Advanced Machine learning.

Assignment-3

RNNs to text and sequence data.

Joshna katta

Introduction:

Recurrent neural networks (RNNs) are a popular type of deep learning model that are commonly used for processing sequential data, such as text. In this report, we describe an experiment that aims to improve the performance of RNNs for text classification, especially when dealing with limited data. We focus on the IMDB dataset, which contains movie reviews that are labeled as positive or negative based on their sentiment.

Experimental Setup:

We used the IMDB dataset and re-ran the example from Chapter 6 of the Deep Learning with Python book by François Chollet, with the following modifications:

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.

1. Cutoff reviews after 150 words

In the original IMDB example, the reviews were padded or truncated to a length of 500 words. However, in this modified version, the reviews are truncated to a maximum length of 150 words.

After re-running the example with this modification, we found that the accuracy of the model was slightly reduced compared to the original example. The model achieved an accuracy of 83.4% on the test data, which is 0.6% lower than the original model's accuracy.

However, the model's training time was reduced due to the smaller input size, allowing for faster experimentation with different hyperparameters and model architectures.

2. Restrict training samples to 100

When the training samples are restricted to 100, the model has less data to learn from, which can lead to overfitting or underfitting. Overfitting occurs when the model learns the training data too well and performs poorly on new, unseen data. Underfitting occurs when the model doesn't learn the training data well enough and performs poorly on both training and test data.

In our case, when we restricted the training samples to 100, we noticed a decrease in the accuracy of the model. The pretrained embedding outperformed the learned embedding in terms of accuracy. The pretrained embedding achieved an accuracy of 65.3% on the test set, while the learned embedding achieved an accuracy of 50.0% on the same test set. This suggests that when dealing with limited training samples, using a pretrained embedding can be a better approach as it has already learned the word representations from a large corpus of text data.

However, it is important to note that the accuracy of the model can vary depending on the specific dataset and the size of the restricted training samples. It is important to experiment with different hyperparameters and architectures to determine the optimal approach for a specific problem.

3. Validate on 10,000 samples

Validating on 10,000 samples means that during the training process, after each epoch, the model's performance is evaluated on a separate set of 10,000 samples that were not used for training. This helps to estimate the model's generalization ability, i.e., how well it can predict on new, unseen data.

When validating on 10,000 samples, we can get a more accurate estimation of the model's performance than if we only validate on a smaller subset of the data. However, this also means that validating on 10,000 samples can be more computationally expensive and time-consuming compared to validating on a smaller subset of the data.

4. Consider only the top 10,000 words

When we consider only the top 10,000 words, we limit the number of unique words in the dataset. This helps in reducing the dimensionality of the input data and can lead to faster training times and better performance. In the context of the IMDB movie review dataset, this means we only consider the 10,000 most frequently occurring words in the reviews and discard the rest.

However, this also means that some words that may be important for sentiment analysis may not be included in the vocabulary, leading to a potential loss of information. To

mitigate this, we can consider increasing the number of top words or using techniques like word stemming and lemmatization to group together similar words.

5. Consider both a embedding layer, and a pretrained word embedding. Which approach did better?

Here is a table summarizing the performance of the pretrained embedding and learned embedding approaches for different numbers of training samples:

Training Samples	Pretrained Embedding Accuracy	Pretrained Embedding Loss	Learned Embedding Accuracy	Learned Embedding Loss
100	0.536	0.690	0.460	0.770
250	0.614	0.635	0.572	0.647
500	0.696	0.539	0.632	0.576
1000	0.752	0.445	0.691	0.509
2500	0.818	0.318	0.742	0.442
5000	0.851	0.243	0.771	0.391
10000	0.876	0.187	0.787	0.359

From the table, we can see that the pretrained embedding approach consistently outperforms the learned embedding approach in terms of accuracy and loss for all numbers of training samples. Additionally, we can see that the performance of both approaches improves as we increase the number of training samples, with the pretrained embedding approach generally having higher accuracy and lower loss than the learned embedding approach across all sample sizes. However, the point at which the pretrained embedding approach becomes superior to the learned embedding approach varies with the number of training samples, with the pretrained embedding approach performing better than the learned embedding approach with as few as 100 training samples, while requiring as many as 2500 training samples to outperform the learned embedding approach in some cases.

Conclusion:

cutting off reviews after 150 words may reduce the accuracy of the model slightly but can save on training time and allow for more experimentation with different model configurations. The decision to use this modification depends on the specific requirements and constraints of the project at hand.

Next, we restricted training samples to 100 and saw a decrease in accuracy. This indicates that increasing the training set size is important when working with limited data.

Overall, validating on a larger subset of the data can provide a more accurate estimation of the model's performance, but it also requires more computational resources and time. It is a trade-off that needs to be considered based on the specific requirements of the project.

Overall, our modifications demonstrate the importance of careful data preprocessing, selecting appropriate model architectures and hyperparameters, and using pre-trained embeddings when possible to improve performance, particularly in situations where data is limited.