Authorship Verification

A Thesis

Submitted to the Faculty

of

Drexel University

by

Ariel Stolerman

in partial fulfillment of the

requirements for the degree

of

Doctor of Philosophy in Computer Science

April 2015

Drexel
UNIVERSITY

Office of Graduate Studies

Dissertation/Thesis Approval Form

This form is for use by all doctoral and master's students with a dissertation/thesis requirement. Please print clearly as the library will bind a copy of this form with each copy of the dissertation/thesis. All doctoral dissertations must conform to university format requirements, which is the responsibility of the student and supervising professor. Students should obtain a copy of the Thesis Manual located on the library website.

Dissertation/Thesis Title: Authorship Verification

Author: Ariel Stolerman

This dissertation/thesis is hereby accepted and approved.

Signatures:

Examining Committee

Chair

Members

Academic Advisor

Department Head

Acknowledgements

First and foremost, I would like to show my deepest gratitude to my advisor, Dr. Rachel Greenstadt, for her guidance from the moment I began this journey. Her experience, direction, passion for research and investment were crucial for my academic, professional and personal growth during this period in my life. Rachel had the rare ability to identify my strengths, which she encouraged, and weaknesses, for which she provided me with guidance and constructive criticism that helped me tremendously along the way. She knew to let me navigate on my own path, giving nudges and correcting it exactly when needed. I am lucky to have had the opportunity to work with Rachel and be mentored by her, deeply honored to have her as my doctoral committee chair, and forever grateful for her guidance and support.

I would also like to extend my sincere gratitude to the other esteemed members of my doctoral committee, Dr. Patrick Juola from Duquesne University, Dr. Santiago Ontañón Villar, Dr. Dario Salvucci and Dr. Ali Shokoufandeh from Drexel University. I am honored to have you in my committee.

I would like to give special thanks to my senior, long-graduated labmates, Dr. Michael Brennan and Dr. Sadia Afroz. Along with Rachel, they have been the pioneers at the Privacy, Security and Automation lab at Drexel, and paved the way for me and my graduate fellows to fascinating areas of computational linguistics research. They have been my mentors in many ways, inspired me greatly, and for that I am truly grateful to them.

I would also like to extend my thanks to my other past and present labmates at PSAL – Aylin Çalişkan İslam, Andrew McDonald, Rebekah Overdorf, Pavan Kantharaju, Jeffrey Segall, Travis Dutko, and the other members of the lab. With all I have shared some of the best parts of this journey – fruitful discussions, conference prep-talks, last-minute paper submissions, and even class assignments. Without you, this period would not have been the same. Finally, I would like to show my gratitude to all the bright and talented supporting authors of my work, and those whom I had the pleasure and privilege of being a supporting author of their work and have not mentioned yet above: John I. Noecker Jr., Michael V. Ryan and Patrick Brennan from Juola & Associates; Dr. Alex (Lex) Fridman, Sayandeep Acharya and Dr. Moshe Kam from Drexel University; and Dr. Damon McCoy from George Mason University.

Last but certainly not least, I would like to thank my family and friends, who have been there every step of the way. Special gratitude to my parents-in-law, Nili and Zelig Tochner, for their

moral and practical support along the road, from the early stages of admissions at schools in a foreign country to the final stages of writing my dissertation. To my mother, Paula Stolerman, and my sister Dana Stolerman-Taggart and her family, for supporting this journey despite the distance from home it had imposed.

Finally, to my lovely wife whom I love dearly, Yael Tochner, who supported and believed in me during this experience. Thank you for opening me up to a great world of opportunities, and accompanying me through this one.

Dedications

To Dr. Zelig A. Tochner, my father-in-law, the igniter, motivator and primary supporter of my pursuit of a doctorate degree. You are a role model of excellence, perseverance and self-growth, and I am forever thankful for your guidance.

To my beloved father, Michael Stolerman, who has passed away before I embarked on this journey. Your love and warmth still accompany me wherever I am, and you will always be in my heart. I know I would have made you proud.

Table of Contents

List of Tables

List of Figures

Abstract

Authorship Verification

Ariel Stolerman
Advisor: Rachel Greenstadt, PhD

In recent years, stylometry, the study of linguistic style, has become more prominent in security and privacy applications involving written language, mostly in digital and online domains. Although literature is abundant with computational stylometry research, the field of authorship verification is relatively unexplored. Authorship verification is the binary semi-open-world problem of determining whether a document is written by a given author or not. A key component in authorship verification techniques is *confidence* measurement, on which verification decisions are based, expressed by acceptance thresholds selected and tuned per need. This thesis demonstrates how utilization of confidence-based approaches in stylometric applications, and their combination with traditional approaches, can benefit classification accuracy, and allow new domains and problems to be analyzed. We start by motivating the usage of authorship verification approaches with two stylometric applications: native-language identification from non-native text and active linguistic user authentication. Next, we introduce the *Classify-Verify* algorithm, which integrates classification with binary verification, applied to several stylometric problems. *Classify-Verify* is proposed as an open-world alternative to restricted closed-world attribution methods, and is shown effective in dealing with possibly missing candidate authors by thwarting misclassifications, coping with various domains and scales, and even adversarial authors who try to fool the classifier.

1. Introduction

Stylometry is the application of authorship attribution using linguistic style learned from text. Stylometry has existed for centuries, with historical, literary and forensic applications [55]. Perhaps the most famous historical application of stylometry is in the case of the 12 disputed Federalist Papers, whose authorship was believed to belong to either James Madison or Alexander Hamilton; this case became a popular dataset for stylometry research [47, 75, 79, 94, 108]. This example is but an illustration of how stylometry has become dominated by computational methods in the last decades, specifically artificial intelligence applications involving natural language processing for quantifying writing style and machine learning techniques for learning and classification. An abundance of literature of stylometric techniques has accumulated over the years, showing constant increase in accuracy and scale [1, 80], applied in legal contexts [24], security [57], plagiarism detection applications [109] and more.

The classic and most common approach for stylometric analysis is closed-world supervised learning: classify an anonymous document to one author out of a known, closed set of authors. When it is indeed the case, stylometric techniques can reach a high level of accuracy. Unfortunately, realistic scenarios, involving problems in online domains or other digital open-ended realms, do not conform to the ideal problem structures these analysis methodologies are designed for. In such cases there is a need for trustworthy, semi-supervised, open-world techniques, namely authorship verification methods.

Authorship verification in the context of computational stylometry is the application of linguistic style learning to detecting whether a given document is written by a given author or not. As research in computational stylometry is plentiful and continues to accumulate, this one-class classification variant of stylometry is relatively unexplored. The assumption with authorship verification problems is that we have prior information to model the style of some candidate author; however, we have little to no information about other possible candidates, making this an open-world problem.

Authorship verification approaches, in a more high-level perspective, are in fact classification techniques that rely on *confidence* measurements. Confidence takes a great role in verification, usually expressed in a verification threshold that determines the rigidity of the authorship verifier in hand, an approach naturally suited for dealing with the uncertainties in open-world scenarios. Verification-driven approaches are thus preferred over the standard closed-world supervised ones where there is great importance to the confidence that accompanies the classification: how sure are we in the given results.

In this thesis, we demonstrate how utilization of authorship verification approaches can increase the attained accuracy and confidence in classification problems, applied in various stylometric domains. Specifically, we show how the combination of verification approaches with traditional closed-world ones increases the performance of such techniques, for both closed-world domain problems, and open-world areas unsuitable for traditional approaches alone. This work demonstrates how authorship verification can be used effectively by laying out configurations, features and algorithms employed over multiple applications, targeting different scenarios in the domain of stylometry problems. The ideas and methods presented in this work fall under three main aspects of authorship verification research, as described next.

I. Generalization and Problem Relaxation for Improved Classification

This work demonstrates how to utilize information that lies in the categorization of the data we aim to classify, using verification approaches along with traditional classifiers in order to improve classification accuracy. In certain problems we may have the option to generalize or relax the analysis, adjusting the tradeoff between the problem granularity and the potential accuracy and confidence measurements: know less, but with higher certainty. Confidence-based verification methods can be used as indicators of when generalizing or relaxing the problem in hand is preferable over finer classification granularity, in order to attain a higher overall classification accuracy.

In Ch. 4 we examine the problem of native-language identification from second-language text. We show how the hierarchical taxonomy of the domain of languages along with verification techniques are exploited in a closed-world classification process for instances with low classification confidence, in order to generalize the question to identifying a broader class – the language family – rather than the language itself. Thus, less knowledge is attained, but with higher certainty. We further develop our technique to a 2-step classification process where verification is used to indicate when narrowing down the search domain is required, leading to closed-world classification corrections and an overall higher native-language identification accuracy.

Our native-language identification algorithm demonstrates how authorship verification approaches can be successfully interleaved with a closed-world, traditional classification process in order to improve its overall accuracy, taking advantage of a broader categorization of the domain.

II. Stylometry-Based Security Applications

Authorship attribution has the ability to introduce high-level analysis that can be utilized for authentication and identification in scenarios where other modalities may fail. For instance, anonymous blog posts whose authors hide behind forged identities and fake technical characteristics (e.g. source IP address) can still be analyzed for authorship, and crossed with sets of potential suspects. Using stylometry in authentication systems can utilize confidence-driven approaches in order to adjust the desired tradeoff between susceptibility to attacks and the rate of false alarms, making authorship verification the natural approach to take in such instances. Whereas traditional closed-world stylometry is limited to distinguishing between entities known to the identifying learning agent, verification techniques are not bounded to a limited set of candidates, and can successfully catch culprits outside the set known to the security system.

In Ch. 5 we focus on the active authentication problem, where user input is constantly monitored by a learning system in order to identify if the user at the keyboard is the legitimate one, or has its session been hijacked by an adversary. This work defines and evaluates a series of configurations, consisting of sampling, learning and classification methodologies, and lays out conclusions concerning the design of such security systems. We begin performing our evaluations using closed-world models, and revisit the different configurations in Ch. 6, demonstrating the effectiveness of using authorship verification techniques combined with closed-world methods for security applications like active authentication.

III. Open-World Settings

The most challenging settings for stylometric methods are those where the true authors may be missing from the set of known candidates, vast amounts of potential authors exist, and training data is not necessarily abundant for ideal analysis; those settings are found in open-world domains, specifically online. In these settings, traditional closed-world approaches fail miserably, as they are not designed to take such scenarios into account. In their place we use authorship verification techniques, which are naturally formulated for such scenarios.

Verification methods shed light on the confidence level derived from the algorithm of choice, and with it the level of rigidity and acceptance of the algorithm's decisions, thus allowing catching instances where the true author may be missing, or simply a wrong attribution has been made. In such settings, often the case in online domains, we want to avoid mistakenly attributing an innocent author to a document, simply because of its similar style to that of the true, missing author.

In Ch. 6 we demonstrate how utilizing verification-driven approaches that allow tuning the desired confidence are well-accustomed for the difficult open-world scenarios mentioned above. We present the *Classify-Verify* algorithm, which shows how the synergy between verification approaches along with traditional closed-world methods exploits the best of both worlds. The algorithm's effectiveness is demonstrated in various settings and domains, including different types of writings and textual sources, varying number of known candidate authors, different settings of assurance in the completeness of the set of known candidates, and adversarial settings where authors attempt to fool the classifiers. In all of the above, *Classify-Verify* outperforms traditional classifiers and thus is given as an alternative that improves precision and accuracy in both open-world and closed-world domains.

Key Contributions

Through examination of the problems above, this work demonstrates how effective authorship verification is, useful to attain improved confidence and accuracy, crossing various domains and scenarios. The questioning of classification results inherited in verification methods is shown useful when combined with traditional methods, allowing new applications like biometric authentication, better accuracy in common closed-world settings, and open-world domains where traditional methods alone were not useful before. This thesis strives to elevate verification-infused classification as the preferable approach for problems in textual and linguistic data analysis domains, both closed-world and open-world alike.

1.1 Thesis Organization

Ch. 2 reviews the current state of authorship verification research, including problem definitions, general stylometry and one-class classification approaches. Ch. 3 presents JStylo, an open-source authorship attribution platform designed for stylometric analysis and research.

In Ch. 4 we utilize verification to identify native-language and language family of the authors from non-native texts, demonstrating problem-relaxation and generalization using authorship verification, used for improving closed-world classification. In Ch. 5 we analyze stylometry applied in active linguistic authentication settings, demonstrating how authorship attribution can be utilized for security systems, motivating the utilization of verification-based approaches in such applications.

In Ch. 6, the main chapter of this thesis, we present the *Classify-Verify* algorithm, a mixed closed/open-world approach that utilizes both authorship attribution and verification techniques for open-world stylometric analysis. The *Classify-Verify* method is applied in varying scenarios including adversarial settings, online domains and the active authentication settings presented in Ch. 5, demonstrating the effectiveness of integrating verification approaches in stylometric analysis across various problems and domains.

Chapter-specific conclusions and directions for future work are discussed throughout the document. In addition, final conclusions and suggestions for prospective research are discussed in Ch. 7.

2. Background

Authorship verification in the context of computational stylometry is the application of linguistic style learning to detecting whether a given document is written by a given author or not. As research in computational stylometry is plentiful [55] and continues to accumulate, the one-class classification problem of authorship verification is explored relatively little. However, with the increase in online communication and digital information, applications such as plagiarism detection [109] and security [57] raise the need for trustworthy authorship verification techniques.

The assumption with authorship verification problems is that we have prior information to model the style of a candidate author (and of course the document in question); we do not, however, have any other information about other possible candidates, making this an open-world problem. That is, we are not required to choose from a closed set of potential authors of the document, but rather have to determine "yes" or "no" for one given author. These one-sided settings define the problem as a one-class classification problem [104], and harden the problem greatly as opposed to a standard binary/multinary classification task.

This chapter reviews the current state of authorship verification. The remainder of the chapter is structured as follows: In Sec. 2.1 we discuss one-class classification approaches, as a basis for the stylometry variant, namely authorship verification. Sec. 2.2 lays out methodologies in computational stylometry, including problem formulation, feature selection and learning approaches. Sec. 2.3 reviews the current state of authorship verification research. Finally, Sec. 2.4 discusses open problems and possible directions for authorship verification research, followed by conclusions in Sec. 2.5.

2.1 One-Class Classification

The one-class classification problem [104] is the problem of distinguishing one class of objects from all others, given training data only for the target class. As opposed to binary [1] classification problems, here a boundary in the space of the objects of interest has to be inferred only from "good" samples. We aspire to define a boundary that contains all of the target class objects, and does not contain any outliers.

The one-class classification problem appears in the literature also as outlier detection [91], novelty detection [14] and concept learning [50], terms which indicate different applications of one-class clas-

[1] We can ignore the more general n-ary case, as any multiclass problem can be reduced to a set of binary problems.

sification. Outlier detection is probably the most common application, used to identify irregularities in a dataset, compensate for an undersampled class in a binary problem by detecting outliers with respect to the other class, or comparing two data sets (e.g. to decide whether a classifier trained on old data needs to be retrained on a new one).

2.1.1 One-Class and Two-Class Classification Problems

When comparing one-class and two-class classification problems, the set of problems in the latter are also relevant in the first, including error definition, solution complexity measurement, the curse of dimensionality [32] and others, where some even become more outstanding [104]. In one-class problems, a boundary between the target class and the rest of the world is determined by one-sided data, making it harder to determine how tight the boundary should be compared to binary problems, where data to support both sides of the boundary is available (where usually equally balanced classes are assumed). The central problem of feature selection that distinguishes well between the target class and outliers also becomes harder.

From an error measurement point of view, because we rely solely on data from the target class, we can minimize (in a controlled fashion) only the number of target objects erroneously classified as outliers, namely false negatives or type II errors (this can be achieved trivially by accepting always.) Two-class problems, on the other hand, contain probability information for all classes, allowing control of type I errors as well, namely false positives. In outlier detection terms, this would have translated to reducing outliers classified as the target class. In order to cope with this lack of information, assumptions of the outlier data distribution have to be made.

Another difficulty that rises with one-class problems is the need for a closed boundary. This requirement, which may have negative effects on two-class classifiers, hardens the problem and affects other problems as well, like the curse of dimensionality, due to the boundary required to be defined in all directions around the target class data. This causes the required sample size to often be larger for one-class problems, compared with the other.

2.1.2 Methods

One method that suggests itself for one-class classification is reducing the problem to binary classification by generating outliers around the target sample set [61, 92]. This method suffers from two problems. First, it requires outliers close to the target. In addition, it scales poorly in high dimensions, especially if the outlier data is required to be artificially generated.

Outlier detection in classification and regression problems can be performed using a Bayesian approach [15, 93]. The probabilities that the classifier weights are correct given the data are used to weigh the output, to restrain classifier outputs for objects distant from the target samples. These methods are fit for outlier detection in classification and regression problems, however not for straight forward outlier detection where only target samples are available.

Although different approaches exist for one-class classification, three sets of methods encompass a wide range of models: density estimators, reconstruction methods and boundary methods [104]. The approaches differ in how characteristics of the data are handled or exploited, such as grouping or feature scalability. The commonalities of all one-class methods are the use of a distance measurement or a resemblance probability of an object to the target class, along with corresponding thresholds (objects are identified as in the target class when the distance is below, or resemblance is above, some threshold.) In addition, an important performance measurement of all one-class methods is the tradeoff between the (fraction of) accepted target objects and rejected outliers, expressed in the ratio of type I and type II error rates, commonly illustrated with Receiver Operating Characteristic (ROC) curves [78].

Additional properties of one-class methods to be considered include: robustness to outliers contaminating the training set; flexibility that allows incorporating known outliers in probability estimates; ease of configuration – number and meaningfulness of the required parameters (like number of hidden layers in neural networks); and computation and memory requirements, especially for the classification phase (as training is mostly done offline). Next we summarize the three main approaches mentioned above. Technical and mathematical details are excluded; those can be found in the referenced articles.

Density Methods

Density estimators assume the outlier data is uniformly distributed, and directly estimate the probability distributions of the target class features. Applying a threshold on the distribution with Bayes rule can differentiate between target objects and outliers. Literature includes approaches where different density models are used to estimate the target class [14, 91]. This approach requires a complete density estimate, and thus large datasets for high dimensional problems. In addition, it assumes the data is a typical sample of the true distribution, which is often false. However, when it is the case, the method is expected to perform well.

The simplest model is Gaussian (normal) density [15], which is insensitive to scaling of the data, has almost no “magic parameters” to be set empirically, and if normal distribution of the data is assumed, we can compute the optimal threshold for any desired type II error rate (however normality of the data is hardly ever the case.) To relax the strict assumptions about the distribution of the data, a mixture of Gaussians [32] model (MoG) can be used. It is a more flexible model, however requires much more data (otherwise it suffers from high variance).

The Parzen density estimation [85] is an extension of MoG, which assumes the features are equally weighted, and therefore sensitive to feature scaling. This is a non-parametric model with no “magic parameters” to set. In addition, training the model is computationally inexpensive, but testing is – making this model less applicable with a large dataset and high dimensional feature space.

Boundary Methods

Boundary methods focus on the definition of a boundary around the target set. These methods avoid estimation of the complete density of the data; instead, only samples of the boundary of the data are required. This approach allows learning from the data when the target density is absent, and when only a small sample set is available. However, boundary methods rely on distance measurements between objects in the target class, which makes them sensitive to feature scalability.

The k-centers method [117] covers the target dataset with k balls placed on the training objects in a fashion that minimizes the maxmin distances between all target objects and the k centers. This method is sensitive to in-set outliers, but works well when these do not exist.

In the (k) Nearest Neighbors method [32], denoted NN-d, a cell (sphere) is expanded centered around the test object until it captures k target objects. A local density measurement is then derived, and the test object is accepted if its local density is larger or equal to that of its first neighbor. Using $k = 1$ would result with calculating the densities of the boundary alone. Known variants like the Local Outlier Factor (LOF) differ by the distance function used or using averaged distances across all k neighbors. Another feature of NN-d is that it may reject regions that are contained in the target distribution. Lastly, the basic NN-d which uses $k = 1$ has no parameters to set, but it is scale sensitive due to the direct usage of distance measurements.

Support Vector Data Description [111] (SVDD) is targeted to directly obtain the boundary around the target data. This method allows efficient mapping of the data to high dimensional space (done implicitly using kernels [112]) to attain a more flexible data description, and outlier sensitivity is controlled more flexibly. In addition, outlier information can be incorporated to better the data description. One-class support vector machines are discussed in [74, 96].

Reconstruction Methods

Reconstruction methods make use of prior knowledge and assumptions about the generation process to fit a model to the data. The object measurements are reconstructed from the model, and the reconstruction error is then used to measure how well the objects fit the model, under the hypothesis that the larger the error is, the more likely it is an outlier. It follows that these methods work well only when the model fits the data well, and outliers do not satisfy the assumptions about the target distribution.

In k-means clustering [15] and Learning Vector Quantization [21] (LVQ) clustering of the data is assumed which can be characterized by prototype objects that impose a Voronoi decomposition of the space. The important difference between the k-means method and the k-center method from the previous section is that it is more robust to outliers on the expense of accepting all target objects. LVQ is a supervised variant of k-means (adding cluster labels). In Self-Organizing Maps [62] (SOM), the prototype placing is constrained to form a low dimensional manifold. These methods are all scale sensitive due to using Euclidean distance in the error definitions.

Principal Component Analysis [15] (PCA), also known as Karhunen-Loève transforms, are used to project objects onto a lower dimensional space. The mapping is optimized to maximize the variance of the projected feature vectors. If a clear linear subspace is available, this method works well. Since feature scaling affects their variance, PCA is scale sensitive. Moreover, since it is focused on variances, the training cannot include outlier information. PCA can be extended to a mixture of PCAs, where each PCA has a different basis and therefore subspace. This method requires a large sample set, it is sensitive to outliers, however it is scale insensitive.

Auto-encoders [51] and diabolo networks are types of neural networks, trained to reconstruct inputs at the output layer. The training process aims to minimize the mean squared error, and it is hypothesized that target objects are reproduced with smaller error than outliers are. These methods, like any neural network, necessitate defining magic parameters such as learning rate and stopping criterion, however they can be optimized and thus perform very well.

2.1.3 Towards Authorship Verification

In this section we reviewed common, non-domain-specific, one-class classification methods. Authorship verification is the application of one-class classification methods to stylometric datasets, i.e. training models using stylistic features extracted from text of a single target author, with no (or very little) knowledge of outliers – texts of other candidate authors. Sec. 2.3 discusses several prominent authorship verification algorithms, which can be mapped back to methods presented in this section. But first, another scope of generalization is presented in the following section – the domain of style based authorship attribution problems, namely stylometry.

2.2 Stylometry

Stylometry is the application of authorship attribution using linguistic style found in text. Stylometry has existed for centuries, with historical, literary and forensic applications [55]. Perhaps the most famous historical application of stylometry is in the case of the 12 disputed Federalist Papers, whose authorship was believed to belong to either James Madison or Alexander Hamilton; this case became a popular dataset for stylometry research, some of which can be found in [47, 75, 79, 94, 108]. These examples are an illustration of how stylometry has become dominated by computational methods in the last decades, specifically artificial intelligence applications involving natural language processing for quantifying writing style and machine learning techniques for learning and classification. Current authorship attribution techniques can achieve more than 80% accuracy on a dataset of 100 authors [1], over 30% for 10,000 authors [66] and even significant accuracy for 100,000 authors [80].

Stylometric research is motivated by the hypothesis that every individual has a unique writing style, a "stylistic fingerprint" that can be quantified and learned. In [55] it is suggested that this uniqueness originates in that the language learning process is individual, thus everyone learns language slightly differently than another, resulting with small variants in how we communicate and specifically write. These differences, originated in conscious and subconscious decisions when we write, are expressed in different levels of the text. They can then be measured with a vast range of features, simple to complex, as discussed later in Sec. 2.2.2.

Authorship attribution problems can be coarsely divided into 3 categories [55]: closed-world problems, where we aim to find an author of a document from a known set of suspects; open-world problems, where again we seek to reveal the authorship of a document, but without prior information on possible authors; and using stylometric analysis to classify text by different characteristics of the document or the author, such as native language of the author, age, gender, number of authors of a document [7, 10, 70] etc.

Authorship verification in the context of stylometry is the problem of determining whether a given document is written by a given author or not based on the linguistic style of the text. As discussed in Sec. 2.1, this problem is much harder than the common supervised stylometry, where a decision has to be made out of a closed suspect list. With authorship verification, only knowledge of one potential author is assumed to be available, with no list of alternatives. Despite the differences, basic concepts of computational stylometry apply in the case of verification as well. Therefore, the rest of this section is dedicated to better understand the building blocks of stylometry research: problem formulation, linguistic features selection and classification methodologies. Thorough surveys of authorship attribution methods, mostly formulated in the standard multi-class classification problem configuration, can be found in [55, 65, 73, 81, 99].

2.2.1 Stylometry Problems

The basic components of any stylometry problem are a test document of unknown authorship and a candidate author or set of authors, represented by a set of writings from which linguistic style can be inferred. From this starting point, several research questions can be asked. For convenience, the following notations are used: let D denote a document, $\mathcal{D}$ a set of documents, A an author and $\mathcal{A}$ a set of authors. Following are the questions of interest:

Who in $\mathcal{A}$ wrote D? The most explored research question in the literature is this authorship attribution version of supervised learning. We are given a set of labeled documents of candidate authors from which we aim to model their style, to later attribute the most stylistically similar author from $\mathcal{A}$ as the author of D. Some of the abundance of methods developed to solve this question are surveyed in [81, 99].

Segment $\mathcal{D}$ (or D) by authors. Or, formulated as a question, how many authors are involved in writing the documents in $\mathcal{D}$ and which $D \in \mathcal{D}$ was written by which author (or: how many involved in writing D, and which author wrote which section/paragraph)? This is the unsupervised version of stylometry, in which we are given unlabeled data that we are required to cluster into sets of stylistically similar documents.

Is D written by A? This is the *authorship verification* problem, where we are given a candidate author and are required to determine whether the document in question is written by this author or not. A close problem to verification is *plagiarism detection*, where usually two texts are compared to find similarities between them. Since authors in stylometry problems are represented by a set of documents they authored, plagiarism detection and authorship verification are essentially the same problem.

These basic queries can be combined to formulate other interesting questions. For instance, a mixed open- and closed-world model in which we may ask: given D and $\mathcal{A}$, is D written by some $A \in \mathcal{A}$? if so, by who? (a question that is addressed thoroughly in Ch. 6) Other instances of questions may be variants of the above, like similarity detection, a relaxation of the unsupervised problem above: given $\mathcal{D}$ and the number of authors of the documents in $\mathcal{D}$, segment it by authors.

Koppel et al. reduce the domain of problems detailed above to one formulation which they address as the "fundamental problem" of authorship attribution [67], and if answered, can solve all problems in the domain: given two documents, determine whether they were written by a single author or not. This formulation is probably the most difficult form of problem to address, and it essentially describes the authorship verification problem discussed in this survey (only not from a particular author's perspective): is some given document written by an author (represented by another document), or not. This further supports the importance and relevance of investigating the authorship verification problem: solving it will unveil solutions to many other problems in the domain of authorship attribution.

In order to answer any of the questions above, one must follow two steps: quantification – extract linguistic features from the data, learning and classification – learn models of the training and test data, and apply the classification process corresponding to the research question. These steps are discussed in the next sections.

2.2.2 Linguistics and Features

In the heart of every learning problem is the representation of the raw data in a space such that the classes of interest are most distinguishable. In stylometry this process translates to extracting linguistic features to model the style of a document or an author, utilizing natural language processing tools and methodologies.

Basic elements of written language, such as morphology, syntax, and lexical semantics [58] help in identifying authors uniquely. These elements encompass a virtually endless space of features, from which the researcher has to carefully choose those with the highest distinguishability between authors. For instance, looking at frequencies of English letter bigrams alone, which are all pairs of letters, includes $26^2 = 676$ features. There is no consensus on one particular set of features that achieves the highest distinguishability in authorship datasets, nor there should be; the selected feature set may be different across domains, volume of available data and class of interest. However, an extensive amount of work and decades of research suggest several types of features are highly effective in stylometric analysis [55], as presented next.

Function Words. Function words are topic independent words, used to describe relationships between other, content related, words. Some classes of function words include articles ("the", "a", "an"), pronouns ("he", "she", "him") and particles ("if", "however", "thus"). The use of function words has shown to be effective in early stages of research, such as the aforementioned case of the Federalist Papers [79]. In addition, since these words are content independent, they provide a good method to perform cross domain analysis, avoid the trap of topic-dependent modeling and capture differences when modeling characteristics like native language of authors of English text [70]. Moreover, function words are the most common class of words to be present in written language, making them a reliable attribute to measure when performing stylistic profiling [55].

Vocabulary. The vocabulary an author uses can deliver very useful information in isolating this author's style from others. For instance, using *centre* and *colour* rather than *center* and *color* may indicate the author is of British nationality and not American [55]. Examples of more general vocabulary attributes include different measurements of vocabulary richness, which aim to capture how wide and complex the vocabulary diversity of an author is. The synonym-based method [27] is a fine example of a pure vocabulary based method, in which the selection of each word is measured and weighted by accounting for how common that word is and how many choices the author had.

Syntax. Syntactic features quantify the grammatical structures an author uses. Similar to function words, this class of features is also content and topic independent. Syntactic features span from very basic and easily extracted, such as punctuation, to complex and computationally challenging, such as parts-of-speech (POS) tags (the syntactic roles of words in a sentence, like verbs, nouns or adjectives). Some features capture both vocabulary and syntactical attributes, such as word n-grams, where frequencies of sets of n consecutive words are measured. Taking this idea further, measuring *character* n-grams can provide a more structural look at the text, avoiding the need to apply morphological analysis sometimes required for a clean word-based analysis [55].

The list of potential features expands far beyond the 3 categories above. Some are numeric measurements that try to capture the complexity of the text, such as word, character or sentence statistics, or different readability or vocabulary richness metrics like Yule's characteristic k. Others are classes of features of the same type, usually measured by frequency (absolute or relative), such as different n-grams in the character, word, sentence or even paragraph level, including lemmas (canonical forms), idiosyncrasies, spelling/grammar errors etc.

In the context of authorship verification, the little research that has been done is not characterized by special features that can be said to be different than standard authorship attribution techniques. Even the unique unmasking algorithm [64, 68] (discussed in detail in Sec. 2.3.1), which measures "depth-of-difference" by looking at certain metadata of the extracted linguistic features, uses word unigrams as basis, which is fairly common in authorship attribution methods. This does not come as a surprise, since both verification and attribution techniques aim to find a quantified representation of the text such that author distinguishability aspires to be high. Therefore it is only natural that features used in attribution are borrowed for verification.

2.2.3 Learning and Classification

Approaches to computational stylometry rely heavily on machine learning methods for supervised and unsupervised learning of the feature space. Most research utilizes out-of-the-box algorithms such as support vectors machines, neural networks, principal component analysis etc., listed in detail in [55]. In addition, there exist methods tailor-made for stylometry, such as the Writeprints method [1] or the synonym-based method [27]. Next we discuss some of the leading learning approaches in use.

Supervised Stylometry. The most prevalent approach in stylometry research is supervised learning, that aims to find an author of some anonymous document from a given labeled set of documents written by known authors. Support vector machine (SVM) classifiers are proven to be effective both performance and accuracy-wise in most cases [55]. However, others such as Bayesian classifiers, linear discriminant analysis, neural networks, and decision trees appear in the literature as well, and proven to be useful in some instances.

Unsupervised Stylometry. Under unsupervised settings, we are given unlabeled text, usually a set of documents, which we are required to segment into clusters of similar authorship. This approach is useful when information of authorship has to be deduced from the data, for instance recognizing authorship of paragraphs in a multi-author document [7]. One common approach for unsupervised learning in stylometry is using principal component analysis, which assists in reducing large feature spaces (often the case in stylometry) by projecting vectors onto a subspace with the highest variance of features, in order to maintain distinguishability. The Writeprints method [1] is an example of a variant of PCA suited for similarity detection. The unsupervised domain of problems is harder than the common supervised approach, and it is of increasing interest in recent years.

2.3 Current State of Authorship Verification

Among the abundance of research on computational stylometry, there is a rather limited work on authorship verification. As document classification has been the target of some one-class classification work, the question of interest has not always been that of authorship. For instance, in [74] one-class SVMs are used for category (i.e. topic) classification of the Reuters article dataset. In addition, that work and others are more focused on the machine learning perspective than on the linguistic profiling question. However, some work has focused on authorship verification, as discussed next.

In this section we provide a survey of verification approaches found in the literature. For two of these approaches a thorough description is provided, due to their uniqueness and importance in the area of authorship verification. The first is the *unmasking* algorithm [64, 68] which entails measuring the "depth-of-difference" between a set of known texts by a candidate author and an anonymous document to be tested. The second is a simplified framework for verification that does not attempt to reduce the model to a binary one by using a distractor set (modeling *not* the author), namely *distractorless verification*, whose importance is in its simplicity and evaluation in real-world settings.

2.3.1 Unmasking

Koppel et al. present the *unmasking* algorithm [64, 68], designed as an iterative process in which the most differentiating features are eliminated gradually, in order to measure the accuracy degradation rate of the learnt models. They hypothesize that when the training author matches the author of the test document, the accuracy degrades much faster than otherwise. They demonstrate how this method distinguishes well between shallow differences, originated in a conscious or unconscious changes in one's style, and deeper differences between styles of different authors. They claim this method is robust and language, period and genre independent.

In addition, they note two important points in authorship verification, excluded from general one-class problems. First, there is an abundance of negative examples in authorship verification problems – any writings known to not belong to any of the tested authors. Such a set will not be complete, however negative information is incorporated to some extent in their algorithm. Second, when considering long text samples, they can be chunked into smaller samples, effectively creating example *sets*, each with one author.

Koppel et al. define an authorship verification problem over example sets of writings, where every chunk in every set consists of at least 500 words, without breaking up paragraphs. This method seems reasonable given they evaluate it over a corpus of books, as detailed in the next section.

They define the problem as follows: for every author A and book X, A_X is the set of all the works of A except X (if A is not the author of X, A_X is simply the entire works of A). The objective is to classify the pairs $\langle A, X \rangle$ as either *same-author* or *different-author* accordingly. These notations are used throughout the remainder of this section.

Corpus

The corpus used contains 21 publicly available 19^{th} century English books of varying genres, written by 10 different authors. Excluding one problematic author-book pair and repeating pairs, the corpus allows constructing 189 distinct *different-author* pairs and 13 distinct *same-author* pairs. The corpus is used to both set the configuration of the classifier generated by their method and to later evaluate it.

The Unmasking Algorithm

The unmasking algorithm is divided into two phases. In the first phase, an iterative classification process is applied in order to construct accuracy degradation curves. The selected feature set for classification contains the 250 words with the highest averaged frequency in the training and test data, giving equal weight to A_X and X. The usage of 250 is chosen as experiments indicated it is a reasonable boundary between common words, and those tied to a particular work. They run an iterative process for 10 iterations, in each a linear-kernel SVM is used for cross-validation. The process is as follows:

1. Record the accuracy of a 10-fold cross-validation experiment on A_X versus X
2. In each fold, eliminate the 3 strongest-weighted positive and 3 strongest-weighted negative features
3. Repeat

Their hypothesis is that if A is the author of X, only a relatively small set of features captures the differences between them, overcoming theme, genre and other distinctions. Therefore in such instances a rather fast degradation is expected when this set of features is gradually removed from the classification process.

Next they quantify the differences between *same-author* and *different-author* curves by extracting the following features from them:

- Accuracy after every iteration
- Accuracy difference between any two consecutive iterations
- i^{th} accuracy drop in one iteration, for $i \in \{0, 1, ..., 9\}$
- i^{th} accuracy drop in two iterations, for $i \in \{0, 1, ..., 9\}$

After classifying the curves based on the vectors above into *same-* and *different-author* sets using ground truth, a meta-learning process is applied in order to determine the roles of the various features. All 13 distinct *same-author* curves were found to hold the following 2 conditions, whereas only 5 of the 189 *different-author* curves hold them:

- Accuracy after 6 iterations dropped below 89%
- The second-highest accuracy drop in two iterations is above 16%

In addition to the accuracy curves above, they suggest an alternative measurement of "depth-of-difference" by looking at the number of features with significant information gain, with respect to some threshold. Similarly it is expected that *same-author* curves will show a more sudden drop than others.

In terms of the one-class classification approaches discussed in Sec. 2.1.2, this method is not a pure one-class approach, as it reduces the problem to a binary classification one, between models built for *same-* and *different-author* curves. However this approach can be classified as a boundary method: support vector machines are used to construct the degradation curves and a boundary is attempted to be defined between these two classes in the meta-learning phase.

Extension: Using Negative Examples

As a last phase, an optional correction method, denoted as *elimination*, that utilizes negative examples is suggested. For every author A, if writings of other authors with similar type to A (e.g. geography, chronology, culture and genre) are available, they may be used to eliminate some false-positives. That is, in case the basic unmasking concludes A wrote some book X, an elimination method is applied with the potential to overrule the unmasking, resulting with classifying X as *not* written by A. The elimination process cannot, however, reverse the decision if the unmasking determines A is not the author of X: consider the set of writings of those similar authors representing not-A; if the elimination determines that X is written by A, it would only mean X's style is closer to A's style than to not-A's style, but not necessarily that it was written by A.

For every author A and set of authors of the same type $\bigcup_{i=1}^{n} A_i$, we define all A_i collectively as not-A, and similarly define not-A_i, for all i. In addition, for any book X denote $A(X)$ as the percentage of X's chunks classified as A (complement to those classified as not-A). Under these notations the elimination process works as follows:

- Learn a model for A against not-A
- Learn a model for A_i against not-A_i, for all i
- Calculate $A(X)$ and $A_i(X)$, for all i
- If $\exists i : A(X) \leq A_i(X)$, conclude that A did *not* write X
- Otherwise, accept the original decision that X is written by A

Evaluation and Results

In the main experiments, the authors first establish a baseline by testing each author-book pair $\langle A, X \rangle$ with a model built for A_X using a one-class SVM with RBF kernel and the 250 most frequent words in A_X. Each pair is classified as *same-author* only if more than half the chunks of X were attributed to A_X. The poor results obtained were 30% true-positive and 24.33% true-negative, which outperformed using other thresholds or kernels.

Next, they evaluate the configuration of the unmasking method obtained over the corpus by applying a series of independent leave-one-out tests. Each book B in turn is first eliminated from the corpus. Curves are then extracted for each $\langle A_X, X \rangle$ pair using unmasking, followed by meta-learning using a linear SVM to distinguish between *same-author* and *different-author* curves. Next, for each author A curves are extracted for $\langle A_B, B \rangle$, which are then classified using the B-independent learned model. They report 19/20 correct *same-author* and 181/189 correct *different-author* classifications, concluding to an overall accuracy of 95.7% with similar false positive and negative rates. An example set of curves extracted for *An Ideal Husband* is shown in Fig. 2.1.

