Heuristic Authorship Obfuscation

Janek Bevendorff[*] **Martin Potthast**[†] **Matthias Hagen**[‡] **Benno Stein**[*]

[*]Bauhaus-Universität Weimar
[†]Leipzig University
[‡]Martin-Luther-Universität Halle-Wittenberg

<first>.<last>@uni-{weimar, leipzig}.de
<first>.<last>@informatik.uni-halle.de

Abstract

Authorship verification is the task of determining whether or not two texts were written by the same author. This paper deals with the adversary task, called *authorship obfuscation*: Preventing verification by altering a to-be-obfuscated text. We introduce an approach that (1) models writing style difference as the Jensen-Shannon distance between the character n-gram distributions of texts, and (2) manipulates an author's subconsciously encoded writing style in a sophisticated manner using heuristic search. To obfuscate, we explore the huge space of textual variants in order to find a paraphrased version of the to-be-obfuscated text that has a sufficient Jensen-Shannon distance at minimal costs in terms of text quality loss. We analyze, quantify, and illustrate the rationale of this approach, define paraphrasing operators, derive obfuscation thresholds, and develop an effective obfuscation framework. Our authorship obfuscation approach defeats state-of-the-art verification approaches, including unmasking and compression models, while keeping text changes at a minimum.

1 Introduction

Can the authorial style of a text be consistently manipulated? More than a century worth of research on stylometry and authorship analysis could not produce a reliable approach to do so manually. In the context of computational authorship obfuscation, a handful of approaches have achieved some limited success but are still rather insufficient. Rule-based approaches are neither flexible, nor is stylometry understood well enough to compile rule sets that specifically target author style. Monolingual machine translation-based approaches suffer from a lack of training data, whereas applying multilingual translation in a cyclic manner as a workaround has proved to be ineffective. In addition, none of the existing approaches offers a means to control the result quality. Given recent advances in controlled text generation, it stands to reason that a lot more can be achieved.

In this paper, we depart from the mentioned obfuscation paradigms and, for the first time, cast author obfuscation as a heuristic search problem. Given a to-be-obfuscated text, we search for a cost-minimum sequence of tailored paraphrasing operations that achieve a significant increase of the text's style distance to other texts from the same author under a generic writing style representation; costs accrue through operations in terms of their estimated text quality reduction. By designing a hybrid search strategy that neglects admissibility only in the pooling phase, we obtain a significant reduction of the exponentially growing search space that is induced by the paraphrasing operators, enabling the use of informed search algorithms. Moreover, we developed a sophisticated framework to deal with the conflicting objectives that naturally arise with such kind of complex text synthesis tasks: a compact representation of the search space of paraphrased text variants, and an effective and efficient, non-monotonic exploration of this search space.[1]

Our key contributions are a greedy obfuscation approach that maximizes obfuscation gain per operation (Section 3); based on that, an obfuscation heuristic that reconciles obfuscation gain with text quality loss (Section 4); as well as an extensive comparative evaluation (Section 5). Relevant code and research data is released publicly on GitHub.[2]

2 Related Work

Authorship analysis dates back over 120 years (Bourne, 1897) and has mostly dealt with authorship attribution (given a text of unknown authorship and texts from known candidate authors, attribute

[1] Up to 10,000 text variants per second on a standard PC.

[2] Code and data: https://github.com/webis-de/acl-19

the unknown text to its true author among the candidates). More recently, the task of authorship verification attracted a lot of interest (given a text of unknown authorship and a set of texts from one known author, verify whether the unknown text is written by that author) since it lies at the heart of many authorship-related problems.

Systematic reviews on authorship analysis have been contributed by Juola (2006) and Stamatatos (2009) and the effectiveness of character 3-grams today is "folklore knowledge," albeit not systematically proven. Still, a complete list of stylometric features has not been compiled to date. Abbasi and Chen (2008) proposed *writeprints*, a set of over twenty lexical, syntactic, and structural text feature types, which has gained some notoriety within attribution, verification, but also for "anonymizing" texts (Zheng et al., 2006; Narayanan et al., 2012; Iqbal et al., 2008; McDonald et al., 2012).

Instead of relying on a rich feature set, Zhao et al. (2006) only extract POS tag distributions and interpret style differences as measurable by the Kullback-Leibler divergence. Teahan and Harper (2003) and Khmelev and Teahan (2003) use compression as an indirect means to measure stylistic difference; later adapted and improved by Halvani et al. (2017). Koppel and Schler (2004) developed the *unmasking* approach based on the 250 most frequent function words, which are iteratively removed, effectively reducing the differentiability between the texts. The idea behind this approach is that texts written by the same author only differ in few superficial features. By removing those superficial features, differentiability between texts by the same author is expected to degrade faster than for texts written by different authors.

Among the first to tackle authorship obfuscation were Rao and Rohatgi (2000), who used cyclic machine translation. Later Brennan et al. (2012) found that machine translation is ineffective and due to its blackbox character also rather uncontrollable. Instead, Xu et al. (2012) proposed within-language machine translation to translate directly between styles. The practicality of this approach, however, is diminished by the general lack of large-scale parallel training data. Another obfuscation approach by Kacmarcik and Gamon (2006) directly targets Koppel and Schler's unmasking. By iteratively removing the most discriminatory text features, the classification performance of an unmasking verifier is degraded—at the cost of rather unreadable texts.

From 2016 to 2018, a shared task on authorship obfuscation was organized at PAN (Potthast et al., 2018). Some of the seven participating teams suggested rather conservative rule-based approaches that do not change a text sufficiently to obfuscate authorship against most state-of-the-art verifiers but other obfuscators "fooled" several verifiers, yet again, generating rather unreadable texts. To score high in terms of text quality and obfuscation performance, the shared task organizers asked for approaches that more carefully *paraphrase* a text (i.e., the meaning should stay the same and the text should still be readable). Our new authorship obfuscation approach is inspired by Stein et al. (2014)'s heuristic paraphrasing idea for "encoding" an acrostic in a given text and by Kacmarcik and Gamon's observation that changing rather few text passages may successfully obfuscate authorship.

