Assignment 3

Models

Model number	Training data	Test Data	Accuracy and loss
1	40000	5000	(0.94, 0.23)
2	15000	5000	(0.92, 0.27)
3	5000	5000	(0.83, 0.47)
4	500	5000	(0.56, 0.61)

Pretrained word embedding Models.

Model number	Training data	Test Data	Accuracy and loss
1	25000	5000	(0.82, 0.67)
2	12000	5000	(0.41, 0.52)
3	5000	5000	(0.54, 0.71)
4	500	5000	(0.47, 0.69)

Observations

- I noticed that a higher training sample allowed the model a great opportunity to learn about the data, which gave it a higher chance of generalizing efficiently with the test set.
- Adjusting the correct Hyper-Parameters also increased the model's performance.
- With these observations we can see why model 1 and 2 performed the best, they had a higher training sample and allowed to model to learn more from the data.
- Models 3 and 4 also had good performance levels but due to the lower training sample they did not generalize as well as the first two models.
- The best performing model overall was number 1 that had the higher training sample data.

Embedding vs pretrained word embedding

- Overall, the performance of an embedding layer versus a pretrained word embedding is dependent on many factors. The complexity and size of the dataset and the training data which is available for the model. In most circumstances a pretrained word embedding tends to perform better than an embedding layer that has been trained from scratch on a specific dataset, because pretrained embeddings have been trained on large datasets can capture the semantic and syntactic relationships between words that are useful for many natural language processing tasks.

- Generally, the larger the dataset as in the hundreds of thousands then a pretrained tends to work better than the embedding layer.

Observations of an Embedding vs pretrained word embedding model

- Again, we notice that model 1 had the overall better performance with the training data size of 25000.
- The worst performing models (models 2 and 4) were not able to study the underlying patterns and relationships present in the training data, which gave poor results in the test data. This appears to be a case of underfitting as the model was too simple to fully comprehend the complexity of the data.
- A lesson learned from this is to consider the design of the system. Simple may be the better option.
- While a larger training data gives the machine ample opportunity to learn the patterns, it is important to avoid using too large data size and causing the model to also underfit.
- As you run the models across different hyper parameters you can begin to identify the perfect performance spot where the model will neither overfit nor underfit.