

Authorship Analysis in the Blogosphere

Robert Elwell and Manish Katyal

University of Texas at Austin
1 University Station B5100
Austin, TX 78712-0198 USA

Abstract

Authorship identification uses statistical machine learning methods to classify documents by author. In this paper, we examine the strength of established approaches in the related task of identifying authorship of posts made on established weblogs. We also experiment with feature engineering using off-the-shelf linguistic parsing tools to examine what value incorporating these approaches may offer in improving performance.

Machine learning methods for authorship analysis have been shown to be both valid and effective tools in a task now known as “writeprinting” a text document. Traditionally, authorship identification has been performed to identify individuals behind popular literary works. With the popularity of the Internet and the explosive growth in web content, authorship identification is now being used for Internet web content forensic analysis.

In this paper, we bring authorship identification analysis to a new arena of internet media—weblogs, also known as blogs. This type of content offers its own set of interesting challenges in comparison to literary works analysis. Blogs consists of short blog posts and are akin to a web-based journal. They are written in an casual manner with little structure. Given the informal nature, blog posts are quite noisy as they contain grammatical and spelling errors.

Developing methods to identify the provenance of a blog post is valuable for many reasons. It can be leveraged for the purposes of tracking popularity of blog content via text quotations on other websites, for the purposes of tracking plagiarism, or even

possibly for the purposes of associating abusive or threatening messages with a single organization or individual.

We focus our analysis on blog posts from six political blogs. We pose authorship identification as a machine learning binary classification problem. Given two blog posts, our system can be used to determine if they were written by the same author. This can be easily extended to identify the actual author of the post. We use a combination of statistical text mining techniques and linguistic analysis techniques to build the features for the blog posts. Linguistic analysis is performed using an off-the-shelf parser. In our approach, we aim to select features that capture the style of writing of the authors as opposed to features that model the topic or subject. For this reason, we restricted our analysis to blogs in the same subject area. Given the current political atmosphere, we believe that focusing specifically on political blogs will show how truly effective any method which performs well will be for any of these uses.

In this paper, we evaluate our data approaches and models not only in the traditional terms of accuracy, but in time training and time to generate the data as well. The optimal approach maximizes accuracy relative to the time taken to effectively model the problem. This is especially important with regards to the need for a versatile yet tractible solution to ascertain identity given an author’s writing, as such a goal has various related applications with their own specific needs and their own specific hypothesis space to consider.

1 Related Work

Authorship analysis has a rich background both in terms of modern research and historical underpinnings. It has been used to great success in determining the stylistic similarity of historically unknown authors and identifying acts of plagiarism, and is currently being scaled to use for internet messages. Stamatatos et al. (2000) outlines work in Greek using only text data acquired from World Wide Web. These employ multiple regression models trained on lexical, stylistic, and syntactic features. However, these messages are still restricted to fairly formalized arenas of writing—mainly planned and edited texts such as official documents, reportage, and academic literature. We believe their work leaves further room for experimentation with linguistic features as they only used one linguistic feature derived from deeper structural dependencies - syntactic chunks.

Zheng et al. (2006) offers both an extensive history of work in authorship analysis, while at the same time experimenting on a data set consisting of more user-generated content. Here, messages are extracted from newsgroups with the purpose of attributing acts of “cybercrime” to a specific author. They propose a detailed taxonomy and a large feature set which is heavily focused frequency statistics and shallow syntax, such as frequency of use of punctuation and function words. However, because of a specific focus on software piracy, a set of keyword features which select for area-specific posts calls the empirical results of the paper into question. Could these results be duplicated without explicit guidance in terms of clever feature selection? Furthermore, would the specific use of these keywords as features create problems for identifying an author writing in an unrelated subject, using a different set of keywords? This is especially important to consider given the fact that “cyber-criminals” often use separate internet aliases for legitimate and illegal activity online.

Li et al. (2006) continue in the area of cybercrime, using data from Google Newsgroups and the Chinese Bulletin Board Systems for the task of classifying a document by authorship, using an author label. They utilize a genetic algorithm on the feature set which optimizes for the maximum degree of dis-

crimination between authors in the training set.

2 Data

The corpus in use consists of a set of politically oriented blogs. They are evenly divided between liberal and conservative blogs. There are 246 conservative blog posts and 267 liberal posts. The blog titles and their respective counts are shown in figure 1. The corpus is divided into 90% training data and 10% testing.

Previous approaches treat authorship identification as a multi-class classification problem. Given a document of unknown origin they aim to predict the author. In the case of blog post authorship identification this implies predicting the blog title. There are several approaches to perform multi-class classification: one-versus all, pair-wise classification etc. We chose instead to approach it as a binary classification problem. Given two documents, we predict if the documents were written by the same author or not. This can be easily extended to learn the identity of the author.

