AUTHORSHIP ATTRIBUTION USING WRITTEN AND READ DOCUMENTS

by

Afsan Abdulali Gujarati

Submitted in partial fulfillment of the requirements
for the degree of Master of Electronic Commerce

at

Dalhousie University
Halifax, Nova Scotia
June 2019

Table of Contents

Abstract

In Authorship Attribution (AA), a task of identifying the author on an unseen document, it is often hard to obtain large amounts of training text written by an author. Our research analysis the influence of the size of training data and proposes a novel alternative of using the documents read by the authors for the AA task. Although it becomes significantly difficult to identify the author of an unseen document with less written data, the classification performance can be drastically improved by using the documents read by the author. The Support Vector Machine method outperformed all the classifiers in the presence of the read documents with an average accuracy of 94.35%, a 23.57% increase after the addition of the read documents. It was found through the feature analysis that there exists a semantic similarity between the written and the read documents that played an important role in improved performance.

List of Important Abbreviations

AA **A**uthorship **A**ttribution
The task of identifying the author of an unseen document.

TS **T**raining **S**ize
This is the first part of the research to analyze the influence of the size of training data in authorship attribution using the written documents.

RD **R**ead **D**ocuments
This is the second part of the research to analyze the performance of the classifiers with and without the presence of the read documents.

TSL35 **T**raining **S**ize on **L**arger dataset with **35** authors
This is the first set of experiments in the first part of the research (TS) to analyze the performance of the classifiers when training them with larger dataset of only written documents by 35 authors (7 upto 64 wdpa).

TSS35 **T**raining **S**ize on **S**maller dataset with **35** authors
This is the second set of experiments in the first part of the research (TS) to analyze the performance of the classifiers when training them with smaller dataset of only written documents by 35 authors (4, 5, and 6 wdpa).

TSS20 **T**raining **S**ize on **S**maller dataset with **20** authors
This is the first set of experiments in the second part of the research (RD) to analyze the performance of the classifiers when training them with smaller dataset of only written documents by 20 authors (4, 5, and 6 wdpa).

RD20 **R**ead **D**ataset with **20** authors
This is the second set of experiments in the second part of the research (RD) to analyze the performance of the classifiers when training them with smaller dataset with written (4, 5, and 6 wdpa) and read documents (40, 50, and 60 rdpa) by 20 authors.

SVM **S**upport **V**ector **M**achine
Supervised machine learning classifier.

MNB **M**ultinomial **N**aïve **B**ayes
Supervised machine learning classifier.

CNG **C**ommon **N**-**G**rams
Supervised machine learning classifier.

WDpA **W**ritten **D**ocuments **p**er **A**uthor
Number of documents written by each individual author that are included in the experiment.

RDpA **R**ead **D**ocuments **p**er **A**uthor
Number of documents read by each individual author that are included in the experiment.

Acknowledgements

Foremost, I would like to express my sincere gratitude to *Tapajyoti (Tukan) Das*, whose constant motivation, support and advice mentored me to maneuver in the right direction. He deserves the credit for helping me select the research problem and prompting me to think more analytically about my approach. I would also like to thank *Colin Conrad*, who believed in me from the very beginning and convinced me to do a Thesis as a part of my program.

I would like to indefinitely thank my supervisor *Dr. Vlado Keselj*, who taught me the necessary art of controlling variables in research and keeping things simple. Most importantly, he transformed my unmethodical approach towards a systematic approach required for research. I appreciate his patience, wisdom and constant nudge to help me strive for better.

I would also like to thank my labmates at the Faculty of Computer Science, who supported me and peacefully listened to my work whenever required. *Dijana Kosmajac* with her ability to think critically assisted me to look into minor details I would have clearly missed out. *Manav Sharma* understood the problem I was working and pointed out an error during a very critical period of the research. I would also like to thank *Stacey Taylor* for carefully listening to my rambling between the tireless work and *Yichuan Xu* for his never-ending supply of snacks.

Finally, I would like to thank my friends and family, especially my brother Anis Gujrati and my friend Saifina Maredia who made sure I didn't lose my sanity throughout the whole process.

Chapter 1

Introduction

1.1 Introduction to Authorship Attribution

Authorship Attribution (AA) is a specialized field in Information Retrieval which was initially devised to identify the author of an unseen document from a set of known authors (closed-class classification). However, the field later evolved to include more complex task such as Authorship Verification (open-class classification) and Author Profiling [14]. In Authorship Verification (open-class classification), the task is not only to identify the author of the unseen document, but also to verify if the author belongs to the set of authors [14]. Whereas in closed-class classification it is known that the author of a document is from the given set of authors. The Author Profiling task is identifying the characteristic of the author such as gender, age, country of origin and so on, based on their written documents [32].

AA is not simply restricted to assigning the author to an unseen document, but its application extends in the domain of fake news detection, plagiarism, identity theft, crime investigation and so on [14]. Especially with the increasing size of the social network, all of these problems have become a real threat. In the domain of digital investigation, a text document (reviews, reports, articles and so on) serves as an evidence to find the owner or the creator of the evidence [2]. AA is also used to identify the author of a programming script, which is helpful to find the creator of malicious software which is a potential threat for the security of the system [46]. In forensic investigations, AA is used to verify the authenticity of a suicide note or any other written piece of evidence. The creator of such threats are getting more informed, which makes it challenging to find hard evidence against them. Therefore, it is important to build a more intelligent system to characterize and identify the creator of such threats.

1.2 Research Motivation

It is estimated that by the year 2020, there will be 40 trillion gigabytes of data and around 90% of total data that exists today was generated in the last two years [19]. Users spend 33% of the total time on social media [29]. There has been a decline in the consumption of content through the traditional media sources (such as cable television, radio, newspapers) and increase in the consumption through digital sources [27].

An article shared by an award winning journalist, Chad Buleen, discusses how the rate at which content consumed is almost twice the rate at which it is created [5]. The Generation Z (or Gen Z — the demographic cohort following the millennial who were born between 1995 and 2015) in a survey said that 73% of them prefers to consume content over creating it on Snapchat. On Instagram, 70% would simply consume content while 39% creates. The Facebook usage dropped among Gen Z. On the other hand, millennials consume more content on Instagram (58%) compared to Snapchat (53%). The consumption on Facebook is 48% by millennials, 14% higher than Gen Z. The study concludes that the content consumption is higher than content creation by the young population.

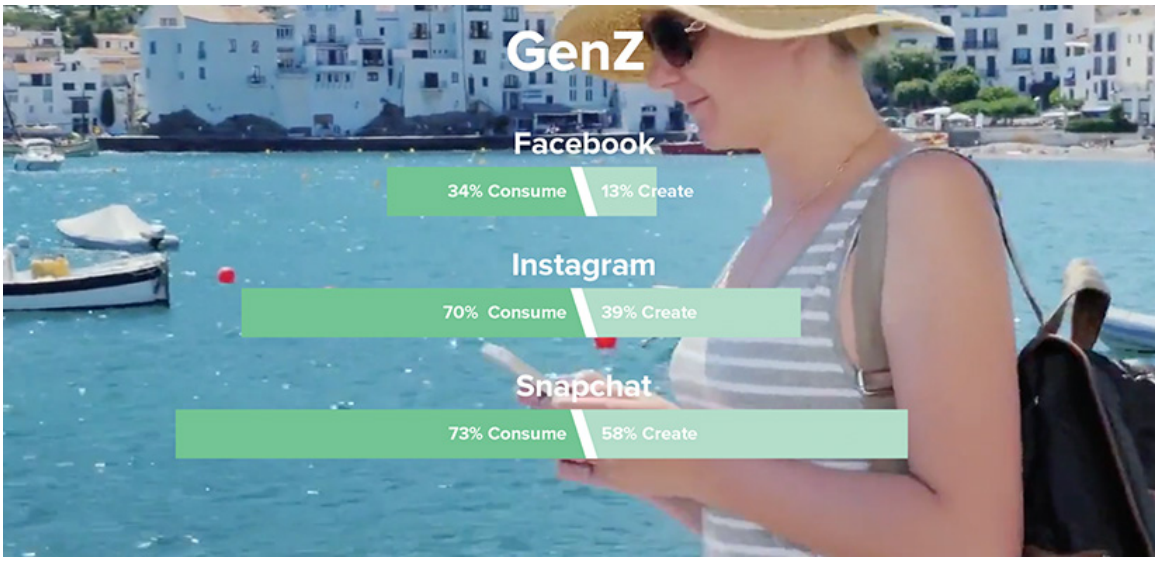


Figure 1.1: **Content consumption VS content creation by Gen Z.** [5] The rate at which content consumed is almost twice the rate at which it is created for the three leading social media platforms — Snapchat, Instagram and Facebook among Generation Z.

This rapid rate of content generation requires stricter security measures for protecting users interest. It is practically insurmountable to manually monitor the false contents and find their creator. Therefore there is a need for automated techniques to filter out the false contents and also find the creator (or author) responsible for its generation. Identification of the author based on their written content is one of the foundational problem addressed by AA.

The problem of AA can be solved through the application of machine learning techniques. However, in the field of machine learning, the performance of the model is as good as the input data. It works on the same principle as GIGO — Garbage In Garbage Out. The confidence of the system is proportional to the amount and quality of input data. Therefore, higher amount of quality data is required to improve the performance of a machine learning model. Considering the statistics of content creation and consumption, it is understood that more content is consumed than created. Therefore, the availability of greater consumed data can be coupled with the created data to improve the performance of the machine learning models in AA task.

Previous research in AA has significantly improved the results with fewer authors and bigger training datasets. However, a combination of smaller training datasets and many authors still pose a problem in AA. In order to understand the behavior of smaller datasets, it is important to analyze the influence of size of the training data in AA task. Also, in order to improve the performance on smaller dataset, the previous research has simply focused on feature engineering; i.e., finding key characteristics that differentiate between authors or augmenting training data using the same written documents. These approaches would be discussed in more detailed in the Chapter 2.

Our research plans to first analyze the influence of the size of training data is analyzed followed by analyzing the performance of the classifiers when using the content consumed by authors (read documents) along with the content created by authors (written documents) for the training purpose.

1.3 Research Objectives

Our research is divided into two parts, **T**raining **S**ize (TS) and **R**ead **D**ocuments (RD) 1.2. In the first part, the influence of the size of training data using the written documents (TS) is analyzed, where the focus is on the performance of the classifiers while increasing the number of training documents. In the second part, the performance of the classifiers with and without the presence of the read documents (RD) is analyzed, where the focus is to identify if the documents read by the authors can assist in improving the performance of the Authorship Attribution (AA) task in limited data situation.

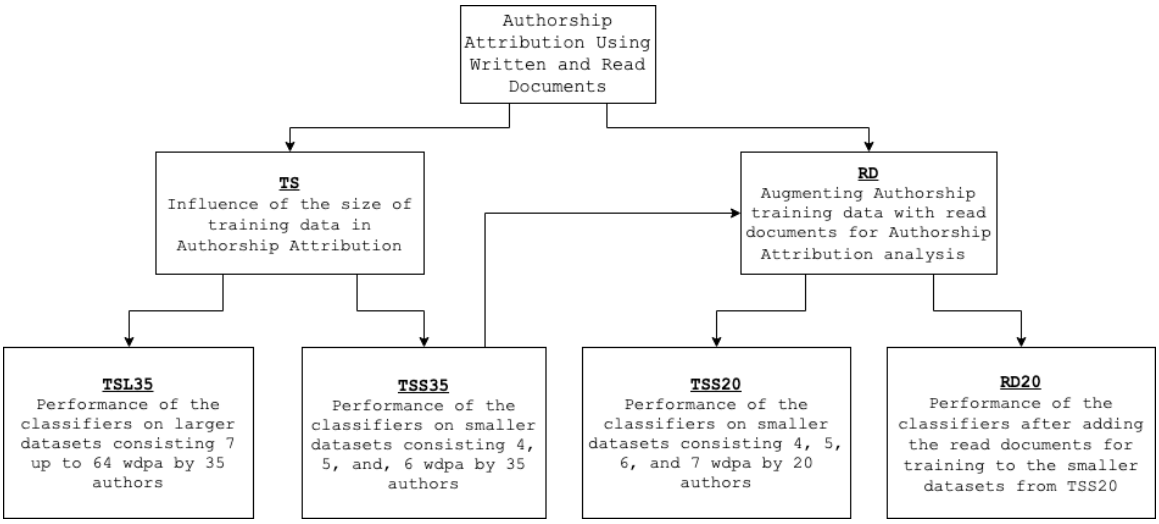


Figure 1.2: **Summary of research objectives.** The figure summarizes the overall structure of our research. The research is divided into two parts TS and RD where RD extends the work done in TS. Both TS and RD are further divided into two sections, where TS focuses on the influence of the size of training data and RD focuses on the performance of the classifiers with and without the presence of the read documents.

The first part of the research (TS — **T**raining **S**ize) is subdivided into **T**raining **S**ize on **L**arger dataset with **35** authors (TSL35) and **T**raining **S**ize on **S**maller dataset with **35** authors (TSS35) 1.2. In the first section (TSL35), the objective is to analyze the performance of the classifiers when training them with larger dataset of only written documents by 35 authors (7 to 64 Written Documents per Author (WDpA)). This analysis would measure the performance of the each individual classifier while increasing the number of training documents. It would also help to choose a classifier for practical application depending upon the number of documents available for

training. In the second section (TSS35), the objective is to focus on smaller datasets, because compared to other text classification problems like spam detection and sentiment analysis, AA has a problem with a lack of training data. This is mostly because although authors write *'a lot'*, it is still relatively small compared to the data produced for other text classification problems. Therefore, the performance of the classifiers when training with only 4, 5 and 6 Written Documents per Author (WDpA) is analyzed.

The second part of the research (RD — **R**ead **D**ocuments) was subdivided into **T**raining **S**ize on **S**maller dataset with **20** authors (TSS20) and **R**ead **D**ataset with **20** authors (RD20) 1.2. Due to data constraint, the experiments in RD consists of only 20 authors compared to 35 in TS. In the first section (TSS20), the experiments similar to TSS35 were repeated by considering only 4, 5, 6, and 7 WDpA by 20 authors for training. In the second section (RD20), the results of TSS20 were benchmarked against RD20 by including ten times the read documents per author along with their written documents from the previous section for training. The improvement in performance of the classifier after the use of the read documents would set a new direction for AA in limited data situation.

1.4 Thesis Overview

This section provides an overview of our thesis. In chapter 2, reviews the work done in previous research to analyze the influence of the size of training data and also to tackle the problem with limited data. The most renowned work done by Mosteller and Wallace on the Federalist Papers and some recent work based on machine learning techniques for AA task on Bengali text [24, 30] are summarized. The review also discusses some well known AA methods that used character n-grams for language independent AA task [15, 47]. In order to tackle the limited data problem, Qian *et al* suggested a tri-training algorithm that used character, syntactic and lexical features [31]. Luyckx and Daelemans analyzed the influence of increasing the number of authors and training data for up to 145 authors using Support Vector Machine (SVM) [18]. In order to statistically differentiate between the performance of each individual classifier, different statistical methods were reviewed and the Friendman's test was used for our research [7].

In chapter 3, the methodology used to conduct the experiments in our research is discussed. Also, a framework is shared which was used to collect the written documents and read documents by business and investing reporters and columnists. The chapter also discusses the basic structure of the dataset, classifiers used, preprocessing carried out on the data and also the experimental setup. Supervised machine learning algorithm such as Support Vector Machine SVM, Multinomial Naïve Bayes (MNB) and Common N-Grams (CNG) (d_0, d_1, d_2, and SPI) methods were used for the AA task. In the first part of our research (TS — **T**raining **S**ize), the experiments were conducted only on the written documents by 35 authors to analyze the influence of the size of training data on the performance of individual classifier. However, in the second part of the research (RD — **R**ead **D**ocuments), the experiments were conducted on the written documents with and without the presence of read documents by 20 authors. While the objective of the TS was to analyze the influence of the size of training data in AA, the objective of RD was to analyze the influence of the read documents in AA. In these experiments, the training data gradually increased and the optimal training parameters were selected for each classifier to calculate the test accuracy.

In chapter 4, the results are discussed and an in-depth analysis is provided of the best performing classifier for both TS and RD. The TS experiments were subdivided into TSL35 and TSS35. TSL35 looked into the documents ranging from 7 up to 64 Written Documents per Author (WDpA) for training and in TSS35 the smaller dataset were considered consisting of 4, 5, and 6 WDpA for training. The performances of individual classifier was compared by looking into their strengths and weaknesses. The performance per author is also discussed, along with their strength and weaknesses. The RD experiments were also subdivided into two sets. First, the experiments only considered the written documents by the 20 selected authors (TSS20). This is the same set of experiments as TSS35, however, it only considered 20 instead of 35 authors due to data constraint. The results of this experiments were benchmark against the results of the experiments with both written and read documents (RD20). The performance of the classifiers before and after the addition of the read documents was analyzed by comparing their accuracy scores. The performance of individual classifier was also analyzed for RD. For in-depth analysis, the features

were compared to understand the significance of read documents and misclassification in case of some authors.

Chapter 5 concludes our research by discussing the limitations, providing a summary and future work of our research.

Chapter 2

Related Work

In this chapter the previous work in Authorship Attribution (AA) relevant to our research is discussed. The first section discusses various techniques and approaches used to tackle the problem of limited data in AA. The review summarizes the early approaches as well as the recent supervised and unsupervised machine learning approaches. The second section of the chapter discusses the use of statistical significance test for model selection in text classification problem.

2.1 Authorship Attribution Techniques

The previous work was divided into three subsections for a systematic review. The first and second subsections compare the early approaches, with the contemporary machine learning approaches relevant to our research. The third section summarizes one of the highly referred approaches in Authorship Attribution, Character N-Gram (CNG) method [15].

2.1.1 Early Authorship Attribution approaches

The earliest statistical approach in Authorship Attribution (AA) proposed word length as a distinct feature to differentiate between authors was suggested by Mendenhall and Corwin in 1887 [14, 23]. Later, other statistical features were proposed such as average sentence length, variation in part of speech, and measures of vocabulary richness among others [14]. However, none of these methods provided a reliable means for the AA task. One of the earlier well-known approaches was proposed by Mosteller and Wallace on the Federalist Papers suggested in 1963 [24].

