Text Detection between an AI Written Passage vs. a Human Written Passage

Saif Khan
Computer Science Department
Minot State University
Minot, ND, 58707
Saif.khan@minotstateu.edu

Kaif Khan
Computer Science Department
Minot State University
Minot, ND, 58707
Kaif.khan@minotstateu.edu

Prof. Muhammad Abusaqer
Computer Science Department
Minot State University
Minot, ND, 58707
Muhammad.abusaqer@minotstateu.edu

Abstract

This paper dives into AIGT (AI-Generated Texts) detection with a specific focus on differentiating passages written by Artificial Intelligent (AI) models from those crafted by human writers. With the increase of advanced AI language models such as ChatGPT the need for robust methods to discern AI-generated text from human-authored content has become increasingly essential.

Our research uses a careful methodology to answer the question, what is the best way to train and detect AIGT from human-generated texts? We explored three different ways to evaluate and figure out which one would give the most accurate results. (1) Classification methodology using Deep Learning (DL) with the help of BERT, and (2) Comparing Machine Learning (ML) tools like Naive Bayes and Deep Learning tools like BERT to test for better accuracy. In addition, we tested the possibility of using (3) Sentiment analysis as a tool to distinguish AIGT and human writers. Leveraging a dataset comprising passages generated by AI models and texts authored by human writers, our dataset includes 487,235 such passages. We reported Accuracy, Precision, Recall, and F1 score, for our study experimentation.

The findings of this study have significant implications for plagiarism detection in educational and professional environments. By providing insights into the effectiveness of text detection methods, our research contributes to advancing text analysis techniques and informs the development of more reliable text authentication processes.

Keywords: Text detection, AI, AI-generated content, Human-authored content, Comparative analysis, Plagiarism Detection.

1 Introduction

Artificial Intelligence (AI) is a rapidly evolving field, and the advancements in Natural Language Processing (NLP) and Large Language Models (LLMs) have brought about text generation capabilities that very closely resemble human writing. Such technological advancement raises some serious red flags, particularly regarding the spread of plagiarism across various facets of modern-day life. Plagiarism, both in educational and professional environments, poses a serious threat to integrity and originality. The rise of ChatGPT from OpenAI [1][2], Gemini from Google [3], and Co-Pilot from Microsoft [4] have made generating texts in large quantity incredibly accessible. In educational settings, the spread of AI-generated texts has made it challenging to distinguish between authentic student work and plagiarized content. This undermines the academic integrity of institutions but also robs genuine learners of the opportunity to showcase their own knowledge and creativity. Moreover, it erodes the foundation of scholarly discourse and intellectual advancement, as original contributions are overshadowed by the spread of copied content.

In our research we dive into the challenges of distinguishing between passages authored by humans and those generated by AI. Our objective is to determine the most optimal approach for constructing and training a model capable of reliably predicting texts generated by artificial intelligence (AIGT). Our aim is to develop a robust methodology that can ensure accurate identification of AIGT, thereby addressing the growing need for effective text authentication in contemporary contexts. Through the help of tools such as Naive Bayes [5], which is a powerful machine learning (ML) algorithm that is a probabilistic classifier, it evaluates the likelihood of certain words or features co-occurring to classify text effectively. Bidirectional Encoder Representations from Transformers (BERT) [6], which is a sophisticated deep learning (DL) model developed by Google that captures contextual information in text, allowing for more precise analysis. We conducted a comparative analysis of both techniques to assess their relative effectiveness. Furthermore, we investigated using Sentiment Analysis [7] as another way to figure out if a text was written by AI. Sentiment analysis involves examining the emotional tone behind the text, which can sometimes be different between human and AI-generated content. By studying sentiment patterns, we can improve the effectiveness of our detection methods. Overall, our goal is to contribute to the development of reliable methods for distinguishing between human and AI-generated passages. By testing these cutting-edge techniques, we aim to address the growing need for accurate text authentication models.

2 Literature Review

As the AI trend increases exponentially, detecting the difference between AI text and human text will become more demanding. It’s necessary that we start training new models that can tell the difference between AI text and human text accurately. This research by Mindner, Schlippe, and Schaaf [8] explores methods to identify text created entirely by AI and text that has been rephrased by AI from an original source. The authors employed a combination of features to train their detection systems, including perplexity, semantic analysis, readability scores, and feedback from other AI models. Their systems achieved high accuracy over 96% F1-score in classifying both basic and more sophisticated human-written and AI-generated texts. Additionally, the systems achieved an accuracy of over 78% F1-score for distinguishing between human-generated and AI-rephrased text. However, the authors acknowledged that as AI language models improve, detection will become more challenging, cautioning that the task of differentiating human and AI authorship is likely to become increasingly difficult as AI language capabilities advance.

