

# Attribution and Obfuscation of Neural Text Authorship: A Data Mining Perspective

Adaku Uchendu  
Penn State University  
PA, USA  
azu5030@psu.edu

Thai Le  
University of Mississippi  
MS, USA  
thaile@olemiss.edu

Dongwon Lee  
Penn State University  
PA, USA  
dongwon@psu.edu

## ABSTRACT

Two interlocking research questions of growing interest and importance in privacy research are *Authorship Attribution* (AA) and *Authorship Obfuscation* (AO). Given an artifact, especially a text  $t$  in question, an AA solution aims to accurately attribute  $t$  to its true author out of many candidate authors while an AO solution aims to modify  $t$  to hide its true authorship. Traditionally, the notion of authorship and its accompanying privacy concern is only toward *human* authors. However, in recent years, due to the explosive advancements in Neural Text Generation (NTG) techniques in NLP, capable of synthesizing human-quality open-ended texts (so-called “neural texts”), one has to now consider authorships by humans, machines, or their combination. Due to the implications and potential threats of neural texts when used maliciously, it has become critical to understand the limitations of traditional AA/AO solutions and develop novel AA/AO solutions in dealing with neural texts. In this survey, therefore, we make a comprehensive review of recent literature on the attribution and obfuscation of neural text authorship from a Data Mining perspective, and share our view on their limitations and promising research directions.

## 1. INTRODUCTION

**Natural Language Generation (NLG)** is a broad term for AI techniques to produce high-quality human-understandable texts in some human languages, and often encompasses terms such as machine translation, dialogue generation, text summarization, data-to-text generation, Question-Answer generation, and open-ended or story generation [72, 119]. Among these, in particular, this survey focuses on the open-ended text generation aspect of NLG. Since the advent of the Transformers architecture in 2018, the field of NLG has experienced exponential improvement. Before 2018, leading NLG models were only able to generate a few sentences coherently. However, after adopting the Transformer architecture into deep learning-based Language models (LMs), NLG models could generate more than a few sentences (i.e.,  $\geq 200$  words) coherently. GPT-1 [92] by OpenAI is one of the first such NLG models. Since then, many other Transformer-based LMs with the capacity to generate long coherent texts have been

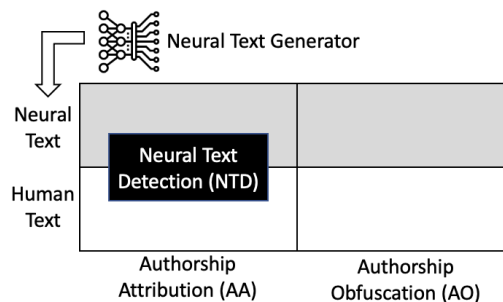


Figure 1: The figure illustrates the quadrant of research problems where (1) the **GRAY** quadrants are the focus of this survey, and (2) The **BLACK** box indicates the specialized binary AA problem to distinguish neural texts from human texts.

released (e.g., FAIR [16, 82], CTRL [59], PPLM [25], T5 [94], Wu-Dao <sup>1</sup>). In fact, as of February 2023, huggingface’s [113] model repo houses about 8,300 variants of text-generative LMs<sup>2</sup>. In this survey, we refer to these LMs as **Neural Text Generator (NTG)** since they are neural network-based LMs with text-generative abilities. Further, we refer to the texts generated by NTG as “**neural**” **texts**<sup>3</sup>, as opposed to normal texts written by humans as **human texts**.

As the qualities of NTGs improve, neural texts become more easily misconstrued as human-written [18, 45, 52, 108, 117], exacerbating the difficulty of distinguishing neural texts from human texts. For instance, therefore, such a text generation capability can be misused to generate misinformation [13, 14, 117], fake reviews [3] and political propaganda [109] at scale with little cost. These problems lead to the need to effectively distinguish neural texts from human texts, the so-called **Neural Text Detection (NTD)** problem, which is a sub-problem of a widely studied problem in the privacy community—i.e., authorship attribution. In fact, two interlocking research questions in privacy research, heavily studied but of growing interest, are **Authorship Attribution (AA)** and **Authorship Obfuscation (AO)**. Given an artifact, especially a text  $t$  in question, an AA solution aims to accurately at-

<sup>1</sup><https://github.com/BAAI-WuDao>

<sup>2</sup>[https://huggingface.co/models?pipeline\\_tag=text-generation](https://huggingface.co/models?pipeline_tag=text-generation)

<sup>3</sup>Other names for neural text include *AI(-generated) text* [66], *Machine(-generated/written) text* [1, 5, 36, 38, 45, 89, 96, 104, 107, 108, 117], *Artificial text* [67], *Computer-generated text* [102], *Deepfake text* [91], *Auto-generated text* [85], and *Synthetic text* [13, 27, 46, 81].

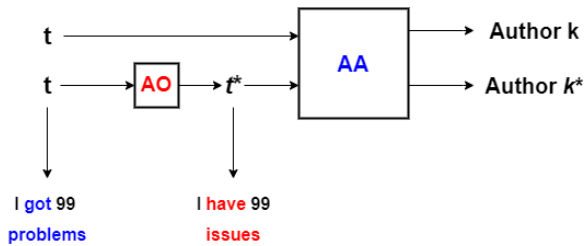


Figure 2: Illustration of both AA and AO problems on neural texts

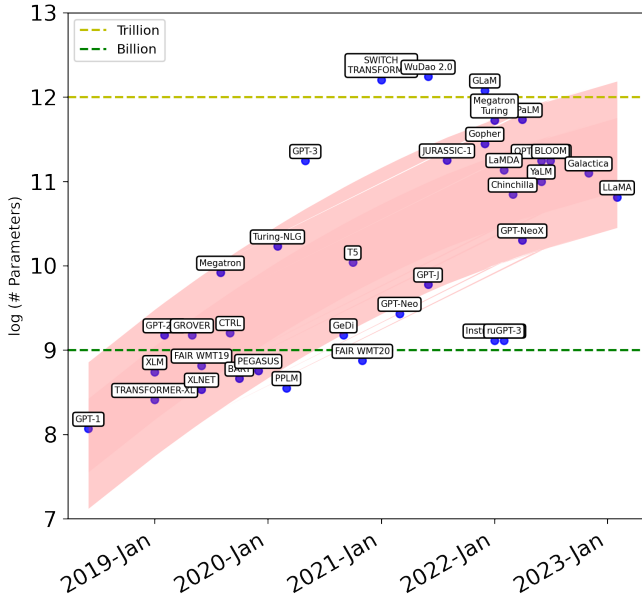


Figure 3: Evolution of Neural Text Generators (NTGs) from 2018 to 2023 (Y-axis is a log plot of # of parameters).

tribute  $t$  to its true author out of  $k$  candidate authors while an AO solution aims to modify  $t$  to hide its true authorship. Therefore, NTD is a specialized case, a Turing Test, of AA with  $k = 2$  authors (i.e., human vs. machine). Figure 1 illustrates the quadrant of research problems, while Figure 2 illustrates how both AA and AO problems work hand-in-hand.

Traditionally, the notion of authorship and its accompanying privacy concern in both AA and AO problems are only toward *human* authors. However, with the arrival of generative AI technologies and due to the potential threats of misused neural texts, one now has to consider authorships by humans, machines, or their combination, and re-thinks about effective solutions for both AA and AO problems for neural texts. Hence, to guide these developments, in this survey, we provide a detailed analysis of both AA and AO problems, their existing solutions, and our perspective on the open challenges. As both AA and AO problems are essentially computational learning problems, we discuss the landscape from *A Data Mining Perspective* and call attention to the security challenges that need to be solved. We believe that the issues of these novel AA/AO problems for neural texts are “nuanced” and therefore require nuanced solutions from the Data Mining and Machine Learning community.

## 2. NEURAL TEXT GENERATION

We first select a handful of Neural Text Generator (NTG) that we focus on in this survey, and introduce a list of popular datasets with neural texts.

### 2.1 Neural Text Generators (NTGs)

Those NTGs studied in this survey are large-scale probabilistic LMs that are capable of generating long-coherent texts (e.g.,  $\geq 200$  words). These LMs are trained on massive amounts of unstructured texts. Based on its architecture structure (e.g., encoder-decoder or decoder only), these LMs use a prompt, a snippet of human-written text, to guide the generation of texts, emulating the most similar style from the training set and predicting one token at a time. Recent works such as [72, 119] survey these NTGs in detail. The progress of NTGs in recent years has been expeditious. As shown in Figure 3, for instance, the sizes of NTGs with respect to their parameters are growing at an exponential rate, yielding the rapid improvement in the quality of neural texts, thus exacerbating the AA/AO problems. Table 1 shows a summary of state-of-the-art NTGs, where many entries are drawn from [107, 108].

Within LMs, in particular, hyperparameters matter a great deal when generating texts. The choice of these hyperparameters, referred to as *decoding strategies*, greatly affects the quality of generated neural texts. According to [50], there are 6 *decoding strategies*: (1) *Greedy sampling* selects the best probable word, (2) *Random sampling* does a stochastic search for a sufficient word, (3) *Top-k sampling* samples from top-k most probable words, (4) *Beam search* searches for most probable candidate sequences, (5) *Nucleus (Top-p) sampling* samples similar to top-k, but its focus is on the smallest possible set of top words, such that the sum of their probabilities is  $\geq p$ , and (6) *Temperature* scales logits to either increase or decrease the entropy of sampling. NTG models often use *Top-k sampling*, *Beam search*, *Nucleus (Top-p) sampling*, and *Temperature* decoding strategies as they produce higher quality texts than other decoding strategies. In fact, [50] reports that neural texts generated with the *Nucleus (Top-p) sampling* strategy are more challenging to attribute authorships.

### 2.2 Neural Text Datasets

To investigate both AA/AO problems for neural texts, one needs benchmark datasets of neural texts. Table 2 describes a list of publicly available datasets that contain neural texts. Labels of the texts in the datasets are either binary (neural vs. human text) or multi-class (having multiple neural texts generated by different NTGs vs. 1 human label). The majority of recent studies have focused on the binary case of the Turing Test to check if a given text is written by a human author or machine (i.e., one of NTGs). Researchers utilized clever labeling and generation methods to build datasets: (1) *Binary dataset*: Researchers first collect human-written texts (e.g., news, blogs, stories, or recipes) and use snippets of these texts as prompts to the chosen NTG to generate a machine-written (neural) text. (2) *Multi-class dataset*: Starting with human-written texts as prompts, this generates multiple neural texts by using different NTG architectures (i.e., human vs. GPT-1, GROVER, PPLM, etc.), different pre-trained sizes of the same NTG architecture (i.e., human vs. GPT-2 small vs. GPT-2 medium vs. GPT-2 large vs. GPT-2 XL, etc.), different decoding strategies (i.e., human vs. GPT-2 top-k vs. GPT-2 top-p, etc.).

