CSE440: Natural Processing II

PROJECT REPORT

Team ID - 03

GROUP MEMBERS:

Name	ID	Section
Umme Abira Azmary	20101539	01
MD Ikramul Kayes	21301576	01
Mollah Md. Saif	20101416	02
Nourin Siddique Ananna	20301012	02

Introduction

Sentiment analysis plays a crucial role in understanding public opinion and user feedback, with applications ranging from product reviews to social media sentiment tracking. In this project, we aim to develop a sentiment classification model for IMDB reviews. We start developing with a shallow recurrent neural network (RNN) which consists of an embedding layer, a dense layer, and an output layer. After that, we improve the model by introducing gated recurrence relation. The two versions of this implementation are represented respectively by a single unidirectional LSTM layer which consists of an embedding layer, a LSTM layer, an output layer and a single bidirectional LSTM layer consists of an embedding layer, a BLSTM layer, and an output layer. The dataset is preprocessed, converted into vectors using GloVe word embeddings, and split into training and testing sets. In the subsequent sections, we showcase these model's performance, employing metrics such as **accuracy**, **precision**, **recall**, and **F1** score. By accomplishing these steps, our goal is to build an effective sentiment classification model and gain insights into the impact of incorporating LSTM layers on model performance.

Data Loading and Preprocessing

The IMDB Reviews dataset is in .csv format. It consists of a collection of reviews from IMDB, each labeled with sentiments indicating whether the review is positive or negative. The dataset is initially splitted into **80-20** ratios for training and testing. We splitted 20 percent of the training dataset for validation. As a result, the training data teaches the model to recognize patterns and relationships within the data and the validation data plays a vital role in fine-tuning the model when it becomes too specialized in the training data.

Next, we applied the **Keras Tokenizer** to tokenize and convert text reviews into sequences of integers. It establishes a mapping between words and their corresponding indices. The tokenizer is fitted on the training set and is applied to both the training and test sets.

Moreover, we have applied padding to both the training and test sets to ensure uniform sequence lengths. According to the instructions, the max sequence length for padding is set to 256. However, we have plotted a distribution graph of sentence lengths which shows the average sequence length as **234.88.** So, we have set the max sequence length(maxlen) for our padding to **235** to obtain a better performance. By doing this, all the reviews have the same length.

Furthermore, we have used a LabelEncoder to convert categorical class labels into numerical representation to ensure compatibility with machine learning models. The encoder assigns unique integers to each class for subsequent model training and evaluation.

Word Embedding

For word representations, we have applied pre-trained GLoVe word embedding. The GLoVe vectors are downloaded from the specified source. The GLoVe vectors offer a rich semantic understanding of words by mapping them to a 100-dimensional embedding matrix, where each row represents the vector for a specific word. Words that are not present in the GLoVe are initialized with zeros. This matrix serves as the weight matrix for the embedding layer in our model.

Training Models

Shallow Model: We have designed a sequential shallow(single hidden layer) model with three layers. The first layer is an embedding layer with output shape 100 which transforms words into dense vectors based on the GloVe word embeddings. Subsequently, a flatten layer is applied to

convert the embedded matrix into a one-dimensional array. Following that, we have applied the first Dense layer, consisting of **10 neurons** with a **ReLU** activation function, which extracts complex features from the flattened input. The final Dense layer, consisting of a **sigmoid activation** function, produces a binary classification output.

LSTM Model: To enhance the model's performance, we have introduced a gated recurrence relation. This version of the model includes an embedding layer, the same as before. Following that, we have applied a single-layer LSTM with **10 units** and a **ReLU** activation function, followed by a dense layer with a sigmoid activation for binary classification.The use of LSTM aims to enhance the model's ability to capture sequential patterns in text data, contributing to improved sentiment analysis performance.

Bidirectional LSTM Model: Another variant of the gated recurrence relation is the implementation of Bidirectional LSTM. The model architecture consists of an embedding layer, the same as before. Following that, a Bidirectional LSTM layer with 10 units and a ReLU activation function, followed by a dense layer with a sigmoid activation for binary classification. The Bidirectional LSTM layer captures contextual information from both forward and backward sequences, enhancing its ability to recognize detailed patterns in text data. Consequently, it aims to improve sentiment analysis performance.

For all of these above models, we have applied **Adam optimizer** to minimize the model's binary cross-entropy loss during training. As per instruction, the epoch size is configured as 20. We have experimented with various batch sizes, such as: 10, 32, 40 and 64. Among these, the batch size 40 shows the most preferable result and so we set the batch size as 40.

