1. Clearly describe your training and test data size and sampling methods, and how you trained the models. Report the best hyper-parameters leading to the best classification accuracy and explain how and why. Also show the best accuracy. For each sets of tested hyper-parameters, show the accuracy and the time for training.

For SVM:
Training data size is 60000, and the test data size is 10000.
The best accuracy for this test data is 0.9787.
The hyper-parameters for this accuracy is: kernel = "poly".

accuracy	Train time	Gamma	C	kernel
0.9787	300s			poly
0.9724	473s	0.000686	25.6	rbf
0.9141	2990s	0.007	3	rbf
0.9427	607s	0.00001	30	rbf
0.9718	408s	0.0005	50	rbf
0.9711	385s	0.0004	60	rbf
0.961278	636s	'scale'	1	rbf
0.9536	952	'scale'	0.5	rbf
0.91783	508s	'scale'	2	linear
0.9226	598	'scale'	0.5	linear
0.7927	11s			LinearSVC

To get the best classification accuracy by using kernel "rbf", is because of the data don't have the linear relationship, so if use linear method to classify it is not the best way. And a larger coefficient indicates that the model is more tolerant of classification errors, and the result may not accuracy. And use the coefficient in default can get the best accuracy. SVM use much time than ANN.

For ANN:
Training data size is 60000, and the test data size is 10000.
The best accuracy for test data is 0.947.
The hyper-parameters for this accuracy is:
solver='Adam', activation='relu', alpha=0.0001, hidden_layer_sizes=(130,100,100,10)

accuracy	Train time	solver	activation	alpha	hidden_layer_sizes	learning_rate_init
0.947	139s	Adam	relu	0.0001	(130,100,100,10)	
0.9352	195s	Adam	relu	0.0001	(130,130,100,100,10)	
0.1028	170s	sgd	relu	0.0001	(130,130,100,100,10)	0.1
0.8516	87.58s	sgd	logistic	0.0001	(100,100,10)	0.1
0.1135	80.8s	sgd	relu	0.0001	(100,100,10)	0.1
0.9325	132s	Adam	relu	0.0001	(130,130,100,10)	
0.9304	93s	Adam	relu	0.0001	(100,100,10)	
0.8336	146s	Adam	logistic	0.005	(20,20)	

To get the best classification accuracy by using optimizer "Adam", and use five hidden layer which has 130,100,100,10. Adam makes the learning rate of each iteration has a certain range, so that the parameters are relatively stable. 4 layers can learn more parameter to make the model better. But to much layer, the time will increase more and the accuracy may not better.

2. For the best trained models of SVM and ANN, design experiments to investigate how the training data size affects the accuracy of both models. Which models are more sensitive to the training data size? Clearly describe your experimental design, accuracy change/comparison, and also training time change with increased training set size.

In my experiment, I will increase the training set in each epoch, and in some epoch, use the same training data size but different number of testing data size to check if the accuracy will be change. I also record the train loss in my ANN models in each epoch and check whether the loss keep decreasing.

For SVM:

Train Size	Test Size	Accuracy	Train Time
500	10000	0.8253	3.39s
1000	10000	0.8865	4.0234s
3000	10000	0.931	6.7479s
5000	2000	0.921	10.3277s
5000	10000	0.9421	10.3277s
10000	4000	0.9445	23.292s
36000	5000	0.9684	143s
48000	8000	0.97575	228s
48000	10000	0.9778	228s
60000	10000	0.9787	300s

For ANN:

Train Size	Test Size	Train Loss	Accuracy	Train Time
500	200	0.0009456	0.595	7.07s
1000	500	0.00080	0.54	7.82s
3000	200	0.0004778	0.67	12.139s
3000	1000	0.000477	0.671	12.70s
5000	2000	0.000423	0.7095	13.386s
5000	3000	0.0004232	0.713	14.177
10000	4000	0.00028091	0.82275	20.26285
36000	5000	0.00029821	0.932	82.93180
48000	8000	0.009842	0.9092	70.57469
48000	10000	0.009842	0.9172	69.411073
60000	10000	0.0068	0.947	136.2963

The result is: ANN are more sensitive to the training data size.

3. Design experiments to show how the distribution of training data sizes affect the accuracy, recall, and precision of both models. For example, if you make the training data highly imbalanced, how would this affect the recall and precision of each class,

particularly the class with small training sets? For this problem, you need to report both the accuracy and also the recall/precision for each class. Clearly describe your experimental design and results.
In my experiment, I will arrange 10 times to test first, for different number I will arrange only 20 instance as train data size, and at the same time other numbers will arrange 5000 instance as train data size. For each epoch, I will record the accuracy, time, precision and recall data.
In ANN model, After 10 times test, I will increase the number of less training data size from 20 to 100, 500, 1000, 2000, 3000, and check the result.
For SVM:

