University of Cincinnati

Date: 2/24/2016

I, Jagadeesh Patchala, hereby submit this original work as part of the requirements for the degree of Doctor of Philosophy in Computer Science & Engineering.

It is entitled:
Data Mining Algorithms for Discovering Patterns in Text Collections

Student's name: Jagadeesh Patchala

This work and its defense approved by:

Committee chair: Raj Bhatnagar, Ph.D.
Committee member: Nan Niu, Ph.D.
Committee member: Yizong Cheng, Ph.D.
Committee member: Anil Jegga, D.V.M., M.Res.
Committee member: Ali Minai, Ph.D.
Committee member: Marepalli Rao, Ph.D.

19567

Last Printed: 3/17/2016 Document Of Defense Form
Data Mining Algorithms for Discovering Patterns in Text Collections

Author: Jagadeesh Patchala

Supervisor: Dr. Raj Bhatnagar

A dissertation submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Computer Science and Engineering in the Department of Electrical Engineering and Computing Systems of the College of Engineering and Applied Science of the University of Cincinnati

Committee:
Dr. Raj Bhatnagar, Chair
Dr. Anil Jegga
Dr. Ali A. Minai
Dr. Marepalli Rao
Dr. Yizong Cheng
Dr. Nan Niu
UNIVERSITY OF CINCINNATI

Abstract

Department of Electrical Engineering and Computing Systems

Doctor of Philosophy

Data Mining Algorithms for Discovering Patterns in Text Collections

by Jagadeesh Patchala
Discovering meta-information from collections of text documents is an important research area due to increasing demands for automated analysis of large text collections. This analysis process usually involves first structuring the unstructured text data and then deriving useful patterns from this structured extracts of the text data. This process involves contributions from multiple fields such as data mining, artificial intelligence, statistics, databases, and linguistics. There are several focus areas within this research domain, each pursuing a different objective. Some of these focus areas relate to information retrieval, document clustering, document classification, web mining, information extraction, and natural language processing. In this dissertation, we have developed several novel methodologies to uncover the hidden information in text document collections and have tried to solve some specific pattern discovery tasks.

The first and second problems we have addressed relate to predicting the author of a text document. We have studied the use of grammatical features for authorship attribution under various conditions. We have proposed new features derived from parse trees of text data and show that these features are effective for classification. We have also proposed a framework that identifies the author based on the consensus agreement from multiple feature types.

The third problem we have addressed relates to discovery of potential drug-repurposing candidates by uncovering hidden thematic structures within medical literature. We have proposed here a Latent Dirichlet Allocation (LDA) based framework to identify drug-repurposing candidates using the Unified Medical Language System (UMLS) concepts that appear in the disease and drug related articles available on MEDLINE.

The fourth problem that we have addressed focuses on identifying shared concepts across different document collections. Here we represent a collection of text documents by a binary concept matrix and seek to discover document-concept bi-clusters. Our main contribution is that instead of finding bi-clusters with all '1' entries in the table we attempt to allow some '0's in our bi-clusters. Such bi-clusters with dense one’s (called relaxed bi-clusters) have the potential to reveal interesting associations that may not be captured by the bi-clusters containing strictly all '1's.

The last problem we have addressed efficiently identifies pairs of relaxed bi-clusters
(relaxed 3-clusters) that have a high overlap along the shared dimension, from two binary datasets that share a common domain over a dimension. By identifying these relaxed 3-clusters one can understand the underlying connections between domain ‘1’ and domain ‘3’ mediated by their shared domain ‘2’. An example is a relaxed 3-cluster generated from two binary datasets, a dataset of documents vs. authors and a dataset of documents vs. words. Such a relaxed 3-cluster provides information about the documents an author or group of authors has written and their relatively large shared set of words occurring in those documents. All the existing algorithms that discover 3-clusters require each bi-cluster in 3-cluster to be of all one’s. We have relaxed this requirement on each bi-cluster and find relaxed bi-clusters (bi-cluster with dense ‘1’\(^s\)) that have a corresponding matching relaxed bi-cluster in the 2\(^{nd}\) dataset. Discovering relaxed bi-clusters and 3-clusters are very valuable in the fields of text mining and bio-informatics.
Acknowledgements

First and foremost, I would like to thank my adviser Dr. Raj Bhatnagar for his continuous support throughout my doctoral study. He gave me complete freedom of exploring the field of my interest and guided me to define the research problems. His immense knowledge on a variety of subjects helped me a lot in exploring research areas that were new to me. He is a very understanding person and gave me a lot of confidence in facing problems both in my research and in my real life. It was my pleasure to work with such a noble and insightful person.

I would like to offer my sincere thanks to Dr. Anil Jegga. He is very knowledgeable in both fields of Biology and Data Mining and introduced me to bioinformatics research area. He is a very good resourceful person and it is my true honor to have worked with him. I see him as a role model to follow in future.

I would like to offer my special thanks to other members of my dissertation committee - Dr. Ali Minai, Dr. Marepalli Rao, Dr. Yizong Cheng, and Dr. Nan Niu for evaluating my work and for offering their valuable suggestions.

I would like to thank my fellow lab mates- Divya Sardana and Vineet Joshi for their constant encouragement, moral support and providing me their valuable time to discuss various ideas.

I would like to express my great appreciation to Chris Amann and Emily Kregor of Graduate School - IT division, University of Cincinnati for providing me Graduate Assistantship - which helped me to survive financially without any hurdles. They are also very understanding and generous enough and gave me a flexible work schedule.

Finally, I wish to acknowledge the moral and financial support provided by my wife, parents, and my extended family.
# Contents

Abstract iii

Acknowledgements vi

1 Introduction 1

1.1 Authorship Attribution of Email Messages Using Parse - Tree Features . 2
1.2 Consensus Among Multiple Features . . . . . . . . . . . . . . . . . . . . . 2
1.3 Concept Modeling Based Drug Re-purposing . . . . . . . . . . . . . . . . 3
1.4 Bi-clustering Based Shared Concept Finding in Documents . . . . . . . . 4
1.5 Learning 3-Clusters from Pairs of Related Datasets . . . . . . . . . . . . . 4
1.6 Outline . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

2 Authorship Attribution of Email Messages using Parse - Tree Features 6

2.1 Background and Literature . . . . . . . . . . . . . . . . . . . . . . . . . . 7
2.2 Approach . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
2.2.1 Closed candidate author set . . . . . . . . . . . . . . . . . . . . . . . 12
2.2.2 Non closed candidate author set . . . . . . . . . . . . . . . . . . . . 13
2.3 Data . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
2.4 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
2.4.1 Comparison of feature types: . . . . . . . . . . . . . . . . . . . . . . 17
2.4.2 Non closed author set: . . . . . . . . . . . . . . . . . . . . . . . . . . 17
2.4.3 Robustness analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
2.5 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19

3 Authorship Attribution By Consensus Among Multiple Features 20

3.1 Background and Literature . . . . . . . . . . . . . . . . . . . . . . . . . . 23
3.2 Approach . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26
3.3 Data . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33
3.3.1 News Articles’ Dataset . . . . . . . . . . . . . . . . . . . . . . . . . . 33
3.3.2 Reuter 50_50 Dataset . . . . . . . . . . . . . . . . . . . . . . . . . . . 34
3.3.3 PAN 2014 English Essays Dataset . . . . . . . . . . . . . . . . . . . . 34
3.3.4 Blogs Dataset . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34
3.4 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
3.4.1 Closed author set using only one feature type . . . . . . . . . . . . 35
3.4.2 Semi-closed author set using only one feature type . . . . . . . . . 39
3.4.3 Closed author set using all the six feature types . . . . . . . . . . . 39
3.4.4 Robustness Analysis: . . . . . . . . . . . . . . . . . . . . . . . . . . . . 40
3.4.5 Effect of candidate author set size: . . . . . . . . . . . . . . . . . . . . 41
# List of Figures

2.1 Sample message from training emails .......................... 9  
2.2 Parse tree generated by Stanford parser for email text in Figure 2.1. 10  
2.3 Sample CFG Grammar productions. ............................... 10  
2.4 Frequencies of productions in an author template. 10  
2.5 Steps to generate the author templates .......................... 11  
2.6 Precision, recall, and accuracy values for different z-score values. 17  
2.7 Accuracy values vs number of features for various feature types. 18  
2.8 z-score plot of KL divergence for texts T1,T2, T3, and T4 . 18  
3.1 Height two, height three and height four sub-trees of a parse tree. 21  
3.2 An example of Dempster’s rule. ................................. 22  
3.3 Relative frequencies of the top 1,000 height two sub-trees in a document. 28  
3.4 Steps to generate the features ................................. 29  
3.5 Steps for author attribution using DS Theory ........................ 33  
3.6 Accuracy values for closed-author set using one feature type. 36  
3.7 Effect of feature set size. ................................. 37  
3.8 Effect of training data size. ........................................ 38  
3.9 Precision and recall values for different candidate author set sizes. 42  
3.10 Precision and recall values for different z-score cutoff values. 42  
3.11 Average z-score of the highest plausibility values. 43  
4.1 Schematic representation of overall workflow. .......................... 50  
4.2 ROC curve for validation data. ................................. 54  
4.3 Stacked bar chart showing the top five topic proportions found in modafinil (drug) and its two indications (bipolar disorder and narcolepsy) and ten random disease sets. 55  
5.1 Synthetic Datasets .................................................. 65  
5.2 Shuffled Datasets .................................................. 65  
5.3 Relaxed bi-clusters identified by proposed algorithm 66  
5.4 Bi-clusters identified in dataset 2 using BicBin algorithm 66  
5.5 Average density of 1’s per bi-cluster for different ‘α’ and ‘β’ values. 67  
5.6 Time taken for different proportion of 1’s in 200 X 200 dataset 68  
5.7 Execution time for different datasets. α, β = 0.90 70  
6.1 Sample Datasets and relaxed 3-clusters identified 72  
6.2 Relaxed 3-clusters algorithm outline 80  
6.3 Expansion process ................................................. 83  
6.4 Original and Shuffled synthetic Datasets .......................... 85
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.5</td>
<td>3-clusters identified in Figure. 6.4</td>
<td>85</td>
</tr>
<tr>
<td>6.6</td>
<td>Seeds-synthetic datasets and 3-clusters identified.</td>
<td>86</td>
</tr>
<tr>
<td>6.7</td>
<td>A 3-cluster from the Plants-States-Weather Dataset’s</td>
<td>87</td>
</tr>
<tr>
<td>6.8</td>
<td>Execution times on various datasets for different parameter values.</td>
<td>90</td>
</tr>
</tbody>
</table>
List of Tables

2.1 Accuracy values for dataset 1 .......................... 15
2.2 Accuracy values for dataset 2 .......................... 15
2.3 Accuracy values for different author sizes ............. 15
2.4 Accuracy values with z-score inference rule for dataset 1 16
2.5 Accuracy values with z-score inference rule for dataset 2 17

3.1 Mass assignment process using eight authors and δ value 0.05 32
3.2 Precision and Recall for semi-closed authorship using individual features ................................. 39
3.3 Precision and Recall values using all feature types (Case-3) ...................................................... 40
3.4 Precision, Recall values with slight changes to test documents ..................................................... 41

4.1 Examples of drugs with multiple indications .................. 54
4.2 Ranked drugs for six rare diseases .......................... 55

5.1 Outline of the results achieved by our algorithm on different datasets. ................................. 69
5.2 UMLS concepts that co-occur in abstracts for Parkinson Disease .............................................. 69
5.3 UMLS concepts that co-occur in abstracts for Restless leg syndrome ........................................ 69
5.4 Size, density variation in newsgroup dataset. ................................................................. 70

6.1 Sample 3-clusters and their size variations with decrease in alpha, beta values ............................. 87
6.2 BBC Dataset - Characteristics ................................. 88
6.3 Average bi-cluster size, One’s % in a 3-cluster of Plants-Weather Datasets ................................. 88
6.4 Average bi-cluster size, One’s % in a 3-cluster, BBC Dataset - Business,Technology ................ 89
6.5 Average bi-cluster size, One’s % in a 3-cluster. BBC Dataset - Entertainment, Sports ................. 89
6.6 Average bi-cluster size (rows, columns) in BBC Dataset - Entertainment, Sports ........................ 89
I dedicate this dissertation to my family and friends for their unconditional love and their partnership in my success
Chapter 1

Introduction

With increasing popularity of digital infrastructure for data storage there is an immense growth in the amount of digital text data. Discovering meta information from large collections of text documents has become a research problem worth significant rewards. The benefits of potential solutions go well beyond simple searches for lists of documents containing some terms. The research promises innovations that can help us understand and make use of information spread across large document repositories. Development of new algorithms to automatically detect the author, topic, or thematic structure of a document has a wide range of applications. Some of the areas in which these results may be used include plagiarism detection, automatic document classification, shared-concept discovery across multiple documents, and email filtering. Each of these applications relies on appropriate representation of text corpora and robust algorithms for pattern discovery.

The main aim of the research proposed in this dissertation is to represent information embedded in text documents in various suitable structures and formats, and to develop algorithms for a variety of knowledge discovery tasks from the text document collections. Especially, we examine the methods to automatically classify the documents based on their authorship, and cluster the documents based on the concepts shared by the documents.
1.1 Authorship Attribution of Email Messages Using Parse-Tree Features

With the rapid growth of the Internet, Email has evolved as a dominant form of communication among people, governments, and organizations. Along with its growth the misuse of email has become a common scenario. One example of misuse is when a sender tries to hide his identity by modifying the header information of the email message. In this case it is very hard to identify the true author. There is thus a need to develop efficient automatic methods that can analyze the content/writing style in the email message and categorize or identify the author of these messages.

Traditional authorship attribution studies using large text documents consider syntactic features as reliable features. Till now there is not much research that studied the usefulness of these features on short text particularly on email genre. In this study we use the frequencies of production rules obtained from parse trees and show that these features are as reliable as character n-grams, and Part of Speech(POS) n-grams even in identifying the authors of emails.

1.2 Consensus Among Multiple Features

Conventional authorship analysis algorithms use wide variety of lexical, syntactic and semantic features. The main problem with most of the lexical features is that they are not stable for an author and moreover it is easy for anyone to mimic these features in their writings and misguide the authorship analysis system. The success of any authorship attribution system relies on the stability of the features for each author. That is, the selected features must distinguish authors from one another and must be consistent across different writings of the same author. However, when we consider a large number of authors for authorship attribution, a single feature type may not be sufficient to capture the uniqueness of the authors’ writings. So, it is advisable to discover more features from different types of features, and combine them to discriminate the authors. We propose three novel feature types- Frequencies of height two (Context Free
Chapter 1. Introduction

Grammar Productions, height three, and height four sub-trees derived from the parse trees of sentences. Along with these three new feature types, we use function words, char trigrams, and POS trigrams as other significant feature types. Most of the current research in authorship analysis uses different types of features and combines all the feature types into one long feature vector, and these feature vectors representing the documents are fed into the machine learning algorithms. We have designed a different approach for combining these feature types, i.e., a consensus approach for combining them. We treat each feature type as a different source and arrive at a degree of belief using the Dempster-Shafer framework on multiple feature types.

We have compared the results with the new proposed feature types with the results we got by using other prominent features like function words, char trigrams, and POS trigrams. Compared to other popular features we show that frequencies of two level and three level sub-trees perform better. We analyzed the performance of our approach with the prominent approaches used in authorship attribution systems, and using multiple real world datasets we show that our approach gave better accuracy compared to other approaches.

1.3 Concept Modeling Based Drug Re-purposing

Topic modeling identifies the main themes that are present in large collection of documents. Once these themes are discovered we can categorize the documents according to the themes. I.e., we can classify the documents that share similar concepts into one group. We apply this idea to a collection of disease/drug articles published in medical literature and propose that diseases/drugs that are grouped together share similar concepts (in this case, genes, gene pathways, or any other biomedical concepts) and are highly likely to be connected to each other.

Most of the repositioned drugs are identified in the form of unexpected findings in late phase clinical trials. The main reason for this is that, there will be a shared gene or a biomedical concept that plays an important role in the original and new indicated disease. We apply LDA based topic model against a set of articles for rare diseases and
approved drugs and propose that the drugs that are grouped along with a rare disease are the potential candidate drugs for that rare disease.

1.4 Bi-clustering Based Shared Concept Finding in Documents

Bi-clustering is used in the domain of text mining since a long time and is popularly known as co-clustering. The text documents are represented in a ‘bag of words’ format as a matrix D. In this matrix, rows represent the documents, columns represent the words from all the documents, where the element $D_{ij}$ denotes the occurrence of word $j$ in document $i$. A bi-clustering algorithm is applied to find the links between a subset of documents (rows) to a subset of words (columns) such that each document will have all the attributes of the selected words set. In other words, the bi-clustering algorithm is used to find the subset of documents that are characterized by a group of selected words. In some applications, instead of finding a bi-cluster that has a strict linkage for all the instances and attributes, we may have to identify bi-clusters that have some diluted linkage between the selected subset of rows and columns. That is in this version of bi-cluster, we allow some of the entries in the bi-cluster to be zero. This new form of bi-cluster will still be able to capture the strict linkages between the rows and columns and additionally help to predict the linkages that may be missing in the former notion of bi-cluster. Along with the fully connected components, we will find the objects that are connected with most of the components. We implemented a search-based algorithm to find these kinds of bi-clusters and our proposed algorithm allows finding the overlapping bi-clusters, which is important in many applications.

1.5 Learning 3-Clusters from Pairs of Related Datasets

The idea of 3-clusters has been increasingly applied in data mining scenarios where two data sets need to be analyzed simultaneously. One can obtain new useful information by simultaneously processing the two datasets that share attributes in at least one of the dimensions. For instance, 3-clusters may be derived from two distinct matrices that
represents the text documents in two different categories. A 3-cluster here, will identify the subset of documents from each of the dataset respectively, corresponding to the same subset of words. In this chapter we extend our concept of finding bi-clusters to 3-clusters and propose an algorithm to find 3-clusters from two binary datasets. The 3-clustering approaches proposed previously are capable of finding only strict linkages, whereas, we are trying to find 3-clusters, which contain a relaxed bi-cluster with in each data context and sharing enough data objects.

1.6 Outline

Remainder of this dissertation consists of the following chapters. Chapter 2 discusses the idea of using grammatical style in author attribution of email messages. Chapter 3 discusses the work related to Authorship Analysis, where we have developed a framework to identify the author of the document based on a consensus agreement with multiple features. Chapter 4 discusses the methodology we have developed to identify the potential drugs for rare diseases using the medical literature. Chapter 5 discusses the research we have done to identify the relaxed bi-clusters from binary datasets. Chapter 6 describes the extension of our bi-clustering approach to finding 3-clusters in binary datasets, and the investigation we did to study its effectiveness under different constraints.
Chapter 2

Authorship Attribution of Email Messages using Parse - Tree Features

As Internet became popular, individuals, companies and organizations has started to incorporate Internet technologies in to their day-to-day activities. With this, Email has became a major form of communication between individuals, companies and governments as it is more convenient, faster and economical compared to traditional mail. A report from The Radicati group, Inc [56], which is a technology market research firm reported that there will be a growth rate of 6% per year in the total number of emails accounts from 2013 to 2017. The average number of Emails sent/received per day is 196.4 Billion in 2015 and has a growth rate of 3% per year till 2017.

Along with the growth of the Email, it’s misuse has also became a common scenario. In 2015, the share of spam volume to the total email volume is near 56%. A news article in [33] report that there is a huge drop of 56% in spam volume in 2014 and 2015. However, they convey that the maliciousness of emails has increased. Examples of such Emails include ‘Asking for sensitive information’, ‘Keeping a malicious link in place of a authentic link’, ‘Sending emails threatening or abusing the user’, ‘Sending computer virus as attachment to emails’, ‘Sending emails from an unauthorized access of an email account of different person’.

Even though all the emails have a header information that contains the details of the sender and the path it has traversed. It is very easy to forge the header information in an email and hide the true identity of the sender. Sometimes a malicious person may
send emails as a true sender by unauthorized access of true sender’s email account and computer. Because of these problems, we might run into situations where the header information might be totally disregarded in establishing the true identity of the author. In these cases, we will be only left with the email text to establish the identity of the author.

In this chapter, we propose a template based authorship attribution framework using the syntactic features extracted from the email documents. We construct the author profile templates using the parse tree feature frequencies of an author’s training email documents. We then construct a similar template from the test email document and compare this template against the author templates to find the best match.

2.1 Background and Literature

Most of the research in authorship analysis deal with larger size text documents such as books, journal publications, literary works, and newspaper articles. Researchers have proposed a wide variety of features to tackle the traditional authorship problem. However, most of the features that work well in traditional authorship analysis may not work well in case of email documents as the size of emails are short and has a different text composition compared to documents from other genres [70]. Some of the lexical features such as average length of the paragraphs or sentences, average number of words in a sentence are not meaningful in case of emails. Also, the frequencies of N-grams is a popular feature set in assigning authorship for large text documents whereas, for emails - these N-grams frequencies vary a lot from one email message to the other.

