9444 Hw1 report ---z5216355

Part1

1.

```
Train Epoch: 10 [0/60000 (0%)]  Loss: 0.811289
Train Epoch: 10 [6400/60000 (11%)]      Loss: 0.633878
Train Epoch: 10 [12800/60000 (21%)]     Loss: 0.585268
Train Epoch: 10 [19200/60000 (32%)]     Loss: 0.604636
Train Epoch: 10 [25600/60000 (43%)]     Loss: 0.317932
Train Epoch: 10 [32000/60000 (53%)]     Loss: 0.514355
Train Epoch: 10 [38400/60000 (64%)]     Loss: 0.662837
Train Epoch: 10 [44800/60000 (75%)]     Loss: 0.607910
Train Epoch: 10 [51200/60000 (85%)]     Loss: 0.355953
Train Epoch: 10 [57600/60000 (96%)]     Loss: 0.672005
<class 'numpy.ndarray'>
```

```
[[768.   7.   8.   4.  60.   8.   4.  15.   9.   9.]
 [  5. 668.  61.  35.  50.  28.  20.  29.  39.  53.]
 [  7. 107. 689.  59.  78. 123. 147.  28.  94.  84.]
 [ 13.  17.  27. 758.  19.  17.   9.  12.  41.   4.]
 [ 30.  29.  27.  15. 625.  19.  27.  86.   6.  52.]
 [ 65.  24.  21.  57.  21. 724.  25.  16.  30.  30.]
 [  2.  59.  47.  13.  33.  29. 722.  52.  44.  19.]
 [ 61.  13.  35.  17.  38.   8.  20. 623.   7.  30.]
 [ 29.  26.  46.  29.  20.  33.  12.  90. 707.  39.]
 [ 20.  50.  39.  13.  56.  11.  14.  49.  23. 680.]]

Test set: Average loss: 1.0095, Accuracy: 6964/10000 (70%)
```

2.

```
Train Epoch: 10 [0/60000 (0%)]  Loss: 0.410617
Train Epoch: 10 [6400/60000 (11%)]      Loss: 0.267416
Train Epoch: 10 [12800/60000 (21%)]     Loss: 0.217333
Train Epoch: 10 [19200/60000 (32%)]     Loss: 0.195651
Train Epoch: 10 [25600/60000 (43%)]     Loss: 0.133383
Train Epoch: 10 [32000/60000 (53%)]     Loss: 0.298880
Train Epoch: 10 [38400/60000 (64%)]     Loss: 0.236187
Train Epoch: 10 [44800/60000 (75%)]     Loss: 0.323990
Train Epoch: 10 [51200/60000 (85%)]     Loss: 0.160607
Train Epoch: 10 [57600/60000 (96%)]     Loss: 0.285371
<class 'numpy.ndarray'>
```

```
[[843.   6.   8.   4.  42.  10.   3.  18.   9.   8.]
 [  8. 823.  10.  10.  32.  11.  17.  15.  25.  16.]
 [  1.  30. 830.  38.  17.  65.  46.  16.  23.  44.]
 [  6.   4.  44. 910.  11.  13.   9.   4.  45.   7.]
 [ 28.  19.  16.   1. 806.  14.  19.  37.   5.  34.]
 [ 28.   8.  16.  16.   7. 841.   8.   9.  10.   5.]
 [  6.  60.  26.   6.  28.  27. 883.  28.  29.  20.]
 [ 40.   6.  14.   3.  16.   1.   9. 807.   1.  17.]
 [ 35.  17.  24.   6.  20.  14.   1.  30. 846.  14.]
 [  5.  27.  12.   6.  21.   4.   5.  36.   7. 835.]]

Test set: Average loss: 0.5280, Accuracy: 8424/10000 (84%)
```

