Assignment 4

Text and Sequence Data

Bhumika Shah

811286091

Objective

This is a binary classification on the IMDB dataset for movie reviews, either positive or negative. The IMDB dataset consists of 50,000 reviews; however, only the top 10,000 words are considered in this dataset. The number of training samples ranges from 100 to 100,000, and 10,000 samples for validation. Before training, the data preparation includes an embedding layer and a pre-trained embedding model. A different approach is tried to see the performance.

Data Preprocessing

- Word Embeddings:

These are the methods of converting the text into word embeddings, representing words as fixed-size vectors. This transformation will be restricted to 10,000 samples. Another review conversion is into numerical sequences, where each number will correspond to a certain word. It is not possible to provide such sequences directly to the neural network.

- Tensor Construction:

Tensors are designed from these numerical sequences as the next step in the attempt to solve this. Each sample takes the form of a (samples, word indices) tensor. To maintain consistency in length across samples, methods like padding with placeholder words or numbers are applied.

Methodology

This study compares two approaches to generating word embeddings for the IMDB dataset:

A. Custom-trained Embedding Layer

B. Pretrained Embedding Layer (GloVe Model)

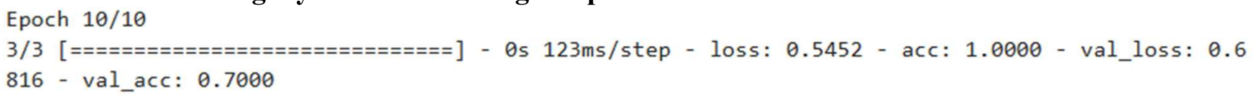
The GloVe model is a widely used pre-trained word embedding model trained on large text datasets. Performance testing was done for both approaches using various sizes of training samples: 100, 1,000, 5,000, and 10,000 samples.

- Custom-trained Embedding Layer: A custom embedding layer was trained on the IMDB dataset and tested its performance for various sample sizes.

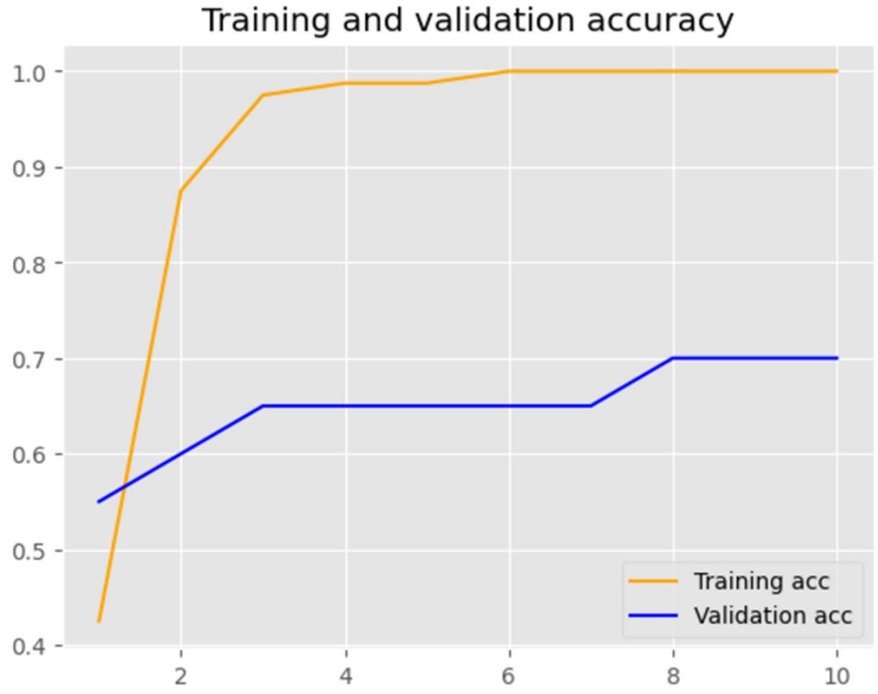
- Pre-trained Word Embedding Layer: A setup similar to the present, with accuracy compared to a custom-trained layer under the same conditions, was done using the GloVe model.

A. Custom-trained Embedding Layer

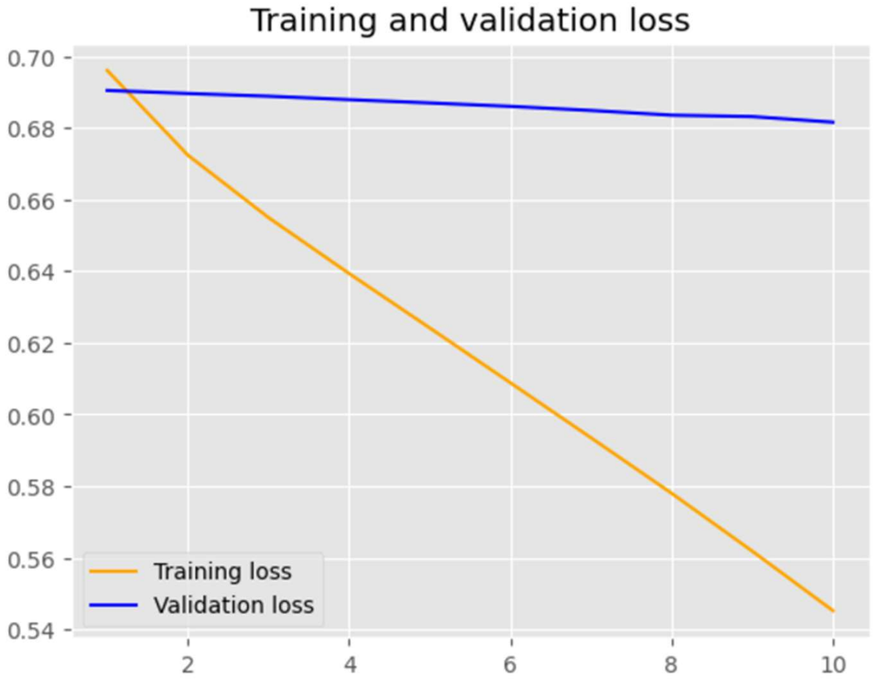
1) A custom embedding layer with a training sample size of 100.



