Part 1:

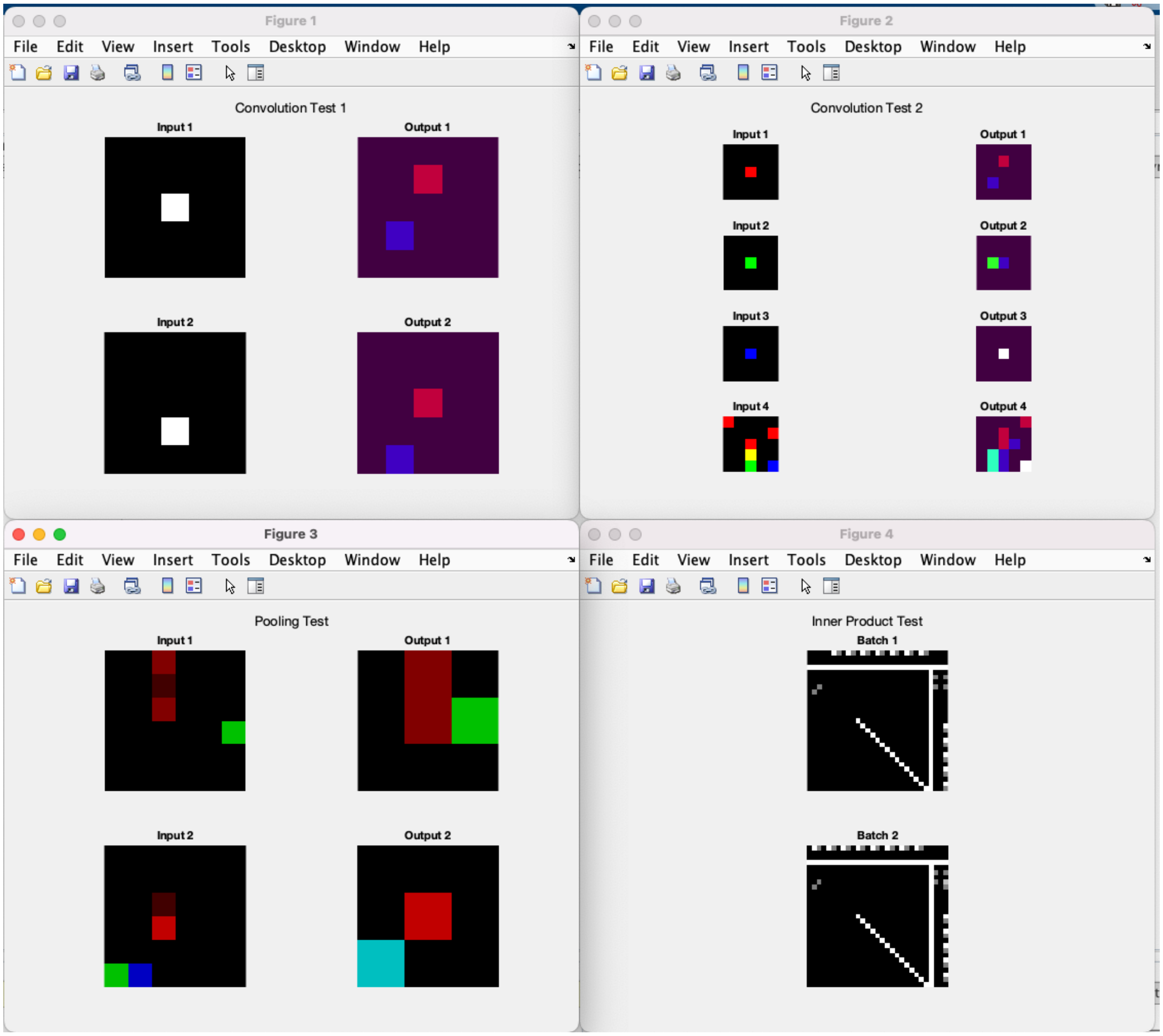
