Assignment 4: Text Data

Overview and Objective

This project has the aim of doing binary sentiment classification on the IMDB movie reviews dataset, in terms of whether the review is a positive or a negative review. There are 50,000 reviews in the dataset, and the top 10,000 words are used to analyze it so as to maximize computational performance.

The project finds out the impact of training data size and embedding strategies on the model performance. Training is done on various sample sizes, including 100, 1,000, 3,000, 5,000, 10,000 and 20,000 reviews and checked on a fixed validation sample of 10,000 reviews. Another aspect that the study compares is models that use a trainable embedding layer with those that use pretrained embeddings (e.g., GloVe).

Dataset Overview

- Dataset Size: 50, 000 labeled movie reviews (half positive and half negative).
- Vocabulary: Restricted to the 10,000 most common words in order to decrease the dimensions.
- Split of Data: Training sets of different sizes; validation set: 10,000 samples.
- Text Length: The reviews will be limited to 150 words to achieve consistent input length.

Approach

1. **Data Preprocessing**

- Sample Limitation: The sample size used in the formation of training subsets is between 100 and 20,000.
- Text Truncation: Each review is cut to 150 tokens so that only the same number of tokens is used.
- Vocabulary Cut: The top 10,000 words are kept and the rest are eliminated since they are considered rare or infrequent

2. **Word Embeddings**

- Word embeddings encode words into numerical vectors which encode semantic meaning and contextual relationships.

- Learned Embeddings: This model is meant to find embeddings during training and utilize them to maximize the sentiment classification task.

- Pretrained Embeddings: Pretrained vectors including GloVe, Word2Vec or FastText are incorporated to accept linguistic understanding gained in big external corpora. Generalization is enhanced by this method particularly in scenarios where the training data is scarce.

Model Architecture

Recurrent Neural Network (RNN)

The fundamental model utilizes an RNN structure and this type of architecture is ideal in the processing of sequential data like text.

- Embedding Layer: Transforms every word to a dense representation of vectors (trainable or not).

- RNN Layer: It represents the dependencies and contextual information in the sequence.

- Dense Output Layer: It generates a binary outcome (positive or negative sentiment).

Hyperparameters

Some of the important hyperparameters are:

- Embedding: Characterizes the size of the vectors of each word.

- RNN units: Determines the capacity of the model to learn time.

- batch size, learning rate, epochs: Optimized.

Training and Validation

- Subsets of different sizes are used to train the model and assess the effect of data availability on the performance.
- Training Sizes: 100, 1,000, 3,000, 5,000, 10,000, and 20,000 samples.
- Validation Set: 10,000 reviews that are kept constant throughout the experiments.
- Measures: Accuracy and loss are calculated on validation and test data.

Methodology

1. Baseline Model (Trainable Embedding Layer)

- Description Uses an embedding layer that is initialised randomly and trained with the model.
- Architecture: - RNN - Embedding - Dense Output.
- Assessment: The measure of performance in terms of validation and test accuracy/ loss to come up with a baseline to compare performance. Embeddings are used pretrained in this model.

2. Model with Pretrained Embeddings

- Description: Uses pretrained word embeddings (e.g., GloVe) to offer semantically high-context word representations.
- Process: The embedding layer is loaded with pretrained vectors and can be either fixed or trainable.
- Assessment: As compared to the baseline to measure the increase in the learning efficiency and accuracy.

Varying Training Set Size

The volume of training data is systematic.

- Objective: Find out the effect of increasing the data on the model generalization and stability.
- Assessment: Models are trained on larger subsets, and analysis of performance trends is done across embedding strategies.

Comparison: Trainable vs. Pretrained Embeddings.

A comparison between the embedded choice and the classification of sentiment shows the effect of embedding choice:

- Embedding Layer-Trainable: Trains task-relevant representations, which can need additional data to succeed.
- Pretrained Embeddings: Are semantically better grounded and are usually more accurate with less training data.

Key Insights

- As training data is increased, it tends to increase model performance at a decreasing rate beyond a given size.
- Embeddings are pre-trained and thus converge faster and perform better when performing on smaller data sets.
- RNN-based architecture is appropriate to sentiment analysis because it effectively utilizes contextual information.

Model	Accuracy (%)	Loss	Validation Accuracy (%)	Validation Loss
One-Hot Model	78.6	0.46	78.7	0.459
Trainable Embedding Layer	50	0.693	50	0.63
Masking Padded Sequences (Embedding Layer)	78.8	0.484	79.6	0.476
Pretrained GloVe Embeddings	77.9	0.459	0.78	0.457

Training Samples	Model Type	Accuracy (%)	Loss
100	Embedding Layer	79.7	0.484
100	Pretrained Embedding Layer	53.5	0.689
500	Embedding Layer	79	0.454
500	Pretrained Embedding Layer	54.9	0.684
1,000	Embedding Layer	79.3	0.448
1,000	Pretrained Embedding Layer	56.8	0.676
5,000	Embedding Layer	80.4	0.460
5,000	Pretrained Embedding Layer	54.74	0.687
10,000	Embedding Layer	78.88	0.455
10,000	Pretrained Embedding Layer	79.48	0.4717
20,000	Embedding Layer	79.7	0.488
20,000	Pretrained Embedding Layer	78.4	0.459

Expected Outcome

The experiments reveal apparent tendencies in embedding strategy and size of the dataset values when it comes to the model performance in sentiment classification.

In the small training set (100- 1,000 samples) case, the models based on a pretrained embedding are always more effective, in terms of more accurate validation and less loss. This is improved by the fact that the pretrained embeddings are able to integrate semantic and syntactic knowledge acquired by large external corpora to substitute the small quantity of task-specific information.

The larger the size of the training dataset (3000-20000 samples) the smaller the difference between the performance of the two strategies. In pretrained and trainable embeddings, the results are similar, which means that when there is enough data, models can acquire strong and contextually relevant word representations on their own.

The difference between the use of the model, which used pretrained embeddings, trained on 20,000 samples, and the trainable embedding model was nuanced, even though the latter had the highest overall performance. This implies that pretrained embeddings provide the most benefit mostly when data is limited.

Conclusion

- Pretrained embeddings are also highly advised in the case of smaller datasets since they accelerate learning and improve generalization by the models.

- To work with larger datasets, trainable embeddings also perform well and offer more flexibility to the peculiarities of the dataset.

Finally, pretrained and trainable embeddings are a matter of choice, which ultimately depends on the amount of available data, available computational resources, as well as the task-specific requirements.