In order to test the alternative information-gain based curves method, they conduct the same experiments as above with curves generated by using varying thresholds between 0 and 0.6, with

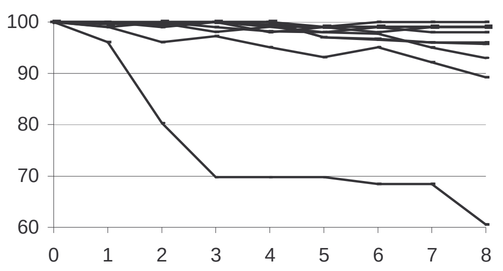


Figure 2.1: Unmasking *An Ideal Husband* by Oscar Wilde, whose curve is below all other authors.

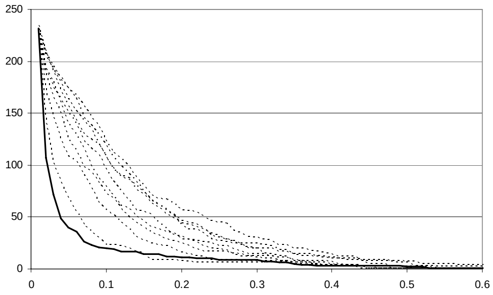


Figure 2.2: Unmasking *An Ideal Husband* by Oscar Wilde using information-gain curves, whose curve is the dark line below all other authors.

steps of 0.01 (see example in Fig. 2.2.) This method worked rather well for *different-author* curves and resulted with 182/189 correct classification, however only 11/20 of the *same-author* curves were correctly classified.

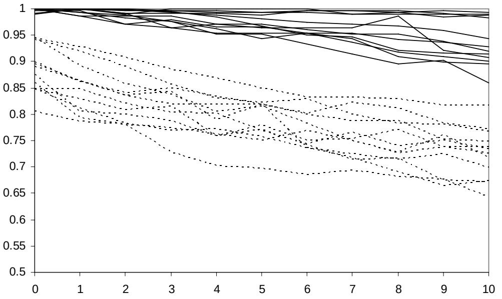


Figure 2.3: Unmasking same/different topic Hebrew-Aramaic collections. Solid lines are *different-author* curves (same topic) and dotted lines are *same-author* curves (different topics).

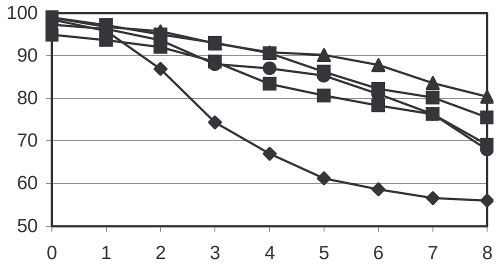


Figure 2.4: Unmasking *Torah Lishmah* against Ben Ish Chai and 4 other authors.

Finally, applying elimination on the tests above (using the original accuracy-curves) resulted with only a single new false-negative. Meanwhile all 8 false-positive classifications were corrected, leading to an overall accuracy of 99.04%.

In a set of additional experiments, they test the parameter choice of 250 words (n), 10 iterations (m) and 3 features to be removed from each extreme of the feature set (k), by repeating the leave-one-out experiments with different parameters. They show the obtained accuracy is somewhat robust with regard to m and k, however as n increases the accuracy degrades. They conclude that this result is due to the importance of significantly stripped-down feature sets to the distinguishability of the curves in the meta-learning phase.

Next, the authors measure topic variability effects on unmasking. They use a corpus of 20^{th} century Hebrew-Aramaic collections of letters by 3 different authors, each containing texts on 3 different topics. They run 2 sets of experiments: each author vs. all others on the same topic, and each author's topic against the same author's other topics. They show that topic related words are quickly eliminated, leaving topic-independent stylistic markers as the only features remaining: *different-author* curves, although tested with the same topics, remain with high accuracy, whereas *same-author* curves degrade rather quickly despite the topic variations. See Fig. 2.3.

As a final test they use the unmasking method to solve the anecdotal literary mystery of the Bashful Rabbi: examining whether the *Torah Lishmah*, a Hebrew-Aramaic collection of letters, is written by the 19^{th}-century rabbinic scholar Ben Ish Chai, who claimed to have found them whereas the consensus among historians is that he is in fact the author but wished to hide it (and may have taken measures to obfuscate his style). For comparison, a similar collection by Ben Ish Chai is used, along with 4 other collections of the same type by 4 other authors from the same area and time. The curves extracted by unmasking shown in Fig. 2.4 suggest that Ben Ish Chai was indeed the author.

Contribution

Recursive feature elimination procedure was first presented by Guyon et al. [42] in the context of gene selection for cancer classification. However, the unmasking paper is novel in utilizing "depth-of-difference" measurement for verification in the domain of written-text, and it is one of the pioneer methods in the field of authorship verification. Unmasking is shown to be limited to tasks with large training data [95], with experiments suggesting a minimum of 5,000 to 10,000 words; nevertheless, this method is proven robust and highly accurate with a large enough dataset, untied to a particular language, period or genre. Moreover, the model suggested may be transferred to other applications for "depth-of-difference" measurement between two given samples.

2.3.2 Distractorless Authorship Verification

Noecker and Ryan address the verification problem avoiding construction of a negative set representing the class of *not* the author in question, relying solely on the author's training data [82]. Their approach uses simplified feature sets, distance measurement and a threshold to determine classification results. Although their best results suffer from rather low F-score measurements, their work provides a baseline distractorless authorship verification framework, robust across different types of writings (language, genre or length independent) with tunable parameters to determine the desired type I and type II error rates, a useful feature for various applications.

The problem defined by the authors is very straight forward: given a document D and a candidate author A, determine the likelihood that A is the author of D.

Corpora

The distractorless verification method is evaluated on two corpora: the Ad-hoc Authorship Attribution Competition corpus (AAAC) [54] and the PAN 2011 Authorship Identification Training Corpus [89].

The AAAC corpus contains texts across different genres, languages and lengths. It consists of texts of 63 authors, with 264 training documents and 98 test documents, across 13 different problems.

The PAN 2011 Authorship Identification Training Corpus contains texts that represent "real-world" writings, one of the reasons it was chosen by the authors. It includes non-coherent, unorganized and short texts, like emails from the Enron dataset. Another reason it is chosen is to be used as control in order to generalize beyond the AAAC corpus. The corpus consists of 5,064 training documents and 1,251 test documents of 10 authors.

Algorithm

The analysis begins with preprocessing of the datasets, in which whitespaces and character case are standardized. In addition, any author-identifying tokens in the PAN corpus are stripped down.

Character n-grams and word n-grams are used as feature sets, extracted using a sliding window technique (with sliding step of 1). Character n-grams, with n from 1 to 20, are chosen due to their high performance known in authorship attribution research, and word n-grams, with n from 1 to 10, are chosen for completeness. These rather simple features are chosen due to their fast calculation and robustness against errors (unlike high-level features, such as POS-tags).

Next, for any given set of features of size n, a centroid of the feature vectors for the author's training documents is extracted to build a model $M = \langle m_1, m_2, ..., m_n \rangle$. For each i, m_i is the average *relative* frequency of feature i across the training documents, where the relative frequency is used to eliminate document length variation effect. In addition, a feature vector $F = \langle f_1, f_2, ..., f_n \rangle$ is extracted from document D, where f_i corresponds to feature i's relative frequency in D.

Finally, a distance function δ and a threshold t are set for the final test. The generic distance function is defined such that if $\delta(x, y) > \delta(x, z)$, x is considered to be *closer to* y than to z (in that sense, δ is more a resemblance metric than a distance metric.) The authors use normalized dot-product (cosine distance), as it is shown effective for authorship attribution and efficient for large-scale datasets. For an author model M and a document model F as described above, the distance is calculated as follows:

$$\delta(M, F) = \frac{M \cdot F}{\|M\|\|F\|} = \frac{\sum_{i=1}^{n} m_i f_i}{\sqrt{\sum_{i=1}^{n} m_i^2} \times \sqrt{\sum_{i=1}^{n} f_i^2}} \tag{2.1}$$

The threshold t is set such that if $\delta(M, F) > t$, we determine that D is written by A. It is empirically determined by analysis of the average δ between the author's training documents.

Following is a summary of the algorithm:

1. Use author A's training documents to build a model $M = \langle m_1, m_2, ..., m_n \rangle$ such that for all i, m_i is the centroid of feature i.

2. Extract the corresponding feature vector $F = \langle f_1, f_2, ..., f_n \rangle$ from the test document D.

3. Determine a threshold t and calculate $\Delta = \delta(M, F)$. If $\Delta > t$, determine that D is written by A. Otherwise, determine it isn't.

The distractorless method is an example of a one-class boundary approach, closest to the k-centers method described in Sec. 2.1.2 using $k = 1$: a single centroid is defined for the target author by averaging over the author's documents, and a boundary around the author is set using the acceptance threshold t.

Evaluation and Results

Summary of accuracy and F-score results are shown in Tab. 2.1. The high accuracy and low F-score suggest a skew in the results in favor of instances where the author is *not* the author of the test document (using the unmasking terminology, there are naturally much more *different-author* pairs than *same-author* ones.) To better illustrate the results, and in accordance with the evaluation technique of one-class classification algorithms mentioned in Sec. 2.1.2, we generated ROC curves by repeating the experiments of this paper for the AAAC corpus to obtain type I and type II error rates. Fig. 2.5 illustrates the best curve obtained, using word trigrams as features. It is seen that when the true positive rate (TPR) approaches 70%, the slope is rather steep, resulting with a false

Table 2.1: Accuracy and F-Score results for the distractorless verification algorithm.

Corpus and Features	**Best Accuracy**	**Best F-Score**
AAAC with char. n-grams	87.44%	47.12%
AAAC with word n-grams	88.04%	44.58%
PAN with char. n-grams	92.23%	51.35%
PAN with word n-grams	91.53%	43.08%

positive rate (FPR) of at most 26.7% (at 68.4% TPR). However at that point on, the slope becomes moderate, resulting with a faster increase rate of FPR (for instance, at a TPR of 80%, FPR is already at 53.8%.)

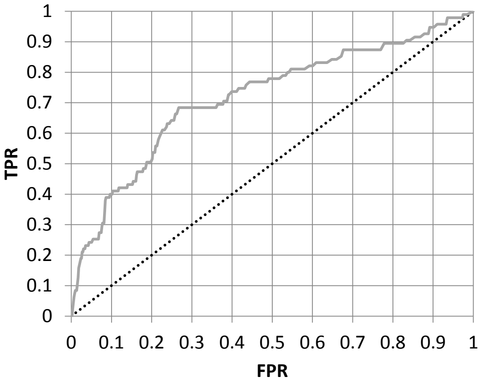


Figure 2.5: ROC curve for the AAAC corpus using the distractorless verification algorithm with word trigrams.

In addition to the analysis above, a relaxed version of the algorithm is introduced where confidence ratings of the verification results are incorporated, and a level of confidence can be set such that any answer beneath it results with no classification (i.e. insufficient information to determine yes or no). Preliminary evaluation where the 20% most difficult problems are eliminated show an increase in accuracy from 92% to 96%. For applications that can absorb the confidence penalty, these results are promising.

Contribution

The distractorless authorship verification algorithm is simply constructed and provides a baseline for verification, eliminating errors generated by using negative examples in training. This framework allows tuning type I vs. type II error rates, as required by the application in which it is incorporated. The high accuracy rates attained over several real-world datasets demonstrate that verification is obtainable with satisfying effectiveness scaling across different domains and types of writings, as opposed to long literary works like those used to evaluate the unmasking algorithm. The ROC curve in the regenerated experiments with the AAAC corpus shown in Fig. 2.5, however, suggests that the accuracy results should be taken with a grain of salt and type I/type II error rates ratio should be carefully measured for the task in hand.

In addition, future developments of this work include interesting directions, from adding confidence levels for classification relaxation, through incorporating a mixture-of-experts style voting system that combines several distance techniques and feature sets, to tuning a default non-application-specific threshold. All the directions above lay solid grounds for improving state-of-the-art techniques in authorship verification.

2.3.3 Other Authorship Verification Approaches

In addition to the two methods presented thus far, the literature includes more work in the field, all rather recent. Other approaches include different focus and formulation of the authorship verification problem, such as in the context of plagiarism detection, one-sided models versus 1-against-all formulation, varying amounts of training data etc. The following summarizes a few more approaches, each with its own uniqueness, focus and contribution to the field.

Linguistic Profiling

One of the most common applications of authorship verification is plagiarism detection, a scenario where we wish to determine whether an individual claiming to be the author of a given text is fraudulent. Some work has examined authorship verification in the context of plagiarism detection [28, 121], however van Halteren [109] lays a general approach for the same task as a verification problem. He defines stylometric analysis as linguistic profiling and addresses both the recognition and verification problems.

Features Three feature sets are used: the first is a lexical feature set, which includes common words, context independent part-of-speech tags, utterance lengths and all possible bi- and trigrams of the above; for efficient processing, features with less than some minimum occurrence across the corpus are filtered and counted as “other”, yielding a total of 100K features. The second is a syntactic feature set, which consists of n-grams of syntactic constituents derived with the Amazon parser as syntactic features, which yields (after filtering) 900K features. Finally, the third feature set is a combination of the former two.

Corpus The Dutch Authorship Benchmark Corpus [11] (ABC-NL1) is used for evaluation, consisting of 72 texts by 8 authors on 9 different topics, and controlled for age, educational level, and genre.

Algorithm The quantified document vectors are denoted as the *profile vectors*, which consist of the feature-wise standard deviations with respect to the mean observed in a profile reference corpus (the entire test corpus, including attributed and non-attributed documents) where all values are normalized for document length. Verification is determined by using a normalized distance-based score with threshold optimization, as follows:

- An author profile vector A is constructed by taking the feature-wise average over all profile vectors extracted from that author's training documents
- A profile vector T is extracted from the sample
- A weighted combination is calculated: $\Delta_T = (\sum_{i=1}^{n} |T_i - A_i|^D |T_i|^S)^{1/(D+S)}$, where D and S are the weighting factors for the author-sample score difference and sample score itself, respectively
- The final score is determined as $Score_T = (\sum_{i=1}^{n} |T_i|^{D+S})^{1/(D+S)} - \Delta_T$ s.t. the higher it is, the closer the sample profile is to the author profile

This algorithm combines a reconstruction approach, where a model is applied on the data using standard deviations calculated with respect to a reference corpus, and a boundary method approach, where a distance is calculated between the author and the sample model and compared to an acceptance threshold that effectively defines the boundary.

Evaluation and Results Evaluation is applied using 9-fold cross-validation, where each fold consists of texts on the same topic. A single fixed acceptance threshold is used, such that every score above it is deemed accepted. The verification methods performance is measured via false accept and false reject rates (i.e. false positives and false negatives, respectively). As there is a tradeoff between the two rates, and the verification application in mind is plagiarism detection, $FAR_{FRR=0}$

is evaluated, i.e. the lowest false accept rate achievable given zero tolerance to false rejects, where an author may be wrongly accused of plagiarism.

The best result reported is $FAR_{FRR=0} = 8.1\%$, achieved with the combined feature set. In addition, the system manages to achieve up to $FAR_{FRR=0} = 2.4\%$ and $FAR_{FRR=0} = 0.2\%$ by using renormalization (essentially using the knowledge that each of the 8 authors wrote exactly one test document) and "oracle" thresholds, respectively. Moreover, authorship attribution (referred as "recognition" in the text) accuracy for 2-way and 8-way comparisons are reported to reach as high as 99.4% and 97%, respectively (and up to 100% with renormalization).

Contribution This work is one of the first to define and address the authorship verification problem. Although the method presented here incorporates some "magic parameters" (acceptance threshold included; parameter selections are explored in [44]) that need to be configured empirically, and the evaluation presented is rather limited in dataset size and real-world-like properties, it lays good grounds for further work on verification.

Plagiarism Detection without Reference Collections

Many plagiarism detection methods target verification of documents with respect to a reference collection of documents by potential original authors. Meyer zu Eissen et al. [120, 121] target to solve a harder problem: detect a plagiarized passage within a document by identifying changes in writing style within the document, which can then lead to searching origins in external sources for the passages suspected to be plagiarized. This type of problem is referred to as *intrinsic plagiarism detection.*

Features The authors use a feature set that consists of several types of features: average sentence length; 18 part-of-speech tags; average number of stopwords; grading measurements which include the Gunning Fog index, Flesch-Kincaid Grade Level, and the Dale-Chall formula; and finally vocabulary richness measurements which include Honore's R, Yule's K, and a novel method: the averaged word frequency class.

Word frequency class is a measurement that assigns a bin for each word based on its frequency relative to the most frequent word (in the measured context: document, paragraph, etc.). For every word w, the class is defined as $\lfloor \log_2(f(w^*)/f(w)) \rfloor$, where $f(w)$ is the frequency of w and w^* is the most frequent word.

Corpus The authors present a new corpus constructed specifically for evaluating plagiarism detection, controlled for: authenticity, homogeneity, possibility to include different types of plagiarism, processable by human and machine and with clear separation between text and metadata. Their corpus consists of computer science articles from the ACM digital library, manually modified to include original and modified paragraphs taken from other articles. Template documents are represented as XMLs from which instances can be created to contain varying number of plagiarized passages.

Algorithm The analysis is performed on each document by first breaking it into passages. Then features are extracted from each passage p to formulate feature vector f_p, and from the document d as a whole to formulate vector f_d. Then each f_p is compared with f_d to produce a relative differences vector, for binary classification (plagiarized/original).

As seen next, the algorithm uses a discriminant analysis classifier for the binary classification, making it a reconstruction method.

Evaluation and Results First the authors evaluate their suggested vocabulary richness measurement, namely the average word frequency class (denoted $AWFC$), compared to Honore's R and Yule's K, in terms of stability with respect to the text length. For that purpose they measure $AWFC$ on 4 different authors as a function of the percentage of analyzed text, which resulted in an almost constant value (different between the authors), independent of that percentage. In addition, compared to the other measurements, $AWFC$ is proven the most non-varying, even for small passages, deeming it useful for intrinsic plagiarism detection.

The main evaluation is applied using 10-fold cross validation over 450 generated document instances which contain 3–6 plagiarized passages, 40-200 words per passage. Using a classical discriminant analysis classifier, the authors report $\approx 70\%/80\%$ precision/recall for 3% plagiarized text, which increase to over 80%/90% for 18% plagiarized text. They obtain similar results using SVMs. Their novel $AWFC$ vocabulary richness is reported to be the most discriminating feature used.

Contribution Other than the new length-robust vocabulary richness measure, namely $AWFC$, this work lays a basis for an important authorship verification application – plagiarism detection research, in one of its harder formulations which they define as intrinsic plagiarism detection. This problem is similar to the unsupervised stylometry problem, where attributing authors to their parts of a multi-authored document is sought [63]. This work provides a novel corpus targeted for plagiarism detection research, and a precision/recall baseline to be further improved in future work.

Intrinsic Plagiarism Analysis with Meta Learning

Stein and Meyer zu Eissen [100] revisit the intrinsic plagiarism detection problem and suggest coupling the solution suggested previously in [120] with meta learning – namely the *unmasking* technique [64]. They present a hypothesis generation method for the intrinsic plagiarism detection problem. They evaluate the hybrid solution on an artificially plagiarized dataset, and on a real-world plagiarized document, demonstrating its efficiency in solving the problem.

Features A feature is addressed as a style marker σ, which is a mapping of text segments to numerical values. Since plagiarized sections are often short, the feature set is to be selected accordingly. The feature set used in the main experiment includes 20 POS tags, and 9 style markers including average sentence length, Honore's R and average German word frequency class.

Corpus The authors evaluate two corpora. The first is comprised of 50 scientific single author German papers, 12-15 pages in length. These documents were plagiarized manually by adding k plagiarized sections to create a template from which 2^k instance documents can be created. Instances were generated to create a varying percentage of plagiarized text $\theta \in [0.05; 0.5]$. Overall there were 16,000 non-plagiarized sections, and 1500 plagiarized paragraphs. For evaluation in real-world settings, a known plagiarized postdoctoral thesis from the 1980s is used.

Algorithm In the improved style marker analysis method suggested in this paper, the features are assumed to be Gaussian distributed for the target class, and uniformly distributed for the outlier class. For a section s in document d, denote S^+ as the event it is not plagiarized, and S^- the event it is. By hypothesizing the a-priori probabilities $P(S^-) = \theta$ and its complement $P(S^+)$, d is decomposed into sections $s_1, ..., s_n$. For a single style marker σ, its expectation and variance are estimated over $s_1, ..., s_n$, $P(\sigma(s)|S^+)$ and $P(\sigma(s)|S^-)$ are computed, and Bayes rule is applied for calculating $P(S^+|\sigma(s))$ and $P(S^-|\sigma(s))$. Applied over m style markers, the final decision is $S \in \{S^+, S^-\}$ that maximizes $P(S) \prod_{i=1}^{m} P(\sigma_i(s)|S)$.

For the meta learning phase (which can follow either the method above, or the intrinsic plagiarism detection method in [120]), first all sections attributed as plagiarized are accumulated to create d^-, and similarly d^+ is formed. Unmasking is then applied to create a curve over d^- against d^+, which in turn is classified as either *same-author* or *different-author* (using the notations in Sec. 2.3.1). If it is classified as *different-author*, the document is determined to be plagiarized.

The improved style marker analysis method is a perfect example of a density method, that assumes densities for the author and outlier classes and applies Bayes rule with a threshold for differentiation. It is followed by a boundary method – the unmasking meta-learning phase.

Evaluation and Results First, the unmasking technique is evaluated on the artificially plagiarized data. *same-author* curves are generated from sections in d^+ against other sections in d^+, and *different-author* curves from d^+ against d^-. As expected, the degradation in the former curves was significantly faster than the latter, proving the usability of unmasking in this scenario.

In the main experiment, the hybrid method above is applied to the artificially plagiarized corpus with a discriminant analysis classifier. The obtained precision and recall for the non-plagiarized class were in 80-90% for the varying θ; for $\theta = 0.5$, precision and recall for the plagiarized sections were 55% and 70%, respectively.

Finally, the method is tested on the plagiarized postdoctoral thesis. It is first divided into 138 sections, out of which 13 were classified as possibly plagiarized (from which 3 are known to be so). Two known plagiarized sections were missed, possibly due to a too coarse decomposition. In the meta learning phase, the d^+/d^+ curve and the d^+/d^- curve are clearly *same-* and *different-author* curves, i.e. the unmasking phase confirmed the plagiarism.

Contribution This work is significant mainly in its introduction of unmasking application to the intrinsic plagiarism detection problem, which can be argued to be a more relevant real-world problem, both in type and length, compared to the literary problems it was initially evaluated over. The hybrid solution suggested here shows successful fusion of verification methodologies, and inspires such combined solutions in future work by demonstrating the potential gain in performance for authorship verification applications.

Many Authors, Limited Data

Luyckx and Daelemans [72] address the problem of performance overestimation of current stylometry approaches, which usually rely on unrealistically large amounts of training data, overestimate the importance of specific features for generalized problems and tested with a small number of authors. They refer to the authorship verification problem as the more realistic case where the number of suspects is unknown. They explore this problem in a 1-vs-all fashion using negative examples on problems with many suspects and little training data. They present a memory-based learning approach and show it is more robust than the commonly used “eager” learning methods such as SVMs or maximum entropy classifiers.

Features Features are selected automatically by their predictive value to the problem, as commonly used in classification problems. The authors focus on part-of-speech (POS) grams as syntactic features, since these are unconsciously selected by authors (as opposed to token level features), extracted using the Memory-Based Shallow Parser. The χ^2 metric is used to select features that best discriminate between the classes, where: $\chi^2 = \sum_{i=1}^{k} \frac{(\chi_i - \mu_i)^2}{\sigma_i}$. The authors use several feature sets and a combination thereof, including lexical distribution n-grams, fine-grained POS grams, coarse-grained POS grams, function words and the Flesch-Kincaid readability index.

Corpus The authors present the *Personae* corpus, a 145-author collection of Dutch essays, about 1400 words each, written by undergraduate level students on a documentary about Artificial Life, thus it is controlled for language, educational level, age, genre and topic. In addition, the students took the Myers-Briggs Type Indicator (MBTI) test for personality evaluation.

Algorithm Two approaches are taken for learning, “lazy” and “eager”. For lazy learning the authors use the Tilburg Memory-Based Learner (TiMBL) [30], an inductive algorithm based on k-nearest-neighbors (using $k = 1$) with feature relevance weighting. With this algorithm, no abstraction of the classes is applied; rather test instances are matched to all training vectors stored in memory, from which the class is determined. As eager learning methods, Sequential Minimal Optimization SVMs and Maximum Entropy learners are used. The authors hypothesize that, as opposed to the eager learners, the lazy approach avoids abstracting away from uncommon forms in the training data, that may be useful for classification under limited data conditions.

Both the k-NN “lazy” learning and SVM “eager” learning approaches are types of boundary methods.

Evaluation and Results The authors aim to examine 3 topics: the effects of large number of authors, the effects of limited data, and finally – performance of authorship verification under these two constraints. For the first, they experiment with gradually increasing the number of suspects in the set. For the second, they experiment with 2 and 145 authors, using 20% of the corpus for testing, and gradually increasing the size of the training data. For the third, each author is tested in a 1-vs-all fashion, using 80% for training and 20% for testing. All experiments are conducted with

5-fold cross validation. The feature set selected using the χ^2 metric is narrowed down to $n = 50$. Data in each fold is divided into 10 fragments, where the generated feature vector is the means vector.

The results for the first two experiments are as expected: accuracy drops as the number of authors increases (down to 34% for 145 authors), feature set combinations outperform individual feature sets, and some feature types (sets) are shown more effective than others – but such generalization cannot be made for individual features. In addition, as the training data increases, so does the accuracy, and the TiMBL classifier is proven more robust to limited data than the eager algorithms tested.

In the main authorship verification experiments, using TiMBL and the lexical feature set yields the highest result of 56.04% precision (with 7.03% recall and 12.49% F-score), significantly higher than found for this kind of problem formulation in the literature thus far.

Contribution The main contribution of this paper is in the realistic problem formulation it evaluates verification on: large number of authors and limited training data. Although the reported results may not be satisfactory for real-world applications, they provide a fair baseline for future realistic experiments, more commonly encountered in forensic contexts.

Particle Swarm Model Selection for Authorship Verification

Escalante et al. recognize that differences in author styles may require a variety of features and classifiers selected ad-hoc per author in order to maximize overall classification accuracy [33]. For that purpose they apply particle swarm model selection, PSMS, to the authorship verification task, which forms a set of binary classifiers (one per author) for a given collection. Accordingly, they formulate the verification problem as a binary 1-vs-all problem. PSMS is the application of Particle swarm optimization to the model selection problem in binary classification. In addition to classifiers, their application is able to automatically select preprocessing and feature selection methods. The PSMS methodology is evaluated on two datasets, concluding that ad-hoc classifier selection per author is preferable over using one classifier for all authors, and provide understanding about the importance of different features to different authors.

Features The authors use bag-of-words, a rather simple feature set, as they focus on the development of classification models. Each document is represented as a boolean vector of size $|V|$, where V is the vocabulary of the entire collection of documents. Each entry j is assigned 1 if the corresponding word $w_j \in V$ appears in the document, and 0 otherwise.

Corpus To avoid overfitting and allow generalization of the evaluated methodology, two datasets are used. The first is MX-PO [29], a Spanish poetry dataset with 281 training documents and 72 test documents by 5 authors, with 8,970 features. The second is CCAT [48], an English news articles collection by 50 authors, with 2500 training and 2500 test documents, and 3,400 features. Only words that appear in a minimum of 5 times in MX-PO and 20 times in CCAT are kept.

Algorithm For a given dataset with M authors, the dataset is labeled M times resulting with M training sets, such that in training set i, all document vectors by author i are labeled $+1$, and all others -1. Then, PSMS is applied on each training set i in order to select the best methods for preprocessing, feature selection and classification for the corresponding author, and optimize their hyperparameters.

For every author i, PSMS is applied on its corresponding labeled set as follows: each full model, i.e. numerical vectors that codify a combination of preprocessing, feature selection and classification methods, is considered a particle. The swarm is the collection of all particles. The optimization problem is minimizing the classification balanced error rate $BER = \frac{E_+ + E_-}{2}$, where E_+ and E_- are the positive and negative error rates, respectively.

Each particle is initialized with position $\mathbf{x}_i^t = \left\langle x_{i,1}^t, ..., x_{i,d}^t \right\rangle$ and velocity $\mathbf{v}_i^t = \left\langle v_{i,1}^t, ..., v_{i,d}^t \right\rangle$, d being the search space dimension. Particles are initialized randomly, and updated in an iterative process that takes into account the current position, the best position obtained by the particle, the best particle in the swarm thus far and additional weighting constants and randomness. Finally, after a fixed number of iterations, a model is selected for the corresponding author, and the process is repeated for all authors.

The PSMS approach utilizes a search space over multiple types of classification techniques, and therefore does not fall under one particular approach from those described in Sec. 2.1.2.

Evaluation and Results Evaluation is applied via cross validation by measuring the average precision, recall, F1 measure and BER for each of the selected classifiers. The classifiers tested include a linear classifier, naïve Bayes, neural networks, SVM and others; feature selection methods include F-test, T-test, AUC criterion and others; preprocessing methods include normalization, standardization and scaling. In addition to optimizing preprocessing, feature selection and classification methods, PSMS is tested with fixing the classifier to nonparametric linear, and with optimizing only the classifier hyperparameters (with several classifiers).

The main results indicate that PSMS outperforms a fixed configuration: for MX-PO, BER is the lowest at 23.68% and F1 measure is the highest at 60.37%, when fixing the classifier to linear; for CCAT, BER is the lowest at 13.54% with the linear classifier, and F1 measure is the highest at 63.41% for the full optimization configuration.

Contribution This work breaks the standard go-to approach often taken in stylometry research, authorship verification included, of using a single learning algorithm across all authors in the given problem. The results obtained with the multi-classifier approach taken here show that using PSMS for individual preprocessing, feature selection and classifier methods benefits performance, and allows learning what features are important for each author. This approach may have great importance especially for hard problems such as authorship verification, where attaining high accuracy is more difficult than standard stylometry problems, and requires creative solutions.

2.4 Synthesis

This chapter reviewed the current state of authorship verification research. We presented this computational stylometry variant of one-class classification problems, and laid out current methodologies to solve it. The methods reviewed here, for which performance evaluations is summarized in Tab. 2.2, provide a solid basis for authorship verification research. Specifically, the two methods reviewed thoroughly, namely unmasking and distractorless verification, show promising approaches that can be utilized and further developed. The advantage of the unmasking technique over the others is expressed in its ingenuity of measuring the “depth-of-difference” between texts while stripping down stylistic characteristics, which targets to remove shallow differences and thus distinguish between same-origin and different-origin texts. The distractorless verification approach is advantageous in its simplicity, modularity, tunability and real-world evaluation baseline set with it. Indeed, we adopt the distractorless verification design as the basic verification technique used in our *Classify-Verify* algorithm, presented in Ch. 6.

All methods presented here attempt to utilize known approaches from the stylometry and machine learning domains, and do so with varying amounts of success. Clearly, this domain of problems is in its infancy, and research should continue to focus on finding more accurate, scalable and robust methods, applied to more privacy, forensics and security oriented domains. Next, several directions

Table 2.2: Evaluation of authorship verification methods presented in this document.

Method	Corpora (# Authors)	Language	Best Measured Performance
Unmasking [64, 68]	Books (10)	English	99.04% accuracy
Distractorless Verification [82]	AAAC (63) PAN 2011 (10)	Varying	88.04% accuracy, 47.12% F-score 92.23% accuracy, 51.35% F-score
Linguistic Profiling [109]	ABC-NL1 (8)	Dutch	$FAR_{FRR=0} = 8.1\%$
Intrinsic Plagiarism Detection (IPD) [120, 121]	450 plagiarized ACM science articles	English	80%/90% precision/recall (for 18% plagiarized text)
IPD with Meta Learning [100]	Scientific papers (50)	German	Authentic: 80-90% precision/recall Plagiarized: 55%/70% precision/recall
Many authors with limited data [72]	Personae (145)	Dutch	56.04%/7.03% precision/recall 12.49% F-score
PSMS [33]	MX-PO (5) CCAT (50)	Spanish English	23.68% BER, 60.37% F-score 13.54% BER, 63.41% F-score

for future development in the field are discussed. Ideas, techniques and approaches presented in the chapters to follow attempt to address some of these important directions.

2.4.1 Directions for Continued Research in Authorship Verification

Expanding Empirical Foundations

The limited amount of work has led to a rather limited number and diversity of datasets that have been tested with different verification methods. The standard closed-world authorship attribution domain, however, is abundant with datasets that can be trivially formulated to test verification. If more datasets are to be used and tested, it can assert the usability of current and future verification methodologies, with emphasis on which techniques are suited to what problems. Verification methods should be tested on datasets that challenge with a high number of potential authors, taking pure one-sided learning approaches, limited amounts of training data, texts with real-world characteristics and the like. The work of Escalante et al., discussed in Sec. 2.3.3 provides a baseline for this direction. Furthermore, extending evaluations to existing datasets that are common in the literature would allow uniform settings, required to truly compare approaches and determine which algorithm is suited best for different problem formulations. General stylometric techniques have been compared over the same baseline before [54], and such experiments should be applied within the domain of one-class methods as well, stressing the constraints in terms of data availability, increased number of authors, various genres and domains etc. Such configurations should simulate real-world problems where one-class approaches can be of use, such as security applications, open-world domains and the like.

Specifically, the Active Linguistic Authentication dataset [57] is a perfect candidate for testing authorship verification methods for a security oriented application. The active authentication problem is to constantly monitor system input in order to provide continuous authentication of the user in front of it with respect to a known legitimate user. The dataset consists of one week's worth of user input in a simulated office environment, including complete keyboard, mouse and web browsing behavior, for a total of 80 users. Authorship verification approaches are natural for this type of problem, as what characterizes the adversary is unknown. The other legitimate users are potential suspects (e.g. employees that share the same work environment and have physical access to their colleague's computer), however effectiveness against external attacks is a desired quality of such security systems. Research of verification methods in the face of this type of incoherent, noisy and inconsistent data is challenging, can greatly contribute to a real-world problem domain, and potentially applicable in other verification related problems.

In addition to the uniquely characterized active authentication dataset, the following datasets are good candidates for evaluation, and some of them are used for evaluations of verification techniques in this thesis:

- *The ICWSM Spinn3r Dataset.* [19] This corpus contains a set of 44 million blog posts made between August 1st and October 1st, 2008. The posts include the text as syndicated, as well as metadata such as the blog's homepage, timestamps, etc. This dataset has been previously used in internet scale authorship attribution [80]. The large number of authors in this dataset provides unique challenges for stylometric techniques, and verification approaches in particular. This dataset is also evaluated in a verification context in Ch. 6.

- *The Extended Brennan-Greenstadt Adversarial Corpus.* [16] The EBG corpus contains writings of 45 different authors, with a minimum of 6,500 words per author divided into documents of approx. 500 words in length. It also contains adversarial documents, where the authors deliberately change their writing style either by hiding it (obfuscation attack), or imitating another author (imitation attack). These unique adversarial settings allow stressing the abilities of verification methods and test how well they perform in the face of a deliberate hiding attempt. This dataset is evaluated in Ch. 6.

- *The Enron Email Dataset.* [60] This corpus contains email communication between employees of the now defunct Enron corporation and was released during a legal investigation of the corporation. The dataset contains emails by 158 users, and has been used extensively for various research purposes.

- *The Ad-hoc Authorship Attribution Competition Corpus.* [54] The AAAC was a stylometry evaluation experiment presented in the 2004 International Conference of the Association for Computers and the Humanities. It contains well-defined problems over a total of 63 authors, spanning over various languages, genres and lengths. Since it has been used to evaluate and compare various stylometric methods, the distractorless verification discussed in Sec. 2.3.2 included, it is a leading candidate for future evaluation of verification approaches.

In addition to the above, other interesting datasets that have been used for stylometric evaluation exist, which consist of tweets, emails, blogs, literary works etc. The corpora above are but selected samples, that have been successfully used for evaluation of one or more stylometric methods, and thus provide a variety of focuses that allow testing different characteristics and uses for authorship verification methods.

Throughout the rest of this document, we revisit various datasets and applications that were approached before in research, some of which are listed above, and examine new applications such as the biometric authentication problem. In Ch. 4 we present a native-language identification algorithm that involves verification over a dataset of English texts by authors of foreign-native. In Ch. 5 we evaluate the Linguistic Active Authentication dataset. We finalize by examining our *Classify-Verify* algorithm in Ch. 6, evaluated over the Extended Brennan-Greenstadt Adversarial corpus, datasets derived from the ICWSM Spinn3r dataset, and again the Active Linguistic Authentication dataset. All the datasets evaluated in this document attempt (and some succeed) to outperform the techniques originally used over these various datasets, demonstrating the effectiveness of verification approaches and strengthening the empirical foundations of research in the field.

Fusion of Classification and Verification, and of Verification Methods

An additional direction to examine is the classification-verification hybrid question, mentioned in Sec. 2.2.1: given a document D of unknown authorship and documents by a set of known authors $\mathcal{A}$, determine the author $A_i \in \mathcal{A}$ of D, *or* that D's author is not in $\mathcal{A}$. The mixture of open- and closed-world settings is highly applicable to real-world scenarios, where we might have a potential suspect list, but would want to avoid false accusations if the real suspect is not among them.

In addition to classification-verification hybrid, combinations should be attempted within the realm of verification methods, utilizing decision fusion architectures for ensemble learning. A mixture-of-experts approach has the potential to increase the overall performance over the standalone verifiers. This idea is demonstrated with PSMS, as discussed in Sec. 2.3.3, in a macro fashion where different classifiers are selected for different authors. However, a micro approach that fuses several verifiers for each author can benefit from an abundance of algorithms even further. Preliminary work with decision fusion of different modalities (keystroke patterns, mouse movement behavior, stylometry etc.) evaluated on the Active Linguistic Authentication dataset supports this idea [36, 37], and motivates utilizing such approach within the authorship verification domain.

The *Classify-Verify* algorithm, presented in Ch. 6 and the main contribution of this thesis, is designed to solve precisely the hybrid problem above, by combining closed-world and open-world techniques. It demonstrates the effectiveness of augmenting traditional closed-world classification procedures with one-sided binary verifiers, enabling to identify low-confidence decisions that indicate a possible misclassification. *Classify-Verify* is also evaluated on the "micro" hybrid approach – fusion of several authorship verifiers for the open-world phase of the algorithm.

Privacy, Security and Adversarial Stylometry

The role of stylometry in privacy and anonymity is a field of increasing interest [90], and introduction of open-world techniques such as verification impose even bigger threats on pseudonymity and the privacy of Internet users. The effects and usability of verification methods should be measured and accounted for in research that examines the privacy risk computational stylometry techniques introduce on one hand, and the effectiveness in tackling open-world problems on the other.

Privacy-oriented stylometry research did not remain untouched, and countermeasures against stylometry have been shown effective using active circumvention [16, 76]. The performance of traditional classifiers in the face of an adversary that deliberately changes his style, have been shown to drop down to levels of random chance. The approach of verification methods, that doubt the legitimacy of possible candidates, as opposed to classical stylometry that always picks the best (or least-worst) candidate, is naturally formulated to thwart attacks.

