Step 1:

There are 8982 training samples and 2246 test samples.

Number of samples per class

Class Sample Count

[[ 0 55]

[ 1 432]

[ 2 74]

[ 3 3159]

[ 4 1949]

[ 5 17]

[ 6 48]

[ 7 16]

[ 8 139]

[ 9 101]

[ 10 124]

[ 11 390]

[ 12 49]

[ 13 172]

[ 14 26]

[ 15 20]

[ 16 444]

[ 17 39]

[ 18 66]

[ 19 549]

[ 20 269]

[ 21 100]

[ 22 15]

[ 23 41]

[ 24 62]

[ 25 92]

[ 26 24]

[ 27 15]

[ 28 48]

[ 29 19]

[ 30 45]

[ 31 39]

[ 32 32]

[ 33 11]

[ 34 50]

[ 35 10]

[ 36 49]

[ 37 19]

[ 38 19]

[ 39 24]

[ 40 36]

[ 41 30]

[ 42 13]

[ 43 21]

[ 44 12]

[ 45 18]]

The above result shows that the data is unbalanced.

Similarly, we check for the test data.

Class Sample Count

[[ 0 12]

[ 1 105]

[ 2 20]

[ 3 813]

[ 4 474]

[ 5 5]

[ 6 14]

[ 7 3]

[ 8 38]

[ 9 25]

[ 10 30]

[ 11 83]

[ 12 13]

[ 13 37]

[ 14 2]

[ 15 9]

[ 16 99]

[ 17 12]

[ 18 20]

[ 19 133]

[ 20 70]

[ 21 27]

[ 22 7]

[ 23 12]

[ 24 19]

[ 25 31]

[ 26 8]

[ 27 4]

[ 28 10]

[ 29 4]

[ 30 12]

[ 31 13]

[ 32 10]

[ 33 5]

[ 34 7]

[ 35 6]

[ 36 11]

[ 37 2]

[ 38 3]

[ 39 5]

[ 40 10]

[ 41 8]

[ 42 3]

[ 43 6]

[ 44 5]

[ 45 1]]

Nature of the train data: As we can see below, every sample is expressed as a list of integer values, which are word indexes in the dictionary.

[1,

2,

81,

8,

16,

625,

42,

120,

7,

1679,

1928,

4,

270,

5,

4,

49,

27,

1299,

381,

2,

231,

81,

4,

120,

1343,

13,

2,

6,

343,

10,

2568,

334,

116,

13,

137,

5,

25,

335,

80,

28,

10,

96,

5,

272,

15,

90,

67,

7,

197,

2,

5132,

7,

2450,

2521,

7112,

6870,

9,

2,

573,

1259,

2,

75,

8,

17,

12]

Step 2:

Running the classifyingnewswire notebook as-is gives us the test accuracy of 78.76% (and validation accuracy of 82.80% for 8 epochs). Lets take this accuracy as our baseline and try to improve the model further.

Tried out the following different models :

a. 1 hidden layer (ClassifyingNewswires\_Dense\_One Layer.ipynb)

val accuracy :

Epoch 11/20

7982/7982 [==============================] - 1s 156us/step - loss: 0.2362 - acc: 0.9455 - val\_loss: 0.8230 - val\_acc: 0.8360

test accuracy [0.9163860769433202, 0.7969723954226221]

test accuracy : 79.69%

val accuracy :83.60%

Observations : Accuracy increased by 1% than the baseline. Lets experiment with increasing the number of layers.

