Style Change Detection

Thesis

Sukanya Nath

Department of Computer Science,
Faculty of Science,
University of Neuchâtel

October 2021

under the supervision of
Prof. Dr. Jacques Savoy

Department of Computer Science,
Faculty of Science,
University of Neuchâtel

UNIVERSITÉ DE NEUCHÂTEL

Faculté des Sciences
Secrétariat-décanat de Faculté
Rue Emile-Argand 11
2000 Neuchâtel – Suisse
Tél : + 41 (0)32 718 21 00
E-mail : secretariat.sciences@unine.ch

IMPRIMATUR POUR THESE DE DOCTORAT

La Faculté des sciences de l'Université de Neuchâtel autorise l'impression de la présente thèse soutenue par

Madame Sukanya NATH

Titre:

"Style Change Detection"

sur le rapport des membres du jury composé comme suit:

- Prof. Jacques Savoy, directeur de thèse, Université de Neuchâtel, Suisse
- Dr Valerio Schiavoni, Université de Neuchâtel, Suisse
- Prof. Josiane Mothe, IRIT, Toulouse, France
- Prof. Mascha Kurpicz-Briki, Berner Fachhoschule Technique et informatique, Bienne, Suisse

Neuchâtel, le 18 novembre 2021 Le Doyen, Prof. A. Bangerter

Abstract

Le domaine de la stylométrie recouvre l'étude des styles d'écriture dans le but de pouvoir attribuer le nom de l'auteur à texte, ou de vérifier si cet auteur est bien la personne nommée ainsi que différentes questions concernant le profilage d'auteur. En analysant les caractéristiques stylistiques d'un texte donné, la stylométrie cherche à déterminer ceux qui sont propres à son auteur et pouvant le discriminer des autres écrivains. La détection de différents styles d'écriture au sein d'un même document implique la présence d'auteurs multiples et cette démarche se nomme la détection de changements de style (ou style change detection, SCD). Une telle détection s'avère particulièrement ardue parce que le nombre d'auteurs n'est pas connu d'avance. De plus nous ne disposons pas de corpus de référence pour chaque auteur potentiel. Le but de cette thèse est d'améliorer les techniques existantes et de proposer de nouveaux modèles capables de discriminer les styles d'écriture en présence de documents ayant plusieurs auteurs et ceci d'une manière simple et pratique. Nous abordons ce problème en le décomposant en trois sous-problèmes. En premier, on doit déterminer si le document a été écrit par une ou plusieurs personnes. Une classification binaire est proposée dans laquelle chaque document est transformé en une matrice de caractéristiques. Cette dernière est alors utilisée comme base pour divers modèles d'apprentissage comme la régression logistique, les machines à vecteurs de support, des forêts aléatoires ou des réseaux de neurones. Si un document est écrit par plusieurs personnes, la deuxième tâche est de déterminer les positions où l'on peut observer un changement de style, sous l'hypothèse que de telles changements ne s'opèrent d'entre les paragraphes. Notre solution propose de voir ceci comme un problème de vérification dans lequel on compare le style de deux paragraphes. Le document est alors subdivisé en paragraphes et, à l'aide de stratégies de word embeddings (e.g., GloVe), un vecteur de caractéristiques peut être associé à chaque paragraphe. Avec une paire de telle représentation, on entraîne un modèle de réseaux de neurones siamesques et une mesure de distance permet de définir leur similarité. Le troisième problème comprend une question touchant le clustering d'auteurs afin de définir le nombre d'auteurs d'un document. En faisant l'hypothèse que les changements de style s'opèrent uniquement entre paragraphes, nous proposons deux algorithmes nommées Threshold Based Clustering (TBC) et Window Merge Clustering (WMC). L'idée générale consiste à segmenter le document en une séquence de textes nommée fenêtres. Chacune est ensuite convertit en un vecteur de caractéristiques. On peut alors mesurer la distance entre de telles représentations que l'on regroupe dans une matrice. L'approche TBC trie les pairs de fenêtres en fonction de leur distance et, ensuite, regroupe la paire la plus similaire pour en former une grappe. Ce processus est répété tant que la similarité reste plus élevée qu'un seuil fixé. Finalement, le nombre de grappe indique le nombre d'auteurs. Notre solution WMC commence de la même manière en générant des grappes avec les paires de fenêtres les plus similaire. Cependant, à chaque itération, la représentation de la grappe est reconstruite sur la base de tous ses membres. Ainsi, le calcul de similarité ne s'effectue plus entre tous les membres d'une grappe avec ceux de la seconde, mais uniquement en fonction de la représentation unique de chaque grappe. Ensuite la matrice

des distances est recalculée après chaque itération. Comme variation à ce dernier problème, on peut assigner à chaque auteur les paragraphes qu'il aurait écrit. En respectant notre hypothèse sur le changement de style, on peut recourir à la position des changements de style obtenue lors de la résolution du deuxième problème. Ensuite, une approche basée sur notre modèle TBS et une classification hiérarchique nous permet de déterminer le nombre d'auteurs. Nos modèles ont été évalués lors des campagnes internationales PAN CLEF et ont obtenu des performances similaires à l'état de l'art.

Keywords— Stylométrie, changement de style, apprentissage automatique, évaluation, PAN CLEF

Abstract

Stylometry is the study of writing styles of authors aiming for authorship attribution, verification, identification and profiling among others. By analyzing the stylometric features in a given text, the characteristic writing style of an author is represented and sufficiently distinguished from another.

The detection of different writing styles in the same document, suggesting multiple authorship, is called style change detection (SCD). Detecting multiple authorship is considerably challenging because the number of participating authors is not known apriori and because of the lack of additional reference corpus. The goal of this thesis is to leverage existing stylometry techniques and devise novel methods to distinguish the writing styles of authors in a multi-authored document in a simple and practical manner.

We address this problem by decomposing it into three sub-problems. At first, we need to determine whether a document is written by one or more authors. A binary classification approach is taken where each document is transformed into a suitable feature matrix and fed to a variety of learning models such as Logistic Regression, support vector machines, Random Forest and neural networks etc.

If a document is written by multiple authors, the second sub-problem is to determine the location of the style changes in the document, under the assumption that style changes may occur only at the end of a paragraph but not within a paragraph. Our approach is to transform the problem to an authorship verification problem, where the stylistic difference between the paragraphs compared, i.e., either the two paragraphs are stylistically different or they are not. The document is broken into paragraphs and with the help of word embeddings such as GloVe, the embedded feature vector for each paragraph is computed. Both feature vectors are then trained with a siamese neural network and their mutual distance is measured with a suitable distance measure.

The third sub-problem is a form of authorship clustering and seeks to ascertain the number of distinct authors of the text. We start with the assumption that style changes may occur within a paragraph and propose two algorithms called Threshold Based Clustering (TBC) and Window Merge Clustering (WMC). The general approach is to segment the document in to chunks of texts called windows. Each window is converted into a feature vector of words or more generally of stylistic patterns. The mutual distances among the window feature vectors are measured using suitable distance measures, and a distance matrix is created for all the windows. The Threshold Based Clustering (TBC) algorithm sorts the pairs of windows in terms of their distance and puts the closest windows to the same cluster using appropriate thresholds for adding a new node and merging clusters. The number of clusters indicates the number of authors.

Window Merge Clustering (WMC) algorithm starts out like the TBC and iteratively puts the closest windows in the same cluster. However, in each iteration, the windows in a cluster are merged to form a concatenated cluster in order to represent each cluster with a combined representation of all of its members together, rather than individual distances. Thus, the

distance matrix is re-calculated at each iteration.

A variation of the third task aims to assign texts based on style changes to their respective authors uniquely. Under the assumption that style changes may occur only at between the paragraphs but not within them, we propose to use the style change location results derived in the solution of the second problem. Thereafter, clustering approaches based on TBC and hierarchical clustering are used to determine the number of clusters or authors.

We evaluate and our methods on the datasets of the PAN CLEF and show that we can achieve state-of-the-art performance.

Keywords— Stylometry, style change detection, machine learning, evaluation, PAN CLEF

Contents

1 Introduction **1**
- 1.1 Goals . . . 4
- 1.2 Contributions . . . 4
- 1.3 Organization of the thesis . . . 5

2 Background **9**
- 2.1 Overview of SCD approaches at PAN CLEF . . . 9
- 2.2 Feature Selection in Authorship Studies . . . 11
- 2.3 Neural Networks in Authorship Studies . . . 14
  - 2.3.1 Feed Forward Neural Networks (FFNN) . . . 15
  - 2.3.2 Convolutional Neural Network . . . 15
  - 2.3.3 Recurrent Neural Network (RNN) . . . 16
- 2.4 Distance Measures . . . 19

3 PAN 2018 **21**
- 3.1 Dataset . . . 21
- 3.2 Feature Selection . . . 22
- 3.3 Classifier Choices . . . 23
  - 3.3.1 Logistic Regression . . . 23
  - 3.3.2 Naïve Bayes . . . 23
  - 3.3.3 SVM . . . 23
  - 3.3.4 Random Forest . . . 23
  - 3.3.5 Neural Network models . . . 24
  - 3.3.6 Simple Rule Based Classifier . . . 24
- 3.4 Results . . . 24

4 PAN 2019 **27**
- 4.1 Dataset . . . 27
  - 4.1.1 Duplicate Sentences . . . 28
- 4.2 Single or Multiple Author task . . . 29
  - 4.2.1 Simple Rule Based Classifier . . . 29
  - 4.2.2 Results . . . 29
- 4.3 Authorship Clustering (Number of Authors) task . . . 29
  - 4.3.1 Feature Selection . . . 31
  - 4.3.2 Definitions . . . 32
  - 4.3.3 Models . . . 33
    - 4.3.3.1 Threshold Based Clustering Algorithm . . . 35
    - 4.3.3.2 Window Merge Clustering Algorithm . . . 36
  - 4.3.4 Evaluation Measure . . . 36
  - 4.3.5 Results . . . 37

5 PAN 2021 and others **39**
5.1 Dataset . . . 39
5.2 Single or Multiple Author Task . . . 40
5.2.1 Results . . . 40
5.3 Paragraph-Level Style Change Location Detection task . . . 44
5.3.1 Data preprocessing . . . 45
5.3.2 Siamese neural networks (SNN) . . . 45
5.3.3 Logistic Regression (BOW) Model . . . 46
5.3.4 Models . . . 46
5.3.4.1 Siamese neural network architecture . . . 48
5.3.4.2 Logistic Regression (BOW) Model . . . 49
5.3.5 Results . . . 49
5.3.6 Blog Authorship Dataset . . . 49
5.4 Authorship Clustering (Authorship Assignment) task . . . 50
5.4.1 Simple Clustering . . . 51
5.4.2 Simple Threshold Based Clustering . . . 51
5.4.3 Hybrid Threshold Based Clustering . . . 51
5.4.4 Agglomerative Clustering . . . 51
5.4.5 Results . . . 52

6 PAN 2020 **53**
6.1 Dataset . . . 53
6.2 Single or Multiple Author Task . . . 54
6.2.1 Results . . . 54
6.3 Paragraph-Level Style Change Location Detection task . . . 57
6.3.1 Results . . . 57

7 Discussion **63**
7.1 Limitations of the Studied Models . . . 64
7.2 Dataset Limitations . . . 64
7.3 Future Work . . . 65

8 Conclusion **67**

List of Figures

1.1 SCD as authorship diarization and authorship clustering . . . 3
1.2 Example problem . . . 6
1.3 Example problem with solution . . . 7

2.1 A Multi-Layer Perceptron . . . 15
2.2 A Convolutional Neural Network . . . 16
2.3 A Recurrent Neural Network . . . 17
2.4 A Long Short-Term Memory unit cell . . . 17
2.5 A Gated Recurrent Unit cell . . . 18
2.6 A Siamese Neural Network . . . 19

3.1 Document length vs style change in PAN 2018 dataset . . . 22
3.2 Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2018 . . . 26
3.3 Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2018 . . . 26

4.1 Number of duplicates grouped by authors in the training set . . . 28
4.2 Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2019 . . . 31
4.3 Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2019 . . . 32
4.4 Clusters representing distance between windows . . . 32
4.5 Distance representation between windows . . . 32
4.6 Hierarchical clusters created by WMC . . . 35

5.1 Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2021 . . . 42
5.2 Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2021 . . . 44
5.3 The selected features of the MFW baseline model . . . 47
5.4 Siamese network architecture . . . 47
5.5 MFW baseline model architecture . . . 48

6.1 Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2020 Dataset-Narrow . . . 57
6.2 Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2020 Dataset-Narrow . . . 58
6.3 Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2020 dataset-wide . . . 60
6.4 Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2020 dataset-wide 61

List of Tables

2.1 Summary of approaches in PAN SCD . . . 12

3.1 Dataset statistics for PAN 2018 . . . 22
3.2 Single or Multiple Author Task Results for PAN 2018 dataset . . . 25

4.1 Dataset statistics for PAN 2019 . . . 27
4.2 Duplicates in documents (Training set) . . . 28
4.3 Single or Multiple Author Task Results for PAN 2019 dataset . . . 30
4.4 Example of a Distance Matrix . . . 31
4.5 Sorted Distance Array . . . 35
4.6 Number of authors task for PAN 2019 dataset - Initial Results (Manhattan distance) . . . 37
4.7 Number of authors task for PAN 2019 dataset - Improved Results using Duplicated Sentences (Manhattan distance) . . . 37
4.8 Comparison of distance measures for TBC algorithm . . . 38
4.9 Comparison of distance measures for WMC algorithm . . . 38

5.1 Dataset statistics for PAN 2021 . . . 40
5.2 Single or Multiple Author Task Results for PAN 2021 (Accuracy) . . . 41
5.3 Single or Multiple Author Task Results for PAN 2021 (Balanced Accuracy) . . . 43
5.4 Intermediate data representation of a document . . . 46
5.5 Finished data representation of a document . . . 46
5.6 Paragraph-level Style Change Location Detection Results for PAN2021 dataset . . . 49
5.7 Segment Length quantile analysis (character length) . . . 50
5.8 Dataset statistics for Blog Authorship SCD corpus . . . 50
5.9 Paragraph-level Style Change Location Detection Results for Blog Authorship corpus . . . 51
5.10 Example of a Distance Matrix for Hybrid Threshold Based Clustering . . . 51
5.11 Authorship Clustering Results for PAN 2021 dataset . . . 52

6.1 Dataset statistics for PAN 2020 . . . 54
6.2 Single or Multiple Author Task Results for PAN 2020 Dataset-Narrow . . . 55
6.3 Single or Multiple Author Task Results for PAN 2020 Dataset-Wide . . . 56
6.4 Paragraph-level Style Change Location Detection Results for PAN2020 Dataset-Narrow (Original) . . . 58
6.5 Paragraph-level Style Change Location Detection Results for PAN2020 Dataset-Wide (Original) . . . 59
6.6 Paragraph-level Style Change Location Detection Results for PAN2020 Dataset-Narrow (Balanced) . . . 59
6.7 Paragraph-level Style Change Location Detection Results for PAN2020 dataset-wide (Balanced) . . . 59

1 Introduction

Text data, among other forms of data, continue to be generated at an unprecedented, dramatic rate. As such, the possibilities of creating useful knowledge from text data are enormous. Some applications of text processing include information retrieval, information extraction, question answering, chatbots and dialogue systems, machine translation, automatic speech recognition and text to speech systems, among others.

However, language and thereby text data is inherently ambiguous, unstructured, and noisy. Language is influenced by a range of factors. For instance, languages evolve over time, introducing newer words while others become obsolete. Languages are also affected by location, giving rise to dialects or slang words and phrases. The medium of communication may influence the colloquialism, such as, the style of writing for a newspaper is markedly different from an informal chat. Additionally, errors may also be introduced because of misspellings or misuse. Despite generating a large amount of data, clean and labelled data may not be readily available for a specific application. The goal of Natural Language Processing (NLP) is to overcome such challenges using linguistics and computer science to enable computers in analyzing and understanding texts.

Authorship analysis is an area of NLP to automate the identification and profiling of authors by distinguishing the individual writing styles of authors. We define the author of a document as the person who is the true creator of a text and excludes anyone involved in modifications such as editing, summarizing, paraphrasing, or rephrasing. The author is also not the person who imagines the story or proposes the general scenario. Authorship style is defined as the choice of written expression including words, the punctuation, the sentence structure, grammatical structure, new paragraph layout, and other idiosyncrasies that are unique to the author. The question that arises from such a definition is that, does the individual style of an author remain stable and unchanging? The answer is not really. We can surely observe the continuous evolution of writing style from childhood to adolescence and further to adulthood of say, 25 years. After this period, a person tends to define a clear personal style that is relatively stable. Some changes could occur over a larger time period (e.g. a decade). Moreover, an author may choose to write about certain specific topics. In order to differentiate personal style from a topical discussion, we should ignore the influence brought about by the choice of topics.

Now, the question of authorship arises because we do not know the true author of a document. Thus, we seek to establish authorship by other means such as, by referencing historical facts, stylometry and

reasoning. The loss of the knowledge of authorship could be unintentional (lost in the passage of time) or deliberate. By hiding one's identity as an author, one may also seek protection behind anonymity. There are many well known cases of female authors writing behind male pseudonyms in the nineteenth century. For instance, Mary Ann Evans assumed the nom de plume George Eliot to avoid ridicule and be taken seriously [1]. More recently, J. K. Rowling authored the book "The Cuckoo's Calling" in the crime genre under the pen name Robert Galbraith in 2013. Juola [2] proposed an analytical approach to study the Rowling case. By pretending to be someone else, an author may seek to mislead the others for benefit (for example fraud) or may seek to take credit for something written by others (plagiarism).

Therefore, it is essential to ascertain authorship to not only ascribe credit to the author, but also to assign responsibility and accountability appropriately. The many applications of authorship analysis include historical scholarship, plagiarism detection, fake news detection, forensics, bot detection and marketing. Authorship analysis, however, is a challenging area of research primarily because of the lack of sufficient writing samples of an author. Further, it may not be possible to have a list of all possible candidate authors.

This thesis focuses on a sub area of authorship analysis which deals with detecting collaborative authorship or multiple writing styles in the same document called style change detection (SCD). Multiple authorship is not a recent phenomenon. Mosteller et al. in [3] used applied Bayesian and classical inference to the case of the Federalist papers. There has been a increasing trend of collaboration in academia leading to an increase in multiple authorship [4–6].

As SCD in a multi-authored document focuses on its intrinsic characteristics, it is close to the problem of intrinsic plagiarism. Intrinsic plagiarism detection inspects the writing style consistency of the author over the length of the document and provides the advantage of being independent of any other reference corpus. However, the contributions of all the authors is not equal. Some authors may contribute significantly, while others may contribute rather little. Besides, the lack of additional writing samples of the authors remains a challenge in detecting multiple authorship.

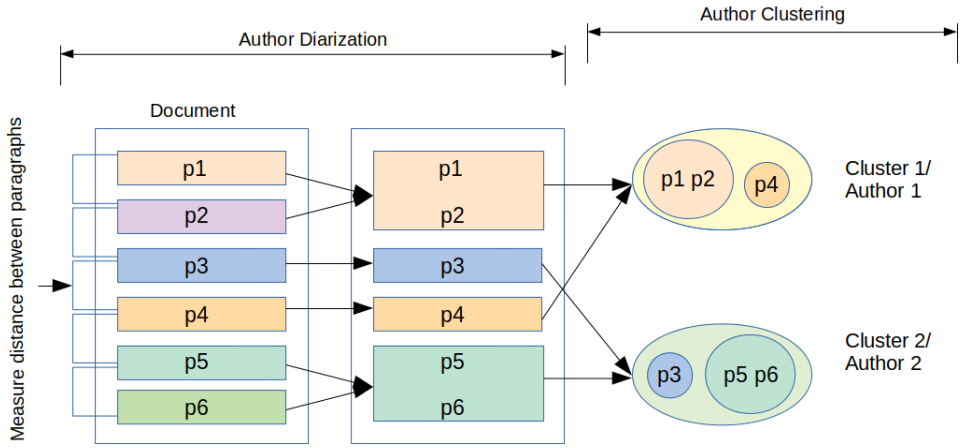
SCD can be defined as a combination of authorship diarization and authorship clustering or authorship linking. Authorship diarization is the problem of breaking a document written by multiple authors into text segments, where one segment is written by a single author. The problem is conceptually similar to the problem of speaker diarization [7] in automated speech processing, where the task is to assign speakers to their respective speech segments. Drawing analogy from speaker diarization, also termed as "Who spoke when?", we may state that author diarization may be termed as "Who wrote where?". Author diarization [8] covers both intrinsic plagiarism detection and within-document clustering problems. Intrinsic plagiarism methods use only the characteristics of the given document and therefore are independent of any external corpora. However, detecting intrinsic plagiarism is a challenging task because of the lack of any external text samples by the authors. A common approach is to break the document into fixed length segments and compare the stylistic distances between the two chunks with a suitable threshold. While this approach is easy to interpret, there are difficulties in choosing the right size of the chunk and a fixed threshold. If the chunk is too large, we risk containing the style changes within a single chunk. If, on the other hand, the chunk is too small, we risk having too little text to extract features and thereby, increase our risk of misclassification. To simplify the challenge, a reasonable assumption can be made that each author writes at least a paragraph and thereby style changes may occur at the end of a paragraph but not within a paragraph.

Authorship linking [9] aims to find the pairs of documents written by the same author in an unsupervised manner, given a sample of documents and an unknown number of authors without any training information. In this case, each document was written by a single author. A close relative of authorship linking is authorship clustering, which seeks to determine the number of distinct authors in a collection of texts. Both authorship clustering and authorship linking are closely related to the authorship verification problem, since they can be broken down into a set of verifications if the two documents are written by the same author or not. Koppel [10] goes on to say that authorship verification is a one-class classification problem,

since negative examples are abundant and are not representative of the entire class. In case of authorship verification, training information may be given with long texts. This is different from the requirement of the SCD problem, where no training information is available and the length of the chunks are rather small (paragraph size).

In Figure 1.1, the break-down of the SCD problem is shown in terms of authorship diarization and authorship clustering/ linking.

Figure 1.1: SCD as authorship diarization and authorship clustering



In Figure 1.2, we show an example multi-author problem from the PAN collection [11] under the assumption that style changes may occur at the end of the paragraphs only. The discussion seems to be about choice of infrastructure tools in a devops environment. On reading the document, it may not be evident how many authors are present. To get an understanding of the human level accuracy of this task, we can attempt to connect certain paragraphs as written by the same author. For example, the discussion of "application-level metrics" continues from paragraph 2 to paragraph 3 which may be an indication of the same author. Paragraphs 1 and 3 also mention percentages - 100% and 10%. Similarly, paragraphs 8, 9 and 10 talk about configuring services for automation and the use of service discovery (Consul) and configuration management tool (Chef). We can also observe that in paragraph 4, the author tends to mention a lot of numbers such as 10x, 20 req/s, 1200, 12 etc. In Figure 1.3, we highlight the sections written by the same author. Author A1 has written paragraphs 1, 2, 3, 8, 9 , 10 and 11, whereas author A2 has written paragraphs 4,5,6 and 7 and author A3 has written paragraph 12. This example highlights the challenging nature of the problem, although it may be possible to estimate the change of authorship in certain cases by human readers.

The SCD problem can be formulated as three sub-problems.

1. **Single or Multiple Author** The first sub-problem determines whether a document is written by one or more authors. A common approach is binary classification, where each document is transformed into a suitable feature matrix and fed to a variety of statistical and machine learning models and their ensembles.

2. **Style Change Location Detection** The second sub-problem is to determine the location of the style changes in the document, given that multiple authors are involved. A major challenge in this problem is the multi-label and variable length output nature of classification. One approach to overcome this difficulty is to break the document up into chunks of text and comparing stylistic differences between them. In a strict formulation, it is assumed that the style changes may occur only within

the paragraphs. This formulation is challenging because the optimal size of the chunks must be estimated without any indications. A reasonable assumption and thereby an easier formulation is to assume that style changes may occur only at the end of the paragraphs. This assumption stems from the idea that each paragraph dwells on a continuous thread of thought by a single author. Such a formulation allows the paragraph sized chunks. We call this formulation as the paragraph-level style change location detection problem.

3. **Authorship Clustering** In the third sub-problem the number of distinct authors is determined by authorship clustering within documents. We shall refer to this task as the number of authors task. In a variation of the third task, the texts written by each author must be isolated and assigned to them uniquely. We shall refer to this task as the authorship assignment task. As in the case of the second sub-problem, the common approach here is to split the text into chunks, then to calculate the distance between the chunks and then group the chunks according to the clustering criteria. Outliers may be identified and dropped followed by some post-processing steps. .

