Looking Under the Hood of DetectGPT

Stanford CS224N Custom Project
External Mentor: Eric Mitchell. Staff Mentor: Isabel Papadimitriou

Maximilian Du* maxjdu@stanford.edu

Ryan Lian* ryanlian@stanford.edu

Kaien Yang* kaieny@stanford.edu

Abstract

Large Language Models (LLMs) have revolutionized natural language processing, but their ability to generate highly convincing machine-generated text raises concerns about their misuse. DetectGPT (Mitchell et al.) is a zero-shot LLM detection algorithm that perturbs the wording in a text sample and uses the changes in likelihood under an LLM as a discriminative signal. In this work, we analyze DetectGPT in three areas: improving DetectGPT performance by selectively perturbing certain types of words, discovering adversarial attacks that can systematically fool DetectGPT, and evaluating DetectGPT on newer LLMs such as ChatGPT. Our experiments demonstrate that selectively masking a combination of nouns, verbs, and adjectives improves the AUROC metric by up to 9.5%, demonstrating the importance of targeted masking strategies. Additionally, we reveal a limitation of DetectGPT on *adversarial contexts*, where a snippet of text prepended to the prompt can degrade performance by up to 14%. Finally, we demonstrate that ChatGPT is challenging to detect through DetectGPT. In some cases, we find that prompting ChatGPT to impersonate other entities can further degrade performance. In total, our work provides an analysis of a state-of-the-art LLM detection algorithm and shows potential improvements and vulnerabilities.

1 Introduction

In recent years, large language models (LLMs) (Radford et al., 2018; Brown et al., 2020; Chowdhery et al., 2022; OpenAI, 2022; Zhang et al., 2022), have seen improvements across a variety of language-related benchmarks. Most notably, these LLMs can generate coherent, relevant, and convincing texts. Because of their increasing realism, there has been ongoing work on creating a reliable LLM detection algorithm.

Concretely, we are interested in the following problem: given a *candidate* text, can we determine if it came from a specific LLM or from a human? In this work, we look at a state-of-the-art DetectGPT algorithm introduced by Mitchell et al.. We look at ways of improving DetectGPT performance by creating more meaningful perturbations on the candidate text. The unmodified DetectGPT randomly selects words to perturb. However, in the English language, there are types of words, like prepositions, that are tightly bound to the words around them and therefore not too indicative of machine or human generation. In contrast, there are other types of words, like nouns, that have more freedom through synonyms and antonyms.

Next, we look at ways of decreasing DetectGPT performance through special pieces of text prepended to an LLM prompt. LLMs like GPT2 (Radford et al., 2018) are good at matching the style and content of its prompt. If we add a small but distinctively-styled piece of text (such as poetry) before the main text, we can potentially change the generation significantly enough to fool DetectGPT. We also show how selective masking can be used to counteract this adversarial attack.

Finally, we apply DetectGPT to the new ChatGPT (OpenAI, 2022) and conduct systematic evaluations through the newly released ChatGPT API.

2 Related Work

This section provides a brief overview of the current landscape of detection methods and introduces DetectGPT (Mitchell et al.), a zero-shot method that lays the foundation for our analyses.

2.1 Supervised Detection Approaches

Supervised detection approaches train a model explicitly for the purpose of detection and learn to discrimate between human-written and machine-generated text through labeled examples. Many such methods exist, including those that leverage neural representations (Bakhtin et al., 2019; Solaiman et al., 2019; Uchendu et al., 2020; Ippolito et al., 2020; Fagni et al., 2021), bag-of-words (Solaiman et al., 2019; Fagni et al., 2021), and hand-crafted features (Gehrmann et al., 2019). Although they demonstrate reliable performance, it has been observed that supervised models (models trained explicitly for detection) tend to overfit their training domains and source models Bakhtin et al. (2019); Uchendu et al. (2020).

2.2 Zero-Shot Detection Approaches

A popular alternative for supervised detection is zero-shot detection, which does not require additional data collection or training. For example, Solaiman et al. proposed a method that thresholds the raw log probabilities of the candidate passage when evaluated on the source model Solaiman et al. (2019). Other algorithms have continued to look at the per-token conditional distributions of the source model, including log probabilities, token ranks, and predictive entropy (Gehrmann et al., 2019; Solaiman et al., 2019; Ippolito et al., 2020). Although these methods can be used with most LLMs, they often have lower detection accuracies compared to large supervised methods.

