Project 1

Characters, Spirals and Hidden Unit Dynamics

Swati Bharti

z5277828

Part 1: Japanese Character Recognition

Question 1:

	o	ki	su	tsu	na	ha	ma	ya	re	wo
o	765	7	6	4	58	8	5	16	10	8
ki	5	673	60	34	53	27	20	30	37	50
su	8	105	689	57	80	125	147	29	94	88
tsu	14	17	27	756	19	17	9	10	43	3
na	30	30	30	15	625	19	25	86	6	53
ha	63	22	21	59	19	727	24	17	32	31
ma	2	60	47	14	33	28	726	53	48	18
ya	62	12	36	18	36	8	21	622	6	30
re	32	24	46	30	20	33	9	87	704	41
wo	19	50	38	13	57	8	14	50	20	678

Test set: Average loss: 1.0105, Accuracy: 6975/10000 (70%)

Question 2:

	o	ki	su	tsu	na	ha	ma	ya	re	wo
o	846	4	8	3	49	9	3	17	9	1
ki	3	807	22	12	33	12	11	16	31	18
su	3	37	823	27	24	72	51	24	29	49
tsu	6	5	45	917	11	12	10	5	46	4
na	23	20	9	1	777	13	13	25	3	28
ha	36	11	16	11	9	821	3	9	11	6
ma	6	63	29	8	36	27	893	34	34	29
ya	41	3	12	7	22	3	7	808	4	18
re	27	21	18	6	18	23	1	28	828	11
wo	9	29	18	8	21	8	8	34	5	836

Test set: Average loss: 0.5415, Accuracy: 8356/10000 (84%)

Question 3:

	o	ki	su	tsu	na	ha	ma	ya	re	wo
o	956	1	8	3	11	2	3	5	2	9
ki	5	919	11	1	7	3	5	2	13	5
su	3	5	883	14	4	31	9	1	4	1
tsu	0	1	29	965	3	9	1	0	1	1
na	20	13	10	0	949	10	11	9	11	12
ha	4	1	15	6	1	916	3	2	2	2
ma	0	36	16	4	5	15	964	6	9	7
ya	7	5	12	3	7	2	4	951	3	9
re	3	5	4	1	12	3	0	5	949	8
wo	2	14	12	3	1	9	0	19	6	946

Test set: Average loss: 0.2296, Accuracy: 9398/10000 (94%)

Question 4:

a)

From this exercise I learnt that how neural networks can be used in real life applications and understood how the different models help in increasing the accuracy on the same dataset.

Relative accuracy of 3 models are:

NetLin	NetFull	NetConv
70%	84%	94%

b)

The confusion matrix for different models is as follows:

In the confusion matrix given below, the color **red** represents the wrong predictions with probability **greater than or equal to 50** and the color **blue** represents the wrong predictions with probability **greater than or equal to 25 but less than 50**.

NetLin

	o	ki	su	tsu	na	ha	ma	ya	re	wo
o	765	7	6	4	58	8	5	16	10	8
ki	5	673	60	34	53	27	20	30	37	50
su	8	105	689	57	80	125	147	29	94	88
tsu	14	17	27	756	19	17	9	10	43	3
na	30	30	30	15	625	19	25	86	6	53
ha	63	22	21	59	19	727	24	17	32	31
ma	2	60	47	14	33	28	726	53	48	18
ya	62	12	36	18	36	8	21	622	6	30
re	32	24	46	30	20	33	9	87	704	41
wo	19	50	38	13	57	8	14	50	20	678

NetFull

	o	ki	su	tsu	na	ha	ma	ya	re	wo
o	846	4	8	3	49	9	3	17	9	1
ki	3	807	22	12	33	12	11	16	31	18
su	3	37	823	27	24	72	51	24	29	49
tsu	6	5	45	917	11	12	10	5	46	4
na	23	20	9	1	777	13	13	25	3	28
ha	36	11	16	11	9	821	3	9	11	6
ma	6	63	29	8	36	27	893	34	34	29
ya	41	3	12	7	22	3	7	808	4	18
re	27	21	18	6	18	23	1	28	828	11
wo	9	29	18	8	21	8	8	34	5	836

NetConv

	o	ki	su	tsu	na	ha	ma	ya	re	wo
o	956	1	8	3	11	2	3	5	2	9
ki	5	919	11	1	7	3	5	2	13	5
su	3	5	883	14	4	31	9	1	4	1
tsu	0	1	29	965	3	9	1	0	1	1
na	20	13	10	0	949	10	11	9	11	12
ha	4	1	15	6	1	916	3	2	2	2
ma	0	36	16	4	5	15	964	6	9	7
ya	7	5	12	3	7	2	4	951	3	9
re	3	5	4	1	12	3	0	5	949	8
wo	2	14	12	3	1	9	0	19	6	946

Analysis from Confusion Matrix:

The **Netlin Model** is the simplest model, and this has more wrong predictions than the other two models. This may be because the model does not have many hidden layers and also the time taken to train is less.

