Advanced methods in Machine Learning
Assignment 1

Isak Falk (16095283)

Due on 31st of January, 2017

Isak Falk (16095283)

Each part is laid out as follows.

(i) Hyperparameters

(ii) Plot of test and train errors through training

(iii) Output of test and train errors through training

(iv) Final test and train errors

(v) Confusion matrix

I will supply a makefile for loading and running the saved models and output the train and test errors for each model.

Part 1: MNIST with TensorFlow

For parts a, b, and c below I used the script `opt_hypreparams.py` to find the optimal hyperparameters, which in this case is the learning rate and the epoch when we stop training (early stopping to avoid overfitting). For each model I ran it for the learning rates $(0.1, 0.01, 0.001, 0.0001)$ over 100 epochs with batch size 200, using the validation set that comes with the mnist dataset in tensorflow I recorded the validation error at the end of each epoch. For each run I thus got the smallest validation error, and at what epoch this occurred. From the pairs (learning rate, optimal epoch) in each run together with the optimal validation error, I took the pair which yielded the smallest validation error for all runs. consisting of the following values for each hyperparameter.

There are some weaknesses with this hyperparameter optimization strategy. We are not optimising over all possible hyperparameters (learning rate, batch size, epochs), but only over learning rate and epochs. Also, in general random search is more efficient than grid search. However, for our purposes, I found that this optimisation gave great results which have to do with that the models here (1a, 1b, 1c, 1d) have a wide range of hyperparameters which yields good results.

A couple of observations is that all models chose 0.1 as the optimal learning rate. This has to do with the fact that TF is numerically stable enough to handle large learning rates. Since higher learning rates in general means we converge faster if we do converge, this hints that for learning rate 0.1 we converged over the 100 epochs, which maybe didn't happen for the lower learning rates. This could potentially mean that if we ran the models with a lower learning rate for a higher number of epochs, we could push the optimal validation error down further. However, the errors are very good and shows that each model improves upon the previous ones.

(a) 1 linear layer, followed by a softmax

Optimal hyperparameters

learning rate 0.1

epochs 85

Graph, confusion matrix and final errors

The title should be 'Error over iterations'.

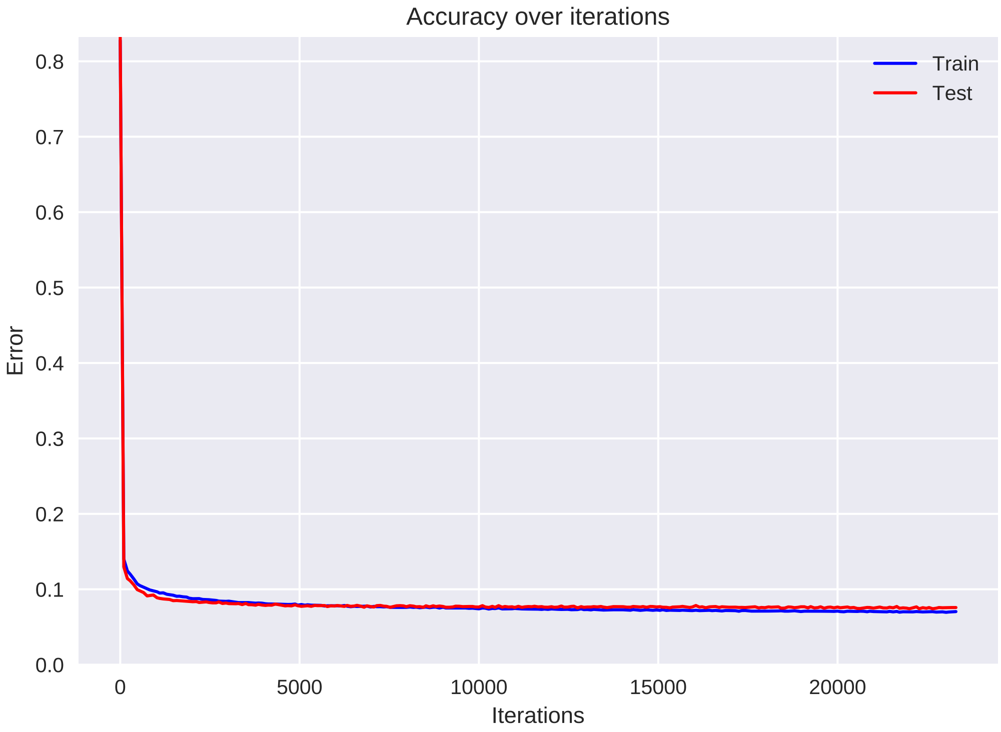
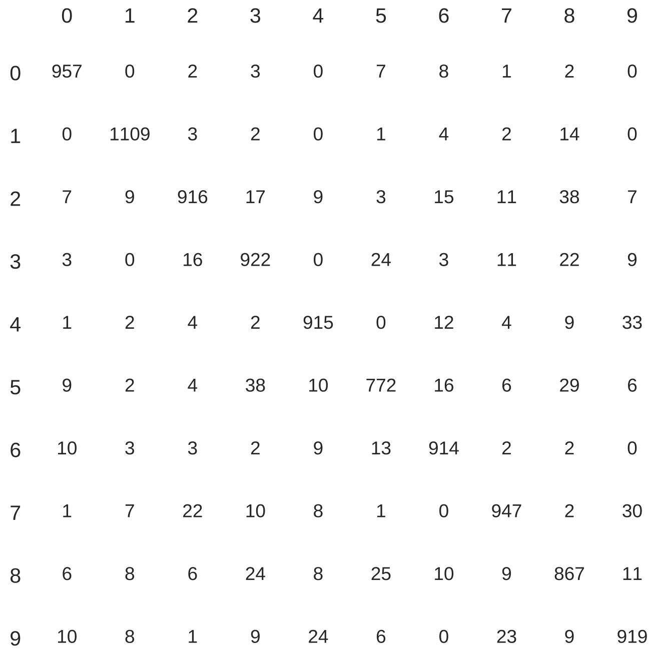


Figure 1: Error plot for model 1a



	0	1	2	3	4	5	6	7	8	9
0	957	0	2	3	0	7	8	1	2	0
1	0	1109	3	2	0	1	4	2	14	0
2	7	9	916	17	9	3	15	11	38	7
3	3	0	16	922	0	24	3	11	22	9
4	1	2	4	2	915	0	12	4	9	33
5	9	2	4	38	10	772	16	6	29	6
6	10	3	3	2	9	13	914	2	2	0
7	1	7	22	10	8	1	0	947	2	30
8	6	8	6	24	8	25	10	9	867	11
9	10	8	1	9	24	6	0	23	9	919

