Handwriting Number Recognition

Hanyi Zhang, Linqiao Shang, Xin Li

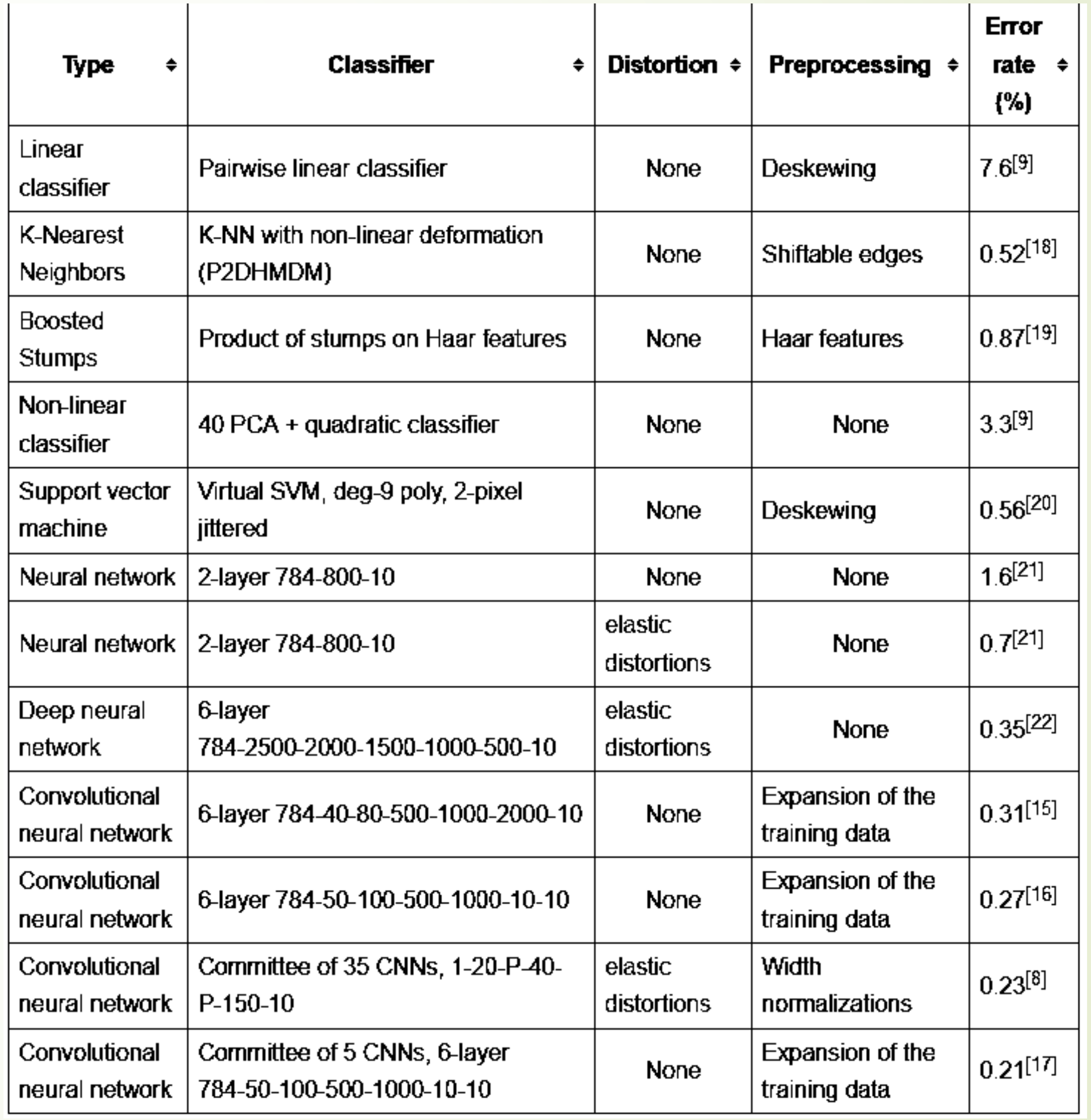
Introduction

- Picture Recognition is a hot topic these years due to the development of Machine Learning and Artificial Intelligence.
- It's an easy task for human to read handwritten words and digits. But it take much more time for machines to decide.
- We start with digits recognition due to its less outputs and symbol for words.

