**RNN on IMDB Review Data**

**Simple Neural Network:**

A basic Recurrent Neural Network architecture implemented with Keras serves for identifying positive and negative reviews. During preprocessing the sequences underwent tokenization while padding occurred to meet an input length of 150 tokens by restricting the vocabulary to the top 10,000 most frequent words. The initial Embedding layer transforms words to dense vectors with size 128 dimensions before SimpleRNN adds 64 units and tanh activation to understand sequential patterns in the data. A Dropout layer set to 0.5 rate was included to prevent overfitting problems. A Dense layer with sigmoid activation serves for binary classification after processing the output.

Multiple training rounds for ten whole epochs were needed. From the early phase of training up to epoch 3 the model achieved significant advances in its training and validation metrics. The model started to recognize meaningful patterns after reaching epoch 2 when the training accuracy achieved 82.13% and validation accuracy registered at 77.81%. At epoch 3 the training accuracy achieved 90.09% yet validation accuracy stabilized and validation loss started to rise marking the first sign of overfitting.

Throughout epochs nine through ten the training accuracy increased past 99.88% while validation accuracy maintained a fixed range between 76-77% and validation loss kept increasing to reach 1.1751 by the final epoch. This proposed performance difference between training and validation signals overfitting as the model demonstrates outstanding results on training points yet struggles to predict new data properly.

**Layer LSTM Model**

The Simple RNN model limitations led me to implement Long Short-Term Memory (LSTM) networks as a more advanced architecture for better handling long-term dependencies found in movie review sequences. The ability of LSTMs to solve the vanishing gradient problem affecting simple RNNs allows them to grasp sentiment and contextual meaning through extended text inputs.

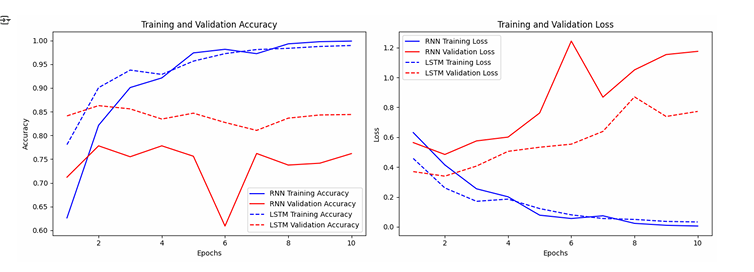
The basic framework of the model stayed consistent yet the SimpleRNN component was exchanged with an LSTM component. The LSTMs trained for 10 epochs using IMDB produced these results for performanceevaluation:

The model began with 78.06% training accuracy while demonstrating 84.11% validation accuracy together with a validation loss of 0.3699. The initial patterns of the data proved easily detectable for the LSTM architecture thus providing better performance than Simple RNN during its first training session.

The second epoch represented the benchmark performance by reaching 90.07% training accuracy along with 86.28% validation accuracy which became the highest achievement. The validation loss descended to 0.3394 which signified the peak of generalization just before overfitting initially appeared during subsequent epochs.

The Long Short-Term Memory model delivered superior performance to Simple RNN during the beginning of training sessions yet provided further performance elevation when trained alone. The validation accuracy reached 86.28% with the LSTM model while it outlasted Simple RNN in terms of generalization performance. Overfitting still appeared in the LSTM model but the onset was postponed along with its impact being reduced.

**Difference in the Simple RNN and LSTM**

****

Throughout training and validation, the Simple Recurrent Neural Network (RNN) shows a distinct difference compared to the Long Short-Term Memory (LSTM) model performance. Both the models show typical increasing trends in their training accuracy throughout the training phase. The LSTM model provides better training success rates throughout the ten epochs before achieving almost faultless results. The Simple RNN achieves better training accuracy yet it stops growing near a much lower figure when compared to LSTM results.

When examining validation accuracy the LSTM model maintains superiority by showing consistent better performance than the Simple RNN. Union of observed data patterns demonstrates that LSTM recognizes an improved capability for transferring learned knowledge to uncharted data. Among the two models the Simple RNN shows progressively worse validation accuracy since it cannot reach the same performance level of its LSTM counterpart. During the middle epochs the Simple RNN shows a sudden decrease in validation accuracy which suggests it overfit the training data at that time.

The training loss data displays expected downward patterns through training time which shows that the models decrease their error rate. The LSTM model demonstrates superior performance by reducing total training loss levels better than the Simple RNN model does. An additional confirmation of this observation emerges from studying the validation loss statistics. A lower stable validation loss displayed by the LSTM model demonstrates its better potential for generalization. The validation loss of Simple RNN significantly increases with an erratic pattern around the seventh epoch, thus demonstrating its weak ability to generalize effectively in comparison to LSTM.