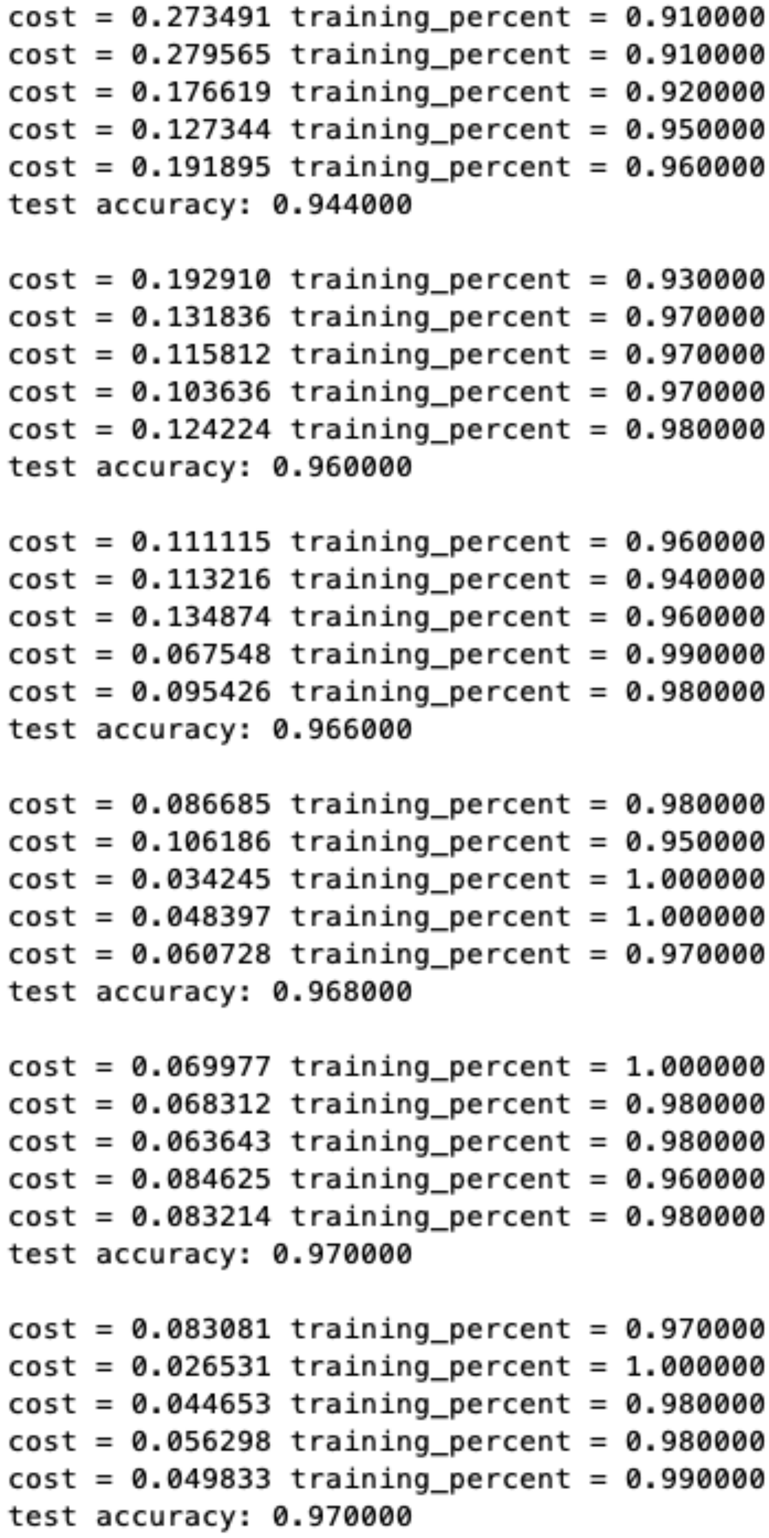


