STAT590B HW2 experiments record

1(a)

Binary classification – 16*16

Optimizer: SGD

Layers	Units list	Learning rate	Dropout?	Best test accuracy
2	64,32	0.001	N	0.7352
4	64,64,32,32	0.001	Y	0.6665
8	64*3, 32*5	0.001	N	0.6419
12	64*3, 32*6, 16*3	0.001	Y	0.5913
16	64*3, 32*9, 16*4	0.001	N	0.5211
2	64,32	0.01	N	0.8116
4	64,64,32,32	0.01	N	0.8189
8	64*3, 32*5	0.01	N	0.8083
12	64*3, 32*6, 16*3	0.01	N	0.8351
16	64*3, 32*9, 16*4	0.01	N	0.7870
2	64,32	0.05	N	0.8527
4	64,64,32,32	0.05	N	0.8483
8	64*3, 32*5	0.05	N	0.8230
12	64*3, 32*6, 16*3	0.05	Y	0.8630
16	64*3, 32*9, 16*4	0.05	Y	0.8513
2	128,64	0.001	Y	0.7136
4	128, 128,64,64	0.001	Y	0.6967
8	128*3, 64*5	0.001	Y	0.5997
12	128*3, 64*6, 32*3	0.001	Y	0.5200
16	128*3, 64*9, 32*4	0.001	Y	0.5200
2	128,64	0.01	Y	0.8311
4	128, 128,64,64	0.01	Y	0.8237
8	128*3, 64*5	0.01	Y	0.8153
12	128*3, 64*6, 32*3	0.01	Y	0.8586
16	128*3, 64*9, 32*4	0.01	Y	0.8028
2	128,64	0.05	N	0.8482
4	128, 128,64,64	0.05	N	0.8410
8	128*3, 64*5	0.05	N	0.8487
12	128*3, 64*6, 32*3	0.05	Y	0.8828
16	128*3, 64*9, 32*4	0.05	N	0.8524

When using the SGD optimizer, it's noticed that the best test accuracy generally increases with higher learning rates.

Additionally, the inclusion of dropouts doesn't seem to significantly affect accuracy.

These findings are substantiated by the training curves for loss and accuracy, and it's worth highlighting that there is no evidence of overfitting (see fig_1).

However, it's worth noting that in cases where the neural network architecture includes a large number of layers, such as 12, and a high number of units, the utilization of dropout can lead to improved accuracy.

Above, the best accuracy achieved was 0.8828 with a network configuration of 16 layers and units distributed as 128*3, 64*6, 32*3. This was accomplished with a learning rate of 0.05 in combination with dropout.

Optimizer: RMSPROP

Layers	Units list	Learning rate	Dropout?	Best test accuracy
2	64,32	0.001	N	0.8615
4	64,64,32,32	0.001	Y	0.8586
8	64*3, 32*5	0.001	N	0.8219
12	64*3, 32*6, 16*3	0.001	N	0.8674
16	64*3, 32*9, 16*4	0.001	N	0.8516
2	64,32	0.01	N	0.8399
4	64,64,32,32	0.01	Y	0.8006
8	64*3, 32*5	0.01	N	0.7910
12	64*3, 32*6, 16*3	0.01	N	0.8351
16	64*3, 32*9, 16*4	0.01	Y	0.8428
2	64,32	0.05	N	0.7988
4	64,64,32,32	0.05	N	0.8483
8	64*3, 32*5	0.05	N	0.8230
12	64*3, 32*6, 16*3	0.05	N	0.8402
16	64*3, 32*9, 16*4	0.05	N	0.8336
2	128,64	0.001	N	0.8689
4	128, 128,64,64	0.001	Y	0.8645
8	128*3, 64*5	0.001	Y	0.8480
12	128*3, 64*6, 32*3	0.001	N	0.8803
16	128*3, 64*9, 32*4	0.001	Y	0.8810
2	128,64	0.01	N	0.8314
4	128, 128,64,64	0.01	N	0.8311
8	128*3, 64*5	0.01	Y	0.7855
12	128*3, 64*6, 32*3	0.01	N	0.8627
16	128*3, 64*9, 32*4	0.01	N	0.8160
2	128,64	0.05	N	0.8340
4	128, 128,64,64	0.05	N	0.8417

8	128*3, 64*5	0.05	N	0.8395
12	128*3, 64*6, 32*3	0.05	N	0.8377
16	128*3, 64*9, 32*4	0.05	N	0.7455

When utilizing the RMSPROP optimizer, different results were observed, with higher accuracy achieved when lower learning rates are used.

