LLM - Detect AI Generated Text

D11725003 林聖傑
R12944052 丁浩軒
R11249007 楊佳悅
R12922042 湯士弘
R12943016 謝言鼎

GitHub 專案連結、展示影片連結

Abstract

In the rapid development of artificial intelligence, particularly in natural language generation, the emergence of Large Language Models (LLMs) like OpenAI's ChatGPT signifies remarkable progress. These models exhibit remarkable capabilities, ranging from practical tasks like email composition and coding to creative endeavors such as poetry and narrative generation. However, this proficiency introduces challenges related to authenticity, ethical use, and so on. The risk of misuse includes academic cheating, fake news generation, and fraudulent product reviews, highlighting the pressing need for detecting AI-generated text. The generalization ability of LLMs makes the detection more difficult to keep pace with their advancements. In this study, we employ three methods, including refining BERT and its variants for contextual classification,using the QLoRA methodology for fine-tuning Large Language Models (LLMs), and exploring traditional machine learning techniques to address the challenge of detecting AI-generated text. In the end, we compare the performance of the three methods to determine the most suitable approach for detecting AI-generated text.

1 Introduction

In the 21st century, artificial intelligence (AI), particularly in natural language generation (NLG), has experienced remarkable growth. The advent of Large Language Models (LLMs) like OpenAI's ChatGPT has been a testament to this progress, showing extraordinary capabilities in churning out text that is not just coherent but also nuanced and contextually relevant. Their applications are vast, ranging from composing emails, essays, and codes to more creative tasks like writing poetry or generating narrative content. However, this burgeoning capability of AI to produce human-like texts at high efficiency also introduces significant challenges, particularly in the realms of authenticity, ethical use, and misuse

The potential for the misuse of these AI technologies is vast and varied. Academic institutions are grappling with the potential for cheating and plagiarism, as students might use these tools to generate essays or homework assignments. The media and journalism sectors face the challenge of fake news generation, where AI could be used to churn out convincing but entirely fabricated news stories. In the business world, there's the risk of fraudulent product reviews or AI-generated spam that can skew consumer perception and disrupt market dynamics.

Detecting AI-generated text, therefore, becomes not just a technical challenge but a pressing societal need. The complexity of this task is compounded by the continuously evolving capabilities of LLMs. As these models become more sophisticated, the line between human and machine-generated text blurs, making detection increasingly challenging. This situation calls for a dynamic, multi-faceted approach to detection, one that evolves in tandem with the AI models it seeks to scrutinize.

One of the significant challenges in detecting AI-generated text is the rapid advancement in the quality of text produced by these models. As AI models become more adept at mimicking human writing styles, traditional detection methods continually face new tests. The evolving nature of these models necessitates adaptive and innovative detection strategies that can keep pace with AI's advancements.

In this study, we employ a three-pronged approach to address the challenge of detecting AI-generated text. Our first strategy adopts a transformer-based approach, where we refine BERT and its variants for contextual classification tasks. The second strategy is centered around Large Language Models (LLMs), utilizing the QLoRA methodology to fine-tune LLMs for the specific task of classifying whether a given text is generated by an LLM. The third and final strategy explores traditional machine learning techniques, incorporating methods like TF-IDF and GloVe.

The structure of the remainder of our research is as follows: Section 2 presents an overview of the dataset utilized in our study, along with the computing metrics employed. Sections 3 delve into detailed introductions of the three methodologies, respectively. This is followed by an exposition of our experimental settings and the results obtained. Finally, Section 4 provides a conclusion of the report, summarizing our findings and their implications.

2 Overview

In this project, we participate in a Kaggle competition and utilize the provided dataset for training and validating our methodologies. The dataset consists of approximately 10,000 essays. These essays are a mix of student-written pieces and texts generated by various Large Language Models (LLMs). Each essay is a response to one of seven predefined prompts. In the competition, essays from two of the prompts compose the training set, while the remaining five prompts constitute the hidden test set. The dataset is structured into three key components: the essay title, the main body of the text, and a label indicating whether the essay was generated by an LLM.

