DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS

STOCKHOLM, SWEDEN 2021

Stylometric Embeddings for Book Similarities

BEICHEN CHEN

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Stylometric Embeddings for Book Similarities

BEICHEN CHEN

Master's Programme, Computer Science, 120 credits
Date: June 20, 2021

Supervisor: Jussi Karlgren
Examiner: Viggo Kann
School of Electrical Engineering and Computer Science
Host company: Storytel AB
Swedish title: Stilometriska vektorer för likhet mellan böcker

Abstract

Stylometry is the field of research aimed at defining features for quantifying writing style, and the most studied question in stylometry has been authorship attribution, where given a set of texts with known authorship, we are asked to determine the author of a new unseen document. In this study a number of lexical and syntactic stylometric feature sets were extracted for two datasets, a smaller one containing 27 books from 25 authors, and a larger one containing 11,063 books from 316 authors. Neural networks were used to transform the features into embeddings after which the nearest neighbor method was used to attribute texts to their closest neighbor. The smaller dataset achieved an accuracy of 91.25% using frequencies of 50 most common functional words, dependency relations, and Part-of-speech (POS) tags as features, and the larger dataset achieved 69.18% accuracy using a similar feature set with 100 most common functional words. In addition to performing author attribution, a user test showed the potentials of the model in generating author similarities and hence being useful in an applied setting for recommending books to readers based on author style.

Keywords

Stylometry, Authorship attribution, Embeddings, Neural networks, Natural language processing, Book recommendations

Sammanfattning

Stilometri eller stilistisk statistik är ett forskningsområde som arbetar med att definiera särdrag för att kvantitativt studera stilistisk variation hos författare. Stilometri har mest fokuserat på författarbestämning, där uppgiften är att avgöra vem som skrivit en viss text där författaren är okänd, givet tidigare texter med kända författare. I denna stude valdes ett antal lexikala och syntaktiska stilistiska särdrag vilka användes för att bestämma författare. Experimentella resultat redovisas för två samlingar litterära verk: en mindre med 27 böcker skrivna av 25 författare och en större med 11 063 böcker skrivna av 316 författare. Neurala nätverk användes för att koda de valda särdragen som vektorer varefter de närmaste grannarna för de okända texterna i vektorrummet användes för att bestämma författarna. För den mindre samlingen uppnåddes en träffsäkerhet på 91,25% genom att använda de 50 vanligaste funktionsorden, syntaktiska dependensrelationer och ordklassinformation. För den större samlingen uppnåddes en träffsäkerhet på 69,18% med liknande särdrag. Ett användartest visar att modellen utöver att bestämma författare har potential att representera likhet mellan författares stil. Detta skulle kunna tillämpas för att rekommendera böcker till läsare baserat på stil.

Nyckelord

Stilometri, Författarbestämning, Vektorrum, Neurala nätverk, Språkteknologi, Bokrekommendationer

Acknowledgments

I would like to thank Storytel and my supervisors Dave and Salla for providing me with a thesis topic that coincides with my personal interest in reading, writing, and literature. Thank you for the support along the way and many fun fikas and knowledge sharing sessions across the team that allowed me to learn something else also other than topics related to the thesis.

A big thank you to Jussi, my KTH supervisor who had so many good ideas and for being the best discussion companion. I would not have made it without you, if only for the motivational support.

Thank you to all the members of the Storytel data science team for making me have a good time during this spring and feel included, especially, Jesper, for sharing tips on the master thesis process, and George, for support and encouragement.

Thank you to my parents for the idea of studying computer science and machine learning in the first place, one of the best decisions of my life.

Thank you, Agnieszka, Barry, Erik, Tal, Ramtin, Poon, Mathias, Carro, Malcolm, Amy, Kasper, Gustave, for the companionship and support during this trying chapter of my life.

Marvin-, there is no doubt that I could accomplish anything without you having my back.

Stockholm, June 2021
Beichen Chen

"A book must be the axe for the frozen sea within us."
- Franz Kafka

Contents

List of Figures

List of Tables

List of acronyms and abbreviations

BNC British National Corpus

CB Content based

CF Collaborative filtering

CNN Convolutional Neural Network

ESN Echo State Network

GRU Gated Recurrent Unit

k-NN k-Nearest Neighbors

LDA Latent Dirichlet Allocation

LSTM Long Short-Term Memory

MLP Multilayer Perceptron

NLP Natural Language Processing

PCA Principal Component Analysis

POS Part-of-speech

ReLU Rectified Linear Unit

RNN Recurrent Neural Network

SGD Stochastic Gradient Descent

SVM Support Vector Machine

TTR Type-Token Ratio

Chapter 1

Introduction

1.1 Background

Stylometry is the field of research aimed at defining features for quantifying writing style (Holmes, 1994). The most studied question in stylometry has been authorship attribution, where given a set of texts with known authorship, we are asked to determine the author of a new unseen document. From a machine learning point of view it could be viewed as a multiclass, single label text categorization task (Sebastiani, 2002). Stylometry is a study dating back to the 1800s and traditionally it has been done by hand, e.g. counting the frequencies of certain words in texts and performing statistical analysis (Mendenhall, 1887; Mosteller and Wallace, 1964). With the availability of computational and machine learning methods, stylometry has evolved to employ a range of different methods such as Support Vector Machines (SVMs), Naïve Bayes, k-Nearest Neighbors (k-NN), and in most recent years neural networks (Stamatatos, 2009; Jockers and Witten, 2010; Diederich et al., 2003; Gamon, 2004; Savoy, 2020; Schaetti and Savoy, 2020; Jasper et al., 2018).

However, studies on stylometry using neural networks have been few. The question of whether deep learning models are useful for author attribution and whether neural models can outperform traditional ones remains open (Schaetti and Savoy, 2020). Neural networks (see Chapter 2.2.1) have the benefit of being able to perform feature engineering, meaning that instead of picking features by hand and using them directly for classification, the features are transformed into a different representation that perhaps benefits classification. An embedding (see Chapter 2.2.2) is such a representation - a vector of

continuous values for which the dimensions carry no concrete meaning but where the relative distances between embeddings for different samples can be useful in measuring similarity and hence perform author attribution.

The dataset used for author attribution is also a defining factor that puts limits on performance. Factors include the number of candidate authors, and the amount of training data. It is difficult to compare numbers reported in studies in an absolute way due to different datasets used, however we can look at how different features affect the performance of a model in the same study, and compare whether the effects are similar across studies.

1.1.1 Use case

A reader of a book, e.g. an audiobook or an e-book, that is happy with their recent reading or listening would likely want to continue that enjoyable experience by reading more material that is similar. The reader can find books that are written by the same author, same genre, time period, in the same cultural area, or with the same topical theme, but it is more difficult to establish which books are similar in style, i.e. in the way the author writes and uses language.

Storytel AB is an audiobook and e-book company that wants to improve their book recommender system by incorporating more aspects of similarity into the recommender. One factor that is hypothesized to have a positive effect on the recommendations generated is the style of the author. Hence, from an applied point of view the purpose of this thesis is to explore how to build and train a model that has potential to generate book similarities and recommendations based on style. This can complement traditional recommender systems based on user ratings or topic, and produce non-obvious recommendations that can positively surprise the user and increase customer value for the company.

1.2 Problem and research questions

The background leads us to ask three research questions on author attribution and a fourth question relating to author similarity. The first three follow naturally from the background however the question on similarity needs an explanation on triplet loss, and we will provide a short one here that will be complemented in Chapter 2.2.3. Triplet loss is a way of training a model to group samples from the same class into clusters. For the author attribution

task, this is exactly what we want, however it is not certain that the trained model will also preserve relative distances between clusters, i.e. authors. Specifically it means can the model answer the question, is author A closer to author B or author C? Not just being able to tell that author A, B, and C, are distinct. It is this relative similarity we need in order for the model to be useful in generating recommendations, hence the fourth research question.

The research questions for this thesis are:

What features will a neural network be able to use most effectively to perform author attribution?

How does the number of candidate authors affect author attribution?

How does using embeddings for author attribution compare to using the raw feature vectors used as input to the neural network?

How does a model trained for author attribution with triplet loss perform on an author style similarity task?

1.3 Delimitations

The following delimitations have been set due to time and resource constraints:

- Network hyperparameter tuning included a range of reasonable values for the smaller dataset, and for the larger dataset the tuning had as a starting point the values that performed best for the smaller dataset for that feature set.
- The optimal group size and minibatch size (see Chapter 3.2.1) were set at a reasonable value and never changed.
- Some lexical featuers such as word lengths or Type-Token Ratio (TTR), semantic features and character level features (see Chapter 2.1.3) were not included in the study due to the literature promoting them less than other features.
- The small model of the spaCy library for dependency parsing and POS tagging was used.

- The feature ablation performed (see Chapter 3.1.1) is not exhaustive as that would take too long. It is instead hypothesis based.
- The network architectures tested were limited to feedforward and LSTM networks (see Chapters 2.2.1 and 2.2.1). Other architectures found in the literature review such as Echo State Network (ESN) or Convolutional Neural Network (CNN) (Chapter 2.2.1) were not tested.
- Stylometry can also study effects of genre and time period, however the thesis focused only on the author.
- In the larger dataset, translated books were not filtered out as it was not obvious how to do this or if it is even possible.

1.4 Ethics and sustainability

Language models in general are biased toward the data that was used to train them, hence it is important we consider the effects of this when working with Natural Language Processing (NLP). In the case of this thesis, the most common functional words (see Chapter 2.1.3 and Table B.1) present a source of bias, e.g. when it comes to gender. It echoes the bias of existing literature on which the choice of top functional words are based, e.g. "his" (#26) and "he" (#15) are higher on the list than "she" (#28) and "her" (#45). However, since we utilize at least the top 50 functional words list for our study this will not matter as the order does not matter once we have all the words as features. It could be though, that certain words are left out due to being lower on the list and that the model will match less well with the contemporary or evolving literature that mirrors the development of society and values.

When working with machine learning sustainability should be considered even though that might seem like something unrelated to the work. It is valuable to ask the question of whether a network requiring more intense resources provides a considerable benefit compared to a simpler network and whether the increase in computing power is worth it. In the case of our study, we will see in the results chapter that actually simpler networks worked better for our specific problem.

Chapter 2

Background

2.1 Style and stylometry

This chapter provides an introduction to style and stylometry, the methods used in the quantitative study of style, and also discusses its limitations.

2.1.1 Style

A style is somebody's personal way of using language (Crystal, 2010). A language offers us many ways of saying something and we have a choice of which way we want to say it. Most of the time, one person will say it in a different way than another. How we choose to express a message can also depend on the circumstances. For instance, the same person writing a short story will write in a different way when writing a master's thesis. Style is all about choice, about the freedom within the constraints provided by a language. Some authors even like to break the normal rules of language to give themselves a distinctive style.

The village does not have a post office.
The village has no post office.
The village doesn't have a post office.
The village hasn't got a post office.
The village hasn't got no post office.
The village ain't got no post office.

The above is one example of how style can vary, progressing from a formal one to a casual one (Crystal, 2010). Below is another example of stylistic variation,

this time coming from the order of words.

Wishfully the teenagers sang songs together.
The teenagers wishfully sang songs together.
The teenagers sang wishfully songs together.
The teenagers sang songs together wishfully.

Apart from these two handpicked and simple examples, there are a multitude of other ways in which style can differ. This makes style complex, and sometimes almost intangible. Just think of all the authors you have read in your life, maybe your favorites. Chances are you consider them having distinct writing styles, but trying to define how their styles differ is most likely difficult. However, being mathematicians or data scientists we try to reduce complex things to something we can measure. From a literary or artistic perspective, this reduction can be questionable, as we will touch more upon in Chapter 2.1.5, but looking at previous studies in stylometry we see that the attempts at quantifying style have yielded some interesting results in classifying and profiling authors even though they might be simplified.

An important distinction to make is between style and content. Can we differentiate between the style and contents of a message? Savoy (2020) answers that although style and content are two different things, they should be viewed as two sides of the same coin. This is because the author chooses a style to support a particular message. Other scholars have a different view, that content is a source of noise in the study of style, since style should be consistent regardless of the topic of the writing. For instance, some studies have focused on how to remove topic information in the study of style (Stamatatos, 2018; Panicheva et al., 2019).

Influences of style include not only the author but also the genre and time period. Burrows (1992) has seen that texts written by the same author but in different genres have more variability than texts belonging to the same genre written by different authors. Some examples are that scientists are required to use the passive voice, or that the style of poetry and the style of a newspaper article are distinct. For the time period, each period has its own stylistic preference. For instance we talk of classical style or postmodern style. By analyzing speeches from U.S. presidents since the country's founding we see that the average sentence length decreases over time, which Savoy (2020) writes is influenced by a fast-paced life or in more recent years the frequent

use of texting and tweeting.

When it comes to the author, the style is shaped by individual preference. It could be the preference of one synonym over the other such as “actually” or “in fact”, or that some people tend to write longer descriptions in detail while others opt for concise scripts, or many more personal idiosyncracies. Stylistics is the field of applied linguistics that study individual style (Savoy, 2020).

2.1.2 Stylometry

As mentioned in the previous chapter, in order to study style as mathematicians or data scientists it is necessary to represent style in measurable form. Stylometry is the field of research aimed at defining features for quantifying writing style (Holmes, 1994). Before going into how the quantification has been done practically, we present some applications of stylometry.

The most studied question in stylometry has been authorship attribution, or author identification (Savoy, 2020). The problem statement is that given a set of texts with known authorship, can we determine the author of a new unseen document? This question could be closed-set or open-set, meaning either the unseen document is guaranteed to be of the same author as one of the texts of known authorship, or that it could be someone completely unknown. Most research has been done in the closed-set setting. From a machine learning point of view it could be viewed as a multiclass, single label text categorization task (Sebastiani, 2002).

A related problem is the one of authorship verification. Here the question is, did a given author write a given text? Hence it is a problem with a binary response. The system is given a set of texts written by the candidate author and one unknown document. This problem is more general than the one of authorship attribution since we can solve the attribution problem if we can solve the verification problem effectively, by transforming the attribution problem to a series of verifications - one per candidate author (Savoy, 2020).

The applications of authorship attribution and verification are many. Traditionally it has been used to attribute anonymous or disputed literary texts to known authors (Burrows, 2002; Hoover, 2004). A well known and one of the first studies was a literary study of texts by three founding fathers of the USA: James Madison, Alexander Hamilton, and John Jay, known as the Federalist

Papers. A number of these papers have disputed authorship and Mosteller and Wallace (1964) used stylometric features to attribute them to Madison. Other applications that have come up in modern times include intelligence, e.g. attribution of messages to known terrorists (Abbasi and Chen, 2005); criminal law - identifying authors of harassing messages; civil law - copyright disputes (Chaski, 2005; Grant, 2007); computer forensic studies - finding out who wrote malicious code (Frantzeskou et al., 2006); and plagiarism detection (Savoy, 2020).

Apart from authorship studies, few studies have been made on the other influencing factors of style such as genre or time period. Karlgren and Cutting (1994) used a number of lexical and syntactic features to classify genres in the Brown corpus.

2.1.3 Stylometric methods and features

The very first attempts at quantifying writing style date back to the study by Mendenhall (1887) which was pioneering during that time. Mendenhall performed statistical analysis on the plays of Shakespeare by counting the frequencies of words having different word lengths. After that, the study by Mosteller and Wallace (1964) was another breakthrough where they used the frequencies of a small set of common words, e.g. “and”, “to”, and Bayesian statistical analysis to produce significant discrimination between the candidate authors.

Since then until the late 1990s research has experimented with a large variety of features such as sentence length, character frequencies, and vocabulary richness. An estimation made by Rudman (1997) showed that almost 1,000 different measures had been suggested by late 1990s. During this period, the methods proposed were computer assisted but not computer based - the aim was not to develop a fully automated system (Stamatatos, 2009).

Since the 1990s, more powerful machine learning algorithms became available to handle multidimensional and sparse data which allowed more expressive expressions (Stamatatos, 2009). New NLP tools have also been developed and enabled new forms of measures for representing style.

The decade of the 2000s also saw the rise of the internet and hence the

explosion of the amount of textual content, e.g. emails, blogs, and online forums. Authorship analysis also followed this trend and studies were made on so called real-world texts instead of trying to solve disputed literary questions (Stamatatos, 2009). Since the existence of social media even more textual content has been generated by billions of authors, and most recently studies have been done on such texts (Shrestha et al., 2017). These real-world texts are much shorter than books and it can be asked whether features and methods that work on one type of text also works on another.

The rest of the chapter describes specific features and the studies that have used those features, however it is worth to mention that this is by no means a defacto list of how stylometric studies should be done. As mentioned in the introductory chapter, style is very complex and the features described below is the current state of research within this area.

List of features

Stylometric features can be grouped into lexical, character, syntactic, and semantic features (Stamatatos, 2009). Lexical features have to do with individual words, while character features with individual letters or characters. Syntactic features have to do with the structure of a sentence, i.e. the grammar and the order of words. Semantic features concern the meaning of words.

Tables 2.1, 2.2, 2.3, and 2.4 show stylometric features gathered from (Savoy, 2020; Stamatatos, 2009; Hollingsworth, 2012).

