Issues in Named Entity Recognition on Early Modern English Letters

Vanessa Woldenga-Racine

A master's thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2019

Reading Committee:

Dr. Fei Xia, Chair

Dr. Ryan Georgi

Program Authorized to Offer Degree:
Department of Linguistics

University of Washington

Abstract

Issues in Named Entity Recognition
on Early Modern English Letters

Vanessa Woldenga-Racine

Chair of the Supervisory Committee:
Dr. Fei Xia
Department of Linguistics

The influx of digitized historical documents into online collections has made the study of these documents much more accessible to researchers and the general public. This data, however, is frequently raw data sometimes obtained through automated methods such as optical character recognition. Without rich metadata, the content of these documents is difficult to search and organize. Tasks commonly undertaken in the field of computational linguistics can aid in this endeavour. These documents often present challenges for modern systems, however, as the text contained in historical documents frequently differs in many ways from the present-day newswire these systems are most often trained on. In this thesis I explore the task of Named Entity Recognition on texts written in Early Modern English. I investigate three methodologies for bootstrapping training data to train a character-based neural net model. The results show substantial improvements upon all baselines, with the best f-measure at 60.31%.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

As more and more historical documents are digitized, their content becomes increasingly available to a wider audience and safeguarded against loss due to deterioration or dectruction of the original document. While this step alone is a great achievement, raw data on its own is not very accessible to the average user. In order for people to be able to browse, search, and query the data it needs to be enriched with various types of metadata (Sporleder, 2010), such as lists of key figures, locations, artifacts, and dates. One useful concept which could make part of this metadata is that of a *named entity*, a term coined for the Sixth Message Understanding Conference (MUC-6) (Grishman and Sundheim, 1996). A named entity is a referent which has one or more rigid designators (Nadeau and Sekine, 2007). A referent is something to which those rigid designators refer. In the field of modal logic, Kripke (1972) defined a rigid designator as a name which has only one possible referent in all possible worlds. Common examples of such rigid designators would be the names of people and locations.

In the digital humanities, researchers are often interested *events* (what happened when and with who?) (Sporleder, 2010). Named entities are frequently important to that end as they often constitute important elements of an event. Identification of such entities is the task I will take on here. Named entity recognition (NER) attempts to find sequences of tokens corresponding to named entities, then to classify those sequences into one of the named entity (NE) types being examined. At present, the standard approach to NER is to treat the task as a supervied sequence labelling problem. These supervied systems, such as the Stanford Named Entity Recognizer (Finkel et al., 2005), are typically trained on corpora labeled with named entities provided by conferences such as MUC and CoNLL, consisting

of newswire from recent history. When these systems are ported to a new domain such as historical newsapers (Mac Kim and Cassidy, 2015), fewer entities are identified and more entities are incorrectly identified. It is also often not possible to train these systems on the desired domain due to the unavailability of a sufficient amount of labeled data.

The period of Early Modern English (c. 1500-1700) was one which saw considerable change in the language, both in speech and writing (Nevalainen, 2006). While standard spellings increased from Modern English, there was still considerable variation. Orthographic choices about capitalization and punctuation were often left to the writer and not some accepted norm or standard (Salmon, 2000). While Early Modern English is intelligible to the present-day English speaker (many students in American and British schools read Shakespeare in secondary school), it is quite different from Modern English. All of these confounding factors would complicate the use of NLP systems trained on Modern Standard English on data of this type. While there have been several attempts at carrying out NER on historical data (Borin et al. (2007), Grover et al. (2008), and Won et al. (2018)), few have looked at data from the period of Early Modern English. Moreover, as far as I can tell, no attempt has been made to explicitly tackle the task of named entity recognition in this domain using neural embedding techniques. In this thesis I will investigate the considerations and issues present when carying out named entity recognition on Early Modern English correspondence. My research hypotheses were as follows:

(1) Training data of sufficient quality can be bootstrapped to train a character-based neural net for named entity recognition. In this context, sufficient quality means that the data does not have so many errors as to result in a model which performs more poorly than the methods used to bootstrap the training data.

(2) A character-based neural net trained on bootstrapped data will perform better than a standard off-the-shelf named entity tagger such as the Stanford Named Entity Recognizer.

This thesis begins with Chapter 2 where I conduct a review of prior work done on various relevant topics, such as NER on historical documents and NER using neural nets. In Chapter 3 I describe in detail the methodology I used for bootstrapping training data and subsequently training a neural net model for named entity recognition. Having made some observations about the first experiment I carried out, I devised a second experiment which I motivate in Chapter 4, where I attempt to create training data with a higher recall than in the first experiment. In Chapter 5 I present my results and in Chapter 6 I discuss them. Finally, my conclusions are found in Chapter 7.

Chapter 2
LITERATURE REVIEW

While there has been considerable work done on named entity recognition (NER), two bodies of work are particularly relevant for the investigation at hand: NER on historical texts and NER using word and character embeddings which are discussed in Sections 2.1, 2.2, and 2.3. Also relevant for the topic at hand is the bootstrapping of training data when there is no labeled data available in the domain, which is discussed in Section 2.4.

2.1 NER on Historical Texts

Numerous researchers have investigated NER on historical texts. Earlier attempts frequently made use of resources such as gazetteers and relied on various rule-based methods. More recently many researchers have looked into off-the-shelf NER systems and how they perform on the domain of historical texts. An off-the-shelf system is a system which already has trained models which people can use to carry out NLP tasks on their own data (such as part-of-speech tagging and named entity recognition).

2.1.1 Knowledge Bases and Rule-Based Methods

Investigating NER on 19th century documents, both Jones and Crane (2006) and Borin et al. (2007) tagged named entities using a pipeline of modules. The system in Jones and Crane (2006) used four modules and tagged not just the most commonly addressed NE types (people and locations), but a total of ten types including ship names, dates, and railroads. These additional NE types were included as the authors found them relevent to the 19th century newspaper dataset they were working with. In the NER pipeline, the first and second modules consisted of knowledge base term lookups. The knowledge base was created from

"encyclopedias, gazetteers, directores and other similar reference works". The third module was rule-based and the final module used language models to find NEs and assign them the correct tag. These language models were based on counts of the senses of a word such as how often the word *Washington* is used to describe a person versus a place, and counts of certain collocations involving named entities. The pipeline achieved varying accuracy across the ten NE types; places and dates were tagged quite well with accuracies around 97%, but product and newspaper names were not, with accuracies of 57.58% and 66.58% respectively. While the additional different NE types are indeed useful in the field of digital humanities, tagging of such NEs is out of the scope of this thesis. The methodology followed herein is unsupervied and relies on the assumption that if the names contained in a corpus are Brown clustered (Brown et al., 1992), the clusters with the most and second most tokens will represent people and locations respectively (Brooke et al., 2016). It is unlikely that similar assumptions based on cluster token counts could reliably be made for other less common NE types.

The pipeline used in Borin et al. (2007) consisted of six modules and begins with a lookup of mulltiword names from a knowledge base. This knowledge base consisted of names extracted from websites and various corpora. The next module performs shallow parsing using finite state grammars for each NE type. The following module attempts to find arguments in sentences which must be animate, while the one after that does a lookup of single-word names. After this, the next module leverages orthographic similarity of names to find yet more named entities. Finally, the last module does a final check of the annotations, attempting to fix errors, combining fragmented annotations, and other similar tasks. On their corpus of Swedish literature, the authors achieved an F-score of 92%.

Grover et al. (2008) looked at parliamentary records from the early 19th and late 17th century, the latter of which is the end of the period of Early Modern English (EME). These records had been digitized through optical character recognition, and therefore contained a moderate amount of noise. Grover et al. (2008) also employed a pipeline, but theirs was a pipeline consisting entirely of rule applications and term lookups. The rules consisted of templates for various NEs such as monarchs and churchmen. The system presented achieved

similar levels of accuracy for both time periods, hovering around an F-score of 70%. What Jones and Crane (2006), Borin et al. (2007), and Grover et al. (2008) have in common is that they all employ rule-based methods and term lookups. Rule-based methods, while they can achieve high precision, achieve lower recall and typically are very costly to develop; it can take a lot of time and effort for trained professionals to craft these rules (Kapetanios et al., 2013). As for knowledge bases, gazetteers available today likely are comprised mostly of Modern English. It is possible that EME is too different from Modern English in its names and spellings for these gazetteers to make an appreciable difference in accuracy of a NER system for EME.

2.1.2 Off-the-Shelf NER Systems

Diverging from the methods discussed above, another approach is to treat NER as a sequence labelling problem due to the fact that whether a given word is an NE is largely dependent on its context, i.e. the words which appear around it. In that vein, Nissim et al. (2004) used the maximum entropy tagger of Curran and Clark (2003) to tag locations in the *Statistical Accounts of Scotland* from the 1790s and the 1830s. They used the standard features of the tagger comprising orthographic, morphological, part-of-speech tag, and NE tag history. They achieved an F-score of 94%, comparable with the state-of-the-art at the time for modern English newswire. While the off-the-shelf tagger performed well on locations, the accuracy of a maximum entropy tagger could end up being quite different for other NE types.

More recently, Mac Kim and Cassidy (2015), Ehrmann et al. (2016), and Won et al. (2018) evaluated more recent off-the-shelf NER systems on historical data. Mac Kim and Cassidy (2015) evaluated two versions of the Stanford NER (Finkel et al., 2005) on OCRed (optical character recognition) Australian newspapers from 1803 to 1954. The Stanford NER, like Nissim et al. (2004) above, treats NER as a sequence modelling task, using a conditional random field (CRF) to find and tag NEs. Finkel et al. (2005) reported an F-score of 92.5, obtained on their dataset which consisted of newswire and seminar announcement emails. The first version of the Stanford NER in Mac Kim and Cassidy (2015) is the pre-trained

version (trained mostly on modern English newswire), while the second version is trained on 600 randomly selected articles from their database, which were annotated with the pre-trained Stanford NER. Both versions resulted in a significant decrease in performance when used on the digitized historical newspapers compared to the F-score reported in Finkel et al. (2005). The use of in-domain training data with the Stanford NER did not increase accuracy on the historical data and in fact resulted in a lower F-score score (67 as opposed to 71). While the decrease in performance is likely at least in part due to OCR noise, it is also possible that the Stanford NER is not well-adapted for this historical domain both in the pre-trained and self-trained cases. For the pre-trained version this could be due to the domain and period of English it was trained on being too different from the English represented in historical texts. As for training the Stanford NER using one's own data, as the results in Mac Kim and Cassidy (2015) suggest a large amount of tagged data is needed to train a powerful model, and these resources are much less available in the historical domain.

