Lab3 Report

Task 1(a)

Sum-of-squares error *vs.* cross-entropy error function

Sum-of-squares error:

Step1: do experiment with different values of parameters:

1) number of hidden layers,
2) number of hidden units in each layer,
3) learning rates,
4) momentum rates,
5) epochs

Step2:

Parameters					**Results**
Hidden layer	**Units**	**Epochs**	**Learning rates**	**Momentum rates**	**Loss & Accuracy (train data)**
2	100/50	500	0.01	0.90	Loss: 0.018 Acc: 0.9155
2	100/50	1000	0.01	0.90	Loss: 0.007 Acc: 0.9589
2	100/50	1000	0.02	0.95	Loss: 0.0025 Acc: 0.9882
2	70/35	1000	0.02	0.95	Loss: 0.0042

					Acc: 0.9772
3	100/50/25	500	0.02	0.95	Loss: 0.0028 Acc: 0.9856
3	100/50/25	1000	0.02	0.95	Loss: 0.0018 Acc: 0.9892

Step3: Best parameters config for •Sum-of-squares error function

Config	Results
Num of hidden layers = 3 Units each layer = 100/50/25 Learning rates = 0.02 Momentum rates = 0.95 Epochs = 1000	Execution time: 43s Acc(train): .0.9892 Loss(train): 0.0018 Acc(test): 0.9482 Loss(test): 0.0084

Step4: Confusion matrix for train data

```
[[373   0   0   0   1   0   1   0   1   0]
 [  0 384   0   0   0   3   0   0   0   2]
 [  0   0 377   2   0   0   1   0   0   0]
 [  0   1   0 383   0   5   0   0   0   0]
 [  1   0   0   0 384   0   2   0   0   0]
 [  0   0   0   0   0 376   0   0   0   0]
 [  0   2   0   0   1   0 374   0   0   0]
 [  0   0   0   1   0   1   0 384   0   1]
 [  0   3   2   0   0   2   0   0 372   1]
 [  1   1   0   3   1   0   0   0   1 375]]
```

Step5: Class accuracies

	precision	recall	f1-score	support
0	0.99	0.99	0.99	376
1	0.98	0.99	0.98	389
2	0.99	0.99	0.99	380
3	0.98	0.98	0.98	389
4	0.99	0.99	0.99	387

5	0.97	1.00	0.99	376
6	0.99	0.99	0.99	377
7	1.00	0.99	1.00	387
8	0.99	0.98	0.99	380
9	0.99	0.98	0.99	382

Step6: Confusion matrix for test data

```
[374   0   0   0   1   0   0   0   1   0]
 [  0 385   0   0   0   0   1   0   0   3]
 [  0   1 377   1   0   0   1   0   0   0]
 [  0   0   0 383   0   5   0   0   0   1]
 [  0   0   0   0 385   0   2   0   0   0]
 [  0   0   0   0   0 376   0   0   0   0]
 [  0   1   0   0   0   0 374   0   2   0]
 [  0   0   0   1   0   0   0 385   0   1]
 [  0   1   0   0   0   0   0   0 379   0]
 [  0   0   0   0   1   1   0   0   1 379]
```

Step7: Class accuracies

	precision	recall	f1-score	support
0	1.00	0.99	1.00	376
1	0.99	0.99	0.99	389
2	1.00	0.99	1.00	380
3	0.99	0.98	0.99	389
4	0.99	0.99	0.99	387
5	0.98	1.00	0.99	376
6	0.99	0.99	0.99	377
7	1.00	0.99	1.00	387
8	0.99	1.00	0.99	380
9	0.99	0.99	0.99	382

cross-entropy error:

Step1: do experiment with different values of parameters:

1) number of hidden layers,

2) number of hidden units in each layer,

3) learning rates,

4) momentum rates,

5) epochs

Step2:

Parameters					**Results**
Hidden layer	**Units**	**Epochs**	**Learning rates**	**Momentum rates**	**Loss & Accuracy (train data)**
2	100/50	50	0.01	0.90	Loss: 0.1312 Acc: 0.9623
2	100/50	100	0.01	0.90	Loss: 0.0864 Acc: 0.9746
2	100/50	100	0.02	0.95	Loss: 0.0536 Acc: 0.9847
2	70/35	100	0.02	0.95	Loss: 0.0443 Acc: 0.9892
3	100/50/25	50	0.02	0.95	Loss: 0.0723 Acc: 0.9772
3	100/50/25	100	0.02	0.95	Loss: 0.0493 Acc: 0.9840

