**ASSIGNMENT 4**

**TEXT AND SEQUENCE**

**VIJAY CHARAN REDDY GOTTAM**

**KSU ID: 811292996**

**Summary:**

This study investigates the effectiveness of various techniques in classifying sentiment (positive or negative) within movie reviews. A large IMDB dataset is employed, focusing on the most frequently occurring 10,000 words. The sentiment classification model is trained on varying data sizes (100, 500, 1,000, and 100,000 samples) with validation occurring on a separate set of 10,000 reviews. Pre-processing techniques are applied to the data before feeding it into a pre-trained word embedding model. Through evaluation of these various approaches, the aim is to identify the most optimal method for sentiment analysis in movie reviews.

**Techniques:**

**Preprocessing of the dataset:**

The IMDB dataset categorizes movie reviews as positive or negative. Preprocessing transforms each review into a sequence of word embeddings, essentially fixed-length vectors representing each word. While the dataset is limited to 10,000 samples, converting reviews from text to integers (each representing a unique word) creates a sequence unsuitable for neural networks. Networks require tensors, so these integer lists need conversion. To achieve this, a tensor with integer data type and a shape of (samples, word indices) is created. However, ensuring all reviews have the same length is crucial. This involves padding shorter reviews with dummy integer values to create uniformity.

**Approach:**

This study investigated two approaches for generating word embeddings in the IMDB sentiment analysis task:

Pre-trained Word Embedding Layer (GloVe): A pre-trained GloVe model was employed. This widely used model learns word representations from massive text datasets, effectively capturing semantic and syntactic relationships between words. Its effectiveness makes it a popular choice for natural language processing tasks.

Custom-trained Embedding Layer: This approach involves training a custom embedding layer specifically for the IMDB dataset. This allows the model to potentially learn word representations tailored to the domain of movie reviews.

This study compared the effectiveness of different word embedding techniques for sentiment analysis on movie reviews. We utilized a pre-trained GloVe model (6B tokens, 400k words) trained on Wikipedia and Gigaword 5 to create word embeddings. To assess these techniques, two separate embedding layers were implemented within the model: a custom-trained layer specific to the IMDB data and the pre-trained GloVe layer. The models' performance was evaluated using various training data sizes ranging from 100 to 10,000 samples.

I investigated the effectiveness of word embeddings by training a custom embedding layer on the IMDB dataset and comparing its accuracy on various sample sizes to a model using a pre-trained embedding layer evaluated across different training data amounts.

**Custom-trained embedding layer**

Custom-trained embedding layer with training sample 100

A graph with blue dots

Description automatically generatedA graph with red dots

Description automatically generated

Custom-trained embedding layer with training sample 500

A graph with blue dots

Description automatically generated A graph of training and validation loss

Description automatically generated

Custom-trained embedding layer with training sample 1000

A graph with blue dots

Description automatically generated A graph with red dots

Description automatically generated

Custom-trained embedding layer with training sample 10000

A graph with a line graph

Description automatically generated A graph with red dots

Description automatically generated

**Pretrained word embedding layer (GloVe)**

Pretrained word embedding layer (GloVe) with training sample 100

A graph with blue lines

Description automatically generatedA graph with red lines and dots

Description automatically generated

Pretrained word embedding layer (GloVe) with training sample 500

A graph with blue dots

Description automatically generated A graph with red dots and a line

Description automatically generated

Pretrained word embedding layer (GloVe) with training sample 1000

A graph with blue dots

Description automatically generated A graph with red lines and numbers

Description automatically generated

Pretrained word embedding layer (GloVe) with training sample 10000

A graph with blue dots

Description automatically generated A graph with red dots

Description automatically generated

**Results:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.NO** | **Embedding Technique** | **Training sample size** | **Loss and Accuracy on Test** | **Accuracy** |
| **1** | Custom-trained embedding layer | 100 | **Loss: 0.694**  **Accuracy:0.498** | **98.7** |
| **2** | Custom-trained embedding layer | 500 | **Loss:0.691**  **Accuracy: 0.525** | **97.0** |
| **3** | Custom-trained embedding layer | 1000 | **Loss:0.679**  **Accuracy:0.568** | **97.5** |
| **4** | Custom-trained embedding layer | 10000 | **Loss:0.340**  **Accuracy:0.568** | **97.6** |
| **5** | Pretrained word embedding layer (GloVe) | 100 | **Loss:0.938**  **Accuracy:0.495** | **100** |
| **6** | Pretrained word embedding layer (GloVe) | 500 | **Loss: 0.905**  **Accuracy:0.501** | **99.2** |
| **7** | Pretrained word embedding layer (GloVe) | 1000 | **Loss:0.989**  **Accuracy:0.501** | **96.2** |
| **8** | Pretrained word embedding layer (GloVe) | 10000 | **Loss:1.305**  **Accuracy:0.501** | **81.9** |

**Custom-trained embedding layer:**

Our investigation revealed that the custom-trained embedding layer achieved impressive accuracy, ranging from 97.0% to 98.7%, with the peak performance occurring at a training sample size of only 100. This superior performance can likely be attributed to the layer's ability to learn domain-specific word representations highly relevant to the task of IMDB review sentiment classification. Interestingly, the accuracy plateaued after 100 samples, suggesting that the custom layer may not require vast amounts of training data to achieve optimal performance.

**Pretrained word embedding layer (GloVe):**

Our evaluation revealed that pre-trained GloVe embeddings delivered strong performance (81.9% - 100% accuracy) even with limited training data, achieving peak accuracy with only 100 samples. This is likely attributable to their ability to capture rich semantic information inherent in language. However, as the training data size increased, the pre-trained embeddings may not have been able to capture the nuances specific to the sentiment analysis task of movie reviews, potentially leading to a decrease in accuracy and overfitting with very large datasets**.**

The study's findings regarding the “best” embedding technique are context-dependent. While the custom-trained embedding layer achieved superior performance overall, particularly with larger datasets, resource constraints and limited training data favor the pre-trained GloVe embeddings. However, careful mitigation strategies are necessary to prevent overfitting when employing pre-trained embeddings with larger datasets.

**Conclusion:**

To get exceptional performance, neural network models for text classification need meticulous evaluation of pretrained networks, embeddings, and training set size. These variables offer valuable data that may be utilized to enhance and adjust models in different contexts. Complex language patterns and semantic representations can be captured by the model thanks to pretrained networks, especially those that have been trained on big datasets. These learned representations are helpful for tasks like text categorization since they were taken from a lot of textual input. In order to give machine learning models context and word meaning, embeddings are necessary. They enhance the model's text classification capabilities by enabling a more complex understanding of the links between words. The size of the training set is a major factor influencing a model's capacity for generalization.

Pre-trained embeddings are thought to improve model performance; nevertheless, the results showed that the straightforward embedding layer model outperformed the pre-trained model. It is often crucial to keep in mind that the pre-trained model in this case is not ideal for the given job and did not refine the embeddings throughout training. Primarily, optimizing the embeddings might potentially improve performance.

In conclusion, we should be attentive when drawing conclusions from these findings since they rely on a limited set of hyperparameters and a small number of training samples. Other results could arise by adjusting the hyperparameters or using additional training data.