Part 1:

1.1

```
<class 'numpy.ndarray'>
[[788.   7.   6.   6.  65.   9.   4.  20.  13.  10.]
 [  5. 679.  61.  32.  48.  31.  18.  28.  34.  47.]
 [ 11. 106. 705.  78.  79. 140. 152.  27.  93.  92.]
 [ 10.  21.  25. 749.  19.  18.  10.  11.  45.   6.]
 [ 28.  30.  30.  17. 637.  21.  29.  97.   7.  62.]
 [ 43.  12.   7.  40.  12. 699.  11.  12.  20.  22.]
 [  2.  59.  44.  14.  32.  28. 738.  64.  46.  24.]
 [ 64.  12.  32.  20.  28.  10.  16. 595.   6.  27.]
 [ 32.  26.  49.  34.  23.  33.  12. 105. 716.  40.]
 [ 17.  48.  41.  10.  57.  11.  10.  41.  20. 670.]]

Test set: Average loss: 1.0080, Accuracy: 6976/10000 (70%)
```

1.2

```
<class 'numpy.ndarray'>
[[823.   5.   6.   3.  75.  10.   4.  15.   9.   4.]
 [  4. 748.  19.  12.  24.  18.  14.  15.  23.  22.]
 [  3.  45. 778.  56.  27. 101.  81.  33.  33.  72.]
 [  8.   7.  33. 872.   9.  24.   9.   5.  49.   3.]
 [ 27.  31.  18.   4. 745.  17.  16.  87.   5.  55.]
 [ 28.   9.   6.  10.   4. 751.   2.   6.   9.   7.]
 [  7.  92.  66.  12.  42.  43. 853.  87.  49.  26.]
 [ 53.   5.  15.   9.  19.   3.  10. 604.   6.  23.]
 [ 35.  21.  28.   7.  25.  21.   4. 100. 806.  14.]
 [ 12.  37.  31.  15.  30.  12.   7.  48.  11. 774.]]

Test set: Average loss: 1.2070, Accuracy: 7754/10000 (78%)
```

1.4

a) Regarding of accuracy of the three models, the accuracy of model three is higher than other models apparently. Clearly, convolutional network could improve the model performance.

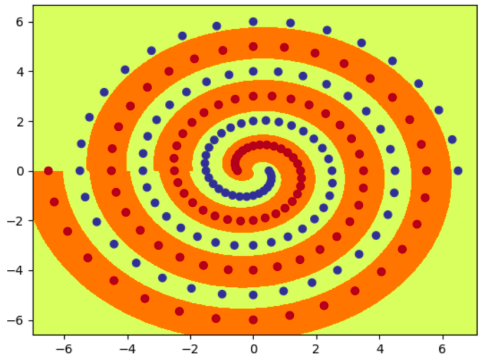
b) As for model one, や 'ya' is hard to recognition. Also, this character is a big problem for model two. This character is similar to another character 'tsu' つ. For the linear function, characters with the same stroke are more difficult to recognize. Model three are most likely to be mistaken for す 'su'. Use the picture as a reference, this character have lots of different type of hand writing in KMNIST Dataset. Therefore, I think is the important reason for mistakes.

c) For model two, I have tried some different number of hidden nodes in full connected layer, I got the best accuracy is 78%;

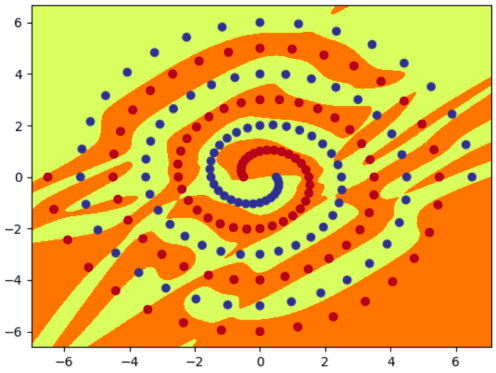
As for modal three, I got some good result by increase the number out_channel for convolution layer 2. At the beginning, I set the out_channel is 20, it is hard to reach the accuracy 93%(always stop at 92% or 91%). However, I got best result when I add the number of out_channel in each layer.