Figure 2: Confusion matrix for model 1a

Output of training and test error during training of the model

```
Epoch: 1, Train error: 0.12127, Test error: 0.11120
Epoch: 2, Train error: 0.10509, Test error: 0.09740
Epoch: 3, Train error: 0.09916, Test error: 0.09160
Epoch: 4, Train error: 0.09482, Test error: 0.08820
Epoch: 5, Train error: 0.09324, Test error: 0.08610
Epoch: 6, Train error: 0.09042, Test error: 0.08490
Epoch: 7, Train error: 0.08844, Test error: 0.08340
Epoch: 8, Train error: 0.08709, Test error: 0.08250
Epoch: 9, Train error: 0.08567, Test error: 0.08210
Epoch: 10, Train error: 0.08427, Test error: 0.08260
Epoch: 11, Train error: 0.08416, Test error: 0.08110
Epoch: 12, Train error: 0.08245, Test error: 0.08110
Epoch: 13, Train error: 0.08185, Test error: 0.08050
Epoch: 14, Train error: 0.08184, Test error: 0.08070
Epoch: 15, Train error: 0.08049, Test error: 0.07920
Epoch: 16, Train error: 0.08069, Test error: 0.07840
Epoch: 17, Train error: 0.07964, Test error: 0.07840
Epoch: 18, Train error: 0.07933, Test error: 0.07780
Epoch: 19, Train error: 0.07955, Test error: 0.07860
Epoch: 20, Train error: 0.07918, Test error: 0.07820
Epoch: 21, Train error: 0.07800, Test error: 0.07760
```

```
Epoch: 22, Train error: 0.07820, Test error: 0.07890
Epoch: 23, Train error: 0.07747, Test error: 0.07850
Epoch: 24, Train error: 0.07735, Test error: 0.07890
Epoch: 25, Train error: 0.07727, Test error: 0.07770
Epoch: 26, Train error: 0.07689, Test error: 0.07850
Epoch: 27, Train error: 0.07665, Test error: 0.07740
Epoch: 28, Train error: 0.07631, Test error: 0.07820
Epoch: 29, Train error: 0.07575, Test error: 0.07720
Epoch: 30, Train error: 0.07631, Test error: 0.07710
Epoch: 31, Train error: 0.07640, Test error: 0.07790
Epoch: 32, Train error: 0.07689, Test error: 0.07680
Epoch: 33, Train error: 0.07536, Test error: 0.07610
Epoch: 34, Train error: 0.07518, Test error: 0.07750
Epoch: 35, Train error: 0.07495, Test error: 0.07720
Epoch: 36, Train error: 0.07464, Test error: 0.07630
Epoch: 37, Train error: 0.07482, Test error: 0.07730
Epoch: 38, Train error: 0.07427, Test error: 0.07610
Epoch: 39, Train error: 0.07365, Test error: 0.07750
Epoch: 40, Train error: 0.07462, Test error: 0.07540
Epoch: 41, Train error: 0.07405, Test error: 0.07640
Epoch: 42, Train error: 0.07358, Test error: 0.07640
Epoch: 43, Train error: 0.07393, Test error: 0.07650
Epoch: 44, Train error: 0.07355, Test error: 0.07550
Epoch: 45, Train error: 0.07342, Test error: 0.07660
Epoch: 46, Train error: 0.07345, Test error: 0.07700
Epoch: 47, Train error: 0.07291, Test error: 0.07600
Epoch: 48, Train error: 0.07320, Test error: 0.07680
Epoch: 49, Train error: 0.07245, Test error: 0.07690
Epoch: 50, Train error: 0.07245, Test error: 0.07710
Epoch: 51, Train error: 0.07233, Test error: 0.07630
Epoch: 52, Train error: 0.07304, Test error: 0.07720
Epoch: 53, Train error: 0.07233, Test error: 0.07720
Epoch: 54, Train error: 0.07202, Test error: 0.07660
Epoch: 55, Train error: 0.07242, Test error: 0.07610
Epoch: 56, Train error: 0.07229, Test error: 0.07610
Epoch: 57, Train error: 0.07216, Test error: 0.07790
Epoch: 58, Train error: 0.07171, Test error: 0.07640
Epoch: 59, Train error: 0.07175, Test error: 0.07650
Epoch: 60, Train error: 0.07151, Test error: 0.07620
Epoch: 61, Train error: 0.07142, Test error: 0.07680
Epoch: 62, Train error: 0.07155, Test error: 0.07590
Epoch: 63, Train error: 0.07162, Test error: 0.07570
Epoch: 64, Train error: 0.07098, Test error: 0.07660
Epoch: 65, Train error: 0.07129, Test error: 0.07560
Epoch: 66, Train error: 0.07127, Test error: 0.07620
Epoch: 67, Train error: 0.07151, Test error: 0.07510
Epoch: 68, Train error: 0.07124, Test error: 0.07570
Epoch: 69, Train error: 0.07069, Test error: 0.07640
Epoch: 70, Train error: 0.07093, Test error: 0.07640
```

```
Epoch: 71, Train error: 0.07111, Test error: 0.07730
Epoch: 72, Train error: 0.07065, Test error: 0.07600
Epoch: 73, Train error: 0.07056, Test error: 0.07530
Epoch: 74, Train error: 0.07075, Test error: 0.07630
Epoch: 75, Train error: 0.07100, Test error: 0.07450
Epoch: 76, Train error: 0.07093, Test error: 0.07560
Epoch: 77, Train error: 0.07058, Test error: 0.07560
Epoch: 78, Train error: 0.07055, Test error: 0.07620
Epoch: 79, Train error: 0.06978, Test error: 0.07430
Epoch: 80, Train error: 0.06989, Test error: 0.07470
Epoch: 81, Train error: 0.07045, Test error: 0.07360
Epoch: 82, Train error: 0.07002, Test error: 0.07570
Epoch: 83, Train error: 0.06975, Test error: 0.07590
Epoch: 84, Train error: 0.06984, Test error: 0.07570
Epoch: 85, Train error: 0.06973, Test error: 0.07620
```

Final train error: 0.06973

Final test error: 0.07620

(b) 1 hidden layer (128 units) with a ReLU non-linearity, followed by a softmax

Optimal hyperparameters

learning rate 0.1

epochs 83

Graph, confusion matrix and final errors

The title should be 'Error over iterations'.

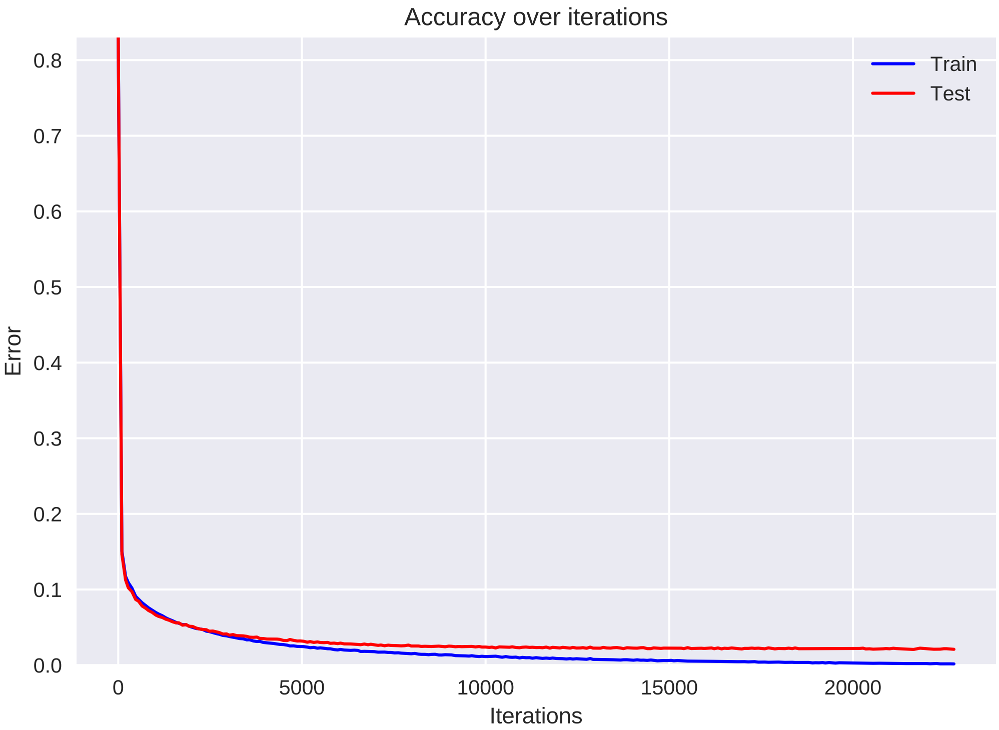
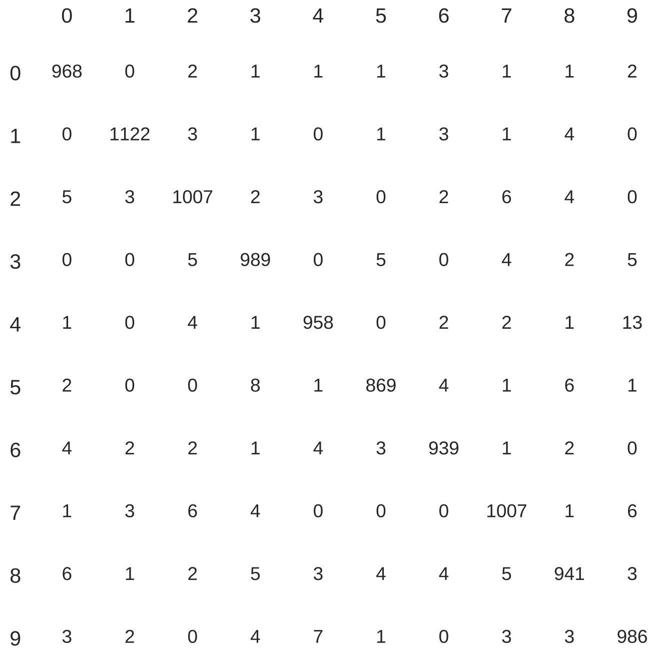


Figure 3: Error plot for model 1b



	0	1	2	3	4	5	6	7	8	9
0	968	0	2	1	1	1	3	1	1	2
1	0	1122	3	1	0	1	3	1	4	0
2	5	3	1007	2	3	0	2	6	4	0
3	0	0	5	989	0	5	0	4	2	5
4	1	0	4	1	958	0	2	2	1	13
5	2	0	0	8	1	869	4	1	6	1
6	4	2	2	1	4	3	939	1	2	0
7	1	3	6	4	0	0	0	1007	1	6
8	6	1	2	5	3	4	4	5	941	3
9	3	2	0	4	7	1	0	3	3	986

