Supervised Contrasting Fine Tuning of PLM for Machine Generated Text Detection

Muralidharan Kumaravel[1] Sai Supreeth Reddy Bandi[2] Kewal Jayshankar Mishra[3]

Abstract

This project presents a study of machine-generated text detection using BERT, employing three different methodologies: Bert - Supervised Finetuning, BERT - Contrastive Fine Tuning, and Unsupervised Domain Adaptation Through Backpropagation for BERT. We used the data from the HC3 corpus and found that while supervised machine-generated text models lacked data domain adaptability, even with the use of domain adaptation methods, the model's performance was still poor. The study highlights the need for further research to improve domain adaptability in machine-generated text detection in supervised setting (Black box detection).

1. Introduction

The rapid advancement of large language models (LLMs), such as OpenAI's GPT series, has led to an unprecedented influx of machine-generated text that closely mimics human-generated content. The resemblance between these texts has reached a level where it is becoming increasingly difficult for humans to distinguish between the two. This has led to the emergence of machine-generated text detection as one of the most challenging and intriguing problems in the field of artificial intelligence research. There are two primary approaches to detecting machine-generated text: "white box" and "black box" detection methods. White box detection methods involve having access to the backbone model that generated the text, while black box detection methods do not require such access. Each approach has its own unique set of techniques and algorithms to address the problem.

White box detection techniques, such as watermarking algorithms and DetectGPT, leverage the knowledge of the underlying model to identify specific patterns or characteristics that are unique to machine-generated text. These methods typically offer higher accuracy, but their reliance on access to the original model limits their applicability in real-world scenarios. On the other hand, black box detection methods, such as supervised detection, employ machine learning algorithms to differentiate between human and machine-generated text without any knowledge of the generating model. Although these methods are more versatile, they may be less accurate than white-box techniques due to their reliance on limited training data and potential biases. As the prevalence of machine-generated text continues to grow, developing robust and reliable detection methods is paramount. The contrastive fine-tuning of pre-trained language models offers a promising avenue to improve the detection of machine-generated text, harnessing the power of existing LLMs while mitigating their weaknesses. This research aims to explore and develop innovative solutions by combining the strengths of both white box and black box detection techniques.

1.1 Research Contribution

OpenAI released their version of Roberta to detect text generated from ChatGPT. But they all mention all the drawbacks that their fine tune Roberta is not a good detection system. We did cross domains experiments to empirically prove lack of data domain adaptability and we tried contrastive learning and domain adaptability method to check whether data domain adaptability is possible in this task. We are the first one to try different methods for data domain adaptability in MGT detection task and we highlights the need for further research to improve domain adaptability in machine-generated text detection in supervised setting.

2. Related Works

With the advancement in LLMs, the text generated by LLMs becomes often indistinguishable from human-written content. Thus, LLMs are used in various tasks, which further creates concerns about the potential unethical use of these LLMs-generated texts.

Detecting LLM's generated text has been intriguing and challenging to the research community. GLTR [1] is a statistical method of detecting and visualizing MGT. In the method for a given text, altered completion is ranked based on the likelihood of being generated from a backbone model, and visualizations are provided. This approach is to identify potential MGT by identifying discrepancies in word usage patterns. The authors of the paper [2] ran experiments with the data collected using various decoding strategies (top k, top p, random sampling) with the methods Fine-tuning BERT, BagOfWords, HistGLTR). Further, they also made humans classify the text as whether is MGT or HGT. The findings show that automated methods perform well when the top k sampling method was used as the decoding strategy to collect the data. On the contrary, humans find it difficult to detect MGT text when top k sampling was used. Further, Fine-tuned BERT shows better performance with a longer sequence length (192 tokens). [3] The authors have created a dataset with a longer sentence length in the Russian language for machine-generated text detection tasks. They used generative models for the Russian language like RuGPT3, and Xglm with three decoding methods (top k (40), top p (0.96), and untruncated sampling similar to the [1]. They train the classifier using the generated data using XLM-Roberta. The authors have evaluated the impact of sequence length on prediction performance. Furthermore, they proposed a hypothesis that having a multiclass classification on fine-tuning increase the performance of detection. They have created a custom architecture where the fine-tuning of XLM Roberta has four different types of output, multiclass, sber-small, sber-large, and Facebook and they analyzed the results. As with an increase in sequence length the prediction performance increases. XLM Roberta for classifying the generated Russian text dataset performed better when sequential mixing of the data from different base models used for generating the text was used. Further, the authors found that using a multiclass head in fine-tuning increases the model performance in machine-generated text detection. Token distribution can make prediction easier.