We chose the binary classification approach because it can be used in cases where some of the documents in the testing set are from authors that have no documents in the training set. As long as the training set has samples from one of the authors in the pair, our approach should work (this has not been tested empirically). Through some early empirical analysis, we found the binary classification to work far better than one-versus all multi-class classification. Additionally, blogs can be written by more than one author and an author may either write or be cited in one or more blogs. Therefore, we believe predicting the blog title given a blog post is not the right approach. Additionally, our approach greatly increases the number of examples for training. Given 510 blog posts, we generated 43,736 examples - 39,362 for training and 4374 for testing. The training and testing sets contain an equal number of positive and negative examples.

The disadvantage of our approach is that it doubles the size of the feature space. Some algorithms such as decision trees can perform poorly in terms of training time and accuracy if the size of the feature space is greatly increased. However we believe this is not a major issue as the size of our feature space

Blog	No. of Posts	Total Words	Average Length in Words	Orientation
South by Southwest	95	519083	5464	Liberal
The Democracy Daily	80	477583	5970	Liberal
Democracy Cell Project	89	408277	5697	Liberal
Wake Up America	92	601513	6538	Conservative
Dr. Sanity	60	434381	7240	Conservative
California Conservative	94	412285	4386	Conservative

Figure 1: Blog statistics

(359 features per blog posts) is relatively small compared to the thousands or even tens of thousands of features that are common in data sets used in text mining.

3 Algorithms

For classification, the algorithms we employ are Naive Bayes, Ada Boost in conjunction with Naive Bayes and Logistic Regression. These are components of the WEKA machine learning toolkit implemented in Java (Witten et al. (1999)). Early empirical testing showed various methods for incorporating support vector machines (SMO) to be too time-consuming in terms of tweaking the many parameters associated with training the model and provided poor results. Experiments with other algorithms in WEKA such as J48 and Simple Logistic Regression ran too long and were thus unusable.

Naive Bayes is a simple probabilistic classifier which creates a generative model of the learning problem by using as its prior probabilities the raw frequencies features within the corpus. Because of this, it retains a naive assumption of featural independence. In structurally sensitive concepts, this can be a problem for the model, resulting in lower accuracy than its more discriminative counterparts in such areas. It is known for its fast training and testing time over large corpora. Simple smoothing approaches have provided a practical solution to issues of sparsity within a high-dimensional feature matrix. Considering this, the model does not overfit the training data, and can handle noise easily. (John and Langley)

Logistic regression is a discriminative Bayesian approach which builds a multinomial model with a ridge estimator. This is also known as a maximum entropy or model because it has been shown to

consistently arrive at a probability distribution with the greatest amount of entropy that is also consistent with the data. This is valuable, because as the least amount of entropy within the probability distribution, the least assumptions about the data are being made by the model. The model is discriminative rather than generative because it models the conditional distribution of $P(Y|X)$ where Y is a class and X is a feature vector and then uses that model to fit the training data, reweighting probabilities to better predict Y from X . A disadvantage to this approach is the increased training time needed in order to fit the training data better. However, it is consistently more accurate than Naive Bayes and AdaBoosted Naive Bayes. While the approach is parametric, it only requires the fine tuning of one parameter, the Gaussian prior, as opposed to the multiple parameters used in SVM training. (Ie Cessie and van Houwelingen (1992)).

AdaBoostM1 is an ensemble learning technique which we use to improve the performance of the base learners described above. It can take a weak learner such as Naive Bayes and create multiple models from it, creating diversity through weighting the decisions the learner gives to better fit the data. While not theoretically explained, it has been empirically tested to improve performance, including preliminary investigations into our data set. The weakness of boosting is the increase in training time due to ensembling the weighted learners over and over again. However, it does offer a valuable increase in accuracy with the use of learners which require little training time. (Freund and Schapire (1996))

AdaBoostM1 could also be used in conjunction with Logistic Regression. However, this was not performed as a single iteration of Logistic Regression took a very long time and using it in conjunc-

tion with AdaBoostM1 that would perform several iterations would take too long.

4 Features

Features extracted from the blog posts are divided into two types: Stylistic and Linguistic. These were divided for the purpose of comparing their value in terms of framing the problem in a learnable way and their relative cost in terms of time to create. Figure 2 lists all features, divided by set.

Feature set one is referred to as the Stylistic feature set. It consists of basic statistical features extracted from the text and is a set of features that are traditionally used in authorship identification. These features are gained through calculating frequency statistics and performing stylometric measurements on the text data with minimum of deep-structural insight. As a result, they can be acquired quite easily and do not require complex linguistic parsing techniques.

This feature set consists of the frequency of function words in the text. The list of words used can be found in Appendix A. Also in the set is Vocabulary richness. Vocabulary richness was determined by dividing the number of tokens in the blog post by the number of tokens in the blog post. The other features in this set are Stylometric features that are statistical frequency-based features derived from the surface features of the text. These features are mainly frequency statistics over specific characters. Those referring to punctuation should play an important role in capturing an author's specific writing style, if the author is consistent in their writing. Frequency of alphanumeric characters may be valuable in capturing certain trends towards capitalization. Frequency of short words normalized over all words has a value in the relative verbosity or brevity of a particular author. Word-based stylometric features would have complexity of $O(W)$, with coding time slightly larger than the extremely low coding time necessary for character-based features, which have a complexity of $O(C)$, where C is all of the characters in the corpus. For each feature here, however, there is a polynomial increase, while function word frequencies can be calculated with only one pass through the data.

Through empirical investigation, we found the

traditional Information retrieval approach of tf-idf normalization of the function words to perform better than raw frequency counts. This is because the tf-idf approach helps weighting words in terms of their importance to a specific blog post with regards to an entire corpus and should thus help identify function words that are typical of each blog author.

Feature set two is referred to as the Linguistics feature set and comprised of statistical features extracted from the syntax and semantics of the post. These features were derived from structures built using the off-the-shelf Stanford Parser¹. This accounts for part-of-speech tagging, parsing, and dependency parsing. From this data, statistics on verb and noun phrase frequency is collected, as well as statistics on tree depth. These features should be valuable in suggesting how an author employs syntax stylistically. For instance, perhaps an author has a tendency to embed a great deal in a single sentence; this would be accounted for by the tree depth metric. Other linguistic features include a passive metric, which is the number of passive sentences divided by the number of total sentences in a document, which could be valuable in identifying the writing style of an author in terms of how often they passivize. We also focus on discourse-level semantics, such as the frequency of use of adverbial, subordinating, and coordinating connectives to investigate whether such features may be author-specific. We also calculate the number of attributive verbs, in case certain authors are more quotative than others. Using a well-known all-in-one parser will also test the capability of the general state of the art in computational linguistics to offer improvement in the area of authorship analysis, assuming that features which abstract these structures are properly represented in the learning task. At the same time, in comparison to the Stylistic features in Feature set one, these linguistic features require sophisticated and complex parsing algorithms. Additionally, these are computationally intensive to extract and took longer to acquire in comparison to feature set one.

Through experimentation, we found the algorithms to perform faster and provide better accuracy with nominal features rather than numeric features. As a result, the features were discretized using a

¹<http://nlp.stanford.edu/downloads/lex-parser.shtml>

Feature No.	Description	Set
1	Vocabulary Richness	1
2-322	Function Words	1
323	Hapax Legomena Count	1
324	Hapax Dislegomena Count	1
325	No. Characters	1
326	Average Word Length	1
327	Average Sentence Length (words)	1
328	No. Sentences	1
329	No. Alphanumeric Characters	1
330	No. Upper Case Characters	1
331	No. Numeric Characters	1
332	No. White Space Characters	1
333	No. Tab Characters	1
334	No. Words	1
335	No. Short Words	1
336	Total No. Characters in Words	1
337	Average Sentence Length (characters)	1
338	No. Word Types	1
339-347	Punctuation Character Frequencies	1
348	Longest NP in Words	2
349	Avg. No. Words per NP	2
350	Longest VP in Words	2
351	Avg. No. Words per VP	2
352	Deepest Tree in Branches	2
353	Avg. Tree Depth in Branches	2
354	Passive Metric	2
355	No. Discourse Connectives	2
356	No. Attributive Verbs	2
357	No. Coordinating Connectives	2
358	No. Subordinating Connectives	2
359	No. Adverbial Connectives	2

Figure 2: Feature Set Per Document

function built within WEKA. This function created equal-width discrete bins for each feature using a leave-one-out method.

It should be noted that there are no specific keywords whose frequency we incorporate in the feature vector, but rather an established set of function words. This is important because an author may change topic, and the use of topical keywords could distract the learner from other features which may better describe the author as a writer. Ultimately, we want to be able to capture the author's writing form rather than content.

5 Results and Discussion

The objective of our experiments was to enable the comparison of classification algorithms and the comparison of the feature sets - Stylistic and Linguistic. The comparison was performed on the basis of accuracy, time taken for training and testing.

Figure 3 compares Naive Bayes, Naive Bayes boosted with AdaBoost and Logistic Regression (MaxEnt).

Logistic Regression, or MaxEnt, has a statistically significant improvement in accuracy over both Naive Bayes-based models. However, its training time is also significantly larger than boosted Naive Bayes. With training time being roughly 70 times that of boosted NB, and a delta of only 4 percent in accuracy, the practicality of such a model is called into question. Because of its high training time, it was not feasible to acquire learning curve results or boosting results for this particular model. Many Linear Regression implementations retrain to weight features until the model converges with the correct set of class labels for all instances. This feature weighting process can be especially time-consuming as the number of features increase. In the future, setting a limit on the number of iterations during the training stage could reduce training time, but probably at a significant cost to accuracy.

Naive Bayes boosted with Ada Boost performed better than Naive Bayes in terms of accuracy but took longer to train. This is because it builds an ensemble of Naive Bayes classifiers. Its training time was however far shorter than Logistic Regression. From the testing time perspective we found AdaBoosted Naive Bayes took the longest to clas-

sify a test instance. This is because a test instance has to be evaluated against an ensemble of models.

Naive Bayes took the shortest training and testing times. This is because it has a very simple model because of the assumption it makes regarding the independence of the features. Its accuracy was far lower than Logistic Regression and lower than AdaBoosted Naive Bayes. This is because it sacrifices accuracy for simplicity and speed.

For comparing the two types of features, given the prohibitive training time for Logistic Regression, only the Naive Bayes and the AdaBoosted Naive Bayes algorithms were used. Accuracy results can be seen at different points of the learning curve in Figure 4 and Figure 5 for Naive Bayes and its boosted counterpart (an asterisk signifies statistical significance).

The learning curve in Figure 4 compares the performance of NaiveBayes and AdaBoosted Naive Bayes on three datasets - Stylistic features, Linguistic features and a combination of both referred to as combined features. AdaBoost with 10 iterations is used. The learning curves are used to judge the efficacy of the features - that is to compare the Stylistic features to the Linguistic features.

AdaBoost-Naive Bayes has higher accuracy than Naive Bayes on all three of the datasets. The best accuracy is on the Combined dataset followed by the Stylistic dataset. At 100 percent of the data, the accuracy of the Combined dataset is only marginally higher than that of the Stylistic dataset. At 60 percent of the data, the accuracy of the Stylistic dataset is marginally higher than the Combined dataset. In comparison to the other datasets, the accuracy of both the classifiers on the Linguistics dataset is quite low. On this dataset, AdaBoost-NaiveBayes again performs marginally better than Naive Bayes. From this we can conclude the Linguistic features provide a very marginal benefit in classification as the performance of the classifiers without these features is almost as good as the classification with the Linguistic features included. The performance of the classifiers generally improves as the amount of training data is increased. This is as expected. The rate of improvement levels off at 40 percent for the classifiers on all the datasets and from then on adding more data results in small but significant improvements in performance.

	Naive Bayes	NB-Boosted	MaxEnt
Accuracy	71.6	73.9*	78*
Training Time	6.5	186.9*	13009.6*
Testing Time	1.1	9.2*	3.9*

Figure 3: Naive Bayes Models Compared to Logistic Regression (MaxEnt)

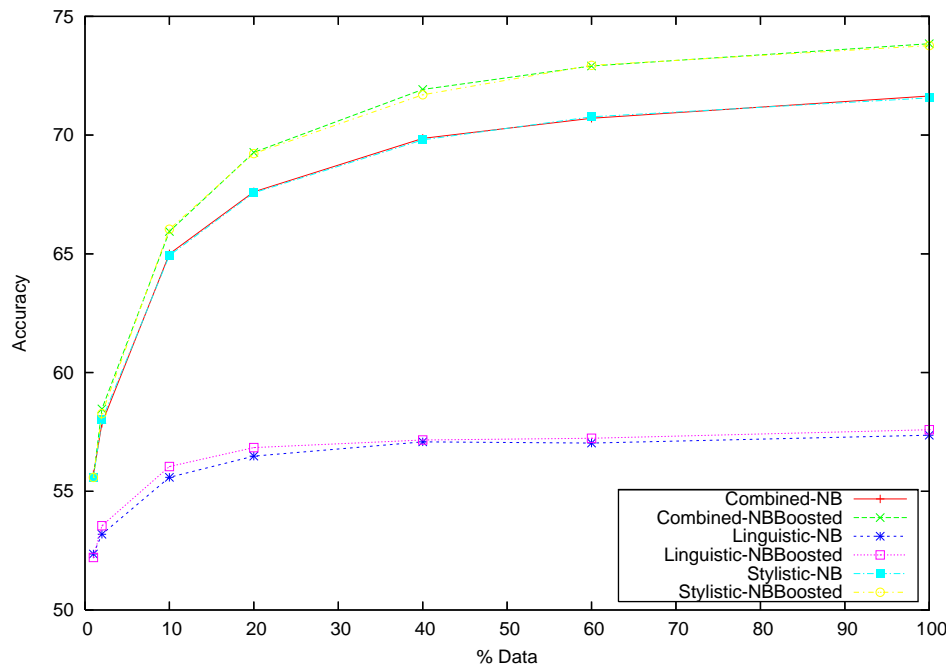


Figure 4: Learning Curves for Naive Bayes and Boosted Naive Bayes

% Data	Data Set	NB Accuracy	Boosted Accuracy
10	Sylistic	64.9	65.9*
	Linguistic	55.6	56*
	Combined	65	66*
40	Sylistic	69.9	71.7*
	Linguistic	57.1	57.1
	Combined	69.9	71.9*
100	Sylistic	71.6	73.8*
	Lingusitic	57.4	57.6
	Combined	71.7	73.9*

Figure 5: Accuracy Comparison Between Data Sets–Boosted vs. Non-Boosted

A comparison of training time for AdaBoosted Naive Bayes and its base learner counterpart can be seen in Figure 6. The training time for AdaBoost-Naive Bayes is significantly higher than that of Naive Bayes. As the amount of training data increases, the time taken for training increases for both classifiers on all of the datasets. The training time on the Linguistics dataset is significantly lower than the other datasets as the number of features is the smallest. The training time on the Combined dataset for both classifiers is higher than that on the Stylistic dataset—again, as the number of features in the Combined dataset is higher than that of the Stylistic dataset.

The testing time for AdaBoost-Naive Bayes is significantly higher than that of Naive Bayes—across all of the datasets. This is shown in Figure 7. AdaBoost builds an ensemble of models and thus the testing time is higher in comparison to using the single model Naive Bayes classifier. The testing time on the Combined dataset is the highest as it has the largest number of features following by the Stylistic dataset. The testing time on the Linguistic dataset is extremely low for both classifiers in comparison to the other datasets.

From the above results and how they relate, we can conclude several points with regards to the data sets and how they interact with the learning models. The stylistic features are preferable to the combined dataset as the training and testing times are lower than the combined dataset and the accuracy is almost the same. The linguistic features are expensive to build in comparison to stylistic features which can be very efficiently extracted. The linguistic features provide a very minor improvement in accuracy when added to the stylistic features. On their own, the linguistic features are fewer in number, and thus, it is easier to train a classifier or an ensemble thereof. However, the accuracy of these features alone is quite low. This may be due to a poor abstraction of features for the learner; perhaps these are not the right aspects of syntax and semantics to offer the learner. Or perhaps, more likely, the more structural side of syntax and semantics are not as telling aspects of a writer’s personal style than certain aspects of their vocabulary and use of punctuation. It may be possible that a writer leaves their “writeprint” on a text in the process of encoding lan-

guage to text, rather than their actual use of language. In any case, as it currently stands, the amount of pre-processing required to extract this data from a text in terms of coding and parsing—especially if parsing is done incrementally as opposed to all at once—does not justify the fairly meager increase in accuracy.

In terms of algorithm selection, the clear victor is AdaBoosted Naive Bayes, which runs significantly faster than the WEKA implementation for logistic regression. This difference in time to train and accuracy between the two models is interesting. The ensemble approach uses multiple models to improve accuracy on the training set. Logistic regression employs a single model with meticulously weighted features to fit the training set. This shows that weighted features alone do not fully comprise the difference between Naive Bayes and logistic regression, though the ensembling and weighting found in AdaBoost does decently approximate the discriminative model with the added boost of reduction in training time.

6 Future Work

A common shortcoming of our approach and that of others is the decline in performance as the number of authors is increased. We empirically found this to be a problem and thus our work and that of others will not scale to a large number of authors. Many real-world problems require the ability to deal with a large number of authors and thus this will be the focus of our future work.

Another area that the literature does not seem to address is unseen authors. This is only obliquely addressed in the sub-area of document similarity analysis, which has a stronger focus on plagiarism than grouping grouping documents by author. The optimum approach would have a better-than-chance performance on associating documents created by an unseen author together in a lucid manner. However, by associating the class label of an instance in the training data with only known authors, accuracy in this space will invariably be 0%. Furthermore, because of the idiosyncracies of each author, an unseen author is more than likely to be classified inconsistently for each text produced.

An extension of our approach could be used for

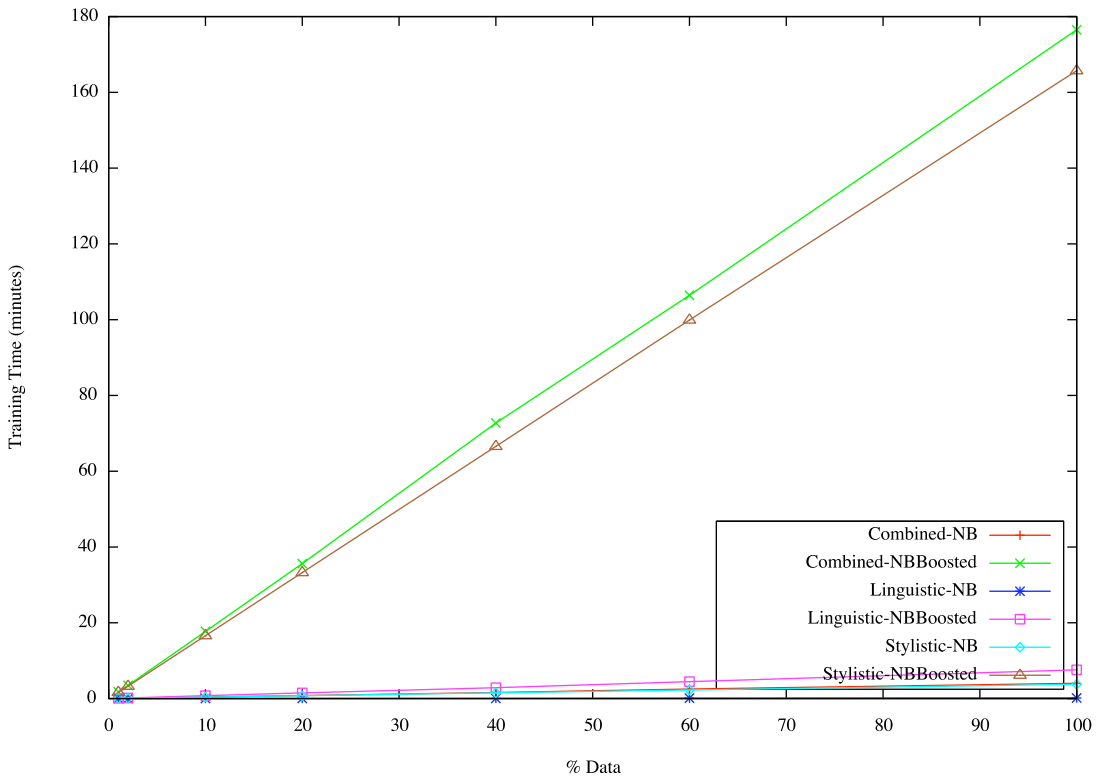


Figure 6: Training Time

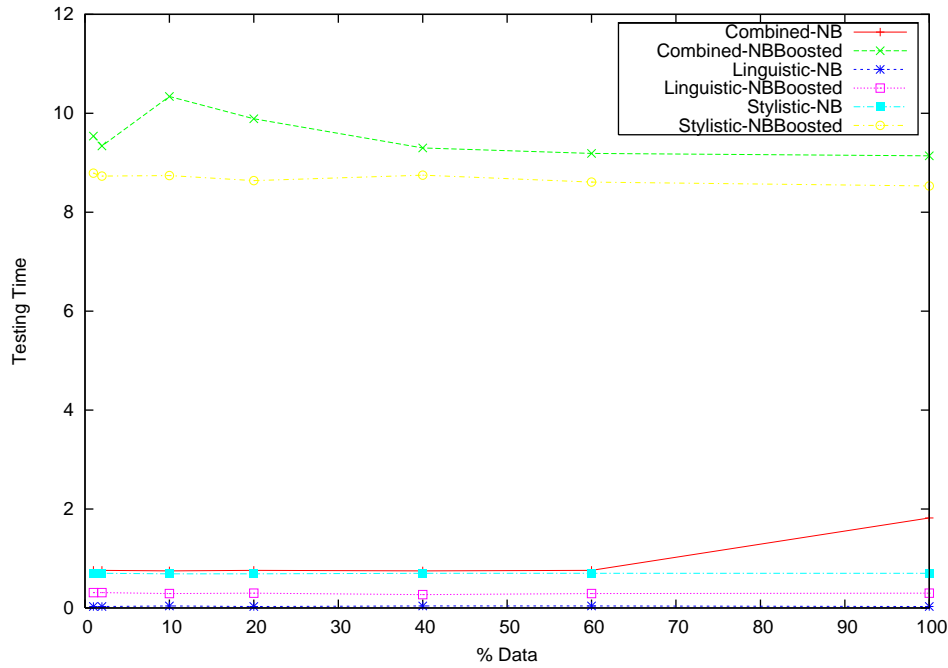


Figure 7: Testing Time

addressing the issues of unseen authors. Given the success of associating text instances together by a boolean authorship value, this raises the possibility for future work in creating chains or clusters of documents by an author. This approach would benefit from the parallels it could draw from work in coreference resolution, as seen in Denis and Baldridge (2007). Integer linear programming approaches could be used to globally select the most coherent chains given a set of probabilities drawn from a Bayesian learner and constraints of membership to a single chain only. Such work would more than likely improve F-score as calculated by the MUC metric (Vilain et al. (1995)) or the B-Cubed metric (Amit and Baldwin (1998)) against non-ILP-constrained chainings. Furthermore, this would offer a more global understanding of how the learner constructs the textual identity of a specific author.

We found the Logistic Regression implementation in WEKA to be very slow but accurate. An avenue for future work is to build an optimized version of this algorithm or to review other toolkits that have a better implementation.

7 Conclusion

In this paper, we have shown the value of Bayesian learning techniques for the purpose of authorship identification. Despite application to a large, sparse data set with the aims of solving a difficult problem, accuracy is significantly better than chance given fairly naive surface-level features. Furthermore, we offered an alternative abstraction of a familiar problem, retooled for future work on anonymous authors.

Because of the successfulness of Bayesian approaches over early experimentation with different learnings, we have evaluated generative versus discriminative approaches in this particular area. From this, we have arrived at the empirical preference for the generative naive bayes model, in terms of its high accuracy relative to training time. Logistic regression, on the other hand, offers a boost in accuracy which is too heavily mitigated by its large training time. In realistic applications, the scale of our current approach makes it most likely a toy problem. An increase in the number of authors, would most likely require an increase in the number of documents per author, and could also be costly in terms

of data generation.

We have also compared the value of different kinds of feature abstractions for the given task. Of the two sets we have evaluated, we found the Stylistic features to be of the most value. We found the linguistic data to be inadequate by itself for the purpose of authorship identification. Also, it was also of little value when combined with Stylistic features. Given the expense of acquiring these features and the lack of performance, we conclude that the Linguistic features we evaluated are not suitable for authorship identification.

With the work we have established here in the area of blog post authorship, there is much work left to do. Here, we offer an overview of some problems pervasive in authorship identification with applications to the internet, as well as confirmation from previous work on internet media of the value of certain models and approaches.

References

- B. Amit and B. Baldwin. Algorithms for scoring coreference chains, 1998. URL citeseer.ist.psu.edu/153897.html.
- Pascal Denis and Jason Baldridge. Global, joint determination of anaphoricity and coreference resolution using integer linear programming. In *Proceedings of HLT/NAACL-2007*, 2007.
- Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996. URL citeseer.ist.psu.edu/freund96experiments.htm.
- George H. John and Pat Langley. Estimating continuous distributions in Bayesian classifiers. URL citeseer.ist.psu.edu/john95estimating.html.
- S. le Cessie and J.C. van Houwelingen. Ridge estimators in logistic regression. *Applied Statistics*, 41(1):191–201, 1992.
- Jiexun Li, Rong Zheng, and Hsinchun Chen. From fingerprint to writeprint. *Commun. ACM*, 49(4):76–82, 2006. ISSN 0001-0782. doi: <http://doi.acm.org/10.1145/1121949.1121951>.
- Efstathios Stamatatos, George Kokkinakis, and Nikos Fakotakis. Automatic text categorization in terms of genre and author. *Comput. Linguist.*,

26(4):471–495, 2000. ISSN 0891-2017. doi:
<http://dx.doi.org/10.1162/089120100750105920>.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. A model-theoretic coreference scoring scheme. In *MUC6 '95: Proceedings of the 6th conference on Message understanding*, pages 45–52, Morristown, NJ, USA, 1995. Association for Computational Linguistics. ISBN 1-55860-402-2. doi:
<http://dx.doi.org.ezproxy.lib.utexas.edu/10.3115/1072399.1072405>.

I. Witten, E. Frank, L. Trigg, M. Hall, G. Holmes, and S. Cunningham. Weka: Practical machine learning tools and techniques with java implementations, 1999. URL
citeseer.ist.psu.edu/witten99weka.html.

Rong Zheng, Jiexun Li, Hsinchun Chen, and Zan Huang. A framework for authorship identification of online messages: Writing-style features and classification techniques. *J. Am. Soc. Inf. Sci. Technol.*, 57(3):378–393, 2006. ISSN 1532-2882. doi: <http://dx.doi.org/10.1002/asi.v57:3>.

8 Appendix A: List of Function Words

ABOUT	ABOVE	AFTER	AGAIN	AGO	ALL
ALMOST	ALONG	ALREADY	ALSO	ALTHOUGH	ALWAYS
AM	AMONG	AN	AND	ANOTHER	ANY
ANYBODY	ANYTHING	ANYWHERE	ARE	AREN'T	AROUND
AS	AT	BACK	ELSE	BE	BEEN
BEFORE	BEING	BELOW	BENEATH	BESIDE	BETWEEN
BEYOND	BILLION	BILLIONTH	BOTH	EACH	BUT
BY	CAN	CAN'T	COULD	COULDN'T	DID
DIDN'T	DO	DOES	DOESN'T	DOING	DONE
DON'T	DOWN	DURING	EIGHT	EIGHTEEN	EIGHTEENTH
EIGHTH	EIGHTIETH	EIGHTY	EITHER	ELEVEN	ELEVENTH
ENOUGH	EVEN	EVER	EVERY	EVERYBODY	EVERYONE
EVERYTHING	EVERYWHERE	EXCEPT	FAR	FEW	FEWER
FIFTEEN	FIFTEENTH	FIFTH	FIFTIETH	FIFTY	FIRST
FIVE	FOR	FORTIETH	FORTY	FOUR	FOURTEEN
FOURTEENTH	FOURTH	HUNDRED	FROM	GET	GETS
GETTING	GOT	HAD	HADN'T	HAS	HASN'T
HAVE	HAVEN'T	HAVING	HE	HE'D	HE'LL
HENCE	HER	HERE	HERS	HERSELF	HE'S
HIM	HIMSELF	HIS	HITHER	HOW	HOWEVER
NEAR	HUNDREDTH	I	I'D	IF	I'LL
I'M	IN	INTO	IS	I'VE	ISN'T
IT	ITS	IT'S	ITSELF	LAST	LESS
MANY	ME	MAY	MIGHT	MILLION	MILLIONTH
MINE	MORE	MOST	MUCH	MUST	MUSTN'T
MY	MYSELF	NEAR	NEARBY	NEARLY	NEITHER
NEVER	NEXT	NINE	NINETEEN	NINETEENTH	NINETIETH
NINETY	NINTH	NO	NOBODY	NONE	NOONE
NOTHING	NOR	NOT	NOW	NOWHERE	OF
OFF	OFTEN	ON	OR	ONCE	ONE
ONLY	OTHER	OTHERS	OUGHT	OUGHTN'T	OUR
OURS	OURSELVES	OUT	OVER	QUITE	RATHER
ROUND	SECOND	SEVEN	SEVENTEEN	SEVENTEENTH	SEVENTH
SEVENTIETH	SEVENTY	SHALL	SHAN'T	SHE'D	SHE
SHE'LL	SHE'S	SHOULD	SHOULDN'T	SINCE	SIX
SIXTEEN	SIXTEENTH	SIXTH	SIXTIETH	SIXTY	SO
SOME	SOMEBODY	SOMEONE	SOMETHING	SOMETIMES	SOMEWHERE
SOON	STILL	SUCH	TEN	TENTH	THAN
THAT	THAT	THAT'S	THE	THEIR	THEIRS
THEM	THEMSELVES	THESE	THEN	THENCE	THERE
THEREFORE	THEY	THEY'D	THEY'LL	THEY'RE	THIRD
THIRTEEN	THIRTEENTH	THIRTIETH	THIRTY	THIS	THITHER
THOSE	THOUGH	THOUSAND	THOUSANDTH	THREE	THRICE
THROUGH	THUS	TILL	TO	TOWARDS	TODAY
TOMORROW	TOO	TWELFTH	TWELVE	TWENTIETH	TWENTY
TWICE	TWO	UNDER	UNDERNEATH	UNLESS	UNTIL
UP	US	VERY	WHEN	WAS	WASN'T
WE	WE'D	WE'LL	WERE	WE'RE	WEREN'T
WE'VE	WHAT	WHENCE	WHERE	WHEREAS	WHICH
WHILE	WHITHER	WHO	WHOM	WHOSE	WHY
WILL	WITH	WITHIN	WITHOUT	WON'T	WOULD
WOULDN'T	YES	YESTERDAY	YET	YOU	YOUR