The **NetFull Model** is fully connected 2-layer network and has 90 hidden nodes. The NetFull makes mistakes similar to NetLin but the number of wrong predictions is less. The major wrong predictions are made for 2 letters:

Target	Predicted
tsu	re
ma	ki

The wrong predictions in this layer may be because this model does not clearly classify the minute details in images as it does it in convolution model where filters are being used to extract the features.

The **NetConv Model** uses a convolution layer and a fully connected layer. The accuracy of this model is higher when compared to the other two and also the number of wrong predictions is also less. The wrong predictions are less than 25 for most of the cases and only for the 3 characters the wrong predictions have value greater than 25. The characters are:

Target	Predicted
su	ha
tsu	su
ma	ki

The wrong predictions may be because of the 2 reasons:

A. The image quality is low i.e. the original image is of size 28*28.
B. The letters are written by humans and everyone's handwriting is different when we scan the characters pixel by pixel.

c)

The effect of meta parameters:

The effect of parameters such as learning rate, momentum and epochs were changed for all the 3 models and was evaluated.

The learning rate was changed to 0.1 and momentum to 0.9 and the results are as follows:

*In the table, **"- "** represents that the particular meta parameter is not used.*

Model Name	Learning Rate	Momentum	Hidden nodes	Accuracy
Netlin	**-**	**-**		**70%**
Netlin	**0.1**	**-**		**67%**
Netlin	**-**	**0.9**		**69%**
Netlin	**0.1**	**0.9**		**60%**
NetFull	**-**	**-**	**90**	**84%**
NetFull	**0.1**	**-**	**90**	**85%**
NetFull	**-**	**0.9**	**90**	**85%**
NetFull	**0.1**	**0.9**	**90**	**76%**
NetConv	**-**	**-**		**94%**
NetConv	**0.1**	**-**		**96%**
NetConv	**-**	**0.9**		**95%**
NetConv	**0.1**	**0.9**		**10%**

The effect of Epochs on the models:

For **NetLin** model the number of epochs does not matter and the accuracy remains the same i.e. between 67% to 70%.

The **Netfull** Model with the increase in number of epochs increased. **Base Accuracy is 84%**

Epochs	Accuracy	Increased by
20	86%	2%
100	87%	3%

If the number of epochs is increased more than 100 the accuracy remains same and does not increase.

The NetConv model maintains the same accuracy when the number of epochs were increased.

Note: The epochs were tested independently i.e. without changing the learning rate and momentum value

Architecture changes in NetConv Model

Increased the number of layers in the model and also changed the filter size but it did not make any difference to the accuracy.

The effect of avgpool filter: With the initial architecture given in the question after adding the **avgpool** filter we get **accuracy of 89%** which is less and the results will be similar to Netfull model. hence this approach is not used in our model.

The number of **epochs** were increased and the results are given below in the table:

Base Accuracy is 89%

Epochs	Accuracy	Increased By
20	91%	2%
30	92%	3%

Once the model achieves 92% the accuracy wont increase for any number of epochs.

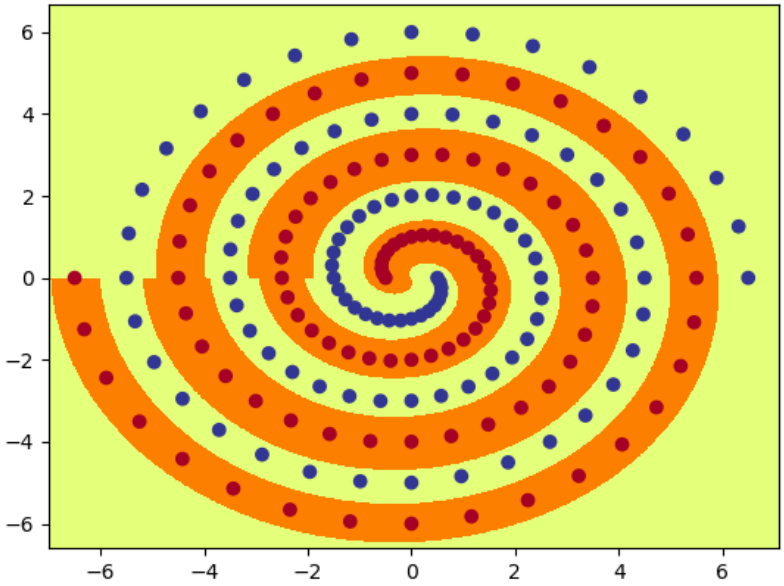
The effect of maxpool filter: With the initial architecture given in the question after adding the **maxpool** filter we get maximum **accuracy (94%)** as it takes the maximum value from the filter values and the new image or output generated will have pixels with maximum intensity.

