Deep Learning Lab 7 Report

Shusrith S

PES1UG22AM155

March 9, 2025

1 Sentiment Analysis and dataset

Sentiment analysis is the task of classifying text as positive or negative, commonly used to analyze reviews and opinions. Machine learning models, from traditional methods like TF-IDF to deep learning techniques like seq2seq models, help identify sentiment patterns and improve accuracy.

The IMDb movie review dataset contains 50,000 labeled reviews, evenly split for training and testing. It is widely used to evaluate sentiment analysis models on longer texts, making it a key benchmark for NLP research and deep learning advancements.

2 Preprocessing

2.1 Cleaning

The dataset consists of scraped internet reviews, containing HTML tags, erratic punctuation and other human aspects of writing on the internet. For the purpose of tokenization and training, all this needed to be cleaned. The regex library was used to remove all HTML tags, punctuation and extra whitespaces, leaving behind only full english words. They were also converted to lowercase and tokenized on a word level.

2.2 Tokenization

The data is to be trained on sequence based models like RNNs and LSTMs and these models try to understand patterns in the tokens to classify the input. For this purpose, these sequence based models work well with simple, low level tokenizers which can encode some functional aspect of the words. For this purpose, a frequency based tokenizer was used. The entire corpus of text data was collected and the frequency of occurrence of the words was calculated. This was then sorted and to maintain a vocabulary size of 20000, only the top 20000 tokens were retained and a vocabulary was created, numbering the tokens from 1 to 20000.

The token 0 was used for unknown characters not present in vocabulary. The token 20001 was used for padding. Using this, a mapping was created and each input was converted to a list of tokens which could then be used for training the models.

2.3 Padding and structuring

Sequence based models can take only fixed sized inputs, so a max length of 200 tokens was set for each input. For each input, chunks of 200 tokens were taken and added to the dataset as new inputs. The last chunk was padded with pad tokens to make the length 200. An extra 30000 inputs were created this way.

