission matrix, E , of size $e \times t$ and a starting vector, S , of length t . t is the number of hidden states and e is the number of unique emissions possible. Together, these are the parameters, θ , of the model.

4.2 Relating HMMs to historical spelling variation

An HMM models a sequence-labelling process by treating one sequence as a series of selections from a list of possible hidden states and the other as a sequence of observations of emissions. In the context of historical text normalisation, the historical word forms are analogous to the observation sequence and the modern to the hidden. The simplest approach is to treat individual characters as states and emissions, but it is possible to define larger elements in the word forms (i.e. n-grams) as the basis for states and emissions.

Recasting Rabiner's second problem for HMMs (Rabiner, 1989) in these terms, the inference task becomes: given an observed sequence of historical elements $H = h_1, h_2, h_3 \dots h_n$ and a model λ , find $M = m_1, m_2, m_3 \dots m_n$ such that $P(M \mid \lambda)$ is maximised. This can be done with the Viterbi algorithm, using the same dynamic programming techniques applied previously to the Levenshtein algorithm. Each element in the historical sequence is treated as a tuple of its temporal position in the sequence and the emission that it represents. A table with one column per temporal position and one row per possible hidden state is constructed. The initial values are calculated by multiplying the starting vector S by the probability of the initial observed emission. The table is then filled in recursively with the result of Equation 4.1.

$$\text{vit}_t(j) = \max_{i=1}^N v_{t-1}(i) T_{ij} E_j(o_t) \quad (4.1)$$

This has three factors: the path probability so far of each previous state, the transition probability from each of those states to each of the possible next states and the probability of the current emission given each possible next state. The maximum value is stored along with a backtrace to the cell in the previous row represented by the value of v_{t-1} that helped maximised it.

4.3 Training an HMM

Given the annotated datasets, the model can be trained in a supervised manner. Based on observations of all the historical-modern word pairs in the training data, the parameters of the model are set using Maximum Likelihood Estimation such that they maximise $P(\text{dataset} \mid \theta)$. This simply involves counting the following: the frequency of transitions between elements m in a modern string to construct the transition matrix T

$$P(m_i \mid m_{i-1}) = \frac{\text{Count}(m_{i-1}, m_i)}{\text{Count}(m_{i-1})} \quad (4.2)$$

the frequency with which each element h in an historical string is dependent on a particular element m in the modern, to construct the emission matrix, E

$$P(h_i \mid m_i) = \frac{\text{Count}(h_i, m_i)}{\text{Count}(h_i)} \quad (4.3)$$

and the frequency with which each possible element in all modern strings is found at the start of a modern string (e.g. after a special start symbol $\$$) to construct the starting vector, S

$$P(h_i \mid \$) = \frac{\text{Count}(\$, h_i)}{\text{Count}(\$)} \quad (4.4)$$

4.4 Potential issues for HMMs

4.4.1 Observation sequence structure

As stated above, there is nothing to theoretically prevent training an HMM using elements larger than single characters. In practice, however, one must consider the nature of maximising $P(M \mid H, \theta)$. The historical strings are most easily considered as a sequence of individual character observations. Even when using an alignment technique that links multiple historical characters to one or more modern ones, it is not straightforwardly possible to later decompose the historical strings in the testing set into the appropriate sequence of multi-character observations, H . This is not true of the hidden modern states, since these are not needed as input at test time — they are captured by the model during training. Indeed, it is precisely these states which the model infers through the Viterbi algorithm.

As a result, when extracting statistics regarding transitions and emissions from the training data, care must be taken to ensure that what is considered a fundamental element in the observable historical strings does not depend on the hidden modern strings. The simplest approach has already been seen in the discussion of the Levenshtein algorithm, which was used to count the edit operations required to transform an historical string into a modern one. A by-product of this process is a character-to-character alignment, as seen in Figure 3.2 (page 19), from which model parameters can be extracted directly. The elements, then, are simply the set of individual characters found in the historical and modern strings. For the English datasets, this results in 27 hidden states and 27 possible emissions, one for each character plus a symbol representing deletion or insertion.

4.4.2 Differences between train and test observation sequences

The simplistic one-to-one alignment approach conflicts with our intuitions about how historical and modern strings should align. In the third alignment of Figure 3.2 (page 19), it would be reasonable to align historical *o* with modern *ou*, rather than place a special symbol into the historical string to represent a missing element.

The use of this special symbol is another practical constraint on the emissions used in an HMM. Its location within historical items, or whether it is even present at all, is unknown at test time. In the example of $H = [w, o, l, d]$ where the expected hidden state sequence is $M = [w, o, u, l, d]$, the test input would actually be $H = [w, o, l, d]$. The model will not be able to generate the correct sequence of states because the input is underspecified. Figure 3.1 (page 17) gave an indication of how many items this affects in each language: 10% of English, and 4% of German, Icelandic and Swedish.

A naïve way to circumvent this would be to place insertion symbols between every character of the input but this dramatically increases the number of inputs at test time and would require additional work to choose the correct output. A better approach, taking advantage of the fact that anything can be used as states, would be to simply train the model using one-to-many alignments and disallow inserts in the historical strings during alignment. This keeps the input at test time a sequence of single characters which are a fully representative subset of

the emissions seen during training, though the number of states will increase and some states will represent bigrams rather than just single characters.

The standard Levenshtein algorithm does not generate such alignments, however, so a different approach is needed. Ristad and Yianilos (1998) outlines a memoryless stochastic transducer, which learns to align elements in strings through the use of an expectation maximisation algorithm by optimising the transducer's parameters (i.e. the alignments possible plus the weights associated with them) with respect to a corpus of training pairs. This technique has been used with success to automatically align phonemes and text strings (Jiampojamarn et al., 2007), a task which shares similarities with aligning historical and modern strings since multiple characters are often needed to represent a single phoneme.

4.4.3 Model assumptions

The HMM assumes two properties of the states and emissions. First, future states in the model are conditionally independent of all past states as well as all emissions: S_{t+1} is independent of $S_1 \dots S_{t-1}$ and $E_1 \dots E_t$. Only S_t is relevant to S_{t+1} . Second, emissions at any point in time depend only on the current state: E_t is conditionally independent of $E_1 \dots E_{t-1}$ and $S_1 \dots S_{t-1}$. Only S_t is relevant to E_t . Taken together, these assumptions result in the future state of the model being separate from the past states.

The assumption of conditional independence has implications for language modelling because it restricts the expressive power of HMMs to the regular level of the Chomsky hierarchy. They can therefore adequately model adjacent relations between symbols but not long-range dependencies, as outlined in Yoon and Vaidyanathan (2006). The restrictions imposed by the assumption of conditional independence are certainly limiting factors for tasks such as POS tagging, particularly in languages such as Swiss German where cross-serial dependencies exist between distant words but also in English where *wh*-movement can significantly reorder the words of clauses.