Ehrmann et al. (2016) did a an evaluation on of four NER systems on a diachronic set of corpora consisting of French newspaper articles from 7 one-year spans between 1804 and 1981. While this work concerns a French and not English, it is still situated within the body of research on historical NER. Historical French texts would differ from modern French texts much like Early Modern English texts differ from Modern English texts. Antiquated vocabulary and inadequate knowledge bases for historical NEs would vex researchers of both historical English and French texts.

The four systems used in Ehrmann et al. (2016) were a "symbolic system with ExPRESS, supervised machine learning with mXS[1] and proprietary web services offering NER functionalities with AlchemyAPI[2] and DandelionAPI[3]". ExPRESS is based on regular expressions over flat feature structures which use language-specific lexicons to obtain the feature structures, but which are language-independent (Piskorski, 2008). Using previously extracted

[1]https://github.com/eldams/mXS

[2]AlchemyAPI has since been acquired by IBM

[3]https://dandelion.eu/

patterns for NEs, mXS finds probabilities of a given sequence being an NE (Nouvel et al., 2011). These patterns were extracted from an annotated corpus using data mining techniques. Both DandelionAPI and AlchemyAPI use knowledge graphs, but AlchemyAPI also uses supervised classification methods for disambiguation. As one might expect, with all years combined the rule-based system had the highest precision (67.6 for PERSON, 84.6 for LOCATION), but poor recall. AlchemyAPI achieved the best F-score of 49.% for PERSON, while Dandelion achieved the best for LOCATION at 64.3%. In general all systems performed worse on the historical data than they did on the baseline data which consisted of transcribed news broadcasts from the year 2010. The difference in F-scores ranged from approximately 2 to 50 points. Diachronically, the authors did not observe a clear increase in precision over time; there was a slight increase for PERSON but a decrease for LOCATION. In terms of recall, scores for LOCATION were generally stable while there was an increase for PERSON. It is possible that the lack of a clear upward trend for both NE types for precision and recall is due to the fact that the French used in their data did not significantly change from the first to last time span. A diachronic evaluation over a longer period of time would be worthwhile but out of the scope of this thesis.

Won et al. (2018) extracted locations from letters in a collection dating between 1764 and 1819, and letters from another collection dating in the mid-1600s. These latter papers were written mostly in EME. The authors evaluated several systems individually, then combined them and evaluated the ensemble. The systems used were the Stanford NER (Finkel et al., 2005), NER-Tagger[4], the Edinburgh Geoparser[5], spaCy[6], and Polyglot-NER[7]. The Edinburgh Geoparses is a rule-based system which finds people and place names using lexicon lookups. The methods used by spaCy are not outlined in any particular work, however the website states that the NER model "is a greedy transition-based parser guided by a linear

[4]http://github.com/glample/tagger

[5]http://www.ltg.ed.ac.uk/software/geoparser/

[6]http://spacy.io

[7]http://polyglot.readthedocs.org

model whose weights are learned using the averaged perceptron loss". Polyglot-NER (Al-Rfou et al., 2015) obtains word embeddings from unlabeled Wikipedia text and uses them to train a word-level classifier. Out of the individual systems the Stanford NER performed best on the older letters with an F-score of 70%, and Polyglot performed best on the more recent letters with an F-score of 62%. The ensemble increased performance by approximately two points on the older data, and 10 points on the newer data. Overall what all the above-mentioned research demonstrates is that NER on historical texts is not a trivial task, and systems built for other domains do not usually port well to this domain.

2.2 NER with Word Embeddings

Leaving the domain of historical texts, many researchers have employed word embeddings to improve sequence predicting models by using the embedding vectors as features in those models. Turian et al. (2009) gave a preliminary investigation of the use of different kinds of embeddings in a regularized averaged perceptron model, as outlined in Ratinov and Roth (2009). They found that all embeddings improved NER performance but Brown clusters showed the most improvement. Five years later Guo et al. (2014) revisited the topic and investigated different ways in which word embeddings could be used as features in linear NLP models, evaluating the methods using a CRF.

Many researchers besides Guo et al. (2014) have used word embeddings as features for training a CRF. Ma et al. (2016) word embeddings enriched with POS tags, taxonomic information, and tagging results from a baseline NER tagger to train their CRF. To obtain the enriched word embeddings they modified the Skip-gram neural net model (Mikolov et al., 2013) such that it would predict these features given a word, instead of the next word in a sequence. Similarly, Passos et al. (2014) enriched their word embeddings with lexicon membership information. Kulkarni et al. (2016) used domain-specific word embeddings for domain adaptation of a CRF named entity recognizer. While all these researchers achieved good results in their use of word embeddings in linear models, this approach is less viable in the domain of historical text. Even though word embeddings can be obtained in an

unsupervised fashion, training these sequence predicting models still requires large amounts of annotated data. Such annotated corpora are much less available for historical text.

Rather than treating NER as a sequence labelling problem, some researchers have approached it as a classification problem. Demir and Ozgur (2014) used word embedding cluster labels as features in a regularized averaged perceptron to create a NER system for the morphologically rich languages of Turkish and Czech. Das et al. (2017) trained a random forest purely with word embeddings themselves as a methodology to be used for resource-poor languages, in their case Bengali. Both of these works used data written in the modern form of the languages studied, taken from news sites in Demir and Ozgur (2014) and from Wikipedia in Das et al. (2017). Given that there are significant differences between the English spoken today and EME, it is unlikely these methods would be as effective for this domain. Brooke et al. (2016) investigated NER in the domain of literature using logistic regression for classification. Instead of classifying word embeddings, however, they used non-neural context vectors. This choice was made address the fact that literature frequently contains numerous rare and invented names. One might expect that texts written during a different epoch of English, such as EME, might also share this characteristic. The way Brooke et al. (2016) model rare names is by gathering a pool of common names (appearing at least 100 times in the whole corpus), then passing over the corpus looking for instances of these names. When one is found in a given document, a context vector is created for all instances of that name in that document. A generally common name can be rare in one piece of literature where that person or location doesn't play a big role in the narrative, while also being common in a different piece of literature. In this way both common and rare name contexts can be modeled. This method relies on a sufficiently large corpus to create a pool of common names, as well as sufficiently large documents to be able to have many instances of a name. As will be discussed in Section 3.1, the corpora used in this project are not amenable to this methodology. I chose to replicate, however, the authors' method of bootstrapping training examples, which will be touched on in Section 2.4 and explained in more detail in Section 3.2.

2.3 NER as a Character-Level Task

In order to address data sparsity issues, some researchers have built systems which rely on character-level representations of a word as opposed to or in conjunction with the word itself. Klein et al. (2003) built a Hidden Markov Model (HMM) using character n-grams. The states are pairs of NE tags and character positions within the current word. The character position is used to ensure that NE tags are consistent within a word. The HMM, while it did not perform as well as the state-of-the-art, out-performed the CoNLL 2003 baseline by about 12 points with an F-score of 83.2%. The authors also built a character-based maximum entropy Markov model which used other features as well such as POS tags, mixed case markers, and others. This model outperformed their HMM significantly, obtaining an F-score of 92.27%.

Kuru et al. (2016) use character-level recurrent neural networks, specifically stacked bidirectional long short term memory (LSTM) networks, to tag NEs in a number of languages. Spanish, Dutch, English, and German datasets were provided by CoNLL-2002 and CoNLL-2003. Additionally, data in Arabic, Czech, and Turkish were used due to their rich inflectional morphologies. The LSTMs used map characters sequences to tag sequences, giving tag distributions for each character position. In this output the most probable tags for each character within a word are not guaranteed to be the same. That is to say it is possible for the first three characters of a word to have PERSON as the most probable tag and the last three characters to have LOCATION as the most probable tag. For this reason, Viterbi decoding is used to find the most probable tag sequence for each sentence. The transition matrices used by the decoder only allow transitions which result in consistent within-word tagging. Because the authors used no external language-specific resources such as gazetteers, they compared their results with the state-of-the art for each language examined both using and not using external resources. While the state-of-the-art which used external resources outperformed their system, their results tended to be on-par with the state-of-the-art which did not use external resources. As the authors note, these results are also achieved without the use of hand-crafted features.

2.4 Bootstrapping Training Data for NER

Bootstrapping is defined by Abney (2002) as "a problem setting in which one is given a small set of labeled data and a large set of unlabeled data, and the task is to induce a classifier". For the task of domain adaptation Wu et al. (2009) developed an algorithm for iteratively improving the performance of a classifier trained in one domain on another target domain. Their algorithm attempts to find data points in the unlabeled data which are "more informative" and add those to the classifier. More informative data points are those which act as bridges between the source and target domain, meaning they are present in both. Their results show significant improvement over bootstrapping systems which do not use those selection criteria.

Some researchers have also looked at bootstrapping training data to be used on its own, and not to improve a pre-trained supervised classifier. Kozareva (2006) devised an algorithm for automatic gazetteer extraction which is then used on its own and to augment a pre-trained classifier. Their results showed that including the gazetteer information in the supervised classifier improved performance but the bootstrapped system did not perform comparably, attaining F-scores of approximately 33% to 63% for their four NE types. Using Wikipedia entries to automatically generate gazetteers for Bengali, Das et al. (2017) train a random forest classifier using these gazetteers as seed lists in their dataset. Better results were attained in this work but the combined F-score of all NE types was still only 65.4%.

