Secure Obfuscation of Authoring Style

Hoi Le(✉), Reihaneh Safavi-Naini, and Asadullah Galib

Department of Computer Science, University of Calgary, Calgary, Canada
{leh,rei}@ucalgary.ca, asadullah.al.galib@hotmail.com

Abstract. Anonymous authoring includes writing reviews, comments and blogs, using pseudonyms with the general assumption that using these pseudonyms will protect the real identity of authors and allows them to freely express their views. It has been shown, however, that writing style may be used to trace authors across multiple Websites. This is a serious threat to privacy and may even result in revealing the authors's identities. In obfuscating authors' writing style, an authored document is modified to hide the writing characteristics of the author. In this paper we first show that existing obfuscation systems are insecure and propose a general approach for constructing obfuscation algorithms, and then instantiate the framework to give an algorithm that semi-automatically modifies an author's document. We provide a secure obfuscation scheme that is able to hide an author's document *securely* among other authors' documents in a corpus. As part of our obfuscation algorithm we present a new algorithm for identifying an author's unique words that would be of independent interest.

We present a security model and use it to analyze our scheme and also the previous schemes. We implement our scheme and give its performances through experiments. We show that our algorithm can be used to obfuscate documents securely and effectively.

Keywords: Obfuscation · Stylometry · Privacy · Anonymity

1 Introduction

Creating accounts on Websites under pseudonyms and using them to write reviews or post comments is a common practice, with the general belief that authors remain anonymous and can freely express their opinions and views. Although major review Websites do not allow mass collection of data by outsiders, it is possible to collect substantial number of reviews and posts from Websites such as IMDB and Netflix and so a natural question is whether authors can be traced across websites that they have posted their blogs, reviews and comments.

Authorship attribution techniques are based on the observation that people write in their own individual styles. Authorship attribution techniques have made

A. Galib—This work is supported in part by Alberta Innovates Technology Futures, in the Province of Alberta, Canada.

R.N. Akram and S. Jajodia (Eds.): WISTP 2015, LNCS 9311, pp. 88–103, 2015.
DOI: 10.1007/978-3-319-24018-3_6

significant progress. Today, with sufficient amount of data it is possible to identify an author among a large number (e.g. 100,000) of authors [1]. An author's writing style is referred to as the *writeprint* and can be extracted as a feature set from the documents written by them, and used to identify their anonymously written documents with accuracy above 90% [2]. Examples of writeprint features are syntactic features (such as part-of-speech tags, function words) and lexical features (e.g., word frequencies, word n-grams). Using writing style to link an author on multiple web sites was proposed in [3,4]. It was shown that this can pose a real threat to the user's anonymity and allow adversaries to learn more about a user than they intended to reveal, including access to their private information such as photos, places that they live and work. In some cases, if the information can be linked to websites such as forums of universities that include the users' real names, the user identity will be revealed. In [3], Narayannan et al. showed that attackers who know a small amount of information about a Netflix subscriber can identify the subscriber in the dataset. Similarly through a linkability study of Yelp reviews [4] authors showed that using letter distribution of alphabet, up to 83% of anonymous reviews can be linked to their authors. To summarize, analysis of writing style allows one to breach users' privacy by tracing their activities across the Internet. This is particularly concerning because users are unaware of the fact and could inadvertently reveal sensitive information.

This problem can be alleviated if authors' writing style is obfuscated. A direct approach is to imitate another author's style by analysing their style using a set of documents on the same topic, learning the style characteristics and modifying one's own style to match those characteristics. This however would be a tedious process that needs sufficient automation and computer support to become acceptable. To support users in hiding their writing styles, a number of approaches have been proposed. Unfortunately, these approaches [5,6] are vulnerable to attacks that allow the adversary to narrow down the number of users and in some cases recover the original author. We review previous works and present our attacks on these works in Section 2.

We also propose a new approach to obfuscation of writing style and give details of an instance of the approach, its security analysis and experimental results supporting feasibility and practicality of the approach.

Our Contributions: Our contributions can be summarized as follows:

– We present attacks on the schemes [5,6] and show the attacks substantially reduce the claimed security and in some cases completely reverse the obfuscation. The attacks exploit the deterministic nature of the algorithms and are successful in revealing the original author of an obfuscated document. In the case of Anonymouth (an instantiation of [6]) which is not completely deterministic, using the data set and experiments that are reported in the paper, we can identify a set of size 2 that includes the author, with probability $\frac{14}{32} = 0.438$. In this experiment, in total 10 authors were considered and so random guessing of a set of size 2 that includes the author would have the success chance of $\frac{9}{\binom{10}{2}} = 0.2$. Hence, our attack doubles the success chance of the attacker compared to this random guess.