Table 2.1 – Lexical features

Feature	**Explanation or example if needed**
Mean sentence length or sentence length	
Frequency of most used (functional) words	E.g. "the", "of", "and".
Vocabulary size	Number of distinct words.
Vocabulary richness	TTR is the ratio between the number of words (vocabulary size) and the number of tokens (text length). Yule's K and Simpson's D are two derivations.
Frequency of least used words	Instead of looking at the most frequent words, one can analyze the number of words occurring just once in the corpus (Hapax Legomena), or twice (Dis Legomena), or 3,...etc.
Lexical density	Indicates the percentage of lexical items (or content-bearing words) appearing in a text. Content bearing is the opposite of functional. This measure doesn't have direct connection with authorship attribution though (Savoy, 2020).
Percentage of big words	A big word is a word composed of six letters or more. A text with a high percentage of big words tends to be more complex to understand. This fact is confirmed by recent studies in neuroscience (Savoy, 2020).
Mean word length	
Frequency of word lengths	E.g. how many words have length 3.
Frequency of word n-grams	An n-gram is a continuous sequence of n items, in this case words. Usually a limited number of most frequent n-grams from the whole corpus.

Table 2.2 – Character features

Feature	**Explanation or example if needed**
Proportion of vowels/consonants	
Frequencies of letters	
Frequencies of letter n-grams	Usually a limited number of most frequent n-grams from the whole corpus.

Table 2.3 – Syntactic features

Feature	Explanation or example if needed
Frequencies of POS tags	
POS tag n-grams frequencies	Usually a limited number of most frequent n-grams from the whole corpus.
Sentence and phrase structure	
Rewrite rule frequencies	See "Syntactic features" section.
Dependency relation frequencies	See "Syntactic features" section.

Table 2.4 – Semantic features

Feature	Explanation or example if needed
Synonyms	Using WordNet, a database of synonyms.

We discuss lexical and syntactic features in detail since those are the ones used in this study.

Lexical features

According to Stamatatos (2009), the vast majority of authorship attribution studies are at least partially based on lexical features, and the most straightforward approach is a representation as a vector of word frequencies. As mentioned above, the use of common, or functional words have been used by Mosteller and Wallace (1964). These have been found to be among the best features to distinguish between authors (Argamon and Shlomo, 2005; Burrows, 1987). Recall the discussion in the previous chapter on content vs. style. These functional words carry no semantic information and are usually excluded from the features of topic based text classification. Functional words are mostly used

subconsciously by authors and are topic-independent, hence their capability to capture authors' purely stylistic choices across topics.

Which functional words and how many of them to use has been quite arbitrary with little explanation of how they were selected. For example Abbasi and Chen (2005) used 150 function words , Argamon et al. (2003) used 303 words, Zhao and Zobel (2005) used 365 words, Koppel and Schler (2003) used 480, and 675 words were used by Argamon et al. (2007).

Stamatatos (2009) writes that a simple and very successful method is to extract the most frequent words found in the entire corpus. Burrows (1992) considers sets of at most 100 frequent words to be adequate to represent the style of an author.

Both Stamatatos (2009) and Savoy (2020) write that vocabulary richness measures are considered unreliable by themselves in authorship attribution, since vocabulary size depends heavily on text length. As the length of the text increases, the size of the vocabulary increases, not linearly but quickly at the beginning and then gradually more slowly. Even the variants Yule's K and Simpson's D (and a number of other variants) that are more stable have shown questionable results. Noting observations of single occurences of very characteristic words, i.e. Hapax Legomena, has been used to establish linguistic fingerprints of authors but since observational statistics at the low end of the frequency scale are susceptible to noise and random occurence this practice is also unreliable.

Common to all the lexical features are that they consider the text as a bag-of-words. Hence, word order and contextual information are disregarded. Word n-grams has been used as an attempt at improvement but they have not always given better results and pose problems such as high dimensionality and sparseness (Stamatatos, 2009). Gamon (2004) writes that word n-grams could potentially capture content related information instead of purely stylistic information. In their experiments they took measures to remove the content dependency in word n-gram frequency features, by changing proper nouns to "NAME" and singular personal pronouns to "Perspro".

Syntactic features

Using syntactic features is considered more elaborate than only using lexical features. Authors usually use similar syntactic patterns in their writing subconsciously, making it a more reliable fingerprint of authorship (Stamatatos, 2009). However this requires accurate NLP tools that are able to perform syntactic analysis, making it language dependent. Naturally the features will contain noise if the parser produces errors.

There are two established ways of representing syntactic regularity in text: constituency grammar and dependency grammar (Matthews, 1981). They essentially aim at the same purpose, which is to represent the structure of a sentence with the correct relationships between the words. In constituency grammar, words are related to each other in a form of hierarchy. For instance, in the sentence “We sat on the bench in the park” the sentence (S) is made up of a noun (N) followed by a verb phrase (VP). Some scholars call this a rewrite rule: S -> N + VP. Furthermore, the verb phrase is also composed of parts. We have VP -> V + PP + PP in Figure 2.1, where “on the bench” and "in the park" are prepositional phrases noting where we sat. Finally, each prepositional phrase includes a preposition followed by a determiner and a noun.

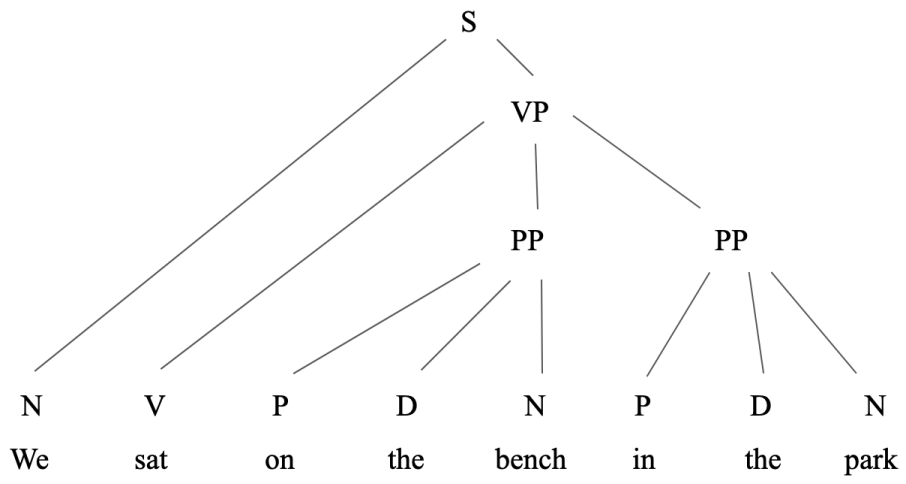

Figure 2.1 – Constituency grammar

In stylometry rewrite rule frequencies have been used as features by Baayen

et al. (1996) and Gamon (2004). Experiments have been inconclusive whether it alone performs better than lexical measures alone, but the combination of the two performed better than lexical measures alone.

Dependency grammar on the other hand doesn't break down a sentence, or parts of a sentence, into smaller "building blocks", rather, there is a relationship between words and the root or a head word. The root is usually the main verb in a sentence - what is happening. In the sentence "We sat on the bench in the park", "sat" is the root. Then, "we" has a nominal subject relation to "sat" because it answers the question "who sat?". Further, "bench" has an object of preposition relationship with "on" since it tells us what we sat on. We also have determiner relationships between "the" and "bench" as well as "the" and "park", specifying which bench and park we are talking about.

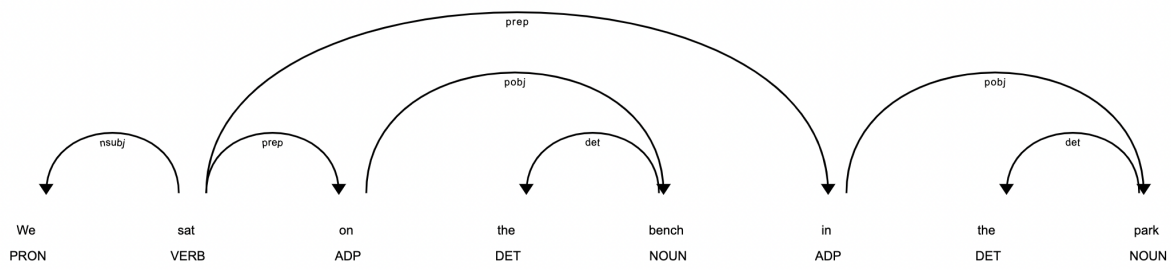

Figure 2.2 – Dependency grammar using spaCy parser

Only one study was found to use dependency grammar. Hollingsworth (2012) writes that most studies using syntactic features use the constituency grammar but that dependency grammar is worth exploring. The study indicates that dependency grammar features can produce as good results as lexical features or POS features, but does not test whether one can achieve better results with both. The way features are extracted is by taking each chain of grammatical dependencies from a token to its root. In the example sentence above, det - pobj - prep - root would be one chain, from "the" to "bench" then to "on", and "sat". The frequencies of these chains are the features, but the study also tested with another approach of simply counting frequencies of individual dependency relations. The results show actually that the chain features scored around the same as individual dependency relation frequencies, and were even worse when less than top 50 features were used.

Apart from studying sentence structure through a parser, a simpler way of extracting features is looking at the POS of each word, such as noun, verb, etc. This requires a POS tagger tool, and they are usually quite accurate (Stamatatos, 2009). POS tag frequencies or n-gram frequencies have been used by many studies (Argamon-Engelson et al., 1998; Gamon, 2004; Diederich et al., 2003; Koppel and Schler, 2003; Kukushkina et al., 2001; Zhao and Zobel, 2007). Nevertheless, POS tag information by itself only gives a hint of the sentence structure since it is a bag-of-words approach. Only by using a sentence parser and grammar structure methods described above can we obtain information on how the words are combined to form phrases, and how the phrases are in turn combined to form sentences.

Karlgren and Eriksson (2007) proposed a model based on just two syntactic features - occurence of adverbial expressions and the occurrence of clauses within sentences. The novelty in the study was a paradigm of looking at the variation of the features within a text instead of treating the entire text as a static whole. The features were represented as binary: 0 or 1. As an example, three sentences in a sequence where the first contained an adverbial, the second and third did not, would be represented as 100 in the adverbial dimension. By using a sliding window approach through the entire text and extracting n-sentence sequence binary representations, the frequencies of these representations were then analyzed. Although inconclusive, Karlgren and Eriksson (2007) and Stamatatos (2009) write that this is a promising technique that needs further investigation, since it treats texts and style more realistically than simply measuring average frequencies.

Feature engineering

All of the features described above are hard crafted features that have been chosen because somebody had a hypothesis of why they can have an influence on style, and then tested to show that they are indeed useful. A different approach that has been taken by more modern studies using neural networks and deep learning is feature engineering, or representation learning. Not many stylometric studies have used this approach, but within other fields of study the application of feature engineering is abundant.

This means that the machine learning model will itself learn the relevant features for solving the task at hand, in this case author attribution. The

advantages are that if lucky, the model might be able to find useful features that one did not think of while manually constructing features, and entirely new patterns might be found that are of interest. On the other hand, one loses interpretability as the automatically learned features are of an abstract dimension.

Some stylometric studies that have used deep learning and a feature engineering approach are described in Chapter 2.2.1. However, feature engineering and hand crafted features need not be entirely separate, since the manual features can be used as an input to learn new higher level features. These ideas will be explored more in the chapters on neural network and embeddings.

2.1.4 Datasets used in stylometric studies

The datasets used in stylometric studies have mostly been very small (Koppel et al., 2011). For example, the study by Gamon (2004) uses a few books by the three Brontë sisters and manages to distinguish between their style with high accuracy. Baayen et al. (1996) uses works by just two authors. The Federalist Papers only have three candidate authors. Koppel et al. (2011) writes that nearly all research in the field only consider the simplest version of the problem - given a long anonymous text, attribute it to a small closed set of candidate authors for whom we have extensive writing samples. This vanilla version of authorship attribution as Koppel et al. (2011) calls it, does not arise very often in the real world. In the real world there may be thousands of candidate authors, and the author of the anonymous text might not be any of the known candidates. Also, the text for each known candidate and/or the unknown text might be very limited. These are difficulties that have hardly ever been addressed in current research.

Some of the few studies that have used larger candidate author sets are by Madigan et al. (2005) (114 authors) and Luyckx and Daelemans (2008) (145 authors). Only Koppel et al. (2006) have considered candidate sets with thousands of authors. The studies by Koppel et al. (2006) and Luyckx and Daelemans (2008) show that similarity based methods are beneficial for datasets with a very large amount of candidate authors.

Koppel et al. (2011) proposes a method for author attribution when the condition is closer to the real world. The method reminds us of ensemble learning. Instead of using one set of features to measure similarity and

attribute an unknown text to a candidate author, they used 100 feature subsets where random features are picked out of all the features, and attributed the unknown text 100 times. The most frequent answer was picked as the predicted author, or "don't know" if the most frequent answer was not frequent enough. For features, they used character 4-grams. Their experiments used a set of around 10,000 texts, supposedly each from a different author although this was unclear in the paper. They also made variations studying the effect of the number of candidate authors, hence some experiments involved a subset of the authors. Their best results were when the number of candidates was 1,000, with a precision of 93.2% and coverage of 42.2%.

2.1.5 Limitations of stylometry

As mentioned in the introductory chapter, stylometry is a simplification of style. Although used widely, literary scholars are often sceptical of quantitative analysis, especially if it involves attempting to reduce literary style to statistics and low level language features (Stubbs, 2014). The deeper thematic and symbolic patterns that are necessary to fully interpet style are not possible with the current stylometric methods.

One example is the use of metaphors (Steen, 2014; Fontaine and Stavick, 2004). Certainly it is visible that some authors have a tendency to use imaginative metaphors while others have a concrete sort of style. Another example is that some authors tend to use suggestive language where the meaning is often between the lines. How can we extract this kind of information using computational methods? Is it even possible?

Further, it is difficult to imagine how the use of other literary tools such as irony and idioms can be detected by a computer. Finally, can a computer know that Murakami likes to make references to music and songs regularly in almost all of his books? In the attempt to reduce style into quantitative representations, we doubtlessly lose some of the nuances and subtleties that may be significant. A computer scientist with an interest in literature might hope to come up with innovative methods that capture more of the style and bring stylometry closer to its origin.

2.2 Machine learning methods for stylometry

Traditionally stylometric studies have involved manually analyzing texts and counting words by hand (Mendenhall, 1887; Mosteller and Wallace, 1964; Christopher, 2016). With the availability of computers and machine learning methods, the study of stylometry has been made simpler. Classifiers proposed include k-NN, Naïve Bayes, and SVM (Savoy, 2020; Stamatatos, 2009; Jockers and Witten, 2010; Diederich et al., 2003; Gamon, 2004). These classification methods all follow the same general procedure. First text samples from a set of authors are preprocessed to extract appropriate features that are able to discern between different authors. The classifier is then trained to discriminate between the style of the different authors based on the training samples. The resulting model is used to predict the most probable author of unknown texts (Kocher and Savoy, 2017).

In the following subchapters we will not focus on the methods mentioned above but on neural networks, a type of machine learning method that has grown in popularity and significance over the past decade. Neural networks have achieved state-of-the-art results in a range of machine learning fields and also specifically within NLP. However, studies on stylometry using neural networks have been few, with some less recent papers using the feedforward network architecture, but no peer reviewed articles using recurrent networks like LSTM or Gated Recurrent Unit (GRU) (see Chapter 2.2.1). The question of whether deep learning models are useful for author attribution, and whether neural models can outperform traditional ones, remains open (Schaetti and Savoy, 2020).

2.2.1 Neural networks

What is a neural network?

A neural network is a kind of machine learning model whose architecture is inspired by the structure of the human brain. Like the brain, a neural network is composed of many neurons linked to one another. A neuron in the machine learning sense is an operator that takes in some input, performs one or several operations on it, and produces some output. This output, is then passed on to the next neuron or neurons in the network. Neural networks are composed of layers of neurons, usually with the raw features providing input to the first

layer, the output of the first layer providing input to the second layer, and so on. There are shallow and deep networks, depending on whether it has few or many layers.

Feedforward neural networks

A feedforward network, also known as Multilayer Perceptron (MLP), is a neural network where a neuron's operation is that it multiplies the input by some weights using a dot product, and applies an optional activation function to the result to get the final output. This final output, a scalar, is then passed on to the next neuron or neurons in the network. In practice, the feedforward network is simply a chain of matrix multiplications, where the input gets transformed into the output by going through multiplications with weight matrices. The optional activation function in each layer provides a non-linear transform to the features.

There have been stylometric studies using feedforward networks. Tweedie et al. (1996) used a two layer feedforward network to perform a binary classification - "is this author A or author B?", on the Federalist Papers. As input features they used 11 common functional words, see Figure 2.3. The predictions from their model agree with the study by Mosteller and Wallace (1964). A similar approach was used by Matthews and Merriam (1993) to attribute some purported works of Shakespeare to either Shakespeare or Fletchers. Here they also used only functional words, this time ratios between different words, resulting in just five input features per sample. A similar network has been applied to authors of newspaper articles by Kocher and Savoy (2017), with accuracies between 42.8% and 69.2% depending on the number of candidate authors. Finally, Hoorn et al. (1999) used a three layer feedforward network to classify poetry, with an accuracy of 80-90% when the candidate author number was two, and around 70% when the number of candidate poets was three. The features they used were character n-gram frequencies.