The work of Mac Kim and Cassidy (2015) was discussed in Section 2.1.2 but is also relevant to the topic of bootstrapping training data. In a similar attempt at obtaining training data for a low-resource domain, the authors created training data by using the pre-trained Stanford NER to tag 600 historical newspaper articles dating back to 1803. They then used these 600 articles to train a new Stanford NER classifier on their domain of data. Their results were better yet than those in the last two works mentioned, but still had room for improvement with a combined F-score of 67% for all NE types. In fact, their domain-trained classifier performed worse than the pre-trained classifier, which obtained a combined

F-score of 71%. While their attempts at bootstrapping training data did not pan out, it inspired one of my experiments which is detailed in Section 3.2.

Finally, Brooke et al. (2016) bootstrapped training data in a way that ended up yielding good results with an F-score of 79.2%. These authors used their previously developed tool GutenTag (Brooke et al., 2015) to find potential names in their corpus (English fiction texts in the Project Gutenberg corpus) based on capitalization patterns. Brown clusters were then induced on the corpus and the clusters corresponding to people and locations were identified. Potential names in these clusters were used as seeds for their classifier, subject to some criteria. The method of obtaining training data used in Brooke et al. (2016) will also be followed here, with some modifications as will be explained in section 3.2.

2.5 The Approaches Followed Here

What I set out to do in this thesis is investigate the efficacy of three methods of bootstrapping training data for use in a neural character embedding named entity recognition system. The first two bootstrapping methods come from Brooke et al. (2016): I will use the name tagger of the GutenTag system (Brooke et al., 2015) to identify potential names in the training data. I will then induce Brown clusters on the data and locate the two clusters corresponding most to people and locations. The cluster information and potential name tagging will then be used to tag words in the training data as NEs. In Brooke et al. (2016) Brown clusters were shown to be able to group people and locations relatively neatly into separate clusters. If the same is true for my dataset, important contextual information for the two NE types could similarly be gleaned using character embeddings. Moreover, because this method of bootstrapping does not rely on outside knowledge bases such as gazetteers, it is possible for it to catch more words which would likely be out-of-vocabulary in pre-trained systems.

The third bootstrapping method follows the work of Mac Kim and Cassidy (2015), using the pre-trained Stanford NER to tag my training data and using that as input to the character embedding system. In their work, the pre-trained Stanford NER worked moderately well for tagging named entities in historical news articles on its own, but using its own output as

input for training did not result in a better-performing system. The authors hypothesize that more training data would be needed to train a better model. The method I will approach here may be less likely to suffer from this issue, as I will be looking at the data at a character level as opposed to the word level, greatly increasing the relative amount of training data.

The neural character embedding system I will use is that of Kuru et al. (2016). These researchers used a character-based stacked bidirectional LSTM network for the use of tagging named entities. This method is advantageous for my EME dataset for two reasons. First, as mentioned above, using character-level information increases the relative amount of training data compared to looking only at words. This allows me to work around the lack of availability of large amounts of EME data. Second, looking at characters as opposed to words avoids the issue of out-of-vocabulary words, of which I would expect there to be many due to the nature of EME data. Complete details surrounding the methodology used in this thesis follow in the next section.

Chapter 3
METHODOLOGY

This chapter begins with a description of the datasets used and how I prepared them in Section 3.1. The next two sections explain in detail the steps shown in Figure 3.1. Section 3.2 focuses on the methods used to bootstrap training data, while in Section 3.3 I describe the character embedding named entity recognizing system of Kuru et al. (2016) which I used for this project.

3.1 Data

3.1.1 Description of the Data

Data examined for this project consists of texts written in Early Modern English (EME) from the years 1500-1700. This period of English is characterized by an increase in standard spellings, however the language also underwent considerable change during that time (Nevalainen, 2006). Despite the increase in spelling standardization, choices about capitalization and punctuation were often left to the author (Salmon, 2000). These characteristics would present difficulties for systems trained on Modern English. As an example, such a system would rely in part on capitalization of proper nouns to find named entities. If proper nouns were not capitalized (or if common nouns *were* capitalized), the system might end up tagging the word incorrectly. A more in-depth discussion of how the features of EME orthography affect NER system performance can be found in Chapter 6.

In order to study NER on EME texts, I used three corpora, the first of which is the Corpus of Early English Correspondence Sampler (CEECS) (Nevalainen et al., 1998). This corpus consists of letters written by various authors from the years 1418 to 1680 with a total word count of 450,085 (Nurmi, 1999). This date range exceeds that used for this project, so

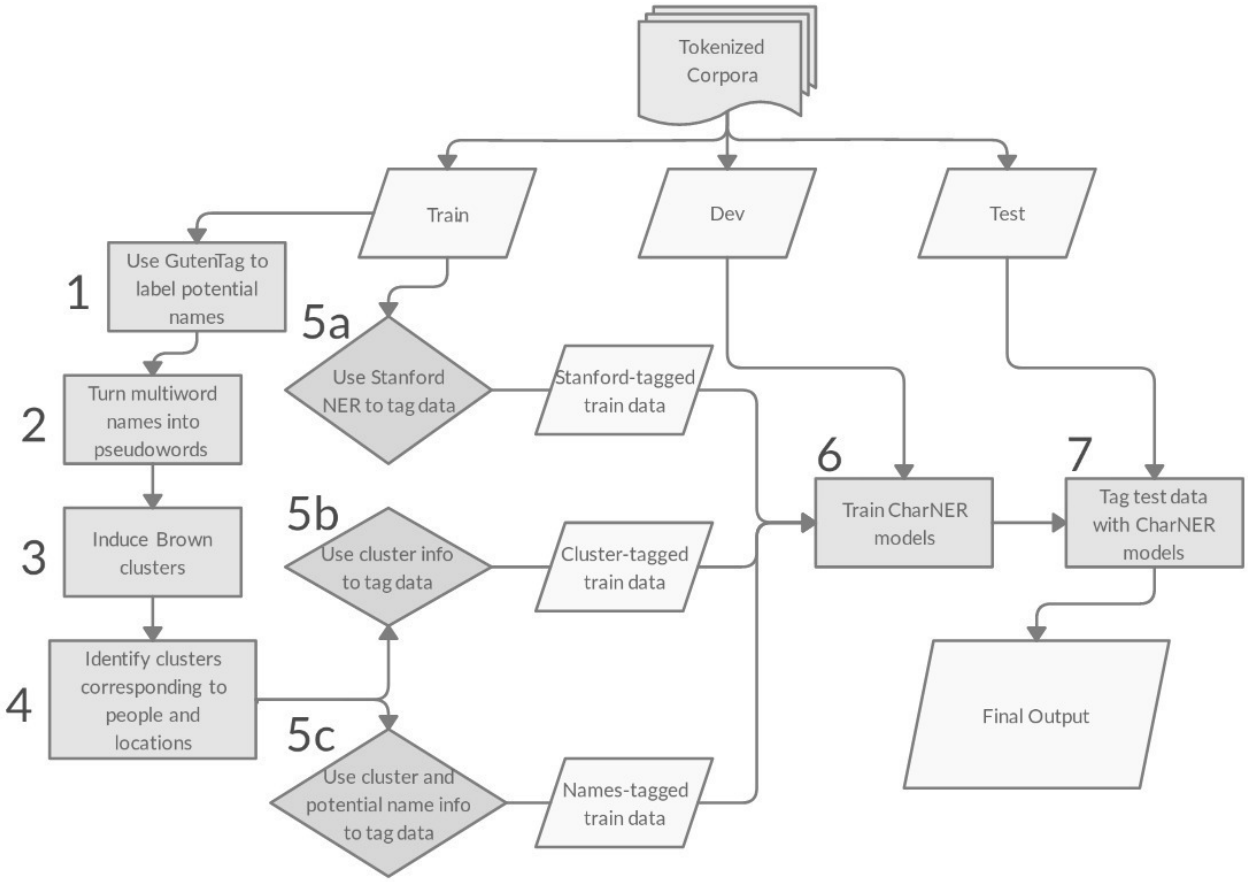


Figure 3.1: Methodology of training data creation and subsequent tagging of test data

only letters written after the year 1500 were used, bringing the total word count down to approximately 350,000.

The second corpus used is the Letter Corpus of the Innsbruck Computer Archive of Machine-Readable English Texts (Markus, 1999). This corpus consists of letters written from 1386-1698 and contains approximately 182,000 words. Since only letters written beginning in the year 1400 were used, this corpus's addition to my project was approximately 74,000 words.

The final corpus included in this project is the Newdigate Newsletters (Hines, 1995). The newsletters were issued by the office of the Secretary of State in England and addressed to Sir

Richard Newdigate. I could not find information on whether authorship of the newsletters changed over the period of correspondence I investigated. The collection includes letters written from 1673 to 1688 and contains approximately 1,000,000 words. Altogether the data used for this project consisted of approximately 1.5 million words. The breakdown of word counts for each corpus and each dataset are shown in Table 3.1.

	train	dev		test	
	words	words	entities	words	entities
CEECS	356,319	7,264	202	6,890	209
Innsbruck	76,436	1,798	27	1,651	48
Newdigate	1,010,655	21,051	978	20,509	1020
total	1,443,410	30,113	1207	29,050	1277

Table 3.1: Word and NE counts for training, dev, and test sets for each corpus.