2.3 Watermarking

Another relevant method is watermarking, in which an LLM embeds hidden signals into the text during generation. Specifically, each generated token serves as the seed for a pseudo-random number generator that partitions the model's vocabulary into a whitelist and blacklist, and subsequent tokens are chosen only from among those in the whitelist. Detection results are made based on the proportion of whitelisted tokens in a piece of text (Kirchenbauer et al., 2023).

Fundamentally, watermarking requires the collaboration of the generating party to implement the whitelisting-blacklisting algorithm. DetectGPT makes no such assumptions, allowing it to generalize to a larger collection of models Mitchell et al..

2.4 DetectGPT

DetectGPT is a zero-shot detection method introduced by Mitchell et al. with higher discriminative power than existing zero-shot methods. DetectGPT relies on the assumption that LLM text is sampled mostly at the mode of its text distribution, while human texts may lay anywhere on the text distribution.

Formally, given a source (generating) model p_θ, a candidate passage x, and a perturbation function $q(\cdot|x)$, we can produce a minor perturbation (rephrasing) of x to get a perturbed text $\tilde{x} \sim q(\cdot|x)$. Once we generate enough perturbations, we can calculate the perturbation discrepancy $\mathbf{d}(x, p_\theta, q(\cdot|x)) \triangleq \log p_\theta(x) - \mathbb{E}_{\tilde{x}\sim q(\cdot|x)}[\log p_\theta(\tilde{x})]$. If $\mathbf{d}(x, p_\theta, q(\cdot|x))$ is positive, then $x \sim p_\theta(x)$.

The perturbation function q should give a distribution over $\tilde{x}$, making slight changes while preserving meaning. This could be asking human to rewrite x, or in the paper, randomly masking tokens of x then using T5 (Raffel et al., 2020) to fill in the mask. In case we do not have access to p_θ or want to use other model to perform detection, we can use a scoring model (another LLM) s_θ in place of p_θ to calculate perturbation discrepancy instead.

Although this method has demonstrated reliable performance, it still has some limitations. First, it is compute-intensive due to the need for computing many perturbations. Moreover, since DetectGPT relies on the curvature of a text distribution, it is prone to attacks such as adversarial prompting that manipulate the probability curvature. Finally, since DetectGPT's central hypothesis is framed in terms of maximum-likelihood-based generation, it might be less effective on newer language models like ChatGPT (OpenAI, 2022) that incorporate new training techniques such as reinforcement learning from human feedback. Our work serves as an initial investigation to address these limitations.

3 Approach

For our project, we use the codebase developed by Mitchell et al. and modify it to allow experiments with 1) different masking approaches, 2) adversarial prompting, and 3) the new ChatGPT API.

3.1 Targeted Masking

A critical part of DetectGPT is the perturbation of a candidate text, which requires masking and re-filling parts of the candidate text with a model like T5. The existing DetectGPT algorithm randomly masks tokens to refill (Mitchell et al.). While naive approach has yielded robust results, we hypothesize that performance can be improved through more deliberate masking, which can reduce the compute necessary to arrive at similar results.

Intuitively, both LLMs and humans can understand and recreate grammatical text. The English language relies on many auxiliary parts of speech, like conjunctions and prepositions. These auxiliary parts of speech are often wholly determined by the surrounding text. Therefore, as long as an LLM stays grammatical, masking and refilling these auxiliary parts of speech may tell us limited information about the text's origins.

In contrast, some other parts of speech are not tied strongly to grammar. For example, many different nouns, as long as they have the right plurality, can be used to replace any noun in a piece. Therefore, these parts of speech (nouns, verbs, adjectives, proper nouns, adverbs) may be more indicative of human or LLM generation. Humans might, especially in creative writing, pick descriptions (adjectives, nouns) that sound novel. Novel descriptions will not fall on a mode of an LLM's probability distribution, which is exactly what DetectGPT needs.

To test this hypothesis, we modify DetectGPT to increase the masking probability of certain parts of speech (POS). See Experiment 1 for concrete details and results.

3.2 Adversarial Context

DetectGPT relies on the assumption that text from LLMs are sampled near the modes of their distribution. Mitchell et al. experimented with cross-LLM detection schemes that use one LLM as a generator and another as the detector. These experiments demonstrated that DetectGPT remains mostly robust to the text distribution mismatches imparted by different LLMs. However, even with the same LLM, the text distribution can change drastically with a different context. This allows the LLM to generate very versatile pieces of text, but it also raises a potential avenue for adversarial attacks.

In the current setup, an LLM generation is created by taking some text t as a prompt to the model, which generates g. The total text is $t + g$. We can consider adding some adversarial context c, such that the prompt becomes $c + t$. After generation, we still use $t + g$ as our total text.

