Foundations and Applications of Authorship Attribution Analysis

Rohan Shukla

Computer Science Senior Thesis

Professor Deepak Kumar

December 15, 2018
Abstract

Authorship attribution is the process of identifying the author of a given work. This thesis surveys the history and foundations of authorship attribution, and then analyzes multiple machine learning methods that are used frequently in this field. In the classic authorship attribution problem, a text with unknown authorship is assigned an author from a set of candidate authors for whom documents of irrefutable authorship exist. Prior to the 1960’s, authorship attribution was a linguistics-focused field in which linguistic experts would determine the authors of unknown texts. In 1964, the analysis of ‘The Federalist Papers’ by Mosteller and Wallace was the first statistically driven approach to authorship attribution. This study marked the beginning of authorship attribution as a computational field rather than a linguistics field.

The modern approach to authorship attribution involves selecting a set of linguistic features from the texts at hand and then applying a machine learning method on that feature set to classify authorship. This thesis analyzes multiple machine learning methods used for this purpose. Principal Components Analysis (PCA) is a popular unsupervised learning method that considers each text’s feature set as a vector in a multivariate vector space and has had success in authorship attribution. Support Vector Machines (SVMs) are a powerful supervised learning technique that creates a linear classifier used to attribute authorship. SVMs have outperformed all other analytical techniques used in authorship attribution. Due to the plethora of electronic texts that exist, authorship attribution has extensive applications in many different fields, with current research focusing primarily on developing application-specific methodologies.
# TABLE OF CONTENTS

Abstract .............................................................................................................................. ii
Table of Contents ..................................................................................................................... iii

1. Introduction ................................................................................................................................. 1
   1.1. The Authorship Attribution Problem ............................................................................... 1
   1.2. Applications of Authorship Attribution ......................................................................... 2

2. History .................................................................................................................................... 3
   2.1. Pre-History to 1964 ....................................................................................................... 3
   2.2. 1964 - The Federalist Papers ......................................................................................... 3
   2.3. 1990’s - The Cusum Failure ......................................................................................... 4
   2.4. 1990’s - The Infamy of “A Funeral Elegy” .................................................................... 5
   2.5. 1995 - The Success of *Primary Colors* ........................................................................ 5
   2.6. Historical Summary ........................................................................................................ 6

3. Linguistic Background ................................................................................................................. 6
   3.1. Overview of Linguistic Features ...................................................................................... 6
   3.2. Lexical Features ............................................................................................................... 7
      3.2.1. Word Frequency Analysis ..................................................................................... 7
   3.3. Character Features .......................................................................................................... 8
   3.4. Syntactic and Semantic Features .................................................................................. 9
   3.5. Application-Specific Features ...................................................................................... 9
   3.6. Summary of Linguistic Background .............................................................................. 10

4. Survey of Unsupervised Learning Methods ............................................................................... 10
   4.1. Profile-Based Approaches ............................................................................................... 11
   4.2. Principal Components Analysis .................................................................................... 13
      4.2.1. PCA in Authorship Attribution .............................................................................. 14
   4.3. Other Unsupervised Learning Methods ......................................................................... 15
   4.4. Summary of Unsupervised Learning Methods ............................................................. 17

5. Foundations of Support Vector Machines ............................................................................... 18
   5.1. Instance-Based Approaches ............................................................................................. 18
   5.2. Background in Linear Classification ............................................................................. 19
   5.3. Hard-Margin Support Vector Machines ........................................................................ 22
   5.4. Soft-Margin Support Vector Machines .......................................................................... 24
   5.5. Basis Expansion and Kernels ......................................................................................... 26
   5.6. Support Vector Machines in Authorship Attribution ..................................................... 30

6. Conclusion ................................................................................................................................. 31

References ..................................................................................................................................... 34
1. Introduction

Authorship attribution is the process of identifying the author of a given work. This thesis surveys the history, foundations, and evolution of authorship attribution, and then analyzes multiple machine learning methods that are used frequently in this field. It begins with a broad overview of the field, mentioning its extensive applications as motivation for the importance of studying this problem. It then examines the history of the field, discussing many of the famous authorship attribution studies that brought the field significant attention from literary scholars and computer scientists. Once the context for the field has been sufficiently established, it will provide the linguistic background that is necessary to understand modern authorship attribution studies. Then, it will survey a variety of machine learning techniques that are used in the field. This machine learning discussion begins with an overview of unsupervised learning, building up to an analysis of the Principal Components Analysis methodology. Then, an overview of supervised learning is given before presenting a deep analysis of Support Vector Machines, which is arguably the best modern machine learning technique. Authorship attribution studies using these learning methods are examined in conjunction with their technical analysis. Finally, the thesis concludes with a summary of key points, mentioning the key authorship attribution questions and debates that researchers are still trying to resolve.

1.1. The Authorship Attribution Problem

Authorship attribution is a field that has existed for hundreds of years, as questions regarding the ownership of words have existed as long as words have been recorded. However, recent advances in statistics and machine learning modeling have created new methodologies for attributing authorship that were previously impossible, as the field’s focus has shifted from
linguistics to computation. In the classic authorship attribution problem, a text with an unknown author is assigned an author from a set of candidate authors for whom documents of irrefutable authorship exist. Characteristics of these candidate authors are determined by analyzing the documents that they have written. These traits are subsequently compared to the linguistic characteristics of the unknown text to determine its author. There are few, if any, generally accepted best practices in the field, making authorship attribution an area of study in computer science with great potential for groundbreaking research and discoveries.

