MACHINE-GENERATED TEXT ATTRIBUTION *

Sinjoy Saha
The Pennsylvania State University

ABSTRACT

Advancements in text generation technology have made it easy for malicious users to generate large volumes of human-like content without technical expertise, raising concerns about misuse, such as fake news and phishing. Detecting AI-generated text is crucial for responsible use and content moderation, leading to increased research in Machine-Generated Text (MGT) Detection. A more specific challenge is attributing MGT to the specific model that generated it, known as MGT Attribution. In this paper, we review existing datasets and methods for model attribution and introduce a new dataset to train a classifier for identifying the source LLM behind a given text. We study the effects of different datasets and varying lengths of generated texts on LLM attribution. We empirically show that it is easier for a classifier model to attribute the input text to an LLM when the LLMs have been prompted with technical questions than being prompted with only simple text completion tasks. We also show that it is easier to attribute longer texts than shorter texts. Finally, we gain some preliminary insights on the effects of parameter size and pre-training dataset on text generation.

***K*eywords** LLM · machine-generated text attribution · classification · BERT

1 Introduction

The field of text generation is undergoing a major shift with the rise of powerful Large Language Models (LLMs) like Generative Pre-trained Transformer (GPT-4) [1], LLaMA-3 [2], PaLM [3], and T5 [4]. These models, with their vast parameter sizes, excel at generating text that mimics human language and have shown impressive performance across various tasks. They generalize across tasks through zero-shot and few-shot learning, significantly reducing the need for task-specific training.

However, these advancements in text generation technology come at a cost where malicious users could easily generate large amounts of human-like language without technical knowledge or intervention. LLMs can be misused for unethical purposes, such as generating fake news or phishing content. Thus, it has become extremely important to be able to detect texts that are generated using LLMs to be able to use LLMs responsibly and moderate content being published at scale. This has led to growing research on detecting AI-generated text, especially in cases of misinformation or propaganda. This task of distinguishing machine-generated text from human-written ones is often termed Machine-Generated Text (MGT) Detection. A further fine-grained sub-task is to be able to attribute the MGT to one of the many large language generation models that were used to generate it. This is often termed MGT Attribution or Model Attribution.

In this paper, we study the prior datasets and methods that have been used for attributing AI-generated text to one of the LLMs. Further, we create our own dataset for training an LLM classifier to label which model was used to generate a given text input.

2 Related Work

Related Work

[5] [6]

In this paper, we

*Submission for the mid-term project for CSE 584 - Machine Learning: Tools and Algorithms.

3 Dataset

The dataset for the model attribution task consists of a human-generated input text which is given as a prompt to the various text generation models and the corresponding text generated by the models.

3.1 Input Prompts

We consider two different sources of texts as the input prompt to the models to study the effect of texts from different domains on the model generations. The two text sources are 1) Wikipedia [7], and 2) GSM8K [8]. The next two sub-sections describe these datasets in detail.

3.1.1 Wikipedia

We built this dataset from the Wikipedia dumps [7] with cleaned English language texts till November 2023. The dataset is downloaded from Hugging Face [2]. The entire dump is of 6.4 million samples. However, due to resource constraints, we consider only $\sim$1500 texts for this study. Each row in the dump is a Wikipedia article from which markdown and unwanted sections are stripped off. Further, we select examples where at least five paragraphs of text are present. We use the regular expression `r'([A-Z][^ .!?]*[.!?])\s+(?=[A-Z])'` to find entire sentences. These sentences are the human-generated "ground truths". Up to five words from the sentences are used as input to prompt the generative models. The 'train' split is used to create the training and validation set and the 'test' split is used for the test set. At most two sentences are chosen from each paragraph to maintain the diversity of the dataset. It is important to note that there is no overlap between the training, validation and test set.