So, the best accuracy achieved was 0.8810 with a network configuration of 16 layers and units distributed as 128*3, 64*9, 32*4. This was accomplished with a learning rate of 0.05 in combination with dropout.

In fact, the choice of optimizer doesn't have a profound impact on the highest accuracy achieved. However, when considering both individual accuracy scores and their averages, it becomes evident that the RMSPROP optimizer outperforms the SGD optimizer. Notably, we observed a limited number of accuracy scores below 0.6 and around 0.5 when using the SGD optimizer. In contrast, the RMSPROP optimizer consistently yielded accuracy scores mostly above 0.7.

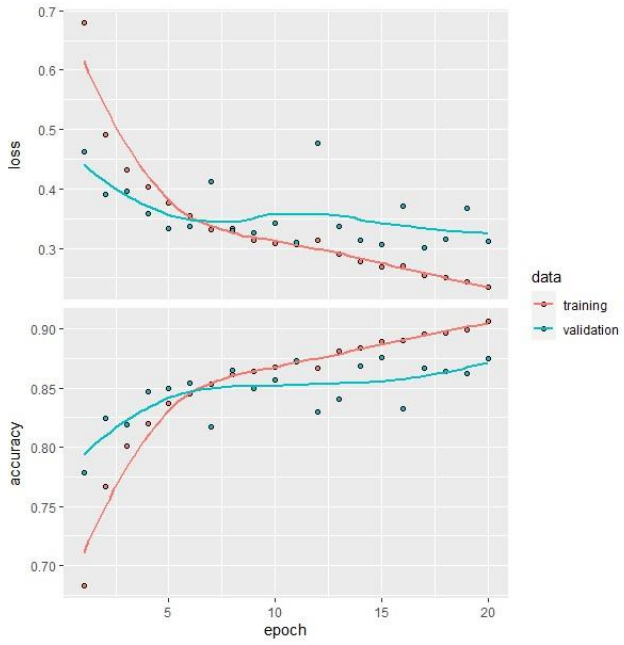

Fig_1 one of curve of loss and accuracy during training

1(b)

Transformed image 2 class classification

Optimizer: RMSPROP

Layers	Units list	Learning rate	Dropout?	Best test accuracy
2	64,32	0.001	Y	0.5222
4	64,64,32,32	0.001	Y	0.5248