1.2. Applications of Authorship Attribution

Despite lacking standard practices, authorship attribution analysis still has a wide array of important applications to many fields, thereby meriting the motivation for its study. Furthermore, the plethora of electronic texts that exist now only create even more avenues for authorship attribution to be utilized. In the field of intelligence, for instance, authorship attribution is used to identify messages by known terrorists and create links between different messages (Stamatatos, 2009). In criminology, it can identify the creators of hazing messages and the veracity of suicide notes (Stamatatos, 2009). In legal contexts, it plays a major role in settling copyright disputes (Stamatatos, 2009). Authorship attribution is not just limited to natural languages either; in computer forensics, for instance, it is used to identify the creators of source code of malware (Stamatatos, 2009). In academic settings, there is plagiarism detection software built on the foundations of authorship attribution (Stamatatos, 2009). Authorship attribution even comes up in the media. In 2018, an anonymous Op-Ed was posted in the New York Times that criticized President Trump, and various authorship attribution techniques have been used to identify which senior member of his administration wrote it (Lieberman, 2018). Ultimately there is no shortage
of useful applications of authorship attribution, which explains why much of the recent work in the field has been dedicated to developing application-specific methods for analyzing texts.

2. History

2.1. Pre-History to 1964

Today, authorship attribution is a computationally-driven process that utilizes some of the most powerful statistical and machine learning techniques that exist. However, the problem of attributing authorship has existed for much longer than the field of computer science; it has existed for as long as words have been documented. The same can be said for authorship analysis, the process of inferring characteristics of an individual based on how he or she writes and speaks. This linguistic process is documented even as far back as in the Old Testament (Judges 12:5) when the Gileadites identified an Ephraimite by his inability to pronounce a certain sound, enabling them to distinguish him from a different population that could properly pronounce that same sound (Juola, 2007). This is no different than associating individuals with a certain county based on their accent or how they spell certain words (such as “color” in America versus “colour” in England). Prior to the 1960’s, all work on authorship attribution was done by “human expert-based methods” (Stamatatos, 2009). These methods consisted of linguistic experts identifying particular key features to an individual’s writing or speaking as grounds for attributing authorship. While these methods will not be examined in-depth, their stylometric roots influenced the next wave of authorship attribution techniques.

2.2. 1964 - The Federalist Papers

The first statistically-driven study of authorship attribution was the analysis of ‘The Federalist Papers’ by Mosteller and Wallace in 1964. These papers were a collection of 85 political essays written between 1787 and 1788 by the anonymous author “Publius,” who pushed
for the ratification of the newly proposed United States Constitution (Mosteller & Wallace, 1964). It has since become known that the authors behind Publius were John Jay, Alexander Hamilton, and James Madison. There was general consensus about which of the authors wrote 73 of the papers, but much dispute surrounded the remaining 12. Mosteller and Wallace set out to analyze these documents by focusing on “function words,” such as conjunctions and prepositions like “and” and “of.” These words have limited intrinsic meaning, but are excellent indicators of an author’s style because they are used subconsciously, represent preferred methods of expressing abstract concepts and relationships (such as ownership), and are topic-independent (Juola, 2007). Mosteller and Wallace applied Bayesian statistical analysis (a dynamic probabilistic model) to a set of 30 common function words from the papers, which identified significant differences between the candidate authors. The Federalist analyses are considered a pioneering work in modern authorship attribution, and new attribution methods are now almost always tested on these papers to determine their accuracy.

### 2.3. 1990’s - The Cusum Failure

Mosteller and Wallace are one of the earliest successes of authorship attribution and brought significant attention to the field as a whole. However, it was not just successes that made the field popular; certain failures and controversies brought authorship attribution under the spotlight as well. In the 1990’s, a sequential analysis technique called cusum (also Qsum) became part of forensic methods for attributing authorship in multiple English court cases. Cusum, short for cumulative sum, measures the stability of a given feature by plotting the difference of the elements of a sequence from the mean of a sequence (Juola, 2007). Unfortunately, though, when the accuracy of cusum was questioned in criminal cases, it was
found to be highly unreliable. In fact, on national British television, it was unable to distinguish writings of a known felon from the Chief Justice of England (Juola, 2007). Nevertheless, this failure brought further attention to the field, even if in a negative light, and inspired the search for better methods.

2.4. 1990’s - The Infamy of “A Funeral Elegy”

Another notable failure in the history of authorship attribution came in the late 1990’s by Don Foster, a professor of English at Vassar College. Foster deployed a slew of computer-based stylometric techniques onto the poem “A Funeral Elegy” by the anonymous “W.S.” and ultimately confirmed that the author was none other than William Shakespeare (Juola, 2007). His discovery was so controversial that it ended up making it to the front page of the New York Times, as Shakespearean scholars completely disagreed with Foster’s analysis. The controversy grew so large that additional scholars applied different authorship attribution methods to the poem, and concluded that the author was in fact John Ford, not Shakespeare (Juola, 2007).

2.5. 1995 - The Success of Primary Colors

Some of Foster’s other findings, though, were deemed legitimate. In 1995, Foster attempted to identify the author of Primary Colors, an anonymously published book detailing Bill Clinton’s 1992 presidential campaign. Foster identified the author as Joe Klein, a political columnist for Time magazine, which became generally accepted as correct. Foster’s work in the field was so significant at the time that he earned the title as the best “literary detective” in the world.
2.6. Historical Summary

There is no shortage of notable work that has been done on authorship attribution. While some of the studies, like the *Federalist* analyses, worked extremely well, the ones which failed unfortunately hindered the public perception of the field altogether. In fact, the well-documented publicity of these failures prevented some scholars from entering the field. Nevertheless, these works sparked many important questions and principles that are considered in modern authorship attribution, and they provided a glimpse into the potential for this type of analysis. Through these foundational works, much more clarity has been gained on different ways to improve technical accuracy along with insight on what an optimal authorship attribution system would look like.