Step3: Best parameters config for •Sum-of-squares error function

Config	Results
Num of hidden layers = 3	Execution time: 5s
Units each layer = 100/50/25	Acc(train): .0.9840
Learning rates = 0.02	Loss(train): 0.0493
Momentum rates = 0.95	Acc(test): 0.9321
Epochs = 100	Loss(test): 0.2391

Step4: Confusion matrix for train data

```
[374   0   0   0   1   0   1   0   0   0]
 [  0 384   0   0   0   0   0   1   1   3]
 [  0   1 377   0   0   0   1   1   0   0]
 [  0   1   2 379   0   1   0   5   1   0]
 [  0   0   0   0 385   0   1   0   0   1]
 [  1   0   1   3   1 368   0   0   0   2]
 [  0   0   0   0   0   0 377   0   0   0]
 [  0   0   0   0   0   0   0 387   0   0]
 [  0   3   1   0   0   0   0   0 376   0]
 [  0   1   0   8   1   0   0  15   2 355]
```

Step5: Class accuracies

	precision	recall	f1-score	support
0	1.00	0.99	1.00	376
1	0.98	0.99	0.99	389
2	0.99	0.99	0.99	380
3	0.97	0.97	0.97	389
4	0.99	0.99	0.99	387
5	1.00	0.98	0.99	376
6	0.99	1.00	1.00	377
7	0.95	1.00	0.97	387
8	0.99	0.99	0.99	380
9	0.98	0.93	0.96	382

Step6: Confusion matrix for test data

```
[374   0   0   0   1   0   1   0   0   0]
 [  0 372   0   0  12   0   4   1   0   0]
```

```
[  1   1 374   0   0   0   3   1   0   0]
[  0   1   0 380   0   7   0   1   0   0]
[  0   0   0   0 386   0   1   0   0   0]
[  0   0   0   0   0 376   0   0   0   0]
[  0   0   0   0  13   0 364   0   0   0]
[  0   0   0   1   1   0   0 385   0   0]
[ 10   5   1   2   5   2   9   0 346   0]
[  2   1   0   7   5   7   0   3   1 356]
```

Step7: Class accuracies

	precision	recall	f1-score	support
0	0.97	0.99	0.98	376
1	0.98	0.96	0.97	389
2	1.00	0.98	0.99	380
3	0.97	0.98	0.98	389
4	0.91	1.00	0.95	387
5	0.96	1.00	0.98	376
6	0.95	0.97	0.96	377
7	0.98	0.99	0.99	387
8	1.00	0.91	0.95	380
9	1.00	0.93	0.96	382

Compare results:

The performance of the cross-entropy loss function is much better than the mse loss function under the same number of epochs. For the similar performance, the cross-entropy loss function needs less epochs. The cross-entropy loss function is suitable for the multiple-class classification.

Task1 (b)

tanh vs relu hidden units

tanh:

Step1: do experiment with different values of parameters:

1) number of hidden layers,
2) number of hidden units in each layer,
3) learning rates,
4) momentum rates,
5) epochs

Step2:

Parameters					**Results**
Hidden layer	**Units**	**Epochs**	**Learning rates**	**Momentum rates**	**Loss & Accuracy (train data)**
2	100/50	20	0.01	0.90	Loss: 0.2546 Acc: 0.9357
2	100/50	50	0.01	0.90	Loss: 0.1200 Acc: 0.9689
2	100/50	50	0.02	0.95	Loss: 0.0832 Acc: 0.9741
2	70/35	50	0.02	0.95	Loss: 0.0785 Acc: 0.9778

3	100/50/25	20	0.02	0.95	Loss: 0.1132 Acc: 0.9623
3	100/50/25	50	0.02	0.95	Loss: 0.0492 Acc: 0.9838

Step3: Best parameters config for •Sum-of-squares error function

Config	Results
Num of hidden layers = 3 Units each layer = 100/50/25 Learning rates = 0.02 Momentum rates = 0.95 Epochs = 50	Execution time: 3s Acc(train): .0.9838 Loss(train): 0.0492 Acc(test): 0.9543 Loss(test): 0.1624

Step4: Confusion matrix for train data

```
[374   0   0   0   1   1   0   0   0   0]
[  0 379   0   0   0   0   0   0   8   2]
[  0   1 371   2   0   2   1   1   1   1]
[  0   1   0 380   0   4   0   0   0   4]
[  0   0   0   0 380   0   2   0   3   2]
[  0   0   1   0   0 374   0   0   0   1]
[  0   1   0   0   1   0 374   0   1   0]
[  0   0   0   2   0   1   0 377   1   6]
[  0   3   0   0   0   1   0   0 376   0]
[  0   0   0   1   2   1   0   0   2 376]
```