NTG	Author	Description
GPT-1 [92]	OpenAI	It used Transformers to model a simple concept - to predict the next token, given the previous token.
GPT-2 [93]	OpenAI	GPT-1 scaled up. There are 4 GPT-2 pre-trained models - <i>small</i> (124 million parameters), <i>medium</i> (355 million parameters), <i>large</i> (774 million parameters), and <i>x-large</i> (1558 million parameters)
GPT-3 [13]	OpenAI	GPT-2, scaled up - increasing parameter and train data size.
GROVER [117]	AllenAI	Similar to GPT-2 architecture and trained to generate political news. There are 3 pre-trained models: <i>GROVER-base</i> , <i>GROVER-large</i> , <i>GROVER-mega</i>
CTRL [59]	Salesforce	Conditional Transformer LM For controllable generation uses control codes to guide generation
XLNet [20]	Facebook	A Cross-lingual Language Model trained on various languages. Only the English model is used for AA
XLNet [116]	Google	A generalized auto-regressive pre-training method that adopts the Transformer-XL framework
FAIR_wmt [16, 82]	Facebook	FAIR_wmt has 3 language models - English, Russian, and German. Only the English model <sup>4</sup> is used, which has 2 models - <i>WMT19</i> [82] and <i>WMT20</i> [16].
TRANSFORMER_XL [24]	Google	Another Transformer model that learns long-term dependency to improve long coherent text generation
PPLM [25]	Uber	The Plug and Play Language Models (PPLM) model upon GPT-2 by fusing the GPT-2 medium with a bag of words (BoW) models. These BoW models are <i>legal</i> , <i>military</i> , <i>monsters</i> , <i>politics</i> , <i>positive_words</i> , <i>religion</i> , <i>science</i> , <i>space</i> , <i>technology</i> . PPLM can plug in any GPT-2 pre-trained model to generate texts
Switch Transformer [35]	Google	Google uses a switch Transformer to build a sparse neural LM with 1.6T parameters are built
GPT-Neo [42]	EleutherAI	EleutherAI replicates GPT-3's architecture. There a 2 model sizes - 1.3B and 2.7B parameters
GPT-NeoX [12]	EleutherAI	A 20 billion parameter autoregressive replication of GPT-3.
GPT-J [112]	EleutherAI	A 6B parameter model similar to the GPT-Neo and GPT-NeoX that uses Mesh Transformer JAX [111] framework to train the model with Pile <sup>5</sup> dataset, a large curated dataset created by EleutherAI
T5 [94]	Google	An encoder-decoder text-to-text Transformer-based model. T5 has 5 pre-trained models - <i>T5-small</i> , <i>T5-base</i> , <i>T5-large</i> , <i>T5-3b</i> , and <i>T5-11b</i>
BART [71]	Facebook	This is another encoder-decoder Transformer-based LM, most effective when fine-tuned
PaLM [17]	Google	PaLM stands for Pathways Language Model. It is a dense decoder-only Transformer-based model trained with [7]'s pathways system framework
OPT-175B [121]	Meta	Meta's response to GPT-3. OPT-175B uses a similar framework to GPT-3 but the training costs 1/7th the carbon footprint of GPT-3.
GeDi [63]	Salesforce	GeDi stands for Generative Discriminator Guided Sequence Generation. Similar to PPLM, GeDi controls text generation using small LMs as generative discriminators

Table 1: Description of state-of-the-art Neural Text Generators (NTGs).

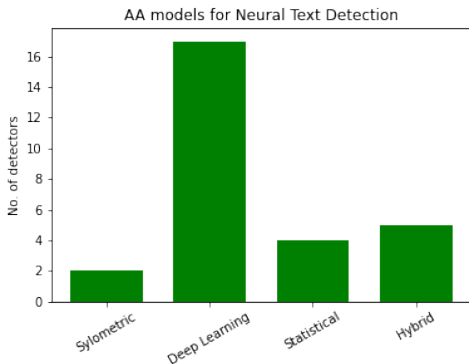


Figure 4: Number of AA solutions for NTD per category in the Taxonomy

### 3. AUTHORSHIP ATTRIBUTION FOR NEURAL TEXTS

Traditional AA problem studies the attribution of an author to a piece of written text out of a number of possible authors. However, in the literature, researchers have also studied a few variations of the AA problem. For instance, the *Author Verification (AV)* problem [8, 60, 60, 100, 101, 105] studies if the given two texts,  $t_1$  and  $t_2$ , are written by the same author? With the rise of neural texts, in addition, a specialized case of AA problem, NTD [5, 44, 51, 104, 107, 117], studies: By and large, however, a good solution for the standard AA problem can lead to a good solution for other variations of the AA problem. As such, we focus on the survey of the standard AA problem for neural texts. Thus, our paper formally defines the AA task as follows.

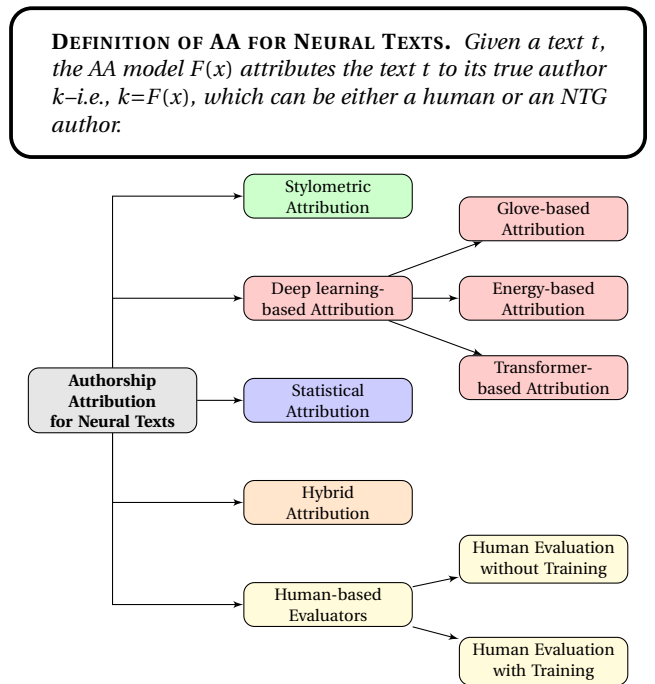


Figure 5: Taxonomy of Authorship Attribution models for NTD

In the following, we survey recent AA solutions that are capable of handling neural texts in different ways, as illustrated in Figure 5: *Stylometric Attribution*, *Deep Learning-based Attribution*, *Statistical Attribution*, and *Hybrid Attribution*.

#### 3.1 Stylometric Attribution

Name	Description	Category	Domain	Labels
GPT-2 dataset [92]	250K Webtext (Human dataset) vs. 250K GPT-2 (small, medium, large, & XL)	Binary	News	GPT-2 & Human
GROVER dataset [117]	Using April 2019 news articles as the prompt, GROVER-Mega generated news articles	Binary	News	GROVER & Human
TuringBench-AA [108]	Used 10K human-written news articles (mostly Politics) from CNN, etc. to generate 10K articles each from 19 NTGs.	Multi-class	News	Human & 19 Machine labels (GPT-1, GPT-2 small, etc.)
TuringBench-TT [108]	The same dataset as <i>TuringBench-AA</i> , except that the datasets are 19 versions of human vs. each of the 19 NTGs.	Binary	News	19 Human vs. Machine combinations (GPT-2, etc.)
Authorship Attribution-AA [107]	Used 1K human-written articles to generate 1K articles each from 8 Artificial Text Generators	Multi-class	News	1 human vs 8 Machine labels (GPT-1, GPT-2, etc.)
Authorship Attribution-TT [107]	The same dataset as <i>Authorship Attribution-AA</i> except that the datasets are 8 versions of human vs. each of the 8 NTGs	Binary	News	8 humans vs Neural combinations
Authorship Attribution-TT mix [107]	The same dataset as <i>Authorship Attribution-AA</i> except that the dataset is human vs. Machine (which is a mixture of all the 8 NTGs)	Binary	News	1 human vs Machine (8 different machines)
Academic Papers & Abstracts [75]	2 datasets - (1) FULL: using a short prompt for a human-written paper generated an academic paper using GPT-2; (2) PARTIAL: Replacing sentences from an Abstract with Arxiv-NLP model generations	Binary	Academic Abstracts	Human & Machine
Hybrid Human-Machine Text [23]	Using human-written text in domains - News, Reddit, and Recipes to generate continuations of the text using GPT-2 XL	Binary	News, Reddit, Recipes	Human prompt & Machine texts
Amazon Reviews [3]	Fine-tuned GPT-2 on 3.6 M Amazon and 560K Yelp reviews	Binary	Reviews	Human & Machine
Human-Machine Pairs [90]	Generated texts with GROVER mega and GPT-2 XL with top-p decoding strategy. Paired human-written texts with a similar neurally generated version	Binary (Human-Machine pairs)	Online forums & News	Human & Machine
NeuralNews dataset [104]	Using the GoodNews [11] dataset as the human-written prompt to generate texts with GROVER. Real images are included in each of the articles.	Binary	News	Human & Machine
SynSciPass [95]	Built dataset using 3 potential sources of neural text: (1) open-ended text generators like GPT-2 & BLOOM (2) paraphrase models like SCITgen and PEGASUS and (3) translation models like Spinbot, real, Google translate, and Opus.	Multi-class	scientific articles	generate, translate, paraphrase, & human
TweepFake [32]	Collected tweets generated by Twitter bots and grouped them into tweets generated by GPT-2, RNN, and other bots	Binary	Tweets	Human & Machine

Table 2: Datasets with neural texts

Stylometry is the statistical analysis of the style of written texts. In traditional AA, stylometric classifiers are built using classical machine learning models trained on ensembles of style-based features such as  $N$ -grams, Part-of-Speech (POS), WritePrints [1], LIWC (Linguistic Inquiry & Word Count)[87], Readability score, and Empath [34]. This has been shown to be a successful approach for traditional AA tasks [68]. Due to such success, these models have been adopted and customized to the task of NTT.

The first attempt at a stylometric classifier to solve the AA task for  $k > 2$  authors is the *Linguistic model* proposed by [107]. It trains a Random Forest classifier with the Authorship Attribution-AA dataset in Table 2 and extracts an ensemble of stylometric features (e.g., *entropy*, *readability score*, & *LIWC* (Linguistic Inquiry & Word Count) [87]). The entropy feature counts the number of unique characters in the text. Readability scores represent the estimated educational level of the author of a piece of text based on lexicon usage. LIWC is a psycho-linguistic dictionary that counts the frequency of words that represents a psychological emotion or linguistic structure [87]. This *Linguistic model* achieves a 90% F1 score and outperforms all the other deep learning-based models. However, this superior performance is a result of the small size of the dataset (only about 1k per data label) [107]. Scaling up the data size in terms of labels and examples will make the

AA task harder, and therefore cause the *Linguistic model* to underperform. This claim is confirmed in their second work using the TuringBench dataset [108]. They compared SOTA deep-learning-based models (BERT and RoBERTa) with several stylometric classifiers - SVM (3-grams), WriteprintsRFC, Random Forest (w/ TF-IDF), Syntax-CNN, Ngram CNN, and N-gram LSTM-LSTM. RoBERTa outperforms all the stylometric classifiers with about a 10-22% increase in F1 scores.

To further explore the benefits of stylometric features leveraged in the traditional AA community, [36] proposes a clever way to use them. This solution aims to solve the special case of AA, *Turing Test* (TT). First, they identify different issues with NTGs which can be captured by specific types of stylometric features. These issues in neural texts are categorized into 4 types: (1) *Lack of syntactic and lexical diversity* which can be captured with Named Entity-tags, POS-tags, and neuralcoref extension<sup>6</sup> (a tool for using a neural network to annotate and resolve coreference clusters) to detect coreference clusters; (2) *Repetitiveness of words* which can be captured by collecting the number of stop-words, unique words, and words from “top-lists” of total words in a text. Also, a “conjunction overlap” measure is defined to calculate the