3 Greedy Obfuscation

We approach obfuscation from a verification perspective: Given texts from the same author, one of which is not publicly known to be written by that author, the goal is to paraphrase that text so that verification attempts against the other texts fail. In this setting, the key element of our heuristic obfuscation approach is a basic, yet powerful distributional representation of writing style: the Jensen-Shannon distance of the character trigram frequency distribution of the to-be-obfuscated text compared to the others. This model serves three purposes at once: (1) as a stopping criterion, (2) as a primary selection criterion for parts of the text that will yield the highest obfuscation gains if changed, and, (3) as part of our heuristic enabling informed search, which reconciles obfuscation gain with potential text quality loss. In what follows, we formally motivate these dimensions.

3.1 Measuring Stylistic Distance

In order to know when to stop obfuscating a text we require a style distance measure. Once a text has been changed sufficiently and its style distance to other texts from the same author exceeds a given threshold, the obfuscator terminates.[3]

By utilizing character trigram frequencies to represent texts, we employ one of the most versatile

[3] Another possibility is to stop once the decision of existing verifiers switches to *different-authors*. However, this would introduce many more hyperparameters and biases regarding the verifiers, let alone the prohibitive runtime overhead.

yet simple features available for authorship analysis, encoding many aspects of authorial style at the same time including vocabulary, morphology, and punctuation. Based on this representation, we consider the well-known Kullback-Leibler divergence (KLD) as a style distance measure:

$$\mathrm{KLD}(P\|Q) = \sum_i P[i] \log \frac{P[i]}{Q[i]}\,, \tag{1}$$

where P and Q are discrete probability distributions corresponding to the relative frequencies of character trigrams in the to-be-obfuscated text and the known texts respectively. For true probability distributions, the KLD is always non-negative.

The KLD has shortcomings. First, it is asymmetric, so it is not entirely clear which character distribution should be P and which should be Q when comparing texts. Second, the KLD is defined only for distributions P and Q where $Q[i] = 0$ implies $P[i] = 0$. Conversely, $P[i] = 0$ yields a zero summand. Since we want to avoid reducing or skewing the measure further by "subsetting" or smoothing the trigrams, we resort to the Jensen-Shannon distance JS_Δ (Endres and Schindelin, 2003) in lieu of the KLD. The JS_Δ is a metric based on the symmetric Jensen-Shannon divergence (JSD) that is defined as

$$\mathrm{JSD}(P\|Q) = \frac{\mathrm{KLD}(P\|M) + \mathrm{KLD}(Q\|M)}{2}\,, \tag{2}$$

with

$$M = \frac{P+Q}{2}\,. \tag{3}$$

Introducing the artificial distribution M circumvents the KLD's problem of samples of one distribution being unknown in the other. Since $M[i]$ can never be 0 for any i with $P[i] + Q[i] > 0$, all summands of either $\mathrm{KLD}(P\|M)$ or $\mathrm{KLD}(Q\|M)$ must also be non-zero. Using the base-2 logarithm in the KLD, the JSD is $[0, 1]$-bounded. The JS_Δ metric is derived as

$$\mathrm{JS}_\Delta(P, Q) = \sqrt{2 \cdot \mathrm{JSD}(P\|Q)}\,. \tag{4}$$

3.2 Adaptive Obfuscation Thresholds

During pilot experiments on our training data, we observed that a fixed JS_Δ threshold as the obfuscation target is a bad idea: it leads to over- or under-obfuscation for text pairs that have an a-priori high or low style distance. It also turned out that JS_Δ is inversely correlated with text length: pairs of long texts are less distant to each other than pairs of short texts, since, the shorter a text the sparser and noisier is its trigram distribution. This even holds if the texts are written by the same author. Figure 1 plots the JS_Δ over the text length in our training data, revealing an approximately logarithmic relationship. The most interesting observation is the almost length-invariant spread of the resulting curves. Moreover, depending on their class, the curves tend to converge towards the upper / lower bounds of this spread with growing length, thus being visibly separated.

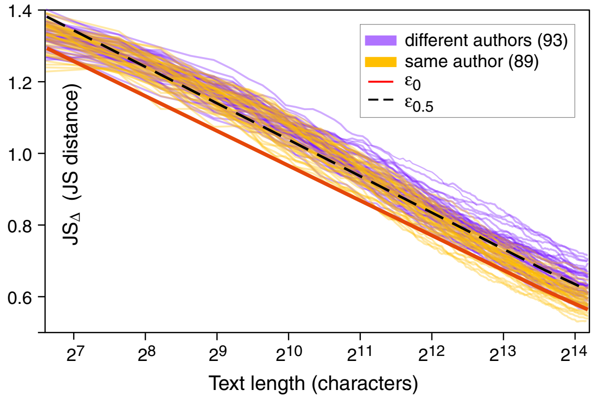

Figure 1: JS_Δ in our training data over text length. Each line corresponds to a text pair The straight lines indicate the 0th and the 50th percentiles of distances within the true *different-authors* cases.

Assuming that the observed JS_Δ-to-length relationship generalizes to other text pairs of similar length—a hypothesis which merits further investigation in future work—, we measure style distance in JS_Δ@L (Jensen-Shannon distance at length) and fit threshold lines to define obfuscation levels. Table 1 details the obfuscation levels ε_k corresponding to a linear least-squares fit on the logarithmic scale through a given level's k-th percentile of the distribution of JS_Δ in the *different-authors* class; the 0th percentile ε_0 and the 50th percentile $\varepsilon_{0.5}$ are displayed in Figure 1. The ε_0 threshold serves as an obfuscation baseline, indicating a *same-author* case as unobfuscated, if the JS_Δ between its documents is below this threshold. Otherwise, we call the obfuscation moderate, strong, stronger, and over-obfuscated, depending on the threshold the JS_Δ exceeds.

