Artificial Text Detection via Examining the Topology of Attention Maps

Laida Kushnareva, Daniil Cherniavskii, Vladislav Mikhailov, Ekaterina Artemova, Serguei Barannikov, Alexander Bernstein, Irina Piontkovskaya, Dmitri Piontkovski, Evgeny Burnaev

To cite this version:


HAL Id: hal-03456191
https://hal.science/hal-03456191
Submitted on 21 Dec 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Artificial Text Detection via Examining the Topology of Attention Maps

Laida Kushnareva¹, Daniil Cherniavskii², Vladislav Mikhailov¹, Ekaterina Artemova¹, Serguei Barannikov²,⁵, Alexander Bernstein², Irina Piontkovskaya¹, Dmitri Piontkovski⁴, and Evgeny Burnaev²

¹Huawei Noah’s Ark lab, Moscow, Russia
²Skolkovo Institute of Science and Technology, Moscow, Russia
³SberDevices, Sberbank, Moscow, Russia
⁴HSE University, Moscow, Russia
⁵CNRS, IMJ, Paris, France

Abstract

The impressive capabilities of recent generative models to create texts that are challenging to distinguish from the human-written ones can be misused for generating fake news, product reviews, and even abusive content. Despite the prominent performance of existing methods for artificial text detection, they still lack interpretability and robustness towards unseen models. To this end, we propose three novel types of interpretable topological features for this task based on Topological Data Analysis (TDA) which is currently understudied in the field of NLP. We empirically show that the features derived from the BERT model outperform count- and neural-based baselines up to 10% on three common datasets, and tend to be the most robust towards unseen GPT-style generation models as opposed to existing methods. The probing analysis of the features reveals their sensitivity to the surface and syntactic properties. The results demonstrate that TDA is a promising line with respect to NLP tasks, specifically the ones that incorporate surface and structural information.

1 Introduction

Recent text generation models (TGMs) based on the transformer architecture (Vaswani et al., 2017) have demonstrated impressive capabilities of creating texts which are very close to human in terms of fluency, coherence, grammatical and factual correctness (Keskar et al., 2019; Zellers et al., 2020; Yang et al., 2019). Extensive GPT-style TGMs (Radford et al., 2018) have achieved outstanding results over a great scope of NLP tasks employing zero-shot, one-shot, and few-shot techniques, even outperforming state-of-the-art fine-tuning approaches (Brown et al., 2020). However, such models can be misused for generating fake news (Zellers et al., 2020; Uchendu et al., 2020), product reviews (Adelani et al., 2020), and even extremist and abusive content (McGuffie and Newhouse, 2020).

Many attempts have been made to develop artificial text detectors (Jawahar et al., 2020), ranging from classical ML methods over count-based features (Uchendu et al., 2019) to advanced transformer-based models (Adelani et al., 2020) and unsupervised approaches (Solaiman et al., 2019). Despite the prominent performance of these methods across various domains, they still lack interpretability and robustness towards unseen models.

This paper introduces a novel method for artificial text detection based on Topological Data Analysis (TDA) which has been understudied in the field of NLP. The motivation behind this approach relies on the fact that (i) the attention maps generated by the transformer model can be represented as weighted bipartite graphs and thus can be efficiently investigated with TDA, (ii) TDA methods are known to capture well surface and structural patterns in data which, we believe, are crucial to the task.

The contributions are summarized as follows. (i) To the best of our knowledge, this work is the first attempt to apply TDA methods over the transformer model’s attention maps and interpret topological features for the NLP field. (ii) We propose three types of interpretable topological features derived from the attention graphs for the task of artificial text detection. We empirically show that a simple linear classifier trained on the TDA features produced over BERT attentions (Devlin et al., 2019) outperforms count- and neural-based baselines up to 10%, and can perform on par with the fully fine-tuned BERT model across three domains: social media, news articles and product reviews. (iii) Testing the robustness towards unseen TGMs, we find that the TDA-based classifiers tend to be more robust as opposed to the existing detectors. (iv) The probing analysis of the features demonstrates their sensitivity to surface and syntactic properties. (v)
Finally, we are publicly releasing the code\footnote{https://github.com/danchern97/tda4atd}, hoping to facilitate the applicability of the TDA methods to other NLP tasks, specifically the ones that incorporate structural information.

2 Related Work

Applications of Topological Data Analysis

TDA has been applied in NLP to study textual structural properties, independent of their surface and semantic peculiarities. These applications include detection of children and adolescent writing (Zhu, 2013), discourse and entailment in law documents (Savle et al., 2019), and exploring discourse properties of the plot summary to identify the movie genre (Doshi and Zadrozny, 2018). Guan et al. (2016) apply the topologically motivated transformation of the document’s semantic graph to summarize it further. However, these studies neither incorporate neural data representations nor explore the properties of neural language models.