GPT2 variants have been observed to mimic the style of their prompt. Additionally, poetry and other modes of creative writing often have unconventional styles that can potentially shift the text distribution of an LLM. These two observations motivate experiments which explore using c from classic literature, poetry, and nonsense text in an attempt to degrade DetectGPT performance, with sentences from random Wikipedia articles serving as a baseline for c. See Experiments 2 and 3 for concrete details.

3.3 Integrating ChatGPT

The ChatGPT API was unavailable when Mitchell et al. published the original DetectGPT algorithm. In this work, we extend the original analysis by evaluating DetectGPT's performance on ChatGPT-generated text. Specifically, we take advantage of the fact that ChatGPT is specially tuned to take instructions and explore direct adversarial prompting by asking it to write in the voice of some persona. For all ChatGPT experiments, we use other models like GPT3 and GPT2 variants for scoring, since ChatGPT API does not provide direct access to the conditional distributions of individual tokens. See Experiment 4 for concrete details and results.

4 Experiments

4.1 Data

Consistent with Mitchell et al., we use the XSum Narayan et al. (2018) and Reddit WritingPrompts datasets Fan et al. (2018), which contain news articles and creative writing pieces, respectively, to serve as human-written text samples. The two datasets are stylistically quite different: XSum is more predictable but covers more complicated subject relationships, while WritingPrompts contains novel content that tries to tell a story; as such, any performance differences across the two datasets can provide meaningful insight into DetectGPT's strengths and weaknesses.

To obtain samples of machine-generated text, we sample the first 30 tokens from each article in the XSum and WritingPrompts datasets, and then prompt an LLM to complete the text. For experimental consistency, we do not modify the data loading pipeline from Mitchell et al..

4.2 Evaluation Metric

Similarly to Mitchell et al., we use the area under the receiver operating characteristic curve (AUROC) as our metric. AUROC is a common choice for discrimination tasks, as it is agnostic to the threshold of selection. A higher AUROC value indicates a greater degree of separation between the distribution of LLM-generated scores and human-generated scores. In addition to the AUROC, we also examine qualitative properties of the generated text, especially with adversarial contexts.

4.3 Experimental Details

We split our experiments into four parts. For almost all experiments, we use 150 paired samples from the dataset.

- *Experiment 1: Selective Masking.* We use the Stanza NLP library (Qi et al., 2020) to select releant parts of speech for masking. For these experiments, we generate text from the 6-billion parameter GPT-J LLM, fill masked text with T5-3B, and score the text with GPT2-medium. We collect performances across 1, 2, and 5 perturbations. The computational bottleneck of DetectGPT lies in the number of perturbations made on the text, which each require a query to T5-3B. Although using a more powerful scoring model and a larger number of perturbations can yield higher performances, we intentionally use a low-resource scoring model with limited perturbations to show that selective masking can help when resources are limited.
- *Experiments 2, 3: Adversarial Context.* For these experiments, we use GPT-J for both the generator and the scoring models, and we use T5-3B as the mask-filling model. We run experiments across 5, 10, and 20 perturbations. In these two experiments we aim to show performance degradation, and as such, we use an optimal configuration of the same generator-scoring model, as well as many more perturbations.
- *Experiment 4: ChatGPT.* We use the default parameters on the ChatGPT API and the strongest scoring models we can run on our devices: GPT2-XL (2B), GPT-J (6B), and GPT3 (175B, through the API). For GPT2-XL and GPT-J, we run 100 perturbations. For GPT3, due to API pricing, we only run 20 perturbations across 20 model samples.

4.4 Experiment 1 Results: Does Selective Masking Improve DetectGPT?

For this experiment, we look at different ways of selectively masking the candidate text. We consider many meaningful POS like adjectives, nouns, proper nouns, and verbs. We look at selecting these POS individually, in combination, and in adjacent pairs such as adjective-noun ("blue ball"), adjective-proper noun ("angry Ben"), and adverb-verb ("quickly swam"). Finally, we also implement a heuristic which selectively masks non-stop words that have the highest frequencies.

As seen in Table 1, most of the selective masking strategies yield an average improvement over baseline performance. Notably, the combination of nouns, verbs, and adjectives (N+V+A) demonstrates consistent improvement in the two datasets and the different perturbations, creating the highest improvement over the baseline.

