Detecting AI Generated Text

Daniel Firebanks-Quevedo [1] **Anthony Penaflor** [1] **Tiffany Peng** [1] **Ayush Rajpal** [1]

Abstract

In recent years, large language models (LLMs) have become seemingly proficient at producing text that is often indistinguishable from that written by humans. Inspired by the 'LLM - Detect AI Generated Text' Kaggle competition (Jules King, 2023), this project examines the effectiveness of various machine learning (ML) techniques in detecting AI-generated text. We evaluated a range of models including Random Forests, fine-tuned BERT variants, and in-context learning techniques, to identify whether an input (i.e., essays) was authored by a student or an LLM. Our methodology involved generating a diverse array of datasets, which included perturbed versions of the original competition data and artificially created texts from different domains, to test the generalizability and robustness of our models. Our findings provide significant insights into the capabilities and limitations of current machine learning techniques in distinguishing between human-written and machine-generated content, contributing to the ongoing discourse on AI-generated text detection. We make the data and code available on Github and Google Drive.

1. Introduction

The recent widespread usage of LLMs has led to an increase in the generation of text that is often indistinguishable from human-written content. This poses significant challenges for various applications, including content moderation, information verification, and school pedagogy. As LLMs become more sophisticated and their use is ubiquitous in school environments, there's an increasing need to detect whether a text has been generated by a student or a machine to combat plagiarism and ensure that the students are meeting the learning goals of a class. Inspired by the "LLM - Detect AI Generated Text" Kaggle competition, our project focuses on developing and comparing different machine learning strategies to accurately identify LLM-generated text.

Our approach integrates a spectrum of machine learning models, from traditional algorithms like Logistic Regression and Random Forests to advanced neural network architectures including DistilBERT and DeBERTA, as well as innovative in-context learning techniques using models like Llama3. The project's main contribution lies in its comprehensive dataset assembly and experimentation. We constructed several new datasets by perturbing the original Kaggle competition dataset and generating additional texts through various methods, including paraphrasing, changing prompts, and simulating different writing styles across domains such as news articles.

Through systematic testing and analysis, we aim to identify strengths and limitations of current detection methods and provide insights into the design of more effective AI detection systems. Our main findings are:

- Open-source LLM-generated essays are subpar in quality, in comparison to closed-source ones like GPT-4.
- Classical ML models like Random Forests and Logistic Regression have extremely high performance in the task of identifying LLM-generated essays given their compute cost.
- Fine-tuning DeBERTa is the most robust choice for generalization abilities.
- In-context learning does not learn the task well enough from examples and can be easily misled by spurious correlations in the prompt.

2. Data

Table 1 (2) contains a summary of all of the datasets we used for training an. The datasets can be found in the shared drive provided.

2.1. Original Dataset

The original dataset sourced in the Kaggle competition contained 1378 essays authored by middle- and high-school students based on 2 prompts, as well as 3 machine-generated es-

[1]Viterbi School of Engineering, University of Southern California, Los Angeles, CA. Correspondence to: Anthony Penaflor <apenaflo@usc.edu>, Daniel Firebanks-Quevedo <firebank@usc.edu>, Tiffany Peng <pengtiff@usc.edu>, Ayush Rajpal <arajpal@usc.edu>.

DATASET	SIZE	% GEN	METHOD
DAIGT-FULL	46246	62	MULTIPLE LLMS
XSUM-1	9998	50	LLAMA3
XSUM-2	9998	50	MISTRAL
HEWLETT-NI-1	7400	50	LLAMA3
HEWLETT-NI-2	7400	50	MISTRAL
HEWLETT-I-1	7722	50	MISTRAL
GPT2-MEDIUM	1000	100	GPT-2 MEDIUM
CNN	5330	46	GPT-J, GPT2
GPT2-ESSAYS	2754	50	GPT2

Table 1. Summary of datasets. Size refers to the total number of data points in the dataset, % Gen refers to the percentage of AI-generated text in the dataset, and Method refers to the LLM used to generate the data.

says. The labels are 0 (human-written) and 1 (AI-generated). The `DAIGT-FULL` dataset (KŁECZEK, 2024) is an extension of the original dataset, a comprehensive compilation of essays generated by various advanced LLMs such as Google Palm, GPT4, and Cohere Command. It introduces new prompts and expands the PERSUADE corpus (Broad, 2023), significantly improving the dataset's utility for training and research purposes.

