**ASSIGNMENT-4: TEXT AND SEQUENCE**

**REPORT**

Group 7: Akhil Seliveri

Ritika Kalyani

**Problem Statement:**

Consider the IMDB example from Chapter 6. Re-run the example modifying the following:  
1. Cutoff reviews after 150 words.  
2. Restrict training samples to 100.  
3. Validate on 10,000 samples.  
4. Consider only the top 10,000 words.  
5. Consider both a embedding layer, and a pretrained word embedding. Which approach  
did better? Now try changing the number of training samples to determine at what  
point the embedding layer gives better performance.

**Data:**

The IMDB dataset in Keras is a common selection for sentiment analysis. It includes movie reviews from the Internet Movie Database (IMDb), which are categorized as either positive or negative depending on the sentiment expressed in the review. The dataset contains 50,000 movie reviews. We evaluate only the top 10,000 words, limit training samples to different sample sizes, validate on 10,000 samples, and end reviews at 150 words. Pre-processing is performed on the data. Following that, we input data to both a pretrained embedding model and an embedding layer, and we evaluate performance using different approaches.

Every review is transformed into a word embedding, where a fixed-size vector represents each word. There is a 10,000-word limit on the vocabulary. Reviews are transformed into sequences of integers, where a word is represented by each integer. By using padding to ensure constant length, integers are transformed into tensors, which makes input into the neural network easier.

**Results:**

* **Model built from scratch using embedding layer:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Model Number** | **Description** | **Test Accuracy** | **Test loss** |
| 1 | Model from scratch | 85 | 0.42 |

Using embedding layers, we first created the model from scratch with 10,000 words. We then stopped reviewing after 150 words, recording an accuracy of 85% and a loss of 0.42.

* **Models with embedding layer and different sample sizes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Model Number** | **Description** | **Test Accuracy** | **Test loss** |
| 1. | 100 samples | 49 | 0.69 |
| 2. | 6000 samples | 83 | 0.38 |
| 3. | 12000 samples | 85 | 0.39 |

After that, we began training the model using three different sample sizes—100, 6000, and 12000—and observed the following outcomes:

1. With 100 samples, the basic model created from scratch shows an accuracy of 49% and a loss of 0.69, both of which are good for the sample size of 100.
2. The accuracy has now increased to 83% on the set of 6000 samples, and the loss has dropped to 0.38, indicating that the model is becoming better.
3. Now that the sample size has been increased by 12000, the accuracy is 85% and the test loss is 0.39. Thus, we find that the model begins to perform better as the sample size grows.

* **Models with embedding layer and pre-trained model with different sample sizes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Model Number** | **Description** | **Test Accuracy** | **Test loss** |
| 1. | 100 samples | 49 | 1.73 |
| 2. | 6000 samples | 49 | 1.73 |
| 3. | 12000 samples | 50 | 1.84 |

We proceed to integrate the pre-trained model with the embedding layer and observe the resulting outcomes:

1. The pretrained model with 100 training examples gives a loss of 1.73, which is greater, and an accuracy of 0.49, which is lower. Let's examine what happens when we improve the sample size.
2. The pretrained model with 6000 training examples gives an accuracy of 0.49, which is identical to the prior model's accuracy, and a loss of 1.73, indicating that no apparent improvements were seen even after the training sample size was doubled. This provides further proof in support of the idea that performance increases from increasing the size of the training sample may not always be significant. Let's expand the sample size even further and see the outcomes.
3. With 12,000 training samples, the pretrained model gives a loss of 1.84, which is greater than the earlier model, and an accuracy of 0.50, which is somewhat higher. We observe little to no improvement in accuracy and an increase in loss, indicating that there may not be much of an impact from adding more training data on improving model performance. Moreover, the increase in loss suggests that the model might be overfitting or that noise in the new data is making it less accurate in its predictions.

Due to their ability to capture a significant portion of the text's underlying semantic information, pretrained embeddings can be effective with small training data, which may explain their great precision even with limited training samples. Still, when the training sample size increases, the pretrained embeddings may not be as good at capturing the fine features of the specific task at hand, which could lead to decreased accuracy. Moreover, using the pretrained embeddings with bigger training sample sizes, as the prompt says, rapidly overfits the model, reducing accuracy.