3.1.2 GSM8K

The second dataset is Grade School Math 8K (GSM8K) [8][3] dataset which consists of 8.5K high-quality grade school math word problems. The motivation for choosing this dataset is to study the impact of technical and mathematical prompts on the various language models. Since many language models are trained for simple everyday text generation tasks and not to solve complex mathematical problems, the output of these prompts might help the classification model distinguish between the various models used for generation. Similar to the previous dataset, the 'train' split is used to create the training and validation set and the 'test' split is used for the test set. Only about 500 questions in the dataset are used as human-generated prompts for the models due to resource constraints.

3.2 Text Generation Models

While the language models chosen for text generation are decoder-only architecture, they vary in parameter sizes and families. The `pipeline` API from the Hugging Face `transformers` library is used for generation[4]. For an open-ended generation, the padding token is set to the EOS token for every model and a temperature of 0.7 and a maximum length of 256 tokens are used. Table 1 gives a summary of the models used.

Table 1: Summary of decoder-only text generation models used to create the dataset.

Model	**Institute**	**# Parameters**
GPT-2-small	OpenAI	117M
GPT-2-XL	OpenAI	1.5B
Phi-2	Microsoft	2.7B
Falcon-7b	TII	7B
Mistral-7B-Instruct-v0.2	Mistral AI	7B

For each of the datasets, these five models are used to generate the output text. Thus, a total of 9885 generations are created and stored as CSV files. Table 2 shows the exact number of samples per dataset split.

[2] https://huggingface.co/datasets/wikimedia/wikipedia

[3] https://huggingface.co/datasets/openai/gsm8k

[4] https://github.com/huggingface/transformers

Table 2: Split for the two datasets containing text generated by five language models.

Dataset	Training	Validation	Test
Wikipedia	5445	995	1005
GSM8K	1680	425	335
Total	7125	1420	1340

4 Method

Given an AI-generated input text, the model attribution task can be framed as a sequence classification task. In this paper, BERT [9] is used as the backbone model for the sequence classification. Specifically, the `bert-base-cased`[5] model is used with a BertForSequenceClassification API from transformers library.

Next, we describe the steps for training the sequence classification model. The datasets with their corresponding splits are loaded from memory. The model generated texts are now the input to the classifier and the models are the labels to be predicted. The BERTokenizer is used to re-tokenize the input texts and along with their corresponding labels, the training, validation and test for each dataset is created. An additional dataset called 'all' is also created which consists of both datasets. The datasets are randomly shuffled using a random seed for repeatability.

Since this is a classification problem, we use the cross-entropy loss for our classifier model. The AdamW optimizer is used with the Training API of transformers, which is the Adam optimizer with a linear weight decay of 0.01. The warm-up steps begin with a learning rate of 0, gradually increasing to 1.0, following a linear weight decay schedule. AdamW is chosen as it is the most effective for training transformer models like BERT. The batch size for training and validation is kept at 32 according to the GPU memory available. For the first experiment using only the Wikipedia dataset, the classifier model is trained for 10 epochs. However, as seen from Figure 1, the validation loss starts to increase at 200 steps indicating that the model starts to over-fit. Hence, subsequent experiments are done only till 5 epochs.

Figure 1: Training and validation loss over training steps for BERT sequence classifier on Wikipedia dataset.

5 Results

In this section, we summarize the results of two studies. First, we study the effect of including text generated by the LLMs for mathematical prompts in the dataset. Second, we study the effect of sequence length on classification performance.

[5]https://huggingface.co/google-bert/bert-base-cased

5.1 Effect of mathematical prompts

For this experiment, we run the BERT classifier for only the Wikipedia dataset and then for the combined Wikipedia and GSM8K dataset. As mentioned in the previous section, the model starts to over-fit quite soon so the combined dataset is trained only for 5 epochs. As evident from Figure 2, the model starts to over-fit after 200 steps. The best model according to validation loss is taken for further analysis.