Train size of amount for 20	Train size of amount for 5000	Test Size	Accuracy	Train Time
0	1-9	10000	0.9292	189s
1	0, 2-9	10000	0.9529	201s
2	0-1,3-9	10000	0.9016	174s
3	0-2,4-9	10000	0.9179	168s
4	0-3,5-9	10000	0.9196	193s
5	0-4,6-9	10000	0.9114	183s
6	0-5,7-9	10000	0.927	188s
7	0-6,8-9	10000	0.9152	180s
8	0-7,9	10000	0.916	165s
9	0-8	10000	0.9064	177s

For 0:

```
conf_mat:
[[ 469    0   95    1    4  143  198    3   15   52]
 [    0 1126    2    1    1    0    3    0    2    0]
 [    0    1 1012    0    2    2    5    7    3    0]
 [    0    2    1  986    0    7    0    5    6    3]
 [    0    0    3    0  967    0    3    0    0    9]
 [    0    0    1   11    1  866    6    0    4    3]
 [    0    5    1    0    3    8  939    0    2    0]
 [    0   12    8    1    0    0    0 1001    0    6]
 [    0    1    1    8    3    5    1    2  950    3]
 [    0    6    3    4    8    5    1    4    2  976]]
              precision    recall  f1-score   support

           0       1.00      0.48      0.65       980
           1       0.98      0.99      0.98      1135
           2       0.90      0.98      0.94      1032
           3       0.97      0.98      0.98      1010
           4       0.98      0.98      0.98       982
           5       0.84      0.97      0.90       892
           6       0.81      0.98      0.89       958
           7       0.98      0.97      0.98      1028
           8       0.97      0.98      0.97       974
           9       0.93      0.97      0.95      1009

   micro avg       0.93      0.93      0.93     10000
   macro avg       0.93      0.93      0.92     10000
weighted avg       0.94      0.93      0.92     10000
```

For 1:

```
conf_mat:
[[ 972    0    1    1    0    3    1    0    2    0]
 [    0  861   23   10   12   22   11  127   69    0]
 [    7    0 1008    0    2    0    5    7    3    0]
 [    0    0    1  986    0    9    0    5    6    3]
 [    3    0    2    0  965    0    3    0    0    9]
 [    3    0    1   10    1  864    5    1    4    3]
 [    4    0    1    0    3    9  938    0    3    0]
 [    0    0    9    2    2    0    0 1009    0    6]
 [    7    0    1    7    3    4    1    2  946    3]
 [    1    0    3    4    8    4    1    6    2  980]]
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       1.00      0.76      0.86      1135
           2       0.96      0.98      0.97      1032
           3       0.97      0.98      0.97      1010
           4       0.97      0.98      0.98       982
           5       0.94      0.97      0.96       892
           6       0.97      0.98      0.98       958
           7       0.87      0.98      0.92      1028
           8       0.91      0.97      0.94       974
           9       0.98      0.97      0.97      1009

   micro avg       0.95      0.95      0.95     10000
   macro avg       0.95      0.96      0.95     10000
weighted avg       0.96      0.95      0.95     10000
```

For 2:

For 3:

```
conf_mat:
[[ 972    0    0    1    0    3    2    0    2    0]
 [   0 1126    0    3    1    0    3    0    2    0]
 [  68   76  230  288   61    3   65   82  152    7]
 [   0    2    0  987    0    7    0    5    6    3]
 [   2    0    0    0  967    0    4    0    0    9]
 [   3    0    0   10    1  865    5    1    4    3]
 [   4    5    1    0    3    6  937    0    2    0]
 [   0   13    1    1    0    0    0 1008    0    5]
 [   7    1    0    7    3    4    1    2  946    3]
 [   1    6    0    4    8    5    1    4    2  978]]
              precision    recall  f1-score   support

           0       0.92      0.99      0.95       980
           1       0.92      0.99      0.95      1135
           2       0.99      0.22      0.36      1032
           3       0.76      0.98      0.85      1010
           4       0.93      0.98      0.95       982
           5       0.97      0.97      0.97       892
           6       0.92      0.98      0.95       958
           7       0.91      0.98      0.95      1028
           8       0.85      0.97      0.91       974
           9       0.97      0.97      0.97      1009

   micro avg       0.90      0.90      0.90     10000
   macro avg       0.91      0.90      0.88     10000
weighted avg       0.91      0.90      0.88     10000
```

For 4:

```
conf_mat:
[[ 972    0    1    1    0    3    1    0    2    0]
 [   0 1126    2    1    0    0    3    1    2    0]
 [   7    1 1008    0    0    0    6    7    3    0]
 [   0    2    1  986    0    7    0    5    6    3]
 [  12   11   22    2  367    3   32   41   14  478]
 [   3    0    1   10    0  864    5    2    4    3]
 [   4    6    1    0    0    6  938    0    2    1]
 [   0   11    8    1    0    0    0 1003    0    5]
 [   7    1    1    7    1    4    1    3  946    3]
 [   1    6    3    4    0    4    2    4    2  983]]
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.97      0.99      0.98      1135
           2       0.96      0.98      0.97      1032
           3       0.97      0.98      0.98      1010
           4       1.00      0.37      0.54       982
           5       0.97      0.97      0.97       892
           6       0.95      0.98      0.96       958
           7       0.94      0.98      0.96      1028
           8       0.96      0.97      0.97       974
           9       0.67      0.97      0.79      1009

   micro avg       0.92      0.92      0.92     10000
   macro avg       0.94      0.92      0.91     10000
weighted avg       0.94      0.92      0.91     10000
```

