ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240647

TopFormer: Topology-Aware Authorship Attribution of Deepfake Texts with Diverse Writing Styles

Adaku Uchendu[a,*], **Thai Le**[b] **and Dongwon Lee**[c]

[a]MIT Lincoln Laboratory, Lexington, MA, USA
[b]Indiana University, Bloomington, IN, USA
[c]The Pennsylvania State University, University Park, PA, USA

Abstract. Recent advances in Large Language Models (LLMs) have enabled the generation of open-ended high-quality texts, that are non-trivial to distinguish from human-written texts. We refer to such LLM-generated texts as *deepfake texts*. There are currently over 72K text generation models in the huggingface model repo. As such, users with malicious intent can easily use these open-sourced LLMs to generate harmful texts and dis/misinformation at scale. To mitigate this problem, a computational method to determine if a given text is a deepfake text or not is desired–i.e., Turing Test (TT). In particular, in this work, we investigate the more general version of the problem, known as *Authorship Attribution (AA)*, in a multi-class setting–i.e., not only determining if a given text is a deepfake text or not but also being able to pinpoint which LLM is the author. We propose **TopFormer** to improve existing AA solutions by capturing more linguistic patterns in deepfake texts by including a Topological Data Analysis (TDA) layer in the Transformer-based model. We show the benefits of having a TDA layer when dealing with imbalanced, and multi-style datasets, by extracting TDA features from the reshaped $pooled\_output$ of our backbone as input. This Transformer-based model captures contextual representations (i.e., semantic and syntactic linguistic features), while TDA captures the shape and structure of data (i.e., linguistic structures). Finally, **TopFormer**, outperforms all baselines in all 3 datasets, achieving up to 7% increase in Macro F1 score. Our code and datasets are available at: https://github.com/AdaUchendu/topformer

1 Introduction

Recent Large Language Models (LLMs) now have a trillion parameters and are able to generate more human-like texts. These larger models pose a few difficulties, the more glaring being that reproducibility is very difficult and expensive. This allows LLMs to remain black-box, with their limitations maliciously exploited. Some of these limitations include: toxic and hate speech generation [34], plagiarism [21], memorization of sensitive information [3] and hallucinated text generation [30] which allow LLMs to be easily exploited to generate authentic-looking and convincing disinformation [23].

The first step to mitigate these limitations of LLMs starts with our ability to determine whether a text is generated by a particular LLM (**deepfake text**[1]) or not. This task is known as *Turing Test (TT)* [41].

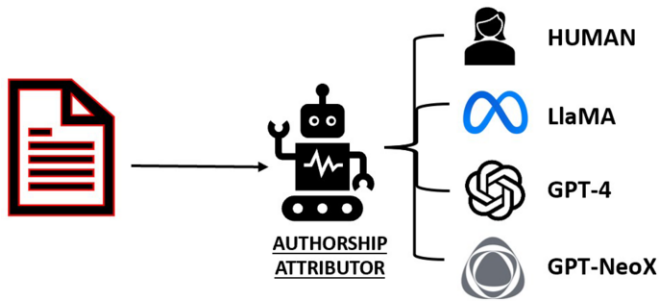


Figure 1: Illustration of the Authorship Attribution (AA) problem with multiple authors - human and many deepfake (LLM) authors.

Further, in this work, generalizing the TT problem further, we are interested in not only determining if a text is a deepfake text or not but also pinpointing which LLM is the author, known as the **Authorship Attribution** (AA) problem [42]. See the illustration of AA in Figure 1. The AA problem in a multi-class setting has not been as rigorously studied as the TT problem has. Naturally, AA is substantially more challenging than TT. However, with the ever-increasing number of popular LLMs that people can use, we believe that it is no longer sufficient to just ask the binary question of "is this written by human or machine?" Solving the AA problem enables more fine-grained and targeted defense tools for users and platforms (e.g., a detector specifically trained and designed to detect ChatGPT deepfake texts). AA solutions can also help researchers and policymakers understand the capacity and limitations of different LLMs, and study which LLMs are more vulnerable to misuse and abuse in what context (e.g., political propaganda, terrorism recruitment, phishing).

Researchers have proposed several solutions to distinguish deepfake texts from human-written texts, utilizing supervised and unsupervised machine learning [42]. In the supervised learning setting, researchers have developed *stylometric*, *deep learning*, and *hybrid* solutions for detecting deepfake texts. Further, in the unsupervised learning setting, *statistical* solutions such as watermarking approaches [15] have been developed. Intuitively, deep learning and hybrid-based techniques achieve the best performance in terms of accuracy. However, in terms of adversarial robustness, statistical-based (i.e., threshold-guided metric-based) techniques are the most robust models, with hybrid models taking second/first place in adversarial robustness [42]. To that end, we propose a hybrid solution which

* Corresponding Author. Email: adaku.uchendu@ll.mit.edu.
[1] We call these LLM-generated texts, *deepfake texts*; However, there are other terms which include - *AI-generated, auto-generated, machine-generated, synthetic, artificial, computer-generated, neural, and LLM-generated texts*

is an ensemble of statistical and deep learning-based techniques to get both benefits - good performance and robustness. We hypothesize that if our model has adversarial robustness properties, it could also be noise-resistant and thus be robust to out-of-distribution and imbalanced datasets.

Thus, we propose **TopFormer**, an ensemble of a Transformer-based (RoBERTa) model and Topological Data Analysis (TDA) techniques. We specifically choose RoBERTa [22] as the base model due to its state-of-the-art (SOTA) performance in feature extraction from text and its larger vocabulary compared to other transformer models such as BERT [8]. TDA is applied for deepfake text detection as it captures the true shape of data amidst noise [28, 29, 26, 38]. Limited access to SOTA LLMs leads to imbalanced and noisy datasets which hamper the ability to train better deepfake text detectors. The evolving nature of deepfake texts, influenced by instruction-tuned LLMs and diverse prompts [7], poses challenges in authorship attribution. In addition, people use LLMs to do a mirage of tasks, including generation, paraphrasing, editing, and summarizing a piece of text. All these factors increase the variability of deepfake texts in our information ecosystem and the label imbalance between deepfake texts and human-written texts. Therefore, we hypothesize that **TopFormer** can address this variability, achieving high performance on datasets reflecting current trends.

2 Related Work

2.1 Authorship Attribution of Deepfake Texts

Since 2019, there have been several efforts to mitigate the malicious uses of LLM-generated texts (deepfake texts) by way of detection. As this task is non-trivial, different techniques have been attempted and can be split into - *stylometric*, *deep learning*, *statistical-based*, and *hybrid classifiers* (ensemble of two or more of the previous types) and more recently *prompt-based*, as well as *human-based approaches* [42]. Furthermore, this task which is modeled as an Authorship Attribution (AA) task of human vs. either one or several deepfake text generators are studied as either binary (*Turing Test*) or multi-class problem. The most popular is the binary class problem as there is more data for it than for multi-class.