This is not a fatal issue for the task of normalising historical spelling variation. Long-range dependencies are not a significant feature of orthography, especially here where the focus is on the relation between historical and modern forms: recall from the preceding exploration of edit operations (Table 3.7 on page 20) that the

majority of historical-modern pairs are quite similar. It would also be bizarre to suggest that the character at the end of an historical word is conditionally dependent upon the first letter of the modern one. Whilst there are pairs where one modern character may be seen as equivalent to two historical ones (e.g. as in *would* and *wold*), this is not the same as conditional dependence.

4.4.4 The problem of “best path” Viterbi

The Viterbi algorithm selects a path through the hidden states of the HMM by selecting the next state which maximises the probability of the path being built. However, the most probable path may not be the “correct” answer since the training data contain a very high proportion of matches, in comparison to deletions, insertions and substitutions, as was demonstrated in Table 3.7 on page 20. There is potential for the top Viterbi path to not match the expected output even though the model is capable of generating it.

The Viterbi algorithm can be modified to return the top k hidden state sequences (Seshadri and Sundberg, 1994) by storing and tracking the k best paths between states so far when recursively filling in the dynamic programming table. This list of candidate paths can either be reported directly, as in Hall (2007) where it was checked to see if it contained the target parse of a sentence, or it can be processed further, as in Charniak and Johnson (2005) where a discriminative maximum entropy classifier was used to rerank the k best parses.

A single definitive result is preferable to a k best list of possible options. Therefore, I propose two simple post-processing techniques. First, a lexical filter which removes from the list any items which are not found in the modern lexicon. This potentially removes strings which, though nonsensical, are permitted by the model. This should result in the correct item being moved to the top of the list. Second, a reranking system which scores each candidate, c , in the list and re-orders it. Equation 4.5 shows an n -gram character model where the probability of the next character is dependent on the previous n characters.

$$P(w_1, \dots, w_m) = \prod_{i=1}^{|c|} P(w_i \mid w_{i-(n-1)}, \dots, w_{i-1}) \quad (4.5)$$

The parameters for the model are approximated using maximum likelihood estimates drawn from a source corpus.

$$P(w_i \mid w_{i-(n-1)}, \dots, w_{i-1}) = \frac{\text{Count}(w_{i-(n-1)}, \dots, w_{i-1}, w_i)}{\text{Count}(w_{i-(n-1)}, \dots, w_{i-1})} \quad (4.6)$$

Candidate strings in the k best list can be evaluated by determining their probability under the trained language model and the list reordered with from highest to lowest.

The HMM is itself a bigram language model, since it uses $P(m_i \mid m_{i-1})$ to transition between elements in the modern characters. However, by using a higher order language model to rerank the HMM’s Viterbi candidates I hope to identify those candidates which have greater probability when considering sub-elements larger than the ones the HMM can. For an HMM trained on 1:1 alignments, this will be two single characters from the alphabet. When trained on 1:2 alignments this will be up to four single characters, i.e. composed of two bigrams.

Given the above properties of HMMs, and despite the possible shortcomings identified, they are a well-motivated approach to take and should be capable of reasonable performance when applied to the normalisation component of the historical spelling variation problem.

4.5 Experiments

4.5.1 Training, testing and development subsets

From each dataset, 80% was set aside for training purposes, 10% for testing and 10% for development. The split was made by randomly shuffling the word pairs and selecting the required percentage. Only the development set was used to evaluate models — the test set was unused. For German, Icelandic and Swedish the data subsets were exactly the same as used by Pettersson (2016).

4.5.2 Model outlines

To address the practical issues facing HMMs, discussed in section 4.4, two models were constructed per dataset.

Model 1 is trained using alignments generated by the standard Levenshtein algorithm, which I implemented myself. Model 2 uses alignments generated by the `m2m-aligner` stochastic transducer tool,¹ configured to allow deletions from

¹<https://github.com/letter-to-phoneme/m2m-aligner>

	Training		Testing	
	Tokens	Types	Tokens	Types
English	142475	17281	17809	4446
German	30920	6690	3865	1603
Icelandic	49373	10003	6172	2463
Swedish	23295	7608	2912	1567

Table 4.1 – Size of training and testing sets per language

modern but not historical strings and to allow each historical character to align with up to two historical characters. The transducer is trained only on the exact same training examples available to the HMM — no additional training examples are used.

The impact upon performance of separately applying either a lexical filter or language model reranking is evaluated, using the best-performing model. The implementation of the lexical filter is simplistic and removes from the model output any normalisation candidates that are not found in a modern lexicon. The language model is a little more complex. Using the KenLM Language Model Toolkit (Heafield, 2011), for each dataset seven language models (of orders 2 through 8) were trained for three data sources — the modern lexicon for that dataset as well as the modern and historical halves of the training data. To rerank an HMM’s output, the probability of each candidate in that output is calculated using the trained language model. The output is then reordered by that probability score.

Finally, a separate instantiation of each model is trained using between 5% and 100% of the available training data. This is done to determine how robust the models are in the face of limited annotated data — a situation common in natural language processing in general but especially so for historical corpora. All prior experiments use 100% of the available training data.

4.5.3 Model evaluation

Once trained, a model is tested by presenting an historical string as a sequence of individual characters. A list of ten normalisation candidates is returned, in order of decreasing probability.

The performance metric used is word accuracy, the standard metric in the literature. This is the percentage of processed historical forms which return the target annotated modern form as the most probable candidate. Accuracy is considered both when looking at the top candidate found by the Viterbi algorithm and the top ten candidates. These are referred to as Top1 and Top10 accuracy.

Where improvements over baseline are reported, this is with reference to baseline 1 of Table 3.5 and is the percentage of historical tokens which already match their modern counterpart in the testing data.

4.6 Results

Accuracy, plus raw improvement over baseline 1, are reported in detail. First, two standard models trained with 100% of the available training data (i.e. 80% of the total dataset). Then, extensions to the best-performing standard model: the lexical filter and language model reranking described in subsection 4.4.4.