Regarding the line fit coefficients given in Table 1, the gradients of higher ε thresholds are slightly steeper, providing further evidence of the convergence rate of *different-authors* cases. The ε_0 threshold line will cross the x axis for text lengths

Threshold	Obfuscation level	Slope	Intercept
$< \varepsilon_0$	No Obfuscation	n / a	n / a
$\geq \varepsilon_0$	Moderate Obfuscation	−0.099	1.936
$\geq \varepsilon_{0.5}$	Strong Obfuscation	−0.103	2.056
$\geq \varepsilon_{0.7}$	Stronger Obfuscation	−0.104	2.083
$> \varepsilon_{0.99}$	Over-obfuscation	−0.107	2.168

Table 1: Obfuscation levels and their log-scale polynomial fit coefficients on our training corpus.

of $x \approx 2^{19.5}$ characters. Since negative distances are not sensible, such book-sized texts may be split into smaller chunks, which then can be obfuscated individually. Note that we were able to reproduce these threshold observations on the PAN 2014 novels corpus (Stamatatos et al., 2014), albeit obtaining slightly different coefficients. In practice, we recommend training the coefficients on an appropriate corpus matching genre and register of the to-be-obfuscated texts.

3.3 Ranking Trigrams for Obfuscation

Our key idea to yield a strong obfuscation (compared to other texts from the same author) is to iteratively change the frequency of those trigrams of the to-be-obfuscated text for which the positive impact on JS_Δ is maximum. In each iteration we rank the trigrams by their influence on JS_Δ via their partial KLD derivative, assuming that probability distribution Q is to be obfuscated:

$$\frac{\partial}{\partial\, Q[i]} \left(P[i] \log_2 \frac{P[i]}{Q[i]} \right) = -\frac{P[i]}{Q[i] \ln 2} \,. \tag{5}$$

Omitting constants, we get the rank-equivalent

$$\mathrm{R}_{\mathrm{KL}}(i) = \frac{P[i]}{Q[i]} \,. \tag{6}$$

R_{KL} gets larger with smaller $Q[i]$. I.e., a single obfuscation step boils down to *removing* one occurrence of the most influential trigram from the to-be-obfuscated text. This can be done naively by simply “cutting it out” (which we tried as a proof-of-concept), or, more sensibly, via a targeted paraphrasing operation replacing a text passage with the trigram by another semantically equivalent text passage without the trigram. Then, the trigrams are re-ranked and the procedure is repeated until JS_Δ exceeds the desired obfuscation threshold. We call this strategy *obfuscation by reduction*. Reversing the roles of P and Q yields an *addition* strategy, which we leave for future work.

The above described greedy obfuscation effectively hindered verification in our pilot experiments. However, the naive cut-it-out variant results in rather unreadable texts, and, it may be easily “reverse engineered” by an informed verifier. Even with more sophisticated paraphrasing operations, a reverse-engineering attack against the greedy strategy seems plausible. Thus, we suggest to augment the greedy approach with an informed search, which is introduced in the next section.

4 Heuristic Search for Obfuscation

An author of a to-be-obfuscated text does obviously not wish her text to be “foozled” due to obfuscation (e.g., by naively cutting out trigrams). Actually, the text has to convey the same message as before and, ideally, it should look “inconspicuous” to an extent that readers do not suspect tampering (Potthast et al., 2016). However, automatic paraphrasing is still in its infancy: Beyond synonym substitution, paraphrasing operators targeting single words have hardly been devised so far. Still, the paraphrasing operators we are looking for do not have to alter a text substantially, which enables us to better estimate an operator’s negative impact on text quality. Furthermore, similar to the presented greedy obfuscation, we can stop modifying a text when the desired obfuscation threshold is reached, which renders our approach “minimally invasive.” The optimization goals can be summarized as follows:

1. Maximize the obfuscation as per the JS_Δ beyond a given ε_k without “over-obfuscating.”
2. Minimize the accumulated text quality loss from consecutive paraphrasing operations.
3. Minimize the number of text operations.

Heuristic search is our choice to tackle this optimization problem. We will pay attention to admissibility for two reasons: (1) to understand (in terms of modeling) the nature of the problem, and (2) to be able to compute an optimum solution if time and space constraints permit. However, due to the exponential size of the induced state space (text versions as nodes, paraphrasing operators as edges), we may give up admissibility while staying within acceptable error bounds. In the following, we derive an admissible obfuscation heuristic and suggest a small, viable set of basic paraphrasing operators as an initial proof of concept.

4.1 An Admissible Obfuscation Heuristic

Let $h(n)$ denote a heuristic estimating the optimal cost for reaching a desired obfuscation threshold from node n, and let $g(n)$ denote the path costs to n starting at the original text node s.

Applying a paraphrasing operator has a highly non-linear effect on text quality (some changes are inconspicuous, others are not) and may also restrict the set of applicable operators (in the same text). For instance, applying the same operator a third time in a row may entail higher (quality) costs compared to applying it for the first time. This means that different paths from s to n can come with different estimations for the rest cost $h(n)$—in a nutshell, the parent discarding property may not hold (Pearl, 1984). A similar effect, but rooted in a different cause, results from the observation that some authors' texts are easier to be obfuscated than others. We can address both issues and re-install the conditions for parent discarding and admissible search by updating the operator costs for future application beyond node n, such that $g(n)$ turns into "normalized path costs."

Based on both the desired obfuscation threshold ε and the JS distance $\mathrm{JS}_{\Delta n}$ of the text at node n to the other text(s) from the same author, we define the prior heuristic as

$$h_{prior}(n) = \varepsilon - \mathrm{JS}_{\Delta n}. \tag{7}$$

The normalized path costs g_{norm} are defined as the cost-to-gain ratio of the accumulated path costs $g(n)$ to total JS_{Δ} change from start node s:

$$g_{norm}(n) = \frac{g(n)}{\mathrm{JS}_{\Delta n} - \mathrm{JS}_{\Delta s}}. \tag{8}$$

Finally, the heuristic $h(n)$ is defined as the product of $h_{prior}(n)$ and $g_{norm}(n)$:

$$h(n) = (\varepsilon - \mathrm{JS}_{\Delta n}) \cdot \frac{g(n)}{\mathrm{JS}_{\Delta n} - \mathrm{JS}_{\Delta s}}. \tag{9}$$

The prior heuristic guarantees convergence towards zero as we approach a goal node that exceeds the obfuscation threshold ε, while the normalized path costs determine the slope of the heuristic.

