This project aims to recognize the hand writing numbers. For solving this problem, I try several classifiers at beginning including, KNN, SVM, Ensemble Methods (Begging and AdaBoost) and CNN. For testing the performance of these methods, I split train set as 34500 train data and 500 test data to evaluate the performance of all methods. The accuracies of all methods are show in the figure 1. Ensemble Methods failed because it cost too much time (over 3 days) and I interrupted process and give up this method. As shown in the Figure 1, the accuracy of CNN and SVM over 95%. So I use these two methods to predict the test set. The advantage for SVM is much faster than CNN. However, when I try to use the classifier to the real test data, the performance of SVM is terrible. Most of entries go to the same class. I also try to change result by inputting different parameters, however, I failed on all attempts. So I choose CNN as my final method.

For CNN method, I find an open-source software library called tensorflow. It is produced by Google. Tensorflow sample I find includes two convolutional layers and two pooling layers. I follow this sample and change the parameters of networks. The input of tensorflow should less than 1, so I divide by 255 on both train_X and text_X. Our input data is 64*64, 4096 features, these features will be map to 2048 features in my code. And then these features will map to 10 labels. For reducing the dependence of training data and preventing overfitting, I also add drop out method to control the complexity of the model. For selecting the batches in the tensorflow, I import coo_matrix from scipy.sparse library, which can random get a certain number data as a batch. I try this methods for 10000 iterations at first. It costs 15 hours and accuracy is 94% in Kaggle. (I cannot install tensorflow-gpu). And then I tried to increase the number of iteration to 30000. However, the accuracy shows the same with 10000 iterations. For finding the error, I try to compare these two results and find that the predictions are different although the accuracy is same. So I decide to run another result and get the most frequent value as result. In the third result, I use a new library called keras, which is also used tensorflow but much easier to understand and implementation. If it shows three different predictions in one entry, I will random pick one. I also try to preprocess the data, for example set the pixel to 0 if it smaller than 0.6 after divided by 255. However, it does not work to improve accuracy. Figure 3 is the data which do not preprocess and Figure 4 is the picture which do preprocess. We found that, the Figure 3 is much clearer than Figure 4. The accuracy of results in Kaggle and their methods is shows in Figures 5.

Method	Accuracy
KNN	0.908
SVM	0.952
Ensemble Methods	N/A
CNN(tensorflow)	0.973

Figure 1 The Accuracy of Methods

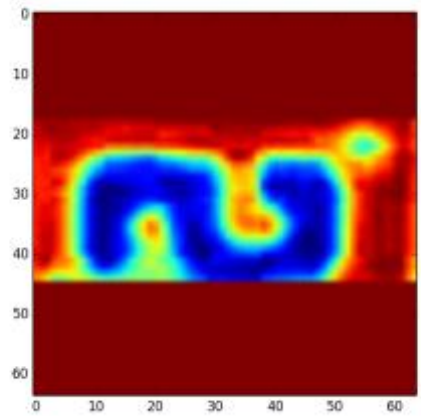

Figure 2 The image of non-preprocess data

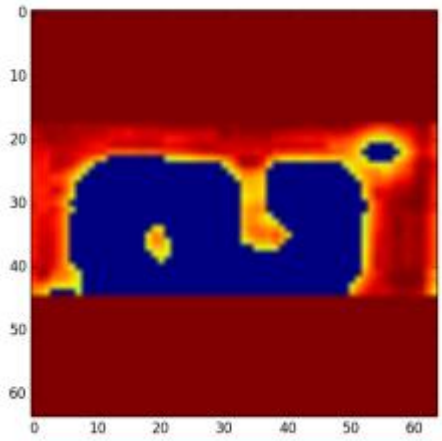

Figure 3 The image of preprocess data

Attempt	Accuracy	Methods
1	0.944	Tensorflow(10000 iterations)
2	0.944	Tensorflow(30000 iterations)
3	0.096	Wrong write to .csv
4	0.956	Most frequent in 3 file
5	0.916	Keras(drop out 0.5)
6	0.968	Keras(drop out 0.25)
7	0.972	Most frequent in 3 files
8	0.968	Add preprocess in Keras(0.25)

Figure 4 The result of all predictions and its method