Machine-Generated Text Detection

CSE847 Machine Learning: Course Project Final Report

JAVEN W. ZAMOJCIN, Michigan State University, USA
SIMON SITU, Michigan State University, USA

Large language models (LLMs) have been extremely effective for text prediction and generation in recent times. LLMs that have been fine-tuned to create assistant models has allowed many to automate certain tasks through a natural language interface. One problem that has come from the high performance and availability of LLMs is that text that has been generated from an LLM can be hard to distinguish from human text, especially in an automated fashion. Several negative consequences may result and have resulted from this, including misleading the public with misinformation, increasing automated bot activity on the internet, and LLM-assisted cheating in academia. These problems can be mitigated through the use of an accurate machine-generated text detector system. We aim to compare different approaches to the task of detecting text that has been generated from a language model.

ACM Reference Format:
Javen W. Zamojcin and Simon Situ. 2024. Machine-Generated Text Detection: CSE847 Machine Learning: Course Project Final Report. 1, 1, Article 1 (December 2024), 11 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

1.1 Motivation

The advent and dramatic innovations of LLMs and generative AI models in general, have signaled a transformative era in text generation, media, and information sharing. These models are capable of generating cogent, abstractive, and coherent text similar to human-written content. Online services such as ChatGPT have significantly improved the accessibility of LLM tools, bringing both their potential for boosting productivity and ethical concerns into the public consciousness. One such concern, and which is the focus of this research paper, is how to circumscribe whether text is machine generated (MGT) or written by humans.

LLMs can produce vast amounts of deceptive fake news, cluttering the information ecosystem. On social media platforms, LLMs can automate the creation of fake accounts ("bots") and the propagation of false information, amplifying the influence of the operators with minimal effort. In academia, these tools may extend beyond just content replication, calling into question the principles of scholarship integrity and undermining our traditional methods of detecting plagiarism. A possible consequence of this is a rise in automatically generated, low-quality papers being published in journals and conferences.

Authors' Contact Information: Javen W. Zamojcin, Michigan State University, East Lansing, Michigan, USA, zamojci1@msu.edu; Simon Situ, Michigan State University, East Lansing, Michigan, USA, situsimo@msu.edu.

ACM XXXX-XXXX/2024/12-ART1
https://doi.org/XXXXXXX.XXXXXXX

1.2 Related Works

Related works frequently have framed MGT detection as a binary classification problem [17] [1] [16] [4] [9], with a focus on the English language. The objective for this task is simply to categorize whether the given text is entirely human or machine generated. Furthermore, MGT detection tasks can be grouped into either supervised or unsupervised methods. Supervised approaches [15] [12] [17] [20] [6] rely on labeled datasets to train the classifiers, while unsupervised approaches [9] [4] [7] [3] [2] [5] [18] [19] identify MGT by analyzing white-box features such as likelihood, log-rank, and employing watermarking techniques [14].

There is also work being done to identity not just whether text was machine-generated or not, but additionally what language model generator was used for the given text [11] [13] [10] [8]. This multi-class classification task is closely related to the author attribution problem; distinguishable signals exist within LLM generated text that help identify the generator source [8]. This is usually broken into the sub-problems of identifying whether two texts were produced by the same generator source, the binary classification task, and identifying what LLM was the generator source [11].

Wang et al. (2024) [14], the researchers who created the M4GT dataset benchmark used in this work, in addition to experimenting with binary and multi-class classification tasks, introduce the task of detecting the change point from human-written to machine-generated text. Specifically, they formulated the problem using documents that were first up to `50%` human-written, followed by a discrete point where the remaining document text was machine-generated. Their motivation for introducing this task was citing that most previous studies overlooked the fact that documents can be a mixture of machine-generated and human-written.

2 Problem description

The task of detecting machine-generated text can be formulated in multiple ways, at various levels of granularity. The data set we use for training models for this MGT task, M4GT benchmark [14] has three different sub-tasks: a binary task for human and non-human detection, a multi-class task for determining the specific model or human that generated the text, and a task for determining the boundaries between human-generated text and machine-generated text within a larger piece of text. In the scope of this work, we just experiment on the first two sub-tasks, binary and multi-class classification.