WHEN A USER TAKES A PHOTO, THE APP SHOULD CHECK WHETHER THEY'RE IN A NATIONAL PARK...
SURE, EASY GIS LOOKUP. GIMME A FEW HOURS.
...AND CHECK WHETHER THE PHOTO IS OF A BIRD.
I'LL NEED A RESEARCH TEAM AND FIVE YEARS.
IN CS, IT CAN BE HARD TO EXPLAIN THE DIFFERENCE BETWEEN THE EASY AND THE VIRTUALLY IMPOSSIBLE.

Related Works

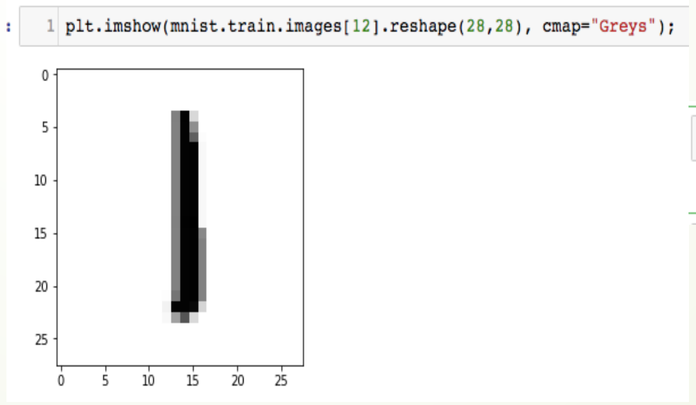
- In 1998, a best-case error rate of 0.42 percent was achieved on the database by researchers using a new classifier called the Gradient-Based Learning.
- In 2016, an error rate of 0.21 percent, improving on the previous best result, was reported by researchers using a method called convolutional neural networks



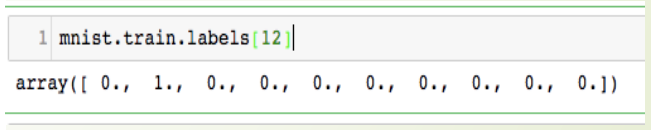
Type	Classifier	Distortion	Preprocessing	Error rate (%)
Linear classifier	Pairwise linear classifier	None	Deskewing	7.6[9]
K-Nearest Neighbors	K-NN with non-linear deformation (P2DHMDM)	None	Shiftable edges	0.52[18]
Boosted Stumps	Product of stumps on Haar features	None	Haar features	0.87[19]
Non-linear classifier	40 PCA + quadratic classifier	None	None	3.3[9]
Support vector machine	Virtual SVM, deg-9 poly, 2-pixel jittered	None	Deskewing	0.56[20]
Neural network	2-layer 784-800-10	None	None	1.6[21]
Neural network	2-layer 784-800-10	elastic distortions	None	0.7[21]
Deep neural network	6-layer 784-2500-2000-1500-1000-500-10	elastic distortions	None	0.35[22]
Convolutional neural network	6-layer 784-40-80-500-1000-2000-10	None	Expansion of the training data	0.31[15]
Convolutional neural network	6-layer 784-50-100-500-1000-10-10	None	Expansion of the training data	0.27[16]
Convolutional neural network	Committee of 35 CNNs, 1-20-P-40-P-150-10	elastic distortions	Width normalizations	0.23[8]
Convolutional neural network	Committee of 5 CNNs, 6-layer 784-50-100-500-1000-10-10	None	Expansion of the training data	0.21[17]

MNIST database

- The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems.
- The MNIST database contains 60,000 training images and 10,000 testing images. Each individual digit is an array with 784 (28x28) values. Each value represents the color for one pixel. If we reshape a training sample to 28x28 and plot it, we can see the original digit, and we could also see the label of this digit. The label format is the *one-hot encoding* style. This means that the label corresponds to the index of the array where the value is 1.



```
plt.imshow(mnist.train.images[12].reshape(28,28), cmap="Greys");
```