Figure 1
File Edit View Insert Tools Desktop Window Help
Convolution Test 1
Input 1
Output 1
Input 2
Output 2
Figure 2
File Edit View Insert Tools Desktop Window Help
Convolution Test 2
Input 1
Output 1
Input 2
Output 2
Input 3
Output 3
Input 4
Output 4
Figure 3
File Edit View Insert Tools Desktop Window Help
Pooling Test
Input 1
Output 1
Input 2
Output 2
Figure 4
File Edit View Insert Tools Desktop Window Help
Inner Product Test
Batch 1
Batch 2

Part 3.1:

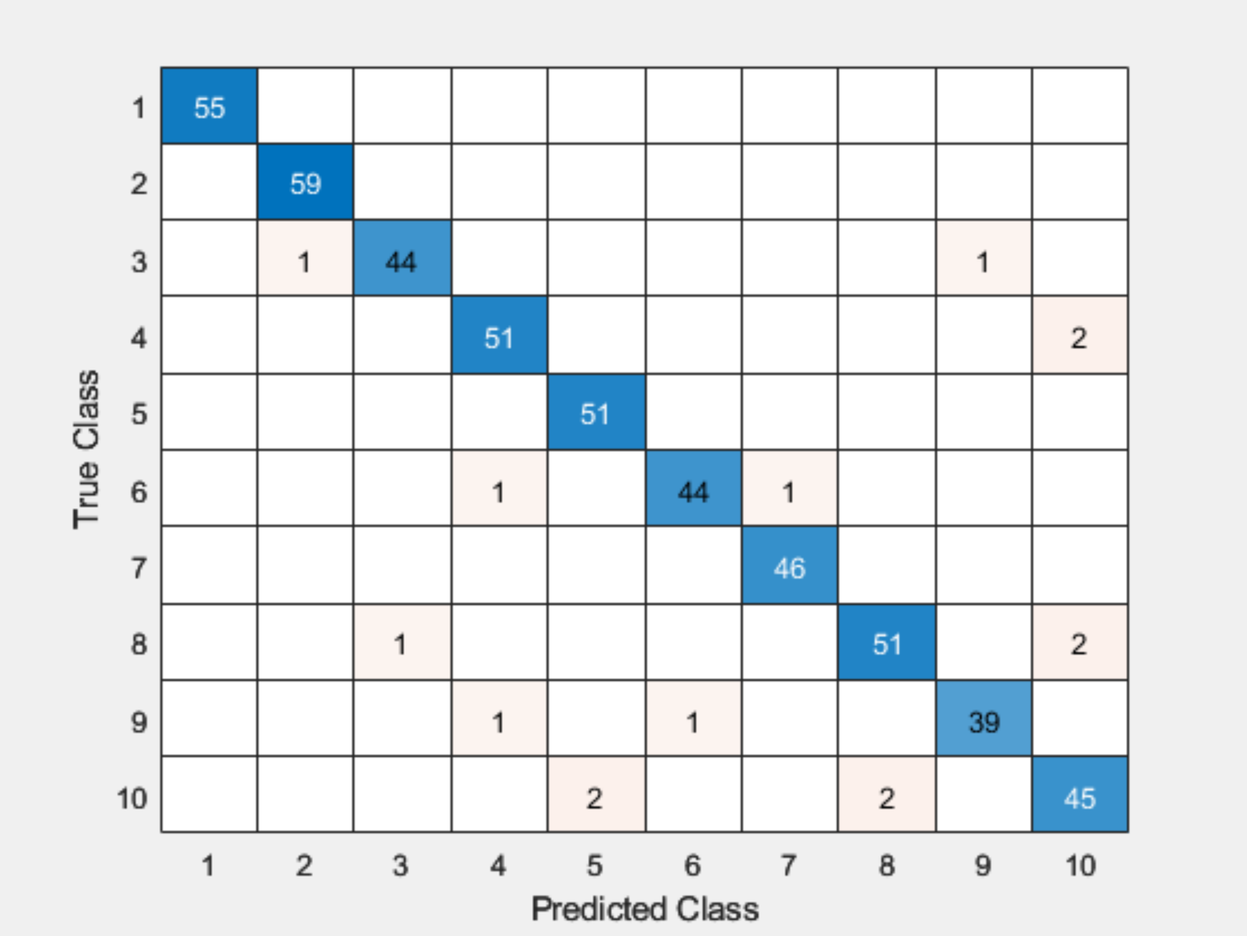
After the 3000 iterations, the test accuracy was 97% as can be seen below:



```
cost = 0.273491 training_percent = 0.910000
cost = 0.279565 training_percent = 0.910000
cost = 0.176619 training_percent = 0.920000
cost = 0.127344 training_percent = 0.950000
cost = 0.191895 training_percent = 0.960000
test accuracy: 0.944000

cost = 0.192910 training_percent = 0.930000
cost = 0.131836 training_percent = 0.970000
cost = 0.115812 training_percent = 0.970000
cost = 0.103636 training_percent = 0.970000
cost = 0.124224 training_percent = 0.980000
test accuracy: 0.960000

cost = 0.111115 training_percent = 0.960000
cost = 0.113216 training_percent = 0.940000
cost = 0.134874 training_percent = 0.960000
cost = 0.067548 training_percent = 0.990000
cost = 0.095426 training_percent = 0.980000
test accuracy: 0.966000

cost = 0.086685 training_percent = 0.980000
cost = 0.106186 training_percent = 0.950000
cost = 0.034245 training_percent = 1.000000
cost = 0.048397 training_percent = 1.000000
cost = 0.060728 training_percent = 0.970000
test accuracy: 0.968000

cost = 0.069977 training_percent = 1.000000
cost = 0.068312 training_percent = 0.980000
cost = 0.063643 training_percent = 0.980000
cost = 0.084625 training_percent = 0.960000
cost = 0.083214 training_percent = 0.980000
test accuracy: 0.970000

cost = 0.083081 training_percent = 0.970000
cost = 0.026531 training_percent = 1.000000
cost = 0.044653 training_percent = 0.980000
cost = 0.056298 training_percent = 0.980000
cost = 0.049833 training_percent = 0.990000
test accuracy: 0.970000
```