Thus, for the *stylometric* solution, several solutions are using POS tags, lexical dictionaries, etc. to capture an author's unique writing style [40, 18]. Next for the *deep learning* solutions, researchers usually fine-tune BERT models and BERT variants [41, 14, 17, 35, 1]. However, as fine-tuning can be computationally expensive and requires a lot of data which does not always exist, some researchers have proposed an unsupervised technique - *statistical-based* solutions [10, 25]. However, while deep learning-based classifiers perform very well, they are highly susceptible to adversarial perturbations [42]. Therefore, researchers propose *hybrid-based* solutions that fuse both *deep learning* and *statistical-based* or *stylometric* solutions to improve performance and robustness [20, 5]. More recently, *prompt-based* techniques are being used for attribution of deepfakes [16, 19, 43]. Lastly, for the *human-based* approaches, to improve human detection of deepfake texts, researchers have utilized two main techniques - training [10, 43] and not training [14].

2.2 TDA Applications in NLP

Topological Data Analysis (TDA) is a technique often used to quantify shape and structure in data by leveraging concepts from algebraic topology. Due to this unique ability to obtain the true shape of data, in spite of noise, it has been implemented in machine learning problems. The NLP field has recently seen a recent uptake in TDA applications due to its benefits. TDA has been previously applied to a variety of Machine learning tasks [13], such as detecting children and adolescent writing [49], novelists attribution [11], law documents analysis [33], movie genre analysis [9], and explanation of syntactic structures of different language families [28, 29]. More recently, TDA techniques have been applied to the deepfake text detection problem [20]. However, they collect the statistical summaries of the TDA representations of BERT attention weights represented as a threshold-guided directed and undirected graph. Using these representations, they classify deepfake texts with Logistic regression for the binary task - human vs. deepfake. Next, Perez and Reinauer [27] uses a similar technique as Kushnareva et al. [20] to show that TDA can improve the robustness of BERT. Finally, TDA has also been applied to representing documents as story trees [12], detecting contradictions in texts [47], examining the linguistic acceptability judgments of texts [6], finding loops in logic [39], speech processing [37], and extracting dialogue terms with Transfer learning [45].

3 Topological Data Analysis (TDA) Features

Topology is defined as "the study of geometric properties and spatial relations unaffected by the continuous change of shape or size of figures," according to the Oxford Dictionary. Topological Data Analysis (TDA) is a "collection of powerful tools that have the ability to quantify shape and structure in data"[2]. There are two main TDA techniques - persistent homology and mapper. We will only focus on persistent homology. Persistent homology is a TDA technique used to find topological patterns of the data [39].

This technique takes in the data and represents it as a point cloud, such that each point is enclosed by a circle. For this analysis, the aim is to extract the persistent features of the data using simplicial complexes. These formations extract features which are holes in different dimensions, represented as *betti numbers* (β_d, d-dimension). The holes in 0-Dimension (β_0), 1-Dimension (β_1) and 2-Dimension (β_2), are called connected components, loops/tunnels, and voids, respectively.

Finally, the TDA features recorded are the $birth$ (formation of holes), $death$ (deformation or the closing of holes), and persistence features in different dimensions. Persistence is defined as the length of time it took a feature to die ($death - birth$). This means that if a point touches another point then one of the points/features has died. The $death$ is recorded with the radii value at which the points overlap. In addition, due to all the shifts and changes, from the 1-Dimension and upwards, some features may appear - a new hole, and this feature is recorded as a $birth$. The $birth$ feature is the radii at which it appeared.

4 TopFormer: Topology-Aware Attributor

To build this TDA-infused Transformer-based model, we focus on the four layers needed to convert the vanilla Transformer-based (RoBERTa) model to a **Top**ological Trans**former**-based model (**TopFormer**) - (1) pre-trained weights of the backbone model, (2) dropout layer with probability p=0.3, (3) Topological layer for calculating, and (4) Linear transformation layer. See Figure 2 for a flow chart describing the architecture of **TopFormer** with the 4 layers and Algorithm 1 for the pseudocode for (**TopFormer**).

[2] https://www.indicative.com/resource/topological-data-analysis/

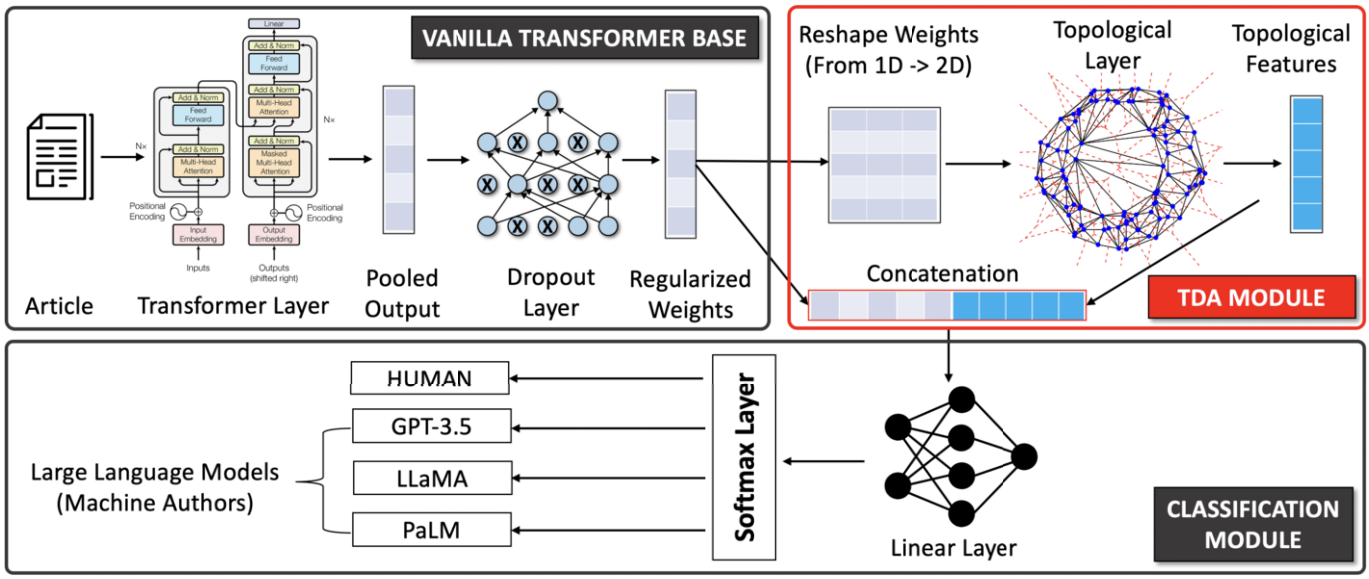


Figure 2: Flowchart of the Topological classification algorithm. The Red frame indicates our methodology and technique to transform a Vanilla Transformer-based model to a Topological Transformer-based model.

