Monolingual Black-Box Machine-Generated Text Detection

Sagar Swami Rao Kulkarni, Shrestha Acharya, Upvandeep Kaur

Abstract

This paper describes our contribution to SemEval-2024 Task 8: Multigenerator, Multidomain, and Multilingual Black-Box Machine-Generated Text Detection. This project is a test of conventional models for the task of classifying text as human-written or machine-generated and understanding the significance of complex models over traditional methods. Our results show that although modern solutions – complex transformers such as Large Language Models have their own challenges, turn out to be better at identifying the origin of full texts.

1 Introduction

SemEval-2024 Task 8 involves binary classification of full texts to ascertain whether they are human-written or machine-generated. The original task overview has been described in the paper[1].

There are two distinct tracks within Subtask A: the monolingual track, focusing exclusively on English sources, and the multilingual track, encompassing multiple languages. This paper focuses on solving the monolingual track for Subtask A. The goal is to implement classifiers to differentiate human-written text from machine-generated ones. This project also compares performances between different traditional models and complex transformers such as LLMs to understand why the latter is suitable for discerning between content generated by humans and that generated by models such as ChatGPT and Davinci.

As the prevalence and accessibility of LLMs continue to rise, their potential for producing seamless and contextually relevant content across a variety of platforms (such as news, social media, and education) cannot be ignored. However, their ability to mimic human language also raises valid concerns about misuse, such as the spread of misinformation and the creation of deceptive automated content. Therefore, it is imperative that we develop efficient methods for automatically distinguishing between machine-generated and human-written text to mitigate the potential risks associated with the widespread use of LLMs. Before building newer solutions, we should understand why and how adding complexity to a machine learning model improves performance for the task at hand. This paper thus aims to understand what necessitates the use of LLMs for differentiating between human-written and machine-generated texts.

In the task of determining whether a given full text is human-written or machine-generated, this paper explores the application of various traditional models such as Linear SVM, Random Forest, AdaBoost with Decision Tree, XGBoost, Naïve Bayes and some pre-trained language models such as RoBERTa Base and RoBERTa Large. Language models like BERT and its variations have undergone extensive training on vast amounts of text, showing impressive abilities in comprehending and producing text that mimics human language. Examining and contrasting these models is vital in discerning their individual strengths and weaknesses in tackling the specific challenges of a given task. This analysis enables us to make informed choices when selecting the most suitable model for a classification problem.

LLMs present unique challenges when it comes to computational resources. As we embark on our efforts, we encountered several struggles and considerations:

[1] https://arxiv.org/abs/2305.14902

Computational Power: The training and inference processes for RoBERTa Base and RoBERTa Large models demand significant resources. To fine-tune them, we require access to high-performance computing resources, such as A100 and V100 GPUs with high RAM and infrastructures such as the Google Colab Pro environment , which we specifically obtained for this project.

Memory Constraints: As large models possess a significant number of parameters, it results in a substantial need for memory during inference. This can present a difficulty for systems with restricted access to GPUs or RAM. For example, attempting to run RoBERTa models on a local system would prove unfeasible.

Availability of Resources: While we did acquire a GPU through Google Colab for our project, it is important to note that the resources offered in this plan may not always be accessible. For instance, the A100 GPUs may not be allocated immediately upon request whenever required. Additionally, there may be instances where we encounter warning messages indicating that the allocated compute resource will be deallocated soon, when attempting to run our notebook.

Cost Considerations: The process of refining large models can come at a high cost, involving both significant amounts of time and financial resources. For this project, one may consider subscribing to cloud services such as Google Colab Pro edition or investing in dedicated hardware solely for training purposes. However, budget limitations may restrict the scope of experimentation.

Model Size vs. Accuracy Trade-off: One way to address the challenge of resource demands is utilizing smaller conventional models to find a solution. However, this approach does involve a trade-off, as it may sacrifice a degree of precision when it comes to predicting the source of texts.

Batch Size and Training Time: Higher batch sizes may take less time to train and contribute towards model performance but require higher computational resources. Thus, it is crucial to strike a good balance.

Fine-Tuning Challenges: To successfully fine-tune LLMs for specific tasks, a significant amount of annotated data and multiple training rounds may be required. Such resources can be quite demanding, making it crucial to find the right balance between model complexity and available means.

To address these struggles, we considered optimization techniques and hyperparameter tuning. Additionally, modelling strategies like ensemble techniques (Random Forest in this project) combine the outputs of multiple models - decision trees, to reach a single result. This led to better outcomes even with a traditional model.

The code for testing conventional models against pre-trained models and the visualizations to showcase their comparison can be viewed at our GitHub repository[2].