<sup>6</sup><https://github.com/huggingface/neuralcoref>

Model Name	Classifier Type	Category	Learning Type	Interpretable	Training dataset
GROVER detector [117]	DL (Transformer-based)	Binary	Supervised		GROVER
GLTR [45]	Statistical	Binary	Unsupervised	✓	GPT-2
GPT-2 detector [51]	DL (Transformer-based)	Binary	Supervised		GPT-2
OpenAI detector [51]	DL (Transformer-based)	Multi-class	Supervised		GPT-2 & TuringBench-AA
RoBERTa-TT [108]	DL (Transformer-based)	Binary	Supervised		TuringBench-TT
BERT-TT [108]	DL (Transformer-based)	Binary	Supervised		TuringBench-TT
RoBERTa-Multi [108]	DL (Transformer-based)	Multi-class	Supervised		TuringBench-AA
BERT-Multi [108]	DL (Transformer-based)	Multi-class	Supervised		TuringBench-AA
TDA-based detector [67]	Hybrid	Binary	Supervised	✓	GPT-2
FAST [123]	Hybrid	Binary	Supervised		GPT-2 & GROVER
Energy discriminator [5]	DL (Energy-based)	Binary	Supervised		GROVER
MAUVE [89]	Statistical	Binary	Unsupervised	✓	GPT-2 & GROVER
Distribution detector [38]	Statistical	Binary	Unsupervised	✓	GPT-2
Feature-based detector [36]	Stylometric	Binary	Supervised	✓	GPT-2, GPT-3, & GROVER
Linguistic model [107]	Stylometric	Multi-class	Supervised	✓	Authorship Attribution-AA
DIDAN [104]	Hybrid	Binary	Supervised		Fake images w/ Human vs. GROVER news
XLNet-FT [81]	DL (Transformer-based)	Multi-class	Supervised		GPT-1, GPT-2, XLNet, & BART Reddit posts
Contra-DeBERTa [4]	DL (Transformer-based)	Multi-class	Supervised		TuringBench
Fingerprint detector [27]	Hybrid	Multi-class	Supervised		GPT-2 bot subreddit posts
DistilBERT-Academia [75]	DL (Transformer-based)	Binary	Supervised		GPT-2 Academia abstract & paper
Sentiment modeling detector [3]	DL (Glove-based)	Binary	Supervised		GPT-2 Amazon Reviews
BERT-Defense [52]	DL (Transformer-based)	Binary	Supervised		GPT-2 Large WebText
RoBERTa-Defense [91]	DL (Transformer-based)	Binary	Supervised		GROVER
RoBERTa w/ GCN [54]	DL (Transformer-based)	Binary	Supervised		GPT-2
DeBERTa v3 [95]	DL (Transformer-based)	Multi-class	Supervised		GPT-2, BLOOM, PEGASUS, OPUS, SCGen, Spinbot
Ensemble [39]	DL (Transformer-based)	Binary	Supervised		TweepFake
CoCo [73]	Hybrid	Binary	Supervised	✓	GPT-2 & GROVER
DetectGPT [80]	Statistical	Binary	Unsupervised	✓	GPT-2, OPT-2.7, GPT-Neo-2.7, GPT-J, & GPT-NeoX

Table 3: Authorship Attribution models (Binary & Multi-class) for NTD

overlap of the top-k words ( $k = 100, 1K, 10K$ ); (3) *Lack of coherence* which can be captured using entity-grid representation to track the appearance of the grammatical role of entities. They also use neuralcoref to detect coreference entity clusters; (4) *Lack of purpose* which is captured using a lexicon-package, empath [34] containing 200 linguistic features [36]. To evaluate the generalizability of these features, an ensemble of all the features is used to build a classifier - *Feature-based detector*. This detector is trained and tested on different sizes of the GPT-2 models. It is further evaluated on GPT-3 and GROVER texts. The classifier performs consistently in detecting texts generated by GPT-3, GROVER, and different model sizes of GPT-2, suggesting it is generalizable to different NTG model sizes [36]. Further results suggest that some of the 4 categories of issues are prevalent in the top-k decoding strategy. Also, more quality-focused features (especially ones focused on Lexical diversity) perform better than statistical features such as the TF-IDF baseline.

Scaling up and creating a more realistic scenario, [27] collect 108 SubReddit blog posts generated by GPT-2 fine-tuned on 500K subreddit posts and comments. Every 108 labels indicate the 108 users of SubReddit (r/SubSimulatorGPT2). These 108 authors are detected using a set of features called “writeprints” features for the AA model [27]. Writeprints [1] is a stylometric feature that collects lexical, content-based, and idiosyncratic features as the baseline. The writeprints classifier underperforms, compared to the RoBERTa-based baseline models. Similarly, a stylometric classifier with 791 stylometric features based on [58] is used to detect neural texts. This classifier has 4 categories of features: *Character*, *word*, *sentence*, and *Lexical Diversity* features. The classifier is an ensemble of classical ML models such as Random Forest and SVM and the stylometric features. BERT, a non-stylometric clas-

sifier, outperforms these stylometric classifiers significantly [56].

Finally, stylometric classifiers are best used when the dataset size is small. When data size increases, these models underperform, allowing deep learning-based models to outperform significantly. Thus, we conclude that stylometric classifiers can only be considered good baselines since they underperform when the problem scales up. Another limitation of stylometry is that it fails to detect neural misinformation due to NTG’s capacity to generate consistent misinformation [96].

## 3.2 Deep Learning-based Attribution

*Stylometric* classifiers struggle to accurately assign the true authorship to human vs. neural texts. In Section 3.1, we observe that some of the stylometric classifiers were outperformed by deep learning-based models. Additionally, [96]’s findings of stylometry failing to detect neural misinformation, further calls for a different technique to solve the AA task for NTD. Therefore, researchers have adopted and advanced deep learning-based techniques for the attributing of neural vs. human text. These models can be further categorized into 3 types of deep learning-based classifiers - *Glove-based*, *Energy-based*, and *Transformer-based* Attribution.

### 3.2.1 Glove-based Attribution

Glove is an unsupervised learning algorithm for extracting the representation of words. It aggregates global word-word co-occurrence statistics from a piece of text [88]. Using GloVe word embeddings with RNN and LSTM-based neural networks was considered SOTA until, 2018 (birth of BERT [26]). Thus Glove-based classifiers now provide a good baseline for text classification tasks. Some of the best-performing AA classifiers in the

traditional AA communities are an ensemble of stylometric features + GloVe pre-trained models w/ a neural network architecture. Several Glove-based classifiers have been used as baselines for the NTD task. *Syntax-CNN* [120], *N-gram CNN* [99], and *N-gram LSTM-LSTM* [53] are baselines for [108]. *Embedding, RNN, Stacked-CNN, Parallel-CNN*, and *CNN-RNN*, are baselines for [107]. They demonstrated that Glove-based classifiers are unsuitable for solving the AA problem when there are  $k > 2$  authors. Furthermore, the *Fingerprint detector* is compared with 4 baseline models - *Gaussian Naive Bayes, Random Forest, Multi-layer Perceptron*, and *CNN* classifiers using the Glove word embeddings of the data. They underperform significantly [27].

Lastly, *Sentiment modeling detector*, a variation of sentiment neuron used to learn a single-layer multiplicative LSTM (mLSTM) [64] is used to detect texts generated by GPT-2 [3]. The goal is to force the model to focus on a specified sentiment [3]. This model outperforms and sometimes performs comparably with the baseline - original mLSTM model [65], achieving a 70% accuracy in detecting GPT-2 generated Amazon and Yelp reviews [3].

### 3.2.2 Energy-based Attribution

Energy-based models (EBMs) are un-normalized generative models based on energy functions [70]. Using the energy functions, EBMs model the probability distribution of its training data and generates high quality data similar to the training set [70]. It is also able to adapt to changes in the Language model. Due to this capability, *Energy-based classifier* is proposed by [5] to detect neural texts. This classifier is trained on 3 datasets of different domains - *Books, CCNews*, and *Wikitext*. Three sizes of the GPT-2 model are used for the generator architectures and three architectures are used for the energy function - *Linear, BiLSTM*, and *Transformer*. Their findings suggest that: (1) as the NTG increases in size, the harder the AA task becomes; (2) the biggest energy function (i.e. *Transformer*) performs the best in detecting texts generated from large language models (e.g., GPT-2 Large & XL); (3) as the length of texts increases, the task becomes even more non-trivial; and (4) the classifiers are able to generalize to data that it is not trained on.

In addition, EBMs are very expensive to train and do not scale well [5]. While the *Energy-based classifier* performs well in the AA problem, achieving over a 90% in all experiments, applying the classifier to a much larger dataset is too expensive to justify.

### 3.2.3 Transformer-based Attribution

Since the advent of the Transformer architecture, the current SOTA text classification models are Transformer-based models. Based on Section 3.2.2, we observe that large models are better at detecting neural texts. However, since EBMs are too expensive, several researchers have adopted Transformer-based models (i.e., BERT, RoBERTa, etc.) for the AA tasks. In Section 3.1, most of the stylometric classifiers were outperformed by Transformer-based classifiers. This further supports the application of this classification technique to the AA problem.

*GROVER detector*<sup>7</sup> [117] is trained on texts generated by the GROVER NTG. It is built with similar architecture as the GPT-2 classifier. *GROVER detector* has been evaluated on neural texts generated by different NTGs (GPT-2, FAIR, PPLM, etc.) [37, 107, 108, 123]. It performs well at detecting neural texts generated by older NTGs

<sup>7</sup><https://grover.allenai.org/detect>

(2018-2019), however, struggles at detecting more recent NTGs accurately. For instance, *GROVER detector* achieved a 58% F1 score in detecting GPT-3 texts with the TuringBench dataset [108]. Next, GPT-2 has a detector trained to detect texts generated by GPT-2 - *GPT-2 detector*<sup>8</sup> [51]. Just like *GROVER detector*, *GPT-2 detector* has also been evaluated on neural texts generated by different NTGs [40, 107, 108, 114] and more easily detects older NTGs than newer NTGs. This is confirmed with *GPT-2 detector*'s performance in detecting GPT-3 texts, achieving a 53% F1 score [108]. The reason is that newer NTGs, such as GPT-3 tend to generate more human-like texts which confuse SOTA older AA models, like *GPT-2 detector*.

There are two RoBERTa-based models (base & large) trained on GPT-2 dataset<sup>9</sup> in huggingface repo<sup>10, 11</sup>. We call both the base and large models, *OpenAI detector*. This AA model has been evaluated on neural texts generated by different NTGs [108, 114]. *OpenAI detector* is the same model as *GPT-2 detector*, except that *OpenAI detector* has been re-purposed for the AA multi-class setting, while *GPT-2 detector* remains for the AA binary setting. *OpenAI detector* performs comparably to the AA models - *BERT-Multi* and *RoBERTa-Multi* when evaluated on the TuringBench-AA dataset [108]. *BERT-Multi* and *RoBERTa-Multi* are BERT and RoBERTa base models, respectively trained on the TuringBench-AA dataset. *BERT-TT* and *RoBERTa-TT* outperform *GROVER detector* and *GPT-2 detector* when evaluated on the TuringBench-TT dataset [108]. *BERT-TT* and *RoBERTa-TT* are BERT and RoBERTa base models, respectively trained on the TuringBench-TT dataset. *BERT-TT*, outperforms all the models, including *RoBERTa-TT* significantly. Furthermore, for all 19 pairs of human vs. NTG, no model consistently outperforms all other models. In fact, *GROVER detector* and *GPT-2 detector* performs poorly in detecting texts generated by GROVER and GPT-2, respectively [108].

*XLNet-FT* is a fine-tuned XLNet classification model trained to detect texts generated by GPT-2 [81]. The generalizability of the model is evaluated on different subreddit post domains. *XLNet-FT* performs consistently, achieving over a 90% accuracy in all experiments, suggesting that it is generalizable [81]. However, when *XLNet-FT* is further evaluated on neural texts generated by top-p and top-k decoding strategy, there is a significant drop in accuracy, suggesting that the AA model may not be generalizable.

Using an *in-the-wild* dataset concept, *RoBERTa-Defense* is evaluated on 4 types of *in-the-wild* datasets [91]. *In-the-wild* datasets are test sets generated from an entirely different NTG from the training set. *RoBERTa-Defense* is trained on human & GROVER Real news in Table 2 and compared to other SOTA AA models - *GLTR* (with two different LMs - BERT & GPT-2 which results in *GLTR-BERT* & *GLTR-GPT2*), *GROVER detector*, *BERT-Defense*, and *FAST*. *RoBERTa-Defense* outperforms all other models significantly. *BERT-Defense*, a BERT model fine-tuned on GPT-2 Large Webtext dataset in Table 2 from which *RoBERTa-Defense* is inspired is evaluated with different decoding strategies [52]. *BERT-Defense* is trained and tested on neural texts generated by different decoding strategies - top-k, top-p, untruncated random, and mixed (i.e., dataset containing equal amounts of each strategy) [52]. The classifier trained on the mixed dataset is the most

<sup>8</sup><https://huggingface.co/openai-detector/>

<sup>9</sup><https://github.com/openai/gpt-2-output-dataset>

<sup>10</sup><https://huggingface.co/roberta-base-openai-detector>

<sup>11</sup><https://huggingface.co/roberta-large-openai-detector>



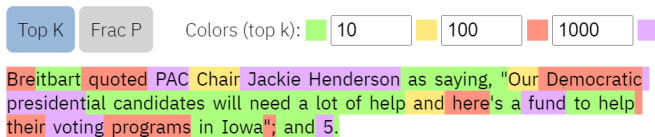


Figure 6: [108] used GLTR [45] on GPT-3 texts. Green represents the most probable words (top-10); yellow the 2nd most probable (next top-100 probable words); Red the least probable (next top-1000 probable words); and purple the highest improbable words. The hypothesis is that neural texts are often populated with mostly Green and yellow words. However, we see that texts generated by GPT-3 are very human-like according to the hypothesis.

generalizable classifier. Similarly, *RoBERTa*, *BERT*, *ELECTRA* [19], and *ALBERT* [69] are evaluated on *in-the-wild* dataset [86]. These models are evaluated, specifically on out-of-domain COVID-19 human-written vs. neural news. The neural news is generated with GPT-2 small, medium, large, XL, and GPT-Neo using top-p and top-k decoding strategies [86]. *ELECTRA* performs better at generalizing to out-of-domain neural texts, achieving an average accuracy of 86% on all out-of-domain datasets.

