ML and DL Based LLM-Human Generated Text Classification

[1,4]Mehmet Eren Soykök, b2200765016@cs.hacettepe.edu.tr
[1,3]Harun Harman, harunharmann@gmail.com[1],
[1,2,] Tugba Gurgen Erdogan, tugba@cs.hacettepe.edu.tr
[1] Hacettepe University, Computer Engineering, Ankara, Turkey
[2] Hacettepe University, Computer Engineering, Software Engineering Research Group

Abstract

While the work in the NLP field is getting more and more successful, some contradictions occur. In the field of text generation by Large Language Models (LLMs), the created texts became difficult to understand whether the texts were written by a human or not. To overcome this contradiction, this project proposes a Machine Learning based solution which classifies the given text in terms of the writer: Human or LLM. This project includes different data preprocessing and NLP techniques, different model developments such as logistic regression, naive bayes, XgBoost and LSTM and their experiments. The results of each experiment are analyzed and discussed. At the end 94-98% accuracy is achieved.

Keywords: Large Language Model, text generation, NLP

1. INTRODUCTION

With the rapid and significant advancements in LLM development, LLMs have become increasingly human-like in their text generation capabilities. These advancements have propelled the field forward, enabling LLMs to produce more nuanced, contextually appropriate, and coherent responses. As a result, they hold promise for revolutionizing various domains, from natural language processing to content creation and beyond. However, alongside these strides come important considerations regarding ethical implications, biases, and potential misuse, underscoring the necessity for responsible development and deployment of this technology.

One of the ethical considerations in this field is to usage purpose of the LLM's. They could be used in article generation, doing assignment, preparing report etc. which are expected to be written from a person. In this work, we proposed to detect if the given text is generated by an LLM or a human. Even though LLMs are advanced models as a result of machine learning, we started to take this into consideration with the project "The solution to the ethical result that emerges can still be realized with the machine learning method."

While working on this project, we followed the Cross Industry Standard Process for Data Mining (CRISP-DM) methodology. This methodology includes five stages: Business understanding, data understanding, data preparation, modeling, and evaluation. This methodology and its stages are explained in detailed in section 3. For the project, since there are many different LLM exists, we tried to collect data generated from different models and different sources. Then, we tried different approaches for preprocessing and embedding (such as TF/IDF or Word2vec) the data. This helped see the effects of existing algorithms on real data. Then different machine learning models are applied to the prepared data. After those steps, we discussed, compared, and analyzed the results.

The rest of the paper includes Section 2: background information and related work about detecting ai generated text and why we need it. Section 3 explains our approach and methodology to the problem. Section 4 provides the details of the experimental phase, the environment, and the discussion of the results. Section 5 is the conclusion of this data science project.

2. BACKGROUND AND RELATED WORK

Table 1: Summary of the related work

Study	Features	Classifiers	Performance	Datasets
[1] Sentiment analysis and random forest to classify LLM versus human source applied to Scientific Texts	**Text, Label**	**Random Forest Classifier**	**Accuracy: 84.14% RMSE: 0.3724**	**Text From Journals and ChatGPT**
[2] Adaptive Ensembles of Fine-Tuned Transformers for LLM-Generated Text Detection	**Text, Label**	**Transformers (NN), Random Forest (RF), Gradient Boosting Decision Trees (GBDT)**	**NN-Accuracies: 0.992 , 0.736 RF-Accuracies: 0.992 , 0.722 GBDT-Accuracies: 0.992 , 0.718**	**DAIGT, Deepfake**
[3] Detection of AI-Generated Text Using Large Language Model	**Text, Prompt, Label**	**Neural Network (Encoder+Discriminator Network+ Watermark Decoder Network)**	**Test ROC AUC Scores: 0.9949-0.9989**	**ChatGPT-3, Wikipedia, Reddit**