Mosteller and Wallace had applied two different approaches to assign the author to the disputed paper. The first method used was the classical approach wherein they applied a set of words over a discriminant function using different weights for the words depending upon its occurrences. The second method was based on the use of

the Bayes theorem. AA is considered to be a special case of text-classification problem, where each author is analogous to class and each document is an instance [25].

2.1.2 Machine learning techniques

Recently various machine learning techniques have been used for the AA task. Phani, S *et al* in their approach to AA in the Bengali language used tf, tf-idf and binary word representation as features and trained the model using classifiers such as Decision Tree, Naïve Bayes and Support Vector Machine (SVM) [30]. They had a promising accuracy of 98.93% when classifying between three authors using SVM. The model's performance improved while using Topic Models as features, especially Naïve Bayes was among the best performing classifier. Their use of flexible patterns, a state-of-art technique for AA with short sentences barely worked with larger documents [36]. Use of neural network with GloVe word embedding with a window size 5 and altering vector dimension gave them an accuracy of around 99%.

SVM is useful for processing documents of significant length as it is capable of handling features space with high dimension and requires less training time [8]. Diederich *et al* simply used all words frequencies without filtering out specific features in their approach. They suggested that the selection could lead to missing out some important details about an author. In its simplest form, linear SVM is a hyperplane that separates the samples of classes with the maximum possible margin. The output formula for linear SVM is given by:

$$u = w \cdot x + b \tag{2.1}$$

where w is the normal vector to the hyperplane, x is the input vector and b is the *bias* or the displacement of the hyperplane from the origin. Maximizing the margin is an optimization problem.

Another computationally efficient classifier to handle the high dimensional feature space is Naïve Bayes [16]. Naïve Bayes uses the independence assumption and prior probability in order to calculate the posterior probability (the probability of a class given a document) [28]. The independence assumption greatly simplifies the computation. Using the independence assumption the probability of a document d belonging to a class c (or a document d written by an author c) can be denoted using

the following formula:

$$P(c|d) = P(c) \times \frac{\prod_{j=1}^{K} P(v_j|c)}{P(d)} \tag{2.2}$$

where d is a document represented by a vector of K features $d = (v_1, v_2, \ldots v_k)$. Among the two commonly used event models viz. Multivariate Bernoulli and Multinomial event model, Multinomial event model has been known to outperform multivariate and it is also more favorable compared to more specialized event models [16].

Previous research trying to address the problem of limited data focused either on augmenting the training data or feature engineering; i.e., finding or creating the optimal features that would help in differentiating authors. Luyckx and Daelemans analyzed the influence of increasing the number of authors and training data for up to 145 authors using Support Vector Machine (SVM) as the classifier. The best accuracy score for their experiment with 145 authors was around 56% while using lexical features [18].

Qian *et al.* suggested a tri-training algorithm that used character, syntactic and lexical features [31]. Unanimously they trained labeled data using each feature type and made prediction over the unlabelled data of the same feature type. The unlabelled documents are labeled based on majority agreement and the process is repeated until all the unlabelled documents are labeled. This technique of data augmentation can prove to be very useful in case of small datasets. Their approach performed significantly better than other self-training and co-training methods. However, the least number of training documents in their experiments was 10 Written Documents per Author (WDpA).

Principal Component Analysis (PCA) and Linear Discriminant Analysis for dimensionality reduction (LDA-DR) are among the two ways of feature extraction used in AA. Feature extraction is transforming the existing features to new features. PCA is used for dimensionality reduction to create features with maximum variance using the existing features also known as principal components. LDA-DR on the other hand creates features to find maximum separation between the classes.

Baayen *et al.* conducted a study to prove the hypothesis that there exist a textual fingerprint in the stylometry of the authors using unsupservised machine learning techniques [3]. They created a dataset using 9 essays by each of the 8 students from similar background. In their study they found that latent discriminant analysis is

more appropriate compared to principal component analysis (PCA) for authorship verification task. Also, they confirmed that PCA fails to provide insight when the authors are from similar background. They reported an increase in accuracy score by 10% when the stylometry was analyzed after controlling the variables such as topic and genre.

One of the recent unsupervised technique was suggested by Iqbal *et al* that used cluster analysis for authorship on unlabelled emails dataset [13]. They created a dataset using character, word, syntactic and domain-specific features. This dataset created was used as the input to the clustering methods and Writeprints technique was used to estimate the number of clusters [1]. Writeprints technique is a supervised form of PCA that is used to extract features with most information. The number of clusters in their research was set to the number of authors. However, using the exact number of author is not practical for applications where the number of authors is unknown.

Anwar *et al.* recently used the Latent Dirichlet Allocation for topic modelling (LDA-TM), an unsupervised machine learning approach for AA of Urdu language text [2]. LDA-TM is an unsupervised probabilistic approach of finding topics that represents a given document along with its probability. They created their own dataset using Urdu news domain with 15 authors and 400 documents per author. They used improved sqrt-cosine similarity classifier, a distance based classifier to train their model. They conducted their experiments on instance-based and profile-based datasets with and without n-grams. The instance-based dataset treated each document individually and profile-based dataset joined all the documents to create a profile for the author. LDA-TM performed best on instance-based dataset with n-gram features with an accuracy of 92.89%.

There has also been attempts to use the deep learning methods for AA. Rhodes used Convolution Neural Network (CNN) for authorship identification on long text documents on two datasets viz. Canada Dataset, consisting of books written by 6 authors and PAN 2012 author identification challenge dataset [33]. Although CNN was in practice since 1990's, it became widespread after a breakthrough in image classification using CNN. Among other challenges the variable-length of text documents makes it difficult to use CNN in Natural Language Processing (NLP) applications.

However, researchers have found means to resize the text documents to a constant size as a solution towards the variable-length length problem. The suggested model performed better on the PAN 2012 dataset using full backpropagation and tuning of word vectors.

Shrestha *et al.* suggested the use of character n-gram feature with CNN for AA on short text. They experimented with both character sequence (character uni-grams) and character n-grams with varying number of tweets and authors. They found that character n-grams perform better than character sequences. They got an accuracy of 76.1% with character n-grams using 1000 tweets per author by 50 authors over the micro-messages dataset by Schwartz et al [36]. They also tried Long Short-Term Memory (LSTM) along with word 2-grams, 3-grams, and 4-grams, but the performance dropped compared to CNN with character n-grams. The accuracy score dropped to the best of 56.2% and 61.7% when using 50 and 100 tweets per author for training respectively. One key problem observed with both the deep learning methods is the sheer quantity of data required to train the model for their optimal performance. Less data is prone to overfitting and therefore, deep learning approaches would not be very helpful in our research considering our focus on relatively smaller datasets.

Schwartz *et al.* suggested the *k-signature* and *flexible patterns* approach for AA of micro-messages such as tweets that was mentioned in previous review [36]. The *k-signature* of an author a is a feature that appears in at least $k\%$ tweets of author a and not in any other author. These signatures assign a unique characteristics to each individual author that can help differentiate between authors. They proved in their research that *k-signature* forms a significant portion in every author's writing. They suggested a novel feature creation technique called *flexible patterns*. The *flexible patterns* consists of High Frequency Words or HFW (appears $> 10^{-4} \times s$ in a corpus of size s) and Content Words or CW (appears $< 10^{-3} \times s$). Flexible patterns consists of at the most 6 HFWs and they start and end with a HFWs. Also, consecutive HFWs could be separated by zero or more CWs. They used SVM to train their model and received the best accuracy of 69.7% when classifying between 50 authors and 1000 tweets per author. They coupled the flexible pattern features with word and character n-gram features. Their approach is a generalization of using word n-grams, but also looking into the fine grained details of author's stylometry.

2.1.3 Character N-Gram approach (CNG)

Kešelj *et al.* suggested a language-independent approach to AA [15]. The approach used character n-grams on original documents without any pre-processing. This approach creates a profile for each author by selecting a set of common n-grams (L most frequent n-grams) for the given n-gram size (n). For an unseen document, they calculated a dissimilarity score (d_0) by comparing the training n-gram profile for each author with the n-gram profile of the unseen document. The author with the least dissimilarity score was assigned as the author of the unseen document. However, Stamatatos found that this approach performed poorly in case of imbalanced and limited training data [47]. Therefore, he suggested three variations of calculating the dissimilarity scores (d_1, d_2, and SPI) to tackle this problem.

Common N-Grams (CNG)(d_1) uses the same formula for calculation of dissimilarity as CNG(d_0) (Eq. 2.3, 2.4). However, it only takes into consideration the n-grams from the test profile ($g \in P(x)$), whereas d_0 considers both test and train profile ($g \in P(x) \cup P(T_a)$) while calculating the dissimilarity score. $f_x(g)$ and $f_{T_a}(g)$ represents the frequencies of n-gram g in the test document and author a's training documents respectively.

$$d_0(P(x), P(T_a)) = \sum_{g \in P(x) \cup P(T_a)} \left(\frac{2(f_x(g) - f_{T_a}(g))}{f_x(g) + f_{T_a}(g)} \right)^2 \tag{2.3}$$

$$d_1(P(x), P(T_a)) = \sum_{g \in P(x)} \left(\frac{2(f_x(g) - f_{T_a}(g))}{f_x(g) + f_{T_a}(g)} \right)^2 \tag{2.4}$$

CNG(d_2) is useful to counteract the problem with the higher value of profile size (L). It considers the normalized training profile which helps to stabilize the overall value of the dissimilarity score in case of higher values of L (Eq. 2.5). Unlike individual train profile $P(T_a)$ which different for each author, normalized profile $P(N)$ is created using the L most common n-grams across all the authors. In Eq. 2.5 $P(N)$ is the normalized profile and $f_N(g)$ is the normalized frequency or the frequency of g in normalized profile $P(N)$ ($f_N(g) = 0$ if $g \notin P(N)$).

$$d_2(P(x), P(T_a), P(N)) = \sum_{g \in P(x)} \left(\frac{2(f_x(g) - f_{T_a}(g))}{f_x(g) + f_{T_a}(g)} \right)^2 . \left(\frac{2(f_x(g) - f_N(g))}{f_x(g) + f_N(g)} \right)^2 \tag{2.5}$$

Simplified Profile Interaction (SPI) is useful in case of imbalanced datasets, where CNG(d_0) tends to favor the authors whose profile is shorter than profile size (L). A common n-gram g adds toward the dissimilarity score of an author with profile $P(T_{a_1})$ profile shorter than L. However, it does not adds towards the score for author with profile P(Ta2) from whom g exists but not included in the L most common n-grams. This is quite common in AA where profile for some authors is shorter than other authors. Unlike other CNG methods, SPI is similarity measure which only uses the counts of common n-grams (cardinality) and not the frequency information from the test and train profiles. In other words, it only counts the number of common n-grams between the test and train profiles instead of calculating the score based on the frequency of each n-gram. In Eq. 2.6, |.| represents the cardinality. A simplified profile of the L most frequent n-grams of size (n) for the test and train documents is represented by $SP(x)$ and $SP(T_a)$ respectively.

$$SPI(SP(x), SP(T_a)) = |SP(x) \cap SP(T_a)| \tag{2.6}$$

Based on our review Support Vector Machine (SVM), Multinomial Naïve Bayes (MNB) and CNG (d_0, d_1, d_2, and SPI) methods were selected as classifiers. Although deep learning method such as Convolution Neural Network (CNN) have proved to give better results, but their requirement of large dataset has made us choose other methods over it. SVM has proved to work over both small and large dataset, such as their application in flexible patterns and in research by Luyckx and Daelemans research over 145 authors [36, 18]. For SVM experiments Stochastic Gradient Descent (SGD) was used to optimize SVM's loss function as it takes less time to converge [8]. Naïve Bayes is computationally efficient and dates back to one of the most recognized work done by Mosteller and Wallace research where they used Bayes theorem (without the independence assumption) to resolve the authorship attribution problem between the Federalist Papers. CNG methods were chosen because of their originality and well received performance in AA. Especially CNG (d_1, d_2, and SPI) methods were of great interest as they are said to work better with smaller and imbalanced dataset. Therefore, CNG methods would prove to be of great interest for our research considering the higher number of authors and finding the best performing model for smaller datasets.

In a machine learning approach, the key components for developing a sound model are dataset, features, and classifier. The classifiers for our research has been discussed previously. Character and word n-grams as features have proved to provide state-of-the-art performance and therefore, they were used as the features created using raw text documents [15, 2, 44, 36]. For the second part of our research (RD), our focus would be on the read documents. Therefore, there was a need of custom dataset that consists of written and read documents by the same set of authors. For collection of the data business and investing reporters and columnists were considered. The written articles by these authors would be used to created the primary written dataset. In order to create the read dataset, the Twitter account of the selected authors was used. For collection of the read documents, it was assumed that any tweet shared on the author's account with an external URL is read by the author. The process of collecting and preparing the data is discussed in Chapter 3.

Among the three key components of machine learning (dataset, features, and classifier), the research focuses more towards the dataset component. Although the research considered improving the features and tuning hyper-parameters of the classifiers, mostly the efforts were directed towards analyzing and improving the dataset in order to report the performance of the chosen classifiers for the selected features.

Although there has been a lot of research done in AA, there's no emphasis on answering, *"How much data is enough?"*. This problem is analogous to the Probably Approximately Correct (PAC) Learning in machine learning [12]. The research focuses on finding the minimum number of documents required to achieve highest accuracy or in other words, sample complexity. Therefore, in the first part of our research (TS), the performance of the individual classifier was analyzed when training with 7 to 64 (TSL35) and 4 to 6 (TSS35) WDpA. This is done with the focus of observing the performance of the classifiers at varying size of training data and how much does the size of of training data influence the performance of the classifier. It would be interesting to observe the performance of the classifier on less number of WDpA (TSS35).

No research done so far has suggested the use of the documents read by the authors for training. Considering the limited data problem, and the amount of data consumed, the use of the documents read by the authors to training the model is

suggested when limited amount of written documents are available. Therefore, in the second part of our research (RD), two separate sets of experiments were conducted, with and without the read documents (RD20 and TSS20) and their results were benchmarked against each other. This would assist in understanding if the read documents can help improve the performance of the classifier in AA.

2.2 Statistical Significance Test for Model Selection

The distribution of scores in multiple classifiers makes it difficult to select a model by visual inspection. Therefore, assistance from the statistical tests make it easier to analyze the difference in models and select the one that statistically outperforms others.

In his research, Demšar discussed the use of Paired t-test, Wilcoxon signed-ranks test, ANOVA and Friedman test for statistical comparisons of the classifiers [7].

The Paired t-test is useful when comparing two classifiers. However, one of the weaknesses of the Paired t-test is its requirement of normality for smaller distribution. Also, the tests to check the normality perform poorly on small sets. They also perform poorly in presence of outliers similar to the mean value.

Wilcoxon signed-ranks test is comparatively safer to use when there is uncertainty about the normality of the distribution. Although this test is considered less powerful when the assumptions of the t-test are met, the contrary is true otherwise.

Both t-test and Wilcoxon test are only useful when comparing two classifiers. When comparing multiple classifiers ANOVA and the Friedman test are useful. The null-hypothesis in these tests is that all the classifiers perform the same and there is no statistical difference between them. ANOVA test is considered less useful while comparing machine learning algorithms due to the violation of its normality assumptions. However, on the other hand, the Friedman test is more useful for comparing multiple classifiers when the distribution fails to meet ANOVA's assumptions. Based on this review, for comparing multiple classifiers the Friedman test is the most appropriate and therefore it was chosen for model selection in our research.

Chapter 3

Methodology

In this chapter the methodology used for conducting our research is discussed. The process of collecting and preparing the written and the read dataset is discussed in greater detail. Also, a framework to replicate the process of data collection is provided. Next, the classifiers used in our experiments are discussed along with their implementation. Pre-processing of data before feeding it to the classifiers is discussed in the pre-processing section. In the final section, the experimental setup required for our research is described. In the experimental setup outlines the process of splitting the data into train, validation, and test set, the hyper-parameter optimization and selection, and lastly the evaluation metrics used to measure the performance of the models.

3.1 Classification Problem Definition

In a document classification problem, the goal is to classify the author of an unseen document among the set of of previously seen authors in the training set. The vector representation of the document is used to train the machine learning algorithm. Any given document d either seen or unseen is represented as a feature vector $(w_1, w_2, ..., w_j)$. If i is the total number of documents in the training set and j is the total number of features across all the documents, a training set T with k authors $\{y_1, ..., y_k\}$ can be represented as,

$$T = \{(D_1, y_1), \ldots, (D_i, y_k)\} \tag{3.1}$$

The goal of a closed-class classification task is, given an unseen document d, identify the author of this document among the authors $\{y_1, ..., y_k\}$ based on the knowledge gained through the training set.

The training set in our research is made up of n documents by each of the k authors; i.e., $i = kn$.

$$T = \{(D_1, y_1), \ldots, (D_n, y_1), \ldots, (D_{(k-1)n+1}, y_k), \ldots, (D_{kn}, y_k)\} \tag{3.2}$$

The performance of our classification task is measured by calculating the accuracy with which a given classifier C finds the author of l unseen documents by each of the k authors from a set of known authors $\{y_1, \ldots, y_k\}$. The value of l increased proportionally with the value of n and it was dependent on the split ratio of training and test set. The test set t can be represented as,

$$t = \{(d_1, y_1), \ldots, (d_l, y_1), \ldots, (d_{(k-1)l+1}, y_k), \ldots, (d_{kl}, y_k)\} \tag{3.3}$$

In the first part of our research where the focus was on the size of the training data (TS), the training set T consisted of documents by 35 authors ($k = 35$). The first part was subdivided into two sections TSL35 and TSS35 referring to large and small training sets respectively.

In TSL35, the experiments were performed while increasing the number training documents per author from 7 up to 64 ($n = \{7, \ldots, 64\}$). In TSS35, the experiments performed, consisted of 4, 5 and 6 documents by each of the author ($n = \{4, 5, 6\}$).

In the second part of the research where the focus was on the influence of the read documents in AA (RD), the training set T consisted of documents by 20 authors ($k = 20$). Similar to the first part of the research, this part was also divided into two sections TSS20 and RD20, referring to smaller training sets and training sets consisting of the read documents respectively.

