Assignment 4

Text and Sequence Data

Summary

Problem:

Consider the IMDB example from Chapter 6. Re-run the example modifying the following:

1. Cutoff reviews after 150 words.

2. Restrict training samples to 100.

3. Validate 10,000 samples.

4. Consider only the top 10,000 words.

5. Consider both a embedding layer, and a pretrained word embedding. Which approach did better? Now try changing the number of training samples to determine at what point the embedding layer gives better performance.

Objective:

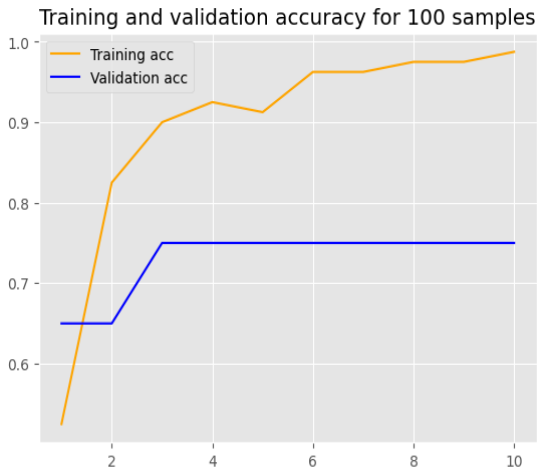
The aim of the current IMBD dataset ‘s binary classification is to sort out the negative and positive reviews. Out of the 50000 reviews in the dataset we are considering only the top 10000 for the given problem and by varying sample sizes with 100, 3000,7000 and 10000. Both validation and test are done by preparing the data. Then, the data is put into an embedding layer and pretrained network which is from Stanford large movie dataset.

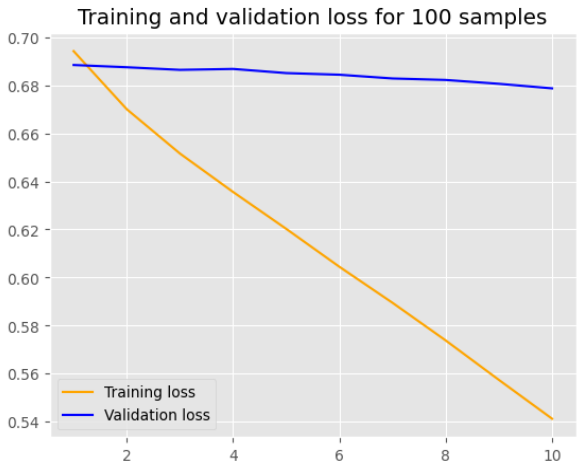
Data Preprocessing and Procedure:

Here each review is converted to word embeddings. In this model, we explored two embedding techniques called customized training embedding layer and a pretrained embedding layer based on the GloVe model. Here, we specially trained the 6B version of the GloVe model which consists of Giga word 5 and Wikipedia data containing 400000 words and 6 billion tokens.

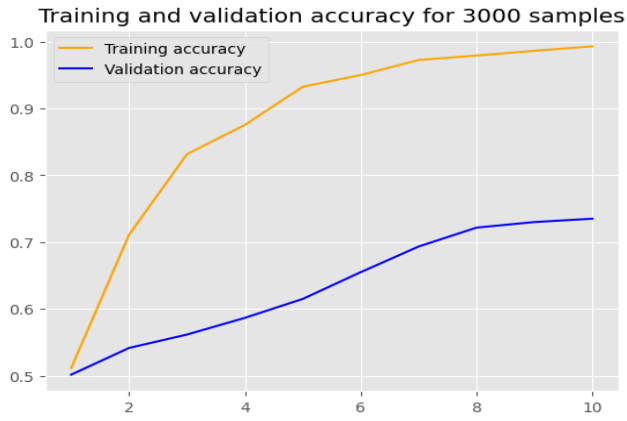
Customized Trained Embedding layer:

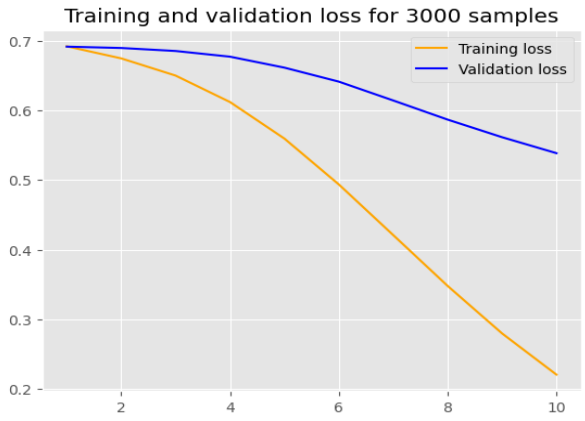
Customized trained embedding layer with 100 samples:



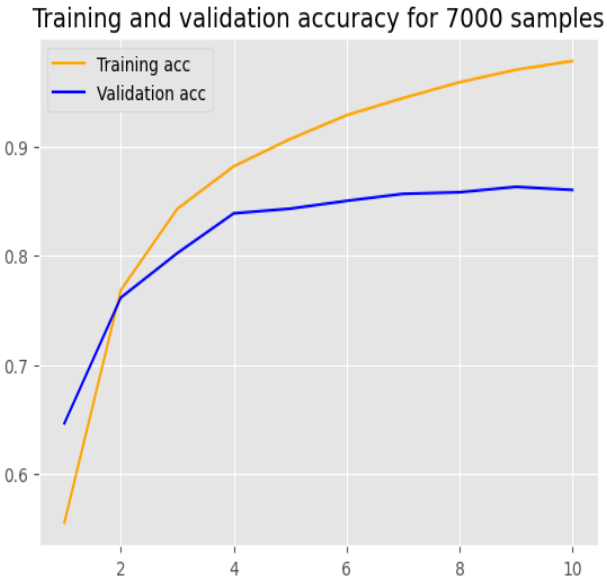


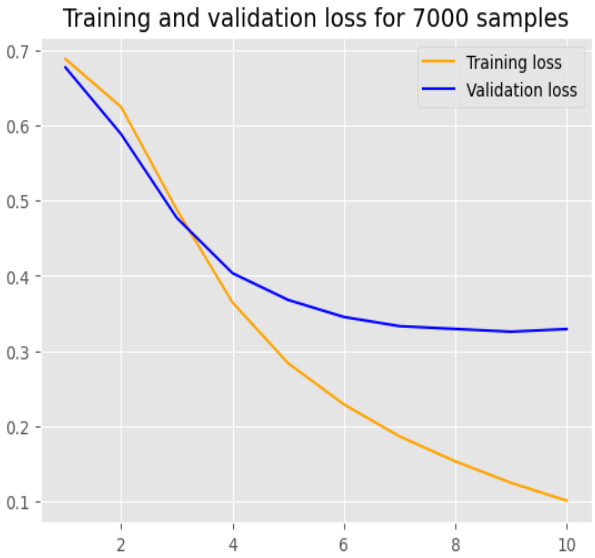
Customized trained embedding layer with 3000 samples:



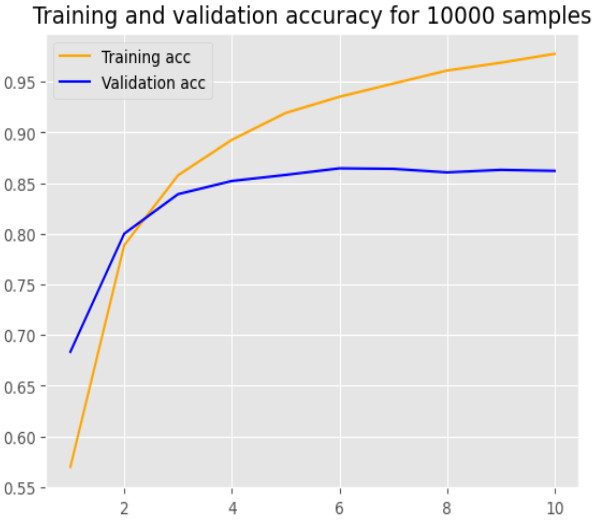


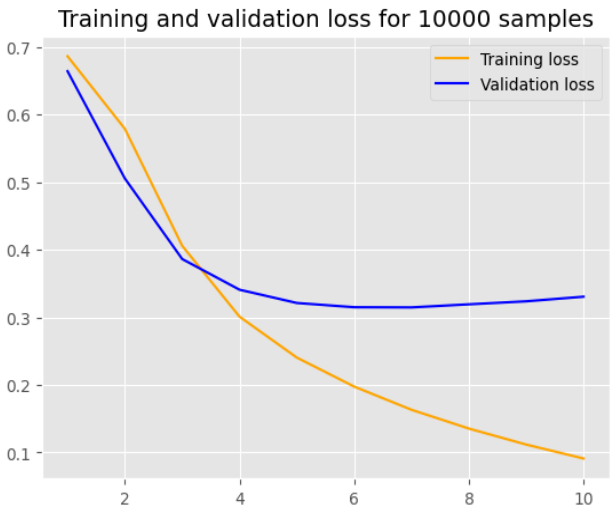
Customized trained embedding layer with 7000 samples:





Customized trained embedding layer with 10000 samples:

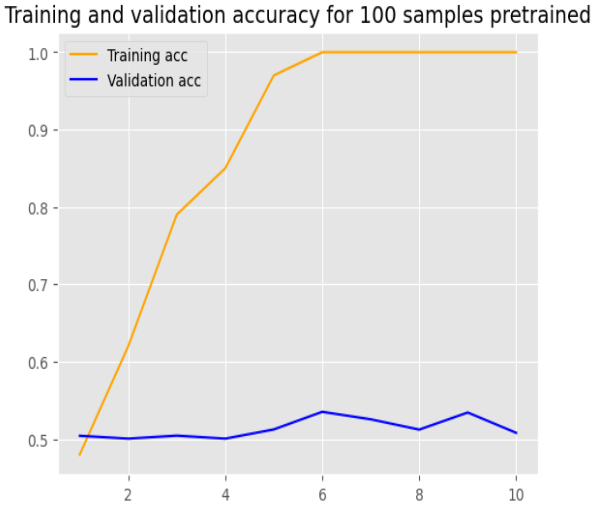


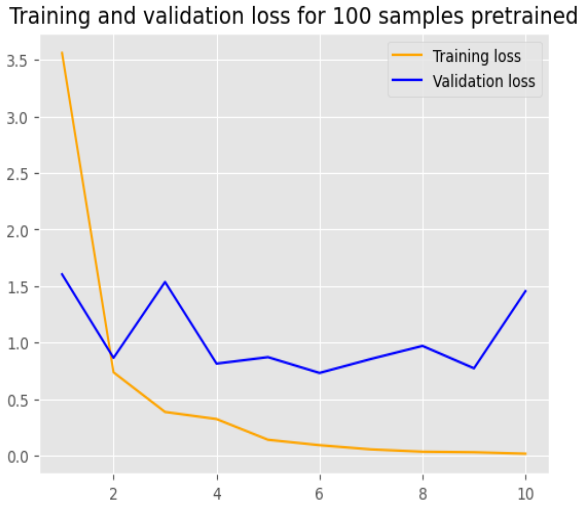


The above customized trained embedding layer varied from 97% to 99% by trying different training samples of 100, 3000, 7000 and 10000. Whereas test loss varies from 30% to 70%.