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **Simple RNN (Epoch 2)** | **LSTM (Epoch 2 - Best Epoch)** | **Difference (LSTM - RNN)** |
| Training Accuracy | 82.13% | 90.07% | +7.94% |
| Validation Accuracy | 77.81% | 86.28% | +8.47% |
| Training Loss | 0.4146 | 0.2605 | −0.1541 |
| Validation Loss | 0.4846 | 0.3394 | −0.1452 |

**Simple Embedding Layer**The model with trainable embedding reached 85.45% validation accuracy in epoch 3 and matched the best performance of LSTM at 86.28%. When trained during the initial periods the model effectively developed semantic word representations which resulted in improved metrics for both training and validation sets.

After epoch 4 began the training accuracy rise to 97.48% while validation accuracy neither improved nor decreased and validation loss started to increase indicating significant overfitting. The system demonstrates outstanding capability to understand the training data yet it reveals diminishing capabilities for generalization.

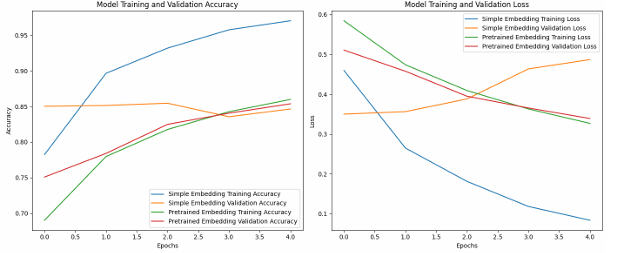
**Pretrained Model using Word Embedding Matrix:**

During the five epochs the pretrained-word-embedding model provided gradual improvements in every metric without producing any signs of abrupt overfitting. High-quality embeddings derived from large datasets such as Wikipedia or Common Crawl present classical smooth performance curves because they naturally encode semantic meaning among their words.

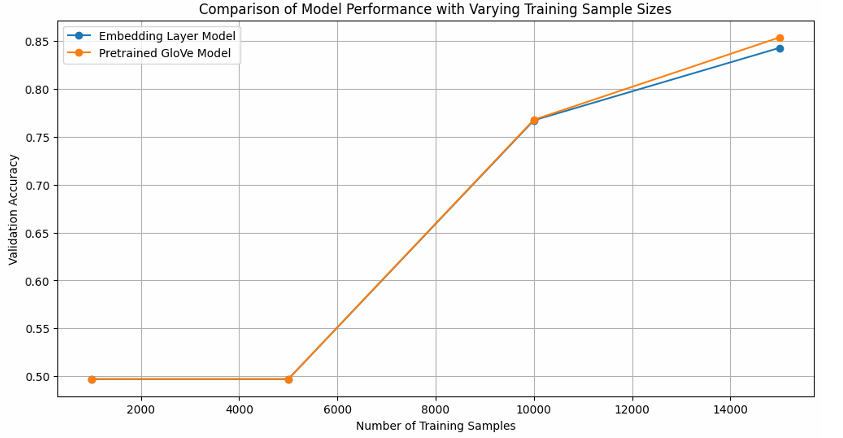
By epoch 5, the model achieved:

85.67% training accuracy, indicating strong learning,

The model achieved 85.37% validation accuracy while maintaining almost equivalent training results and displaying 0.3391 validation loss which outperformed the custom-trained embedding model's optimal loss of 0.3882. The model exhibits strong generalizability because its validation accuracy matches training performance closely because of the high quality of its pretrained vectors.



**Models on Different Number of Samples**

****

A comparison of validation accuracy appears in the provided graph which displays results of **Embedding Layer Model** alongside Pretrained Glove Model at different training sample numbers. As shown in the chart by the blue line the validation accuracy of the Embedding Layer Model shows steady growth with higher training sample counts. Models start at an initial low accuracy level using fewer training samples before reaching maximum validation accuracy at the point where it is trained with the most numerous samples. The Embedding Layer Model achieves better word representation skills and generalization potential through an expanded dataset.

The **Pretrained Glove Model** (shown through the orange line) demonstrates an opposite relationship than the other model types. A limited training sample count initiates the validation accuracy at a superior value compared to the Embedding Layer Model. When the training sample quantity extends from minimal to next level there is no perceivable enhancement in system performance. The validation accuracy makes a significant upward leap when training samples pass a specific threshold before gradually matching Embedding Layer Model results at maximum sample size. The Pretrained Glove Model demonstrates strong word embedding competency after its training on extensive external data because it already understands embeddings well despite limited task data. A Pretrained Glove Model requires extended task-specific training data for complete potential realization because its initial word embedding performance requires adequate adjustment for a specific task. Once the available data becomes sufficiently large the performance levels of both approaches converge to identical results.