3. Linguistic Background

3.1. Overview of Linguistic Features

Authorship attribution analysis involves selecting a set of linguistic features from the texts at hand, and then selecting an analysis method to be applied on that feature set. Linguistic features are qualities of a written document that express the writer’s style. In other words, they are different ways of quantifying and classifying writing style. Finding the optimal linguistic features for a set of texts is a difficult task, as there are many different ways to simplify and model facets of natural languages. This section will examine a variety of different ways to accomplish this feat. Before delving into the different machine learning models that ultimately attribute authorship, it is fundamental to have a strong grasp on the different types of stylometric features that these advanced computational models use as the basis for their classification.
3.2. Lexical Features

Historically, scholars of authorship attribution primarily categorized texts into their lexical features, which consider a text merely as a sequence of word-tokens where each token is a word, number, or punctuation mark (Stamatatos, 2009). Despite the simplicity of this method, it has the advantage of being applicable to any language, as long as a tokenizer for that language exists (Stamatatos, 2009). Lexical analysis looks at features like vocabulary richness, word frequencies, word n-grams, and spelling errors. Vocabulary richness, representing the diversity of words used in a given text, is generally unreliable to use because it is heavily dependent on the length of that text. In general, as text length increases, vocabulary richness does as well.

3.2.1 Word Frequency Analysis

Word frequency analysis is perhaps the best of all lexical methods. In this approach, texts are viewed as “vectors of word frequencies” (Stamatatos, 2009). As mentioned earlier, it has been found that applying this approach to function words is one of the best ways to differentiate authors (Argamon & Levitan, 2005). In certain scenarios, it can be useful to consider topic words, which are words that specifically relate to the topic of the texts at hand. Generally, because there are only so many function words, the dimensionality of the function word vector space is far smaller than that of a topic-based one, making it not only a more general analysis that can be applied across different topics, but a computationally simpler one too.

Frequency analysis on a text is often compared to the Zipf distribution of the Brown corpus, which is a generally accepted approximation of the distribution of words in the English language (Juola, 2007). One way to incorporate some contextual information into lexical analysis is by the use of n-grams, which are sets of n continuous words (Juola, 2007). For instance, in the
sentence “the dog jumped over the fence,” examples of 2-grams (bigrams) are “the dog” and “the fence,” while “over the fence” is a 3-gram (trigram). The dimensionality of n-gram vector spaces increases dramatically with n, and sometimes n-gram analysis captures too much content-specific information instead of stylistic information, but it is still an effective tool to be combined with other methods. Finally, spelling and formatting errors serve as a simple lexical way of distinguishing authors based on their tendencies to repeat certain mistakes.

3.3. Character Features

A simpler, yet more modern approach to lexical analysis is that of character features, which considers a text as a sequence of characters (Stamatatos, 2009). This methodology enables the counting of alphabetic characters, digits, punctuation characters, their frequencies, and so on. This approach can capture more specific nuances of writing style than lexical analysis, as it analyzes at a deeper level. This increased specificity is also useful because of its tolerance to noise (Stamatatos, 2009). For instance, the words “sports” and “spotrs” would not be recognized as even remotely similar by lexical analysis, but in character analysis, there are still common n-grams between the two spellings. The frequency of character n-gram analysis has been shown as a very important stylistic feature (Stamatatos, 2009). One of the big questions, though, is finding the optimal “n” to use in the analysis. Larger values of n find more lexical and contextual information, but at the expense of increased dimensionality. The choice of natural language at hand also plays a role in the selection. Character analysis generally involves the use of the same compression algorithm on all the texts at hand, since texts written by the same author will likely reduce to a similar bit size (Stamatatos, 2009).

1 Compression algorithms will be explored in Section 4
3.4. Syntactic and Semantic Features

Two more complex methods for stylometric analysis are syntactic features and semantic features. Syntactic features exploit the fact that authors generally deploy similar syntactic patterns in their writing (Stamatatos, 2009). Theoretically, this reason makes syntactic information more reliable than lexical features for authorship attribution analysis. However, the problem is that obtaining these features is computationally more expensive, requiring the use of complex natural language processing (NLP) tools. Even common syntactic models like context-free grammars, which represent a language as a set of rewrite rules with abstract symbols, have been shown to be too computationally intensive, and even fail to encapsulate important dependencies in authorial style (Juola, 2007). Semantic analysis involves relating different syntactic features to one another, which is an even more complex process than syntactic analysis. While NLP technology tends to handle low-level feature analysis well, it still is not at a point where it can process such deep linguistic relationships.

3.5. Application-Specific Features

In certain situations, it can be useful to consider application-specific features of texts. For example, if analyzing emails, looking at structure and layout (e.g. indentation, signatures, etc.) may provide valuable clues to an individual’s authorial style. When analyzing computer code, looking at the use of comments, placement of curly braces, and other non-core features to a program also offer similar benefits (Juola, 2007). Application-specific features are not as general and are vulnerable to editorial changes, but nevertheless can be worth examining. A similarly creative area to examine is that of “metadata.” When writing a document on Microsoft Word, for instance, the program will record the time the document was written, its original name and date,
and other primitive information. This data is useful for author analysis, such as determining an author’s preferred time to write. There is limited consensus on the best ways to utilize application-specific features, but this is one of the main areas researchers are currently exploring.

3.6. Summary of Linguistic Background

There are a wide variety of different features that can be used in authorship attribution analysis. The selection and identification of the proper feature set is absolutely essential for running meaningful analytics thereafter. Part of what makes this selection so difficult is that there is no general consensus on the best features to use. Typically, it is best to pick features that are frequent and unstable. Frequently appearing features tend to capture more stylistic variation (Stamatatos, 2009). Unstable features, which represent features for which an author has the most choice (e.g. picking between synonyms) also give strong insights into one’s authorial style. Ultimately, whatever feature set is used will be reduced to a group of ordered vectors, which are subsequently analyzed by a variety of machine learning methods to determine similarity (Juola, 2009). Thus, the next core question becomes which of these techniques to use.

4. Survey of Unsupervised Learning Methods

Once a linguistic feature set has been selected from the texts at hand, the next step in the authorship attribution problem is to pick an analytical technique that can utilize this data to determine the author of the unknown text. This thesis focuses on a few powerful machine learning methods that modern authorship attribution studies implement, beginning with an overview of different attribution techniques and their associated type of machine learning models. Machine learning is a type of data analysis in which a system intakes training data and
identifies patterns within it, which it uses to make predictions and categorization about future data points of the same topic. Machine learning methods are classified as unsupervised or supervised. Unsupervised learning models find innate patterns in unlabeled datasets and classify new points based on how well they match those patterns. Supervised learning models use pre-labeled data to make an explicit boundary between classes that classifies new points. This section will examine Principal Components Analysis as an example of unsupervised learning, followed by a brief discussion of other unsupervised learning approaches. The following section will introduce supervised learning, followed by a technical analysis of Support Vector Machines, which is one of the most successful modern machine learning methods.