3 Models

3.1 RNN

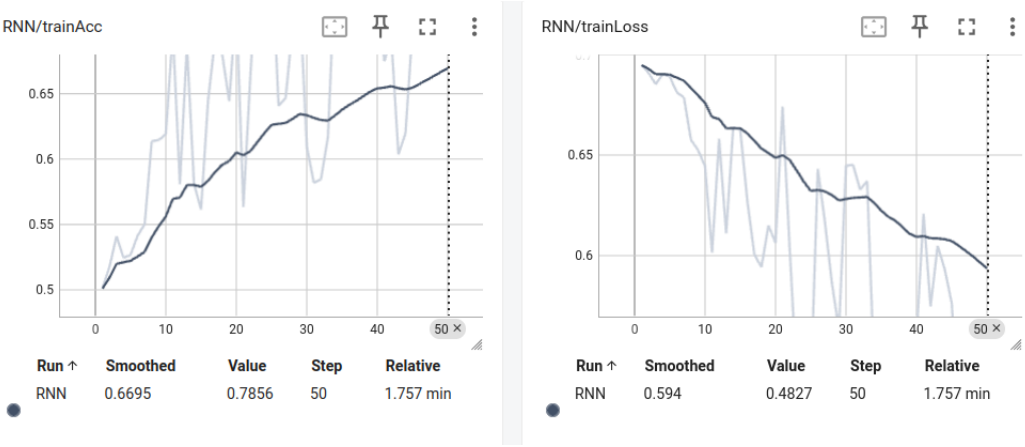

Figure 1: RNN train curves

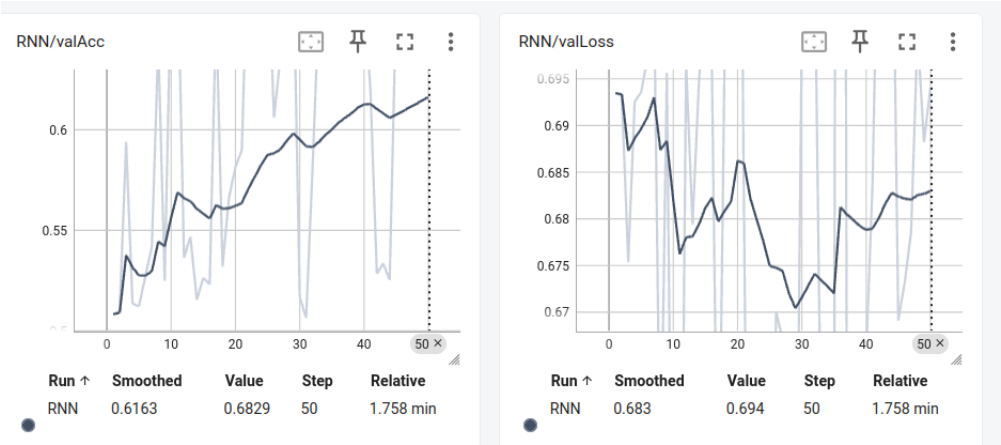

Figure 2: RNN val curves

Metric	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Value	0.4827	0.7856	0.694	0.5300

Table 1: Training and Validation Metrics for RNN Model

Metric	Accuracy	Precision	Recall	F1-score
Value	0.5300	0.5322	0.5568	0.5442

Table 2: Final Validation Metrics

	Predicted 0	Predicted 1
Actual 0	7367	7286
Actual 1	6598	8288

Table 3: Confusion Matrix

A basic RNN model was used to train initially. 128 RNN units were and was trained for 50 epochs. The final accuracy was 78% for train and 53% for validation. The loss curves show that the model starts to overfit and the curve has very high variability, especially in val loss. This shows that the model does not perform too well and needs either more units or more layers to show better performance and some normalization technique to reduce the high variance.