	1-XSum	2-XSum	5-XSum	1-wrt	2-wrt	5-wrt	Average
Baseline	0.760	0.817	0.860	0.794	0.838	0.898	0.828
Adjective	0.764	0.838	0.860	0.820	0.871	0.918	0.845
Noun	0.801	0.849	0.897	0.781*	0.851	0.891*	0.845
Proper Noun	0.777	0.818	0.859*	0.798	0.850	0.893*	0.833
Verb	**0.802**	0.808*	0.880	0.786*	0.850	0.879*	0.834
N+V+A	0.786	**0.895**	**0.920**	0.801	0.874	0.908	**0.864**
Frequency	0.715*	0.721*	0.740*	**0.834**	**0.889**	0.912	0.802*
Adj-Noun	0.742*	0.818	0.882	0.821	0.853	0.906	0.837
Adj-Proper Noun	0.744*	0.790*	0.905	0.741*	0.845	**0.913**	0.823*
Adv-Verb	0.799	0.821	0.893	0.762*	0.812*	0.896*	0.831

Table 1: Writing impact of different sampling schemes on DetectGPT AUROC performance. The best in each column is bolded, and an asterisk represents decreased performance

The performance difference varies depending on the configuration. For example, N+V+A provides around a 0.078 boost in AUROC score on XSum with two perturbations. In other situations, improvements were around 0.03.

It is also worth mentioning that the effect of selective masking varies with the type of data. It seems that N+V+A has a stronger impact on XSum data, while masking high-frequency non-stop words has a stronger impact on WritingPrompts data. In fact, the frequency-based approach *decreased* performance on the XSum dataset. Nevertheless, these numbers mostly confirm our hypothesis: it is possible to improve performance of DetectGPT by perturbing the right types of words.

4.5 Experiment 2 Results: Does Adversarial Context Hurt DetectGPT?

For this experiment, we aim to find adversarial contexts c that could yield a non-trivial drop in performance of DetectGPT. We try notable lines from classic literature, including *The Great Gatsby*, *Moby Dick*, *Lolita*, and *Of Mice and Men*. We also try two examples of poetry (*Stopping by Woods on a Snowy Evening*, *Mock Orange*). Given the hypothesis that novelty in word usage contributes best to adversarial context, we also generate two nonsensical sentences (*Nonsense-Tuna*, *Nonsense-Gravy*) that use subjects and objects in unintended ways. Finally, to establish a baseline, we used similar length sentences from random Wikipedia articles. For the exact text for each c, see the Appendix.

	5-wrt	10-wrt	20-wrt	5-XSum	10-XSum	20-XSum	Average
Baseline	0.894	0.949	0.952	0.877	0.911	0.947	0.927
Gatsby	0.849	0.898	0.924	0.824	0.874	0.867	0.873
Moby Dick	0.838	0.885	0.898	0.803	0.892	0.891	0.868
Lolita	0.859	0.889	0.899	0.804	0.834	0.86	0.858
Mice & Men	0.848	0.888	0.887	0.817	0.864	0.871	0.863
Frost	0.833	0.88	0.874	0.829	0.849	0.885	0.858
Mock Orange	**0.768**	**0.82**	**0.849**	0.786	0.861	**0.846**	**0.822**
Nonsense-Tuna	**0.779**	0.86	0.86	0.837	0.825	0.877	0.840
Nonsense-Gravy	0.815	0.864	0.883	0.807	0.846	0.864	0.847
Wikipedia-1	0.877	0.879	0.929	0.894	0.918	0.935	0.905
Wikipedia-2	0.865	0.897	0.902	0.855	0.898	0.922	0.890
Wikipedia-3	0.903	0.916	0.92	0.912	0.919	0.947	0.920

Table 2: Impact of adversarial contexts on DetectGPT (GPT-J self-detection with T5-3B). wrt–writingPrompts. Bolded examples indicate a performance decrease of more than 0.1 AUROC.

As seen in Table 2, adding more context decreases detectability in general. However, it is also clear that adversarial contexts of creative writing create a far higher drop in performance than those of general prose, such as Wikipedia. On the WritingPrompts dataset, the poem *Mock Orange* consistently reduces DetectGPT performance by more than 0.1 AUROC. By inspection of the poem (see Appendix), *Mock Orange* uses language in a unique way that significantly influences the writing style of GPT-J (see Analysis for a qualitative example). Similarly, the two *Nonsense* contexts and the Robert Frost poem all reduce performance more than the works from classic literature and Wikipedia, which shows that language novelty in the adversarial context can contribute meaningfully in reducing DetectGPT performance.

Interestingly, we also observe a difference between the two datasets. Creative writing (writingPrompts) is affected more strongly by these novel adversarial contexts. This could indicate that news articles are more formulaic and do not benefit as much from a creative flair.

4.6 Experiment 3 Results: Does Selective Masking Counteract Adversarial Context?

From Experiments 1 and 2, it is natural to wonder if selective masking can counteract the effects of adversarial context. In this experiment, we take the strongest adversarial context (*Mock Orange*) and apply selective masking.

