2003

Using Natural Language Parsers for Authorship Attribution

Westerly A D Magnera

Rochester Institute of Technology
Department of Computer Science
Thesis

Using Natural Language Parsers for Authorship Attribution

by Westerly A.D. Magnera

A thesis submitted to
The Faculty of the Computer Science Department
in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science

Approved by:

Dr. Edith Hemaspaandra
Chairman

Myroslava Dzikovska
Reader

Dr. Carol Marchetti
Observer

Rochester Institute of Technology
Wallace Memorial Library
Thesis Reproduction Permission Statement

Title of Thesis
Using Natural Language Parsers for Authorship Attribution

I, Westerly A.D. Magnera, prefer to be contacted each time a request for reproduction is made. If permission is granted, any reproduction will not be for commercial use or profit. I can be reached at the following address:

29 Avalon Drive
Rochester, NY 14618
(585) 271-7857

Signature of Author: _____ Date: 6/23/03

Acknowledgments

My sincerest appreciation and gratitude to Myroslava Dzikovska for going above and beyond the call of duty. Her continuous feedback, comments, and corrections were invaluable as I tried to learn all I could about natural language processing. This thesis is significantly better because of her efforts and might never have been finished if it was not for her guidance and help.

Gratitude and regards also go out to Edith Hemaspaandra. Her comments were always helpful and her support though the entire thesis invaluable.

Many other professors also helped in the development of this thesis, especially Carol Marchetti. I wish to thank her for her help with the statistical aspects of this thesis. Her assistance with deciding which statistics to run and her reassurance that the results were reasonable were of great help.

Many others deserve thanks for putting up with me during these last few months. I wish to thank my husband who put up with a great number of late nights alone as I worked furiously away, my friends who put up with me cancelling events at the last minute and missing others altogether, and all my co-workers who put up with me and my daily highs and lows as I work on this project.

Abstract

The goal of authorship attribution is to find a set of unconscious writing characteristics or style features that distinguish text written by one person from text written by another. Once these features are found, they can be used to pair a text with the individual who wrote it. It is now well accepted that authors develop distinct and unconscious writing features. Over one thousand stylometric features (style markers) have been proposed in a variety of research disciplines[44] but none of that research has looked at the syntactic structure of the text. I conjectures that the distinct writing features of an author are not limited to these features already studied, but also include syntactic features. To support this hypothesis, I ran experiments using two open source parsing programs and analyzed the results to see if features given to me from these programs were enough for me to determine who is the most probable author of a text.

Parsing programs are designed to determine syntactic structures in natural language. They take a text or a writing sample and produce output showing the grammatical relationship between the words in the text. They provide a means to test the hypothesis that authors' syntactic use of words provide enough identifying characteristics to differentiate between them.

Using two open source natural language parsing programs, the Link Grammar Parser and Collins' Parser, this research tested to see if an authors sentence structure is unique enough to provide a means of recognizing the probable author of a text. Initial data was collected on a pool of test authors. Sample texts by each author were run through both parsers. The output of each parser was analyzed using two multivariate analysis methods: discriminant analysis and cluster k-means.

My results show that syntactic sentence structures may be a viable method for authorship attribution. The Link Grammar shows promise as a way to augment authorship attribution methods already out there. Collins' Parser provided even better results that should be solid enough to stand on their own as a new and viable alternative to methods that already exist. Collins' parser also provided new predictors that might improve current authorship attribution methods. For example, elements and phrases with wh- words and the length of noun phrases are highly corrolated with authorship in this study.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The goal of authorship attribution is to find a set of unconscious writing characteristics or style features that distinguish text written by one person from text written by another. Once these features are found, they can be used to pair a text with the individual who wrote it. It is now well accepted that authors develop distinct and unconscious writing features. Over one thousand stylometric features (style markers) have been proposed in a variety of research disciplines[44] but none of that research has looked at the syntactic structure of the text. I conjecture that the distinct writing features of an author are not limited to these features already studied, but also include syntactic features. To support this hypothesis, I ran experiments using two open source parsing programs and analyzed the results to see if features given to me from these programs were enough for me to determine who is the most probable author of a text. I then compared my results with some current authorship attribution methods.

Traditional authorship analysis methods include handwriting analysis, latent semantic indexing (LSI), analysis of entropy, and an analysis of the frequency and probability of words and word combinations. Computer programs can collect and analyze the data for most of these methods. Simple word counting programs can give the frequency of specific words[1] and pattern searching programs can find the probability of word combinations[2]. Programs such as zip provide a quick way to approximate entropy values[3].

[1]See Section 2.2.1 for more information on word frequency and authorship attribution methods

[2]See Section 2.2.2 for more information on word combinations and collocations.

[3]See Section 2.2.3 for more information about entropy and how it is used in authorship

LSI techniques collect much more data about the context in which words are and are not used[4]. Modern computer vision programs can distinguish some handwriting styles.

Syntactic patterns are more difficult for computers to recognize than entropy, word frequency, and word patterns. This is because natural languages are naturally ambiguous and computers are less proficient at resolving that ambiguity than humans. For humans, the ambiguity may *not* even be apparent since humans naturally integrate the text with knowledge about the situation and the world. For example, the sentence "*Flying planes make Jill duck.*" cause no problem to most readers. Planes in the air cause her to crouch down. Few would assume Jill ducks while she is flying a plane and even fewer would be concerned that Jill might become a duck. Parsing programs are computer programs that accept a sentence and find the most probable parse for it. By finding the proper parse, the sentence can be understood. For example, the word *duck* from our previous sentence could be incorrectly parsed as a noun or correctly parsed as a verb. Generally parsing programs take a text or a writing sample and produce output showing the grammatical relationship between the words in the text. This relationship indicates the usage of each word in the context of the sentence, the length and form of sentence constituents, and where these constituents exist within the sentence. Parsing programs are now reasonably good at dealing with ambiguity and are readily available as share-ware. They provide a means to test the hypothesis that authors' syntactic use of words provide enough identifying characteristics to differentiate between them.

Using two open source natural language parsing programs, the Link Grammar Parser and Collins' Parser, this research tests to see if an author's sentence structure is unique enough to provide a means of recognizing the probable author of a text. Initial data was collected on a pool of eight test authors. At least two sample texts by each author was run through both parsers.

The output of each parser was analyzed using two statistical methods: discriminant analysis and cluster k-means. Discriminant analysis first defines each author by a set of predictors. Once each author has a unique definition, a new text is classified as belonging to one of the authors. The success rate of each specific set of predictors is based on how many texts are classified as

attribution

[4]See Section 2.2.2 for more information about LSI and how it is used in authorship attribution

written by the same author that actually wrote them. Cluster k-means uses a set of predictors to group the texts. Unlike discriminant analysis, though, cluster k-means never defines initial groups. Instead, it creates groups based on the predictors. The groups that cluster k-means returns can then be match with the real groups of authors to see if they match.

The Link Grammar provided interesting results. Sets of links were used as predictors for both the discriminant analysis and the cluster k-means.

The results of the discriminant analysis were that three out of four texts were then correctly classified as by the author who wrote them. Seventy-five percent is not high enough to be used as an authorship attribution method on its own, but when used with other authorship attribution techniques, such as average number of words per sentence, using links as predictors can be used in some applications for authorship attribution. It is also interesting to note that the links found most valuable as predictors are closely related to function words and punctuation, both stylistic features already used in many authorship attribution techniques.

Collins' Parser provided significantly better results than the Link Grammar. Before Collins' Parser can be used, each word must be tagged with the part of speech matching it MXPOST Tagger was used for this. The predictors for both the discriminant analysis and the cluster k-means included both the sentence constituents returned by the Collins' Parser and the tags returned by MXPOST.

Discriminant analysis, accurately matched a text with its author 93% of the time. This is significantly higher than the Link Grammar and might be high enough to be used on its own as a new and viable alternative to authorship attribution methods that already exist. Collins' Parser, like the Link Grammar, validated the use of some function words and punctuation marks as predictors of authorship. Collins' Parser, though, also revealed new indicators of authorship. For example, the length of noun phrases and the use of wh- elements such as who, what, where, when, and why were both found to be highly correlated with authorship.

With both the Link Grammar and Collins' Parser, the cluster k-means did not consistently return groups that matched the groups of authors and was not found to be valuable.

This thesis begins by providing background and then explains the experiments run and the conclusions made. Chapter 2 describes what authorship analysis is, why it is important, and some of the current methods used to determine who an author may be. Chapter 3 then gives some background

about parsers and parsing techniques. Chapter 4 is devoted to the Link Grammar. It begins by describing in greater detail how the Link Grammar works and then describes the experiments and results returned by the statistical analyses. Collins' Parser is described in similar detail along with its own experimental results in Chapter 5. Chapter 6 then compares the results returned by the parsers. Finally, Chapter 7 suggests future areas of research. This includes a section on LinGO, a parser considered but not included in this study.

Chapter 2

Authorship Attribution Methods

Authorship attribution methods are based on developing a set of features that can be used to classify a text as written by a specific individual. This depends heavily on the now well-supported theory that authors develop distinct writing features, preferably unconsciously, that can be found and analyzed to match them with other text they have written. Humans are creatures of habit and tend to work within certain comfortable boundaries. Even over years and in a varying situations, humans exhibit the same patterns of behavior, speech, and writing styles.

Modern reasons for identifying the author of texts generally fall into one of two categories: academic research of historical texts and crime prevention and detection. Section 2.1 look at these reasons.

Section 2.2 introduces some of the authorship attribution methods that already exist and are used to find the distinct writing features that can match author with text. The methods included focus on those easily executed with the help of computers and most are based on stylometry, the statistical analysis of literary style.

2.1 Uses for Authorship Attribution Programs

Generally, authorship attribution is used by one of two sectors of society: the academic world and criminal justice.

2.1.1 Determining Historical Sources

Stylometry dates back to 1887 with Mendenhall and his study of word lengths[28]. In 1938, Yule began calculating the average sentence length of authors as well[48], and by 1965 Morton was using both of these ideas to test Greek prose[30].

Since then, the academic community has turned to stylometrics to prove or disprove theories about numerous texts of disputed authorship. These include the *Federalist* papers[31][32][4], Shakespeare[5][29][46], the Junius Letters[17], and both the Old[2] and the New Testaments[33]. Stylometrics was used to demonstrate that a single person wrote the Iliad and that both the Iliad and Odyssey were written by the same person[30]. Most recently, Professor Donald Foster of Vassar College analyzed **Primary Colors**, an anonymously written book about a "fictional" presidential primary campaign, and proposed Joe Klein of Newsweek to be the author; a proposition which was confirmed by handwriting analysis and later by Klein's own admission[37].

Analysis of unconscious style traits is used for more than just identifying authorship. It has also been used to place a chronology on writings by a single author, such as Plato's dialogs[30]. It has been used to determine the gender and native language of people writing e-mails[45]. It has even been used to analyze relationships between authors, for example the Brontë sisters[7].

2.1.2 Crime Detection and Prevention

Crime prevention focuses on using small samples of writing from ransom notes and threats to add valuable information to the profile of the author. Referred to as forensic writer profiling, the information that can be gleaned from a text may include the state of mind, possible motives, and characteristics of the author[38]. For example, it is relatively simple to determine if a hand-written note was written by someone who is left handed. Vocabulary and unique grammatical characteristics may profile the author as being less educated or from a specific part of the country. Other mistakes may reveal that the author is attempting to hide his or her identity by making said mistakes. For example, in the JonBenet Ramsey ransom note, the words *bussiness* and *possessions* are misspelled while *deviation* and *attaché* are correct, even to the point of including the accent mark. This may imply that despite the

misspellings, the author was neither uneducated nor a non-native English speaker.

Handwriting and writing samples may also be compared to samples from suspects. In this case, more traditional authorship attribution methods can be used.

Authorship is also important in the case of theft and plagiarism. Plagiarism is a growing concern for both software writers and authors of literature. With the dramatic increase in access to sources via the Internet, instances of plagiarism are on the rise. Intellectual property rights are becoming harder to defend as the intellectual property is easier to come by. Fortunately, code written by different programmers contains similar style characteristics to text written by different authors. Reliable methods for determining the authorship of both the literature and software can improve the detection and prosecution of cases of plagiarism and theft.

Determining the authors of software can also help catch the authors of malignant code. Reliably identifying authors of viruses, worms, Trojan horses, and crackers can help prosecute such crimes.

2.2 Stylometric Methods

Stylometric analysis is primarily used to establish the authorship of a disputed text. It is the statistical study of language styles and provides a quantitative way of describing the character of a text. It can be used for authorship attribution because of the well accepted theory that authors have unconscious style features that can match them with the texts they write. Stylometrics differs from stylistics, or the study of writing styles, because of this focuses on unconscious stylistic features.

Over one thousand stylometric features have appeared in research[44] including character-based features, word-based features, punctuation-based features, the use of function words, profanity, etc. This section takes a look at some of them. Section 2.2.1 looks at methods which focus on the usage of individual words by the author. Section 2.2.2 describes two methods which look at the context in which words appear. Finally, Section 2.2.3 covers some remaining stylometric methods.

2.2.1 Word Based Methods

Word usage can vary considerably between authors. The use of some words can vary considerably between works by the same author. Some techniques based on the words in a text include word length, misspelled words and their frequency, the use of capitalization, the use of punctuation (for example a single hyphen, "-", versus a longer hyphen, "–", versus a very long hyphen, "—"), the use of contractions, and British versus American spellings. They may also include vocabulary differences and sizes, and the frequency of specific words.

Vocabulary Richness

Studies of the "richness" or "diversity" of an authors vocabulary is one of the fundamental techniques of stylometry. H.S. Sichel's felt that every word in an author's vocabulary had a certain probability of occurrence and the combination of the probabilities for all words provides enough uniqueness between authors to differentiate between them[41]. The frequency of each word in a text is calculated and the Poisson distribution is then found. The Poisson distribution gives the probability of the word appearing x times in a large text.

Other methods relating to vocabulary focus on those words which only appear once (hapax legomena) or twice (hapax dislegomena) within the text. In this case, the frequency of these infrequent words being used in the text is the subject of the analysis.

Function Words

The most popular stylometric technique at this time uses function words. Function words are determinators, prepositions, pronouns, and a variety of other short and frequently used words. See Table 2.1 for examples.

Function words are independent of the idea being conveyed and some believe their use is more subconscious than the usage of content words (longer words used to convey more meaning). As an example, see Table 2.2. This table lists the most common words in the book **Tom Sawyer** by Mark Twain. As can be seen, the only word in the list that is not a function word is *Tom*, and this word is frequent because of the text chosen. Text analysis using function words has been remarkably successful[31][6].

Word Types	Examples
Prepositions	over, under, around, through, of, in, without, between
Pronouns	she, they, I, anybody, it, one
Determiners	a, the, that, my, more, much, either, neither
Conjunctions	and, that, when, while, although, or, nor
Modal verbs	can, must, will, should, ought, need, used
Auxiliary verbs	be, is, am, are, have, got, do
Common Adjectives	big, late, high

Table 2.1: Function Words

Word	Freq.	Word	Freq.	Word	Freq.
the	3332	was	1161	I	783
and	2972	it	1027	his	772
a	1775	in	906	you	686
to	1725	that	877	Tom	679
of	1440	he	877	he	877

Table 2.2: Most Frequent Words in *Tom Sawyer*[26]

Much of the analysis of function words is done using principal component analysis. Principle component analysis obtains a frequency count of the function words and then divides those values by the total number of words to obtain an estimated probability. Each author is identifiable by their unique set of probability values. This technique was used in breaking the pseudonymity of newsgroups[37] to name just one use. By analyzing the use of function words in posts to a newsgroup, Rao and Rohatgi determined which pseudonyms were used by the same author.