Another research done by Xiaomeng Hu, Pin-Yu Chen, Tsung-Yi Ho proposes RADAR [9], a framework that leverages adversarial learning to train an AI text detector, involving a unique approach: training a detector and a paraphraser in opposition to each other. The paraphraser learns to generate realistic text that evades detection as AI-generated, while the detector hones its ability to identify such text. This competitive training process enhances the robustness of the detector against paraphrased AI text. RADAR employs three neural networks: a target large language model, a detector, and a paraphraser. The detector is then trained to distinguish only AI-generated text, while the paraphraser attempts to create realistic text that bypasses detection. Through a feedback loop, the detector's evaluations inform the paraphraser's updates, and vice versa, strengthening both models iteratively. Hu, Chen, and Ho's (2023) experiments demonstrated that RADAR outperformed existing AI text detection models, particularly when dealing with paraphrased text. Extensive testing across various datasets and large language models proved RADAR's effectiveness in addressing the challenges associated with AI-generated content detection. However, it is important to acknowledge that the research is limited to some specific LLMs used for training and evaluation, the effectiveness of RADAR in the future may change.

Pengyu Wang introduced SeqXGPT [10], a novel method for fine-grained, sentence-level detection of AI-generated text, focusing on utilizing log probability lists from white-box large language models to identify patterns indicative of AI-generated text. SeqXGPT departs from previous methods by employing log probability lists, capturing the internal uncertainty of an LLM when generating text. The approach involves three key steps: Perplexity Extraction and Alignment, where perplexity lists are generated from a pre-trained LLM for each sentence Feature Encoding, utilizing a combination of convolutional and self-attention neural network layers to learn complex patterns distinguishing human-written and AI-generated sentences and Linear Classification, where a classification layer analyzes encoded features to label sentences as human-written or AI-generated. Wang demonstrated that his method achieved high precision, recall, and F1-score in detecting AI-generated sentences. While excelling at identifying statistical inconsistencies within AI-generated text, SeqXGPT could benefit from integrating semantic information to enhance discernment of human-like AI-generated sentences. Despite this, the model presents a promising avenue for AI text detection, showcasing potential for addressing challenges associated with identifying AI-generated content at the sentence level while maintaining generalizability across domains.

Professor Chaka from University of South Africa examines the effectiveness of various AI content detection tools [11], including GPTZero and CopyLeaks, in identifying essays written by humans and AI, specifically focusing on the use of large language models like ChatGPT. The research emphasizes the importance of these tools in safeguarding academic integrity by combating plagiarism, particularly with the increasing sophistication of AI-generated essays. It employs a multi-faceted approach, evaluating the six AI text detectors on their ability to classify essays across four distinct writing styles: argumentative, descriptive, expository, and narrative. This approach provides valuable insights into the strengths and weaknesses of these detection tools across various writing formats commonly found in academic settings. The paper finds that all six detectors performed decently but not flawlessly in distinguishing human-written from AI-generated essays. While CopyLeaks exhibited the most promising results, none of the detectors achieved perfect accuracy. The author highlights limitations in current detection tools, due to their relatively new development stage. This finding demonstrates the need for continued research and development to enhance the accuracy of AI content detection tools as LLMs continue to evolve.

Research done by Junchao Wu [12] shows the capabilities of large language models that have sparked a surge on methods to differentiate between human-written and AI-generated text, which is crucial for overcoming plagiarism and ensuring online content authenticity. Several key approaches that were employed by Wu are: Statistical Analysis involves analyzing properties like perplexity, which can indicate AI-generated content due to less variation in word choices compared to humans. Feature-Based Detection examines readability scores and word categories, with human-written text typically demonstrating broader vocabulary and sentence complexity. Adversarial Training pits a detector against a paraphraser, improving robustness against paraphrased text by enhancing the detector's ability to discern cleverly rephrased AI content. Inconsistency Detection targets illogical elements or factual inaccuracies, which may indicate AI authorship due to the difficulty in replicating human nuance and accuracy. These methods are promising but still under development, with ongoing refinement necessary to keep pace with advancements in AI text generation. Wu acknowledged that there are limited high-quality datasets to train these models on to get accurate results.