In TSS20, the experiments performed, consisted of 4, 5, 6, and 7 documents by each of the 20 author ($n = \{4, 5, 6, 7\}$). All the experiments up to this point consisted only of written documents in their training set. However, the training sets in RD consisted of both the written and the read documents by the authors. The training sets in RD can be represented as,

$$\begin{aligned} \bar{T} = \{&(D_1, y_1), \ldots, (D_n, y_1), \ldots, (D_{(k-1)n+1}, y_k), \ldots, (D_{kn}, y_k), \ldots, \\ &(\bar{D}_1, y_1), \ldots, (\bar{D}_m, y_1), \ldots, (\bar{D}_{(k-1)m+1}, y_k), \ldots, (\bar{D}_{km}, y_k)\} \end{aligned} \tag{3.4}$$

where each pair $(\bar{D}_p, y_q)$ represents a document $\bar{D}_p$ which is read by the author y_q. We set $m = n \times 10$; i.e., for each author we have 10 times more read documents

that written documents by the same author.

Regardless of the change in the structure of the training sets across all the experiments, the structure of the test set (t) remains same across all the experiments i.e. it consists of l written documents by each of the k authors (3.3).

3.2 Written Dataset

Unlike most of the previous research that used existing dataset, a new dataset was created for the research. The purpose of creating our own dataset was the unavailability of the read documents required for our research by the same set of authors in the written dataset. Therefore, the written and read documents were collected by the same set of authors. The process of collection and preparation of the written and the read dataset would be discussed in this and the following section 3.3.

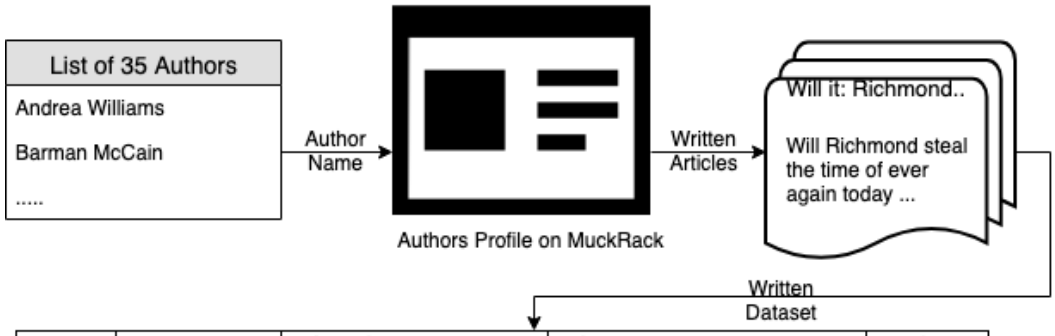
For the first part of our research, news articles written by 35 business and investing reporters and columnists were analyzed.[1] Our dataset consisted of a varying number of articles from each individual author. However, to limit the influence of external variable the following parameters were fixed while creating the written dataset:

- No articles older than two years were included in the written dataset
- All the articles included in the experiments were written by a single author
- A maximum of 100 articles was taken per author
- The sampling was stratified throughout the whole process, so as to maintain a balanced dataset

Also, the authors in consideration are business and investing reporters and columnists. It was assumed that the similarity in their area of expertise should also bring similarity in stylometry of the authors due to the use of overlapping words, phrases and ideas from their common domain. It is also expected that these overlaps and repetitions should make it difficult for the classifiers to differentiate between the authors.

[1] https://www.theglobeandmail.com/report-on-business/follow-your-favourite-business-and-investing-reporters-and-columnists/article4498088/

Figure 3.1 provides the overview of data collection of articles by the business and investing reporters and columnists. Initially, a list of 35 authors were selected based on their order of appearance at the source. The profiles of these 35 authors was searched on MuckRack.[2] MuckRack has a list of all the articles written by the author. The articles were considered written only for The Globe And Mail within the same window of two years. Their Twitter Id were also collected for these authors that would be useful to prepare the read dataset. Due to anonymization, the authors were renamed in the sample data. Also, an identification (Author ID) was given to each author, starting from A1 up to A35 in random order, which will be used in this research to address the respective author.



Author ID	Author Name	Article Title	Article Text	Datetime
A1	Andrea Williams	Will it: Richmond the steal ...	Will Richmond steal the time...	2017-04-16
A1	Andrea Williams	Cenovus shares tumble...	Cenovus Energy Inc. shares...	2017-03-30
A2	Barman McCain	Trump can't put one more...	There is much about more...	2017-01-20
A3	Kiara Ohio	Franklin Bruce, Excel...	The quick limit of growth across ...	2017-05-02
	.....	.....	.....	.....

Figure 3.1: **Process of collection and creation of written dataset.** The figure above shows the process through which the data was collected and processed for the creation of written dataset.

The article title was too short for our analysis and it was missing for some of the articles. Therefore, although the article title was collected while creating the written dataset, only the main body of the article (Article Text) was used for our experiments.

Figure 3.2 shows the word count per document for each of the individual author. It is calculated by taking the average of word count over at the most 100 written

[2]https://muckrack.com/

documents by each author (WDpA). Although most of the authors had at least 100 documents in the written dataset, authors A3, A31 and A32 had only 32, 19 and 54 documents respectively. This could well justify the lower word count for author A32, along with the author's tendency to write shorter articles. The average of word count for all the authors was 756 words, with a maximum of 1127 for author A22 and a minimum of 170 for author A32. These word count per author are considered to verify its influence on the performance for any individual author in Chapter 4.

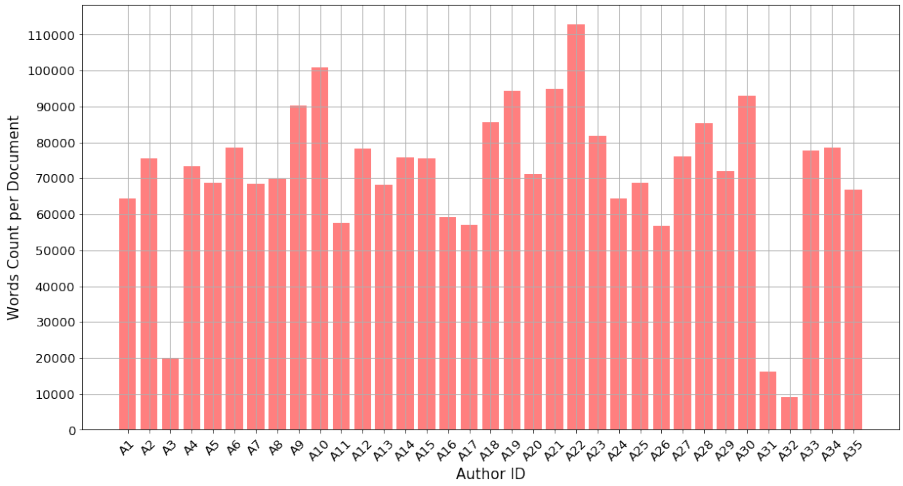


Figure 3.2: **Average Word Count per Document for each of the 35 authors calculated over 100 WDpA from the written dataset** The x-axis represents the Author IDs and y-axis represents the Average Word Count per Document.

3.3 Read Dataset

It was very crucial to figure out a way to know what the authors were reading. Our initial attempt was using Quora as the means to prepare our dataset and assume that every post upvoted by the user was read. Also, Quora is an *ideal* place to collect both the written and the read documents. However, considering the restrictions due to privacy and lack of developmental support, alternative options were considered. Eventually, it was decided to choose *Twitter*, which is both publicly available and provides developmental support.

For collection of the read data, Twitter was used as the medium. Instinctively, it was assumed that anything with an external URL shared by the authors on their Twitter account is read by them. The written dataset supported the preparation of

the read dataset. Same set of 35 business and investing reporters and columnists were used from the written dataset along with their Twitter Id as the input to start the process of collecting the read documents.

Figure 3.3 summarizes the process of collecting and creating the read dataset. As mentioned earlier, the process started with the list of 35 authors and their Twitter Id as the input. The tweets with external URLs were collected for each of the authors. However, before actually downloading the articles, the tweets were filtered out based on an *exclusion list* and the *threshold metrics*.

Table 3.1 shows the list of recurring domains that were excluded from our list while collecting the read documents. 45 domains based on the *threshold metrics* were selected. The *threshold metrics* is a metric for selecting a domain that appeared at least twice for at least two authors. To elaborate, the domain *(ipolitics.ca)* was selected that appeared more than two times across three authors. It appeared in the tweets of 5 (at least two) authors A2, A6, A7, A8, and A16. Out of these 5 authors, it appeared 14, 6 and 3 (more than twice) for author A6, A8, and A16 respectively. Remaining 44 domains were selected on the same threshold metric. The purpose of the *threshold metrics* was to select the recurring domains across multiple authors to reduce the number of parser required to obtain data from these domains, while also having documents for multiple authors from the same source. To summarize, a list of 45 domains was selected that passed the *threshold metrics* and were not in the list of *excluded domains* (Table 3.1).

The 45 domains were used to download the read documents for all the 35 authors. However, after doing basic analysis of the data collected, the authors that had considerable read documents from these 45 domains were filtered out. The output was 20 authors with at least 60 Read Documents per Author (RDpA). This same set of 20 authors was selected from the written dataset for our experiments. Similar to the written dataset, only the main body (Document Text) was selected and neither the title of the document nor the tweet text.

Similar to the written dataset, the word count per document was calculated for each author by taking 10 Written Documents per Author (WDpA) and 70 Read Documents per Author (RDpA). 10 WDpA and 70 RDpA are the maximum number of documents used for the experiments from the written and read datasets respectively

Table 3.1: **List of excluded external domains** The table shows the list of top appearing domains in authors Twitter feed and the reason to exclude them from data collection process.

Reason for Exclusion	**Domain**
No text content available	instagram.com youtube.com web.tmxmoney.com
Unable to download pages due to restrictions	bloomberg.com ft.com wsj.com blogs.wsj.com rbc.com huffingtonpost.com
Same domain as the written articles and most of the authors shared their own articles in their Twitter feeds	beta.theglobeandmail.com

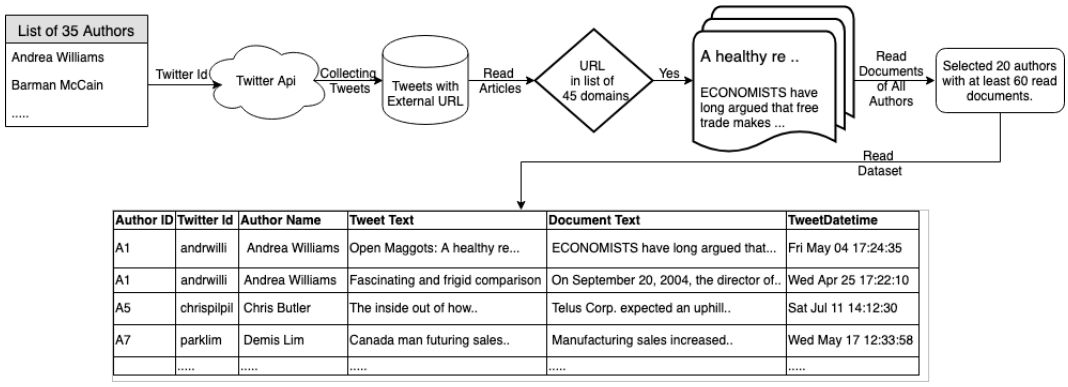


Figure 3.3: **Process of collection and creation of read dataset.** The figure above shows the process through which the data was collected and processed for the creation of read dataset.

for the second part of the research (RD). The average of word count per document all the authors was 779 and 1084 for written and read documents respectively (Figure 3.4). The spike in the word count for read documents can be attributed to the word count for author A20 (≈3500 words). When looking into the read documents for author A20, it was found that 8 out of the 70 documents had a word count greater than 10,000 words. Of these 8 documents, 6 came from the same domain. It was generally observed that the size of the read documents was more than the written documents for almost all the authors.

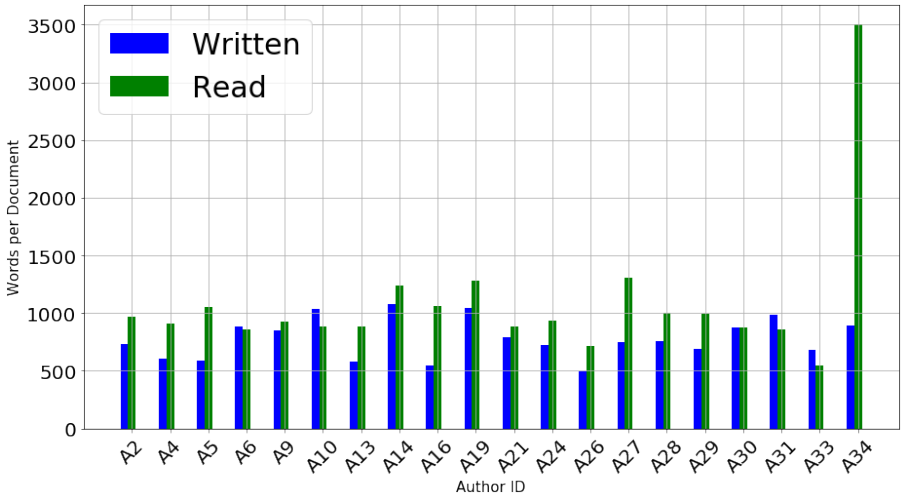


Figure 3.4: **Word counts per read and written documents for each of the 20 authors.** Words per document (y-axis) for each author (x-axis), using 10 written (WDpA) and 70 read (RDpA) documents. The blue (first) bars correspond to written, and the green (second) bars correspond to read documents.

3.4 Datasets formation for the experiments

The dataset mentioned in previous two sections (3.2 and 3.3) are the primary datasets for written and read documents. These datasets defines the universe of written and read documents w.r.t our research. For our TSL35, TSS35, and TSS20 experiments, first T written documents were considered by each author before diving them into train, validation and test 3.5. T is the total number of documents by each author selected for that particular experiment. However, throughout our research the experiments are discussed in terms of documents in the train set. For instance, in the first experiment of TSL35, the classifiers were trained with 7 written documents by each

author. In this experiment 10 ($T = 10$) written documents were selected by each author from the primary written dataset, however, after splitting the dataset into train, validation and test set (7:2:1), only 7 written documents by each author were available for training (Table 3.2). The training validation and test set are discussed in more detail in a later section (Section 3.7.1). In Authorship Attribution, the amount of data used for training is very important and therefore, each experiment is referred in terms of the number of documents and word counts used for training the classifier.

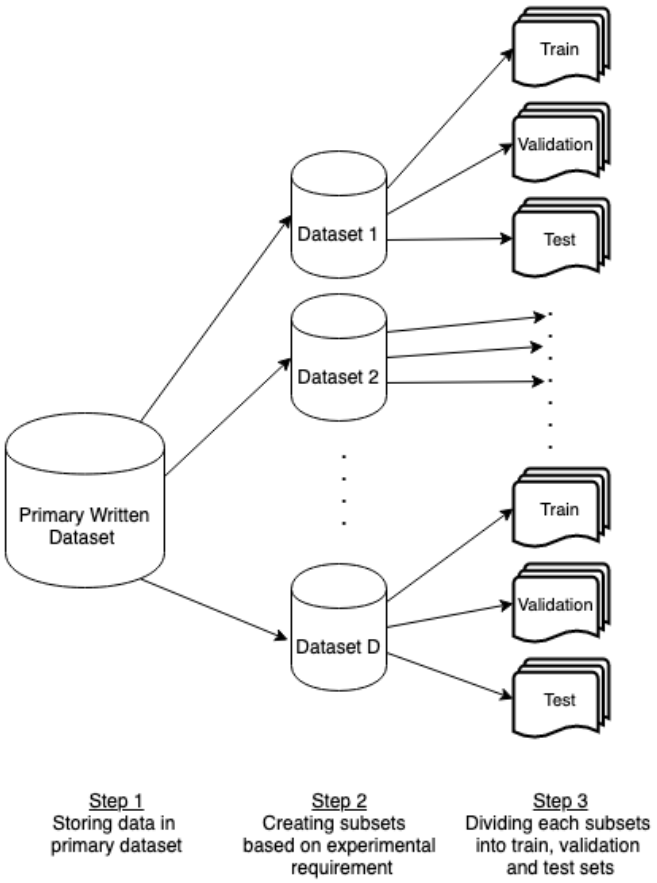


Figure 3.5: **Formation of the train, validation and test set from the primary written dataset.** The primary written dataset was split into sub datasets (subsets) by taking T number of documents by each author of the A authors. These experiment specific datasets were further split into train, validation and test set depending upon the split ratio.

Similar to TSL35, the same approach was used to create the required dataset in TSS35 and TSS20 experiments (Table 3.3) but with a different split ratio for train, validation and test sets (6:2:2). In TSS35 35 authors were considered. However, in

Table 3.2: **Formation and splitting of datasets for TSL35 experiments.** Each column represents separate experiments when the training data is gradually increased from 7 up to 64 written documents per author. The Total Documents refer to the number of written documents by each of the author selected from the primary written dataset. This is split into the ratio of 7:2:1 for train, validation and test set.

Total Documents	10	20	30	40	50	60	70	80	90	100
Train Documents	7	13	20	26	33	39	46	52	58	64
Validation Documents	2	4	6	8	10	12	14	16	18	20
Test Documents	1	2	3	4	5	6	7	8	9	10

TSS20 only 20 authors were considered due to data constraint as discussed in previous section (Section 3.3) with an additional experiment with 7 training documents per author.

Table 3.3: **Formation and splitting of datasets for TSS35 and TSS20 experiments.** Each column represents separate experiments. The experiment with 7 training documents is only part of TSS20 experiments and not TSS35 experiments. The Total Documents refer to the number of written documents by each of the author selected from the primary written dataset. This is split into the ratio of 6:2:2 for train, validation and test set.

Total Documents	7	8	9	10
Train Documents	4	5	6	7
Validation Documents	1-2	1-2	2	2
Test Documents	1-2	1-2	2	2

In the last set of experiments that included both the written and the read documents for training, exactly the same approach was used to create the datasets for experiments as TSS20 3.6. However, the key difference is the addition of the read documents to the training set (Table 3.4). For every 1 written document, 10 read documents were added in training set. For instance, in the experiments with 4 written documents for training, an additional 40 read documents were added for training. However, the same written documents were used for validation and test from TSS20 experiments.