Consistency and Admissibility

A heuristic $h(n)$ is admissible if it does not exceed $h^*(n)$, the true cost of reaching an optimum goal via state n, for all n in the search space. Monotonicity $h(n) \leq c(n, n') + h(n')$ is a sufficient condition for admissibility, yet easier to show. Rewriting it as

$$-h(n') + h(n) \leq g(n') - g(n),$$

and inserting in the heuristic Equation 9 yields

$$-\frac{(\varepsilon - \mathrm{JS}_{\Delta n'}) \cdot g(n')}{\mathrm{JS}_{\Delta n'} - \mathrm{JS}_{\Delta s}} + \frac{(\varepsilon - \mathrm{JS}_{\Delta n}) \cdot g(n)}{\mathrm{JS}_{\Delta n} - \mathrm{JS}_{\Delta s}} \leq g(n') - g(n)\,.$$

Defining $\bar{g}(n) = \mathrm{JS}_{\Delta n} - \mathrm{JS}_{\Delta s}$ as change function and inserting previous definitions we get

$$\frac{-h_{prior}(n') \cdot g(n')}{\bar{g}(n')} - \frac{-h_{prior}(n) \cdot g(n)}{\bar{g}(n)} \leq g(n') - g(n)\,.$$

We know $h_{prior}(n)$ to be monotonically decreasing, inverse to $\bar{g}(n)$, and converging towards zero as we approach a goal. If the cost and change functions $g(n)$ and $\bar{g}(n)$ are equivalent up to scale, they cancel out each other (up to scale), the slope of their quotient becomes zero, and the inequality turns into equality. Otherwise, if $g(n)$ dominates $\bar{g}(n)$, the inequality still holds. Though, if $\bar{g}(n)$ dominates $g(n)$, the sign of the quotient's gradient flips (as can be proved by the quotient rule), breaking the inequality and violating consistency. But since JS_{Δ} is bounded by $\sqrt{2}$ globally, the change function $\bar{g}(n)$ cannot be superlinear.

Limitations of our argument: (1) occasionally $\bar{g}(n)$ can locally dominate $g(n)$, and (2) both functions are presumed differentiable at n. In practice, the latter may hardly ever be true as texts are noisy, text operation side effects are unpredictable, and, the cumulative change function is not guaranteed to be monotonic. Still, step costs $c(n, n')$ will never be negative, which makes $g(n)$ monotonic but not necessarily differentiable. Thus, the heuristic function will not be fully consistent and may even overestimate.

In a practical scenario we can directly control the cost but not the change function, so we will have to deal with problems of overestimation and local optima. Generally, the first few steps of a search path are the most problematic since with little prior information the heuristic has to extrapolate based on very few data points, but is still expected to accurately estimate the remaining costs. Hence, an early heuristic is particularly susceptible to noise and can only give a coarse estimate. With more cumulative cost and change information available, the heuristic will stabilize towards the mean cost-gain proportion and eventually converge. This stabilization occurs quickly. In real application scenarios, we keep overestimation at a minimum or even avoid it at all and therefore obtain an approximately admissible heuristic due to the JS_{Δ}'s boundedness.

4.2 Search Space Challenges

Given a longer text (one page or more), the number of potential operator applications is high. The most direct way to expand a node is to generate a successor with each applicable operator for each occurrence of each selected n-gram, but this will inevitably result in an immense number of very similar states with identical costs and almost identical JS_Δ change. I.e., the main challenge is to find a sensible middle ground between accepting a non-optimal solution too quickly or not finding a solution at all. Recall that one can easily turn the A* search into a depth-first or breadth-first search by making successor generation too cheap or too costly: depth-first search will always find a (non-optimal) solution after a sufficient number of operations, while breadth-first will never terminate before running out of memory.

We can accept a near-optimal solution, so selecting one or two occurrences of an n-gram (instead of all) will be sufficient. A potential problem is that the applicability of a high-quality operator is often restricted. However, one can increase the application probability by selecting not only the top-ranked n-gram but a small number of different near-top n-grams. This way, we have multiple high-impact n-grams with different contexts to work with, and we increase the chances of applying the operator opening alternative paths for the search. In practice, JS_Δ change is not a monotonic function and steepest-ascent hill climbing does not guarantee an overall lowest-cost path. Thus, we applied each operator to two occurrences of the top ten n-grams and selected from these (up to 140 successors) six randomly for expansion. However, even with only six successors we still generate millions of nodes very quickly and will eventually run out of memory without finding a solution. Fortunately, we can assume that exploring more neighbors will not produce much better results after a while, so we can restart the search from a few promising nodes and still discard other open nodes.

4.3 Paraphrasing Operators

Our prototype employs the seven basic text operators shown in Table 2. These are to be understood as a pilot study, more state-of-the-art text generation operators can be added easily. The most versatile yet most disruptive basic modification are (1) the removal of an n-gram, and (2) flipping two of its (or adjacent) characters. Such operations only are a last resort, and we hence set their costs much higher than those of other operators. As steps towards real paraphrasing, we also perform context-free synonym and hypernym replacement based on WordNet (Miller, 1995) as well as context-dependent replacements and deletions using the word 5-gram model of Netspeak (Stein et al., 2010). Also, a map of similar punctuation characters indicates inconspicuous character swaps.

	Operator name	Cost value
(1)	n-gram removal	40
(2)	Character flips	30
	Context-free synonyms	10
	Context-free hypernyms	6
	Context-dependent replacement	4
	Character maps	3
	Context-dependent deletion	2

Table 2: Implemented text operators and their assigned step costs in our heuristic obfuscation prototype.