CUSUM is an authorship attribution program that claims to be able to attribute authorship to a legal text such as a letter or written confession[1]. It uses statistics about the use of function words in combination with statistics about the average sentence length and the standard deviation from that average.

Anthony Kenny also uses statistics about function words and sentence length. Along with these methods, he adds data about unusual words or phrases used within a text[20].

Using Content Words

Other techniques analyze the use of content words. Methods using content words aim to capture the denotative and connotative meaning of the text. Content words encompass the presentation style, knowledge, and logical organization of the author and may also include the use of stylistic patterns such as rhetorical devices or unique writing techniques.

Content words are words that have concrete meanings and include nouns, adjectives, verbs, adverbs, numbers, and interjections. They add substance to the meaning of the text. Generally, this means that content words do not include function words nor do they include words used only once in the text since these words do not add substantially to the meaning of the text. See Table 2.3 for some examples of content words.

A set of content words for a text can be found using this simple algorithm:
1. List all words in the document.
2. Remove all function words.
3. Remove all words that only appear once.

[1]For more information on A.Q. Morton's methods, including areas where his program has been applied, see [8][3][18].

Word Types	Examples
Nouns	Bob, professor, answer, Sarah
Adjectives	glad, new, large, red
Verbs	search, read, hold, have, drive
Adverbs	really, completely, very, also, enough
Ordinals and Cardinals	three, ten, first, forth
Interjections	huh, ugh, ok, well

Table 2.3: Content Words

This using content words is significantly less popular at this time than using function words, though. Because content words can be very subject specific, it is more difficult to find stylistic features that appear uniformly across all texts written by an author. In general, the content words used in an email are significantly different than the content words used in a formal report, even if the authors are the same. Content words are also assumed to be less unconscious and therefore become less helpful if the author is trying to hide his or her identity.

2.2.2 Context Based Methods

While looking at the words in a text can reveal a lot about the author, looking at the context in which those words appear also reveals stylistic patterns. Methods exploiting authors' use of individual words are described in Section 2.2.1 but authors also use combinations of words differently from each other. This section focuses on techniques that look at the use of combinations of words.

N-Grams

N-grams are sequences of n words and may also be called collocations. N-gram probabilities indicate how likely it is that a collocation, will occur in a text. Some frequently occurring bigrams, or n-grams of size two, are *of the*, *in a*, *to be*, and *has been*. Other common collocations include compounds such as *tea cup*, phrasal verbs like *show off*, and common colloquialism such as *milk and cookies*.

In Section 2.2.1, methods for authorship attribution using individual words were examined. N-grams provide an alternative focus for analysis.

Formally, the probability of an n-gram is equal to the probability that word W_n will occur next given that the preceding words are $W_{n-N}, ..., W_{n-2}, W_{n-1}$ as in equation 2.1. In other words, to calculate an n-gram of size three, or trigram, for the word *hat* in the phrase "*The cat in the hat.*", *hat* is W_n and it is given than the previous words are *in the*. So, the probability of the trigram *in the hat* is equal to the probability that the word *hat* follows the words *in the*.

$$P(W) = P(W_n|W_{n-N}, ..., W_{n-2}, W_{n-1}) \tag{2.1}$$

Calculating the probabilities for n-grams are computationally heavy. As a result, bigrams are the most popular n-grams used, and rarely are studies made using n-grams larger than trigrams. Some techniques involve searching for more than one size of n-grams simultaneously. These techniques try to make use of both the greater detail that can be returned by larger sizes of n-grams and the comparatively faster calculations for n-grams of smaller sizes.

Traditionally, n-grams have been used in speech recognition systems with large vocabulary systems and training texts to provide the program with a likelihood of what the next word might be. They can also be applied to data mining to reduce noise returned from a search. Larger values for n lead to less fault tolerance and smaller values of n lead to more fault tolerance. N-grams can also be used to define what type of information is in a body of documents. In this context, they capture the dependencies between words within a specified context.

Kjell and Frieder cite a number of uses of n-grams in authorship attribution including their own work analyzing the frequency of bigrams in *The Federalist Papers*[22]. Others have since used similar methods on biblical Hebrew texts[43].

Latent Semantic Indexing

Latent semantic indexing (LSI) is traditionally used for text indexing. It tackles two problems: that one word often takes on many different meanings within a single document and that an author will often use different words to describe the same thing. Many synonyms can be used by an author but these synonyms will repeatedly appear in the same contexts. At the same time,

X	2	1	3
Y	1	0	1
Z	0	2	2
	eggs	**bacon**	**coffee**

Table 2.4: Example of a Term-Document Matrix

most words have multiple definitions and may mean vastly different things. The appearance of the same word in very different contexts may imply the use of the same word for a different meaning.

By examining the context that words appear in and the context in which the word never appears, a set of constraints is created defining a "semantic space" for the word. This semantic space takes advantage of the semantic relationship between words and the text by containing information on word-word, word-passage, and passage-passage relationships[25]. Using these contexts, the subtle differences in semantics of a single word can be ascertained creating a "contextual definition" that often correlates well with the actual definition. Other words that appear in similar contexts as the initial word can then reference each other as having similar meaning.

LSI begins by creating a list of content words across all documents using the technique described in Section 2.2.1. Then a term-document matrix is created by charting the frequency of each content word in each document. For example, given documents X, Y, and Z and content words eggs, bacon, and coffee, the matrix shown in Table 2.4 may be created.

Following the creation of the term-document matrix, the matrix is decomposed using singular value decomposition (SVD). SVD first calculates the term space of the documents. Continuing to use our breakfast example, the three terms each get an axis resulting in a three dimensional graph. Then a vector is plotted for each document starting at the origin and ending at the position specifying the frequency of each of the three terms. Figure 2.1 shows a possible term space for our breakfast example if many more documents were examined. In Figure 2.1, vectors are represented as points in an effort to keep the image readable. The vectors would connect the origin to each point. When more than three terms are used, the graph branches out into more dimensions. SVD then projects the results of the term space into a smaller space with less dimensions thereby blurring the lines between words with similar contextual meaning. The choice of number of resulting

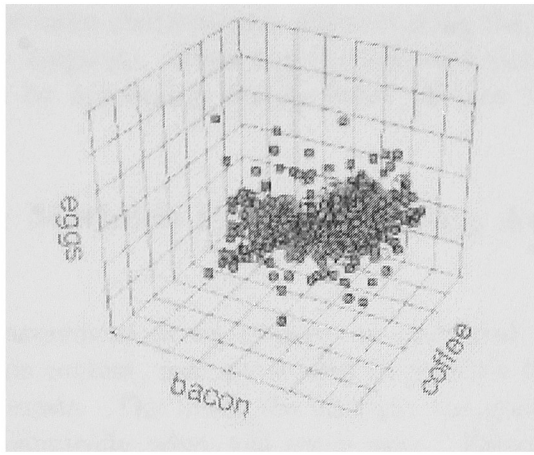

Figure 2.1: Example of Term Space[47]

dimensions can produce different and interesting results as can be seen in the work by Soboroff et al.[43].

New vectors for each document have now been defined by SVD. The size and direction of these vectors describe the document's relationship to the content words. Singular value decomposition projects multi-dimensional graphs into graphs using less dimensions while still preserving as much information as possible about the relative distances between the document vectors. Some information is lost in this process, but much of that information may be noise. The result is that similar words are superimposed on each other while different words maintain their distance.

LSI is used frequently in search engines and data mining and has been proven to improve information retrieval[34]. In this context, documents with similar words are considered to have similar semantic meanings and although the technique does not compute the actual meaning of the document, the results correlate well with how humans classify documents[47]. For example, if clock, time, and date appear in the same document frequently enough, LSI classifies them as semantically close and searches for clock and time would also return documents containing the word date [47].

Soboroff et al. use the LSI of n-grams to visualize the relationship between document authorship. The text used was a combination of both Aramaic and biblical Hebrew texts attributed to different authors. Their results found

that the most prominent characteristic differentiating the texts were writing style, followed by language. Their conclusions state that their techniques group documents by authorship despite other obvious characteristics like language[43].

2.2.3 Other Methods for Determining Authorship

Entropy

Entropy is a measurement of uncertainty as computed using methods of probability. In this context, entropy is used to measure the amount of flux or disorder that exists. The lower the entropy, the greater the ability to determine probabilistically what will come next. Entropy among similar documents by the same author is low, whereas entropy between different documents in different writing styles is much higher.

Zipping algorithms can be used to estimate entropy. The greater the entropy, the less compression is achievable and vice versa. Zipping algorithms depend on being able to reconstruct data with low entropy easily using predefined instructions. For example, given the input AAAAA, the algorithm could learn a more concise representation of this, e.g., repeat the letter A five times or Ax5. Inputs such as QMSEW are much more difficult to encode concisely unless they occur frequently in which case the algorithm would shorten it.

Although zipping files only provides an estimated level of entropy, it has already been shown that zipping programs can be used to determine language trees, writing styles, and even authorship. The repeated patters of an author are exploited by the zipping algorithm in order to compress the document further. Therefore, documents written by the same author will compress more (revealing less entropy) than documents written by different authors[24].

Handwriting Analysis

A person's handwriting has been conjectured to be as unique as their writing style. For example, various styles of writing the single letter W are quite distinctive and can point to one author over another. For example, in Figure 2.2 the second and third set of Ws are from samples of text taken from two authors. The first is from a document which was written by one of the two

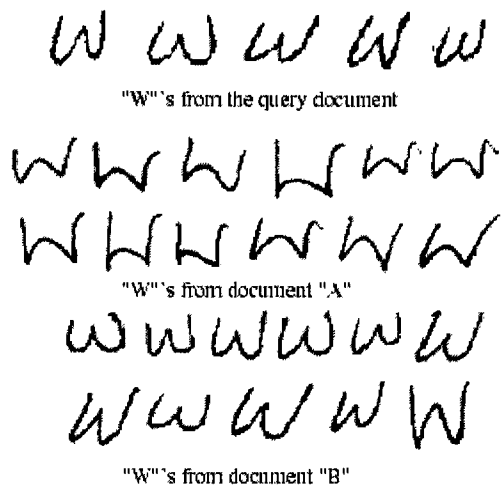

Figure 2.2: Examples of the Letter *W* From Three Documents[10]

authors. Even an untrained eye can quickly determine which author also penned the *W*s in the first line.

Various techniques in computer vision have been applied to handwriting analysis programs, including pattern recognition, machine learning, and feature selection. Here, as in writing style analysis, the distance between two texts is calculated and used to classify them as either written by the same individual or by different individuals[10].

Markov Chains

Markov chains are similar to n-grams except that they examine text by looking at sequences of letters rather than collocations. They predict future letters given previous letters. For example, for an initial letter T, H would have a higher probability of being next than Q. If the letter just read in was a Y, the following character would probably be a space.

Generally, Markov chains are visualized as nodes of discrete states (in this case, letters) with transitions between them. Each transition has a probability of being followed. Figure 2.3 shows how the letter T might be related to itself, H, and E. Obviously, a full Markov chain would include

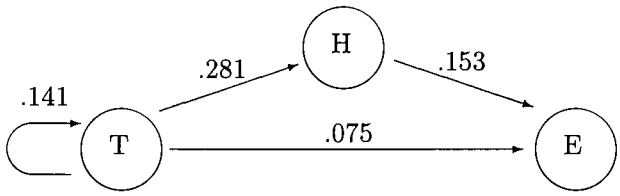

Figure 2.3: An Example of a Markov Chain

relationships between ever letter in the alphabet.

Khmelev and Tweedie used Markov chains of bigrams of letters to determine authorship. Unfortunately, their study returned disappointing results. This may be in part because bigrams of letters may be a linguistic unit too small to contain meaningful information about the author. Future work with Markov chains using larger n-gram units may lead to more satisfactory conclusions[21].

Chapter 3

An Introduction to Parsers

The two programs used in this thesis are natural language parsing programs. These programs are used to capture the syntax used by an author. As explained in Chapter 2, current authorship attribution methods look for specific words or small collocations. This thesis looks at the structure of the sentences to see if authors employee different syntactic structures regularly enough to differentiate them.

Parsing programs are used to help computers understand natural languages, or languages used by humans such as English, Russian, and Japanese. The process of making computers understand these natural languages has developed into a research area called natural language processing. As might be imagined, natural language processing is a very large task and draws on the research and expertise of a variety of other fields.

Natural Language Processing (NLP) focuses on enabling interaction between a human and a computer in the human's native language. Broadly speaking, NLP systems are used in three areas: human-computer interfaces using natural language, improving data mining and information extraction, and automated translation programs between two natural languages.

Natural language interpretation systems generally contain some kind of parser, a semantic interpreter, and a contextual interpreter. These three components interact with each other as in Figure 3.1, although the components may not as distinct as the figure implies and their actual execution might overlap. Parsing is the stage used in this study. A parser is a program that uses a lexicon, and syntactic rules to transform a sentence or series of sentences into one or more reasonable syntactic structures.

This chapter provides background on parsing techniques. Section 3.1 dis-

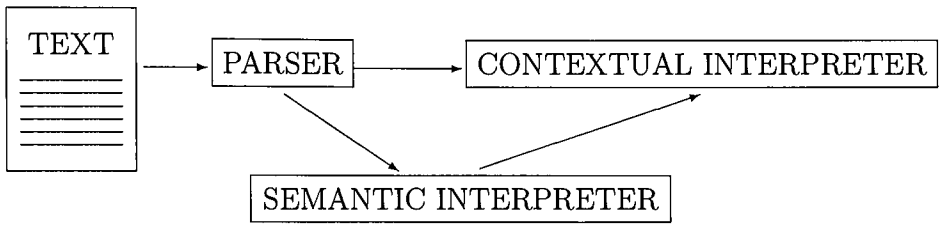

Figure 3.1: Natural Language Interpretation Flow Chart

cusses the biggest challenge to parsing natural languages: ambiguity. Section 3.2 then describes what the output of good parsing can tell us about a text. Finally, Section 3.6 describes some of the techniques used to parse natural languages. Most of Section 3.6 focuses on techniques not used by the two programs in this study since their techniques will be covered in more detail in Chapter 4 and Chapter 5 where the Link Grammar and Collins' Parser are discussed.

3.1 Ambiguity

Ambiguity is the single most important problem in NLP Formal languages are specifically designed to eliminate ambiguity, but natural languages are riddled with it at every level. We rarely recognize the potential ambiguities since we automatically integrate our knowledge of the situation and world into everything we see and hear. For humans, ambiguities generally only occur in the form of puns and misunderstandings. For computers, every potential interpretation of a sentence must be systematically analyzed to find the one that is most reasonable.

One of the problems related to ambiguity is also known as the representation problem. Words often have many meanings. For example, *cook* may refer to a person or the act of cooking. People automatically read a sentence with the correct sense of the word, but NLP programs must eliminate each meaning one by one until the correct meaning is established.

