Authorship Attribution and "The Earliest Program for a System of German Idealism"

George Saussy and **Sean Sullivan**
Yale University

Abstract

The authorship remains controversial for *The Earliest Program for a System of German Idealism*, an essay found written in Georg Hegel's handwriting. With the rise of machine learning techniques and more sophisticated statistical models for author attribution, we seek to apply multiple models to identify the author. We specifically use a Naïve Bayes classifer, a perceptron model, and a support vector machine to attempt to tackle the problem. Although the results do not provide a confident solution, we suggest that Friedrich Schelling is the most probable author, and suggest paths for further research.

1 Introduction

The Earliest Program for a System of German Idealism is a fragment found among the notes of Georg Hegel in 1796. Though the original copy was written in Hegel's handwriting, it is unknown as of yet whether Hegel himself was the first author of the text, or if it was a transcription. The literature has primarily targeted Friedrich Schelling and Friedrich Hölderlin as possible alternative authors: the two, along with Hegel, were known as the "Tübingen Three" during their enrollment studying at the Tübingen Stift. Under two pages in length, the article presents an ethic for the future of idealism, sharing many common elements with the ideas that each possible author would later document in future writings. The trouble lies in identifying who the author actually was.

1.1 Background

The problem of author identification is not specific to only this one instance. In fact, the problem appears naturally when analyzing texts such as poems possibly written by Shakespeare, messages from cyber-criminals, and parts of the Bible. Historically, the problem has been handled by human experts on the potential authors comparing the new text with an author's known body of work. While this method can be effective, there is often too much data for humans to handle quickly. This strongly depends on a researchers understanding of literary techniques and close reading. Instead, computational approaches from natrual language processing apply mathematical models to this problem.

There are two main classes of computational approaches to this. The first class considers statistical models, such as the Naïve Bayes classifier. Classifiers such as the Naïve Bayes consider the text as a "bag of words," making statistical judgments based on word counts and evaluating between authors.

The second class of computational approaches to the document classification problem considers machine learning techniques ("ML"). ML has grown in popularity with a number of successful projects appearing in fields such as facial recognition, document compression, and most recently the game Go. Much of machine learning is part of the ongoing project to build thinking machines. Today, researchers of ML focus on software and learning algorithms. What has been found to be effective is to try to replicate the human brain. This is done by emulating the architecture of the brain, where computational nodes are connected in ways similar to neu-

rons in the brain. Modifications to this basic design make up the majority of ML.

For text classification, these methods work by building classifiers that abstract properties from a body of text to represent an author's style. These abstractions are then applied to the target text to determine authorship. The most common ML method is the "feed forward neural network". This method is particularly advantageous since it can be implemented in the same form to a wide variety of problems. The essence of the method is to build a virtual brain and teach it the style of the author. A modification to this model is radial basis function networks, where the construction of the virtual brain is modified, so that one may take advantage of aspects intrinsic to the data. These, in contrast with a vanilla neural network, are not bottled and reused.

Although these computational methods are different, they are similar in a few key features. Primarily, they all depend on the assumption that across an author's body of work, there are unconscious patterns that act as a footprint. They all work by, in some sense, comparing an author's body of work to a text to be classified. As such, each classifier's performance varies with how closely the corpus relates to the genre and period from which the target text is from.

The state of the art in authorship attribution can be found in a machine learning technique known as the support vector machine (SVM). SVMs were developed to handle problems that other data processing algorithms failed to model correctly or quickly. The technique specifically differs from the neural network with the idea that the iterative learning process used in regression models could improve the method. The difference allows the machine to tease out the properties needed to optimize its classifications. This results in a form of certainty: if there exists a classifier that can guarantee that the source of data can be classified, then an SVM will converge on it.

1.2 Relevance

SVMs came into their own when applied to The Federalist Papers. Twelve of the Papers' authorship was unknown, although the literature had all but decided that they were written by either James Madison or Alexander Hamilton. G. Fung was able to prove that Madison was the author of all 12 using an SVM trained on the functional words used in the paper.

For the world of German literary history, the *System* holds a similar level of importance. The text was written while all three of the prospective authors were at university together, and the three eventually became some of the most important figures in late German Romanticism in literature and philosophy. The essay preempted much of their work, and so knowing its author would inform a great deal about the context of the period.

1.3 Approach

The Naïve Bayes classifier, the Perceptron learning model, and the Support Vector Machine approaches were each used to determine the true author of the text. The results were created independently and measured against each other for success.

1.3.1 Naïve Bayes

The Naïve Bayes algorithm is simple. First, apply whatever tokenization features desired to the corpus. Count all the tokens, and for each author, keep a register of how many times they use each token. With this dictionary, take any test document and apply Bayes' Rule to determine the likelihood of any particular author given the document. The greatest likelihood author is the predicted answer. The accuracy can be tested by allowing the model to make predictions on known texts not included in the training set. Upon determining optimal parameters by repeated training and testing, the model is tested on the unknown datum.

1.3.2 Perceptron

The perceptron model takes some portion of the corpus as training data (as above), and performs a set of repeated operations. Define some output function $y = f(z)$ for an input vector z. Further, define each datum as an input vector of n dimensions. The dimensions describe *featues*, which can be designed as desired. Initialized weights w_i (also of dimension n). Then, for each example j in the training set, take the dot product of the current value of w for this iteration t with the input vector x_j. That is, let $z = x_j \cdot w(t)$, then $y = f(z)$. Compare $y_j(t)$ with d_j, and update the weights appropriately. Iterate until convergence to some error threshold or until some

number of iterations.

1.3.3 SVM

2 Evaluation

The implementation of these models is included in the submitted directory. In particular, the directory contains three subdirectories titled appropriately for each model.[1] The directory also contains any files or programs necessary to quickly replicate the results documented in this write-up.[2] The data files taken from the German Project Gutenberg are available in the "rawdata" subdirectory. Code was written in Python to perform the analyses for the Naïve Bayes and Perceptron classifiers. The SVM classifier uses LIBSVM, an open source implementation of an SVM, to perform the analysis. A supplemental Python script was written to reformat the data for ease of use.

The corpus of text to train the classifier was taken from the German Project Gutenberg hosted by Der Spiegel. This corpus provides texts by each of the three possible authors which were used for training and testing. It is worth noting that the data was not extensive enough to filter for properties like genre and time period, so the entirety of the available corpus was used for data.

The classifiers were tested by applying the completed models to the unknown text. Before testing on the novel text, parameters were adjusted to optimize the accuracy of the models. Each result was treated independently, and compared after the fact.

2.1 Naïve Bayes Model

The Naïve Bayes classifier is the most widely used statistical technique for this type of problem, often for its low computational cost. The classifier works quite simply: first, it records every word an author uses and how often each word is used; then, it compares those frequencies to that of a target text. Results between authors are then compared. If some document could belong to a series of authors, the Naïve Bayes classifier seeks to maximize the likelihood of a particular author being the creator of this document by instead maximizing the likelihood of this document "appearing" in the author's total possible works. It does so by appealing to a statistical equation known as Bayes' Rule, and by evaluating likelihood with frequency of usage. While it is easy to use, it does not always yield the best results. Importantly, this model makes very strong independence assumptions about individual words.

2.2 Perceptron Model

A Perceptron is an ML method. It was one of the early methods by which researchers attempted to replicate the human brain. It works in a similar way to the Naive Bayes, in that it involves word counts. It is different, however, in that it specifically uses the word counts to model properties of a typical document. Then, the "typical" data for each potential author is compared to the target text to determine authorship. While this method is in the same spirit as the Naive Bayes model, it is not the same—in fact, it can be much more computationally taxing. In fact, the run-time of such a model can be exponential.

2.3 SVM

The SVM, like a Perceptron and unlike a Neïve Bayes model, is from ML. However, it was mentioned earlier that even among ML techniques, the SVM is unique because of its partial basis in regression models. Normally, the aim with a regression model is to find how the average of a data set is sensitive to certain parameters of individual aspects each data point. With SVMs, the engineer allows the machine to amplify and weaken the quanta being optimized by the model. This provides faster run time as well as more optimization control for the engineer.

In contrast, a weakness of SVMs is that they are particularly sensitive to the training data. If the data is sparse (i.e. few texts or text on such a wide variety of topics or genres that the texts are not consistent), then the classifier will be adept at classifying only text similar to the input, but it will handle novel data poorly. Further, if the author has enough control over their style that they have no subconscious patterns, then no classifier can be guaranteed to work. There also remains the underlying risk with all ML techniques, which cannot tell the designer with what level of certainty they report their results.

[1] e.g., "./naivebayes/"

[2] See README for more details.

3 Implementation

First, a copy of the corpus was created and reformatted to be more easily input into the program. In each model subdirectory, the "[author]corpus" file refers to the original, and "[author]corpus2" refers to the reformatted copy.

All original code was written in Python and documented in the style of the Doxygen documentation generator.[3]

The Naïve Bayes model was constructed using two scripts which train and test the data. The model prints the log-likelihood that each author wrote the text, as well as the confusion matrix given by the model. For each author, the model was trained on 90% of the data and tested on the other 10%. To help smooth the data across anomalies, +1 smoothing was used in this model. Further, the implementation performed the analysis using two modes: first, with just the functional language words, and second, with the combined vocabulary of the authors instead of the just the functional words.

For the perceptron model, an "eta" parameter specifies the learning factor for the model and the "itrs" parameter specifies the number of iterations the the learning step should go through. The program prints the strength with which a perceptron trained to recognize each author believes the System Program was written by that author.

The support vector machine implementation uses a copy of the LIBSVM suite by Chih-Chung Chang and Chih-Jen Lin.[4]

4 Results & Analysis

4.1 Naïve Bayes

The Naïve Bayes analysis was performed once using only functional words and once using a combined dictionary. The confusion matrices in Fig. 1 shows that the combined dictionary performs significantly better (98% total accuracy) than the functional dictionary does (96% total accuracy).

After these results, the combined dictionary yielded that Schelling was the most likely author of the text.

Func

	Hegel	Hölderlin	Schelling
Hegel	34	1	2
Hölderlin	0	54	0
Schelling	1	0	6

All

	Hegel	Holderlin	Schelling
Hegel	37	2	0
Holderlin	0	54	0
Schelling	0	0	7

Figure 1: The confusion matrix given by the Naïve Bayes classifier run with the functional dictionary (above) and with the full dictionary (below).

4.2 Perceptron

With the perceptron model, different learning factors were tested for convergence and stability. After some experimentation, a trend was observed where the reported "true author" depended on the learning factor with a continuous trend, making it difficult to analyze. Fig. 2 shows accuracy vs. iterations plots for two different values of η. Notice that they produce vastly different results.

4.3 SVM

When run on the System, the SVM reported that Hölderlin wrote the text. This is, notably, different from the Naïve Bayes classifier's result. Upon further examination, when the training data is split and tested, the model is only 55% accurate. Reading the classifier output of the test data, one sees the classifier had classified every text as being from Hölderlin. This implies that the training data is not separable, and that because Hölderlin had more works in the training corpus, the classifier was optimized when it guessed every time that a new work was presented it was from Hölderlin.

4.4 Further Directions

From the results it is clear that the SVM classifier did not work. In order to correct for the separability problem it encountered, more advanced ma-

[3] The website and documentation can be found at http://www.stack.nl/∼dimitri/doxygen/.

[4] Documentation for the software can be found at csie.ntu.edu.tw/ cjlin/libsvm/ and the GitHub repo for the software can be found at github.com/cjlin1/libsvm.

