COMPUTER-AIDED WRITEPRINT MODELLING FOR CYBERCRIME INVESTIGATIONS

MICHAEL SCHMID

A THESIS

IN

THE CONCORDIA INSTITUTE FOR INFORMATION SYSTEMS ENGINEERING

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF MASTER OF APPLIED SCIENCE IN INFORMATION SYSTEMS SECURITY

CONCORDIA UNIVERSITY

MONTRÉAL, QUÉBEC, CANADA

MAY 2012

© MICHAEL SCHMID, 2012

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Michael Schmid**

Entitled: **Computer-Aided Writeprint Modelling for Cybercrime Investigations**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science in Information Systems Security

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

	Dr. Nizar Bouguila	Chair
	Dr. Mohammad Mannan	Examiner
	Dr. Samar Abdi	External Examiner
	Dr. Benjamin Fung	Supervisor
Approved by	**Dr. Chadi Assi**	Graduate Program Director

_____ 20 _____

Dr. Robin A. L. Drew, Dean

Faculty of Engineering and Computer Science

Abstract

Computer-Aided Writeprint Modelling for Cybercrime Investigations

Michael Schmid

E-mail has become the most common way to communicate on the Internet, but e-mail security and privacy mechanisms are still lacking. This has proven to be a very valuable characteristic for criminals, who can easily take advantage of e-mail's various weaknesses to remain anonymous. Consequently, cybercrime investigators need to rely on computer-aided writeprint modelling methods and tools to identify the real author of malicious e-mails with transformed semantic content. In this paper, we propose a customized version of associative classification, a well-known data mining method, as well as a Support Count method, to address the authorship attribution problem. Experimental results on real-life data suggest that our proposed algorithms can achieve good classification accuracy on the e-mail author attribution problem through the use of writeprint modelling.

Acknowledgments

I would like to thank my supervisor, Dr. Benjamin Fung, for his many suggestions, enthusiasm and support over the last year. His guidance was indispensable throughout this process.

I must also show gratitude to my parents, who have paid for my education and without which I would not be here in the first place to live through the exciting times that is the world today.

To my father, Richard Schmid, my mother, Mary Schneider, my siblings and my fiance,

Chloe Boulay Parizeau.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The Internet has revolutionized the world of communications; many malicious entities engaging in illegal activities such as spam, ransom notes, threats, extortion, black mail and scams have moved their focus online. E-mail has become the standard way to communicate on the Internet, but e-mail security and privacy mechanisms are still lacking. In many cases of misuse, it is trivial for criminals to obfuscate their identify. Consequently, cybercrime investigators [11,12] need to rely on computer-aided writeprint modelling methods [28,48] and tools to identify the real author of malicious e-mails with transformed semantic content. Authorship attribution [58] is the statistical study of linguistic and computational characteristics of the written documents of individuals. This scientifically grounded process can identify the true author [22] of an anonymous textual communication without relying on the authenticity of e-mail header information.

There are three main security-related weaknesses inherent in e-mail systems. Although researchers have proposed security enhancements to remedy these problems, there is no standard that ensures compatibility and so the following issues are still prevalent.

1) E-mail meta data [36] and apparent routing information can be spoofed, making them

unreliable information to use as evidence. Even the origin of an e-mail can be hidden quite easily by using anonymous e-mail servers.

2) Computer viruses, Trojan scripts and other executables can be transmitted inside an e-mail. In fact, e-mails are one of the most common methods of distributing these types of malicious files, which can have devastating effects on systems.

3) The very nature of Internet e-mail access makes it difficult to link an actual person with the location the e-mail was sent from. Most e-mail systems are accessible from anywhere, whether it be from an Internet cafe or a public Wi-Fi access point. Even when it can be determined which computer sent a given message, computers are commonly used by multiple people. Botnets or other malicious executables can make it possible for an attacker to remotely send an e-mail using an intermediary's machine [13], leaving little to no trace of the real origin of the message.

E-mail misuses cannot effectively be prevented or bypassed by any current mechanism. However, it has been shown [32, 34] that the transformation of semantic content in e-mails for the purpose of writeprint modelling can help prosecute an offender by clearly linking them to a malicious e-mail with tangible supporting evidence.

1.2 Threat Model

Assume that an individual writes a threatening e-mail to a victim, but anonymizes the e-mail by obfuscating it's origin and other identifying information. A list of suspects can be drawn based on who might have negative feelings towards the victims. Armed with a warrant, the police may seize computers and equipment from the suspects' properties and extract all e-mails written by the various suspects. Assuming the author removed any trace of the threatening e-mail in question, an investigator must model the writeprints of each suspect and then determine which of the suspects is the most likely author. Whereas it is trivial and common to spoof e-mail header information, it is more difficult and less well

known to obfuscate a writing style. We propose that the process of matching a suspect's writeprint to malicious e-mails can help prosecute criminals by providing yet another solid piece of evidence to the case.

1.3 Problem

The problem of identifying the author of an anonymous e-mail in cyber forensics can be described as follows: a cyber forensic investigator is tasked with determining that the author of a given malicious e-mail m is likely to be one of the suspects $\{S_1, \ldots, S_n\}$. To support the finding in a court of law [11], the investigator must gather convincing evidence that identifies the most plausible author from the suspects $\{S_1, \ldots, S_n\}$. Whereas in forensic science, an individual's fingerprint can uniquely [23] identify them, in cyber forensic science, an investigator would like to use advanced data mining methods to model the writeprint of an individual from his/her e-mails and use it to support whether or not he/she is the author of a given message. Extracting a writeprint from transformed semantic content using computer systems is by far the most efficient and most accurate way of achieving this.

A significant amount of research has studied stylistic and structural features [19,51,58], but very few studies have considered the combination of features that form a writeprint. The writeprint of a given e-mail is the combination of features that define its lexical, syntactical, structural, semantic and content-specific attributes. Association rule discovery techniques can process a set of transformed e-mails written by the same author and then build an accurate classifier. A classifier consists of patterns that represent the respective author's most prominent combinations of features. The notion of frequently occurring combinations are used by these high performance and highly scalable techniques to extract interesting information that would be hard to discover otherwise.

1.4 Contributions of the Thesis

Associative classification [4] (AC) is a data mining method that uses association rule discovery techniques to build a classifier. To our knowledge, this is the first time AC has been applied to the authorship attribution problem and the results on real-life data suggest better accuracy and performance compared with existing techniques. These techniques result in a concise and representative classifier that can serve as clear evidence to support the identification of the true author of an e-mail.

There are many different flavors of AC, namely CBA [41], CPAR [30], CMAR [39], MCAR [52] and others. Given the need to quantify the match between various authors' writing styles and an anonymous e-mail, we have concentrated our research on CMAR, Classification by Multiple Association Rule [39]. This variation on AC uses a subset of rules as opposed to a single *best* rule, to determine which class, or author in our case, is the best match. We propose that authorship attribution, being a special case of classification, can benefit from a class-specific support threshold. This enables the classifier to better represent each author. Furthermore, we submit that a prioritization of complex patterns over general patterns in the rule pruning process makes for more accurate classification while still providing excellent performance enhancing characteristics.

We have also developed a new process called Support Count (SC) that is inspired by our contributions to CMAR. Support Count simplifies the classification process by examining features of an author's writing style as a lump sum score as opposed to discovering specific, but not necessarily useful patterns. This novel process is described in detail in Section 4.4.

A thorough set of experiments on the Enron [1] e-mail data set suggest that these methods are more accurate with similar or better performance than current state of the art authorship attribution methods.

1.5 Thesis Organization

The rest of the paper is organized as follows:

Chapter 2 provides a literature review on authorship analysis and classification.

Chapter 3 formally defines the authorship attribution problem and the notion of writeprint by class association rule (CAR) list.

Chapter 4 describes our new data mining approach for modelling a writeprint from transformed semantic content.

Chapter 5 evaluates the accuracy and efficiency of our proposed methods on the Enron e-mail data set [1].

Finally, Chapter 6 concludes the paper and outlines possible future research directions.

Chapter 2

Literature Review

2.1 Related Work

Previous works on authorship attribution are mostly applications of text classification analysis [18, 26, 42, 43]. Authorship analysis [5] is the study of linguistic and computational characteristics of the written documents of individuals [6, 8]. The process starts by extracting writing styles from an author's previously written texts. Each individual's writing style can then be used to differentiate one author from another [45]. Stylometric features [2] can be broadly categorized into five different types, namely lexical, syntactic, structural, content-specific, and idiosyncratic features [3, 32, 34].

In this section, we review the commonly employed stylometric features, summarize the techniques of e-mail authorship attribution found in authorship attribution literature and then discuss associative classification and its role in this study.

2.2 Writing Style Features

There is no standard predefined set of features [49] that best differentiates the writing styles of different suspects. However, studies [27] have identified the most representative features

in terms of accurately classifying an anonymous text. Punctuation and n-gram features have proven to be the most accurate, but the combination of these features was discovered to be even more powerful. Typically writing patterns contain the characteristics of word usage, composition and structure, common spelling and grammatical mistakes, words sequence, vocabulary richness, hyphenation and punctuation. One comprehensive study on stylistic features presented by Abbasi et al. [3] discusses this in more detail. The most common writing style features, namely, lexical, syntactical and structural are outlined in the following paragraph.

Textual measurements of both characters and words or tokens are called lexical features. The most relevant character measurement features are frequency of letters, number of characters per token and number of characters per sentence. Word-based lexical [50] features may include word length distribution, words per sentence and vocabulary richness. Initially, researchers thought that vocabulary richness [55, 56] and word usage [31] were discriminating features to be used for authorship attribution, but these have proven more useful in authorship characterization problems. Syntactic features include the distribution of function [57] words, such as *about, at, each, in, of, if, to*. Punctuation, another syntactic feature, plays a significant role in authorship attribution [9,24,53]. Paragraph structure and punctuation also have semantic content, measured as structural and syntactic features. Average paragraph length and number of paragraphs per document are commonly measured structural features. The presence of a sender signature [10] including contact information is a domain specific structural feature considered in the analysis of e-mail documents.

2.3 E-mail Authorship Analysis

Authorship analysis has resolved authorship attribution disputes over literary and conventional writings [44]. However, e-mail authorship attribution poses special challenges due to its characteristics of size, vocabulary and composition when compared to literary

works [20]. Literary documents are usually large in size, comprising of at least several paragraphs; they have a definite syntactic and semantic structure. In contrast, e-mails are short and usually do not follow well defined syntactic or grammatical rules. Thus, it is harder to model the writing patterns of their author. Ledger et al. [37] established that authorship analysis results would not be significant for texts containing less than 500 words, creating the need for better models [32, 34] able to handle the characteristics inherent in e-mails. Moreover, e-mails are more interactive and informal in style, and people are not as conscious about spelling and grammatical mistakes particularly in informal communications. Therefore, techniques that are successful in literary and traditional works are not always applicable in e-mail authorship attribution problems.

