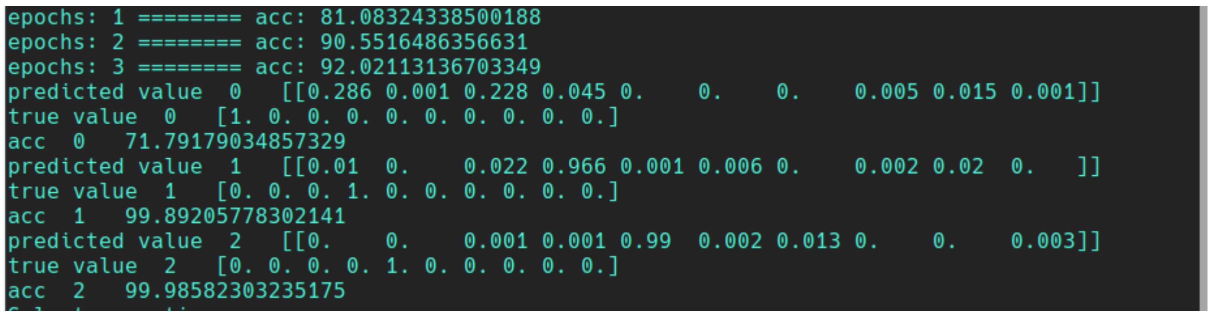
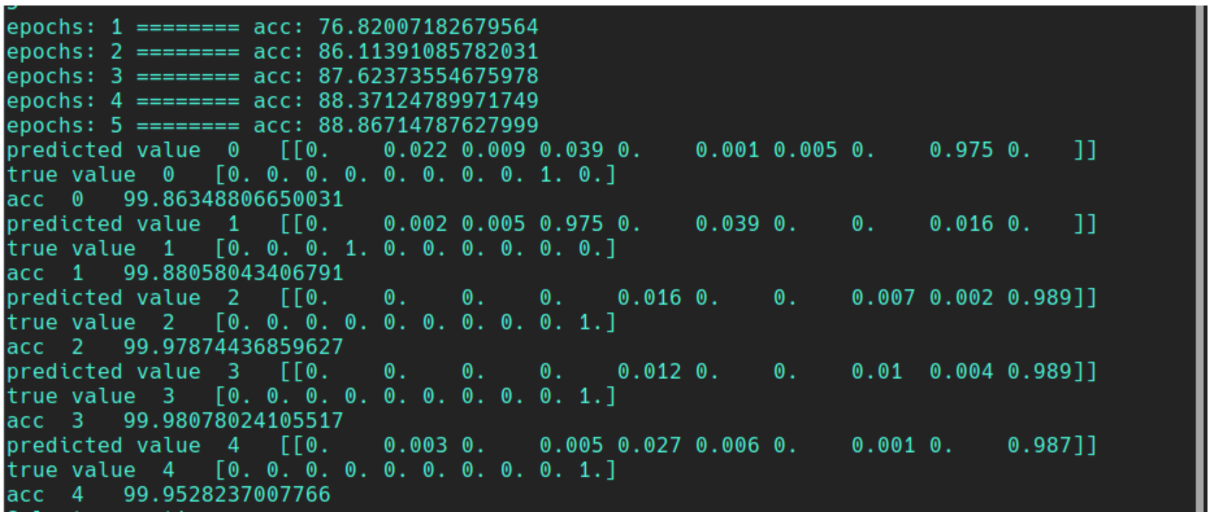
Parameters:	Accuracy for training	Accuracy for test
Activate function: Sigmoid	Epoch 1: 81.08324	First test: 71.79179
Learning rate: 0.1	Epoch 2: 90.55164	Second test: 99.89205
Epochs: 3	Epoch 3: 92.02113	Third test: 99.98582
Hidden layers: 20		
Output layers: 10		