For 6:

```
conf_mat:
[[ 972    0    1    1    1    3    0    0    2    0]
 [   0 1127    2    1    1    2    0    0    2    0]
 [   8    1 1011    0    2    0    0    7    3    0]
 [   0    2    1  986    0    7    0    5    6    3]
 [   4    0    3    0  966    0    0    0    0    9]
 [   3    0    2   10    1  868    0    1    4    3]
 [  98   12   92    1  182  115  417    0   38    3]
 [   0   12    8    1    0    0    0 1001    0    6]
 [   7    1    1    7    3    4    0    2  946    3]
 [   2    6    3    4    8    4    0    4    2  976]]
              precision    recall  f1-score   support

           0       0.89      0.99      0.94       980
           1       0.97      0.99      0.98      1135
           2       0.90      0.98      0.94      1032
           3       0.98      0.98      0.98      1010
           4       0.83      0.98      0.90       982
           5       0.87      0.97      0.92       892
           6       1.00      0.44      0.61       958
           7       0.98      0.97      0.98      1028
           8       0.94      0.97      0.96       974
           9       0.97      0.97      0.97      1009

   micro avg       0.93      0.93      0.93     10000
   macro avg       0.93      0.92      0.92     10000
weighted avg       0.93      0.93      0.92     10000
```

For 8:

```
conf_mat:
[[ 972    0    1    0    0    2    2    1    2    0]
 [   0 1126    3    0    1    0    3    0    2    0]
 [   7    1 1007    0    2    0    5    7    3    0]
 [   4    7  105  373    0  238    0   26  218   39]
 [   3    0    2    0  965    0    3    0    0    9]
 [   3    0    1    0    1  872    5    1    6    3]
 [   4    5    1    0    3    6  937    0    2    0]
 [   0   12    9    0    0    0    0 1001    0    6]
 [   7    1    1    0    3    6    1    3  949    3]
 [   1    6    3    0    8    5    1    5    3  977]]
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.97      0.99      0.98      1135
           2       0.89      0.98      0.93      1032
           3       1.00      0.37      0.54      1010
           4       0.98      0.98      0.98       982
           5       0.77      0.98      0.86       892
           6       0.98      0.98      0.98       958
           7       0.96      0.97      0.97      1028
           8       0.80      0.97      0.88       974
           9       0.94      0.97      0.96      1009

   micro avg       0.92      0.92      0.92     10000
   macro avg       0.93      0.92      0.91     10000
weighted avg       0.93      0.92      0.91     10000
```

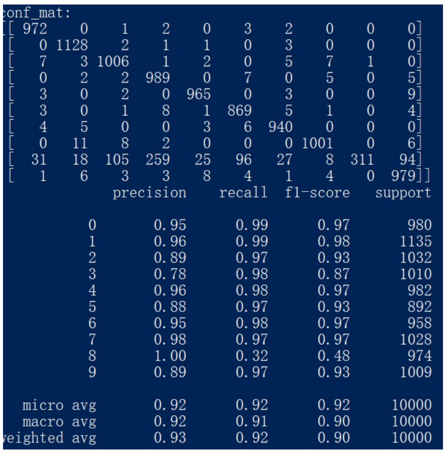
For 5:

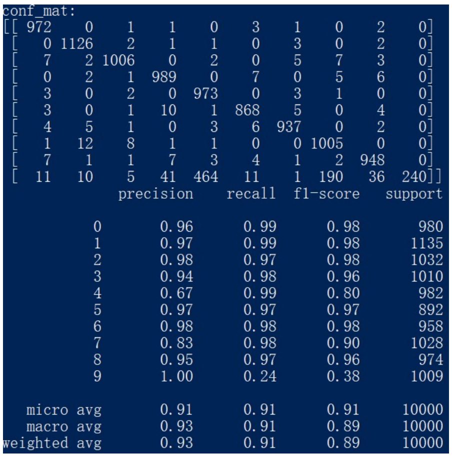
```
conf_mat:
[[ 973    0    1    2    0    0    1    0    3    0]
 [   0 1125    2    1    1    0    4    0    2    0]
 [   6    1 1009    0    1    0    5    7    3    0]
 [   0    2    1  992    0    0    0    5    7    3]
 [   3    0    2    0  965    0    3    0    0    9]
 [  26   17    5  344   13  181   38    4  214   50]
 [   4    5    3    0    2    0  942    0    2    0]
 [   0   12    8    1    0    0    0 1001    0    6]
 [   7    1    1    6    4    0    1    2  949    3]
 [   2    7    2    5    8    0    1    4    3  977]]
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       980
           1       0.96      0.99      0.98      1135
           2       0.98      0.98      0.98      1032
           3       0.73      0.98      0.84      1010
           4       0.97      0.98      0.98       982
           5       1.00      0.20      0.34       892
           6       0.95      0.98      0.96       958
           7       0.98      0.97      0.98      1028
           8       0.80      0.97      0.88       974
           9       0.93      0.97      0.95      1009

   micro avg       0.91      0.91      0.91     10000
   macro avg       0.93      0.90      0.89     10000
weighted avg       0.93      0.91      0.89     10000
```