– We propose a general approach for designing stylometry obfuscation schemes and give an instantiation that provides a secure obfuscation system. In this approach, a document is represented by a set of *features*. One can obfuscate a document with respect to a subset of features as described below. Consider a corpus of documents of N authors and assume the corpus is used to determine a set of features for both a user that is represented by a set of documents, and a single document. A feature f_i^D in a document D has value w_i^D, and is also referred to as a *feature point* (e.g. frequency of occurrences, a measure of the uniqueness of the feature, etc.). To obfuscate a document D of the user U with respect to a subset of features $\mathcal{F}^D$, the following steps are applied to elements of $\mathcal{F}^D$ in sequence. For a feature $f_i^D \in \mathcal{F}^D$, a feature point $w_i^{u'}$ of another author U' is selected and w_i^D is modified so that it becomes "close" to $w_i^{u'}$. Here "close" means the distance between w_i^D and $w_i^{u'}$ is made small under a distance measure. The resulting document will be used as the input of the same process for the next feature in the sequence.

In our instantiation of this approach we will use Basic-9 as the feature set. A feature point is a non-negative real number representing one of 9 characteristics of a user's writings. Moving "close" to a feature point means a user modifies their document such that the corresponding feature in that document becomes close to that target feature point.

We also present a new unique word identification algorithm using information theoretic measures and use it as an identifying feature for users during obfuscation. This algorithm may have other applications and would be of independent interest.

– We present a security model for stylometry obfuscation algorithms and will use it to analyse other schemes as well as our scheme. The model provides a framework for the evaluation of style obfuscation systems. In this model, we describe attackers' capabilities, discuss possible attack strategies and define the success of their attacks.

The rest of the paper is organized as follows. Section 2 summarizes the related works and we present attacks to the security flaws of those previous works. In Section 3 and 4 we introduce our approach and implementation. We provide a security analysis in Section 5. Section 6 presents our experimental results. We conclude the paper in Section 7.

2 Related Work

Rao and Rohatgi suggested round trip machine translation (for example, English - German - English) as a possible method for document anonymization [7]. However, with the improvement in machine translation, it has been shown empirically that round trip machine translation is not effective in obfuscating writing style. There are also proposals to allow users to obfuscate their writing styles in an automated or semi-automated way. Kacmarcik et. al [5] used word frequencies in one's writings as the writing style. WinMine is a tool that uses Decision Tree algorithm and is used as the core of their work. Each author's writing is represented by a set of features

(a feature is a word with a frequency attached to it). Inputs of the algorithm are feature sets of K authors, and the output is the root of the Decision Tree which is the most discriminating feature between these authors. This root is removed and included in the *most important feature set*. The procedure is repeated with the rest of the features from the authors to get the next most discriminating one until the set of the most important features is completed. The algorithm also provides a threshold for each feature that helps distinguish the authors. For example, if the feature f_i weight is less than 0.034, it belongs to author U_1, if its weight is greater than 0.074, it belongs to author U_2 (suppose K=2). The system will suggest to an author how to adjust their features so that the weights of the features are close to the corresponding weights of the farthest authors of those features. In their implementation of the approach they considered $K = 2$, and in experiments changes were only made to features in order to evaluate the obfuscation results (no actual changes were made in the document).

McDonald et al. [6] used Basic 9 feature set in their work (a brief description of this feature set is presented in Section 4). Their system helps the user to semi-automatically anonymize their document using the following approach. Consider an author U who wants to obfuscate their writing style in a document D. Suppose there exists a set of sample documents from other authors as well as a sample set from U. Features extracted from all authors are clustered (for each feature type separately) such that each cluster has at least K features. For each cluster, the following weight is then measured: $W = num\_elements \times (centroid - f_i^u)$, where $num\_elements$ is the number of elements in the cluster, $centroid$ is the centroid of that cluster, and f_i^u is the feature weight of the corresponding feature extracted from sample documents of U. The cluster that has the greatest weight will be selected, and author U should adjust their corresponding feature in D to be close to that cluster's centroid.

As we will describe in Section 2.1, the main drawback of the above two algorithms is that the obfuscated documents could be linked to the original authors.

There are other works [8,9] on automatic replacement and style transformation. However, these works either provide a general approach or are not acceptable in practice. In [8], it is suggested to transform the writing style of a document incrementally using a loop, where in each run of the loop, the style is slightly changed and this is repeated until some target condition is satisfied. An example of this latter category is [9] where all the words in a document are automatically replaced with their synonyms. This may decrease the readability, and could significantly affect the semantic of the document.

2.1 Attacks on the Existing Text Obfuscation Approaches

The algorithm in [5] is deterministic, and the steps of the algorithm can be perfectly reversed, as we present below.