Iqbal et al. [33] have shown that the e-mail authorship attribution problem deserves special attention and can be solved by designing algorithms that deal with the specific challenges related to e-mail authorship analysis. Our research differs from that work by applying a popular data mining technique called associative classification whereas Iqbal et al. [33] concentrated on frequent item sets. Our proposed method boasts improved classification accuracy and performance, as will be shown in detail in Chapter 5.

A popular classification method, the Support Vector Machine [7, 16, 21, 35] (SVM), was applied [18, 51] over a set of stylistic and structural features for e-mail authorship attribution. de Vel et al. [15, 19] performed extensive experiments and concluded that classification accuracy decreases when the training data set size decreases, when the number of authors increases, or the average length of documents decreases. This helps to explain the decline in classification accuracy seen when processing documents with e-mail-like characteristics. de Vel et al. [18] further found that the performance of SVM was reduced by increasing the number of function words from 122 to 320, contradicting the argument that SVM supports high dimensionality and leading to the conclusion that adding more features

does not necessarily improve accuracy. However, it has been proposed [32, 34] that identifying combinations of key features that are able to differentiate between writing styles of different suspects and filtering out useless features can improve accuracy.

Generally each type of the four feature sets are applied independently, which sometimes produces conflicting results [18]. For example, word usage and composition style may vary from one structural pattern to another. Previous authorship attribution techniques also suffered from the problem of having too many features, making it difficult to determine the right feature sets to use for a given set of e-mails. de Vel et al. [18] have shown that adding useless features may decrease classification accuracy when a classifier captures these features as noise. Using noisy features for classification also diminishes the justification of evidence for supporting the finding, creating a legal problem from a technical problem. One of our approaches overcomes this limitation by flexibly extracting the evidence (combinations of frequently occurring features) from the data itself, automatically filtering out noise with user-supplied thresholds that are not content or domain specific. The default values of these thresholds are sufficient but there is great value in allowing the user to set values based on the specific nature of a given problem.

2.4 Associative Classification

Let $A = \{A_1, \ldots, A_m\}$ be a set of features. A record r is a set of feature items $\{a_1, \ldots, a_m\}$ where $a_i \in A_i$. Features can be categorical or continuous and we assume that all possible feature items are mapped to a set of consecutive positive integers. To map continuous features into consecutive positive integers, their ranges are discretized into intervals. By doing so, all features are treated uniformly and can therefore appropriately factor into the frequent pattern set discovery process. Since these features represent a value of a writing style feature, their combined effect makes it simple and intuitive to model an author's writeprint.

Let C be a finite set of distinct class labels, each representing an author in our context. A training data set is a set of records, each with an associated class label $c \in C$. A classifier L is a function that maps a record r to a class $c \in C$.

Associative classification (AC) is an association rule discovery process in which only one class is considered as a rule's consequent; in any given rule such as $A \rightarrow B$, B must always be a class and B will never be in any rule's antecedent. Therefore, the intersection of the set of feature items and class labels is always empty, $A \cap C = \emptyset$.

The task of classification in general is to build a classifier from a training data set to accurately classify each test record from a test data set. There are many different approaches used for classification, such as decision tree [47], naive Bayesian [25, 45, 46], neural network [40], etc. A more recent approach is to explore strong relationships between specific sets of object features and their class labels; frequent patterns in records with the same class label can then be used to infer the class of other records with similar patterns. The important advantage in using AC over classical classification approaches is that the output of an AC algorithm is represented by simple If-Then rules which are easy and intuitive to understand and interpret.

Definition 1 (Match)**.** Let $p = \{a_1, \ldots, a_k\}$ be a set of distinct feature items called a *pattern*, where a_j is a feature item of some attribute $A_i \in A$. A record r *matches* a pattern p if $p \subseteq r$. ■

Definition 2 (Support)**.** Given a training data set T, let c be a class label. For rule $R : P \rightarrow c$, the number of records in T matching pattern p and having class label c is called the support of R, denoted as $sup(R)$. ■

Definition 3 (Confidence)**.** The ratio of the number of records matching pattern p with a class label c versus the total number of records matching that pattern p is called the confidence of R, denoted as $conf(R)$. ■

For example, if 90% of suspect S_i's e-mails contain 3 paragraphs, then the confidence of rule $R : 3\ paragraphs \rightarrow S_i$ is 90%. We can use this rule to classify future records that match this pattern. The support threshold is used to avoid noise; without a minimum support threshold, even non-representative relationships might become rules. Typically AC finds the complete set of class association rules (CAR) that pass the user-supplied minimum support and confidence thresholds. When a new record requires classification, the classifier will select the matching rule with the highest confidence and support and use it to predict the class label. Newer AC techniques will prune and rank rules and sometimes even use multiple rules to predict the class label of an unknown record as there are situations in which the single best rule may not be the most intuitive or even most appropriate choice. Many studies [30, 39, 41, 52] show that AC is intuitive, efficient and effective.

Authorship attribution requires special attention when it comes to using AC techniques to obtain the best results; with multiple distinct classes and the need to consider much more than simply the strongest class, it becomes evident that a classifier should consider as much information as possible.

Example 1 demonstrates why a single matching rule may not always be the best choice.

***Example* 1.** Suppose we want to determine who the author of an anonymous e-mail with feature items (2, 5, 8) is. The top 3 most confident rules matching the e-mail are as follows:

Rule R1: 2 $\rightarrow$ Author 0 (support: 100, confidence: 90%)

Rule R2: 5 $\rightarrow$ Author 1 (support: 200, confidence: 89%)

Rule R3: 8 $\rightarrow$ Author 1 (support: 150, confidence: 88%)

Most AC techniques that select the most confident rule would classify this e-mail as belonging to Author 0, but a closer look suggests that this decision has been made with little regard to the rest of the rule list. All three rules have similar confidence but both $R2$ and $R3$ have higher support, which means that the values of those features were found more often in the training data set for Author 1. Author 1 is therefore a more intuitive choice and

our algorithm should reflect that. ■

Situations like this make it clear that in order to make a reliable and accurate prediction, especially when the result could mean the difference between *guilty* and *innocent*, an aggregate measure analysis based on multiple rules intuitively and reliably leads to better quality classification, as shown in the literature [39].

Chapter 3

The Problem

Let $\{S_1, \ldots, S_n\}$ be the set of suspected authors of an anonymous e-mail e and let E_i be a collection of e-mails, consisting of at least 30 messages, written by suspect $S_i \in \{S_1, \ldots, S_n\}$. The problem of authorship attribution by multiple class association rule is to identify the most plausible author S_a from the suspects $\{S_1, \ldots, S_n\}$, whose collection of e-mails E_a best matches with the feature items in the malicious e-mail e. Intuitively, a collection of e-mails E_i matches e if E_i and e share similar patterns of writing style features in strongly representative combinations. The primary objective of cyber forensic investigators is to automatically and efficiently model the patterns, or writeprint, of each suspect. They can then present such patterns as evidence identifying the author of the malicious e-mail e.

In terms of associative classification, what exactly is suspect S_i's writeprint? Specifically, we want to extract [54] rules derived from patterns that uniquely and strongly represent the writing style of each suspect S_i, but do not represent the writing style of any other suspect S_j, where $i \neq j$. In the rest of this section, we discuss the pre-processing of e-mails and formally define the notions of frequent patterns, classification rules and the problem of accurately quantifying the similarity between each set E_i and an anonymous e-mail e.

3.1 Pre-processing

Each author has a folder containing his/her e-mails, with headers and appended *forward* or *reply* content removed. The selection of e-mails has been performed manually because, unlike header information, the body section of an e-mail is unstructured and must be evaluated on a case by case basis. E-mails with less than a few sentences or attached content are not included, as they would not represent an author's writing style. Once the data has been vetted, pre-processing [17] can begin.

Let E_i be a collection of e-mails written by suspect S_i. The goal of pre-processing is to output a set of records listing the feature values for each e-mail in E_i. In the rest of this section, the term *feature* refers to one of the many stylometric features described in Section 2.2.

We discretize each normalized feature into a set of intervals, for example, $[0-0.25]$, $(0.25-0.5]$, $(0.5-0.75]$, $(0.75-1]$. Each interval is called a feature item. The normalized feature frequency is then matched with these intervals. We assign a value of 1 to the feature item if the interval contains the normalized feature frequency; otherwise we assign a value of 0. This simplifies the procedure by determining the presence or absence of a pattern. Some commonly used discretization techniques are:

- *Equal-width discretization*, where the range of each interval is equivalent.
- *Equal-frequency discretization*, where each interval contains approximately the same number of records.
- *Clustering-based discretization*, where clustering is performed on the distance between neighbouring points.

The discretization technique used in our study is called *equal-frequency discretization*, where each interval contains approximately the same number of records. This method

Table 1: Feature Vectors

E-mails	Features							
	Feature X			Feature Y		Feature Z		
	X1	X2	X3	Y1	Y2	Z1	Z2	Z3
e1	1	0	0	1	0	1	0	0
e2	0	1	0	0	1	0	1	0
e3	0	1	0	1	0	0	0	1
e4	1	0	0	1	0	0	1	0
e5	0	1	0	0	1	0	1	0
e6	0	1	0	0	1	0	0	1
e7	0	0	1	1	0	0	1	0
e8	0	0	1	0	1	0	1	0
e9	1	0	0	0	1	1	0	0
e10	0	1	0	1	0	1	0	0

was chosen because it allows for each interval of the feature vector to be split according to the training data. This ensure that each feature item is as distinct as possible from the other authors.

Example 2 demonstrates the typical pre-processing algorithm.

***Example* 2.** Consider the ten e-mails in Table 1 from which we extract three features $\{X, Y, Z\}$. We first discretize each feature into feature items. For example, a stylometric feature X having a normalized range of $[0, 1]$ can be discretized into three intervals $X1 = [0, 0.33], X2 = (0.33, 0.66]$ and $X3 = (0.66, 0.1]$, representing three feature items. Similarly, feature Y is discretized into $Y1 = [0, 0.5]$ and $Y2 = (0.5, 1]$, and feature Z into $Z1 = [0, 0.33], Z2 = (0.33, 0.66]$ and $Z3 = (0.66, 01]$. An e-mail e_1 having features $Z = 0.3, Y = 0.25$, and $Z = 0.25$ can be represented by the following feature vector: $\langle 1, 0, 0, 1, 0, 1, 0, 0\rangle$. ■

3.2 Frequent Patterns

Intuitively, the writing style of a collection of e-mails E_i written by suspect S_i is a combination of feature items that frequently occur in the training set of e-mails E_i. These frequently occurring patterns are modelled with the concept of *frequent itemset* [4] described as follows.

