On the Feasibility of Internet-Scale Author Identification

Draft. A version of this paper will appear at IEEE S&P 2012.

Arvind Narayanan relax@stanford.edu
Hristo Paskov hpaskov@stanford.edu
Neil Zhenqiang Gong neilz.gong@berkeley.edu
John Bethencourt bethenco@cs.berkeley.edu
Emil Stefanov emil@berkeley.edu
Eui Chul Richard Shin ricshin@berkeley.edu
Dawn Song dawnsong@cs.berkeley.edu

February 20, 2012

***Abstract*—We study techniques for identifying an anonymous author via linguistic stylometry, *i.e.*, comparing the writing style against a corpus of texts of known authorship. We experimentally demonstrate the effectiveness of our techniques with as many as 100,000 candidate authors. Given the increasing availability of writing samples online, our result has serious implications for anonymity and free speech — an anonymous blogger or whistleblower may be unmasked unless they take steps to obfuscate their writing style.**

While there is a huge body of literature on authorship recognition based on writing style, almost none of it has studied corpora of more than a few hundred authors. The problem becomes *qualitatively* different at a large scale, as we show, and techniques from prior work fail to scale, both in terms of accuracy and performance. We study a variety of classifiers, both "lazy" and "eager," and show how to handle the huge number of classes. We also develop novel techniques for confidence estimation of classifier outputs. Finally, we demonstrate stylometric authorship recognition on texts written in *different contexts.*

In over 20% of cases, our classifiers can correctly identify an anonymous author given a corpus of texts from 100,000 authors; in about 35% of cases the correct author is one of the top 20 guesses. If we allow the classifier the option of not making a guess, via confidence estimation we are able to increase the precision of the top guess from 20% to over 80% with only a halving of recall.

I. Introduction

Anonymity and free speech have been intertwined throughout history. For example, anonymous discourse was essential to the debates that gave birth to the United States Constitution—the Founding Fathers wrote highly influential "federalist" and "anti-federalist" papers under pseudonyms such as Publius and Cato [1]. Fittingly, the Constitution protects the right to anonymous free speech, as the US Supreme Court has ruled repeatedly. A 1995 decision reads [2]:

> Anonymity is a shield from the tyranny of the majority ... It thus exemplifies the purpose behind the Bill of Rights, and of the First Amendment in particular: to protect unpopular individuals from retaliation ... at the hand of an intolerant society.

Today, anonymous speech is more-or-less equivalent to anonymous online speech, and more relevant than ever all over the world. WikiLeaks has permanently and fundamentally changed international diplomacy [3], and anonymous activism has helped catalyze the recent popular uprisings in the Middle East and North Africa [4].

Yet a right to anonymity is meaningless if an anonymous author's identity can be unmasked by adversaries. There have been many attempts to legally force service providers and other intermediaries to reveal the identity of anonymous users. While sometimes successful [5], [6], in most cases courts have upheld a right to anonymous speech [7], [8], [9]. All of these efforts have relied on the author revealing their name or IP address to a service provider, who may in turn pass on that information. A careful author need not register for a service with their real name, and tools such as Tor can be used to hide their identity at the network level [10]. But if authors can be identified based on nothing but a passive comparison of the content they publish to other content found on the web, no legal precedents or networking tools can possibly protect them.

After all, any manually generated material will inevitably reflect some characteristics of the person who authored it, and these characteristics may be enough to determine whether two pieces of content were produced by the same person. For example, perhaps some anonymous blog author is prone to several specific spelling errors or has other recognizable idiosyncrasies. If an adversary were to find material with similar characteristics that the author had posted in an identified venue, the adversary might discover the true identity of the blog's author.

We investigate large-scale stylometric author identification, enough to constitute a widespread threat to anonymity. Previous work has shown that the author of a sample text can often be determined based on a manual or automated analysis of writing style, but only when the author is already known to be among a small set of possibilities (up to 300). Koppel et al. study authorship attribution with a larger number of authors, but this is not necessarily based on writing style (for a detailed discussion, see Section II). Before this work, it was unknown whether this type of attack could apply in any scenario resembling the Internet in scale.

Our work. To answer the above question, we have assembled a dataset comprising over 2.4 million posts taken from 100,000 blogs—almost a billion words. We chose blogs as our data source rather than IRC or message boards because blog posts are more readily available due to the RSS standard. Moreover, blogs are a common choice for political expression, raising privacy issues.

This dataset forms the basis for a series of large-scale experiments, in which we extract a set of features from each post in the dataset and use them to train classifiers to recognize the writing style of each of the 100,000 blog authors. We experimented with representative lazy and eager classifiers: nearest neighbor (NN), naive Bayes (NB) and support vector machines (SVM) and regularized least squares classification (RLSC).

Our technical contributions.

1) We describe features of the data that make it difficult to scale classification beyond a few hundred authors (Section IV; Analysis). In particular, the "masking" problem rules out many naive approaches (Section V). Otherwise-excellent techniques like linear discriminant analysis (LDA) fail because they are unable to take advantage of some normalization techniques due to the sparsity of the data (Section V-A). Efficiency is a crucial consideration for Internet-scale analyses. We describe our techniques for complexity improvements, both asymptotic (Section V-B; RLSC vs. SVM) and (large) constant factors (Section VI).
2) As with Koppel et al. [11], we find that straightforward lazy classifiers like NN perform surprisingly well. However, unlike the explanation of [11] (similarity-based approaches work better than classifiers when the number of classes is large), or that of [12] (lazy methods are particularly well-suited to NLP applications), we find that the difference is simply due to the difficulty of configuring more complex models with more parameters given a small number of training examples in each class, such as those based on an analysis of the covariance matrix. In particular we find that normalization makes a huge difference; an RLSC classifier with appropriate normalization performs equally well as NN.
3) We develop techniques for confidence estimation. The first is a variant of the "gap statistic" [13] that measures the difference between the best and the second-best matching classes. The second is to run two different classifiers and only output a result if they agree. The third is to combine the above two by meta-learning. Toward a related goal, we explore the strategy of combining two different classifiers using the respective confidence scores and other input features. We show that a meta-learner can achieve a boost in accuracy by picking the output of one or the other classifier by estimating which is more likely to be correct.
4) Finally, we use pairs (or tuples) of blogs listed under a user's Google profile as a way of generating unlabeled text that has a different *context* from the corresponding labeled text. We argue that validating stylometric classification on such a cross-context dataset, which is largely unexplored in previous work with online corpora, is an important measure of the applicability to many real-world scenarios.

Results and impact of our work. In experiments where we match a sample of just 3 blog posts against the rest of the posts from that blog (mixed in with 100,000 other blogs), the nearest-neighbor/RLSC combination is able to identify the correct blog in about 20% of cases; in about 35% of cases, the correct blog is one of the top 20 guesses. Via confidence estimation, we can increase precision from 20% to over 80% with a recall of 50%, which means that we identify 50% of the blogs overall compared to what we would have if we always made a guess.

The efficacy of the attack varies based on the number of labeled and anonymous posts available. Even with just a single post in the anonymous sample, we can identify the correct author about 7.5% of the time (without any confidence estimation). When the number of available posts in the sample increases to 10, we are able to achieve a 25% accuracy. Authors with relatively large amounts of content online (about 40 blog posts) fare worse: they are identified in over 30% of cases (with only 3 posts in the anonymous sample).

Our results are robust. Our numbers are roughly equivalent when using two very different classifiers, nearest neighbor and RLSC; we also verified that our results are not dominated by any one class of features. Further, we confirmed that our techniques work in a cross-context setting: in experiments where we match an anonymous blog against a set of 100,000 blogs, one of which is a different blog by the same author, the nearest neighbor classifier can correctly identify the blog by the same author in about 12% of cases. Finally, we also manually verified that in cross-context matching we find pairs of blogs that are hard for humans to match based on topic or writing style; we describe three such pairs in Appendix A.

The strength of the deanonymization attack we have presented is only likely to improve over time as better techniques are developed. Our results thus call into question the viability of anonymous online speech. Even if the adversary is unable to identify the author using our methods in a fully automated fashion, he might be able to identify a few tens of candidates for manual inspection as we detail in Section III. Outed anonymous bloggers have faced consequences ranging from firing to arrest and political persecution [14], [15].

Our experiments model a scenario in which *the victim has made no effort to modify their writing style*. We do not claim that our work will work in the face of intentional obfuscation; indeed, there is evidence that humans are good at consciously modifying their writing style to defeat stylometry [16]. A semi-automated tool for writing-style obfuscation was presented in [17]. We hope that our work will motivate the development of completely automated tools for transforming one's writing style while preserving the meaning. At the same time, privacy advocates might consider user education in order to inform privacy-conscious users about the possibility of stylometric deanonymization attacks. Regardless of future developments, our work has implications for the authors of all the sensitive anonymous speech that *already* exists on the Web and in various databases around the world.

II. Related Work

Stylometry. Attempts to identify the author of a text based on the style of writing long predate computers. The first quantitative approach was in 1877 by Mendenhall, a meteorologist, who proposed the word-length distribution as an author-invariant feature [18]. In 1901 he applied this technique to the Shakespeare–Bacon controversy [19]. The statistical approach to stylometry in the computer era was pioneered in 1964 by Mosteller and Wallace who used function words and Bayesian analysis[1] to identify the authors of the disputed Federalist Papers [20].

The latter work was seminal; dozens of papers appeared in the following decades, mostly focusing on identifying different features. For an overview, see Stamatatos's survey [21]. These studies considered a small number of authors (under 20) due to limitations in data availability. Since the late 1990s, the emergence of large digital corpora has transformed the nature of the field. Research in the last decade has been dominated by the machine-learning paradigm, and has moved away from the search for a single class of features toward an inclusive approach to feature extraction. Some important works are [22], [23], [24].

Like most stylometric techniques, these studies consider "topic-free" models and are able to discriminate between 100–300 authors. They have studied different domains (e-mail, blogs, etc.) Our own work is probably closest in spirit to Writeprints [23], especially in our approach to feature extraction.