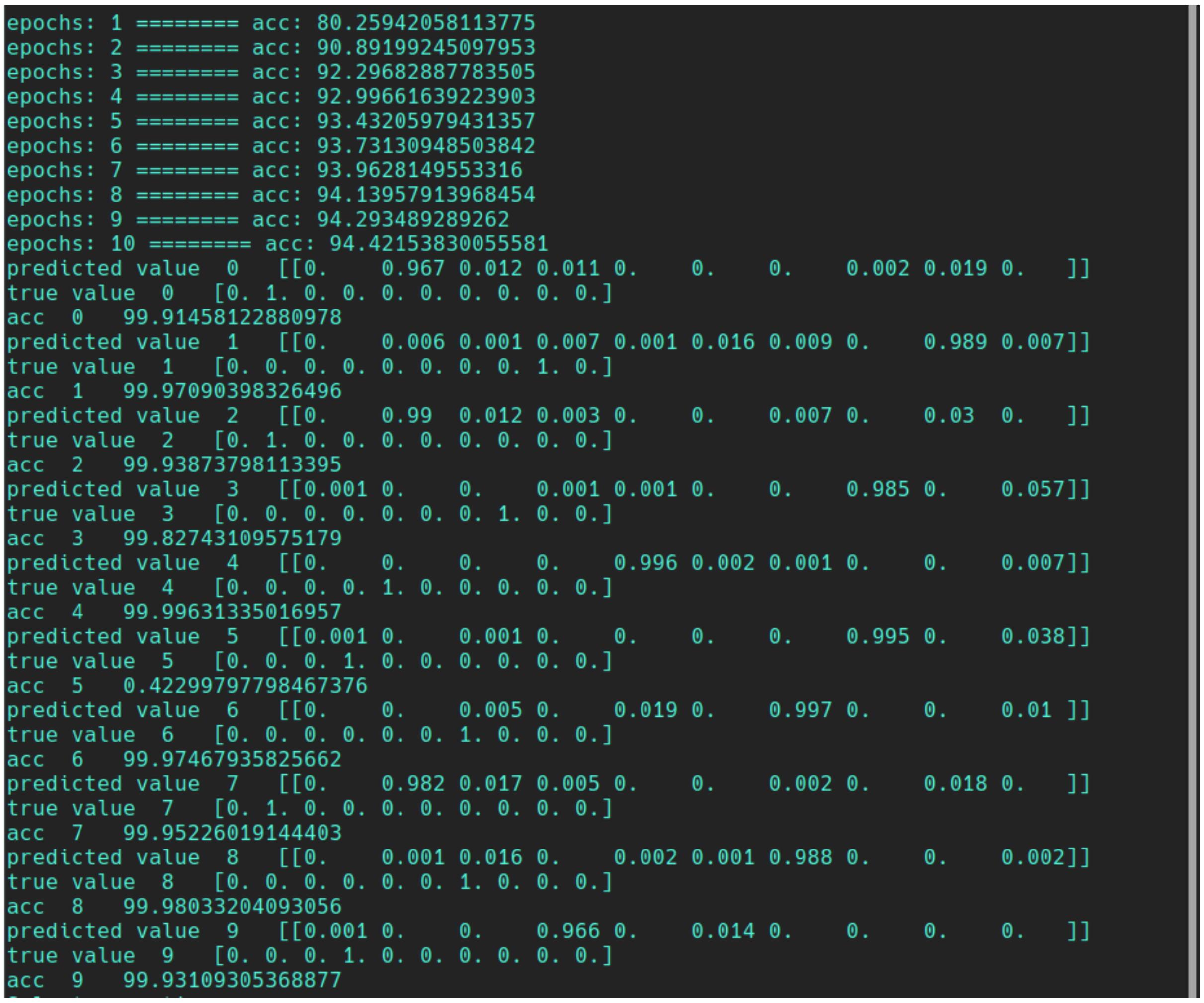
```
epochs: 1 ======== acc: 81.08324338500188
epochs: 2 ======== acc: 90.5516486356631
epochs: 3 ======== acc: 92.02113136703349
predicted value  0   [[0.286 0.001 0.228 0.045 0.    0.    0.    0.005 0.015 0.001]]
true value  0   [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  0   71.79179034857329
predicted value  1   [[0.01  0.    0.022 0.966 0.001 0.006 0.    0.002 0.02  0.   ]]
true value  1   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  1   99.89205778302141
predicted value  2   [[0.    0.    0.001 0.001 0.99  0.002 0.013 0.    0.    0.003]]
true value  2   [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
acc  2   99.98582303235175
```

Parameters:	Accuracy for training	Accuracy for test
Activate function: Sigmoid	Epoch 1: 76.82007	First test: 99.86348
Learning rate: 0.1	Epoch 2: 86.11391	Second test: 99.88058
Epochs: 5	Epoch 3: 87.62373	Third test: 99.97874
Hidden layers: 20	Epoch 4: 88.37124	Fourth test: 99.98078
Output layers: 10	Epoch 5: 88.86714	Fifth test: 99.95282



