2.3 Training loop

I expected an accuracy of about 50% which I achieved. Using a single linear classifier is like using a straight line to separate blue and red dots, which is not an adequate tool.

2.4 Multi-layer perceptron

I achieve a accuracy of approximately 100%. With multiple non linear layers, the network is able to create the desired circle shaped boundary

2.5 Feature transform

I changed the coordinate system to polar coordinates, since the classification of the points seemed to be dependent only on the radius, which turnded out to be correct. In contrast to the Multi-layer perceptron, It needs more epochs to converge.

3.3 Multi-layer perceptron

Classifying with a single layer already achieved an accuracy of 91%. However when parsing text from images, this might not be enough. Using multiple layers increased that accuract, however only by 3 % to 94%

3.4 Convolutional network

With the convolutional architecture I was able to achieve an accuracy of 98.37% with 5 epochs

3.5 Comparison of number of parameters

MLP

	Biases	Weights	Parameters
L1	32	32 * 784 = 25088	25120
L2	10	10 * 32 = 320	330
Total	42	25408	25450

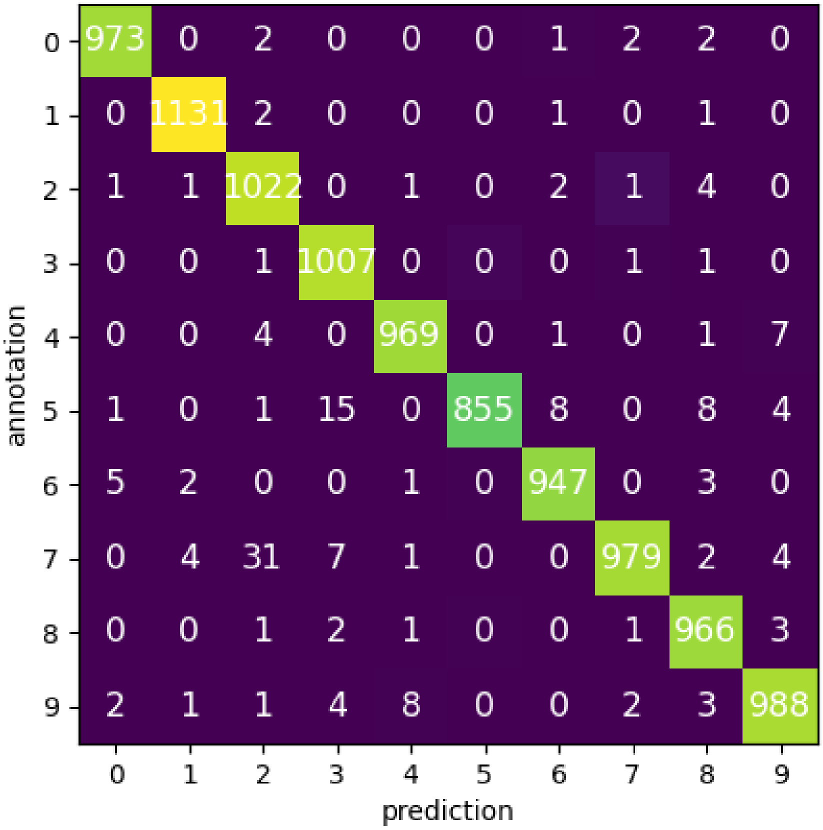
There is one bias per neuron and one 1 weight per input of every neuron. The results for the different layers can be seen in the table above. This leads to a total of 25450 weights.

Convolutional Network

Number of weights in a convolutional layer is according to the Slides C_out * (K * K * C_in + 1)

	Biases	Weights	Parameters
L1	0	8 * (3 * 3 * 1 + 1) = 80	80
L2	0	16 * (3 * 3 * 8 + 1) = 1168	1168
L3	0	32 * (3 * 3 * 16 + 1) = 4640	4640
L4	10	10 * 32 = 320	330
Total	10	6208	6218

3.6 Confusion matrix



annotation \ prediction	0	1	2	3	4	5	6	7	8	9
0	973	0	2	0	0	0	1	2	2	0
1	0	1131	2	0	0	0	1	0	1	0
2	1	1	1022	0	1	0	2	1	4	0
3	0	0	1	1007	0	0	0	1	1	0
4	0	0	4	0	969	0	1	0	1	7
5	1	0	1	15	0	855	8	0	8	4
6	5	2	0	0	1	0	947	0	3	0
7	0	4	31	7	1	0	0	979	2	4
8	0	0	1	2	1	0	0	1	966	3
9	2	1	1	4	8	0	0	2	3	988