AI6127 – Assignment # 1

Reinelle Jan Bugnot
G2304329L

We begin with the IMDB dataset with a train-val-test split resulting in 17,500 training samples, 7,500 validation samples, and 25,000 test samples. A starting RNN model is provided, utilizing randomly initialized embedding weights before the RNN, and a fully connected layer to perform classification.

Part 1 (Task 2)

The first experiment involves testing different optimizers, namely SGD, Adam, and Adagrad. The learning rate for all optimizers was fixed to 0.001 and then trained on 20 epochs. Other optimizer-specific parameters was kept to their default values. The results of training and testing are presented in Fig. 1 and Table I, respectively.

TABLE I. Test Results for base RNN model trained for 20 epochs, using 3 different optimizers (Adam, Adagrad, SGD).

	adam	**adagrad**	**SGD**
Test Loss	**0.556**	0.670	0.682
Test Acc	**74.19%**	58.23%	56.02%

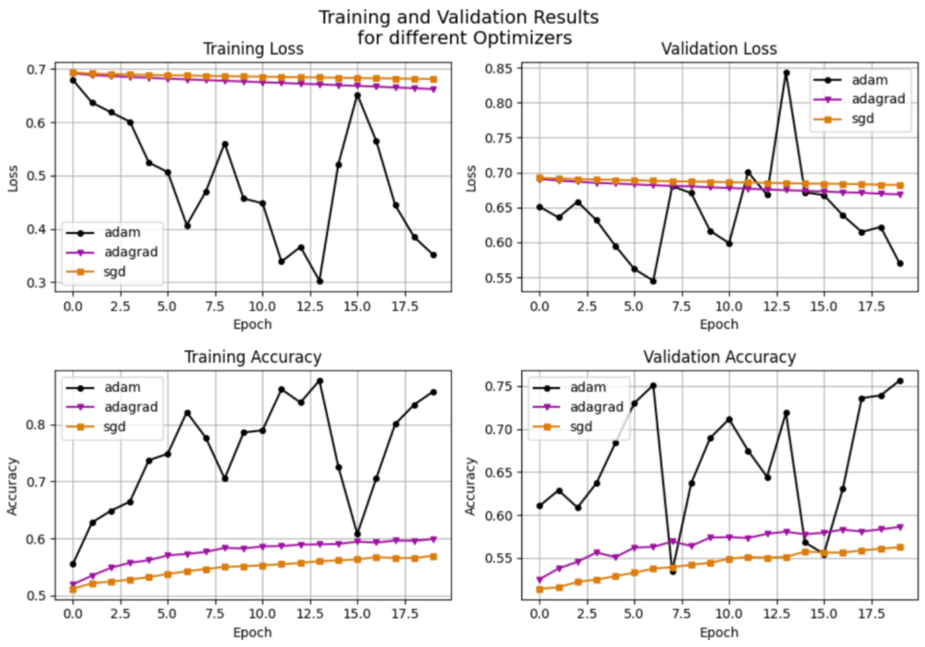

Fig. 1. Training and Validation Loss and Accuracy for base RNN model trained for 20 epochs, using 3 different optimizers (Adam, Adagrad, SGD).

Table I clearly shows that Adam outperformed both Adagrad and SGD by a significant margin. This trend is also evident in the training and validation loss and accuracy graphs, where Adam achieved the lowest loss and highest accuracy among the three optimizers. This superiority of Adam is attributed to its ability to compute learning rates adaptively, enabling faster convergence compared to SGD and Adagrad. These findings suggest that the Adam optimizer is the most suitable choice for training, validation, and testing in this particular task.

Part 2 (Task 3)

The next task involves training the base RNN model across different number of epochs; namely, 5, 10, 20, and 50. The optimizer used for all instances is Adam with a learning rate of 0.001. The result of this experiment is shown in Fig. 2 and Table II.

TABLE II. Test Results for base RNN model trained for varying number of epochs.

	5 epochs	**10 epochs**	**20 epochs**	**50 epochs**
Test Loss	0.606	0.597	0.593	**0.554**
Test Acc	68.06%	69.41%	72.95%	**73.38%**

