# Large Scale Author Obfuscation Using Siamese Variational Auto-Encoder: The SiamAO System

**Chakaveh Saedi**
Department of Computing,
Macquarie University,
Sydney, Australia
chakaveh.saedi@hdr.mq.edu.au

**Mark Dras**
Department of Computing,
Macquarie University,
Sydney, Australia
mark.dras@mq.edu.au

## Abstract

Author obfuscation is the task of masking the author of a piece of text, with applications in privacy. Recent advances in deep neural networks have boosted author identification performance making author obfuscation more challenging.

Existing approaches to author obfuscation are largely heuristic. Obfuscation can, however, be thought of as the construction of adversarial examples to attack author identification, suggesting that the deep learning architectures used for adversarial attacks could have application here. Current architectures are proposed to construct adversarial examples against classification-based models, which in author identification would exclude the high-performing similarity-based models employed when facing large number of authorial classes.

In this paper, we propose the first deep learning architecture for constructing adversarial examples against similarity-based learners, and explore its application to author obfuscation. We analyse the output for both success in obfuscation and language acceptability, as well as comparing the performance with some common baselines, showing promising results in finding a balance between safety and soundness of the perturbed texts.

## 1 Introduction

The ability of machine learning to infer information about the author of a piece of text raises issues about privacy in textual data. Blogs, reviews, even tweets can be significantly revealing when authors follow textual authorial patterns, which can lead to disclosure of sensitive information. This has led to real-world problems, such as with Amazon's machine learning-based recruitment system, which was discontinued when it turned out to disadvantage female candidates.[1] Cases like this have generated interest in NLP in concealing authorial characteristics such as gender or age, for example by producing representations that make this information difficult to infer (Li et al., 2018).

Author identification is the task of inferring the actual identity of the author. The potential number of author candidates can be very large, making author identification different from author profiling where the possible values of an attribute (e.g. gender) are typically limited to a small closed set, as in standard classification tasks. Depending on the number of included authorial classes, approaches in author identification are either classification-based or similarity-based, in the framing of Stamatatos (2009). Similarity-based approaches are proven to be better suited when facing large numbers of authors (Koppel et al., 2011), and have also underpinned several successful methods in the annual PAN authorship shared tasks[2] such as Seidman (2013) and Khonji and Iraqi (2014).

Author obfuscation is the task of concealing the identity of an author. This task is fairly challenging even for humans (McDonald et al., 2012), as authors are often not aware of hidden patterns in their writing; and the computational task is relatively underexplored. Some work has been carried out as part of a PAN authorship obfuscation task, since 2016, while other research has been independent of this. These approaches have included using backtranslation or heuristic application of paraphrase rules (Rosso et al., 2016; Hagen et al., 2017; Potthast et al., 2018), and more recently applying heuristic solution methods to the task framed as an optimization problem (Bevendorff et al., 2019; Li et al., 2019).

This work is licensed under a Creative Commons Attribution 4.0 International License. License details: http://creativecommons.org/licenses/by/4.0/.

[1] https://bit.ly/2ycdnVV

[2] https://pan.webis.de/: shared tasks that are run annually on various aspects of authorship related tasks.

Author obfuscation can be seen as the generation of adversarial examples to attack an author identification system. Work in other areas of adversarial example generation (Iyyer et al., 2018; Alzantot et al., 2018; Xiao et al., 2020; Bai et al., 2020) has seen rapid progress with the application of deep learning, and could potentially be adapted here. For example, Zhao et al. (2018b) define a GAN-style architecture to generate 'natural' adversarial examples that — unlike approaches searching the input space — works on the dense representation of each data point. Dense representations lie on the manifold that defines the data distribution and finding close points to them leads to natural adversarial examples. They apply this both to image classification tasks and a standard three-class natural language inference task, producing natural-looking adversarial examples.

Such architectures have so far only been defined for producing adversarial examples against classification-based learners (limited number of classes). In author identification, this would exclude the high-performing similarity-based approaches. In this paper we introduce SIAMAO, an architecture that can generate adversarial examples against a similarity-based learner (specifically a deep Siamese network (Saedi and Dras, 2019)) and evaluate whether it can obfuscate against authorship identification. SIAMAO draws on ideas from Variational Autoencoders (VAEs), and the specific use of them by Bowman et al. (2016) for generating novel sentences close to some input, and from the Adversarially Regularized Autoencoders (ARAEs) of Zhao et al. (2018b): the intuition here is for the autoencoder to regenerate close to the original text but with some perturbation to fool an authorship identification system.

Our main contributions are: (i) A method for integrating Siamese networks into VAEs in order to generate adversaries against similarity based models, and testing it under author obfuscation. (ii) A performance comparison on properties of the obfuscated text between our model and baselines: our focus is on how well the obfuscated text can fool an author identification system, how much the obfuscator changes the text, and how acceptable the resulting text is. We find that SIAMAO provides a promising deep learning approach to this task.

## 2 Previous Work

### 2.1 Author Identification

There has been longstanding interest in determining the identity of authors of pieces of texts. Early work has been surveyed by Stamatatos (2009), and much of the activity on the problem has been carried out in the context of PAN authorship tasks (Kestemont et al., 2019, for example). Other work has occurred outside that context, such as the high-performing CNN approach of Ruder et al. (2016).