Brooke et al. (2016) do not provide a total word count of the corpus used for their project, but describe which texts they use. From the Project Gutenberg[1] corpus of books they use only English fiction documents, resulting in a total of 10,844 texts. One work of fiction included in Project Gutenberg is the novel *Clarissa, or, The history of a young lady* (Richardson, 2014) and consists of nearly one million words (World Heritage Encyclopedia). While most novels are not one million words in length, the fact that merely one novel in Project Gutenberg amounts to almost as many words used for this project, as well as the fact that Brooke et al. (2016) used a total of over 10,000 texts, it is very likely that their total corpus was considerably larger than mine. In addition, most individual letters in the corpora I used were less than 3000 words in length which is considerably shorter than most novels. The methods that Brooke et al. (2016) used rely on a sufficiently large corpus so as to obtain a pool of common names, as well as sufficiently large individual documents so

[1]http://www.gutenberg.org

```
His Maty has been pleased to appoynt Sir John Narborough to goe in the
Centurion to Algiers about the Matter of the Redemption of the Captives
there remaineing , who is accordingly fitting out with all hast & hopes to
sayle in two dayes yester Night
```

Figure 3.2: Tokenized sentence from the Newdigate corpus

as to have multiple instances of these common names contained within. For these reasons, the methodology used by Brooke et al. (2016) to model the contexts of rare names is not amenable to my corpus.

3.1.2 Preparing the Corpus

For this project, training, development, and test sets were required for use in the character embedding system. I tokenized each document in the corpora into sentences and individual tokens using the Spacy (Honnibal and Montani, 2017) tool for Python. Tokenized documents were written to files with one sentence per line and each token separated by whitespace as in Figure 3.2.

Once each document was tokenized, I concatenated all of the documents and shuffled the sentences with a random seed. Approximately 2% of data was held out for development (1,588 sentences) and 2% was held out for testing (1,595 sentences) while the rest was used for training. I annotated the development and test sets with people and location tags following the MUC named entity tagging guidelines in Chinchor et al. (1999) using the annotation tool, YEDDA (Yang et al., 2017).

3.2 Bootstrapping Training Points

One problem with NLP tasks in the domain of historical texts is that there is much less annotated data available for training models. Indeed, none of the datasets used in this project were annotated. Annotating by hand many thousands of sentences is often not

```
His Maty has been pleased to appoynt Sir John_Narborough to goe in the
Centurion to Algiers about the Matter_of_the_Redemption_of_the_Captives there
remaineing , who is accordingly fitting out with all hast & hopes to sayle
in two dayes yester Night
```

Figure 3.3: Tokenized sentence with pseudowords

feasible, so other methods need to be explored to create training data.

One method of bootstrapping I experimented with here follows Brooke et al. (2016) and corresponds to Steps 1-5c in Figure 3.1. In Step 1, the GutenTag tool (Brooke et al., 2016) finds potential names by leveraging capitalization patterns as well other heuristic methods such as common intervening words in names. As an example, capitalized words on either side of a word such as *of* or after *the* are tagged as names. The name extraction module accounts for sentence-initial capitalization by only tagging a word as a name if it appears at least once in the corpus in non-sentence initial position. Another heuristic used by GutenTag is the presence of address terms and their abbreviations such as *Doctor* and *Dr.*. The lists of such terms and their abbreviations used by GutenTag are good for Modern English, but there are quite a few which are used frequently in my corpora which are missing from those lists. As such I added further content to those lists such as *Sr.* for *Sir*. I also added all abbreviations without the terminating period, as this is common in EME texts as well.

Once these potential names are identified, Step 2 begins and multiword names are made into pseudowords in anticipation of the next step. The sentence in Figure 3.2 gets transformed as in Figure 3.3. As can be seen, the GutenTag algorithm has misidentified a string of words as a name due to their capitalization.

In Step 3 I induce Brown clusters Mikolov et al. (2013) using the tool of Liang (2005). Brown clustering is a method in which words are clustered together in vector space based on the words which appear around them. The default settings of Liang (2005) were used except that the number of clusters is reduced from 1000 to 50. The reasoning is that we

are attempting to find clusters which correspond broadly to people and locations, and not possible subclusters within those clusters. (This is also why multiword names are converted to pseudowords, as we want the clustering to capture a whole NE, even if it is multiple words.) The number 50 specifically was chosen for the number of clusters because it is approximately the size of the tagset used in the Penn Treebank (Marcus et al., 1993).

In Brooke et al. (2016), after induction clusters are ranked based on the number of name tokens contained within (as per GutenTag's name identification). The highest ranking cluster is taken to be the cluster corresponding to people and the second highest is taken to be the cluster corresponding to locations, while all other clusters are labelled as a catch-all OTHER category. It it worth noting that while this bootstrapping method is unsupervised, it relies on rules of orthography for English capitalization. Given this fact it would not necessarily be suited for other languages. In German, for example, all nouns are capitalized so one wouldn't be able to extract names based on capitalization patterns.

This ranking of clusters did not accurately identify the people and location clusters in my dataset, so in Step 4 I added a layer of supervision by examining the clusters manually and finding the two corresponding to people and locations. The words in these clusters constituted seed lists for subsequent tagging of the training data. There was considerable noise in the lists due to the lack of regular capitalization rules in written EME. As such, I filtered out seeds which contained stop words (excluding the common intervening words mentioned about). I started with the stopwords list used by Spacy, and added other EME stopwords (such as *tis* and *mee*) as well as EME spellings of the original stopwords by replacing *i* and *th* with *y* (so that *him* becomes *hym* and *the* becomes *ye*, etc.). Finally, in Step 5c tokens in the training data are tagged in the BIO scheme[2] as NEs based on whether they are both present in the people or location clusters and whether they had been tagged by GutenTag as potential names. The sentence presented in Figure 3.3 is transformed as in

[2]BIO stands for beginning, inside, and outside. For a multiword entity, the first word would be tagged as B and subsequent words tagged as I. Single-word entities are tagged simply as B, and non-entities are tagged as O.

```
His/O Maty/O has/O been/O pleased/O to/O appoynt/O Sir John/B-Person
Narborough/I-Person to/O goe/O in/O the/O Centurion/O to/O
Algiers/B-Location about/O the/O Matter/O of/O the/O Redemption/O of/O the/O
Captives/O there/O remaineing/O ,/O who/O is/O accordingly/O fitting/O out/O
with/O all/O hast/O &/O hopes/O to/O sayle/O in/O two/O dayes/O yester/O
Night/O
```

Figure 3.4: Named entity-tagged sentence

Figure 3.4. Notice that the phrase *Matter of the Redemption of the Captives* is not tagged as an NE, since in the clustering step this pseudoword was not a member of the people or locations cluster.

The second method of bootstrapping I experimented with consists of steps 1-5b in Figure 3.1. Steps 1-4 are carried out in the same way outlined above. In Step 5, tokens in the training data are tagged as NEs based on their presence in the people or location clusters, regardless of whether they had been identified as potential names by GutenTag. This was because I noticed that there were a number of names in the clusters which hadn't been identified as such due to a lack of capitalization, but were in fact NEs such as *ffrance* and *spaine*.

The final method of bootstrapping training examples I explored was inspired by Mac Kim and Cassidy (2015) and is Step 5a in Figure 3.1. Rather than using the Stanford NER's output as input to a domain-specific Stanford NER classifier, I used it as input to the character embedding neural net system. After the sentence tokenization step, the pre-tokenized data was fed into the Stanford Named Entity Recognizer (Finkel et al., 2005). As the Stanford NER does not output data in the BIO format, its output needed to be converted from IO to BIO. To do this I changed any contiguous sequence of I tags into B followed by I.

In each of the three bootstrapping methods (5a, 5b, and 5c), I also experimented with which taggings should be kept or excluded in an effort to increase precision and recall. These

experiments are explained in detail in Chapter 4.

3.3 NER with Character-Based Neural Networks

In Step 5 I use the various bootstrapped training data sets to train a CharNER (Kuru et al., 2016) character embedding neural net model and in Step 6 I use that model to tag the test data. While the method outlined in Kuru et al. (2016) was supervised in that it used annotated corpora for training, the bootstrapping method of Brooke et al. (2016) could also be used to create training examples in an unsupervised fashion. As with the training data used in Kuru et al. (2016), word-level tags must be converted to character-level tags. If a given word is tagged as a named entity such as PERSON, then when that word is split into characters each character is assigned the PERSON tag. In the case of phrasal NEs the conversion is similar. Each character of all the words will receive the NE tag of the whole word, as will any spaces between those words. Characters which are not part of an NE are tagged O indicating they are outside any NE.

Once the data is in the appropriate form, the methodology of Kuru et al. (2016) can be followed. The character-based data is used to train stacked deep bidirectional long short-term memory (BLSTM) neural networks in order to predict a tag sequence for a character sequence. The architecture of the character-based neural net system is shown in Figure 3.5. Long short-term memory (LSTM) networks are a type of recurrent neural network (RNN). LSTMs were developed by Hochreiter and Schmidhuber (1997) to address the issue with the original RNNs that they do not handle long-term dependencies well. The reason for this is that vanishing gradients result in a system which is either overly sensitive to noise or overly inefficient (Bengio et al., 1994). LSTMs contain a special memory cell which controls which information should be added and taken away from a cell. There exist many variations of the LSTM; that presented by Gers and Schmidhuber (2000) was used here.

