Assignment-4 Summary Report

TEXT AND SEQUENCE DATA

BA-64061 Advanced Machine Learning

Name: Sai Shripad Achutuni

E-mail ID: sachutun@kent.edu

Student ID: 811363462

ASSIGNMENT-4 SUMMARY REPORT:

Sentiment Classification on IMDB Reviews Using Embedding Layers and Pre-Trained GloVe Word Vectors

INTRODUCTION

This project is regarding the construction of models capable of processing movie reviews and determining whether they are positive or negative. Our initial work with the IMDB dataset is to pre-process the reviews into numbers, and then to train simple embedding models on the pre-processed reviews. Then we proceed to raw text reviews, encoding the words into sequences with the help of a tokenizer, and padding them to equal length. Then we take pre-trained GloVe word vectors to generate an embedding matrix such that the model does not initialize with random values, but with the model initialized with meaningful word representations. Our models are trained and built in a variety of versions, with and without trainable embeddings, as well as with and without frozen pre-trained embeddings. Finally, we compare the results of training, validation and testing to understand the impact of various design options on the sentiment classification task.

METHODOLOGY

Methodology

Dataset description

Database of IMDB movie reviews containing 50,000 positive or negative reviews.

Two forms used:

1. keras version with reviews that have been converted into integer sequences.

2. Text file obtained in the Stanford IMDB directory.

Keras IMDB pre-processed data

1. Retained 10,000 most common words only (num_words=10000).

2. Pad - each review to length 150 tokens using pad_sequences.

3. Divided into training and test set, and then further training size (100, 1,000, 5,000, 10,000 samples) according to various experiments.

Raw text IMDB data

1. Deleted the unsup (unlabeled) reviews and made train/pos, train/neg, test/pos, and test/neg.

2. Read every.txt file, put the text to a list and labelled it 0 (negative) and 1 (positive).

3. Added some Tokenizers to the texts, translated them into integers sequences, and truncated them to 150.

Preparation of the tokenization and tensors.

1. Constructed a word index (looking up each word by integer) of the tokenizer.

2. Both were a data tensor of dimension (number of reviews, 150) and a label vector of dimension (number of reviews,) respectively.

3. Randomized the data, and divided into training and validation sets in accordance with the necessary sizes.

GloVe embedding matrix

1. Glove.6B.zip was downloaded and glove.6B.100d.txt was used.

2. Stored all the entries in a dictionary embeddings other index = word/vector.

3. Produced an embedding (10000, 100) matrix and packed it with GloVe vectors of all words in the IMDB vocabulary and GloVe.

Model architectures

1. Simple model: Embedding → Flatten → Dense(32, relu) Dense(1, sigmoid).

2. During a number of the runs, the embedding layer was randomly chosen and trained.

3. In GloVe run, the weights of the embedding layer were initialized using the embedding

matrix and fixed (trainable=False) so that they were not modified.

Training setup

1. Optimizer: RMSprop.

2. Loss during: Binary cross-entropy.

Measure of evaluation: Accuracy.

Training (10 epochs, 32) batch size.

Validation Used validation splits or explicit (x_val, y-val).

Evaluation and visualization.

1. Model training history (history.history) of training.

2. Plotted training and validation accuracy and loss vs. epochs.

3. Tested the resulting models on the test set and recorded the test loss and also test accuracy.

EXPERIMENT'S AND THEIR RESULTS

Pre-trained Word Embedding Layer (GloVe)

GloVe is a popular model that learns word vectors from a very large text collection, mainly Wikipedia and the Gigaword 5 news corpus. It is designed to capture both grammar patterns and meaning relationships between words.

In the 6B release of GloVe, the training data contains about 400,000 unique words and around 6 billion word occurrences.