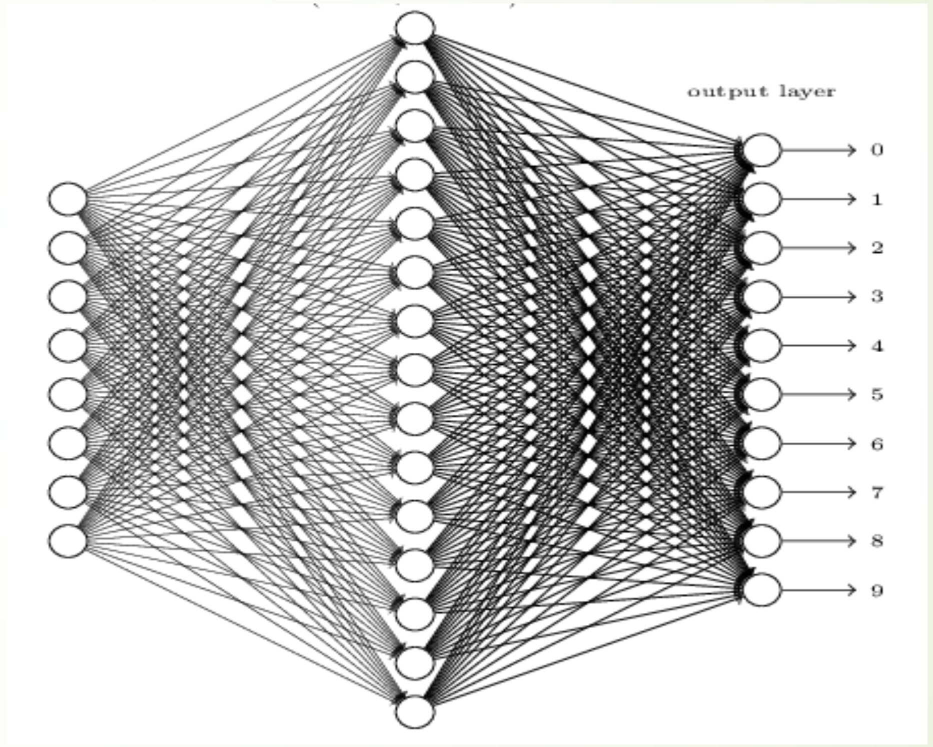


```
mnist.train.labels[12]
```

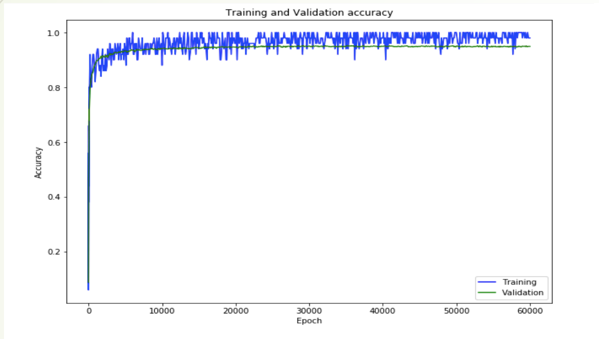
```
array([ 0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.])
```

Neural Network

- Input Layer
- Hidden Layer
- Output Layer
- $C(w,b) \equiv \frac{1}{2n}\sum_x ||y-a||^2$
- $w'_k = w_k - \frac{\eta}{m}\sum_j \frac{\partial C_{Xj}}{\partial w_k}$
- $b'_l = b_l - \frac{\eta}{m}\sum_j \frac{\partial C_{Xj}}{\partial b_l}$

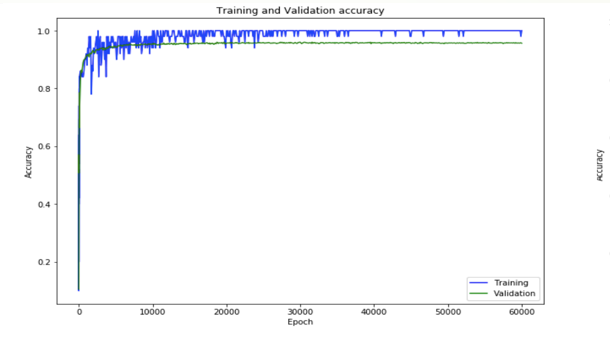


Neural Network Result



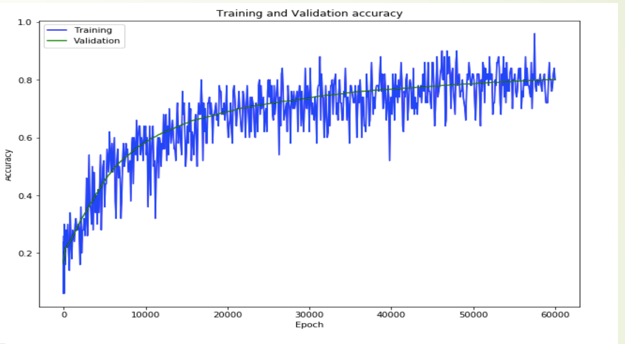
```
]:  # Print final accuracies
    print("Validation Accuracy:", accuracy.eval({x: mnist.validation.images, y:
    print("Test Accuracy:", accuracy.eval({x: mnist.test.images, y: mnist.test.l
Validation Accuracy: 0.9486
Test Accuracy: 0.9505
```