2.2. Artificially Generated Datasets

To test the generalization abilities of our initial models we generated more datapoints using different methods. We created perturbed versions of the original Kaggle dataset using paraphrases, produced more essays using different prompts, and generated text from a different domain (news articles). A manual examination of some of the texts generated in these multiple settings shows that most of the open-source models have subpar generation abilities in comparison to the closed-sourced GPT-4 and PaLM, from the `DAIGT-FULL` dataset.

2.2.1. X-SUM Extensions

Following the methodolgy used by the DetectGPT paper (Mitchell et al., 2023), we created an artificial dataset of news articles using the original texts found in the XSUM (Narayan et al., 2018) dataset, and AI-generated articles in the same style. Specifically, we prompt 2 LLMs using the first 30 tokens of an X-SUM human-written news article and let the model complete the text with a minimum of 350 tokens and a maximum of 650 tokens. Unlike the DetectGPT paper, we use `Llama3-7b` and `Mistral-7b` instead of GPT-2, GPT-Neo and GPT-J. We sampled around 5000 news articles with the LLMs using `top_k=0.4` and `top_p=0.96`.

2.2.2. Hewlett 2012 Kaggle Competition

We found a Kaggle competition from 2012 (Ben Hamner, 2012) for the task of automated essay scoring, which contained essays from students in 7-10th grade, for 7 different prompts. These essays come from different prompts than the ones in the original DAIGT Kaggle competition, but they are meant to be written by students of the same schooling level, which works as a good out-of-distribution scenario. We generate 3 datasets of 7.4-7.7k examples each, in 2 different settings:

- Non-instruction (NI): Prompt the model with the first 30 tokens of a human-generated essay with no additional instructions. Using both `Llama3-8B` and `Mistral-7B-Instruct`.
- Instruction (I): Construct the LLM prompt by giving it a general description of the task + the essay instructions. Using `Mistral-7B-Instruct`. (Due to compute limits, we were not able to use `Llama3-8B` for this setting)

After manually evaluating some of the generated essays we found that the quality was very inconsistent, with some essays being well structured and others containing content that were non-essay related. For example, some Mistral-generated essays contained hashtags in a format similar to a tweet ("Let us continue to work towards a more digitally inclusive society! #DigitalInclusion #ComputerLiteracy #TechForAll"), or the original prompt itself ("You will be provided with 2 essays as examples, one written by a human and another written by a language model").

2.2.3. CNN News Articles

Our study utilized a dataset available on the Hugging Face platform, comprising human-authored CNN news articles from 2011 to 2022 (Hadas Unger). Originally, this dataset was designed to be classified into six categories: news, sports, politics, business, health, and entertainment. We adapted it for a different purpose in our research. The training set included over 32,556 articles, impractical for our needs due to the extensive volume. Consequently, we chose to work with the test dataset, which contained 5,736 articles, aligning better with our project's constraints. We utilized the GPT-2 and GPT-J language models to generate a total of 2,330 articles. The initial 1,700 articles were produced using the first 30 tokens of randomly selected original texts. For the remaining 630 articles, we used the first 100 tokens from original texts, also chosen at random. The generation process highlighted the resource intensity of the GPT-J model, as we were only able to produce 130 articles with it, in contrast to the more efficient GPT-2. Given that the original articles had an average length of 920 words, we set maximum lengths of 1,200 and 1,000 words for the GPT-J and GPT-2 models, respectively, to closely approximate the original article lengths. To balance the dataset and avoid skewness, we randomly selected 3,000 articles from the original dataset and merged them with our generated content, resulting in a comprehensive set of 5,330 articles for analysis.

