Title: Different Learning Methods for AI-Generated Text Detection
Name: Ng Man Tik

Introduction

The advent of generative models, particularly ChatGPT and GPT-4, marks a significant evolution in artificial intelligence, profoundly impacting various fields such as academic writing, story generation, and software development [1-5]. The capabilities of Large Language Models (LLMs) have evolved to produce text nearly indistinguishable from human writing, as evidenced by recent studies [8]. However, this technological leap brings forth new challenges, notably the difficulty in distinguishing between AI-generated and human-authored texts. This ambiguity raises concerns regarding information quality—given LLMs' dependency on potentially outdated or biased datasets—and the potential for misuse in areas like fake news dissemination and academic dishonesty [9-11].

Current research efforts have focused on developing methods to detect machine-generated content, typically through fine-tuning existing language models with extensive datasets. However, these approaches often overlook the nuanced reality where texts are neither purely machine-generated nor entirely human-written, failing to reflect the complex interactions between AI and human input in real-world applications.

This project aims to bridge this gap by advancing the detection of AI-generated texts while accounting for the hybrid nature of contemporary written content. We propose a novel approach integrating Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), and Attention mechanisms, targeting the nuanced differences between AI-generated and human texts. This methodology not only seeks to improve detection accuracy across various text lengths and sources but also aims to effectively handle texts that blend AI and human contributions.

The significance of this research extends beyond the academic and creative writing spheres, touching on broader societal implications such as ethical standards, copyright laws, and information transparency. By developing more reliable methods to identify AI-generated content, this project contributes to the ongoing discourse on the role of AI in content creation, addressing concerns surrounding authenticity and human creativity in the digital age. Through this endeavor, we aim to provide actionable insights and tools to navigate the evolving landscape of AI-generated content responsibly.

Related Work

Text Classification in Natural Language Processing: Text classification serves as a cornerstone in the field of Natural Language Processing (NLP), essential for tasks ranging from sentiment analysis to fake news detection. Traditionally, this field has relied on machine learning techniques like Naïve Bayes, Decision Trees, and Support Vector Machines (SVMs), utilizing feature extraction methods such as bag-of-words or TF-IDF [12-14]. However, the emergence of Large Language Models (LLMs) like ChatGPT has shifted the paradigm, making the detection of machine-generated text increasingly complex due to their advanced human-like writing styles. This transformation underscores a critical challenge: distinguishing between human and LLM-generated texts, which have become remarkably similar, blurring the lines of authorship [15-17].

Evolution of Models - LSTM, CNN, and Hybrid Approaches: In the domain of text classification, the introduction of deep neural networks such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) has initiated a paradigm shift from traditional feature extraction methods to more dynamic and intuitive analysis processes. Unlike earlier methods that relied heavily on manual feature engineering, CNNs and RNNs have paved the way for automatic pattern recognition in text, significantly enhancing the efficiency and effectiveness of text classification. Specifically, CNNs excel in identifying local textual features, providing detailed insights into the structure and composition of the text, while RNNs, especially Long Short-Term Memory networks (LSTMs), are adept at understanding sequence dependencies, capturing the temporal and contextual nuances of written language [18].

The convergence of CNNs and RNNs has led to the development of hybrid models that combine the strengths of both architectures, offering a more comprehensive approach to text analysis. This integrated approach significantly improves feature extraction within text sequences, leading to enhanced model accuracy and interpretability [19-25]. Innovations in this area, such as those by G. Wang et al., have introduced methods for embedding label sets into vector spaces, facilitating more effective computation and analysis of text data [26]. Additionally, the integration of self-attention and label-embedding techniques, as demonstrated by Dong, Y et al. [27] and Y. Xiao et al. [28], further enriches model capabilities, enabling more focused and relevant analyses of textual content.

One notable application of this combined approach is the utilization of CNN and Attention mechanisms as encoders with BiLSTM decoders. This configuration has been effectively applied in scenarios such as stock prediction, showcasing the model's ability to interpret complex sequential data accurately. Similarly, the amalgamation of CNNs and BiLSTMs with interactive Attention mechanisms has proven beneficial in critical areas like fake news detection, underlining the model's proficiency in identifying subtle semantic nuances and patterns within texts.

In our project, we intend to further refine this combined model architecture by integrating CNNs with BiLSTMs and embedding strategic attention layers to enhance feature extraction and interpretation. This refined approach is specifically aimed at improving the model's ability to differentiate between texts generated by humans and those produced by LLMs, a task growing ever more challenging with the advancing capabilities of modern language models. By fine-tuning the interaction between these model components and adjusting their configurations, we anticipate not only higher accuracy in text classification but also a deeper insight into the distinguishing characteristics of human versus LLM-generated texts.

Fine-tuning LLMs for Text Classification: The advent of transformer architectures has introduced a new frontier in NLP, with models like BERT, Roberta, and XLNet setting new benchmarks in text understanding and classification [29-33]. These models' fine-tuning, particularly for tasks like distinguishing LLM-generated texts, has shown promising results. However, the computational demand of these models poses a significant barrier for individuals with limited resources. Our work seeks to address this by employing a smaller, more efficient model, Mistral-7B, leveraging techniques such as LoRA to enable fine-tuning with reduced resource requirements [34].

Addressing Potential Attacks on Text Classification Models:
Despite achieving high performance in identifying machine-generated texts, models remain susceptible to various adversarial attacks that could significantly impair their effectiveness. Recent research highlights that even high-performing models can falter when confronted with specific, subtly altered inputs. For instance, the application of a lightweight paraphrase model to alter the wording and semantic distribution of machine-generated texts has demonstrated potential in undermining zero-shot detection capabilities [35-36]. This reveals the models' vulnerability to nuanced changes that preserve meaning while altering textual structures.

Further complicating the landscape, Shi et al. [37] and He et al. [38] have

documented the efficacy of permutation strategies in deceiving text detection systems. Techniques such as content cutoff [39], sequence shuffling [40], token mutation [41], and strategic word swapping [42] pose significant challenges, indicating that these methods can effectively mask the machine-generated nature of texts, thereby evading detection by otherwise robust models.

In response to these challenges, our project plans to leverage the MixSet dataset [43], renowned for its incorporation of texts that blend human and machine elements. This dataset serves as a critical resource for simulating real-world applications, where texts often exhibit characteristics of both human and AI contributions. By employing this dataset, we aim to evaluate and enhance the resilience of our models against a range of adversarial tactics. Specifically, we will investigate the model's performance against paraphrased outputs—a common form of attack aiming to 'humanize' machine-generated content. This approach will not only test the models' detection capabilities under manipulated conditions but also contribute to the ongoing discourse on securing AI-driven text analysis tools against emerging threats.