n= 30, *η*= 1.0, *m*= 50, a=0.9486



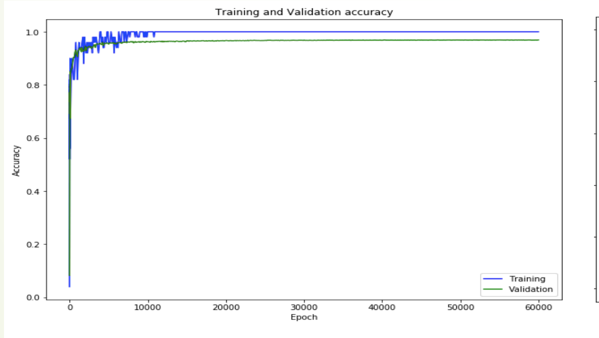
```
# Print final accuracies
print("Validation Accuracy:", accuracy.eval({x: mnist.validation.images, y: m
print("Test Accuracy:", accuracy.eval({x: mnist.test.images, y: mnist.test.la
Validation Accuracy: 0.9564
Test Accuracy: 0.9588
```

n= 100, *η*= 1.0, *m*= 50, a=0.959

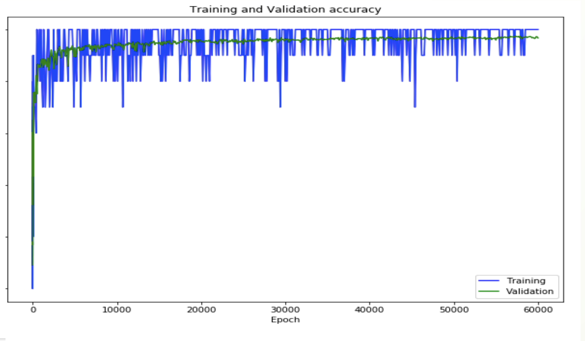


```
# Print final accuracies
print("Validation Accuracy:", accuracy.eval({x: mnist.validation.images, y:
print("Test Accuracy:", accuracy.eval({x: mnist.test.images, y: mnist.test.
alidation Accuracy: 0.8004
est Accuracy: 0.7979
```

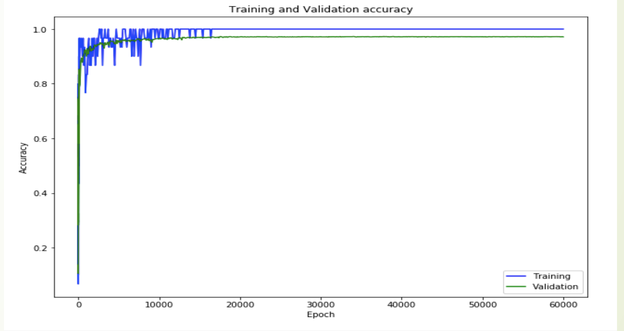
n= 300, *η*= 0.001, *m*= 50



```
# Print final accuracies
print("Validation Accuracy:", accuracy.eval({x: mnist.validation.images, y
print("Test Accuracy:", accuracy.eval({x: mnist.test.images, y: mnist.test
Validation Accuracy: 0.969
Test Accuracy: 0.9664
```



```
Print final accuracies
rint("Validation Accuracy:", accuracy.eval({x: mnist.validation.images,
rint("Test Accuracy:", accuracy.eval({x: mnist.test.images, y: mnist.te
ation Accuracy: 0.9672
Accuracy: 0.9662
```