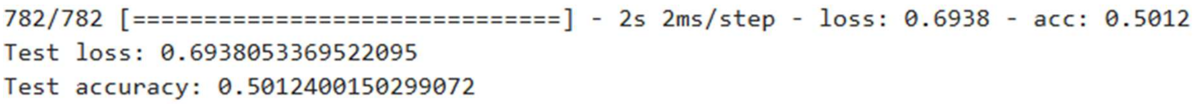
```
Epoch 10/10
3/3 [==============================] - 0s 123ms/step - loss: 0.5452 - acc: 1.0000 - val_loss: 0.6
816 - val_acc: 0.7000
```



Training and validation accuracy



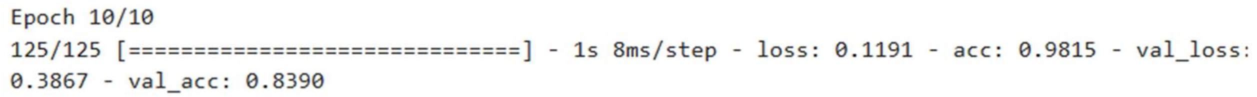
Training and validation loss



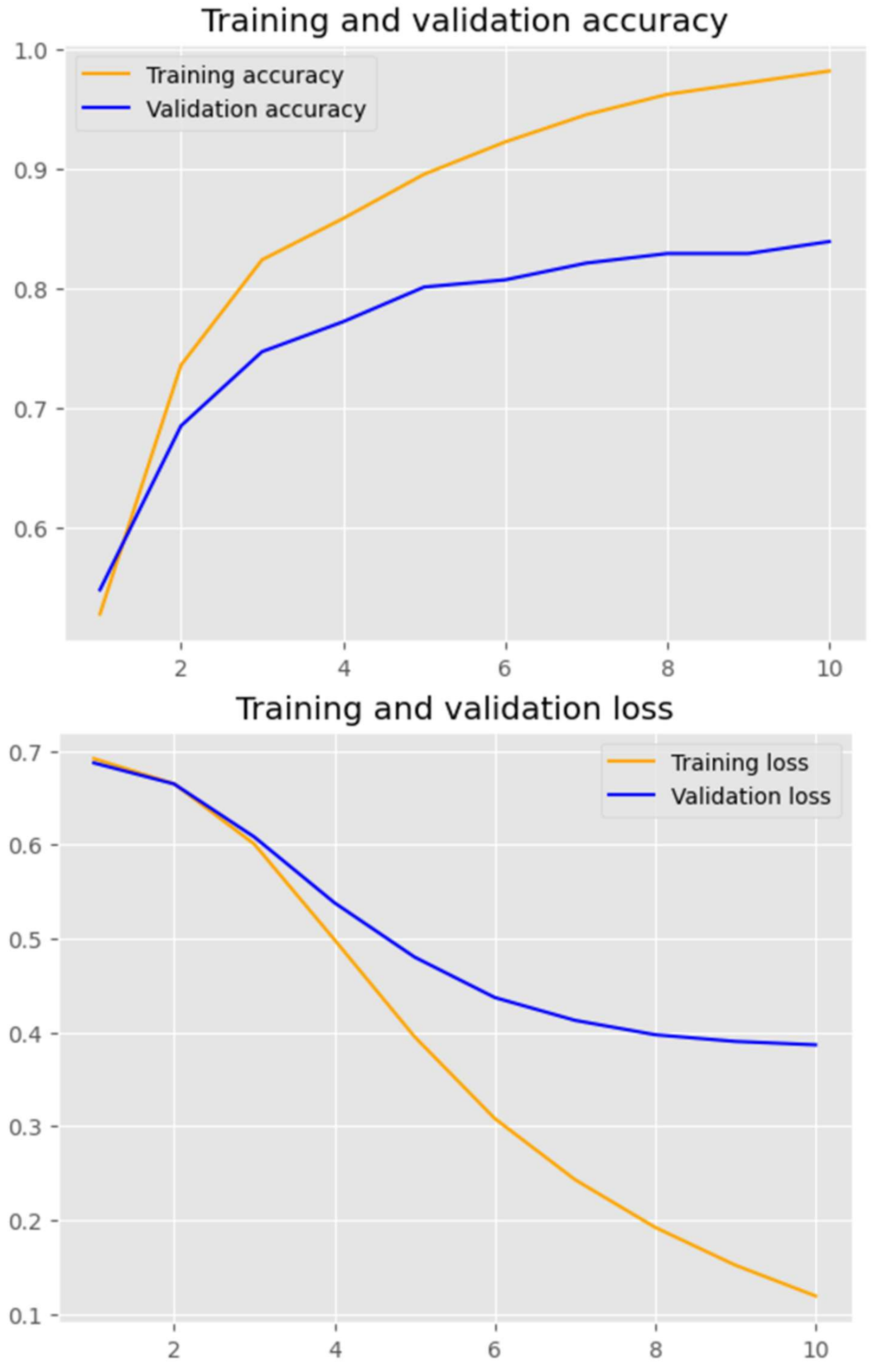
```
782/782 [==============================] - 2s 2ms/step - loss: 0.6938 - acc: 0.5012
Test loss: 0.6938053369522095
Test accuracy: 0.5012400150299072
```

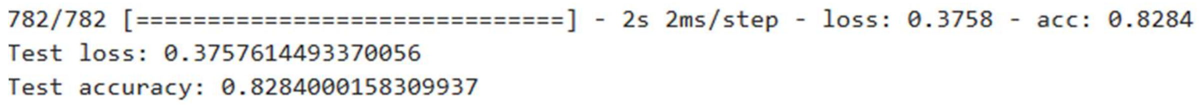
The model was trained using a custom-trained embedding layer with a sample size of 100, achieving a training accuracy of 100% and a test loss of 0.69.