In the paper of J.Sanchez [1] during the data pre-processing step, stopwords were deleted from the data, and stemming was applied. Four different lexicon methods were applied to the remaining words. These are Bing, Afinn, Nrc, and Loghran-Mcdonald methods. With these methods, words took certain values according to the method applied. The random forest model, one of the ensemble classification models, was applied. Z.Lai et al. [2] tried 3 different methods. The first of these is to use a single classifier, the other one is to train with a non-adaptive ensemble based on the Hard voting method, and the last one is to apply ensemble learning based on a Neural network and Random Forest. The hard voting method gave more successful results than the single classifier method. The best results were achieved when the ensemble learning method was used. At the same time, overfitting was avoided due to the nature of this method. M. Prajapati et al. [3] proposed 2 different approaches. Pitch-black-box and colorless-box detection techniques are used with deep learning method. Due to the development of LLMs, the Pitch-black-box approach is more restrictive and therefore gives lower results than colorless-box detection.

3. METHODOLOGY

In this project, CRISP-DM Methodology is followed. Below, the diagram of the method is presented.

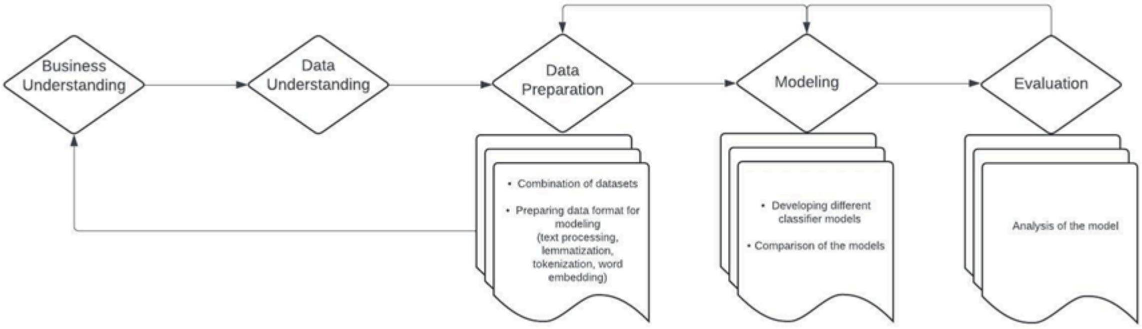
Business Understanding
Data Understanding
Data Preparation
Modeling
Evaluation
• Combination of datasets
• Preparing data format for modeling (text processing, lemmatization, tokenization, word embedding)
• Developing different classifier models
• Comparison of the models
Analysis of the model

Figure 1: CRISP-DM Methodology Diagram.

3.1. Business Requirements

This phase includes understanding the problem and determining the goals. The problem is the advancements in the LLM that could create conflicts with human writing. The goal of this project is to develop an essay-detecting system by using Natural Language Processing to determine whether the essay is generated by AI or not. Increasing the accuracy rate to distinguish artificial intelligence-generated essays from real human essays.

3.2. Data Requirements

This phase refers to the specific needs of the data that help to solve the problem. These needs could be various such as data types, data quality, data volume, data sources, data integration, and data accessibility etc. In this project the data requirements are as follows:

- Data Type: Since this is an NLP project, the data type we worked with is strings.
- Data Source: We benefit from public datasets that are already labeled. Also, different sources are taken into account. Also, the dataset should include texts generated from

different LLM models and different person’s article/texts because in language usage of different LLMs and different people are also unalike.

- Data Volume: For more generalization, high volume of data is needed.

3.3. Model Requirements

In this step, the goal is to choose the best model for the problem and data. We want to solve a problem based on applying NLP techniques and using machine learning models for classification. Since our problem is a binary classification problem, it would be the right step to use logistic regression or a decision tree. Also, many other experiments are done using deep learning and machine learning-based models.

3.4. Dataset and Features

First of all, in order to ensure the healthy training of the model, we aimed to keep the dataset size as much as possible. To achieve this, we merged multiple datasets. Rather than containing essays solely on a single topic, the datasets contain essays on various topics. This is crucial for the model to generalize and for performance to evaluate. The datasets utilized various competitions so they do not contain unlabeled data. Thus, it can be a resource for supervised learning. Although there is a numerical difference in the number of examples between the two classes in the merged dataset, there are sufficient examples for both classes.