4.6.1 Standard models

	Model 1				Model 2			
	Accuracy		Improvement		Accuracy		Improvement	
	Top1	Top10	Top1	Top10	Top1	Top10	Top1	Top10
English	71.099	87.141	-6.278	9.764	79.078	90.538	1.701	13.161
German	90.709	95.756	0.282	5.329	90.347	97.774	-0.08	7.347
Icelandic	57.673	89.629	10.33	42.286	62.745	93.891	15.402	46.548
Swedish	64.595	83.723	6.697	25.825	65.694	89.011	7.796	31.113

Table 4.2 – Model performance (% accuracy) and raw improvement over baseline (%) after training with 100% of the training set

Model 2, trained with 1:2 alignments generated by a stochastic transducer, outperforms Model 1. The more sophisticated alignment process better captures the character-level relationship between historical and modern forms, which is not always 1:1 as seen in Figure 3.1. Improvement over baseline by Model 2 is positive for all languages except German. This can be attributed to the lack of an

modern	historical	1	2	3	4	5	6	7	8	9	10
our	oure	oure	ouri	our	oura	owre	aure	oore	ouro	ourt	ouru
being	beyyng	besing	beeing	beaing	beieng	beting	beding	being	beying	beeeng	beseng
written	wretten	wretten	urettten	wrettin	written	oretten	wrethen	wreaten	urettin	wrerten	gretten
before	byfore	bifore	before	bofore	bafore	bifori	befori	bifor	befor	bfore	bifora
would	woulde	woulde	would	wouldi	woulda	woulee	woule	goulde	ooulde	wolde	wouldo
think	thinke	thinke	thince	think	thinki	thenke	thinee	thonke	thence	thine	thinca
me	mee	mee	mea	me	mei	mie	mer	me	med	met	mai
parcel	parcell	parcell	partell	percell	parsell	parcall	parell	parcel	parcill	parcoll	earcell
fail	fayle	faile	fail	faili	eaile	faila	fale	haile	fable	fayle	failo

Table 4.4 – Filter success examples. Removing candidates not found in the lexicon (light grey) results in the target form becoming the top candidate (dark grey)

identification system used to select candidates for normalisation. By processing every token in a text, there is the possibility of achieving accuracy below the baseline score. However, Top10 accuracy is generally very good which suggests that the model certainly has potential if the correct items can be extracted from that list of ten. I will now look at attempts to do so, focusing on Model 2 for brevity of presentation.

4.6.2 Lexical filter

Recall that the lexical filter takes the ten candidates produced by the modified Viterbi algorithm and removes any items that are not found in a modern lexicon. Rather than compare the results to the baseline, I compare them to the accuracy when the filter is *not* applied, to make clear the impact of the lexical filter. Results for Model 2, trained on 100% of the training data, are presented in Table 4.3. The impact of the lexical filter is generally negative, with only English seeing any improvement. Illustrative examples for English are shown in Table 4.6.

	Top1	Top10
English	7.053	-1.904
German	-3.364	-8.178
Icelandic	-28.294	-58.775
Swedish	-12.122	-27.885

Table 4.3 – Impact of lexical filter on model accuracy (raw % improvement over baseline model)

In general, the lexical filter performs exactly as expected by removing candidates which are arguably nonsense permitted by the model’s training (Table 4.4). However, in many cases it undoes the success of the model, removing candidates

which are actually correctly predicted. The problem lies in the nature of the lexicon used. For example, the removal of apostrophes when preprocessing the datasets creates “illegal” words which would not appear in any lexicon. Similarly, many archaic words (mostly due to morphological changes) are simply not found in a modern lexicon. In the case of German, the lexicon does not contain every possible compound combination. Nor does the Icelandic lexicon contain any archaic inflections. Therefore, careful creation of a lexicon suited to the task of historical spelling normalisation would therefore need to include more domain knowledge rather than simply being scraped from modern sources. For example, including data gazetteers would be beneficial in avoiding proper names being filtered out.

Another way to restrain the filter would be by having a better way to identify variants in the first place, as discussed in section 3.3. By normalising only the tokens not found in a modern lexicon, lexical filtering improves somewhat, as seen in Table 4.5. A combination of variant identification plus a better modern lexicon could result in increased lexical filter performance.

	Top1	Top10
English	6.0	-1.9
German	-2.2	-6.0
Icelandic	-3.9	-27.1
Swedish	5.6	-6.6

Table 4.5 – Impact of lexical filter on model accuracy, normalising only identified variants (raw % improvement over baseline model)

4.6.3 Reranking

Improvement over the non-reranked model is shown in Figure 4.1. Results are mixed and generally represent worse performance over the standard model. The exception is English.

In general, higher degree language models improve over lower ones but the overall change relative to the standard model only becomes positive in the case of English and German. Furthermore, the text used to train the language model impacts its efficacy in reranking candidates. It is not surprising that a language model trained on modern text (i.e. the normalised historical text) performs best

historical	modern	1	2	3	4	5	6	7	8	9	10	type
mens	mens	mens	mins	menc	mans	men	mees	mons	mns	mes	mene	no punctuation in lexicon
nother	nother	nother	nothar	nather	nothir	nether	nuther	nother	nther	nothea	wother	archaic segmentation
forsee	forsee	forsee	forsea	forse	forsei	forsie	forser	forse	horsee	forse	forste	incorrect normalisation
etc	etc	etc	et	eac	eec	eth	ett	atc	erc	itc	ecc	abbreviation
rieul	rieul	rieul	rioul	rieel	riel	riaul	rieal	ritul	reul	riiel	riul	proper place name
chirurgions	chirurgions	chirurgions	chirurgions	chirurgions	chirurgions	chirurgions	chirurgions	cherurgions	chirergions	cheregeons	chirorgeons	archaic vocabulary
chitting	chitting	chitting	chetting	thitting	chitteng	chithing	chiting	shitting	chitting	chisting	chitaing	proper personal name
testification	testification	testification	testivication	testification	testefication	testification	tistification	testification	testification	testificaton	testification	morphological productivity
adoing	adoing	adoing	adaing	ading	edong	adeing	atoing	adong	ading	arong	adoong	archaic morphology

Table 4.6 – Filter issues. Items in light grey are removed by the filter. In all cases, the model actually generates the correct form as the top candidate, which the filter then removes.

Figure 4.1 – Impact of language model reranking on Top1 accuracy

in all cases: this model potentially already encodes many of the strings the HMM is attempting to generate. However, having access to this data is highly unlikely in a real world situation — it is precisely that data that researchers want to generate.

The poor performance is likely due to multiple factors. Corpus size alone cannot account for the disparity, as the German dataset is one of the smallest. This may be mitigated by the very low level of variation in the German data, especially compared to Icelandic and Swedish. The degree to which these factors influence performance is unclear and requires further analysis.

4.6.4 Volume of training data

Figure 4.2 – Impact of training data volume on model Top1 accuracy. Baselines shown with dashed lines

The question of how much training data is required to achieve reasonable performance is an important one, as discussed previously in subsection 2.3.1. As can be seen in Figure 4.2, using as little as 5% of the training data achieves an accuracy that is not too distant from using 100%. English in particular displays a plateau effect, most likely due to the fact that the English corpus is large enough that 5% of the training set still contains over 800 unique words, unlike the other smaller datasets where 5% would mean between 300 and 500 unique words.