Upon a preliminary examination of the training dataset, we discovered a significant imbalance in the labeling, as detailed in Table 2. The dataset contains 1,375 student-written essays, in stark contrast to a mere 3 essays generated by LLMs. To address this discrepancy and augment the scarcity of LLM-generated data, we incorporated the Daigt V2 dataset [3] as a supplementary resource.

AUROC (the Area Under the Receiver Operating Characteristic curve) is a robust and versatile metric for evaluating binary classification models, especially useful in cases of imbalanced datasets and when an intuitive, comprehensive measure of model performance is required. Given that the task at our hand is a binary classification problem, we utilize AUROC not only to compute training loss but also to validate model performance during the inference stage, ensuring a thorough and reliable evaluation of our models.

	Human-authored Essays	**LLM-generated Essays**
original training dataset	1,375	3
Daigt V2	27,371	17,497

Table 1: Comparison of essays in original training dataset and Daigt V2

3 Methods

3.1 Pre-trained model

We used four pre-trained models bert-base-cased、bert-base-uncased、distilbert、RoBERTa , aiming to determine which model exhibited the best performance.

bert-base-cased is the fundamental version of BERT that preserves the case information of words. The model can differentiate between uppercase and lowercase letters, treating "Apple" and "apple" as distinct vocabulary. We tried to use ‘bert-base-uncased’ to reduce the vocabulary size and computational load, since the primary difference between bert-base-cased and bert-base-uncased lies in the casing of the input text. For example, "Apple" and "apple" would be treated as the same token if I choose ‘bert-base-uncased’.

RoBERTa enhances BERT by using a larger training dataset and removing Next Sentence Prediction (NSP). During training, RoBERTa employs longer sequences and dynamic masking and adopts a single target, predicting the mask for each word in a sentence to enhance the model performance.

DistilBERT is a simplified version of BERT. DistilBERT keeps a similar architecture and reduces the model size by decreasing the dimensions of embedding layers and the number of Transformer layers. Compared to BERT, DistilBERT demonstrates impressive performance in some applications while having a smaller model size, making it more helpful in resource-constrained conditions.

Experiment settings:

Training configs

- batch size = 32
- lr = 2e-5
- epoch = 4
- max_lengh = 512
- optimizer : AdamW

Tokenizer : BertTokenizer

The BERT tokenizer utilizes the WordPiece method to segment words into subwords, similar to Byte Pair Encoding (BPE). In this process, the tokenizer compresses data based on the most frequently occurring byte pairs in the input string. The subwords replace the identified byte pairs, and the iteration continues by considering the next most frequent byte pair. The key distinction between WordPiece and BPE lies in the approach to subword selection: WordPiece selects subwords by maximizing the probability, whereas BPE selects subwords based on frequency.

Results:

Model	**Public Score**
bert-base-uncased	0.5
RoBERTa	0.754
bert-base-cased	0.782
distilbert	0.792

The performance result with 'bert-base-uncased' was inferior compared to the 'bert-base-cased'. Due to its retention of case information, bert-base-cased may be more suitable for the task. In summary, when comparing only bert-base-cased, RoBERTa, and DistilBERT, although these three models differ in architecture, their performance and training times are quite similar for the task of detecting AI-generated text. The highest scores achieved by DistilBERT may also imply that, for this task, using simpler models could yield relatively better performance.

3.2 LLM models

In addition to text generation, Large Language Models (LLMs) can perform a multitude of tasks, including classification, which is pertinent to our project. We employed two open-source models, Llama2-7b and Mistral-7b, combined with QLoRA techniques, to tackle this challenge.

Llama 2 is an advanced series of large language models (LLMs) ranging from 7 to 70 billion parameters, designed for complex natural language processing and dialogue tasks. A key innovation in Llama 2 is the grouped-query attention mechanism, enhancing scalability and performance.

Safety and ethical considerations are central to its design, incorporating measures like supervised fine-tuning and reinforcement learning with human feedback.

Mistral 7B, the latest in the open-source LLM domain, distinguishes itself through the incorporation of grouped-query attention (GQA) and sliding window attention (SWA). These mechanisms notably improve its proficiency in managing lengthy sequences and hasten inference. Mistral 7B shows exceptional capabilities in complex tasks such as reasoning, mathematics, and code generation, surpassing the performance of Llama2 13B in these specific areas.