3.5. Data Exploration and Understanding

There were several datasets utilized. DAIGT Proper Train Dataset [4], DAIGT V2 Train Dataset [5] and LLM-Detect AI Generated Text Dataset [6] are combined. The "text" and "label" columns make up the combined dataset. The column labeled "text" contains an essay. This is the foundation for the model’s input. For binary classification, the values 0 or 1 are present in the "label" column.

Also, this column is used for class labels. The dataset consists of 188.601 different texts and each text has a different length. Class distribution and boxplot of text length figures are in Figure 2 and Figure 3.

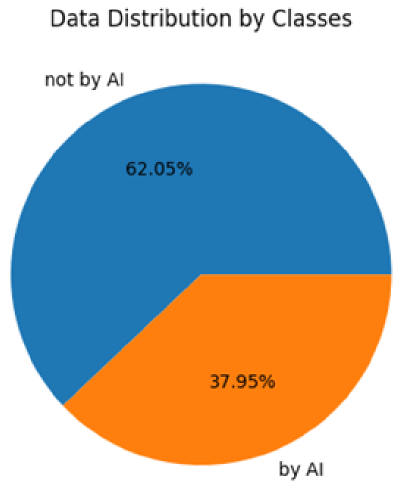

Figure 2: The distribution of classes.

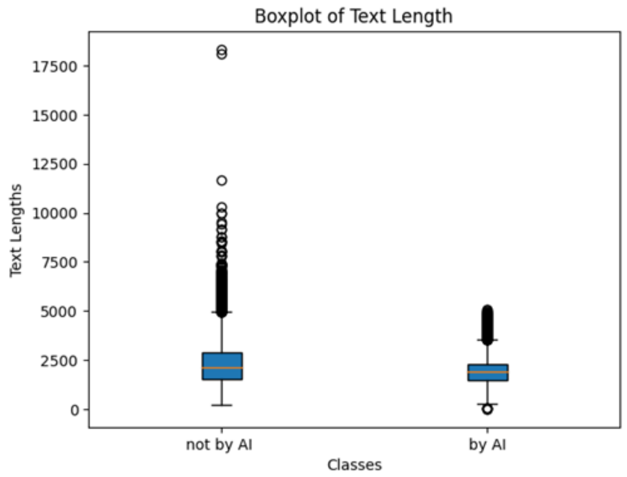

Figure 3: The length plot of each text in data.

Also the text length distribution according to label can be shown on the Figure 4.

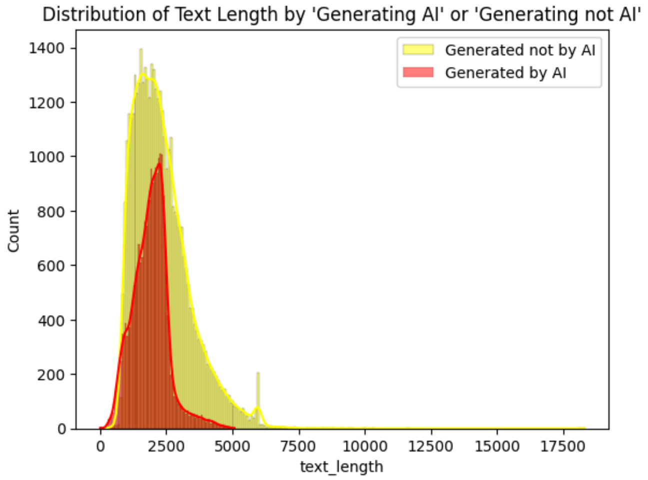

Figure 4: Text Length Distribution According to Labels.

.

3.6. Data Preprocessing

In this phase, we cared about the specific structural differences in LLM-generated texts and human-generated texts. Since our approach needs NLP applications, we applied some basic preprocessing methods. Two different approaches are taken into consideration: Word2vec embedding, and TF-IDF vectorization.

3.6.1. Word2vec Based Approach

First, we checked the null values for the dataset and found out that there were no null values. After this control, we converted the capital letters in the sentences to lowercase letters.