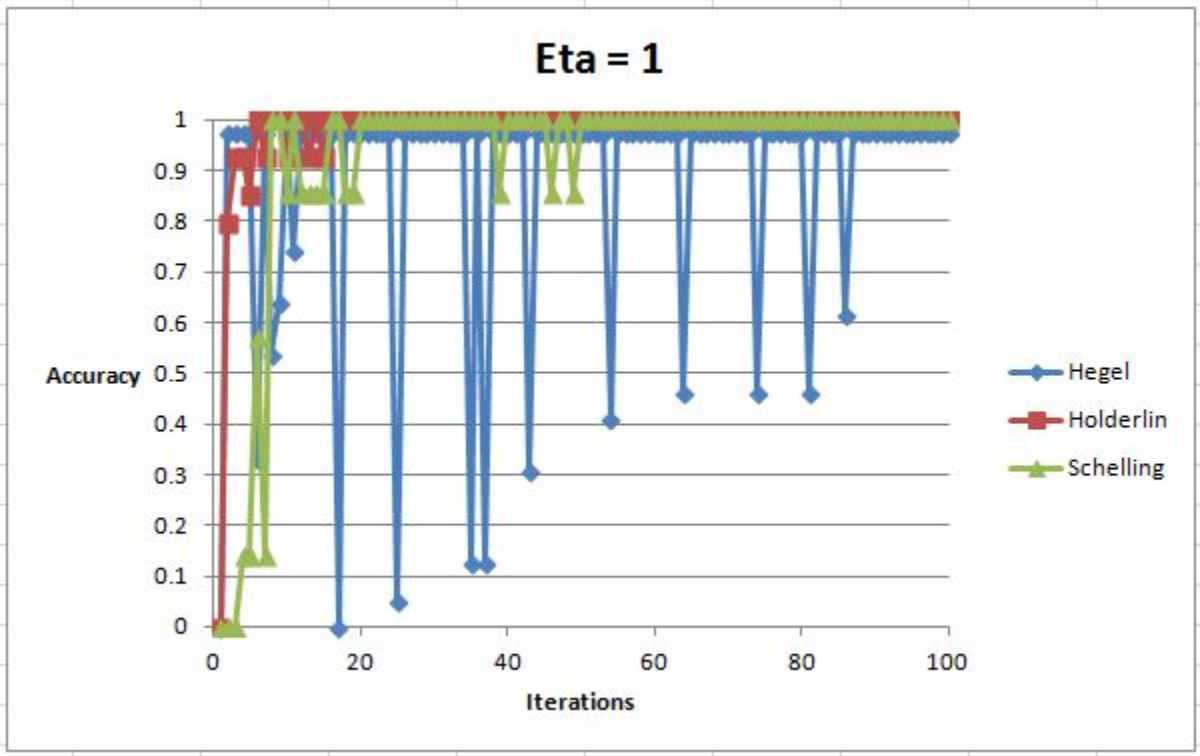

Figure 2: Perceptron model with $\eta = 0.05$

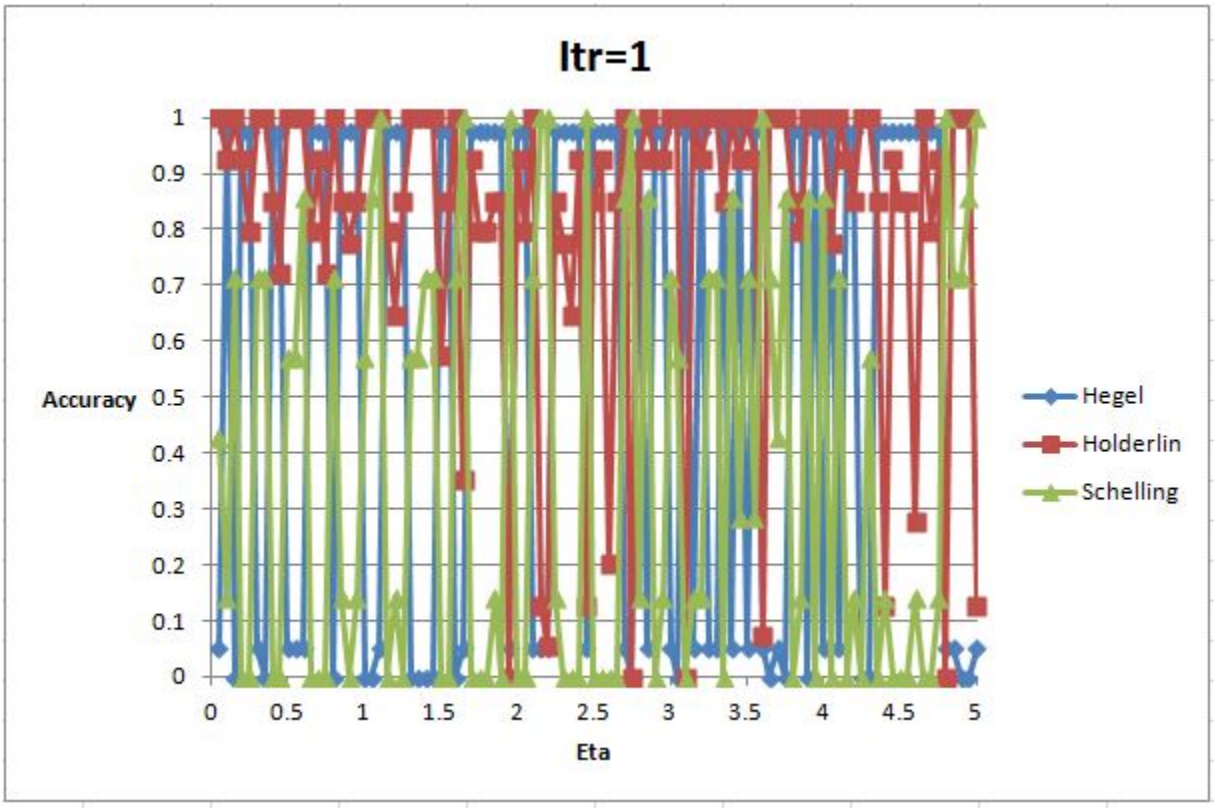

Figure 3: Perceptron model with $\eta = 1$

chine learning models might be applied. However, with only about 60 works from Schelling, it is as of yet unclear to what extent a multilayer classifier will improve over an SVM; building the model with a more complicated model may run into issues with data sparsity.

For this reason, it seems that the most apt candidate for this type of classification problem would be a neural network. This would provide another machine learning technique to analyze the data without worrying about separability.

It would also be worth trying to apply more specific parameters to the Naïve Bayes model, to see if we can better distinguish between the authors for each class. This is, however, unrealistic without more data.

It is also worth noting that authorship attribution models are still being developed. Even the techniques used here can be improved in certain ways—tokenizing and stemming the text could easily result in more reliable Naïve Bayes results, and even methods of counting co-occurrences of words to improve accuracy.

5 Discussion

The first thing to notice is the failure of the SVM. Unfortunately, basically nothing can be drawn from this model, except that the data is inseparable.

It was found that the perceptron model was extremely sensitive to the learning parameter. This is consistent with the results from the SVM and indicate that the data in the training corpus is not separable. Unfortunately, this means that no inferences can be effectively made here either.

The results of the Naïve Bayes model, then, is the best way to make predictions based on this data, which states that Schelling is the most likely author. However, this result is relatively unreliable since the other models gave wildly different results. Further, considering the fact that the accuracy increased when using the combined dictionary rather than the functional dictionary, it is possible that key words may very greatly from work to work. Without a sufficiently large sample, it is possible to severely bias a classifier, which could have resulted in the poor performance of the functional dictionary.

5.1 Past Work

This experiment is one of many instances of the author attribution that has been increasingly in demand in the modern world. Models of perceptron learning and support vector machines date back to the 1950s and 1960s, but with the rise of computational power and the increase in available information through the internet in the 1990s, applications of these techniques to authorship analysis saw its birth.[5] This project seeks to continue this work.

5.2 Assumptions

It is worth noting some key assumptions made by each model. In particular, the Naïve Bayes model relies specifically on an independence assumption; that is, that texts can be treated as a "bag of words." This is not particularly viable, since semantic meaning and syntactic structure often plays a large role in determining what words are chosen in a particular context. However, since we lack the tools to deftly encode that meaning, and since the model tends to work relatively well, this assumption is reasonable.

Acknowledgments

We would like to thank Professor Kirk Wetter for his guidance on the literary aspects of this work.

References

Stamatatos, E. A Survey of Modern Authorship Attribution Methods. University of the Aegean. http://www.icsd.aegean.gr/lecturers/stamatatos/papers/survey.pdf

Doxygen document builder, http://www.stack.nl/∼dimitri/doxygen/

LIBSVM suite taken from Chih-Chung Chang and Chih-Jen Lin, found at csie.ntu.edu.tw/ cjlin/libsvm/

[5] Stamatatos, E. A Survey of Modern Authorship Attribution Methods. University of the Aegean. http://www.icsd.aegean.gr/lecturers/stamatatos/papers/survey.pdf