Analyzing Keyword Indicators of LLM-Generated Content in an Online Niche Community

Matthew Mattei[1]

[1]Stanford University

Professor Jerry Cain

Abstract—*Detecting AI-generated content presents challenges, particularly at scale, where processing large volumes of data is necessary. Many small online communities lack the resources for comprehensive solutions, yet remain vulnerable to AI-generated scams and advertisements. However, the limited textual contribution of these communities to large language models (LLMs) creates a unique distinction from LLM-generated text, offering potential for cost-effective detection solutions tailored to smaller datasets. This suggests the possibility of developing efficient AI-detection methods that adequately moderate and safeguard these communities.*

Keywords—*AI-detection, LLM, communities*

1. Introduction

1.1. Background

AI Detection is a hard problem. To find differences between AI-generated text from the highest-performing models and most human-written text, one needs to process hundreds of gigabytes of data, perform complex analysis on the texts' writing styles, compare the probabilities certain words appear, and more. Notably, these forms of analysis and processing are beyond the abilities and scope of the sorts of groups in the greatest need of AI detection software: the many small communities that populate the internet.

1.2. Problem Statement

Small internet communities range from TV show wikis, to Animal Jam game chat boards, to Silver Surfer forums. These communities live and die by the engagement their community members have with each other. If generative AI is used to disseminate spam (advertisements, scams, etc.) across such community platforms, the communities themselves could disappear as members tire of AI-generated content and leave. However, members merely leaving is one of the better outcomes. Many small internet communities, such as the latter two I mentioned, are populated by more vulnerable members of the internet, such as children and senior citizens. These vulnerable populations are less likely to recognize AI-generated content themselves and, thus, more likely to fall susceptible to bad actors.

1.3. Research Gap

Despite the increasing statistical similarity between AI-generated and human-written text, there remains a critical gap in literature regarding effective AI detection strategies meant for smaller datasets and simpler analytical processes. Traditional methods may be inadequate, due to their significant time and resource costs, for addressing the unique challenges faced by small online communities, necessitating new approaches that leverage the distinctive characteristics of these platforms.

1.4. Project Objective

This project aims to conduct an analysis of common words present in both community-written text and LLM-generated text designed to replicate community-style writing. The objective is to determine if certain words appear in significantly different rates between these two categories. Specifically, we aim to explore whether the rate of appearance for specific words can effectively signal the presence of LLM-generated text when only community-written text and a comparably sized corpus of LLM-generated text are available for analysis.

By focusing on this objective, the project aims to develop a deeper understanding of the potential of word appearance rate analysis as a tool for distinguishing between human-written and LLM-generated content in certain contexts in the hopes of contributing insight to enhance the detection and mitigation of AI-generated spam, scams, and misinformation.

2. Methodology

2.1. Data Collection and Preparation

1. Extract a substantial portion of written text from a small internet community. Convert the extracted text into two distinct data formats: one representing all words in response posts (post responding to other posts in the forum) and another describing the contents of each response post.

2. Utilize an LLM to generate a comparable volume of text based on community topics. Convert the generated text into two data formats akin to the previous community-written data.

2.2. Word Frequency Analysis

1. Identify the ten most common words in the community-written text once for each of the following three conditions: considering all English words, excluding the top 500 most common English words, and excluding the top 1000 most common English words. By excluding many common English words for some of the comparisons, a broader collection of words used in the community can be considered and analyzed even if they appear less frequently than the most common English words.

2. Determine the frequency of occurrence of these words in each community-written response post and each LLM-generated response.

2.3. Bootstrap Analysis

1. Conduct bootstrap analysis on each post and response for the three sets of ten words.

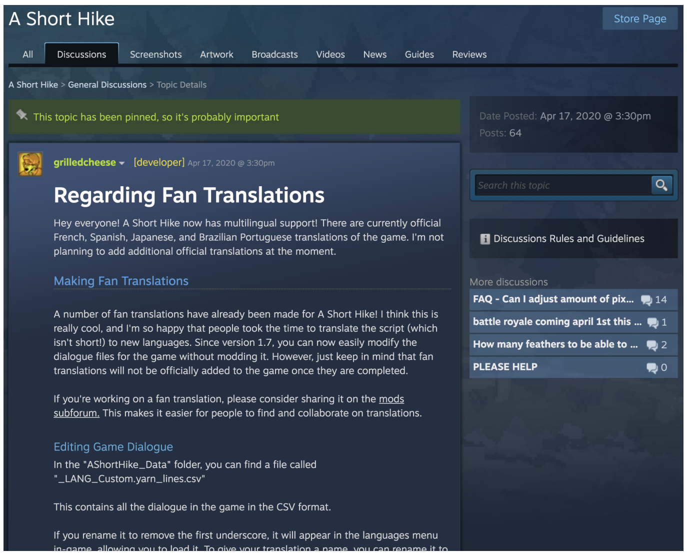
2. Calculate the observed difference in the average number of occurrences of each word per response in community-written responses versus LLM-generated responses. According to the null hypothesis, it's possible that an observed difference in means arises purely from random chance. As such, the corresponding p-value for each observed difference will also be calculated through bootstrapping to determine which words appear in statistically significantly different rates.