Due to the nuances of neural texts, [4] proposes to combine the advantages of contrastive learning [43] with a Transformer-based classifier. Thus, they propose *Constra-DeBERTa* which is a DeBERTa model [49] trained with a contrastive learning approach. Contrastive learning is a technique that clusters similar examples together and separates dissimilar examples in a representation space [4, 43]. However, while *Constra-DeBERTa* outperforms other SOTA traditional AA models, it only performs comparably to *RoBERTa-Multi* on detecting the TuringBench dataset. Similarly, [95] trains *DeBERTa v3* [48] on the SynSciPass dataset to answer the question, if a piece of text is neurally generated, how was it generated? The answer choices are *generated*, *paraphrased*, or *translated* vs. human-written. Using this dataset, *DeBERTa v3* achieves a 99.6% F1 score. Next, *DistilBERT-Academia* is trained on human vs. GPT-2 academic abstracts and papers [75] and achieves a 62.5% and 70.2% accuracy on the FULL and PARTIAL Academic datasets, respectively. Furthermore, *RoBERTa w/ GCN* (Graph Convolutional Networks) is used to detect human-written news with entities manipulated and replaced by GPT-2 [54]. The GCN [61] model is used to capture factual knowledge of neural vs. human news articles. *RoBERTa* outperformed the proposed model - *RoBERTa w/ GCN* on most of the GPT-2 detection tasks [54].

Lastly, *ruBERT*, a Russian BERT model is used to distinguish Russian neural texts from Russian human-written texts as a shared task [97]. This fine-tuned *ruBERT* (Russian BERT) achieves 82.6% accuracy for  $k = 2$  authors and 64.5% accuracy for  $k > 2$  authors [85]. Finally, using an *Ensemble* classifier (BART, BERTweet, and TwitterRoberta), [39] achieves an 84% accuracy in distinguishing between GPT-2 and human tweets. However, using the same model for GPT-3 generated tweets achieves a 54% accuracy [39]. This suggests that GPT-3 generates more human-like tweets than GPT-2.

### 3.3 Statistical Attribution

We observe that while there are some well-performing stylometric and deep learning-based models, there is still a lot of room for

improvement, especially in building generalizable models. The biggest feat is in building classifiers that perform consistently well in detecting neural texts generated by top-p and top-k decoding strategies. Thus, statistical models are proposed to combat these limitations. To assess the validity of statistical techniques, a hypothesis using  $k$ -order Markov approximations are formulated [110]. This statistical formulation proves the hypothesis that human language is stationary and ergodic as opposed to neural language. The formal hypothesis testing framework is used to establish limits in error exponents between human and neural text [110]. This suggests that statistical AA models for neural texts could be successful. There are currently only four statistical classifiers that capture the writing style of neural texts by modeling their statistical distribution.

The first statistical AA classifier proposed for NTD is *GLTR* [45]. *GLTR* performs 3 tests - (1) probability of the word; (2) the absolute rank of the word; (3) the entropy of the predicted distribution to detect neural texts. *GLTR* has a demo<sup>12</sup> that highlights words by distribution and is used to assist humans in detecting neural texts. See Figure 6 [108] to see how *GLTR* detects texts generated by GPT-3. This classifier improved human performance in detecting neural texts from 54% to 72%. However, since 2019 when it was built, more sophisticated NTGs have been built. These newer NTGs have more human-like statistical distribution, making it harder for *GLTR* to distinguish neural texts from human texts. *GLTR*, especially underperforms in detecting GPT-3 texts, achieving a 35% F1 score, which is significantly less than a random guess (50%) [108].

*MAUVE* is another statistical classifier [89]. This AA classifier measures the gap between the distribution of human and neural texts. Using KL-divergence, *MAUVE* models two types of errors that highlight the unique distributions in human vs. neural texts [89]. Human detection of texts generated by GROVER and GPT-2 correlated strongly with *MAUVE*'s highlight of differences between human and neural texts. *Distribution detector*, an unsupervised AA model for calculating the distribution of repeated n-grams in neural texts is used to detect neural texts [38]. The hypothesis is that NTGs are more repetitive than humans which is also one of the hypotheses of [36]. *Distribution detector* achieves over 90% and 80% accuracy in detecting texts generated by GPT-2 using top-k and top-p decoding strategies, respectively.

Most recently, a zero-shot unsupervised neural text detector, *DetectGPT* [80], is proposed. The hypothesis of this statistics-based detector is that neural texts tend to lie in areas of negative curvature of the log probability function [80]. Therefore, if a piece of neural text is perturbed, the curvature of the log probability will still bear a strong similarity to the unperturbed neural texts. Hence, [80] considers an AO technique that slightly modifies the original neural text, while preserving semantics. After running several perturbation experiments, a threshold for perturbation discrepancy is defined and used to detect neural text. Thus, by measuring the curvature of log probability with the strict constraint of perturbation discrepancy threshold, *DetectGPT* can detect texts generated by a neural method. Finally, *DetectGPT* detects GPT-3 generated texts with an average of 85% AU-ROC, performing comparably to *RoBERTa* [73]. Lastly, *DetectGPT* has an online demo<sup>13</sup>.

<sup>12</sup><http://gltr.io/dist/index.html>

<sup>13</sup><https://detectgpt.ericmitchell.ai/>

### 3.4 Hybrid Attribution

There are advantages in each of the AA model categories, however, each of them is still unable to accurately attribute neural vs. human texts to their authors consistently. Furthermore, the issue of different decoding strategies, also, make the AA models unable to generalize well [36, 50, 52, 91]. Therefore, a few researchers have proposed hybrid classifiers, which are ensembles of two or more of the AA categories.

*TDA-based detector*, an ensemble of the Transformer-based and statistical AA techniques is used to solve the NTD task. This classifier involves obtaining the attention matrices of BERT’s word representations of texts generated by GPT-2 and GROVER. Next, using these BERT word representations, 3 interpretable TDA-based features are extracted: (1) *Topological Features*: Calculating the first 2 betti numbers (i.e., topological features based on the connectivity of n-dimensional simplicial complexes) of the attention matrices; (2) *Features derived from barcodes*: Calculating characteristics of the barcode plots of the persistent homology of the attention matrix; (3) *Features based on the distance to patterns*: Calculating the distance in features in the attention graph. This feature is used to capture linguistic patterns. Finally, *TDA-based detector* is a logistic regression model, trained on an ensemble of the three TDA features [67]. Comparing this model to pre-trained and fine-tuned BERT models, it performs comparably to BERT models fine-tuned on GPT-2 small Webtext, GPT-2 XL Amazon Reviews, and GROVER News [67]. While more analysis is required to understand why the *TDA-based detector* performs well, this approach has interpretable qualities that should be explored in future work.

*Fingerprint detector* is another hybrid classifier for NTD. It is an ensemble of fine-tuned RoBERTa embeddings and CNN classifier [27]. *Fingerprint detector* solves the AA task by detecting 108 neural authors. The *Fingerprint detector* achieves a 70% accuracy (top-10). This shows promise in the area of detection of neural texts *in-the-wild*, where there are  $k > 100$  authors. To continue the quest for generalizable classifiers, *FAST* uses a Graph Neural Network (GNN) architecture with RoBERTa to capture the factual structure of neural and human texts [123]. It detects neural texts by calculating the RoBERTa word embeddings of the texts and then extracting the graphical representation [123]. Next, it uses a GNN to capture sentence representations that consider coherence [123]. The experiments included detecting texts generated by GROVER and GPT-2. *FAST* outperforms *GROVER detector* and other baselines significantly. Surprisingly, it performs the best at detecting human-neural text pairs, achieving over 93% accuracy while unpaired texts achieve over 84% accuracy.

*CoCo* is a coherence-based contrastive learning model [73]. It is architecturally similar to *FAST* in that it uses a graphical neural network to represent the sentences of human-written vs. neural texts. Since human-written texts are more coherent than neural texts, they sentences share more entities [73]. The texts are represented as RoBERTa embedding weights which are concatenated with the sentence-level graphical representations of the texts. These concatenated features are input for an LSTM with attention. Lastly, *CoCo* is trained using the sum of the coss-entropy loss and contrastive loss [73] to improve model performance. Thus, it achieves an F1 score of 83% and 94% using the full dataset for GROVER, and GPT-2, respectively [73]. Furthermore, calculating the graphical metrics showed that in terms of the number of vertex and edges, human-written texts have significantly more

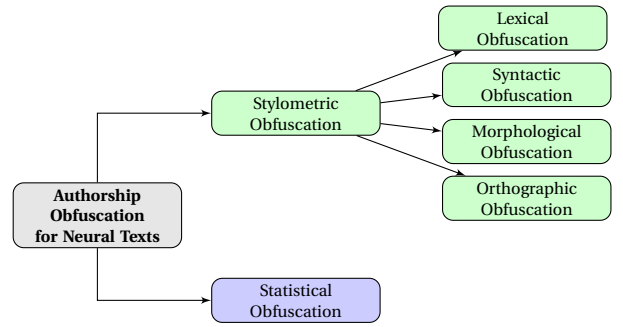


Figure 7: Taxonomy of Authorship Obfuscation algorithms/techniques for NTD

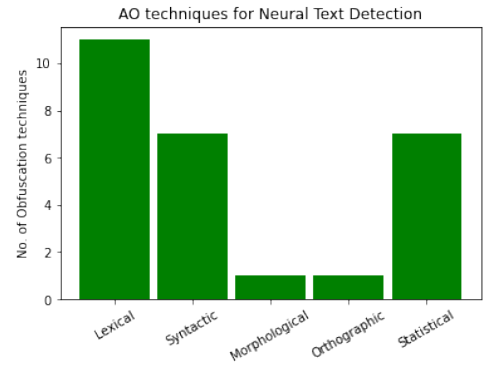


Figure 8: Number of AO techniques for NTD per category in the Taxonomy

graphical features than neural texts.

Lastly, [104] explore the most realistic scenario of misinformation where malicious users of NTGs, pair neurally generated misinformation with fake/real images to increase the authenticity of the news article. *DIDAN* is a multi-modal NTD evaluated on a multi-modal dataset containing both texts and images [104]. This NTD encodes the texts with BERT encoder and investigates Visual-Semantic representations from images and texts. These features are used to evaluate the semantic consistency between linguistic and visual components in a news article [104]. An *authenticity score* is defined to represent the probability of an article being human-written. It is calculated by extracting the co-occurrences of named entities in the news articles and captions [104]. They build different variations of dataset, some only containing text. Using only the text dataset, *DIDAN* is compared to *GROVER detector* as well as other baseline models, and outperforms all of them [104].

