MNIST 手写数字集识别

分了两个模型：

① Softmax Model

② CNN Mmodel

使用框架：Keras

1 mnist 数据数据集获取

方式一：使用 tf.contrib,.learn 模块加载 mnist 数据集（弃用），如下

```
#使用 tf.contrib.learn 模块加载 MNIST 数据集(Deprecated 弃用)
import tensorflow as tf
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets('./mnist/dataset/') 这种方法官方已经遗弃了
```

运行之后会出现 Warning 提示，该方式已经不推荐使用

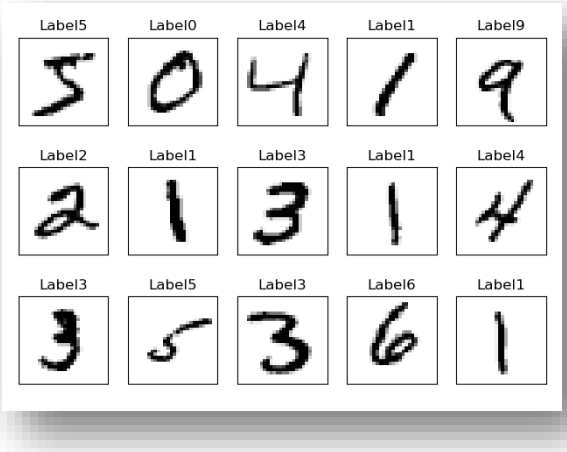
方式二：使用 keras.dataset 模块加载 mnist 数据集，如下

```
from keras.datasets import mnist
(x_train, y_train), (x_test,y_test) = mnist.load_data('mnist/mnist.npz')
##这里是相对路径，其实绝对路径在这里哦 C:\Users\korey\.keras\datasets\mnist\mnist.npz
```

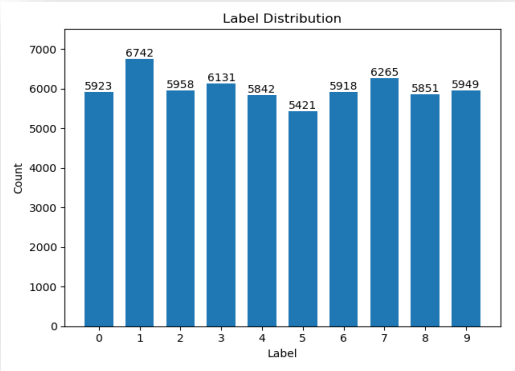
这种方式要比第一种简单很多，但是要注意这里的 Path 是个相对路径。

训练集：60000；测试集：10000

可视化数据集（15 个）



训练集数据分布



观察到数据分布均匀，下面进行 Model 搭建

2 Softmax Model

2.1 数据处理

Softmax 网络的数据输入是对像素点的输入，图片为 28*28，那么展开就有 784 个像素点，代码说明如下

```
X_train = x_train.reshape(60000, 784) # （60000,784）
X_test = x_test.reshape(10000, 784)#（10000,784）
#将数据类型装换为 float32,如果不怎么做的话，后面的归一化操作得到的只有 0 和 1 两个数
X_train = X_train.astype('float32')
X_test = X_test.astype('float32')
#数据归一化
X_train /= 255
X_test /= 255
```

2.2 one-hot 编码

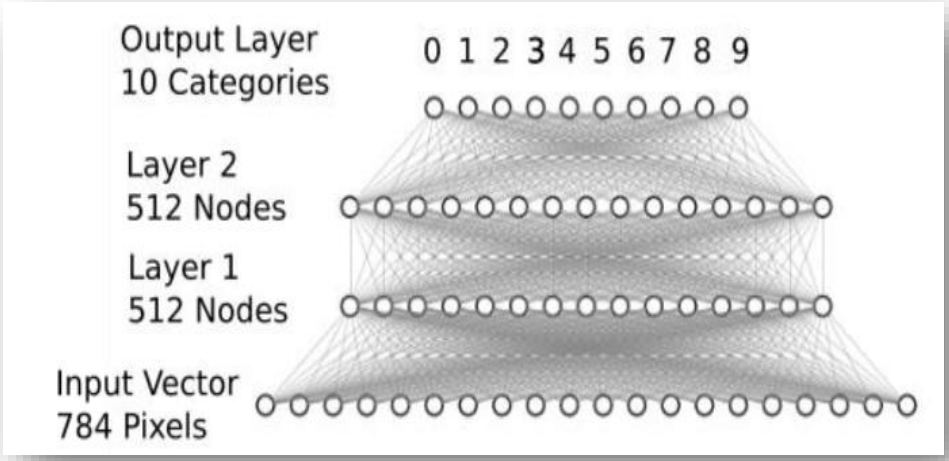
对于 softmax 网络的输出是每个类对应的概率，然后最大概率则是预测的类别，所以这里需要对标签进行 one-hot 编码，对于高层的 Keras 框架来说，one-hot 编码已经被封装好了，如下：

```
#one-hot
from keras.utils import np_utils
n_classes =10
Y_train = np_utils.to_categorical(y_train, n_classes) ## 实现 one-hot 编码
Y_test = np_utils.to_categorical(y_test, n_classes)
```

