Identifying Author Fingerprints in Texts via Graph Neural Networks

Identifying Author Fingerprints in Texts via Graph Neural Networks

Thesis

to obtain the degree of Master in Computer Science with Specialization in Data Science and Technology at Delft University of Technology,
to be publicly defended on Monday, August 31st 2020 at 09:00

by

Tomas Sipko

Born in Vilnius, Lithuania

Multimedia Computing Group,
Faculty of Electrical Engineering,Mathematics and Computer Science (Faculteit Elektrotechniek, Wiskunde en Informatica),
Delft University of Technology,
Delft, The Netherlands.

Thesis committee:

Chair:	Dr. Pablo Cesar, Faculty EEMCS, TU Delft
Daily Supervisor:	Dr. Elvin Isufi, Faculty EEMCS, TU Delft
Committee Member:	Dr. Joana P. Gonçalves, Faculty EEMCS, TU Delft

Keywords: authorship attribution; graph neural networks; graph signal processing

Copyright © 2020 by T. Sipko

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

Contents

SUMMARY

The world is generating more and more network data in many different areas (e.g., sensor networks, social networks and even text). A unique characteristic of these data is the coupling between data values and underlying irregular structure on which these values are defined. Thus, researchers developed Graph Neural Networks (GNNs) to use deep learning approaches on these irregular network data. GNNs developers tried to replicate the recent success of Convolutional Neural Networks (CNNs) and developed its graph counterpart Graph Convolutional Neural Network (GCNN) and more different variations of GNNs (e.g. EdgeNet). However, all these architectures are relatively young, and the impact of different parameters to classification result is not well researched compared to regular neural network architectures. To address this issue, we propose to use authorship attribution problem to research the impact of different architectures and their variations to classification accuracy and how GNNs can be used to improve on authorship attribution task compared to the baseline architectures. Explicitly, we define the dataset which is going to be used throughout the experiments and the method to convert text excerpts of authors into the network that can be classified with GNNs (called WAN). WAN is as a network that captures unique author fingerprint. We also define the set of GNN architectures (and different combinations and variations of them), baseline architecture (SVM) and experiments that are used with those architectures. This experiment setting allows us to compare different GNN architectures among themselves and the baseline architecture. Also, we define a method to reduce the dimensions of author fingerprints (WANs) and use these sparse author fingerprints for the same experiments with the same architectures. Numerical results show the improvement over the baseline architectures in nearly all defined experiments. Also, we found that more complex GNN architectures (e.g. EdgeNets) are superior to shallower architectures with more laborious experiments (e.g. classification by gender). More complex architectures also require hyperparameter re-tuning in order to achieve optimal results. Furthermore, experiments with sparse author fingerprints showed that we could achieve comparable results to standard fingerprints with faster training times and significantly reduced dimensions. GNN architectures used with sparse author fingerprints were usually superior to baseline architectures.

Acknowledgements

Science is highly-complex, unnecessary for the survival, however fascinating endeavour. I want to thank every person that made this journey happen.

Firstly, my family for providing with the necessary resources. Master studies in TU Delft were purely my interest which only served my curiosity and provided yet another challenge. Unquestionable support from my parents helped to make the first steps in the foreign country, which I knew little of, shadowed distractions and allowed me to focus solely on why I came here.

Secondly, I would like to thank my supervisor - Elvin for helping to generate ideas and displaying genuine care and attention throughout this relatively long period of writing the thesis. I can be very forgetful and absent-minded; thus, your unsweetened style only helped me achieving this document and explore the peculiarities of science.

Finally, I would like to thank my friends who were around throughout studies. First of all - Norbertas for being a mentor while I was only starting my life in the Netherlands and Lukas who helped me in the hardest times. Rytis - for always being a true inspiration and a man who would push me to do what is best for me. Finally, the group that helped me to retain a balanced life and well-rounded personality even at the time of uncertainty - Goda, Ole, Robin, Thomas.

Tomas Sipko
Delft, January 2020

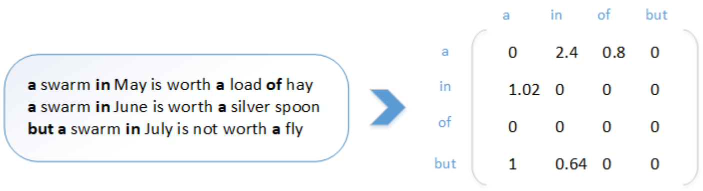
1 INTRODUCTION

Currently, people are producing more and more written text. Naturally, the need for text processing tools grows with the amount of text produced. Authorship attribution is one of the tasks related to a text which plays an important role in many applications such as plagiarism detection and forensic investigation [1]. Usually, approaches to solving this problem involve analysing author's unique writing style and/or topics they tend to write about.

Authorship attribution is a unique task standing out from a regular text classification. In some cases, authors can be categorized by the topic of their work, however, in a more homogeneous set of topics, it is necessary to capture the author's stylistic features [1]. In this work, we focus on stylistic features. In particular, we use function words to characterize the authors' styles. Function words on their own carry little meaning, however, they dictate the grammatical relations between words. It includes words like prepositions, conjugations and pronouns [2]. Research showed that the frequency of function words appearing in the text is discriminatory enough to classify different authors [2–4].

Since function words proved to be a powerful feature in authorship attribution, more complex methods based on function words were developed. In this work, we use the function Word Adjacency Networks (WAN) [2]. The key principle is to build a network of words, where the likelihood of two function words being close to each other in the text is encoded. The method turns a text excerpt into a graph, where the weight of edges depicts the probability of certain words being close to each other in the text (Figure 1.1). WANs can extract discriminatory author fingerprints from texts that can be further used to attribute texts to authors. This task translates into a graph classification problem [5]. However, many machine learning methods are unable to use the underlying structure of the graph from WAN for learning. Thus, in this work, we used Graph Neural Networks (GNNs) - an emerging tool for classifying data on irregular graph structures [6].

GNNs allows us to achieve better generalization while training since they can use the topology underlying the data. Inspired by recent success of regular Convolutional Neural Networks (CNNs) [7] and advancements in graph theory [8] researchers defined Convolutional Graph Neural Networks [9] and more complex architectures such as Ed-


a swarm **in** May is worth **a** load **of** hay
a swarm **in** June is worth **a** silver spoon
but a swarm **in** July is not worth **a** fly

$$\begin{array}{c|cccc} & a & in & of & but \\ \hline a & 0 & 2.4 & 0.8 & 0 \\ in & 1.02 & 0 & 0 & 0 \\ of & 0 & 0 & 0 & 0 \\ but & 1 & 0.64 & 0 & 0 \end{array}$$

Figure 1.1: Schematic illustration of converting a text excerpt to a network. On the left there is a text excerpt with function words in bold. On the right there as a matrix, where each entry is a likelihood of the function word being close to another function word (the values are arbitrary for presentation reasons).

geNet [10]. In essence, EdgeNet is a GNN architecture that differently from conventional approaches learns on the node level. Since the weights are not shared among the nodes it allows capturing more detail, however, results in higher complexity. We used GNNs together with WANs for authorship classification since other machine learning methods are unable to use of the underlying structure.

GNNs allow analyzing the importance of the underlying structure of the data and improve upon WAN author fingerprints. While using different GNN architectures and comparing them to machine learning methods that do not use the topology we can gain a better understanding of what is the impact of utilizing the topology. Also, with the help of EdgeNet, we are able to sparsify WAN fingerprints, therefore, identifying core author fingerprints as their stylistic signatures. This allows us to improve upon learning time while retaining similar classification errors and identify the words that play the biggest role in authorship attribution task.

The main research questions we focus on this works are:

RQ1. How different GNNs can be used to improve on the state-of-the-art of authorship attribution?

RQ2. How can we use authorship attribution problem to gain a better understanding of GNNs?

RQ3. How can we use sparse author fingerprints for learning and gaining a better understanding of what is being learned by GNNs?

To address these research questions we carry out an extensive set of experiments with the authorship attribution dataset with extracted WANs of 21 18th century authors. This allows us to use GNNs for authorship attribution problem and compare with the state-of-the-art. Also, we use a set of different GNN architectures with different hyperparameters to gain a better understanding of the capabilities of GNN architectures. We experimented with architectures different in complexity, depth, and hyperparameters. Besides, we evaluated how certain architectures deal with more difficult or easier tasks by running experiments that differ in complexity. This includes, for instance, classifying authors by gender, nationality and both gender and nationality. Finally, we used EdgeNet's capability to learn on node level and were able to learn new graphs from WANs. Then, we extracted sparse author fingerprints from learned graphs, visualized them, and compared their performance on authorship attribution to the performance while using normal fingerprints.

This thesis is structured as follows: in Chapter 2 we introduce related work on authorship attribution and GNNs. Chapter 3 discusses a necessary background material used in this thesis. Chapter 4 introduces the architectures and experiments, together with the method to extract sparse author fingerprints. Furthermore, Chapter 5 discusses the numerical results which were obtained from the experiments and Chapter 6 summarizes the work done in this thesis.

2 Related Work

In this chapter we review literature related to the current thesis. Firstly, we introduce the problem we are trying to improve - authorship attribution 2.1. Further, we review existing literature on the tool that we used to tackle the problem - Graph Neural Networks in Section 2.2.

2.1. Authorship attribution

Authorship attribution is a well-known task in a machine learning community. In essence, the task requires attributing an unknown text to a specific author [11]. In contrary to other classification problems, solving an authorship attribution task relies on selecting the right features for classification, since many authors have their writing style. Thus, selecting the right features for classification is crucial, and the focus of research usually shifts to exploring a new method of capturing stylistic features of authors, rather than using state of the art classification algorithms [11].

The core assumption made for solving an authorship attribution task is that each author has a unique and distinguishable writing style that is consistent over his or her publications is not entirely true, because authors can change their writing style for specific audiences and occasions. Thus, the complexity of the task is to find the correct way to capture consistent features of author writing.

We can group available research of features used for authorship attribution into those categories [12]:

- Stylistic features. Includes the features such as length of the sentences and words, the richness of vocabulary
- Syntactic cues. These methods look for syntactic structures in texts
- Word-based features. Methods based on the analysis of certain words used by authors

The first method has inconclusive results as these features usually differ between different works of an author and depends heavily on a genre of the text [12]. The second method is computationally expensive due to in-depth syntactic analysis of the document. The last option includes using function words as features and proved to be reliable indicators of authorial style [13]. Function words are words that express a grammatical relationship and are mostly devoid of contextual meaning. They include conjunctions such as "and" and "or," prepositions such as "in" and "at" quantifiers such as "some" and "all," modals such as "may" and "could" as well as determiners such as "the" and "that". The intuition behind the usefulness of function words for authorship attribution is the following. Due to their high frequency and highly grammaticalized role in the text, they are highly unlikely to be subject to conscious control of an author.

Besides, these words are a core part of the language and are present in various genres. This is handy and solves the problem of the first group of authorship attribution features because the usage of function words does not depend on a topic or type of the text. Also, simple statistics of the frequency of function words varies significantly across different authors [4]. This difference indicates these word-based features can to differentiate authors even without additional text processing.

However, function words themselves are highly reductionistic for gaining a deep insight into the author's stylistic features. Therefore, researchers shift to more complex textual features that make use of function words. One of those is building a Word Adjacency Network (WAN) which in essence is a network that captures the likelihood of function words being close to each other [2]. This line of work has proved itself being successful for capturing stylistic insights of authors. The authors have used simpler linear classifiers, such as SVM in order to test their method.

Although authors achieved considerably good performance using the WAN method with linear classifiers, we see an opportunity to utilize Graph Neural Networks for the task. WAN method is based on building a graph from the text excerpt, which can be further used for classification using Graph Neural Networks.

2.2. GNN STATE-OF-THE-ART

The ability to use neural networks on irregular data received increased interest over the past years. Most of these data are represented on graphs; thus, GNNs were developed [14–16]. With recent advances in graph signal processing, we can define Fourier transform [8] and convolution operations on graphs [16], which can be used in the Neural Network setting. These advances allow us to use convolution operation on data, that is on top of the graph, while still keeping interconnections and making use of generalization and local feature capturing that Convolutional Neural Networks (CNNs) were superior recently [17]. Equipped with the definition of the graph convolution researchers can take inspiration from the recent success of CNNs and experiment with its graph counterparts.

In this thesis, we experiment with various types of GNNs. They utilize the graph in different ways leading to different computational complexity. Therefore, in this thesis, we extensively use Graph Convolutional Neural Networks (GCNNs) [9] as a baseline model for author attribution problem. The GCNN we use contains properties of permutation equivariance [18], meaning that is can exploit internal symmetries of underlying topology and stability to perturbations, which yields that the architecture is stable to small changes in underlying topology. Compared with other more advanced methods (EdgeNets) it can generalize relatively well with fewer parameters, leading to faster learning times. In literature, there are many definitions and implementations of GCNNs [19]. In this work, we rely on the GCNN defined in [9].

Furthermore, we explore more advanced GNN implementations such as EdgeNets [10]. Despite the certain differences in GCNNs, they share the same feature of reusing the coefficients across all neighbourhoods. In contrary, EdgeNet offers more flexibility and can learn coefficients on a graph node level. Thus, it captures more detail and can be used to learn graphs. Also, with a certain configuration of EdgeNet, it is possible to learn a new graph from the input graph, since EdgeNet learns coefficients on a node level.

Also, the ability to capture the topology of the data using GNNs intrinsically leads to improved explainability of black-box machine learning models. Since learning is on the graph, the explainability methods can exploit underlying topology for generating the explanation of the decision by GNN model. Authors of GNNExplainer [20] were focusing on identifying one or more parts of the graph whose influence on the decision was the highest. Given an input graph, the method yields the sub-graph of the input graph and a subset of node-level features that are relevant for a particular prediction. Reduced input graph and feature space allow for a better human interpretation. GNNExplainer belongs to model-agnostic explanation methods group, meaning it can provide explanations for any graph-based machine learning task.