While a regular LSTM uses only previous context to extract features, it may also be desirable to use the following context. Indeed the context window used by Brooke et al. (2016) above used both preceding and following words around the target token to derive features

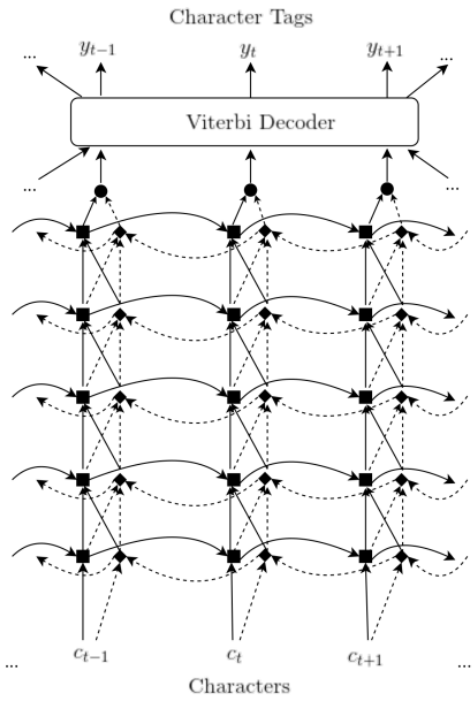


Figure 3.5: "CharNER model: 5-layer Bidirectional LSTM Network with a Viterbi decoder. Each square and diamond node represents a forward and backward hidden LSTM layer, respectively. Circles denote the output layer (i.e. softmax layer). Solid lines show forward connections and dashed lines show backward connections. The model takes characters as an input sequence and each character is represented with a one-hot vector (c_t). A Viterbi decoder takes the sequence of character tag probabilities that is produced by the softmax layer and produces most likely character tag sequence (y) that is consistent at the word level." (Kuru et al., 2016)

for that token. Graves and Schmidhuber (2005) developed a bidirectional LSTM in order to add this functionality. In this model two LSTMs are used to incorporate information about future inputs: a forward LSTM and a backward LSTM. The forward LSTM proceeds forward through an input sequence while the backward LSTM proceeds backward through that same sequence. Finally, following Graves et al. (2013) a deep BLSTM was used, specifically a network consisting of five layers.

The output of this neural network does not necessarily output the same NE tag for each character of a word. In the name *Sutton* for example, the first three characters could receive the PERSON tag while the last three receive the LOCATION tag. A Viterbi decoder (Viterbi, 1967) is used to find the best sequence of states for each character in a given sentence. The probability distributions output by the BLSTM are used as emission probabilities. The decoder enforces the use of the same tag within a given word by using transition matrices which only allow three types of transition: 1) transition from one NE tag to the same NE tag; 2) transition from an NE tag to O (the "outside NE" tag); and 3) transition from O to

an NE tag.

The advantages of using a character-level model as opposed to a word-level model for my dataset are twofold. First, it mitigates the effects of limited amounts of data: by treating each character as a token, the dataset will have a much greater number of tokens than if it were used at the word level. Second, a character-level model would mitigate the effects of nonstandard spellings and capitalization which would result in out-of-vocabulary words in a word-level model.

3.4 Evaluation

I used three baselines for the purposes of evaluation in this project. The Stanford Named Entity Recognizer, being a widely used off-the-shelf system with good performance for modern English (as well as other languages) (Finkel et al., 2005), is the first. The second and third methods involve using Steps 1-5b and Steps 1-5c to tag the test data directly. All three baselines allow for a comparison of the bare bootstrapping method with the use of the bootstrapped training data to train a character-based neural net model. I used the evaluation script developed for the CoNLL-2000 chunking shared task (Sang and Buchholz, 2000) for the purposes of evaluation. This script was also subsequently used for the CoNLL-2003 language-independent NER shared task (Sang and De Meulder, 2003) as it uses the same format of input. In the evaluation used therein, a tagged named entity is only considered correct if both its span and its tag type match that of the gold standard. That is, no credit is given for having the correct span but the wrong NE type or for having the correct NE type but the wrong span.

Chapter 4

BOOTSTRAPPING: SOME INITIAL CHALLENGES

The first two experiments for bootstrapping training data branch from the work of Brooke et al. (2015). In this paper the authors used a heuristic method of tagging potential names then Brown cluster the corpus. The two clusters with the highest number of potential name tokens are taken as the clusters referring to people and locations respectively. Then only the words in these clusters which had been identified as names were given NE tags and used as training data. Brooke et al. (2015) used context vectors to train a classifier whereas I used a character-based neural net. Because training data is bootstrapped, it suffers from noisy tagging. That is, it can't be guaranteed that if a token is not tagged as an NE it is because it is not an NE. As such, in all experiments sentences which contained no NE tag were removed from the training set to avoid training a model that certain words are not NEs when in fact they are. This does not completely eliminate this problem as there will most likely be sentences in which one NE is tagged but one or more are not. Experimentation with whether the removal of sentences with no tagged NE was beneficial was out of the scope of this thesis. It is possible that removing them could result in the model not learning contexts indicating that a word should *not* be tagged as an NE. Further experimentation would need to be done to determine the benefit.

In following the method of Brooke et al. (2015) for bootstrapping, I made two observations about the clusters and the potential names within when processing my data. First, there was a considerable amount of noise in both the whole clusters and the subsets of those clusters which had been identified as potential names. *Absconded*, for example, was both tagged as a name and was present in the cluster corresponding to locations. My second observation was that the clusters contained many names which hadn't been tagged as potential names

by GutenTag. As an example, a commonly occurring region in my corpus, *fflanders*, was not identified as a potential name (likely due to its lack of capitalization) but was still part of the cluster corresponding to locations. I devised two experiments to attempt to address these issues.

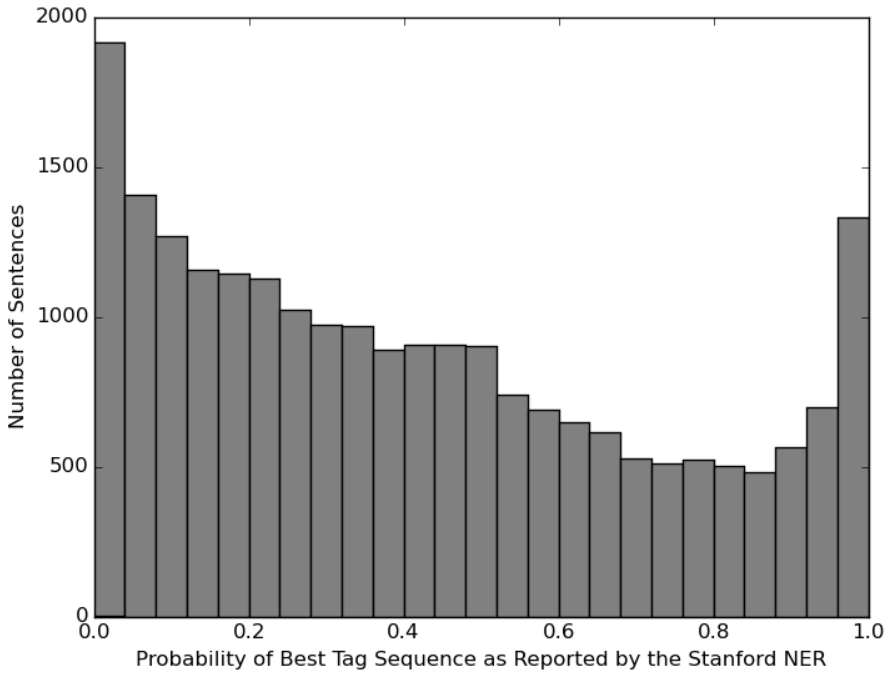
In the first experiment I follow Brooke et al. (2015) more closely and only tag tokens as an NE if they are both part of one of the selected clusters and were identified as a potential name by GutenTag. Brooke et al. (2015) had an additional criterion for use of a token as a name, which is that a token would only be tagged as an NE if it was a common name (appearing more than 100 times in the corpus). Given that my dataset was considerably smaller, this same criterion couldn't be used. In both the selected clusters there were fewer than five names which occurrred over 100 times. As such, I experimented with using a range frequency thresholds (from 0 to 100 in increments of 10) to determine whether to tag a given token as an NE. To address the second issue identified above, I also looked at using members of the clusters which had not been identified as potential names. In this experiment, every word in the cluster is tagged as an NE using the same frequency criterion as in the first experiment. Namely, a range of frequency thresholds were used to create the training data.

The final experiments make use of the off-the-shelf NER tool, the Stanford Named Entity Recognizer (NER) (Finkel et al., 2005). Tokenized sentences were used as input to the Stanford NER and its output was used as training data for the character embedding system, similar to the work of Mac Kim and Cassidy (2015). In Mac Kim and Cassidy (2015) they used the pre-trained Stanford NER's output to train a new Stanford NER model, whereas I use its output to train the neural character embedding system. On my development data the Stanford NER did not perform exceptionally well with its precision at 57.46%. In an attempt to create better quality training data, I devised this last experiment. The Stanford NER allows the user to output the probability of the best tag sequence of a given sentence. The distribution of these probabilities for my training data are shown in Figure 4.1. As can be seen, there are a large number of sentences whose best tag sequence has a very low probability. If the probability of a tag sequence correlates with whether that tag sequence is

correct in my dataset, it may be beneficial to remove sentences whose tag sequence is very low. As such, I ran experiments where the top $k\%$ of sentences were used in the range of 10 to 100 in increments of 10.

In all the experiments the CharNER system ran for 600 epochs using 128 nodes (the default settings of the CharNER system) attempting to find the parameters which resulted in the best f-score on the development data. These parameters were then used to tag the test data which was in turn evaluated against the gold standard tagged data.