Part 3.2:

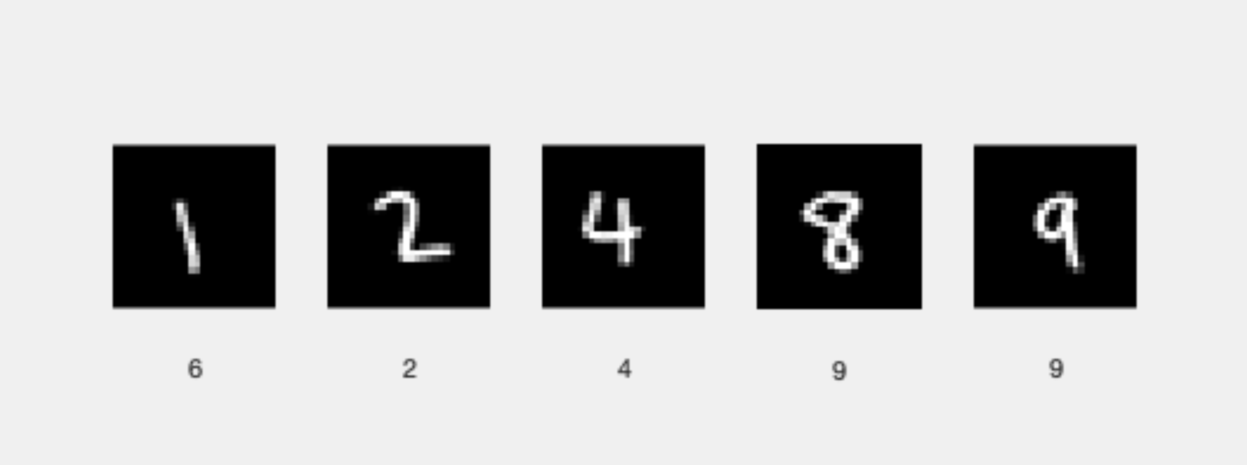
The confusion matrix is:



	1	2	3	4	5	6	7	8	9	10
1	55									
2		59								
3		1	44						1	
4				51						2
5					51					
6				1		44	1			
7							46			
8			1					51		2
9				1		1			39	
10					2			2		45

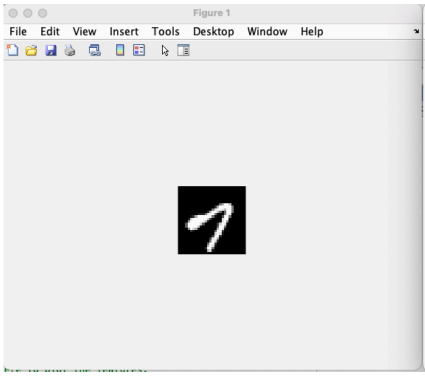
From this it can be seen that the numbers 9 and 7 were the most confused pairs. There were 2 instances where the actual number was 7 and the network predicted a 9 and there were 2 cases where the actual number was 9 and the predicted number was 7. The reason for this is quite clear as the dashed 7 (7) looks a lot like a 9, which can confuse the network and make it think that a 9 is a 7 and vice-versa.