Experiment settings:

QLoRA	**Training configs**
● r = 64 ● α = 16 ● dropout = 0.1 ● target modules = (q_proj, v_proj) ● quant type = nf4 ● use double quant	● batch size = 1 ● gradient accumulation = 16 ● lr = 5e-5 ● epoch = 1

Results:

model	**Public Score (AUROC)**
Llama2 7b	0.701
Mistral 7b	0.775

In our experiment, Mistral 7b exhibited superior performance over its counterpart llama2, both being models of similar size. However, it's noteworthy that despite this relative advantage, Mistral 7b did not surpass the scores achieved by BERT models. Moreover, the training time for Mistral 7b was notably longer than that for BERT. This observation suggests that, in the context of this specific task, Large Language Models like Mistral 7b may not be as effective as anticipated.

3.3 Machine Learning

Further Analysis of training dataset

After observing suboptimal performance from BERT-based models and LLMs, we conducted an in-depth analysis of the datasets, seeking alternative approaches to the task. The findings were enlightening.

Student essays and LLM-Generated texts diverge significantly in length, complexity, and lexical patterns. Students' essays tend to be longer, more varied vocabulary, while LLM essays are more concise, shown in Table 1 and Figure 2 [2].

Despite differences, both of them align closely with prompts, like **electoral college** and **presidential election**. Keyword frequency also highlights prompt-driven language in student essays.

	Student-authored Essays	**LLM-generated Essays**
Avg. word count	556.77	260.67
Avg. unique word count	275.33	146.33
Avg. sentence count	30.03	13
Avg. word length	4.69	4.94