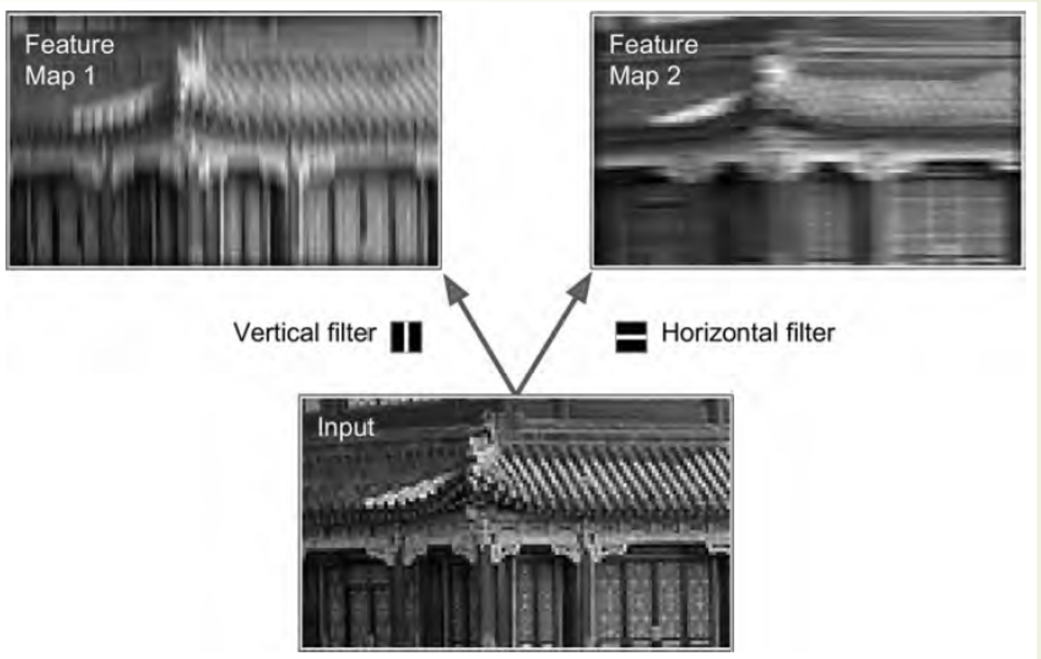


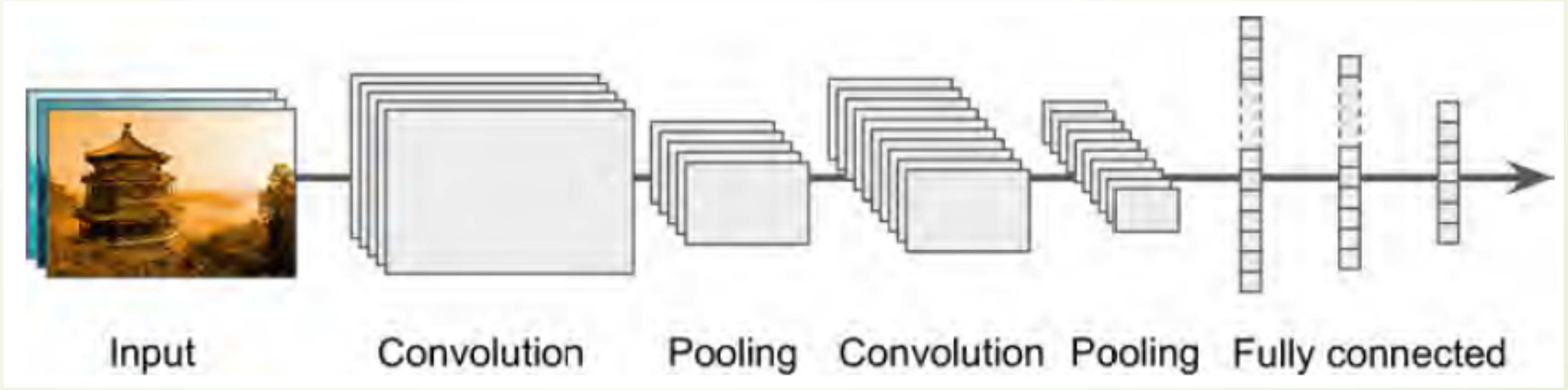
```
# Print final accuracies
print("Validation Accuracy:", accuracy.eval({x: mnist.validation.images, y
print("Test Accuracy:", accuracy.eval({x: mnist.test.images, y: mnist.test
Validation Accuracy: 0.9714
Test Accuracy: 0.9689
```

n= 300, *η*= 1.6, *m*= 50, a= 0.969 *n*= 300, *η*= 1.6, *m*= 10, a= 0.9672 *n*= 300, *η*= 1.6, *m*= 30, a= 0.9714