To summarize the process of dataset formation for each experiment, it started by taking T documents from the primary written dataset. These T documents are split into train, validation and test set. While conducting the experiments with the read

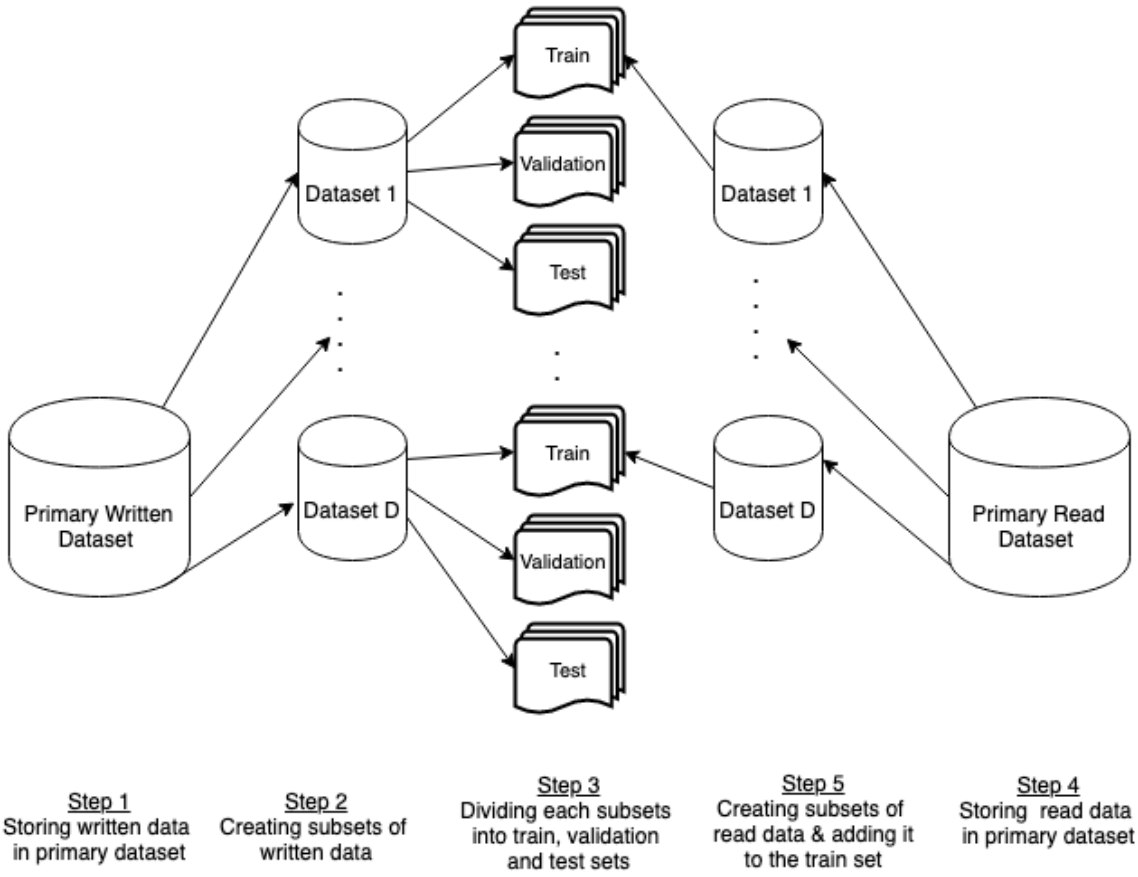


Figure 3.6: **Formation of the train, validation and test set from the primary written dataset and adding read documents in train set.** Similar to the primary written dataset, the primary read dataset was split into subsets. Each subset of read documents consisted of 10 times the written documents in train set.

Table 3.4: **Formation and splitting of datasets for RD20 experiments.** Each column represents the number of documents by each author in separate experiments. For every 1 written document by an author, 10 times the read documents were added by the same author . The Total Written Documents refer to the number of written documents by each of the author selected from the primary written dataset. This is split into the ratio of 6:2:2 for train, validation and test set.

Total Written Documents	7	8	9	10
Train Written Documents	4	5	6	7
Train Read Documents	40	50	60	70
Total Train Documents	44	55	66	77
Validation Documents	1-2	1-2	2	2
Test Documents	1-2	1-2	2	2

dataset (RD20), the read documents were only added to the training set.

3.5 Selected Classifiers

As discussed in the Chapter 2, Support Vector Machine (SVM), Multinomial Naïve Bayes (MNB) and Common N-Grams (CNG)(d_0, d_1, d_2, and SPI) methods were chosen for classification. The CNG methods have proved to be consistent regardless of the language on raw documents, which could be useful for our research.

Stochastic Gradient Descent with loss hinge (SGDClassifier) was used to implement SVM [39]. Similarly, MultinomialNB for MNB was used, a Python sklearn module [43]. However, for CNG(d_0, d_1, d_2, and SPI) Python implementation by Potthast was used *et al* [21, 20]. Changes were made to CNG to suit our input and flow.

3.6 Pre-processing

Inspired from the language independent approach suggested by Kešelj *et al*, it was decided to use the documents in their original form for the experiments without preprocessing for the experiments [15]. The necessary cleaning required such as replacing encoded or special characters and removing markup tags were fixed during the process of collecting and preparing the datasets. Also, on visual inspection of both the written and the read dataset, it was found that all the available text documents were of written professionally, without any short-hand words or slang language usage. This can to due to the fact that the authors selected for the experiments are journalists who prefers to use proper sentences and read documents that are well structured. For feature creation, a vector representation of the documents was used with word and character n-grams. To weigh the features based on their relevancy, term frequency - inverse document frequency (tf-idf) was used for vectorization purpose (TfidfVectorizer — a python module) [38].

3.7 Experimental Setup

The experiments were carried out in three steps. First, preparing a dataset by increasing the number of documents by each author for training. Second, intuitively

try a combination of parameters for each algorithm to find the optimal parameters. Third, evaluate the results of each classifier. The next few sections discusses the process of experimentation in greater detail.

3.7.1 Train/validation/test sampling for TS

For the first part of the research (TS), the experiments were subdivided into two sections. First, to analyze the influence of larger training datasets consisting 7 to 64 WDpA (TSL35). Second, to analyze the performance of classifiers with smaller datasets consisting 4 to 6 WDpA (TSS35).

Based on our review of previous Authorship Attribution (AA) research, it was initially decided to split the data into the ratio of 8:2 for training (80%) and testing (20%) respectively [18, 35, 10, 45]. However, for TSL35, the augmentation of larger datase problem was done by allocating more data to the training set. Therefore, for TSL35 the dataset was split into a ratio of 9:1 for training (90%) an test (10%) set respectively. $\approx 22.3\%$ of training samples were used for validation set, this makes up for 20% of the entire dataset. The split ratio finally becomes 7:2:1 for train, validation and test set respectively. The validation set was used for parametric optimization i.e. to choose the optimal hyper-parameters that gave the best validation accuracy.

For TSS35, a split ratio of 8:2 was used for training (80%) and testing (20%). In TSS35 the influence of training data with smaller dataset was measured and therefore less portion was used for training and more for testing compared to TSL35.

The samples or documents were selected for the experiment based on stratified random sampling technique [4]. Stratified random sampling means that equal number of samples ($randomstate = 42$) were randomly selected for each of the author while preparing the training, validation and test split. This provides a balanced dataset while also reducing the selection bias by giving every document of any given author an equal chance to be selected.

Table 3.5 summarizes the word and document count in the train, validation and test set for TSL35 experiments. The total documents (*Total Docs*) in validation set is almost double the test count. The total words (*Total Words*) in validation is higher than the total words in test, but the average words per document (*Words per Doc*) is nearly the same for both validation and test set.

Table 3.5: **TS — Train, Validation and Test set summary for TSL35.** Dataset to analyze the performance of classifiers over documents sizes ranging from 7 up to 64 written documents per author for training by 35 authors.
Words per Doc is the average number of words in each document (*Total Words/Total Docs*) and *Words per Author* is the average number of words for each author (*Total Words/35 authors*).

Train					Validation				Test			
Docs per Author	Total Docs	Total Words	Words per Doc	Words per Author	Total Docs	Total Words	Words per Doc	Words per Author	Total Docs	Total Words	Words per Doc	Words per Author
7	242	186,433	770	5,327	73	49,133	673	1,404	35	31,810	908	909
13	483	369,091	764	10,545	145	109,547	755	3,130	70	55,342	790	1,581
20	716	545,074	761	15,574	214	158,362	740	4,525	104	75,720	728	2,163
26	944	711,098	753	20,317	283	205,507	726	5,872	137	101,406	740	2,897
33	1,173	870,612	742	24,875	351	269,384	767	7,697	170	120,266	707	3,436
39	1,398	1,044,295	746	29,837	418	309,763	741	8,850	202	149,683	741	4,277
46	1,620	1,205,093	743	34,431	484	362,893	749	10,368	234	177,767	759	5,079
52	1,841	1,378,869	748	39,396	550	420,471	764	12,013	266	201,546	757	5,758
58	2,055	1,551,310	754	44,323	615	468,676	762	13,391	297	230,512	776	6,586
64	2,264	1,717,729	758	49,078	677	516,271	762	14,751	327	253,884	776	7,254

The split ratio for train and test for TSS35 experiments was set to 8:2 to have less instances for training. This was done to augment the situation of smaller dataset. From the training some part was shared for validation which was equal to the ratio of test documents in the TSS35 experiments. This can be verified in Table 3.6, where the total number documents (*Total Docs*) are almost identical for both validation and test set. The words per document for validation and test set are not as close as they were in TSL35 experiments, this is due to averaging over less number of documents in TSS35 experiments.

Table 3.6: **TS — Train, Validation and Test set summary for TSS35.** Dataset to analyze the performance of classifier on smaller dataset with only written documents by 35 authors.
Words per Doc is the average number of words in each document (*Total Words/Total Docs*) and *Words per Author* is the average number of words for each author (*Total Words/35 authors*).

Train					Validation				Test			
Docs per Author	Total Docs	Total Words	Words per Doc	Words per Author	Total Docs	Total Words	Words per Doc	Words per Author	Total Docs	Total Words	Words per Doc	Words per Author
4	164	124,209	757	3,549	41	33,336	813	952	40	29,480	737	842
5	188	144,212	767	4,120	47	30,651	652	876	45	37,787	839	1,080
6	211	157,146	744	4,490	53	37,657	710	1,076	51	41,510	813	1,186

3.7.2 Train/validation/test sampling for RD

The second part of the research (RD), was an extension of the first part (TS) and therefore, the same split ratio was same as that of TSS35. Since this part of the research (RD) dealt with less authors (due to the data constraint), datasets required for the experiments were recreated.

The goal of RD was to analyze the influence of the read documents in AA. Therefore, in order to benchmark the results of the TSS20 against the RD20, two separate sets of experiments were conducted. For the first set of experiments (TSS20), only the written documents were considered for training starting with 4 to 7 WDpA. For the second set of experiments (RD20), both written and read documents were considered for training. In RD20, the number of written documents were the same as in the TSS20 experiments. However, for every count of written documents, ten times its RDpA were considered. For instance, for 4 WDpA, 40 RDpA were considered and so on. The ratio of written to read (1:10) was chosen intuitively based on some preliminary experiments. Eventually, in both sets of the experiment, the classifiers were validated and tested on the same set of written documents, which roughly consisted of 1–2 WDpA.

Unlike TS, RD only deals with 20 authors instead of 35 authors. This also changes the document (*Total Docs*) and word counts (*Total Words*) in our dataset. Although the count of total validation and test documents have reduced because of less authors, their ratio remains the same (Table 3.7). It can be observed that the average words per document fluctuates inversely with the size of the data; i.e., smaller datasets have higher fluctuation. This can be attributed to the higher number of words for a single document in a small pool of documents with fewer words. From the total word count given for the training data (*Total Words*) the minimum and maximum number of words per author (*Words per Author*) for training are ≈ 3750 and ≈ 5195. Luyckx and Daelemans mentioned in their research that for an AA task, the reliable minimum number of words required per author for training is around $5,000$ words [9]. The word counts for each of the author is less or around the $5,000$ mark and therefore, our datasets can be truly considered small dataset.

Also, when comparing the TSS20 datasets with 20 authors (Table 3.6) to the TSS35 datasets with 35 authors (Table 3.7). The average words per document (*Words*

per Doc) and words per author (*Words per Author*) remains in close range to each other.

Table 3.7: **RD — Train, Validation and Test set summary for TSS20.** Dataset consisting of only written documents by 20 authors.
Words per Doc is the average number of words in each document (*Total Words/Total Docs*) and *Words per Author* is the average number of words for each author (*Total Words/20 authors*).

Train					Validation				Test			
Docs per Author	Total Docs	Total Words	Words per Doc	Words per Author	Total Docs	Total Words	Words per Doc	Words per Author	Total Docs	Total Words	Words per Doc	Words per Author
4	93	74,963	806	3,748	24	15,903	662	795	23	17,974	781	899
5	107	84,855	793	4,243	27	19,107	707	955	26	20,256	779	1,013
6	120	91,083	759	4,554	31	28,119	907	1,406	29	18,822	649	941
7	134	103,883	775	5,194	34	24,931	733	1,247	32	27,010	844	1,351

For summarizing the train, validation and test for RD20 in RD, two more columns were added (Table 3.8). These columns describe the number of written (WDpA) and read (RDpA) documents considered per author for each experiment. Comparing the number of documents and words in validation (*Total Docs* and *Words per Doc*) and test from TSS20 experiments (Table 3.7) with RD20 experiments (Table 3.8), it can be confirmed that the same written documents are used for validation and test respectively. In other words, the classifiers were trained, validated and tested only using the written documents in TSS20 experiments. However, in RD20 they were trained using both the written and the read documents, while validated and tested using only written documents.

Table 3.8: **RD — Train, Validation and Test set summary for RD20.** Dataset consisting of both written and read documents by 20 authors.
Words per Doc is the average number of words in each document (*Total Words/Total Docs*) and *Words per Author* is the average number of words for each author (*Total Words/20 authors*). *Doc per Author* is the sum of written (*Written Docs per Author*) and read (*Read Docs per Author*) documents by each author.

Train							Validation				Test			
Written Docs per Author	Read Docs per Author	Docs per Author	Total Docs	Total Words	Words per Doc	Words per Author	Total Docs	Total Words	Words per Doc	Words per Author	Total Docs	Total Words	Words per Doc	Words per Author
4	40	50	1,006	1,054,456	1,048	52,723	24	15,903	662	795	23	17,974	781	899
5	50	56	1,113	1,179,248	1,059	58,962	27	19,107	707	955	26	20,256	779	1,013
6	60	60	1,205	1,293,709	1,073	64,685	31	28,119	907	1,406	29	18,822	649	941
7	70	64	1,286	1,320,356	1,026	66,018	34	24,931	733	1,247	32	27,010	844	1,351

3.7.3 Hyperparameter optimization

For parameter optimization, a grid search approach was used for SVM and MNB [42]. Similar to the grid search approach in SVM and MNB, a list of all possible combinations of parameters was created for CNG, fed it to the algorithm and recorded the parameters for which the best accuracy was observed on the validation set. Changes were made to the parameters based on the literature review and observation of the results on the validation set. The parameter optimization was only done on the validation set and not on test set. The most optimal parameters were recorded for all the algorithms so that they can be used to record the accuracy on the test set.

Our experimentation started with the CNG methods for which the traditional train, validation, test split approach was used, without using k-fold cross-validation. Therefore, although grid search uses cross-validation to report the best training accuracy, the experiments were continued with SVM and MNB using the same approach of tuning and testing the parameters on validation set and report the final unbiased results on the test set to maintain consistency between the SVM and MNB experiments with the CNG experiments. Also, for our research, the results of the existing classifiers are benchmarked against each other and therefore, the use of train, validation and test versus cross-validation will hardly affect the results of our research as long as all the classifiers get the same treatment. The use of cross-validation is time consuming. Although smaller datasets are used in most of our experiments, the grid search approach for hyper-parameter selection increases the training time because each experiment has to go through all possible combinations of the hyper-parameters. Cross-validation on top of the grid search would increase the training time by k-fold times which without having any major impact to the overall results. Therefore, our experiments were continued with using cross-validation.

The parameters shared for SVM and MNB are based on *scikit-learn v0.19.1*. All the other parameters not mentioned were set to their default values. Parameters for SVM, MNB and CNG methods are shared in Table 3.9, 3.10, and 3.11 respectively.

For selection of the hyperparameters, intuitively two or three different values were selected initially. Based on the results using the initial values, the later hyperparameter values were selected in the close proximity of the values yielding good performance and finalized on the list of values shared in the Table 3.9, 3.10, and 3.11.

Table 3.9: **Parameters for SVM.** Description from sklearn's documentation [38, 39]

Parameter	Description	Value
`use_idf`	Enable inverse-document-frequency reweighting	True
`ngram_range`	The lower and upper boundary of the range of n-values for different n-grams to be extracted. All values of n such that $min\_n \leq n \leq max\_n$ will be used.	(2, 3), (3, 4), (3, 6), (4, 6), (1,2)
`analyzer`	Whether the feature should be made of word or character n-grams. Option char_wb creates character n-grams only from text inside word boundaries; n-grams at the edges of words are padded with space. For instance ('let go' - range(2,2)): • word:['let go'] • char:[' g', 'et', 'go', 'le', 't '] • char_wb:[' g', **' l'**, 'et', 'go', 'le', **'o '**, 't ']	word, char, char_wb
`penalty`	The penalty (aka regularization term) to be used.	*l1, l2*
`alpha`	Constant that multiplies the regularization term	$1e^{-2}$, $1e^{-3}$, $1e^{-4}$

Table 3.10: **Parameters for MNB.** Description from sklearn's documentation [38, 43]

Parameter	Description	Value
`use_idf`	Enable inverse-document-frequency reweighting	True
`ngram_range`	The lower and upper boundary of the range of n-values for different n-grams to be extracted. All values of n such that $min\_n \leq n \leq max\_n$ will be used.	(2, 3), (3, 4), (3, 6), (4, 6), (1,2)
`analyzer`	Whether the feature should be made of word or character n-grams. Option char_wb creates character n-grams only from text inside word boundaries; n-grams at the edges of words are padded with space. For instance ('let go' - range(2,2)): • word:['let go'] • char:[' g', 'et', 'go', 'le', 't '] • char_wb:[' g', **' l'**, 'et', 'go', 'le', **'o '**, 't ']	word, char, char_wb
`alpha`	Additive (Laplace/Lidstone) smoothing parameter (0 for no smoothing)	0.01, 0.25, 0.6, 0.75, 1

Table 3.11: **Parameters for CNG(d_0, d_1, d_2, and SPI) methods.**

Parameter	Description	Value
`n`	Ngram size. Unlike SVM and MNB, a constant size of ngram is used instead of the range. For instance ('let go' - `n`$= 1, 2$): • `n`$= 1$: ['l', 'e', 't', 'g', 'o'] • `n`$= 2$: ['le', 'et', 't ', ' g', 'go']	4, 5, 6, 7, 9, 10
`L`	Profile size. These are the number of top ngrams that are selected for a given author.	5000, 7000, 9000, 11000, 13000, 15000, 17000, 19000

While training the model, all possible combination of the above parameters were tried for the respective algorithm. The one with the best training accuracy was selected by the algorithm and the same parameters are used to calculate the validation accuracy. The experiments were repeated with different set of hyperparameters until the best performing set of hyperparameters was found. The hyperparameters were recorded for the trial with best validation accuracy, which was later used on the test set. One thing to note is that these parameters would differ for each experiment. For instance, the parameters to perform best for the dataset with 7 WDpA could be significantly different from the dataset with 13 WDpA and so on.