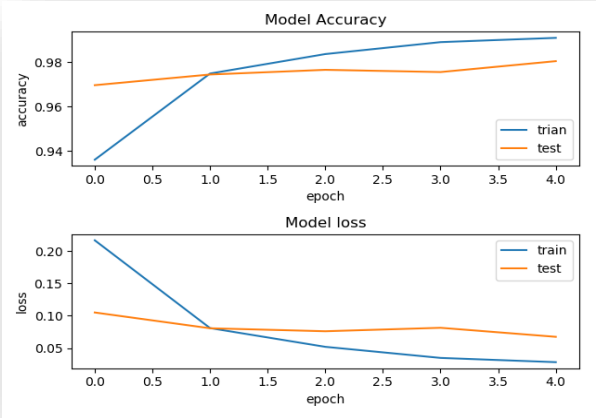
举个例子：原始标签如果是 5，那么对应的 one-hot 编码后应该是[0., 0., 0., 0., 1., 0., 0., 0., 0., 0.]。

2.3 Keras Softmax Model

利用 Keras 的 Sequential 模型，可以对框架进行快速搭建，如下：



可视化结果如下:



2.4 模型保存和加载

采用 model.save()进行模型保存，利用 load_model 模块进行加载，详细见代码;

3 CNN Model

3.1 数据处理

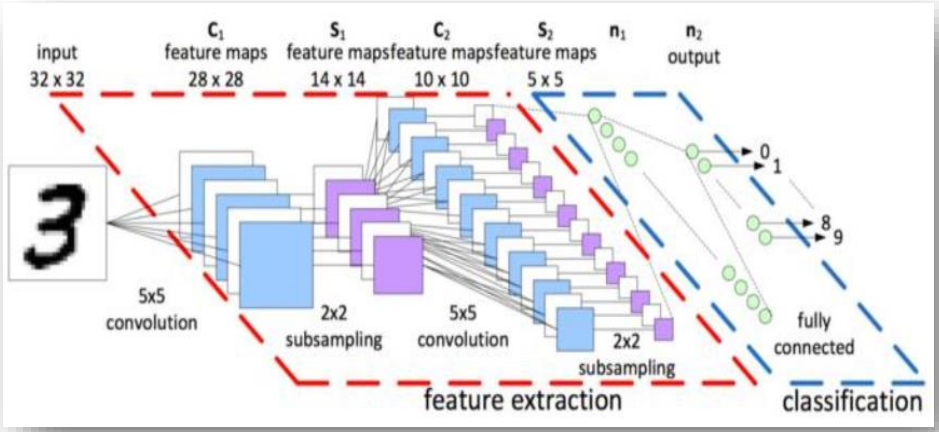
这里注意一下 channel_last(默认)和 channel_first 的区别

```
"""数据规范化"""
from keras import backend as K
#在 C:\Users\korey\.keras\keras.json 配置文件中查看默认的数据 shape 是 channels_last
img_rows, img_cols = 28, 28
if K.image_data_format() == 'channels_first': #[batch, channel, height, width]
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
```

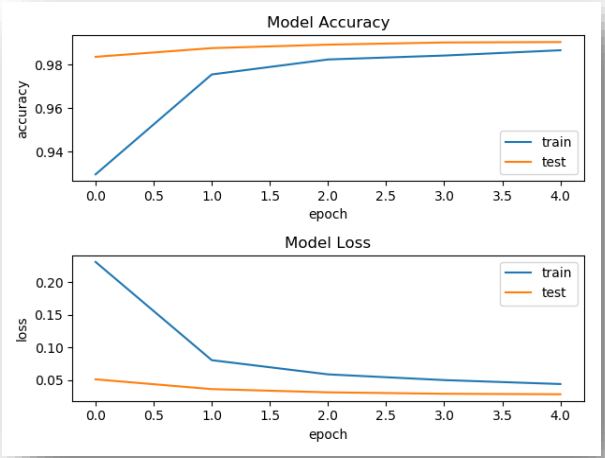
```
else:# channel_last  [batch, height, width, cbannel]
    x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)

# 数据类型转换 float32
X_train = x_train.astype('float32')
X_test = x_test.astype('float32')
# 数据归一化
X_train /= 255
X_test /= 255
```

3.2 one-hot 编码（同上 softmax 网络）

3.3 Keras CNN Model



结果可视化如下：（效果要比单纯的 softmax 网络要好）



3.4 模型保存和加载

利用 model.save()和 load_model 模块。