The research in the emerging scope of TDA applications to neural networks and neural data representations has mainly focused on artificial datasets or common problems in computer vision. The desired topological properties of the data representation can be incorporated into the objective function during the training of a neural network, improving its robustness and performance on the downstream tasks such as human action recognition and image classification (Som et al., 2020), image simplification (Solomon et al., 2021), image segmentation (Clough et al., 2020) or generation (Gabrielson et al., 2020). Another line aims to develop the topological criteria of the network’s generalization properties (Rieck et al., 2019; Corneanu et al., 2020; Naitzat et al., 2020; Barannikov et al., 2020) or its robustness to adversarial attacks (Corneanu et al., 2019).

Exploring Attention Maps

Several studies have shown that attention maps of pre-trained language models (LMs) capture linguistic information. For the sake of space, we will discuss only a few well-known recent works. Clark et al. (2019) attempt to categorize the types of attention patterns observed in the BERT model. In particular, they discover certain attention heads in which prepositions attend to their objects or coreferent mentions attend to their antecedents. Further, they explore the typical behavior of the attention heads and introduce five patterns, e.g. attending to the next token or previous token, which the vast majority of the attention heads follow. Htut et al. (2019) explore the syntactic information encoded in intra-word relation in the attention maps. A maximum spanning tree (MST) is constructed from the computed attention weights and mapped to the corresponding dependency tree for a given sentence. This method achieves a prominent Undirected Unlabeled Attachment Score (UUAS), indicating that the attention graphs indeed can capture the dependency-based relations. Michel et al. explore the importance of the attention heads with respect to a downstream task. They show that a large proportion of the attention heads can be pruned without harming the model downstream performance. Beneficially, the pruned model speeds up at the inference time. Finally, visualization of the attention maps (Hoover et al., 2020) allows introspecting the model’s inner workings interactively.

Supervised Artificial Text Detectors

Several well-established classical ML methods have been applied to the task of artificial text detection combined with topic modeling and linguistic features (Manjavacas et al., 2017; Uchendu et al., 2019, 2020). The rise of pre-trained LMs has stimulated various improvements of the detectors. The RoBERTa model (Liu et al., 2019) has demonstrated an outstanding performance with respect to many TGMs and domains (Adelani et al., 2020; Fagni et al., 2021). The capabilities of generative models such as GROVER (Zellers et al., 2020) and GPT-2 (Radford et al., 2019) have been also evaluated on the task (Bahri et al., 2021). Last but not least, Bakhtin et al. (2019) discriminate artificial texts by training a ranking energy-based model over the outputs of a pre-trained LM.

Unsupervised Artificial Text Detectors

Another line of methods incorporates probability-based measures combined with a set of pre-defined thresholds (Solaiman et al., 2019). Such methods open up a possibility of the human in the loop approach where a human makes decisions with the help of pre-trained LMs (Ippolito et al., 2020). The GLTR tool (Gehrmann et al., 2019) supports human-model interaction by visualizing the properties of a text inferred by the model, which improves the human detection rate of artificial texts. A promising direction is involving acceptability and pseudo-perplexity metrics (Lau et al., 2020;
Salazar et al., 2020) that can be used to evaluate text plausibility.

3 Background

3.1 BERT Model

BERT is a transformer-based LM that has pushed state-of-the-art results in many NLP tasks. The BERT architecture comprises L encoder layers with H attention heads in each layer. The input of each attention head is a matrix X consisting of the d-dimensional representations (row-wise) of m tokens, so that X is of shape \( m \times d \). The head outputs an updated representation matrix \( X^{\text{out}} \):

\[
X^{\text{out}} = W^{\text{attn}}(XW^V)
\]

with \( W^{\text{attn}} = \text{softmax} \left( \frac{(XW^Q)(XW^K)^T}{\sqrt{d}} \right) \),

where \( W^Q, W^K, W^V \) are trained projection matrices of shape \( d \times d \) and \( W^{\text{attn}} \) is of shape \( m \times m \) matrix of attention weights. Each element \( w_{ij}^{\text{attn}} \) can be interpreted as a weight of the \( j \)-th input’s relation to the \( i \)-th output: larger weights mean stronger connection between the two tokens.

3.2 Attention Map and Attention Graph

An attention map displays an attention matrix \( W^{\text{attn}} \) (Equation 1) in form of a heat map, where the color of the cell \( (i, j) \) represents the relation of the \( i \)-th token to the output representation of the \( j \)-th token. We use a graph representation of the attention matrix. The attention matrix is considered to be a weighted graph with the vertices representing tokens and the edges connecting pairs of tokens with strong enough mutual relation (the higher the weight, the stronger the relation). The construction of such graph appears to be quite problematic: a threshold needs to be set to distinguish between weak and strong relations. This leads to instability of the graph’s structure: changing the threshold affects the graph properties such as the number of edges, connected components, cycles. The choice of the optimal thresholds is essential to define which edges remain in the graph. TDA methods allow extracting the overall graph’s properties which describe the development of the graph with respect to changes in the threshold.

