Title: Implementing Text Classification using Transformers and LSTM

Hesam Farjad

Abstract

This report outlines the implementation of a text classification model using Transformers and LSTM. The process involves installing the necessary libraries, preparing the environment, preprocessing the data, training and evaluating the model, and analyzing the results. The goal is to accurately classify input texts into different categories, enabling applications such as sentiment analysis, spam detection, and document categorization.

1 Introduction

Text classification is a fundamental task in natural language processing that involves assigning predefined categories or labels to text inputs. It has extensive applications in various domains, including social media analysis, customer feedback analysis, and content filtering. In this report, we present the steps taken to implement a text classification model using a combination of Transformers and LSTM.

2 Installing Libraries and Preparing the Environment

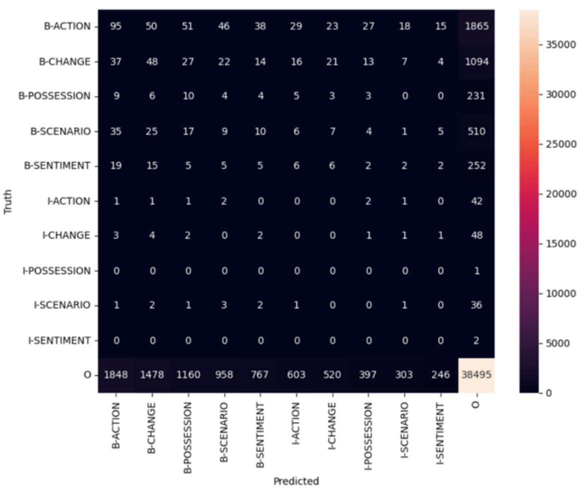
The initial step in implementing the text classification model is to install the required libraries. In this project, we utilize the Transformers library, which provides a powerful set of tools for working with transformer-based models. Additionally, we import other necessary libraries to support our implementation. To establish the connection, we mount the drive and ensure the environment is set up correctly.

3 Data Preprocessing

To prepare the data for training and testing, we need to perform preprocessing steps. We start by reading the input data from a JSON file and converting it into a Pandas data frame. This enables easier manipulation and analysis of the data. We then focus on preparing the labels and tokens for modeling. The labels are converted into a dictionary, assigning a unique numerical value to each unique label. The tokens, representing the text inputs, are grouped into lists based on a specified window size. Padding is applied to ensure uniform length across the lists.

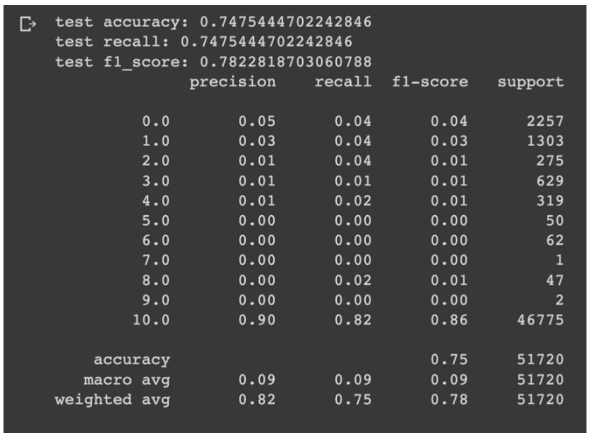
4 Model Training and Evaluation

This section details the process of training and evaluating the text classification model. We use the NumPy library to manipulate and transform the data. The training and testing datasets are converted into data loaders, which facilitate efficient batch processing during model training. We define a custom model class, incorporating an embedding layer, LSTM layer, dropout layer, and SoftMax layer. The model is trained using cross-entropy loss as the objective function and optimized using the Adam optimizer. The number of epochs is determined dynamically, allowing for flexibility in the training process.



5 Model Testing and Analysis

In this section, we discuss the testing phase of the trained model. Test data is preprocessed in a similar manner as the training data, and predictions are made using the trained model. To evaluate the model's performance, we calculate metrics such as accuracy and F1 score. Additionally, a confusion matrix is generated to provide a visual representation of the model's predictions and their alignment with the true labels. The analysis of these results allows us to assess the model's effectiveness in accurately classifying text inputs.



```
test accuracy: 0.7475444702242846
test recall: 0.7475444702242846
test f1_score: 0.7822818703060788
              precision    recall  f1-score   support

         0.0       0.05      0.04      0.04      2257
         1.0       0.03      0.04      0.03      1303
         2.0       0.01      0.04      0.01       275
         3.0       0.01      0.01      0.01       629
         4.0       0.01      0.02      0.01       319
         5.0       0.00      0.00      0.00        50
         6.0       0.00      0.00      0.00        62
         7.0       0.00      0.00      0.00         1
         8.0       0.00      0.02      0.01        47
         9.0       0.00      0.00      0.00         2
        10.0       0.90      0.82      0.86     46775

    accuracy                           0.75     51720
   macro avg       0.09      0.09      0.09     51720
weighted avg       0.82      0.75      0.78     51720
```

6 Conclusion

This report has presented an overview of the implementation of a text classification model using Transformers and LSTM. The combination of these powerful techniques enables effective processing and classification of text data. By following the outlined steps, we were able to install the necessary libraries, pre-process the data, train and evaluate the model, and analyse its performance. The results demonstrate the model's ability to classify text inputs accurately, opening avenues for various applications such as sentiment analysis and spam detection. Further improvements and optimizations can be explored to enhance the model's performance in specific text classification tasks.