Figure 4.1: Probability distribution of the best tag sequences for all sentences in the training set as reported by the Stanford NER



Chapter 5

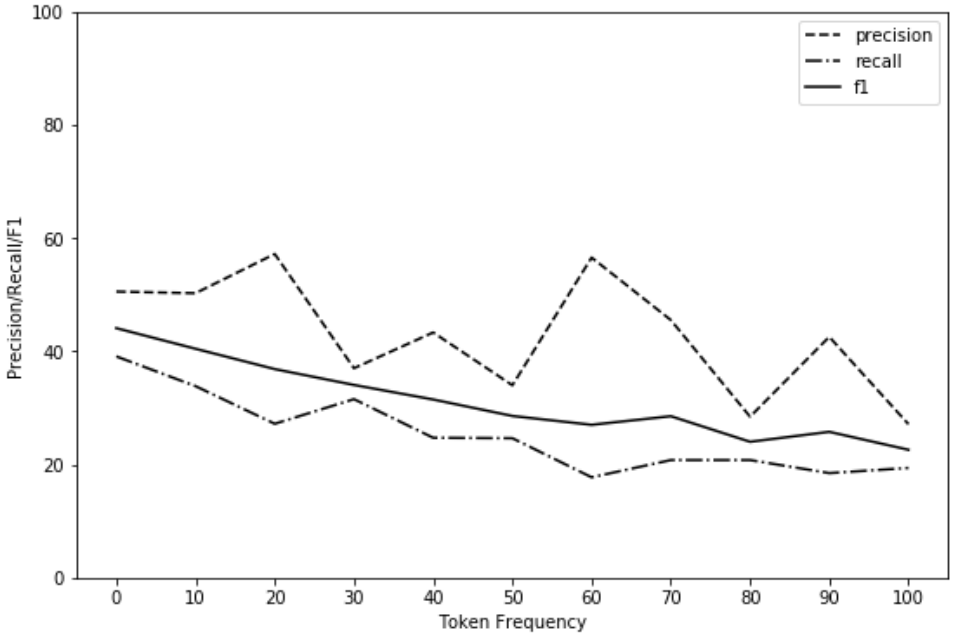
RESULTS

The results of the various experiments on the dev and test sets are found in this chapter. The dev set was used to tune the following hyperparameters: epoch of the neural net, word frequency threshold (in the *names* and *cluster* experiments), and the top $k\%$ of sentences to be used (in the Stanford NER output experiment). The time taken to train in a single epoch ranged from approximately ten to thirty minutes. The results of using potential names as training points with a range of frequencies in the training data can be seen in Figure 5.1. Best f-score and recall results were seen when the minimum frequency of a word was 0, meaning all potential names in the people and location clusters were used. It is a similar situation for the results of using all words within the chosen clusters, as shown in Figure 5.2. In both cases, the higher the frequency threshold (and thus the greater the excluded instances), the more that overall performance decreased. It is unlikely that a high-precision tradeoff could be useful for the present work, as there were very few unique tokens in the PEOPLE and LOCATION clusters with frequencies over 90 (three in the people cluster and four in the location cluster). A comparison of the change in f-score between the two methods is shown in Figure 5.3. The f-score is slightly higher in the *cluster* experiment as opposed to the *names* experiment.

The results on the dev set of using the Stanford NER output is shown in Figure 5.4. There is general trend upward as k increases from 10% to a peak in performance when $k = 70$.

The best model of the three experiments was used to tag the test set, which was subsequently evaluated. The precision, recall, and f-score of the test results, along with those of the three baselines are shown in Table 5.1. As can be seen, each experiment substantially out-performed its corresponding baseline, with the *stanford-out* obtaining the highest

Figure 5.1: Results on the dev set of using potential names only to bootstrap training data



precision, recall, and f-score.

Figure 5.2: Results on the dev set of using all words in the cluster to bootstrap training data

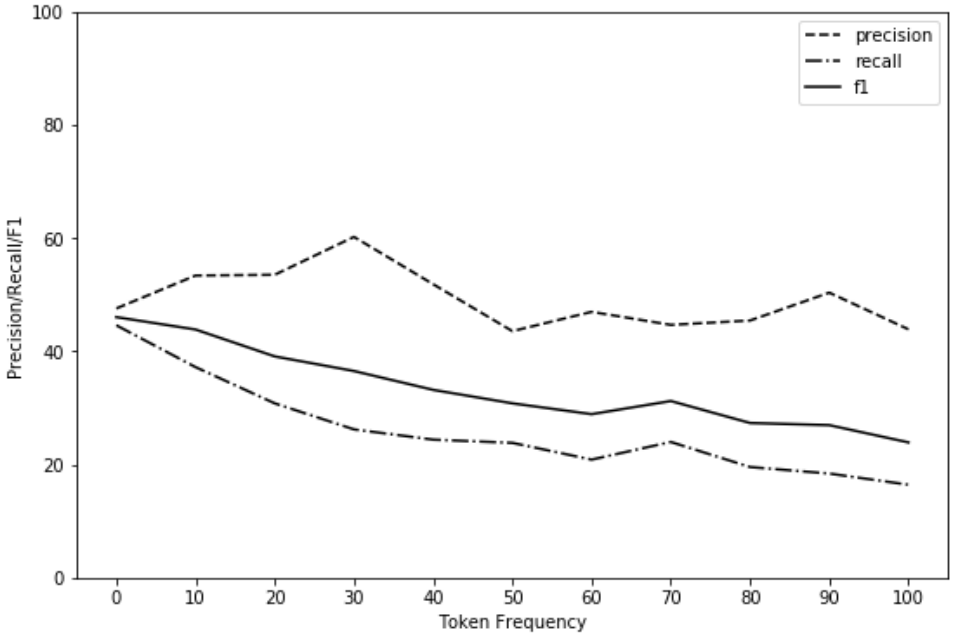


Figure 5.3: Comparison of dev set results for the *names* and *cluster* methods of bootstrapping data

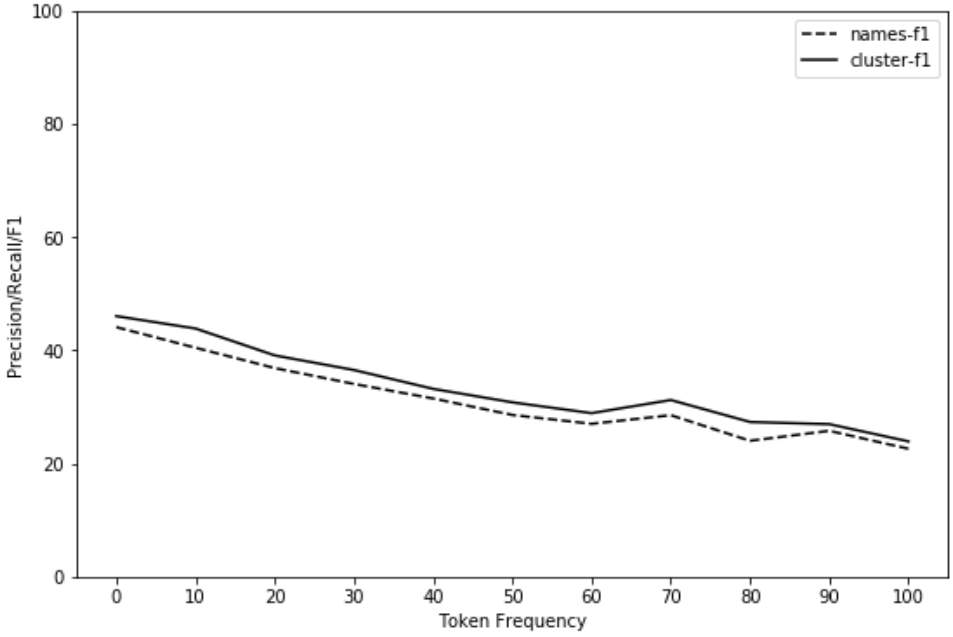


Figure 5.4: Results on the dev set of using the Stanford NER output to bootstrap training data

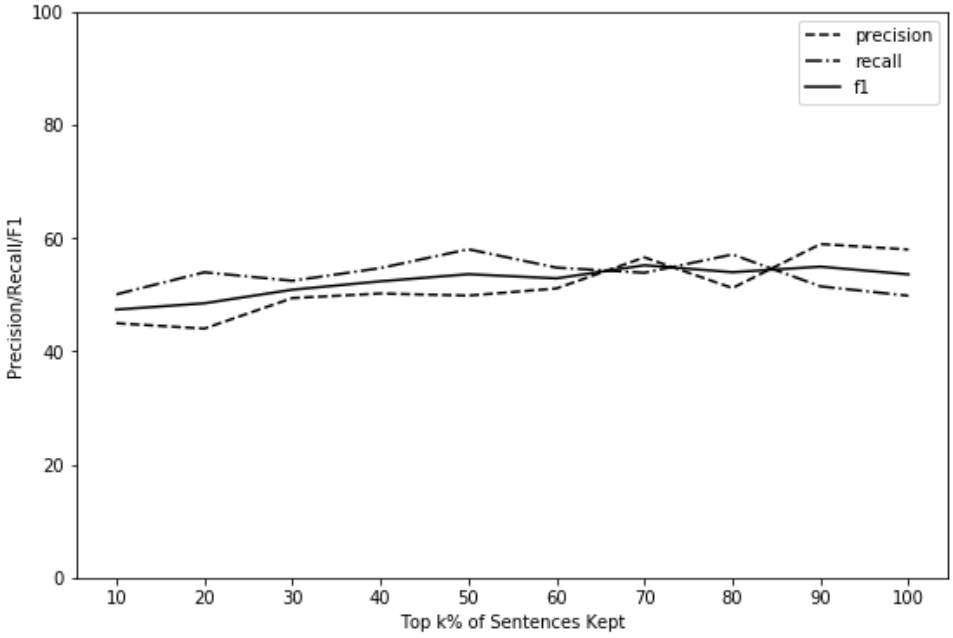


Table 5.1: Comparison of experiments with baselines on the test set. Each of the top three rows are for the three experiments using the character-based neural net model. The bottom three rows are the baselines for each experiment where the method of bootstrapping training data was used to tag the test set without the use of the neural net. *names* is the experiment where only words initially tagged as potential names were subsequently tagged as NEs. *baseline-names* is the corresponding baseline wherein the test set was tagged with the previously identified potential names. *cluster* is the experiment where all words in the PEOPLE and LOCATION clusters were tagged as NEs. *baseline-cluster* is the corresponding baseline wherein the test set was tagged with all the words in the PEOPLE and LOCATION clusters. *stanford-out* is the experiment where the Stanford NER's output was used as training data for the character embedding system. *baseline-stanford* is the baseline wherein the Stanford NER was used to tag the test set.