The work presented in [70], [52] use various content specific features like greetings and farewell messages, presence or absence of a signature, contact information for email authorship attribution. They [52] had used support vector machine (SVM) based classifier and showed that combining the results from various machine learning classifiers increased the accuracy of the system. The problem with these kind of content specific features is that not all authors use them and among the author’s who use them, they
won’t use them on each and every message. Moreover, it is very easy for anyone to copy the signatures of other person. The work presented in [83] used frequencies of function words, special characters, vocabulary richness. They reported an accuracy of 75% (for 3 authors) with function words and an accuracy of 81% considering all the feature types they mentioned.

The authors in [67] had used various lexical and syntactic features such as Hapax legomenon’s, ratio of short words (size of words <3) to total vocabulary, number of digits, sentence length, function words, etc. However, the work in [16] analyzed these lexical features on email documents and reported that these lexical features are not very useful in discriminating the email authors. But, they showed that the function words have these discriminating capability. [35] has proposed a clustering based authorship attribution framework using lexical and function word frequencies. Using the enron email dataset and an author set size of five, they reported an accuracy of 90%, for ten authors - 80%, and for twenty authors - 74% [34].

Syntactic features represent characteristics of messages in terms of writing styles and language proficiencies of authors and are therefore more reliable for classification. Even though people analyzed various types of traditional features on email authorship attribution, no one had studied the usefulness of context free grammar (CFG) production frequencies on emails. We believe that these grammatical features are innate to the writing capabilities of an author and will be embedded in one’s writing without their consciousness. Moreover, our intuition is that the topic, context, and the length of the message don’t have a significant impact on the grammatical style of the author. So, here we examine the effectiveness of these CFG production frequencies on short text like emails with an author set size as large as 30 authors.

2.2 Approach

Given the training data for all authors, first we identify the training documents corresponding to each author and combine all the training documents that belong to each
“The project coordinators believe they can reach a solution with Accenture in which Enron pays no additional fees but the project is completed in a slightly scaled back form. This allows Accenture to have a completed project and no more cash goes out the door.”

Figure 2.1: Sample message from training emails

author. At this point, we will have one large training text file for each author. Using the Stanford statistical parser [39], we parse the text and generate the parse tree’s for each of the sentences in the text. After generating the parse trees, we will remove the terminal nodes in each of these parse trees, and create the grammar productions from these modified parse trees. By removing the terminal nodes, we are eliminating the productions that lead to the words used by an author, and are capturing only the grammatical style of the author rather than the content/context. Now, we calculate the frequencies of productions for each author and consider the frequent 250 productions for each author. It might be possible that some of the productions will occur only in few authors but not in all. So, we do a union of the frequent 250 productions of each author and create the template using the combined frequent production set. For example, Let’s say we have fifteen authors in our candidate author set, then we will create fifteen such templates and each template consists of the relative frequencies of the productions occurred in that author’s text. The parse tree generated for the sample sentence in Figure 2.1 is shown in Figure 2.2.

The sample CFG grammar productions identified in Figure 2.2 are shown in Figure 2.3. Figure 2.3 shows a few productions in which the non-terminal $S$ represents a sentence, $NP$ represents a noun phrase, $VP$ represents a verb phrase, etc. Figure 2.4 shows the frequencies of productions in a typical author template. The main steps followed to create the author profiles are summarized in Figure 2.5.

Given the test email whose authorship is to be determined, we parse the text and generate the signature template using the same set of productions used in creating the author templates. These author templates and test signature template are nothing but probability distributions and by using divergence as a metric we calculate the similarity between author templates and test signature.
Chapter 2. Authorship Attribution of Email Messages using Parse - Tree Features

**Figure 2.2:** Parse tree generated by Stanford parser for email text in Figure 2.1.

```
S -> NP VP
NP -> DT NN
NP -> PRP
NP -> NN
VP -> VB NP
VP -> VB PP
PP -> PRP NP
```

**Figure 2.3:** Sample CFG Grammar productions.

**Figure 2.4:** Frequencies of productions in an author template.
1. Combine all the training email messages written by an author into a single text file.
2. Parse the sentences in this text file and generate the parse trees for sentences.
3. Count the occurrence of each grammar production in the parse trees of the author text.
4. Take the frequent 250 productions (in terms of their counts of occurrence) for each author.
5. Construct a union set of the frequent 250 productions taken from each author. These productions constitute the common template structure. Each template for an author contains the relative frequencies for these productions found in the author’s training text.
6. Normalize the frequencies in a template so that they all add up to 1.0.

**Figure 2.5: Steps to generate the author templates**

Divergence measures the dissimilarity between two probability distributions. The probability distributions $P$ and $Q$ are similar when they have a very low divergence value and are dissimilar when the divergence value between them is very high. We had used three different divergence measures namely KL-divergence [44], J-divergence [36], and AGM divergence [66]. Each of these measures capture difference between two probability distributions from different perspectives. The main reason to consider different divergence measures is to show the robustness of production frequencies in attributing the author of emails.

**Kullback-Leibler (KL) divergence** [44] measures the dissimilarity between two probability distributions and represents the information lost when $Q$ is used to represent $P$. It is calculated as:

$$D_{KL}(P \parallel Q) = \sum_i P_i \log_2(P_i/Q_i)$$

**Jeffrey’s Kullback-Leibler (J) divergence** [36] is a variant of the KL divergence and is calculated as follows:

$$D_J(P \parallel Q) = \sum_i (P_i) \log_2(P_i/Q_i) + \sum_i (Q_i) \log_2(Q_i/P_i)$$
Arithmetic and Geometric Mean (AGM) divergence [66] is a measure based on the arithmetic and geometric mean inequality of two distributions. It is calculated as:

\[
D_{AGM}(P \parallel Q) = \sum_i \left[ \frac{(P_i + Q_i)}{2} \log_2 \left( \frac{(P_i + Q_i)}{(2\sqrt{P_i Q_i})} \right) \right]
\]

Here, \( P \) denotes the probability distribution of production frequencies in a test document’s signature and \( Q \) denotes the probability distribution of productions in an author’s template.

There might be some production’s which don’t occur in the author’s templates or in the test document signature. If we leave these values as zero, especially when these zero value’s occur in the denominator, we might run into non-computable situations while calculating the divergence values. Moreover, an occurrence of zero in one distribution and non-zero in the other distribution conveys that the two distributions are significantly different. To capture this notion, we replace the zero values in author and test document templates with very small values, say 0.05, 0.1, and 0.25. Then, we calculate the average of the divergence between the author template and signature template for the three cases of the zeros being replaced by three different values. In this way, we calculate the divergence values between the test signature and each of the author templates.

After calculating the divergence values we have to devise an classification criteria to do the authorship assignment. Here, we might encounter two types of situations.

**2.2.1 Closed candidate author set**

In this scenario, we are sure that the test document is written by some one from the candidate author set. So we simply assign the authorship to the author whose template has the smallest divergence value.
2.2.2 Non closed candidate author set

In this case, the test document may or may not be written by the authors in the candidate author set. It may not be appropriate to simply assign the authorship to the author with smallest divergence value in this scenario. Because, when the test document is written by some one outside of the candidate author set, then the divergence values for all the authors will have similar divergence values with not much significant difference between them. where as when the test document is written by some author in the candidate author set, it is expected that the true author will have a significantly small divergence value compared to rest of the authors.

So, to capture this notion, we identify the smallest divergence value and calculate the z-score of this lowest divergence value with respect to the other divergence values. This score will tell us the number of standard deviation values by which the smallest divergence is away from the mean of all the divergence values. We call this z-score value as Minimum separation ($MinSep$) value and calculate it as follows:

$$MinSep(D) = [\text{Min}(D) - \mu(D)]/\sigma(D)$$  \hspace{1cm} (2.1)

Here, D is the set of divergence values of a test document against all the author templates.

Now, the decision rule for authorship attribution is as follows. The author with the smallest divergence score should have atleast a $MinSep$ score $\leq -2$ to assign the authorship of the test document to him. In non closed author set scenario's, this rule will aid us to identify the scenarios where no single author stands out as the potential author and we can then say that the author is from outside of the author set.
Chapter 2. Authorship Attribution of Email Messages using Parse - Tree Features

2.3 Data

The data used in this research has been taken from the publicly available Enron email dataset. The raw Enron email dataset contains 200,399 messages belonging to 158 different authors, with an average of 757 messages per author. We have selected 2 distinct sets of 15 authors from these 158 authors and have extracted a random set of 120 email messages sent by each author. We have considered only the "sent items" folder from every author and have excluded the quoted text from within each message. An email is included in the training or test data only if it has a reasonable length, i.e., we considered only those emails that have at least five sentences. We selected 60 email messages out of the 120 messages for each author as part of the training set. From the remaining 60 email messages for each author, we extracted four different test sets of randomly selected three messages, six messages, and ten messages, thus creating a total of twelve test sets for each author.

2.4 Results

To evaluate our proposed system, we used accuracy as a metric to identify the prediction performance. Here, we define accuracy as follows:

\[
\text{Accuracy} = \frac{\text{Number of emails whose author is correctly identified}}{\text{Total number of emails tested for the authors}}.
\]

Apart from the three divergence measure based classifiers, we also use two more machine learning algorithms namely Naïve Bayes and Support Vector Machines (SVM). We used the Matlab’s Naïve Bayes classifier with multinomial distribution and LIBSVM [11] one-vs-one multi-class classifier with an non-linear kernel (RBF) for SVM based classification. The reason to use these two instance based classifiers is to compare the performance of our proposed divergence based approach with these classifiers.

We have two datasets containing fifteen authors in each dataset and have three test cases. In the first test case, we have twelve email messages for each author (four sets of
Table 2.1: Accuracy values for dataset 1

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>KL divergence</th>
<th>J divergence</th>
<th>AGM divergence</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test run 1</td>
<td>91.0</td>
<td>79.09</td>
<td>84.32</td>
<td>81.73</td>
<td>89.03</td>
</tr>
<tr>
<td>Test run 2</td>
<td>82.73</td>
<td>74.56</td>
<td>78.64</td>
<td>78.29</td>
<td>84.31</td>
</tr>
<tr>
<td>Test run 3</td>
<td>84.60</td>
<td>78.23</td>
<td>76.87</td>
<td>75.62</td>
<td>83.49</td>
</tr>
<tr>
<td>Overall</td>
<td>86.11</td>
<td>77.29</td>
<td>79.94</td>
<td>78.54</td>
<td>85.61</td>
</tr>
</tbody>
</table>

Table 2.2: Accuracy values for dataset 2

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>KL divergence</th>
<th>J divergence</th>
<th>AGM divergence</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test run 1</td>
<td>83.28</td>
<td>72.31</td>
<td>54.32</td>
<td>63.18</td>
<td>85.27</td>
</tr>
<tr>
<td>Test run 2</td>
<td>77.35</td>
<td>66.72</td>
<td>57.64</td>
<td>69.24</td>
<td>70.11</td>
</tr>
<tr>
<td>Test run 3</td>
<td>81.23</td>
<td>71.36</td>
<td>49.87</td>
<td>74.74</td>
<td>78.42</td>
</tr>
<tr>
<td>Overall</td>
<td>80.62</td>
<td>70.13</td>
<td>53.94</td>
<td>69.05</td>
<td>77.93</td>
</tr>
</tbody>
</table>

three emails per author), twenty four emails per author in test case two (four sets of six emails), and forty emails in test case three (four sets of ten emails).

Given the test email, we calculate the divergence values between the test signature and all the fifteen author’s templates in the candidate author set and assign authorship to the author with the smallest divergence value. These results are reported in Table 2.1 and Table 2.2. From the Table 2.1, and 2.2, we can observe that the overall accuracy is near 70 - 80% in both the datasets irrespective of the classifier used. With this, we can say that frequencies of grammar productions have enough information to discriminate the authors of the emails.

To analyze the effect of candidate author set size, we considered the candidate set sizes of 5 - 30 in multiples of five and randomly selected the authors. For each candidate author set size, we repeated the process of random selection of author’s ten times and the average accuracy values are reported in Table 2.3. As expected, we can see that the accuracy decreases slowly with the increase in the author set size.

Table 2.3: Accuracy values for different author sizes

<table>
<thead>
<tr>
<th>Author set size</th>
<th>KL divergence</th>
<th>J divergence</th>
<th>AGM divergence</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>93.8±1.4</td>
<td>89.7±2.2</td>
<td>91.0±2.3</td>
</tr>
<tr>
<td>10</td>
<td>90.3±2.2</td>
<td>88.2±1.7</td>
<td>88.6±1.8</td>
</tr>
<tr>
<td>15</td>
<td>88.7±1.1</td>
<td>82.5±1.3</td>
<td>83.2±2.7</td>
</tr>
<tr>
<td>20</td>
<td>82.9±2.8</td>
<td>73.9±2.2</td>
<td>75.3±3.2</td>
</tr>
<tr>
<td>25</td>
<td>78.2±3.2</td>
<td>68.1±1.3</td>
<td>71.8±1.8</td>
</tr>
<tr>
<td>30</td>
<td>74.6</td>
<td>65.8</td>
<td>69.1</td>
</tr>
</tbody>
</table>
Table 2.4: Accuracy values with z-score inference rule for dataset 1

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>KL divergence</th>
<th>J divergence</th>
<th>AGM divergence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test run 1</td>
<td>86.67</td>
<td>71.33</td>
<td>77.78</td>
<td></td>
</tr>
<tr>
<td>Test run 2</td>
<td>75.56</td>
<td>57.78</td>
<td>62.25</td>
<td></td>
</tr>
<tr>
<td>Test run 3</td>
<td>77.63</td>
<td>63.89</td>
<td>68.89</td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>79.95</td>
<td>64.33</td>
<td>69.64</td>
<td></td>
</tr>
</tbody>
</table>

From the results, we can observe that KL-divergence gave better performance compared to the other two divergence measures. The reason for this is as follows: Generally, the test message signature will have more number of zero frequencies (replaced with small values) compared to the author templates. While calculating KL-divergence we consider message signature as $Q$ distribution and when these zero frequency values occur in the denominator, they will give a large divergence value. However, it is expected that such zero frequencies mismatches will be minimal giving a low divergence value for true author of the test document. For the other two divergence measures, we consider message profile $Q$ in both numerator and denominator and average the divergence values. So, the effect of zero frequencies is not as pronounced as in KL-divergence.

The accuracy values obtained after applying our ‘$MinSep \leq -2$’ rule are shown in Table 2.4 and 2.5. By comparing these results with the results from Table’s. 2.1 & 2.2, we can notice that our inference rule will decrease the accuracy. However, this rule will give us a higher degree of confidence that the author assigned by our approach will be the true author of the test email.

We analyzed the effect of the $MinSep$ cutoff value by calculating the accuracy, precision and recall values for different values. In this analysis, we had used a set of thirty authors and sixty emails for training and sixty emails for testing. The results are shown in Figure 2.6. We can observe that there is not much change in the precision and recall values for $MinSep$ values 0, -0.5, and -1. But there is a significant increase in precision for $MinSep$ values -1.5 and -2. This tells us that most of the misclassified messages don’t have a significantly low divergence value compared to rest of the authors.
Table 2.5: Accuracy values with z-score inference rule for dataset 2

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>KL divergence</th>
<th>J divergence</th>
<th>AGM divergence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test run 1</td>
<td>74.67</td>
<td>65.17</td>
<td>47.31</td>
</tr>
<tr>
<td>Test run 2</td>
<td>63.92</td>
<td>52.65</td>
<td>50.72</td>
</tr>
<tr>
<td>Test run 3</td>
<td>72.35</td>
<td>61.94</td>
<td>43.74</td>
</tr>
<tr>
<td>Overall</td>
<td>70.31</td>
<td>59.92</td>
<td>47.25</td>
</tr>
</tbody>
</table>

Figure 2.6: Precision, recall, and accuracy values for different z-score values.

2.4.1 Comparison of feature types:

We use the frequencies of character trigrams and function words (most frequent words) as two baselines to compare our proposed grammar production frequencies. The accuracy values for these three feature types with various feature set sizes are shown in Figure 2.7. Here, for this analysis, we considered thirty authors and each author has sixty emails for training and sixty emails for testing. We can see that grammar productions gave better accuracy compared to the other two feature types. The reason for this could be that emails are short and the function words and character trigrams vary a lot from one message to other message of an author depending on the context/topic. Where as, the grammatical style remains same across the emails written by an author and this style is well captured by the CFG grammar productions.

2.4.2 Non closed author set:

To analyze the effect of our inference rule in non-closed author set scenario, we considered hundred emails whose authors are not in the candidate author set pool (thirty
FIGURE 2.7: Accuracy values vs number of features for various feature types.

FIGURE 2.8: z-score plot of KL divergence for texts T1, T2, T3, and T4 authors) and tried to assign the authorship for these hundred emails. With a MinSep score of -2, we are able to reject seventy nine emails with out assigning an author. This means that our rule correctly identified (79% of times) that the original authors of the test emails are outside of the candidate author set. For the remaining twenty one emails, we assigned the authorship to one of the candidate authors. The average MinSep score for these twenty one emails is '-2.29'.
2.4.3 Robustness analysis

To analyze the robustness of our proposed approach, we had done some tests to understand how a small change to the original test email text will change the results. For this we created four different text messages as follows. $T_1$ is the original message of author nine with at least ten sentences. $T_2$ is created by removing half of the sentences from $T_1$. $T_3$ is obtained by adding a few (one - three) sentences of author three to $T_2$ where as $T_4$ is obtained by adding at least four sentences of author three to $T_2$. Here, $T_1$ and $T_2$ are text’s written entirely by author nine and $T_3$ has majority of text written by author nine. $T_4$ consists equal length text from author’s nine and three. We repeated this text generation process ten times and the average $\text{MinSep}$ scores are shown in Figure 2.8. From Figure 2.8 we can notice that there is not much change in the $\text{MinSep}$ values for text’s $T_1$, $T_2$, $T_3$. The $\text{MinSep}$ values vary a lot for text $T_4$ compared to other texts and $T_4$ has a low divergence score for author three. However, it’s $\text{MinSep}$ score is not significant and we reject to assign authorship. From this, we can conclude that CFG grammar productions are robust to minor changes in author’s text.

2.5 Conclusion

Most of the research in email authorship analysis use various lexical, stylistic, and content specific features. However, like in traditional authorship analysis there are not many studies that studied the usefulness of syntactic features like grammar productions in email authorship attribution. Most of the research is limited to the use of function words. Here, in this research, we show that CFG grammar production features are also very effective in determining the author of the emails. We had proposed a template matching based authorship framework using the relative frequencies of grammar productions. We had also devised a new decision rule that attributes the author to messages when the true author of the message is outside the candidate author set. With experiments, we also demonstrated that the proposed approach is robust to minor changes in test document and perform better compared to other feature type’s like character trigrams and function words.
Chapter 3

Authorship Attribution By Consensus Among Multiple Features

The problem of authorship attribution on text documents was there from the beginning of the literature. With the increase in the number of text documents/textual data in electronic format, it has become a non-trivial problem with new applications in the area of identifying the authors of emails [70], on-line messages [82], anonymous posts in blogs, articles, and in authenticating the documents exchanged in e-commerce. The authorship attribution problem is sub-divided into two sub-problems.

- Closed class authorship attribution problem.
- Open class authorship attribution problem.

In closed class authorship attribution problem, we will have a set of potential authors and someone from the given author set will be the original author of the document that is to be analyzed. In the open class authorship problem, we will have a suspected set of authors, but the document to be analyzed may or may not be written by the authors in the suspected set. If someone outside of the author set writes the document, then the authorship attribution system must be able to reject all the authors from the suspected author set.
Even though people started research on this problem long ago [76], only from the last few years researchers are able to develop reliable, robust, and accurate solutions. Previous studies [4], [54], [40] have proposed a wide variety of features to tackle this problem. However, most of the studies [3], [21], [37] agree that the frequencies of function words and parts of speech (POS) tag n-grams are robust and perform well in solving this problem. The main use of the function words in one’s writing is to describe the grammatical relationships between the content words, and the POS tag n-grams capture the short term grammatical dependency. The success of these two feature types shows the usefulness of syntactic information in authorship attribution. With the development of the state-of-the-art parsing software, we are able to completely automate the process of extracting the syntactic style of the author. So, with the help of these tools we derive a new type of features that aid in solving this problem.