5 Evaluation

To evaluate our approach, we report on: (1) an efficiency comparison of greedy versus heuristic obfuscation, (2) an effectiveness analysis against well-known authorship verification approaches (unmasking, compression-based models, and PAN participants), as well as (3) a review and discussion of an example obfuscated text.

Our experiments are based on PAN authorship corpora and our new Webis Authorship Verification Corpus 2019 of 262 authorship verification cases (Bevendorff et al., 2019), half of them *same-author* cases, the other half *different-authors* cases (each a pair of texts of about 23,000 characters / 4,000 words). Instead of the more particular genres studied at PAN, our new corpus contains longer texts and more modern literature from Project Gutenberg. We also took extra care to cleanse the plain text, unified special characters, and removed artifacts; in particular, we ensured that no author appears in more than one case. The training-test split is 70-30 so as to have a decent training portion. The corpus is released alongside the code of our search framework and other research data.

5.1 Search Over Greedy Obfuscation

Table 3 contrasts the efficiency of the greedy obfuscation with that of our heuristic search approach, measured in terms of medians of total text operations and path costs. Heuristic search achieves a decrease of operations of up to 19% for texts that

Efficiency	**Cases**		**Median**		
	Subset	#	Greedy	A*	Gain
Total operations	all	41	148	145	−2 %
	1+ ops	28	241	202	−16 %
	100+ ops	21	291	236	−19 %
Path costs	all	41	5,960	1,968	−67 %
	1+ ops	28	9,680	2,712	−72 %
	100+ ops	21	11,680	2,935	−75 %

Table 3: Efficiency of greedy obfuscation vs heuristic obfuscation for an obfuscation threshold of $\varepsilon_{0.5}$.

Confidence	**Unobfuscated**			**Obfuscated**		
Hyperplane threshold	Classified cases [%]	Effectiveness Prec.	Rec.	Classified cases [%]	Effectiveness Prec.	Rec.
0.8	11.3	1.00	0.17	2.5	1.00	0.02
0.7	15.0	1.00	0.24	6.2	1.00	0.05
0.6	18.8	1.00	0.24	11.3	0.75	0.07
0.5	26.3	1.00	0.29	24.0	0.86	0.15
0.0	100.0	0.74	0.63	100.0	0.71	0.42

Table 4: Unmasking performance on our test data at various confidence thresholds before and after obfuscation. Recall treats unclassified cases as false negatives.

need at least 100 operations and an accumulated path cost decrease of up to 75%. Since the greedy obfuscation approach cannot choose among different operators, it must rely on the most effective one to achieve the obfuscation goal, incurring significant path costs. Given that both obfuscators employ adaptive thresholds, there are cases which do not require any (or only little) obfuscation, whereas others need more than 100. The latter are of particular interest, since it is here where heuristic obfuscation outperforms greedy obfuscation the most.

5.2 Obfuscation against Unmasking

One of today's most effective and robust verification approaches is unmasking by Koppel and Schler (2004). It decomposes to-be-compared texts into two chunk sets, and iteratively trains a linear classifier to discriminate between them while removing the most significant features in each iteration to measure the increased reconstruction error. This error increases faster for *same-author* cases since those share more function words than do *different-authors* cases. Fooling unmasking verification provides us with evidence that our obfuscation technique works at a deeper level than just the few most superficial text features. Unmasking further produces curve plots of the declining classification accuracy, which render the effects of obfuscation accessible to human inspection and interpretation.

Following Koppel and Schler, we use the chunk frequencies of the 250 most common words as features, determine classification accuracy by 10-fold cross validation using an SVM classifier, and remove ten features per iteration. The final curves and their gradients are used to train another SVM to separate curves originating from *same-author* cases from *different-authors* curves. Following the example of the PAN competitions where the incentive was to classify only high-confidence cases, we consider decisions for cases which can be classified with pre-determined confidence thresholds (i.e., the distance to the hyperplane), which allows to maximize precision at the cost of recall.

Table 4 contrasts the performance of unmasking before and after obfuscation on the test data. With increasing confidence thresholds, between 19 % down to 11 % of the cases are decidable before obfuscation, decreased by a factor of 2 to 4 after obfuscation. On average, 205 trigrams were obfuscated; as little as about 3 % of a text.

5.3 Obfuscation against Compression Models

Another verification approach that differs from traditional feature-engineering are compression-based models. We use the approach by Halvani et al. (2017), who recommend the compression-based cosine (CBC) by Sculley and Brodley (2006) calculated on the text pairs after compression with the PPMD algorithm (Howard, 1993).

Figure 2 shows CBC values on a random selection of 20 exemplary *same-author* cases from our test dataset before and after obfuscation with the decision threshold highlighted. Quite impressively, almost none of the cases are classified correctly anymore after obfuscation. Overall, the accuracy drops from originally 71 % to 55 %, which is equivalent to random guessing. This strong effect can be explained as follows: Sculley and Brodley describe their metrics in terms of the Kolmogorov complexity, but the reason why natural language allows for very good compression ratios is its predictability (printed English has an entropy of at most 1.75 bits per character (Brown et al., 1992)). PPMD uses finite-order Markov language models for compression, which are effective at predicting characters in a sentence, but sensitive to increased entropy, which is the result of our obfuscation.

5.4 PAN Obfuscation Evaluation

We further conducted an extensive evaluation of our obfuscation scheme against the top submissions to

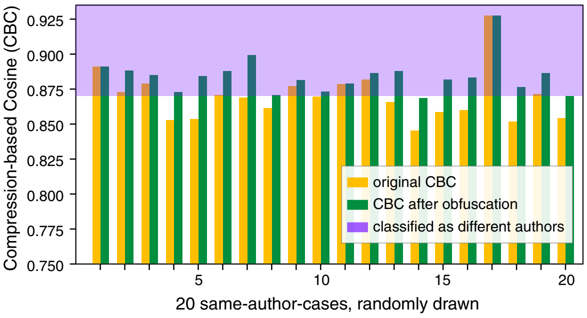

Figure 2: CBC values of 20 PPMD-compressed *same-author* pairs before and after obfuscation up to the obfuscation threshold $\varepsilon_{0.7}$. The classification threshold by which *same-author* and *different-authors* cases could be distinguished is highlighted in the top portion.

the verification task at PAN 2013–2015 (Juola and Stamatatos, 2013; Stamatatos et al., 2014, 2015). The results are shown in Table 5. On all verifiers tested, we achieve an average AUC and C@1 reduction of around 10 and 6 percentage points on three of the corpora. With only minimal text modifications, this puts us in second place on the PAN13 and PAN15 corpora, and fourth on PAN14 Essays compared to other obfuscators submitted to PAN (Hagen et al., 2017). The PAN14 Novels corpus turns out to be the most challenging for our approach and there are multiple reasons for that. First, the texts are significantly longer. This makes it difficult to assess the overall obfuscation with a global measure like JS_{Δ}. As a result, only few sentences were actually obfuscated with most of the text left untouched. Insofar, we were surprised to see any significant effect at all (best individual result: 13 percentage points). To make matters worse, the flat search landscape spanned by our obfuscation operators leads to an increasing number of reopened states on these longer texts, greatly reducing the efficiency of the heuristic search. This reveals an important detail to explore in future work: obfuscation operations need to be distributed across the whole text and progress needs to be measured on smaller parts of it to ensure uniform obfuscation of everything and avoid obfuscation "hot spots". Secondly, the number of "known" texts varies substantially, which demands more research into how we can calculate a minimal yet sufficient JS_{Δ}@L stopping criterion if a larger amount of known material is available. Thirdly, the corpus consists primarily of works by H. P. Lovecraft paired with fan fiction, which incurs unforeseeable global corpus features that verifiers can exploit, but which we do not consider for obfuscation. Lastly, we identify *kocher15* as the most difficult verifier for us to obfuscate. Employing an impostor approach on the most frequent words, it was not the best-performing verifier in the first place, but proves most resilient against our "reductive" obfuscation, which tends to obfuscate only n-grams that are already rare for maximum effect. We expect that augmenting a reduction obfuscation with the previously-mentioned extension strategy will yield better results and an overall safer obfuscation.

5.5 Example of an Obfuscated Text

Assessing the text quality in tasks that involve generation, such as translation, paraphrasing, and summarization, is still mostly manual work. Frequently used measures like ROUGE cannot be applied in the context of obfuscation, since our obfuscated texts are up to 97 % identical to their unobfuscated versions. This is why we resort to manually inspecting obfuscated texts and the changes made. Below is an excerpt of an original text along with the obfuscations applied to it. Selected trigrams are underlined, removed words are struck out, and inserted words are highlighted:

> ’It was the only chance ~~we had~~w ehad
> to win.’ Duke swallowed the idea
> slowly. He couldn’t picture a
> ~~planet~~satellite giving up its last
> protection for ~~a~~phi desperate effort
> to end the war on purely offensive
> drive. Three billion people watching
> the home fleet take off,
> ~~knowing~~deciding the skies were
> ~~open~~resort for all the ~~hell~~mischief
> that a savage enemy could send! On
> Earth, the World Senate hadn’t
> permitted the building of one
> ~~battleship~~frigate, for fear of
> reprisal. [...]

Excerpt of *Victory* by Lester del Rey

We selected an example where, by chance, different operators were applied in close vicinity. This "density" of operations is not representative. We can see both high- and low-quality replacements at work. Most can be attributed to the WordNet synonym operator. The replacement of "a" with "phi" is clearly such a case. The more suitable replacements originate from more context-dependent replacements, whereas "we had" → "w ehad" is a result of the flip operator.

For comparison with related work, we carried out a human assessment of a few random obfusca-

Verifier	Unobfuscated			Obfuscated			Difference		
	AUC	C@1	FS	AUC	C@1	FS	AUC	C@1	FS
a) PAN13									
bagnall15	0.86	0.79	0.68	0.74	0.64	0.48	0.11	0.15	0.20
castillojuarez14	0.49	0.43	0.21	0.50	0.53	0.27	-0.02	-0.10	-0.06
castro15	0.93	0.77	0.71	0.87	0.73	0.64	0.06	0.03	0.08
frery14	0.62	0.57	0.35	0.37	0.40	0.15	**0.25**	0.17	0.20
khonji14	0.86	0.76	0.65	0.70	0.60	0.42	0.16	0.16	0.23
kocher15	0.75	0.64	0.48	0.77	0.65	0.50	-0.02	-0.01	-0.02
layton14	0.62	0.67	0.41	0.47	0.53	0.25	0.15	0.13	0.16
mezaruiz14	0.75	0.65	0.49	0.57	0.53	0.30	0.18	0.12	0.19
mezaruiz15	0.73	0.71	0.52	0.50	0.53	0.26	0.24	**0.18**	**0.26**
modaresi14	0.50	0.50	0.25	0.47	0.50	0.24	0.03	0.00	0.02
moreau14	0.77	0.62	0.48	0.61	0.51	0.32	0.16	0.11	0.17
moreau15	0.71	0.47	0.33	0.60	0.47	0.28	0.12	0.00	0.05
singh14	0.39	0.33	0.13	0.44	0.43	0.19	-0.06	-0.10	-0.06
zamani14	0.75	0.70	0.53	0.71	0.70	0.50	0.05	0.00	0.03
Average							**0.10**	**0.06**	**0.10**
b) PAN14 Essays									
bagnall15	0.57	0.55	0.31	0.43	0.45	0.19	0.14	0.10	0.12
castillojuarez14	0.55	0.58	0.32	0.55	0.58	0.32	0.00	0.00	0.00
castro15	0.62	0.59	0.36	0.51	0.53	0.27	0.11	0.05	0.09
frery14	0.72	0.71	0.51	0.68	0.68	0.46	0.04	0.03	0.05
khonji14	0.60	0.58	0.35	0.41	0.50	0.20	0.19	0.09	0.15
kocher15	0.63	0.59	0.37	0.61	0.57	0.35	0.02	0.02	0.02
layton14	0.59	0.61	0.36	0.51	0.53	0.27	0.08	0.08	0.09
mezaruiz14	0.57	0.56	0.32	0.49	0.51	0.25	0.08	0.04	0.07
mezaruiz15	0.52	0.52	0.27	0.32	0.37	0.12	**0.21**	**0.16**	**0.16**
modaresi14	0.60	0.58	0.35	0.57	0.57	0.32	0.04	0.01	0.03
moreau14	0.62	0.60	0.37	0.51	0.53	0.27	0.11	0.07	0.10
moreau15	0.57	0.52	0.30	0.50	0.51	0.26	0.07	0.01	0.04
singh14	0.70	0.66	0.46	0.61	0.61	0.37	0.09	0.04	0.08
zamani14	0.58	0.55	0.32	0.48	0.49	0.23	0.11	0.06	0.09
Average							**0.09**	**0.05**	**0.08**
c) PAN14 Novels									
bagnall15	0.68	0.68	0.47	0.61	0.59	0.36	0.07	0.09	0.10
castillojuarez14	0.63	0.62	0.39	0.59	0.56	0.33	0.04	0.05	0.06
castro15	0.64	0.51	0.33	0.50	0.39	0.19	**0.14**	**0.12**	**0.13**
frery14	0.61	0.59	0.36	0.59	0.57	0.34	0.02	0.02	0.02
khonji14	0.75	0.61	0.46	0.71	0.58	0.41	0.04	0.03	0.05
kocher15	0.63	0.57	0.36	0.66	0.59	0.39	-0.03	-0.02	-0.03
layton14	0.51	0.51	0.26	0.50	0.50	0.25	0.01	0.01	0.01
mezaruiz14	0.66	0.61	0.41	0.64	0.62	0.40	0.02	0.00	0.01
mezaruiz15	0.56	0.51	0.28	0.57	0.51	0.29	-0.01	0.00	0.00
modaresi14	0.71	0.72	0.51	0.69	0.69	0.47	0.02	0.03	0.03
moreau14	0.60	0.52	0.31	0.56	0.51	0.29	0.04	0.01	0.03
moreau15	0.64	0.50	0.32	0.61	0.53	0.32	0.03	-0.03	0.00
singh14	0.66	0.58	0.38	0.63	0.56	0.35	0.03	0.02	0.03
zamani14	0.73	0.65	0.48	0.71	0.63	0.44	0.03	0.02	0.03
Average							**0.03**	**0.02**	**0.03**
d) PAN15									
bagnall15	0.81	0.76	0.61	0.72	0.71	0.51	0.09	0.05	0.10
castillojuarez14	0.64	0.64	0.41	0.55	0.55	0.30	0.09	0.09	0.11
castro15	0.75	0.69	0.52	0.72	0.68	0.49	0.03	0.01	0.03
frery14	0.54	0.46	0.25	0.47	0.43	0.20	0.07	0.04	0.05
khonji14	0.82	0.65	0.53	0.59	0.49	0.49	**0.23**	**0.16**	**0.24**
kocher15	0.74	0.69	0.51	0.72	0.66	0.48	0.02	0.02	0.03
layton14	0.67	0.50	0.34	0.49	0.50	0.25	0.18	0.00	0.09
mezaruiz14	0.65	0.61	0.40	0.55	0.54	0.30	0.10	0.07	0.10
mezaruiz15	0.74	0.69	0.51	0.55	0.53	0.29	0.19	**0.16**	0.22
modaresi14	0.40	0.41	0.16	0.39	0.40	0.16	0.01	0.00	0.00
moreau14	0.66	0.58	0.38	0.52	0.49	0.25	0.14	0.09	0.13
moreau15	0.71	0.64	0.45	0.52	0.49	0.26	0.19	0.15	0.20
singh14	0.78	0.50	0.39	0.66	0.50	0.33	0.12	0.00	0.06
zamani14	0.74	0.67	0.50	0.71	0.66	0.47	0.04	0.00	0.03
Average							**0.11**	**0.06**	**0.10**

Table 5: Results of the top verifiers of PAN 2013–2015 before and after obfuscating the four task corpora. FS (Final Score) is the product of AUC and C@1. On average, we degrade AUC by at least 10 and C@1 by about 6 percentage points on three of the corpora, though much less on the PAN14 Novels corpus. Most noticeably, we can reduce the FS of *bagnall15* (winning submission of PAN 2015) by 10–20 percentage points on all four corpora. The best obfuscation results on each corpus are marked bold. Verifiers that were improved are highlighted in red.

tion samples as per the PAN obfuscation task. We achieved an overall grade of about 2.6 (1 = excellent, 5 = fail), which places us somewhere within the top three submissions.

While the obfuscated text probably is not fit for publication, it does look promising even with our basic set of paraphrasing operators. The text was generated within a few minutes and passes the verifiers without being recognized as a same-author case. Texts from other cases look similar: a mixture of poor and good operations, where according to our own review about half of the changes made are still rather nonsensical. Since our set of operators is just a proof of concept, we will devise more sophisticated ones and better weighting schemes in future work, which is vital for achieving acceptable text quality. Promising approaches already exist, such as neural editing and paraphrasing (Grangier and Auli, 2017; Guu et al., 2017).

6 Conclusion

We introduced a promising new paradigm for authorship obfuscation and implemented a first fully functional prototype. We identified and addressed the following challenges: measuring style similarity in a manner that is agnostic to state-of-the-art verifiers, identifying those parts of a text that have the highest impact on style, and devising and analyzing a search heuristic amenable for informed search. Our study opens up interesting avenues for future research: obfuscation by addition instead of by reduction, development of more powerful, targeted paraphrasing operators, and, theoretical analysis of the search space properties.

We consider heuristic search-based obfuscation a key enabling technology that, combined with tailored deep generative models for paraphrasing, will yield better and stronger obfuscations.

References

Ahmed Abbasi and Hsinchun Chen. 2008. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Trans. Inf. Syst.*, 26(2):7:1–7:29.