The M4GT (Multi-lingual, Multi-domain, and Multi-generator corpus of MGTs) benchmark contains a total of `138465` samples, `65177` of which are human-written documents and `73288` are machine-generated. The document source domains include `Wikipedia`, `Wikihow`, `Reddit ELI5`, `arXiv abstracts`, and `PeerRead`. The LLM source generators include `davinci-003`, `ChatGPT`, `Cohere`, `Dolly-v2`, and `BLOOMz`, with each contributing approximately `~15000` samples. Minimal data preprocessing was done to the dataset in order to mitigate removing features that are key to signaling the generator source or that might be particular to human writings. The dataset also includes multi-lingual sub-sets, but for this research we only experiment using the English documents.

3 Methodology

We trained three different models for both the binary and multi-class classification tasks. The models we used are a Support Vector Machine (SVM), a Long-Short Term Memory Neural Network (LSTM), and a fine-tuned DistilBERT model. The data is made up of 10 sources of text generation, including human-made text [Figure 1]. For the multi-class task, the different versions of the same model are grouped together, yielding 7 classes in total [Figure 2].

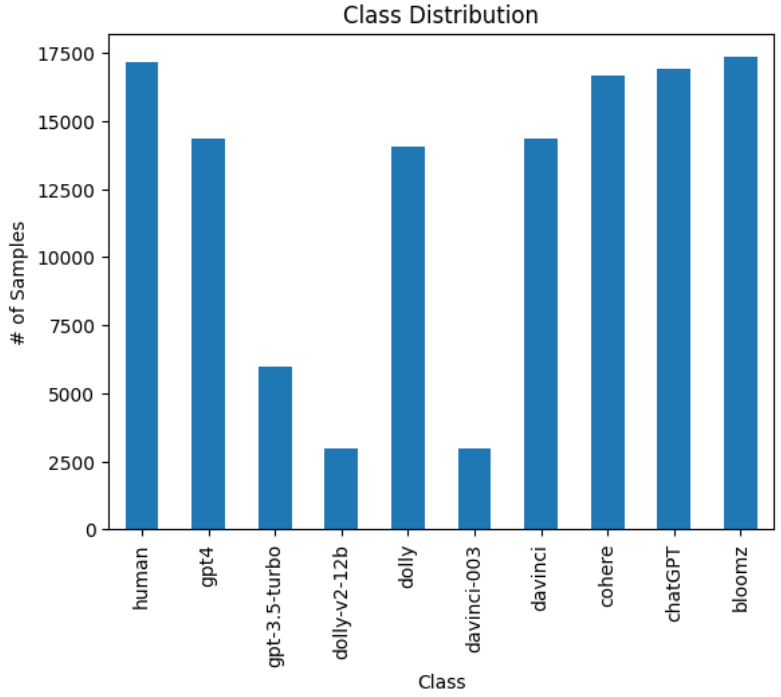

Fig. 1. Distribution of data using 10 classes.

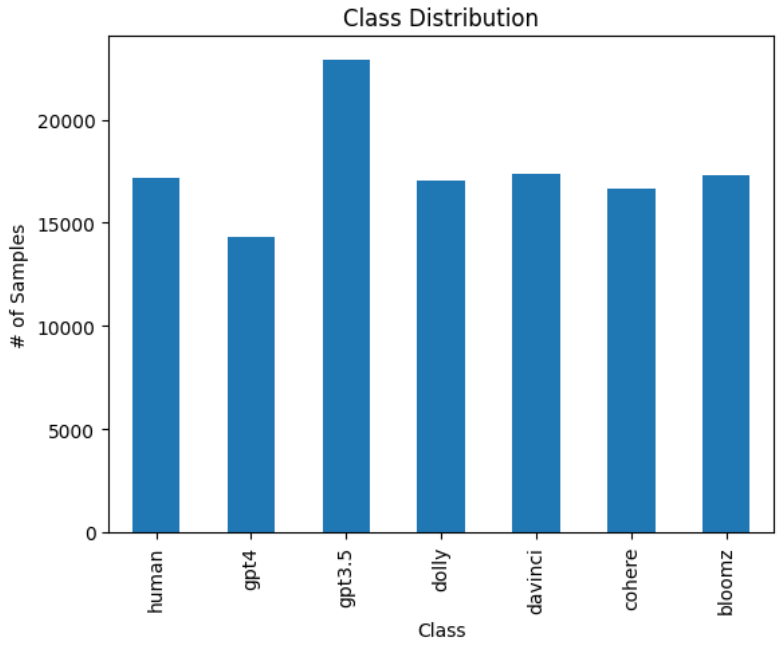

Fig. 2. Distribution of data using 7 classes.