This theory regarding the performance of open-world verifiers in the face of an adversary is demonstrated in Ch. 6, where we evaluate our *Classify-Verify* algorithm with the Extended Brennan-

Greenstadt Adversarial corpus. The algorithm is validated as effective in identifying attacks/active circumvention, proving the importance of verification methods in security and privacy problems in the stylometry domain.

2.5 Conclusions

Stylometry is a wide field of research that lays many interesting and important directions for computational linguistics, AI and security research, with extensive foundations and a great deal of background work. Authorship verification is an important subdomain, that poses great challenges, with high applicability to real-world security and privacy problems. Stylometry provides a fertile ground of datasets and problems, which draws research on verification methodologies to the field, that can later be generalized and transferred to other domains.

The survey presented in this chapter illustrates how effective current authorship verification methods are for a variety of problems in the field. However, these methods only scrape the surface, and much is yet to be explored. Mainly, empirical experimentation, classification/verification mixture, ensemble learning and the increasing interest in stylometry-related privacy and security problems, motivate authorship verification research and unfold important and practical directions to explore.

In the following chapters we examine some of the ideas identified in this chapter as important directions for future research of authorship verification. We demonstrate the gain of utilizing authorship verification approaches in varying applications and domains. Specifically, we focus on showing the effectiveness that can be attained from combining verification approaches with traditional classifiers, the foundation of the *Classify-Verify* algorithm presented in Ch. 6. We demonstrate the effectiveness of verification techniques when used in scenarios such as open-world applications, adversarial settings and the like, and discuss privacy and security implications of these applications.

3. JStylo: an Authorship Attribution Framework

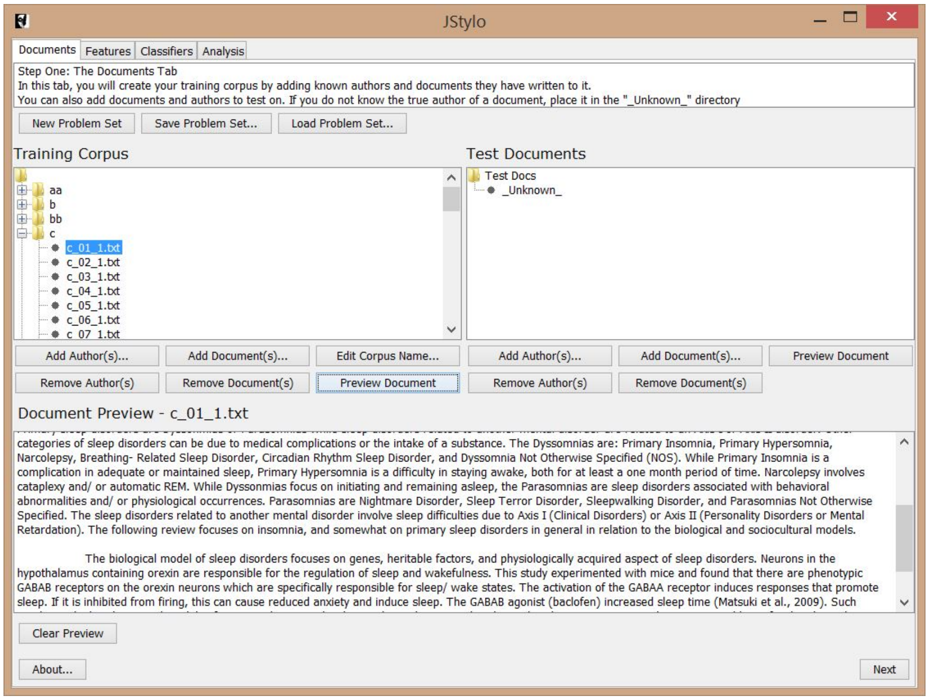
JStylo is an open-source Java framework for authorship attribution research, developed as part of this thesis at the Privacy, Security and Automation lab at Drexel University [76]. The author of this thesis is the developer of the first version of JStylo, released in early 2012, and also a primary contributor to later versions. The ownership of the JStylo repository and development has since been transferred to other members of PSAL. More information can be found at the PSAL website: `http://psal.cs.drexel.edu/`.

The JStylo framework uses natural language processing tools and packages to mine text for linguistic features, and supervised machine learning methods for classification of documents based on those features. JStylo allows loading anonymous documents to be classified, and documents of candidate authors. It then mines the texts in order to learn the style of the candidate authors and the test documents, and matches the latter to the authors they best fit stylistically. JStylo provides a graphic user interface and a Java API to be extended or used in large-scale projects. JStylo uses the JGAAP authorship attribution framework API [53] as baseline, and extends its abilities, customizability and feature extraction variety. The main workflow of JStylo consists of four consecutive phases, as described next: defining a problem set, defining a feature set, selecting classifiers and running the analysis.

Problem Set. A problem set is defined by a training corpus, constructed of documents of all candidate authors (supervised learning), and a set of anonymous documents whose authorship is to be determined. It can be exported to and imported from a simple extensible markup language (XML) file. The set of test documents is optional; users can apply a cross-validation analysis on the training documents. A screenshot of JStylo at the problem set definition phase is presented in Fig. 3.1.

Feature Set. A feature set is defined by a set of various stylistic features to be extracted from the text in order to model the style of its author. JStylo provides over 50 different configurable features, spanning over different levels of the text, including parts-of-speech, function words, word or character n-gram frequencies etc. JStylo supports several predefined feature sets, including a close formulation of the extensive feature set used by the *Writeprints* algorithm [1].

Each feature is defined by text preprocessors/filters, applied prior to feature extraction (e.g. punctuation removal, whitespace canonization); the “core” of the feature which is the feature ex-



Step One: The Documents Tab
In this tab, you will create your training corpus by adding known authors and documents they have written to it.
You can also add documents and authors to test on. If you do not know the true author of a document, place it in the "_Unknown_" directory

Document Preview - c_01_1.txt

categories of sleep disorders can be due to medical complications or the intake of a substance. The Dyssomnias are: Primary Insomnia, Primary Hypersomnia, Narcolepsy, Breathing- Related Sleep Disorder, Circadian Rhythm Sleep Disorder, and Dyssomnia Not Otherwise Specified (NOS). While Primary Insomnia is a complication in adequate or maintained sleep, Primary Hypersomnia is a difficulty in staying awake, both for at least a one month period of time. Narcolepsy involves cataplexy and/ or automatic REM. While Dyssonmias focus on initiating and remaining asleep, the Parasomnias are sleep disorders associated with behavioral abnormalities and/ or physiological occurrences. Parasomnias are Nightmare Disorder, Sleep Terror Disorder, Sleepwalking Disorder, and Parasomnias Not Otherwise Specified. The sleep disorders related to another mental disorder involve sleep difficulties due to Axis I (Clinical Disorders) or Axis II (Personality Disorders or Mental Retardation). The following review focuses on insomnia, and somewhat on primary sleep disorders in general in relation to the biological and sociocultural models.

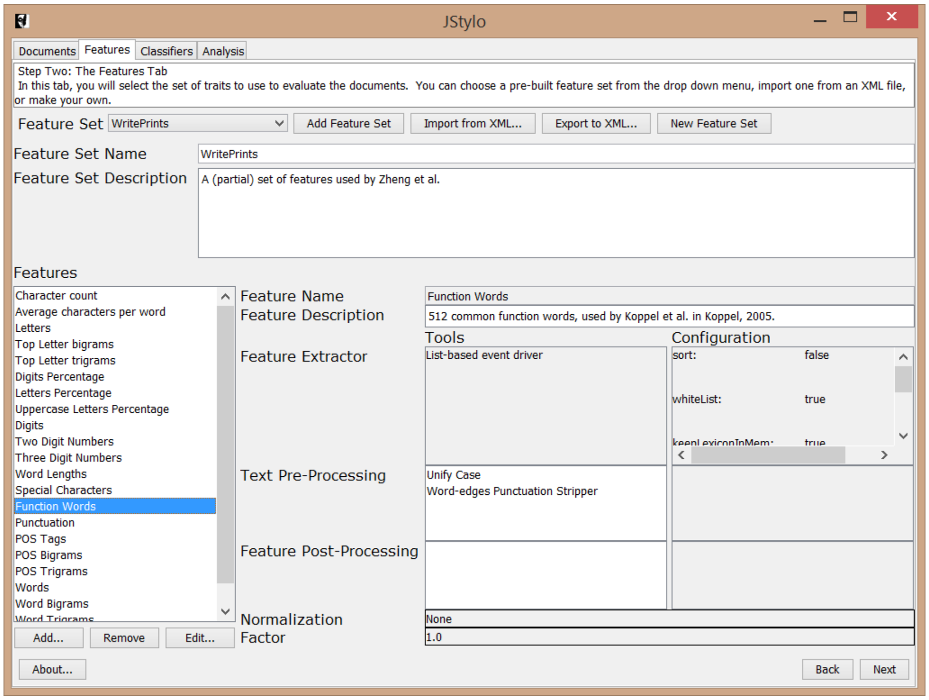
The biological model of sleep disorders focuses on genes, heritable factors, and physiologically acquired aspect of sleep disorders. Neurons in the hypothalamus containing orexin are responsible for the regulation of sleep and wakefulness. This study experimented with mice and found that there are phenotypic GABAB receptors on the orexin neurons which are specifically responsible for sleep/ wake states. The activation of the GABAA receptor induces responses that promote sleep. If it is inhibited from firing, this can cause reduced anxiety and induce sleep. The GABAB agonist (baclofen) increased sleep time (Matsuki et al., 2009). Such

Figure 3.1: JStylo step 1: problem set definition, where training documents are defined grouped under their respective authors, and test documents are optionally added.

tractor itself; post-processors to be applied after extraction (e.g. picking the top features frequency-wise); and applying normalization or factoring (e.g. normalizing over the total number of words). Most of these components are based on the JGAAP API. A screenshot of JStylo at the feature set definition phase is presented in Fig. 3.2.

Classification and Analysis. The classifiers available in JStylo include a subset of Weka [43] classifiers commonly used, such as support vector machine, Naïve Bayes, decision tree, etc. In addition, JStylo provides an implementation of the *Writeprints* algorithm, due to its successful performance as shown in [1]. The analysis can be run either as k-fold cross validation over the training set, or train models with the training set and test the documents in the test set. Screenshots of JStylo at the classifier definition and analysis phases are presented in Fig. 3.3–3.4.

The complete authorship attribution flow in JStylo is illustrated in Fig. 3.5. The main advantages of JStylo over other authorship attribution platforms are in allowing integration of multiple features to represent various stylistic characteristics of documents in one style model (vector), and in the high level of feature-set customizability (per-feature preprocessors, feature extractors and postprocessors). Its user-friendly graphic interface and Java API allow a high level of usage by



Step Two: The Features Tab
In this tab, you will select the set of traits to use to evaluate the documents. You can choose a pre-built feature set from the drop down menu, import one from an XML file, or make your own.

Feature Set: WritePrints

Feature Set Name: WritePrints

Feature Set Description: A (partial) set of features used by Zheng et al.

Feature Name: Function Words

Feature Description: 512 common function words, used by Koppel et al. in Koppel, 2005.

Feature Extractor: List-based event driver — sort: false; whiteList: true; keepLexiconInMem: true

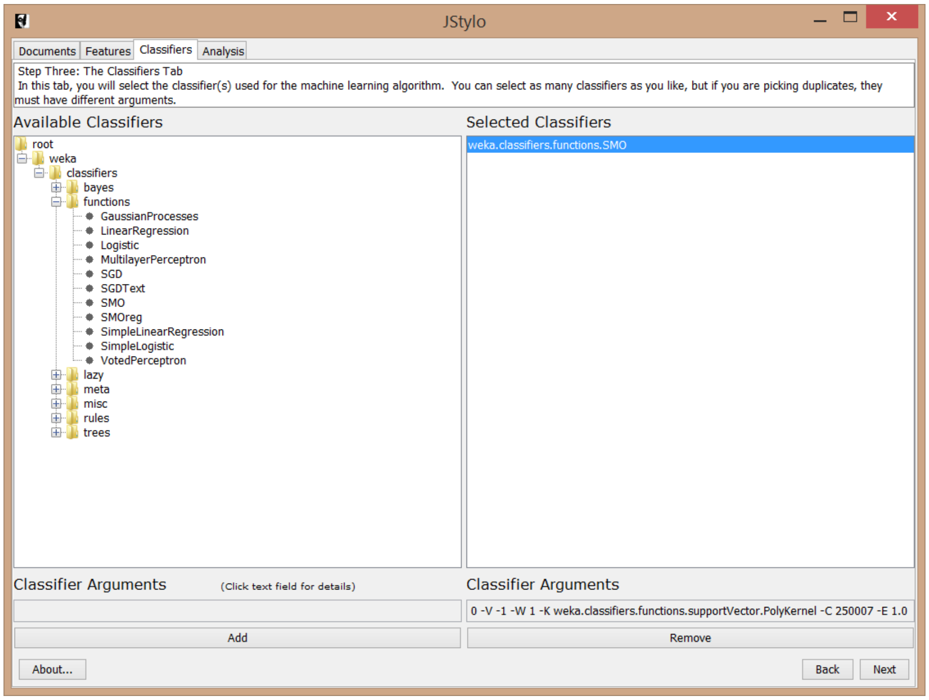
Text Pre-Processing: Unify Case; Word-edges Punctuation Stripper

Feature Post-Processing:

Normalization: None

Factor: 1.0

Figure 3.2: JStylo step 2: feature set definition, where features are combined, each defined with its own extractor, pre/post-processors, normalization and factorization.

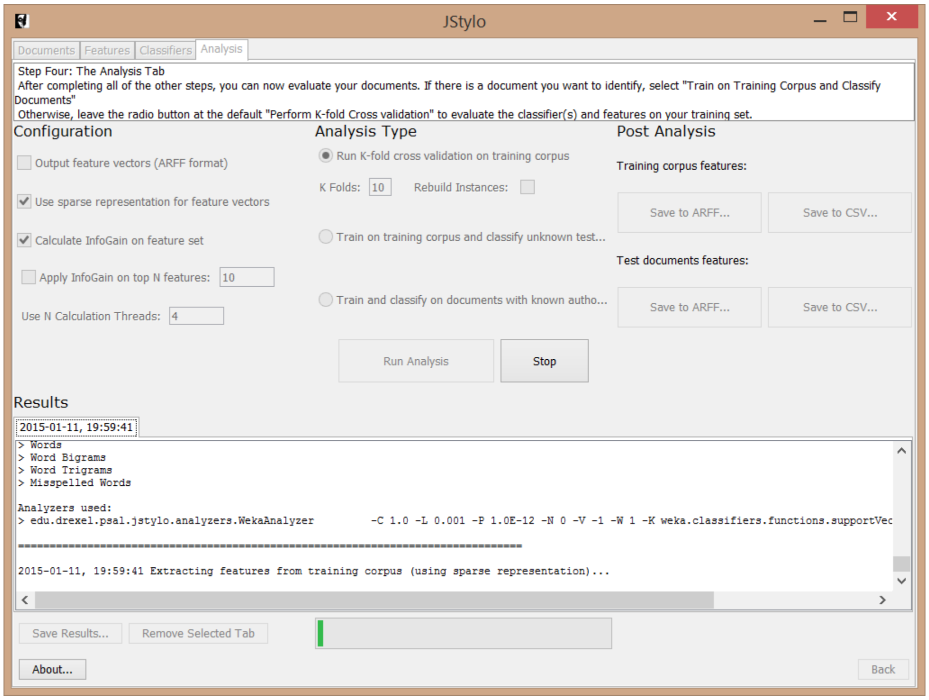


Step Three: The Classifiers Tab
In this tab, you will select the classifier(s) used for the machine learning algorithm. You can select as many classifiers as you like, but if you are picking duplicates, they must have different arguments.

Selected Classifiers: weka.classifiers.functions.SMO

Classifier Arguments: 0 -V -1 -W 1 -K weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0

Figure 3.3: JStylo step 3: classifier selections, where classifiers are defined and configured for the learning phase.



Step Four: The Analysis Tab
After completing all of the other steps, you can now evaluate your documents. If there is a document you want to identify, select "Train on Training Corpus and Classify Documents"
Otherwise, leave the radio button at the default "Perform K-fold Cross validation" to evaluate the classifier(s) and features on your training set.

Configuration

- ☐ Output feature vectors (ARFF format)
- ☑ Use sparse representation for feature vectors
- ☑ Calculate InfoGain on feature set
- ☐ Apply InfoGain on top N features: 10
- Use N Calculation Threads: 4

Analysis Type

- ● Run K-fold cross validation on training corpus
- K Folds: 10 Rebuild Instances: ☐
- ○ Train on training corpus and classify unknown test...
- ○ Train and classify on documents with known autho...

Post Analysis

Training corpus features:

Test documents features:

Results

2015-01-11, 19:59:41

```
> Words
> Word Bigrams
> Word Trigrams
> Misspelled Words

Analyzers used:
> edu.drexel.psal.jstylo.analyzers.WekaAnalyzer        -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K weka.classifiers.functions.supportVec

================================================================================

2015-01-11, 19:59:41 Extracting features from training corpus (using sparse representation)...
```

Figure 3.4: JStylo step 4: the analysis phase, where resource configurations are set and the analysis type is defined and run.

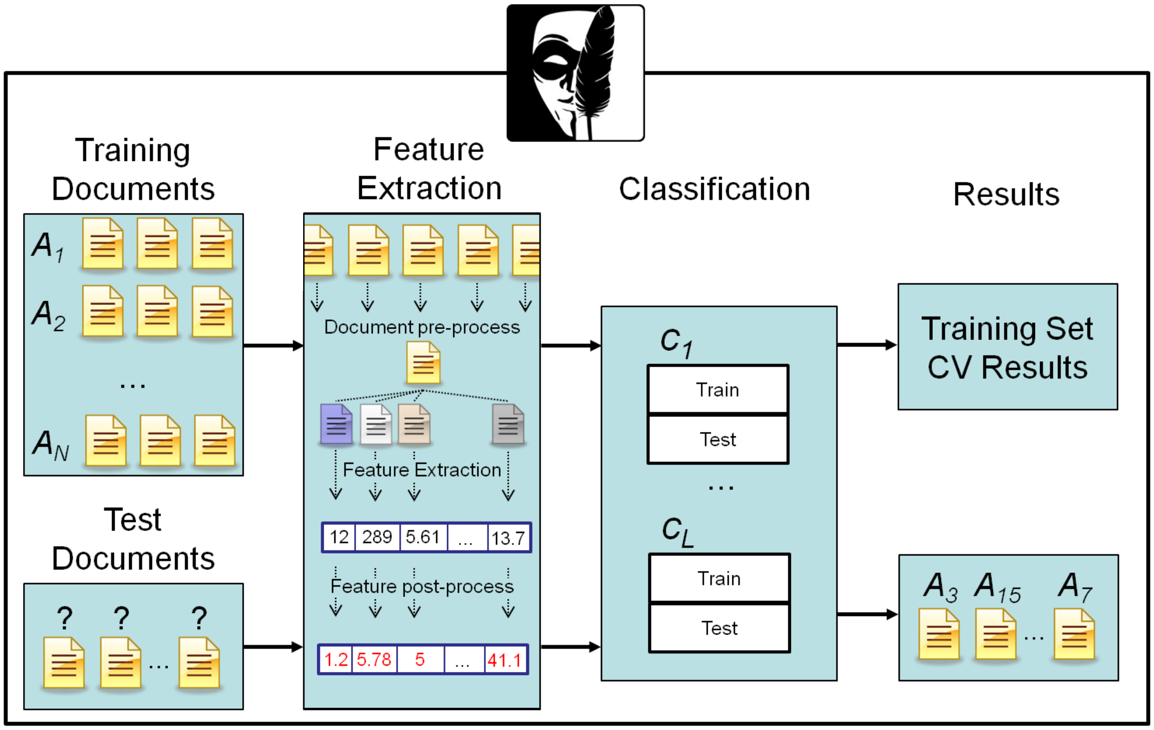


Figure 3.5: A flow of the authorship attribution process in JStylo.

both linguistic researchers and computer scientists, providing a convenient platform for stylometry research.

In addition to a standalone authorship attribution framework, JStylo is also the attribution engine of Anonymouth [76, 77], an authorship anonymization platform, designed to help anonymizing texts and thus assist online users in maintaining their content anonymity (whereas anonymity at the network level can be maintained by onion routing systems like Tor [31]). A workflow in Anonymouth includes initialization with a "camouflage" corpus – a set of texts by authors to blend the target document's style into, a set of texts by the author to learn his/her style, and the target document to be anonymized. Anonymouth aims to provide suggestions that help diverting the target document's style towards the other texts such that its style is indistinguishable from theirs (or more precisely, an average thereof), with respect to some given feature set and classification method. The rest of the process is iterative: using JStylo to apply attribution on the document with the "blend" dataset and author texts as training data; output the probability of the document being written by the author vs. any of the other "blend" authors, expressed as anonymity level; provide suggestions for variations to better anonymize the document; and repeat until the document is sufficiently anonymized.

The JStylo and Anonymouth code bases are maintained on GitHub, at `https://github.com/psal/jstylo` and `https://github.com/psal/anonymouth`, respectively. JStylo is used for authorship attribution analysis throughout all of the experiments presented in this document.

4. Native Language and Language Family Identification

*** This work was completed with support from Aylin Çalişkan İslam.* [101]

Mining text for features to infer characteristics on its author is a well-researched area. One author property that has been researched is native language, extracted from the author's writing in a non-native language. Identifying native language is applicable in security and privacy, as it can reveal traits of an anonymous author. Learning the native language of an anonymous author can assist in profiling criminals or terrorists, but it may also undermine the privacy of legitimate anonymous authors by helping to unveil their identity.

Influences of native language (L1) on second language (L2), referred as the L1-L2 transfer effect, is seen in writing and can be utilized to identify native language. In this chapter we examine aspects of a broader class – the language family to which the native language of an author belongs. In the rest of the chapter, native language and native language family will be referred as L1 and LF, respectively.

First, we examine the correct classification rates of LF compared to L1. As L1 is a subset of LF, the number of L1 classes is greater than or equal to the number of corresponding LF classes. Therefore, higher LF classification accuracy can be trivially achieved by taking the family of the attributed L1 in a L1 classification task. This can be helpful in cases where high accuracy is preferred over classification granularity. We introduce a novel, improved method that achieves higher correct classification rate for LF identification, compared to the trivial method.

The main contribution presented in this chapter is in showing how L1 identification accuracy can be increased by incorporating family information via LF identification, in a two-step classification approach that incorporates verification. In our two-step classification algorithm, we examine for the first time the idea of combining a verification approach with traditional closed-world classification: the verification test is applied after an initial L1 classification, and determines whether additional information in the form of LF identification should be extracted to gain a decision with higher certainty in a second classification step. Our algorithm is shown effective in increasing the overall accuracy of the L1 identification application, and thus a successful hybrid of verification and classification approaches.

We use stylometric analysis and machine learning techniques to identify L1 and LF. We conduct a series of experiments by mining English text written by non-native English authors for linguistic features. We use 4 different feature sets detailed in Sec. 4.3: the first is primarily based on features used in past L1 identification research; the second is the same as the first, with additional grammatical features; the third is only the grammatical features added to the second set; and the fourth is the top weighted features extracted from the second set. We evaluate performance of the attained results by examining the accuracy – true-positive rate.

Sec. 4.1 provides background and summarizes prior work. Sec. 4.2 describes the corpus used for evaluation. Sec. 4.3 describes the configuration used for feature extraction and classification. Sec. 4.4 discusses the different experimental configurations and their evaluation. We finalize with discussion on the attained results in Sec. 4.5, followed by conclusions and directions for future research in Sec. 4.6.

4.1 Background

Introductory studies in the area identify the written or spoken language itself, focusing on telephone dialogue corpora [5, 119]. Further studies focus on extracting specific information from text or speech after identifying the language being used. Wanneroy et al. [113] investigate how non-native speech deteriorates language identification and use acoustic adaptation to improve it. Choueiter et al. [25] classify different foreign accented English speech samples by using a combination of heteroscedastic LDA and maximum mutual information training. Tomokiyo and Jones [105] characterize part-of-speech sequences and show that Naïve Bayes classification can be used to identify non-native utterances of English.

The first work to utilize stylometric methods for native language attribution is introduced by Koppel et al. [69, 71]. They explore frequencies of sets of features, and use them with multi-linear support vector machines to classify text by the author's native language. They use a set of features consisting of function words, letter n-grams, errors and idiosyncrasies, and experiment on a dataset of authors of five different native languages taken from ICLEv1 [40], reaching to 80.2% accuracy. Tsur and Rappoport [107] revisit Koppel's work using only the 200 most frequent character bigrams, and achieve 65.6% accuracy, with only a small degradation when removing dominant words or function words.

Brooke and Hirst [18] present a method of utilizing native language corpora for identifying native language in non-native texts. They use word-by-word translation of large native language corpora to create sets of second language forms that are possible results of language transfer, later used in unsupervised classification. They achieve results above random chance for L1 identification, however insufficiently accurate.

More related work can be found in [6, 17, 22, 34, 39, 110, 114, 115]. The work mentioned above and the approach taken in this chapter both utilize the L1-L2 transfer effect to gain information about an author's native language. Gibbons proved the impact of native language *family*'s typological properties on L2 [38]. This work is the first to combine stylometry and native language family's effect on L2, utilized for L1 identification.

4.2 Corpus

We use the ICLEv2 [41] corpus that contains English documents written by intermediate to advanced international learners of English, with language backgrounds of 16 mother-tongues. The first version of the corpus was used in significant previous work [69, 71, 107]. They were able to use 258 documents of sizes 500-1000 words for each language. We use version 2 of the corpus and restrict all documents in our experiments to those with 500-1000 words as well. However, we found that constraining our documents to these lengths allows us to use only 133-146 documents per language. We conduct a series of experiments with different sub-corpora constructed of documents representing 11 native languages out of the 16 available in the corpus. The native languages we use are: Bulgarian, Czech, Dutch, French, German, Italian, Norwegian, Polish, Russian, Spanish and Swedish, all Indo-European languages. These languages represent 3 language-families in a coarse partition: Germanic, Slavic and Romance, which are used as the LF classes in the experiments to follow. All sub-corpora configurations are detailed in Sec. 4.4.

Since we are looking at a set of languages from both L1 and LF aspects, we maintain only the sub-corpora that allow a sufficient amount of languages in each represented family, i.e. 3 languages in each of the Germanic, Slavic and Romance families. Therefore we removed 5 of the 16 available languages in the corpus.

4.3 Methodology

4.3.1 Feature Selection

Koppel et al. represent each document as a 1,035-dimensional feature vector, consisted of 400 function words, 200 most frequent letter n-grams, 185 misspellings and syntactic errors and 250 rare POS bigrams. The 250 rare POS bigrams are the least common bigrams extracted from the Brown Corpus [35], and their appearances are considered to be erroneous or non-standard. In our experiments we use 4 different feature sets, partially based on that used by Koppel et al., configured as follows:

Basic: includes the 400 most frequent function words, 200 most frequent letter bigrams, 250 rare POS bigrams and 300 most frequent spelling errors. The 400 most frequent function words were taken from a list of 512 function words used in the original experiments by Koppel et al. For the 200 letter n-grams, we choose *bi*grams, as they are shown to be effective for the task in previous research. The 250 rare POS bigrams are extracted from the Brown Corpus using the POS tagger by Toutanova et al. [106]. Finally, we simplify the error types by considering only misspelled words, based on a list of 5,753 common misspellings, constructed from Wikipedia common misspellings and those used for the *Writeprints* stylometric similarity algorithm [1]. We ignore any misspellings with less than 2 appearances across the entire sub-corpus. Since many of the rare POS bigrams and misspellings have no appearances, the effective vector lengths vary between 653-870 features. Total number of features, rare POS bigrams and spelling errors for the experimental variations are detailed in Tab. A.1.

Extended: identical to the former, with the addition of the 200 most frequent POS bigrams across the entire sub-corpus used for each experiment. These syntactic features are selected as an additional representation of grammatical structures in the text. There are several methods for natural language classification, including genetic, typological and areal [20]. We consider the typological classification that uses structural features to compare similarities between languages and classify them into families. Therefore we choose grammatical evidence in L2 as features that may represent similar transfer effects among languages in the same family.

Grammatical: consists of only the 200 most frequent POS bigrams, representing the grammatical level of the text.

InfoGain: We use the 200 features with the highest information gain extracted from the extended feature set using Weka [43], calculated for any given feature by measuring the expected reduction in entropy caused by partitioning the test instances according to that feature. Feature-type distributions for the experimental variations are detailed in Tab. A.2.

4.3.2 Classifier

We train a sequential minimal optimization support vector machine (SVM) classifier [88] with a linear kernel. SVMs are chosen as they are used extensively in prior work and outperform other methods tested, including decision trees, nearest-neighbors, Bayesian and logistic regression classifiers.

4.4 Evaluation

We conduct 5 different experiments using various sub-corpora and the 4 feature sets described previously, with L1 and LF classification tasks. We evaluate the results by using the true-positive rate to capture accuracy. The figures in the following sub-sections use the labels *L1* and *LF* as before, and *Random* for random chance in that specific experiment (e.g. 20% for a 5-class task). Any numbers in parentheses represent the number of classes for that label's corresponding experiment (e.g. *L1 (9)* refers to a 9-class classification task). Following is a detailed description of the different variations, and the attained results.

4.4.1 5-Class Languages

In order to achieve baseline results for L1 identification, we experiment with a 5-language dataset similar to that used by Koppel et al., constructed of documents of Bulgarian, Czech, Russian, Spanish and French native authors. Our restriction to 500-1000 word long documents allow us to use only 139 documents per language, randomly sampled. With this experiment we aim to evaluate our chosen feature sets for L1 identification, to set a baseline for the experiments to follow. We run 10-fold cross validation, also used for the experiments in Sec. 4.4.2, Sec. 4.4.3 and Sec. 4.4.5.

The extended feature set yields the highest accuracy: 77.26%. The accuracy for all other feature sets are 75.39%, 70.36% and 54.96% for the basic, InfoGain and grammatical feature sets, respectively, as illustrated in Fig. 4.1.

4.4.2 9-Class Languages, 3-Class Families

Once a baseline for L1 is established, we continue to LF identification. In this experiment we compare 9-L1 identification with the corresponding 3-LF identification. For the 9-L1 task we randomly sample documents of 9 languages, 3 for each of the Germanic, Slavic and Romance language families, in order to maintain the same number of languages per family in every experiment. We

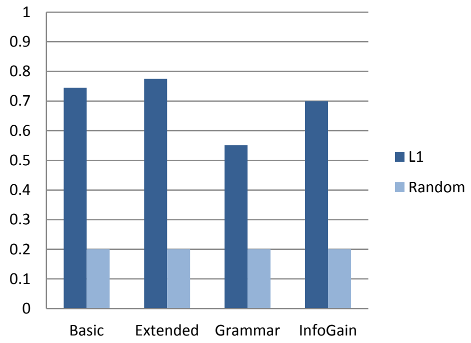


Figure 4.1: Accuracy for 5-class L1 identification.

construct 16 different 9-L1 sets, choosing 3 out of 4 Germanic languages, 3 out of 4 Slavic languages and the only 3 Romance languages available. In each experiment we use the same number of documents per language, varying between 133-146.

For LF identification we conduct 3 sets of experiments, each containing 16 3-LF experiments, corresponding to the 16 that were performed for L1 identification.

First, we run the trivial experiment of attributing the family of the predicted language resulted from the L1 identification experiments. This method is denoted as the *trivial* method.

Next, we run the same experiments conducted for L1, with the only difference of using LF as the class rather than L1. For instance, any instance previously labeled as *Italic*, *French* or *Spanish* is now labeled as *Romance*. As a result of that configuration, each experiment also contains the same number of documents per LF, varying between 399-438. This method is denoted as the *standalone* method.

Lastly, we run experiments combining the standalone and trivial approaches. We hypothesize that if L1 is attributed with high confidence, so is the LF of that attributed L1; however, if the confidence level decreases, a standalone LF experiment achieves better results. We run the L1 identification experiments and set a threshold as the averaged probability of the predicted class across the entire test set, based on the class probability distribution outputted by the SVM classifier. To obtain proper probability estimates, we fit logistic regression models to the outputs of the SVM. Every instance classified with probability above the threshold is attributed the family using the trivial method, and every instance below – using the standalone method. This two-step classification method is denoted as the *combined* method, and it uses traditional closed-world classification with an intermediate verification step to determine the type of decision to perform on the second step –

accepting the trivial LF from the outputted L1, or performing a separated LF experiment.

Averaged results for the L1 and LF (trivial, standalone and combined) identification experiments are illustrated in Fig. 4.2. The accuracy for L1 identification is 67.78%, 65.64%, 59.34% and 44.02% for the extended, basic, InfoGain and grammatical feature sets, respectively. Out of the 3 LF identification methods, the *combined* method achieved the best accuracy, supporting our hypothesis, with 90.57%, 86.24%, 86.2% and 85.29% for the InfoGain, grammatical, extended and basic feature sets, respectively.

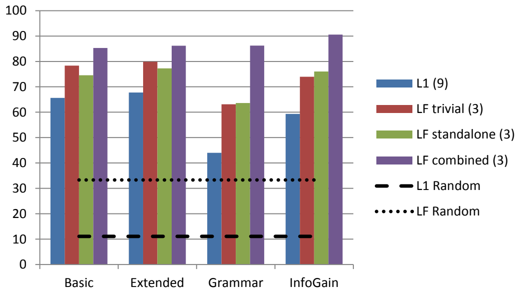


Figure 4.2: Accuracy for 9-class L1 and 3-class LF identification. The combined method for LF outperforms the other two.

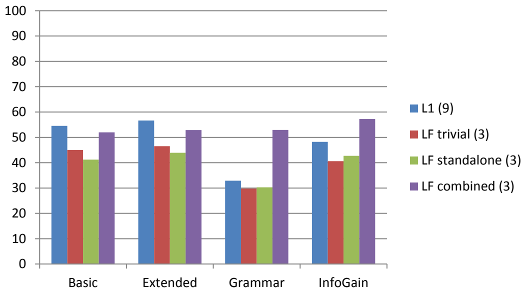


Figure 4.3: Effective accuracy for 9-L1 and 3-LF identification. Accuracy for L1 exceeds most accuracy results for LF, except for the combined method on the grammatical and InfoGain feature sets.

Although it seems LF identification outperforms L1, if we measure the added knowledge over the corresponding random classification rates ($1/9 = 11.1\%$ for L1, $1/3 = 33.3\%$ for LF) to get the *"effective"* accuracy, illustrated in Fig. 4.3, L1 is more accurate in most cases. The LF combined method is the only one that exceeds the effective accuracy of L1, with the grammatical and InfoGain feature sets. Combined with the standard (non-effective) results, it appears that the InfoGain feature set with the LF combined method achieves the highest accuracy with the most added knowledge over random classification. The smallest difference between L1 and LF identification accuracy is seen for the grammatical feature set.

4.4.3 3-Class Languages, 3-Class Families

In order to have the same random-chance baseline for both L1 and LF tasks, we compare 3-L1 with 3-LF identification, using the same sub-corpus as before.

For L1 we construct 9 experiments, in each randomly sampling 3 languages from 1, 2 and 3 different language families (3 experiments each). The reason for this choice is that as more families are used, the farther the chosen languages are from one another. Therefore the choice above is intended to balance the effect of LF in those experiments. We use 133 documents per language for all experiments.

For LF we construct 2 sets of 9 experiments, in order to examine the notion that languages in the same family have more family-distinguishable commonalities as opposed to random sets of languages. In the first, we create 3 random sets of languages to be considered as families. We randomly sample documents from all 11 languages to construct sets for the 3 randomly-generated families used as classes. Here we also maintain 133 documents per language family. In the second we run a similar configuration, only using the actual language families.

The averaged accuracy attained for L1 is 84.23%, 82.29%, 81.67% and 66.97% for the extended, InfoGain, basic and grammatical feature sets, respectively. These results, illustrated in Fig. 4.4, show that L1 identification consistently outperform the results of both sets of LF experiments.

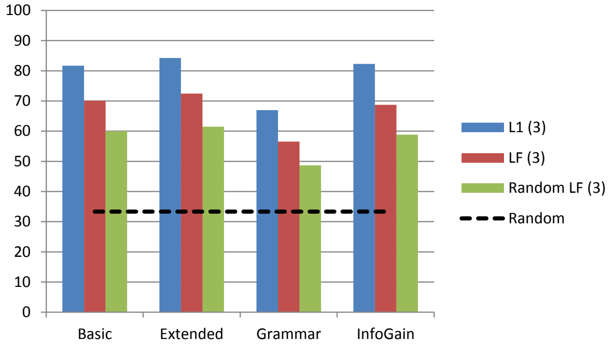


Figure 4.4: Accuracy for 3-L1, 3-LF and 3-randomly-generated families identification. Using the original families achieves the highest accuracy for LF identification.

The accuracy attained for actual language families is 72.43%, 70.09%, 68.72% and 56.55% for the extended, basic, InfoGain and grammatical feature sets, respectively, which consistently outperform that of the randomly-generated families: 61.46%, 60.01%, 58.81% and 48.67%. This confirms that true language families are ideal when grouping L1 into sets. As in the previous experiment, the difference in accuracy between L1 and LF identification was the smallest with the grammatical feature set.

4.4.4 3-Class Families: Train on 2, Test on 1

In order to examine similarities of L1-L2 transfer effects for languages in the same family, we conduct 3 additional 3-class experiments in which we train the classifier on 2 languages from each family (a total of 6 languages) and test on 3 other languages representing the 3 families. Each language in the training and test sets contains 133 documents.

The averaged accuracy obtained is 55.05%, 53.21%, 48.28% and 48.20% for the extended, basic, InfoGain and grammatical feature sets, respectively, as illustrated in Fig. 4.5. These results support the notion that languages in the same family have more commonalities than languages in different families. Moreover, similarities of languages in the same family are distinguishable from similarities of languages in different families.

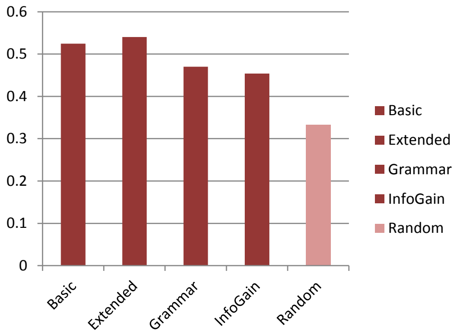


Figure 4.5: Accuracy of training on 2 languages and testing on 1 other language for each LF. Similarities of languages in the same family are distinguishable from similarities of languages in different families.

4.4.5 9-Class Languages, Reclassify by Family

This experiment, the main contribution presented in this chapter, illustrates how LF classification can improve L1 classification by incorporating verification in a two-step classification process. We conduct the same 16 9-L1 experiments as in Sec. 4.4.2. A threshold is then set as in the *combined* method, such that each classified instance with predicted probability less than that threshold is treated as misclassified. For all allegedly-misclassified instances we attribute the family they belong to, using various methods detailed later. As last step we reclassify those instances using a training set constructed only of the 3 languages in the family they are classified as, and consider these results as L1 classification-correction for those instances. We measure the overall change in accuracy.