4.7 Summary

An HMM trained with 1:2 alignments normalises historical datasets in four languages with an accuracy of up to 15.4% above baseline. Extending the search for normalisation candidates beyond the top-ranked option has the potential to increase accuracy by between 7.3% and 46.5%, depending on the language. However, selecting the correct candidate from that list, through filtering or reranking, was not consistent across all languages and often hurt performance. This is likely due to (i) limitations of the modern lexicon used for filtering (ii) the lack of a system for identifying variants in order to prevent over-applying normalisation. For reranking, positive impact was generally lower than for filtering. A combination of training data volume and degree of variation within the historical text may account for the especially poor performance for Icelandic and Swedish.

One strength of the standard HMM model is robustness in the face of limited training data. Using as little as 5% of the training data achieves results that remain within 3% of those achieved with 100%. This applies to all datasets, despite the relative size differences between them.

These results highlight the importance of applying models to more than one language. If only English had been used here, the results would have been uniformly positive. However, this would give a misleading impression of the suitability of a k -best HMM trained on small volumes of data, with lexical filtering applied, to historical text normalisation.

Finally, the predictions made in section 3.6 (page 20) are borne out. In terms of raw accuracy scores, the German text is indeed easiest to normalise, whilst Icelandic is the most difficult. English and Swedish are second and third respectively, as predicted, but whether this really is due to the difference in training data size is not directly determinable from the results. A three-way analysis of

the interaction of corpus size, variation within text and model performance could answer this.

	LC-ICAMET1 (English)	LC-ICAMET2 (English)	GerManC (German)	IcePaHC (Icelandic)	GaW (Swedish)
Baseline	77.4	75.8	90.4	47.3	57.9
Rule-based ¹		82.9	87.3	67.3	79.4
Dictionary lookup ¹		91.7	94.6	81.7	86.2
Rule-based + dictionary ¹		92.9	95.1	84.6	90.8
SMT ¹		94.3	96.6	71.8	92.9
HMM ²	79.1		90.3	62.7	65.7
HMM+filter ²	85.1		88.1	58.8	71.3
HMM+rerank ²	84.9		92.8	58.2	65.1

Table 4.7 – Comparison of word accuracy (%) for HMM models to selected prior work. Best-performing model highlighted per dataset
1=Pettersson (2016); 2=this work

At this point, there is enough information to compare previously-untried HMM methods to those outlined in chapter 2. The extended summary is shown in Table 4.7. Because the same datasets are used here as in other work, direct comparisons can be made. However, the version of the LC-ICAMET corpus used here differs slightly from that used by Pettersson (2016), so direct comparison is not possible. HMM performance is below that of all other methods and even below baseline in the case of German. This leads to the conclusion that HMMs are no better suited to historical text normalisation than existing methods. The reasons for this likely lie in the issues outlined in section 4.4. Attempts to overcome these issues, as was seen, were met with mixed and limited success.

Chapter 5

Neural Network Models

In this chapter, I describe and evaluate current work on the application of neural network models to the historical spelling variation problem. I also cover a neural architecture which has been used with success in morphological inflection. In experiments similar to those used to evaluate HMMs, I assess the performance of two neural models.

5.1 Neural networks for sequence labelling

The application of neural networks to the historical spelling variation problem has focused on recurrent neural networks (RNNs), in particular those incorporating long short-term memory (LSTM) units in the hidden layer. The most successful work employs an encoder-decoder model. In this, the encoder transforms the variably-sized input into a fixed-length vector. The decoder then uses this new representation to compute the most likely output. The network is trained by optimising a cost function, such as cross-entropy, with the objective of minimising or maximising some objective for the training and testing data sets. Variations on this architecture have been used for many NLP tasks, such as speech recognition (Lu et al., 2015), machine translation (Bahdanau et al., 2014), morphological reinflection (Kann and Schütze, 2016), natural language generation (Shang et al., 2015), POS tagging (Ma and Hovy, 2016) and text summarisation (Nallapati et al., 2016).

In terms of historical text normalisation, an LSTM-based encoder-decoder model has attractive properties, which address issues raised of HMMs in section 4.4. They can better capture long-range dependencies in the input since

the assumption of HMMs, of conditional independence, is not made. It is also possible to learn directly from pairs of historical and modern words without pre-processing them into an alignment sequence, as shall be seen, through the use of an “attention mechanism” which helps the model to learn how items in the modern word rely on those in the historic. A further consequence of this is that there is no longer any need to consider how special characters for insertions/deletions should be dealt with in word pairs that are not the same length.

Neural networks are not without their own issues. Chief of these is the large amount of training data that is generally assumed to be needed. This has been shown to be less of an issue for historical spelling variation than might be expected (Korchagina, 2017; Bollmann and Sogaard, 2016). This is perhaps due to the fact that these models operate at the character level, meaning that even a few thousand word pairs can contain enough information about the spelling variation to achieve reasonable results. Another is the time required to train such models but here the small size of historical datasets is something of a boon. More problematic is the issue of interpretability. Generative models like the HMM can be used to extend our understanding of the historical spelling variation problem because they directly model (albeit in a simplistic fashion) the processes behind the spelling variation problem. Neural networks offer very little in the way of this, by comparison.

5.1.1 Application to historical spelling variation

Bollmann and Sogaard (2016) were the first to normalise historical text using neural networks, with a stack of three bi-directional LSTM units (Hochreiter and Schmidhuber, 1997) — the bi-directional encoding allows the network to consider all parts of the input at any time step during decoding, rather than just previous inputs. However, this was not an encoder-decoder model: after each character of the input was fed to the model, an output was immediately generated. As a result, the network had to be trained on aligned word pairs, generated by the Levenshtein algorithm.

The authors trained a separate model for each text in the Anselm corpus (Dipper and Schultz-Balluff, 2013). The justification for not training a single model on the entire corpus, was that the texts differ by region and era, and therefore exhibit different characteristics in their spelling variation. The average

text length was 7353 tokens. In addition to this standard training, multi-task learning was applied where the network was additionally trained on 10000 random tokens from other texts in the corpus. Average word accuracy was 79.9% for the standard model, 80.55% for the multi-task learning setup. Rule-based approaches, applied using the Norma tool, achieved 77.83% and 77.48% respectively.

The effect of training volume was investigated for one text of 4718 tokens, using between 100 and 2718 tokens, with the expected result that more is better. However, the LSTM model performed poorly compared to the rule-based model with low volumes of training data. The former ranged from 40% to 80%, the latter from 68% to 80%.

In their most recent work, Bollmann and Søgaard (2017) employ an encoder-decoder model. There was therefore no need to pre-align word pairs, as described previously. The auxiliary training task changed from simply taking tokens from random texts to pairs of modern words and their phonetic transcription taken from the CELEX lexical database (Kerkman et al., 1995), which the authors describe as “learning to pronounce”. Furthermore, a soft attention mechanism (Xu et al., 2015) was used. This uses the input seen so far at each time step to create a vector which summarises how relevant the input being considered is to the next possible output. The model learns this during training.