Part 3.3:

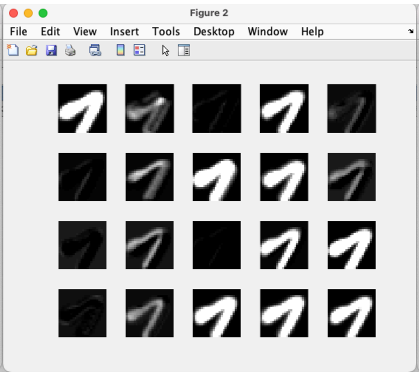


Based on the samples I wrote and entered as input, the success rate was about 60%.

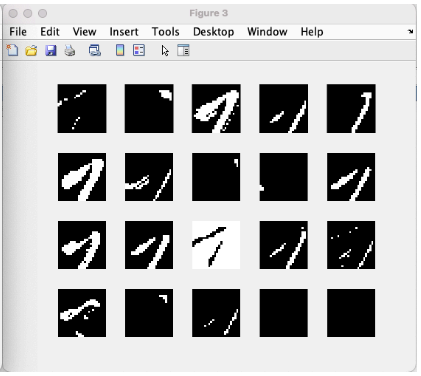
Part 4.1 and 4.2:



Input



Conv_layer

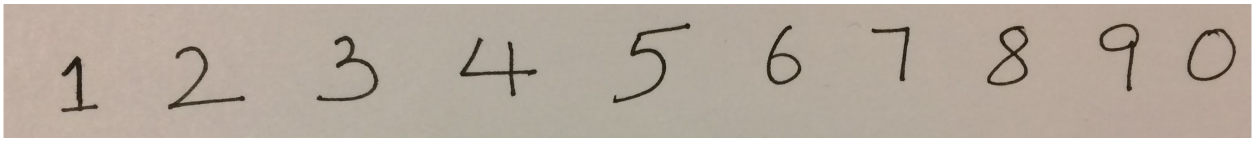


RELU layer

The first convolution layer feature maps seems to dilate the image and make the numbers thicker. It looks like it is doing a sort of max pooling as it picks the max image in a neighbourhood. Since these are greyscale images, it looks like its is trying to further maximize the values and almost binarize the image. The Relu activation maps are taking the input from the convolution layer and making all the spots close to zero darker especially those from the convolution layer that were negative and its leaving the larger values as they are. In some cases the images values are being flipped over completely (higher values are lowered and lower values are made larger). Since pixels with negative values are automatically made zero, these layers look similar, except the Relu layer seems to take on extreme values compared to the original image and convolution layer.

Part 5

Image1 output:



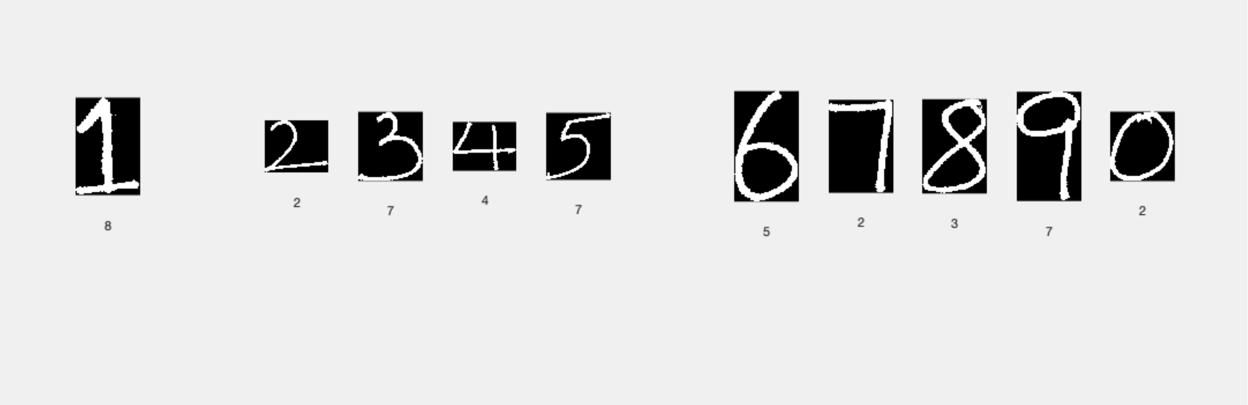
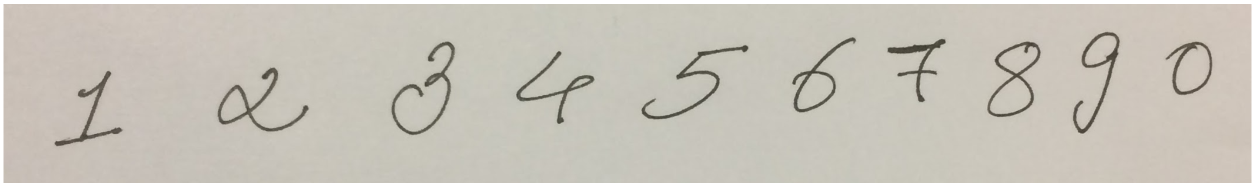


