ANN for Digit Recognition

Shi Yuhan(3035540736)
EEE(general)

1. Training with tensorflow.

1.1 Vary the learning rate in the training process

In the process of training with tensorflow, I compare the training results of different learning rates. It is found that the smaller the learning rate is, the higher the accuracy of the training results is, but also the longer the training time is.I combine the two factors and select the more suitable learning_rate=0.05.

Learning rate	hidden_layer_size	Iteration	Time	Accuracy
0.001	100*100	20	1:08.27	0.9454
0.005	100*100	20	1:06.82	0.9572
0.01	100*100	20	1:06.62	0.9562
0.01	100*100	10	33.50	0.9498
0.05	100*100	10	33.58	0.9341
0.1	100*100	10	33.69	0.901

1.2 Vary the hidden layer and hidden nodes in the training process

Learning rate	hidden_layer_size	Iteration	Time	Accuracy
0.005	150*150	10	34.26	0.9436
0.005	200*200	10	34.42	0.9471
0.005	80*80	10	33.47	0.9392
0.005	100	10	31.80	0.9584
0.005	200	10	32.03	0.9563
0.005	400	10	32.43	0.9554
0.005	80	10	32.02	0.9515
0.005	20	10	32.10	0.9366
0.005	20*20	10	33.68	0.9396
0.005	20*20*20	10	35.49	0.9219
0.005	80*80*80	10	36.07	0.9255
0.005	100*100*100	10	36.44	0.9297
0.005	200*200*200	10	37.71	0.9324

2.Training with sklearn.

Changing the learninn rate has little effect on the accuracy of the training results, so the intermediate value of 0.01 is used for training.2.1. vary the learning rate in the training process

Learning rate	hidden_layer_size	Iteration	Time(s)	Accuracy
0.001	100*100	20	12.56	-0.4522
0.005	100*100	20	11.90	-0.4522
0.01	100*100	20	11.97	-0.4522
0.05	100*100	20	11.71	-0.4522
0.1	100*100	20	12.02	-0.4522

Learning rate	hidden_layer_size	Iteration	Time(s)	Accuracy
0.01	100	20	4.91	-0.4651
0.01	50	20	2.04	-0.4279
0.01	20	20	0.95	-0.5858
0.01	80	20	3.36	-0.3743
0.01	70	20	3.46	-0.4456
0.01	80	50	9.122	-0.0085
0.01	80	100	18.933	0.1827
0.01	80	80	14.624	0.1372
0.01	80*20	50	12.86	0.0418
0.01	80*80	50	21.29	0.0183
0.01	80*100	50	25.22	0.1183
0.01	80*80	100	42.16	0.361
0.01	80*80	80	33.26	0.2809
0.01	80*80*20	50	24.46	0.0122
0.01	80*80*50	50	30.16	0.0361
0.01	80*80*80	50	33.04	0.1012

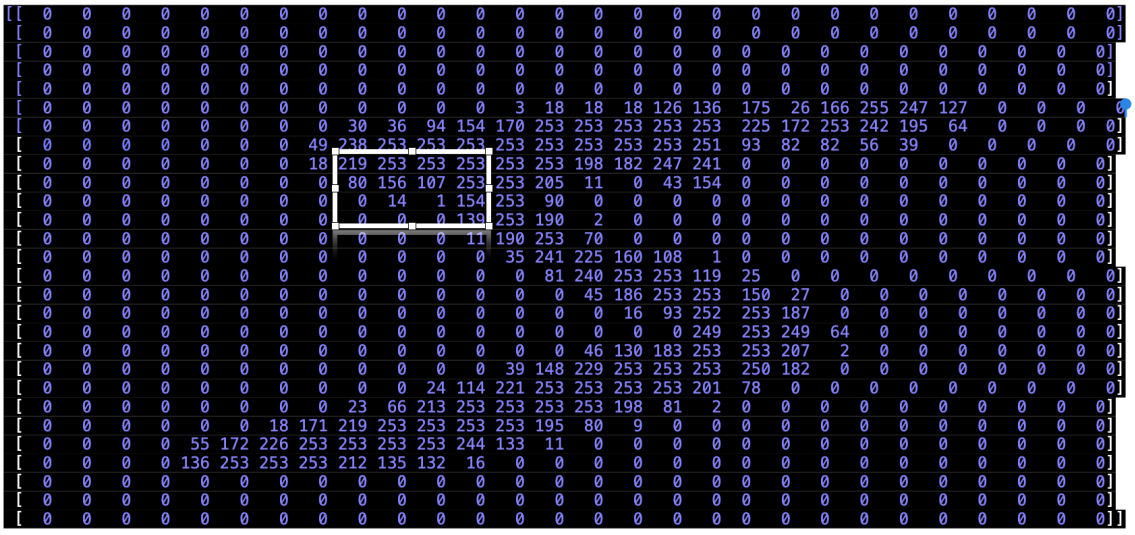
In single_layer training, the training result is the best when the number of hidden node is 80. Therefore, when training the neural network of two hidden layers, 80 is still the number of hidden nodes of the first layer. Vary the number of hidden nodes of second_layer to get the best training results.

