MIDTERM PROJECT

CSE584: Machine Learning- Tools & Algorithms

Name: Vedant Sawant
Email: vzs5407@psu.edu

Project Report

1. Dataset Curation

This section covers the details about the dataset used in the project, from collection to splitting and cleaning.

1.1 Dataset Collection

Truncated Texts (xi): As part of my data collection process, I used the 'Sentences' dataset available on Kaggle, provided by M. Fekadu. This dataset provided a diverse set of sentences to create the truncated sentences. The dataset can be found at this URL. This served as the basis for generating completions using various Large Language Models (LLMs). Another source was by utilizing various online random sentence generators which helped me to get comprehensive data. These tools helped me to get sentences which are grammatically and contextually diverse.
LLM Completion (xij): Used 5 different Large Language Models to complete each of the truncated text. Following 5 Large Language Models were used:
- bigscience/bloom-560m
- meta-llama/Llama-3.2-1B
- facebook/opt-1.3b
- EleutherAI/pythia-1b
- EleutherAI/gpt-j-6B

1.2 Dataset Annotation

Structured the whole data in tabular format where each row represents the completed sentence and the corresponding LLM label.

1.3 Data Organization

Structured the whole data in tabular format where each row represents the completed sentence and the corresponding LLM label.

1.4 Data Cleaning

Taking sentences from dataset and also from online random sentence generators created some duplicates xi. So removed all the duplicates from the datasets.

1.5 Data Splitting

Training Set: Use 70% of the data for training the classifier.
Validation Set: Used 30% of the data for evaluating the final performance.

1.6 Data Amount

Total number of xi generated is 2589. These 2589 xi is used for generating xij for 5 LLM which gives us a total of 12945 entries in the dataset.

---

2. Training Process

This section outlines the steps involved in designing the model and training it on the curated dataset.

2.1 Classifier Design

Utilized transfer learning by using pre-trained models like Bidirectional Encoder Representations from Transformers (BERT) . BERT is a model that excels in understanding and analyse the context with help of bidirectional training. Utilized "bert-base-uncased" model of transformer library. This is a smaller version of BERT with 12 layers. All text are converted to lowercase and improved generalization.

2.2 Training Process

This section shows the training process followed for the model.

2.2.1 Data Pre-processing

- Performed label encoding on the output labels of the dataset by converting the strings under labels into unique integers for each label.
- BertTokenizer class is part of Hugging Face Transformation library, which provide tools for working with BERT model. The tokenizer is important to convert the text into token IDs which model can understand and can be trained on it. The tokenized output is then added with a padding of maximum length being 32. Added special token which are needed by BERT tokenizer. Lastly converted the output in PyTorch tensors as I am implementing with PyTorch library.
- Added attention masks which helps in focussing on important tokens instead of focussing on the irrelevant padding part.
- Used TensorDataset to create a dataset from tensors and then used Dataloader class to create a iterable of the dataset. Facilized batching by breaking down the dataset into chunks for processing.

2.2.2 Optimization

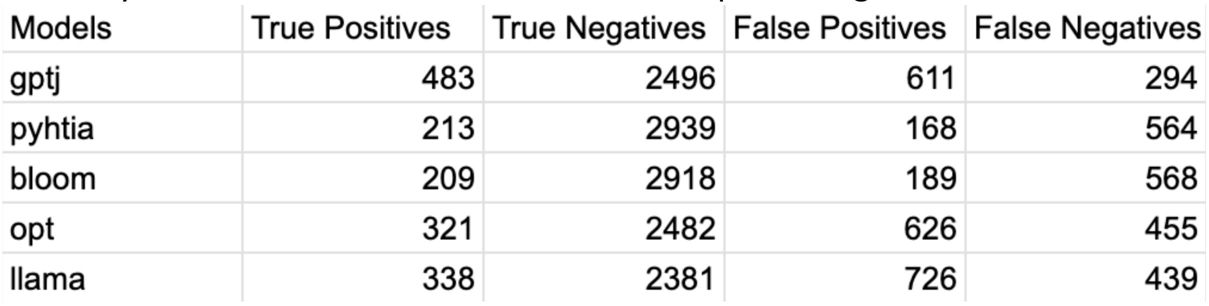
- Set the learning rate parameters to 2e-5 and batch size to 16 and number of epochs to 3. These parameters were the recommended for BERT training.
-Using Adam Optimizer with weight decay which helps in reducing overfitting by penalizing. We used Adam because of its Adaptive Learning Rate which helps in reaching the minima

faster compared to regular Gradient Descent.
-The loss function used for optimizing the algorithm is Cross-Entropy Loss function which is most commonly used for multiclass classification.