Increased the number of hidden layers, but as the number of hidden layers increased, the test accuracy decreased further. I think complex models might be overfitting thus decreasing the validation and test accuracy though training accuracy increases.

eg : with 4 hidden layers , test accuracy decreased to 75.77%

[1.3772751512213255, 0.757791629616745]

Lets try other variations.

b. tanh activation (ClassifyingNewswires\_Dense\_tanh.ipynb)

val accuracy :

Epoch 6/20

7982/7982 [==============================] - 1s 129us/step - loss: 0.4688 - acc: 0.9168 - val\_loss: 0.8549 - val\_acc: 0.8210

test accuracy [0.9275175208824613, 0.7898486198215535]

test accuracy : 78.98%

val accuracy :82.10%

Observations : Accuracy improved slightly than the baseline here too.

c. 256 units (ClassifyingNewswires\_Dense\_More\_Units.ipynb)

val accuracy :

Epoch 4/20

7982/7982 [==============================] - 2s 240us/step - loss: 0.4212 - acc: 0.9099 - val\_loss: 0.8379 - val\_acc: 0.8220

test accuracy [0.9449090764443999, 0.7938557435971546]

test accuracy : 79.38%

val accuracy :82.20%

Observations : Accuracy decreased slightly than the baseline here .

Since the test accuracy increased by increasing the number of units, the number of units was increased further.

d. 512 units (ClassifyingNewswires\_Dense\_More\_Units\_512.ipynb)

val accuracy :

Epoch 7/20

7982/7982 [==============================] - 5s 592us/step - loss: 0.1578 - acc: 0.9549 - val\_loss: 0.8595 - val\_acc: 0.8210

test accuracy [1.0212833917470034, 0.7983081032947462]

test accuracy : 79.83%

val accuracy :82.10%

Observation : 1% increase!

e. 1024 units (ClassifyingNewswires\_Dense\_More\_Units\_1024.ipynb)

val accuracy :

Epoch 15/20

7982/7982 [==============================] - 9s 1ms/step - loss: 0.0761 - acc: 0.9575 - val\_loss: 1.2737 - val\_acc: 0.8180

test accuracy[1.4267499973385438, 0.7880676759212865]

test accuracy : 78.80%

val accuracy :81.80%

Observation : Accuracy Decreased with further increasing the number of units!

Out of 256,512,1024 units, 512 yields the highest accuracy.

Now, with 512 units, other combinations such as using tanh and increasing the number of dense layers were tried.

f. 512 units with tanh activation (ClassifyingNewswires\_Dense\_More\_Units\_512\_tanh.ipynb)

val accuracy :

Epoch 3/20

7982/7982 [==============================] - 5s 568us/step - loss: 0.3446 - acc: 0.9203 - val\_loss: 0.8685 - val\_acc: 0.8170

test accuracy [0.9138838559097195, 0.8063223508459484]

test accuracy : 80.63%

val accuracy :81.70%

Observations : This one is best accuracy, I got. Accuracy increased by approx 2 %.

g. 512 units with tanh activation 3 hidden layers(ClassifyingNewswires\_Dense\_More\_Units\_512\_tanh\_more\_layers.ipynb)

val accuracy :

Epoch 3/20

7982/7982 [==============================] - 4s 562us/step - loss: 0.3708 - acc: 0.9181 - val\_loss: 0.8666 - val\_acc: 0.8150

test accuracy [0.9527861412783765, 0.7902938556904678]

test accuracy : 79.02%

val accuracy :81.50%

Observations : Though the accuracy is better than the baseline, increase in hidden layers decreased the accuracy.

h. 512 units with 4 dense layers (ClassifyingNewswires\_Dense\_More\_Layers\_More\_Units.ipynb)

val accuracy :

Epoch 14/20

7982/7982 [==============================] - 5s 686us/step - loss: 0.0625 - acc: 0.9629 - val\_loss: 1.4731 - val\_acc: 0.8100

test accuracy [1.4318975691060774, 0.7894033837934105]

test accuracy : 78.94%

val accuracy :81.00%

Observations : More layers decreased the accuracy further. Probably complex models have started overfitting the data points.

Trying out different combinations, it was seen that 512 units with tanh yielded the highest test accuracy among the variations.