While most approaches tackle this as a classification task using standard machine learning classifiers, this is only suitable where the number of authors is small and known in advance, as argued by Koppel et al. (2011). An alternative approach is *similarity-based* models, where a metric is used to measure similarity between texts; this is appropriate for large number of authors, which is the context of the work in the present paper. Similarity-based methods include the WritePrints method (Abbasi and Chen, 2008) and that of Koppel et al. (2011). The latter, for example, represents documents as bags of character n-grams, and measures distances between documents over repeated samples by various fixed metrics (e.g. cosine similarity, Ruzicka).

An end-to-end trainable deep learning author obfuscation architecture needs a deep learning component for author identification. A deep learning similarity-based approach to author identification has been proposed by Saedi and Dras (2019), using a Siamese network. This approach outperforms alternatives on up to 5000 authors, and is suitable for our work.

### 2.2 Author Obfuscation

Author obfuscation is a less explored area which shares interest with fields including style transfer (Prabhumoye et al., 2018) or attribute masking (Reddy and Knight, 2016). The goal is to change or perturb a text, so that the accuracy of a specific authorship inference mechanism is worsened while the modified text conveys the original message.

Early research like that of Kacmarcik and Gamon (2006) worked at the level of machine learning features, proposing to eliminate those that are more effective in classification; this, however, resulted in mostly unreadable texts. At the level of working directly with text, one approach uses *backtranslation*: input text is translated to a pivot language and translated back to the original one, producing a more or less similar text. The result is greatly affected by the availability of a successful bidirectional machine translator (Rao et al., 2000; Prabhumoye et al., 2018).

Other approaches have been largely rule-based or heuristic in nature. Most rule-based obfuscators are designed against specific techniques. The

PAN organization has included author obfuscation among the authorial tasks. The 7 participants of PAN2018 author obfuscation were also mostly rule-based, but with different levels of aggressiveness (Potthast et al., 2018), and they varied in how well they defeated inference attackers and preserved the essence of the original text. In a recent comprehensive model, Bevendorff et al. (2019) also approached obfuscation from a verification perspective. This heuristic model calculates Jensen-Shannon distance over 3-gram frequency representations, iteratively applies perturbation operators (e.g. char-flip, deletion, context-free synonymy), picks the best nodes in the search space, and continues until the original classification result changes. They proposed "operator cost" to keep the text modification minimum and as minimally disruptive as possible. This was evaluated on the relatively small datasets of the PAN tasks. Outside of the PAN context (and of NLP research in general), Li et al. (2019) proposed TextBugger, a different heuristic model that first extracts a list of most important words based on the effect they have on the classification, and then modifies the selected words.

### 2.3 Adversarial Examples

Author obfuscation can be viewed as constructing adversarial examples against an authorship identification inference attacker: this is precisely the viewpoint of TextBugger. However, as noted above, TextBugger takes a heuristic approach to this, while state of the art approaches to constructing adversarial examples in many tasks involve deep learning architectures (Iyyer et al., 2018; Alzantot et al., 2018; Xiao et al., 2020; Bai et al., 2020). And even though these are well explored in the context of continuous representations that occur in image processing, with operators like affine transformations or lighting changes, it is less straightforward for the discrete nature of text. While there is some existing work, we note that all aim to construct adversarial examples against a *classification model* that typically handles only a small number of classes.

One possibility is to use auto-encoders: Minor data distortions can be formalized as an optimization problem to minimize the classification accuracy. Such optimization has been proven successful in image processing (Biggio et al., 2013; Goodfellow et al., 2014). In the context of textual adversarial examples, approaches take ideas from a range of sources, including encoder-decoder architectures, variational auto-encoders and GANs (Kusner et al., 2017; Pu et al., 2016; Pol et al., 2019, for example). A key work that we draw on in this paper is that of Zhao et al. (2018b). Rather than working directly in the text space, they search for adversaries that lie on the data manifold: in their text application, this attacks a (three-class) textual entailment classifier. First, projections of data points are learnt, then the distance between each adversary and the closest real data point is measured in the vector space to choose the best fake sample. Finally, the selected adversary is mapped back to the input space. Their system combines ideas from encoder-decoder architecture, VAEs and GANs, and has two main training objectives: (1) bringing the encoder and generator output close to each other; and (2) making the sampled noise (i.e. generator's input) less random by using a module they call the 'Inverter'. The inverter is a network that learns to sample close-to-input points in the data manifold. Their search algorithm identifies the best adversary by incrementally increasing the search space till the classification result of the sampled point(s) is different from that of the original input data.

While not explicitly cast as adversarial example generation, the process of paraphrase generation can be seen in this light. Gupta et al. (2018) proposed a VAE-LSTM containing 2 LSTM-encoders which encode both the original sentence and the paraphrase. Encoded vectors are used in the sampling process of the VAE. On the decoder side, there is an encoder for original sentences and a decoder for paraphrase generation that is fed the embedding vector and the encoder output. In our approach, our encoder is a CNN but we also use two encoded vectors for sampling, and the modified embeddings are used by the decoder.

An optimization-based alternative to these deep learning approaches was proposed by Alzantot et al. (2018), using population-based optimization. They encode the sentences and perturb them in the vector space. Unlike the above work, they propose a gradient-free optimization by employing genetic algorithms. Perturbation is at the word level based on semantic similarity of candidates and original vectors going through cross-over and mutation instead of expanding the search space iteratively. We use a similar notion of perturbation operators, including cross-over.