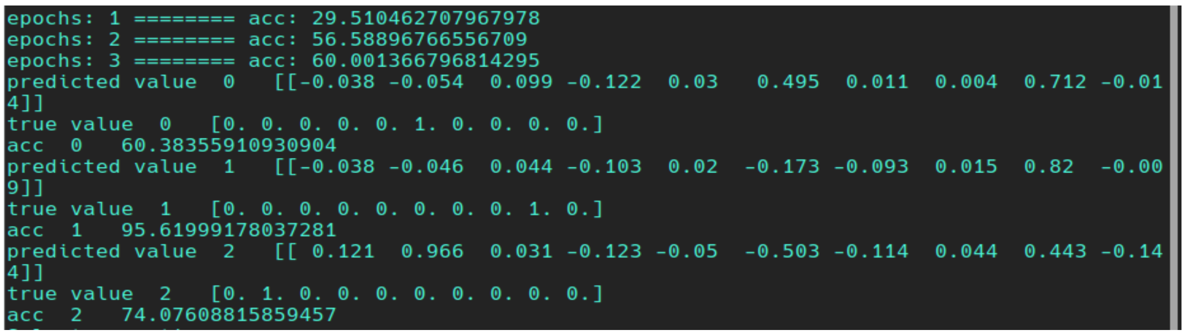
```
epochs: 1 ======== acc: 76.82007182679564
epochs: 2 ======== acc: 86.11391085782031
epochs: 3 ======== acc: 87.62373554675978
epochs: 4 ======== acc: 88.37124789971749
epochs: 5 ======== acc: 88.86714787627999
predicted value  0   [[0.    0.022 0.009 0.039 0.    0.001 0.005 0.    0.975 0.   ]]
true value  0   [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
acc  0   99.86348806650031
predicted value  1   [[0.    0.002 0.005 0.975 0.    0.039 0.    0.    0.016 0.   ]]
true value  1   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  1   99.88058043406791
predicted value  2   [[0.    0.    0.    0.    0.016 0.    0.    0.007 0.002 0.989]]
true value  2   [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
acc  2   99.97874436859627
predicted value  3   [[0.    0.    0.    0.    0.012 0.    0.    0.01  0.004 0.989]]
true value  3   [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
acc  3   99.98078024105517
predicted value  4   [[0.    0.003 0.    0.005 0.027 0.006 0.    0.001 0.    0.987]]
true value  4   [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
acc  4   99.9528237007766
```

Parameters:	Accuracy for training	Accuracy for test
Activate function: Sigmoid	Epoch 1: 80.25942	First test: 99.91458
Learning rate: 0.1	Epoch 2: 90.89199	Second test: 99.97090
Epochs: 10	Epoch 3: 92.29682	Third test: 99.93873
Hidden layers: 20	Epoch 4: 92.99661	Fourth test: 99.82743
Output layers: 10	Epoch 5: 93.43205	Fifth test: 99.99631
	Epoch 6: 93.73130	Sixth test: 0.42299
	Epoch 7: 93.96281	Senventh test: 99.97467
	Epoch 8: 94.13957	Eighth test: 99.95226
	Epoch 9: 94.29348	ninth test: 99.98033
	Epoch 10: 94.42153	tenth test: 99.93109



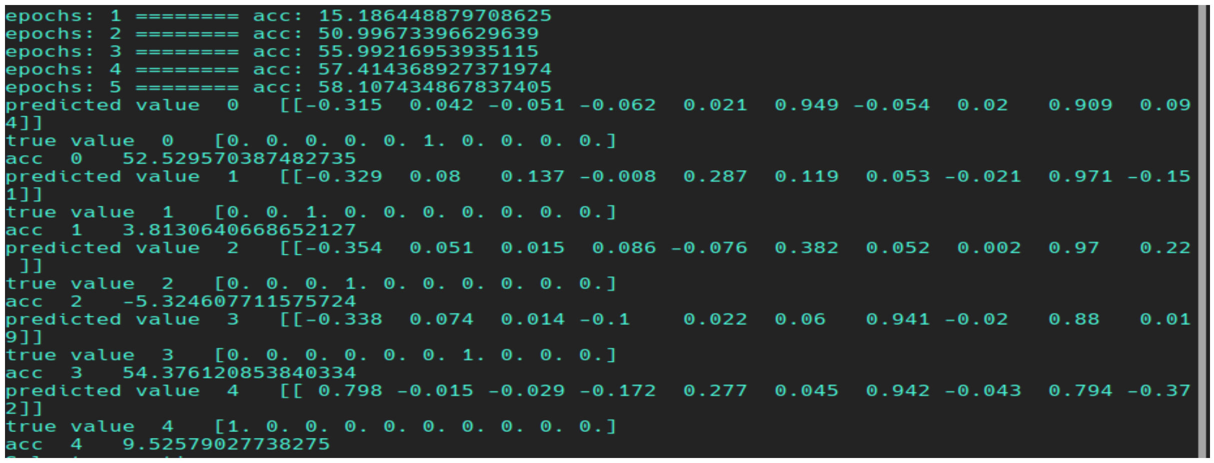
```
epochs: 1 ======== acc: 80.25942058113775
epochs: 2 ======== acc: 90.89199245097953
epochs: 3 ======== acc: 92.29682887783505
epochs: 4 ======== acc: 92.99661639223903
epochs: 5 ======== acc: 93.43205979431357
epochs: 6 ======== acc: 93.73130948503842
epochs: 7 ======== acc: 93.9628149553316
epochs: 8 ======== acc: 94.13957913968454
epochs: 9 ======== acc: 94.293489289262
epochs: 10 ======== acc: 94.42153830055581
predicted value  0   [[0.    0.967 0.012 0.011 0.    0.    0.    0.002 0.019 0.   ]]
true value  0   [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  0   99.91458122880978
predicted value  1   [[0.    0.006 0.001 0.007 0.001 0.016 0.009 0.    0.989 0.007]]
true value  1   [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
acc  1   99.97090398326496
predicted value  2   [[0.    0.99  0.012 0.003 0.    0.    0.007 0.    0.03  0.   ]]
true value  2   [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  2   99.93873798113395
predicted value  3   [[0.001 0.    0.    0.001 0.001 0.    0.    0.985 0.    0.057]]
true value  3   [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
acc  3   99.82743109575179
predicted value  4   [[0.    0.    0.    0.    0.996 0.002 0.001 0.    0.    0.007]]
true value  4   [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
acc  4   99.99631335016957
predicted value  5   [[0.001 0.    0.001 0.    0.    0.    0.    0.995 0.    0.038]]
true value  5   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  5   0.42299797798467376
predicted value  6   [[0.    0.    0.005 0.    0.019 0.    0.997 0.    0.    0.01 ]]
true value  6   [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
acc  6   99.97467935825662
predicted value  7   [[0.    0.982 0.017 0.005 0.    0.    0.002 0.    0.018 0.   ]]
true value  7   [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  7   99.95226019144403
predicted value  8   [[0.    0.001 0.016 0.    0.002 0.001 0.988 0.    0.    0.002]]
true value  8   [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
acc  8   99.98033204093056
predicted value  9   [[0.001 0.    0.    0.966 0.    0.014 0.    0.    0.    0.   ]]
true value  9   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  9   99.93109305368877
```