3.1 SVM

The first model we experimented on was the Support Vector Machine (SVM). For converting the text into numerical features used in both tasks, we used count vectorization and TF-IDF (Term Frequency-Inverse Document Frequency) techniques. The parameters used were 3000 for maximum features (unique words), a minimum document frequency of 2, a maximum document frequency of 0.7 (percentile), a unigram word analyzer, a linear kernel, a dynamic number of max iterations, and no removal of stop words. Additionally, 3-fold cross validation was used and the resulting metrics (accuracy, recall, precision, F1) were averaged across the folds. The specific SVM classifier used was the Sklearn library's SVC implementation.

3.2 LSTM

The next model we experimented on was the Long-Short Term Memory Neural Network (LSTM). For this model we used the Keras deep-learning framework with a Pytorch back-end. For preprocessing, we used the Keras `TextVectorization` class with the parameters of `10000` maximum unique tokens, lowercase and punctuation stripping for standardization, a maximum padded sequence length of `200` tokens, whitespace delimitation, and unigram tokenization. For both tasks, a Keras Sequential model with an embedding layer with input dimensions of `10000` (maximum unique tokens) and output dimensions of 128, an LSTM layer with a hidden size of `100`, and a final dense layer with an output size matching the number of classes for the respective task (2 or 7). We do not use softmax and worked with the logits instead. Additionally, we used an `Adam` optimizer, `SparseCategoricalCrossEntropy` for the loss function, a fitting batch size of `64`, and a dataset train-test split of `0.85/0.15`. The binary classification task used `5` training epochs and the multi-class task used `10` training epochs.

3.3 DistilBERT

And the last model we experimented on was DistilBERT. The pretrained model we imported from HuggingFace had a model ID of `distilbert-base-uncased`, and was configured for our task of sequence classification for either 2 or 7 class labels. For tokenization, we used the `Transformers` library's `AutoTokenizer` configured for our specific model, with a maximum length padding scheme and truncation enabled. For splitting the dataset into training, validation, and test subsets, we used splits of `0.80/0.10/0.10` respectively. Additionally, due to resource limitations, only a subset size of `10000` was used for the training samples and `1000` for both the validation and testing subsets. For both tasks, `3` training epochs were used for the fine-tuning phase.

The model architecture for both tasks included a word embeddings layer with dimensions (`30522, 768`), a positional embeddings layer with dimensions `(512, 768)`, a normalization layer, and a dropout layer with probability `p=0.1`. Following these layers were the `6` transformer blocks each consisting of an attention layer (`768x768` for each Q, K, V, and Out matrices), a normalization layer, a forward-feed network layer (`768x3072` + `3072x768`) with `GELU` activation, and finally two sequential linear layers (`768x768` + `768xnum_labels`).

4 Results

After training each model on each task, we evaluated each on testing data and obtained the respective confusion matrices. As for the SVM, we averaged the confusion matrices over the folds. The primary metric to consider is the recall of the human class. This measures the rate the model will incorrectly classify a human sample as a machine-generated sample. Such detection systems typically want to minimize the rate of false detections, so we will focus on this metric in our analysis.