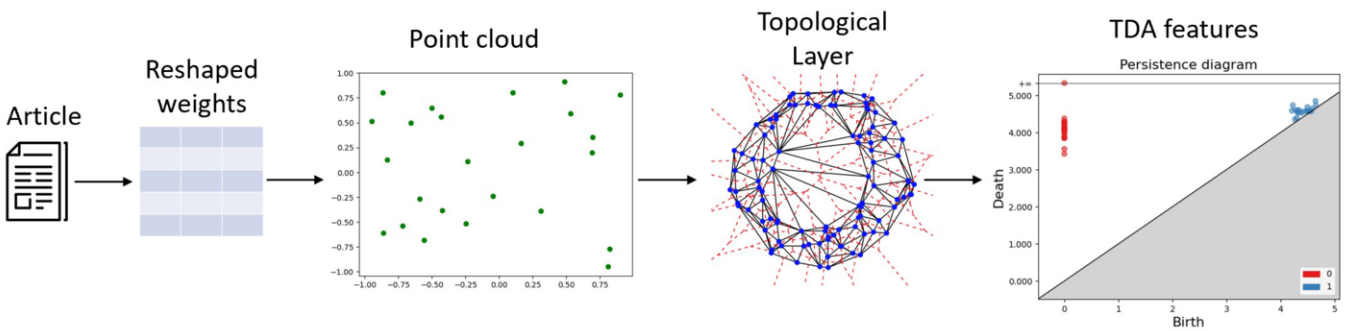


Figure 3: Illustration of how we extract the TDA features using the reshaped Transformer-based model's regularized weights as input. First, we reshape the regularized $pooled\_output$ from 1×768 dimensions to 24×32 and use this 2D matrix as input for the Topological layer. The Topological layer treats this 2D matrix as a point cloud plot and extracts TDA features ($birth$ & $death$). Next, these TDA features are plotted in a figure known as *Persistent Diagram*, where the $birth$ features are on the x-axis and $death$ features are on the y-axis. While we plot the features from the 0-Dimension (connected components) and 1-Dimension (loops), only 0-Dimension features are used for our task.

Algorithm 1 TopFormer Algorithm (Forward Pass)

Input: Document x, pretrained transformer model $M(\cdot)$, fully-connected neural network $f(\cdot)$.
Output: Predicted authorship label y'
$x_{\text{pooled}} \leftarrow$M(x) //Retrieve pooled output from M
$x_{\text{pooled}} \leftarrow$reshape($x_{\text{pooled}}$) //Reshape to a suitable form
$x_{\text{TDA}} \leftarrow$TDA($x_{\text{pooled}}$) //Extract topological features
$x^* \leftarrow$concatenate(x_{pooled}, x_{TDA})
$y' \leftarrow$softmax($f(x^*)$)
Return: y'

To train our end-to-end Topological model, we first fine-tune RoBERTa-base model. As we fine-tune the model, we take the $pooled\_output$ which is a 1×768 vector containing the latent representations of the model from the final layer. We find that the weights are richer than BERT because it is a robustly trained BERT model and has over 20K more vocabulary size than BERT. These latent representations capture word-level and sentence-level relationships, thus extracting contextual representations [22]. Due to the contextual representations captured, the weights essentially extract semantic and syntactic linguistic features [36, 31].

Next, we pass this $pooled\_output$ which is a 1×768 vector into a regularization layer, called $dropout$. This $dropout$ layer drops a predefined percentage (30% in our case) of our $pooled\_output$ to make our model more generalizable and less likely to overfit. This yields an output $dropout(pooled\_output)$ with the same dimensions as the input - 1×768 vector.

Before, we use our regularized output - $dropout(pooled\_output)$ as input for the Topological layer[3], we first reshape it from 1D $\rightarrow$ 2D. TDA requires at least a 2D matrix to construct simplicial complexes that persistent homology technique uses to extract the $birth$ and $death$ of TDA features (connected components, specifically) [26]. This is because the simplicial complexes can only be extracted from the point cloud (which can be visualized as a scatterplot of the dataset) and to get this point cloud we need a dataset with 2-coordinates. We include this Topological layer in the backbone model because: (1) RoBERTa has richer latent representations than BERT [22]; (2) TDA is robust to noise, out-of-distribution, and limited data [38]; (3) TDA is able to capture features that other fea-

[3] https://github.com/aidos-lab/pytorch-topological/tree/main

ture extraction techniques cannot capture [28, 29]; and (4) TDA extracts the true structure of data points [26]. To convert the regularized weights from 1D $\rightarrow$ 2D is non-trivial because we need to get the best shape to obtain useful TDA features which are stable and uniform (vectors of the same length) across all input of a particular dataset. Therefore, we tried different 2D sizes and found that the closer it is to a dense square matrix, the more stable the TDA features are. Stable in this context means that for every input, the TDA layer outputs the same number of features in a vector. Therefore, we convert the 1×768 vector to a 24×32 matrix, since it is the closest to a square matrix as 768 is not a perfect square. We also find through experimentation that when row $>$ column, TDA features are unstable. Unstable for our task means that the Topological layer output different vector sizes of TDA features given the input. Also, sometimes the feature vector can only contain nan values based on the input which means that it was unable to extract TDA features. Thus, we find that $pooled\_output$ must be reshaped such that row $\leq$ column and 24×32 satisfies this claim. Finally, using the 2D matrix as input to our Topological layer, we obtain the 0-Dimension (β_0) features following the process illustrated in Section 3. This yields a 23×3. These 3 columns represent the $birth$ time, $death$ time, and persistence features, respectively. Persistence is defined as the length of time it took a feature to die. Next, this 2D matrix is flattened to a vector size of 1×69 so it can be easily concatenated with the 1D $dropout(pooled\_output)$. See Figure 3 for an illustration of how the TDA features are extracted. We interpret these TDA features as capturing linguistic structure, as it is capturing the structure and shape of textual data.

Lastly, we concatenate the regularized backbone weights ($dropout(pooled\_output)$) of size 1×768 with the TDA features of size 1×69. This yields a vector of size 1×837. Thus, this 1×837 vector serves as input for the final layer of feature transformation, the Linear layer. The Linear layer's latent space increases from 768 to 837 in order to take the concatenated vector as input. TDA increases the latent space by 69 dimensions. However, we observe that unlike other TDA-based Transformer classifiers [20, 27], which use attention weights in which the size is dependent on the length of text, our TDA technique increases the latent space minimally. Finally, the output of this Linear layer is a vector that is the size of $batch\_size \times number\_of\_labels$. Thus, if we have $batch\_size = 16$ and $number\_of\_labels = 20$, we obtain a vector of size: 16×20. Finally, we pass this vector as input into the softmax layer for multi-class classification.