Experiment	Precision	Recall	F1
names	55.45%	40.58%	46.86%
cluster	53.8%	44.67%	48.81%
stanford-out	**64.19%**	**56.87%**	**60.31%**
baseline-names	18.19%	8.65%	11.73%
baseline-cluster	15.59%	9.92%	11.35%
baseline-stanford	19.27%	17.3%	18.23%

Chapter 6

DISCUSSION

As demonstrated in Chapter 5 the character-based neural net (NN) system Char-NER (Kuru et al., 2016) modestly outperformed the Stanford Named Entity Recognizer (Finkel et al., 2005) when using the Stanford NER output as training data. It cannot be said with certainty based on these experiments that this modest improvement is all that character-based neural nets can offer on the domain of data examined here. Results were likely significantly affected by a lack of accurate training data as will be discussed in the following sections. Despite the noise present in the training data in all three experiments, results only suffered in attempts to weed out the noise.

6.1 Brown Cluster Experiments

As was shown in the previous chapter, the Brown clustering experiments were outperformed by the Stanford NER experiment. Tagging the training data based on cluster data proved to be problematic, as there was a considerable amount of noise in the PEOPLE and LOCATION clusters. The PEOPLE cluster contained 5,574 unique words. Of those words, approximately 58% were actual names as determined by a manual review of all 5,574 words in the cluster. The cluster is mostly made of very infrequent words, however, and those infrequent words are more commonly not names. There are 258 words in the cluster with a minimum frequency of 10, and of those words 79% are names. The LOCATIONS cluster contained 4,960 unique words and approximately 37% of those words were actual names as determined by a manual review of all 4,960 words in the cluster. As with the PEOPLE cluster the ratio is more encouraging for words with a minimum frequency of 10; of 377 of such words, approximately 77% were actual names. This noise resulted in many words which were not names being tagged as such.

Most frequently these were common nouns or other words which happened to be capitalized. Despite the fact that words with a minimum frequency of 10 in the clusters were more likely to be true names, the effect on recall of excluding lower frequency names was greater than the gain in precision. Interestingly, the word *London* was not present in either of the clusters and as a result was rarely tagged by the CharNER system. When it was tagged, it was tagged as PERSON. These issues were present in both the names-only and entire-cluster experiments. The entire-cluster experiment did in fact result in more uncapitalized NEs being tagged, but the addition of extra noise to the training data apparently outweighed the benefits since system performance in the two experiments was approximately the same. Despite these issues, there were also some benefits as compared to the Stanford NER output experiment which will be discussed in Section 6.2.1.

6.2 Stanford NER Output Experiment

While the experiment using the Stanford NER's output produced the best results, it was not without its issues. As in the Brown cluster experiments, words which were not named entities were frequently tagged as such. In addition, noise was often introduced in the span chosen by the Stanford NER. As an example, *Ser William* would be tagged as an entity when it should only be *William* as titles are not part of the entity as per the MUC tagging guidelines (Chinchor et al., 1999). A very significant way in which this training data is lacking was not readily apparent until error analysis of the test results were done. These observations and more will be discussed in the section that follows.

6.2.1 Error Analysis

A comparison of the tagged test set when $k = 70$ with the gold standard tagged data was carried out to determine what the most common errors were. I manually examined every error the system made. In many cases for a given error there was more than one factor at play which contributed to the mistagging. In these cases I classed the error based on which factor I felt had the largest impact. One error had the biggest impact by far, accounting for

22% of the errors (error #1 in Table 6.1). In this case, words which were not NEs (and often not even proper nouns) were capitalized, which resulted in them being tagged as NEs. In general words were capitalized much more often in Early Modern English. Common nouns, emphasized words, and important terms were all frequently capitalized (Salmon, 2000). In the output of this experiment, words like *Councell* (*council*) and *Warr* (*war*) were frequently tagged though they should not be. Even other parts of speech like *Get* were frequently tagged. It is possible that the use of a part-of-speech (POS) tagger to validate the Stanford NER's tagging in the creation of training data may reduce the impact of the lack of standardization for capitalization in EME. Taggings could be checked to see if they had also been tagged as a proper noun or some other part of speech, and the tag would be kept or removed respectively. While such a POS tagger would also be affected by the idiosyncracies of EME orthography and spelling, Rayson et al. (2007) demonstrated that a hybrid rule-based and statistical POS tagger for modern English achieved over 80% accuracy on two EME corpora.

The next most common error accounted for approximately 15% of errors (error #2 in Table 6.1). In these cases there was a context surrounding the NE which would suggest there is an NE, but not a very strong one. Commonly these NEs would be modifying another noun (e.g. *Lisbon letters*) or part of a locational preposition phrase (e.g., *arrived before Leghorne*). In order for the neural net model to learn these contexts it would likely need more and more precise training data which my bootlegged data could not provide.

Also accounting for approximately 15% of errors was was essentially the opposite of the first error discussed above (error #3 in Table 6.1). This error constituted uncapitalized NEs freqently not being tagged. Proper names began to be capitalized more often in the beginning of the period of Early Modern English (Salmon, 2000), though as this was not yet the standard this is why such examples abound. As was one of my aims by devising the entire-cluster experiment, that experiment was better at tagging such NEs and in this respect had an advantage over the Stanford NER output experiment.

In approximately 14% of errors there was some strongly identifying context that was not learned by the model, most often a preceeding title (error #4 in Table 6.1). Looking at

Error Type	%
1. capitalized non-NE word	22
2. weak context	15
3. uncapitalized NE	15
4. strong context	14
5. tagged title	10
6. unclear	5
7. short sentence	3
8. nonstandard spelling	3
9. god mentions	3
10. misc. EME orthography	5
11. misc. other	3
12. my error	2
total	100

Table 6.1: Error breakdown for the test set

the training data, it became apparent that the Stanford NER was good at identifying NEs which were preceded by a title such as *Mr.* or *Dr.* but very bad at identifying NEs which were preceded by their uncapitalized and unpunctuated equivalents *mr* or *dr* which are very common in my datasets. Additionally, there were many titles present in my datasets which are likely to be much more common to the domain of my data than to the domain of data the Stanford NER was trained on. Some examples are *Ld* (an abbreviation for *Lord*) and *Esqr* (an appreviation for *Esquire*). The Modern English texts that the Stanford NER was trained on likely did not have many mentions of lords and esquires or cases where *Mr.* and other such titles were uncapitalized and not followed by a period since this is not the standard today. This explains why some titles may not have triggered the tagging of the following NE but it is less clear why cases where there was some other preceding identifying context did

not result in the tagging of an NE. Such cases are those like in *the Prince of Orange* where *Orange* is not tagged.

The next most frequent class of error accounted for 10% of the errors and was also related to titles (error #5 in Table 6.1). In this case titles were tagged as NEs though they should not be. Some examples are *Genll* (*General*), and *Capt* (*Captain*). Frequently if a title was a multiword title (such as *Lord Commissioner* or *Major Genll*), the first word would not be tagged but the second word would. Errors of this class caused an unfortunate cascade of errors when the title was followed by a name due to the fact that evaluation only considers a tagged NE to be correct if it is an exact match to the gold standard. This means that even though the actual NE following a title was frequently tagged, the entire NE nevertheless considered incorrect. Standalone titles did not cause such a cascade but still contributed to the error rate. Because the GutenTag system explicitly uses heuristics involving titles, and because I added several EME titles to its algorithm, the Brown clustering experiments do a much better job of excluding titles from taggings.

Approximately 5% of errors were cases in which it was not entirely clear to me why the NE was not tagged (error #6 in Table 6.1). Frequently these errors involved a common modern-day NE with modern day spelling such as *Paris* and *Edward*. Also included in this category were cases where a first and last name were present in the text but only one of the names was tagged. Given that non-NE consecutive capitalized words were frequently tagged as NEs, it is strange that both names were sometimes not tagged.

Issues with sentence tokenization were responsible for approximately 3% of errors (error #7 in Table 6.1). Frequently there were "sentences" with a length of just one or two tokens. In these cases there would be very little context for the model to work off of, and they were difficult even for me to identify while annotating the test data. These sentences needed more knowledge of the dataset and the topic of the data to tag, and even then I was uncertain of some of my annotations.

Another 3% of errors were attributed to nonstandard spellings of NEs (error #8 in Table 6.1). Although standardization was on the rise during the time of EME, there was also

considerable change in the language underway (Nevalainen, 2006). As such the spelling of an NE frequently differed from the modern day spelling. As an example, the town of Middleborough is spelled as *Mydelborow* in the test set. These spelling variations would pose a problem for a tagger trained on modern day spellings.

Per the guidelines in Chinchor et al. (1999), mentions of *God* (when used as a name but not as a descriptor) are to be tagged as people. This word is frequently left untagged, or peculiarly tagged as a location, and accounts for 3% of the errors (error #9 in Table 6.1).

Approximately 5% of errors were attributed to various idiosyncracies of EME orthography such as abbreviations (error #10 in Table 6.1). Frequently words are abbreviated which are not abbreviated in the present day. A particularly common abbreviation is *Wm* for the name *William*. Also included in this category are cases where a lack of, or unmodern punctuation were responsible for incorrect taggings. Lists without delimiting commas and the use of a colon instead of a period for an abbreviation were frequent culprits.

Miscellaneous circumstances such as non-English words and the tagging of organizations were responsible for approximately 3% of errors (error #11 in Table 6.1). Finally, 2% of the errors were due to my own mistagging of the test data (error #12 in Table 6.1).

Chapter 7
CONCLUSION

In this thesis I investigated the task of named entity recognition on letters and newsletters written in Early Modern English. This domain of data varies from the most frequent data used in NLP tasks in two ways. First, the data used for this project is written in Early Modern English as opposed to present-day English. While there are many similarities between this variety of English and that which we use today, there are also many differences, as anyone who has read Shakespeare can attest to. Not only does Early Modern English differ from our English today, there was a great deal of change in the language within that period itself (Nevalainen, 2006). Second, it is most often newswire that is used to train NLP systems, which differs significantly from letters. In fact, the language used in letters is more similar to the language used in speech than most other forms of written text (Biber, 1995). The hypotheses of my thesis are repeated below:

(1) Training data of sufficient quality can be bootstrapped to train a character-based neural net for named entity recognition. In this context, sufficient quality means that the data does not have so many errors as to result in a model which performs more poorly than the methods used to bootstrap the training data.

