Amazon Product Review Sentiment Analysis

Anonymous ACL submission

1 Introduction

The project fine tunes the BERT model in order to classify Amazon product reviews into positive, neutral, or negative sentiment categories based on the content of the reviews

2 Data

The dataset used for the project is "Amazon-Reviews-2023" of McAuley-Lab. The dataset consists of Amazon product reviews collected from the McAuley Lab dataset for the year 2023. The dataset is roughly consists of 700 000 samples. The dataset is consists of columns like: User, Item, ReviweText, Category, Rating, Rtoken, Mtoken. I selected out only "ReviewText" and "Rating". The dataset rating column consists of 5 classes each signifying the grades of: 1, 2, 3, 4, 5. I selected 10000 samples of each class. Finally I classified the grades of 1 and 2 as "negative", 3 as a "neutral", 4 and 5 as a "positive". They will be indexed as 1, 2, 3 correspondingly.

3 Methods

As for the methods I fine-tuned a Distilled BERT model for sentiment analysis on Amazon reviews. First of all I initialized the "bert-base-uncased" model. Afterwards I splitted the prepared dataset into three sets as for training, validation, and testing. Training set contains 21000 samples validation set 3000 and testing set contains 6000 samples. And both of the training batch size and testing batch sizes are 32. The logic for fine tuning is referenced from GitHub repository "bert-product-rating-predictor" by "csbanon".

4 Experiments

Class weights for the weighted cross-entropy loss are computed to handle class imbalance in the dataset. The frequency distribution of labels is analyzed to determine appropriate weights, ensuring fair representation of all classes during training. The results are follwing:

```
Star distribution   : [0.3333333432674408, 0.3333333432674408, 0.3333333432674408]
Weights             : [0.3333333432674408, 0.3333333432674408, 0.3333333432674408]
```

Then the training loop iterates over epochs, batches, and data samples to train the model. During each epoch, training loss, accuracy, and other metrics are monitored and logged. So the the result of the traing on each epoch and final result is as follows:

```
Training Loss: 1.1147043704986572   Training Accuracy: 34.375
Saving checkpoint at /content/drive/MyDrive/Colab Notebooks/checkpoint.dat
Batch 100 of epoch 1 complete.
Training Loss: 0.8339582268554385   Training Accuracy: 60.51980198019802
Saving checkpoint at /content/drive/MyDrive/Colab Notebooks/checkpoint.dat
Batch 200 of epoch 1 complete.
Training Loss: 0.7307419619750027   Training Accuracy: 66.83768656716418
Saving checkpoint at /content/drive/MyDrive/Colab Notebooks/checkpoint.dat
Batch 300 of epoch 1 complete.
Training Loss: 0.7000133623910505   Training Accuracy: 68.75
Saving checkpoint at /content/drive/MyDrive/Colab Notebooks/checkpoint.dat
Batch 400 of epoch 1 complete.
Training Loss: 0.6706147437678311   Training Accuracy: 70.19170822942644
Saving checkpoint at /content/drive/MyDrive/Colab Notebooks/checkpoint.dat
Batch 500 of epoch 1 complete.
Training Loss: 0.6517189834527151   Training Accuracy: 71.19510978043913
Saving checkpoint at /content/drive/MyDrive/Colab Notebooks/checkpoint.dat
Batch 600 of epoch 1 complete.
Training Loss: 0.6376672569606546   Training Accuracy: 71.88019966722129
```

So the result of the accuracy testing on the testing set shows the validation accuracy of 75.83.

```
Running validation on model trained on 2 epochs
Validation Loss: 0.5567184556671914
Validation Accuracy: 75.83333333333333
```

5 Results analysis

The training accuracy steadily increases over the course of Epoch 1, starting at 34.375% for the first batch and reaching 71.88% by the end of training. So the overall training accuracy for Epoch 1 is 72.19%. Regarding the training loss it decreases after each iteration reaching 0.6317. Regarding the validation accuracy it reaches 75.74% after epoch 1 and 75.83% after training for 2 epochs. The training process demonstrates a consistent improvement in training accuracy over the course of Epoch 1, indicating that the model effectively learns from the training data as it progresses. This upward trend suggests that the model is successfully capturing the underlying patterns in the text reviews. Moreover, the validation accuracy remains consistently

high, suggesting that the model generalizes well to unseen data and does not exhibit signs of overfitting to the training set.

Conclusion

In conclusion, The experiments with a fine-tuned Distilled BERT model showed promising results. The training accuracy improved steadily, reaching 72.19% by the end of the first training round, and the training loss consistently decreased. Validation accuracy also got better, reaching 75.83% after two rounds of training. These results indicate that following model learned well from the training data and could understand the patterns in the reviews. The high validation accuracy suggests that model can work well with new data without getting too specific to the training set. Overall, following approach seems promising for accurately understanding the sentiment in Amazon product reviews, which could help businesses understand customer feedback better.

References

Dataset: McAuleyLab/Amazon-Reviews
Model: BERT
Codes used: csbanon/bert-product-rating-predictor