Part 2:

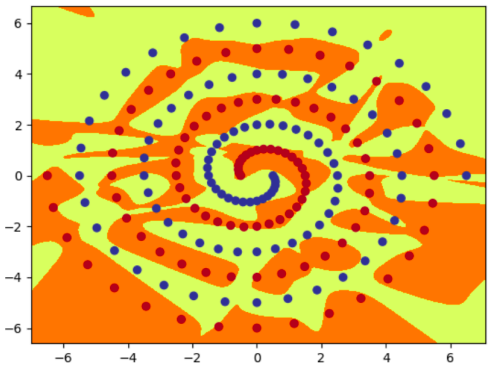
2.2 hidden nodes =6 will help PolarNet to learn all data within 20000 epochs.



2.4 initial weight: 0.3 hidden nodes: 9



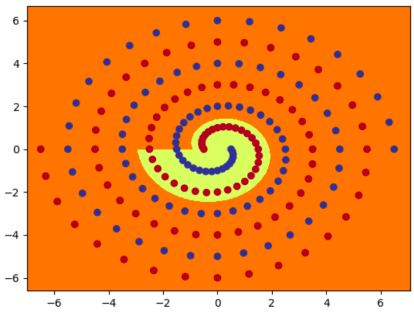
2.6



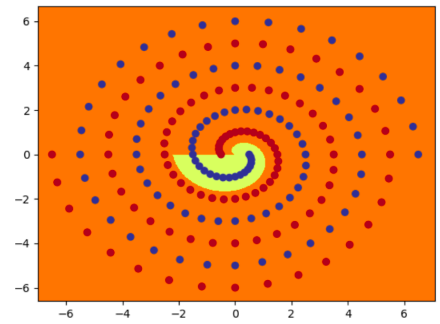
2.7

Polar :

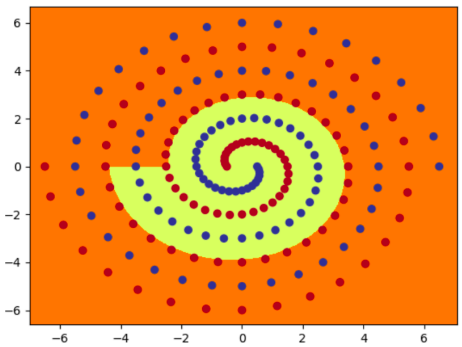
Polar1_0:



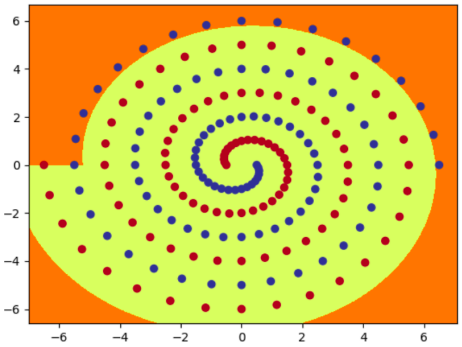
Polar1_1:



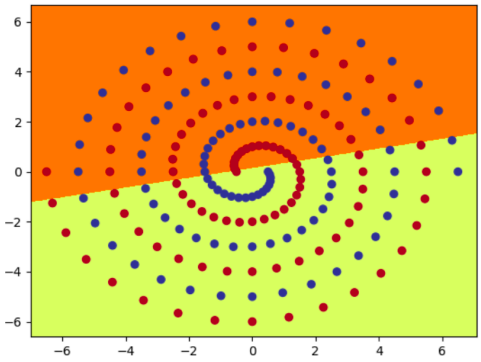
Polar1_2:



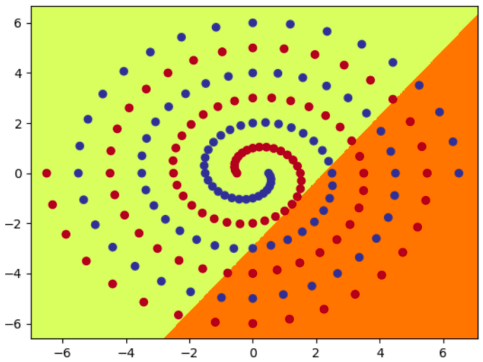
Polar1_3:



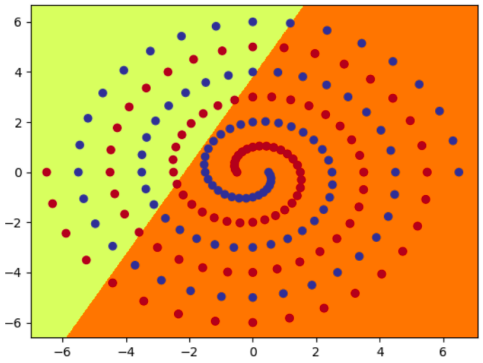
Raw1_0:



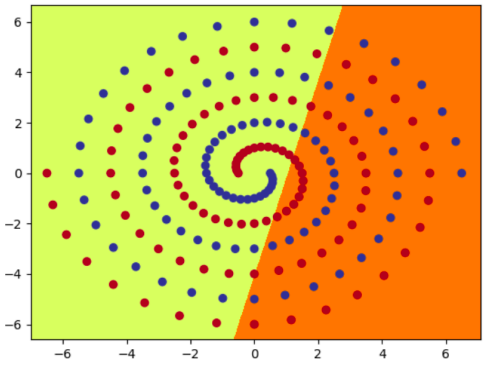
Raw1_1:



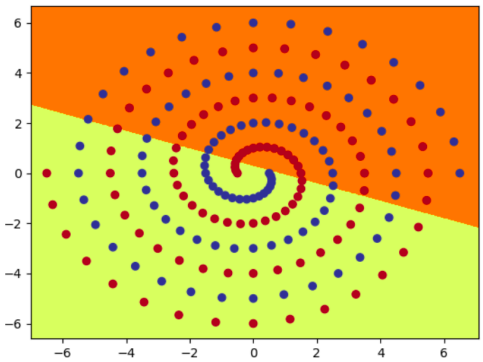
Raw1_2:



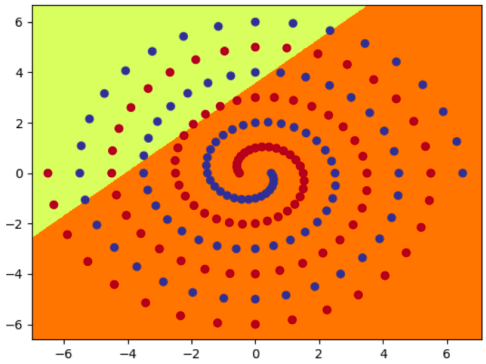
Raw1_3:



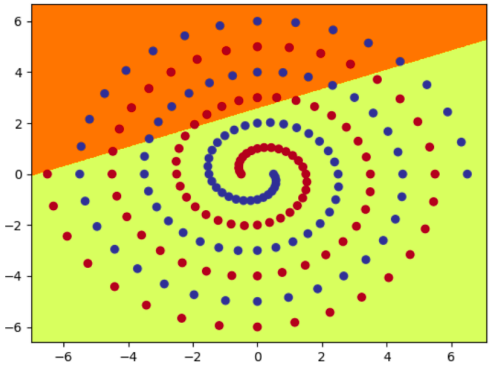
Raw1_4:



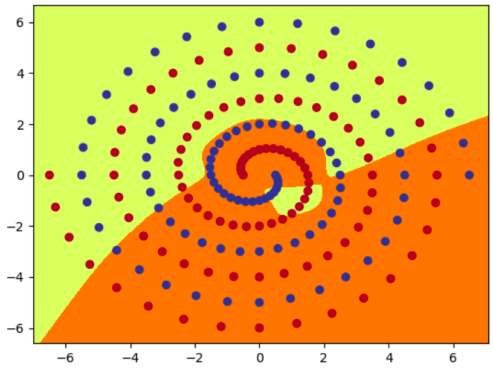
Raw1_5:



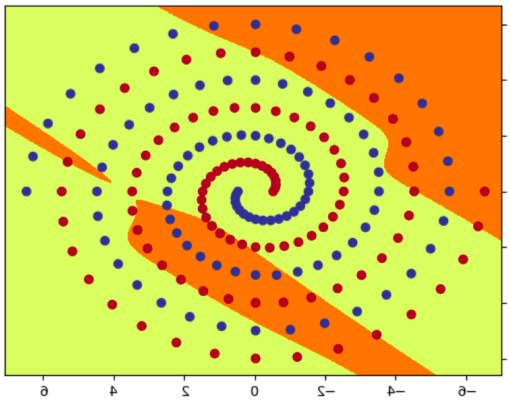
Raw1_6:



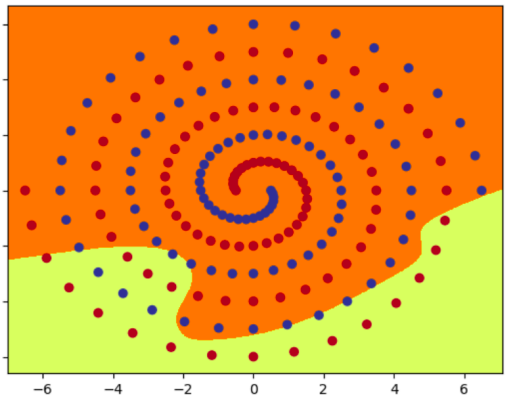
Raw2_0:



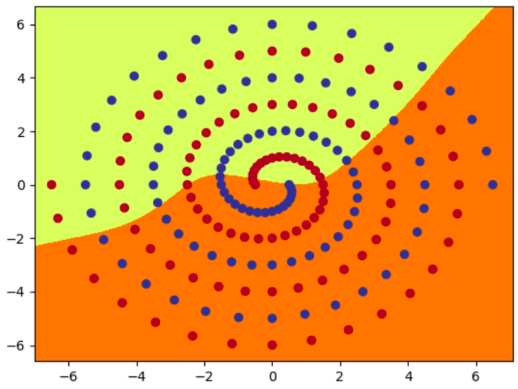
Raw2_1:



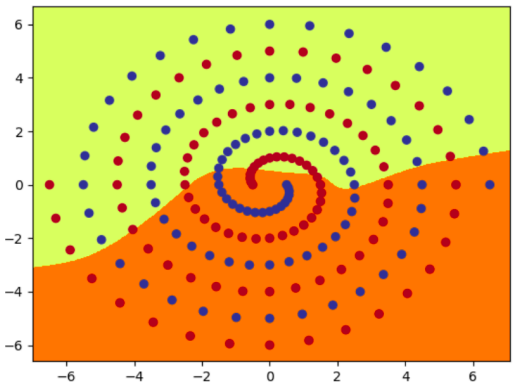
Raw2_2:



Raw2_3:

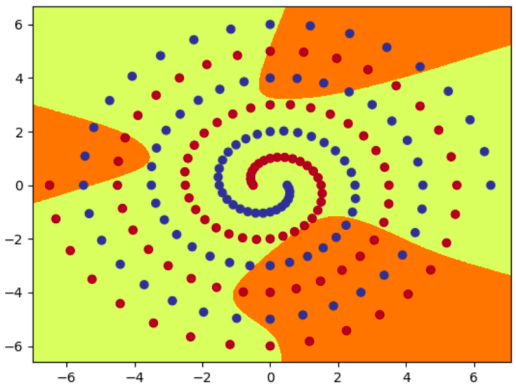


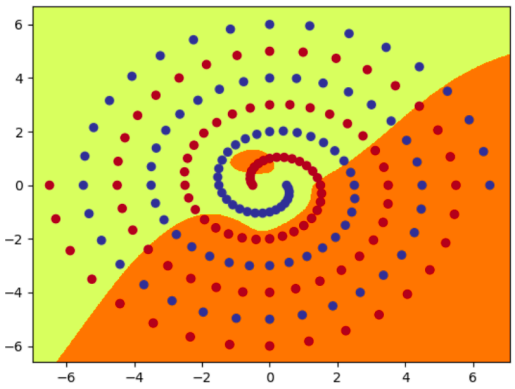
Raw2_4:



Raw2_5:

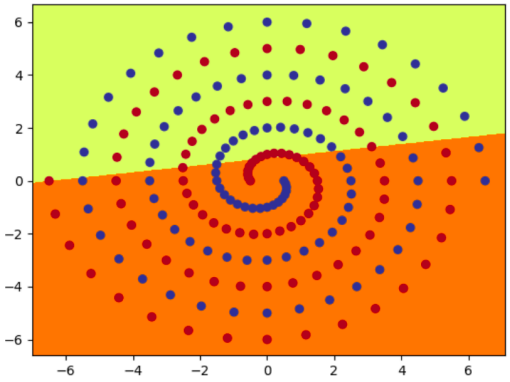
Raw2_6:

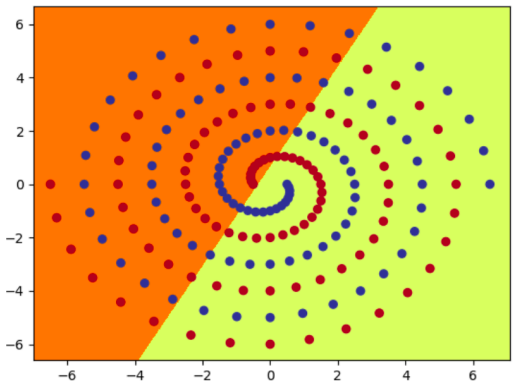




short1_0:

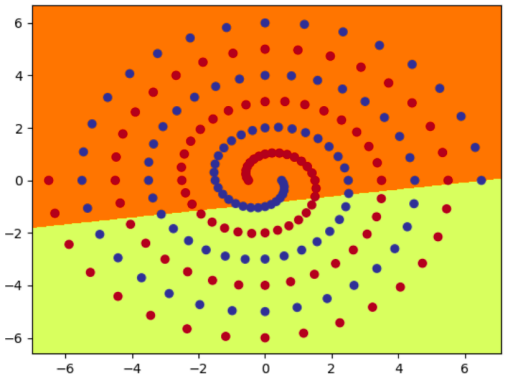
short1_1:

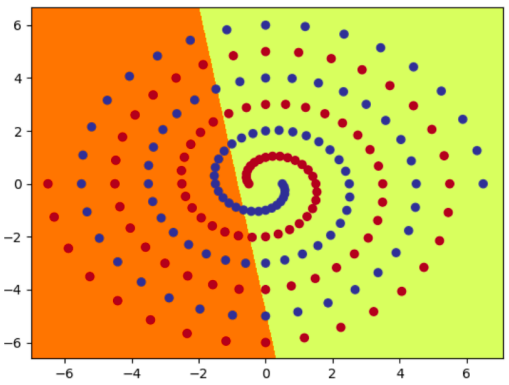




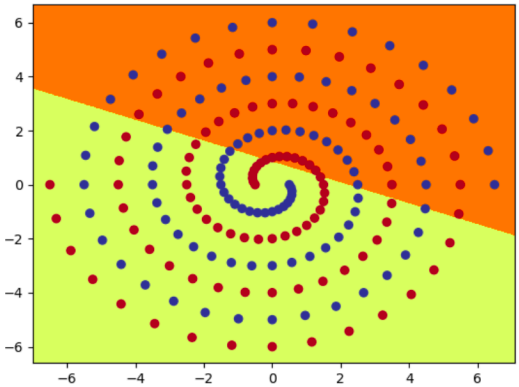
short1_2:

short1_3:

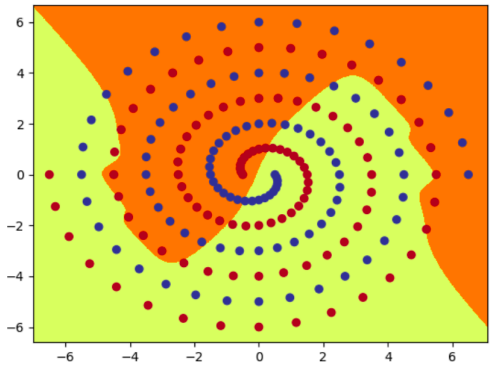




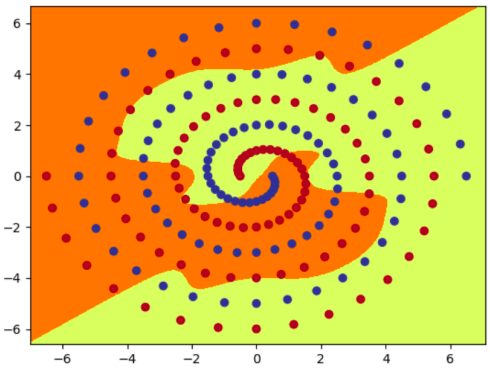
short1_4:



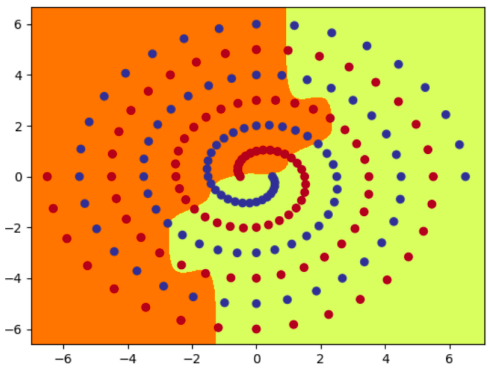
short2_0:



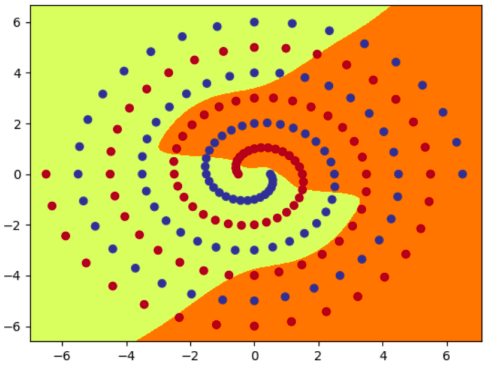
short2_1:



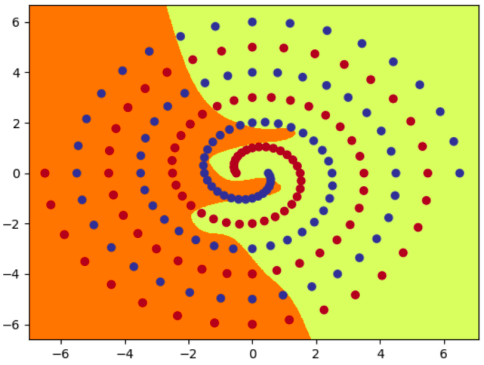
short2_2:



short2_3:



short2_4:



2.8

a)

For the hidden nodes, every node is to extract a feature from the data. In the hidden layer, function is trying to find the relationship with the original data with target. It adjusts the weight for different feature to fit the aim class.

b)

From the test, I find problem when initial weight at larger or small number, its speed will slow down when it reaches special loss and accuracy for both RawNet and ShortNet.

c)

Clearly, if data has been processed for classification, the learning speed and success will be increase tremendously. As for RawNet model, we train the model with a set of unprocessed data as input. Therefore, its speed and success is lower than other two models. On the one hand, we modified the input to tensor(r,a) in RawNet. On the other hand, we use the short-

cut as identity to help function balance the weights for each feature more sensiable.

d)

I have tried SGD for polar, but it performance is not good. In order to classified all test data within 20000 epochs, I need to increase the hidden nodes to catch the more useful feature. As for add more hidden layer, it can improve the classification performance.