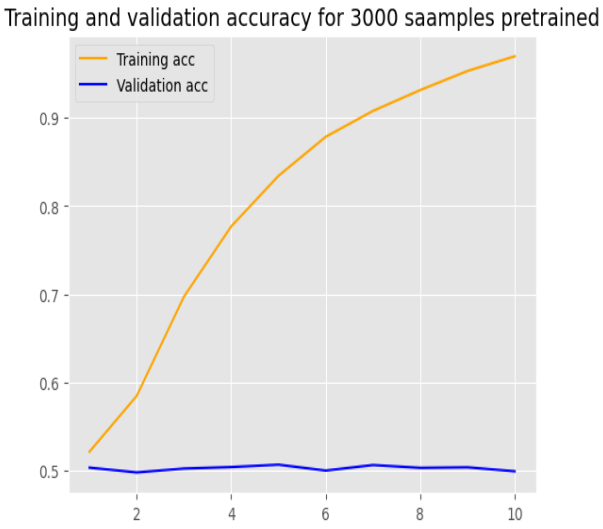
Using Pretrained word embedding layer

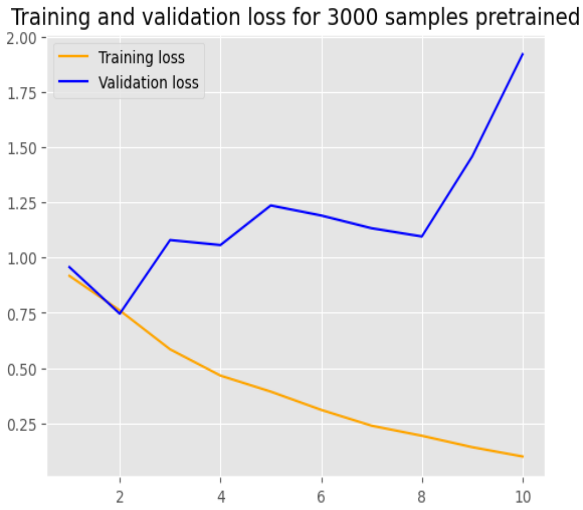
Pretrained word embedding layer with 100 samples:



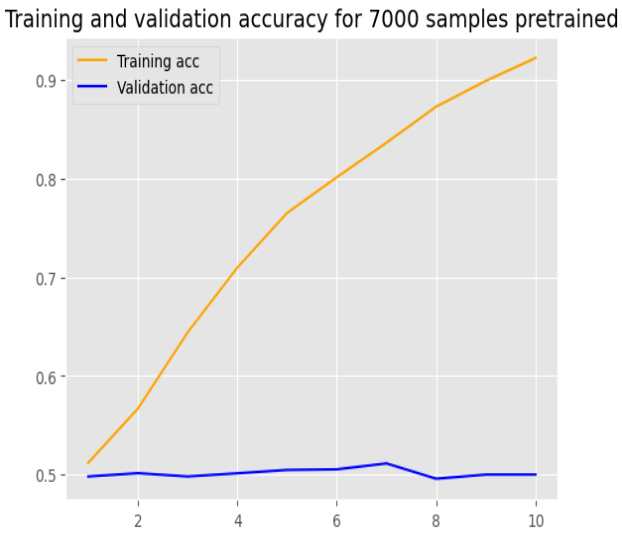


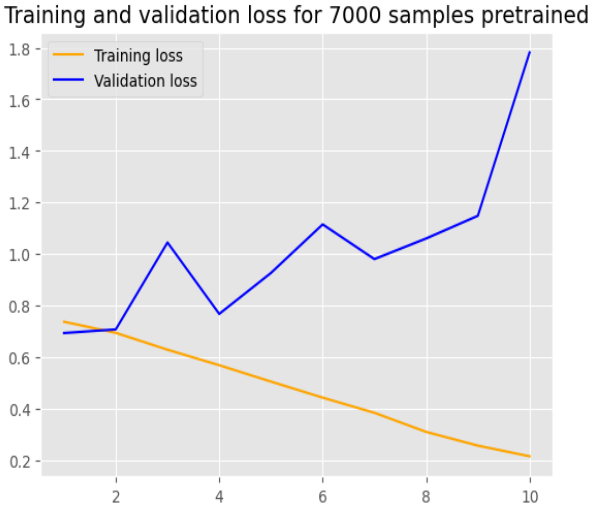
Pretrained word embedding with 3000 samples:



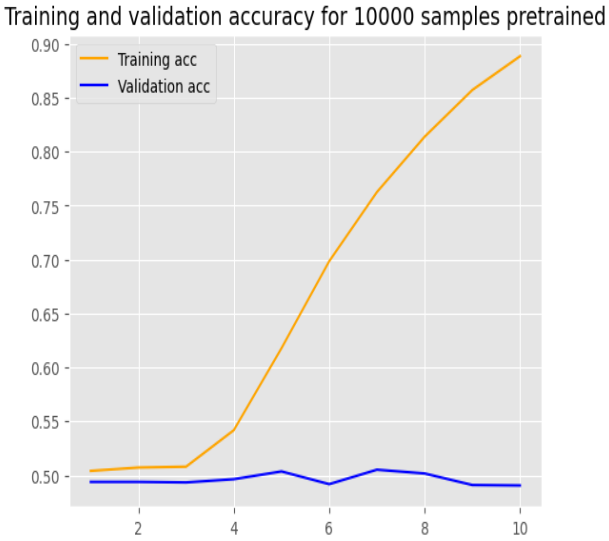


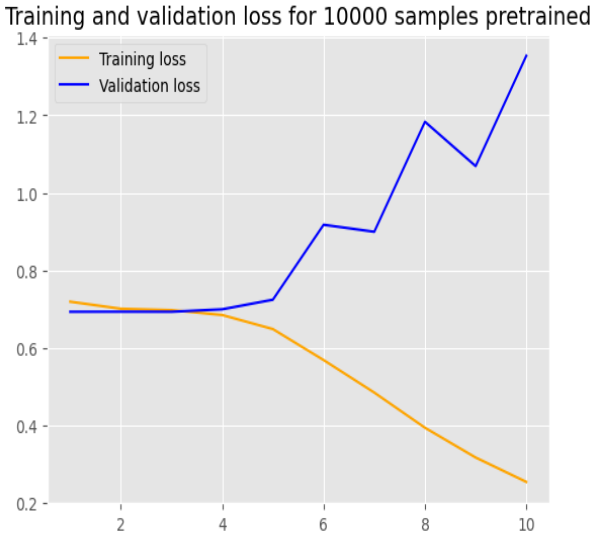
Pretrained word embedding layer with 7000 samples:





Pretrained word embedding layer with 10000 samples:





And we used the customized training embedding layer with different training samples sizes as 100, 3000, 7000 and 10000. Then we observed the training and test accuracy and that results we compared with pretrained word embeddings (GloVe) while using different sample sizes like the same as above method. And the results were compared in below table:

In the below table the model has been trained with sample sizes of 100, 3000, 7000 and 10000 and their training and test accuracy and test loss.

Embedding Technique	Training Sample Size	Training accuracy	Test Accuracy	Test Loss
Customized-trained Embedding Layer	100	0.98	0.49	0.69
	3000	0.99	0.73	0.54
	7000	0.97	0.84	0.34
	10000	0.97	0.85	0.33
Pretrained word embedding (GloVe)	100	100	0.49	1.6
	3000	0.97	0.49	1.94
	7000	0.92	0.5	1.72
	10000	0.88	0.49	1.34

Conclusion:

From the above table, the customized trained embedding layer outperformed the Pretrained word embedding (GloVe), when we see the above data for the pretrained word embedding the sample size 10000 got less training accuracy. However, the embedding layer which is customized is preferable choice because of few computer resources and an average training sample is needed as long we did not consider the overfitting. Lastly, we can say that for small sample sizes computationally efficient whereas larger sample sizes may offer better accuracy but may strain memory and processing capabilities.

The results showed that the customized training model can work well when there is a shortage of data, but when there are more training instances available will use pretrained embeddings is more beneficial. If we have large training data, pretrained embeddings offers better foundation due to training on pretraining large databases that enables faster and more efficient learning of word representation.