3.

```
Train Epoch: 10 [0/60000 (0%)]  Loss: 0.021563
Train Epoch: 10 [6400/60000 (11%)]      Loss: 0.002089
Train Epoch: 10 [12800/60000 (21%)]     Loss: 0.005271
Train Epoch: 10 [19200/60000 (32%)]     Loss: 0.001468
Train Epoch: 10 [25600/60000 (43%)]     Loss: 0.009206
Train Epoch: 10 [32000/60000 (53%)]     Loss: 0.008536
Train Epoch: 10 [38400/60000 (64%)]     Loss: 0.002256
Train Epoch: 10 [44800/60000 (75%)]     Loss: 0.008977
Train Epoch: 10 [51200/60000 (85%)]     Loss: 0.002891
Train Epoch: 10 [57600/60000 (96%)]     Loss: 0.003410
<class 'numpy.ndarray'>
```

```
[[945.   4.   8.   2.  24.   4.   5.  10.   5.  11.]
 [  5. 924.   4.   5.   7.   7.   2.   8.  17.   7.]
 [  3.  20. 896.  21.   9.  42.  21.  16.  16.  12.]
 [  0.   0.  28. 947.   4.   4.   1.   1.   7.   6.]
 [ 27.  10.  11.   1. 929.   9.  15.   7.  10.   9.]
 [  3.   0.   7.  11.   4. 909.   4.   2.   6.   1.]
 [  0.  26.  16.   3.   9.  12. 946.   7.   7.   5.]
 [  7.   3.   9.   0.   5.   5.   4. 933.   3.   7.]
 [  5.   4.   6.   4.   7.   2.   0.   2. 923.  10.]
 [  5.   9.  15.   6.   2.   6.   2.  14.   6. 932.]]

Test set: Average loss: 0.3528, Accuracy: 9284/10000 (93%)
```

4.

a. Net lin accuracy : 70%

Net full accuracy : 84%

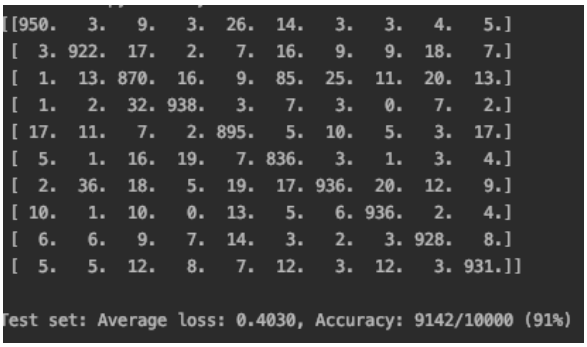
Net Conv : 93%

We can see from the accuracy score that the Net conv is the most and the accuracy score of Net lin is the least. So the model with convolutional layers is the best model of the three. Net lin function cannot behavior well because a layer of neural networks cannot fit complex images. It is in a situation which is underfitting. Although the Net full function is all connection, the ability to fit is also insufficient. And the model with convolution layers is invariant to geometric transformation, deformation and illumination and also can scan the whole image to be detected with a small computational cost.

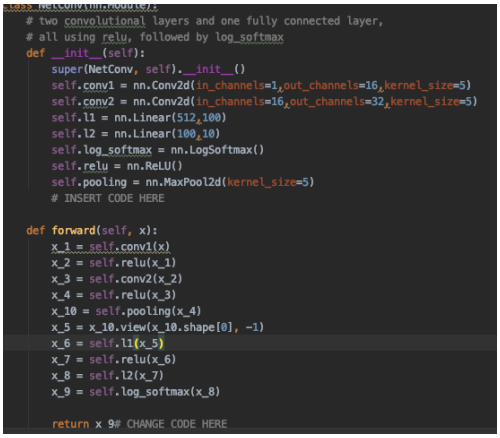
b. the confusion matrix of Net lin

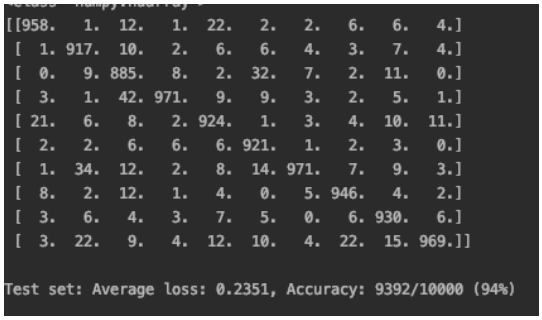
the max number except for the diagonal is 147 in the 7th line, and the character is “ma” similar to the character “su”
the confusion matrix of Net full
the max number except for the diagonal is 65 in the 6th line, and the character is “ha” similar to the character “su”
the confusion matrix of Net Conv
the max number except for the diagonal is 42 in the 6th line, and the character is “ha”, similar to the character “su”
reason for the mistaken is that the handwriting between two characters are similarly.

c. modify parameters
（1）first change the first convolution layer out_channel from 16 to 8 and change the second convolution layer out_channels from 32 to 16, and finally the accuracy change from 93% to 91%, it is because when the convolution layer is input and output channel is low , it will be underfitting.