Another form of ambiguity is the semantic ambiguity that comes from a syntactically valid sentence. Consider the sentence *Salespeople sold the dog biscuits.* It is uncertain whether the salespeople are selling biscuits to the dog or selling dog biscuits. A perfectly valid parse may even state that the

salespeople are selling a dog named biscuits[11] even though most humans would not consider that as a reasonable interpretation of the sentence.

The previous example is a globally ambiguous sentence. Another semantic ambiguity is local ambiguity. For example, *The company that bought Takeover Inc. sold toothpaste.* Here the meaning of the sentence is clear when taken as a whole, but if just the last three words are considered, the meaning is misinterpreted.

Local ambiguities are not a problem for NLP systems but global ambiguities and the representation problem must be dealt with. Adding levels of knowledge helps deal with both. Parsers add syntactic knowledge to the system. Syntactic knowledge contains the grammatical rules of the language. It defines how words can be put together and how they relate to one another to form grammatically correct sentences. It is syntactic knowledge that defines the functional structure of a sentence including prepositional phrases, verb phrases, verb tenses, subjects, and objects. It is this level of knowledge that this thesis uses to determine authorship of texts.

3.2 What Parsing Reveals

Syntax describes the rules by which meaning is given to a sequence of words. For any idea, numerous syntactically correct sentences can be written to describe that idea. Each time an author begins to write, he or she decides which syntactic rules to follow and how to organize his or her sentences. This thesis conjectures that syntactic patterns develop in an authors writings and that these patterns are an unconscious style feature that may be used to determine authorship.

One of the first things syntax reveals is the role of the words in the sentence. For example, in the sentence "*Scott hit Joe.*" Scott is recognized as the subject, hit the verb, and Joe the object being acted on. The meaning of the sentence is very different if the subject and object are reversed as in "*Joe hit Scott.*" Authorship attribution techniques using function words are already quite common (see Section 2.2.1). By using parsers, this thesis attempts to find other word usage patterns that can help with authorship attribution.

Among other things, syntax reveals the relationship between words. For example, the sentence "*Visiting aunts are boring.*" means something quite different from the sentence "*Visiting aunts is boring.*" In the first sentence,

the verb *are* reveals that *aunts* is the subject and *visiting* merely describes what kind of aunts are boring. In the second sentence, *visiting* is the action, *aunts* states who is being visited and the rest of the statement explains the speakers opinion about the act of visiting aunts. Knowing which part of speech each word belongs to may be important since one author may use long noun phrases with lots of descriptors while another author may consistently use a variety of verbs. With parsers, the use of writing style features such as these can be measured.

3.3 Some Definitions

Relationships between words are defined by sentence constituents. Constituents are linguistic units that is part of a larger construction. For example, noun phrases and verb phrases are constituents of sentence. Although constituents can include any linguistic unit down to the level of individual words, generally the term is used for units smaller than sentences and larger than individual words.

Every constituent has a single head or headword. The head determines the characteristics of the rest of the constituent. All other words or constituents within the one constituent are dependent on the head word. The head word may contain information about singularity or plurality, tense, and meaning. For example, in the noun phrase *The dark night*, *night* is the head. *Bought* is the head in the verb phrase *might be bought*. In the prepositional phrase *in the box*, the head is the preposition *on*.

Both of the parsers used in this thesis are lexicalized. Parsing with a lexicalized grammar means that sentences are parsed by applying grammar rules to the lexicon of words.

3.4 Chomsky Hierarchy

Noam Chomsky defined four classes of language grammars: Regular Grammars, Context-Free Grammars, Context-Sensitive Grammars, and Unrestricted Grammars. Each language class type builds on the previous language. In other words, any language that can be recognized by a Regular Grammar can also be recognized by the other three. Context-Free Languages can be recognized by Context-Sensitive and Unrestricted Grammars. Finally,

Grammar Structure	Shorthand	Examples
sentence	S	He threw a red ball.
noun phrase	NP	a red ball, he,
verb phrase	VP	run, is going
determinant	DET	a, the, that
noun	NOUN	ball, he, Brazil
verb	VERB	run, be, think

Table 3.1: Shorthand for Grammar Structures

Context-Sensitive Languages are recognized by Unrestricted Grammars.

Terminals, non-terminals, and rules are used to define each of the grammars. In addition to these three elements, a beginning point in the grammar is also defined.

Terminals are symbols that cannot be broken down further and non-terminals are symbols that can be broken into smaller parts. For example, a noun phrase can be a noun, or a determinant followed by a noun. A noun can be "ball," "he," or "Brazil." Both "noun phrase" and "noun" are non-terminals, but "Jill" is a terminal because there is no further way to break it down into a smaller part. In most parsers, words are terminals and parts of speech are non-terminals.

Rules generally have the form $X_1, X_2, ..., X_n$ consists of $Y_1, Y_2, ..., Y_m$, or,

$$symbol_{x_1} symbol_{x_2} ... symbol_{x_n} \rightarrow symbol_{y_1} symbol_{y_2} ... symbol_{y_m} \quad (3.1)$$

although the values that can be on the left and to the right must follow the formulation of the grammar.

For the sake of simplicity, some shorthand will be used in this section. Reference Table 3.1 for a description of the shorthand.

3.4.1 Regular Grammars

Regular Grammars are a very restricted grammar. The rules of the grammar state which non-terminals can turn into which terminals, but only one non-terminal can be on the left of the rule. The right side can either be a single

NOUN → ball	VERB → run
NOUN → the NOUN	VERB → ate
NOUN → Brazil	VERB → am
NOUN → cook	VERB → cook

Figure 3.2: An Example of a Regular Grammar

terminal or a single terminal followed by a non-terminal. Figure 3.2 shows a simple Regular Grammar.

These languages are typically used in defining patterns and the lexical structure of formal languages but do not provide enough flexibility to define natural languages well.

3.4.2 Context-Free Grammars

Context-Free Grammars (CFGs) are a powerful class of grammars because they are general enough to be able to specify rules for most natural languages and constrained enough for there to be efficient algorithms to work with them. Collins' Parser uses techniques directly related to CFGs. More details about Collins' implementation can be seen in Chapter 5. Rules for CFGs must contain a single non-terminal on the left but can have any combination of terminals and non-terminals on the right.

Figure 3.3 shows examples of some CFG syntactic rules. The first rule states that a sentence is made of a noun phrase followed by a verb phrase. Following rules then describe what make up both a noun phrase and a verb phrase. The rules from Figure 3.2 can be added to the rules in Figure 3.3 because Context-Free Languages include all Regular Languages. Now sentences such as "*He ran.*" and "*The cook ate.*" can be accepted by our grammar.

More extensive CFGs would include many more phrase structures and probably subdivide large categories like VERB into sub-categories where information like the number of objects to be expected could be included in the encoding. This provides even more levels of complexity.

3.4.3 Context-Sensitive Grammars

Although CFGs are quite powerful, natural languages are generally accepted as being sensitive to context. In other words, the meaning of the word "cook"

S → NP VP	NP → DET NOUN
VP → VERB	NP → NOUN
VP → VERB NP	NP → DET NOUN NOUN
VP → VP NP NP	NP → NP NP

Figure 3.3: An Example of a Context-Free Grammar

depends on the context in which it is used. This context cannot be expressed with CFGs.

The rules for Context-Sensitive Grammars (CSGs) allow for even greater flexibility in defining a language. Both sides of the rule can be any combination of terminals and non-terminals. The only restriction is that the number of symbols appearing on the right must equal or exceed the number of symbols appearing on the left. This gives "context" to when a terminal can exist. The context gives the grammar the ability to decide, given the context, whether a verb phrase is followed by one or more noun phrases or a noun is singular or plural.

Practically all programming languages and some debate that natural languages fall into this category. The added flexibility of the CSG also makes using it more complicated. Searching through a large number of CSG rules to find which ones can be applied to any given sentence becomes very hard. As more rules are added to the grammar, the amount of time needed to find a solution increases exponentially, resulting in parsing programs that take too long to be practical.

3.4.4 Unrestricted Grammars

The rules for Unrestricted Grammars (sometimes called Recursively Enumerable or Phrase-Structure Grammars) are defined by any combination of terminals on the left and any combination of terminals and non-terminals on the right. This provides the most flexibility in describing a language, but also makes accepting a sentence as being in the language defined by the grammar a very difficult task to complete.

3.5 General Approaches to Parsing

Now that CFGs are understood, some more general parsing approaches can be described. The examples will all use the CFG from Figure 3.3 but these approaches can be used with any of the techniques described in this chapter. This section explains the approaches and describes why some approaches make more sense to use with some techniques or languages.

3.5.1 Top-Down Versus Bottom-Up

Top-down parsers start with the top level, or most general form defined in the grammar rules and slowly resolve each rule until they reach the words of the sentence. For example, a top-down parse of the sentence "*The can will rust*" using the CFG from Figure 3.3 would look something like this:

S → NP VP
→ DET NOUN VP
→ The NOUN VP
→ The can VP
→ The can MV VERB
→ The can will VERB
→ The can will rust

Just as the top-down parsers start with the top level and work their way down to the literal sentence, bottom-up parsers start with the sentence and resolve each segment of the sentence by following the rules of the grammar. To continue with our example, the following is a bottom-up parse of "*The can will rust*" using the same grammar rules.

The can will rust → DET can will rust
→ DET NOUN will rust
→ DET NOUN MV rust
→ DET NOUN MV VERB
→ NP MV VERB
→ NP VP
→ SEN

3.5.2 Depth-First Versus Breadth-First

Depth-first search and breadth-first search are strategies for searching though a large grammar to come up with a reasonable parse.

Depth-first starts by always using the first rule that it can until either it accepts the rule as part of the solution or it finds that the rule cannot be part of any reasonable parse. In the latter case, it backtracks through the rules until an alternative can be found. This method is easier to program and takes less memory than a breadth-first search. It is best if there are not many rules to choose between, but a lot of rules may need to be applied to find the parse.

Breadth-first follows all possible grammar rules at once. By working its way step-by-step through all possibilities simultaneously, the order in which the rules appear in the grammar becomes unimportant and this method would generally be better if the probabilities of the rules were not known. This method is also faster if their are a large number of rules that could be applied at any given time.

3.5.3 Left-Right Versus Right-Left

Just as the names might imply, the left-right approach begins at the left end of the sentence working its way to the right until the end of the sentence is found. Right-left parsers work from the right to the left. Generally, for English, the left-right approach works best. With speech processing systems, the input is in left to right order. The left-right approach also more closely follows human processing since we too read from left to right, processing the information as we go.

3.6 Parsing Techniques

A well-defined grammar for natural languages is the groundwork on which programs such as parsers are based. The parsers take the grammar along with a text and formalize syntactic structures containing information about each word and how it relates to the other words in the sentence. Parsers are directly heavily dependent on the grammars used to describe the language. This section gives a basic overview of some grammars and parsing techniques.

Grammars are formal definitions for the structure of a language. A good grammar must incorporate three things: it must generalize well and be exten-

S → NP VP	(1.0)	NP → DET NOUN	(0.5)
VP → VERB NP	(0.8)	NP → NOUN	(0.3)
VP → VP NP NP	(0.2)	NP → DET NOUN NOUN	(0.15)
		NP → NP NP	(0.05)

Figure 3.4: An Example of a Probabilistic Context-Free Grammar[11]

sible, thereby increasing the number of correct sentences it accepts; it must be selective thereby increasing the number of incorrectly formed sentences are rejected; and ideally, the grammar should be simple and understandable. The formal definition for the grammars often take the form of rules. Examples of rules in a grammar might include the following:

- A sentence contains a noun phrase followed by a verb phrase.
- A noun phrase can consist of a determinant and a noun or a noun.
- A verb phrase can consist of one or more verbs, or a verb followed by one or more noun phrases.

3.6.1 Probabilistic Context-Free Grammars

One of the difficulties with CFGs is deciding which rule to use when more than one can be applied. Probabilistic context-free grammars (PCFGs) add a probability assignment to each rule. By following the most probable rule first, it should take less time to find a correct parse for the sentence. Figure 3.4 is the same CFG from Figure 3.3 with the addition of number values in parentheses after each rule. These numbers indicate the probability with which that rule is used in standard English. Note that the sum of the probabilities for each part of speech equals one.

The probability that a series of rules gives the correct parse for a sentence is calculated by multiplying the probabilities of each rule used. The most probable parse is found by maximizing this product. Equation 3.2 restates this. Here, r_c is the probability of rule r being used to expand the sentence subconstituent c. The probability that sentence s follows the rules $r_1 r_2 ... r_n$ is calculated from the product of r_c for each sentence subconstituents, c, occurring in s

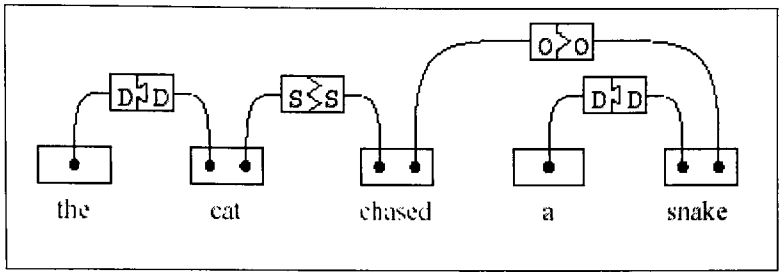

Figure 3.5: An Example of a Link Parse

$$p(s, [r_1 r_2 ... r_n]) = \prod_c p(r_c) \tag{3.2}$$

Depth first search works well with PCFG since the grammatical rules can be ordered in such a way as to always try to use the most probable rules first. Collins' Parser (See Chapter 5) is a PCFG that matches a lexicon with grammar rules.

3.6.2 Link Grammars

Link grammars are a different context-free way to parse sentences[42]. They are based on direct connections between words. The connections, or links, can be thought of as two electrical cords. One side comes from one word while the other side comes from the other. Each cord can only hook to one other and the cord it connects with must be of the same type and the opposite side. Figure 4.1 shows an example of a link parse. As can be seen, each link has a unique shape or letter assigned to its connection. Each link is also hard coded to be either the left side or the right side of the connection and is given a length relative to other links.

There are four rules that links must follow: planarity, connectivity, exclusion, and ordering. Planarity simply states that no link can cross another. Connectivity is satisfied as long as every word in the sentence is connected, whether directly or indirectly. Exclusion states that two words cannot be connected to each other twice by different links. Ordering states that if a word has two links extending off the same side, then the link that is shorter must connect to a word that is closer. For example, in Figure 4.1 the word *snake* has two links that extend to the left. The **D** link is shorter than the

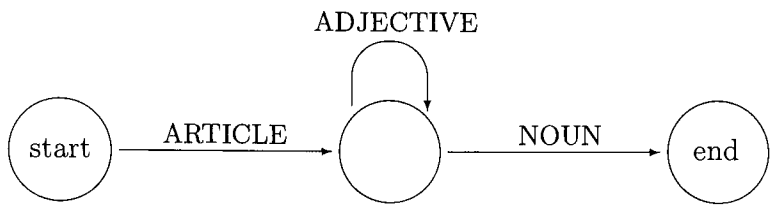

Figure 3.6: Transition Network for a Noun Phrase

O link so the word connected to *snake* via the **D** link must be closer in the sentence to the word *snake* than the word linked to *snake* via the **O** link.