2) A custom-trained embedding layer with a training sample size of 5000.



```
Epoch 10/10
125/125 [==============================] - 1s 8ms/step - loss: 0.1191 - acc: 0.9815 - val_loss:
0.3867 - val_acc: 0.8390
```

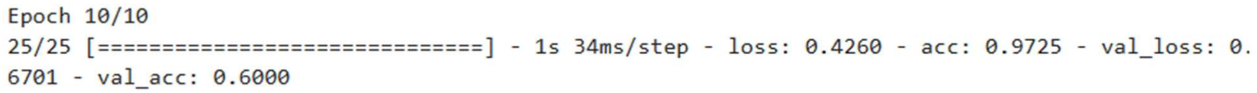




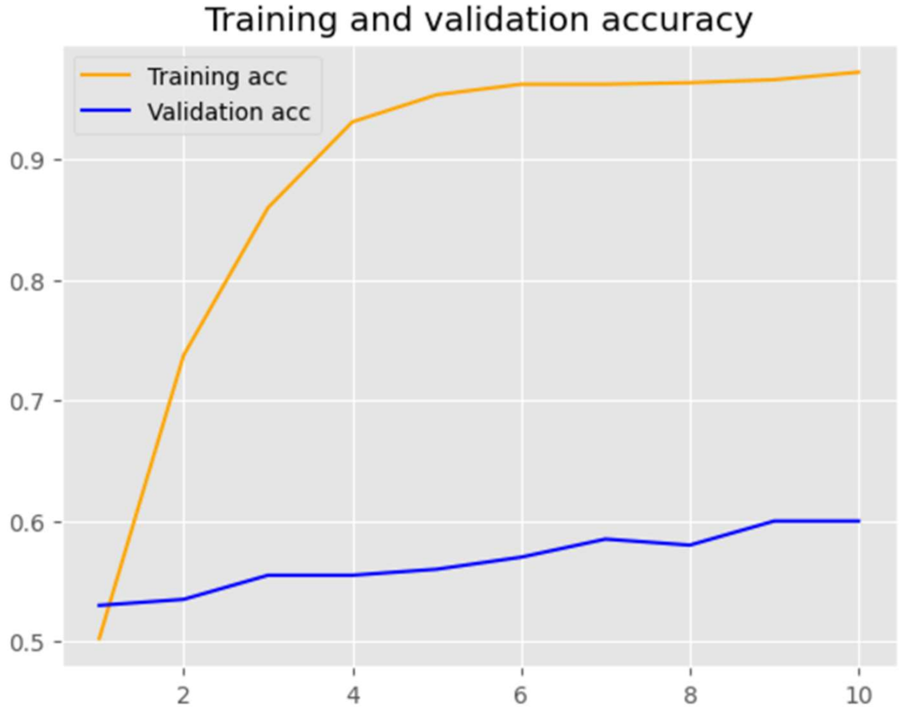
```
782/782 [==============================] - 2s 2ms/step - loss: 0.3758 - acc: 0.8284
Test loss: 0.3757614493370056
Test accuracy: 0.8284000158309937
```

Using a sample size of 5,000, the model achieved a training accuracy of 98.15% and a test loss of 0.37.

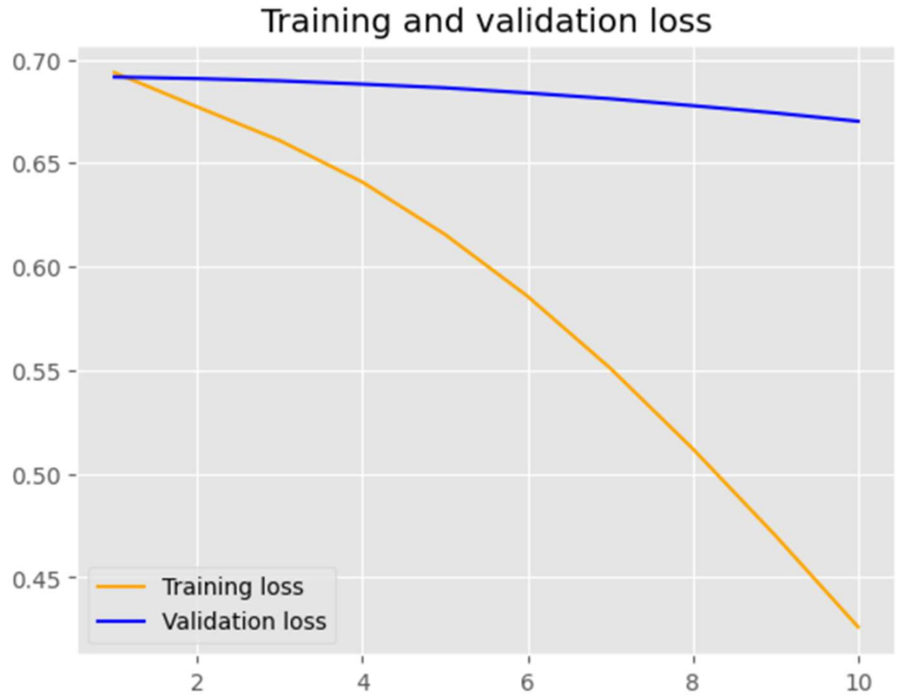
3) A custom-trained embedding layer with a training sample size of 1000



