CSE440 Project
Fall 2023

For: Dr. Farig Yousuf Sadeque
Date of Submission: 17 December, 2023

By:

Name	ID
Niaz Nafi Rahman	21301700
Shreya Adrita Banik	21301669
Zaid Rehman	21301389
Tahsina Moiukh	21301612

1. Introduction:

This CSE440 project focuses on creating a neural network classifier for IMDB reviews, aiming to evaluate the effectiveness of different models. We utilized essential tools like Pandas for data handling and Keras, powered by TensorFlow, for neural network implementation. Our primary objective was to build a shallow recurrent neural network and explore its performance enhancements through LSTM layers. This report provides insights into our approach, detailing the tools and libraries used to achieve our goals.

2. Data Loading:

The dataset was loaded using the Pandas library in Python. The primary columns include the review and sentiment columns. It is noteworthy that this code is designed for execution in **Google Colab**. To facilitate seamless data access, we kindly request that users run the code in Google Colab and ensure prior uploading of two essential files to their respective Google Drive accounts. The first file, titled 'IMDB Dataset.csv,' should contain the dataset with primary columns 'review' and 'sentiment.' Additionally, users are required to upload the GloVe word embeddings file, 'glove.6B.100d.txt,' which is crucial for our pre-processing steps. This approach ensures user interactivity and data accessibility, aligning with the collaborative and dynamic nature of Google Colab.

3. Preprocessing and Word Embeddings:

For effective preprocessing we did the following:

- **Tokenization:** We utilized the Keras Tokenizer class to tokenize the reviews
- **Padding Sequence**: To ensure uniform input dimensions for our model, we applied padding to the tokenized sequences.
- **Sentiment Label Conversion**: sentiment labels were converted to binary format (positive as 1, negative as 0).
- **Embedding Matrix:** We constructed an embedding matrix based on the tokenized words and their corresponding GloVe vectors.
- **Word Embeddings:** We utilized a pre-existing file containing GloVe word embeddings in 100-dimensional space. These embeddings were loaded and applied to our dataset, enabling us to convert individual words within reviews into their corresponding 100-dimensional vectors.

4. Data Splitting:

We divided our dataset into training and testing sets using an 80-20 split that is 80% of the data was used for training and 20% of the data for testing.

5. Shallow Neural Network Model:

The shallow model consists of an embedding layer with 100-dimensional GloVe vectors, followed by two dense layers. The first dense layer has 10 units with a ReLU activation function, and the final layer outputs a sigmoid activation for binary classification.

- **Batch Size**: We tried batch sizes of 16,32,64,128,256 and among them as accuracy of the batch size of 32 was highest which we used for training for 20 epochs.
- **Training Performance metrics:**
  Accuracy : 0.5715804696083069
  F1score_m: 0.71368408203125
- **Testing Performance metrics:**
  Accuracy: 0.5577580332756042
  F1score_m: 0.6987027525901794

6. Gated Recurrence Relations Models:

The shallow model was enhanced by replacing the first dense layer with two versions of LSTMs:

a) Single Unidirectional LSTM:

The first dense layer was replaced with a single unidirectional LSTM layer.

- **Training Performance metrics:**
  Accuracy: 0.8796749711036682
  F1score_m: 0.8724069595336914
- **Testing Performance metrics:**
  Accuracy: 0.8410999774932861
  F1score_m: 0.8303547501564026

b) Single Bidirectional LSTM:

We employed a single bidirectional LSTM layer, capturing information from both forward and backward directions.