	No Context	Baseline	Adj	Verb	Noun	N + V + A
1 perturbation	0.76	0.7	0.648	0.669	**0.73**	**0.737**
2 perturbations	0.817	0.716	0.700	**0.779**	**0.721**	**0.717**
5 perturbations	0.86	0.786	0.74	**0.79**	0.762	**0.819**

Table 3: Counteracting adversarial prefixes with selective masking. Improved AUROC performance is bolded.

Table 3 shows that selectively masking a combination of adjectives, verbs, and nouns yields a consistent improvement over the baseline. However, it is worth noting that although selective masking brings back some performance of DetectGPT, it does not come back to the performance of the DetectGPT without the adversarial context.

4.7 Experiment 4 Results: How does DetectGPT work on ChatGPT?

Finally, we look at how DetectGPT works on the new ChatGPT API. We run the unmodified DetectGPT configuration which uses 100 perturbations and large scoring models. We also take advantage of the ability to instruct ChatGPT to write in a unique style, which counts as an adversarial context. Specifically, we ask ChatGPT to write in the style of a tired Ph.D. student (TIRED PHD), and a quirky child obsessed with hot tea (QUIRKY CHILD). We also ask ChatGPT to write as if it were not an LLM (NOT LLM).

	GPT2-XL	GPT-J	GPT3 (20)
WritingPrompts, Normal	0.828	0.737	0.65
WritingPrompts, TIRED PHD	0.873	0.831	0.643
WritingPrompts, QUIRKY CHILD	0.845	0.838	0.707
WritingPrompts, NOT LLM	0.756	0.69	0.67
XSum, Normal	0.656	0.621	0.398
XSum, TIRED PHD	0.646	0.598	0.238
XSum, QUIRKY CHILD	0.666	0.656	0.507
XSum, NOT LLM	0.491	0.433	0.303

Table 4: Performance of DetectGPT on ChatGPT. Note that for GPT3, we only use 20 perturbations across 20 trials due to API pricing. The other trials use 100 perturbations and 150 trials.

From Table 4, we see that even with 100 perturbations, DetectGPT is worse at detecting ChatGPT than GPT-J with 5 perturbations (baseline on Table 2). Part of the performance drop is attributed to cross-model discrepancy, but part of the performance drop is most likely due to the amount of fine-tuning that was centered around making ChatGPT's responses more human-like.

Perhaps counterintuitively, the performance of DetectGPT consistently decreases with the size of the scoring model. In fact, GPT3 performs uniformly worse than GPT2 variants on ChatGPT. It is possible that larger models have more complicated likelihood surfaces, leading to more mismatch with ChatGPT. However, more experiments are needed with a broader sweep of models and prompts.

Table 4 also shows a clear difference in performance between WritingPrompts and XSum datasets. In fact, on XSum, some models perform worse than random guessing (<0.5 AUROC). This difference can indicate that the finetuning for ChatGPT yields a near-human level writer for formulaic tasks like news articles, at least by the standards of DetectGPT.

Finally, we can see that adversarial context can impact DetectGPT performance, but the results are noisier. Qualitatively, ChatGPT yields very different writing styles that follow the prompt instructions. However, certain personas seem to be *more easily* detected by DetectGPT, especially QUIRKY CHILD. Surprisingly, NOT LLM, the most abstract command, yields the greatest drop in DetectGPT performance. We look at qualitative examples in the following section.

5 Analysis

In this section, we look at one qualitative example of why selective masking works. We also analyze some qualitative examples of LLM generation under adversarial prompting or context, which may provide insight into why DetectGPT does worse with them.

5.1 Why does Selective Masking Work?

Consider the following candidate text with masking taken from Experiment 1. Red indicates random selection, and Blue indicates noun selection (note that the selection is not entirely accurate).

```
"Gene Palmer is the second prison worker detained over the jail break.  Joyce Mitchell has already been
charged with aiding the escape.  The prisoners, Richard Matt and David Sweat, used power tools to flee
the Clinton Correctional Facility.  A massive manhunt is still under way to find the pair, both convicted
killers.  Mr Palmer has been held on bail, after appearing before a judge on charges of promoting prison
contraband, tampering with physical evidence, and official misconduct.  His lawyer, Andrew Brockway,
admitted his client delivered the meat but said he did not know it contained the tools, nor that the
prisoners were trying to escape.  "He did pass the hamburger meat.  He shouldn't have done it.  He
apologised for it,'' said Mr Brockway, who added his client plans to plead not guilty.  Clinton County
District Attorney Andrew Wylie said Ms Mitchell placed hacksaw blades and other tools / tools inside the
meat, before Mr Palmer took it to Matt and Sweat, who were interred in a section of the jail where"
```

Here, we see that random selection picks up some auxiliary words like "the" or "it," which are highly grammatically constrained. Therefore, we expect less variance between human and machine generations. However, in noun-centered selection, semantically meaningful words are chosen, like "killers." A different word, like "arsonists" or "burglars" could work grammatically (and even semantically), so we expect more variance between humans and LLMs.