Attack on the system in [5]: Consider a document D written by U which is anonymized following the algorithm described in [5], together with the initial

training set. From the initial training set, we can obtain the most important feature set $\mathcal{F}$ from the obfuscated document D_O and for each feature in $\mathcal{F}$, compare its feature point with the same feature of authors from the training set. If an author U' in the training set results in the highest distance for every feature in $\mathcal{F}$, then U' is the original author of the anonymized document D_O.

Attack on the system in [6]: Suppose the obfuscated document is D_O, and the initial training documents of other authors and U are given. We can de-obfuscate the document D_O as follows: (i) Cluster each feature point of all authors into clusters, (ii) Compare centroids of these clusters with the corresponding feature point in D_O. The cluster that has centroid match with feature point of D_O is the cluster that was used in the obfuscation process. Call these clusters which are collected from all feature types is CLS. (iii) For each candidate author c, calculate its weight W with each cluster in CLS; (iv) If a candidate author U' which has the highest W values with every cluster in CLS, U' is the original author of D_O.

Anonymouth is an implementation of the algorithm [6] in which some randomness has been used for the initial stage of the K-mean algorithm. However, the resulting clusters from the K-mean algorithm stay mostly the same. Hence, we will show that this would not be enough to protect the obfuscated document. We exploit the fact that the targets in the obfuscation process can be calculated and they are related to the features which also can be extracted from the obfuscated document. We represent this relation by the distances between the possible targets and the extracted features of the obfuscated document. To normalize these distances over different types of features, the distances are converted into percentages. An average distance over all features is then calculated for each candidate author and all candidates are sorted in increasing order of this distance. This distance for obfuscated document and its real author must be small, and so the smaller distance means that it is more likely to be the real author. Hence we consider the top 2 of this list. The result is the original author appears at these positions with probability $\frac{14}{32}$ while the random chance is 0.2. This shows that this attack can be used to narrow down the set of possible authors quite effectively.

Due to limited space, our detailed attack will be introduced in a longer version of this paper.

3 Secure Stylometry Obfuscation

3.1 Problem Description

An author U wishes to obfuscate their writing style in a document D.

We assume there is a corpus $\mathcal{C}$ that contains public documents of U and also of other authors. The total number of authors is N. Writing styles of U and other authors are represented by a set of features that can be extracted from their authored documents.

3.2 A General Approach to Secure Obfuscation

Assume that the feature set of an author U_k in $\mathcal{C}$ consists of ℓ features, denoted by $\mathcal{F}_k = \{f_1^k, f_2^k, \cdots, f_\ell^k\}$. For sufficiently long documents, a document can be seen as a collection of sub-documents and so the algorithm used for extracting a user's feature set can be used to extract a feature set for a document. This means that we can extract the same feature set $\mathcal{F}_D$ from a document D that needs to be obfuscated, $\mathcal{F}_D = \{f_1^D, f_2^D, \cdots, f_\ell^D\}$.

In order to hide a feature f_i^D among the same feature of other authors in the corpus $\mathcal{C}$, we cluster authors' features $f_i^1, f_i^2, f_i^3, \cdots, f_i^N$ into a number of clusters and randomly select one as the target cluster for hiding f_i^D. The target cluster should not contain extreme values that are abnormal such as too small or too large.

1. Input:
– $\mathcal{C}$: corpus of N authors.
– K: number of clusters
– D: the document that needs to be obfuscated.
2. Algorithm:
(a) Consider feature i of N authors in the corpus $\mathcal{C}$. Denote this set as $PublicFeatures = \{f_i^1, f_i^2, f_i^3, \cdots, f_i^N\}$
(b) Run K-means algorithm to cluster $PublicFeatures$ into K clusters.
(c) User selects a target cluster (which could be randomly).
(d) The algorithm selects a random point p in the selected cluster.
(e) Modify D to corresponding feature in p, and output a temporary document D_{tmp}. Set $D = D_{tmp}$.
3. Perform step 2 for all features in the feature set of document D. Use classifier δ (described later) to classify D_{tmp}. If D_{tmp} is classified to user U with a probability less than or equal to a random chance, output $D_O = D_{tmp}$. Otherwise, replace $D = D_{tmp}$, and repeat from step 2.

The elements in the approach are described in detail as follows.

Parameter K. K is the number of clusters for each feature and ranges from 1 to $N - 1$. Choosing K depends on the number of features that a user wants in a target cluster, specially level of privacy, and the distance that the user is willing to move their document. Users may prefer many features in a cluster, or resulting in small distances to adjust. In the worst case, K and the target cluster may be chosen randomly by the user or the program (if there are too many features in the feature set), which still guarantees that the obfuscation system is secure as we will analyse in Section 5.

Feature set. This approach in general can work for any feature set. In our implementation, we use Basic-9 feature set which is widely used in experiments on text classification and text obfuscation [6,10] as well as more feasible for users to modify them compared to other complex features.