We consider the fact that each person will have his own writing style, which remains stable across his writings. We take advantage of this and try to use one’s syntactic style as a strong marker to differentiate one author from the other. To capture the syntactic style of the author with various levels of granularity we derive different levels of subtrees from the parse tree’s of author’s text and use them as features for authorship attribution. In this research we propose and discuss three different feature types of parse subtrees for authorship analysis - frequencies of height two, height three, and height four sub-trees derived from the parse trees of author’s texts. These features represent the inborn writing style of the author’s. Examples of various level sub-trees derived from a syntactic parse tree are shown in Figure 3.1.
Chapter 3. Authorship Attribution By Consensus Among Multiple Features

Figure 3.2: An example of Dempster’s rule.

<table>
<thead>
<tr>
<th></th>
<th>Height two sub-trees</th>
<th>Height three sub-trees</th>
<th>Height four sub-trees</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Author ‘a’</td>
<td>0.34</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>Author ‘b’</td>
<td>0.2</td>
<td>0.27</td>
<td>0.5</td>
</tr>
<tr>
<td>Author ‘c’</td>
<td>0.36</td>
<td>0.33</td>
<td>0.1</td>
</tr>
<tr>
<td>‘a’, ‘b’, ‘c’</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>

When we have a small set of authors, the uniqueness of an author can be easily identifiable, where as this uniqueness explicitly dilutes when the author set becomes large. So, we propose that it is highly desirable to include more features that belong to various feature types and combine the information from multiple features. So, in addition to the proposed three features, we add three more reliable feature types- frequencies of function words, character trigrams, and POS trigrams.

Each of these six feature types does not provide completely independent source of author discrimination information but each feature type captures some novel/additional information that is not captured by the other feature types. So, instead of combining them as independent source of evidence, we attempt to find only the consensus among their inferences. We use Dempster Shafer (DS) framework [17], [61] to find the consensus among each of these feature types and assign the authorship. We compare this approach with various other approaches and empirically show that the proposed approach is better than the others.

With the help of the example shown in Figure 3.2, we will explain the differences between DS based consensus approach, voting and the aggregate approaches. Here, we have three authors ‘a’, ‘b’, and ‘c’, and three feature types - height two, height three, and height four sub-trees. The scores assigned by each feature type are shown in their respective columns. When an author is assigned using the consensus approach, author ‘a’ will be selected as the true author. This is because the extent to which different features agree on an author is highest for author ‘a’ among the three authors. Author ‘b’
is selected as the true author when we follow the combined approach. In this case, we sum up the scores produced by each feature type and assign authorship to the author with highest score \((0.2 + 0.27 + 0.5)\). Here, for author ‘b’, only the height four subtrees provide high confidence (0.5), whereas the height two and height three subtrees have very less confidence in author ‘b’. If we consider voting, then author ‘c’ will be selected as true author, as two feature types (height two, height three subtrees) out of three have high confidence on author ‘c’. However, the height four subtrees has a very low confidence for author ‘c’.

3.1 Background and Literature

The first work on authorship analysis was performed in [50] to solve the disputed authorship of the federalist papers between three candidate authors. The authors analyzed the difference between the distributions of the function words among three authors’ writings. The author in [9] developed a set of 50 high frequency words and analyzed their frequencies for the same federalist papers. The work in [28] studied the use of shorter words, and words that start with vowels. Such word based methods needs a lot of effort to choose the most appropriate set of words that best differentiates the set of authors. Moreover, it is desirable to claim that a particular feature type is highly robust/reliable for authorship analysis, whereas it is not acceptable to claim that a particular feature is superior to others. For example, one can claim that frequencies of CFG productions are robust but making a claim about a particular CFG production is not acceptable. There are ten’s of features like average number of words per sentence [25], sentence length [76], word length [77], frequency distributions of word lengths and vocabulary richness features which ranges from a simple feature like the average number of distinct words against total words to a complex feature that measures how many times an author has used distinct synonym words instead of a particular word. Even though vocabulary richness is measured as a reliable method, it takes huge human effort to manually detect them and need more sophisticated tools to automatically capture them from text.
The problem with most of these features is that they are tested with only a limited author set size (typically less than five) and they lose their reliability once the author set size increases.

The work in [48] has used semantic features based on the wordnet. They analyzed the information regarding the synonyms and hyponyms of the words and applied latent semantic analysis to automatically detect the similarities between words. However, they didn’t provide any detailed description about the usefulness of the semantic information in their approach. Research in [62] has used the character n-grams extensively for authorship analysis, and showed that they are reliable markers even when the author set size is large. The main advantage of character n-grams is that they are simple to extract and are able to capture the variations in writing style, contextual information, and punctuation marks. One more advantage of character n-grams is that they are able to capture the strange punctuations and spelling errors. The drawback of these features is that if you choose a lower ‘n’ (n=1) they may not be able to capture all the information and as you increase the ‘n’ the number of features will explode. So, most studies had suggested using a value of ‘3’ especially for English. Also, these character n-grams are highly topic dependent and will not perform well when the author has written documents across multiple topics. I.e., when the topic of the text is different in training and test documents of an author.

The work by [3], [81] has used POS tag n-grams and [40] used syntactic errors as features. The idea in using syntactic features is that author’s tend to use the similar syntactic patterns subconsciously throughout their writings. The success of using function words for authorship analysis indicates the usefulness of syntactic information in authorship attribution and attracted researchers to derive more syntactic features. The work in [46] developed features that are extracted by partial parsing of text. Even though it requires highly robust and accurate tools to develop these syntactic features, the main advantage of using these features is their topic/context independence. I.e., the grammatical style of the author doesn’t change with the topic/context of text.

Research work by [70], [42] have used context/application specific features like greetings, farewell messages, signatures, font colors etc., while analyzing the authorship of
emails and blog posts. An author mostly changes these features frequently and more over it is easy for any other author to mimic these features in their writings and mis-guide the authorship attribution system that depends on these features.

Throughout the literature, it was the frequencies of function words, character n-gram frequencies, and syntactic features like POS tag n-grams and frequencies of rewrite rules derived from shallow parsing are the features that are identified as reliable in the field of authorship analysis.

There are two types of authorship attribution approaches that researchers have prominently pursued.

- Profile based methods.
- Instance based methods.

Most of the early research [54], [50], [78] in authorship attribution is based on the profile-based methodology. In this approach, we concatenate all the training text belonging to an author into one single text and extract the features from this single text. These features represent the signature of the author. For a given test document we extract the features and calculate the similarity of the test signature with each of the author signatures. The author who has a high similarity value for a test signature is assigned as the author of the test document.

With the development of sophisticated machine learning algorithms researchers have shifted their focus from using profile based methods to instance based approaches. In this approach, each training document is considered as a unit that contributes separately to author attribution model. Each training document is represented in a feature space and a classification algorithm is then trained on the available training data and the trained classifier acts as the author attribution model. The major requirement for the instance-based approaches is that each text document should be large enough so that the extracted features accurately capture the style in the documents. Researches have used support vector machines (SVMs) [70], [19], linear discriminant analysis [12], decision trees [82], neural networks [49], and Naïve Bayes classifier. Many studies
have preferred the use of Support Vector machines (SVMs) than other classifier models as SVM’s can handle high dimensional data.

However, till now there is no concrete study that compares the profile-based approaches with the instance based approaches. In other words, no one has studied the problem of when we should prefer the one to the other approach.

3.2 Approach

We built our authorship analysis system using profile-based methodology. If the author of the test document is outside the candidate author set, then it is very difficult for a classifier trained on the candidate author set to reject the authorship attribution of the test document. The classifier will find a closest match to the test document and assigns its author as the author of the test document, whereas if we follow the profile-based approach we can compare the distances between the test document and the author profiles and can apply the intuitive idea that the distance between the test document signature and the original author profile will be much smaller compared to the distances between the document signature and the remaining author profiles. If the author of the test document doesn’t belong to the candidate authors, then the distances for all the profiles will be similar. I.e., all the profiles will have very large distance values.

We considered this idea and built our authorship attribution system based on the profile based methodology. Our proposed system will have three main steps.

- Identification and extraction of features for each author, and for test documents.
- Calculate the similarity values between a test document signature and the author profiles.
- Combine the evidence from various feature types using Dempster Shafer theory and attribute the author.

The success of any authorship analysis system mainly relies on the type of features included in the system. In our research, our emphasis is on extracting the syntactic style
of the author. So, to capture this style at various levels of granularity we consider various feature types such as frequencies of two level, three level, and four level sub-trees derived from the parse trees of text in the documents. The fact that the documents written by some authors are well identified by a particular feature type compared to other authors indicates the need to include multiple feature types and arrive at an attribution based on all the features. So, in addition to above three features, we include three more feature types - frequencies of function words, character trigrams and POS trigrams as features.

As we don’t have manually annotated author’s documents to train the parser, we parse each of the author text using the statistical parser [39] trained on the general wall street journal corpus. Once the parse trees are generated, we extract all the terminal words whose parents belong to any of the category in the closed world class, as function words. After generating the frequencies of function words, we remove the terminal symbols from each of the parse trees and generate the two level, three level and four level sub-trees using the modified parse trees. Removing these terminal symbols from the parse tree will enable us to strictly capture the grammatically style of an author omitting the context/topic. Now, we generate the part-of-speech trigrams using the terminal nodes in the modified parse tree. The character trigrams are generated by directly scanning the text.

To create the author profiles we first combine all the documents written by an author into one single entity and parse the text from this entity. We generate the frequencies of function words, POS trigrams, two level sub-trees, three level sub-trees and four level sub-trees from the parsed text using the procedure described in the previous paragraph. Once the frequencies of the features are obtained, we construct the author profiles for each feature type by following the procedure described below.

The most important criteria in selecting the features are that the feature must be stable across multiple writings of an author and should significantly vary from one author to the other author. We sort the features according to their frequency of occurrence in the training text and select the features that are above some threshold ‘k’. Typically the selected value for ‘k’ will be the number of authors in the candidate author set. We
Chapter 3. Authorship Attribution By Consensus Among Multiple Features

Figure 3.3: Relative frequencies of the top 1,000 height two sub-trees in a document.

retain a few hundred (>15) features at this step and identify the most discriminating 'n' features by using the entropy values. We calculate the relative frequency of each feature for all the authors and calculate the entropy of each feature by using the relative frequencies across the authors. The formula to calculate the entropy is as follows:

\[
Entropy(X) = \sum_{i=1}^{N} P(X_i) \log_2 P(X_i)
\]  

(3.1)

Here, 'X' stands for an individual feature, and \( P(X_i) \) represents the relative frequency of the feature 'X' for author 'i'. If the frequency of a feature is same for every author, then the entropy will be high; else the entropy will be low implying that the feature is most discriminative and very particular to an author or to small group of authors. Now we sort the features according to the entropy values (in ascending order) and select the top 'n' features according to our feature set size. At this point we have eliminated the redundant features and have the top frequent and most discriminating 'n' features to construct the author template. A sample template generated for feature type 'height two sub-trees' is shown in Figure 3.3.

We repeat the above procedure for each of the feature type and generate the author templates. This entire process is shown in Figure 3.4.
Combine all the articles in training text set for each author.

- Now, for each author
  1. Generate the character trigrams from raw text and compute their frequencies.
  2. Tokenize the text into sentences.
  3. Parse the sentences using a parser and extract the POS trigrams, function words, CFG productions, three level sub-trees, and four level sub-trees and compute the frequencies for each feature type.
  4. Sort the features within each feature type according to their decreasing frequencies.
  5. Choose the top $n$ discriminating features using the entropy metric for each individual feature across authors.

**Figure 3.4: Steps to generate the features**

Now for each author we have six templates $T_1$ to $T_6$, with each template representing the probabilities of the features in that particular feature type. Given a test document, we generate the feature template for each of the six feature types and call them $S_1$, $S_2$, $S_3$, $S_4$, $S_5$, and $S_6$. We calculate the similarity between the test document signature templates and the six templates for each of the author using the Kullback-Leibler (KL) divergence metric. The KL-divergence for the $m^{th}$ feature set $D_{KL-m}$ is computed using the $m^{th}$ template $T_m$ for an author and $m^{th}$ signature $S_m$ for a document.

KL divergence can be interpreted as a measure of discrepancy between two probability distributions and is computed as:

$$D_{KL}(P \parallel Q) = \sum_i P_i \log_2(P_i/Q_i)$$ (3.2)

That is, it is a weighted sum of the logarithms of the ratios of individual relative frequencies ($P_i$ and $Q_i$ values) of productions in the two signatures being compared. In the above formula, ‘$P$’ represents the author template distribution and ‘$Q$’ represents the distribution of the test document signature template whose authorship needs to be determined. It is highly likely that some of the features in author or signature templates will have zero probability values. When we include the features with zero probabilities in calculating the KL-divergence, we may run into non-computable situations when these zero probability features occur in the denominators. If we exclude these kinds of features, then we will lose a lot of valuable information as the feature that have non-
zero probability value in one distribution and a zero probability value in other distribution provides a high confidence of belief that the two distributions are different. To capture this valuable information we include the features with zero probabilities and replace their values with a very small value and normalize the probability values in a template so that they all sum to 1. The lower the replaced value, the more it enhances the influence of the missing (zero probability) feature in calculating the divergence.

After calculating the divergence values, we devised four decision rules to address the following cases:

1. Closed authorship attribution problem using only one feature type.
2. Open authorship problem with one feature type.
4. Open authorship problem with multiple feature types.

**Case 1:**

In closed authorship problem we know that the test document will be written by some one from the given candidate author set. So, we calculate the divergence value $D(S_m, T_m(i))$ for all the authors and assign the authorship to the author who had a low divergence value. In $D(S_m, T_m(i))$, $S_m$ represents the test document signature for $m^{th}$ feature type and $T_m(i)$ represents the template for $m^{th}$ feature type for author ‘i’.

**Case 2:**

In open class authorship problem, the true author of the test document may or may not be present in the candidate author set. To assign an authorship in this case, we use the intuitive idea that if the true author is not present in the candidate set then the divergence values for all the authors will be large. There won’t be any significant difference between the divergence values of the authors. We capture this notion by calculating the z-score of the divergence values $D(S_m, T_m(i))$ for all the authors and assign an authorship to the author with smallest divergence value only when he has a z-score $\leq -2.0$. If the z-score is not less than -2.0, then we say that the true author is not in the candidate author set.
Case 3:

In this case, the true author will be from the candidate author set and we use information acquired from various feature types and assign an author based on the consensus among the feature types. These feature types are not independent and each of the feature type will capture some additional information that is not provided by the remaining feature types. We combine the six feature types using the DS theory and assign the author based on the combined belief and plausibility values.

Let us say there are ‘$k$’ authors in the pool of known authors and for the $m^{th}$ feature we compute the ‘$k$’ divergence values as $\text{Div}(m,i) = D(S_m, T_m(i))$, where $i$ takes the value from 1 to $k$. When repeated for each of the six feature types, the divergence values can be seen as six vectors, each of length ‘$k$’, containing ‘$k$’ divergence values $\text{Div}(m,1)$ through $\text{Div}(m,k)$. These divergence values are converted to mass assignments as per the requirement of DS theory. We do this conversion as follows. We first normalize the divergence values by scaling them as follows:

$$\text{NormDiv}(i,j) = \frac{(\max_j(\text{Div}(i,j)) - \text{Div}(i,j))}{(\max_j(\text{Div}(i,j)) - (\min_j(\text{Div}(i,j)))}.\quad (3.3)$$

Here, $i$ represents the author number, $j$ represents the feature type and $\text{Div}(i,j)$ denotes the divergence value of $i^{th}$ author for $j^{th}$ feature type.

After calculating the scaled values, the author whose template best matches the test signature template will have a scaled value of 1 and the most dissimilar author template will have a value of 0. The remaining authors will have values in between 0 and 1. If two or more authors have scaled values close to one other with a difference less than a threshold $\delta$, then we group those authors into one single set and assign a combined value which is the average of the values of authors in that group. Once these equally likely author groups are formed, we normalize the values again so that the scores sum to 1. We call these values as mass assignments for a particular feature type. For example, say, for eight authors ($a, b, c, d, e, f, g, h$) the scaled values be: $(1, 0.97, 0.94, 0.92, 0.87, 0.81, 0.73, 0)$. The mass assignment for these eight authors is shown in Table 3.1.
Table 3.1: Mass assignment process using eight authors and δ value 0.05

<table>
<thead>
<tr>
<th>Author groups</th>
<th>Scaled scores</th>
<th>Average mass values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;a, b&gt;</td>
<td>&lt;1, 0.97&gt;</td>
<td>0.985 0.157</td>
</tr>
<tr>
<td>&lt;a, b, c, d&gt;</td>
<td>&lt;1, 0.97, 0.94, 0.92&gt;</td>
<td>0.957 0.153</td>
</tr>
<tr>
<td>&lt;b, c, d&gt;</td>
<td>&lt;0.97, 0.94, 0.92&gt;</td>
<td>0.943 0.151</td>
</tr>
<tr>
<td>&lt;b, c, d, e&gt;</td>
<td>&lt;0.97, 0.94, 0.92, 0.87&gt;</td>
<td>0.925 0.148</td>
</tr>
<tr>
<td>&lt;d, e&gt;</td>
<td>&lt;0.92, 0.87&gt;</td>
<td>0.895 0.143</td>
</tr>
<tr>
<td>&lt;f&gt;</td>
<td>&lt;0.81&gt;</td>
<td>0.81 0.129</td>
</tr>
<tr>
<td>&lt;g&gt;</td>
<td>&lt;0.73&gt;</td>
<td>0.73 0.116</td>
</tr>
<tr>
<td>&lt;h&gt;</td>
<td>&lt;0&gt;</td>
<td>0 0</td>
</tr>
</tbody>
</table>

The mass assignments for each of the feature types are calculated and combined using Dempster’s rule of combination to obtain the consensus mass assignment. Given two mass assignments \( m_1, m_2 \), the combined mass assignment \( m_{12} \) is computed using Dempster’s rule of combination:

\[
m_{12}(A) = \frac{1}{1-K} \sum_{B \cap C = A \neq \emptyset} m_1(B)m_2(C), \quad K = \sum_{B \cap C = \emptyset} m_1(B)m_2(C) \tag{3.4}
\]

Here, \( K \) is the measure of the amount of conflict between two mass assignments, and \( A, B, C \) represent sets of authors.

Using this consensus mass assignment we compute the belief and plausibility values of each candidate author and select the author with the highest plausibility value as the true author of the test document.

The plausibility \( pl(A) \) is the sum of all the masses of sets ‘B’ that intersect the set of interest-‘A’ and is defined as follows:

\[
pl(A) = \sum_{B \mid B \cap A \neq \emptyset} m(B) \tag{3.5}
\]

Case 4:

For this situation we compute the plausibility values of authors as computed in case-3 and then determine if the author with the highest plausibility has a plausibility value greater than 0.5, and its z-score among all plausibility values of at least 2.0. That is,
1. For each feature type.
   • Calculate the divergence values between authors’ signatures and test profile.
   • Normalize the divergence values such that the new-scaled values lie between 0 and 1.
   • Create groups of authors based on the scaled values. The mass value for a group will be the average of the scaled values of authors in that group.
   • Normalize the mass values so that the sum of the mass values is 1.
2. Combine the mass functions from each feature type using dempster rule of combination.
3. Calculate the plausibility of each author from the combined mass function.
4. If the author with highest plausibility has a ‘z-score $>2$’ and ‘plausibility value $>0.5$’, attribute the author as the author of the test document.

FIGURE 3.5: Steps for author attribution using DS Theory

the selected author should have a good absolute plausibility value which is also significantly higher than the average plausibility values for the entire group of authors. If we don’t have an author that satisfies the above criteria, then we declare that the true author of the test document is not in the list of candidate authors. The algorithm we followed to do the authorship attribution using DS theory is presented in Figure 3.5.

3.3 Data

3.3.1 News Articles’ Dataset

We had selected ten authors seven from New York Times and three from Guardian and downloaded the articles from their respective website during the time frame 2010 -2012. From the topics ‘sports’, ‘education’, ‘immigration’, ‘elections’, ‘health’, and ‘politics’, each author has written columns in at least three topics. This dataset has a total of five hundred articles with fifty articles for each author. Out of fifty articles - thirty are for training and twenty for testing. On average each article in this dataset has eight hundred words per article.
Chapter 3. Authorship Attribution By Consensus Among Multiple Features

3.3.2 Reuter 50_50 Dataset

This dataset is a subset of the Reuter text collection dataset and is publicly available at the University of California, Irvine (UCI) machine-learning repository [7]. It has fifty authors and five thousand news articles published in Reuter’s. Each author has hundred documents, fifty for training and fifty for testing. On average, each article has five hundred words. Authors of [62], and [29] used this dataset to evaluate the use of character n-grams in authorship analysis.

