Statistically-augmented Neural Detection of AI-generated text

Stanford CS224N Custom Project

Jeffrey Heo
Department of Computer Science
Stanford University
jeffheo@stanford.edu

Simon Kim
Department of Computer Science
Stanford University
ksimon12@stanford.edu

Michael Yan
Department of Computer Science
Stanford University
yanm@stanford.edu

Abstract

The rapid advancement of generative language models has led to an increasing concern of their potential misuse and a growing need to develop effective detection methods. In this paper, we propose a novel approach of augmenting a pre-trained RoBERTa sequence classifier with hand-picked statistical features extracted from the input text. We create a custom "statistical embeddings" layer that extracts various statistical features from the input sequence and injects it into the classification pipeline. We then finetune our model on an open-source dataset of GPT-3 generated text and human-written Wikipedia introductions for 150,000 topics. We evaluate our model against the baseline RoBERTa classifier with a focus on robustness against distribution shifts. Our project aims to leverage statistical information to improve the generalizability of a neural classifier.

1 Key Information to include

- Mentor: N/A
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

In the last year, we have witnessed the debut of powerful language models with the ability to produce text that is nearly indistinguishable from human-written content. OpenAI's ChatGPT, which amassed over 100 million users within two months of its launch, is used by over 13 million users each day to generate wide ranges of content. As research behind these models continues to accelerate, there is a growing need to address their potential misuse. A pertinent example would be in the maintaining of academic integrity via detection of NLG-powered plagiarism. Universities worldwide are instituting rules prohibiting the use of language models for exams and assignments. The ability to distinguish A.I.-generated text from human writing would be especially useful in enforcing such policies.

There are currently two mainstream methods of synthetic text detection. One is a purely zero-shot approach that attempts to flag machine-generated text using a class of well-known statistical indicators. The other is a neural approach which attempts to finetune existing models to be used for the detection task. Between the two methods, the trade-off between efficacy and efficiency are clear.

In particular, finetuning a language model to detect "itself" has proven to be an effective strategy for text detection. However, existing research suggests that a purely neural approach struggles from distribution shifts, and is overly reliant on the decoding strategy with which the training data was generated (Solaiman et al., 2019). In other words, though the neural-based approach is highly *performant* for specific samples of text, it is significantly less *robust*.

Our project attempts to mitigate this shortcoming by combining the statistics- and neural-based approaches in order to achieve the best of both worlds. We hypothesize that, regardless of the language model, machine-generated and human-written text have fundamental statistical differences that can be learned. Our main thesis is that explicitly encoding such statistical features into the input can prevent the classifier from overfitting on the model-intrinsic features present in the training data, thus improving its performance against a greater variety of generated text.

To do so, we add a custom "statistical embeddings" layer to a RoBERTa model pre-trained for the synthetic text detection task. This embeddings layer extracts various statistical features from the input sequence and injects them into the classification pipeline. We finetune both our baseline RoBERTa detector along with our augmented detector on an open-source dataset of GPT-3 generated text and human-written Wikipedia introductions. We observe similar performance for both models on the test set partitioned from the same dataset (i.e. equal distribution). Both classifiers are then evaluated against machine-generated and human-written "long answers" in the PubMedQA dataset (i.e. different topic and distribution from the training set). Despite a marginal increase in accuracy for the WikiIntro dataset, we were unable to see significant improvements in robustness using our approach. We suspect that this may in part be attributed to the sparseness of our statistical feature vectors. The specific results and analysis are presented in section 5.

3 Related Work

There has been significant prior research regarding the statistical, zero-shot approach to synthetic text detection. A study done in 2017 by Nguyen-Son et al. (2017). reported promising results through statistical analysis alone, relying on (i) word distribution frequencies, (ii) complex phrase features, and (iii) sentence- and paragraph-level consistency. However, since 2017, the capabilities of NLG models have grown exponentially, and thus it is expected that previous results will be significantly challenged by state-of-the-art models such as GPT-3.