Table 2: Text metrics analysis on training dataset

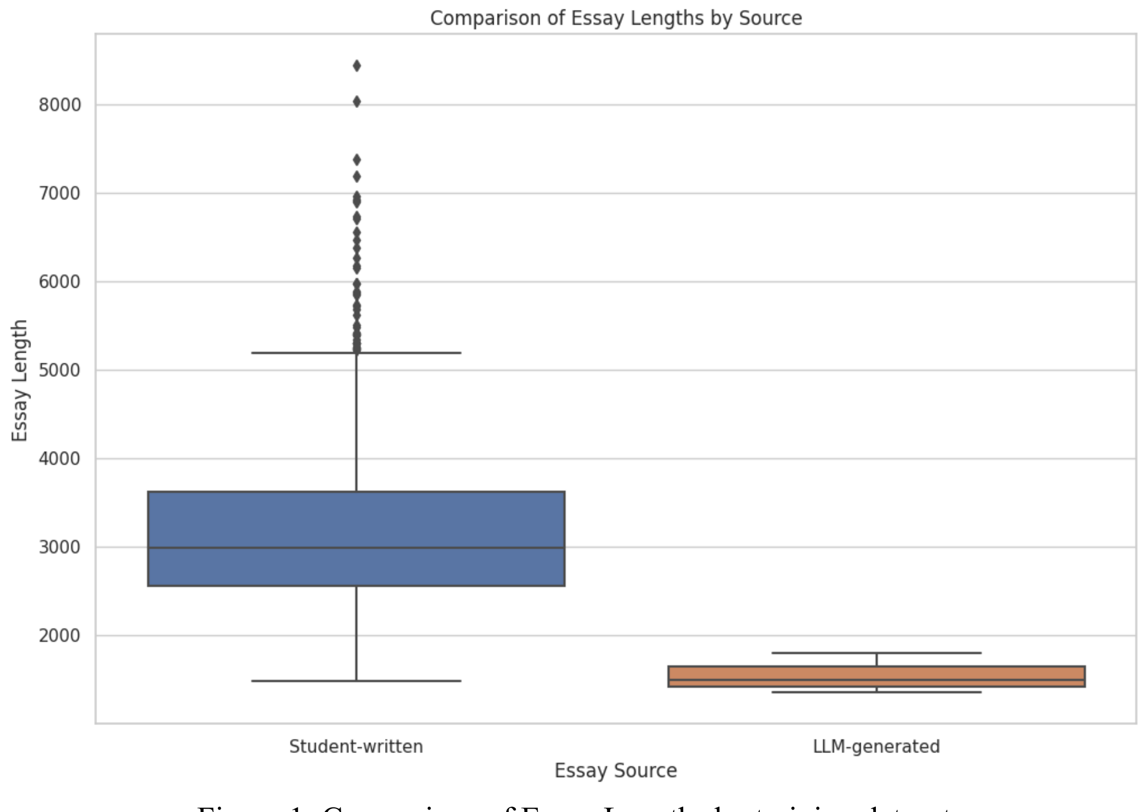

Figure 1: Comparison of Essay Lengths by training dataset

Common words of training dataset

Figures 2 and 3 indicate that the vocabulary in both student-authored and LLM-generated essays is linked to the given prompt. Excluding prepositions, the keywords utilized by the LLM-generated essays show an even stronger correlation with the prompt.

In Figure 4, **electoral** and **college** are commonly used words in Daigt V2 dataset's LLM essays. This indicates that this data aligns with the common words of the original dataset shown in Figure 3. This is the reason why we chose Daigt V2 as an external dataset to supplement the quantity of LLM essays.