One disadvantage of using a feedforward network is that the text is usually represented as a bag of words, where features are frequencies of words, characters, n-grams and such. With this approach the word order is lost and thus we have the same representation for different sentences if the words used are the same (Le and Mikolov, 2014).

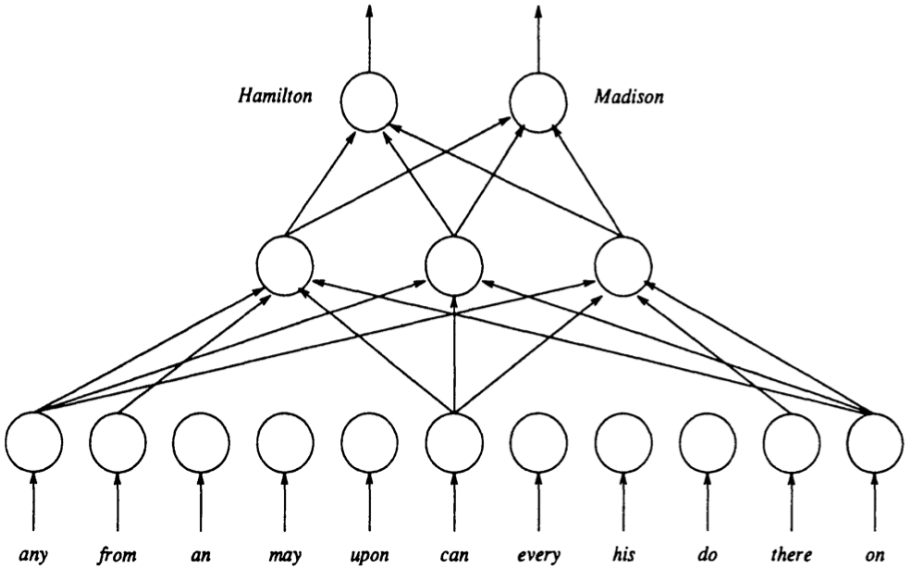

Figure 2.3 – Structure of the Federalist neural network (Tweedie et al., 1996). In this figure, some arrows are omitted to avoid clutter - there should be arrows connecting each input to each neuron in the hidden layer.

Recurrent neural networks

A Recurrent Neural Network (RNN) is another type of neural network, built essentially from a single cell i.e. neuron that is repeated N number of times. RNNs are used for sequential data, for instance texts. For example, for a text with 1,000 words, if each word was modeled as one input vector, N would be 1,000. Operations within the cell will produce an output for this particular cell, and also a cell state that is passed on to the next cell in the sequence. Because of the transfer of cell state from one cell to the next, RNNs have a memory and can retain information from earlier parts of a sequence. However, unfortunately for long sequences the memory retention of a vanilla RNN is not good enough (Olah, 2015).

An LSTM, or Long Short-Term Memory network, is a type of RNN capable of learning long-term dependencies, that vanilla RNNs have trouble with. They were specifically designed to overcome this issue (Hochreiter and Schmidhuber, 1997), with four layers within each cell performing different operations such as masking which part of the information from the previous cell to forget, and which part of the new input to add to the cell state. However,

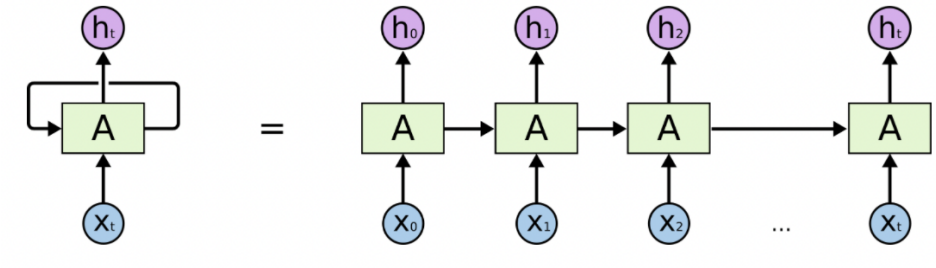

Figure 2.4 – Recurrent neural network (Olah, 2015)

it remains an open question whether this capability can help in authorship attribution (Schaetti and Savoy, 2020). It depends on the dataset size, which is usually limited in this field.

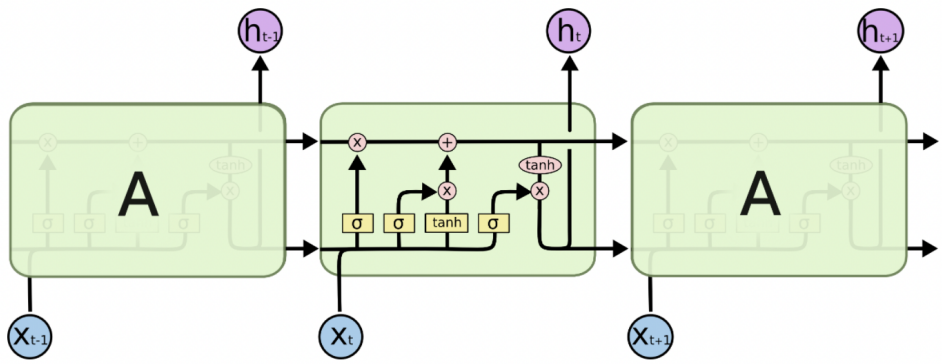

Figure 2.5 – LSTM inner operations (Olah, 2015)

The paper by Schaetti and Savoy (2020) is one of the few studies that have used RNNs for authorship attribution. Both shallow and deep models are evaluated: ESN, LSTM, and GRU. The features they use are word embeddings, character n-gram embeddings, and they perform the task at document and sentence level using the Reuters C50 dataset. The dataset contains news articles written by 50 authors, 100 articles each. The articles have been selected to minimize the topic factor. At the document level, their study shows that ESN models can perform better (93.40%) than the best classical method SVM (92.13%). At the sentence level, GRU outperforms Naïve Bayes with 86.57% and 74.86% respectively. Both the ESN and GRU model used word embeddings as input features. They also tested using only POS and functional words as input to

the ESN, however this did not yield good performance (59.9% and 64.9%). LSTM models trained with word embedding features performed at 84.50% and 83.45% on document and sentence levels, making it worse than classical methods at document level but better at sentence level. The article did not explain how the neural networks were trained, i.e. what kind of loss they used.

Since LSTM and GRUs have produced state-of-the-art results in many other NLP tasks, Schaetti and Savoy (2020) found it surprising they did not in their study. Their hypothesis is that it is due to the small size of available data per class compared to other NLP tasks, and this benefits shallower architectures. Another theory is that time-related dependencies are not long enough compared to ESN.

A different study by Jasper et al. (2018) used LSTM with Fasttext pre-trained word embeddings to perform authorship attribution on short texts of 1-3 sentences. They argue that while traditional stylometric studies have manually defined and extracted features that tangibly represent style, deep learning has become the standard in pattern recognition tasks and have the ability to learn the features required for solving the task at hand. With deep learning we can obtain feature vectors of much lower dimension than the raw features. Another study prior to their study used unsupervised learning to produce the feature vectors (Ding et al., 2016), but Jasper et al. (2018) used a supervised approach aimed at spatial clustering. In addition to capturing the writing style of an author, the model is presented negative samples in order to find discriminatory features to compare authors and distinguish them from each other. This is done through using triplet loss (see Chapter 2.2.3) to train the model. In addition, they used a data augmentation strategy, sample grouping, and dropout within the network. Their model was evaluated using the PAN 2014 challenge dataset, an author verification task, and achieved an Area Under the Receiver Operating Characteristic Curve (AUROC) of 0.92.

Convolutional neural networks

Shrestha et al. (2017) presents a method for authorship attribution of tweets using CNNs on character n-grams. Their hypothesis is that the model is able to capture patterns at the morphological, lexical, and syntactic levels by starting at short sequences of characters and generating representations for longer sequences using convolutions. They compare their proposed models

against other deep learning models such as LSTM trained on bigrams, and CNN trained on word embeddings. They found that the CNN trained on character bigrams performs best, at 76.1% with a dataset of 50 authors of 1,000 tweets each. Character based inputs, writes Shrestha et. al, specialize on the style of a text while word based inputs specialize on the content which is less important for authorship attribution.

2.2.2 Embeddings

The term embeddings refers to a distributed representation of an entity as a vector of continuous values. One of the most widely used types of embeddings are word embeddings, developed by Mikolov et al. (2013). Sometimes "word vector" and "word embeddings" are used interchangably. The dimensions of the vector are abstract hence do not carry any meaning, and the absolute position of the vector in the vector space also does not carry any meaning. It is the relative distances between the vectors that are of significance and tells us how similar one word is to another. Word vectors are trained through semi-supervised learning by using a sliding window of a number of words through the entire text. Mikolov et al. (2013) proposed two ways of training: skip-gram and continuous bag-of-words. The skip-gram approach works as follows: for each window, using the word in the middle (focus word) one tries to predict the words on both sides of the middle word (context words) and the task is to maximize the probability that the context words occur with the focus word.

The concept of embeddings has been applied to other entities other than words, for example images (Kiela and Bottou, 2014) and items (Barkan and Koenigstein, 2016). There have been attempts at representing paragraphs or entire documents as a vector (Le and Mikolov, 2014), however this approach does not take into account the style of the text. Jasper et al. (2018) attempts to represent the style of a text using embeddings obtained from an LSTM network (see Chapter 2.2.1) however does not provide an explanation of why the input features should be off-the-shelf word embeddings. Although all of these are called embeddings, the way of obtaining them may differ.

The clear benefit of embeddings is the ability to compare one entity with another to find the similarity between them with a simple and efficient computation, e.g. cosine similarity. Embeddings also help compress the size of the data in the case that the feature vector is larger than the resulting

embedding.

One way to perform author attribution using embeddings is k-NN. The k specifies how many closest neighbours to take into account. For instance, using a 3-NN approach, we would calculate the distance between the unknown text's embedding and all known embeddings. From the top 3 candidates, we would attribute the unknown text to the candidate author that appears most frequent, or the first one if all three are different (Kocher and Savoy, 2017).

A limitation to embeddings is that interpretability is lost since we no longer know which original features contribute to the predicted outcome. Savoy (2020) writes that a proposed attribution should be supported by linguistic reasoning, or by highlighting some language pattern similarities that cannot occur by chance.

Principal Component Analysis (PCA) is a dimensionality reduction method (James et al., 2013) that can be useful for visualizing embeddings. Since the embeddings can have high dimensionality, PCA reduces them to two or three dimensions that can be plotted and an intuitive view of relative distances and clusters can be obtained. In essence, PCA aims to transform the vectors into a new space and find a new set of orthogonal axes for the embeddings, where each new axis, or principal component, is a linear combination of the original features. The principal components obtained are in order of how much variance in the data there is among the axis, hence taking the top two or three principal components we can often already get a good indication of how the embeddings relate to each other also in the original, high dimensional space.

2.2.3 Triplet loss

Backpropagation and gradient descent

The loss function in a neural network is what drives the learning - it is a measure of how far the predicted outputs are from the actual outputs, i.e. labels. The loss is propagated backwards from the output layer to every layer that has contributed indirectly to the output, and during this process a gradient is calculated for the weights of each layer - the gradient of the loss with respect to the model parameters (weights). The gradient tells us which direction the model parameter should move to give the largest increase in the loss, so if we want to decrease the loss as much as possible, we update the parameters in the

opposite direction of the gradient (Goodfellow et al., 2016).

Triplet loss function

Triplet loss is a function for constructing the loss in order to learn spatial clusters in data that are of distinct classes. The input to the loss function is usually embeddings since one can easily calculate the distance between two samples. Hence, usually the pipeline consists of raw data being fed to a neural network that outputs an embedding for each sample, that are then grouped into triplets to calculate a loss. An example of this is the figure from the study by Schroff et al. (2015).

Figure 2.6 – Example model structure using triplet loss (Schroff et al., 2015)

In each triplet, there is a sample of one class called the anchor (x), another sample of the same class called the positive (x^+), and a third sample of a different class called the negative (x^-). The idea is that, for an untrained network, we would find many triplets where the negative sample is closer in distance to the anchor than the positive sample. These misclassified triplets give us a loss according to the function:

$$L(x, x^+, x^-) = \frac{1}{N}\Big(\sum_{i}^{N} max\{d(x_i, x_i^+) - d(x_i, x_i^-) + m, 0\}\Big) \qquad (1)$$

where m is a margin of our choice, and d is the Euclidean distance function (Jasper et al., 2018). N is the number of triplets in a batch or minibatch, as we will describe later.

As the network learns, the weights are updated so as to push samples of different classes away and samples of the same class together, resulting in clusters (Figure 2.7).

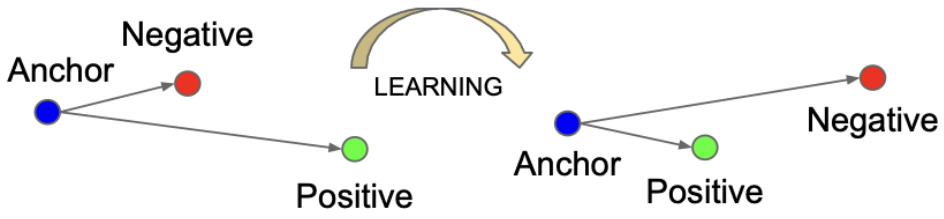

Figure 2.7 – Learning with triplets (Schroff et al., 2015)

Triplet mining

Choosing the triplets to train with can be done in multiple ways. First let us explain the different types of triplets that can be found in the dataset embeddings (Moindrot, 2018).

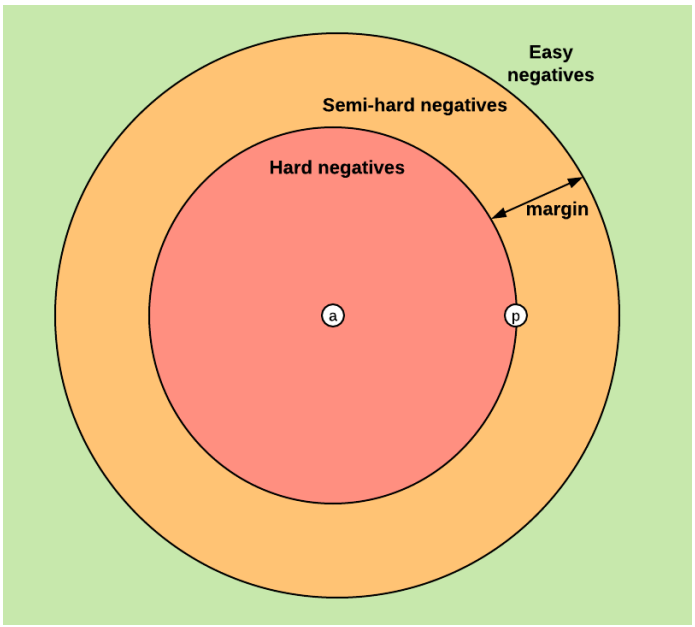

Figure 2.8 – Types of triplets (Moindrot, 2018)

First we have easy triplets, where the loss is 0 since $d(x, x^+) + m < d(x, x^-)$. The negative sample is already further away from the anchor than the positive sample plus a margin. These triplets are not used in training since they are already correct.

Next there are the hard triplets, where the negative is closer to the anchor than the positive: $d(x, x^-) < d(x, x^+)$. These are triplets that the network has to work hard to get right, since they need to be pushed further than the semi-hard triplets.

In semi-hard triplets the negative is further away from the anchor than the positive, but not far away enough since it still lies within the margin. Hence the loss will still be positive: $d(x, x^+) < d(x, x^-) < d(x, x^+) + margin$.

In Figure 2.8, the three triplet types are illustrated with respect to the position of the negative sample.

Two strategies for selecting triplets for training exist: batch all and batch hard. In batch all, all valid triplets are selected (i.e. the anchor and positive have the same class, the negative has a different class), and the loss of hard and semi-hard triplets are averaged. In batch hard, each sample in a batch acts as the anchor once, for which the hardest positive (the positive furthest away from anchor) and hardest negative (the negative closest to the anchor) is chosen. According to Hermans et al. (2017), batch hard works best since they are actually not the absolute hardest, but moderate triplets, in the entire dataset since they are only hardest within a small subset of the data. However, Moindrot (2018) writes that it depends on the dataset and should be decided by comparing the performance.

2.3 Book recommendation systems

The above chapters have given us an understanding of stylometry and machine learning methods for studying style. We have mostly discussed the problem of authorship attribution, which is the most widely studied task within stylometry (Savoy, 2020). Now we will look at a different although related task, which is the search for author similarities. Instead of asking the question, “given these candidate authors, who wrote this text?” the question is now “given a text, which other authors write in a style most similar to this text?"

The purpose of asking this question is that one can use the similarity rankings as a basis for performing recommendations. The following sections briefly introduce some of the common recommender systems.