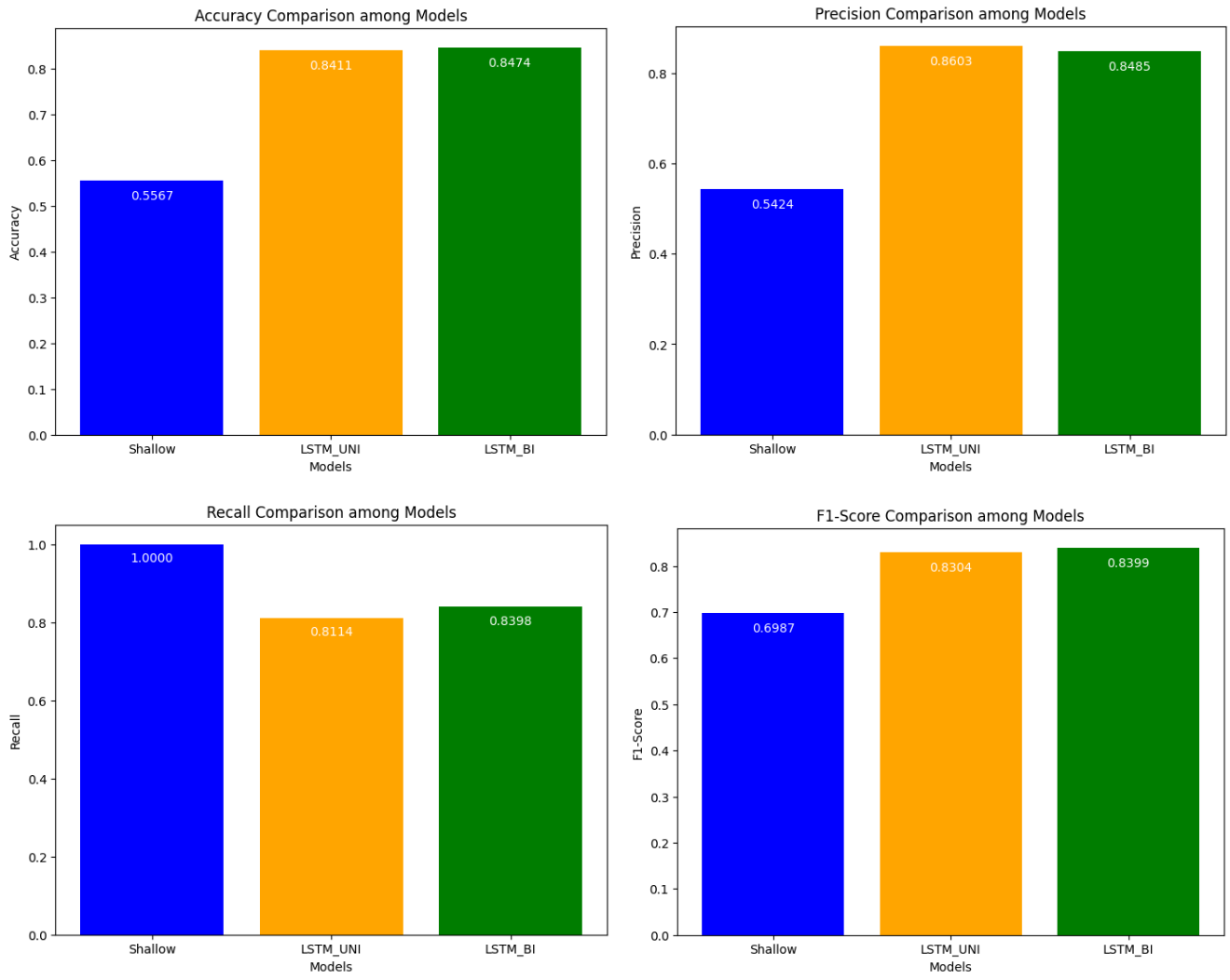
- **Training Performance metrics:**

Accuracy: 0.909850001335144
F1score_m: 0.9068063497543335

- **Testing Performance metrics:**

Accuracy: 0.8474000096321106
F1score_m: 0.8398622870445251

7. Model Performance Analysis:



Both the unidirectional and bidirectional models had an overall better performance than the shallow model in terms of accuracy and F1 score. However, the bidirectional LSTM outperformed the shallow model and the single unidirectional LSTM model in terms of F1 score. This is because the bidirectional LSTM leverages information from both directions, enhancing its ability to capture long-term dependencies.

8. Improvements:

- Using GRU: It is similar to LSTM but with a simpler structure, requiring less computations making it potentially quicker to train.
- Using pre-trained models: Fine-tuning models pre-trained on large datasets like BERT can be used as an improvement.

9. Conclusion:

In conclusion, this project revealed that a simple neural network didn't perform well for sentiment analysis. But when LSTM models were introduced—both the unidirectional and bidirectional ones—the performance improved significantly. This highlighted how important LSTM layers are for capturing detailed patterns in sentiment analysis.

However, even after tweaking various settings like batch size during hyperparameter tuning, the overall performance of the model didn't change noticeably. This shows that finding the right settings for these models is tricky. In summary, this work encapsulates the significance brought forth by the introduction of LSTM models within the realm of sentiment analysis. It delineates the pivotal role played by these architectures in enhancing the analysis of sentiments, showcasing their ability to capture nuanced patterns within textual data.

10. References:

Dataset:
https://drive.google.com/file/d/1hyn6U44cvUNedMnzPHzHsiBgXMHFBjBw/view
Glove.6B.100d:
https://drive.google.com/file/d/1CLrwskD-M8ce0M5agxRjB3OhKuKiCcCn/view
Tutorials:
https://realpython.com/python-keras-text-classification/
https://medium.com/mlearning-ai/the-classification-of-text-messages-using-lstm-bi-lstm-and-gru-f79b207f90ad