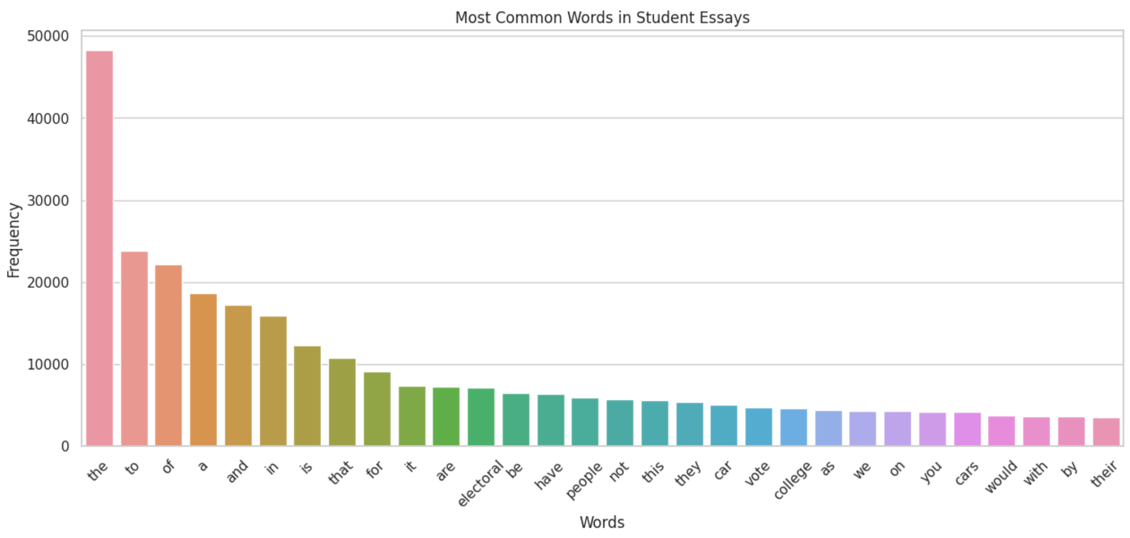

Figure 2: Most Common Words in Student Essays

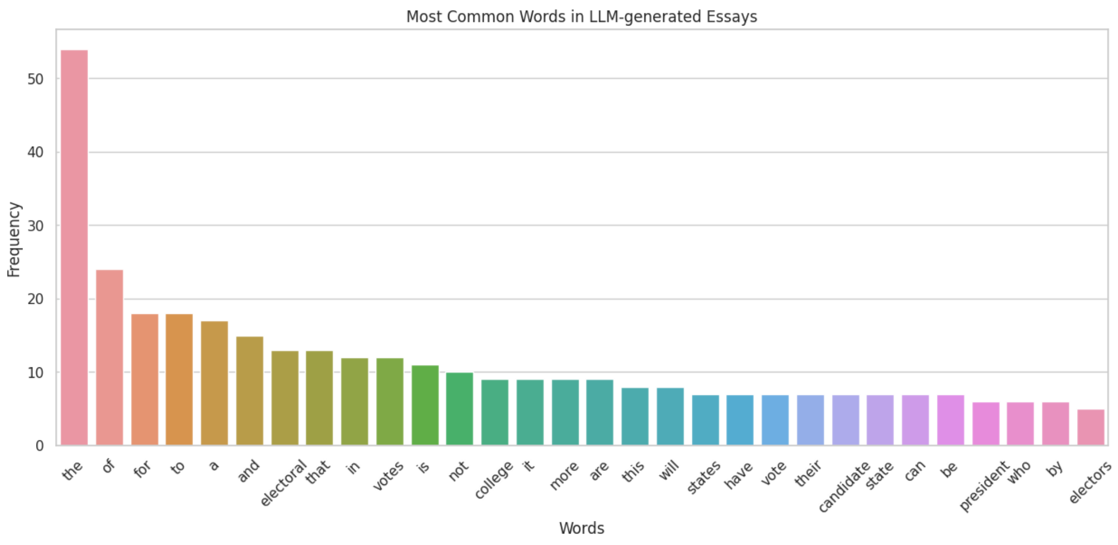

Figure 3: Most Common Words in LLM-generated Essays

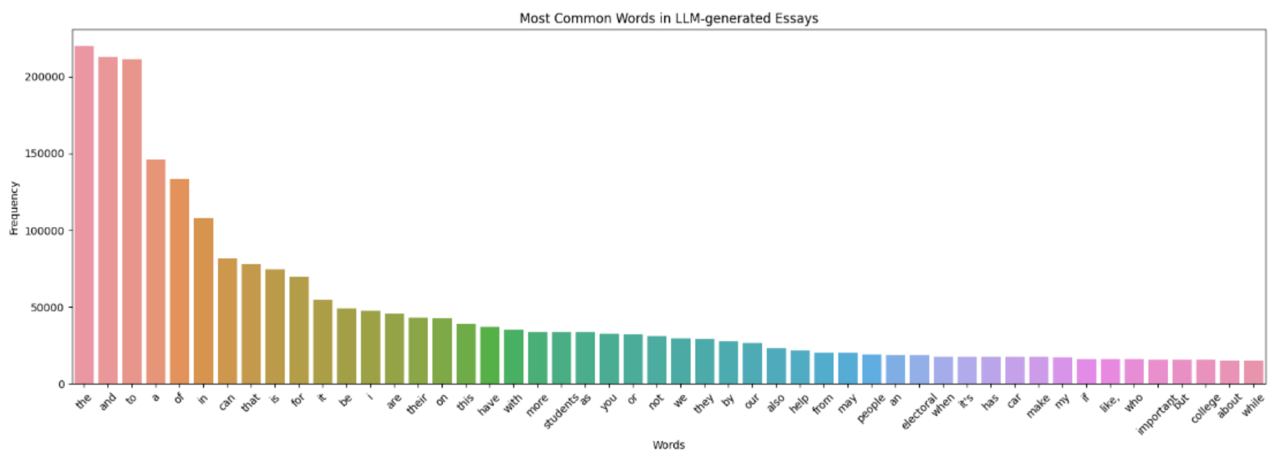

Figure 4: Most Common Words in LLM-generated Essays in Daigt V2 Dataset

Training - TF-IDF and GloVe

We use the same Tokenizer and Classifier for both the TF-IDF and GloVe methods.

Tokenizer	**Classifier**
SentencePieceBPETokenizer	VotingClassifier with MultinomialNB, SGDClassifier, LGBMClassifier, CatBoostClassifier, LogisticRegression

Table 3: Tokenizer and Classifier

TF-IDF

TF-IDF, Term Frequency-Inverse Document Frequency, is a widely used weighting technique in information retrieval and text mining. It is a statistical measure that evaluates the importance of a word in a document or a corpus by considering both its frequency in the document and its rarity across the entire collection of documents.

