Efficient Unsupervised Authorship Clustering Using Impostor Similarity

Patrick Verga, James Allan, Brian Levine, Marc Liberatore

School of Computer Science, University of Massachusetts, Amherst
{pat, allan, brian, liberato}@cs.umass.edu

Abstract. *Some real-world authorship analysis applications require techniques that scale to thousands of documents with little or no* a priori *information about the number of candidate authors. While there is extensive research on identifying authors given a small set of candidates and ample training data, almost none is based on real-world applications of clustering documents by authorship, independent of prior knowledge of the authors. We adapt the* impostor method *of authorship verification to authorship clustering using agglomerative clustering and, for efficiency, locality sensitive hashing. We validate our methods on a publicly available blog corpus that has been used in previous authorship research. We show that our efficient method matches previous results for authorship verification on the blog corpus. We extend previous results to show that the impostor algorithm is robust to incorrectly selected impostors. On the authorship clustering task, we show that the impostor similarity method clearly outperforms other techniques on the blog corpus.*

1 Introduction

The Internet contains a massive number of documents written by unknown and anonymous authors. Determining the authorship of these documents is important in many contexts, including: commercial applications, such as fraud detection on review web sites; forensics applications to aid law enforcement; and understanding the privacy implications of public posting of written materials.

There is a large body of research on *authorship attribution*, which is the problem of discerning the author of a document given a set of a training data from a small number of candidate authors. In contrast, the related problem of *unsupervised authorship clustering*, given an open set of authors and unlabeled documents, is relatively unexplored.

In this paper, we advance solutions to the problem of clustering posts written by the same author without prior knowledge of the possible authors. We claim that agglomerative clustering is the most appropriate choice for this task as it requires no prior information or estimate of the number of clusters, except determining a threshold for stopping.

We compare several feature sets and distance metrics in our agglomeration. We then show that we can greatly increase efficiency and achieve slight gains

in accuracy by using *locality sensitive hashing* [6] as a preprocessing step. We evaluate our method on a widely used blog corpus.

Contributions. We make the following contributions.

- We demonstrate the extension of impostor similarity to authorship clustering on a widely used blog corpus. Impostor similarity has previously only been used for authorship verification. We show that impostor similarity performs better than other baseline measures as a similarity metric for agglomerative clustering. When considering same-author pairs within clusters, impostor similarity has a precision of 0.41 at recall 0.5, substantially better than the precision of 0.24 that a simple n-grams-based approach yields. (Section 5.1)
- Impostor similarity computational costs are orders of magnitude higher than simpler methods, such as n-grams. We show that by using locality sensitive hashing as a preprocessing step for agglomerative clustering, the number of comparisons can be reduced by approximately 95%, while achieving nearly identical precision and recall results (Section 5.2).
- Because the author of the documents in question will typically be unknown in a real world context, it is possible to generate mistaken impostors: impostor documents that are actually by that author. We show that the impostor technique is robust to mistaken impostors. If 5% of the impostor set are actually mistaken impostors, results are not substantially degraded (Section 5.4).

2 Related Work

Authorship analysis, the automated extraction of information about the author of a document or set of documents, can be divided into four subfields [14].

- *Authorship attribution* is the problem of discerning the author of a document given a set of a training data from a small number of candidate authors [9,26]. It is by far the most explored subproblem and has been applied to Twitter [16], web forums [22], source code [2], and blogs [20] among other applications.
- *Authorship profiling* is the problem of discerning attributes about the author, such as age, gender, or personality [24].
- *Authorship clustering* (also called unsupervised authorship analysis, authorship distinction, or similarity detection) is the grouping together of documents by the same author without any information about the authors [17,18] or number of candidate authors.
- *Authorship verification* is the problem of deciding, given two documents, whether they are written by the same author [8,15].

In this work, we focus on authorship verification and unsupervised authorship clustering. Koppel et al. [12] claim that the fundamental problem of authorship analysis is authorship verification and that all other problems can be reduced to many iterations of this binary question. One successful approach to authorship verification is the *impostor method*, introduced by Koppel et al. [15] It was

independently found to be the most effective strategy in the PAN 2013 authorship verification challenge [10, 25]. The impostor method has been applied to only the problem of authorship verification. In this paper, we extend the technique to the authorship clustering problem.