3.7.4 Evaluation metrics

It was important to use an evaluation metric that was consistent with other research so that it is easier to put our results into perspective when comparing it with other works. Therefore, the accuracy score was selected as the evaluation metric [18, 35, 10, 45]. The core requirement to avoid the misuse of accuracy score is to have an equal number of samples from each class or in other words a stratified sampling. This requirement was considered while creating the train, validation and test sets. If this requirement is not met, the accuracy score would give an unrealiable measure of classifier's performance, as it would favor the class with more instances (best-class classification).

For greater insight into the results, the confusion matrices of the results were also analyzed. Confusion matrix that compares the true authors to the predicted authors, provides a deeper understanding of the strength and weaknesses of a classifier and also an easy to understand report of each author's performance. This can later help in digging into the features of the classifier for the selected instances to learn the key features that differentiates not only between authors, but also between classifiers. This would be discussed in greater detail along with our results in Chapter 4.

The *accuracy_score* and *confusion_matrix* method provided by *sklearn metrics* were used for calculating the classifier level and author level accuracy scores in our research [40, 41].

3.7.5 Implementation and Usage

The three important component for running the experiments were programming language to write the scripts, server for running the scripts and platform for result analysis.

Python as a programming language provides an extensive support for data analysis. Therefore, the entire project was developed using Python *v3.6.5*. The list of important modules required for the experiments are mentioned in Table 3.12 along with their version. All details about the modules including their installation and documentation can be found on PyPi [6].

Table 3.12: List of required modules for the experiments along with their version.

Module	**Version**
nltk	3.3
pandas	0.23.0
scikit-learn	0.19.1
numpy	1.16.2
eli5	0.8.2
lime	0.1.1.34
tweepy	3.6.0
beautifulsoup4	4.6.0
requests	2.17.1
matplotlib	2.2.2

For the purpose of collecting and preparing the dataset, *requests*, *tweepy*, *beautifulsoup4*, and *pandas* module were used. The URLs were collected from the authors tweets using the *tweepy* modules for the read dataset. The collected URLs were requested for data using *requests* module and the downloaded HTML documents were cleaned and processed using *beautifulsoup4* and *pandas*.

In order to collect data for the read dataset from the domains selected via *threshold metrics*, 45 different HTML parser were used for each of the selected 45 domains. The selection of these 45 domains have been discussed previously in Read Dataset section 3.3. Data was collected only from the domains that through a basic get request. As discussed previously, the domains that restricted the collection of data through the get requests were added in *exclusion list* (Table 3.1).

Nltk module was used for tokenization and processing of the text documents.

Scikit-learn package provided support and necessary modules to run SVM and MNB classifier [17]. It was also used for creating features (TfidfVectorizer) and evaluation (accuracy_score and confusion_matrix). For CNG methods, an existing implementation in Python by Potthast *et al.* was used [20, 21]. ELI5 and LIME modules were used to debug the machine learning algorithms in the later part of the research while analyzing the influence of the read documents (RD) [37, 48, 34].

In our research, three different variables viz. classifier, hyper-parameters and number of training documents were modified simultaneously. For each classifier, all possible combinations of hyper-parameters were tried from the given set, for an increasing size of text documents. The experiments were repeated for every new set of hyper-parameters. Due to time and computational constraints, limited sets of hyper-parameters were tried before finalizing a particular set. For faster execution, processes were distributed on different server that shared a centralized disk space. The servers used for our research are provided under the Research Compute Environment by Faculty of Computer Science at the Dalhousie University [26]. Also, our experiments were mostly run overnight or over the weekend during the time of minimum server usage. The list of servers used to the experiments and their hardware specifications are mentioned in Table 3.13

Table 3.13: List of servers used along with their hardware specifications [22].

Server Name	**CPU**	**Memory**	**Internal Storage**
Hector	16 cores @ 2.67Ghz	48GB	120GB
CGM6	32 cores @ 2Ghz	256GB	12TB
CGM7	32 cores @ 2Ghz	256GB	12TB

The results were analyzed using Google Colab [11]. It is a free Jupyter notebook environment with added support for hardware accelerators (GPU and TPU). However, our use of Google Colab was mostly restricted to analyzing the results and creating graphs using *matplotlib* module for greater insight into the result of our experiments. Google is very convenient to use without the need of any setup and runs completely in the cloud. Also, the hardware accelerators in Google Colab environment were useful to quickly recreate and run the experiments for debugging the classifiers or analyzing the features.

Chapter 4

Results

In this chapter the performance of each experiment is examined. In the first section the results to understand the influence of the size of training data (TS) are analyzed. In the second section, the influence of the read documents in Authorship Attribution (AA) (RD) is analyzed. In each of the two sections, the performance of the individual classifier, their strengths and weakness is discussed. The performance of the authors is also analyzed by comparing and inspecting the confusion matrices from different experiments.

4.1 Results for TS — Training Size

A separate result analysis was done for the experiments with documents ranging from 7 to 64 Written Documents per Author (WDpA) (TSL35) and for documents ranging from 4 to 6 WDpA (TSS35) for training.

4.1.1 Result analysis of TSL35 experiments

The classifiers performed much better than anticipated for 35 authors. On an average Multinomial Naïve Bayes (MNB) performed best with an accuracy of 82.65% in TSL35 experiments. The probability of a document to be marked correctly on a random chance in a stratified sampling would be 2.86% (i.e. 1 in 35). However, even for a model with as many as 7 WDpA for training, the least observed accuracy among all the classifiers (except Common N-Grams (CNG)(d_0)) was 71.43% for Support Vector Machine (SVM). The only surprising result among the lot was for CNG(d_0) with an accuracy of 11.43%. However, CNG(d_0) performed equally well with increasing training data. On the contrary, CNG modifications suggested by Stamatatos (d_1, d_2, and SPI) performed surprisingly well when trained with 7 WDpA and tested on 1 WDpA. The best-observed accuracy was 85.71% for both CNG(d_1) and CNG(SPI). However, it dropped to the level of other classifiers when more training data was provided.

Table 4.1: **TSL35 — Accuracy Scores (in %) from 7 to 33 WDpA for training.** The table headers describe the number of documents and words used for training the classifiers.

Total Documents	242	483	716	944	1173
Total Words/100,000	1.86433	3.69091	5.45074	7.11098	8.70612
Documents per Author	7	13	20	26	33
Words per Author/1000	5.326	10.545	15.573	20.317	24.874
SVM	71.4286	75.7143	73.0769	**81.7518**	**86.4706**
MNB	77.1429	74.2857	**75.9615**	81.0219	83.5294
CNG(d_0)	11.4286	**77.1429**	**75.9615**	78.8321	**86.4706**
CNG(d_1)	**85.7143**	75.7143	75	78.8321	83.5294
CNG(d_2)	82.8571	72.8571	75	78.8321	81.1765
CNG(SPI)	**85.7143**	75.7143	**75.9615**	78.1022	**86.4706**

Table 4.2: **TSL35 — Accuracy Scores (in %) from 39 to 64 WDpA for training.** The table headers describe the number of documents and words used for training the classifiers.

Total Documents	1398	1620	1841	2055	2264
Total Words/100,000	10.44295	12.05093	13.78869	15.5131	17.17729
Documents per Author	39	46	52	58	64
Words per Author/1000	29.837	34.431	39.396	44.323	49.077
SVM	87.6238	**88.0342**	86.4662	**87.2054**	87.7676
MNB	88.6139	85.8974	**86.8421**	85.1852	**88.0734**
CNG(d_0)	86.1386	86.3248	86.0902	84.5118	86.8502
CNG(d_1)	85.6436	84.6154	84.5865	84.5118	86.2385
CNG(d_2)	86.6337	83.3333	84.2105	83.165	85.6269
CNG(SPI)	**89.1089**	86.3248	85.7143	83.8384	85.6269

Although these results are consistent with Statamatos's research, the sudden drop from an average 84.75% to an average 74.73% for CNG(d_1, d_2, and SPI) raises the question if it is truly consistent for smaller datasets.

It was also important to understand the reason for the low score (11.43%) for CNG(d_0) while its counterpart (d_0, d_1, and SPI) performed astoundingly well. There are two potential reasons for this performance based on the literature review. The first reason could be the calculation of dissimilarity score over the n-grams from both train and test profile. Unlike other CNG methods (d_1, d_2, and SPI), CNG(d_0) considers both the train and test profiles ($g \in P(x) \cup P(T_a)$) while calculating the

dissimilarity score. In case of smaller training set, the commonality between the train and test profile ($P(x) \cup P(T_a)$) is smaller and therefore, the list of n-gram profiles to calculate the dissimilarity score over is longer. The larger number of n-grams could potentially lead to miscalculation of the dissimilarity score and thereby predicting the wrong author. The second reason could be the higher value of profile size (L). In our experiment the minimum number of profile size (L) was 5000 and went up to 19000 3.11. Also, the best performing values for profile size and n-grams size for this CNG(d_0) experiment (7 Documents per Author) were 5000 and 4 respectively. The average words per author (Words per Author) from Table 3.5 for this same experiment was 5,327. Therefore, this is an instance of some authors having shorter profiles compared to other others and the authors with shorter profile gets the preference. Both the above arguments could well justify the improved performance of other CNG methods (d_1, d_2, and SPI).

Consistent with the Probably Approximately Correct learning, it was observed that more data resulted in better scores [12]. Therefore, it can be safely concluded that more data helps to predict the authors more precisely. On average, the standard deviation between the results from TSL35 is 5.09 (excluding CNG(d_0)), which can be considered considerable given its variation from the mean for each classifier. Apparently, the maximum standard deviation is observed for CNG(d_0) due to its initial dip. However, excluding the initial result for CNG(d_0) the standard deviation comes to 4.480 which is analogous to other CNG methods. The highest deviation in the result is observed for SVM (6.63) which has consistently and considerably performed well even with smaller training data. After the set of 39 WDpA, both SVM and MNB improved in performance, whilst the CNG method dipped a little.

Although CNG(d_0) performed poorly in the initial experiment, it out-performed others on three occasions, which is second best after SVM and CNG(SPI) and among the same line of MNB. The best result across all the experiments was observed with CNG(SPI) with an accuracy of 89.11% with only 39 WDpA, which is about half of our maximum training capacity.

4.1.2 Performance analysis of authors in TSL35 experiments

The performance of individual authors was analyzed from TSL35 experiment (Figure 4.1). Each subplot in the figure can be analyzed as a separate case study. However, considering the scope of the research, the graph is only summarized to understand the performance of classifiers in case of each author. Therefore, to put the performance of the individual authors into perspective, they were classified (Figure 4.1) into four different categories *ideal*, *consistent*, *inconsistent*, and *disordered*. Among the 35 authors, 5 authors had an *ideal* performance where they were hardly misclassified by any of the classifiers (A6, A21, A22, A32, and A33). 3 authors had a *consistent* performance (A4, A18, and A19). 13 authors had an *inconsistent* performance (A2, A5, A7, A8, A9, A11, A16, A20, A23, A26, A27, A29, and A35). Remaining 14 authors had a performance that was *disordered* and different for each classifier (A1, A3, A10, A12, A13, A14, A15, A17, A24, A25, A28, A30, A31, and A34).

In case of *ideal* category, the authors A22 and A32 were perfectly classified by all the classifiers including CNG(d_0). However, A6, A21 and A33 missed out a document when training with 7 WDpA by CNG(d_0). Moreover, they were perfectly classified in all the later experiments by all the classifiers.

The misclassification of the authors in the *consistent* category was observed across multiple classifiers. Only 3 authors were observed to have a *consistent* performance (A4, A18, and A19). In case of A4, the performance of all the classifiers dropped at 20 WDpA and to the lowest at 52 WDpA. In case of A18, only CNG(SPI)'s performance dropped at 13 and both CNG(SPI) and MNB misclassified a few documents at 46 WDpA. A19 had an analogous behaviour as A4, its accuracy score dropped at 26 and 33 WDpA. The *consistent* performance of the classifiers for these authors suggest that the use of features other than the n-grams can assist in differentiating between these authors. For the authors in this category, SVM performed the best with the highest average accuracy of 95.16% followed by MNB and CNG methods.

Many authors had an *inconsistent* performance, where the performance of the classifiers for these authors was asynchronous with each other for certain experiments. Starting with author A2, the CNG methods outperformed SVM and MNB for 26 WDpA and it was the other way around with 52 WDpA. In the category, SVM mostly performed more consistently compared to other classifiers (A2, A20, A23,

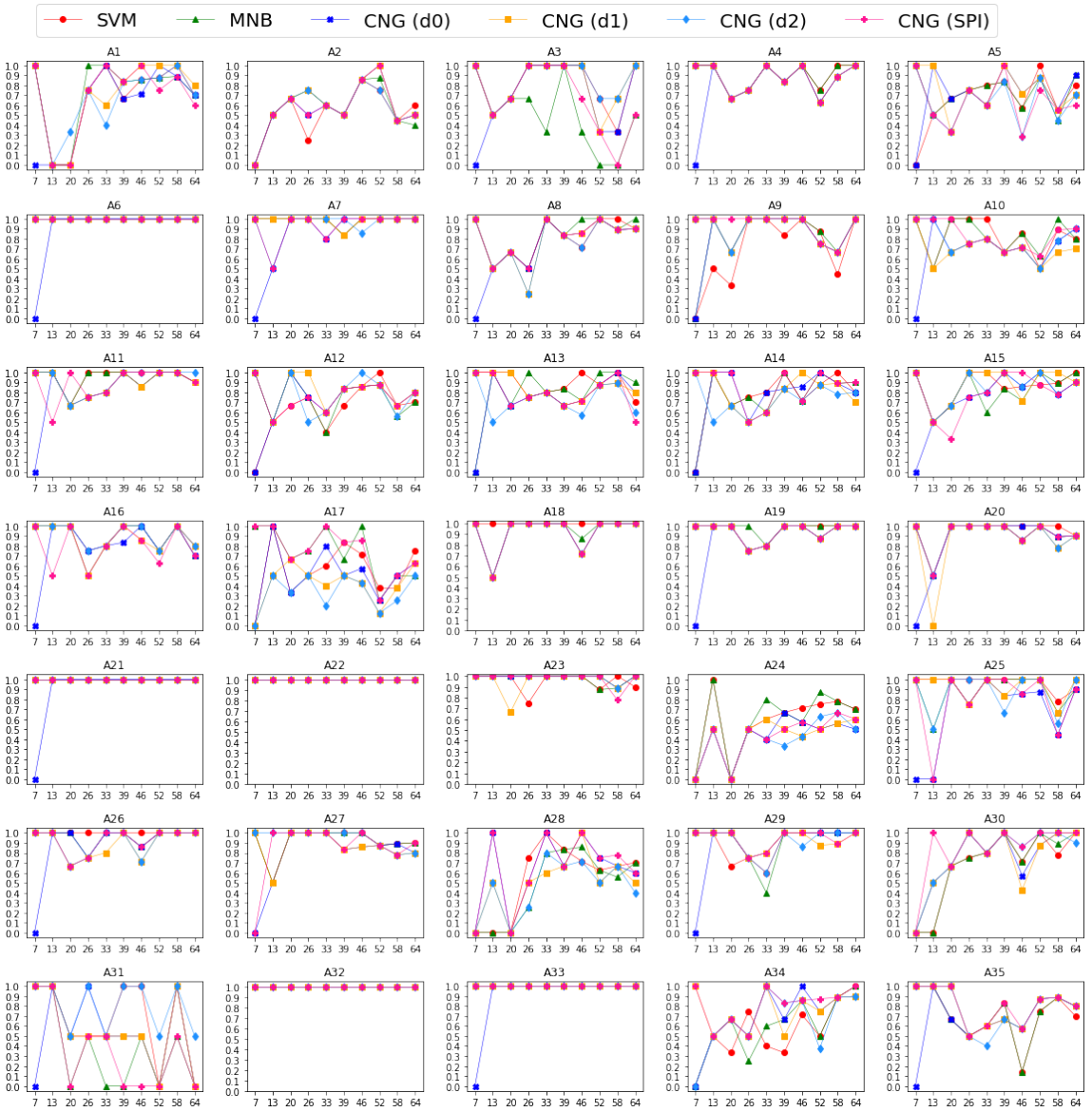


Figure 4.1: **Accuracy score plot for each of the authors across all the classifiers in TSL35 experiments** Each plot shows the accuracy score for the respective author for all the six classifiers used in TSL35 experiments. The x-axis shows the average number of written documents per author used for training and y-axis shows the accuracy score converted between the scale of 0.0 to 1.0 where 0.0 represents 0% and 1.0 represents 100%. The title at the top center of each graph represents the author id (e.g. A1, A2 and so on).

and A29). Another key observation in this category was the close relation between CNG(d_1) and CNG(d_2). Their performance was identical across all the authors, drop in performance of one was usually coupled with the drop in performance of other and vice versa. In this category MNB performed the best with an average accuracy of 85.80% followed by CNG (d_2, d_{d1}, SPI), SVM and CNG(d_0).

The majority of authors had a *disordered* performance, where the performance of majority of classifiers was incomparable for a given author. Among all, the most *disordered* cases were of A24, A28 and A31. In case of A24 all the classifiers misclassified all the documents with 7 WDpA. The performance improved after adding a few more documents, but it dropped again at 20 WDpA. The classifiers tried to catch up, but their performance was worse than initial experiments. Among all though, MNB was mostly at the top in this category. The average accuracy of A24 (46.94%) was least among all the authors, followed by A28 (53.031%) and A31 (53.33%). CNG(SPI) was the best performing classifier in this category with an average accuracy of 71.28% followed by SVM (71.00%), CNG (d_1 (70.29%) and d_2 (68.95%)), MNB (67.44%) and CNG(d_0) (67.44%).

