Generated Text Detection by Fine-tuning the Simple ChatGPT Detection model with Expanded Dataset and Prompt Engineering

CSC3160 Project

Ma Yufei (119010233)
School of Data Science
Chinese University of Hong Kong, Shenzhen
`yufeima@cuhk.edu.cn`

Li Xiangyi (119010153)
School of Data Science
Chinese University of Hong Kong, Shenzhen
`xiangyili@cuhk.edu.cn`

Abstract

This research investigates the efficacy of augmenting datasets and leveraging adversarial prompting attacks to refine the ChatGPT Detector RoBERTa model. We generated and harvested data from different language models and employed various feature engineering techniques on the dataset. The results showed that substantial improvement in the model's accuracy could be achieved with an expanded dataset and adversarial prompting. The model's performance could vary depending on the quality and diversity of the data used for fine-tuning.

1 Key Information to include

- Contributions of each member:
  Yufei Ma (50%): Feature engineering on the dataset. Literature review. Prompting generation and AutoGPT setup. Wrote the majority of the reports for the proposal, milestones, and the final report.
  Xiangyi Li (50%): Set up multiple LLM models/APIs (ChatGPT, Dolly, Bloom, Cohere). Set up VMs and GPU computes and chron jobs. Designed training and testing pipelines. Designed test benches and data analysis with results.
- I would like to give up 1% in poster session and move the reminding 4% to project report.

2 Introduction

In the era of rapid technological advancements, natural language processing (NLP) and generative models have made significant strides, giving rise to sophisticated text generation systems like ChatGPT. These systems boast an uncanny ability to emulate human-like text, often blurring the distinction between human and AI-generated content. While the implications of this progression are far-reaching, it also ushers in a plethora of concerns about the misuse of such technology, which may fuel the propagation of disinformation and creation of fabricated news.

Addressing these concerns has prompted researchers to dedicate their efforts towards devising reliable methodologies for detecting machine-generated text. Several algorithms like GLTR, DetectGPT and RoBERTa for the detection seem to have high accuracy [1, 2, 3]. However, according to Sadasivan et al. [3], these methods can be vulnerable since paraphrasing attacks and adversarial humans can greatly reduce the classification accuracy.

Our research, therefore, embarks on the quest to investigate the efficacy of augmenting datasets and leveraging adversarial prompting attacks enlightened by AutoGPT to refine the ChatGPT Detector RoBERTa model. This endeavor is driven by the desire to boost the model's performance, thereby ensuring its capability to accurately discern between human and AI-generated text. This challenge holds immense importance and is fraught with difficulties, given the imperative need for high-precision text detection in curbing the potential misuse of AI-generated content and mitigating its adverse effects on society.

3 Related Work

According to Tang et al. [4], the detection task is typically approached as a binary classification problem, aiming to capture textual features that differentiate between human-authored and LLM-generated texts. Several studies have been conducted to explore a good tool for AI-generated text detection. A study by Gehrmann et al. [1] developed GLTR (Giant Language model Test Room), which was based on statistical methods, as a tool to support detecting content generated by models. GLTR has the assumption that most systems sample from the head of the distribution, thus word ranking information of the language model can be used to distinguish LLM-generated text. Another detection algorithm named DetectGPT proposed by Mitchell et al. [2] uses probability curvature as a criterion to built a tool that don't need fine-tuning. They found out that unlike human-written text, model-generated text tends to lie in areas where the log probability function has negative curvature. Thus, the DetectGPT can be discriminative with only log probabilities computed by the model of interest and random perturbations of the passage from another generic pre-trained language model. Despite algorithms with machine learning techniques, deep learning models like RoBERTa can also do the detection according to Hugging Face[5, 6]. In 2023, Guo et al. [7] embarked on an in-depth evaluation and detection of ChatGPT, comparing it to human experts. They conducted GLTR and RoBERTa and compared their performances on the HC3 dataset. The results show that RoBERTa has the highest F1 score 98.78.