Data

This study leverages three primary datasets, each offering unique insights and challenges relevant to distinguishing between AI-generated and human-written texts. Below is a detailed exploration of these datasets:

1. M4 Dataset [44]:

- **Source**: The M4 dataset is compiled from a variety of sources, including Wikipedia, Reddit's Explain Like I'm Five (ELI5), WikiHow, PeerRead, and academic papers from arXiv, reflecting a wide range of topics and writing styles.
- **Size & Composition**: It consists of 122,481 entries, providing a substantial volume for training and validating the models. The dataset's diversity enables the examination of text classification across different domains.
- **Usage**: In our project, we focus specifically on the English-language texts and select content relevant to defined fields from the various sources. This approach ensures the data's relevance to our research objectives.
- **Challenges**: The dataset's heterogeneity, due to its multi-source nature, may introduce variability in writing style and complexity. This diversity, while beneficial for model robustness, might complicate the training process and the model's ability to generalize across different text types. Moreover,

number of human-written text and LLM-generated text is in 1:x where x is number of LLM in the dataset, which will show the imbalanced if all of the data loaded during training.

2. MixSet Dataset:

- **Source**: This dataset contains mixed instances of AI-generated and human-edited texts from real-life scenarios, such as emails and game reviews, divided into six categories.
- **Size & Composition**: With 3,600 entries, the MixSet offers a focused environment to test the model's ability to identify texts featuring varying degrees of AI and human input.
- **Usage**: The entire dataset will be employed to challenge the detection capabilities of the models against mixed content, providing insights into their performance in real-world scenarios.
- **Challenges**: The primary difficulty lies in the mixed nature of the texts. Standard models trained on purely AI-generated or purely human-written texts may falter when faced with these hybrid examples, necessitating a nuanced approach to training and evaluation.

3. MGTBench Dataset [45]:

- **Source**: Featuring essays, stories, and articles generated by both humans and seven different AI models, MGTBench is designed to assess how well models can distinguish between various origins of text.
- **Size & Composition**: The dataset includes 24,000 entries, offering a wide range of text lengths and complexities for comprehensive testing.
- **Usage**: Initially, the project will utilize texts generated specifically by the GPT-4ALL model. This focus may expand to include outputs from other AI models to enhance the robustness and generalizability of our detection techniques.
- **Challenges**: Variability in text quality and length presents a significant hurdle, potentially affecting the consistency of model training and evaluation. Addressing these variations is crucial for accurate performance assessment across different text formats and sources.

Approach

1. Data Preparation

Before loading the datasets for training and testing process, prepare the data from the dataset first. It first needs to remove all the dataset where its language is not English in the dataset. Then, it needs to read different json files and then copy the machine-text or human-text with manually labeling as 0 if it is human-text and 1 if machine-text. After that, we need to clean the data by removing rows with missing

values in the ‘text’ column to ensure data quality. Finally, the result data need to be separated into the training, validation and test dataset in 8:1:1 ratio for loading the datasets.

2. Text Processing

For the text processing, we applied standard natural language processing techniques taught in lecture because the feature engineering part will be performed by CNN and Attention in the model. We first tokenized the textual content using a basic English tokenizer for splitting text into words and tokens while removing all the punctuation and stop words then construct a vocabulary with 10000 most frequent tokens to reduce computational complexity and memory requirements. Finally, I encoded the text by replacing the token to index for processing by neural networks with padding and truncation to standardizing the sequences with 200tokens only.

3. Model Architecture

In the following part, I will explain the following models used: SVM, CNN-BiLSTM with attention and Mistral-7B

3.1 SVM

3.1.1 General Model Architecture

To task the performance of those traditional machine-learning method for classification of LLM-generated text with human text, I incorporated the SVM for text classification.

Before deploying the model, the preprocessed the data will further employs the TF-IDF to converts the text data into a matrix of TF-IDF features to enabling the SVM to know the textual features. The TF-IDF vectors is set with the maximum length of 1000. After that, the radius basis function will be act as the kernel of SVM and do the classification.

3.2 Comparable Baselines

The model will be comparable with the same SVMs, but with different kernels which are linear, sigmoid and polynomic to understand how well each variant perform across the same text sources.

3.2 CNN-BiLSTM with double attention

3.2.1 Embedding layer

The first layer is the embedding layer, after loading the preprocessed text, it will map

each token to a high-dimensional vector using one-hot encoding to put them into dense representations to capture semantic properties such similar vectors for later layers to understand the content.

3.2.2 Convolutional Layers

After converting to the word vectors, it will be applied into multiple convolutional layers. Each of them consists of a set of filters or kernels that slide across the word vectors to detect specific features or patterns at different position of the text. Assume the input text is with dimension d, those vectors will form an input matrix with dimensions corresponding to the sequence length and vector size which is L x d. Then, the matrix can be processed by the multi-channel convolutional layer that employs kernels of varying size in 2, 3, and 4 words to capture different local textual features. Those kernels will focus on different n-gram combinations while the global ax pooling reduces the feature map into condensed representation.

3.2.3 Pre-LSTM Attention

After generating the feature maps using CNN layers, attention mechanism is used to aim to weigh the importance of different n-gram features extracted by the CNN layers before they are processed by the LSTM which can focus on more relevant features extracted from the convolutional layers. It does so by computing a weight sum of the features based on the attention weights, resulting in an attended feature vector which represents a focused summary of the most relevant features.

3.2.4 Bi-directional LSTM

Receiving the attended feature vector from the Pre-LSTM Attention. This layer can be able to capture the long-term dependencies within sequence data with the additional information the feature sequences. The layer is using bi-directional version of LSTM to capture the context from both sides due to the fact that the meaning or choice of wording should be depend form both sides not just words before it. It can thus offer a more complete understanding of each word within its surrounding context.

3.2.5 Post-LSTM Attention Mechanism

The Post-LSTM attention is applied to focus on local text features before sequence processing, the Post-LSTM Attention assesses the importance of different parts of the text after considering its full context. This attention step assigns weights to each position in the BiLSTM's output sequence, identifying which parts are most relevant for the classification decision. Using it can emphasize the most informative parts of

the texts given from the output of BiLSTM layer to make the final classification decision. The result will finally input to the fully connected layer to give the binary classification of human (0) or machine-text (1).