For 7:

```
conf_mat:
[[ 972    0    1    1    0    2    1    0    2    1]
 [   0 1125    3    1    1    0    3    0    2    0]
 [   7    2 1009    0    3    0    5    1    4    1]
 [   0    2    1  989    0    7    0    0    6    5]
 [   3    0    2    0  964    0    3    0    0   10]
 [   3    0    1   10    1  865    5    0    4    3]
 [   4    5    1    0    3    6  937    0    2    0]
 [   4   41  137   85   27    5    0  365    8  356]
 [   7    1    1    7    3    4    1    1  946    3]
 [   1    6    3    4    8    4    1    0    2  980]]
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.95      0.99      0.97      1135
           2       0.87      0.98      0.92      1032
           3       0.90      0.98      0.94      1010
           4       0.95      0.98      0.97       982
           5       0.97      0.97      0.97       892
           6       0.98      0.98      0.98       958
           7       0.99      0.36      0.52      1028
           8       0.97      0.97      0.97       974
           9       0.72      0.97      0.83      1009

   micro avg       0.92      0.92      0.92     10000
   macro avg       0.93      0.92      0.90     10000
weighted avg       0.93      0.92      0.90     10000
```

For 9:


```
conf_mat:
[[ 972    0    1    2    0    3    2    0    0    0]
 [   0 1128    2    1    1    0    3    0    0    0]
 [   7    3 1006    1    2    0    5    7    1    0]
 [   0    2    2  989    0    7    0    5    0    5]
 [   3    0    2    0  965    0    3    0    0    9]
 [   3    0    1    8    1  869    5    1    0    4]
 [   4    5    0    0    3    6  940    0    0    0]
 [   0   11    8    2    0    0    0 1001    0    6]
 [  31   18  105  259   25   96   27    8  311   94]
 [   1    6    3    3    8    4    1    4    0  979]]
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       980
           1       0.96      0.99      0.98      1135
           2       0.89      0.97      0.93      1032
           3       0.78      0.98      0.87      1010
           4       0.96      0.98      0.97       982
           5       0.88      0.97      0.93       892
           6       0.95      0.98      0.97       958
           7       0.98      0.97      0.97      1028
           8       1.00      0.32      0.48       974
           9       0.89      0.97      0.93      1009

   micro avg       0.92      0.92      0.92     10000
   macro avg       0.92      0.91      0.90     10000
weighted avg       0.93      0.92      0.90     10000
```


```
conf_mat:
[[ 972    0    1    1    0    3    1    0    2    0]
 [   0 1126    2    1    1    0    3    0    2    0]
 [   7    2 1006    0    2    0    5    7    3    0]
 [   0    2    1  989    0    7    0    5    6    0]
 [   3    0    2    0  973    0    3    1    0    0]
 [   3    0    1   10    1  868    5    0    4    0]
 [   4    5    1    0    3    6  937    0    2    0]
 [   1   12    8    1    1    0    0 1005    0    0]
 [   7    1    1    7    3    4    1    2  948    0]
 [  11   10    5   41  464   11    1  190   36  240]]
              precision    recall  f1-score   support

           0       0.96      0.99      0.98       980
           1       0.97      0.99      0.98      1135
           2       0.98      0.97      0.98      1032
           3       0.94      0.98      0.96      1010
           4       0.67      0.99      0.80       982
           5       0.97      0.97      0.97       892
           6       0.98      0.98      0.98       958
           7       0.83      0.98      0.90      1028
           8       0.95      0.97      0.96       974
           9       1.00      0.24      0.38      1009

   micro avg       0.91      0.91      0.91     10000
   macro avg       0.93      0.91      0.89     10000
weighted avg       0.93      0.91      0.89     10000
```

For ANN:

Train size of amount for 20	Train size of amount for 5000	Test Size	Accuracy	Train Time
0	1-9	10000	0.847	33.065
1	0, 2-9	10000	0.8044	40.224
2	0-1,3-9	10000	0.8344	34.74
3	0-2,4-9	10000	0.8625	30.956
4	0-3,5-9	10000	0.8137	40.306
5	0-4,6-9	10000	0.8546	31.180
6	0-5,7-9	10000	0.849	37.856
7	0-6,8-9	10000	0.8468	34.474
8	0-7,9	10000	0.8758	54.465
9	0-8	10000	0.8555	37.9898

