Named Entity Resolution for Historical Texts

Audrey Holmes

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2019

Reading Committee:

Sarah Ketchley

Gina-Anne Levow

Program Authorized to Offer Degree:
Department of Linguistics

University of Washington

Abstract

Named Entity Resolution for Historical Texts

Audrey Holmes

Chair of the Supervisory Committee:
Sarah Ketchley

The field of digital humanities has spurred an increase in applications of computational linguistics to historical documents, but the field remains underdeveloped. Standard natural language processing (NLP) techniques developed using contemporary texts tend to perform poorly when applied to historical documents due to challenges such as spelling variation, semantic shifts, and lack of standard orthography. In this thesis, we compare performance of common Named Entity Recognition (NER) libraries including Stanford CoreNLP, spaCy, and Flair on historical texts. We also present a method for named entity resolution designed specifically for historical texts, which combines domain adapted word embeddings with phonetic and lexical similarities. This has the potential to increase the speed of digitization of historical documents and improve search capabilities across historical corpora. The algorithm is one of the first trained on historical documents and improves upon common approaches to spelling normalization for historical documents using only lexical and/or phonetic similarity. Additionally, we provide a user interface so that scholars without programming expertise can easily use the tools developed in this thesis. Future work will include linking historical named entities to contemporary references and constructing knowledge graphs for historical corpora.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1
INTRODUCTION

Natural language processing (NLP) has had a profound effect on many industries and domains, to the point that it is virtually impossible to avoid interactions with products and systems enhanced by computational linguistics. Think of the auto-correct on your smartphone; voice assistants such as Siri, Alexa, and Cortana; and chat bots among many other applications. Despite the ubiquity and rapid development in these fields, various other areas have not received the same benefits from NLP. In particular, the sub-field of digital humanities concerning digitization and interpretation of historical texts has made limited use of NLP. This thesis will focus on improving two common NLP tasks in the context of historical texts: named entity recognition and named entity resolution.

1.1 Challenges with Historical Documents

Most state-of-the-art NLP techniques do not work as well on historical texts as they do on contemporary texts. The reason for this is that NLP tasks are generally developed on contemporary corpora. When these techniques and models are later applied to historical corpora, many new challenges arise including spelling variation, lack of standard orthography, and semantic shifts, among others.

Spelling variation presents a major challenge to NLP researchers working with historical texts. Piotrowski [19] outlines three common sources of this variation: synchronic, diachronic, and uncertainty. The first, *synchronic variation*, refers to different spellings due to a lack of standard orthography. As a result, it is not uncommon to see several variations

of the same word in a single document with no notion of a "correct" spelling. For example, one might see the forms *be* and *bee* used to denote the same meaning.

The second, *diachronic variation*, refers to time-based spelling variation. Over time, spellings can change as languages evolve and new standards are introduced. For example, some historical English texts have the letter f representing the modern s sound.

Finally, *uncertainty* refers to variation based on other factors such as OCR or transcription errors, damaged or obscured text, and illegible handwriting. Such errors are common when dealing with historical documents.

In addition to spelling variation, we run into problems when we try to apply pre-trained models to historical documents. Take the word *gay* for example. Assuming that spelling variation isn't an issue (*i.e.* it is not spelled *gaye*), we still run into trouble. In a historical context, this word is most likely a synonym for *happy*. In a contemporary context, on the other hand, it certainly won't have the final *e*, and it is much more likely to mean *homosexual* as opposed to *happy*.

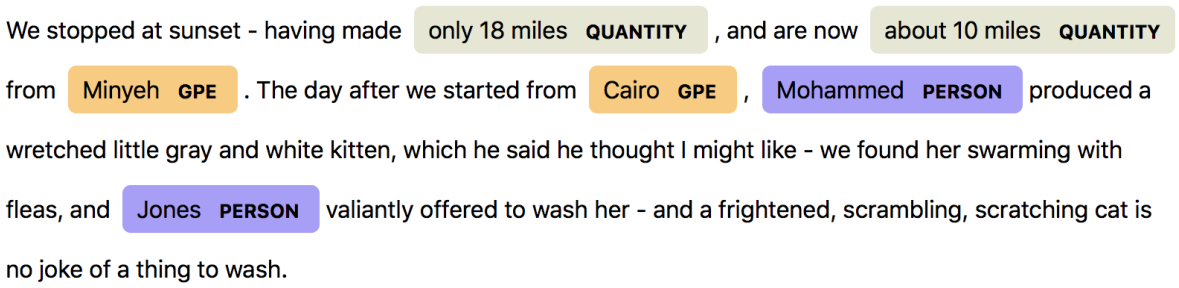
1.2 Background

We describe the tasks of named entity recognition and named entity resolution.

1.2.1 Named Entity Recognition

In NLP, *named entities* are units of information within text—often proper nouns—that refer to specific categories such as people, places, and organizations. Depending on the context, other types of entities may be included such as numbers, dates, works of art, or medical terms. The task of named entity recognition (NER) involves extracting and classifying these entities from text. It is a well-researched task with several out-of-the-box implementations including spaCy, Stanford NER, and Flair. Named entity recognition is frequently used in

Figure 1.1: Named Entity Recognition



downstream tasks such as information extraction, text classification, etc. An example of named entities extracted using spaCy is shown in Figure 1.1. The figure both highlights the surface forms of the named entities and classifies them as quantity, geopolitical entity, or person.

1.2.2 Named Entity Resolution

Given a set of named entities, the task of named entity resolution—also known as named entity disambiguation, entity linking, and entity matching—involves recognizing which entities refer to the same idea/person/place/etc. and which ones refer to distinct entities. For example, the word *Toledo* could refer to a city in Ohio or a city in Spain. Understanding this difference can be critical to understanding the meaning of a text. Similarly, *NYC*, *New York*, and *the Big Apple* all refer to the same location, namely *New York City*. Downstream tasks can be simplified by understanding that these lexically different named entities are in fact referring to the same place.

1.3 Goals and Contributions

This thesis focuses on improving two common NLP tasks in the context of historical texts: named entity recognition and named entity resolution. Chapter 2 will give an overview of previous approaches to these tasks in the historical and contemporary domains. Chapter 3

will provide details on several datasets that we use for training and evaluation: Emma B. Andrews Diaries, Van Gogh Letters, and documents from the Digital Mitford Project. In Chapter 4, we will evaluate how three popular named entity recognition software packages perform on historical corpora. In particular, we will look at Stanford NER [13], spaCy [1], and Flair [3]. Additionally, we will present one of the first approaches to named entity resolution designed specifically for historical texts and show that it outperforms common baselines that utilize lexical and phonetic similarity. We also introduce a new web application for automatically marking up historical documents without any prior programming knowledge.