[4] In this paper, the author took a different approach to supervised training. They used stylometric features along with a PERT algorithm to find points of change in the author style to detect AI-written tweets. They constructed their own dataset consisting of human and machine-generated text and used a dataset named Tweet Fake. [5] The authors have proposed to use custom coherence graph representation along with Roberta sentence level representation to perform contrastive fine-tuning for machine-generated text detection. For a given sequence of text, they performed NER to extract entities and used an edge for the entity present in the sentence. And they used sentences embedded from Roberta-base for the nodes, and the edge between entities if they occur in multiple sentences. Then they used 2-level multiple GCN for different relations in the graph and summation them gave the final graph coherence representation. Further, they applied attention-based LSTM to the extracted graph coherence features, that is the final representation of Zgraph. With this representation and along with the sentence representation from Roberta, they performed CE + SCL as a downstream task.

3. Proposed Methods

3.1 Dataset

HC3 corpus consists of 24322 questions from different domains and a total of 58546 Human answers and 26903 ChatGPT answers. This dataset was created with the intention of comparing ChatGPT's responses to those of humans. These include open-domain, financial, medical, wiki and reddit data. The sources for each domains are ELI5 dataset, WikiQA dataset, Crawled Wikipedia dataset, Medical Dialog dataset and FiQA dataset.

3.2 Baseline

Using BERT for sentence embedding, a feed-forward neural network with 512 neurons, followed by an output layer with 2 neurons, makes up the methodology. A SoftMax activation function is used in the top layer to produce probability scores for each class, enabling quick and precise classification. This is a naïve approach to perform binary classification that the given text is machine generated or human generated.

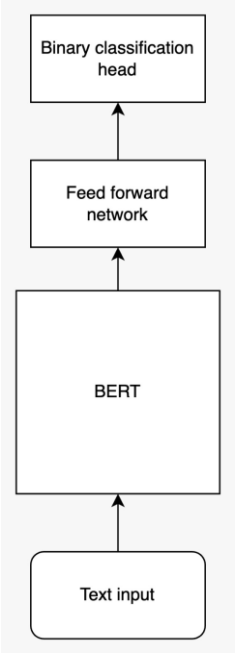

Fig 1: Architecture for baseline (BERT Supervised fine tuning for binary classification.

3.2 Supervised Contrastive Fine-tuning of BERT

When compared to the baseline method, in this method there is a secondary contrastive loss, which helps the model to separate the embedding for two different classes. The PLM is backpropagated and fine-tuned via contrastive tuning, which takes advantage of class similarity. In order to perform supervised contrastive fine-tuning for the binary classification task, we used BERT as the foundational model. In the below equation, L_{SC} is the contrastive loss between classes.