Text classifier δ. There is a wide range of text classifiers to perform text classification. Many of these text classifiers are implemented in Weka [11] which is a common tool set used in text classification related research. The tool set includes implementations for SVM, NaiveBayes, Neural Network, Decision Tree, etc. To select an appropriate classifier with respect to the corpus $\mathcal{C}$, the classifiers should be evaluated using cross-validation method: **N**-fold cross-validation splits a data set into **N** folds and runs classification experiment **N** times, each time one fold of data is used as test set and the classifier is trained on the other $\mathbf{N}-1$ folds of data. The classification accuracy is averaged over the results of **N** runs, and the classifier that gives higher accuracy is the one should be selected as δ for the obfuscation process.

K-mean clustering algorithm. K-mean clustering algorithm starts by dividing members in a dataset into K clusters, with at least one item in each cluster. The data points are randomly assigned to the clusters resulting in clusters that have roughly the same number of data points. The distance between each data point to each cluster's mean is then measured, and the mean is defined for each problem normally as the average value of all elements in a cluster. If the data point is not closest to its own cluster, it is moved to the closest cluster. This step is repeated until there is no data point moving from one cluster to another.

The above approach is flexible and the set of features can be chosen so that the document change is acceptable by the user.

4 Our Implementation

In this section, we present an implementation of the above approach using Basic-9 feature set. This set can be divided into subsets, (i) *Sentence related features* including: Average Sentence Length, Sentence Count, (ii) *Lexical features* including: Unique Word Counts, Average Syllables per Word, Character Count, Character Count without Space, and (iii) *Readability related features* including: Gunning Fox Readability Index, Complexity, Flesch Reading Ease Score. These features are described in Table 1. In stylometry Basic-9 feature set is less powerful than Writerprints which consists of low-level features such as frequencies of 1-, 2-, 3-grams. Basic-9 feature set, however, is convenient to provide suggestions to change the document for the users following these suggestions.

4.1 Preprocessing

We implemented all Basic-9 features except Unique Word Count, using the standard definitions of these features [12]. For Unique Word Count we defined a new algorithm using the information theoretic measure of mutual information, defined as follows.

For two random variables X and Y with joint probability distribution $P(X,Y)$, the mutual information measure is defined as the reduced uncertainty of variable X when variable Y is known, or vice versa. Our Unique Word Count extractor works as follows.

Table 1. Descriptions of Basic-9 feature set.

Feature	Description
Unique Word Counts	Number of unique words
Average Sentence Length	Average number of words in a sentence
Sentence Count	Number of sentences in a document
Average Syllables per Word	Average number of syllables per a word used
Gunning Fox Readability Index	A weighted average of the number of words per sentence, and the number of long words per word: $(= 0.4[\frac{\text{words}}{\text{sentences}} + 100(\frac{\text{complex words}}{\text{words}})])$
Complexity	equivalent to lexical density of a document $(= \frac{N_{lex}}{N_{tok}})$ N_{lex} is the number of lexical words token, N_{tok} is the total number of tokens.
Character Count	Number of characters used in a document
Character Count without Space	Number of characters used in a document (without spaces)
Flesch Reading Ease Score	The readability of a document: the higher values are, the easier to read. $(= 206.83 - 1.015\frac{\text{total words}}{\text{total sentence}} - 84.6\frac{\text{total syllables}}{\text{total words}})$

Unique Word Extractor. To extract a list of unique word for an author U from a corpus $\mathcal{C}$ we will do the following. Let W_U denote the list of all the words that U used in the documents in the corpus $\mathcal{C}$.

The importance of a word $w_i \in W_U$ to U is modelled by the mutual information between two random variables X_U that represents the presence of U in the corpus, and X_{w_i} that represents the presence of the word w_i in the corpus. The mutual information is calculated as,

$$I(w_i, U) = I(X_U, X_{w_i}) = H(X_{w_i}) - H(X_{w_i}|X_U)$$

Here $p(X_{w_i} = 1)$ is the probability that w_i appears in a document in $\mathcal{C}$, $p(X_U = 1)$ is the probability that U is the author of a document in $\mathcal{C}$, and $p(X_{w_i} = 1, X_U = 1)$ is the probability that w_i appear in a document written by U in $\mathcal{C}$. These probabilities are calculated as relative frequencies,

$$p(X_{w_i} = 1, X_U = 1) = \frac{n_{w_i \wedge U}}{n}$$
$$p(X_{w_i} = 1) = \frac{n_{w_i}}{n}$$
$$p(X_U = 1) = \frac{n_U}{n}$$

where n is the number of documents in the corpus, $n_{w_i \wedge U}$ is number of documents in $\mathcal{C}$ that are written by U which contain w_i, and n_{w_i} is number of documents in $\mathcal{C}$ contain w_i, and n_U is number of documents written by U.

