Mining Textual Patterns: A Data-Driven Approach to Distinguish AI-Authored Content

Anirudh Prashant Kalghatkar
Department of Computer Science
University of Colorado Boulder
Anirudh.Kalghatkar@colorado.edu

Sagar Swami Rao Kulkarni
Department of Computer Science
University of Colorado Boulder
SagarSwamiRao.Kulkarni@colorado.edu

Pavan Sai Appari
Department of Computer Science
University of Colorado Boulder
Pavan.Appari@colorado.edu

Sachin Kashinath Rathod
Department of Computer Science
University of Colorado Boulder
Sachin.Rathod@colorado.edu

Nihal Srinivasu
Department of Computer Science
University of Colorado Boulder
Nihal.Srinivasu@colorado.edu

MOTIVATION

The introduction of sophisticated large language models (LLMs) has significantly transformed the generation and accessibility of written text, providing both opportunities and challenges across various domains. These advanced AI systems are capable of producing text that is often indistinguishable from that written by humans, revolutionizing fields such as content creation, customer service, and language translation. However, this technological advancement poses a significant challenge: differentiating between LLM-generated language and human-authored information. This challenge is particularly crucial in academic environments, where the integrity and originality of student work are of paramount importance. The capacity to discern whether an essay was written by a student or created by an LLM is critical for upholding academic standards and ensuring fair grading practices.

As LLMs become more sophisticated, traditional plagiarism detection methods, which typically rely on identifying similarities between texts, are losing their effectiveness. These classic approaches struggle to detect the nuanced and coherent outputs generated by modern LLMs, necessitating the development of more subtle and robust detection algorithms. This research effort addresses the critical need for reliable AI detection methodologies by leveraging a large dataset comprising both student essays and LLM-generated texts. By analyzing this dataset, the initiative aims to identify distinctive features and patterns that can be used to differentiate between human and AI-generated content.

The goal of this initiative is to enhance the tools available for maintaining academic integrity, providing educators and institutions with reliable methods to detect and address the use of AI in student work. By fostering open research and promoting transparency, this project not only aims to improve the detection of LLM-generated content but also contributes to the broader field of AI detection in real-world contexts. The outcomes of this research will be beneficial for instructors and educational institutions, helping to ensure that assessments reflect genuine student effort and learning. Furthermore, this work underscores the importance of adapting to technological advancements while preserving the core values of education and academic integrity.

LITERATURE SURVEY

The rapid advancement of natural language generation technologies, exemplified by OpenAI's ChatGPT and Google's LaMDA, has significantly enhanced the ability of AI systems to replicate human writing styles. These improvements in large language models (LLMs) have made it increasingly challenging to distinguish AI-generated text from human-authored content. This development raises concerns about potential misuse in academic dishonesty, misinformation, and phishing attacks. Consequently, there is a growing interest in developing robust methods for detecting AI-generated text to mitigate these risks.

Several approaches have been proposed to address this challenge. Mitchell et al. (2023) [1] introduced DetectGPT, a zero-shot detection method that analyzes the "probability curvature" of language model outputs to identify potential differences between human and AI-generated text. This approach is promising because it leverages the inherent properties of language models, offering an efficient way to detect AI-generated content without requiring extensive training data specific to each model.

In another effort, Rivera Soto et al.[2] proposed a method that uses writing style representations learned from human-authored texts to detect AI-generated content. This method has proven robust against new language models and can identify the specific language model used for text generation with minimal examples. By focusing on generalizable stylistic markers, this technique demonstrates versatility across different models and datasets. However, challenges remain, such as the long-term effectiveness of these methods against rapidly advancing AI capabilities and the computational costs involved. The need to continuously adapt to advancements in language models and manage the computational load underscores the ongoing research required in this field.

Additionally, frameworks like those discussed by Brundage et al. (2018)[3] highlight the dual nature of AI applications and advocate for collaborative efforts to mitigate risks associated with malicious AI use across various domains. They emphasize the potential security threats posed by AI in digital, physical, and political contexts and call for cooperation among researchers, governments, and the private sector to address these issues. While high-level strategies provide a broad direction, the rapid technological advancements necessitate continuous innovation and adaptation in detection techniques.