Two other related problems of authorship are authorship profiling and authorship obfuscation. In authorship profiling, the objective is to define some demographic characteristics of the author such as gender, age, bot or human, social class, native language, location, among others. In case of authorship obfuscation, the objective is to prevent detection by deliberately mask their writing style.

1.1 Goals

The aim of this thesis is to investigate the following research questions:

1. Define the problem of SCD and its variations.
2. Are existing feature selection strategies suitable for the problem of SCD? How effective are word embedding approaches?
3. Compare existing methods and devise novel methods to address SCD problem and discuss the advantages and limitations of the approach.
4. How to suitably evaluate the comparison of proposed models?
5. Are there sufficient datasets for evaluation? If not, how to address the problem of lack of data?

1.2 Contributions

In this thesis, an end to end approach for the SCD problem is presented. We list our contributions as follows:

1. We broke down the SCD problem into a series of sub problems and provide solutions and evaluations to them individually.
2. We compared different feature selection strategies including word embeddings while creating feature vectors and observed their influence in the performance of the models.
3. We cast the problem of detecting the locations of style changes in a document (under the assumption that style changes occur at the end of a paragraph and not within the paragraphs) as a one-shot learning problem and introduce neural network based approaches to compare the stylistic similarities.

4. We proposed (TBC and WMC) and compared clustering approaches for detecting multiple authorship in a document under both assumptions:
   (a) The style changes may occur within the paragraph.
   (b) The style changes occur within at the end of the paragraph only.
5. We created a new dataset based on the Blog Authorship Corpus for the SCD problem.

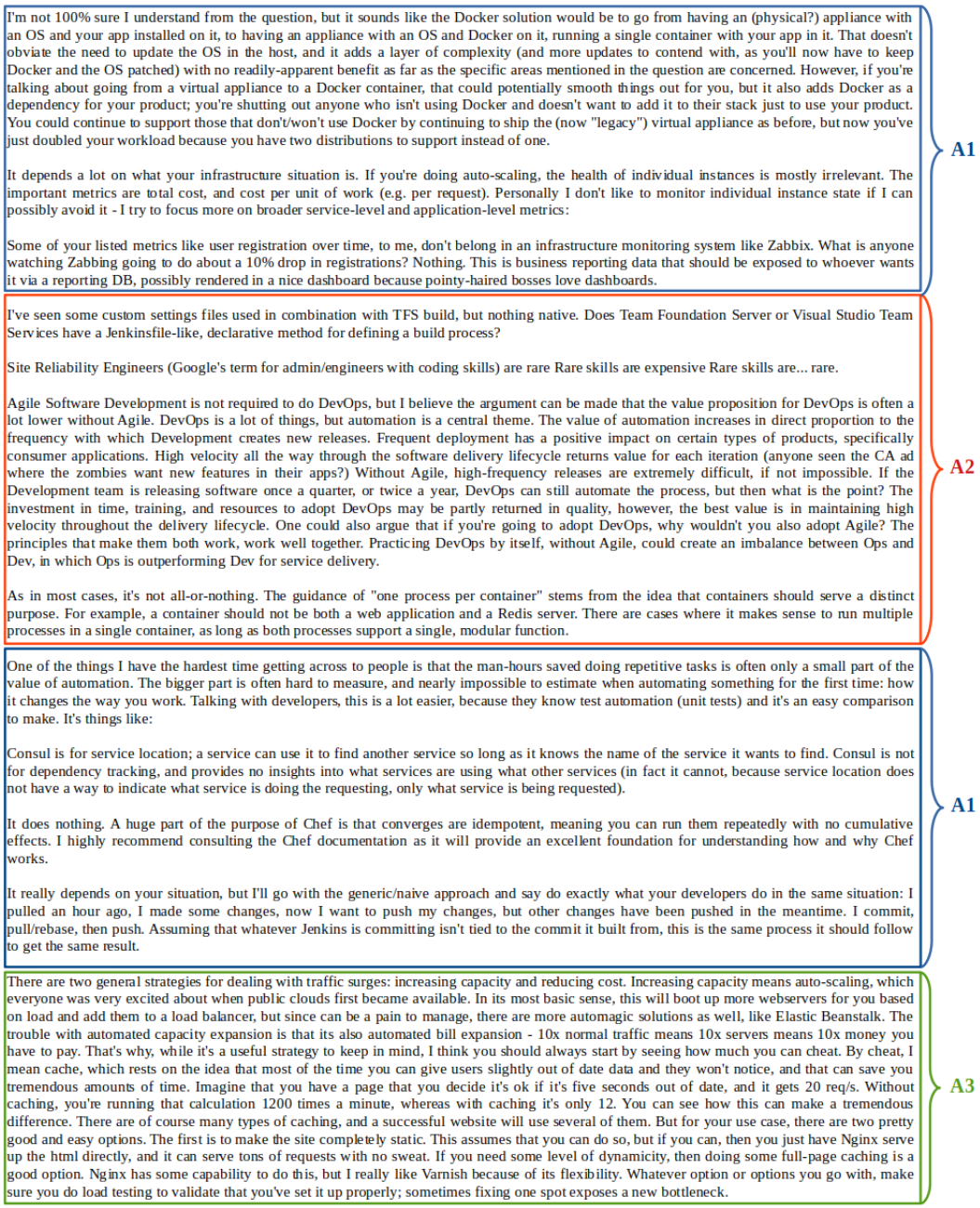
We find that simple approaches can outperform complex models such as those based on neural networks in certain cases. However, neural networks have the possibility to create some transformations that are beneficial in certain cases. It is easier to justify simple approaches, but it is difficult to explain the predictions made by neural networks due to their black-box nature.

1.3 Organization of the thesis

The rest of the thesis is organized as follows: Chapter 2 discusses the background of the SCD problem. Chapters 3, 4, 5 and 6 describe the PAN datasets along with the tasks proposed by PAN along with suitable models and evaluations. Then in Chapter 7, we analyze our approach along with limitations and potential future directions. Finally in Chapter 8, we draw our conclusions.

Figure 1.2: Example problem

Figure 1.3: Example problem with solution



2
Background

This chapter discusses a background of the SCD problem in terms of PAN challenges and approaches taken. We describe the different feature selection strategies, neural network approaches and distance measures pertaining to SCD and authorship studies.

2.1 Overview of SCD approaches at PAN CLEF

In this section, we present an overview of the approaches taken over the years from 2018 to 2021 at the PAN CLEF for the SCD problem. In Table 2.1 we present a summary of these approaches.

In PAN 2018, the task was to determine if a document was single authored or multi-authored. Most of the approaches treated the problem as a classification problem. The winning approach presented by Zlatkova in [12] used lexical features such as word ngrams and typcial beginnings and endings of words. This was followed by creating a stacked ensemble of multiple models such as SVM RBF, Random Forest, AdaBoost Trees, Multilayer Perceptron, LightGBM with a final meta classifier Logistic Regression. Another ensemble approach was presented by [13] who calculated the weighted sum of probabilities from an ensemble of three models - statistical, hashing, and Logistic Regression based counting classifier. The feature selection contained a mix of different features such as the number of sentences, text length, unique words percentage, punctuation symbols fraction, letter symbols fraction, character ngrams, word ngrams. In the approach by Khan [14], the document was segmented into sentences which are further grouped to form two or more groups. Each group was then converted to a feature vector using most frequent and least frequent word pairs, punctuations, stop words, etc. Two consecutive sentence groups are then compared with a match score.

Two neural network approaches were also introduced in PAN 2018. Hosseinia in [15], introduced a Parallel Recurrent Neural Network based approach where the similarity between parts of the document to the entire document is compared. At first, the document is split into fixed length mini-documents. For the source input, each mini-document is then converted into feature vectors using multiple features such as word ngrams, char ngrams, POS tags etc. For the target input, a combined representation is obtained by averaging the feature vectors of all the mini-documents. The source is then passed through the encoding and decoding layers. Eventually, the source is transformed into the target through an objective function.

Schaetti [16] also presented a neural network approach with Convolutional neural networks having a character embedding layer, three convolutional layers and 25 filters each.

In PAN 2019, two tasks were presented. The first task was the single or multiple author encountered in PAN 2018. The second task was to find the number of authors and based on authorship clustering. In this challenge, the winning approach was algorithmic and presented by Nath in [17]. The given document is segmented paragraph wise to form called windows. Each window is converted into feature vectors of words by most frequent terms (50 features). The mutual distances among the windows are used to create the distance matrix. Threshold Based Clustering algorithm (TBC) selects the smallest distance from the distance matrix iteratively to form clusters governed by the thresholds for adding new member nodes or merging clusters. It is assumed that the windows assigned to the same cluster have been written by the same author. The number of clusters yields the number of authors. The idea is that the Window Merge Clustering algorithm (WMC) follows the suit of TBC. However, on being added to a cluster, the feature matrix of the cluster is recalculated to obtain a combined representation of all the member nodes. A hierarchical cluster is then trimmed according to a threshold. If there is more than one cluster, then the document was considered multi-authored.

Zuo [18] presented two different approaches for each task. For the first task, a binary classification approach was taken with tf-idf features and a feedforward neural network. For finding the number of authors, the documents were divided into paragraph level segments. Various features were extracted such as POS, contracted word forms, spelling variations considering British and American spellings, function word frequencies, readability features. The similarity matrix was then generated from segments by an ensemble of K-means, hierarchical clustering, and a feed forward neural network. Eventually K-means was used to generate the number of clusters from the similarity matrix.

Then, in PAN 2020, in addition to the single or multiple author task, a second task based on paragraph-level style change location detection was introduced. In the approach by Castro-Castro [19], the document was split into paragraphs. Each paragraph was then represented using a mix of 185 features using data values, Boolean, float or n-gram vector. The features were composed of character-based, lexical, and syntactic features without the use of semantic words. More specifically the following features were extracted: a Boolean value indicating if the same word is used to finish a sentence and to begin the next sentence, average length of words, average number of sentences, proportion of nouns over adjectives, average sentence length, BOW (Bag-of-words) of conjunctions. The similarity between the features was defined by the type of data - Boolean (both entries should be equal), float (threshold based) or vector (threshold based). The similarity between paragraphs was defined using a percentage measure. Then the B0-maximal clustering was performed to group the paragraphs into clusters, assuming every cluster represents an author.

The best performance was achieved by Iyer and Vosoughi [20] who used a BERT [21] model to learn sentence level features in PAN 2020. Then from each document, the sentences were extracted and fed in to BERT (Bidirectional Encoder Representations from Transformers) for sentence level representations. The sentence level representations were then further assimilated into a paragraph level representation and fed into a Random Forest model as a classifier for both tasks.

In PAN 2021, a fuller version of the SCD problem was attempted with three sub problems. The first sub-problem was to determine if a document was single or multiple authored. The second sub-problem was to determine the paragraph-level style change locations. Thirdly, find the authorship clustering arrangement with a unique authorship assignment. Contrasted with PAN 2018, in PAN 2021, a majority of the approaches were based on neural networks.

Deibel in [22], used separate approaches for all the three tasks. For Task #1 an approach based on per-document embeddings and a multi-layer perceptron was used. For Task #2, embeddings and textual features were extracted from each paragraph and fed to a bidirectional LSTM (Long Short-Term Memory). The fastText [23] word vectors were used for embedding and textual features were based on corrected type-token ratio, mean sentence length, mean word length, function word frequency, readability. For Task

#3, a simple attribution-based approach is used. If no style change is detected between paragraphs, the new paragraph is attributed to the author of the previous paragraph. On the other hand, if a style change between paragraphs is detected, the new paragraph is compared to already attributed paragraphs to either assign it to a known author or to attribute it to a new author.

In the approach by Nath [24], an approach based on one shot learning and authorship verification is taken. At first, the document is split into paragraphs and fed pairwise to a siamese bidirectional LSTM/ GRU (Gated Recurrent Unit) network with a GloVe word embedding layer to compute similarity between them. The Task #1 results are then inferred from Task #2 output.

Singh [25] also cast the problem as an authorship verification task. From each paragraph, features based on tf-idf values of character ngrams, POS tags tri-grams, POS tag chunks, stop-word and POS tag hybrid tri-grams, Part-of-Speech tag ratios, unique spellings and special characters were selected. Further, function words frequency, average word length, word length distribution and a variety of features for vocabulary richness (ratio of hapax legomena vs dis-legomena, type-token ratio, Yule's K, and Simpson's D etc.) were also added as features. The vector difference between two paragraphs at a time was computed and fed to a Logistic Regression classifier for Task #1 and Task #2. For Task #3, hierarchical clustering on a distance matrix was created from the classifier scores to cluster the paragraphs written by the same author.

The best performance was attained by Zhang et al [26] with a pretrained BERT-Base and a fully convolutional neural network model to detect style changes between paragraphs.

Strøm [27] introduced an approach with a twofold feature extraction: at first, embedded text features were created using BERT on a sentence level which were then further aggregated at paragraph level. Secondly, a set of textual features are extracted on a paragraph level: character-based lexical features such as distinct special character, spaces, punctuation, parentheses and quotation marks, sentence, word based features such as distribution of POS-tags, token length, number of sentences, sentence length, average word length, etc. along with contracted word forms, readability features and function word frequency. For Task # 1, a document level representation is fed to stacked ensemble of multiple classifiers. Likewise, tor Task # 2, paragraph level representations are compared using a stacked ensemble of classifiers. For Task #3, if a document is classified as being multi-authored based on the output of task 1, the probability of similarity for each pair of paragraphs is computed. If the threshold is greater than 0.5, both paragraphs are attributed to the same author.

From PAN 2018 to PAN 2021, we can observe a growth in word-embedding based features and neural network based approaches primarily based on BERT. We can also observe an improvement in performance by neural network and embedding based approaches.

2.2 Feature Selection in Authorship Studies

The key hypothesis in authorship attribution is that everyone has a unique writing style which can be identified with stylometric analysis. Van Halteren describes the specific properties of authors as their stylome [28]. To solve the different authorship problems using statistical and machine learning models, the text data needs to be converted to numeric feature vectors. A variety of measures and statistics have been introduced to capture the uniqueness or idiosyncrasies of an author's style numerically.

In the survey by Stamatatos [29], stylometric features are classified as lexical, character, syntactic, semantic or application specific. Lexical features view the given text as a series of tokens. For instance, there are simple measures such as word length or sentence length. Mendehall [30] compared the frequency distribution of words of different lengths to differentiate the writing styles of Bacon and Shakespeare. However, the difference in literary presentation on account of Bacon's prose and Shakespeare's verse could account for the differences observed [31]. Univariate features such as sentence-length [32], type to token ratio [33], Yule's characteristic K [34] or Simpson's D [35] measure the vocabulary richness or

Table 2.1: Summary of approaches in PAN SCD

Paper Reference	PAN Corpus	Features	Problem Statement	Approach
Zlatkova [12]	2018	Lexical	Task 1: S/MA	Stacked Ensemble (SVM RBF, Random Forest, AdaBoost Trees, Multilayer Perceptron, LightGBM) with meta classifier Logistic Regression
Hosseinia [15]	2018	Lexical, Syntactic, Character	Task 1: S/MA	Parallel Recurrent Neural Network
Safin and Ogaltsov [13]	2018	Lexical, Character	Task 1: S/MA	Ensemble (Statistical, Hashing, Counting classifiers)
Schaetti [16]	2018	Lexical, Character	Task 1: S/MA	CNN
Khan [14]	2018	Lexical	Task 1: S/MA	Algorithmic Approach
Nath [17]	2019	Lexical	Task 1: S/MA Task 2: AC (N)	Algorithmic Approach (Clustering)
Zuo [18]	2019	Lexical, Syntactic	Task 1: S/MA Task 2: AC (N)	Task 1: FFNN Task 2: Ensemble (Kmeans, hierarchical clustering, FFNN)
Castro-Castro [19]	2020	Lexical, Syntactic	Task 1: S/MA Task 2. Paragraph SCD	Algorithmic approach (B-maximal Clustering)
Iyer [20]	2020	Embedding	Task 1: S/MA Task 2. Paragraph SCD	Random Forest
Deibel [22]	2021	Lexical, Syntactic	Task 1: S/MA Task 2. Paragraph SCD Task 3: AC (Ass.)	Bidirectional LSTM NN
Nath [24]	2021	Embedding	Task 1: S/MA Task 2. Paragraph SCD Task 3: AC (Ass.)	Siamese Bidirectional LSTM/ GRU NN

Summary of approaches in PAN SCD

Singh [25]	2021	Lexical, Syntactic, Character	Task 1: S/MA Task 2. Paragraph SCD Task 3: AC (Ass.)	Logistic Regression + Hierarchical clustering
Zhang [26]	2021	Embedding	Task 1: S/MA Task 2. Paragraph SCD Task 3: AC (Ass.)	FCNN
Strøm [27]	2021	Lexical, Syntactic, Embedding	Task 1: S/MA Task 2. Paragraph SCD Task 3: AC (Ass.)	Stacked Ensemble

Where,
S/MA = single or multiple author, Paragraph SCD = paragraph-level style change location detection,
AC = authorship clustering, N = number of authors, Ass. = authorship assignment,
RNN = recurrent NN, CNN = convolutional NN, PRNN = parallel RNN, FFNN = feed forward NN,
FCNN = fully CNN

vocabulary properties of the text. These methods were however not found to be adequately suitable [36] for authorship attribution. Yet another idea is to count the occurrence of the most frequent words of the text (Bag-of-words), as suggested by Burrows [37]. Some variations also include punctuations as features. In such an approach, function words tend to be selected as they occur more frequently than content words. Therefore, such a representation can be said to be topic independent. However, bag-of-words approaches do not take the order of the words into consideration. Word ngrams [38] however, consider context of the words.

Character features focus on character types or character ngrams and view the given text as a series of characters. The advantage of such a representation is to capture the context including punctuation around a word although it is intuitively less interpretable than the BOW approach. Character ngrams features are generated in much larger numbers than BOW due to the overlap between the selected features. However, character features are more noise tolerant. Character ngrams have been widely used in authorship attribution with some success [39, 40]. However, the capturing longer dependencies or sequential information with ngrams is cumbersome due to computational constraints.

Syntactic features are style markers that focus on syntactic preferences of the authors, such as Parts-of-Speech (POS) based approaches [41]. POS bigrams [42] or more generally POS ngrams have also been proposed and shown to be effective.

Then there are semantic features such as synonyms, hypernyms and semantic dependencies. [43] used latent semantic analysis to identify semantic similarities between words. [44] used a combination of features based on sentiment, basic emotions, POS tags, perceptual features and average sentence length for cross-language authorship attribution. They found that for languages other than English, the quality of the high-level features extracted could be poor, thereby, affecting performance.

Application-specific features can focus on particular structural characteristics such as greetings, indentation, signatures etc. Layout of a web page or document can also be a stylistic marker.

In addition to these categories, we would also like to include word-embedding approaches. Word

embeddings capture both semantic and syntactic meanings of words to produce high quality real-valued vector representations by training on a large corpus. Word embedding approaches are becoming widespread and are a powerful means of building fixed length distributed representations of words. However, learning such representations is challenging because words can have multiple related meanings (polysemy).

Almeida in [45] describes two ways to obtain such representations. Prediction-based models use local data or the word's context and have developed from neural language models. Bengio in [46] proposed learning distributed representation of words and thereby probability functions for sequences of words using such representations. Many improvements [47, 48] were suggested to reduce computational cost. In 2013, Mikolov [49, 50] proposed the popular word2vec skip-gram negative sampling (SGNS) and continuous bag-of-words (CBOW) models for learning distributed words from large amounts of unstructured text data. In 2016, Fasttext [51, 52] was introduced by using n-gram embeddings instead of skip-grams with an improved performance especially in certain languages such as German, French, and Spanish.

Another type of models known as count-based models, act globally on corpus-wide statistics such as word counts and frequencies. Latent Semantic Analysis[53] can be considered one of the early models of this approach which applied singular value decomposition to a matrix containing word counts per document. A more recent and popular model in this family is the GloVe [54] embeddings which use ratios of co-occurrences probabilities rather than actual probabilities to capture the semantic information between pairs of words in a co-occurrence matrix. Using a weighted least-squares objective and a log-linear model, the similarity between word pairs in the co-occurrence matrix is maximized.

In 2018, [55] combined bidirectional LSTM with language models to create ELMo (Embeddings from Language Models). Instead of previous approaches such as word2vec [50] or GloVe [**?**] which capture a single independent representation for each word, ELMo represents words as functions of the entire input sentence and can learn multiple contexts for a single word. ELMo word vectors are computed on top of a two-layer bidirectional language model (biLM).

The most recent developments in learning distributed representation are attention based [56] models such as BERT (Bidirectional Encoder Representations from Transformers) [21], CTRL(Conditional Transformer Language Model for Controllable Generation) [57], GPT (Generative Pre-trained Transformer) [58], GPT-2 [59] and GPT-3 [60]. As in the case of ELMo [55], the transformer-based language models can also represent multiple contexts for a word. However, they bring the advantage of parallel processing which speeds up processing and solves vanishing gradient problem.

However, one of the main limitations of transformer-based models is that, they are colossal as compared to previously introduced approaches and require proportionately high computational power and memory.

2.3 Neural Networks in Authorship Studies

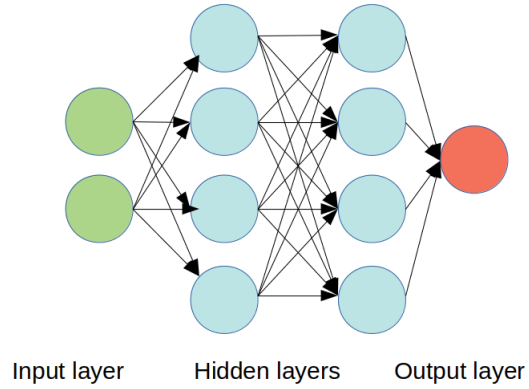
Over the last decade, we have seen remarkable improvement in various NLP areas such as machine translation, speech recognition, conversational bots. Deng [61] describes the growth of NLP after 2009 as the third wave of growth based on deep learning. The availability of large amount of data with the proliferation of the internet has enabled the use of deep learning approaches. One of the biggest advantages of deep learning approaches is reducing the dependence on human expertise needed for feature selection, thereby reducing cost. However, labelled data, which is a requirement for deep learning, can be costly and thereby a limiting factor. Another criticism of deep learning [62, 63] is the black-box nature of the learning and the need for explainability.

Nevertheless, promising developments are being made continuously on these fronts. In this section, at first we describe some of the common neural network architectures used in NLP.

2.3.1 Feed Forward Neural Networks (FFNN)

Supervised learning approaches based on FFNNs have been around a long time [64–66]. An FFNN is a simple neural network with one or more hidden layers and a nonlinear activation function (such as sigmoid, tanh). A diagram of a multi-layer perceptron is shared in Figure 2.1. FFNN have been used widely in authorship studies, however more recently, an emphasis towards more sophisticated networks is observed. Stańczyk in [67], used lexical and syntactic features for an authorship attribution task distinguishing the literary works of two polish writers Henryk Sienkiewicz and Bolesław Prus. [68] used FFNN to identify books authored by Leo Tolstoy, from the ones authored by George Orwell and Boris Pasternak. FFNN based language models have been used in authorship attribution [69].

Figure 2.1: A Multi-Layer Perceptron

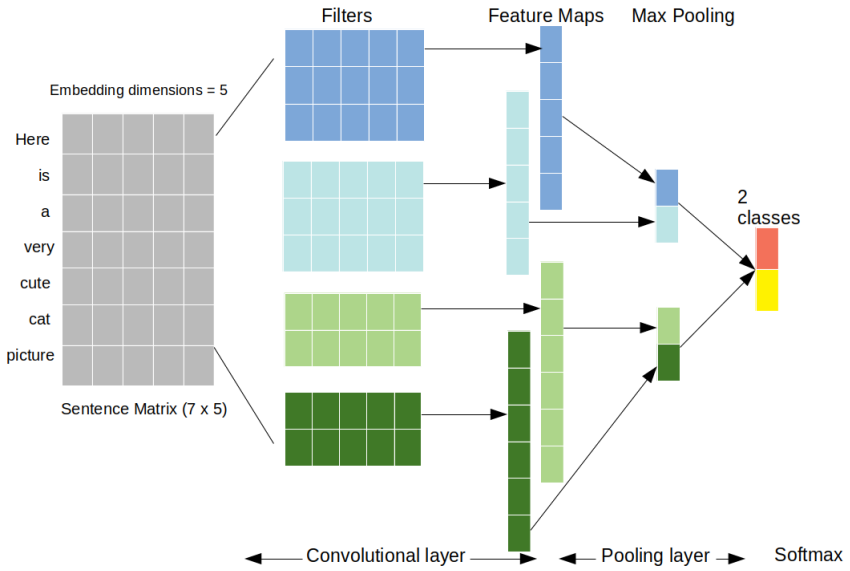


2.3.2 Convolutional Neural Network

Convolutional neural networks (CNN) [70] are a type of deep neural networks where a set of filters is slid across the input volume to extract features. At each respective input position, the dot product of the filter and the input at the specific position is calculated. This representation is known as the feature map or activation map. Then, an activation function such as sigmoid, ReLu, Leaky ReLu performs a non-linear transformation of the feature map or maps. In Figure 2.2, a diagram of a CNN is shown. In this figure, a sentence has been represented as where each token corresponds to a row (row size = 7, number of tokens) while the columns are based on the size of the embedding (here d=5). There are four convolutional filters shown, of which, two are of size 2 and two others are of size 3, which are slid across the input matrix to extract feature maps. Then 1-max pooling operation is performed over each feature map to yield the a single feature vector for the final layer. The softmax layer then assigns probabilities to each class, in this case, there are two classes.