---

3. Results

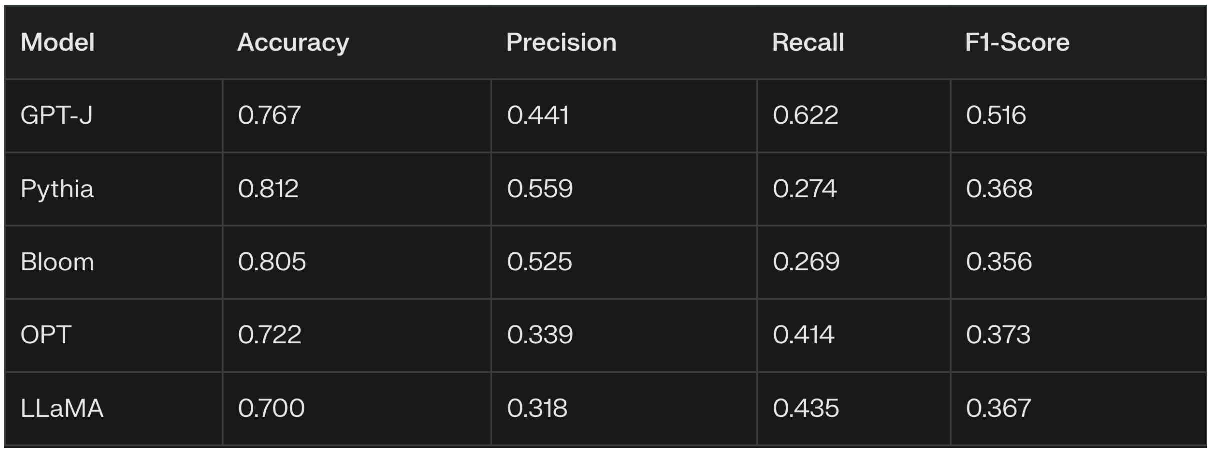
We split the data almost evenly for all classes of the labels. The overall accuracy of the BERT model came out to be 40 percentage.



Models	True Positives	True Negatives	False Positives	False Negatives
gptj	483	2496	611	294
pyhtia	213	2939	168	564
bloom	209	2918	189	568
opt	321	2482	626	455
llama	338	2381	726	439

Figure 1.1: Values of Confusion Matrix for each label

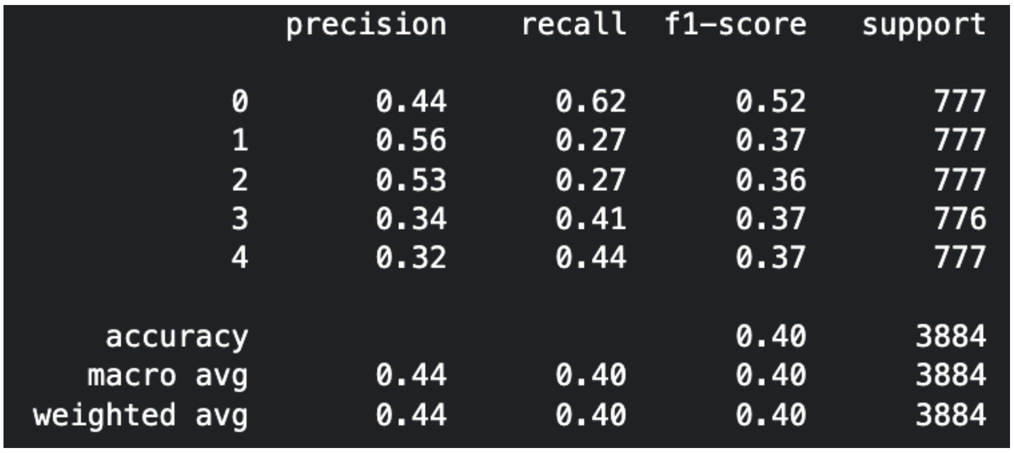
Calculated all the metrics needed to analyse the performance of the model. Based on table above we calculated accuracy, f1 score, precision, recall.