3.3 Topological Data Analysis

TDA instruments permit tracking the changes of a topological structure across varying thresholds for different objects: scalar functions, point clouds, and weighted graph (Chazal and Michel, 2017). Given a set of tokens \( V \) and an attention matrix of pair-wise weights \( W \), we build a family of graphs termed as filtration: an ordered set of graphs for the sequence of increasing thresholds. Figure 1 depicts the filtration for a toy example. First, we build a graph for a small threshold, using which we filter out the edges with the weights lower than this threshold. Next, we increase the threshold and construct the next graph. Then we compute the core topological features of different dimensions: for \( d = 0 \) these are connected components, for \( d = 1 \) – “loops” (loosely speaking, they correspond to basic cycles in a graph), and \( d \)-dimensional “holes” for higher dimensions. The amounts of these features at each dimension \( \beta_0, \beta_1, ..., \beta_d \) are referred to as Betti numbers and serve as the main invariants of the objects in topology (see Appendix B for formal definitions). While the threshold is increasing and the edges are being filtered, new features may arise. For example, the graph can decay into several connected components. At the same time, the features can also disappear when a cycle is broken. For each feature, we check the moment in the filtration when it appears (i.e., its “birth”) and when it disappears (i.e., its “death”). These moments are depicted on a diagram called barcode (see Figure 1). The barcode’s horizontal axis corresponds to the sequence of thresholds. Each horizontal line (“bar”) corresponds to a single feature (“hole”): the line lasts from the feature’s “birth” to the feature’s “death”. Barcodes characterize the “persistent” topological properties of the graph, showing how stable topological features are.

We now detail building the attention graphs, the filtration procedure, and the proposed features which are derived from the attention graphs.

4 Persistent Features of the Attention Graphs

Informal Definition and Interpretation We extract three groups of features from the attention graphs. Topological features (Section 4.1) include a set of standard graph properties: the number of connected components, the number of edges, and the number of cycles. These features are calculated for each pre-defined threshold separately and
Figure 1: Let us consider an attention map computed on the sentence “I love you” with the BERT model (Layer: 1, Attention Head: 6) which is depicted in (a). After matching the vertices of the corresponding graph (b) and removing directions as shown in (c) and (d), we get graph (e) with 5 vertices and 6 edges. For the sake of better visualization, we do not draw edges with a weight less than 0.2. The graph has one connected component ($\beta_0 = 1$) and two “loops” ($\beta_1 = 2$). After filtering out edges with small weight, we get graph (f) which has one new connected component (it is often referred to as “birth” of a new component) and does not have any “loops” (i.e., the loops that we can see in the previous version of the graph have “died”). Consequently, after removing all edges, we get graph (g) where 3 new connected components are born, and now there are 5 connected components ($\beta_0 = 5$) in total. The barcode (h) depicts 0-dimensional features (connected components) for the filtration ((e), (f) and (g)). Here, the X-axis denotes the filtration parameter $\epsilon$, and the Y-axis denotes the number of the bars. We ignore the “infinite” feature persisting through the whole filtration. Note that conventionally on barcodes the $x$ axis is inverted.
then concatenated. We consider two following variants of the feature calculation: for a directed and an undirected attention graph. **Barcode features** (Section 4.2) are extracted from barcodes. **Distance to patterns** (Section 4.3) is the group of features derived from the attention maps by computing the distance to the attention patterns (Clark et al., 2019).

To give the linguistic interpretation of our features, recall that the graph structures are used in lexicology for describing semantic change laws (Hamilton et al., 2016; Lipka, 1990; Arnold, 1973). The evolution of the meaning of a word with time can be represented as a graph, in which edges represent a semantic shift to different word meanings. Two typical patterns are distinguished in the graph structure: *radiation* – the “star” structure, where the primary meaning is connected to other connotations independently; *concatenation*, or *chaining shift* – the “chain” structure when the connotations are integrated one-by-one. Note that the typical attention patterns (Clark et al., 2019) have the same “radiation” and “concatenation” structure. In pre-trained LMs, the evolution goes through the layers of the model, changing the representation of each token, ending up with highly contextualized token representations, and the aggregated representation of the whole sentence (in the form of the [CLS] token).

We consider persistent features as the numerical characteristic of the semantic evolution processes in the attention heads. Topological features deal with clusters of mutual influence of the tokens in the sentence and the local structures like chains and cycles. The barcode features characterize the severity and robustness of the semantic changes. The features with long persistence (large distance between “birth” and “death”) correspond to the stable processes which dominate the others, while short segments in the barcode define processes highly influenced by noise. Pattern features provide a straightforward measure of the presence of typical processes over the whole sentence. The so-called “vertical” pattern corresponds to the “radiation” around the single token when the meaning of the sentence or a part of the sentence is aggregated from all words equally. “Diagonal” pattern represents consequent “concatenation” structure, going through all the sentence and thus reflecting the dependence of each token’s meaning on its left context.

### 4.1 Topological Features
First, we fix a set of thresholds $T = \{t_i\}_{i=1}^k, 0 < t_1 < \ldots < t_k < 1$. Consider an attention head $h$ and corresponding weights $W_{attn} = (w_i^h)$. Given a text sample $s$, for each threshold level $t \in T$ we define the weighted directed graph $\Gamma_s^h(t)$ with edges $\{j \rightarrow i \mid w_i^h \geq t\}$ and its undirected variant $\Gamma_s^h(t)$ by setting an undirected edge $v_i v_j$ for each pair of vertices $v_i$ and $v_j$ which are connected by an edge in at least one direction in the graph $\Gamma_s^h(t)$.

We consider the following features of the graphs:

- the first two Betti numbers of the undirected graph $\Gamma_s^h(t)$. The feature calculation procedure is described in Appendix A;
- the number of edges $(e)$, the number of strongly connected components $(s)$ and the amount of simple directed cycles $(c)$ in the directed graph $\Gamma_s^h(t)$.

To get the whole set of topological features for the given text sample $s$ and the attention head $h$, we concatenate the features for all the thresholds, starting from $T$.

### 4.2 Features Derived from Barcodes
For each text sample we calculate barcodes of the first two persistent homology groups (denoted as $H_0$ and $H_1$) on each attention head of the BERT model (see Appendix B for further details). We compute the following characteristics of these barcodes:

- The sum of lengths of bars;
- The mean of lengths of bars;
- The variance of lengths of bars;
- The number of bars with time of birth/death greater/lower than threshold;
- The time of birth/death of the longest bar (excluding infinite);
- The overall number of bars;
- The entropy of the barcode.

### 4.3 Features Based on Distance to Patterns
The shape of attention graphs in distinct attention heads can be divided into several patterns (Clark et al., 2019). We hypothesize that appearance of such patterns in a particular head or “intensity” of the pattern (i.e., the threshold $t$ on which the pattern appears) may carry essential linguistic information. Thus, we formalize these attention patterns and calculate the distances to them as follows.

Let $A = (a_{ij})$ be an incidence matrix of the
Table 1: Statistics for the datasets used in the experiments on the artificial text detection task. H=Human; M=Machine.

| Text Source       | Train | Validation | Test | |Vocab| | Length |
|-------------------|-------|------------|------|-------|-------|-------|
|                   | H     | M          | H    | M     | H    | M     | H     | M     |
| WebText GPT-2 Small; pure sampling | 20K | 20K | 2.5K | 2.5K | 2.5K | 2.5K | 220K | 532K |
| Amazon Review GPT-2 XL; pure sampling | 5K | 5K | 1K | 1K | 4K | 4K | 47K | 49K |
| RealNews GROVER top-p sampling | 5K | 5K | 1K | 1K | 4K | 4K | 98K | 75K |

Graph $\Gamma$ with $n$ vertices, where $a_{ij} = 1$ for all edges $(ij) \in E$ and 0 for all other $i, j$. Let $\Gamma = (V, E)$ and $\Gamma' = (V, E')$ be two graphs with the same set of vertices, and let $A$, $A'$ be their incidence matrices. As a distance $d$ between such graphs we use Frobenius norm of the difference $||A - A'||_F = \sqrt{\sum_{i,j} (a_{ij} - a'_{ij})^2}$, normalized by the norms of the matrices of compared graphs:

$$d(\Gamma, \Gamma') = \frac{||A - A'||_F}{\sqrt{||A||_F^2 + ||A'||_F^2}} = \sqrt{\frac{\sum_{i,j} (a_{ij} - a'_{ij})^2}{\sum_{i,j} (a_{ij}^2 + a'_{ij}^2)}}.$$

Such distance takes values between 0 and 1. For the unweighted graphs we have:

$$d(\Gamma, \Gamma') = \sqrt{\frac{|E \Delta E'|}{|E| + |E'|}},$$

where $E \Delta E' = (E \setminus E') \cup (E' \setminus E)$ is the symmetric difference of sets $E$ and $E'$.

We consider distances from the given graph $\Gamma$ to attention patterns $\Gamma_i$ as the graph features $d_i(\Gamma) = d(\Gamma, \Gamma_i)$, and the patterns posed by (Clark et al., 2019):

- Attention to the previous token. $\Gamma_{\text{feature}} : E = (i + 1, i), i = 1, n - 1$.
- Attention to the next token. $\Gamma_{\text{feature}} : E = (i, i + 1), i = 1, n - 1$.
- Attention to [CLS]-token. [CLS]-token corresponds to the vertex 1 of the set $V = [1, n]$ as it denotes the beginning of the text. $\Gamma_{\text{feature}} : E = (i, 1), i = 1, n$.
- Attention to [SEP]-token. Suppose $i_1, \ldots, i_k$ are the indices of [SEP]-tokens. Then $\Gamma_{\text{feature}} : E = (i, i_t), i = 1, n, t = 1, k$.