The experiments are conducted 3 times, each with a different method for LF attribution for the instances below the threshold: 1) The standalone method – running LF identification over all those instances, using the same training set (with LF as classes rather than L1), 2) The trivial method – using the family of the predicted language of those instances, and 3) Random family selection.

We measure the net fix in accuracy – change in the correctly classified instances, taking into account corrected classifications and new misclassifications. For all feature sets, LF attribution using the standalone method yields the highest fix rate, followed by LF attribution using the trivial method. The randomly attributed family method consistently yields negative fix rate, i.e. reduced overall accuracy. See figure 4.6.

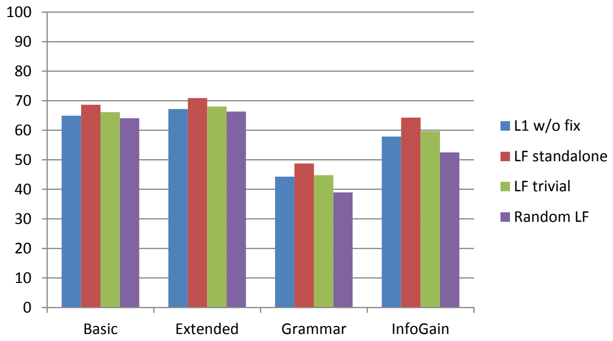


Figure 4.6: Accuracy for L1 identification without fix and with fixing using LF attribution by the standalone method, trivial method and random selection of family. The standalone method yields the highest net fix in L1 classification accuracy.

The extended feature set yields the best results. Starting at a baseline of 67.17% for L1 identification without any fix, the true-positive rates obtained for this feature set are 70.9% and 68.05% for attributing LF by the standalone and the trivial methods, respectively. The increase in accuracy is statistically significant, with $p < 0.01$. The random family attribution method yields a decrease in accuracy to 66.35%.

It is notable that although best results are achieved with the extended feature set, the standalone method achieves higher increase in accuracy in some of the other feature sets. The increase rates for this method are: 6.43%, 4.48%, 3.73% and 3.67% for the InfoGain, grammatical, extended and basic feature sets, respectively.

4.5 Discussion

There are two notable observations regarding using verification approaches in L1/LF identification, where acceptance thresholds are interleaved in the process. The first is seen in Sec. 4.4.2, where using the *combined* method for LF identification derives higher accuracy than both the trivial and the standalone methods. This may suggest that when L1 is predicted with high confidence, LF is predicted well, but when the confidence level is low, it is better to apply standalone LF classification. Since the combined method uses the best of the two others, it outperforms both.

The second and most important observation regarding utilization of verification approaches combined with traditional classifiers is seen in Sec. 4.4.5: L1 identification is improved by up to 6.43% in accuracy for 9-L1 classification by introducing information about the language family, thus providing a smaller set of language classes in which the actual language is more likely to be found. Attributing LF by standalone experiments yields higher L1 classification accuracy than attributing it by the family of the predicted language, which supports the idea that the family of the attributed L1 is the actual family with higher probability than LF attributed by a standalone experiment, only when L1 is attributed with high confidence (i.e. above the selected verification threshold).

The results in Sec. 4.4.2 and Sec. 4.4.3 suggest that all 4 feature sets achieve better accuracy for L1 than for LF (standalone) classification. This may occur since for L1 we attempt to distinguish between individual languages as they transfer to English. However, LF identification necessitates finding features that intersect between languages in a particular family, and distinguish well between different families as they are transferred to English. This makes LF identification a more difficult task.

The results obtained for randomly generated families in Sec. 4.4.3 and Sec. 4.4.5, which are consistently lower than using the actual families, suggest that the contribution of using the latter yields the best performance. That is, languages in the same family have more commonalities distinguishing them from other families, than random sets of languages have. The advantage of partitioning languages into sets by their actual families is supported also by the results in Sec. 4.4.4.

When conducted the experiments in Sec. 4.4.4 we also attempted training on one language and testing on two languages from each family. The results attained were close to random chance, as opposed to the results in Sec. 4.4.4. This might have been due to over-fitting when trained only on one language. By training on two and testing on one we manage to capture more family-wide features, rather than the specific language ones.

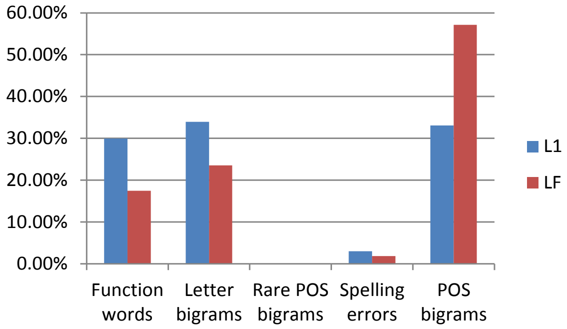


Figure 4.7: Feature-type average percentage distribution for the 3-L1 vs. 3-LF InfoGain feature set.

The slight decrease in accuracy of the basic feature set experiment in Sec. 4.4.1 compared to the results obtained by Koppel et al. [69, 71] (75.39% and 80.2%, respectively) might be caused by two main issues. First, we use only 133-139 documents per language instead of the 258 used originally. Second, we do not consider grammatical error-types, but only spelling errors. By that we may miss capturing some L1-L2 transfer effects that help distinguish between different native languages.

Looking at the results using the different feature sets, in most cases the extended feature set outperforms the rest. This may suggest that adding grammatical features increases accuracy for both L1 and LF. Furthermore, in all experiments using *only* the grammatical features achieves a rather good accuracy (significantly higher than random chance), considering that we use only 200 of these features. This supports the notion that grammatical features are useful for both L1 and LF identification.

Another interesting notion regarding the grammatical feature set is seen in the portion these features consist of the InfoGain feature set for the experiments in Sec. 4.4.3: 33.05% for L1 and 57.16% for LF, illustrated in Fig. 4.7. This suggests that the grammatical level of the text has greater significance for identifying LF compared to L1. When analyzing the portion lexical features consist of the InfoGain feature set, an opposite trend is seen: function words and letter bigrams consist 29.94% and 33.94% of the features for L1, as opposed to 17.44% and 23.55% for LF, respectively. This suggests that analysis of the lexical level of the text is more beneficial for L1 detection than for LF detection. Although less significant, the same trend is seen with spelling errors: 3% for L1 and 1.83% for LF.

4.6 Conclusions

The main conclusion of this chapter is, when trying to gain information about the native language of an English text author, integrating family identification can increase the total accuracy, using the method introduced in Sec. 4.4.5, where all low-confidence classifications are reapplied within a smaller set of candidates – languages within the family attributed to those instances using a standalone experiment. This utilization of LF identification as an intermediate generalization step, determined by setting a threshold over the L1 classification probability distributions, demonstrates how verification-based approaches can be used in stylometry applications to improve the overall results, specifically when integrated in closed-world classification procedures like the above. The confidence measurements may provide knowledge of when additional information should be sought out to improve the baseline classification accuracy.

Furthermore, when dealing with a large number of L1 classes, higher accuracy can be attained by reducing the level of specification to language families, which can be obtained with high accuracy using the combined method, that integrates both the trivial LF by predicted L1 and LF by standalone experiment methods using the average confidence level as threshold.

In addition, using the most frequent POS bigrams, which represent the grammatical level of the text, is shown to increase accuracy in both L1 and LF identification tasks, mostly for the latter. Using lexical features as function words and character bigrams is helpful especially for L1 identification.

We suggest several directions for future work. First, new feature sets that may capture other similarities between languages in the same family should be evaluated. For instance, since languages in the same family tend to share basic vocabulary, it may have some level of transfer to L2 that could be captured by a synonym-based classifier. For instance, “verde” in Spanish and “vert” in French may be translated to “verdant”, whereas “grün” in German and “groen” in Dutch may be translated to “green”.

In addition, the approach of increasing accuracy by applying knowledge of a broader class on the task can be further explored for applications in other stylometry-based information extraction tasks. For instance, using wide age ranges as the broader class for classifying age of anonymous authors, or personality prototypes for personality type identification. Testing such applications in adversarial settings, where authors deliberately attempt to hide those characteristics we are trying to expose, can also be of interest for forensic applications of such trait-recognition methods.

In conclusion, the LF/L1 identification approaches presented in this chapter demonstrate the effectiveness of integration of several levels of categorization of the data we analyze, in order to extract more information about the original classes we are attempting to identify – in this case, author native language. Moreover, these methods demonstrate how utilizing a verification approach in the classification process, which serves as a binary indicator of what approach is to be taken, and how confident we are in the classifier decisions, eventually results in fixing misclassifications (more than creating new ones) that would have been missed otherwise. This approach of integrating authorship verification with traditional closed-world classifiers is revisited in Ch. 6, where we present the *Classify-Verify* algorithm for mixed open and closed-world stylometry.

5. Realtime Stylometric Modalities for Active Authentication

*** This work was completed with support from, and supported the work of Alex Fridman, Sayandeep Acharya, John Noecker Jr., Michael Ryan, Patrick Brennan, Patrick Juola and Moshe Kam.* [36, 37, 56, 57, 102]

Active authentication is the process of continuously verifying a user based on his/her on-going interaction with the computer. The challenge of identity verification for the purpose of access control is the tradeoff between maximizing the probability of intruder detection, and minimizing the cost for the legitimate user in time and distractions due to false alerts, and extra hardware requirements for physical biometric authentication. In recent years, behavioral biometric systems have been explored extensively in addressing this challenge [3]. These systems rely on input devices such as the keyboard and mouse that are already commonly available with most computers, and are thus low cost in terms of having no extra equipment requirements. However, their performance in terms of detecting intruders, and maintaining a low-distraction human-computer interaction (HCI) experience has been mixed [13], showing error rates ranging from 0% [83] to 30% [84] depending on context, variability in task selection, and various other dataset characteristics.

The bulk of biometric-based authentication work is focused on verifying a user based on a static set of data. This type of one-time authentication is not sufficiently applicable to a live multi-user environment, where a person may leave the computer for an arbitrary period of time without logging off. This context necessitates continuous authentication when a computer is in a non-idle state. In particular, to represent this general real-world scenario, we use the Active Linguistic Authentication Dataset [57]. This dataset consists of data collected in a simulated office environment, which contains behavioral biometrics associated with typical human-computer interaction (HCI) by an office worker.

The application of stylometry as a high-level modality for authenticating users in a continuous user verification system is novel; initial evaluation of authorship attribution technologies are proven promising, reaching more than 90% identification accuracy over 14 users [57], as detailed in Sec. 5.3.2.

In this chapter, we consider a set of stylometric classifiers, also referred to as sensors, as a representative selection of high-level behavioral biometrics. This work aims to evaluate authorship attribution approaches in realistic settings for active authentication, which require constant monitoring and frequent decision making about the legitimacy of the user at the computer in a dynamic and time-sensitive environment. Moreover, this work is designed as a preliminary evaluation of one modality among many to consider for an active authentication system. The main goal for these stylometric modalities is to be interleaved with other low- and high-level modalities, such as keyboard dynamics [97], mouse movements [4], web browsing behavior [116] and the like, in one centralized decision fusion system. Usage of such modalities, stylometry included, can provide a cost-effective alternative to sensors based on physiological biometrics [49].

Although this work is targeted for active authentication, a live security application of stylometric analysis, its implications on the usability and configuration of stylometric sensors are relevant for forensic contexts as well: consider a standard post-mortem forensic analysis of user input data aggregated throughout an entire day; this work lays grounds for what features to look at in such "noisy" settings, the size of windows to look at, the effects of looking at overlapping windows, how idle periods in data input should be considered, etc.

The remainder of the chapter is structured as follows. Sec. 5.1 reviews background and related work. Sec. 5.2 discusses the simulated work environment dataset that we used for evaluation. The stylometry biometrics applied on the dataset are discussed in Sec. 5.3, followed by evaluation in Sec. 5.4. We conclude and discuss directions for future work in Sec. 5.5.

5.1 Background

A defining problem of active authentication arises from the fact that a verification of identity must be carried out continuously on a sample of sensor data that varies drastically with time. The classification therefore has to be made based on a "window" of recent data, dismissing or heavily discounting the value of older data outside that window. Depending on what task the user is engaged in, some of the biometric sensors may provide more data than others. For example, as the user browses the web, mouse and web activity sensors will be actively flooded with data, while keystroke dynamics or stylometric sensors may only get a few infrequent key presses. This motivates the work on multimodal authentication systems where the decisions of multiple classifiers are fused together [98]. In this way, the verification process is more robust to the dynamic mode of real-time HCI.

In this chapter we examine only the effectiveness of stylometry sensors under active authentication settings. The main goal of this work is combination in a multi-modal biometric system (see [36, 56]). The idea of decision fusion is motivated by the work in [8] that greater reduction in error rates is achieved when the classifiers are distinctly different (i.e. using different behavioral biometrics), with several fusion approaches available to be applied [23, 45, 59].

In active authentication settings, authorship verification is applied, where unknown text is classified by a unary author-specific classifier. The text is attributed to an author if and only if it is stylistically close enough to that author. Although pure verification is the ultimate goal, standard authorship attribution as a closed-world problem is an easier (and sometimes sufficient) goal. In either case, classifiers are trained in advance, and used for real-time classification of processed sliding windows of input keystrokes. If enough windows are recognized as an author other than the real user, it should be considered as an intruder. Application of stylometric analysis to this sort of task brings higher level inspection into the process, compared to other lower level biometrics like mouse movements or keyboard dynamics [12, 118].

In pure authorship attribution settings, where classification is done off-line on complete texts (rather than sequences of input keystrokes) and in a supervised setting where all candidate authors are known, state-of-the-art stylometry techniques perform very well. For instance, at PAN-2012[1], some methods achieved more than 80% accuracy on a set of 241 documents, sometimes with added distractor authors.

In active authentication settings, a few challenges arise. First, open-world stylometry is a much harder problem, with a tendency to high false-negative (false reject) rates. Verification techniques such as those discussed in Ch. 2 have shown effectiveness in more standard authorship verification settings. However, the amount of data collected by sliding windows of sufficiently small durations required for an efficient authentication system, along with the lack of quality coherent literary writings may result with these methods performing insufficiently for our goal. Second, the inconsistent frequency nature of keyboard input along with the relatively large amount of data required for good performance of stylometric techniques make a large portion of the input windows unusable for learning writing style.

On the other hand, this type of setting allows some advantages in potential features and analysis method. Since the raw data consists of all keystrokes, some linguistic and technical idiosyncratic features can be extracted, like misspellings caught prior to being potentially auto-corrected and vanished from the dataset, or patterns of deletions (selecting a sentence and hitting *delete* versus repeatedly hitting *backspace* deleting character at-a-time). In addition, it is more intuitive in this kind of setting to consider overlap between consecutive windows, resulting with a large dataset,

[1] `http://pan.webis.de`

grounds for local voting based on a set of windows and control of the frequency in which decisions are outputted by the system.

5.2 Corpus

We utilize the complete Active Linguistic Authentication Dataset (presented initially in [57] while still being collected), a dataset designed specifically for the purpose of behavioral biometrics evaluation, based on data collected in a simulated work environment. For the collection of the data, an office space was allocated and organized. The space contained 5 desks, each with a laptop, mouse and headphones. This equipment and supplies were chosen to be representative of a standard office workplace. One of the important properties of this dataset is that of uniformity. Due to the fact that the computers and input devices in the simulated office environment were identical, the variation in behavioral biometrics data can be more confidently attributed to variation in characteristics of the users, rather than effects of variations in physical environmental settings.

The complete dataset contains data collected from 80 users. Due to crashes in the mouse, keyboard, web browser tracking software, or sick days taken, a few more than 80 subjects participated, to cover the missing data and reach the 80 users goal. However, within the final 80 users data, some users had significantly less data than the rest. In order to eliminate user activity variance effects on our evaluation, we set a threshold of 60,000 seconds minimum activity (16.67 hours). This filtering left us with 67 qualifying users for the evaluation presented in this chapter.

During each week of the data collection, 5 temporary employees were hired for a total of 40 hours of work. Each day they were assigned two tasks. The first was an open-ended blogging task, where they were instructed to write blog-style articles related in some way to the city in which the testing was carried out. This task was allocated 6 hours of the 8 hour workday. The second task was less open-ended. Each employee was given a list of topics or web articles to write a summary of. The articles were from a variety of reputable news sources, and were kept consistent between users except for a few broken links due to the expired lifetime of the linked pages. This second task was allocated 2 hours of the 8 hour workday.

Both tasks encouraged the workers to do extensive online research by using the web browser. They were allowed to copy and paste content, but they were instructed that the final work they produced was to be of their own authorship. As expected, the workers almost exclusively used two applications: Microsoft Word 2010 for word processing and Internet Explorer for browsing the web.

Table 5.1: Character count statistics for the 67-user active authentication sub-corpus across all 5 simulated work days.

Statistic	**Value**
Min. per user	17,027
Max. per user	263,165
Avg.	84,206
Total	5,641,788

Although the user generated documents are available in the dataset, the evaluation in this chapter is based on the stream of keystrokes recorded throughout the work day, with the purpose of simulating the settings which a real-time authentication system will have to work under.

The 67-user dataset is further parsed in order to provide one large stream of mouse/keyboard events. For every user, the entire 5 days of data were concatenated into one stream (in JSON format), and marked to be divided into 5 equally sized folds, for later cross-validation evaluation. In addition, any inactivity period exceeding 2 minutes is marked, to be considered as idle. For the purposes of our evaluations, a subset of events including only keyboard strokes are kept (whereas mouse events are to be used by other sensors [36]). The format of one continuous stream allows to utilize the data to its full in evaluation of a real-time, continuous active authentication system. Keystroke events statistics for the parsed 67-user dataset are summarized in Tab. 5.1. The keystroke events include both the alpha-numeric keys and also special keys such as `shift`, `backspace`, `ctrl`, `alt`, etc. In counting the key presses in Tab. 5.1, we count just the down press and not the release.

5.3 Methodology

5.3.1 Challenges and Limitations

An active authentication system presents a few concerns. First, a potential performance overhead is expected to accompany deployment of such a system, as it requires constant monitoring and logging of user input, and on-the-fly processing of all its sensor components. With stylometric sensors, large amounts of memory and computation power may be consumed by language processing tools (e.g. dictionary based features, part-of-speech taggers etc.), therefore a careful configuration should be applied to balance the tradeoff between the accuracy of the system and its expected resource consumption behavior. This issue becomes more prominent in a multi-modal system, where multiple sensors are used.

Another concern with this type of authentication system is its user input requirements. In non-active authentication schemes, the user is required to provide credentials only when logging in, and perhaps when certain operations are to be executed. The provided credentials consist of some sort of personal key (password, private key etc.), dedicated for the purpose of identifying the system's users. In active authentication systems based on stylometric modalities, all of the user keyboard input is required. In a multi-modal system, all interaction may be required, including mouse events and web browsing behavior. The precise sequence and timing of keyboard events is essential for the system's performance. However, this type of input is not designed for stylometric analysis and authentication, and most probably contains sensitive and private information, collected when the user types in passphrases to log into accounts, writes something personal s/he wishes to keep confidential, or simply browses the web. To cope with these security and privacy issues, some actions can be taken in the design of such a system: the collected data should be managed carefully, by avoiding storage of raw collected data (i.e. save only parsed feature vectors extracted from the data) and use encrypted storage for the data that is stored. The privacy issue specifically applies to stylometric modalities, where the contents of the user input is of importance, and can potentially be highly sensitive.

5.3.2 Initial Evaluation

In [57] we present an initial evaluation of a subset of the Active Linguistic Authentication Dataset, when data of only 14 users was available (as the dataset was still in collection). Two methods of evaluation were applied.

First, each day's worth of work was analyzed as one unit, or document, for a total of 69 documents (5 days for 14 users, minus a missing day by one user). One-vs-all analysis was applied, using a simple nearest-neighbor classifier with Manhattan or Intersection distance metric. The feature sets consisted of character n-grams, with n ranging between 1 and 5. The best configuration resulted in 88.4% accuracy.

In the second analysis, a number-of-characters-based sliding window technique was applied to generate the input segments to be classified, to better simulate the performance of a realistic active stylometric authentication system. The generated windows were non-overlapping, with window sizes set to 100, 500 and 1,000 words (tokens separated by whitespace). The motivation for requiring a minimum window size is in order to allow sufficient data required for stylistic profiling of the window. An extensive linguistic feature set, inspired by that used in the Writeprints [1] stylometric similarity method was used, along with a linear SVM and a nearest-neighbor classifier. The best result achieved was 93.33% accuracy with 0.009/0.067 FAR/FRR.

These reported results are sufficient to determine that using stylometric biometrics for active authentication is beneficial; however, the approach taken in this analysis, although satisfactory as preliminary results, lacks addressing a few key requirements in an active authentication system. First, only 14 subjects were used for the initial analysis, and its performance over a large set of users is yet to be evaluated. Stylometry research has thus far provided solutions for large author sets, but even those were never attempted on a dataset with such incoherent and noisy qualities; the performance of the approaches above may certainly be proven inefficient when the author set increases in size.

Perhaps the main issue with this method of analysis is the units determined for learning and classification. Day-based windows are certainly not useful for active authentication, which aims to provide intruder alert as quickly as possible (in a time frame of minutes, perhaps seconds). Even the second data-based-windows analysis is insufficient: each window may have an arbitrary length in time on which it spans, and collecting the minimum amount of words may allow an intruder enough time to apply his/her attack. Moreover, due to the possibility of time-wise long windows, which may cross idle periods, data of different users can be mixed (e.g. first half of the window is the legitimate user input, whereas the second half, an idle-period later, is by an intruder) causing contamination of “bad” windows with “good” data, which may throw off the classifier and cause it to miss an alert.

In the next section we provide an analysis in a more realistic setting of the authentication system. We focus on a time-wise sliding window (rather than data-wise), and allow overlapping windows in order to provide the system the ability to output frequent decisions. With this approach, the system is compelled to decide whether to accept/reject the latest window in a timely manner, based on the data it has acquired thus far, or to determine it cannot make a decision. A balance between the amount of collected data, the required time-wise size of windows and desired decision frequency is inspected in the following section.

5.3.3 Real-Time Approach

The stylometric classifiers, or sensors, presented in this section are based on the simplest settings of closed-world stylometry: we use classifiers trained on the closed set of all 67 users, where each classification results with one of those users as the author. A more sophisticated approach would use open-world verifiers, where each legitimate user is paired to its own classifier, in a one-class/one-vs-all formulation. Such verification approach is more naturally suited for this open-world scenario, where possible imposters can originate outside the set of legitimate users (e.g. an intruder from outside an office that takes over an unlocked computer, rather than a vicious colleague); however, in this chapter we consider the case of a closed set of possible users, as a baseline for future verification-based classifiers. Another look at the active authentication problem from a verification point of view is taken in Ch. 6.

In the preprocessing phase, we parse the keystrokes data files to produce a list of documents (text windows) consisting of overlapping windows for each user, with the following time-based sizes in seconds: 10, 30, 60, 300, 600 and 1,200. For the first 3 settings we advanced a sliding window with steps of 10 seconds of the stream of keystrokes, and for the last 3 – steps of 60 seconds. The step size determines how often a decision can be made by the sensor. In addition, although the window generation is configured with a fixed time-wise size and step, e.g. $\{300, 60\}$, in practice the timestamps of the generated windows correlate with the keystroke events, by relaxing the generation to $\{\leq 300, \geq 60\}$ (empty windows are discarded.) In a live system a similar approach is expected to be used: a window is “closed” and a decision is made for it when the size limitation time is up, hence ≤ 300. In addition, when determining the beginning of a window followed by another window, a difference of at least one character is expected (otherwise the second window is simply a subset of the first); therefore if the time span between the first character in a window and the one that follows is greater than the determined step size, effectively a greater step size will be applied, hence ≥ 60.

We choose to ignore idle periods within the generated windows, as if the stream of data is continuous with no more than 2 minutes delay between one input character and the next. This is applied in the dataset by preprocessing the keystroke timestamps, such that any idle period longer than 2 minutes is artificially narrowed down to precisely 2 minutes. Furthermore, the data is aggregated and divided into 5 equally-sized folds for analysis purposes, thus potentially contains windows that originally contain an idle period between days. Although this preprocessing suffers from the issues of possible mixed legitimate/non-legitimate user-input windows, or mixed time-of-day windows (e.g. end of one day and beginning of the next) if applied in a real system, in our analysis it is applied to allow generating as many windows as possible. Since the analysis presented here is not applied on legitimate/non-legitimate mixed windows, idle-crossing windows are reasonable for our purposes.

Specifically for stylometry-based biometrics, selecting the size of the window affects a delicate tradeoff between the amount of captured text (and probability for correct stylistic profiling of that window) and response time of the system, whereas other biometrics can perform satisfactorily with small windows (even the size of seconds). This is somewhat overcome by using small steps (and overlapping windows), leaving this as a problem only at the beginning of the day, until the first window is generated. Similar to the analysis in [57], during preprocessing only keystrokes are taken (key releases were filtered out) and all special keys are converted to unique single-character placeholders. For instance `BACKSPACE` is converted to β and `PRINTSCREEN` is converted to π. Any representable special keys like `\t` and `\n` are taken as is (i.e. tab and newline, respectively).

Feature Set

The chosen feature set is probably the most crucial part of the configuration. The constructed feature set, denoted the *AA* feature set hereinafter, is a variation of the *Writeprints* [1] feature set, which includes a vast range of linguistic features across different levels of the text. A summarized description of the features is presented in Tab. 5.2. By using a rich linguistic feature set we are able to better capture the user's writing style. With the special-character placeholders, some features capture aspects of the user's style usually not found in standard authorship problem settings. For instance, frequencies of backspaces and deletes provide some evaluation of the user's typo-rate (or lack of decisiveness). Feature extraction is applied using the *JStylo* authorship attribution framework [76], discussed in Ch. 3. Definition and implementation of all the features the *AA* feature set consists of is available in JStylo, making our evaluations easily reproducible.

Two important processing procedures are applied in the feature extraction phase. First, every word-based feature (e.g. the function words class, or different word-grams) is applied a tailor-made preprocessing tool developed for this unique dataset, that applies the relevant special characters on the text. For instance, the character sequence `ch`$\beta\beta$`Cch`$\beta\beta$`hicago` becomes `Chicago`, where β represents backspace. Second, since the windows are determined by time and not amount of collected data as in [57], normalization is crucial for all frequency-based features (which consist the majority of the feature set). These features are simply divided by the most relevant measurement related to the feature. For instance, character bigrams are divided by the total character count of the window.

Classification

For classification we use sequential minimal optimization (SMO) support vector machines [88] with a linear kernel and complexity parameter $C = 1$, available in Weka [43]. Support vector machines are commonly used for authorship attribution and are known to achieve high performance

Table 5.2: The *AA* feature set. Inspired by the *Writeprints* [1] feature set, includes features across different levels of the text.

Group	**Features**
Lexical	Avg. word-length
	Characters
	Most common character bigrams
	Most common character trigrams
	Percentage of letters
	Percentage of uppercase letters
	Percentage of digits
	Digits
	2-digit numbers
	3-digit numbers
	Word length distribution
Syntactic	Function words
	Part-of-speech (POS) tags
	Most common POS bigrams
	Most common POS trigrams
Content	Words
	Word bigrams
	Word trigrams

and accuracy. As mentioned earlier, these are closed-world classifiers, i.e. classify each window to one of the known candidate users (with the legitimate user as the true class). No acceptance thresholds are integrated in the classification process.

Finally, the data is analyzed with the stylometry sensors using a varying threshold for minimum characters-per-window to consider, spanning from 100 to 1000 with steps of 100. For every threshold set, all windows with less than that amount of characters were thrown away, and for those windows the sensors output no decision. The different thresholds allow assessing the tradeoff in the sensor's performance in terms of accuracy and availability: as the threshold increases, the window is richer with data and will potentially be classified with higher accuracy, but the portion of total windows that pass the threshold decreases, making the sensor less available. Note that even the largest threshold (1000 *characters*) is considerably smaller than recommended for stylometry analysis – a minimum of 500 *words*. After filtering, only configurations with training data available for *all* users are kept, which expectedly yielded removal of sensors configured to small windows with high minimum number of characters thresholds.

After removal according to the rule above, 37 stylometry sensors are kept that span over a variety of time-wise window sizes and minimum character-wise window sizes. For the rest of the chapter, the stylometry sensors are denoted as $S_{n,m}$, where n denotes the time-wise window size in seconds and m denotes the minimum characters-per-window configuration.

5.4 Evaluation and Results

The generated user data streams, divided into 5 equally sized folds, are intended to be evaluated in a multi-modal decision fusion active authentication system. Such a system requires knowledge of the expected FAR/FRR rates of its different sensors, in order to make a cumulative weighted decision. Therefore the intended evaluation is based on 5-fold cross validation, where in each of the 5 validations, 3 folds are used for training, 1 fold is used for characterization of the sensors expected FAR/FRR, and the last fold is used for testing. Thus each of the 5 validations outputs a decision for each test instance (from the last fold) and a global FAR/FRR characterization of the sensor in that validation. Eventually, the results of all 5 validations are averaged to determine the performance of the system. The configuration of the validations is cyclic, such that in the first folds 1, 2 and 3 are used for training, 4 for characterization and 5 for testing; in the second, 2, 3 and 4 are used for training, 5 for characterization and 1 for testing, and so on.

The evaluation technique described above is applied in this section in order to measure how the stylometric sensors are expected to perform in a multi-modal system. Since the false accept rate (FAR) and false reject rate (FRR) produced in the *characterization* phase of the main experiments provide an evaluation of the reliability of the decisions made in the test phase, we use them to evaluate the standalone performance of the stylometric sensors. Averaged FAR and FRR results are shown in Fig. 5.1. Fig. 5.2 illustrates the averaged percentage of remaining windows, after removing all those below the minimum characters threshold.

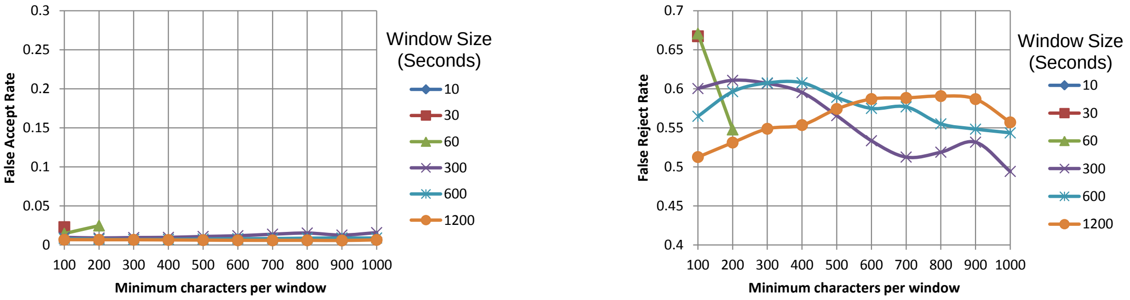


Figure 5.1: Averaged false accept and false reject rates (FAR/FRR) for all characterization phases using the stylometric sensors with varying time-wise window sizes and varying threshold for minimum number of characters per window.

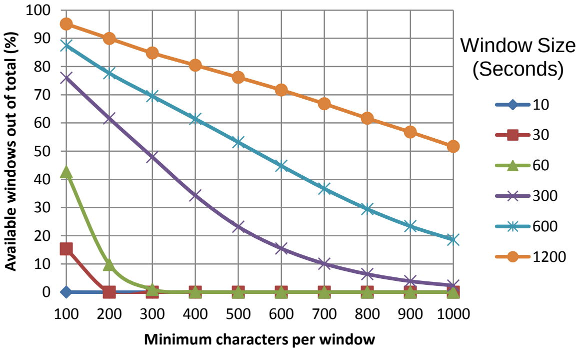


Figure 5.2: Percentage of remaining windows out of the total windows after filtering by the minimum characters-per-window threshold.

The high FRR and low FAR suggest that the majority of the sensors are rather strict, i.e. they almost never falsely identify an intruder as legitimate, but in the price of a high false-alarm rate. The FRR results indicate that as the window size (in seconds) increases, the less the minimum characters-per-window threshold affects performance. Same trend is seen with the FAR results: the large windows (300, 600 and 1,200) show insignificant differences across varying minimum characters thresholds.

The availability of decisions as a function of the minimum characters-per-window thresholds completes the image of how the stylometry sensors perform. For instance, $S_{1200,100}$, triggered every 60 seconds (the step configuration of the 1200-second-windows sensors), will produce a decision 95% of the time, with accuracy of approx. 0.5/0 FRR/FAR.

5.5 Conclusions

The initial stylometric evaluation of the active authentication dataset in [57] provides a proof of concept for the effectiveness of stylometric biometrics in an active authentication system; however, the shortcomings of this preliminary evaluation are put to the test with settings simulating a more realistic active authentication environment, with many users and high frequency decision making constraints. Under such settings, the effectiveness of stylometric sensors deteriorates drastically, down to 0.5 false rejection and 0 false acceptance rates. Nevertheless, these results are shown useful in a mixture-of-experts approach that fuses multi-modal sensors applied in [36], where adding stylometric sensors to the fusion reduced FAR and FRR from ≈ 0.008 to ≈ 0.002.

The results attained in the real-time evaluation, produced by using a closed-world approach, indicate that perhaps using a verification-based approach that allows tuning the allowable FAR and FRR can benefit the stylometric sensors in the context of active authentication. Utilizing confidence measurements has been applied in one aspect here: demanding user stream windows to contain a minimum amount of data, expressed in minimum-number-of-characters thresholds. This utilization is shown effective in canceling out sensors configured to low time-wise lengths, deeming only windows of 5 minutes and above to have some level of usability.

The approach of preferring accuracy over availability, a "better safe than sorry" approach, is the motivation of the algorithm presented next in Ch. 6 – the *Classify-Verify* algorithm. Evaluation of the active authentication dataset with the *Classify-Verify* algorithm (Sec. 6.3.5) is shown to increase the confidence and overall performance of the classifiers used, by introducing a binary verification step to determine whether to accept the underlying classifier's decision or not.

6. From Closed to Open-World Stylometry: The *Classify-Verify* Algorithm

*** This work was completed with support from Rebekah Overdorf and Sadia Afroz.* [103]

The web is full of anonymous communication with high value for digital forensics, that was never meant to be analyzed for authorship attribution. The potential for authors of anonymous documents to be "de-anonymized" raises privacy concerns. These concerns become greater if the de-anonymizing party need not to come up with an exact list of suspects to reliably perform attribution. This is the scenario we explore in this chapter.

The effectiveness of stylometry has considerable implications for anonymous and pseudonymous speech. Recent work has exposed limits on stylometry through active circumvention [16, 76]. Stylometry has thus far focused mostly on limited, closed-world models. In the classic stylometry problem, there are relatively few authors (usually fewer than 20, nearly always fewer than 100), the set of possible authors is known, every author has a large training set and all the text is from the same genre. However, problems faced in the real-world often do not conform to these restrictions.

Controversial, pseudonymous documents that are published on the Internet often have an unbounded suspect list. Even if the list is known with certainty, training data may not exist for all suspects. Nonetheless, classic stylometry requires a fixed list and training data for each suspect, and an author is always selected from this list. This is problematic both for forensics analysts, as they have no way of knowing when widening their suspect pool is required, and for Internet activists as well, who may appear in these suspect lists and be falsely accused of writing certain documents.

In this chapter we explore a mixed closed-world and open-world authorship attribution problem where we have a known set of suspect authors, but with some probability (known or unknown) that the author we seek is not in that set. For simplicity of notation, we denote documents whose author is in the known suspect set as *in-set* documents, and those whose author is missing – *not-in-set* documents. The key contributions presented in this chapter are:

The *Classify-Verify* algorithm. This novel method augments authorship classification with a verification step, and obtains similar accuracy on open-world problems as traditional classifiers in closed-world problems. Even in the closed-world case, *Classify-Verify* can improve results by replacing wrongly identified authors with "unknown." *Classify-Verify* can be tuned to different levels of rigidity, to achieve the desired false positive and false negative error rates. It can also be automatically tuned to maximize the desired evaluation measurement, taking into account the expected proportion of documents by authors in the suspect list versus those who are absent – *in-set* and *not-in-set* rates. Alternatively, it can be automatically set with robust thresholds when that expected proportion is unknown.

Classify-Verify is evaluated on several datasets in simulated scenarios with *in-set* values ranging from 10% (of the test authors having training data) to 100% (complete closed-world settings). Results are compared to two baselines established using 1) only *Classify*: closed-world classifiers applied in open-world settings; and 2) only *Verify*: open-world binary verifiers, configured as 1-of-n classifiers. *Classify-Verify* is shown to outperform both baselines.

Adversarial settings. Previous work has shown that traditional classification performs near random chance when faced with writers who change their style. *Classify-Verify* filters out most of the attacks in the Extended-Brennan-Greenstadt Adversarial corpus [16], an improvement over previous work which requires training on adversarial data for attack detection [2].

Large datasets. One of the real-world targets of the *Classify-Verify* method, being a mixture of a closed- and open-world approach, is to be used in online domains, where problems may include a large number of authors. We perform a set of experiments using a subset of the Spinn3r blog dataset [19] which includes 911 candidate authors, and show that *Classify-Verify* is successful in these many authors, online domain settings.

Active authentication settings. Behavioral biometric systems [3] aim to actively authenticate users for access control purposes. Based on usage of common input devices such as keyboard and mouse, these systems can block intruders that managed to bypass common gateway security measures, like passwords. The Active Linguistic Authentication Dataset [57] provides user input data in a simulated work environment, targeted for evaluation of such systems. As shown in Ch. 5, real-time stylometric evaluation of sliding user input windows has some effectiveness [102], yet this problem can be naturally formulated as a *Classify-Verify* problem, and in this chapter it is evaluated as such. *Classify-Verify* is shown to perform well even in such settings, facing dynamic, “noisy” and inconsistent user input.

The Sigma Verification method. This method is based on the *distractorless* verification method [82] (discussed in Sec. 2.3.2), which measures the distance between an author and a document. Sigma Verification incorporates pairwise distances within the author's documents and the standard deviations of the author's features, and although does not outperform the distractorless method always, it is yet shown as a better alternative suitable for datasets with certain characteristics.

This chapter is structured as follows: Sec. 6.1 recapitulates the closed-world and open-world authorship attribution problems, and defines the *Classify-Verify* problem. Sec. 6.2 details the evaluation methodology applied, including experimental setup, datasets, feature sets, closed-world and open-world settings, and finally the *Classify-Verify* algorithm and configuration. Sec. 6.3 presents all evaluation results for the various experimental configurations applied with the *Classify-Verify* method. Sec. 6.4 discusses conclusions and directions for future work.