3.3.3 PAN 2014 English Essays Dataset

The datasets created by the PAN [63] are some of the standard datasets in the field of authorship attribution/verification. So, we considered to use the PAN 2014 English essay dataset as one of our dataset. This particular dataset has two hundred problems and in each problem we will be given a small number (1 - 5) of training documents and a test document. We have to verify whether the test document is written by the same author or not. But the framework proposed in this chapter require training documents from other authors to assign the authorship. So, we added one more negative class by using the essays from the same Uppsala Student English (USE) corpus [6]. This PAN 2014 essays dataset is an subset of the larger USE corpus. Now, we can compare the divergence between the test document and two training classes and reject the authorship if the two divergence values are not significantly different. The average size of the document in this dataset is eight hundred and fifty words.

3.3.4 Blogs Dataset

To analyze the behavior of the proposed feature types and approach for large candidate author set size we plan to use the blogs dataset [41, 60] as one of our test datasets. This dataset has thousands of authors. Among these we selected the top hundred authors after sorting them by their blog posts. For each author we used ten posts for testing and the remaining posts as training data. On average we have nine hundred posts per author with an average post size of two hundred words.
3.4 Results

We measured the performance of our approach with accuracy, precision and recall values and compared the results with other approaches. In this study, we define the terms accuracy, precision and recall as follows.

Accuracy: “Number of documents whose author is correctly classified” / “Total number of documents tested”.

Precision: “Number of documents correctly assigned to the author” / “Total number of documents assigned to the author”.

Recall: “Number of documents correctly assigned to the author” / “Total number of documents written by the author”.

3.4.1 Closed author set using only one feature type

When we want to assign an author based on a particular feature type we use the methodology described in case 1 of our approach. Here, our aim is to compare and contrast the various feature types. In addition to our KL-divergence based approach, we used the method described in [41] which we call as ’Koppel 2011’ from now on, and two more instance based classifiers - Naïve Bayes, Support Vector Machine (SVM). The results for the two instance based classifiers are obtained by feeding them the same features of each feature types that are used to create the author templates of that feature type. Matlab’s Naïve Bayes (NB) classifier with multinomial distribution and LIBSVM one-vs-one classifier with non-linear kernel (RBF) are used for Naïve Bayes based classification and SVM. We choose these algorithms to compare our approach as these methods have reported better performance in authorship attribution research. The method ‘Koppel 2011’ needs two parameters: $k1$ - the number of sub feature sets to be selected, and $k2$ - the fraction of feature items to be included from the original feature set. Their experiments show that the optimal value for $k1$ and $k2$ to be 100 and 0.5. So, the same parameters are used in our test runs.
The feature set size is kept constant at 1500 for all the features except function words. We define function words to be the words that belong to the class of closed class words. Closed class words are the words that are used to explain the grammatical relationships between the words. e.g. prepositions, pronouns, conjunctions, auxiliary verbs etc. This results in few functions words and we limit them to 500. The accuracy values for each feature type using various approaches are shown in Figure 3.6.

We can notice that (Figure.3.6) the frequencies of height two, and three sub-trees, POS trigrams achieved better performance in news article, blogs and PAN datasets. Character trigrams did better in Reuter_50_50 dataset. Our reason for this behavior is that there is a lot of topic consistency for most of the authors in the Reuter_50_50 dataset and this makes it easy for character n-grams to identify the authors as character n-grams capture the theme/topic along with the grammatical style. The height four sub-trees do not perform well as features because the number of possible height four sub-trees explodes due to too many possible CFG production combinations, and even the most frequent height four sub-trees have a very low frequency count. With such low counts these features fail to efficiently distinguish the author among the others.
Figure 3.7: Effect of feature set size.

Effect of feature set size:

In this analysis, the training and test sets are kept constant and the number of features in a template are increased gradually in multiples of 500 from 500 - 3500 features. Authorship is assigned for each test document using the KL-divergence approach and accuracy values are reported in Figure 3.7. All the feature types attained good performance when one thousand - two thousand feature items are considered. They are stable in performance when a few more hundred feature items are added but the accuracy gradually decreased as we added more and more features. From these results, we can say that one can achieve reasonable performance by considering the most frequent and discriminating few hundred features. But, there is no overall consensus on the best size for the feature sets. This depends on the kind of the feature type selected and the training data available.
Effect of training data size:

In this analysis, the test documents are kept constant in each dataset and the number of training documents are increased gradually. The feature set size is kept constant at fifteen hundred for all the features except the function words (five hundred for function words). The accuracy values for different training data size’s are shown in Figure 3.8. Character trigrams and function words gave better performance when we have limited training data. The grammar based features (various level sub-tree frequencies, and POS trigrams) required more training data to give reasonable performance. However, given enough training data, all the grammar based features gave superior performance. With this we can conclude that grammar based features work well when we have large training data.
Chapter 3. Authorship Attribution By Consensus Among Multiple Features

Table 3.2: Precision and Recall for semi-closed authorship using individual features

<table>
<thead>
<tr>
<th></th>
<th>Char. trigrams</th>
<th>Func. words</th>
<th>POS 3grams</th>
<th>Height 2 sub-trees</th>
<th>Height 3 sub-trees</th>
<th>Height 4 sub-trees</th>
</tr>
</thead>
<tbody>
<tr>
<td>News article</td>
<td>prec. 69</td>
<td>prec. 51</td>
<td>prec. 71</td>
<td>prec. 77</td>
<td>prec. 74</td>
<td>prec. 59</td>
</tr>
<tr>
<td></td>
<td>rec. 72</td>
<td>rec. 47</td>
<td>rec. 56</td>
<td>rec. 77</td>
<td>rec. 74</td>
<td>rec. 59</td>
</tr>
<tr>
<td>Reuter</td>
<td>prec. 64</td>
<td>prec. 56</td>
<td>prec. 57</td>
<td>prec. 54</td>
<td>prec. 57</td>
<td>prec. 51</td>
</tr>
<tr>
<td></td>
<td>rec. 66</td>
<td>rec. 53</td>
<td>rec. 49</td>
<td>rec. 53</td>
<td>rec. 51</td>
<td>rec. 49</td>
</tr>
<tr>
<td>PAN 2014</td>
<td>prec. 54</td>
<td>prec. 46</td>
<td>prec. 46</td>
<td>prec. 48</td>
<td>prec. 43</td>
<td>prec. 47</td>
</tr>
<tr>
<td></td>
<td>rec. 50</td>
<td>rec. 37</td>
<td>rec. 39</td>
<td>rec. 44</td>
<td>rec. 40</td>
<td>rec. 36</td>
</tr>
<tr>
<td>Blogs</td>
<td>prec. 41</td>
<td>prec. 46</td>
<td>prec. 46</td>
<td>prec. 47</td>
<td>prec. 46</td>
<td>prec. 44</td>
</tr>
<tr>
<td></td>
<td>rec. 43</td>
<td>rec. 43</td>
<td>rec. 41</td>
<td>rec. 41</td>
<td>rec. 41</td>
<td>rec. 37</td>
</tr>
</tbody>
</table>

3.4.2 Semi-closed author set using only one feature type

Here, we follow the case-2 of the methodology section and do the authorship attribution. The feature set size is kept constant at 1500 features and z-score threshold is set at -1.5 in all the datasets except PAN dataset. PAN data set has only two classes and the author with lowest divergence score will always have a z-score of -1. So, we simply assign author to the class with lowest divergence score.

In addition to the test data considered in closed authorship case, we added some more test documents written by outside authors to each dataset. We added 200 documents from reuter_50_50 dataset to news article test set and vice versa. In blogs dataset, we added additional 500 blog posts to test data. The authors of these 500 blog posts are outside of the candidate pool. The precision and recall values are shown in Table 3.2. We can notice that grammar based features have high performance in news article and blogs dataset, where as character trigrams have marginally better performance in Reuter_50_50 and PAN dataset.

3.4.3 Closed author set using all the six feature types

The main aim of this analysis is to show that our proposed DS based combination of features work very well compared to other approaches. Here, we use the same test
Chapter 3. Authorship Attribution By Consensus Among Multiple Features

Table 3.3: Precision and Recall values using all feature types (Case-3)

<table>
<thead>
<tr>
<th></th>
<th>Divergence</th>
<th>Voting</th>
<th>Koppel 2011</th>
<th>SVM</th>
<th>DS combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>News Article</td>
<td>prec.</td>
<td>77±4.4</td>
<td>69±2.9</td>
<td>80±3.4</td>
<td>76±3.8</td>
</tr>
<tr>
<td></td>
<td>rec.</td>
<td>73±5.1</td>
<td>71±2.7</td>
<td>83±2.9</td>
<td>72±4.4</td>
</tr>
<tr>
<td>Reuter_50_50</td>
<td>prec.</td>
<td>56±3.9</td>
<td>64±3.2</td>
<td>64±4.2</td>
<td>63±3.5</td>
</tr>
<tr>
<td></td>
<td>rec.</td>
<td>61±4.8</td>
<td>59±3.6</td>
<td>62±2.6</td>
<td>64±3.1</td>
</tr>
<tr>
<td>PAN 2014</td>
<td>prec.</td>
<td>54±3.4</td>
<td>52±3.5</td>
<td>58±3.8</td>
<td>51±4.4</td>
</tr>
<tr>
<td></td>
<td>rec.</td>
<td>57±3.6</td>
<td>49±4.2</td>
<td>59±3.3</td>
<td>54±3.1</td>
</tr>
<tr>
<td>Blogs</td>
<td>prec.</td>
<td>58±4.1</td>
<td>54±3.8</td>
<td>62±3.9</td>
<td>57±3.6</td>
</tr>
<tr>
<td></td>
<td>rec.</td>
<td>61±3.2</td>
<td>52±3.4</td>
<td>63±3.4</td>
<td>54±3.9</td>
</tr>
</tbody>
</table>

Data as in case-1 (closed author with one feature type). However, instead of keeping the feature set size constant at 500, we consider the feature set sizes between 500 - 3000 in multiples of 500 and report the average precision and recall values. The threshold $\delta$ is kept constant at 0.05. The method labeled ‘Divergence’ is achieved by concatenating all the six feature templates of an author into one large template and then calculating the divergence scores. ‘Voting’ is performed by identifying the authors with lowest divergence score for each feature template and then assigning the authorship to the author selected by maximum number of feature templates. From the results in Table 3.3 DS combined approach has higher precision and recall in all the datasets. We believe that this be due to the grouping of authors who have significantly close values and then looking for consensus among multiple feature types.

Robustness Analysis:

Here, we want to analyze the robustness of the proposed consensus based approach to minor changes in test documents. For this reason, we added 50 words of text written by a random author to each of the test documents. Similarly we added 100 words to each test document and created an other test set. The $\delta$ value is kept constant at 0.05 and feature set size is varied from 500 - 3000 in multiples of 500. The precision and recall values are reported in Table 3.4. The PAN and Blogs dataset are sensitive to small changes in test documents. In Blogs dataset, the average size of an document is


Table 3.4: Precision, Recall values with slight changes to test documents

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Adding 50 words</th>
<th>Adding 100 words</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>News Article</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>precision</td>
<td>88±3.0</td>
<td>86±4.3</td>
<td>83±3.8</td>
</tr>
<tr>
<td>recall</td>
<td>82±2.1</td>
<td>80±3.9</td>
<td>77±5.7</td>
</tr>
<tr>
<td><strong>Reuter_50_50</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>precision</td>
<td>73±3.7</td>
<td>71±2.8</td>
<td>65±2.7</td>
</tr>
<tr>
<td>recall</td>
<td>66±5.0</td>
<td>64±3.7</td>
<td>60±3.6</td>
</tr>
<tr>
<td><strong>PAN 2014</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>precision</td>
<td>57±3.2</td>
<td>47±3.6</td>
<td>41±3.4</td>
</tr>
<tr>
<td>recall</td>
<td>58±3.5</td>
<td>50±4.3</td>
<td>39±4.1</td>
</tr>
<tr>
<td><strong>Blogs</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>precision</td>
<td>68±2.4</td>
<td>61±2.1</td>
<td>54±2.4</td>
</tr>
<tr>
<td>recall</td>
<td>65±1.1</td>
<td>58±2.4</td>
<td>51±2.3</td>
</tr>
</tbody>
</table>

small (200 words). Even though the document size is large in PAN dataset, even this dataset is sensitive to changes. We believe this might be because of limited training data (1-5 documents). There is no significant drop in performance in news article dataset when we added 50 or 100 random words. Here, the average document size is large (800 words) compared to other datasets. Similarly, Reuter dataset is robust when we added 50 words to test documents. But the performance decreased when we added 100 words. The average size of a document in Reuter dataset is 500 words.

**Effect of candidate author set size:**

In order to study the effect of candidate author set size on DS combined approach, we kept the feature set size constant at 1500 feature items and analyzed the precision and recall values by gradually increasing the author set size. For each author set size, we repeated the experiment ten times by randomly selecting the authors and average scores are reported in Figure 3.9. As expected, both the precision and recall values slowly decrease with an increase in the author set size.

3.4.4 **Semi-closed author set using all the six feature types**

To assign the authorship, we follow the case-4 of the methodology section and use the same test datasets mentioned in section 3.4.2. For this analysis, we considered a feature
Chapter 3. Authorship Attribution By Consensus Among Multiple Features

Figure 3.9: Precision and recall values for different candidate author set sizes.

Figure 3.10: Precision and recall values for different z-score cutoff values.
set size of 1500 and $\delta$ value of 0.05 in all the datasets. The precision and recall values obtained for various z-score cutoff values are shown in Figure 3.10. A low plausibility value for an author indicates that there is equally compelling evidence also available for other competing authors, and therefore no attribution is made. The inference rule $(\text{plausibility} > 0.5)$ rejects the authorship attribution of the documents that have low plausibility values. This rule gives a high degree of confidence and reliability that the author assigned will be the true author.

In PAN dataset we have only two classes, the author or non-author. Sets of possible authors and standard deviation of their plausibility values are not meaningful in this context and hence this test has not been performed for the PAN dataset. In other three datasets, recall does not drastically go down even after a z-score cutoff of 2.5. This is because when we have a large candidate set, the DS combined approach which look for consensus among feature types assigns a very low plausibility value for the authors who fail to mutually agree on authorship for selected feature types. This leaves out most of the authors with low plausibility values and only a few authors manage to retain significant plausibility values. So, the author with maximum plausibility value will have a high z-score value giving us a high recall. The averages of the maximum z-scores of plausibility values, that is the z-scores for the selected authors are shown in Figure 3.11. The higher the average z-score of the selected author, the better is the discrimination of the selected author from the rest of the authors in the pool.
3.5 Conclusion

In this research, there are two main contributions for the problem of authorship analysis. First, we propose a novel set of features—frequencies of height two, height three, and height four sub-trees as strong markers for authorship analysis. We showed the robustness of these features in solving the problem. We compared the performance of other prominent features with the proposed features and conclude that when abundant training data is available grammar based features achieve better performance than character trigrams and function words. The second contribution of this research is that we proposed a consensus-based methodology to combine the information acquired from various feature types and assign authorship. We showed the superiority of the proposed approach by comparing it with other approaches used in literature. There are quite a few other insights about author attribution systems that our results in the paper have brought out and they can be used to design effective author attribution systems.
Chapter 4
Concept Modeling based Drug Repurposing

Drug development is very time consuming and costly. This process may take around 10-20 years and will cost 500 million to 2 billion dollars [1]. In addition to increasing research costs, drug companies are under more financial stress due to the expiry of the existing patents. Moreover, if the new discovered drug consists of a new chemical compound, it may take a long time for Food and Drug administration (FDA) department to study the new chemical compound and approve the new drug. It is also reported in [24] that most of the drug development projects fail even before they are tested on humans. The drug making companies are constantly looking for new ways to reduce the costs and improve their productivity. So, because of the above factors drug companies have shifted their focus towards “drug repurposing” or “indication expansion”.

The main goal of the drug repurposing is to identify a new disease that can be potentially treated with the existing drug. As these existing drugs have already passed through the experimental and safety studies, drug companies can significantly reduce their time and cost. The main idea behind any of the drug repurposing study is that, if a drug frequently co-occurs with some biomedical or genomic concepts (For example, genes, pathways, proteins, chemical compounds, side effects, and symptoms) and most of these concepts also frequently co-occur with a disease/diseases in biomedical literature, it is highly likely that the diseases are new indications for the selected drug.

Probabilistic topic models [64] have recently attained a lot of attention and have been
used to address various issues (e.g., social network analysis, document classification, authorship analysis, gene-drug relationship extraction from literature, image classification, etc.). Topic models are algorithms that discover the hidden thematic structure in a large collection of documents. They automatically extract the set of thematic topics that describe a document collection, and assign the topics to each of the documents in the collection with a probability value. Each topic consists of a set of words. The documents that share most of the words will have a similar topic probability distribution compared to the other documents. One can use this idea and apply topic modeling to group the biomedical literature articles. The documents that share a high number of biomedical concepts will have similar topic probability distributions.

This chapter describes our study—using topic modeling to predict the potential drug-repurposing candidates. A collection of Medline articles is downloaded from PubMed for all the approved drugs mentioned in the Drugbank (www.drugbank.ca) and for a few rare diseases. Following this, a Linear Dirichlet Allocation (LDA) based Topic model is built based on the Unified Medical Language System (UMLS) concepts occurred in the articles. For each rare disease, probabilities of the topics are compared against the topics probabilities of all the approved drugs to identify the potential drug-repurposing candidates. The performance of the proposed framework is evaluated with manual literature-based validations. Additionally, we identified the pairs of original and repurposed indications (which were already discovered) for a selected drug and analyzed the performance of the proposed approach.

4.1 Background and Related Work

Many computational drug-repurposing strategies have been published in literature. The work in [20] has classified these computational strategies into two types, ‘drug based’ and ‘disease based’. In drug-based approaches the main emphasis is on chemical and pharmaceutical perspective where as in ‘disease based’ approaches the main focus is on the disease management, symptoms, and pathology of the disease. Drug
based approach is used when one wants to identify the potential repurposing candidates based on the precise chemical or pharmacological properties of a targeted drug or when a vast amount of chemical data is available for all the drugs. Disease-based approaches are used when there is no sufficient information on the pharmacology of a drug or when our main emphasis is on finding new drugs for a particular disease. The work in [22] has proposed network based approach that use various data sources related to genes, diseases, and proteins to discover the repurposing candidates. From information source perspective, these computational approaches are classified in to two categories.

- Ontology based approaches.
- Literature based approaches.

### 4.1.1 Ontology Based Approaches

These approaches depend on the already existing ontologies to find out the hidden relations between the drugs and the diseases. The work in [10] assume that similar side effects of any two drugs may be caused by some common targets and can be used to identify the potential new drug – target interactions. Using this idea, they generated a side-effect based drug-drug relation network from UMLS ontology by calculating the side effect likeliness between the phenotypes of drugs. Instead of phenotype database, the work in [74] use Pharmacogenomics Knowledgebase (PharmGKB) and side effect resource (SIDER) knowledge base, to construct a Naïve Bayes model in order to discover the potential new candidate drugs. The work in [14] use the drug-target information from DrugBank and constructed a bipartite network based on the extracted data. They used the network similarity to predict the new targets of drugs. The work in [45] used the same approach as [14], but added the drug chemical structure into consideration while calculating the similarity.
4.1.2 Literature Based Approaches

These approaches use the intuitive idea that if some biomedical concepts highly co-occur with a particular drug and these same concepts also frequently co-occur with some disease/diseases in the literature, then it is likely that the target drug can also be used to treat these diseases. The idea of discovering hidden relationships from biomedical literature was first developed by Swanson. He manually examined the literature and based on the biomedical concepts shared between the fish oil and Raynaud’s disease, he proposed that fish oil might be useful in the treatment of Raynaud’s disease. Based on this, he developed a computational model and called it as ‘ABC model’ [65]. The basic idea of this model is that if a biomedical concept ‘X’ connects to some concept ‘Y’ and this concept ‘Y’ in turn connects to some concept ‘Z’, then there exists some hidden relationship between concepts ‘X’ and ‘Y’. This ‘ABC model’ which is often called as literature–based discovery consists of three main steps:

- Mapping of the terms from literature to biomedical concepts.
- Finding the associations between the extracted concepts.
- Identify the new drugs/diseases for a target based on the similarity or strength of the concepts co-occurred.

Research in [75] used Medical Subject Headings (MeSH) to annotate the articles in literature and then applied $tf-idf$ weightings, followed by rule-based classifiers to identify the new indications. According to the work in [30], one of the main drawbacks of using MeSH terms to annotate the concepts is the insufficient information involving genes etc. The work in [73] used the Online Mendelian Inheritance in Man (OMIM) concepts to annotate the articles and used mutual information between the concepts as the link weight between the concepts. The work in [8] has constructed a probabilistic topic model based on the Medical Dictionary for Regulatory Activities (MedDRA) terms that appeared in the Boxed Warning, Warnings and Precautions, and Adverse Reactions sections of the drug labels. Their idea is that the drugs that have the similar side effects will be grouped together and are highly likely to be repurposed.
Here, we use UMLS concepts to map the text in the pubmed articles. We then use a probabilistic topic model to identify the hidden structure in the literature articles, followed by the similarity calculation of the topic probability distributions of various drugs.