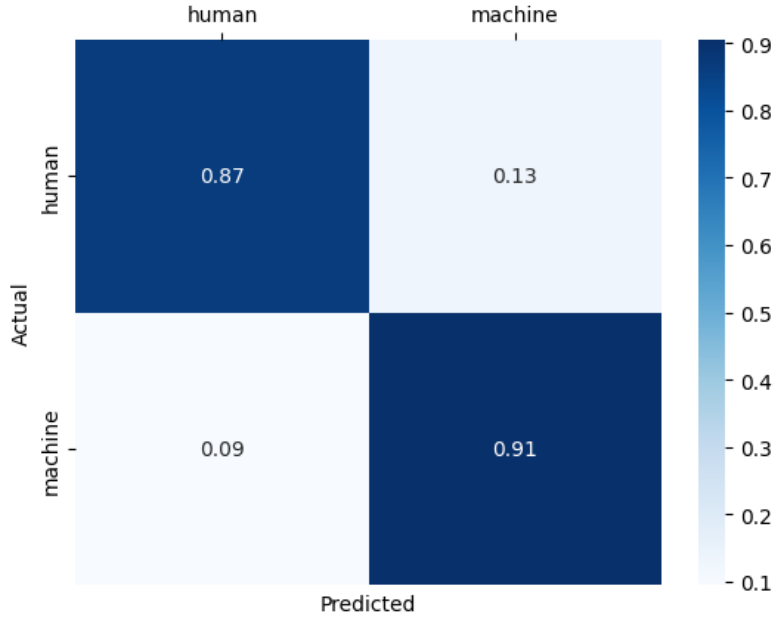

Fig. 3. Confusion matrix for the SVM model on the binary data.

For each model, the recall of the human class when trained on the multi-class task is consistently lower than their binary counterparts. The likely reason for this is that the multi-class task is more complex than the binary task, and we do not significantly increase the model size or training data size to account for the added complexity.

The LSTM model obtained the highest accuracy for both tasks, and also had the highest recall in the human class. The LSTM model has advantages over the SVM model, since it is designed for sequential data and also has built-in non-linearity. We found it surprising that the finetuned DistilBERT model did worse than both of the other models. We found it to be extremely easy to overfit, and training it using more samples beyond the 1000 sample subset that we sampled from the training set did not improve the model on the test set.

The binary SVM coefficients can be used to provide insights into the predictive features of the data. The negative weights correspond to the features that are more likely to be predictive of the negative class (human) and the positive weights correspond to predictive features of the positive class (machine-generated). The features of the SVM are the TF-IDF values, which correspond to words in the data. We visualized this using separate bar charts for the positive and negative classes [Figure 9]. The chart shows the words that tend to be used in the text of each respective class. There are artifacts such as "endoftext" which is a token used by the `Cohere` model, and "_url_0_" which is likely from the process of scraping the sources for human-written text.

Class	Precision	Recall	F1 Score
human	0.8718	0.8670	0.8694
machine	0.9015	0.9051	0.9033

Table 1. Metrics for the binary SVM classifier. Accuracy: 88.89%

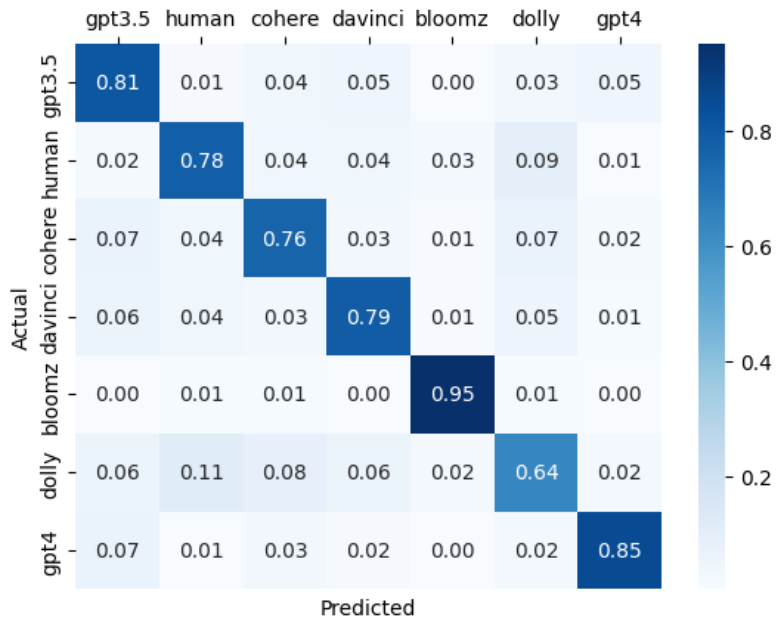

Fig. 4. Confusion matrix for the SVM model on the multi-class data.