3.2.6 Comparing Baselines

For the comparison experiment, the following models with similar complexity were used:

- Ordinary version of BiLSTM: Using BiLSTM directly in extracting the sequential dependencies of the sequences for classification
- Attention with BiLSTM: Attention is appended after BiLSTM for further focusing the important features in the result produced in BiLSTM
- CNN with BiLSTM: CNN is performed in feature extraction before doing the classification with BiLSTM
- CNN-BiLSTM-Attention: A similar approach to the proposed model but removing the attention layer between CNN and BiLSTM to test the effectiveness of that layer.

3.3 Minstra-7B-bnb-4bit

3.3.1 General Model Architecture

Another model going to be used is Mistral 7B, according to their paper released, this model completely outperforms Llama 2 13B, the popular model in LLM field on all benchmarks, and even outperforms Llama 34B on many benchmarks, showing the small LLM's ability is comparable with large ones with proper settings. In the project, the Mistral-7B's variant which is block-wise model-update Filtering and Bit-centering (BNB) is used for enhancing the model efficiency and memory usage. Moreover, the quantized 4bits version would be used for reducing the model's size and can be trainable even in T4 GPU.

3.3.2 Training Setting

In the project, it will use 'FastLanguageModel' from UnSLoth library for downloading the model and setting the maximum sequence for up to 2048 tokens. Meanwhile, LoRA would be used to train only 4% of its parameters with gradient accumulation and precision training.

After the data loaded, rather applying standard natural language processing techniques like what I did in SVM and LSTM model, the Supervised Fine-Tuning method would be performed where the text data and their labels will be structured as a suitable prompt format for the model retraining on the Machine-text

classification task.

3.3.3 Comparing Baseline

A zero-shot version of another LLM model Roberta would be compared by directing inputting the test data into the model to classify the task.

4. Loss Functions

Since with the nature of the M4 dataset, the data is in the form of 1 human written text with several machine-generated text from various models over certain topic, loading it will inevitably have the imbalanced data issue, leading those models learning towards the majority class only. In view of this, for the loss function used, rather than the traditional Binary classification loss, the Focal Loss [46] will be used as the loss function.

Focal Loss is designed to modulate the contribution of each example to the loss based on the classification error. The key idea is to focus training more on hard-to-classify examples and reduce the relative loss for well-classified instances. It is an extension of standard Cross-Entropy Loss and we can enhance the sensitivity of the models towards minority classes and improve the overall balance in performance across different classes.

5. Evaluation Metrics

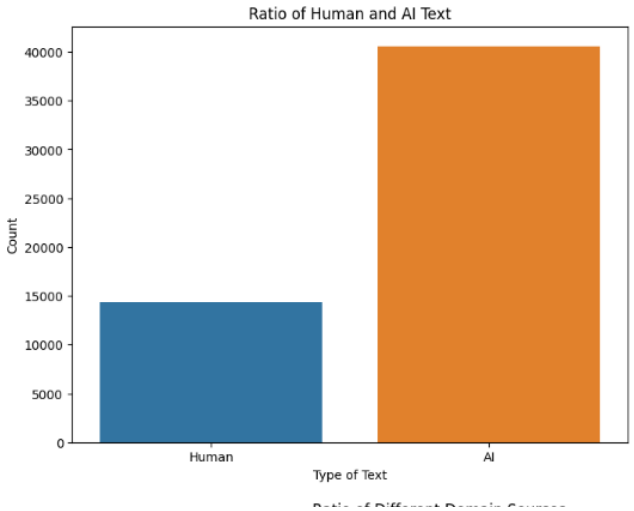
In the testing process, I will record the model different metrics like the accuracy, F1 score, recall and the auc for knowing the overall performance of the model. Meanwhile, a domain-specific metric will be recorded to understand the model's power in different source of the domain. Moreover, the data of its performance of identifying the text generating from different model will also be recorded to know the performance over it, especially on the human-text.

Preliminary Results

Data distribution

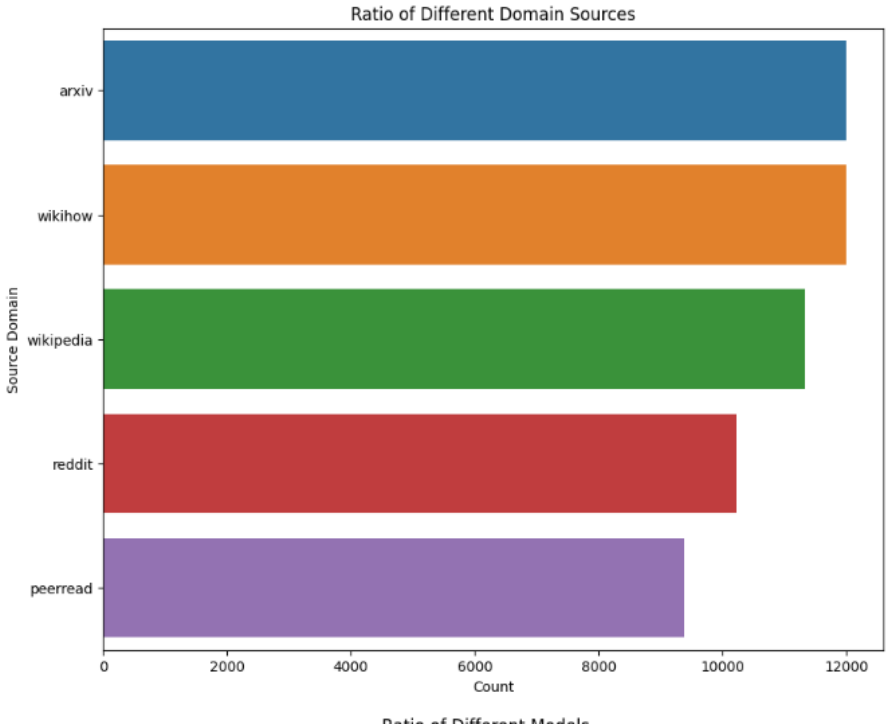
Since loading the complete M4 dataset is too time-consuming and too troublesome to do the fixing if there are any bugs inside any code, I have just loaded around 45k data in M4 dataset, splitting some of them as training, validation and testing data to do the classification. Here is the distribution of the data:

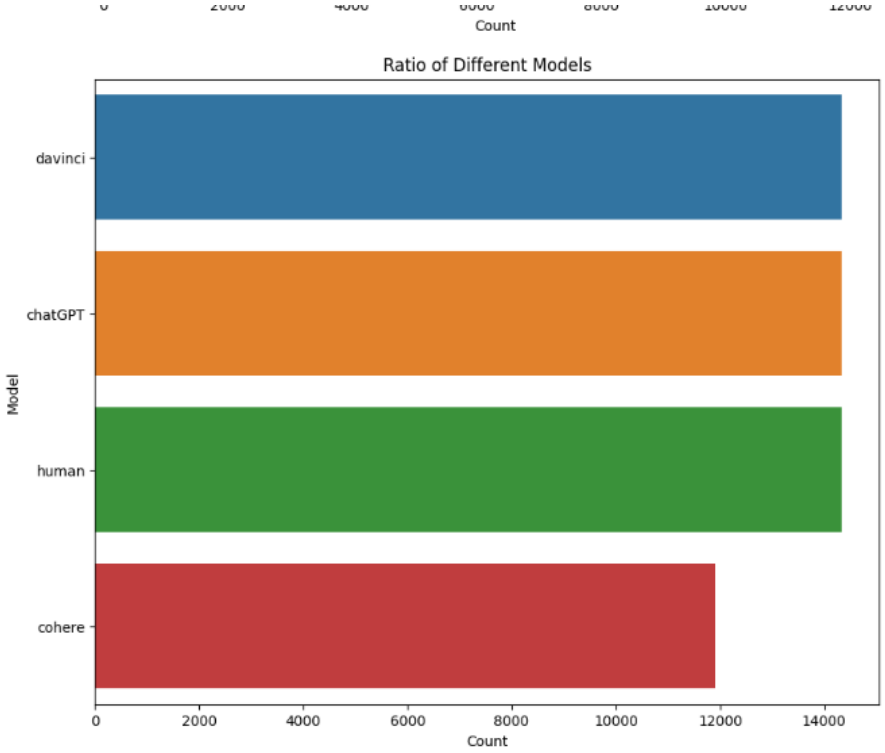
```
4  pleas gener wikihow articl length 1000 charact...    wikihow    human
14339
```




```
                                                text  label  \
0  we present measur d_ sj 2317 meson product cro...      1
1  how use dexnav featur pokémon omega rubi alpha...      1
2  lashi languag indigen languag spoken lashi peo...      1
3  in paper role conserv law evolutionari process...      0
4  it invas make ideal fill bed grow lawn howev p...      0

                                              prompt     source    model
0  gener abstract work titl d_ sj 2317 meson prod...      arxiv   cohere
1  gener wikihow articl minimum 200 word titl how...    wikihow   cohere
2  write wikipedia articl titl lashi languag arti...  wikipedia  chatGPT
3  gener 150 220 word abstract work titl conserv ...      arxiv    human
4  pleas gener wikihow articl length 1000 charact...    wikihow    human
14339
```





Model training setting

1. SVM

Since I can just directly call the SVM models by using libraries, there are no additional settings other than the TF-IDF mentioned in above section.

2. CNN-BiLSTM-DoubleAttention

For the model and its variants, I have used the same initialization parameters to ensure they have similar complexity, here is the list of them:

They are trained with 10 epochs with AdamW with learning rate of 0.001, weight decay of 0.005 and the scheduler of ReduceLROnPlateau to train the models.

3. Mistral-7b-bnb-4bit

The most of the training settings are mentioned in the above sections, and this model is trained with Adamw 8bit versions with learning rate of 2e-4 and weight decay of 0.01. There are also linear learning rate schedular.

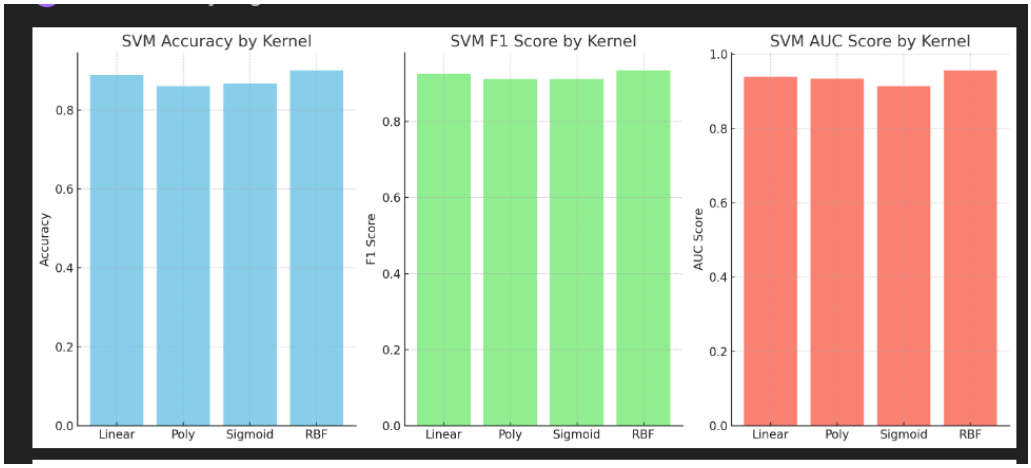
Results

SVM

Metrics

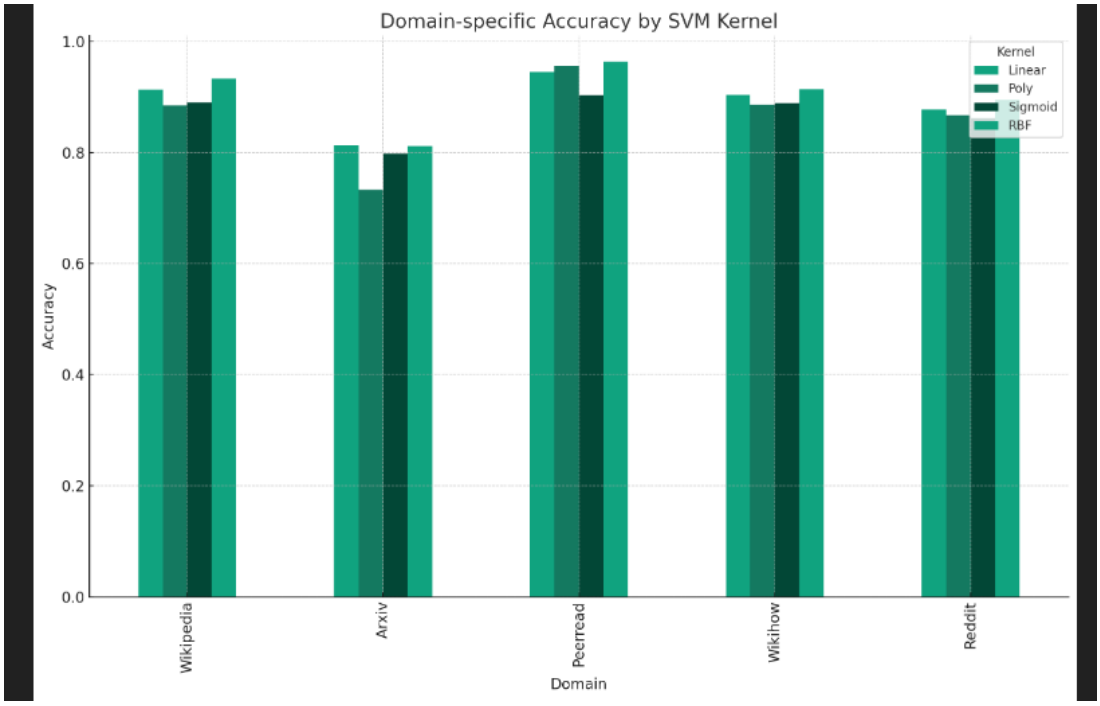
	Linear	Poly	Sigmoid	RBF
Accuracy	0.887797781	0.860520095	0.866612111	**0.900436443**
F1 Score	0.925894787	0.911889719	0.911674393	**0.934654174**