Another line of work would focus on generating the subset of node-level features that best explains the decision, rather than generating graph structures. GraphLIME [21] is a method that was inspired by the recent success of LIME model [22]. LIME is another model-agnostic tool which uses a linear model to find features that explain the decision of complex black-box model. However, LIME is based on calculating the proximity between the point to be explained and other points around. Thus, the authors of GraphLIME adjusted the linear model to work in a graph setting for successful node-level feature extraction.

3

Background

In this chapter, we introduce the research on which we base this thesis. Firstly, to explain WAN extraction method of GNNs for authorship attribution problem, we are going to introduce, and the notion of the graph, graph signal in Section 3.1 and graph neural networks in Section 3.2. Then, we are going to introduce WANs and authorship attribution dataset in Section 3.4.

3.1. ALGEBRAIC REPRESENTATIONS OF A GRAPH

A graph is defined as a tuple $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, ..., v_N\}$ is a set of nodes with cardinality $|\mathcal{V}| = N$, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ with cardinality $|\mathcal{E}| = M$ is a set of edges containing all the tuples of nodes, where there is an interconnection. The graph can be directed, if the edges has a directions, meaning that if there is an edge from node i to node j, it does not imply that there is also an edge from node j to node i. The graph is said to be undirected, if there is no edge orientation for all the tuples in $\mathcal{E}$. Moreover, if a node j is reachable from the node i in at most K hops, we say that node j is in a K-hop neighbourhood of node i.

Given that graph G is undirected, adjacency matrix $\mathbf{W}$ is defined as $N \times N$ symmetric matrix, where $W_{i,j} = W_{j,i} > 0$ is the weight of an edge $e_{i,j} = (v_i, v_j)$ and $W_{i,j} = 0$ indicates that there is no edge between nodes v_i, v_j. Degree matrix $\mathbf{D}$ is a diagonal matrix, containing the sum of all edge weights connected to v_i

$$D_{i,i} = \sum_{j=1}^{N} W_{i,j} \tag{3.1}$$

The Laplacian matrix is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{W} \tag{3.2}$$

Furthermore, we would like to represent data on top of the graphs. Thus, we define a graph signal as a mapping from the nodes set to the set of complex numbers $x_i : v_i \to \mathbb{R}$. For the convenience, signal values of each node are put into a vector $\mathbf{x} \in \mathbb{R}^N$ The example of a graph signal is given in Figure 3.1.

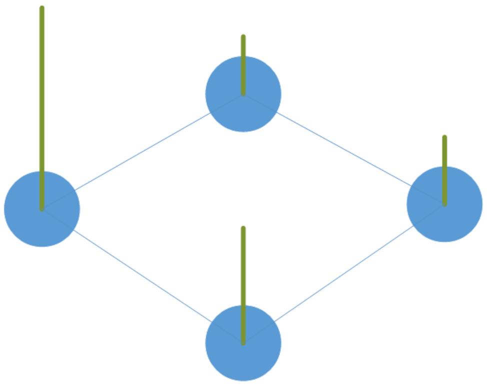


Figure 3.1: An example of positive graph signal [8]. The green lines represents the signal values. The blue circles and lines represents the topology of the graph.

We will refer to the graph representation matrices as a graph shift operator S, which is the symmetric and real-valued matrix.

We can now introduce one of the key operations in graph signal processing (GSP) - shifting signal $\mathbf{x}$ over the graph,

$$\mathbf{x}^{(0)} = \mathbf{I}_N\mathbf{x} \tag{3.3}$$

$$\mathbf{x}^{(1)} = \mathbf{S}\mathbf{x}^{(0)} \tag{3.4}$$

where $\mathbf{x}^{(1)}$ is a one-shift of $\mathbf{x}$ over $\mathbf{S}$, and $\mathbf{x}^{(0)}$ is considered as the zero-shifting over the graph (graph signal itself) (Figure 3.3).

3.2. Graph Convolutional Neural Network

A Graph Convolutional Neural Network (GCNN) is a learning model composed of a concatenation of L layers [6]. It is based upon graph convolutions, and it efficiently incorporates the graph structure. Firstly, we introduce graph convolutions, and then we describe a layered structure of GCNNs.

We denote the graph convolution as a linear shift-and-sum operation of a graph signal. The graph convolution can be expressed as

$$\mathbf{H}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k \mathbf{S}^k \mathbf{x} \tag{3.5}$$

where parameter h_k belongs to the set of learnable parameters $\mathbf{h} = [h_0, \dots h_K]^T$. The k-shifted graph signal $\mathbf{S}^k\mathbf{x}$ is a summary of information in the k-hop neighbourhood, weighted by parameter h_k. The output of graph convolution filter resembles the output of filtering graph signal $\mathbf{x}$ with a Finite Impulse Response filter $\mathbf{H}(\mathbf{S})\mathbf{x}$.

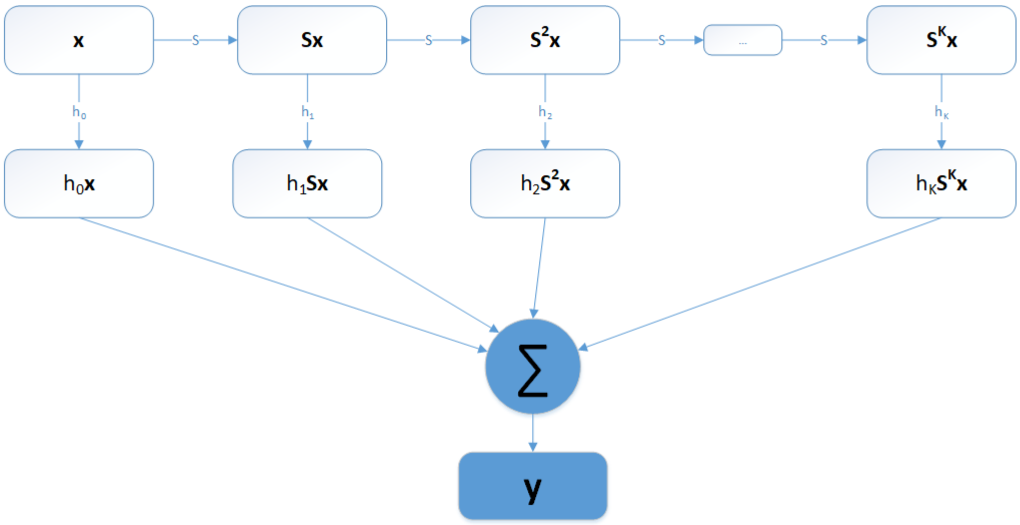


Figure 3.2: Convolution operation on a graph signal. A Graph signal $\mathbf{x}$ is shifted using $\mathbf{S}$ K times. The importance of each shift is weighted by parameter h. The result of each shift is then summed and averaged.

GCNN combines graph convolutional layers which are followed by a nonlinearity (e.g. ReLU). By building a GCNN of L layers, we can express the output at layer l as

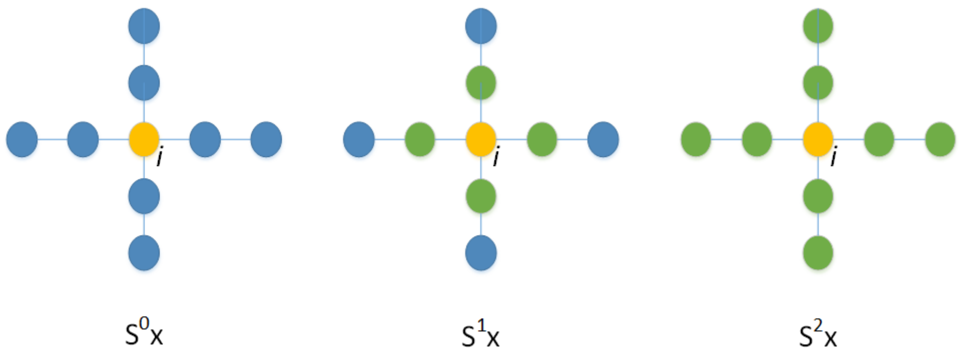


Figure 3.3: Effect of shifting a graph signal associated to each $\mathbf{S}^k\mathbf{x}$ term in Equation 3.5. The shifts are applied for a localized node i (yellow). Affected local neighbours are coloured in green. The shift $\mathbf{S}^0\mathbf{x}$ affects only the node i itself; The shift $\mathbf{S}^1\mathbf{x}$ affects all 1 hop neighbours of node i; finally, the shift $\mathbf{S}^2\mathbf{x}$ affects all 2 hop neighbours of node i

$$\mathbf{x}_l = \sigma_l(\mathbf{H}_l(\mathbf{S})\mathbf{x}_{l-1}), l = 1, \dots, L. \tag{3.6}$$

This cascading allows applying graph convolutions on nonlinear signal transformations coming from the previous layer.

Furthermore, by incorporating multiple features per layer we can substantially increase the representational power of GCNN. We write the output of $(l-1)$st layer as $\mathbf{x}_{l-1} := [\mathbf{x}^1_{l-1};...;\mathbf{x}^{F_{l-1}}_{l-1}]$, where each feature is of dimension N. Likewise, the output of layer l - u_l can be written as a collection of F_l features $\mathbf{u}_l := [\mathbf{u}^1_l;\dots;\mathbf{u}^{F_l}_l]$ where $\mathbf{u}^f_l$ is also of length N. Features of layer L are obtained through convolution and linear aggregation of the features of previous layer. Secondly, we define coefficients of K-tap linear time invariant filter as $\mathbf{h}^{fg} := [[\mathbf{h}^{fg}]_0;...;[\mathbf{h}^{fg}]_{K_{l-1}}]$. The collection of coefficients is used to process the gth feature of $(l-1)$st layer to produce the intermediate feature at layer l $\mathbf{u}^{fg}_l$ [9]. Finally, we interpret a collection $\mathbf{h}^{fg}_l$ as a graph filters that process the features $\mathbf{x}^g_{l-1}$ via graph convolutions. Now we can express intermediate features as

$$\mathbf{u}^{fg}_l = \sum_{k=0}^{K-1}[\mathbf{h}^{fg}_l]_k\mathbf{S}^k\mathbf{x}^f_l := \mathbf{H}^{fg}_l\mathbf{x}^f_l \tag{3.7}$$

which we can interpret as an extractor of local (graph) convolutional features that aggregate through successive local operations. A filter with K taps incorporates information from nodes in $(K-1)$-hop neighbours [9] (Figure 3.3).

Each of the convolutional layers described above consists of $(K+1)$ learnable coefficients, because all nodes share the same scalar h_k to weight the information k hops away from a reference node. Since each convolutional layer l is characterized by the number of $F_{l-1} \times F_l$ coefficient vectors $\mathbf{h}^{fg}_l = [h^{fg}_{0l}, \dots, \mathbf{h}^{fg}_{Kl}]^T$, the total number of learnable parameters for a convolutional layer is $M(K+1)F_{l-1}F_l$ (where M is a number of edges in a graph) resulting in complexity $\mathcal{O}(M(K+1)F_{l-1}F_l)$.

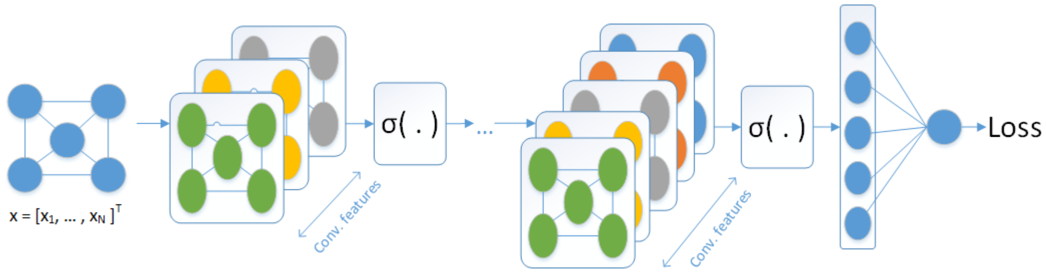


Figure 3.4: Schematic version of a GCNN architecture. The graph signal **x** is passed through F graph filters (called a filter bank). Then it is passed through an activation function $\sigma(.)$ (e.g., ReLu). These operations are called a graph convolution layer and are cascaded L times. Finally, the features of the final convolutional layer are flattened and passed to a fully connected layer to compute the final output which we use for loss minimization.

3.3. EDGENETS

GCNN is helpful to learn meaningful representations on graphs but does not help to identify the essential parts of graphs. This is due to the filter has the same parameters for all edges. In this section, we introduce another GNN - Edgenet, which can learn parameters at the node level and helps to identify relevant parts of the graph.

EdgeNet shares the same layered structure as GCNNs (3.2). However, it uses a different graph filter with different parameters over different edges. We detail it next.

3.3.1. EDGE LINEAR GRAPH FILTERS

In [10], authors use a graph shift operator **S** as the basis for defining filters for graph signals, as it captures sparsity between graph signal and a graph itself. Thus, they define a $N \times N$ diagonal matrix $\mathbf{\Phi}_0$ which matches the sparsity pattern of $\mathbf{I}_N$. The matrix $\mathbf{\Phi}_0$ consists of learnable parameters, and the network can learn at the node level. We can define these parameters as $\phi_{ij}^{(k)}$ to weight the information of its neighbour j at an iteration k. Similarly, consider a collection of matrices $\mathbf{\Phi}_1, ..., \mathbf{\Phi}_K$ which shares the support of $\mathbf{I}_N + \mathbf{S}$. Then, the signal at node i, $z_i^{(k)}$, can be computed as

$$z_i^{(k)} = \sum_{j \in \mathcal{N} \cup i} \phi_{ij}^{(k)} z_j^{(k-1)}. \tag{3.8}$$

We can see recursive expression (3.8) implies that signal $z^{(k)}$ is produced from signal $z^{(k-1)}$ using recursion. By setting $k = 0$ the network can learn the new graph out of the input signal. Setting $k = 1$ results in aggregation of neighbouring nodes at each node i. Setting the $k = 2$ results in aggregation of neighbouring nodes, which already aggregated nodes from their neighbours and so on (Figure 3.5).