Mutual information of words in W_U with U are ranked, and the top ℓ words are selected to the most important or "unique" word set.

The advantage of extracting "unique words" as above instead of the standard approaches, as in [13], is that the resulting extracted words are more representing for one's writing. Using the standard definition, words which appear once in a particular context would be selected as unique words, thus, do not necessarily represent one's writings.

4.2 Obfuscation Algorithm

Algorithm 1 is our instantiation of the approach described in Section 3.2.

The user must select the number of clusters $K_i (i = 1, 2, ..., 9)$, and then after applying the clustering algorithm, select one of the resulting clusters (which could be done randomly). Each K_i in the K set ranges from 1 to $N - 1$, where N is the total number of authors in the corpus $\mathcal{C}$.

Among a number of choices for the classifiers in the Weka set, by running 10-fold cross-validation analysis over the training corpus, in Section 6, SVM is selected as the best classifier for our scheme.

5 Security Analysis

We present an attacker model $\mathcal{A}$ for a text obfuscation system and use it to evaluate our system. Let D be a document that is to be obfuscated using an algorithm Π, with respect to a corpus $\mathcal{C}$, a classification algorithm δ and a feature set $\mathcal{F} = \{f_1, f_2, \cdots, f_\ell\}$. The attacker can be modelled as follows.

Attacker capabilities:
– Attacker knows the obfuscation algorithm Π;
– Attacker knows the corpus of training documents from all authors $\mathcal{C}$;
– Attacker can extract the same feature set $\mathcal{F}$ for an author or a document as in the obfuscation algorithm Π;
– Attacker use the same text classifier δ as used in the obfuscation algorithm Π.

The attackers will use the following attack strategies:

1. Backtracking.

In backtracking, the adversary takes the steps of the obfuscation algorithm in the reverse order, starting from the obfuscated document, and taking reverse steps. This adversary will have success chance of 1 in some deterministic obfuscation algorithms.

2. Exhaustive search on authors in the corpus.

In exhaustive attack, the attacker considers each author in the corpus as the candidate author. The distance for the features of each candidate and the obfuscated document can be derived (such as the distance in Section 2.1) and used to select the most likely author. For example, the author who has the closest distance is the original author.

We also consider the effect of using a different classification algorithm and show that the attacker will not have a higher chance if they use a different classification algorithm.

Suppose U is the original author of an obfuscated document D_O. Consider an attacker with capabilities as above, the attacker will try to make the candidate set $\mathcal{S}$ as small as possible. We define the success of the attack by the size of a set $\mathcal{S}$ that includes the original author. An attack is $(\Delta, \mathcal{S}, t)$ - successful if the probability that the author is in $\mathcal{S}$ is at least $1 - \Delta$ and $|\mathcal{S}| \leq t$ (with $\Delta \geq 0$ and $1 \leq t \leq N$):

$$P_t[\text{Author in } \mathcal{S}] \geq 1 - \Delta$$

Input : document D, corpus $\mathcal{C}$, and
$K = \{K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_8, K_9\}$.

Output: obfuscated document D_O

$FeatureExtractor(\mathcal{C}) \longleftarrow (\mathcal{F}_1, \mathcal{F}_2, \cdots, \mathcal{F}_N)$

/* Each $\mathcal{F}_k$ includes 9 elements of Basic-9 features:
$\mathcal{F}_k = \{f_1^k, f_2^k, f_3^k, f_4^k, f_5^k, f_6^k, f_7^k, f_8^k, f_9^k\}$ */

$FeatureExtractor(D) \longleftarrow \mathcal{F}_D$

/* $\mathcal{F}_D = \{f_1^D, f_2^D, f_3^D, f_4^D, f_5^D, f_6^D, f_7^D, f_8^D, f_9^D\}$ */

Select a classifier δ.

for *each feature i in the Basic-9 feature set* **do**

  /* Clustering $f_i^1, f_i^2, \cdots, f_i^N$ into K_i clusters */

  $SimpleKmeans(f_i^1, f_i^2, \cdots, f_i^N, K_i)$

  U selects a cluster

  /* A target t_i is chosen randomly in that cluster. */

  $t_i = RandomSelection()$

  U modifies D to adjust feature f_i^D to the target t_i.

  Output a temporary document D_{tmp}. Set $D = D_{tmp}$.

end

$\delta(D_{tmp})$ /* Reclassify D_{tmp}. */

if $\delta(D_{tmp}) \longleftarrow U_i$ **and** $U_i \neq U$ **and** $P[\delta(D_{tmp} \leftarrow U] \leq \frac{1}{N}$ **then**

  output $D_O = D_{tmp}$.

end

if $D_O \equiv \emptyset$ **then**

  Re-run the algorithm. /* Now $D = D_{tmp}$ */

end

Algorithm 1. Secure Obfuscation

The most identifying attack is when $t = 1$.