The primary application of author identification during this period shifted from literary attribution to forensics — identifying authors of terroristic threats, harassing messages, etc. In the forensic context, the length of the text to be classified is typically far shorter and the known and unknown texts might be drawn from very different domains (*e.g.*, academic writing versus a threatening letter) [25]. Some forensic linguists who focus on legal admissibility scorn the use of online corpora for training of classifiers due to concerns over the level of confidence in the ground truth labels [26].

Stylometry has various goals other than authorship attribution including testing for multiple authorship [27], authenticity verification (*e.g.*, of suicide notes, disputed wills, etc.) [25], detection of hoaxes, frauds and deception [28], text genre classification [29] and author profiling (*e.g.*, age, gender, native language, etc.) [30].

Stylometry has been used to attribute authorship in domains other than text, such as music [31] and code [32], which also have grammars and other linguistic features shared with natural language. Other forensic tasks such as identifying the file type of a binary blob [33] and recovering text typed on a keyboard based on acoustic emanations [34] use similar techniques, although the models are simpler than linguistic stylometry.

The study of authorship attribution is scattered across various disciplines, and is not limited to stylometric techniques. We point the reader to Juola's excellent survey [35]. *Stylistics* in literary criticism also has authorship attribution as one of its goals, but it is subjective and non-quantitative; attempts to apply it to forensic author identification have been critized — rightly, in our opinion — as pseudoscience [36]. Contrariwise, quantitative methods are referred to as "non-traditional" in the literary community, and rather amusingly, Joseph Rudman in particular has argued that they should be rejected altogether for being the product of a machine [37].

Plagiarism detection can be seen as complementary to stylometric authorship attribution: it attempts to detect common content between documents, even if the style may have been changed [38].

Privacy and online anonymity. Very little work has been done to investigate the privacy implications of stylometry. In 2000 Rao and Rohatgi studied whether individuals posting under different pseudonyms to USENET newsgroups can have these pseudonyms linked based on writing style [39]. They used function words and Principal Component Analysis on a dataset of 185 pseudonyms belonging to 117 individuals to cluster pseudonyms belonging to the same individual. They also performed a smaller experiment demonstrating that matching between newsgroup and e-mail domains was possible, but they did not find matching between RFC and newsgroup postings to be feasible.

Only Koppel et al, have attempted to perform author identification at anything approaching "Internet scale" in terms of the number of authors. They reported preliminary experiments in 2006 [40] and follow-up work in 2011 [11] where they study authorship recognition with a 10,000-authors blog corpus. This work is intriguing but raises a number of methodological concerns.

The authors use only 4-grams of characters as features. It is not clear to what extent identification is based on recognition of the *author* vs. the *context.* On the other

[1] *i.e.*, what we now call the Naive Bayes classifier.

hand, we use various linguistic features that are known to characterize authors, and explicitly eschew context-specific features such as topic markers.

Indeed, character-based analysis is likely to be very susceptible to topic biases, and Koppel et al. state that this distinction is immaterial to them. Therefore, while an interesting and impressive result, it is hard to characterize their work as stylometric authorship recognition.

The susceptibility of character n-gram–based features to context biases is exacerbated by the fact that (as far as we can tell) Koppel et al. perform no pre-processing to remove common substrings such as signatures at the end of posts. Therefore we suggest that the results should be interpreted with caution unless the technique can be validated on a cross-context dataset.

Similarly, in another paper, Koppel et al. again study authorship recognition on a 10,000-author blog corpus [30]. Here they test both content-based features (specifically, 1,000 words) and stylistic features. The former succeeds 52–56% of the time, whereas the latter only 6% of the time. In this paper, the authors note that many character n-grams "will be closely associated to particular content words and roots," in line with our observations above.

Moving on, Nanavati et al. showed that stylometry enables identifying reviewers of research papers with reasonably high accuracy, given that the adversary, assumed to be a member of the community, has access to a large number of unblinded reviews of potential reviewers by serving on conference and grant selection committees [41]. Several researchers have considered whether the author of an academic paper under blind review might be identified solely from the citations [42], [43].

Other research in this area has investigated manual and semi-automated techniques for transforming text to successfully resist identification [16], [17]. These papers consider off-the-shelf stylometric attacks; the resilience of obfuscation to an attack crafted to take it into account has not been studied and is a topic for future research.

A variety of recent technological developments have made it possible for a website to track visitors and to learn their identity in some circumstances. First, browser fingerprinting allows tracking a user across sites and across sessions [44]. Second, by exploiting bugs such as history stealing [45], a malicious site might be able to find a visitor's identity if they use popular social networking sites. These deanonymization techniques can be seen as complementary to ours.

Recent work has also demonstrated the ease of connecting different online profiles of a person [46], [47]; this makes it easier for an adversary carrying out our attack to assemble labeled texts for large numbers of Internet users.

Behavioral biometrics — voice [48], gait [49], handwriting [50], keystroke dynamics [51], etc. — are ways to fingerprint an individual that go beyond purely physical characteristics. Stylometric fingerprints can be considered a extension of behavioral biometrics. Our work can also be seen as an extension of *deanonymization* research: while Sweeney showed that a combination of our attributes is surprisingly unique [52], Narayanan and Shmatikov showed that the same principle applies to our tastes and preferences [53]. The present work focuses on style and behavior; the respective deanonymization algorithms show a natural progression in complexity as well as in the amount of data required.

III. Attack Model and Experimental Methodology

In this section, we discuss how a real attack would work, the motivation behind our experimental design and what we hope to learn from the experiments.

Primarily, we wish to simulate an attempt to identify the author of an anonymously published blog. If the author is careful to avoid revealing their IP address or any other explicit identifier, their adversary (e.g., a government censor) may turn to an analysis of writing style. We assume that the author does not make any attempt to hide their writing style, whether due to lack of awareness of the possibility of stylometric deanonymization, or lack of tools to do so. A semi-automated obfuscation tool was presented in [17].

By comparing the posts of the anonymous blog with a corpus of samples taken from many other blogs throughout the Internet, the adversary may hope to find a second, more easily identified blog by the same author. To have any hope of success, the adversary will need to compare the anonymous text to far more samples than could be done manually, so they can instead extract numerical features and conduct an automated search for statistically similar samples.

This approach may not yield conclusive proof of a match; instead, we imagine the adversary's tools returning a list of the most similar possibilities for manual followup. A manual examination may incorporate several characteristics that the automated analysis does not, such as choice of topic(s) (as we explain later, our algorithms are "topic-free"), location[2] etc. Alternately, a powerful adversary such as law enforcement may require Blogger, WordPress, or another popular blog host to reveal the login times of the top suspects, which could be correlated with the timing of posts on the anonymous blog to confirm a match. Thus the adversary's goal might be to merely narrow the field of possible authors of the anonymous text enough that another approach to identifying the author becomes feasible. As a result, in our experiments we test classifiers which return a ranking of classes by likelihood, rather than those which can only return the most likely class.

[2] For example, if we were trying to identify the author of the once-anonymous blog Washingtonienne [54] we'd know that she almost certainly lives in or around Washington, D.C.

Category	Description	Count
Length	number of words/characters in post	2
Vocabulary richness	Yule's K[3] and frequency of *hapax legomena*, *dis legomena*, etc.	11
Word shape	frequency of words with different combinations of upper and lower case letters.[4]	5
Word length	frequency of words that have 1–20 characters	20
Letters	frequency of *a* to *z*, ignoring case	26
Digits	frequency of 0 to 9	10
Punctuation	frequency of . ? ! , ; : ( ) " - '	11
Special characters	frequency of other special characters ` ~ @ # $ % ^ & * _ + = [ ] { } \ | / < >	21
Function words	frequency of words like 'the', 'of', and 'then'	293
Syntactic category pairs	frequency of every pair (A, B), where A is the parent of B in the parse tree	789

Table I
THE FEATURES USED FOR CLASSIFICATION. MOST TAKE THE FORM OF FREQUENCIES, AND ALL ARE REAL-VALUED.

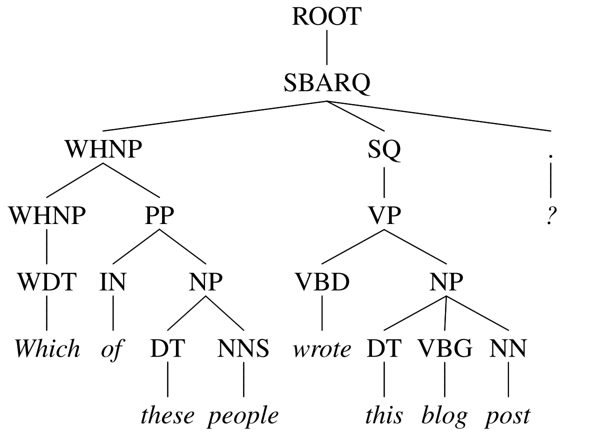

Figure 1. A sample parse tree produced by the Stanford Parser.

Conversely, the adversary might be interested in casting a wide net, looking to unmask one or some few of a group of anonymous authors. In this case, it would help the adversary to have confidence estimates of each output, so that he can focus on the ones most likely to be correct matches. Therefore we consider confidence estimation an important goal.

As in many other research projects, the main challenge in designing our experiments is the absence of a large dataset labeled with ground truth. To measure the feasibility of matching one blog to another from the same author, we need a set of blogs already grouped by author. However, manually collecting a large number of blogs grouped this way is impractical, if the experiments are to be anything resembling "Internet scale."

Therefore, our first approach to conducting experiments is to simulate the case of an individual publishing two blogs by dividing the posts of a single blog into two groups; we then measure our ability to match the two groups of posts back together. Specifically, we select one of a large number of blogs and set aside several of its posts for testing. These test posts represent the anonymously authored content, while the other posts of that blog represent additional material from the same author found elsewhere on the Internet. We next train a classifier to recognize the writing style of each of the blogs in the entire dataset, taking care to exclude the test posts when training on the blog from which they were taken. After completing the training, we present the test posts to the classifier and use it to rank all the blogs according to their estimated likelihood of producing the test posts. If the source of the test posts appears near the top of the resulting list of blogs, the writing style of its author may be considered especially identifiable. As will be shown in Section VI, our experiences applying this process have revealed surprisingly high levels of identifiability: using only three test posts, the correct blog is ranked first out of 100,000 in 20% of trials.