There has been very little research done on the problem of authorship clustering. Iqbal et al. [7] cluster data using k-means clustering in order to create a stylistic profile for each author. However, this technique requires an estimate of k, which may not be available. Additionally, k-means clustering is somewhat sensitive to ordering, which could further degrade its usefulness.

Layton et al. [18] applied agglomerative clustering to authorship clustering by constructing the similarity matrix from evidence accumulation clustering, applying k-means many times with different features and k values, and defining the similarity between two documents to be the number of times they are put into the same cluster. However, these experiments were done on a very small set of authors (at most 13) each represented by thousands of words of text. Also, the use of k-means as a similarity metric requires an estimate or range of estimates for k which may not be available. In contrast, we use a different similarity metric, clustering a larger set of authors represented by a very small amount of text (500 words).

Several previous studies have used locality sensitive hashing (LSH) to increase the efficiency of clustering. For example, Koga et al. [11] demonstrated the use of LSH as an approximation of single linkage in agglomerative clustering.

While most authorship analysis experiments focus on a small set of authors, Narayanan et al. [20] attempted authorship attribution on a set of 100,000 blogs. They used a normalized version of the writeprints feature set [1] with several different classifiers achieving their best results with a nearest-neighbor approach and regularized least-squares classification. However, their experiment was essentially authorship verification and constructed such that each test post had exactly one match amongst the other 99,999 blogs, which is not necessarily the case in real applications.

3 Impostor Similarity

Our method for authorship clustering depends upon a prior result, the impostor method of authorship verification. We provide relevant details in this section.

The impostor method of authorship verification was introduced by Koppel et al. [15] and was the highest performing technique at the PAN 2013 Authorship Identification competition [10]. Given two documents X and Y, the goal of authorship verification is to determine whether they are written by the same author. Rather than simply comparing X and Y, X is compared to Y as well as to *impostors*, documents similar to Y. A simple similarity measure comparison would require a uniform threshold at which the documents are determined to be by the same author. However, this threshold will inevitably change between document pairs as many factors will contribute to their similarity score, such as

Algorithm 1 : Impostor Method for Authorship Verification [15]. Given the feature representation of two documents and a set of impostors for each document (each of size K) return the Impostor Similarity score of those two documents. If the similarity is greater than some threshold, the documents are determined to be written by the same author.

procedure IMPOSTORSIMILARITY(X, Y, $Impostors_X$, $Impostors_Y$)
// X, Y are two documents as feature vectors
// $Impostors_X$, $Impostors_Y$ are sets of impostors for each document
    $similarity \leftarrow 0$
    for $k = 1 : K$ **do**
        Choose random subset of impostors and features
        if $similarity_k(X, Y) > similarity_k(X, I_Y) \forall I_Y \in Impostors_{Yk}$ **then** $similarity{+}{+}$
    for $k = 1 : K$ **do**
        Choose random subset of impostors and features
        if $similarity_k(Y, X) > similarity_k(Y, I_X) \forall I_X \in Impostors_{Xk}$ **then** $similarity{+}{+}$
    return $similarity$

genre or theme. The impostor method's indirect similarity measure is designed to help alleviate this thresholding issue.

Algorithm 1 defines the impostor similarity algorithm. It takes as input two test documents, X and Y, and two sets of potential impostors represented as feature vectors. From the set of impostors, the s most similar impostors to X and Y are chosen. It runs K comparisons between each document and impostor as a form of *unmasking* [13]: the intuition is that two documents may coincidently be similar over some set of features or averaged over a large set of features but are unlikely to be similar over many different sets of features unless they really are similar. Similarly to Koppel et al. [15], our experiments suggest that the impostor algorithm performs better when based on Min-Max similarity, defined below, than when based on cosine similarity.

$$MinMax(\boldsymbol{X}, \boldsymbol{Y}) = \frac{\sum_{i=1}^{n} min(x_i, y_i)}{\sum_{i=1}^{n} max(x_i, y_i)} \tag{1}$$

A homogeneous feature set such as *tfidf* is best suited for this technique. Several recent studies have shown that these simpler feature sets perform as well or better than more complex stylometric feature sets [5]. We evaluate several tokenizations: terms, stems, and 4-character n-grams each with simple *tfidf* weighting. We find 4-character n-grams to perform slightly better than terms or stems obtained from the Porter Stemmer [23].