Using previous information, the edge varying graph filter can be described as a sum

$$\mathbf{H}(\mathbf{S}) = \sum_{k=0}^{K} \mathbf{z}^{(k)} = \sum_{k=0}^{K} \mathbf{\Phi}^{(k:0)}\mathbf{x} \tag{3.9}$$

where $\mathbf{\Phi}^{(k:0)} := \mathbf{\Phi}^{(k)} \dots \mathbf{\Phi}^{(0)}$ is a product matrix. The output of (3.9) can be interpreted as a sum of scaled and shifted copies of the original signal in time. When interpreting $\mathbf{\Phi}^{(k:0)}$ as a scaled shift, where each shift is a recursive composition of individual shifts $\mathbf{\Phi}^{(k)}$ we can say that (3.9) behaves like a filter. Each shift respects the structure of $\mathcal{G}$ but reweighs individual edges when needed.

Edge Linear Graph filters (ELGF) allows us to learn parameters on a node level and results in a higher order of parameters - $K(M+N)$ than FIR graph filter - K. The increased parameter number allows the network to capture more detail among word connections. However, it may require more data to learn. When the dataset size is smaller (like in our case), it means that the number of filters and filter taps K has to be reduced.

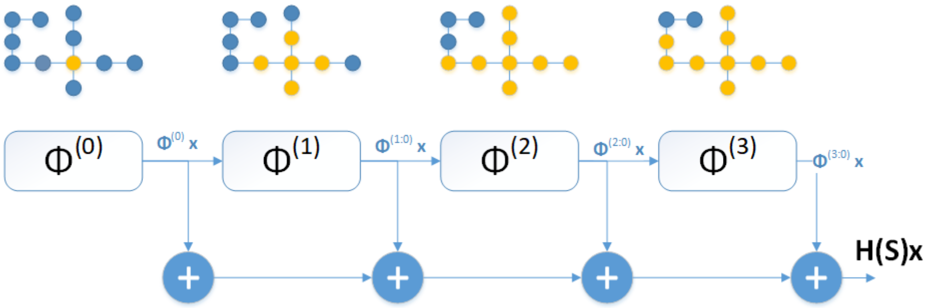


Figure 3.5: Edge Varying graph filter. Each matrix $\mathbf{\Phi}^{(k)}$ acts as a different shift operator. The result is a locally combined graph signal. (Top-left) is the reference node (in yellow) and (Top) shows the result of the local information needed to calculate the result of $\mathbf{z}^{(k)} = \mathbf{\Phi}^{(k:0)}\mathbf{x}$ a the reference node. (Bottom) is the schematic illustration of ELGF with a $K = 3$.

3.4. WORD ADJACENCY NETWORKS

To go from the written text to an input that can be classified by GNNs, we need to extract the graph from the text. We used an already available dataset of graphs extracted from text excerpts using Word Adjacency Network (WAN) method [2]. Further, we are going to introduce the contents of the dataset and explain the WAN method.

3.4.1. DATASET

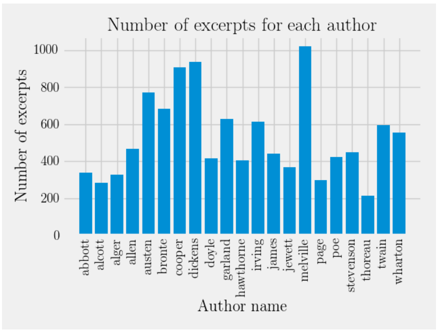
The data set contains text excerpts from 21 authors of the 18th century written in English. Authors may be divided into two categories by nationality: British and American [2]. Besides, there are both female and male writers.

Each author contains its profile: some randomly selected excerpts (varying between 212 to 1022 excerpts). The number of excerpts available for each author differs and is summarized in a Figure 3.6a. Thoreau has the lowest number of excerpts (212) and Melville, the highest (1022). There is a significant difference in some excerpts for particular authors, which will influence the learning algorithms.

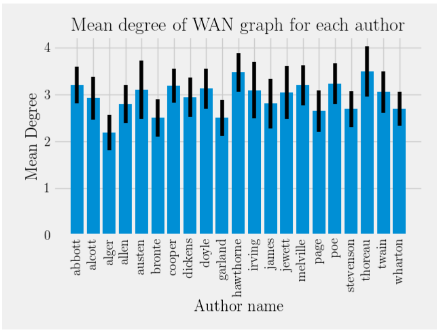
We applied graph centrality measures: mean degree and degree distributions (for each author) - to gain more insights into the dataset. There is a significant difference between authors, where Alger has the lowest mean degree of just above 2 and Hawthorne

	British	American
Male	Stevenson, Allen, Dickens, James, Doyle	Abbott, Alger, Cooper, Garland, Hawthorne, Melville, Thoreau, Twain, Irving, Poe
Female	Austen, Bronte, Jewett, Wharton	Alcott

Table 3.1: Authors split in 4 categories by gender and nationality.



(a) Number of excerpt for each author.



(b) Average degree of WAN graphs for each author

Figure 3.6: Centrality metrics of 18th century authorship dataset.

and Thoreau the largest - around 3.5 (Figure 3.6b).

We plotted the additional graph centrality measures for each author to look for outliers and possible correlations with classification accuracy. These can be found in Appendix B.

3.4.2. AUTHOR FINGERPRINTS

Following 3.4.1, we built Word Adjacency Network (WAN) for each author excerpt. WAN is a graph where nodes are function words, and the weight of an edge represents the likelihood of finding the words close to each other in the text. Firstly, we define a proximity measure between function words and then formalize building WAN of function words.

Function Word Proximity. To identify stylistic features of authors; we study the proximity between function words in a text. We start with a given text t, which we separate into sentences defined as pieces of text located between pairs of delimiters. The words in a sentence are assigned subsequent indexes to define directed proximity between two words which is parametrized by a discount factor $\alpha \in (0,1)$ and a window length D. If we denote by $e(\omega)$ the position of word ω within its sentence, the directed proximity $d(\omega_1,\omega_2)$ (Figure 3.7) from word ω_1 to word ω_2 when $0 < e(\omega_2) - e(\omega_1) \leq D$ is defined as

$$d(\omega_1,\omega_2) := \alpha^{e(\omega_2)-e(\omega_1)-1}. \tag{3.10}$$

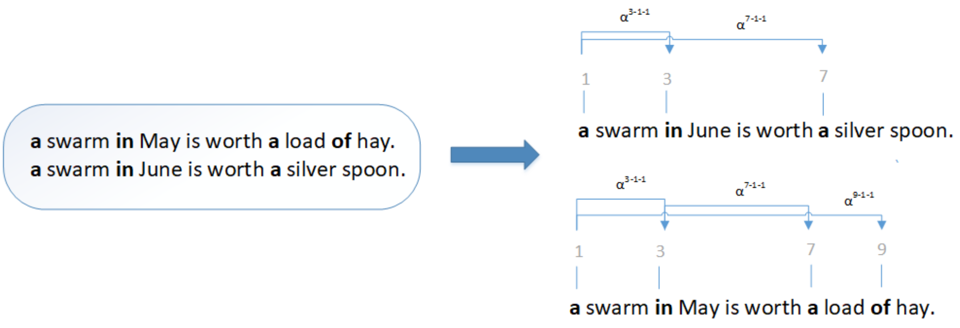

a swarm in May is worth a load of hay.
a swarm in June is worth a silver spoon.
α^{3-1-1}
α^{7-1-1}
1
3
7
a swarm in June is worth a silver spoon.
α^{3-1-1}
α^{7-1-1}
α^{9-1-1}
1
3
7
9
a swarm in May is worth a load of hay.

Figure 3.7: Schematic representation of calculation of word proximity. The text is divided into the sentences. Then we calculate word proximity measure $d(\omega_1, \omega_2)$ for each function word. In this example we only calculate proximity measures for word "a". Grey numbers on top of the function words symbolizes the index of a word in a sentence ($e(\omega)$). Numbers on top of the lines are values of a proximity measure.

Word Adjacency Networks of Function Words. To construct WANs of function words, we accumulate the proximities in (3.10) for all the function words that appear in a text. To formalize it, we say that given each text excerpt t the network $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t)$ is constructed, where $\mathcal{V} = \{1, \ldots, N\}$ is a set of nodes composed by a collection of function words and $\mathcal{E}_t : N \times N \to \mathbb{R}_+$ is a similarity measure between pairs of nodes.

To calculate the similarity $\mathcal{E}_t$, we divide the text t into sentences s_t^h and denote by $s_t^h(e)$ the word in the e-th position within sentence h of text t. Then, we determine proximities within each sentence as per (3.10) and accumulate the measure over all sentences. Therefore, the proximity score from function word i, to function word j can be written as

$$\mathcal{E}_t(i,j) = \sum_{h,e} \sum_{d=1}^{D} \alpha^{d-1} \mathbb{1}\left[s_t^h(e) = i\right] \mathbb{1}\left[s_t^h(e+d) = j\right]. \tag{3.11}$$

Using similarities $\mathcal{E}_t$ from text excerpts, we can construct a graph for each text excerpt for further classification using GNNs. However, we are also interested in building a WAN fingerprint for each author in a dataset. Thus, we construct a network $\mathcal{G}_c$ for every author a_c as $\mathcal{G}_c = (\mathcal{V}, \mathcal{E}_c)$, where

$$\mathcal{E}_c = \sum_{t \in T^{(c)}} \mathcal{E}_t \tag{3.12}$$

is the component wise sum of similarity networks for each text excerpts by a particular author. It serves as a graph fingerprint of an author.

Normalization of WAN. However, a matrix $\mathcal{E}_c$ depends on the length and number of excerpts of author a_c. Thus, we normalize the similarity measures

$$\hat{\mathcal{E}}_c(f_i, f_j) = \frac{\mathcal{E}_c(f_i, f_j)}{\sum_j \mathcal{E}_c(f_i, f_j)} \tag{3.13}$$

for all $f_i, f_j \in F$.

We can construct a graph, where normalized similarity measure matrix $\hat{\mathscr{E}}_c$ serves as an author fingerprint. This matrix can be further used as a graph shift operator for graph neural networks.

Discussion. A WAN is a graph being a union of function words common for each author. A WAN may carry unnecessary information for a particular author, but that is crucial for the other one. Thus, it hinders the fingerprint of a particular author. Thus, we hypothesize that each author can be identified by a sparser WAN graph which is hidden in $\mathscr{G}_c$. We aim to identify each author most meaningful fingerprint that can distinguish the main author characteristics through graph neural networks.

3.5. DISCUSSION

We aim to improve on authorship attribution problem and explore the emerging field of GNNs. Extracting WAN graphs allows building meaningful author's fingerprints. Since WANs are graphs, those fingerprints can be later classified by state-of-the-art GNN architectures, which are able to utilize the underlying topology of a fingerprint.

GNNs allows us to improve upon the task, but also to take a glimpse inside what is being learned. With the EdgeNet architecture, we can extract sparser author fingerprints from WANs. This allows us to use these sparser WAN fingerprints for training and improve training times and in some cases, the classification accuracy. This method also allows us to visualize the sparsified fingerprint as a graph, where we can interpret what function words are the most important for the certain author.

In further chapters, we describe how to extract meaningful sparse author fingerprints.

4
Method

In this chapter, we explain the method used for extracting sparse author fingerprints and evaluating GNNs for authorship attribution. Firstly, we introduce how to extract sparse author fingerprints from WANs using EdgeNets in Section 4.1.1. Then we introduce the experiments we conducted in Section 4.2.3 and all necessary settings for the experiment.

4.1. SPARSE AUTHOR FINGERPRINTS

WAN author fingerprints, combined with GCNN, gives good classification accuracy. However, we argue that it may carry unnecessary information, and it is possible to extract sparse author fingerprints. Meaning that dimensionality of the learning data would be reduced, learning times would decrease, and accuracy would stay sufficient. Thus, we hypothesize that each author can be identified by a sparse WAN graph which hides in $\mathcal{G}_c$. We aim to identify each author most meaningful fingerprint that can distinguish the main author characteristics through graph neural networks which are deep learning algorithms over graphs.

4.1.1. TECHNIQUE

To extract sparse author fingerprints we used EdgeNets (3.3). We used the fact that EdgeNets are able to learn parameters at node level to extract author fingerprints. Our hypothesis is that parameters that has high value in learned matrix $\boldsymbol{\phi}^{(0)}$ are indicate that a particular node (or a function word in authorship attribution domain) is of higher importance.

To extract nodes of "higher importance" for each author, we built a simple EdgeNet architecture with 1 EVGF layer and 1 filter tap matrix. This configuration means that EdgeNet architecture is learning the graph itself and we are aggregating information from 1 hop neighbours and have only one matrix $\boldsymbol{\phi}^{(0)}$ to extract the nodes from. Then, we train this architecture for one versus all classification task. We do the training for each author to collect matrices $\boldsymbol{\phi}^{(0)}$. The classification accuracy is generally lower; however, we are only looking for collecting author fingerprint that is learned by the EdgeNet.

After the matrices are collected, we use them as the graph shift operator (GSO). To reduce dimensionality, we need to identify the most important nodes. We identify the most important nodes for each author following this rule for each entry a_{ij} in matrix $\boldsymbol{\phi}^{(0)}$

$$a_{ij} > \max(\boldsymbol{\phi}^{(0)}) - \theta \max(\boldsymbol{\phi}^{(0)}) \tag{4.1}$$

where we adjust parameter θ to the desired number of nodes extracted To find the optimal sparsity of graph shift operator since most of the values in this matrix are close to zero this method allows us to increase the sparsity of author fingerprint that is obtained building a WAN network using aggregation from graph filters.