Chapter 2
RELATED WORK

Past related work falls into two general categories: approaches to named entity resolution that have been developed on contemporary corpora and approaches that have been developed to deal with the specific challenges associated with historical texts. We give examples of both in this chapter. Additionally, we provide an overview of common NER software that we use for experiments in Chapter 5.

2.1 Named Entity Recognition Software

Stanford NER [13] is a Java-based NER software package that utilizes a linear chain Conditional Random Field (CRF) classifier to predict sequences of entity labels. It is one of the most common benchmarks for new NER systems.

spaCy [1] is another popular NER software package. It is distributed by Explosion AI[1]. Because spaCy is much faster than most NER implementations and packaged in Python, it has become a popular tool for industrial NLP. Unlike Stanford NER's sequence-based tagging approach, spaCy uses a transition-based approach. Thus, it starts with an empty stack, defines state-changing actions using a Convolutional Neural Network (CNN), and predicts a sequence of those actions.

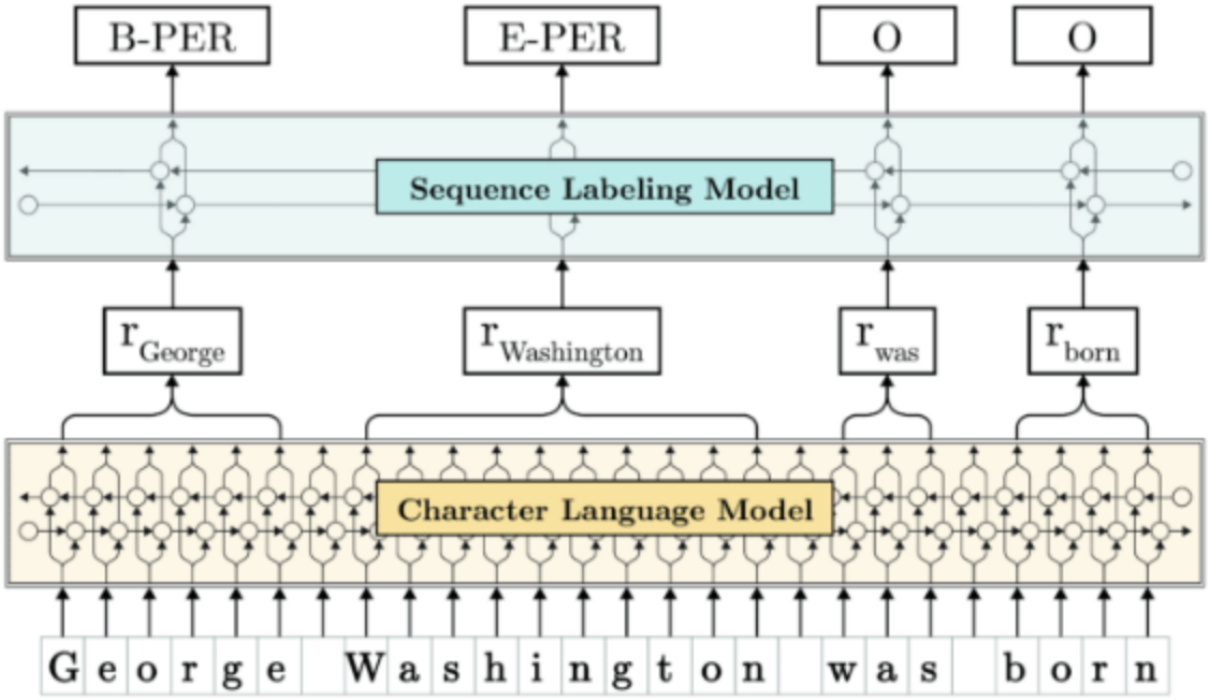
Flair [3] is the newest of the three NER software packages used in this thesis. It was created by Zalando Research[2] and released publicly in 2018. Flair is built with Python and PyTorch.

[1] `https://explosion.ai`

[2] `https://research.zalando.com`

Figure 2.1 shows the network architecture for Flair. A sequence of characters is input into a language model. Then the resulting word embeddings are used as input for a bi-directional Long Short Term Memory CRF (BiLSTM-CRF) that outputs sequences of entity labels.

Figure 2.1: Flair Architecture



Source: https://research.zalando.com/welcome/mission/research-projects/flair-nlp/

2.2 Spelling Normalization for Historical Texts

Spelling variation is a major challenge for historical texts. It is similar to the task of named entity resolution. However, in the context of historical documents, spelling variation is more frequently researched than named entity resolution.

2.2.1 Lexical and Phonetic Similarity

One family of approaches to solving spelling variation in historical texts involves matching words in a document to words in a modern dictionary based on a combination of lexical and phonetic similarity. This approach is used by the VARD 2 tool for spelling normalization of Early Modern English [6] developed by Baron and Rayson. Other projects include the papers by Peterssen et al. [18] and Exteberria et al. [12]. One reason for the popularity of this approach is that it requires little or no labeled training data.

The most common way to determine the lexical similarity of two strings is to use *edit distance*, also called *Levenshtein distance*. Edit distance is used in many modern spell checkers. It is calculated by counting the minimum number of insertions, deletions, and substitutions needed to transform one string into another. Thus, two identical strings would have an edit distance of 0. The strings *Abu Simb**e**l* and *Abu Simb**a**l* would have an edit distance of 1 since you can convert one into the other with just one substitution ($a \rightarrow e$).

In order to measure phonetic similarity, Exteberria et al. [12] use a WFST-driven (weighted finite state transducer) tool called *Phonetisaurus* to map grapheme sequences to phoneme sequences. VARD 2 uses a variant of the *SoundEx* system, originally developed for phonologically indexing census data [6].

2.2.2 Semantic Similarity

In practice, lexical and phonetic similarity measures are useful for normalizing synchronic spelling variation. However, diachronic variation can be more challenging as different forms of a word may look and sound completely different. For example, *Bombay* and *Mumbai* refer to the same place but have low phonetic and lexical similarity scores. Amoia and Martinez [4] address this problem by including semantic similarity. Using a collection of loosely aligned historical and contemporary German recipes, they automatically construct

a diachronic dictionary. They use agglomerative clustering based on lexical and semantic similarity to measure word similarity. In order to determine semantic similarity, they use mutual information of trigrams. Under this definition, words are considered similar if they occur in similar contexts.