The number of **epochs** were increased but the accuracy remained the same for this model.

Part 2: Twin Spirals Task

Question 2:

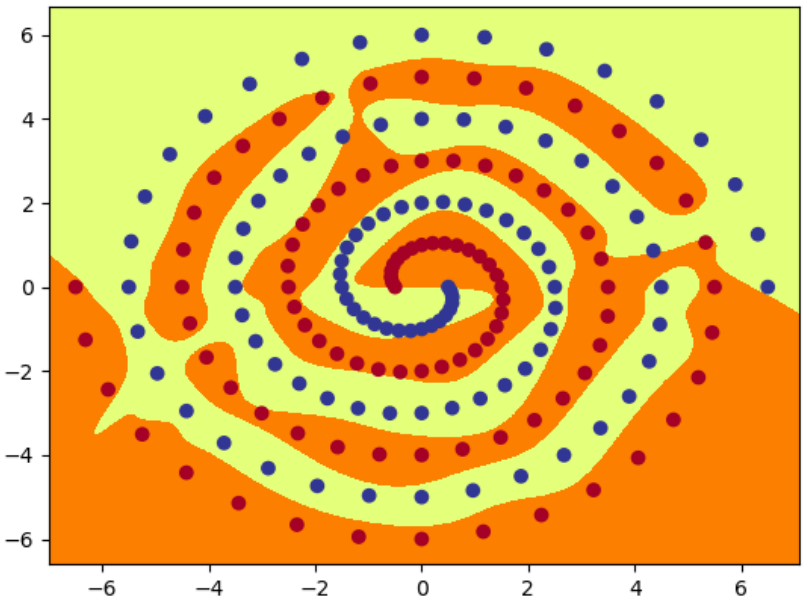
Minimum number of Hidden nodes is: 8



polar_out.png

Question 4:

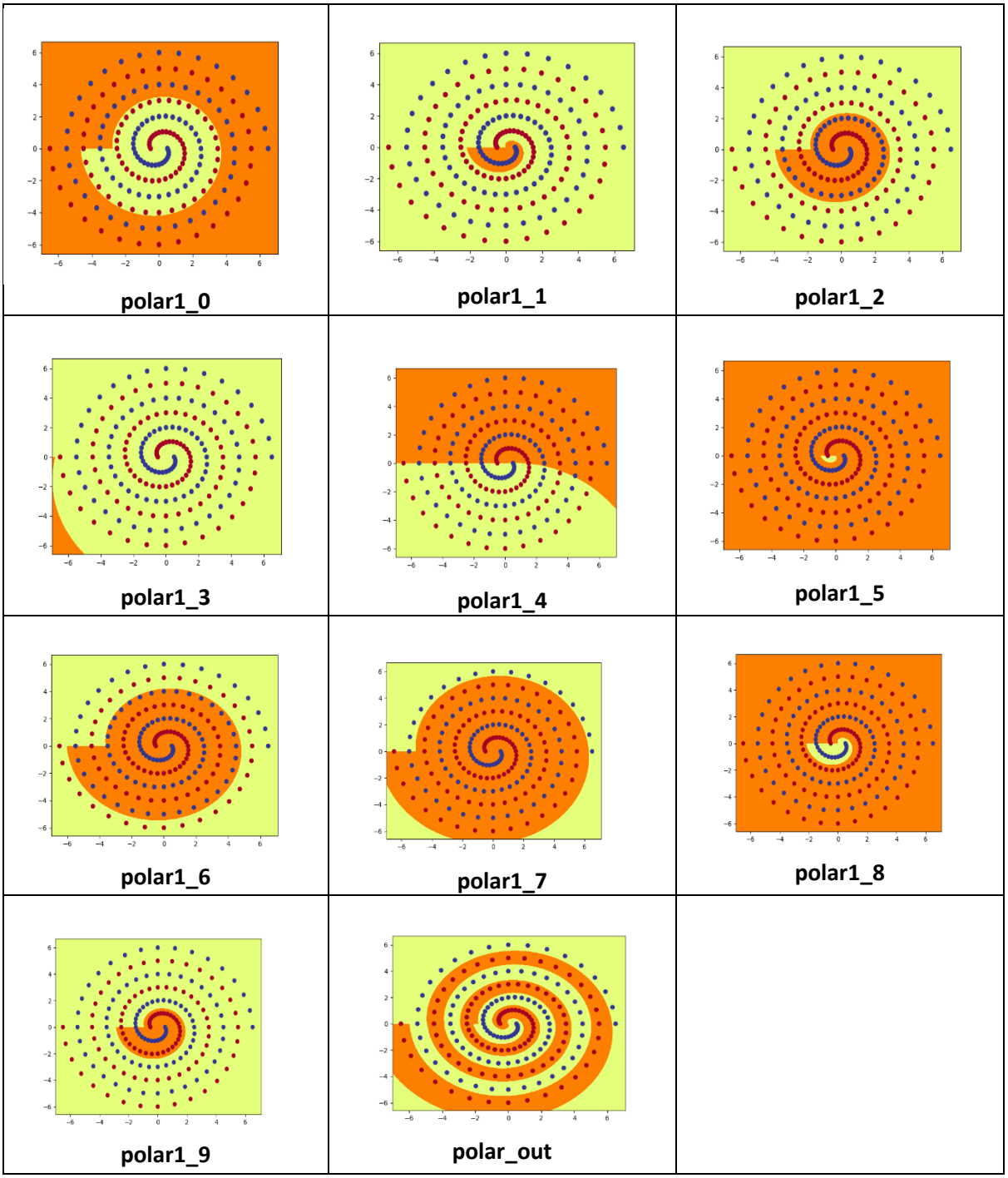
The minimum number of hidden nodes is 8, the initial weight is 0.121