Let $V = \{f_1, \ldots, f_n\}$ denote the universe of all feature items. Let E_i be a set of e-mails where each e-mail e is represented by a set of feature items such that $e \subseteq V$. An e-mail e contains a feature item f_i if the numerical feature value of the e-mail e falls within the interval of f_i. For example, e-mail e_1 in Table 1 can be represented by a set of feature items $e_1 = \{X1, Y1, Z1\}$. Table 2 lists the ten e-mails from Table 1 in feature item set notation.

Let $F \subseteq V$ be a set of feature items called a pattern. An e-mail e contains a pattern F if $F \subseteq e$. Let us assume that a pattern that contains k feature items is called a k-pattern. For example, the pattern $F = \{f1, f4, f6, f8\}$ is a 4-pattern and $F = \{f2, f5\}$ is a 2-pattern. The support of a pattern F, with regard to a single author, is the number of e-mails in E_i that contain F. If the support of F is greater than or equal to a user-specified minimum support threshold, then F is considered a frequent pattern in the set of e-mails E_i.

Definition 4. *(Frequent pattern).* Let E_i be the set of e-mails written by suspect S_i. Let $sup(F|E_i)$ be the number of e-mails in E_i that contain the pattern F, where $F \subseteq V$. A pattern F is a frequent pattern in E_i if $sup(F|Ei) \geq min\_sup$, where the minimum support threshold min_sup is a real number representing the minimum number of times a pattern must appear or a minimum percentage of records that contain the pattern. ■

The writing style of a suspect S_i is therefore represented as a set of frequent patterns, denoted by $FP(E_i) = \{F_1, \ldots, F_k\}$, extracted from the set of e-mails E_i. These patterns are used to derive a high quality class association rule list that consists of the strongest frequent patterns by means of pruning and ranking. The pruning and ranking of rules differs from one technique to another. These differences will be outlined in Chapter 4.

Table 2: Feature Items

E-mail	Items
e1	{X1, Y1, Z1}
e2	{X2, Y2, Z3}
e3	{X2, Y2, Z3}
e4	{X1, Y1, Z2}
e5	{X2, Y2, Z3}
e6	{X2, Y2, Z3}
e7	{X3, Y1, Z2}
e8	{X3, Y1, Z2}
e9	{X1, Y1, Z1}
e10	{X2, Y2, Z2}

***Example* 3.** Consider Table 1. Suppose the user-specified threshold $min\_sup = 0.3$, which means that a pattern $F = \{f_1, \ldots, f_k\}$ is frequent if at least three out of the ten e-mails contain all feature items in F. $\{X3\}$ is not a frequent pattern because it has support 2/10 = 0.2. $\{X2\}$ is a 1-frequent pattern because it has support 0.5 and $\{Y2\}$ is also a 1-frequent pattern with support 0.5. $\{X2, Y2\}$ is a 2-frequent pattern as $X2$ and $Y2$ appear together five times. Finally we have a 3-frequent pattern with $\{X2, Y2, Z3\}$ with a support of 0.4. Example 4 will show how to efficiently compute all frequent patterns. ■

3.3 Associative Classification Writeprint

Fingerprint identification in forensic science, known as dactyloscopy, is the process of comparing two instances of friction ridge skin impressions to determine whether these impressions could have come from the same individual. In authorship attribution applied by cyber forensic specialists, we can do something similar by identifying the writeprint of an individual from his/her written text as a distinguishable writing style. Writeprints, as described in this thesis, can not uniquely tell apart every individual in the world, but a properly identified writeprint is accurate enough to identify the writing pattern of an individual. It is our prerogative to show that the author of an anonymous text can be determined given a

list of suspects and a sufficient quantity of their previously written texts.

The notion of frequent patterns in Definition 4 captures a suspect's writeprint. However, two suspects S_i and S_j may share some similar writing patterns. Therefore it is helpful to filter out common frequent patterns and retain only the frequent patterns that are unique to each suspect. Setting a confidence threshold in typical associative classification (AC) techniques does something similar, by discarding patterns that are not mostly associated with a single class. This leads us to the notion of writeprint: a writing style that strongly represents an individual.

Once a set of class association rules (CAR) is discovered, it is trivial to remove common rules and group them into a list for each author. These grouped rules consist of the strongest unique patterns that make up each author's writing style. In cases where there are many authors, AC may not create enough rules to represent every one. Whereas the goal of authorship attribution is to most accurately associate an anonymous e-mail with its most probable author, the goal of classification is simply to associate a given feature set with the *best class*. This notion of *best class* is not always compatible with the goals of authorship attribution. For example, AC is most often used to predict the outcome of some situation, whereas authorship attribution attempts to associate an author with his unique writing style. Therefore, some classes may be neglected if their frequent patterns do not have enough support or confidence, or if the training data set is too small. This is not an acceptable situation in authorship analysis as each author should have an equally representative writeprint if an e-mail is to be classified correctly in cases where there are rules with low support and/or confidence for some suspect.

Many studies have presented ways of greatly reducing the number of class association rules for reasons of efficiency, given that usually only the strongest rule would be used for classification anyway. Our approach uses multiple rules [39] and so it is important not to prune or discard too much information. In general, rules with low support and confidence

are pruned or outranked by more powerful rules, regardless of their class association. This means that a given author may be assigned to an unknown e-mail simply because he/she has a stronger writeprint than the true author, and not based on a normalized measure of similarity. This would be the equivalent of identifying a matching fingerprint against two samples: one with a full print and another just with a partial print. The full print has more potential to match or to mismatch the unknown print, whereas the partial sample, even if it matches the unknown print very well, could still be discarded as its potential to fully match the unknown print is inherently lower.

Definition 5. A writeprint, denoted by $WPCAR(E_i)$, is a set of patterns where each pattern F has $sup(F|E_i) > min\_sup$ and $sup(F|E_j) < min\_sup$ for any E_j where $i \neq j$, min_sup is a user-specified minimum threshold. In other words, $WPCAR(E_i) \subseteq FP(E_i)$, and $WPCAR(E_i) \cap WPCAR(E_j) = \emptyset$ for any $1 \leq i, j \leq n$ and $i \neq j$.

A CAR list encapsulates the writeprint of each author and as such is different from the typical notion of writeprint.

The first distinction is that the combination of feature items that composes the writeprint of a suspect S_i is dynamically generated based on the embedded pattern in e-mails E_i. This flexibility allows us to succinctly model the writeprint of different suspects by using different combinations of feature items. In contrast, the traditional notion of writeprint considers one feature at a time without considering combinations thereof.

Second, every rule R in our notion of writeprint captures a writing pattern that can only be found in one suspect's collection of e-mails but not in any other suspect's collection of e-mails. The cyber forensic investigator could precisely point out a matched pattern in the malicious e-mail to support his/her conclusion of authorship identification. In contrast, a traditional classifier like decision tree could use the same set of features to capture the writeprint of different suspects. It is quite possible that the classifier would capture some common writing patterns and use them as evidence that points to multiple authors, drawing

a conclusion based on ambiguous evidence. Our notion of writeprint avoids this ambiguity and, therefore, provides more convincing and reliable evidence.

The removal of common patterns certainly improves the quality of the derived writeprint, especially for the purpose of evidence collection. However, one must understand the advantages as well as the disadvantages inherent in this technique. If there are a large number of suspects, it is entirely possible for one author's writeprint to completely intersect with the union of the other authors' writeprints, leaving the first author without any writeprint at all. This could happen if the set of common rules is equivalent to the total set of rules for one class.

A unique writeprint would intuitively derive more accurate matches and provide better evidence, however, there are some situations where discarding common patterns between authors could be detrimental to classification accuracy. Furthermore, the common pattern removal process is not particularly efficient, as every rule must be checked against every other rule; this notion makes for poor scalability and is readily apparent in the run-time numbers when AuthorMiner [32] is run with more than 3 authors.

3.4 Refined Problem Statement

The problem of authorship attribution by multiple class association rules can be refined into three subproblems:

(1) to discover the writeprint class association rule list $WPCAR(E_i)$ from the training set of e-mails $E_i \in \{E_1, \ldots, E_n\}$.

(2) To determine the author of the malicious e-mail e by classifying $WPCAR(E_i)$, $\ldots, WPCAR(E_n)$ using all matching rules.

(3) To extract evidence for supporting the conclusion on authorship. The evidence has to be intuitive enough to convince a judge and jury in a court of law.

These three subproblems summarize the challenges in a typical investigation procedure

and reflect the use of associative classification in this process.

To solve subproblems (1) and (2), we mine rules by extracting the set of frequent patterns $FP(E_i)$ from E_i all while ranking and pruning them to build a representative final CAR list. For subproblem (3), the matching group of rules with the best score serves as evidence for supporting the conclusion.

Chapter 4

Authorship Classification Explained

In Section 4.1 to 4.3, we present a data mining approach that utilizes the concept of frequent stylometric patterns and associative classification (AC) to address the three authorship analysis problems described in Section 3.4. Section 4.4 introduces a novel associative classification inspired algorithm called Support Count.

4.1 Mining Classification Rules

A class association rule (CAR) list is compiled by mining a training data set to find the complete set of rules passing user-supplied minimum support and confidence thresholds. This is comparable to any frequent pattern mining or association rule mining task. Classification by Multiple Association Rule (CMAR) [39] and its implementation [14] by Frans Coenen form the basis of the two AC methods described in this study. The algorithm we use to mine rules is a variant of FP-growth [29]: the most scalable and efficient mining algorithm, both in terms of performance and memory usage. Making use of efficient tree structures [14], first a partial support tree and then a total support tree, database scans are minimized and there is no need to generate candidate sets. The benefits of this method are especially apparent when processing large data sets with a low support threshold and a large

Table 3: Training Records

Row	*Feature A*	*Feature B*	*Feature C*	*Feature D*	*Author ID*
1	*a2*	*b1*	*c2*	*d1*	*A*
2	*a1*	*b1*	*c2*	*d3*	*B*
3	*a3*	*b2*	*c1*	*d2*	*A*
4	*a1*	*b1*	*c3*	*d3*	*C*
5	*a1*	*b1*	*c2*	*d3*	*C*

number of features. This situation is commonly seen in authorship attribution problems. Furthermore, accuracy is generally better when low support and confidence thresholds are used, making this choice of algorithm a fitting decision.

Example 4 demonstrates the concept of class association rule mining.

***Example* 4.** Let T be a training data set as shown in Table 3. Setting the support threshold to 2 and confidence to 50%, the algorithm extracts class association rules as follows.

1) The training set T is scanned, retrieving the set of feature items that pass the minimum support threshold . The set $F = \{a_1, b_1, c_2, d_3\}$ is called a frequent item set, as each element in the set appears at least twice. All other feature items appear only once and are pruned.

2) The feature items in F are then sorted in descending order to become $F = \{b_1, a_1, c_2, d_3\}$. The database is then scanned again to construct an FP-tree as shown in Figure 1a. A FP-tree is a prefix tree with regard to the F-list. For each tuple in the training data set, feature items appearing in the F-list are extracted and sorted accordingly. For example, for the first tuple, (b_1, a_1) are extracted and inserted in the tree as the left-most branch. The author ID is attached to the last node in the path. For efficiency, tuples in the training data set share prefixes. For example, the second tuple carries feature items (b_1, c_2, a_1) in the F-list and shares a common prefix b_1 with the first tuple. The sub-path with the left-most branch is therefore shared. All nodes with the same feature items are linked together as a queue starting from the header table.