2.3.1 Collaborative filtering

Collaborative filtering (CF), like its name suggests, makes use of the ratings of community users to make its predictions. This can be user based, where the system finds similarity between users, or item based, where similarity is computed between two books. CF is simple to develop as it only requires an item-user rating matrix (Alharthi et al., 2018*b*). However the matrix can be sparse especially where there are new items or users, known as the cold start problem. Since there is little user data about newly released books they tend not to get recommended. On the other end, very popular books can be considered by the system to be similar to nearly everything. Two other problems with CF are the gray sheep problem and shilling attack. In the former, a group of users given the "gray sheep" name has special tastes that neither agree or disagree with the majority of users and could be hard for the system to handle (Zheng et al., 2017). In the latter problem, fake ratings are employed as a form of promotion (Su and Khoshgoftaar, 2009).

2.3.2 Content based systems

Content based (CB) recommenders draws its predictions from the content of the books themselves. This could be the title, summary, outline, whole text or metadata - author, year of publication, publisher, genre, etc. (Alharthi et al., 2018*b*). The CB recommender could predict recommendations based on single books the user has preferred or their purchase or reading history of several books. Two approaches are to extract the topic or style of books, the former known as topic modelling and the latter the subject of this thesis.

CB is helpful in solving the new item problem since it does not rely on community ratings. The recommendations made by a CB system are able to be justified through the features used. However, difficulties arise when items have inadequate data e.g. descriptions or when the CB becomes overspecialized and recommendations produced are not diverse enough (Pazzani and Billsus, 2007; Lops et al., 2011). In addition, analysis of text requires NLP tools in order to transform the unstructured text into features.

Stylometry based recommendations

As described previously, stylometry can help distinguish authors from each other but studies have also been done on whether stylometry can help find similarities between authors and provide a basis for a CB recommender system. Vaz et al. (2013) uses k-NN and Pearson correlation to find books most similar to books of a user's predefined favorite author. They used two different representations: one based on word frequencies and another on some other stylometric features not clearly specified in the paper. In another study, Vaz et al. (2012) proposed a stylometric CB system where features include vocabulary richness, document length, POS bigrams and most frequent words. They also tried representing books as a vector of words extracted by Latent Dirichlet Allocation (LDA) which proved to be the better variant. The Rocchio algorithm (Manning et al., 2008) was used to discover stylometric features that is of most importance to the reader. A dataset called LitRec was used for evaluation, and it showed that a hybrid of stylometric CB with CF performs better than CB or CF individually. Alharthi et al. (2018*a*) takes the deep learning model from Shrestha et al. (2017) and transfers the learned representation from the second last layer to a recommender system that predicts a user's future ratings. They write that their study is the first to apply information learned from an author identification model to book recommendations.

2.3.3 Hybrid and other systems

Many studies combine CF and CB to improve the quality of recommendations, such as Rajpurkar and Bhatt (2015) and Pathak et al. (2013) and the aforementioned Vaz et al. (2012). Other than CF and CB, there exists some other strategies like basing recommendations on demography or on association rules. An association rule means that two books have occurred in the same transaction (e.g. library loan, e-store purchase, put on a bookshelf in a streaming service) a significant amount of times, hence defining them to be associated with each other. Some other sources of data for performing recommendations are user reviews and social media (Alharthi et al., 2018*b*).

2.3.4 Evaluation

There is a fundamental problem in the evaluation of recommender systems - the fact that there is no ground truth. For instance, it is not possible to objectively state that an author (or book) A is more similar to author B than author C, except perhaps in relatively extreme cases where an overwhelming majority of readers would agree. Hence, if there is not a definite answer, how do we evaluate recommender systems?

Evaluation strategies

Previous studies have put evaluation methods into three categories: offline experiments, online experiments, and user studies (Shani and Gunawardana, 2011). Offline experiments essentially means comparing the system's recommendations against some kind of "ground truth". This pseudo-ground-truth consists of a dataset with ratings from users. This approach may or may not make sense for evaluating a CB system, since it is basically evaluating whether the CB system can make the same recommendations as a CF system which defeats the purpose of developing the CB system. Most of the studies presented in the survey paper by Alharthi et al. (2018*b*) on book recommender systems use offline evaluation.

In user studies, a recruited group of test users interact with the proposed recommender system in some way followed by quantitative or qualitative analysis (Alharthi et al., 2018*b*). The user can answer questions about the system, e.g. if they find the recommendations to be reasonable when presented lists of similar books, or an observer can monitor their behavior if interacting with a live version of the system. User studies have high cost and require volunteered or hired users to spend time on the evaluation tasks. It is difficult to do this on a large scale. User studies are also sensitive to the sample of users chosen since their perception might not represent the entire population. It is also important to keep the real goal of the research secret when performing a user study since users will try to prove the research aims correct when they are aware of them (Shani and Gunawardana, 2011).

Finally, online experiments also known as A/B testing involves launching the recommender system in production to a group of real users, and analyzing their behavior compared to a control group. Alharthi et al. (2018*b*) writes that this evaluation method gives the most reliable evidence of system performance.

However the downside is that if the tested recommender performs poorly the system can lose users and revenues. It is recommended to conduct offline experiments and/or user studies before performing online evaluation.

Chapter 3

Methods

This chapter presents the details of the data, preprocessing, and experiment setup. Two datasets were used, one smaller and one larger. Two kinds of networks were used in this study: feedforward networks and LSTM. However, LSTM was only tested on the small dataset as poor results led us to focus on the feedforward networks that gave us better results.

3.1 Data and preprocessing

This study uses two datasets from Storytel. The smaller dataset is a subset of the larger one, containing 25 authors all having English as the original language, see Appendix A. The authors were chosen so that most of them are well known.

The larger dataset consists of 11,063 books. They were picked from Storytel's e-books collection in a way that tries to have an even spread of genres. The entire collection was first grouped by genre, then sorted by author according to the number of books written by each author. Iterating through all genres, all the books written by the author with most number of books were picked, and then, in a second iteration, all the books written by the author with next most number of books, and so on until 10,000 books were reached. After that, all the books from authors in the smaller dataset were added to the books already picked, if they were not already present. In total, the dataset had 362 authors.

The preprocessing for books consisted of the following steps:

1. 20,000 characters were stripped from the beginning of the book, to take away material like table of contents, about the author, etc.

2. The book was split into fragments
3. Features were extracted for each fragment

Specifically, the books were split into fragments of 2,000 words each for the smaller dataset, and 11,000 characters each for the larger dataset. The switch to character based was so that it would be possible to trace back the location of start and end characters for each fragment in the text in order to display it later. It was calculated that 2,000 words were on average around 11,000 characters.

For each fragment, characters before the first sentence and after the last sentence were removed, using ".", "?", and "!" as sentence delimiters. During tokenization punctuation was also removed.

For the smaller Storytel dataset, only the first 20 fragments were used for each author, resulting in a size of 500 samples. For the large dataset, 367,732 fragments were obtained. However, we saw that the number of fragments for each author differed widely, and hence authors with less than 100 fragments were removed, and authors with more than 1,000 fragments were downsized to 1,000 fragments. The final large dataset had 210,962 samples. Since some authors were completely removed by this procedure, 316 authors remained.

3.1.1 Features

The experiments started with using a LSTM network on the small dataset. The features used were off-the-shelf word embeddings from FastText with a vector size of 300. This was compared to using the same embeddings but removing topic information by substituting all nouns with the vector for the word "cat".

Next, feedforward networks were tested with various feature module combinations, also known as feature ablation. Due to limited time the ablation was not exhaustive and a hypothesis driven approach was used. The different modules are:

- Top 10 functional words
- Top 50 functional words
- Dependency relations
- POS tags (all or only verbs)

- Top 51-100 functional words
- POS 5-grams

The functional words correspond to the lexical group of features described in the background chapter. The POS tags, dependency relations, and POS 5-grams correspond to the syntactic group. Since dependency relation and POS tag frequencies themselves do not contain information on the ordering of words used, POS 5-grams were used as a complement.

For all the features, the frequency of the words or syntactic traits were calculated. For functional words, the frequency was calculated per 1,000 words. For dependency relations and POS tags, they were calculated per sentence. For POS 5-gram, the absolute frequency was used. All features were normalized before training. The software used for dependency parsing and POS tagging is spaCy (spaCy, 2021).

The functional words were extracted from the British National Corpus (BNC) list (Kilgarriff, 1996). For instance, the top 10 functional words are as follows. For a full list of the rest of the features, see Appendix B.

Table 3.1 – Top 10 functional words

the	be
of	and
a	in
to	have
it	for

The POS 5-grams used were a subset the complete set of 5-grams generated by using a sliding window on all the texts in the larger Storytel corpus. The complete set contained 15,593 different POS 5-grams. Using the same principle as the chi-square (χ^2) test for significance (Gajawada, 2019), the effect of each feature is assessed by comparing its observed frequency to the expected frequency of occurrence for a given author, given a marginal distribution based on the size of the material for an author and the observed frequency of the feature in the entire corpus. Equation (2) shows how χ^2 values are calculated where O is the observed frequency and E the expected frequency. After this, the 5-grams are ranked according to for how many authors it was significant. In this case, we set the threshold to a χ^2 value

above 200. The top 100 5-grams were picked as features for this study.

$$\chi^2 = \frac{(O - E)^2}{E} \tag{2}$$

3.1.2 Dataset division

The datasets were divided into training, validation, and test sets with the division 70%, 10% and 20% respectively. The division was done on a class basis ensuring that there is the same distribution of each author in each subset of the data.

3.2 Experiment design

In this section the triplets selection and minibatch division strategy is described, followed by the neural network setup and hyperparameters tested.

3.2.1 Minibatch division and triplets

This study uses online triplet mining using the batch all approach. This means that for each minibatch, all possible valid triplets are generated, and then trained on the semi-hard and hard triplets. Easy triplets are filtered out as they are already correct. Valid triplets are ones where the positive sample is of the same author as the anchor, and the negative sample is of a different author. The batch hard approach was tested in the beginning on the Brown dataset and resulted in a collapsed model, so it was not used in further experiments.

In order to guarantee that there are a number of positive samples for each anchor in a minibatch, the minibatch division process is designed in such a way that "groups" of samples of the same author are randomly put into minibatches. In the small dataset, we use an optimal group size of 5, meaning that first, all the samples are divided into groups of 5 of the same author, and in cases where there are left over samples, they are added to groups so that some groups have size 6. The next step is to randomly shuffle these groups, and finally they are divided into minibatches. The following are the settings used for the two datasets:

Storytel small: minibatch size 50, optimal group size 5
Storytel large: minibatch size 100 (took too much time if larger), optimal

group size 10

In order for the model to train on as many different divisions of minibatches as possible, and hence get a chance to train on as many variations of triplets as possible, we change minibatches every epoch.

3.2.2 Network setup

For the feedforward network, features were fed into the input layer, and after going through hidden layers the output layer was used as the embedding. The following configurations of embedding size (E), and hidden layer size (H) were tested for the small dataset. A network with just one hidden layer was compared to a network with more hidden layers. Also, apart from these configurations we also tested a larger hidden layer where H = E. The network was tested with and without activation functions such as sigmoid and Rectified Linear Unit (ReLU).

Table 3.2 – Embedding size and hidden layer size

E	H
16	8
24	12
32	16
48	24
96	48

The learning rate was kept at 0.001 unless the loss diverged and exploded, in which case it was lowered to 0.0005 and then if not sufficient, to 0.0001. The momentum was kept at 0.9 with the Stochastic Gradient Descent (SGD) optimizer. The margin for triplet loss was set to 0.001. The networks were trained for 1,000 epochs, with stops at 200 and 400 epochs to monitor the accuracies. Three learning rate schedules were tested: none, step with 0.2 gamma every 400 epochs, and 0.2 gamma every 200 epochs. Gamma is the multiplicative factor for the learning rate decay.

For the larger dataset, we started out with testing the configurations that gave the best performance in the small dataset, as well as tested some larger configurations such as E = 256, H = 128. Here we trained the networks for 3,000 epochs, because it took longer to converge than the small dataset, and

monitoring accuracies also along the way. The exception is for the model with top 50 functional words as features, where it was trained for 1,000 epochs as it already converged. Different activation functions were tested: none, sigmoid, and ReLU. The larger dataset was trained with an NVIDIA K80 GPU.

For the LSTM network, we applied 2 layers and a sequence length of 1,000. This means that for each fragment of text we use the first 1,000 words and each word corresponds to one FastText word vector, and every resulting sample is a 1,000 x 300 matrix. The output, or embedding, size was chosen to be 128. The output at the last timestep is used as the embedding for the fragment. A network with 3 layers was also tested but did not improve the performance much and increased the training time.

For all of the experiments, we used a seed of 42 in order to keep the minibatch generation and initialization of neural network constant. PyTorch was used to train the networks.

3.2.3 Nearest neighbor accuracy

Since the task of this study is to perform author attribution, we use nearest neighbor as a method to attribute unseen test fragments to an author in our training dataset. We use a combination of four metrics: Euclidean 1-NN, Euclidean 3-NN, Cosine 1-NN, and Cosine 3-NN. 1-NN means that the author that the test fragment is attributed to is simply the author of the closest fragment. 3-NN means that we take the three closest fragments, and take the author that is majority of those, or the first one if all three are different. Finally, we take an average of the four different metrics.

For the small dataset, we used Scikit-learn's implementation of nearest neighbor (Scikit-learn, 2020), but it proved to be too slow for the larger dataset with many more test samples. An implementation called Annoy was used instead where the n_trees parameter was set to 10 and n set to 20 (Bernhardsson, 2021).

3.3 User testing of author similarity

The above sections describe the methods for assessing author attribution. The next step is seeing whether a model with good performance on author attribution also performs well in generating author similarity. For this purpose,

a user study was designed in which two models are assessed: the worst and best performing models using the larger dataset. These are the model trained using top 50 functional words, and the one using top 100 functional words + dependency relations + POS tags.

The user test uses triplets to test if the models evaluate author similarity the same as humans do. Instead of attribution where the positive sample is the same author as the anchor, here the positive sample is a different author that is closer in style to the anchor than the negative sample. First one fragment was picked for each author yielding a subset of the data. Afterwards, ten random authors were picked for each model as anchors. Next, using the nearest neighbor method the first and 50th nearest neighbor in the data subset were picked as the positive and negative sample respectively. The user test presented excerpts of the fragments in triplets and asked the user to choose which of the two answer excerpts they think is written by the same author as the question excerpt. Even though none of them are actually written by the same author as the question excerpt, the instructions were phrased this way to be intuitive to the user and mask the real purpose of the study. An illustrative example is as below, taken from *Pride and Prejudice* (Austen, 1813), *The Brothers Karamazov* (Dostoyevsky, 1880) and *Around the World in Eighty Days* (Verne, 1873).

Question: More than once did Elizabeth, in her ramble within the park, unexpectedly meet Mr. Darcy. She felt all the perverseness of the mischance that should bring him where no one else was brought, and, to prevent its ever happening again, took care to inform him at first that it was a favourite haunt of hers. How it could occur a second time, therefore, was very odd! Yet it did, and even a third.

Alternative 1: You must know that there is nothing higher and stronger and more wholesome and good for life in the future than some good memory, especially a memory of childhood, of home. People talk to you a great deal about your education, but some good, sacred memory, preserved from childhood, is perhaps the best education. If a man carries many such memories with him into life, he is safe to the end of his days, and if one has only one good memory left in one's heart, even that may sometime be the means of saving us.

Alternative 2: The train entered the station, and Passepartout jumping out first, was followed by Mr. Fogg, who assisted his fair companion to descend. Phileas Fogg intended to proceed at once to the Hong Kong steamer, in order to get Aouda comfortably settled for the voyage. He was unwilling to leave her while they were still on dangerous ground. Just as he was leaving the station a policeman came up to him, and said, "Mr. Phileas Fogg?"

There were 20 questions in total, where ten were triplets generated from the first model, and ten for the second. In the evaluation, we compare the human answers for all questions, which we consider ground truth, to how each model scored for all questions.

Chapter 4

Results

This chapter presents the results of the experiments and is divided into sections for the datasets used. We present the performance of each model in accuracy, which is the average accuracy of using 1-NN and 3-NN with both Euclidean distance and Cosine similarity. Also, the same average accuracy using only raw features instead of the embeddings from the trained model is presented as a comparison. The raw features are just the normalized values of the preprocessed features.

4.1 Storytel small feedforward

Table 4.1 shows the results from the feedforward networks trained on the small dataset. The hyperparameters indicated are the best ones from the grid search. The schedule parameters indicate the step and gamma used for the learning rate schedule. If not indicated, it means no schedule was used. A deeper network, larger embedding or hidden layer sizes, or applying activation functions did not give any increases in performance. For all of the models in the table, no activation function was used.

Using the best model - Top 50 functional words + all POS tags + dependency relations, and a dimensionality reduction technique called PCA, we visualized the embeddings obtained, shown in Figure 4.1. It is easier to see the distinct authors if we plot two at a time, and such comparisons can be seen in Figures 4.2 - 4.6. With PCA, the two dimensions obtained are abstract and hence the axes do not contain any meaning - it is the relative distance between samples that are of interest.