It seems that AI-generated text detector can efficiently classify AI-generated text from human-written text. However, according to Sadasivan et al. [3], both algorithms with certain model and methods applied watermarking techniques are vulnerable. They show that paraphrasing attacks and adversarial humans can make the classification results no better than random guess. Another research done by Krishna et al. [8] also pointed out that paraphrasing evades detectors of AI-generated text. To improve the algorithm, they proposes a retrieval method as a possible effective defense, which is to search a database of sequences previously generated by the language model's API and match the candidate text within a certain threshold. However, though information retrival is possible to make detection possible in cases where the detecting party could acquire access to such retrival API that might be provided by popular LLM providers, as LLM are emerging fiercely with new LLM products launching each day and tools like the LangChain [9] and open-source LLM are being democratized, harvesting all sequences generated by these LLM is impractical.

The dataset used by [7] is collected from ChatGPT without special prompts. We want to enhance the robustness of the model by adding dataset after prompt-engineering, and fine-tuning the model. According to Si et al. [10], promting engineering can improve the reliability of AI results like ChatGPT, which may lead to AI generated text harder to be detected. Based on this we hope to boost AI model's ability to evade the detector through adversarial prompting.

According to [7], by the time it was released, the research team needed to deploy web crawlers to harvest data from the ChatGPT web client. The ChatGPT API is not available to the public, and the access to other open-source LLMs and LLM API like [11, 12, 13] is not available. Thus the potential for the fine-tuning the RoBERTa base model used in [7] to gain robustness and generalization performance could not be fully exploited due to insufficient data source. In this paper, we first continue the method in [7] by expanding the dataset. Then, according to community feedbacks to this dataset and similar detection tools, we fine-tune the baseline model incrementally with newly generated data to test data generated by newer models to draw the conclusion whether the RoBERTa based approach is generalizable.

4 Approach

We started off by manually testing the capabilities of the baseline ChatGPT Detector with RoBERTa model (short as BR model) by manually acquiring data and see if it's accurate. There are several ways we've tried to harvest both human written and LLM generated texts.

1. Manually come up with prompts to generate texts through the ChatGPT web client

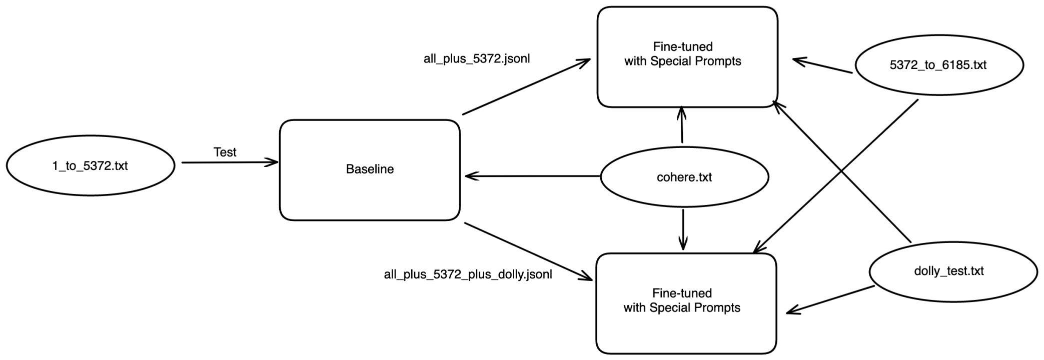
1_to_5372.txt
Test
Baseline
all_plus_5372.jsonl
Fine-tuned with Special Prompts
5372_to_6185.txt
cohere.txt
all_plus_5372_plus_dolly.jsonl
Fine-tuned with Special Prompts
dolly_test.txt

Figure 1: Workflow of the fine-tuning and evaluation

2. Deploy AutoGPT locally to mass produce prompts with the objective to find easy prompts that could evade the baseline model's detection
3. Generate texts using other language models like Claude, Dolly, Alpaca with the help of tools like the LangChain and Vercel.
4. Scrape the internet for historical documents like the Origin of Species to test if it will be recognized as generated texts.

To boost and test the generalization performance and robustness of the dataset, we conduct our experiment implementing these steps.

- Generate new ChatGPT answers using `gpt-3.5-turbo` API with special prompts learned from the exploration stage as output-1.
- Generate texts using other language models, in our case `databricks/dolly-v2-3b` and `cohere/command-xlarge-nightly` as output-2 and output-3 respectively
- Besides the baseline detector model, we train two models with data from output-1 and output-1 and output-2 combined. We then test the performance of output-0 (the output from [7]) to output-3 to the three models to conclude about the generalization performance.