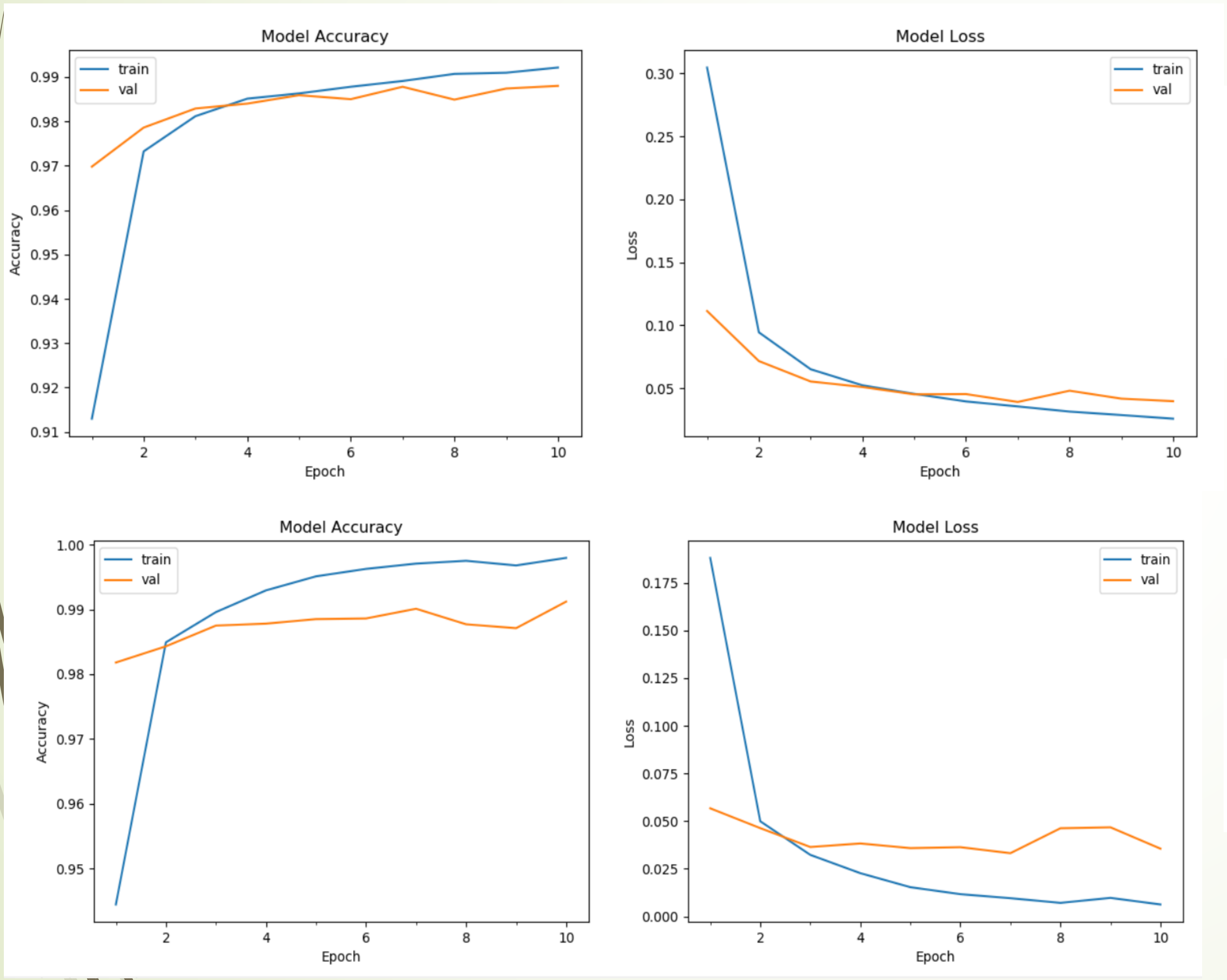
Convolution Neural Network Solution

- Convolutional layer
- Pooling layer
- Fully connected layer
- Dropout layer (optional)

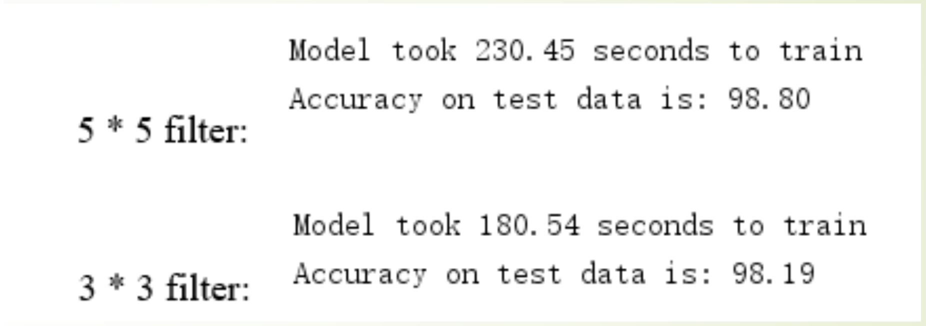




Convolution Neural Network result



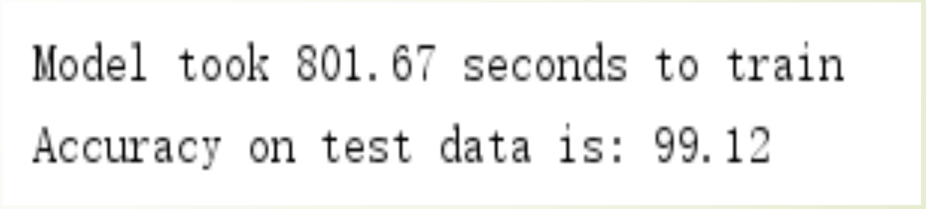
- 1 convolutional Layer (5*5)



