**Assignment 4**

**Text and Sequence**

**Report**

**Introduction:**

The main goal of this study is to evaluate the performance of Transformers, also known as Recurrent Neural Networks (RNNs), on text data, namely the IMDB movie review dataset. Analyzing model performance, testing strategies to increase accuracy with limited data, and figuring out the best methods for improving predictions are some of the main objectives.

To prepare the IMDB dataset for analysis, several preprocessing steps were taken:

* Review lengthening: To make reviews easier to read, we ensured that they were all no more than 150 words.
* Using a small training set: To make it easier to explore without overwhelming our models, we chose to train them on a mere 100 reviews.
* Testing on a larger sample: To see how well our models perform on a broader scale, we put aside 10,000 reviews.
* Concentrating on popular words: To keep things more straightforward and pertinent, we just looked at the 10,000 most frequently occurring words in the evaluations.

**Methodology:**

Baseline Model: To transform words into dense vectors, the baseline model is a straightforward RNN with an embedding layer. It consists of a thick layer for ultimate prediction and a recurrent layer to capture word dependencies. To avoid overfitting, hyperparameters such as the learning rate and embedding dimension were set, and validation accuracy was tracked. Test accuracy assessed the generalization of the model using an independent dataset.

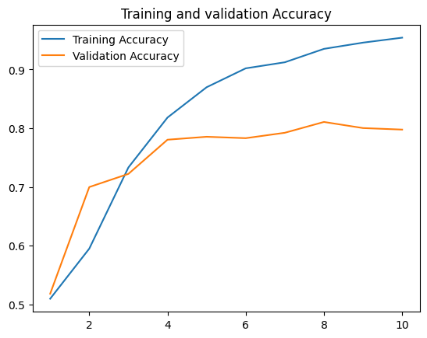
Pretrained Word Embeddings: The model's comprehension of semantics was improved by the use of pretrained word embeddings, such as GloVe. Once placed into the embedding layer, these embeddings allowed extensive text data to capture complex word associations. The weights of the embeddings might be adjusted or left constant throughout training. Test accuracy and validation evaluated the enhancement resulting from pretrained embeddings.

Altering Training Set Size: From small subsets to entire datasets, experiments were conducted to examine how well the model performed with varying training set sizes. Understanding the scalability and robustness of the model was obtained by modifying the training data. Test accuracy and validation were tracked in order to examine performance over a range of training data sizes. This allowed for a comparison of the effectiveness of the embedding layer with pretrained embeddings in various data circumstances.

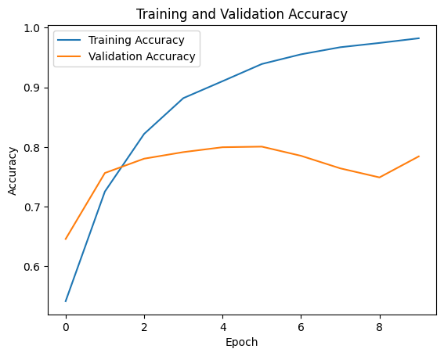
**Results:**

***One Hot Encoded Model:*** A One Hot Encoded Model represents words as binary vectors, with each word uniquely encoded by setting only one element to 1 in a sparse matrix, typically used for text classification tasks due to its simplicity and effectiveness in handling categorical data.

The One Hot Encoded Model achieves a validation accuracy of 0.80 and a loss of 0.43

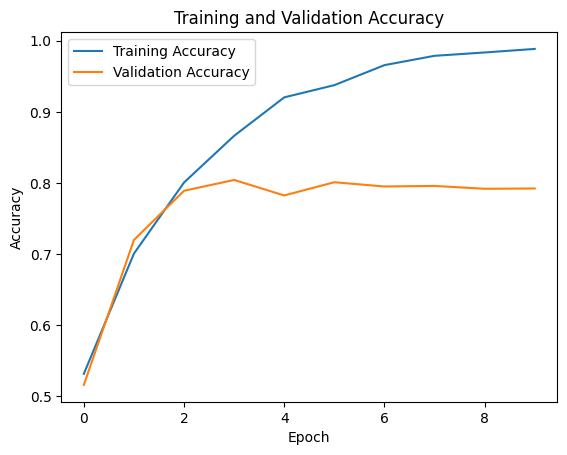


***Trainable Embedding Layer:*** A Trainable Embedding Layer dynamically learns word representations based on the task's objectives during training, achieving an accuracy of 0.77 and a loss of 0.46



