WORD EMBEDDING USING EMBEDDING LAYERS AND A PRE-TRAINED MODEL

I have taken the IMDB dataset to perform sentiment analysis in this project. The dataset consists of 50000 reviews (25000 positive and 25000 negative), divided into 25000 for training and 25000 for testing.I have preprocessed and trained a bidirectional LSTM model for sentiment analysis.

In this project, I have examined how word embedding works using a pre-trained model and also using embedding layers. Specifically, I have done the following: -

1. Cutoff reviews after 150 words.
2. Restricted training samples to 100.
3. Validated on 10000 samples.
4. Considered only the top 10000 words.
5. Determining which approach works best requires considering an embedding layer and pre-trained word embedding.
6. Determining the optimum number of training samples by changing the training samples.

Let's discuss each case separately. The findings are as follows: -

1. Training on a first basic sequence model: -
   In this, I have trained the dataset on a first basic sequence model. The findings are as follows: -

Model	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Test Accuracy
Model 1 - First Basic Sequence Model	0.9712	0.8080	0.1094	0.4928	0.816

The first basic sequence model was created to establish the baseline performance for the project and to compare it against all the other models. From the above findings, we can say that the model has learned to fit the training data well but maybe overfitting to the validation set. The test accuracy is higher than the validation accuracy, indicating that the model has performed well on unseen data.

2. Training a Model that uses word embedding from scratch: -
   In this model, I have created a model from scratch that uses word embedding. In this model, I have not enabled masking to see how the model performs and the findings are as follows: -

Model	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Test Accuracy
Model 2 – Embedding layer from scratch	0.9869	0.7760	0.0458	0.7038	0.794

The second model performed which used word embedding achieved better training accuracy but slightly lower accuracy compared to the first basic sequence model. The training loss was lower than the first model, but the validation loss was higher. This suggests that the model is overfitting to the training data and is not generalizing well to unseen data. This usually happens when masking is not enabled as the model may be learning from padded values that don't contain meaningful information. From the above findings, we can infer that creating word embedding from scratch can improve training accuracy, but proper regularization techniques such as masking should be implemented to prevent overfitting.

3. Training a Model that uses word embedding from scratch with Masking enabled: -
   In this model, I have created a model that is like the 2nd model, but the only modification is that I Have enabled masking. The findings are as follows: -

Model	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Test Accuracy
Model 3– Embedding layer from scratch with Masking enabled	0.9859	0.7997	0.0487	0.5620	0.798

In this model, we have an additional model called masking. Masking means the model ignores padded zeros and focuses only on the actual input data. Training accuracy is slightly lower than the second model, but the validation accuracy has improved significantly. This means that this model can handle variable length sequences better than the previous model. The above findings show that masking is an important feature to consider while using word embedding.

4. Using a Pre-trained word embedding: -
   This model is trained using pre-trained word embedding. We have used the GloVe word embedding to see how the model performs. The findings are as follows: -

Model	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Test Accuracy
Model 4– Pretrained Word Embedding	0.8200	0.7787	0.4091	0.4828	0.775

From the above findings, we can say that using pre-trained word embedding like GloVe may not always result in better performance. The training accuracy for this model is relatively lower than all the previous models. This indicates that the model did not learn the features of the data well. This may be because the pre-trained model may have been trained on a different corpus and may not have captured the nuances and context of the specific language used in the dataset. Overall, we can say that while pre-trained embedding can be useful in some scenarios, it is always necessary to experiment with different embeddings or to fine-tune the embedding for better performance.

5. Using different Training samples: -
   Finally, I have tried using different training samples to determine the point where the embedding layer performs the best. The findings for different training samples are as follows: -

Model	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Test Accuracy
1000 Training Samples	0.9880	0.8130	0.0336	0.6396	0.803
5000 Training Samples	0.9902	0.7210	0.0366	1.3098	0.800
10000 Training Samples	0.9895	0.8200	0.0319	0.7755	0.82
15000 Training Samples	0.9933	0.8080	0.0269	0.7837	0.806
20000 Training Samples	0.9890	0.8090	0.0331	0.6517	0.801
25000 Training Samples	0.9927	0.8130	0.0206	0.7688	0.805

From the above table, we can infer that the performance of the model does not necessarily improve by increasing the training samples. For example, the model trained with 5000 samples has a lower validation accuracy and higher validation loss compared to the model trained with 1000 samples. This could be due to overfitting or the noise present in the additional training data.

To have a clear understanding, I have summarized the results of all models into a combined table. The table is as follows: -

Model	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Test Accuracy
Model 1 -First Basic Sequence Model	0.9712	0.8080	0.1094	0.4928	0.816
Model 2 – Embedding layer from scratch	0.9869	0.7760	0.0458	0.7038	0.794
Model 3– Embedding layer from scratch with Masking enabled	0.9859	0.7997	0.0487	0.5620	0.798
Model 4– Pretrained Word Embedding	0.8200	0.7787	0.4091	0.4828	0.775
Model 5 - Embedding layer with 1000 Training Samples	0.9880	0.8130	0.0336	0.6396	0.803
Model 5 - Embedding layer with 5000 Training Samples	0.9902	0.7210	0.0366	1.3098	0.800
Model 5 - Embedding layer with 10000 Training Samples	0.9895	0.8200	0.0319	0.7755	0.82
Model 5 - Embedding layer with 15000 Training Samples	0.9933	0.8080	0.0269	0.7837	0.806
Model 5 - Embedding layer with 20000 Training Samples	0.9890	0.8090	0.0331	0.6517	0.801

Model 5 - Embedding layer with 25000 Training Samples	0.9927	0.8130	0.0206	0.7688	0.805

Findings and Conclusion: -

- The first basic sequence model resulted in a good performance on test data, indicating that the model was able to generalize well to unseen data.
- Training a model that used word embedding from scratch without masking resulted in overfitting to the training data, suggesting that proper regularization techniques such as masking should be implemented.
- The model with word embedding from scratch with masking performed better than the one without masking, which shows it is always important to consider masking while embedding.
- The model with word embedding from scratch with masking performed better than the pre-trained word embedding.
- Using pre-trained word embedding may not always result in better performance.
- Increasing the training samples does not guarantee better performance as seen in the model trained with 5000 samples.
- The training loss decreased with the increase in training samples, indicating the model was able to learn better with more data.
- The validation loss increased with the increase in training samples, indicating that the model was not able to generalize well on unseen data.
- Proper regularization techniques such as masking and experimenting with different embeddings or fine-tuning the embedding can improve the performance of the model.
- Regularization techniques such as masking and experimenting with different embeddings can improve the performance of the model.
- The optimal number of training samples can be locked at 15000 as the model with 15000 training samples performed the best.

In conclusion, all the models have performed similarly to one another. There is not much change in the accuracies. This says that there is no one-size-fits-all approach when it comes to embedding layers. Different models should be tested to determine the right model for the given dataset. We can use regularization techniques, different embedding sizes, and LSTM layers, training pre-trained embedding on the given dataset and increasing the training sample to a certain point to enhance better performance.