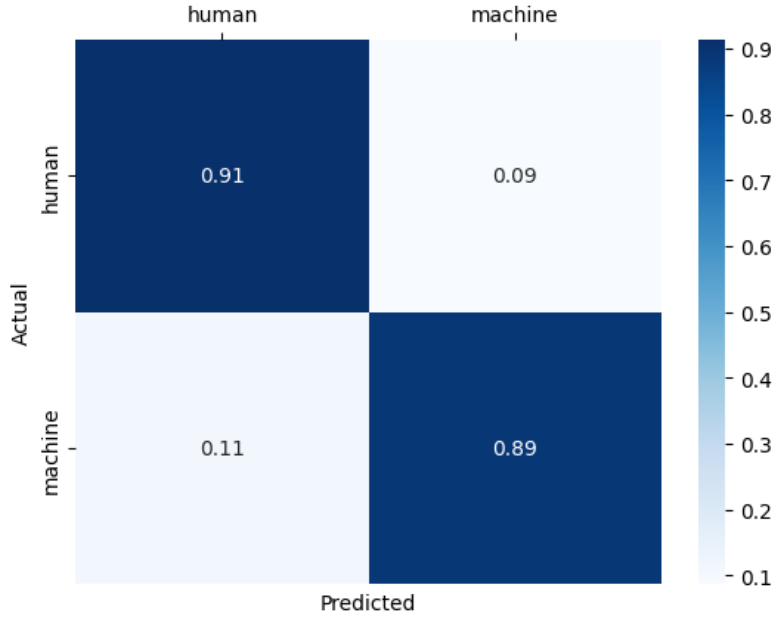

Fig. 5. Confusion matrix for the LSTM binary classifier.

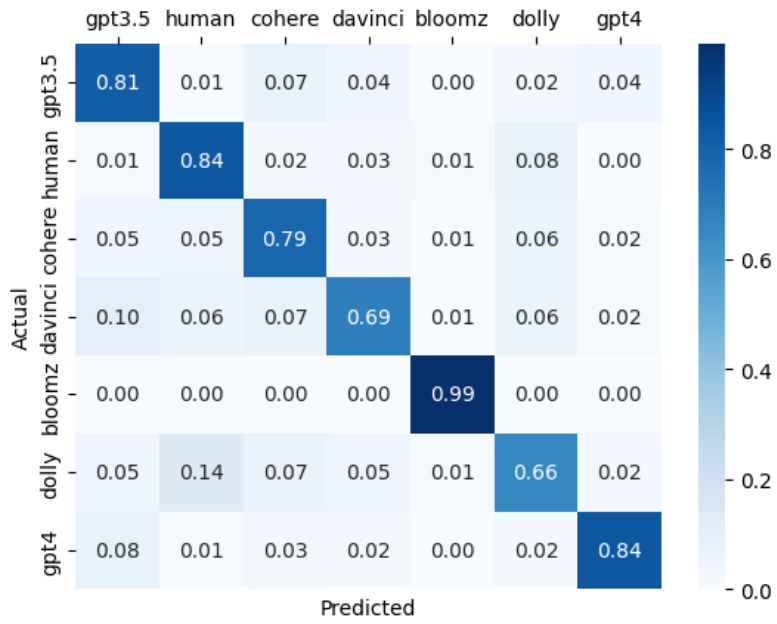

Fig. 6. Confusion matrix for the multi-class LSTM classifier.