3) The set of class association rules based on the F-List can be divided into 4 subsets

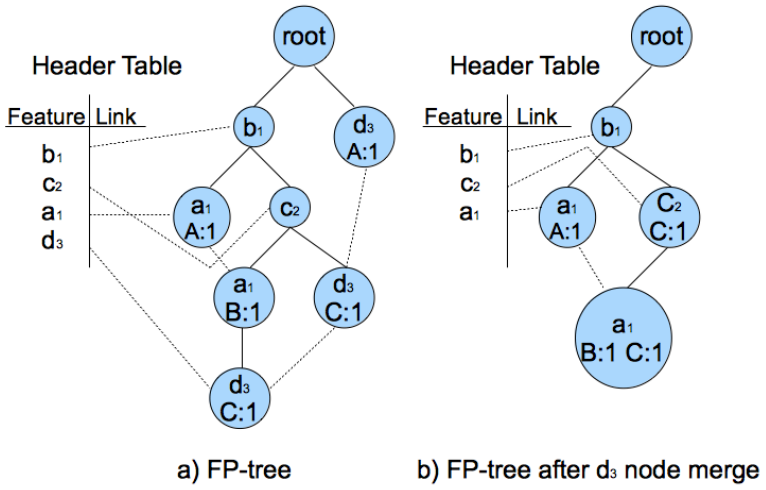

Figure 1: FP-Tree

without overlap:

-those with d_3

-those with a_1 but not d_3

-those with c_2 but not d_3 or a_1

-those with b_1 exclusively

These subsets are discovered iteratively one at a time.

4) Finding the subset of rules having d_3, the algorithm traverses nodes having feature item d_3 and looks upwards to construct a d_3-projected database, which contains three tuples: $\{b_1, c_2, a_1, d_3\} : C, \{b_1, c_2, d_3\} : C$ and $d_3 : A$. Given that all tuples containing d_3 are included, the problem of finding all frequent patterns with d_3 in the entire training set can be simplified to mining patterns in our d_3-projected database. Passing the support threshold, b_1 and c_2 are frequent feature items in the d_3-projected database. d_3 does not count as a local frequent attribute because in a d_3-projected database, d_3 is present in every tuple and therefore is trivially frequent. The projected database can be mined recursively by constructing FP-trees and other projected databases, as described in detail by Han et al. [29]. In our d_3-projected database, b_1 and c_2 always appear together, they are both sub-patterns

of b_1c_2 and have the same support count as b_1c_2. The rule $R : b_1c_2d_3 \rightarrow C$ with support 2 and confidence 100% is generated based on author distribution. After processing all rules that include d_3, those nodes can be merged into their parent nodes. This means that any class association registered in any d_3 node is registered with its parent node, effectively shrinking the FP-Tree to what is shown in Figure 1b. This operation is performed while the d_3-projected database is built. This process is then repeated for the remaining subsets of rules. ■

4.2 Pruning Classification Rules

Class association rule mining can generate an enormous number of rules; it is advantageous and rather simple to prune redundant or noisy rules in order to build a concise yet high quality classifier.

The associative classification variant [39] used in this study employs the same rule ordering protocol as described by Liu et al. [41]. The final rule list will thus be ordered according to three ranking rules.

Given two rules $R1$ and $R2$, $R1$ has a higher rank than $R2$, denoted $R1 > R2$, if the following conditions are met:

1) $conf(R1) > conf(R2)$

2) $conf(R1) = conf(R2)$ but $sup(R1) > sup(R2)$

3) $conf(R1) = conf(R2)$ and $sup(R1) = sup(R2)$ but $|ant(R1)| < |ant(R2)|$

Rule $R1 : P \rightarrow c$ is said to be a general rule with regard to rule $R2 : P' \rightarrow c'$, if and only if $P \subseteq P'$.

The first round of pruning uses general and high-confidence rules to prune more specific and lower confidence rules. Given two rules $R1$ and $R2$, where $R1$ is a general rule with regard to $R2$. CMAR [39] prunes $R2$ if $R1$ also has a higher rank than $R2$. The rationale is that we only need to consider general rules with high confidence, and thus more specific

rules with low confidence should be pruned. However, we will see that this is not ideal behaviour in an authorship attribution problem.

Rule $R1 : P \rightarrow c$ is said to be specific with regard to rule $R2 : P' \rightarrow c'$, if and only if $P \supseteq P'$.

While in general this pruning is harmless to accuracy and effective at making the process more efficient, one of our contributions is to prioritize more specific rules rather then more general rules. Therefore CMARAA orders rules slightly differently, changing condition 3) above to:

3) $conf(R1) = conf(R2)$ and $sup(R1) = sup(R2)$ but $|ant(R1)| > |ant(R2)|$

Part of the first round of pruning in CMARAA is therefore the opposite of what is done in CMAR [39]. More specific rules with higher ranking are selected over more ambiguous rules. Intuitively the writing style patterns of an author should be as precise as possible in order to most accurately represent their more frequently occurring textual measurements. This change, in concert with other contributions that define CMARAA, allows the orithm to achieve better results in terms of classification accuracy.

More general and more specific rule pruning is pursued when the rule is first inserted into the classification rule (CR) tree. When this happens, retrieval over the tree is triggered to check if the rule can be pruned or if it can prune other rules that are already inserted. Our experimental results suggest that this pruning is effective.

The second round of pruning is done by selecting only positively correlated rules. For each rule, we test whether P is positively correlated with c by chi square testing. Only the rules that are positively correlated, i.e., those with a value passing a significance level threshold, are used for classification. All rules that fail the correlation test are pruned.

Chi square correlation based pruning is used to reflect only strong implications to perform classification. By removing rules that are not positively correlated, we reduce noise and make the classification process more efficient without negatively affecting accuracy.

This pruning is done when a rule is being inserted into the CR-tree since the values necessary for performing the chi square test are readily available at this point, making this process very efficient with negligible cost in terms of run-time. The specifics of chi square testing are outside the scope of this paper.

The third pruning method selects a subset of high quality rules for classification by pruning rules based on database coverage. A database coverage threshold [38] is used to reduce the number of CAR's significantly, while maintaining the same representative number of rules per training record. This process is described in Algorithm 1.

Algorithm 1 Database coverage rule based selection

Input: a list of rules and a database coverage threshold τ
Output: a concise but representative subset of class association rules
Protocol:
1. Order rules by rank;
2. For each training record, set the cover-count to zero;
3. For each rule R, find all matching training records. If rule R can appropriately classify at least one record, increase the cover-count of all records matching rule R by one. Remove a training records once its cover-count passes the database coverage threshold τ.

The database coverage method used by Li et al. [39] is similar to the one used by Liu et al. [41]. The major difference is that, instead of removing one data object from the training data set immediately after it is covered by some selected rule, it is left in the training set until it is covered by at least three rules. The effect of this difference is that there will be more rules to consult when classifying a new data object and therefore the unknown object will have a better chance of being classified accurately.

This pruning is pursued when the rule mining process is complete and it is the last pruning of rules described in CMAR [39].

One of this thesis' contributions is the addition of another round of pruning for CMARAA. This last pruning method has been brought over from Iqbal et al. [32,34], and is called common frequent item set elimination. When rules are being inserted into the CR-tree, any rule

with the same antecedent as another distinct rule with a different class is flagged for removal. Once the CR-tree is processed, the flagged rules are removed. The reason that common rules are not removed immediately once discovered is that another rule for another author that is also common may also exist. It is necessary therefore to leave all rules in place until the process of generating all CAR's is complete.

4.3 Authorship Classification

Once a set of rules is discovered and pruned for classification, as discussed in Sections 4.1 and 4.2, we are ready to classify anonymous e-mails. Given a test record, we collect the subset of matching rules from the CAR list. The rest of this section outlines how to determine the class label based on this subset of rules.

If all the rules matching the new object have the same class label, the test record is classified without contest.

If there exist two or more rules with different class labels, we create groups of rules for each class. All rules in a group share the same class label and each group has a distinct label. We then compare the strength of each group and associate the record with the strongest one.

To compare the strength of groups, we need to measure the combined effect of each group. Intuitively, if the rules in a group are highly positively correlated and have good support, the group should have a strong effect.

Liu et al. [39] performed an empirical study to determine the most accurate way to measure the strength of groups. Typical AC algorithms use the strongest rule as a representative, which means that the single rule with the highest rank is selected. The danger of simply choosing the rule with the highest rank is that this may be favorable to minority classes, as illustrated by Example 5.

***Example* 5.** In an authorship attribution exercise, there are two rules:

(a) R1 Observed

R1	Author A	Author B	Total
Feature A	410	40	450
No Feature A	20	30	50
Total	430	70	500

(b) R2 Observed

R2	Author A	Author B	Total
Feature B	209	1	210
No Feature B	241	49	290
Total	450	50	500

(c) R1 Expected

R1	Author A	Author B	Total
Feature A	387	63	450
No Feature A	43	7	50
Total	430	70	500

(d) R2 Expected

R2	Author A	Author B	Total
Feature B	189	21	210
No Feature B	261	29	290
Total	450	50	500

Table 4: Observed and Expected Values

$R1 : FeatureA = no \rightarrow AuthorB(support = 450, confidence = 60\%)$

$R2 : FeatureB = yes \rightarrow AuthorA(support = 200, confidence = 99.5\%)$

See observed and expected values for rules $R1$ and $R2$ in Table 4.

Based on the observed and expected values, the chi square value is 97.6 for $R1$ and 36.5 for $R2$. For an anonymous email with no feature A and feature B, we may predict that the author would be Author B rule $R1$, if the choice between rules is based only on chi square values. However, rule $R2$ is intuitively much better than rule $R1$ since rule $R2$ has much higher confidence. This presents a challenge in determining which rule is the strongest. ■

Using the compound of correlation of rules as a measure is one alternative. For example, we can sum up the values in a group as the strength measure of the group, but this would suffer from the same problem that it may favor minority classes.