This evolving landscape demands more efficient and scalable methods to differentiate AI-generated texts from human-authored ones, ensuring the integrity and reliability of written content across different applications. The development of such methods is crucial not only for maintaining academic honesty but also for protecting the public from misinformation and malicious content. As AI capabilities continue to grow, so too must our efforts to develop sophisticated detection mechanisms that can keep pace with these advancements.

PROPOSED WORK

In our project, we aim to explore various data mining techniques to develop an effective system for distinguishing between AI-generated and human-written text. Our proposed approach includes several key components:

Data Exploration and Preparation: We will begin by conducting extensive Exploratory Data Analysis (EDA) on the "Augmented Data for LLM - Detect AI Generated Text" dataset from Kaggle. This will involve a thorough examination of the dataset to understand its structure, identify any anomalies, and detect potential class imbalances. Addressing class imbalances will be crucial for ensuring that our models are trained on balanced data, which is essential for robust model performance. Additionally, we plan to experiment with different data splitting strategies to ensure that our model development and evaluation are as rigorous and unbiased as possible.

Text Preprocessing: We propose to explore various text preprocessing techniques to prepare the data for model training. This may include traditional methods such as Term Frequency-Inverse Document Frequency (TF-IDF) and Count Vectorization, as well as more advanced techniques like word embeddings. By comparing these preprocessing methods, we aim to identify the techniques that best capture the nuances differentiating AI-generated text from human-written content.

Classification Algorithms: We plan to investigate a range of machine learning algorithms that have shown promise in text classification tasks. This includes traditional algorithms such as Support Vector Machines (SVM) and Logistic Regression, ensemble methods like Random Forest and XGBoost, and deep learning approaches potentially incorporating various neural network architectures. Our goal is to assess the strengths and weaknesses of each algorithm within the context of distinguishing between AI-generated and human-authored texts.

Model Development and Optimization: We will develop a systematic approach for model selection and hyperparameter tuning to ensure optimal performance. This will involve exploring various optimization techniques to enhance the accuracy and efficiency of our models. By systematically evaluating and refining our models, we aim to develop a highly effective system for detecting AI-generated text, contributing to the broader field of AI detection and supporting the integrity of written content in academic and other contexts.

EXPERIMENTS

Data segmentation, exploration and Preparation:

Data segmentation is a fundamental part of our experimental setup. We carefully select a representative subset from a larger dataset to capture the diversity and complexity of both AI-generated and human-written texts. This approach serves two main purposes: it allows for an efficient and manageable evaluation of preprocessing and classification methods, and it ensures that our models are exposed to a broad range of textual features. This is crucial for preventing overfitting and improving the generalizability of our results.

Our project utilizes the "Augmented data for LLM - Detect AI Generated Text" dataset from Kaggle, which includes 433,564 instances divided into 277,999 human-written essays and 155,565 AI-generated texts. This extensive and varied dataset forms the basis of our efforts to distinguish between human and AI-generated content, offering a balanced mix of styles, topics, and writing nuances.

The dataset's rich diversity ensures a comprehensive learning environment for our models, covering a wide range of linguistic features and writing complexities. For rigorous analysis, we split the dataset into training, validation, and testing sets, following a standard distribution of 60% for training, 20% for validation, and 20% for testing. This distribution is carefully chosen to maximize learning, allow effective model tuning, and ensure unbiased performance evaluation.

During the dataset preparation, we observed a significant imbalance in the distribution of class samples. This imbalance could negatively impact the performance and accuracy of predictive models, as models often become biased towards the more frequent classes. To address this issue and ensure equal representation of all classes, we employed a strategy to balance the dataset. Specifically, we randomly removed excess samples from the over-represented class until the number of instances in each class was approximately equal.

This approach not only maintains statistical integrity but also enhances the generalizability of the models by preventing overfitting to the dominant class. By doing so, we created a more equitable and robust dataset, better suited for rigorous machine learning analysis and applications. Figure 1 illustrates the initial imbalance within the dataset.

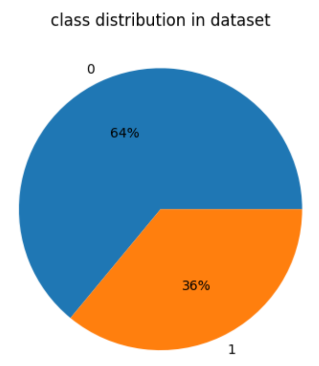

Figure 1: Class imbalance in the dataset. 0: human written text, 1: AI-generated text.