$$L_{Final} = L_{CE} + \lambda L_{SC}$$

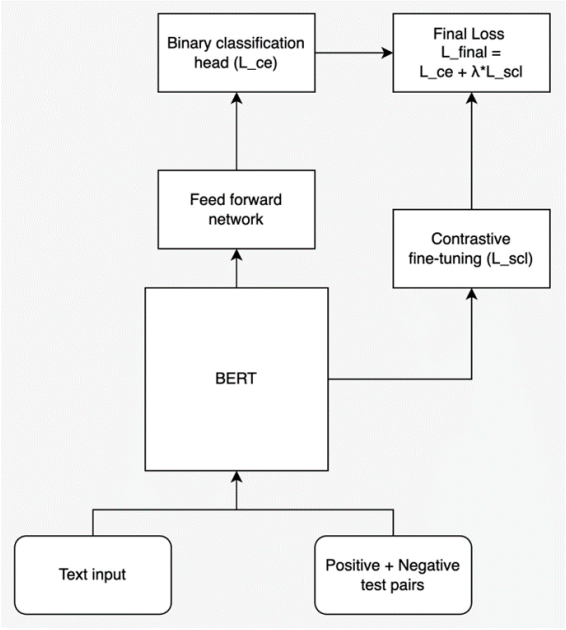

Fig 2: Supervised Contrastive Fine Tuning of BERT

3.3 - Unsupervised domain adaptation through Backpropagation

It seeks to modify a model that has been trained on one domain to perform well on another without needing labeled data from the target domain. The objective is to learn a representation that is constant between domains. For implementation, typically consists of three steps: Using a sizable, varied dataset that contains data from the source and target domains, pre-train a deep neural network. This pre-training stage aids the network's learning of a set of features that may be applied to other domains. Utilizing backpropagation, fine-tune the pre-trained network on the labeled source domain data to reduce the task-specific loss function. The learnt features from the pre-training step are kept when the network is adjusted to the source domain task. And finally, the domain loss is maximized using a gradient reversal layer in the feature extractor (BERT) with a scaling factor such that the model is tricked into learning the universal representation for the given task. In the below equation LDL is the domain loss and gradient ascent is performed on it and LCE is the cross-entropy loss for the MGT task, and this loss is minimized.

$$L_{Final} = L_{CE} + \lambda L_{DL}$$

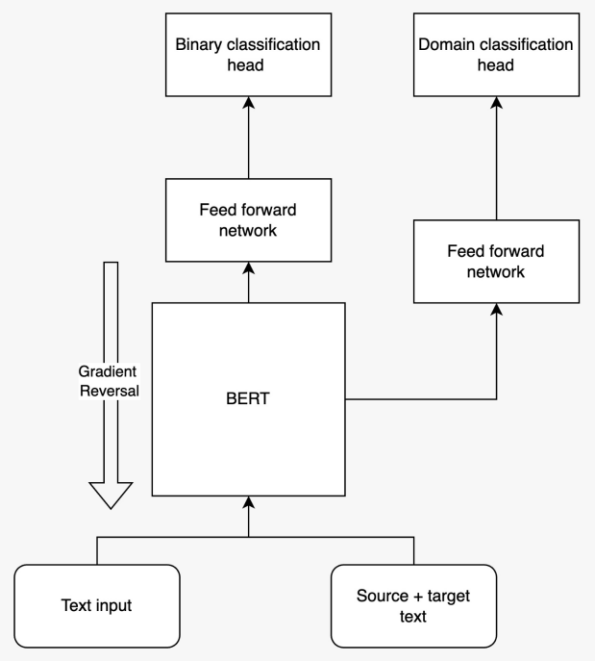

Fig 3: Unsupervised Domain Adaptation Through Backpropagation for BERT

Train Domain	Test Domain	Train Sequence Length	Test Sequence Length	Accuracy	F1 Score
Medicine	Wiki	128	128	0.5	0.33
Medicine	**Medicine**	**128**	**128**	**0.98**	**0.99**
Finance	Wiki	128	32	0.527	0.39
Finance	**Finance**	128	32	**0.9559**	**0.955**
Wiki	Reddit	128	128	0.5691	0.5674
Wiki	**Wiki**	128	128	**0.91**	**0.91**

Table 1: Some of the best and worst performance from 100 combination results (Baseline)

4. Experiment

As the dataset consists of data from 5 different domains, our goal is to check whether there is a lack of domain adaptation in the task of machine generated text. For the baseline, we did cross combination of 5 different domains with 2 different truncated sequence lengths (128, 32). Therefore, there are 100 combinations of train and testing data. That is there are 100 different models with 100 different results. For the supervised contrastive fine tuning of BERT, we did on the whole dataset without splitting it into different domains. And for the domain adaptation there are in total 80 different combinations of domain and sequence lengths. All the experiments were designed using PyTorch and HuggingFace frameworks.