Using the Anselm corpus again, Bollmann and Søgaard trained and evaluated two classes of model: with and without multi-tasking learning. Each of these was also evaluated with and without the attention mechanism. The results represent the current state of the art for that particular corpus, with the base model plus attention averaging 82.72% accuracy and the multi-task learning *without* attention averaging 82.76%. The authors took this result to indicate the equivalence of multi-task learning and the attention mechanism.

5.1.2 Shortcomings of the encoder-decoder work

The accuracy of the encoder-decoder model is impressive, but some methodological issues must be highlighted.

First, each of the 44 texts in the Anselm corpus had its own model. This is a reasonable approach to take, reflecting both the reality of how texts may be normalised in the real world as well as the fact that texts generally exhibit different degrees and kinds of variation — training a single general model may

not be the best approach.

However, in both Bollmann and Søgaaard (2016) and (2017), each model was evaluated only on the first 1000 tokens of the text (constituting between 4 and 13%) and trained on the entirety of the remainder: between approximately 2000-11000 tokens. No justification is ever offered for it, though it is conceivably due to the small size of many documents: a 10% testing set could contain very few tokens. It would have been worthwhile to determine the accuracy of a “general” model, trained and evaluated on many texts.

Second, there was no investigation of languages other than German. Though a monolingual focus is not uncommon in the literature, this is an unfortunate situation. As was seen in section 4.6, using only one dataset can give a misleading impression of the generality of a given model — what works for German may not work for Icelandic. A broader evaluation would have been enlightening, especially as this was the first published application of the encoder-decoder architecture to historical text normalisation.

Finally, the “learning to pronounce” approach requires additional resources in the form of phonetic transcriptions of modern words, taken from the CELEX database. This covers only Dutch, English and German. Therefore, to apply this method to other languages would require the creation of phonetically transcribed training data, which is a not insignificant undertaking. I also question the sense of learning to pronounce only the modern words — a much better approach would involve also learning the pronunciation of historical words, though how this data would be generated is an open question.

It should also be noted that the baselines reported are not based on the texts but on the normalisation accuracy achieved by other models. Therefore, it is not possible to state with certainty that the accuracies reported are an actual improvement and, if so, by exactly how much.

5.2 Drawing parallels with morphology

The task of morphological generation as laid out in Cotterell et al. (2016) has two variants, inflection and reinflection. The former takes as input a lemma and a set of morphosyntactic features and outputs a suitably inflected form. The latter generates the target inflected form from a combination of a non-lemma form and either a set of source-target features or only target features. Though clearly not

one and the same problem, the parallels between the challenges facing models of both morphology and historical spelling variation normalisation are striking. Both can be used to improve a downstream NLP task by reducing the number of unknown tokens. There is often a paucity of data when working with low-resource languages and historical text. Each task can be viewed as a restricted series of edit operations: given an input string, change a few parts of it until it matches the output string. There are differences too, of course. The input in a morphological inflection/reinflection task may contain more data than a single word. And while there is a one-to-one mapping between input and output for morphology, this is a many-to-one mapping for historical and modern word forms.

5.2.1 The hard monotonic attention model

Aharoni et al. (2017) describe a neural model for the task of inflection generation which uses a “hard” attention mechanism. Whilst the soft attention mechanism used by Bollmann and Søgaard (2017) considers all hidden states up to the current time step through the input, the hard attention mechanism focuses on only some of the most recent hidden states at a time. At each time step, two actions are possible: a character from the model’s alphabet can be appended to the output sequence, or the system can generate a special symbol which advances the focus of attention to the next input.

The motivation for this mechanism is the observation that alignment between characters in a pair of inflected words generally proceeds in a monotonic fashion. This is in comparison to alignments between sentences in different languages where word order differences (in particular transposition) may result in a non-monotonic alignment between the words. The model is not limited to conditioning the output on a slice of the current input, however: both the encoder and decoder layer are composed of LSTM units which captures long-range relations between the input and output.

Each training example is a triple consisting of a lemma, a target inflection and the target morphological features. From this, a sequence of write/advance actions is generated from the character-level alignment of lemma and target through an unsupervised Chinese Restaurant Process (Sudoh et al., 2013). This produces 1:1 alignments, with insertions/deletions permitted which represent the advance action. The model is then trained to mimic this sequence of write/advance ac-

tions. At test time, a lemma is presented along with the target morphological features. This generates a sequence of output characters mixed with advance symbols. These are stripped out to leave the predicted inflected form. The model performed well in the SIGMORPHON2016 tasks, often comfortably ahead of soft attention models as well as systems based on hand-crafted transformation rules.

5.2.2 Applying hard monotonic attention to historical spelling variation

As described in chapter 2, historical text normalisation has been treated as a transduction task through the application of rewrite rules. The question is whether the monotonic assumption made by Aharoni et al. (2017) holds as well for spelling variation as it does for morphology. Figure 5.1 shows example alignments for three different tasks discussed so far. This illustrates that monotonicity between historical and modern words certainly is possible, as long as transpositions of characters does not occur. The Levenshtein algorithm can be modified¹ to count transpositions and there are very few in the data: 0.83% of word pairs the entire English dataset contain transposed characters, 0.06% in German, 1.02% in Icelandic, 0.18% in Swedish. This is not surprising: recall that historical spelling variation is not the result of hitting keys out of order. The larger number of transpositions for English and Icelandic, relative to the other languages, may have an impact on the performance of a hard monotonic model for these specific datasets. A reasonable prediction would be that the model may perform poorly for these English and Icelandic when compared to the others.

Figure 5.1 – Alignment examples for spelling, morphology and translation. Only the first two are monotonic

¹This modified version is known as the Levenshtein-Damerau algorithm

The model can easily be adapted to learn how to normalise historical text. The only difference is that there is no need to provide morphological features. The training data is therefore simplified to pairs of historical-modern word forms, which are automatically aligned in order to create a transduction sequence.

5.3 Experiments

I apply the encoder-decoder architecture, using two attention mechanisms, to the datasets described in chapter 3. The soft attention model uses code made available² as part of Bollmann and Søgaard (2017). I adapted the code³ from Aharoni et al. (2017) to the task of historical text normalisation. The default hyperparameters from each paper are retained and models trained for fifty epochs. Loss is calculated against the training set.

I investigate the impact of training data volume on the English, German, Icelandic and Swedish datasets, using from 5 to 100% of the available training data. This goes some way towards addressing the first and second issues raised in subsection 5.1.2. The third issue is avoided by taking at face value the claim in Bollmann and Søgaard (2017) that the soft attention mechanism performs almost as well as the multi-task system (and using both harms performance) and using only soft attention. Finally, the same baseline is used as in all other experiments in this work.