2.2.4. GPT-2 Generated Essays

To balance the original Kaggle competition dataset (comprised of mostly human-written essays) we aimed to generate an equivalent number of AI-generated essays. Due to the extensive resource demands and time constraints associated with the GPT-J model, we opted for the GPT-2 model. Our methodology involved taking the first 50 words from each human-written essay as prompts to generate the remainder of the essays. The human-written essays had an average length of 556 words, with a standard deviation of 160 words. Thus, we set the maximum length of the GPT-2 generated essays at 1,000 tokens. We successfully generated 1,376 essays, which, when combined with the original dataset, resulted in a total of 2,754 essays for our analysis.

2.2.5. GPT-2 Medium Essays

We utilized the GPT-2 Medium model to help generate more AI essays to test our models on. Compared to the base model GPT-2, GPT-2 Medium contains about 221 million more parameters and features 24 layers compared to the 12 layers GPT-2 has. The pre-trained GPT-2 Medium model was fed 15 prompts, the same 15 prompts as contained in the `DAIGT` dataset, with a description of what to generate. The model was only given prompts to generate the essays with a minimum of 200 tokens and a maximum of 400 tokens. Aside from tokens, three other variables were introduced: `temperature`, `top_p`, and `top_k`. Temperature was introduced to add randomness to the generated text, with 1 being the most random. In our case, we set `temperature=0.7`. Top_p and top_k were used to help our model choose the next word when generating the essay, with `top_p` focusing on the most probably words and `top_k` limiting the number of words to consider. The values we used were `top_p=0.9` and `top_k=50`. It's worth noting that our `top_k` value is relatively high to help encourage more diverse essays. 1,000 essays were generated using GPT-2 Medium and, like the Hewlett datasets, the quality was very inconsistent. The generated essays contained ”images”, for example, ”Car Free Cities (click to enlarge).” The essays also would contain the original prompt, just as the Hewlett datasets had (e.g., ”Describes ideal Summer projects to help students learn and develop skills.”).

3. Methods

3.1. DeBERTa

Decoding-enhanced Bidirectional Encoder Representations from Transformers with Disentangled Attention (DeBERTa), is an advanced transformer-based neural network model optimized for natural language processing (NLP) tasks. DeBERTa's builds on the general BERT model and introduces novel techniques such as the disentangled attention mechanism, enchanced masked decoder, and virtual adversarial training - a method that adds small perturbations to input data to help the model learn more robust features. The model lays a robust environment for users to train and fine-tune their models for complex tasks. This was very evident during our research which made it a top-candidate to detect AI-generated text.

3.2. Adversarial Attacks Utilizing Paraphrasing Beam Generation

Given the high accuracy of our Deberta model, we decided to test the robustness by introducing adversarial attacks based off the paper 'Can AI-Generated Text be Reliably Detected?' (Vinu Sankar Sadasivan, 2024). The paper first describes the challenges and methodologies of detecting AI generated text. We leveraged the methods and insight presented by Sadasivan's paper to measure our models robustness against detecting AI-generated text by paraphrasing the text using the T5-base paraphraser.

To paraphrase the text $S = (s_1, s_2, ..., s_n)$ using the T5-based paraphraser, we apply the model's paraphrasing function f_{T5} to each sentence s_i individually. This yields a paraphrased version of each sentence $s'_i = f_{\text{T5}}(s_i)$. The resulting paraphrased text is denoted as $S' = (s'_1, s'_2, ..., s'_n)$.

Once we have the paraphrased text S', we feed it into our originally trained model for further evaluation. Let's denote our original model as f_{orig}. We then apply f_{orig} to S', resulting in the model's predictions for the paraphrased version of the text.

Mathematically, this can be expressed as, $\hat{y} = f_{\text{orig}}(S')$ where $\hat{y}$ represents the model's predictions for the paraphrased version of the passage S'.

Illustration:

$$S = (s_1, s_2, ..., s_n) \xrightarrow{\text{T5-based paraphraser}} S' = (s'_1, s'_2, ..., s'_n)$$

$$S' \xrightarrow{\text{Originally trained model}} \hat{y}$$

In this illustration, each sentence in the original text S is paraphrased individually using the T5-base paraphraser, resulting in the paraphrased passage S'. The paraphrased passage S' is then inputted into our originally trained model, and the model's predictions $\hat{y}$ for the paraphrased passage are obtained.