3.2 LSTM

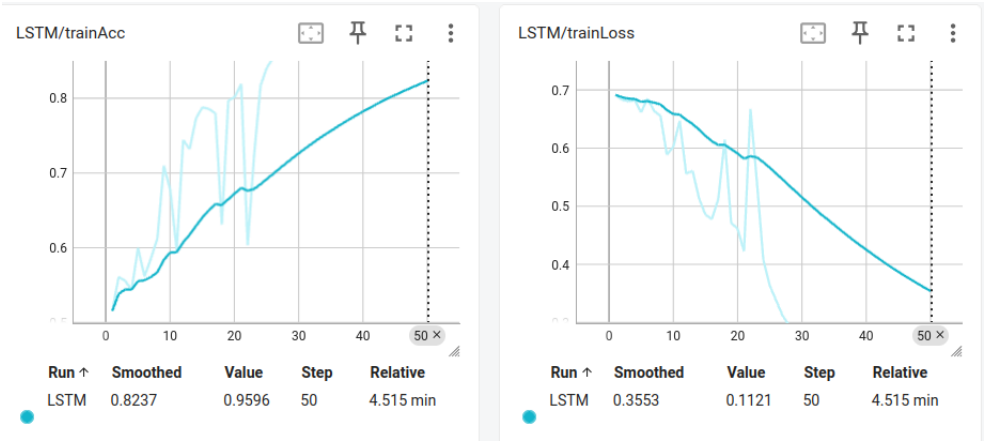

Figure 3: LSTM train curves

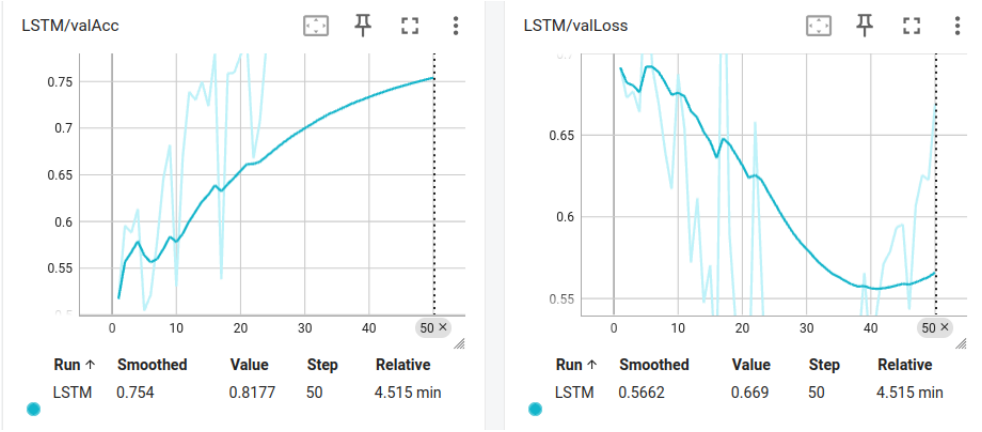

Figure 4: LSTM val curves

Metric	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Value	0.1121	0.9596	0.669	0.8177

Table 4: Training and Validation Metrics for LSTM Model

Metric	Accuracy	Precision	Recall	F1-score
Value	0.8177	0.8133	0.8204	0.8168

Table 5: Final Validation Metrics

	Predicted 0	Predicted 1
Actual 0	12338	2315
Actual 1	3277	11609

Table 6: Confusion Matrix

A simple unidirectional LSTM model was trained for 50 epochs with 128 units. The LSTM model shows better training performance with a significantly better training accuracy and loss. But the validation loss remains high, indicating a high amount of overfitting. The curves show a large number of spikes and drops, indicating large variance and pointing to the need for normalization. The model requires strong regularization techniques because of the large number of gates present used to prevent exploding and vanishing gradients. The high training accuracy however, shows that this can perform a lot better than the RNN if regularized properly. This model had the higest recall.

3.3 GRU

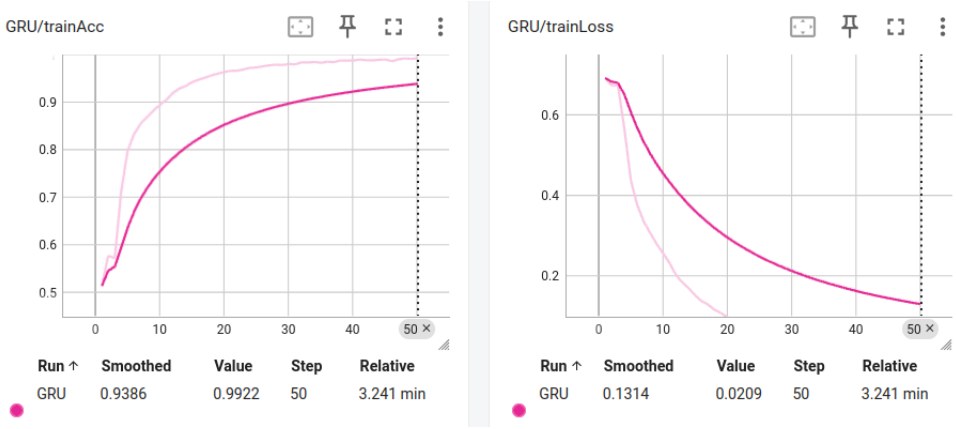

Figure 5: GRU train curves

Figure 6: GRU val curves

Metric	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Value	0.0209	0.9922	1.2133	0.8198

Table 7: Training and Validation Metrics for GRU Model

Metric	Accuracy	Precision	Recall	F1-score
Value	0.8198	0.8260	0.8099	0.8179

Table 8: Final Validation Metrics

	Predicted 0	Predicted 1
Actual 0	12097	2556
Actual 1	3130	11756

Table 9: Confusion Matrix

A GRU model was trained for 50 epochs with 128 units. The model showed very less fluctuation and very smooth curves, but a very high amount of overfitting. The validation loss goes upwards continuously and validation accuracy flat lines after 20 epochs. The overall metrics are very similar to that of the LSTM but there is a stronger amount of overfitting evident from the graphs. This model had the highest F1 score.