There is no notion of constituents such as prepositional phrases or phrasal verbs nor are there specifically defined syntactic functions like subject and object. Rather, relationships between words are represented as word pairs linked together. This means that words directly associated with each other, such as subjects and verbs, are directly linked making it easy to compare agreement and gather statistical information about word relationships. A more detailed discussion of link grammars can be found in Chapter 4.

3.6.3 Transition Networks

Transition networks are based on representations of parts of speech as arcs connecting nodes. Each arc represents a discrete bit of knowledge. As new experiences and information connect more and more nodes, a network is built[19]. Each network uses the arcs to define rules for a larger part of speech. A sentence constituent can be tested to see if it is a specific part of speech by following the arcs in the associated transition network. The constituents must begin at the start node and end at the end node.

For example, look at Figure 3.6. This transition network defines a noun phrase. The arcs are "ARTICLE," "ADJECTIVE," and "NOUN." A potential noun phrase is tested by seeing if the words follow the parts of speech on the arcs. For example, *a big red ball* starts at the start node and follows the article arc with the word *a*. The words *big* and *red* follow the adjective arc back to the same node. Finally, the word *ball* follows the noun arc to the final node.

These transition networks, though, are not powerful enough to describe all the languages describable by CFGs. Recursive transition networks add to normal transition networks by allowing arcs to refer to other transition

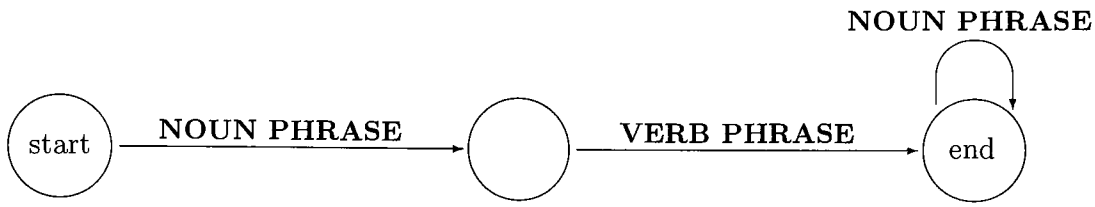

Figure 3.7: Recursive Transition Network Example

Arc Type	Example	Explanation
category	NOUN	the current word must be of the named category
word	of	the current word must be identical to the one named
push	**NP**	the named network must be successfully traversed
jump	jump	can always be traversed

Table 3.2: Arc Labels for RTNs[1]

networks. Figure 3.7 is an example of a recursive transition network. All three of the arcs in Figure 3.7 are actually references to other transition networks. For example, the "**NOUN PHRASE**" arcs are satisfied only if the sentence begins and ends with phrases that can be described by Figure 3.6. Larger recursive transition networks can are recursive because they can refer to themselves or to other networks that may eventually return to themselves. This recursive ability allows recursive transition networks to define full grammars rather than just subconstituents of words. Table 3.2 shows the types of arcs possible in a recursive transition network.

Chapter 4

The Link Grammar

Link grammars are based on grammatical systems in which a sentence is accepted by the grammar if legal links can be connected between all of the words. They were first discussed in Section 3.6.2. Connected links can be visualized as arcs connecting pairs of related words as in Figure 4.1. Each arc is created by making a connection between the link from the initial word and the link of the terminal word.

Links must satisfy the requirements of the natural language being defined. The application of links also must follow specific rules, for example, no link can cross, and the links must form a connected graph. Link grammars are first introduced in Section 4. The remainder of this chapter will give more detail about link grammars in general and focus on the Link Grammar developed by Daniel Sleator and Davy Temperley[42].

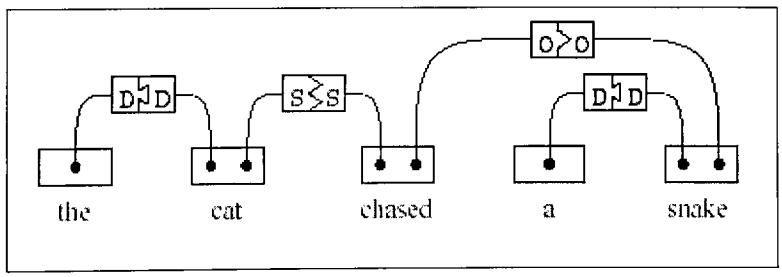

Figure 4.1: An Example of a Link Parse

4.1 Theory

A link grammar is based on a dictionary containing all the words in the language and the link types that extend off of that word. The link types must include details about whether the link is a left link that must be connected to a word on the right or a right link that must be connected to a word on the left. Some words may have have multiple links but these links must follow certain rules. The dictionary describes which links can extend off of the word at the same time, and which links are "longer" than other links. If a word has multiple links extending in the same direction from it, the link that is "longer" must connect to a word that appears further away from than the other link.

Linking requirements are defined individually for each word. This makes the system lexical. Irregular verbs and colloquialisms are easier to express in the link grammar because each word has its own definition in the dictionary. Adding and making updates to the dictionary are also simplified by this technique.

Link grammars are related to categorical grammars or phrase structure grammars. Sleator and Temperley show how to represent a categorical grammar in a link grammar form, although the reverse is stated to be more difficult. The two grammar types have other traits in common as well. Both are context free[42] making them good candidates for parsing natural languages. Both are also good at succinctly expressing a structure for a natural language that can be understood by both human users and computer programs. Since Collins' Parser is a categorical grammar, this relationship is of interest.

There are four rules that all links must follow: planarity, connectivity, ordering, and exclusion.

Planarity simply states that no link can cross another. When all the arcs for the links are drawn over a sentence, no arc can pass through the arc of another link. If any arc does cross another then the rule of planarity is not satisfied.

Connectivity is satisfied as long as every word in the sentence is connected to every other word in the sentence. The connection can be either direct or indirect, but the arcs and words must make a connected graph.

Ordering states that if a single word has two links that extend off the same side, the links that are shorter must link words closer to each other than links that are longer. For example, in Figure 4.1 the word snake has two links that extend to the left. The dictionary states that the **D** link is

shorter than the **O** link. Therefore, the word connected to *snake* via the **D** link must appear physically closer in the sentence to the word *snake* than the word linked to *snake* via the **O** link.

Exclusion states that no two words can be connected more than once by any set of links. Each pair of words can have, at most, only one link directly connecting them.

There is no notion of constituents such as prepositional phrases or verb phrases, nor are there any specifically defined syntactic functions like subject and object. Rather, relationships between words are represented as word pairs linked together. As the lexicon is developed, constituents emerge as words are linked together and with the rest of the sentence, but these relationships between links and constituents are not what the link grammar sets out to do.

One of the biggest challenge for link grammars is conjunctions. The dictionary states very specifically which links can extend off of a word at the same time. By introducing conjunctions into a sentence, a link that otherwise would not be allowed to appear more than once may need to connect more than two words. For example, if the sentence "*Tom, Scott, and Joe study together.* needs three links connecting the verb *study* to the nouns before it, *Tom*, *Scott*, and *Joe*. The type of link that connects a verb to a preceding noun would be an **S** link, but the dictionary generally states that this link can only link a verb to one noun. The creator of a link grammar must find some way of working around this difficulty.

4.2 Program

The goal of the link grammar, as with any grammar, is to "distinguish, as accurately as possible, syntactically correct English sentences from incorrect ones.[42]" This goal is achieved with a dictionary of approximately eight hundred definitions for twenty-five thousand words and a program which takes as input a sentence and, using the dictionary, returns the links satisfying the link grammar rules for that sentence. This section describes the Link Grammar developed by Sleator and Temperley[1].

The output graphically shows the links, indicates the time it took, and outputs a bit of other information about the sentence. For example, if the

[1]Directions for how to install and run the Link Grammar can be found at `http://www.link.cs.cmu.edu/link/`.

```
++++Time                                          0.04 seconds (0.04 total)
Found 2 linkages (2 had no P.P. violations)
  Linkage 1, cost vector = (UNUSED=0 DIS=0 AND=0 LEN=18)

    +------------------------------Xp-------------------------------+
    +-------------Wd-------------+                                  |
    |      +---------Ds---------+              +-------Js------+   |
    |      |     +-------A------+              |   +-----Ds----+   |
    |      |     |       +---A--+---Ss--+--MVp-+   |     +--A--+   |
    |      |     |       |      |       |      |   |     |     |   |
LEFT-WALL the quick.a brown.a fox.n jumped.v over the lazy.a dog.n .

Press RETURN for the next linkage.
linkparser>
```

Figure 4.2: An Example of Link Grammar Output

input was:

```
The quick brown fox jumped over the lazy dog.
```

The output would look like Figure 4.2. Above the sentence are the links, represented using straight lines. The types of links are written in capital letters while the lower case letters indicates sub-categories of the links. For example, a link of type **A** connects "brown" to "fox." Another link of type **A** links "quick" to "fox." A "LEFT-WALL" exists at the start of every sentence and either a "RIGHT-WALL" or a punctuation mark ends every sentence. This provides a way of connecting the front of the sentence to the back of the sentence and assuring that all links occur within them.

The Link Grammar indicate the role each word plays in the sentence via subscripts. For example, *The cat ran.* would be represented as *The cat.n ran.v* where .n indicates a noun. and .v indicates a verb. Unknown words are indicated as such with a `[?]` before the subscript and linked using a part of speech that fits best in the sentence.

As stated in Section 4.1, one of the biggest challenge for link grammars is conjunctions that are not in the dictionary definition. The Link Grammar deals with conjunctions by forming a list of the words joined by a conjunc-

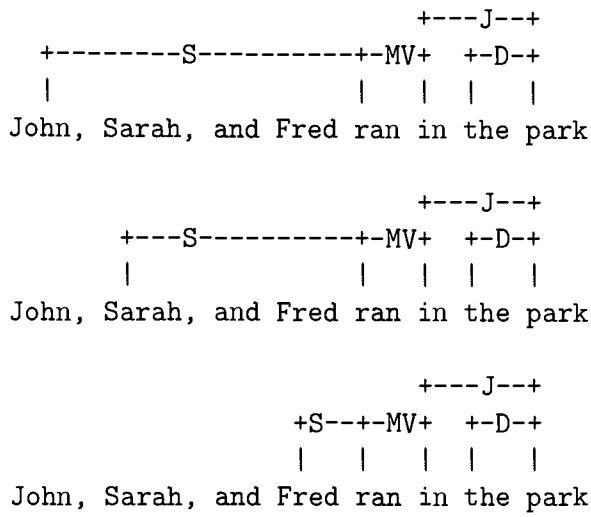

Figure 4.3: Link Grammar Output for a Sentence With Conjunctions

tion. Each word in that list is then resolved with the rest of the sentence individually. This can be seen in Figure 4.3.

4.3 Experiments and Results

Experiments were run on thirty-two texts by eight literary authors of the twentieth century. Brief biographies of the authors and the books included by each can be found in Appendix A. The texts were collected from internet via sites offering free access to classical books. Most texts were collected from the Project Gutenberg site. Texts were all preprocessed to remove the headers and footers added in by the internet sites, chapter headers, title and author information, footnotes, and other included information that was not written by the author or did not form full sentences. Each text was then run through a program which formatted the text so each sentence appeared on its own line.

Once all this preprocessing was completed, the texts were ready to be parsed using the Link Grammar. Each text was run through the parser using a batch process and the output was saved to a file. That file now contained all of the data about the link types for each sentence in the text.

Calculations on each text were run to find the average number of links per sentence for each of the one hundred seven link types included in the Link Grammar.

The data about all link types in all texts were compiled into a table. Link types whose average usage was in the tenths of a percent were eliminated. Reasonable authorship attribution should work on small text samples and link types averaging less than one percent of the sentences were deemed too infrequent for useful attribution. This left forty-three link types. Further examination revealed that the frequency of use for sixteen of the link types was not correlated with the authors. These link types were then removed from consideration. The remaining twenty-seven link types and a brief description of what they connect can be found in Table 4.1.

Link	Description
A	connects a pre-noun "attributive" adjective to the following nouns: "The *big dog* chased me."
AL	connects a few determiners like "all" or "both" to following determiners: "*All the* people are here."
AN	connects noun-modifiers to following nouns: "The *tax proposal* was rejected."
B	serves various functions involving relative clauses and questions. It connects transitive verbs back to their objects in relative clauses, questions, and indirect questions: "The *dog* we *chased*," "*who* did you *see*?"; it also connects the main noun to the finite verb in subject-type relative clauses: "The *dog* who *chased* me was black."
C	links conjunctions to subjects of subordinate clauses: "He left *when he* saw me." It also links certain verbs to subjects of embedded clauses: "He *said he* was sorry."
CO	connects "openers" to subjects of clauses: "*apparently/on* Tuesday, *they* went to a movie."
D	connects determiners to nouns: "*the dog* chased *a cat* and *some birds*."
DG	connects the word "The" with proper nouns: "*the Riviera*," "*the Mississippi*."
E	is used for verb-modifying adverbs which precede the verb: "He is *apparently leaving*."

continued on next page

continued from previous page

Link	Description
EB	connects adverbs to forms of "be" before an object or prepositional phrase: "He *is apparently* a good programmer."
J	connects prepositions to their objects: "The man *with* the *hat* is here."
K	connects certain verbs with particles like "in." "out." "up" and the like: "He *stood up* and *walked out.*"
L	connects certain determiners to superlative adjectives: "He has *the biggest* room."
M	connects nouns to various kinds of post-noun modifiers e.g. prepositional phrases: "The *man with* the hat." participle modifiers: "The *woman carrying* the box." prepositional relatives: "The *man to* whom I was speaking." and other kinds.
MV	connects verbs and adjectives to modifying phrases that follow, like adverbs: "The dog *ran quickly.*" prepositional phrases: "The dog *ran in* the yard." subordinating conjunctions: "He *left when* he saw me." comparatives, participle phrases with commas, and other things.
O	connects transitive verbs to their objects, direct or indirect: "She *saw me.*" "I *gave him* the *book.*"
OF	connects certain verbs and adjectives to the word "of." "She *accused* him *of* the crime." "I'm *proud of* you."
P	connects forms of the verb "be" to various words that can be its complements: prepositions, adjectives, and passive and progressive participles: "He *was* [*angry*/*in* the yard/*chosen*/*running*]."
PP	connects forms of "have" with past participles: "He *has gone.*"
R	connects nouns to relative clauses. In subject-type relatives, it connects to the relative pronoun: "The *dog who* chased me was black." In object-type relatives, it connects either to the relative pronoun or to the subject of the relative clause "The *dog that* we chased was black." "The *dog we* chased was black."
RW	connects the right-wall to the left-wall in cases where the right-wall is not needed for punctuation purposes.
S	connects subject nouns to finite verbs: "The *dog chased* the cat." "The *dog* [*is* chasing/*has* chased/*will* chase] the cat."

continued on next page

continued from previous page	
Link	**Description**
TH	connects words that take "that [clause]" complements with the word "that." These include verbs "She *told* him *that....*" nouns "The *idea that....*" and adjectives "We are *certain that.*"
TO	connects verbs and adjectives which take infinitival complements to the word "to." "We *tried to* start the car." "We are *eager to* do it."
W	connects the subjects of main clauses to the wall, in ordinary declaratives, imperatives, and most questions except yes-no questions. It also connects coordinating conjunctions to following clauses: "We left *but she* stayed."
X	is used with punctuation, to connect punctuation symbols either to words or to each other. For example, in this case, *poodle* connects to commas on either side: "The dog, a *poodle*, was black."
YS	connects nouns to the possessive suffix "'s." "*john 's* dog is black."