The training data amount of 0	Train size of amount for 5000	Test Size	Accuracy	Time
20	1-9	10000	0.847	33.065
100	1-9	10000	0.8749	46.294
500	1-9	10000	0.8776	40.692
1000	1-9	10000	0.9237	41.27
2000	1-9	10000	0.9282	36.894
3000	1-9	10000	0.8973	53.714
5000	1-9	10000	0.9302	47.582

20 instance for less training size:

For 0:

```
conf_mat:
[[  87    1  178   22    5  152  259   92   51  133]
 [   0 1116    4    1    0    1    1    1   11    0]
 [   0    3  935   21    8    0   11   35   15    4]
 [   0    0    5  982    0    6    0    4   12    1]
 [   0    7    2    1  836    0    3    1    9  123]
 [   0    2    2   39    5  746   20    2   69    7]
 [   0    2    3    2   11   13  896    0   29    2]
 [   0    8    6   10    1    0    1  977    6   19]
 [   0    2    3   15    1    1    3    2  947    0]
 [   0    9    0   28    2    3    2    7   10  948]]
```

	precision	recall	f1-score	support
0	1.00	0.09	0.16	980
1	0.97	0.98	0.98	1135
2	0.82	0.91	0.86	1032
3	0.88	0.97	0.92	1010
4	0.96	0.85	0.90	982
5	0.81	0.84	0.82	892
6	0.75	0.94	0.83	958
7	0.87	0.95	0.91	1028
8	0.82	0.97	0.89	974
9	0.77	0.94	0.84	1009
avg / total	0.87	0.85	0.82	10000

For 1:

```
conf_mat:
[[ 952    0   10    4    0    4    3    2    1    4]
 [   7  155   73   71    2    2    2  249  573    1]
 [   6    0  902   72    4    0    6   24   17    1]
 [   0    0    1 1000    0    4    0    1    3    1]
 [  15    0    3    6  665    0    6   31   24  232]
 [  34    0    2  123    0  680   14    0   32    7]
 [  32    0    2    8    7    3  895    1    5    5]
 [   1    0    9   36    0    0    0  960    7   15]
 [   2    0    0   63    0    0    3    2  901    3]
 [   5    0    0   44    0    1    1   10   14  934]]
```

	precision	recall	f1-score	support
0	0.90	0.97	0.94	980
1	1.00	0.14	0.24	1135
2	0.90	0.87	0.89	1032
3	0.70	0.99	0.82	1010
4	0.98	0.68	0.80	982
5	0.98	0.76	0.86	892
6	0.96	0.93	0.95	958
7	0.75	0.93	0.83	1028
8	0.57	0.93	0.71	974
9	0.78	0.93	0.84	1009
avg / total	0.85	0.80	0.78	10000

For 2:

```
conf_mat:
[[ 969    1    1    2    0    3    0    0    2    2]
 [   0 1066    0    7    0    1    1   20   40    0]
 [  41   32   69  413   26    0   63  213  174    1]
 [   0    0    1  996    0    5    0    0    7    1]
 [   8    0    0    1  839    0    9    5   15  105]
 [  22    1    0   53    1  750   16    0   44    5]
 [  20    2    0    3    5   25  880    1   22    0]
 [  14    1    2   21    3    0    1  947    6   33]
 [   0    0    0   49    3    0    1    2  917    2]
 [   7    4    0   53    3    1    2    5   23  911]]
```

	precision	recall	f1-score	support
0	0.90	0.99	0.94	980
1	0.96	0.94	0.95	1135
2	0.95	0.07	0.12	1032
3	0.62	0.99	0.76	1010
4	0.95	0.85	0.90	982
5	0.96	0.84	0.89	892
6	0.90	0.92	0.91	958
7	0.79	0.92	0.85	1028
8	0.73	0.94	0.82	974
9	0.86	0.90	0.88	1009
avg / total	0.86	0.83	0.80	10000

For 3:

```
conf_mat:
[[ 976    0    2    0    0    0    0    1    1    0]
 [   1 1110    7    0    1    1    4    8    2    1]
 [  14    0  937    2   16    0   17   14   27    5]
 [  11    2   27  151    3  282    4   19  439   72]
 [   6    1    1    0  877    0   11    4    4   78]
 [  49    0    2    1    4  748   22    3   45   18]
 [  19    1    2    0    4    2  927    0    2    1]
 [   5    4    6    0    3    0    1  989    6   14]
 [  10    1    3    0    4    1    3    2  944    6]
 [  11    4    1    4    3    0    0   10   10  966]]
```

	precision	recall	f1-score	support
0	0.89	1.00	0.94	980
1	0.99	0.98	0.98	1135
2	0.95	0.91	0.93	1032
3	0.96	0.15	0.26	1010
4	0.96	0.89	0.92	982
5	0.72	0.84	0.78	892
6	0.94	0.97	0.95	958
7	0.94	0.96	0.95	1028
8	0.64	0.97	0.77	974
9	0.83	0.96	0.89	1009
avg / total	0.88	0.86	0.84	10000