As proposed in Koppel et al.'s original paper, we define a space-free 4-character n-gram as "(a) a string of characters of length 4 that includes no spaces or (b) a string of four or fewer characters surrounded by spaces." In other words, the document is first split on white-space, then tokens that are 4 characters or fewer

are added as n-grams. Larger tokens have each 4-character subsequence obtained with a sliding window added.

The impostor similarity algorithm requires several parameters: the number of iterations; the total number of impostors to choose for each document; the size of the impostor subset for each iteration; and the size of the feature subset for each iteration. In their original paper, the authors choose 100 iterations. They then select the 250 most-similar impostors for each document, and at each iteration they randomly select 25 impostors from the total set of 250 and randomly select 40% of the features. They show that their algorithm is not particularly impacted by these parameter choices.

4 Authorship Clustering with Impostor Similarity

In this section, we outline our proposed author clustering algorithm, the questions we ask, our experimental evaluations, and the data we use in our experiments.

4.1 Algorithm

Our proposed author clustering algorithm takes as input the set of documents to be clustered. It begins by initializing a similarity matrix over all documents to all zeroes. Then, the impostor similarity between each document and its K nearest neighbors (KNN) as found by locality sensitive hashing (LSH) is calculated. By considering only pairs derived from the LSH-approximate KNN, we reduce the calculation of the similarity matrix from $O(N^2)$ to $O(N(k + lsh))$ where $k \ll N$ and lsh is the runtime of LSH, which is small[1]. We use standard hierarchical agglomerative clustering (HAC) techniques [4], and join clusters using *average linkage*, where the average similarity between all of the documents in the two clusters is used to decide when to merge clusters. Additionally, by taking advantage of the sparse similarity matrix, each agglomeration step is far quicker as the most similar clusters can be calculated quickly by only considering non-zero entries.

4.2 Evaluation Criteria

We use the impostor similarity method to cluster documents by authorship. We compare impostor similarity against several baselines: *terms*, n-grams, and stems, each of which are tfidf weighted. n-grams uses the same feature vectors described in Section 3, *terms* creates features by splitting on white space and "stems" finds the Porter stem of the terms. We use min-max similarity with each of these baseline features.

Our evaluations address the following questions.

- Which similarity method best supports unsupervised authorship clustering?
- What efficiency gains are provided by incorporating locality sensitive hashing?

[1] See O'Donnell et al. [21] for a full discussion of the complexity of LSH.

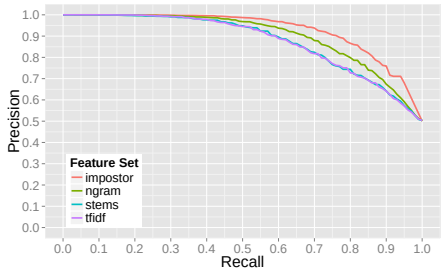

Fig. 1: We reproduce Koppel et al.'s results. Precision and recall for same author versus different author pairs for blog posts following experiment described by Koppel et al. Our baselines are higher than those originally reported because Koppel et al. used tf weighting for their baselines and tfidf for impostor; we use tfidf weighting for all experiments.

– Is there an advantage to using many similarity metrics simultaneously?
– How resilient is impostor similarity in the presence of mistaken impostors?

In general, we measure performance using precision versus recall. Specifically, our results show standard interpolated precision at a recall value r, defined as the highest precision found for any recall level $r' \geq r$.

4.3 Blog Corpus

Our experiments are performed over a well-known corpus of blog posts; which we describe in this section. We also show our reproduction of the experiments by Koppel et al. [15].

Our evaluations are based on the Blog Authorship Corpus from Schler et al. [24]. It consists of the collected posts of 19,320 bloggers gathered from blogger.com in August 2004. The corpus incorporates a total of 681,288 posts and over 140 million words; it is approximately 35 posts and 7250 words per person. Each blog contains the gender, age, industry and astrological sign (if known) of each author. For each age group there are an equal number of male and female bloggers. The age of authors range from 13–47 years old.

Each blog is made up of a variable number of posts by the author. Following the same methodology of Koppel et al. [15], we broke each blog into 500 word chunks. Their goal (and ours in Section 5.1) is to find matches between the first and last 500 words of each author's complete set of blog posts. In order to generate larger sets of posts for each author, we take randomly selected non-overlapping 500 word chunks from the middle sections of the blog, where each separate post by the author appears in no more than one chunk.