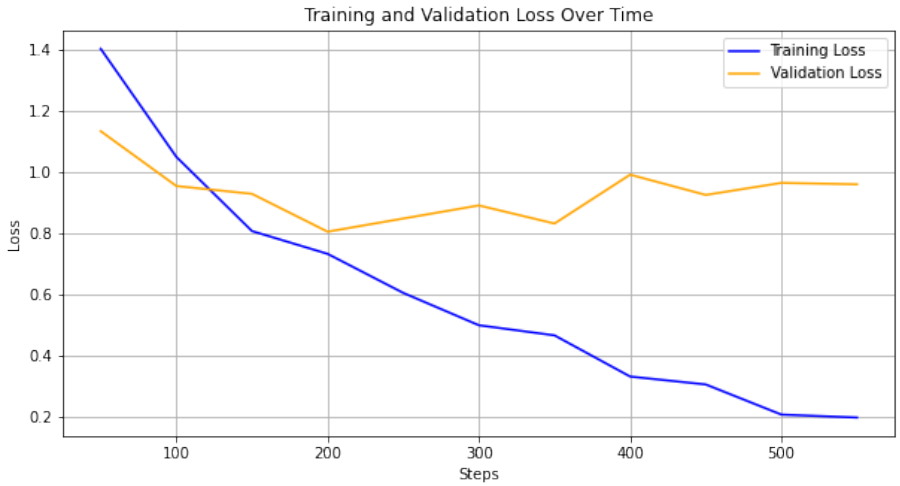


Figure 2: Training and validation loss over training steps for BERT sequence classifier on combined Wikipedia and GSM8K dataset.

Figure 3 shows the plots for accuracy, macro f1-score, precision and recall for the two experiments. It is to be noted that the random baseline accuracy is at 0.20 and our classifier model does significantly better than that. An interesting observation is that the accuracy of the classifier model increases from 0.60 to 0.65 after including the math dataset. This shows that when LLMs are prompted with mathematical questions, the classifier can better distinguish and attribute the text generated by different models. Also, the precision remains almost the same in both cases, which shows that the classifier does not get easily confused with the additional math dataset.

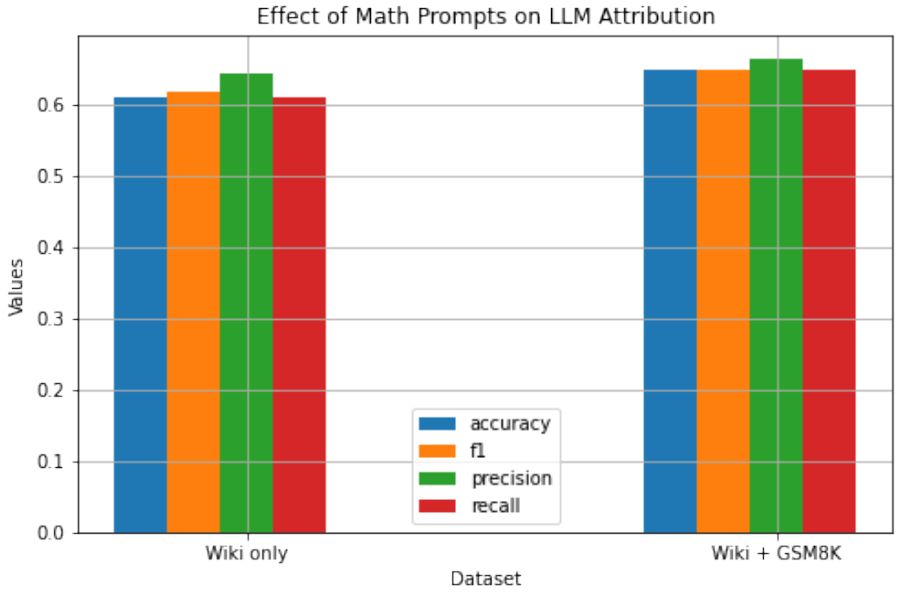
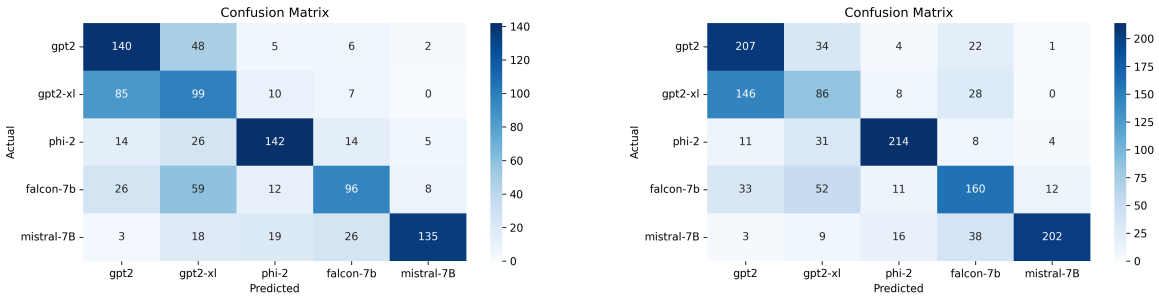