For 4:

```
conf_mat:
[[ 949    1    9    8    0    3    4    0    6    0]
 [   0 1104    5    4    0    2    0    1   19    0]
 [   5    0  887  122    0    0    0    3   15    0]
 [   0    0    5  990    0   10    0    0    4    1]
 [  13   15   28   70   62    0   40  206   98  450]
 [  10    0    3   46    0  770   13    0   47    3]
 [   5    2   25   34    0   26  827    1   36    2]
 [   0    5   17   81    0    0    0  914   10    1]
 [   1    1    1   17    0    1    1    1  951    0]
 [   0   13   10  217    0    3    2   51   30  683]]
```

	precision	recall	f1-score	support
0	0.97	0.97	0.97	980
1	0.97	0.97	0.97	1135
2	0.90	0.86	0.88	1032
3	0.62	0.98	0.76	1010
4	1.00	0.06	0.12	982
5	0.94	0.86	0.90	892
6	0.93	0.86	0.90	958
7	0.78	0.89	0.83	1028
8	0.78	0.98	0.87	974
9	0.60	0.68	0.64	1009
avg / total	0.85	0.81	0.78	10000

For 5:

```
conf_mat:
[[ 887    0   32    1    1    0    0    1    9   49]
 [   0 1102    8    5    0    0    2    2   16    0]
 [   1    0  894   86    7    0    3    8   33    0]
 [   0    0    4  994    0    0    1    1    8    2]
 [   1    5    2    1  925    0    1    6    9   32]
 [  25    1   17  434   17   36   16    4  309   33]
 [  15    1    3    6   17    0  889    1   25    1]
 [   1    3   13   24    3    0    0  959    7   18]
 [   0    0    1   21    1    1    2    2  941    5]
 [   2   13    2   17   14    0    2   17   23  919]]
```

	precision	recall	f1-score	support
0	0.95	0.91	0.93	980
1	0.98	0.97	0.98	1135
2	0.92	0.87	0.89	1032
3	0.63	0.98	0.76	1010
4	0.94	0.94	0.94	982
5	0.97	0.04	0.08	892
6	0.97	0.93	0.95	958
7	0.96	0.93	0.95	1028
8	0.68	0.97	0.80	974
9	0.87	0.91	0.89	1009
avg / total	0.89	0.85	0.83	10000

For 6:

```
conf_mat:
[[ 967    0    2    4    0    2    0    0    5    0]
 [   0 1098    8    5    1    1    0    6   16    0]
 [   7    1  890  101    8    0    1   10   14    0]
 [   0    0    0  993    0    7    0    2    8    0]
 [   9    0    2    4  864    0   12   10   38   43]
 [  15    0    2   53    0  772    0    0   49    1]
 [ 226    4   30   41  228   29  208    5  181    6]
 [   1    3   35   39    1    0    0  930   10    9]
 [   0    0    2   16    0    0    0    1  955    0]
 [  13    4    4   92    2    4    3    6   68  813]]
```

	precision	recall	f1-score	support
0	0.78	0.99	0.87	980
1	0.99	0.97	0.98	1135
2	0.91	0.86	0.89	1032
3	0.74	0.98	0.84	1010
4	0.78	0.88	0.83	982
5	0.95	0.87	0.90	892
6	0.93	0.22	0.35	958
7	0.96	0.90	0.93	1028
8	0.71	0.98	0.82	974
9	0.93	0.81	0.86	1009
avg / total	0.87	0.85	0.83	10000

For 7:

```
conf_mat:
[[ 962    0    5    6    0    1    1    0    4    1]
 [   0 1099    1    2    0    2    2    0   26    3]
 [   2    1  902   68   12    0    6    0   37    4]
 [   0    0    3  982    0   10    0    0    6    9]
 [  13    3    1    3  819    0    4    0    6  133]
 [  11    1    0   32    1  772   17    0   37   21]
 [  33    3    2    4    5   21  869    1   17    3]
 [   6   47  103  370   10    3   14  133   33  309]
 [   0    2    0   16    1    1    1    0  952    1]
 [   2    2    1   12    1    2    1    0   10  978]]
```

	precision	recall	f1-score	support
0	0.93	0.98	0.96	980
1	0.95	0.97	0.96	1135
2	0.89	0.87	0.88	1032
3	0.66	0.97	0.78	1010
4	0.96	0.83	0.89	982
5	0.95	0.87	0.91	892
6	0.95	0.91	0.93	958
7	0.99	0.13	0.23	1028
8	0.84	0.98	0.91	974
9	0.67	0.97	0.79	1009
avg / total	0.88	0.85	0.82	10000