5.4 Results and comparisons

Accuracy for both models is generally high, above baseline (between approximately 8 to 43%) for all languages. More training data improves performance, but annotating just 10% of even a small corpus, as in the case of Swedish, still achieves accuracy above 80%. The predictions made in section 3.6 still hold, with German achieving the highest accuracy and Icelandic the lowest.

These results support the assumption of hard monotonic alignment between historical and modern words. The soft attention model, making no such assumptions about the structure of the data, still performs well but fails to normalise text as accurately. Concerns that the greater number of transpositions in the

²<https://bitbucket.org/mbollmann/acl2017/>

³<https://github.com/roeeaharoni/morphological-reinflection>

Figure 5.2 – Accuracy per volume of training data used. Results from section 4.6 shown for comparison purposes. Baseline shown with dashed line

English and Icelandic data prove to be unfounded.

The justification for training document-specific models, given by Bollmann and Søgaaard (2017), is not strongly supported. For each language, a general model trained on documents from as many as four different centuries achieves highly competitive accuracy, possibly due to the greater volume of training data doing so makes available.

5.5 Summary

Having evaluated HMM, soft attention and hard attention models, the final version of Table 2.1 can be produced. This permits direct comparison of many of the methods discussed throughout this work, in addition to HMMs and hard attention neural networks which have been applied here to historical spelling variation for the first time. Of these, the hard attention model achieves state-of-the-art performance on all datasets.

	Anselm (German)	LC-ICAMET1 (English)	LC-ICAMET2 (English)	GerManC (German)	IcePaHC (Icelandic)	GaW (Swedish)
Baseline	Not given	77.4	75.8	90.4	47.3	57.9
Rule-based ¹			82.9	87.3	67.3	79.4
Dictionary lookup ¹			91.7	94.6	81.7	86.2
Rule-based + dictionary ¹			92.9	95.1	84.6	90.8
SMT ¹			94.3	96.6	71.8	92.9
HMM ²		79.1		90.3	62.7	65.7
HMM+filter ²		85.1		88.1	58.8	71.3
HMM+rerank ²		84.9		92.8	58.2	65.1
LSTM (plain) ³	80.6					
LSTM+MTL ⁴	82.8					
LSTM+soft	82.7 ⁴	91.1 ²		97.2 ²	87.1 ²	95.3 ²
LSTM+hard ²		94.6		99.7	91.0	98.6

Table 5.1 – Comparison of word accuracy (%) for all models discussed and/or evaluated in this work. Where direct comparison is possible, the best-performing model is that presented in this work
1=Pettersson (2016); 2=this work; 3=Bollmann and Søgaaard (2016);
4=Bollmann and Søgaaard (2017)

Chapter 6

Comparison of models

Each model can already be distinguished by its accuracy score. But are there further differences in the normalisation predictions each model makes? In this final chapter I will compare the output of the HMM, soft attention and hard attention models, trained on 100% of available training data. By examining the weaknesses of the models, I will be better able to determine directions for future work on historical spelling normalisation.

6.1 Qualitative analysis

Looking at all 30,758 test items in the four datasets which only a single model failed, the HMM had 7,240 unique failures, the soft attention model 358 and the hard attention model 111. A selection of these unique failures (for English) are presented in Table 6.1. Several errors for neural models are somewhat bizarre and inscrutable, whilst the HMM can be characterised as over-applying common patterns.

In many cases for English, the problem lies in the quality of the gold standard text as discussed in section 3.1. One example is modern *hagh* for historical *hagh* rather than the expected *hague*. The HMM and soft model match the gold standard, but the hard model actually predicts the “correct” answer and is penalised. It appears that several hundred historical words are still not consistently annotated.

Homophony is another issue. Similar-sounding words like *where* and *were* share many historical variants. There is no way to determine the target word without context. The hard and soft models normalise *there* to *there* rather than

Model	Historical	Modern	Prediction
HMM	lady	lady	ladi
HMM	comfortyd	comforted	comfortid
HMM	eny	any	eny
Soft	meseemeth	meseemeth	meseems
Soft	qualities	qualities	qualitios
Soft	subscribed	subscribed	subsmribed
Hard	pacification	pacification	patification
Hard	leyden	leiden	itiden
Hard	ymagine	imagine	ymagine

Table 6.1 – Unique normalisation failures for each model

their and are penalised. A similar problem arises when an historical word form is a valid variant of more than one modern word. In the English dataset, almost 500 historical words have this property. Examples are historical *curt* (modern *court* and *curt*) and *hire* (*hear*, *her*, *hire*). The result is a negative impact on accuracy because every model always normalises that input to only one output.

6.2 Quantitative analysis

Accuracy scores reported so far have been at the token level, with all occurring historical words counted in that statistic. Another perspective is found at the type level, focusing on unique historical words. A system which is successful at normalising a few very common tokens (especially those with little very variation such as modern *that* which appears 903 times as *that* and 9 times as *thatt*) may appear to perform as well as one which normalises many rare tokens. The hard attention model, which performed best in token accuracy, also out-performs other models at the type level (Table 6.2). The hard model results are therefore not an artefact of the distributional characteristics of words within the data. Both rare and common words are often successfully normalised.

In chapter 3 I investigated the work (i.e. the number of edit operations) that must be undertaken to transform an historical word into its modern form. This relationship can be extended (Figure 6.1) to include the prediction of a model which has been given that historical word as an input for normalisation.

	English	German	Icelandic	Swedish
HMM	48.8	80.7	50.3	56.8
Soft	71.0	93.7	83.1	92.1
Hard	83.5	99.6	92.9	98.3

Table 6.2 – Type accuracy (%) for each model, per dataset. Best-performing model highlighted

Comparing the prediction to the historical word tells us how much work the model did, whilst comparing the prediction to the modern word tells us how far short the model fell. For a perfect model averaged over all inputs, work done will always equal work required and work yet to be done will always equal zero. For the three models in this work, it should be obvious that the hard attention model both does the most work and leaves the least work undone.

Figure 6.1 – Relationship between historical, modern and predicted word forms

Recall there are five possible edit operations: match (do nothing), substitute one character for another, insert a new character, delete a character, transpose adjacent characters. Are these operations handled equally by the models, with most accuracy errors being due to the quality or quantity of training data? Or are some operations easier than others? Figure 6.2 shows that models are more successful on historical words which are fairly similar to their modern counterpart due to having fewer operations that require making changes. In general, the hard attention model is better able to normalise “harder” historical words which differ more from their modern counterparts.

Figure 6.2 – Mean number of substitute, insert, delete and transpose operations per correct/incorrect item per model for each dataset

To get a fuller picture of model accuracy at the level of edit operations, two quantities were combined. The first is the operation accuracy in the cases where a model prediction is correct. This is augmented with data from the cases where the prediction was wrong but progress was made towards the correct answer — the intersection of undertaken operations (determined by comparing the historical word to the prediction) and the operations expected (determined by comparing the historical word to the modern).