Figure 4: Confusion matrix for model 1b

Output of training and test error during training of the model

```
Epoch: 1, Train error: 0.11309, Test error: 0.10510
Epoch: 2, Train error: 0.08775, Test error: 0.08280
Epoch: 3, Train error: 0.07627, Test error: 0.07330
Epoch: 4, Train error: 0.06745, Test error: 0.06540
Epoch: 5, Train error: 0.06013, Test error: 0.05960
Epoch: 6, Train error: 0.05573, Test error: 0.05580
Epoch: 7, Train error: 0.05087, Test error: 0.05160
Epoch: 8, Train error: 0.04840, Test error: 0.04890
Epoch: 9, Train error: 0.04342, Test error: 0.04470
Epoch: 10, Train error: 0.04089, Test error: 0.04280
Epoch: 11, Train error: 0.03751, Test error: 0.03970
Epoch: 12, Train error: 0.03511, Test error: 0.03810
Epoch: 13, Train error: 0.03302, Test error: 0.03740
Epoch: 14, Train error: 0.03075, Test error: 0.03600
Epoch: 15, Train error: 0.02918, Test error: 0.03410
Epoch: 16, Train error: 0.02733, Test error: 0.03310
Epoch: 17, Train error: 0.02578, Test error: 0.03360
Epoch: 18, Train error: 0.02462, Test error: 0.03100
Epoch: 19, Train error: 0.02313, Test error: 0.03100
Epoch: 20, Train error: 0.02251, Test error: 0.02930
Epoch: 21, Train error: 0.02131, Test error: 0.02870
```

```
Epoch: 22, Train error: 0.02067, Test error: 0.02940
Epoch: 23, Train error: 0.01965, Test error: 0.02740
Epoch: 24, Train error: 0.01787, Test error: 0.02770
Epoch: 25, Train error: 0.01764, Test error: 0.02700
Epoch: 26, Train error: 0.01731, Test error: 0.02610
Epoch: 27, Train error: 0.01664, Test error: 0.02530
Epoch: 28, Train error: 0.01596, Test error: 0.02630
Epoch: 29, Train error: 0.01496, Test error: 0.02520
Epoch: 30, Train error: 0.01427, Test error: 0.02430
Epoch: 31, Train error: 0.01395, Test error: 0.02460
Epoch: 32, Train error: 0.01367, Test error: 0.02450
Epoch: 33, Train error: 0.01333, Test error: 0.02490
Epoch: 34, Train error: 0.01238, Test error: 0.02500
Epoch: 35, Train error: 0.01218, Test error: 0.02410
Epoch: 36, Train error: 0.01171, Test error: 0.02420
Epoch: 37, Train error: 0.01138, Test error: 0.02350
Epoch: 38, Train error: 0.01029, Test error: 0.02380
Epoch: 39, Train error: 0.01005, Test error: 0.02390
Epoch: 40, Train error: 0.01027, Test error: 0.02420
Epoch: 41, Train error: 0.00951, Test error: 0.02300
Epoch: 42, Train error: 0.00878, Test error: 0.02340
Epoch: 43, Train error: 0.00904, Test error: 0.02330
Epoch: 44, Train error: 0.00845, Test error: 0.02350
Epoch: 45, Train error: 0.00825, Test error: 0.02310
Epoch: 46, Train error: 0.00822, Test error: 0.02340
Epoch: 47, Train error: 0.00769, Test error: 0.02260
Epoch: 48, Train error: 0.00755, Test error: 0.02320
Epoch: 49, Train error: 0.00700, Test error: 0.02310
Epoch: 50, Train error: 0.00705, Test error: 0.02180
Epoch: 51, Train error: 0.00647, Test error: 0.02250
Epoch: 52, Train error: 0.00644, Test error: 0.02290
Epoch: 53, Train error: 0.00624, Test error: 0.02300
Epoch: 54, Train error: 0.00616, Test error: 0.02350
Epoch: 55, Train error: 0.00587, Test error: 0.02270
Epoch: 56, Train error: 0.00551, Test error: 0.02180
Epoch: 57, Train error: 0.00513, Test error: 0.02280
Epoch: 58, Train error: 0.00525, Test error: 0.02130
Epoch: 59, Train error: 0.00487, Test error: 0.02170
Epoch: 60, Train error: 0.00478, Test error: 0.02230
Epoch: 61, Train error: 0.00476, Test error: 0.02290
Epoch: 62, Train error: 0.00456, Test error: 0.02230
Epoch: 63, Train error: 0.00425, Test error: 0.02260
Epoch: 64, Train error: 0.00404, Test error: 0.02200
Epoch: 65, Train error: 0.00389, Test error: 0.02170
Epoch: 66, Train error: 0.00356, Test error: 0.02200
Epoch: 67, Train error: 0.00347, Test error: 0.02220
Epoch: 68, Train error: 0.00338, Test error: 0.02170
Epoch: 69, Train error: 0.00313, Test error: 0.02210
Epoch: 70, Train error: 0.00293, Test error: 0.02160
```

```
Epoch: 71, Train error: 0.00278, Test error: 0.02200
Epoch: 72, Train error: 0.00262, Test error: 0.02160
Epoch: 73, Train error: 0.00244, Test error: 0.02190
Epoch: 74, Train error: 0.00247, Test error: 0.02140
Epoch: 75, Train error: 0.00258, Test error: 0.02130
Epoch: 76, Train error: 0.00247, Test error: 0.02260
Epoch: 77, Train error: 0.00238, Test error: 0.02200
Epoch: 78, Train error: 0.00204, Test error: 0.02150
Epoch: 79, Train error: 0.00204, Test error: 0.02170
Epoch: 80, Train error: 0.00198, Test error: 0.02150
Epoch: 81, Train error: 0.00211, Test error: 0.02100
Epoch: 82, Train error: 0.00175, Test error: 0.02100
Epoch: 83, Train error: 0.00169, Test error: 0.02140
```

Final train error: 0.00169

Final test error: 0.02140

(c) 2 hidden layers (256 units) each, with ReLU non-linearity, follow by a softmax

Optimal hyperparameters

learning rate 0.1

epochs 86

Graph, confusion matrix and final errors

The title should be 'Error over iterations'.

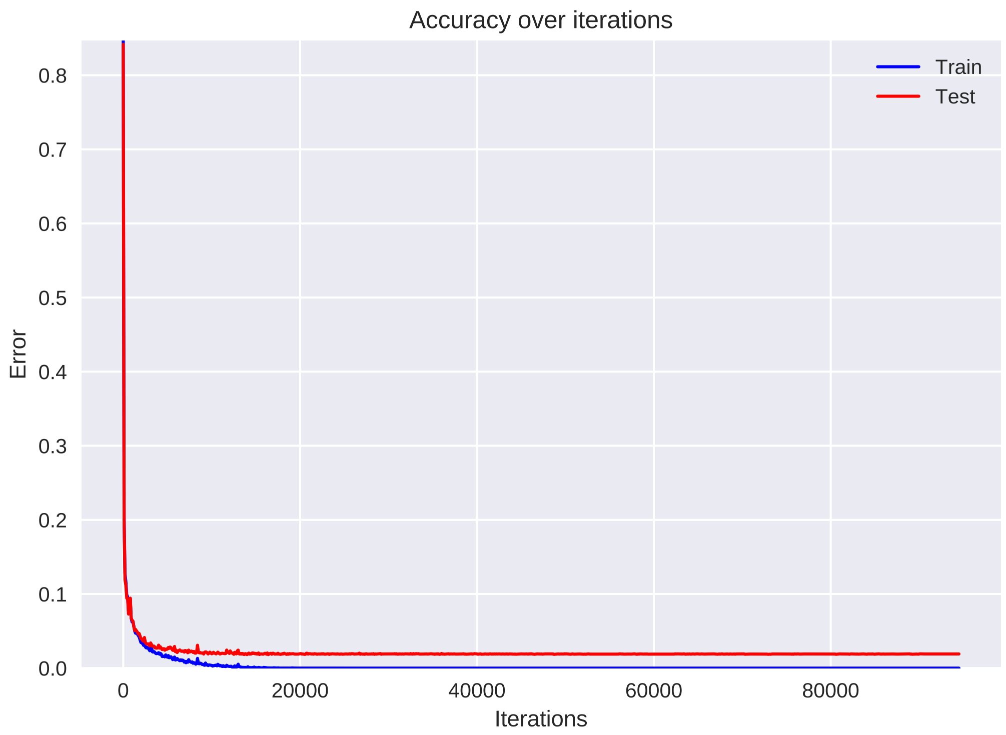
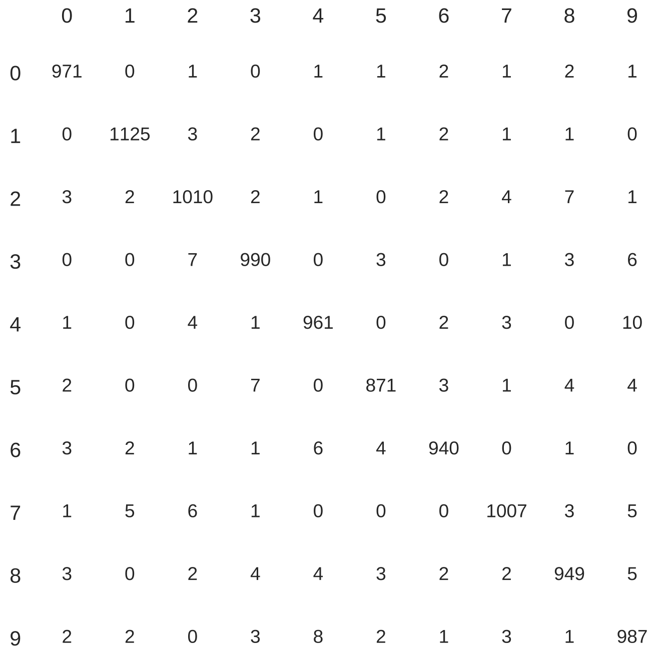


Figure 5: Error plot for model 1c



	0	1	2	3	4	5	6	7	8	9
0	971	0	1	0	1	1	2	1	2	1
1	0	1125	3	2	0	1	2	1	1	0
2	3	2	1010	2	1	0	2	4	7	1
3	0	0	7	990	0	3	0	1	3	6
4	1	0	4	1	961	0	2	3	0	10
5	2	0	0	7	0	871	3	1	4	4
6	3	2	1	1	6	4	940	0	1	0
7	1	5	6	1	0	0	0	1007	3	5
8	3	0	2	4	4	3	2	2	949	5
9	2	2	0	3	8	2	1	3	1	987