We have also tried the other way of extracting the sparse author fingerprints by removing the most significant entries in $\boldsymbol{\phi}^{(0)}$ and testing how it affects the accuracy of classification. However, this method resulted in higher amounts of nodes to keep and lower accuracy.

4.1.2. SPARSE GRAPH SHIFT OPERATOR

Following subsection 4.1.1, we transformed extracted sparse author fingerprint matrices to GSO for GCNNs. Most of the values in extracted matrices were zero. Thus we removed rows and columns from the sparse author fingerprint matrix where all the values were zero. Removing rows and columns led to reduced dimensionality in data. We adjusted accordingly: removed entries which were removed from GSO.

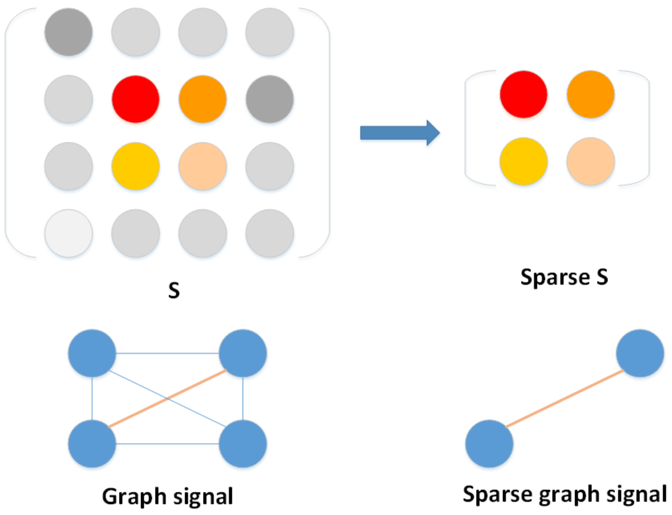


Figure 4.1: Schematic representation of extracting sparse author fingerprints. We take the GSO **S**, identify the node with a highest value $S_{i,j}$ (in red). Then we calculate which nodes in **S** has value $a_{i,j} > S_{i,j} - \theta S_{i,j}$ for each $a_{i,j} \in \mathbf{S}$. These nodes are depicted in brighter colours (yellow and red). Those rows and columns that have no nodes which exceeds the threshold (grey colour) are removed from **S** and graph signal **x**.

4.2. AUTHORSHIP ATTRIBUTION

To assess the effectiveness of using GNNs for authorship attribution, we used multiple GNN architectures with various parameters.

4.2.1. ARCHITECTURES

Graph Convolutional Neural Network. Graph Convolutional Neural Network (GCNN) is the simplest and the most shallow architecture used in this thesis. It consists of graph convolutional, activation function, fully connected layer, and a final sigmoid activation function (Figure 4.2).

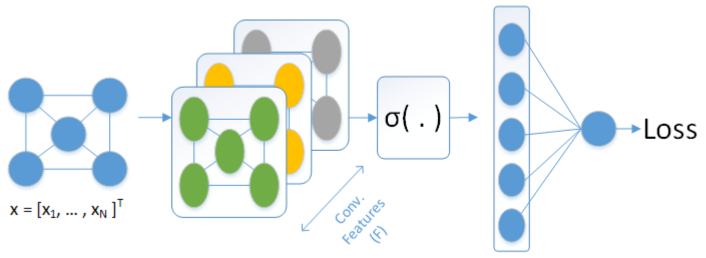


Figure 4.2: The schematic illustration of 1 layer architecture.

The graph convolutional layer is implemented using a Finite Impulse Response (FIR) graph filter. Thus, the number of learnable parameters in each layer is $F \times (K-1)$, where F is the number of filter banks and K is the filter order. This architecture was used as a baseline in all experiments as it is the simplest and fastest. In our experiments, we have also used GCNN with two convolutional layers to test how the depth of the network influences classification.

Edge-varying Graph Neural Network (EdgeNet). EdgeNet was used in two ways: as a comparison with GCNN and as a tool to extract the sparse author graph signatures.

This architecture is more complicated than GCNN in terms of learnable parameters (since it stores weights for each edge of the graph). Thus, we compared the performance of the EdgeNet against the simpler GCNN. The layer structure of this architecture is the same as in 4.2.1, only that graph convolutional layer is replaced by an edge varying layer, which utilizes edge varying graph filter.

We have also used a hybrid EdgeNet - GCNN architecture, where the first layer is edge varying graph filter, and the second layer is a graph convolution. We assume that more complex edge varying layer will capture more details in a text, and the following graph convolutional layer will extract only the most essential features.

Baseline architectures. We also used linear learning models such as K-Nearest Neighbour (KNN) and SVM classifiers. We included these models to compare how graph-based deep models are performing against simpler ones.

4.2.2. HYPERPARAMETERS

Loss function. Each architecture that we use for binary classification task utilizes Binary Cross entropy loss function. Using binary cross-entropy loss meant that the output of a fully connected layer had to be squished between $[0;1]$. This was achieved using sigmoid activation function after fully-connected layer.

Optimizer. All architectures are using ADAM optimizer with a learning rate of 0.001.

Hyperparameters. We used a batch size of 16 for all the experiments. Other hyperparameters such as a number of filter taps and convolutional features are varied throughout the experiments.

4.2.3. EXPERIMENTS

To test the performance of GNNs for the author classification task, we came up with a set of experiments which utilizes multiple aspects of the dataset:

- **One v.s. All**. We wanted to classify each author in one-v.s.-all fashion, meaning that we take the author we want to classify and the rest of the authors (since there are 21 authors in the dataset, this means 21 different classifications). The task remains a binary classification problem with 21 experiments in total.

- **One v.s. One**. To observe the impact GNNs have on classification results, we included a binary classification task, where we take an author, and a second author whose averaged signal is the closest to the selected author.

- **Gender**. We believe there are patterns between male/female authors. To fully exploit the dataset and identify those patterns, we included a binary classification task by gender.
- **Nationality**. Similarly, to find the patterns between the nationality of authors, we split it into two groups by nationality and included a binary classification task by nationality.
- **Gender and Nationality**. To test how GNNs would perform against simpler models on multi-label classification, we have split the dataset into 4 groups - by gender and nationality.
- **Sparse author fingerprints**. We have conducted all the tests described above using sparse author fingerprint instead of the regular one.

Firstly, we wanted to explore the performance of GNNs on authorship attribution task. Experiments were designed to exploit the dataset from different perspectives (by authors, gender and nationality). Also, we wanted to include tasks that are different in difficulty, where one v.s. one classification would be the easiest, and multi-label gender-nationality classification would be the hardest. We hypothesized that GNNs being the more complex model (compared to baseline), and being able to capture connections between words would perform better on more complex tasks.

Furthermore, we wanted to test how does GNNs perform when we use the sparse author fingerprints. Since the data dimensionality is lower than regular WAN fingerprints, we would expect a drop in performance. However, we tested a hypothesis that GNNs would outperform simpler models because of its ability to capture the structure.

We used the one v.s. all task as a baseline for evaluating GNNs for author attribution task. Thus, we did a grid search for best performing hyperparameters. We took two hyperparameters into the consideration: the number of shifts (K) and the number of convolutional features (F) after convolutional layer. We took into consideration these values for each parameter:

$$F = \{2,4,8,16,32,64\} \tag{4.2}$$
$$K = \{1,2,3,4,5,6,7,8,9,10\} \tag{4.3}$$

This resulted in $F \times K = 60$ different experiments. For each combination, we collected these metrics: accuracy, f1 score, and AUC score.

4.2.4. EXPERIMENTAL SETUP

To run an experiment using GNNs, we need a GNN architecture, graph shift operator and the graph signal (training data). In this subsection, we explain how we define a graph shift operator, the graph signal and how we run the experiments.

Graph shift operator. The GSO used depends on the classification task. For one v.s. one classification, we used an averaged WAN of one of the authors. An averaged WAN is a normalized similarity measure of an author described in 3.4. Similarly, for one v.s.

all classification task, we use an averaged WAN of the author we want to classify against others.

Furthermore, for gender or nationality classification, since we still operate in terms of binary classification problem, we used averaged WANs of male (for gender) and averaged WAN of American authors (for nationality).

For multi-class classification problem of gender-nationality we have experimented with all combinations of GSOs, this includes all combinations of two sets $\{male, female\}$, and $\{American, British\}$.

Graph signal. As a graph signal we used the frequency count of function words contained in the WANs. For example, in specific excerpt, Poe used a function word "the" 50 times, so the signal value at node "the" would be 50. The graph signal serves as a training data for GNNs and is obtained together with the dataset.

Experiment reproducibility. To obtain accurate information about the performance we ran every single experiment 10 times. All provided accuracy results are the averaged from multiple experiments.

DATA SPLITS

Authorship dataset consists of a different number of excerpts of 21 British and American authors. One of the experiments was one-v.s.-all classification. Thus, we had to split the dataset in such way that it would be balanced in terms of number of excerpts of the author we are trying to classify against others. Also, experiments included classifying authors by gender and nationality, which requires appropriate splitting.

Equal data-split. All experiments from one-v.s.-one and one-v.s.-all were carried using equal data split. The strategy used to produce equal data-splits was such:

1. Take all the excerpts of the author we are trying to classify (against all others) and split by train-test-validation ratio.
2. Randomly take the same amount of excerpts (as were taken for the author we try to classify) from the rest of the authors and split it by train-test-valid ratio
   (a) Make sure that the same amount of excerpts were taken from each remaining author (if we need to take 200 excerpts from all the remaining authors, we divide this number by the number of remaining authors and take that many excerpts from each).
   (b) Make sure that the excerpts for each remaining author are taken uniformly (so that with each data split we would take different excerpts than in previous data split from each author).

This data splitting technique makes sure that the dataset is balanced and that each run of an experiment would use the unseen excerpts from authors.

Gender data-split. To produce the data-split for classification by gender, we have divided authors into two groups (based on gender). Naturally, the number of excerpts in one group (in this particular case, male) was more significant than in the other group. To maintain data-split balance, we took the same amount (number of excerpts from the smaller group) of excerpts from each group by randomly taking a required number of excerpts from the larger group. We did not regulate for the number of excerpts taken from each author to be equal.

Nationality data-split. To produce the data-split for classification by nationality, we have divided authors into two groups (based on nationality). Further, we followed the same process as in gender data-split (4.2.4).

Gender-Nationality data-split To classify authors by gender and nationality, we have divided the dataset into 4 parts: British female, British male, American female and American male authors. The smallest group was American female with only 1 author and the largest - American male with 10 authors. Thus, the data-split was imbalanced. In order not to reduce the number of excerpts used in training, we decided to train with imbalanced data-split. We compensated for this using weighted cross-entropy loss function, which penalizes miss-classification of the smaller group more than, the larger group (depending on the size difference of the groups).

4.3. SUMMARY

In this chapter, we introduced all the crucial tools for further numerical experiments. Firstly, we introduced the method of extracting sparse author fingerprints using EdgeNet architecture. Setting EdgeNet parameters in a certain way (1 feature and 1 hop neighbourhood) we can learn new graphs from the graph signal and graph shift operator. The learned graph has many edges with low weights; thus, we introduced the technique of removing those nodes and reducing the input dimensions of training data. These sparse author fingerprints then can be extracted for each classification task (e.g., for each author, for a specific gender or nationality).

Furthermore, we discussed all the necessary setup and architectures used for further experiments. Mainly, we used the GNN architectures: GCNN, EdgeNet and hybrid EdgeNet-GCNN architecture with a baseline model of SVM (and kNN). This set of architectures will allow us to compare the difference between more complex/shallow architectures.

5 Experimental Results

In this section, we present the results for each experiment defined in Section 4.2.3. For each experiment, we provide insights and compare the results among different GNN architectures. Also, we repeated the same experiment with the sparse author fingerprint instead of a WAN fingerprint.

5.1. 1 vs All Classification

As described in Section 4.2.3, we took our simplest one-layer GCNN architecture (4.2.1) as a baseline and then ran a grid search, looking for the most optimal number of filter taps (K) and convolutional features (F). Further, we have compared the results from the best performing experiments with the results of other architectures described in Section 4.2.1. The results of an experiment can be found in Table 5.1.

5.1.1. Results

Firstly, we observed the difference in classification accuracy among authors. The lowest error was achieved for Austen with 0.03 and the highest error was observed for Allen 0.13. The difference in errors is significant and indicates that classifying certain authors against the rest is more difficult. We hypothesize that WAN method could not capture a unique fingerprint for certain authors (compared to all the rest). However, the results in general are satisfying with average classification error (including all authors) of 0.07.

In addition, we have observed that the best GCNN configuration usually outperformed simple feed-forward neural network (with an exception of couple of authors) and linear classifiers. Although, the difference in error was not significant.

Furthermore, we wanted to investigate how making an GCNN architecture more complex by adding additional layers will affect the classification error. Thus, we have experimented with additional variations of one-layer GCNN architecture:

- 2 layer GCNN architecture (with a search for optimal second layer features)
- 3 layer GCNN architecture (using the best performing features from two-layer architecture)

More complex 2 or 3 layer GCNN architectures were usually outperformed by the single layer GCNN architecture. Interestingly, the deeper architectures could not improve on classifying authors where classification error was to the higher end (on 1 layer GCNN architecture). We believe this is because increasing the depth (and naturally the complexity) of GCNNs does not improve classification error.

Finally, we compared the best results of GCNN against more complex EdgeNet architecture. Firstly, we used the most optimal hyperparameters obtained from grid search. This resulted in an increase of classification error compared to GCNNs. We hypothesized that it may be due to complexity of EdgeNets. Since EdgeNets have more learnable coefficients and our dataset size stayed the same, it may be more optimal to run EdgeNets with smaller number of filter taps and convolutional features. Thus, we did a grid search for optimal hyperparameters for EdgeNets. This resulted in significant decrease of classification error (compared to using the best hyperparameters from GCNN grid search), however, it have not improved the classification error of the most optimal GCNN run.