## 4. AUTHORSHIP OBFUSCATION FOR NEURAL TEXTS

In the task of NTD, AA models are evaluated under adversarial settings to assess their robustness. Due to the security risk, NTGs pose, it is important that AA models are robust to adversarial perturbations. The problem of administering adversarial perturbations to texts to cause an accurate AA model to assign inaccurate authorship is called **Authorship Obfuscation (AO)**. This is because AO is the process of masking an author’s writing style/sig-



Authorship Obfuscation	Adversarial Attack
Has a strict definition of writing style	Has a loose definition of writing style
Requires consistent/uniform change in writing style	Does not require consistent/uniform change in writing style
Requires semantics to be preserved	Does not always require semantics to be preserved

Table 4: Differences between Authorship Obfuscation and Adversarial Attack

AO technique	Scenario	Category	Interpretable	Preserves semantics	Obfuscated dataset
Homoglyph [37, 114]	Black-box	Stylometric (Orthographic)	✓	✓	GROVER & GPT-2
Upper/Lower Flip [37]	Black-box	Stylometric (Morphological)	✓	✓	GROVER & GPT-2
DeepWordBug [22, 102]	Black-box	Stylometric (Lexical)			GPT-2 & GPT-3
Misspellings attack [37, 114]	Black-box	Stylometric (Lexical)	✓		GROVER & GPT-2
Whitespace attack [37]	Black-box	Stylometric (Lexical)	✓		GROVER & GPT-2
Deduplicate tokens [90]	Black-box	Stylometric (Lexical)	✓		Human-Machine Pairs
Shuffle tokens [90]	Black-box	Stylometric (Syntactic)	✓		Human-Machine Pairs
Retain only (non)-stopwords [90]	Black-box	Stylometric (Syntactic)	✓		Human-Machine Pairs
Retain tokens in high/low frequency [90]	White-box	Statistical	✓		Human-Machine Pairs
Replace text with likelihood ranks [90]	White-box	Statistical	✓		Human-Machine Pairs
Replace text with specific linguistic features [90]	White-box	Statistical	✓		Human-Machine Pairs
TextFooler [22, 91]	Black-box	Stylometric (Lexical)		✓	GPT-2 medium, GPT-2 XL, GPT-3, GROVER
Varying sentiment [9]	Black-box	Stylometric (Lexical)	✓		GROVER
Source-target exchange [9]	Black-box	Stylometric (Syntactic)	✓		GROVER
Entity replacement [9]	Black-box	Stylometric (Lexical)	✓		GROVER
Alter numerical facts [9]	Black-box	Stylometric (Syntactic)	✓		GROVER
Syntactic perturbations [9]	Black-box	Stylometric (Syntactic)	✓		GROVER
Article shuffling [9]	Black-box	Stylometric (Syntactic)	✓		GROVER
Selecting highest human-class probability [85]	Black-box	Statistical			Russian neural & human-written texts
Adding detector’s log-probability to sampling technique [85]	White-box	Statistical			Russian neural & human-written texts
Varying the text decoding strategy (and its parameters) [91]	White-box	Statistical		✓	GPT-2 Large, GPT-2 XL, GPT-3, GROVER
Varying the number of priming tokens [91]	White-box	Statistical			GPT-2 Large, GPT-2 XL, GPT-3, GROVER
DFTFooler [91]	Black-box	Stylometric (Lexical)		✓	GPT-2 Large, GPT-2 XL, GPT-3, GROVER
Random Perturbations [91]	Black-box	Stylometric (Lexical)			GPT-2 Large, GPT-2 XL, GPT-3, GROVER
ALISON [115]	Black-box	Stylometric (Syntactic)	✓	✓	TuringBench
Mutant-X [115]	Black-box	Stylometric (lexical)		✓	TuringBench
Avengers [115]	Black-box	Stylometric (lexical)		✓	TuringBench

Table 5: Authorship Obfuscation techniques for Neural Texts. *With cited papers that implemented them.*

nature to conceal the identity, usually for privacy reasons [76]. It is a strict case of Adversarial attacks [122], as the goal is to obfuscate writing style and preserve semantics, such that both human and automatic detection is evaded. See the differences between the two in Table 4. AO is a well-studied problem in the traditional AA community [57, 76, 77, 78, 118], and has been extended to the niche AA community for NTD. This AO for neural texts problem is formulated as:

**DEFINITION OF AO FOR NEURAL TEXTS.** *Given an AA model  $F(x)$  that accurately assigns authorship of text  $t$  to  $k$  which can be either a human or an NTG author, the AO model  $O(x)$  slightly modifies  $t$  to  $t^*$  (i.e.,  $t^* \leftarrow O(t)$ ) such that the authorship is disguised (i.e.,  $F(t^*) \neq k$ ) and the difference between  $t^*$  and  $t$  is negligible.*

Thus, we survey all AO techniques employed to obfuscate neural texts in different categories, as illustrated in Figure 7: *Stylometric*

*Obfuscation*, and *Statistical Obfuscation*.

## 4.1 Stylometric Obfuscation

In order to build a robust stylometric classifier, as is observed in Section 3.1, an ensemble of features that capture several linguistic structures such as - Lexical, Syntax, etc. are required. However, to obfuscate an author’s writing style, only one of the linguistic structures may be perturbed. Therefore, all the stylometric AO techniques only target a specific linguistic structure, unlike AA classifiers. Based on the stylometric obfuscation techniques used to obfuscate neural texts, we further divide this category into 4 categories - *Lexical*, *Syntactic*, *Morphological*, and *Orthographic* Obfuscation.

### 4.1.1 Lexical Obfuscation

Lexical relates to the word choice of a piece of text. Thus lexical obfuscation algorithms aim to mask authors’ writing styles by replacing certain keywords with their synonyms while preserving semantics. Below, we discuss different techniques used to

achieve lexical obfuscation for neural texts. Misspellings attacks may be considered a trivial AO technique, however, it is effective in obfuscation. The misspellings attack uses a list of commonly misspelled words<sup>14</sup> to determine which words to replace with their misspelled version. This AO technique is successful in obfuscating texts generated by GPT-2 and GROVER, and thus, evades detection of the following AA models - *GLTR*, *GROVER*, and *GPT-2* detector [114]. *GROVER detector* is further evaluated with this AO technique by obfuscating texts generated by GROVER. With only less than 10% of the texts perturbed, this attack is 94% successful [37]. However, this attack can be maneuvered by spell check algorithms, making the obfuscation technique, not robust [114]. In addition to misspelling, a whitespace attack ("will face" → "willface") is used to evaluate the robustness of *GROVER detector*. With only less than 4% of the texts perturbed, the attack is 85% successful [37].

Interesting artifacts/characteristics of neural texts still remain somewhat elusive. Therefore, perturbing these neural texts could reveal characteristics that have evaded the AA & AO community. Hence, using linguistic and statistical perturbations of words in the text, [90] extract important characteristics of neural text. For the linguistic-based perturbations, a lexical obfuscation technique is implemented - *Deduplicate tokens* which keeps the first occurrence of a token/word as is and replaces other occurrences with */MASK* token. This AO technique surprisingly improves the AA performance, suggesting that reducing the number of token occurrences may remove trivial features, causing the AA classifiers to focus on the more important features [90]

Next, texts generated by GROVER are obfuscated with the following techniques: (1) *varying sentiment*: changing the sentiment of words by replacing the word with another word of a different sentiment (positive → negative); and (2) *entity replacement*: replace entity with a useless entity [9]. Results suggest that both *GROVER* and *GPT-2 detectors* are vulnerable to these lexical-based perturbations.

Mutant-X [76] and Avengers [47] are used as baseline AO techniques to obfuscate the TuringBench dataset [115]. Mutant-X uses a genetic algorithm to search for suitable word replacement such that the semantics are preserved and the internal/substitute AA model misclassifies [76]. This process is notorious for its expensive computational complexity [103]. An internal model is used because, in the real world, the original AA model may not be known. Moreover, Mutant-X generates the obfuscated text and tests if it has evaded the AA model. If evasion is not successful, the process is repeated and tested for the defined max number of iterations [76]. These factors significantly increase the runtime of Mutant-X. Furthermore, the success of Mutant-X is dependent on how strong the internal AA model, which undermines the generalizability of Mutant-X. Hence, Avengers [47], an improved version of Mutant-X is proposed. Avengers is an ensemble version of Mutant-X. Unlike, Mutant-X, the internal AA model is an ensemble AA model, such that each classifier out of  $N$  classifiers focuses on different linguistic structures - syntax, semantics, etc.

DeepWordBug [41], a realistic character-level black-box attack is used to evaluate the robustness of 3 types of model - Statistical classification model [83], RoBERTa [74], and an Ensemble model (Statistical model + RoBERTa) [22]. It perturbs charac-

ters such that misclassification is maximized, while the Levenshtein edit distance is minimized [22, 41]. These models were trained with GPT-2 medium webtext and tested 3 separate test datasets - human vs. neural webtext from GPT-2 medium, GPT2 XL, and GPT-3 [22]. While deep learning-based classifiers achieve a higher performance in unperturbed/clean texts, statistical classifiers were found to be more robust to obfuscation [22]. Thus, the Ensemble model merges the advantages of high performance and adversarial robustness of the 2 models. DeepWordBug did not reasonably degrade the Ensemble model's performance, suggesting that DeepWordBug is not robust for this task. However, when DeepWordBug is used to evaluate the robustness of *GROVER detector* (Mega model) and *OpenAI detector* (base and large models) by perturbing the GPT-2 generated Yahoo answers & Yelp Polarity vs. Human datasets, it is successful [102]. In fact, DeepWordBug is found to be a very successful AO technique, reducing the accuracy of the Yahoo and Yelp datasets from 67.9% to 0.4% and 87.4% to 6.9%, respectively [102]. This suggests that the *OpenAI* and *GROVER detectors* are not robust to this kind of AO technique when evaluated on GPT-2 generated Yahoo answers & Yelp Polarity.

In addition, TextFooler [55], a realistic word-level black-box attack is used to evaluate the robustness of the Statistical model, RoBERTa, and Ensemble model (Statistical model + RoBERTa) [22]. TextFooler replaces words with synonyms based on cosine similarity within the embedding space [22, 55]. Based on the results, TextFooler is a robust AO technique, especially for Transformer-based models. Furthermore, as a substitute for human judgment, *MAUVE* is used to measure the human judgment of obfuscated texts. [22] finds that adversarial perturbation reduces *MAUVE* score which means that text quality is degraded and therefore neural texts are likely to be detected accurately by humans.

To further evaluate the robustness of the AA models - *GLTR* (*GLTR-BERT* & *GLTR-GPT2*), *GROVER detector*, *BERT-Defense*, *FAST*, and *RoBERTa-Defense*, texts generated by GPT-2 and GROVER are obfuscated with TextFooler, Random Perturbations [91], and DFTFooler [91] AO techniques. Random Perturbations is an attack method that replaces random words with synonyms while preserving the semantics. DFTFooler is similar to TextFooler but only needs a publicly available pre-trained LM to generate obfuscated texts. This makes DFTFooler not as computationally costly as TextFooler [91]. Also, DFTFooler perturbations are transferable [91]. To find a valid word substitution using DFTFooler, there are 4 steps: (1) synonym extraction; (2) POS checking; (3) Semantic Similarity checking; and (4) Choose a synonym with low confidence as measured by a LM. BERT and GPT-2 XL are used as the LM for DFTFooler. Results suggest that TextFooler is a stronger AO technique than DFTFooler, but performs comparably, achieving 23-91% Evasion Rate [91]. Evasion rate is defined as the fraction of perturbed neural text that evades detection by an AA model. A high evasion rate indicates a high attack success. Furthermore, using a bi-directional LM (BERT) as the backend for DFTFooler, and increasing the number of words perturbed, increases the evasion rate of DFTFooler. Based on the results, *FAST* is the most adversarially robust AA model. This may be due to the hybrid nature of the model as it combines the benefits of stylometric, statistical, and deep learning-based techniques as discussed in section 3. Another reason for *FAST*'s superior performance is its use of semantic features based on entities [91].

#### 4.1.2 Syntactic Obfuscation

<sup>14</sup>[https://en.wikipedia.org/wiki/Wikipedia:Lists\\_of\\_common\\_misspellings/For\\_machines](https://en.wikipedia.org/wiki/Wikipedia:Lists_of_common_misspellings/For_machines)

Syntax relates to the order of words in a piece of text. Thus, syntactic obfuscation techniques change the original arrangement of words in a document, in order to obfuscate the author's writing style. Below, we discuss such techniques used on neural texts. Characteristics of neural texts are extracted by perturbing the syntactic structure of the texts with the following syntactic perturbation techniques: (1) *Shuffle tokens* which randomly shuffles the word order of the texts; (2) *Retain only (non)-stopwords* which removes all words, except for stopwords [90]. The accuracy of the AA model only dropped marginally. This suggests that these AO techniques are not robust. It also implies that these syntactic features are not important for NTD.

