Assignment 4
Summary

DineshKumar Yechuri
811295667

Objective:

The binary classification problem of the IMDB dataset aims to classify movie reviews into positive and negative categories. The dataset comprises 50,000 reviews; the top 10,000 terms are assessed; training samples are restricted to 100, 5000, 1000, and 100,000; validation is carried out on 10,000 samples. The data have been prepared. After then, the input is fed into a pretrained embedding model and the embedding layer, and various strategies are tried to gauge performance.

Data Preprocessing:

- As part of the dataset preparation process, each review is transformed into a set of word embeddings, where each word is represented by a fixed-size vector. The 10,000 sample limit is applicable. In addition, rather than using a string of words, a set of integers representing individual words was generated from the reviews. Even if I have the list of numbers, the input of the neural network is inappropriate for it.

- Tensors need to be constructed using the numbers. One might create a tensor with integer data type and form (samples, word indices) using the integer list. In order for me to do that, I have to make sure that every sample is the same length, which means I have to use dummy words or numbers to ensure that every review is the same length.

Procedure:

For this IMDB dataset, I looked into two distinct methods for creating word embeddings:

1. Custom-trained embedding layer
2. pre-trained word embedding layer using the GloVe model.

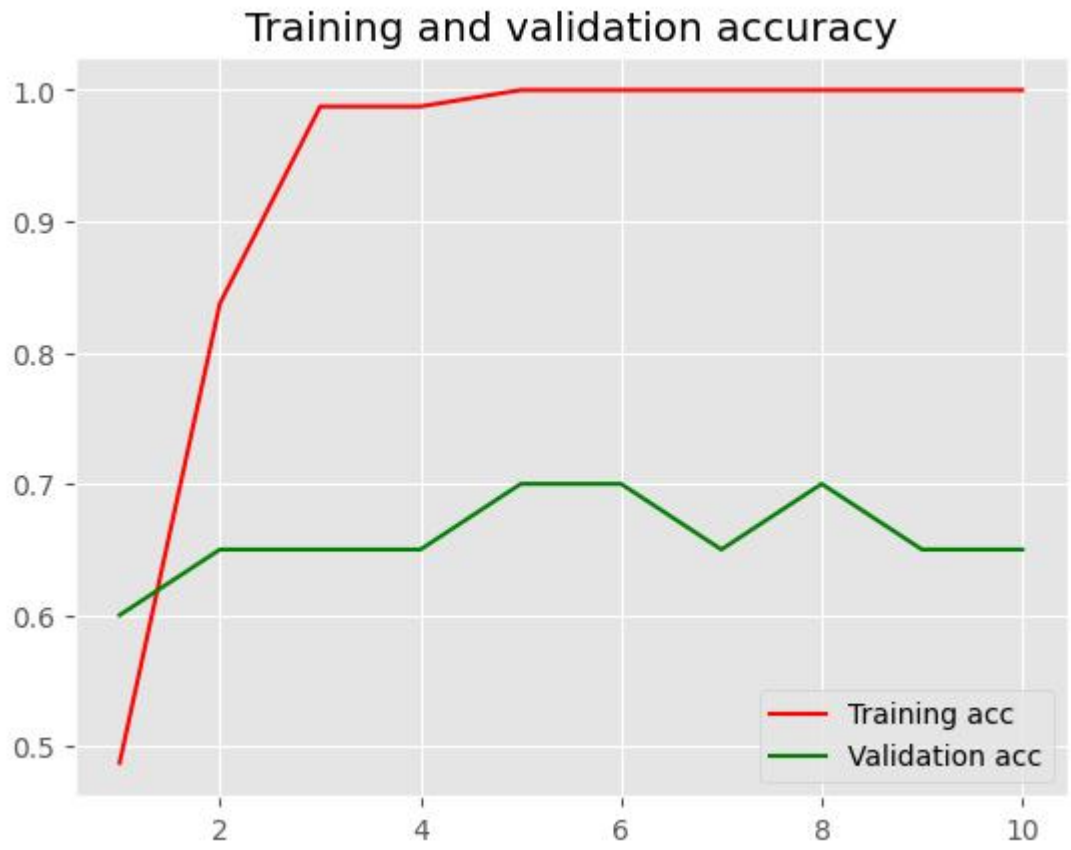
In our experiment, we employed the popular pretrained word embedding model GloVe, which is trained on copious amounts of textual data.

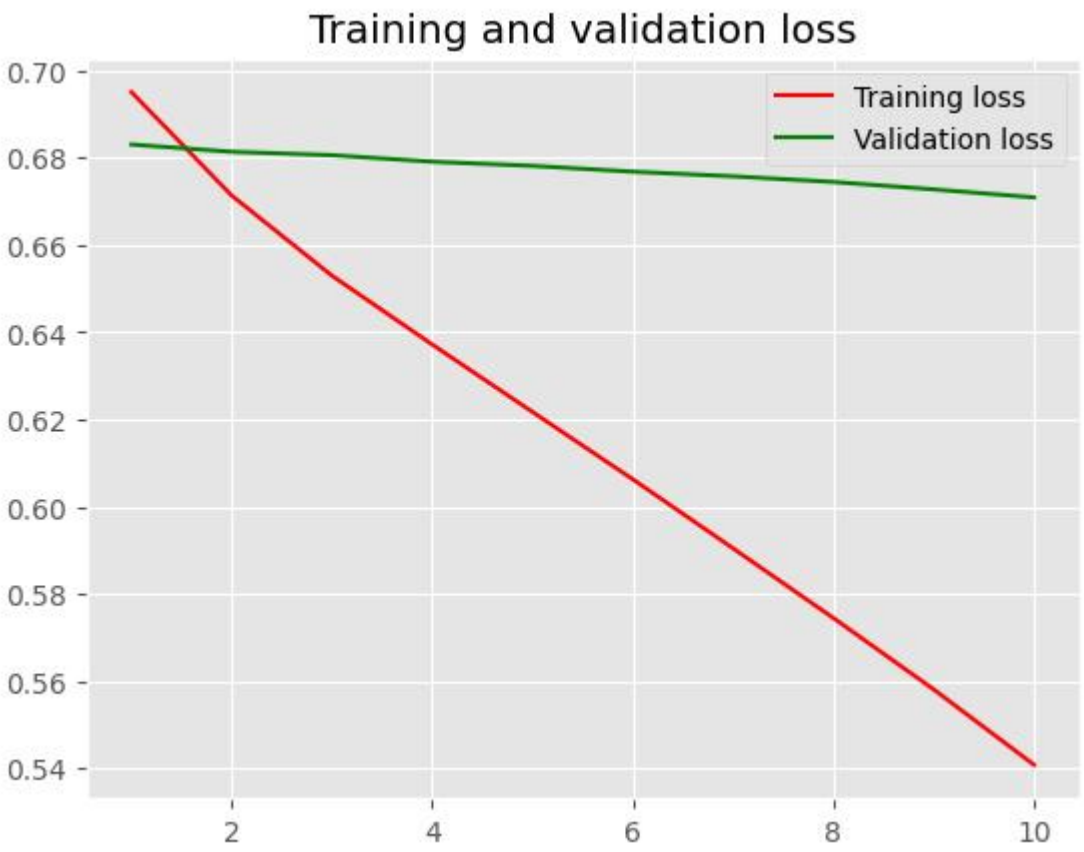
- To evaluate the effectiveness of various embedding tactics, I employed two distinct embedding layers: one with a custom-trained layer and the other with a pre-trained word embedding layer, using the IMDB review dataset. I evaluated the two models' accuracy using training sample sizes of 100, 5000, 1000, and 10,000.