The performance of the authors, A24, A28, and A31 with lowest accuracy from the *disordered* category was investigated. The performance of A31 can be justified by the fewer documents available by this author for training. Only 19 documents were available by this author which was further split into train, validation and test. Also, among these 19 documents, the average word count for author A31 was the second lowest, which suggest the authors tendency to write shorter articles (Figure 3.2). The conclusion that lower word count, leads to higher missclassification rate would be incorrect as author A32 had the lowest average word count (Figure 3.2), yet the author was perfectly classified by all the classifiers in all the TSL35 experiments. The total available documents by A32 were 54 before splitting them. Although this is still lower than all the other authors, it is still three times the documents available for author A31. Therefore, it can concluded that in case of author A31 the number of documents were solely responsible for its documents misclassification across all the TSL35 experiments.

Both the author A24 and A28 had sufficient amount of total documents and their average word count per document was also above par compared to other authors.

Therefore, those reasons were dismissed for their low performance. Surprisingly for both A24 and A28, the classifier misclassified all the test documents at 20 WDpA. Hence, it was decided to look into the confusion matrices of all the classifiers for that particular experiment to understand what was happening underneath (Figure 4.2).

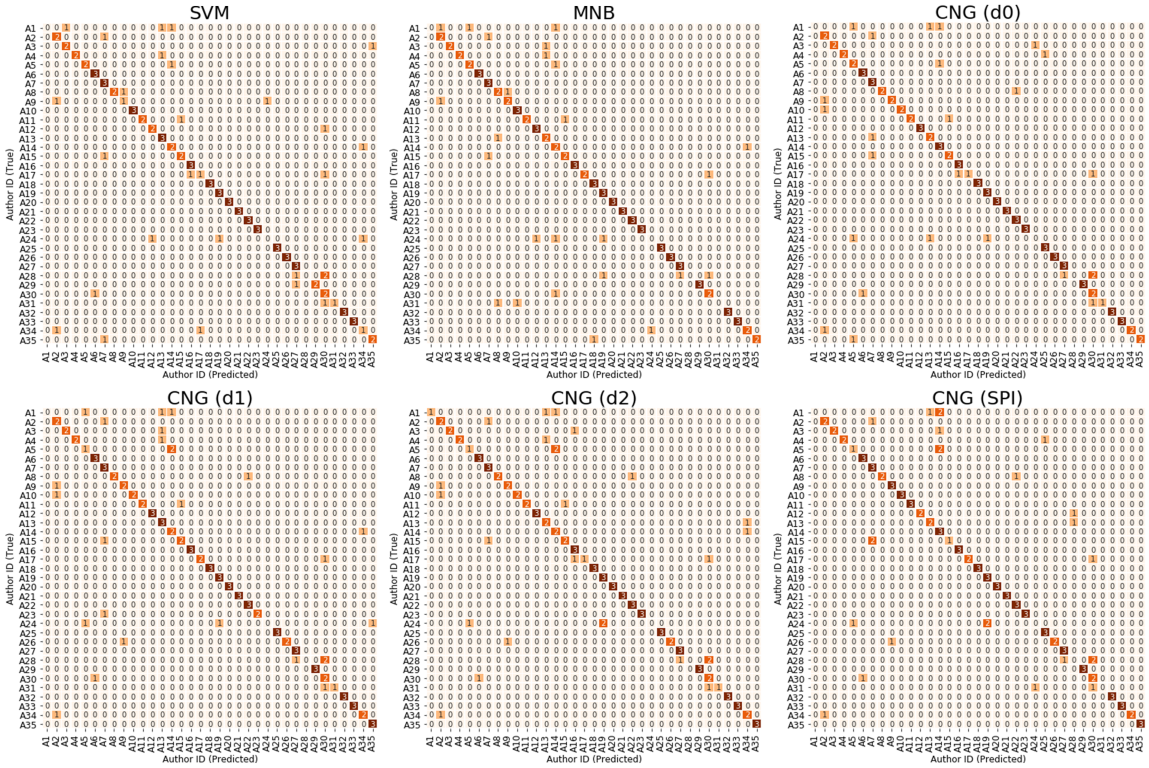


Figure 4.2: **Confusion matrices of all the classifiers from TSL35 experiments with 20 WDpA for training.** Confusion matrices showing the true authors (y-axis) versus the predicted authors (x-axis) of the test documents for each of the six classifiers when trained with 20 WDpA. The title at the top center of each confusion matrix shows the name of the respective classifier.

Interestingly for author A28, all the test documents were confused exactly for author A27 and A30 by all the classifiers (except one instance in MNB). Considering that they were all misclassified for the same author, it was assumed that they were actually written by author A27 and A30, but reported as A28. However, when looking over the internet, it was realized that author A28 had no articles written with A27 and A30. In case of A24, at least one of the three test documents was confused for author A19 by all the classifiers. When investigating the relation between author A24 and A19, it was found that they are both female authors who have previously co-authored each others articles. Therefore, in case of both A24 and A28, the only

Table 4.3: **TSS35 — Accuracy Scores (in %) for 4, 5, and 6 WDpA for training.** The table headers describe the number of documents and words used for training the classifiers.

Total Documents	164	188	211
Total Words/100,000	1.24209	1.44212	1.57146
Documents per Author	4	5	6
Words per Author/1000	3.548	4.12	4.489
SVM	65	66.6667	62.7451
MNB	72.5	68.8889	62.7451
CNG(d_0)	5	2.22222	7.84314
CNG(d_1)	**75**	73.3333	**70.5882**
CNG(d_2)	72.5	**77.7778**	64.7059
CNG(SPI)	72.5	75.5556	64.7059

option is the similarity in features of the actual author and predicted author. This can be confirmed through the repeated misclassification by all the classifiers (more so for A28 than A24).

4.1.3 Result analysis of TSS35 experiments

In this section the results of the experiments on the written documents by 35 authors (TSS35) are summarized, to verify the performance of the classifiers on smaller datasets. Performance of CNG(d_0) continued to drop even with smaller datasets. For a set with 5 training document per author, the accuracy score of CNG(d_0) was just below the random chance at around 2.23%. Contrarily, CNG(d_1, d_2, and SPI) performed consistently better, but not as good as they did in the initial set TSL35. Since the results are far above random chance, it safe to conclude that there are signals in the written documents while simply using n-grams of their written text. However, to get better results all the classifier requires more training data. The best accuracy from TSL35 was for CNG(d_2), 77.78% with 5 WDpA for training.

4.1.4 Statistical significance test of TS results

In order to verify if the results of the classifiers were statistically different from each other, the Friedman test was performed based on our literature review. The value of alpha (α) for the test was set to a standard value of 0.05. As per the Friedman

test for statistical significance, the calculated p-value was 0.259 and therefore the null hypothesis could not be rejected, which means that the result distribution is statistically the same. Based on this result, it can be concluded that all classifiers performed statistically same. Each one had their strength and weaknesses, the CNG method (except d_0) performs well with less training data, on the other hand, Support Vector Machine (SVM), MNB and CNG(d_0) perform well with more training data.

4.2 Results for RD

This section looks into the results of the second part of the research (RD - **R**ead **D**ocuments), to analyze the influence of the read documents in AA. Similar to TS, the accuracy scores for the experiments were viewed w.r.t the number of documents and word count. In our experiments, the word counts for a minimum 4 documents and maximum 7 documents per author were ≈ 3748 and ≈ 5194 respectively. These word counts for each author are under the limit of a reliable minimum of 5,000 words to be truly considered small datasets [18].

4.2.1 Results analysis of TSS20 experiments

For the experiments with only written documents by 20 authors (TSS20), the results (Table 4.4) were on the same line as with 35 authors except for CNG(d_0). Although on an average MNB (76.68%) outperformed the other classifiers, it can mostly be attributed to its accuracy score for 7 WDpA (84.37%). CNG(SPI) was the best performing classifier for 4 WDpA with an accuracy score of 73.91%. The accuracy score went up when WDpA was increased from 4 to 5, however, it dropped when the WDpA was increased from 5 to 6. The fact that the scores dropped for all the classifiers suggest its relation with the documents rather than the actual performance of any individual classifier. This will be analyzed in much more detail in the later section.

The interesting result in these experiments with 20 authors is for CNG(d_0). CNG(d_0) which had performed below par with 35 authors, has performed quite well with 20 authors. This means that CNG(d_0)'s performance is not only related to the size of the training data, but also to the number of authors. On the other hand,

Table 4.4: **TSS20 — Accuracy Scores (in %) when training only with Written Documents by 20 authors.** The table headers describe the number of documents and words used for training the classifiers.

Total Documents	93	107	120	134
Total Words/10,000	7.49636	8.4855	9.1083	10.3883
Documents per Author	4	5	6	7
Words per Author	3748	4242	4554	5194
SVM	65.22	**80.77**	65.52	71.88
MNB	69.57	76.92	**75.86**	**84.38**
CNG(d_0)	60.87	**80.77**	**75.86**	71.88
CNG(d_1)	65.22	76.92	68.97	81.25
CNG(d_2)	69.57	**80.77**	72.41	81.25
CNG(SPI)	**73.91**	76.92	65.52	71.88

CNG(d_1, d_2, and SPI)'s performance was found to be relatively better with 35 authors than 20 authors when compared to CNG(d_0), SVM and MNB with these sizes of training data.

When comparing the classifiers with each other for the statistical difference using the Friedman's test [7], it was found that the accuracy score distribution across all the classifiers was statistically same with a p-value of 0.523 (α=0.05).

4.2.2 Results analysis of RD20 experiments

There was a substantial improvement in the results for RD20 compared to TSS20 (Table 4.5). In TSS20 experiments, the classifiers were trained with only written documents by 20 authors. However, in RD20 experiments, the classifiers were trained with both written and read documents by same set of 20 author. In RD20 experiments, on average, the classifiers performed around 10.80% better when compared with TSS20 experiments. Performance of the SVM classifier was the best among all the classifiers with an average accuracy score of 94.35%, which was a drastic 23.57% increase from the previous (TSS20) experiments. There was a boost in performance all across, except one experiment with MNB (7 WDpA and 70 RDpA), where the performance dropped by 6.25%. The highest improvement was observed for SVM (6 WDpA, 60 RDpA) from 65.52% to 93.1%, a massive increase of 27.58%.

Although the classifiers performed quite well with more training data, the key

Table 4.5: **RD20 — Accuracy Scores (in %) when training with Written and Read Documents by 20 authors.** The table headers describe the number of documents and words used for training the classifiers.

Total Documents	1006	1113	1205	1286
Total Words/10,000	105.4456	117.9248	129.3709	132.0356
Documents per Author	50	56	60	64
Words per Author	52722	58962	64685	66017
SVM	**91.30**	**96.15**	**93.10**	**96.88**
MNB	86.96	92.31	**93.10**	78.13
CNG(d_0)	82.61	92.31	82.76	75.00
CNG(d_1)	73.91	76.92	68.97	87.50
CNG(d_2)	73.91	80.77	75.86	84.38
CNG(SPI)	82.61	84.62	82.76	90.63

observation was the performance when training only with 4 wdpa. While the classifiers were struggling at an average accuracy of 67.39% with TSS20 alone, their average accuracy increased to an average 81.88% with RD20. SVM was the best performing classifier (4 WDpA and 40 RDpA) with an accuracy of 91.30%. It was also confirmed through the Friedman's test that the accuracy scores for this set of experiments (RD20) are statistically different with a p-value of 0.024 (α=0.05) and SVM had the best performance with an average accuracy of 94.35%.

4.2.3 Feature analysis of TSS20 and RD20 experiments

To understand the significant improvement in scores between TSS20 and RD20, test document D21 by author A2 and D18 by author A5 were chosen which were both misclassified in 3 out of 4 TSS20 experiments, but classified correctly in all instances of RD20 (Figure 4.3, and 4.4).

Before investigating the misclassified documents in TSS20, the features were compared for the authors in both TSS20 and RD20. For the feature comparison, ELI5, a Python library for debugging machine learning classifiers and prediction analysis was used [48, 37]. Again, in this analysis the focus was on the best performing classifier with RD20 (SVM). In both TSS20 (4 WDpA) and RD20 (4 WDpA, 40 RDpA), the optimal features were word unigrams and bigrams.

Looking into the features for author A2, there is an apparent difference before

(Table 4.6-a) and after (Table 4.6-c) the addition of the read data, as new positive words with higher weight are added in the feature set. While predicting, ELI5 shows the feature-based SVM score for each author and selects the one with the highest score. This score is calculated for each author based on the relevance of its features with the given document. Although this score could be a positive or a negative number, it was generally negative in our experiments. During the experiment with only written data, author A16 had the highest SVM score (-0.955) due to the presence of feature words such as *environmental* and *indigenous* (among other top words) (Table 4.6-b) that also appeared repetitively in D21. However, this changed after the inclusion of the read documents as more feature words for author A2 such as *lng*, *ml*, and *gas* appeared in D21, which gave author A2 the highest score (-0.729).

While training only with written documents, test document D-A9 had a high frequency of words such as mr and trump (Table 4.7-b) that gave author A19 the highest score (-0.354), while author A9 received the second best score of (-0.843). However, after adding the read data, newly added feature words such as year, over, inflationary, and deficit (Table 4.7-a, 4.7-c) gave author A9 the highest score (-0.264).

Looking further into these instances, it can be found that these top words that appear after the inclusion of the read documents such as lng, gas, pacific, sales, quarter and inflationary are not particularly influencing the writing style of the author, but they are actually influencing the topics, based on which the author writes. Therefore, from these observations, it can be can concluded that the author's reading and writing are based on similar topics.

4.2.4 Performance analysis of SVM in RD20 experiments

Support Vector Machine (SVM) was the best performing classifiers in presence of the read documents (RD20) and therefore, in order to understand its performance at individual author level, the confusion matrices were compared from before (Figure 4.3) and after (Figure 4.4) the addition of the read documents. The Author ID on the left and bottom of each matrix represents the true authors and predicted authors respectively. Looking at the matrices generally, it can be clearly seen how the predicted authors align with the true authors after the addition of the read documents. In particular, the most improvement is observed while training with only 6 WDpA

Table 4.6: **Comparing top (SVM) features from document D-A4 for author A4 and A29.** Top ten word unigrams and bigrams document that was written by author A4 before and after the addition of the read documents. This documents was predicted to be written by author A29 before the addition of the read documents. However, it was classified correctly as A4 after the addition of the read documents to the training dataset.

(a) A4 - Only 4 WDpA

Weight	Feature
0.802	vancouver
0.612	price
0.503	detached
0.424	october
0.395	in october
... 2267 more positive ...	
... 29494 more negative ...	
-0.441	<BIAS>
-0.46	and
-0.475	of
-0.637	to
-0.785	the

(b) A29 - Only 4 WDpA

Weight	Feature
0.72	carbon
0.647	coal
0.504	**environmental**
0.489	**indigenous**
0.46	carbon pricing
0.395	climate
0.391	energy
... 2638 more positive ...	
... 37653 more negative ...	
-0.416	<BIAS>
-0.563	of
-0.869	the

(c) A4 - 4 WDpA and 40 RDpA

Weight	Feature
1.167	vancouver
0.784	lego
0.626	air canada
0.513	glover
0.497	**lng**
0.481	**ml**
0.459	**gas**
0.448	caplansky
0.441	cathay
... 31290 more positive ...	
... 280838 more negative ...	
-0.864	<BIAS>

versus 6 WDpA and 60 RDpA.

SVM misclassified five documents across all the RD20 experiments, of which three were misclassified even in TSS20 experiments. Although the document by author A14 was misclassified in RD20 experiment with 4 WDpA and 40 RDpA, in the experiment

Table 4.7: **Comparing top (SVM) features from document D-A9 for author A9 and A19.** Top ten word unigrams and bigrams document that was written by author A9 before and after the addition of the read documents. This documents was predicted to be written by author A19 before the addition of the read documents. However, it was classified correctly as A9 after the addition of the read documents to the training dataset.

(a) A9 - Only 4 WDpA

Weight	Feature
0.737	debt
0.736	profits
0.723	quarter
0.611	trade
0.503	business investment
0.42	second quarter
. . . 3207 more positive . . .	
. . . 41138 more negative . . .	
-0.468	of
-0.471	<BIAS>
-0.501	and
-0.533	the

(b) A19 - Only 4 WDpA

Weight	Feature
1.278	**mr trump**
1.086	**trump**
0.87	**mr**
0.563	mnuchin
0.545	mr mnuchin
0.463	he
0.408	undocumented
. . . 5543 more positive . . .	
. . . 38255 more negative . . .	
-0.371	to
-0.465	<BIAS>
-0.777	the

(c) A9 - 4 WDpA and 40 RDpA

Weight	Feature
0.723	sales
0.635	**year over**
0.609	**over year**
0.572	billion
0.564	in
0.534	quarter
0.525	rose
0.484	subsector
0.47	billion in
. . . 9284 more positive . . .	
. . . 130072 more negative . . .	
-0.945	<BIAS>

with 7 WDpA and 70 RDpA both the documents by the same author were classified correctly. On the other hand, for TSS20 experiment with 7 WDpA, both the documents by author A14 were misclassified, this could suggest that more read data could help build a stronger profile for the author. All the instances of author A14

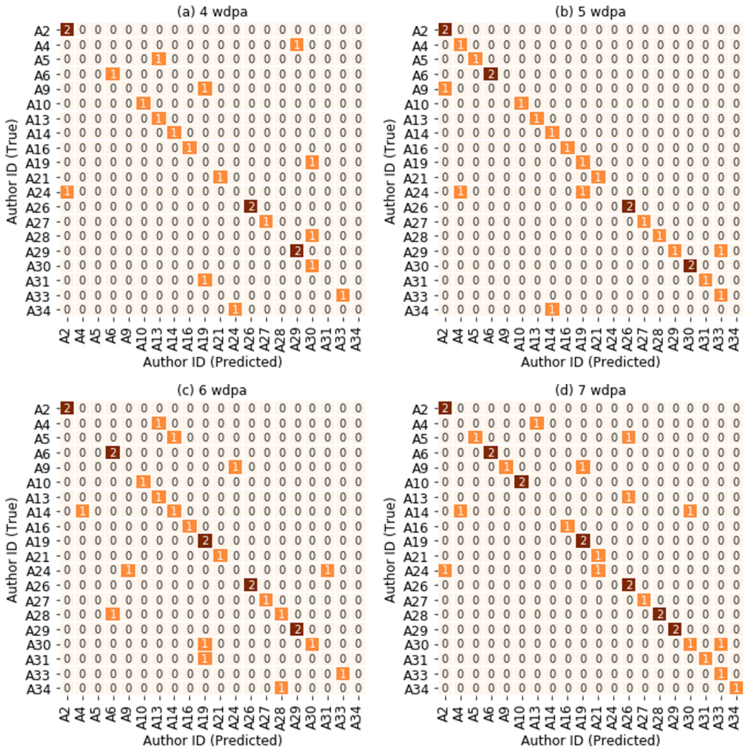


Figure 4.3: **Confusion Matrices for TSS20 experiments using SVM classifier.** Confusion Matrices displaying the predicted and true authors while training the model using SVM with only written documents and gradually increasing the WDpA in the training data from 4 up to 7 WDpA.

were classified correctly in the absence of the read documents. However, with an additional 50 RDpA in training data, the classifier confused author A27 with A6, which was classified correctly in TSS20 experiments. The misclassification was due to the presence of words such as mobile, wireless and communication in the test document that appeared more in the feature set of author A6 than A27.