Janek Bevendorff, Benno Stein, Matthias Hagen, and Martin Potthast. 2019. Bias Analysis and Mitigation in the Evaluation of Authorship Verification. In *Proceedings of ACL 2019*, (to appear).

Edward Gaylord Bourne. 1897. The authorship of the federalist. *The American Historical Review*, 2(3):443–460.

Michael Brennan, Sadia Afroz, and Rachel Greenstadt. 2012. Adversarial Stylometry: Circumventing Authorship Recognition to Preserve Privacy and Anonymity. *ACM Trans. Inf. Syst. Secur.*, 15(3):12.

Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L. Mercer. 1992. An estimate of an upper bound for the entropy of English. *Computational Linguistics*, 18(1):31–40.

Dominik Maria Endres and Johannes E. Schindelin. 2003. A new metric for probability distributions. *IEEE Trans. Information Theory*, 49(7):1858–1860.

David Grangier and Michael Auli. 2017. Quickedit: Editing text & translations via simple delete actions. *arXiv*, 1711.04805.

Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2017. Generating sentences by editing prototypes. *arXiv*, 1709.08878.

Matthias Hagen, Martin Potthast, and Benno Stein. 2017. Overview of the Author Obfuscation Task at PAN 2017: Safety Evaluation Revisited. In *Working Notes Papers of the CLEF 2017 Evaluation Labs*.

Oren Halvani, Christian Winter, and Lukas Graner. 2017. Authorship verification based on compression-models. *arXiv*, 1706.00516.

Paul G. Howard. 1993. The Design and Analysis of Efficient Lossless Data Compression Systems. *Technical Report*, CS-93-28, Brown University, 1993.

Farkhund Iqbal, Rachid Hadjidj, Benjamin C.M. Fung, and Mourad Debbabi. 2008. A novel approach of mining write-prints for authorship attribution in e-mail forensics. *Digital Investigation*, 5:S42–S51.

Patrick Juola. 2006. Authorship Attribution. *Foundations and Trends Information Retrieval*, 1(3):233–334.

Patrick Juola and Efstathios Stamatatos. 2013. Overview of the Author Identification Task at PAN 2013. In *CLEF 2013 Working Notes Papers*.

Gary Kacmarcik and Michael Gamon. 2006. Obfuscating Document Stylometry to Preserve Author Anonymity. In *Proceedings of ACL 2006*.

Dmitry V. Khmelev and William John Teahan. 2003. A repetition based measure for verification of text collections and for text categorization. In *Proceedings of SIGIR 2003*, pages 104–110.

Moshe Koppel and Jonathan Schler. 2004. Authorship Verification as a One-Class Classification Problem. In *Proceedings of ICML 2004*, pages 1–7.

Andrew W. E. McDonald, Sadia Afroz, Aylin Caliskan, Ariel Stolerman, and Rachel Greenstadt. 2012. Use Fewer Instances of the Letter "i": Toward Writing Style Anonymization. In *Proceedings of PETS 2012*, pages 299–318.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Arvind Narayanan, Hristo S. Paskov, Neil Zhenqiang Gong, John Bethencourt, Emil Stefanov, Eui Chul Richard Shin, and Dawn Song. 2012. On the feasibility of Internet-scale author identification. In *Proceedings of SP 2012*, pages 300–314.

Judea Pearl. 1984. *Heuristics - intelligent search strategies for computer problem solving*. Addison-Wesley series in artificial intelligence.

Martin Potthast, Matthias Hagen, and Benno Stein. 2016. Author Obfuscation: Attacking the State of the Art in Authorship Verification. In *Working Notes Papers of the CLEF 2016 Evaluation Labs*.

Martin Potthast, Felix Schremmer, Matthias Hagen, and Benno Stein. 2018. Overview of the Author Obfuscation Task at PAN 2018: A New Approach to Measuring Safety. In *Working Notes Papers of the CLEF 2018 Evaluation Labs*.

Josyula R. Rao and Pankaj Rohatgi. 2000. Can Pseudonymity Really Guarantee Privacy? In *Proceedings of USENIX 2000*.

D. Sculley and Carla E. Brodley. 2006. Compression and machine learning: A new perspective on feature space vectors. In *Proceedings of DCC 2006*, pages 332–332.

Efstathios Stamatatos. 2009. A Survey of Modern Authorship Attribution Methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556.

Efstathios Stamatatos, Walter Daelemans, Ben Verhoeven, Patrick Juola, Aurelio López López, Martin Potthast, and Benno Stein. 2015. Overview of the Author Identification Task at PAN 2015. In *CLEF 2015 Working Notes Papers*.

Efstathios Stamatatos, Walter Daelemans, Ben Verhoeven, Martin Potthast, Benno Stein, Patrick Juola, Miguel A. Sanchez-Perez, and Alberto Barrón-Cedeño. 2014. Overview of the Author Identification Task at PAN 2014. In *CLEF 2014 Working Notes Papers*.

Benno Stein, Matthias Hagen, and Christof Bräutigam. 2014. Generating Acrostics via Paraphrasing and Heuristic Search. In *Proceedings of COLING 2014*, pages 2018–2029.

Benno Stein, Martin Potthast, and Martin Trenkmann. 2010. Retrieving Customary Web Language to Assist Writers. In *Proceedings of ECIR 2010*, pages 631–635.

William J Teahan and David J Harper. 2003. Using compression-based language models for text categorization. In *Language modeling for information retrieval*, pages 141–165.

Wei Xu, Alan Ritter, Bill Dolan, Ralph Grishman, and Colin Cherry. 2012. Paraphrasing for style. In *Proceedings of COLING 2012*, pages 2899–2914.

Ying Zhao, Justin Zobel, and Phil Vines. 2006. Using Relative Entropy for Authorship Attribution. In *Proceedings of AIRS 2006*, pages 92–105.

Rong Zheng, Jiexun Li, Hsinchun Chen, and Zan Huang. 2006. A framework for authorship identification of online messages: Writing-style features and classification techniques. *Journal of the American Society for Information Science and Technology*, 57(3):378–393.