CNNs have been shown to be very effective in computer vision problems such as video classification [71]. [72] shows how a simple word level CNN is suitable for sentence classification in multiple datasets related to sentiment analysis on movie and customer reviews, question type classification, sentence subjectivity vs objectivity and opinion polarity. Character level encoded CNNs were also shown to be suitable for text classification by [73] for news articles and review star-ratings. [74] used CNNs for authorship attribution on a short-text dataset by [75] and achieved positive results. Ruder in [76] applied character level and multi-channel CNN for large-scale authorship attribution. [77] applied CNNs for authorship identification on a dataset of scientific publications with positive results.

Figure 2.2: A Convolutional Neural Network



2.3.3 Recurrent Neural Network (RNN)

Sequential data provides the possibility to explore the dependencies among the sequences. RNNs [78] are a form of deep neural networks which have internal states for processing variable length sequence data, making them useful in speech and text processing amidst other applications. In Figure 2.3, a diagram of a simple RNN is shown where x, h and y are the input, hidden and output (optional) states respectively. While processing an input sequence, at a certain time step t, the hidden state h of the RNN is updated by

$$h_t = f(h_{t-1}, x_t) \tag{2.1}$$

where f is the activation function such as sigmoid. LSTM and GRU units, discussed later in this subsection, can also be a choice of more complex activation functions. The aim of an RNN is to predict the subsequent symbol in a sequence by training to learn the probability distribution over given sequences. The main drawback of RNNs is that gradients tend to vanish or explode for long term dependencies in RNNs making it difficult to retain information over several time steps.

Since the backpropagated error in RNNs depends on the size of the weights, over several time steps, the error tends to either vanish or explode. This is known as the vanishing or exploding gradient problem. Vanishing gradients may cause the network learning time to increase dramatically or may prevent convergence. On the other hand, exploding gradients can cause oscillating weights.

Long Short-Term Memory (LSTM) To solve the vanishing gradient problem [79] of RNNs, a gated neural network called LSTM [80] was introduced to ensure a constant error backpropagation through the network. In Figure 2.4 a diagram of a LSTM cell is shown.

The LSTM unit has a cell and three gates, viz. the input, the output and the forget gates. The input gate determines the amount of content to be added to the cell while the forget gate determines the amount of information to be forgotten. The equations for a time step during the forward pass of an LSTM are shown below. Equations 2.2, 2.3 and 2.6 represent the forget, input and output gates respectively where, the current input at time t, x_t and previous hidden state information h_{t-1} are their respective weights and

Figure 2.3: A Recurrent Neural Network

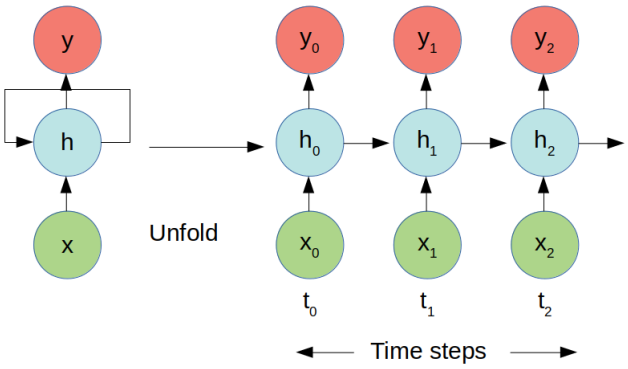
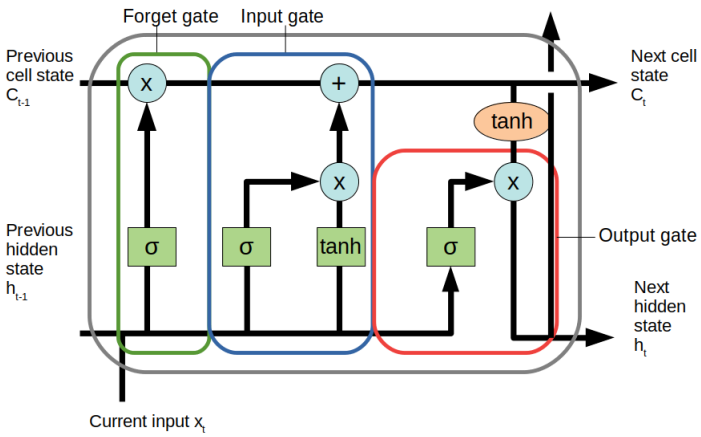


Figure 2.4: A Long Short-Term Memory unit cell



biases. In Equation 2.4, a candidate vector $\tilde{C}_t$ is then computed by taking the $tanh$ of previous hidden state h_{t-1} and current input at time t x_t with its respective weight w_C. The new cell state C_t shown in Equation 2.5, is then computed by combining the previous cell state information C_{t-1} with the input gate output and candidate vector. This in turn is further combined with the input gate to yield the next cell state value as shown in Equation 2.5. The next hidden state h_t is computed by combining the output gate value o_t with the new cell state C_t information as shown in Equation 2.7.

$$f_t = \sigma(w_f.[h_{t-1}, x_t] + b_f) \tag{2.2}$$

$$i_t = \sigma(w_i.[h_{t-1}, x_t] + b_i) \tag{2.3}$$

$$\tilde{C}_t = tanh(w_C.[h_{t-1}, x_t] + b_C) \tag{2.4}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{2.5}$$

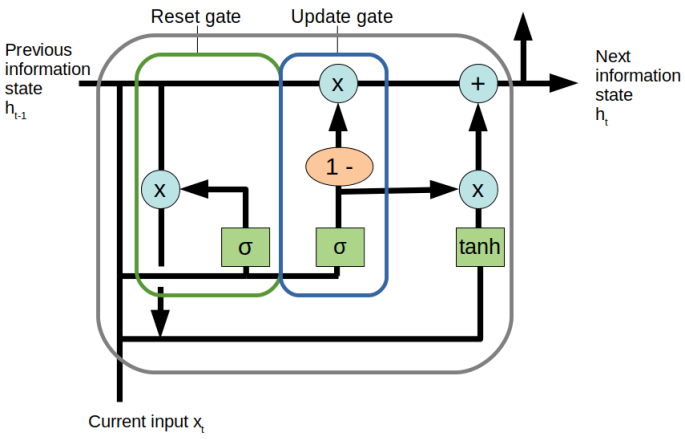
$$o_t = \sigma(w_o.[h_{t-1}, x_t] + b_o) \tag{2.6}$$

$$h_t = o_t * tanh(C_t) \tag{2.7}$$

LSTMs can be unidirectional or bidirectional. Unidirectional LSTMs preserve pre-word information but do not adequately consider the post-word information [81]. In other words, only the positive time direction is followed in such cases. Bidirectional LSTMs have been proposed to follow both pre-word and post-word information. Two LSTM layers can be stacked together in bidirectional LSTMs, such that one layer is fed an as it is input sequence, while the other layer is fed a reversed input sequence. By training the layer in both positive and negative time directions, both the pre-word and the post-word contexts can be learned.

Gated Recurrent Unit (GRU) Another popular network architecture which alleviates the vanishing gradient problem is the GRU. As compared to LSTM, in GRU, the forget and input gates are combined into a single update gate which controls the amount of information to be updated. GRU's do not have a cell state but use the hidden state to transfer information. The reset gate allows the unit to forget the previously computed state. In Figure 2.5, a diagram of a GRU cell is shown.

Figure 2.5: A Gated Recurrent Unit cell



The equations representing the forward pass for the GRU unit at a certain time step t are shown below. As shown in Equation 2.8, the update gate value is computed by taking the sigmoid of the previous state information h_{t-1} and current input x_t multiplied by their respective weights. In a similar way, the reset gate computes its output value in Equation 2.9. In Equation 2.10, the intermediate state $\tilde{h}_t$ is computed by taking the $tanh$ of the combination of reset gate output r_t, previous state information h_{t-1} and the current input x_t. Eventually, the next state information is computed by taking the sum of element-wise multiplication of the update gate u_t and h_{t-1} with the element-wise multiplication of $(1 - u_t)$ and $\tilde{h}_t$.

$$u_t = \sigma(w_u.[h_{t-1}, x_t]) \tag{2.8}$$

$$r_t = \sigma(w_r.[h_{t-1}, x_t]) \tag{2.9}$$

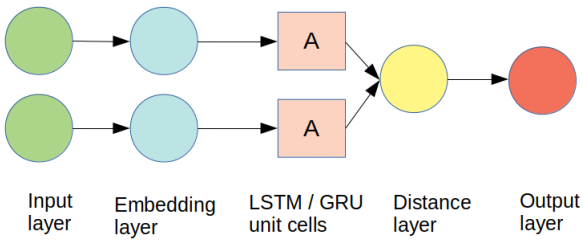
$$\tilde{h}_t = tanh(w.[r_t * h_{t-1}, x_t]) \tag{2.10}$$

$$h_t = (1 - u_t) * h_{t-1} + u_t * \tilde{h}_t \tag{2.11}$$

Whereas in LSTMs, the output gate controls the amount of memory content exposure, in case of GRUs, the entire memory content is exposed. Changing the hyperparameters such as batch-size and number of units can cause large oscillations as shown in [82]. GRUs have a simpler design as compared to LSTMs, they have fewer parameters to calculate, thereby reducing training time. However, LSTMs have higher expressive power and may perform better than GRUs when the dataset is sufficiently large. A clear winner between the two architectures was not found during empirical evaluations conducted in [83] and [84]. However, in [85], Gupta found that GRU model performed remarkably better than the LSTM at authorship attribution on the Reuters C50 [86] dataset. GRUs like the LSTMs can also be unidirectional or bidirectional.

Siamese Neural Networks (SNN) Siamese Neural Networks have been shown to compare similarity between two object representations. In the computer vision domain, SNNs have been successfully applied for image similarity [87] and face verification [88]. In the context of textual data, sentence similarity was explored by [89] and semantic similarity by [90]. In Figure 2.6, we show an image of an SNN. In addition to the input and embedding layers, we have used a block layer A to represent different RNN configurations. For instance, block A could be a single LSTM/ GRU cell or a bidirectional LSTM/GRU cell or even a combination RNN with attention mechanisms. The basic idea is that, the two input objects undergo exactly the same transformations through the two branches of the network up until the distance layer. Here, following a suitable distance metric (Manhattan, Cosine etc.), the distance between the two objects is computed and passed through a sigmoid function to yield the output.

Figure 2.6: A Siamese Neural Network



2.4 Distance Measures

In this section, we discuss some common distance measures. An appropriate distance measure needs to be chosen while determining the inter-textual distance. In Equation 2.12 we show the Manhattan distance measure based on L1-norm calculated as the sum of the absolute differences between the two vectors. This distance is equal to 0 when both the vectors are identical, otherwise positive. One drawback of this measure is the absence of a bounded maximal value.

$$Dist_{Manhattan}(A,B) = \sum_{i=1}^{m} |a_i - b_i| \tag{2.12}$$

The Tanimoto distance measure is a variation of the Manhattan distance that includes a normalization aspect.

$$Dist_{Tanimoto}(A,B) = \frac{\sum_{i=1}^{m} |a_i - b_i|}{\sum_{i=1}^{m} max(a_i, b_i)} \tag{2.13}$$

The Euclidean distance measure shown in Equation 2.14 is based on the L2-norm and measures the length of a segment between two points. The range of this distance measure is similar to the Manhattan distance measure. The lower bound is 0 when both vectors are identical and positive and unbounded otherwise.

$$Dist_{Euclidean}(A,B) = \sqrt{\sum_{i=1}^{m} (a_i - b_i)^2} \tag{2.14}$$

Another member of the L2 family is the Matusita distance measure shown in Equation 2.15 with a lower bound of 0 and an unbounded positive upper bound.

$$Dist_{Matusita}(A,B) = \sqrt{\sum_{i=1}^{m} (\sqrt{a_i} - \sqrt{b_i})^2} \tag{2.15}$$

Equation 2.16 represents the cosine similarity. The similarity measure is calculated by taking the inner product of the two vectors and normalizing with the product of the Euclidean norms of both vectors. The range of this distance measure is limited from 0 indicating no similarity to 1 indicating identical vectors.

$$Sim_{Cosine}(A,B) = \frac{\sum_{i=1}^{m} a_i.b_i}{\sqrt{\sum_{i=1}^{m} a_i^2} \cdot \sqrt{\sum_{i=1}^{m} b_i^2}} \tag{2.16}$$

The cosine distance measure can be derived from the cosine similarity. There are many variants proposed such as equation 2.17 and equation 2.18. In this thesis, we shall refer to equation 2.18, as it is the most popular variant.

$$Dist_{Cosine}(A,B) = 1 - Sim_{cosine}(A,B) \tag{2.17}$$

$$Dist_{Cosine}(A,B) = cos^{-1}(Sim_{cosine}(A,B))/\pi \tag{2.18}$$

[9] examined a range of different distance measures for authorship linking. They found that none of the selected distance functions performed optimally in all cases and that the cosine distance fared the worst.

3
PAN 2018

PAN is a series of scientific events and shared tasks on digital text forensics and stylometry. Different tasks such as authorship verification and profiling, authorship, plagiarism detection amongst others are included. From 2018 onwards, PAN has introduced a series of challenges on SCD with variations in tasks year-on-year. In the PAN 2018 challenge [91], the task was to determine if a document is written by a single or multiple authors. In other words, given a document, determine if any style changes are present. This thesis explores this question in multiple PAN datasets. From now on, this task will be referred to as the single or multiple author task. There are two primary challenges in this task. At first, no prior training set is available for the authors. Further, the documents have a wide variation of topics and are relatively short (a few hundred to a few thousand words).

The rest of the chapter is organized as follows. Section 3.1 describes the dataset. In Section 3.2, we discuss some common feature selection approaches. Then in Section 3.3, we discuss suitable classification approaches for the single or multiple author task. Finally, we discuss the results in Section 3.4.

3.1 Dataset

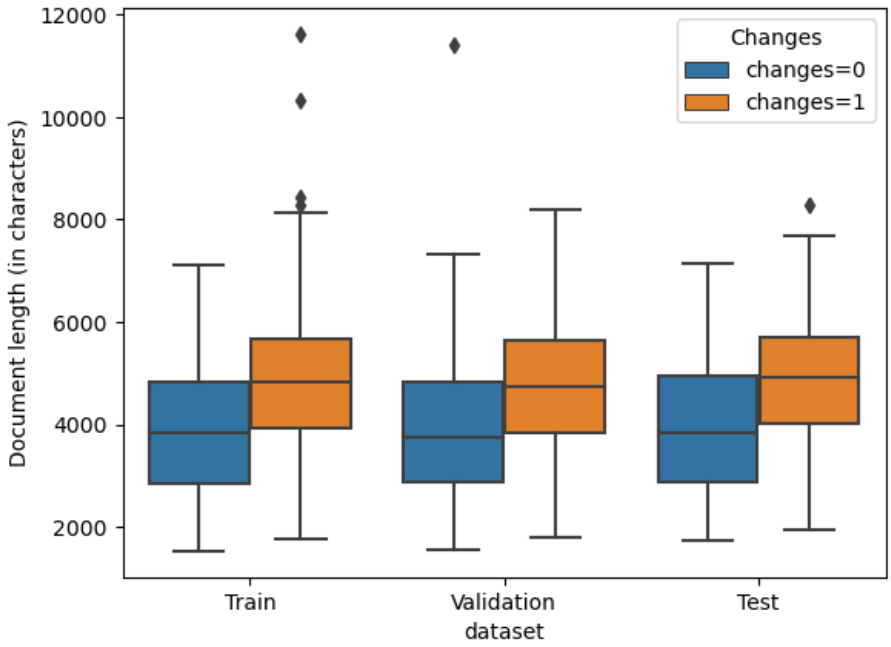
The PAN 2018 dataset was built from user posts of various sites under the StackExchange network and contain different genres such as programming, politics, sports or religion. The dataset statistics are shown in Table 3.1. There are 2980 training, 1492 validation and 1352 test documents. The proportion of the two classes, namely single author (changes=0) and multiple author (changes=1) are balanced at 50% each for all the three sets of documents. The mean number of characters for training, validation and test sets are shown in Table 3.1 along with the estimated standard deviation.

In Figure 3.1, we present a base-plot to visualize the difference of document length when a style change occurs or not. We can observe in case of the training set, that the character mean document length (4786) of documents with a style change (changes = 1) is clearly higher than the character mean document length (3871) of documents without a style change (changes = 0). The same trend reflects on the validation and test sets too.

Table 3.1: Dataset statistics for PAN 2018

Dataset Statistics	Train	Validation	Test
Size	2980	1492	1352
Percentage of single author documents	50%	50%	50%
Percentage of multi-author documents	50%	50%	50%
Mean document length (in characters)	4329	4305	4372
Std. dev. of document length (in characters)	1277	1300	1253

Figure 3.1: Document length vs style change in PAN 2018 dataset



3.2 Feature Selection

Feature selection is the process of transforming text into a numerical vector suitable for machine learning algorithms. Text feature selection strategies may focus on different aspects, such as representing the vocabulary or the arrangement or sequence of words and characters. Such features may be useful in differentiating writing styles by capturing the habitual tendency of authors to use certain words, phrases or grammatical constructs. In stylometry different types of features such as bag-of-words (BOW), Parts-of-Speech (POS), word ngrams and character ngrams have been used to identify author style.

Words may also be categorized as content words and function words. Content words contain the meaning or information, while function words or stop words provide the grammatical structure. It is possible to argue against choosing function words as features because they contain very little information. Such a similar justification may also be given against using punctuations as features. However, since we are exclusively looking at style rather than the meaning, stop words and even punctuations may be good features to represent style. In Section 3.4, we show the results of retaining stop words and punctuations.

There are many ways of vectorizing and representing the chosen types of features. For instance, the binary term frequency notes only the presence or absence of the term in the document. Term frequency

counts the number of occurrences of the term in the document. In addition to the term frequency, tf-idf also takes into account how important a term is to the document collection. Such a weighting scheme is well known in information retrieval.

Lately, word embeddings have become a popular feature representation technique. Here, the main idea is to learn a distributed feature representation of a word from the context of occurrence. Word embeddings assume that similar words occur in similar contexts and therefore the vector representations of similar words should also be similar. Some popular approaches are Word2Vec [49] and GloVe [54] embeddings.

The features in our approach composed of 500 terms each for BOW, character bigrams and character trigrams. For neural network based approaches, word embeddings such as GloVe were also used.

3.3 Classifier Choices

3.3.1 Logistic Regression

A linear regression represents a linear equation of different features with respect to a numeric target variable with the aim to estimate a probability. The Logistic Regression method is based on a sigmoid function to convert the output of the linear equation into a probability thereby enabling classification. Logistic Regression is a well-known approach in statistics with many studies highlighting its strengths and drawbacks. However, a primary limitation is the assumption of linearity between the dependent variable and the logit of the independent variables. Also, Logistic Regression requires low correlation between independent variables.

3.3.2 Naïve Bayes

Naïve Bayes predicts classes by estimating the MAP (maximum a posteriori) from the prior and the likelihood. The prior estimates the probability of the occurrence of each class in the data, whereas the likelihood estimates the probability of a feature (or evidence) present in the data for a given class. Naïve Bayes is a simple method with low computational costs. However, Naïve Bayes assumes the independence of the features, recognized to be false. The effectiveness of this method could be used as a good baseline value.

3.3.3 SVM

Support Vector Machine (SVM) models create a hyperplane such that the classes have maximum separation in an N dimensional feature space, where N is the number of features. When the classes are not linearly separable, kernel functions could be applied to transform the feature space in to a higher dimensional space. The aim of such a transformation is to find a linear separation between the classes in this newly generated hyper-space. Some commonly used SVM kernels are radial basis, polynomial function and sigmoid functions. While SVMs offer the chance to change the feature space dimensions, it may be difficult to choose the right kernel. Also, a grid search for tuning hyperparameters such as the cost of misclassification (C) as well as the values for the parameters of the kernel function, may be resource intensive.

3.3.4 Random Forest

Random Forests are an ensemble of decision trees which collectively vote towards the final prediction. As an ensemble, Random Forest is robust and reduces the tendency of decision trees towards over fitting. However, the importance of each feature is rather difficult to estimate accurately.

3.3.5 Neural Network models

We have discussed neural network approaches in Section 2.3.1. Here, we chose CNNs, LSTMs and BiLSTMs as neural network models.

3.3.6 Simple Rule Based Classifier

From Figure 3.1, we know that in this dataset, the length of documents with style changes tends to be higher than those without. We also know that the character mean document length of documents with a style change (changes = 1) is 4786. Therefore, we also add a simple rule based classifier such that, all documents with length greater than (or equal to) 4786 is simply predicted as changes = 1.

3.4 Results

Since the datasets are perfectly balanced, we have chosen accuracy as the evaluation measure. We show the results of our experiments in Table 3.2. We performed a 10 fold cross validation to check the fit of the models over the entire dataset. We observed that the models with the highest cross validation accuracy are SVM RBF (BOW) and SVM Poly degree 3 (Char Trigrams). SVM Poly degree 3 also performed the best on the test set. We also observed that our simple rule based classifier performed quite well.

Naïve Bayes is a simple model and did not perform so well in our experiments. SVM Sigmoid (Char Bigrams/ Trigrams) also did not perform well, which shows how important it is to choose the right kernel.

Although CNN models performed reasonably well, the LSTM models stayed close to the random baseline. Hyperparameter tuning is an important requirement in improving neural network performance. The small size of the training dataset is a real problem when applying deep learning models.

With a balanced dataset and a binary decision, the overall performance (test accuracy column of Table 3.2 clearly indicates that the SCD is a challenging problem. On the other hand, we must recognize that the text length is rather short (4372 characters or 728 words assuming that one word contains 6 letters).

Next, we wanted to observe if the choice of features namely, BOW, character bigrams and trigrams, presented any significant difference in terms of performance. Thus we conducted t-test among two types of features at a given time, paired by the choice of model to observe any significant difference for the test accuracy. On comparing BOW and character bigrams, the p-value was found to be 0.1733. While comparing BOW and character trigrams, the p-value was found to be 0.3579. The p-value for the paired t-test between character bigrams and character trigrams was found to be 0.1360. In all the three comparisons, we found strong evidence for the null hypothesis and therefore the differences in test accuracy were not statistically significant for these three types of features. Since the embedding based feature selection approaches were combined with neural networks, we do not perform any paired t-tests on them.

In Figure 3.2, we show the mean frequencies of 20 most frequent BOW features and for both the classes for the training dataset. Since we have included punctuations, they tend to occur in the most frequent terms along with some function words and some personal pronouns.

We can clearly observe an interesting trend. The mean frequencies of these features in documents with no style change (or changes=0) are consistently lower than those in documents with style change (or changes=1). The trend also holds true for the test set as shown in Figure 3.3. This indicates a linearly separable dataset and offers some explanation for the performance of the models. Some possible explanation behind this occurrence could lie with the procedure of dataset creation although such as trend, may not necessarily hold for other datasets.

Thus, the experimental results show that the chosen feature selection and methods are effective in achieving good performance for the single or multiple author problem.