Step5: Class accuracies

	precision	recall	f1-score	support
0	1.00	0.99	1.00	376
1	0.98	0.97	0.98	389
2	1.00	0.98	0.99	380
3	0.99	0.98	0.98	389
4	0.99	0.98	0.99	387
5	0.97	0.99	0.98	376

6	0.99	0.99	0.99	377
7	1.00	0.97	0.99	387
8	0.96	0.99	0.97	380
9	0.96	0.98	0.97	382

Step6: Confusion matrix for test data

```
[376   0   0   0   0   0   0   0   0   0]
[  0 377   0   0   0   0   4   1   5   2]
[  0   0 379   0   0   0   1   0   0   0]
[  0   1   0 385   0   2   0   0   0   1]
[  1   0   0   0 383   0   3   0   0   0]
[  0   0   1   1   0 372   0   0   0   2]
[  1   0   0   0   0   0 376   0   0   0]
[  0   0   0   1   0   0   0 385   0   1]
[  0   1   0   0   1   0   1   0 377   0]
[  0   0   0   4   2   1   0   1   2 372]
```

Step7: Class accuracies

	precision	recall	f1-score	support
0	0.99	1.00	1.00	376
1	0.99	0.97	0.98	389
2	1.00	1.00	1.00	380
3	0.98	0.99	0.99	389
4	0.99	0.99	0.99	387
5	0.99	0.99	0.99	376
6	0.98	1.00	0.99	377
7	0.99	0.99	0.99	387
8	0.98	0.99	0.99	380
9	0.98	0.97	0.98	382

`relu:`

Step1: do experiment with different values of parameters:

1) number of hidden layers,

2) number of hidden units in each layer,

3) learning rates,

4) momentum rates,

5) epochs

Step2:

Parameters					**Results**
Hidden layer	**Units**	**Epochs**	**Learning rates**	**Momentum rates**	**Loss & Accuracy (train data)**
2	100/50	20	0.01	0.90	Loss: 0.3054 Acc: 0.9189
2	100/50	50	0.01	0.90	Loss: 0.1312 Acc: 0.9623
2	100/50	50	0.02	0.95	Loss: 0.0979 Acc: 0.9689
2	70/35	50	0.02	0.95	Loss: 0.1427 Acc: 0.9610
3	100/50/25	20	0.02	0.95	Loss: 0.1759 Acc: 0.9456
3	100/50/25	50	0.02	0.95	Loss: 0.1144 Acc: 0.9597

Step3: Best parameters config for •Sum-of-squares error function

Config	Results

Num of hidden layers = 3 Units each layer = 100/50/25 Learning rates = 0.02 Momentum rates = 0.95 Epochs = 50	Execution time: 3s Acc(train): .0.9597 Loss(train): 0.1144 Acc(test): 0.9232 Loss(test): 0.3124

Step4: Confusion matrix for train data

```
[375   0   0   0   0   0   0   0   1   0]
[  0 304   4   1   1   0  16   3  55   5]
[  1   0 378   0   0   0   1   0   0   0]
[  0   0   1 377   0   5   0   1   2   3]
[  0   0   0   0 365   0  11   0   5   6]
[  0   0   2   1   0 371   0   0   0   2]
[  0   0   0   0   0   0 376   0   1   0]
[  0   0   0   1   0   0   0 385   0   1]
[  1   1   1   0   0   0   0   0 377   0]
[  0   0   1   5   1   1   0   7   6 361]
```

Step5: Class accuracies

	precision	recall	f1-score	support
0	0.99	1.00	1.00	376
1	1.00	0.78	0.88	389
2	0.98	0.99	0.99	380
3	0.98	0.97	0.97	389
4	0.99	0.94	0.97	387
5	0.98	0.99	0.99	376
6	0.93	1.00	0.96	377
7	0.97	0.99	0.98	387
8	0.84	0.99	0.91	380
9	0.96	0.95	0.95	382

Step6: Confusion matrix for test data

```
[364   0   2   0   1   4   0   0   1   4]
[  0 368   0   3   1   0   0   0  11   6]
[  1   3 363   9   0   0   0   0   4   0]
[  0   1   0 379   0   4   0   0   0   5]
[  1   2   0   0 372   0   3   0   1   8]
```

```
[  0   0   0   1   0 370   0   0   0   5]
[  3   5   1   0   1   0 363   0   4   0]
[  0   0   0   5   0   3   0 364   0  15]
[  0   1   2   1   0   1   0   0 369   6]
[  0   0   0   7   1   1   0   0   0 373]
```

Step7: Class accuracies

	precision	recall	f1-score	support
0	0.99	0.97	0.98	376
1	0.97	0.95	0.96	389
2	0.99	0.96	0.97	380
3	0.94	0.97	0.95	389
4	0.99	0.96	0.98	387
5	0.97	0.98	0.97	376
6	0.99	0.96	0.98	377
7	1.00	0.94	0.97	387
8	0.95	0.97	0.96	380
9	0.88	0.98	0.93	382

Compare the results:

The performance of tanh is slightly better than relu for the same setting. The runtime is similar for the two activations.

