| **Natural Language Processing - IMDB Movie Review** | | | | | | | |
|---|---|---|---|---|---|---|---|
| | Description | Hyperparameters | Number of Epochs | Training Loss | Training Accuracy | Test Accuracy | Comments |
| Part 1a | Given model - Word Embedding Layer + Mean Pooling + Fully Connected Layer + Relu + Output Layer | ADAM optimizer with LR=0.001, **Schdeuler: Step_Size = 5, Gamma = 0.1** BatchSize=200, VocabularySize=8000, HiddenUnits=500 | 20 | Final:0.08355 | ~97% | should be around 86-88% | Describe more about the model/results such as why certain hyperparamters were chosen or the effect it had on the accuracy/training time/overfitting/etc. |
| | Same architecture as the given model | ADAM optimizer with LR=0.001, Schdeuler: Step_Size = 5, Gamma = 0.1 BatchSize=200, **VocabularySize=50000**, HiddenUnits=500 | 20 | Final: ~0.012 | ~99% | ~86% | The model reached around 99% training accuracy by 6th epoch. Contrary to the training accuracy, the test accuracy stayed around 86%. We also observed increase in testing loss, while decreasing in training loss. These are classical symptoms of overfitting |
| | Same architecture as the given model | ADAM optimizer with LR=0.001, Schdeuler: Step_Size = 5, Gamma = 0.1 BatchSize=200, **VocabularySize=1000, HiddenUnits=50** | 20 | FInal:0.332338 (Stagnating) | ~86% | ~85% | With vocab_size and the # of hidden_units reduced, I was able to lower the model complexity. The model stagnated in performance (in all metrics) around 6th epoch. The model can benefit from revision that allows for complexity |
| | Word Embedding Layer + Mean Pooling + Fully Connected Layer + ReLU +Droup out+ Fully Connected Layer + ReLU +Output Layer | ADAM optimizer with LR=0.001, Schdeuler: Step_Size = 5, Gamma = 0.1 BatchSize=200, VocabularySize=8000, HiddenUnits=1000 | 30 | Final ~0.0542 | ~98% | ~87.5 | I decided to play around with the architecture. Technically, I know that having more layers would lead to more complex model. I anticipated my model to overfit given its tendencies. I ran the model for 30 epochs, and i observed training accuracy steadily rising towards about 98%. Although, a bit high, i recognize that it didnt grow as fast as when vocab size equaled 50,000. However, the testing loss generally increased as the model continued to trained. This evidently showed overfitting |
| | | | | | | | |
| Part 1b | Given model- Fully connected layer +relu+ output layer | vocab_size = 100,000<br>HiddenUnits = 500<br>ADAM optimizer with LR = 0.001<br>BatchSize = 200 | 20 | Flattened around 0.24 | ~90% | 86% | The training loss and accuracy began platteauing earlier on, and so did the testing accuracy and loss. It seems that the model can benefit from more complexity |
| | Same Architecture | vocab_size = 100,000<br>HiddenUnits = 500<br>ADAM optimizer with LR = 0.001<br>BatchSize = 200<br>Reduced drop out rate to p = 0.1 | 20 | Flattened around 0.22 | ~91% | 86% | I added more complexity by decreasing the dropout rate. I noticed that the training loss decreased slightly, while testing loss and accuracy stablized. We will introduce more complex model for next time |
| | Same Architecture | "vocab_size = 100,000<br>HiddenUnits = 1000<br>ADAM optimizer with LR = 0.001<br>BatchSize = 200<br>Reduced drop out rate to p = 0.1" | 20 | 0.2 and decreasing | ~92% | 86% | I increased the number of hidden units while keeping the drop rate at 0.1. I noticed that the traininig seem to continually falling. Testing accuracy improved incrementally. However, testing accuracy and loss stayed the same as the previous attempt. |
| | Fully connected layer + relu + fully connected layer +relu+ output layer | vocab_size = 100,000<br>HiddenUnits = 500<br>ADAM optimizer with LR = 0.001<br>BatchSize = 200<br>Reduced drop out rate to p = 0.3 | 20 | Flattened around 0.21 | ~91% | 86% | I increased the model complexity by adding one more hidden layer. The drop rate was increased to 0.3 from 0.1. Change in performance of this model due to revision seemed negligible |
| | | | | | | | |
| Part 2a | Embedding + LSTM + Output Layer | vocab_size = 8000<br>HiddenUnits = 300<br>Sequence_length = 100<br>ADAM = 0.001<br>Scheduler = 5 epochs, gamma = 0.1<br>batchsize = 200 | 20 | flattened around 0.23 | ~90% | ~81% | I do not know why but I cannot get the accuracy of 87%. I noticed the testing loss oscilating back and forth around 5th epochs |
| | Same Architecture | vocab_size = 8000<br>HiddenUnits = 300<br>Sequence_length = 300<br>ADAM = 0.001<br>Scheduler = 5 epochs, gamma = 0.1<br>batchsize = 200 | 20 | 0.1 around 9th epoch | ~97% | ~74% | In this case, I increased the model complexity by increasing the sequence_length. I have observed the training accuracy increasing at a fast rate. It reached training accuracy of 90% within 5th epoch, while testing accuracy plateaued around 74%. The model is definitely overfitting |
| | Same Architecture | vocab_size = 8000<br>HiddenUnits = 50<br>Sequence_length = 10<br>ADAM = 0.001<br>Scheduler = 5 epochs, gamma = 0.1<br>batchsize = 200 | 20 | flattened around 0.66 | ~60% | ~59% | We note that by dramatically reducing hidden units and sequence_length, we have reduced the complexity of the model. With such a simple model, it evidently shows that the model is underfitting (where both accuracies and losses flattening out throughout the training) |
| | Custom 3 | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Part 2b | LSTM + Output Layer | vocab_size = 100000<br>num_hidden_units = 500<br>seq_length = 100<br>ADAM = 0.001<br>Scheduler = 5 epochs, gamma = 0.1<br>batchsize = 200 | 20 | 0.29 and decreasing | ~87.15% | ~85% | I know I did not hit the projected accuracy. I am wondering if this is due to the fact that I have a scheduler for every 5 epochs while using ADAM. The training loss seemse to go down every epoch at a decreasing rate. And it does seem to slow down every 5 epochs. I did not have time to try to redo the homework without the scheduler |
| | Same Architecture | vocab_size = 100000<br>num_hidden_units = 50<br>seq_length = 10<br>ADAM = 0.001<br>Scheduler = 5 epochs, gamma = 0.1<br>batchsize = 200 | 20 | 0.62 plateaued | ~65% | ~65% | We note that decreasing num_hidden_units drastically reduces the model to a simpler model. This is evident in the models performance, as both losses and accuracies plateaued. |
| | Same Architecture | vocab_size = 100000<br>num_hidden_units = 800<br>seq_length = 150<br>ADAM = 0.001<br>Scheduler = 5 epochs, gamma = 0.1<br>batchsize = 200 | 13(originally planned for 20) | 0.22 | 90% | 88% | I had to stop the training prematurely around WRITE EPOCH. My origional effort was to increase complexity so that it would overfit. Therefore, I increased the number of hidden units as well as seq length( 800 and 150 respectively). During training, both test and train accuracy stayed closed to each other. They also increased steadily without plateauing. To my surprise, this version of model performed better than the model with the original hyperparameters |
| | Custom 3 | | | | | | |