Finally, TDA features are compatible with non-TDA features [24], making it a suitable technique for extracting subtle linguistic patterns that distinguish deepfake texts from human-written ones. Thus, **TopFormer** captures semantic, syntactic, and structural linguistic features.

5 Experiments

5.1 Datasets

5.1.1 OpenLLMText

OpenLLMText [4] dataset was collected from five authors - *human, LLaMA, ChatGPT, PALM,* and *GPT-2*. For all the LLaMA, ChatGPT, and PALM labels, the text was generated by paraphrasing the human-written texts. GPT-2 was prompted to completely generate new texts given some of the human-written texts in the prompt. Thus, due to the different NLG methods employed, this data also has 3 writing style labels - human, paraphrasers, and generators. However, to make the dataset further reflect the real world where there is gross data imbalance (human label $\geq$ machine label), we take 1% of all the deepfake labels. And we could do this because the initial dataset had 340K samples. See Table 1 for the train, validation, and test splits.

Dataset	Train	Valid	Test	# Labels
OpenLLMText	53K	10K	7.7K	5
SynSciPass	87K	10K	10K	12
Mixset	2.4K	340	678	8

Table 1: Dataset summary statistics.

5.1.2 SynSciPass

SynSciPass [32] dataset is comprised of scientific articles, authored, by both human and deepfake authors. In addition to being grossly imbalanced (i.e., 79K examples for human & 600-850 examples for deepfake labels). This is because the deepfake texts are generated with open-ended text-generators like GPT-3, SynSciPass's deepfake labels are generated with 3 types of text-generators. These are open-ended generators, translators like Google translate (e.g. English $\rightarrow$ Spanish $\rightarrow$ English), and paraphrasers like SCIgen and Pegasus. Using these different text-generation techniques introduces high variance in writing styles in this dataset. We use the 12 labels - 1 human & 11 deepfake text-generators. However, due to the different NLG methods employed, this data also has 4 writing style labels - human, generators, translators, and paraphrasers. See Table 1 for the train, validation, and test splits.

5.1.3 Mixset

The Mixset [48] dataset has 3 labels - human, machine, and mixset. Machine labels are for the texts that were fully generated with the LLMs, not paraphrased, while mixset is for texts that were paraphrased either by humans or different LLMs using diverse paraphrasing prompts to paraphrase both the human-written texts and LLM-generated texts. We use 8 labels for the distinct authorships - *human, GPT-4, LLaMA, Dolly, ChatGLM, StableLM, ChatGPT-turbo*, and *ChatGPT*. Only LLaMA and GPT-4 were used for paraphrasing, while ChatGLM, Dolly, StableLM, ChatGPT, and ChatGPT-turbo are used for generation (i.e., generating entirely new texts). Thus, due to the different NLG methods employed, this data also has 3 writing style labels - human, paraphrasers, and generators. Finally, to keep in the context of real-world settings, due to the small size of the data, we were only able to get sufficient samples per label with as low as 10% of the non-human (deepfake) labels. See Table 1 for the train, validation, and test splits.

5.2 Authorship Attribution Models

We evaluate a diverse set of authorship attribution models:

- **GPT-who** [44]: is a psycholinguistically-aware domain-agnostic statistical-based deepfake text detector. It obtains token probabilities using GPT-2 XL pre-trained weights and calculates the uniform information density features and uses Logistic regression for classification.
- **Contra-BERT** [2]: is a BERT-base model that is trained with contrastive and cross-entropy loss functions. Ai et al. [2] shows that the model can achieve SOTA performance when applied to both traditional AA (i.e., only human authors) and deepfake text attribution.

MODEL	Precision	Recall	Accuracy	Macro F1
GPT-who	0.1924	0.2000	0.9619	0.1961
Contra-BERT	**0.8112**	0.6587	0.9761	0.6696
BERT	0.7914	0.7190	**0.9792**	0.7255
RoBERTa	0.8099	0.7428	0.9764	0.7288
Gaussian-RoBERTa	0.6231	0.5009	0.9740	0.5321
TopFormer	0.8069	**0.7727**	0.9766	**0.7522**

Table 2: OpenLLMText Authorship Attribution results. The best performance is **boldened** and the second best is underlined.

MODEL	Precision	Recall	Accuracy	Macro F1
GPT-who	0.2224	0.1434	0.9126	0.1622
Contra-BERT	0.7967	0.7307	0.9703	0.7492
BERT	0.8585	0.8148	0.9791	0.8327
RoBERTa	0.9012	0.8554	0.9853	0.8719
Gaussian-RoBERTa	0.8929	0.8809	0.9872	0.8847
TopFormer	**0.9177**	**0.8978**	**0.9892**	**0.9058**

Table 3: SynSciPass Authorship Attribution results. The best performance is **boldened** and the second best is underlined.

MODEL	Precision	Recall	Accuracy	Macro F1
GPT-who	0.2825	0.2446	0.6647	0.6647
Contra-BERT	0.7338	0.7411	0.8882	0.7287
BERT	0.7982	0.8214	0.9118	0.8034
RoBERTa	0.7697	0.7976	0.9000	0.7705
Gaussian-RoBERTa	0.4014	0.3862	0.7404	0.7404
TopFormer	**0.8181**	**0.8268**	**0.9176**	**0.8294**

Table 4: Mixset Authorship Attribution results. The best performance is **boldened** and the second best is underlined.

- **BERT:** We use BERT-base cased pre-trained model. This is also typically used as a strong baseline for deepfake text attribution (often outperforming proposed detectors) [42].
- **RoBERTa:** We use RoBERTa-base pre-trained model. This is another typical strong baseline for deepfake text attribution (often outperforming proposed detectors) [42].
- **Gaussian-RoBERTa:** RoBERTa-base model with a Gaussian layer to add Gaussian noise to the weights. The hypothesis is that if **TopFormer** achieves superior performance randomly, then adding a Gaussian layer should have a similar effect.
- **TopFormer:** RoBERTa-base with a Topological layer, following the process described in Section 4 and Figure 2.

We train all the models, except GPT-who and Contra-BERT with the same hyperparameters & parameters - MAX_length = 512, dropout probability $p = 0.3$, learning rate of 2e-5, cross-entropy loss, batch size of 16 and 5 epochs. Also, we tested our **TopFormer** model with other loss functions (contrastive loss, topological loss, and Gaussian loss) and found cross-entropy to be the best. Lastly, we evaluate these models using established evaluation metrics for machine learning - Precision, Recall, Accuracy, and Macro F1 score.