2.4. Decisions and Assumptions

- This project will use the Steam Discussion Forum for "A Short Hike" (a video game, see Figure 1) as an experimental representative for a small internet community. The forum is defined by a relatively small number of threads (only around 283 as of the time of this project's completion), a still fairly active user base (last thread was started on March 12th), and primarily user-written posts (little to no bot-written posts or replies to affect the analysis).

- This project will use Google's Gemini 1.5 Pro model to produce LLM-generated text since it is one of the most advanced LLM models publicly available, with a free API for content generation, making it a more likely potential target for spam-related content production at the time of this project's completion.

- This project will use the "English Word Frequency" dataset on Kaggle, uploaded by Rachael Tatman, for the purposes of determining the most common English words.


A Short Hike
Store Page
All | Discussions | Screenshots | Artwork | Broadcasts | Videos | News | Guides | Reviews
A Short Hike > General Discussions > Topic Details

This topic has been pinned, so it's probably important

grilledcheese [developer] Apr 17, 2020 @ 3:30pm

Regarding Fan Translations

Hey everyone! A Short Hike now has multilingual support! There are currently official French, Spanish, Japanese, and Brazilian Portuguese translations of the game. I'm not planning to add additional official translations at the moment.

Making Fan Translations

A number of fan translations have already been made for A Short Hike! I think this is really cool, and I'm so happy that people took the time to translate the script (which isn't short!) to new languages. Since version 1.7, you can now easily modify the dialogue files for the game without modding it. However, just keep in mind that fan translations will not be officially added to the game once they are completed.

If you're working on a fan translation, please consider sharing it on the mods subforum. This makes it easier for people to find and collaborate on translations.

Editing Game Dialogue

In the "AShortHike_Data" folder, you can find a file called "_LANG_Custom.yarn_lines.csv"

This contains all the dialogue in the game in the CSV format.

If you rename it to remove the first underscore, it will appear in the languages menu in-game, allowing you to load it. To give your translation a name, you can rename it to

Date Posted: Apr 17, 2020 @ 3:30pm
Posts: 64

Search this topic

Discussions Rules and Guidelines

More discussions
FAQ - Can I adjust amount of pix... 14
battle royale coming april 1st this ... 1
How many feathers to be able to ... 2
PLEASE HELP 0

Figure 1. *"A Short Hike" Discussion Thread*

3. Results

Github containing project code and produced data

3.1. Data Collection and Preparation

`words.csv` was produced by functions in `steam_discussion_scraper.py` to contain nearly all words in community-written responses to posts in the "A Short Hike" discussion forum (excludes quotes and special characters). Additionally, `op.csv` was produced by `steam_discussion_scraper.py` to contain community-written posts in the "A Short Hike" discussion forum in the format "Topic Title", "Topic Content" (excludes posts with no community-written reply) for the purposes of generating LLM responses. Finally, `llm.csv` was produced by functions in `generate_llm_responses.py` to contain the words in LLM-generated responses to nearly all community-written posts in the "A Short Hike" discussion forum (excludes responses to posts that had no community-written replies, a signal for potential junk posts, as well as posts Gemini failed to generate a response to for any reason).

3.2. Word Frequency Analysis

`0dump.json`, `1dump.json`, and `2dump.json` each contain a list of dictionaries containing counts of the words present in each community-written post, a list of dictionaries containing counts of the words present in each LLM-generated response, and a dictionary containing the 10 most common words in the community-written posts that also appear in at least one LLM-generated response.

Each of the three files has had its content lightly filtered, with contractions and most non-English words being removed from responses. Additionally, certain words from the community-written posts and LLM-generated responses have been discarded from consideration as the 10 most common words:

- `0dump.json` allows all English words to be considered among the 10 most common words.

- `1dump.json` discards the 500 most common English words from consideration.

- `2dump.json` removes the 1000 most common English words from consideration.