Figure 6: Confusion matrix for model 1c

Output of training and test error during training of the model

```
Epoch: 1, Train error: 0.06420, Test error: 0.06240
Epoch: 2, Train error: 0.03524, Test error: 0.03820
Epoch: 3, Train error: 0.02255, Test error: 0.03030
Epoch: 4, Train error: 0.01562, Test error: 0.02520
Epoch: 5, Train error: 0.01220, Test error: 0.02380
Epoch: 6, Train error: 0.00991, Test error: 0.02260
Epoch: 7, Train error: 0.00800, Test error: 0.02310
Epoch: 8, Train error: 0.00636, Test error: 0.02030
Epoch: 9, Train error: 0.00411, Test error: 0.02080
Epoch: 10, Train error: 0.00364, Test error: 0.01910
Epoch: 11, Train error: 0.00245, Test error: 0.02200
Epoch: 12, Train error: 0.00173, Test error: 0.01900
Epoch: 13, Train error: 0.00069, Test error: 0.01850
Epoch: 14, Train error: 0.00071, Test error: 0.01820
Epoch: 15, Train error: 0.00035, Test error: 0.01900
Epoch: 16, Train error: 0.00011, Test error: 0.01880
Epoch: 17, Train error: 0.00013, Test error: 0.01890
Epoch: 18, Train error: 0.00005, Test error: 0.01880
Epoch: 19, Train error: 0.00005, Test error: 0.01870
Epoch: 20, Train error: 0.00004, Test error: 0.01910
Epoch: 21, Train error: 0.00000, Test error: 0.01940
```

```
Epoch: 22, Train error: 0.00000, Test error: 0.01930
Epoch: 23, Train error: 0.00000, Test error: 0.01900
Epoch: 24, Train error: 0.00000, Test error: 0.01950
Epoch: 25, Train error: 0.00000, Test error: 0.01940
Epoch: 26, Train error: 0.00000, Test error: 0.01890
Epoch: 27, Train error: 0.00000, Test error: 0.01940
Epoch: 28, Train error: 0.00000, Test error: 0.01900
Epoch: 29, Train error: 0.00000, Test error: 0.01950
Epoch: 30, Train error: 0.00000, Test error: 0.01920
Epoch: 31, Train error: 0.00000, Test error: 0.01910
Epoch: 32, Train error: 0.00000, Test error: 0.01850
Epoch: 33, Train error: 0.00000, Test error: 0.01880
Epoch: 34, Train error: 0.00000, Test error: 0.01940
Epoch: 35, Train error: 0.00000, Test error: 0.01940
Epoch: 36, Train error: 0.00000, Test error: 0.01930
Epoch: 37, Train error: 0.00000, Test error: 0.01900
Epoch: 38, Train error: 0.00000, Test error: 0.01930
Epoch: 39, Train error: 0.00000, Test error: 0.01920
Epoch: 40, Train error: 0.00000, Test error: 0.01880
Epoch: 41, Train error: 0.00000, Test error: 0.01900
Epoch: 42, Train error: 0.00000, Test error: 0.01920
Epoch: 43, Train error: 0.00000, Test error: 0.01910
Epoch: 44, Train error: 0.00000, Test error: 0.01900
Epoch: 45, Train error: 0.00000, Test error: 0.01910
Epoch: 46, Train error: 0.00000, Test error: 0.01860
Epoch: 47, Train error: 0.00000, Test error: 0.01900
Epoch: 48, Train error: 0.00000, Test error: 0.01890
Epoch: 49, Train error: 0.00000, Test error: 0.01900
Epoch: 50, Train error: 0.00000, Test error: 0.01880
Epoch: 51, Train error: 0.00000, Test error: 0.01930
Epoch: 52, Train error: 0.00000, Test error: 0.01880
Epoch: 53, Train error: 0.00000, Test error: 0.01880
Epoch: 54, Train error: 0.00000, Test error: 0.01880
Epoch: 55, Train error: 0.00000, Test error: 0.01880
Epoch: 56, Train error: 0.00000, Test error: 0.01910
Epoch: 57, Train error: 0.00000, Test error: 0.01910
Epoch: 58, Train error: 0.00000, Test error: 0.01910
Epoch: 59, Train error: 0.00000, Test error: 0.01920
Epoch: 60, Train error: 0.00000, Test error: 0.01920
Epoch: 61, Train error: 0.00000, Test error: 0.01860
Epoch: 62, Train error: 0.00000, Test error: 0.01910
Epoch: 63, Train error: 0.00000, Test error: 0.01920
Epoch: 64, Train error: 0.00000, Test error: 0.01870
Epoch: 65, Train error: 0.00000, Test error: 0.01910
Epoch: 66, Train error: 0.00000, Test error: 0.01880
Epoch: 67, Train error: 0.00000, Test error: 0.01900
Epoch: 68, Train error: 0.00000, Test error: 0.01900
Epoch: 69, Train error: 0.00000, Test error: 0.01920
Epoch: 70, Train error: 0.00000, Test error: 0.01910
```

```
Epoch: 71, Train error: 0.00000, Test error: 0.01880
Epoch: 72, Train error: 0.00000, Test error: 0.01880
Epoch: 73, Train error: 0.00000, Test error: 0.01900
Epoch: 74, Train error: 0.00000, Test error: 0.01900
Epoch: 75, Train error: 0.00000, Test error: 0.01910
Epoch: 76, Train error: 0.00000, Test error: 0.01920
Epoch: 77, Train error: 0.00000, Test error: 0.01920
Epoch: 78, Train error: 0.00000, Test error: 0.01900
Epoch: 79, Train error: 0.00000, Test error: 0.01900
Epoch: 80, Train error: 0.00000, Test error: 0.01910
Epoch: 81, Train error: 0.00000, Test error: 0.01910
Epoch: 82, Train error: 0.00000, Test error: 0.01920
Epoch: 83, Train error: 0.00000, Test error: 0.01890
Epoch: 84, Train error: 0.00000, Test error: 0.01890
Epoch: 85, Train error: 0.00000, Test error: 0.01900
Epoch: 86, Train error: 0.00000, Test error: 0.01890
```

Final train error: 0.00000

Final test error: 0.01890

(d) 3 layer convolutional model, followed by a softmax

Optimal hyperparameters

learning rate 0.1

epochs 53

Graph, confusion matrix and final errors

The title should be 'Error over iterations'.

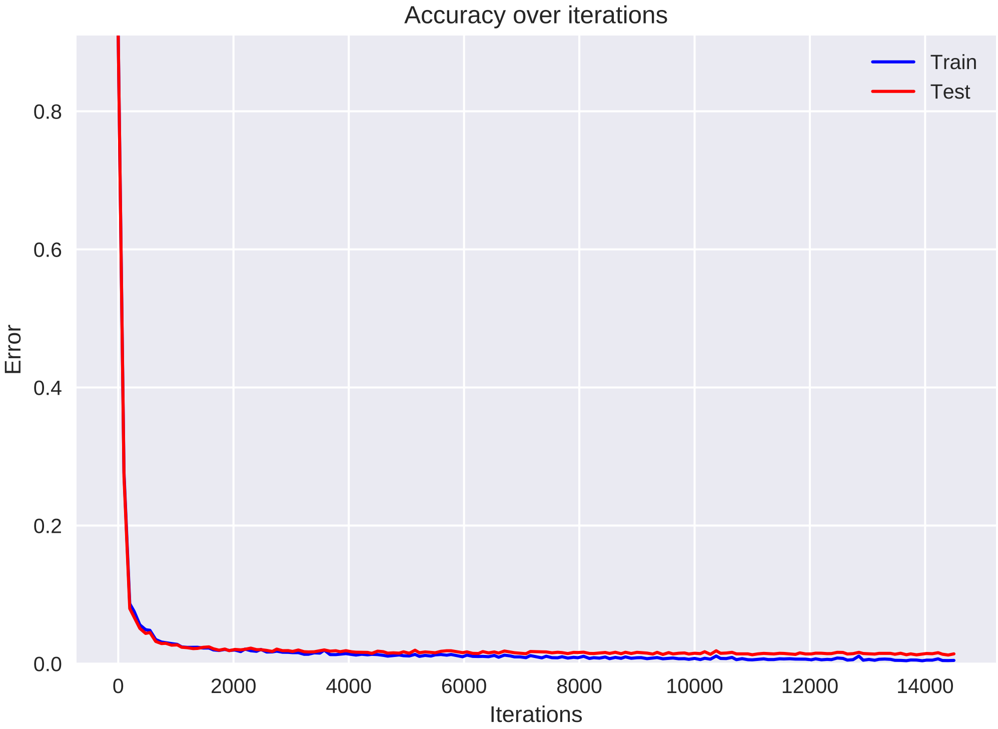
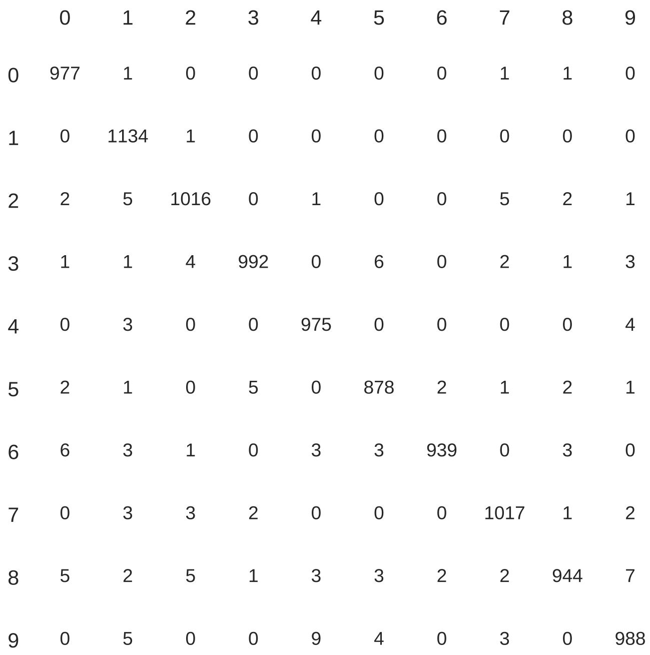