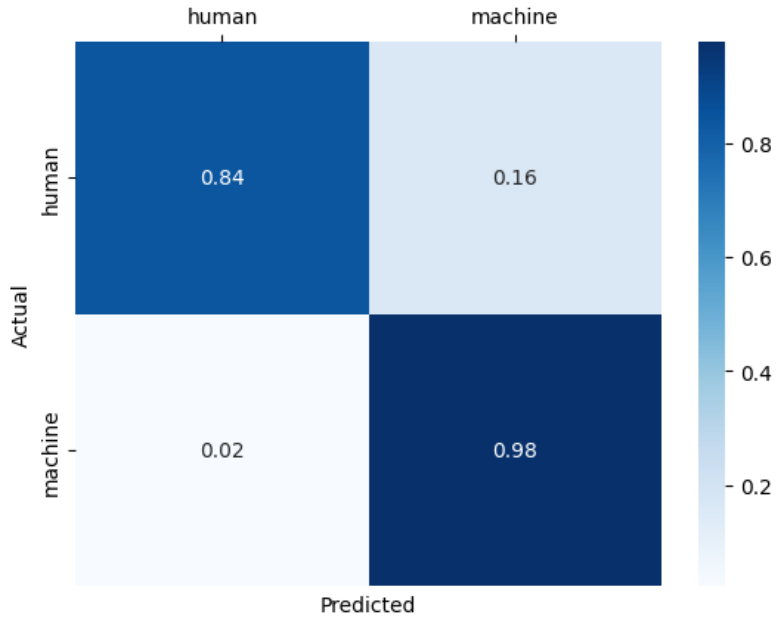

Fig. 7. Confusion matrix for the binary DistilBERT model.

Class	Precision	Recall	F1 Score
gpt3.5	0.8082	0.8087	0.8084
human	0.7708	0.7780	0.7744
cohere	0.7428	0.7595	0.7511
davinci	0.7888	0.7903	0.7895
bloomz	0.9256	0.9519	0.9386
dolly	0.6892	0.6400	0.6637
gpt4	0.8440	0.8523	0.8481

Table 2. Metrics for the multi-class SVM classifier. Accuracy: 79.70%

Class	Precision	Recall	F1 Score
human	0.8611	0.9138	0.8867
machine	0.9331	0.8908	0.9114

Table 3. Metrics for the binary LSTM classifier. Accuracy: 90.06%

Class	Precision	Recall	F1 Score
gpt3.5	0.8013	0.8146	0.8079
human	0.7519	0.8404	0.7937
cohere	0.7255	0.7855	0.7543
davinci	0.7861	0.6915	0.7357
bloomz	0.9604	0.9920	0.9760
dolly	0.7379	0.6594	0.6964
gpt4	0.8682	0.8425	0.8552

Table 4. Metrics for the multi-class LSTM classifier. Accuracy: 80.38%

Class	Precision	Recall	F1 Score
human	0.9656	0.8372	0.8968
machine	0.8918	0.9783	0.9330

Table 5. Metrics for the binary DistilBERT classifier. Accuracy: 91.88%

Class	Precision	Recall	F1 Score
gpt3.5	0.5248	0.9671	0.6803
human	0.9531	0.8164	0.8795
cohere	0.9375	0.4169	0.5772
davinci	0.6639	0.5591	0.6070
bloomz	0.8952	0.9948	0.9424
dolly	0.6737	0.6737	0.6737
gpt4	0.9084	0.5010	0.6459