2.3 Swoosh Algorithms for Efficient Entity Matching

While some research focuses on algorithms for finding matching entities, Benjelloun *et al.* instead focus on efficient extensible algorithms for matching and merging [7]. Assuming black-box functions for matching and merging entities, Swoosh algorithms efficiently disambiguate entities. Thus, Swoosh algorithms provide a useful framework for efficient implementation of other matching algorithms. In particular, the *R*-Swoosh algorithm can be used when the following *ICAR* properties hold (which will be the case for the entity resolution system described in Chapter 4):

1. *Idempotence* A record r always matches itself, and merging r with itself yields r.

2. *Commutativity* If a record r_1 matches r_2, then r_2 matches r_1. Furthermore, the records will merge to the same entity, regardless of order.

3. *Associativity* Let $< x, y >$ represent the merge of entities x and y. For all r_1, r_2, r_3 such that $< r_1, < r_2, r_3 >>$ and $<< r_1, r_2 >, r_3 >$ exist, $< r_1, < r_2, r_3 >>=<< r_1, r_2 >, r_3 >$.

4. *Representativity* If matching entities r_1 and r_2 merge to r_3, then for any r_4 such that r_1 matches r_4, then r_3 must also match r_4.

The idea behind the *R*-Swoosh algorithm is that once two records have been merged, the original records can be discarded, as all the information is captured in the new merged record. The full *R*-Swoosh algorithm is described in Algorithm 1.

Algorithm 1 *R*-Swoosh Algorithm

Given: a set of entities S and an empty set O

while S $\neq \emptyset$ **do**

  currentEntity $\leftarrow$ an entitiy from S

  remove *currentEntity* from S

  potentialMatch $\leftarrow$ *null*

  for all entities e in O **do**

    if M(*currentEntity*, e) =*true* **then**

      potentialMatch $\leftarrow e$

      break

    end if

  end for

  if *potentialMatch* = *null* **then**

    add *currentEntity* to S

  else

    mergedEntity $\leftarrow$<*currentEntity*, *potentialMatch*>

    remove *potentialMatch* from O

    add *mergedEntity* to S

  end if

end while

return O

2.4 Named Entity Disambiguation for Contemporary Texts

Although entity resolution for historical texts is relatively underdeveloped, significant research has been done using contemporary texts.

2.4.1 Multi-pass Sieves for Coreference Resolution

Lee *et al.* describe a multi-pass sieve architecture for coreference resolution in [16]. This approach combines machine learning models in a deterministic manner. The sieve approach applies independent coreference resolution models in order of decreasing precision. Thus, it is highly modular and new coreference resolution models can be added easily. The sieve is successful because high precision models merge obvious matches. Thus, subsequent models with lower precision but higher recall can leverage the context from all of the merged entities as opposed to just individual entities.

2.4.2 Entity Linking

Entity linking (EL) or *Named Entity Disambiguation* (NED) involves matching the surface form of a named entity in a document to a unique entry in an existing knowledge base such as Wikipedia or Freebase. In [9], Cucerzan presents a method for large-scale entity matching in which he matches surface forms to Wikipedia entries based on the similarity of contexts in the documents and the Wikipedia pages as well as agreement in category tags. In [11], Eshel *et al.* presents an entity disambiguation approach for short noisy texts. This work employs an Attention-RNN to produce a probability-like score for each surface form and candidate entity match.

In [15], Kolitsas *et al.* present an end-to-end entity linking approach. This approach combines entity linking and mention detection, thereby eliminating the need for a separate NER task. This is achieved by training context-aware word embeddings by concatenating the hidden states of a bi-LSTM on top of character embeddings. Entities in the knowledge base

are represented using pre-trained embeddings. For each span of up to a fixed length in a document, the dot product is used as a compatibility score with entities in the knowledge base.

All of the afore-mentioned entity linking approaches require large amounts of labeled training data. Moreover, they assume that Wikipedia contains entries for at least most of their named entities, which is generally not the case when dealing with historical documents.

2.5 Cold Start Knowledge Base Population

Given the existence of many large knowledge bases, entity linking can be extremely useful. However, it can also be the case that an entity in a document does not have a corresponding entry in the knowledge base. In this case, a knowledge graph can be constructed from unstructured data. Román *et al.* describe an unsupervised NED approach using semantic networks in [21]. Here, named entities are represented by nodes in a graph, while information extracted from the documents is used to represent edges connecting nodes in the graph. Then entity disambiguation can be reframed as determining the connectedness of any given nodes in the graph. This eliminates the problem of entries missing from an existing knowledge base, but it has the disadvantage that the generated knowledge graph is unique to a given dataset and not as widely accepted as, for example, Wikipedia.

Chapter 3

HISTORICAL CORPORA

3.1 Text Encoding Initiative

The Text Encoding Initiative (TEI) is a non-profit consortium that publishes a set of guidelines for encoding digital humanities data. TEI is used by many universities, libraries, museums, and other research institutions because it provides an XML schema that supports many of the challenges and nuances of digitizing historical texts [2]. Importantly for the purposes of this thesis, the TEI schema includes named entity tags and reference attributes for disambiguation.

3.2 Datasets

While carrying out the research in this thesis, we worked with several datasets detailed below. These datasets are used for training and evaluation. A summary of the datasets can be found in Tables 3.1, 3.2, and 3.3.

3.2.1 Emma B. Andrews Diaries

There are 19 unpublished diary volumes spanning the years 1889-1912. As part of the Emma B. Andrews Diary Project [10], the diaries have been annotated in TEI format with PERSON, LOCATION, and ORGANIZATION named entity tags. Additionally, the canonical forms for each named entity tag are included. Emma's diaries provide details of her travels along the Nile River in Egypt and chronicle Theodore M. Davis's excavations of 20 tombs in the Valley of the Kings.

3.2.2 Van Gogh Letters

There are 902 letters spanning the years 1872-1890 [14]. They have been annotated in TEI format in their original languages as well as in English translation. In this thesis, we work with the English translations. The letters are annotated with PERSON and WORK OF ART named entity tags as well as numeric indices for entity disambiguation. Most of the letters are correspondence between Vincent van Gogh and his brother Theo van Gogh. However, the collection also includes letters to other notable artists of the time.