- Using the IMDB review dataset, we started by building a specially-trained embedding layer. After training each model on many dataset samples, we used a testing set to determine its accuracy. Next, we compared these precisions to a model that had a pre-trained word embedding layer and had already been evaluated on various sample sizes.

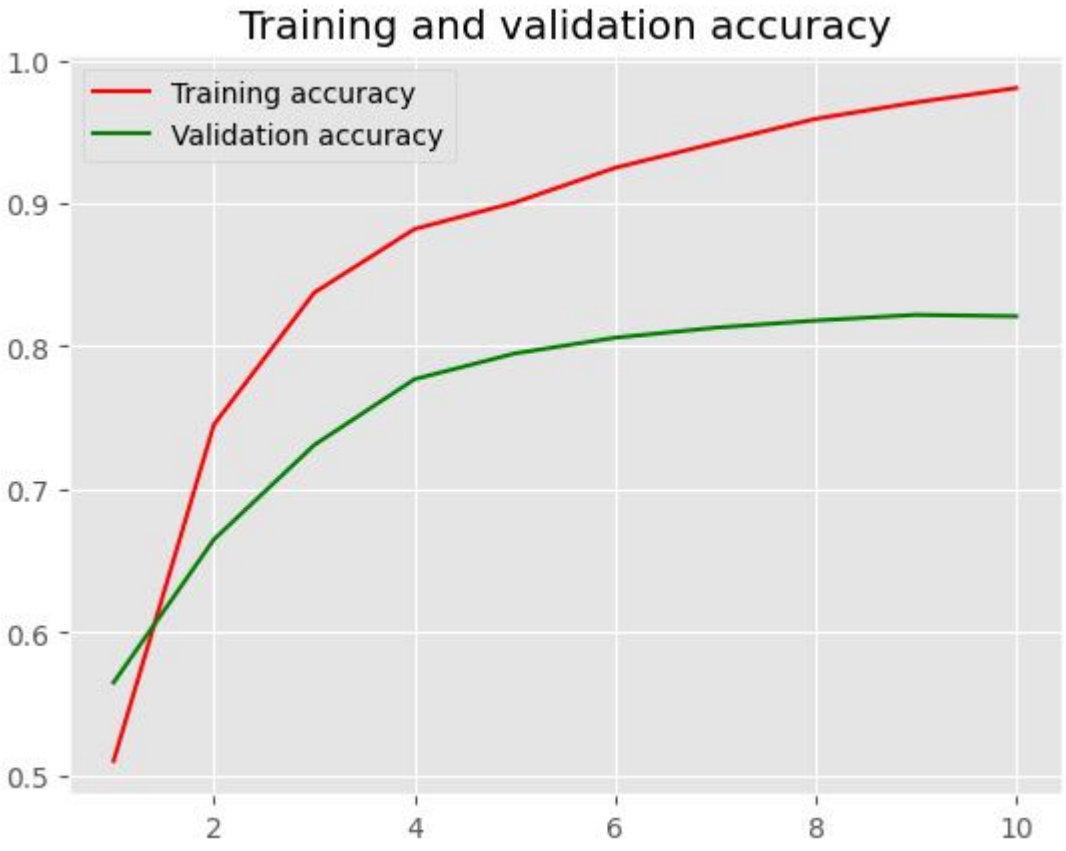
CUSTOM-TRAINED EMBEDDING LAYER

1. Custom-trained embedding layer with training sample size = 100

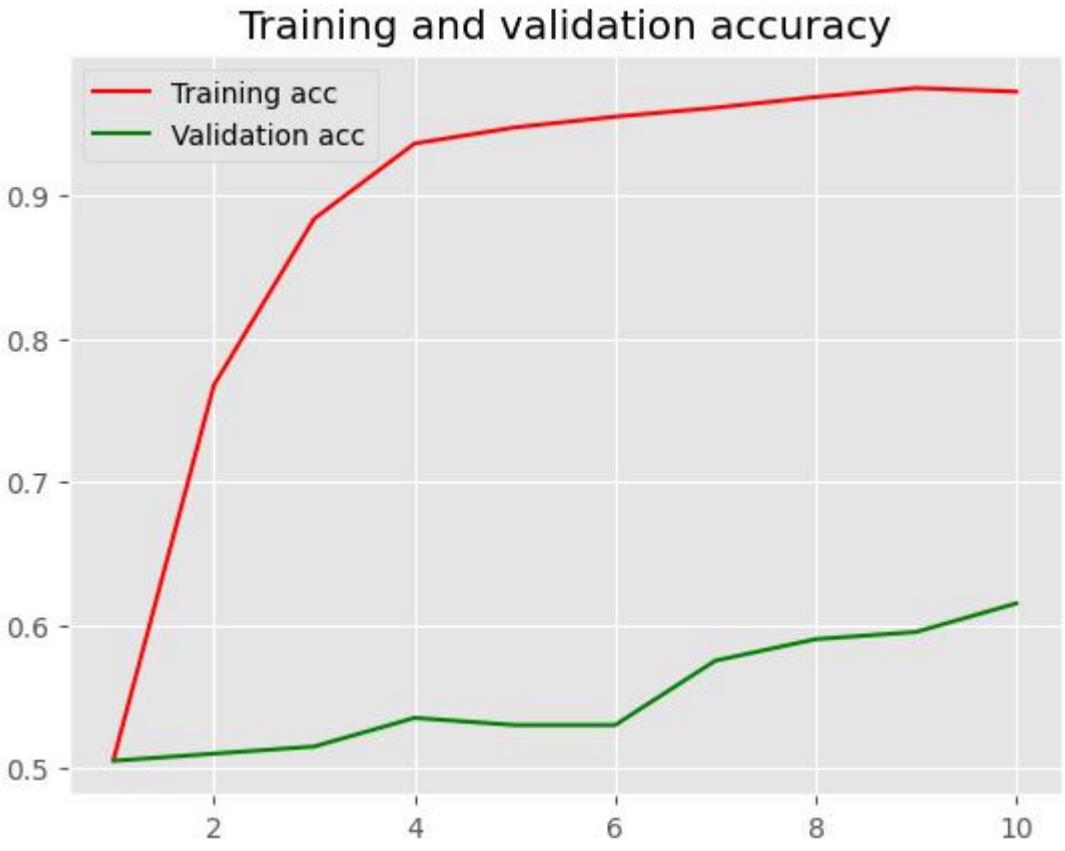