Table 6. Metrics for the multi-class DistilBERT classifier. Accuracy: 72.21%

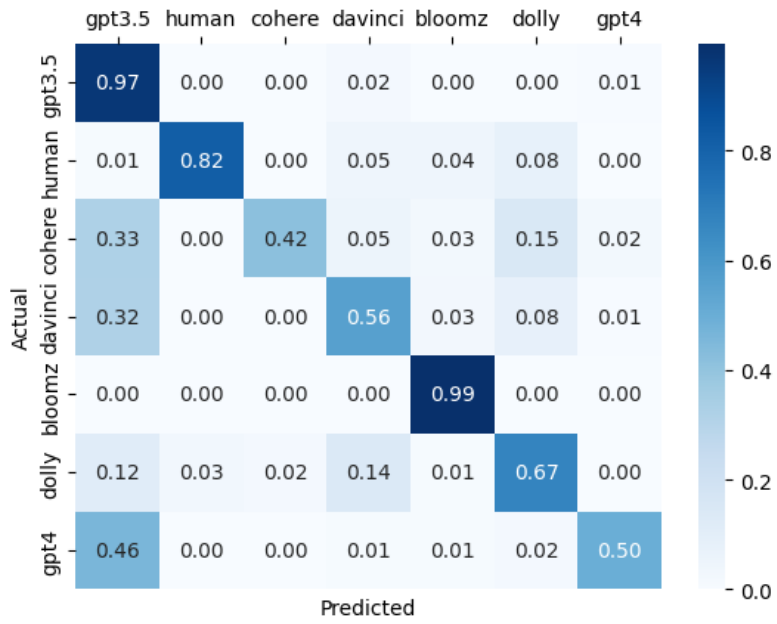

Fig. 8. Confusion matrix for the multi-class DistilBERT model.

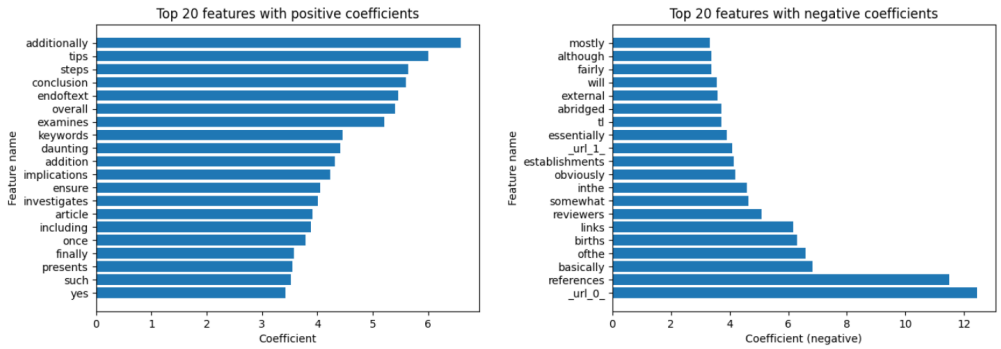

Fig. 9. Most important words based on the largest positive and negative coefficients of the binary SVM model corresponding to the respective TF-IDF features.

5 Conclusion

In conclusion, we investigated the problem of detecting machine-generated text. We explored two separate sub-problems: binary classification of whether text was completely machine-generated or human-written, and multi-classification of not only whether text was completely machine-generated, but also which LLM was the source generator. On both of these sub-tasks, we implemented, trained, and evaluated three different machine-learning models: linear SVM, LSTM, and DistilBERT.

Our findings were that with the given conditions, text can be correctly classified as either completely machine-generated or human-written with a fairly high-degree of accuracy, especially when using fine-tuned transformer-based language models. The task of multi-class authorship

classification was a harder problem with significantly lower accuracy reliability. The biggest limitations of our work are how well our models can generalize to data generated by unseen sources, and overlooking the fact that documents may consist of a non-uniform mixture of machine-generated and human-written texts, not necessarily just one or the other.

6 Future Work

There are a number of further experiments we would like to try in the future. Firstly, the linear kernel SVM model produced surprisingly good results with just TF-IDF vectorization features. We would like to see the effect of using a non-linear kernel for the SVM to obtain a performance gain. We could also explore the addition of other features for text data such as Word2Vec or BERT embeddings. The results that we obtained for the DistilBERT model could likely be improved through more hyperparameter tuning and experimentation. Another potential improvement to make is the removal of the artifacts found during the exploration of feature importances using the SVM model. These artifacts likely skewed our results, so the removal of these from the dataset would improve the quality of our results and also improve the data quality. We would also like to see these models trained to predict the authorship boundaries in a text with two different generators, such as the case of augmenting human text using language models, and additionally elaborate upon the potential task complexity by allowing for multiple or no change points within each text document.

Acknowledgments

To the Wang et al. research team, who curated the M4GT benchmark dataset.