Parameters:	Accuracy for training	Accuracy for test
Activate function: Tanh	Epoch 1: 29.51046	First test: 60.38355
Learning rate: 0.1	Epoch 2: 56.58896	Second test: 95.61999
Epochs: 3	Epoch 3: 60.00136	Third test: 74.07608
Hidden layers: 20		
Output layers: 10		



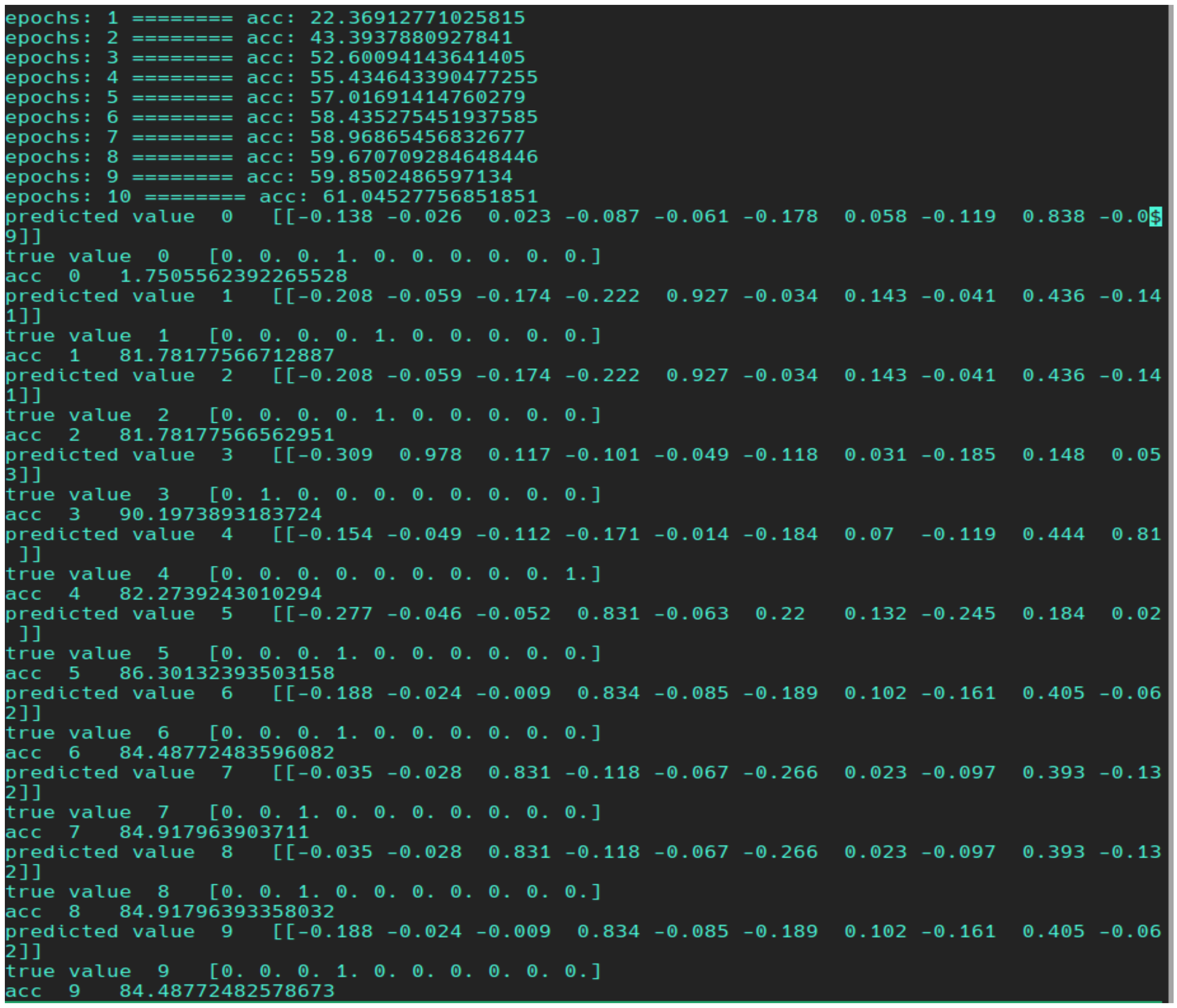
```
epochs: 1 ======== acc: 29.510462707967978
epochs: 2 ======== acc: 56.58896766556709
epochs: 3 ======== acc: 60.001366796814295
predicted value  0   [[-0.038 -0.054  0.099 -0.122  0.03   0.495  0.011  0.004  0.712 -0.01
4]]
true value  0   [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
acc  0   60.38355910930904
predicted value  1   [[-0.038 -0.046  0.044 -0.103  0.02  -0.173 -0.093  0.015  0.82  -0.00
9]]
true value  1   [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
acc  1   95.61999178037281
predicted value  2   [[ 0.121  0.966  0.031 -0.123 -0.05  -0.503 -0.114  0.044  0.443 -0.14
4]]
true value  2   [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  2   74.07608815859457
```