It was observed that the performance of all the classifiers (including SVM) went down when training documents were increased from 5 WDpA to 6 WDpA in TSS20 experiments. The consistent drop across all the classifiers made it clear that the problem was with the feature set and not the classifiers. Therefore, continuing our analysis of SVM features and found that a lot more negative features with less weight were added to the feature set when the training data was increased from 5 WDpA to 6 WDpA. These negative features overlapped between multiple authors. Therefore, while calculating the prediction scores for each author, it was easy for a classifier to

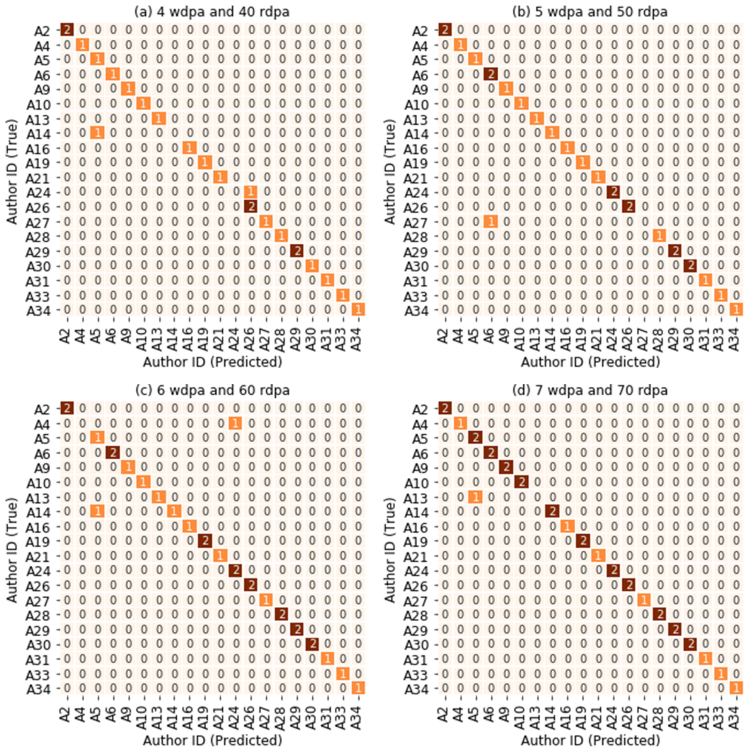


Figure 4.4: **Confusion Matrices for RD20 experiments using SVM classifier**. Confusion Matrices displaying the predicted and true authors while training the model using SVM with both written and read documents, while gradually increasing the WDpA and RDpA in the training data from 4 up to 7 and 40 up to 70 respectively.

confuse one author with another. However, after the addition of the read documents, the weight to these negative features increased and also more unique positive features were added. Both the documents that were misclassified in RD20 experiment with 6 WDpA and 60 RDpA are the same documents that were misclassified in TSS20 experiment with 6 WDpA.

4.2.5 Performance analysis of MNB in RD20 experiments

The author level performance of Multinomial Naïve Bayes (MNB) classifier was also analyzed, which was the second best performing classifier in RD20 experiments with an average accuracy score of 87.62%. The total number of misclassified instances across all the experiments for MNB reduced from 25 documents (Figure 4.5) without the read documents to 14 documents (Figure 4.6) with the read documents. There were only 2 misclassified documents in experiments with 5 and 6 WDpA in presence

of the read documents (Figure 4.6-b, c) compared to 6 and 7 misclassified documents without the read documents (Figure 4.5-b, c). MNB had the only experiment with lower accuracy score in presence of read documents when experimenting with 7 WDpA and 70 RDpA (78.13%) compared to TSS20 experiment with only 7 WDpA (84.38%). This drop in performance would be looked into greater detail while analyzing the performance of each MNB experiments. For feature analysis Python LIME module was used instead of ELI5, as ELI5 did not support the MNB estimator [34] for debugging.

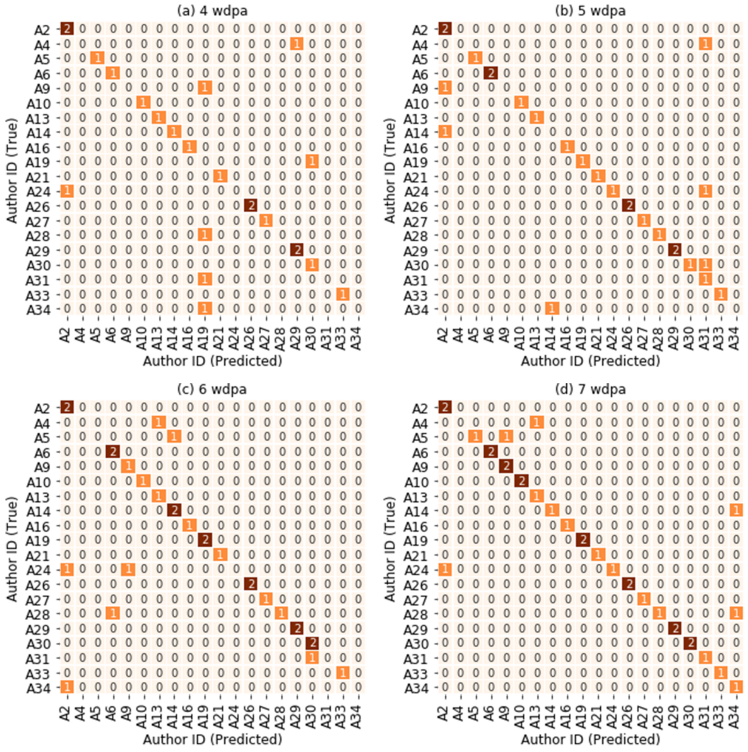


Figure 4.5: **Confusion Matrices for TSS20 experiments using MNB classifier.** Confusion Matrices displaying the predicted and true authors while training the model using MNB with only written documents and gradually increasing the WDpA in the training data from 4 up to 7 WDpA.

3 out of 14 instances were misclassified in the first RD20 experiment (Figure 4.6-a). The misclassified instance of author A4 and A24 were the same from TSS20 experiment with 4 WDpA. Therefore, the analysis focused on the only unique misclassified instance in this experiment for author A14 which was correctly classified in absence of the read documents. With MNB as the classifier, most dominant feature

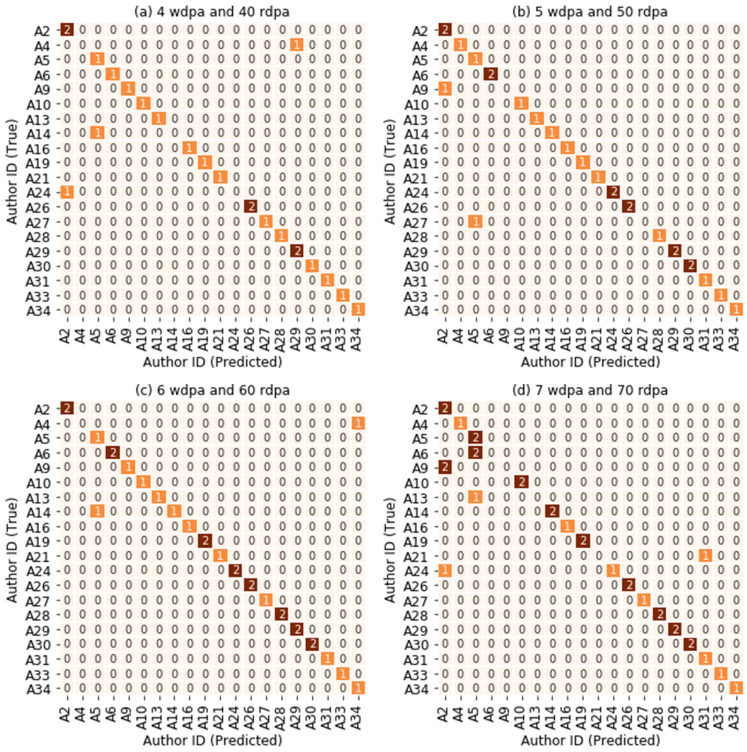


Figure 4.6: **Confusion Matrices for RD20 experiments using MNB classifier**. Confusion Matrices displaying the predicted and true authors while training the model using SVM with both written and read documents, while gradually increasing the WDpA and RDpA in the training data from 4 up to 7 and 40 up to 70 respectively.

words in favor of A14 were *digital, banking, bank, environment, traditional, studio,* and *Digital* (Figure 4.7). MNB classified the test document in favor of A14 with a probability of 0.85 in TSS20 experiments with only 4 WDpA. However, after adding the read document, this probability went down to 0.03 in favor of A14 and 0.39 in favor of A5. After the addition of the read documents, MNB found more words from the documents in A5 compared to A14 (Figure 4.8). This shows that probablistically the author of the given document is A5. However, when compared to the linear model (SVM), the classifier clearly distinguished the features for A14 from A5 and other authors for most of the experiments.

In case of the experiment with 5 WDpA in presence of the read documents, 2 documents were misclassified (Figure 4.6-b). The same document by A9 was confused

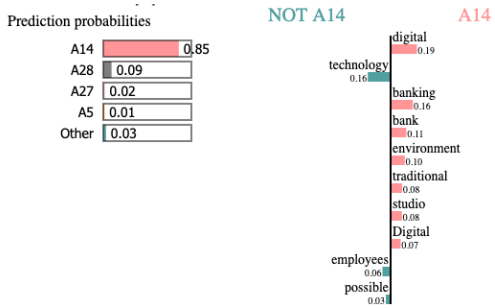


Figure 4.7: **Top features for author A14 while training with only 4 WDpA using MNB.** The prediction probabilities on the left shows the probability of the authors for the given document. The right shows the top ten features for the given document that are in favor and against author A14.

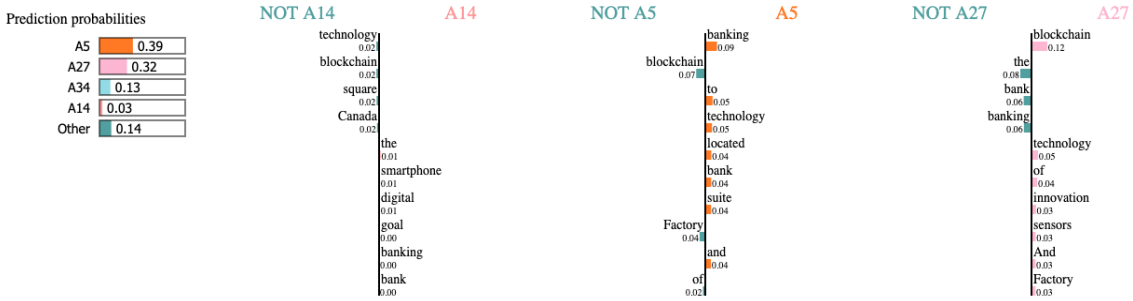


Figure 4.8: **Top features for the author A14, A5, and A27 while training with 4 WDpA and 40 RDpA using MNB**. This document was correctly classified with only written document but misclassified after the addition of the read documents.

for A2 even in the absence of the read documents. This suggests that the read documents did not add enough features to probabilistically change the decision of the classifier. However, the document by author A27 was confused for A5 after adding the read documents. It is assumed the cause for this confusion was similar to our previous investigation with 4 WDpA. However, the features were explored to confirm our assumption. Contradictory to our assumption, the misclassification was hardly due to the addition of stronger positive features for other author, but it was due to the change of a positive feature to a negative feature (*mobile*). Initially, in the absence of the read documents, *mobile* was among the top ten features for A27 with a probability of 0.11 for word mobile in favor of A27. In other words, the presence of word **mobile** (among other words) in a document, suggest that the document belongs to author A27 (Figure 4.9). However, after the addition of the read document, this probability for word *mobile* changed fromad in favor of A27 to against A27 with an

even greater probability of 0.16. This means that the presence of word *mobile* (among other words) suggests that document does not belong to author A27 (Figure 4.10). On the other hand, the word *mobile* had an even greater probability (0.17) in favor of author A5. Although there were other words that strongly favored author A27 for the test document, the higher probabilities of words such as *mobile, stock, Mr* and *Inc* favored author A5 over A27 for the given test document.

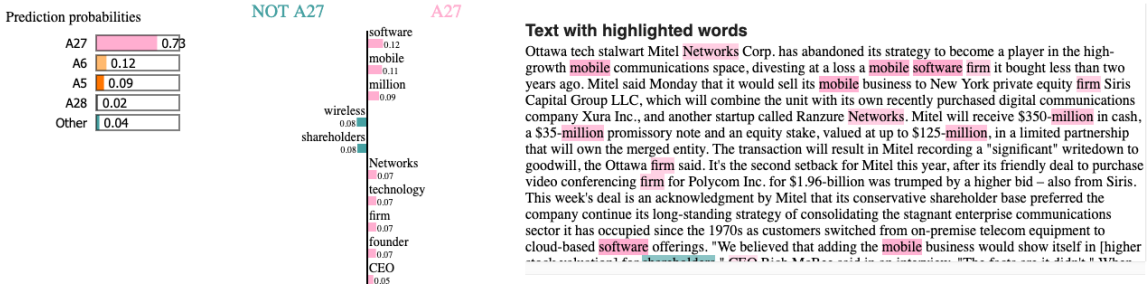


Figure 4.9: **Top features for author A27 while training with only 5 WDpA using MNB**. The prediction probabilities on the left shows the probability of the authors for the given document. The center shows the top ten features for the given document that are in favor and against author A27. The right shows the sample from the test document highlights the top features in relation to author A27.

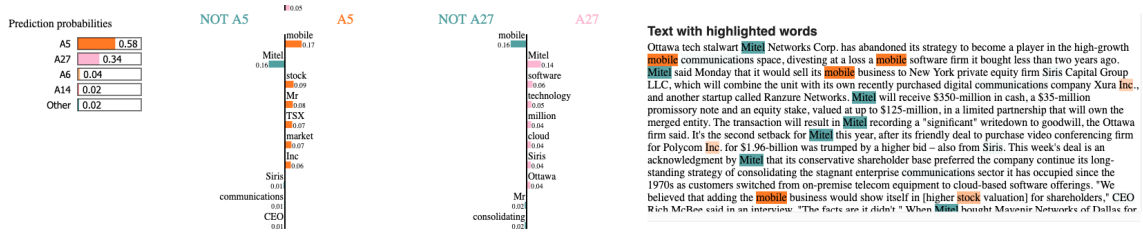


Figure 4.10: **Top features for authors A5 and A27 while training with 5 WDpA and 50 RDpA using MNB.** This document was correctly classified as A27 with only written document but it was confused for A5 after the addition of the read documents.

Two documents were misclassified in the experiment with 6 WDpA in presence of the read documents (Figure 4.6-c). The document by author A4 was confused in both the experiments, with and without the read documents (Figure 4.5-c and Figure 4.6-c). Author A14 which was confused for A5 in 4 WDpA and 40 RDpA experiment was confused again for the same author, but for a different document.

Looking into the features, the reason for misclassification was found to be same as discussed in previous case of author A27 with 5 WDpA and 50 RDpA. The addition of the read documents changed the top feature words (*BMO, mortgage, mortgages, market* and *vancouver*) for the document from in favor of author A14 (Figure 4.11) to against A14 (Figure 4.12).

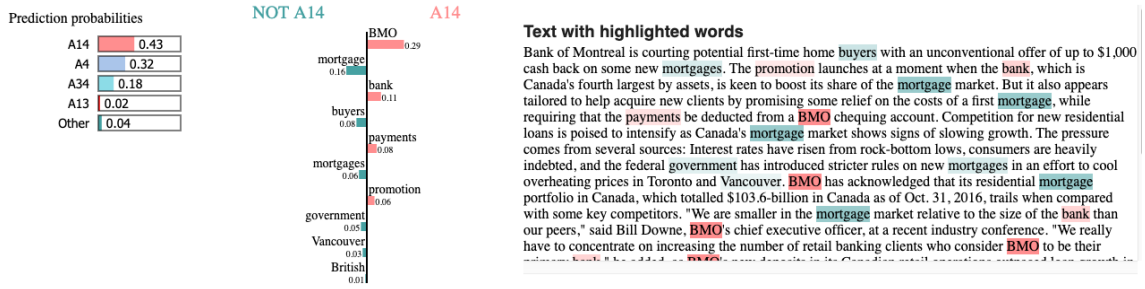


Figure 4.11: **Top features for author A14 while training with only 6 WDpA using MNB.** The prediction probabilities on the left shows the probability of the authors for the given document. The center shows the top ten features for the given document that are in favor and against author A14. The right shows the sample from the test document that highlights the top features for author A14.