5.1.2. Sparse author fingerprints

As mentioned in Section 4.1.1, we have extracted sparse author fingerprints for 1 vs all task. As the dimensionality of the data reduced, we expected a drop in classification accuracy. However, we also wanted to test the hypothesis of how GNNs are able to improve compared with SVMs in the lower dimensionality setting. Our assumption was that due

Table 5.1: Results of 1 vs all authorship classification task. "2L" means two layer architecture. For EdgeNet we used the best parameters from GCNN hyperparameter search. Also, we made a separate hyperparameter search for EdgeNets (indicated by "HP search").

	GCNN	std	Edg 1	std	Edg (HP search)	std	GCNN 2L	std	SVM	std	KNN	std
Abbott	0.96	0.015	0.94	0.02	0.95	0.02	0.95	0.01	0.94	0.03	0.86	0.03
Stevenson	0.90	0.015	0.90	0.02	0.87	0.03	0.90	0.02	0.89	0.03	0.67	0.02
Alcott	0.95	0.023	0.95	0.02	0.94	0.02	0.94	0.02	0.94	0.02	0.72	0.03
Alger	0.92	0.021	0.91	0.02	0.91	0.02	0.91	0.02	0.92	0.03	0.68	0.04
Allen	0.87	0.018	0.85	0.03	0.86	0.03	0.87	0.03	0.87	0.01	0.75	0.04
Austen	0.97	0.0099	0.96	0.01	0.00	0.00	0.97	0.01	0.97	0.01	0.91	0.02
Bronte	0.93	0.015	0.91	0.02	0.00	0.00	0.92	0.01	0.93	0.02	0.72	0.02
Cooper	0.95	0.0085	0.94	0.01	0.93	0.01	0.95	0.01	0.95	0.01	0.82	0.02
Dickens	0.89	0.011	0.88	0.02	0.85	0.01	0.89	0.01	0.89	0.01	0.76	0.02
Garland	0.90	0.011	0.88	0.02	0.86	0.01	0.89	0.02	0.89	0.01	0.67	0.02
Hawthorne	0.90	0.025	0.88	0.03	0.00	0.00	0.90	0.02	0.90	0.02	0.81	0.03
James	0.94	0.016	0.94	0.02	0.92	0.02	0.94	0.02	0.95	0.01	0.79	0.02
Melville	0.88	0.013	0.89	0.02	0.00	0.00	0.88	0.02	0.89	0.02	0.79	0.02
Page	0.90	0.031	0.89	0.02	0.00	0.00	0.89	0.02	0.88	0.01	0.74	0.03
Thoreau	0.96	0.02	0.95	0.03	0.95	0.03	0.95	0.01	0.95	0.02	0.88	0.04
Twain	0.91	0.022	0.90	0.02	0.00	0.00	0.92	0.02	0.92	0.02	0.86	0.03
Doyle	0.93	0.016	0.92	0.02	0.92	0.02	0.93	0.03	0.95	0.01	0.78	0.03
Irving	0.92	0.017	0.91	0.02	0.91	0.03	0.91	0.02	0.92	0.02	0.83	0.03
Poe	0.94	0.019	0.93	0.02	0.00	0.00	0.93	0.01	0.93	0.02	0.88	0.02
Jewett	0.94	0.017	0.94	0.02	0.94	0.02	0.94	0.02	0.94	0.02	0.76	0.03
Wharton	0.96	0.011	0.94	0.01	0.95	0.01	0.95	0.02	0.95	0.01	0.71	0.03

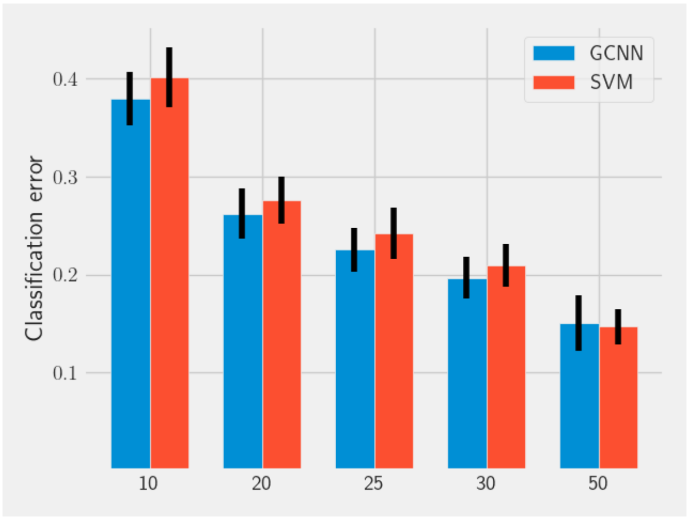


Figure 5.1: Comparison of error rate of GCNN and SVM using sparse author fingerprints. X-axis represents the percentage of the maximal GSO value of the graph (Section 4.1.1). The higher the percentage, the higher the dimensions of the training set.

to ability to capture topology GCNNs will be able to classify authors better. Looking at Figure 5.1 we can see the trend of GCNNs outperforming SVMs where the dimensionality of the data is lower. The results support our hypothesis that GCNNs are able to make use of underlying topology.

In addition, we wanted to test for the most optimal reduction in dimensionality, which would still yield considerably good accuracy. We ran a grid search for the percentage parameter θ (described in 4.1.1) still using the same 1 vs all experiment. Values of θ were such

$$\theta = \{0.01, 0.02, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.5\} \tag{5.1}$$

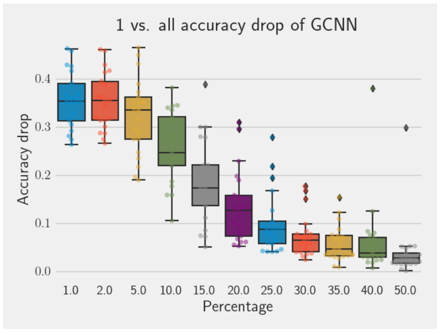
meaning that when the percentage value θ is the lowest, the dimensionality of input and a GSO will be the lowest.

We wanted to see the broader picture of accuracy-dimensionality relation, thus we have calculated an **Average accuracy drop** value for each percentage. The accuracy drop value was calculated as

$$AccuracyDrop_{percentage} = AccBest_{author} - Acc_{perc} \tag{5.2}$$

Then we took an average of all accuracy drop for each author and calculated an **Average accuracy drop** value for each percentage.

Figure 5.2 visualizes average accuracy drop for each percentage in a grid search. We can observe dramatic performance drop when the percentage is low. However, we have to account that the input dimensions with this percentage is usually around 3. Furthermore, we notice that with percentage of 30 the network is able to keep relatively high performance with great reduction in dimensions (which leads to faster training times).



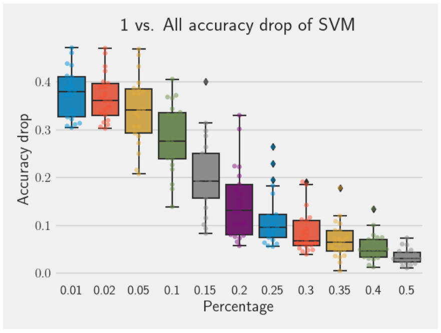


Figure 5.2: Average accuracy drop of GCNN and SVM with reduced dimensions of input and GSO. Lower percentage means less dimensions. We can see that accuracy stabilizes around percentage of 30.

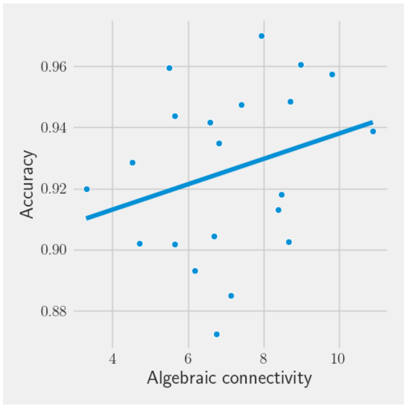


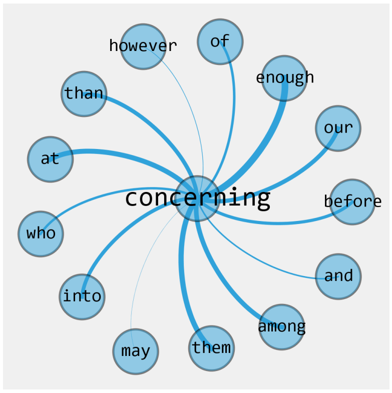
Figure 5.3: Scatterplot of accuracy of 1 vs all classification against Fiedler's eigenvalue.

We also looked for correlation between centrality measures of the WAN fingerprints of particular author and classification performance. One of the centrality measures that we were interested in was the second smallest eigenvalue of the graph Laplacian matrix, also called the Fiedler's eigenvalue (or algebraic connectivity). The magnitude of this value shows how well connected the graph is. We noticed the small correlation of accuracy and algebraic connectivity Figure 5.3. This may indicate that the authors who uses higher variety of combinations of function words are classified better than the ones that uses less combinations of function words.

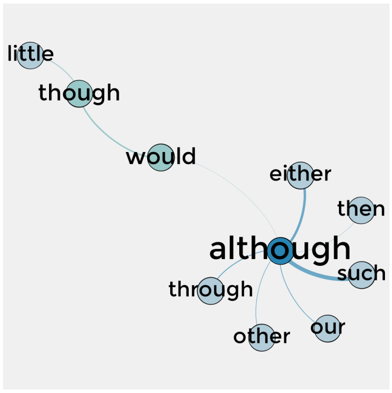
5.1.3. Visualization Of Sparse Author Fingerprints

With the reduced dimensions of author fingerprints, we were able to visualize them and look for the most important interconnections of function words. Since we had obtained a sparse fingerprint for each author, we were able to map it to original function words.

Visualized fingerprints kept the topology and gave some interesting insights into author's writing style. For example, a fingerprint of Melville (Figure 5.4a) forms a shape of a star. The word "concerning" seems to be important for classifying excerpts of Melville

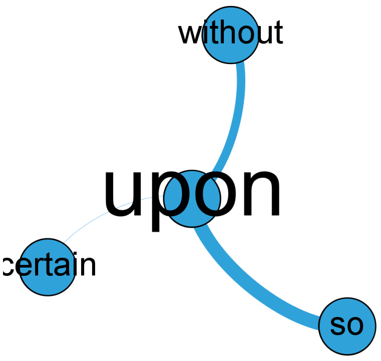


(a) Sparse fingerprint of Melville.

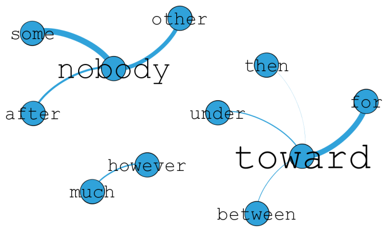


(b) Sparse fingerprint of Poe.

as it forms connections with others. Furthermore, we can observe star-like shape in a sparse fingerprint of Poe, where the word "although" seems to be pivotal (Figure 5.4b). We can also observe more simple star-like shapes in fingerprints of Wharton (Figure 5.5a) where the word "upon" is in the center and Jewett (Figure 5.5b) where multiple star shapes are formed. Visualizations for other authors can be found in Appendix A.



(a) Sparse fingerprint of Wharton.



(b) Sparse fingerprint of Jewett.

5.2. 1 VS 1 CLASSIFICATION

In similar fashion to the 1 vs all classification, we experimented with 1 vs 1 classification. In this setting we were mostly interested in the comparison between the best performing GNN (GCNN) and SVM architectures.

Since there are 21 authors, we decided to reduce the amount of experiments by classifying a particular author against the author that is the most similar. Similarity between authors was computed by calculating Euclidean distance between graph signals of authors. We hypothesized that classification task would become harder when graph signals of authors are the most similar and the role of the topology would be more important.

5.2.1. RESULTS

Firstly, we can see that GNNs are able to achieve relevantly low error rate classifying authors with each other (Table 5.2). The lowest error rate was only 0.01 for Wharton, and the highest 0.05 for Garland.

We noticed that the accuracy of GCNN is constantly better on harder classification task (Table 5.2). Where the average improvement of GCNN among all authors is **0.008** and the highest improvement is **0.019**. This means, that when the problem is harder, the ability to use underlying topology of the graph helps GCNNs to achieve better results than more simple methods.

5.2.2. SPARSE AUTHOR FINGERPRINT

Furthermore, we ran the same 1 vs 1 tests using sparse author fingerprints. We wanted to test the hypothesis that reduced dimensionality will result in GCNN performing even better than SVM, since loss of dimensions makes the task even more complex.

We can see that GCNN improves on the accuracy of SVM (Table 5.2). On average (for each author) the classification accuracy using GCNN is better by 0.014. Also, the overall improvement against SVM using sparse fingerprints is higher (0.014 against 0.008). Thus, we can conclude that advantage of using GCNNs is magnified when the complexity of the problem is increased.

Similarly to 1 vs all classification task, we have experimented with the most optimal reduction of dimensionality. We ran the same grid search for the percentage parameter θ (described in 4.1.1). We have summarized the results using average accuracy drop metric Figure 5.7.

In general, we observed similar results to 1 vs all classification task. Some differences being a longer tail in figure (meaning that after a certain number of nodes kept, the increase in performance is insignificant). Also, with percentage of 0.5 the performance has improved, meaning that EdgeNet was able to learn meaningful representation and actually get rid of certain nodes that were detrimental for classification accuracy. We could not see such an improvement on 1 vs all task, since it is harder to identify detrimental function words (nodes) in the full set of authors (since every word is important for certain author).

Table 5.2: Accuracy results of 1 vs 1 authorship classification task using normal WAN and sparse author fingerprints.