3.4 Bidirectional LSTM

Figure 7: BiLSTM train curves

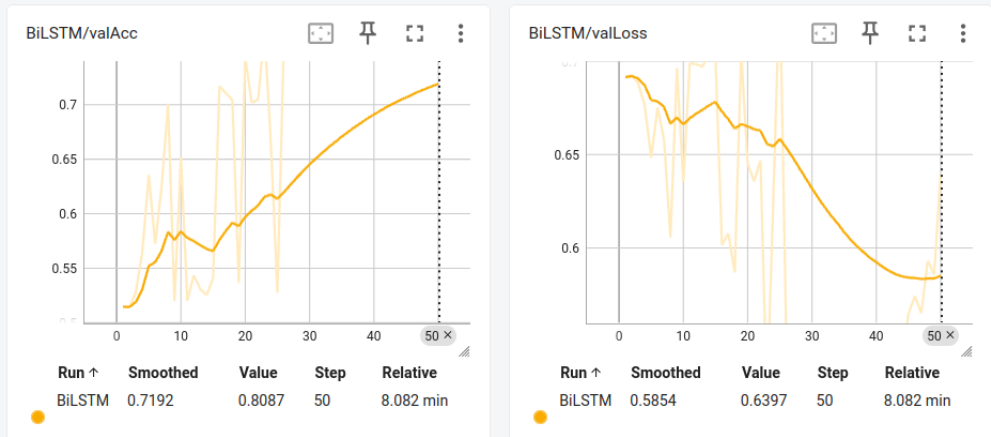

Figure 8: BiLSTM val curves

Metric	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Value	0.1521	0.9434	0.6397	0.8087

Table 10: Training and Validation Metrics for BiLSTM Model

Metric	Accuracy	Precision	Recall	F1-score
Value	0.8087	0.8159	0.7923	0.8039

Table 11: Final Validation Metrics

	Predicted 0	Predicted 1
Actual 0	11641	3012
Actual 1	2792	12094

Table 12: Confusion Matrix

A bidirectional LSTM was trained with 128 units for 50 epochs. The model continued to overfit and show high fluctuations, but the addition of bidirectional context looks to have reduced the extent of overfitting. The rest of the metrics are similar to that of a unidirectional LSTM.