5 Experiments

5.1 Data

In the exploration stage, we were able to obtain generated texts by multiple models through Vercel's AI Playground, some of which were based on Meta's LLaMA model. We also obtained similarly small amounts of generated texts with locally deployed Stanford's Alpaca model [14]. Commercial products like the New Bing, Character.ai are also being tested. As of the writing of this project, PaLM 2 has only opened the waitlist for its API in Firebase two on May 10 [11], and open-source models based on Meta's LLaMA [15] could not distribute easily due to licensing reasons. It turned out except the New Bing, other products or language models evades the detection of the detector quite easily, as well as ChatGPT with special prompt.

In the exploration stage, we also performed feature analysis comparing different response types (ChatGPT response without special prompts, and ChatGPT response with special prompts) regarding the average word length, number of sentences, number of words, and perplexity score. The result is displayed in Figure 2.

Then, we set up multiple APIs and HuggingFace open-source models to mass generate texts data and later label them as `fake` in the training dataset.

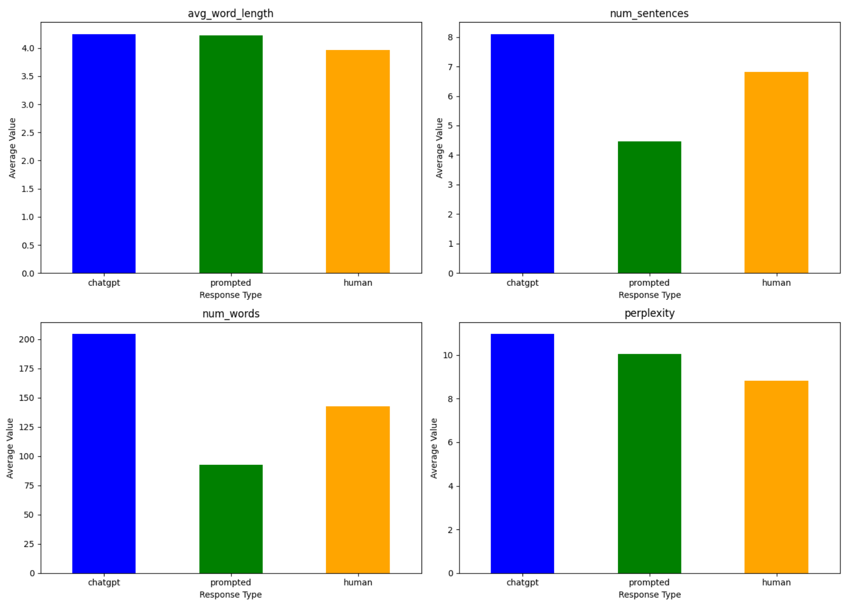

Figure 2: Feature analysis for different response type

5.2 Evaluation method

We measure the robustness and generalization performance by the accuracy of its classification of Generated and Human texts on a new or test dataset. Table 1 provides an example of how we treat the structure of the accuracies for different models.

5.3 Experimental details

For text generations, we tried out these models or APIs:

- ChatGPT API with the default `gpt-3.5-turbo` model: In setting up this API, we encountered problems mainly related to networking, billing, and rate limiting. To successfully generate a dataset of around 5,000 rows, we managed to
  - Networking: we purchased a virtual machine from a cloud computing provider, set up a data tunnel based on the GOST framework [**?**], and used the VM to access the ChatGPT API [16].
  - Billing: we managed to set up a virtual credit card using cryptocurrencies to pay for an $120 API usage plus a $5 grant.
  - Rate limiting: the use case for the ChatGPT is not for massive and incessant for individual users. Thus, we managed to set up a fallback logic with Linux bash scripts to ensure the process would not get hung up when rate limited or when the model is being overloaded.
- Dolly model with the `databricks/dolly-v2-3b` model: Before Dolly [12], we tried several other models to generate data like the BLOOM model [13], the FLAN model [17], the Alpaca model [14], and the OpenAssistant `OpenAssistant/oasst-sft-6-llama-30b` model [**?**]. These could generate texts that successfully evade the baseline detector, that is, being detected less than 50% of the time. However, the text generation with these models could not scale due to various reasons, including non-compatible prompting and output, licensing issues, lack of mass generation APIs, and incompatibility with the Colab environment that currently has no workarounds. We thus used the Dolly model and generated around 800 rows of data. This was due to the Colab's interactive shell's RAM problem when using the Dolly model, and models with larger parameters failed to generate any data as well.