4.1. Profile-Based Approaches

Historically, most analytical methods used in authorship attribution were classified as profile-based approaches, in which all the identified texts for a given author are concatenated into a single text file from which that author’s style is deduced. Mosteller and Wallace, for example, implemented this methodology in their analysis of ‘The Federalist Papers.’ Since profile-based approaches consider all of an author’s text as a single entity, the individual differences between those training texts become disregarded. In fact, the features extracted from this joint text may end up being rather different from each individual text sample. In Figure 1 (below), we see the concatenation of texts that creates a profile for each author. With these profiles, some unsupervised learning method then creates a probabilistic similarity estimation, to which texts of unknown authorship are applied and thereby given an author.
Profile-based approaches build a profile for each candidate author by analyzing the common style markers that appear across all of that author’s works. Then, when the unknown text is analyzed, its authorship is attributed to the author profile with which it shares the most similarity. This process is precisely what unsupervised learning does, which is one of the two kinds of machine learning. Unsupervised learning methods use the training data\(^2\) to learn a joint probability distribution, which means each class\(^3\) is assigned its own probability (Ng et al., 2001). Then, once the unknown text has been analyzed, these methods use Bayes Theorem in conjunction with the learned probability distribution to identify the most likely author, which is calculated as the conditional probability (Ng et al., 2001). Effectively, unsupervised learning models look for superficial patterns across the training dataset and classify new data points depending on whether or not they contain these same patterns. Another popular term for this

---

\(^2\) In authorship attribution, training data are the texts of known authorship for each candidate author  
\(^3\) In authorship attribution, classes represent authors
methodology is generative modeling, as it generalizes training data in order to make its classification (Hastie et al. 2017).

For comparison, supervised learning methods take in pre-labeled data points and create an explicit boundary between classes that classifies new data points, rather than assigning probabilities to each class as the means for classification. Most modern authorship attribution methods consider each text as an individual representation of an author’s style, giving way to the name of instance-based approaches, which utilize supervised learning techniques. The difference between profile-based and instance-based approaches is the highest-level distinction that can be made between attribution methods, because it determines the underlying mathematics behind those models.

4.2. Principal Components Analysis

One popular unsupervised learning method is Principal Components Analysis (PCA). In PCA, each training text’s feature set is considered a vector within a multivariate vector space (Juola, 2007). In authorship attribution, the feature set generally used is word frequencies of function words (Burrows, 2002). PCA works by reducing each of these potentially correlated vectors into a smaller set of uncorrelated vectors, which are called the principal components (Hastie et al., 2017). The first principal component accounts for the most significant amount of variance in the data, and each subsequent principal component accounts for the corresponding maximal amount of remaining variance. In other words, PCA simplifies a dataset by identifying a smaller set of basis vectors that describe as much of the original data as possible (Juola, 2007). Thus, PCA can be successful even with large datasets because of its ability to reduce dimensionality. For two-dimensional visual purposes, a common practice is to plot the two basis
vectors with the largest eigenvalues\(^4\) against one another. This cluster of points is then often subjected to cluster analysis to determine the probabilities for each class (Can, 2014). PCA has been used significantly in authorship attribution and has had high success rates.

4.2.1. PCA in Authorship Attribution

One well known study that implemented PCA was done in 2003 by Jose Binongo on the disputed authorship of the 15th Book of Oz. The Oz books were a series written by Lyman Frank Baum that began with the famous *The Wonderful Wizard of Oz*, published in 1900. Over the next 20 years, Baum would go on two write 13 more books building on his initial bestseller. Baum’s health deteriorated as he aged, and he eventually suffered a stroke before being able to complete his 15th book, *The Royal Book of Oz*, which he had started prior to his death. The publishers of the Oz series needed someone to complete the book, so they chose successful children’s writer Ruth Plumly Thompson to take on the task. When it was ultimately published in 1921, Baum was given credit for writing it while Thompson was said only to have “enlarged and edited” the work (Binongo, 2003). Decades later, experts began to question this authorship and suspected that Thompson wrote the entire book on her own.

Binongo obtained the texts for Baum and Thompson from Project Gutenberg’s website, which contains many machine-reading versions of texts. He decided to use a feature set of the frequencies of 50 most commonly occurring function words (Binongo, 2003). Since it’s not possible to visualize a 50-dimension vector space, PCA was used to reduce this space into two dimensions, where natural clustering occurred that helped distinguish the author. The first PC

\(^4\) An eigenvector of a linear mapping is a non-zero vector that only changes by a scalar amount (the eigenvalue), when the linear mapping is applied to it. Eigenvalues are used as a measure of variance.
(PC) accounted for 20% of the variation, while the second PC accounted for 7%. The two-dimensional cluster representation of this analysis can be seen below in Figure 2.

![Figure 2. Distribution of points for Baum and Thompson (Binongo, 2003)](image)

This analysis helped reveal that Thompson tended to use function words indicating position (such as “up”, “down”, “over”, etc.) more than Baum; in fact, almost twice as much as Baum (Binongo, 2003). Baum preferred to use the words “which” and “that” and had a higher usage of negative words like “but” and “not” (Binongo, 2003). Thus, when the Royal Book of Oz was added to the scatterplot, it fell on the side of Thompson’s works, statistically indicating that she was the likely author of the work. This has since become the generally accepted truth about the authorship of the 15th book of Oz.

4.3. Other Unsupervised Learning Methods

PCA is one of the most effective unsupervised learning methods in use today. However, there are other methods in this category worth discussing. This paper will briefly introduce
multidimensional scaling, cluster analysis, and compression models before moving onto an in-depth analysis of Support Vector Machines, a commonly used supervised learning technique.