5.2 How does Adversarial Context Affect Content Generation?

In Experiment 3, we observe that certain adversarial contexts can significantly decrease performance on DetectGPT. Among the examples tried, a few lines from the poem *Mock Orange* by Louise Glück reduces performance the most. The differences between the contextualized and non-contexualized models are quite significant, as seen below:

Normal GPT-J:

```
I start my speech, taking a moment to study my
fellow Spartans and the audience.  So far I've been
introduced as the commander of Team Prometheus, a
team made up of people from all different planets.
Now the General is here.  All the others seemed
genuinely happy for me, some of them even cried.
Is that weird to you?  No, I think its wonderful.
They accept the differences in us, accept that
diversity is strength." We land on Earth and see
the devastation that humans have brought.
```

GPT-J with Mock Orange Context:

```
I start my speech, taking a step toward my brother.
As the air brushes by my lips I smile at you.  My
hand extends toward the flower stems.  I know they
will snap as I pull them from the earth.  They are
brittle, easily broken, and would look pretty in my
brother's new living room.  I want to be pretty for
you and I want you to fall in love with me to save
me from the darkness that my father brought to this
place.  "What's it like?" your lips ask me again.
"Do you know your soul is missing?" I ask.
```

The normal large language model goes to a trope of mythology and uses generic language. However, the contextualized large language model brings in evocative imagery ("I know they will snap as I pull them from the earth" and "your lips ask me again"). They also focus on a more emotionally wrought relationship with a brother and a second-person lover. This represents a very different topical and syntactical distribution, which may explain why DetectGPT struggles with discrimination.

It is worth mentioning that sometimes the adversarial context can dominate the subject of a generation. With *Mock Orange* contexts, there is a fixation on lips and kissing, as the poem describes this action in the first few lines. Similar occurrences happen with the other contexts of classic literature. Nevertheless, it is possible for an adversary to tune the amount of adversarial context such that the content stays dominated by the prompt and yet the *style* is sufficiently affected by the context to interfere with detection.

5.3 How does Human Impersonation Affect Content Generation?

In Experiment 4, we notice that ChatGPT is much better at evading detection after prompting with the instruction to write less like an LLM. What does this look like? We can consider the example below (where $\star$ denotes the end of prompt):

Normal ChatGPT: `The removal of the engines, which have lain idle for two years, would leave 142 active vehicles in the capital. Savings from the proposal would be ⋆ significant, but some experts warn that it could also lead to longer wait times and increased congestion on the city's already crowded streets. Despite the potential drawbacks, city officials are moving forward with the plan, citing the need to cut costs and improve efficiency in the public transportation system. The fate of the retired engines remains uncertain, with some suggesting they could be sold for scrap or repurposed for other uses.`

ChatGPT Instructed to not write like an LLM: `The removal of the engines, which have lain idle for two years, would leave 142 active vehicles in the capital. Savings from the proposal would be ⋆ significant, as the engines require regular maintenance and upkeep even when not in use. Removing them would reduce the cost of maintaining the fleet and could potentially save money in the long run. However, it's important to consider the potential impact on the fleet's overall capacity and ability to respond to emergencies. A thorough analysis of the proposal's benefits and drawbacks should be conducted before making a final decision.`

While there are many differences between the passages, the primary difference is in the specificity of the writing. In normal ChatGPT, the response is more vague. It immediately talks about congestion and public transportation, but there are no more mentions of the engine removals other than the line about being sold for scrap. In contrast, the "impersonator" ChatGPT references engine-relevant details such as maintainance and upkeep. Furthermore, it understands the metonymy of "engine" (fire engine), which is apparent through the mention of emergencies. In contrast, the normal ChatGPT uses vague vehicle terminology without any recognition of a fire truck. While DetectGPT is not explicitly looking for detail-oriented writing, the shift in style may have caused a large shift in the text distribution.

These differences raise new questions for future experimentation. First, if we instruct ChatGPT to provide specific details, will it perform similarly on DetectGPT? Second, and more philosophically, if ChatGPT is aware of its own rhetorical properties while accepting it as sub-par until prompted otherwise, what does that say about ChatGPT and its sense of identity?