3.5 Stacked LSTM

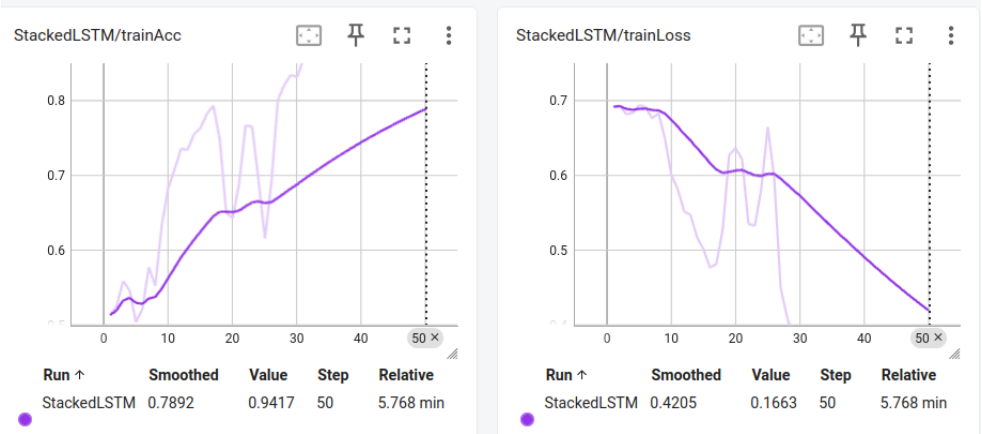

Figure 9: Stacked LSTM train curve

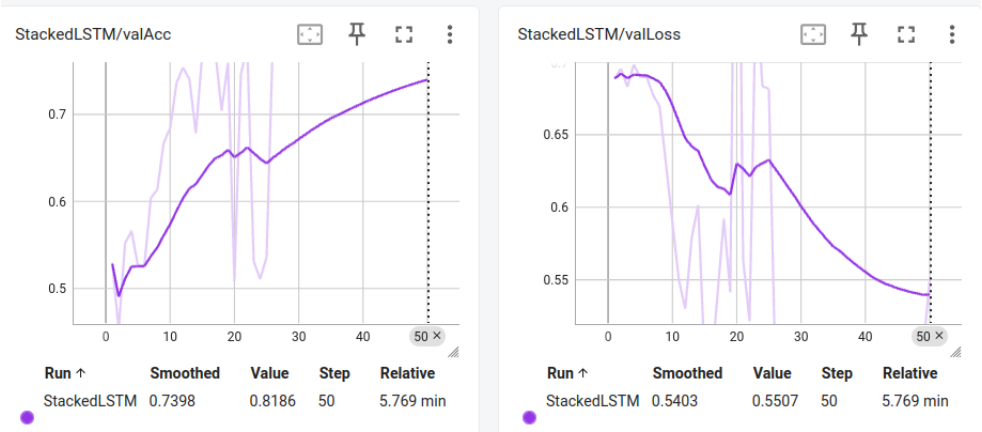

Figure 10: Stacked LSTM val curve

Metric	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Value	0.1663	0.9417	0.5507	0.8186

Table 13: Training and Validation Metrics for BiLSTM Model

Metric	Accuracy	Precision	Recall	F1-score
Value	0.8186	0.8229	0.8055	0.8141

Table 14: Final Validation Metrics

	Predicted 0	Predicted 1
Actual 0	11804	2849
Actual 1	2500	12386

Table 15: Confusion Matrix

A stacked LSTM model was trained, having two layers, one with 128 units and the other with 64 units. The model had the lowest validation loss, even if the validation accuracy remained the same as the other models. It had high fluctuations but the multiple layers showed promise of gradually improving accuracy compared to the flatlining showed by other models. Regularization techniques and training for epochs would likely show better results with this model.