Figure 3: Accuracy, F1-score (macro), precision and recall for only Wikipedia and both Wikipedia and GSM8K.



(a) Wikipedia only

(b) Both Wikipedia and GSM8K

Figure 4: Confusion matrices for only Wikipedia and both Wikipedia and GSM8K.

Further, a couple of interesting observations can be seen from the confusion matrices plotted in Figure 4. First, the classifier model gets confused between GPT-2 and GPT-2-XL. This may be due to the fact that both models share very similar datasets for pre-training. Although GPT-2-XL is an order of magnitude larger than GPT-2, it can be said that there is a 50% chance that a text generated by GPT-2-XL could have been generated by GPT-2. The second observation is that the classifier sometimes attributes texts generated by Falcon-7B to either GPT-2-XL or GPT-2. Again, this may be due to their similar architecture and pre-training datasets. This shows that the parameter sizes of models for open-ended simple text generation may not matter too much. It is only when the models are asked to perform domain-specific or technical tasks the larger parameter models generate drastically different texts than the smaller models.

5.2 Effect of length of generated text

In this section, we study the effect of varying the number of tokens used for attributing the generated text to an LLM. To keep the experiments straightforward only the Wikipedia dataset is used in this case. The length of the input text is truncated to 64, 128 and 256 and then padded to 512 to feed into the BERT classifier model. Figure 5 shows the different performance metrics for the three experiments. For all three cases, the classifier does better than the random baseline accuracy of 0.20. However, it performs increasingly better when additional tokens are available from the LLMs. This shows that shorter texts are more difficult to attribute than longer texts. This is due to the fact that the probabilities of the shorter texts coming from any of the models are much higher. As the texts get longer, the probabilities keep getting multiplied and diminished. Thus, it gets easier for the classifier to distinguish with more available context.