This approach was implemented for multiple datasets on a given model to determine how a trained model first performs under the original test data S and subsequently S'. As expected, the model generalized poorly under adversarial conditions despite performing excellent in standard testing as shown in Figures 1-2. The model could not accurately identify generated instances which could potentially have been addressed by modifying the threshold of the already trained model for Figure 1.

However, the next examples present a more complex issue; the models can neither accurately identify generated or non-generated text. This is an indicator that the model can be fooled by changes that are not obvious during the normal training process. Therefore, simply increasing volume of data or introducing complex adversarial examples to the training set would not generalize well to new and emerging threats. Making it paramount to continuously train models to perform well against adversarial attacks, and exploring methods to be able to verify the current models robustness within a well defined space.

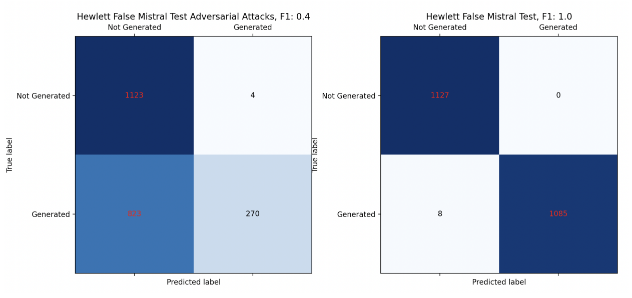

Figure 1. Deberta Adversarial and Non-Adversarial Accuracy and F1 Scores on Hewlett Mistral Dataset

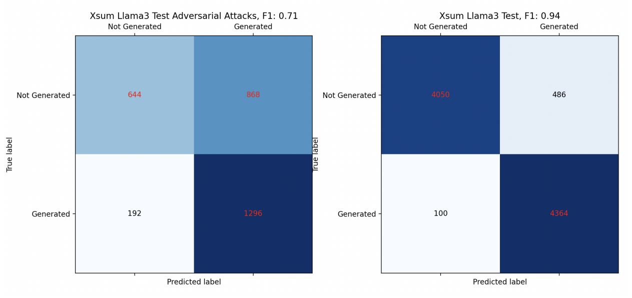

Figure 2. Deberta Adversarial and Non-Adversarial Accuracy and F1 Scores on Xsum Llama3 Dataset

3.3. TF-IDF with Multinomial Naive Bayes, Random Forest, and Logistic Regression

TF-IDF, or Term Frequency-Inverse Document Frequency, is a statistical measure that evaluates the importance of a word in a document relative to a collection of documents or a corpus. The TF component counts the frequency of a word in a specific document, while the IDF component reduces the weight of commonly used words across all documents, thereby emphasizing words that are more unique to each document. This method is advantageous as it balances word frequency with distinctiveness, providing a more nuanced feature space compared to simpler count-based methods.

In our study, we used TF-IDF in conjunction with three different models to classify texts into human-written and AI-generated categories: Multinomial Naïve Bayes, Random Forest, and Logistic Regression.

Multinomial Naïve Bayes (MNB) with TF-IDF is particularly effective for text classification, including distinguishing between human and AI-generated texts. MNB calculates the probability of each category based on the frequency of words, utilizing the TF-IDF scores as input features, which enhances its ability to identify the relevance of terms within the documents.

Random Forest (RF), an ensemble learning method using multiple decision trees, increases classification accuracy. Configured with 10 trees and the entropy criterion, it focuses on maximizing information gain, while an alternative setup of 100 trees with the Gini index aims to minimize misclassification. Applying TF-IDF with Random Forest provides a robust feature set that reduces the risk of overfitting and improves the model's ability to generalize across diverse text data types.

Logistic Regression (LR) with TF-IDF is another robust model for text classification, particularly useful for binary outcomes like distinguishing between human-written and AI-generated texts. This model estimates probabilities using a logistic function, transforming the TF-IDF weighted inputs into predictions about document categories. It excels in scenarios where relationships between features and class labels are approximately linear and can be adapted for multi-class problems using techniques like one-vs-rest (OvR).

By leveraging TF-IDF, these models significantly enhance their predictive capabilities by focusing on relevant features that effectively distinguish between human and AI-generated texts.