From changing the parameters we can see that the input and output channel can also decide the accuracy.
（2）secondly , adding a pooling layer, after the second relu layer, before the first linear function ,and after adding the pooling layer ,the accuracy increase to 94%.

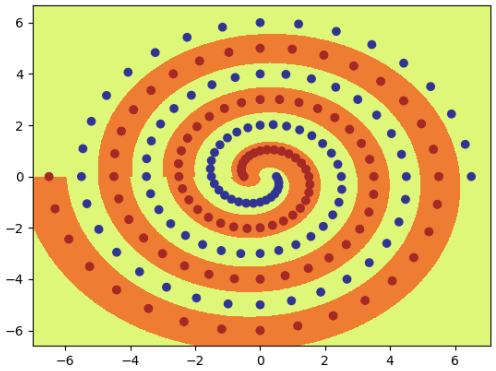




The pooling layer has the advantage of Amplification of receptive field, Increase translational invariance, Make the network easier to optimize.So it makes the accuracy better. So we can see that changing the parameters like input and output channel and adding a pooling layer can makes the accuracy different.

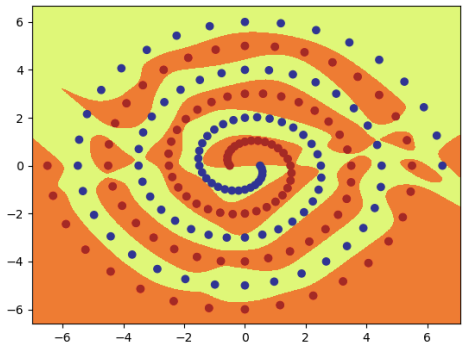
Part2

2. hidden node =10



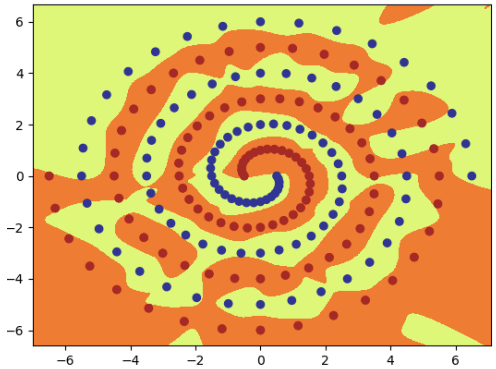
Polar_out.png

4. hidden node =10 initial weights =0.16



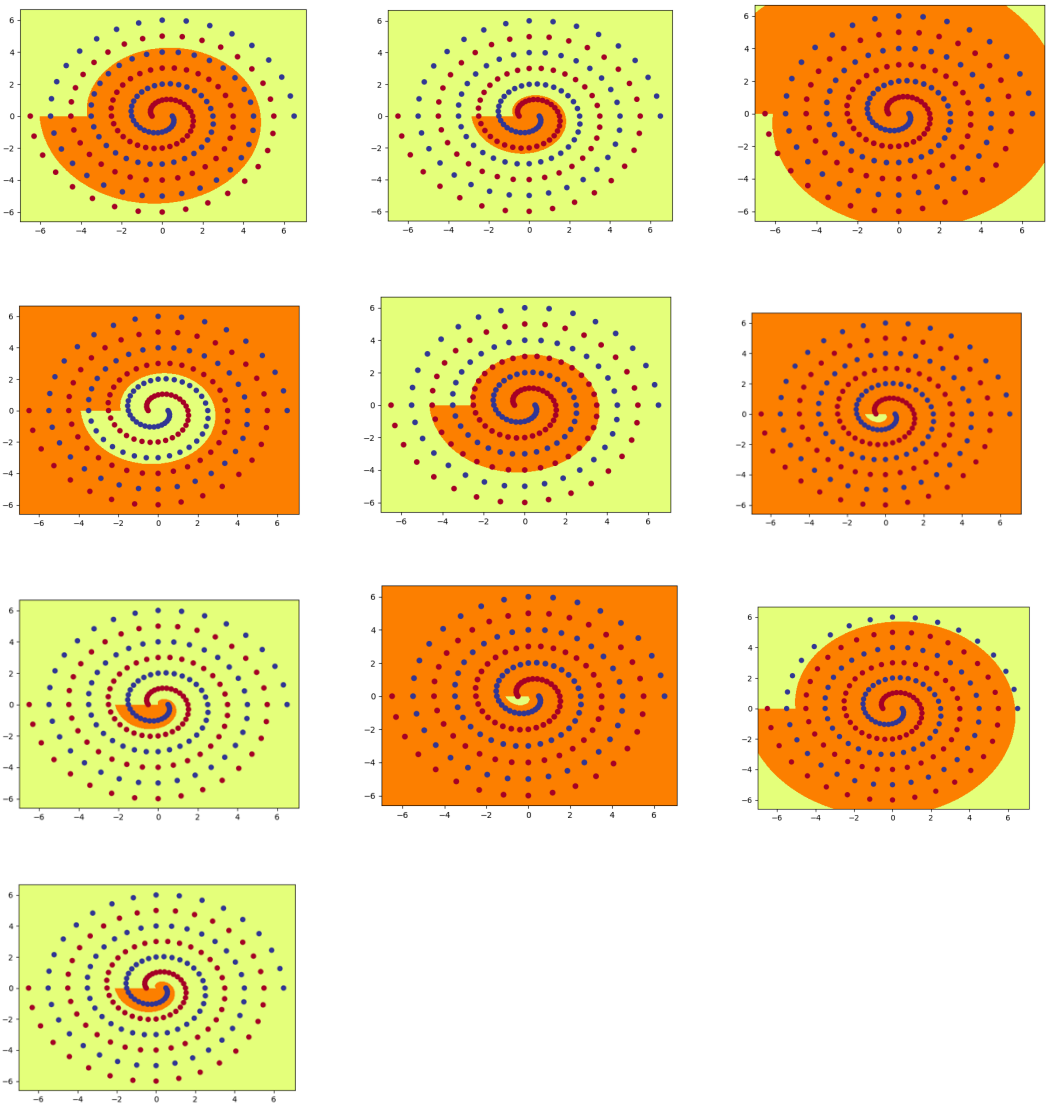
Net_raw.png

6.hidden node =10 initial weights = 0.16



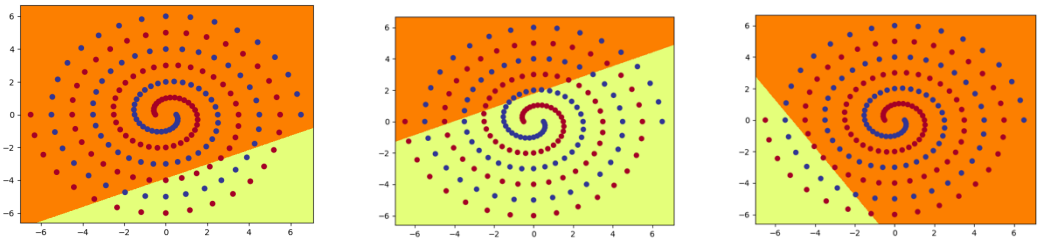
Net_short.png

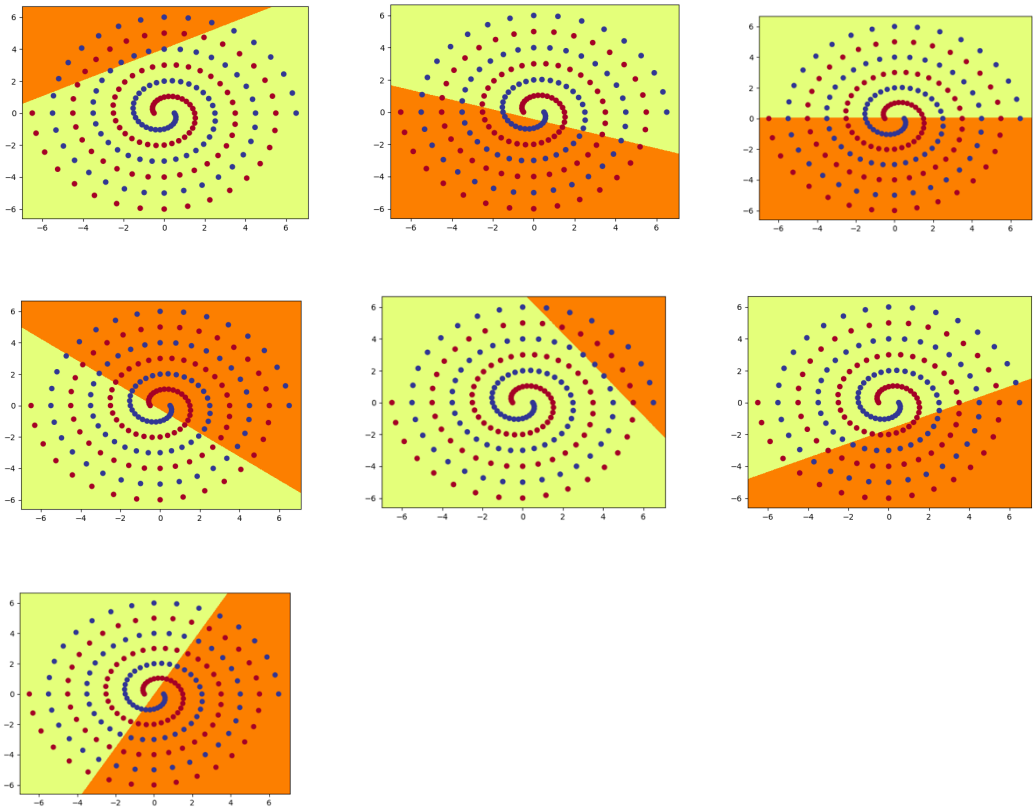
7. PolarNet



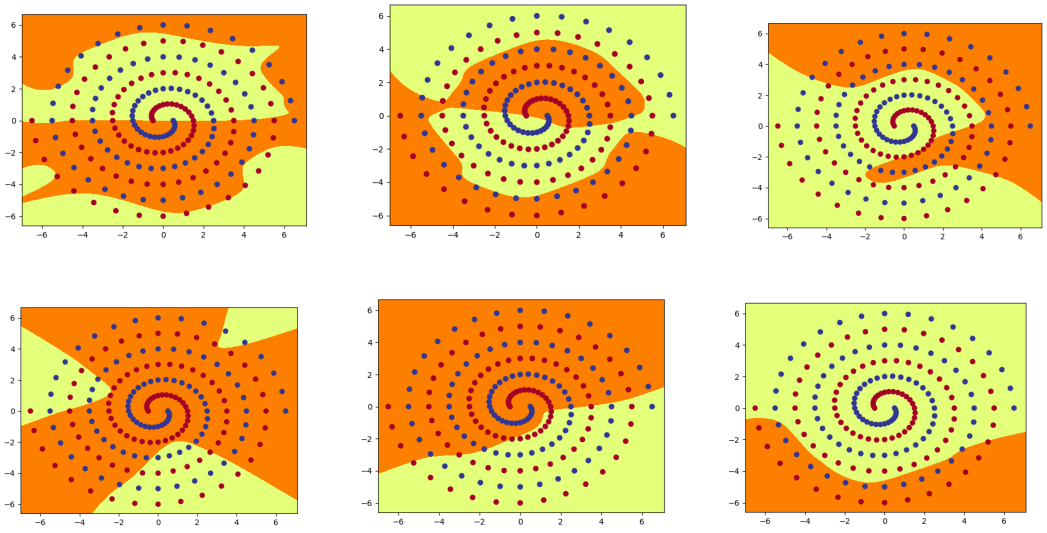
RawNet

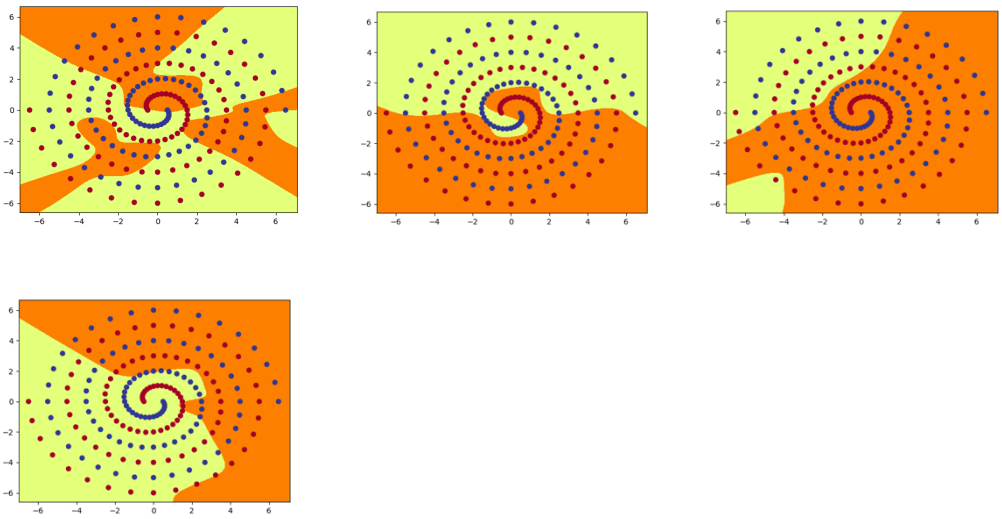
Hidden layer 1





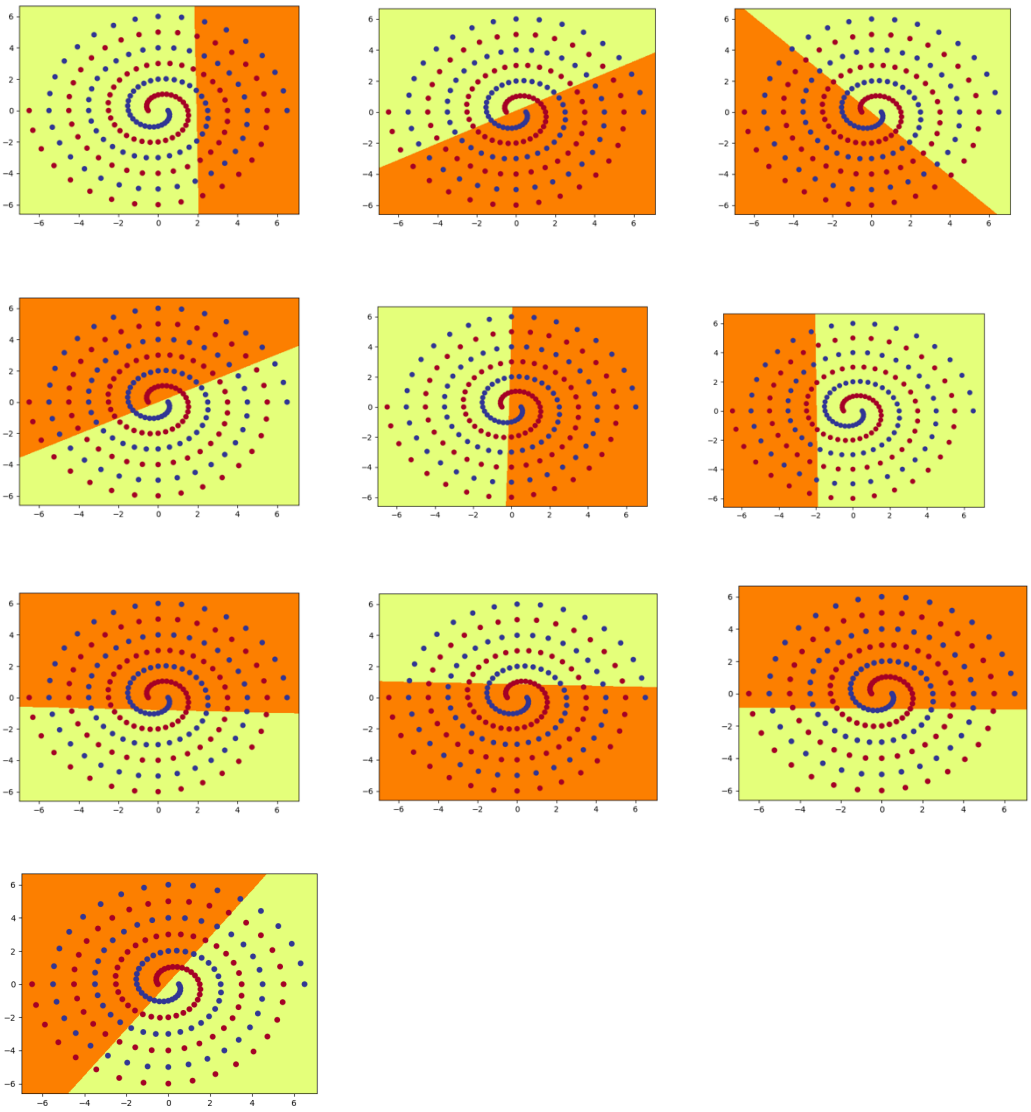
Hidden layer 2



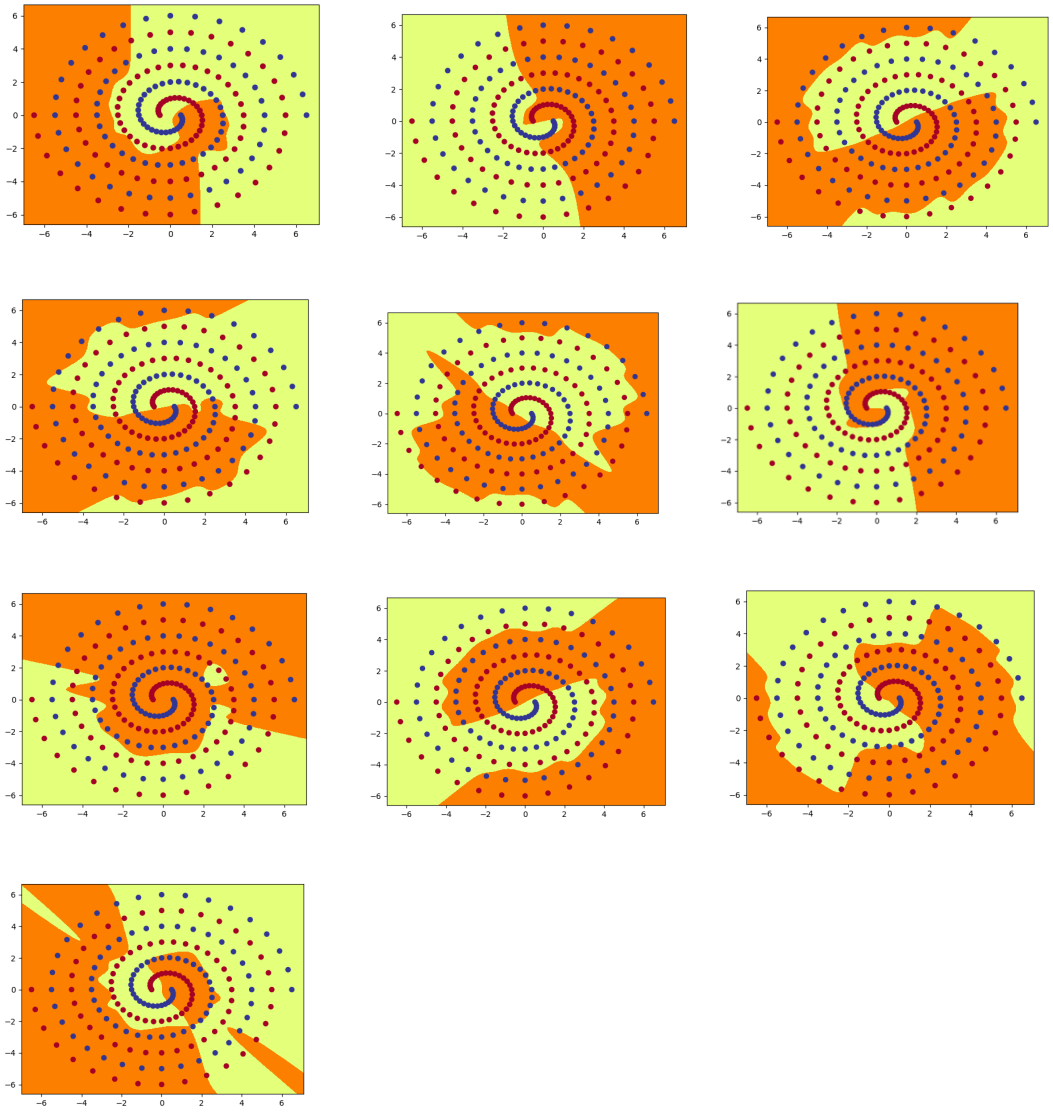


ShortNet

Hidden layer 1