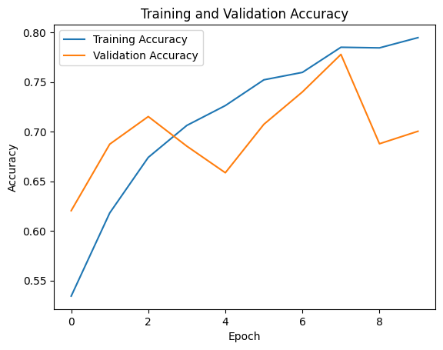
***Masking Padded Sequences in the Embedding Layer:***

Masking padded sequences in the Embedding Layer involves ignoring padded elements during training, ensuring consistent sequence lengths, with an achieved accuracy of 0.79 and a loss of 0.43



***Model with Pretrained GloVe Embeddings:***

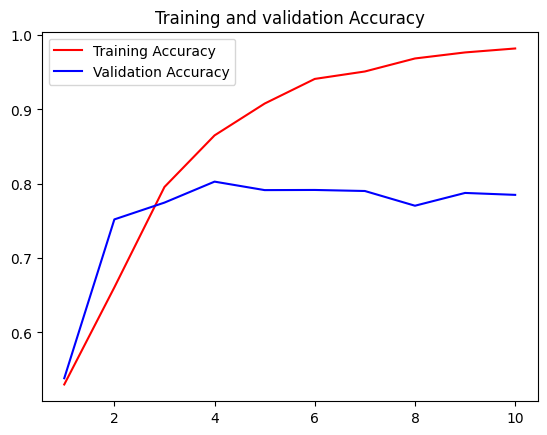
A model utilizing pre-trained GloVe embeddings incorporates externally learned word representations, with a validation accuracy of 0.77 and a loss of 0.47



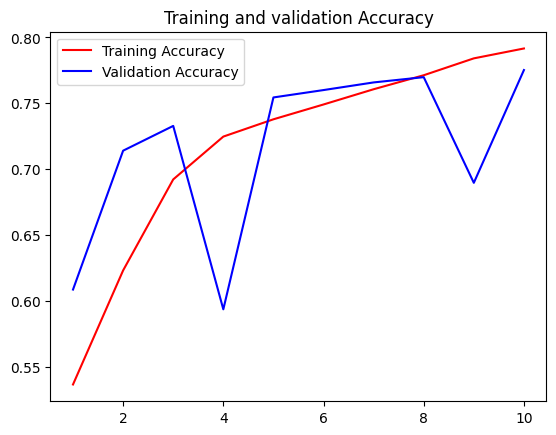
***Comparing Model Performance with Different Training Set Sizes***

**Sample size=100**

An Embedding Layer trained with 100 samples achieves a loss of 0.44 and an accuracy of 0.80