AUC Score	0.938373151	0.933698033	0.914235356	**0.956517463**



Domain

	Linear	Poly	Sigmoid	RBF
Wikipedia	0. 913004882	0. 884598313	0. 890368398	**0. 933422104**
Arxiv	**0. 812836439**	0. 732505176	0. 797929607	0. 811594203
Peerread	0. 945089757	0. 956177402	0. 902851109	**0. 963041183**
Wikihow	0. 903604359	0. 886001676	0. 888935457	**0. 914082146**
Reddit	0. 877073171	0. 866829268	0. 861951220	**0. 895121951**



From those tables and results figures, it is clearly shows that SVM models with RBF kernel outperforms the others in terms of accuracy, F1 score and AUC score, showing it power in handling the dataset.

In terms of domain, RBF kernel shows consistent and nearly all the best performance over each domain. While some kernel like Poly kernel even performed well on Peeread domain, varied a lot on other domain. Showing those kernels are not comparable to RBF kernel in terms of consistency.

CNN-BiLSTM with double attention

Here are the results of this model and its variations

For simplicity

BiLSTM -> Model A

CNNBiLSTM -> Model B

AttentionBiLSTM -> Model C

CNNBiLSTM with Attention -> Model D

CNNBiLSTM with Double Attention -> Model E

General Metric

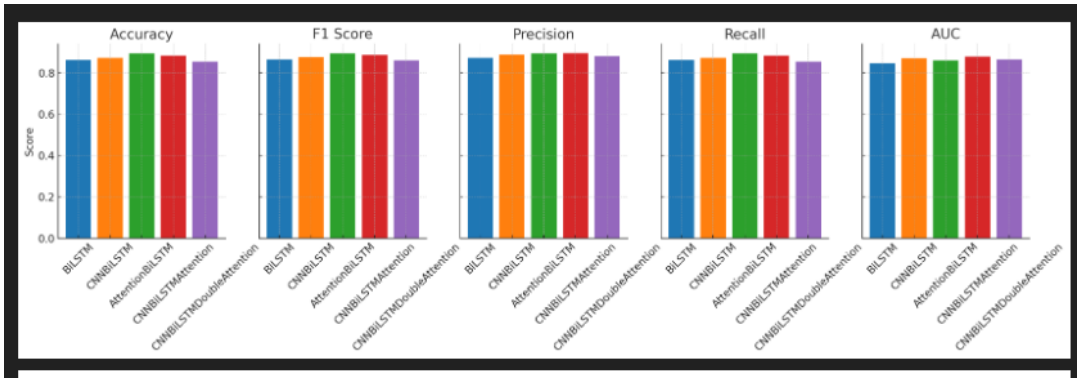
	Model A	Model B	Model C	Model D	Model E
accuracy	0.86147	0. 871704	**0. 89339**	0.88325	0.852923
F1 score	0.864837	0.875738	**0.893401**	0.88632	0.85916
precision	0.8718700	0.886632	**0.89340**	0.89396446	0.879765
recall	0.861476	0.87170	**0.89339**	0.8832554	0.85292
auc	0.84647	0.87024	0.8602775	**0.8777912**	0.86460