Figure 7: Error plot for model 1d



	0	1	2	3	4	5	6	7	8	9
0	977	1	0	0	0	0	0	1	1	0
1	0	1134	1	0	0	0	0	0	0	0
2	2	5	1016	0	1	0	0	5	2	1
3	1	1	4	992	0	6	0	2	1	3
4	0	3	0	0	975	0	0	0	0	4
5	2	1	0	5	0	878	2	1	2	1
6	6	3	1	0	3	3	939	0	3	0
7	0	3	3	2	0	0	0	1017	1	2
8	5	2	5	1	3	3	2	2	944	7
9	0	5	0	0	9	4	0	3	0	988

Figure 8: Confusion matrix for model 1d

```
Epoch: 1, Train error: 0.08809, Test error: 0.07890
Epoch: 2, Train error: 0.04680, Test error: 0.04010
Epoch: 3, Train error: 0.03102, Test error: 0.02960
Epoch: 4, Train error: 0.02504, Test error: 0.02430
Epoch: 5, Train error: 0.02236, Test error: 0.02230
Epoch: 6, Train error: 0.01991, Test error: 0.02050
Epoch: 7, Train error: 0.01978, Test error: 0.02060
Epoch: 8, Train error: 0.01780, Test error: 0.01820
Epoch: 9, Train error: 0.01973, Test error: 0.02090
Epoch: 10, Train error: 0.01562, Test error: 0.01820
Epoch: 11, Train error: 0.01549, Test error: 0.01910
Epoch: 12, Train error: 0.01367, Test error: 0.01800
Epoch: 13, Train error: 0.01782, Test error: 0.01840
Epoch: 14, Train error: 0.01325, Test error: 0.01580
Epoch: 15, Train error: 0.01309, Test error: 0.01710
Epoch: 16, Train error: 0.01635, Test error: 0.01710
Epoch: 17, Train error: 0.01127, Test error: 0.01570
Epoch: 18, Train error: 0.01184, Test error: 0.01700
Epoch: 19, Train error: 0.01149, Test error: 0.01500
Epoch: 20, Train error: 0.01229, Test error: 0.01540
Epoch: 21, Train error: 0.01065, Test error: 0.01670
Epoch: 22, Train error: 0.01011, Test error: 0.01360
Epoch: 23, Train error: 0.01169, Test error: 0.01620
```

```
Epoch: 24, Train error: 0.00891, Test error: 0.01550
Epoch: 25, Train error: 0.01036, Test error: 0.01630
Epoch: 26, Train error: 0.01018, Test error: 0.01790
Epoch: 27, Train error: 0.00958, Test error: 0.01660
Epoch: 28, Train error: 0.01213, Test error: 0.01960
Epoch: 29, Train error: 0.00865, Test error: 0.01570
Epoch: 30, Train error: 0.00840, Test error: 0.01440
Epoch: 31, Train error: 0.00753, Test error: 0.01530
Epoch: 32, Train error: 0.00858, Test error: 0.01460
Epoch: 33, Train error: 0.00775, Test error: 0.01510
Epoch: 34, Train error: 0.00745, Test error: 0.01440
Epoch: 35, Train error: 0.00795, Test error: 0.01380
Epoch: 36, Train error: 0.00669, Test error: 0.01490
Epoch: 37, Train error: 0.00611, Test error: 0.01450
Epoch: 38, Train error: 0.00993, Test error: 0.01910
Epoch: 39, Train error: 0.00856, Test error: 0.01580
Epoch: 40, Train error: 0.00600, Test error: 0.01270
Epoch: 41, Train error: 0.00596, Test error: 0.01460
Epoch: 42, Train error: 0.00825, Test error: 0.01560
Epoch: 43, Train error: 0.00580, Test error: 0.01430
Epoch: 44, Train error: 0.00985, Test error: 0.01720
Epoch: 45, Train error: 0.00567, Test error: 0.01380
Epoch: 46, Train error: 0.00535, Test error: 0.01400
Epoch: 47, Train error: 0.00527, Test error: 0.01460
Epoch: 48, Train error: 0.00747, Test error: 0.01570
Epoch: 49, Train error: 0.00522, Test error: 0.01360
Epoch: 50, Train error: 0.00520, Test error: 0.01370
Epoch: 51, Train error: 0.00547, Test error: 0.01550
Epoch: 52, Train error: 0.00400, Test error: 0.01400
Epoch: 53, Train error: 0.00707, Test error: 0.01400
```

Final train error: 0.00707

Final test error: 0.01400

Part 2: MNIST without TensorFlow

Note: I have by convention put the models of TF and numpy (implementation) in a correspondence such that throughout the document I call the first model model a, the second model model b and so on. This also means that my code follows this convention. This also means that I name the models 1a, 1b, 1c and 1d and for part 2 I name them model 2a, 2b, 2c, 2d, where the models (1a, 2a), (1b, 2b), (1c, 2c), (1d, 2d) are the same in terms of architecture.

Consequentially, we instead have Problems like P2:b which corresponds to implementing model 2a and similarly for P2:c, P2:d and P2:e.

(a) Compute the folowing derivatives

We first introduce some notation. We let $\{x_i, y_i\}$ be the data set with y_i being the digit and let t_i be the one-shot vector of example i. For an arbitrary module we let x be the input, z be the linear transformation $z = Wx + b$ and $y = \sigma(z)$, the non-linear mapping. For the final module we let p be the predicted probability vector over digits. In the examples below we drop the index on i and just write them as x, z, y, t, p.

We have that

$$loss = \sum_{i=1}^{N} -\log(p(y_i|x_i)) = \sum_{i=1}^{N} L_i$$

Since derivatives are linear operators, we only care about the derivative of L_i since if we know that, we know the derivative of the *loss*. We write L_i as L as we drop the dependency on the example for the time being.

In order to stay consistent with vector notation, we index the classes from 1 to 10 meaning that 0 is in the first class, 1 in the second and so on. We can shift this back easily if we want to.

(i) Derivative of the loss function wrt. the scores z: $\frac{\partial L}{\partial z}$.

We first calculate the element-wise partial derivative of the loss with regards to the entry z_j, the j'th entry of the output logits in the final layer of the input x. Then

$$\begin{aligned}\frac{\partial L}{\partial z_j} &= \frac{\partial \log\left(\sum_{c=1}^{10} \exp(z_c)\right)}{\partial z_j} - \frac{\partial z_y}{\partial z_j} \\ &= \frac{\exp(z_j)}{\sum_{c=1}^{10} \exp(z_c)} - \delta(j = y)\end{aligned}$$

Vectorizing this we have that the gradients can be written as

$$\frac{\partial L}{\partial z} = (p - t)^T$$

where p is the predicted class probabilities given the input x and t is the one-hot vector of the true label.

(ii) Given the model in (P1:a), compute the derivative of the loss wrt input x, derivative of the loss with to the layers parameters W , b.

We have the following explicit relationships from the modules.

$$
\begin{aligned}
z &= Wx + b \implies \\
z_i &= \sum_{j=1}^{n} W_{ij} x_j + b_i \\
p &= \exp(z)/sum(\exp(z)) \implies \\
p_i &= \frac{\exp(z_i)}{\sum_{c=1}^{10} \exp(z_c)}
\end{aligned}
$$

a.

$$
\begin{aligned}
\frac{\partial L}{\partial x_i} &= \frac{\partial L(z_1(x_i), \ldots, z_{10}(x_i))}{\partial x_i} \\
&= \sum_{c=1}^{10} \frac{\partial L}{\partial z_c} \cdot \frac{\partial z_c}{\partial x_i} \\
&= \sum_{c=1}^{10} (p_c - \delta(y = c)) \sum_{j=1}^{784} (W_{cj} \frac{\partial x_j}{\partial x_i} + \frac{\partial b_c}{\partial x_i}) \\
&= \sum_{c=1}^{10} (p_c - \delta(y = c)) W_{ci} \\
&= [(p - t)^T W]_i
\end{aligned}
$$

Hence we can vectorize this as

$$
\frac{\partial L}{\partial x} = (p - t)^T W
$$

b.