4.2 Approach

Our entire experimental process consists of the following steps:

- Download the selected articles from MEDLINE corpus.
- Map the downloaded abstracts into biomedical concepts.
- Construct the Latent Dirichlet Allocation (LDA) model on the mapped concepts.
- Identify the potential candidate drugs by calculating the similarity between the topic probabilities of the selected disease and drugs.

The entire process we follow is shown in Figure 4.1.

4.2.1 MEDLINE Abstract Collection

We had selected six rare diseases and the list of all the FDA approved clinical drugs are collected from the DrugBank (www.drugbank.ca) website. There are a total of 1716 approved drugs in the drug bank list. In addition to these 1716 drugs, we added some more rare diseases we selected, and queried the PubMed using the NCBI E-Utilities feature [59]. For each of the queried term, the number of articles ranged in between 50 – 100,000. 10 drugs are removed from the selected drug list as each of these 10 drugs has fewer than 50 publications. After downloading the abstracts, we removed the abstracts that are cited in both drugs and diseases. We removed these abstracts in order to avoid over fitting of our model to any particular drug or a disease. Now for each of the diseases and drugs in our list, we randomly selected 500 abstracts for each, for our analysis. Thus our final dataset consists of 1694 drugs and 20 rare diseases, which resulted in 850,000 abstracts (nearly 1700 drugs/diseases * 500 abstracts). We
**Figure 4.1:** Schematic representation of overall workflow.
repeat the random selection of the 500 abstracts for each of the drugs/diseases for 10 times. That is we have 10 datasets with each dataset having 850K abstracts.

4.2.2 Concept Mapping

We used Metamap [5], a tool developed by National Library of Medicine (NLM), to map the text from the downloaded abstracts into UMLS concepts. While mapping the text we restricted the Metamap tool to include the concepts from five semantic types namely, Anatomy, Chemicals and Drugs, Disorders, Genes and Molecular Sequences, and Physiology. MetaMap provides a list of candidate mappings (along with their score) for each of the abstracts. We process these mappings produced by Metamap and exclude the concepts that have a score < 350. Previous studies have discovered that including concepts that have a low Metamap score will include a lot of redundant information. This is the reason for keeping a cutoff on the metamap score. Instead of the concept names, gene names, or protein names we use their respective ID’s to represent an abstract. This is to increase the specificity of our approach by avoiding redundancy. If we use the names instead of id’s the topic model may consider a single word entity into multiple ones. For example, the word “cystic fibrosis” may be split into two words “cystic” and “fibrosis”.

4.2.3 Topic Modelling

For topic modeling purpose, we have used the Linear Dirichlet Allocation (LDA) model, which is a generative probabilistic topic model. The reason for using LDA instead of a probabilistic latent semantic index (pLSI) is that, the pLSI is not a generative model and will not fully capture the dependencies between documents, topics and words. The intuitive idea behind the LDA model is that a document is generated with a mixture of topics and each word is selected with some probability given a document topic. Given a document ‘d’, θ(d)= P(t) stands for the multinomial distribution over topics and P(w|t) be the probability distribution over words ‘w’ given topic ‘t’. Then, the
words in the document ‘d’ are generated using the following process.

\[ P(w_i) = \sum_{j=1}^{t} P\left( \frac{w_i}{T_i} = j \right) P(T_i=j), \]  
(4.1)

Here, \( T \) stands for the number of topics. We used MALLET (MAchine Learning for Language Toolkit) \cite{47} a java-based package to build our topic model. Apart from identifying the topics in the documents, topic modeling also aims to minimize the redundancy in the documents. So, the work in \cite{8} considers topic modeling as matrix factorization method and used the following heuristic to determine the number of inherent topics in the data.

\[ \text{argmin}_k \left\| \sum_{i=1}^{k} e_i - \sum_{i=k}^{n} e_i \right\| \]  
(4.2)

Here ‘\( e_i \)’ stands for the eigen values obtained after performing the principal component analysis (PCA) on the document-term matrix. We calculate the scores for the values \( k=1 \) to \( n \), and identify the ‘\( k \)’ value where the function is minimized.

**4.2.4 Similarity Calculation**

We use the Jeffrey’s Kullback-Leibler (J) divergence \cite{36} to compute the differences between the topic distributions in the selected disease and drug profiles. Initially we used the KL-divergence to measure the similarity between profiles. Even though KL-divergence is predominantly used to calculate the distance between two probability distributions, it is not a true metric as it is not symmetric. The KL-divergence of \( P \) and \( Q \) is not equal to KL-divergence of \( Q \) and \( P \), unless \( P \) and \( Q \) are equal. So in current study, we calculate the symmetric form of KL-divergence (named J-divergence), which is given by \( D(P, Q) = D_{KL}(P \parallel Q) + D_{KL}(Q \parallel P) \). Also, in our preliminary experiments considering symmetric form of KL-divergence gave better results. We use the intuitive idea that the drugs that are likely to be repurposed for a disease will have significantly small divergence values when compared to the average of the drug distances to that disease. We capture this notion by imposing the condition that for a drug to be considered as repositioning candidate it should have a z-score of -1.5 compared to the
average. In other words, if a drug’s divergence value is significantly smaller than the average divergence of all the drugs we consider it as a potential candidate for repositioning. We thus rank the drugs that have z-score of -1.5 or lower according to their J-divergence values and display them as probable drugs that can be repositioned for that disease.

4.3 Results

To evaluate the approach we used the metrics - accuracy, balanced accuracy, and precision. They are defined as follows:

\[
\text{Accuracy} = \frac{\text{Number of true positives} + \text{Number of true negatives}}{\text{Number of true positives} + \text{False positives} + \text{False negatives} + \text{True negatives}}
\]

\[
\text{Precision} = \frac{\text{Number of true positives}}{\text{Number of true positives} + \text{False positives}}
\]

\[
\text{Balanced accuracy} = \frac{\text{sensitivity} + \text{specificity}}{2}
\]

4.3.1 Validation using known drug - disease pairs

To validate our approach we had used six examples of repositioned drugs (11 disease-drug pairs; 5 drugs with two indications each and one class of depression-related drugs as repositioning candidates for dysmorphology). The selected drugs and their known indications are shown in Table 4.1. The goal was to see how topic model-based approach would rank the drug against its multiple indications (i.e., drug-A vs. disease-1 and drug-A vs. disease-2). As described in methodology section, we download the drug and disease related abstracts, and exclude the abstracts which cite in both drug and disease. We mixed 9 random drug profiles with each disease-drug pair and calculated the rank of the original drug for the disease. We repeated this process 10 times for each disease-drug pair and calculated the performance metrics. For the validation sets, accuracy, balanced accuracy, and precision was 0.83, 0.75, and 0.32 respectively with an AUC of 0.74. The ROC curve is shown in Figure 4.2. The target drug was ranked at top 61% of the time.
TABLE 4.1: Examples of drugs with multiple indications

<table>
<thead>
<tr>
<th>Drug</th>
<th>Indication-1</th>
<th>Indication-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Formoterol</td>
<td>Asthma</td>
<td>Stuttering</td>
</tr>
<tr>
<td>Mitoxantrone</td>
<td>Multiple Sclerosis</td>
<td>Prostate cancer</td>
</tr>
<tr>
<td>Modafinil</td>
<td>Narcolepsy</td>
<td>Bipolar disorder</td>
</tr>
<tr>
<td>Ropinirole</td>
<td>Parkinson’s disease</td>
<td>Restless legs syndrome</td>
</tr>
<tr>
<td>SSRIs</td>
<td>Depression</td>
<td>Dysmorphic disorders</td>
</tr>
<tr>
<td>Terbutaline</td>
<td>Asthma</td>
<td>Preterm labor</td>
</tr>
</tbody>
</table>

FIGURE 4.2: ROC curve for validation data.

4.3.2 New Indication Search for Rare Diseases

We have 10 datasets, with each dataset containing the profiles for rare diseases and 1694 drugs. In each dataset, the top twenty most similar drug profiles for each of the six rare diseases are identified by calculating the similarity between a rare disease and the 1694 drugs. Now for each rare disease, we record how many times a particular drug has been identified in the top and rank the drugs by their frequency of occurrence in the list. Table 4.2 displays the top 15 ranked drugs for each of the 6 rare diseases. Literature search showed that most of the top ranked drugs could be related to their mapped respective rare diseases suggesting the utility of our approach in discovering drug-repositioning candidates.

Pipobroman is the top ranked drug for polycythemia vera (PV) and there are several studies reporting the efficacy of it in PV [38, 53]. Similarly, ruxolitinib has ranked second for PV and third for primary myelofibrosis. Ruxolitinib has been recently reported to provide clinical benefits in patients with advanced PV [71] and in primary myelofibrosis [27].
Table 4.2: Ranked drugs for six rare diseases

<table>
<thead>
<tr>
<th>Rank</th>
<th>Polycythemia vera</th>
<th>Primary myelofibrosis</th>
<th>Dravet syndrome</th>
<th>Meningioma</th>
<th>Narcolepsy</th>
<th>Netherton syndrome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pipobroman</td>
<td>Riluzole</td>
<td>Lomustine</td>
<td>Cedropin</td>
<td>Zopiclone</td>
<td>Clocortolone</td>
</tr>
<tr>
<td>2</td>
<td>Anagrelide</td>
<td>Rp54029</td>
<td>Temozolomide</td>
<td>Clozapin</td>
<td>Lomustine</td>
<td>Flurbiprofen</td>
</tr>
<tr>
<td>3</td>
<td>Ruxolitinib</td>
<td>Enzalutamide</td>
<td>Docetaxel</td>
<td>Ketazolam</td>
<td>Clozapin</td>
<td>Flurbiprofen</td>
</tr>
<tr>
<td>4</td>
<td>Orphenvin</td>
<td>Rp54029</td>
<td>Clozepine</td>
<td>Ketazolam</td>
<td>Clozapin</td>
<td>Flurbiprofen</td>
</tr>
<tr>
<td>5</td>
<td>Hydromorphone</td>
<td>Choline</td>
<td>Rosafacine</td>
<td>Estazolin</td>
<td>Clozapin</td>
<td>Flurbiprofen</td>
</tr>
<tr>
<td>6</td>
<td>Tofacitinib</td>
<td>Valproic Acid</td>
<td>Alimentamycin</td>
<td>Camazapam</td>
<td>Clozapin</td>
<td>Flurbiprofen</td>
</tr>
<tr>
<td>7</td>
<td>L-Tyrosine</td>
<td>Lomotrige</td>
<td>Carmustine</td>
<td>Zopiclone</td>
<td>Flurbiprofen</td>
<td>Clozolicpinolide</td>
</tr>
<tr>
<td>8</td>
<td>Pomalidomide</td>
<td>Tofacitinib</td>
<td>Lamotrige</td>
<td>Zopiclone</td>
<td>Flurbiprofen</td>
<td>Clozolicpinolide</td>
</tr>
<tr>
<td>9</td>
<td>Pegademase bovine</td>
<td>Rasburiglucin</td>
<td>Lamotrige</td>
<td>Zopiclone</td>
<td>Flurbiprofen</td>
<td>Clozolicpinolide</td>
</tr>
<tr>
<td>10</td>
<td>Tofacitinib</td>
<td>Rasburiglucin</td>
<td>Lamotrige</td>
<td>Zopiclone</td>
<td>Flurbiprofen</td>
<td>Clozolicpinolide</td>
</tr>
<tr>
<td>11</td>
<td>Acenocoumarol</td>
<td>Rasburiglucin</td>
<td>Lamotrige</td>
<td>Zopiclone</td>
<td>Flurbiprofen</td>
<td>Clozolicpinolide</td>
</tr>
<tr>
<td>12</td>
<td>Rasburiglucin</td>
<td>Rasburiglucin</td>
<td>Lamotrige</td>
<td>Zopiclone</td>
<td>Flurbiprofen</td>
<td>Clozolicpinolide</td>
</tr>
<tr>
<td>13</td>
<td>Ginseng</td>
<td>Rasburiglucin</td>
<td>Lamotrige</td>
<td>Zopiclone</td>
<td>Flurbiprofen</td>
<td>Clozolicpinolide</td>
</tr>
<tr>
<td>14</td>
<td>Rasburiglucin</td>
<td>Rasburiglucin</td>
<td>Lamotrige</td>
<td>Zopiclone</td>
<td>Flurbiprofen</td>
<td>Clozolicpinolide</td>
</tr>
<tr>
<td>15</td>
<td>Rasburiglucin</td>
<td>Rasburiglucin</td>
<td>Lamotrige</td>
<td>Zopiclone</td>
<td>Flurbiprofen</td>
<td>Clozolicpinolide</td>
</tr>
</tbody>
</table>

Figure 4.3: Stacked bar chart showing the top five topic proportions found in modafinil (drug) and its two indications (bipolar disorder and narcolepsy) and ten random disease sets.

4.3.3 Topic concepts as indicators for drug repurposing

By identifying the concepts that are present in the highly shared topics between a drug profile and disease profile we can know the inherent hidden relationships between the drug and a disease. For example, the topic 4 in Figure 4.3 shared between modafinil and biopolar disorder showed words/concepts related to neuropsychiatric or behavioral conditions (e.g., mental depression, major depressive disorder, attention deficit hyperactivity disorder, antidepressive agents, mental association, methylphenidate, attention, lithium, sleep, etc.) while topic 0 shared between modafinil and narcolepsy was predominantly sleep-related (sleep disorders, cataplexy, narcolepsy-cataplexy syndrome, sleep, REM, obstructive sleep apnea, drowsiness, hypersomnia, wakefulness, REM sleep behavior disorder, etc.)
4.4 Conclusion

We used topic modeling to estimate the probability distribution of topics for each of the drugs or diseases and calculate the disease-drug similarity. Instead of using the abstracts directly, we used mapped biomedical concepts for topic modeling which increased the specificity and also overcome the problem of biomedical stop words to some extent. Also, our approach compares disease and drug directly where as most of the previous approaches focus on either drug-drug or disease-disease relationships to find drug repositioning candidates. Further, to the best of our knowledge, this study is the first to use topic modeling on MEDLINE abstracts for drug repositioning candidate discovery for rare diseases. Since, literature related to drugs and diseases are constantly updated, the dynamic and temporal nature of the disease and drug concepts can be utilized for a more robust drug repositioning and computational pharmacovigilance systems. We also note that a high conceptual similarity between a disease and drug may not always suggest alternate indication but semantic relatedness or potential contraindication or even drug related side-effects. Although we focus on drug repositioning in this study, based on our results, the current approach can also be employed to understand the molecular basis of side-effects or suggest safer alternatives (e.g., drugs with fewer side-effects for similar indication) by ranking drugs against diseases based on side-effects topics.
Chapter 5

Bi-clustering Based Shared Concept Finding in Documents

With the increase in use of the Internet, a huge amount of documents are generated in online format and it is getting difficult to manually organize these documents. So, numerous text-mining techniques are proposed in literature to automatically classify or cluster these documents into their natural groups.

Clustering is used to group the similar objects together. Given a collection of text documents, clustering will help in organizing the documents and there by enhances the future navigation and search. To apply any clustering algorithm on the documents, one should first represent the documents in a bag of words format. The main idea is to extract the relevant and content specific words from the documents using some special tools like tagging/mapping and use the extracted words as features to represent the document. Thus, the entire document collection is represented by a document - word matrix \( M \) - with rows, columns of \( M \) corresponding to documents and words. A non-zero entry \( M_{i,j} \) in the matrix \( M \) indicates the number of occurrences of the word \( j \) in document \( i \). But the problem is that, out of the entire corpus, only a small number of words occur in each document. This property makes the Matrix \( M \) very sparse and limits the ability to define similarity on full dimensional clustering algorithms. For example, if two or more texts share only a subset of words, the standard clustering algorithms will fail to identify this.

Bi-clustering is a technique that allows simultaneous clustering of rows and columns
of a matrix. Bi-clustering of the textual data will allow us to group the documents that share a subset of words. For example, we can cluster the words based on the documents they co-occurred. Some applications where such kind of clustering is used can be in the automatic construction of a statistical thesaurus or in the auto enhancement of search queries. Several bi-clustering algorithms have been proposed that deal with the binary matrices. These algorithms are able to identify pure bi-cluster, which is a subset of documents and words that have a strict linkage between them. However, in some applications we might have to find non-pure bi-clusters. For example, we may want to classify a document into a group even if it shares 90% of the shared subset of words that belongs to that group. As the matrix generated from textual data will be very sparse, a 90% overlap of words is statistically significant. In some cases, the concept tagging/mapping process fails to properly identify the occurrence of a concept in a document and because of this a zero value (noise) is induced into the dataset. When an algorithm that discovers strict bi-clusters is employed on such a binary dataset, it fails to identify the noisy document into a bi-cluster that shares all the remaining subset of concept.

To solve this problem we propose a novel search based bi-clustering algorithm for binary matrices that allow non-pure bi-clusters. This new form of bi-cluster will still be able to capture the strict linkages between the rows and columns and additionally help to identify the ones that have some missing components. Using the proposed algorithm one can get a finer control over the number of zero’s that can be permittable in a bi-cluster.

In addition to text mining, one application where finding these non-pure bi-clusters is much useful is biomedical domain. For example, let’s say we have a dataset of genes vs diseases. Identifying a non-pure bi-cluster will give us insight on the genes that cause a set of diseases. The missing links (zero values) in the bi-cluster comes because of two cases. 1) There is no actual link between that disease and gene. 2) The relationship between that gene and disease is not well studied. If the missing zero is because of the 2nd case, then this non-pure bi-cluster have a significant interest to the researcher.
5.1 Background and Literature

Bi-clustering has many applications in text mining, Biomedical data analysis, Time series analysis, target marketing etc. Cheng and Church in [15] are the first people to work on the bi-clustering problem. The main goal of Cheng and Church in [15] is to find the bi-clusters with a low mean squared residual value, which is a measure of the bi-cluster homogeneity. Their approach involves a local search procedure that involves deleting/adding of rows and columns to the bi-clusters. To find all the bi-clusters that satisfy their objective, they repeat their procedure on the remaining rows and columns that are not present in the previous obtained bi-clusters. Research in [18] had proposed a bi-clustering algorithm to simultaneously cluster the documents and words. They represent the documents with a bipartite graph of words and try to find the minimum cut vertex partitions in this graph.

Numerous bi-clustering algorithms [23, 31, 80] are proposed in literature for continuous valued matrices. However, only very few algorithms are proposed in literature that are suitable for binary data. A divide and conquer based approach was used by [55] to find the binary maximal bi-clusters. The work in [79] proposed the binary matrix factorization (BMF) algorithm using the idea of matrix factorization. The objective is to decompose the input binary matrix ‘A’ into two small binary matrices ‘B’ and ‘C’, such that $A = B \times C$. The work in [57] proposed the BiBit algorithm taking advantage of the bit pattern based processing and selective search strategy and show that their approach is not affected by the shape and density of the matrix.

However, since these methods search for maximal bi-clusters with all 1’s, they will produce many small fragmented bi-clusters if a bi-cluster of interest is not pure or if a small noise is presented in the background. Most of the real world dataset’s are very sparse and finding bi-clusters with small number of zero’s (relaxed bi-cluster) will reveal interesting relationships that may not be captured by the bi-clusters with all one’s (strict bi-cluster’s). To address this, the work in [43] proposed an algorithm to find bi-clusters with dense one’s which are statistically significant. Based on the same concept, the work of [68] adapts a score function to effectively discover statistically significant
bi-clusters in sparse binary datasets. However, both of these algorithms identify a bi-cluster (with dense 1’s), replaces all the elements of generated bi-cluster to zero in the input matrix, and then identify the 2\textsuperscript{nd} statistically significant bi-cluster. Because of this, the algorithm proposed in [43, 68] fails to find overlapping bi-clusters, which are important in many domains. Also, the bi-clusters in [43, 68] are guaranteed to be maximal in only one direction but not in both directions. In this proposal, we design a new algorithm that is capable of finding overlapping maximal relaxed bi-clusters which are statistically significant.