Table 4.1 – Results for the Storytel small dataset with feedforward networks

Features	Hyperparameters	Acc. (%)	Raw features acc. (%)
Top 10	E = 16, H = 8	65.00	45.50
Pos-verb	E = 64, H = 32, sch = [200, 0.2]	40.00	39.50
Pos-all	E = 48, H = 24, sch = [200, 0.2]	77.00	65.00
Deps	E = 16, H = 8	72.00	65.00
Top 10 + pos-verb	E = 32, H = 16, sch = [200, 0.2]	69.00	54.50
Top 10 + pos-all	E = 32, H = 16	84.25	68.00
Top 10 + deps	E = 48, H = 24	86.50	72.00
Top 10 + deps + pos-all	E = 24, H = 12	88.00	70.50
Top 50	E = 48, H = 24	82.25	75.50
Top 50 + pos-verb	E = 96, H = 48, sch = [200, 0.2]	84.75	73.00
Top 50 + pos-all	E = 24, H = 12	85.50	77.50
Top 50 + deps	E = 96, H = 48, sch = [400, 0.2]	89.00	82.00
Top 50 + pos-all + deps	E = 96, H = 48	91.25	78.00

4.2 Storytel small LSTM

In the table below results from the LSTM experiments are shown. The first row is by using off-the-shelf word embeddings from FastText and the second row is the same embeddings but substituting all nouns with the embedding for "cat" and hence removing topic information.

Table 4.2 – Results for the Storytel small dataset with LSTM

Features	Hyperparameters	Acc. (%)
FastText	E = 128, 2 layers	11.50
FastText cat	E = 128, 2 layers	13.00

4.3 Storytel large feedforward

For the large Storytel corpus, the feedforward models using sigmoid function performed better than models with ReLU or without an activation function. It was used in all the models in Table 4.3. Here pos means that all POS tags were used, equivalent to the pos-all module in the small dataset.

Table 4.3 – Results for the Storytel large dataset with feedforward networks

Features	Hyperparameters	Acc. (%)	Raw features acc. (%)
Top 50	E = 96, H = 48	49.83	46.81
Top 50 + deps	E = 96, H = 48	58.58	50.51
Top 50 + pos	E = 96, H = 48	58.54	50.91
Top 50 + deps + pos	E = 96, H = 48	63.12	51.43
Top 100 + deps + pos	E = 96, H = 48	67.58	55.90
	E = 256, H = 128	69.18	
	E = 448, H = 224	69.19	
Top 100 + deps + pos + pos-5-grams	E = 256, H = 128	70.54	47.22
	E = 448, H = 224	70.10	

Also for this dataset PCA was used to reduce dimensions of the resulting embeddings and visualize them. The model used was Top 100 + deps + pos with a setup of E = 256 and H = 128. This model was chosen as the best model since a larger network only gave marginal increase of performance, and the addition of POS 5-grams also gave marginal improvement while adding preprocessing steps and computational cost. For the visualizations, since there are over 300 classes the best was to single out an author or multiple authors while plotting the rest of the authors as gray. These can be seen in Figures 4.7, 4.8, and 4.9.

4.4 Author similarity user test

The results of the user test are shown in Table 4.4. Since each model contributed to ten triplets, the other ten triplets from the other model were run through the model to see if the position of the answer embeddings in relation to the question embedding is the same or different from the model that originally generated the triplet. For instance, in some cases, one model evaluated author A as being closer to author B than author C while the other model gave that author C was the closer one. Taking the human evaluation as ground truth, the table below shows how the two models scored. In total 11 people participated in the user study, and the threshold is the minimum percentage of votes that the majority answer has to have for the question to be counted. For example, a question in which 54.5% of users voted for one answer and 45.5% for the other is not counted in the rows where the threshold is above 60%.

Table 4.4 – Results for the author similarity user test

Features	**Threshold (%)**	**Acc. (%)**	**# correct/total**
Top 50	50.00	70.00	14/20
	60.00	62.50	10/16
	70.00	75.00	9/12
	80.00	88.89	8/9
Top 100 + deps + pos-all	50.00	75.00	15/20
	60.00	68.75	11/16
	70.00	66.67	8/12
	80.00	77.78	7/9

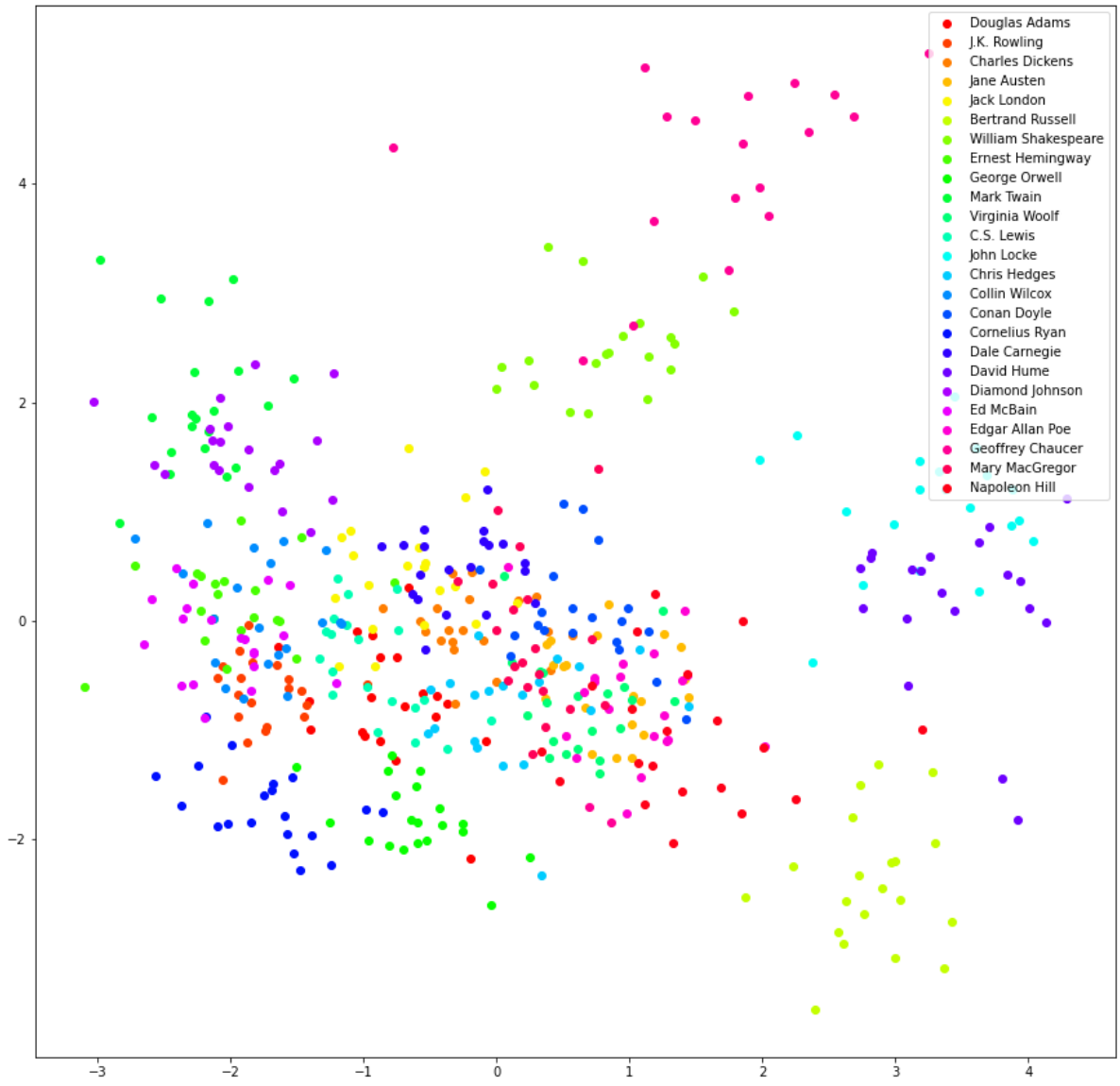

Figure 4.1 – All of the authors in Storytel small. The green, purple, and cyan dots on the bottom right are the philosophers Bertrand Russell, David Hume, and John Locke. The pink and green on the upper right are poets William Shakespeare and Geoffrey Chaucer.

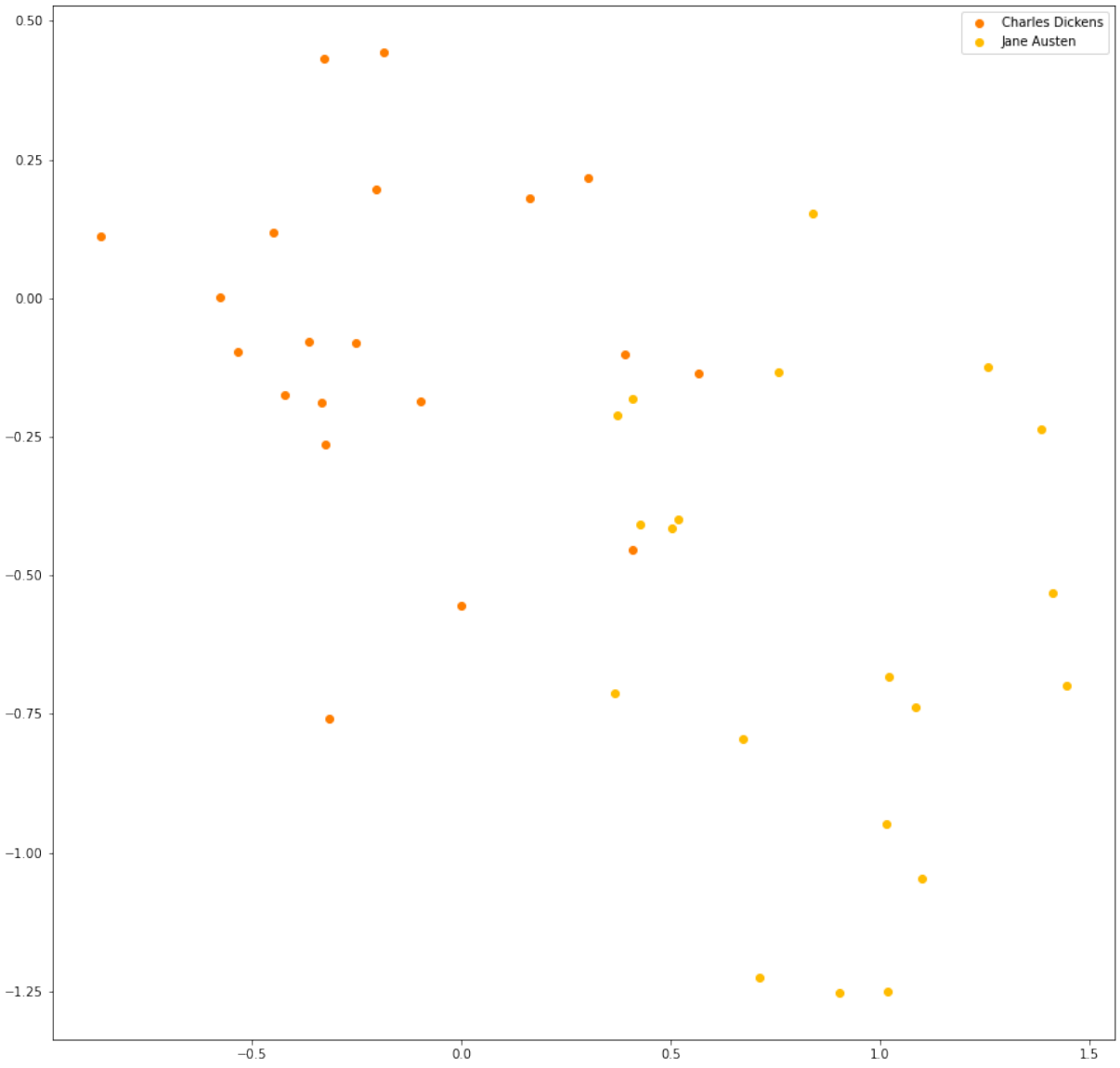

Figure 4.2 – Small dataset - Charles Dickens and Jane Austen - two English writers from the 19th century

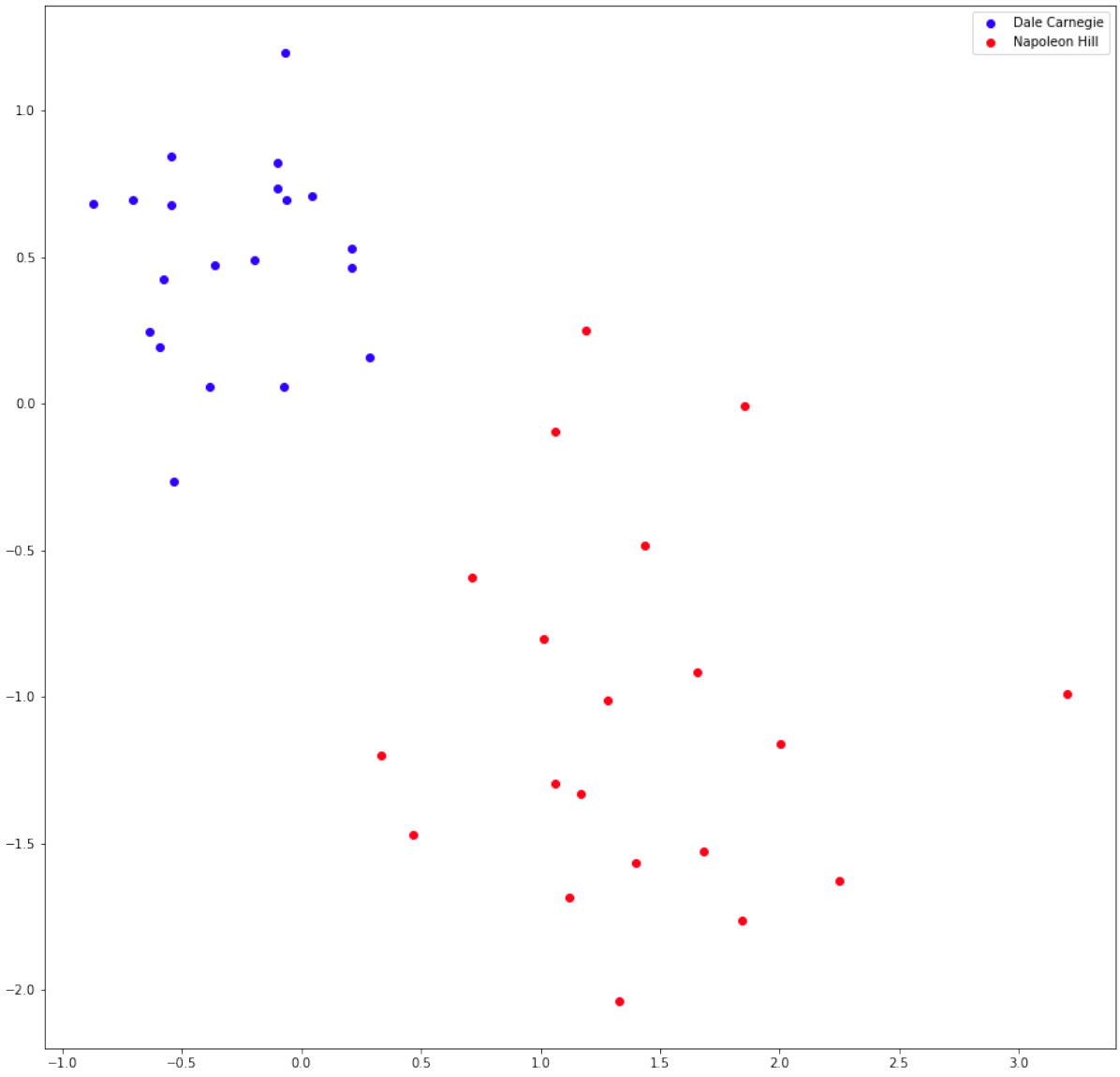

Figure 4.3 – Small dataset - Dale Carnegie and Napoleon Hill - two writers of personal development books