Table 4.1: Summary of Link Types for Twenty-Seven Relevant Links[42]

Two forms of multivariate analysis were used to analyze this output: discriminant analysis and cluster K-means.

4.3.1 Discriminant Analysis

An Overview of Discriminant Analysis

Discriminant analysis is used to classify items into two or more groups. In this case, the items are the texts and the groups are the eight authors. Performing a discriminant analysis is a two step process. The first step, discriminant predictive analysis, is to calculate the linear discriminant function. The second step, discriminant classification analysis, is to classify items with the discriminant function into one of the groups.

For discriminant predictive analysis, a sample set of items and the group to which each belongs must be known. The sample set for this analysis were the thirty-two texts. This step also needs a set of p predictors. The linear discriminant function describes the groups according to these predictors. Predictors are used to correlate each item with its group. The predictors used included a variety of combinations of the link types mentioned in Table 4.1. The discriminant function is modeled after Equation 4.1 where t is the

number of groups, D_t is the predicted discriminant score for group t, $x - i$ is predictor i, and λ_{t_i} is the weight given to predictor i in group t.

$$D_t = \lambda_{t_0} + \lambda_{t_1} x_1 + \lambda_{t_2} x_2 + ... + \lambda_{t_p} x_p \tag{4.1}$$

A discriminant function in this form is calculated for each group. The goal of this step is to calculate t functions that maximize the differences in the groups.

Once the discriminant functions are calculated, the discriminant classification analysis can begin. This step classifies items into one of the groups. Initial classifications are done with items whose group membership is known, or verification items. This serves to verify the discriminant functions. The verification items classified were the same thirty-two texts. The result of performing a discriminant classification on verification items is a $t \times t$ confusion matrix that compares the predicted membership and the actual membership for each verification item. This matrix is the measure used in this thesis to determine how well the discriminant function predicts membership.

This matrix can also be used to calculate the proportion correct for each group. The proportion correct is the number of items correctly classified as in that group divided by the number of items that actually belong to the group. The proportion correct can also be calculated for the analysis as a whole. This value, again, is the number of items classified correctly divided by the total number of items being classified. The proportion correct will be the values used through much of this thesis to evaluate the predictive power of predictor sets.

The relative importance of any individual predictor can be calculated by running a discriminant analysis with just that one predictor. The proportion correct is the same as the relative importance for that indicator.

Unfortunately, linear models often capitalize on chance and base their predictions too heavily on the initial items. The result is overestimated performance or an optimistic apparent error rate when classifying items used in the creation of the discriminant functions. To eliminate this overestimate, the discriminant classification analysis was calculated using cross-validation. Cross validation is a simple way to test the accuracy of the groupings and compensate for an optimistic apparent error rate. Cross validation systematically eliminates each item, recalculates the discriminant functions, and then tries to classify the removed item into a group. This means that the texts are not included in the discriminant predictive analysis step when they are

Reference Name	Link Types Subset
LS1	**AL AN CO DG E K PP RW X YS**
LS2	**AL AN CO DG K PP RW YS**
LS3	**AL AN CO DG PP YS**
LS4	**AL AN CO DG E PP X YS**
LS5	**AL AN CO DG E PP X**
LS6	**AL AN C CO DG E K PP RW YS**
LS7	**AL AN CO DG** PP **RW YS**
LS8	**AL AN CO DG** E PP **YS**
LS9	**AL AN CO DG E PP**
LS10	**AL AN CO DG PP**

Table 4.2: Subsets of Link Types and Reference Names for the Subsets

used for discriminant classification analysis. The proportion correct can then be calculated on the result. This proportion correct value is much more indicative of the predictive power of the predictors than the proportion correct value found from classifying items that were also used in the discriminant function.

Results

After starting to calculate a discriminant analysis using these predictors, a number of these link types were found to be highly correlated with each other. **J** was the first link type to be removed due to a high correlation. The other link types eliminated from consideration were **A**, **B**, **C**, **D**, **EB**, **J**, **L**, **M**, **MV**, **O**, **OF**, **P**, **R**, **S**, **TH**, **TO**, and **W**.

Ten link types remained for consideration for authorship attribution: **AL**, **AN**, **CO**, **DG**, **E**, **K**, **PP**, **RW**, **X**, and **YS**. Subsets of these ten link types are listed in Figure 4.2 along with reference names for them. These sets will be referred to by their names from this point on.

These ten link types were processed individually and as a group to try to find an optimal subset. A discriminant function was calculated for each link type. This discriminant function only used that single link type as a predictor. Then all texts were then classified with that discriminant function

Links	AL	AN	CO	DG	E	PP	X
PC	.250	.313	.375	.188	.344	.531	.125

Table 4.3: Relative Importance Using Cross Validation for Each Element of LS1

using cross-validation. This calculation produced the relative importance, or discriminatory power, of each of the link types. The results of this analysis can be seen in Table 4.3. PC stands for the proportion correct, or the proportion of texts correctly classified using that link type.

After initial tests were run, LS2 seemed to be a good starting point. Its predictive power was .688. Each link in this set was then systematically removed and another discriminant analysis was run using the given subset. Most of the time, removing a link type decreased the predictive power of LS2, but removing link type **RW** improved the predictive power to .719 PC. Removing link type **K** made no change to the predictive power so it too was removed from the set resulting in LS3.

Once again, each link in LS3 was removed and another analysis was run. Once again, removing most link types decreased the proportion correct, but removing **YS** improved the predictive power to .750 and removing **CO** made no change.

The same kind of analysis was run for LS4 and LS5. Each link type in the set was removed and a discriminant analysis was run with improvements being made as the proportion correct rose. The result were five combinations of link types which all resulted in a .750 PC. These five combination are LS6-LS10.

The average number of words per sentence is a standard statistic included in many authorship analysis techniques. When this statistic was added to the predictors, most of these sets improved. LS8 improved the most to achieve .844 PC. LS9 improved to .813 PC and LS10 improved to .781 PC. LS6 remained the same with .750 PC. LS7 did the worse. It dropped to .668 PC.

Table 4.4 shows detailed results from the discriminant analysis of LS8. Group names match with the initials of the authors. Columns indicate the texts written by each author and rows indicate the text attributed to each author by the discriminant function. For example, Agatha Christie (AC) wrote two of the books in the corpus, but the discriminant function also attributed one book but Henry James (HJ) to her. The bottom three rows

	True Group							
Group	**AC**	**PC**	**EF**	**IG**	**HJ**	**BR**	**EW**	**PW**
AC	2	0	0	0	1	0	0	1
PC	0	1	0	0	0	0	0	0
EF	0	0	4	0	1	0	0	0
IG	0	0	0	2	0	0	0	0
HJ	0	0	0	0	1	0	0	0
BR	0	0	0	0	0	2	0	0
EW	0	1	0	0	0	0	4	0
PW	0	0	0	0	0	0	1	11
Total	2	2	4	2	3	2	5	12
Correct	2	1	4	2	0	2	4	11
PC	1.000	0.500	1.000	1.000	0.333	1.000	0.800	0.917

Table 4.4: Discriminant Classification Analysis with Cross-Validation for LS8

state the total number of texts written by each author, the number of texts correctly attributed by the discriminant function to that author, and the proportion correct.

As can be seen, Gillmore had the lowest proportion correct with only one of her three texts attributed to her. Colum only faired slightly better with one of his two texts attributed to him. These individual results were unusual. Throughout the tests, one text by Wodehouse was attributed consistently to Forster. Other common mistakes included misattributing texts by Wodehouse to Wharton and texts by James to Christie.

Table 4.5 shows the λ values for the discriminant functions for LS8. Recall that a separate discriminant function exists for each group. Each predictor is then weighted by its λ value. A constant value is then added in at the end. The first column of Table 4.5 are these constant values. The second column are the weights for the words per sentence (WPS). The rest of the columns are the link types used as predictors in LS8. This table tells us a little about the weights given to each predictor. Note that Table 4.5 are the λ values for LS8 when all the texts are used to find the discriminant functions and cross-validation is not used.

Group	Const.	WPS	AL	AN	CO	DG	E	PP	YS
AC	-176	37	-6858	-278	234	494	-635	557	-492
PC	-600	72	-12275	-642	399	1073	-1390	1009	-1093
EF	-181	37	-6012	-259	198	552	-618	628	-670
IG	-340	35	-5524	66	317	743	-532	842	-1077
HJ	-168	31	-5741	-256	276	463	-439	527	-479
BR	-451	65	-8904	-421	186	630	-958	356	-849
EW	-365	53	-9510	-361	262	750	-935	912	-781
PW	-233	42	-7599	-266	209	632	-772	768	-708

Table 4.5: λ Values for Discriminant Functions for LS8

4.3.2 Cluster K-Means Analysis

An Overview of Cluster K-Means Analysis

A cluster K-means analysis was also performed on the authors. Cluster K-means is generally used to find groups that seem "close" to each other in observations whose groups are otherwise unknown. Unlike discriminant analysis, it tries to find the similarities between items and group them by their similarities. This way, items in the groups found are more like other items within the same group then the items found in other groups. Cluster K-means was used to see if the groups it returned matched well with the actual groups of authors.

Cluster K-means is a partitioning method which means that cluster centers are determined and items are reallocated to different groups until the similarity measure of all groups is maximized. The similarity is inversely proportional to the Euclidean distance measure in an n-dimensional space where n is equal to the number of predictors used.

The only information provided to the cluster K-means analysis is the number of groups to look for. Each item is initially randomly assigned to a group. Items are then reallocated to other groups to minimize the total distances (or maximize the total similarities) of the items in each group while maximizing the distance (or minimizing the similarity) between items in separate groups.

It should be noted that cluster K-means will not necessarily improve matching author with texts when more texs are added. Nor can a cluster analysis be "trained" to relate an author with a text. It is only used here as a

Author	Group	Author	Group	Author	Group
AC	1	HJ	2	PW	2
AC	2 **2**	HJ	3	PW	2
PC	5	HJ	4 **4**	PW	2
PC	8 **8**	BR	6	PW	3
EF	1	BR	6 **6**	PW	3
EF	1	EW	5	PW	3
EF	1	EW	5	PW	3
EF	3 **1**	EW	7	PW	3
IG	5	EW	7	PW	4
IG	5 **5**	EW	8 **7**	PW	4
				PW	4
				PW	8 **3**

Table 4.6: Groups of Texts Returned by Cluster Analysis Ordered by Author

complementary analysis method to the discriminant analysis used in Section 4.3.1. Cluster analysis should show if authors styles are intrinsically different enough from each other to form distinct groups but is not likely to then be able to attribute a text to a particular author.

Results

Three cluster K-means analyses were run. The first used all the link types listed in Table 4.1. A second analysis was run using LS6 and the third used LS10. All three analyses returned the same results. Table 4.6 show the texts organized by author and the group ID assigned to each text by the cluster K-means analysis. The number in bold is the group ID given to that author. The group ID assigned to each author optimizes the proportion correct for each author while maintaining a one-to-one correlation between the authors and the group IDs.

As Table 4.6 shows, there is not a clear correlation between authors and the groups returned by the cluster analysis. Six of the eight authors have texts belonging to more than one group and three have texts belonging to three or more groups.

Table 4.7 shows the authors and the groups the K-means analysis returned. The values in bold mark the total number of texts whose author and

	Authors								
Group	**AC**	**PC**	**EF**	**IG**	**HJ**	**BR**	**EW**	**PW**	**PC**
1	1	0	**3**	0	0	0	0	0	0.750
2	**1**	0	0	0	1	0	0	3	0.200
3	0	0	1	0	1	0	0	**5**	0.714
4	0	0	0	0	**1**	0	0	3	0.250
5	0	1	0	**2**	0	0	2	0	0.400
6	0	0	0	0	0	**2**	0	0	1.000
7	0	0	0	0	0	0	**2**	0	1.000
8	0	**1**	0	0	0	0	1	1	0.333
PC	0.500	0.500	0.750	1.000	0.333	1.000	0.400	0.417	

Table 4.7: Authors and the Groups Returned by Cluster K-Means

group ID match where the group ID for the author is the same as the one given in Table 4.6. Proportions correct are calculated for both each author and each group and shown in the appropriate column or row. The total proportion correct for the cluster analysis was .531, or a little better than 50%.

This result is significant. Out of eight authors, a cluster K-means correctly assigns text to an author more than half the time. This shows a reasonable grouping intrinsic within each author's texts. Nevertheless, these results only serve as an interesting feature to note. They are not strong enough to warrant further analysis and definitely not strong enough to have any application in authorship attribution.

Chapter 5

Collins' Statistical Parser

Michael Collins' parsing techniques have influenced NLP greatly. Before developing his parser, his parsing models surpassed the current parsing benchmarks for the Penn Treebank[27]. Later, for his thesis, Collins implemented his theories into a statistical parser. This parser is the second program used in authorship attribution.

Collins' Parser[14] takes as input a tagged sentence like the one below, and produces a phrase structure tree like the one in Figure 5.1. Tagging is explained in greater detail in Section 5.2.

John/NNP Smith/NNP, the/DP president/NN of/IN IBM/NNP,
announced/VBD his/PRP$ resignation/NN yesterday/NN.

The resulting tree shows how the sentence is made of a noun phrase and a verb phrase, each of which contain other parts of speech. The second to last level in the tree will always be the tag and the bottom most level is the word itself.

5.1 Theory

Collins' Parser is a lexicalized probabilistic context-free grammar (PCFG). Section 3.6.1 described PCFGs. A lexicalized grammar contains a lexicon and a set of grammar rules. The lexicon matches words with linguistic representations. In the case of Collins' Parser, these linguistic representations are tree structures. The grammar rules define how the linguistic representations can be combined to form larger linguistic representations. Using a

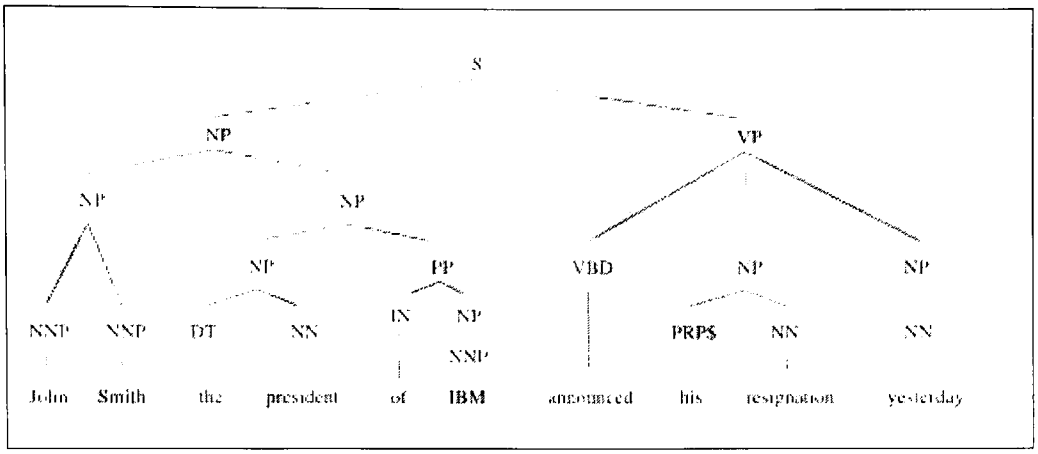

Figure 5.1: An Example Phrase Structure Tree for the Tagged Sentence "*John Smith, the president of IBM, announced his resignation yesterday.*" [14]

lexicalized grammar to parse a language means applying the grammar rules to the words and linguistic representations.

