ADVANCED MACHINE LEARNING

ASSIGNMENT-04

INTRODUCTION:

Recurrent Neural Networks (RNNs) have the ability to maintain an internal state or memory that permits them to process sequences of inputs one element at a time while also taking into account the previous elements in the sequence, thereby making them an effective tool for processing sequential data, such as text and speech. Additionally, a wide range of natural language processing (NLP) problems, comprising language modeling, machine translation, sentiment analysis, and named entity recognition, have been successfully addressed by RNNs.

Table: Performance Results

Sample size	**One hot encoded sequence**		**Embedded Layer**		**Embedded Masked**		**Pre-Trained**	
	Test Loss	**Test Accuracy**	**Test Loss**	**Test Accuracy**	**Test Loss**	**Test Accuracy**	**Test Loss**	**Test Accuracy**
Initial	0.3096	0.880	0.3973	0.858	0.3016	0.878	0.2982	0.873
100	0.4422	0.804	0.4779	0.775	0.4476	0.797	0.4799	0.767
500	0.3558	0.847	0.3782	0.833	0.3680	0.840	0.3730	0.833
1000	0.3893	0.836	0.3705	0.833	0.3733	0.839	0.3765	0.839
2000	0.3587	0.846	0.3806	0.838	0.3836	0.831	0.3672	0.837
4000	0.4063	0.833	0.3966	0.832	0.3760	0.840	0.3708	0.833
8000	0.3528	0.846	0.3751	0.835	0.3726	0.837	0.3688	0.839

Summary of the results:

- Based on various sample sizes and embedding methods, the table compares how well a neural network model performs on a particular task. Four different embedding types were evaluated: one hot encoded sequence, embedded layer, embedded masked, and pre-trained embeddings.
- The initial findings demonstrated that the model used a single hot encoded sequence embedding to obtain a high test accuracy of 0.880. However, all embeddings saw a decrease in test accuracy and a rise in test loss when the sample size was increased to 100, with the exception of the embedded layer, embedded masked, and pre-trained embeddings, which displayed comparable results.
- The test loss decreased for all four embedding approaches when the sample size was increased to 500, with the embedded layer exhibiting the lowest test loss of 0.3782 and the highest test accuracy of 0.833. Similar outcomes were seen when the sample

size was increased to 1000, with the embedded layer achieving the lowest test loss of 0.3705 and the best test accuracy of 0.833.
- The embedded layer and embedded masked outperformed the other embeddings when the sample size was increased to 2000 and 4000, respectively. When the sample size was extended to 8000, however, all embeddings had comparable test loss and accuracy, with the embedded layer achieving test losses of 0.3751 and test accuracys of 0.835.

Conclusion:

Based on the table provided, the following conclusions can be drawn:

- The initial results showed that one hot encoded sequence embedding had the lowest test loss of 0.3096 and the highest test accuracy of 0.880.
- As the sample size increased from 100 to 500 and 1000, the embedded layer consistently performed the best with the lowest test loss and highest test accuracy.
- For larger sample sizes of 2000, 4000, and 8000, the embedded layer and embedded masked showed better performance compared to the other embeddings.
- Overall, the embedded layer showed consistent performance across different sample sizes, while the other embeddings showed mixed results.
- When the sample size was large enough, the performance of all embeddings converged.
- Therefore, the embedded layer is the recommended technique for this RNN model, as it consistently performed the best across different sample sizes.

A larger dataset should ideally be used for training in order to improve model performance. Additionally, based on the precise requirements and limitations of the job or application at hand, the cutoff length and number of training samples should be carefully chosen.

To choose the best model parameters, additional testing and review may be required. Overall, it is possible to train models with less data, but it's important to be aware of any potential limits and take extra measures, such as data augmentation, transfer learning, or using pre-trained models, to improve the model's performance.