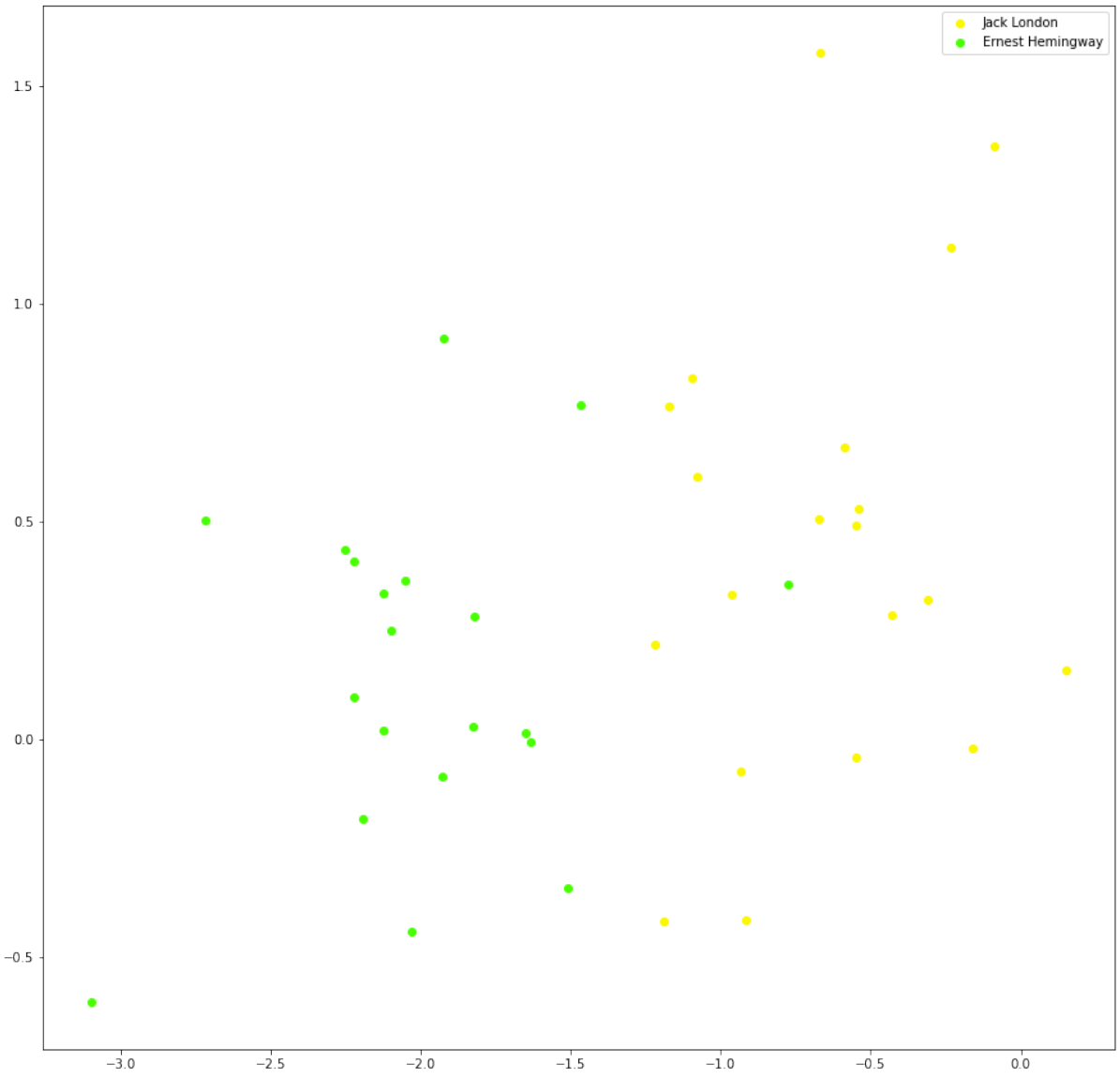

Figure 4.4 – Small dataset - Jack London and Ernest Hemingway - two American writers from the 19th - 20th centuries

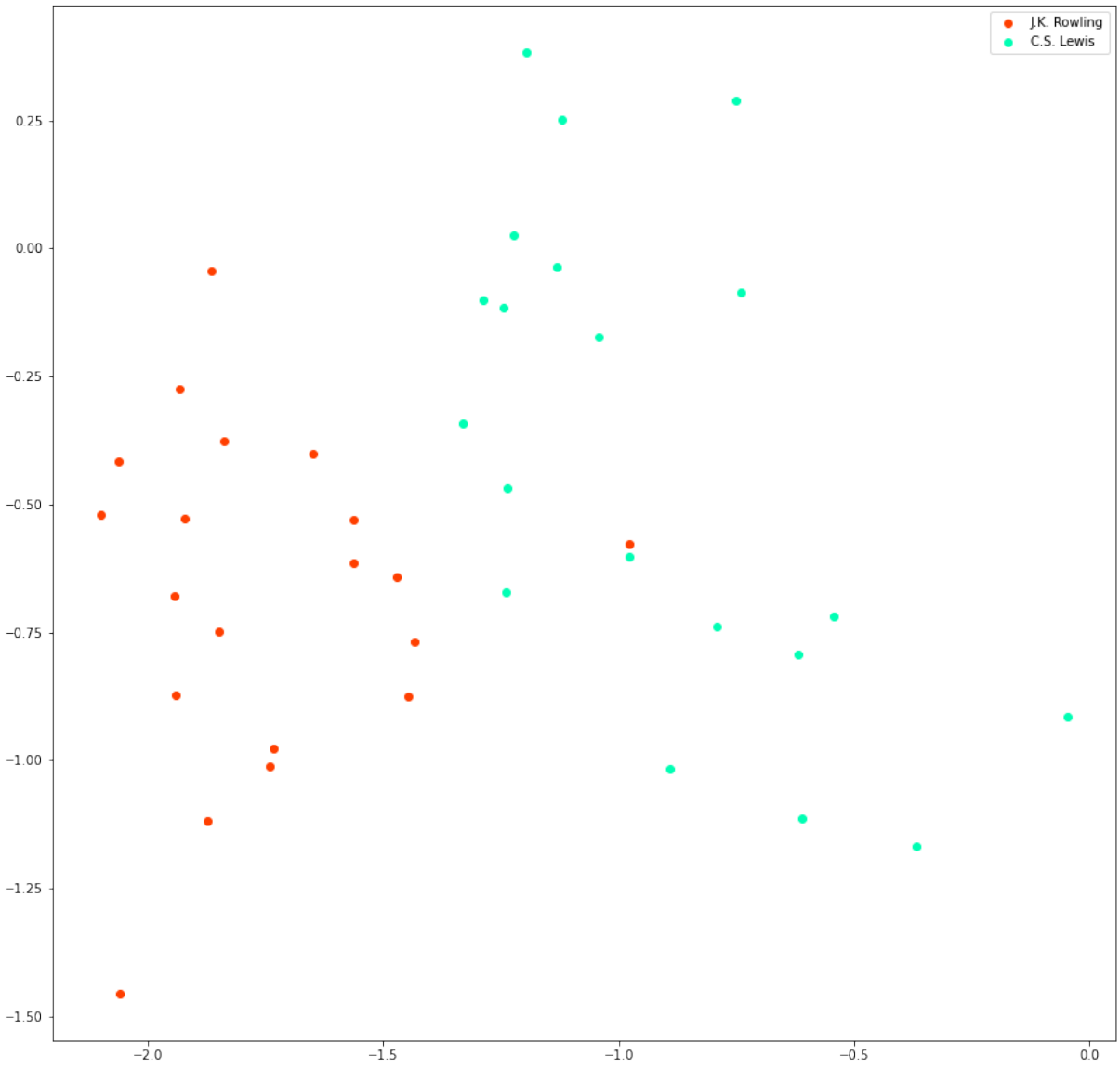

Figure 4.5 – Small dataset - J.K. Rowling and C.S. Lewis - two children's fantasy book writers

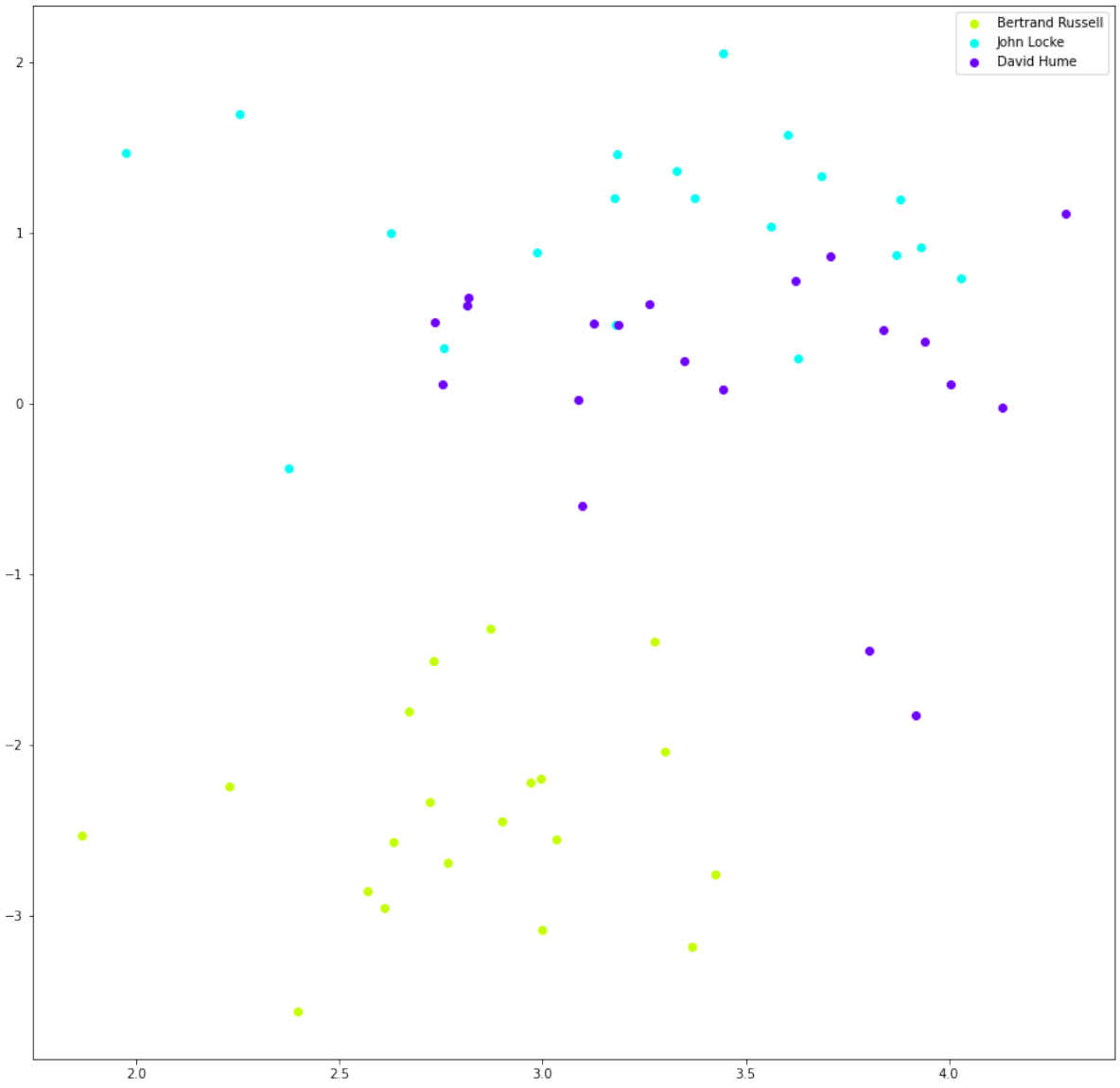

Figure 4.6 – Small dataset - John Locke, David Hume, and Bertrand Russell - three philosophers from different centuries

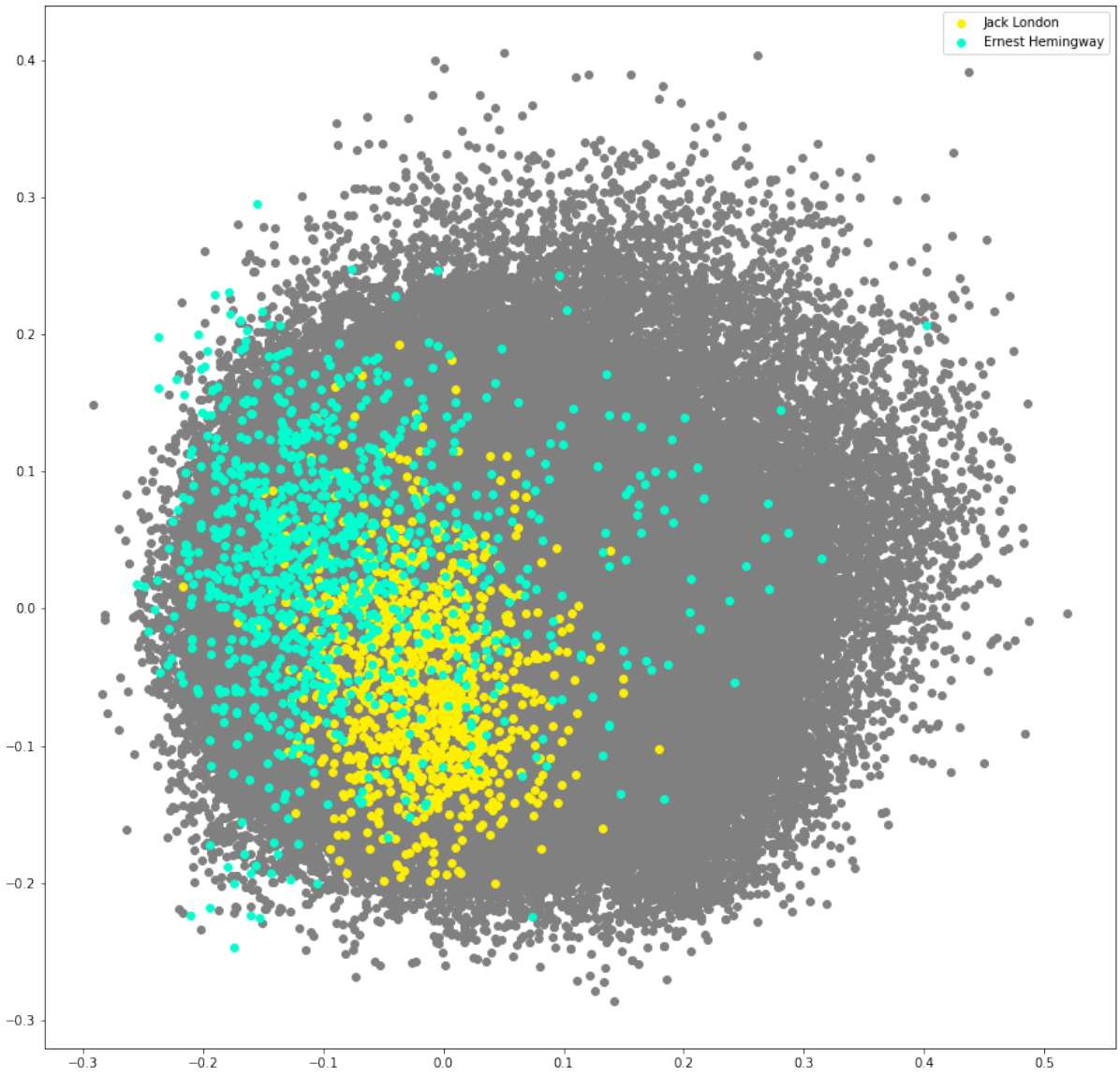

Figure 4.7 – Large dataset - Jack London and Ernest Hemingway - two American writers from the 19th - 20th centuries

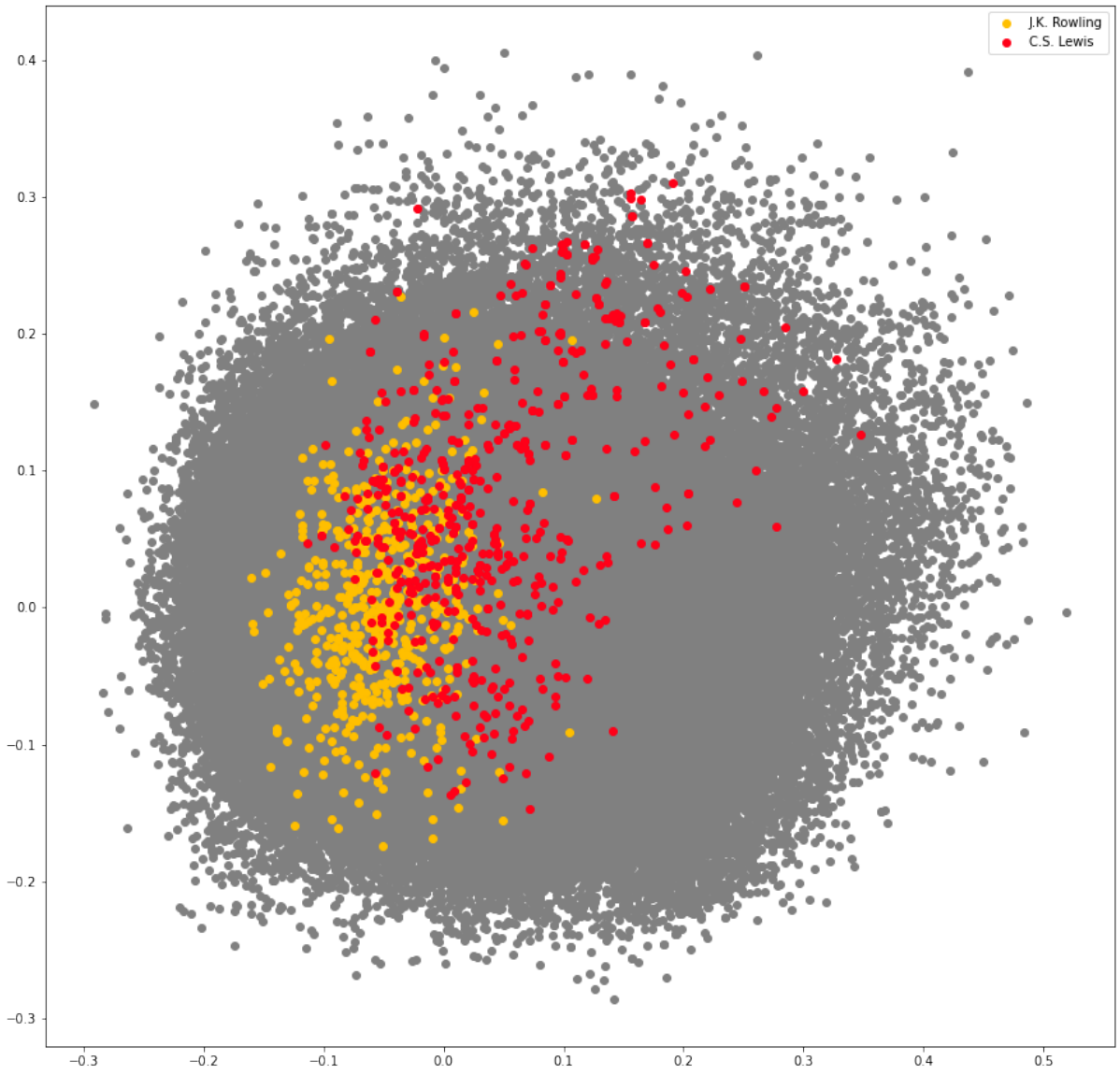

Figure 4.8 – Large dataset - J.K. Rowling and C.S. Lewis - two children's fantasy book writers