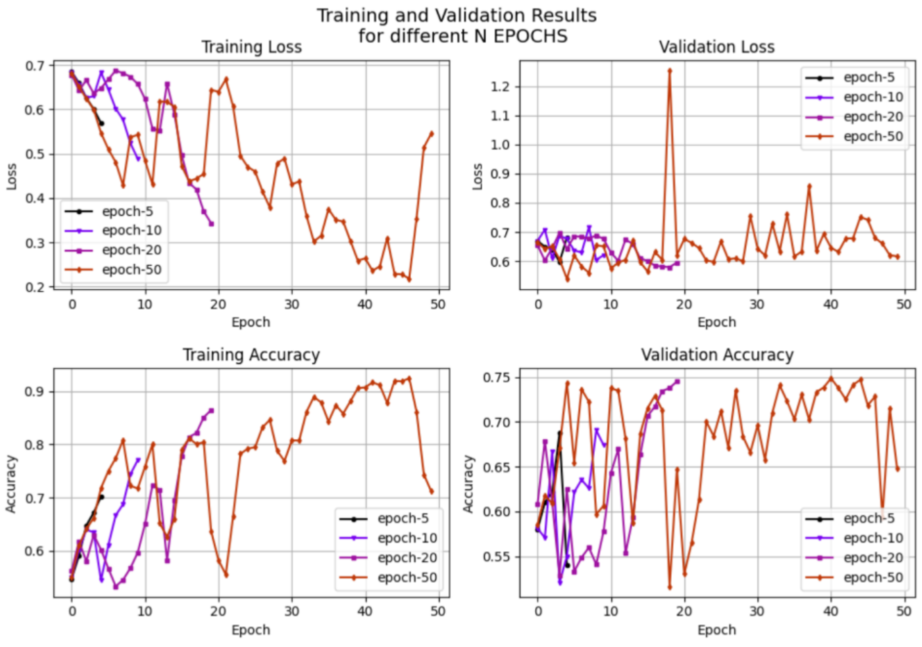

Fig. 2. Training and Validation Loss and Accuracy for base RNN model trained for varying number of epochs.

Fig. 2 illustrates the impact of different numbers of epochs on training and validation results. Generally, we observe a decreasing trend in training loss and an increasing trend in training accuracy as the number of epochs increases, indicating that the model learns from the training data over time. However, there is no improvement in validation loss and accuracy with increasing epochs, indicating poor generalization of the model on unseen data, which is a clear indication of overfitting. This is further validated by the test results on Table 2, where the best test results returned around 73% accuracy, compared to the 90%+ accuracy achieved during training.

Part 3 (Task 4)

The next experiment involves testing the impact using pretrained word2vec embeddings vs using random weights initialization. Both RNN models are trained for 50 epochs with Adam optimizer set at a learning rate of 0.001. All other parameters are set constant and the same between the two models. The results are then shown in Fig. 3 and Table III.

TABLE III. Test Results for base RNN model with random embedding weights initialization vs an RNN model with pretrained word2vec embeddings.

	RNN	RNN + w2v
Test Loss	**0.554**	1.244
Test Acc	**73.38%**	62.65%

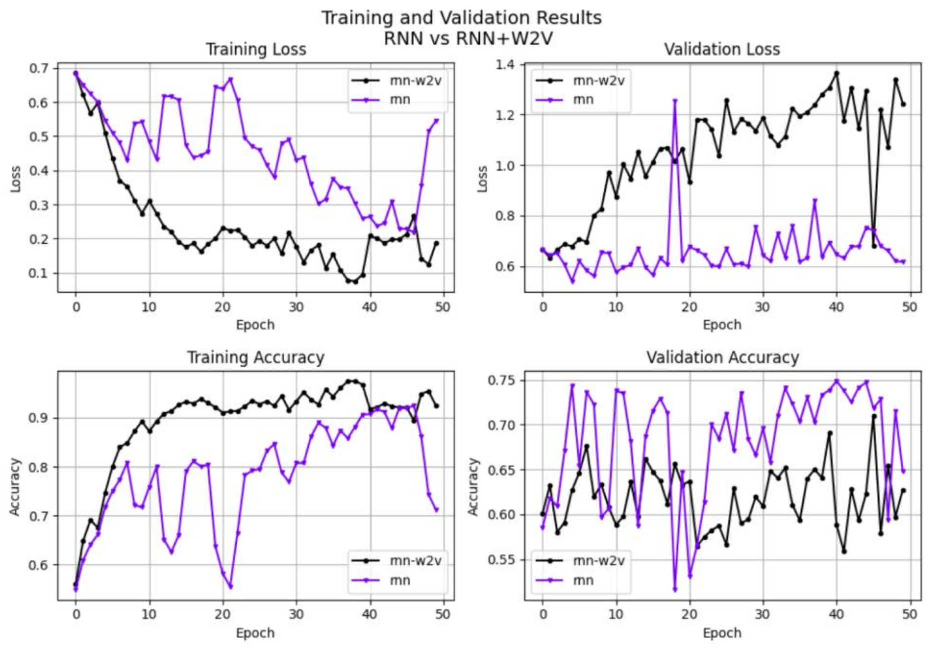

Fig. 3. Training and Validation Loss and Accuracy for base RNN model with random embedding weights initialization vs an RNN model with pretrained word2vec embeddings.

