COGS 260: Assignment 2

Lianming Shi
Department of Electrical and Computer Engineering
University of California, San Diego
9500 Gilman Dr, La Jolla
San Diego, CA 92093
l5shi@eng.ucsd.edu

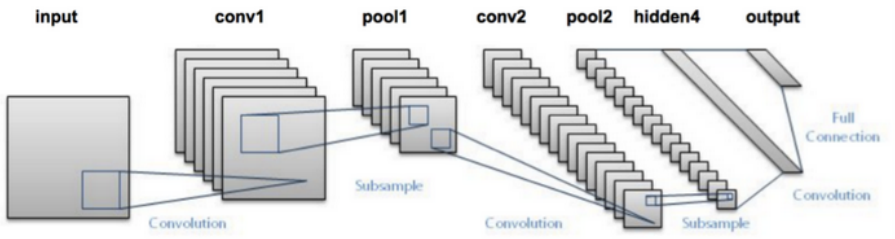
Abstract

In this assignment, we are supposed to compare the performance of different methods of image recognition on MNIST dataset. The database consists of a total of 70000 including 60000 training samples and 10000 test samples. I used K-NN, Support Vector Machine, Spatial Pyramid Matching, Convolutional Neural Networks, and Deep Belief Nets. After training the CNN model with the whole MNIST dataset, I divide the dataset into 10000 training-set and 2000 test-set to overcome memory and computational challenges on other methods.

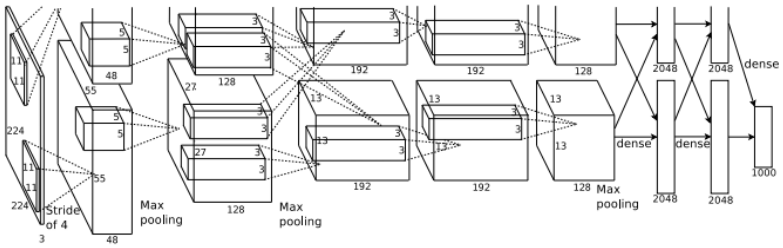
1. Convolutional Neural Network

1.1 Method

In this part, I used Convolutional Neural Networks to train a model based on MNIST dataset. The network architectures I choose are LeNet and AlexNet. LeNet Start with an image of 32 x 32 x 1 and goal was to recognize handwritten digit. Alex network had a very similar architecture as LeNet but was deeper, with more filters per layer, and with stacked convolutional layers.