## 3 SiamAO

Here we present SiamAO, an author obfuscation neural network that integrates a large scale Siamese author identifier in a VAE architecture to generate

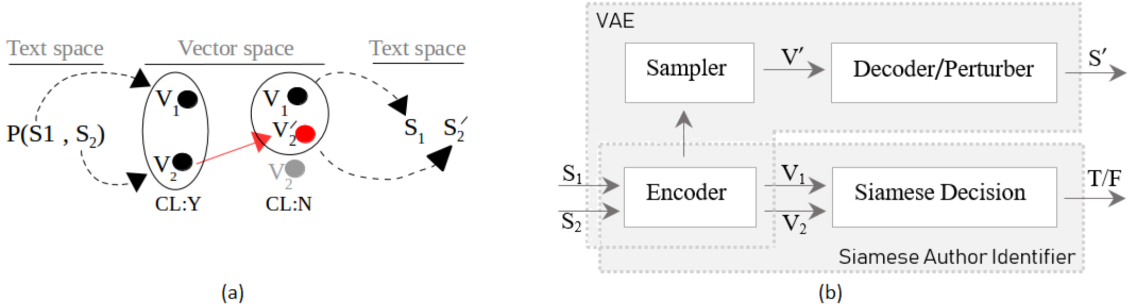

Figure 1: (a) A pair of texts $(S_1, S_2)$ are mapped into the vector-space $(V_1, V_2)$ . Perturbation operator (continues red arrow) modifies one of the inputs. This changes the verification result from $Y$ to $N$. $S_2'$ is the perturbation output when mapped back to the text space. (b) Schematic view of SiamAO; the network is composed of a Siamese author identifier and a VAE which share an encoder.

adversarial text. This system takes a pair of texts as input and generates an adversary with the aim of changing the author identification results. Figure 1 shows (a) a high-level schema of the process and (b) the components of SIAMAO respectively. A key innovation is incorporation of a similarity-based author identification approach, in contrast to other work described in §2 that only constructs adversarial examples against classification-based inference.

### 3.1 Author Identification

Our similarity-based author identification component is taken from Saedi and Dras (2019). The model consists of (1) a dual encoding sub-network and (2) a decision sub-network. The encoding sub-network (a deep CNN model) receives an input pair of texts $(S_1, S_2)$ and maps each $S_i$ into the vector space $(V_i)$. The decision sub-network compares $V_1$ and $V_2$ and generates a similarity score (more information available in supplementary material). We adopt the version of the model that proved best overall in the source work for large numbers of authors: the encoding sub-networks are character-level rather than word-level, and $L_1$ distance is employed in the decision network. This similarity-based model produces a score between a pair of texts that can be interpreted as an answer to the *author verification* problem: are these two texts by the same author? These two components of the author identification system are marked as *Encoder* and *Siamese Decision* in Figure 1-(b) respectively.

### 3.2 Author Obfuscation

Our overall approach to generating adversarial examples draws on the VAE architecture of Gupta et al. (2018) for paraphrase generation, and the idea of Zhao et al. (2018a) to generate perturbations in the encoded space. Implementation details can be found in supplementary material.

**Encoder-Decoder** A successful VAE for text perturbation requires a strong encoder as well as a decoder capable of perturbation. In our proposed architecture, shown in Figure 1-(b), the author identifier network and the VAE share the encoder. This results in an *authorial feature aware* decoder since the encoder is trained on author verification. SIAMAO's decoder is trained for a) normal decoding (i.e. as a decoder: loss is 0, when input=output= target) and for b) obfuscation (i.e. as a perturber: loss is 0, when input≠output=target). In both cases, the input to the decoder is sampled from $V_i$. However, when trained for obfuscation, perturbation operators modify the sampler's input and output.

**Sampler** In finding adversarial examples, we have two aims: (1) like Zhao et al. (2018b), we look for points that lie close to the original in terms of the manifold that defines the data distribution; and (2) we look for adversaries that can change the author verification results while preserving the original message. In other words, we need to generate a piece of text that is very close to the original one but different enough to change the verification result. In SiamAO, when training the decoder for normal decoding, $V_i$ is directly used for sampling (i.e. to generate $V'$ from the normal distribution). However, when training the decoder for obfuscation, unlike non-Siamese models, we have access to two sample inputs $(V_1, V_2)$ which can help to remain within the acceptable area[3] in the vector space. We start by interpolating between these two inputs. Specifically, if $V_1 = [v_{11}, \ldots, v_{1n}]$ and $V_2 = [v_{21}, \ldots, v_{2n}]$, and $V_1 > V_2$ and the distance between them is is

[3] There are infinite data points in the vector space, not all of them can be mapped back to a meaningful piece of text; an acceptable area in the vector space has similar distribution to the input space.

$d$, either of $V_1' = [v_{11} - d', \ldots, v_{1n} - d']$ and $V_2' = [v_{21} + d', \ldots, v_{2n} + d']$ (where $d' = d/3$) can be used by the sampler.[4]

**Perturbation Operators** To combine embedding and sampled vectors in the decoding step, we could concatenate them as in most VAE models. There is a risk, however, that the network focuses on the embedding part and mostly ignores the sampled vector which results in very few changes in the text such that it is unable to mislead the classifier. To add perturbations to the vectors, we adopt some of the techniques of Alzantot et al. (2018). In SiamAO, when training the decoder for obfuscation, after moving the original vector and sampling as explained above, we use *cross-over* as the final perturbation step. Cross-over, taken from genetic algorithms, keeps vector elements mostly the same, only making changes at specific indices. The inputs to the cross-over operator are the $V'$ vector and the character-level embedding.[5] After cross-over, we sum the two vectors. Alternative methods are compared in §5.