More recently, Tian released "GPTZero" which analyzes the perplexity and burstiness of a given text to detect whether it was machine-generated. However, GPTZero has been shown to work poorly when dealing with shorter sequences or against adversarial rewording and paraphrasing (Tian, 2022). Perhaps the most significant breakthrough in this regard is "DetectGPT", released in January 2023 by Mitchell et al. (2023). It achieved significantly improved performance compared to other zero-shot classification methods by using the observation that texts generated by LLMs tend to occupy negative curvature regions of the model's log probability function. Though the results are remarkable, the paper reports that supervised models still perform better than DetectGPT for in-distribution data, thus motivating our approach that the neural-based approach has greater potential once its lack of robustness is resolved.

Research in neural-based text detection has been greatly accelerated by high-performing language models that can be applied to a myriad of downstream tasks. The most notable result to date is OpenAI's RoBERTa-based sequence classifier which was released alongside GPT-2. The model was trained on a behemoth corpus of GPT-2 generated text and was able to outperform human detection with an accuracy of 95%. However, research shows that the model's accuracy drops significantly when tested against distribution shifts Solaiman et al. (2019). We later confirm this result in section 5.

4 Approach

Our main contribution to the synthetic text detection literature is in the combining of statistical and neural methods. We attempt to improve upon OpenAI's RoBERTa model by adding a statistical embeddings layer that extracts and encodes useful information about the input sequence. The exact ensemble of features used is described in more detail in section 4.2.

We were greatly inspired by the Transformer architecture and its way of encoding the notion of position in self-attention. Namely, the Transformer architecture extracts positional information and adds it to the input embeddings such that the order of words is accounted for by the model. We perform a similar procedure in which a statistical feature vector is first constructed by our statistical embeddings layer, then added into the input embeddings. The model architecture and design decisions are discussed in more detail in section 4.3.

4.1 Baseline

As described above, our baseline is OpenAI's RoBERTa sequence classifier pre-trained for GPT-2 text detection. The sequence classifier consists of the original RoBERTa model and an additional classification head. The classification head is a standard feed-forward network with dropout regularization, a $\tanh$ non-linearity, and an output projection. We first compare our main model's performance against this pre-trained RoBERTa classifier (trained on GPT-2 data). We then finetune the RoBERTa classifier on GPT-3 data and perform a similar comparison. The results of the baseline experiments are presented in section 5.

4.2 Statistical Feature Selection

We postulate that using an ensemble of statistical features can increase the robustness of our model. In particular, we focus on robustness against *adversarial* prompts, e.g. asking ChatGPT to "generate a text that uses many exclamation marks and commonly-used words." It is intuitive that including multiple features in our detection provides a straightforward hedge against such attacks, as targeting more and more statistical features will only impede the model's ability to generate fluent text.

In this section, we describe in brief detail the statistical features used by our model. We note that the input sequence is first lemmatized using the WordNet corpus, prior to being processed by the statistical embeddings layer.

1. **Zipf**: Zipf's law refers to the observation that many types of real-world data follow a particular distribution. Human-written text is one such example. Formally, given a distribution d_i of the i-th most common lemma in a document,

$$d_i \propto \frac{1}{i} \tag{1}$$

   We use a log scale for both the rank and the frequency of a lemma and perform linear regression. The slope of the regression line f is used as our Zipf feature of the input text.

2. **Clumpiness**: This feature focuses on the fact that human-written text tends to contain particular words more frequently than others, as opposed to machine-generated text which is more uniform (i.e. less clumpy). We calculate the text's Gini coefficient as an indicator:

$$G = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} |x_i - x_j|}{2 \sum_{i=1}^{n} \sum_{j=1}^{n} x_j} \tag{2}$$

3. **Burstiness**: Burstiness measures the intermittent increases or decreases of the frequency of an event, which makes it another good indicator of the relative uniformity of lemma usage. There are many valid ways to measure burstiness. We choose to calculate the index of dispersion, which measures the ratio of the variance to the mean.

$$D = \frac{\sigma^2}{\mu} \tag{3}$$

4. **Kurtosis**: Kurtosis, or the fourth-standardized moment, encodes the presence of outliers. A higher Kurtosis score indicates greater extremity of deviations in the distribution.

$$Kurt[X] = \frac{\mathbb{E}[(X - \mu)^4]}{(\mathbb{E}[(X - \mu)^2])^2} \tag{4}$$

5. **Punctuation**: Another statistical measure is punctuation distribution. We hypothesize that machine-generated text differs from human-written text in the frequency and spread of punctuations (Alexandra N. M. Darmon and Porter, 2020).