6 Results

Our proposed model - TopFormer is evaluated on its ability to more accurately attribute human- vs. deepfake-authored articles to their true authors. We specifically used imbalanced multi-style datasets - OpenLLMText, SynSciPass, and Mixset to simulate the current landscape of deepfake texts vs. human-written texts in the real world. From Tables 2, 3 and 4, we observe that TopFormer excels in the AA task, consistently outperforming the baseline SOTA deepfake AA models in all 3 datasets.

	OpenLLMText		SynSciPass		Mixset	
MODEL	Macro F1	%Δ	Macro F1	%Δ	Macro F1	%Δ
RoBERTa	0.7320	-	0.9064	-	0.9255	-
TopFormer	0.7331	0%	0.9746	7% ↑	0.9240	0%

Table 5: Deepfake Text Style Detection results. %Δ is the % Gains in Macro F1.

	OpenLLMText		SynSciPass		Mixset	
MODEL	Macro F1	%Δ	Macro F1	%Δ	Macro F1	%Δ
BERT	0.7255	-	0.8327	-	0.8034	-
Gaussian-BERT	0.6607	6%↓	0.7933	4%↓	0.5556	24%↓
TopFormer w/ BERT	0.6786	4%↓	0.8471	2% ↑	0.8480	5% ↑

Table 6: Results from using BERT as the Transformer-based backbone. %Δ is the % Gains in Macro F1.

TopFormer outperforms in all 3 datasets, achieving the highest Macro F1 scores. Additionally, the customized baseline deepfake attribution models - GPT-who and Contra-BERT underperform by a large margin. However, Gaussian-RoBERTa underperforms RoBERTa in all datasets, except for SynSciPass which minimally outperforms by 1%. Other deepfake attribution baseline models - BERT & RoBERTa underperform as well, with BERT outperforming RoBERTa in only the Mixset dataset, possibly because of the small nature of the dataset.

In addition, adding a TDA layer to RoBERTa increases the training time and inference time by about 0.5-2 hours, depending on data size. However, just inference time remains fast.

7 Further Analysis

7.1 Deepfake Text Style Detection

To further show the utility of TopFormer in the ever-evolving LLM landscape, we evaluated our model on the different writing styles of the current LLMs. Initially, users would only prompt the models with some of the text and the models would generate the rest. However, due to more recent improvements in NLG, LLM users, could create LLM-generated texts through paraphrasing, open-ended generation, summarization, translation, etc. Therefore, it is not only important to know which LLM authored a piece of text, but it could also be beneficial to figure out if it is through translation, paraphrasing, or open-ended generation. This knowledge, given the context and domain, can help ascertain the intention of the user - whether to improve their writing, plagiarize, evade detection, generate disinformation, etc.

We evaluate OpenLLMText and Mixset on 3 labels - human, paraphrasers, and generators, while SynSciPass has 4 labels - human, paraphrasers, translators, and generators. See Table 5 for results. We observe that TopFormer only outperforms when evaluated on the SynSciPass dataset, improving by 7%. This could be because the SynSciPass dataset has one more writing style than the other datasets and TDA seems to thrive on high variability. Additionally, TopFormer performs comparably to RoBERTa on the other 2 datasets - OpenLLMText and Mixset. This suggests that TDA-based techniques can also benefit deepfake text style detection, especially with high style variability.

	MODEL	METRICS			
		Pre	Rec	Acc	Mac.F1
TB	RoBERTa	**0.7555**	**0.7505**	**0.8333**	**0.7433**
	Gaussian-RoBERTa	0.6500	0.6487	0.7609	0.6452
	TopFormer	0.7499	0.7452	0.8283	0.7313
M4	RoBERTa	0.9657	0.9681	0.9852	0.9656
	Gaussian-RoBERTa	0.9644	0.9719	0.9844	0.9673
	TopFormer	**0.9745**	**0.9762**	**0.9893**	**0.9748**

Table 7: Authorship Attribution results in Precision (Pre), Recall (Rec), Accuracy (Acc), and Macro F1 (Mac.F1) on TuringBench (TB) and M4. The best and second best is **boldened** and underlined.

7.2 Alternative Transformer-based Backbone

We choose RoBERTa as our backbone because it remains the SOTA or at least a decent baseline for text classification tasks, including deepfake text attribution [42]. However, we wanted to investigate how well a different backbone - BERT will perform when a Topological layer is included in the model. We compare vanilla BERT to TopFormer w/ BERT. In addition, we compare these models to Gaussian-BERT to show TopFormer's performance is not related to being trained with noise but extracting useful features. See results in Table 6.

TopFormer w/ BERT outperforms when evaluated on the SynSciPass and Mixset datasets (by a 2% and 5% increase). While underperforming for OpenLLMText dataset by a 4% decrease. TopFormer w/ BERT performed the best on the smallest dataset - Mixset, suggesting that it may be more suitable for smaller datasets. Additionally, Gaussian-BERT consistently underperformed by large margins, suggesting that TopFormer w/ BERT's performance is not due to training with noise but is learning and extracting useful (additional) features. Finally, these results suggest that including a Topological layer in a Transformer-based model could increase the performance depending on the heterogeneity of the dataset.

7.3 Homogeneous Deepfake Text Benchmarks

We claim that TDA improves performance when evaluated on datasets that better reflect the current landscape - multi-style deepfake texts. To show where TDA does not work so well, we evaluate TopFormer on the more homogeneous datasets, where there are only 2 styles - human and one style of deepfake text generation. For a fairer comparison, we also skewed each dataset to have only 10% of deepfake labels. We use TuringBench (TB) [41] which has 20 labels (i.e., 1 human & 19 deepfake labels) and Monolingual English M4 [46] which has 6 labels (i.e, 1 human & 5 deepfake labels). Table 7 shows that TopFormer marginally underperforms and outperforms on the TB and M4 datasets by 1% for both. This is because of the homogeneity of the dataset, however, we can still observe that TopFormer performed comparably to the base model - RoBERTa, suggesting that even when TDA fails there is no significant loss of performance.

8 Ablation study

We compare different inputs to the Topological layer as follows.