6.1 Problem Statement

The *Classify-Verify* problem is a mixture of the *authorship attribution* and *authorship verification* problems defined in Sec. 2.2.1. To recap, the *authorship attribution* problem is: given a document D of unknown authorship and documents by a set of known authors $\mathcal{A} = \{A_1, ..., A_n\}$, determine the author $A_i \in \mathcal{A}$ of D. This problem assumes D's author is in $\mathcal{A}$. The *authorship verification* problem is: given a document D and an author A, determine whether D is written by A.

Finally, the *Classify-Verify* problem is the following: given a document D of unknown authorship and documents by a set of known authors $\mathcal{A}$, determine the author $A_i \in \mathcal{A}$ of D, *or* that D's author is not in $\mathcal{A}$. This problem is similar to the attribution problem, with the addition of the class "unknown", denoted as $\perp$. This problem may include an additional parameter $p = Pr[A_D \in \mathcal{A}]$, the probability that D's author is in $\mathcal{A}$. As mentioned above, we address documents whose authors are in the set of suspects $\mathcal{A}$ as *in-set* documents, and those with absent authors – *not-in-set* documents. The *in-set* probability is denoted in short as p (hence the *not-in-set* probability is $1 - p$.)

6.1.1 Hypothetical Scenario

The *Classify-Verify* problem is illustrated in the following hypothetical scenario. Consider Bob's workplace which he shares with $n - 1$ other employees, under the management of Alice. Bob leaves his desk to get a cup of coffee, and incautiously forgets to lock his computer. When he returns, he discovers that a vicious (and sufficiently long) email has been sent in his name to Alice! He quickly goes to Alice in order to explain, and Alice decides to check the authorship of the email to assert Bob's innocence (or refute it). Luckily Alice has access to the company's email database, so she can model the writing style of her n employees. Unluckily, the security guard at the door tends to doze off every once in a while, resulting with unauthorized people wondering off in the company's halls, such that the expected portion of authorized people in the office at any given time is p.

A closed-world system would only be able to consider the n employees and identify one of them as the culprit. This would be problematic if the email was written by one of the unauthorized entrants. A *Classify-Verify* approach would be able to consider this possibility.

6.1.2 Problems with Closed-World Models

Applying closed-world stylometry in open-world settings suffers from a fundamental flaw: a closed-world classifier will *always* output some author in the suspect set. If it outputs an author, it merely means the document in question is written in a style more similar to that author's style than the others, and the probability estimates of the classifier reflect only who is the least-worst choice. Meanwhile, the absence of the document's author from the set of suspects remains unknown. If we relax the precision of our results to k-accuracy [80], i.e. target to narrow down our set of suspects to k rather than just one, the problem will not be solved – all k options will still be wrong.

This problem becomes prominent especially in online domains, where the number of potential suspects can be virtually unbounded, and we may have only a handful of candidate authors in hand. Failing to address the limitations of closed-world models may result in falsely attributed authors with consequences for both the forensic analyst and the innocent Internet user.

The *Classify-Verify* method applies an *abstaining classification* approach, according to which classification decisions are rejected when the classifier's confidence in the decision is low [26, 46, 87], thus reducing the misclassification rate. With the *Classify-Verify* method, closed-world classification is initially applied, followed by an open-world author-specific verifier to determine whether to accept or reject the classifier's decision. Thus, the closed-world assumption is broken, and its limitations are removed by allowing to reject possibly wrong attributions.

6.2 Methodology

6.2.1 Real-Time Evaluation Methodology

Initial evaluation of the *Classify-Verify* method in different *in-set*/*not-in-set* scenarios is applied in theoretical settings [103], as detailed next. For a given dataset of n authors, each document is evaluated twice: once as *in-set*, and once as *not-in-set*. n variants of the chosen closed-world classifiers are evaluated on n variants of the dataset, where classifier C_i is evaluated on dataset i which contains training data for all authors but A_i (i.e. $n-1$ authors). Then, any document by A_i is classified as *in-set* by one of the $n-1$ classifiers trained on A_i ($C_{i+1 \bmod n}$ is arbitrarily chosen), and as *not-in-set* by C_i. Then, to evaluate performance for some given value of p, the *in-set* proportion, a classification confusion matrix is generated to contain a weighted average of p-weighted *in-set* accuracy and $(1-p)$-weighted *not-in-set* accuracy. Baseline performance of the closed-world classifiers in open-world settings for any value of p is evaluated as simply p times the accuracy in pure closed-world settings. Most of the evaluation is focused on the scenario where $p = 0.5$ (documents are equally likely to be *in-set* and *not-in-set*).

This configuration provides a solid theoretical performance measurement, however insufficient in order to predict how well *Classify-Verify* works in real-world settings, where authors of test documents are truly missing from the set of candidates. To provide a better evaluation, an extensive set of experiments is applied for a range of *in-set* scenarios with $p = 0.1, ..., 1.0$ with steps of 0.1 (i.e. 10% *in-set* authors, 20% etc. up to 100% – pure closed-world settings). For each value of p for a dataset with n authors, a set of 10 experiments is applied, where for each experiment a set of $p \times n$ authors are randomly chosen as *in-set* authors, and training data only for those authors is maintained, such that any document written by an author *not* in the chosen set is *truly* a *not-in-set* document. The final results reported for each p are an average over the corresponding 10 experiments. As opposed to the initial theoretical evaluation, applying *Classify-Verify* in these real-world settings predicts how well it performs when faced with authors actually missing.

Finally, each experiment is evaluated with n-fold cross-validation. We choose F1-score as evaluation criterion:

$$F1\text{-}score = 2 \times \frac{precision \times recall}{precision + recall}$$
$$precision = \frac{tp}{tp + fp}, recall = \frac{tp}{tp + fn}$$

F1-score is a natural choice as it is a weighted measurement of precision and recall, between which the tradeoff is expressed based on the rigidity of the underlying verifier - the likelihood to reject classifier decisions. The more likely the verifier is to reject, the lower recall and the higher precision would be; however, when the verifier is lenient and more likely to accept classifier decisions, recall is expected to increase due to non-rejected true positives, and precision – decrease, as a result of a higher false acceptance rate.

6.2.2 *Flexible* vs. *Strict* Evaluation

We apply two types of evaluation: *flexible* and *strict*. With *flexible* evaluation, each thwarted misclassification, whether originated by the document being *not-in-set* or simply in-set classifier mistake, is deemed a "true" classification - i.e. truly assigned to $\perp$ (unknown). This method is common in evaluating abstaining classifiers [46], which favors precision over recall: we rather know less, but have high confidence in the results we get. That goal is achieved by essentially setting the ground-truth of any instance the underlying classifier has misclassified as "unknown". Thus, if the verifier rejects the classifier's wrong decision – i.e. assigns it to the class "unknown" – it is counted as a true positive.

In addition to *flexible* evaluation, we apply the *strict* approach, according to which a thwarted misclassification originated in an *in-set* document is deemed "false": we penalize the classifier if it had the potential to find the true author (as it is *in-set*), but did not, regardless of whether that misclassification was thwarted by the underlying verifier. *Strict* evaluation provides a fuller picture of the performance of the *Classify-Verify* method in different scenarios, showing the distribution of misclassifications between those originated in the classifier making a mistake among *in-set* authors, and documents that are truly *not-in-set*. In all figures presented in Sec. 6.3 that illustrate both *flexible* and *strict* F1-scores, the difference between the two lines represents the misclassified *in-set* documents that were thwarted by the underlying verifier and assigned to "unknown" instead.

6.2.3 Datasets

We utilize several datasets for evaluating the *Classify-Verify* method. Most of the evaluations in this chapter focus on two of these datasets: the Extended-Brennan-Greenstadt (EBG) Adversarial corpus [16] and the ICWSM 2009 Spinn3r Blog dataset [19].

The EBG corpus, denoted *EBG*, consists of a set of 45 authors with 500-word documents for a total of at least 6,500 words per author. The authors were instructed to provide the writings from sources like formal essays, college applications etc., therefore it contains mostly "clean" and formal writing style. It is evaluated in Sec. 6.3.1 and Sec. 6.3.2. *EBG* also contains adversarial documents where the authors were instructed to attempt hiding their writing style in order to circumvent classification, which is evaluated in Sec. 6.3.3.

The Spinn3r blog corpus contains 44 million blog posts, and is used to create two datasets: a smaller version that consists of blogs by 50 authors, denoted $BLOG_S$ (where the s stands for "small"), and a large version that consists of blogs by 911 authors, denoted $BLOG_L$. $BLOG_S$ is set to contain a similar number of authors as *EBG*, used as a control corpus to avoid overfitting configurations on *EBG*, and is evaluated alongside *EBG* in Sec. 6.3.1 and Sec. 6.3.2. $BLOG_L$, evaluated in Sec. 6.3.4, is used to examine the *Classify-Verify* method in online domain settings with a high number of authors.

Finally, we utilize the Active Linguistic Authentication Dataset [57], denoted *AAUTH*, which contains user data collected in a simulated work environment, as discussed and evaluated in Sec. 5. 80 temporary workers were assigned research and writing tasks for a period of one week, during which a complete keyboard, mouse, application and browsing behavior was monitored and recorded. We follow the real-time dataset setup applied in Sec. 5 and use a subset data by 67 users that passed a threshold of 16.67 hours of total minimum activity. This dataset is evaluated in Sec. 6.3.5.

6.2.4 Feature Set

The feature set used for stylometric evaluation has great effect on the attribution accuracy. As the field of stylometry provides a vast range of features that can be adopted for document quantification [55], we evaluate two feature sets: the *Writeprints* and the $\langle 500, 2\rangle$-*chars* feature sets. As in the previous chapters, feature extraction is applied using JStylo [76].

The Writeprints algorithm [1] is a PCA variant developed for stylometric similarity detection problems, and has been proven very effective with over 90% accuracy for a set of 50 candidate authors. This algorithm utilizes a vast set of features across different levels of the text, including lexical, syntactic, and content related features. We adopt the feature set used by this algorithm, denoted *Writeprints*, which has been shown effective for the *EBG* dataset [16].

For simplicity, in addition to *Writeprints*, we utilize a feature set that consists only of one type of feature: the k most common word n-grams or character n-grams, with k from 50 to 1000 with steps of 50, and n from 1 to 5 with steps of 1. The most-common feature selection heuristic is commonly used in stylometry [1, 64, 82] to improve performance and avoid over-fitting, as are the chosen ranges of k and n. F1-score closed-world results for evaluation of *EBG* using 10-fold cross-validation with SVM and character/word n-grams are illustrated in Fig. 6.1.

Of word and character n-grams, characters perform better with the best F1-score results attained with character bigrams at ≈ 0.93 (for $k = 400$ and above), compared to the best score of 0.879 for words, attained using $n = 1$ and $k = 1000$. Both feature sets outperform the *EBG* evaluation with *Writeprints* at F1-score of 0.832 (as obtained originally in [16]). Finally, we choose the 500 most

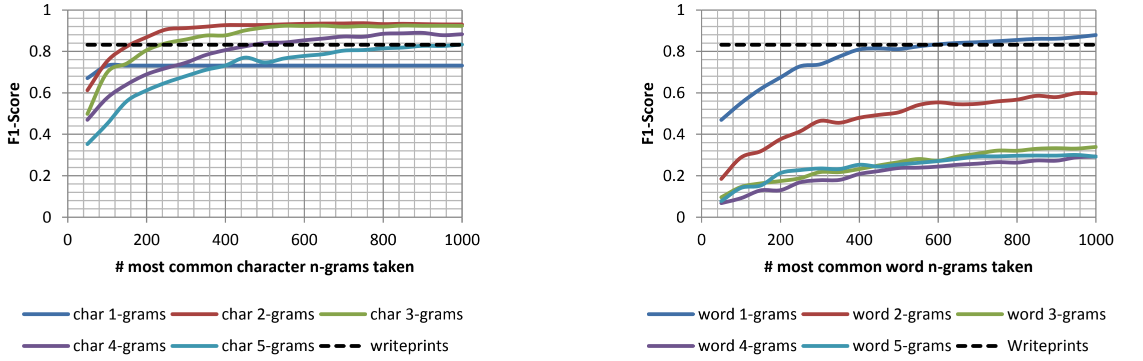


Figure 6.1: F1-scores for evaluation of the EBG corpus using different character (left) and word (right) n-grams with varying limits of the feature set size.

common character bigrams as our feature set (at F1-score of 0.928), denoted $\langle 500, 2\rangle$-*chars*, used throughout all of our experiments. It is chosen for its simplicity, performance and effectiveness.

For control, we evaluate the effectiveness of using $\langle 500, 2\rangle$-*chars* compared to using the *Writeprints* feature set on $BLOG_S$, via 10-fold cross validation with SVM. Although both results are lower than those obtained for *EBG*, $\langle 500, 2\rangle$-*chars* outperformed *Writeprints* with F1-score of 0.64 versus 0.509, respectively. Both $\langle 500, 2\rangle$-*chars* and *Writeprints* are evaluated next in Sec. 6.2.5 and Sec. 6.2.6 in different *in-set*/*not-in-set* settings, where $\langle 500, 2\rangle$-*chars* is shown to outperform *Writeprints* in almost every configuration, therefore used for all experiments in this chapter.

6.2.5 Classify: Closed-World Setup

Closed-world classifiers applied to stylometry problems are used to attribute authorship of an unknown test document to an author from a closed set of candidates. As discussed in Sec. 6.1, this type of configuration is not suited for open-world settings, since the classifier always provides the best (or least-worst) decision assuming the true author is in the candidate set. We utilize closed-world classifiers to provide baseline performance measurements on our evaluated datasets, to illustrate how the *Classify-Verify* method can outperform them in open-world settings, and for the closed-world classification step of the *Classify-Verify* method discussed later in Sec. 6.2.7.

We use a linear kernel sequential minimal optimization support vector machine (SMO SVM) classifier [88], implemented in Weka [43] with complexity parameter $C = 1$. SVMs are proven effective for authorship attribution, including the datasets evaluated in Sec. 6.3 [16, 55, 102, 103].

Finally, to reinforce our feature set selection discussed in the previous section, we evaluate

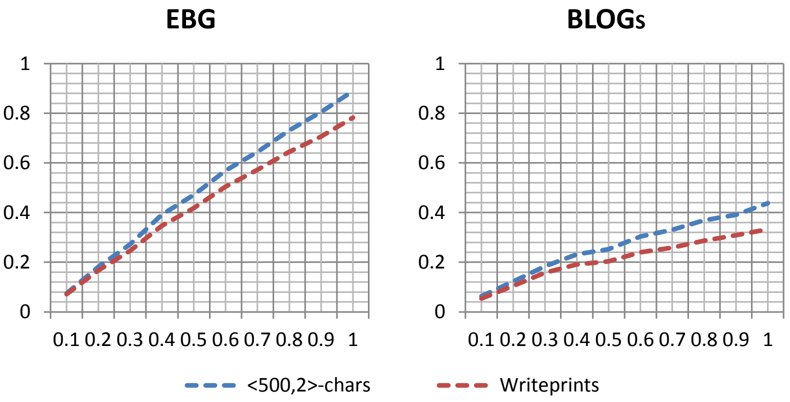


Figure 6.2: F1-scores for classification using SVM with $\langle 500, 2\rangle$-*chars* and *Writeprints*.

Writeprints and $\langle 500, 2\rangle$-*chars* on *EBG* and $BLOG_S$ in all $p = 0.1, ..., 1.0$ scenarios (probabilities of *in-set*) using a purely closed-world approach. The evaluation confirms that $\langle 500, 2\rangle$-*chars* indeed outperforms *Writeprints* for all values of p on both datasets. However small the difference is, especially for lower values of p, simplicity and feature extraction performance rule in favor of $\langle 500, 2\rangle$-*chars*. F1-scores as a function of p are illustrated in Fig. 6.2.

6.2.6 Verify: Open-World Setup

Open-world verifiers applied to stylometry problems are used to determine whether a given test document is written by a single candidate author or not. This formulation is much harder than closed-world scenarios, as the verifier must determine what is deemed “close-enough”, manifested in an acceptance threshold. Naïve approaches such as reducing the problem to a one-versus-all closed-world formulation using a “distractor set” of documents not written by the candidate author are insufficient in many cases, as discussed thoroughly in Sec. 2.

In the next sections we discuss and evaluate several verification methods. The first family of methods is *classifier-induced* verifiers, which require an underlying (closed-world) classifier and utilize its class probabilities output for verification.

The second family of methods is *standalone* verifiers, which rely on a model built using author training data, independent of other authors or classifiers. We evaluate two verification methods: the first is the *distractorless* verification method, denoted V [82], which was thoroughly reviewed in Sec. 2.3.2. It is used as a baseline as it is a straight forward verification method, proven robust across different domains, and does not use a distractor set (model of "*not-A*"). We then present the *Sigma Verification* method, which applies variations to V by adding per-feature standard deviations normalization (denoted V_σ) and adding per-author threshold normalization (denoted V^a; the method with both adjustments combined is denoted V_σ^a). We evaluate and compare V with its new variants.

Classifier-Induced Verifiers

One promising aspect of the closed-world model that can be used in open-world scenarios is the confidence in the solution given by distance-based classifiers. A higher confidence in an author may, naturally, indicate that the author is in the suspect set while a lower confidence may indicate that s/he is not and that this problem is, in fact, an open-world situation. Following classification, verification can be formulated simply by setting an acceptance threshold t, measure the confidence of the classifier in its classification, and accept the classification if and only if it is above t.

Next we discuss several verification schemes, based on classification probabilities outputted by closed-world classifiers. For each test document D with suspect authors $\mathcal{A} = \{A_1, ..., A_n\}$, a classifier produces a list of probabilities P_{A_i} which is, according to the classifier, the probability D is written by A_i ($\sum_{i=1}^{n} P_{A_i} = 1$). We denote the probabilities $P_1, ..., P_n$ as the reverse order statistic of P_{A_i}, i.e. P_1 is the highest probability given to an author (i.e. the chosen one), P_2 the second highest and so on.

These methods are obviously limited to classify-verify scenarios, as verification is dependent on classification results (therefore evaluated in the *Classify-Verify* evaluation section, Sec. 6.3). We use SVM classifiers with the $\langle 500, 2\rangle$-*chars* feature set, fitting logistic regression models to the SVM outputs for proper probability estimates. The classifier-induced verification methods evaluated in Sec. 6.3 are:

P_1 The classifier's probability output for the chosen author. If it is above some threshold, we deduce the classifier is confident enough of its top choice, relative to all others, therefore accept. the P_1 statistic is the one used for the 2-step classification approach applied for native language identification in Sec. 4.4.5.

P_1-P_2-*Diff* The difference between the class probability of the chosen author and the next best choice, i.e. $P_1 - P_2$. If the chosen author probability is far enough from the rest, it is assumed to be true.

Gap-Conf Measurement of the gap-confidence [86] statistic. This method is identical to P_1-P_2-*Diff*, with the difference that each probability i for author A_i is generated by an A_i-vs-all classifier:

instead of training a single SVM, we train n one-versus-all SVMs, one per author. For a given document D, each classifier i in turn produces 2 probabilities: the probability P_{A_i} that D is written by A_i and the probability it is not. *Gap-Conf* is the difference $P_{A_i} - P_{A_j}$ between the top two candidates A_i and A_j. The hypothesis is similar to P_1-P_2-*Diff*: the probability of the true author should be much higher than that of the second-best choice.

Standalone Verification

Standalone verifiers are classifier-independent and are defined solely by a model built for the candidate author and an acceptance threshold. If the distance between the author model and the test document model is below the threshold, the document is deemed written by the author. We utilize 3 standalone methods – *distractorless* verification [82], V, and two novel variants thereof, V_σ and V^a (combined together to V_σ^a), referred to as *Sigma verification*. These methods are configured as follows:

V ***Distractorless* Verification.** Described thoroughly in Sec. 2.3.2, this method uses average relative frequency vectors to model the author and test document, combined with cosine distance (dot product) and an acceptance threshold. This method is proven robust across domains and languages.

V_σ **Per-Feature SD Normalization.** A variant of V that uses the variance of the author's writing style. If an author has a rather unvaried style, we aim for a tighter bound for verification, whereas for a more varied style we can loosen the model to be more accepting. For that we use the standard deviation of an author, denoted SD, on a per-feature basis. We first calculate the SD of all features for each author. When computing distance between an author and a document, we divide each feature-distance by its SD, so if the SD is smaller, A and D move closer together, otherwise they move farther apart. This idea is applied in [9] for authentication through typing biometrics.

V^a **Per-Author Threshold Normalization.** Another variant of V that adjusts the verification threshold t on a per-author basis, based on the average pairwise distance between all of the author's documents, denoted δ_A. V does not take this into account and instead uses a fixed threshold. Using δ_A to determine the threshold is, intuitively, an improvement because it accounts for how spread out the documents of an author are. This allows the model to relax if the author has a more varied style. Similarly to V, this "varying" threshold is still applied by setting a single threshold t across all authors; however for V^a every author-document distance measurement δ is adjusted by *subtracting* δ_A prior to being compared with t, thus allowing per-author thresholds but still requires the user to set only *one* fixed threshold value.

Tab. 6.1 details the differences in distance calculations and threshold test among V, V_σ and V^a. We denote $\delta_{D,A}$ as the overall distance measured by some distance metric δ between the feature vector of document D and the centroid vector of author A across all of A's documents, denoted $C(A)$. In addition we denote the respective feature level representation of δ as follows: $\delta_{D,A} = \Delta(D_i, C(A)_i)_{i=1}^n$, where n is the number of features (dimension of D and $C(A)$). Finally, we define $\sigma(A)$ as the standard deviation vector of author A's features, and δ_A as the pairwise distance between all of A's documents.

Table 6.1: Differences in distance calculation and t-threshold test for V, V_σ and V^a.

Test Distance	$\delta < t$	$\delta - \delta_A < t$
$\delta_{D,A} = \Delta(D_i, C(A)_i)_{i=1}^n$	V	V^a
$\delta_{D,A}^\sigma = \Delta(\frac{D_i}{\sigma(A)_i}, \frac{C(A)_i}{\sigma(A)_i})_{i=1}^n$	V_σ	V_σ^a

Note that using V^a may derive nonintuitive thresholds (e.g. negative thresholds when using cosine distance, which normally produces values in $[0, 1]$). However this is only to adjust to the distance shift from $\delta_{D,A}$ (used in V) by δ_A to $\delta_{D,A} - \delta_A$ used by V^a, i.e. it is a byproduct of the per-author threshold normalization.

Standalone Verification: Evaluation

We evaluate the methods above on *EBG*, and $BLOG_S$ as control. The evaluation is done by examining false positive rates and their corresponding false negative error rates. *EBG* is evaluated only on the non-adversarial documents, and $BLOG_S$ is evaluated in its entirety. The evaluation is done using 10-fold cross-validation. In each fold, every test document is tested against every one of the authors models, including its own. ROC curves for evaluation of V, V_σ and V_σ^a on *EBG* and $BLOG_S$ are illustrated in Fig. 6.3.

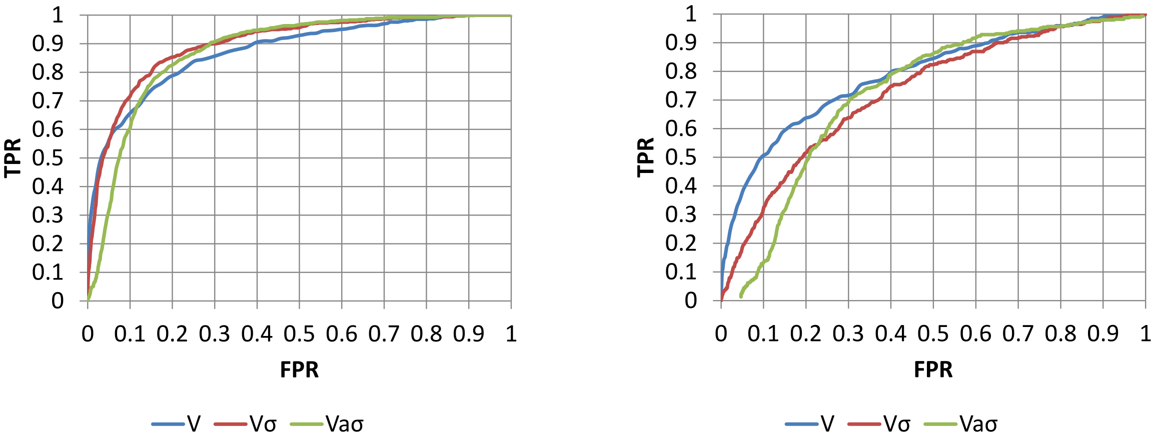


Figure 6.3: ROC curves for V, V_σ and V_σ^a evaluation on EBG (left) and $BLOG_S$ (right).

The results are mixed: on EBG, V_σ and V_σ^a significantly outperform V (from FPR of 0.05 and 0.114, respectively). However on $BLOG_S$, V significantly outperforms both V_σ and V_σ^a. These differences may be due to the discrepancy between the literary styles of the two datasets, where accounting for how “wide” an author’s style is with V_σ and V_σ^a seems more fitting to formal texts (EBG), and a simpler method as V is better suited for less structured and formal texts (blogs). The results suggest that there is no one method preferable over the other, and selecting a verifier for a problem should rely on empirical testing over a stylistically similar training data.

As for the effect of adding the per-author threshold adjustments, for both corpora V_σ outperforms V_σ^a on low FPR until they intersect (at $FP = 0.27$ and $FP = 0.22$ for EBG and $BLOG_S$, respectively), at which point V_σ^a begins to outperform V_σ. These properties allow various verification approaches to be used per need, dependent on FPR/FNR constraints the problem in hand may impose.

Finally, similarly to closed-world classification, in order to validate the feature selection for the underlying verifiers in the *Classify-Verify* method, we evaluate the performance of V for $p = 0.1, ..., 1.0$ using $\langle 500, 2\rangle$-*chars* and *Writeprints*. The evaluation is applied in a closed-world fashion, where distances are measured between the test document and each author model, and the author with the shortest distance is the chosen class. Evaluation results reveal that $\langle 500, 2\rangle$-*chars* is well suited for verification as well as classification, where it is almost identical to *Writeprints* on EBG, and outperforms it on $BLOG_S$. F1-scores are illustrated in Fig. 6.4.

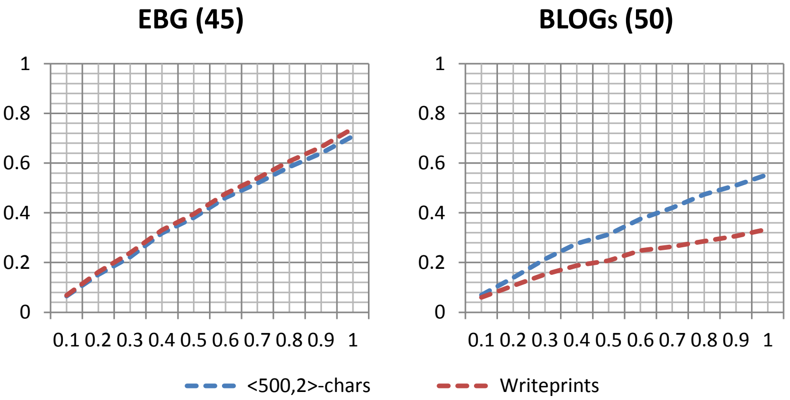


Figure 6.4: F1-scores for classification using standalone verification with $\langle 500, 2\rangle$-*chars* and *Writeprints*.

6.2.7 The *Classify-Verify* Algorithm

The *Classify-Verify* algorithm combines an underlying closed-world classifier trained on a set of candidate authors $\mathcal{A} = \{A_1, ..., A_n\}$, along with a set of verifiers for each author in $\mathcal{A}$, thus expanding closed-world authorship problems to open-world, by essentially adding another class: *"unknown"*.

First, the document in question D is classified, and some author A_i is chosen. Then, the test document is fed into A_i's corresponding verifier (for standalone verifiers; classifier-induced verifiers are global and based on the classifier alone.) If the verifier accepts, based on a verification threshold t, it outputs A_i as the chosen author. Otherwise, it outputs $\perp$, signifying "unknown". These two steps are aimed to complement each other: the classifier provides the best choice possible from a known set of candidates, and the verifier has to deal with only that best choice (rather than verify each author in the set). The classifier contributes its better suitability for choosing one of many, and the verifier provides its open-world ability to reject decisions with low confidence. *Classify-Verify* is essentially a classifier over the suspect set $\mathcal{A} \cup \{\perp\}$.

The verification acceptance threshold t selection determines the rigidity of the configuration, which affects the precision and recall attainable by the algorithm. The basic approach taken in most experiments in Sec. 6.3 utilize an "oracle" threshold: a range of manually set thresholds are tested, and the results for the threshold that yields the best results are presented (as if we predicted the best threshold in advance, hence "oracle"). However, we propose an empirical approach for automatically setting the acceptance thresholds, given various knowledge of p, the probability of any given test

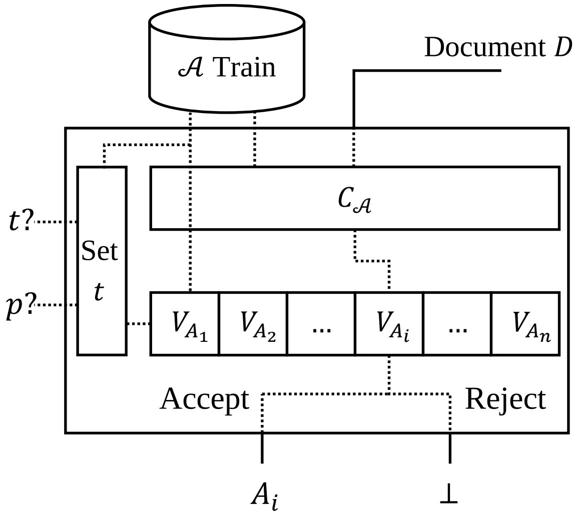


Figure 6.5: The flow of the *Classify-Verify* method on a test document D and a suspect set $\mathcal{A}$, with optional inputs of a manual threshold t and a known *in-set* portion p.

document being *in-set*. The automatic threshold selection techniques are as follows:

p-Induced. For any given value of p in a n-fold cross-validation experimental setup, for each test fold i we apply n-but-i-fold validation on the training set using a range of thresholds. For each fold i, the threshold that yielded the highest results on the training validation is selected. The idea is that thresholds that perform well on training simulations are expected to perform similarly under test. As opposed to using oracle thresholds, p-induced thresholds are not in danger of overfiting to the test data, as they are calculated independent of it.

Robust. For the case where the value of p is unknown, we require a method to select the threshold such that it is useful for any possible value of p. The robust threshold selection technique is similar to p-induced, only instead of selecting the threshold that performs well in simulation for a particular value of p, it is calculated for all $p = 0.1, ..., 1.0$, and the threshold that yields the highest averaged results is selected for testing. The robust threshold does not guarantee the highest measurement; however, it aims to maximize the expected value of that measure, independent of p, and thus robust for any open-world settings.

Finally, the flow of the *Classify-Verify* algorithm is illustrated in Fig. 6.5, and the algorithm is described in Alg. 1.

Algorithm 1 *Classify-Verify*

```
Input: Document $D$, suspect author set $\mathcal{A} = \{A_1, ..., A_n\}$, target measurement $\mu$
  Optional: in-set portion $p$, manual threshold $t$
Output: $A_D$ if $A_D \in \mathcal{A}$, and $\perp$ otherwise
  $C_\mathcal{A} \leftarrow$ classifier trained on $\mathcal{A}$
  $\mathcal{V}_\mathcal{A} = \{V_{A_1}, ..., V_{A_n}\} \leftarrow$ verifiers trained on $\mathcal{A}$
  if $t, p$ not set then
    $t \leftarrow$ Robust threshold maximizing $\mu$ of Classify-Verify cross-validation on $\mathcal{A}$
  else if $t$ not set then
    $t \leftarrow$ $p$-induced threshold maximizing $\mu$ of Classify-Verify cross-validation on $\mathcal{A}$
  end if
  $A \leftarrow C_\mathcal{A}(D)$
  if $V_A(D, t) = True$ then
    return $A$
  else
    return $\perp$
  end if
```

6.3 Evaluation

6.3.1 Main Evaluation

In our main experiments we evaluate the *Classify-Verify* method on the *EBG* and *BLOG*$_S$ datasets, using every configuration mentioned above: SVM classifiers with standalone verifiers, SVM with classifier-indcued verifiers and only standalone verifiers. For standalone verifiers, those are used in the "classify" phase in a closed-world fashion such that the author with the shortest distance from the test instance is the chosen one.

F1-scores illustrated in Fig. 6.6 show results for the best performing standalone and classifier-induced *Classify-Verify* configurations, on both datasets. The results suggest that *Classify-Verify* is successful in thwarting misclassifications, originated *in-set* or *not-in-set*, when applied in real-world settings for any value of p. All F1-scores illustrated in Fig. 6.6 are detailed in Tab. B.1–B.2

For both datasets, of all the mix-and-match classifiers and verifiers, the best results are gained with SVM + P_1. In general, *Classify-Verify* is shown to perform better with closed-world SVM for the "classify" phase rather than closed-world formulated verifiers. The small difference between the *EBG strict* and *flexible* results with SVM suggests that the *in-set* misclassification rate is rather low. On the other hand, same results for *BLOG*$_S$ show higher SVM *in-set* misclassification rates, which grows alongside p. However, when the origin of thwarted misclassifications is set aside and we examine the bottom-line *flexible* performance, it reaches as high as baseline closed-world classifiers in pure closed world settings for *EBG*, and surpasses it for *BLOG*$_S$. Complete results using *all*

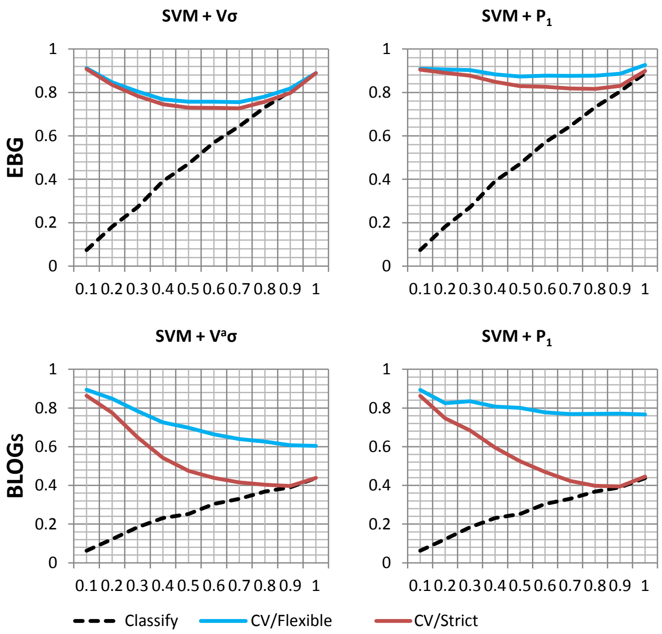


Figure 6.6: *Classify-Verify* F1-scores on *EBG* and $BLOG_S$ as a function of $p = 0.1, ..., 1.0$, with the best standalone and classifier-induced verifiers. *Classify-Verify* successfully thwarts *in-set* and *not-in-set* misclassifications; applied in open-world settings, it matches and even outperforms standard classifiers in closed-world settings. P_1 outperforms all others on both datasets.

Classify-Verify configurations with both the $\langle 500, 2\rangle$-*chars* and *Writeprints* feature sets on *EBG* and $BLOG_S$ are found in Fig. B.1–B.4.

Due to the better performance of *Classify-Verify* with SVM used in the "classify" phase, we continue only with that classifier for the rest of the experiments in this chapter, and present only the best standalone and classifier-induced results.

6.3.2 Auto-Selected Verification Thresholds

We apply the two automatic acceptance threshold techniques discussed in Sec. 6.2.7, namely p-induced and robust. F1-scores illustrated in Fig. 6.7–6.8 suggest that both automatic threshold selection techniques perform well in real-world settings for any value of p, and even similarly to using oracle thresholds (that yield the best results on the test data) used in the previous section. All F1-scores illustrated in Fig. 6.7–6.8 are detailed in Tab. B.1–B.2. Complete F1-scores across all configurations are illustrated in Fig. B.5–B.8.

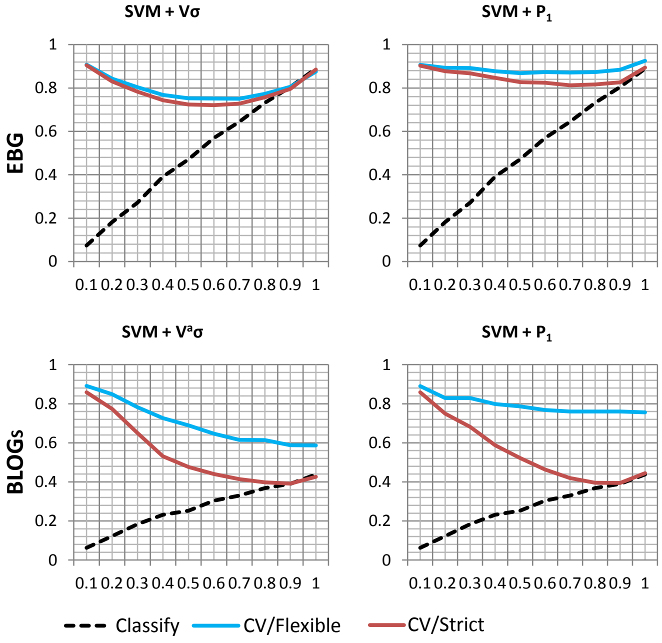


Figure 6.7: *Classify-Verify* F1-scores on *EBG* and $BLOG_S$ as a function of $p = 0.1, ..., 1.0$ using p-induced verification thresholds. Attained results are similar to those attained with "oracle" threshold in Sec. 6.3.1, and outperform closed-world classifiers in any setting.

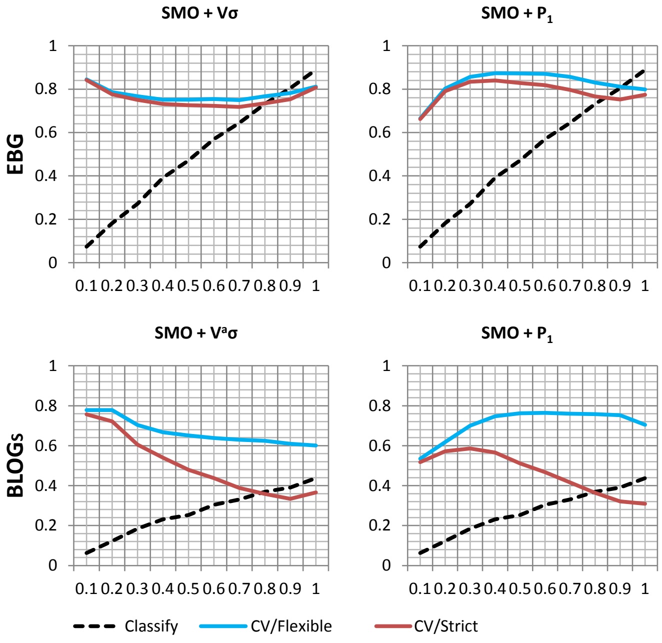


Figure 6.8: *Classify-Verify* F1-scores on *EBG* and $BLOG_S$ as a function of $p = 0.1, ..., 1.0$ using robust verification thresholds. Attained results are not as high as p-induced thresholds, however considerably high with the advantage of being ready for any p scenario.