2 Background

The data used for this task is a series of texts in English language along with their respective labels. Each text in the list of texts is a "full text" meaning it is either completely human-written or completely machine-generated. There is no case where the text is a mixture of both human-written and machine-generated. Each text purely belongs to one and only one category of the two categories. Each category is represented by a number from 0 and 1. 0 represents human-written texts and 1 represents machine-generated texts. The machine-generated texts are from various sources such as – ChatGPT, Davinci, Cohere and Dolly. The distribution of sources across these sources has been shown with the help of a pie chart in Fig 2.1.

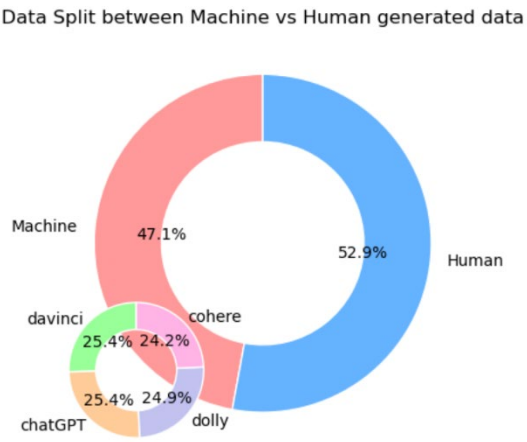

Fig 2.1 : Distribution of texts as per their sources

[2] https://github.com/upka7379/NLP-Shared-Task-8/tree/main

Nearly half of the dataset, 47% or 56k records, is a list of machine-generated full texts and the rest 63k records are human-written. One example of a segment of one of the texts from the list of machine-generated full texts looks as follows:
"Forza Motorsport is a popular racing game that provides players with the ability to race on various tracks and in different vehicles…"
The above text is sourced from WikiHow generated by model ChatGPT and has label value equal to 1. Similarly, the other models Davinci, Cohere and Dolly use different data sources and generate texts. On the other hand, a segment of one of the texts from the list of human-written full texts looks as follows:
"If you're a photographer, keep all the necessary lens, cords, and batteries in the same quadrant of your home or studio..."
Thus, we see there are no spelling or grammatical errors that can help to distinguish the two categories easily. However, the set of frequently used words may differ amongst the two categories and should be explored.

3 System Overview

For this task, the text classification system employed was a holistic combination of traditional machine learning models and advanced transformer-based architectures. The flow diagram adapted by the system is depicted in Fig 3.1.

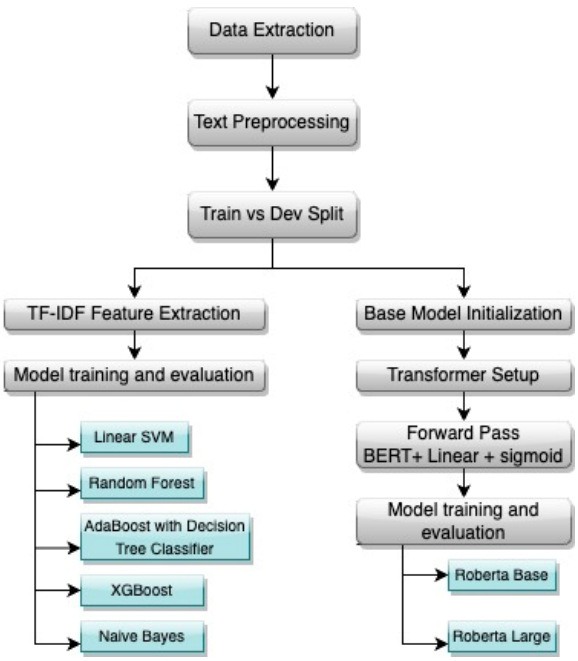

Fig 3.1 : System flow

All the models deployed approaches to achieve high accuracy while addressing the challenging aspects of this specific task. The first step in this task was to preprocess the text. The preprocessing steps included feature extraction, lowering the text case, removing the stop words, word tokenization etc.

For traditional models, we employed the TF-IDF (Term Frequency-Inverse Document Frequency) technique to transform raw sentences to numerical vectors. This helped in capturing the significance of words in a document which is essential for the success of traditional machine learning algorithms.

$$TF - IDF_{i,j} = TF_{i,j} * log\ (\frac{N}{DF_i})$$

First, we employed Linear SVC to establish an optimal hyperplane in the feature space to separate the distinct classes. Similarly, we trained Random Forest to handle large datasets, feature randomness, and built-in validation through Out of Bag (OOB) samples to get better classification results. Next, the AdaBoost algorithm operating in conjunction with decision trees was used to classify human versus machine generated text. The model iteratively emphasizes misclassified samples and combines the predictions of multiple weak learners. The final prediction is a weighted sum of individual weak learners, as depicted by,

$$F(x) = \sum_{t=1}^{T} \alpha_t\ f_t(x)$$

where T is the number of weak learners, α_t is the weight assigned to each, and $f_t(x)$ is the prediction. Further, XGBoost was deployed to the same dataset as it optimizes a differentiable loss function and incorporates regularization terms to prevent overfitting.

$$Objective = \sum_{i=1}^{n} Loss(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

Finally Naïve Bayes was the last traditional model used in this classification task. We used Multinominal NB along with incorporating two additional features i.e. text length and average word length.

To overcome the limitations of traditional models and to achieve higher prediction F1 score,

we further moved to transformer-based systems. We introduce transformer-based models (RoBERTa Base and RoBERTa Large) that leverage pre-trained contextualized embeddings. The transformer architecture is designed with a focus on contextual understanding, allowing it to capture intricate semantic relationships within the text. The RoBERTa Base model ensured the processing of input sequences and extracting [CLS] token before passing it through a linear layer with 768 input features and 1 output feature. For RoBERTa large, the linear layer had 1024 features. The final output was then transformed to [0, 1] using a sigmoid activation.

$$output_i = sigmoid\ (linear(ROBERTa(input_i)))$$

This equation represents the forward pass through our transformer-based models. The RoBERTa model extracts contextualized embeddings, which are then linearly transformed and passed through a sigmoid activation for binary classification. Lastly, the threshold of 0.5 i.e. p>=0.5 signifying as 1, was able to accurately differentiate between human-written and machine-generated text.

4 Experimental Setup

We studied SemEval 2024 Task 8 Subtask A using 7 different models namely Linear SVC, AdaBoost, XG Boost, Naive Bayes, Random Forest, RoBERTa Base, and RoBERTa Large. To train efficiently, we divided the train dataset into data into 90% for training and 10% for validation. We tweaked stop word removal and tokenization for each model. We have made use of tools like Scikit-learn, XGBoost, NLTK, and Hugging Face Transformers.

For the Linear SVC model, we addressed high training times by utilizing ThunderSVM's GPU-based implementation with a T4 GPU.

We trained RoBERTa Base on a V100 GPU with 20GB of memory, while RoBERTa Large utilized an A100 GPU with 40GB of memory. Specifics for RoBERTa Base included early stopping after 3 epochs, a batch size of 8, a learning rate of 1*10-4, and the SGD optimizer. Further tuning involved a batch size of 16, a learning rate of 1*10-5, and the Adam optimizer.

RoBERTa Large underwent early stopping after 3 epochs, with a batch size of 16, a learning rate of 1*10-4 and the SGD optimizer. Subsequent optimization included a batch size of 20, a learning rate of 1*10-5 and the Adam optimizer.

Model	Epoch	Batch size	Learning rate	Optimizer	F1 score
RoBERTa Base	3	8	1*e-4	SGD	0.793
	3	16	1*e-5	Adam	0.842
RoBERTa Large	3	16	1*e-4	SGD	0.836
	4	20	1*e-5	Adam	0.873

Table 4.1 :Hyperparameters for RoBERTa Models

The hyperparameters experimented with for both RoBERTa Base and RoBERTa Large are detailed in Table 4.1 above.

Crucially, GPU and RAM specs were vital for managing memory during training. We employed an early stopping mechanism to prevent overfitting in RoBERTa models. We evaluated models using precision, recall, F1-score, accuracy, and task-specific metrics for SemEval 2024 Task 8 Subtask A (bstrai/classification_report).

5 Results

This project mainly uses F1 scores as a metric to compare models. The class-wise F1 scores for each of the models applied in this project have been displayed in Fig 5.1 below.

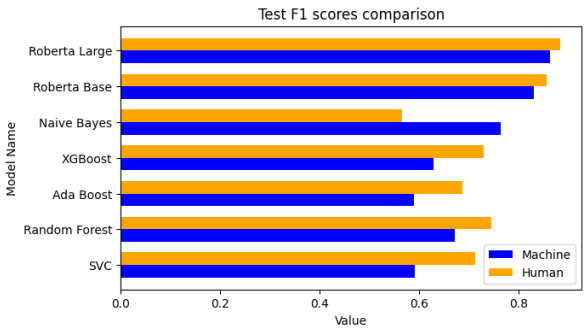

Figure 5.1 : Class-wise F1 scores for all models

We can see that for some models the F1 score for one class differs significantly from the other. For example, in the case of SVC, the F1 score for predicting human-written texts as positive class is roughly 0.7, whereas the F1 score for predicting machine-generated texts as positive class is only 0.6 approximately. Thus, it is imperative to check the class-wise F1 scores of a model.

The resulting average F1 scores for each of the applied models have been summarized below in Table 5.1.

Model Name	Average F1 Score
SVC	0.65
Random Forest	0.71
AdaBoost with Decision Tree Classifier	0.63
XGBoost	0.67
Naïve Bayes	0.66
RoBERTa Base	0.84
RoBERTa Large	0.87

Table 5.1 : Average F1 scores for all models

Table 5.1 showcases that amongst the conventional models. The ensemble technique of Random Forest achieves the best average F1 score of 0.71. The ensemble approach helps reduce overfitting and variance and mitigate the impact of individual noisy data points or outliers present in the training set. Unlike SVC, Random Forest does not require feature scaling which helps in dealing with datasets with features on different scales. However, the Random Forest approach is not as good as the large models as they achieve an F1 score higher than 0.84. RoBERTa Large is the best performing model amongst the whole lot with an F1 score of 0.87.

To understand the best performing models in depth and to check the performances of RoBERTa Base and Large on the unseen test dataset, the confusion matrices are displayed as follows:

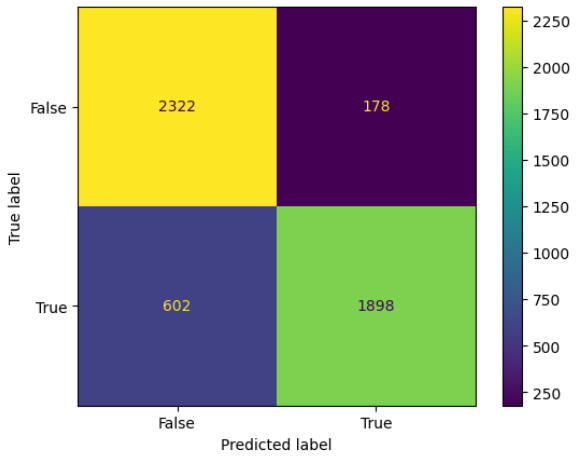

Figure 5.2 : RoBERTa Base Confusion Matrix

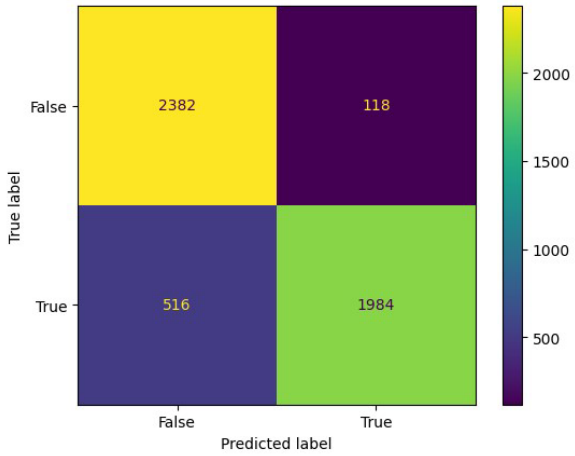

Figure 5.3 : RoBERTa Large Confusion Matrix

From the confusion matrices above, we can see that majority of the labels have been predicted correctly in these models. The RoBERTa Large model is the most capable model for classifying texts into human-written and machine-generated. The model achieves an impressive F1 score of 0.87 indicating the proportion of correctly classified instances out of the total. The macro and weighted averages reflect a balanced performance across both classes, with a slightly higher emphasis on precision.
These results suggest that the model is effective in discriminating between the two classes, particularly excelling in Class 0. The balance between precision and recall, along with the high F1 score, positions the model as a robust solution for the binary classification task. Fine-tuning may be considered to address specific performance aspects based on the task requirements and the desired trade-off between precision and recall.

6 Conclusion

The SemEval 2024 Task 8 Subtask A Monolingual track requires the implementation of large pre-trained models to achieve better classification of full-texts into human-written or machine-generated text. Large models such as RoBERTa Base and RoBERTa Large hold the potential to classify texts better than conventional models based on TF-IDF features due to prior training on various easily accessible textual sources over the web. The data sources for large models may comprise of human-written as well as machine-generated texts because of which the model may be able to find patterns in the style of texts and predict the class with better accuracy.

Acknowledgments

We would like to thank Prof. James Martin for educating and providing us with the opportunity to participate and contribute to the research of SemEval 2024 Task 8.

References

Liam Dugan , Daphne Ippolito , Arun Kirubarajan, Sherry Shi, Chris Callison-Burch 2023. *Real or Fake Text?: Investigating Human Ability to Detect Boundaries between Human-Written and Machine-Generated Text*. University of Pennsylvania.

Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, Arthur Szlam 2019. *Real or Fake? Learning to Discriminate Machine from Human Generated Text.* Facebook AI Research. Harvard University.