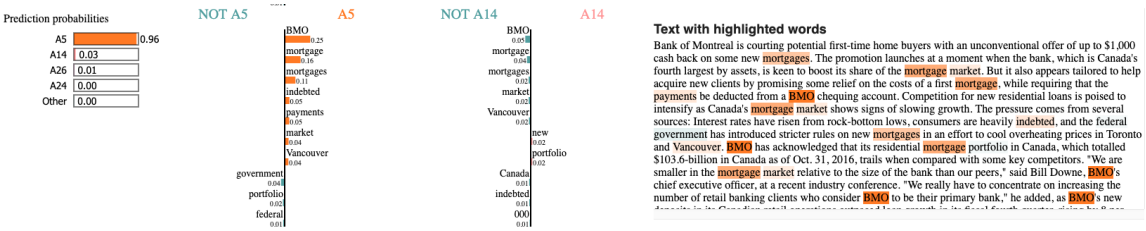


Figure 4.12: **Top features for authors A5 and A14 while training with 6 WDpA and 60 RDpA using MNB.** This document was correctly classified as A14 with only written document but it was confused for A5 after the addition of the read documents.

Finally the last set of experiment was investigated where the performance suddenly dropped to 78.13% for MNB. Compared to the MNB experiment (7 WDpA) without the read documents (Figure 4.5-d), mostly all the misclassified documents were different in presence of the read documents (Figure 4.6-d). Instead of analyzing all the 7 misclassified documents, it was decided to look into the documents by author A6 and A9 which were predicted as A5 and A2 respectively. It was realized that the

previous issues of one feature word dominating the probability of the entire document was mainly the cause of most of the misclassification. Even in case of this experiment with 7 WDpA and 70 RDpA author A6 was confused for author A5, majorly because of the presence of *Krstajic* and *Khandelwal* in first (Figure 4.13) and second (Figure 4.14) document respectively. Similar observations were made for author A9 whose both the documents were predicted as A2 after the addition of the read documents. Therefore, it was found that after the addition of the read documents, the probability of certain topic based features exceeds the probability of other features. The greater probability of a single feature in turn affects the overall probability of predicting the author of the given document.

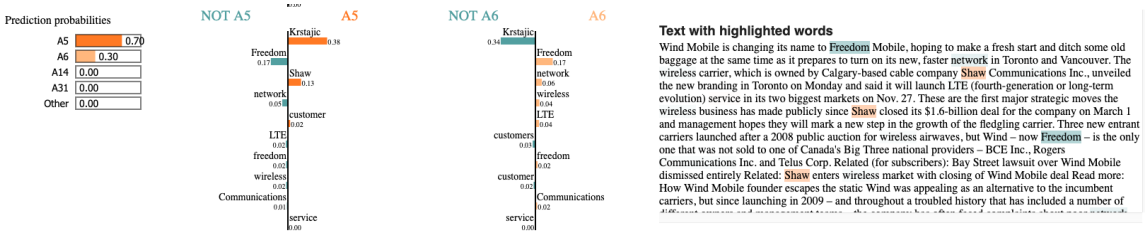


Figure 4.13: **Top features for authors A5 and A6 while training with 7 WDpA and 70 RDpA using MNB.** This is first of the two misclassified documents by author A6. This document was correctly classified as A6 with only written document but it was confused for A5 after the addition of the read documents. The top features represents the dominance of feature word *Krstajic* that favored author A5 over A6 after adding the read documents.

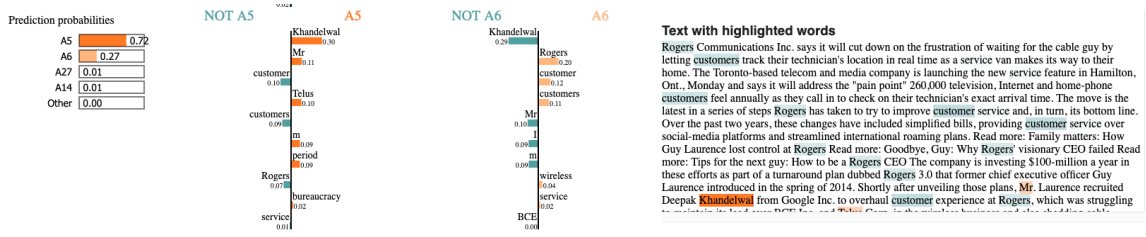


Figure 4.14: **Top features for authors A5 and A6 while training with 7 WDpA and 70 RDpA using MNB.** This is second of the two misclassified documents by author A6. This document was correctly classified as A6 with only written document but it was confused for A5 after the addition of the read documents. The top features represents the dominance of feature word *Khandelwal* that favored author A5 over A6 after adding the read documents.

Chapter 5

Conclusion

This chapter concludes our research by providing the summary, limitations, and future work of our research. The first section reviews the research and provide a summary of our problem, solution, and result. The second section discusses the shortcomings of our research and possible solutions to it. The last section discusses how the research can be improved in future by looking into the solutions for some of the limitations and analyzing the results by trying alternate techniques.

5.1 Research Summary

Authorship Attribution (AA) has evolved a lot over the past decades. With greater internet access all over the world, more and more data is generated every single day. However, getting more training data in AA is still a challenge. On the other hand, a recent study has found that the young generation consumes more and creates less data. This complements the problem of limited training data in AA, while also suggests a new alternative to the problem i.e. the use of consumed data by the user (author). Therefore, our research first analyzed the influence of the size of training data in AA. The strengths and weaknesses of less training data was also looked into. This was used as the input to the second part of the research to analyze the impact of the read documents (data consumed) in AA task.

The focus of the first part of the research (TS — **T**raining **S**ize) was to measure the influence of size of the training data on the classification performance. Through the result, it was found that all classifiers performed statistically the same. However, Common N-Grams (CNG) methods (d_1, d_2, and SPI) performs well for smaller training data. Although CNG(d_0) requires comparatively more training data, it performs consistently well with more training data. Similarly, Support Vector Machine (SVM) and Multinomial Naïve Bayes (MNB) improved in performance with more training data. From our results it can be concluded that given clean training data, the model

can pick up signals to identify the author of an unseen document without the need of any pre-processing. However, going below the 7 Written Documents per Author (WDpA), the performance of the classifiers drops by an average of 8.3%. In the second part of the research (RD) a novel approach of using the read documents by the authors is proposed for training along with their written documents to boost the classifier's performance.

From our experiments (RD — **R**ead **D**ocuments), it was found that there is a semantic similarity between the author's written and read documents. The performance of all classifiers improved significantly after adding authors' read documents to the training dataset. Although it might be challenging to get author's read data in absence of a medium (such as Twitter in our case), considering the abundance of documents read by each user, it can surely be useful for AA task. The read data enabled SVM to outperform all the other classifiers with the best accuracy of 96.15% while classifying between 20 business and investing reporters and columnists.

The features were also explored to better understand the performance of these classifiers. Support Vector Machine (SVM) turned out to be a more suitable classifier for our purpose compared to all the other classifiers. Initially, it was expected the CNG methods to outperform the other classifiers, especially the extensions suggested by Stamatatos (d_1, d_2 and SPI). However, SVM and MNB mostly outperformed them in RD. Although MNB was the second best performing classifier, its probablistic approach might not be suitable with larger number of read documents. Therefore, based on our research it can be concluded that a linear classifier such as SVM, can most reliably identify the author of a given document when the read documents are added to the training dataset along with the written documents.

Two significant contributions are made through our research. First, looking into the influence of the size of the training data in AA and how much data is enough to reliably classify a document. Second, the proposal of a novel approach of using the documents read by the authors for training along with their written documents can be used to improve the performance of Authorship Attribution task in case of limited data.

5.2 Limitations

Our research broadly looked into two things, the influence of the size of training data in authorship attribution and using the read documents to improve the performance with smaller datasets. However, there are certain limitations of the research that would be discussed in this section.

Our research used the articles by the business and investing reporters and columnists. These authors in our research are professional with very well structured form of writing. Even the articles they read or the articles used to create the read datasets were well structured and written by other professional authors. However, in real-world situation, it would be difficult to replicate our research in the areas where the articles lack structure and quality compared to the articles used in our research.

For the collection of the read documents Twitter was used as the medium. Although the results turned out surprisingly positive, it is important to note that in case of unknown authors, it would be difficult or sometimes impossible to fetch their read documents without knowing the true identity of the authors under investigation. Therefore, our research is only useful to improve the results in case of known authors with a possible medium to collect their read documents.

The overall implementation to collect the read data is comparatively challenging. In order to collect the read documents, HTML parser was created for the 45 selected domains. HTML parser is required for different domains in order to collect data from them. Also, not all domains allow the collection of data due to restriction. The data could not be collected from the domains mentioned in the category of 'Unable to download pages due to restrictions' in Table 3.1. One workaround to this limitation is the collection of the written and the read documents from the same domain. For instance, it is possible to collect the read and the written documents from a single platform such as 'Stackoverflow', 'Twitter', and 'Reddit'. However, working on the dataset collected through this medium would altogether be a different problem.

Although all possible combinations of the chosen hyper-paramters were tried, the initial choice of hyper-parameters for the experiments was intuitive. Therefore, there is a possibility of slightly different result, given a different choice of initial hyper-parameters. Also, the number of parameter combinations tried were limited due to time and computational constraint.

The ratio of the written to the read documents in the later part of the research was used as 1:10; i.e., ten read documents for every written document by the author. Therefore, staying within the scope of our research the ratio (1:10) was selected based on a few preliminary experiments, where a few combinations were tried and selected the one with the best performance. Measuring the influence of changing the written to the read documents ratio would be altogether a different research problem.

5.3 Future Work

As discussed in earier section, our major contribution through this research was towards the dataset component in Authorship Attribution task. Although the idea of using the read documents to improve the results is a novel approach, the ability to simplify the process of collecting and processing the read documents would make it possible to extend it towards practical applications.

Also, in our research the focus was only on the word and character ngrams as the features for the selected classifiers. Measuring the performance of the classifiers with different set of features could also lead to interesting results. It would be interesting to extend this research and analyze the results in the complete absence of written documents by some or all authors.

The ratio of the written to read documents in our research was set to 1:10; i.e., using ten read documents per author for every written document for the same author. This ratio was decided intuitively based on preliminary experiments. Therefore, in the future work, it was be important to measure the influence of varying ratio of written to read document. This would enable to understand, if better results can be achieved for even lower number of written or read documents using the same or new set of classifiers.

In our research, due to the focus on limited training data deep learning methods were not used. The improvement in performance of the AA task after the addition of the read documents provides us a larger training corpus. Therefore, the use of deep learning methods on this larger training corpus could facilitate the creation of a semantic profile for authors based on their read documents that could assist in the AA task.

Through our research the semantic similarity was discovered between the written

and the read documents by the authors. This can be confirmed in future research by using topic modelling approach such as Latent Dirichlet Allocation (LDA) in AA when the model is trained with both the written and the read documents.

References

[1] Ahmed Abbasi and Hsinchun Chen. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems (TOIS)*, 26(2):7, 2008.

[2] Waheed Anwar, Imran Sarwar Bajwa, M Abbas Choudhary, and Shabana Ramzan. An empirical study on forensic analysis of urdu text using lda-based authorship attribution. *IEEE Access*, 7:3224–3234, 2019.

[3] Harald Baayen, Hans van Halteren, Anneke Neijt, and Fiona Tweedie. An experiment in authorship attribution. In *6th JADT*, pages 29–37, 2002.

[4] KRW Brewer. Design-based or prediction-based inference? stratified random vs stratified balanced sampling. *International Statistical Review*, 67(1):35–47, 1999.

[5] Chad Buleen. Report: Young social users would rather consume content than create it. `https://www.clearvoice.com/blog/report-young-social-users-would-rather-consume-content-than-create-it/`. Accessed: 2019-05.

[6] Python community. Python package index. `https://pypi.org/`. Accessed: 2019-06.

[7] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

[8] Joachim Diederich, Jörg Kindermann, Edda Leopold, and Gerhard Paass. Authorship attribution with support vector machines. *Applied intelligence*, 19(1-2):109–123, 2003.

[9] Maciej Eder. Does size matter? authorship attribution, small samples, big problem. *Digital Scholarship in the Humanities*, 30(2):167–182, 2013.

[10] Michael Gamon. Linguistic correlates of style: authorship classification with deep linguistic analysis features. In *Proceedings of the 20th international conference on Computational Linguistics*, page 611. Association for Computational Linguistics, 2004.

[11] Google. Welcome to colaboratory! `https://colab.research.google.com/notebooks\/welcome.ipynb`. Accessed: 2019-06.

[12] David Haussler. *Probably approximately correct learning*. University of California, Santa Cruz, Computer Research Laboratory, 1990.

[13] Farkhund Iqbal, Hamad Binsalleeh, Benjamin CM Fung, and Mourad Debbabi. Mining writeprints from anonymous e-mails for forensic investigation. *digital investigation*, 7(1-2):56–64, 2010.

[14] Patrick Juola et al. Authorship attribution. *Foundations and Trends® in Information Retrieval*, 1(3):233–334, 2008.

[15] Vlado Kešelj, Fuchun Peng, Nick Cercone, and Calvin Thomas. N-gram-based author profiles for authorship attribution. In *Proceedings of the conference pacific association for computational linguistics, PACLING*, volume 3, pages 255–264. sn, 2003.

[16] Ashraf M Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. Multinomial naive bayes for text categorization revisited. In *Australasian Joint Conference on Artificial Intelligence*, pages 488–499. Springer, 2004.

[17] Scikit Learn. scikit-learn machine learning in python. `https://scikit-learn.org/stable/index.html`. Accessed: 2019-06.

[18] Kim Luyckx and Walter Daelemans. Authorship attribution and verification with many authors and limited data. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 513–520. Association for Computational Linguistics, 2008.

[19] Bernard Marr. How much data do we create every day? the mind-blowing stats everyone should read. Accessed: 2019-05.

[20] Florian Friedrich Martin Potthast. keselj03 - an approach to authorship attribution. `https://github.com/pan-webis-de/keselj03`. Accessed: 2018-08.

[21] Florian Friedrich Martin Potthast. stamatatos07 - an approach to authorship attribution. `https://github.com/pan-webis-de/stamatatos07`. Accessed: 2018-08.

[22] Chris Maxwell. Fcs shared research infrastructure. `https://cdn.dal.ca/content/dam/dalhousie/pdf/faculty/computerscience/\services/dalfcs-shared-research-infrastructure-2016-03-01.pdf`. Accessed: 2019-06.

[23] Thomas Corwin Mendenhall. The characteristic curves of composition. *Science*, 9(214):237–249, 1887.

[24] Frederick Mosteller and David Wallace. Inference and disputed authorship: The federalist. 1964.

[25] John Nerbonne. The exact analysis of text. *Foreword to the 3rd edition of Frederick Mosteller and David Wallace Inference and Disputed Authorship: The Federalist Papers CSLI: Stanford*, 2007.

[26] Faculty of Computer Science. Technical support resources. https://www.dal.ca/faculty/computerscience/current/technical_resources.html\. Accessed: 2019-06.

[27] Neil Patel. 38 content marketing stats that every marketer needs to know. https://neilpatel.com/blog/38-content-marketing-stats-that-every-marketer-needs-to-know/. Accessed: 2019-05.

[28] Fuchun Peng and Dale Schuurmans. Combining naive bayes and n-gram language models for text classification. In *European Conference on Information Retrieval*, pages 335–350. Springer, 2003.

[29] Christo Petrov. Big data statistics 2019. https://techjury.net/stats-about/big-data-statistics/. Accessed: 2019-05.

[30] Shanta Phani, Shibamouli Lahiri, and Arindam Biswas. Authorship attribution in bengali language. In *Proceedings of the 12th International Conference on Natural Language Processing*, pages 100–105, 2015.

[31] Tieyun Qian, Bing Liu, Li Chen, and Zhiyong Peng. Tri-training for authorship attribution with limited training data. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 345–351, 2014.

[32] Francisco Rangel, Paolo Rosso, Moshe Koppel, Efstathios Stamatatos, and Giacomo Inches. Overview of the author profiling task at pan 2013. In *CLEF Conference on Multilingual and Multimodal Information Access Evaluation*, pages 352–365. CELCT, 2013.

[33] Dylan Rhodes. Author attribution with cnns. *Avaiable online: https://www. semanticscholar. org/paper/Author-Attribution-with-Cnn-s-Rhodes/0a904f9d6b47dfc574f681f4d3b41bd840871b6f/pdf (accessed on 22 August 2016)*, 2015.

[34] Marco Tulio Ribeiro. lime 0.1.1.34. https://pypi.org/project/lime/. Accessed: 2019-05.

[35] Ruchita Sarawgi, Kailash Gajulapalli, and Yejin Choi. Gender attribution: tracing stylometric evidence beyond topic and genre. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 78–86. Association for Computational Linguistics, 2011.

[36] Roy Schwartz, Oren Tsur, Ari Rappoport, and Moshe Koppel. Authorship attribution of micro-messages. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1880–1891, 2013.

[37] scikit-learn developers. Debugging scikit-learn text classification pipeline. https://eli5.readthedocs.io/en/latest/tutorials/sklearn-text.html. Accessed: 2019-05.

[38] scikit-learn developers. sklearn.feature_extraction.text.tfidfvectorizer. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html. Accessed: 2019-05.

[39] scikit-learn developers. sklearn.linear_model.sgdclassifier. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html. Accessed: 2019-05.

[40] scikit-learn developers. sklearn.metrics.accuracy_score. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html. Accessed: 2019-05.

[41] scikit-learn developers. sklearn.metrics.confusion_matrix. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html. Accessed: 2019-05.

[42] scikit-learn developers. sklearn.model_selection.gridsearchcv. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html. Accessed: 2019-05.

[43] scikit-learn developers. sklearn.naive_bayes.multinomialnb. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html. Accessed: 2019-05.

[44] Prasha Shrestha, Sebastian Sierra, Fabio Gonzalez, Manuel Montes, Paolo Rosso, and Thamar Solorio. Convolutional neural networks for authorship attribution of short texts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 669–674, 2017.

[45] Thamar Solorio, Sangita Pillay, Sindhu Raghavan, and Manuel Montes-Gomez. Modality specific meta features for authorship attribution in web forum posts. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 156–164, 2011.

[46] Eugene H Spafford and Stephen A Weeber. Software forensics: Can we track code to its authors? *Computers & Security*, 12(6):585–595, 1993.

[47] Efstathios Stamatatos. Author identification using imbalanced and limited training texts. In *18th International Workshop on Database and Expert Systems Applications (DEXA 2007)*, pages 237–241. IEEE, 2007.

[48] TeamHG-Memex. A library for debugging/inspecting machine learning classifiers and explaining their predictions. https://github.com/TeamHG-Memex/eli5. Accessed: 2019-05.