Another approach to unsupervised learning involves calculating intertextual differences as distances, which are interpreted as measures of similarity. It turns out that for true distances, it is possible to embed this distance structure into a higher-dimensional space while preserving the properties of the original distances (Juola, 2007). Multidimensional Scaling (MDS) is the process of embedding distances into a higher dimensional space that is better suited for the observation of patterns. In 2008, Patrick Juola published a study titled *Becoming Jack London* that found a clear stylistic change in London’s writing between his earlier and later works. Specifically, he found the turning point to be 1912. Juola used MDS to find a set of points in a three-dimensional space that well-mimicked the inter-document distances, from which the 1912 separator was found (Juola, 2008). This technique has not been widely used in authorship attribution, but still serves to show the scope of possible unsupervised learning techniques in use.

Cluster analysis, like MDS, uses intertextual distances as its basis. It groups the closest pairs of items into a cluster and then considers that cluster as its own single data point. This process repeats, where on each iteration the number of points is reduced by one, until there is just one single cluster remaining in the spot best representing all of the points. A 2017 study by a group of Indian professors applied cluster analysis on English news articles, obtaining 97% accuracy with K-Means clustering (Rao et al., 2017). Cluster analysis is not used too much on its own for authorship attribution, but is often combined with PCA, as mentioned above.

Finally, compression models are another example of unsupervised learning that encode the input data with fewer bits than the original representation (Hastie et al., 2017). In profile-based approaches, compression algorithms turn the original concatenated file into a compressed
file, and the difference in bit size is used as a measure of similarity. Many different compression algorithms have been used for this task (RAR, LZW, GZIP, BZIP2, and others) and RAR is generally found to be the most accurate (Stamatatos, 2009). RAR uses prediction by partial matching (PPM), which uses a set of previously seen characters in the uncompressed text file to predict the next characters. In 2001 a group of Russian researchers applied different data compression algorithms to 385 texts of 82 authors, finding RAR to perform the best (Kukushkina, 2001). The analysis of such algorithms is beyond the scope of this paper, but nevertheless data compression is another unsupervised learning technique that has had some success in authorship attribution.

4.4. Summary of Unsupervised Learning

Unsupervised learning is ideal for exploratory analysis where no data labels exist beforehand. These models can identify inherent patterns and structures in the data. In highly complex datasets, they help uncover insights that humans would not be able to find and work well to reduce the dimension space. However, in the classic authorship attribution problem, we do have labels for the training texts, namely the authors of each training text, so it makes sense to take advantage of this additional information. For this reason, instance-based approaches utilizing supervised learning tend to outperform the profile-based approaches using unsupervised learning. The following section will introduce instance-based approaches before analyzing the theory and applications of Support Vector Machines.
5. Foundations of Support Vector Machines

5.1. Instance-Based Approaches

In the standard authorship attribution problem, a text of unknown authorship is assigned an author from a set of candidate authors for whom text samples of known authorship are available. In the context of machine learning, this means that each training text is already labeled. This labeling enables the use of supervised learning techniques, which require that the training data be categorized prior to analysis. Instance-based approaches consider each text of an author separately. Each individual text can be thought of as a vector of features which contributes to the attribution model at hand. Rather than assigning individual probabilities to each class, supervised learning methods create an explicit boundary which can classify any new data points. A visual depiction of this methodology can be seen in Figure 3 below.

![Figure 3. Standard methodology for instance-based approaches (Stamatatos, 2009)]
Instance-based approaches build a profile for each candidate author by analyzing the style markers for each author’s works, which are considered as individual entities. They use the pre-known authorship of the training texts to create explicit boundaries between the classes. Then, when the unknown text is analyzed, its authorship is classified based on where the boundaries place it. This is the classic supervised learning process: supervised learning methods use labeled training data to learn a conditional probability distribution of each class (Ng et al., 2001). Then, any new data points are classified based on what this distribution dictates (Ng et al., 2001). In general, supervised methods outperform unsupervised ones because of their ability to utilize training data labels (Hastie et al., 2017). Another popular term for this methodology is discriminative modeling, as it explicitly discriminates between classes because of its prior knowledge from the data labels.

5.2. Background in Linear Classification

As stated above, supervised learning models create an explicit boundary between classes, rather than assigning individual probabilities to each class. Therefore, supervised learning is generally used for the purposes of classification, which is the process of assigning new data points to classes based on the assumptions of the labeled training data (Dasgupta, 2018). Support Vector Machines are a linear classification method, meaning they implement a linear predictor function to make their classification. In machine learning terminology, this decision boundary is called a hyperplane\(^5\) used for binary decision making. For a simplistic example, on the left of Figure 4, line L is a one-dimensional hyperplane in the two-dimensional space which separates the two classes. A discussion of linear classification is necessary in order to understand SVMs.

\(^5\) A hyperplane is a subspace whose dimension is one less than that of its surrounding (ambient) space
The equation for a line is given by

\[ y = mx + b, \]  
\[ (1) \]

where \( m \) is the slope of the line, and \( b \) is the \( y \)-intercept. This can be rearranged as

\[ mx + b - y = 0. \]  
\[ (2) \]

Recall that in instance-based approaches, each text is considered as a vector of features. This feature set can be of arbitrarily many dimensions, which we will call \( d \). Thus, we will name a vector \( w \in \mathbb{R}^d \). Note then that we can now rewrite equation 2 with a dot product of \( w \) and \( x \), as

\[ w \cdot x + b = 0. \]  
\[ (3) \]

This equation will be the form of all future linear decision boundaries.

Let us assume we have some decision boundary of this form that separates two classes, and now want to classify points. All points on one side of the line will be labeled as positive, with all points on the other side labeled as negative, as seen on the right of Figure 4. Each new point is plugged into the equation, generating some real number value. If the value is positive, that point is given a positive label, and if the value is negative, the point is given a negative label. More generally, for each point \( x \), the resulting value from the linear function yields the label sign.
for x, which we will call y. Our classifier is correct whenever it properly predicts the true label sign for a point x. More formally, it is correct when the following condition holds

\[ y (w \cdot x + b) > 0. \]  

(4)

When y is negative, that means x is on the negative side of the line, and thus the resulting product of these negative values is positive. When y is positive, x must lie on the positive side of the line, and that resulting product is clearly positive. Thus, this equation will be used to evaluate the classifier’s correctness.