raw_out.png

Question 5:

polar_output for 1 hidden layer with 10 nodes each



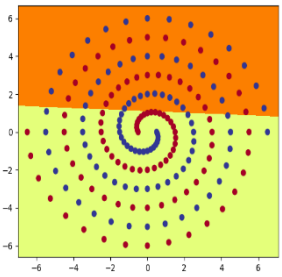
polar1_0 | polar1_1 | polar1_2

polar1_3 | polar1_4 | polar1_5

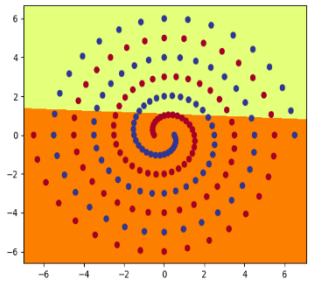
polar1_6 | polar1_7 | polar1_8

polar1_9 | polar_out

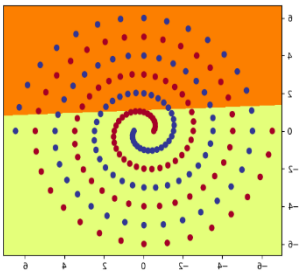
Raw_output for 2 hidden layers with 10 nodes each



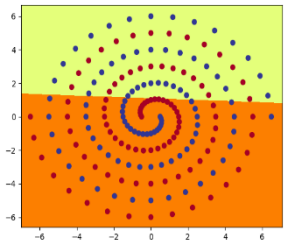
raw1_0



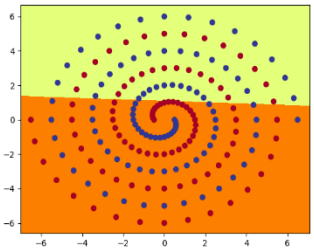
raw 1_1



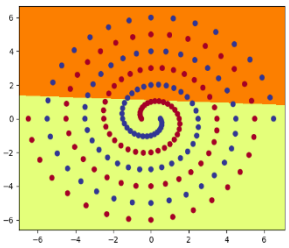
raw 1_2



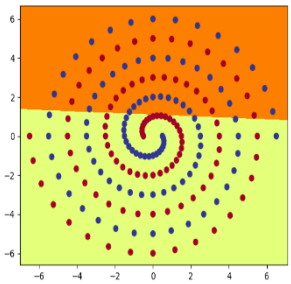
raw 1_3



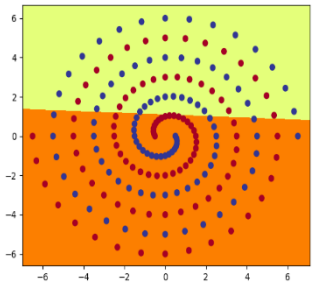
raw 1_4



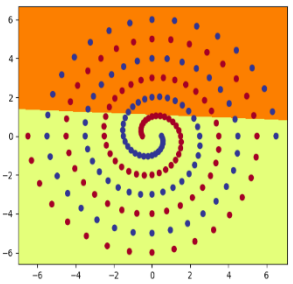
raw 1_5



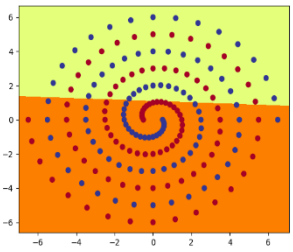
raw 1_6



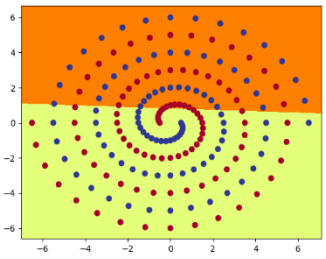
raw 1_7



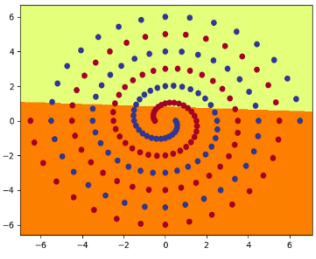
raw 1_8



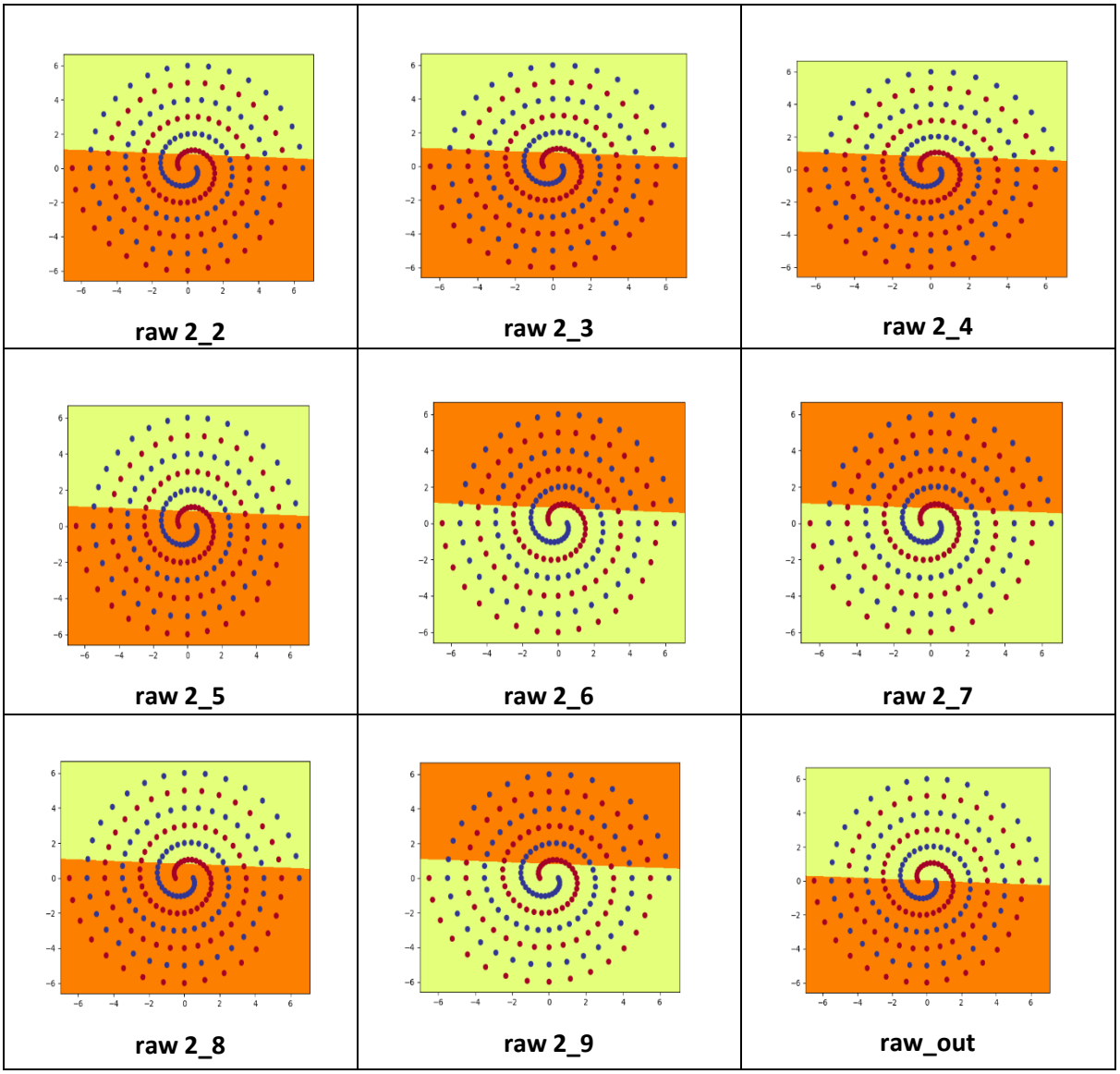
raw 1_9



raw 2_0



raw 2_1



Question 6:

a)

Function computed by Hidden layers in PolarNet and RawNet:

PolarNet	RawNet
Contains one hidden Layer	Contains two hidden Layer
Polar coordinates are given as input	Raw input is given as input
In this model, the hidden layers generate the spiral and since this uses polar coordinates a single point will have many positions in the (x, y) plane and because of this the spiral gets generated in single layer	In this model, there are 2 hidden layers. The first hidden layer generate linearly separable features, second will generate convex features which will be then combined in the output layer to get the spiral shape.

b)

For **weights less than 0.121** the accuracy will be around 50 for any number of iterations i.e. the model suffers from **vanishing gradient problem**.

For weights between 0.121 and 0.3 the model gets trained faster i.e. within 20000 epochs and also the accuracy is better.

