# MNIST dataset recognition with CNN

## 1 Explanation of convolution and pooling

### 1.1 Convolution

In mathematical form, convolution is an operation of two functions of a real-valued argument. We use convolution to average together several measurement. It is a weight average and the weights can be shared among convolution operations. In a convolutional layer, we have two arguments, as input and kernel. The output is sometimes referred to as the feature map.

In the first convolutional layer of our example, we have input "x_image" which has 28*28 features. Then we use 32 convolutional kernels the size of each is 7*7. Because the padding is "SAME" which means we use zero-padding, we get 32 groups of 28*28 feature maps. (If the padding is "VALID", the size of feature map will be 22*22.) The number of trainable parameters is 32*(7*7+1) = 1600 (include bias.)

In the second layer, we use 5*5 kernels and the number of trainable parameters is 64*(5*5+1) = 1664.

### 1.2 Pooling

Although we use convolution operation to average measurement and reduce the number of connections, the number of neurons is so large that may lead to over-fitting. Therefore, we add an operation to reduce the size of feature maps, called pooling or subsampling. In addition, pooling dilate the receptive field of neurons in next layer and keep invariant to small changes.

In the first pooling operation of our example, we use the function "max_pool_2*2", which means we divide feature maps into many parts, the size of each is 2*2. Then we sample the maximum of each part and get new feature maps the size of each is 14*14. The number of trainable parameters is 32*(1+1) = 64.

In the second pooling operation, the number of trainable parameters is 64*(1+1) = 128.

# 2 Results

## 2.1 Screenshot of running results

Run with learning rate equals 1e-4, keep_pro_rate equals 0.7 and max_epoch equals 2000.

```
Extracting MNIST_data\train-images-idx3-ubyte.gz
Extracting MNIST_data\train-labels-idx1-ubyte.gz
Extracting MNIST_data\t10k-images-idx3-ubyte.gz
Extracting MNIST_data\t10k-labels-idx1-ubyte.gz
0.12
0.871
0.914
0.932
0.948
0.959
0.962
0.967
0.97
0.972
0.97
0.975
0.976
0.982
0.98
0.981
0.977
0.98
0.984
0.984
```

The final accuracy is 98.4%.

## 2.2 Explanation and analysis

The accuracy is significantly higher than the results from full connection neural net which is about 90%, because the convolutional operation build a larger receptive field which means machine can learn more abstract features. Therefore, CNN is good at image recognition missions. In this example, there are too many neurons in full connection layer, so we use drop-out to avoid over-fitting.

By enlarging the max epoch, the accuracy can reach 99.0%. I also tried other activation functions (sigmoid and tanh), and finally found the ReLU function can lead to higher accuracy. Change the keep_pro_rate and learning rate did not obviously increase accuracy.

Therefore, to improve accuracy, we may need a deeper network and larger training dataset. Also, we can use some methods like data augmentation to get larger dataset from definite data.