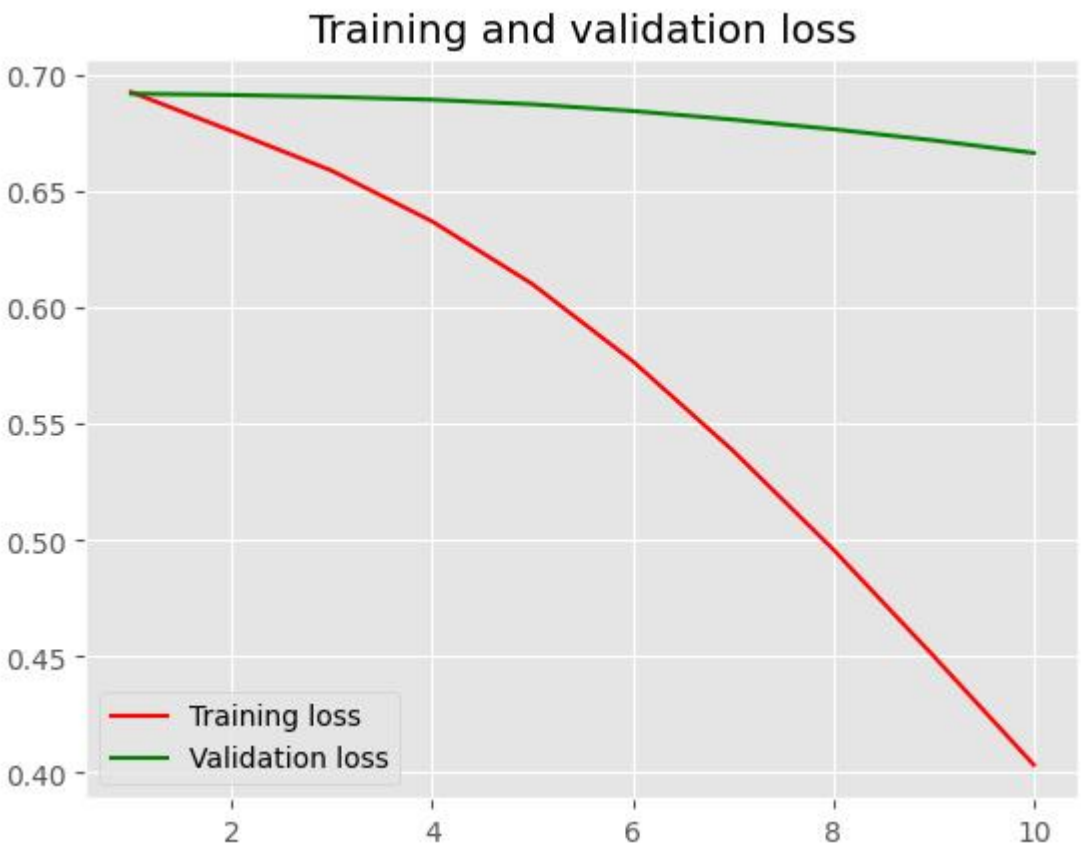


2. Custom-trained embedding layer with training sample size = 5000

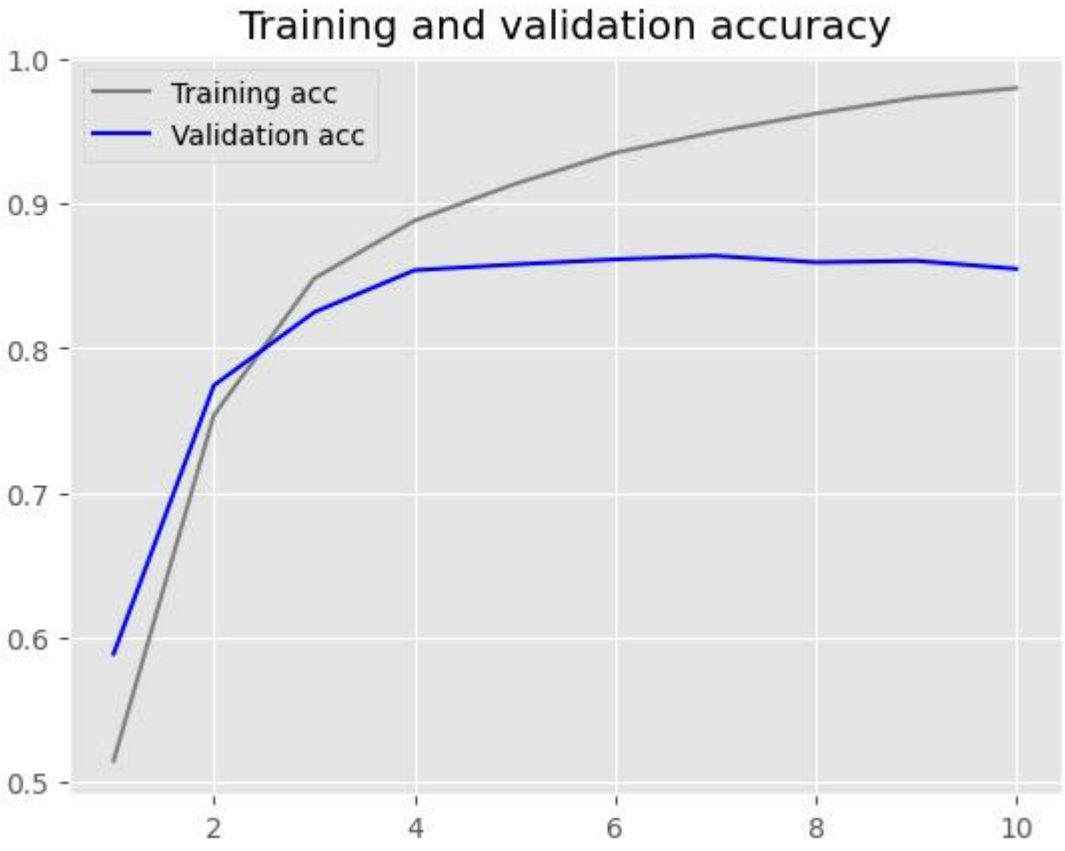


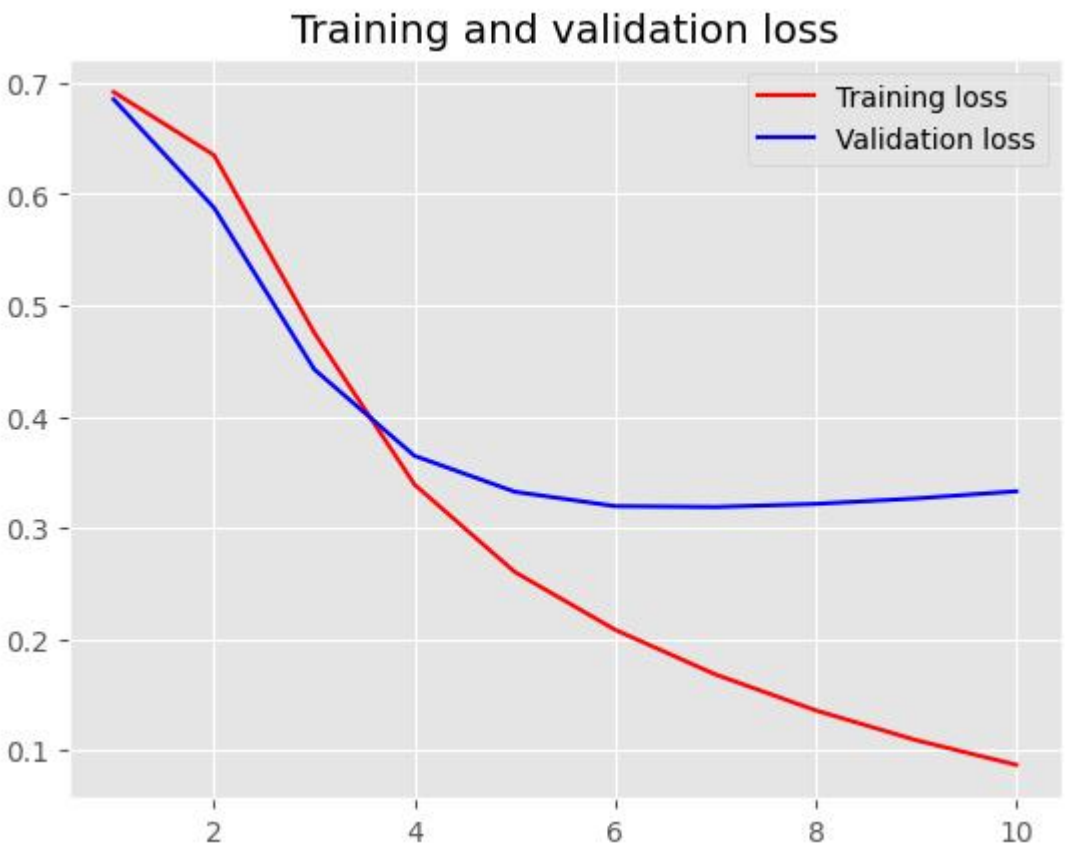
3. Custom-trained embedding layer with training sample size = 1000