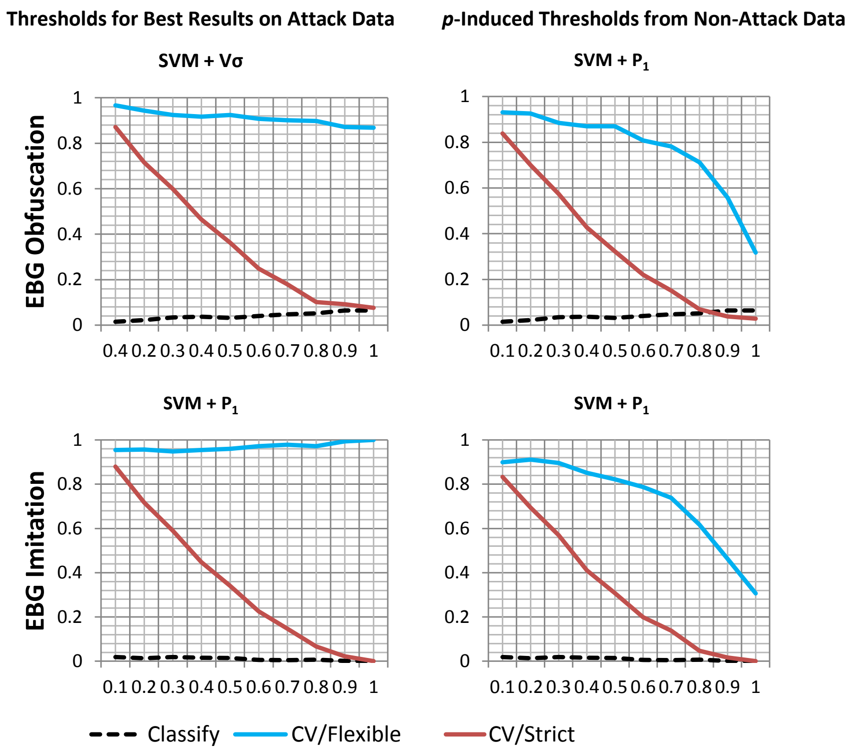


Figure 6.9: *Classify-Verify* F1-scores on *EBG* Imitation and Obfuscation attack documents, as a function of $p = 0.1, ..., 1.0$. *Classify-Verify* successfully thwarts attacks in any setting, even when configured with non-attack auto-selected p-induced thresholds.

6.3.3 Adversarial Settings

The *EBG* corpus provides unique documents where the authors are instructed to attempt circumventing stylometric techniques by changing their writing style in two fashions: obfuscation and imitation attacks. In the obfuscation attack, the authors were guided to try changing their writing style with no particular theme. In the imitation attack, the authors are guided to imitate the unique writing style of the author Cormac McCarthy (the obfuscation task was given first, to prevent subjects from being affected by the imitation task when hiding their style.) Both attacks are proven effective in circumventing stylometry accuracy down to random chance [16]. We repeat the original evaluations of *EBG* in adversarial settings, and expand them for all *Classify-Verify* scenarios with $p = 0.1, ..., 1.0$. In these scenarios we may look at the class $\perp$ as "possible attack", rather than simply "unknown".

F1-score results in Fig. 6.9 illustrate circumvention performance in two scenarios: using thresholds that yield the best performance for the circumvention detection problem, and p-induced thresh-

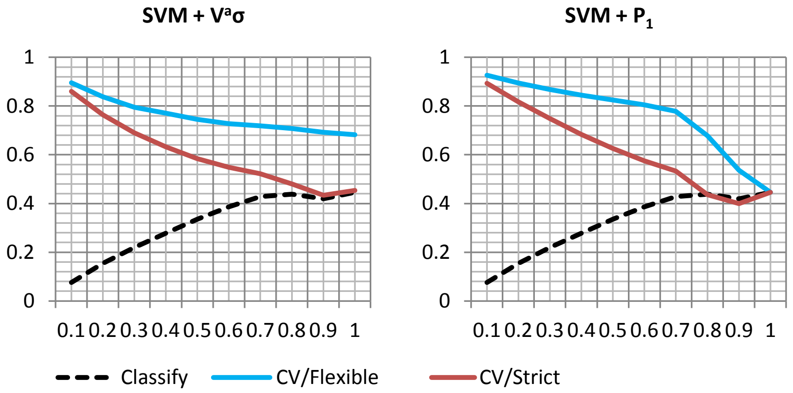


Figure 6.10: *Classify-Verify* F1-scores on $BLOG_L$ as a function of $p = 0.1, ..., 1.0$. Even in an online domain problem with many authors, *Classify-Verify* outperforms standard classifiers and successfully thwarts misclassifications in almost any setting.

olds derived from *Classify-Verify* performance in *standard* settings. For the latter, it means empirically selected thresholds that yield best performance on the non-attack documents, applied in attack scenarios. *Flexible* results suggest that *Classify-Verify* is successful in circumventing attacks for all values of p, especially for a configuration that targets attack detection. However lower, attack detection in non-attack environment is also proven effective across the range of p values. All F1-scores illustrated in Fig. 6.9 are detailed in the Tab. B.3. Complete F1-scores across all configurations are illustrated in Fig. B.9–B.12.

6.3.4 Many Authors in Online Domain Settings

The *Classify-Verify* method aims to provide a solution for hybrid problems commonly defined over online domains, where the set of candidate authors may be very large. Therefore we evaluate it on the $BLOG_L$ dataset, which contains blogs by 911 authors. F1-scores illustrated in Fig. 6.10 suggest that *Classify-Verify* is successful in such settings as well. The results are similar to those attained for $BLOG_S$, however with a slight difference: the best standalone verifier (V_σ^a) presents a more robust *flexible* performance than the best classifier-induced P_1 verifier, and outperforms it for $p = 0.9$ and $p = 1.0$. All F1-scores illustrated in Fig. 6.10 are detailed in Tab. B.4. Complete F1-scores across all configurations are illustrated in Fig. B.13.

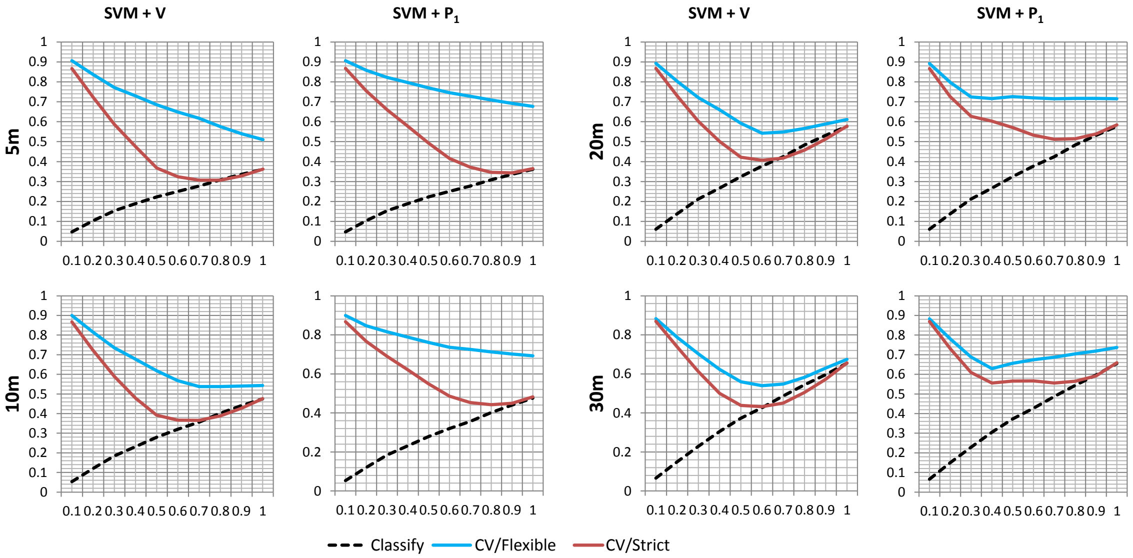


Figure 6.11: *Classify-Verify* F1-scores on *AAUTH* as a function of $p = 0.1, ..., 1.0$. *Classify-Verify* successfully thwarts misclassifications and outperforms standard classifiers in any setting, in spite of the noisy and inconsistent nature of the data.

6.3.5 Active Authentication Settings

We evaluate *Classify-Verify* on the *AAUTH* dataset, a dataset that presents unique settings highly applicable to continuous authentication security systems. We follow the experimental setup in Ch. 5 [36, 37, 102] and evaluate the dataset on documents generated from sliding windows of a fixed size and overlap, over the user keyboard input streams. Since the small-sized windows have shown low availability and contribution to the decision process, we focus on 4 sizes of sliding windows: 5, 10, 20 and 30 minutes (the last is not applied in the original *AAUTH* evaluation), all with an overlap of 1 minute. For instance, a system that uses a 5-min window with 1-min overlap is able to produce a decision every minute based on the past 5 minutes (starting at 5 minutes after the day begins).

It is notable that this unique dataset contains not just final products of user writings, but a complete keyboard input, with 1-character placeholders for special characters like `alt` represented by α, `backspace` as β etc. This provides the ability to capture not only writing style, but typing style as well. For instance, `ch`$\beta\beta$`Cch`$\beta\beta$`hicago` instead of `Chicago` captures typo and correction frequencies.

F1-scores, illustrated in Fig. 6.11, suggest that *Classify-Verify* is successful in thwarting false

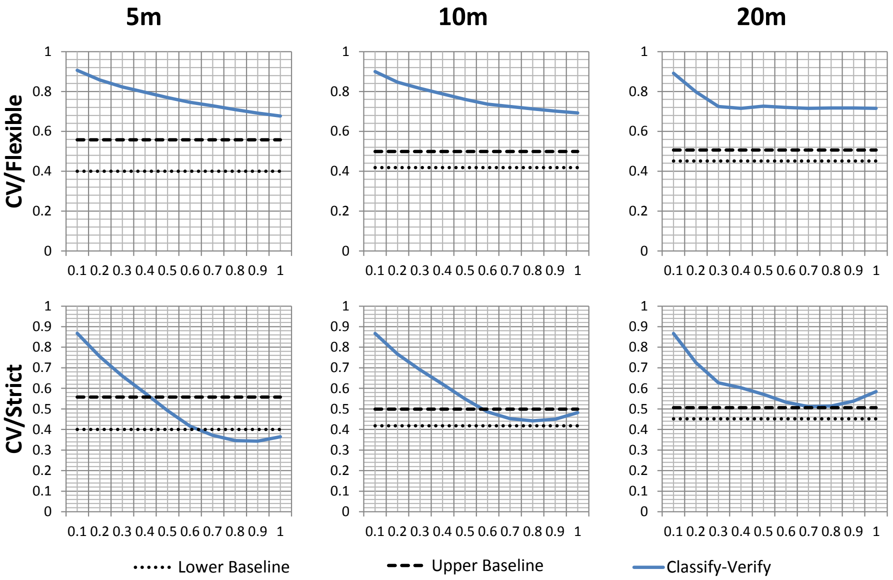


Figure 6.12: *Classify-Verify* F1-scores on $AAUTH$ using SVM with P_1 as a function of $p = 0.1, ..., 1.0$ for 5, 10 and 20 minute windows, compared to the range of F1-scores derived in the original evaluation in Ch. 5 in closed-world settings ($p = 1$). *Classify-Verify* in *flexible* configuration outperforms the original evaluation for any value of p; *strict* configuration results are mixed.

positives in active authentication settings as well. As expected, baseline classification performance improves as the size of the window increases, since more data exists to model that window's style. However, *flexible* results using SVM with P_1 suggest that the smaller windows, 5 and 10 minutes, perform better. This may be a consequence of having little data in such small windows, which results in a high rate of misclassifications that are later thwarted by the verifier, resulting in true-positives for the "unknown" class under the *flexible* configuration. These windows do not, in fact, provide us more information than the larger ones (20 and 30 minutes), as can be seen by their lower F1-scores for the *strict* evaluation. However, the *flexible* results support the effectiveness of *Classify-Verify* in that it successfully maintains a rather high and steady F1-score, regardless of the quality of the data in hand – when the window size shrinks, and therefore more misclassifications occur, the rate of instances classified as "unknown" legitimately grows. All F1-scores illustrated in Fig. 6.11 are detailed in Tab. B.5. Complete F1-scores across all configurations are illustrated in Fig. B.14.

In order to truly asses *Classify-Verify* performance over the data, we compared it with the results attained by the original evaluation in Ch. 5. Fig. 6.12 illustrates *Classify-Verify* F1-scores for 5, 10 and 20 minute windows as a function of $p = 0.1, 0.2, ..., 1.0$, using the best performing *Classify-*

Verify configuration – SVM with P_1. As baseline, we use the range of F1-scores attained by the original evaluation in Ch. 5, illustrated in lower and upper bounds. This range is a result of different minimum characters applied for window filtering, ranging from 100 to 1000, leading to a varying amount of data to model the window by, and thus its likelihood to be correctly classified. Moreover, baseline results are illustrated *only* in closed-world settings; it is likely to assume baseline results would become worse as p decreases.

The comparison suggests that *Classify-Verify* is indeed successful in thwarting misclassifications, and outperforms baseline results with the *flexible* configuration. The comparison with the *strict* results suggests that *Classify-Verify* becomes more aggressive in classification rejections as the window size decreases, and therefore is outperformed by the baseline methodology for small windows and high p value. However, given *Classify-Verify* provides a decision for *all* windows (no minimum characters filters are applied) in varying *in-set/not-in-set* scenarios, and the aggressive rejection rate can be relaxed by manually tuning thresholds for small windows, *Classify-Verify* is still suggested as the preferred technique.

6.3.6 Additional Experiments

Aside from the various experimental settings evaluated in the previous sections, we evaluate two additional approaches in an attempt to further improve the *Classify-Verify* performance. Unfortunately, both approaches have proven unsuccessful in improving the straight forward *Classify-Verify* approach taken above; therefore we layout the experimental settings and approaches as grounds for future work.

Verification Fusion

As shown effective on $AAUTH$, fusion of verifiers in a decision fusion center where the verifiers are distinctively different can increase the overall precision [8, 36]. For that purpose we construct a decision fusion center (DFC) and apply the the Chair-Varshney fusion rule [23], used to formulate a “fusion verifier” for the *verify* phase of *Classify-Verify*.

For any *classify* phase output author A_i, the verification problem in the *verify* phase is defined as a binary hypothesis testing problem with 2 hypotheses:

$$\begin{cases} H_0 & \text{reject – output } \perp \\ H_1 & \text{accept – output } A_i \end{cases} \tag{6.1}$$

The DFC produces a global decision based on local decisions $\{u_1, u_2, ..., u_n\}$ generated by the different verifiers, for which we use subsets of V, V_σ, V_σ^a and P_1, where:

$$u_i = \begin{cases} -1 & \text{if } H_0 \text{ is declared} \\ +1 & \text{if } H_1 \text{ is declared} \end{cases} \tag{6.2}$$

The optimal decision rule is expressed as follows:

$$f(u_1, u_2, ..., u_n) = \begin{cases} 1 & \text{if } a_0 + \sum_{i=1}^{n} a_i u_i > 0 \\ -1 & \text{otherwise} \end{cases} \tag{6.3}$$

With the a priori probabilities $P_0 = \Pr(H_0)$ and $P_1 = \Pr(H_1)$, and P_M^i and P_F^i representing the false accept and false reject rates of the i^{th} verifier, respectively, the optimal weights minimizing the global probability of error are given by:

$$a_0 = \log \frac{P_1}{P_0} \qquad a_i = \begin{cases} \log \frac{1-P_M^i}{P_F^i} & \text{if } u_i = +1 \\ \log \frac{1-P_F^i}{P_M^i} & \text{if } u_i = -1 \end{cases} \tag{6.4}$$

P_M^i and P_F^i are calculated empirically via cross-validation on the training data. The a priori probabilities of the hypotheses P_0 and P_1 are unknown, therefore a_0 is set in a range of values, and an "oracle" value is used – the one that yields the highest results on the test set.

F1-scores using fusion for the *verify* phase in *Classify-Verify* on *EBG* and *BLOG*$_S$ are illustrated in Fig. 6.13. The attained results do not surpass the best *Classify-Verify* configuration already known from Sec. 6.3.1, perhaps due to the high similarity of the fused verifiers, whereas the fusion algorithm assumes the verifiers are distinctly different.

Divide-and-Conquer: Scaling *Classify-Verify*

In addition to applying fusion, we attempted to apply a divide-and-conquer scheme on *EBG* and *BLOG*$_S$. Since the underlying classifiers perform better the smaller the problem is (in terms of number of authors), we tried the following formulation for a problem over n authors:

- Divide the problem into k randomly chosen, non-overlapping problems of size $\frac{n}{k}$
- Apply *Classify-Verify* on each of the k subproblems, classifying *all* test documents
- For every document D that has been classified k times (one per subproblem), discard all subproblems that outputted $\perp$

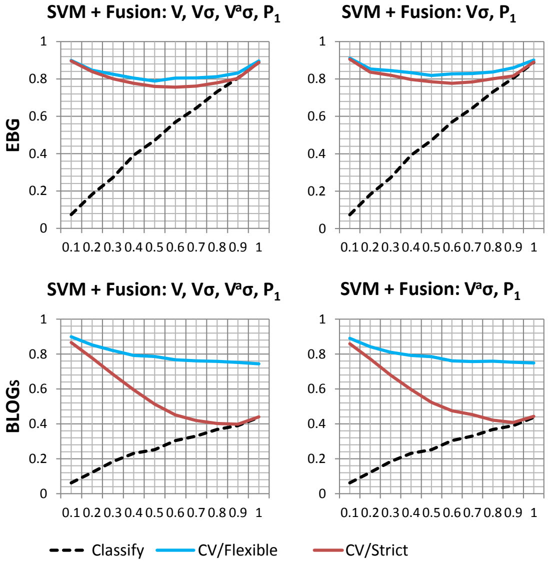


Figure 6.13: *Classify-Verify* F1-scores on *EBG* and $BLOG_S$ using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$, using the Chair-Varshney fusion algorithm for verification. The left column shows results for fusing V, V_σ, V_σ^a and P_1; the right column shows results for fusing only the best classifier-induced and standalone verifiers. None outperforms *Classify-Verify* with the best verifier alone, unfused.

- Run a final *Classify-Verify* round for each D among all non-rejected subproblem author outputs, and assign D the chosen author, or $\perp$ if rejected

The idea behind this formulation is to utilize *Classify-Verify* to discard batches of authors in the first round, resulting with a smaller set of authors for the final round, thus increasing the probability of a successful classification. In practice, none of the scaling formulations applied have outperformed applying *Classify-Verify* on the entire set. F1-scores for different problem breakdowns are detailed in Tab. 6.2.

6.4 Conclusions

Whether stylometry can be applied accurately in open-world settings has important privacy implications for both anonymous authors and those who fall in a suspect set and thus in danger of being falsely accused of authorship. These implications extend to security applications involving linguistic detection metrics, like authentication systems based on usage of common input devices.

The *Classify-Verify* method presented in this chapter provides a solution for mixed closed/open-world scenarios, with validated performance in real open-world settings, tested in a varying range

Table 6.2: F1-scores for *Classify-Verify* applied in a divide-and-conquer formulation on *EBG* and $BLOG_S$. Numbers in the leftmost column represent the configured subproblem size k, and in parentheses – its size in practice. None of the scaling experiments outperform applying *Classify-Verify* straight-forwardly on the complete problem.

EBG	**Classify Only**	***Flexible Classify-Verify***	***Strict Classify-Verify***
All 45	**0.889**	**0.926**	**0.898**
Scaling 5	0.587	0.619	0.59
Scaling 10 (11-12)	0.80	0.851	0.815
Scaling 15	0.833	0.865	0.843
Scaling 20 (22-23)	0.868	0.894	0.874
$\boldsymbol{BLOG_S}$			
All 50	**0.437**	**0.766**	**0.445**
Scaling 5	0.137	0.791	0.138
Scaling 10	0.327	0.553	0.328
Scaling 15 (16-17)	0.366	0.67	0.369
Scaling 20 (25)	0.43	0.732	0.442

of *in-set*/*not-in-set* probabilities.

Classify-Verify is proven effective not only in open-world settings where authors may be missing from the training set, but can also improve results in closed-world settings, by abstaining from low-confidence classification decisions. It is shown to perform well over various domains and problems, including formal writings, online blogs and noisy keyboard streams; small to large number of candidates, with *in-set* values that range from 10% to 100%; authors unaware they are being attempted identification, to those who try to attack the detection systems by hiding their style; and in different perspectives over performance that differentiate between the origin of thwarted misclassifications expressed as *flexible* and *strict* evaluation. In all the configurations above, *Classify-Verify* triumphs traditional classifiers by dismissing misclassification in favor of truly claiming: author is unknown.

From the various mixtures of underlying closed-world classifiers and open-world verifiers used in the *Classify-Verify* engine, the classifier-induced method that uses SVM with a simple threshold applied on the chosen class probability, namely P_1, seems to outperform the rest. This suggests that there is enough information within such closed-world classifiers to apply abstaining techniques, thwart misclassifications and thus increase precision.

We conclude that *Classify-Verify* should be the preferable approach taken over standard stylometry classifiers in both closed and open-world settings. We propose *Classify-Verify* should be adopted for other security and privacy domains, like it has been successfully applied for website fingerprinting attacks [52].

In addition, we propose future work to focus on examining additional verifiers. As shown effective on $AAUTH$, fusion of verifiers in a decision fusion center where the verifiers are distinctively different can increase the overall precision [8, 36]. However, as attempted in Sec. 6.3.6, applying fusion using the Chair-Varshney fusion algorithm [23] on EBG and $BLOG_S$ does not produce results that surpass the best *Classify-Verify* configuration already known, perhaps due to high similarity of the fused verifiers. We propose to apply fusion with additional distinct linguistic verifiers (differentiated by classification algorithm and/or features), and attempt fusion techniques in active authentication settings.

7. Conclusion

The application of stylometry to authorship attribution is a practice that stretches back decades, with amounts of accumulated research and satisfactory solutions that deem problems in the field – over hundreds and even thousands of potential authors – solved. Nevertheless, the continuous growth of online discourse, domains with ever-increasing pools of authors and effective countermeasures against author identification, require the development of authorship attribution approaches that are robust to these challenging settings.

This work has shown how *authorship verification*, the stylometric application of one-class machine learning approaches, is efficient in tackling these novel problem domains by setting confidence levels as gatekeepers on classifiers decision making, rather than operate under the assumptions of classic stylometry that chooses the best out of a closed set of options, regardless of the possibility the true author could be absent.

The work on native language and language family identification discussed in Ch. 4 demonstrated the utilization of verification in generalizing classification problems in favor of improving their solution. The main contribution of this work is the methodology by which native language of a non-native English author is extracted from English text. In this two-step classification process, verification is used to identify low-confidence decisions, and narrow down the decision domain for those instances to languages in only one prospective language family (identified in an additional standalone classification process). The success of this approach illustrates how thresholding over class probability distributions extracted from distance-based classifiers can benefit the verification process. Moreover, it shows how verification over those classifier statistics help to identify when additional information is required, and how that information can be extracted from the same data – but in a different, broader perspective, language families in this particular case. This type of verification applied in a generalization process can be applied to other domains in machine learning in general, and stylometry in particular, where classes can be clustered into identifiable groups, like language families.

The work on active linguistic authentication discussed in Ch. 5, and revisited in Ch. 6, illustrated the usage of linguistic style learning for security, manifested in stylometric modalities applied in an active authentication system. The various sliding-window configured sensors over the user's keyboard input demonstrated how continuous security applications can benefit from introducing this type of high-level verification. Thresholds set over decision availability of the different sensors, that along with the inspected window size are inversely proportional to the decision frequency, demonstrate the delicate tune-ups required for producing an accurate and usable decisions for access control purposes. With that in mind, these modalities have been shown to provide a unique quantification of user identity, and therefore should be considered for usage in such systems.

Both applications above demonstrate the effectiveness and importance of authorship verification utilization in various applications, leading to the main novelty presented in this work, the *Classify-Verify* algorithm. As discussed and demonstrated extensively in Ch. 6, this hybrid approach for mixed closed-world and open-world stylometry, which interleaves classic one-of-many classification with a binary verification decision step, has been shown to provide a more accurate and *confident* view of the stylometric problem domain. With *Classify-Verify*, misclassifications are thwarted in favor of a somewhat less complete, yet more accurate decision process, in which verification-driven certainty is the leading principle. *Classify-Verify* demonstrates a fusion of the best of both worlds, where high accuracy closed-world procedures are applied to narrow several choices down to one, which is later put to a binary test by a suspicious, tunable verifier. We conclude that this hybrid approach is preferable over standard, limited closed-world approaches, and should be adopted especially for open-world (or semi-open-world) problem domains, including but not limited to: problems with a large candidate set, adversarial settings, challenging active-authentication systems, online domains and the like.

The applications of authorship verification presented in this document provide a strong incentive to adopt this approach for stylometric analysis. Verification is well-structured to handle uncertainty that may originate in dynamic environments and mass data, such as the Internet, underground communities and any online domain. The advantages of applying verification to these fast-emerging realms compared to classic approaches are not only numerous, but even necessary. This thesis proposes future research of stylometric applications should focus on verification tools and methodologies, such as those presented in this document. Research should utilize the vast set of corpora collected over years of stylometry research in order to establish a solid ground of empirical evaluations of verification approaches; Promising approaches such as decision fusion should be further explored; And these approaches should be tested in extreme measures that include adversarial settings and other security and privacy oriented applications.

Bibliography

[1] Ahmed Abbasi and Hsinchun Chen. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Trans. Inf. Syst.*, 26(2):1–29, 2008.

[2] S. Afroz, M. Brennan, and R. Greenstadt. Detecting hoaxes, frauds, and deception in writing style online. In *Proceedings of the 33rd conference on IEEE Symposium on Security and Privacy*. IEEE, 2012.

[3] A.A.E. Ahmed and I. Traore. A new biometric technology based on mouse dynamics. *Dependable and Secure Computing, IEEE Transactions on*, 4(3):165 –179, july-sept. 2007.

[4] A.A.E. Ahmed and I. Traore. A new biometric technology based on mouse dynamics. *Dependable and Secure Computing, IEEE Transactions on*, 4(3):165 –179, July-Sept. 2007.

[5] Bashir Ahmed, Sung-Hyuk Cha, and Charles Tappert. Language identification from text using n-gram based cumulative frequency addition. Proc. CSIS Research Day, May 2004.

[6] Charles S. Ahn. Automatically detecting authors' native language. Thesis, Naval Postgraduate School, March 2011.

[7] Navot Akiva and Moshe Koppel. Identifying distinct components of a multi-author document. In *EISIC*, pages 205–209, 2012.

[8] K.M. Ali and M.J. Pazzani. *On the link between error correlation and error reduction in decision tree ensembles*. Citeseer, 1995.

[9] Livia CF Araujo, Luiz HR Sucupira Jr, Miguel G Lizarraga, Lee L Ling, and Joao BT Yabu-Uti. User authentication through typing biometrics features. *Signal Processing, IEEE Transactions on*, 53(2):851–855, 2005.

[10] Shlomo Argamon, Moshe Koppel, James W. Pennebaker, and Jonathan Schler. Mining the blogosphere: Age, gender and the varieties of self-expression. *First Monday*, 12(9), 2007.

[11] Harald Baayen, Hans van Halteren, Anneke Neijt, and Fiona Tweedie. An experiment in authorship attribution. In *6th JADT*, pages 29–37. Citeseer, 2002.

[12] Ned Bakelman, John V. Monaco, Sung-Hyuk Cha, and Charles C. Tappert. Continual keystroke biometric authentication on short bursts of keyboard input. In *Proceedings of Student-Faculty Research Day, CSIS, Pace University*, 2012.

[13] Francesco Bergadano, Daniele Gunetti, and Claudia Picardi. User authentication through keystroke dynamics. *ACM Trans. Inf. Syst. Secur.*, 5(4):367–397, November 2002.

[14] Chris M Bishop. Novelty detection and neural network validation. In *Vision, Image and Signal Processing, IEE Proceedings-*, volume 141, pages 217–222. IET, 1994.

[15] Christopher M Bishop et al. Neural networks for pattern recognition. 1995.

[16] Michael Brennan, Sadia Afroz, and Rachel Greenstadt. Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Trans. Inf. Syst. Secur.*, 15(3):12:1–12:22, November 2012.

[17] Julian Brooke and Graeme Hirst. Native language detection with 'cheap' learner corpora. In *The 2011 Conference of Learner Corpus Research (LCR2011)*, 2011.

[18] Julian Brooke and Graeme Hirst. Measuring interlanguage: Native language identification with l1-influence metrics. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may 2012. European Language Resources Association (ELRA).

[19] Kevin Burton, Akshay Java, and Ian Soboroff. The icwsm 2009 spinn3r dataset. In *Proceedings of the Third Annual Conference on Weblogs and Social Media (ICWSM 2009), San Jose, CA*, 2009.

[20] Lyle Campbell and William J. Poser. *Language Classification: History and Method.* Cambridge University Press, 2008.

[21] Gail A Carpenter, Stephen Grossberg, and David B Rosen. Art 2-a: An adaptive resonance algorithm for rapid category learning and recognition. *Neural networks*, 4(4):493–504, 1991.

[22] Maria Luisa Carrio-Pastor. Contrasting specific english corpora: Language variation. *International Journal of English Studies, Special Issue*, pages 221–233, 2009.

[23] Z. Chair and P.K. Varshney. Optimal data fusion in multiple sensor detection systems. *Aerospace and Electronic Systems, IEEE Transactions on*, AES-22(1):98 –101, jan. 1986.

[24] Carole E Chaski. Best practices and admissibility of forensic author identification. *JL & Pol'y*, 21:333–725, 2013.

[25] Ghinwa F. Choueiter, Geoffrey Zweig, and Patrick Nguyen. An empirical study of automatic accent classification. In *ICASSP*, pages 4265–4268, 2008.

[26] C Chow. On optimum recognition error and reject tradeoff. *Information Theory, IEEE Transactions on*, 16(1):41–46, 1970.

[27] Jonathan H. Clark and Charles J. Hannon. A classifier system for author recognition using synonym-based features. In *Proceedings of the artificial intelligence 6th Mexican international conference on Advances in artificial intelligence*, MICAI'07, pages 839–849, Berlin, Heidelberg, 2007. Springer-Verlag.

[28] Paul Clough. Plagiarism in natural and programming languages: an overview of current tools and technologies, 2000.

[29] Rosa María Coyotl-Morales, Luis Villaseñor-Pineda, Manuel Montes-y Gómez, and Paolo Rosso. Authorship attribution using word sequences. In *Progress in Pattern Recognition, Image Analysis and Applications*, pages 844–853. Springer, 2006.

[30] Walter Daelemans, Jakub Zavrel, Ko Van der Sloot, and Antal Van den Bosch. Timbl: Tilburg memory-based learner. *version*, 4:02–01, 2003.

[31] Roger Dingledine, Nick Mathewson, and Paul F. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium, August 9-13, 2004, San Diego, CA, USA*, pages 303–320, 2004.

[32] Richard O. Duda and Peter E. Hart. Pattern classification and scene analysis. *Wiley-Interscience, New York*, 1973.

[33] Hugo Jair Escalante, Manuel Montes y Gómez, and Luis Villaseñor Pineda. Particle swarm model selection for authorship verification. In *CIARP*, pages 563–570, 2009.

[34] Dominique Estival, Tanja Gaustad, Son B. Pham, Will Radford, and Ben Hutchinson. Author profiling for english emails. In *10th Conference of the Pacific Association for Computational Linguistics (PACLING 2007)*, pages 262–272, 2007.

[35] Winthrop Nelson Francis and Henry Kucera. *Frequency Analysis of English Usage: Lexicon and Grammar.* Houghton Mifflin, 1983.

[36] Alex Fridman, Ariel Stolerman, Sayandeep Acharya, Patrick Brennan, Patrick Juola, Rachel Greenstadt, and Moshe Kam. Decision fusion for multimodal active authentication. *IT Professional*, 15(4):29–33, 2013.

[37] Lex Fridman, Ariel Stolerman, Sayandeep Acharya, Patrick Brennan, Patrick Juola, Rachel Greenstadt, and Moshe Kam. Multi-modal decision fusion for continuous authentication. *Computers & Electrical Engineering*, (0):–, 2014.

[38] Erin Elizabeth Gibbons. The effects of second language experience on typologically similar and dissimilar third language. Thesis, Brigham Young University, Center for Language Studies, 2009.

[39] Felix Golcher and Marc Reznicek. Stylometry and the interplay of topic and l1 in the different annotation layers in the falko corpus. In *Humboldt-Universitat zu Berlin*, QITL-4, 2009. [Online: Stand 2012-03-22T16:09:09Z].

[40] Sylvaine Granger, Estelle Dagneaux, and Fanny Meunier. *International Corpus of Learner English : Version 1 ; Handbook and CD-ROM.* Pr. Univ. de Louvain, Louvain-la-Neuve, 2002.

[41] Sylvaine Granger, Estelle Dagneaux, Magali Paquot, and Fanny Meunier. *The International Corpus of Learner English, Version 2: Handbook and CD-Rom.* Pr. Univ. de Louvain, Louvain-la-Neuve, 2009.

[42] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Mach. Learn.*, 46(1-3):389–422, March 2002.

[43] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[44] Hans Van Halteren. Author verification by linguistic profiling: An exploration of the parameter space. *ACM Trans. Speech Lang. Process.*, 4(1):1:1–1:17, February 2007.

[45] S. Hashem and B. Schmeiser. Improving model accuracy using optimal linear combinations of trained neural networks. *Neural Networks, IEEE Transactions on*, 6(3):792–794, 1995.

[46] Radu Herbei and Marten H Wegkamp. Classification with reject option. *Canadian Journal of Statistics*, 34(4):709–721, 2006.

[47] David I Holmes and Richard S Forsyth. The federalist revisited: New directions in authorship attribution. *Literary and Linguistic Computing*, 10(2):111–127, 1995.

[48] John Houvardas and Efstathios Stamatatos. N-gram feature selection for authorship identification. In *Artificial Intelligence: Methodology, Systems, and Applications*, pages 77–86. Springer, 2006.

[49] L.W. James. Fundamentals of biometric authentication technologies. *International Journal of Image and Graphics*, 1(01):93–113, 2001.

[50] Nathalie Japkowicz. *Concept-learning in the absence of counter-examples: an autoassociation-based approach to classification.* PhD thesis, Rutgers, The State University of New Jersey, 1999.

[51] Nathalie Japkowicz, Catherine Myers, and Mark Gluck. A novelty detection approach to classification. In *International joint conference on artificial intelligence*, volume 14, pages 518–523. Lawrence Erlbaum Associates Ltd., 1995.

[52] Marc Juarez, Sadia Afroz, Gunes Acar, Claudia Diaz, and Rachel Greenstadt. A critical evaluation of website fingerprinting attacks.

[53] P. Juola. Jgaap, a java-based, modular, program for textual analysis, text categorization, and authorship attribution.

[54] Patrick Juola. Ad-hoc authorship attribution competition. In *Proc. 2004 Joint International Conference of the Association for Literary and Linguistic Computing and the Association for Computers and the Humanities (ALLC/ACH 2004)*, Göteborg, Sweden, June 2004.

[55] Patrick Juola. Authorship attribution. *Foundations and Trends in information Retrieval*, 1(3):233–334, 2008.

[56] Patrick Juola, John I. Noecker, Ariel Stolerman, Michael V. Ryan, Patrick Brennan, and Rachel Greenstadt. Keyboard-behavior-based authentication. *IT Professional*, 15(4):8–11, 2013.

[57] Patrick Juola, John Noecker Jr., Ariel Stolerman, Michael V. Ryan, Patrick Brennan, and Rachel Greenstadt. A dataset for active linguistic authentication. In *Proceedings of the Ninth Annual IFIP WG 11.9 International Conference on Digital Forensics*, Orlando, Florida, USA, January 2013. National Center for Forensic Science.

[58] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice-Hall Inc., 2 edition, 2009.

[59] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas. On combining classifiers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(3):226–239, 1998.

[60] Bryan Klimt and Yiming Yang. Introducing the enron corpus. In *CEAS*, 2004.

[61] Mark W Koch, Mary M Moya, Larry D Hostetler, and R Joseph Fogler. Cueing, feature discovery, and one-class learning for synthetic aperture radar automatic target recognition. *Neural Networks*, 8(7):1081–1102, 1995.

[62] Teuvo K. Kohonen, Manfred R. Schroeder, and Thomas S. Huang. *Self-Organizing Maps*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition, 2001.

[63] Moshe Koppel, Navot Akiva, Idan Dershowitz, and Nachum Dershowitz. Unsupervised decomposition of a document into authorial components. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1356–1364, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[64] Moshe Koppel and Jonathan Schler. Authorship verification as a one-class classification problem. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 62–, New York, NY, USA, 2004. ACM.

[65] Moshe Koppel, Jonathan Schler, and Shlomo Argamon. Computational methods in authorship attribution. *Journal of the American Society for information Science and Technology*, 60(1):9–26, 2009.

[66] Moshe Koppel, Jonathan Schler, and Shlomo Argamon. Authorship attribution in the wild. *Language Resources and Evaluation*, 45(1):83–94, 2011.

[67] Moshe Koppel, Jonathan Schler, Shlomo Argamon, and Yaron Winter. The "fundamental problem" of authorship attribution. *English Studies*, 93(3):284–291, 2012.

[68] Moshe Koppel, Jonathan Schler, and Elisheva Bonchek-Dokow. Measuring differentiability: Unmasking pseudonymous authors. *J. Mach. Learn. Res.*, 8:1261–1276, December 2007.

[69] Moshe Koppel, Jonathan Schler, and Kfir Zigdon. Automatically determining an anonymous author's native language. In *Proceedings of the 2005 IEEE international conference on Intelligence and Security Informatics*, ISI'05, pages 209–217, Berlin, Heidelberg, 2005. Springer-Verlag.

[70] Moshe Koppel, Jonathan Schler, and Kfir Zigdon. Determining an author's native language by mining a text for errors. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 624–628, New York, NY, USA, 2005. ACM.

[71] Moshe Koppel, Jonathan Schler, and Kfir Zigdon. Determining an author's native language by mining a text for errors. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 624–628, New York, NY, USA, 2005. ACM.

[72] Kim Luyckx and Walter Daelemans. Authorship attribution and verification with many authors and limited data. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 513–520, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[73] M.B. Malyutov. Authorship attribution of texts: A review. In Rudolf Ahlswede, Lars Bumer, Ning Cai, Harout Aydinian, Vladimir Blinovsky, Christian Deppe, and Haik Mashurian, editors, *General Theory of Information Transfer and Combinatorics*, volume 4123 of *Lecture Notes in Computer Science*, pages 362–380. Springer Berlin Heidelberg, 2006.

[74] Larry M Manevitz and Malik Yousef. One-class svms for document classification. *The Journal of Machine Learning Research*, 2:139–154, 2002.