We showed that the backtracking attack was successful against deterministic algorithms to determine the author in Section 2.1.

Evaluation of Secure Obfuscation. Given the obfuscated document D_O and the public corpus $\mathcal{C}$, the attacker can extract its feature set using the same feature extractor algorithm as the obfuscator. Suppose all feature weights in the feature set match the target values generated by the obfuscation process, so $\mathcal{F}_{D_O} = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9\}$ (t_i is the target generated in the obfuscation process for feature f_i^D of the document D). According to our algorithm, these target values are selected in chosen clusters by a random algorithm $RandomSelection()$. As these clusters are chosen by the user, if this information is not leaked, the

attacker cannot determine which clusters that the values in $\mathcal{F}_{D_O}$ belong to, and only can guess these clusters with a random chance. The attackers are also unable to find the link between the user and their chosen clusters.

The exhaustive search was successful against Anonymouth in reducing the size of potential authors from the full author set to a limited set $\mathcal{S}$ with $|\mathcal{S}| = 2$ and $P_2[\text{Author in } \mathcal{S}] = \frac{14}{32}$.

In *Secure Obfuscation*, the targets in the obfuscation process are chosen randomly, hence the attacker cannot perform the same attack to our algorithm as on Anonymouth in Section 2.1. Instead, we perform a *similar* attack on the documents which are obfuscated by our scheme (Section 6). The attack generates a sorted list of the candidate authors for each obfuscated document based on the average distance measured between the obfuscated document's features and a candidate author's (instead of targets from clusters as in the attack on Anonymouth). Note that we allow the user to appear at the top two elements of the list, the success probability of the attack is $P_2[\text{Author in } \mathcal{S}] = \frac{3}{20}$ that is smaller than a random chance as $P_2[\text{Author in } \mathcal{S}] = \frac{2}{10} = 0.2$.

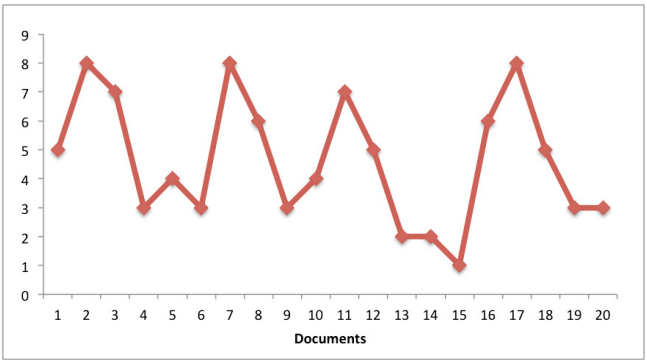

Fig. 1. The tested obfuscated documents as the x-axis, the positions of the original authors in the sorted candidate lists as the y-axis.

We note that as the backtracking strategy cannot continue after the *RandomSelection*() step. Hence combinations of backtracking with other strategies also cannot proceed after this point. In other words, backtracking strategy cannot improve the strength of other attacks.

Cross-classifier attack. Consider our obfuscation scheme which is based on a classification algorithm δ and assume we obfuscate a document D of author U to D_O such that δ classifies D_O to U'. There is a chance that δ misclassifies D_O, which means that D_O may still link to U. Therefore, if the attacker uses another classifier δ' which can link D_O to the real author, he may gain success. We analyzed the success chance of the attacker which is:

$$Succ = \frac{1}{N-1}\left(\left(1 - p_\delta\right) p_{\delta'} + p_\delta \left(1 - p_{\delta'}\right)\right) \leq \frac{1}{N-1}$$

where p_δ and $p_{\delta'}$ are the precision rates of δ and δ' respectively. This shows that, with this attack, the attacker cannot gain a success with a chance better than a random chance which is $\frac{1}{N}$.

6 Experiments

In our experiments, we demonstrate that our algorithm obfuscates documents successfully with Basic-9 feature set, and examine the effect of changing the background corpus on the obfuscation results. In our experiments, the background corpus is Brennan-Greenstadt Adversarial Stylometry Corpus [6] and is comprised of documents from 10 authors. In this corpus, there are at least 5 documents for each author, each document consisting of 500 words. These documents are written about different topics. We asked 10 participants to contribute their documents and join our experiments.

Each participant submitted at least 6 documents of at least 500 words. One document was used to obfuscate and the rest was used as training data. All the documents were in English and extracted to plaintext forms. We asked each participant to obfuscate at least 2 documents.

To select an appropriate classifier for our data set we followed the approach in Section 3.2. We ran ten-fold validation analysis on various classifiers (J48, Neural Network, Naive Bayes, etc.) in the Weka classifier set. The analysis was performed on the public corpus of authors' documents. SVM was selected as it yielded in high classification accuracy of 86%.