8	64*3, 32*5	0.001	Y	0.4804
12	64*3, 32*6, 16*3	0.001	Y	0.5270
16	64*3, 32*9, 16*4	0.001	Y	0.5270
2	64,32	0.01	Y	0.4998
4	64,64,32,32	0.01	Y	0.5351
8	64*3, 32*5	0.01	Y	0.5270
12	64*3, 32*6, 16*3	0.01	Y	0.5270
16	64*3, 32*9, 16*4	0.01	Y	0.5270
2	64,32	0.05	Y	0.5270
4	64,64,32,32	0.05	Y	0.5123
8	64*3, 32*5	0.05	Y	0.5270
12	64*3, 32*6, 16*3	0.05	Y	0.5270
16	64*3, 32*9, 16*4	0.05	Y	0.5270
2	128,64	0.001	Y	0.5402
4	128,64,64,32	0.001	Y	0.5373
8	128*3, 64*5	0.001	Y	0.5263
12	128*3, 64*6, 32*3	0.001	Y	0.5285
16	128*3, 64*9, 32*4	0.001	Y	0.5285
2	128,64	0.01	Y	0.5468
4	128,64,64,32	0.01	Y	0.5380
8	128*3, 64*5	0.01	Y	0.5509
12	128*3, 64*6, 32*3	0.01	Y	0.5501
16	128*3, 64*9, 32*4	0.01	Y	0.5285
2	128,64	0.05	Y	0.5285
4	128,64,64,32	0.05	Y	0.5490
8	128*3, 64*5	0.05	Y	0.5285
12	128*3, 64*6, 32*3	0.05	Y	0.5285
16	128*3, 64*9, 32*4	0.05	Y	0.5443
2	32,16	0.001	N	0.5244
4	32,32,16,16	0.001	N	0.5244
8	32*5, 16*3	0.001	N	0.5244
12	32*9, 16*3	0.001	N	0.5244
16	32*12, 16*4	0.001	N	0.5244
2	32,16	0.01	N	0.5244
4	32,32,16,16	0.01	N	0.5244
8	32*5, 16*3	0.01	N	0.5244
12	32*9, 16*3	0.01	N	0.5244
16	32*12, 16*4	0.01	N	0.5244
2	32,16	0.05	N	0.5244
4	32,32,16,16	0.05	N	0.5244
8	32*5, 16*3	0.05	N	0.5244
12	32*9, 16*3	0.05	N	0.5244
16	32*12, 16*4	0.05	N	0.5244

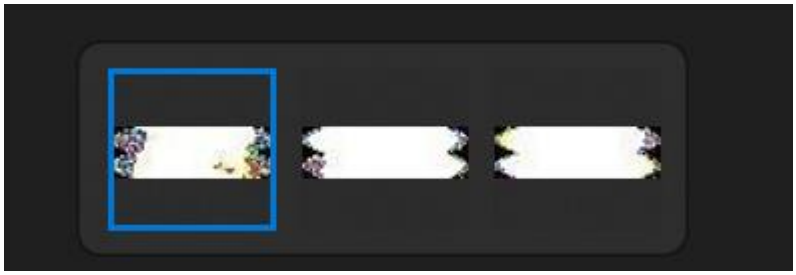

Fig_2 the image of transform on 1(b)

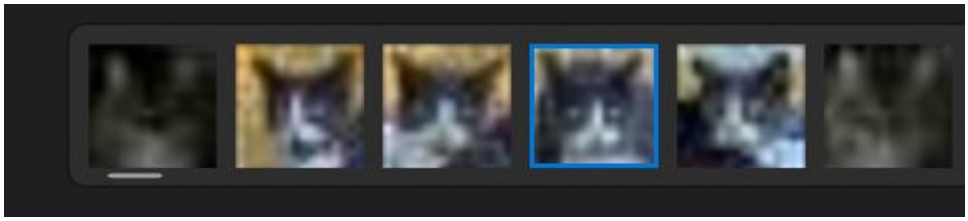

Fig_3 the image of resize on 1(a)

```
> data_trans[4,,,][,,1]
              [,1]         [,2]         [,3]          [,4]          [,5]       [,6]
 [1,]  0.000000000   0.00000000   0.00000000   0.000000000  5.857049e-05   0.310642
 [2,]  0.000000000   0.00000000   0.00000000   0.036092513  1.032525e+01  82.037777
 [3,]  0.000000000   0.02876238  10.75287961  85.027493971  2.492332e+02 447.265310
 [4,]  0.402965769  33.12139447 157.98501985 308.933007674  4.419716e+02 529.948466
 [5,] 18.390635670 140.93546662 264.71322507 382.240260319  4.772562e+02 514.664388
 [6,] 31.628410925 158.87512143 276.53611699 390.155620455  4.797751e+02 521.506285
 [7,]  3.218687908  82.63684233 218.18076284 344.843241470  4.615634e+02 541.866001
 [8,] -0.528772438  -3.37495975  55.04076165 197.666996685  3.569824e+02 505.035556
 [9,] -0.001057123  -0.07052703  -7.54502070  -5.761540910  7.620100e+01 255.977164
[10,]  0.000000000   0.00000000   0.06857295  -0.003640255 -1.161836e+01 -22.898452
[11,]  0.000000000   0.00000000   0.00000000   0.000000000 -1.576475e-03   1.001480
[12,]  0.000000000   0.00000000   0.00000000   0.014041756  3.764074e+00  28.979630
[13,]  0.000000000   0.01117308   3.88438648  29.735928524  8.523477e+01 142.745465
[14,] -0.181827412  11.71282376  53.98193131 102.342522142  1.325255e+02 147.291659
[15,]  2.001203809  47.72164336  86.79366299 113.743690957  1.290200e+02 145.042856
```