4. Custom-trained embedding layer with training sample size = 10000

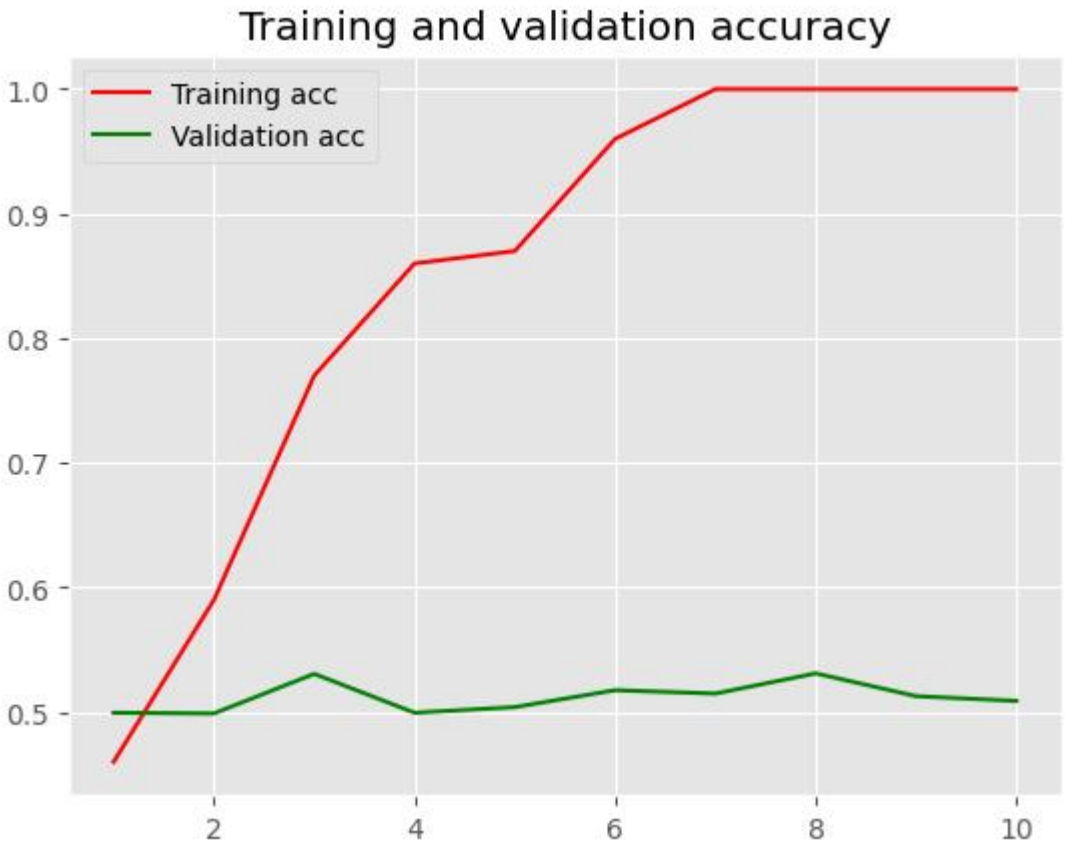


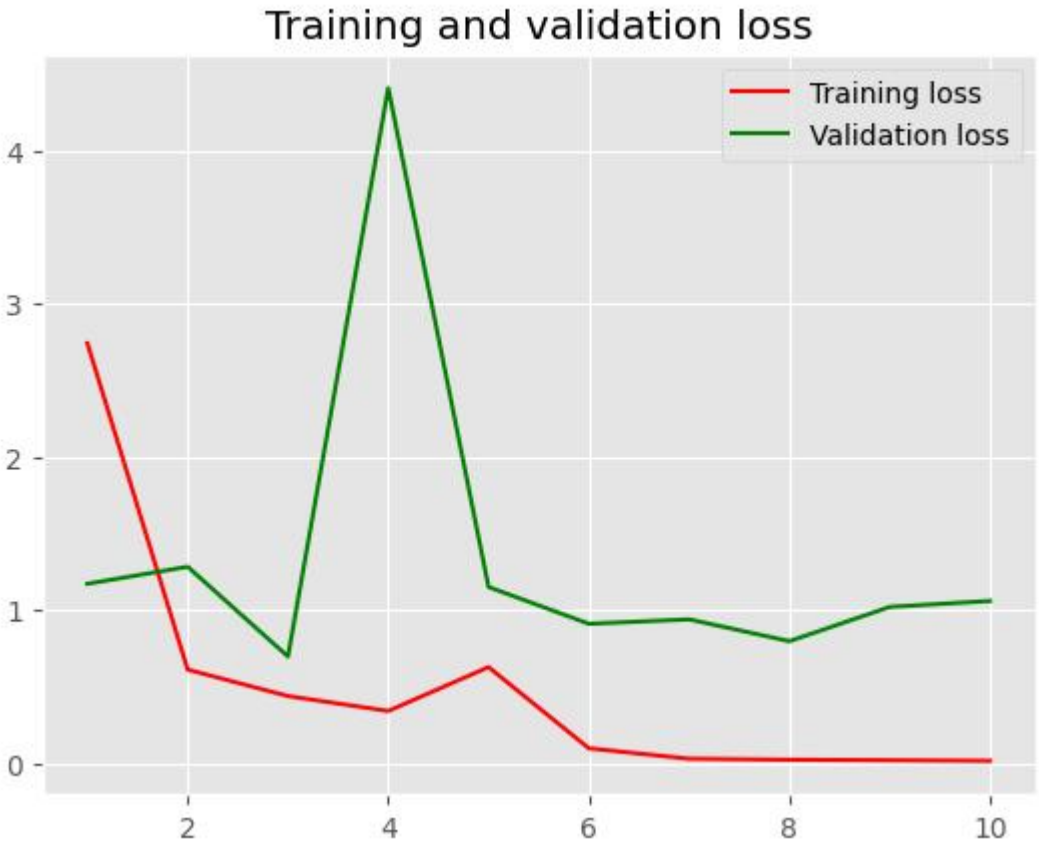


Depending on the size of the training sample, the accuracy of the custom-trained embedding layer varied from 97.25% to 100%. The training sample size of 100 produced the best accuracy.