1. **Pooled_output**: Our technique uses the reshaped 2D matrix of the *pooled_output* as input. We call this model, **TopFormer** *(Ours)*.
2. **Attention weights**: We use the attention weights which is of size $Max\_length \times 768$ as input for the Topological layer. This technique is inspired by Kushnareva et al. [20], Perez and Reinauer [27], who use threshold-guided directed and undirected graphs of the attention weights for binary classification. Thus, instead of increasing the computational cost by building graphs with the attention weights, we use the attention weights as input for extracting the TDA features. This technique provides a fairer comparison to TopFormer *(Ours)*. We will call the model using the attention weights - **TopFormer**_*attn*.
3. **Last_hidden_state weights**: We use the last hidden state that is a 3D matrix with size $12 \times Max\_length \times Max\_length$ as input for the TDA layer. This increases the size of the Linear latent space from 768 to 19,164. We call this model - **TopFormer**_*hidden*
4. **Correlation of Pooled_output**: This is another intuitive technique where we multiply the transposed *pooled_output* vector (with size 768×1) to the *pooled_output* (with size 1×768) to obtain a 2D matrix, creating a correlation matrix of the vector. This yields a 768×768 matrix which is input for the TDA layer, increasing the size of the Linear latent space from 768 to 3068. We call this model - **TopFormer**_*pool_corr*.

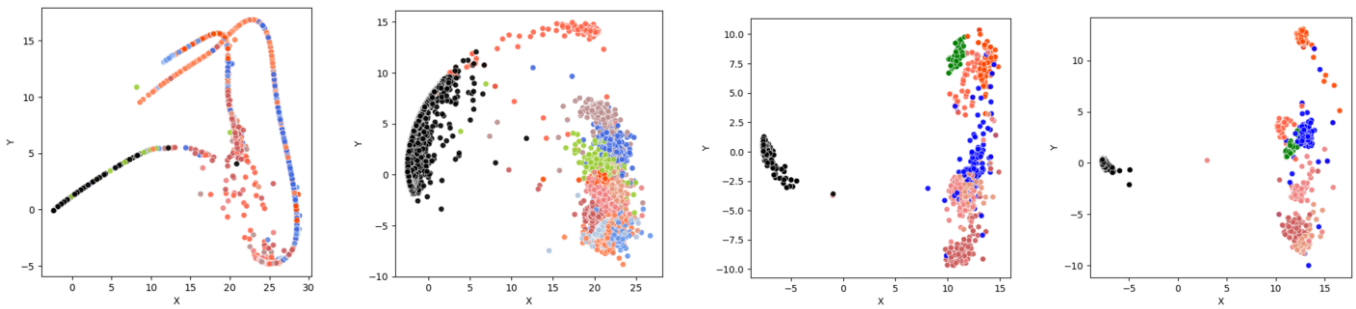
9 Discussion

9.1 Improvement with TDA is Not Random

Since RoBERTa is a black-box model, to confirm that TopFormer's performance is not due to training with the right kind of "noise," we compare with the Gaussian model - Gaussian-RoBERTa. The hypothesis is that if TDA's performance is due to noise, then the Gaussian models trained on noise should perform similarly. We observe from Tables 2, 3, and 4 that Gaussian-RoBERTa underperforms for both OpenLLMText and Mixset and marginally improves by 1% for the SynSciPass dataset. While Gaussian-BERT underperforms for all 3 datasets in Table 6. However, TopFormer w/ BERT outperforms their base models for 2 out of 3 datasets, improving by a larger margin on the Mixset dataset (5%, Table 6). In addition, we observe that the baseline SOTA deepfake AA models - GPT-who and Contra-BERT underperform in the task. Finally, TopFormer is the only model that was able to consistently outperform all other models. Thus, the large decrease in performance in the Gaussian models for the 3 datasets, suggests that our TDA-based techniques performances are not by chance.

9.2 TDA Captures Structural Features which Complement RoBERTa Weights

There are currently 5 linguistic levels - phonology, pragmatics, morphology, syntax, and semantics. Phonology has to do with spoken speech sounds and pragmatics is how language is used to convey meaning. The most relevant for written text classifications are syntax, morphology, and semantics. This means that to accurately capture the linguistic features from a piece of text, syntactic, morphology, and semantic features are needed. However, RoBERTa captures a broad range of linguistic patterns, such as contextual representations in terms of syntactic and semantic relationships. This means that on the surface level, RoBERTa is only able to capture word order and thus can infer syntactic and some semantic relationships. However, TDA features capture the true shape or structure of data even with the presence of noise. These features in the context of NLP, can be interpreted as linguistic structures. Finally, by combing these 3



(a) RoBERTa-SynSciPass. (b) TOPFORMER-SynSciPass. (c) RoBERTa-Mixset. (d) TOPFORMER-Mixset.

Figure 4: PCA plots from RoBERTa and TOPFORMER training embeddings for the SynSciPass and Mixset datasets on the classification tasks. The **black** clusters are the human labels and the other clusters are the deepfake text labels.

MODEL	OpenLLMText		SynSciPass		Mixset	
	Macro F1	%Δ	Macro F1	%Δ	Macro F1	%Δ
RoBERTa	0.7288	-	0.8719	-	0.7705	-
TOPFORMER_attn	**0.8970**	17% ↑	0.8923	2% ↑	0.6438	8% ↓
TOPFORMER_hidden	0.1961	53% ↓	0.0794	79% ↓	0.0955	72% ↓
TOPFORMER_pool_corr	0.7993	7% ↑	0.0794	79% ↓	0.2045	53% ↓
TOPFORMER (Ours)	0.7522	2% ↑	**0.9058**	4% ↑	**0.8294**	6% ↑

Table 8: Ablation study results. The best performance is **boldened** and the second best is underlined. %Δ is the % Gains in Macro F1.

linguistic features, our model - TOPFORMER can more accurately distinguish deepfake texts from human-written ones, as observed in Tables 2, 3, 4 and 5.

This can be further observed in Figures 4, where we plot the PCA features which is a linear combination of RoBERTa's word embeddings vs. TOPFORMER's word embeddings of the SynSciPass and Mixset datasets (i.e., the 2 datasets where TOPFORMER outperformed RoBERTa, the backbone by a high margin). We observe the more distinct clusters in TOPFORMER's plots for both datasets. This suggests that TDA can extract additional features for more accurate attribution of authors.

9.3 TOPFORMER *Performs Well on Style Detection*

To evaluate the robustness of TOPFORMER to data with multi-style labels, we run further experiments to compare vanilla RoBERTa to TOPFORMER. For this task, we use the style labels for each dataset. OpenLLMText and Mixset have the same 3 labels - *human, paraphrasers,* and *generators*, while SynSciPass has an additional label to the 3 labels - *translators*. See Table 5 for results. For the SynSciPass dataset, we observe that TOPFORMER significantly outperforms the baseline (7% increase, Table 5). This could be because the SynSciPass dataset has an additional writing style (translators) than the other datasets, and since TDA performs well when data is noisy and heterogeneous, this higher variability improves performance.

