Homework5 / Test of Generalization

李豪韋 (HW-Lee) ID 103061527

Overview

This project is followed by the last project: building a system successfully classifying most of the digits into their own classes. In the process of training, we tried to use any algorithm to make the I/O function close to given data as much as possible. In order to avoid overfitting, we even used cross-validation: separating the dataset into k pieces ($k = 8$ in last project) and then hiding one of them as testing data.

However, although we used some methodologies to make the system better, it is essential that we need to run a 'REAL' process. As a result, if we call the last project a 'training stage', this project can be called 'generalization stage', which involves no other algorithm to change the behavior of our system. There are 2000 images given in this project, which consists of two classes of digits (200 images of each class). Optimistically, the accuracy we get with testing data will be close to that with training data.

The more detailed information of the code is publicly available on https : //github.com/HW-Lee/2015-NN-Homeworks/tree/master/HW05.

Recap: The structure I used.

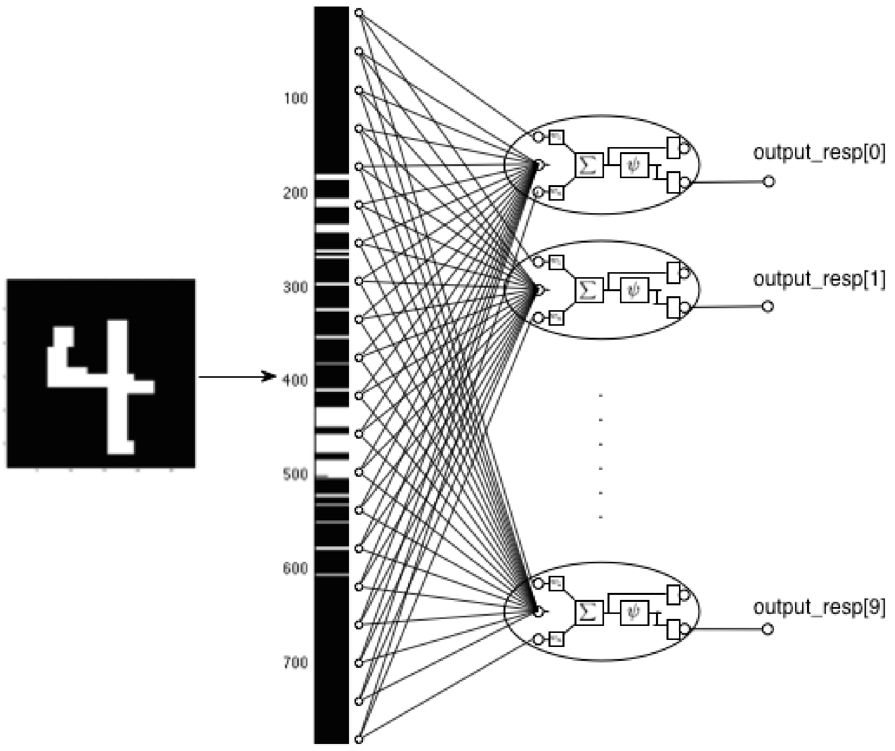

Figure 1: The structure of net. It simply consists of 10 neurons without any hidden unit, which acts like a linear filter, and the image data will be reshaped from 2D to 1D then all pixels will be regarded as an input port. However, the only thing different from the linear filter implemented in Hw2 is that the linear outputs are cascaded to the logistic function that maps any real value into a value ranged from 0 to 1, in order to make the output a measure of 'confidence'. Finally, the prediction is followed by the rule of choosing the class which turns out the highest confidence.