5.2 Approach

First, we will explain about the kind of bi-clusters we are interested in. Let’ say, we have a sparse binary dataset ‘DS’ with ‘M’ rows, ‘N’ columns, and sparsity ‘D’, with ‘D\ll 10. Here, $D$ represents the proportion of non-zero cell entries present in the binary dataset. In this dataset, the probability of finding a submatrix with dense one’s (‘$d$’ value close to 100 - ‘$d$’ represents the sparsity of submatrix), obtained by random selection of a subset of rows, columns will be very low. The larger such bi-cluster the smaller will be the probability of occurrence of that submatrix. Such large bi-clusters will be of high interest as they reveal many interesting associations. So, our goal here is to find all such submatrices that have a low probability of occurrence in the dataset. We define the term ‘relaxed bi-cluster’ as follows:

5.2.1 Relaxed Bi-cluster

A relaxed bi-cluster is a maximal submatrix with $d\%$ of 1’s, where $d \gg D$ and formed by random selection of subset of rows and columns. That is, the submatrix must constitute a dense selection of ‘1’ entries and be statistically significant with a reasonably large size. The merit of the relaxed bi-cluster is given by a variant of the chernoff bound.
5.2.2 Merit score

In order to compare the quality of the bi-clusters during the search, we need to derive a metric to evaluate the merit of a bi-cluster. We treat each element of the dataset ‘DS’ to be an outcome of a Bernoulli trial with success probability ‘p’, where ‘p’ is defined as $D/(M \times N)$. The relaxed bi-cluster ‘RB’ with ‘m’ rows, and ‘n’ columns can now be represented using a random variable $X_{m,n}$ that follows a binomial distribution with parameters ‘m*n’ and ‘p’. As we are interested in bi-clusters of dense ones, we look for submatrices that have a very low $P(X_{m,n} > k)$. Here, ‘k’ represents the number of one’s in the bi-cluster. As we are dealing with sparse datasets, the probability values become extremely small. So, instead of using the actual probability values, we consider the multiplicative version of the Chernoff bound which is an exponentially decaying upper bound. It is given by the formula:

$$P(X_{m,n} > k) \leq \begin{cases} e^{-\frac{(k-mnp)^2}{3mnp}}, & k \in [mnp, 2mnp]; \\ e^{-\frac{(k-mnp)^2}{k+mnp}}, & k > 2mnp. \end{cases} \quad (5.1)$$

We should note that this bound is valid only when ‘k>mnp’. That is ‘k’ should be greater than the expected number of one’s in a submatrix of size ‘m*n’. Instead of calculating the entire probability value, we consider only the exponent part and look for submatrices that have a larger exponent value. It is given as follows:

$$Score = \begin{cases} \frac{(k-mnp)^2}{3mnp}, & k \in [mnp, 2mnp]; \\ \frac{(k-mnp)^2}{(k+mnp)}, & k > 2mnp. \end{cases} \quad (5.2)$$

In 5.2 we are not taking the size of the rows (m) and size of columns (n) into account separately. So to differentiate the number of rows and columns in a submatrix we normalize the score from 5.2 with some functions of ‘m’ and ‘n’. So, we use the normalizing factor $m^\alpha n^\beta$ with $\alpha, \beta$ values between [0.5, 1]. The final function used to
calculate the merit of a bi-cluster is given by the following equation:

\[
Score_{\alpha,\beta} = \begin{cases} 
\frac{(k-mnp)^2}{3mnp(m^*n^*)}, & k \in [mnp, 2mnp] ; \\
\frac{(k-mnp)^2}{(m^*n^*)(k+mnp)^2}, & k > 2mnp.
\end{cases}
\] (5.3)

5.2.3 Search algorithm

Our search procedure to find the relaxed bi-clusters consists of three steps:

- Generating the seeds.
- Extending the seeds until the merit value is decreased.
- Merging the bi-clusters that have a high degree of overlap.

Seed Generation:

We will enumerate all possible 3*3 bi-clusters with strict 1’s as seeds and store them in a priority queue ‘Q’. At this point all the seeds will have the same merit score. Even though, we generate a large number of seeds in this process, the number of potential seeds will drastically go down as they fail to expand further in the search process.

Seed Extension process:

Our search strategy is primarily based on the A* search algorithm and is guided by the heuristic ‘Merit score’. After generating the seeds, each seed will be extended till we find the maximal relaxed bi-cluster that satisfies our merit function constraints. First, we implement a priority queue to store all the seeds based on the merit value. During each iteration the top seed \( s_1 \) from priority queue is removed and extended by adding a row or a column. In this process, we generate multiple new potential seeds that satisfy our criteria. If the merit value of newly formed intermediate seed is decreased compared to the parent seed \( s_1 \), then we add the parent seed \( s_1 \) to the final candidate set. Else, we add all the intermediate seeds into the priority queue based on their merit value. While adding the new seeds in each iteration, we also check for any seeds that
Chapter 5. Bi-clustering Based Shared Concept Finding in Documents

**Input:** Dataset $DS$, Priority Queues $Q, \alpha, \beta, \delta$

**Output:** A list of relaxed bi-clusters $F$

```
while Priority Queues $Q$ not empty do
  $s_1 \leftarrow$ Remove the top seed from $Q$
  Generate intermediate seed set $Q'_1 \leftarrow GenerateExtensions(DS, s_1, \alpha, \beta)$
  if $Q'_1$ not empty then
    $Q_1 \leftarrow \{Q_1 \cup Q'_1\}$
  else
    insert $s_1$ into $O$, which is the list of relaxed bi-clusters identified in $DS$
  end
for each bi-cluster $a$ in $O$ do
  $b \leftarrow$ Identify the bi-clusters in $O$ that have a JaccardSimilarity $> \delta$.
  for each bi-cluster $c$ in $b$ do
    $c \leftarrow$ Merge the rows, columns from $a, b$ and generate new bi-cluster.
    Insert $c$ into output list $F$.
  end
end
```

Algorithm 1: Steps involved in search process

are subsumed with in the intermediate seed and remove them from the priority queue. We repeat this process until the priority queue ‘$Q$’ gets empty which indicates that all maximal relaxed bi-clusters that satisfy our criteria are identified. The procedure described above to extend the seeds is explained in algorithm 1.

Given a seed and dataset, the Algorithm 2 (GetExtensions) will produce the extended versions of the given seed by adding a row/column. Given a bi-cluster, we calculate the row sums of all the rows for the subset of columns in the given bi-cluster and sort the rows in decreasing order according to rowsums. Now we add the row, which is next to the rows (rows in bi-cluster) in the sorted row list. We repeat the same procedure to identify the column that should be added to bi-cluster. A row/column that is not qualified to add to the bi-cluster at this stage can be qualified to add at a later stage. So it is not possible to prune any of the rows or columns.

We have two parameters ‘$\alpha$’ and ‘$\beta$’ in our merit function. The decision whether to add a row or a column to a bi-cluster is influenced by these ‘$\alpha$’ and ‘$\beta$’ values. If both the values are equal, then a row and a column that are of equal size with equal proportion of ones are treated equal. If ‘$\alpha$’ is greater than ‘$\beta$’, then a column is given preference over a row and vice versa if ‘$\alpha$’ is less than ‘$\beta$’.
Input: Dataset $DS$, seed $b, \alpha, \beta$
Output: A list $l$ with expanded seeds for $b$

1. Initiate $l = \text{none}$
2. Initiate $\text{score} = b.\text{Score}_{\alpha, \beta}$
3. $rs \leftarrow \text{Compute the rowsums over the subset of columns in } b$
4. $r' \leftarrow \text{Sort rows in } DS \text{ according to rowsums}$
5. $r'' \leftarrow \text{Identify the rows with highest sum and exclude the rows in } b$
6. $cs \leftarrow \text{Compute the columnsums over the subset of rows in } b$
7. $c' \leftarrow \text{Sort columns in } DS \text{ according to columnsums}$
8. $c'' \leftarrow \text{Identify the columns with highest sum and exclude the columns in } b$
9. $l \leftarrow \text{Generate new bi-clusters by adding a row/column from } r'', c'' \text{ to } b$
10. for each extended bi-cluster $eb$ in $l$ do
    11.     Compute $\text{Score}_{\alpha, \beta}$
12. end
13. if none of the bi-clusters in $l'$ have $\text{Score}_{\alpha, \beta} > \text{score}$ then
    14.     return $l = \text{none}$
15. end

Algorithm 2: Steps involved in seed extension process

5.2.4 Merge bi-clusters with minor variations

After the search process is completed, we might have many minor variants for each bi-cluster in the list. So, for each bi-cluster in the list, we look for minor variants by calculating the Jaccard similarity score between each pair of bi-clusters in the list and merge the pair if they have very high similarity score.

5.3 Results

We evaluated our algorithm on numerous synthetic and real dataset’s. We used the synthetic datasets to analyze the behavior of our algorithm with respect to input parameters and density of input dataset. We also applied two more well-known bi-clustering algorithms on the same datasets to compare the results. One of them is the BiBit algorithm [57], which is designed to detect bi-clusters with all 1’s in a binary dataset. The second algorithm we used is the BicBin Algorithm [68] which aims to find relaxed bi-cluster’s in a binary dataset. We had selected these algorithms for comparison as these algorithms are trying to capture the similar notion as ours.
5.3.1 Synthetic datasets

In order to test the efficacy of our bi-clustering algorithm, we generated two matrices with sizes 41 X 40 and 108 X 108. In the first dataset, we planted six bi-clusters of different size, density and with various degree of overlap between the bi-clusters (Figure 5.1a). In the second dataset, we planted 10 overlapping bi-clusters as shown in Figure 5.1b. Furthermore, we shuffled the original matrices to make it complex for the algorithm to find the embedded bi-clusters (Figure 5.2a, 5.2b).

In addition to the proposed algorithm, we applied the BicBin and BiBit algorithms on these two datasets to compare the results. The ‘α’ and ‘β’ parameters are set to 0.9 in our algorithm and in BicBin. We set the minimum size of rows and columns to 3 for BiBit algorithm. In dataset one, our proposed algorithm outputted 19 bi-clusters...
Chapter 5. Bi-clustering Based Shared Concept Finding in Documents

(A) Bi-clusters identified in dataset 1

(B) Bi-clusters identified in dataset 2

Figure 5.3: Relaxed bi-clusters identified by proposed algorithm

Figure 5.4: Bi-clusters identified in dataset 2 using BicBin algorithm
Figure 5.5: Average density of 1’s per bi-cluster for different ‘α’ and ‘β’ values.

(Figure 5.3a). It successfully uncovered all the six patterns in their maximal sizes along with numerous smaller bi-clusters in the matrix. The BicBin algorithm is also able to find the core of the six embedded bi-clusters, but it failed to output the maximal bi-clusters for six bi-clusters. It also failed to find some significant smaller bi-clusters that are identified by our algorithm. On the other hand, BiBit algorithm which is capable of finding bi-clusters with all elements equal to one, excluded the rows or columns that have one or more zeros. Further, we noticed that each bi-clusters identified by the BiBit algorithm is subsumed in one or more bi-clusters in our proposed algorithm. The BiBit algorithm has outputted 136 bi-clusters even in this small dataset.

In the second dataset, our algorithm gave 28 bi-clusters (Figure 5.3b). Along with the 10 embedded square bi-clusters, it identified 9 rectangular bi-clusters with more rows than columns (for example, bi-cluster with rows 1-28, columns 11-18), and 9 rectangular bi-clusters that have more columns than rows (for example, bi-cluster with rows 11-18, columns 1-28). Even after multiple runs, the BicBin algorithm failed to identify all the 10 embedded bi-clusters (Figure 5.4).

To study the effect of ‘α’ and ‘β’ parameters on our algorithm, we constructed 200 X 200 matrices with varying proportion of 1’s starting from 5 to 25 percent in multiples of 5. For example, to construct a 200 X 200 matrix with 10 percent 1’s, we first constructed a 200 X 200 matrix with all of its cells marked as 0, then for each row we randomly changed 20 cells to 1’s. We ran our algorithm on each of these synthetic datasets multiple times by changing the ‘α’ and ‘β’ input parameter values. As the ‘α’ and ‘β’ values
increased, the density of 1’s per bi-cluster increases (Figure 5.5). With ‘α’ and ‘β’ values close to 0.65, the algorithm finds more relaxed bi-clusters compared to the bi-clusters identified with higher ‘α’ and ‘β’ values, say 0.95 or 0.90. Also, as the ‘α’ and ‘β’ values decreased, the algorithm outputs fewer bi-clusters and the average bi-cluster size increases (the algorithm tries to identify more relaxed bi-clusters). The run time of the algorithm also highly depends on the proportion of 1s in the given dataset. (Figure 5.6). From our experiments on multiple datasets, we figured out the ideal value to start for ‘α’ and ‘β’ to be 0.75 - 0.85.

5.3.2 Real world datasets

We tested our algorithm on six real world datasets. The first dataset we used is created by selecting the documents corresponding to the topic ‘windows’ from the newsgroup dataset [51]. The second and third datasets represents the UMLS (Unified Medical Language System) concepts identified in the subset of the abstracts obtained by querying the pubmed database using the disease names ‘Parkinson Disease’ and ‘Restless Leg Syndrome’ as MeshTags. After obtaining the subset of pubmed abstracts, we used the Metamap program to tag the UMLS concepts in each of the abstracts. The fourth dataset is a biomedical dataset that is related to genes causing ‘premature death’ and how these genes are regulated. The fifth dataset is related to genes and diseases that cause abnormal spleen. The sixth dataset is the plants dataset downloaded from UCI Machine learning repository [7]. This dataset consists the information regarding
Table 5.1: Outline of the results achieved by our algorithm on different datasets. $\alpha, \beta = 0.90$

<table>
<thead>
<tr>
<th>Dataset size</th>
<th>Newsgroup Disease</th>
<th>Parkinson Disease</th>
<th>Restless leg Syndrome</th>
<th>Premature Death</th>
<th>Abnormal Plants</th>
<th>Plants size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset size</td>
<td>500 X 800</td>
<td>855 X 573</td>
<td>561 X 573</td>
<td>171 X 1099</td>
<td>168 X 757</td>
<td>3000 X 67</td>
</tr>
<tr>
<td>Density (%)</td>
<td>4.27</td>
<td>1.43</td>
<td>1.48</td>
<td>3.7</td>
<td>1.23</td>
<td>13.24</td>
</tr>
<tr>
<td>Biclusters identified</td>
<td>1721</td>
<td>1090</td>
<td>941</td>
<td>682</td>
<td>267</td>
<td>1432</td>
</tr>
<tr>
<td>Avg. rows/Bicluster</td>
<td>6.5</td>
<td>5.8</td>
<td>8.59</td>
<td>7.03</td>
<td>4.9</td>
<td>84.2</td>
</tr>
<tr>
<td>Avg. cols/Bicluster</td>
<td>4.2</td>
<td>3.6</td>
<td>3.8</td>
<td>12.83</td>
<td>4.3</td>
<td>43.2</td>
</tr>
<tr>
<td>Avg. density/Bicluster</td>
<td>0.97</td>
<td>0.98</td>
<td>0.97</td>
<td>0.91</td>
<td>0.96</td>
<td>0.98</td>
</tr>
<tr>
<td>Time Taken (secs)</td>
<td>1703</td>
<td>1267</td>
<td>984</td>
<td>842</td>
<td>632</td>
<td>2235</td>
</tr>
</tbody>
</table>

Table 5.2: UMLS concepts that co-occur in abstracts for Parkinson Disease

<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
<th>C3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neostriatum</td>
<td>Alzheimer's Disease</td>
<td>Muscle Rigidity</td>
</tr>
<tr>
<td>Shy-Drager Syndrome</td>
<td>Interleukin-1</td>
<td>dopamine transporter</td>
</tr>
<tr>
<td>Lewy Body Disease</td>
<td>Tumor Necrosis Factors</td>
<td>Dopa mine Hydrochloride</td>
</tr>
<tr>
<td>Corpus striatum structure</td>
<td>Parkinson Disease</td>
<td>Parkinson Disease</td>
</tr>
<tr>
<td>Parkinsonian Disorders</td>
<td>Lipopolysaccharides</td>
<td>Tremor</td>
</tr>
<tr>
<td>Multiple System Atrophy</td>
<td>Tumor Necrosis Factor-alpha</td>
<td>physiological aspects</td>
</tr>
</tbody>
</table>

Table 5.3: UMLS concepts that co-occur in abstracts for Restless leg syndrome

<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
<th>C3</th>
</tr>
</thead>
<tbody>
<tr>
<td>modafinil</td>
<td>Depressive Symptoms</td>
<td>Mental Depression</td>
</tr>
<tr>
<td>Sleep Apnea, Obstructive</td>
<td>Dopamine Hydrochloride</td>
<td>Respiration</td>
</tr>
<tr>
<td>Melatonin</td>
<td>Sleep</td>
<td>Sleep Disorders</td>
</tr>
<tr>
<td>Sleeplessness</td>
<td>Sleep Disorders</td>
<td>Sleeplessness</td>
</tr>
<tr>
<td>Clonazepam</td>
<td>Parkinson Disease</td>
<td>Inspiration function</td>
</tr>
<tr>
<td>Sleep Disorders</td>
<td>REM Sleep Behavior Disorder</td>
<td>Hypersomnia</td>
</tr>
<tr>
<td>Restless Legs Syndrome</td>
<td>Restless Legs Syndrome</td>
<td>Restless Legs Syndrome</td>
</tr>
</tbody>
</table>
less time compared to our algorithm (Figure 5.7), it failed to identify some of the bi-clusters outputted by our algorithm. The effect of $\alpha, \beta$ on size, density of a bi-cluster in newsgroup dataset are shown in Table 5.4 We can notice that as we decrease these $\alpha, \beta$ parameter values, the size of the bi-cluster get’s increased, and density is decreased as more rows/columns with high % of zero’s get added to bi-cluster.

### 5.4 Conclusion

In this chapter, we proposed an algorithm that finds relaxed bi-clusters in large sparse binary datasets. Using the proposed algorithm we tried to group the documents together based on the words shared by the documents. Unlike the previous algorithms which group the documents with the same set of words, we are able to classify a document into a group even if it shares a high percentage of words to the subset of words that belongs to that group. We have also evaluated our algorithm on numerous synthetic and real biomedical datasets and compared our algorithm with BicBin and BiBit bi-clustering algorithms.
Chapter 6

Learning Relaxed 3-clusters from Pairs of Related Datasets

6.1 Introduction & Background

Most of the datasets are created independently of each other. However, the data in some of these datasets is interrelated to each other and these datasets need to be processed simultaneously to get the relationship between the data in these datasets. One of the goals of such processing is to discover some interesting subspace clusters in the integrated data. In some of the cases these datasets would be really huge containing a vast amount of data or the data in the datasets might be of sensitive in nature. So, these datasets can’t be brought to one place and merged. Moreover, if we perform our analysis on this combined dataset, the goal of finding the relationship between the data items of the two dataset’s can not be easily achieved. So, there is a need to design algorithms that coordinates the interaction between the two dataset’s leaving the dataset’s where they reside and do the analysis. In this chapter, we propose an algorithm that find 3-cluster’s from two related binary datasets. The proposed algorithm can easily be generalized to any number of datasets. The nature of the 3-clusters our algorithm will find in binary datasets is shown in Figure 6.1. The diseases shown in red color in Figure 6.1d are the shared diseases that contributed to form the 3-cluster.

Let’s say, for example, we have a binary dataset that consists of various diseases (rows) and genes (columns). Each row of this dataset represents a disease and each column
(A) Diseases vs. Genes

\[
\begin{array}{cccccc}
1 & 1 & 1 & 1 & 1 & 0 \\
2 & 1 & 1 & 1 & 1 & 0 \\
3 & 1 & 1 & 1 & 0 & 0 \\
4 & 0 & 0 & 0 & 1 & 0 \\
5 & 0 & 0 & 0 & 1 & 1 \\
6 & 0 & 0 & 0 & 1 & 1 \\
\end{array}
\]

(B) Diseases vs. Drugs

\[
\begin{array}{cccccc}
1 & 1 & 1 & 1 & 1 & 0 \\
2 & 1 & 1 & 0 & 0 & 0 \\
3 & 1 & 0 & 1 & 0 & 0 \\
4 & 0 & 0 & 0 & 0 & 0 \\
5 & 0 & 0 & 0 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

(C) Relaxed bi-clusters in 6.1a & 6.1b

\(<\{1, 2, 3\}, \{a, b, c, d\}>\)

\(<\{1, 2, 3, 4, 5, 6\}, \{c, d\}>\)

\(<\{5, 6\}, \{c, d, a\}>\)

\(<\{1, 2, 3\}, \{m, n, o, p\}>\)

(D) 3-clusters

\(<\{1, 2, 3\}, \{a, b, c, d\}>\)

\(<\{5, 6\}, \{c, d\}>\)

\(<\{1, 2, 3, 4, 5, 6\}, \{c, d\}>\)

\(<\{m, n, o, p\}>\)

### Figure 6.1: Sample Datasets and relaxed 3-clusters identified

represents a gene. A cell entry at \(i, j\) will be ‘1’ if the gene ‘j’ exists in disease ‘i’ else, the cell entry will be ‘0’. A strict bi-cluster identified from this dataset consists of a submatrix of all 1’s such that adding a gene or disease is not possible without including some 0’s in to the submatrix. The strict bi-clusters identified in this dataset will give us valuable insight on the ‘groups of genes’ that exists in a certain ‘group of diseases’. Now let’s say we have one more dataset that contains diseases as rows and drugs as columns. A strict bi-cluster in this dataset represents the ‘groups of drugs’ that can be used in a ‘group of diseases’. Now if we consider a pair of strict bi-clusters one from each dataset which have a high degree of overlap on the ‘diseases’, this pair of bi-clusters will provide us some information on the relationship between the genes and drugs mediated by the diseases. Discovery of such underlying relationships are very useful in many applications.