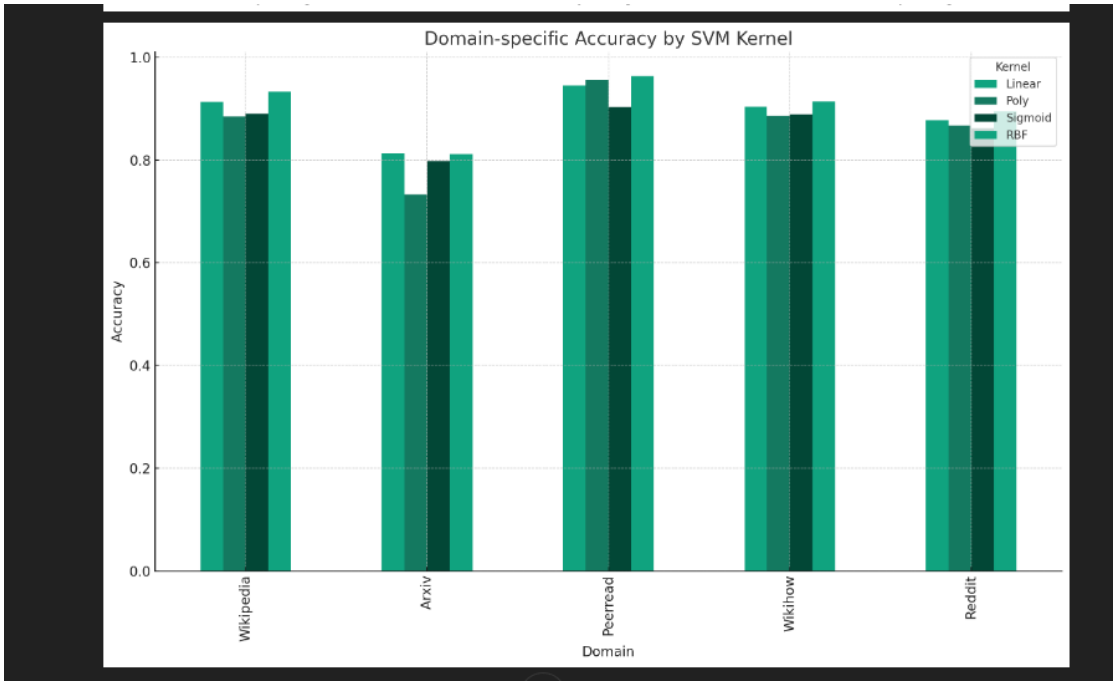
Accuracy in terms of Domain

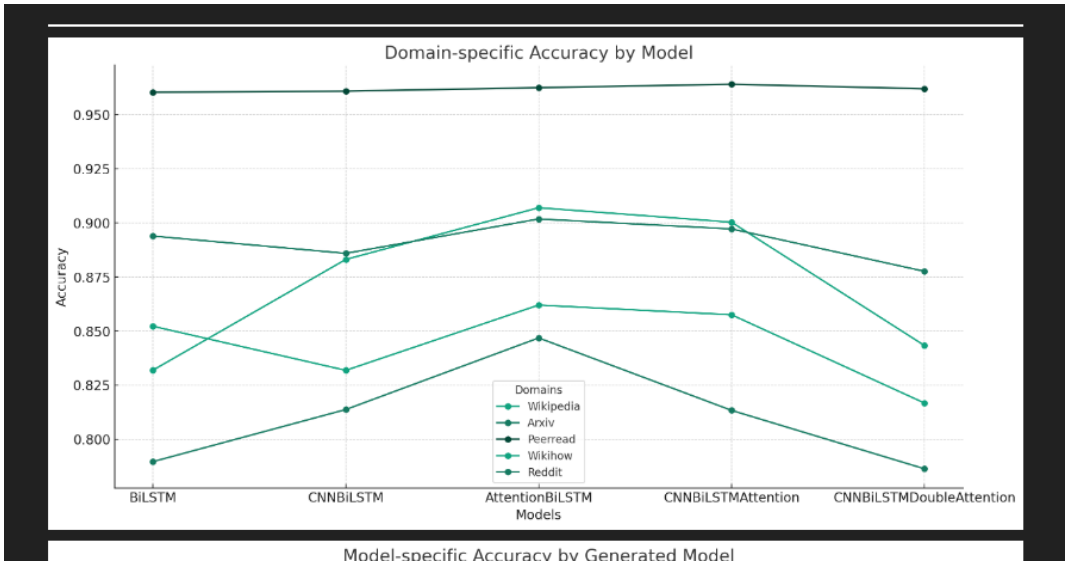
	Model A	Model B	Model C	Model D	Model E
Wikipedia	0.852197	0.831779	**0.86196182**	0.8575233	0.8166888
Arxiv	0.7896480	0.813664	**0.84679089**	0.8132505	0.786335
Peerread	0.96040126	0.96092925	0.96251319	**0.96409714**	0.9619852
wikihow	0.83193629	0.88306789	**0.9069572**	0.90025146	0.843252
reddit	0.8938570	0.88591725	**0.9017969**	0.89720016	0.8775595

Accuracy in terms of source of generating the text

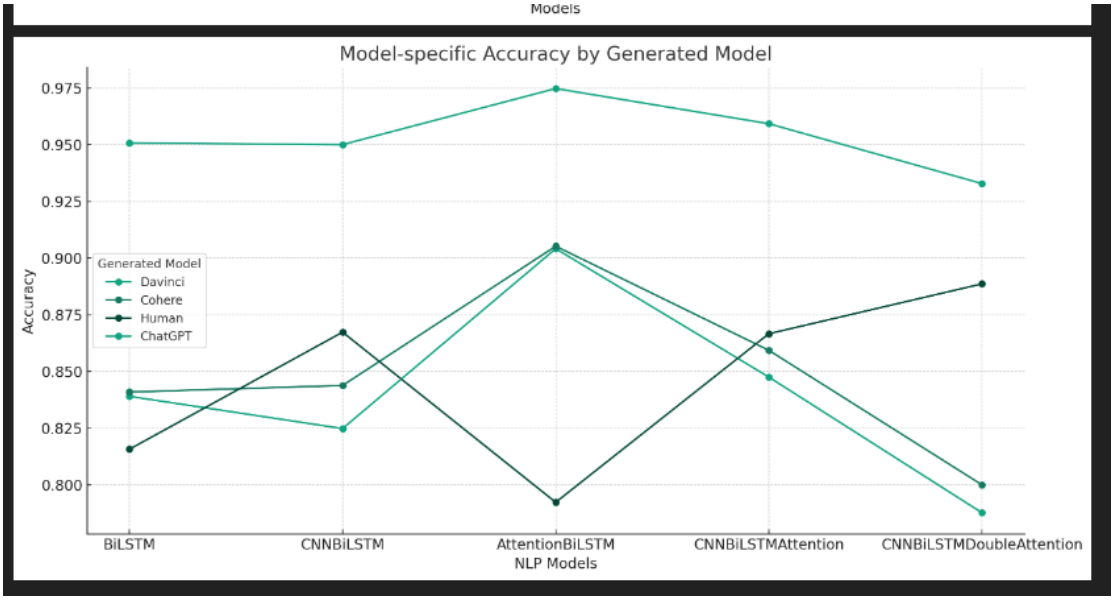
	Model A	Model B	Model C	Model D	Model E
davinci	0.83899504	0.82484076	**0.90410474**	0.84748761	0.7876857
cohere	0.840867	0.843761301	**0.9052441**	0.859312	0.8
human	0.81568088	0.86726272	0.7922971	0.8665749	**0.8885832**
chatGPT	0.95073891	0.95003518	**0.97466572**	0.9591836	0.932793







Model-specific Accuracy by Generated Model



In terms of the overall model performance, the AttentionBiLSTM (model C) showcases superior performance in terms of basically all the metrics against other models.

In terms of domain-specific accuracy, still the AttentionBiLSTM model consistently shows high accuracy across most domains, especially in Peerread and Wikihow domains. While others vary levels of effectiveness across different sources.

In terms of source generating the text, even though AttentionBiLSTM performed well when detecting different LLM model text, it performed poorly when detecting human text. However, the CNNBiLSTM with double attention perform consistent in terms of that. In detecting the text generated by ChatGPT, all the models performed well in it, showcasing that those models can greatly capture the features of whether it is generated by ChatGPT.