Interestingly, the model initialized with pretrained word2vec embeddings (*rnn-w2v*) exhibits superior training performance based on the training loss and accuracy charts compared to the randomly initialized model (*rnn*). However, on validation and testing, the rnn model outperforms *rnn-w2v*. This shows that the *rnn-w2v* model, while initially learning faster and showing promising results during training, may not be as effective at generalizing to new data as the *rnn* model. This could indicate that the pretrained embeddings in *rnn-w2v* lead to a higher degree of overfitting, making it less adaptable to the validation and test datasets. On the other hand, the *rnn* model's ability to learn from scratch appears to contribute to a more robust representation, enabling it to maintain better performance on unseen data.

Part 4 (Task 5)

The final task is to test the performance of several models with randomly initialized embeddings, trained for 50 epochs with Adam as the optimizer, set at a learning rate of 0.001. The models to be tested are as follows:

1. A one-layer feedforward neural network with a hidden dimension of 500 (ffnn-1)
2. A two-layer feedforward neural network with a hidden dimension of 500 and 300 (ffnn-2)
3. A three-layer feedforward neural network with a hidden dimension of 500, 300, and 200 (ffnn-3)
4. A CNN model with 3 feature maps of sizes 1, 2, and 3 (cnn)
5. An LSTM model (lstm)
6. A Bi-LSTM model (bi-lstm)

The results of training and testing is shown in Fig. 4 and Table IV.

TABLE IV. Test Results for different benchmark models.

	FFNN-1	**FFNN-2**	**FFNN-3**	**CNN**	**LSTM**	**Bi-LSTM**
Test Loss	**0.352**	0.918	1.203	0.523	1.146	0.657
Test Acc	85.53%	84.98%	85.23%	**89.58%**	85.05%	83.91%

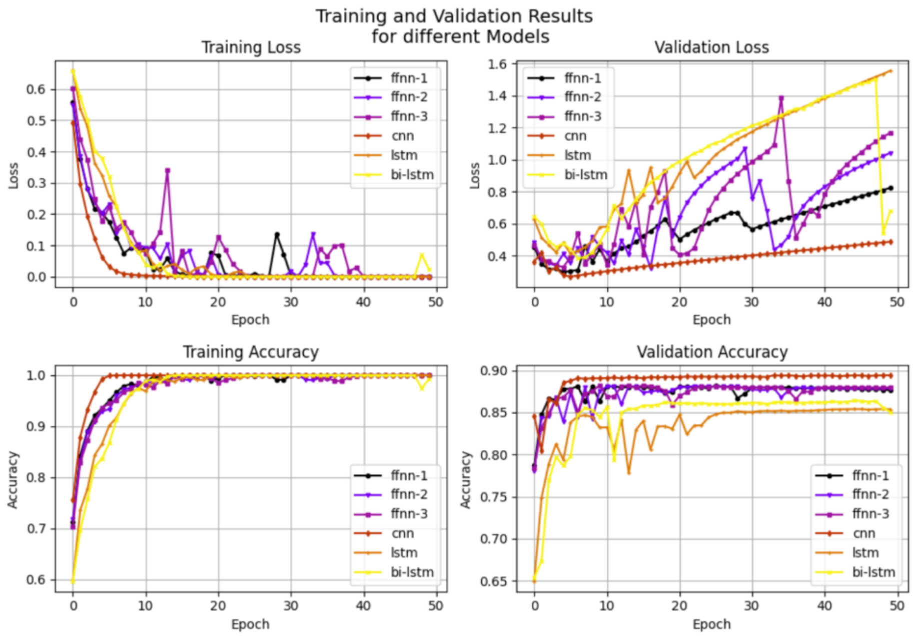

Fig. 3. Training and Validation Loss and Accuracy for different benchmark models.

Compared to the previous RNN models, all six models in this experiment achieved near-zero training loss and near-100% training accuracy. However, the validation loss for all models exhibits an increasing trend after around 10 epochs, indicating overfitting. This increasing trend in validation loss suggests that while they are capable of learning the training data very well, their ability to generalize is compromised beyond a certain point. This could be addressed in future experiments by implementing early stopping or more sophisticated regularization techniques to prevent overfitting and improve generalization on unseen data.

Among all the models tested, the CNN model achieved the lowest validation loss and highest validation and test accuracy, outperforming the rest. This can possibly be attributed to the CNNs capability to use multiple feature maps to learn and represent different features simultaneously. This is beneficial in sentiment analysis as it allows the model to capture a variety of different linguistic features that could be indicative of sentiment. Furthermore, Unlike RNNs and LSTMs, which can struggle with long sequences due to vanishing or exploding gradients, CNNs are not sensitive to the length of the input sequence because they process each input feature independently. The CNN model's robustness to overfitting, as evidenced by its relatively stable validation loss and high accuracy, makes it a promising candidate for further development and application in sentiment classification tasks.