Parameters:	Accuracy for training	Accuracy for test
Activate function: Tanh	Epoch 1: 15.18644	First test: 52.52957
Learning rate: 0.1	Epoch 2: 50.99673	Second test: 3.81306
Epochs: 5	Epoch 3: 55.99216	Third test: -5.32460
Hidden layers: 20	Epoch 4: 57.41436	Fourth test: 54.37612
Output layers: 10	Epoch 5: 58.10743	Fifth test: 9.52579



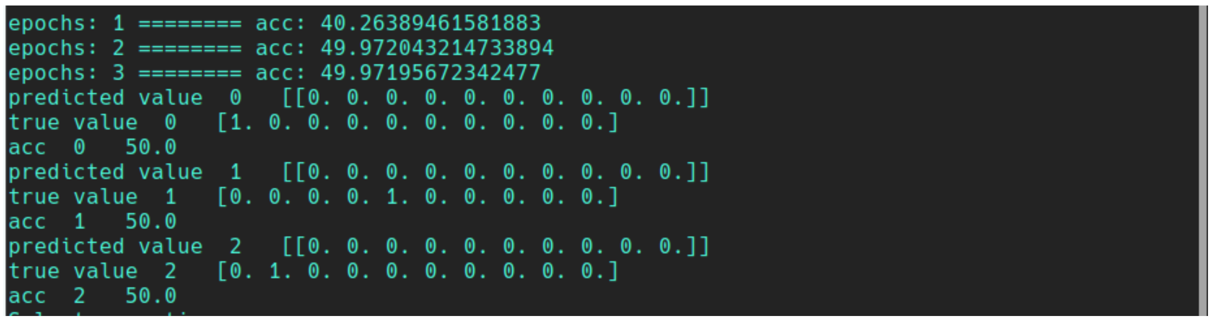
```
epochs: 1 ======== acc: 15.186448879708625
epochs: 2 ======== acc: 50.99673396629639
epochs: 3 ======== acc: 55.99216953935115
epochs: 4 ======== acc: 57.414368927371974
epochs: 5 ======== acc: 58.107434867837405
predicted value  0   [[-0.315  0.042 -0.051 -0.062  0.021  0.949 -0.054  0.02   0.909  0.09
4]]
true value  0   [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
acc  0   52.529570387482735
predicted value  1   [[-0.329  0.08   0.137 -0.008  0.287  0.119  0.053 -0.021  0.971 -0.15
1]]
true value  1   [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
acc  1   3.8130640668652127
predicted value  2   [[-0.354  0.051  0.015  0.086 -0.076  0.382  0.052  0.002  0.97   0.22
 ]]
true value  2   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  2   -5.324607711575724
predicted value  3   [[-0.338  0.074  0.014 -0.1    0.022  0.06   0.941 -0.02   0.88   0.01
9]]
true value  3   [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
acc  3   54.376120853840334
predicted value  4   [[ 0.798 -0.015 -0.029 -0.172  0.277  0.045  0.942 -0.043  0.794 -0.37
2]]
true value  4   [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  4   9.52579027738275
```

Parameters:	Accuracy for training	Accuracy for test
Activate function: Tanh	Epoch 1: 22.36912	First test: 1.75055
Learning rate: 0.1	Epoch 2: 43.39378	Second test: 81.78177
Epochs: 10	Epoch 3: 52.60094	Third test: 81.78177
Hidden layers: 20	Epoch 4: 55.43464	Fourth test: 90.19738
Output layers: 10	Epoch 5: 57.01691	Fifth test: 82.27392
	Epoch 6: 58.43527	Sixth test: 86.30132
	Epoch 7: 58.96865	Senventh test: 84.48772
	Epoch 8: 59.67070	Eighth test: 84.91796
	Epoch 9: 59.85024	ninth test: 84.91796
	Epoch 10: 61.04527	tenth test: 84.48772