Reason for such results:

1. The data are greatly imbalanced, the number of human text and machine-text are in 1:3, meaning that those models can perform great results just by capturing the AI generated text well. (Even using the focal loss to deal with this problem)
2. Overfitting, in the experience, I have trained all the models in 10 epochs, in fact, some models like CNNBiLSTM with double attention, already overfit when running epoch 7 to 8, losing its performance all running that much epochs. It will be somehow relieved if using the complete size of dataset.

Mistral-7b-bnb-4bit

Even though I have completed the code to fine-tune the model, successfully without any bug, I don't have time trying to train the model with that dataset because I will otherwise late submit the results (Maybe appending the results after the submission as the appendix).

Roberta

I have not implemented the code to do the zero-shot of this model yet. Though it is easy to do so because I will just be doing the inferencing to test the zero-shot power of it.

Obstacles and next steps

For the obstacles, there has a huge problem when training the models with an imbalanced dataset as mentioned before. Moreover, even limiting the vector sized, the time needed to train all the SVM with different kernel are too time-consuming, it took 4.5 hours to train 4 kernels with just the part of the dataset in M4, worrying that it will take longer if larger data is used.

For the next step, I will think about how to tidy the data first to become more balanced for the training process. Meanwhile, I will try inventing other dataset for doing the out-of-distribution test in terms of the source or the model generating the text. And testing the model's performance in AI-Human mixture text. For the model's itself, I will try reducing the size of those model to preventing the overfitting issue as well.

Reference list

[1] J. Lee, T. Le, J. Chen, and D. Lee, "Do language models plagiarize?" in Proceedings of the ACM Web Conference 2023, 2023, pp. 3637–3647.

[2] A. Pagnoni, M. Graciarena, and Y. Tsvetkov, “Threat scenarios and best practices to detect neural fake news,” in Proceedings of the 29th International Conference on Computational Linguistics. International Committee on Computational Linguistics, 2022, pp. 1233–1249. [Online]. Available: https://aclanthology.org/2022.coling-1.106
[3] Y. Mirsky, A. Demontis, J. Kotak, R. Shankar, D. Gelei, L. Yang, X. Zhang, M. Pintor, W. Lee, Y. Elovici et al., “The threat of offensive ai to organizations,” Computers & Security, p. 103006, 2022.
[4] C. Stokel-Walker, “Ai bot chatgpt writes smart essays should academics worry?” Nature, 2022.
[5] E. Kasneci, K. Seßler, S. Kuchemann, M. Bannert, ¨ D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Gunnemann, E. H ¨ ullermeier ¨ et al., “Chatgpt for good? on opportunities and challenges of large language models for education,” Learning and Individual Differences, vol. 103, p. 102274, 2023.
[6] OpenAI, “GPT-4 technical report,” CoRR, vol. abs/2303.08774, 2023. [Online]. Available: https: //doi.org/10.48550/arXiv.2303.08774
[7] Anthropic, “Model card and evaluations for claude models,” 2023. [Online]. Available: https://www-files.anthropic.com/production/ images/Model-Card-Claude-2.pdf
[8] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann et al., “Palm: Scaling language modeling with pathways,” ArXiv preprint, vol. abs/2204.02311, 2022. [Online]. Available: https: //arxiv.org/abs/2204.02311
[9] A. Yuan, A. Coenen, E. Reif, and D. Ippolito, “Wordcraft: story writing with large language models,” in 27th International Conference on Intelligent User Interfaces, 2022, pp. 841–852.
[10] B. A. Becker, P. Denny, J. Finnie-Ansley, A. LuxtonReilly, J. Prather, and E. A. Santos, “Programming is hard-or at least it used to be: Educational opportunities and challenges of ai code generation,” in Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1, 2023, pp. 500–506.
[11] Q. Zheng, X. Xia, X. Zou, Y. Dong, S. Wang, Y. Xue, Z. Wang, L. Shen, A. Wang, Y. Li et al., “Codegeex: A pre-trained model for code generation with multilingual evaluations on humanevalx,” ArXiv preprint, vol. abs/2303.17568, 2023. [Online]. Available: https://arxiv.org/abs/2303.17568
[12] Li, P., Xu, W., Ma, C., Sun, J.,Yan, Y. (2015). IOA: Improving SVM based sentiment classification through post processing. In Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015) (pp. 545- 550).
[13] Gurkhe, D., Pal, N., Bhatia, R. (2014). Effective sentiment analysis of social media datasets using Naive Bayesian classification. International Journal of Computer

Applications, 975(8887), 99.
[14] Pang, B., Lee, L., Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. arXiv preprint cs/0205070.
[15] B. Guo, X. Zhang, Z. Wang, M. Jiang, J. Nie, Y. Ding, J. Yue, and Y. Wu, “How close is chatgpt to human experts? comparison corpus, evaluation, and detection,” ArXiv preprint, vol. abs/2301.07597, 2023. [Online]. Available: https://arxiv.org/abs/2301.07597
[16] Y. Ma, J. Liu, and F. Yi, “Is this abstract generated by ai? a research for the gap between ai-generated scientific text and human-written scientific text,” ArXiv preprint, vol. abs/2301.10416, 2023. [Online]. Available: https://arxiv.org/abs/2301.10416
[17] A. Munoz-Ortiz, C. G ˜ omez-Rodr ´ ´ıguez, and D. Vilares, “Contrasting linguistic patterns in human and llmgenerated text,” ArXiv preprint, vol. abs/2308.09067, 2023. [Online]. Available: https://arxiv.org/abs/2308. 09067 [43] S. Giorgi, D. M. Markowitz, N. Soni, V. Varadarajan, S. Mangalik, and H. A. Schwartz, “”i slept like a baby”: Using human traits to characterize deceptive chatgpt and human text,” in Proceedings of the IACT - The 1st International Workshop on Implicit Author Characterization from Texts for Search and Retrieval held in conjunction with the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2023), Taipei, Taiwan, July 27, 2023, ser. CEUR Workshop Proceedings, M. Litvak, I. Rabaev, R. Campos, A. M. Jorge, and A. Jatowt, Eds., vol. 3477. CEUR-WS.org, 2023, pp. 23–37. [Online]. Available: https://ceur-ws.org/ Vol-3477/paper4.pdf
[18] Y. Wu and G. Deng, "Interactive attention network fusion Bi-LSTM and CNN for text classification," in Proc. SPIE 12254, International Conference on Electronic Information Technology (EIT 2022), 122542F, May 23, 2022. [Online]. Available: https://doi.org/10.1117/12.2638585
[19] Wang, J., Yu, L. C., Lai, K. R., Zhang, X. (2016, August). Dimensional sentiment analysis using a regional CNN-LSTM model. In Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers) (pp. 225-230).
[20] Deng, J., Cheng, L., Wang, Z. (2021). Attention-based BiLSTM fused CNN with gating mechanism model for Chinese long text classification. Computer Speech & Language, 68, 101182.
[21] Liu, G.,Guo, J. (2019). Bidirectional LSTM with attention mechanism and convolutional layer for text classification. Neurocomputing, 337, 325-338.
[22] Cheng, Y., Yao, L., Zhang, G., Tang, T., Xiang, G., Chen, H., ... Cai, Z. (2020). Text sentiment orientation analysis of multi-channels CNN and BiGRU based on attention