Figure 1 shows our reproduction of the experiment from Koppel et al. [15]. Here pairs of documents are constructed by choosing the first 500 words and last 500 words from an author. Then 250 same author pairs and 250 different author pairs are created by matching up first and last 500 word chunks either from the same author or two different authors. Our results for impostor similarity

are within two percentage points of theirs. (When attempting to reproduce their results, we found by analyzing the code that ran their experiments that their reported results were based upon 50 randomly selected impostors per iteration out of a set of 500 randomly selected blog posts; our reproduction is based upon these parameters.)

5 Evaluation

In this section, we describe and report on experiments designed to answer the questions posed in Section 4.2. All of the clustering experiments in this section are analyzed in three ways.

- A left graph shows the total number of *same author pairs* within all output clusters compared with the total number of same author pairs within the truth clusters.
- A middle graph shows the result of finding the best output cluster *per-author*. We define the best output cluster as the one that contains the most posts by that author.
- A right graph shows the best truth cluster *per-cluster*. Here we define the best truth cluster as the one that is most occurring in the output cluster.

The chosen output cluster's precision and recall is calculated by comparing it to its paired truth cluster. Clustering results are averaged over 200 trials.

5.1 Impostor Similarity vs Baselines

We evaluate our clustering algorithm on the blog corpus. In each trial, we select 250 authors at random from our set of 20,000 authors. Then, for each author, we randomly draw from a Zipfian distribution to determine the number of posts to cluster from that author. We created author posts by taking non-overlapping 500-word chunks from an author's blog, such that each post occurs in no more than one chunk. We then perform our author clustering algorithm, described in 4.1.

In addition to evaluating impostor similarity, we perform the same procedure using Min-Max distance with varied feature sets: *(i)* terms; *(ii)* stems; and *(iii)* 4-character n-grams. Each of these features is tfidf weighted. Min-Max outperformed cosine distance in our studies and all results shown use Min-Max. We also experimented with several writeprints variations using JStylo stylometry library [19], including the normalized writeprints feature set used in [20], but it performed significantly worse than other methods.

As Figure 2 demonstrates, the impostor similarity metric outperforms baseline measures as the basis for agglomerative clustering on the blog corpus. In this figure and following figures, plots end when confidence intervals on precision grow too large.

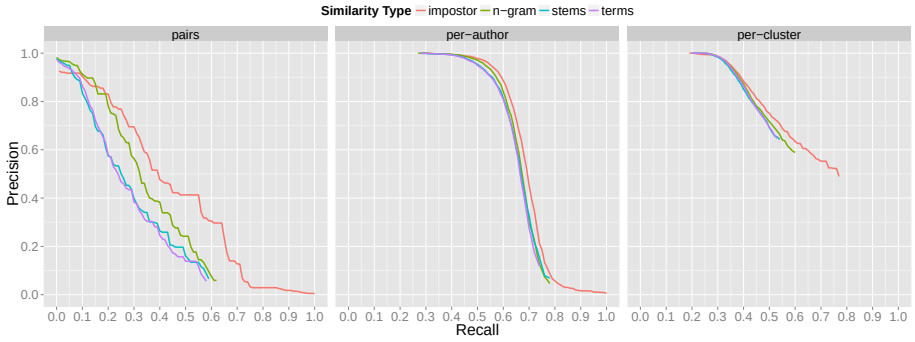

Fig. 2: Impostor similarity provides the best performance for authorship clustering (4.1). Precision and recall of HAC on 250 authors with post counts drawn from Zipfian distribution. In this figure and following figures, plots end when confidence intervals on precision grow large. (left) Total correctly classified pairs / total pairs; (middle) Choosing best classified cluster per author. (right) Choosing best author per classified cluster. By all metrics, the performance of impostor-based similarity is generally best.

5.2 Increased Efficiency and Accuracy using Locality Sensitive Hashing (LSH)

As the number of documents in a corpus increases, the all-pairs comparisons required by agglomerative clustering can become prohibitively expensive. These high costs are exacerbated by impostor similarity's use of 100 iterations, and comparisons against 50 impostors per iteration, which is 5,000 times more comparisons than a simple distance metric.