3.2.3 Digital Mitford

There are 97 of Mary Russell Mitford's letters and personal papers spanning the years 1819-1925. These have been fully annotated in TEI format as part of the Digital Mitford Project [5]. Annotations include PERSON and LOCATION named entities tags and their canonical forms. Additionally, we use five annotated full-length plays written by Mary Russell Mitford. The plays are also annotated with PERSON and LOCATION named entity tags and canonical forms. Entity references are consistent across the plays and the letters.

Table 3.1: Corpus Size

Dataset	**Document Count**	**Token Count**
Andrews Diaries	19	177,155
Van Gogh Letters	902	915,880
Digital Mitford	97	158,992

Table 3.2: Corpus Tags

Dataset	**Entity Tags**	**Entity Tag Count**
Andrews Diaries	PERSON, LOCATION, ORGANIZATION	3,293
Van Gogh Letters	PERSON, WORK OF ART	12,305
Digital Mitford	PERSON, LOCATION, ORGANIZATION	4,878

Table 3.3: Corpus Tag Counts

Tag	**Andrews**	**Van Gogh**	**Mitford**	**Count**
PERSON	2,102	11,429	3,736	17,267
LOCATION	1,101	0	846	1,947
ORGANIZATION	90	0	296	386
WORK OF ART	0	876	0	876

Chapter 4

SYSTEM DESIGN AND IMPLEMENTATION

This chapter introduces word embeddings and the NYSIIS algorithm, which are then applied to a novel machine learning algorithm for entity resolution.

4.1 Phonetic Representation

Since spelling variation can be prevalent in historical documents, we can represent words phonetically. For phonetic representation, we use the NYSIIS algorithm.

4.1.1 NYSIIS

NYSIIS [8] is an algorithm developed as part of the *New York State Identification and Intelligence System* that improves upon SoundEx. Like SoundEx, NYSIIS uses a rule-based system to transform names into phonetic codes and works best with American names. However, several variations exist that could be substituted with, for example, Eastern European names. We will see in Chapter 5 that NYSIIS does not perform well on the Dutch and French names found in the Van Gogh letters and that a language-specific phonetic representation would be more suitable. Unlike SoundEx, NYSIIS maps sequences of characters to phonetic codes as opposed to individual letters. Moreover, its transformations respect the relative positions of vowels. The NYSIIS algorithm is described in Algorithm 2. Using NYSIIS, we can determine whether or not any two entities are phonetically equivalent, even if they are spelled differently. For example, the names Mahomed, Mohammed, and Mohamed, would all be represented by NYSIIS as *MANAD*.

Algorithm 2 NYSIIS Algorithm Part 1

Given: Name N

if N begins with substring s where $s ==$ MAC **then**

$s \leftarrow$ MCC

else if N begins with substring s where $s ==$ KN **then**

$s \leftarrow$ N

else if N begins with substring s where $s ==$ K **then**

$s \leftarrow$ C

else if N begins with substring s where s in [PH, PF] **then**

$s \leftarrow$ FF

else if N begins with substring s where $s ==$ SCH **then**

$s \leftarrow$ SSS

end if

if N ends with substring t where t in [EE or IE] **then**

$t \leftarrow$ Y

else if N ends with substring t where t in [DT, RT, RD, NT, ND] **then**

$t \leftarrow$ D

end if

Algorithm 3 NYSIIS Algorithm Part 2

if $N_{1:}$ contains substring s where $s ==$ EV **then**
  $s \leftarrow$ AF
end if
if $N_{1:}$ contains substring s where s in [A, E, I, O, U] **then**
  $s \leftarrow$ A
end if
if $N_{1:}$ contains substring s where $s ==$ Q **then**
  $s \leftarrow$ G
end if
if $N_{1:}$ contains substring s where $s ==$ Z **then**
  $s \leftarrow$ S
end if
if $N_{1:}$ contains substring s where $s ==$ M **then**
  $s \leftarrow$ N
end if
if $N_{1:}$ contains substring s where $s ==$ KN **then**
  $s \leftarrow$ N
end if
if $N_{1:}$ contains substring s where $s ==$ K **then**
  $s \leftarrow$ C
end if
if $N_{1:}$ contains substring s where $s ==$ SCH **then**
  $s \leftarrow$ SSS
  if $N_{1:}$ contains substring s where $s ==$ PH **then**
    $s \leftarrow$ FF
  end if
end if

Algorithm 4 NYSIIS Algorithm Part 3

if $N_{-1} ==$ S **then**
  $N \leftarrow N_{0:-1}$
end if
if N ends with substring t where $t ==$ AY **then**
  $t \leftarrow$ Y
end if
if $N_{-1} ==$ A **then**
  $N \leftarrow N_{0:-1}$
end if

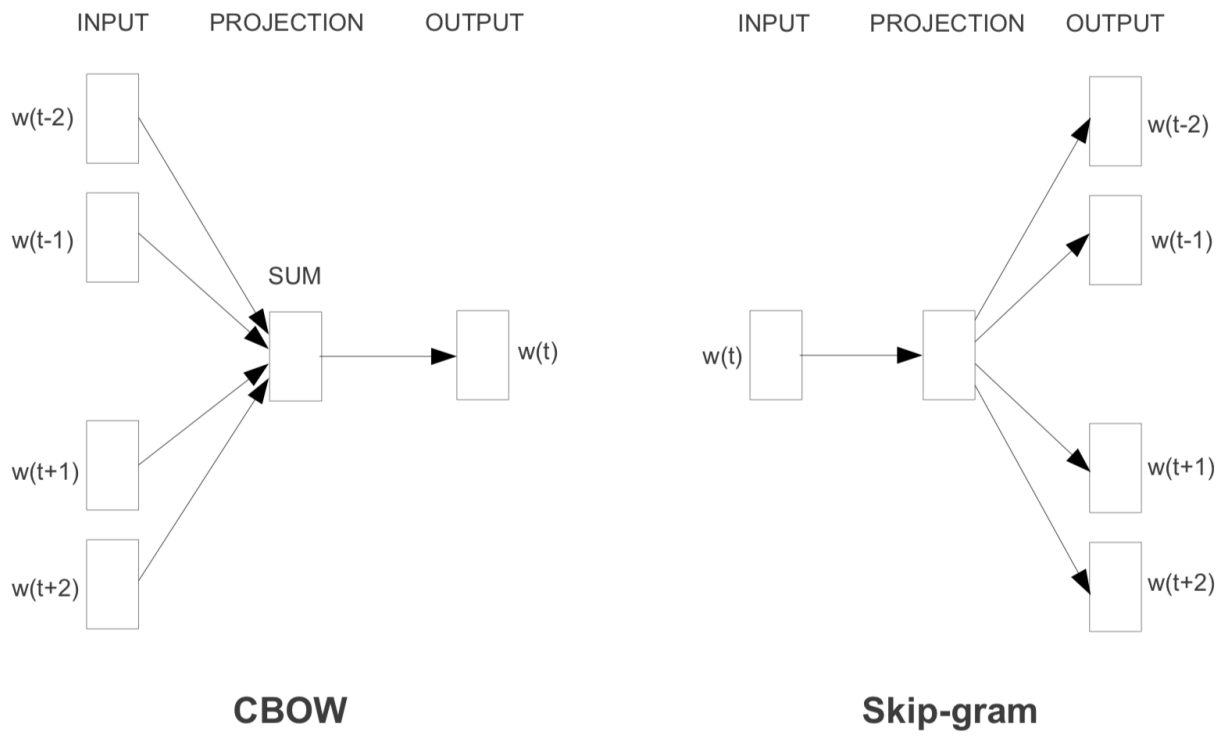
4.2 Word Embeddings

Word embeddings are vector representations of words, where each vector is of fixed length with numeric entries. Word embeddings preserve basic geometry, such as adding, subtracting, distance, etc. One of the most popular algorithm for creating word embeddings is *word2vec* [17].

4.2.1 word2vec

There are two flavors of *word2vec*: skip-gram and continuous bag of words (CBOW), which are both shown in Figure 4.1. Both are shallow neural networks with one hidden layer that take one-hot encoded word vectors as input. The CBOW model is trained by taking a target word's *context*, (i.e. the word(s) immediately before and after the word) and predicting the target word. The skip-gram model does the opposite, predicting context given a base word. We will use the CBOW implementation of *word2vec*. Although *word2vec* is trained on a prediction task, the word embeddings are comprised of the fitted weights in the hidden layer rather than the predicted word(s).

Figure 4.1: word2vec Architecture



Source: https://arxiv.org/pdf/1301.3781.pdf

4.2.2 Domain-Weighted Word Embeddings

There are many pre-trained word embeddings readily available. In this thesis, we use spaCy's pre-trained word embeddings [1]. These consist of 20,000 300-dimensional vectors. However, we will show in Chapter 5 that generic word embeddings perform poorly on historical texts. Therefore, we train custom *word2vec* word embeddings on domain specific text using the *gensim* framework [20].We use *gensim* to train a continuous bag-of-words (CBOW) model on historical texts with a window of size 3 that outputs 300-dimensional word vectors.

Domain specific word embeddings are problematic, however, because the available text just isn't as readily available in the same quantities as contemporary texts. We tackle this

shortcoming by combining the breadth of generic word embeddings with the specificity of domain-weighted word embeddings.

Equation 4.1 shows our formula for domain-weighted word embeddings, where v_g is a word's generic embedding and v_s is a word's domain specific embedding, and α is a weighting parameter. In Chapter 5, we experiment with setting different values for α. In the case where one embedding is missing, we substitute a random vector.

$$v_a = v_g + \alpha \cdot v_s \tag{4.1}$$

4.3 Ensemble Model

So far, we have three approaches to entity resolution: lexical similarity, phonetic similarity, and semantic similarity (domain-weighted word embeddings). Our goal is to combine them in such a way that increases overall performance. To do so, we implement a voting process, where for each pair of named entities, each of the algorithms casts a vote for a match or not. Each vote is weighted by the probability score assigned by the algorithm and the algorithm's relative performance compared to the other two algorithms. The ensemble model is described in 4.2, where n is the number of individual classifiers, α_i is the F_1 score for the i-th classifier, and x_i is the probability score assigned by the i-th classifier.

$$Y = \frac{\sum_{i=1}^{n} \alpha_i \cdot x_i}{\sum_{i=1}^{n} \alpha_i} \tag{4.2}$$

4.4 Web Interface

A web interface[1] was created for historians without programming experience to run the algorithms presented in this thesis. Figure 4.2 shows a screenshot of the user interface's landing page. A user can upload their raw text and get output in TEI format with named entity labels and references. Additionally, the tool will add appropriate TEI headers and metadata. Figure 4.3 shows example output that has been marked up in TEI format.

[1] `http://www.historical-markup.com`

Figure 4.2: Historical Markup User Interface

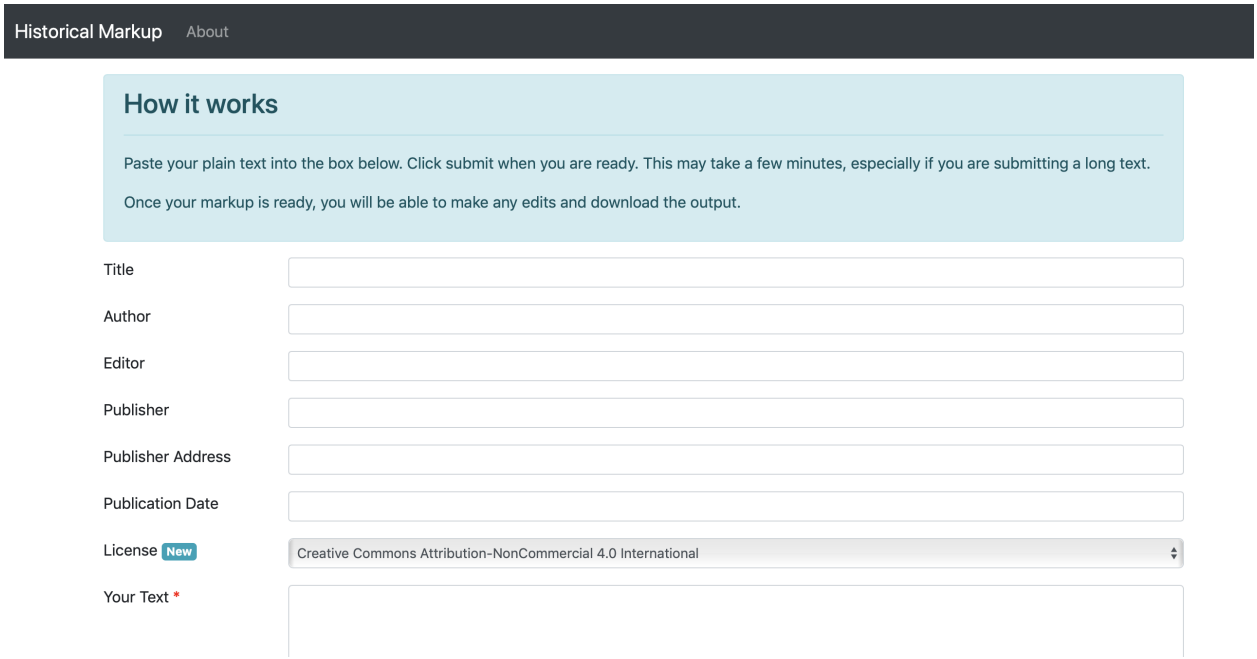


Figure 4.3: Historical Markup Output

Chapter 5
EXPERIMENTS AND RESULTS

This chapter presents experiments and results relating to named entity recognition and the entity resolution methods described in Chapter 4.

5.1 Named Entity Recognition

Named entity recognition is an important prerequisite for entity matching. In realistic scenarios, error will be introduced from imperfect NER labeling into the downstream task of entity matching. To better understand this error, we compare several popular off-the-shelf NER software packages: Stanford NLP [13], spaCy [1], and Flair [3].

5.1.1 NER Evaluation

The entity types PERSON, PLACE, and ORGANIZATION are considered for each software package. Any entity types that are not labeled in a given dataset are ignored (e.g. the Van Gogh letters do not have organizations or places labeled, so only people are considered). A predicted entity label is considered correct if it matches the entity type of the gold standard label and if more than half of the characters in the predicted and gold standard entities overlap. The character overlap criterion is used instead of an exact match to account for different labeling conventions such as whether or not to include personal titles in PERSON entities. Examples of this are shown in Table 5.1. Precision, recall, and F_1 scores are computed for each software-dataset pair. The results are shown in Table 5.2.

The Flair library gives the best results across all metrics on the Mitford letters and the Emma Andrews diaries. The results for the Van Gogh letters are split, favoring spaCy and

Table 5.1: NER Evaluation Examples

Predicted Tag	**Actual Tag**	**Predicted Text**	**Actual Text**	**Is Match?**
PLACE	PLACE	Paris	Paris	Yes
PLACE	PERSON	Paris	Paris	No
PERSON	PERSON	Mr. Jones	Jones	Yes
PLACE	PLACE	Valley of Kings	the Valley of Kings	Yes
PLACE	PLACE	Basilica	St. Peter's Basilica	No
PLACE	PLACE	St. Peter's	St. Peter's Basilica	Yes

Stanford NER. Possible reasons for this difference include the fact that the Van Gogh letters are translated into English. Additionally, only PERSON named entity tags are evaluated due to the TEI labeling in the dataset.

Table 5.2: Evaluation of NER Software on Historical Corpora

	Stanford NLP			**spaCy**			**Flair**		
	P	R	F_1	P	R	F_1	P	R	F_1
Digital Mitford	0.526	0.487	0.506	0.474	0.430	0.451	**0.535**	**0.507**	**0.521**
Andrews Diaries	0.535	0.486	0.509	0.239	0.232	0.235	**0.626**	**0.652**	**0.639**
Van Gogh Letters	0.312	**0.392**	0.347	**0.386**	0.323	**0.352**	0.278	0.375	0.319

5.2 Entity Matching

5.2.1 Entity Matching Evaluation

In order to evaluate and compare entity matching algorithms, we compute pairwise precision, recall, and F_1 scores. The algorithms are evaluated on gold standard entity labels. Additionally, in order to evaluate a more realistic context, we compute the same metrics on predicted

entity labels from Flair, the highest performing NER software package, as shown in Table 5.2. When evaluating performance using the Flair named entities, we do not consider the named entities that were not picked up by Flair. However, any incorrectly identified named entities are marked as incorrect in the matching context.

In all experiments, we use a 70-30 train-test split. Additionally, 20% of the training data is held out for tuning parameters. Unless otherwise specified, metrics are given for test data.

5.2.2 Baseline Approaches

The most common methods for entity linking in historical corpora involve clustering on edit distance. Accordingly, we use the Levenshtein similarity metric given in Equation 5.1. The results are shown in Table 5.3. This is matching entities across each corpus, not just within individual documents.

$$S_{Levenshtein}(x, y) = 1 - \frac{D_{Levenshtein}(x, y)}{max(len(x), len(y))} \tag{5.1}$$

In addition to a pure edit distance approach, we use the NYSIIS algorithm [8], an extension of *SoundEx*, to create a phonetic representation of each entity. In order to create a phonetic similarity score, we use the Levenshtein similarity of the phonetic representations. The results of combining edit distance and NYSIIS are show in Table 5.4. Adding phonetic information adds a small performance boost that is not statistically significant.

5.2.3 Generic Word Embeddings

Table 5.5 shows the entity matching results using only generic pre-trained word2vec word embeddings. On their own, the generic word embeddings perform worse than the baseline approaches. This is due in large part to a high percentage of out-of-vocabulary words present in historical texts. Moreover, many words that appear in historical texts have different

Table 5.3: Baseline: Entity Matching

	Labeled Entities			**Predicted Entities**		
	Precision	Recall	F_1	Precision	Recall	F_1
Digital Mitford	0.612	0.795	0.692	0.670	0.387	0.490
Andrews Diaries	0.751	0.756	0.753	0.718	0.730	0.724
Van Gogh Letters	0.710	0.834	0.767	0.085	0.043	0.082

Table 5.4: Entity Matching with Phonetic Score

	Labeled Entities			**Predicted Entities**		
	Precision	Recall	F_1	Precision	Recall	F_1
Digital Mitford	0.614	0.795	0.693	0.676	0.389	0.493
Andrews Diaries	0.754	0.754	0.754	0.719	0.726	0.723
Van Gogh Letters	0.711	0.830	0.766	0.086	0.043	0.082

meanings than contemporary texts that word embeddings are trained on.

5.2.4 Domain Specific Word Embeddings

Table 5.6 shows the entity matching results using our domain custom-trained domain specific word embeddings. These embeddings perform slightly better than generic word embeddings. However, they still have a significant problem with out-of-vocabulary words due to the small size of the training corpus.