LeNet-Structure



AlexNet-Structure

1.2 Experiment

LeNet optimizer: Adam

Learning rate: 0.01

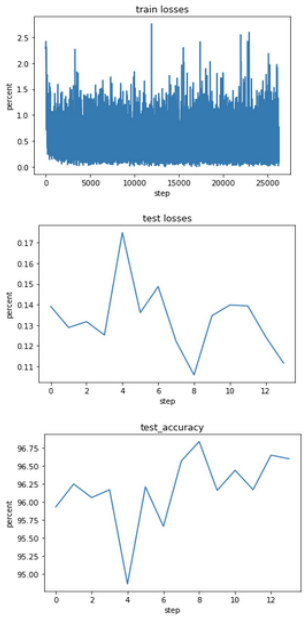


Fig.1

Learning rate: 0.001

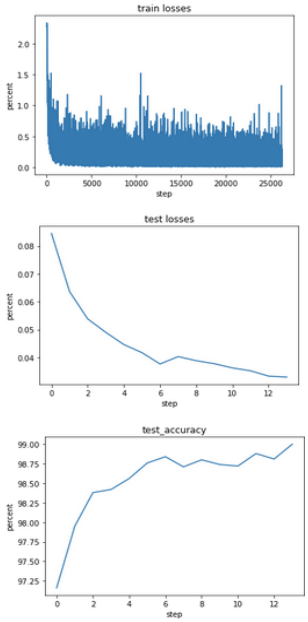


Fig.2

Learning rate: 0.0001

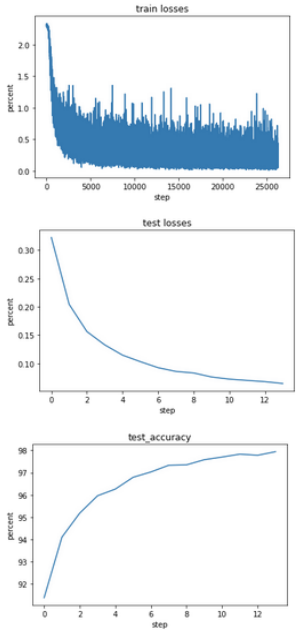


Fig.3

LeNet: optimizer: SGD

Learning rate: 0.01

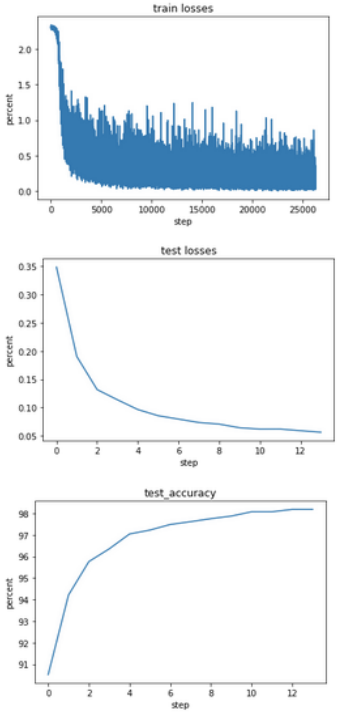


Fig.4

Learning rate: 0.001

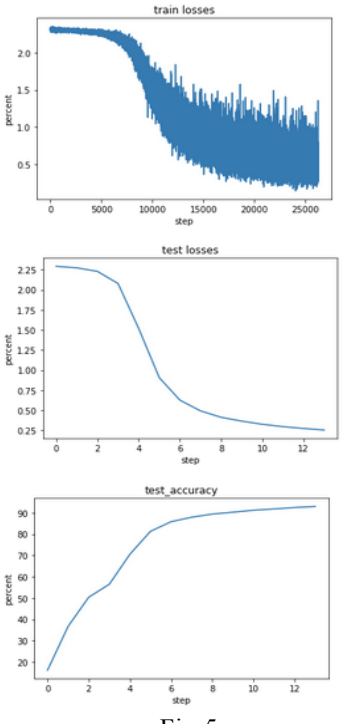


Fig.5

Learning rate: 0.0001

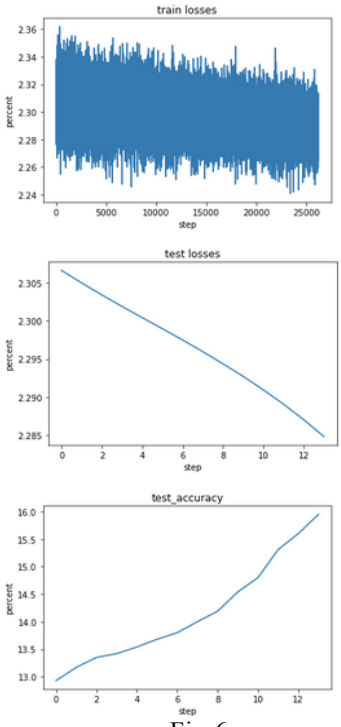


Fig.6

AlexNet: optimizer: Adam

Learning rate: 0.001

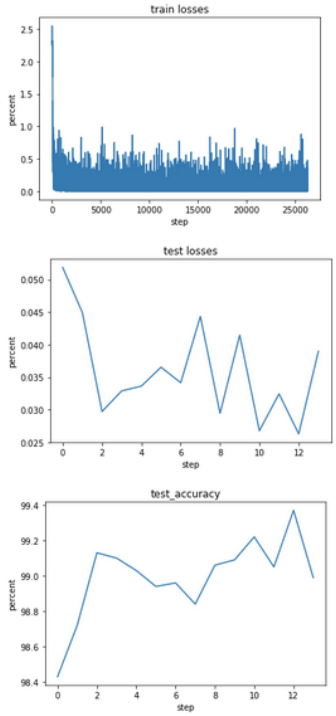


Fig.7

Learning rate: 0.0001

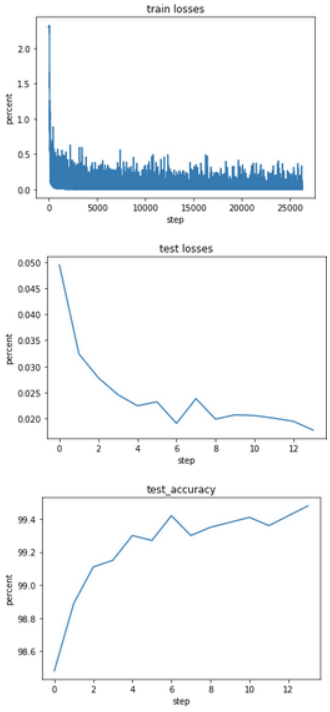


Fig.8

AlexNet: optimizer: SGD

Learning rate: 0.01

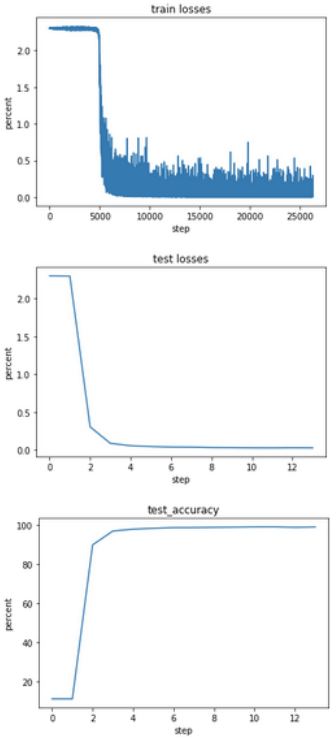


Fig.9

Learning rate: 0.0001

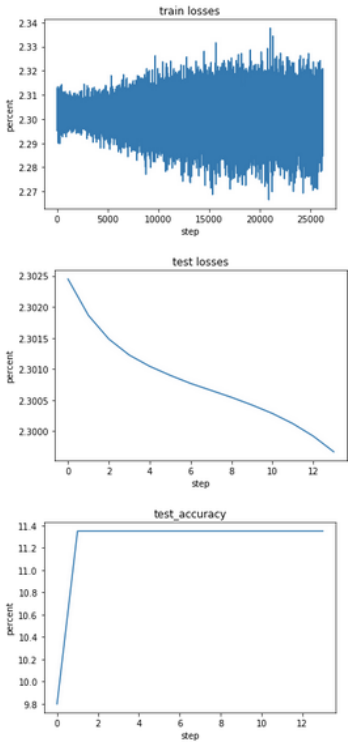


Fig.10

1.3 Discussion

I used different optimizers and learning rates on the two network architectures. The figures above are the loss and test accuracy based on these different parameters. For optimizer, the performance of Adam is much better than SGD due to high accuracy, i.e. Fig 2 and Fig 5. Indeed, Adam use adaptive method with momentum to update the weights, which can smoothen the optimization process. Overall, Adam is an excellent algorithm because it achieves good results fast even though with different learning rate.

For learning rates, we are likely to find that when learning rate is too large or too small, the accuracy will decrease. The reason is that if the learning rate is small, then training is more reliable, but optimization will take a lot of time because steps towards the minimum of the loss function are tiny, i.e. Fig 1. If the learning rate is large, then training may not converge or even diverge. Weight changes can be so big that the optimizer overshoots the minimum and makes the loss worse, i.e. Fig 6.

For network architectures, AlexNet have a better performance overall, because AlexNet has more converlutional layers than LeNet so that AlexNet was able to ensure its neural network extract more features which leads to higher performance.

