RAHUL KADARU

811318179

Introduction: The goal of this research is to develop a model that can distinguish between good and negative movie reviews using the IMDB dataset. The top 10,000 most frequently occurring words are the only subset of the data that is the subject of the analysis. Different sample sizes (100, 500, 1000, and 100,000) will be used for training.

Key Challenge The core question is: which word embedding method yields superior performance in sentiment classification?

Data and pre Preprocessing:

- The IMDB movie review dataset with sentiment labels (positive or negative) is used in the analysis.
- Preprocessing entails limiting vocabulary and transforming reviews into word embeddings.
- Reviews are converted into integer sequences, where each integer corresponds to a distinct word, up to the top 10,000 words.

Technique: Two word embedding methods are compared

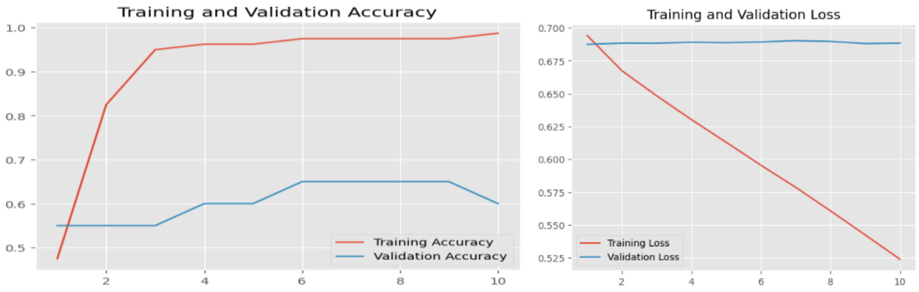
Custom-trained Embedding Layer: Particularly, the IMDB review data is used to train a different embedding layer.

Pre-trained Word Embedding Layer (GloVe):

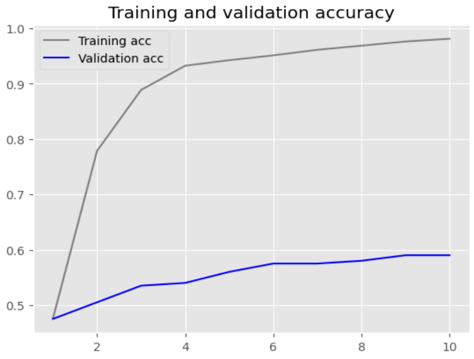
- To train word embeddings and capture syntactic and semantic links between words, this well-liked model uses a sizable corpus of text data (Wikipedia and Gigaword 5).
- 400,000 words and 6 billion tokens are used in the 6B version.

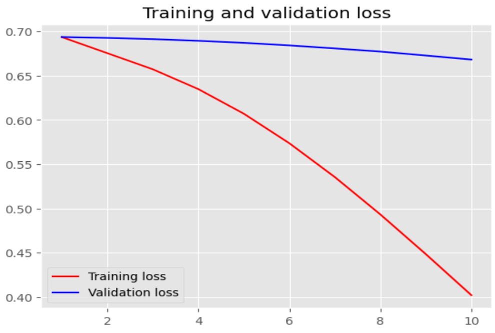
CUSTOM-TRAINED EMBEDDING LAYER

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



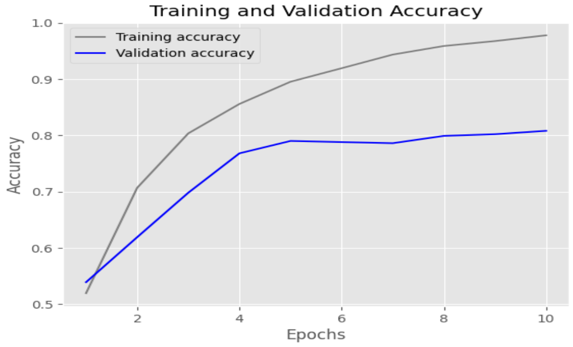
2.Custom-trained embedding layer with training sample size = 1000