Results

1. Confusion Matrix and False Predicted Averaged Results

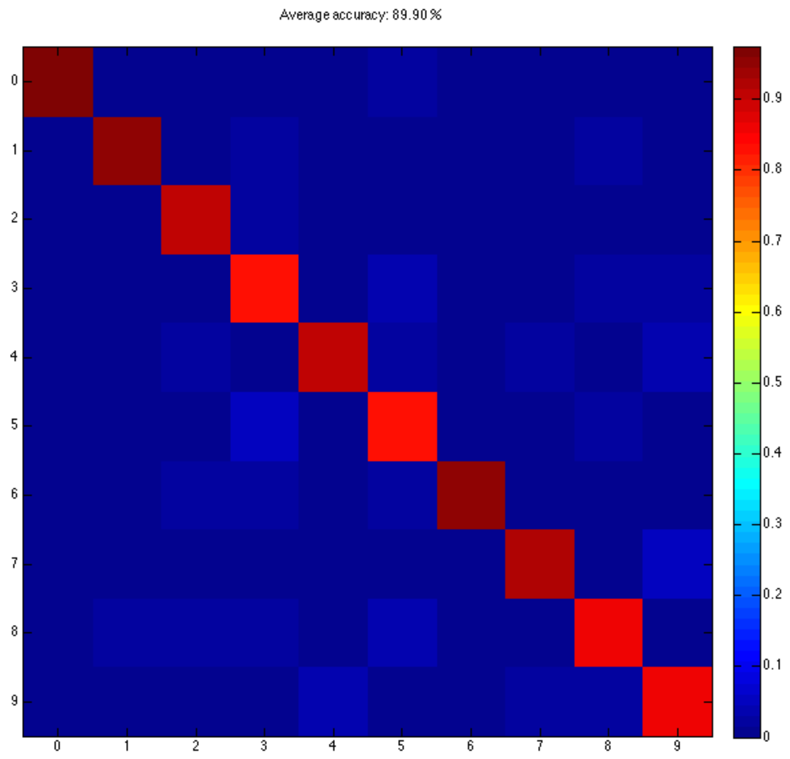

Figure 2: the average accuracy is 0.899

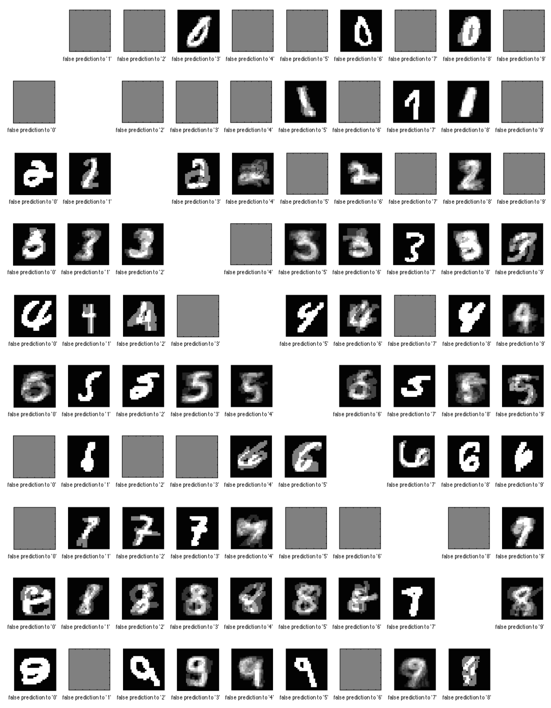

Figure 3: averaged false predicted images

2. Averaged Response of each class

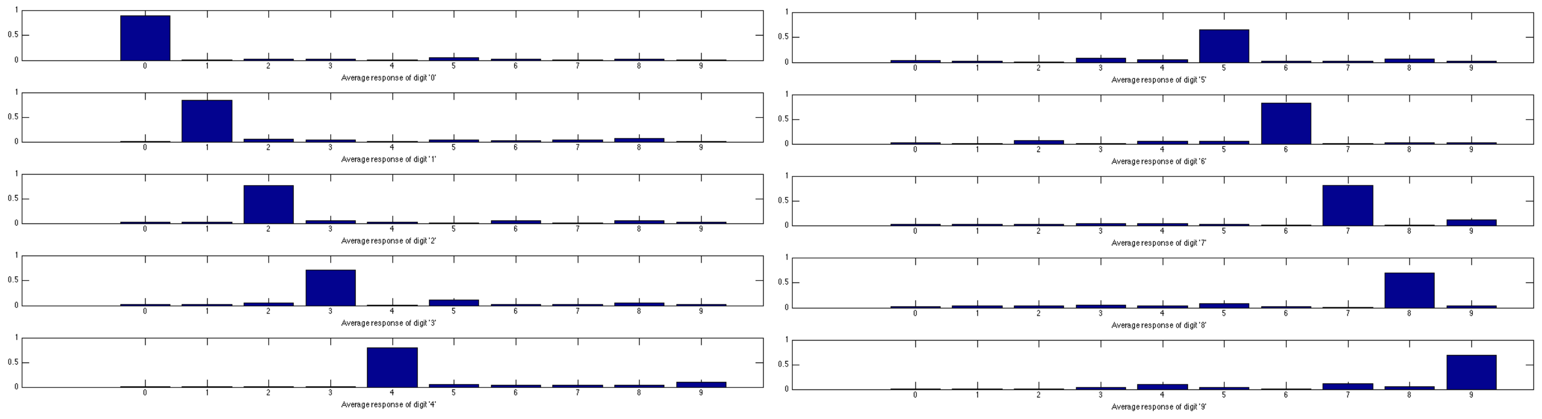

Figure 4: Averaged Response (left-side: 0-4, and right-side: 5-9, from top to bottom)

Discussion

1. Compare to the accuracy evaluated with cross-validation

result / truth	’0’	’1’	’2’	’3’	’4’	’5’	’6’	’7’	’8’	’9’
’0’	**0.99**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00
’1’	0.00	**0.94**	0.02	0.00	0.00	0.02	0.00	0.00	0.02	0.00
’2’	0.01	0.01	**0.87**	0.00	0.01	0.00	0.01	0.03	0.05	0.01
’3’	0.01	0.00	0.04	**0.84**	0.01	0.04	0.02	0.03	0.01	0.00
’4’	0.00	0.01	0.03	0.01	**0.85**	0.01	0.02	0.03	0.02	0.02
’5’	0.05	0.00	0.00	0.07	0.03	**0.79**	0.01	0.00	0.04	0.01
’6’	0.00	0.00	0.00	0.01	0.01	0.03	**0.94**	0.00	0.01	0.00
’7’	0.01	0.01	0.01	0.00	0.01	0.00	0.00	**0.93**	0.00	0.03
’8’	0.01	0.03	0.01	0.04	0.01	0.00	0.01	0.02	**0.85**	0.02
’9’	0.04	0.02	0.01	0.02	0.06	0.02	0.01	0.04	0.01	**0.77**

result / truth	’0’	’1’	’2’	’3’	’4’	’5’	’6’	’7’	’8’	’9’
’0’	**0.975**	0.000	0.000	0.010	0.000	0.000	0.005	0.000	0.010	0.000
’1’	0.000	**0.955**	0.000	0.000	0.000	0.015	0.000	0.005	0.025	0.000
’2’	0.005	0.010	**0.900**	0.010	0.025	0.000	0.020	0.000	0.030	0.000
’3’	0.010	0.025	0.020	**0.825**	0.000	0.055	0.025	0.005	0.020	0.015