Image2 output:



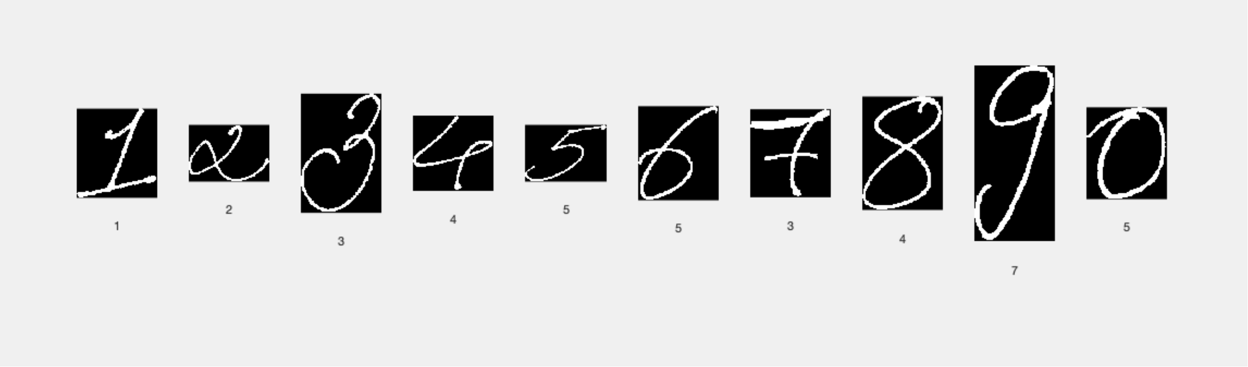
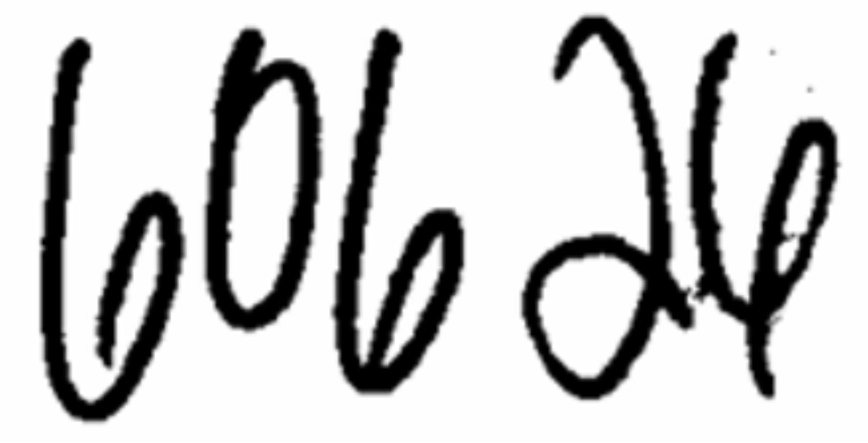


Image3 output:



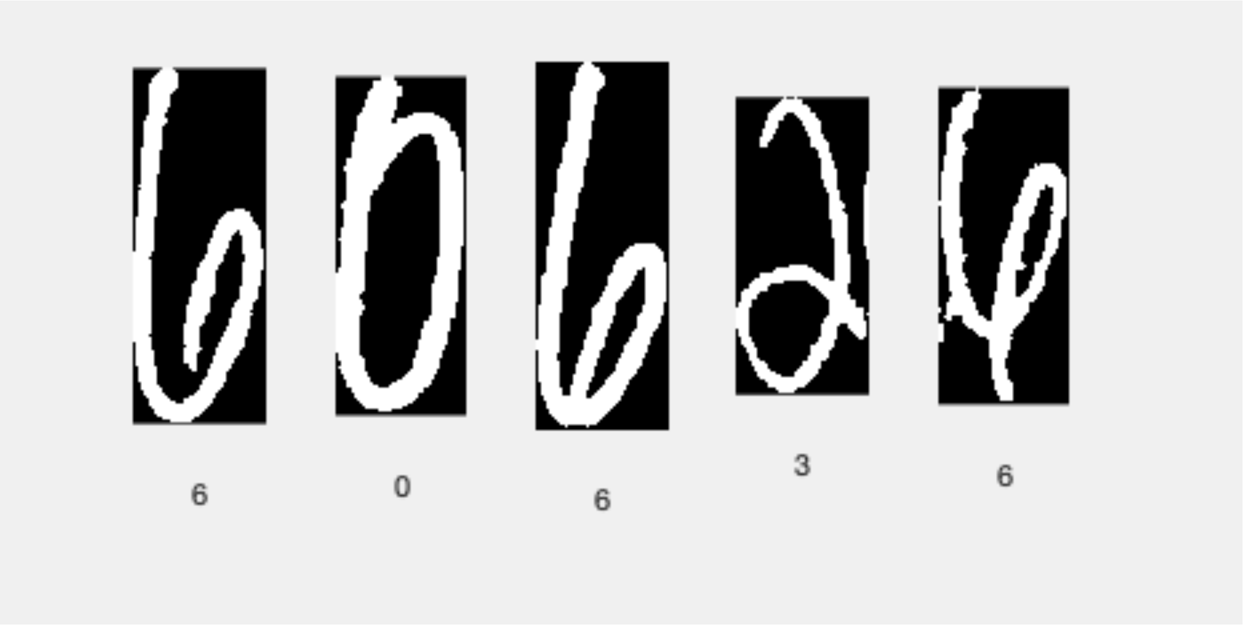
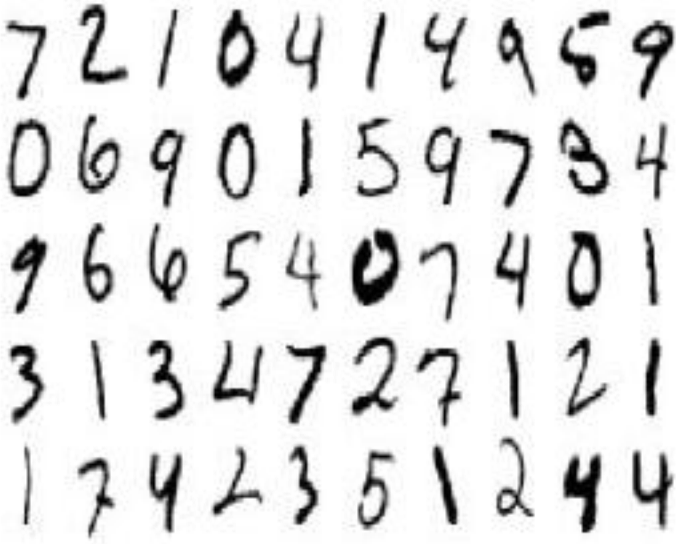
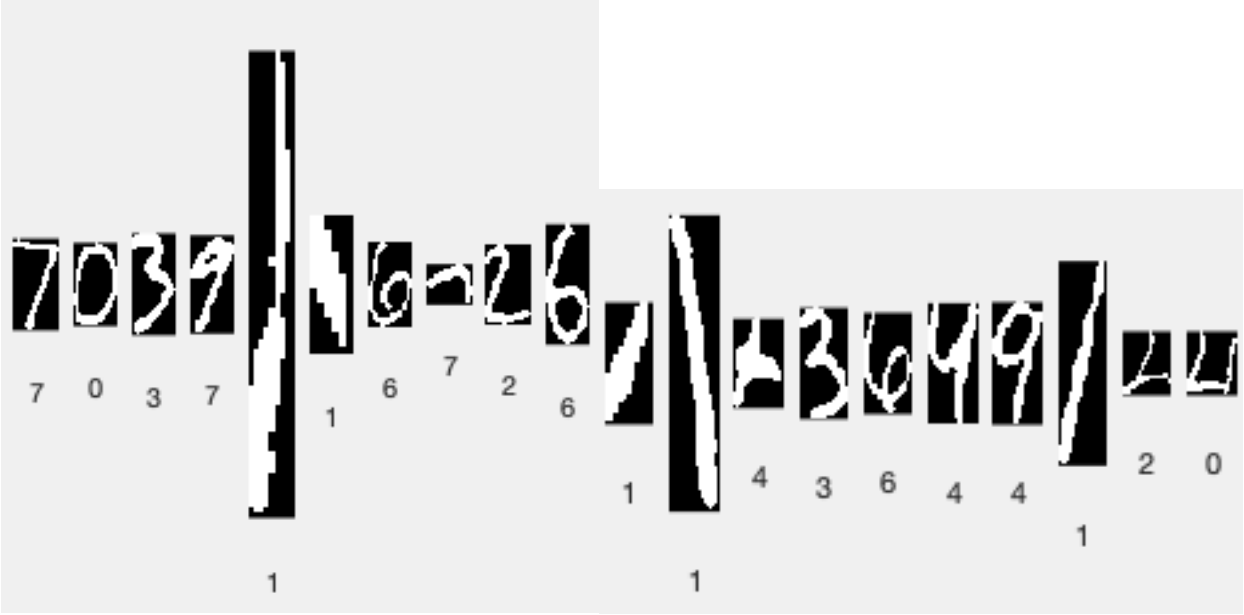