Subsequently, each JSON file is processed to remove all words, and their counts, that aren't included in the file's respective most-common-word dictionary from every community-written post and LLM-generated response, producing `second_0dump.json`, `second_1dump.json`, and `second_2dump.json`.

3.3. Bootstrap Analysis

For each set of the 10 most common words, all community-written posts and LLM-generated responses were processed using the following code for each word from the set:

```python
def calculate_p_value(posts: list, llm_responses: list, word: str):
    """
    Calculates the p-value for the for the observed difference in
    the mean number of times the given word appears across all posts and all llm responses.
    """
    post_nums = [post.get(word, 0) for post in posts]
    llm_response_nums = [response.get(word, 0) for response in llm_responses]
    n = len(post_nums)
    m = len(llm_response_nums)
    observed_diff = abs((sum(post_nums)/n - sum(
llm_response_nums)/m))
    uni_sample = post_nums + llm_response_nums
    count = 0

    for x in range(10000):
        a_resample = [uni_sample[random.randint
(0, len(uni_sample) - 1)] for i in range(n)]
        b_resample = [uni_sample[random.randint
(0, len(uni_sample) - 1)] for i in range(m)]
        mua = sum(a_resample)/n
        mub = sum(b_resample)/m
        dif = abs(mua - mub)
        if dif >= observed_diff:
            count += 1
    return [abs(sum(post_nums)/n - sum(
llm_response_nums)/m), float(count / 10000)]
```

Code 1. *P-value calculation code*

The observed differences in the average number of times the given word appears per community-written post and per LLM-generated post, as well as the associated p-values for each word, were then stored in a combined dictionary. If a word appeared multiple times across the three 10-most-common-word-sets, only the observed difference in means and p-value from the last calculation were saved (since each calculation for the same word would be roughly equivalent if slightly different). The resultant dictionary values are as found in Table 1. From these values, the words that appear in both community-written text and LLM-generated text but with a statistically significant observed difference in average rate of appearance, having p-values that successfully disproved the null hypothesis ($p < 0.05$), are in Table 2.

Table 1. *Observed differences in means and p-values for found words*

Word	Observed Difference in Means	P-value
from	0.02896	0.3282
that	0.87215	0.0
and	1.51214	0.0
as	0.18321	0.0
the	3.70389	0.0
with	0.48113	0.0
to	2.58141	0.0
there	0.07410	0.0231
this	0.26858	0.0
of	0.93878	0.0
feathers	0.01890	0.3539
feather	0.00422	0.8158
everything	0.01681	0.1517
once	0.01681	0.1652
might	0.09865	0.0
switch	0.02080	0.1589
things	0.07149	0.0002
thank	0.04917	0.0047
thanks	0.06807	0.0018
played	0.00525	0.7280
translation	0.02669	0.0919
trying	0.05311	0.0006
actually	0.02595	0.0578
hope	0.16869	0.0
boat	0.02208	0.1479

Table 2. *Observed differences in means and p-values for found words with p-value < 0.05*

Word	Observed Difference in Means	P-value
that	0.87215	0.0
and	1.51214	0.0
as	0.18321	0.0
the	3.70389	0.0
with	0.48113	0.0
to	2.58141	0.0
there	0.07410	0.0231
things	0.07149	0.0002
thank	0.04917	0.0047
thanks	0.06807	0.0018
trying	0.05311	0.0006
hope	0.16869	0.0

4. Conclusion

As can be seen in Table 2, for the conducted experiment, there were a number of words that appeared at statistically significantly different rates between community-written responses and LLM-generated responses. Since the community-written post dataset effectively managed to sample the entire community, the difference in rates of word appearance between the LLM-generated content and the community-written content suggests different usages of key words based on the environment developed by the community. For example, in the "A Short Hike" Steam Discussion forums, many users rarely use the word "the" due to the casual nature of the forum, preferring quick, short responses between users. In contrast, the Gemini 1.5 Pro model tends to speak fairly formally, as one might expect of an approximation of all written English language, resulting in different rates of the use of the word "the". As a result, the different rates of appearances of the found words could act as tools for moderators to analyze posts and be more informed when determining whether or not a post was written by an LLM. When generalized to internet communities at large, the results of this experiment imply that many communities, especially on the smaller end, are able to take advantage of the energy and environment their community has cultivated. These results were produced using a fairly simple bootstrapping algorithm and datasets with sizes measured in kilobytes rather than gigabytes. Community members could easily replicate this strategy to determine localized key words that differ in rate of appearance between their community's texts and a few LLMs to improve or fix their moderation strategies. Thus, vulnerable members of these communities can be made safer and the community as a whole more vibrant.