Step 3: Shallow Machine Learning

Experimented with

1. Logistic Regression and
2. MLP Classifiers.

Metrics used to measure performance :- F1 score

**Variations**: Carried out number of variations with different hyper parameters for two models. Also, used pipelines and Count and TF-IDF vectorizers with different penalties text,pre-processing such as stop-words removal, stemmer etc. All details with comments are listed in the given notebook.

**Observations**:- Logistic regression resulted in better F1 score at 0.81.

Step 4:

I started with very simple RNN architectures of SimpleRNN, GRU, LSTM and Bidirectional RNNs.

Limited each news wire length to 700 (max length = 700) and padded the train and test data.

Vectorised the train and test labels using one hot encoding.

Finally, Built various models to see which one yields the best accuracy.

Dense layer of 46 units and softmax activation was used post RNN layer.

Optimiser Used- rmsprop

loss- categorical\_crossentropy

a. Single layer GRU with 32 units

val accuracy :

Epoch 11/30

6287/6287 [==============================] - 31s 5ms/step - loss: 1.4215 - acc: 0.6066 - val\_loss: 1.7101 - val\_acc: 0.5596

test accuracy[1.7885624602344137, 0.5609973286372258]

test accuracy : 56.09%

val accuracy :55.96%

b. Single layer LSTM with 32 units

val accuracy :

Epoch 27/30

6287/6287 [==============================] - 40s 6ms/step - loss: 0.6013 - acc: 0.8511 - val\_loss: 1.5282 - val\_acc: 0.6764

test accuracy[1.78689185541226, 0.6357969723953696]

test accuracy : 63.57%

val accuracy :67.64%

c. Single layer Simple RNN with 32 units

val accuracy :

Epoch 13/30

6287/6287 [==============================] - 10s 2ms/step - loss: 0.9049 - acc: 0.7937 - val\_loss: 1.7214 - val\_acc: 0.5840

test accuracy[1.9050514481477405, 0.518699910952805]

test accuracy : 51.86%

val accuracy :58.40%

d. Single layer Bidirectional LSTM network with 32 units

val accuracy :

Epoch 26/30

6287/6287 [==============================] - 71s 11ms/step - loss: 0.7616 - acc: 0.8061 - val\_loss: 1.5163 - val\_acc: 0.6690

test accuracy[1.782485910005476, 0.6024042743184369]

test accuracy : 60.24%

val accuracy :66.90%

Observation for a,b,c,d:

By trying the above a,b,c,d models, it was seen that single layer LSTM with 32 units model performed better than others. However, none of them have the accuracies that are near out baseline.

So, I went ahead to experiment a few things(such as increasing the number of layers, the number of units etc.) with it further.

Following models were tried :

e. Single layer LSTM with 32 units with dropout 0.2 and recurrent\_dropout 0.2

val accuracy :

Epoch 30/30

6287/6287 [==============================] - 53s 8ms/step - loss: 0.7643 - acc: 0.8056 - val\_loss: 1.5902 - val\_acc: 0.6646

test accuracy[1.689304114874197, 0.6268922529471102]

test accuracy : 62.68%

val accuracy :66.46%

f. Stacking RNNs 3 LSTM networks with 32 units with dropout 0.2 and recurrent\_dropout 0.2

val accuracy :

Epoch 30/30

6287/6287 [==============================] - 141s 22ms/step - loss: 1.0960 - acc: 0.6922 - val\_loss: 1.8388 - val\_acc: 0.5948

test accuracy[2.0048044679001413, 0.5569902048616248]

test accuracy : 55.69%

val accuracy :59.48%

Observation : Accuracy further decresed by stacking RNNs.

g. Changing dropout - Stacking RNNs 3 LSTM networks with 32 units with dropout 0.3 and recurrent\_dropout 0.6

val accuracy :

Epoch 16/30

6287/6287 [==============================] - 127s 20ms/step - loss: 1.5115 - acc: 0.5752 - val\_loss: 1.8831 - val\_acc: 0.5340

test accuracy[1.932011440197273, 0.533837934158152]

test accuracy : 53.38%

val accuracy :53.40%

Observation: Didn’t help by changing dropout for the stacked RNNs.