At this point, the astute reader is no doubt brimming with objections to the methodology described above. How can we be sure that any linking we detect in this way is unintentional? Suppose a blog author signs each of their posts by adding their name at the end; they would not be at all surprised to discover that an automated tool can determine that the posts have the same author. More subtly, rather than linking posts based on similarities in writing style, our classifiers may end up relying on similarities in the content covered by the writing, such as specific words related to the topic of the blog. Not only is this problematic given that we attempt to match test posts to labeled posts from the same blog, we additionally anticipate that anonymously-authored blogs will frequently tend to cover different topics of greater sensitivity, compared to identified blogs written by the same author. We take the following strategies in avoiding these pitfalls:

1) We begin by filtering out any obvious signatures in the posts by checking for common substrings. We also remove markup and any other text that does not appear to be directly entered by a human in order to avoid linking based on the blog software or style templates used.
2) We carefully limit the features we extract from each post and provide to the classifier. In particular, unlike previous work on author identification, we do not employ a "bag of words" or any other features that can discover and incorporate arbitrary content. Our word-based features are limited to a fixed set of function words which bear little relation to the subject of discussion (e.g., "the," "in," etc.). While we do make use of single character frequencies, we exclude bigrams and trigrams, which may be significantly influenced

[3] We computed Yule's K in the same way as in [23].

[4] All upper case, all lower case, only first letter upper case, camel case (CamelCase), and everything else.

by specific words.

3) We follow up the experiments described above (post-to-blog matching) with additional experiments which actually involve matching distinct blogs to one another. Specifically, we assembled a small collection of sets of blogs with the same author; for these experiments (blog-to-blog matching), we set aside one blog as the test content, mix the others from the same author into the full dataset of 100,000 blogs, and then measure our ability to pick them back out.

The results of the blog-to-blog matching experiments roughly match the post-to-blog matching results, and we also found the results were not dominated by any one class of features. These facts have given us confidence that our methods are in fact discovering links in writing *style*—not blog style templates or the topic of a blog.

In addition to serving as a "sanity check" for our results in the abovementioned manner, the cross-context setting is arguably closer to the adversary's practical task in some scenarios. This is certainly not always the case: in some applications of stylometric authorship recognition, the available labeled text might be from the same context as the unlabeled text. This was the case in Mosteller and Wallace's study of the disputed federalist papers; in the blogging scenario, an author might decide to selectively distribute a few particularly sensitive posts anonymously through a different channel.

Yet in other cases, the unlabeled text might be political speech, whereas the only labeled text by the same author might be a cooking blog. Context encompasses much more than topic: the tone might be formal or informal; the author might be in a different mental state (e.g., more emotional) in one context versus the other, etc.

Thus, we can expect an author's writing style to differ more across different contexts than it does within the same context, and indeed, our cross-context results are numerically somewhat weaker. In the future we hope to understand exactly how writing style varies across contexts and to utilize this to improve cross-context classification.

IV. Data Sources and Features

Having given some high-level motivation for our experimental approach and methodology, we now detail our sources of data, the steps we took to filter it, and the feature set implemented.

Data sources. The bulk of our data was obtained from the ICWSM 2009 Spinn3r Blog Dataset, a large collection of blog posts made available to researchers by Spinn3r.com, a provider of blog-related commercial data feeds [55]. For the blog-to-blog matching experiments, we supplemented this by scanning a dataset of 3.5 million Google profile pages for users who specify multiple URLs [47]. Most of these URLs link to social network profiles rather than blogs, so we further searched for those containing terms such as "blog," "journal," etc. From this list of URLs, we obtained RSS feeds and individual blog posts.

Feature	**Information Gain in Bits**
Frequency of '	1.097
Number of characters	1.077
Freq. of words with only first letter uppercase	1.073
Number of words	1.060
Frequency of (NP, PRP) (noun phrase containing a personal pronoun)	1.032
Frequency of .	1.022
Frequency of all lowercase words	1.018
Frequency of (NP, NNP) (noun phrase containing a singular proper noun)	1.009
Frequency of all uppercase words	0.991
Frequency of ,	0.947

Table II
The top 10 features by information gain.

We passed both sets of posts through the following filtering steps. First, we removed all HTML and any other markup or software-related debris we could find, leaving only (apparently) manually entered text. Next, we retained only those blogs with at least 7,500 characters of text across all their posts, or roughly eight paragraphs. Non-English language blogs were removed using the requirement that at least 15% of the words present must be among the top 50 English words, a heuristic found to work well in practice. Of course, our methods could be applied to almost any other language, but some modifications to the feature set would be necessary. To avoid matching blog posts together based on a signature the author included, we removed any prefix or suffix found to be shared among at least three-fourths of the posts of a blog. Duplicated posts were also removed.

At the end of this process, 5,729 blogs from 3,628 Google profiles remained, to which we added 94,271 blogs from the Spinn3r dataset to bring the total to 100,000. Of the 3,628 retained Google profiles, 1,763 listed a single blog; 1,663 listed a pair of blogs; other 202 listed three to five. Our final dataset contained 2,443,808 blog posts, an average of 24 posts per blog (the median was also 24). Each post contained an average of 305 words, with a median of 335.

Features. We extracted 1,188 real-valued features from each blog post, transforming the post into a high-dimensional vector. These feature vectors were the only input to our classifiers; the text of the blog post played no further role after feature extraction.

Table I summarizes the feature set. All but the last of these categories consist of features which reflect the distributions of words and characters in each of the posts. Many of them, such as the distribution of word length and frequency of letters in the alphabet, come from previous work on author identification [23]. We also analyze the capitalization of words, as we expect the level of adherence to capitalization conventions to act as a distinguishing component of an

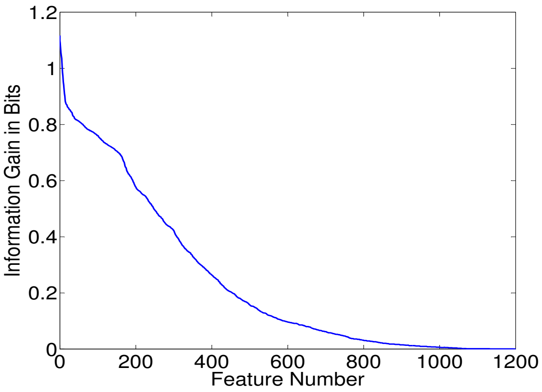

Figure 2. Information gain by feature.

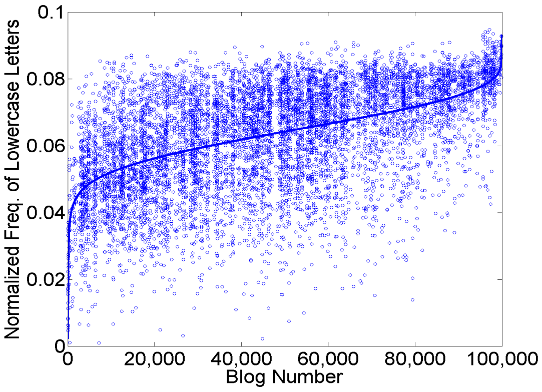

Figure 3. Per-post values (dots) and per-blog means (line) of an example feature across the dataset.

author's writing style given the unedited, free-form nature of written content on the Internet. We compute each of the letter frequency features as the number of occurrences of a specific letter in a post divided by the length of the post in characters. Other single-character frequencies are computed likewise, and word-frequency features are computed analogously, but at the level of words. Appendix C shows the 293 function words we use. These words are topic-independent.

The last category of features in Table I, we use the Stanford Parser [56] to determine the syntactic structure of each of the sentences in the input posts. As output, it produces a tree for each sentence where the leaf nodes are words and punctuation used, and other nodes represent various types of syntactic categories (phrasal categories and parts of speech). Figure 1 shows an example parse tree as produced by the Stanford Parser, with tags such as *NN* for noun, *NP* for noun phrase, and *PP* for prepositional phrase. We generate features from the parse trees by taking each pair of syntactic categories that can appear as parent and child nodes in a parse tree tree, and counting the frequency of each such pair in the input data.

Two previous studies, Baayen et al. [57] and Gamon [58], used rewrite-rule frequencies extracted from parse trees of sentences as features.[5] Our syntactic-category pairs are similar, but less numerous (in Gamon's work, for instance, the number of possible rewrite rules is over 600,000). The rationale for our choice was that this could make our classifiers more robust and less prone to overfitting, not to mention more more computationally tractable.

We re-emphasize that we strove for these features to reflect aspects of writing style that remain unchanged for some given author, regardless of the topic at hand. In particular, the frequencies of 293 function words contain little meaning on their own and instead express grammatical relationships, so they avoid revealing information about the topic while capturing the writing style of the author.

To gain a better intuitive understanding of the relative utility of the features and for use in feature selection, we computed the information gain of each feature over the entire dataset [59]. We define information gain as

$$IG(F_i) = H(B) - H(B|F_i) = H(B) + H(F_i) - H(B, F_i),$$

where H denotes Shannon entropy, B is the random variable corresponding to the blog number, and F_i is the random variable corresponding to feature i. Since the features are real-valued, and entropy is defined only for discrete random variables[6], we need to sensibly map them to a set of discrete values. For each feature, we partitioned the range of observed values into twenty intervals. We reserved one bin for the value zero, given the sparsity of our feature set; the other nineteen bins were selected to contain approximately equal numbers of values across all the posts in the dataset.

A portion of the result of this analysis is given in Table II, which lists the ten features with greatest information gain when computed as described. Several other binning methods were found to produce similar results. With information gains, measured in bits, ranging from 1.097 to 0.947, these features can all be considered roughly equal in utility. Perhaps least surprisingly, the length of posts (in words and characters) is among the best indicators of the blog the posts were taken from.[7] Several punctuation marks also appear in the top ten, along with the three most common patterns of upper- and lowercase letters and two syntactic category pairs. The information gains of all 1,188 features are shown in Figure 2.

To give a sense of the typical variation of feature values both within a blog and between different blogs, Figure 3 displays a representative example of one of these ten features: the frequency of all lowercase words. The plot was

[5] An example of a rewrite rule is A:PP $\rightarrow$ P:PREP + PC:N, meaning that an adverbial prepositional phrase is constituted by a preposition followed by a noun phrase as a prepositional complement [21].

[6] A definition also exists for continuous random variables, but applying it requires assuming a particular probability mass function.

[7] However, these features may not be as useful in the cross-context setting.

generated by sorting the 100,000 blogs according to the mean of this feature across their posts. The means are shown by the solid line, and the values for individual posts are plotted as dots. For legibility, the figure only shows every third post of every one hundredth blog. As one might expect, the values vary from post to post by much larger amounts than the differences in mean between most pairs of blogs, indicating that this feature alone carries a fairly small amount of information. The corresponding plots for features with lower information gain look similar, but with less variation in means or more variation between individual posts from the same author. From this plot and the information gains shown in Figure 2, it's clear that many features will need to be considered in combination if we are to effectively identify authors.

Analysis. Perhaps the most important aspect of our data set is the large number of classes and training examples. As the number of classes increases, the nature of the classification task changes fundamentally in two ways: accuracy and computational demands. An immediate consequence of having more classes is that they become more densely distributed; the number of classes that are ”close” to one another increases. As such, the decision boundary that separates each class now has to accurately distinguish it from a much larger number of close alternatives. This general principle manifests in different ways; “masking” (Section V-B) is a common problem.

At the same time, the large number of examples places hard limits on the kinds of classifiers we can use — anything with a superlinear training complexity is infeasible. A similar restriction applies to the way we adapt binary classifiers to handle multiple classes. In particular, the popular and robust “all-pairs” multiclass regime falls out of the picture because it requires comparing all pairs classes.

While the above argument holds for any large dataset, ours is particularly problematic because there are a small number of training examples per class. On average, there are only 24 posts for each author, and indeed, 95% of all authors have 50 posts or less. Compounding the lack of data is the sparsity of each sample; each post has approximately 305 nonzero dimensions. These factors are troublesome compared to the 1188 dimensions we use to represent the data because there are very few examples with which to estimate class specific parameters. In particular, methods such as nearest-neighbors that rely on estimating class means should do well, but anything that requires estimating a covariance matrix (e.g. RLSC and SVM) for each class will require heavy regularization.

Finally, Figure 4 is a plot of the eigenvalue spectrum we obtain from running a Principal Component Analysis on the data (note that the y-axis is logarithmic). Based on this analysis, a number of dimensions are spurious — we could represent the data to 96% accuracy using only 500 dimensions. While this presents a sizeable reduction in dimensionality, it is not enough to allow for reliable estimation of anything more than a class centroid. We would have to represent the data at 27% accuracy in order to shrink it to a number of dimensions that is not limited by the low number of samples per class.

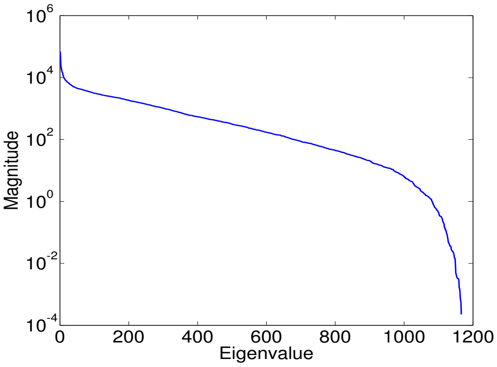

Figure 4. Eigenvalue spectrum.

V. Classification

A. Normalization

We utilized three types of normalization:

- row-norm Rescale each row (training or test point) to have norm 1, i.e., divide each entry in a row by the norm of that row.
- feature-mean Rescale each column (feature) to have mean 0 and variance 1, i.e., subtract the mean from each column and divide by the standard deviation.
- feature-mean-nonzero Rescale each column so that the nonzero entries have mean 1, i.e. divide by the mean of the nonzero entries of each column (since all features are nonnegative).

Row normalization is essential because it allows us to compare different documents based on the relative distribution of features in the document and irrespective of its norm. Similarly, column normalization allows us to combine information from different features when they are measured in different units. While the feature-mean is a standard normalization in statistics, our feature-mean-nonzero warrants some explanation. Like feature-mean, this normalization provides scale invariance but it calculates the normalization parameter from the support of each feature, i.e. the nonzero entries. Since the majority of our columns are sparse, focusing on the support allows us to capture structure specific to the nonzero entries. A statistic such as the average would be unecessarily skewed towards zero if we calculated it the standard way. Moreover, since we divide the features by the mean of their support rather than subtracting it, all nonzero entries are still positive and zero entries are still zero so they are distinct from the average value of the support, which is 1. Finally, the order of normalization is important; we perform column normalization followed by row normalization.

B. Classifiers

This section discusses the algorithms and configurations we used to identify authors. We denote a labeled example as a pair $(\vec{x}, y)$, where $\vec{x} \in R^n$ is a vector of features and $y \in \{1, 2, \ldots, m\}$ is the label. In our case, $n = 1188$, the number of features; and $m = 100000$, the number of blogs in our dataset. After training a classifier on the labeled examples, we present it with one or more unlabeled examples $\vec{x}_1, \vec{x}_2, \ldots$, test posts taken from a single blog in the case of post-to-blog matching experiments, or an entire blog in the case of blog-to-blog matching. In either case, we rank the labels $\{1, 2, \ldots, m\}$ according to our estimate of the likelihood that the corresponding author wrote the unlabeled posts.

One key difference between our experiments and the typical classification scenario is that we know that $\vec{x}_1, \vec{x}_2, \ldots$ have the same label. To exploit this knowledge we collapse the vectors $\vec{x}_1, \vec{x}_2, \ldots$ to their mean, and classify that single vector. Another possibility that we did not try would be to classify each point separately and use some form of voting to combine the decisions. Two points of note: first, collapsing the feature vectors is not mathematically equivalent to treating all of the author's writing as a single document. Second, we have described how we handle different points with the same label for *classification*, not training. During training we collapse the feature vectors for some classifers, but not others, as described below.

Nearest Neighbor. To train the nearest neighbor classifier we collapse each class' training examples into its centroid. The most common form of the nearest neighbor algorithm treats each training example separately and therefore stores all of the training data. However, this approach is too slow for our dataset. Thus, we only store one centroid per class, effectively providing a "fingerprint" for each blog. A new point is classified by computing its Euclidean distance to each of the fingerprints and using the label of the closest class.

Linear Discriminant Analysis. Linear Discriminant Analysis (LDA) is a popular algorithm for text classification and is among the top performers if we do not normalize our data. However, as we discuss more precisely in Appendix B, it cancels out any feature normalizations that we perform. We find feature normalization to be particularly necessary with our data and large number of classes, so we do not include this classifier in our final results.

Naive Bayes. The standard Naive Bayes classifier assumes individual features are distributed according to a Gaussian distribution and are conditionally independent for each class. Mathematically, it works similarly to our nearest neighbor implementation, but it also takes each feature's variance in account. Specifically, we train by computing the mean μ_i and variance [8] σ_i^2 of each feature i for each class. We classify a point $\vec{x} = (x_1, \ldots, x_n)$ by selecting the closest class, where distance is computed as

$$\sum_{i=1}^{n} \frac{(x_i - \mu_i)^2}{\sigma_i^2} + \log(\sigma_i^2)$$

Note that this a weighted Euclidean distance that places greater value on features with less variance and that normalizes across classes by taking into account the $\log(\sigma_i^2)$, the total variance of each class.

Binary Classifiers Thus far, we have discussed inherently multiclass classifiers. However, there is a large literature on binary classifiers that we would like to experiment with. We can extend a binary classifier to the multiclass setting by training numerous binary classifiers. The two most popular methods for doing this are the one-vs-all and all-pairs regimes. In the former, a classifier is trained for each class by labelling all points that belong to that class as positive and all others as negative. A point is labelled by running it through all of the classifiers and selecting the classifier which votes most positively. In an all-pairs regime, a classifier is trained for each pair of classes, resulting in a number of classifiers that is quadratic in the number of classes. This approach is far too costly when there are 100,000 classes, so we only consider one-versus-all.

Masking. As demonstrated by our experiments, the standard one-versus-all regime can mask certain classes so that points from them will always be mislabeled. Restricting ourselves to linear binary classifiers — which are the only kind of binary classifers we consider in this paper — the root of this problem stems from the one-versus-all regime's assumption that each class can be separated from all of the others using a linear decision boundary. This assumption is reasonable only when there are many more dimensions than classes; as soon as the number of classes is on the same order as the dimensionality of the data, we run into trouble. In our case, there are far more classes than dimensions and the masking problem is particularly severe. In experiments in which we used 50,000 testing points, each from a distinct class, only 628 distinct classes were predicted by RLSC. Moreover, the same 100 classes appeared in over 82% of the predicted labels. This masking problem therefore makes the standard one-versus-all scheme unusable with any binary classifier. We present a weighting scheme that alleviates this problem in the section on Regularized Least Squares Classification.

Support Vector Machine (SVM). SVMs are a popular binary classifier [60], [61] and we use the SVM Light implementation [62]. Ideally, we would provide each SVM

[8] A small-sample correction of 5×10^{-6} was added to each variance to prevent it from being zero. This occurs frequently because our features are sparse so that some features are never seen in a particular class.

with all of the posts, but the large size of our dataset makes this infeasible. We use posts from a sample of 1000 blogs as negative examples and retain all positive examples for each one-vs-all classifier. To improve the ability of an SVM to distinguish between its associated class and all others, we ensure that this set of 1000 blogs includes the 100 closest classes as determined by the Euclidean distance of the class centroids. The remaining 900 are selected uniformly at random. we use a simple linear kernel for classification.

Regularized Least Squares Classification (RLSC). RLSC is an adaptation of the traditional least squares regression algorithm to binary classification. It was introduced as a computationally efficient alternative to SVMs and has similar classification performance. The main difference between the two is that RLSC uses a squared loss to penalize mistakes while SVM uses a non-differentiable "hinge loss." The former admits a closed form solution that uses optimized linear algebra routines; SVMs rely on convex optimization and take longer to train. This efficiency does not require us to subsample the data when training so that unlike SVMs, RLSC uses all of the training data. Perhaps the most notable computational difference between the two is that it takes the same time to solve for a single RLSC classifier as it does to solve for T separate ones in a one-vs-all scheme with T classes. Thus, our computation time with RLSC is $O(nd^2+d^3)$ versus $O(nd^2+Td^3)$ with SVM for n training examples in d dimensions.

Since RLSC is a binary classifier, we employ a one-vs-all regime to extend it to multiple classes. However, we found that the aformentioned masking problem makes RLSC perform very poorly. We remedy the situation by weighting the training data given to each one-vs-all classifier. In particular, consider what happens when we train a one-vs-all classifier with $100,000$ classes: there are, on average, $99,999$ times more negative examples than positive ones. There is little incentive for the classifier to learn anything since it will be overwhelmingly correct if it simply labels everything as negative. We counteract this problem by penalizing errors on positive examples more so that they have effectively the same weight as negative examples. After some careful algebra, this approach, which is the version of RLSC we will use in the remainder of the paper, preserves the efficient runtime of RLSC and improves accuracy by two orders of magnitude.

Classifier Configurations. While we would like to report our classifiers' performance with all combinations of features and normalization, we only report a tractable subset that illustrates the effects of normalization and masking. We used the Naive Bayes classifier with the 400 features with greatest information gain and no normalization to showcase the best performance without any normalization. Next, the nearest neighbor classifier uses two choices of normalization: row-norm and feature-mean; row-norm and feature-mean-nonzero to demonstrate the benefits of our sparse feature normalization method over the standard one. In the way of binary classifiers, we report the SVM with a linear kernel and a standard one-vs-all regime using row-norm and feature-mean normalizations to illustrate the issue of masking. Finally, RLSC is shown with a linear kernel and weighted one-vs-all regime using row-norm and feature-mean-nonzero normalizations and is one of the top performers. Although SVM would have similar accuracy as RLSC using the same configuration, the latter was more useful because its speed allowed us to experiment with many configurations.

Lazy vs. eager. The existing literature on authorship attribution distinguishes between "eager" and "lazy" classifiers [12] as ones which spend time computing a model from training data versus ones, such as nearest neighbors, that simply store the data and rely on a good distance function for classification. This distinction appears in Koppel et al. under a different name when they discuss similarity-based (lazy) and machine-learning (eager) classification paradigms. We find this black-and-white distinction to be unhelpful because methods often fall on a spectrum. Indeed our implementation of nearest neighbor — a prototypical lazy method —- summarizes information by learning the mean of each class. Moreover, the one-versus-all method for SVM or RLSC bears a substantial resemblance to the lazy approach because dot products are a similary measure. Similarly, LDA, which at first blush is an eager method, is also a similarity-based method because it classifies by finding the Mahalanobis distance — a generalization of Euclidean distance — between the mean of each class and the point in question. Finally unlike Koppel et al., our experiments indicate that when properly configured, lazy and eager approaches have a very similar performance profile for all values of N, the number of classes.

C. Confidence estimation and combining classifiers

A confidence estimator is a real-valued function that takes the input/output pair of a classifier as input, and that outputs a positive score. Larger scores correspond to higher confidences; the score is intended to be monotonically increasing in the probability of correctness of the classifier's output. A more sophisticated confidence estimator might be able to produce an actual probability score, but this stronger requirement is not a goal for us. Applying different thresholds to the confidence score allows to achieve various trade-offs between *precision* and *recall*.

In more detail, consider a classifier C and a confidence estimator conf_C. This allows us to build a classifier C_t parametrized on a threshold t, which outputs $C(x)$ on input x if $\mathsf{conf}_C(x, C(x)) \geq t$ and $\perp$ otherwise. The recall of C_t is the ratio of correct outputs of C_t to the correct outputs of C, and its precision is the probability of a correct answer on only the examples it attempts to answer.

In general, recall and precision are inversely related; if we attempt to label more examples, we will produce more examples with correct labels, but even more examples with incorrect labels. A good estimate of confidence is critical to the recall-precision tradeoff. If the assumption about the monotonicity of the relationship between confidence and probability of correctness holds, then recall and precision will be inversely related. However, if we are unfortunately misguided to attempt an answer the harder a point is to label, lower recall rates will also have a lower precision!

We use the "gap statistic" that was described in [13]; a similar heuristic called "eccentricity" was described in [53]. The gap statistic applies to classifiers that output a distance or a similarity score between a test point and each class. This is a property that's required in practice for many purposes, for example, to employ a one-versus-all strategy. Recall that nearest neighbor, SVM and RLSC all output a distance or distance-like score.

At its core, the gap statistic simply outputs the magnitude of the difference between the best match (i.e., lowest distance) and second best match. The rationale is a simple Bayesian argument: if the output of the classifier is incorrect, the best score is unlikely to be well-separated from the rest of the scores—there is no particular reason to expect one incorrect class to match significantly better than other incorrect classes.[9]

On the other hand, it is possible, although by no means certain, that the best score is highly separated from the rest when the classifier output is correct. Intuitively, the more clustered together the points of a class are, i.e., the more distinct an author's writing style, the more likely this is to happen. Flipping around the prior and posterior, we see that a high gap statistic implies that it is more likely that the classifier output is correct. We found that when using the gap statistic with nearest-neighbors, it was also useful to perform the row-norm normalization. No normalization was necessary for RLSC.

Finally, while the gap statistic is the key ingredient of the confidence score, it is not the only one. We use a meta-learner that combines the gap with one other feature, namely the number of available training points for the top scoring class. The rationale for this feature is the obvious one: classifications based on more training data are more likely to be correct. In general, we can imagine using any (or all!) features of the input, although we do not have a strong reason to believe that this will result in a significant improvement.

An entirely different approach to confidence estimation is, given a classifier C, to employ an auxiliary classifier C_{aux}, and to measure the level of agreement between the outputs of C and C_{aux}. In the simplest case, we check whether or not the top match in both cases is the same, and output 1 or 0 accordingly. Thresholding based on this confidence measure is equivalent to only producing an output if C and C_{aux} produce the same output. Thus the roles of C and C_{aux} are symmetrical.

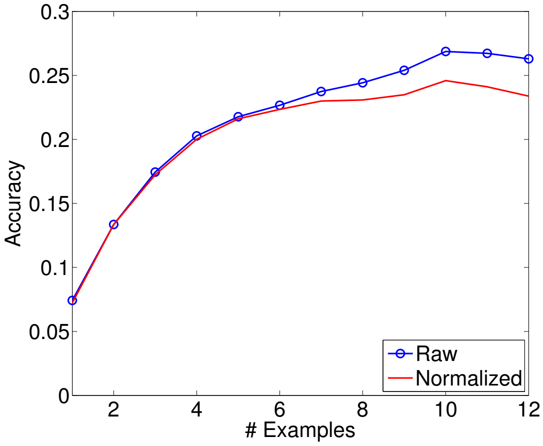

Figure 5. Effect of test set size.

This leads naturally to the idea of combining two (or more) classifiers, both for improving accuracy and for confidence estimation. The two goals are closely linked to each other. As before, we can use a meta-learner based on all available features to pick the output of one or the other classifier, although we choose to restrict ourselves to the gap statistic and the number-of-training-points feature.[10]. For confidence estimation, we use the above, together with a measure of agreement between the two classifiers. The latter is derived by looking at the top k outputs of each classifier.

As the reader is no doubt aware, there is a trememdous amount of work in the machine learning literature on stacked generalization, boosting and ensemble methods. Our goal in investigating combinations of classifiers is not to break new ground in these areas, but rather to show that it improves accuracy, demonstrating that our numerical results are only a lower bound. We suspect that an adversary interested in squeezing out the last percentage-point of precision will be able to go quite a bit farther.

VI. Experimental Results

In Figure 7 which summarizes our most important results, we provide the full distribution of outcomes obtained by applying various classifiers to the post-to-blog experiment III. We now give the details of the procedure used to obtain these results.

In each trial, we randomly selected three posts of one blog and set them aside as the testing data. The classifiers were then used to rank each blog according to its estimated

[9] Which is not to say that it can't happen: in the pathological worst case, all blog posts in the test set might consist of text entirely quoted from another blog, and yet not denoted as such via HTML tags, making it challenging to filter out automatically.

[10] Since there are two classifiers, this gives us a four-dimensional space

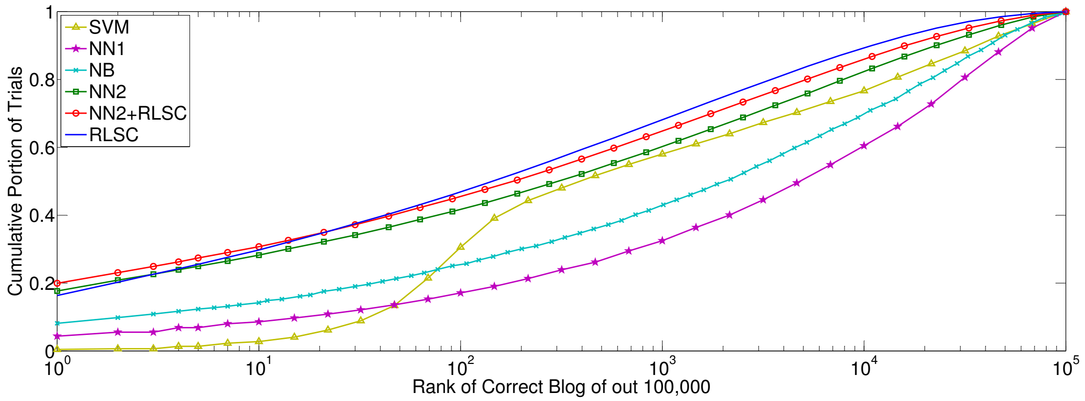

Figure 7. Results of the post-to-blog matching experiments, using three posts (roughly 900 words). The y-axis is the probability that the correct class was among the top K ranks, for each K (x-axis).