The robustness of *GROVER detector* is further evaluated by syntactic obfuscation techniques on texts generated by GROVER. These techniques are: (1) *source-target exchange*: interchanging the source and target; (2) *alter numerical facts*: distort numerical facts; (3) *syntactic perturbations*: changing the word form by adding/removing punctuation ("There is" → "There's"); and (4) *article shuffling*: replacing  $N\%$  of a real (human-written) article's sentences with  $N$  sentences of a fake (neural) article [9]. All AO techniques were successful, except *article shuffling*. Also, stylometric classifiers were found to be more robust, except when perturbed under stricter constraints, such as perturbing a large percentage of texts [9].

*ALISON* [115] is another syntactic AO technique. It reduces inference time by 100-200x when compared to SOTA AO techniques such as Mutant-X [76], and Avengers [47]. It has an internal classifier trained on a set of linguistic AA features, which allow *ALISON* to generate suitable phrase replacements that preserve semantics. *ALISON* is used to evaluate the robustness of 3 Transformer-based models - BERT, DistilBERT, and RoBERTa by obfuscating 2 datasets - TuringBench and Blog Authorship Corpus [115]. It is able to obfuscate the datasets well, causing the AA models to underperform on obfuscated texts. Furthermore, *ALISON* is able to preserve the semantics of the original text much better than their baseline AO techniques (i.e., Mutant-X and Avengers).

### 4.1.3 Morphological Obfuscation

Morphology is the study of word forms. Thus, morphological obfuscation techniques change the configuration of a word (e.g. "won't" → "will not"). Upper/Lower Flip ("Leaving" → "Leaving") is a morphological AO technique that may be considered trivial. However, it is successful in obfuscating texts generated by GROVER which significantly reduces the performance of *GROVER detector* [37]. With only about 2% of the texts perturbed, it achieved a 96% success rate in evading *GROVER detector*'s detection [37].

### 4.1.4 Orthographic Obfuscation

Orthography is the spelling convention of a language. Thus, orthographic obfuscation techniques aim to change the original spelling convention used in a piece of text to mask an author's writing style. Below, we discuss such techniques. Homoglyph attack is an orthographic perturbation technique that changes the unicode of texts. It changes English characters to cyrillic characters. This attack is almost imperceptible to the human eye and therefore, preserves semantics. The robustness of *GPT-2 detector*, *GROVER detector*, and *GLTR* is evaluated by obfuscating texts generated by GPT-2 with the homoglyph attack. *GPT-2 detector*'s performance dropped from 97.44% to 0.26% Recall. The perturbed texts caused *GROVER detector* to grossly misclassify neural texts as human-written texts and *GLTR* to shift the distribu-

tion (i.e., color scheme) of the perturbed texts [114]. *GROVER detector* is further evaluated on obfuscated texts generated by GROVER [37]. Homoglyph attack achieves a 97% success rate in perturbing *GROVER detector* [37]. However, this AO technique can easily be rendered ineffective by using spell check algorithms [114].

## 4.2 Statistical Obfuscation

In order to extract statistical characteristics from neural texts, the following statistical AO techniques are implemented: (1) *Replace text with likelihood ranks*; (2) *Replace text with specific linguistic features, such as Part of Speech, Dependency Trees, Constituent Trees and Named Entities*; and (3) *Retain tokens in high/low frequency regions* which defines a frequency gap score to calculate and extract the high and low-frequency words in the text [90]. The following 3 datasets are perturbed - human-machine pairs, Writing Prompt dataset [33], and CnDARIO (Chinese Novel Dataset crawled from mixed online sources) generated with GPT-2 fine-tuned with Chinese Literature. These datasets are in 3 different domains, respectively - Online Forum, News, and Literature. Results suggest that the *high/low frequency region* perturbations is the most effective obfuscation technique [90]. This further suggests that the *high/low-frequency region* feature could be an effective feature for distinguishing neural texts from human texts.

*ruBERT* for NTD is evaluated on 2 AO techniques - (1) *calculating the class probabilities of each label and only selecting the texts with the highest human class probability*; (2) *adds the detector's log-probability to the beam search decoding strategy during generation* so that only more human-like texts are generated [85]. These attacks achieve 46% and 56% success rates, respectively. *RoBERTa-Defense* is evaluated by changing the sampling distribution of the neural texts in the test set. This obfuscation technique involves: (1) *varying the text decoding strategy (and its parameters)*; and (2) *varying the number of priming tokens* [91]. The quality of the obfuscated neural texts is measured by GRUEN [124], a metric used to measure the linguistic quality of AI-generated texts (neural texts). GRUEN has a high correlation with human judgments. The score ranges from 0–1, and a higher value indicates high linguistic quality. Linguistic quality is based on grammaticality, non-redundancy, discourse focus, structure, and coherence [91]. Using GRUEN, a *successful attack* is defined as an attack that degrades the performance of the AA models, with little to no linguistic quality (GRUEN score) degradation [91]. The results suggest that changing the decoding strategy is an effective AO technique. Even *GLTR*, a statistical AA model is fooled by this AO technique [91].

## 5. EVALUATION OF AA/AO METHODS

### 5.1 Machine-based Evaluation

#### 5.1.1 Authorship Attribution

To evaluate AA models, literature often used popular classification metrics such as *Precision*, *Recall*, *Accuracy*, and *F1 score*. For instance, [114] used the recall metric to evaluate the robustness of AA models toward AO techniques. Previous works evaluate the generalizability of AA models, not only on a standard single test set [36, 50, 52, 81, 91] but also on several out-of-sample distributions [102]. For example, [102] evaluate *GROVER detector* (Mega model) and *OpenAI detector* (base and large models) on three variants of test sets, namely *in-distribution*, *out-of-distribution* and *in-the-wild* datasets. *In-distribution* datasets are test sets

sampled from the same distribution of the training set, while *Out-of-distribution* datasets are those sampled from a different distribution from the training set. This test dataset is created using PPLM [25] and GeDi [63] to control the GROVER and GPT-2 generations [102]. The *in-the-wild* datasets are test sets generated from an NTG, different from the NTG used to generate a training set [102]. To build this *in-the-wild* dataset, training sets contain texts generated from GPT-2 and GROVER pre-trained models, while test sets contain texts generated from GPT-3 and a fine-tuned GPT-2 model [102]. In general, AA models perform well on *in-distribution* datasets, but suffer on *out-of-distribution* datasets, and even more on *in-the-wild* datasets.

### 5.1.2 Authorship Obfuscation

To evaluate AO models, literature often uses the *success rate* [37, 85], a fraction of successfully obfuscated (i.e., misclassified) texts which were accurately attributed prior to obfuscation. Another measure for AO models is *evasion rate* [91] which is the fraction of perturbed neural texts that evade the detection by an AA model. Next, due to the time and financial cost required to carry out a human-based evaluation, *MAUVE*, a metric that statistically emulates human judgments in terms of the linguistic quality (i.e., coherency) of neural texts vs. human-written texts, has been used on the AO problem [22]. That is, *MAUVE* is used as a substitute for human evaluation of obfuscated text vs. non-obfuscated text [22]. Furthermore, based on the strict definition of AO, it is sometimes important that the obfuscated text preserves the semantics of the original text. Hence, literature has measured the degradation of semantics between original and obfuscated texts, namely METEOR [6], Universal Sentence Encoder (USE) [15] Cosine similarity, and GRUEN [124]. These metrics all correlate with human judgments and a high score indicates an obfuscated text with well-preserved semantics.

## 5.2 Human-based Evaluation

There is currently no known human-based evaluation of AO models. The closest one is the machine-based simulation by MAUVE [22]. As such, in this section, we focus our review on the human-based evaluation of AA models, especially in the context of NTD.

### 5.2.1 Human Evaluation without Training

A set of research works recruited human participants (often from crowdsourcing platforms such as Amazon Mechanical Turk (AMT)) and tested whether they can distinguish neural texts from human-written texts. A simple introduction to the tasks is given, but no special training is done for human participants in this line of research. For instance, [117] examined the quality of texts generated by GROVER vs. human-written texts by humans and found that humans find GROVER-written news more believable than human-written news. [28] asked human participants to detect infilled texts filled by neural texts (e.g., she drank [blank] for [blank]) and found that humans had difficulty in detecting the infilled texts filled by neural texts. [108] introduced a benchmark for AA research, *TuringBench*, evaluated the performance of humans in distinguishing 19 pairs of human vs NTG (e.g., human vs. GPT-1 or human vs. FAIR) using TuringBench, and found that humans on average scored 51-54% of accuracies, only slightly better than random guessing. Unsurprisingly, [13] also found that humans were unable to accurately detect GPT-3 texts from human-written texts. [52] evaluated the quality of top-p and top-k decoding strategies, and found that (1) AA models detect neural texts generated by top-k decoding better than humans, but (2)

humans detect neural texts generated by top-p decoding better than AA models. Lastly, [3] found that both humans and AA models struggled to detect neural fake reviews.

### 5.2.2 Human Evaluation with Training

Another line of research attempted to first train human participants about NTD tasks and measure the performance improvements afterward. For instance, when human participants were trained to use GLTR [45] in detecting neural texts, thanks to the color scheme of GLTR (see Figure 6), human performance increased from 54% to 72% in accuracy. To further evaluate neural texts, [31] proposed a framework to collect a large number of human annotations via a game, Rofit<sup>15</sup>, on the quality of neural vs. human texts. Human participants were told to detect the boundary at which an article transitions from human-written to neurally-generated. Only 16% of human participants were able to correctly identify the accurate boundaries [31]. [18] studied three training strategies—instruction-based, example-based, and comparison-based, and found that example-based training is the most effective to improve human performance for solving NTD tasks (achieving the average accuracy of 55%) across the domains of story, recipe, and news. Next, [104] investigated how accurately humans can classify real vs. generated articles with and without images, using different types of news datasets: real captions and articles, real captions and generated articles, generated captions and real articles and generated captions and articles. By conducting an AMT-based study, untrained and trained human participants were able to achieve an average of 46% and 68%, respectively. Further investigation on the trustworthiness of the different article types based on style, content, consistency, and overall trustworthiness reveals that humans were skeptical about the overall trustworthiness of news articles across all four types [104]. Finally, recently, [29] proposed a framework for scrutinizing neural texts through crowdsourced data annotation in a scalable fashion, where neural texts were shown to have various error types: language-errors (i.e., lack of coherency and consistency in text), factual-errors, and reader-issues (i.e., text is too obscure or filled with too much jargon so that understanding is negatively impacted).

In conclusion, literature has found that humans alone cannot detect neural texts accurately, achieving detection accuracies only slightly better than random guessing. When humans are properly trained about the characteristics of neural texts, further, this detection accuracy tends to increase but only by small margins.

## 6. OPEN CHALLENGES

Although there have been several meaningful works on the current landscape of AA and AO models, the two research problems are still in their early development, especially the direction of NTD. In this section, as such, we discuss some of the remaining challenges.

### 6.1 Need for Comprehensive Benchmark

Generally, existing literature tends to create or use particular datasets in silos, making their findings limited and incomparable across the literature. As mentioned in [36, 81], however, the study of NTD can be greatly improved with the availability of more comprehensive and generalizable datasets whose coverage varies across diverse: (1) domains (e.g., news, online forum, recipe, stories), (2) language models, (3) decoding strategies, or (4) length of texts.

<sup>15</sup><http://rofit.io/>

Further, not all AA/AO models share their codebase and experimental configurations, making the comparative analysis difficult. However, generating and maintaining a large number of neural texts across different settings cost significant resources and effort. [108] attempted to propose a benchmark for AA research, *TuringBench*, but it does not satisfy the needs fully. Therefore, it is greatly needed to develop a comprehensive benchmark with diverse datasets of AA/AO problems, along with the codebase of known methods in a unified environment, so that objective comparison can be carefully performed to understand the pros and cons of existing solutions and brainstorm new ideas for improvement.

## 6.2 Call for Complex AA/AO Variations

With the introduction of “machine” in writing high-quality texts, the set-up of “authors” in future scenarios can be more complex. For instance, one could generate a more realistic text using multiple language models in sequence (e.g., each language model improves upon the text generated by another language model in a previous step) or in parallel (e.g., each language model generates only parts of a long text). Symmetrically, it is also plausible to use multiple AO solutions in sequence or parallel to improve the overall performance of obfuscation. Yet another possible scenario is to think of “human-in-the-loop” attribution or obfuscation. For instance, would a team (of humans, of machines, or of humans and machines) outperform an individual (of human or machine) in solving AA or AO task? To our best knowledge, there is no study of AA/AO for such complex scenarios.