We propose the use of locality sensitive hashing as a preprocessing step to reduce the number of pairs we must consider. As described in Section 4, LSH allows for search of approximate k-nearest neighbors. It works by applying multiple hash functions such that similar points are hashed together with high probability. The LSH function we use is the random hyperplane based hash function [3] implemented in Apache Mahout[2].

As Figure 3 shows, using LSH, we consider approximately 50% of direct pairs between same authors while having to consider less than 3% of the total pairs.

Not only are we able to drastically reduce the number of comparisons to generate the similarity matrix, at the same time we decrease the cost of the actual agglomeration step as we can take advantage of the very sparse form of the similarity matrix at each iteration to find the most similar element. On a 48-core 2.1ghz AMD Opteron Processor, clustering 250 blog authors using LSH preprocessing takes about 4.5 minutes. Using all pairs on the same data takes about 76 minutes.

Figure 4 shows we achieve nearly identical precision-recall results using our LSH approximation of an all-pairs approach. Because LSH does not return all

[2] See `http://mahout.apache.org/`.

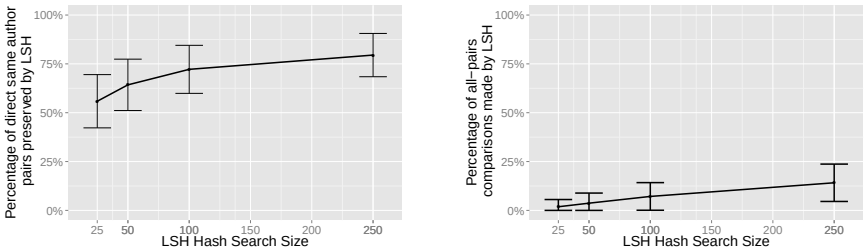

Fig. 3: Locality Sensitive Hashing is an effective method of significantly reducing the number of comparisons required for agglomerative authorship clustering. The x-axis represents the values for K in approximate K-nearest neighbor search as an independent variable. The left graph shows the percent of total same author pairs preserved by the LSH preprocessing (i.e., higher is more complete). The right graph shows the percent of the total pairs that need to be considered for the same set (i.e., lower is more efficient). Numbers show hashing of 250 authors with blog counts drawn from a Zipfian distribution (on average 3.5 posts per author).

pairs, higher recall values are, in some cases, not achievable. But for most recall values, precision is about the same.

5.3 Performance of Combined Methods

Each of the similarity functions returns information that distinguishes two documents or document clusters. We next evaluate whether these functions are complementary: i.e., if simultaneous use of them results in a performance gain.

We performed the pairwise similarity experiment described in Section 4.3, with an important difference, to test the synergy of similarity function. Specifically, we computed each type of similarity and placed different combinations into feature vectors, which we then used to train a logistic regression model. The results are shown in Figure 5.

We find that the regression results are nearly identical to simply using the impostor similarity with a hard threshold. We also find that nearly no information is gained by adding the additional features to impostor similarity. We also tried the same approach using various SVMs and achieved nearly identical results.

5.4 Impostor Robustness

Koppel and Winter [15] left open the question of whether the algorithm is affected strongly by using *mistaken impostors*. In this scenario, when trying to classify if X and Y are written by the same author, the set $Imposters_Y$ contains one or more documents that are actually written by X or vice versa. Mistaken impostors have the effect of lowering the similarity score between X and Y, as at each iteration the algorithm has a higher chance of selecting the mistaken impostor as the most similar document.

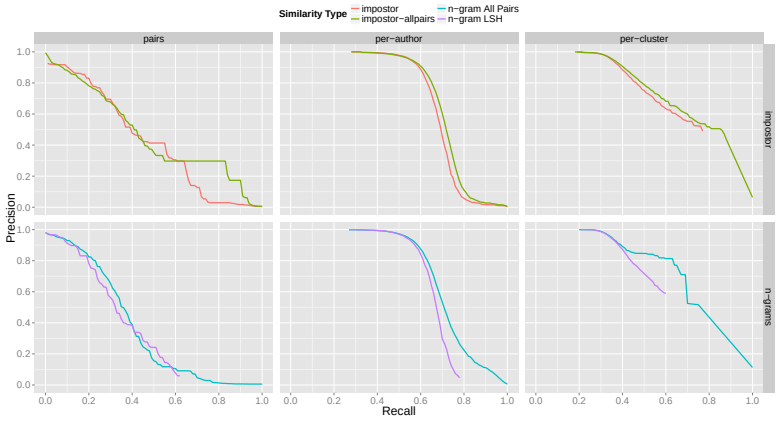

Fig. 4: Authorship clustering based on LSH versus a comparison of all pairs: in most cases, the use of LSH does not result in a reduction in precision-recall performance even though run times are greatly diminished. Each plot shows the precision and recall of HAC on 250 authors with post counts drawn from Zipfian distribution on blog posts. (left) Total correctly classified pairs / total pairs; (middle) Choosing best classified cluster per author. (right) Choosing best author per classified cluster.