[75] Colin Martindale and Dean McKenzie. On the utility of content analysis in author attribution: The federalist. *Computers and the Humanities*, 29(4):259–270, 1995.

[76] Andrew McDonald, Sadia Afroz, Aylin Caliskan, Ariel Stolerman, and Rachel Greenstadt. Use fewer instances of the letter ”i”: Toward writing style anonymization. In *Privacy Enhancing Technologies Symposium (PETS)*, 2012.

[77] Andrew WE McDonald, Jeffrey Ulman, Marc Barrowclift, and Rachel Greenstadt. Anonymouth revamped: Getting closer to stylometric anonymity. 2013.

[78] Charles E Metz. Basic principles of roc analysis. In *Seminars in nuclear medicine*, volume 8, pages 283–298. Elsevier, 1978.

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

[80] Arvind Narayanan, Hristo Paskov, Neil Zhenqiang Gong, John Bethencourt, Emil Stefanov, Eui Chul Richard Shin, and Dawn Song. On the feasibility of internet-scale author identification. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 300–314. IEEE, 2012.

[81] Smita Nirkhi and RV Dharaska. Comparative study of authorship identification techniques for cyber forensics analysis. *International Journal of Advanced Computer Science & Applications*, 4(5), 2013.

[82] John Noecker Jr. and Michael Ryan. Distractorless authorship verification. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA).

[83] M.S. Obaidat and B. Sadoun. Verification of computer users using keystroke dynamics. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 27(2):261 –269, apr 1997.

[84] T. Ord and SM Furnell. User authentication for keypad-based devices using keystroke analysis. In *Proceedings of the Second International Network Conference (INC-2000)*, pages 263–272, 2000.

[85] Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

[86] Hristo Spassimirov Paskov. *A regularization framework for active learning from imbalanced data*. PhD thesis, Massachusetts Institute of Technology, 2010.

[87] Tadeusz Pietraszek. Optimizing abstaining classifiers using roc analysis. In *Proceedings of the 22nd international conference on Machine learning*, pages 665–672. ACM, 2005.

[88] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schoelkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.

[89] Martin Potthast et al. Pan 2011 lab: Uncovering plagiarism, authorship, and social software misuse. Conference CFP at `http://pan.webis.de/`, 2011.

[90] Josyula R Rao and Pankaj Rohatgi. Can pseudonymity really guarantee privacy? In *Proceedings of the Ninth USENIX Security Symposium*, pages 85–96, 2000.

[91] Gunter Ritter and María Teresa Gallegos. Outliers in statistical pattern recognition and an application to automatic chromosome classification. *Pattern Recognition Letters*, 18(6):525–539, 1997.

[92] Stephen Roberts, Lionel Tarassenko, James Pardey, and David Siegwart. A validation index for artificial neural networks. In *Proceedings of Int. Conference on Neural Networks and Expert Systems in Medicine and Healthcare*, pages 23–30, 1994.

[93] Stephen J Roberts, William Penny, and David Pillot. Novelty, confidence and errors in connectionist systems. In *Intelligent Sensors (Digest No: 1996/261), IEE Colloquium on*, pages 10–1. IET, 1996.

[94] Joseph Rudman. The non-traditional case for the authorship of the twelve disputed federalist papers: A monument built on sand. *Proceedings of ACH/ALLC 2005*, 2005.

[95] Conrad Sanderson and Simon Guenter. Short text authorship attribution via sequence kernels, markov chains and author unmasking: an investigation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 482–491, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[96] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[97] D. Shanmugapriya and G. Padmavathi. A survey of biometric keystroke dynamics: Approaches, security and challenges. *Arxiv preprint arXiv:0910.0817*, 2009.

[98] T. Sim, S. Zhang, R. Janakiraman, and S. Kumar. Continuous verification using multimodal biometrics. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(4):687–700, 2007.

[99] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556, 2009.

[100] Benno Stein and Sven Meyer zu Eissen. Intrinsic plagiarism analysis with meta learning. In *PAN*, 2007.

[101] Ariel Stolerman, Aylin Caliskan, and Rachel Greenstadt. From language to family and back: Native language and language family identification from english text. In *Proceedings of the 2013 NAACL HLT Student Research Workshop*, pages 32–39, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[102] Ariel Stolerman, Alex Fridman, Rachel Greenstadt, Patrick Brennan, and Patrick Juola. Active linguistic authentication revisited: Real-time stylometric evaluation towards multi-modal decision fusion. In *The Tenth Annual IFIP WG 11.9 International Conference on Digital Forensics*, January 2014.

[103] Ariel Stolerman, Rebekah Overdorf, Sadia Afroz, and Rachel Greenstadt. Classify, but verify: Breaking the closed-world assumption in stylometric authorship attribution. In *The Tenth Annual IFIP WG 11.9 International Conference on Digital Forensics*, January 2014.

[104] D.M.J. Tax. One-class classification. Master's thesis, Delft University of Technology, Delft, June 2001.

[105] Laura Mayfield Tomokiyo and Rosie Jones. You're not from 'round here, are you?: naive bayes detection of non-native utterance text. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA, 2001. Association for Computational Linguistics.

[106] Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Human Language Technology Conference (HLT-NAACL 2003)*, 2003.

[107] Oren Tsur and Ari Rappoport. Using classifier features for studying the effect of native language on the choice of written second language words. In *Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition*, CACLA '07, pages 9–16, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[108] Fiona J Tweedie, Sameer Singh, and David I Holmes. Neural network applications in stylometry: The federalist papers. *Computers and the Humanities*, 30(1):1–10, 1996.

[109] Hans van Halteren. Linguistic profiling for authorship recognition and verification. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 199–206, Barcelona, Spain, July 2004.

[110] Hans van Halteren. Source language markers in europarl translations. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 937–944, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[111] Vladimir Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 1995.

[112] Vladimir N. Vapnik. *Statistical learning theory.* Wiley, 1998.

[113] R. Wanneroy, E. Bilinski, C. Barras, M. Adda-Decker, and E. Geoffrois. Acoustic-phonetic modeling of non-native speech for language identification. In *Proceedings of the ESCA-NATO Workshop on Multi-Lingual Interoperability in Speech Technology (MIST)*, The Netherlands, 1999.

[114] Sze-Meng Jojo Wong and Mark Dras. Contrastive analysis and native language identification. In *Proceedings of the Australasian Language Technology Association Workshop 2009*, pages 53–61, Sydney, Australia, December 2009.

[115] Sze-Meng Jojo Wong, Mark Dras, and Mark Johnson. Topic modeling for native language identification. In *Proceedings of the Australasian Language Technology Association Workshop 2011*, pages 115–124, Canberra, Australia, December 2011.

[116] R.V. Yampolskiy. Behavioral modeling: an overview. *American Journal of Applied Sciences*, 5(5):496–503, 2008.

[117] Alexander Ypma and Robert PW Duin. Support objects for domain approximation. Citeseer, 1998.

[118] Nan Zheng, Aaron Paloski, and Haining Wang. An efficient user verification system via mouse movements. In *Proceedings of the 18th ACM conference on Computer and communications security*, CCS '11, pages 139–150, New York, NY, USA, 2011. ACM.

[119] Marc A. Zissman. Automatic language identification using gaussian mixture and hidden markov models. In *Proceedings of the 1993 IEEE international conference on Acoustics, speech, and signal processing: speech processing - Volume II*, ICASSP'93, pages 399–402, Washington, DC, USA, 1993. IEEE Computer Society.

[120] Sven Meyer zu Eissen and Benno Stein. Intrinsic plagiarism detection. In *Advances in Information Retrieval*, pages 565–569. Springer, 2006.

[121] Sven Meyer zu Eissen, Benno Stein, and Marion Kulig. Plagiarism detection without reference collections. In *Advances in Data Analysis*, pages 359–366. Springer Berlin Heidelberg, 2007.

Appendix A. Native Language and Language Family Identification

A.1 Feature Breakdown by Experiment

Table A.1: Total number of attributes, rare POS bigrams percentage and spelling errors percentage for the basic feature set.

Task	**Total Features**		**Rare POS Bigrams**		**Spelling Errors**	
Value	Avg.	SD	Avg. %	SD %	Avg. %	SD %
5-L1	771.5	3.53	0.25%	0.001%	20.22%	0.09%
9-L1 vs. 3-LF	831	17	0.26%	0.07%	26.92%	1.40%
3-L1 vs. 3-LF: L1	705.44	18.84	0.25%	0.09%	12.64%	1.51%
3-L1 vs. 3-LF: LF	699.44	13.57	0.30%	0.11%	11.41%	0.84%
2-L1 vs. 2-sub-LF: L1	665.75	14.72	0.18%	0.07%	7.71%	1.04%
2-L1 vs. 2-sub-LF: LF	673.25	8.61	0.18%	0.18%	8.16%	0.69%
3-LF: train on 2, test on 1	766.33	3.21	0.26%	0.001%	21.18%	0.51%

A.2 *InfoGain* Feature Distributions

Table A.2: Feature-type average percentage (first row) and standard-deviation percentage (second row) distribution for the InfoGain feature set.

Task	Function Words	Letter Bigrams	Rare POS Bigrams	Spelling Errors	Common POS Bigrams
5-L1	25%	34%	0%	4.25%	36.75%
	1.41%	9.89%	0%	0.35%	8.13%
9-L1 vs 3-LF: L1	20.4%	29.37%	0%	2.68%	47.53%
	5.95%	7.76%	0%	1.58%	12.81%
9-L1 vs 3-LF: LF	25.28%	48.62%	0%	0.46%	25.62%
	5.11%	6.32%	0%	0.99%	4.82%
3-L1 vs 3-LF: L1	29.94%	33.94%	0.05%	3%	33.05%
	11.29%	18.21%	0.33%	5.36%	13.55%
3-L1 vs 3-LF: LF	17.44%	23.55%	0%	1.83%	57.16%
	6.64%	13.60%	0%	2.73%	19.41%
2-L1 vs. 2-sub-LF: L1	17.5%	8.25%	0%	2%	72.25%
	1.63%	4.43%	0%	2.16%	5.80%
2-L1 vs. 2-sub-LF: LF	27.37%	18.12%	0%	2.12%	52.37%
	18.64%	5.61%	0%	1.70%	23.47%
3-LF train on 2, test on 1	23.5%	49.5%	0%	0.5%	26.5%
	3.60%	1%	0%	1%	5.19%

Appendix B. The *Classify-Verify* Algorithm

B.1 Complete *Classify-Verify* Evaluation on *EBG* with $\langle 500, 2\rangle$-*chars*

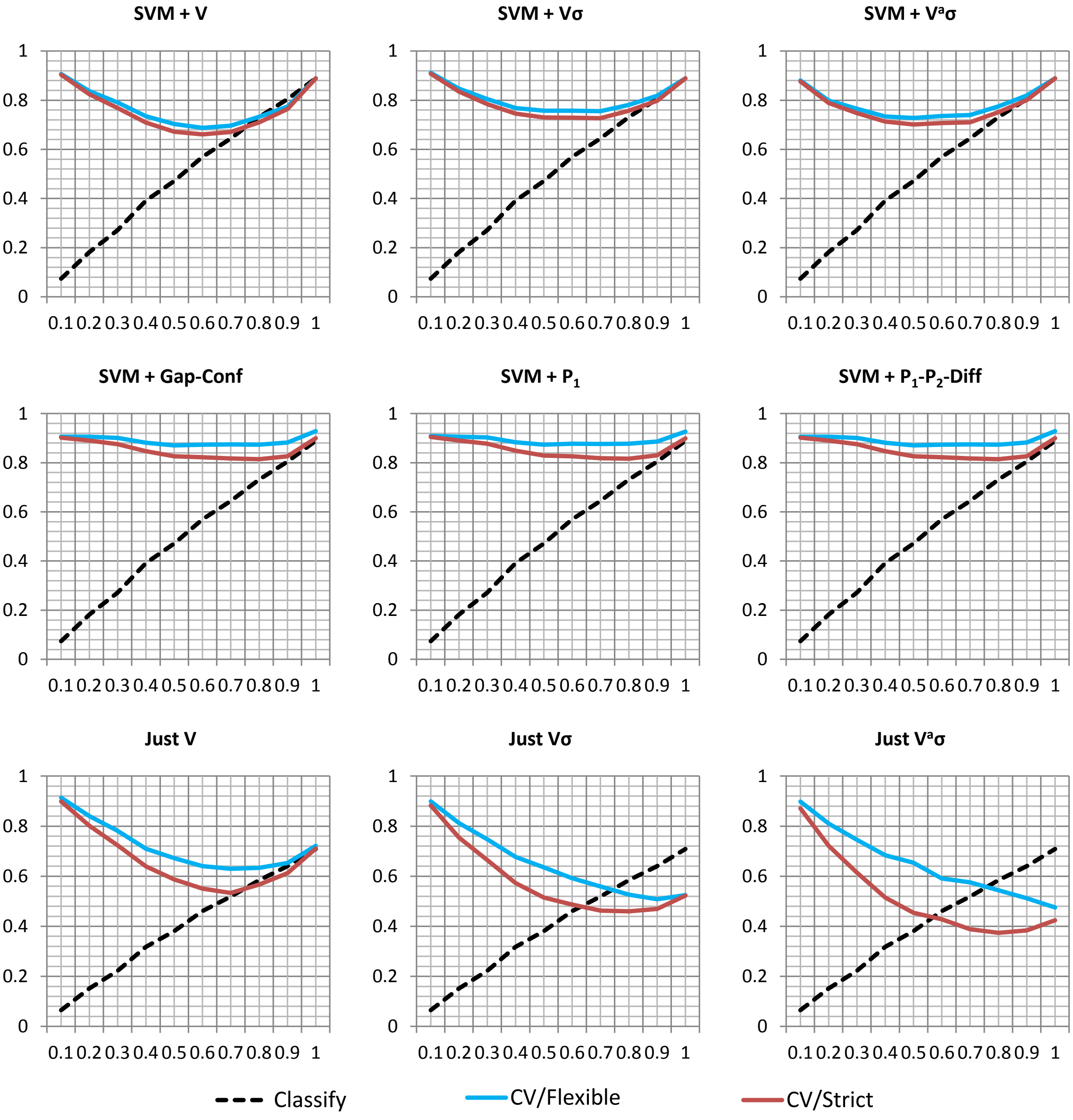


Figure B.1: *Classify-Verify* F1-scores on *EBG* using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$.

B.2 Complete *Classify-Verify* Evaluation on *EBG* with *Writeprints*

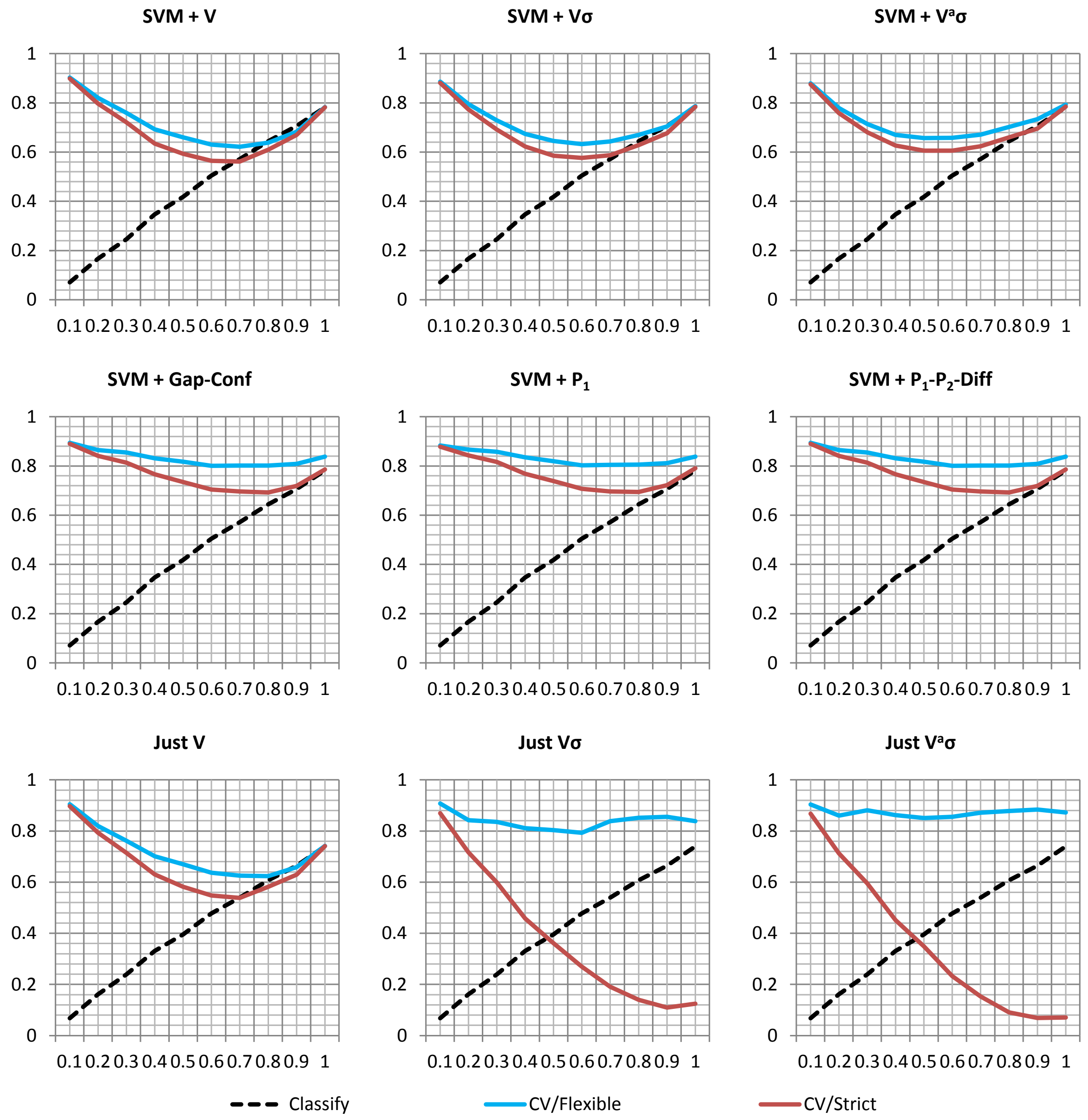


Figure B.2: *Classify-Verify* F1-scores on *EBG* using the *Writeprints* feature set as a function of $p = 0.1, ..., 1.0$.

B.3 Complete *Classify-Verify* Evaluation on $BLOG_S$ with $\langle 500, 2\rangle$-*chars*

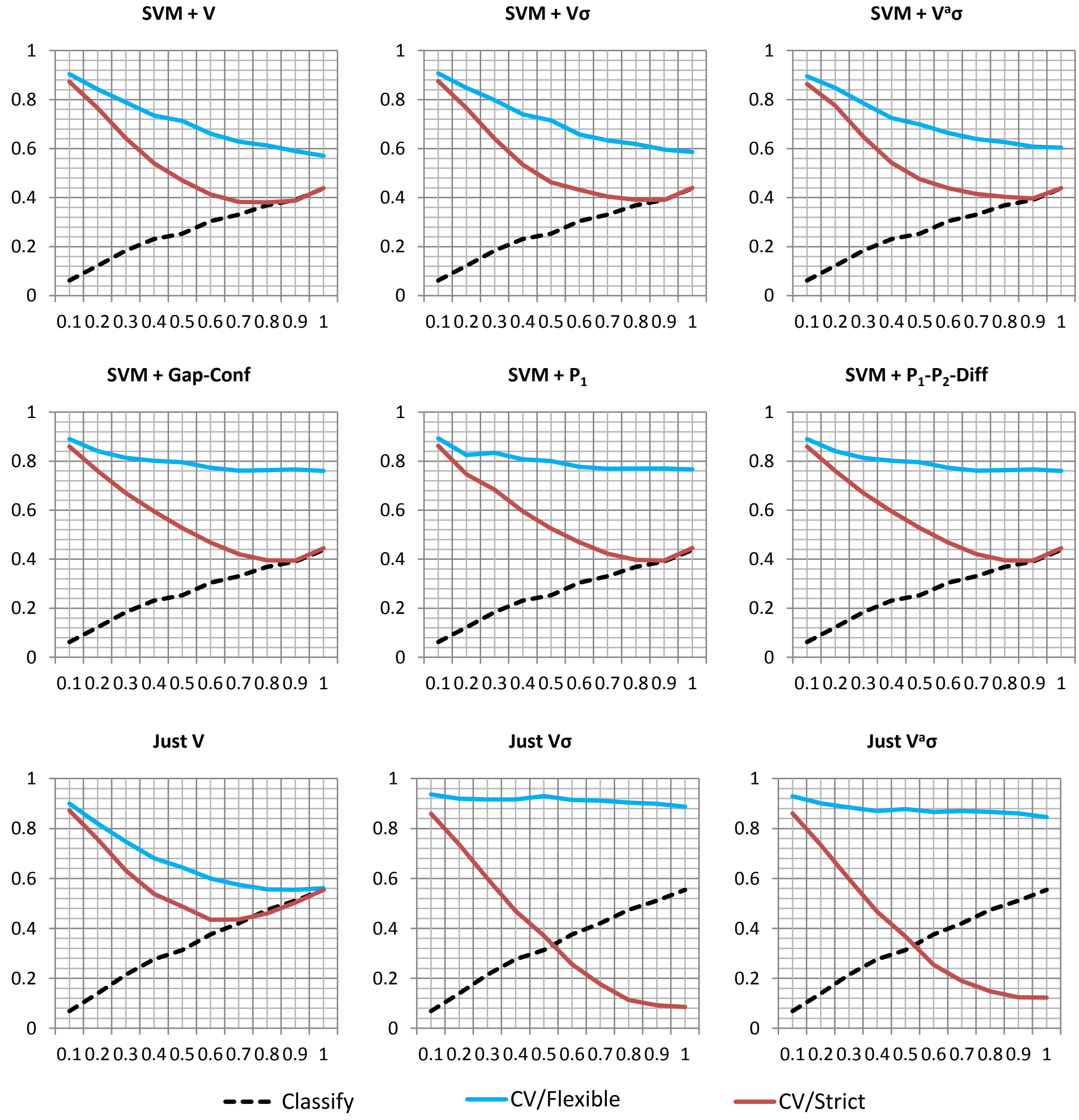


Figure B.3: *Classify-Verify* F1-scores on $BLOG_S$ using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$.

B.4 Complete *Classify-Verify* Evaluation on *BLOG*$_S$ with *Writeprints*

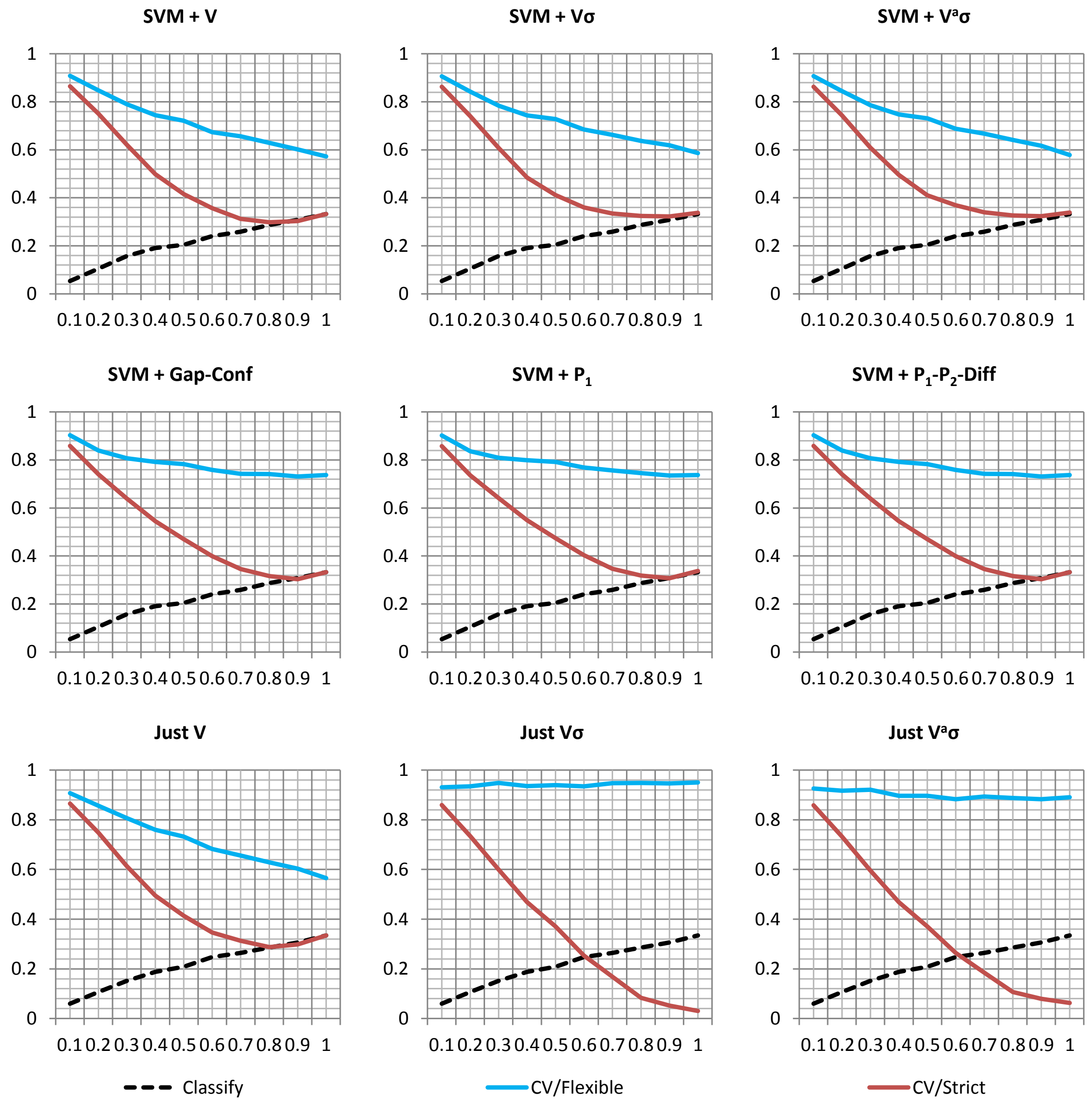


Figure B.4: *Classify-Verify* F1-scores on *BLOG*$_S$ using the *Writeprints* feature set as a function of $p = 0.1, ..., 1.0$.

B.5 Complete *Classify-Verify* Evaluation on *EBG* with $\langle 500, 2\rangle$-*chars* Using p-Induced Verification Thresholds

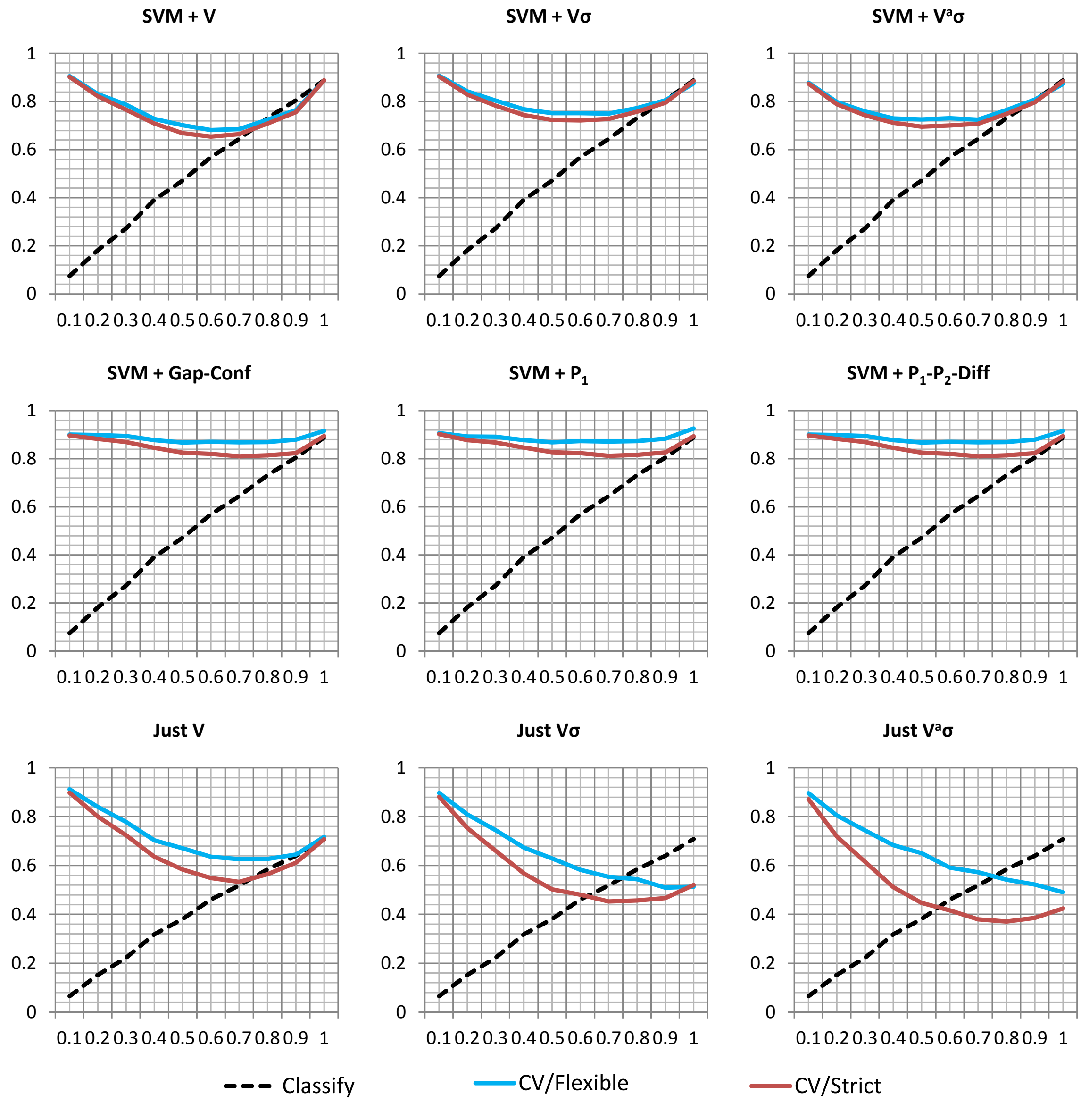


Figure B.5: *Classify-Verify* F1-scores on *EBG* using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$, using p-induced verification thresholds.

B.6 Complete *Classify-Verify* Evaluation on *EBG* with $\langle 500, 2\rangle$-*chars* Using *Robust* Thresholds

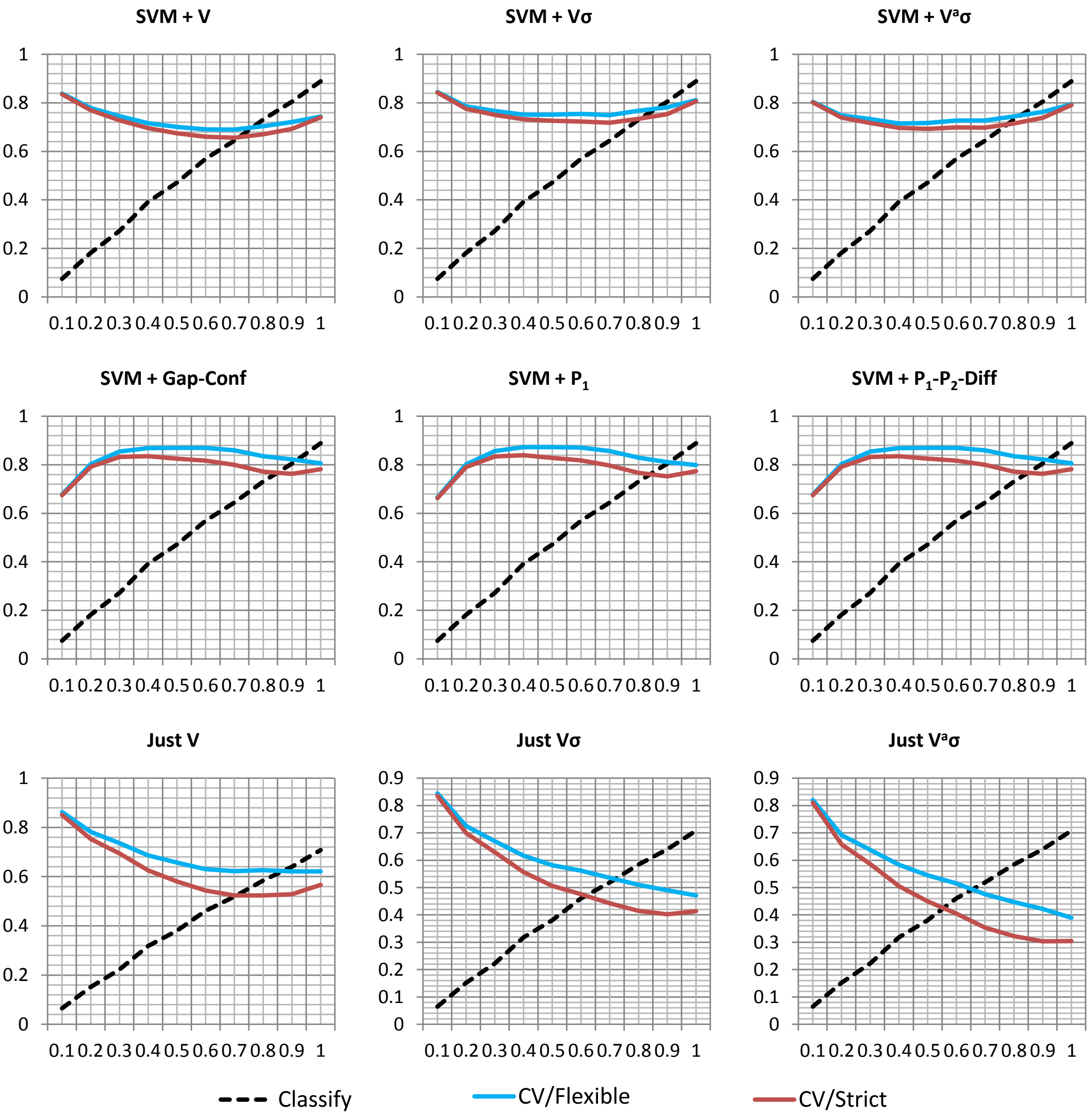


Figure B.6: *Classify-Verify* F1-scores on *EBG* using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$, using robust verification thresholds.

B.7 Complete *Classify-Verify* Evaluation on $BLOG_S$ with $\langle 500, 2\rangle$-*chars* Using p-Induced Verification Thresholds

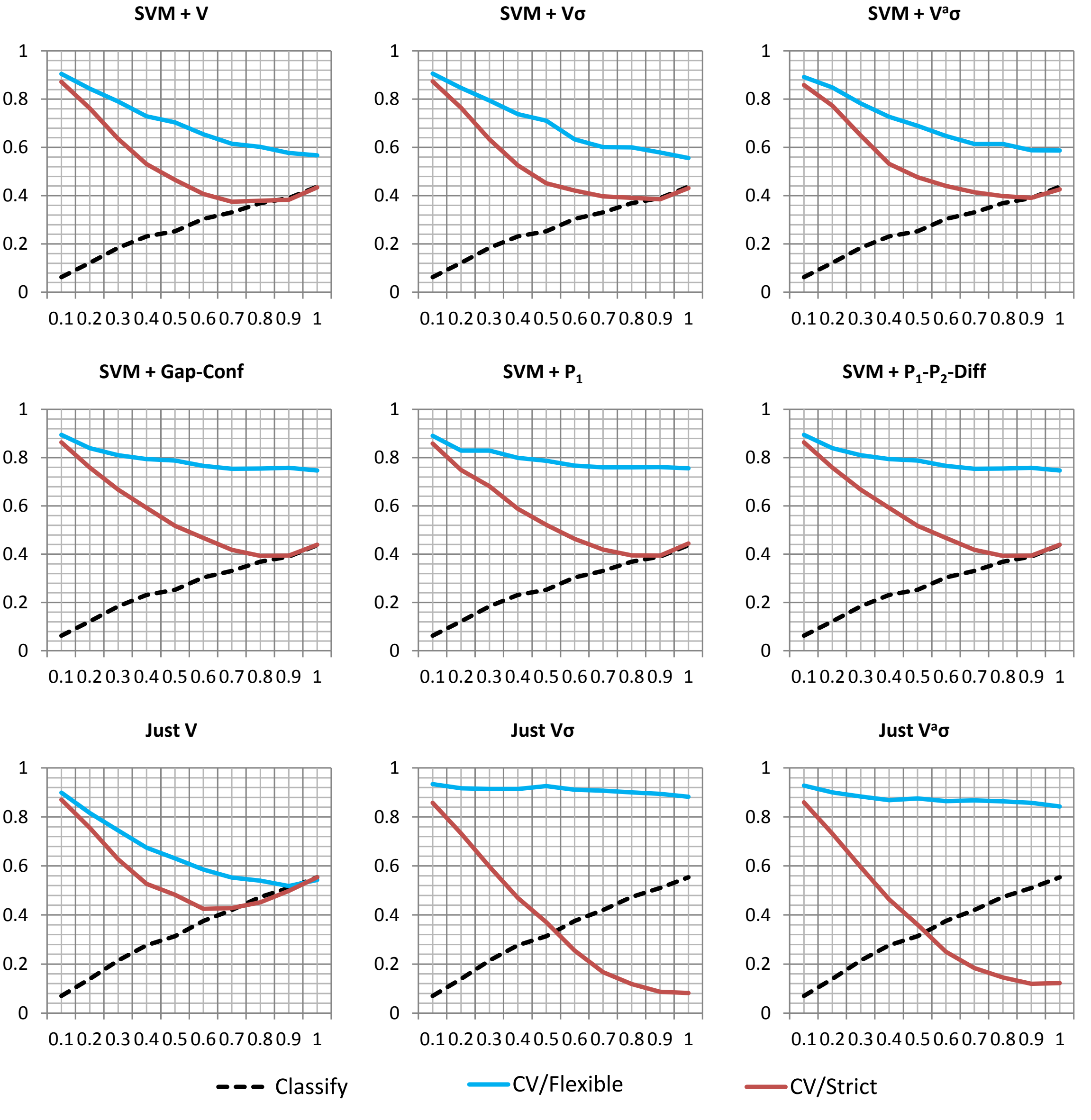


Figure B.7: *Classify-Verify* F1-scores on $BLOG_S$ using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$, using p-induced verification thresholds.