With any lexicalized grammar, each parse must decide which grammar rule to follow at each step in the parse. As sentences get longer and more complex, the number of choices that must be made increases. The probabilities associated with the grammar assist in that decision making process. The most probable parse is also dependent on a distance metric. This distance metric is based on the distance between the head and the modifier and the order in which they appear.

Collins' Parser uses the probability of word-word dependencies to determine the most likely parse. Each word-word dependency contains a head and a modifier. The head is the central constituent of a phrase that determines the phrase's syntactic character and at this level is the word that fulfills that role. The parsing algorithm is a bottom-up depth-first search and uses the probabilities of each grammar rule to decide which path to take. Collins' Parser also uses a beam search strategy to improve the efficiency of the program. A beam search strategy sets a pre-defined beamsize, β. When searching for a parse for a series of words, the constituent with the highest probability, P_h, is saved. All other constituents that occur within the series of words must have a probability higher than $\frac{P_h}{\beta}$.

Later dependencies are built up from the first level word-word dependencies. These later dependencies connect words to sub-trees or sub-trees to sub-trees until a full tree structure has been created. Each of these later dependencies also contain a head and a modifier. The head of any of these later dependencies is the head of the head sub-tree. In this way, the head of the sentence propagates up the tree until it reaches the top. For example, the main noun in a noun phrase is the head of the phrase. The main verb of a verb phrase is its head. The head of the entire sentence is the verb phrase. In Figure, 5.1, *Smith* is the head of the **NP** *John Smith* and *John Smith the president of IBM. announced* is the head of the verb phrase. Because the verb phrase is the head phrase of the sentence, *announced* propagates up the tree and serves as the head for the entire sentence. It is this use of head words and head phrases that makes Collins' Parser different from the Link Grammar described in Chapter 4.

This relationship between the head and the modifier is similar to the Link Grammar in Section 4 since independent "links" could be made between the head and each of its modifiers.

The parsing program is based on the probabilities of dependencies between heads in the parse tree[14]. If two possible parse constituents have the same words, labels, head, and distance value, then the constituent with the higher probability is accepted. The parser follows a bottom-up algorithm. A formal grammar is replaced with the probabilities of the training set.

5.2 MXPOST Tagger

Collins' Parser expects the input to be tagged. Tagging involves matching each word in a sentence with the part of speech the word plays in that sentence. MXPOST, the tagger used by Collins, was used to tag all of the text to be used in this study.

MXPOST uses a variety of contextual features to attach a tag to a word. Two words on either side of the word are used. The two words before the word to be tagged are already assigned tags and this information is included in the context. Within the word itself, prefixes and suffixes are used to help identify unknown words. Also included is information about whether the word contains a number, a hyphen, or an uppercase letter.

MXPOST was trained on the Penn Treebank and it was this previously trained version of MXPOST that was used. The input required for MXPOST,

though, requires that it be tokenized. An input file with one sentence per line is broken into its constituent parts. That means that all conjunctions and punctuation marks are isolated from the originating words. For example, using the sentence "*No, it wasn't Black Monday.*" the output would look like the following:

```
No , it was n't Black Monday .
```

This tokenized input is now tagged and the output looks like the following:

```
No_RB ,_, it_PRP was_VBD n't_RB Black_NNP Monday_NNP ._.
```

Unfortunately, this output is still not quite formatted correctly for Collins' Parser. The number of words per sentence must be added to the beginning of each line and the underscores must be removed. The resulting input for Collins' Parser looks like the following:

```
8 No RB , , it PRP was VBD n't RB Black NNP Monday NNP . .
```

5.3 Program

This parser was initially trained on the Penn Treebank. The statistical model used for this thesis is the same one produced from that original training set.

Three models can be used with Collins' Parser. The model used in the following experiments was Model 1 with the associated events file and grammar. A beamsize of 10000 was used. The punctuation flag, adjacency condition flag, and verb condition flag were all turned on. The NPB flag was also set so as to minimize extra levels of NPs.

An example output for the sentence "*No it wasn't Black Monday.*" can be seen in Figure 5.2. The first line states the number of edges in the parse followed by the overall log probability of the parse.

This is followed by the parse printed out word for word. These lines each contain the constituents the word is a part of, the log probabilities for each constituent, and finally the word itself.

The next line shows the entire parse. Words are separated from their tags with a '/' and all punctuation marks are marked with a PUNC tag. Parentheses delineate trees. The tree for this example is shown in Figure 5.3. Non-terminals, or the head words, contain further information all separated

```
PROB 534 -34.129 0
TOP -34.129 S -31.1239 INTJ -0.108482 UH 0 No
      NP -0.00523075 NPB -0.000436731 PRP 0 it
      VP -15.8421 VBD 0 was
         RB 0 n't
         NP -4.28928 NPB -4.23083 NNP 0 Black
                NNP 0 Monday
(TOP~was~1~1 (S~was~3~3 (INTJ~No~1~1 No/UH ,/PUNC, ) (NPB~it~1~1
it/PRP ) (VP~was~3~1 was/VBD n't/RB (NPB~Monday~2~2 Black/NNP
Monday/NNP ./PUNC. ) ) ) )
TIME 0
```

Figure 5.2: An Example of Collins' Parser Output for the Sentence "*No, it wasn't Black Monday.*"

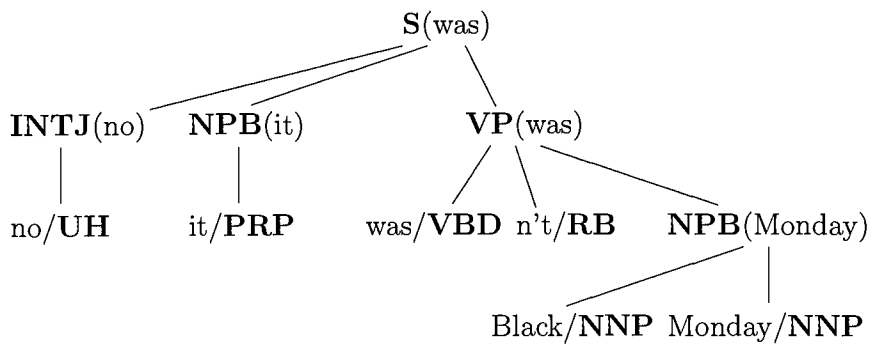

Figure 5.3: Parse Tree for the Sentence "*No, it wasn't Black Monday.*"

by a ' ' mark. This information, in order, is the non-terminal label, the head word, the total number of subtrees, and the constituent number from with the head word is found.

Finally, the last line states how long the parser took to parse the sentence in seconds.

5.4 Experiments and Results

The experiments run with the Collins' Parser were very similar to those run with the Link Grammar. See Section 4.3. Texts and authors used were the same and preprocessing occurred in the exactly same way.

The texts by each author were then divided into files of 2500 sentences each. Each sentence in these files were then tokenized and formatted. The result was run though Collins' Parser in batch and recombined to form a single parse file for each text.

Statistics about grammatical features were collected from all sentences parsed with a probability greater than zero. The features included the average number per sentence for each tag and twenty-six non-terminals. Also included were the average number of subconstituents in each non-terminal. This resulted in 89 features.

All of the features collected were compiled into a large table. Features whose average value were in the tenths of a percent were eliminated. Reasonable authorship attribution should work on small text samples and just like with the Link Grammar, features averaging in the tenths of a percent were deemed too infrequent to be of much practical use. After further analysis, forty-three features proved interesting and a statistical analysis of these followed. Table 5.1 lists the features that the tagger returned and Table 5.2 lists the constituent features returned by the parser.

Tags	Description
CC	coordinating conjunction
DT	determiner
IN	preposition or subordinating conjunction
JJ	adjective
MD	modal

continued on next page

continued from previous page

Tags	Description
NN	noun, singular or mass
NNS	noun, plural
NNP	proper noun , singular
PDT	predeterminer
POS	possessive ending
PRP	personal pronoun
PUNC	punctuation
RB	adverb
RBR	adverb, comparative
RP	particle
TO	"to"
VB	verb, base form
VBD	verb, past tense
VBG	verb, gerund or present participle
VBN	verb, past participle
VBP	verb, non-3rd person singular present
VBZ	verb, 3rd person singular present
WDT	wh- determiner
WRB	wh- adverb

Table 5.1: Summary of Tags Used as Features

Constituent	Description
ADJP	adjective phrase
ADJP_s	average number of subconstituents in an ADJP
ADVP	adverb phrase
NP	noun phrase
NP_s	average number of subconstituents in a NP
NPB	base noun phrase, (used to eliminate recursive definitions for NP)
NPB_s	average number of subconstituents in a NPB
PP	prepositional phrase
PRT	particle
S	simple declarative clause

continued on next page

```
PROB 534 -34.129 0
TOP -34.129 S -31.1239 INTJ -0.108482 UH 0 No
      NP -0.00523075 NPB -0.000436731 PRP 0 it
      VP -15.8421 VBD 0 was
         RB 0 n't
         NP -4.28928 NPB -4.23083 NNP 0 Black
                NNP 0 Monday
(TOP~was~1~1 (S~was~3~3 (INTJ~No~1~1 No/UH ,/PUNC, ) (NPB~it~1~1
it/PRP ) (VP~was~3~1 was/VBD n't/RB (NPB~Monday~2~2 Black/NNP
Monday/NNP ./PUNC. ) ) ) )
TIME 0
```

Figure 5.2: An Example of Collins' Parser Output for the Sentence "*No, it wasn't Black Monday.*"

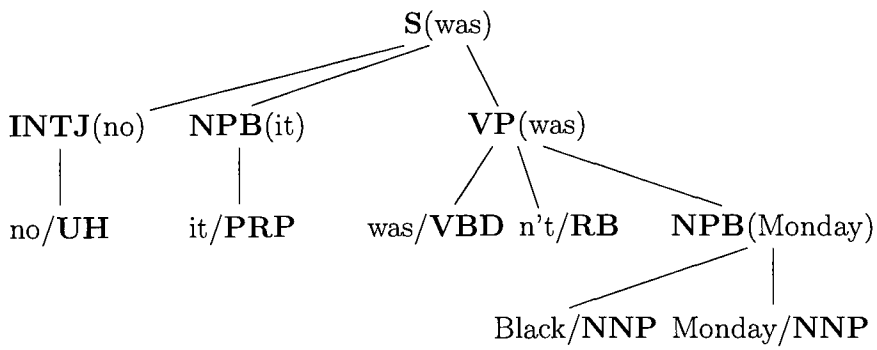

Figure 5.3: Parse Tree for the Sentence "*No, it wasn't Black Monday.*"

5.4.1 Discriminant Analysis

A basic introduction to discriminant analysis, cross-validation, and the methods used in this section can be found in Section 4.3.1.

Initial attempts to run a discriminant analysis with all of the forty-tree features listed in Tables 5.1 and 5.2 did not succeed. Too many of the features were highly correlated with each other. The highest proportion correct found was only .516. Further analysis focused on the average number of constituents in a sentence and the average number of subconstituents found within each constituent. The reason for this focus was two-fold. First, simply looking at the tags given to the words did not used the full power of the parser (in fact, did not use the parser at all since a separate tagging program was used to find the tags). Second, it was hoped that more interesting indicators not previously studied in authorship attribution studies might be found if the constituent features were examined more closely rather than the tags.

The constituent features study that followed did indeed find interesting features that proved to be good indicators of authorship. All the constituent features were initially used in a discriminant analysis. This resulted in .742 PC. Constituent features were individually removed due to high correlations between them and other constituent features. The resulting set was **ADJP_s**, **ADVP**, **PRT**, **SBAR_s**, **SBARQ**, **SQ_s**, **VP**, **VP_s**, **WHADVP**, **WHNP**, **NP_s**, **NPB**, **NPB_s** and had .806 PC.

Each constituent feature was then removed individually to see if by removing any one constituent feature the proportion correct was improved. Removing **NPB**, **VP_s**, **SBAR_s**, and **ADJP_s** each improved the proportion correct to .839. Removing **SQ_s** and **PRT** made no difference in the proportion correct. These constituent features were removed from the test set in various combinations to see which one or more than one optimized the improvement. The best proportion correct was .903.

Again, each feature from the resulting set was removed to see if improvements were made. No improvements were found. The result was the optimal set of constituent features found below. This set of features returned a proportion correct of .903 and will now be referred to as CS for Constituent Set.

ADVP NP_s NPB_s PRT SBARQ VP WHADVP WHNP

The relative importance of each of these features are shown in Table 5.4.

CS	ADVP	PRT	SBAQ	VP	WHADVP	WHNP	NP_s	NPB_s
PC	.452	.323	.258	.452	.290	.290	.323	.355

Table 5.4: Relative Importance Using Cross Validation for Each Element of CS

The next series of experiments added back in all constituent features previously removed as well as each tag feature. These features were added to CS one by one to see if improvements were made. Six features improved the CS to .935 PC. One was a constituent features and the remaining five were tag features. These features are listed below and will now be referred to as SFS for Supplementary Feature Set.

Constituent Feature: **ADJP**
Tag Features: **PUNC RB TO VBG VBP WDT**

Combinations of features from SFS were added to CS, but no combination improved CS beyond .903 PC, a proportion correct that was achieved by CS alone. Any one feature from SFS could be added to CS with the same results, though. In all cases, all texts were correctly classified except for two texts by Henry James. Table 5.5 shows the results. Group names are the initials of the authors. Columns indicate the texts written by each author and rows indicate the text attributed to each author by the discriminant function. The last three rows state the number of texts by each author, the number of texts correctly attributed to that author, and the proportion correct.

The λ values for the discriminant functions differed greatly depending on what feature from SFS was added to CS. No consistent pattern was found in the weights given to the features.

5.4.2 Cluster Analysis

A basic introduction to cluster K-means analysis and the methods used in this section can be found in Section 4.3.2.

A cluster analysis was performed on CS. Just as in the Link Grammar, this analysis resulted in a little correlation between the groups and the actual authors, but not enough to warrant more than a passing curiosity.

	True Group							
Group	**AC**	**EF**	**IG**	**HJ**	**BR**	**EW**	**PW**	
A.C.	2	0	0	1	0	0	0	
E.F.	0	4	0	0	0	0	0	
I.G.	0	0	3	0	0	0	0	
H.J.	0	0	0	1	0	0	0	
B.R.	0	0	0	0	3	0	0	
E.W.	0	0	0	1	0	5	0	
P.W.	0	0	0	0	0	0	11	
Total	2	4	3	3	3	5	11	
Correct	2	4	3	1	3	5	11	
Proportion	1.000	1.000	1.000	0.333	1.000	1.000	1.000	1.000

Table 5.5: Discriminant Classification Analysis with Cross-Validation for CS and One Feature from SFS

Further analyses were performed using CS in combination with the elements in SFS. These too resulted in very little correlation. The only combinations which improved over the analysis using CS are listed below, but even these combinations resulted in very poor correlations between the authors and the groups returned by the cluster analysis.