For each document, we output for users a list of suggested target values of features respecting to that corpus and a list of unique word. Guidelines on how to change these features in their documents to achieve these target values were suggested to users, e.g. removing or rephrasing unique words (note that we did not ask users to increase or decrease their number of unique words), reduce the sentence count by combining sentences together. We transformed number of characters to an approximate number of words, so that changing character counts related features would become changing number of words.

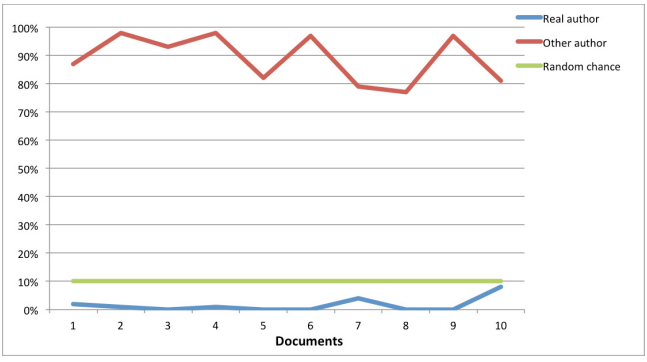

Fig. 2. Authorship attribution results of the obfuscated documents.

The temporary document which was output after changing the features was then re-classified using the SVM classification algorithm. Using above approach 19 out of 20 document were obfuscated after users change their features. On average, it took users one round of modifying features to anonymize the documents. In Figure 2 we demonstrate the authorship attribution results of the first 10 obfuscated documents. These documents were attributed to the original authors with less than or equal to a random chance which was $\frac{1}{10}$ and also were attributed to other authors with more significant chances.

6.1 Effect of Background Corpus on the Obfuscation Results

The background corpus, or set of reference authors and documents, is important for document anonymization as the algorithm calculates the target value for each feature based on the background corpus and suggests changes to users based on these target values.

We tested if documents anonymized using one background corpus are also anonymized against a different background corpus. To test this, we added 3 more authors to the existing corpus. The documents from these authors were about different topics (similar to the documents in the initial background corpus). The results (Figure 3a) showed that the effectiveness of the anonymization could change if the background corpus is changed. Although all the obfuscated documents were still anonymized however the chances that those documents were classified to the original authors might slightly increase or stay the same. This was also true for the case when we added 6 authors to the existing corpus.

We also tested the results using the whole new 10-author corpus. When we switched to the new background corpus, the chances that those obfuscated documents were attributed to the original authors all increased (Figure 3b). This was predictable as the documents were anonymized respecting to a different background corpus.

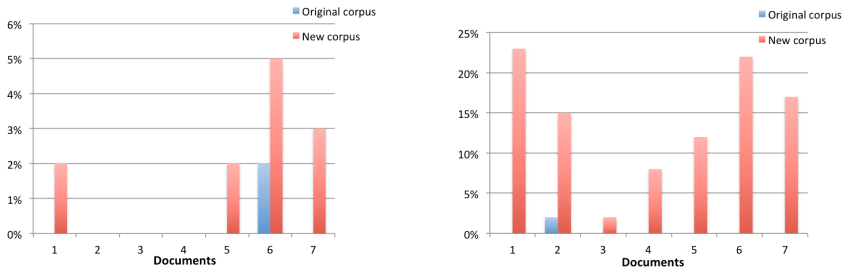

Fig. 3. Authorship attribution accuracy of the obfuscated documents in a new background corpus: (a) 3 authors were added to the original corpus, (b) new 10 authors were used.

6.2 An Example of an Obfuscated Document

A Short Paragraph Before Being Obfuscated: "This work does not make big improvement comparing to the previous works, the reasons are follows. In Obana's work, the adversarial cheating probability is defined as the overall failure probability for all the players that adversary control. We can not compare the cheating probability under the different definition of cheating success probability. In the proof of theorem 1, the authors show the cheating probability is... But I don't know how they get the probability less than... In Obana's proof of cheating probability, he got the probability from the forging probability of strong universal hash functions of strength t + 1. But in this proof, the g(X) is not proved to be the strong universal hash function, so they can not directly get the probability less than 1 q. There are some mistakes of writing: in page 8, kil+1,it+1 should be kit+1,it+1 . There are similar mistake in this page. To sum up, the authors do not define the cheating probability in the same way as Obana's work and do not explain the relation between two definitions. The improvement of sharing size is small comparing with Choudhury's work. The proof of theorem 1 is not well explained. I suggest the paper should consider the above issues."