Table 3.2: Single or Multiple Author Task Results for PAN 2018 dataset

Model	**Mean Cross Validated Accuracy**	**Test Accuracy**
Logistic Regression (BOW)	0.64	0.63
Naïve Bayes (BOW)	0.55	0.53
SVM Linear (BOW)	0.64	0.63
SVM RBF (BOW)	**0.69**	0.67
SVM Poly. degree 3 (BOW)	0.68	0.67
SVM Sigmoid (BOW)	0.60	0.59
Random Forest (BOW)	0.68	0.66
Logistic Regression (Char. Bigrams)	0.63	0.60
Naïve Bayes (Char. Bigrams)	0.54	0.50
SVM Linear (Char. Bigrams)	0.64	0.60
SVM RBF (Char. Bigrams)	0.68	0.66
SVM Poly. degree 3 (Char. Bigrams)	0.67	0.66
SVM Sigmoid (Char. Bigrams)	0.37	0.37
Random Forest (Char. Bigrams)	0.66	0.67
Logistic Regression (Char. Trigrams)	0.65	0.64
Naïve Bayes (Char. Trigrams)	0.54	0.53
SVM Linear (Char. Trigrams)	0.64	0.62
SVM RBF (Char. Trigrams)	0.68	0.66
SVM Poly. degree 3 (Char. Trigrams)	**0.69**	**0.68**
SVM Sigmoid (Char. Trigrams)	0.37	0.36
Random Forest (Char. Trigrams)	0.65	0.66
CNN (Integer encoding)	0.63	0.63
CNN (GloVe embedding)	0.61	0.58
LSTM (Integer encoding)	0.51	0.51
BiLSTM (GloVe embedding)	0.53	0.51
Simple Rule Based	-	0.63

Figure 3.2: Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2018

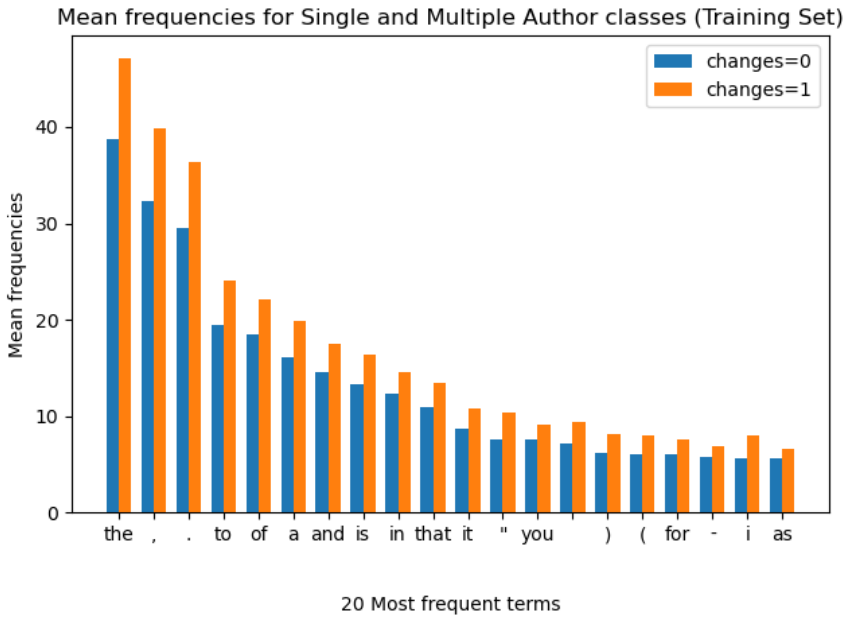
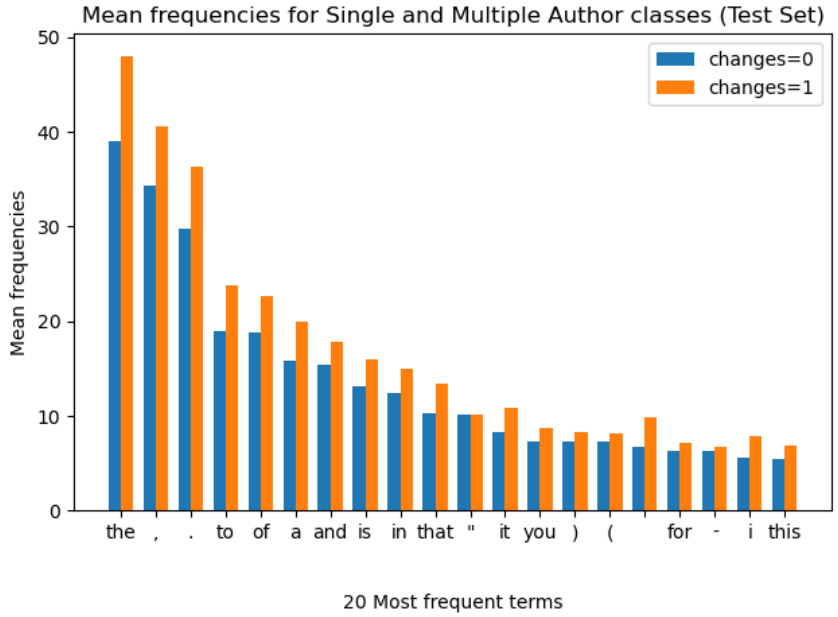


Figure 3.3: Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2018



4
PAN 2019

The PAN 2019 SCD challenge [92] proposed two tasks. The first task was to identify if any style changes were present in a given document. This is equivalent to the single or multiple author task, encountered in PAN 2018.

The second task sought the number of authors. Finding the number of authors is similar to authorship clustering on a document level. In complete authorship clustering, all the texts written by the same author should be present in the same cluster, and one text should be assigned to only one author.

4.1 Dataset

In this section, we perform some preliminary statistics on the data to gain a comprehensive understanding of the data. The given dataset was composed of forum posts from different StackExchange network sites. All documents were written in English. The topics of discussion varied greatly. As the documents were based on forums, each document was independent of others. Moreover, the identification of each possible author is not provided across the entire dataset. However. we can identify the authors inside a document. A basic statistical overview of the PAN dataset is provided in Table 4.1.

The training, validation and test datasets consist of 2546, 1272 and 1210 documents respectively. The number of authors ranges from 1 to 5. Each document is composed of responses to a single thread at a forum.

Table 4.1: Dataset statistics for PAN 2019

Dataset Statistics	**Train**	**Validation**	**Test**
Size	2546	1272	1210
Percentage of single author documents	50%	50%	50%
Percentage of multi-author documents	50%	50%	50%
Mean Document Length (in characters)	7538	7370	7476
Std. Dev. of Document Length (in characters)	3729	3719	3758
Number of authors	1-5	1-5	1-5

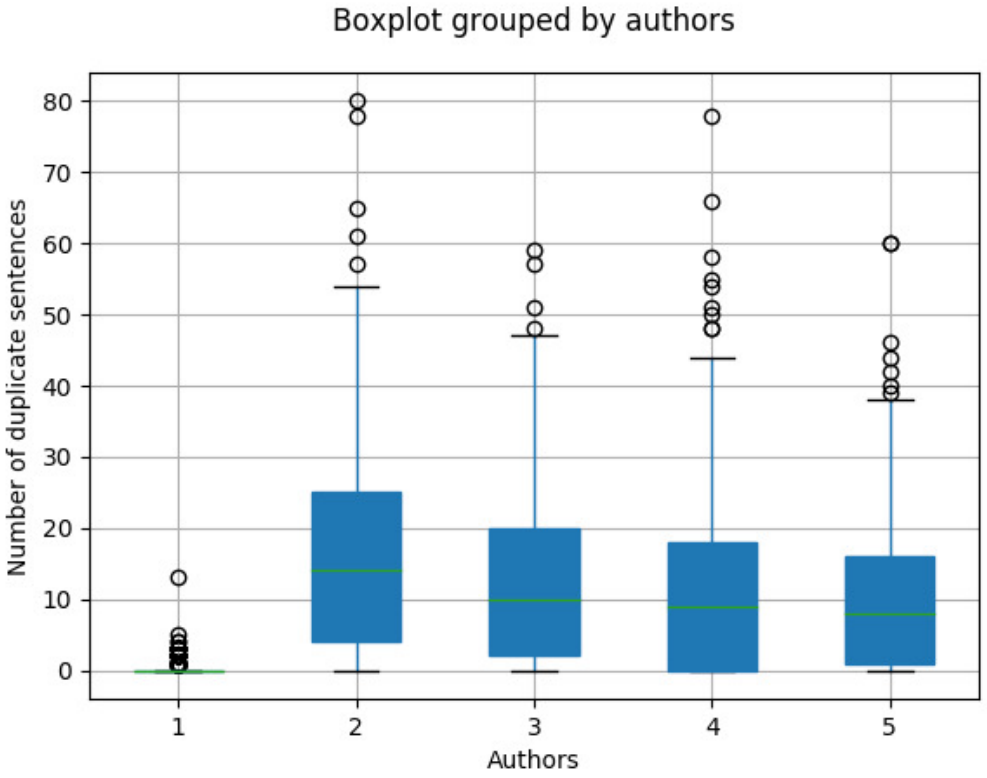


Figure 4.1: Number of duplicates grouped by authors in the training set

Table 4.2: Duplicates in documents (Training set)

	Documents	
Number of Authors	without duplicated sentences	Total
1	1142 (89.7%)	1273
2	48 (14.8%)	325
3	65 (20.8%)	313
4	83 (25.3%)	328
5	66 (21.5%)	307

4.1.1 Duplicate Sentences

It was observed that certain documents contain duplicate sentences. In Figure 4.1, we show a boxplot of the number of duplicated sentences per document, grouped by the number of authors for the training set. It can be noted that for documents with a single author, the number of duplicates is rather low.

On the other hand, the documents written by multiple authors have varying number of duplicate sentences. In Table 4.2, we compare the number of documents containing no duplicate sentences with the number of authors for the training set. We observe that the number of duplicate sentences is low for single authors and relatively high for multiple authors. The same trend was observed in validation and test datasets too.

4.2 Single or Multiple Author task

The first task in PAN 2019 is the same as PAN 2018 with a binary classification. Therefore, we have used the same feature selection strategies as mentioned in Section 3.2. We also used the classifier models introduced in 3.3 except for a change in the rule based classifier.

4.2.1 Simple Rule Based Classifier

We defined a simple rule based classifier based on our knowledge of duplicates shown in Table 4.2. Since, the single authored documents do not tend to have duplicate sentences, we can specify a simple prediction based on duplicated sentences. If a document contains at least one duplicated sentence, then the model predicts the document as being multi-authored, otherwise, as single authored.

4.2.2 Results

The results of our experiments are shown in Table 4.3. We observed that the CNN (Integer Encoding) achieved the best mean CV accuracy of 0.86. However, the simple rule based classifier achieved the best test accuracy of 0.88. This is an indication that even a simple learning rule could be effective while exploiting a feature present in the dataset. We must however recognize that the presence of a duplicate sentence does not correspond to a feature that could normally occur in the real world.

As in the case of PAN 2018, to observe the effect of choice of features namely, BOW, character bigrams and trigrams on the performance, we conducted t-test among two types of features at a given time, paired by model type for test accuracy. The p-value for the paired t-test between test accuracies of BOW and character bigrams was found to be 0.2220. While comparing BOW and character trigrams, the p-value was found to be 0.2422. The p-value for the paired t-test between character bigrams and character trigrams was found to be 0.5957. Once again, strong evidence was found for the null hypothesis indicating that the differences in test accuracy were not statistically significant among these three types of features.

We also note that the overall accuracy rate achieved in PAN 2019 is higher than that achieved in PAN 2018.

In Figure 4.2, we plot the mean frequencies of 20 most frequent BOW features and for both the classes for the training dataset. Figure 4.3 plots the trend for the test dataset. The top 20 most frequent features are quite similar to those selected in the PAN 2018 dataset. We observe the interesting trend again. Here too, the mean frequencies of these features in documents with no style change (or changes=0) are consistently lower than those in documents with style change (or changes=1). In fact, the difference between the classes seems to be larger than in the case of the PAN 2018 dataset. This could a possible explanation of why the performance of the models in the PAN 2019 is better than in PAN 2018 dataset. We also observe that the mean frequencies tend of the PAN 2019 dataset tend to be greater than the PAN 2018 dataset. This is possibly because the mean document length of PAN 2019 (approx. 7500 characters) is higher than that of PAN 2018 (approx. 4329 characters).

4.3 Authorship Clustering (Number of Authors) task

In the second task, the number of authors of a given document must be determined. To solve this question, our approach is to segment the given document into chunks of text, called windows. Each window is converted into feature vectors of words (around 50 features). The mutual distances among the window feature vectors are measured by a suitable distance measure and fed to two clustering algorithms called Threshold Based (TBC) and Window Merge (WMC). The number of clusters yielded corresponds to the number of authors. The results of the two algorithms are then compared against each other.

Table 4.3: Single or Multiple Author Task Results for PAN 2019 dataset

Model	**Mean Cross Validated Accuracy**	**Test Accuracy**
Logistic Regression (BOW)	0.78	0.78
Naïve Bayes (BOW)	0.65	0.66
SVM Linear (BOW)	0.77	0.77
SVM RBF (BOW)	0.82	0.83
SVM Poly degree 3 (BOW)	0.80	0.82
SVM Sigmoid (BOW)	0.59	0.59
Random Forest (BOW)	0.81	0.81
Logistic Regression (Char Bigrams)	0.77	0.78
Naïve Bayes (Char Bigrams)	0.59	0.59
SVM Linear (Char Bigrams)	0.76	0.78
SVM RBF (Char Bigrams)	0.81	0.82
SVM Poly degree 3 (Char Bigrams)	0.79	0.81
SVM Sigmoid (Char Bigrams)	0.38	0.36
Random Forest (Char Bigrams)	0.80	0.81
Logistic Regression (Char Trigrams)	0.77	0.78
Naïve Bayes (Char Trigrams)	0.64	0.63
SVM Linear (Char Trigrams)	0.76	0.77
SVM RBF (Char Trigrams)	0.81	0.82
SVM Poly degree 3 (Char Trigrams)	0.79	0.81
SVM Sigmoid (Char Trigrams)	0.33	0.31
Random Forest (Char Trigrams)	0.78	0.79
CNN (Integer encoding)	**0.86**	0.86
CNN (GloVe embedding)	0.71	0.73
LSTM (Integer encoding)	0.5	0.5
BiLSTM (GloVe embedding)	0.54	0.55
Simple Rule Based	-	**0.88**

Figure 4.2: Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2019

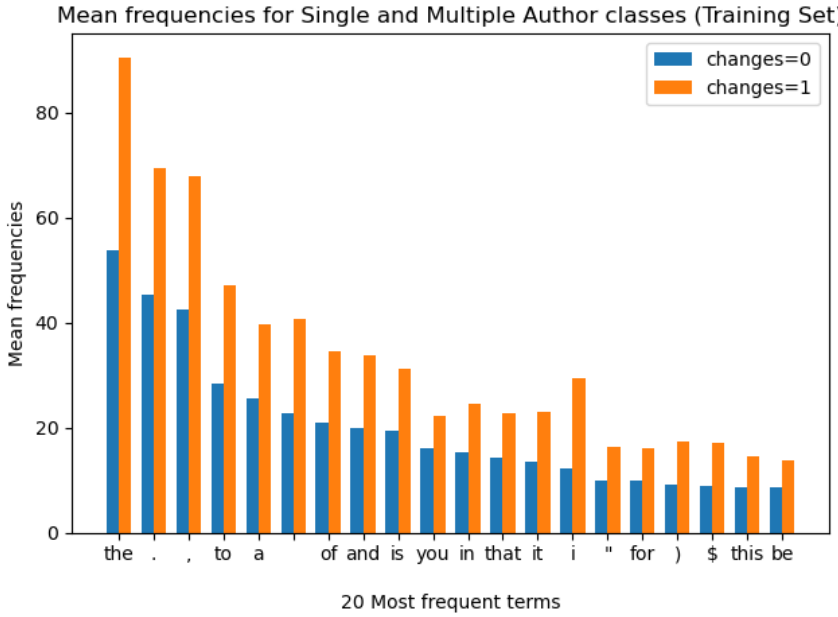


Table 4.4: Example of a Distance Matrix

	W_0	W_1	W_2	W_3
W_0	0	0.8	0.2	0.55
W_1	0.8	0	0.56	0.42
W_2	0.2	0.56	0	0.77
W_3	0.55	0.42	0.77	0

4.3.1 Feature Selection

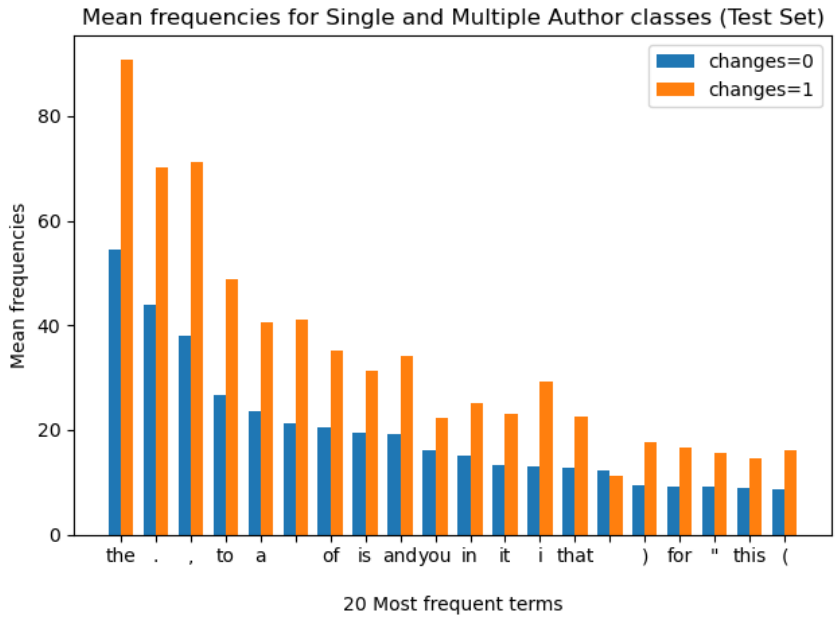
To preserve the logical document structure, we wanted to break down the document into paragraphs. However, we note that the style changes may occur inside the paragraphs in the PAN 2019 dataset. Besides, the size of the paragraphs is highly variable. Thus, we needed to define the size of a snippet of text, also known as a window of text. A window was initially assumed to be equal to a paragraph of text. However, as discussed, such a simple approach created highly variable lengths of windows. Thus, the window tokenizer was modified to merge extremely short paragraphs (less than 200 chars) with the previous paragraphs. Similarly, large paragraphs were split into smaller windows to get reasonably length-wise balanced windows.

Thereafter, the top 50 Most Frequent Tokens (MFT) were selected from each document (and not the whole corpus). Each window was converted to a feature vector based on the relative normalized frequency of the selected tokens.

For the primary experiments, we have used the Manhattan distance measure. However, we compare the changes observed with other distance measures such as Euclidean, Matusita, Cosine and Tanimoto. An example of the representation of a document in a distance matrix is shown in Table 4.4. Let us assume that the given document has 4 windows (W0-W3), then a symmetric 4x4 matrix will be created representing the distance between the windows.

Figure 4.5 represents the windows as the nodes and the distances among them as the edges of a graph.

Figure 4.3: Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2019



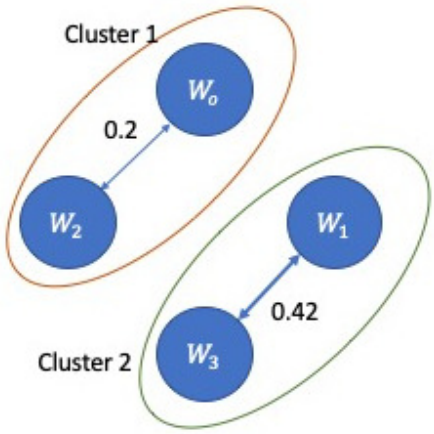


Figure 4.4: Clusters representing distance between windows

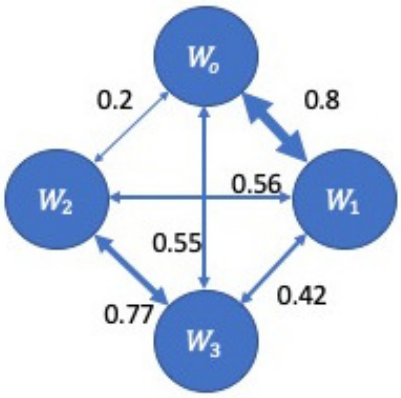


Figure 4.5: Distance representation between windows

Figure 4.4 shows the corresponding clusters created from the distance matrix. The two clusters are created based on the smallest distances between windows.

4.3.2 Definitions

Before we describe our models, we introduce a few definitions used in our methods.

- **Distance Matrix**: A distance matrix represents the distances among all the windows in a square

matrix. The diagonal elements are zero because they represent the distances between a window and itself. Table 4.4 shows a distance matrix representing the relationships in Figure 4.5. For the purpose of this experiment, we will only deal with symmetric distance matrices.

- **Sorted Distance Array**: Such an array contains non-duplicated entries of the distance matrix, sorted in ascending order. In terms of a graph, an entry in the distance matrix corresponds to the edge distance d between nodes W_i and W_j. The top right half of the distance matrix (cells of Table 4.4 highlighted in gray) are extracted and converted to an array of the form (W_i, W_j, d) where W_i, W_j are windows and d is the distance between them. An example of sorted distance array is shown in Table 4.5.

- **Cluster**: The most similar windows of text are grouped together into clusters. The average distance of a cluster is the sum total of all the distances between all members divided by the number of edges. For example, in Figure 4.4, the average distance of cluster 1 is 0.2/1 or 0.2.

- **Cluster List**: Cluster list contains all the clusters. Ideally, each cluster corresponds to an author, implying that the author has written the text in the windows. Therefore, the number of clusters corresponds with the number of authors. However, in some cases, this is not true. For example, it may so happen that an author has written only a few sentences which are contained by a single window. Such a window, if sufficiently distant from other clusters, is likely to be excluded by all clusters. As a result, there is a possibility that the number of authors calculated is lower than the ground truth. On the other hand, if the document is broken down into tiny windows, it is possible that there are too few characters, i.e., not sufficient information. As a result, the distance measurement among such windows will be higher, resulting in the formation of a higher number of small clusters. Thus the predicted number of authors is higher than the ground truth.

- **Thresholds** : The following thresholds are required by the threshold based clustering algorithm (TBC) to determine if a new member may be added to an existing cluster or if two clusters may be merged together. Thresholds ensure that existing clusters expand reasonably.

  - **Add Window Threshold**: A new member may be added to a cluster, if upon adding the new member, the resulting average distance of the modified cluster is not greater than x% of the average distance of the existing cluster.
  - **Merge Cluster Threshold**: Two clusters may be merged together, if the resulting average distance of the new cluster is not greater than x% of the average distance of either of the existing clusters.

  However, fixing a certain threshold as discussed above does not take into account the size of the existing cluster. It may be desirable that, clusters with only a few members or having a small average distance, expand more rapidly as compared to larger clusters. Therefore, the size of the cluster and its average distance were used to penalize the respective threshold values of the clusters. Such an adaptation, helped to customize the threshold according to the state of the cluster.

4.3.3 Models

In this section we describe the algorithms used by our models.