For 8:

```
conf_mat:
[[ 972    0    0    6    1    0    1    0    0    0]
 [   2 1116    3    4    0    3    2    4    0    1]
 [   5    0  956   37   12    0    6   15    0    1]
 [   0    0    2  990    0    8    0    3    0    7]
 [   4    0    8    0  940    0    4    2    1   23]
 [  18    0    2   47    4  785   15    1    0   20]
 [  14    1    2    1   11    3  926    0    0    0]
 [   8    2   19   41    3    1    0  941    0   13]
 [  43    3   58  501    4   38   18    7  183  119]
 [  13    7    1   15   12    1    0   10    1  949]]
```

	precision	recall	f1-score	support
0	0.90	0.99	0.94	980
1	0.99	0.98	0.99	1135
2	0.91	0.93	0.92	1032
3	0.60	0.98	0.75	1010
4	0.95	0.96	0.95	982
5	0.94	0.88	0.91	892
6	0.95	0.97	0.96	958
7	0.96	0.92	0.94	1028
8	0.99	0.19	0.32	974
9	0.84	0.94	0.89	1009
avg / total	0.90	0.88	0.86	10000

For 9:

```
conf_mat:
[[ 965    1    5    2    1    3    1    1    1    0]
 [   0 1112    1    1    1    1    2   12    5    0]
 [   6    0  948   33    8    1    7    7   22    0]
 [   1    0    3  986    1    3    0    4   12    0]
 [  16    2    1    0  950    0    1    1   11    0]
 [  14    2    1   38    3  749    9    0   76    0]
 [  19    3    0    4   15    4  910    0    3    0]
 [  11    3   14   10    5    1    1  971   12    0]
 [   0    1    0   10    1    0    2    2  958    0]
 [  31   87    4   22  320   11   12  133  383    6]]
```

	precision	recall	f1-score	support
0	0.91	0.98	0.94	980
1	0.92	0.98	0.95	1135
2	0.97	0.92	0.94	1032
3	0.89	0.98	0.93	1010
4	0.73	0.97	0.83	982
5	0.97	0.84	0.90	892
6	0.96	0.95	0.96	958
7	0.86	0.94	0.90	1028
8	0.65	0.98	0.78	974
9	1.00	0.01	0.01	1009
avg / total	0.89	0.86	0.81	10000

different instance for less training size of 0:

For 100:

```
conf_mat:
[[ 485    1   32   47    8   34  174   21   60  118]
 [   0 1115    1    4    0    0    1    0   14    0]
 [   0    3  837   89   10    1   14   19   57    2]
 [   0    0    1 1000    0    1    0    0    8    0]
 [   0    1    1    1  933    0    7    2    8   29]
 [   0    2    5  101    5  635   23    0  110   11]
 [   0    3    2    5   13    7  909    1   17    1]
 [   0    9    9   22    4    0    0  948   18   18]
 [   0    0    0   25    5    0    3    1  940    0]
 [   0   10    2   30    5    1    2    4    8  947]]
```

```
             precision    recall  f1-score   support

          0       1.00      0.49      0.66       980
          1       0.97      0.98      0.98      1135
          2       0.94      0.81      0.87      1032
          3       0.76      0.99      0.86      1010
          4       0.95      0.95      0.95       982
          5       0.94      0.71      0.81       892
          6       0.80      0.95      0.87       958
          7       0.95      0.92      0.94      1028
          8       0.76      0.97      0.85       974
          9       0.84      0.94      0.89      1009

avg / total       0.89      0.87      0.87     10000
```

For 500:

```
conf_mat:
[[607   0  65  64   3  96  71   6   8  60]
 [  0 968  33   2   1   1   1 126   2   1]
 [  0   1 938  41   9   0   7  27   8   1]
 [  0   0   3 986   1   4   1   6   4   5]
 [  0   3   4   1 876   0   8  12   3  75]
 [  5   8   6 106   3 677  10   4  60  13]
 [  1   1   7   7   6   4 920   0  11   1]
 [  0   0   5   4   2   0   1 998   5  13]
 [  0  10   7  68   2   1   2   7 877   0]
 [  0   8   1  51   2   2   2  11   3 929]]
```

```
             precision    recall  f1-score   support

          0       0.99      0.62      0.76       980
          1       0.97      0.85      0.91      1135
          2       0.88      0.91      0.89      1032
          3       0.74      0.98      0.84      1010
          4       0.97      0.89      0.93       982
          5       0.86      0.76      0.81       892
          6       0.90      0.96      0.93       958
          7       0.83      0.97      0.90      1028
          8       0.89      0.90      0.90       974
          9       0.85      0.92      0.88      1009

avg / total       0.89      0.88      0.88     10000
```