2. K-Nearest Neighbor

2.1 Method

In this part, I used K nearest neighbor classification to train my model based on MNIST dataset. K-NN is a non-parametric learning algorithm. The “K” is KNN algorithm is the nearest neighbor we wish to take vote from. If K = 1, then the object is simply assigned to the class of its single nearest neighbor. I tried different k (1-9) and then different distance function (Euclidean and Manhattan distance). After that, I was able to compare their performance based on accuracy.

2.2 Experiment

Euclidean Distance

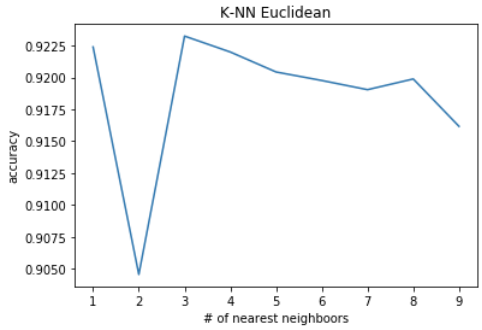


Fig.11

Manhattan Distance

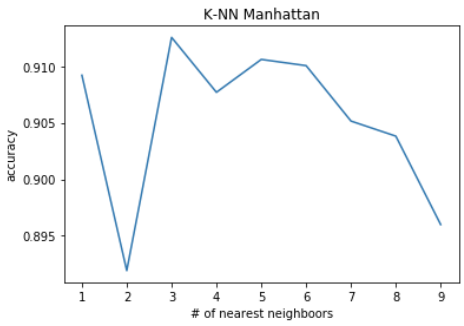


Fig.12

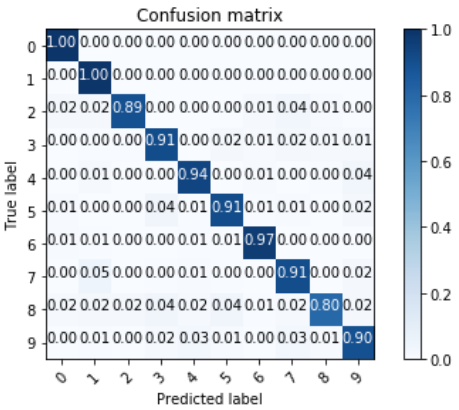


Fig.13

Best Accuracy: 0.923

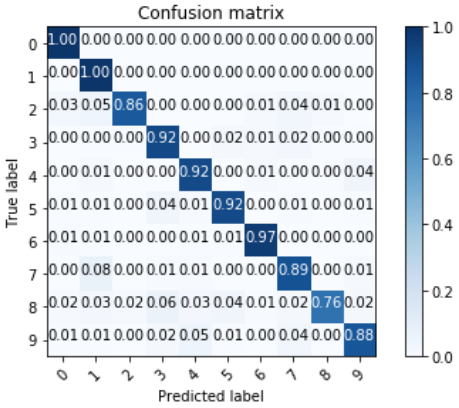


Fig.14

Best Accuracy: 0.912

2.3 Discussion

By comparing Fig.11 and Fig. 12, we can find that K=3 is the optimal decision for this dataset. It makes sense because if K is too large, the output will be constant regardless of surrounding neighbor, and you will get a model that under-fits the data. If K is too small, you will get a model that over-fits the data.

Fig.13 and Fig.14 are the confusion matrices with different K. Euclidean Distance function performs much better than Manhattan distance function due to its high accuracy. Indeed, Manhattan distance is simply the sum of absolute differences between points. Euclidean distance is the straight line distance between points regardless of the network that you are using.

3. Support Vector Machine

3.1 Method

In this part, I used support vector machine to train a model based on MNIST data. Support Vector Machine is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. I used SVM function form sklearn library. Then, I was able to generate the confusion matrices.