A better way would be to integrate both correlation and popularity into the group measure and in this spirit, we have adopted a weighted measure [38] called Weighted Chi Square (WCS). For each rule $R : P \rightarrow c$, let $sup(c)$ be the number of records in the training data set that are associated with class label c and let $|T|$ be the number of data records in the entire training data set. Equation 1 defines the max chi square value, used as the upper bound of the chi square value of the rule.

$$max\chi^2 = (min(sup(P), sup(c)) - \frac{sup(P)sup(c)}{|T|})^2|T|e \tag{1}$$

where

$$e = \frac{1}{sup(P)sup(c)} + \frac{1}{sup(P)(|T| - sup(c))} + \frac{1}{(|T| - sup(P)sup(c))} + \frac{1}{(|T| - sup(P))(|T| - sup(c))}$$

For each group of rules, the weighted measure of the group is calculated using Equation 2.

$$\sum \frac{(\chi^2)^2}{max\chi^2} \tag{2}$$

As demonstrated, the ratio of the chi square value against its upper bound, max chi square, is used to overcome the bias of the chi square value favoring any minority class. Liu et al. [39] noted that it was difficult to theoretically verify the soundness or effect of measures on strength of groups of rules. Instead, they explored the effect of measures empirically, and according to their experimental results, WCS was the best of a good set of candidate measure formulas.

4.4 Support Count Algorithm

This section describes an associative classification (AC) inspired technique, called Support Count (SC), that simplifies frequent pattern mining into frequent item counting. The rest of this section explains why some of the hard work performed on the training data set during the AC rule mining process is not necessarily useful to the final classification step.

During the course of this study, it became evident that the goals of AC in light of authorship attribution were to match the patterns in an anonymous e-mail with the frequent patterns extracted from a collection of e-mails E_i written by a suspect $S_i \in \{S_i, \ldots, S_n\}$. A typical step in evaluating classifiers of this sort is to separate a data set into training and

testing sets, and classifying a single e-mail was essentially the same concept as performing a *leave one out* test. Therefore, discovering all possible feature item combinations could be a waste of time if the test data did not include certain feature items. To maximize efficiency, we use the test data set to determine what patterns to look for. For example, if a feature item never appears in the test data set, it does not make sense to consider that feature item in the CAR discovery process.

In most criminal investigations, an investigator would have a very small number of anonymous e-mails to classify, and not large collections that constantly change as in the case of most AC problems. Therefore it makes sense to concentrate solely on the anonymous e-mail.

Clearly, it would be efficient and harmless to discard feature items that do not appear in a given test object before beginning the frequent item set discovery process, as those feature items will not be considered. Furthermore, it is not necessary to consider all possible combinations of feature items, as simply counting the occurrences in the training set is sufficient. This is achieved by naturally attributing more points to training records that contained the same features as the test object. Therefore if a pattern is present in the anonymous e-mail and a given training record, their *match* would be strong.

Algorithm 2 demonstrates the process undertaken in SC and shows how very simple it is. The algorithm can be implemented with a very small amount of code. In Lines 1 through 10, SC looks at the anonymous test object and counts the occurrences of each feature item in the various authors' training sets. In this manner, each author is given a total support count. In Lines 12 through 14 the total score for each author is divided by the number of rows in each author's training set in order to normalize the scores. This avoids a situation where an author with a larger training set would be favored. Lines 15 through 28 return the author with the highest score and then verify the result to determine overall accuracy. This process is simple, yet effective.

Algorithm 2 *Support Count*

Input: Anonymous e-mail $\{A_1, \cdots, A_n\}$ represented by $\{Fi_1, \cdots, Fi_n\}$
Input: Sets of training e-mails $\{M_1, \cdots, M_n\}$ by $\{S_1, \cdots, S_n\}$.

```
1: //create array AC to hold one integer counter for each author
2: for all A_i ∈ {A_1, ··· , A_n} do
3:     for all Fi_i ∈ {Fi_1, ··· , Fi_n} do
4:         for all M_i ∈ {M_1, ··· , M_n} do
5:             if Fi ∈ M then
6:                 AC[S|M] + +;
7:             end if
8:         end for
9:     end for
10: end for
11: //normalize count
12: for all S_i ∈ {S_1, ··· , S_n} do
13:     AC[S] = AC[S]/totalRecordsForAuthor(S);
14: end for
15: //find best count/author
16: bestCount = 0;
17: bestAuthor = 0;
18: for all S_i ∈ {S_1, ··· , S_n} do
19:     if AC[S] > bestCount then
20:         bestCount = AC[S];
21:         bestAuthor = S;
22:     end if
23: end for
24: if M|S == bestAuthor then
25:     correct + +
26: else
27:     incorrect + +;
28: end if
```

SC's strength lies in its ability to consider the complete training set. No information is discarded as noise or redundancy, since that noise will exist for the other authors as well. The greatly reduced execution complexity allows for this lack of pruning, something that would not be possible for a typical associative classification task due to the potentially incredibly high number of different combinations of frequent item sets. Furthermore, with the test data record setting the stage for information discovery, only useful information from the data set is explored.

These characteristics allow for SC to outperform the other algorithms in terms of performance and accuracy. The fact that SC is simpler is also an advantage as the concept

may be explained in brief to a judge and/or jury with a much higher chance of being understood, and thus deemed reliable. Whereas AuthorMiner [32] and CMAR [39] provide strong tangible evidence, they are far from simple and would require a much higher level of understanding and expertise to fully grasp. In light of that, a jury would most likely feel better about information they can understand themselves as opposed to only trusting an expert's opinion on a complicated data mining task's reliability.

Thus, in addition to the implementation of CMAR [39], CMARAA and CBA [41], we have implemented the SC algorithm as well, initially out of curiosity, but the results are very interesting. The SC method differs from AC in that it does not use support or confidence thresholds, so there are no variations in the results from the use of different values for these user-supplied inputs. This makes the algorithm require less oversight and function optimally regardless of context. Execution complexity is proportional to the number of stylometric features used, the number of e-mails in the training set and the number of e-mails in the test set. This is all true for associative classification implementations as well, but the difference is that the SC method does not need to discover frequent item sets, mine rules, or even construct a classifier.

Experimentally this method produces the best accuracies with the smallest memory requirement, as will be shown in Chapter 5.

Chapter 5

Experimental Evaluation

To evaluate the accuracy, efficiency and scalability of the Classification by Multiple Association Rule (CMAR) [39] algorithm, our proposed augmented implementation of it, CMAR for Authorship Attribution (CMARAA), and our novel Support Count (SC) method, we have performed an extensive performance study. In this chapter, we report our experimental results on comparing CMAR [39], CMARAA, and SC against two well known classification methods: Classification by Association (CBA) [41] for comparison against a baseline associative classification (AC) algorithm and AuthorMiner [32, 34] (AM), the previous leader in data mining based authorship attribution. It shows that CMARAA and SC outperform both CBA [41] and AM [32, 34] in terms of average accuracy.

5.1 Experiment Setting

All tests have been performed on a 3.4GHz Core i7 with 12G main memory, running Mac OS 10.7.3. CMAR [39] and CBA [41] were implemented by Frans Coenen, in the course of demonstrating the power and scalability of their Apriori-TFP method [14]. AuthorMiner [32, 34] was implemented by Iqbal et al. [32, 34].

The e-mail collections used in this study are all from the publicly available Enron e-mail

data set [1]. Specifically, hundreds of e-mails have been selected from authors Paul Allen, John Arnold, Sally Beck, John Dasovich, Mark Headicke, Vince Kaminsky, Steven Kean, Kam Keiser, Philip Love, Kay Mann, Susan Scott, Carol St Clair, Kate Symes and Kim Watson. E-mail headers were removed using a bash script and all content not written by the respective authors for the specific message has been removed manually. The purpose of this cleaning was to ensure that each e-mail consisted of text written solely by their author with no content included from attachments, forwards, replies, etc. In general only about 20% of e-mails in the original data set were retained as the rest consisted of text shorter than five sentences, forwarded materials or attachments, and incoming e-mails. Similarly for test objects, it would not prudent to expect the accurate classification of any message consisting of a few words or a single sentence. Messages that short would be trivial to alter in such a way that they contain no specific writing style whatsoever, or even a spoofed writing style if the malicious entity was well versed in authorship attribution methodologies.

All results are accuracy averages compiled from running the various algorithms on data sets with the same number of authors but each containing a different combination of authors, in order to show that results are stable and not simply the result of lucky guesses or hand picked sets that support our conclusions. As to be expected, sets of authors with low accuracies show lower accuracies across the board and vice-versa. Each test is fair and the data sets have not been altered in any way to give advantage to any one algorithm over the others.

For the experiments, the parameters are set as follows.

For CBA and CMAR, we set the support threshold to 10% and the confidence threshold to 0%. The reason the confidence threshold has been set to 0% is to demonstrate the effect the final round of pruning in CMARAA has on accuracy. A high confidence threshold would effectively eliminate common rules among multiple authors as explained previously. Furthermore, setting a different confidence threshold for the various algorithms

would make fair comparison impossible. Given that the confidence threshold in CMAR, CBA and CMARAA does not affect accuracy, it was deemed safe and fair to simply set it to 0%. Please note that the confidence and support thresholds are set mainly to improve efficiency, not accuracy. CMAR's database coverage threshold is set to 3 as per its original implementation. The chi square test in the second round of pruning is performed with a 5% confidence threshold.

CMARAA uses the same support and confidence thresholds as CBA and CMAR but the support threshold is considered on a per author basis instead of applying it to the size of the entire training data set. This allows for better extraction of frequent patterns on a per author basis. This is important because without this distinction, CMARAA would technically be looking for frequent patterns across all authors instead of patterns that are representative for a single author. Without this contribution, it would be harder to associate anonymous e-mails with their author, as the results demonstrate quite clearly.

The Support Count method does not take confidence or support into account.

All figures and results consider that the authorship identification accuracy is measured by the percentage of correctly matched authors in the testing set. For example, if there are 4 test records and 3 of them are correctly matched to an author, the accuracy will be 75%. The tests have been performed by splitting the entire data set into training and testing partitions with a ratio of 90% training data to 10% testing data. This means that if there are 1000 records in the entire data set, 900 records will be used for training and 100 for testing. This ratio best represents a real life scenario where there would be a small number of records in need of classification and a large number of training records. Whereas generally there might only be only one email requiring classification during an investigation, it is necessary to classify multiple records for the sake of determining general accuracy. If only one test record were to be used in our tests, the accuracy would obviously be 100% or 0%.