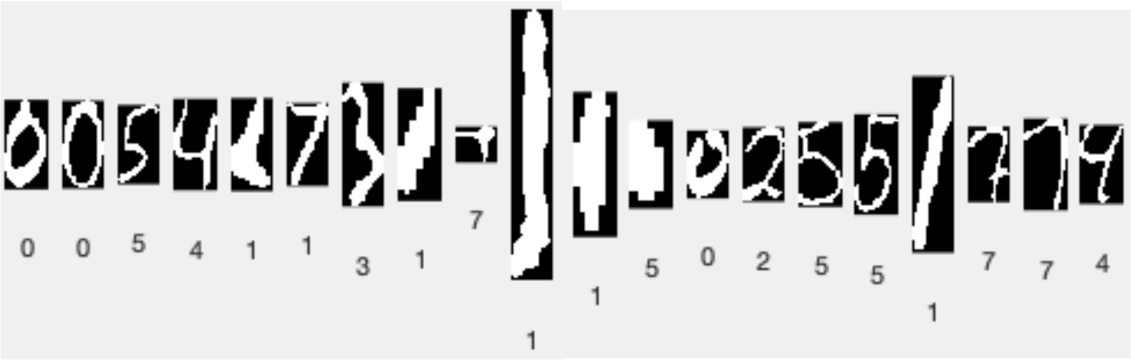
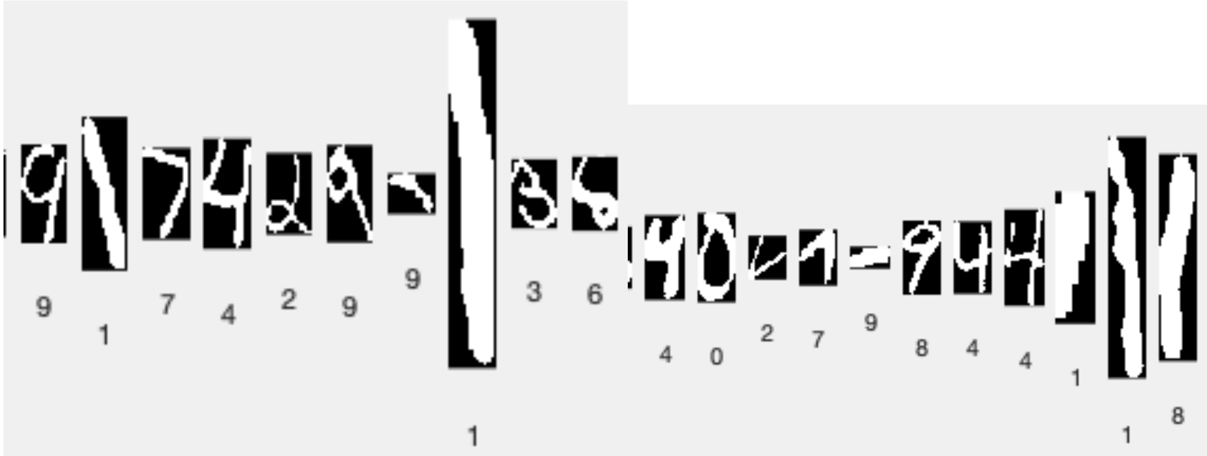


Image4 output:








9
1
7
4
2
9
9
1
3
6
4
0
2
7
9
8
4
4
1
1
8