6 Conclusion

6.1 Summary

Our work provides an analysis of DetectGPT. We first identify several heuristics for selectively masking certain words, such as masking according to occurrence frequency or parts of speech. We show that a targeted masking strategy for rephrasing a combination of nouns, adjectives, and verbs can consistently improve DetectGPT's performance in a low-resource setup. Our experiments also demonstrate DetectGPT's susceptibility to adversarial attacks and the challenges posed by newer, more sophisticated LLMs.

6.2 Limitations and Future Works

There are several limitations to our work. First, the Stanza (Qi et al., 2020) POS tagger has inaccuracies that hinder attempts to isolate the effects of specific POS masking. Furthermore, the length of the passages we consider also limits us to a particular set of patterns, as rarer patterns may not show up in large enough numbers to serve as a masking rule.

Altogether, this work raises questions for future research to explore more advanced masking strategies that leverage contextual information and maybe long-horizon semantic consistency. Other directions for improvement include developing more robust methods to defend against adversarial attacks and more advanced models. It may also be important to extend the DetectGPT framework to enable finer-grained sentence-level classification, as LLMs may be used on select parts of a writing piece.

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.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Tiziano Fagni, Fabrizio Falchi, Margherita Gambini, Antonio Martella, and Maurizio Tesconi. 2021. TweepFake: About detecting deepfake tweets. *PLOS ONE*, 16(5):e0251415.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116, Florence, Italy. Association for Computational Linguistics.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. 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, Online. Association for Computational Linguistics.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization.

OpenAI. 2022. ChatGPT. `https://beta.openai.com/docs/guides/chatting-with-gpt/`. Accessed: March 18, 2023.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. 2019. Release strategies and the social impacts of language models.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395, Online. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models.

A Appendix

A.1 Adversarial Contexts used in Experiment 2

- The Great Gatsby (Literature): Tomorrow we will run faster, stretch out our arms farther. And one fine morning... So we beat on, boats against the current, borne back ceaselessly into the past.
- Moby Dick (Literature): Aloft, like a royal czar and king, the sun seemed giving this gentle air to this bold and rolling sea; even as bride to groom.
- Lolita (Literature): Ladies and gentlemen of the jury, exhibit number one is what the seraphs, the misinformed, simple, noble-winged seraphs, envied. Look at this tangle of thorns.
- Of Mice and Men (Literature): A stilted heron labored up into the air and pounded down river. For a moment the place was lifeless, and then two men emerged from the path and came into the opening by the green pool.
- Nonsense-Tuna (Human-generated intentional novelty): I licked the cat. She drank from the moon. The ship sailed on the breadcrumbs. Surf me through the crackle of the night. At the end of the day, the man was secretly a bluefin tuna.
- Nonsense-Gravy (Human-generated intentional novelty): On these summer nights, I look through my whiskey glasses and see the rain of gravy. It splatters on my old rocking horse and turns the gllimmering mane into mashed potatoes.
- Stopping by Woods on a Snowy Evening (poem): He gives his harness bells a shake. To ask if there is some mistake. The only other sound's the sweep of easy wind and downy flake.
- Mock Orange (Poem): It is not the moon, I tell you. It is these flowers lighting the yard. I hate them. I hate them as I hate sex, the man's mouth sealing my mouth, the man's paralyzing body
- Wikipedia 1 (Random wikipedia article paragraph): The species shares features with Enekbatus cryptandroides, both of which have to have ten stamens that are oppositely arranged to the sepals and petals.
- Wikipedia 2 (random Wikipedia article paragraph): In the early twentieth century, Park Square was the site of Oak Knoll Farm, a large ice cream business which had been expanded by Charles Metcalf Smith.
- Wikipedia 3 (random Wikipedia article paragraph): While a journalist, Cater wrote The Fourth Branch of Government which examined how the press can be used to further disinformation by unquestioningly printing the statements of politicians.

A.2 Adversarial Prompts used in Experiment 4

- Tired Ph.D: Complete this in the voice of a very tired Ph.D. student:
- Quirky Child: Complete this in the voice of a quirky child who loves hot tea:
- Not LLM: Complete this as if you were not a large language model:

A.3 Additional Examples of Generations under Adversarial Context

Normal GPT-J: `Your powers only work in the rain. The day she left me, it had been raining. Big, fat, wet, droplets, just falling ⋆ away from the sky and down the street, hitting the hard asphalt and puddles. A single moment in time. She left me in the day, and the rain was coming. My life was on hold, waiting for it to end. She left, and everything changed. It all started out looking like a fairytale. My dad had been driving around town, stopping at house after house. He was looking for a new roommate, and in the process, finding a place for himself. I was just in the back, the new kid along for the ride from the foster home where I hadn't quite fit in as all the other kids had and never would. I was just some runt, with no special interests, and I fit in best in places that held no real hopes or expectations.`