Since all the training data are labeled beforehand, we can objectively evaluate the success of the linear classifier on each data point and update it accordingly. The first linear boundary that is chosen is unlikely to perfectly classify every point. Linear classifiers are given loss functions to quantify the error on mislabeled points. There are many ways to quantify error, but one simple one is to have the penalty equal the amount by which the classifier is wrong, which is equal to

\[-y (w \cdot x + b). \]

(5)

Thus, the goal of a linear classifier is to minimize the loss function, thereby making this an optimization problem. A simple example of this method is Stochastic Gradient Descent (SGD). In SGD, w and b are initialized to some random values, the first data point is tested, and if there is a loss, w and b get updated slightly\(^6\), and the process repeats until all points are properly classified (Dasgupta, 2018).

When the step size, which is the constant in the partial derivatives for w and b, is set to 1, the result is the simple but powerful Perceptron algorithm, developed by Cornell University researcher Frank Rosenblatt in 1957 (Rosenblatt, 1957). Pseudocode for this algorithm is given

---

\(^6\) The updates move w and b in the negative direction of the partial derivatives for w and b in -y (w * x + b)
below. The Perceptron algorithm is guaranteed to find a linear classifier if the data is linearly separable. With this background in linear classification, we can now properly analyze SVMs.

**Perceptron Algorithm Pseudocode**

Initialize $w$ and $b$ to 0
Cycle through training data $(x,y)$
    If $y(w \cdot x + b) \leq 0$ (i.e. the point is misclassified)
    
    $w = w + yx$
    $b = b + y$

5.3. Hard-Margin Support Vector Machines

The Perceptron algorithm is guaranteed to find a linear classifier, but not necessarily the optimal one. In the right portion of Figure 4, the linear classifier is extremely close to the negatively labeled points. Theoretically, a new point just to the right of the boundary would get classified as positive, even though it would be closer to the negative points. The ideal linear classifier is central between the classes and has the most buffer around it, as this helps generalize any new data points most accurately. In other words, it seeks to maximize a margin, which is the space between two hyperplanes. Margin maximization is precisely what Support Vector Machines (SVMs) seek to do. In Figure 5, the dark line in the center is the linear boundary generated by the SVM, with equidistant hyperplanes on each side. The width of the margin is given by $1/||w||$. Thus, by minimizing $w$, the margin is maximized, thereby yielding another convex optimization problem. The data points directly on the margin are called support vectors. SVMs are so powerful because the linear classifiers they generate are functions only of the support vectors. Thus, even with extremely large datasets, only the support vectors need to be considered to find the optimal linear decision boundary. The term hard-margin refers to the fact

---

7 Pseudocode adopted from Dasgupta, 2018
that the data must be linearly separable and no points can cross over the dotted hyperplanes. The case where data crossover does occur is examined in the following section on soft-margin SVMs.

Recall from the prior section that the linear separability condition (LSC) was to find values \( w \in \mathbb{R}^d \) and \( b \in \mathbb{R} \) such that for all training data points \( i \)

\[
y(i)(w \cdot x(i) + b) > 0. \tag{6}
\]

By scaling \( w \) and \( b \), we can generate the equivalent LSC of

\[
y(i)(w \cdot x(i) + b) > 1. \tag{7}
\]

This new LSC makes it easy to generate equations for the parallel hyperplanes used in SVMs. Specifically, these hyperplanes will be of the form

\[
w \cdot x + b = 1 \text{ and } w \cdot x + b = -1. \tag{8, 9}
\]

All positively labeled points must be on or above the top hyperplane, represented by equation 8, while all negatively labeled points must be on or below the bottom hyperplane, represented by equation 9. Therefore, no points can exist in the space between these two hyperplanes, and that is precisely the distance (margin) that must be maximized.

Figure 5. Visual depiction of the hard-margin SVM (Bagchi, 2014)
The goal of the SVM is to minimize $||w||^2$ such that the LSC holds for all training data. In mathematics, the theory of duality states that every optimization problem has a dual representation with the same optimal solution (Dasgupta, 2018). Since the primal problem here is a minimization one, the dual problem will be a maximization one. The new equation is now

$$w = \sum_{i}^{n} \alpha_i y^{(i)} x^{(i)}. \quad (10)$$

where each $\alpha$ is Lagrange multiplier coefficient for each data point (Hastie et al., 2017). What makes this new equation so important is that each $\alpha$ is nonzero only for the data points right on the margin, and zero for all other points. Thus, $w$ is now expressed as a function just of support vectors. This core property enables SVMs to find optimal linear decision boundaries in even very large datasets, just by considering the support vectors. As we will see in the next section, this same property holds even when the training data is not linearly separable.

5.4 Soft-Margin Support Vector Machines

In most real-world situations, data will not be perfectly linearly separable. In these situations, the hard-margin SVM will fail to find a linear classifier for the training data. In Figure 6 below, for instance, one green data point crosses over one of the parallel hyperplanes, while one of the blue points completely crosses over the decision boundary. In order to handle crossover between the data, the soft-margin SVM introduces slack variables (labeled by $\varepsilon$) for each data point to capture the extent to which the LSC has been violated (Dasgupta, 2018). In the soft-margin SVM, support vectors are now defined as all data points exactly on the margin and on the wrong side of the margin.

---

8 Lagrangian Duality is a highly complex mathematical topic and not the focus of this thesis. Please reference the cited work to see a complete derivation of the Lagrangian equation.
The soft-margin SVM still yields an optimization problem, but a less straightforward one. Before, the goal was to minimize $||w||^2$, but now the problem becomes minimizing the sum of

$$||w||^2 + C \sum \xi_i. \quad (11)$$

$C$ is a number that we set which determines how expensive slack is. As $C$ grows, slack becomes more expensive. Technically, an infinite value of $C$ yields the hard-margin SVM, where no crossover among data points is allowed. A smaller value of $C$ enables a larger margin, but at the expense of more constraints being violated.

