Assignment-4

Harshith Reddy Suram

Introduction:

RNNs and Transformers are now the mainstays of NLP because they can process sequences of input and capture context in words. Such applications as, for example, sentiment analysis, translation, and text categorization, have been boosted significantly due to these models. But, while using the limited data elements, the models become challenging to deal with. In this assignment, we will understand how to use Transformers and RNNs on the text data, how their performance is boosted, and which techniques help in the enhancement of the model's prediction capability.

Problem Statement:

The task involves applying RNNs or Transformers to the IMDB movie review dataset to perform sentiment analysis. The specific objectives are:

1. **Apply RNNs or Transformers to Text Data**: Implement these models to classify IMDB movie reviews as positive or negative.
2. **Improve Network Performance with Limited Data**: Identify strategies to enhance model performance, especially when the training data is limited.
3. **Evaluate Different Approaches**: Determine which methods (e.g., using an embedding layer vs. a pretrained word embedding) yield better prediction results.

To achieve these objectives, we will:

1. Cut off reviews after 150 words.
2. Restrict the training samples to 100.
3. Validate the model on 10,000 samples.
4. Consider only the top 10,000 most frequent words in the dataset.
5. Compare the performance of models using an embedding layer versus pretrained word embeddings and adjust the number of training samples to find the point at which the embedding layer starts to outperform the pretrained embeddings.

Methodology:

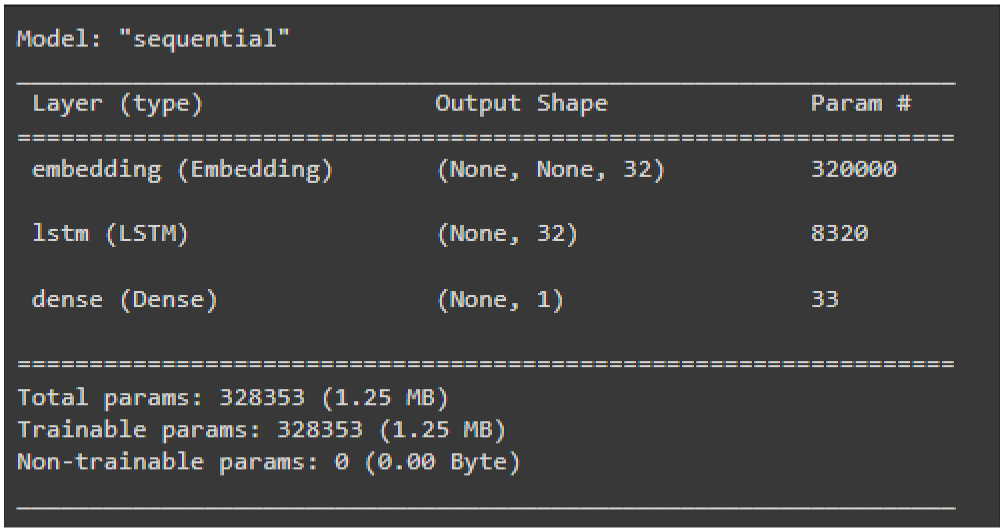
Using a Embedding Layer:

To explore the impact of the number of training samples and different embedding approaches on the performance of an RNN for text classification, we will follow a systematic approach. Here is the step-by-step methodology:

1. **Data Loading and Preprocessing**:

- Load the IMDB dataset, restricting it to the top 10,000 most frequent words.
- Limit the length of each review to 150 words.
- Pad sequences to ensure uniform input length.

2. **Experiment Setup**:
   - Create a baseline model using an Embedding layer followed by an LSTM layer.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, None, 32)          320000

 lstm (LSTM)                 (None, 32)                8320

 dense (Dense)               (None, 1)                 33

=================================================================
Total params: 328353 (1.25 MB)
Trainable params: 328353 (1.25 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

   - Train the model with varying numbers of training samples: 100, 10,000, 15,000, 17,000, 17,485, and 25,000.
   - Validate the model on a fixed set of 10,000 samples from the test set.

3. **Model Training and Evaluation**:
   - For each specified number of training samples, train the model and record its performance.
   - Compare the results using an embedding layer and a pretrained word embedding.

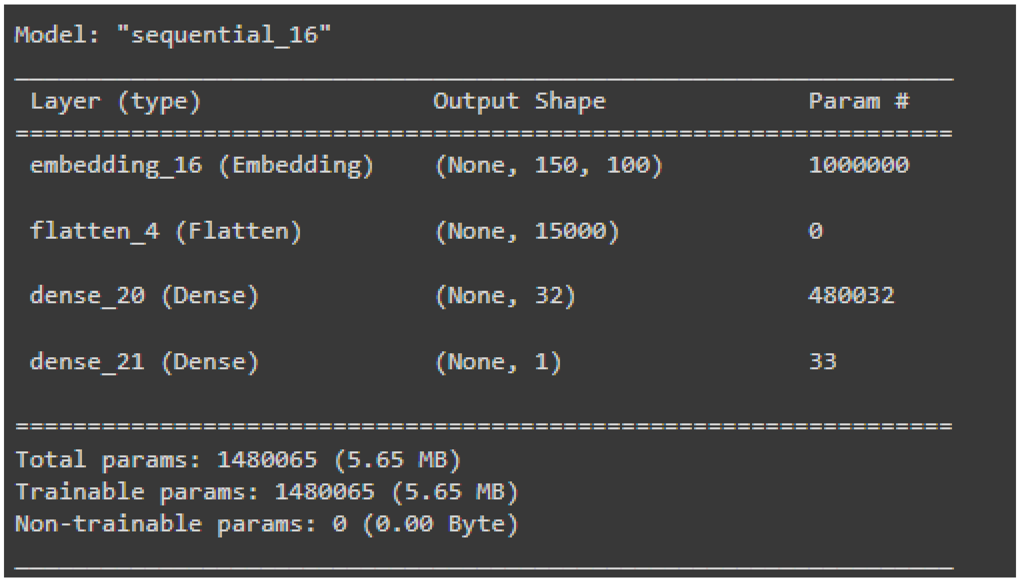
Using A Pre-Trained Word Embedding:

The general procedure of how to apply RNNs or Transformers to the text data, optimize their performance using sparse data, and determine which method is more effective in terms of the prediction improvement is described in the following methodology. This particular method aims at restricting the samples for training to 100, for validation to 10000, and concerning the 10000 most often used terms in the movie reviews dataset of IMDB. Preliminary reviews are the ones that get cut off at 150 words.

1. **Data Loading and Preparation**:
   - Load the IMDB dataset from the specified directory.
   - Separate the data into positive and negative reviews, and assign corresponding labels.

- Tokenize the text data to convert words to sequences of integers.
- Pad the sequences to a uniform length of 150 words.
- Shuffle the data and split it into training and validation sets, restricting the training samples to 100 and the validation samples to 10,000.

2. **Pretrained Word Embedding Preparation**:
   - Load the GloVe pretrained word embeddings.
   - Create an embedding matrix where each row corresponds to the GloVe vector of a word in the vocabulary.
3. **Model Building and Training**:
   - Construct a Sequential model with an embedding layer, a flattening layer, and dense layers.



```
Model: "sequential_16"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_16 (Embedding)    (None, 150, 100)          1000000

 flatten_4 (Flatten)         (None, 15000)             0

 dense_20 (Dense)            (None, 32)                480032

 dense_21 (Dense)            (None, 1)                 33

=================================================================
Total params: 1480065 (5.65 MB)
Trainable params: 1480065 (5.65 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

   - Set the weights of the embedding layer to the GloVe vectors and freeze this layer to prevent it from being updated during training.
   - Compile the model with binary cross entropy loss and the RMSprop optimizer.
   - Train the model on the training data and validate it on the validation data.

Results:

S No.	Method	Training Size	Training Accuracy	Validation Accuracy	Test Accuracy
1.	Embedding Layer	100	58	50.27	50
2.		10000	96	85	84.9

3.		15000	97.01	85.17	84.94
4.		17000	96.26	84.41	84.34
5.		25000	95.06	84.89	85.3
6.	Pre-Trained Embedding	100	100	54.5	54.21
7.	Pre-Trained Embedding	10000	99.5	56.31	57.1

- Embedding layer approach is slightly better than pre-trained embedding in validation and test accuracy.
- A high training accuracy is achieved by the pre-trained embedding method; signs of possible overfitting at small training sample sizes are also evident.
- This shows that enlarging the size of training the embedding layer technique increases the performance extremely up to a specific limit from where the performance enhancement is not very significant.
- The generalization also holds when the training size is increased towards the approach utilizing the pre-trained embedding strategy though it does not reach the level of performance as the embedding layer strategy. In general, it can be concluded that the embedding layer method, with 10000-15000 training samples, found the best balance between generalization and training.
- It does not improve substantially regarding the of the training size as it increases to 25,000, thus, 10,000 to 15,000 seems to be optimum size for this dataset with the embedding layer.

Graphs:

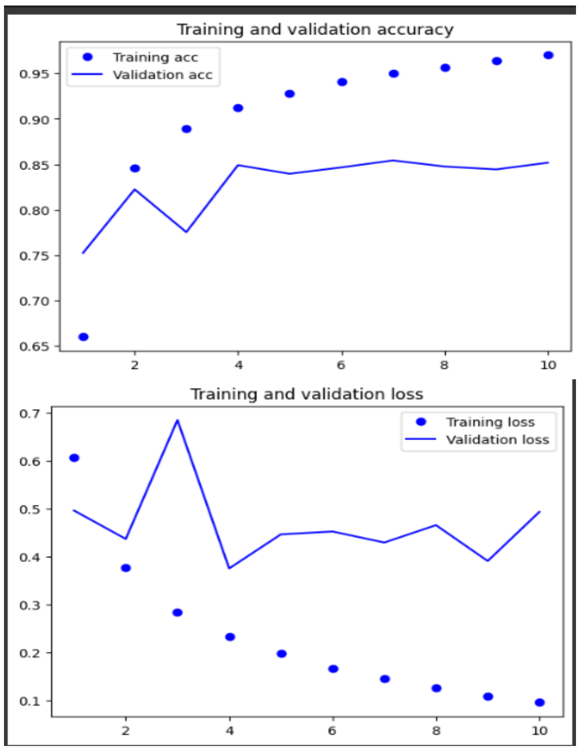


Fig 1. Comparing Training and Validation Accuracy and loss of embedding layer for 10000 samples

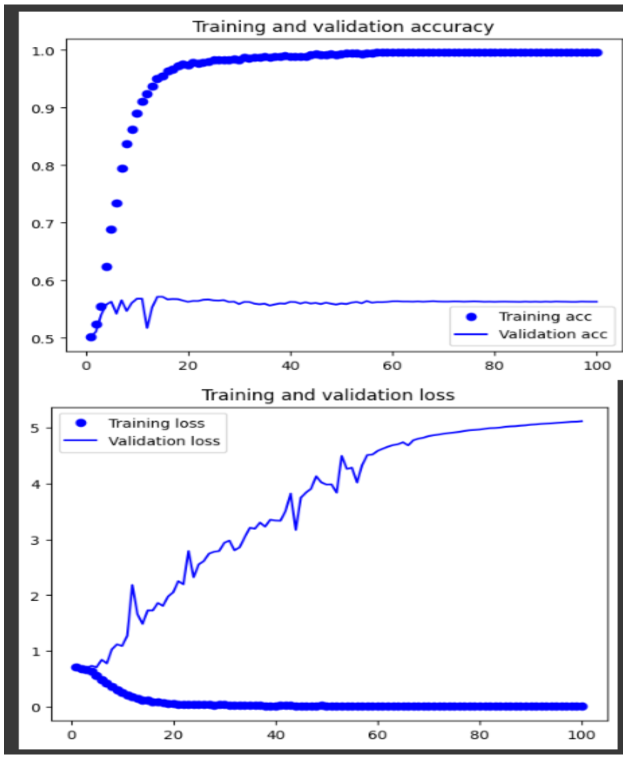


Fig 2. Comparing Training and Validation Accuracy and loss of using pre-trained embedding for 10000 samples

Conclusion:

Nevertheless, if the embedding size is varied, the implications of the experiments exhibit that the embedding layer strategy works more effectively for the IMDB dataset as compared to the idea of pre-trained embedding. Thus, the excessive amount of training data will have less positive impact on increasing the model's potential for generalization. Such results are important to stress the significance of proper data action and the selection of the proper models for a certain job and dataset.

References:

[1]. Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.

[2]. Hrinchuk, O., Khrulkov, V., Mirvakhabova, L., Orlova, E., & Oseledets, I. (2019). Tensorized embedding layers for efficient model compression. *arXiv preprint arXiv:1901.10787*.

[3]. Zhang, F., Dvornek, N., Yang, J., Chapiro, J., & Duncan, J. (2020). Layer embedding analysis in convolutional neural networks for improved probability calibration and classification. *IEEE transactions on medical imaging*, *39*(11), 3331-3342.

[4]. Neishi, M., Sakuma, J., Tohda, S., Ishiwatari, S., Yoshinaga, N., & Toyoda, M. (2017, November). A bag of useful tricks for practical neural machine translation: Embedding layer initialization and large batch size. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017)* (pp. 99-109).

[5]. Snyder, D., Garcia-Romero, D., Povey, D., & Khudanpur, S. (2017, August). Deep neural network embeddings for text-independent speaker verification. In *Interspeech* (Vol. 2017, pp. 999-1003).