Task2:

Convolutional networks

Step1: do experiment with different values of parameters:

1) number of convolutional layers,
2) number of filters,
3) kernel sizes,
4) strides,

5) epochs

Step2:

Parameters					**Results**
Convolutional layer	**filters**	**Epochs**	**Kernel sizes**	**strides**	**Loss & Accuracy (train data)**
1	32	20	3	1	Loss: 0.9020 Acc: 0.8837
1	32	50	3	1	Loss: 0.2764 Acc: 0.9406
1	16	50	3	1	Loss: 0.3792 Acc: 0.9233
1	32	50	5	1	Loss: 0.2638 Acc: 0.9395
1	32	50	3	2	Loss: 0.8516 Acc: 0.8093
2	32/64	20	3	1	Loss: 0.5201 Acc: 0.8843
2	32/64	50	3	1	Loss: 0.1984 Acc: 0.9476
2	32/64	50	5	1	Loss: 0.1553 Acc: 0.9626

2	16/32	50	3	1	Loss: 0.3253 Acc: 0.9147
2	32/64	50	7	1	Loss: 0.1281 Acc: 0.9665

Step3: Best parameters config for •Sum-of-squares error function

Config	Results
Num of Convolutional layers= 2 filters= 32/64 kernel size = 7 strides = 1 Epochs = 50	Execution time: 62s Acc(train): .0.9665 Loss(train): 0.1281 Acc(test): 0.9388 Loss(test): 0.2011

Step4: Confusion matrix for train data

```
[371   0   0   0   3   0   1   0   1   0]
[  0 375   1   0   0   0   1   2   6   4]
[  0   3 373   0   0   1   1   2   0   0]
[  0   2   1 376   0   5   0   1   2   2]
[  0   0   0   0 373   0   9   0   0   5]
[  0   0   2   1   1 361   0   0   0  11]
[  0   2   0   0   1   0 374   0   0   0]
[  1   2   0   1   1   0   0 379   1   2]
[  0  21   1   0   2   0   0   2 354   0]
[  0   3   0   8   6   1   0   4   1 359]
```

Step5: Class accuracies

	precision	recall	f1-score	support
0	1.00	0.99	0.99	376
1	0.92	0.96	0.94	389
2	0.99	0.98	0.98	380
3	0.97	0.97	0.97	389
4	0.96	0.96	0.96	387
5	0.98	0.96	0.97	376

6	0.97	0.99	0.98	377
7	0.97	0.98	0.98	387
8	0.97	0.93	0.95	380
9	0.94	0.94	0.94	382

Step6: Confusion matrix for test data

```
[371   0   0   0   4   0   1   0   0   0]
[  0 373   2   0   0   0   0   2   8   4]
[  0   3 370   2   0   1   1   2   1   0]
[  0   0   2 378   0   5   0   0   2   2]
[  0   0   0   0 372   0   9   0   1   5]
[  0   0   1   2   0 367   0   0   1   5]
[  0   2   0   0   1   0 374   0   0   0]
[  0   1   0   1   1   0   0 379   1   4]
[  0  18   4   3   1   2   0   1 350   1]
[  0   2   0  10   3   1   0   5   2 359]
```

Step7: Class accuracies

	precision	recall	f1-score	support
0	1.00	0.99	0.99	376
1	0.93	0.96	0.95	389
2	0.98	0.97	0.97	380
3	0.95	0.97	0.96	389
4	0.97	0.96	0.97	387
5	0.98	0.98	0.98	376
6	0.97	0.99	0.98	377
7	0.97	0.98	0.98	387
8	0.96	0.92	0.94	380
9	0.94	0.94	0.94	382

Compare the results:

Filters: the more filters in a certain range, the better th e performance

Kernel size: the larger the kernel size in a certain range, the better the performance

Convolutional layers: the more convolutional layers in a ce rtain range, the better the performance.

Stride: the smaller the stride, the better the performance.

Epochs: the more the epochs, the better the performance.