Assignment 4

Trained models with different sample sizes:

The model has been trained with sample sizes of 1000, 2500, 5000, 7500, and 10000 using techniques such as one-hot encoded sequence, Embedded, embedded masked, and pre-trained and calculated accuracy and loss using test data. The observations are recorded in below table,

sample size	one hot encoded sequence		Embedded		Embedded masked		pre trained	
	Test Loss	Test Accuracy	Test Loss	Test Accuracy	Test Loss	Test Accuracy	Test Loss	Test Accuracy
1000	0.681	0.662	0.6708	0.5857	0.6586	0.608	0.6787	0.6134
2500	0.652	0.565	0.7138	0.6067	0.7351	0.6192	0.6193	0.6696
5000	0.659	0.599	0.7226	0.7108	0.8353	0.7032	0.5391	0.7248
7500	0.493	0.696	0.5375	0.7924	0.765	0.7645	0.5137	0.7836
10000	0.459	0.803	0.5413	0.8024	0.6191	0.7878	0.4452	0.8065

Train sample 1000, Validation 10000, Test 25000:

Starting setup:

1. The data is related to IMDB dataset and is used for this assignment.
2. The first step is to train the sample of 1000 with a review of 150 words max and a total of 10000 words are taken as input for the model.
3. The second step is to validate the model against 10000 validation samples of both positive and negative reviews.
4. The loss function we used is “binary cross-entropy” and was used as it was a classification model with optimizer “Adam”.

Models Trained:

1. One hot-encoded sequence model performance on test data has achieved accuracy of 0.662 and loss of 0.681.
2. The embedded model without masking gave a loss of 0.6708 and an achieved accuracy of 0.5857.
3. An embedded model with masking gave a test loss of 0.6586 and an achieved accuracy of 0.608.

4. Global Vectors for word representation (GloVe), a pre-trained model gave a test loss of 0.6787 and achieved accuracy of 0.6134.

Based on the analysis, using RNNs the IMDB data significantly performed good in both test loss and test accuracy with embedded layers compared to other word embedding techniques.

As the sample size increased the performance of RNN model improved. The sample being increased from 1000 to 10,000 samples, the test accuracy is increased. As the model is trained for larger data the better it would outperform. Because the model learns from the data which can perform better.

When we particularly compare between the standard embedded and masked embedded layers. Standard embedded layer performs better in test accuracy. In masking technique allows model to focus only on actual word embeddings. As we observed there is no effect of masking on the IMDB dataset.

In pre trained word embeddings – GloVe embeddings resulted a better and effective model compared to training embedded layer. The pre-trained model achieved a test accuracy of 0.8065 when trained on 10,000 samples, outperforming both masked and standard embedded layers in terms of test accuracy.

Conclusion:

The model's accuracy is usually dependent on the data. The more the data is trained on a model the performance would be better. As the sample size increases, the model has more data to learn from and is likely to generalize better to unseen data.

Depending on the given task and requirement, the model architecture may change and with that the sample size will change, we need to try with different configurations and find the optimal size. In the provided Imdb dataset, data sizes of 5000 and 10000 were able to give good results in terms of accuracy and loss on unseen data, the lower values of these metrics indicate better performance. These sample sizes consistently improved performance across different embedding techniques, including masked and standard embedded layers, as well as GloVe embeddings.

The GloVe pre-trained embeddings model consistently showed better performance in terms accuracy and loss across different sample sizes, outperforming both embedded and one-hot encoded sequence models. The masked embedded layers also showed promising results, but they did not consistently outperform the standard embedded layers or the pre-trained GloVe embeddings. Therefore, it can be concluded that the GloVe embeddings is more efficient for sentiment analysis tasks compared to other embedding techniques, as they capture extensive semantic and syntactic information from large corpora, reduce the need for large training data, provide a standardized representation, and are easy to implement and use.