	GCNN	Std	SVM	Std	GCNN (FP)	Std	SVM (FP)	Std
Abbott	0.97	0.01	0.97	0.01	0.90	0.02	0.90	0.02
Stevenson	0.96	0.01	0.95	0.02	0.94	0.03	0.93	0.02
Alcott	0.98	0.01	0.97	0.02	0.92	0.02	0.91	0.01
Alger	0.98	0.01	0.97	0.01	0.90	0.02	0.87	0.02
Allen	0.96	0.01	0.95	0.01	0.81	0.02	0.79	0.02
Austen	0.98	0.01	0.98	0.01	0.88	0.02	0.88	0.02
Bronte	0.97	0.01	0.97	0.01	0.89	0.01	0.88	0.01
Cooper	0.98	0.01	0.97	0.01	0.80	0.01	0.78	0.02
Dickens	0.97	0.01	0.96	0.01	0.93	0.02	0.94	0.02
Garland	0.95	0.02	0.94	0.01	0.89	0.02	0.87	0.02
Hawthorne	0.98	0.01	0.97	0.01	0.95	0.02	0.94	0.01
James	0.98	0.01	0.97	0.02	0.93	0.02	0.93	0.02
Melville	0.96	0.01	0.95	0.02	0.91	0.02	0.88	0.02
Page	0.97	0.02	0.97	0.01	0.90	0.04	0.90	0.02
Thoreau	0.99	0.01	0.97	0.02	0.90	0.02	0.87	0.02
Twain	0.96	0.01	0.95	0.01	0.91	0.02	0.89	0.02
Doyle	0.98	0.01	0.97	0.01	0.95	0.01	0.95	0.01
Irving	0.98	0.01	0.98	0.02	0.92	0.02	0.90	0.02
Poe	0.99	0.01	0.99	0.01	0.95	0.02	0.94	0.02
Jewett	0.97	0.02	0.96	0.01	0.88	0.02	0.85	0.02
Wharton	0.99	0.01	0.97	0.01	0.91	0.02	0.89	0.01

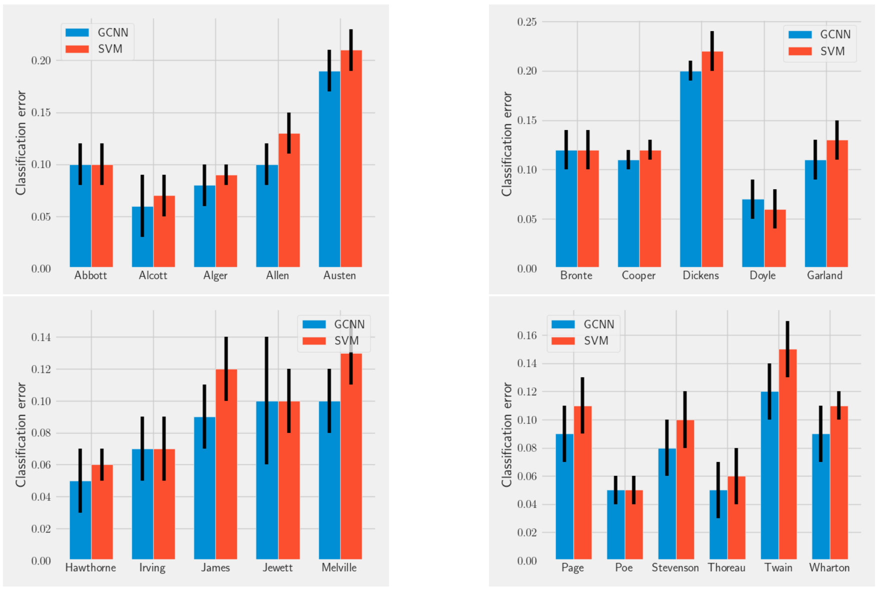


Figure 5.6: Comparison of classification error in 1 vs 1 classification task using sparse author fingerprints.

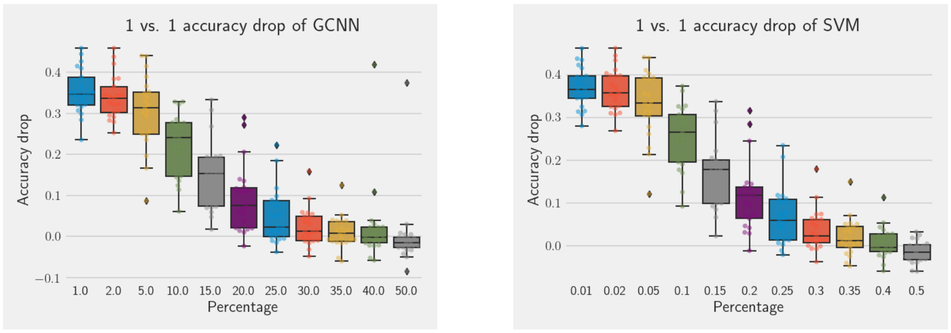


Figure 5.7: Average accuracy drop of GCNN and SVM with reduced dimensions of input and GSO. Lower percentage means less dimensions. We can see that accuracy stabilizes around percentage of 30. For sparse author fingerprints we used ratio of 0.3.

5.3. Classification by Gender

To classify authors by gender, we had to make an adjustment to the dataset splitting technique. For this task, we have divided authors into two groups (by gender). Since in this particular dataset there are more male authors than female, we ensured dataset balance by randomly selecting needed number of text excerpts from male authors.

For this experiment we have used the best performing hyperparameters from 1 vs all classification for GCNNs and Edgenets. For hybrid architecture we used the best parameters from both architectures.

5.3.1. Results

Classifying text excerpts by gender would require identifying certain patterns that are present between male/female authors. We believed this task would be harder than 1 vs all or 1 vs 1 classification and expected more complex architectures to do better. We have also included a hybrid Edgenet - GCNN architecture, where the first layer is contains edge varying graph filter, and the second layer is a standard convolutional filter.

The results confirmed our hypothesis - Edgenet was superior against GCNN, together with SVM. However, the hybrid architecture performed the best Table 5.3. The lowest achieved error of the experiment was 0.08 proves that the GNNs are a powerful tool for identifying stylistic patters and could be further used for developing privacy preserving methods.

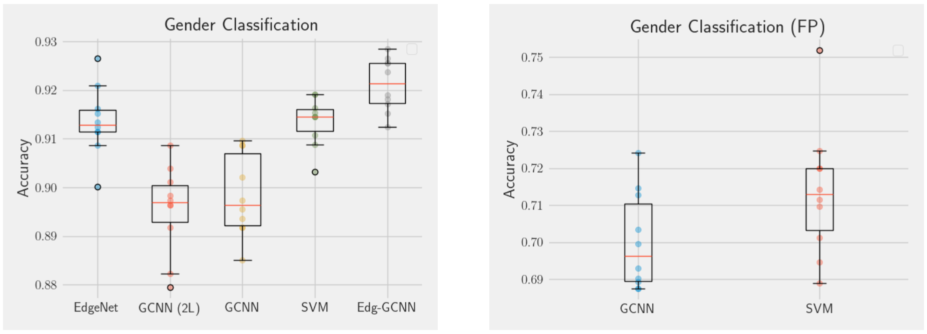


Figure 5.8: Accuracy of classification by gender of various architectures using WAN and sparse author fingerprints. For sparse author fingerprints we used ratio of 0.3.

Table 5.3: Results of author classification by gender. "GCNN_2" symbolizes 2 layer GCNN architecture.

	Edg	GCNN_2	GCNN	SVM	Hybrid	GCNN (FP)	SVM (FP)
Accuracy	0.91	0.9	0.9	0.91	0.92	0.7	0.71
Std	0.0067	0.0086	0.0081	0.0046	0.0052	0.014	0.017
F1	0.91	0.9	0.9	0.91	0.92	0.7	0.73
AUC	0.91	0.9	0.9	0.91	0.92	0.7	0.71

5.3.2. Sparse Author Fingerprints

When using sparse author fingerprint on more complex task of classifying authors by gender we noticed a considerable performance decrease (0.2 comparing GCNN). We believe the drop in performance is high due to complexity of the task and the fact that there were not enough dimensions to accurately capture the difference in graph signals.

5.4. Classification by Nationality

Similarly to classification by gender 5.3, we expected GNNs to be able to identify the patterns that are present between British and American authors. Likewise, we split the dataset into 2 groups by authors' nationality and used the best performing hyperparameters on architectures.

5.4.1. Results

We hypothesized that classifying authors by nationality will also be a harder task than 1 vs all or 1 vs all classification. We were also interested how will this task compare to classification by gender.

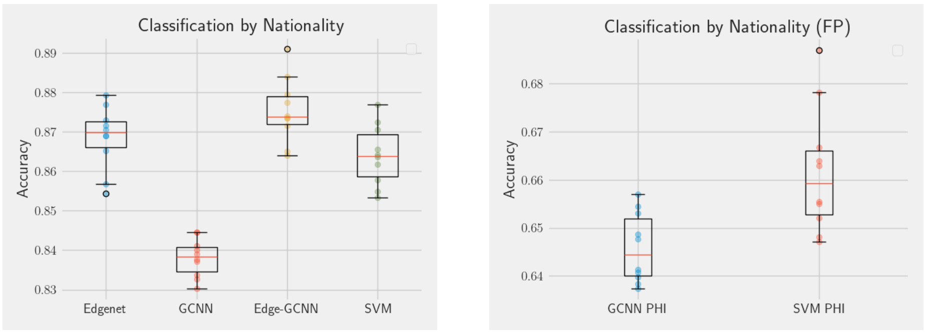


Figure 5.9: Accuracy of classification by nationality using WAN and sparse author fingerprints. For sparse author fingerprints we used ratio of 0.3.

Table 5.4: Results of classification by authors' nationality. For sparse author fingerprints we used ratio of 0.3.

	Edgenet	GCNN	Edge-GCNN	SVM	GCNN (FP)	SVM (FP)
Accuracy	0.87	0.84	0.88	0.86	0.65	0.66
F1	0.87	0.84	0.87	0.86	0.59	0.61
AUC	0.87	0.84	0.87	0.86	0.01	0.01
Std	0.01	0.00	0.01	0.86	0.65	0.66

As it is seen from Table 5.4, classification accuracy achieved in this task was lower than the one in gender classification. Interestingly, EdgeNet architecture was significantly better than GCNN. The hybrid architecture was the best in this setting as well, showing that it can consistently do better when classification tasks are harder.

Interestingly, GCNN architecture was outperformed by the baseline SVM architecture by a relatively high margin of 0.02. It proves that baseline architecture is able to achieve considerably good accuracy for the task. However, as it is seen from the data, introducing graph models may improve the performance by 0.01 point.

5.4.2. Sparse Author Fingerprint

We also tested the impact of using sparse author fingerprints for classifying authors by their nationality. We wanted to see the impact of reduced dimensionality on more complex task. We can see that the accuracy drop is considerably higher than with more simple tasks (1 vs 1 and 1 vs all) - 0.19 (comparing GCNN with sparse and normal author fingerprint). Also, GCNN with sparse author fingerprint was outperformed by baseline model 5.9. We believe that it happened due to dimensionality being too low for this more complex task.

5.5. Gender and nationality

To further assess the GNNs for authorship attribution we decided to test algorithms in multi-label classification problem. The dataset was split into 4 parts by author's gender and nationality. Since the group of American female writers was considerably smaller than other groups, the dataset was heavily imbalanced. For GNNs we tackled this problem by assigning weights for *Cross-entropy loss*. Meaning, that the loss function would penalize misclassification of smaller group more than misclassification of a larger group. However, we could not use the same strategy for baseline SVM model. We did include the results of SVM in this section, but we understand that it does not represent a real picture because the dataset is heavily imbalanced.

We expected that introducing multi-label problem would increase the complexity of the task. Thus, we hypothesized that the performance would decrease, however, we were interested in comparing the performance of binary classification tasks with multi-label classification task using GNN architectures.

5.5.1. Results

The results are depicted in Table 5.5. As expected, we saw considerable drop in classification performance. Compared with the classification by gender, the difference in accuracy between the best performing architectures was 0.09; compared with the classification by nationality - 0.05. This can be explained by the fact that classifier has to make a more complex decision boundary for multi-label problem, compared with the binary problem.

Furthermore, we can see the same trend of hybrid Edge-GCNN architecture outperforming other architectures. However, in this case the difference is more dramatic (0.03 compared to GCNN) than in binary problems (0.01 compared to EdgeNet in classification by gender and classification by nationality). This fact proves the effectiveness of combining two approaches - more complex EdgeNet architecture that is able to learn at node level and more shallow GCNN architecture that enables weight sharing and reduces the complexity.

Important to point out that EdgeNet in multi-label classification case did considerably worse than other GNN architectures. The reason could be that in multi-label classification the size of a group becomes even smaller, thus more complex EdgeNet architecture does not get enough data for it's complexity. We leave testing EdgeNets with bigger dataset in multi-label classification task for further research.

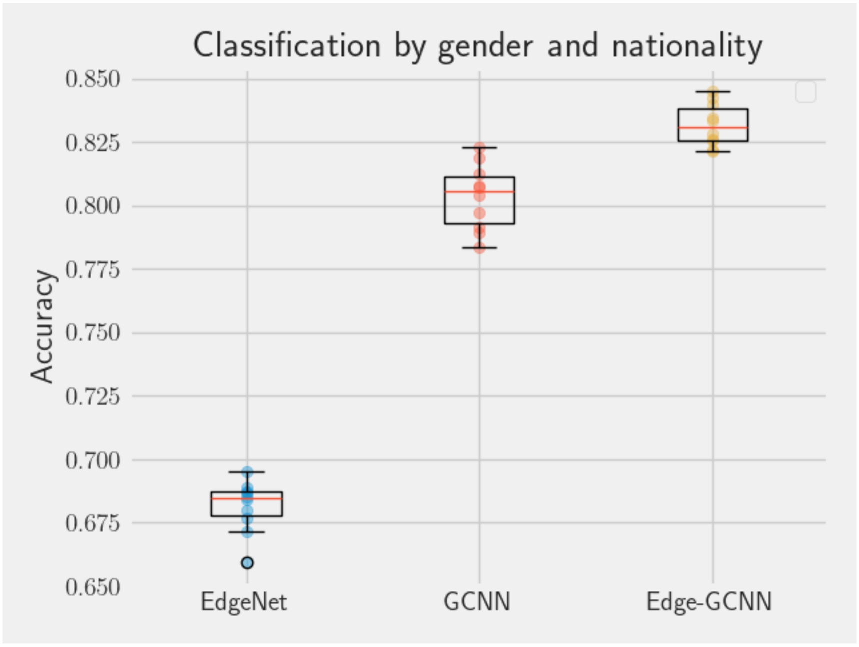


Figure 5.10: Accuracy of classification by gender and nationality of various GNN architectures.