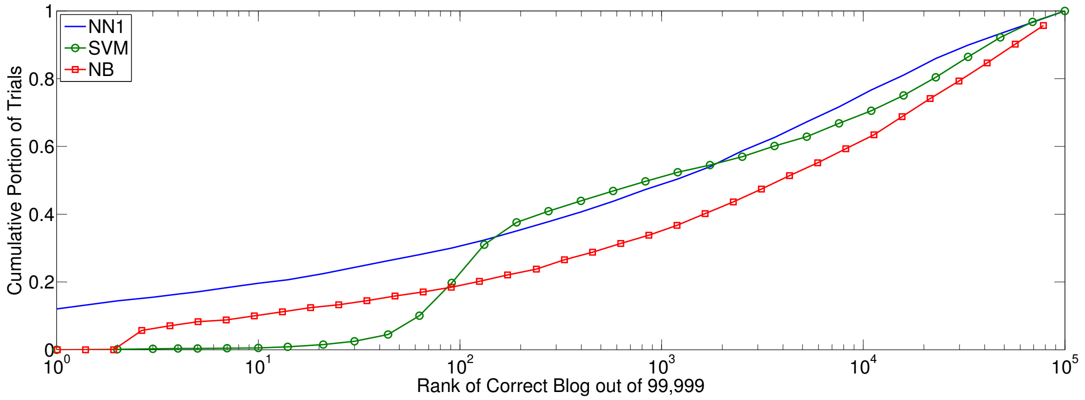

Figure 8. Results of the blog-to-blog matching experiments.

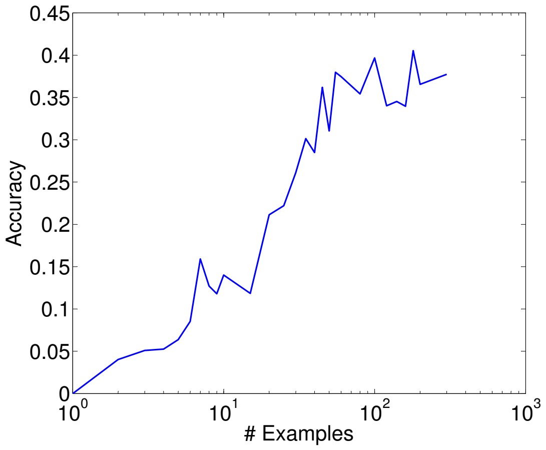

Figure 6. Effect of training set size.

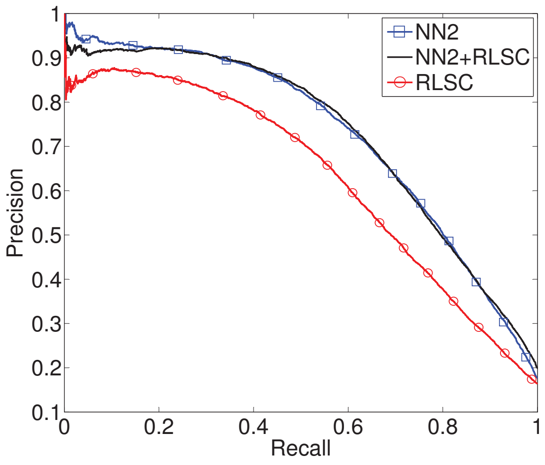

Figure 9. Confidence estimation: precision vs. recall.

likelihood of producing the test posts. Of course, we were careful to ensure that the classifiers were not given the test posts during training. For this experiment, we only selected blogs from the Spinn3r dataset as the source of test posts, but we used the classifiers to rank all 100,000 blogs. In each trial, we recorded the rank of the correct

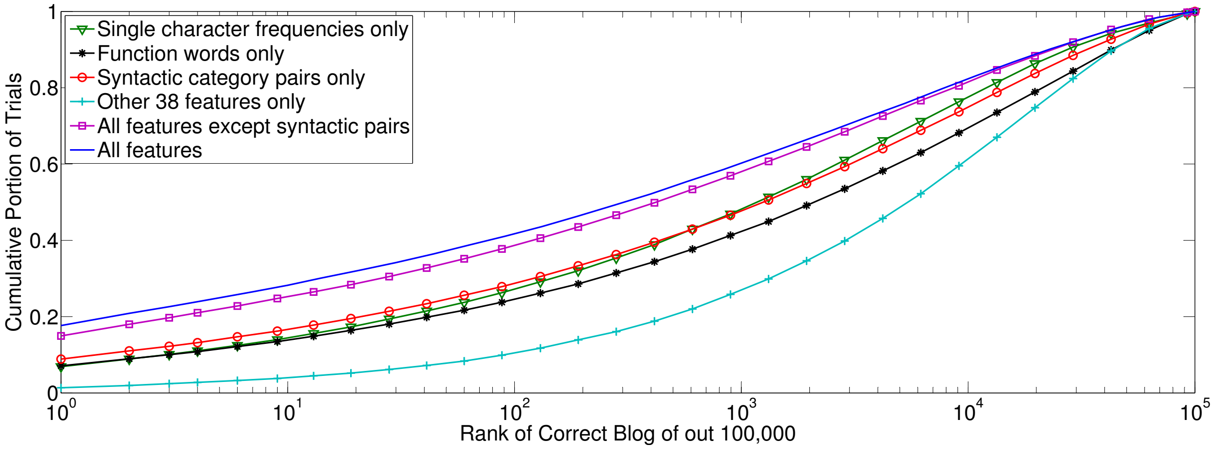

Figure 10. Post-to-blog matching using three posts and NN2 while limiting the feature set.

blog; Figure 7 displays the CDF of these rankings. NN1 refers to nearest neighbor with the normalizations row-norm and feature-mean and NN2 uses row-norm and feature-mean-nonzero. NN2+RLSC is a combination of those two classifiers using a meta-classifier as described in the previous section.

Several interesting results can be directly read off the graphs.

- SVM's accuracy drops off rapidly to the left of $rank = 100$. This provides evidence for the masking problem. In particular, SVM's accuracy when looking at only the top ranked blog is essentially zero.
- Naive Bayes and nearest neighbor with the straightforward normalization (NN1), by default perform surprisingly well for such simple methods, with the top ranked class being the correct one in about 8% of cases for Naive Bayes.
- Better normalization makes a tremendous difference for nearest neighbor: it quadruples the accuracy at $rank = 1$ and almost triples it even at $rank = 100$.[11]
- RLSC is the best individual classifier for $rank \geq 3$. It has a nearly 30% probability of placing the correct blog among the top 10 results.
- The metaclassifier using NN2 + RLSC is better than any individual classifier for small k. It picks the correct class as the top ranked match about 20% of the time, and places it within the top 20 ranks about 35% of the time.

Impact of training set and test set size. Figure 5 displays the results of the post-to-blog matching experiment when the number of unlabeled testing examples varies. At an average of 305 words, one post is similar in size to a comment on a news article or a message board post, so these results are indicative of the ability to match an isolated online posting against our blog corpus. Some blogs do not have enough data so we can only use a subset of the blogs as the number of testing examples increases beyond 3. The "Normalized" line adjusts for population bias by subtracting the difference in accuracy between each subset and all of the blogs when measured at 3 testing examples.

Our results indicate that while the identifiability of small amounts of text is markedly reduced, we still achieve a 7.5% accuracy. Furthermore, the curves appear to reach an asymptote at around 10 samples per class with an accuracy of around 25%. We believe that this accuracy is a lower bound on the performance that could be achieved with more data; with an average of 24 posts per blog, our algorithms have little data to work with once we remove 10 posts for testing.

Figure 6 shows the impact of training set size. The number of training examples is a different factor from the amount of text (in words), although of course they are highly correlated. The former seems a better predictor of performance in our experiments. One possible reason is that text split up in different posts over time allows us to capture the variance in an author's stylometric markers. This argues against methods that work by collapsing all available text into a single document.

Two points are evident from the graph: first, accuracy is poor with less than 7-10 training examples. Second, we continue making significant gains until about 40 examples or so, after which accuracy plateaus. This suggests that authors who wish to publish anonymously should consider the amount of material they have already written that appears online.

Confidence estimation Figure 9 shows the results of confidence estimation for 3 classifiers with precision traded off against recall via the metaclassifier described in Section V-C. A precision of 50% is achieved with a recall of just

[11] Although RLSC with row-norm and feature-mean isn't shown, the difference between feature-mean and feature-mean-nonzero was even more pronounced for RLSC.

under 80%, and conversely, a 50% recall gives over 80% precision!

Blog-to-blog matching. Each trial of the blog-to-blog matching experiment consisted of randomly selecting one of the blogs obtained from the Google profiles, then ranking the other 99,999 blogs based on their similarity to its posts. The rank of the other blog listed on the same Google profile was then saved as the outcome, producing the results given in Figure 8. In the (uncommon) case of more than one additional blog listed on the Google profile, the highest ranked was considered the outcome, on the grounds that an adversary would be interested in linking an anonymous blog to any material from the same author.

We only show the classifiers NB, SVM and NN1. NN2 produced little or no gains compared to NN1; we suspect that this is because the difference in normalization is mainly effective when the number of test samples is low. Unfortunately, we perfected our RLSC technique only very recently, and have not yet used it for blog-to-blog matching since it requires some analytical modifications.