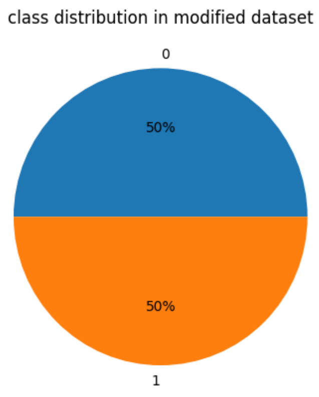

Figure 2: Class distribution in the modified dataset. 0: human written text, 1: AI-generated text.

Preprocessing:

Data preprocessing is an essential step that converts raw text into a structured format suitable for machine learning algorithms. In our project, we utilize two main preprocessing techniques provided by the sklearn library in Python:

TFIDF Vectorizer: The Term Frequency-Inverse Document Frequency (TF-IDF) Vectorizer measures the importance of words within texts relative to a larger corpus. This technique emphasizes terms that are unique to specific texts while reducing the weight of commonly occurring words. By doing so, it helps highlight distinctive features in the text. The TF-IDF Vectorizer is implemented using sklearn's TfidfVectorizer, offering a sophisticated feature extraction method. This technique is particularly valuable for distinguishing between human-written texts and those generated by artificial intelligence, as it identifies unique linguistic patterns and terminology.

Count Vectorizer: The Count Vectorizer provides a straightforward method based on word frequency. It counts the number of times each word appears in the text, capturing the basic occurrence of words. Despite its simplicity, this technique offers a valuable perspective on the textual data, allowing for the identification of common terms and overall word distribution. Sklearn's CountVectorizer converts the text into a numerical format that classification algorithms can easily analyze. This method complements the TF-IDF Vectorizer by providing a more basic, yet insightful, view of the text's structure.

Both preprocessing techniques are crucial for the success of our machine learning models. They ensure that the input data is well-structured and meaningful, facilitating accurate analysis and classification. By transforming raw text into numerical representations, these methods allow algorithms to effectively learn from and differentiate between various types of text data.

Using these preprocessing steps, we aim to enhance the performance and generalizability of our models, ensuring they can accurately analyze and classify both human-written and AI-generated texts. This comprehensive approach to data preprocessing forms the foundation of our experimental setup, enabling efficient and effective evaluation of different machine learning techniques.

LIMITATIONS OF DATASET:

Bias: AI-generated text can inherit biases from its training data, potentially skewing the dataset and any models trained on it.

Generalizability: AI-generated text may not accurately represent real-world human text, making it challenging for models to generalize to real-world scenarios.

Out-of-distribution data: Models trained on AI-generated text may struggle with data that significantly differs from their training data.

POTENTIAL ETHICAL CONSIDERATIONS:

Privacy Concerns: As the dataset has unclear origins it could mean inclusion of texts from private sources without proper anonymization.

False Positives and Bias: Models may misidentify human-written text as AI-generated and might be biased against certain writing styles or topics.

Transparency and Explainability: Understanding how the model identifies AI-generated text is crucial for assessing fairness and identifying biases.

CLASSIFICATION ALGORITHMS:

We implemented several classification algorithms to differentiate between AI-generated and human-written text. We evaluated traditional, ensemble, and deep learning methods to find the most effective approach. We used CountVectorizer and TF-IDF for preprocessing, each capped at 1000 features, to transform the text data into numerical representations for the models.