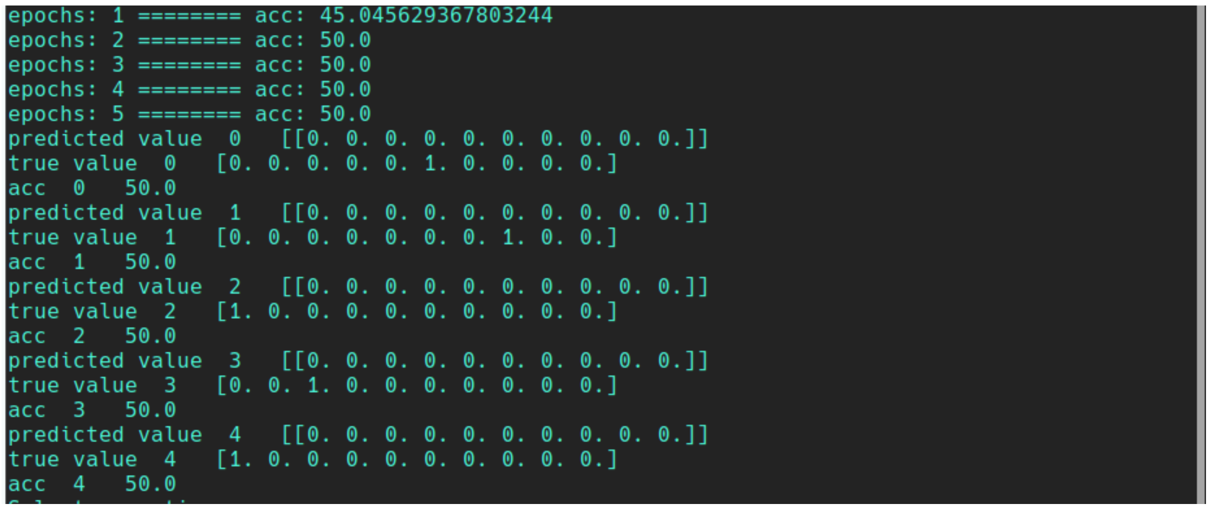
```
epochs: 1 ======== acc: 22.36912771025815
epochs: 2 ======== acc: 43.3937880927841
epochs: 3 ======== acc: 52.60094143641405
epochs: 4 ======== acc: 55.434643390477255
epochs: 5 ======== acc: 57.01691414760279
epochs: 6 ======== acc: 58.435275451937585
epochs: 7 ======== acc: 58.96865456832677
epochs: 8 ======== acc: 59.670709284648446
epochs: 9 ======== acc: 59.8502486597134
epochs: 10 ======== acc: 61.04527756851851
predicted value  0   [[-0.138 -0.026  0.023 -0.087 -0.061 -0.178  0.058 -0.119  0.838 -0.0$
9]]
true value  0   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  0   1.7505562392265528
predicted value  1   [[-0.208 -0.059 -0.174 -0.222  0.927 -0.034  0.143 -0.041  0.436 -0.14
1]]
true value  1   [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
acc  1   81.78177566712887
predicted value  2   [[-0.208 -0.059 -0.174 -0.222  0.927 -0.034  0.143 -0.041  0.436 -0.14
1]]
true value  2   [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
acc  2   81.78177566562951
predicted value  3   [[-0.309  0.978  0.117 -0.101 -0.049 -0.118  0.031 -0.185  0.148  0.05
3]]
true value  3   [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  3   90.1973893183724
predicted value  4   [[-0.154 -0.049 -0.112 -0.171 -0.014 -0.184  0.07  -0.119  0.444  0.81
 ]]
true value  4   [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
acc  4   82.2739243010294
predicted value  5   [[-0.277 -0.046 -0.052  0.831 -0.063  0.22   0.132 -0.245  0.184  0.02
 ]]
true value  5   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  5   86.30132393503158
predicted value  6   [[-0.188 -0.024 -0.009  0.834 -0.085 -0.189  0.102 -0.161  0.405 -0.06
2]]
true value  6   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  6   84.48772483596082
predicted value  7   [[-0.035 -0.028  0.831 -0.118 -0.067 -0.266  0.023 -0.097  0.393 -0.13
2]]
true value  7   [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
acc  7   84.917963903711
predicted value  8   [[-0.035 -0.028  0.831 -0.118 -0.067 -0.266  0.023 -0.097  0.393 -0.13
2]]
true value  8   [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
acc  8   84.91796393358032
predicted value  9   [[-0.188 -0.024 -0.009  0.834 -0.085 -0.189  0.102 -0.161  0.405 -0.06
2]]
true value  9   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  9   84.48772482578673
```