$$
\begin{aligned}
\frac{\partial L}{\partial W_{ij}} &= \frac{\partial L(z_1(W_{ij}), \ldots, z_{10}(W_{ij}))}{\partial W_{ij}} \\
&= \sum_{c=1}^{10} \frac{\partial L}{\partial z_c} \cdot \frac{\partial z_c}{\partial W_{ij}} \\
&= \sum_{c=1}^{10} (p_c - \delta(y = c)) \sum_{l=1}^{784} (\frac{\partial W_{cl}}{\partial W_{ij}} x_l + \frac{\partial b_c}{\partial W_{ij}}) \\
&= \sum_{c=1}^{10} (p_c - \delta(y = c)) x_j \delta(c = i, l = j) \\
&= (p_i - \delta(y = i)) x_j
\end{aligned}
$$

We can write this matrix as

$$\frac{\partial L}{\partial W}^T = \begin{bmatrix} (p_1-\delta(y=1))x_1 & (p_1-\delta(y=1))x_2 & \dots & (p_1-\delta(y=1))x_{784} \\ (p_2-\delta(y=2))x_1 & \ddots & & \vdots \\ \vdots & & & \vdots \\ (p_{10}-\delta(y=10))x_1 & \dots & \dots & (p_{10}-\delta(y=))x_{784} \end{bmatrix}$$

We recognise that this is just an outer product such that we can write

$$\frac{\partial L}{\partial W} = x(p-t)^T$$

c.

$$\begin{aligned} \frac{\partial L}{\partial b_i} &= \frac{\partial L(z_1(b_i),\dots,z_{10}(b_i))}{\partial b_i} \\ &= \sum_{c=1}^{10}(p_c-\delta(y=c))\sum_{l=1}^{784}(\frac{\partial W_{cl}x_l}{\partial b_i}+\frac{\partial b_c}{\partial b_i}) \\ &= \sum_{c=1}^{10}(p_c-\delta(y=c))\delta(c=i) \\ &= (p_i-\delta(y=i)) \end{aligned}$$

This is the same as for $\frac{\partial L}{\partial z}$ so we may vectorize this as

$$\frac{\partial L}{\partial b} = (p-t)^T$$

(iii) Compute the derivative of a convolution layer wrt. to its parameters W and wrt. to its input (4-dim tensor).

We introduce some notation: We let the input to a 3x3x16 conv2d layer be denoted by I, where I is a tensor of dimensions (E,N,N,D), where the quadruplet by convention represent for all tensors $(example, height, width, depth)$ where we start counting from the upper left front corner for the input of each example. We let an element of this input be indexed by $I(e,x,y,d)$. Similarly we let the output of this layer be denoted by C, where C is a tensor of dimensions (E,N,N,Z) where the width and height of C and I are equal, but the depths are independent of each other.

For this 3x3x16 conv2d layer we let the weight which maps $I(\cdot,\cdot,\cdot,\cdot)$ to $C(\cdot,\cdot,\cdot,z)$ be denoted by W^z, where W^z is a tensor of dimension $(3,3,D)$, and there will be 16 of these weights, $\{W^1,\dots,W^{16}\}$, one for each output filter.

For convolution, we let $K = I * W$ denote the image I convoluted with the kernel W. Specifically in our case, we have that the convolution $K(x,y) = \sum_{a,b\in\{-1,0,1\}} I(x-a,y-b)W(a,b)$ with padding such that K is of the same dimension of I, i.e. we let
$W(\cdot,0) = W(\cdot,N+1) = W(0,\cdot) = W(N+1,\cdot) = 0$.

For our case when the number of channels are greater than one, we have that the output can be written as

$$C(e,x,y,z) = \sum_{d=1}^{D}(I(e,\cdot,\cdot,d) * W^z(\cdot,\cdot,d))(x,y) + b^z$$

a. Since the only non-trivial case is for when the example is the same for both input and output, we drop the dependency on e and assume it to be the same for I and C

$$\begin{aligned}\frac{\partial C(i,j,k)}{\partial I(x,y,z)} &= \frac{\partial \sum_{d=1}^{D}(I(\cdot,\cdot,d) * W^k(\cdot,\cdot,d))(i,j) + b^k}{\partial I(x,y,z)} \\ &= \sum_{d=1}^{D} \frac{\partial (I(\cdot,\cdot,d) * W^k(\cdot,\cdot,d))(i,j) + b^k}{\partial I(x,y,z)} \\ &= \frac{\partial (I(\cdot,\cdot,z) * W^k(\cdot,\cdot,z))(i,j) + b^k}{\partial I(x,y,z)} \\ &= \frac{\partial (I(\cdot,\cdot,z) * W^k(\cdot,\cdot,z))(i,j)}{\partial I(x,y,z)} \\ &= \sum_{a,b \in \{-1,0,1\}} \frac{\partial I(i-a,j-b,z)}{\partial I(x,y,z)} W^k(a,b,z)\end{aligned}$$

From here we see that only the points within one square from $I(x,y,z)$ (including the diagonals) in the $x-y$-plane and on depth of z will have non-zero derivative with respect to $I(x,y,z)$. Assume we have such a point, otherwise the expression will be zero, then let $a_{x,i}, b_{y,j}$ be the indices in $\{-1,0,1\}$ such that $(i - a_{x,i}, j - b_{y,j}) = (x,y)$. Then we have the final expression

$$\frac{\partial C(i,j,k)}{\partial I(x,y,z)} = W^k(a_{x,i}, b_{y,j}, z)$$

which in terms of the batch we can write as

$$\frac{\partial C(e_1,i,j,k)}{\partial I(e_2,x,y,z)} = \delta(e_1 = e_2) W^k(a_{x,i}, b_{y,j}, z)$$

b. Similarly here we drop the dependency since we can just stack the examples in a 4d tensor after working it out example by example.

$$\begin{aligned}\frac{\partial C(i,j,k)}{\partial W^l(x,y,z)} &= \frac{\partial \sum_{d=1}^{D}(I(\cdot,\cdot,d) * W^k(\cdot,\cdot,d))(i,j) + b^k}{\partial W^l(x,y,z)} \\ &= \sum_{d=1}^{D} \frac{\partial (I(\cdot,\cdot,d) * W^k(\cdot,\cdot,d))(i,j) + b^k}{\partial W^l(x,y,z)} \\ &= \frac{\partial (I(\cdot,\cdot,z) * W^k(\cdot,\cdot,z))(i,j) + b^k}{\partial W^l(x,y,z)} \\ &= \frac{\partial (I(\cdot,\cdot,z) * W^k(\cdot,\cdot,z))(i,j)}{\partial W^l(x,y,z)} \\ &= \sum_{a,b \in \{-1,0,1\}} I(i-a,j-b,z) \frac{\partial W^k(a,b,z)}{\partial W^l(x,y,z)} \\ &= \sum_{a,b \in \{-1,0,1\}} I(i-a,j-b,z) \delta(k = l, a = x, b = z)\end{aligned}$$

$$= I(i - x, j - y, z)\delta(l = k)$$

For the part 2 of the network building I have used a modular approach. Using OOP I let the layers be classes with specific methods that calculate the backpropagation step and feedforward step, together with helper methods for setting the input and output of each layer as needed in the backpropagation.

On top of this I built a NeuralNetwork class which takes as input a list of instances of the layers which then does the training, plotting and various other methods as needed to get the necessary plots, figures and hyperparameter optimisation of the models.

For each model I do 3 runs over the learning rate. The learning rate for each model is picked from the list $[lr_0 * 10^{-i}]$ where $i \in \{0, 1, 2\}$ and lr_0 is the biggest power of ten such that I can train the model without numerical issues (NaN's and infinities). This meant that for the models (2a, 2b, 2c, corresponding to the models in part 1) I had initial learning rates (lr_0) of 0.1, 0.001 and 0.0001. I run the models (2a, 2b, 2c, corresponding to the models in part 1) for 150, 300 and 500 epochs each due to make sure that they converged with respect to the initial learning rates for each model. For each finished epoch I record the validation error and add this to a list. When the run is finished I record the smallest error in this list and at the end of what epoch this occurred. After each finished run I compare the best error for this run with the optimal error for any run and set the optimal error to be the error of this run if it is smaller. In the end I return the learning rate and the epoch at which the lowest error occurred and set this to the be the learning rate and the epoch to stop of the model.

I wasn't able to get the CNN to work properly and had to leave it out.

(b) Implement and train the model in (P1:a)

Optimal hyperparameters

epochs 107

learning rate 0.001

Graph, confusion matrix and final errors

The title should be 'Error over iterations'.