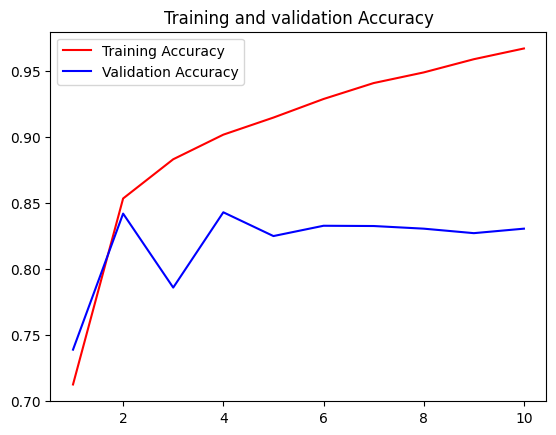


A pretrained embedding layer achieves an accuracy of 0.773 and a loss of 0.4755

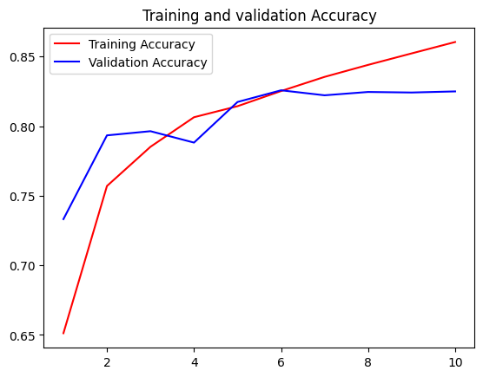


**Sample size= 500**

An embedding layer achieves an impressive accuracy of 0.831 and a loss of 0.3863

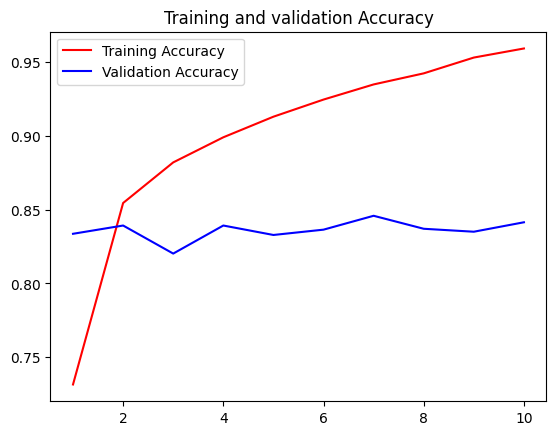


A pretrained embedding layer achieves a validation accuracy of 0.82 and a loss of 0.38

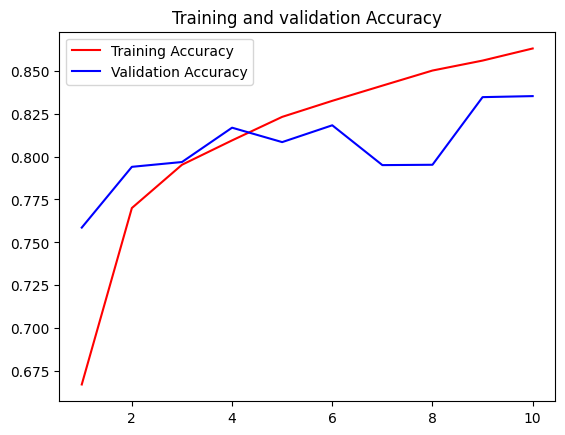


**Sample size= 1000**

An embedding layer achieves an accuracy of 0.829 and a loss of 0.3839

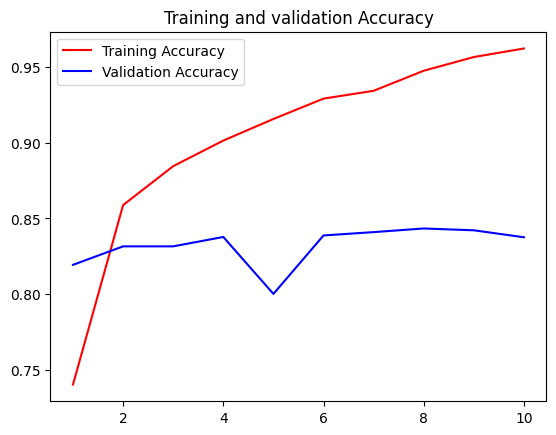


A pretrained embedding layer achieves a validation accuracy of 0.833 and a loss of 0.3731