6. **Stop word Ratio**: Stop words are words that are generally filtered out in NLP applications due to their disproportionate frequency and comparative semantic neutrality (e.g. “a”, “an”, “the”, etc.) We believe that human-written text will contain a higher ratio of such stop words.

Early data analysis of our WikiIntro dataset (detailed in section 5.1) yields the following plots shown in **Figure 1**. We observe differences in the distribution of the Zipf, Clumpiness, and Kurtosis scores between human-written text and generated text. Given that the sample size is equivalent for both groups of text, we see that the relative modes of the distribution are different.

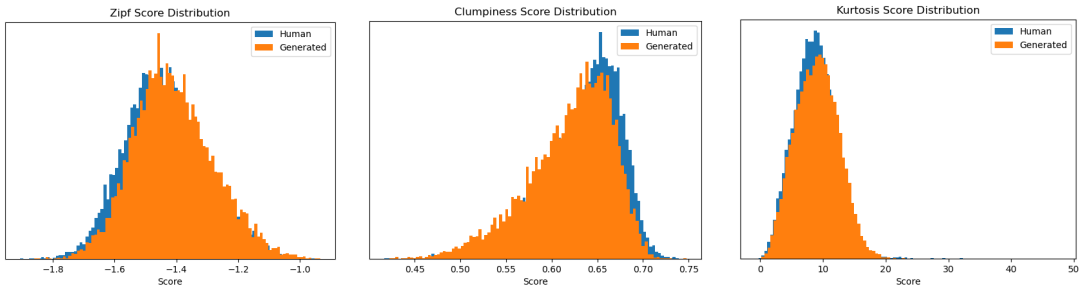

Figure 1: Zipf, Clumpiness, and Kurtosis score distributions for the Wikipedia dataset

4.3 Model Architecture

Our model incorporates a “Statistical Embeddings Layer” which extracts the aforementioned statistical feature vectors from the input sequence and concatenates them to produce a single vector s. The statistical embedding is then passed into a feed forward layer which applies a linear transformation to s then ReLU for non-linearity.

We then experiment with two versions of our model. The early fusion approach sums the output of the above to the inputs of the RoBERTa model (**Figure 2a**). This is largely due to our confidence in the quantity and quality of training data we collected which warrants an early fusion approach. The late fusion approach infuses the statistical features to the *output* of the RoBERTa model instead, prior to being processed by the classification head(**Figure 2b**).

The baseline code for the pre-trained RoBERTa sequence classifier comes from OpenAI Google (2018). We have written code to load and process our dataset as well as extract various statistical features. Furthermore, we make original modifications to the RoBERTa model by adding the layers described above as well as by enabling early and late fusion. Overall, the main innovation of our project lies in our approach to combine neural and statistical methods in the task of machine-text generation.