’4’	0.005	0.010	0.010	0.000	**0.910**	0.005	0.015	0.000	0.005	0.040
’5’	0.025	0.005	0.005	0.035	0.030	**0.825**	0.020	0.005	0.035	0.015
’6’	0.000	0.005	0.000	0.000	0.010	0.010	**0.955**	0.010	0.005	0.005
’7’	0.000	0.010	0.010	0.005	0.030	0.000	0.000	**0.925**	0.000	0.020
’8’	0.010	0.020	0.015	0.025	0.015	0.020	0.015	0.005	**0.855**	0.020
’9’	0.005	0.000	0.005	0.025	0.035	0.005	0.000	0.050	0.010	**0.865**

   The result shows that 1) the system is neither overfitting nor underfitting, 2) the simple structure is sufficiently complex to learn the task.

2. Is deep learning, which contains more than 3 layers, needed to learn this task?

   The answer is no, I think, because 1) the dataset is well-preprocessed: the digits have roughly same size, the skewness can be roughly ignored... 2) the dataset has been binarized, which significantly decreases the variance of outputs space. In addition, convolutional neural network might be able to increase the accuracy, but might cause overfitting issue or multiple elements activate with same features issue.

3. Are wrongly predicted results really difficult to be recognized?

   Actually it is not difficult. The reason why the system misclassifies these instances might be that the training dataset is too small to make the system figure out the most critical features of each class, and the accuracy will be lifted after increasing the number of training instances.