- Attention to punctuation marks. Let $i_1, \ldots, i_k$ be the indices of the tokens which correspond to commas and periods. $\Gamma_{\text{feature}} : E = (i, i_t), i = 1, n, t = 1, k$. Note that this pattern can be potentially divided into Attention to commas and Attention to periods.

5 Experiments

5.1 Artificial Text Detection

Data We prepare three datasets from different domains to conduct the experiments on the task of artificial text detection. Table 1 outlines statistics for the datasets. Each split is balanced by the number of samples\(^2\) per each target class.

WebText & GPT-2 comprises a subset of natural and generated texts from the datasets proposed by Radford et al. (2018). (i) WebText contains filtered and de-duplicated natural texts from Reddit; (ii) GPT-2 Output Dataset\(^3\) includes texts generated by various versions of the GPT-2 model fine-tuned on WebText. We use texts generated by GPT-2 Small (117M) with pure sampling.

Amazon Reviews & GPT-2 consists of a subset of Amazon product reviews (Amazon, 2019) and texts generated by GPT-2 XL (1542M) with pure sampling, fine-tuned on this dataset (Solaiman et al., 2019).

RealNews & GROVER (Zellers et al., 2020) includes a subset of the news articles from RealNews (that are not present in the GROVER training data) and news articles generated by GROVER with top-p sampling.

\(^2\)Each sample is truncated to 128 BertTokenizer tokens (bert-base-uncased).

\(^3\)https://github.com/openai/gpt-2-output-dataset
Table 2: The results of the artificial text detection experiments. The performance is measured by the accuracy score (%).

<table>
<thead>
<tr>
<th>Model</th>
<th>WebText &amp; GPT-2 Small</th>
<th>Amazon Reviews &amp; GPT-2 XL</th>
<th>RealNews &amp; GROVER</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-IDF, N-grams</td>
<td>68.1</td>
<td>54.2</td>
<td>56.9</td>
</tr>
<tr>
<td>BERT [CLS trained]</td>
<td>77.4</td>
<td>54.4</td>
<td>53.8</td>
</tr>
<tr>
<td>BERT [Fully trained]</td>
<td>88.7</td>
<td>60.1</td>
<td>62.9</td>
</tr>
<tr>
<td>BERT [SLOR]</td>
<td>78.8</td>
<td>59.3</td>
<td>53.0</td>
</tr>
<tr>
<td>Topological features</td>
<td>86.9</td>
<td>59.6</td>
<td>63.0</td>
</tr>
<tr>
<td>Barcode features</td>
<td>84.2</td>
<td>60.3</td>
<td>61.5</td>
</tr>
<tr>
<td>Distance to patterns</td>
<td>85.4</td>
<td>61.0</td>
<td>62.3</td>
</tr>
<tr>
<td>All features</td>
<td>87.7</td>
<td>61.1</td>
<td>63.6</td>
</tr>
</tbody>
</table>

Baselines We use bert-base-uncased⁴ model from the HuggingFace library (Wolf et al., 2020) for the BERT-based baselines described below. (i) BERT [CLS trained] is a linear layer trained over [CLS]-pooled text representations. Note that the weights of the BERT model remain frozen. (ii) BERT [Fully trained] is a fully fine-tuned BERT model. We also train Logistic Regression classifier from scikit-learn library (Pedregosa et al., 2011) over (iii) TF-IDF, N-grams with the N-gram range ∈ [1, 2] and (iv) BERT [SLOR] (Pauls and Klein, 2012), an pseudo-perplexity-based acceptability measure inferred with the BERT model under the implementation by Lau et al. (2020)⁵.

Models We train Logistic Regression classifier over the persistent graph features derived from the attention matrices from the BERT model: (i) Topological features (Section 4.1), (ii) Barcode features (Section 4.2) and (iii) Distance to patterns (Section 4.3). (iv) All features is the concatenation of the features mentioned above. The training details for the baselines and models are outlined in Appendix C.

Results Table 2 outlines the results of the artificial text detection experiments on the three datasets. Note the diversity of the experiment setting where the methods are tested with respect to the TGM, TGM’s size, the decoding method, domain, and stylistic properties (texts from the Amazon Reviews & GPT-2 are shorter as compared to those of WebText & GPT-2 and RealNews & GROVER). The overall tendency is that the proposed TDA-based classifiers outperform the count-based (TF-IDF, N-grams) and two BERT-based baselines (BERT [CLS trained], BERT [SLOR]) up to 10%. The concatenation of the features achieves the performance on par with the fully trained BERT model on all datasets.