References

[1] Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical Detection and Visualization of Generated Text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Marta R. Costa-jussà and Enrique Alfonseca (Eds.). Association for Computational Linguistics, Florence, Italy, 111–116. https://doi.org/10.18653/v1/P19-3019

[2] Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2024. Spotting LLMs With Binoculars: Zero-Shot Detection of Machine-Generated Text. arXiv:2401.12070 [cs.CL] https://arxiv.org/abs/2401.12070

[3] Xinlei He, Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2024. MGTBench: Benchmarking Machine-Generated Text Detection. arXiv:2303.14822 [cs.CR] https://arxiv.org/abs/2303.14822

[4] Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic Detection of Generated Text is Easiest when Humans are Fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 1808–1822. https://doi.org/10.18653/v1/2020.acl-main.164

[5] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2024. A Watermark for Large Language Models. arXiv:2301.10226 [cs.LG] https://arxiv.org/abs/2301.10226

[6] Xiaoming Liu, Zhaohan Zhang, Yichen Wang, Hang Pu, Yu Lan, and Chao Shen. 2023. CoCo: Coherence-Enhanced Machine-Generated Text Detection Under Data Limitation With Contrastive Learning. arXiv:2212.10341 [cs.CL] https://arxiv.org/abs/2212.10341

[7] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature. arXiv:2301.11305 [cs.CL] https://arxiv.org/abs/2301.11305

[8] Shaoor Munir, Brishna Batool, Zubair Shafiq, Padmini Srinivasan, and Fareed Zaffar. 2021. Through the Looking Glass: Learning to Attribute Synthetic Text Generated by Language Models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty (Eds.). Association for Computational Linguistics, Online, 1811–1822. https://doi.org/10.18653/v1/2021.eacl-main.155

[9] Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine

Wang. 2019. Release Strategies and the Social Impacts of Language Models. arXiv:1908.09203 [cs.CL] https://arxiv.org/abs/1908.09203

[10] Rafael Rivera Soto, Kailin Koch, Aleem Khan, Barry Chen, Marcus Bishop, and Nicholas Andrews. 2024. Few-Shot Detection of Machine-Generated Text using Style Representations. arXiv:2401.06712 [cs.CL] https://arxiv.org/abs/2401.06712

[11] Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship Attribution for Neural Text Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 8384–8395. https://doi.org/10.18653/v1/2020.emnlp-main.673

[12] Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon Lee. 2021. TURINGBENCH: A Benchmark Environment for Turing Test in the Age of Neural Text Generation. arXiv:2109.13296 [cs.CL] https://arxiv.org/abs/2109.13296

[13] Saranya Venkatraman, Adaku Uchendu, and Dongwon Lee. 2024. GPT-who: An Information Density-based Machine-Generated Text Detector. arXiv:2310.06202 [cs.CL] https://arxiv.org/abs/2310.06202

[14] Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Osama Mohanned Afzal, Tarek Mahmoud, Giovanni Puccetti, Thomas Arnold, et al. 2024. M4GT-Bench: Evaluation Benchmark for Black-Box Machine-Generated Text Detection. *to appear in ACL 2024* (2024).

[15] Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, Thomas Arnold, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024. M4: Multi-generator, Multi-domain, and Multi-lingual Black-Box Machine-Generated Text Detection. arXiv:2305.14902 [cs.CL] https://arxiv.org/abs/2305.14902

[16] Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, Thomas Arnold, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024. M4: Multi-generator, Multi-domain, and Multi-lingual Black-Box Machine-Generated Text Detection. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, Yvette Graham and Matthew Purver (Eds.). Association for Computational Linguistics, St. Julian's, Malta, 1369–1407. https://aclanthology.org/2024.eacl-long.83

[17] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2020. Defending Against Neural Fake News. arXiv:1905.12616 [cs.CL] https://arxiv.org/abs/1905.12616

[18] Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. 2023. Provable Robust Watermarking for AI-Generated Text. arXiv:2306.17439 [cs.CL] https://arxiv.org/abs/2306.17439

[19] Xuandong Zhao, Yu-Xiang Wang, and Lei Li. 2023. Protecting Language Generation Models via Invisible Watermarking. arXiv:2302.03162 [cs.CR] https://arxiv.org/abs/2302.03162

[20] Wanjun Zhong, Duyu Tang, Zenan Xu, Ruize Wang, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Neural Deepfake Detection with Factual Structure of Text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 2461–2470. https://doi.org/10.18653/v1/2020.emnlp-main.193

Received 03 December 2024