Because for example words “Book” and “book” are considered differently according to the models. Then, we removed the punctuation and the special characters from the texts because this helps reduce the noise in the texts and also in English punctuation determines the boundary of some of the words. In other words, it helps standardization of the text. After these standardization steps, we tokenized the text word by word using “ nltk ” library. Moreover, the tokenized words had a different structure. That’s why we applied lemmatization to every token to gain the root of the words so that no conflict happens. After applying lemmatization, while we were controlling the tokens, we noticed that there were some words could not be transformed into the root version. To overcome this problem, we used the approach “POS (Part of Speech) Tagging”. This method tags the input word as adjective, adverb, noun etc. and then these tags help the lemmatizator to understand the root of the word. For instance, knowledge of the word “running” is a verb helps in finding the root “run”.

The previous operations were about text processing. The next and the last step is to turning these texts into a numeric structure called “embeddings”. There are several ways to gain embeddings. We applied “Word2Vec” embedding. This approach captures semantic similarities between words by mapping them to dense vector representations where similar words are positioned closer to each other in the vector space.

3.6.2. TF-IDF Based Approach

Another approach is using TF-IDF to turn texts into a numeric value. TF-IDF is a method for calculating the frequency of the words in a text. It is used to extract its importance in context. The first N common words are used to train models.

First, punctuation marks and numbers were cleared from the dataset. Each word within the essays was tokenized and stopwords were also removed. The lemmatization process was applied to the remaining words according to their characteristics such as nouns, verbs and adverbs. Finally, the TF-IDF vectorization process was applied to the remaining words and the frequency values of each word were found. Using the remaining words, the total number of each word in the dataset was found and the first N of them was selected. With these first N words, X and Y datasets were created first. It was then divided into train, validation, and test sets.

After the preprocessing, the word cloud with a meaningful one can be shown in Figure 4 below.

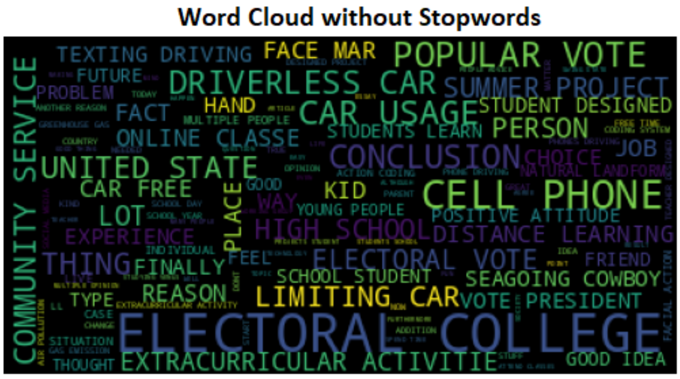

Figure 5: Word Cloud

3.7. Model Evaluation

Algorithm Selection/Model Design: Machine learning methods such as logistic regression, naive bayes, XGBoost, and deep learning methods based on LSTM are used.

Accuracy, loss and AUC plot for LSTM using top N=25, N=50 and N=100 words can be shown figures below.