In our research we have conducted an in-depth analysis to compare BERT and Naive Bayes model, including testing to see if Sentiment Analysis could help in detecting AI-generated text versus human-generated text. Our research tries to show which methodology of training a model can result in more accurate detection.

3 Methodology and Results

In this section we start by explaining how we collected and explored our data using pandas. Then, in section 3.1 we dive into how we trained and tested the BERT model. In section 3.2 we discuss how we trained and built our Naive Bayes model. In 3.3 we dive into the comparative analysis of BERT and Naive Bayes. In 3.4 we look at how Sentiment Analysis can be used to detect AIGT.

Dataset Exploration

The dataset we used was sourced from Kaggle and provided by Shayan Gerami [13], which comprises of two primary columns: 'text' and 'generated'. The 'text' column contains randomly selected passages, each paired with a corresponding label in the 'generated' column. These labels indicate whether the passage was authored by a human ('0') or generated by an artificial intelligence ('1'). The dataset consists of 487,235 rows, with no

NULL values in either column. Among these rows, 305,797 passages are labeled as human-authored, while 181,438 are attributed to AI generation. This distribution results in a ratio of 37.2% AI-generated passages to 62.7% human-authored passages within the dataset. Furthermore, we divided our dataset into 80% training data and 20% testing data for our experiment. We decided to use accuracy, precision, recall, and F-1 score as evaluation metrics. The reason we chose accuracy [14] is because it refers to the proportion of correctly classified instances out of the total instances evaluated. Precision [15] refers to the proportion of true positive predictions out of all positive predictions made by the model. Recall [16] measures the proportion of true positive instances that are correctly identified by the model out of all actual positive instances in the dataset. And finally, f-1 score [17], which is a metric that combines both precision and recall into a single value, providing a balance between these two metrics.

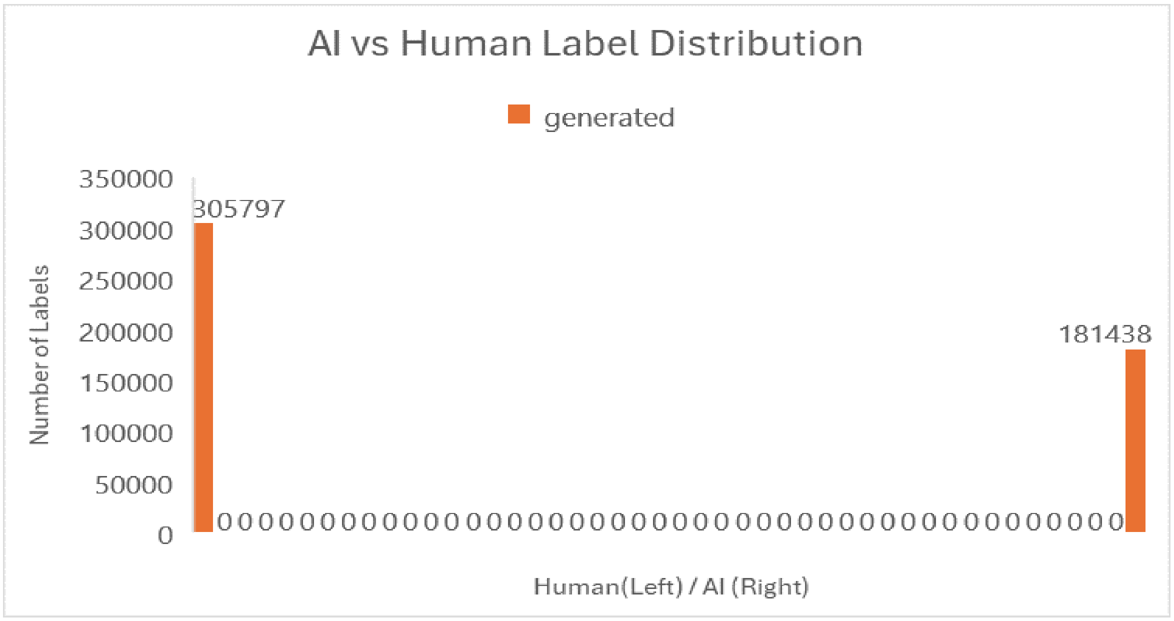

Figure 1: 305,797 datasets are labeled as ‘0’ which means human generated and 181,438 are labeled ‘1’ which means AI generated.