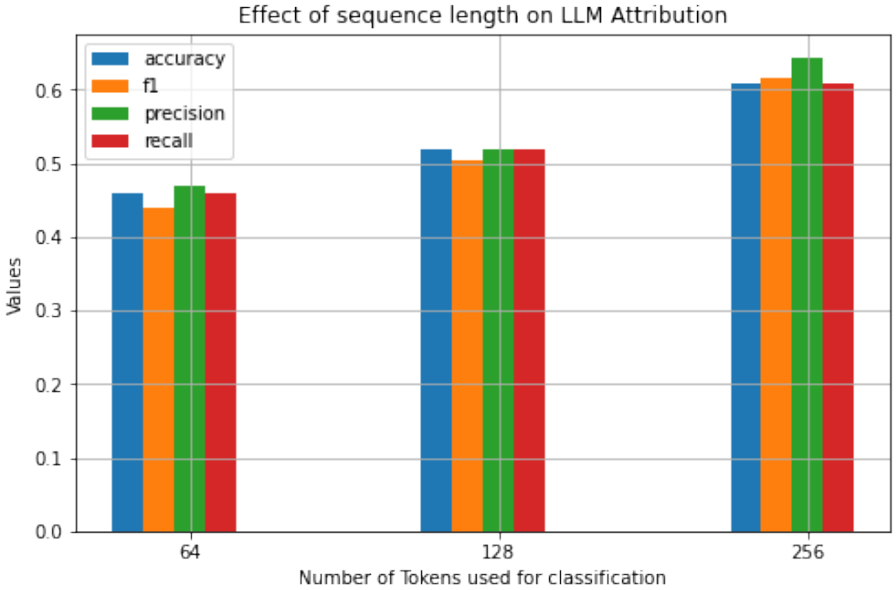
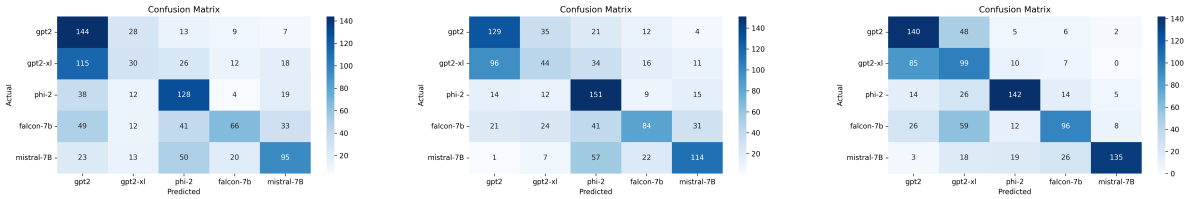


Figure 5: Accuracy, F1-score (macro), precision and recall for varying generated text sequence lengths (64, 128 and 256 tokens) used for classifier model.



(a) # Tokens = 64 (b) # Tokens = 128 (c) # Tokens = 256

Figure 6: Confusion matrices for varying generated text sequence lengths (64, 128 and 256 tokens) used for the classifier.

Figure 6 shows the confusion matrices for the varying sequence lengths. Again, the classifier model confuses between GPT-2 and GPT-2-XL. Interestingly, for sequences shorter than 128 tokens the classifier also gets confused between Phi-2 and Mistral-7B.

6 Conclusion

In this work, we study the effects of different datasets and varying lengths of generated texts on a classifier performing the task of LLM attribution. We show that it is easier for the classifier model to attribute the input text to an LLM when the LLMs have been prompted with high-school-grade math word problems than being prompted with only simple text completion tasks. We also show that if longer machine-generated texts are available, it is easier for the classification model to distinguish between the LLMs. It is also seen that models of the same family and with similar pre-training datasets such as GPT-2 and GPT-2-XL, even with much more parameters, "sound" similar to the classifier from their generated texts. This shows the importance of the quality and diversity of pre-training data on text generation. Overall, we see that Phi-2 is the easiest for the classifier to attribute. A deeper analysis of the actual model outputs may be required to draw better insights. Further experiments may be performed to study other factors such as parameter size and model architecture on natural language generation.

References

[1] OpenAI. Gpt-4 technical report, 2024.

[2] Abhimanyu Dubey et al. The llama 3 herd of models, 2024.

[3] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[4] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[5] Harika Abburi, Kalyani Roy, Michael Suesserman, Nirmala Pudota, Balaji Veeramani, Edward Bowen, and Sanmitra Bhattacharya. A simple yet efficient ensemble approach for AI-generated text detection. In Sebastian Gehrmann, Alex Wang, João Sedoc, Elizabeth Clark, Kaustubh Dhole, Khyathi Raghavi Chandu, Enrico Santus, and Hooman Sedghamiz, editors, *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 413–421, Singapore, December 2023. Association for Computational Linguistics.

[6] Areg Mikael Sarvazyan, José Ángel González, Marc Franco-Salvador, Francisco Rangel, Berta Chulvi, and Paolo Rosso. Overview of autextification at iberlef 2023: Detection and attribution of machine-generated text in multiple domains, 2023.

[7] Wikimedia Foundation. Wikimedia downloads.

[8] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.