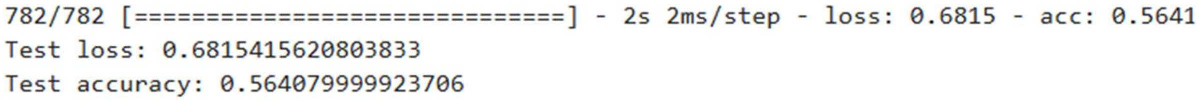
```
Epoch 10/10
25/25 [==============================] - 1s 34ms/step - loss: 0.4260 - acc: 0.9725 - val_loss: 0.
6701 - val_acc: 0.6000
```



Training and validation accuracy



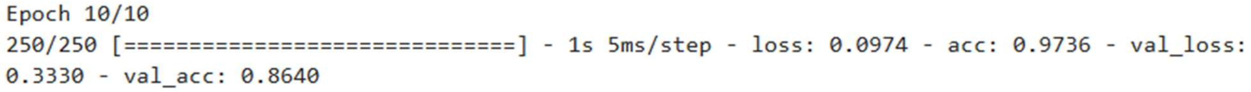
Training and validation loss



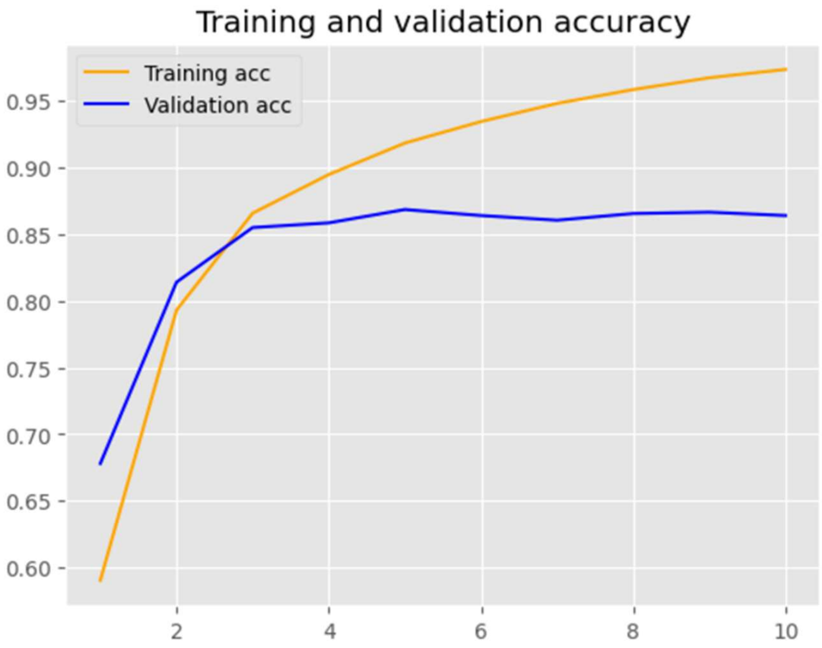
```
782/782 [==============================] - 2s 2ms/step - loss: 0.6815 - acc: 0.5641
Test loss: 0.6815415620803833
Test accuracy: 0.564079999923706
```

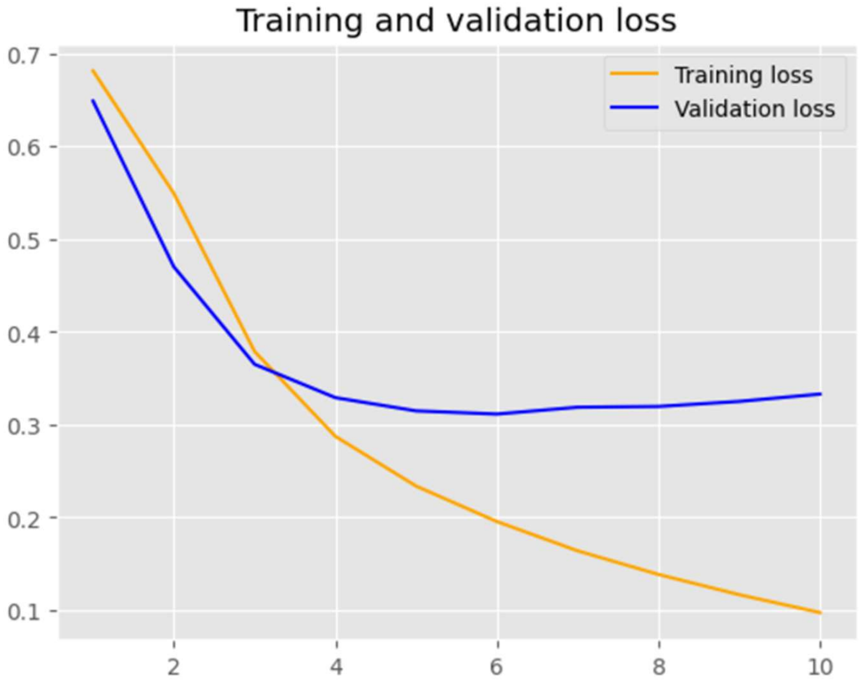
With a sample size of 1,000: Achieved 97.25% training accuracy and a test loss of 0.68.

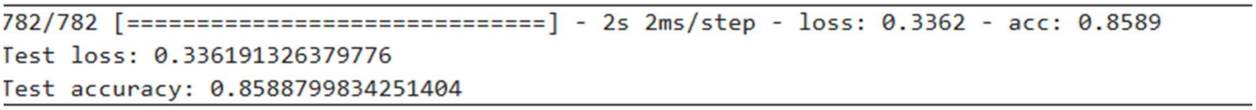
4) A custom-trained embedding layer with a training sample size of 10,000.



```
Epoch 10/10
250/250 [==============================] - 1s 5ms/step - loss: 0.0974 - acc: 0.9736 - val_loss:
0.3330 - val_acc: 0.8640
```







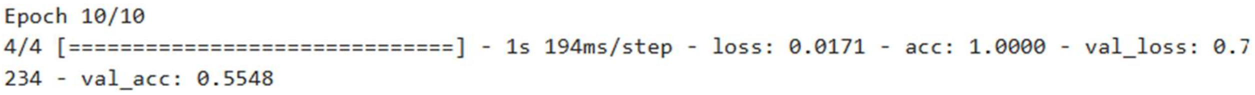
```
782/782 [==============================] - 2s 2ms/step - loss: 0.3362 - acc: 0.8589
Test loss: 0.336191326379776
Test accuracy: 0.8588799834251404
```

With a sample size of 10,000: Achieved 97.36% training accuracy and a test loss of 0.33.