Cross validation is generally the process used to find the optimal value of $C$. In machine learning, cross validation works by training the given model on subsets of the training data and then testing it on the remainder of the training data (Dasgupta, 2018). For example, in Figure 7, the training data is split into five groups, and there are five separate experiments that are conducted where the model is trained on one group, with the rest left for testing. After each experiment, the error value is recorded, and then at the end of the last experiment, the average error number is calculated. This will be the error value for one value of $C$. Then, this entire cross validation process is repeated for different values of $C$. Ultimately, the $C$ with the lowest average error value is the one that will be used for the soft-margin SVM. This process enables SVMs to
avoid the problem of overfitting, where a machine learning method too specifically models the training data, thereby preventing it from generalizing to any new data points.

The soft-margin SVM is a highly competent linear classifier that functions well even when data is not linearly separable. In fact, this method is far more powerful than meets the eye. The next section will introduce the topics of basis expansion and kernels, which allow the soft-margin SVM to create more complex boundaries, such as quadratic and polynomial ones, in situations where the training data is far from being linearly separable.

5.5 Basis Expansion and Kernels

In many situations, especially with very large datasets, it is highly unlikely that the training data will even be close to linearly separable. Under these circumstances, the soft-margin SVM will need to be adjusted to account for the high levels of noise and deviation in the data. The adjustment is called basis expansion, which is the process of embedding data of one dimension into a higher dimension, in which a linear hyperplane can then be found to separate the data. A picture showing the intuition behind embeddings, basis expansion and kernels can be seen below in Figure 8.

Figure 7. Visual depiction of cross validation methodology (Becker 2017)
Basis expansion with SVMs works on the dual form of the soft-margin SVM representation, except each instance of $x$ is substituted for its expanded version, which will be called $\phi(x)$. This optimization problem is explicitly stated below in Figure 9. The dual form enables us to use the $\alpha_i$ values like before, which prevents us from having to work with $w$, which will be of a much higher dimension. Instead of working with $w$, the solution entails computing dot products between these expanded versions of $x$, which is a far more efficient process than solving the primal form of the problem. For any points $x$ and $z$ in the original feature space, the value of the kernel function $k$ equals the dot product

$$k(x, z) = \phi(x) \cdot \phi(z).$$

(12)

**Figure 8.** An example of basis expansion via a quadratic kernel (Dante, 2017)

**Figure 9.** The Lagrangian dual representation of the soft-margin SVM (Dasgupta, 2018)
In general, for polynomials of degree p, the expanded vector $\phi(x)$ will represent all possible combinations of terms in $x$ up to degree p. It turns out that any degree-p polynomial in $x$ is linear in $\phi(x)$ and vice versa (Dasgupta, 2018). Therefore, we can theoretically always embed the dataset into a high-enough dimension for a linear classifier to be found. For polynomials of too high a degree, writing out all possibilities for $\phi(x)$ would be far too difficult of a task. This is precisely why the dual-form of the SVM is used; we need only to consider the dot products of the expanded vectors that is computed with a kernel function, rather than having to write out these expanded versions. As an example of this relative simplicity, a valid kernel function to compute dot products for polynomials up to order p is

$$k(x, z) = (1 + x \cdot z)^p$$  \hspace{1cm} (13) \hspace{1cm} (Dasgupta, 2018). This equation is formally known as the inhomogeneous polynomial kernel.

Kernel functions effectively act as measures of similarity by showing how close two data points are to one another\(^9\). As one might imagine, the choice of kernel function becomes extremely important, especially for complex datasets with unclear relationships between the points. It turns out that $k$ can be any similar function $k(x, z) = \phi(x) \cdot \phi(z)$ for some embedding $\phi$ as verified by Mercer’s Condition, which is stated below in Figure 10 for clarity. Effectively, this condition checks that the $m \times m$ matrix created by combinations of the similarity function $k$’s output is non-negative for every non-zero column vector (Hastie et al., 2017).

\[ Mercer’s \text{ condition}: \text{ same as requiring that for any finite set of points } x^{(1)}, \ldots, x^{(m)}, \text{ the } m \times m \text{ similarity matrix } K \text{ given by } \]
\[ K_{ij} = k(x^{(i)}, x^{(j)}) \]
\[ \text{is positive semidefinite}. \]

\textbf{Figure 10.} Statement of Mercer’s Condition for kernel functions in SVMs (Dasgupta, 2018)

---

\(^9\) This is because that the closer two vectors are aligned, the higher their dot product will be
One of the most popular kernels is the Gaussian kernel, also called the Radial Basis Function (RBF) kernel. On any two data points \( x \) and \( y \), the Gaussian kernel is given by

\[
k(x, z) = e^{-\frac{|x-z|^2}{s^2}},
\]

where \( s \) is an adjustable scaling parameter (Hastie et al., 2017). While this may initially look fairly complex, it is actually a rather intuitive notion of similarity. When \( x \) and \( z \) are equal, the exponent becomes zero, making the value of the function equal to one. As the distance between \( x \) and \( z \) grows, \( k(x, z) \) becomes smaller and smaller, with a limit of 0. Therefore, all values of the Gaussian kernel lie between 0 and 1. The value of \( s \) works similarly. As \( s \) goes to infinity, the exponent goes to 0, meaning the similarity between all points is one. On the flip side, as \( s \) gets close to 0, the exponent becomes a massive number, which magnifies the difference between the given two points\(^{10}\). In the Gaussian kernel computation of the dot product \( \phi(x) \cdot \phi(z) \), it turns out that the extended feature vectors are infinite dimensional, but these never need to be constructed because the formula given above is simply used instead. Much of the widespread use of the Gaussian Kernel is due to the fact that it can classify each point in the data set individually. Furthermore, the boundaries generated through the Gaussian kernel are very smooth\(^{11}\), simplifying and expanding the mathematical analysis that can be done on them.