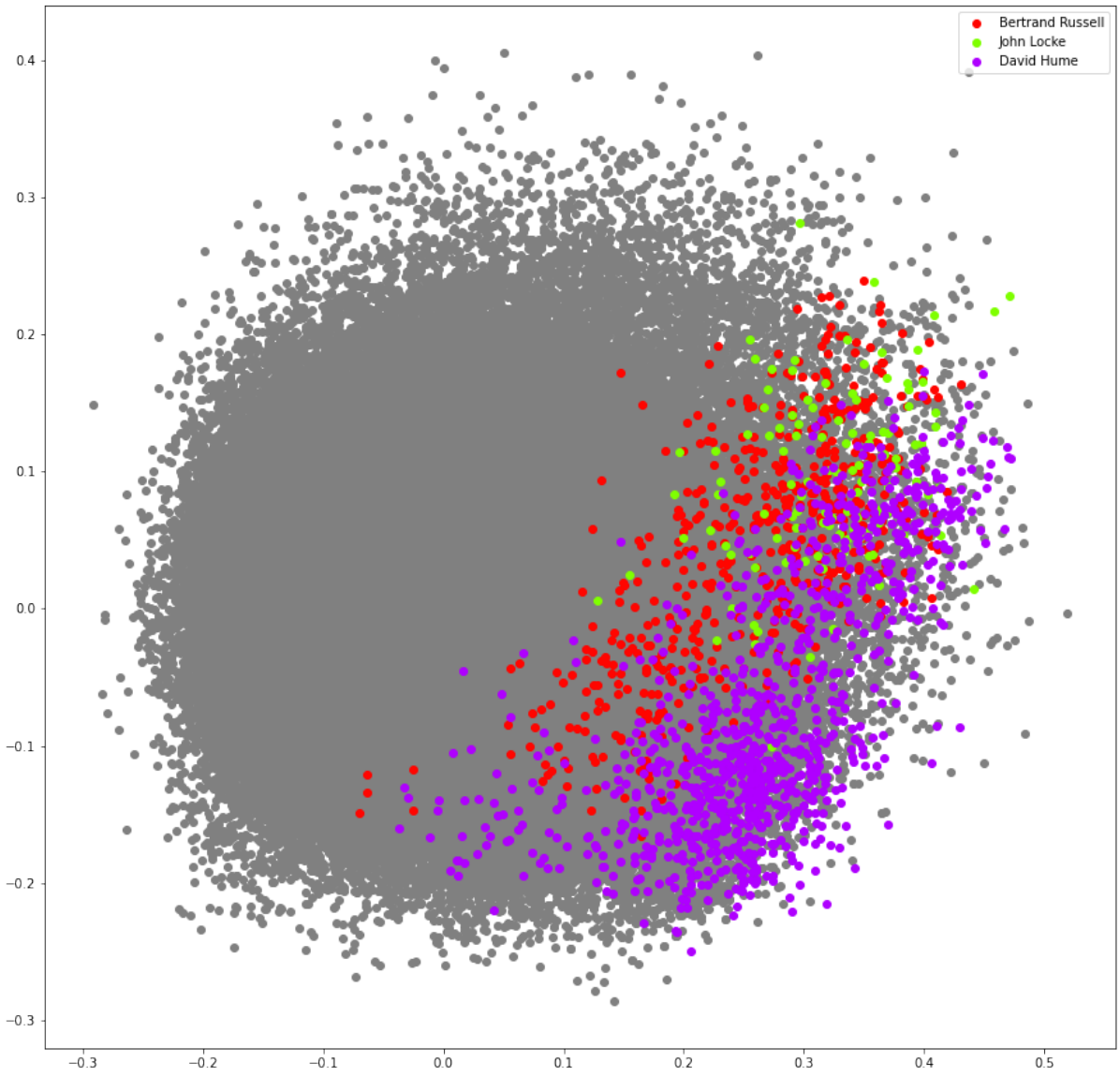

Figure 4.9 – Large dataset - John Locke, David Hume, and Bertrand Russell - three philosophers from different centuries

Chapter 5

Discussion

In this chapter the results are discussed with respect to each of the research questions, followed by some general notes.

What features will a neural network be able to use most effectively to perform author attribution?

We see that the LSTM networks performed poorly compared to the feedforward (Tables 4.1 - 4.3). This is unexpected seeing that in the literature there are two studies (Jasper et al., 2018; Schaetti and Savoy, 2020) indicating high performance using this method and word embeddings as input. In the future it is worth investigating further why this is the case and trying to reproduce their experiments on the same dataset they used. However, from a stylistics point of view it is not surprising that word embeddings as input produce poor results, as they are not designed to capture stylistic information.

For this study the more interesting analysis is of the results of the feedforward networks. For the feature sets in the small dataset, we see in Table 4.1 that Top 10 functional words already achieves quite good results, but Top 50 brings this accuracy up a level (thus Top 10 was left out of the large dataset experiments). It is interesting that Top 10 + deps or Top 10 + pos-all achieves on par with Top 50, meaning that the 40 words added to the functional words list also adds syntactic information.

It is also interesting that the Pos-verb set already achieves 40% accuracy. Verbs contain more stylistic information than nouns which mostly contain topical information, and hence that we can achieve this performance using information

from verbs only is worth noting.

Deps and pos-all seem to be relatively interchangeable - by themselves they achieve similar accuracy, and also leaving one of them out does not make such a big difference (compare Top 10 + pos-all + deps to Top 10 + pos-all or Top 10 + deps) but leaving both out brings the accuracy down by a lot (Top 10). However, this difference is less pronounced when using the Top 50 as a base. This is likely due to, as mentioned before, that the added functional words also add syntactic information.

For the large dataset, here in Table 4.3 we see that the accuracies have dropped compared to the small dataset however it can still be considered good seeing that there are over 300 authors and at random the accuracy would be magnitudes lower. We see that adding dependency and POS tag information brings the accuracy up, and that it is valuable to have both. This is interesting since they are both grammatical information modules, just with a different paradigm each. It might be so that one paradigm does not give all the information needed for author attribution. For instance, with POS tags one can know if a word is a noun, but unlike with dependency relations it is unable to tell if it is a subject or an object.

Increasing the number of functional words up to 100 increased the accuracy again, this time requiring a larger network. However, we see that the network size effect on accuracy reaches a limit as the largest network is hardly better than the second largest, for Top 100 + deps + pos.

The addition of POS 5-grams did not add much to the performance. This could possibly be due to the sparsity of the 5-grams, or it could be that the ordering of words do not play a major role in author attribution. Gamon (2004) achieved good performance using POS 3-grams and the choice of n in n-grams here might make a difference although it is not comparable due to their study using only three authors. More research is needed to investigate the effect of different choices of n, and perhaps how the POS n-grams perform on their own.

Looking at the PCA plots of the smaller dataset (Figures 4.1 - 4.6), we can see that authors which one would judge to be similar (similar time period, English or American, philosophers, children's writers) can be separated, sometimes completely using just the two principal components. In the plot with all 25 authors the colors are not easily distinguished however it is visible that the

philosophers are positioned distinctly and so are the poets. From this it seems that the model does preserve similarity between authors. The other writers are more difficult to distinguish but one can see that they do form clusters. It is important to keep in mind that the plot is just using two principal components and the original embeddings are of dimension 96, hence explaining the high accuracies obtained although the plot seems to be cluttered.

The plots of the larger dataset (Figures 4.7 - 4.9) show that similar authors are now not linearly separable as was often the case in the smaller dataset. Still, one sees that samples from the authors compared do have a different centroid and distribution and form clusters. It is worth noting that the two plots for novelists show the samples occupying a similar space on the left side, while the plot with the philosophers show that they occupy a completely different space on the right side.

How does the number of candidate authors affect author attribution?

The small dataset reached higher accuracies than the larger (Table 4.1 and 4.3). This is expected since when the number of candidate authors grow, it is also more likely that an author is similar to one of the others.

It is difficult to compare the numbers of performance in this study to previous studies, as the datasets used and thus number of candidate authors varies greatly. As mentioned in Luyckx and Daelemans (2008), author attribution studies with two, three, or a few authors has yielded accuracies over 95%. In the author attribution part of their own study using lexical and syntactic features and a variant of SVM, the 2-author task achieved an average accuracy of 96.90% while in 20-author and 145-author the accuracy decreased to 76% and 34% respectively. The results from this thesis showed that 69% accuracy could be achieved on a 316 author dataset. This is an indication that some combination of our general methodology, our feature generation, and our dataset outperform previous studies.

How does using embeddings for author attribution compare to using the raw feature vectors used as input to the neural network?

This study contributed to the field of stylometry by utilizing neural network trained embeddings as a way of performing author attribution. Many previous studies had used the raw features of word or syntactic tag frequencies to

perform attribution, e.g. with SVMs. This study shows that using a neural network to transform raw features into embeddings can increase performance. For example, for the best model of the larger dataset using embeddings increased performance by 13% (Table 4.3). This is in line with the feature engineering approach used by many other machine learning fields. It has been observed that 2 layers of feedforward network is already enough to generate the embeddings and that a deeper network actually decreases the performance.

Interesting to note from the larger dataset is that networks using a sigmoid activation function performed better than ones without. It stands in contrast to the smaller dataset where adding a sigmoid function actually decreased performance. The sigmoid function adds non-linearity to the transformation of raw features into embeddings. An explanation could be that with the larger number of authors and also larger number of fragments per author, the data is not easily linearly separable. This is also seen in the plots of the larger dataset as mentioned above. It shows one example of how methods used for author attribution depends on the dataset and one method which works for one dataset might not work for another.

How does a model trained for author attribution with triplet loss perform on an author style similarity task?

For the user testing of author similarity, the validity of the method can be questioned. We are trying to measure how well the models that perform more or less well on author attribution, perform on author similarity. For this purpose we are using human evaluation from a user test as the ground truth but this is far from objective. In fact, the evaluation of how well a model performs on author similarity can never be objective because there is no real ground truth. Having said this, the method applied in this study is one way to attempt this task for which there is no perfect solution.

The user test shows that there is not a big difference between how the two models performed in author similarity (Table 4.4), even though the difference between how they performed on author attribution is rather large (Table 4.3). As seen in Table 4.4, the model that performs worse in attribution, Top 50, in fact scores higher in author similarity in the cases of a higher threshold. Since higher threshold means that the human evaluation is more reliable, it could be said that the scores using higher threshold are also more correct. On the other hand, some triplets might just have a smaller difference between the

positive and the negative's distance to the anchor and this might be reflected in the answers of the user study. A 60% vs 40% outcome in one question might mean that the triplet was more difficult than one in which the outcome was 90% vs 10%, but that makes the answer no less valid. Also, using a higher threshold the number of questions available is reduced and that also reduces reliability.

In general, the user study shows that both models indeed perform better than random and hence it could be said that they do manage to generate relevant author style similarities, especially when looking at the scores with the highest threshold. Using an 80% threshold, the Top 50 model gets almost always the same answer as the humans do when judging which text is similar to the anchor. It could be possible that a model which performs worse in attribution can generate better similarities due to the embedding clusters being more spread out, but to say this without speculation needs a more extensive user test. Finally, it is needed to note that the size of the user study was very small and the validity of the results are affected by this.

Also, as mentioned above for the plots, it seems that similar authors (e.g. novelists) occupy a similar space and authors with more distinct writing style such as philosophers and poets occupy another space. This also indicates that the model is able to preserve similarity in addition to performing the author attribution task.

General

The impact of translated work in the large dataset should be considered, as translated work often mirror the style of the translator as well as or instead of the style of the original author. Moreover, when extracting books to be included in the large dataset only duplicates where the title matched exactly were filtered out, meaning that sometimes books could appear more than once if included in a collection.

The experiments reported here have a number of hyperparameters, the variation space of which have not been exhaustively explored. Potential features remain to be introduced into the process, and the experiment would benefit from being extended to further datasets. Really ambitious researchers might attempt to address some of the aspects brought up in the limitations of stylometry chapter, to see what are the boundaries of what a computer can achieve.

Chapter 6

Conclusion

Overall, the results from this study support the previous studies' outcome that syntactic modules increase the performance of author attribution models that only use lexical features. Nevertheless, lexical features prove to be very powerful and future studies could try to increase the number of functional words even further. Future studies could also put more focus on using linguistics to actually explain the phenomenon observed with the functional words and syntactic modules.

The number of candidate authors affect attribution methods in that the higher the number of candidates, the more difficult it is to tell them apart and hence perform attribution. The results from the smaller and larger dataset reflect this. Over 90% accuracy was achieved with the smaller dataset of 25 authors. However, it is worth noting that a 69% accuracy on the larger set of 316 authors is still impressive as it is far larger than random.

The use of neural networks to transform raw features into embeddings proves to be useful as it increased the performance, e.g. by 13% in the best model of the larger dataset.

This study also made an attempt to evaluate whether models that perform well in author attribution also perform well in author similarity. Although the methodology is not a perfect attempt and the user test was very small at scale, it generated some results that are interesting and that future work can build on with e.g. larger user tests or different methodology. The results indicate that the models trained on author attribution with triplet loss are also able to perform better than random on an author style similarity task that compares

the model to human evaluation. In some cases the model performed almost exactly as humans would judge style similarity.

It is worth remembering that this is a difficult task, as author style can be said to be rather intangible and is intuitively felt rather than computationally analyzed by humans while reading. It is interesting that a quantitative method using low level language features can achieve distinctive power, which is where the current state of stylometry research is at. However it should be kept in mind that many deeper stylistic patterns are undetected, providing much room for further exploration.

In addition to the above academic contribution, the conclusions of this study indicate that there is much potential for Storytel AB to utilize the work, to build upon it and generate more customer value with an innovative recommender system where author style is incorporated. Recommendations and ideas for moving forward are found in Appendix C.

References

Abbasi, A. and Chen, H.-c. (2005), 'Chen, h.: Applying authorship analysis to extremist-group web forum messages. ieee intelligent systems 20(5), 67-75', *Intelligent Systems, IEEE* **20**, 67 – 75.

Alharthi, H., Inkpen, D. and Szpakowicz, S. (2018*a*), Authorship identification for literary book recommendations, *in* 'Proceedings of the 27th International Conference on Computational Linguistics', Association for Computational Linguistics, pp. 390–400.

Alharthi, H., Inkpen, D. and Szpakowicz, S. (2018*b*), 'A survey of book recommender systems', *Journal of Intelligent Information Systems* **51**.

Argamon-Engelson, S., Koppel, M. and Avneri, G. (1998), Style-based text categorization: What newspaper am I reading?, *in* 'Proceedings of the AAAI Workshop on Text Categorization', pp. 1–4.

Argamon, S. and Shlomo, L. (2005), Measuring the usefulness of function words for authorship attribution.

Argamon, S., Šarić, M. and Stein, S. S. (2003), Style mining of electronic messages for multiple authorship discrimination: First results, *in* 'Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', KDD '03, Association for Computing Machinery, p. 475–480.

Argamon, S., Whitelaw, C., Chase, P., Hota, S. R., Garg, N. and Levitan, S. (2007), 'Stylistic text classification using functional lexical features', *Journal of the American Society for Information Science and Technology* **58**(6), 802–822.

Austen, J. (1813), *Pride and Prejudice*, Project Gutenburg. Retrieved: 2021-05-24.
URL: https://www.gutenberg.org/ebooks/1342

Baayen, H., van Halteren, H. and Tweedie, F. (1996), 'Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution', *Literary and Linguistic Computing* **11**(3), 121–131.

Barkan, O. and Koenigstein, N. (2016), 'Item2vec: Neural item embedding for collaborative filtering', *CoRR* **abs/1603.04259**.
URL: *http://arxiv.org/abs/1603.04259*

Bernhardsson, E. (2021), 'Annoy (approximate nearest neighbors oh yeah)', https://github.com/spotify/annoy. Accessed: 2021-04-20.

Burrows, J. (2002), ''Delta': a Measure of Stylistic Difference and a Guide to Likely Authorship', *Literary and Linguistic Computing* **17**(3), 267–287.
URL: *https://doi.org/10.1093/llc/17.3.267*

Burrows, J. F. (1987), 'Word-Patterns and Story-Shapes: The Statistical Analysis of Narrative Style', *Literary and Linguistic Computing* **2**(2), 61–70.