3.4. USE + DistilBERT

The model that combines the Universal Sentence Encoder (USE) (Google, 2020) and DistilBERT (HuggingFace) to predict whether an essay was written by a human or generated by an AI leverages the strengths of both pre-trained models to create a text classification model.

The Universal Sentence Encorder was used to encode the essay into fixed-length vector representations. USE captures the semantic information from the text which allows it to generate embeddings that represent the content and context of each essay. We use these embeddings as the input features for the classification model.

DistilBERT was used for text preprocessing and tokenization and prepared the essays before encoding them with USE. Since DistilBERT is able to understand and process natural language, the quality of our input data was therefore improved. DistilBERT can take a maximum input length of 512 tokens meaning that anything after 512 gets truncated.

The combination of USE's semantic capture and DistilBERT's contextual understanding allowed our model to differentiate between human-written and AI-written essays to help improve classification accuracy.

3.5. Llama3 and In-context Learning

In-context learning (ICL) refers to the practice of updating a pre-trained LLM with additional data while preserving its existing knowledge, through prompt-based adaptation (Brown et al., 2020). The main advantage of this approach over fine-tuning is that it enables the model to focus on the nuances of a specific task without extensively retraining on a large dataset.

For our task of detecting AI-generated text, we used the non-instruction-tuned version of the Llama3-8B (AI@Meta, 2024) model. We wrote a prompt containing the task instructions, examples of both human- and LLM-written text (2 shots per label), and an unlabeled text example for the model to label. The format of the prompt is specified in Section 8: Appendix A. According to existing literature, the model would learn to discern between authentic and generated text based on identifying the appropriate input/label space from the labeled demonstrations, the expected format for the predicted output (Lyu et al., 2023), and potentially a task vector. (Hendel et al., 2023).

In order to ensure that the model predicts the expected labels, we make the model generate a single token (since the labels `"0"`/`"1"` are represented with one token in the model we used), and get the logprobs of the top 100 words for the generated token. The model contains the labels within the top 100 words around 95-99% of the time, so we choose the label that has the highest logprob as the model prediction.

4. Results

Overall, we were able to replicate the same levels of accuracy ($\geq 95\%$) achieved in the Kaggle competition, using even simpler models such as Logistic Regression and Random Forests. The most robust model against perturbations and out-of-distribution datasets is the fine-tuned version of DeBERTa. We did not find any advantages to using the specialized sentence embeddings from the Universal-Sentence-Encoder, nor the knowledge acquired during the pre-training of Llama3 to perform well in this task. A full summary of our results can be seen in Table 2 (3.5).

4.1. TF-IDF + Multinomial Naive Bayes

In our study, the Multinomial Naïve Bayes model, despite being more basic compared to the more complex Random Forest and Logistic Regression models used with TF-IDF, demonstrated competitive performance on certain datasets. Nonetheless, its limitations became evident in specific instances, as reflected in the accuracy scores across different datasets: 49.36% for `XSUM-1`, 58% for `HEWLETT-NI-1`, 5.5% for `GPT2-Medium`, and 56.41% for `CNN`. An interesting variability was observed within the `CNN` dataset results; accuracy fluctuated dramatically, dropping to as low as 5-6% in some trials, while reaching 55-60% in others. Furthermore, the current assessment yielded an F1 score of only 3.33%, highlighting challenges in the model's consistency and reliability in certain contexts.

4.2. TF-IDF + Random Forest

In our analysis, the Random Forest algorithm outperformed the Multinomial Naïve Bayes, which was anticipated given its more complex nature. We employed two variants of Random Forest: one with 10 decision trees using entropy as the criterion, and another with 100 decision trees using the Gini index. The choice of the Gini index for the larger model was based on its efficiency in computation time compared to entropy. As predicted, the Random Forest model with 100 decision trees and the Gini criterion demonstrated superior performance. Overall, Random Forest exhibited greater consistency in its results compared to the simpler Multinomial Naïve Bayes model. Additionally, Random Forest proved to be a simple yet effective model for distinguishing AI-generated text from human-written text.

4.3. TF-IDF + Logistic Regression

The performance of Logistic Regression was on par with Random Forest and even surpassed it in several instances. For example, in the GPT2-Medium dataset, Logistic Regression achieved an accuracy of 39.4%, outperforming Random Forest's 27.4%. Additionally, it slightly exceeded Random Forest by 0.5-2% in datasets such as DAIGT-FULL, XSUM-1, and HEWLETT-NI-2. Similar to Random Forest, Logistic Regression demonstrated its efficacy as a straightforward yet robust model for text classification. Moreover, its performance remained stable across multiple evaluations on the same datasets, underscoring its reliability in consistent predictive accuracy.

	TF-IDF + MNB	TF-IDF + RF	TF-IDF + LR	USE + DistilBERT	DeBERTa	Llama3-ICL
DAIGT-FULL	94.29/91.79	98.16/97.47	99.22/98.95	86.83/79	**100/100***	60.68/55.13*
XSUM-1	61.36/53.82	77.72/77.39	79.40/79.36	62.18/39	**95.33/95**	50/33
XSUM-2	49.36/47.51	61.92/62.22	61.32/61.40	56.35/15	**93.5/93.7**	51.95/44.85
HEWLETT-NI-1	58.00/28.78	94.81/94.87	94.65/94.75	60.48/40	**99.54/96.26**	50.07/33.89
HEWLETT-NI-2	92.65/91.99	97.41/97.32	98.00/97.97	78.87/66	**99.63/99.89**	51.93/42.53
HEWLETT-I-1	99.84/99.85	**100.00/100.00**	99.79/99.80	97.84/97	**100/100**	80.49/80.49
GPT2-MEDIUM	5.50/10.43	27.40/43.01	39.40/56.53	20.7/34	0.0/0.0	87.4/93.28
CNN	56.41/3.33	87.02/84.92	85.22/81.71	83.94/75	**97.74/97.50**	54.65/54.57
GPT2-ESSAYS	95.79/95.72	98.26/98.30	96.37/96.39	85.43/77	**100/100**	52.5/38.82

Table 2. Accuracy/F1 score in % for all the datasets/models we tested. For the models with * on the score, we only tested the performance on a subset of 10k data points.

4.4. USE + DistilBERT

This method had the second worst performance out of all the models. While it achieved moderate success on certain datasets it struggled to predict `XSUM-1`, `XSUM-2`, and `HEWLETT-NI-1`. As seen in Table 2, the model performed the worst on `GPT-2 Medium` generated essays. This could be attributed to the higher diversity in these essays, driven by a `top_k` setting of 50, which introduced more variability. Despite essays being generated from the same prompts as those in the `DAIGT` dataset, the model failed to accurately predict their origin.

The combination of USE and DistilBERT faces inherent limitations in tasks requiring identification of whether an essay was written by a human or an AI. USE, focusing on broad semantic meanings, may not capture the subtle differences between AI and human text. Meanwhile, DistilBERT, being a distilled version of BERT, sacrifices depth of linguistic understanding for increased processing speed. This reduction in depth likely affects the model's ability to detect the nuanced linguistic cues that are crucial for distinguishing between AI-generated and human-written essays.

4.5. DeBERTa

Deberta demonstrated remarkable performance across all datasets. The hyper parameters were tuned on two datasets until the learning rate, optimizer, and epochs demonstrated an optimized training cycle. Some datasets required the reduction of epochs to reduce overfitting, which was evident in Hewlett-NI-1 - once the epochs were reduced to two, some cases one, the model generalized well to unseen data. The anomaly is GPT-2-Medium, the process consisted of training on DAIGT and then running inference. Takeaways from this is that DeBERTa generalizes well on data close to it's training data, but not on patterns from other datasets.

4.6. In-context Learning

This method had the worst performance out of all the models we used. In most of the datasets the accuracy was no better than random (50%), as it predicted a single label for the majority of the test examples. A potential hypothesis behind this behavior is that the model noticed a pattern in the order of the labeled demonstrations: label 0, followed by label 1; therefore when we fed a test example, it would most of the times predict 0 because it expected that it would follow such order. We tested this hypothesis by changing the order of the demonstrations and found that putting label 1 first and then label 0 leads to more examples being predicted with label 1, which leads to a better performance on `GPT2-medium`.

Another potential limitation is that we didn't try to test model performance with multiple prompts, as suggested in (Voronov et al., 2024) and (Mizrahi et al., 2024), which could have hindered our performance on this dataset and model configuration.

The model performs the best on `HEWLETT-I-1`, the dataset containing essays that were generated using the original instructions. This is probably due to the essays being generated from scratch as opposed to being completed from the start of a human-generated essay, which may lead to more clear distinctions between essay types.

5. Process

At the beginning we set ourselves to replicate the results from the Kaggle competition in order to get familiar with different models available. Once we were able to achieve high accuracies, we set on 2 different directions: to try to lower the accuracy by perturbing the original data using paraphrases, and to try out new data using prompts specifically asking the model to generate essays that could deceive a detector. We also decided to try the performance of in-context learning in this task because it wasn't used much in the Kaggle competition, and because it would allow us to experience with more modern techniques.

6. Contributions

The results and deliverables of this project contributes many starting points from where other research can branched out from. To include but not limited to, performance comparisons between AI detection models, additional datasets to train models on, and continuations points to further explore the research of developing robustness against adversarial attacks and perturbations.

7. Conclusion

We presented multiple models, datasets, robustness verification methods. We experimentally validated our models across multiple case studies. For future work, we would be interested in improving the robustness of our models against adversarial attacks, and defining verified properties of our models.

References

AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

Ben Hamner, Jaison Morgan, l. M. S. T. V. A. The hewlett foundation: Automated essay scoring, 2012. URL https://kaggle.com/competitions/asap-aes.

Broad, N. Persuade v2, 2023. URL https://www.kaggle.com/datasets/nbroad/persaude-corpus-2/.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020.

Google. universal-sentence-encoder, 2020. URL https://tfhub.dev/google/universal-sentence-encoder/4.

Hadas Unger, Ayoub CHERGUELAINE, F. B. Cnn news articles 2011-2022. URL https://huggingface.co/datasets/AyoubChLin/CNN_News_Articles_2011-2022.

Hendel, R., Geva, M., and Globerson, A. In-context learning creates task vectors, 2023.

HuggingFace. Distilbert. URL https://huggingface.co/docs/transformers/en/model_doc/distilbert.

Jules King, Perpetual Baffour, S. C. R. H. M. D. Llm - detect ai generated text, 2023. URL https://kaggle.com/competitions/llm-detect-ai-generated-text.

KŁECZEK, D. Daigt v2 train dataset, 2024. URL https://www.kaggle.com/datasets/thedrcat/daigt-v2-train-dataset/data.

Lyu, X., Min, S., Beltagy, I., Zettlemoyer, L., and Hajishirzi, H. Z-icl: Zero-shot in-context learning with pseudo-demonstrations, 2023.

Mitchell, E., Lee, Y., Khazatsky, A., Manning, C. D., and Finn, C. Detectgpt: Zero-shot machine-generated text detection using probability curvature, 2023.

Mizrahi, M., Kaplan, G., Malkin, D., Dror, R., Shahaf, D., and Stanovsky, G. State of what art? a call for multi-prompt llm evaluation, 2024.

Narayan, S., Cohen, S. B., and Lapata, M. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J. (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1797–1807, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1206. URL https://aclanthology.org/D18-1206.

Vinu Sankar Sadasivan, Aounon Kumar, S. B. W. W. S. F. Can ai-generated text be reliably detected?, 2024. URL https://arxiv.org/pdf/2303.11156.

Voronov, A., Wolf, L., and Ryabinin, M. Mind your format: Towards consistent evaluation of in-context learning improvements, 2024.

8. Appendix A: Sample prompt templates used in ICL

```
"You will be provided with 2 essays as
    examples, one written by a human
    and another written by a language
    model.
Text: <TEXT>
Label: 0
Text: <TEXT>
Label: 1
You are an expert detector and your
    task is to identify whether the
    next essay was written by a human
    or language model.
```

```
Answer 0 if it was generated by a human
     or 1 if it was generated by a
   language model.
Text: <TEXT>
Label:"
```