3.2 Experiment

Parameter: C=5, gamma = 0.05

Without normalized

Normalized

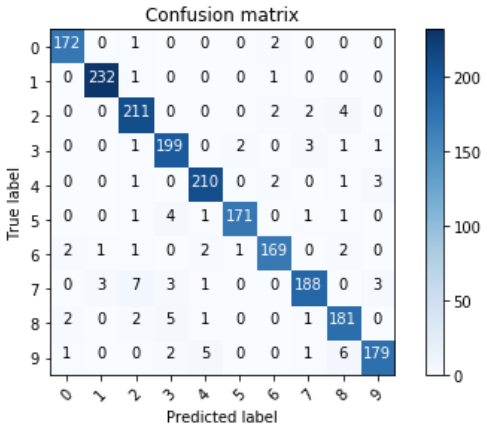


Fig.15

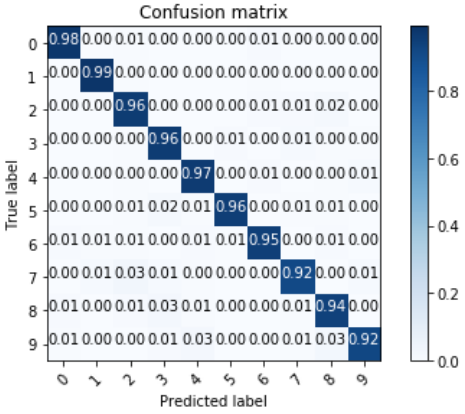


Fig.16

Parameter: C=5, gamma = 0.1

Without normalized

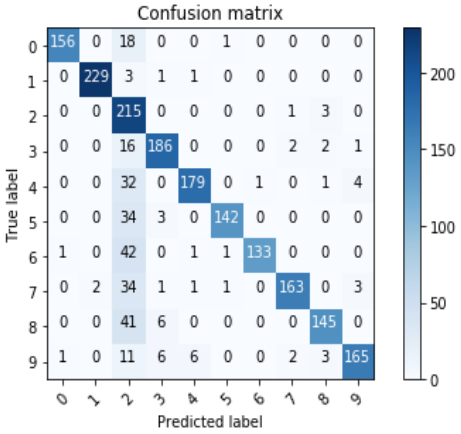


Fig.17

Normalized

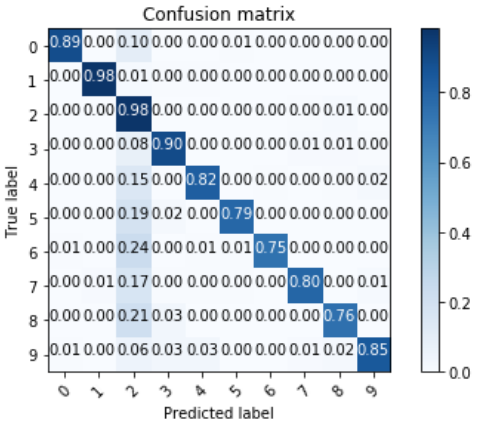


Fig.18

Parameter: C=0.5, gamma = 0.05

Without normalized

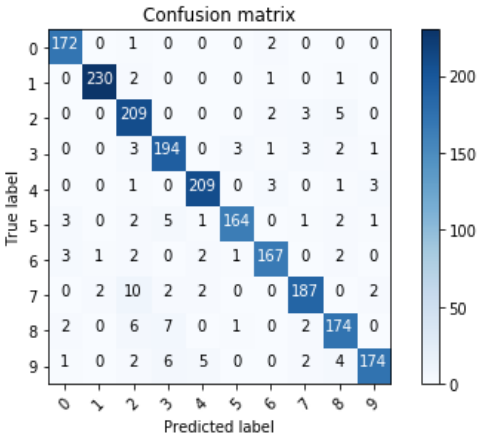


Fig.19

Normalized

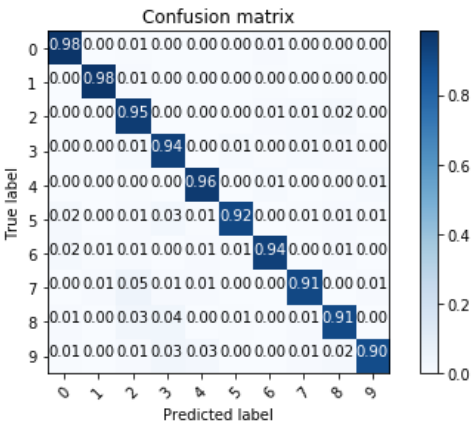


Fig.20

Table1. Different Parameter performance

Parameters	Accuracy
C=5, gamma = 0.05	0.955
C=5, gamma = 0.1	0.852
C=0.5, gamma = 0.05	0.939