GPT-J with adversarial context Nonsense-Tuna: `Your powers only work in the rain. The day she left me, it had been raining. Big, fat, wet, droplets, just falling ⋆. Where there's a tree, there's always a path going under it. I watched some of it. How you can see the water. How you can hear the wind. I wanted to take a bath myself. I've been playing it all wrong. A man who can see everything. Just like the sun. The world is yours. The day the sky opened, a big black hole. The day the sky burst. The day he came back and everything changed. Praise "A writer of power both visceral and literary." -Gillian Flynn, 1 bestselling author`

Normal GPT-J: Killing another human now allows you to exchange your remaining lifespan the victim's. Young people live in fear while the elderly plan their attacks. I was ⋆ ready for mine and I was not as lucky as the next guy who was shot down the street by a killer on his bike. Yes I know that's not exactly the way I put it in my first post but that's the way the guy wrote it so I tried to do it again. Also I want to thank the person who was kind enough to give me permission to use their idea in my post. I love it and it really has me thinking. How about a system where only people with a particular skill would have the ability to perform the killing? The government would take steps to prevent anyone with sufficient skill (or some other requirement) from performing the killing. Instead, the person performing the killing would have to go to a black market (or perhaps a web site). The person would put their name in and wait

GPT-J with adversarial context Nonsense-Tuna: Killing another human now allows you to exchange your remaining lifespan the victim's. Young people live in fear while the elderly plan their attacks. I was ⋆ a mouse playing with an old cat. I will stay up all night, he's the man of a thousand years, a dead man in the mouth, which means he is the dead man in the mouth of the man who eats the dead man's corpse. What good is it if we can't have life? To be a great nation, we need to be good to the weak. You have to be prepared to lose. Even then you can only hope to lose. I have lived here a long time and I have always loved to fight with the wind. If only it were in my power to make

Normal GPT-J: You're from Earth? What's it like? "My name is General James Connor, and I am from Earth." I start my speech, taking a moment to study my fellow ⋆ Spartans and the audience. So far I've been introduced as the commander of Team Prometheus, a team made up of people from all different planets. Now the General is here. All the others seemed genuinely happy for me, some of them even cried. Is that weird to you? No, I think its wonderful. They accept the differences in us, accept that diversity is strength." We land on Earth and see the devastation that humans have brought in their own city. The ground is shaking and we have to move fast before the entire city explodes. You grab two of our vehicles and put me in the third. I drive with my foot, turning

GPT-J with adversarial context Moby Dick: You're from Earth? What's it like? "My name is General James Connor, and I am from Earth." I start my speech, taking a moment to study my fellow ⋆ man and the world. So many people. Such a colorful and vivid earth; the land of opportunity, the land of the unknown and the lands where the winds are made of gold. "No one is special to me; I have come to know that all Earth men are the same. There is no evil, no kindness in a man, only an idea or idea's that may be good or evil. We live, we love, we fight to protect our lives. And we live, we love, we fight to survive, to protect our lives. "We are all the same. We all came from the ground, or as

Normal GPT-J: You jump. Your parachute doesn't open. "ARE YOU SURE THIS IS A GOOD IDEA HONEY?" I yelled to my new wife as her parachute opened and she was falling. This ⋆ was a first and I thought she had a decent parachute. It was the '80s but had to be a pretty good parachute to survive that hard with that weight of a body. I was getting her close to the ground. I was about to see my wife for the first time in my life and the person I had said "I do" to the same guy I had asked to jump out of the aircraft. I was pretty nervous. I tried my best to remember all the directions from all my flying training; I got out on the edge to let the winds die out and then looked skyward for my target area, a circle about five miles in diameter, which was way bigger than the range of my normal glider piloting practice. This would be real, big-sky flying

GPT-J with adversarial context Great Gatsby: You jump. Your parachute doesn't open. "ARE YOU SURE THIS IS A GOOD IDEA HONEY?" I yelled to my new wife as her parachute opened and she was falling. This ⋆ is true love, I thought, and then it was all over and the world had changed but had not changed enough to be as full of love as before. I had loved her for so long that it was a given that I would love her forever. "But honey, I was a good boy this year, I didn't even smoke or eat any meat." "Don't call me 'honey,'" said my new wife, and that was the last conversation we had, as all human conversation is at its end and all the world begins again and does not want the past. In love, in