- **ADVP PRT SBARQ VP WHADVP WHNP NP_s NPB_s ADJP**
- **ADVP PRT SBARQ VP WHADVP WHNP NP_s NPB_s RB**
- **ADVP PRT SBARQ VP WHADVP WHNP NP_s NPB_s TO**

Another analysis was performed using all of the features in Table 5.2. These results were in line with those using CS and various elements from SFS.

Further subsets of features were analyzed with similar results. The only cluster analysis which stood out as remarkably better than any of the others was one using the features in Table 5.1. Table 5.6 shows the results of this analysis.

	Authors							
Group	**AC**	**EF**	**IG**	**HJ**	**BR**	**EW**	**PW**	**PC**
1	0	0	0	0	0	0	**4**	1.000
2	**2**	0	0	0	0	0	7	0.222
3	0	0	0	**1**	0	0	0	1.000
4	0	**4**	1	0	0	0	0	0.800
5	0	0	**2**	0	0	0	0	1.000
6	0	0	0	1	**2**	0	0	0.667
7	0	0	0	1	1	**5**	0	0.714
PC	1.000	1.000	0.667	0.333	0.667	1.000	0.364	

Table 5.6: Authors and the Groups Returned By Cluster K-Means Performed on Tag Set

Chapter 6

Conclusions and Comparisons

Section 4.3 and Section 5.4 present the experiments and results of those experiments for the Link Grammar and Collins' Parser respectively. This chapter states the conclusions found from these results. Section 6.1 states the conclusions made from the results of the Link Grammar. Section 6.2 states the conclusions made made from Collins' Parser. Section 6.3 then compares the results of the two parsing programs.

6.1 Link Grammar

The Link Grammar shows promising results. Using discriminant analysis with cross-validation and one of six link type sets listed in Table 4.2, a text was correctly matched with one of eight authors 75.0% of the time. When simple and common authorship attribution techniques are included the results improved further. A dramatic increase in the accuracy was seen when the average words per sentence was included in the analysis. Adding this feature improved the proportion correct to .844. Further improvements may be made if link type usage was used with other authorship attribution methods such as bigrams or the use of function words.

The number of link types included in the analysis seemed to be quite important. Increasing the number of link types included in the analysis increased the proportion correct but did not increase the proportion correct using cross-validation (the PC value used throughout this study). Generally, increasing the number of link types included dramatically decreased the proportion correct using cross-validation. If too few link types were used,

though, both PC values would decrease. The optimal number of link types seemed to be in the range of five to ten.

A couple of the links types found to be significant in authorship attribution are not currently included in other authorship attribution methods. The link type **AN**, for example, connects noun modifiers with the following noun. The usage of noun modifiers is not currently included in any authorship attribution methods found in this study.

The link type **DG** connects the word *the* with a proper noun. Although studies involving the use of the word *the* are included in authorship attribution methods that use function words, none specifically look at the use of the word *the* near proper nouns. The link type **D** connects determinators to the following noun. It may be significant to note that this link type was not found to be a significant predictor of authorship. *The* is more commonly used before nouns than proper nouns, and perhaps methods using function words would improve if the study of the usage of the function word *the* was limited to specific uses, such as uses near proper nouns.

Methods looking at the use of the possessive are also not generally included in authorship studies yet the link type **YS** found as a reasonable predictor of authorship.

Many of the link types, though, do align closely with methods already in use. Function words generally include the determinators indicated by the link type **AL**, the "openers" in **CO**, the adverbs in **E**, and forms of the verb "have" in **PP**. Both **RW** and **YS** are punctuation dependent and authorship attribution methods often employee some analysis of punctuation usage as well.

Four British authors were included in the study: Agatha Christie, P.G. Wodehouse, Edward Forster, and Bertrand Russell. Throughout the tests using the Link Grammar, one text by Wodehouse was consistently attributed to Forster. Occasionally, texts by Wodehouse were attributed to Christie. Initial speculation might assume that the Link Grammar was using dialect to discern between the authors. This does not seem to be the case, though. Wodehouse also had texts consistently attributed to Edith Wharton (an American author) and Henry James (another American author) would be attributed frequently to Christie and occasionally to Wodehouse or Forster.

Text attribution and misattribution did not seem to be connected to gender either. The two most frequent misattributions, texts by James attributed to Christie and Wodehouse attributed to Wharton, crossed gender lines.

Henry James and Edith Wharton knew each other during their years

of writing. They were both raised in New York City and frequently corresponded. Wharton's admiration of James extended to the point of trying to model his writing style in her book **The Reef**. It is interesting to note that this novel is very rarely misattributed in this study and even more rarely misattributed to James. This implies that the features picked up by the Link Grammar are unconscious and not easy to emulate.

6.2 Collins' Parser

The results using the Collins' Parser show great promise with regards to authorship attribution methods based on syntactic structures. A wide variety of features can be used as predictors to return higher than 90% accuracy using discriminant analysis. All of these results are significantly higher than the results returned by the Link Grammar, even when the Link Grammar made use of the average words per sentence.

Many of the syntactic features found significant in this study are not currently examined in current authorship attribution methods. The existence of wh- element and phrases were repeatedly found to be good indicators of authorship. The constituents **SBARQ**, **WHADVP**, and **WHNP**, and the tag **WDT** all match with phrases and words linked to the use of wh- elements. No authorship attribution method currently makes use of wh- elements.

No authorship attribution methods examines the length of phrases either. This study shows that the length of noun phrases are good indicators . **NP_s** and **NPB_s** are the average number of subconstituents found within each **NP** and **NPB** respectively. These two features were found consistently to be good predictors of authorship.

The use of verbs in the gerund, past participle, and non-3rd person singular present also predicted authorship. The use of verbs in this detail has not been used in other authorship attribution studies.

Other indicators found do match with methods currently in use. Methods using function words may look at particles, some adverbs, and the word "to." Punctuation is used by other attribution methods.

Just as with the Link Grammar, Collins' Parser did not seem to misattribute text according to gender or dialect. Many of the texts by Henry James were misattributed to Agatha Christie, just as with the Link Grammar. Unlike the Link Grammar, though, texts by Wodehouse were generally misattributed to Christie rather than Wharton. Many of the mistakes of at-

tribution involving text by James were attributed to Wharton even though it was Wharton who emulated James' style,

6.3 Parsers Compared

Link types seem to be more regular across writing styles than tags or constituents. Although this may be a useful feature for parsing, it means that link types do not contribute to authorship attribution as well. The length and usage of constituents are quite good at discriminating between authors. Tags are also valuable in attributing authorship.

In both the Link Grammar and Collins' Parser, the results reinforced the use of function words as a valuable means to determine authorship. Both, though, also suggested other words and word types that could serve as features to distinguish between authors. The most obvious such feature are wh-words. These words could easily be counted, just like function words, and added to authorship attribution methods already in existence.

Styles in writing and dialects did not appear as major patterns in this study. Misattributions made the Link Grammar did not follow lines of gender or dialect at all. Collins' Parser may have used differences in British verses American English more, but even then, one of the most common misattributions was attributing texts by Henry James to Agatha Christie.

Texts by Bertrand Russell were very rarely misattributed to others. The cluster analysis performed with the Link Grammar returned a one-to-one correspondence between his texts and Group 6. This is impressive since no other cluster analysis of features from either parser returned such a direct correlation between a group and an author. Russells distinctiveness may be at least in part due to his more formal writing style. he was the only non-fiction writer included in this study and the differences inherent in the writing styles of fiction verses non-fiction writers may have contributed to him being so much easier to distinguish.

Texts by Colum were also rarely misattributed to others. This may be because of an Irish dialect, or a poetry background.

Gillmore and Forster were also consistently attributed to them. Gillmore may also have a dialect since she was born in Brazil, but she was born in an American home and lived most of her childhood in Boston so dialect cannot be the only reason. Forster's history also does not lend to explaining why his texts were so easily attributable to him. Gillmore was also well grouped by

the cluster analysis of both parsers. Forster was well grouped by the cluster analysis of Collins' Parser. Stylistic features inherent in the writing styles of these two authors must make them distinctive from their counterparts.

As mentioned, Wharton and James are well recognized as having similar writing styles. It is then impressive that texts by these two authors can be correctly attributed to them. Despite Wharton's admiration and emulation of James' writing style, her texts were rarely attributed to him. In the cluster analysis of Collins' Parser, she was also found to be distinctive enough to form a reasonably strong correlation between her and Group 7.

This study has shown two things. First, grammatical structures are a reasonable way to distinguish between texts by different authors. Links, tags, and sentence constituents provide enough information to attribute authorship. Second, parsers provide a means of learning about features that can be used to attribute authorship that have otherwise been overlooked. Some features found in this study, such as the use of wh- words, may not need the full power of a parser to be used in authorship studies, but it was the analysis of the information returned by the parser that pointed to them as another feature that can be used in other sources.

Chapter 7

Future Work

The results in this thesis are interesting, but there are still a number of different areas which could be expanded on. This chapter looks at some other research that could be done to expand on this thesis. Section 7.1 looks at alteration to the corpus that may expand on and improve on this study. Section 7.2 looks similarly at the statistical methods used. Section 7.3 suggests other authorship attribution methods that could be combined with the results of this thesis to produce better authorship attribution methods. Finally, Section 7.4 suggests another parser that might produce results that could compliment or add to the results already found.

7.1 Corpus

Perhaps most important for future studies, this work should be repeated. It is quite possible that the link types found in the Link Grammar and the features found in Collins' Parser worked well for the eight authors used in this study, but perhaps other link types or features would predict authorship better in general.

More than two texts should be included for each author. Some authors may only have one text included in the initial data since their other text may be removed for cross validation.

Future research should go into recreating these experiments with different sets of texts. Rather than comparing a variety of authors who are writing about different topics and are from different parts of the English speaking world, a corpus should be used that contains authors with a similar English

	Link Type						
	AL	**AN**	**CO**	**DG**	**E**	**PP**	**X**
Linear PC	.313	.406	.438	.250	.375	.531	.281
Quadratic PC	.531	.563	.438	.375	.531	.625	.438

Table 7.1: Comparison of the Relative Importance of Seven Link Types using Linear verses Quadratic Discriminant Analysis

dialect, similar style, and a similar reason for writing. A study using texts of this sort would test for link types and features that distinguish authors and do not depend on dialect, topic, or writing styles.

Other research could compare and contrast the link types and features which best perform authorship attribution in different styles of writing. Perhaps works of non-fiction are better distinguished by different features then works of fiction. Perhaps email and journal entries would need a completely different set of features again. A corpus with a set of authors each writing in a variety of different styles should be tested. Research using texts of this sort would more accurately test to see if the link types and features used as predictors correlate well to an author no matter what writing style he or she employees at that moment.

Parsers exist for a wide variety of languages. Studies using these other parsers may find that similar link types or features are useful for authorship attribution across languages.

7.2 Statistical Methods

This study focused its statistical analysis on linear discriminant analysis. Cluster K-means was also used to explore the natural groupings of the texts. Future studies might consider using a quadratic discriminant analysis. Quadratic discriminant analysis normally results in better groupings. Table 7.1 compares the proportion correct using linear discriminant analysis verses quadratic discriminant analysis. This table shows seven link types and their relative importance. This data was collected on the thirty-two texts used in this study and did not use cross-validation. As can be seen, performing a quadratic discriminant analysis improved the individual predictive power of these link types consistently and significantly.

Further work needs to be done to bring down the percent of data lost to the parser before this technique can be used. Loosing more than 25% of the sentences is unacceptable, despite the otherwise good results. This loss of data may be minimized by a better method of determining when sentences begin and end.

7.3 Combining with Other Authorship Attribution Methods

As was found in Section 4.3.1 and Section 4.3.2, the Link Grammar alone was not enough to offer a viable alternative method for authorship analysis. When statistics about links types were combined with statistics about average words per sentence, the results improved. Therefore, the use of links may be helpful in raising the accuracy of other authorship analysis methods. Further research should include combining link types proven to correlate well to authorship with other authorship attribution methods. Perhaps bigrams or the use of function words can be combined with the frequency of link types to raise the accuracy of authorship attribution.

It may also be found that parsers can be used to find unique author traits but may not be needed in the long run. Taggers may be enough to add to the reliability of authorship analysis. Their importance was shown already in Section 5.4.1 and Section 5.4.2. It may also be that simply counting wh-words may also help in authorship attribution and the additional information gained from parsers may not be worth the extra computational overhead. Nevertheless, parsing programs do seem to provide a means of learning about a wide variety of authorship traits that may not have been noticed otherwise.

7.4 LinGO: Another Parser

The LinGO (Linguistic Grammars Online) project attempts to provide grammars and grammar development environments online and free for use by the public. The resources they offer include the LKB (Lexical Knowledge Base) grammar development environment and the English Resource Grammar (ERG), which includes a lexicon.

The LKB project provides a general purpose parser as a teaching tool. Written in Common Lisp, it provides a development environment for gram-

mars and lexicon for processing a wide variety of grammars. Running LKB opens an interaction window with menus along the top allowing for a wide variety of commands. Before most commands can be used, a grammar needs to be loaded into the system.

7.4.1 LinGO ERG

ERG is the grammar made available from LinGO for use with LKB. ERG uses the popular HPSG (Head-Driven Phrase Structure Grammar) framework to create a broad coverage grammar. HPSG is similar to some of the examples given in Chapter 3 in that it develops parse trees based on syntactic phrases and sentence constituents. Sentence constituents are types of groups of words that can be replaced with other groups of words of the same type and still produce a grammatically correct sentence.

HPSG assumes two things about languages. The first assumption is that language is built from a set of linguistic objects such as phrases, clauses, person, number, and words. The second is that the structure of language, or its grammar, is best represented as a set of rules. This is different from transformational grammars which define rules for the grammar by describing operations on the linguistic objects.

HPSG finds syntactic features within the sentence such as the noun phrase *the little boy* and the verb phrase *ran fast* in the sentence *The little boy ran fast.* To ensure subject/verb agreement, these sentence constituents are sub-categorized into a variety of types. For example, *the little boy* would be of type NP_{1s} or noun phrase of type first person singular. Once agreement between the subject and verb has been verified, often the subcategories are not needed any more and the type would simply be NP, or noun phrase. These sentence constituents all contain a head-word. The head-word is the most important word of the constituent and contains the important features of the constituent. In the previous example, the head-words might be *boy* and *ran* since they are the noun and verb respectively of the noun phrase and verb phrase. It is this head-word that indicates the sub-category of the sentence constituent.

The ERG also uses a type hierarchy. The type hierarchy defines which linguistic objects can be contained within other linguistic objects. For example, a noun phrase may contain an adjective, a determinator, or both.