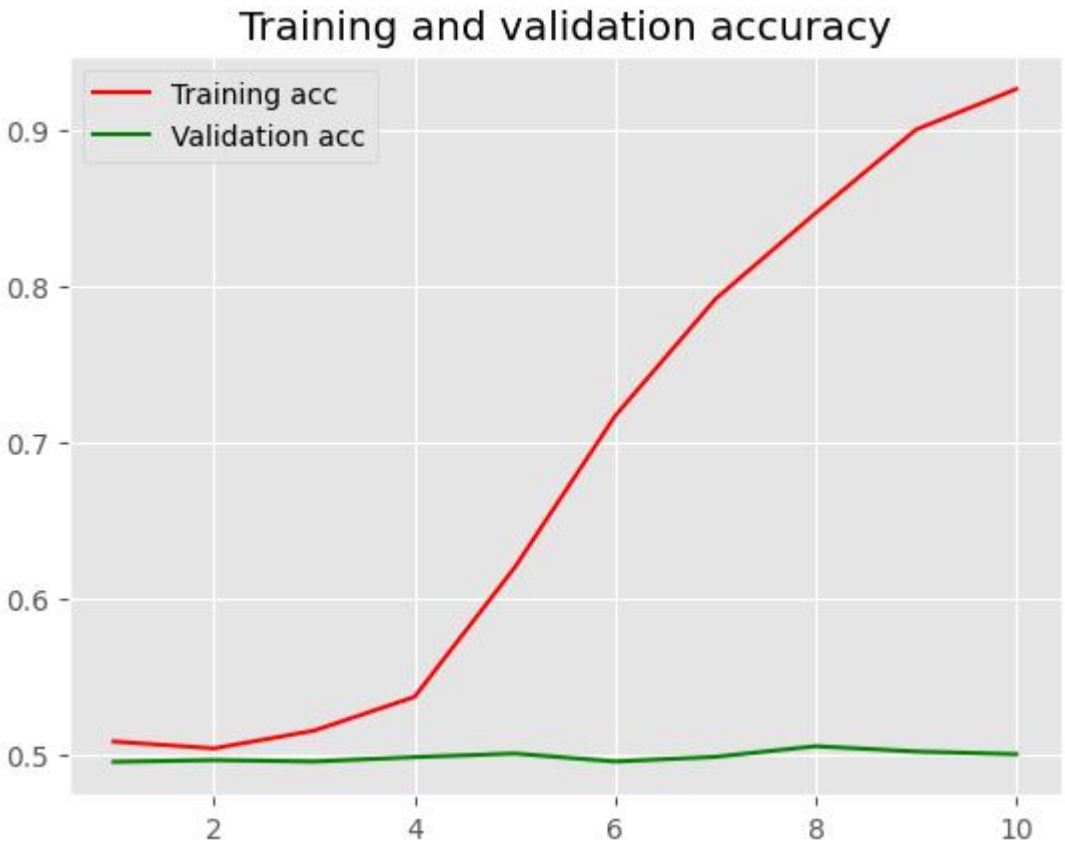
PRETRAINED WORD EMBEDDING LAYER

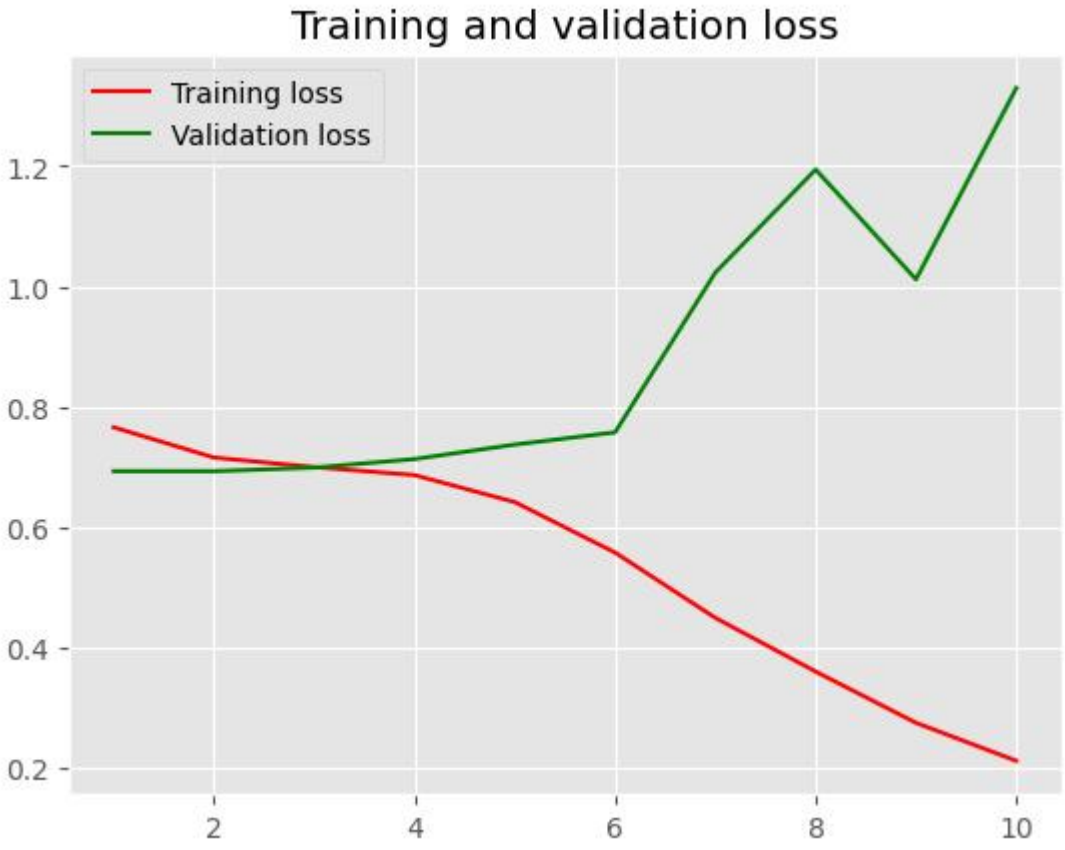
1. pretrained word embedding layer with training sample size = 100