9.4 TOPFORMER *Performs comparably on Homogeneous Datasets*

To further evaluate where TOPFORMER performs well and underperforms, we run further experiments on homogeneous datasets - TuringBench (TB) and M4, which have only 2 distinct writing styles (i.e., human and deepfake generation). We observe from Table 7 that TOPFORMER underperforms and outperforms marginally by 1% for both datasets, always outperforming Gaussian-RoBERTa. This suggests that even when TOPFORMER does not perform optimally, it still performs comparably to RoBERTa, suggesting no huge loss in performance.

9.5 *Larger is Not Always Better: Reshaped pooled_output is a Better TDA Input*

Most researchers that apply TDA for NLP tasks commonly use word2vec embeddings or attention weights as input to extract TDA features [27, 20, 47, 12, 9, 33]. We observe that while using the attention weights, the last hidden state, and the correlation matrix of the *pooled_output* as input is more intuitive as they are already in the right format for the TDA layer (> 1D vector), the reshaped 2D matrix of the *pooled_output* contains richer features for the TDA layer. See Table 8 for the ablation results of different inputs for the TDA layer. In confirmation with Kushnareva et al. [20], we discover that using all the inputs except for the reshaped *pooled_output* to extract TDA features is unstable. This could be why previous studies first constructed directed and undirected graphs with the attention weights before TDA feature extraction to encourage stability [27, 20]. Nevertheless, these techniques significantly increase computational costs.

The attention weights are large matrices, such as $Max\_length \times 768$, yielding a 1×1533 TDA output, which increases the latent space of the Linear layer from 768 to 2301. While the last hidden state and the correlation matrix increase the latent spaces to 19164 and 3068, respectively. Additionally, to maintain consistent dimensions for TDA features, we employ normalization techniques when TDA features for some articles differ from the majority. In contrast, our technique increases the dimensions of the linear layer from 768 to 837 for all datasets without requiring normalization. Finally, the results of the ablation study in Table 8 suggests that our reshaped *pooled_output* technique is superior as TOPFORMER consistently outperformed in all 3 datasets. Even though, we observe a surprising increase in performance for TOPFORMER_attn and TOPFORMER_pool_corr evaluated on the OpenLLMText dataset. This still confirms the utility of TDA, while suggesting that TOPFORMER (OURS) is the most consistent technique and computationally less expensive than other techniques.

10 Conclusion and Future Work

We propose a novel solution to accurately attribute the authorship of deepfake vs. human texts - **TOPFORMER**. This technique entails including a Topological layer in a Transformer-based model, such that

the Linear layer's input is a concatenation of the backbone's regularized weights and the TDA features. Next, we evaluate our model on realistic datasets that reflect the landscape of how people use LLMs, either to generate, paraphrase, summarize, or edit a piece of texts. Our novel technique, TOPFORMER outperforms all SOTA baseline models when evaluated on these 3 realistic datasets.

Lastly, in the future, we will scrutinize our models under stricter constraints such as evaluation on adversarial robustness, known as *Authorship Obfuscation*, open-set datasets, and out-of-distribution datasets, such as low-resource languages, multilingual, and insufficiently sized datasets.

11 Ethical Statement

Since the advent of LLMs, society has had to grapple with the many benefits and huge potential malicious uses of such technology. To mitigate these risks, some being obvious security risks like the creation of authentic-looking misinformation, researchers have been working on several niche areas in LLM, including *deepfake text detection*. However, due to the opposing benefits of LLMs (i.e., great good and great evil), we understand that the proposed mitigation of risks that LLMs pose can also be turned around and used for evil. Thus, we understand that detection models like TOPFORMER which can more accurately detect deepfake texts, especially in stricter conditions (i.e., imbalanced sample, multi-style labels, noisy data), can also be used to test when deepfake texts can evade detection after obfuscation. However, due to the serious security risks that LLMs pose, we believe that the benefits of accurate authorship attribution of deepfake text authors outweigh the potential risks.

Acknowledgement

This work was in part supported by NSF awards #1820609, #2114824, and #2131144. Adaku thanks Dr. Bastian Rieck for the insightful discussions on how to improve our TDA technique. Also, Adaku would like to thank Dr. Charlie Dagli for reading the paper drafts and providing invaluable recommendations.

References

[1] M. Abassy, K. Elozeiri, A. Aziz, M. N. Ta, R. V. Tomar, B. Adhikari, S. E. D. Ahmed, Y. Wang, O. M. Afzal, Z. Xie, et al. Llm-detectaive: a tool for fine-grained machine-generated text detection. *arXiv preprint arXiv:2408.04284*, 2024.

[2] B. Ai, Y. Wang, Y. Tan, and S. Tan. Whodunit? learning to contrast for authorship attribution. In *AACL*, pages 1142–1157, 2022.

[3] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. B. Brown, D. Song, U. Erlingsson, et al. Extracting training data from large language models. In *USENIX Security Symposium*, volume 6, 2021.

[4] Y. Chen, H. Kang, V. Zhai, L. Li, R. Singh, and B. Raj. Token prediction as implicit classification to identify llm-generated text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13112–13120, 2023.

[5] Z. Chen and H. Liu. Stadee: Statistics-based deep detection of machine generated text. In *International Conference on Intelligent Computing*, pages 732–743. Springer, 2023.

[6] D. Cherniavskii, E. Tulchinskii, V. Mikhailov, I. Proskurina, L. Kushnareva, E. Artemova, S. Barannikov, I. Piontkovskaya, D. Piontkovski, and E. Burnaev. Acceptability judgements via examining the topology of attention maps. In *Proceedings of the 2022 EMNLP*, volume 2022, pages 88–107, 2022.

[7] A. Deshpande, V. Murahari, T. Rajpurohit, A. Kalyan, and K. R. Narasimhan. Toxicity in chatgpt: Analyzing persona-assigned language models. In *Proceedings of the 2023 EMNLP*, 2023.

[8] 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.

[9] P. Doshi and W. Zadrozny. Movie genre detection using topological data analysis. In *Statistical Language and Speech Processing: 6th International Conference, SLSP 2018, Mons, Belgium, October 15–16, 2018, Proceedings 6*, pages 117–128. Springer, 2018.

[10] S. Gehrmann, S. Harvard, H. Strobelt, and A. M. Rush. Gltr: Statistical detection and visualization of generated text. *ACL 2019*, page 111, 2019.

[11] S. Gholizadeh, A. Seyeditabari, and W. Zadrozny. Topological signature of 19th century novelists: Persistent homology in text mining. *big data and cognitive computing*, 2(4):33, 2018.

[12] P. Haghighatkhah, A. Fokkens, P. Sommerauer, B. Speckmann, and K. Verbeek. Story trees: Representing documents using topological persistence. In *Proceedings of the Thirteenth LREC 2022*, pages 2413–2429, 2022.