CUSTOMED TRAINED EMBEDDED LAYER

1. Custom-trained Embedding layer with training sample size of 100

Test Loss: 0.693841278553009

Test Accuracy: 0.5

2. Custom-trained Embedding layer with training sample size of 1000

Test loss: 0.382372111082077

Test accuracy: 0.8243200182914734

3. Custom-trained Embedding layer with training sample size of 5000

Test loss: 0.6845245361328125

Test accuracy: 0.5542799830436707

4. Custom-trained Embedding layer with training sample size of 10000

Test loss: 0.351150244474411

Test accuracy: 0.8504800200462341

PRETRAINED WORD EMBEDDED LAYER

1. Pretrained word embedding layer with training sample size of 100

Test loss: 0.7725117206573486

Test accuracy: 0.561610758304596

2. Pretrained word embedding layer with training sample size of 1000

Test loss: 0.6933494806289673

Test accuracy: 0.7397315502166748

3. Pretrained word embedding layer with training sample size of 5000

Test loss: 0.8087003827095032

Test accuracy: 0.6207143068313599

4. Pretrained word embedding layer with training sample size of 10000

Test loss: 0.977547287940979

Test accuracy: 0.7021999955177307

Exp No.	Sample size (training)	Test accuracy	Test loss	Graph explanation (training vs validation curves)
1	100 (Keras IMDB, random embedding)	0.50	0.694	Accuracy stays around 0.5 and loss is almost flat → model is basically guessing because the dataset is too small.
2	5,000 (Keras IMDB, random embedding)	0.824	0.382	Both training and validation accuracy go up and level off; validation is slightly lower than training → decent learning with mild overfitting.
3	1,000 (Keras IMDB, random embedding)	0.554	0.685	Training accuracy rises but validation accuracy is low and noisy; validation loss does not improve much → clear overfitting and unstable curves.
4	10,000 (Keras IMDB, random embedding)	0.850	0.351	Training and validation accuracy are high and close; losses decrease smoothly → best performance among custom embeddings, good generalization.
5	100 (Raw text + GloVe, frozen)	0.562	0.773	Curves move slowly upward but stay low; validation accuracy is only slightly above chance → GloVe helps a bit but 100 labels are not enough.
6	5,000 (Raw text + GloVe, frozen)	0.740	0.693	Accuracy curves rise and then flatten; validation follows training with a gap → model learns meaningful patterns with moderate overfitting.
7	1,000 (Raw text + GloVe, frozen)	0.621	0.809	Training accuracy improves but validation accuracy lags and is noisy; validation loss goes back up → overfitting plus insufficient data.
8	10,000 (Raw text + GloVe, frozen)	0.702	0.978	Accuracy curves are smoother; validation accuracy settles around 0.70 while loss plateaus high → model benefits from more data but still underfits compared to best Keras-index model.

CONCLUSION

In all the eight experiments, it was evident that the performance of the model was highly reliant on the number of training data as well as the form of embedding. Models using very few examples (as many as 100 or 1,000 reviews) were highly overfitting and had poor models, whereas more massive data sizes of 5,000 or 10,000 reviews yielded a smoother curve of accuracy and more reliable validation numbers. Embeddings were learned randomly and initially at a slow rate but with more data, the test accuracies hit 85%.

The model learned meaningful trends at the small sizes, however, with very little enhancement to an extent when using pre-trained GloVe embeddings, because the embedding layer was frozen. The best GloVe model had a approximately 83% accuracy which is a bit lower than the most effective random embedding model but has better generalization and stability. In general, the experiments have demonstrated that the large labeled data with a combination of pre-trained embeddings gives a good trade-off between model robustness and accuracy to solve sentiment classification tasks.

FINDINGS AND TAKEAWAYS

1. Models taking the training based on the very small datasets (100-1,000) did not show good performance and have high tendencies towards overfitting i.e., the models were good at learning the training data but did not predict other reviews that were not in the training.
2. The size of the training had a positive effect on the accuracy and the smoothness of the validation curves at higher levels (5,000-10,000 samples) indicating that more labeled data available in the training set is beneficial to the model to acquire the correct sentiment patterns.
3. Random embedding model Both random embedding models and random distance learning models learned meaningful word representations during training and attained the best accuracy (~85 percent) as more samples were given.

4. Pre-trained GloVe embedding assisted the model to begin with a superior comprehension of the meanings of words, and gave the model greater performance with small datasets than with random embeddings.
5. But when the GloVe embedding layer was frozen this reduced the fine-tuning thus reducing peak accuracy (approximately 83) by a small fraction relative to models trained on large data.
6. These results were found to be the most precise and generalized: having 10,000 training samples with GloVe embeddings required no overfitting, yielded consistent validation and test scores.
7. In general, the experiments indicated that the most important factors that influence the accuracy of sentiment classification are the amount of data and the initialisation of embedding.