Algorithm 1 Threshold Based Clustering Algorithm

procedure CREATECLUSTER($dist\_arr$) # Convert sorted distance array to clusters
  $cluster\_list \leftarrow \phi$
  for all $entry\ e \in dist\_arr$ **do** # where e is of the form ($W_i,\ W_j,\ d$)
    $W_i,\ W_j \leftarrow$ GETWINDOWS(e)
    if $W_i \notin c$ **and** $W_j \notin c' \forall\ c, c' \in cluster\_list$ **then** # where c, c' are clusters
      $c \leftarrow \{W_i, W_j\}$
      $cluster\_list \leftarrow cluster\_list \cup c$
    end if
    if $\exists\ c' : c' \in cluster\_list$ **and** $W_i \in c'$ **and** $W_j \notin c\ \forall c \in cluster\_list$ **then**
      $c'' \leftarrow c' \cup \{W_j\}$
      if $average\_distance(c'') < add\_node\_threshold$ **then**
        $c' \leftarrow c''$ # cluster c' is updated by adding a new Window W_j
      end if
    end if
    if $\exists\ c' : c' \in cluster\_list$ **and** $W_j \in c'$ **and** $W_i \notin c$ **and** $\forall c \in cluster\_list$ **then**
      $c'' \leftarrow c' \cup \{W_i\}$
      if $average\_distance(c'') < add\_window\_threshold$ **then**
        $c' \leftarrow c''$ # cluster c' is updated by adding a new Window W_i
      end if
    end if
    if $\exists\ c, c' : c, c' \in cluster\_list$ **and** $W_i \in c$ **and** $W_j \in c'$ **then**
      $c'' \leftarrow c \cup c'$
      if $average\_distance(c'') < merge\_cluster\_threshold$ **then**
        $cluster\_list \leftarrow cluster\_list \cup \{c''\} - \{c, c'\}$
        # Replace previous smaller clusters with a new merged cluster
      end if
    end if
  end for
  return $len(cluster\_list)$ # Number of clusters represent number of authors
end procedure

Table 4.5: Sorted Distance Array

W_i	W_j	Distance
W_0	W_2	0.2
W_1	W_3	0.42
W_0	W_3	0.55
W_1	W_2	0.56
W_2	W_3	0.77
W_0	W_1	0.8

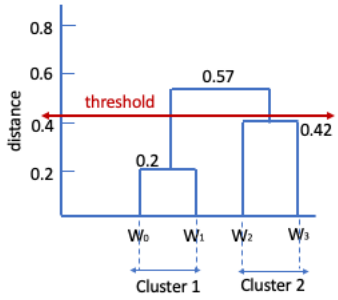


Figure 4.6: Hierarchical clusters created by WMC

4.3.3.1 Threshold Based Clustering Algorithm

The intuition behind the TBC algorithm is that, a good way to start clustering is to select the closest windows in terms of distance, because, such windows are likely to belong to the same cluster and be written by the same author. As such, the distance matrix is transformed into an array sorted by distance, viz. the sorted distance array or $dist\_arr$ and fed to the Algorithm 1. The idea is to select the windows with the smallest distances iteratively such that, when forming a cluster, the closest members are included first. Thereafter, the members that are further away, are included. Hence, the clusters would be expected to grow from small and dense to large and spread out. Such a growth is controlled by the thresholds shown in Section 4.3.2. For each entry in the $dist\_arr$, the windows may have any one of the following conditions:

- None of the windows are present in any existing cluster of the cluster list. Therefore, a new cluster must be created with these two windows and added to the cluster list.
- One of the windows is already a member of a certain existing cluster, while the other isn't. Thus, the non-member window may be added to an existing cluster, provided that the Add Window Threshold condition is met. While adding a new member, all the corresponding edges between the new member and existing members are extracted from the $dist\_arr$ and added to the cluster.
- Both windows belong to two different existing clusters which may be merged if the conditions of the Merge Cluster Threshold are satisfied. As in the previous case, all the corresponding edges are added to the cluster

It is to be noted, that the condition that both windows belong to the same cluster is not possible, as otherwise, such an entry would already have been traversed while adding new members or cluster merging. Further, the algorithm always computes the distance between windows and not directly between the clusters.

4.3.3.2 Window Merge Clustering Algorithm

The Window Merge Clustering algorithm (WMC) iteratively combines the most similar windows to generate a new set of windows. In this initial step, each window forms a cluster. Then we need to determine the most similar pair of clusters. In a third step, the distance matrix between clusters is re-calculated for the next iteration. As a result, the clusters formed are hierarchical. The idea is to represent each cluster with a combined representation of all of its members together, rather than individual distances.

Figure 4.6 shows the formation of the clusters by the Window Merge Clustering. The height of the cluster corresponds to the distance between the two clusters. In the first iteration, cluster $C^1{}_{(W_0,W_1)}$ is formed. The texts of windows and W_0 and W_1 are concatenated to create a new window W_0W_1 which also represents the corresponding new cluster. Thus, the resulting new set of windows is W_0W_1, W_2 and W_3. In the next iteration, cluster $C^2{}_{(W_2,W_3)}$ is formed because the distance between W_2 and W_3 is the lowest in the distance matrix. This gives rise to the concatenated new window W_2W_3. Thus, at this point, the updated windows are W_0W_1 and W_2W_3. Eventually, these windows are selected in the third iteration as they are the only remaining entry in the distance matrix, resulting in the cluster $C^3{}_{(W_0W_1,\, W_2W_3\,)}$.

To determine the number of clusters, we use a truncation threshold, which is a certain percentage of the height of the largest cluster. In this example shown in Figure 4.6, the truncation threshold is set to 0.43 or 75% of 0.57 which is the height of $C^3{}_{(W_0W_1,\, W_2W_3\,)}$. With the help of the truncation threshold, we can control the size of the clusters. Also, as in the case of the TBC, the number of clusters relates to the number of authors. In our experiments, we found that the truncation threshold of 0.3 performed the best.

Algorithm 2 Window Merge Algorithm

```
1:  procedure CreateHierarchicalCluster(dist_matrix, text_windows)
2:      window_merge_order ← φ
3:      while size(dist_matrix) > 1 do
4:          W_i, W_j ← FindMinDistanceWindows(dist_matrix)
5:          merge_order ← window_merge_order ∪ {W_i, W_j}
6:          new_text_windows ← MergeWindows(W_i, W_j, text_windows)
7:          dist_matrix ← CalculateDistMatrix(new_text_windows)
8:      end while
9:      authors ← Truncate(window_merge_order, truncation_threshold)
10:     return authors
11: end procedure
```

4.3.4 Evaluation Measure

We follow the evaluation strategy considered by PAN [92]. At first, we consider accuracy rate as one of the evaluation measures. However, the target variable - number of authors is ordered. For example, given a ground truth of 2 authors, a prediction of 3 authors is closer to the ground truth than, say, a prediction of 5 authors. Accuracy rate being a simple metric, cannot capture the order or ranking of the classes.

The Ordinal Classification Index (OCI) [93] is an error measure introduced by [93] for ordinal data classification. OCI captures how much the result diverges from the ideal prediction and the inconsistency of the classifier with respect to the relative order of the classes. A confusion matrix is given with the ground truth class in rows and the predicted class in columns, following the order of the classes. In OCI, the ideal path lies along the main diagonal. Deviations from the ideal path and inconsistencies are then suitably penalized. Since the OCI is an error measure, a lower value of OCI is preferred. The OCI is defined in Equation 4.1.

Table 4.6: Number of authors task for PAN 2019 dataset - Initial Results (Manhattan distance)

	Accuracy			**OCI**			**Rank**		
	Train	Validation	Test	Train	Validation	Test	Train	Validation	Test
TBC	0.33	0.32	0.33	**0.73**	**0.73**	**0.73**	0.30	0.29	0.30
WMC	**0.40**	**0.41**	**0.40**	0.75	**0.73**	0.76	**0.33**	**0.34**	**0.32**

Table 4.7: Number of authors task for PAN 2019 dataset - Improved Results using Duplicated Sentences (Manhattan distance)

	Accuracy			**OCI**			**Rank**		
	Train	Validation	Test	Train	Validation	Test	Train	Validation	Test
TBC	**0.57**	**0.57**	**0.59**	**0.87**	**0.86**	**0.86**	**0.35**	**0.36**	**0.36**
WMC	0.55	0.57	0.56	0.93	0.90	0.93	0.31	0.33	0.32

$$OC^{\gamma}_{\beta_1;\beta_2} = min\Bigg\{1 - \frac{\overbrace{\sum_{(r,c)\epsilon pathn_{r,c}}}^{\text{path along diagonal}}}{\underbrace{N + (\sum_{\forall(r,c)} n_{r,c}\big|r-c\big|^{\gamma})^{\frac{1}{\gamma}}}_{\text{M}}} + \underbrace{\beta_1 \sum_{(r,c)\epsilon path} n_{r,c}\big|r-c\big|^{\gamma}}_{\text{path deviation penalty}} + \underbrace{\beta_2 N_{disc\_pos}}_{\text{path inconsistency penalty}}\Bigg\} \tag{4.1}$$

Where,
N is size of the dataset.
γ is an integer and $\gamma = 1$ in PAN evaluation.
$n_{r,c}$ the number of points from the r^{th} class predicted as being from the c^{th} class in the confusion matrix.
B_1 is the coefficient of penalization for deviation of path from diagonal.
B_2 is the coefficient of penalization for inconsistency of path.
M is the Minkowski distance representing the measure of the dispersion of the data in the confusion matrix.
N_{disc_pos} is the penalization term for the number of discordant pairs of vertices in the path measuring the inconsistency of the path.

Since a higher value of accuracy and lower value of OCI is preferred, the two measures are combined into a single rank measure, with equal weight on accuracy and OCI.

$$rank = \frac{accuracy + (1 - OCI)}{2} \tag{4.2}$$

4.3.5 Results

The Add Window and Merge Cluster thresholds of the TBC algorithm were varied from 5% to 100%. We observed the optimum performance at 50% for both the thresholds. In our experiments, both thresholds are set to 50%. We show the evaluation of our algorithms in Table 4.6 using the Manhattan distance measure 2.12. In our evaluation, we find that WMC performs better than the TBC for accuracy but lags behind slightly for OCI. It is to be noted that since OCI is an error measure, the lower the better. With our ranking measure, WMC leads over TBC.

Table 4.8: Comparison of distance measures for TBC algorithm

		Manhattan	**Matusita**	**Euclidean**	**Tanimoto**	**Cosine**
Accuracy	Train	0.33	**0.34**	0.32	0.33	0.33
	Validation	0.32	0.33	0.33	0.34	**0.35**
	Test	0.33	0.32	0.33	0.33	**0.34**
OCI	Train	**0.73**	**0.73**	**0.73**	**0.73**	**0.73**
	Validation	0.73	0.72	**0.71**	0.73	0.73
	Test	**0.73**	0.74	**0.73**	**0.73**	**0.73**
Rank	Train	0.30	**0.31**	0.30	0.30	0.30
	Validation	0.29	**0.31**	0.31	0.30	**0.31**
	Test	**0.30**	0.29	**0.30**	**0.30**	**0.30**

Table 4.9: Comparison of distance measures for WMC algorithm

		Manhattan	**Matusita**	**Euclidean**	**Tanimoto**	**Cosine**
Accuracy	Train	**0.40**	0.37	0.39	0.35	0.34
	Validation	**0.41**	0.39	0.40	0.40	0.35
	Test	**0.40**	0.39	0.39	0.35	0.34
OCI	Train	0.75	**0.74**	**0.74**	0.75	0.75
	Validation	0.73	**0.72**	0.73	0.73	0.73
	Test	0.76	**0.74**	**0.74**	**0.74**	0.75
Rank	Train	**0.33**	0.32	0.32	0.30	0.29
	Validation	**0.34**	0.33	**0.34**	**0.34**	0.31
	Test	**0.32**	**0.32**	**0.32**	0.31	0.30

In Section 4.1.1, we proposed the idea of using the number of duplicates sentences to predict the number of authors. We showed how low number of duplicate sentences were indicative of a single author. We modified the algorithms TBC and WMC such that, if no duplicate sentences were present in a document, only a single author was predicted. However, if one or more duplicate sentences were observed, the algorithm was executed normally. In Table 4.7, the impact of using such a feature is shown. We observe that while the accuracy measure improved greatly, but the OCI worsened. Overall, the ranking improved for TBC but not for WMC. Here, TBC has performed better than WBC. However, we must note that half of the dataset contains only single author documents (1273/2546 = 50%). The percentage of 2-author, 3-author, 4-author and 5-author documents are 12.7%(325/2546), 12.3 % (313/2546), 12.9% (328/2546), 12.1% (307/2546) respectively. Thus, the dataset is not balanced for the authorship clustering task. By finding the correlation between duplicated sentences and single author documents, the accuracy was given a boost. However, the use of duplicated sentences is specific to this dataset and is not feasible in the real world.

As such, we use the original method for testing the effect of using different distance measures namely Manhattan, Matusita, Euclidean, Tanimoto and Cosine as discussed in Section 2.4. We observe that overall WMC performs better than TBC in terms of rank for most of the measures. Cosine seems to do better marginally for the TBC algorithm but perform the worst for the WMC algorithm. There is no clear winner among the distance measures observed.

In PAN 2019, we participated and presented the TBC/WMC approach and achieved the best performance [92].

5
PAN 2021 and others

In this thesis, we present the chapter on PAN 2021 before PAN 2020, as PAN 2021 has a balanced dataset with respect to all the three tasks. In Chapter 6, we discuss the imbalance present in the PAN 2020 datasets.

In PAN 2021, the first task is to find out if a document is authored by one or multiple authors, also known as the single or multiple author task and is a binary classification task. The second task focused on locating the style changes in a document, under the assumption that the style changes may occur only between paragraphs but never within paragraphs. Subsequently, we shall refer to this task as the paragraph-level style change location detection task. Thirdly, by using the knowledge of the style change locations, assign the texts uniquely to their respective authors. We shall refer to this task as the authorship clustering task.

For the single or multiple author task, we continue to use the approaches discussed in Chapter 3.

In Section 6.3, we describe how style change location detection can be interpreted as a one shot learning task, which is a type of classification where a single example of each class is given. An example of one shot learning can be found in the computer vision domain where the aim is to train classifiers for object identification, where only one image per class is available.

For authorship clustering, we present some approaches in Section 5.4 based on the TBC method introduced in Chapter 4 and other agglomerative approaches.

5.1 Dataset

The PAN 2021 SCD training dataset was generated from StackExchange questions and answers combined to form documents. The training set contained 11,200 documents while the validation set and the test set contained 2,400 documents each. Table 5.1 shows some statistics related to the dataset. The mean document length in characters of the training, validation and test sets were 1741, 1743 and 1737 respectively. Also, the mean document length in words of the training, validation and test sets were 351, 351 and 349 respectively. The mean number of paragraphs in a document is 7. We noted that the mean number of words per paragraph is 51 while the maximum number of words per paragraph is 1207 in our training set. Here, we can observe that the length of the paragraphs which are the inputs to our models were relatively short, however there may be a few exceptions.

The number of style changes ranged from 0 to 20. The number of authors per document ranged from 1 to 4. The proportion of authors over documents were equal, i.e., 25% of the documents were written by one author, another 25% were written by two authors and so on. Therefore, the percentage of single author documents is 25% while the percentage of multi-author documents is 75%.

To have a better overview of the dataset, a new measure must be provided. The change ratio is the number of style changes observed over the total number of paragraph changes in the whole dataset. It is to be noted that, we assume that no style change occurs within a paragraph but rather at the end of a paragraph. A high change ratio indicates that style changes occurred between most paragraphs. On the other hand, a low change ratio indicates that style changes occurred less frequently between paragraphs. An imbalanced dataset may prompt the model to learn to predict the majority class. In this dataset, the change ratio was 0.54. Thus, the dataset can be considered balanced with respect to the paragraph level changes.

Table 5.1: Dataset statistics for PAN 2021

Dataset statistics	Training	Validation	Test
Size	11200	2400	2400
Percentage of single author documents	25%	25%	25%
Percentage of multi-author documents	75%	75%	75%
Change ratio	**0.54**	**0.54**	**0.54**
Number of authors	1-4	1-4	1-4
Range of style changes	0-20	0-19	0-19
Mean Document Length (in characters)	1741	1743	1737
Std. Dev. of Document Length (in characters)	825	872	856
Mean Document Length (in words)	**351**	**351**	**349**
Std. Dev. of Document Length (in words)	167	176	173
Mean number of paragraphs per document	7	7	7
Std. Dev. of number of paragraphs per document	3	3	3
Mean paragraph length (in words)	**51**	**51**	**51**
Std. Dev. of paragraph length (in words)	29	29	29
Maximum paragraph length (in words)	**1207**	**445**	**385**
Mean paragraph length (in characters)	**252**	**253**	**254**
Std. Dev. of paragraph length (in characters)	143	144	143

5.2 Single or Multiple Author Task

In this task, we continue to use the same feature selection strategies and classifier models as mentioned in Section 3.2 and Section 3.3 respectively.

5.2.1 Results

The results of our experiments are shown in Table 5.2. We observed that Random Forest (BOW) and Random Forest (Char Trigrams) had the highest mean CV accuracy of 0.79 with Random Forest (Char Bigrams) also close with a value of 0.78. The highest test accuracy of 0.79 was achieved by Random Forest (BOW) with Random Forest (Char Trigrams) and Random Forest (Char Bigrams) following closely.

However, in section 5.1, we observed that the dataset is rather unbalanced with the multi-author class as the majority class representing 75% of all the cases. The minority class or single-author class represented

Table 5.2: Single or Multiple Author Task Results for PAN 2021 (Accuracy)

Model	Mean Cross Validated Accuracy	Test Accuracy
Logistic Regression (BOW)	0.76	0.76
Naïve Bayes (BOW)	0.67	0.67
SVM Linear (BOW)	0.75	0.76
SVM RBF (BOW)	0.76	0.76
SVM Poly degree 3 (BOW)	0.73	0.73
SVM Sigmoid (BOW)	0.69	0.69
Random Forest (BOW)	**0.79**	**0.79**
Logistic Regression (Char Bigrams)	0.74	0.74
Naïve Bayes (Char Bigrams)	0.66	0.65
SVM Linear (Char Bigrams)	0.74	0.74
SVM RBF (Char Bigrams)	0.75	0.76
SVM Poly degree 3 (Char Bigrams)	0.73	0.73
SVM Sigmoid (Char Bigrams)	0.60	0.60
Random Forest (Char Bigrams)	0.78	0.78
Logistic Regression (Char Trigrams)	0.73	0.74
Naïve Bayes (Char Trigrams)	0.63	0.63
SVM Linear (Char Trigrams)	0.72	0.74
SVM RBF (Char Trigrams)	0.75	0.75
SVM Poly degree 3 (Char Trigrams)	0.72	0.72
SVM Sigmoid (Char Trigrams)	0.59	0.59
Random Forest (Char Trigrams)	**0.79**	0.78
CNN (Integer encoding)	0.75	0.76
CNN (GloVe embedding)	0.70	0.71
LSTM (Integer encoding)	0.75	0.75
BiLSTM (GloVe embedding)	0.71	0.71

25% of all the cases. Thus, we can assume that our random baseline is 0.75 in our experiments. As such, the Random Forest models have performed slightly better than the random baseline.

This shows that machine learning models face challenges when the dataset is imbalanced. When the imbalance is severe as in our case, the classifier has fewer examples to learn the features to model the minority class.

In case of severe class imbalance, using accuracy as the evaluation metric does not capture the class imbalance problem. One way to address the class imbalance problem, is to choose a different evaluation metric such as balanced accuracy. Balanced accuracy is defined as the recall/sensitivity for each class, averaged over the number of classes. In case of a binary classification, the balanced accuracy is equal to the arithmetic mean of sensitivity (true positive rate) and specificity (true negative rate) as shown in 5.1. When both classes are balanced, the balanced accuracy is equal to the raw accuracy.

$$balanced\_accuracy = \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right) \tag{5.1}$$

Where, given a confusion matrix, TP, FN, TN and FP represents true positives, false negatives, true negatives and false positives respectively.

In Table 5.3, we present the results of the balanced accuracy metric. Since the baseline for the balanced accuracy is at 0.5, we can observe that several classifier models are performing clearly better than the random baseline. The highest balanced accuracy for the test dataset was achieved by Logistic Regression (BOW) and SVM Linear (BOW).

We conducted paired t-tests to observe the effect of the choice of features namely, BOW, character bigrams and character trigrams on the balanced test accuracy. The p-values for the t-test between BOW and character bigrams, BOW and character trigrams and character bigrams and character trigrams were found to be 0.15, 0.12, 0.11 respectively. Thus, in all the three comparisons, no significant difference in test accuracy in the feature groups was found during a paired t-test.

Figure 5.1: Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2021

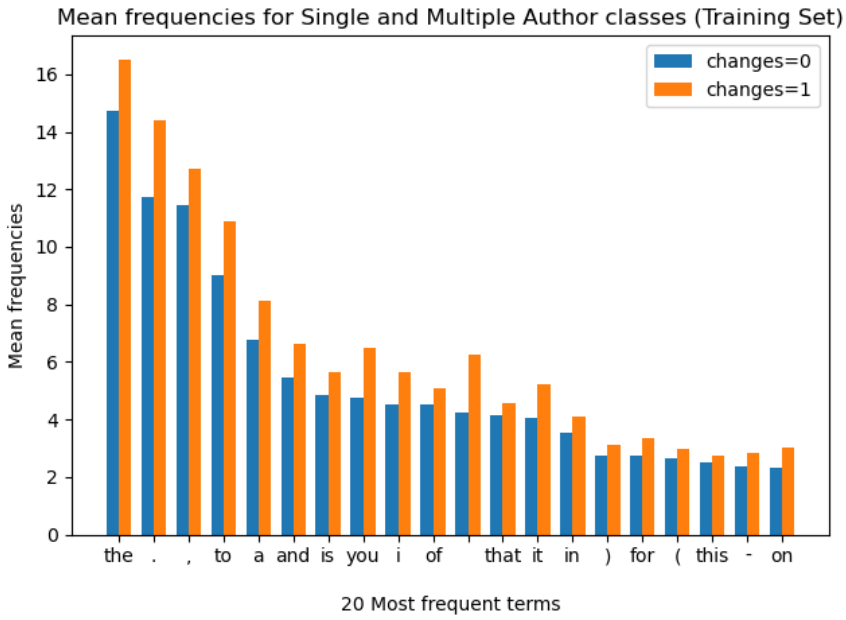
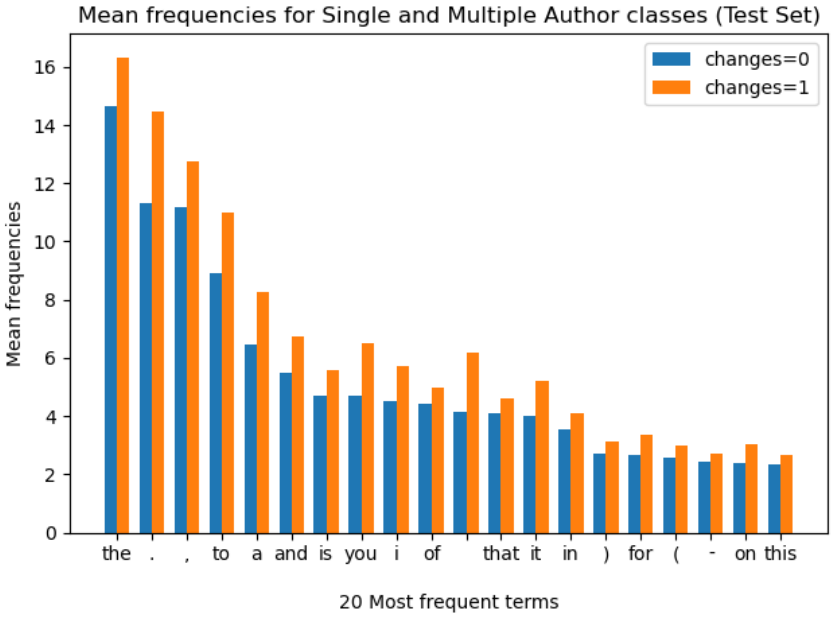


Table 5.3: Single or Multiple Author Task Results for PAN 2021 (Balanced Accuracy)

Model	Mean CV Balanced Accuracy	Test Balanced Accuracy
Logistic Regression (BOW)	**0.63**	**0.63**
Naïve Bayes (BOW)	0.59	0.58
SVM Linear (BOW)	0.62	**0.63**
SVM RBF (BOW)	0.52	0.52
SVM Poly degree 3 (BOW)	0.50	0.50
SVM Sigmoid (BOW)	0.57	0.58
Random Forest (BOW)	0.60	0.60
Logistic Regression (Char Bigrams)	0.60	0.60
Naïve Bayes (Char Bigrams)	0.60	0.58
SVM Linear (Char Bigrams)	0.60	0.59
SVM RBF (Char Bigrams)	0.52	0.52
SVM Poly degree 3 (Char Bigrams)	0.51	0.51
SVM Sigmoid (Char Bigrams)	0.47	0.47
Random Forest (Char Bigrams)	0.59	0.59
Logistic Regression (Char Trigrams)	0.57	0.57
Naïve Bayes (Char Trigrams)	0.59	0.59
SVM Linear (Char Trigrams)	0.56	0.56
SVM RBF (Char Trigrams)	0.50	0.50
SVM Poly degree 3 (Char Trigrams)	0.51	0.51
SVM Sigmoid (Char Trigrams)	0.45	0.44
Random Forest (Char Trigrams)	0.60	0.60
CNN (Integer encoding)	0.56	0.56
CNN (GloVe embedding)	0.58	0.58
LSTM (Integer encoding)	0.50	0.50
BiLSTM (GloVe embedding)	0.56	0.54

Figure 5.2: Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2021



In Figure 5.1, we plot the mean frequencies of 20 most frequent BOW features and for both the classes for the training dataset. Figure 5.2 plots the trend for the test dataset. The top 20 most frequent 20 are again quite similar to those selected in the PAN 2018 dataset and mostly contain function words and punctuations. While the mean frequencies of these features in documents with no style change (or changes=0) are still lower than those in documents with style change (or changes=1), however the difference has shrunk by a considerable margin. This could have impacted the linear separability of the two classes and thereby the performance of the models observed with the balanced accuracy.

We also observe that the mean frequencies of the PAN 2021 dataset tend to be much smaller than than the PAN 2018 and PAN 2019 dataset. This is because the mean document length of PAN 2021 (approx. 1700 characters) is lower than that of PAN 2019 (approx. 7500 characters) and PAN 2018 (approx. 4300 characters).

5.3 Paragraph-Level Style Change Location Detection task

In this task, the participants were asked to detect the location of the changes at paragraph-level. A major challenge in this problem is its multi-label variable length-output nature of classification. This is because the possible number of style changes in a document is dependent on the number of paragraphs in the document (which is variable for each document). Variable length outputs are challenging to model. Thus, by comparing stylistic similarity of two consecutive paragraphs at a time and thereby using an authorship verification approach, we can easily impose a fixed length output.

Now, we discuss the relevant background for this problem with respect to neural network approaches. Iyer and Vosoughi [20] used a BERT [21] model to learn sentence level features in the paragraph-level SCD task in PAN 2020. The sentence level features were further assimilated into a paragraph level representation and fed into a Random Forest model as a classifier. In our knowledge, no complete end-to-end neural network based model has been used for locating style changes in a document.

The style change location detection task may be viewed as a formulation of the one shot learning problem where a prediction must be made, given that, only a single example of a new class is available. One shot learning has been used in the computer vision domain to identify if two given images are similar [87]. In case of SCD, each pair of texts written by the same author can be considered as a new class. In other words, the one shot learning approach may be used to identify if two texts are similar in terms of style.

In the PAN data set, since the possibility of a style change occurs at the end of a paragraph, each document can be transformed such that any two consecutive paragraphs are paired as a single data point. The label of authorship corresponds to whether they were written by the same author or by two different authors. In this way, the SCD problem can be transformed in to the one shot learning problem.

Siamese neural networks are a widely used approach for the one shot learning problem. Koch [87] has shown the effectiveness of Convolutional Siamese neural networks for one shot image recognition. A Siamese neural network is composed of two identical twin neural networks with same weights which learn the hidden representation of two different input vectors simultaneously. The output of both these twin networks is evaluated for similarity using a suitable distance measure. In other words, the two inputs have the same output label which indicates whether they belong to the same class.

Similar approaches have been made for learning textual similarity. Mueller and Thyagarajan [94] have shown that Recurrent Siamese neural networks with LSTM (Long Short-Term Memory) cells can be used for learning sentence similarity using the SICK dataset [95]. Boenninghoff [96] achieved positive results with Hierarchical Recurrent Siamese neural networks for authorship verification with a small PAN dataset. Convolutional Siamese neural networks have also been used for large scale author identification in [97]. However, the length of the texts in their dataset (1000 words in BL2K dataset [97] and 2000 words in the FF dataset [97]) were larger as compared to PAN datasets. The mean number of words per document in the PAN 2020 SCD dataset is 351, while the mean number of words per paragraph is only 51. This is important because smaller texts have limited contextual information and therefore, more challenging, as compared to larger texts.

In the rest of this section, we present an end-to-end Siamese neural network based approach for SCD on short texts. The network architecture is based on Mueller [94], however, the unidirectional LSTM is replaced with bidirectional LSTMs and GRUs. Moreover, we use the GloVe embedding [54] and Cosine distance function. A baseline approach is also presented based on a Logistic Regression (BOW) model.

5.3.1 Data preprocessing

5.3.2 Siamese neural networks (SNN)

As mentioned previously, each document is converted into a set of data points of consecutive paragraphs. It is assumed that each paragraph is written only by one author. This intermediate data representation is shown in Table 5.4.

In this step the text of each paragraph is converted into a numeric representation format which is required for neural networks. At first, the text is transformed into lower case, then all punctuation (except the ' character) is removed. These sequences of words were tokenized and indexed, then converted to a sequence of integers (e.g., "because" = 01, "he"=02, etc.). The index created in this step represents the vocabulary.

Neural networks also require a fixed length input sequence but the length of the text in each paragraph is variable. Therefore, the maximum length threshold of the sequences needs to be set. Sequences having length less than the threshold are padded with zeroes, while sequences having length greater than the threshold are truncated. We have seen in Table 5.1 that the mean length of a paragraph is 51 words while the maximum length is 1207 words. Setting the maximum length threshold as 1207 would encourage a lot of sparse representation and increase the training time and was therefore, not considered reasonable.

After some tuning, we found that the maximum length threshold of 300 words worked the best in our experiments. The final data representation is shown in Table 5.5.

Table 5.4: Intermediate data representation of a document

author paragr. 1	**author paragr. 2**	**paragr. 1 text**	**paragr. 2 text**	**style**
1	2	I kust * realized that some ...	Because he leased ...	different
2	2	Because he leased ...	In general, I advice ...	same
2	1	In general, I advice ...	My mac is the ...	different
1	3	My mac is the ...	You should also ...	different
3	4	You should also ...	You're fine for ...	different
4	1	You're fine for ...	My modem was ...	different
1	3	My modem was ...	WPA (if possible, ...	different
3	3	WPA (if possible, ...	As to the host...	same

*The typographical error was present in the text.

Table 5.5: Finished data representation of a document

paragr. 1 text (x1)	**paragr. 2 text (x2)**	**style (y)**
[3, 24, 25, 26, 27]	[5, 6, 7, 0, 0]	1
[5, 6, 7, 0, 0]	[8, 9, 3, 10, 0]	0
[8, 9, 3, 10, 0]	[2, 11, 12, 4, 0]	1
[2, 11, 12, 4, 0]	[13, 14, 15, 0, 0]	1
[13, 14, 15, 0, 0]	[16, 17, 18, 0, 0]	1
[16, 17, 18, 0, 0]	[2, 19, 20, 0, 0]	1
[2, 19, 20, 0, 0]	[21, 22, 23, 0, 0]	1
[21, 22, 23, 0, 0]	[28, 29, 4, 30, 0]	0

5.3.3 Logistic Regression (BOW) Model

The data processing for the Logistic Regression (BOW) model is different from that of the Siamese neural networks. We start with the intermediate representation of data shown in Table 5.4.

Thereafter, the texts of all the training documents are concatenated and a BOW model based on 200 most frequent words were selected as features. The processing steps also included conversion to lower case, but no punctuations or stop words are removed since we are analyzing the style rather than the content. The pre-selected features are shown in Figure 5.3. The normalized frequencies of the tokens of both paragraphs corresponding to the pre-selected features, are produced as the input feature vectors x1 and x2 to the model.

Other variations such as removing stop words and punctuations led to a deterioration in performance probably because stop words may be a characteristic of writing style. Decreasing the number of selected features to 50 or 100 degraded the performance compared to our choice of 200. In addition, increasing the number of features led to very sparse representations and did not improve the performance.

5.3.4 Models

In this section we discuss the architecture details of the models proposed.

```
the . , to a and you is i of it that in for   on this be if have with your can as not are but or n't
's do will from so an use would /| there server one all at my which then by when what some like using
they just need up only more other any also should has was does get windows data want we could file
time no way same out how them about system into network make than work 'm may problem see 're set
files each because new run 've different - ip used even user where know might very these running
first two access something now most their drive case me much code well another good machine able
after here really sure its find however try address $ computer still think had since both over am go
many without example servers router 2 such 'd 1 disk probably 'll change possible been ca being
going create etc things those number database between while through point look client back domain
software port did check install start before add dns down question better linux connection
application every issue solution right our doing take
```

Figure 5.3: The selected features of the MFW baseline model

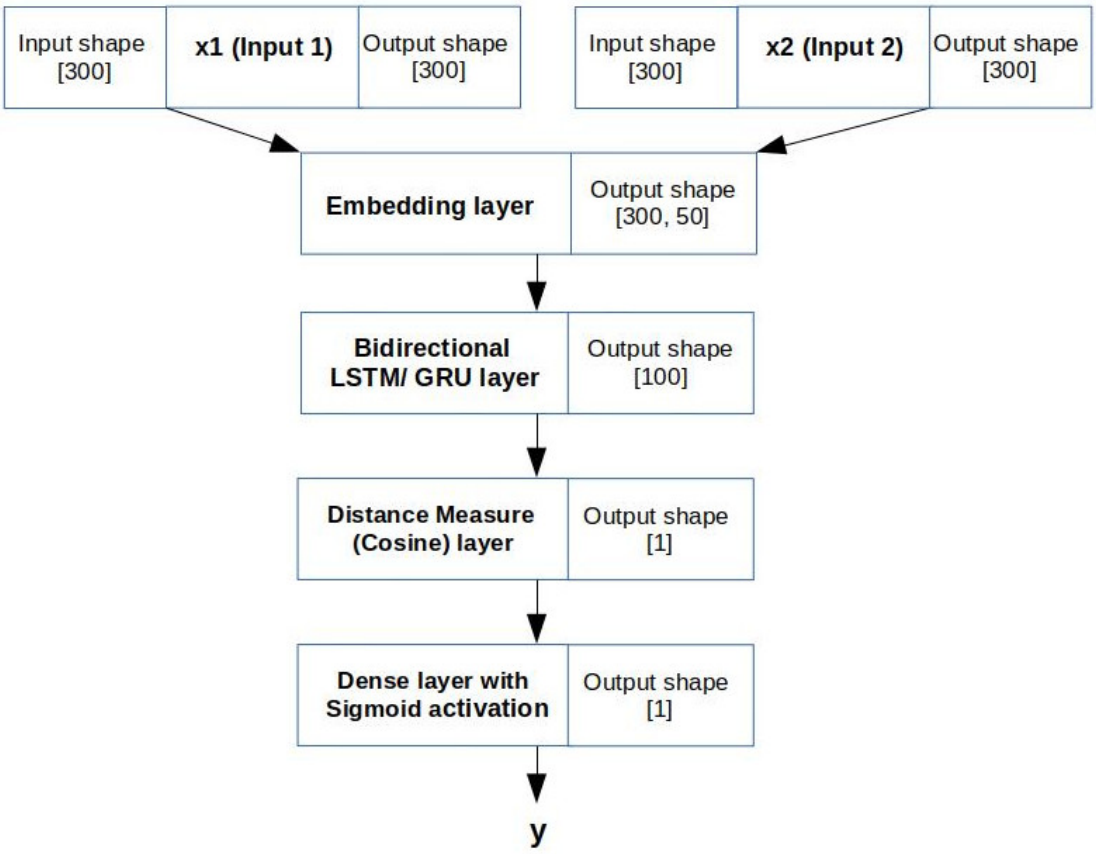


Figure 5.4: Siamese network architecture

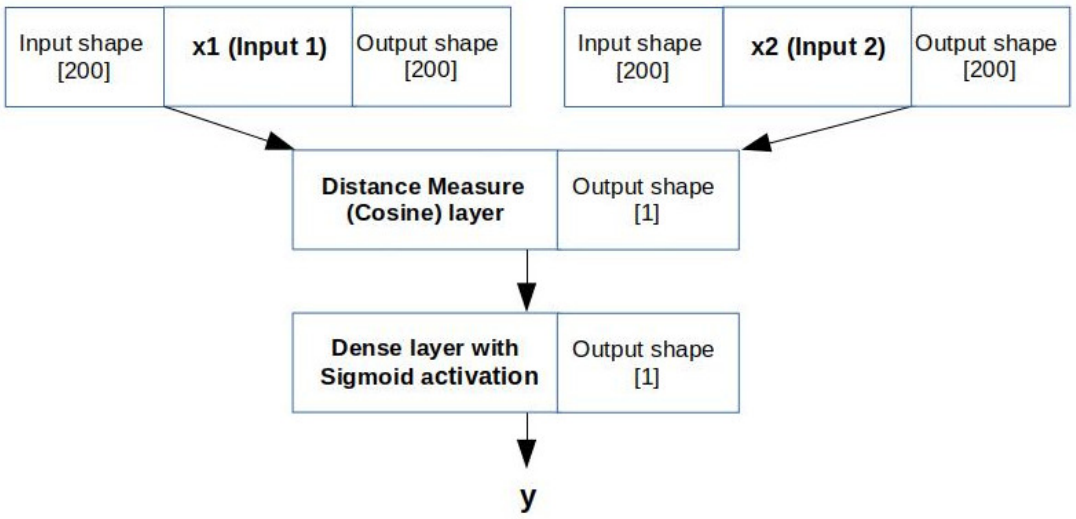


Figure 5.5: MFW baseline model architecture

5.3.4.1 Siamese neural network architecture

In Figure 5.4, the Siamese neural network architecture is presented. The two paragraphs to be evaluated for similarity are converted into sequences of integers as described in Section 3.1. Here the inputs x1 and x2 are the vectors of sequences of integers representing the corresponding paragraphs 1 and 2. The maximum size of each sequence is 300 as mentioned in Section 3.1. Therefore, the shape of both the inputs is [300].

The inputs are passed through a 50 dimensional GloVe embedding layer which embeds the inputs into their corresponding higher dimensional word vectors, resulting in an output shape of [(300, 50)]. The next layer is the bidirectional LSTM (or GRU) layer containing n=50 units with the corresponding output shape as [100]. Then, a distance function computes the similarity between the outputs of the bidirectional LSTM (or GRU) layers and outputs the shape [1]. The last layer is a dense layer with sigmoid activation which outputs the label y (binary classes where 1 = different authors and 0 = same authors). The binary cross entropy loss function is applied.

As mentioned, the embedding had 50 dimensions while the size of the indexed vocabulary was 55,904. As such, the number of parameters in the embedding layer was 2,795,200 (50 x 55,904). The number of parameters of the bidirectional LSTM layer was 40,400 while that of the bidirectional GRU layer was 30,600.

Adding another layer of bidirectional LSTM or increasing the number of units in the layer did not yield better results but increased the running time.

The total number of parameters in the Siamese neural network with bidirectional LSTM (SNN-LSTM) and the Siamese neural network with bidirectional GRU (SNN-GRU) was found to be 2,835,602 and 2,825,802 respectively.

We compared different distance functions such as Manhattan, Cosine and Matusita. However, the Cosine distance was found to perform slightly better than the others and was the chosen distance function for our models.

5.3.4.2 Logistic Regression (BOW) Model

The input feature vectors x1 and x2 were both of size [200)] as 200 features were selected. The distance layer measures the similarity between these two feature vectors. As in the case of Siamese neural networks, we compared different distance functions such as Manhattan, Cosine and Matusita. We found that the Cosine distance performed similarly to the others. The simple architecture of the Logistic Regression (BOW) model is shown in Figure 5.5.

5.3.5 Results

In Table 5.6, the results of our experiments are presented. The F1 score is used as the evaluation measure. The random baseline assumes that all documents are single author.

The SNN-LSTM model performed the best across the training, validation and test sets. The SNN-GRU model performed almost as well as the SNN-LSTM whereas the Logistic Regression (BOW) model performed the worst of the three. Both SNN-LSTM and SNN-LSTM performed much better than the random baseline.

Table 5.6: Paragraph-level Style Change Location Detection Results for PAN2021 dataset

	F1 score		
Model	**Training**	**Validation**	**Test**
SNN-LSTM	**0.70**	**0.65**	**0.65**
SNN-GRU	0.68	0.64	0.64
Logistic Regression (BOW)	0.51	0.51	0.35
Random Baseline	0.35	0.35	0.35

The SNN-LSTM model performed slightly better than the SNN-GRU in our experiments. As LSTMs have higher expressive power than GRUs, owing to their more complex design and higher number of parameters, they might perform better than GRUs, given that sufficient data is available. However, due to the higher number of parameters, the training time of SNN-LSTM was somewhat higher than SNN-GRU.

5.3.6 Blog Authorship Dataset

Additionally, we generated a SCD dataset from the Blog Authorship corpus by Schler et al. [98]. The Blog Authorship corpus is composed of blog posts of 19,320 authors collected from blogger.com. The dataset has been widely used in Authorship attribution and identification tasks such as [99], [100] and [97].

To generate a dataset for SCD, we used a similar strategy as shown in in [97]. Each document was divided into 8 segments, of which, 4 segments were used to generate two same author data point pairs. The remaining 4 segments were combined with 4 segments from another author to generate 4 different author data point pairs. We will use the words segment and paragraph interchangeably for a chunk of text.

The new dataset initially contained 54,042 data points. However, the length of the segments had a mean value of 4,906 characters and a high standard deviation of 12,361 characters. To avoid segments with large length variability, we performed a quantile analysis of segment length as shown in Table 5.7.

We divided the range of segment length distribution with the help of quantile values and their corresponding threshold values. A p_{th} quantile is defined as the value v (threshold) of the distribution which includes $p * N$ observations, with $0 < p < 1$ and N being the number of observations. In other words, the dataset sample represented by a certain threshold contains all segments with length less than the threshold. Thus 1.0 quantile of the segment length distribution is shown as 330,277, as it includes 100% of all observations. Similarly, 0.85 quantile of the segment length distribution includes 85% of all observations. As shown in Table 5.7, the mean segment length for the 0.85 quantile dataset sample is 1767 while the

Table 5.7: Segment Length quantile analysis (character length)

Quantile	Segment Length Cut off Threshold	Dataset Size	Mean Segment Length	Std. Dev. Of Segment Length	Change Ratio
1	330277	54042	4906	12361	0.50
0.95	19262	50100	2799	3543	0.51
0.9	10849	46248	2158	2257	0.53
0.85	**7156**	**42480**	**1767**	**1606**	**0.54**
0.8	5249	38961	1495	1212	0.56
0.75	3943	35545	1294	953	0.57
0.7	3157	32219	1132	761	0.59
0.65	2561	29032	999	614	0.61
0.6	2104	26008	890	501	0.63
0.5	1483	20423	717	335	0.67

Table 5.8: Dataset statistics for Blog Authorship SCD corpus

Dataset statistics	Training	Validation	Test
Size	29735	6372	6373
Change ratio	0.54	0.54	0.54
Mean number of words per segment	375	379	371
Std. Dev. Of number of words per segment	343	348	340
Maximum number of words per segment	3232	1874	1872
Mean number of characters per segment	1766	1781	1751
Std. Dev. Of number of characters per segment	1605	1626	1600

standard deviation is 1606 with a sample size of 42,480. We also note that decreasing the segment length of the dataset increases the change ratio and introduces imbalance. Thus, we chose the 0.85 quantile sample set for our experiments as the segment length standard deviation is close to the mean, while the change ratio can be considered balanced and the dataset is still fairly large.

We describe the dataset statistics for the selected dataset in in Table 5.8. The training, validation and test sets are of sizes 29735, 6372 and 6373 respectively. We also found the proportion of both classes, viz. style change=0 and style change = 1 to be balanced. We can also observe that the mean number of words per segment is around 370 which is higher than in the case of the PAN 2021 dataset.

We employ the SNN-LSTM, SNN-GRU and Logistic Regression (BOW) models as discussed previously. We share the results of our experiments in Table 5.9. Both SNN-LSTM amd SNN-GRU performed very well. The Logistic Regression (BOW) model also performed clearly better than the baseline. One of the factors positively influencing the performance is that the length of the segments in this dataset are large (as compared to the PAN2021).

5.4 Authorship Clustering (Authorship Assignment) task

In this task, the texts are to be assigned uniquely to the respective authors. We can consider this task similar to the number of authors task encountered in Section 4.3. In the subsequent sections, we discuss our approaches and models. We used the BOW approach in the feature selection for all our models.

Table 5.9: Paragraph-level Style Change Location Detection Results for Blog Authorship corpus

	F1		
	Training	**Validation**	**Test**
SNN-LSTM	0.95	0.87	0.87
SNN-GRU	0.94	0.85	0.84
Logistic Regression (BOW)	0.63	0.62	0.61
Random Baseline	0.35	0.35	0.35

Table 5.10: Example of a Distance Matrix for Hybrid Threshold Based Clustering

	$W_0(P_1)$	$W_1(P_1, P_2)$	$W_2(P_3, P_4, P_5)$	$W_3(P_6)$
$W_0(P_1)$	0	0.8	0.2	0.55
$W_1(P_1, P_2)$	0.8	0	0.56	0.42
$W_2(P_3, P_4, P_5)$	0.2	0.56	0	0.77
$W_3(P_6)$	0.55	0.42	0.77	0

5.4.1 Simple Clustering

The simple clustering approach accepts the predictions of paragraph-level style change location detection task. In this approach, we initiate a new authorship cluster whenever we encounter a style change. If on the other hand, no style changes are encountered the author is assumed to be the same. For example, say we are given a document with 6 paragraphs and predicted location changes are $[0, 1, 0, 0, 1]$. We can observe that there is a style change between paragraphs 2 and 3 and also between paragraphs 5 and 6. In this case, the simple clustering approach would predict the following clusters $[1, 1, 2, 2, 2, 3]$.

5.4.2 Simple Threshold Based Clustering

This approach is based on the Threshold Based Clustering (TBC) method discussed in Section 4.3.3.1. In the simple TBC approach, the predictions of the paragraph-level style change location detection are ignored. We recalculate the distances between the paragraphs to generate a distance matrix. Using appropriate distance measures and Add Window and Merge Cluster thresholds, the authorship clusters are predicted.

5.4.3 Hybrid Threshold Based Clustering

In this approach, the style change information obtained from paragraph-level style change location are used to define new windows. For example, say we are given a document with 6 paragraphs and predicted location changes are $[0, 1, 0, 0, 1]$. This approach assumes that paragraphs 1 and 2 are part of the same window, say W_2, while paragraphs 3,4 and 5 are part of another window, say W_3. Paragraphs belonging to the same window are concatenated. Then, from each window, 200 features are selected. A distance matrix is created by measuring the distance between pairs of windows. On this distance matrix, a clustering algorithm can be applied. We show an example of the new distance matrix in Table 5.10.

5.4.4 Agglomerative Clustering

Agglomerative Clustering is a form of Hierarchical clustering that builds hierarchical clusters by merging or splitting them successively using a bottom up approach. At first, each paragraph will be represented in its own cluster. Subsequently, clusters are successively merged together.

The commonly used linkage criteria which determine how the choice of merge operation are as follows.

Table 5.11: Authorship Clustering Results for PAN 2021 dataset

	F1 Score		
Models	**Training**	**Validation**	**Test**
Simple Clustering	**0.36**	**0.35**	**0.35**
Simple TBC	0.29	0.29	0.30
Hybrid TBC	0.26	0.25	0.26
Agglomerative Clustering (Ward)	0.24	0.23	0.24
Agglomerative Clustering (Complete)	0.24	0.24	0.24
Agglomerative Clustering (Average)	0.25	0.24	0.24
Agglomerative Clustering (Single)	0.24	0.25	0.25
Random Baseline (single author)	0.17	0.17	0.17

- **Ward linkage** minimizes the sum of squared differences within all clusters.
- **Complete linkage** minimizes the maximum distance between pairs of clusters.
- **Average linkage** minimizes the average of the distances between all items belonging to both clusters.
- **Single linkage** minimizes the distance between the closest observations of pairs of clusters. In other words, the pair of clusters sharing the smallest distance is selected.

Among the distance measures such as Manhattan, Cosine and Euclidean, the best results were observed for the Cosine distance.

5.4.5 Results

The evaluation measure used is F1 score. We show the results of our experiments in Table 5.11. The random baseline assumes that each document is written by a single author only and therefore has only one cluster.

We can observe that the Simple Clustering approach performed the best, followed by Simple TBC. The different agglomerative clustering models performed relatively similarly.

It is to be noted that while we knew that the maximum number of authors is 4 in the given dataset, such information was not explicitly fed into the models. As a result, it is possible that a higher number of authors were predicted in certain cases.

In PAN 2021, we participated for tasks 1 and 2 and presented the siamese neural network approach for both tasks. For task 1, we attained the 3rd position whereas for task 2, we attained the 5th position with respect to performance [101].

6
PAN 2020

In PAN 2020 SCD challenge [11], the first task was to find if a document is written by single or multiple authors. The second task focused on the paragraph-level style change detection task. In this case, the system must identify whether the paragraphs have been written by the same author.

6.1 Dataset

The PAN 2020 dataset was also composed of forum posts from different 'StackExchange' network sites. 'StackExchange' is a network of over 170 communities and covers a wide variety of topics such as technology, culture/recreation, life/arts, science, professional and business. Taking advantage of the variety of topics, the dataset was divided into two sub-datasets.

- **dataset-narrow**: This dataset contained topics related to computer technology only.
- **dataset-wide**: In this dataset, in addition to computer technology, other topics such as economics, literature, philosophy, and mathematics were included.

In Table 6.1, we share some statistics belonging to the train, validation and test samples for both the datasets. In the dataset-narrow, there are 3418, 1713 and 1701 documents, whereas in the dataset-wide, there are 8030, 4019 and 3995 documents respectively. The percentage of single and multi-author documents are both 50%. As such, the dataset can be considered balanced for the single and multiple author task.

We introduced the change ratio in Section 5.1. As mentioned previously, a low change ratio indicates that style changes occurred less frequently between paragraphs. In both the datasets, we observed that the change ratio was low. The change ratio of dataset-narrow is 0.12 while that of dataset-wide is 0.17. This indicates that both datasets are highly imbalanced with respect to paragraph level changes. We shall discuss the effect of change ratio on the evaluation for the paragraph-level style change location detection task in Section 6.2.1.

In both datasets, the number of authors ranged between 1 and 3. The number of style changes ranged from 0 to 10. The mean number of paragraphs per document is 50 for dataset-narrow and 35 for dataset-wide. In the PAN 2020 dataset, the number of style changes in a document occurred occasionally. As a

Table 6.1: Dataset statistics for PAN 2020

Dataset Statistics	**Dataset-Narrow**			**Dataset-Wide**		
	Train	**Validation**	**Test**	**Train**	**Validation**	**Test**
Size	3418	1713	1701	8030	4019	3995
Percentage of single author documents	50%	50%	50%	50%	50%	50%
Percentage of multi author documents	50%	50%	50%	50%	50%	50%
Change Ratio	**0.12**	**0.12**	**0.12**	**0.17**	**0.17**	**0.17**
Number of authors	1-3	1-3	1-3	1-3	1-3	1-3
Range of style changes	0-10	0-10	0-10	0-10	0-10	0-10
Mean Document Length (in characters)	11781	11869	11694	12058	12083	12059
Std. Dev. of Document Length (in characters)	3317	3344	3319	3645	3676	3632
Mean number of paragraphs per document	50	50	50	35	35	35
Std. Dev. of number of paragraphs per document	22	22	22	18	19	19
Mean number of words per paragraph	49	49	48	70	71	70
Std. Dev. Of number of words per paragraph	83	83	82	124	127	125
Maximum number of words per paragraph	1966	2248	2705	3238	2704	3623
Mean number of characters per paragraph	236	237	235	343	346	342
Std. Dev. of number of characters per paragraph	398	395	395	601	612	602

result, the dataset was not considered balanced from the point of view of the paragraph-level style change location detection task.

6.2 Single or Multiple Author Task

We have used the same feature selection strategies and classifier models as mentioned in Section 3.2 and Section 3.3 respectively.

6.2.1 Results

The results of the dataset-narrow are shown in Table 6.2. We observed that SVM RBF (BOW) had the highest mean CV accuracy of 0.78. In addition to SVM RBF (BOW), two other models, namely, SVM Poly degree 3 (BOW) and Random Forest (BOW) achieved the best performance of 0.77. We conducted a t-test to observe the statistical significance in differences of performances of these three models. On comparing SVM RBF (BOW) and SVM Poly degree 3 (BOW), for $\alpha = 0.5$, the $p\_value = 0.0055$ was observed. Since $p < \alpha$, we can reject the null hypothesis and may conclude that the performance of the

Table 6.2: Single or Multiple Author Task Results for PAN 2020 Dataset-Narrow

Model	**Mean Cross Validated Accuracy**	**Test Accuracy**
Logistic Regression (BOW)	0.67	0.67
Naïve Bayes (BOW)	0.68	0.68
SVM Linear (BOW)	0.67	0.67
SVM RBF (BOW)	**0.78**	**0.77**
SVM Poly degree 3 (BOW)	0.76	**0.77**
SVM Sigmoid (BOW)	0.58	0.57
Random Forest (BOW)	0.76	**0.77**
Logistic Regression (Char Bigrams)	0.66	0.67
Naïve Bayes (Char Bigrams)	0.66	0.66
SVM Linear (Char Bigrams)	0.67	0.68
SVM RBF (Char Bigrams)	0.75	0.74
SVM Poly degree 3 (Char Bigrams)	0.74	0.76
SVM Sigmoid (Char Bigrams)	0.49	0.50
Random Forest (Char Bigrams)	0.76	0.76
Logistic Regression (Char Trigrams)	0.66	0.68
Naïve Bayes (Char Trigrams)	0.65	0.66
SVM Linear (Char Trigrams)	0.67	0.69
SVM RBF (Char Trigrams)	0.76	0.75
SVM Poly degree 3 (Char Trigrams)	0.75	0.76
SVM Sigmoid (Char Trigrams)	0.49	0.50
Random Forest (Char Trigrams)	0.74	0.73
CNN (Integer encoding)	0.67	0.72
CNN (GloVe embedding)	0.58	0.61
LSTM (Integer encoding)	0.50	0.50
BiLSTM (GloVe embedding)	0.53	0.55

two algorithms is significantly different.

The results of dataset-wide are shown in Table 6.3. Here the best mean CV accuracy of 0.73 was attained by SVM RBF (Char Trigrams) and SVM Poly degree 3 (Char Trigrams). The best test accuracy of 0.75 was attained by SVM Poly degree 3 (Char Trigrams).

In both datasets, since the number of single and multi-author documents are well balanced, we can observe that the models used by us have performed well by a considerable margin.

Deep Learning approaches (CNN, LSTM, BiLSTM) did not perform well in this task. Thus, the use of appropriate kernels (in SVM models) and the determination of the effective parameter values can play an important role in the overall effectiveness of the system.

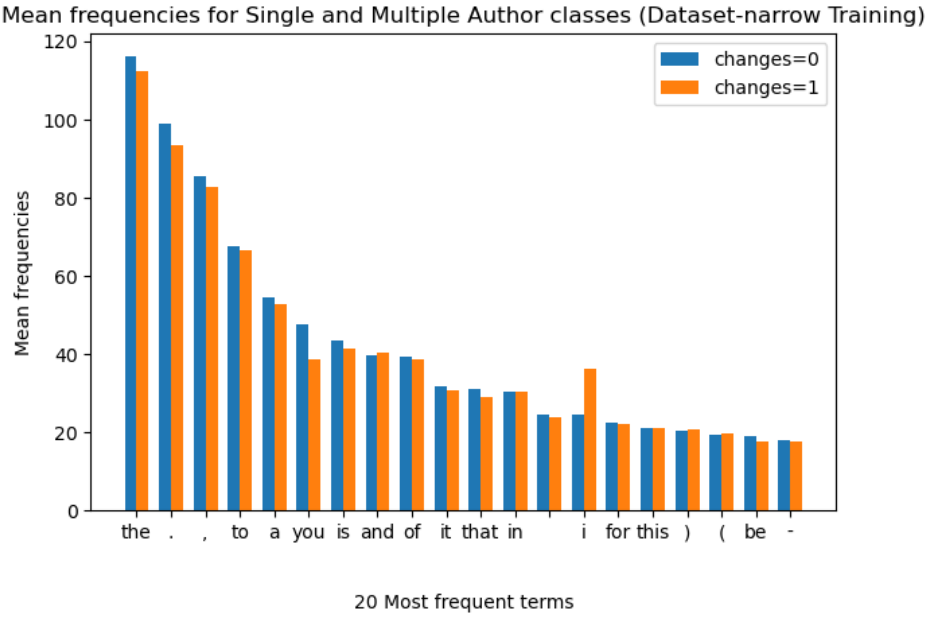
We conducted paired t-tests to observe the effect of the choice of features namely, BOW, character bigrams and character trigrams on the balanced test accuracy. For dataset-wide, the p-values for the t-test between BOW and character bigrams, BOW and character trigrams and character bigrams and character trigrams were found to be 0.08, 0.54, 0.36 respectively. For dataset-narrow, the p-values for the t-test between BOW and character bigrams, BOW and character trigrams and character bigrams and character trigrams were found to be 0.11, 0.16, 1 respectively. Thus, for both datasets, no significant difference in test accuracy in the feature groups was found during a paired t-test.

Again, we try to observe the differences of the mean frequencies of 20 most frequent BOW features and for both the classes in Figure 6.1 and Figure 6.2 for dataset-narrow. The trend here seems to have changed for certain features. We even observe that the mean frequencies of some features (such as "the",

Table 6.3: Single or Multiple Author Task Results for PAN 2020 Dataset-Wide

Model	Mean CV Accuracy	Test Accuracy
Logistic Regression (BOW)	0.62	0.63
Naïve Bayes (BOW)	0.62	0.61
SVM Linear (BOW)	0.62	0.64
SVM RBF (BOW)	0.71	0.72
SVM Poly degree 3 (BOW)	0.69	0.70
SVM Sigmoid (BOW)	0.54	0.56
Random Forest (BOW)	0.72	0.72
Logistic Regression (Char Bigrams)	0.63	0.62
Naïve Bayes (Char Bigrams)	0.60	0.61
SVM Linear (Char Bigrams)	0.63	0.62
SVM RBF (Char Bigrams)	0.70	0.70
SVM Poly degree 3 (Char Bigrams)	0.68	0.69
SVM Sigmoid (Char Bigrams)	0.49	0.48
Random Forest (Char Bigrams)	0.70	0.71
Logistic Regression (Char Trigrams)	0.63	0.63
Naïve Bayes (Char Trigrams)	0.59	0.58
SVM Linear (Char Trigrams)	0.63	0.63
SVM RBF (Char Trigrams)	**0.73**	0.74
SVM Poly degree 3 (Char Trigrams)	**0.73**	**0.75**
SVM Sigmoid (Char Trigrams)	0.48	0.48
Random Forest (Char Trigrams)	0.70	0.70
CNN (Integer encoding)	0.56	0.65
CNN (GloVe embedding)	0.55	0.56
LSTM (Integer encoding)	0.50	0.50
BiLSTM (GloVe embedding)	0.52	0.53

Figure 6.1: Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2020 Dataset-Narrow



".", "is", "you") in documents with no style change (or changes=0) are greater than those in documents with style change (or changes=1). For some features ("i"), the opposite trend is observed while for yet some others ("in") the difference is close. Even if the trend may have reversed for certain features, the data may still be linearly separable. We plot the same trends for the dataset-wide in Figure 6.3 and Figure 6.4 for training and test sets respectively and observe a similar trend as in the case of dataset-narrow. Since the mean document length of PAN 2020 is the highest among all the PAN datasets, the mean frequencies of the PAN 2020 dataset are the largest.

6.3 Paragraph-Level Style Change Location Detection task

We have encountered the paragraph-level style change location detection task in the Chapter 5. We follow the same approaches as discussed previously in Section 5.3

6.3.1 Results

Following the PAN-CLEF practice, we used the macro average F1 measure for the evaluation of this task. The results of the evaluation are shown in Table 6.4 for dataset-narrow and Table 6.5 for dataset-wide. We can observe that none of the models perform better than the random baseline. Since the F1 measure is macro averaged over all the classes, all classes are given equal importance.

In PAN 2020, we participated and presented an adapted version of TBC and WBC for both tasks and attained the 3rd position [11].

We have noted that the given datasets are highly imbalanced by measuring the change ratio in Section 6.1. Since the number of paragraph changes is quite low in the dataset, the model learns to predict no change i.e., both paragraphs are written by the same author.

Figure 6.2: Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2020 Dataset-Narrow

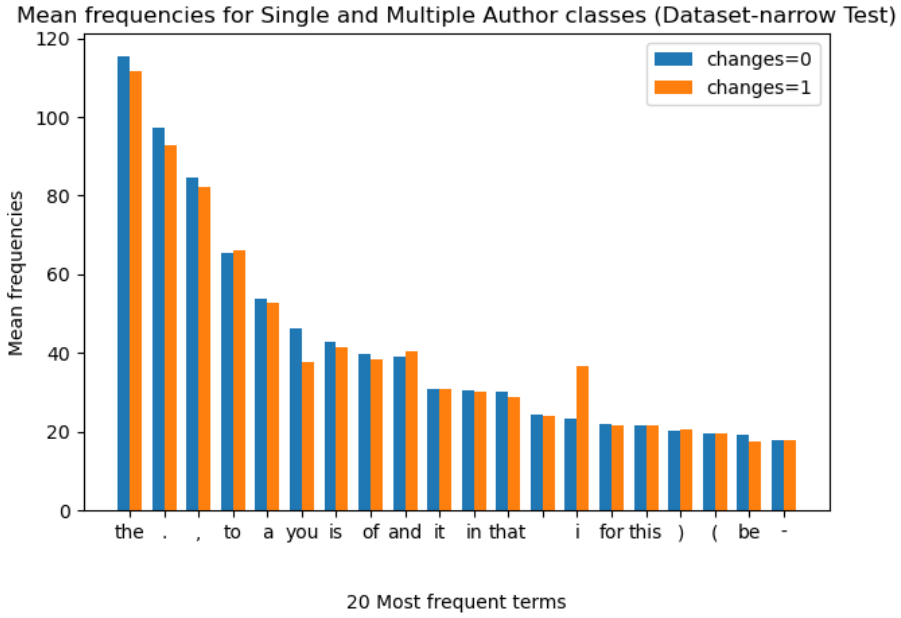


Table 6.4: Paragraph-level Style Change Location Detection Results for PAN2020 Dataset-Narrow (Original)

	F1 score		
Model	**Training**	**Validation**	**Test**
SNN-LSTM	0.47	0.47	0.47
SNN-GRU	0.47	0.47	0.47
Logistic Regression (BOW)	0.47	0.47	0.47
Random Baseline	0.47	0.47	0.47

Machine learning models do not perform so well on highly unbalanced data as the models tend to learn to predict the more frequent class. As a result, we wanted to observe if the models performed well on balanced datasets. To obtain a balanced dataset, one may either undersample the majority class (contains no style change) or oversample the minority class (contains style change). To avoid the impact of duplicated entries of the minority class, we decided to undersample the majority class randomly. By using such a sampling strategy, since we randomly undersample the majority class, the integrity of the document is no longer maintained.

The new and balanced dataset-narrow contained 19,604, 9,994 and 9,722 training, validation and test datapoints. Similarly, the new and balanced dataset-wide contained 46,008, 23,148 and 23,030 training, validation and test data points. We note that here a single datapoint contains a pair of paragraphs p1 and p2 and the associated label indicating if both the paragraphs were written by the same author or two different authors.

We show the results of the model performance on the balanced datasets in Table 6.6 for dataset-narrow and Table 6.7 for dataset-wide. We observed that for dataset-narrow, SNN-GRU outperformed SNN-LSTM by a small margin. We can also observe that the Logistic Regression (BOW) model performed clearly

Table 6.5: Paragraph-level Style Change Location Detection Results for PAN2020 Dataset-Wide (Original)

	F1 score		
Model	**Training**	**Validation**	**Test**
SNN-LSTM	0.46	0.46	0.46
SNN-GRU	0.45	0.45	0.45
Logistic Regression (BOW)	0.45	0.45	0.45
Random Baseline	0.45	0.45	0.45

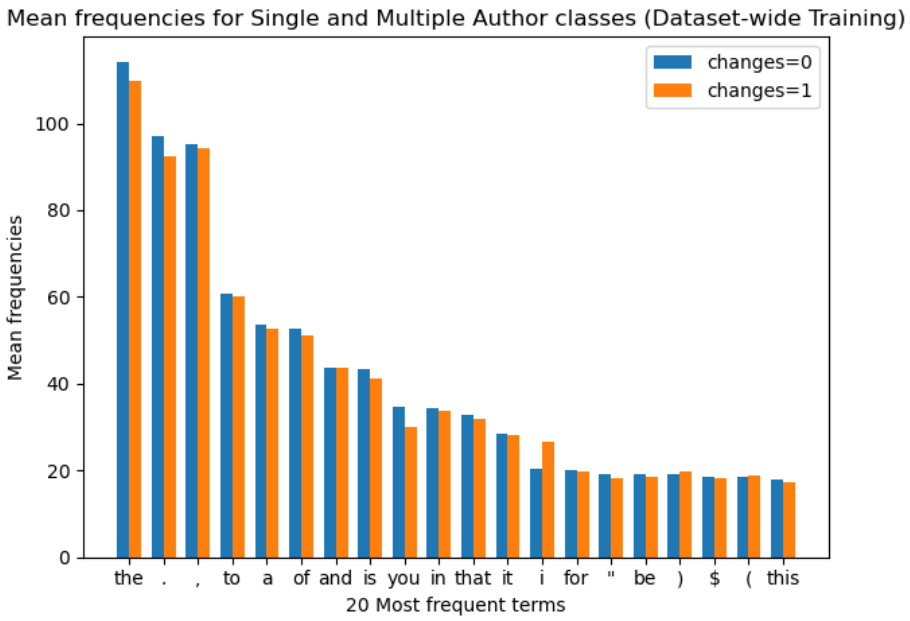
Table 6.6: Paragraph-level Style Change Location Detection Results for PAN2020 Dataset-Narrow (Balanced)

	F1 score		
Model	**Training**	**Validation**	**Test**
SNN-LSTM	0.58	0.56	0.56
SNN-GRU	0.61	0.58	0.57
Logistic Regression (BOW)	0.43	0.43	0.43
Random Baseline	0.33	0.33	0.33

Table 6.7: Paragraph-level Style Change Location Detection Results for PAN2020 dataset-wide (Balanced)

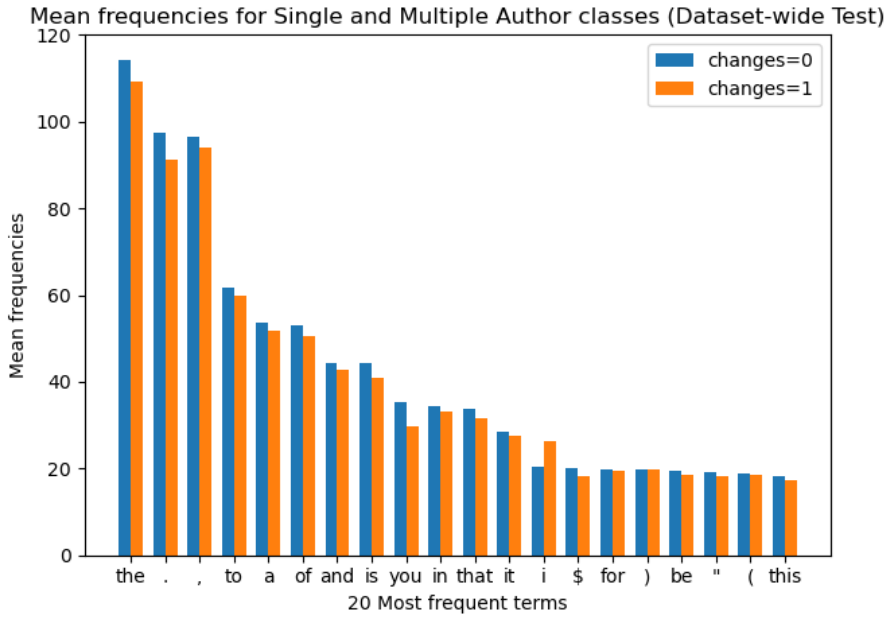
	F1 score		
Model	Training	Validation	Test
SNN-LSTM	0.66	0.64	0.63
SNN-GRU	0.63	0.61	0.61
Logistic Regression (BOW)	0.52	0.52	0.52
Random Baseline	0.33	0.33	0.33

Figure 6.3: Mean frequencies for Single and Multiple Author classes (Training Set) in PAN 2020 dataset-wide



better than the random baseline. The performance of the models on dataset-wide was better than that on dataset-narrow. Here, SNN-LSTM performed the best followed by SNN-GRU and then by Logistic Regression (BOW).

Figure 6.4: Mean frequencies for Single and Multiple Author classes (Test Set) in PAN 2020 dataset-wide



7 Discussion

This thesis explored SCD approaches in different datasets. This chapter presents a summary of contributions and findings throughout the thesis, along with the limitations of the proposed approaches and datasets. Further, we propose directions for future research.

1. **Single or Multiple Author Task**: The task here was to find out whether a document is written by one or more authors. Our feature selection strategy was to use 500 features for BOW, character bigrams and character trigrams. For neural network based approaches, word embeddings such as GloVe was also used. As learning models, we have evaluated Logistic Regression, Naïve Bayes, SVM, Random Forest, CNN, LSTM and in some cases a simple rule based one.

   Our experimental results show that the chosen feature selection and methods are effective in achieving good performance for the single or multiple author problem for all the PAN datasets. Even though PAN 2021 was not balanced in terms of the target class proportions, by choosing an appropriate evaluation strategy we could demonstrate the effectiveness of our approach.

2. **Paragraph-Level Style Change Location Detection task**: The task here is to determine the location of the style changes in the document, given that multiple authors are involved. We assume that the style changes may occur only between the paragraphs. We explored this problem in PAN 2021, PAN 2020 (both dataset-narrow and dataset-wide) and Blog Authorship Dataset. Our approach was to convert the problem into a series of authorship verification cases between two paragraphs at a time. Both paragraphs were transformed into GloVe embedded vectors and trained in Siamese LSTM/ GRU models to estimate their similarity. In our experiments we observed that, SNN models are capable of identifying same author vs different author texts even for relatively short texts (mean paragraph length of PAN 2021 was 51 words). We also introduced another dataset called the Blog Authorship corpus, to test the paragraph level SCD problem. In case of the Blog Authorship corpus, the size of the paragraphs/segments were larger (mean segment length was ~370 words) where we observed much improved results. In case of PAN 2020, the dataset was unbalanced as the style changes occurred occasionally in the documents. When we balanced the dataset by sampling, we could observe positive results of our approach.

3. **Authorship Clustering**: Here the number of distinct authors is determined by authorship clustering within documents. In PAN 2019, to determine the number of authors, we contributed two algorithms

namely TBC and WMC and observed WMC performing better than TBC. We examined and compared the effect of different distance measures and did not find a clear winner among them. In PAN 2021, to assign the paragraphs to their respective authors (authorship assignment), we compared different clustering approaches including variations of TBC and other agglomerative approaches. We also included results from the paragraph level SCD task as a guide for Hybrid TBC. We observed that the simple clustering technique performed the best followed by Simple TBC.

7.1 Limitations of the Studied Models

1. Casting the single or multiple author problem as a binary classification is a simple strategy. In our experiments, we have shown the effectiveness of such an approach. However, it is not easy to explain the rationale behind the model selecting either target class.

2. The assumption that changes occur only at the end of a paragraph is a simplification and may not be reasonable in the real world.

3. Our approaches have been tested mostly on generated datasets from Q&A and blogs in English. However, further verifications are required on texts written in other languages or covering other genres such as novels, scientific papers etc.

4. Clustering approaches such as TBC and WMC need to be tuned for various thresholds.

5. In our experiments, we did not observe a significant difference between the performance of LSTM and GRU networks. Both these networks require a high number of parameters and large amount of training examples and higher training time than other traditional approaches. As such, this can be a limiting factor when being applied to the SCD problem.

7.2 Dataset Limitations

Although, collaboration is very common nowadays, it is challenging to have large labelled datasets. PAN offers a number of datasets on this topic that simulate real world scenarios and help to fill this gap. The variation in topics help to keep the focus on the style changes.

However, we observed that certain PAN datasets had some peculiar characteristics. For instance, we observed that for PAN 2018, the character mean document length of documents with style changes was clearly higher than those without any style changes. For the PAN 2019 dataset, we observed that duplicated sentences tended to exist in documents written by multiple authors.

Although datasets in the real word are expected to be unbalanced, the machine learning models prefer balanced datasets as they can learn to give equal priority to each class. The PAN 2021 dataset for instance, was not balanced for the single or multiple author problem. In the PAN 2020 dataset, the style changes occurred occasionally in the documents. Thus, the dataset was not balanced with respect to the paragraph-level style change location detection problem.

Further, in the single or multiple author problem, we compared the top 20 features of the BOW approach and plotted their mean frequencies for both target classes. We could observe some clear differences between both target classes for PAN 2018 and 2019 datasets. It is not clear why the documents written by single authors tend to have lower mean frequencies than those written by multiple authors.

Such peculiar characteristics may have been introduced during the creation of the dataset. By appropriate sampling strategies, we can minimize the effects of unbalanced data. In some cases, we changed the evaluation metrics to better represent unbalanced data.

7.3 Future Work

In this section, we propose some future directions with the aim of extending the approaches presented in this thesis

1. **Creating SCD Datasets**: As highlighted previously, there is a dearth of datasets dedicated to the problem of SCD. In this work, we have focused on Q&A, blogs for generating datasets, whereas there are many potential sources such as studying collaboration in academia work. The creation of standardized datasets and benchmarks will help in developing an effective evaluation strategy.

2. **Languages**: The majority of PAN challenges with SCD has focused exclusively on the English language although PAN offers corpora for different languages for tasks such as author profiling and attribution. In the future, we would like to test with different languages to evaluate our language independent approaches.

3. **New neural network approaches**: A lot of progress is being made in NLP with attention based transformer models such as BERT [21]. Recently, BERT based models are being explored for use in various authorship attribution [102], authorship verification [20, 103, 104] based tasks. A possible direction would be to explore attention based siamese neural networks for SCD as well as authorship verification. Another possible direction would be to explore various other word embedding techniques including but not limited to FastText, ngram2vec, SGNS, ELMo.