- Cohere NLP API: This is a commercial product that provides NLP APIs for tasks like chat completion and text summary for developers to integrate with different applications. This is very easy to set up, and we generated 100 rows of the data as the final test score to test the three models.

5.4 Results

The result is shown in Table 1.

Model	**Baseline Test Dataset**	**Prompt Test Dataset**	**Dolly Test Dataset**	**Cohere Test Dataset**
Baseline Detector with RoBERTa	0.9918	0.5899	0.17734	0.1300
Fine-tuned Detector with RoBERTa	0.9367	0.9124	0.2769	0.1500
Fine-tuned Detector with RoBERTa + Dolly	0.9971	0.9772	0.9500	0.5100

Table 1: Performance of different models on various datasets.

6 Analysis

Table 1 demonstrated the potential of fine-tuning the ChatGPT Detector model with additional data and adversarial prompting. Our experiments revealed that such approach can indeed improve the accuracy of AI text detection, providing a viable way to enhance the robustness of AI text detectors against advanced generative models.

One limitation of our study is that the performance of our proposed model could vary depending on the quality and diversity of the data used for fine-tuning. This is shown that even 800 rows generated by a new language model could significantly corrects the overfitting problem in the baseline and fine-tuned model with output-1 and output-2. However, the model still struggles to beat random guess when being tested on the texts generated by Cohere. Therefore, continuous monitoring and updating of the model by feeding more data and adversarial prompting would be essential to maintain the performance of such a detector, especially considering the rapid evolution of generative models. Other challenges like robustness to paraphrases and human adversarial attacks and expanding the dataset to even larger are not investigated.

However, the significant improvements with a mere 800 rows of Dolly generated data could mean that ICL may address problems of blindness to texted generated by novel language models.

7 Conclusion

Our research explored the potential of fine-tuning the Simple ChatGPT Detection model using an expanded dataset and adversarial prompting. The primary objective was to improve the model's robustness and generalization performance. We generated and harvested data from different language models and employed various feature engineering techniques on the dataset. While the results showed that substantial improvement in the model's accuracy could be achieved with expanded dataset and thus hinted the potential of more powerful detectors with larger datasets, limitations of such detection method is being shown like overfitting to known dataset which could lead to the accuracy of detection to be less than 50%. Follow-up studies could be expanding dataset as well as figuring out ICL with few-shots text rows generated by a target novel LLM.

As language models continue to evolve rapidly, keeping pace with the detection models is a daunting task. Future work could focus on leveraging techniques like retrieval method, prompt engineering, and adversarial attacks to further enhance the robustness of AI-generated text detection models. It is imperative to continue this research to mitigate the potential misuse of AI-generated content and its negative implications on society.

References

[1] Sebastian Gehrmann, Hendrik Strobelt, and Alexander M. Rush. Gltr: Statistical detection and visualization of generated text. 2019.

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

[3] Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. Can ai-generated text be reliably detected? 2023.

[4] Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. The science of llm-generated text detection. 2023.

[5] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. 2020.

[6] Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. Quantifying the carbon emissions of machine learning. 2019.

[7] Ziyuan Wang Minqi Jiang Jinran Nie Yuxuan Ding Jianwei Yue Yupeng Wu Biyang Guo, Xin Zhang. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. 2023.

[8] Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. 2023.

[9] Harrison Chase. LangChain, October 2022. `https://github.com/hwchase17/langchain`.

[10] Zhengyuan Yang Shuohang Wang Jianfeng Wang Jordan Boyd-Graber Lijuan Wang Chenglei Si, Zhe Gan. Prompting gpt-3 to be reliable. 2023.

[11] Sundar Pichai. Google i/o 2023 keynote: Sundar pichai on ai products. `https://blog.google/technology/ai/google-io-2023-keynote-sundar-pichai/#ai-products`, May 2023.

[12] Databricks. Dolly: The first open commercially viable instruction-tuned llm. `https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm`, April 2023.

[13] BigScience Workshop. BLOOM (revision 4ab0472), 2022.

[14] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

[15] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[16] OpenAI. Brand guidelines: Language and assets for using the openai brand in your marketing and communications., October 2023.

[17] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners, 2022.

A Appendix (optional)