5.2 Robustness towards Unseen Models
This setting tests the robustness of the artificial text detection methods towards unseen TGMs on the WebText & GPT-2 dataset. The baselines and models are trained on texts from the GPT-2 small model and further used to detect texts generated by unseen GPT-style models with pure sampling: GPT-2 Medium (345M), GPT-2 Large (762M) and GPT-2 XL (1542M). Note that such a setting is the most challenging as it requires the transfer from the smallest model to that of the higher number of parameters (Jawahar et al., 2020).

Results Figure 2 demonstrates the results on the robustness experimental setup. The simple linear classifier trained over the Topological features demonstrates the minor performance drop on the task of detecting artificial texts by the larger GPT-style models as opposed to the considered methods. However, the TDA-based classifier performs slightly worse than BERT [Fully trained] on the test subset by GPT-2 Small.

5.3 Attention Head-wise Probing
Data SentEval (Conneau et al., 2018) is a common probing suite for exploring how various linguistic properties are encoded in the model representations. The probing tasks are organized by the type of the property: surface, syntactic and semantic. We use the undersampled tasks⁶ to ana-

⁴https://huggingface.co/bert-base-uncased
⁵https://github.com/jhlau/acceptability-prediction-in-context
⁶Each probing task is split into 25K/5K/5K train/validation/test sets. The sets are balanced by the
lyze what properties are stored in the topological features.

**Method** Attention head-wise probing (Jo and Myaeng, 2020) allows investigating the patterns of how attention heads from each layer of the model contribute most to a probing task. Logistic Regression is trained over the intermediate outputs of the model $h_{i,j}$, where $i$ and $j$ denote the indices of the layer and the attention head. We use the publicly available code\(^7\) to train the classifier over two groups of the input features: (i) the intermediate outputs $h_{i,j}$ produced by the frozen BERT model and (ii) the topological features derived from $h_{i,j}$ as outlined in Sections 4.1, 4.2. The performance is evaluated by the accuracy score, and the heat maps of the probing scores are constructed to inspect how a certain linguistic property is distributed across different layers and attention heads. Refer to Jo and Myaeng (2020) for more details.

**Results** The results demonstrate that the topological features tend to be sensitive to the surface and syntactic properties as opposed to the semantic ones. Figure 3 shows heat maps of the attention head-wise evaluation on LENGTH (Figure 3a, *surface* property) and DEPTH (Figure 3b, *syntactic* property) tasks\(^8\). While the sentence length is distributed across the majority of the frozen attention heads, specifically at the lower-to-middle layers $[1 \rightarrow 5]$, the topological features capture the property at layer $[1]$ and by fewer heads at layers $[2, 4 \rightarrow 5, 9 \rightarrow 11]$. The depth of the syntax tree is encoded in the frozen heads at the lower-to-middle layers $[1 \rightarrow 5]$, whereas the barcode features predominantly localize the property at the middle-to-higher layers $[5 \rightarrow 9]$.

The overall pattern for the surface and syntactic tasks is that the persistent graph features can lose some information on the linguistic properties during the derivation of the features from the attention matrices. The localization of the properties after the derivation gets changed, and the head-wise probe performance may significantly decrease. Notably, the majority of the semantic tasks receive rapid decreases in the probe performance on the persistent graph features as compared to the frozen heads. The reason is that the features operate purely on the surface and structural information of the attention graph, leaving semantics unattended.

6 Discussion

Structural Differences between Natural and Generated texts The TDA-based classifiers rely on the structural differences in the topology of the attention maps to distinguish between natural and generated texts. Figure 4 shows that the distributions of the sum of bars in $H_0$ differ for natural and generated texts. For the former, it is shifted to the left. We provide more examples of the distribution shift for different heads and layers in Figure 5, Appendix B. The weights for natural texts are concentrated more on the edges of the maximum spanning tree (MST), so that the model focuses on the sentence structure, or on the “skeleton” of the MST. The weights for the artificially generated texts are distributed more evenly among all edges. As the TDA-based classifiers appear to be robust towards unseen TGMs, we may conclude that such structural properties are inherent to the models of different sizes, so that shifts in the distribution of the sum of bars in $H_0$ hold for texts generated by different TGMs. This feature appears to be the key one as utilizing it alone for the prediction provides us with the 82% accuracy score on the WebText & GPT-2 dataset.