For AuthorMiner and the AC algorithms, the training sets are used to discover frequent

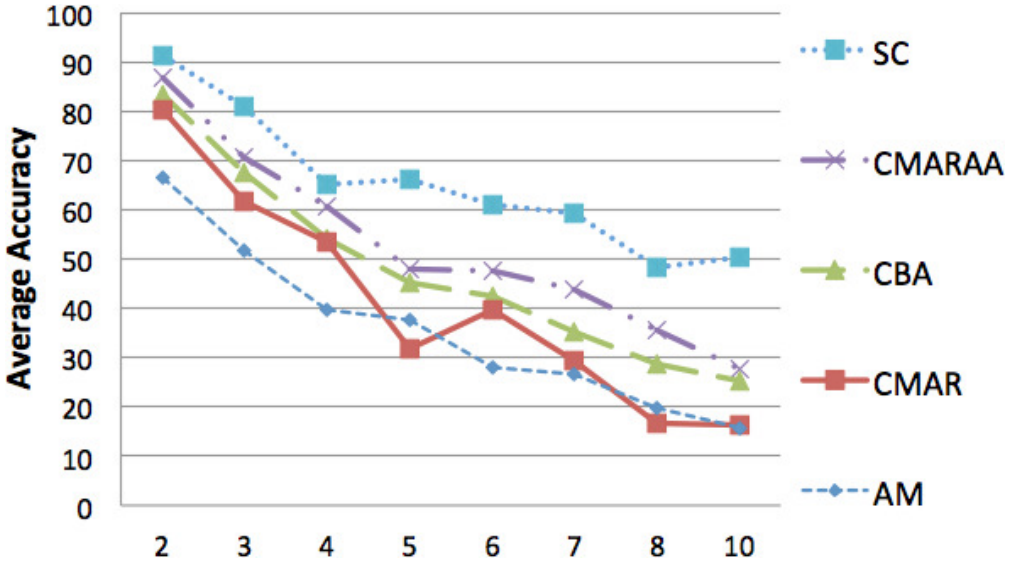

Figure 2: Accuracy vs. Number of Authors

patterns for the classifier and then each e-mail in the test set is classified and verified. The training and test set splits are done on a per author basis so each author's data set is split by the user-supplied percentage and the respective author sets are combined into a global training set and testing set. This separation is done using the same method for all tested algorithms in order for results to be directly comparable.

For tests to be repeatable, the training and testing set split is done in order and not at random, with the first portion belonging to the training set and the rest to the test set. E-mails are named numerically and are input in ascending alphabetical order. This does potentially cause the results to be skewed towards how easy or hard it is to classify e-mails found at the end of each set, but each algorithm must deal with this issue, again ensuring that results are comparable and fair to the strengths of each algorithm. The rest of this section describes the figures showing the results of the various tests.

5.2 Accuracy

Figure 2 shows average classification accuracies over sets of collections of e-mails for 2 to 5 authors for each algorithm tested in this study. Accuracies range from 30% to 92% with the most accurate algorithms being Support Count (SC), CMAR for Authorship Attribution (CMARAA), Classification by Association (CBA), Classification by Multiple Association Rule (CMAR), and AuthorMiner (AM), respectively. The various methods' accuracies decrease similarly when the number of authors increases, with CMAR seeing the biggest drop when there are 5 authors or more. We did not include the tests for more than 10 authors in this figure as most investigations consider a low number of suspects according to a source on a cyberforensics team; additionally the graph itself would be harder to read with any more information.

With the minimum support threshold held constant, CMAR no longer generates rules with every run when there are many authors; this is due to the fact that the writing styles of authors are usually distinct from one another and so feature items count will not always pass the minimum support threshold, which is a percentage of records across the entire training set. For example, if the support threshold is 10%, and there are 10 authors with 100 e-mails each, then a feature item unique to one author would need to appear in every single e-mail in order to be considered frequent. This characteristic hurts CMAR as the number of authors increases. One of the main contributions of this study, implemented in CMARAA, reduces the support threshold to consider only one author's training set, allowing feature items that are frequent, even if unique to one author, to be considered. The results support the implementation of this contribution.

Figure 3 shows average accuracies for each algorithm tested on ten distinct sets containing 2 authors each. Accuracies range from 65% to 92% with the most accurate algorithms being SC, CMARAA, CBA, CMAR and AM respectively. Each of the ten 2-authors tests,

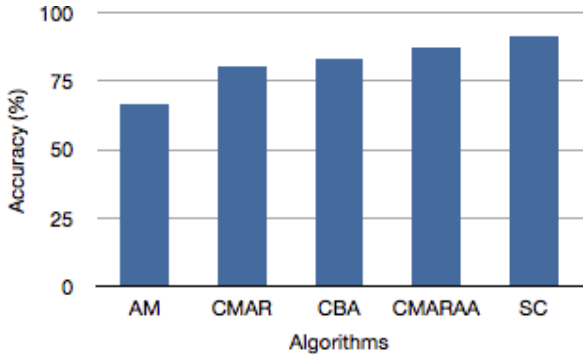

Figure 3: Accuracy of Different Algorithms (2 authors)

Dataset	Algorithm				
	AM	CMAR	CBA	CMARAA	SC
2a	65	100	100	100	100
2b	61	76.47	61.76	91.18	91.18
2c	72	80	84	84	92
2d	78.8	74.12	92.94	89.41	100
2e	61.72	74.07	76.54	74.07	91.35
2f	81.25	97.92	93.75	93.75	91.67
2g	57.65	80.18	92.79	83.78	93.69
2h	64	72.00	80.00	84.00	80.00
2i	73.47	77.55	75.51	87.76	97.96
2j	50	68.75	75	81.28	75
Average	66.49	80.106	83.23	86.923	91.29

Figure 4: Accuracy of Different Algorithms (2 authors)

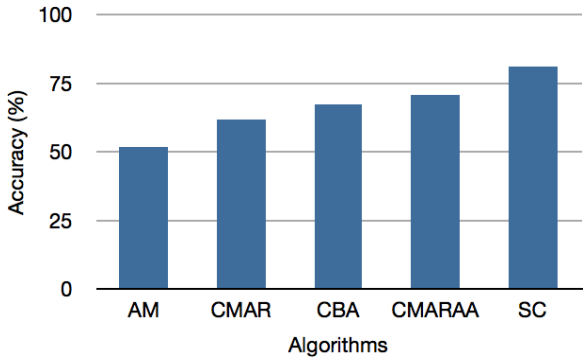

Figure 5: Accuracy of Different Algorithms (3 authors)

as shown in Figure 4 demonstrate accuracy similarities in their results. Set 2j, for example, scores lowest across the board, whereas higher scoring sets generally scored higher for each algorithm. This characteristic says more about how unique the writing styles of the included authors are than the strength of the algorithms; naturally, two authors who have similar writing styles will be harder to tell apart, regardless of which technique is employed.

Figure 5 shows average accuracies for each algorithm tested on six distinct sets containing 3 authors each. Due to the relatively limited number of complete author collections, some sets do contain common authors, but all sets compare a different combination of authors. The same is true for future graphs showing 4, 5, 6, 7, 8 and 10 authors.

Average accuracies range from 51% to 82% with the most accurate algorithms being SC, CMARAA, CBA, CMAR and AM respectively. Each of the six tests, as shown in Figure 6 once again show similar results from one set to another. The range of accuracies grows from 27% to 30% between the highest and lowest average, showing that differentiating between more authors makes for more variance in results. The algorithms largely maintain their respective strength in accuracies, with few exceptions of data sets being better represented by one algorithm over another. The SC method is 10% better on average than the runner up, CMARAA, compared to its previous lead of only 5% on average for 2 authors.

Dataset	Algorithm				
	AM	CMAR	CBA	CMARAA	SC
3a	64.55	64.56	68.35	77.22	82.27
3b	40.00	60.00	63.33	73.33	73.33
3c	48.93	61.70	61.70	63.83	76.59
3d	40.00	54.55	56.36	63.64	90.90
3e	45.11	58.64	75.19	71.43	82.70
3f	72.41	70.34	79.31	73.79	80.68
Average	51.83	61.63	67.37	70.54	81.08

Figure 6: Accuracy of Different Algorithms (3 authors)

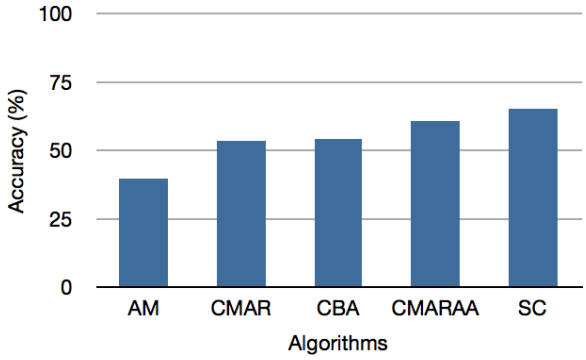

Figure 7: Accuracy of Different Algorithms (4 authors)

Figure 7 shows average accuracies for each algorithm tested on five distinct sets containing 4 authors each. Again, some sets do contain common authors, but all sets compare a different set of authors. Accuracies range from 39% to 65% with the most accurate algorithms being SC, CMARAA, CBA, CMAR and AM respectively. Each of the five tests, as shown in Figure 8 show a little bit less variability in scores for algorithms for specific sets. The overall range decreases to 25% between AM and SC. The tendencies remain constant though, with the same order of best accuracies. This time, however, some algorithms show better results than other algorithms that had done better for every set before. CBA and CMAR, for example, each win twice against each other, tying for set 4e.

Figure 9 shows average accuracies for each algorithm tested on five distinct sets containing 5 authors each. Some sets do contain common authors, but all sets compare a

Dataset	Algorithm				
	AM	CMAR	CBA	CMARAA	SC
4a	45.00	25.00	32.50	47.50	57.50
4b	39.47	50.00	48.68	57.89	65.78
4c	32.60	65.22	67.39	69.57	68.8
4d	34.5	64.6	60.18	70.8	66.37
4e	46.05	61.84	61.84	57.89	67.10
Average	39.52	53.332	54.12	60.73	65.11

Figure 8: Accuracy of Different Algorithms (4 authors)

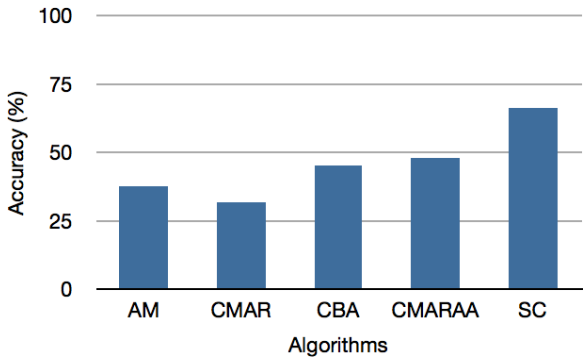

Figure 9: Accuracy of Different Algorithms (5 authors)

different set of authors. Accuracies range from 28% to 61% with the most accurate algorithms being SC, CMARAA, CBA, CMAR and AM respectively. Each of the five 6-author tests, as shown in Figure 10 show a little bit more variability in scores. The overall range increases to 36% between CMAR and SC. This time, however, some algorithms show better results than other algorithms that had done better for every set before. For example, AM's strategies give it the win over CMAR for 5 authors.