For 1000:

```
conf_mat:
[[ 863    0   29   13    3   23   33    0    5   11]
 [   0 1118    5    1    0    0    1    0   10    0]
 [   1    1  938   44    6    1    8   12   20    1]
 [   0    0    1  990    0    4    0    2   12    1]
 [   0    6    4    1  859    0   10    8    8   86]
 [   8    0    3   35    0  732   14    1   93    6]
 [   4    4    4    2    7    2  908    0   26    1]
 [   2    5   10   19    4    1    0  928   17   42]
 [   0    2    5   10    1    0    2    0  951    3]
 [   0    9    0   25    5    5    1    5    9  950]]
```

```
             precision    recall  f1-score   support

          0       0.98      0.88      0.93       980
          1       0.98      0.99      0.98      1135
          2       0.94      0.91      0.92      1032
          3       0.87      0.98      0.92      1010
          4       0.97      0.87      0.92       982
          5       0.95      0.82      0.88       892
          6       0.93      0.95      0.94       958
          7       0.97      0.90      0.94      1028
          8       0.83      0.98      0.90       974
          9       0.86      0.94      0.90      1009

avg / total       0.93      0.92      0.92     10000
```

For 2000:

```
conf_mat:
[[ 902    0   11   11    0   11   25    0    5   15]
 [   0 1119    2    3    1    1    0    4    5    0]
 [   2    0  922   54    5    0   13   23   12    1]
 [   0    0    0  996    0    5    0    4    5    0]
 [   0    3    2    1  814    0   13    5    1  143]
 [   3    1    7   64    0  762   11    0   31   13]
 [   2    1    5    6    1   10  927    0    6    0]
 [   1    2    8    7    1    0    0  994    4   11]
 [   0    3   12   52    2    1    3    4  891    6]
 [   2   13    1   14    3    2    4   11    4  955]]
```

```
             precision    recall  f1-score   support

          0       0.99      0.92      0.95       980
          1       0.98      0.99      0.98      1135
          2       0.95      0.89      0.92      1032
          3       0.82      0.99      0.90      1010
          4       0.98      0.83      0.90       982
          5       0.96      0.85      0.90       892
          6       0.93      0.97      0.95       958
          7       0.95      0.97      0.96      1028
          8       0.92      0.91      0.92       974
          9       0.83      0.95      0.89      1009

avg / total       0.93      0.93      0.93     10000
```

For 3000:

```
conf_mat:
[[ 907    1   17    3    0   32    4    0    6   10]
 [   0 1128    2    1    0    0    1    0    3    0]
 [   0   12  825   92    5    3   11    9   74    1]
 [   0    0    1  981    1   21    0    0    3    3]
 [   5    7    2    7  701    0   28    4   38  190]
 [   5    1    1   38    0  801   15    0   25    6]
 [   4    2    1    4    5   15  914    0   12    1]
 [   1   23    6   79    3    1    0  853   31   31]
 [   1    1    0   19    1    1    2    1  947    1]
 [   1   11    0   39    1    4    1    1   35  916]]
```

```
             precision    recall  f1-score   support

          0       0.98      0.93      0.95       980
          1       0.95      0.99      0.97      1135
          2       0.96      0.80      0.87      1032
          3       0.78      0.97      0.86      1010
          4       0.98      0.71      0.83       982
          5       0.91      0.90      0.91       892
          6       0.94      0.95      0.95       958
          7       0.98      0.83      0.90      1028
          8       0.81      0.97      0.88       974
          9       0.79      0.91      0.85      1009

avg / total       0.91      0.90      0.90     10000
```

For 5000:

```
conf_mat:
[[ 959    1    1    2    2    6    4    1    3    1]
 [    0 1127    1    4    0    0    0    0    3    0]
 [    4   10  885   38    9    4    9   25   47    1]
 [    1    0    3  975    0   19    0    1   10    1]
 [    6    5    0    0  875    0    4    9    8   75]
 [   19    2    0   24    1  775    8    0   52   11]
 [   15    3    2    2   10   25  885    0   15    1]
 [    2   63    3    8    4    0    1  922   10   15]
 [    0    7    1    9    0    3    1    1  947    5]
 [    3   10    2   10    2    7    1   13    9  952]]
```

```
             precision    recall  f1-score   support

          0       0.95      0.98      0.96       980
          1       0.92      0.99      0.95      1135
          2       0.99      0.86      0.92      1032
          3       0.91      0.97      0.94      1010
          4       0.97      0.89      0.93       982
          5       0.92      0.87      0.90       892
          6       0.97      0.92      0.95       958
          7       0.95      0.90      0.92      1028
          8       0.86      0.97      0.91       974
          9       0.90      0.94      0.92      1009

avg / total       0.93      0.93      0.93     10000
```

As a result, if the training data set is unbalanced, when I test the data, the precision of less training data is higher than other number, the recall is lower than other number. And the prediction of this number are tend to balance for all number. With the increasing of the number which has the less training data, the precision of this number is decreased and the recall of this number is increased, the accuracy is increased.