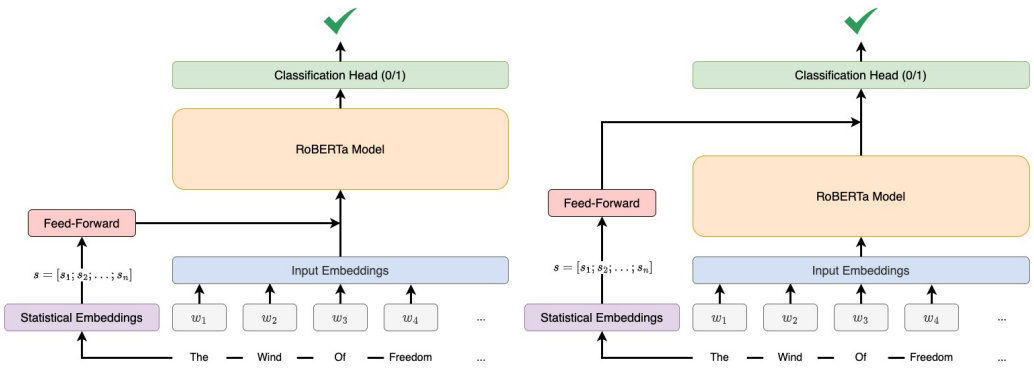

Figure 2: (a) Early Fusion Model and (b) Late Fusion Model

Human Written Intro (Truncated)	*GPT-3 Generated Intro (Truncated)*
"A full-time job is employment in which a person works a minimum number of hours defined as such by their employer. Full-time employment often comes with benefits that are not typically offered to part-time, temporary, or flexible workers, such as annual leave, sick leave, and health insurance. Part-time jobs are mistakenly..."	"A full-time job is employment in which a person works a set number of hours each week, typically 40 hours. In some countries, full-time employment is the norm, while in others it is less common. The term "full-time job" can refer to a variety of different types of employment, including traditional jobs..."

Figure 3: Examples of human-written and GPT-3 completed introductions for *"Full-time job"*

5 Experiments

5.1 Data

We use two publicly available datasets for the training and testing of our baseline and main model. Both datasets consist of human-written text and machine-generated text that are processed and appropriately labeled to be used in our pipeline (examples in **Figure 3**). The first dataset is a collection of GPT-3 generated text and human-written Wikipedia introductions for 150,000 topics (Aaditya Bhat, 2023). We use this data to train our baseline model (as the pretrained model has only seen GPT-2 text) and our main model, both early and late fusion. We use a 80-10-10 split of our dataset for training, validation, and testing. We also pre-process the data by randomly truncating each pair of human-written and model-generated text to the same length.

We then utilize the PubMedQA dataset to test the transferability of our model. In particular, we make use of the "long answers" that contain both human-written and machine-generated samples (Jin et al., 2019). We decided to use the PubMedQA dataset because of its difference in both the topic and the NLG model from our training set, which can provide a good insight into the robustness of our model. The results and analysis of our experiments are presented **section 6**.

5.2 Evaluation method

We use prediction accuracy and area under the receiving operating characteristics (AUROC) as our metrics. AUROC is commonly used to evaluate the performance of binary classification models. It measures the ability of a model to distinguish between two classes by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR). The ROC curve is a graphical representation of the performance of a binary classifier as the discrimination threshold is varied. The area under this curve (AUROC) represents the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. AUROC ranges from 0.0 to 1.0, where a value of 0.5 indicates that the model is no better than random guessing, and a value of 1.0 indicates a perfect classifier.

5.3 Experimental details

The first step of our research involves testing the existing "gold standard" detection model (OpenAI's model pretrained on GPT-2) on our WikiIntros GPT-3 dataset. We use the results of this as the baseline performance before making changes to model architecture and passing in our statistical features.

Following this, we fine-tune the pretrained model using our GPT-3 training set, and test it to establish a baseline for the "gold standard" model's performance after undergoing fine-tuning with a relatively small training set (LR of 2e-05, 5 epochs). We use this to quantify the performance of the existing model when tuned on newer data.

The core of our research focuses on our improved model, which incorporates the aforementioned statistical features. We train this model on the same training set (LR of 2e-05, 10 epochs) and compare the performance with the results from the previous two parts. We perform this training process twice, once on our early-fusion model and once on our late-fusion model.

6 Results & Analysis

Running the tests, we observe that the finetuning the RoBERTa model on the WikiIntros dataset allows the model to achieve near perfect accuracy – and a 16% improvement over the same model trained on datasets generated by older LLM models, in this case GPT-2 (detailed in **Table 1**). We see that our late-fusion model achieves marginally higher performance than this – suggesting that the incorporation of statistical features does help the model improve accuracy.