The results reported in Table 6.3 for all five operations give a much more detailed insight into the where models fail. The HMM bolsters its overall performance by being almost perfect at predicting matches. Other models improve upon this with far superior accuracy for other operations. The hard model is uniformly good across the board, with perhaps one exception for insertions in English. Concerns about non-monotonic alignments being problematic for the hard model are somewhat borne out with lower accuracy for this operation but the low number of opportunities to both observe transpositions in the training data or apply them at test time make it difficult to draw a solid conclusion.

6.3 Future work

The remarkable accuracy of the hard attention model almost obviates a lengthy discussion of how it can be improved upon. Not only does it avoid issues regarding alignments or string length differences which cause trouble for HMMs (section 4.4), it performs extremely well even with minimal training data and for languages with very different linguistic properties (section 5.4). The model does require some information about alignment, generated through an unsuper-

		Delete	Insert	Substitute	Transpose	Match
English	Gold standard	1840	1079	2830	153	69374
	HMM	1.576	0.834	42.686	0	99.542
	Soft	75.652	62.280	73.216	62.745	99.103
	Hard	84.022	71.918	83.958	79.085	99.575
German	Gold standard	182	132	154	0	19503
	HMM	0.549	1.515	51.299	N/A	99.569
	Soft	84.615	75	85.714	N/A	99.457
	Hard	98.901	95.455	97.403	N/A	99.985
Icelandic	Gold standard	706	251	3660	77	22491
	HMM	16.431	11.952	58.716	0	98.515
	Soft	83.144	74.502	93.011	35.065	98.009
	Hard	93.484	86.454	94.977	24.675	98.609
Swedish	Gold standard	914	217	936	3	13376
	HMM	34.464	2.304	34.188	0	98.729
	Soft	91.357	88.018	94.124	66.667	99.439
	Hard	96.827	98.157	97.863	100	99.948

Table 6.3 – Accuracy (%) per edit operation for each class of model, with highest accuracy highlighted per dataset. The observed count of each operation in the gold standard development set annotations is provided for reference

vised statistical method, so it may be possible that a better method exists for generating these alignments, e.g. the stochastic transducer that produced the 1:2 alignments for the HMM.

Another avenue may be explicitly training a neural network to perform the whole set of edit operations. The hard attention model learns the ability to monotonically advance the focus of its attention from the alignment data during training. It may be productive to also train on lists of Levenshtein-derived edit operations, such that the model directly learns how to delete, insert, substitute, transpose and match.

The issue of contextual disambiguation was briefly mentioned (section 6.1).

Currently, models consider words in isolation. A model which can contextually disambiguate historical words will have a higher type accuracy. The question is how to use context to address spelling variation when that context itself is also subject to the same variation. One approach could be to identify anchor words which are relatively invariant in order to bootstrap normalisation in a top-down approach. This would mean the model would select which words to normalise first, rather than proceeding in a linear bottom-up fashion from the first word to the last. Its own output would be used to aid normalisation as it progressed. How much of this would be algorithmic and part of the model and how much would be heuristic is an open question.

It should also be remembered that normalisation, though very much the focus of research into historical spelling variation, is only one part of the problem (section 1.2). There is still much to be done on identifying variants, without recourse to general purpose resources such as modern lexicons. But even the simplistic approach described in section 3.3 could be combined with the hard attention model and released as a software package for researchers working with historical texts. Given the success of a generally-trained model, it may even be possible to make such models publicly available without the need for users to have access to significant computational power. They need only select a model which matches the language and (approximate) era of the corpora they wish to normalise.

Finally, despite the performance of the hard attention model it is still a supervised method and requires annotated training data. The required volume of this data has turned out to be surprisingly small, but an exploration of unsupervised techniques could become more pressing as unannotated historical documents of more and more languages become available.

6.4 Conclusion

I began this thesis with a thorough investigation of the relation between historical and modern texts, in order to better understand what a normalisation model is required to be capable of. I connected these findings to the properties of HMMs and investigated the ability of such models to normalise historical text. Results were disappointing but a previous technological gap in approaches to the historical variation problem was filled.

I applied very recent work on historical normalisation, which used LSTMs with a soft attention mechanism, to a number of new datasets in order to better assess the performance of that model as well as address methodological issues in that work regarding how training and testing is executed. Results were good. It was also shown that a generally-trained model can perform well — there is no need to train document-specific models.

State-of-the-art accuracy results were then achieved in all experiments, using an LSTM with a hard attention mechanism. I adapted this architecture from recent work in morphological inflection and applied it to historical text normalisation for the first time. The assumption of a hard monotonic alignment between historical and modern words does indeed hold and gives a significant advantage over models which perhaps consider too much information from all parts of a word at any time.

The initial investigation of what work is required to turn historical words into their modern counterparts was augmented by a detailed analysis of how different models perform this work. It was shown that the best-performing models are able to model a wide variety of edit operations.

Finally, the same datasets were used for all models in this work, allowing direct comparison between those models but also with results from previous work. The volume of training data required to achieve reasonable accuracy was also investigated for all models and it was shown that a generally-trained hard attention model can perform competitively even when trained with as little as 10% of the available data — between 300 to 800 unique historical words. This is in comparison to most work which trains with as much data as possible, and is an important finding for an area of research where annotated data is both scarce and expensive to create.

Bibliography

- Aharoni, R., Goldberg, Y., and Ramat-Gan, I. (2017). Morphological inflection generation with hard monotonic attention. *Proceedings of ACL*. <https://arxiv.org/abs/1611.01487>.
- Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Baron, A. and Rayson, P. (2008). VARD2: A tool for dealing with spelling variation in historical corpora. In *Postgraduate conference in corpus linguistics*.
- Bjarnadóttir, K. (2012). The database of modern Icelandic inflection. In *Proceedings of Language Technology for Normalization of Less-Resourced Languages, workshop at the 8th International Conference on Language Resources and Evaluation, LREC*.
- Bollmann, Marcel, B. J. and Søgaard, A. (2017). Learning attention for historical text normalization by learning to pronounce.
- Bollmann, M. (2012). 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*.
- Bollmann, M. (2013). Automatic normalization for linguistic annotation of historical language data. Master’s thesis, Ruhr-Universität Bochum.
- Bollmann, M., Petran, F., and Dipper, S. (2011). Applying rule-based normalization to different types of historical texts - an evaluation. In *Language and Technology Conference*, pages 166–177. Springer.
- Bollmann, M. and Søgaard, A. (2016). Improving historical spelling normalization with bi-directional LSTMs and multi-task learning.