3.6 Stacked BiLSTM

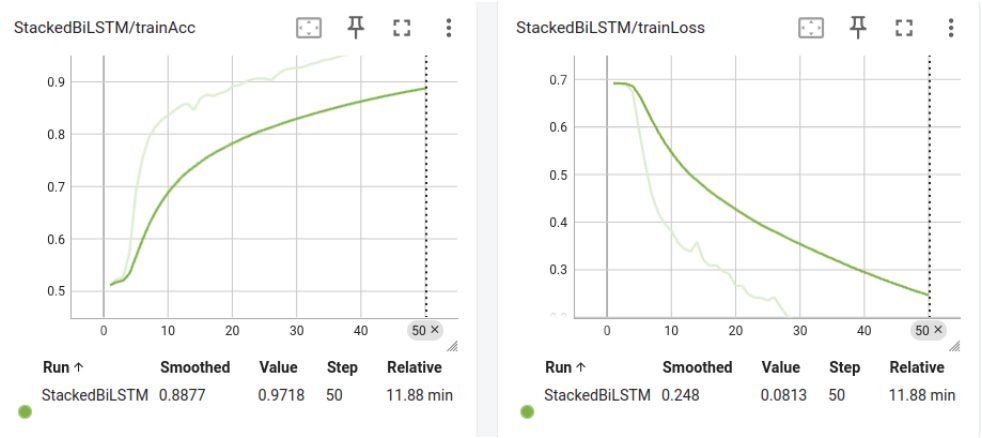

Figure 11: Stacked BiLSTM train curves

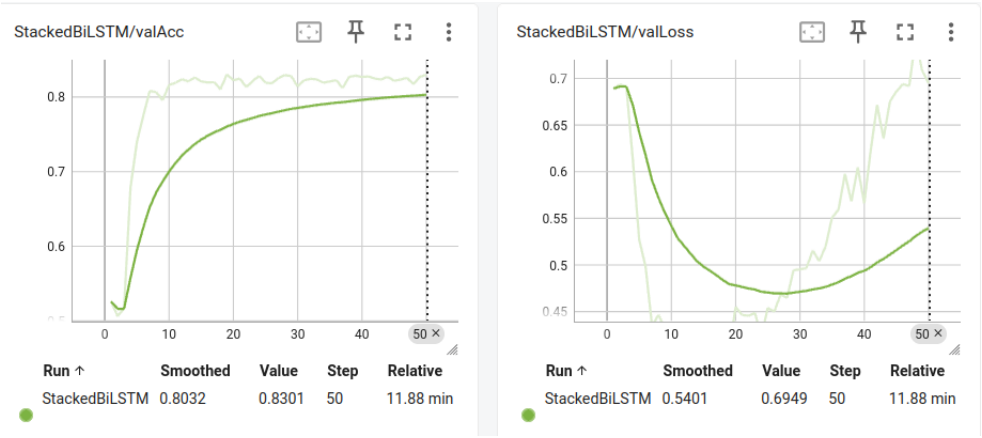

Figure 12: Stacked BiLSTM val curves

Model	Train Accuracy	Train Loss	Val Loss	Val Accuracy
Stacked BiLSTM	0.9718	0.0813	0.6949	0.8301

Table 16: Training and Validation Metrics for Stacked BiLSTM

Metric	Accuracy	Precision	Recall	F1-score
Value	0.8301	0.8471	0.7707	0.8071

Table 17: Final Validation Metrics

	Predicted 0	Predicted 1
Actual 0	12386	2267
Actual 1	2949	11937

Table 18: Confusion Matrix

A stacked bidirectional LSTM with 2 layers, having 128 units in the first layer and 64 units in the second layer was trained for 50 epochs. The bidirectional context resulted in smoother curves, and this model also had the highest precision. There was still a strong indication of overfitting from the curves. This model had the best validation accuracy.

4 Hyperparameters

All the models were trained for 50 epochs, using an Adam optimizer and a learning rate of 0.001. A sigmoid was used as the activation function for the fully connected layer with 0.5 as the threshold for prediction. 20000 was the vocabulary size and 128 was the embedding dimension.

For all the models, the hidden state input was a randomly initialized tensor for each epoch, but within an epoch, the previous hidden state was passed on for each batch. The hidden states were detached from gradient calculation for truncated backpropagation through time.

5 Test Results

Model	**Test Accuracy**
RNN	0.6829
LSTM	0.8177
GRU	0.8198
BiLSTM	0.8087
Stacked LSTM	0.8186
Stacked BiLSTM	0.8301

Table 19: Test Accuracy for All Models

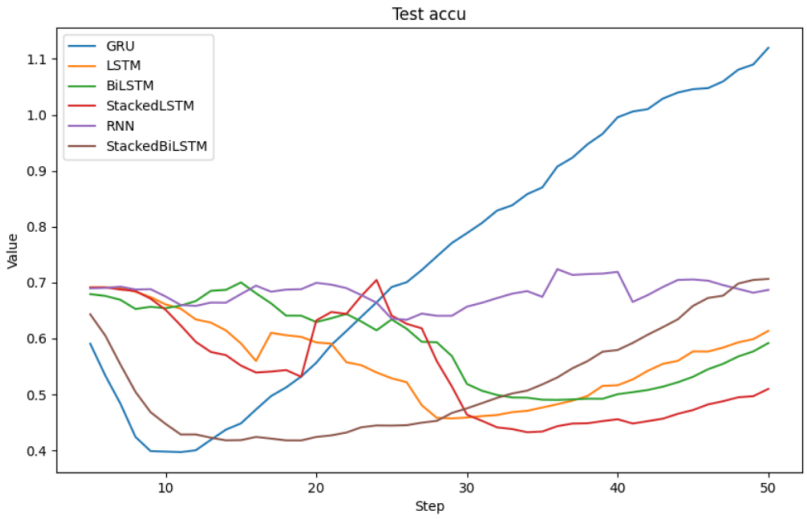

Figure 13: Test loss