3.1 BERT Classification

In our research, we trained the powerful Bidirectional Encoder Representations from Transformers (BERT) model. The implementation of the BERT model involved utilizing several Python libraries [18], including: 1) Scikit-learn (sklearn) [19] which offers a wide range of tools for data preprocessing, model selection, and evaluation. We chose sklearn for its efficient implementation of classification algorithms and metrics computation. 2) Pandas [20] commonly used for data manipulation. 3) PyTorch (torch) [21] which is a deep learning framework, we used torch to build and train our BERT model. 4) From the Transformers library, we imported essential components for BERT model implementation, including BertForSequenceClassification and BertTokenizer [22]. These modules provided pre-trained BERT models and tokenization functionalities necessary for sequence classification tasks. 5) AdamW optimizer and get_linear_schedule_with_warmup [23] [24] are key components for fine-tuning BERT models. AdamW offers efficient optimization with weight decay, while the scheduler adjusts the learning rate during training to improve convergence. Additionally, we employed performance evaluation metrics such as precision, recall, F1 score, and accuracy to assess the BERT model's effectiveness in classifying text sequences.

3.1.1 Training and Evaluation

We evaluated the performance of the BERT model on a comprehensive training set consisting of 80% of the dataset, which comes out to be 389,788 datasets. Each comprises passages authored by either humans or AI language models. The training phase and testing period spanned 3 days, 17 hours, and 19 minutes, significantly longer than the time required for our Naive Bayes model, as detailed in section 3.2. The results of our BERT model testing show exceptional performance metrics, with an accuracy of 0.999, precision of 0.999, recall of 0.998, and an F1 score of 0.998. These scores are remarkably close to perfection, which highlights the accuracy of the BERT classification model compared to Naive Bayes.

Metrics	**Results**
Accuracy	99.9%
Precision	99.9%
Recall	99.8%
F-1 Score	99.8%

Table 1: Table shows the evolution metrics of BERT Model.

3.2 Naive Bayes Classification

In our research, we utilized the Naive Bayes algorithm for text classification. Naive Bayes is commonly used in text classification tasks due to its computational efficiency, particularly with high-dimensional feature spaces found in NLP applications. Despite its simplistic assumptions, Naive Bayes often yields competitive performance and can provide quick and interpretable results. We implemented the Naive Bayes algorithm using Python libraries such as scikit-learn (sklearn) and pandas. We also used the TFIDFVectorizer [25], short for Term Frequency-Inverse Document Frequency Vectorizer, which is a feature extraction technique that converts text documents into numerical vectors based on the frequency of terms and their importance across the corpus. It calculates a weight for each term in the document, considering both its frequency in the document and its rarity across the entire corpus. Also using Multinomial Naive Bayes (MultinomialNB) [26] which is a variant of the Naive Bayes algorithm suitable for classification tasks with discrete features, such as word counts in text classification. It models the likelihood of observing each feature given the class and uses the probabilities to predict the most likely class label for a given sample.

3.2.1 Training and Evaluation

We trained the Naive Bayes model on 487,235 training and testing dataset. The training process took approximately 63 seconds to complete, demonstrating the algorithm's efficiency in processing large datasets compared to more complex models like BERT (section 3.1). Upon training completion, we evaluated the model's performance using standard evaluation metrics such as accuracy, F1 score, and precision. The Naive Bayes model achieved an overall accuracy of 0.95, with an F1 score of 0.96 for human-authored passages and 0.93 for AI-generated passages, precision for human-authored passages was 0.93, while for AI-generated passages, it was 0.98, recall for human passages was 0.99 and AI-generated passages was 0.88.

Metrics	**Results**
Accuracy	95%
Precision (Human/AI)	Human - 93% AI - 98%
Recall (Human/AI)	Human - 99% AI – 88%
F-1score (Human/AI)	Human – 96% AI – 93%

Table 2: Table shows evaluation metrics of Naive Bayes Model.