5 * 5 filter:
```
Model took 230.45 seconds to train
Accuracy on test data is: 98.80
```

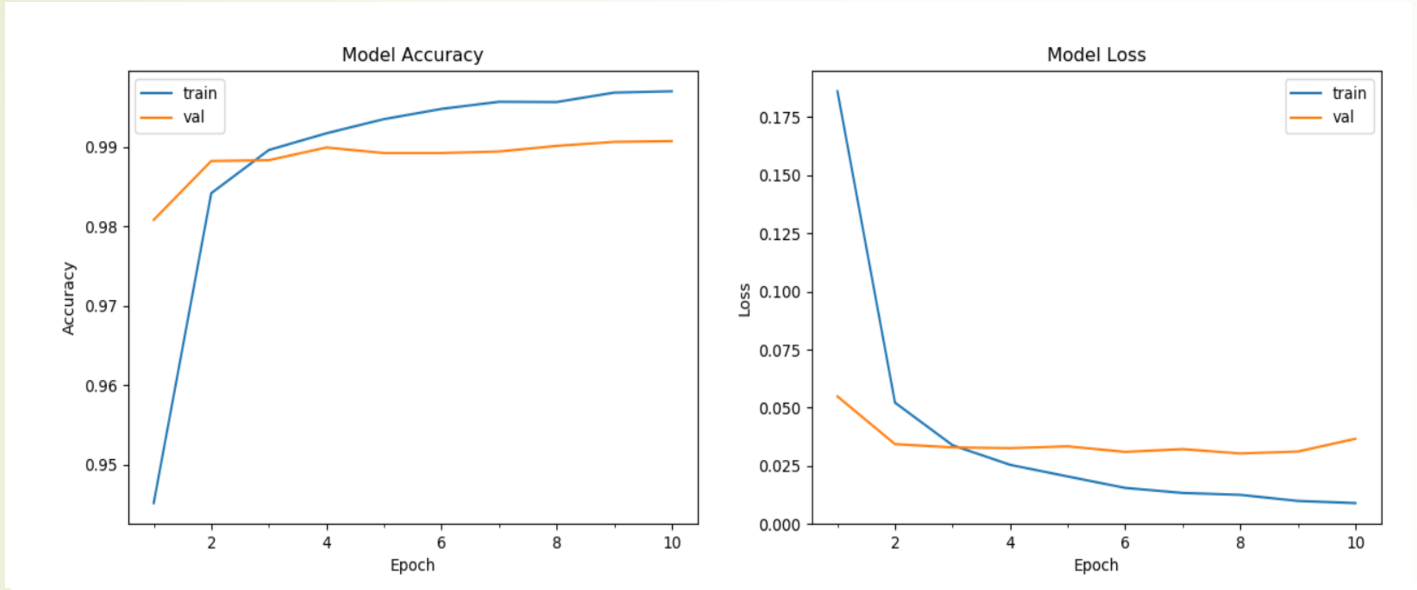
3 * 3 filter:
```
Model took 180.54 seconds to train
Accuracy on test data is: 98.19
```

- 2 convolutional layers (3*3)

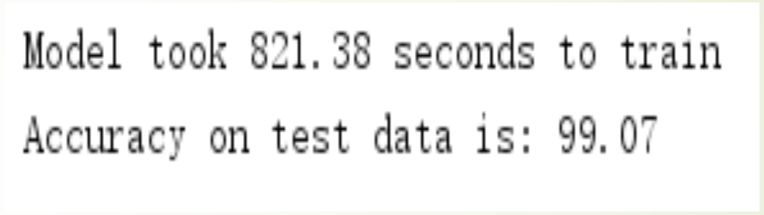


```
Model took 801.67 seconds to train
Accuracy on test data is: 99.12
```