Dataset	Algorithm				
	AM	CMAR	CBA	CMARAA	SC
5a	30	8	28	38	50
5b	43.7	42.63	56.78	54.27	66.8
5c	35.8	50.81	55.65	57.25	72.5
5d	37.34	28.31	52.97	42.92	72.1
5e	41.17	28.43	33.3	47.06	69.61
Average	37.602	31.636	45.34	47.9	66.202

Figure 10: Accuracy of Different Algorithms (5 authors)

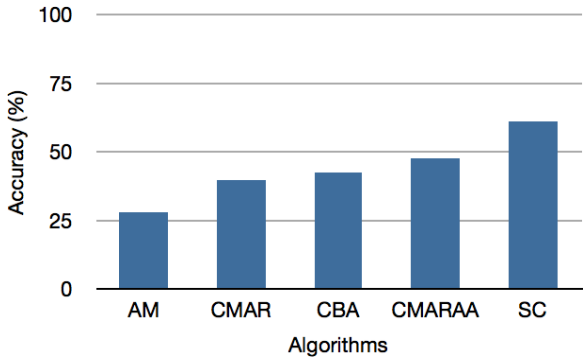

Figure 11: Accuracy of Different Algorithms (6 authors)

Dataset	Algorithm				
	AM	CMAR	CBA	CMARAA	SC
6a	31.8	42.45	42.46	64.8	64.8
6b	26.4	40.2	50.98	38.72	56.86
6c		41.3	41.3	44.02	63.04
6d	25.8	34.19	35.48	43.23	59.35
Average	28	39.535	42.56	47.6925	61.013

Figure 12: Accuracy of Different Algorithms (6 authors)

Figure 11 shows average accuracies for each algorithm tested on four distinct sets containing 6 authors each. Some sets do contain common authors, but all sets compare a different set of authors. Accuracies range from 26% to 60% with the most accurate algorithms being SC, CMARAA, CBA, CMAR and AM respectively. Each of the four 7-author tests, as shown in Figure 12 show a similar variability in scores. The overall range decreases to 33% between AM and SC. This time, however, CMAR's accuracy bests AM as it did for the first few experiments on 2, 3 and 4 author sets.

Figure 13 shows average accuracies for each algorithm tested on four distinct sets containing 7 authors each. Some sets do contain common authors, but all sets compare a different set of authors. Accuracies range from 26% to 60% with the most accurate algorithms being SC, CMARAA, CBA, CMAR and AM respectively. Each of the four 7-author tests, as shown in Figure 14 show a similar variability in scores. The overall range increases to 34% between AM and SC from the 6-author tests.

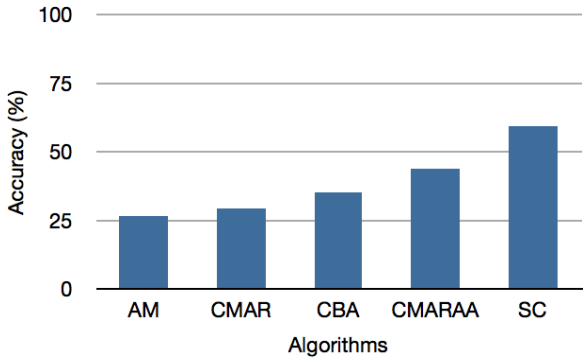

Figure 13: Accuracy of Different Algorithms (7 authors)

Dataset	Algorithm				
	AM	CMAR	CBA	CMARAA	SC
7a	27.7	39.39	38.89	58	62.12
7b	23.07	32.05	41.03	38.4	55.98
7c	28.79	19.9	19	37.17	53.92
7d	26.55	26.14	41.91	41.9	64.73
Average	26.528	29.37	35.21	43.8675	59.188

Figure 14: Accuracy of Different Algorithms (7 authors)

Figure 15 shows average accuracies for each algorithm tested on four distinct sets containing 8 authors each. All sets contain common authors but each compares a different set of authors. Accuracies range from 19% to 50% with the most accurate algorithms being SC, CMARAA, CBA, AM and CMAR respectively. Each of the 4 tests, as shown in Figure 16 show less variability in scores for algorithms for specific sets. The overall range decreases to 30% between CMAR and SC. For test set 8a, CMAR was unable create any classification rules, most likely due to the support threshold issue discussed previously. Even though we don't include this as 0% accuracy, the results for CMAR with 8 authors is significantly lower, coming in last place this time. CMARAA's second place score again demonstrates why changes were made to CMAR in customizing AC for authorship attribution. Even as CBA and AuthorMiner's scores continue their decline, CMARAA's decline is much less drastic. SC dips below 50% for the first time, but this may be due to the lack of enough data to fully separate the four 8-author data sets.

Figure 17 shows average accuracies for each algorithm tested on three sets containing

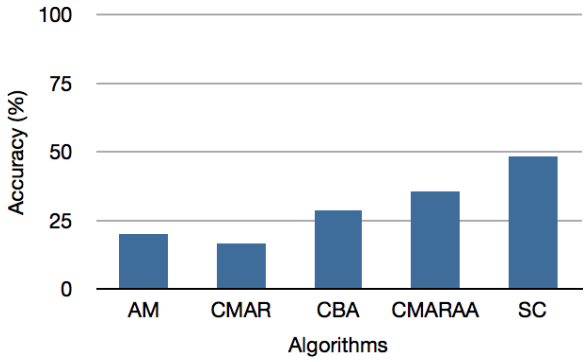

Figure 15: Accuracy of Different Algorithms (8 authors)

Dataset	Algorithm				
	AM	CMAR	CBA	CMARAA	SC
8a	18.98		13.92	20.34	48.10
8b	16.79	12.89	23.44	44.50	46.10
8c	19.50	19.48	40.68	43.20	42.80
8d	24.10	17.62	36.78	33.72	55.60
Average	19.84	16.66	28.71	35.44	48.15

Figure 16: Accuracy of Different Algorithms (8 authors)

a different set of 8 authors. There is significant overlap in these sets, which is why we test fewer sets. The results show the same tendencies as the majority of the other tests. Accuracies range from 15% to 50% with the most accurate algorithms being SC, CMARAA, CBA, AM and CMAR respectively. This is the largest variance between results as of yet, with 45% separating AM and SC. Execution time becomes an issue for AuthorMiner which uses a more traditional Apriori algorithm for Frequent item set discovery. CMAR and CBA are not affected as they use the CR-Tree [14] structure, which allows for much more efficient storage and execution time. Running AM on set 10c took more than a day of execution time, as well as a whooping 8GB of memory all the while; given that it did not finish within what we consider a reasonable amount of time, the result for 10c has been discarded and does not affect the average score.

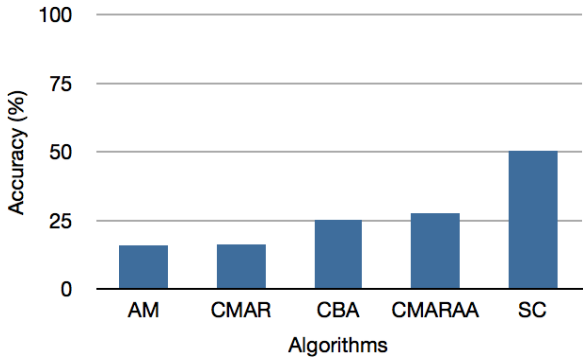

Figure 17: Accuracy of Different Algorithms (10 authors)

Dataset	Algorithm				
	AM	CMAR	CBA	CMARAA	SC
10a	13.87	14.84	30.97	36.81	45.16
10b	17.69	22.12	28.32	24.18	50.73
10c		12.00	16.36	22.10	55.63
Average	15.78	16.32	25.22	27.70	50.51

Figure 18: Accuracy of Different Algorithms (10 authors)

5.3 Efficiency

We do not provide extensive CPU or I/O metrics in this study. In the context of a criminal investigation, we assume that execution time is of little importance compared to the quality of classification and evidence collection. With that said, Figure 19 shows the average number of seconds each algorithm took to complete the classification process on sets of 2, 3, 4 and 5 authors. The algorithms that required the least CPU time for 2 authors were CBA, CMAR, SC, CMARAA and AM respectively. For 3 authors, CMAR came in first, with CBA, SC, CMARAA and AM in second, third, fourth and fifth place respectively. The same order of fastest times held true for 4 authors. For 5 authors, SC came in first place followed by CBA, then CMAR, then CMARAA and finally AuthorMiner. This trend continues as more authors are added, but given that this was not a focus of our study, we do not include them here.

The runtime presented in Figure 19, with the exception of SC, is primarily the result of

Number of Authors	Average Algorithm Run Time (seconds)				
	AM	CMAR	CBA	CMARAA	SC
2	15.717	3.1324	3.06	8.1525	3.13
3	52.83	7.27	7.83	55.39	8.42
4	575.74	8.44	8.51	96.34	9.09
5	443.4666	13.239	12.98	108.7104	11.3

Figure 19: Efficiency

their respective frequent item set discovery processes. AuthorMiner uses the original Apriori [4] algorithm for discovering frequent item sets, whereas CMAR, CBA and CMARAA all use a much faster variant of the FP-Growth [29] algorithm, using more efficient tree structures to represent frequent item sets and class association rules. Technically AM could use the FP-growth [29] method, but improving its performance was not a goal of this study. SC does not store any data or require any sort of frequent pattern discovery process, which becomes more important as the number of authors increases.

These results demonstrate reliable and repeatable proof that authorship attribution data mining methods can be very powerful. This also proves, once again, that the writing styles of authors can be modelled by extracting patterns from transformed semantic content to create individually recognizable writeprints.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Cybercrime investigations are in need of a state of the art computer aided writeprint modelling algorithm that can provide reliable evidence to support attribution. When someone's innocence or guilt is on the line, it is very important to have access to the most accurate and reliable methods of linking suspects to the evidence collected.

This thesis explores the application of a promising data mining technique called associative classification on the e-mail authorship attribution problem for the first time. Additionally, we propose that class-based associative classification allows for the extraction of a more accurate and complete writeprint. We also submit that modifying the rule pruning and ranking system described in the popular Classification by Multiple Association Rule (CMAR) algorithm to prioritize more specific patterns can also provide a more accurate writeprint. The removal of common patterns between authors from the extracted writeprint also makes for more convincing evidence in a court of law thanks to a disambiguation of each writing style. The novel Support Count algorithm does even better in terms of accuracy and efficiency by utilizing the entire training set and concentrating solely on support count. Thus, associative classification and the Support Count algorithm can help fight

crime by solving the e-mail authorship attribution problem and providing investigators with convincing evidence using our new computer-aided method.

6.2 Limitations

Although our proposed technique can achieve good accuracy in a reasonable span of time, it must be noted that there certainly are limitations. First of all, a writeprint can only be modelled given a sufficient quantity of training data. This means one of the suspects must be the criminal and the investigator must be able to legally obtain at least 50 e-mails from each suspect. Without meeting those requirements, the conclusion on authorship would be poorly supported and therefore should not hold much weight as evidence.

It is also possible that a malicious entity with knowledge of stylometry would be able to obfuscate their writing style using a variety of manual and automatic methods. It has been proposed that simply passing a text through a translation processor twice, to and from another language, would sufficiently alter the text to remove many aspects of characteristic writing style features. A malicious entity could even develop a model of some one else's writeprint and replicate those styles in their own writings, this would effectively change the writing style to point to an innocent person and leave no trace of the true author.