For very large values the model wont converge easily and suffers from **exploding gradient problem i.e. for values greater than 3.**

c)

Effect of changing the batch size from 97-194:

This change gave a positive result i.e. the model's performance increased with respect to time and number of epochs to converge.

Using SGD instead of Adam:

This change had negative impact when there were no meta-parameters defined i.e. the model took more time to train with the size of 194.But, when the batch size was reduced to 10 the model converged faster than the latter one.

Changing tanh to ReLU:

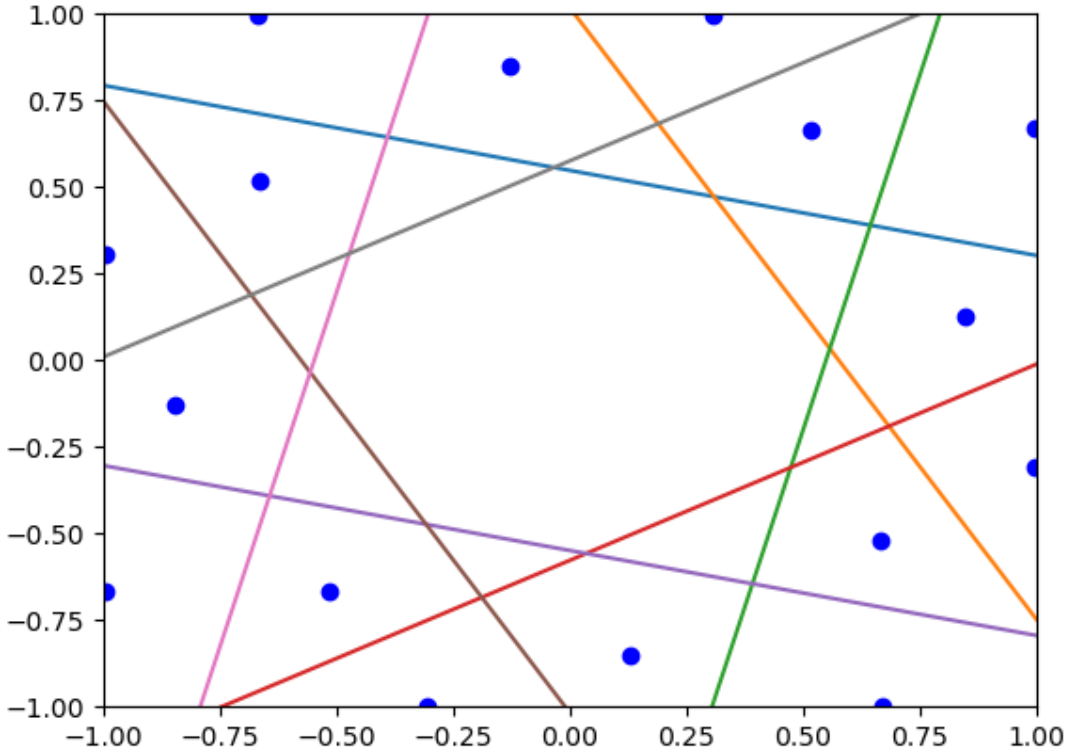
The accuracy will be decreased because in ReLU, the gradient descent won't converge when activation function is less than zero. i.e. the model will suffer from vanishing gradient problem and does not generate the spiral output which we require. Hence, ReLU is not suitable.

Adding more layers:

Adding more did not have any effect on the model.

Part 3: Hidden Unit Dynamics

Question 1:



star16.png

Question 2:

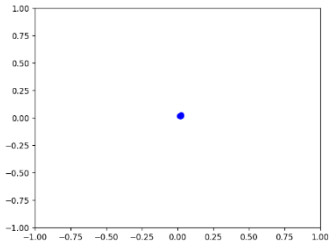


image_1

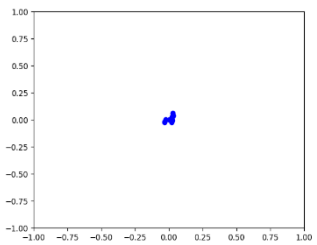


image_2

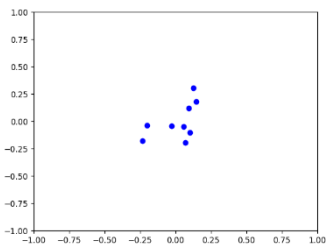
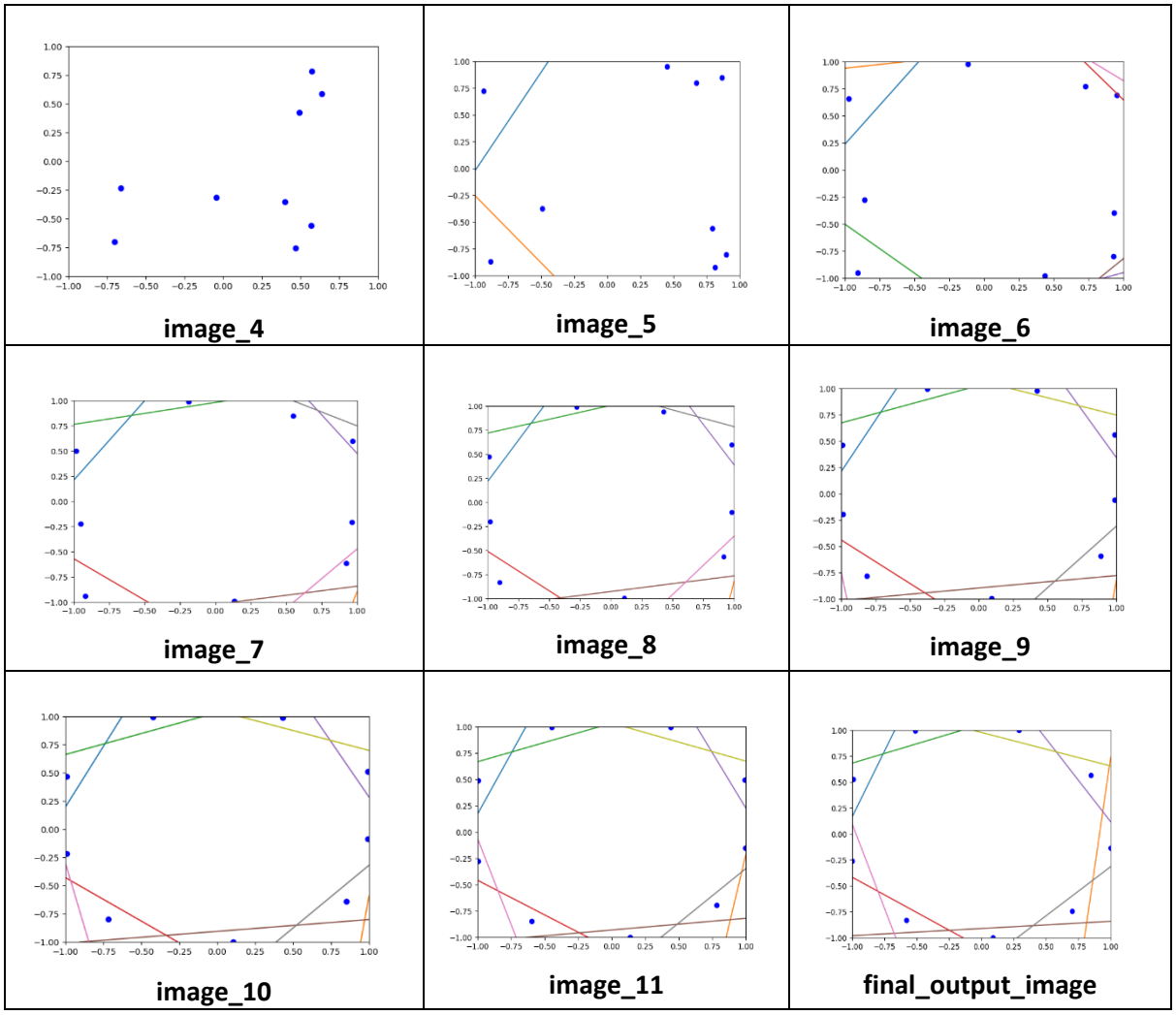


image_3



The above 12 images are obtained at different epochs of the execution.

From the images, we can infer that the encoder takes 300 epochs to separate all the hidden unit activations from each other and after till 3000 epochs the encoder adds the output boundaries to separate the hidden unit activations. The encoder needs to separate the 9 hidden unit activations with 9 output boundaries, so till 3000 epochs the model tries to separate all hidden unit activations. In **image_11**, we can see that 8 hidden unit activations are separated and the 9^{th} hidden unit activation is starting to separate from others (orange line).

In the **final_output_image**, we can see that all the 9 hidden unit activations are clearly separated with the help of 9 output boundaries.

image wise description:

image_1: All the hidden unit activations are present in a cluster.

image_2: Some hidden unit activations are separated from the cluster**.**

image_3: All the hidden unit activations are clearly visible with minimum separation.

image_4: the hidden unit activations are separating away from each other**.**

image_5: The output boundaries for the 2 hidden unit activations are visible.

image_6: 7 output boundaries are visible to separate the hidden unit activations, but only 2 the hidden unit activations are clearly separated.

image_7: 8 output boundaries are visible to separate the hidden unit activations, but only 5 the hidden unit activations are clearly separated.

image_8: 8 output boundaries are visible to separate the hidden unit activations, but only 6 the hidden unit activations are clearly separated.