## 6.3 Need for Interpretable AA/AO Models

Currently, there are only a few interpretable AA models (e.g., GLTR) and AO techniques (e.g., Homoglyph) for neural texts, as summarized in Tables 3 and 4, respectively. That is, when an AA model detects a text as machine-generated or human-written, or when an AO model modifies parts of a text to hide authorship, it often cannot explain “why?” Ideally, however, such models should be able to provide an easy-to-understand and intuitive explanation, especially to users with no linguistic expertise, as to why a given text is attributable to a particular NTG or why a particular phrase of a text is critical to reveal an author’s identity. In addition, more research is needed to develop an intuitive human interface or visualization toward explainable AA/AO models.

## 6.4 Need for Improved Human Training

In parallel to improving the performance of AA/AO solutions, it is equally important to raise the awareness of AA/AO problems in the presence of neural texts, and to be able to train human users to detect neural texts better (e.g., identify phishing or misinformation message that includes neural texts as parts) or use AO solutions to hide one’s authorship (e.g., an activist posting his/her message on social media without revealing true identity). As we illustrate in Section 5.2, however, humans are not good at detecting neural texts, and there are not many AA/AO solutions suitable for novice users to benefit from in solving AA/AO tasks. Worst, still, is that even a few AA models such as *GLTR* that were shown to be able to help human users to detect neural texts better have become less effective with the advancement of NTGs. Therefore, great needs exist to have a better way to train human users in solving AA/AO tasks.

## 6.5 Call for Robust AA/AO Solutions

In section 4, we surveyed all literature that evaluated the robustness of AA/AO models, and found that most existing AO techniques do not preserve the original semantics of text well and thus cannot easily evade the attribution of AA solutions, especially human detection. Similarly, as we adopt more sophisticated hybrid approaches for AA tasks, successful AO attacks to hide authorship will become more challenging. Part of the reason for these vulnerabilities in existing AA/AO solutions is that the bulk of existing literature has studied either AA or AO problem in separation, thus greatly limiting their robustness against the other problem. Therefore, to stay relevant and synonymous with a real-life scenario, both AA and AO solutions need to learn from each other, and co-train/co-evolve, as in a min-max optimization game.

## 7. APPLICATIONS

**Deepfake Detection:** Successful solutions for AA/AO tasks can be useful in many applications. For instance, recently, the generation of realistic AI-made images<sup>16</sup> and videos, so-called “deepfakes”, have flooded the Web. While most of these deepfakes are made for humor, some are malicious in generating misinformation, spreading political propaganda, or attacking individuals [79]. In literature, in particular, [104] studies the realistic scenario where real images would be paired with neural texts to increase the authenticity of a news article as well as evade detection. In such a setting, successful AA solutions can point out the non-human nature of neural texts to users or can be used to extract features of neural texts for downstream deepfake detection models.

**Chatbot Detection:** Another application is for AA solutions to detect suspicious messages (e.g., phishing or chatbot messages) that may have been (partially) generated by NTG. Similar to neural texts of news format, shorter or informal chatbot messages are also hard to discriminate when generated by machines [98, 106]. An example of the state-of-the-art chatbot is ChatGPT [84] which has been used to generate medical writings [10, 40], finance writings [30], etc. These applications of ChatGPT have also increased the likelihood of cheating in academic writing [21]. Thus, AA models for neural text detection will be beneficial in distinguishing chatGPT-generated texts from human-written texts.

**Anonymity Preservation:** On the other hand, successful AO solutions can be used to help individuals who have needs to share their writings without jeopardizing their secret identity. For instance, an NGO activist or whistleblower may submit her op-ed to news media after making sure that no popular AA solutions can attribute the writing to her.

## 8. CONCLUSION

With the rise of neural texts that were generated by large-scale language models, we are currently in an arms race between generation and detection of *neural texts*. In this work, we comprehensively survey two important problems of neural texts: **Authorship Attribution** (AA) and **Authorship Obfuscation** (AO). We first categorize existing AA solutions into four types of stylometric, deep learning-based, statistical, and hybrid attribution. Similarly, we categorize existing AO solutions into two types of stylometric and statistical obfuscation, and elaborate pros and cons of representative methods therein. In addition, we discuss different evaluation methods for AA and AO problems in the context of neural texts, and finally, share a few important challenges that we

<sup>16</sup><https://thispersondoesnotexist.com/>



feel lacking currently. By and large, we believe that the data mining community is well-positioned to be able to contribute to significant improvement in both AA and AO research. Despite their close implications in security and privacy, with respect to the underlying methods used, their problem formulation as supervised or unsupervised learning, and their focus on the accuracy and running time as major metrics. Lastly, to mitigate the challenges of accurate detection of neural text, [62] proposes watermarking these text-generative language models. This entails embedding humanly imperceptible signals into the language models such that they generate semantically relevant texts, unnoticeable to humans but noticeable to detectors. These watermarking [2, 62] techniques attempt to solve the security risks that these language models pose. However, as these watermarking techniques have not yet been widely adopted, we still have to rely on AA and AO solutions for neural text detection. Also, as such watermarking techniques are a recent/new development, their robustness to strong AO techniques has not yet been evaluated.

## 9. ACKNOWLEDGMENT

This work was in part supported by NSF awards #1820609, #2114824, and #2131144.

## References

- [1] A. Abbasi and H. Chen. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems (TOIS)*, 26(2):1–29, 2008.
- [2] S. Abdelnabi and M. Fritz. Adversarial watermarking transformer: Towards tracing text provenance with data hiding. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 121–140. IEEE, 2021.
- [3] D. I. Adelani, H. Mai, F. Fang, H. H. Nguyen, J. Yamagishi, and I. Echizen. Generating sentiment-preserving fake online reviews using neural language models and their human-and machine-based detection. In *International Conference on Advanced Information Networking and Applications*, pages 1341–1354. Springer, 2020.
- [4] B. Ai, Y. Wang, Y. Tan, and T. Samson. Whodunit? learning to contrast for authorship attribution. *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing*, 2022.
- [5] A. Bakhtin, S. Gross, M. Ott, Y. Deng, M. Ranzato, and A. Szlam. Real or fake? learning to discriminate machine from human generated text. *arXiv preprint arXiv:1906.03351*, 2019.
- [6] S. Banerjee and A. Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.
- [7] P. Barham, A. Chowdhery, J. Dean, S. Ghemawat, S. Hand, D. Hurt, M. Isard, H. Lim, R. Pang, S. Roy, et al. Pathways: Asynchronous distributed dataflow for ml. *Proceedings of Machine Learning and Systems*, 4:430–449, 2022.
- [8] J. Bevendorff, B. Chulvi, E. Fersini, A. Heini, M. Kestemont, K. Kredens, M. Mayerl, R. Ortega-Bueno, P. Pezik, M. Potthast, et al. Overview of pan 2022: Authorship verification, profiling irony and stereotype spreaders, style change detection, and trigger detection. In *European Conference on Information Retrieval*, pages 331–338. Springer, 2022.
- [9] M. M. Bhat and S. Parthasarathy. How effectively can machines defend against machine-generated fake news? an empirical study. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pages 48–53, 2020.
- [10] S. Biswas. Chatgpt and the future of medical writing, 2023.
- [11] A. F. Biten, L. Gomez, M. Rusinol, and D. Karatzas. Good news, everyone! context driven entity-aware captioning for news images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12466–12475, 2019.
- [12] S. Black, S. Biderman, E. Hallahan, Q. Anthony, L. Gao, L. Golding, H. He, C. Leahy, K. McDonnell, J. Phang, et al. Gpt-neox-20b: An open-source autoregressive language model. In *Proceedings of BigScience Episode\#5-Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, 2022.
- [13] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- [14] B. Buchanan, A. Lohn, M. Musser, and K. Sedova. Truth, lies, and automation. *Center for Security and Emerging Technology*, 2021.
- [15] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, et al. Universal sentence encoder for english. In *Proceedings of the 2018 conference on empirical methods in natural language processing: system demonstrations*, pages 169–174, 2018.
- [16] P.-J. Chen, A. Lee, C. Wang, N. Goyal, A. Fan, M. Williamson, and J. Gu. Facebook ai’s wmt20 news translation task submission. In *Proceedings of the Fifth Conference on Machine Translation*, pages 113–125, 2020.
- [17] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.
- [18] E. Clark, T. August, S. Serrano, N. Haduong, S. Gururangan, and N. A. Smith. All that’s ‘human’ is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, 2021.

- [19] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.
- [20] A. Conneau and G. Lample. Cross-lingual language model pretraining. *Advances in neural information processing systems*, 32, 2019.
- [21] D. R. Cotton, P. A. Cotton, and J. R. Shipway. Chatting and cheating. ensuring academic integrity in the era of chatgpt. 2023.
- [22] E. Crothers, N. Japkowicz, H. Viktor, and P. Branco. Adversarial robustness of neural-statistical features in detection of generative transformers. *arXiv preprint arXiv:2203.07983*, 2022.
- [23] J. Cutler, L. Dugan, S. Havaldar, and A. Stein. Automatic detection of hybrid human-machine text boundaries. 2021.
- [24] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. Le, and R. Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, 2019.
- [25] S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, and R. Liu. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*, 2020.
- [26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [27] N. Diwan, T. Chakraborty, and Z. Shafiq. Fingerprinting fine-tuned language models in the wild. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4652–4664, 2021.
- [28] C. Donahue, M. Lee, and P. Liang. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, 2020.
- [29] Y. Dou, M. Forbes, R. Koncel-Kedziorski, N. A. Smith, and Y. Choi. Scarecrow: A framework for scrutinizing machine text. *arXiv preprint arXiv:2107.01294*, 2021.
- [30] M. Dowling and B. Lucey. Chatgpt for (finance) research: The bananarama conjecture. *Finance Research Letters*, page 103662, 2023.
- [31] L. Dugan, D. Ippolito, A. Kirubarajan, and C. Callison-Burch. Rofit: A tool for evaluating human detection of machine-generated text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 189–196, 2020.
- [32] T. Fagni, F. Falchi, M. Gambini, A. Martella, and M. Tesconi. Tweepfake: About detecting deepfake tweets. *Plos one*, 16(5):e0251415, 2021.
- [33] A. Fan, M. Lewis, and Y. Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, 2018.
- [34] E. Fast, B. Chen, and M. S. Bernstein. Empath: Understanding topic signals in large-scale text. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 4647–4657, 2016.
- [35] W. Fedus, B. Zoph, and N. Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.
- [36] L. Fröhling and A. Zubiaga. Feature-based detection of automated language models: tackling gpt-2, gpt-3 and grover. *PeerJ Computer Science*, 7:e443, 2021.
- [37] R. Gagiano, M. M.-H. Kim, X. J. Zhang, and J. Biggs. Robustness analysis of grover for machine-generated news detection. In *Proceedings of the The 19th Annual Workshop of the Australasian Language Technology Association*, pages 119–127, 2021.
- [38] M. Gallé, J. Rozen, G. Kruszewski, and H. Elsahar. Unsupervised and distributional detection of machine-generated text. *arXiv preprint arXiv:2111.02878*, 2021.
- [39] M. Gambini, T. Fagni, F. Falchi, and M. Tesconi. On pushing deepfake tweet detection capabilities to the limits. In *14th ACM Web Science Conference 2022*, pages 154–163, 2022.
- [40] C. A. Gao, F. M. Howard, N. S. Markov, E. C. Dyer, S. Ramesh, Y. Luo, and A. T. Pearson. Comparing scientific abstracts generated by chatgpt to original abstracts using an artificial intelligence output detector, plagiarism detector, and blinded human reviewers. *bioRxiv*, pages 2022–12, 2022.
- [41] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE, 2018.
- [42] L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.
- [43] T. Gao, X. Yao, and D. Chen. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, 2021.
- [44] S. Gehman, S. Gururangan, M. Sap, Y. Choi, and N. A. Smith. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, 2020.
- [45] S. Gehrmann, H. Strobel, and A. M. Rush. Gltr: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116, 2019.
- [46] J. Guerrero and I. Alsmadi. Synthetic text detection: Systemic literature review. *arXiv preprint arXiv:2210.06336*, 2022.