Because its initial application was for spoken language translation, this grammar does particularly well with partial ill-structured sentences and in-

Sentence	Parses
Would either Monday or Tuesday morning of the next week work?	2
How did you say your Wednesday was, the twenty ninth?	11
Tuesday, the only time I would have would be at three in the afternoon.	78
So if we cannot make it on Thursday afternoon, we will have to, you know, look for something.	96
Now that we have finished our last meeting, we need to arrange another one within the next two weeks.	128
So actually next the whole of next week is gone.	0
And it is already been pushed off more than two weeks.	0
How does Wednesday the twenty fourth after one o'clock?	0

Table 7.2: Example Sentences and the Number of Parses ERG Returned[15]

complete clauses. Table 7.2 contains examples of sentences ERG parsed and the number of parses it returned. The last three were not successfully parsed by the ERG.

7.4.2 Using ERG With LKB

Once installed, LKB is simple to run.[1] Simply start with the command line argument `lkb` from within the directory. A graphical user interface loads up. From the `Load` menu option, choose `Complete Grammar` and select the script for the grammar to be used. In the case of ERG, a number of grammars can be loaded. The smallest is in `tiniest/grammar1/script` although it was found to be too small for even the simplest of sentences. The most used grammar can be found at `esslli/enddayfour/script`. If this grammar is found to be lacking, further words can be added by inserting them in the `lexicon.tdl` file.

[1]LKB and ERG can be found at `http://lingo.stanford.edu/`.

7.4.3 Reasons This Program Wasn't Included

LKB and ERG do not include a way of running batch processes. As a result, parsing large amounts of text by a number of different authors becomes cumbersome and tedious.

The output produced by the program is also strictly in a GUI format. This inhibits greatly the creation of automated analysis and data gathering processes.

Because of these two reasons, this parser was not included in an authorship attribution analysis for this thesis. This is not to say that this parser will not produce interesting results. In fact, its parsing style compliments the Link Grammar and Collins' Parser well and could produce another set of viable features usable in authorship attribution.

Appendix A

The Authors and Texts Studied

The corpus for this study included eight authors. This appendix includes a brief biography of each author and the books included in the corpus.

A.1 Padraic Colum, 1881-1972

Primarily a poet of the Irish Literary Revival, Colum also wrote childrens' stories, plays, and biographies including one of James Joyce. **The Golden Fleece And The Heroes Who Lived Before Achilles** re-tells classical Greek myths through the story of Jason and the Argonauts. This format for telling the stories of Theseus and the Minotaur, the labors of Hercules, and others awarded him runner up to the first Newberry Medal for best contribution to American children's literature in 1922. **The King of Ireland's Son** is also a retelling of myths. This time he wrote Celtic folklore in the form of short stories.

A.2 Agatha Christie, 1891-1976

Known as the "Queen of Crime," Agatha Christie is one of the world's best known mystery writers. Born in England, she traveled quite a bit as she wrote her novels, plays, and short stories. **The Mysterious Affair at Styles** is Christie's first novel. It features a Belgian detective, Hercule Poirot, as he attempts to outwit a murderer at a manor house in the UK. **The Secret Adversary** was written two years later. It features Tommy and Tuppence, an irrepressible couple short on money who decide to hire themselves out as

"young adventurers, willing to do anything, go anywhere." Their first job puts them into danger and forces them to try to save both their own lives and the life of another mysterious character.

A.3 Pelham Grenville Wodehouse, 1881-1975

P.G. Wodehouse, more commonly known as Plum, was born in England, and died in America. He wrote lyrics, short stories, and theater critiques but is probably best known for his contributions to the British comic *Punch*. He also wrote about one hundred books. Two of his most popular characters are Jeeves (an intelligent and observant butler) and Wooster (his cognitively challenged master). In 1975 he was knighted and entered into Madame Tussaud's Wax Museum. Shortly afterwards, in an interview with the BBC, he said there was very little left for him to do with his life.

In this corpus, he served to provide a large body of texts so that continuity within an author could be analyzed. The novels included in the corpus included **The Clicking of Cuthbert**, **The Coming Of Bill**, **A Damsel In Distress**, **The Gold Bat**, **A Head Of Kay's**, **The Little Nugget**, **Little Warrior**, **Love Among the Chickens**, **Indiscretions Of Archie**, **The Intrusion Of Jimmy**, **Piccadilly Jim**, **A Prefect's Uncle**, and **The Pothunters**.

A.4 Edward Morgan Forster, 1879-1970

Born in London, Forster traveled extensively and many of his stories are based on these trips abroad. He started by writing novels and later in life turned to writing short stories and non-fiction. His first novel, written in 1905, was **Where Angels Fear to Tread**. Two years later, he wrote **The Longest Journey** and a year after that, he wrote **A Room With A View**. This last book, perhaps one of his most famous, is based on material from visits to Italy with his mother. Lucy Honeychurch, the main character, witnesses a murder in the first half of the book and in the second half, deals with her prejudices as she decides whom to marry. In 1910, Forster wrote **Howards End**, a book set at an English country house and focuses on the class between one family whose love is art and literature, and another family whose passion is business.

A.5 Bertrand Russell, 1872-1970

British philosopher, logician, essayist, and peace advocate, Russell is best known for his work on mathematical logic and analytic philosophy. He is the only non-fiction writer included in this study. **The Analysis of Mind** is a series of fifteen lectures focusing on the psychology and physics of the human mind. **Political Ideals** (1917) are lectures on what make good political ideals and **Proposed Roads to Freedom** (1918) is a commentary on society.

A.6 Inez Haynes Gillmore, 1879-1970

Gillmore was born in Rio de Janeiro to Brazil to American parents, although she spent most of her childhood in Boston. She was very active in American women's suffrage and many of her books reflect that. One of the most important early feminist novels, **Angel Island**, was written by Gillmore in 1914. It is about five men who are shipwrecked on an island when they meet five flying women. **The Californiacs** (1916) and its companion novel, **The Native Son** (1919) both explain why Californians are special and different from the rest of Americans.

A.7 Henry James, 1843-1916

Henry James was born in New York City to a wealthy family. Throughout his life, he was friends with some of the most well-known intellectual minds in American philosophy and literature. His own books made him very little money. In fact, his friend, Edith Wharton, secretly arranged to pay the advance on one of his books. **The Tragic Muse** (1890), **The Spoils of Poynton** (1897), and **The Awkward Age** (1899) are the three books this study examines.

A.8 Edith Wharton, 1862-1937

Wharton, also born in New York City, is best known for her stories about upper class life. Many of the themes of her books focused on the conflict between the individual and society, repressed sexuality, and the old money

verses the nouveau riche. She traveled regularly to Europe and wrote reports for American newspapers during World War I.

Wharton's friends included Henry James and much of her writing tried to emulate his style, especially her novel **The Reef** (1912). Her other novels which were included in this study are **The Touchstone** (1900), **The House of Mirth** (1905), **Summer** (1917), **The Age of Innocence** (1920), and **The Glimpses of the Moon** (1922).

Bibliography

[1] Allen, James. **Natural Language Understanding**, The Benjamin/Cummings Publishing Company, Inc., 1987.

[2] Bee, R.E. *Some Methods in the Study of the Masoretic Text of the Old Testament.* **Journal of the Royal Statistical Society**, 134(4), 611-622, 1971.

[3] Bissell, D. *Statistical Methods for Text Analysis by Word-Counts.* Swansea: European Business Management School, University of Wales 1995.

[4] Bosch, Robert A. and Jason A. Smith. *Separating Hyperplanes and the Authorship of Disputed Federalist Papers.* **The American Mathematical Monthly**, Volume 105, Number 7, August-September 1998, pp. 601-608.

[5] Brainerd, Barron. *The Computer in Statistcal Studies of William Shakespear.* **Computer Studies in the Humanities and Verbal Behavior**, Volume 4, Number 1, 1973, pp 9-15.

[6] Burrows, J.F. *Word Patterns and Story Shapes: The Statistical Analysis of Narrative Style.* **Literary and Linquistic Computing**, Vol.2, 1987, pp. 61-70.

[7] Burrows, J.F. *Not Unless You Ask Nicely: the Interpretive Nexus between Analysis and Information.* **Literary and Linquistic Computing**, Vol.7, Issue 2, June 1992, pp. 91-109.

[8] Canter, D and J. Chester. *Investigation into the claim of weighted Cusum in authorship attribution studies.* **Journal of Forensic Linguistics** 4 (1) p.18.

[9] Cavazza, Marc and Pierre Zweigenbaum. *Semantic Analysis and In-Depth Understanding of Technical Texts.* **Applied Artificial Intelligence.**

[10] Cha, Sung-Hyuk and N. Sargur, L.R.B. Schomaker and L.G. Vuurpijl (Eds.), *Multiple Feature Integration for Writer Verification.* **Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition**, September 11-13, 2000, pp. 333-342.

[11] Charniak, Eugene. *Statistical Techniques for Natural Language Parsing.* **AI Magazine**, (1997).

[12] Collins, Michael John. *A New Statistical Parser Based on Bigram Lexical Dependencies.* Proceedings of the 34th Annual Meeting of the ACL, Santa Cruz, 1996.

[13] Coffman, Frank. *Stylometric and Themetric Approaches and Initial Hypothesis for Application to the Literature of Robert E. Howard.* **The Cross Plainsman: A Journal of Robert E. Howard Studies**, A Member Journal of REMupa. April 2002.

[14] Collins, Michael John. *A New Statistical Parser Based on Bigram Lexical Dependencies.*

[15] Copestake, Ann and Dan Flickinger. *An Open Source Grammar Development Environment and Broad-Coverage English Grammar Using HPSG.*

[16] Diederich, Joachim, Jörg Kindermann, Edda Leopold, and Gerhard Paass. *Authorship Attribution with Support Vector Machines.*

[17] Ellegard, Alver. *A Statistical Method for Determining Authorship: the Junius Letters 1769-1772.* **Gothenburg Studies in English** No. 13, Acta Universitatis Gothenburgensis, Elanders Boktryckeri Aktiebolag, Goteborg.

[18] Farrington, J.M. *Analysing For Authorship: A Guide to the Cusum Technique.* Cardiff: University of Wales Press 1996.

[19] Gazdar, Gerald and Chris Mellish. **Natural Language Processing in Lisp.**

[20] Kenny, Anthony. **The Computation of Style: An Introduction to Statistics for Students of Literature and Humanities**. Pergamum P, Oxford, 1982.

[21] Khmelev, Dmitri V. and Fiona J. Tweedie. *Using Markov Chains for Identification of Writers.* **Literary and Linguistic Computing**, 2001, vol.16, no.4, pp.299-307.

[22] Kjell, Bradley and Ophir Frieder. *Visualization of Literary Style.* **IEEE International Conference on Systems, Man, and Cybernetics**, October, 1992.

[23] Kohlhase, Michael and Alexander Koller. *Resource-Adaptive Model Generation as a Performance Model.* **Language and Computation**, Hermes Science Publishing Ltd, Vol 0 No. 0, pp. 1-26.

[24] Koster, Raph. *The Elements of Style.* **The Economist**, February 7, 2002.

[25] Landauer, Thomas K., Peter W. Foltz, and Darrell Laham. *An Introduction to Latent Semantic Analysis.*

[26] Manning, Christopher D. and Hinrich Schütze. **Foundations of Statistical Natural Language Processing**, The MIT Press, Cambridge, MA, 2001.

[27] Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. *Building a Large Annotated Corpus of English: The Penn Treebank.* **Computational Linguistics**, 19:313-330, 1993.

[28] Mendenhall, T.C. *The Characteristic Curves of Composition.* **Science**, IX, 1887, pp 237-249.

[29] Mendenhall, T.C. *A Mechanical Solution to a Literary Problem.* **Popular Science Monthly**, Volume 60, Number 2, 1901, pp 97-105.

[30] Morton, A.Q. *The Authorship of Greek Prose.* **Journal of the Royal Statistical Society (A)**, Vol. 128, 1965, pp 169-233.

[31] Mosteller, Frederick and David L. Wallace. **Inference and Disputed Authorship: The Federalist**. Addison-Wesley, Reading, MA, 1964.

[32] Mosteller, Frederick and David L. Wallace. **Applied Bayesian and Classical Inference: The Case of the Federalist Papers**. Springer Series in Statistics, Springer-Verlag, 1984.

[33] Neumann, Kenneth J. *The Authenticity of the Pauline Epistles in the Light of Stylostatistical Analysis.* **SBLDS**, no 120, Atlanta Scholars Press, 1990.

[34] Papadimitriou, Christos H., Prabhakar Raghavan, Hisao Tamaki, and Santosh Vempala. *Latent Semantic Indexing: A Probablistic Analysis.*

[35] Ram, Ashwin and Kenneth Moorman. *Towards a theory of reading and understanding.* to appear in **Understanding Language Understanding: Computational Models of Reading**, MIT Press.

[36] Ramshaw, Lance A. and Mitchell P. Marcus. *Exploring the Statistical Derivation of Transformational Rule Sequences for Part-of-Speech Tagging.* June 3, 1994.

[37] Rao, Josyula R. and Pankaj Rohatgi. *Can Pseudonymity Really Guarantee Privacy?* **Proceedings of the 9th USENIX Security Symposium**, August 2000.

[38] Robinson, W.A. CFE/CMG (1997).

[39] Rudman, Joe, David I Holmes, Fiona J. Tweedie, and R. Harald Baayen. *The State of Authorship Attribution Studies: (1) The History and the Scope; (2) The Problems – Towards Credibility and Validity.* **Joint International Conference of the Association for Computers and the Humanities and the Association for Literary and Linguistic Computing**, June, 1997.

[40] Shastri, Lokendra. *A Computational Model of Tractable Reasoning - taking inspiration from cognition.*

[41] Sichel, H.S. *On a Distribution Law for Word Frequencies.* **Journal of the American Statistical Association**, 70, 542-547, 1975.

[42] Sleator, Daniel D. and Davy Temperley. *Parsing English with a Link Grammar.*

[43] Soboroff, Ian M., Charles K. Nicholas, James M. Kukla, and David S. Ebert. *Visualizing Document Authorship Using N-grams and Latent Semantic Indexing.* **CIKM 97 Workshop on New Paradigms in Information Visualization and Manipulation**, November 13-14, 1997, ACM Press.

[44] Tweedie, Fiona J. and R. Harald. Baayen. *How variable may a constant be? Measure of lexical richness in perspective.* **Computers and the Humanities**, 32 (1998) 323-352.

[45] Vel, Olivier de, Malcolm Corney, Alison Anderson, and George Mohay. *Language and Gender Author Cohort Analysis of E-mail for Computer Forensics.*

[46] Williams, C.B. *Mendenhall's Studies of Word-Length Distribution in the Works of Shakespeare and Bacon.* **Biometrika**, Vol. 62, pp 207-212, 1975.

[47] Yu, Clara, John Cuadrado, Maciej Ceglowski, and J. Scott Payne. *Patterns in Unstructured Data: Discovery, Aggregation, and Visualization.* A Presentation to the Andrew W. Mellon Foundation National Institute for Technology and Liberal Education (NITLE)

[48] Yule, G.U. *On Sentence Length as a Statistical Characteristic of Style in Prose with Application to Two Cases of Disputed Authorship.* **Biometrika**, 30, 363-390, 1938.

[49] Zelle, John M. and Mooney, Raymond J. *Learning Semantic Grammars with Constructive Inductive Logic Programming.*