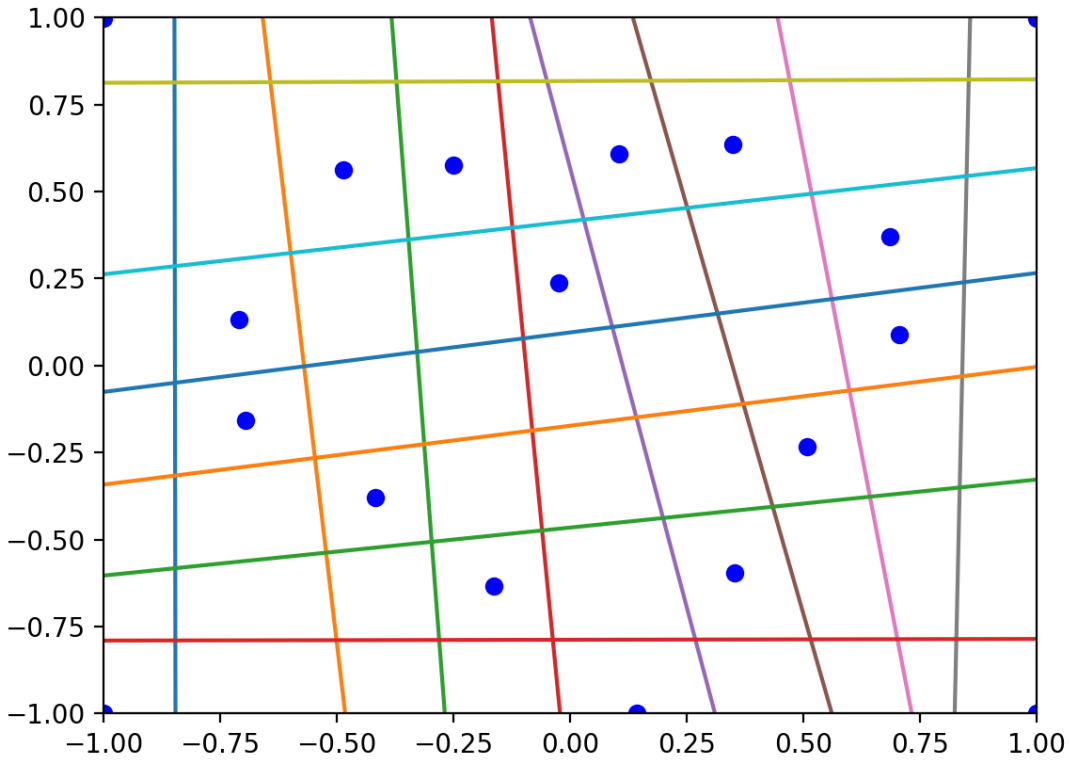
image_9: 9 output boundaries are visible to separate the hidden unit activations, but only 7 the hidden unit activations are clearly separated.

image_10: 9 output boundaries are visible to separate the hidden unit activations, but only 7 the hidden unit activations are clearly separated.

image_11: 9 output boundaries are visible to separate the hidden unit activations, but only 8 the hidden unit activations are clearly separated.

final_output_image: all 9 the hidden unit activations are clearly separated with the help of 9 output boundaries.

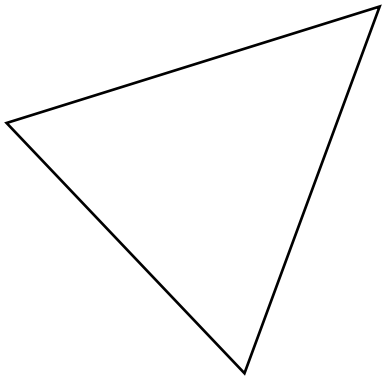
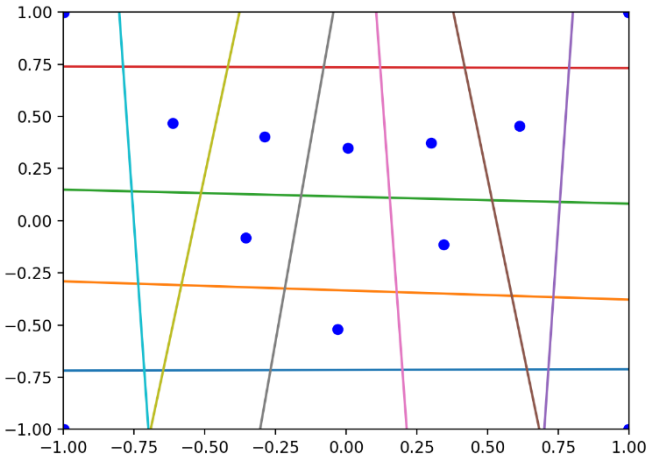
Question 3:



heart18.png

Question 4:

target 1

Design Created	Design created from target 1 matrix
	

target 2

Design Created	Design created from target 2 matrix