The example explained above deals with three different domains, namely - diseases, genes, drugs. A pair of bi-clusters, one from each dataset, with a high degree of overlap over the shared dimension - diseases - is a very insightful pair as it meaningfully connects groups of items from three different domains. We call such pair as a 3-cluster.

Some of the scenarios where 3-clusters from pairs of datasets can be useful are: documents vs. words and documents vs. authors; states vs. plants and states vs. weather.
parameters; keywords vs. Ad campaigns and keywords vs. URL’s. A 3-cluster identified from datasets ‘documents vs. words’ and ‘documents vs. authors’ will give us information on the authors and their choice of words. Similarly a 3-cluster identified in ‘states vs. plants’ and ‘states vs. weather parameters’ will give us relationship between the groups of plants and their ideal growth conditions.

Numerous heuristic search based algorithms [13, 72] that identify strict bi-clusters in a binary dataset are proposed in literature. However, to identify the 3-clusters we have to further improve these algorithms so that they can achieve the capacity to deal with multiple datasets simultaneously. Generally there are two naive approaches by which we can identify the 3-clusters from pairs of related datasets. 1) In the first approach we will first find all the bi-clusters in each of the individual datasets and then compare them to find the matching ones that have a high degree of overlap on shared dimension. With this approach we generate a lot of unwanted bi-clusters that don’t have a corresponding matching bi-cluster in 2nd dataset. A lot of computation time is wasted in generating these unwanted bi-clusters in each dataset. So this approach is computationally expensive. The spaces of possible bi-clusters must remain implicit and we need a heuristic search algorithm that simultaneously traverses the bi-cluster search spaces of the two datasets for efficiently identifying the pairs of matching bi-clusters. 2) In the second approach, we join the datasets along the shared dimension and try to find the bi-clusters in the combined dataset. Most of the bi-clusters generated from the combined dataset will not span across the attributes in both the datasets. So the idea of maximizing the match between the bi-clusters of two distinct datasets cannot be easily achieved. Even if we find a bi-cluster that has attributes from both the datasets, the individual bi-cluster from each dataset may not be maximal. So we lose information about individual datasets. To overcome these drawbacks, we propose a heuristic based search algorithm that simultaneously traversals the search spaces of the two datasets. In most of the cases, large datasets reside on multiple servers and our algorithm communicate minimal information to guide the search process. This approach can easily be generalized to multiple datasets.

Till now all the 3-clustering algorithms [2, 32] require each individual bi-cluster to be of
Chapter 6. Learning Relaxed 3-clusters from Pairs of Related Datasets

all 1’s and use formal concept analysis to address this 3-clustering problem. However, in many real world situations the datasets are sparse and relaxing the strict requirement of all 1’s is desirable in many scenarios. For example let’s consider a scenario where we are given a binary dataset of disease vs genes. From this sample dataset we had identified a strict bi-cluster of size 8 X 10 (8 diseases, and 10 genes). Now say we have one more gene (11th) that is present in 7 of the diseases out of 8. It is highly desirable to consider this 11th gene in to the group. However, including this gene into the bi-cluster will bring in a zero in to the strict bi-cluster and it will no longer be a strict bi-cluster. This gene will be of high attention from data mining perspective. The ‘0’ for this gene in the 8th disease can be of the following reasons. 1) There might be no actual connection between the selected gene and the disease. 2) The gene and the disease have a moderate relationship score and the dataset construction process had missed the link because of choosing a slightly higher cut-off score than the score of the gene for that disease. 3) Researchers had never studied the connection between the selected gene and disease. If the dataset is created from a solidly validated source then case 1 might be the likely cause for that gene missing in that disease. But in most scenarios the datasets are representatives of our evolving understanding of the underlying process. So case 2 and 3 are likely to happen in many real world situations and it is very desirable to discover 3-clusters in which each individual bi-cluster have some missing 1’s. These 0’s are pointers towards potential new knowledge or towards the errors induced in to the dataset due to the dataset construction process. We call such bi-clusters with few zeros as the ‘relaxed bi-clusters’.

First, we will define the notion of the ‘relaxed bi-cluster’ in terms of the probability of finding it by a random selection of subset of rows, columns, in the context of the sparseness of the entire dataset. Let’s say, we have a sparse binary dataset ‘DS’ with ‘M’ rows, ‘N’ columns, and sparsity ‘D’, with ‘D’<10. Here, D represents the proportion of non-zero cell entries present in the binary dataset. In this dataset, the probability of finding a submatrix with dense one’s (‘d’ value close to 100 - ‘d’ represents the sparsity of submatrix), obtained by random selection of a subset of rows, columns will be very low. The larger such bi-cluster the smaller will be the probability of occurrence of that submatrix. Such large bi-clusters will be of high interest as they reveal many interesting
associations. So, our goal here is to find all such ‘relaxed bi-clusters’ and 3-clusters that have a low probability of occurrence in the dataset’s.

6.2 Primitives

In this research work, we are dealing with binary dataset’s and we define the important terminology used in our approach. For example consider we have a dataset with $R$ rows, $C$ columns, and sparsity $P$. This tells us that there are $P\%$ ‘1”s in the dataset and the remaining entries are all ’0’s.

6.2.1 Definitions

Bi-cluster

A bi-cluster is a maximal submatrix with all 1’s formed by a random selection of subset of rows ‘r’ and subset of columns ‘c’ from the row set ‘R’ and column set ‘C’ of the Dataset.

Relaxed Bi-cluster

We define the desirability of a bi-cluster in terms of the probability of it being constructed by random selection from the complete dataset. The lower this probability value, the higher will be the number of 1’s in it, and also higher will be its interestingness. A relaxed bi-cluster is a maximal submatrix formed by the random selection of rows and columns with $p \%$ of 1’s, where $p >> P$ and whose probability of occurrence is very low. The probability of occurrence is given by the Chernoff bound value.

Relaxed 3-cluster

It is represented by a pair of relaxed bi-clusters $B_1, B_2$ whose intersection of rows is greater than certain threshold and taken from the datasets $D_1$ and $D_2$. 
6.3 Approach

6.3.1 Heuristic functions

According to the definition of 3-cluster, it contains a pair of relaxed bi-clusters, each
from one dataset and have a high overlap over the shared dimension. So to caption
this notion of 3-cluster we define the following heuristic functions.

**Fitness score:**

Each individual bi-cluster in the 3-cluster pair should have a very low probability of
occurrence in its dataset. So, we have to derive a metric that captures this notion and
help us in finding all such bi-clusters in a dataset.

Let’s say we have a dataset with p% 1’s. We consider each cell entry of the dataset to
be an outcome of the Bernoulli trial whose success probability is ‘p’. Now, a rectangle
formed by a random selection of subset of rows and columns is represented by the
variable $X_{r,c}$, which follows a binomial distribution with parameters (r*c) and ‘p’,
where ‘r’ and ‘c’ are the number of rows and columns selected in the rectangle. Let
‘k’ be the number of 1’s in the rectangle. When ‘k’ is close to the value (r*c*p), the
probability of finding such rectangles is very large where as when ‘k’ $\gg$ (r*c*p) the
probability of finding such rectangle will be very low. This is we will have a very low
$P(X_{r,c} > k)$. We are interesting in such rectangles where the probability of finding them
is very low. The probability values of these rectangles are given by the Chernoff bound
values shown in 6.1.

$$P(X_{r,c} > k) \leq \begin{cases} e^{-\frac{(k-rcp)^2}{3rcp}}, & k \in [rcp, 2rcp]; \\ e^{-\frac{(k-rcp)^2}{k+rcp}}, & k > 2rcp. \end{cases} \quad (6.1)$$

$$F.Score = \begin{cases} \frac{(k-rcp)^2}{3rcp}, & k \in [rcp, 2rcp]; \\ \frac{(k-rcp)^2}{(k+rcp)}, & k > 2rcp. \end{cases} \quad (6.2)$$
We should note that this bound is valid only when \( k > rcp \). That is \( k \) should be greater than the expected number of one’s in a rectangle of size \( r^*c^* \). Instead of computing the entire equation in 6.1 we calculate only the exponent part. That is instead of looking for rectangle’s whose Chernoff value is small, we look for rectangles whose exponent part in Chernoff value is large. One of the drawbacks of using this exponent value (6.2) is that it doesn’t take into account the size of the rectangle \( (r^*c^*) \).

For example consider a rectangle of size \((10X10)\) with \( k=100 \), and \( p=0.016 \). With 6.2 we get a score of 95.30. Now let’s consider one more rectangle of size \((10X11)\) with \( k=101 \), and \( p=0.016 \). It will have a score of 95.84. Ideally we want the rectangle \((10X10)\) to get higher score as it is very dense compared to the other rectangle. Moreover, we are not taking the size of the rows \( r \) and size of columns \( c \) into account separately. So, to differentiate the number of rows and columns in a rectangle we normalize the score from 6.2 with some functions of \( r^* \) and \( c^* \). So, we use the normalizing factor \( r^\alpha c^\beta \) with \( \alpha, \beta \) values between \([0.5, 1]\). The \( \alpha, \beta \) values are used to control the shape and size of the rectangles. That is given a row and a column with same size and same number of 1’s in them, when \( \alpha \) is greater than \( \beta \), adding a column will give higher score than adding a row and vice versa when \( \beta \) is greater than \( \alpha \). The final function used to calculate the merit of a bi-cluster with in it’s dataset is given by the following equation:

\[
F.Score_{\alpha, \beta} = \begin{cases} 
\frac{(k - rcp)^2}{3rcp(r^\alpha c^\beta)}, & k \in [rcp, 2rcp] \\
\frac{(k - rcp)^2}{(r^\alpha c^\beta)(k + rcp)}, & k > 2rcp.
\end{cases}
\] (6.3)

**Interestingness score**

This metric will evaluate the quality of a bi-cluster with respect to the 2\textsuperscript{nd} dataset. Let’s say: For example, there are 2 bi-clusters \( eb_1 \) and \( eb_2 \) that have the same size, density and same fitness score in dataset \( D_1 \). If \( eb_1 \) have a large number of matching bi-clusters in dataset \( D_2 \) and \( eb_2 \) has only a few matching bi-clusters that overlap in \( D_2 \), then it is appropriate to give more weightage to \( eb_1 \) compared to \( eb_2 \). So that the bi-clusters with a good matching score in 2\textsuperscript{nd} dataset will be expanded first. To capture this spirit, we define interestingness as follows:
Chapter 6. Learning Relaxed 3-clusters from Pairs of Related Datasets

Let’s say we have a bi-cluster seed $b_1$ from $D_1$ and $k$ bi-cluster seeds from $D_2$. We calculate the Jaccard similarity score between $b_1$ and all the $k$ bi-cluster seeds in $D_2$ over the shared domain. These similarity scores are sorted in descending order and are marked as $JS_1, JS_2, ..., JS_k$. Now the interestingness score is computed as in (6.4). A bi-cluster that have more matching bi-clusters in 2nd dataset has a value close to 1, whereas a bi-clusters with only very few partial matching bi-clusters will have a score close to zero.

\[
I.score = 0
\]
\[
for i = 1 : k \\
I.Score = I.score + JS_i(1 - I.Score)
\]  

(6.4)

Merit Score

The fitness score will tell us the significance of a bi-cluster in terms of its own dataset and Interestingness score provides the worthiness of the bi-cluster in terms of 2nd dataset. We want to first expand the bi-clusters that are promising in both the datasets, so we multiply the fitness and interestingness scores and use it as our final merit score (6.5).

\[
MeritScore = (F.Score) * (I.Score)
\]  

(6.5)

6.3.2 Algorithm Description

The main steps we follow in our algorithm are described in Figure6.2. Initially, we select the $\alpha,\beta$ values in both the datasets and by doing an exhaustive search generate all 3*3 candidate seeds in each of the datasets. Each of these initial seeds is a rectangle with all 1’s and all these seeds have the same fitness score. We identify the worthiness of each of the seeds in terms of the 2nd dataset by calculating the interestingness score for each of the seeds. Now we remove the seeds that have a interestingness score less than some threshold $\gamma$. The final candidate seeds in each of the datasets are sorted in decreasing
order according to their merit score and are stored in priority queue’s $Q_1$ and $Q_2$. Both
the priority queue’s will have a large number of seeds in the beginning but most of the
seeds fail to expand because of the sparsity of the dataset and eventually queue’s will
become empty at a much faster rate. As these seeds are produced by an exhaustive
search and only those seeds that are not promising in terms of 2nd are removed, it is
guaranteed that all the qualified relaxed bi-clusters that can be part of 3-clusters can be
identified by using these seeds. The seed generation process is explained in algorithm
3.

```
Input: Datasets $DT_1, DT_2$, Minimum interestingness score $\gamma$, Normalizing
values $\alpha, \beta$
Output: Priority Queues $Q_1, Q_2$ with candidate seeds
1. $f_1 \leftarrow$ Generate all (3*3) submatrices with one’s in $DT_1$
2. $f_2 \leftarrow$ Generate all (3*3) submatrices with one’s in $DT_2$
3. for each submatrix $S \in f_1$
do
4. Compute $I.Score$, the interestingness score between $S$ and $f_2$
5. if $I.Score < \gamma$ then
6. Compute $F.Score_{\alpha,\beta}$
7. Merit score $\leftarrow$ $(F.Score_{\alpha,\beta} \times I.Score)$
8. Insert $S$ into $Q_1$
end
10. end

11. for each submatrix $S \in f_2$
do
12. Compute $I.Score$, the interestingness score between $S$ and $f_1$
13. if $I.Score < \gamma$ then
14. Compute $F.Score_{\alpha,\beta}$
15. Merit score $\leftarrow$ $(F.Score_{\alpha,\beta} \times I.Score)$
16. Insert $S$ into $Q_2$
17. end
18. end
```

Algorithm 3: Steps to generate candidate seeds

The search strategy used in this algorithm is based on the A* search algorithm. Once
the candidate seeds are generated, we remove the top seed $t_1$ from the priority queue
$Q_1$ and try to expand the seed by adding a row/column to that seeds. This expansion
will generate few more candidate seeds. We calculate the fitness score and interestingness
score of these newly generated seeds and if the fitness score of these seeds is greater
than the fitness score of its parent seed $t_1$ we insert them back to the priority queue $Q_1$
based on their final merit score. The insertion of these newly generated intermediate
seeds into the priority queue will retain all the candidate seeds in the open frontier
Generate 3x3 seeds in datasets $D_1$, $D_2$

Delete seeds that don’t have a match in 2nd dataset

Place seeds in priority queue’s $Q_1$ and $Q_2$

if priority queue’s $Q_1$ and $Q_2$ empty

Expand top seed in $Q_1$

If new seeds generated

yes

Insert in priority queue $Q_1$

no

Insert the seed in final list $f_1$

Identify 3-clusters using $f_1$ and $f_2$ by calculating Jaccard score on shared domain

Stop

Expand top seed in $Q_2$

If new seeds generated

yes

Insert in priority queue $Q_2$

no

Insert the seed in final list $f_2$

Figure 6.2: Relaxed 3-clusters algorithm outline
of the search process. If the top seed $t_1$ is not able to generate any child seeds (by adding a row/column to $t_1$) with higher fitness score, then we store the seed $t_1$ in the list of final relaxed bi-clusters ($f_1$) in dataset 1. This means that adding a row/column will make the seed to lose its significance in its dataset. Then we check the priority queue $Q_1$ for any seeds that are completely subsumed by the seed $t_1$. Removing these subsumed seeds will eliminate the unnecessary cost involved in expanding these seeds as if expanded they will form the same seed $t_1$ which is already generated.

Now the procedure explained above for seed $t_1$ in priority queue $Q_1$ is repeated in priority queue $Q_2$ for top seed $t_2$. The process of expansion of the top seed and insertion of new intermediate seeds is repeated in consecutive steps for both the priority queue’s. We have to notice that we are using the final merit score to sort the seeds in the priority queue and the candidate seed is removed from the queue only when its fitness score is decreased compared to its parent seed. The fitness score constraint will ensure that the bi-clusters placed in the closed relaxed bi-cluster list are maximal relaxed bi-clusters with in their respective datasets. Also, sorting the priority queue with final merit score enables the bi-clusters which are significantly good globally in terms of both the datasets to be expanded first. By doing this we can eliminate the cost of expanding the unwanted seeds that don’t have a corresponding matching bi-cluster in 2\textsuperscript{nd} dataset. Once both the priority queue’s became empty, as a last step we calculate the jaccard similarity score over the shared domain between the bi-clusters in final closed bi-cluster’s lists - $f_1$, $f_2$. The pseudocode for the process explained above is presented in Algorithm 4.

Given a seed and dataset, Algorithm 5 (GetExtensions) will generate the extended versions of the given seed by adding a row/column. This Seed extension process is explained using the sample dataset and seed given in Figure 6.3.

Let’s say, the seed to be expanded is $R_3, R_4, R_5; C_2, C_3, C_4$. Given the seed, we will calculate the row sums for the corresponding columns ($C_2, C_3, C_4$) and the column sums for the corresponding rows ($R_3, R_4, R_5$). Now these rows and columns are sorted according to their row and column sums. Then we generate the child seeds by adding a row/column one at a time. Here, the rows ($R_6, R_7$) and columns ($C_5, C_6$) have the
Chapter 6. Learning Relaxed 3-clusters from Pairs of Related Datasets

**Input:** Datasets $DT_1, DT_2$, Priority Queues $Q_1, Q_2, \alpha, \beta, \delta$

**Output:** A list of 3-clusters

1. **while** Priority Queues $Q_1, Q_2$ not empty **do**
   2. $t_1$ ← Remove the top bi-cluster from $Q_1$
   3. Generate bi-cluster set $Q'_1 \leftarrow \text{GenerateExtensions} (DT_1, t_1, \alpha, \beta, Q_2, F_2)$
   4. **if** $Q'_1$ not empty **then**
      5. $Q_1 \leftarrow \{Q_1 \cup Q'_1\}$
   5. **else**
      6. insert $t_1$ into $F_1$, which is the list of final bi-clusters identified in $DT_1$
   7. $t_2$ ← Remove the top bi-cluster from $Q_2$
   8. Generate bi-cluster set $Q'_2 \leftarrow \text{GenerateExtensions} (DT_2, t_2, \alpha, \beta, Q_1, F_1)$
   9. **if** $Q'_2$ not empty **then**
      10. $Q_2 \leftarrow \{Q_2 \cup Q'_2\}$
   11. **else**
      12. insert $t_2$ into $F_2$, which is the list of final bi-clusters identified in $DT_2$
   13. **end**
   14. **end**
   15. **end**

16. **for each** bi-cluster $a$ in $F_1$ **do**
   17. $b$ ← Identify the bi-clusters in $F_2$ that have a Jaccard Similarity $> \delta$ on shared dimension.
   18. Insert $a, b$ into 3-clusters list.
   19. **end**

**Algorithm 4:** Steps involved in search process

---

**Input:** Dataset $D$, seed $b, \alpha, \beta, Q, F$

**Output:** A list $l$ with expanded seeds for $b$

1. Initiate $l = \text{none}$
2. Initiate score $= b.F.Score_{\alpha,\beta}$
3. $rs$ ← Compute the rowsums over the subset of columns in $b$
4. $r'$ ← Sort rows in $D$ according to rowsums
5. $r''$ ← Identify the rows with highest sum and exclude the rows in $b$
6. $cs$ ← Compute the columnsums over the subset of rows in $b$
7. $c'$ ← Sort columns in $D$ according to columnsums
8. $c''$ ← Identify the columns with highest sum and exclude the columns in $b$
9. $l$ ← Generate new bi-clusters by adding a row/column from $r'', c''$ to $b$
10. **for each extended bi-cluster $eb$ in $l$ do**
    11. Compute $F.Score_{\alpha,\beta}$
    12. Compute $I.Score (eb, \{Q \cup F\})$
    13. Compute Merit.Score $= (F.Score_{\alpha,\beta} * I.Score)$
    14. **end**
15. **if** none of the bi-clusters in $l'$ have $F.Score_{\alpha,\beta} > \text{score}$ **then**
    16. return $l = \text{none}$
17. **end**

**Algorithm 5:** Steps involved in seed extension process
same number of one’s and so we add each of these row/column to the seed. The inter-
mediate seeds generated for the sample are shown in Figure 6.3. For each child node
we compute the fitness and interesting score and insert the intermediate nodes in to
the queue if the child seeds have a low fitness score than its parent seed. The bi-cluster
that has a cross is not considered as the fitness score is decreased with respect to the
parent seed. The pseudo code for GetExtensions algorithm is presented in Algorithm
2.