**Objectives** The first part of the objective is a standard one for VAEs, the reconstruction loss, in Eqn (1). In terms of generating adversarial examples, training our generative model consists of (1) training for normal decoding and (2) training for perturbation. In the latter, the decoder learns to make changes to the input and the sampler learns to pick a vector that flips the Siamese author verification original (binary) decision ($y_{\text{OD}}$) to the perturbed decision ($y_{\text{PD}}$), in Eqn (2). Eqn (3) combines these two component losses.

$$l_{\text{cons}} = (\mathbf{E}_{q_\phi(V'|S)}[\log p_\theta(S|V')] - \text{KL}(q_\phi(V'|S) \parallel p(V')) \tag{1}$$

$$l_{\text{sampler}} = \text{MSE}(y_{\text{PD}}, |1 - y_{\text{OD}}|) \tag{2}$$

$$l_{\text{pert}} = \alpha \times l_{\text{cons}} + (1 - \alpha) \times l_{\text{sampler}} \tag{3}$$

Eqn (1) provides a lower bound on the model evidence $p(S|\theta, \phi)$, KL stands for Kullback–Leibler divergence. $\alpha$ is set to 0.5 in all our experiments, making the backpropagation uniform on the sampler and the decoder.

**Obfuscation Training** For the perturbation objective, we generate training data by applying widely used text modification operators very similar to rule-based systems such as Bevendorff et al. (2019) and Li et al. (2019). We emphasise that unlike common rule-based or heuristic techniques, these operators are merely to generate the data entries as the target while training the decoder for obfuscation. Our selected modification rules can be categorized into four classes: *shape similarity* (e.g. ä →a, O → 0), *sound similarity* (e.g. ee → ea), *swap* (e.g. ie → ei) and *punctuation modification* (e.g. . → .. or :” → :). As Bevendorff et al. (2019), we only apply these changes to a subset of instances in each text piece, which we select uniformly randomly with probability 1/3.

## 4 Experimental Setup

### 4.1 Evaluation Framework

There is not yet a standard evaluation framework for this kind of work. Hence we observe various different evaluation techniques in the literature. This has also resulted in project specific definitions. For instance, in both PAN2018 and the Text-Bugger system, mis-spelled words are considered as valid “paraphrasing” due to the little impact they cause on human understanding. They argue character-level perturbation (i.e. mis-spelled words) are visually and semantically similar to the original ones (e.g. *their* and *thier*, *some* and *s0me*) and can deliver the original message (Potthast et al., 2018; Li et al., 2019; Rawlinson, 2007). Work on adversarial example attacks has two broad types of evaluation. Misclassification or attack success (how well the adversarial examples fool the inference mechanism); and utility or imperceptibility (how well the adversarial examples preserve important aspects of the original). Work on author obfuscation generally fits with this, although in disparate ways; the PAN tasks,[6] for example, consider safety (broadly misclassification), soundness (textual entailment between original and adversarial texts) and sensibleness (inconspicuousness, or looking like regular text); the latter two are related to the typical utility criteria. Working on large authorial classes, we could not employ the exact set-up in PAN evaluation, however, our evaluation metrics also assess misclassification and utility.

#### 4.1.1 Misclassification

We calculate “Perturbation Wins” ($PW$): the average proportion of times where a perturbed vector or text misleads an authorship identification inference model (Alzantot et al., 2018; Potthast et al., 2018).

**Robust Vector Representation** We first look at a system-internal evaluation. As noted above, the

[4] We conducted experiments with the average vector and the $1/3$ distance shift as explained here. We leave finding the best interpolation for further study.

[5] Specifically, we apply five crossovers between the $V'$ and the embedding vectors at random positions.

[6] https://pan.webis.de/clef18/pan18-web/author-obfuscation.html

system objective is to generate a vector representation which is similar to the original message while eliminating clues to authorship. Having $(S_1, S_2)$ as an input pair with $(V_1, V_2)$ as their corresponding representations in vector space, $V_i$ is perturbed to $V_i'$ which is then sent back to Siamese decision by replacing $V_i$. It shows whether the Siamese author verification's original decision on $(V_1, V_2)$ is different from the decision on $(V_1, V_2')$ and $(V_1', V_2)$. This gives a preliminary result: if the system cannot produce vectors that can fool the decider, it will not produce successfully perturbed texts.

**Perturbation Win in Text Space** The author identification work of Saedi and Dras (2019) had as its primary evaluation, following the first work on deep Siamese networks (Koch et al., 2015), $N$-way one-shot classification: a 'query' text is compared against texts by $N$ authors, one of whom is also the author of the query text. The $N$-way task is tackled by assigning pairwise similarities to the query text and each author, in effect carrying out $N$ author verification attempts. $N$-way inference performance is evaluated by the average accuracy over 150 $N$-way classifications. We consider $N \in \{3, 5, 10, 50\}$.