To test the damaging effects of mistaken impostors, we recreated Koppel and Winter's experiment that tried to identify 250 same author pairs and 250 different author pairs. However, we intentionally inserted documents written by X into $Impostors_Y$ and documents written by Y into $Impostors_X$ ranging from 0.4–10% of the total impostor set.

As Figure 6 shows, we found in our experiments that having as large as 5% of the impostors be mistaken impostors did not have a significant impact on the overall results of the algorithm. However, at 10% (i.e., 25 out of 250), the number of mistaken impostors does have a substantive effect on performance.

6 Conclusion

We demonstrated the extension of impostor similarity to the problem of authorship clustering, evaluating its performance on a widely used blog corpus. We recreated the authorship verification results of prior work.

When extended to clustering, we showed that at a recall of 0.5, impostor similarity achieves a precision of 0.41 for the blog corpus, performing higher than other methods.

We addressed the problem of impostor similarity's higher computational costs: by using locality sensitive hashing as a preprocessing step for agglomerative clustering, the number of comparisons can be reduced by approximately 95%, while achieving nearly identical precision and recall results.

We also examined the resilience of impostor similarity to the use of mistaken impostors, which is likely to occur in some real data sets. We show that if 5%

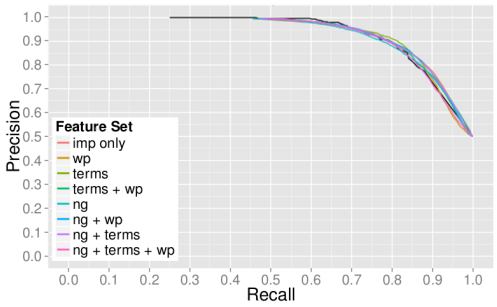

Fig. 5: Ensemble classifiers using two or more feature vectors do not significantly improve performance of authorship verification. Using regression meta classifier on various feature set combinations. All feature sets use impostor similarity. NG = ngrams, WP = writeprints featureset from [20].

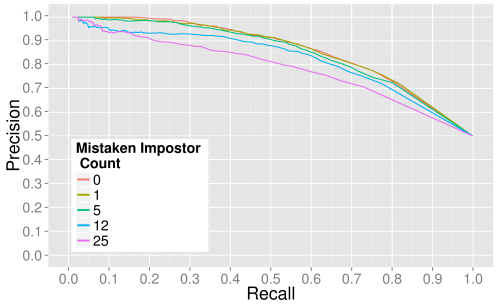

Fig. 6: In our experiments, up to 5% (i.e., 12/250) of impostors can be mistaken impostors before performance is substantively degraded for authorship verification. The different lines show the varying number of mistaken impostors. Note: the values slightly differ from the pair experiment in Figure 1 due to using only authors with sufficient data to create up to 25 mistaken impostors.

of the impostor set is actually a mistaken impostor, results are not significantly degraded.

References

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):7:1–7:29, April 2008.
2. Steven Burrows and Seyed Tahaghoghi. Source Code Authorship Attribution using n-grams. In *Proc. Australasian Document Computing Symposium*, pages 32–39, 2007.
3. Moses Charikar. Similarity Estimation Techniques from Rounding Algorithms. In *Proc. ACM Symposium on Theory of Computing (STOC)*, pages 380–388, 2002.