4. **Explainability in Neural networks**: While neural networks are being widely adapted in NLP, explainability techniques are still actively developing. [105] describes that explainability techniques may be local (explains the model's prediction on a specific input) vs global (explains the model's predictive process in general) or self-explaining (generates explanation during the prediction process) vs post-hoc (generates explanation after prediction requiring additional steps). Techniques for visualization of important features are surrogate model comparisons are some popular techniques. As such, more research for neural networks in NLP is required and anticipated.

   In the future, adapting the neural network approaches in SCD with explanations about feature importance, examples, surrogate model comparisons will enhance the reliability and acceptability.

8 Conclusion

The goal of this thesis was to identify and propose suitable methods for SCD. We believe that we have achieved this goal in the following manner with the following contributions. We defined the sub problems in SCD and dealt with them separately. The advantage of doing so is that every step can be evaluated independently. We experimented with different feature selection methods to select the best possible strategies for our problems while attempting a general topic independent approach. Then, methods with suitable evaluation strategies were proposed using PAN datasets from 2018 to 2021. Additionally, we explored peculiar characteristics of the data and used them to our advantage.

We approached the single or multiple author problem as a binary classification task where popular feature selection strategies such as BOW, character bigrams, character trigrams, word embeddings were applied with a feature set size of 500. Many well known models such as Logistic Regression, SVM, Random Forest, Naïve Bayes and also neural network models such as CNN, LSTM, BiLSTM were employed as the classifier. For all the datasets from PAN 2018 to PAN 2021, we found encouraging results. We achieved a test accuracy of 0.68 for PAN 2018, 0.88 for PAN 2019, 0.77 for dataset-narrow and 0.75 for dataset-wide in PAN 2020 and 0.79 for PAN 2021. For this task, our approach performed better than the proposed solutions in the PAN 2019 evaluation. In all other cases, we performed better than certain approaches, but clearly better than the random baseline. We found that BOW, character bigrams and trigrams were equally effective as features for this problem. SVM models performed consistently well with different kernels. We plotted the mean frequencies for 20 most frequent terms and observed an interesting trend where a clear difference between the two target classes was observed for all the PAN datasets. A possible explanation for this trend lies with the dataset generation procedure.

One of the difficulties faced in the paragraph-level style change location detection problem was its multi-label variable length output classification nature. We tackled this issue by comparing the stylistic similarities between two consecutive paragraphs at a time. We also presented a novel end-to-end neural network based approach for the paragraph-level style change location detection problem encountered in PAN 2020 and PAN 2021. As possible architectures, we selected and two models on the Deep-Learning paradigm, the LSTM and the GRU. Both paragraphs were encoded with GloVe embeddings and pushed through a siamese network to generate their individual representations which were further compared using cosine distance measure. We found that the SNN-LSTM (test accuracy of 0.65) performed comparably with the SNN-GRU (test accuracy 0.64). The Logistic Regression (BOW) model performed at the level of

the random baseline with a test accuracy of 0.35. Further, we created a new dataset based on the Blog Authorship corpus. Here the SNN-LSTM achieved a test accuracy of 0.87 while the SNN-GRU achieved 0.84 and the Logistic Regression (BOW) model achieved 0.61, clearly beating the random baseline of 0.35. We must however note that in case of the Blog authorship corpus based dataset, the length of a segment or paragraph (mean length of 1767 characters) was much higher than that of PAN 2021 (mean length of 252 characters). This shows that length of the text is an important factor in the problem.

In PAN 2020, the datasets were unbalanced with respect to their change ratio. On the balanced dataset-narrow dataset, SNN-LSTM (test accuracy 0.56) performed comparably with SNN-GRU (test accuracy 0.57). The Logistic Regression (BOW) model achieved a test accuracy of 0.43. On the balanced dataset-wide dataset, SNN-LSTM achieved a test accuracy 0.63 followed by SNN-GRU with 0.61 and Logistic Regression with 0.52.

The results of our experiments suggest that a siamese neural network based approach can be a viable solution to the paragraph-level style change location detection for multi-author documents.

Under the authorship clustering problem, two tasks were presented. In PAN 2019, we were asked to predict the number of authors of a document. We took an algorithmic approach and proposed two algorithms. At first, we split the document in to paragraph sized windows. Then we took top 50 most frequent words in each document and created a feature vector for each paragraph. The mutual distances among the window feature vectors were measured (using distance measures such as Manhattan, Euclidean, Matusita, Tanimoto, Cosine etc.) to create a distance matrix. In the threshold Based (TBC) algorithm, the windows which are the closest in the distance matrix are selected iteratively to try and to put them in an appropriate cluster based on Add Window and Merge Cluster thresholds. In the Window Merge Clustering algorithm (WMC), in each iteration the most similar windows are combined to generate a new window. Thereafter, the distance matrix is recalculated with similar steps followed as in the case of TBC. In addition to accuracy, we also introduced the error based OCI measure to incorporate the ordering of the classes. In our evaluation, we found that WMC (test accuracy 0.4, OCI 0.76, rank 0.32) performed better than the TBC (test accuracy 0.33, OCI 0.73, rank 0.30) for accuracy but lagged behind slightly for OCI. When we incorporated the special case of duplicated sentences as a feature, we observed a reversed trend with TBC (0.36) attaining a higher rank than WMC (0.32). An improvement in accuracy was observed with WMC achieving a test accuracy of 0.56 and TBC achieving 0.59. However, this also increased the error measure OCI, such that WMC reached a value of 0.93 while TBC reached 0.86. Further, we compared the effect of different distance measures on the performance of the two algorithms. We found that overall WMC performed better than TBC in terms of rank for most of the measures. Cosine seems to do better marginally for the TBC algorithm but performed the worst for the WMC algorithm. There was no clear winner among the distance measures observed.

The second task under the authorship clustering problem is to assign paragraphs uniquely to the respective authors (authorship assignment) and was presented in PAN 2021. We proposed different approaches such as simple clustering based on the results of the paragraph-level style change location detection which simply new authorship cluster whenever a style change was encountered. TBC along with hybrid TBC which incorporated the results of the paragraph-level style change location detection were also proposed. Finally, we also proposed agglomerative clustering with different linkages such as Ward, complete, single and average. We found that the Simple Clustering approach performed the best, followed by Simple TBC while the agglomerative clustering models performed relatively similarly. Our results show that clustering remains a challenging task and further research is required to resolve the issue adequately.

One of the main reasons why SCD is hard to solve, is due to the lack of writing samples of authors or the knowledge of the number of candidate authors. Another major challenge is the short length of the texts or paragraphs. When we increased the length of the paragraphs or segments in case of the Blog authorship corpus, we observed a sharp increase in performance.

In this thesis, we attempted to combine the simple approaches with various types of approaches such

as algorithmic or neural networks to tackle the SCD problem. While some of our approaches have shown positive results for the single or multiple author and paragraph-level style change location detection task, authorship clustering for under SCD remains challenging. In conclusion, we state while simple approaches can be effective in certain cases, we must experiment more with combinations of models and also the rapidly evolving neural networks in authorship studies.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Prof. Jacques Savoy for giving me the opportunity to pursue a doctorate and for inspiring me to be curious and persisistent. I take the opportunity to also thank the members of the jury, namely Prof. Josiane Mothe (IRIT, University of Toulouse), Prof. Mascha Kurpicz-Briki (Bern University of Applied Sciences) and Dr. Valerio Schiavoni (Université de Neuchâtel) for the time they devoted to evaluating this dissertation.

I thank my colleagues and friends at the University of Neuchatel for their support and for the interesting times we spent together. I am also indebted to my husband Shiva, family and friends who continue to put their faith in me and without whom, none of this would be remotely possible.

Bibliography

[1] Patricia Lorimer Lundberg. George eliot: Mary ann evans's subversive tool in" middlemarch?". *Studies in the Novel*, 18(3):270–282, 1986.

[2] Patrick Juola. The rowling case: A proposed standard analytic protocol for authorship questions. *Digital Scholarship in the Humanities*, 30(suppl_1):i100–i113, 2015.

[3] Frederick Mosteller and David L Wallace. *Applied Bayesian and classical inference: the case of the Federalist papers*. Springer Science & Business Media, 2012.

[4] Dorte Henriksen. The rise in co-authorship in the social sciences (1980–2013). *Scientometrics*, 107(2):455–476, 2016.

[5] Sheldon R Gelman and Margaret Gibelman. A quest for citations? an analysis of and commentary on the trend toward multiple authorship. *Journal of Social Work Education*, 35(2):203–213, 1999.

[6] Edgar J Manton and Donald E English. The trend toward multiple authorship in business journals. *Journal of Education for Business*, 82(3):164–168, 2007.

[7] Xavier Anguera, Simon Bozonnet, Nicholas Evans, Corinne Fredouille, Gerald Friedland, and Oriol Vinyals. Speaker diarization: A review of recent research. *IEEE Transactions on audio, speech, and language processing*, 20(2):356–370, 2012.

[8] Estathios Stamatatos, Michael Tschnuggnall, Ben Verhoeven, Walter Daelemans, Gunther Specht, Benno Stein, and Michael Potthast. Clustering by authorship within and across documents. In *Working Notes Papers of the CLEF 2016 Evaluation Labs. CEUR Workshop Proceedings/Balog, Krisztian [edit.]; et al.*, pages 691–715, 2016.

[9] Mirco Kocher and Jacques Savoy. Evaluation of text representation schemes and distance measures for authorship linking. *Digital Scholarship in the Humanities*, 34(1):189–207, 2019.

[10] Moshe Koppel and Jonathan Schler. Authorship verification as a one-class classification problem. In *Proceedings of the twenty-first international conference on Machine learning*, page 62, 2004.

[11] Eva Zangerle, Maximilian Mayerl, Günther Specht, Martin Potthast, and Benno Stein. Overview of the style change detection task at pan 2020. CLEF, 2020.

[12] Dimitrina Zlatkova, Daniel Kopev, Kristiyan Mitov, Atanas Atanasov, Momchil Hardalov, Ivan Koychev, and Preslav Nakov. An ensemble-rich multi-aspect approach for robust style change detection. *CLEF 2018 Working Nots of CLEF*, 2018.

[13] Kamil Safin and Aleksandr Ogaltsov. Detecting a change of style using text statistics. *Working Notes of CLEF*, 2018.

[14] Jamal Ahmad Khan. A model for style change detection at a glance. *Training*, 593(293):113, 1981.

[15] Marjan Hosseinia and Arjun Mukherjee. Experiments with neural networks for small and large scale authorship verification. *arXiv preprint arXiv:1803.06456*, 2018.

[16] Nils Schaetti. Character-based convolutional neural network and resnet18 for twitter authorprofiling. In *Proceedings of the Ninth International Conference of the CLEF Association (CLEF 2018), Avignon, France*, pages 10–14, 2018.

[17] Sukanya Nath. Style change detection by threshold based and window merge clustering methods. In *CLEF (Working Notes)*, 2019.

[18] Chaoyuan Zuo, Yu Zhao, and Ritwik Banerjee. Style change detection with feed-forward neural networks. In *CLEF (Working Notes)*, 2019.

[19] Daniel Castro-Castro, Carlos Alberto Rodríguez-Lozada, and Rafael Muñoz. Mixed style feature representation and b-maximal clustering for style change detection. In *CLEF (Working Notes)*, 2020.

[20] Aarish Iyer and Soroush Vosoughi. Style change detection using bert. In *CLEF*, 2020.

[21] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[22] Robert Deibel and Denise Löfflad. Style change detection on real-world data using lstm-powered attribution algorithm. In *CLEF*, 2021.

[23] Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. Learning word vectors for 157 languages. *arXiv preprint arXiv:1802.06893*, 2018.

[24] Sukanya Nath. Style change detection using siamese neural networks. In *CLEF*, 2021.

[25] Rhia Singh, Janith Weerasinghe, and Rachel Greenstadt. Writing style change detection on multi-author documents. In *CLEF*, 2021.

[26] Zhijie Zhang, Zhongyuan Han, Leilei Kong, Xiaogang Miao, Zeyang Peng, Jieming Zeng, Haojie Cao, Jinxi Zhang, Ziwei Xiao, and Xuemei Peng. Style change detection based on writing style similarity. In *CLEF*, 2021.

[27] Eivind Strøm. Multi-label style change detection by solving a binary classification problem. In *CLEF*, 2021.

[28] Hans Van Halteren, Harald Baayen, Fiona Tweedie, Marco Haverkort, and Anneke Neijt. New machine learning methods demonstrate the existence of a human stylome. *Journal of Quantitative Linguistics*, 12(1):65–77, 2005.

[29] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556, 2009.

[30] Thomas C Mendehall. A mechanical solution of a literary problem. *The Popular Science Monthly*, 60, 1901.

[31] Carrington B Williams. Mendenhall's studies of word-length distribution in the works of shakespeare and bacon. *Biometrika*, 62(1):207–212, 1975.

[32] G Udny Yule. On sentence-length as a statistical characteristic of style in prose: With application to two cases of disputed authorship. *Biometrika*, 30(3/4):363–390, 1939.

[33] David Mitchell. Type-token models: a comparative study. *Journal of Quantitative Linguistics*, 22(1):1–21, 2015.

[34] C Udny Yule. *The statistical study of literary vocabulary*. Cambridge University Press, 2014.

[35] Edward H Simpson. Measurement of diversity. *nature*, 163(4148):688–688, 1949.

[36] David L Hoover. Another perspective on vocabulary richness. *Computers and the Humanities*, 37(2):151–178, 2003.

[37] John Burrows. 'delta': a measure of stylistic difference and a guide to likely authorship. *Literary and linguistic computing*, 17(3):267–287, 2002.

[38] Rosa María Coyotl-Morales, Luis Villaseñor-Pineda, Manuel Montes-y Gómez, and Paolo Rosso. Authorship attribution using word sequences. In *Iberoamerican Congress on Pattern Recognition*, pages 844–853. Springer, 2006.

[39] Bradley Kjell, W Addison Woods, and Ophir Frieder. Discrimination of authorship using visualization. *Information processing & management*, 30(1):141–150, 1994.

[40] John Houvardas and Efstathios Stamatatos. N-gram feature selection for authorship identification. In *International conference on artificial intelligence: Methodology, systems, and applications*, pages 77–86. Springer, 2006.

[41] Efstathios Stamatatos, Nikos Fakotakis, and Georgios Kokkinakis. Computer-based authorship attribution without lexical measures. *Computers and the Humanities*, 35(2):193–214, 2001.

[42] Moshe Koppel and Jonathan Schler. Exploiting stylistic idiosyncrasies for authorship attribution. In *Proceedings of IJCAI'03 Workshop on Computational Approaches to Style Analysis and Synthesis*, volume 69, pages 72–80, 2003.

[43] Philip M McCarthy, Gwyneth A Lewis, David F Dufty, and Danielle S McNamara. Analyzing writing styles with coh-metrix. In *FLAIRS Conference*, pages 764–769, 2006.

[44] Dasha Bogdanova and Angeliki Lazaridou. Cross-language authorship attribution. In *LREC*, pages 2015–2020. Citeseer, 2014.

[45] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey. *arXiv preprint arXiv:1901.09069*, 2019.

[46] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. In *Advances in Neural Information Processing Systems*, pages 932–938, 2001.

[47] Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In *International workshop on artificial intelligence and statistics*, pages 246–252. PMLR, 2005.

[48] Yoshua Bengio and Jean-Sébastien Senécal. Quick training of probabilistic neural nets by importance sampling. In *International Workshop on Artificial Intelligence and Statistics*, pages 17–24. PMLR, 2003.

[49] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*, 2013.

[50] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[51] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[52] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[53] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[54] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[55] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[57] Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*, 2019.

[58] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[59] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[60] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[61] Li Deng and Yang Liu. *Deep learning in natural language processing*. Springer, 2018.

[62] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*, 2019.

[63] Sarthak Jain and Byron C Wallace. Attention is not explanation. *arXiv preprint arXiv:1902.10186*, 2019.

[64] Kumpati S Narendra and Mandayam AL Thathachar. Learning automata-a survey. *IEEE Transactions on systems, man, and cybernetics*, (4):323–334, 1974.

[65] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[66] Bernard Widrow and Marcian E Hoff. Associative storage and retrieval of digital information in networks of adaptive "neurons". In *Biological Prototypes and Synthetic Systems*, pages 160–160. Springer, 1962.

[67] Urszula Stańczyk and Krzysztof A Cyran. Machine learning approach to authorship attribution of literary texts. *International journal of applied mathematics and informatics*, 1(4):151–158, 2007.

[68] Suvad Selman. Distinction of the authors of texts using multilayered feedforward neural networks. *Southeast Europe Journal of Soft Computing*, 1(1), 2012.

[69] Zhenhao Ge, Yufang Sun, and Mark Smith. Authorship attribution using a neural network language model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[70] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[71] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[72] Yoon Kim. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014.

[73] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *arXiv preprint arXiv:1509.01626*, 2015.

[74] Prasha Shrestha, Sebastian Sierra, Fabio A González, 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.

[75] 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.

[76] Sebastian Ruder, Parsa Ghaffari, and John G Breslin. Character-level and multi-channel convolutional neural networks for large-scale authorship attribution. *arXiv preprint arXiv:1609.06686*, 2016.

[77] Julian Hitschler, Esther Van Den Berg, and Ines Rehbein. Authorship attribution with convolutional neural networks and pos-eliding. In *Proceedings of the Workshop on Stylistic Variation*, pages 53–58, 2017.

[78] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[79] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

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

[81] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[82] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*, 2017.

[83] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *ArXiv preprint arXiv:1412.3555*, 2014.

[84] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. In *International conference on machine learning*, pages 2342–2350. PMLR, 2015.

[85] Shriya TP Gupta, Jajati Keshari Sahoo, and Rajendra Kumar Roul. Authorship identification using recurrent neural networks. In *Proceedings of the 2019 3rd International Conference on Information System and Data Mining*, pages 133–137, 2019.

[86] Zhi Liu. Uci machine learning repository. `ttps://archive.ics.uci.edu/ml/datasets/Reuter_50_50`, 2017.

[87] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015.

[88] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE, 2005.

[89] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 148–157, 2016.

[90] Tharindu Ranasinghe, Constantin Orăsan, and Ruslan Mitkov. Semantic textual similarity with siamese neural networks. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1004–1011, 2019.

[91] Author Obfuscation. Overview of pan 2018. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 9th International Conference of the CLEF Association, CLEF 2018, Avignon, France, September 10-14, 2018, Proceedings*, volume 11018, page 267. Springer, 2018.

[92] Eva Zangerle, Michael Tschuggnall, Günther Specht, Martin Potthast, and Benno Stein. Overview of the Style Change Detection Task at PAN 2019. In Linda Cappellato, Nicola Ferro, David E. Losada, and Henning Müller, editors, *CLEF 2019 Labs and Workshops, Notebook Papers*. CEUR-WS.org, September 2019.

[93] Jaime S Cardoso and Ricardo Sousa. Measuring the performance of ordinal classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(08):1173–1195, 2011.

[94] Jonas Mueller and Aditya Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[95] Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. A sick cure for the evaluation of compositional distributional semantic models. In *Lrec*, pages 216–223. Reykjavik, 2014.

[96] Benedikt Boenninghoff, Robert M Nickel, Steffen Zeiler, and Dorothea Kolossa. Similarity learning for authorship verification in social media. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2457–2461. IEEE, 2019.

[97] Chakaveh Saedi and Mark Dras. Siamese networks for large-scale author identification. *ArXiv preprint arXiv:1912.10616*, 2019.

[98] Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. Effects of age and gender on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 6, pages 199–205, 2006.

[99] Moshe Koppel, Jonathan Schler, Shlomo Argamon, and Eran Messeri. Authorship attribution with thousands of candidate authors. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 659–660, 2006.

[100] Min Yang and Kam-Pui Chow. Authorship attribution for forensic investigation with thousands of authors. In *IFIP International Information Security Conference*, pages 339–350. Springer, 2014.

[101] Eva Zangerle, Maximilian Mayerl, Martin Potthast, and Benno Stein. Overview of the style change detection task at pan. 2021.

[102] Georgios Barlas and Efstathios Stamatatos. A transfer learning approach to cross-domain authorship attribution. *Evolving Systems*, pages 1–19, 2021.

[103] Jacob Tyo, Bhuwan Dhingra, and Zachary Lipton. Siamese bert for authorship verification. In *CLEF*, 2021.

[104] Xiaogang Miao, Haoliang Qi, Zhijie Zhang, Guiyuan Cao, Ruilan Lin, and Wenbin Lin. Dual neural network classification based on bert feature extraction for authorship verification. In *CLEF*, 2021.

[105] Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen. A survey of the state of explainable ai for natural language processing. *arXiv preprint arXiv:2010.00711*, 2020.

[106] Efstathios Stamatatos. Intrinsic plagiarism detection using character n-gram profiles. *Threshold*, 2(1,500), 2009.

[107] Harald Baayen, Hans van Halteren, Anneke Neijt, and Fiona Tweedie. An experiment in authorship attribution. In *6th JADT*, volume 1, pages 69–75. Citeseer, 2002.

[108] Moshe Koppel, Jonathan Schler, and Shlomo Argamon. Computational methods in authorship attribution. *Journal of the American Society for information Science and Technology*, 60(1):9–26, 2009.

[109] Benedikt Boenninghoff, Steffen Hessler, Dorothea Kolossa, and Robert M Nickel. Explainable authorship verification in social media via attention-based similarity learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 36–45. IEEE, 2019.

[110] Jacques Savoy. *Machine Learning Methods for Stylometry: Authorship Attribution and Author Profiling*. Springer International Publishing, 2020.

[111] Mirco Kocher and Jacques Savoy. Distance measures in author profiling. *Information Processing & Management*, 53(5):1103–1119, 2017.

[112] James W. Pennebaker. *The secret life of pronouns: What our words say about us*. Bloomsbury Press, New York, 2011.

[113] Martin Potthast, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. A decade of shared tasks in digital text forensics at pan. In *European Conference on Information Retrieval*, pages 291–300. Springer, 2019.

[114] Jacques Savoy. Comparative evaluation of term selection functions for authorship attribution. *Digital Scholarship in the Humanities*, 30(2):246–261, 2013.

[115] Jacques Savoy. Analysis of the style and the rhetoric of the 2016 us presidential primaries. *Digital Scholarship in the Humanities*, 33(1):143–159, 2017.

[116] Jacques Savoy. Is Starnone really the author behind Ferrante? *Digital Scholarship in the Humanities*, 33(4):902–918, 2018.

[117] Walter Daelemans, Mike Kestemont, Enrique Manjavancas, Martin Potthast, Francisco Rangel, Paolo Rosso, Günther Specht, Efstathios Stamatatos, Benno Stein, Michael Tschuggnall, Matti Wiegmann, and Eva Zangerle. Overview of PAN 2019: Author Profiling, Celebrity Profiling, Cross-domain Authorship Attribution and Style Change Detection. In Fabio Crestani, Martin Braschler, Jacques Savoy, Andreas Rauber, Henning Müller, David E. Losada, Gundula Heinatz, Linda Cappellato, and Nicola Ferro, editors, *Proceedings of the Tenth International Conference of the CLEF Association (CLEF 2019)*. Springer, September 2019.

[118] Martin Potthast, Tim Gollub, Matti Wiegmann, and Benno Stein. TIRA Integrated Research Architecture. In Nicola Ferro and Carol Peters, editors, *Information Retrieval Evaluation in a Changing World - Lessons Learned from 20 Years of CLEF*. Springer, 2019.

[119] David I Holmes. The evolution of stylometry in humanities scholarship. *Literary and linguistic computing*, 13(3):111–117, 1998.

[120] Benno Stein, Nedim Lipka, and Peter Prettenhofer. Intrinsic plagiarism analysis. *Language Resources and Evaluation*, 45(1):63–82, 2011.

[121] Muna AlSallal, Rahat Iqbal, Vasile Palade, Saad Amin, and Victor Chang. An integrated approach for intrinsic plagiarism detection. *Future Generation Computer Systems*, 96:700–712, 2019.

[122] Efstathios Stamatatos. Plagiarism detection based on structural information. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1221–1230, 2011.

[123] Nils Schaetti. *An Empirical Comparison of Recurrent Neural Network Models on Authorship Analysis Tasks*. PhD thesis, Université de Neuchâtel, 2020.

[124] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.

[125] Harold Jeffreys. *Theory of Probability,*. OUP Oxford, 1948.

[126] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

[127] Yoshua Bengio. *Learning deep architectures for AI*. Now Publishers Inc, 2009.