Table 5.5: Results of classification by authors' gender and nationality.

	EdgeNet	GCNN	Edge-GCNN	SVM
Accuracy	0.687	0.80	0.83	0.83
Std	0.0097	0.01	0.008	0.003
F1	0.49	0.79	0.81	0.81

5.6. Discussion

In this chapter, we presented the experimental results of the thesis and drew conclusions.

Firstly, in Section 5.1 we have experimented with classifying authors against all others. It was one of the baseline experiments where we experimented with different hyperparameters (number of convolutional features and filter taps) and different architecture depth (2 layer GCNN). In this experiment, 1 layer GCNN proved to perform the best, outperforming baseline model and more complex EdgeNet architecture. Also, we found that certain authors are easier to classify and has only 0.04 error rate, however, others can have up to 0.13 error rate. In addition, we used sparse author fingerprints for this task. We found that with significantly reduced dimensionality we are able to achieve comparable performance to standard author fingerprint. Also, on average GCNNs performed better with the task than baseline model.

Secondly, in Section 5.2 we have experimented with slightly easier problem of classifying author against other author. We classified authors that are the most similar to each other (calculating average Euclidean distance between graph signals). We noticed significant accuracy improvement (compared to 1 vs all classification). In addition, GCNNs were also performing the best in this problem outperforming baseline and EdgeNet architectures. Furthermore, we used sparse author fingerprints and found similar results to 1 vs all classification, where we were able to achieve comparable classification performance with significantly reduced dimensionality. Also, GCNNs outperformed baseline architecture and proved to be useful for this task.

In addition we experimented with more complex problems in Sections 5.3, 5.4 and 5.5 (gender, nationality and gender and nationality classification respectively). On harder problems, we found out that the hybrid architecture of EdgeNet and GCNN consistently performed the best, outperforming baseline model, GCNN and EdgeNet. Also, with the exception of classification by gender and nationality, we noticed that more complex GCNN architectures tend to perform better with more complex tasks.

6 CONCLUSION

In this chapter, we conclude the research carried out in this thesis. Firstly, we summarize the thesis in Chapter 6.1. Then, we provide answers to the research questions posed in Chapter 1. Finally, we provide possible future work directions in Chapter 6.3.

6.1. THESIS SUMMARY

In this thesis, we have experimented with authorship attribution problem using emerging tool working on non-regular data - Graph Neural Networks. We used GNNs to improve on the state of the art of authorship attribution problem and extract sparse author fingerprints that enable us to attribute authors with relatively high accuracy using only 5 words. Also, we have used the authorship attribution problem in order to explore the particularities of different GNN architectures.

In Chapter 1, we introduced the problem and the context of the field and posed research questions that we wanted to answer in this work. In Chapter 2, we reviewed literature which relates to our research. Specifically, we reviewed state-of-the-art in the field of GNNs and latest research on authorship attribution problem. In Chapter 3 we introduced the basis on which this work is built

- Word Adjacency Networks as a tool to extract graphs from authors' text excerpts
- Graph Convolutional Neural Network as a baseline model which was used to classify texts
- Edgenet network as a tool to learn new graphs and extract sparse fingerprints from WAN graphs.

In Chapter 4, we discussed our main contribution, the method of extracting sparse author fingerprints, defined experiments which were carried out to explore the features of GNNs and improve on authorship attribution problem. Also, we defined architectures, parameters and dataset splits that were used for experiments.

Finally, in Chapter 5, we provided numerical results for the experiments. We compared the performance of different GNN architectures and configurations. We concluded that GNNs could improve on the state-of-the-art of authorship attribution problem. Also, GCNNs were better suited for less complex tasks (e.g. 1 vs 1 or 1 vs all classification); however, more complex architectures such as EdgeNet (or hybrid) did better on more complex tasks (classification by gender or nationality).

Besides, we tested the performance of sparse author fingerprints and compared them to WAN fingerprints. We noticed that even with a significant amount of reduction in dimensionality, we could retain relatively high performance and improve on training times. Also, when working with sparse fingerprints, GNNs performance was significantly better than SVM.

6.2. ANSWERS TO RESEARCH QUESTIONS

In this section, we provide answers to research questions posed in chapter 1 based on carried out research in this thesis.

RQ1. How different GNNs can be used to improve on the state-of-the-art of authorship attribution?

RQ2. How can we use authorship attribution problem to gain a better understanding of GNNs?

To address the first two research question, in Chapter 4 we introduced a set of experiments which we will use for performance testing (one vs one, one vs all, gender, nationality, gender and nationality) and a set of architectures that were used for testing. Also, we described the strategy to find the most optimal hyperparameters of GNNs. In Chapter 5, we described test results and confirmed that in all tasks, GNNs were superior to SVMs in terms of classification accuracy. GCNNs were superior in easier tasks (one vs one, one vs all) and EdgeNets were better in more complex tasks (gender, nationality and gender and nationality), which means that GNNs can make use of underlying topology.

Furthermore, we noticed some additional findings. GCNNs usually performs better with higher amount of convolutional features, and EdgeNets usually needs hyperparameter re-tuning. This could be due to the increased complexity of EdgeNets. Furthermore, we could not find a relationship between increased depth and increased performance. Meaning that for authorship attribution using WANs depth of GNNs is not a deciding factor. Also, hybrid EdgeNet-GCNN architecture outperformed others and could be used for harder tasks.

To address the final research question:

RQ3. How can we use sparse author fingerprints for learning and gaining a better understanding of what is being learned by GNNs?

We introduced the technique (Chapter 4) to learn a new graph from WAN graph using EdgeNet and extract a sparse fingerprint from it. Further, we experimented with sparse author fingerprint on the same experiments and looked for an optimal reduction in dimensions with regards to classification accuracy. We established that by using sparse author fingerprints, we could retain 95% of maximal accuracy and achieve a considerable reduction in dimensions. In a case of 1 vs 1 classification reduction in dimensions even resulted in improved classification accuracy.

Furthermore, we used sparse fingerprints to visualize the relationship between the most critical function words. Since the input dimensions were reduced significantly, and we used graph data, we were able to visualize sub-graphs of WAN fingerprints with the essential words and connections for a particular author. Also, with the help of sparse fingerprints, it is possible to classify author against others only using 3 words and a random text excerpt of 1000 words.

6.3. FUTURE WORK

To conclude the work done in the thesis, we discuss possible future work. It can be categorized into two categories: meaningful sparse author fingerprints and application of sparse fingerprints in other fields.

Meaningful sparse author fingerprints The first possible direction is to look for other methods of extracting meaningful sparse author fingerprints. We could utilize Graph Attention Networks (GATs) [23] and Graph Convolutional Attention Networks (GCATs) [10]. These networks utilize combining neighbouring information through attention mechanisms and allow implicitly assigning different weights to different neighbours. It can further be used to learn a graph from the input signal and extract a sparse fingerprint.

Also, we can improve on currently used method utilizing EdgeNets by setting the L_1 penalty norm, which will penalize network for assigning weights for more nodes.

Sparse fingerprints in different applications The same method of using sparse fingerprints may be applied in different setups and tasks (not only authorship attribution). Possible applications could be: identifying hyperlinks in blogs, bonds in molecules, meaningful wireless connections in robot coordination.

Sparse fingerprints could also be applied in fields of chemistry, biology and physics. Molecules can be conveniently converted into graphs by seeing atoms as nodes and bonds as edges. Some works used GNNs as feature extractors for scent molecules [24]. The features extracted using GNNs achieved the state of the art performance compared to the expert-crafted general features. Others used GCNNs for extracting molecular fingerprints [25]. Here our method of extracting fingerprints can be employed and compared to the state of the art techniques.

Different datasets Authorship dataset provided a sufficient amount of data and experiments to start researching and gaining a deeper understanding of GNNs. However, in some cases, we were limited with dataset size (e.g., gender-nationality classification). For further understanding of GNNs we would like to employ a bigger dataset.

6.4. BROAD IMPACTS

GNNs, specifically GCNN and EdgeNet, are powerful tools to learn graphs. With the sparse author fingerprints extracted using EdgeNets, we can attribute authors with relatively high confidence from 3-10 function words from a random text excerpt of 1000 words. More shockingly, using the same setup, we can identify the gender of the author from the text. Also, the ability to combine information from the nearest and furthest neighbours allows GCNN to improve on baseline models in authorship attribution and possibly can be applied to similar spheres.

The critical sphere where the research can be applied is plagiarism detection [26]. With the expansion of published research [27] and other intellectual information (in forms of blogs, articles or computer code) the fast and reliable method for authorship attribution may revolutionize the community that generates intellectual content in the form of the text. Sparsifying author fingerprints and still retaining stylistic features could become a fast and reliable way to identify authors. Furthermore, reducing the dimensions of the input data while retaining the most significant features can be beneficial for many research directions. Reducing the dimensions helps to deal with the curse of dimensionality [28] as it reduces the dimensions of the input data, thus, reduces the amount of data needed for training. The one field of science dealing with over-abundance of high-dimensional data is Biochemistry. Thus, sparsifying the small-molecule compounds may boost the predictive power when inferring their properties and activities [29]. Inferred properties can be used in many applications - drug discovery, molecule fingerprinting or reducing the amount of data stored in molecule databases.

REFERENCES

[1] Y. Sari, M. Stevenson, and A. Vlachos, *Topic or style? exploring the most useful features for authorship attribution,* in *Proceedings of the 27th International Conference on Computational Linguistics* (Association for Computational Linguistics, Santa Fe, New Mexico, USA, 2018) pp. 343–353.

[2] S. Segarra, M. Eisen, and A. Ribeiro, *Authorship attribution through function word adjacency networks,* IEEE Transactions on Signal Processing **63**, 5464–5478 (2015).

[3] H. van Halteren, H. Baayen, F. Tweedie, M. Haverkort, and A. Neijt, *New machine learning methods demonstrate the existence of a human stylome,* Journal of Quantitative Linguistics **12**, 65 (2005), https://doi.org/10.1080/09296170500055350 .

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

[5] C. D. Hofer, R. Kwitt, M. Niethammer, and A. Uhl, *Deep learning with topological signatures,* CoRR **abs/1707.04041** (2017), arXiv:1707.04041 .

[6] F. Gama, E. Isufi, G. Leus, and A. Ribeiro, *From graph filters to graph neural networks,* (2020), arXiv:2003.03777 [cs.LG] .

[7] A. Dhillon and G. K. Verma, *Convolutional neural network: a review of models, methodologies and applications to object detection,* Progress in Artificial Intelligence **9**, 85 (2020).

[8] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, *The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains,* IEEE Signal Processing Magazine **30**, 83 (2013).

[9] F. Gama, A. G. Marques, G. Leus, and A. Ribeiro, *Convolutional neural network architectures for signals supported on graphs,* IEEE Transactions on Signal Processing **67**, 1034–1049 (2019).

[10] E. Isufi, F. Gama, and A. Ribeiro, *Edgenets:edge varying graph neural networks,* (2020), arXiv:2001.07620 [cs.LG] .

[11] Y. Zhao and J. Zobel, *Searching with style: Authorship attribution in classic literature.* (2007) pp. 59–68.

[12] E. Stamatatos, *A survey of modern authorship attribution methods,* Journal of the American Society for Information Science and Technology **60**, 538 (2009), https://onlinelibrary.wiley.com/doi/pdf/10.1002/asi.21001 .

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

[14] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, *Geometric deep learning: Going beyond euclidean data,* IEEE Signal Processing Magazine **34**, 18–42 (2017).

[15] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, *How powerful are graph neural networks?* (2018), arXiv:1810.00826 [cs.LG] .

[16] F. Wu, T. Zhang, A. H. de Souza Jr., C. Fifty, T. Yu, and K. Q. Weinberger, *Simplifying graph convolutional networks,* (2019), arXiv:1902.07153 [cs.LG] .

[17] M. Henaff, J. Bruna, and Y. LeCun, *Deep convolutional networks on graph-structured data,* (2015), arXiv:1506.05163 [cs.LG] .

[18] F. Gama, J. Bruna, and A. Ribeiro, *Stability properties of graph neural networks,* (2019), arXiv:1905.04497 [cs.LG] .

[19] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, *A comprehensive survey on graph neural networks,* IEEE Transactions on Neural Networks and Learning Systems , 1–21 (2020).

[20] R. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, *Gnnexplainer: Generating explanations for graph neural networks,* Advances in neural information processing systems **32**, 9240 (2019).

[21] Q. Huang, M. Yamada, Y. Tian, D. Singh, D. Yin, and Y. Chang, *Graphlime: Local interpretable model explanations for graph neural networks,* ArXiv **abs/2001.06216** (2020).

[22] M. Tulio Ribeiro, S. Singh, and C. Guestrin, *" why should i trust you?": Explaining the predictions of any classifier,* arXiv , arXiv (2016).

[23] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, *Graph attention networks,* (2017), arXiv:1710.10903 [stat.ML] .