Our misclassification evaluation in text space involves calculating perturbation wins on both author verification and $N$-way classification. In the $N$-way evaluation, a perturbed query text is presented. We use two authorship inference models for this: the standalone Siamese authorship identification system of Saedi and Dras (2019), and the system of Koppel et al. (2011). This latter is a key inference attacker in PAN tasks, and also the only similarity-based system with available code. Koppel works on iterative representation of pieces of text using a subset of all extracted character 4-grams and similarity measurements (Ruzicka metric) to identify the author of a piece of text (Koppel et al., 2011).

In addition to the $N$-way evaluation above, we evaluated misclassification under Koppel with 1000 authors, randomly selected from SIAMAO's testset. (Koppel does not require training, apart from counting character n-grams, and so is fast to use for many authors.) In the results we call this setup K-LG.

#### 4.1.2 Utility: Text similarity

We use the following measures to quantify the similarity between original and perturbed texts. (1) Bleu score (BL) (Papineni et al., 2002), measuring n-gram overlap between original and generated texts, previously used to assess difference in style transfer (Shen et al., 2017). (2) Edit distance (ED), considering the texts as strings and counting the minimum number of operations required to transform the original texts into their perturbed counterparts (Przybocki et al., 2006; Li et al., 2019). This metric is believed to be used in commercial translation memory models (Bloodgood and Strauss, 2014). (3) Euclidean Distance (EC) between the vector representations: closeness in vector space typically corresponds to greater semantic similarity (Li et al., 2019; Alzantot et al., 2018).

#### 4.1.3 Utility: Language acceptability

The perturbed text should be natural-looking, in terms of grammaticality / acceptability. Prediction of language acceptability is now a standard NLP task, e.g. the CoLA task that is part of the GLUE benchmark (Wang et al., 2019). However, that is a binary task: sentences are judged acceptable or not. There is, instead, a notion of gradient grammaticality, where sentence grammaticality is measured on a scale of 0 to 1 (Lau et al., 2014); this could be more suited to capturing the changes we might see in our adversarial examples.

BERT has previously been fine-tuned to produce a high-performing model for the CoLA task (Devlin et al., 2019). For gradient grammaticality, a variety of models predating BERT have been trained on the Statistical Models of Grammaticality (SMOG) dataset,[7] and have been shown to correlate fairly well with human judgements (Lau et al., 2014, 2017). Given the improvements over earlier models shown by BERT on the CoLA task, we built our model of language naturalness by fine-tuning BERT-large on the SMOG dataset. We refer to this model as BERT-SMOG. To validate our BERT-SMOG, we compare with models proposed in Lau et al. (2017) on the original dataset: its Pearson's $r$ correlation with human judgements is around 0.8, much higher than their best scoring model (which predates contextual LMs).

In this evaluation category, we also provide the scores for the more common binary acceptability. For this, we fine-tuned BERT only on the CoLA dataset (BERT-CoLA). Evaluating BERT-CoLA on CoLA testset, our results are in line with the published benchmarks (Devlin et al., 2019). Final evaluations are done on a subset of 700 randomly selected sentences from the Fanfiction database going through backtranslation, RAND modification and SIAMAO.

[7] Project website: `https://clasp.gu.se/about/people/shalom-lappin/smog`.

### 4.2 Data

Several datasets have been used for author identification, including various PAN datasets. We use the dataset from Saedi and Dras (2019) consisting of 10000 authors from the domain of fanfiction,[8] as one that is large enough to train a deep learning system. We followed the FF-5K (5000 author) dataset setup under the one-shot evaluation (i.e. disjoint authors between train and test sets). This test set consists of over 10000 pairs covering 1665 authors not seen in training (more information in the supplementary material).

### 4.3 Models

**Core Models** As in a VAE, our SIAMAO system starts with text that looks somewhat random, and as training proceeds comes to look more like the original text, encouraged by the reconstruction loss. At each epoch, then, there will be varying effects on misclassification and utility. Training the model for 6 epochs, we present results for both epoch 3 (SIAMAO$_3$) and epoch 5 (SIAMAO$_5$) to show the effect training has on different aspects of text modification with opposing objectives.

**Baselines** The author obfuscation approaches of the PAN competition are typically tailored to the PAN setup (classification-based, over a relatively small number of authors). Heuristic-based approaches are potentially applicable, but could not be applied here.[9]

We therefore used backtranslation as our key baseline, as one that has recently produced decent results in related tasks (Prabhumoye et al., 2018). Our experiments are done on two sets of languages with different accuracy in Google machine translation, English-French (BT-FR: good quality MT) and English-Persian (BT-PR: average-high MT). Random character modification (RAND), following the same rules explained in §3.2, is another baseline.

**Variant Models** To examine the effect of choices in the architecture (in particular, in §3.2 under Perturbation Operators), we explored various ways of transferring the encoder's outputs to the sampler and generating the input to the decoder. The encoder generates two vectors, $V_1$ and $V_2$. These vectors can be directly sent to the sampler (e.g. JUSTSUM method below), or go through some changes in the vector space before being fed to the sampler (e.g. SHIFT and AVE below). The sampler uses its input vector to sample a similar point ($V'$) from the normal distribution, which is then sent to the decoder. The decoder needs both $V'$ and embedding to generate an output sentence.