3.3 Discussion

Overall, SVM method has a good performance. By tuning the parameters C and gamma, I found that the accuracy decreases as gamma increase or C decrease. It makes sense because gamma controls the curvature of the hyperplane and, thus, high gamma value will cause overfitting. Moreover, parameter C represents cost control softness of the margin, and high C value will reduce error rate.

4. Spatial Pyramid Matching

4.1 Method

In this part, I used Spatial Pyramid Matching to train a model based on MNIST dataset. The SPM algorithm basically divides an image into several regions in the spatial pyramid way, and then use bag of wards algorithm on each region. After that, we are able to determine each region's information based on its histogram returned by SPM, which enables us to find its category.

4.2 Experiment

LEVEL 1

	precision	recall	f1-score	support
0	0.68	0.79	0.73	175
1	0.86	0.97	0.91	234
2	0.76	0.65	0.70	219
3	0.55	0.59	0.57	207
4	0.67	0.55	0.60	217
5	0.64	0.36	0.46	179
6	0.79	0.75	0.77	178
7	0.62	0.70	0.66	205
8	0.50	0.51	0.50	192
9	0.57	0.74	0.64	194
avg / total	0.67	0.67	0.66	2000

Fig.21

LEVEL 2

	precision	recall	f1-score	support
0	0.81	0.87	0.84	175
1	0.93	0.98	0.96	234
2	0.78	0.79	0.78	219
3	0.69	0.72	0.70	207
4	0.81	0.72	0.77	217
5	0.81	0.66	0.73	179
6	0.85	0.83	0.84	178
7	0.83	0.79	0.81	205
8	0.66	0.65	0.65	192
9	0.69	0.82	0.75	194
avg / total	0.79	0.79	0.79	2000

Fig.22

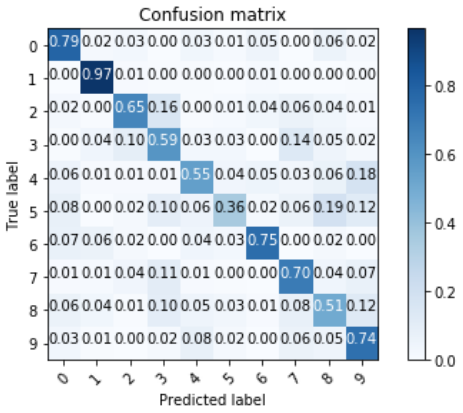


Fig.23

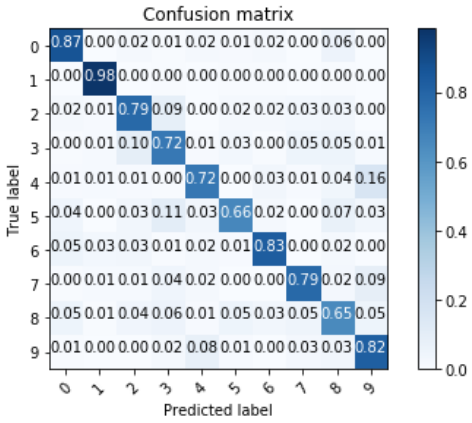


Fig.24

```
Accuracy: 0.66
```

```
Accuracy: 0.78
```

4.3 Discussion

Obviously, the performance of level two SPM, with 78% accuracy, to train the mode is better than only level one, with 66% accuracy. Indeed, higher level of SPM will have a better resolution. Thus, as SPM level increases we are more likely to extract more features from an image, which lead to higher accuracy rate.