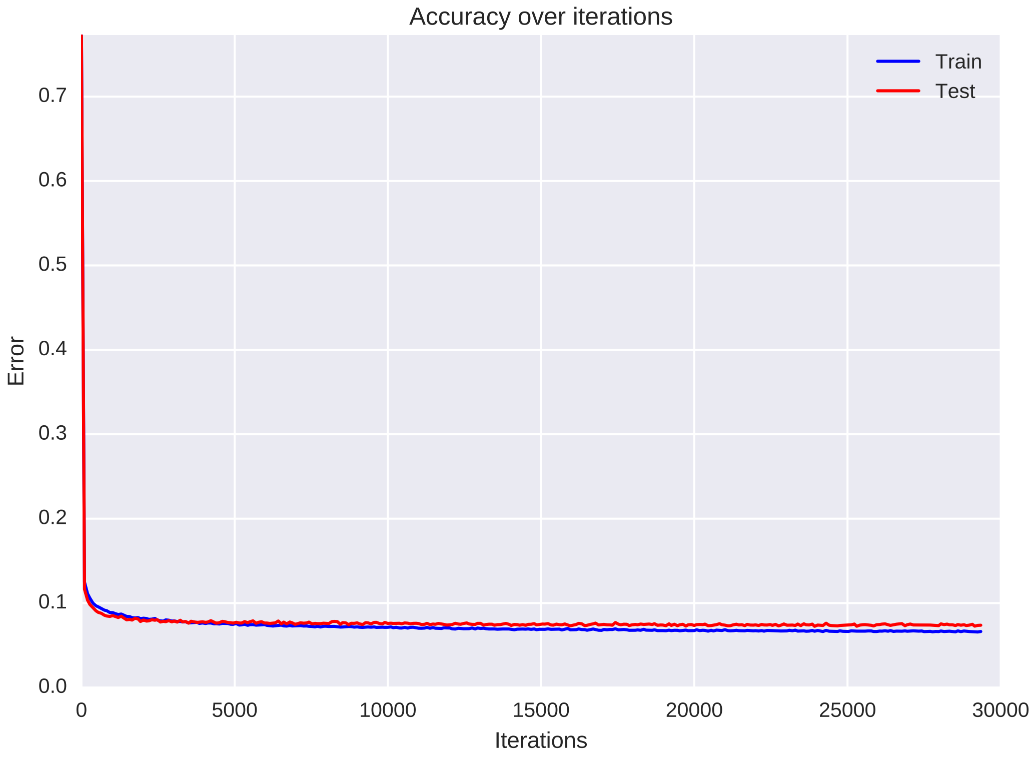
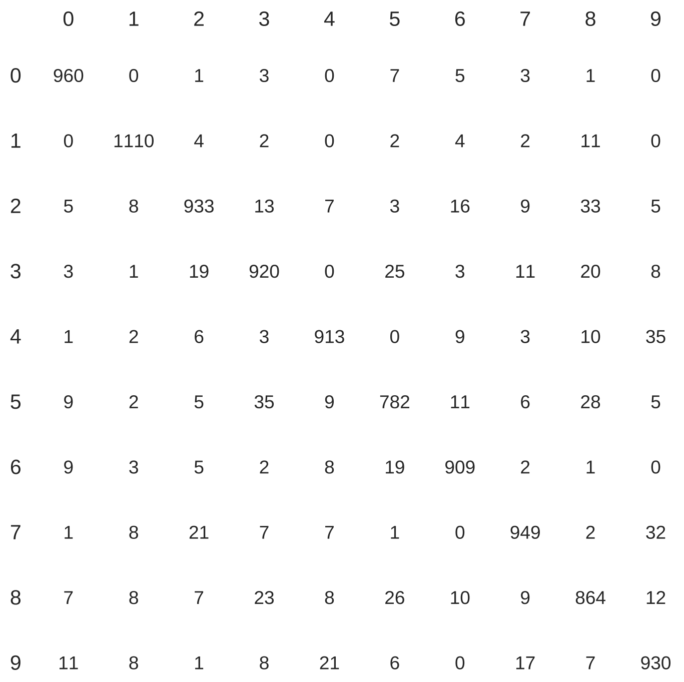


Figure 9: Error plot for model 2a



	0	1	2	3	4	5	6	7	8	9
0	960	0	1	3	0	7	5	3	1	0
1	0	1110	4	2	0	2	4	2	11	0
2	5	8	933	13	7	3	16	9	33	5
3	3	1	19	920	0	25	3	11	20	8
4	1	2	6	3	913	0	9	3	10	35
5	9	2	5	35	9	782	11	6	28	5
6	9	3	5	2	8	19	909	2	1	0
7	1	8	21	7	7	1	0	949	2	32
8	7	8	7	23	8	26	10	9	864	12
9	11	8	1	8	21	6	0	17	7	930

Figure 10: Confusion matrix for model 2a

Output of training and test error during training of the model. Due to the large number of epochs we only show output of every ten epoch and the final epoch.

```
Epoch: 1, Train error: 0.10449, test error: 0.09820
Epoch: 10, Train error: 0.08035, test error: 0.07770
Epoch: 20, Train error: 0.07385, test error: 0.07730
Epoch: 30, Train error: 0.07211, test error: 0.07740
Epoch: 40, Train error: 0.07040, test error: 0.07520
Epoch: 50, Train error: 0.06922, test error: 0.07620
Epoch: 60, Train error: 0.06765, test error: 0.07380
Epoch: 70, Train error: 0.06747, test error: 0.07400
Epoch: 80, Train error: 0.06758, test error: 0.07380
Epoch: 90, Train error: 0.06673, test error: 0.07390
Epoch: 100, Train error: 0.06642, test error: 0.07430
Epoch: 107, Train error: 0.06604, test error: 0.07300
```

Final train error: 0.06604

Final test error: 0.07300

(c) Implement and train the model in (P1:b)

Optimal hyperparameters

epochs 281

learning rate 0.0001

Graph, confusion matrix and final errors

The title should be 'Error over iterations'.

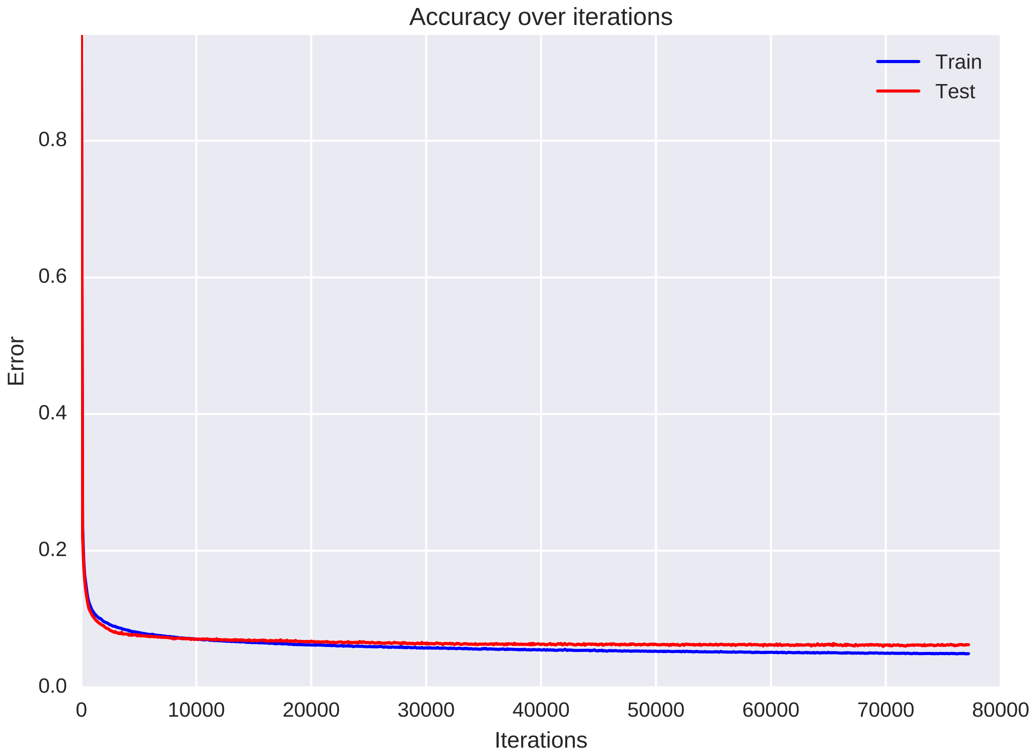
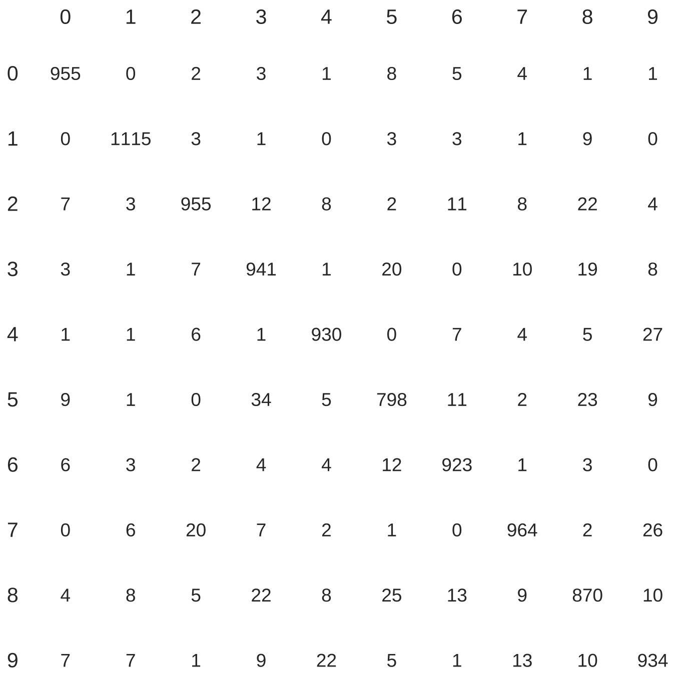