mechanism. Journal of Computer Research and Development, 57(12), 2583.
[23] Bahdanau, D., Cho, K., Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
[24] Du, J., Gui, L., Xu, R.,He, Y. (2017). A convolutional attention model for text classification. In National CCF conference on natural language processing and Chinese computing (pp. 183-195). Springer, Cham.
[25] SiChen, L. (2019, October). A neural network based text classification with attention mechanism. In 2019 IEEE 7t
[26] Wang, G., Li, C., Wang, W., Zhang, Y., Shen, D., Zhang, X., ... Carin, L. (2018). Joint embedding of words and labels for text classification, in: Proceedings of the 56th Annual Meeting of the Association for Computa405 tional Linguistics (Volume 1: Long Papers), 2018, pp. 2321–2331.
[27] Dong, Y., Liu, P., Zhu, Z., Wang, Q., & Zhang, Q. (2019). A Fusion Model-Based Label Embedding and SelfInteraction Attention for Text Classification. IEEE Access, 8: 30548-30559. https://doi.org/10.1109/ACCESS. 2019.2954985
[28] Xiao, Y., Li, Y., Yuan, J., Guo, S., Xiao, Y., & Li, Z. (2021). History-based attention in Seq2Seq model for multi-label text classification. Knowledge-Based Systems, 224, 107094.
[29] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, “Pre-trained models for natural language processing: A survey,” Science China Technological Sciences, vol. 63, no. 10, pp. 1872–1897, 2020.
[30] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), J. Burstein, C. Doran, and T. Solorio, Eds. Association for Computational Linguistics, 2019, pp. 4171–4186. [Online]. Available: https://doi.org/10.18653/v1/n19-1423
[31] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized BERT pretraining approach,” CoRR, vol. abs/1907.11692, 2019. [Online]. Available: http://arxiv.org/abs/1907.11692
[32] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, “Xlnet: Generalized autoregressive pretraining for language understanding,” in Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alche-Buc, E. B. Fox, and R. Garnett, ´ Eds., 2019, pp. 5754–5764. [Online]. Available:

https://proceedings.neurips.cc/paper/2019/hash/dc6a7e655d7e5840e66733e9ee67cc69-Abstract.html
[33] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “GLUE: A multi-task benchmark and analysis platform for natural language understanding,” in 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=rJ4km2R5t7
[34] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. Le Scao, T. Lavril, T. Wang, T. Lacroix, and W. El Sayed, "Mistral 7B," 2023. [Online]. Available: https://arxiv.org/abs/2310.06825.
[35] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, and S. Feizi, “Can ai-generated text be reliably detected?” ArXiv preprint, vol. abs/2303.11156, 2023. [Online]. Available: https://arxiv.org/abs/2303. 11156
[36] M. S. Orenstrakh, O. Karnalim, C. A. Suarez, and M. Liut, “Detecting llm-generated text in computing education: A comparative study for chatgpt cases,” ArXiv preprint, vol. abs/2307.07411, 2023. [Online]. Available: https://arxiv.org/abs/2307.07411
[37] Z. Shi, Y. Wang, F. Yin, X. Chen, K.-W. Chang, and C.-J. Hsieh, “Red teaming language model detectors with language models,” ArXiv preprint, vol. abs/2305.19713, 2023. [Online]. Available: https: //arxiv.org/abs/2305.19713
[38] X. He, X. Shen, Z. Chen, M. Backes, and Y. Zhang, “Mgtbench: Benchmarking machine-generated text detection,” ArXiv preprint, vol. abs/2303.14822, 2023. [Online]. Available: https://arxiv.org/abs/2303.14822
[39] D. Shen, M. Zheng, Y. Shen, Y. Qu, and W. Chen, “A simple but tough-to-beat data augmentation approach for natural language understanding and generation,” ArXiv preprint, vol. abs/2009.13818, 2020. [Online]. Available: https://arxiv.org/abs/2009.13818
[40] H. Lee, D. A. Hudson, K. Lee, and C. D. Manning, “SLM: Learning a discourse language representation with sentence unshuffling,” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, 2020, pp. 1551–1562. [Online]. Available: https://aclanthology.org/2020.emnlp-main.120
[41] G. Liang, J. Guerrero, and I. Alsmadi, “Mutationbased adversarial attacks on neural text detectors,” ArXiv preprint, vol. abs/2302.05794, 2023. [Online]. Available: https://arxiv.org/abs/2302.05794
[42] Z. Shi and M. Huang, “Robustness to modification with shared words in paraphrase identification,” in Findings of the Association for Computational

Linguistics: EMNLP 2020. Association for Computational Linguistics, 2020, pp. 164–171. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.16
[43] D. Chen, “MixSet: Official code repository for Mixset,” GitHub, 2024. [Online]. Available: https://github.com/Dongping-Chen/MixSet1
[44] Y. Wang, J. Mansurov, P. Ivanov, J. Su, A. Shelmanov, A. Tsvigun, C. Whitehouse, O. M. Afzal, T. Mahmoud, A. F. Aji, and P. Nakov, “M4: Multi-generator, Multi-domain, and Multi-lingual Black-Box Machine-Generated Text Detection,” in arXiv:2305.14902, May 2023. [Online]. Available: https://arxiv.org/abs/2305.149024
[45] X. He, X. Shen, Z. Chen, M. Backes, and Y. Zhang, “MGTBench: Benchmarking Machine-Generated Text Detection,” in arXiv:2303.14822, Mar. 2023. [Online]. Available: https://arxiv.org/abs/2303.148222
[46] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," Aug. 2017, revised Feb. 2018. [Online]. Available: https://arxiv.org/abs/1708.02002.