Assignment 4

Text and Sequence Data

Summary

Problem:

This task involves binary classification of IMDB movie reviews to distinguish between positive and negative sentiments. By modifying the example from Chapter 6, we implemented the following changes:

1. Limited each review to a maximum of 150 words.
2. Used only 100 samples for training.
3. Validated using 10,000 samples.
4. Restricted vocabulary to the top 10,000 words.
5. Evaluated performance using both a custom embedding layer and a pretrained word embedding model.

The aim was to assess the effectiveness of each approach and identify the point at which the custom embedding layer outperforms pretrained embeddings.

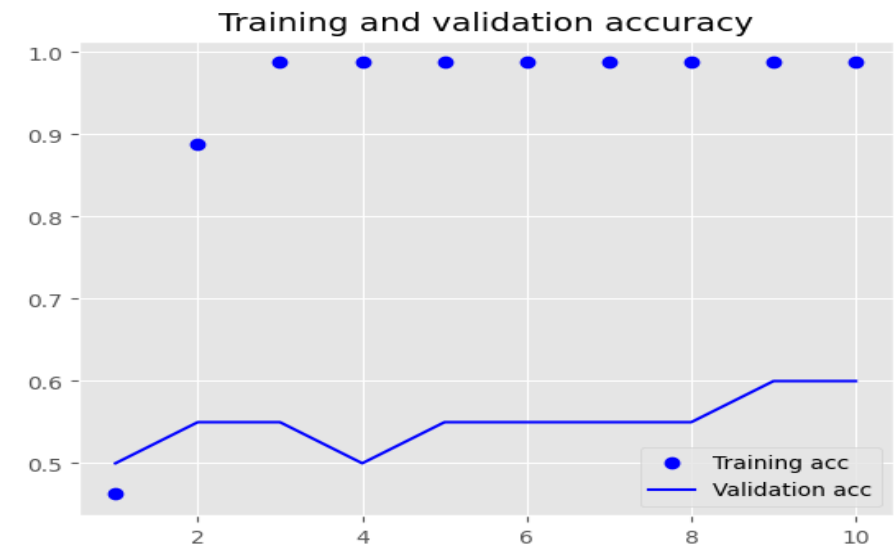
Objective: The primary goal is to classify the IMDB dataset's 50,000 reviews into positive and negative categories. This is done by selecting the top 10,000 words and varying the training sample sizes (100, 3,000, 7,000, and 10,000). The dataset was prepared for validation and testing and then processed through both custom-trained and pretrained embedding layers using data from Stanford's large movie dataset.

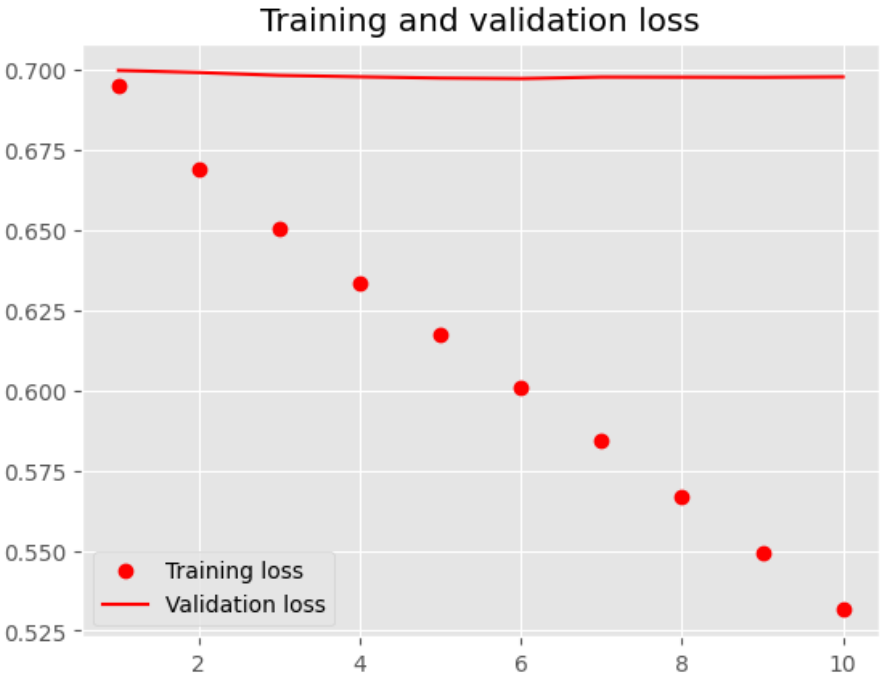
Data Preprocessing and Procedure:

Data Preprocessing and Procedure: Each review was converted into word embeddings using two methods:

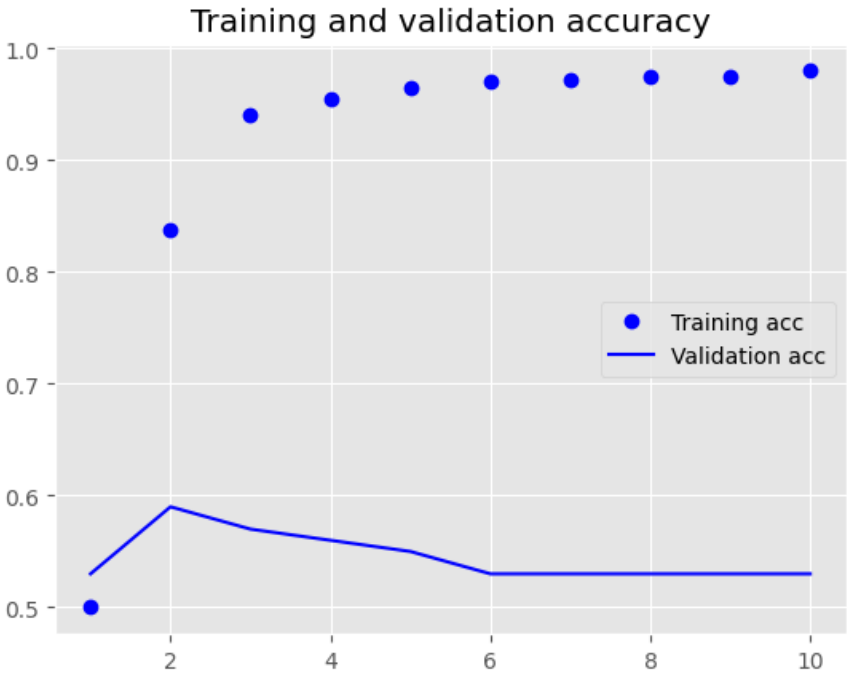
Custom-Trained Embedding Layer: A specialized embedding layer trained on the data.

Pretrained Embedding Layer: A GloVe model (6B version) trained on GigaWord 5 and Wikipedia data, which includes 400,000 words and 6 billion tokens.

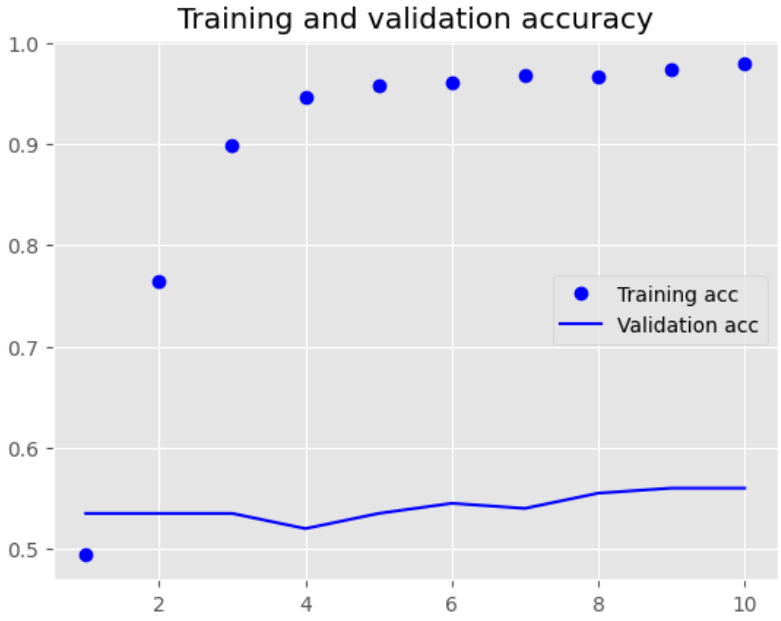

Training and validation accuracy
1.0
0.9
0.8
0.7
0.6
0.5
2
4
6
8
10
Training acc
Validation acc


Training and validation loss
0.700
0.675
0.650
0.625
0.600
0.575
0.550
0.525
2
4
6
8
10
Training loss
Validation loss

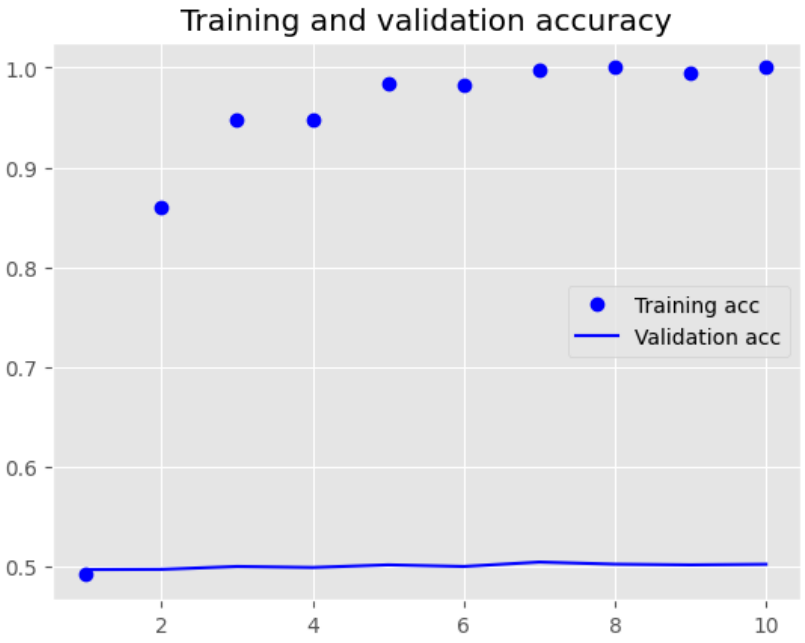
Customized trained embedding layer with 3000 samples:

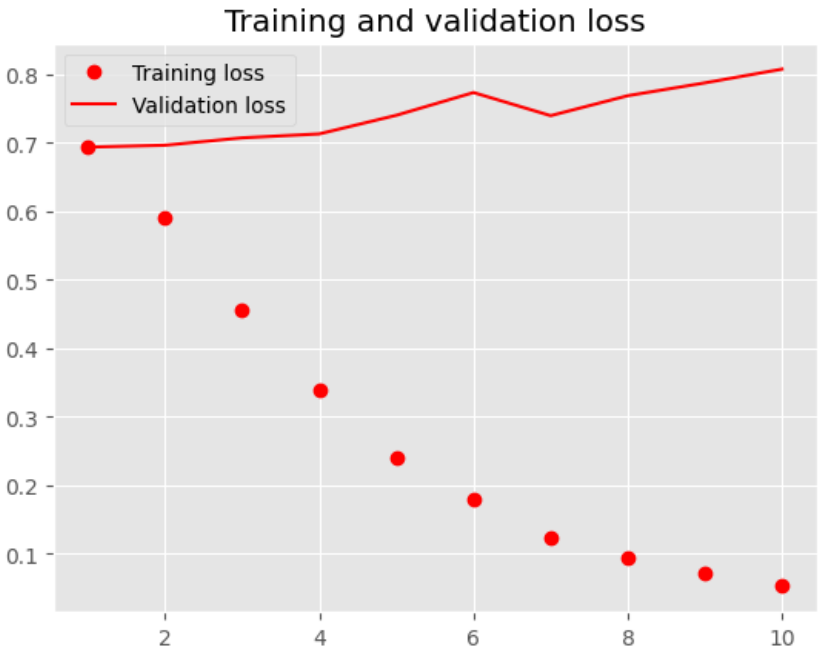


Customized trained embedding layer with 7000 samples:



Customized trained embedding layer with 10000 samples:

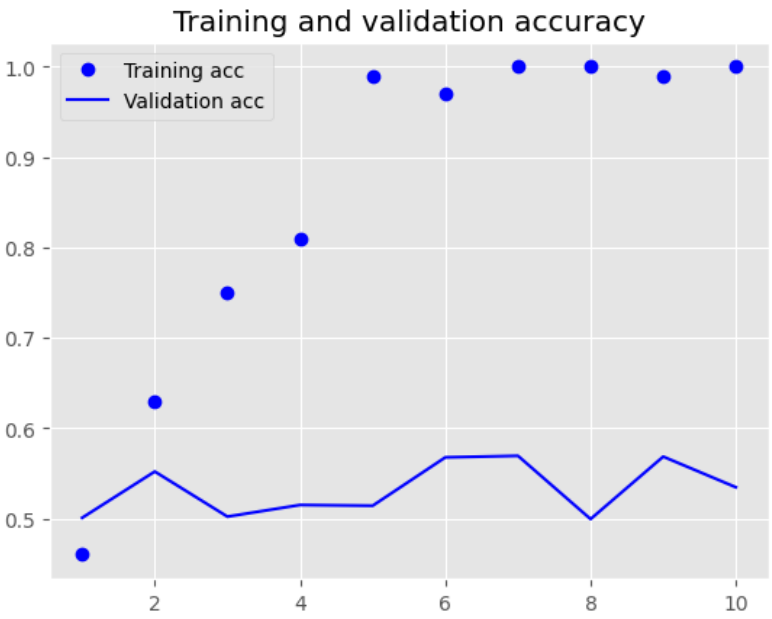


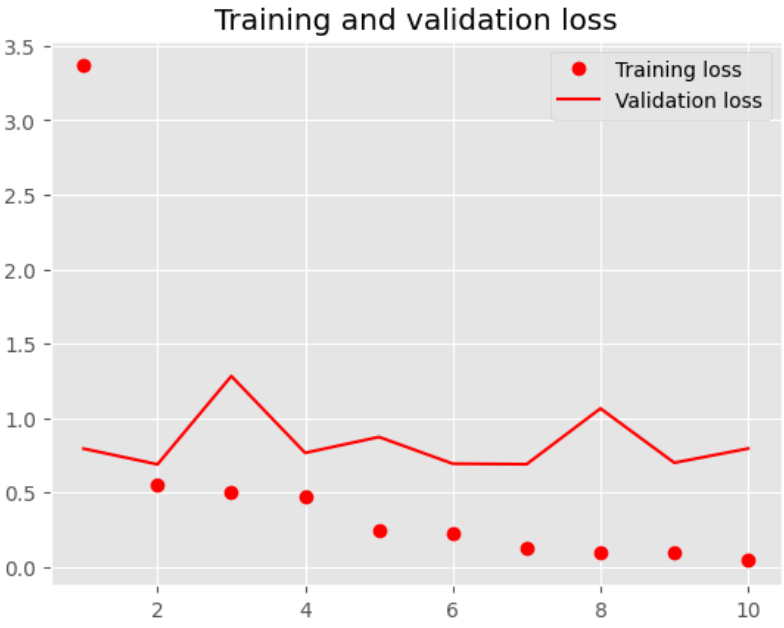


The above customized trained embedding layer varied from 97% to 99% by trying differenttraining samples of 100, 3000, 7000 and 10000. Whereas test loss varies from 30% to 70%.