5. Deep Belief Nets

5.1 Method

In this part, I used Deep belief Network to train a model based on MNIST dataset. DBN is multi-layer belief networks. The first step of training DBN is to learn a layer of features from the visible units. Then, the next step is to treat the activations of previously trained features as visible unites and learn features of features in a second hidden layer. Finally, the whole DBN is trained when the learning for the final hidden layer is achieved. After that, I was able to project the activations of the last layer in two-dimensional space by using t-SNE.

5.2 Experiment

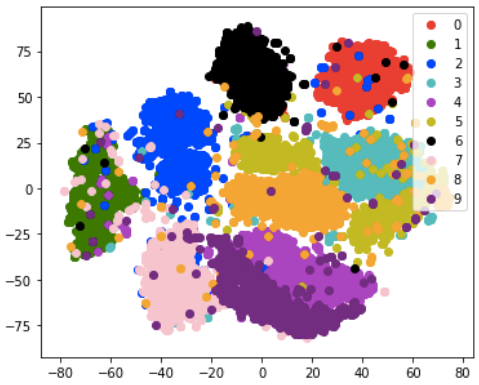


Fig.25 raw input data

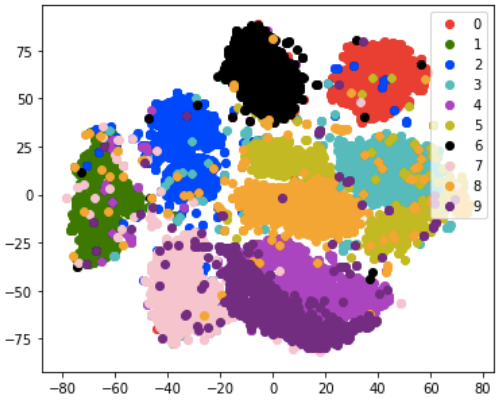


Fig.26 DBN processed data

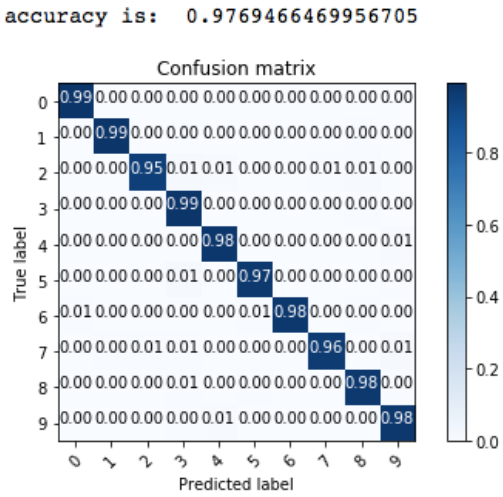


Fig.27

5.3 Discussion

By projecting the activations of the last layer in two-dimensional space using t-SNE, I observed that data points with the same class label clustered together. Fig 25 and Fig 26 are ground truth and DBN processed 2-D visualization of digits cluster from different classes, respectively. Fig 25 shows that Fig 26 indicates that there are only small mixtures in each cluster. Overall, the accuracy is 97.69%. Thus, the generative model--DBN classifier performs well.

6. Summary

Table 2: performance of different model

Train data	Test data	Method	Best Accuracy
60000	10000	CNN-LENET	98.8
60000	10000	CNN-ALEXNET	99.1
60000	10000	DBN	97.7
10000	2000	K-NN	92.3
10000	2000	SVM	95.5
10000	2000	SPM	78.3

For large dataset, CNN-AlexNet has the best performance.

For small dataset, SVM has the best performance.

References

[1] The MNIST Database of handwritten digits. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[2] K-Nearest Neighbor http://scikit-learn.org/stable/modules/neighbors.html

[3] Support Vector Machine SVM: http://scikit-learn.org/stable/modules/svm.html

[4] Spatial Pyramid Matching: https://github.com/CyrusChiu/Image-recognition

[5] Deep Brief Networks: https://github.com/fuzimaoxinan/Tensorflow_deep-belief-network