The accuracy of the two-class image classification after transformation is quite poor, hovering around 0.5. It's safe to say that it barely captures any useful information, and most accuracy scores are identical. Looks like it makes mesne, since from the fig_2 above is one of three images resulting from the application of the Radon transform and 2D discrete wavelet transforms (DWT) to each of the RGB channels. We extracted the low-frequency (LL) components, and it is challenging to distinguish between a cat and a dog in these components. On the contrary, in Fig_3, the image of the resize on 1(a) can be recognized as a cat.

Additionally, upon examining the resulting array, we observe a significant number of zeros. The maximum value is 5420.857, and the minimum value is -112.1084. These findings suggest that our performance with the transformed data is suboptimal.

So, we did not run the situation when the SGD optimizer.

1(c)

In conclusion, after applying the transformation in 1(b), a significant amount of information has been lost, making classification challenging. Consequently, the performance of 1(a) is notably superior to that of 1(b).

Regarding 1(a), we conducted experiments with varying units for each layer and explored the impact of using dropouts. The results indicate that dropout doesn't have a substantial effect on accuracy. However, it's important to note that in scenarios where the neural network architecture involves a substantial number of layers, such as 12, and a high number of units, employing dropout can result in improved accuracy.

Furthermore, it's worth mentioning that in this dataset, the RMSPROP optimizer consistently outperforms the SGD optimizer.

1(d)

The following results pertain to the accuracy of LDA and SVM. Based on the preceding analysis, it is evident that, after transformation, there is a noticeable loss of information and it becomes challenging to distinguish between them.

LDA: 0.5340

SVM: 0.5960

2(a)

Multiclass classification

Optimizer: RMSPROP

Layers	Units list	Learning rate	Dropout?	Best test accuracy
2	64,32	0.001	N	0.7550
4	64,64,32,32	0.001	N	0.7726
8	64*3, 32*5	0.001	Y	0.7535
12	64*3, 32*6, 16*3	0.001	Y	0.7880
16	64*3, 32*9, 16*4	0.001	N	0.7659
2	64,32	0.01	Y	0.7347
4	64,64,32,32	0.01	N	0.6943
8	64*3, 32*5	0.01	Y	0.6529

12	64*3, 32*6, 16*3	0.01	Y	0.6516
16	64*3, 32*9, 16*4	0.01	N	0.3209
2	64,32	0.05	N	0.3209
4	64,64,32,32	0.05	N	0.3573
8	64*3, 32*5	0.05	N	0.3573
12	64*3, 32*6, 16*3	0.05	N	0.7550
16	64*3, 32*9, 16*4	0.05	N	0.7726
2	128,64	0.001	N	0.7570
4	128,64,64,32	0.001	Y	0.7049
8	128*3, 64*5	0.001	N	0.7320
12	128*3, 64*6, 32*3	0.001	N	0.7639
16	128*3, 64*9, 32*4	0.001	Y	0.7649
2	128,64	0.01	Y	0.6749
4	128,64,64,32	0.01	Y	0.6050
8	128*3, 64*5	0.01	N	0.4183
12	128*3, 64*6, 32*3	0.01	Y	0.5418
16	128*3, 64*9, 32*4	0.01	N	0.3362
2	128,64	0.05	Y	0.4123
4	128,64,64,32	0.05	N	0.4766
8	128*3, 64*5	0.05	N	0.3362
12	128*3, 64*6, 32*3	0.05	N	0.3362
16	128*3, 64*9, 32*4	0.05	N	0.3362
2	32,16	0.001	Y	0.6997
4	32,32,16,16	0.001	N	0.7369
8	32*5, 16*3	0.001	N	0.7054
12	32*9, 16*3	0.001	N	0.7404
16	32*12, 16*4	0.001	N	0.7587
2	32,16	0.01	Y	0.6482
4	32,32,16,16	0.01	Y	0.6293
8	32*5, 16*3	0.01	N	0.5510
12	32*9, 16*3	0.01	N	0.3576
16	32*12, 16*4	0.01	N	0.4907
2	32,16	0.05	N	0.3191
4	32,32,16,16	0.05	N	0.3191
8	32*5, 16*3	0.05	N	0.3191
12	32*9, 16*3	0.05	N	0.3191
16	32*12, 16*4	0.05	N	0.3191