The five methods we report are 1) SHIFT (the core method we define in §3): $V_1$ and $V_2$ are shifted towards each other by $1/3$ of their distance; the resulting vectors are sent to the sampler. 2) JUSTSUM: $V_i$ is the input to the sampler. 3) AVE: the element-wise average of $V_1$ and $V_2$ is the input to the sampler. In all these three methods the sum over cross-over between embedding vector and $V'$ is the input to the decoder. For both 4) CATEMB and 5) NOCROSS, the first step is the same as the SHIFT method. Then, in the former, the concatenation of embedding and $V'$ is the input to the decoder; in the latter sum of embedding and $V'$ is the input to the decoder.

## 5 Evaluation Results and Analysis

### 5.1 Misclassification

**Robust Vector Representation** Replacing vectors with their perturbed version as explained in §4.1.1 changes the inputs to the Siamese Decision sub-network (e.g $(V_1, V_2) \rightarrow (V_1, V_2')$). This modification results in PW of over $90\%$, indicating authorial information can be hidden in vector space using SIAMAO.

**Perturbation Win in Text Space** In terms of the classification across a large number of authors, K-LG in Table 1 shows that Koppel's accuracy of $0.644$ over 1000 authors drops dramatically under all modifications. SIAMAO$_3$ causes the maximum fall in accuracy, RAND ranks second, followed by BT-PR. For SIAMAO, as expected, at epoch 5, where the VAE-style architecture has reconstructed the perturbed text to be closer to the original, the drop in classification accuracy is smaller.

The two middle columns in Table 1 show the accuracy on original and perturbed data for $N$-way classification ($N \in \{3, 5, 10, 50\}$). We see different behaviour across the two author identifiers and under different $N$s. Koppel classification accuracy decreases with all methods, with one of the SIAMAO methods generally best. None of the methods — SIAMAO, backtranslation, or random changes — seem to be effective against the Siamese author identifier, which is rather surprising. However, in one way these results are in line with what Zhao et al. (2018b) reported: success rate is noticeably

[8] https://github.com/ChakavehSaedi/Siamese-Author-Identification.

[9] TextBugger (Li et al., 2019) does not have an available associated code. Bevendorff et al. (2019) do helpfully provide code, but we could not get it to work for our setup.

|  | Koppel Author Identification |  |  |  | Siamese Author Identification |  |  |  | K-LG |
|---|---|---|---|---|---|---|---|---|---|
| Model | 3-way | 5-way | 10-way | 50-way | 3-way | 5-way | 10-way | 50-way |  |
| Original | 0.640 | 0.567 | 0.427 | 0.327 | 0.933 | 0.853 | 0.707 | 0.400 | 0.644 |
| SIAMAO$_3$ | 0.513 | 0.493 | 0.353 | 0.260 | 0.913 | 0.867 | 0.773 | 0.433 | 0.407 |
| SIAMAO$_5$ | 0.540 | 0.487 | 0.360 | 0.220 | 0.940 | 0.873 | 0.760 | 0.433 | 0.446 |
| RAND | 0.593 | 0.513 | 0.400 | 0.240 | 0.933 | 0.900 | 0.793 | 0.507 | 0.414 |
| BT-FR | 0.613 | 0.526 | 0.433 | 0.273 | 0.827 | 0.740 | 0.573 | 0.353 | 0.558 |
| BT-PR | 0.607 | 0.500 | 0.340 | 0.293 | 0.913 | 0.847 | 0.733 | 0.413 | 0.429 |

Table 1: First two columns, Koppel and Siamese author identification accuracy on $N$-way classification. K-LG shows Koppel accuracy on 1000 authors.

| Model | PW | EC | ED | BL |
|---|---|---|---|---|
| SIAMAO$_3$ | 0.238 | 2597 | 289 | 0.098 |
| SIAMAO$_5$ | 0.375 | 1927 | 215 | 0.168 |
| RAND | 0.340 | 3442 | 222 | 0.070 |
| BT-FR | 0.399 | 4649 | 235 | 0.486 |
| BT-PR | 0.512 | 4891 | 427 | 0.256 |

Table 2: Perturbation win (PW), Euclidean distance (EC), edit distance (ED), and Bleu score (BL), comparing perturbed text against the original.

Rosie laughs as my ~~cheeks~~ cheaks burn, and I watch as she stands elegantly, blocking the sun from my eyes. 'I want to go swimming,' she ~~tells~~ tel1s me.

I felt my ~~cheeks~~ cheaks ~~flush~~ flu5h slight1y. "Sh...well, uh..." Sam and Embry chuckled at my ~~response~~ resp0n5e .

Figure 2: Samples of SIAMAO's perturbation that successfully fooled classification.

lower when the classifier (i.e. Siamese author identifier in our case) is stronger.

For the binary classification task of author verification that underpins the classification across all authors and $N$-way classification, we give some results under PW in Table 2. It is interesting that while the proportion of perturbation wins in the verification context is relatively low, it still results in noticeable drops in the overall classification scores for Koppel as noted above. This is likely to be because the similarity scores are changed enough to affect the selection among $N$ authors while not changing the pairwise binary prediction.

## 5.2 Text Similarity

Table 2 provides the Bleu scores, edit and Euclidean distances in the verification task, under random, back-translation and SIAMAO modifications. For our two variants of SIAMAO, SIAMAO$_3$ results in more modifications than SIAMAO$_5$, reflecting the nature of VAEs. However, due to the other objective of the network, training must improve perturbations too. We observe higher perturbation win as well as higher Blue score for SIAMAO$_5$. Given the fact that Blue score is calculated on word n-grams, this suggests the model may have learnt to modify texts mostly at spaces that do not break words (e.g. punctuation modification).

