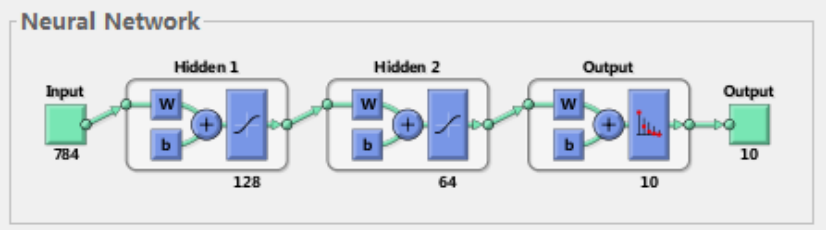
(1) Explain your network structure and parameter setting.

```
net = patternnet([128,64]);
net.trainFcn = 'trainscg';
net.trainParam.epochs = 1000;
net.trainParam.lr = 0.001;
net.trainParam.max_fail=20;
```

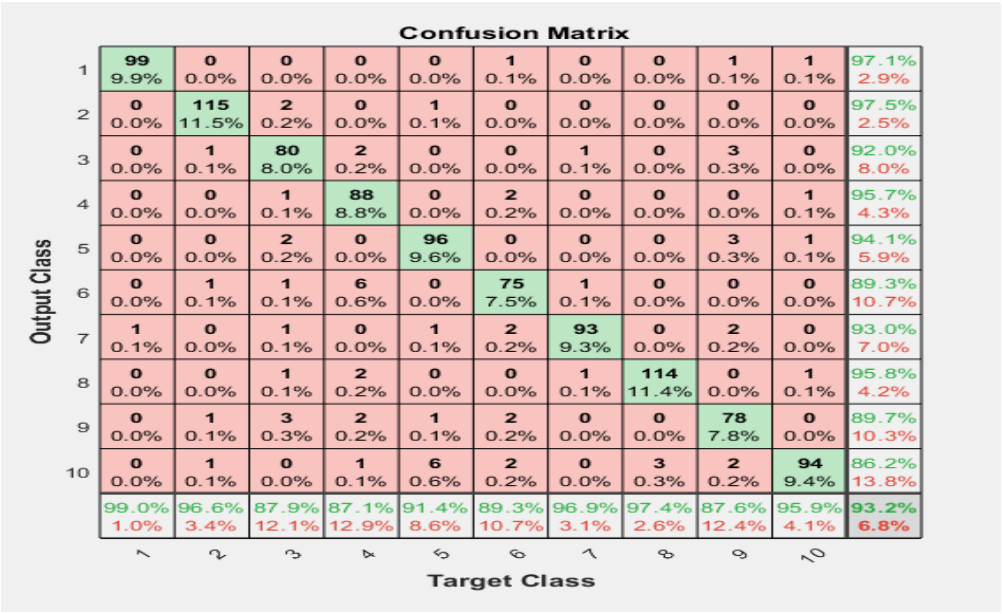


In the training section, I use Pattern recognition network and build a four layers neural network. The input layer has 784 nodes which are corresponding to 784 pixels of image and output layer has 10 nodes which are corresponding to 0-9. There are two hidden layers in between. The first hidden layer has 128 neurons and the second one has 64 neurons.
For parameter setting, I use 'trainscg' as my training function. Trainscg is a network training function that updates weight and bias values according to the scaled conjugate gradient method. Most of the parameters are set to default values expect validation check. Six validation check is not enough for this neural network, and it leads to a low accuracy. Therefore, I set validation check to 20.

(2) Your testing accuracy and confusion matrix.

Testing accuracy is 93.2%

Each row of the confusion matrix represents the output of sim() function. In other words, it is the result of Image recognition.
Each column of the confusion matrix represents the target value which is stored in the test_label.
This model has a high accuracy. It works well at most time. But sometimes this model confuses 4 with 9 and 3 with 5.



(3) Testing

In the neural network model, different parameters are adjusted to obtain the best accuracy. The parameters are number of hidden layers, number of neurons in each hidden layer, learning rate and training function.

Test ID	Number of hidden layers	Number of neurons	Learning rate	Training function	Accuracy
1	2	[128,64]	0.001	trainscg	93.20%
2	2	[128,64]	0.01	trainscg	92.20%
3	2	[128,64]	0.1	trainscg	91.90%
4	1	50	0.001	trainscg	91.80%
5	1	500	0.001	trainscg	89.10%
6	1	128	0.001	trainscg	92.10%
7	2	[200,128]	0.001	trainscg	92.30%
8	2	[50,25]	0.001	trainscg	89.70%
9	3	[200,128,64]	0.001	trainscg	92.20%
10	2	[128,64]	0.001	traingd	58.40%
11	2	[128,64]	0.001	traingdm	45.40%
12	2	[128,64]	0.001	traingda	92.10%
13	1	128	0.001	traingda	91.60%
14	2	[128,64]	0.001	traingdx	92.50%
15	1	128	0.001	traingdx	90.60%
16	3	[200,128,64]	0.001	traingdx	92.60%
17	2	[128,64]	0.001	trainrp	85.50%
18	2	[128,64]	0.001	traincgp	91.40%
19	2	[128,64]	0.001	traincgb	91.20%

Explanation of PCA

Principal Component Analysis is a statistical method which is used to reduce dimension. For this coursework, there are 784 features in each image. However, most features are empty which cost handling time and storage. Therefore, PCA can be implemented to rebuild k features which is mapped from 784 features. It means each features of k contains different information of original image. In this way, the input data is simplified.