We also applied our confidence estimator and obtained results similar to the post-to-blog experiments; for example, about 50% precision with 50% recall. The full results are omitted.

Impact of feature choice. To help confirm the validity of our results, we also manually inspected a small sample of the blogs that were most easily matched in each experiment, since these would be the ones most likely to contain any post signatures or other illegitimate text that might have escaped our filtering. Nothing that could significantly affect the results was found. As a further check, we ran our experiments with the NN2 classifier using only subsets of the features in order to determine whether one particular type of feature was especially crucial to its performance. The results are shown in Figure 10. We can break our features into four groups, and these are the first 4 feature sets shown in the graph. It is clear that none of the feature sets alone come close to the 18% accuracy we get when using all of the features. Using single character frequencies, function words, or syntactic category pairs alone gives about the same performance at $7-8\%$. We also see that the syntactic pair features used alongside those of [23] boost performance by 2%.

Performance. We implemented our classifiers using a variety of programming languages and libraries, including Matlab and Python/Numpy. In the end, Matlab proved to be the fastest because of its optimized linear algebra routines. Of the classifiers we experimented with, nearest-neighbor is the simplest and therefore the fastest to train and test with. Computing the class centroids (i.e., collapsing each class) takes a negligible amount of time and classifying 150,000 points from 50,000 distinct classes takes 4.5 minutes. RLSC requires substantially more processing to train, so it takes 12 minutes to train on all of the data, minus the three points that are removed for testing. After training, RLSC behaves like NN in how it makes predictions.

Finally, we present several "tricks" that lead to a dramatic speedup in running time. The most significant factor during the prediction phase is to express computations as matrix multiplications. For example, RLSC predictions are obtained by computing the inner product of all one-versus-all classifiers with each testing sample. The Euclidean distance used in nearest-neighbors can be expanded to a similar series of inner products. Once our computations are represented as matrix multiplications, we can the compute everything in large batches that make full use of the computer's hardware. This trick alone brought prediction time down from 10 hours to the aforementioned 4.5 minutes. Interestingly, we found it was several times faster to avoid sparse format representations of our data because they are not as optimized. Finally, we employ a form of dynamic programming for RLSC that decreases the number of computations dramatically. This too brings down training time from two days to 12 minutes.

VII. Limitations, future work and conclusions

Our work has a few important limitations. First, the attack is unlikely to work if the victim intentionally obfuscates their writing style. Second, while we have validated our attack in a cross-context setting (*i.e.*, two different blogs), we have not tested it in a cross-domain setting (*e.g.*, labeled text is a blog, whereas anonymous text is is an e-mail). Both these limitations are discussed in detail in Section III. Finally, our method might not meet the requirements of forensics applications where the number of authors might be very small (the classification task might even be binary) but amount of text might be very limited.

We point out three main avenues for future research in terms of sharpening our techniques. First, studying more classifiers such as regularized discriminant analysis, and fully understand the limitations and advantages of each.

Second, investigating more principled and robust approaches for extending binary classifiers to a highly multi-class setting. One important avenue is "all pairs," where every class is compared against every other class. As one might imagine, this method gives good results, but is computationally prohibitive. A "tournament" between classes using a binary classifier is a promising approach, and we have some early results in this area.

Finally, while we have demonstrated the viability of our techniques in a cross-context setting, a more thorough investigation of different classifiers is necessary, along with an analysis of the impact of training and test set size. Understanding and modeling *how* writing style is affected by context has the potential to bring major gains in accuracy.

To conclude, we have conducted the first investigation into the possibility that stylometry techniques might pose a wide-spread privacy risk by identifying authors of anonymously published content based on nothing but their style of expression. Previous work has applied similar techniques to distinguish among up to 300 authors; we consider the scenario of 100,000.

Our findings indicate these techniques remain surprisingly effective at this scale: in about 20% of trials in each experiment, the author of the sample is ranked first among all 100,000. Authors with large amounts of text already online are even more vulnerable. Even in the cases where the author of a text is not the one estimated most likely to have produced it, some risk of identification remains. Specifically, in the median case, the likely authors of a sample can be narrowed down to a set of 100–200, a reduction by a factor of 500–1000. While this alone is unlikely to produce a match with any certainty, if it is combined with another source of information, it may be enough to make the difference between an author remaining anonymous and being identified.

Importantly, our findings only represent a lower bound on the severity of this type of risk, and we can expect the development of more sophisticated techniques to worsen the situation. Useful future work to address the privacy threat would include further characterizations of the most vulnerable authors, and improved writing-style obfuscation techniques.

Acknowledgement. We would like to thank the authors of [28] for sharing the notes of their reimplementation of the Writeprints algorithm from [23] and the authors of [47] for sharing their Google profiles dataset. We are also grateful to Roberto Perdisci and anonymous reviewers for helpful comments.

References

[1] A. Furtwangler, *The Authority of Publius: A Reading of the Federalist Papers*. Cornell University Press, 1984.

[2] Supreme Court of the United States, "McIntyre vs. Ohio Elections Commission, 514 U.S. 334," 1995.

[3] M. L. Sifry, *WikiLeaks and the age of transparency*. Counterpoint, 2011.

[4] Mike Giglio, "The Facebook Freedom Fighter," *Newsweek*, Feb. 2011.

[5] J. Bone, "Vogue model Liskula Cohen wins right to unmask offensive blogger," *The Times*, Aug. 2009.

[6] J. Brumley, "Unmasked blogger blames First Baptist, sheriff's office," *The Florida Times-Union*, Apr. 2009.

[7] M. E. Larios, "ePublius: Anonymous Speech Rights Online," *Rutgers Law Record, Vol. 37, p. 36, 2010*, 2010.

[8] G. H. Pike, "The Right to Remain Anonymous," *Information Today, Vol. 25, No. 4, p. 17, April 2008*, 2008.

[9] J. Mayer, "Any person... a pamphleteer: Internet Anonymity in the Age of Web 2.0," *Undergraduate Senior Thesis, Princeton University*, 2009.

[10] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *USENIX Security Symposium*, 2004.

[11] M. Koppel, J. Schler, and S. Argamon, "Authorship attribution in the wild," *Language Resources and Evaluation*, vol. 45, no. 1, pp. 83–94, 2011.

[12] K. Luyckx and W. Daelemans, "The effect of author set size and data size in authorship attribution," *Literary and Linguistic Computing*, vol. 26, no. 1, pp. 35–55, 2011.

[13] H. Paskov, "A regularization framework for active learning from imbalanced data," Master's thesis, MIT, 2008.

[14] C. Rubin, "Surprised Employer Fires Sex Blogger," *Inc. Magazine*, May 2010.

[15] M. Schwartz, "The Troubles of Korea's Influential Economic Pundit," *WIRED*, Oct. 2009.

[16] M. Brennan and R. Greenstadt, "Practical attacks against authorship recognition techniques," in *IAAI*, 2009.

[17] G. Kacmarcik and M. Gamon, "Obfuscating document stylometry to preserve author anonymity," in *The Association for Computer Linguistics*, 2006.

[18] T. C. Mendenhall, "The characteristic curves of composition," *Science*, vol. ns-9, no. 214S, pp. 237–246, 1887. [Online]. Available: http://www.sciencemag.org/content/ns-9/214S/237.short

[19] ——, "A mechanical solution of a literary problem," *Popular Science Monthly*, vol. 60, no. 2, pp. 97–105, 1901.

[20] F. Mosteller and D. L. Wallace, *Inference and Disputed Authorship: The Federalist*. Addison-Wesley, 1964.

[21] E. Stamatatos, "A survey of modern authorship attribution methods," *J. Am. Soc. Inf. Sci. Technol.*, vol. 60, pp. 538–556, March 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1527090.1527102

[22] D. Madigan, A. Genkin, D. D. Lewis, S. Argamon, D. Fradkin, and L. Ye, "Author identification on the large scale," in *Joint Meeting of the Interface and Classification Society of North America*, 2005.

[23] A. Abbasi and H. Chen, "Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace," *ACM Transactions on Information Systems*, vol. 26, no. 2, Mar. 2008.

[24] O. Y. de Vel, A. Anderson, M. Corney, and G. M. Mohay, "Mining email content for author identification forensics," *SIGMOD Record*, vol. 30, no. 4, pp. 55–64, 2001.

[25] C. E. Chaski, "Who's at the keyboard? authorship attribution in digital evidence investigations," *IJDE*, vol. 4, no. 1, 2005.

[26] ——, "The Keyboard Dilemma and Authorship Identification," in *IFIP Int. Conf. Digital Forensics*, 2007, pp. 133–146.

[27] D. I. Holmes, “A stylometric analysis of mormon scripture and related texts,” *Journal of the Royal Statistical Society Series A Statistics in Society*, vol. 155, no. 1, pp. 91–120, 1992. [Online]. Available: http://www.jstor.org/stable/2982671

[28] S. Afroz, M. Brennan, and R. Greenstadt, “Detecting Hoaxes, Frauds, and Deception in Writing Style Online,” in *IEEE Symposium on Security and Privacy*, 2012.

[29] E. Stamatatos, N. Fakotakis, and G. Kokkinakis, “Text genre detection using common word frequencies,” in *Proceedings of the 18th conference on Computational linguistics - Volume 2*, ser. COLING ’00. Stroudsburg, PA, USA: Association for Computational Linguistics, 2000, pp. 808–814. [Online]. Available: http://dx.doi.org/10.3115/992730.992763

[30] M. Koppel, J. Schler, and S. Argamon, “Computational methods in authorship attribution,” *JASIST*, vol. 60, no. 1, pp. 9–26, 2009.

[31] E. Backer and P. van Kranenburg, “On musical stylometry–a pattern recognition approach,” pp. 299 – 309, 2005, in Memoriam: Azriel Rosenfeld. [Online]. Available: http://www.sciencedirect.com/science/article/B6V15-4F9MTM7-2/2/b5a4da4e8d5c8cb34d6f1a097a0483c4

[32] M. Shevertalov, J. Kothari, E. Stehle, and S. Mancoridis, “On the use of discretized source code metrics for author identification,” in *International Symposium on Search Based Software Engineering*, 2009.