- [47] M. Haroon, F. Zaffar, P. Srinivasan, and Z. Shafiq. Avengers ensemble! improving transferability of authorship obfuscation. *arXiv preprint arXiv:2109.07028*, 2021.
- [48] P. He, J. Gao, and W. Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.
- [49] P. He, X. Liu, J. Gao, and W. Chen. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*, 2021.
- [50] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2019.
- [51] Huggingface. Gpt-2 output detector demo. <https://huggingface.co/openai-detector/>, 2019.
- [52] D. Ippolito, D. Duckworth, C. Callison-Burch, and D. Eck. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822, 2020.
- [53] F. Jafariakinabad, S. Tarnpradab, and K. A. Hua. Syntactic recurrent neural network for authorship attribution. *arXiv preprint arXiv:1902.09723*, 2019.
- [54] G. Jawahar, M. Abdul-Mageed, and L. Lakshmanan. Automatic detection of entity-manipulated text using factual knowledge. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 86–93, 2022.
- [55] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025, 2020.
- [56] K. Jones, J. R. Nurse, and S. Li. Are you robert or roberta? deceiving online authorship attribution models using neural text generators. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 16, pages 429–440, 2022.
- [57] G. Karadzhov, T. Mihaylova, Y. Kiprova, G. Georgiev, I. Koychev, and P. Nakov. The case for being average: A mediocrity approach to style masking and author obfuscation. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 173–185. Springer, 2017.
- [58] R. Kaur, S. Singh, and H. Kumar. Authorship analysis of online social media content. In *Proceedings of 2nd International Conference on Communication, Computing and Networking*, pages 539–549. Springer, 2019.
- [59] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.
- [60] M. Kestemont, E. Manjavacas, I. Markov, J. Bevendorff, M. Wiegmann, E. Stamatatos, B. Stein, and M. Potthast. Overview of the cross-domain authorship verification task at pan 2021. In *CLEF (Working Notes)*, 2021.
- [61] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
- [62] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein. A watermark for large language models. *arXiv preprint arXiv:2301.10226*, 2023.
- [63] B. Krause, A. D. Gotmare, B. McCann, N. S. Keskar, S. Joty, R. Socher, and N. F. Rajani. Gedi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952, 2021.
- [64] B. Krause, L. Lu, I. Murray, and S. Renals. Multiplicative lstm for sequence modelling. *arXiv preprint arXiv:1609.07959*, 2016.
- [65] B. Krause, I. Murray, S. Renals, and L. Liang. Multiplicative lstm for sequence modelling. In *5th International Conference on Learning Representations*, pages 2872–2880, 2017.
- [66] S. Kreps, R. M. McCain, and M. Brundage. All the news that’s fit to fabricate: Ai-generated text as a tool of media misinformation. *Journal of Experimental Political Science*, 9(1):104–117, 2022.
- [67] L. Kushnareva, D. Cherniavskii, V. Mikhailov, E. Artemova, S. Barannikov, A. Bernstein, I. Piontkovskaya, D. Piontkovski, and E. Burnaev. Artificial text detection via examining the topology of attention maps. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 635–649, 2021.
- [68] K. Lagutina, N. Lagutina, E. Boychuk, I. Vorontsova, E. Shliakhtina, O. Belyaeva, I. Paramonov, and P. Demidov. A survey on stylometric text features. In *2019 25th Conference of Open Innovations Association (FRUCT)*, pages 184–195. IEEE, 2019.
- [69] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.
- [70] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.
- [71] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020.
- [72] J. Li, T. Tang, W. X. Zhao, and J.-R. Wen. Pretrained language models for text generation: A survey. *arXiv preprint arXiv:2105.10311*, 2021.
- [73] X. Liu, Z. Zhang, Y. Wang, Y. Lan, and C. Shen. Coco: Coherence-enhanced machine-generated text detection under data limitation with contrastive learning. *arXiv preprint arXiv:2212.10341*, 2022.
- [74] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

- [75] V. Liyanage, D. Buscaldi, and A. Nazarenko. A benchmark corpus for the detection of automatically generated text in academic publications. In *LREC*, 2022.
- [76] A. Mahmood, F. Ahmad, Z. Shafiq, P. Srinivasan, and F. Zaffar. A girl has no name: Automated authorship obfuscation using mutant-x. *Proc. Priv. Enhancing Technol.*, 2019(4):54–71, 2019.
- [77] A. Mahmood, Z. Shafiq, and P. Srinivasan. A girl has a name: Detecting authorship obfuscation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2235–2245, 2020.
- [78] A. W. McDonald, S. Afroz, A. Caliskan, A. Stolerman, and R. Greenstadt. Use fewer instances of the letter “i”: Toward writing style anonymization. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 299–318. Springer, 2012.
- [79] Y. Mirsky and W. Lee. The creation and detection of deep-fakes: A survey. *ACM Computing Surveys (CSUR)*, 54(1):1–41, 2021.
- [80] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, and C. Finn. Detectgpt: Zero-shot machine-generated text detection using probability curvature. *arXiv preprint arXiv:2301.11305*, 2023.
- [81] S. Munir, B. Batool, Z. Shafiq, P. Srinivasan, and F. Zaffar. Through the looking glass: Learning to attribute synthetic text generated by language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1811–1822, 2021.
- [82] N. Ng, K. Yee, A. Baevski, M. Ott, M. Auli, and S. Edunov. Facebook fair’s wmt19 news translation task submission. *arXiv preprint arXiv:1907.06616*, 2019.
- [83] H.-Q. Nguyen-Son, N.-D. T. Tieu, H. H. Nguyen, J. Yamagishi, and I. E. Zen. Identifying computer-generated text using statistical analysis. In *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1504–1511. IEEE, 2017.
- [84] OpenAI. Optimizing language models for dialogue. <https://openai.com/blog/chatgpt/>, 2022.
- [85] M. Orzhenovskii. Detecting auto-generated texts with language model and attacking the detector. *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference “Dialogue 2022*, 2022.
- [86] A. Pagnoni, M. Graciarena, and Y. Tsvetkov. Threat scenarios and best practices to detect neural fake news. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1233–1249, 2022.
- [87] J. W. Pennebaker, M. E. Francis, and R. J. Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71(2001):2001, 2001.
- [88] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.
- [89] K. Pillutla, S. Swayamdipta, R. Zellers, J. Thickstun, S. Welleck, Y. Choi, and Z. Harchaoui. An information divergence measure between neural text and human text. *arXiv preprint arXiv:2102.01454*, 2021.
- [90] J. Pu, Z. Huang, Y. Xi, G. Chen, W. Chen, and R. Zhang. Unraveling the mystery of artifacts in machine generated text. pages 6889–6898, 2022.
- [91] J. Pu, Z. Sarwar, S. M. Abdullah, A. Rehman, Y. Kim, P. Bhattacharya, M. Javed, B. Viswanath, V. Tech, and L. Pakistan. Deepfake text detection: Limitations and opportunities. *44th IEEE Symposium on Security and Privacy*, 2023.
- [92] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. URL [https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/language\\_understanding\\_paper.pdf](https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/language_understanding_paper.pdf), 2018.
- [93] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [94] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.
- [95] D. Rosati. Synscipass: detecting appropriate uses of scientific text generation. In *Proceedings of the Third Workshop on Scholarly Document Processing*, pages 214–222, 2022.
- [96] T. Schuster, R. Schuster, D. J. Shah, and R. Barzilay. The limitations of stylometry for detecting machine-generated fake news. *Computational Linguistics*, 46(2):499–510, 2020.
- [97] T. Shamardina, V. Mikhailov, D. Chernianskii, A. Fenogenova, M. Saidov, A. Valeeva, T. Shavrina, I. Smurov, E. Tutubalina, and E. Artemova. Findings of the the ruatd shared task 2022 on artificial text detection in russian. *arXiv preprint arXiv:2206.01583*, 2022.
- [98] J. Shao, A. Uchendu, and D. Lee. A reverse turing test for detecting machine-made texts. In *Proceedings of the 10th ACM Conference on Web Science*, pages 275–279, 2019.
- [99] P. Shrestha, S. Sierra, F. Gonzalez, M. Montes, P. Rosso, and T. Solorio. Convolutional neural networks for authorship attribution of short texts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 2017.
- [100] E. Stamatatos. Authorship verification: a review of recent advances. *Research in Computing Science*, 123:9–25, 2016.
- [101] E. Stamatatos, M. Kestemont, K. Kredens, P. Pezik, A. Heini, J. Bevendorff, M. Potthast, and B. Stein. Overview of the authorship verification task at pan 2022. *Working Notes of CLEF*, 2022.
- [102] H. Stiff and F. Johansson. Detecting computer-generated disinformation. *International Journal of Data Science and Analytics*, 13(4):363–383, 2022.

- [103] M. Tabatabaei, J. Hakanen, M. Hartikainen, K. Miettinen, and K. Sindhya. A survey on handling computationally expensive multiobjective optimization problems using surrogates: non-nature inspired methods. *Structural and Multidisciplinary Optimization*, 52(1):1–25, 2015.
- [104] R. Tan, B. Plummer, and K. Saenko. Detecting cross-modal inconsistency to defend against neural fake news. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2081–2106, 2020.
- [105] J. Tyo, B. Dhingra, and Z. C. Lipton. On the state of the art in authorship attribution and authorship verification. *arXiv preprint arXiv:2209.06869*, 2022.
- [106] A. Uchendu, J. Cao, Q. Wang, B. Luo, and D. Lee. Characterizing man-made vs. machine-made chatbot dialogs. In *TTO*, 2019.
- [107] A. Uchendu, T. Le, K. Shu, and D. Lee. Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395, 2020.
- [108] A. Uchendu, Z. Ma, T. Le, R. Zhang, and D. Lee. Turing-bench: A benchmark environment for turing test in the age of neural text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2001–2016, 2021.
- [109] O. Varol, E. Ferrara, C. Davis, F. Menczer, and A. Flammini. Online human-bot interactions: Detection, estimation, and characterization. In *Proceedings of the international AAAI conference on web and social media*, volume 11, pages 280–289, 2017.
- [110] L. R. Varshney, N. S. Keskar, and R. Socher. Limits of detecting text generated by large-scale language models. In *2020 Information Theory and Applications Workshop (ITA)*, pages 1–5. IEEE, 2020.
- [111] B. Wang. Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX. <https://github.com/kingoflolz/mesh-transformer-jax>, May 2021.
- [112] B. Wang and A. Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. <https://github.com/kingoflolz/mesh-transformer-jax>, May 2021.
- [113] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al. Huggingface’s transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.
- [114] M. Wolff and S. Wolff. Attacking neural text detectors. *arXiv preprint arXiv:2002.11768*, 2020.
- [115] E. Xing, T. Le, and D. Lee. Alison: Fast stylometric authorship obfuscation. *Technical Report*, 2023.
- [116] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764, 2019.
- [117] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, pages 9051–9062, 2019.
- [118] W. Zhai, J. Rusert, Z. Shafiq, and P. Srinivasan. A girl has a name, and it’s... adversarial authorship attribution for de-obfuscation. *arXiv preprint arXiv:2203.11849*, 2022.
- [119] H. Zhang, H. Song, S. Li, M. Zhou, and D. Song. A survey of controllable text generation using transformer-based pre-trained language models. *arXiv preprint arXiv:2201.05337*, 2022.
- [120] R. Zhang, Z. Hu, H. Guo, and Y. Mao. Syntax encoding with application in authorship attribution. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2742–2753, 2018.
- [121] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.
- [122] W. E. Zhang, Q. Z. Sheng, A. Alhazmi, and C. Li. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41, 2020.
- [123] W. Zhong, D. Tang, Z. Xu, R. Wang, N. Duan, M. Zhou, J. Wang, and J. Yin. Neural deepfake detection with factual structure of text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2461–2470, 2020.
- [124] W. Zhu and S. Bhat. Gruen for evaluating linguistic quality of generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 94–108, 2020.