Project Status Report

Alex Sotiropoulos and **Jugal Wadhwa** and **Akshay Padte** and
Sathiya Murthi Sankaran and **Zheng Bao**
University of Southern California, Los Angeles
{asotirop, jwadhwa, padte, sathiyam, baojames}@usc.edu

1 Tasks Performed

The first task we performed involved reviewing the code repository used in the paper written by Shi et al. (2023). To replicate their results, we performed runs of different parameters, such as `base_model`, `attack_model`, and `attack_method`. Due to our current compute limitation, we reduced the sample size to just 20 (originally 100). On average, we observed the AUROC score for DetectGPT without the attack was around 0.8. For attacked passages using random (called query-free attack in the original paper) word replacement with synonyms generated by the attack model it was 0.33. The score of attacked passages with word replacement optimized using a genetic algorithm to yield lower log probability under the base model (called query-based attack in the original paper) was 0 (all attack passages misclassified).

We also reviewed the code repository used for the more recent zero-shot technique Binoculars from the paper written by Hans et al. (2024). The paper suggests a two-model approach where two large pre-trained LLMs are used to estimate the perplexity and cross-perplexity of a sample text, which are then used to compute a score that can detect LLM-generated text more robustly even if the model used to generate the text is not related to the pre-trained LLMs used to compute the score. The paper claims to significantly improve detection compared to other zero-shot techniques that rely on perplexity as they would fail to detect LLM-generated text when the prompt used to generate the text is structured in a way that results in higher perplexity, a characteristic of human-written text. We attempted to run the model locally but faced compute limitations as the GPU ran out of memory. However, the repository also provides an online demo to test the model, giving a simple binary output on whether the text was Human or AI-generated. We tested the sample texts we used to test DetectGPT. We got the following accuracies: 0.4 for attacked passages using random word replacement, and 0.0 for attacked passages using genetic algorithm-based word replacement.

We identified another consideration after reviewing the approaches for red teaming outlined in Shi et al. (2023). As mentioned earlier, one approach the authors take is replacing select words with synonyms with the lowest negative log probability. We noticed that this approach tends to result in a tremendous decrease in the text quality. For example, examining the quotes below, we can see that this method results misrepresenting common phrases ("white power" to "Fair power"), and even creating phrases that an average person would not use ("60-year-old" to "a golden ager employee").

It is clear that while this method successfully misleads zero-shot detection methods effectively, it does so at the cost of text quality. In reality, someone trying to mask machine-generated text as their own would desire a certain level of quality while retaining the overall meaning of the text sample. Ultimately, the goal of an attacker is not only to fool the detector but also to convince a human reader of the genuineness of the text. Thus, we propose improving their process by introducing a minimum quality threshold for attacked text excerpts.

LLM-GENERATED, Unmodified:

> [...] was singled out by Zack Davies who was heard saying "white power" [...]. Trevor Jones was also jailed for 19 years over the attack at the store which left a 60-year-old employee [...].

QUERY-BASED ATTACK:

> [...] was singled out by Ethan Davies who was Registered saying "Fair power" [...]. Trevor Jones was also jailed for 19 years over the Onslaught at the store which went away a golden ager employee [...].

2 Risks and Challenges

One challenge is running compute intensive tasks at a smaller cost which would mean obtaining compute resources are a reasonable rate.

Prioritizing text quality presents a challenge, primarily due to its subjective nature. To navigate this, we plan to look for existing analytical tools that offer concrete metrics for assessing text quality. This will enable us to have a quality threshold, facilitating comparative analyses.

3 Plan to Mitigate Risks and Address the Challenges

These systems would run much more smoothly on systems with GPUs and much larger amounts of compute power. Hence a possible solution is to obtain an instance with GPU or TPU. A cloud provider like GCP or AWS are possible solutions in this case and are looking to review any platform that would provide the resources at a cheaper rate.

Another possible solution to address the compute limitation is to use the smaller version of the models. But this would make any comparisons with other papers that run the larger models meaningless, and we would likely have to re-run the experiments to get their results.

In case we have difficulty making use of automatic evaluation method for quality, a comprehensive evaluation framework that includes human judgment with precise guidelines becomes essential. For human evaluators to achieve a standardized assessment process, it will be crucial to establish a comprehensive set of criteria that detail the aspects of quality to be evaluated, such as fluency, coherence, relevance, and factual accuracy. Further, we would need to have multiple independent evaluators for each passage and use the average of their scores to mitigate individual biases and variance in evaluations, while ensuring evaluators are blind to whether a passage is original or "attacked" to prevent bias.

Currently, the code replaces all words except stop-words. One possible solution discussed is to prevent the replacement of proper nouns and possibly even nouns to reduce the amount of changes as such words would typically not be replaced when done by humans and this would also possibly help preserve the quality of sentences.

4 Individual Contributions

- Akshay: Went through the code for the paper by Shi et al. (2023) and ran it with different parameters. Fixed certain issues due to Open AI library version. Raised the issues about the quality of the passages of query-based attack method. Made the argument for why quality of attack passages is important (relates to the use-case of someone using LLMs to generate text in the first place), and suggested a quality threshold for the attacks.

- Alex: Created our project repository; made updates for outdated packages to get code working; reviewed code sections pertaining to red teaming against watermarking; email introduction to TA.

- Jugal: Read the code for both DetectGPT and Binoculars method. Gained a basic understanding of a the code flow in regards to detection in both techniques. Tried to gain an understanding of the resources needed in regards to the execution of both techniques. Tried to look at possible providers for compute resources to determine a solution for execution of the code.

- Sathiya: Studied the paper by Hans et al. (2024) and reviewed the code for the Binoculars method. Set up a local environment and attempted to run the model. Tested the non-attacked and attacked samples on the online demo for Binoculars and checked performance. Explored options available to determine the content quality and readability.

- Zheng: Studied red-teaming techniques. Explored automatic and human evaluation methods and pipelines for assessing model-generated texts.

References

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2024. Spotting llms with binoculars: Zero-shot detection of machine-generated text.

Zhouxing Shi, Yihan Wang, Fan Yin, Xiangning Chen, Kai-Wei Chang, and Cho-Jui Hsieh. 2023. Red teaming language model detectors with language models.