[33] W.-J. Li, K. Wang, S. J. Stolfo, and B. Herzog, “Fileprints: identifying file types by n-gram analysis,” in *IEEE Systems, Man, and Cybernetics Information Assurance Workshop*, 2005.

[34] L. Zhuang, F. Zhou, and J. D. Tygar, “Keyboard acoustic emanations revisited,” in *ACM conference on Computer and communications security*, 2005.

[35] P. Juola, “Authorship attribution,” *Foundations and Trends in Information Retrieval*, vol. 1, pp. 233–334, December 2006. [Online]. Available: http://portal.acm.org/citation.cfm?id=1373450.1373451

[36] C. E. Chaski, “Empirical evaluations of language-based author identification techniques,” *Forensic Linguistics*, vol. 8, no. 1, pp. 1–65, 2001. [Online]. Available: http://www.extenza-eps.com/EPL/doi/abs/10.1558/sll.2001.8.1.1

[37] J. Rudman, “The State of Authorship Attribution Studies: Some Problems and Solutions,” *Computers and the Humanities*, vol. 31, no. 4, pp. 351–365, 1998. [Online]. Available: http://www.springerlink.com/index/L023Q7047388133X.pdf

[38] H. Maurer, F. Kappe, and B. Zaka, “Plagiarism - a survey,” *Journal of Universal Computer Science*, vol. 12, no. 8, pp. 1050–1084, 2006.

[39] J. R. Rao and P. Rohatgi, “Can pseudonymity really guarantee privacy?” in *Proceedings of the 9th conference on USENIX Security Symposium - Volume 9*. Berkeley, CA, USA: USENIX Association, 2000, pp. 7–7. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251306.1251313

[40] M. Koppel, J. Schler, S. Argamon, and E. Messeri, “Authorship attribution with thousands of candidate authors,” in *SIGIR*, 2006, pp. 659–660.

[41] M. Nanavati, N. Taylor, W. Aiello, and A. Warfield, “Herbert west: deanonymizer,” in *Proceedings of the 6th USENIX conference on Hot topics in security*, ser. HotSec’11. Berkeley, CA, USA: USENIX Association, 2011, pp. 6–6. [Online]. Available: http://dl.acm.org/citation.cfm?id=2028040.2028046

[42] S. Hill and F. J. Provost, “The myth of the double-blind review? Author identification using only citations,” *SIGKDD Explorations*, vol. 5, no. 2, pp. 179–184, 2003.

[43] J. K. Bradley, P. G. Kelley, and A. Roth, “Author identification from citations,” Tech. Rep., Dec. 03 2008.

[44] P. Eckersley, “How unique is your web browser?” Electronic Frontier Foundation, Tech. Rep., 2009. [Online]. Available: https://panopticlick.eff.org/browser-uniqueness.pdf

[45] G. Wondracek, T. Holz, E. Kirda, and C. Kruegel, “A practical attack to de-anonymize social network users,” in *IEEE Symposium on Security and Privacy*, 2010.

[46] D. Irani, S. Webb, K. Li, and C. Pu, “Large online social footprints–an emerging threat,” in *International Conference on Computational Science and Engineering*, 2009.

[47] D. Perito, C. Castelluccia, M. A. Kâafar, and P. Manils, “How unique and traceable are usernames?” in *PETS*, 2011, pp. 1–17.

[48] A. K. Jain, A. Ross, and S. Prabhakar, “An introduction to biometric recognition,” *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 14, pp. 4–20, 2004.

[49] L. Wang, T. Tan, S. Member, H. Ning, and W. Hu, “Silhouette analysis-based gait recognition for human identification,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 12, pp. 1505–1518, 2003. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1251144

[50] R. Plamondon and S. N. Srihari, “On-line and off-line handwriting recognition: A comprehensive survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 63–84, January 2000. [Online]. Available: http://dl.acm.org/citation.cfm?id=331097.331275

[51] F. Monrose and A. D. Rubin, “Keystroke dynamics as a biometric for authentication,” *Future Generation Computer Systems*, vol. 16, no. 4, pp. 351–359, 2000. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167739X9900059X

[52] L. Sweeney, “k-anonymity: A model for protecting privacy,” *Ieee Security And Privacy*, vol. 10, no. 5, pp. 557–570, 2002. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.163.9182&rep=rep1&type=pdf

[53] A. Narayanan and V. Shmatikov, “Robust de-anonymization of large sparse datasets,” in *IEEE Symposium on Security and Privacy*, 2008, pp. 111–125.

[54] R. Leiby, "The Hill's sex diarist reveals all (well, some)," *The Washington Post*, May 204.

[55] K. Burton, A. Java, and I. Soboroff, "The ICWSM 2009 Spinn3r dataset," in *International AAAI conference on weblogs and social media*, 2009.

[56] D. Klein and C. D. Manning, "Accurate unlexicalized parsing," in *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, 2003, pp. 423–430.

[57] H. Baayen, H. van Halteren, and F. Tweedie, "Outside the cave of shadows: using syntactic annotation to enhance authorship attribution," *Lit Linguist Computing*, vol. 11, no. 3, pp. 121–132, Sep. 1996.

[58] M. Gamon, "Linguistic correlates of style: authorship classification with deep linguistic analysis features," in *Proceedings of the 20th international conference on Computational Linguistics*, ser. COLING '04. Stroudsburg, PA, USA: Association for Computational Linguistics, 2004. [Online]. Available: http://dx.doi.org/10.3115/1220355.1220443

[59] G. Forman, "An extensive empirical study of feature selection metrics for text classification," Hewlett-Packard Labs, Tech. Rep., 2002.

[60] I. G. Bernhard E. Boser and V. Vapnik, "A training algorithm for optimal margin classifiers," in *COLT*, 1992, pp. 144–152.

[61] C. Cortes and V. Vapnik, "Support-vector networks," in *Machine Learning*, vol. 20, no. 3, 1995, pp. 273–297.

[62] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods - Support Vector Learning*, 1999.

APPENDIX A. EXAMPLES OF MATCHED BLOGS

We present three pairs of blogs where one blog was correctly matched to the other (from among 100,000) in our blog-to-blog experiments. In the first pair, one blog is about personal life while the other discusses social, economic and educational issues. Within our crawled data, a post in the first blog talks about the blogger's love-hate relationship with Walmart, whereas a post in the second blog involves teacher layoffs and school funding. For the second pair, one blog covers travelling while the other describes interesting events involving the blogger. For the third pair, one blog talks about God while the other discusses pieces of the blogger's life.
First pair:
http://thoughtsonlll.blogspot.com
http://toourleaders.blogspot.com
Second pair:
http://still-wanderings.blogspot.com
http://heatherandavaga.blogspot.com
Third pair:
http://livingbythewordofgod.blogspot.com
http://lsfarnsworths.blogspot.com

APPENDIX B. CHALLENGES IN SCALING LDA

One of the most important normalizations we use is to divide each dimension by the average value of its support. LDA amounts to computing $(x-\mu)^T C^{-1}(x-\mu)$ where x is a point to be labelled, μ a class centroid, and C a covariance matrix. We can normalize the dimensions of a point by computing $x = Dz$ where D is a diagonal matrix whose entries are the normalizations. Recalling that C is the sum of the outer products of all training examples, it follows that $C = DKD$ where K is the covariance matrix we obtain without normalization. Similarly, $\mu = D\gamma$ so that after plugging in, our LDA prediction is

$$(z-\gamma)^T D(DKD)^{-1}D(z-\gamma) = (z-\gamma)^T K^{-1}(z-\gamma)$$

cancelling out any normalization.

APPENDIX C. FUNCTION WORDS

Below is a list of the function words that we used. It appears to be generally agreed that there are about 300 function words in English; we arrived at this list by merging several lists available online so as to bring the number as close to 300 as possible.

a, able, aboard, about, above, absent, according, accordingly, across, after, against, ahead, albeit, all, along, alongside, although, am, amid, amidst, among, amongst, amount, an, and, another, anti, any, anybody, anyone, anything, are, around, as, aside, astraddle, astride, at, away, bar, barring, be, because, been, before, behind, being, below, beneath, beside, besides, better, between, beyond, bit, both, but, by, can, certain, circa, close, concerning, consequently, considering, could, couple, dare, deal, despite, down, due, during, each, eight, eighth, either, enough, every, everybody, everyone, everything, except, excepting, excluding, failing, few, fewer, fifth, first, five, following, for, four, fourth, from, front, given, good, great, had, half, have, he, heaps, hence, her, hers, herself, him, himself, his, however, i, if, in, including, inside, instead, into, is, it, its, itself, keeping, lack, less, like, little, loads, lots, majority, many, masses, may, me, might, mine, minority, minus, more, most, much, must, my, myself, near, need, neither, nevertheless, next, nine, ninth, no, nobody, none, nor, nothing, notwithstanding, number, numbers, of, off, on, once, one, onto, opposite, or, other, ought, our, ours, ourselves, out, outside, over, part, past, pending, per, pertaining, place, plenty, plethora, plus, quantities, quantity, quarter, regarding, remainder, respecting, rest, round, save, saving, second, seven, seventh, several, shall, she, should, similar, since, six, sixth, so, some, somebody, someone, something, spite, such, ten, tenth, than, thanks, that, the, their, theirs, them, themselves, then, thence, therefore, these, they, third, this, those, though, three, through, throughout, thru, thus, till, time, to, tons, top, toward, towards, two, under, underneath, unless, unlike, until, unto, up, upon, us, used, various, versus, via, view, wanting, was, we, were, what, whatever, when, whenever, where, whereas, wherever, whether, which, whichever, while, whilst, who, whoever, whole, whom, whomever, whose, will, with, within, without, would, yet, you, your, yours, yourself, yourselves