h Stacking RNNs increasing number of units and changing dropout

Stacking 3 layered 128 unit LSTMs with dropout 0.3 and recurrent\_dropout 0.6

val accuracy :

Epoch 30/30

6287/6287 [==============================] - 361s 57ms/step - loss: 1.1098 - acc: 0.7067 - val\_loss: 1.6961 - val\_acc: 0.6200

test accuracy[1.7734851947556833, 0.5863757792160327]

test accuracy : 58.63%

val accuracy :62.00%

Observation: Model was made more complex to see if that helps. Though it improved the accuracy, it didnt show amazing results.

i. Stacking RNNs increasing number of units and changing dropout

Stacking 3 layered 128 unit LSTMs with dropout 0.2 and recurrent\_dropout 0.2

val accuracy :

Epoch 23/30

6287/6287 [==============================] - 398s 63ms/step - loss: 0.9916 - acc: 0.7377 - val\_loss: 1.5199 - val\_acc: 0.6393

test accuracy[1.716234741312844, 0.5761353517894966]

test accuracy : 57.61%

val accuracy :63.93%

Observation : Decreased further ! :(

j. Stacking RNNs increasing number of units adding dense layers

Stacking 3 layered 128 unit LSTMs and stacking 2 layers of Dense networks(512 units each)

val accuracy :

Epoch 5/30

6287/6287 [==============================] - 420s 67ms/step - loss: 10.4600 - acc: 0.3510 - val\_loss: 10.4244 - val\_acc: 0.3532

test accuracy[12.71650271190667, 0.21104185218165628]

test accuracy : 21.10%

val accuracy :35.32%

Observation: This has to be the most mind boggling model and by far a weird one seen. Grphs yielded straight lines for both validation and train data over a range of epochs. That means the model did not improve did not learn further.

k. Changing the learning rate of rmsprop.(0.001)

val accuracy :

Epoch 30/30

6287/6287 [==============================] - 51s 8ms/step - loss: 2.1539 - acc: 0.4350 - val\_loss: 2.0769 - val\_acc: 0.4553

test accuracy[2.047542837827416, 0.4919857524487979]

test accuracy : 49.19%

val accuracy :45.53%

Observation: No good :(

As seen from e through k, trying out models with multiple layers, different dropouts and recurrent\_dropouts , changing the number of units did give varying results. But none of them was as good as a simple lstm model with 32 units (point b) in RNNs.

Reuters\_RNNs.ipynb (a - d)

Reuters\_RNNs\_Optimised\_More\_Epochs.ipynb(e - f)

Reuters\_RNNs\_Optimised\_More\_Epochs\_Different\_Combinations.ipynb (g - i)

Reuters\_RNNs\_Optimised\_More\_Epochs\_Different\_Combinations\_1.ipynb(j-k)

Step 5 :

We tried shallow machine learning, RNNs and Dense NNs. Machine learning and RNNs did not yield good accuarcies.Probably RNNs might have started overfitting. Both of the techniques yielded results that were much below the baseline. Many variations were tried out. Dense NNs performed well. Though many variations were tried out and some yielded slightly lower accuracies than the baseline, they did not dip drastically as in the case of other two techniques. The best accuracy acheieved was with the 512 units with tanh activation Dense model.

What I learnt is, for a given problem , we should try out different techniques (as in machine learning/ RNN/ Dense NN). Sticking to one technique might not give the best results even if we try out different variations(as in changing number of units/layers) of it.

Also, I observed that increasing the number of layers of Dense NN /RNN helps to a point, and after that the accuracies get worse as we increase more. There might be overfitting happening in those models.

Also, sometimes simple models yield the best of accuracies amongst other complex models.

Trying out different techniques and variations and understanding the trend is the key.