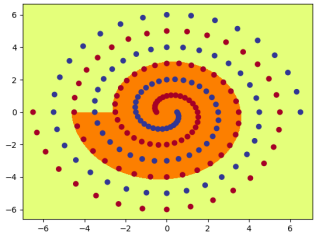
Hidden layer 2



8.

a. We can see from the pictures of the hidden layer 1 and hidden layer 2 that all the hidden layer 1 pictures are straight line, and hidden layers 2 are curve. The final output of raw_net and short_net are the combination of all curve.

Take this picture for example, the first circle is intensive, so it has a large weight, different manage different layer ,color and density.



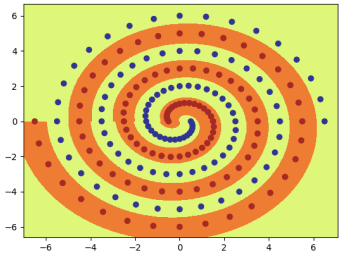
b. net raw init = 0.1 5400 epochs

init = 0.13 6700 epochs
init = 0.14 35500 epochs
init = 0.15 cannot fit within 50000 epochs
init = 0.16 5500 epochs

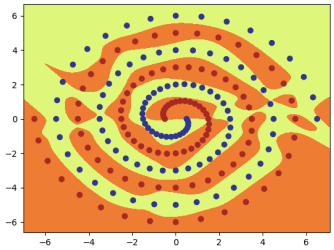
net short init =0.1 3900 epochs
init =0.13 4500 epochs
init =0.14 4100 epochs
init =0.15 1400 epochs
init =0.16 3000 epochs

After the experiment, we can conclude that with different initial value, the success of learning is different, for net raw, the best is when initial value is 0.16 which has the least epochs. And for net short, the best is when initial is 0.15 which has only 1400 epochs. We can conclude that the success of learning has close relationship with initial value. But the speed doesn`t have direct connect with initial value.

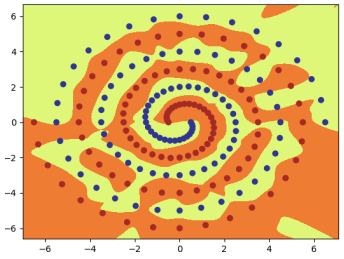
c. comparing output pictures of polar, net raw and net short, we can see that the polar picture is the most naturalness. Although the other two can put all the yellow point into the yellow background and the same with blue. But it is not the effect that we want. Multi-layer neural network must fit better, but generalize function is not enough. Sometimes simpler model can fit better and can prevent overfitting.



Polar_out.png



net_raw.png



net_short.png

d. We use net raw as test

batch size

45: 9400 epochs

97: 37500 epochs

194:10200 epochs

We can see that with the changing of batch size, the effect of fitting is different, when the size is 45, it fit best, about 9400 epochs. And when the batch size is 194, it fit worst.

When using SGD instead of Adam, It doesn't fit at all, when running over 30000 epochs the accuracy still at about 52.00%.So changing SDG from Adam can make the result a different.

When changing the Tanh to ReLu, and the init is 0.13, when it is Relu, it needs 2200 epochs, and when it is Tanh, it needs 4800 epochs, we can see that the effect of Relu is better than Tanh.

From above, changing the batch size, changing from SGD to Adam or changing from Tanh to Relu can make different effect.