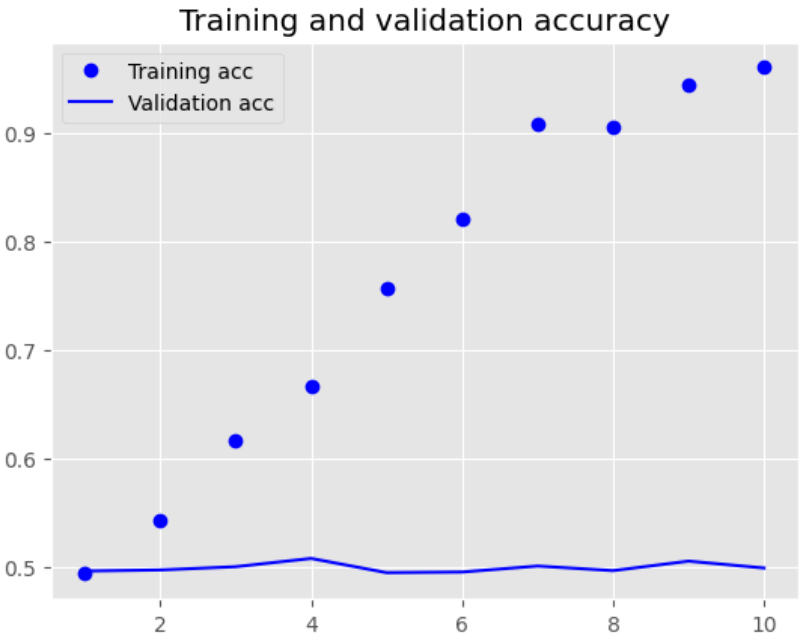
Using Pretrained word embedding layer

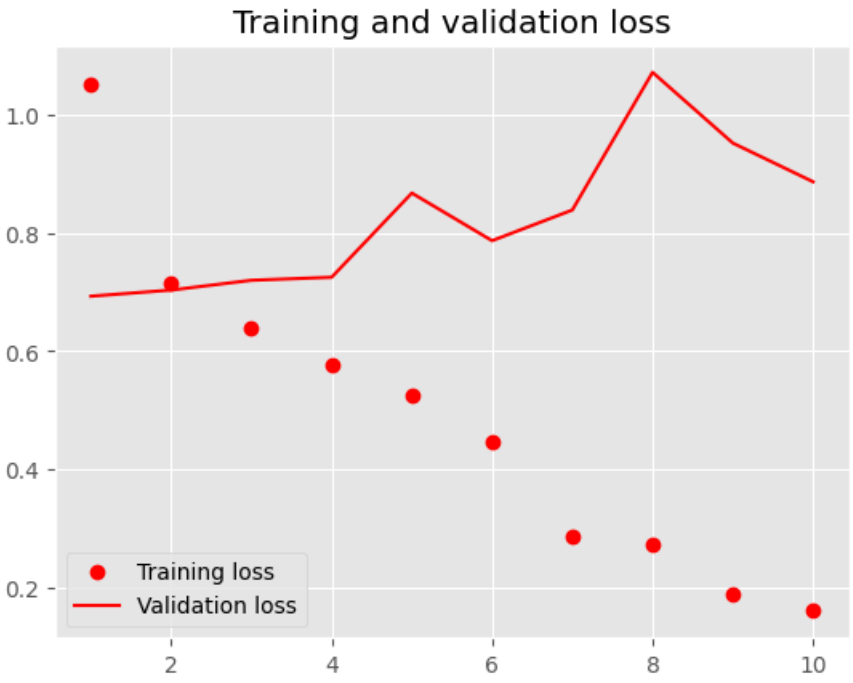
Pretrained word embedding layer with 100 samples:



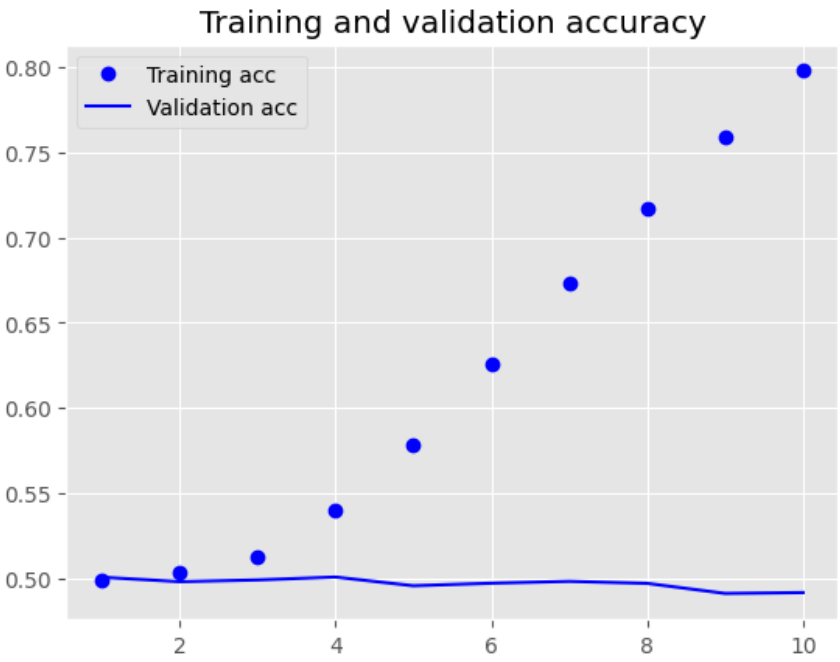


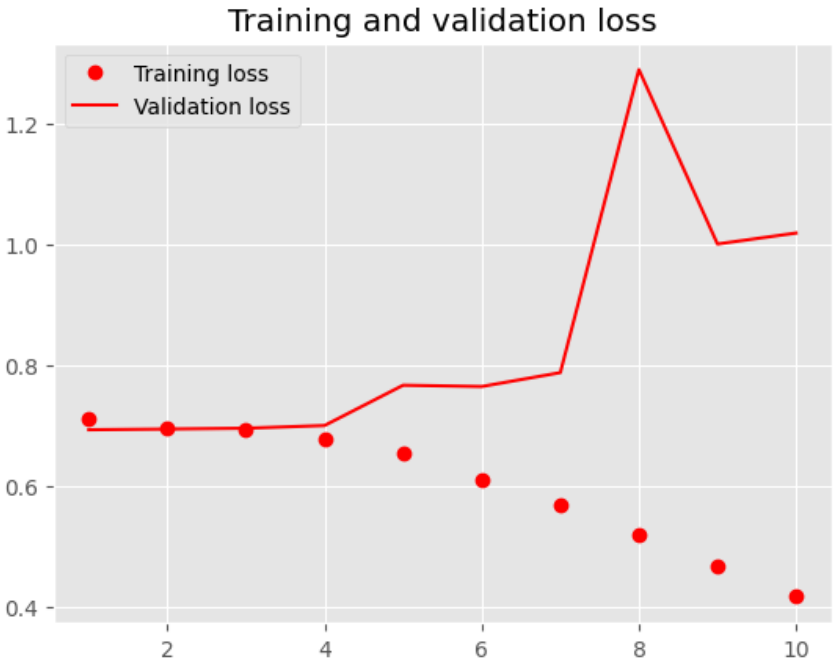
Pretrained word embedding with 3000 samples:





Pretrained word embedding layer with 7000 samples:





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 and10000 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:

The table demonstrates that the custom-trained embedding layer outperformed the pretrained word embedding model (GloVe). Notably, the pretrained embedding model showed reduced training accuracy with a sample size of 10,000. The custom embedding layer is generally a better choice due to its lower computational resource requirements and suitability for moderate training sample sizes, provided overfitting is not a concern.

In conclusion, custom-trained embeddings are computationally efficient and effective for smaller datasets, while larger datasets may benefit from pretrained embeddings. Pretrained embeddings, trained on extensive databases, provide a solid foundation for learning word representations, enabling faster and more efficient performance when substantial training data is available.