Algorithms that have been currently implemented are:

i) Support Vector Machine (SVM):

Support Vector Machines (SVM) are a powerful class of supervised machine learning models used to classify data into distinct categories. We chose SVMs because they are particularly effective in high-dimensional spaces, which is advantageous for text classification tasks where each dimension can represent a word or phrase from a large vocabulary. For our project, we set the C parameter to 0.15 to balance the trade-off between accurately classifying the training data and maintaining a simple decision boundary. Additionally, we opted for the RBF (Radial Basis Function) kernel, which is popular in SVM classification for handling non-linear relationships between class labels and attributes. The RBF kernel's flexibility allows for more complex decision boundaries based on distances in the feature space, which is crucial for capturing the intricate patterns often found in text data.

$$f(x)=\text{sigmoid}(w^T x+b)$$

where w is the weight vector, x is the feature vector, and b is the bias. This formula represents the decision boundary created by the SVM, which helps classify the data into distinct categories.

ii) Multinomial Naive Bayes (NB) Classifier:

We chose the Multinomial Naive Bayes classifier for its simplicity and efficiency, making it an excellent baseline model for text classification tasks. This variant of the Naive Bayes algorithm is particularly suited for handling discrete features, such as word counts. The algorithm assumes that features follow a multinomial distribution, which aligns well with text data where word frequencies are critical. The probabilistic nature of Multinomial Naive Bayes allows for straightforward interpretation of feature influences on classification outcomes. Its ease of implementation and effectiveness in handling text classification make it a valuable addition to our suite of models, providing a reliable benchmark for distinguishing between AI-generated and human-authored texts.

$$P(C_k\,|x)=\frac{P(C_k)\prod_{i=1}^{n}P(x_i|C_k)}{P(x)}$$

where $P(C_k)$ is the prior probability of class C_k, $P(x_i\,|C_k)$ is the likelihood of the feature x_i given class C_k, and $P(x)$ is the evidence. This formula calculates the posterior probability of a class given the features, which helps in making the classification decision.

iii) Logistic Regression:

Logistic regression is chosen for its simplicity and interpretability, making it a reliable choice for text classification tasks. We initialized our logistic regression models with adjusted parameters, including increasing the maximum number of iterations (max_iter) to ensure convergence. We specified the 'sag' solver, which is well-suited for large datasets due to its efficiency in managing sparse data and its faster convergence compared to other solvers like 'liblinear' or 'lbfgs'. To ensure a robust evaluation, we employed k-fold cross-validation, partitioning the dataset into k subsets and training the model k times, with each fold serving as the validation set once. This method provides dependable performance assessments, offering insights into the model's overall effectiveness and consistency. The mean cross-validation score estimates the model's average performance, while the standard deviation indicates the consistency across different folds. These metrics are critical for understanding the model's stability and reliability, guiding decisions related to model selection and hyperparameter optimization.

$$P(y=1|x)=\frac{1}{1+e^{-(w^T x+b)}}$$

where w is the weight vector, x is the feature vector, and b is the bias. This formula represents the logistic function used to estimate the probability of a binary outcome based on the input features.

iv) Random Forest:

We selected the Random Forest classifier for its robustness and capability to handle high-dimensional data effectively, making it suitable for text classification tasks. We trained the Random Forest model with 100 decision trees on the Bag-of-Words (BoW) vectorized training data and evaluated its performance using accuracy, precision, recall, and F1 score. These metrics are essential for assessing the model's effectiveness in binary classification tasks. By adjusting the number of decision trees to 100, we struck a balance between model complexity and computational efficiency. Utilizing the BoW representation for text vectorization, we captured the frequency of words in each document, aiding the classification process.

$$\text{Prediction}=\frac{1}{T}\sum_{t=1}^{T}\text{Tree}_t\,(x)$$

where T is the number of trees, and $\text{Tree}_t(x)$ is the prediction of the t-th tree. This formula aggregates the predictions from multiple decision trees to produce the final classification output.

To ensure the interpretability and explainability of our model's decisions in applications related to academic integrity, we focused on a transparent and structured approach. Initially, we performed thorough exploratory data analysis to understand the data and employed feature extraction methods like TF-IDF Vectorizer and Count Vectorizer to capture textual nuances. The choice of algorithms—such as Support Vector Machines (SVM), Multinomial Naive Bayes, Logistic Regression, and Random Forest was justified based on their suitability for text classification tasks.

Future Implementation of XGBoost and Neural Networks:

In addition to the above mentioned algorithms, we plan to implement XGBoost and neural network-based models to further enhance our classification capabilities. XGBoost is known for its gradient boosting framework that often provides superior performance through its ensemble approach and efficient handling of various types of data. Neural networks, particularly deep learning models, offer advanced pattern recognition and feature extraction capabilities, which could improve our classification accuracy. We will evaluate these models' performance to determine their effectiveness in distinguishing between AI-generated and human-written texts, potentially integrating them into our final solution.

Hyperparameter Tuning:

To ensure optimal performance, we performed hyperparameter tuning for each model implemented as follows:.

- SVM: Kernel type and regularization parameter (C) were tuned.
- MultinomialNB: The smoothing parameter (alpha) was adjusted.
- Linear Regression: Regularization parameters were optimized.
- Random Forest: The number of trees (n_estimators), maximum depth, and minimum samples split were fine-tuned.

Despite the extensive hyperparameter tuning, the SVM and MultinomialNB classifier exhibited lower performance, likely due to the sensitivity to the hyperparameters and the specific feature distribution. However, both Linear Regression and Random Forest models consistently demonstrated high accuracy and robust performance across both preprocessing techniques.

RESULTS

Accuracy, Precision, F1-Score and Recall of the model are of prime importance when evaluating the effectiveness of the classifier in discerning between human-written and AI-generated text.

Table 1 and 2 provide values of the metrics for SVM, Multinomial Naive Bayesian, Linear Regression and Random forest classifiers in two different preprocessing settings CountVectorization and TF-IDF respectively.

Classifier	Accuracy	Precision	F1-Score	Recall
SVM	54.4%	40.2%	47.7%	58.5%
MultinomialNB	63.7%	49.3%	60.4%	78.2%
Linear Regression	98.3%	97.3%	97.7%	98.0%
Random Forest	98.3%	98.6%	96.6%	97.6%

Table 1: Results of Classifiers with CountVectorization

Classifier	Accuracy	Precision	F1-Score	Recall
SVM	55.3%	41.5%	50.2%	63.6%
MultinomialNB	66.7%	52.2%	61.1%	73.6%
Linear Regression	98.5%	97.6%	97.9%	98.1%
Random Forest	98.2%	98.1%	96.8%	97.5%

Table 2: Results of Classifiers with TF-IDF

After preprocessing with either TF-IDF or Count vectorization, with a maximum of 1000 features, the majority of the classifiers that have been tested thus far show high accuracy rates. The SVM stands out, nonetheless, because of its substantially poorer accuracy. The structure of the TF-IDF representation and the sensitivity of an SVM to the selected hyperparameters may have contributed to this disparity, as they may not have been ideal for the distribution of the data in feature space. Even with this SVM and TF-IDF anomaly, the overall performance trends are in line with expectations, demonstrating the effectiveness of the used techniques in text classification tasks.

EVALUATION

Our suggested approaches will be quantitatively evaluated using standard performance metrics such as accuracy, precision, recall, and F1-score to ensure a comprehensive assessment of our models.

Accuracy will measure the overall correctness of the model by calculating the ratio of correctly predicted instances to the total number of instances. This metric provides a broad view of the model's performance but may not fully capture its effectiveness, particularly in cases where class imbalances exist.

Precision will evaluate the quality of the model's positive predictions by calculating the ratio of true positive predictions to the total number of positive predictions (both true and false positives). High precision indicates that the model has a low false positive rate, meaning it is adept at predicting instances of a class without mistakenly labeling irrelevant instances as part of that class.

Recall (or sensitivity) will measure the model's ability to identify all relevant instances within a specific class by calculating the ratio of true positive predictions to the total number of actual positive instances (both true positives and false negatives). High recall indicates that the model successfully captures most of the relevant instances, though it may come at the expense of increasing false positives.

F1-Score provides a harmonic mean of precision and recall, balancing the two metrics to give a single measure of performance that accounts for both false positives and false negatives. The F1-score is particularly useful when the data has class imbalances, as it provides a more nuanced view of the model's effectiveness by considering both precision and recall simultaneously.

By employing these metrics, we can comprehensively evaluate the performance of our models. Accuracy will give us a general sense of correctness, precision will inform us about the reliability of our positive predictions, recall will show us how well our model captures all relevant instances, and the F1-score will provide a balanced measure of the model's overall performance in both precision and recall scenarios. This multifaceted evaluation will ensure that our model not only performs well overall but also excels in distinguishing AI-generated text from human-written content across various contexts.

MILESTONES

In **Week 1**, we will conduct a kickoff meeting to assign roles, followed by an in-depth literature review. We will define the project's scope and create a timeline. Additionally, we will collect datasets and set up the necessary tools.

In **Week 2**, we will define and implement feature extraction methods, select initial models, and train and evaluate these models while documenting baseline performance.

In **Week 3,** we will focus on training and comparing advanced models, evaluating and validating them, and conducting error analysis to refine the models.

In **Week 4**, we will conduct the final model evaluation and compare results, create visualizations, compile findings into a final report, and prepare and rehearse our presentation.

Throughout the project, we will have regular check-ins and maintain continuous documentation to ensure smooth progress and thorough record-keeping.

CURRENT STATE OF WORK

Firstly, a subset of data has been extracted and both tf-idf and count vectorizer methods have been applied.

SVM has been built to establish the baseline performance level on this task.

MultinomialNB, Logistic Regression and Random forest are built with a comparable set of evaluation metrics defined.

CHALLENGES

During the initial stages of our project, we encountered several challenges, particularly in the preprocessing phase. Preprocessing the dataset was more time-consuming than initially anticipated.Hence we had run the model using the google colab GPU for the MultinomialNB and SVM model which reduced the time to train the models .

We experimented with various preprocessing methods, including Term Frequency-Inverse Document Frequency (TF-IDF) and Count Vectorization, to capture the nuances that differentiate AI-generated text from human-written content. The process of selecting the appropriate preprocessing techniques was critical, as it significantly impacts the performance of our models.

One of the significant challenges in applying our methods to different types of text or AI models is the variability in text characteristics across different domains. Texts from diverse fields such as literature, technical writing, and social media possess unique stylistic and structural features, which complicates the development of a one-size-fits-all detection model. The intricacies of language use in different contexts necessitate customized preprocessing steps and feature extraction methods tailored to capture the specific attributes of each text type. This variability requires us to continually refine our models and preprocessing techniques to ensure robustness and accuracy across varied datasets.

Exploring and selecting suitable machine learning (ML) models also proved to be a complex task. We evaluated a range of classification algorithms, including traditional methods like Support Vector Machines (SVM) and Logistic Regression, and ensemble methods such as Random Forest. This process was iterative and required extensive experimentation to identify the models that offered the best performance for our specific dataset and objectives. Moreover, the rapid evolution of AI models introduces an additional layer of complexity, as newer models may exhibit text generation patterns that differ significantly from those seen in previous models. This necessitates ongoing updates and recalibrations of our detection systems to keep pace with advancements in AI-generated text.

Additionally, we delved into exploring neural network architectures, recognizing the potential of deep learning (DL) to provide more robust detection mechanisms. This exploration involved assessing various neural network designs to determine their suitability for our classification task.

CHANGES

As of now we haven’t received any feedback on the project proposal , and assuming that we are in the right direction on this project , we haven’t changed our main goal and will be continuing our project according to the timeline we had worked on.

One significant modification was the introduction of the Multinomial Naive Bayes (MultinomialNB) classifier. This algorithm, known for its effectiveness in text classification tasks, was integrated into our model selection process to enhance our detection capabilities.

We are also planning to focus towards more advanced DL techniques. We have begun exploring and selecting neural network architectures, aiming to leverage their ability to capture complex patterns in the data. This shift was driven by the need for more sophisticated algorithms to handle the intricacies of distinguishing AI-generated text.

To ensure comprehensive evaluation, we finalized the use of TF-IDF and Count Vectorization methods for each classification algorithm. These vectorization techniques were chosen for their ability to effectively transform text data into meaningful numerical representations, essential for model training and evaluation.

REFERENCES

[1] Mitchell, Eric, et al. "Detectgpt: Zero-shot machine-generated text detection using probability curvature" International Conference on Machine Learning. PMLR, 2023.

[2] Soto, Rafael Rivera, Kailin Koch, Aleem Khan, Barry Chen, Marcus Bishop, and Nicholas Andrews. "Few-Shot Detection of Machine-Generated Text using Style Representations." arXiv preprint arXiv:2401.06712 (2024).

[3] Brundage, Miles, et al. "The malicious use of artificial intelligence: Forecasting, prevention, and mitigation." arXiv preprint arXiv:1802.07228 (2018).