[24] B. Sanchez-Lengeling, J. N. Wei, B. K. Lee, R. C. Gerkin, A. Aspuru-Guzik, and A. B. Wiltschko, *Machine learning for scent: Learning generalizable perceptual representations of small molecules,* (2019), arXiv:1910.10685 [stat.ML] .

[25] D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, *Convolutional networks on graphs for learning molecular fingerprints,* (2015), arXiv:1509.09292 [cs.LG] .

[26] T. Foltýnek, N. Meuschke, and B. Gipp, *Academic plagiarism detection: A systematic literature review,* ACM Comput. Surv. **52** (2019), 10.1145/3345317.

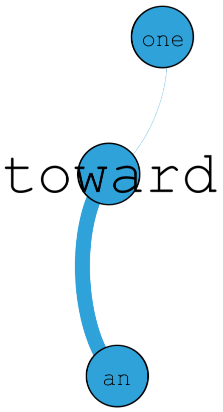
[27] M. Ware and M. Mabe, *The stm report: An overview of scientific and scholarly journal publishing,* (2015).

[28] C. Taylor, *Applications Of Dynamic Programming To Agricultural Decision Problems* (CRC Press, 2019).

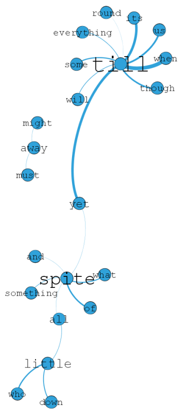
[29] H. Altae-Tran, B. Ramsundar, A. S. Pappu, and V. Pande, *Low data drug discovery with one-shot learning*, ACS Central Science **3**, 283 (2017), pMID: 28470045, https://doi.org/10.1021/acscentsci.6b00367 .

A

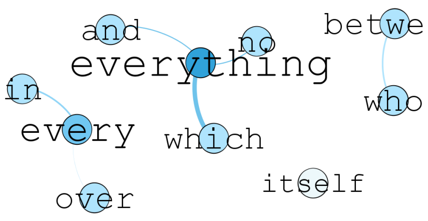
Visualisation of sparse author fingerprints



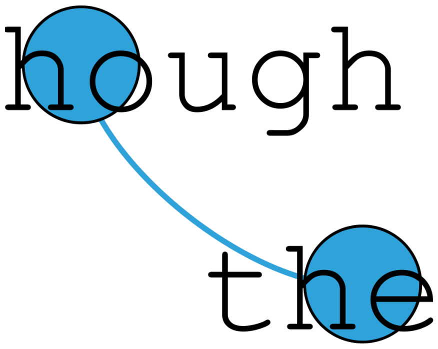
(a) Sparse fingerprint visualization of Abbot.



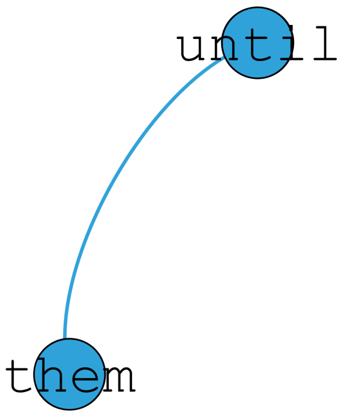
(b) Sparse fingerprint visualization of Alcott.



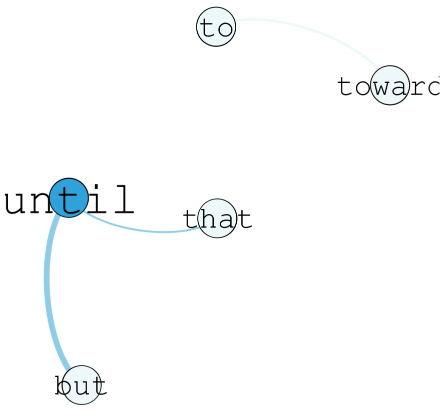
(c) Sparse fingerprint visualization of Alger.



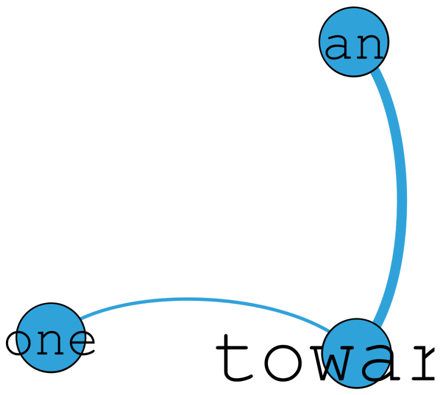
(d) Sparse fingerprint visualization of Allen.



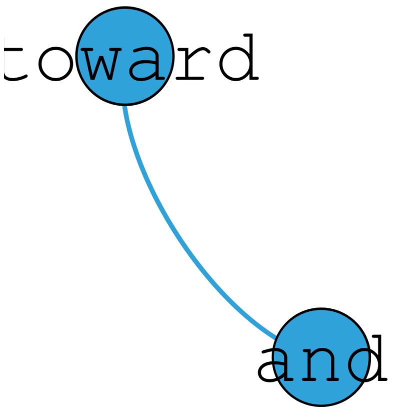
(e) Sparse fingerprint visualization of Austen.



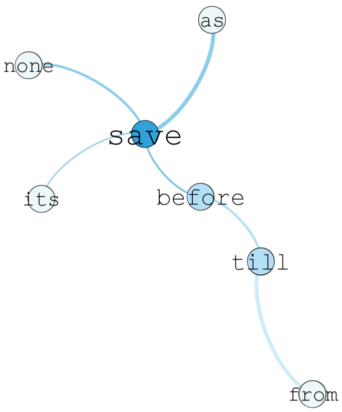
(f) Sparse fingerprint visualization of Bronte.



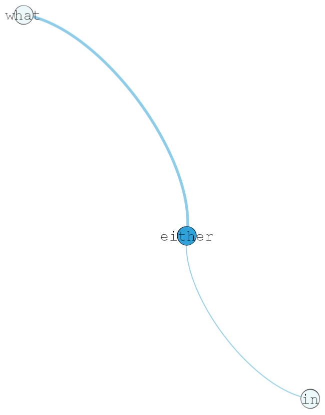
(g) Sparse fingerprint visualization of Cooper.



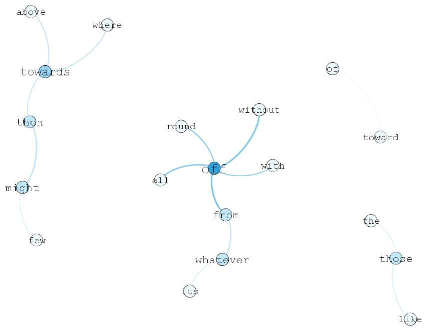
(h) Sparse fingerprint visualization of Dickens.



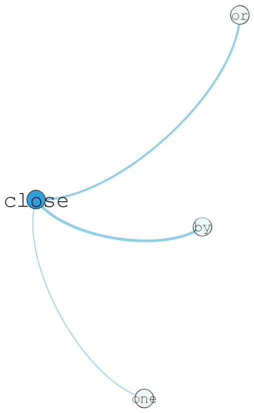
(i) Sparse fingerprint visualization of Doyle.



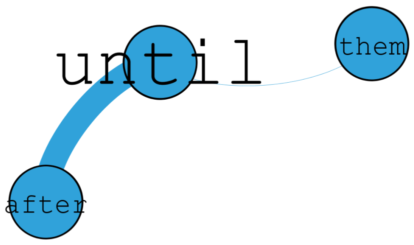
(j) Sparse fingerprint visualization of Garland.



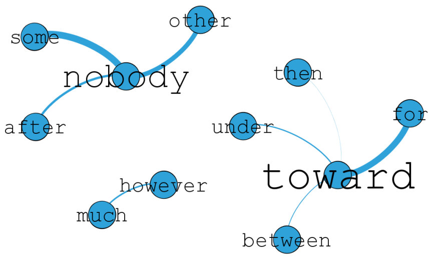
(k) Sparse fingerprint visualization of Hawthorne.



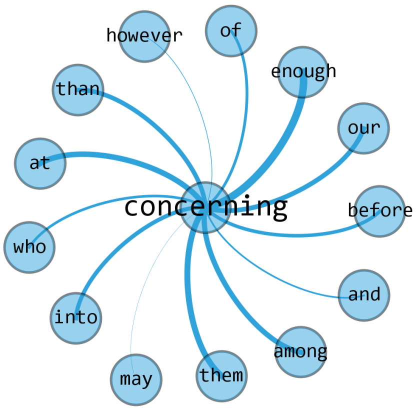
(l) Sparse fingerprint visualization of Irving.



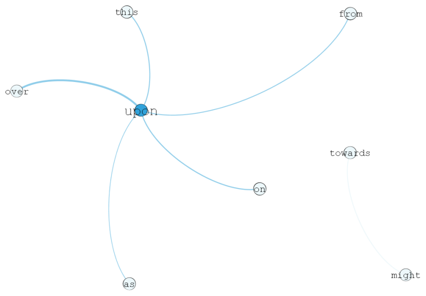
(m) Sparse fingerprint visualization of James.



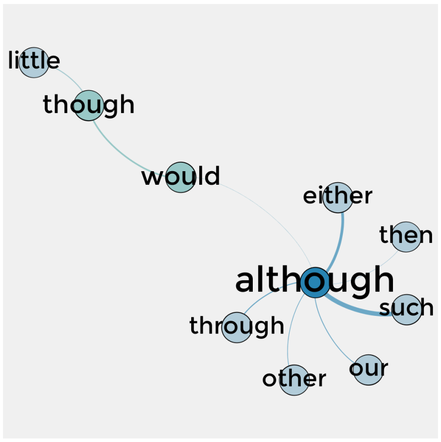
(n) Sparse fingerprint visualization of Jewett.



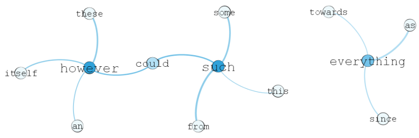
(o) Sparse fingerprint visualization of Melville.



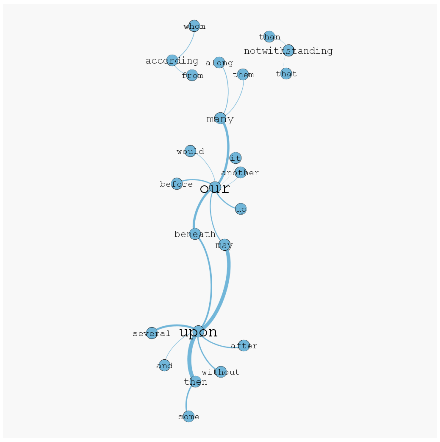
(p) Sparse fingerprint visualization of Page.



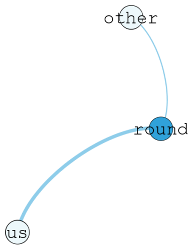
(q) Sparse fingerprint visualization of Poe.



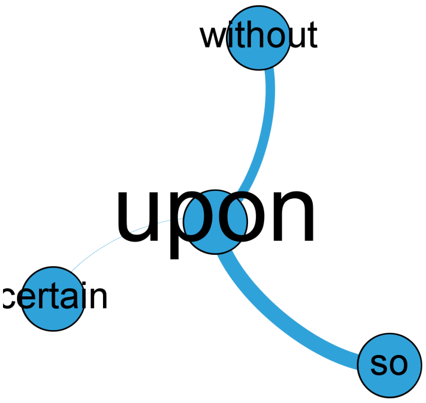
(r) Sparse fingerprint visualization of Stevenson.



(s) Sparse fingerprint visualization of Thoreau.



(t) Sparse fingerprint visualization of Twain.



(u) Sparse fingerprint visualization of Wharton.

Figure A.1: Visualizations of sparse author fingerprints.

B

Graph Centrality Measures of the dataset

Additional graph centrality measures of the dataset.

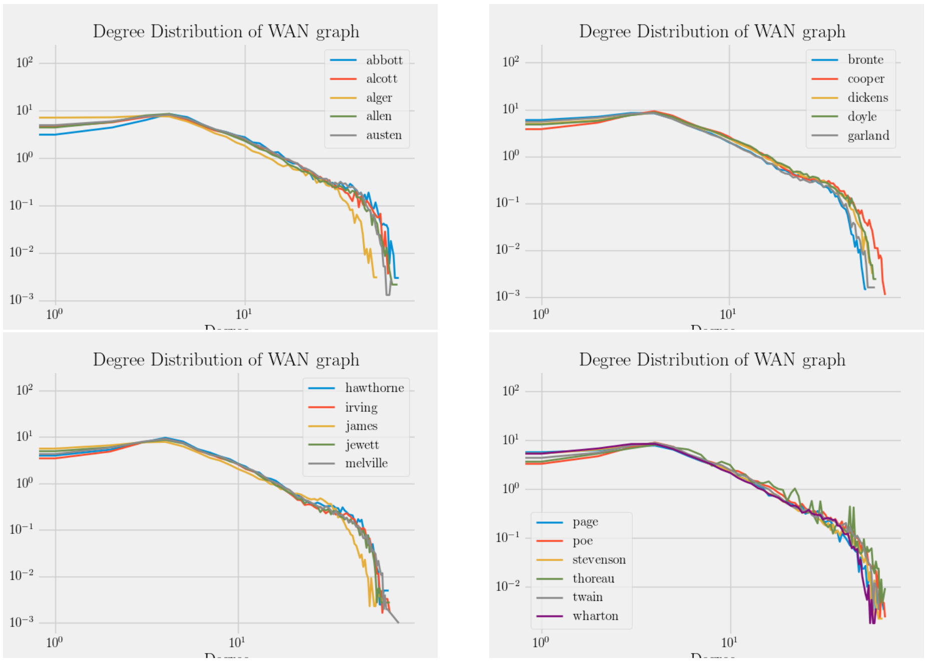


Figure B.1: Degree distribution of WAN fingerprint for each author in the dataset. Authors are sorted in alphabetical order.