Burrows, J. F. (1992), 'Not Unles You Ask Nicely: The Interpretative Nexus Between Analysis and Information', *Literary and Linguistic Computing* **7**(2), 91–109.

Chaski, C. (2005), 'Who's at the keyboard? authorship attribution in digital evidence investigations.', *IJDE* **4**.

Christopher, B. (2016), 'How statistics solved a 175-year-old mystery about alexander hamilton', https://priceonomics.com/how-statistics-solved-a-175-year-old-mystery-about/. Accessed: 2021-02-20.

Crystal, D. (2010), *A Little Book of Language*, Yale University Press.

Diederich, J., Kindermann, J., Leopold, E. and Paass, G. (2003), 'Authorship Attribution with Support Vector Machines', *Applied Intelligence* **19**, 109–123.

Ding, S., Fung, B., Iqbal, F. and Cheung, K.-W. (2016), 'Learning stylometric representations for authorship analysis', *IEEE Transactions on Cybernetics* **PP**.

Dostoyevsky, F. (1880), *The Brothers Karamazov*, Project Gutenburg. Retrieved: 2021-05-24.
URL: *https://www.gutenberg.org/ebooks/28054*

Fontaine, J. and Stavick, J. (2004), 'Like nobody else: the secrets of metaphorical style', *DELTA: Documentação de Estudos em Lingüística Teórica e Aplicada* **20**.

Frantzeskou, G., Stamatatos, E., Gritzalis, S. and Katsikas, S. (2006), Effective identification of source code authors using byte-level information, *in* 'Proceedings of the 28th International Conference on Software Engineering', ICSE '06, Association for Computing Machinery, p. 893–896.

Gajawada, S. K. (2019), 'Chi-square test for feature selection in machine learning', https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223s. Accessed: 2021-03-20.

Gamon, M. (2004), Linguistic correlates of style: authorship classification with deep linguistic analysis features, *in* 'COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics', COLING, Geneva, Switzerland, pp. 611–617.
URL: *https://www.aclweb.org/anthology/C04-1088*

Goodfellow, I., Bengio, Y. and Courville, A. (2016), *Deep Learning*, MIT Press. http://www.deeplearningbook.org.

Grant, T. (2007), 'Quantifying evidence in forensic authorship analysis', *International Journal of Speech, Language and the Law* **14**(1), 1–25.

Hermans, A., Beyer, L. and Leibe, B. (2017), 'In defense of the triplet loss for person re-identification', *CoRR* **abs/1703.07737**.
URL: *http://arxiv.org/abs/1703.07737*

Hochreiter, S. and Schmidhuber, J. (1997), 'Long short-term memory', *Neural computation* **9**, 1735–80.

Hollingsworth, C. (2012), Using dependency-based annotations for authorship identification, *in* P. Sojka, A. Horák, I. Kopeček and K. Pala, eds, 'Text, Speech and Dialogue', Springer Berlin Heidelberg, pp. 314–319.

Holmes, D. I. (1994), 'Authorship attribution', *Computers and the Humanities* **28**, 87–106.

Hoorn, J., Frank, S., Kowalczyk, W. and van der Ham, F. (1999), 'Neural network identification of poets using letter sequences', *Literary and Linguistic Computing* **14**(3), 311–338.

Hoover, D. (2004), 'Testing burrows' delta', *Literary and Linguistic Computing* **19**, 453–475.

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013), *An Introduction to Statistical Learning*, Springer.

Jasper, J., Berger, P., Hennig, P. and Meinel, C. (2018), *Authorship Verification on Short Text Samples Using Stylometric Embeddings: 7th International Conference, AIST 2018, Moscow, Russia, July 5–7, 2018, Revised Selected Papers*, pp. 64–75.

Jockers, M. L. and Witten, D. M. (2010), 'A comparative study of machine learning methods for authorship attribution', *Literary and Linguistic Computing* **25**(2), 215–223.

Karlgren, J. and Cutting, D. (1994), Recognizing text genres with simple metrics using discriminant analysis, *in* 'Proceedings of the 15th Conference on Computational Linguistics - Volume 2', COLING '94, Association for Computational Linguistics, p. 1071–1075.

Karlgren, J. and Eriksson, G. (2007), Authors, genre, and linguistic convention, *in* 'Proceedings of the SIGIR Workshop on Plagiarism Analysis, Authorship Attribution', pp. 23–28.

Kiela, D. and Bottou, L. (2014), Learning image embeddings using convolutional neural networks for improved multi-modal semantics, *in* 'Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)', Association for Computational Linguistics, Doha, Qatar, pp. 36–45.
URL: *https://www.aclweb.org/anthology/D14-1005*

Kilgarriff, A. (1996), 'Bnc database and word frequency lists', `http://www.kilgarriff.co.uk/bnc-readme.html`. Accessed: 2021-03-20.

Kocher, M. and Savoy, J. (2017), 'Distributed language representation for authorship attribution', *Digital Scholarship in the Humanities* **33**(2), 425–441.

Koppel, M. and Schler, J. (2003), 'Exploiting stylistic idiosyncrasies for authorship attribution'.

Koppel, M., Schler, J. and Argamon, S. (2011), 'Authorship attribution in the wild', *Language Resources and Evaluation* **45**, 83–94.

Koppel, M., Schler, J., Argamon, S. and Messeri, E. (2006), Authorship attribution with thousands of candidate authors, pp. 659–660.

Kukushkina, O. V., Polikarpov, A. A. and Khmelev, D. V. (2001), 'Using literal and grammatical statistics for authorship attribution', *Problems of Information Transmission* **37**, 172–184.

Le, Q. V. and Mikolov, T. (2014), 'Distributed representations of sentences and documents', *CoRR* **abs/1405.4053**.
URL: *http://arxiv.org/abs/1405.4053*

Lops, P., de Gemmis, M. and Semeraro, G. (2011), *Content-based Recommender Systems: State of the Art and Trends*, pp. 73–105.

Luyckx, K. and Daelemans, W. (2008), *Authorship Attribution and Verification with Many Authors and Limited Data*, Vol. 1.

Madigan, D., Genkin, A., Lewis, D. D., Argamon, S., Fradkin, D. and Ye, L. (2005), Author identification on the large scale, *in* 'Proceedings of CSNA-05'.

Manning, C. D., Raghavan, P. and Schütze, H. (2008), *Introduction to information retrieval*, Cambridge University Press.

Matthews, P. H. (1981), *Syntax*, Cambridge University Press.

Matthews, R. and Merriam, T. (1993), 'Neural computation in stylometry i: An application to the works of shakespeare and fletcher', *Literary and Linguistic Computing* **8**.

Mendenhall, T. C. (1887), 'The characteristic curves of composition', *Science* **ns-9**(214S), 237–246.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. and Dean, J. (2013), 'Distributed representations of words and phrases and their compositionality', *CoRR* **abs/1310.4546**.
URL: *http://arxiv.org/abs/1310.4546*

Moindrot, O. (2018), 'Triplet loss and online triplet mining in tensorflow', https://omoindrot.github.io/triplet-loss. Accessed: 2021-02-20.

Mosteller, F. and Wallace, D. L. (1964), *Inferencce and Disputed Authorship: The Federalist*, Addison-Wesley.

Olah, C. (2015), 'Understanding lstm networks', https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Accessed: 2021-02-20.

Panicheva, P., Litvinova, O. and Litvinova, T. (2019), Author clustering with and without topical features, *in* A. A. Salah, A. Karpov and R. Potapova, eds, 'Speech and Computer', Springer International Publishing, Cham, pp. 348–358.

Pathak, D., Matharia, S. and Murthy, C. (2013), Nova: Hybrid book recommendation engine, pp. 977–982.

Pazzani, M. J. and Billsus, D. (2007), *Content-Based Recommendation Systems*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 325–341.

Rajpurkar, S. and Bhatt, D. (2015), 'Book recommendation system', *–International Journal for Innovative Research in Science Technology* **1**(11).

Rudman, J. (1997), 'The state of authorship attribution studies: Some problems and solutions', *Computers and the Humanities* **31**(4), 351–365.

Savoy, J. (2020), *Machine Learning Methods for Stylometry*, Springer Nature Switzerland.

Schaetti, N. and Savoy, J. (2020), 'Comparison of visualisable evidence-based authorship attribution methods using recurrent neural networks'.

Schroff, F., Kalenichenko, D. and Philbin, J. (2015), 'Facenet: A unified embedding for face recognition and clustering', *CoRR* **abs/1503.03832**. **URL:** *http://arxiv.org/abs/1503.03832*

Scikit-learn (2020), 'Nearest neghbors', https://scikit-learn.org/stable/modules/neighbors.html. Accessed: 2021-03-20.

Sebastiani, F. (2002), 'Machine learning in automated text categorization', *ACM Comput. Surv.* **34**(1), 1–47.

Shani, G. and Gunawardana, A. (2011), *Evaluating Recommendation Systems*, Springer US, pp. 257–297.

Shrestha, P., Sierra, S., González, F., Montes, M., Rosso, P. and Solorio, T. (2017), Convolutional neural networks for authorship attribution of short texts, *in* 'Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers', Association for Computational Linguistics, pp. 669–674.

spaCy (2021), 'English pipelines', https://spacy.io/models/en. Accessed: 2021-02-20.

Stamatatos, E. (2009), 'A survey of modern authorship attribution methods', *Journal of the American Society for Information Science and Technology* **60**(3), 538–556.

Stamatatos, E. (2018), 'Masking topic-related information to enhance authorship attribution', *Journal of the Association for Information Science and Technology* **69**(3), 461–473.

Steen, G. (2014), Metaphor and style, *in* P. Stockwell and S. Whiteley, eds, 'The Cambridge Handbook of Stylistics', Cambridge University Press, chapter 21, pp. 315–328.

Stubbs, M. (2014), Quantitative methods in literary linguistics, *in* P. Stockwell and S. Whiteley, eds, 'The Cambridge Handbook of Stylistics', Cambridge University Press, chapter 4, pp. 46–62.

Su, X. and Khoshgoftaar, T. M. (2009), 'A survey of collaborative filtering techniques', *Advances in Artificial Intelligence* .

Tweedie, F. J., Singh, S. and Holmes, D. I. (1996), 'Neural network applications in stylometry: The Federalist Papers', *Computers and the Humanities* **30**, 1–10.

Vaz, P. C., Martins de Matos, D. and Martins, B. (2012), Stylometric relevance-feedback towards a hybrid book recommendation algorithm, *in* 'Proceedings of the Fifth ACM Workshop on Research Advances in Large Digital Book Repositories and Complementary Media', BooksOnline '12, Association for Computing Machinery, p. 13–16.

Vaz, P., Ribeiro, R. and de Matos, D. M. (2013), Book recommender prototype based on author's writing style, *in* 'OAIR'.

Verne, J. (1873), *Around the World in Eighty Days*, Project Gutenburg. Retrieved: 2021-05-24.
URL: https://www.gutenberg.org/ebooks/103

Zhao, Y. and Zobel, J. (2005), Effective and scalable authorship attribution using function words, *in* G. G. Lee, A. Yamada, H. Meng and S. H. Myaeng, eds, 'Information Retrieval Technology', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 174–189.

Zhao, Y. and Zobel, J. (2007), Searching with style: Authorship attribution in classic literature., Vol. 62, pp. 59–68.

Zheng, Y., Agnani, M. and Singh, M. (2017), Identifying grey sheep users by the distribution of user similarities in collaborative filtering, *in* 'Proceedings of the 6th Annual Conference on Research in Information Technology', RIIT '17, Association for Computing Machinery, p. 1–6.

Appendix A

Books used in the small dataset

Table A.1 – Author and title of books used in the small dataset

Author	Title
Douglas Adams	Long Dark Tea-Time of the Soul
J.K. Rowling	Harry Potter and the Prisoner of Azkaban
Charles Dickens	Oliver Twist
Jane Austen	Emma
Jack London	Children of the Frost
Bertrand Russell	Mysticism and Logic and Other Essays
William Shakespeare	Julius Caesar
William Shakespeare	A Midsummer Night's Dream
William Shakespeare	Shakespeare's Sonnets
Ernest Hemingway	Farewell to Arms
George Orwell	1984
Mark Twain	Adventures of Huckleberry Finn
Virginia Woolf	Orlando
C.S. Lewis	The Chronicles of Narnia - Complete 7 Books in One Edition
John Locke	An Essay Concerning Human Understanding
Chris Hedges	American Fascists: The Christian Right and the War On America
Collin Wilcox	The Lonely Hunter
Conan Doyle	The Hound of the Baskervilles
Cornelius Ryan	The Longest Day: The Classic Epic of D-Day
Dale Carnegie	How to Win Friends and Influence People
David Hume	A Treatise of Human Nature
Diamond Johnson	Little Miami Girl
Ed McBain	The Spiked Heel
Edgar Allan Poe	Tales of the Grotesque and Arabesque
Geoffrey Chaucer	The Canterbury Tales
Mary MacGregor	The Story of Greece
Napoleon Hill	Think and Grow Rich!

Appendix B

Feature modules

Table B.1 – Top 100 functional words

1-20	21-40	41-60	61-80	81-100
the	this	all	think	only
be	but	get	my	new
of	from	her	come	very
and	they	make	than	when
a	his	who	more	may
in	she	out	about	way
to	or	up	now	look
have	which	see	last	like
it	as	know	your	use
for	we	time	me	such
i	an	take	no	how
that	say	them	other	because
you	will	some	give	good
he	would	could	just	find
on	can	so	should	man
with	if	him	these	our
do	their	year	people	want
at	go	into	also	day
by	what	its	well	between
not	there	then	any	even

Table B.2 – Dependency relations from spaCy

acl	acomp	advcl	advmod
agent	amod	appos	attr
aux	auxpass	case	cc
ccomp	compound	conj	csubj
csubjpass	dative	dep	det
dobj	expl	intj	mark
meta	neg	nmod	npadvmod
nsubj	nsubjpass	nummod	oprd
parataxis	pcomp	pobj	poss
preconj	predet	prep	prt
quantmod	relcl	xcomp	

Table B.3 – POS tags from spaCy

ADD	AFX	CC	CD
DT	EX	FW	HYPH
IN	JJ	JJR	JJS
LS	MD	NFP	NN
NNP	NNPS	NNS	PDT
POS	PRR	PRP$	RB
RBR	RBS	RP	SYM
TO	UH	VB	VBD
VBG	VBN	VBP	VBZ
WDT	WP	WP$	WRB
XX			

Appendix C

Roadmap to application

The following steps are recommendations of how to use the work in this thesis and of further research in an applied setting.

A/B testing of recommendations incorporating style

This is the natural next step as a small user study has been conducted in this thesis yielding results with potential.

Investigate the partial overlap/similarity between authors and books

An experiment that was not done due to lack of time but nevertheless would be very interesting is to investigate the partial overlaps between different authors or books. For example, taking two books, if we divide the book texts into a large number of fragments and produce embeddings for them, it is probable that a proportion of the fragments from one book would overlap with a proportion of the fragments of the other book, but not all. What this could mean is that some parts of an author's work is similar in style with another author's work, even though the two are not similar as a whole. For instance, perhaps the two books both contain chapters describing character introspection, while the other chapters are completely different. Although not globally similar, this kind of partial similarity is still of interest to the reader, e.g. one who likes psychological novels.

Increase the number of functional words, add semantic features to the feature set

Taking the best model from the large dataset, one could experiment with even more functional words and also investigate semantic features, which is something that this thesis did not do. However, as mentioned in the discussion it is not yet conclusively established whether a model that performs better in author attribution actually generates better style similarity.

Mood embeddings

During the process of this thesis it was pondered upon whether style also incorporated some information about the mood of a book. However, in the end the theory was that style and mood are two different things, even though there may be correlations. For instance, during the user test for style similarity most users found the questions very difficult, likely indicating that humans are not intuitively very good at telling author styles apart from each other especially when our definition of style is a reductionist one based on frequencies of common words and syntactic structures. However, it should be easier for humans to tell the mood of a book from another and to find similarities. Hence, mood is probably also a stronger predictor of how much users would appreciate the recommendations.

The mood could possibly be found in verbs and adjectives, for example there are lists of verbs of motion, verbs of utterence, verbs of introspection, etc. and adjectives can be grouped together based on the mood. Using these as features and inspecting the similarities they generate may prove to be an interesting experiment.

Exploring more relationships between embeddings

Inspired by the Word2Vec ability to find relationships between embeddings such as: vec(“Madrid”) - vec(“Spain”) + vec(“France”) is closer to vec(“Paris”) than to any other word vector, we can experiment with stylometric embeddings and see if we also can find relationships. For example, we can check what is the embedding that is halfway between two books, and see if this book is a "hybrid". A reader would certainly like to find out which book is the style combination of two of their favorite books.

Customer value

Having style and possibly mood based recommendations may add to the value proposition of a streaming service. The features can potentially be toggled on and off giving users the choice of what kind of recommendations they want - the traditional recommendations based on user ratings, only style or mood based, or a combination. Whether or not the user actually finds new interesting books thanks to this recommender, the perceived value of the service may be increased just by the option of being able to receive recommendations based on style or mood.

TRITA -EECS-EX-2021:525