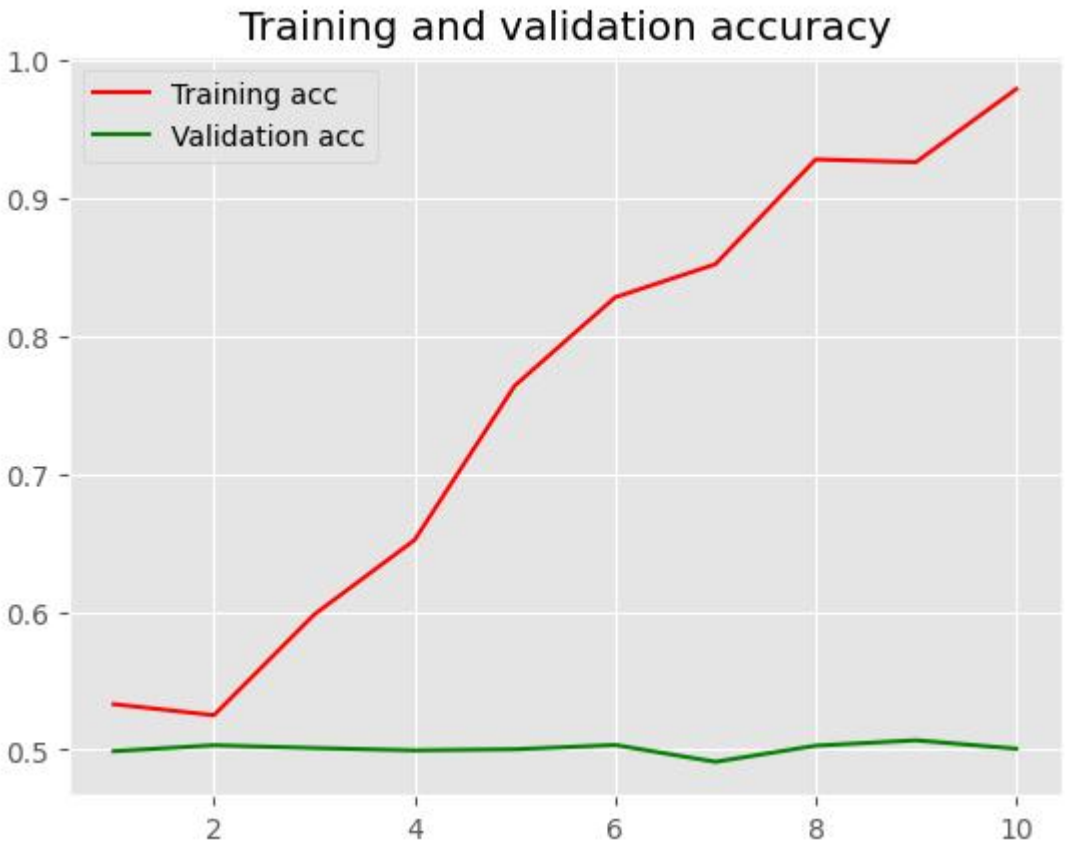


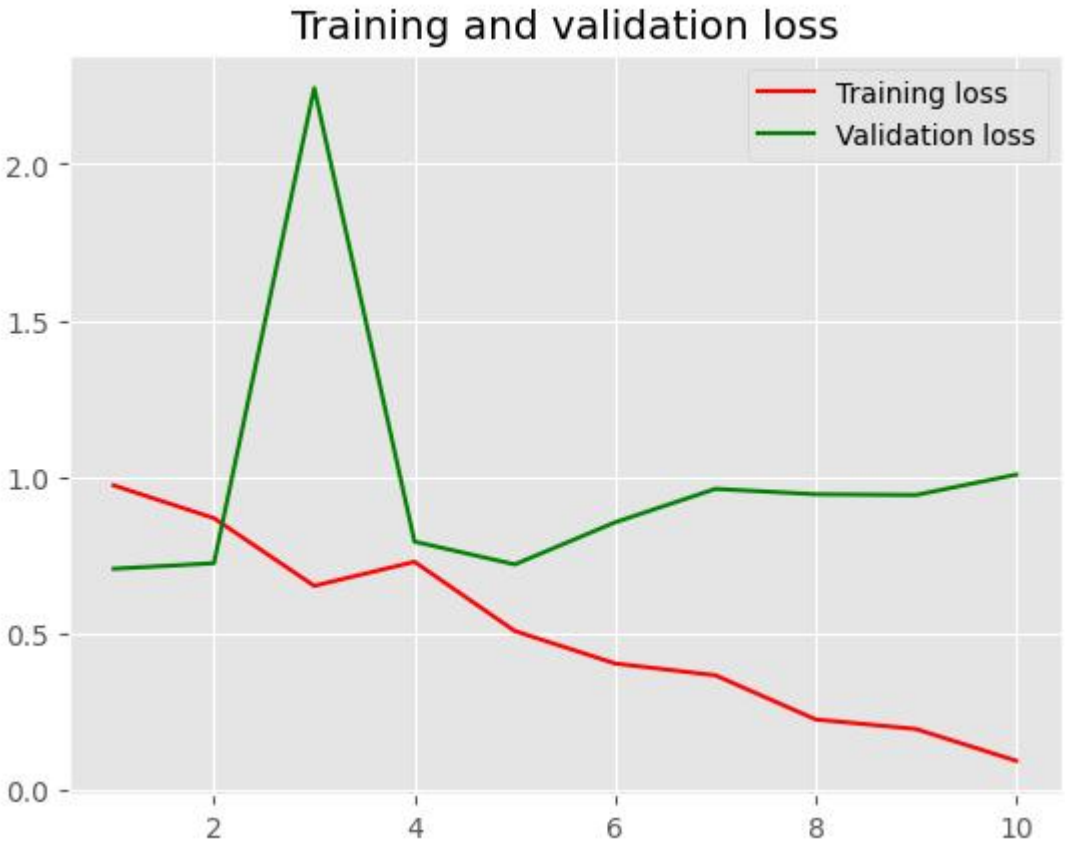
2. pretrained word embedding layer with training sample size = 5000