Semantics is Limited The TDA-based methods do not take the semantic word similarity into account, as they only capture inter-word relations derived from the attention graphs. The probing analysis supports the fact that the features do not encode the semantic properties, carrying only surface and structural information. However, this information appears to be sufficient for the considered task.
7 Conclusion

This paper introduces a novel method for the task of artificial text detection based on TDA. We propose three types of interpretable topological features that can be derived from the attention maps of any transformer-based LM. The experiments demonstrate that simple linear classifiers trained on these features can outperform count- and neural-based baselines, and perform on par with a fully fine-tuned BERT model on three common datasets across various domains. The experimental setup also highlights the applicability of the features towards the TGM architecture, TGM’s size and the decoding method. Notably, the TDA-based classifiers tend to be more robust towards unseen GPT-style TGMs as opposed to the considered baseline detectors. The probing analysis shows that the features capture surface and structural properties, lacking the semantic information. A fruitful direction for future work is to combine the topological features with those that encode the semantics of the input texts, and test the methods on a more diverse set of the TGM architectures, decoding methods and transformer LMs to infer the attention graphs from. We are publicly releasing the code, hoping to stimulate the research on the TDA-based investigation of the inner workings of the transformer-based models and the applicability of TDA methods to other NLP tasks.

Acknowledgement

The work of Serguei Barannikov and Evgeny Burinaev, related to topological data analysis and machine learning in Sections 4 and 5, is supported by Ministry of Science and Higher Education grant No. 075-10-2021-068. Ekaterina Artemova is supported by the framework of the HSE University Basic Research Program.
References


Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc’Aurelio Ranzato, and Arthur Szlam. 2019. Real or fake? learning to discriminate machine from human generated text.


Topological Features Calculation

Algorithm 1 Topological Features Calculation

Require: Text sample \( s \)
Require: Set of chosen attention heads \( H_M \) of model \( M \)
Require: Thresholds array \( T \)
Require: Topological feature \( f \) of unweighted graph
Ensure: Features array \( Features \)

procedure FEATURES_CALC\((s, H_M, T)\)
\begin{align*}
2: & \quad \text{for all } h \in H_M, t \in T \text{ do} \\
4: & \quad \text{Calculate attention graph } \Gamma^h_s = (V, E, W_{h,s}^t) \text{ on sample } s \text{ on head } h \\
6: & \quad \text{Calculate } f(\Gamma^h_s(t)) \\
10: & \quad Features \leftarrow \left[ f(\Gamma^h_s(t)) \right]_{t \in T} \\
12: & \quad \text{return } Features \\
\end{align*}

end procedure

Persistent Homology and Betti Numbers

Recall that a simplicial complex \( K \) is a collection of subsets of a finite set called simplices such that each subset of any element of \( K \) also is an element of \( K \); such subsets of a simplex are called faces. In particular, an undirected graph is a simplicial complex where the simplices correspond to the edges and vertices of the graph. The set of all formal \( \mathbb{Z} \)-linear combinations of the \( p \)-dimensional simplices (that is, \( (p + 1) \)-element finite sets from the collection) of \( K \) is denoted \( C_p(K) \). These linear combinations \( c = \sum_j \gamma_j \sigma_j \) are called \( p \)-chains, where the \( \gamma_j \in \mathbb{Z} \) and the \( \sigma_j \) are \( p \)-simplices in \( K \).

The boundary, \( \partial(\sigma_j) \), is the formal sum of the \( (p - 1) \)-dimensional faces of \( \sigma_j \) and the boundary of the chain is obtained by extending \( \partial \) linearly,

\[ \partial(c) = \sum_j \gamma_j \partial(\sigma_j) \]

for \( c \) as above.

The \( p \)-chains that have boundary 0 are called \( p \)-cycles, they form a subgroup \( Z_p(K) \) of \( C_p(K) \). The \( p \)-chains that are the boundary of \( (p + 1) \)-chains are called \( p \)-boundaries and form a subgroup \( B_p(K) \) of \( C_p(K) \). The quotient group \( H_p(K) = C_p(K) / B_p(K) \) is called the \( p \)-th homology of \( K \). Their ranks \( \beta_p = rk(H_p(K)) \) of these abelian groups are called Betti numbers. The homology and the Betti numbers are classical topological invariants of \( K \).

In particular, a graph \( G = (E, V) \) contains 0-dimensional and 1-dimensional faces. It follows that its topological form is essentially described by the numbers \( \beta_0 \) and \( \beta_1 \), which are the only nonzero Betti numbers. Here \( \beta_0 \) is the number of connected components of \( G \), and \( \beta_1 \) is the number of independent cycles of the graph (which is equal to \(|E| - |V| + \beta_0|\)).

A subcomplex of \( K \) is a subset of simplices that is closed under the face relation. A filtration of \( K \) is a nested sequence of subcomplexes that starts with the empty complex and ends with the complete complex,

\[ \emptyset \subset \mathcal{K}_1 \subset \mathcal{K}_2 \subset \cdots \subset \mathcal{K}_m = K. \]

In particular, to any weighted undirected graph \( G = (V, E) \) one can associate naturally the filtration

\[ \emptyset \subset G_{t_1} \subset \cdots \subset G_{t_m} = G, \]

where \( \{t_i\} \) is the set of all weights of the edges, \( G_{t_i} = (V, E_{t_i}) \) and \( E_{t_i} \) consists of all edges of \( E \) with weight less or equal to \( t_i \).