3.3 Sentiment Analysis

In addition to machine learning techniques for text classification, we explored the possibility of sentiment analysis as a supplementary tool for distinguishing between AI-generated and human-authored texts. Sentiment analysis, also known as opinion mining, aims to determine the sentiment or emotional tone behind a piece of text, whether it be positive, negative, or neutral. By analyzing the underlying sentiments expressed in textual content, we tried to uncover potential indicators that could differentiate between AI-generated and human-authored texts. To conduct sentiment analysis, we used the VADER [27] (Valence Aware Dictionary and sEntiment Reasoner) sentiment analysis tool, a lexicon and rule-based sentiment analysis tool. VADER is well-suited for analyzing short passages and informal text, making it a perfect choice for our experiment. This tool provides sentiment scores for text passages, indicating the intensity of positive, negative, and neutral sentiments expressed within the text. Sentiment analysis has its limitations, particularly in contexts where textual content may show ambiguity or sarcasm. Future researchers could explore more sophisticated sentiment analysis techniques, including deep learning-based approaches, to enhance the accuracy and robustness of sentiment analysis in detecting AI-generated texts. Additionally, integrating sentiment analysis with other text analysis methods, such as topic modeling and linguistic analysis, could further refine AI text detection models and improve their performance across many genres and domains. We chose to make a histogram to represent our findings. In these graphs we can see human labeled passages resulted in a higher negative sentiment and a lower positive sentiment vs the AI labeled passages resulted in a lower negative sentiment and a higher positive sentiment. More research is required into this to make sure it is accurate.

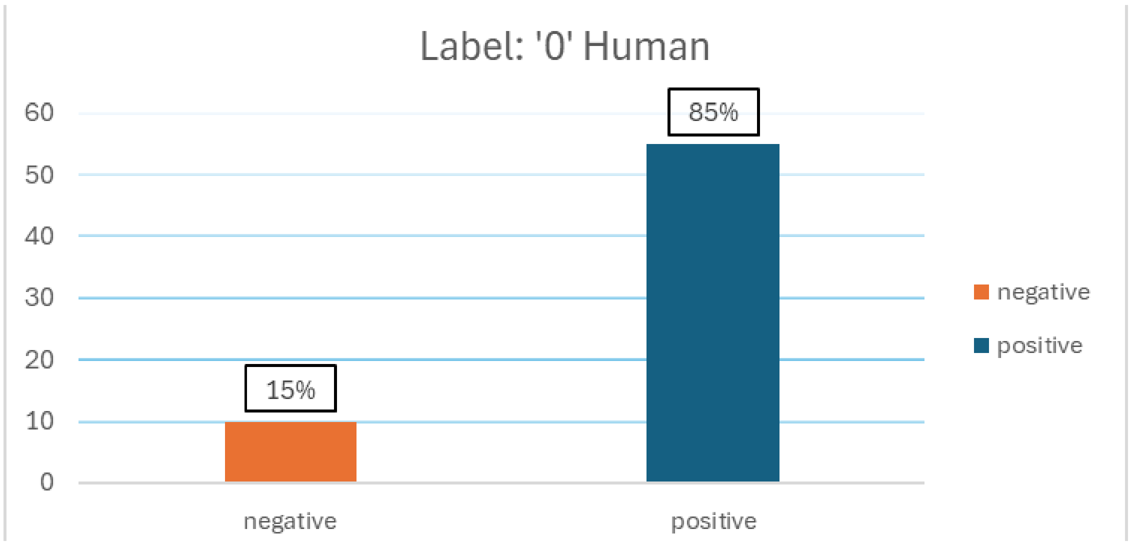

Figure 2: Sentiment Analysis on Human written passages. Indicates a higher negative sentiment.