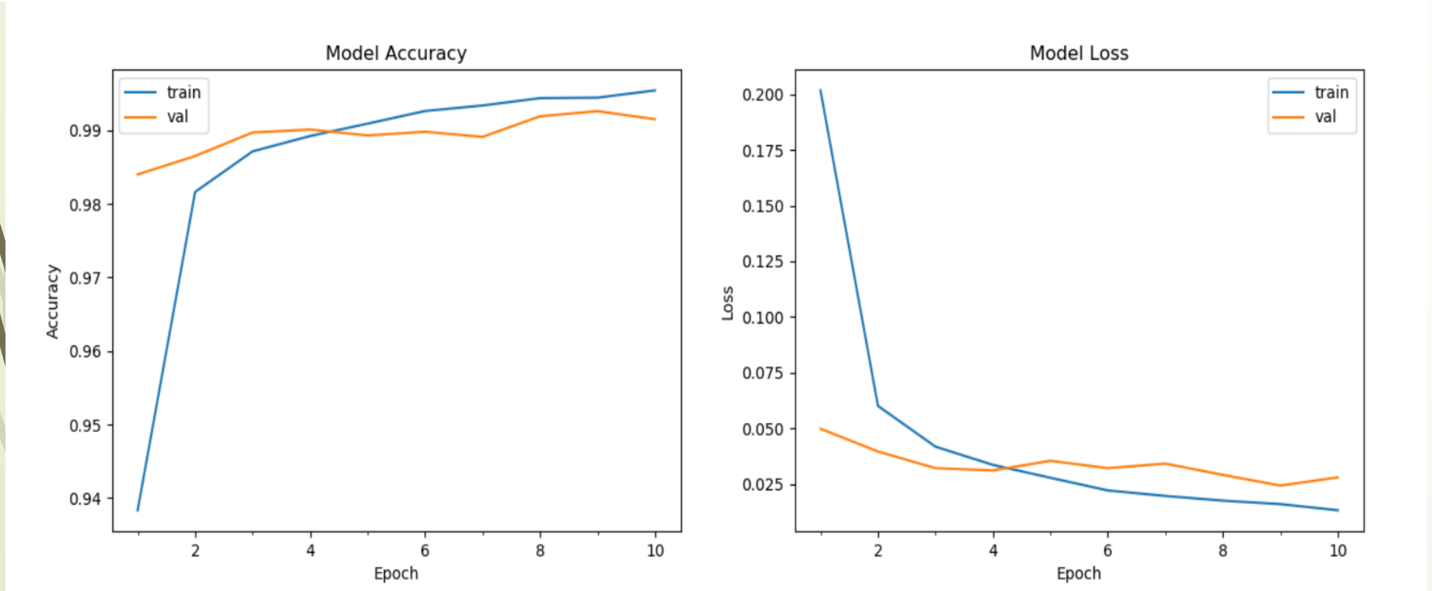
Convolution Neural Network result



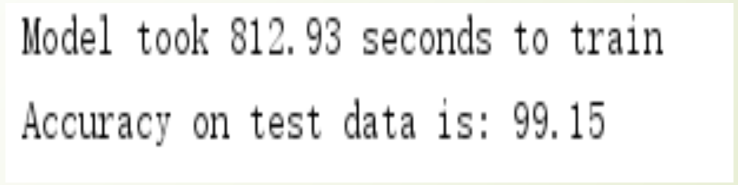
- Add drop out layer
- Dropout parameter = 0.1



```
Model took 821.38 seconds to train
Accuracy on test data is: 99.07
```



- Add drop out layer
- Dropout parameter = 0.2



```
Model took 812.93 seconds to train
Accuracy on test data is: 99.15
```

Comparison

- **Accuracy** : the CNN has reached the accuracy of 0.9915 and the Neural Network reached 0.967. Although we can always adjust some of the parameters to optimize the result. But the result clearly shows that the CNN is better than Neural Network in handwriting digit recognition problem.
- **Training Time**: the Neural Network takes about 150s to train, but the CNN takes 230 seconds if using 5*5 filter. It may seem like neural network method trains faster than CNN. But theoretically the CNN should be faster than neural network. We consider this as a lack of input dimensions. Here we only take 28x28 as the input diameters. If we add one more dimension of the picture---color. The result should be CNN faster than neural network.
- **Parameters/Weights** : they both use quite different parameters, due to their characteristics. But it clearly shows that the CNN has less parameters than Neural Network. And quite easy to understand what the parameter is and easy to adjust.

Conclusions and Improvements

- We can use deep learning method, introducing much deeper (and larger) networks to be trained. Like adding more hidden layers.
- If time is curtail to a training process. Less layers or even less inputs can be applied to learning.
- Parameters can be tuned to improve the training. And Epochs can be applied more times to ensure the accuracy.

- Highest accuracy of 0.9915 has been achieved.