For the three classes, it is evident that the performance is not as strong as that for the two classes. When utilizing the RMSPROP optimizer, it holds true that higher accuracy results are achieved with lower learning rates. Similarly, when using the SGD optimizer, higher accuracy is attained with higher learning rates.

Furthermore, with the RMSPROP optimizer, the best accuracy achieved was 0.7285 using a neural network configuration of 4 layers with units distributed as 128, 64, 64, and 32. This accomplishment was made possible with a learning rate of 0.05 in combination with dropout.

Likewise, with the SGD optimizer, the highest accuracy reached was 0.7880 using a neural network configuration of 12 layers with units distributed as 64*3, 32*6, and 16*3. This achievement was also made possible with a learning rate of 0.05 in conjunction with dropout.

Optimizer: SGD

Layers	Units list	Learning rate	Dropout?	Best test accuracy
2	64,32	0.001	N	0.5155
4	64,64,32,32	0.001	Y	0.4233
8	64*3, 32*5	0.001	Y	0.3499
12	64*3, 32*6, 16*3	0.001	N	0.3576
16	64*3, 32*9, 16*4	0.001	Y	0.3809
2	64,32	0.01	N	0.6568
4	64,64,32,32	0.01	N	0.6516
8	64*3, 32*5	0.01	Y	0.4974
12	64*3, 32*6, 16*3	0.01	Y	0.5264
16	64*3, 32*9, 16*4	0.01	Y	0.3499
2	64,32	0.05	N	0.7218
4	64,64,32,32	0.05	Y	0.6970
8	64*3, 32*5	0.05	Y	0.6710
12	64*3, 32*6, 16*3	0.05	N	0.6960
16	64*3, 32*9, 16*4	0.05	N	0.5800
2	128,64	0.001	N	0.5234
4	128,64,64,32	0.001	N	0.4401
8	128*3, 64*5	0.001	Y	0.3647
12	128*3, 64*6, 32*3	0.001	N	0.3546
16	128*3, 64*9, 32*4	0.001	N	0.3915
2	128,64	0.01	Y	0.6672
4	128,64,64,32	0.01	N	0.6415
8	128*3, 64*5	0.01	Y	0.6013
12	128*3, 64*6, 32*3	0.01	Y	0. 5190
16	128*3, 64*9, 32*4	0.01	N	0.5274
2	128,64	0.05	N	0.7218
4	128,64,64,32	0.05	Y	0.7285
8	128*3, 64*5	0.05	N	0.7235
12	128*3, 64*6, 32*3	0.05	N	0.7064

16	128*3, 64*9, 32*4	0.05	N	0.6509
2	32,16	0.001	Y	0.4726
4	32,32,16,16	0.001	N	0.4733
8	32*5, 16*3	0.001	N	0.3828
12	32*9, 16*3	0.001	N	0.3578
16	32*12, 16*4	0.001	N	0.3491
2	32,16	0.01	N	0.6395
4	32,32,16,16	0.01	Y	0.6283
8	32*5, 16*3	0.01	N	0.4570
12	32*9, 16*3	0.01	N	0.5559
16	32*12, 16*4	0.01	N	0.4138
2	32,16	0.05	Y	0.7039
4	32,32,16,16	0.05	N	0.6777
8	32*5, 16*3	0.05	N	0.6432
12	32*9, 16*3	0.05	N	0.6566
16	32*12, 16*4	0.05	N	0.5594