5. Results

For the baseline experiment, when the train and test domain was same the model performed good. For most of the cross-domain training and testing the model performance dropped significantly. With the same domain the F1 scores ranges from 0.95 – 0.99, But in the cross domain the F1 score ranges from 0.5 – 0. 75.

In the supervised contrastive fine-tuning experiment on the entire dataset, the model achieved an accuracy of 0.6762. Further, in the experiment of unsupervised domain adaptation through backpropagation using BERT there are two hyperparameters, alpha (gradient reversal scaling factor) and beta (Domain loss scaling factor). With alpha = 1 and beta = 1, all the 80 combinations results were poor. Therefore, we fixed the train domain as Wiki and test domain as Reddit of same sequence length we performed hyperparameter search for alpha and beta. Even the best hyperparameter, the model achieved accuracy 5% less than the baseline and the f1 score was 45% lesser than the baseline.

Train Domain	Test Domain	Train Sequence Length	Test Sequence Length	Accuracy	F1 Score
Finance	Wiki	128	32	0.5	0.33
Wiki	**Reddit**	**128**	**128**	**0.2534**	**0.202**
Medicine	Wiki	128	128	0.5	0.33

Table 2: Some of the performance from 80 combination results (Bert with domain adaptation)

6. Discussion

It is evident that from the previous experiments, in supervised setting in the task of machine generated text detection, the method lacks data domain adaptability. Contrastive learning inherently has data domain adaptability but in the task of machine generated text detection Contrastive learning performed poorly. To check whether domain adaptability is possible in the task of machine generated text detection, we performed unsupervised domain adaptation through backpropagation for BERT. Even with the best hyperparameter setting domain adaptation is not possible in this task. Which indicates that in the supervised setting of machine generated text detection the models are learning contextual features instead of learning the necessary linguistic features for this task.

7. Conclusion and Future Works

In this project we explored using BERT for binary classification for machine generated text detection, supervised contrastive fine tuning of BERT for MGT detection. We observed that in both settings the models lack data domain adaptability. Therefore, we tried Unsupervised Domain Adaptation through backpropagation for BERT with the best hyperparameters. Further, we observed that data domain adaptability is impossible in the task of machine generated text in supervised setting as the model learns contextual feature representation instead of features that are necessary for the task of MGT detection. The future works is an open question which is What architectures people can come up with such a way that it learning the features that corresponds to machine and human generated text?

8. References

[1] Gehrmann, S., Strobelt, H. and Rush, A.M., 2019. Gltr: Statistical detection and visualization of generated text. arXiv preprint arXiv:1906.04043.

[2] Ippolito, D., Duckworth, D., Callison-Burch, C. and Eck, D., 2019. Automatic detection of generated text is easiest when humans are fooled. arXiv preprint arXiv:1911.00650.

[3] Gritsay, G., Grabovoy, A., & Chekhovich, Y. (2022, September). Automatic Detection of Machine Generated Texts: Need More Tokens. In 2022 Ivannikov Memorial Workshop (IVMEM) (pp. 20-26). IEEE.

[4] Kumarage, T., Garland, J., Bhattacharjee, A., Trapeznikov, K., Ruston, S. and Liu, H., 2023. Stylometric Detection of AI-Generated Text in Twitter Timelines. arXiv preprint arXiv:2303.03697.

[5] Liu, X., Zhang, Z., Wang, Y., Lan, Y. and Shen, C., 2022. CoCo: Coherence-Enhanced Machine-Generated Text Detection Under Data Limitation With Contrastive Learning. arXiv preprint arXiv:2212.10341.