The Obfuscated Paragraph: "This work does not make a big improvement comparing to the previous works, the reasons are follows. In the previous work, the adversarial cheating probability is defined as the overall failure probability for all the players that adversary control. We can not compare the cheating probability with the different definition of cheating success probability. In the proof of theorem 1, the authors show the cheating probability is..., and they get the probability less than... In proof the previous work about the cheating chance, he got the probability from the forging probability of strong universal hash functions of strength t + 1. But in this proof, the g(X) is not proved to be the strong universal hash function, so they can not directly get the probability less than 1 q. There are some mistakes of writing: in page 8, kil+1,it+1 should be kit+1,it+1 . There are similar mistake in this page. To sum up, the authors do not define the cheating probability in the same way as the previous work and do not explain the relation between two definitions. The improvement of sharing size is small comparing with previous work. The proof of theorem 1 is not well written. I suggest the paper should address the above issues."

Remark: In the above example, we highlighted some of the modifying applied by the author to the document. Author increased the sentence length by adding sentences together. Unique words were replaced or removed in the document. Author also adopted new words that they did not normally use in their document to change the complexity score of the document. The classification result of the obfuscated document with SVM classifier and Basic-9 feature set is presented in author 6 in Figure 2. The result shows that the probability that the obfuscated document is classified to the original author is nearly zero.

6.3 Performance Evaluation

Over 10 participants, we estimated that on average, obfuscation process requires 17 to 20 minutes depends on each user for a 500 word document. The time started from the time that the program output the suggested targets and the user knew about the task clearly, and also had a basic guide on how to change feature weights in their documents as noted above. This is reasonably efficient for document obfuscation. During our experiments, after modification process and if the document was already anonymized, we did not ask participants to change the *readability-related features* that had been affected because of the change in other features.

7 Concluding Remarks

We considered privacy of authors on anonymous websites by considering obfuscating their writing styles. We showed that the current research does not guarantee security and proposed secure obfuscation as an approach to hiding an author's identity among other authors in a corpus. We implemented our scheme using Basic-9 feature set and SVM classifier.
Our work provides a clear analysis of security for stylometry obfuscation schemes, and our algorithm can help users to obfuscate documents in practice. Refining our work to include other feature types will be an interesting direction for future research.

References

1. Narayanan, A., Paskov, H., Gong, N.Z., Bethencourt, J., Chul, E., Shin, R., Song, D.: On the feasibility of internet-scale author identification. In: Proceedings of the 33rd Conference on IEEE Sympsoium on Security and Privacy. IEEE (2012)
2. Stamatatos, E.: A survey of modern authorship attribution methods. J. Am. Soc. Inf. Sci. Technol. **60**, 538–556 (2009)
3. Narayanan, A., Shmatikov, V.: Robust de-anonymization of large sparse datasets. In: Proceedings of the 2008 IEEE Symposium on Security and Privacy, SP 2008, pp. 111–125. IEEE Computer Society, Washington, D.C. (2008)
4. Mishari, M.A., Tsudik, G.: Exploring linkablility of community reviewing. CoRR, vol. abs/1111.0338 (2011)
5. Kacmarcik, G., Gamon, M.: Obfuscating document stylometry to preserve author anonymity. In: Proceedings of the COLING/ACL on Main Conference Poster Sessions, COLING-ACL 2006, pp. 444–451. Association for Computational Linguistics, Stroudsburg (2006)
6. McDonald, A.W.E., Afroz, S., Caliskan, A., Stolerman, A., Greenstadt, R.: Use fewer instances of the letter "i": toward writing style anonymization. In: Fischer-Hübner, S., Wright, M. (eds.) PETS 2012. LNCS, vol. 7384, pp. 299–318. Springer, Heidelberg (2012)
7. Rao, J.R., Rohatgi, P.: Can pseudonymity really guarantee privacy? In: Proceedings of the 9th Conference on USENIX Security Symposium, SSYM 2000, vol 9, pp. 7–7. USENIX Association, Berkeley (2000)

8. Khosmood, F., Levinson, R.A.: Automatic natural language style classification and transformation. In: Proceedings of the 2008 BCS-IRSG Conference on Corpus Profiling, IRSG 2008, pp. 3–3. British Computer Society, Swinton (2008)
9. Khosmood, F., Levinson, R.: Automatic synonym and phrase replacement show promise for style transformation. In: Proceedings of the 2010 Ninth International Conference on Machine Learning and Applications, ICMLA 2010, pp. 958–961. IEEE Computer Society, Washington, D.C. (2010)
10. Brennan, M., Greenstadt, R., Brennan, M., Greenstadt, R.: Practical attacks against authorship recognition techniques emerging applications track
11. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. SIGKDD Explor. Newsl. **11**, 10–18 (2009)
12. Mitchell, T.M.: Machine learning. McGraw Hill, New York (1997)
13. Le Thi, H., Safavi-Naini, R.: An information theoretic framework for web inference detection. In: Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence, AISec 2012, pp. 25–36. ACM, New York (2012)