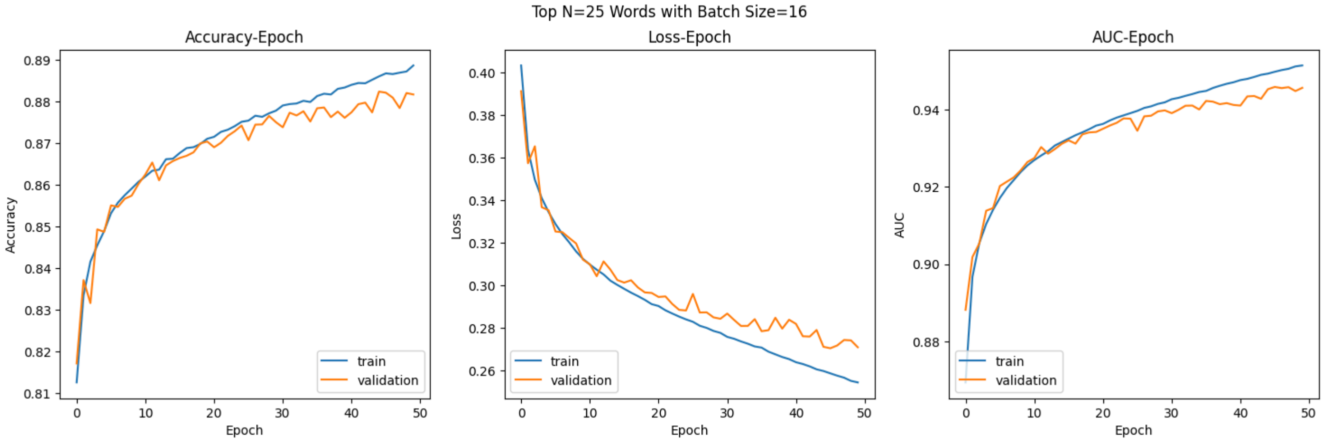

Figure 6: LSTM Train and Validation Plots using Top N=50 Words

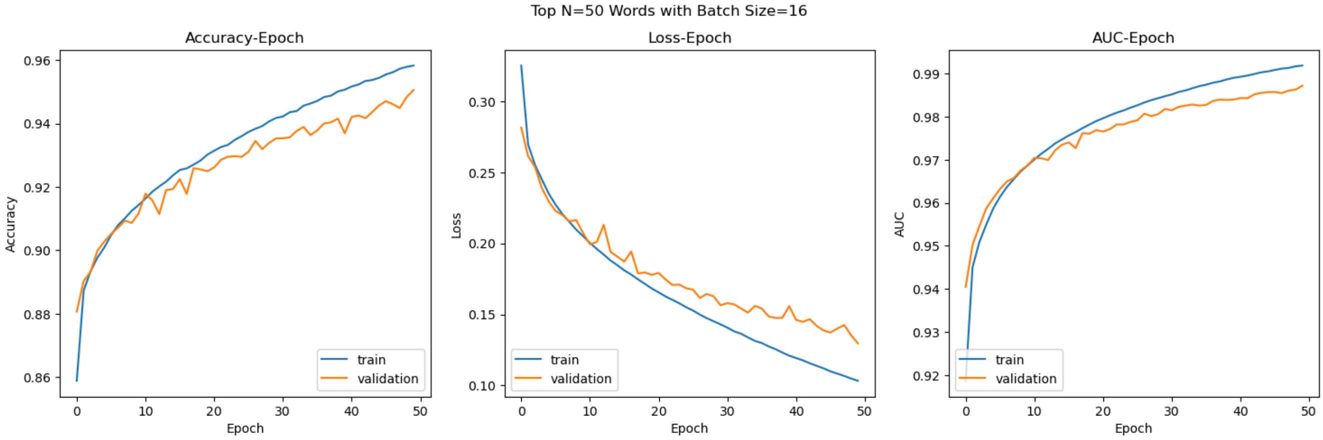

Figure 7: LSTM Train and Validation Plots using Top N=50 Words

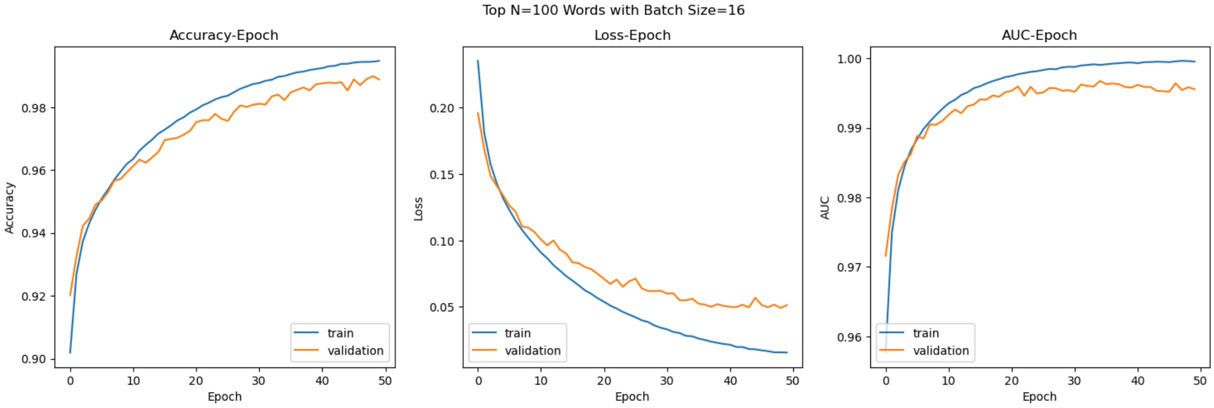

Figure 8: LSTM Train and Validation Plots using Top N=100 Words

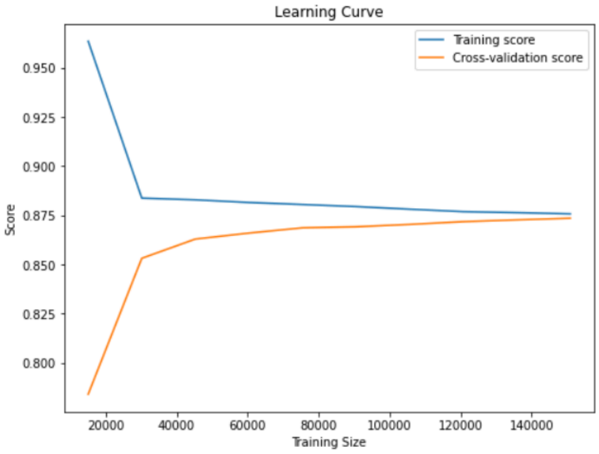

Figure 9: Logistic Regression Learning Curve

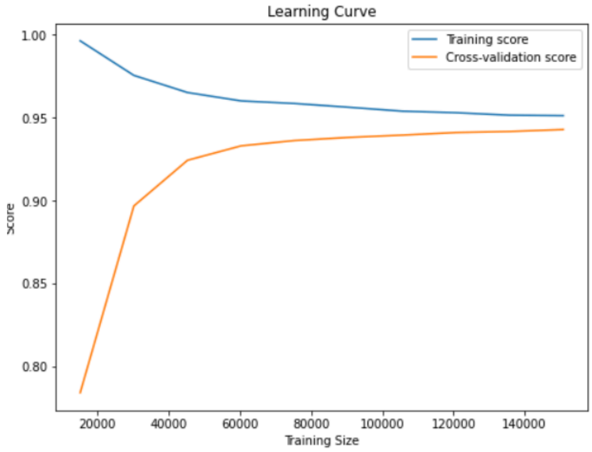

Figure 10: XGBoost Classifier Learning Curve

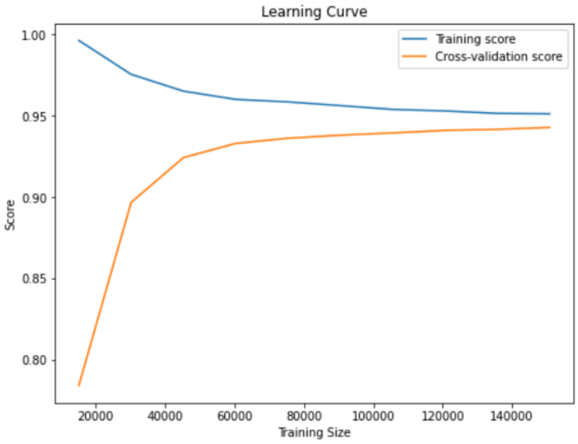

Figure 11: RandomForest Classifier Learning Curve

4. EXPERIMENTS

4.1. Experimental Setup

For this work, we preferred the Jupyter Notebook environment and our local machine. Because the local machine's hardware system was enough for the scope of the project.

On the other hand, the project is run on Python, with the below frameworks and libraries:

- Pandas
- Numpy
- NLTK
- Gensim
- Sklearn
- Joblib
- Matplotlib
- Wordcloud
- Tensorflow
- Keras
- XGBoost
- Seaborn

Environment: Nvidia RTX 3060, 16 GB RAM, Nvidia GTX 1650 16 GB RAM

LSTM Architecture

LAYER (TYPE)	OUTPUT SHAPE	PARAMS
lstm_1 (LSTM)	(None, 1 , 128)	117248
dense (Dense)	(None, 1 , 64)	8256
dense_1 (Dense)	(None, 1 , 1)	56

Table 2: Architecture of LSTM.

The architecture of the LSTM model is shown on the graph above. There is a LSTM layer with 128 layers. One dense layer is used for the hidden layer with 64 layers and one dense layer is used for the output layer. Adam optimizer with 0.001 learning rate is used. The ReLU and sigmoid activation functions are used respectively. There is no weight decay for the optimizer. Binary-cross entropy is used as a loss metric. The batch size is 16. The model is trained for 50 epochs.

4.2. Experiment Results

4.2.1. TF/IDF Based Approach Results

	Train Loss	Train Accuracy	Train AUC	Train F1 Score	Validation Loss	Validation Accuracy	Validation AUC	Test Loss	Test Accuracy	Test AUC	Test F1 Score
N=25											
Logistic Regression		0.805		0.642					0.803		0.641
Naive Bayes		0.705		0.000					0.703		0.000
XGBoost		0.901		0.830					0.889		0.811
LSTM	0.255	0.888	0.951		0.271	0.882	0.946	0.272	0.880	0.945	
N=50											
Logistic Regression		0.846		0.730					0.842		0.725
Naive Bayes		0.703		0.009					0.700		0.010
XGBoost		0.946		0.909					0.937		0.893
LSTM	0.101	0.960	0.992		0.130	0.951	0.989	0.136	0.948	0.986	
N=100											
Logistic Regression		0.897		0.821					0.894		0.818
Naive Bayes		0.733		0.224					0.732		0.231
XGBoost		0.977		0.961					0.9674		0.945
LSTM	0.020	0.993	0.999		0.052	0.988	0.996	0.048	0.989	0.997	

Table 3: Comparison of the developed models.

4.2.2. Word2Vec Based Approach Results

	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.88	0.87	0.88	0.87
XGBoost Classifier	0.94	0.94	0.94	0.94
Random Forest Classifier	0.94	0.94	0.94	0.94

Table 4: Model results with test set.

5. CONCLUSION

Looking at the models, it is clear that the deep learning method works much better as expected. The XGBoost model, built on the basis of decision trees, also came in second best, with no bad results. In the most used N-word based dataset approach, as the number of N increased, the accuracy rate increased for each model, and the loss value decreased significantly in deep learning.
Choosing this number appropriately when choosing the most common N words is an important factor for calculation cost and calculation time. The choice should be made as a trade-off after a few tries. Better results can be obtained by using a larger data set where the main topics on which essays are written are more diverse.

6. REFERENCES

Write references manually or Use mendeley desktop and word plugin to manage your references: *https://www.mendeley.com/download-reference-manager/macOS*

[1] Javier J. Sanchez-Medina: “Sentiment analysis and random forest to classify LLM versus human source applied to Scientific Texts”, 2024

[2] Zhixin Lai, Xuesheng Zhang, Suiyao Chen: “Adaptive Ensembles of Fine-Tuned Transformers for LLM-Generated Text Detection”, 2024;

[3] M. Prajapati, S. K. Baliarsingh, C. Dora, A. Bhoi, J. Hota and J. P. Mohanty, "Detection of AI-Generated Text Using Large Language Model," 2024 International Conference on Emerging Systems and Intelligent Computing (ESIC), Bhubaneswar, India, 2024, pp. 735-740, doi: 10.1109/ESIC60604.2024.10481602.

[4] Kleczek, D. (2023). DAIGT Proper Train Dataset, Version 4. Retrieved March 2024 from https://www.kaggle.com/datasets/thedrcat/daigt-proper-train-dataset

[5] Kleczek, D. (2023). DAIGT V2 Train Dataset, Version 2. Retrieved March 2024 from https://www.kaggle.com/datasets/thedrcat/daigt-v2-train-dataset

[6] Thite, S. (2023). LLM - Detect AI Generated Text Dataset, Version 1. Retrieved March 2024 from https://www.kaggle.com/datasets/sunilthite/llm-detect-ai-generated-text-dataset/data

APPENDIX