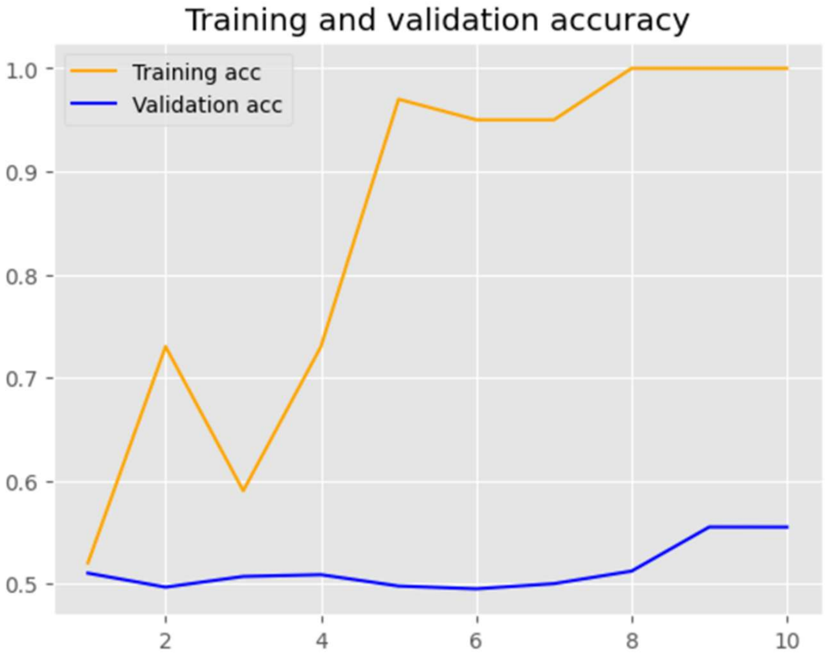
Result: With training sample sizes of 100, 1,000, 5,000, and 10,000, the model's accuracy ranged from 97.25% to 100%, with the highest accuracy of 100% achieved using a sample size of 100.

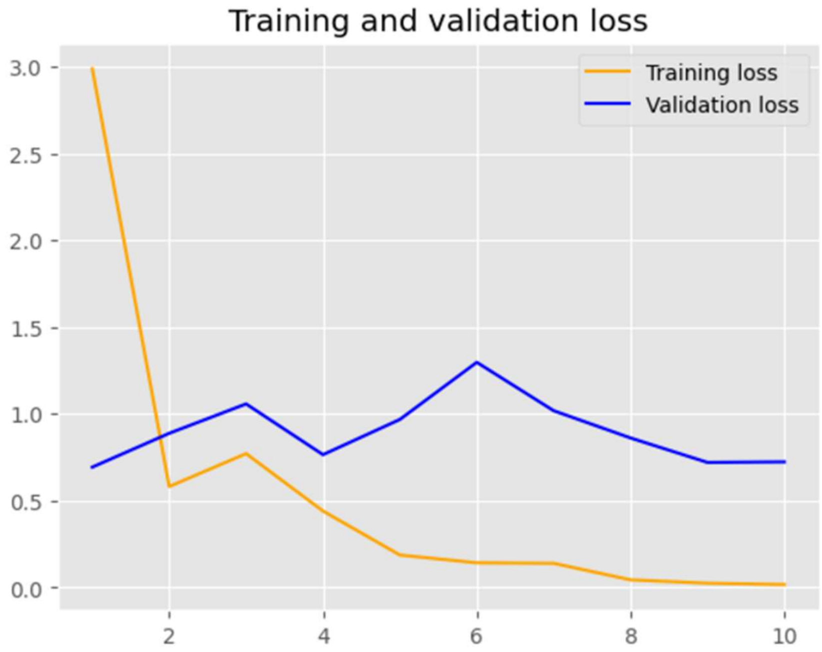
B) Pretrained Word Embedding Layer (GloVe)

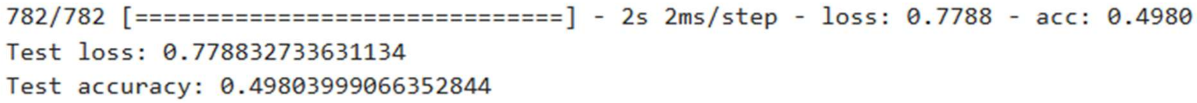
1) A pre-trained word embedding layer with a training sample size of 100.



```
Epoch 10/10
4/4 [==============================] - 1s 194ms/step - loss: 0.0171 - acc: 1.0000 - val_loss: 0.7
234 - val_acc: 0.5548
```