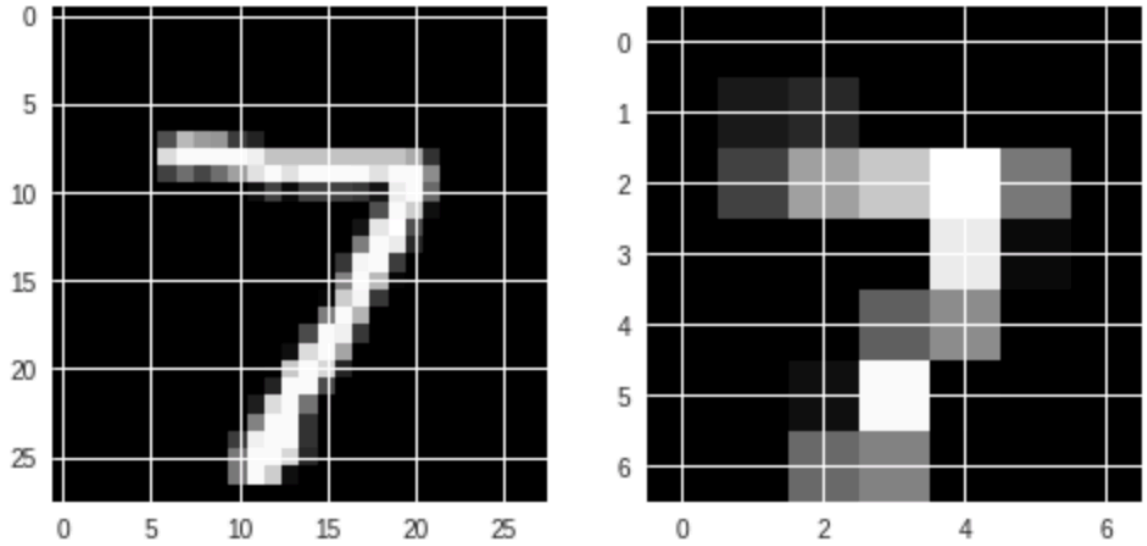
Use the same method to decide the number of hidden nodes in third_layer.

3. Summary

3.1. 28*28 to 7*7

The mnist dataset has a training set with 60,000 samples and a test set with 10,000 samples. Each picture in the MNIST data set is in the 28 * 28 format, and the data range for each pixel is 0-253, representing the gray value. To convert it to 7x7 format, you first need to convert the image data to FLOAT32 format and then divide by 255 to avoid data overflow. Then, every 4*4 pixels are averaged to obtain the value of the new pixel.





3.2. sklearn

In sklearn training, using MLPRegressor, the input is 7*7=49 node values, and the output is 0-9 ten values (eg: if the input image corresponds to a value of 9, the output is 9), the activation function is Relu.

Feed the function with x_train dataset. Comparing the predicted data with y_train dataset, the accuracy of the training is the average of the differences between all predicted values and the true values.

3.3. tensorflow

In the process of training with the tensorflow framework, a multi-layer neural network is established. The input_layer is 7*7=49 nodes, and the output_layer is 0-9 ten nodes, denoted by 0/1, 1 represents the current class, (eg: If the current input picture corresponds to a value of 4, the output value should be 0000100000 when the training result is correct. If the value is 0, the corresponding output should be 100000000).

Taking a two-layer hidden layer neural network as an example, the input of the first hidden layer is the input_layer, the output is the input of the second hidden layer, and the output of the second hidden layer is the input of the output_layer. There are several nodes in each layer. The result of the output layer is compared with the y(label) of the training set, and the difference between the label and training result is placed in the optimizer for optimization. The difference is narrowed during multiple iterations.

Due to the large number of training sets, we can improve the training efficiency by randomly selecting a part of the data set for training.

3.4. Comparison

When the number of hidden layers increases, the accuracy of the training results may increase. But when it exceeds a certain level, over-fitting will occur, and the accuracy will decrease. The same effect will occur when the number of hidden nodes increases. But the time interval has no big impact.

When the number of iterations is increased, the accuracy of the training results will be better, but the training time will increase significantly.

Comparing the two methods, the training process of MLPRegressor is relatively simple, and the time required is relatively short. For the training precision, the accuracy of the classifier is higher than the overall level of linear regression, but both need to be improved.