Furthermore, this method does not replace the need for an expert in authorship attribution, but instead attemps to automate the process, perhaps simply to guide an investigation or guide the expert. Manually determining similarities in writing styles is an expensive and time consuming process that could be streamlined using techniques like our own.

6.3 Future Work

The contributions of this thesis allow for better accuracies in authorship attribution problems by means of associative classification. It is highly possible that the modifications

made to our variant of associative classification would also benefit other domains that use similar techniques. Therefore future research could explore our extensions on other classification problems to see if the advantages of our changes are domain specific or perhaps more general in nature.

This paper focuses solely on the problem of authorship attribution of emails; our technique could also be applied to other types of written work. For example, the anonymous nature of the journal paper review process could be explored in order to attempt to determine the reviewer of a given paper by extracting a writeprint from a review and comparing it to other known texts written by the known list of reviewers. Another type of short written document of interest to cybercrime investigations are chat logs; it would be interesting to see if our technique could accurately identify a suspect based on his/her chat sessions. Given that people may use aliases when communicating online, it would be useful to associate different aliases to the same person and developing a writeprint would certainly help achieve that goal.

Our novel Support Count method may also be applied to other domains that rely on classification. The simplicity and novel nature of this algorithm would make for a very interesting study to see if the excellent accuracy and performance gains can be achieved on problems other than authorship attribution.

The Support Count method is not optimized to provide the same level of quality evidence that CMARAA and AM provide naturally. However, a simple extension of the code that would allow SC to count individual features and then display the top few features in terms of support could provide a similar measure of convincing evidence. Future work may require this contribution if law enforcement wishes to implement our techniques for use in real investigations.

The Internet has brought the world of crime online and as crime prevention is still prevalent, cybercrime prevention is much less developed. Laws, rules and regulations in

the cyber world are still being drafted to try to contain the explosion of cyber-criminal activities of the past 15 years. There is still much research to be done and much progress to be made in this regard. The key is to keep pushing forward and try to make our world safe, fair and orderly.

Bibliography

[1] Enron dataset http://www.cs.cmu.edu/˜enron/.

[2] S. Aaronson. Stylometric clustering, a comparative analysis of data-driven and syntactic features. Technical report, UC Berkeley, 1999. http://www.cs.berkeley.edu/aaronson/sc/report.doc.

[3] A. Abbasi and H. Chen. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems*, 26(2):1–29, 2008.

[4] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 207–216, 1993.

[5] S. Argamon and M. Saric. Style mining of electronic messages for multiple authorship discrimination: first results. In *Proc. of the 9th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 475–480, Washington, D.C., 2003. ACM.

[6] R. H. Baayen, H. van Halteren, and F. J. Tweedie. Outside the cave of shadows: using syntactic annotation to enhance authorship attribution. *Literary and Linguistic Computing*, 2:110–120, 1996.

[7] C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[8] J. F. Burrows. Word patterns and story shapes: the statistical analysis of narrative style. *Literary and Linguistic Computing*, 2:61–67, 1987.

[9] J. F. Burrows. An ocean where each kind...: Statistical analysis and some major determinants of literary style. *Computers and the Humanities*, 23(4-5):309–321, 1989.

[10] V. R. Carvalho and W. W. Cohen. Learning to extract signature and reply lines from email. In *Proc. of the conference on email and anti-spam*, Mountain View, CA, 2004.

[11] H. Chen, W. Chung, Y. Qin, M. Chau, J. J. Xu, G. Wang, R. Zheng, and H. Atabakhsh. Crime data mining: an overview and case studies. In *Proc. of the annual national conference on digital government research*, pages 1–5. Digital Government Society of North America, 2003.

[12] H. Chen, W. Chung, J. J. Xu, G. Wang, Y. Qin, and M. Chau. Crime data mining: A general framework and some examples. *Computer*, 37(4):50–56, 2004.

[13] N. Chou, R. Ledesma, Y. Teraguchi, and J. C. Mitchell. Client-side defense against web-based identity theft. In *Proc. of the Network and Distributed System Security Symposium, NDSS 2004*, San Diego, California, USA, 2004.

[14] F. Coenen, G. Goulbourne, and P. Leng. Tree structures for mining association rules. *Data Mining and Knowledge Discovery*, 8:25–51, 2004.

[15] M. Corney, O. de Vel, A. Anderson, and G. Mohay. Gender-preferential text mining of e-mail discourse. In *Proc. of the 18th Annual Computer Security Applications Conference (ACSAC)*, page 282, 2002.

[16] N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines*. Cambridge University Press, UK, 2000.

[17] D. Das and A. F. T. Martins. A survey on automatic text summarization. Web site: http://www.cs.cmu.edu/ nasmith/LS2/das-martins.07.pdf, 2007. Language Technologies Institute, Carnegie Mellon University.

[18] O. de Vel. Mining e-mail authorship. *KDD*, August 200.

[19] O. de Vel, A. Anderson, M. Corney, and G. Mohay. Mining e-mail content for author identification forensics. *SIGMOD Record*, 30(4):55–64, 2001.

[20] O. de Vel, A. Anderson, M. Corney, and G. Mohay. Multi-topic e-mail authorship attribution forensics. In *Proc. of ACM Conference on Computer Security - Workshop on Data Mining for Security Applications*, 2001.

[21] J. Diederich, J. Kindermann, E. Leopold, and G. Paass. Authorship attribution with support vector machines. *Applied Intelligence*, 19:109–123, 2000.

[22] W. Elliot and R. Valenza. Was the earl of oxford the true shakespeare? *Notes and Queries*, 38:501–506, 1991.

[23] J. M. Farringdon. Analyzing for authorship: A guide to the cusum technique. 1996.

[24] R. S. Forsyth and D. I. Holmes. Feature finding for text classification. *Literary and Linguistic Computing*, 11(4):163–174, 1996.

[25] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29:131–163, 1977.

[26] M. Gamon. Linguistic correlates of style: authorship classification with deep linguistic analysis features. In *Proc. of the 20th International Conference on Computational Linguistics*, pages 611–617, Geneva, Switzerland, 2004.

[27] J. Grieve. Quantitative authorship attribution: An evaluation of techniques. *Literary and Linguistic Computing*, 22(3), July 2007.

[28] R. Hadjidj, M. Debbabi, H. Lounis, F. Iqbal, A. Szporer, and D. Benredjem. Towards an integrated email forensics analysis framework. *Digital Investigation*, 5(3-4):124–137, 2009.

[29] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *SIGMOD'00*, May 2000.

[30] Jiawei Han and Xiaoxin Yin. Cpar: Classification based on predictive association rules. In *Proc. of the third Society for Industrial and Applied Mathematics*. Society for Industrial and Applied Mathematics, 2003.

[31] D. I. Holmes. The evolution of stylometry in humanities. *Literary and Linguistic Computing*, 13(3):111–117, 1998.

[32] F. Iqbal, H. Binsalleeh, B. C. M. Fung, and M. Debbabi. Mining writeprints from anonymous e-mails for forensic investigation. *Digital Investigation*, pages 1–9, 2010.

[33] F. Iqbal, H. Binsalleeh, B. C. M. Fung, and M. Debbabi. A unified data mining solution for authorship analysis in anonymous textual communications. *Information Sciences: Special Issue on Data Mining for Information Security*, in press.

[34] F. Iqbal, R. Hadjidj, B. C. M. Fung, and M. Debbabi. A novel approach of mining write-prints for authorship attribution in e-mail forensics. *Digital Investigation*, 5(1):42–51, 2008.

[35] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proc. of the 10th European Conference on Machine Learning ECML*, pages 137–142. Springer Verlag, 1998.

[36] J. Klensin. Simple mail transfer protocol. Technical Report 1, Internet Engineering Task Force, April 2001. RFC 2821.

[37] G. R. Ledger and T. V. N. Merriam. Shakespeare, Fletcher, and the two Noble Kinsmen. *Literary and Linguistic Computing*, 9:235–248, 1994.

[38] W. Li. Classification based on multiple association rules, April 2001.

[39] W. Li, J. Han, and J. Pei. Cmar: Accurate and efficient classification based on multiple class-association rules. In *Proc. of the International Conference of Data Mining*, 2001.

[40] R. P. Lippmann. An introduction to computing with neural networks. *IEEE Acoustics Speech and Signal Processing Magazine*, 4(2):4–22, 1987.

[41] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. *In proc. of Knowledge Discovery and Data Mining*, August 1998.

[42] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. pages 281–297, 1967.

[43] L. M. Manevitz, M. Yousef, N. Cristianini, J. Shawe-taylor, and B. Williamson. One-class svms for document classification. *Journal of Machine Learning Research*, 2:139–154, 2001.

[44] T. C. Mendenhall. The characteristic curves of composition. *Science*, 11(11):237–249, 1887.

[45] F. Mosteller and D.L. Wallace. *Applied Bayesian and classical inference: The case of the Federalist Papers*. Springer-Verlag, New York, 2nd edition, 1964.

[46] J. Pearl. Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proc. of the 7th Conference of the Cognitive Science Society*, pages 329–334, 1985.

[47] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[48] G. Salton. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[49] M. Sewell. Feature selection, 2007. http://machine-learning.martinsewell.com/feature-selection/feature-selection.pdf.

[50] E. Stamatatos, N. Fakotakis, and G. Kokkinakis. Computer-based authorship attribution without lexical measures. *Computers and the Humanities*, 35(2):193–214, 2001.

[51] G. Teng, M. Lai, J. Ma, and Y. Li. E-mail authorship mining based on svm for computer forensic. In *Proc. of the 3rd International Conference on Machine Learning and Cyhemetics*, August 2004.

[52] F. Thabtah, P. Cowling, and Y. Peng. Mcar: multi-class classification based on association rule. In *ACS/IEEE 2005 International Conference on Computer Systems and Applications*, page 33. aiccsa, 2005.

[53] F. J. Tweedie and R. H. Baayen. How variable may a constant be? measures of lexical richness in perspective. *Computers and the Humanities*, 32:323–352, 1998.

[54] M. White, T. Korelsky, C. Cardie, V. Ng, D. Pierce, and K. Wagstaff. Multidocument summarization via information extraction. In *Proc. of the first international conference on Human language technology research*, pages 1–7, Morristown, NJ, USA, 2001. Association for Computational Linguistics.

[55] G. U. Yule. On sentence length as a statistical characteristic of style in prose. *Biometrika*, 30:363ñ–390, 1938.

[56] G. U. Yule. *The statistical study of literary vocabulary.* Cambridge University Press, Cambridge, UK, 1944.

[57] Y. Zhao and J. Zobel. Effective and scalable authorship attribution using function words. In *Proc. of the Second AIRS Asian Information Retrieval Symposium*, pages 174–189. Springer, 2005.

[58] R. Zheng, Y. Qin, Z. Huang, and H. Chen. Authorship analysis in cybercrime investigation. In *Proc. of the NSF/NIJ Symposium on Intelligence and Security Informatics (ISI)*, pages 59–73, 2003.