2(d)

Transformed image 3 class classification

Optimizer: RMSPROP

Layers	Units list	Learning rate	Dropout?	Best test accuracy
2	64,32	0.001	Y	0.3831
4	64,64,32,32	0.001	Y	0.3846
8	64*3, 32*5	0.001	Y	0.3479
12	64*3, 32*6, 16*3	0.001	Y	0.3518
16	64*3, 32*9, 16*4	0.001	Y	0.3514
2	64,32	0.01	Y	0.3761
4	64,64,32,32	0.01	Y	0.4009
8	64*3, 32*5	0.01	Y	0.3823
12	64*3, 32*6, 16*3	0.01	Y	0.3843
16	64*3, 32*9, 16*4	0.01	Y	0.3514
2	64,32	0.05	Y	0.3514
4	64,64,32,32	0.05	Y	0.3722
8	64*3, 32*5	0.05	Y	0.3516
12	64*3, 32*6, 16*3	0.05	Y	0.3732
16	64*3, 32*9, 16*4	0.05	Y	0.3514
2	128,64	0.001	Y	0.3992
4	128,64,64,32	0.001	Y	0.3900
8	128*3, 64*5	0.001	Y	0.3199
12	128*3, 64*6, 32*3	0.001	Y	0.3590
16	128*3, 64*9, 32*4	0.001	Y	0.3590

2	128,64	0.01	Y	0.3893
4	128,64,64,32	0.01	Y	0.3955
8	128*3, 64*5	0.01	Y	0.4009
12	128*3, 64*6, 32*3	0.01	Y	0.4032
16	128*3, 64*9, 32*4	0.01	Y	0.3590
2	128,64	0.05	Y	0.3590
4	128,64,64,32	0.05	Y	0.3461
8	128*3, 64*5	0.05	Y	0.3888
12	128*3, 64*6, 32*3	0.05	Y	0.3828
16	128*3, 64*9, 32*4	0.05	Y	0.3590
2	32,16	0.001	N	0.3526
4	32,32,16,16	0.001	N	0.3526
8	32*5, 16*3	0.001	N	0.3526
12	32*9, 16*3	0.001	N	0.3526
16	32*12, 16*4	0.001	N	0.3526
2	32,16	0.01	N	0.3526
4	32,32,16,16	0.01	N	0.3526
8	32*5, 16*3	0.01	N	0.3526
12	32*9, 16*3	0.01	N	0.3526
16	32*12, 16*4	0.01	N	0.3526
2	32,16	0.05	N	0.3526
4	32,32,16,16	0.05	N	0.3526
8	32*5, 16*3	0.05	N	0.3526
12	32*9, 16*3	0.05	N	0.3526
16	32*12, 16*4	0.05	N	0.3526

Considering the three-class scenario after transformation, it comes as no surprise that the performance is subpar.

2(c)

When dealing with the three-class scenario, we observe similar results as with the two-class scenario. However, it becomes evident that the performance is not as strong when considering three classes compared to two classes. Especially, for transform data, the accuracy is around 0.3-0.4.

2(d)
LDA: 0.4096
SVM: 0.4664

In addition to the work we've previously described in R, our group members have explored alternative approaches:
We began by converting RGB images to grayscale, concatenating them with the Radon

transform, and then feeding the resulting 64x64 grayscale images into a two-layer neural network. The outcomes were as follows:

- For the 2-class problem, our best model achieved a 92% validation accuracy. It was a 2-layer model with 128 nodes in each layer, no dropout, and utilized the SGD optimizer with a learning rate of 0.0001.
- In the case of the 3-class problem, our best model reached an 80% validation accuracy. This model was also a 2-layer architecture with 64 nodes in each layer, included a dropout rate of 0.5, and employed the SGD optimizer with a learning rate of 0.0001.

Additionally, we experimented with the 'image_dataset_from_directory' method for 16x16 images in a 2-class setup. We found that using 8 layers without dropout and a learning rate of 0.001 yielded the best accuracy of 0.8828.