Performance Analysis

We have evaluated the model by assessing the training and testing accuracies, visualizing confusion matrices, and calculating F1 scores. The reason for calculating both the accuracy and F1 score is that accuracy measures the proportion of correctly classified instances, but it may be misleading when classes are imbalanced. In such cases, F1 shows a more balanced assessment as it considers both precision and recall, providing a clearer picture of the model's true effectiveness in sentiment classification.

Shallow model

Batch Size	Training Accuracy	Testing Accuracy	Testing Precision	Testing Recall	Testing F1 Score
10	0.5012	0.4964	1.00	0.006	0.0012
32	0.9616	0.8592	0.8942	0.8172	0.8540
40	**0.9721**	**0.8582**	**0.8567**	**0.8629**	**0.8598**
64	0.9734	0.8632	0.8603	0.8698	0.8650

LSTM model

Batch Size	Training Accuracy	Testing Accuracy	Testing Precision	Testing Recall	Testing F1 Score
10	0.9664	0.8349	0.8413	0.8287	0.8349
32	0.6317	0.5437	0.6113	0.2594	0.3642
40	**0.9733**	**0.8682**	**0.8731**	**0.8641**	**0.8685**
64	0.9565	0.8104	0.8206	0.7982	0.8093

BLSTM model

Batch Size	Training Accuracy	Testing Accuracy	Testing Precision	Testing Recall	Testing F1 Score
10	0.9543	0.7876	0.7636	0.7621	0.7967
32	0.9501	0.8376	0.8446	0.8306	0.8375
40	**0.9678**	**0.8407**	**0.8587**	**0.8186**	**0.8382**
64	0.6629	0.6158	0.6181	0.6216	0.6198

Model Comparison

Model Name	Training Accuracy	Testing Accuracy	Testing Precision	Testing Recall	Testing F1 Score
Shallow	0.9721	0.8582	0.8567	0.8629	0.8598
LSTM	**0.9733**	**0.8682**	**0.8731**	**0.8641**	**0.8685**
BLSTM	0.9678	0.8407	0.8587	0.8186	0.8382

Discussion

The LSTM model outperforms both the shallow and BLSTM models in terms of testing accuracy, precision, recall and F1 score. For shallow models, it lacks the ability to capture sequential dependencies and long-term context in the data. For the BLSTM model, even though it demonstrates better performance in capturing contextual information by considering both forward and backward sequences, the choice of a maximum sequence length for padding introduces a potential challenge. Because the original max sequence length of the dataset is 2943

and using it creates a sparse matrix during padding. To address this, we have set the max sequence length to 235, which represents the median value of the sentence sequences. However, this adjustment may impact the backward observation of the BLSTM model, as long sentences are not intact after padding. On the other hand, the LSTM model's ability to capture sequential patterns in the data appears to be more beneficial for this particular task. As LSTM has the ability to capture long-term context in the data. However, the f1 score for LSTM and BLSTM is closer which identifies that BLSTM may be unable to detect the sentiment of a few of the sentences which have a longer length.

Improvements:

A couple of improvements to enhance model performance:

1) **Early Stopping and increasing Epochs:** From the graph, we observe that both BLSTM and LSTM models are overfitting. In this situation, if we integrate early stopping, we can stop the training when the improvement is not increasing and save the best weights. Also, increasing the number of epochs may boost the performance of the model.
2) **Integrate GRU Instead of BLSTM:** GRU is often faster than BLSTM as it has one less gate and no cell state to update. So, it may show a better performance.
3) **Dropout for Overfitting:** Implementing 10%-25% dropout can help address overfitting issues which will help the model to train well.
4) **Increase Padding Size:** As BLSTM isn't showing better results, calculating the max length based on other parameters for padding could improve its performance.

Conclusion:

In conclusion, our sentiment classification project for IMDB reviews engages diverse recurrent neural network architectures—shallow, LSTM, and bidirectional LSTM. From these, evaluation metrics like accuracy, precision, recall, and F1 score favored the LSTM model over the shallow and bidirectional LSTM counterparts. Despite bidirectional LSTM's enhanced contextual awareness, its performance may be affected by padding choices and adjusting max sequence length may improve this. The LSTM model's proficiency in capturing sequential patterns and long-term context outshone the others. Focusing on early stopping, increasing epochs, GRU implementation, dropout for regularization, and optimizing padding sizes may improve the overall performance of the model.