- Borin, L., Forsberg, M., and Lönngren, L. (2010). Swedish associative thesaurus [electronic resource].
- Charniak, E. and Johnson, M. (2005). Coarse-to-fine N-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Cotterell, R., Kirov, C., Sylak-Glassman, J., Yarowsky, D., Eisner, J., and Hulden, M. (2016). The SIGMORPHON 2016 shared task—morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22.
- Dipper, S. and Schultz-Balluff, S. (2013). The anselm corpus: Methods and perspectives of a parallel aligned corpus. In *Proceedings of the workshop on computational historical linguistics at NODALIDA 2013; May 22-24; 2013; Oslo; Norway. NEALT Proceedings Series 18*, number 087, pages 27–42. Linköping University Electronic Press.
- Eisenstein, J. (2013). What to do about bad language on the internet. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 359–369.
- Evans, M. (2011). *Aspects of the idiolect of Queen Elizabeth I: A diachronic study on sociolinguistic principles*. PhD thesis, University of Sheffield.
- Fisher, J. H. (1977). Chancery and the emergence of standard written English in the fifteenth century. *Speculum*, 52(4):870–899.
- Hall, K. (2007). K-best spanning tree parsing. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 392–399, Prague, Czech Republic. Association for Computational Linguistics.
- Han, B., Cook, P., and Baldwin, T. (2012). Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 421–432, Stroudsburg, PA, USA. Association for Computational Linguistics.

- 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*, pages 1–6.
- Heafield, K. (2011). KenLM: faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, United Kingdom.
- Helgadóttir, S., Svavarsdóttir, Á., Rögnvaldsson, E., Bjarnadóttir, K., and Loftson, H. (2012). The tagged Icelandic corpus (MÍM). In *Proceedings of the Workshop on Language Technology for Normalisation of Less-Resourced Languages*, pages 67–72.
- Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Jiampojarn, S., Kondrak, G., and Sherif, T. (2007). Applying many-to-many alignments and Hidden Markov Models to letter-to-phoneme conversion. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 372–379, Rochester, New York. Association for Computational Linguistics.
- Kann, K. and Schütze, H. (2016). Single-model encoder-decoder with explicit morphological representation for reinflection. *arXiv preprint arXiv:1606.00589*.
- Kerkman, H., Piepenbrook, R., Baayen, R., and van Rijn, H. (1995). The CELEX lexical database.
- Korchagina, N. (2017). Normalizing medieval german texts: from rules to deep learning. In *Proceedings of the NoDaLiDa 2017 Workshop on Processing Historical Language*, number 133, pages 12–17. Linköping University Electronic Press.
- Lee, J., Cho, K., and Hofmann, T. (2016). Fully character-level neural machine translation without explicit segmentation. *arXiv preprint arXiv:1610.03017*.
- Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

- Lu, L., Zhang, X., Cho, K., and Renals, S. (2015). A study of the recurrent neural network encoder-decoder for large vocabulary speech recognition. In *INTERSPEECH*, pages 3249–3253.
- Ma, X. and Hovy, E. H. (2016). End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. *CoRR*, abs/1603.01354.
- Markus, M. (1993). The concept of ICAMET (Innsbruck computer archive of Middle English texts). In *Corpora Across the Centuries: Proceedings of the First International Colloquium on English Diachronic Corpora, St Catharine's College Cambridge, 25-27 March 1993*, number 11, page 41. Rodopi.
- Mitankin, P., Gerdjikov, S., and Mihov, S. (2014). An approach to unsupervised historical text normalisation. In *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage, DATECH '14*, pages 29–34, New York, NY, USA. ACM.
- Nallapati, R., Xiang, B., and Zhou, B. (2016). Sequence-to-sequence RNNs for text summarization. *CoRR*, abs/1602.06023.
- Pettersson, E. (2016). *Spelling normalisation and linguistic analysis of historical text for information extraction*. PhD thesis, Uppsala University.
- Pettersson, E., Megyesi, B., and Nivre, J. (2013a). Normalisation of historical text using context-sensitive weighted levenshtein distance and compound splitting. In *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013); May 22-24; 2013; Oslo University; Norway. NEALT Proceedings Series 16*, number 085, pages 163–179. Linköping University Electronic Press.
- Pettersson, E., Megyesi, B., and Tiedemann, J. (2013b). An SMT approach to automatic annotation of historical text. In *Proceedings of the workshop on computational historical linguistics at NODALIDA 2013; May 22-24; 2013; Oslo; Norway. NEALT Proceedings Series 18*, number 087, pages 54–69. Linköping University Electronic Press.
- Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

- Ristad, E. S. and Yianilos, P. N. (1998). Learning string edit distance. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 20(5):522–532.
- Rocio, V., Alves, M. A., Lopes, J. G., Xavier, M. F., and Vicente, G. (2003). *Automated Creation of a Medieval Portuguese Partial Treebank*, pages 211–227. Springer Netherlands, Dordrecht.
- Sariev, A., Nenchev, V., Gerdjikov, S., Mitankin, P., Ganchev, H., Mihov, S., and Tinchev, T. (2014). Flexible noisy text correction. In *Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on*, pages 31–35. IEEE.
- Scherrer, Y. and Erjavec, T. (2013). Modernizing historical slovene words with character-based SMT. In *Proceedings of the 4th Biennial International Workshop on Balto-Slavic Natural Language Processing*, pages 58–62, Sofia, Bulgaria. Association for Computational Linguistics.
- Seshadri, N. and Sundberg, C.-E. (1994). List Viterbi decoding algorithms with applications. *IEEE Transactions on Communications*, 42(234):313–323.
- Shang, L., Lu, Z., and Li, H. (2015). Neural responding machine for short-text conversation. *arXiv preprint arXiv:1503.02364*.
- Sproat, R. and Jaitly, N. (2016). RNN approaches to text normalization: A challenge. *CoRR*, abs/1611.00068.
- Sudoh, K., Mori, S., and Nagata, M. (2013). Noise-aware character alignment for bootstrapping statistical machine transliteration from bilingual corpora. In *EMNLP*, pages 204–209.
- Teubert, W. (2003). German Parole Corpus. Electronic resource.
- Wagner, R. A. and Fischer, M. J. (1974). The string-to-string correction problem. *J. ACM*, 21(1):168–173.
- Wieling, M., Prokić, J., and Nerbonne, J. (2009). Evaluating the pairwise string alignment of pronunciations. In *Proceedings of the EACL 2009 Workshop on Language Technology and Resources for Cultural Heritage, Social Sciences, Humanities, and Education, LaTeCH-SHELT&R ’09*, pages 26–34, Stroudsburg, PA, USA. Association for Computational Linguistics.

- Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057.
- Yoon, B.-J. and Vaidyanathan, P. (2006). Context-sensitive hidden Markov models for modeling long-range dependencies in symbol sequences. *IEEE Transactions on Signal Processing*, 54(11):4169–4184.