Model	Accuracy	Precision	Recall	F1-Score
GPT-J	0.767	0.441	0.622	0.516
Pythia	0.812	0.559	0.274	0.368
Bloom	0.805	0.525	0.269	0.356
OPT	0.722	0.339	0.414	0.373
LLaMA	0.700	0.318	0.435	0.367

Figure 2.2: Evaluation metrics for prediction of each label

Pythia has the highest accuracy indicating that deep learning classifier is able to correctly classify Pythia sentences.



```
              precision    recall  f1-score   support

           0       0.44      0.62      0.52       777
           1       0.56      0.27      0.37       777
           2       0.53      0.27      0.36       777
           3       0.34      0.41      0.37       776
           4       0.32      0.44      0.37       777

    accuracy                           0.40      3884
   macro avg       0.44      0.40      0.40      3884
weighted avg       0.44      0.40      0.40      3884
```

Figure 3.3: Classification Report for Predictions

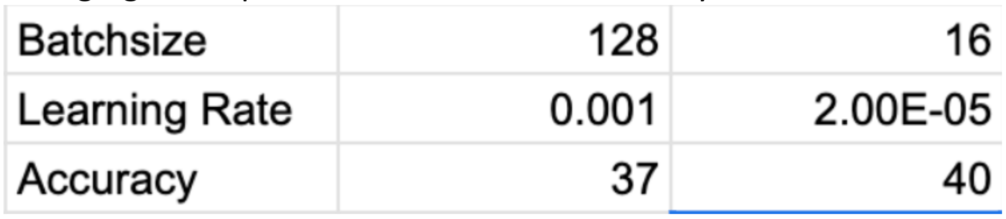
---

4. Experiments

This section contains the experiments conducted to improve the model's performance.

4.1 Parameters Tuning

I tweaked the parameters of learning rate and batch size to check how does changing these parameters affect the accuracy.



Batchsize	128	16
Learning Rate	0.001	2.00E-05
Accuracy	37	40

Figure 4.4: Effect of change of Batch size and Learning rate on accuracy of prediction

4.2 Cross Validation

Split the training data into 5 parts and used every section as validation as other for training. This method of training is called Cross Validation. This helps in fitting the model even if dataset is less. But Drawback of this method is it overfits the model. The training accuracy of the model came out to be 92% compared to validation accuracy which was very low which indicate that the model overfit the dataset and so not considering this method.

4.3 LSTM Deep Learning Classifier

Constructed a deep learning classifier from scratch. Implemented Single layer LSTM and Double Layer LSTM Recurrent Neural Network to perform the text classification. Despite achieving high training accuracy the validation was stagnating in 25%, which implicates the model has overfitted. To solve this implemented dropout to penalize large weight but these strategies yielded an improvement from 25 to 30%.

5. Future Work

Expanding the dataset would be the highest priority, as I think current dataset size may not enough for generalization. A larger dataset could provide more information and make it easy for the model to learn pattern and improve performance which was not able during smaller dataset. Base BERT model has been the foundation of the project but experimenting with bigger BERT variations may offer enhanced capabilities in analysing the text. Currently one text sentence is directly taken as input. There is potential research into feature engineering where multiple features can be introduces apart from the sentences. By achieving this there may be a potential improvement for the accuracy of the model.

6. Literature Survey

This section presents a review of relevant literature and research papers related to the project.

6.1 Przybyla, P., Durán-Silva, N., & Gómez, S.E. (2023). *I've Seen Things You Machines Wouldn't Believe*

Increasing capabilities of LLM like ChatGPT has made it difficult to detect AI written text. They introduced a model that combines grammatical sense, word frequency, linguistic networks and fine-tuned pre-trained LLM to get a neural network capable of classifying between machine generated texts.

6.2 Contrasting Linguistic Patterns in Human and LLM-Generated News Text – Alberto Muñoz-Ortiz, Carlos Gómez-Rodríguez, David Vilares

Conducts a quantitively analysis to compare the context and linguistic features written by human and different models of LLaMa family. The study morphological, syntactic, psychometric and sociolinguistic aspects of the speech. In their finding they exhibit difference in behaviour between human and LLM. Human display emotions through sentences like disgust and fear. Human generated text varies in sentences. Overall this research highlights of the parameters which can be used to differentiate output generated by different LLMs.

6.3 Smart Expert System: Large Language Models as Text Classifier Zhiqiang Wang, Yiran Pang, Yanbin Lin

This paper explores the application of LLM for text classification. It uses a LLM to reduce the requirements of preprocessing and domain expert knowledge which performs better than traditional machine learning and neural network model in tasks like sentiment analysis and spam detection. It uses data aggregation, zero-shot prompting, few shot learning. It introduces a novel metric U/E rate to assess cases where LLM produce irrelevant results. Practially it is seen that fine tuned LLM always perform better than traditional models but the resource demand is high. This paper concludes with a need for addressing fine tuning challenges and enhancing system accessibility.