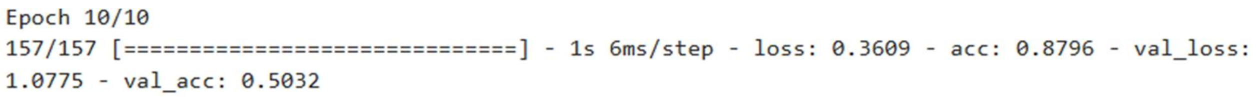




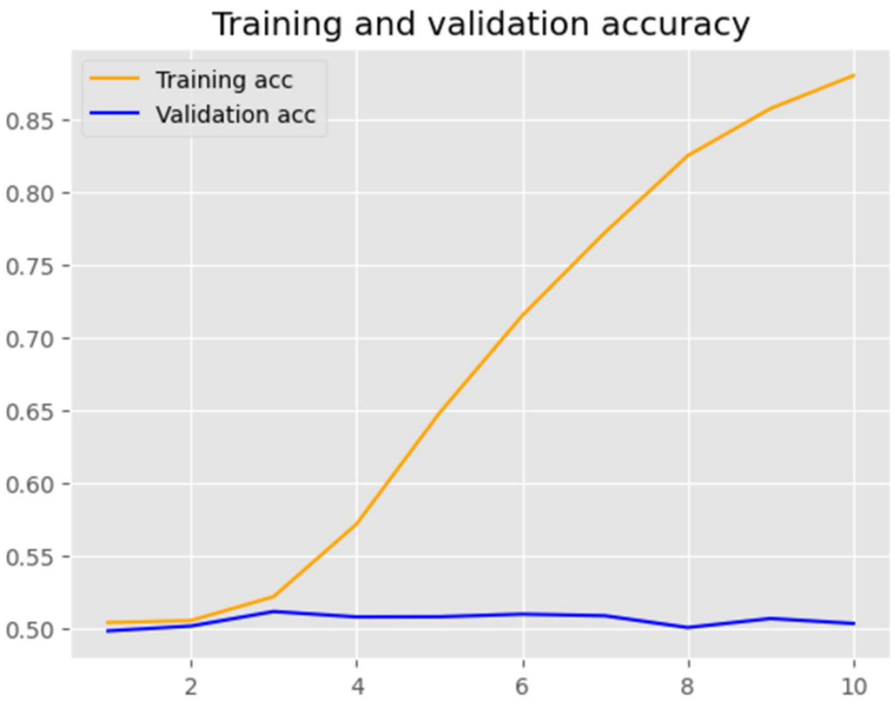
```
782/782 [==============================] - 2s 2ms/step - loss: 0.7788 - acc: 0.4980
Test loss: 0.778832733631134
Test accuracy: 0.49803999066352844
```

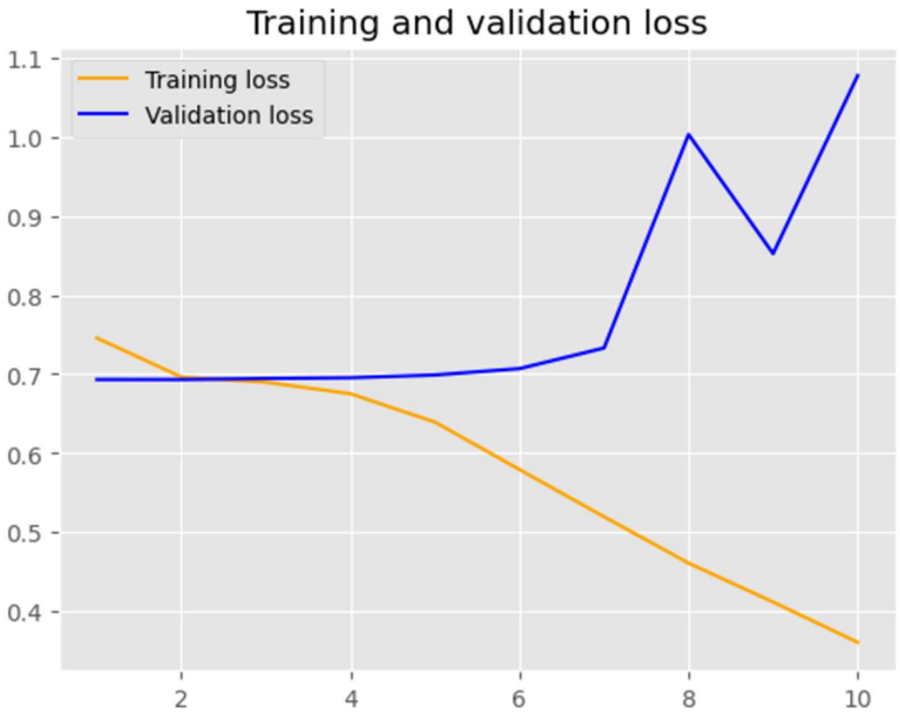
The model was trained using pre-trained word embedding (GloVe) with a sample size of 100, achieving a training accuracy of 100% and a test loss of 0.77.

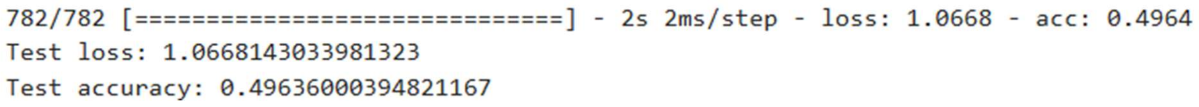
2) **A pre-trained word embedding layer with a training sample size of 5000.**



```
Epoch 10/10
157/157 [==============================] - 1s 6ms/step - loss: 0.3609 - acc: 0.8796 - val_loss:
1.0775 - val_acc: 0.5032
```



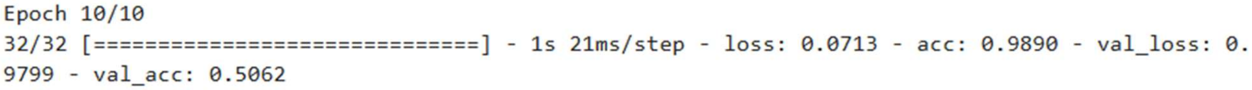




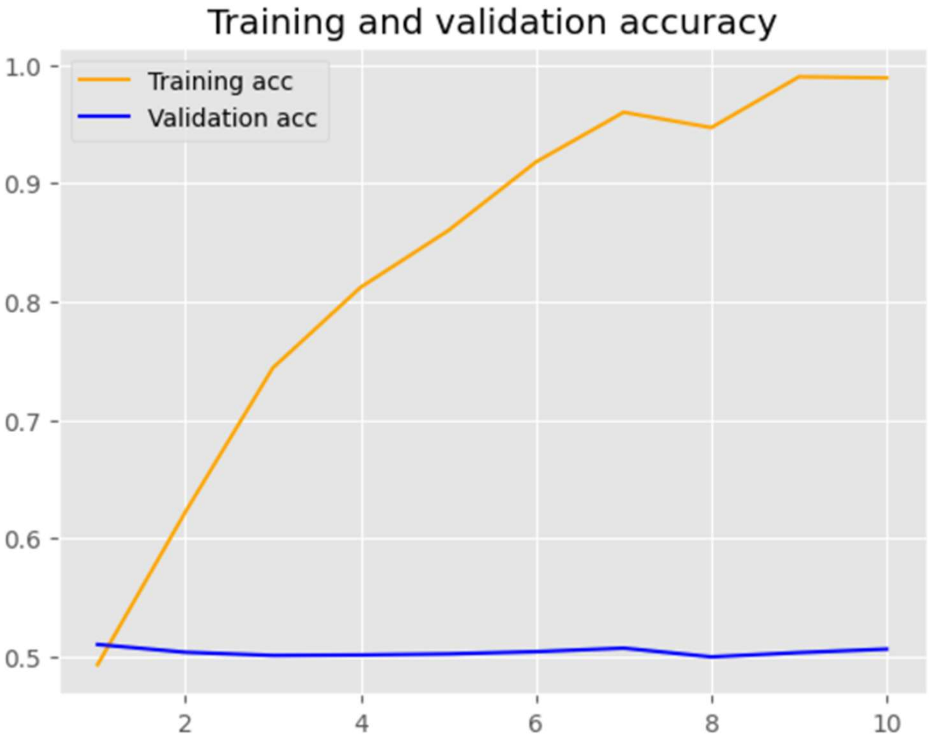
```
782/782 [==============================] - 2s 2ms/step - loss: 1.0668 - acc: 0.4964
Test loss: 1.0668143033981323
Test accuracy: 0.49636000394821167
```

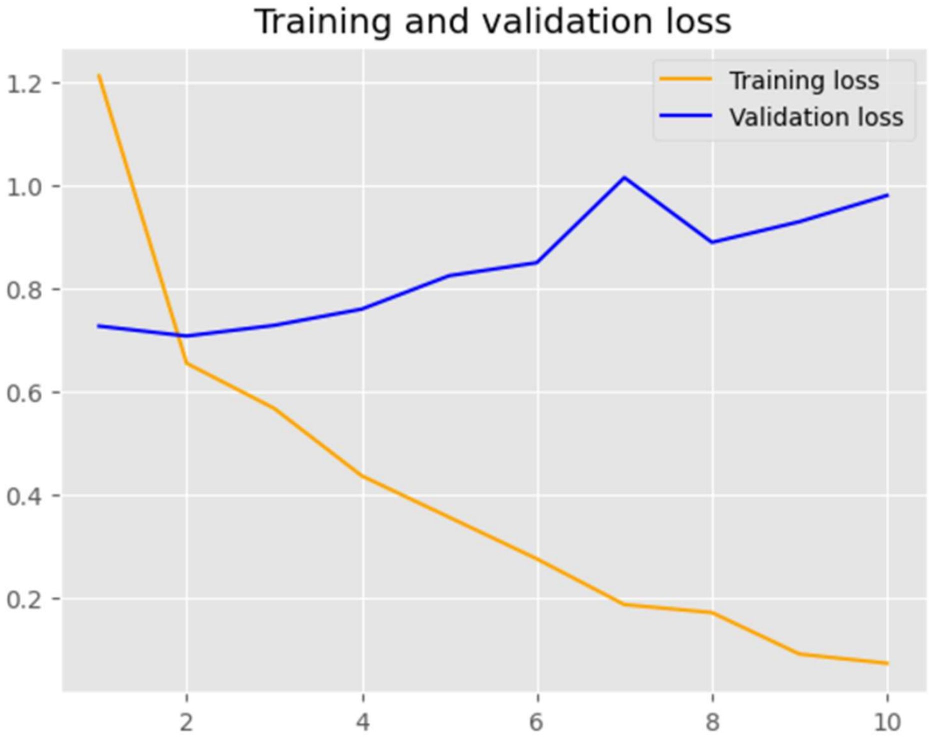
The model was trained using a sample size of 5,000, achieving a training accuracy of 87.96% and a test loss of 1.06.

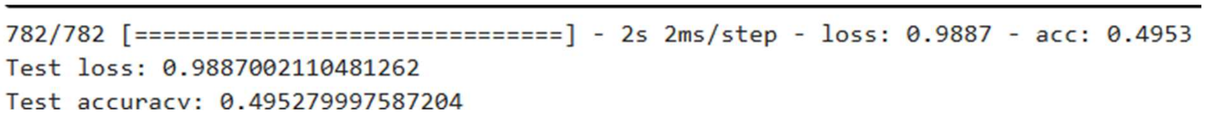
3) **A pre-trained word embedding layer with a training sample size of 1000.**



```
Epoch 10/10
32/32 [==============================] - 1s 21ms/step - loss: 0.0713 - acc: 0.9890 - val_loss: 0.
9799 - val_acc: 0.5062
```



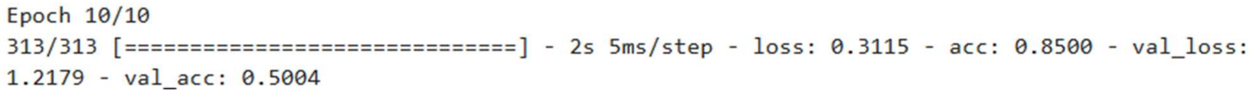




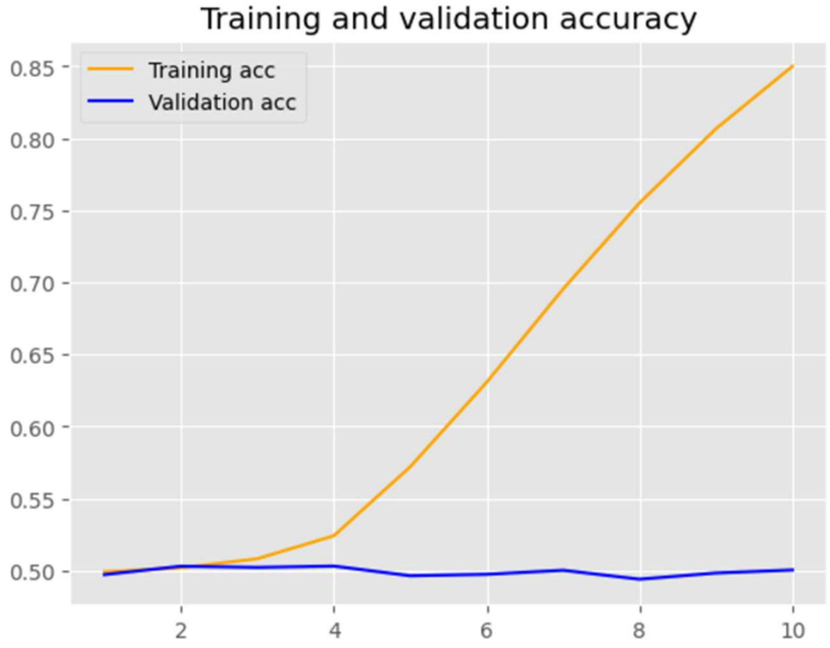
```
782/782 [==============================] - 2s 2ms/step - loss: 0.9887 - acc: 0.4953
Test loss: 0.9887002110481262
Test accuracy: 0.495279997587204
```

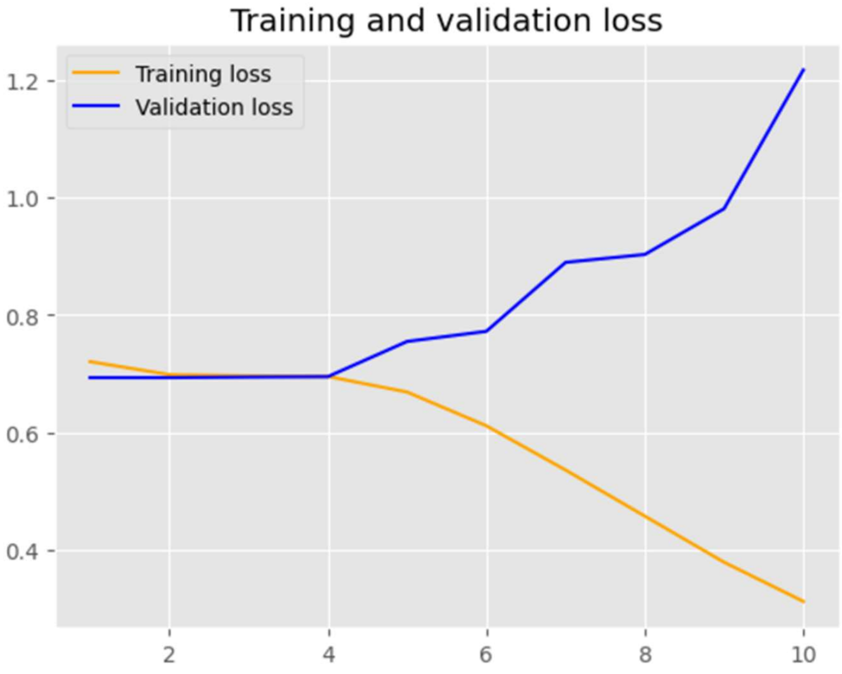
With a sample size of 1,000: Achieved 98.90% training accuracy and a test loss of 0.98.

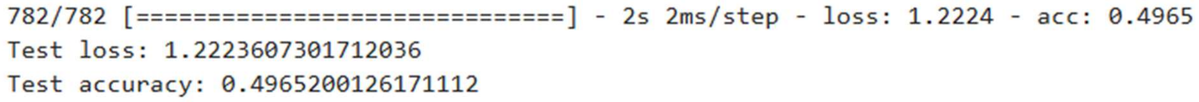
4) A pre-trained word embedding layer with a training sample size of 10,000.



```
Epoch 10/10
313/313 [==============================] - 2s 5ms/step - loss: 0.3115 - acc: 0.8500 - val_loss:
1.2179 - val_acc: 0.5004
```







```
782/782 [==============================] - 2s 2ms/step - loss: 1.2224 - acc: 0.4965
Test loss: 1.2223607301712036
Test accuracy: 0.4965200126171112
```

With a sample size of 10,000: Achieved 85% training accuracy and a test loss of 1.22.

Result: The model's accuracy ranged from 85% to 100%, depending on the sample size. While the highest accuracy of 100% was achieved with 100 samples, the model showed a tendency to overfit as the sample size increased, resulting in a decline in accuracy.

Conclusion

The analysis shows that custom-trained embedding layers are always more accurate and have lower test loss compared to pre-trained GloVe embeddings, especially as the training dataset size increases. Custom embeddings follow a smooth trend in performance improvement as dataset sizes grow larger, culminating in very high accuracy and low test loss. In contrast, pre-trained GloVe embeddings are more apt to be overfitted on small datasets and generalize poorly with larger datasets, leading to higher test losses and lower accuracy.

While custom embeddings work better on larger datasets, pre-trained embeddings are a valid choice when computation resources or training data are limited. However, the risk of overfitting increases significantly in such cases. As such, the choice of embedding methods presents a balancing act: while more flexible models perform better, they do so at great computational cost.