Figure 11: Error plot for model 2b



	0	1	2	3	4	5	6	7	8	9
0	955	0	2	3	1	8	5	4	1	1
1	0	1115	3	1	0	3	3	1	9	0
2	7	3	955	12	8	2	11	8	22	4
3	3	1	7	941	1	20	0	10	19	8
4	1	1	6	1	930	0	7	4	5	27
5	9	1	0	34	5	798	11	2	23	9
6	6	3	2	4	4	12	923	1	3	0
7	0	6	20	7	2	1	0	964	2	26
8	4	8	5	22	8	25	13	9	870	10
9	7	7	1	9	22	5	1	13	10	934

Figure 12: Confusion matrix for model 2b

Output of training and test error during training of the model. Due to the large number of epochs we only show output of every ten epoch and the final epoch.

```
Epoch: 1, Train error: 0.16507, test error: 0.15850
Epoch: 10, Train error: 0.08964, test error: 0.08240
Epoch: 20, Train error: 0.07851, test error: 0.07510
Epoch: 30, Train error: 0.07273, test error: 0.07190
Epoch: 40, Train error: 0.06993, test error: 0.07100
Epoch: 50, Train error: 0.06711, test error: 0.06900
Epoch: 60, Train error: 0.06475, test error: 0.06810
Epoch: 70, Train error: 0.06269, test error: 0.06730
Epoch: 80, Train error: 0.06167, test error: 0.06680
Epoch: 90, Train error: 0.05971, test error: 0.06510
Epoch: 100, Train error: 0.05869, test error: 0.06530
Epoch: 110, Train error: 0.05773, test error: 0.06430
Epoch: 120, Train error: 0.05675, test error: 0.06280
Epoch: 130, Train error: 0.05598, test error: 0.06320
Epoch: 140, Train error: 0.05525, test error: 0.06340
Epoch: 150, Train error: 0.05431, test error: 0.06260
Epoch: 160, Train error: 0.05411, test error: 0.06300
Epoch: 170, Train error: 0.05296, test error: 0.06280
Epoch: 180, Train error: 0.05264, test error: 0.06240
Epoch: 190, Train error: 0.05204, test error: 0.06280
```

```
Epoch: 200, Train error: 0.05200, test error: 0.06260
Epoch: 210, Train error: 0.05158, test error: 0.06250
Epoch: 220, Train error: 0.05131, test error: 0.06300
Epoch: 230, Train error: 0.05131, test error: 0.06360
Epoch: 240, Train error: 0.04993, test error: 0.06150
Epoch: 250, Train error: 0.05033, test error: 0.06260
Epoch: 260, Train error: 0.04982, test error: 0.06080
Epoch: 270, Train error: 0.04911, test error: 0.06150
Epoch: 280, Train error: 0.04911, test error: 0.06280
Epoch: 281, Train error: 0.04887, test error: 0.06150
```

Final train error: 0.04887

Final test error: 0.06150

(d) Implement and train the model in (P1:c)

Optimal hyperparameters

epochs 175

learning rate 0.0001

Graph, confusion matrix and final errors

The title should be 'Error over iterations'.

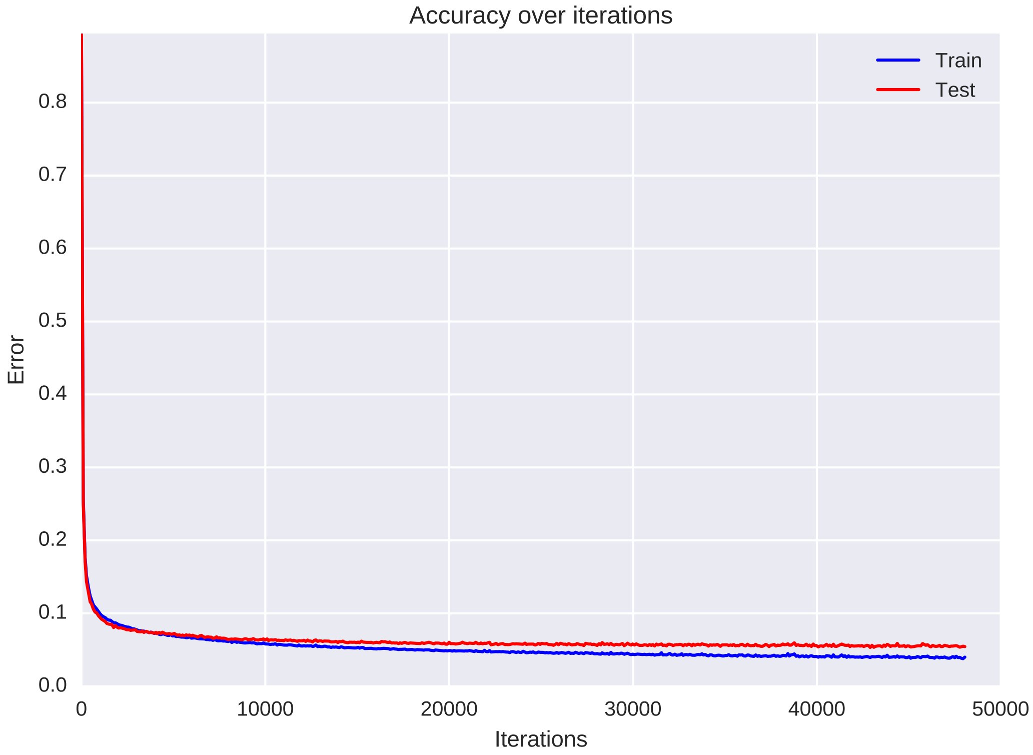
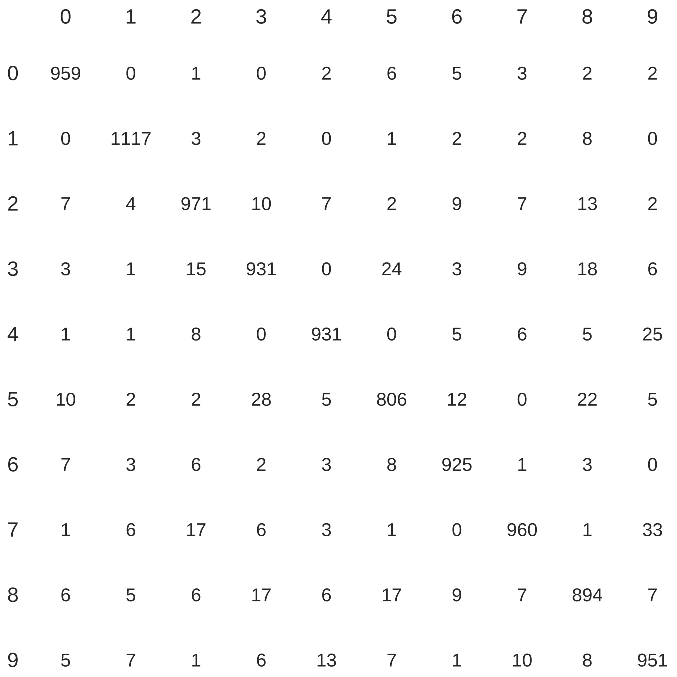


Figure 13: Error plot for model 2c



	0	1	2	3	4	5	6	7	8	9
0	959	0	1	0	2	6	5	3	2	2
1	0	1117	3	2	0	1	2	2	8	0
2	7	4	971	10	7	2	9	7	13	2
3	3	1	15	931	0	24	3	9	18	6
4	1	1	8	0	931	0	5	6	5	25
5	10	2	2	28	5	806	12	0	22	5
6	7	3	6	2	3	8	925	1	3	0
7	1	6	17	6	3	1	0	960	1	33
8	6	5	6	17	6	17	9	7	894	7
9	5	7	1	6	13	7	1	10	8	951

Figure 14: Confusion matrix for model 2c

Output of training and test error during training of the model. Due to the large number of epochs we only show output of every ten epoch and the final epoch.

```
Epoch: 1, Train error: 0.15253, test error: 0.14430
Epoch: 10, Train error: 0.07893, test error: 0.07760
Epoch: 20, Train error: 0.06747, test error: 0.07040
Epoch: 30, Train error: 0.06082, test error: 0.06410
Epoch: 40, Train error: 0.05733, test error: 0.06390
Epoch: 50, Train error: 0.05413, test error: 0.06090
Epoch: 60, Train error: 0.05089, test error: 0.06080
Epoch: 70, Train error: 0.05038, test error: 0.06020
Epoch: 80, Train error: 0.04713, test error: 0.05870
Epoch: 90, Train error: 0.04584, test error: 0.05840
Epoch: 100, Train error: 0.04529, test error: 0.05880
Epoch: 110, Train error: 0.04509, test error: 0.05650
Epoch: 120, Train error: 0.04296, test error: 0.05780
Epoch: 130, Train error: 0.04271, test error: 0.05640
Epoch: 140, Train error: 0.04458, test error: 0.05790
Epoch: 150, Train error: 0.03985, test error: 0.05610
Epoch: 160, Train error: 0.04415, test error: 0.05700
Epoch: 170, Train error: 0.03978, test error: 0.05510
Epoch: 175, Train error: 0.03933, test error: 0.05550
```

Final train error: 0.03933

Final test error: 0.05550

Error table

Experiment	P1:a	P1:b	P1:c	P1:d	P2:b	P2:c	P2:d	P2:e
Training error rate	0.06973	0.00169	0.00000	0.00707	0.06604	0.04887	0.03933	-
Test error rate	0.07620	0.02140	0.01890	0.01400	0.07300	0.06150	0.05550	-