Name: Stephen Pasco

CSCI S-89B Introduction to Natural Language Processing

Assignment 1

**Problem 1 (15 points)**

Please consider the following example of a Neural Network for text classification found in

`3.6-Classifying\_newswires\_a\_multiclass\_classification\_example.ipynb`:

model = models.Sequential()

model.add(layers.Dense(64, activation='relu', input\_shape=(10000,)))

model.add(layers.Dense(64, activation='relu'))

model.add(layers.Dense(46, activation='softmax'))

model.compile(optimizer='rmsprop',

loss='categorical\_crossentropy',

metrics=['accuracy'])

model.fit(partial\_x\_train,

partial\_y\_train,

epochs=20,

batch\_size=512,

validation\_data=(x\_val, y\_val))

results = model.evaluate(x\_test, one\_hot\_test\_labels)

Use a training set of 7,982 samples and validation set of 1,000 samples to train the network. Identify the optimal number of epochs based on the validation accuracy.

1. Plot the results for training and validation accuracy versus number of epochs.
2. Report the test accuracy of the model when trained with the optimal number of epochs. To Compute the test accuracy, use the remaining 2,246 test examples (please see `3.6-Classifying\_newswires\_a\_multiclass\_classification\_example.ipynb` for details).

Problem 1: SOLUTION:

(a)  
Based on the validation accuracy plot, you can observe that the validation accuracy peaks around 9 epochs before starting to decrease, which indicates overfitting. Therefore, the optimal number of epochs is approximately 9.

A graph of a training and validation accuracy

AI-generated content may be incorrect.

(b)

Changed epochs to 9.

```python  
model = models.Sequential()

model.add(layers.Dense(64, activation='relu', input\_shape=(10000,)))

model.add(layers.Dense(64, activation='relu'))

model.add(layers.Dense(46, activation='softmax'))

model.compile(optimizer='rmsprop',

loss='categorical\_crossentropy',

metrics=['accuracy'])

model.fit(partial\_x\_train,

partial\_y\_train,

epochs=9,

batch\_size=512,

validation\_data=(x\_val, y\_val))

results = model.evaluate(x\_test, one\_hot\_test\_labels)  
```

The results variable contains two values when evaluating the model on the test set:

|  |  |
| --- | --- |
| **Loss** | **Accuracy** |
| 0.9619247317314148 | 0.7787176966667175 |

1. **Loss:** This is the categorical\_crossentropy loss on the test data. A lower loss indicates that the model's predicted probability distribution is closer to the true label distribution for the test examples. In this case (epochs = 9), the loss is approximately 0.962.
2. **Accuracy:** This is the percentage of test examples for which the model correctly predicted the topic. A higher accuracy indicates better performance. In this case, the accuracy is approximately 0.779 or 77.9%.

**Problem 2 (25 points)**

Modify the network in Problem 1 in a way that the test accuracy reported in Problem 1(b) is improved.

1. Plot the results for training and validation accuracy versus number of epochs.
2. Report the test accuracy of the model when trained with the optimal number of epochs.

SOLUTION:

(a)

**Problem 3 (25 points)**

Consider your network from Problem 2 and experiment with various optimizers and corresponding hyperparameters. Please try at least four optimizers and plot the validation/train accuracy in each case. Present the best validation accuracy found in each case as a table indicating considered optimizers and hyperparameters.

Finally, use the remaining 2,246 test examples to compute the test accuracy for your best model. Compare the test accuracy with the ones obtained in Problems 1 and 2.

SOLUTION:

--- (Student’s Solution Here) ---