(2) A character-based neural net trained on bootstrapped data will perform better than a standard off-the-shelf named entity tagger such as the Stanford Named Entity Recognizer.

While the results obtained in these experiments leave much to be desired, all three experiments substantially improved upon their respective baselines. This demonstrates that (1) training data of sufficient quality can be bootstrapped for use in a character-based neural

net model, and (2) the use of bootstrapped training data to train a character-based neural net is preferrable over the methods used in those baselines for Early Modern English, even over the Stanford Named Entity Recognizer (NER). Moreover, the error analysis identified relevent issues to consider for future work on this domain of data as well as possible avenues for improving results. Additionally, the ability of Brown clusters to capture types of named entities is interesting and warrants future research into how they could be leveraged in bootstrapping training data. As was discussed in Chapter 6, the Brown clustering and Stanford NER methods of bootstrapping training data complemented each other in some ways, and it is possible that combining the two methods could lead to more precise training data with more training points.

BIBLIOGRAPHY

Steven Abney. Bootstrapping. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 2002.

Rami Al-Rfou, Vivek Kulkarni, Bryan Perozzi, and Steven Skiena. Polyglot-NER: Massive multilingual named entity recognition. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pages 586–594. SIAM, 2015.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

Douglas Biber. *Dimensions of register variation: A cross-linguistic comparison*. Cambridge University Press, 1995.

Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: Analyzing text with the natural language toolkit*. O'Reilly Media, Inc., 2009.

Lars Borin, Dimitrios Kokkinakis, and Leif-Jöran Olsson. Naming the past: Named entity and animacy recognition in 19th century swedish literature. In *Proceedings of the Workshop on Language Technology for Cultural Heritage Data (LaTeCH 2007)*, pages 1–8, 2007.

Julian Brooke, Adam Hammond, and Graeme Hirst. Gutentag: an NLP-driven tool for digital humanities research in the Project Gutenberg corpus. In *Proceedings of the Fourth Workshop on Computational Linguistics for Literature*, pages 42–47, 2015.

Julian Brooke, Adam Hammond, and Timothy Baldwin. Bootstrapped text-level named entity recognition for literature. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 344–350, 2016.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, December 1992. ISSN 0891-2017. URL `http://dl.acm.org/citation.cfm?id=176313.176316`.

Nancy Chinchor, Erica Brown, Lisa Ferro, and Patty Robinson. 1999 named entity recognition task definition. *MITRE and SAIC*, 1999.

James Curran and Stephen Clark. Language independent NER using a maximum entropy tagger. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, 2003.

Arjun Das, Debasis Ganguly, and Utpal Garain. Named entity recognition with word embeddings and Wikipedia categories for a low-resource language. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 16(3):18, 2017.

Hakan Demir and Arzucan Ozgur. Improving named entity recognition for morphologically rich languages using word embeddings. In *ICMLA*, pages 117–122, 2014.

Miriam Dobson and Benjamin Ziemann. *Reading Primary Sources: The Interpretation of Texts from Nineteenth and Twentieth Century History*. Routledge, 2008.

Maud Ehrmann, Giovanni Colavizza, Yannick Rochat, and Frédéric Kaplan. Diachronic evaluation of NER systems on old newspapers. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016))*, number EPFL-CONF-221391, pages 97–107. Bochumer Linguistische Arbeitsberichte, 2016.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

William A Gale, Kenneth W Church, and David Yarowsky. One sense per discourse. In *Proceedings of the Workshop on Speech and Natural Language*, pages 233–237. Association for Computational Linguistics, 1992.

Felix A Gers and Jürgen Schmidhuber. Recurrent nets that time and count. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 3, pages 189–194. IEEE, 2000.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610, 2005.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International conference on Robotics and Automation*, pages 6645–6649. IEEE, 2013.

Ralph Grishman and Beth Sundheim. Message Understanding Conference-6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, volume 1, 1996.

Claire Grover, Sharon Givon, Richard Tobin, and Julian Ball. Named entity recognition for digitised historical texts. In *LREC*, 2008.

Jiang Guo, Wanxiang Che, Haifeng Wang, and Ting Liu. Revisiting embedding features for simple semi-supervised learning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 110–120, 2014.

Phillip Hines. Newdigate newsletters, 1995. URL `http://clu.uni.no/icame/manuals/NEWDIGAT/INDEX.HTM`.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

Alison Jones and Gregory Crane. The challenge of Virginia Banks: an evaluation of named entity analysis in a 19th-century newspaper collection. In *Digital Libraries, 2006. JCDL'06. Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 31–40. IEEE, 2006.

E. Kapetanios, D. Tatar, and C. Sacarea. *Natural Language Processing: Semantic Aspects*. CRC Press, 2013. URL `https://books.google.ca/books?id=Wm3SBQAAQBAJ`.

Dan Klein, Joseph Smarr, Huy Nguyen, and Christopher D Manning. Named entity recognition with character-level models. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003-Volume 4*, pages 180–183. Association for Computational Linguistics, 2003.

Zornitsa Kozareva. Bootstrapping named entity recognition with automatically generated gazetteer lists. In *Student Research Workshop*, 2006.

Saul A Kripke. Naming and necessity. In *Semantics of Natural Language*, pages 253–355. Springer, 1972.

Vivek Kulkarni, Yashar Mehdad, and Troy Chevalier. Domain adaptation for named entity recognition in online media with word embeddings. *arXiv preprint arXiv:1612.00148*, 2016.

Onur Kuru, Ozan Arkan Can, and Deniz Yuret. CharNER: Character-level named entity recognition. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 911–921, 2016.

Percy Liang. *Semi-supervised learning for natural language*. PhD thesis, Massachusetts Institute of Technology, 2005.

Yukun Ma, Jung-jae Kim, Benjamin Bigot, and Tahir Muhammad Khan. Feature-enriched word embeddings for named entity recognition in open-domain conversations. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on Information and Automation for Sustainability*, pages 6055–6059. IEEE, 2016.

Sunghwan Mac Kim and Steve Cassidy. Finding names in trove: named entity recognition for Australian historical newspapers. In *Proceedings of the Australasian Language Technology Association Workshop 2015*, pages 57–65, 2015.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Manfred Markus. *Manual of ICAMET (Innsbruck Computer Archive of Machine-Readable English Texts)*. Leopold-Franzens-Universitat Innsbruck, 1999.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013. URL `http://arxiv.org/abs/1310.4546`.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

Terttu Nevalainen. *Introduction to Early Modern English*. Edinburgh University Press, 2006.

Terttu Nevalainen, Helena Raumolin-Brunberg, Jukka Kernen, Minna Nevala, Arja Nurmi, and Minna Palander-Collin. The Corpus of Early English Correspondence Sampler, 1998. URL `http://www.helsinki.fi/varieng/CoRD/corpora/CEEC/index.html`.

Malvina Nissim, Colin Matheson, and James Reid. Recognising geographical entities in Scottish historical documents. In *Proceedings of the Workshop on Geographic Information Retrieval at SIGIR 2004*. Citeseer, 2004.

Damien Nouvel, Jean-Yves Antoine, and Nathalie Friburger. Pattern mining for named entity recognition. In *Language and Technology Conference*, pages 226–237. Springer, 2011.

Arja Nurmi. The Corpus of Early English Correspondence Sampler (CEECS). *ICAME Journal*, 23:53–64, 1999.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. Lexicon infused phrase embeddings for named entity resolution. *arXiv preprint arXiv:1404.5367*, 2014.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Jakub Piskorski. Express–extraction pattern recognition engine and specification suite. In *Finite-state Methods and Natural Language Processing: 6th International Workshop, FSMNLP 2007. Revised Papers*, page 166. Universitätsverlag Potsdam, 2008.

Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics, 2009.

Paul Rayson, Dawn Archer, Alistair Baron, Jonathan Culpeper, and Nicholas Smith. Tagging the bard: Evaluating the accuracy of a modern POS tagger on Early Modern English corpora. 07 2007.

Samuel Richardson. *Clarissa, or, The history of a young lady.* Signet Classics, 2014.

Vivian Salmon. *ORTHOGRAPHY AND PUNCTUATION*, volume 3 of *The Cambridge History of the English Language*, page 1355. Cambridge University Press, 2000. doi: 10.1017/CHOL9780521264761.003.

Erik F Sang and Sabine Buchholz. Introduction to the CoNLL-2000 shared task: Chunking. *arXiv preprint cs/0009008*, 2000.

Erik F Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*, 2003.

Caroline Sporleder. Natural language processing for cultural heritage domains. *Language and Linguistics Compass*, 4(9):750–768, 2010.

Joseph Turian, Lev Ratinov, Yoshua Bengio, and Dan Roth. A preliminary evaluation of word representations for named-entity recognition. In *NIPS Workshop on Grammar Induction, Representation of Language and Language Learning*, pages 1–8, 2009.

Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

Miguel Won, Patricia Murrieta-Flores, and Bruno Martins. Ensemble named entity recognition (NER): Evaluating NER tools in the identification of place names in historical corpora. *Frontiers in Digital Humanities*, 5:2, 2018.

World Heritage Encyclopedia. Longest novel. *Project Gutenberg Self-Publishing Press*. URL `http://self.gutenberg.org/articles/longest_novel`.

Dan Wu, Wee Sun Lee, Nan Ye, and Hai Leong Chieu. Domain adaptive bootstrapping for named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1523–1532. Association for Computational Linguistics, 2009.

Jie Yang, Yue Zhang, Linwei Li, and Xingxuan Li. YEDDA: A lightweight collaborative text span annotation tool. *arXiv preprint arXiv:1711.03759*, 2017.