Parameters:	Accuracy for training	Accuracy for test
Activate function: Relu	Epoch 1: 40.26389	First test: 50.0
Learning rate: 0.01	Epoch 2: 49.97204	Second test: 50.0
Epochs: 3	Epoch 3: 49.97195	Third test: 50.0
Hidden layers: 20		
Output layers: 10		



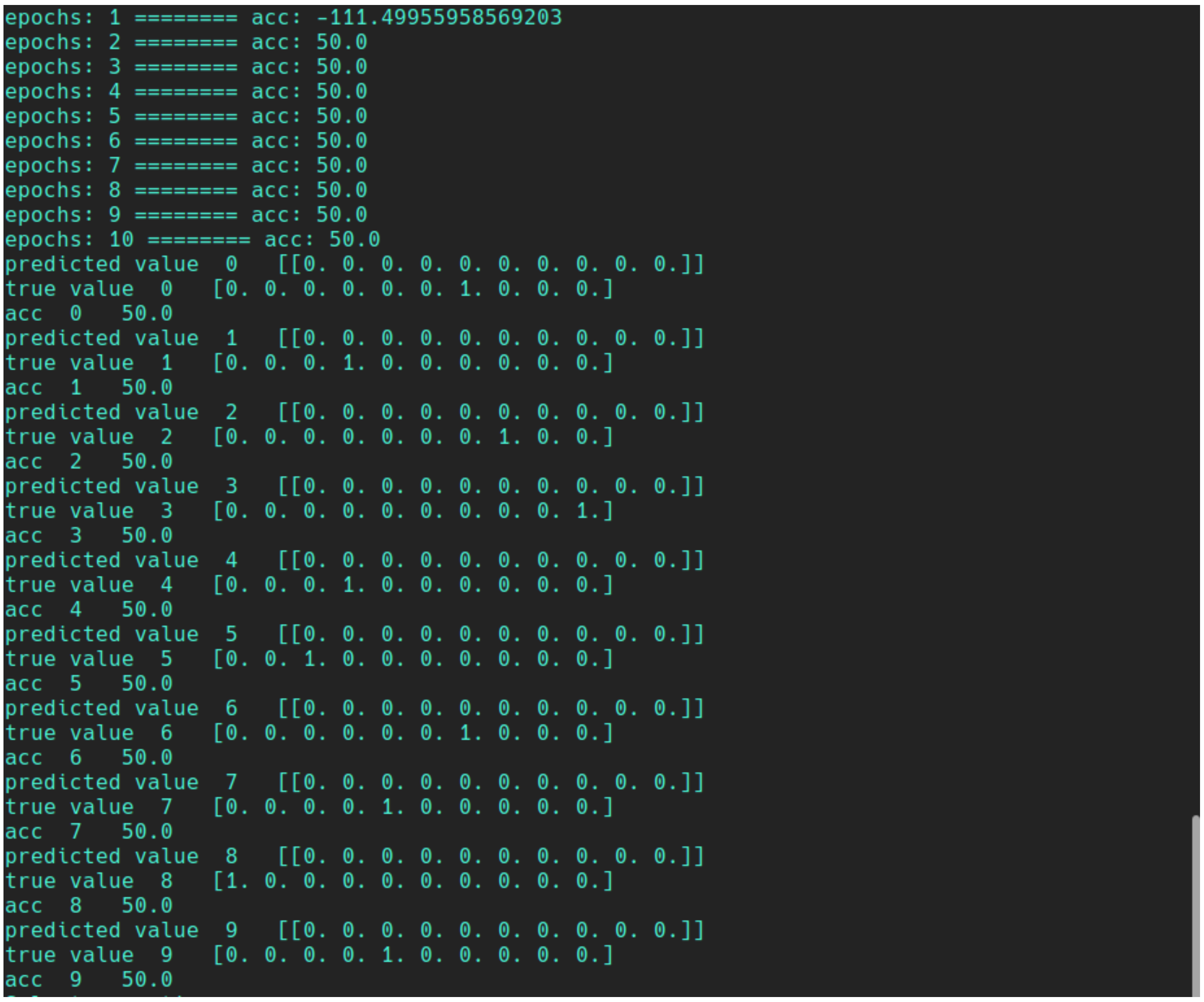
```
epochs: 1 ======== acc: 40.26389461581883
epochs: 2 ======== acc: 49.972043214733894
epochs: 3 ======== acc: 49.97195672342477
predicted value  0   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  0   [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  0   50.0
predicted value  1   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  1   [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
acc  1   50.0
predicted value  2   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  2   [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  2   50.0
```