In our calculations, the increasing filtration is obtained by reversing the attention matrix weights: \( w \mapsto 1 - w \).

The \( p \)-th persistent homology of \( K \) is the pair of sets of vector spaces \( \{H_p(K_{t_i})|0 \leq i \leq l\} \) and maps \( \{f_{i,j} : H_p(K_{t_i}) \to H_p(K_{t_j})|1 \leq i < j \leq l\} \), where the maps are induced by the inclusion maps \( K_{t_i} \to K_{t_j} \).

Each persistent homology class \( \alpha \) in this sequence is “born” at some \( K_{t_i} \) and “dies” at some \( K_{t_j} \) or never dies (Barannikov, 2021, 1994). One can visualize this as an interval \([t_i, t_j]\) or \([t_i, +\infty]\). The collection of all such intervals is called the barcode of the filtration. It is the most useful invariant of the filtration. Note that the information about the persistent homology classes is generally essential to calculate the barcode, whereas the information about the Betti numbers only is insufficient.
In the case of the filtration associated to a weighted graph \( (2) \), the \( H_0 \)-th barcode (respectively, \( H_1 \)-th) consists of the intervals of the form \([0, d_i]\) (resp., \([s_i, \infty]\)) only. Given a number \( l \), the number of intervals of length at most \( l \) for \( H_0 \) (respectively, the number of intervals with the left endpoint at most \( l \)) is therefore equal to the the Betti number \( \beta_0(K_l) \) (resp., \( \beta_1(K_l) \)). We see that in this case, we can recover the barcode from the collection of all Betti numbers \( \beta_i(K_{t_j}) \) where the thresholds set \( \{t_j\} \) is the set of all weights of the edges in the graph. On the other hand, a calculation of the \( H_0 \)-barcode gives all Betti numbers \( \beta_i(K_{t_j}) \) simultaneously for all thresholds.

In TDA, the following filtered simplicial complex is also commonly associated with a graph. The \( n \)-dimensional simplices of the clique complex (or Whitney complex) \( X = X(G) \) of a graph \( G = (E, V) \) are the \((n + 1)\)-cliques of \( G \), that is, complete subgraphs with \( n + 1 \) vertices. For example, its 0-simplices are the vertices, the 1-simplices are the edges, and the 2-simplices are the triangles. The clique complex has a richer topological structure than the graph itself since it may have nonzero Betti numbers \( \beta_n(X) \) for \( n \geq 2 \). Note that the \( H_0 \)-barcode of the filtered clique complex coincides with the \( H_0 \)-barcode of \( (2) \). Several software packages are aimed at calculating the per-
sistent homology of graph clique complex. For this purpose, we have used Ripser++ (Bauer, 2021; Zhang and Xiao, 2020).

The entropy of the barcode is a measure of the entropy of the points in a persistence diagram. Precisely, if we have a barcode as a list of pairs of “birth”, “death”: \( D = \{ (b_i, d_i) \}_{i \in I} \), then the entropy is defined as:

\[
E(D) = - \sum_{i \in I} p_i \log(p_i)
\]

where \( p_i = \frac{d_i - b_i}{L_D} \), and \( L_D = \sum_{i \in I} (d_i - b_i) \).

Each bar in \( H_0 \)-barcode has the form \([0, d_i]\) where \( 1 - d_i \) is the attention weight of the edge which “kills” the connected component corresponding to this bar. The sum of lengths of bars in the \( H_0 \)-barcode coincides with \( 128 - M \) where \( M \) is the sum of edge weights of the attention graph maximal spanning tree. Examples of the shift of distributions of the sum of bars’ lengths between natural and generated texts are shown in the top and middle rows in Figure 5.

### C Training Details

**Topological Features and TF-IDF** Similar to (Solaiman et al., 2019), the training of the linear classifiers is run with the regularization parameter \( L^2 \in [1e^{-5}, 5e^{-5}, \ldots, 0.1, 0.5, 1] \) and the maximum number of iterations \( \max_{\text{iter}} \in [1, 2, 3, 5, 10, 100] \) tuned on the validation set. The topological features are concatenated for each attention head for each encoder layer, and further concatenated. The total number of features for each method is equal to \( 12 \times 12 \times \) number of features per method.

**BERT-based** classifiers are trained with the linear scheduler with the initial learning rate \( lr \in [1e^{-5}, 5e^{-5}, 1e^{-4}, \ldots, 1e^{-1}, 1] \) and epochs number \( e \in [2, 3, 5, 10, 15, 20] \). BERT [CLS trained] is trained for 20 epochs, and BERT [Fully trained] is trained for 2-5 epochs depending on the dataset. We use early stopping controlled by the accuracy on the validation set for each text detection dataset.