6.4 Results and Analysis

We had analyzed the performance of our proposed algorithm on many synthetic and
real world datasets. Results from multiple datasets demonstrate the usefulness of the
results obtained and the success of our algorithm in identifying the 3-clusters. This
is the first work to identify the 3-clusters in which each of the individual bi-cluster’s
allow some zero values. So, we use the following two bi-clustering algorithms - Bibit
[58] and BiClBin [68] algorithms to compare our approach. Given two binary related
dataset’s that share a domain, we will first run the bi-clustering algorithms in each
of the individual datasets and then identify the pairs of bi-clusters that have a high overlap over the shared domain.

### 6.4.1 Synthetic datasets

We created a pair of binary datasets with size (108X108), and twenty % 1’s. In the first dataset, we planted ten overlapping bi-clusters of size 18 X 18 with all 1’s and changed some of these 1’s to 0’s. In the 2nd dataset, we planted 2 bi-clusters with different sizes and changed a small number of 1’s to 0’s. Then we shuffled the rows, columns in both the datasets (Figure 6.4). The proposed algorithm has identified the 2 bi-clusters in 2nd dataset and all the bi-clusters in the first dataset that have a match to the 2 bi-clusters in 2nd dataset. The 3-clusters that are identified from these datasets are presented in Figure 6.5c. Our algorithm successfully omitted the expansion of bi-clusters in the 1st dataset that don’t have a matching bi-cluster in 2nd dataset. For example, it didn’t output the bi-clusters that are exclusively made of row numbers greater than 70, also it didn’t include the column dominant bi-cluster [21:28;11:38] shown in blue in Figure6.5a as it don’t have a corresponding bi-cluster that shares the same set of rows in dataset 2.

With the bicbin algorithm we are able to identify the 2 bi-clusters in 2nd dataset but it failed to identify multiple bi-clusters in dataset 1. This is because bicbin identifies a bi-cluster and then replaces it’s contents in the dataset to 0’s. This makes it difficult to find the overlapping bi-clusters using bicbin. The Bibit algorithm is able to find the core of all the planted bi-clusters. However, it has outputted numerous small bi-clusters with all 1’s excluding the rows/columns that have a 0 entry.

### Seeds Dataset

As the 2nd pair of dataset’s we used the seeds dataset from UCI machine learning repository ([7]). It contains information about the geometrical properties of three varieties of wheat seed kernels. The real values are converted to binary data by doing a equal frequency quantization. Finally this dataset has size (35*210). The 2nd dataset of same size with 5% of 1’s in each row is created. Now, we took the subspace with
Chapter 6. Learning Relaxed 3-clusters from Pairs of Related Datasets

(A) Synthetic dataset 1  
(B) Synthetic dataset 2  
(C) Shuffled dataset 1  
(D) Shuffled dataset 2

FIGURE 6.4: Original and Shuffled synthetic Datasets

(A) Relaxed bi-clusters in Dataset 1  
(B) Relaxed bi-clusters in Dataset 2  
(C) 3-clusters identified

FIGURE 6.5: 3-clusters identified in Figure. 6.4
rows \{1, 6, 16 and 32\} in first dataset and embedded it in 2\textsuperscript{nd} dataset and shuffled the rows, columns in both the datasets. Our algorithm has successfully discovered all the relaxed bi-clusters that share the subspace \{1, 6, 16, 32\} from both the datasets. The two datasets and the results are shown in Figure 6.6.

### 6.4.2 Real Datasets

We considered the plants data [7] and the weather data [69] as one pair of related dataset’s. Here, we have three domains: plant names, state names, weather conditions. The plants dataset has 22,000 plant names and each row in the plants dataset represents a plant and it’s presence/absence in fifty states of USA where as the weather dataset has 25 weather conditions and each row in weather dataset represent a weather condition and it’s presence/absence in the states. A relaxed 3-cluster identified from this pair of datasets will give us an useful insight on the group of plants that grow in a set of states and their ideal weather parameters to grow. A relaxed 3-cluster found in the two data sets (‘plants vs. states’ and ‘weather vs. states’) is shown in Figure 6.7.

The first two columns represent the relaxed bi-cluster from plants dataset and the last two columns display the corresponding relaxed bi-cluster from weather dataset. The plant names shown in green color are grown in all the states in 2nd column except in the state of ‘Tennessee’ where as the plant name with yellow color is not grown in the state of ‘Georgia’. We can see that there is a high degree of overlap between the states that have a weather characteristics ‘temperature>59.50’ and ‘snow<=2.3’. From this we can infer that these are the ideal weather conditions for this plant. Also ‘Hawaii’, and ‘New mexico’ share the same weather conditions but these plants are not grown in...
Figure 6.7: A 3-cluster from the Plants-States-Weather Dataset’s

Table 6.1: Sample 3-clusters and their size variations with decrease in
alpha, beta values

<table>
<thead>
<tr>
<th>Plant names</th>
<th>States (In plants dataset)</th>
<th>States (In weather dataset)</th>
<th>Weather conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>'acer barbatum'</td>
<td>'Arkansas'</td>
<td>'Arkansas'</td>
<td>'temperature &gt; 59.50'</td>
</tr>
<tr>
<td>'aesculaceae indica'</td>
<td>'Florida'</td>
<td>'Florida'</td>
<td>'snow &lt;= 2.3000'</td>
</tr>
<tr>
<td>'aesculus pavia'</td>
<td>'Georgia'</td>
<td>'Georgia'</td>
<td></td>
</tr>
<tr>
<td>'aletris aurca'</td>
<td>'Louisiana'</td>
<td>'Louisiana'</td>
<td></td>
</tr>
<tr>
<td>'alium ampeloprasum'</td>
<td>'Mississippi'</td>
<td>'Mississippi'</td>
<td></td>
</tr>
<tr>
<td>'anemone berlandieri'</td>
<td>'South Carolina'</td>
<td>'South Carolina'</td>
<td></td>
</tr>
<tr>
<td>'arenaria lanuginosa'</td>
<td>'Texas'</td>
<td>'Texas'</td>
<td></td>
</tr>
<tr>
<td>'asclepias rubra'</td>
<td>'Alabama'</td>
<td>'Alabama'</td>
<td></td>
</tr>
<tr>
<td>'asimina parviflora'</td>
<td>'Virginia'</td>
<td>'Hawaii'</td>
<td></td>
</tr>
<tr>
<td>'axonopus'</td>
<td>'North Carolina'</td>
<td>'New Mexico'</td>
<td></td>
</tr>
<tr>
<td>'axonopus fucatus'</td>
<td>'Tennessee'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'berchemia'</td>
<td>'Oklahoma'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'berchemia scandens'</td>
<td>'Missouri'</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No.</th>
<th>Size</th>
<th>% ones</th>
<th>% change</th>
<th>Size</th>
<th>% ones</th>
<th>% change</th>
<th>Size</th>
<th>% ones</th>
<th>% change</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[21 X 3] [6 X 2]</td>
<td>(100 %)</td>
<td>112%</td>
<td>[29 X 3] [7 X 3]</td>
<td>(88.5 %)</td>
<td>144%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[24 X 3] [6 X 2]</td>
<td>(100 %)</td>
<td></td>
<td>[29 X 3] [7 X 3]</td>
<td>(76.1 %)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>[127 X 5] [6 X 3]</td>
<td>(99.3 %)</td>
<td>100%</td>
<td>[138 X 5] [8 X 3]</td>
<td>(97.3 %)</td>
<td>109%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[99.3 %] [100 %]</td>
<td></td>
<td></td>
<td>[99.3 %] [100 %]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>[9 X 16] [8 X 2]</td>
<td>(99.3 %)</td>
<td>100%</td>
<td>[9 X 18] [8 X 2]</td>
<td>(94.4 %)</td>
<td>111%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[99.3 %] [100 %]</td>
<td></td>
<td></td>
<td>[99.3 %] [100 %]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>[49 X 11] [7 X 2]</td>
<td>(99.5 %)</td>
<td>104%</td>
<td>[58 X 11] [8 X 2]</td>
<td>(89.2 %)</td>
<td>129%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[99.5 %] [100 %]</td>
<td></td>
<td></td>
<td>[99.5 %] [100 %]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

these states for some reason. We can suggest that these plants may grow well in these
two states if they are introduced’. These are all very valuable insights obtained by an
integrated analysis of the two datasets.

The $\alpha$, $\beta$ values are used to control the size, shape, and number of 1’s of the individual
relaxed bi-clusters in a 3-cluster. Table 6.1 show how these $\alpha$, $\beta$ values effect the indi-
vidual bi-clusters. From the 2nd 3-cluster ([127 X 5],[3 X 6]) we can notice that as we
decrease the parameter values, we are able to include few more 0’s into the plants bi-
cluster. With $\alpha$, $\beta$ values 0.95, both the bi-clusters have 4 states in common and has 127
plants but with $\alpha$, $\beta$ values 0.75, 11 more plants are added to the plants bi-clusters. Out
Chapter 6. Learning Relaxed 3-clusters from Pairs of Related Datasets

Table 6.2: BBC Dataset - Characteristics

<table>
<thead>
<tr>
<th>Category</th>
<th>Number of Documents</th>
<th>% ’1’s in dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business</td>
<td>510</td>
<td>6.8%</td>
</tr>
<tr>
<td>Entertainment</td>
<td>386</td>
<td>6.2%</td>
</tr>
<tr>
<td>Politics</td>
<td>417</td>
<td>8.2%</td>
</tr>
<tr>
<td>Sports</td>
<td>511</td>
<td>5.8%</td>
</tr>
<tr>
<td>Technology</td>
<td>401</td>
<td>8.3%</td>
</tr>
</tbody>
</table>

Table 6.3: Average bi-cluster size, One’s % in a 3-cluster of Plants-Weather Datasets

<table>
<thead>
<tr>
<th></th>
<th>0.65</th>
<th>0.75</th>
<th>0.85</th>
<th>0.95</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Algorithm</td>
<td>337 (76%)</td>
<td>314 (89%)</td>
<td>294 (97%)</td>
<td>287 (100%)</td>
</tr>
<tr>
<td>BicBin</td>
<td>298 (69%)</td>
<td>272 (84%)</td>
<td>262 (96%)</td>
<td>261 (100%)</td>
</tr>
</tbody>
</table>

of these 11 plants, 8 plants grow in 4 of the 5 states and these 4 states are overlapped to the states in weather dataset. These plants are not included in the 3-cluster generated for \( \alpha, \beta 0.95 \). With this we can show that the relaxed bi-clusters aid us to cluster the instances even if they highly share the attributes of that group. That is, if a particular plant is grown in 4 out of 5 states our relaxed bi-clusters will allow to group that plants with the plants that are grown in all 5 states.

We used the BBC Dataset [26] to continue our analysis on the proposed algorithm. The properties of the dataset are shown in Table 6.2. We used the categories - ‘Business vs Technology’ as a pair of documents, similarly ‘Entertainment vs. Sports’ as the other pair of documents. The goal is to find out the groups of words that are shared across the groups of documents in the selected categories.

BicBin and BiBit algorithm’s are used to compare the results from our proposed algorithm. The three table’s 6.3,6.4,6.5 show that we are able to find 3-clusters with much larger individual bi-cluster size. BiBit didn’t allow 0’s in the individual bi-clusters so it has a low value for average bi-cluster size. With BicBin we are able to produce individual bi-clusters of much larger size even with the same \( \alpha, \beta \) values. The problem with BicBin algorithm is that it is able to find the relaxed bi-clusters that are maximal in only one direction. After finding a relaxed bi-cluster it replaces the bi-cluster values with 0’s in the dataset. So it lose the ability to find all the overlapping bi-clusters.
Table 6.4: Average bi-cluster size, One’s % in a 3-cluster, BBC Dataset - Business, Technology

<table>
<thead>
<tr>
<th></th>
<th>0.65</th>
<th>0.75</th>
<th>0.85</th>
<th>0.95</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Algorithm</td>
<td>64 (68%)</td>
<td>57 (82%)</td>
<td>45 (93%)</td>
<td>43 (100%)</td>
</tr>
<tr>
<td>BicBin</td>
<td>45 (70%)</td>
<td>41 (79%)</td>
<td>36 (96%)</td>
<td>26 (100%)</td>
</tr>
</tbody>
</table>

Table 6.5: Average bi-cluster size, One’s % in a 3-cluster. BBC Dataset - Entertainment, Sports

<table>
<thead>
<tr>
<th></th>
<th>0.65</th>
<th>0.75</th>
<th>0.85</th>
<th>0.95</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Algorithm</td>
<td>48 (62%)</td>
<td>44 (76%)</td>
<td>42 (87%)</td>
<td>38 (100%)</td>
</tr>
<tr>
<td>BicBin</td>
<td>43 (67%)</td>
<td>41 (80%)</td>
<td>38 (92%)</td>
<td>34 (100%)</td>
</tr>
</tbody>
</table>

To analyze the effect of $\alpha, \beta$ on the shape of the individual bi-clusters in a 3-cluster, we computed the average size of rows, columns in the bi-clusters for different values of $\alpha, \beta$. These values for BBC - Entertainment & Sports dataset is shown in Table 6.6. We can observe that when $\alpha$ value is greater than $\beta$, (the row in which $\alpha$ is kept constant at 0.95 and $\beta$ value is changed from 0.65 - 0.95) the column size is more compared to the result obtained when $\alpha = \beta$. This is because when $\alpha$ is greater than $\beta$, given a row and a column with same size and % of 1’s, a column will be given more weightage compared to the row. This makes the search process to add more columns than the rows to the parent seed.

6.4.3 Execution Time Analysis

As our algorithm is heuristic search based, the run time of our algorithm is highly dependent on the density of the datasets. We compared the execution time of our algorithm with BicBin and BiBit algorithms and the results are shown in Figure 6.8. BiBit has a faster execution time on all the 3 datasets. However, it is only able to find

Table 6.6: Average bi-cluster size (rows, columns) in BBC Dataset - Entertainment, Sports

<table>
<thead>
<tr>
<th>$\alpha, \beta$</th>
<th>0.65</th>
<th>0.75</th>
<th>0.85</th>
<th>0.95</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha=\beta$</td>
<td>(14)(5)</td>
<td>(13)(5)</td>
<td>(9) (4)</td>
<td>(8) (4)</td>
</tr>
<tr>
<td>$\alpha=0.95$</td>
<td>(12)(11)</td>
<td>(9)(8)</td>
<td>(9) (5)</td>
<td>(8) (4)</td>
</tr>
<tr>
<td>$\beta=0.95$</td>
<td>(21)(5)</td>
<td>(18)(4)</td>
<td>(11) (4)</td>
<td>(8) (4)</td>
</tr>
</tbody>
</table>
3-clusters with strict one’s. In plants-weather dataset, BicBin and our algorithm has taken the same amount of time. The reason is that the plant occurrence is mostly in tune with weather conditions and not much pruning has taken place as almost every group of plants have corresponding states that share the weather conditions. Our algorithm executed faster compared to BicBin in both the BBC datasets. Compared to BBC {Entertainment,Sports} Dataset one can notice a significant time difference between BicBin and our algorithm in (Figure 6.8) the BBC {Business,Technology} Dataset. The reason is that BBC {Business,Technology} dataset has higher number of one’s compared to BBC {Entertainment,Sports}. As BicBin generates the bi-clusters first in individual datasets, it spent a lot of time in generating a lot of unwanted bi-clusters that don’t have a corresponding match in second dataset.

6.5 Conclusion

In this chapter, we had proposed a heuristic based search algorithm that finds 3-clusters from two related binary datasets. We used a coordinated A* search algorithm on multiple search spaces. Here the search is conducted using a single node and each of the datasets helps in evaluating the heuristic function. The score of the bi-cluster is given by its significance in it’s dataset and also in the 2nd dataset. With this only bi-clusters that are promising in both the datasets will be expanded. This helps us to avoid the expansion of unwanted bi-clusters in each of the individual datasets. The 2nd main contribution of our work is unlike the other 3-clustering algorithms which require the individual bi-cluster of a 3-cluster be all 1’s, we allow some 0’s in each of the individual bi-clusters of a 3-cluster. There are no other algorithm in literature that tries to capture
the 3-clusters whose notion is similar to us. With the help of multiple real dataset’s we had demonstrated that 3-clusters generated by our algorithm are much more meaningful. Even though we worked with a pair of related dataset’s (3 domains) to identify the 3-clusters, our approach can easily be extended to find N-clusters from multiple related dataset’s (N domains).
Chapter 7

Conclusion & Future Directions

7.1 Conclusion

The main focus of this dissertation is on designing various algorithms to uncover the hidden information from text document collections and to solve specific pattern discovery tasks. The analysis process usually involves first structuring the unstructured text data and then deriving useful patterns from this structured text data. Most of the problems we have solved has a broad range of applications in different domains. In email authorship analysis we have studied the usefulness of syntactic features like frequencies of grammar productions and show that these grammar features are also very effective in identifying the author of emails. In our second work we use information acquired from various types of features and proposed a consensus based methodology to assign the authorship of an test document. The proposed drug repurposing framework will use the biomedical concept co-occurrence information found in the Literature of the drug and disease related documents and find the likely drug repurposing candidates for a certain rare disease. Given a binary dataset, the concept of relaxed bi-cluster proposed in Chapter 5 will capture the strict linkages between the rows and columns and additionally help to predict the linkages that might be missing due to a background noise. The 3-clustering algorithm proposed will help to identify the hidden interactions between two binary related datasets. Given two binary dataset’s: ‘Domain 1 vs. Domain 2’, and ‘Domain 1 vs. Domain 3’, the proposed algorithm will efficiently find the associations between ‘domain 2‘ and ‘domain 3’ mediated by ‘domain 2‘. All the proposed algorithms are evaluated on multiple real datasets and various aspects of each
of the algorithm are studied thoroughly. Though these algorithms are proposed to address various text classification problems, some of the algorithms proposed can easily be adopted to other domains.

7.2 Future work

The proposed authorship methodology solves the problem of identifying the author of a document given a set of a training data from a small number of candidate authors. However, we might encounter situations where we have thousands of documents/emails with no prior knowledge on the authors of the documents. Here the goal will be to group the documents written by the same author with out any information about the authors. As we showed the reliability of grammatical features in our work, one good extension to perceive can be to use the proposed novel set of features in developing an unsupervised authorship clustering framework where we group the documents according to their author. Also one more extension can be to address the problem of identifying the smallest set of discriminating features needed to discriminate the author from the rest of authors.

The proposed drug repurposing framework uses the UMLS (Unified Medical Language System) concepts to create the drug and disease profiles. However, UMLS has limitations with gene and genomic annotation representations. So we should extend our approach to include more biomedical ontologies. Also, the literature on drugs and diseases is constantly updated so our approach should be extended to include the dynamic nature of the literature.

The notion of the relaxed bi-cluster we defined allows to include some 0’s in the bi-cluster’s. However, we are not able to control the placement of these 0’s in a bi-cluster. That is let’s say we have a bi-cluster of size (10*10) with 90% 1’s. Ideally we want the ‘10’ 0’s in the relaxed bi-cluster to be spread out over the rows/columns of it. But we might encounter some situations where all the ‘10’ 0’s be scattered only across one or two rows/columns. Our design of the heuristic/algorithm will not eliminate this possibility from happening. So one immediate extension is to design more sophisticated
heuristics which allow to control the number of 0’s in each row/column of a relaxed bi-cluster.

In the 3-clustering algorithm, till both the priority queue’s get empty we can’t guarantee that all qualified individual bi-clusters are generated. As the seeds are stored in priority queue according to their merit value, all the low promising seeds will be at the bottom of the priority queue. As the search process continue’s, the priority queue will be filled up with these low promising seeds. It will be ideal to occasionally prune these seeds. So one future work can be to design more pruning conditions to prune out these low promising seeds without being expanded.
Bibliography


[71] Srdan Verstovsek et al. “A phase 2 study of ruxolitinib, an oral JAK1 and JAK2 inhibitor, in patients with advanced polycythemia vera who are refractory or intolerant to hydroxyurea”. In: Cancer 120.4 (2014), pp. 513–520.