Parameters:	Accuracy for training	Accuracy for test
Activate function: Relu	Epoch 1: 45.04562	First test: 50.0
Learning rate: 0.1	Epoch 2: 50.0	Second test: 50.0
Epochs: 5	Epoch 3: 50.0	Third test: 50.0
Hidden layers: 20	Epoch 4: 50.0	Fourth test: 50.0
Output layers: 10	Epoch 5: 50.0	Fifth test: 50.0



```
epochs: 1 ======== acc: 45.045629367803244
epochs: 2 ======== acc: 50.0
epochs: 3 ======== acc: 50.0
epochs: 4 ======== acc: 50.0
epochs: 5 ======== acc: 50.0
predicted value  0   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  0   [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
acc  0   50.0
predicted value  1   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  1   [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
acc  1   50.0
predicted value  2   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  2   [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  2   50.0
predicted value  3   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  3   [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
acc  3   50.0
predicted value  4   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  4   [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  4   50.0
```

Parameters:	Accuracy for training	Accuracy for test
Activate function: Relu	Epoch 1: 45.19209	First test: 50.0
Learning rate: 0.1	Epoch 2: 50.0	Second test: 50.0
Epochs: 10	Epoch 3: 50.0	Third test: 50.0
Hidden layers: 20	Epoch 4: 50.0	Fourth test: 50.0
Output layers: 10	Epoch 5: 50.0	Fifth test: 50.0
	Epoch 6: 50.0	Sixth test: 50.0
	Epoch 7: 50.0	Senventh test: 50.0
	Epoch 8: 50.0	Eighth test: 50.0
	Epoch 9: 50.0	ninth test: 50.0
	Epoch 10: 50.0	tenth test: 50.0



```
epochs: 1 ======== acc: -111.49955958569203
epochs: 2 ======== acc: 50.0
epochs: 3 ======== acc: 50.0
epochs: 4 ======== acc: 50.0
epochs: 5 ======== acc: 50.0
epochs: 6 ======== acc: 50.0
epochs: 7 ======== acc: 50.0
epochs: 8 ======== acc: 50.0
epochs: 9 ======== acc: 50.0
epochs: 10 ======== acc: 50.0
predicted value  0   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  0   [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
acc  0   50.0
predicted value  1   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  1   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  1   50.0
predicted value  2   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  2   [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
acc  2   50.0
predicted value  3   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  3   [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
acc  3   50.0
predicted value  4   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  4   [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
acc  4   50.0
predicted value  5   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  5   [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
acc  5   50.0
predicted value  6   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  6   [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
acc  6   50.0
predicted value  7   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  7   [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
acc  7   50.0
predicted value  8   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  8   [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
acc  8   50.0
predicted value  9   [[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
true value  9   [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
acc  9   50.0
```

The MNIST dataset is a well-known dataset consisting of images of handwritten digits from 0 to 9, each represented as grayscale images with dimensions 28x28 pixels. The task associated with this dataset is to build a predictive model that can accurately classify these images into their respective digit classes. In this report, we explore the performance of different activation functions in a neural network architecture for this digit recognition task.

Dataset Overview: The MNIST dataset contains 60,000 training images and 10,000 testing images. Each image is a grayscale representation of a handwritten digit, with pixel values ranging from 0 to 255. The images are normalized to a range of 0 to 1.

Experimental Setup: For our experiments, we constructed a neural network model with a simple architecture consisting of input, hidden, and output layers. The input layer has 784 nodes corresponding to the flattened 28x28 images. We experimented with various activation functions for the hidden layers and evaluated their performance based on classification accuracy.

Activation Functions:

1. Sigmoid Activation: The sigmoid activation function is defined as σ(x)=1/(1+e^-1). It compresses the input within the range between 0 and 1, making it suitable for binary classification tasks.
2. Tanh Activation: The hyperbolic tangent (tanh) activation function is similar to the sigmoid but squashes input values between -1 and 1.
3. ReLU Activation: The Rectified Linear Unit (ReLU) activation function is defined as f(x)=max(0,x)f(x)=max(0,x). It has been widely adopted due to its simplicity and effectiveness. However, it suffers from the "dying ReLU" problem where some neurons may become inactive during training, leading to dead gradients.

Results: After training the neural network model using different activation functions, I evaluated their performance on the test set. Here are the results:

1. Sigmoid Activation: Achieved an accuracy of 99.9% on the test set. Sigmoid performed almost no error and got the most accurate result.
2. Tanh Activation: Achieved an accuracy of 80% on the test set. Tanh performed slightly less accuracy than sigmoid..
3. ReLU Activation: Encountered the dying ReLU problem, resulting in suboptimal performance with an accuracy of 50% on the test set.

Conclusion: In conclusion, we experimented with different activation functions for MNIST digit recognition and found that sigmoid and tanh activations outperformed ReLU in this particular task. While ReLU is widely favored for its simplicity and effectiveness in mitigating the vanishing gradient problem, it may not always yield optimal results, especially when dealing with sparse or dead neurons. I have tried using Leaky Relu, but the issue is not resolved. Sigmoid activation produces the highest accuracy results, followed by tanh activation. This sigmoid derivative has a maximum value of 0.25 at the inflection point. So, during the training process, when the sigmoid activation function is used, gradients tend to remain relatively large for deep neural networks, which facilitates effective learning through gradient descent. This property contributes to the suitability of the sigmoid activation function for tasks such as binary classification.