B.8 Complete *Classify-Verify* Evaluation on $BLOG_S$ with $\langle 500, 2\rangle$-*chars* Using *Robust* Thresholds

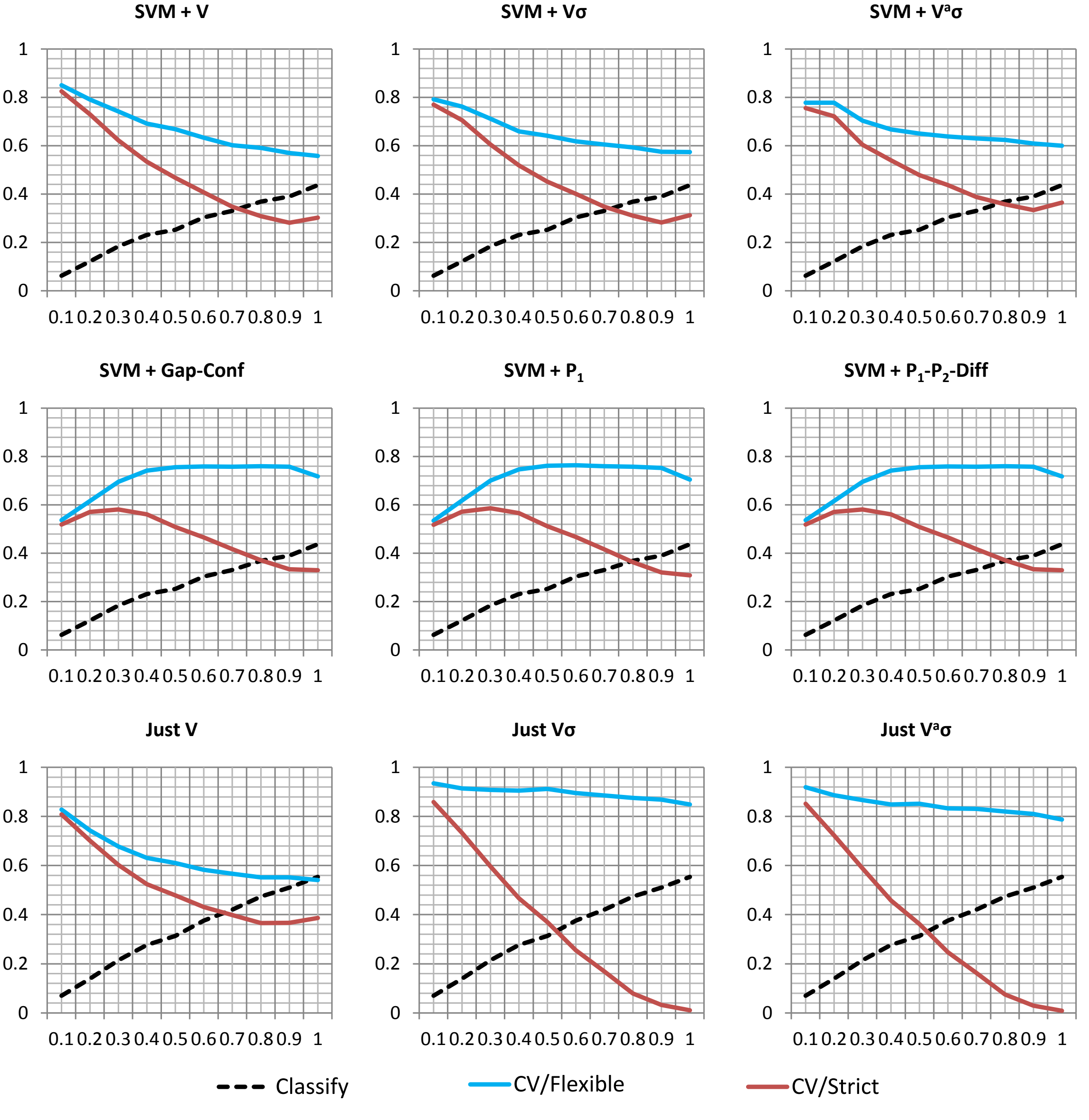


Figure B.8: *Classify-Verify* F1-scores on $BLOG_S$ using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$, using robust verification thresholds.

B.9 Complete *Classify-Verify* Evaluation on *EBG* Imitation Attack Documents with $\langle 500, 2 \rangle$-*chars*

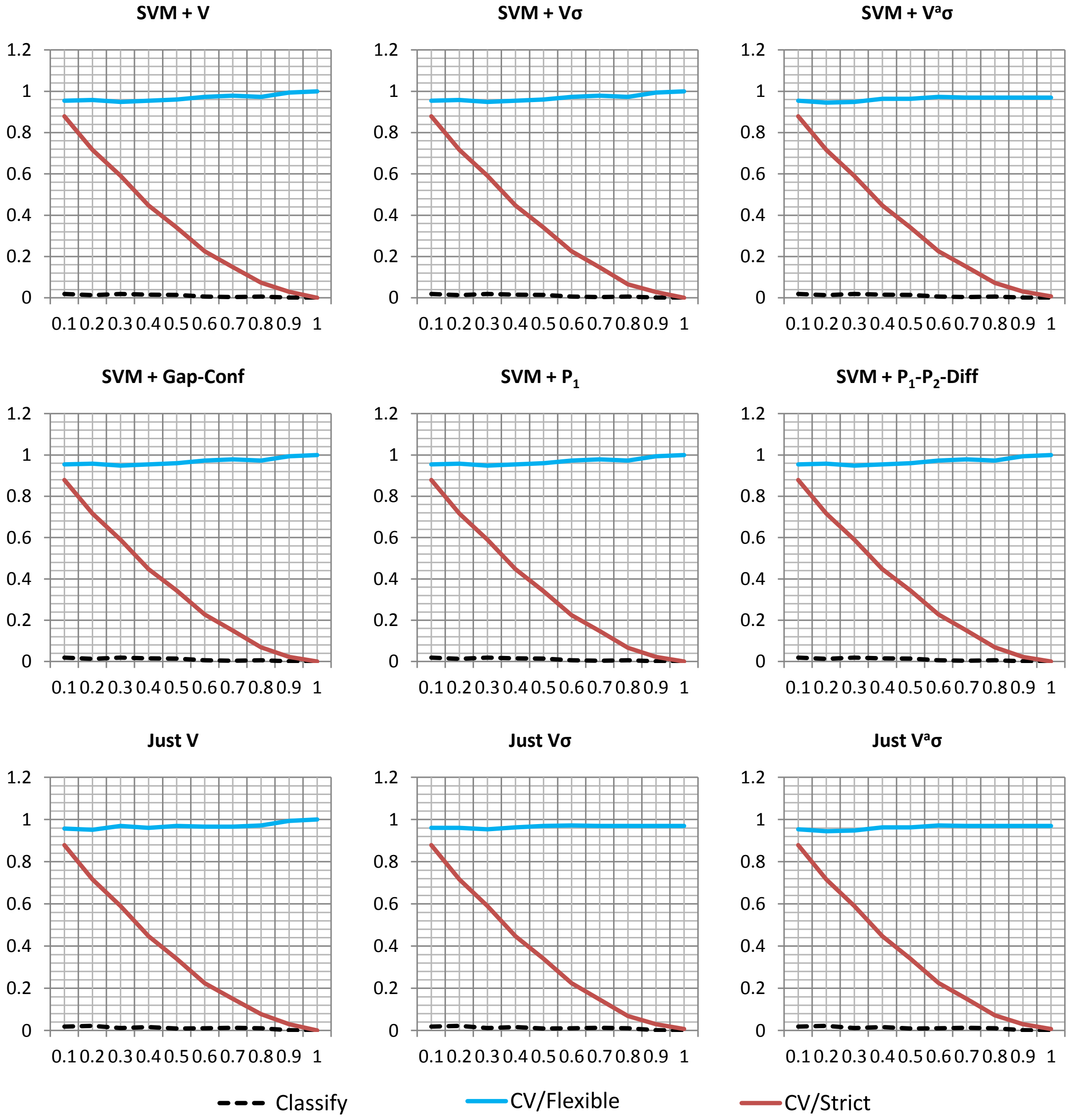


Figure B.9: *Classify-Verify* F1-scores on *EBG* imitation attack documents using the $\langle 500, 2 \rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$.

B.10 Complete *Classify-Verify* Evaluation on *EBG* Imitation Attack Documents with $\langle 500, 2\rangle$-*chars* Using Non-Attack p-Induced Thresholds

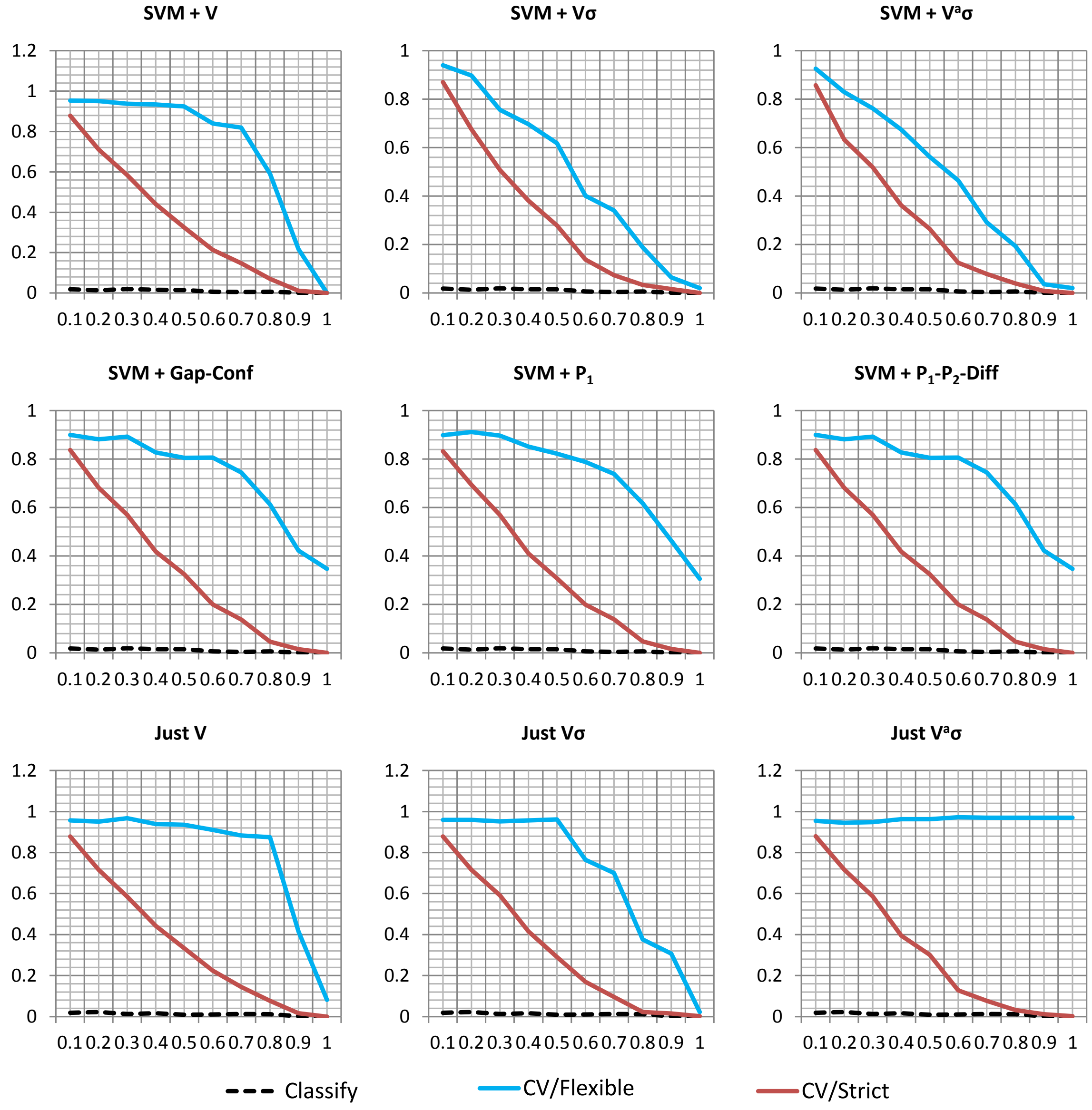
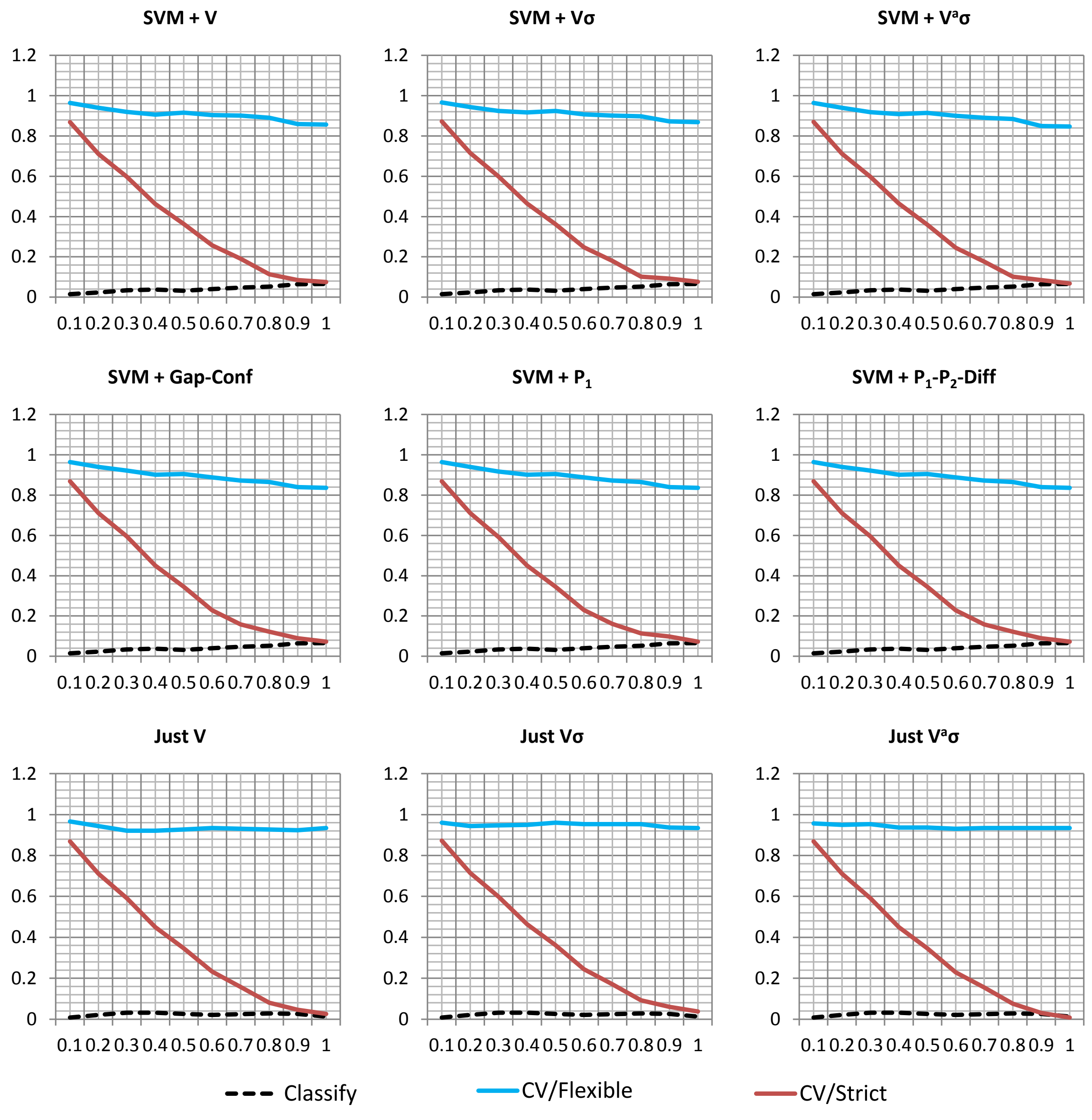


Figure B.10: *Classify-Verify* F1-scores on *EBG* imitation attack documents using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$, using p-induced verification thresholds calculated in non-attack settings.

B.11 Complete *Classify-Verify* Evaluation on *EBG* Obfuscation Attack Documents with $\langle 500, 2\rangle$-*chars*


SVM + V

SVM + Vσ

SVM + $V^a\sigma$

SVM + Gap-Conf

SVM + P_1

SVM + P_1-P_2-Diff

Just V

Just Vσ

Just $V^a\sigma$

Figure B.11: *Classify-Verify* F1-scores on *EBG* obfuscation attack documents using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$.

B.12 Complete *Classify-Verify* Evaluation on *EBG* Obfuscation Attack Documents with $\langle 500, 2\rangle$-*chars* Using Non-Attack p-Induced Thresholds

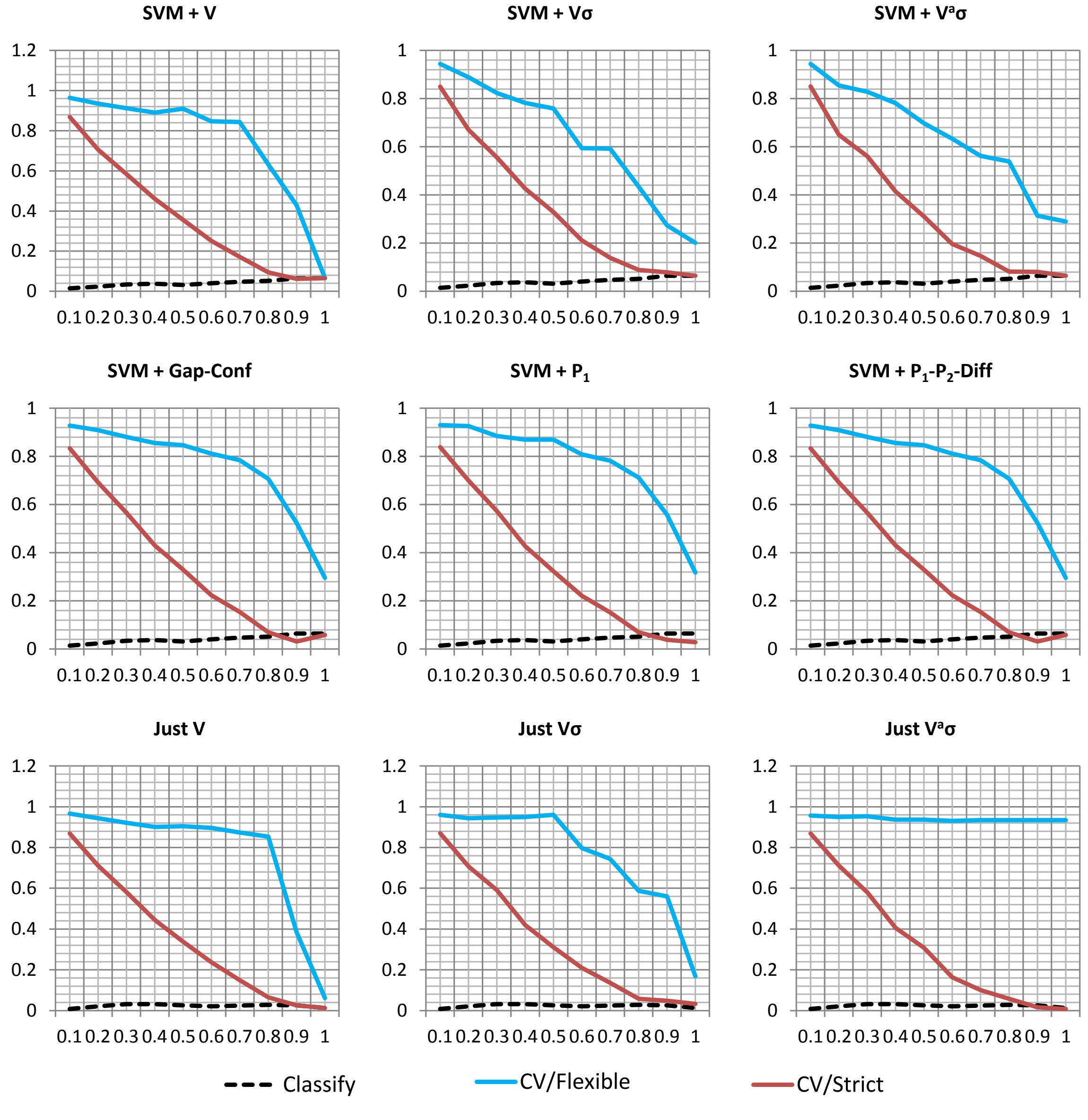


Figure B.12: *Classify-Verify* F1-scores on *EBG* obfuscation attack documents using the $\langle 500, 2\rangle$-*chars* feature set as a function of $p = 0.1, ..., 1.0$, using p-induced verification thresholds calculated in non-attack settings.

B.13 Complete *Classify-Verify* Evaluation on $BLOG_L$ with $\langle 500, 2\rangle$*-chars*

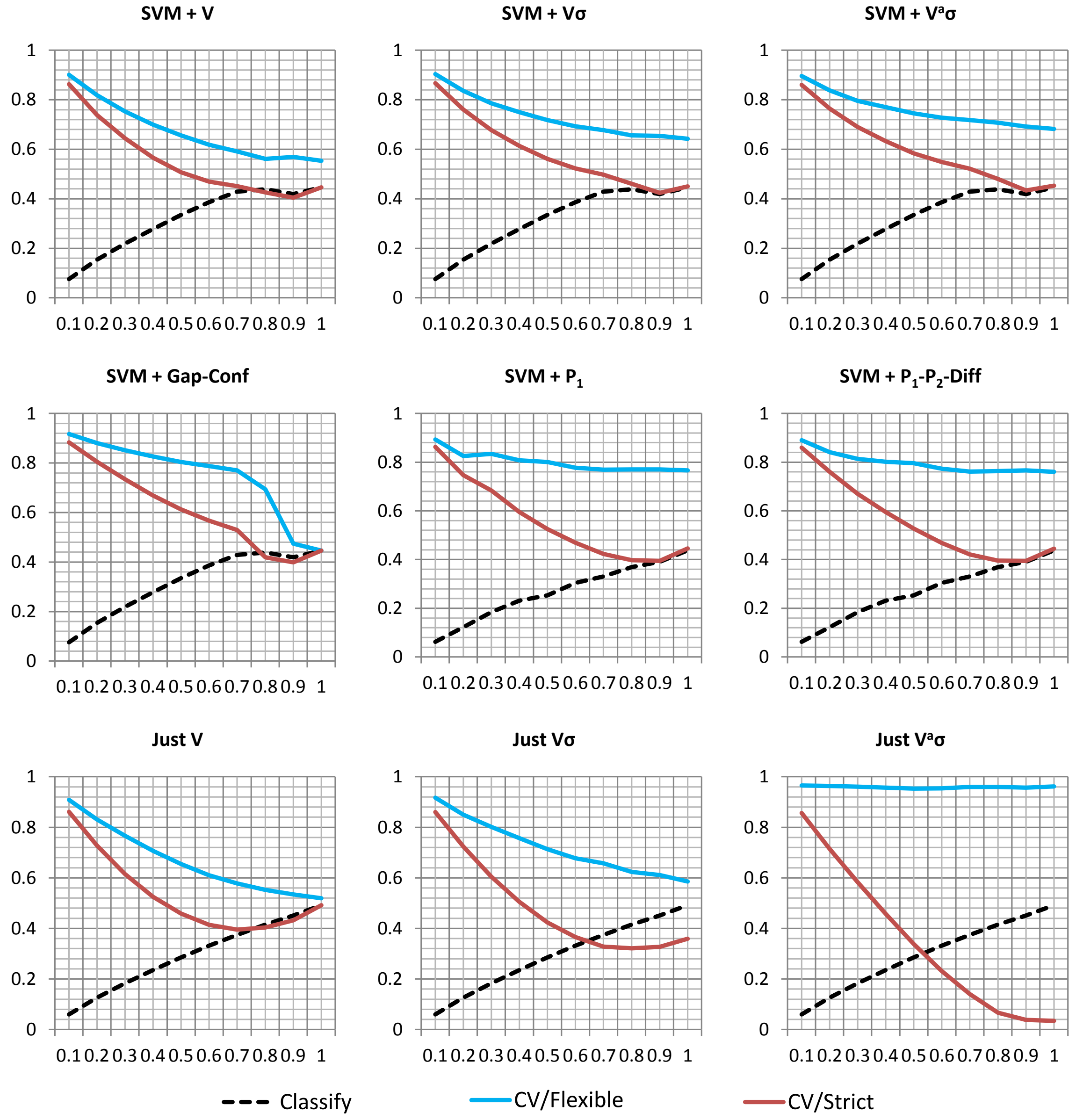


Figure B.13: *Classify-Verify* F1-scores on $BLOG_L$ using the $\langle 500, 2\rangle$*-chars* feature set as a function of $p = 0.1, ..., 1.0$.

B.14 Complete *Classify-Verify* Evaluation on *AAUTH* with ⟨500, 2⟩-*chars*

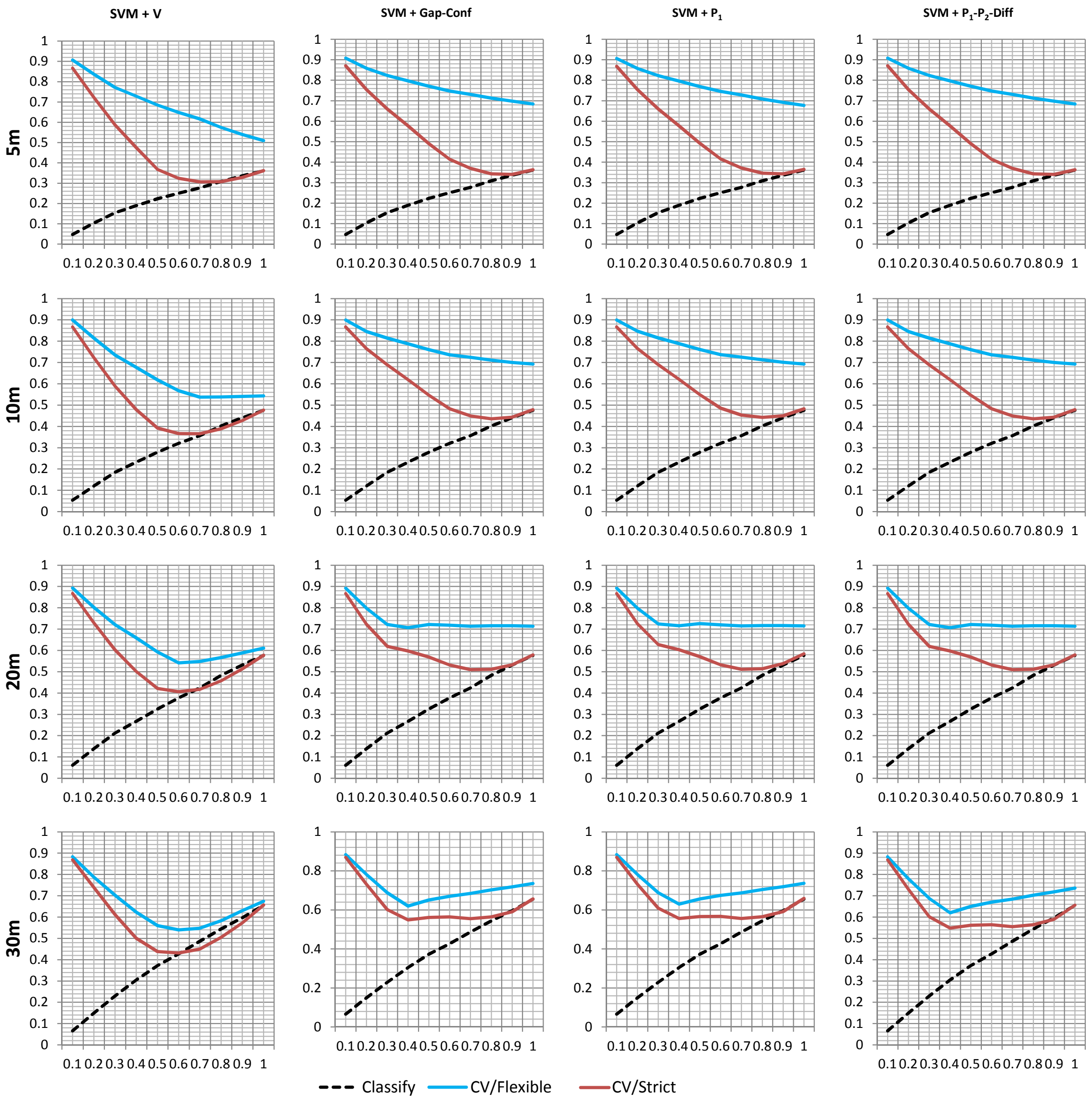


Figure B.14: *Classify-Verify* F1-scores on *AAUTH* using the ⟨500, 2⟩-*chars* feature set as a function of $p = 0.1, ..., 1.0$, for user input sliding windows of size 5, 10, 20 and 30 minutes with 1 minute overlap.

B.15 Complete F1-Scores for All Figures Illustrated in Sec. 6.3

Table B.1: Complete F1-scores for *Classify-Verify* applied on *EBG*, for the figures illustrated in Sec. 6.3.1 and Sec. 6.3.2.

EBG										
p	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
SVM + V_σ										
Classify	0.074	0.182	0.272	0.392	0.471	0.569	0.644	0.731	0.805	0.889
CV/*flexible*	0.911	0.847	0.804	0.768	0.757	0.757	0.755	0.781	0.818	0.889
CV/*strict*	0.907	0.835	0.783	0.745	0.729	0.728	0.727	0.757	0.798	0.889
SVM + P_1										
Classify	0.074	0.182	0.272	0.392	0.471	0.569	0.644	0.731	0.805	0.889
CV/*flexible*	0.908	0.905	0.902	0.883	0.873	0.876	0.876	0.877	0.886	0.926
CV/*strict*	0.905	0.890	0.876	0.849	0.829	0.825	0.818	0.816	0.829	0.898
***p*-induced thresholds, SVM +** V_σ										
Classify	0.074	0.182	0.272	0.392	0.471	0.569	0.644	0.731	0.805	0.889
CV/*flexible*	0.908	0.842	0.803	0.768	0.752	0.751	0.750	0.773	0.805	0.875
CV/*strict*	0.904	0.829	0.782	0.743	0.723	0.721	0.728	0.757	0.795	0.885
***p*-induced thresholds, SVM +** P_1										
Classify	0.074	0.182	0.272	0.392	0.471	0.569	0.644	0.731	0.805	0.889
CV/*flexible*	0.906	0.893	0.892	0.877	0.868	0.873	0.871	0.873	0.884	0.926
CV/*strict*	0.902	0.877	0.867	0.846	0.827	0.823	0.812	0.816	0.825	0.894
Robust thresholds, SVM + V_σ										
Classify	0.074	0.182	0.272	0.392	0.471	0.569	0.644	0.731	0.805	0.889
CV/*flexible*	0.844	0.785	0.767	0.751	0.752	0.754	0.749	0.767	0.781	0.811
CV/*strict*	0.842	0.776	0.750	0.731	0.725	0.722	0.717	0.734	0.753	0.807
Robust thresholds, SVM + P_1										
Classify	0.074	0.182	0.272	0.392	0.471	0.569	0.644	0.731	0.805	0.889
CV/*flexible*	0.665	0.802	0.856	0.873	0.872	0.870	0.856	0.829	0.811	0.799
CV/*strict*	0.662	0.791	0.834	0.839	0.828	0.818	0.796	0.766	0.752	0.774

Table B.2: Complete F1-scores for *Classify-Verify* applied on $BLOG_S$, for the figures illustrated in Sec. 6.3.1 and Sec. 6.3.2.

$\boldsymbol{BLOG_S}$										
p	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
SVM + V_σ^a										
Classify	0.062	0.122	0.184	0.231	0.252	0.303	0.330	0.368	0.390	0.437
CV/*flexible*	0.894	0.848	0.784	0.725	0.698	0.663	0.639	0.626	0.607	0.604
CV/*strict*	0.863	0.774	0.648	0.542	0.475	0.438	0.415	0.403	0.396	0.439
SVM + P_1										
Classify	0.062	0.122	0.184	0.231	0.252	0.303	0.330	0.368	0.390	0.437
CV/*flexible*	0.893	0.824	0.834	0.807	0.800	0.777	0.769	0.770	0.770	0.765
CV/*strict*	0.862	0.746	0.683	0.595	0.525	0.469	0.422	0.397	0.394	0.445
***p*-induced thresholds, SVM +** V_σ^a										
Classify	0.062	0.122	0.184	0.231	0.252	0.303	0.330	0.368	0.390	0.437
CV/*flexible*	0.891	0.848	0.782	0.726	0.689	0.647	0.614	0.614	0.587	0.586
CV/*strict*	0.858	0.773	0.649	0.531	0.476	0.440	0.414	0.398	0.391	0.426
***p*-induced thresholds, SVM +** P_1										
Classify	0.062	0.122	0.184	0.231	0.252	0.303	0.330	0.368	0.390	0.437
CV/*flexible*	0.890	0.829	0.829	0.798	0.786	0.767	0.760	0.760	0.760	0.756
CV/*strict*	0.858	0.749	0.681	0.588	0.521	0.463	0.419	0.395	0.394	0.444
Robust thresholds, SVM + V_σ^a										
Classify	0.062	0.122	0.184	0.231	0.252	0.303	0.330	0.368	0.390	0.437
CV/*flexible*	0.778	0.777	0.703	0.667	0.650	0.638	0.630	0.624	0.609	0.600
CV/*strict*	0.756	0.721	0.604	0.539	0.478	0.437	0.388	0.358	0.333	0.365
Robust thresholds, SVM + P_1										
Classify	0.062	0.122	0.184	0.231	0.252	0.303	0.330	0.368	0.390	0.437
CV/*flexible*	0.534	0.618	0.700	0.747	0.762	0.763	0.760	0.758	0.752	0.704
CV/*strict*	0.517	0.572	0.585	0.565	0.511	0.467	0.416	0.363	0.320	0.308

Table B.3: Complete F1-scores for *Classify-Verify* applied on *EBG* in adversarial settings, for the figures illustrated in Sec. 6.3.3.

***EBG* in Adversarial Settings**										
p	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Obfuscation: oracle thresholds on attack data, SVM + V_σ										
Classify	0.014	0.022	0.034	0.037	0.031	0.039	0.047	0.051	0.064	0.064
CV/*flexible*	0.966	0.943	0.924	0.917	0.924	0.907	0.901	0.897	0.872	0.868
CV/*strict*	0.871	0.715	0.598	0.464	0.362	0.248	0.180	0.101	0.091	0.075
Obfuscation: p-induced thresholds from non-attack data, SVM + P_1										
Classify	0.014	0.022	0.034	0.037	0.031	0.039	0.047	0.051	0.064	0.064
CV/*flexible*	0.930	0.925	0.884	0.869	0.870	0.808	0.781	0.712	0.558	0.317
CV/*strict*	0.838	0.699	0.573	0.427	0.322	0.221	0.151	0.069	0.038	0.028
Imitation: oracle thresholds on attack data, SVM + P_1										
Classify	0.018	0.013	0.019	0.015	0.014	0.006	0.004	0.006	0.001	0
CV/*flexible*	0.954	0.957	0.948	0.954	0.960	0.972	0.978	0.972	0.993	1
CV/*strict*	0.879	0.716	0.589	0.447	0.340	0.224	0.147	0.066	0.021	0
Imitation: p-induced thresholds from non-attack data, SVM + P_1										
Classify	0.018	0.013	0.019	0.015	0.014	0.006	0.004	0.006	0.001	0
CV/*flexible*	0.899	0.911	0.896	0.851	0.822	0.787	0.738	0.616	0.463	0.306
CV/*strict*	0.832	0.694	0.568	0.410	0.307	0.198	0.138	0.047	0.016	0

Table B.4: Complete F1-scores for *Classify-Verify* applied on $BLOG_L$, for the figures illustrated in Sec. 6.3.4.

$BLOG_L$										
p	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
SVM + V_σ^a										
Classify	0.076	0.154	0.219	0.277	0.335	0.386	0.428	0.438	0.419	0.446
CV/*flexible*	0.895	0.837	0.794	0.770	0.745	0.728	0.718	0.707	0.691	0.682
CV/*strict*	0.860	0.763	0.689	0.632	0.583	0.548	0.521	0.479	0.433	0.453
SVM + P_1										
Classify	0.076	0.154	0.219	0.277	0.335	0.386	0.428	0.438	0.419	0.446
CV/*flexible*	0.926	0.894	0.868	0.845	0.824	0.805	0.777	0.677	0.536	0.446
CV/*strict*	0.892	0.817	0.748	0.683	0.625	0.575	0.532	0.436	0.398	0.446

Table B.5: Complete F1-scores for *Classify-Verify* applied on $AAUTH$, for the figures illustrated in Sec. 6.3.5.

AAUTH										
p	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
30m, SVM + P_1										
Classify	0.065	0.149	0.229	0.304	0.372	0.426	0.486	0.543	0.596	0.655
CV/*flexible*	0.883	0.780	0.688	0.629	0.655	0.674	0.687	0.704	0.719	0.736
CV/*strict*	0.868	0.730	0.608	0.556	0.565	0.566	0.555	0.564	0.591	0.659
30m, SVM + V										
Classify	0.065	0.149	0.229	0.304	0.372	0.426	0.486	0.543	0.596	0.655
CV/*flexible*	0.883	0.786	0.702	0.623	0.560	0.539	0.547	0.582	0.629	0.674
CV/*strict*	0.868	0.738	0.611	0.500	0.438	0.431	0.449	0.504	0.572	0.656
20m, SVM + P_1										
Classify	0.060	0.138	0.212	0.267	0.324	0.377	0.424	0.483	0.531	0.576
CV/*flexible*	0.891	0.797	0.724	0.715	0.726	0.719	0.714	0.716	0.717	0.714
CV/*strict*	0.867	0.724	0.627	0.602	0.570	0.532	0.511	0.514	0.538	0.584
20m, SVM + V										
Classify	0.060	0.138	0.212	0.267	0.324	0.377	0.424	0.483	0.531	0.576
CV/*flexible*	0.892	0.802	0.720	0.659	0.592	0.541	0.548	0.566	0.588	0.610
CV/*strict*	0.868	0.729	0.602	0.501	0.421	0.406	0.417	0.457	0.513	0.577
10m, SVM + P_1										
Classify	0.053	0.120	0.184	0.231	0.279	0.320	0.356	0.402	0.440	0.475
CV/*flexible*	0.899	0.846	0.815	0.788	0.760	0.736	0.725	0.712	0.701	0.692
CV/*strict*	0.867	0.766	0.690	0.620	0.548	0.485	0.453	0.441	0.449	0.482
10m, SVM + V										
Classify	0.053	0.120	0.184	0.231	0.279	0.320	0.356	0.402	0.440	0.475
CV/*flexible*	0.899	0.814	0.735	0.677	0.618	0.567	0.537	0.538	0.540	0.543
CV/*strict*	0.867	0.722	0.589	0.479	0.391	0.366	0.365	0.388	0.426	0.476
5m, SVM + P_1										
Classify	0.047	0.103	0.153	0.189	0.223	0.251	0.277	0.308	0.336	0.361
CV/*flexible*	0.906	0.858	0.822	0.796	0.769	0.745	0.727	0.708	0.691	0.677
CV/*strict*	0.867	0.754	0.660	0.577	0.492	0.416	0.371	0.346	0.343	0.364
5m, SVM + V										
Classify	0.047	0.103	0.153	0.189	0.223	0.251	0.277	0.308	0.336	0.361
CV/*flexible*	0.906	0.836	0.771	0.729	0.684	0.648	0.617	0.575	0.539	0.509
CV/*strict*	0.867	0.722	0.586	0.475	0.367	0.324	0.306	0.307	0.328	0.362