On reason for the high performance across the models is the nature of the dataset. There are certain high-level differences between the human-written text and AI-generated text, in that the generated text is more generalized/overview focused, while the human-written (which are the Wikipedia pages) might include references to other similar terms/disambiguations. This may contribute to the model overfitting on this particular dataset.

Table 1: Comparison of Model Accuracies and AUROC Metrics

	Wiki Intros		**PubMedQA**	
Model	**Test Accuracy**	**AUROC**	**Test Accuracy**	**AUROC**
Baseline	0.832	0.9117	0.5144	0.5308
Baseline + GPT-3 Finetune	0.9907	0.999	0.5171	0.5359
Early-Fusion Model	0.86626	0.9938	0.5035	0.5294
Late-Fusion Model	0.9956	0.999	0.5058	0.5360

6.1 Explaining Poor Model Performance within PubMedQA

We observed that none of the models performed well on the PubMedQA dataset, implying poor robustness toward distribution shifts. One potential explanation is that a scientific Q&A database has much lower variance when it comes to response content. Correct responses, AI-generated or not, will contain the same content and likely in a similar format, given the nature of the prompt (in this case the question). Whereas in the WikiIntros dataset, there is a much more open-ended prompt, simply describing the topic at hand. Apart from the nature of the dataset, we could also attribute the poor robustness of the model to certain shortcomings of experimental approach. For instance, we believe that the model benefited only marginally from the statistical features due to small magnitudes and sparseness; extracting information regarding punctuation, especially, yielded highly sparse features. A greater variety of statistical features with non-negligible magnitudes may be needed to augment a neural classifier better. Given the nature of our approach in which we performed an element-wise addition of the statistical embedding vector to the input embedding (for early fusion) or output vector of RoBERTa (for late fusion), a relatively sparse vector with low magnitude elements may not have been the optimal way to incorporate our statistical features. Concatenation, instead of element-wise addition, may have been a more favorable choice of fusing in the aforementioned statistical features. Moreover, for the model to learn the linear transformation of these statistical features, we may need to train the model for a larger number of epochs.

7 Conclusion

Through our research, we were able to identify statistical features that when fed into our RoBERTa model (late-fusion) was able to marginally outperform the existing gold standard model for AI-generated text detection task on the WikiIntros dataset. Both models performed very well on the task with near perfect accuracy, and were significantly better than existing models trained on generated text from older models (GPT-2).

Our research sought to achieve greater robustness towards distribution shifts through augmenting a neural classifier with statistical features of the input text. When we tested our baseline and early/late fusion models that were trained on the WikiIntros dataset using the PubMedQA dataset, we witnessed the late-fusion model achieve marginal improvements in the AUROC metric compared to the baseline. However, the improvement itself is too marginal, and the magnitude still remains in the vicinity of 0.5, suggesting the model is only marginally better than a random guess even after incorporating

statistical features. As delineated in section 6.1 above, we believe that this incorporation of statistical features is worthy of further exploration through modifying and increasing the number of statistical features, modifying the method of feeding the statistical features into the model, and more.

A core limitation of our work lies in the fact that we have tested our model's robustness towards data distribution shift using just one dataset distinct from the wiki introduction dataset used in the model training procedure: the PubMedQA dataset. Had we tested our models with a larger number of distinct datasets, our research would have been able to test the relationship between the incorporation of statistical embeddings into machine-text detection and model transferability more extensively.

References

Aaditya Bhat. 2023. Gpt-wiki-intro (revision 0e458f5).

Sam D. Howison Alexandra N. M. Darmon, Marya Bazzi and Mason A. Porter. 2020. Pull out all the stops: Textual analysis via punctuation sequences. *European Journal of Applied Mathematics*, 32(6):1069–1105.

Google. 2018. modeling_roberta.py.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. *CoRR*, abs/1909.06146.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature.

Hoang-Quoc Nguyen-Son, Ngoc-Dung T. Tieu, Huy Hoang Nguyen, Junichi Yamagishi, and Isao Echizen. 2017. Identifying computer-generated text using statistical analysis. *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1504–1511.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models. *CoRR*, abs/1908.09203.

Edward Tian. 2022. Gptzero.