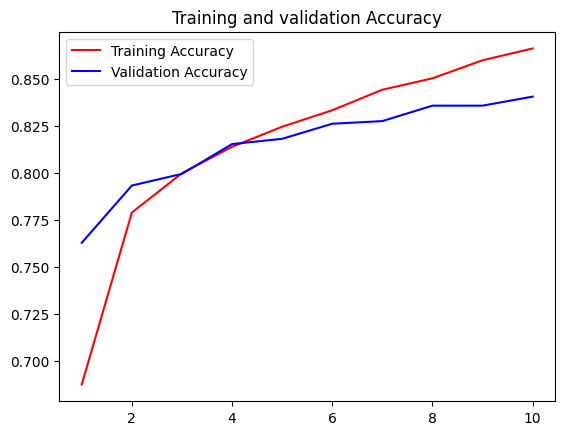


**Sample size= 5000**

An embedding layer achieves a validation accuracy of 0.832 and a loss of 0.3724

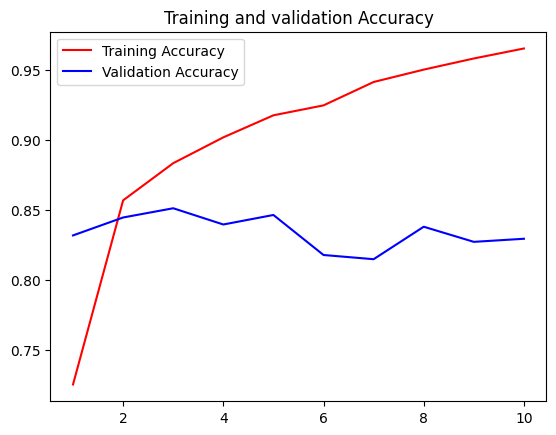


A pretrained embedding layer achieves an accuracy of 0.837 and a loss of 0.3684

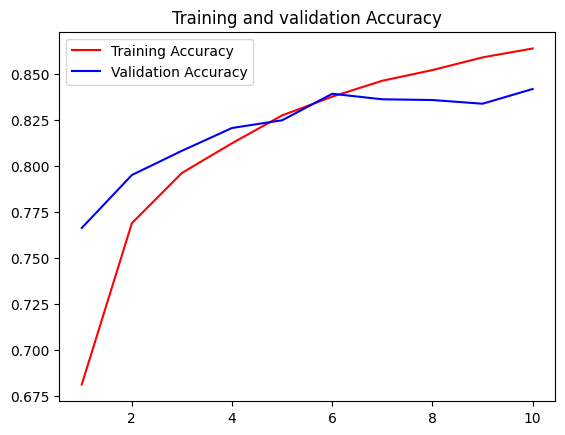


**Sample size= 10000**

An embedding layer achieves a validation accuracy of 0.834 and a loss of 0.3736



A pretrained embedding layer achieves a validation accuracy of 0.837 and a loss of 0.3672



**Results Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample Size** | **Model** | **Accuracy** | **Loss** |
| 100 | One hot encoded | 0.804 | 0.435 |
| Embedded | 0.774 | 0.464 |
| Embedded mask | 0.799 | 0.439 |
| Pre trained | 0.773 | 0.473 |
| 100 | Embedded | 0.801 | 0.448 |
| Pre trained | 0.773 | 0.475 |
| 500 | Embedded | 0.831 | 0.386 |
| Pre trained | 0.825 | 0.385 |
| 1000 | Embedded | 0.829 | 0.383 |
| Pre trained | 0.833 | 0.373 |
| 5000 | Embedded | 0.832 | 0.372 |
| Pre trained | 0.837 | 0.368 |
| 10000 | Embedded | 0.834 | 0.373 |
| Pre trained | 0.837 | 0.367 |

**Conclusion:**

The findings emphasize the differences between trainable embedding layers and pretrained word embeddings, with the best option relying on the amount of training data that is available.

Trainable embedding layers outperformed pretrained embeddings, showing higher validation accuracy (0.801 vs 0.773) and lower loss when the training set was severely constrained to just 100 samples. This implies that learning embeddings from scratch that are suited to the particular job and data is more advantageous when there is very little data available.

But the benefit moved to pretrained embeddings as the size of the training set grew. Pretrained embeddings began to marginally outperform trainable embeddings with 1000 training samples. The difference became even more noticeable between training samples 5000 and 10,000, where pretrained embeddings consistently produced better validation accuracy (around 0.837) than trainable embeddings (0.832–0.834 accuracy).

The results show that while trainable embeddings can work well when there is a shortage of data, using high-quality pretrained embeddings is more beneficial when there are more training instances available. When working with moderate to large training set sizes, pretrained embeddings offer a solid foundation by transferring knowledge from pretraining on large databases, enabling faster and more efficient learning of word representations.