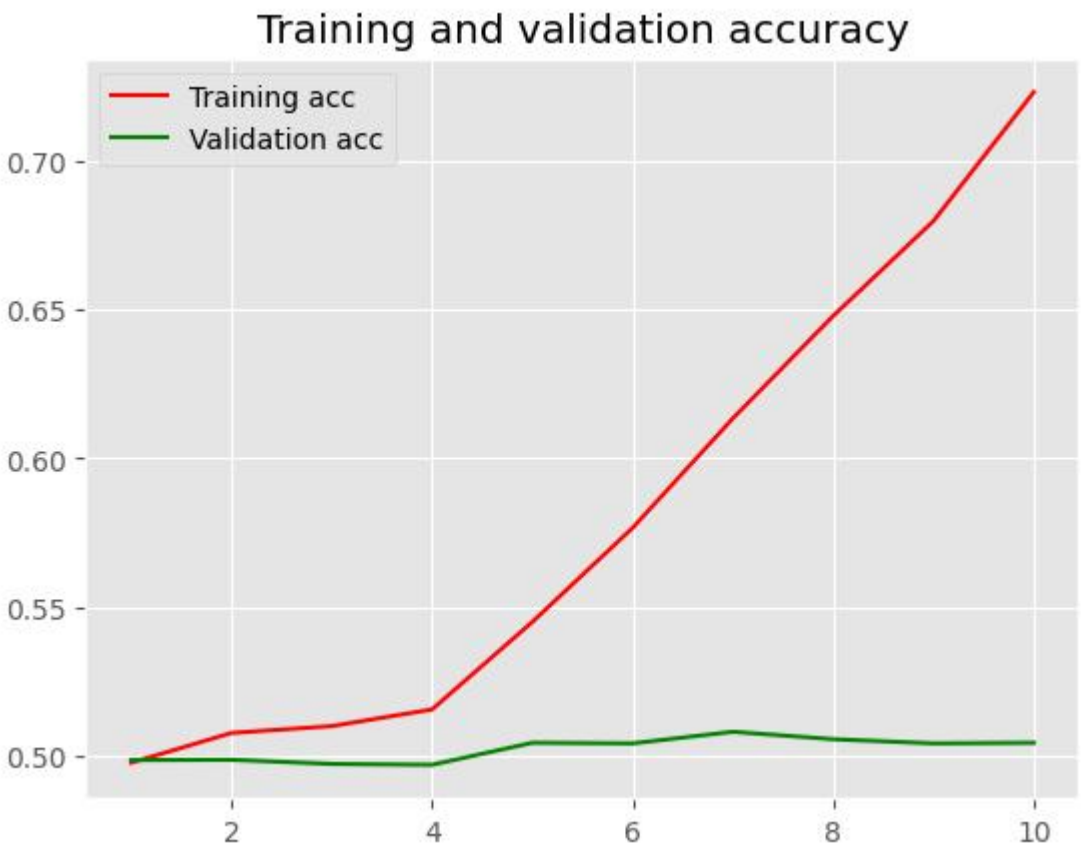


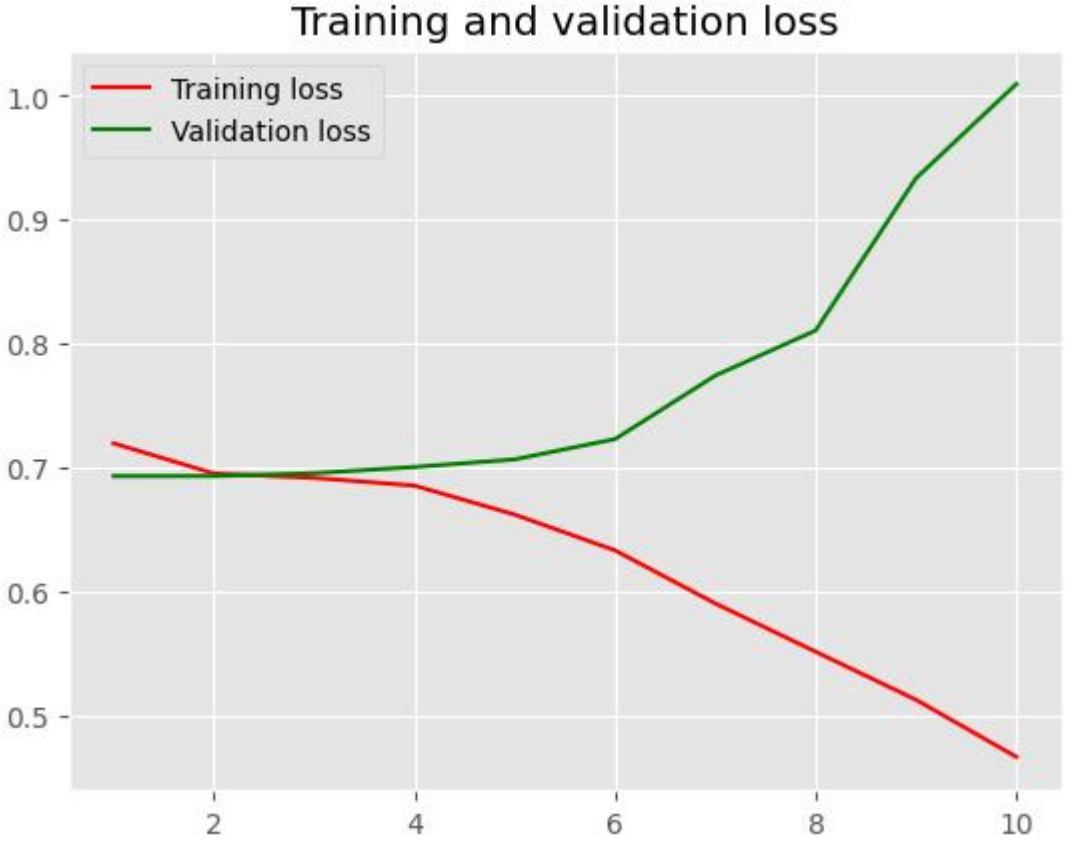
3. pretrained word embedding layer with training sample size = 1000





4. pretrained word embedding layer with training sample size = 10000


Training and validation accuracy
Training acc
Validation acc
0.70
0.65
0.60
0.55
0.50
2
4
6
8
10


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

The accuracy of the pretrained word embedding layer (GloVe) varied according on the size of the training sample, ranging from 72.31% to 100%. With 100 training samples, the most accurate result was achieved. Furthermore, employing the pretrained embeddings with larger training sample sizes causes the model to rapidly overfit, which lowers accuracy. These results make it challenging to decide which strategy is the "best" to utilize with confidence because it depends on the requirements and limitations of the task at hand.

Results:

Embedding Technique	**Training Sample Size**	**Training Accuracy (%)**	**Test loss**
Custom-trained embedding layer	100	100	0.69
Custom-trained embedding layer	5000	98.05	0.37
Custom-trained embedding layer	1000	97.25	0.66
Custom-trained embedding layer	10000	97.99	0.33
Pretrained word embedding (GloVe)	100	100	1.15
Pretrained word embedding (GloVe)	5000	92.62	1.30
Pretrained word embedding (GloVe)	1000	97.90	1
Pretrained word embedding (GloVe)	10000	72.31	0.99

Conclusion:

In this experiment, however, the custom-trained embedding layer performed better than the pretrained word embedding layer, especially when training with more training sample numbers. If computational resources are restricted and a modest training sample size is required, the pretrained word embedding layer might be a "better choice" despite the risk of overfitting.