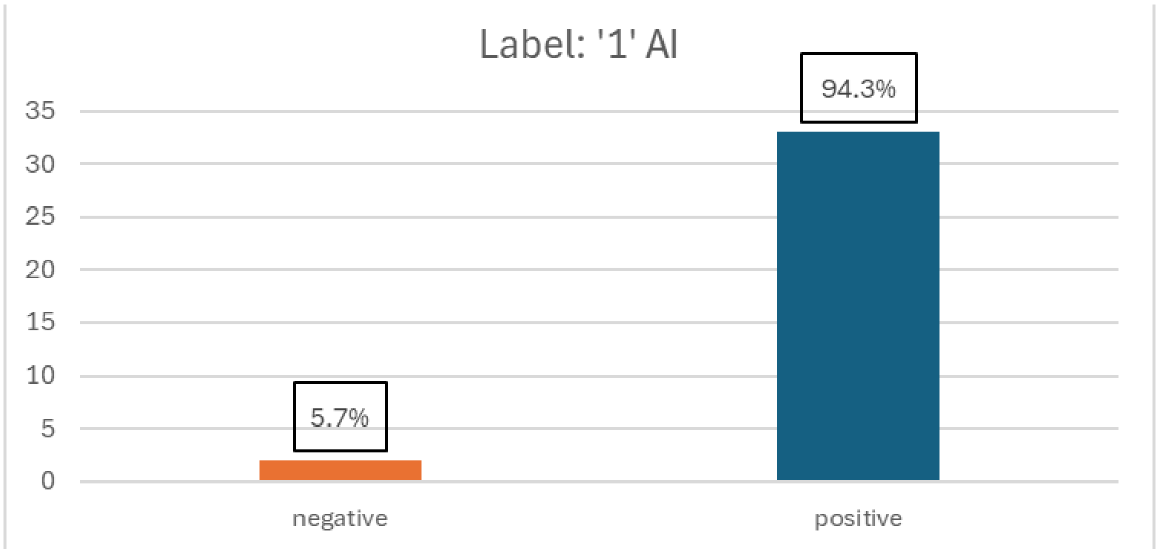

Figure 3: Sentiment Analysis on AI written passages. Indicates a lower negative sentiment.

4 Results and Discussion

In conducting a comparative analysis between BERT and Naive Bayes models for distinguishing AI-generated from human-generated text, it's evident that BERT show near perfect accuracy, with an impressive accuracy rate of 99.9% compared to Naive Bayes' accuracy of 95%. In terms of accuracy BERT classification is by far more accurate than Naive Bayes, if someone is building a detection model, they should consider using BERT for more accurate results. The downside of using BERT is its slow training process compared to Naive Bayes which took 63 seconds to complete training and testing vs 3 days 17 hours and 19 minutes on the same dataset. If one is planning on using BERT, they should consider having a powerful computer with perhaps a powerful GPU for fast and accurate results using BERT. As for Naive Bayes, a powerful computer can speed up the training process faster than 63 seconds. Our studies contribute to the growing need for accurate detection methods for plagiarism in schools and in daily life as we go into an age of AI generated content. Every content we can imagine today can and will be generated by an AI in the future. Having a model that can help detect it can help in making important decisions for various day to day lives.

5 Conclusion

In this study, we delved into the realm of AI-generated text detection, leveraging machine learning techniques to detect between passages crafted by artificial intelligence models and those authored by humans. Our investigation centered on the comparative analysis of Naive Bayes and BERT, shedding light on their respective strengths and limitations in addressing the evolving challenges posed by advanced AI language models. Our research shows the importance of robust text analysis techniques in the face of spreading AI-generated content. With the rise of sophisticated AI language models such as ChatGPT and Google's Bard, the need for reliable methods to differentiate between AI-generated and human-authored texts has become increasingly imperative. By evaluating the efficacy of Naive Bayes and BERT in this context and testing to see if Sentiment Analysis can help detect AIGT we contribute to advancing text analysis techniques and inform the development of more reliable and efficient text authentication processes.

6 Limitations

Despite the strides made in our research, several limitations still exist. The use of a single dataset and binary classification task may limit the generalizability of our findings across diverse text genres and AI models. Future research could explore more comprehensive datasets utilizing a wider range of text sources and writing styles. Additionally, investigating ensemble methods or hybrid approaches that integrate multiple classification techniques could further enhance the robustness and accuracy of AI-generated text detection models.

7 References

[1] *Chatgpt*, chat.openai.com. Accessed 2 Apr. 2024.

[2] "Introducing Chatgpt." *Introducing ChatGPT*, openai.com/blog/chatgpt. Accessed 2 Apr. 2024.

[3] *Google*, Google, gemini.google.com/. Accessed 2 Apr. 2024.

[4] "Copilot for Microsoft 365 - Business Plans: Microsoft 365." *Business Plans | Microsoft 365*, www.microsoft.com/en-us/microsoft-365/business/copilot-for-microsoft-365?ef_id=_k_Cj0KCQjw2a6wBhCVARIsABPeH1txUHWk9GGZJ6BFwholx_VpQWzyT1r-QF0SEvCfsuKkBz5SfIfd70gaAq-oEALw_wcB_k_&OCID=AIDcmm9xzw3cn3_SEM__k_Cj0KCQjw2a6wBhCVARIsABPeH1txUHWk9GGZJ6BFwholx_VpQWzyT1r-QF0SEvCfsuKkBz5SfIfd70gaAq-oEALw_wcB_k_&gad_source=1&gclid=Cj0KCQjw2a6wBhCVARIsABPeH1txUHWk9GGZJ6BFwholx_VpQWzyT1r-QF0SEvCfsuKkBz5SfIfd70gaAq-oEALw_wcB&rtc=1. Accessed 2 Apr. 2024.