5.2.5 Domain-Weighted Word Embeddings

The domain-weighted word embeddings in Equation 4.1 require setting a parameter α that controls the strength of the domain weighting. To set an appropriate value for α, we compare entity matching scores on labeled data. This analysis is done on 20% of the training data set

Table 5.5: Entity Matching with Generic Word Embeddings

	Labeled Entities			**Predicted Entities**		
	Precision	Recall	F_1	Precision	Recall	F_1
Digital Mitford	0.429	0.591	0.496	0.519	0.294	0.395
Andrews Diaries	0.537	0.552	0.543	0.629	0.630	0.630
Van Gogh Letters	0.621	0.688	0.674	0.035	0.032	0.033

Table 5.6: Entity Matching with Domain Specific Word Embeddings

	Labeled Entities			**Predicted Entities**		
	Precision	Recall	F_1	Precision	Recall	F_1
Digital Mitford	0.438	0.595	0.502	0.522	0.306	0.465
Andrews Diaries	0.539	0.568	0.553	0.635	0.649	0.640
Van Gogh Letters	0.650	0.698	0.688	0.037	0.035	0.034

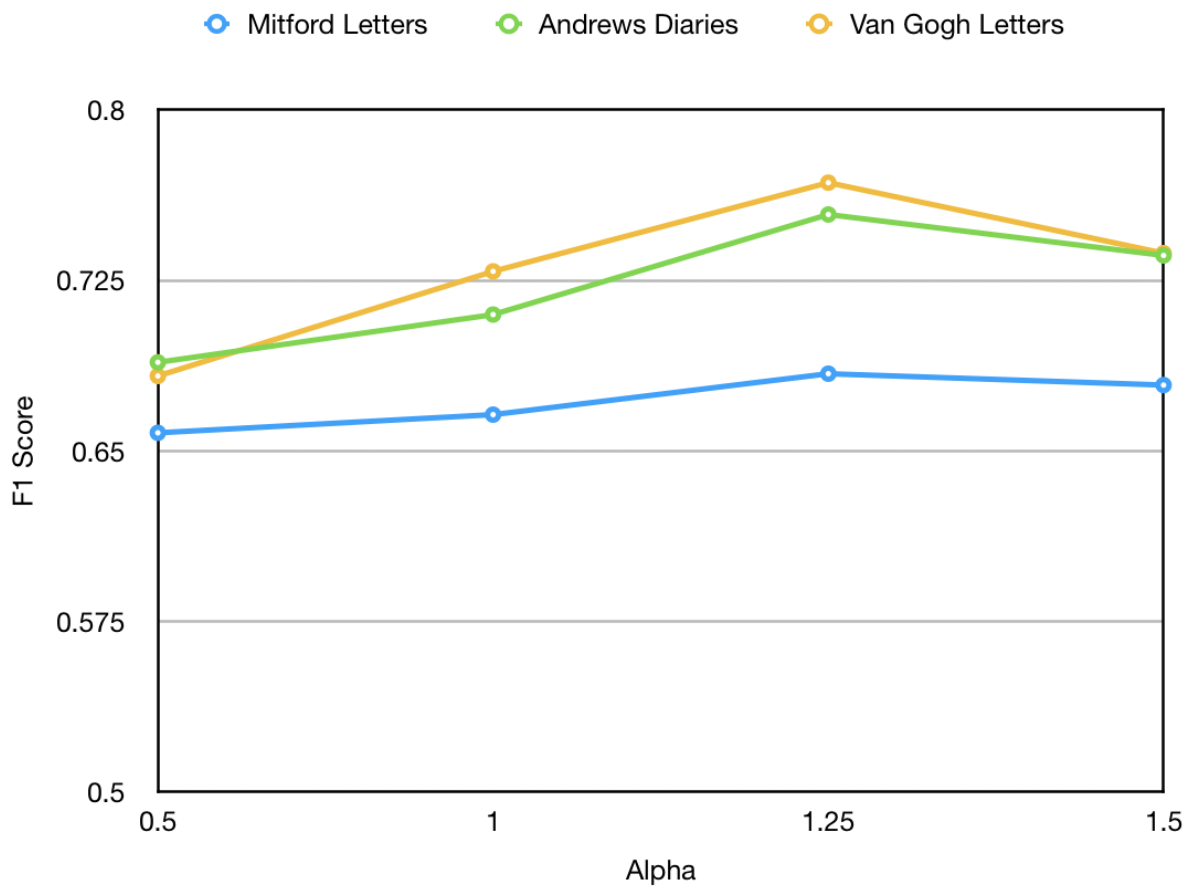
held out for this purpose. The results are shown in Figure 5.1, with α set to 1.25 achieving the best results.

Table 5.7 shows the entity matching results using our domain-weighted word embeddings, where α is set to 1.25. These word embeddings perform significantly better than generic and domain specific word embeddings on their own and slightly better than the baseline approaches. Using the domain adapted word embeddings results in fewer out-of-vocabulary words but does not completely solve the issue.

5.2.6 Ensemble Model

The results in Table 5.8 show entity matching performance metrics using the ensemble model described in Chapter 4. This model outperforms the baseline approaches as well as the

Figure 5.1: Parameter Expierment: Alpha



domain-weighted word embeddings. This is true for both gold standard entity labels as well as predicted entity labels.

5.2.7 Summary

We summarize the F_1 scores from different experiments in Table 5.9. Across all three document collections, the ensemble model with domain weighted word embeddings outperforms all baselines when applied to named entities predicted using Flair. The ensemble model also outperforms baselines on human-labeled named entities.

Table 5.7: Entity Matching with Domain-Weighted Word Embeddings

	Labeled Entities			**Predicted Entities**		
	Precision	Recall	F_1	Precision	Recall	F_1
Digital Mitford	0.616	0.800	0.684	0.723	0.419	0.524
Andrews Diaries	0.755	0.754	0.754	0.754	0.720	0.731
Van Gogh Letters	0.714	0.833	0.768	0.091	0.043	0.085

Table 5.8: Entity Matching with Ensemble Model

	Labeled Entities			**Predicted Entities**		
	Precision	Recall	F_1	Precision	Recall	F_1
Digital Mitford	0.727	0.816	0.769	0.654	0.614	0.632
Andrews Diaries	0.813	0.834	0.816	0.789	0.701	0.743
Van Gogh Letters	0.788	0.790	0.789	0.102	0.086	0.091

5.3 Error Analysis

We analyze why the ensemble model outperforms the baselines and provide examples of where it still makes mistakes.

5.3.1 Named Entity Recognition

We see that although the predicted entity scores look pretty good for the Andrews Diaries and Digital Mitford documents, the Van Gogh scores are quite low. This is primarily due to the fact that the Flair library used to predict the named entity labels does not perform as well on the Van Gogh letters as it does on the other document collections, which is shown in Table 5.2.