Basis expansion through kernels is one of the most important and fascinating topics in all of SVMs. The underlying mathematics guarantee that nonlinear classification in some dimension space can always be transformed into a linear classification in a higher dimension space, which is why SVMs are so versatile and widely used. SVMs work better in higher dimensions, but this is only possible with larger training data sets, which are not always possible to obtain. Overall,

\(^{10}\) This is similar to a simpler machine learning method called the Nearest Neighbor Classifier

\(^{11}\) A smooth function has continuous derivatives up to some desired order (WolframAlpha)
SVMs avoid the machine learning pitfalls of overly complex dimensionality and overfitting data points and are considered the best modern linear classifier.

5.6 Support Vector Machines in Authorship Attribution

Support Vector Machines have been used widely in authorship attribution and have performed extremely well. In fact, they have worked so well that some researchers argue that it is worth ignoring all other techniques and using just SVMs (Juola, 2007). One of the best studies of SVM-driven authorship attribution was conducted by Australian professor Joachim Diederich titled *Authorship Attribution with Support Vector Machines*. This study applied SVMs and a few other methods (decision trees, neural networks, Linear Discriminant Analysis, and PCA) onto a series of texts from a German newspaper. SVMs performed so well that using just using word frequencies as the only feature set was sufficient to identify the correct authors. All other more complex feature sets and feature-weighting techniques yielded nearly identical results (Diederich et al., 2003). Diederich found that SVMs are especially well suited for authorship attribution because of their ability to process very large texts and databases of large texts. Training time was also found to be comparable to all other methods used in the study.

In 2004, the Association for Literacy and Linguistic Computing (ALLC) and the Association for Computers and the Humanities (ACH) hosted the “Ad-hoc Authorship Attribution Competition” to test and compare the different methodologies being used in the field (Juola, 2007). The contest consisted of 13 distinct problems varying in length, style, genre, and language. The top scoring participants in this contest were Israeli computer scientists Moshe Koppel and Jonathan Schler, who used a SVM with a linear kernel function (Juola, 2007). For English texts, their feature set was unstable word choice, representing words with common substitutes (Juola, 2007). In non-English languages, they used the 500 most frequent words as
the feature set. Their average success rate was 71%, which was considered excellent in 2004. These accuracy rates quickly grew over the years. Just three years later in 2007, a study at Bilkent University applied SVMs to a bag-of-words\textsuperscript{12} feature set, achieving a 95% success rate, finding that SVMs worked well in generalizing diverse topics with different vocabulary (Bozkurt et al, 2007). Today, it is widely considered the best method available for authorship attribution.

6. Conclusion

Authorship attribution is the process of identifying the author of an unknown text. This area existed for hundreds of years as a linguistics field, but has since become a highly computational field following the analysis of ‘The Federalist Papers’ by Mosteller and Wallace in 1964. In the typical authorship attribution problem, there will be a set of candidate authors for whom text samples of known authorship are available. The analytical process involves selecting some set of linguistic features for each text, called the feature set, and then selecting some computational method onto which the feature set is applied.

There is no generally accepted answer for the best linguistic feature set to pick. However, it is usually best to pick features that are frequent and unstable, because frequently appearing features capture more stylistic variation and unstable features represent features for which an author has the most substitutes available (like synonyms). The feature set selected ultimately gets reduced to a group of ordered vectors, and this vector space becomes the input for the different computational models that attribute authorship.

The best methods for attributing authorship are machine learning models, of which there are two main kinds. Unsupervised learning techniques find innate patterns within datasets, while

\textsuperscript{12} The bag-of-words language model considers a text as a collection of words, disregarding all grammar and occasionally word order (Bozkurt et al., 2007)
supervised learning techniques use pre-labeled training data to build an explicit boundary between classes. Some unsupervised learning models include PCA, MDS, Cluster Analysis and Compression Algorithms, all of which have been applied to authorship attribution with some success. Supervised learning methods like SVMs have been applied the most to authorship attribution and have achieved the most success.

While some machine learning techniques like SVMs generally outperform other ones, no one method can be used alone to solve this problem. Rather, it is optimal to use a variety of techniques and compare the results to avoid the machine learning problems of overfitting and dimensionality. Additionally, it is important to consider the context in which authorship attribution studies are conducted. In order for the results of a study to be meaningful, the study should have enough training texts of sufficient size that are well distributed over the candidate authors, along with a sufficient length for the unknown text(s) (Stamatatos, 2009). The best authorship attribution methods are the most generalizable ones, meaning they can perform well on different genres and natural languages (Stamatatos, 2009).

While the results of authorship attribution studies continue to gain accuracy and application-specific methodologies are improving, there are still a variety of important, unanswered questions in the field. For example, there is no consensus about the minimum length a text must be in order for its stylometric features to be sufficiently captured (Stamatatos, 2009). Many researchers are focusing on the ability of certain features to capture information on authorship, text-type (e.g. newspaper vs. book), and topic, and the tradeoffs between features that account for either some or all of these categories (Stamatatos, 2009). Another area of recent exploration is the ability of methods to be trained on texts from one genre and then applied to texts of a different genre by the same authors, which is highly useful for forensics (Juola, 2007).
As NLP tools improve and more studies are conducted, one can only imagine the variety of different ways that texts will be extracted for insights and applied to other areas thereafter.

While once not seen in the best light, the field of authorship attribution has grown significantly in recent years as machine learning methods have advanced and the evaluation criteria for authorship attribution studies have evolved. The future of the field appears to be headings towards the development of highly generalizable methodologies and the improvement of application-specific methods. As such, authorship attribution presents a unique opportunity for researchers and scientists to solve one of the most complex problems in information retrieval, and will create a plethora of applications for many fields going forward.
References


Rosenblatt, Frank (1957), The Perceptron--a perceiving and recognizing automaton. Report 85-460-1, Cornell Aeronautical Laboratory.