In terms of the baselines, BT-PR and BT-FR result in more modifications than RAND (higher edit and Euclidean distances). However, they achieve the highest Bleu score as well as perturbation win. SIAMAO ranks in the middle, with SIAMAO$_5$ showing the least text modification, being significantly more successful than RAND in all the four metrics but less successful in perturbation win and Bleu compared to the back-translation models.

In Figure 2 we give two sample extracts of perturbed texts from SIAMAO that fooled classifiers, to illustrate how the system changes text. It can be seen that the perturbation operators described in §3.2 are applied only at some places: for example, the replacement of *s* by *5* does not occur at all possible locations, and similarly *l* by *1*.

**Training and finding a balance** An obfuscation model has several objectives that contradict each other. So, the network learning process involves finding a balance between them; specifically, finding important positions in the input text to minimally modify, as well as improving obfuscation success. Using SIAMAO's test set after each training epoch, we evaluated the 4 aforementioned parameters. Figure 3 displays the trends for edit distance, Euclidean distance, Bleu and perturbation win follow during SIAMAO's 6 training epochs.

Epochs 1 to 3 present rather sharp upward trends for edit distance, Euclidean distance and perturbation win, coinciding with an expected major drop in Bleu. Epochs 3 to 5, on the other hand, show Bleu score increasing to its maximum in epoch 4 while edit and Euclidean distance experience a noticeable fall. Epoch 5 reaches an favorable balance in the parameters plus the most successful modification from a privacy point of view. However, this doesn't continue in epoch 6 which is an indicator of over-training.

|  | Original | BT-FR | BT-PR | SIAMAO3 | SIAMAO5 | RAND |
|---|---|---|---|---|---|---|
| BERT-SMOG | 0.751 | 0.733 | 0.724 | 0.522 | 0.535 | 0.506 |
| BERT-CoLA | 0.773 | 0.788 | 0.796 | 0.485 | 0.549 | 0.567 |
| # OOVs | 16.7 | 14.5 | 9.6 | 76.4 | 73.5 | 95.7 |

Table 3: Language acceptability scores on a subset of original and perturbed Fanfiction data. Also included are average number of OOV tokens in texts.

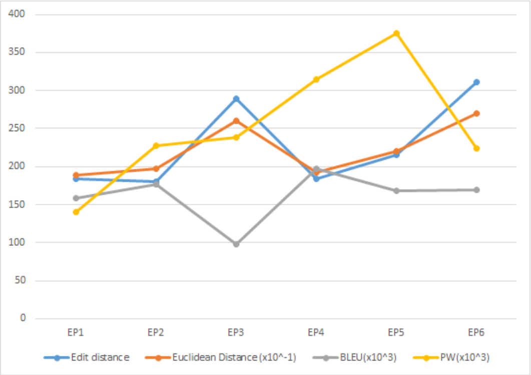

Figure 3: Effect of 6 epochs of training on edit distance, Euclidean distance, Bleu and perturbation win.

### 5.3 Language Acceptability

Table 3 presents the results for language acceptability as measured by BERT-SMOG and BERT-CoLA on a subset of perturbed Fanfiction database. One issue with applying these models to the obfuscated text is that SIAMAO is more likely to generate out-of-vocabulary (OOV) words (e.g. *cheaks*) than the backtranslation models, and this affects the acceptability score, even if the OOV words themselves might be considered reasonable. The table thus also contains average number of OOV tokens in generated texts.

The scores on the original are relatively high; the scores on the backtranslation models are close. This is not surprising given that number of OOV tokens is similar (in fact, it is surprising that number of OOV tokens is actually lower than in the original texts, even more for BT-PR than for BT-FR— perhaps the original OOVs are lost in translation). The average number of OOVs is much larger for the SIAMAO models and RAND. To understand the effect of number of OOVs, we took the original CoLA dataset and systematically replaced words with OOV tokens, and carried out some curve fitting of number of OOVs against BERT-CoLA score; this would allow us to estimate how a score might drop with an increasing number of OOVs. An exponential decay function appears to be a good fit. However, because CoLA sentences are much shorter than the generated texts, it is not possible to use such a curve for direct extrapolation.[10] Nevertheless, it does illustrate that it is not surprising for the language acceptability scores to be lower for the SIAMAO models, and that this is not necessarily indicative of substantially worse quality.

### 5.4 Variant Models

As noted in §4.3, we studied the effect of different ways of transferring the encoder's outputs to the sampler, beyond just a standard concatenation as in regular VAEs. SHIFT approach outperforms the other variants in most respects (misclassification, etc) while being similar in the text similarity measures (edit and Euclidean distance). This supports the intuition that regular VAE concatenation is not sufficient for this task, and perturbation operators of the sort we have proposed are necessary (scores are included in the supplementary material).

## 6 Conclusion and Further Work

This work is the first to propose a deep learning architecture for generating textual adversarial examples that incorporates a similarity-based inference model rather than a standard classifier-based one. We explored this in the context of authorship obfuscation, where the goal is to hide the author from a similarity-based authorship identifier. Results indicate that our SIAMAO model can degrade the performance of a key standard authorship identification system, compared to baseline systems, with modifications that are of similar magnitude or lower. All approaches had difficulty against a Siamese authorship identification system, however.