4. W. Bruce Croft, Donald Metzler, and Trevor Strohman. *Search Engines: Information Retrieval in Practice*. Addison-Wesley, 2010.
5. Hugo Escalante, Thamar Solorio, and Manuel Montes-y-Gómez. Local Histograms of Character N-grams for Authorship Attribution. In *Proc. Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT)*, pages 288–298, 2011.
6. Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proc. ACM Symposium on Theory of Computing (STOC)*, pages 604–613, 1998.
7. Farkhund Iqbal, Hamad Binsalleeh, Benjamin Fung, and Mourad Debbabi. Mining Writeprints from Anonymous e-Mails for Forensic Investigation. *Elsevier Digital Investigation*, 7(1-2):56–64, October 2010.
8. Farkhund Iqbal, Liaquat A. Khan, Benjamin Fung, and Mourad Debbabi. e-Mail Authorship Verification for Forensic Investigation. In *Proc. ACM Symposium on Applied Computing*, pages 1591–1598, 2010.
9. Patrick Juola. Authorship attribution. *Found. Trends Inf. Retr.*, 1(3):233–334, December 2006.
10. Patrick Juola and Efstathios Stamatatos. Overview of the Author Identification Task at PAN 2013. In *Proc. Conference and Labs of the Evaluation Forum (CLEF)*, September 2013.
11. Hisashi Koga, Tetsuo Ishibashi, and Toshinori Watanabe. Fast agglomerative hierarchical clustering algorithm using Locality-Sensitive Hashing. *Knowledge and Information Systems*, 12(1):25–53, July 2006.
12. M. Koppel, J. Schler, S. Argamon, and Y. Winter. The Fundamental Problem of Authorship Attribution. *English Studies*, 93(3):284–291, 2012.
13. Moshe Koppel and Jonathan Schler. Authorship Verification as a One-class Classification Problem. In *Proc. International Conference on Machine Learning (ICML)*, pages 489–495, July 2004.
14. Moshe Koppel, Jonathan Schler, and Shlomo Argamon. Authorship Attribution: What's Easy and What's Hard? *Journal of Law and Policy*, 21(2):317–332, 2013.
15. Moshe Koppel and Yaron Winter. Determining if two documents are written by the same author. *Journal of the Association for Information Science and Technology*, 65(1):178–187, 2014.
16. R. Layton, P. Watters, and R. Dazeley. Authorship Attribution for Twitter in 140 Characters or Less. In *Proc. Cybercrime and Trustworthy Computing Workshop (CTC)*, pages 1–8, July 2010.
17. Robert Layton, Paul Watters, and Richard Dazeley. Unsupervised authorship analysis of phishing webpages. *Proc. International Symposium on Communications and Information Technologies (ISCIT)*, pages 1104–1109, October 2012.
18. Robert Layton, Paul Watters, and Richard Dazeley. Automated unsupervised authorship analysis using evidence accumulation clustering. *Natural Language Engineering*, 19:95–120, 1 2013.
19. Andrew McDonald, Sadia Afroz, Aylin Caliskan, Ariel Stolerman, and Rachel Greenstadt. Use Fewer Instances of the Letter "I": Toward Writing Style Anonymization. In *Proc. International Conference on Privacy Enhancing Technologies (PETS)*, pages 299–318, 2012.
20. A. Narayanan, H. Paskov, N.Z. Gong, J. Bethencourt, E. Stefanov, E.C.R. Shin, and D. Song. On the Feasibility of Internet-Scale Author Identification. In *Proc. IEEE Symposium on Security and Privacy*, pages 300–314, May 2012.

21. Ryan O'Donnell, Yi Wu, and Yuan Zhou. Optimal Lower Bounds for Locality-Sensitive Hashing (Except When Q is Tiny). *ACM Trans. Comput. Theory*, 6(1):5:1–5:13, March 2014.
22. S.R. Pillay and T. Solorio. Authorship attribution of web forum posts. In *Proc. eCrime Researchers Summit (eCrime)*, pages 1–7, Oct 2010.
23. Martin Porter. An Algorithm for Suffix Stripping. *Program*, 14(3):130–137, 1980.
24. J. Schler, M. Koppel, S. Argamon, and J. Pennebaker. Effects of Age and Gender on Blogging. In *Proc. AAAI Spring Symposia on Computational Approaches to Analyzing Weblogs*, 2006.
25. Shachar Seidman. Authorship Verification Using the Impostors Method: Notebook for PAN. In *Proc. Conference and Labs of the Evaluation Forum (CLEF)*, pages 13–16, September 2013.
26. Efstathios Stamatatos. A Survey of Modern Authorship Attribution Methods. *J. Am. Soc. Inf. Sci. Technol.*, 60(3):538–556, March 2009.