Another problem we see is that most NER packages do poorly recognizing compound named

Table 5.9: Entity Matching F_1 Summary

	Lexical		**Lexical + Phonetic**		**Ensemble**	
	Labeled	Predicted	Labeled	Predicted	Labeled	Predicted
Digital Mitford	0.692	0.490	0.693	0.493	**0.769**	**0.632**
Andrews Diaries	0.753	0.724	0.754	0.723	**0.816**	**0.743**
Van Gogh Letters	0.767	0.082	0.766	0.082	**0.789**	**0.091**

entities. For example, none of the NER packages used in this thesis recognize that the phrases *Mr. and Mrs. Christian* and *the Scott-Elliots* both represent two distinct people, although most systems recognize *Mrs. Christian* and *Mr. Scott-Elliot* as PERSON named entities. These NER errors make it more difficult for entity resolution systems to correctly disambiguate similar named entities.

5.3.2 Person Name Variation

Moreover, we see that the Andrews Diaries have the best scores for both labeled and predicted entities. Some of the boost in performance for the Andrews Diaries comes from a high percentage of LOCATION labels. As shown in Tables 3.2 and 3.3, the Andrews Diaries entity labels are 33.4% LOCATION tags, compared to 17.3% for Digital Mitford and 0% for the Van Gogh letters. Person names tend to have more variation than place names. For example, in the Andrews Diaries, *Seville* is only ever referred to as *Seville*, whereas *Theodore Davis* is also referred to as *Theo* and *Theodore*, among others. Lexical similarity performs well on the Andrews Diaries because that document collection has more exact entity matches than the other collections.

Another challenge for entity disambiguation that arises with historical documents is the tendency of referring to women by their husband's name. For example, the baseline approaches relying on lexical and phonetic similarity resolve *Henry Christian* and *Mrs. Henry*

Christian as the same person. When we introduce semantic similarity, we improve the ability to distinguish between males and females. However, even this is not perfect, as husbands and wives tend to appear in similar contexts.

Chapter 6
DISCUSSION AND FUTURE WORK

6.1 Named Entity Recognition

One disadvantage of the method presented in this thesis is that it does not explicitly address named entity recognition for historical documents. Although a case was presented that the Flair library for NER performs on historical texts better than other top libraries, it does not alter the fact that Flair was created for use on contemporary documents. We expect that a significant performance boost could be achieved by training a domain specific NER model. This is particularly true in the case of the Van Gogh letters, where Flair does not perform well, and thus the output of the entity matching method also performs poorly.

Additionally, this thesis does not address domain specific categories of named entities. For example, it would be beneficial to have categories such as vessels, hotels, and works of art, as these often occur in historical texts. These additional categories would underscore the need for robust named entity resolution. Ships, for example, are often named after people. Thus, *Queen Elizabeth* the ship would need to be distinguished from *Queen Elizabeth* the monarch.

6.2 Future Work

Although this thesis contributes to the application of NLP methods to historical texts, the historical domain remains largely unexplored by computational linguists.

6.2.1 Entity Linking

A natural extension of the work presented in this thesis would be to match entity references across documents and time periods. For example, it would be useful to map historical place

names to their contemporary names for tracking historical journeys using modern maps. Additionally, if one were able to link named entities to an outside reference, such as Wikipedia, Freebase or a relevant historical resource, scholars could find additional reference materials that mention the same people and places.

Another natural extension would be to model relationships between named entities in a knowledge graph. For example, one could construct a knowledge graph to link Vincent Van Gogh as the creator of *Starry Night* and Theo Van Gogh as the brother of Vincent, where Vincent, Theo, and *Starry Night* are nodes in the graph while brother and creator are edges connecting the nodes.

6.3 Conclusion

We have presented a comparison of popular off-the-shelf NER software packages and their performance on historical texts. Furthermore, we have presented an algorithm for named entity resolution and achieved better performance than on several baseline approaches. This algorithm has been made available via a web interface to scholars without programming experience and will help to advance research in the digital humanities.

BIBLIOGRAPHY

[1] spaCy v2.0. *https://spacy.io*.

[2] TEI: Text encoding initiative. *http://www.tei-c.org*.

[3] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

[4] Marilisa Amoia and Jose Manuel Martinez. Using comparable collections of historical texts for building a diachronic dictionary for spelling normalization. *Proceedings of the 7th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH)*, pages 84–89, August 2013.

[5] Digital Mitford: The Mary Russell Mitford Archive. *http://digitalmitford.org*, December 2018.

[6] Alistair Baron and Paul Rayson. VARD 2: A tool for dealing with spelling variation in historical corpora. *Proceedings of the Postgraduate Conference in Corpus Linguistics*, 2008.

[7] Omar Benjelloun, Hector Garcia-Molina, David Menestrina, Qi Su, Steven Euijong Whang, and Jennifer Widon. Swoosh: a generic approach to entity resolution. *The VLDB Journal*, page 255, 2009.

[8] Paul E. Black. NYSIIS. *Dictionary of Algorithms and Data Structures*, 2019.

[9] Silviu Cucerzan. Large-scale named entity disambiguation based on wikipedia data. *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2007.

[10] The Emma B. Andrews Diary Project. *http://www.emmabandrews.org/project*, December 2018.

[11] Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuda Yamada, and Omer Levy. Named entity disambiguation for noisy text. *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL)*, pages 58–68, 2017.

[12] Izaksun Exteberria, Iñaki Alegria, and Larraitz Uria. Combining phonology and morphology for the normalization of historical texts. *Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences and Humanities (LaTeCH)*, pages 100–105, August 2016.

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

[14] Vincent Van Gogh. *Van Gogh Letters*. Van Gogh Museum *http://vangoghletters.org/vg/*.

[15] Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. End-to-end neural entity linking. *Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL 2018)*, pages 519–529, 2018.

[16] Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4).

[17] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *NIPS*, 2013.

[18] Eva Pettersson, Beáta Megyesi, and Joakim Nivr. Normalisation of historical text using context-sensitive weighted Levenshtein distance and compound splitting. *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA)*, 2013.

[19] Michael Piotrowski. *Natural Language Processing for Historical Texts*. Number 17 in Synthesis Lectures on Human Language Technologies. Morgan & Claypool, 2012.

[20] Radim Řehůřek and Petr Sojka. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.

[21] Jorge H. Román, Kevin J. Hulin, Linn M. Collins, and James E. Powell. Entity disambiguation using semantic networks. *Journal of the American Society for Information Science and Technology*, 63(10):2087–2099, 2012.