Data Annotaion and Analysis of Fine-tuned dehatebert-mono-english model on the Uncleaned and Cleaned data of the Hate speech

Sai Hemanth Kilaru,
skilaru@arizona.edu

$-> GitHubRepository <-$

Abstract

This research analyses and evaluates the performance of the hateBERT model, a transformer-based model fine-tuned for hate speech detection, using both uncleaned and cleaned versions of a dataset which is created and annoated by myself. Using a zero-shot methodology, we observe the model's ability to predict hate speech without the need for any domain-specific training or fine-tuning. The model's performance was evaluated using metrics such as accuracy and F1-score. The results show that preprocessing, including text cleaning, significantly improves the model's performance on the hate speech detection task than the raw data, demonstrating the importance of data quality for the machine learning tasks.

1 Introduction

Detecting the Hate/Offensive speech has become an essential task for moderating content on social media platforms such as X,instagram,etc.The rise of this kind of harmful speech online has led to the development of various machine learning models that aim to detect and flag these hate speech in real time cases like the tweets/comments etc.. While pre-trained models like HateBERT have shown very good prediction results in this area, their performance can vary significantly depending on the quality of the data they are being tested on. This study shows how the HateBERT model performs on both uncleaned and cleaned datasets. The main goal is to evaluate how much preprocessing (data cleaning) impacts model performance, especially in the case of zero-shot approach, where the model has not been fine-tuned on the specific task at hand. In this project, we use two datasets: one with raw, uncleaned data and the other cleaned to remove irrelevant content, such as links and special characters and evaluates the model performance on both of these using accuracy and F1 scores.

2 Challenges Faced

During the project,several challenges were faced that impacted both the performance and compatibility.One of the main difficulties was selecting proper tweets from the large Kaggle corpus. The dataset contained a wide variety of data,and from the various sources of internet and it was crucial to filter and balance the samples to ensure an equal distribution of HATE/OFFENSIVE and NEUTRAL labels, so i have carefully modified few rows and made it suitable for my instance. Another challenge came, particularly when extracting the scores for HATE and NON HATE labels of the model . This resulted in occasional indexing errors,data type errors,etc.I have solved that by careful error analysis and looked for the issue in the web sources like sackoverflow. Selecting the right pretrained language model was also a challenge, as I chose the normal BERT model earlier , which resulted in very poor accuracy scores and model didn't even properly analysed my given data and labels . Then , i looked for the pre-trained hate speech model on the web and found the hateBERT model,and i chose it as the model's pre defined labels matched my intended labels and the model gave decent accuracies through my zero-shot approach (but not very good accuracies).

3 Methodology

Here is the methodology for my entire project:

3.1 Data Annotation

The first step was the annotation of the dataset. I worked with a subset of 150 tweets selected from a larger kaggle corpus, ensuring a balanced distribution of hate speech and non-hate speech labels and modified a few rows on my own according to my own perspective of labels. The dataset was manually annotated into two labels based on my annotation guidelines:

- '1': HATE/OFFENSIVE:Tweets containing harmful or offensive language.
- '0': NEUTRAL:Tweets without any offensive language.

I, as Annotator-1, annotated the dataset, while Annotator-2 (my roommate, Sriram Theerdh Manikyala), also followed the same annotation guidelines,annotated the data. This annotation process was carried out to ensure that the dataset was labeled perfectly according to the guidelines. The labels were applied to each tweet by carefully reviewing the content for any form of offensive or hate speech.

3.2 Inter-Annotator Agreement Evaluation (Cohen's Kappa Score)

To assess the consistency between the annotations made by Annotator-1 and Annotator-2, I calculated the Cohen's Kappa score. This score measures the level of agreement between the two annotators. I have evaluated it matchemetically and yielded a score of 0.94, indicating almost perfect agreement between the two annotators. This confirms the reliability of the labeled dataset and gives a sign to proceed to further steps.

3.3 Data Cleaning

In our case, based on the clear observation of the data ,my data dosen't contain any null/weak columns.so,i proceeded with the text cleaning.This involued removal of unnecessary things like string(LINK),url's,hashtags,@mentions and the case-insensitive scenarios of the words,etc. They were done in order to get the cleaned dataset for my project and safely stored the clean rows in a new data file.

3.4 Model Setup and Zero-shot training

After searching for the hate speech models in the hugging face, i found named named 'Hate-speech-CNERG/dehatebert-mono-english' from the hugging face,in which the labels almost matches my data labels and after thorough study of the model documentation, i have finalized this model for the classification task.Here, without any fine tuning on the specific labels. The model was directly fed with my 150 rows considering its own pre-defined labels named 'HATE','NO HATE'(mostly similar to my labels). it generated the hate scores and non hate scores for each label based on that . if the hate score > non hate score, then it will be labelled as '1' else it will be labelled '0'; this matches my style of labelling and stored those predicted labels in a new dataframe and saved that in a dataset for both the uncleaned and the cleaned data.

3.5 Model Evaluation and comparision

After getting the model's labels, my labels and model's labels get compared and the evaluation metrics are generated for both the uncleaned and the cleaned data.

4 Evaluation Metrics

To evaluate the performance of the model on the datasets, two primary metrics were used:

- **Accuracy**: The proportion of correct predictions (i.e., how many times the predicted label matched the true label) is calculated.
- **F1 Score**: The F1 score balances both the precision and recall, was used to evaluate the model's ability to classify both the labels correctly. This metric is particularly useful in binary classification tasks, where class imbalance might disturb the results.

Both metrics were calculated for the uncleaned and cleaned datasets, and the model's preformance is compared on both the cases.

5 Results and Discussion

The performance of the hateBERT model was evaluated on both uncleaned and cleaned datasets. The scores are generated on the basis of my 150 samples.The table below shows the results based on Accuracy and F1 Scores:

Metric	**Uncleaned dataset**	**Cleaned dataset**
Accuracy	0.7066	0.7933
F1 Score	0.65625	0.7350

Table 1: Average Metric Scores for Fine-Tuned and Base Mistral-7B Models

The results demonstrate that the cleaned dataset outperforms the uncleaned dataset in both Accuracy and F1 Score.so, data pre-processing plays a key role in the zero shot model evaluation.

However, during the evaluation, I encountered indexing errors and unexpected output structures

when extracting scores for the labels. These errors typically occurred when the output from the model was not consistent To address this, I implemented error-handling mechanisms, such as the try-except block, to catch such issues and ensure that the program could continue running smoothly without crashing. This helped resolve inconsistencies and allowed me to proceed with the analysis. After overcoming such kind of issues,i successfully finished my project.

6 Conclusion

This study evaluated the performance of the hateBERT model for hate speech prediction using zero-shot approach on both uncleaned and cleaned datasets. The results showed a clear improvement in performance after cleaning the data. For the uncleaned dataset, the model achieved an accuracy of 0.7066 and an F1 score of 0.65625. After preprocessing the data, the model's performance increased, with an accuracy of 0.7933 and an F1 score of 0.7350.

These results highlight the significant impact of data cleaning on model performance, demonstrating that removing irrelevant elements, such as mentions and URLs, helps the model focus on the content that matters. While the zero-shot learning approach provided useful insights, further improvements could be made through fine-tuning the model or employing more advanced data preprocessing techniques. Overall, this study reinforces the importance of clean, well-annotated data in achieving reliable results in hate speech prediction tasks.

References

[1] English Hate Speech Superset; Superset combining all publicly available hate speech corpora in English*kaggle*. Retrieved from https://www.kaggle.com/datasets/elgringofrances/english-hate-speech-superset?select=en_hf_102024.csv

[2] Hate-speech-CNERG. (2023). *dehatebert-mono-english*. Retrieved from https://huggingface.co/Hate-speech-CNERG/dehatebert-mono-english

[3] Yash Jain. (2021). *Text Cleaning Using Regex in Python*. Retrieved from https://medium.com/@yashj302/text-cleaning-using-regex-python-f1dded1ac5bd

[4] Hugging Face. (2023). *BertForSequenceClassification*. Retrieved from https://huggingface.co/docs/transformers/main/en/model_doc/bert#transformers.BertForSequenceClassification