[5] Xu, Shuo. "Bayesian Naïve Bayes classifiers to text classification." Journal of Information Science 44.1 (2018): 48-59.

[6] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

[7] Liu, Bing. Sentiment analysis and opinion mining. Springer Nature, 2022.

[8] Mindner, Lorenz, et al. "Classification of Human- and Ai-Generated Texts: Investigating Features for Chatgpt." *SpringerLink*, Springer Nature Singapore, 1 Jan. 1970, link.springer.com/chapter/10.1007/978-981-99-7947-9_12.

[9] Hu, Xiaomeng, et al. "Radar: Robust AI-Text Detection via Adversarial Learning." *arXiv.Org*, 24 Oct. 2023, arxiv.org/abs/2307.03838.

[10] Wang, Pengyu, et al. “SEQXGPT: Sentence-Level AI-Generated Text Detection.” *arXiv.Org*, 15 Dec. 2023, arxiv.org/abs/2310.08903.

[11] *(PDF) Detecting AI Content in Responses Generated by CHATGPT, ...*, www.researchgate.net/publication/372559876_Detecting_AI_content_in_responses_generated_by_ChatGPT_YouChat_and_Chatsonic_The_case_of_five_AI_content_detection_tools. Accessed 25 Mar. 2024.

[12] Wu, Junchao, et al. “A Survey on LLM-Generated Text Detection: Necessity, Methods, and Future Directions.” *arXiv.Org*, 24 Oct. 2023, arxiv.org/abs/2310.14724.

[13] Gerami, Shayan. “Ai vs Human Text.” *Kaggle*, 10 Jan. 2024, www.kaggle.com/datasets/shanegerami/ai-vs-human-text?resource=download.

[14] “Is Accuracy a Good Measure of Model Performance?” *Fiddler AI*, www.fiddler.ai/model-accuracy-vs-model-performance/is-accuracy-a-good-measure-of-model-performance. Accessed 2 Apr. 2024.

[15] “Precision.” C3 AI, 19 Sept. 2023, c3.ai/glossary/machine-learning/precision/#:~:text=Precision%20is%20one%20indicator%20of,the%20number%20of%20false%20positives).

[16] “Recall: A Key Metric for Evaluating Model Performance.” *Aporia*, 29 Feb. 2024, www.aporia.com/learn/recall-a-key-metric-for-evaluating-model-performance/.

[17] K, Joos. “The F1 Score.” *Medium*, Towards Data Science, 31 Aug. 2021, towardsdatascience.com/the-f1-score-bec2bbc38aa6.

[18] “The Python Standard Library.” *Python Documentation*, docs.python.org/3/library/index.html. Accessed 2 Apr. 2024.

[19] “Learn.” *Scikit*, scikit-learn.org/stable/. Accessed 2 Apr. 2024.

[20] “Pandas.” *Pandas*, pandas.pydata.org/. Accessed 2 Apr. 2024.

[21] “Torch.Library¶.” Torch.Library - PyTorch 2.2 Documentation, pytorch.org/docs/stable/library.html. Accessed 2 Apr. 2024.

[22] “Bert-for-Sequence-Classification.” PyPI, pypi.org/project/bert-for-sequence-classification/. Accessed 2 Apr. 2024.

[23] “Tfa.Optimizers.Adamw : Tensorflow Addons.” TensorFlow, tensorflow.org/addons/api_docs/python/tfa/optimizers/AdamW. Accessed 2 Apr. 2024.

[24] “Advisor.” Snyk Advisor, snyk.io/advisor/python/transformers/functions/transformers.get_linear_schedule_with_warmup. Accessed 2 Apr. 2024.

[25] Kumar, Vipin, and Basant Subba. "A TfidfVectorizer and SVM based sentiment analysis framework for text data corpus." 2020 national conference on communications (NCC). IEEE, 2020.

[26] “Sklearn.Naive_bayes.Multinomialnb.” Scikit, scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html. Accessed 2 Apr. 2024.

[27] “Vader sentiment.” *PyPI*, pypi.org/project/vaderSentiment/. Accessed 2 Apr. 2024.