[13] F. Hensel, M. Moor, and B. Rieck. A survey of topological machine learning methods. *Frontiers in Artificial Intelligence*, 4:681108, 2021.

[14] 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.

[15] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein. A watermark for large language models. In *International Conference on Machine Learning*, pages 17061–17084. PMLR, 2023.

[16] R. Koike, M. Kaneko, and N. Okazaki. Outfox: Llm-generated essay detection through in-context learning with adversarially generated examples. *arXiv preprint arXiv:2307.11729*, 2023.

[17] T. Kumarage, A. Bhattacharjee, D. Padejski, K. Roschke, D. Gillmor, S. Ruston, H. Liu, and J. Garland. J-guard: Journalism guided adversarially robust detection of ai-generated news. In *IJCNLP-AACL*, pages 484–497, 2023.

[18] T. Kumarage, J. Garland, A. Bhattacharjee, K. Trapeznikov, S. Ruston, and H. Liu. Stylometric detection of ai-generated text in twitter timelines. *arXiv preprint arXiv:2303.03697*, 2023.

[19] T. Kumarage, P. Sheth, R. Moraffah, J. Garland, and H. Liu. How reliable are ai-generated-text detectors? an assessment framework using evasive soft prompts. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1337–1349, 2023.

[20] 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 EMNLP*, pages 635–649, 2021.

[21] J. Lee, T. Le, J. Chen, and D. Lee. Do language models plagiarize? In *Proceedings of the ACM Web Conference 2023*, pages 3637–3647, 2023.

[22] 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.

[23] J. Lucas, A. Uchendu, M. Yamashita, J. Lee, S. Rohatgi, and D. Lee. Fighting fire with fire: The dual role of llms in crafting and detecting elusive disinformation. In *Proceedings of the 2023 EMNLP*, pages 14279–14305, 2023.

[24] S. McGuire, S. Jackson, T. Emerson, and H. Kvinge. Do neural networks trained with topological features learn different internal representations? In *NeurIPS Workshop on Symmetry and Geometry in Neural Representations*, pages 122–136. PMLR, 2023.

[25] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, and C. Finn. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning*, pages 24950–24962. PMLR, 2023.

[26] E. Munch. A user's guide to topological data analysis. *Journal of Learning Analytics*, 4(2):47–61, 2017.

[27] I. Perez and R. Reinauer. The topological bert: Transforming attention into topology for natural language processing. *arXiv preprint arXiv:2206.15195*, 2022.

[28] A. Port, I. Gheorghita, D. Guth, J. M. Clark, C. Liang, S. Dasu, and M. Marcolli. Persistent topology of syntax. *Mathematics in Computer Science*, 12(1):33–50, 2018.

[29] A. Port, T. Karidi, and M. Marcolli. Topological analysis of syntactic structures. *arXiv preprint arXiv:1903.05181*, 2019.

[30] V. Rawte, A. Sheth, and A. Das. A survey of hallucination in large foundation models. *arXiv preprint arXiv:2309.05922*, 2023.

[31] E. Reif, A. Yuan, M. Wattenberg, F. B. Viegas, A. Coenen, A. Pearce, and B. Kim. Visualizing and measuring the geometry of bert. *Advances in Neural Information Processing Systems*, 32, 2019.

[32] D. Rosati. Synscipass: detecting appropriate uses of scientific text gen-

eration. In *Proceedings of the Third Workshop on Scholarly Document Processing*, pages 214–222, 2022.

[33] K. Savle, W. Zadrozny, and M. Lee. Topological data analysis for discourse semantics? In *Proceedings of the 13th International Conference on Computational Semantics-Student Papers*, pages 34–43, 2019.

[34] E. Sheng, K.-W. Chang, P. Natarajan, and N. Peng. Societal biases in language generation: Progress and challenges. 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 4275–4293, 2021.

[35] R. A. R. Soto, K. Koch, A. Khan, B. Y. Chen, M. Bishop, and N. Andrews. Few-shot detection of machine-generated text using style representations. In *The Twelfth International Conference on Learning Representations*, 2024.

[36] I. Tenney, D. Das, and E. Pavlick. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, 2019.

[37] E. Tulchinskii, K. Kuznetsov, L. Kushnareva, D. Cherniavskii, S. Barannikov, I. Piontkovskaya, S. Nikolenko, and E. Burnaev. Topological data analysis for speech processing. *arXiv preprint arXiv:2211.17223*, 2022.

[38] R. Turkes, G. F. Montufar, and N. Otter. On the effectiveness of persistent homology. *Advances in Neural Information Processing Systems*, 35:35432–35448, 2022.

[39] S. Tymochko, Z. New, L. Bynum, E. Purvine, T. Doster, J. Chaput, and T. Emerson. Argumentative topology: Finding loop (holes) in logic. *arXiv preprint arXiv:2011.08952*, 2020.

[40] 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.

[41] A. Uchendu, Z. Ma, T. Le, R. Zhang, and D. Lee. Turingbench: 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.

[42] A. Uchendu, T. Le, and D. Lee. Attribution and obfuscation of neural text authorship: A data mining perspective. *SIGKDD Explorations*, page vol. 25, 2023.

[43] A. Uchendu, J. Lee, H. Shen, T. Le, T.-H. K. Huang, and D. Lee. Does Human Collaboration Enhance the Accuracy of Identifying LLM-Generated Deepfake Texts? In *11th AAAI Conf. on Human Computation and Crowdsourcing (HCOMP)*, Nov. 2023.

[44] S. Venkatraman, A. Uchendu, and D. Lee. Gpt-who: An information density-based machine-generated text detector. *arXiv preprint arXiv:2310.06202*, 2023.

[45] R. Vukovic, M. Heck, B. Ruppik, C. van Niekerk, M. Zibrowius, and M. Gasic. Dialogue term extraction using transfer learning and topological data analysis. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 564–581, 2022.

[46] Y. Wang, J. Mansurov, P. Ivanov, J. Su, A. Shelmanov, A. Tsvigun, C. Whitehouse, O. M. Afzal, T. Mahmoud, T. Sasaki, et al. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1369–1407, 2024.

[47] X. Wu, X. Niu, and R. Rahman. Topological analysis of contradictions in text. In *Proceedings of the 45th International ACM SIGIR*, pages 2478–2483, 2022.

[48] Q. Zhang, C. Gao, D. Chen, Y. Huang, Y. Huang, Z. Sun, S. Zhang, W. Li, Z. Fu, Y. Wan, et al. Llm-as-a-coauthor: Can mixed human-written and machine-generated text be detected? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 409–436, 2024.

[49] X. Zhu. Persistent homology: An introduction and a new text representation for natural language processing. In *IJCAI*, pages 1953–1959, 2013.