As this is the first work in this direction, many improvements are possible, particularly in the area of language acceptability. These improvements would be both to SIAMAO, in encouraging the adversarial examples towards greater acceptability, also in terms of the automatic evaluation metrics. Employing other deep learning adversarial architectures as a base would also be interesting.

[10] Adjusting the scores on the original texts to match the number of OOVs in the SIAMAO and RAND models leads to values close to the curve asymptote, of around 0.3.

## References

Ahmed Abbasi and Hsinchun Chen. 2008. Writeprints: A Stylometric Approach to Identity-Level Identification and Similarity Detection in Cyberspace. *ACM Transactions on Information Systems*, 26(2).

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Tao Bai, Jun Zhao, Jinlin Zhu, Shoudong Han, J. Chen, and Bo Li. 2020. Ai-gan: Attack-inspired generation of adversarial examples. *ArXiv*, abs/2002.02196.

Janek Bevendorff, Martin Potthast, Matthias Hagen, and Benno Stein. 2019. Heuristic authorship obfuscation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1098–1108.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. 2013. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer.

Michael Bloodgood and Benjamin Strauss. 2014. Translation memory retrieval methods. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 202–210, Gothenburg, Sweden. Association for Computational Linguistics.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 NAACL-HLT*. Association for Computing Machinery.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Matthias Hagen, Martin Potthast, and Benno Stein. 2017. Overview of the author obfuscation task at pan 2017: Safety evaluation revisited. In *CLEF*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Gary Kacmarcik and Michael Gamon. 2006. Obfuscating document stylometry to preserve author anonymity. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 444–451. Association for Computational Linguistics.

Mike Kestemont, Efstathios Stamatatos, Enrique Manjavacas, Walter Daelemans, Martin Potthast, and Benno Stein. 2019. Overview of the cross-domain authorship attribution task at {PAN} 2019. In *Working Notes of CLEF 2019-Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019*, pages 1–15.

Mahmoud Khonji and Youssef Iraqi. 2014. A Slightly-modified GI-based Author-verifier with Lots of Features (ASGALF). In *Working Notes for CLEF 2014 Conference*.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2.

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

Matt J. Kusner, Brooks Paige, and José Miguel Hernández-Lobato. 2017. Grammar variational autoencoder. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1945–1954. JMLR.org.

Jey Han Lau, Alexander Clark, and Shalom Lappin. 2014. Measuring gradience in speakers' grammaticality judgements. In *CogSci*.

Jey Han Lau, Alexander Clark, and Shalom Lappin. 2017. Grammaticality, acceptability, and probability: A probabilistic view of linguistic knowledge. *Cognitive science*, 41 5:1202–1241.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. TextBugger: Generating Adversarial Text Against Real-world Applications. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS)*.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30, Melbourne, Australia. Association for Computational Linguistics.

Andrew WE McDonald, Sadia Afroz, Aylin Caliskan, Ariel Stolerman, and Rachel Greenstadt. 2012. Use fewer instances of the letter ”i”: Toward writing style anonymization. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 299–318. Springer.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Adrian Pol, Victor Berger, Gianluca Cerminara, Cécile Germain, and Maurizio Pierini. 2019. Anomaly detection with conditional variational autoencoders. In *18th IEEE International Conference on Machine Learning and Applications*. ICMLA.

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 *CLEF (Working Notes)*.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation.

Mark Przybocki, Gregory Sanders, and Audrey Le. 2006. Edit distance: A metric for machine translation evaluation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. 2016. Variational autoencoder for deep learning of images, labels and captions. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2352–2360. Curran Associates, Inc.

Josyula R Rao, Pankaj Rohatgi, et al. 2000. Can pseudonymity really guarantee privacy? In *USENIX Security Symposium*, pages 85–96.

G. Rawlinson. 2007. The significance of letter position in word recognition. *IEEE Aerospace and Electronic Systems Magazine*, 22(1):26–27.

Sravana Reddy and Kevin Knight. 2016. Obfuscating gender in social media writing. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 17–26.

Paolo Rosso, Francisco M. Rangel Pardo, Martin Potthast, Efstathios Stamatatos, Michael Tschuggnall, and Benno Stein. 2016. Overview of pan'16 - new challenges for authorship analysis: Cross-genre profiling, clustering, diarization, and obfuscation. In *CLEF*.

Sebastian Ruder, Parsa Ghaffari, and John G Breslin. 2016. Character-level and multi-channel convolutional neural networks for large-scale authorship attribution. *arXiv preprint arXiv:1609.06686*.

Chakaveh Saedi and Mark Dras. 2019. Siamese networks for large-scale author identification. *arXiv preprint arXiv:1912.10616*.

Shachar Seidman. 2013. Authorship Verification Using the Imposters Method. In *Working Notes for CLEF 2013 Conference*.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yatie Xiao, Chi-Man Pun, and Bo Liu. 2020. Adversarial example generation with adaptive gradient search for single and ensemble deep neural network. *Information Sciences*, 528:147–167.

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. 2018a. Adversarially regularized autoencoders. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5902–5911, Stockholmsmässan, Stockholm Sweden. PMLR.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018b. Generating natural adversarial examples. In *International Conference on Learning Representations (ICLR)*.