The method is commonly employed to highlight significant terms and facilitate tasks such as document ranking, text classification, and information retrieval.

We achieved excellent performance using the TF-IDF vectorizer, with an AUROC (Area under ROC) of 0.961. As we employed a voting classifier, we also conducted ablation experiments on various classifiers to further investigate, as shown in Table 4.

Ablation Experiments	**Public Score (AUROC)**
Five Weighted Classifiers	0.96
Without MultinomialNB	0.954
Without LogisticRegression	0.961
Without SGDClassifier	0.953
Without LGBMClassifier	0.96

Without CatBoostClassifier	0.955
MNB+SGD+Cat	0.957

Table 4: Ablation Experiments on classifier with TF-IDF

It was observed that, in terms of performance, the utilization of different classifiers did not significantly impact the results. This could be because phrases and sentences need to be compared to distinguish the style of LLM vs human. With trigrams and above, this is easily done with vectorizers. Therefore, we believe that, for traditional machine learning methods, the primary factor causing score variations is the influence of the vectorizer.

GloVe

GloVe (Global Vectors for Word Representation) is a word embedding technique designed to capture semantic relationships between words. It relies on global statistical information to generate vector representations for words.

The training process involves modeling a global word-word co-occurrence matrix, which records the frequency of words appearing together in a large text corpus. By performing eigenvalue decomposition on this co-occurrence matrix, GloVe obtains vector representations for each word, effectively capturing semantic relationships. One of GloVe's strengths is its ability to efficiently train on large-scale corpora, producing word vectors with rich semantic information.

GloVe performs worse than TF-IDF. The difference might be attributed to TF-IDF, a statistical method used to assess word significance within a collection of documents or a single document in a corpus. In contrast, Glove is a word embedding technique trained from large text corpora, placing greater emphasis on a global understanding of semantics.

For the task of distinguishing AI-generated text, using the source of the text as an analytical method gets higher performance, aligning well with our expectations.

In addition, we also discovered that we could use the test data for the tokenizer fitting task inside the Kaggle notebook because the competition submission executes the notebook during submission. However, fitting the tokenizer with test data doesn't have a big influence on the performance of the model.

	GloVe	**TF-IDF**
tokenizer fit with test data	0.559	0.961
tokenizer fit with train data	0.55	0.948

Table 5: The public score comparison between the TF-IDF method and the GloVe method.

We infer that since GloVe focuses more on the overall understanding of semantics, TF-IDF will evaluate the importance of each word in the input article. From the previous analysis of common words, there are differences in word usage between essays generated by AI and those written by humans, and we achieved excellent performance using the TF-IDF method.

4 Conclusion

In the end, we selected the best-performing model among the three methods, resulting in the final performance ranking of TF-IDF > DistilBERT > Mistral 7b. Our inference suggests that large language models like Mistral 7b may not be as effective for the specific task of detecting AI-generated text, given their relatively poorer performance and longer training times. Simpler model architectures achieve comparable results with shorter training times. However, neither of these approaches surpasses the performance of TF-IDF, which emphasizes the assessment of word importance.

For this binary classification task, our results highlight that machine learning methods outperform language models.

Model	**Public Score (AUROC)**	**Efficiency**
TF-IDF	0.961	5 hours
distilbert	0.792	2 hours
mistral-7b	0.775	6 hours

Table 6: The public score comparison between three methods.

Limitations of the Kaggle Competition

In this Kaggle competition, we cannot submit our predictions on, the training and testing process are required to be executed within the Kaggle notebook environment. Consequently, it took a significant time challenge during the training process, primarily due to the limitations of available accelerators, which included GPU T4 x2, GPU P100, and TPU VM v3-8.

References

[1] LLM - Detect AI Generated Text - kaggle,
https://www.kaggle.com/competitions/llm-detect-ai-generated-text/overview\
[2] AI or Not AI? Delving Into Essays with EDA - kaggle,
https://www.kaggle.com/code/pamin2222/ai-or-not-ai-delving-into-essays-with-eda
[3] DAIGT V2 train dataset - kaggle,
https://www.kaggle.com/datasets/thedrcat/daigt-v2-train-dataset