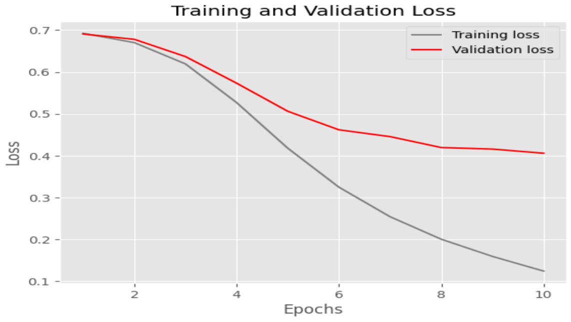




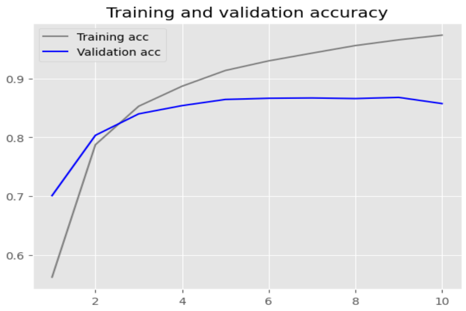
For Custom-trained embedding layer with training sample size = 1000 , The test accuracy and Test loss is 0.57 and 0.67

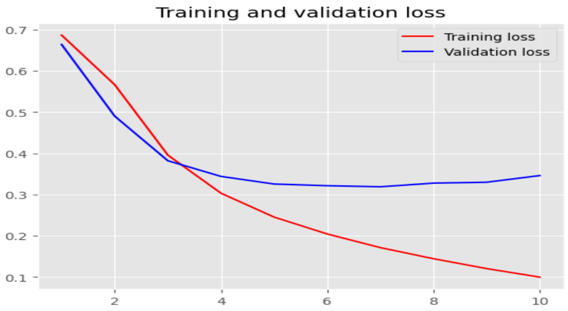
3 . Custom-trained embedding layer with training sample size = 5000





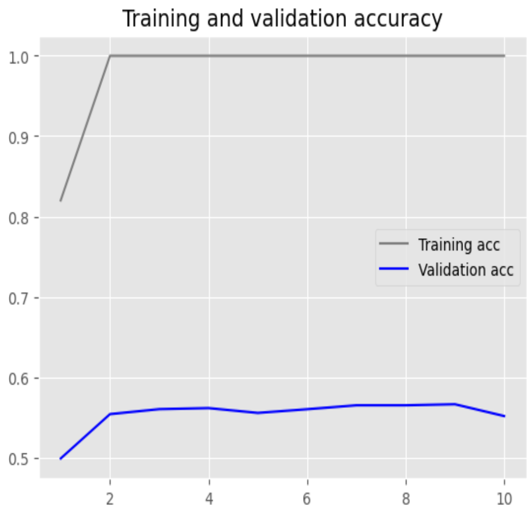
4. Custom-trained embedding layer with training sample size = 10000

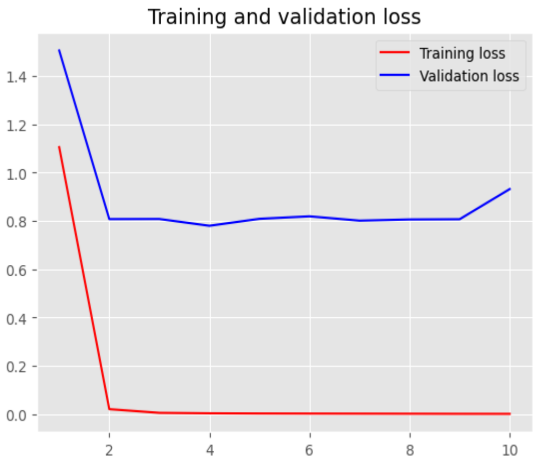




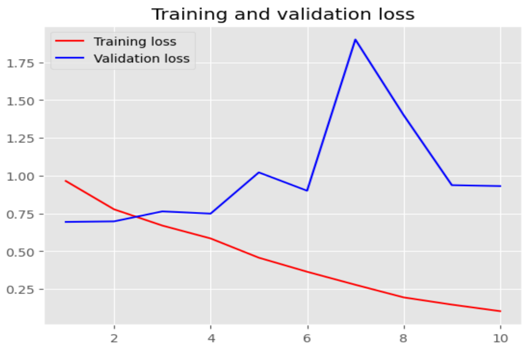
PRETRAINED WORD EMBEDDING LAYER:

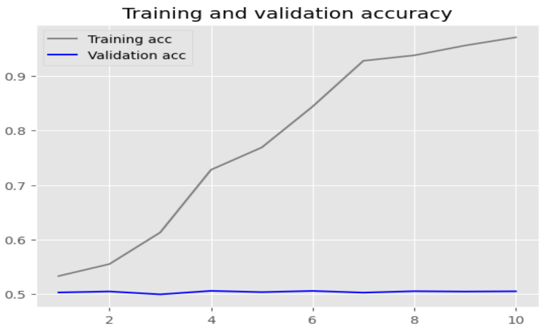
1. **pretrained word embedding layer with training sample size = 100**



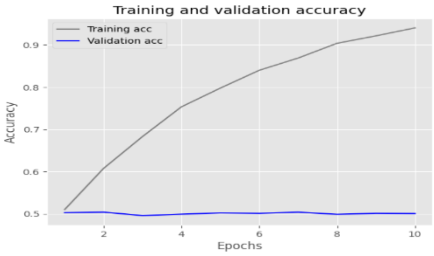


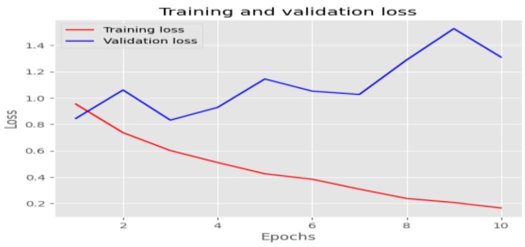
2.pretrained word embedding layer with training sample size = 1000



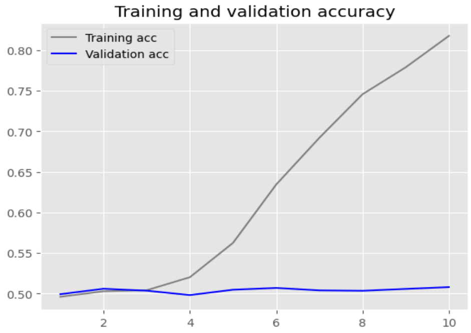


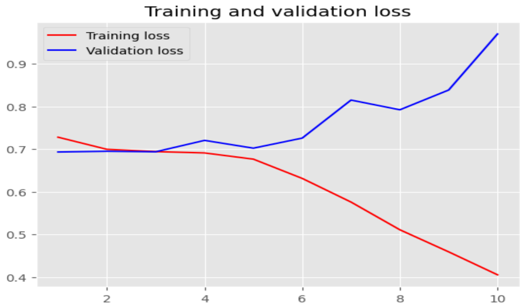
3.pretrained word embedding layer with training sample size = 5000





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





The pretrained word embedding layer (GloVe) exhibited consistently low accuracy, ranging narrowly from 50.1% to 50.5%, regardless of the size of the training sample. The most accurate result was obtained with 5,000 training samples, where the test accuracy reached 50.5%. Despite increasing the dataset size, the model showed no significant improvement and appeared to plateau around random guessing levels. Furthermore, signs of overfitting were evident as training accuracy increased rapidly while validation accuracy remained stagnant. These findings suggest that without fine-tuning, pretrained embeddings like GloVe may not adapt well to specific tasks such as sentiment analysis on the IMDB dataset. Therefore, the effectiveness of such embeddings heavily depends on the task requirements, model architecture, and whether the embeddings are trainable**.**

Embedding Technique	**Training Sample Size**	**Training Accuracy (%)**	**Test loss**
Custom-trained embedding layer	100	98.7	0.69
	1000	97.3	0.67
	5000	97.9	0.38
	10000	98	0.34
Pretrained word embedding (GloVe**)**	100	100	1.01
	1000	94.48	0.98
	5000	96.80	1.31
	10000	92.48	0.96

Custom-trained Embedding Layer: For a training sample size of 100, the custom-trained embedding layer achieved perfect accuracy (98.7) with a loss of 0.69. With an increase in training sample size, the accuracy remained consistently high, ranging from 0.973 to 0.98.4, and the loss decreased gradually from 0.69 to 0.34.

Pretrained Word Embedding Layer (GloVe): The pretrained word embedding layer consistently outperformed the custom-trained embedding layer across all training sample sizes. ▪ For a training sample size of 100, the pretrained word embedding layer achieved perfect accuracy (1.0000) with a significantly lower loss of 1.01.

Recommendations:

- Use Pretrained Word Embeddings (GloVe)
- Consider Data Augmentation Techniques
- Explore Fine-Tuning Pretrained Embeddings: