





- **常用的回归**：线性、决策树、SVM、KNN；集成回归：随机森林、Adaboost、GradientBoosting、Bagging、ExtraTrees
- **常用的分类**：线性、决策树、SVM、KNN，朴素贝叶斯；集成分类：随机森林、Adaboost、GradientBoosting、Bagging、ExtraTrees
- **常用聚类**：k均值（K-means）、层次聚类（Hierarchical clustering）、DBSCAN
- **常用降维**：LinearDiscriminantAnalysis、PCA

实现代码：

```
clf = svm.SVC()
clf.fit(train_images, train_labels.values.ravel())
clf.score(test_images, test_labels)
```

结果：

2027	new	yuemin		0.93700	3	4d
2028	new	Weijia.wu		0.93700	1	2d
Your Best Entry ↑						
Your submission scored 0.93700					Tweet this!	
2029	new	sanketn		0.93657	1	1mo

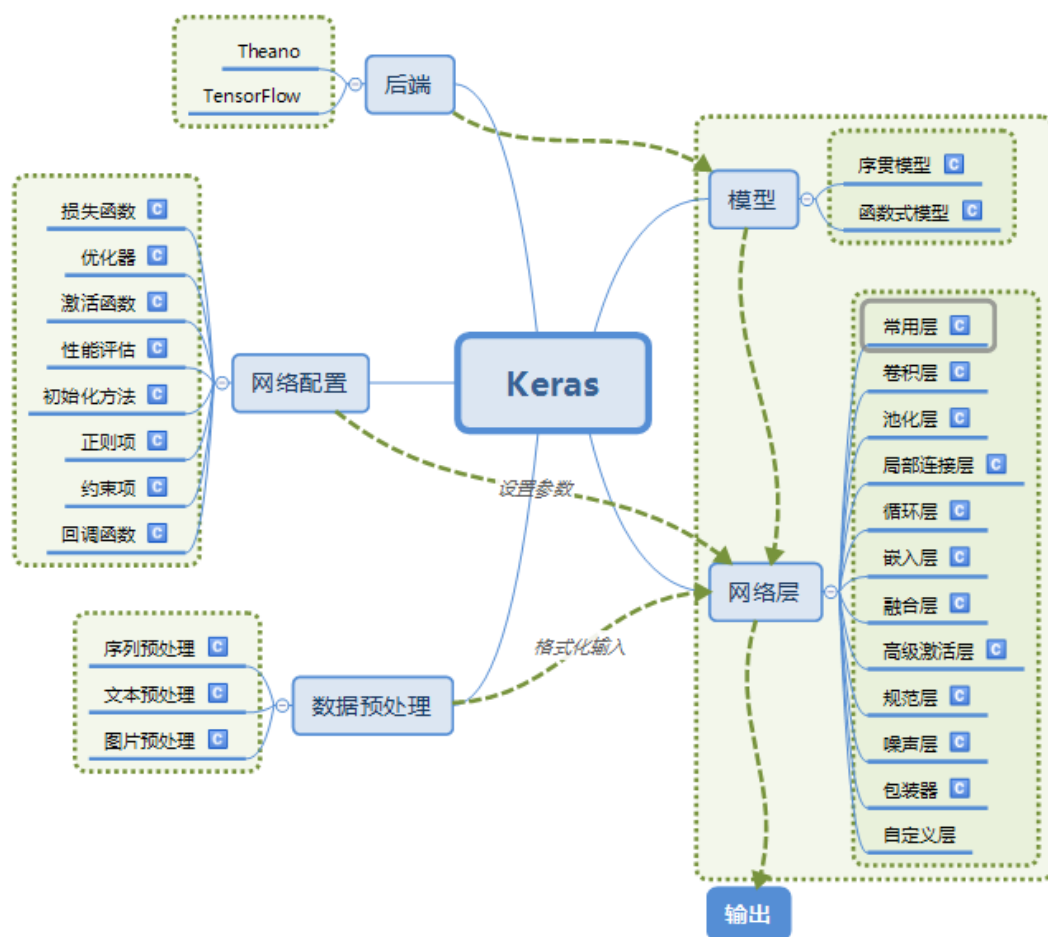
Deep neural network the Keras way

1、pandas(Python Data Analysis Library):

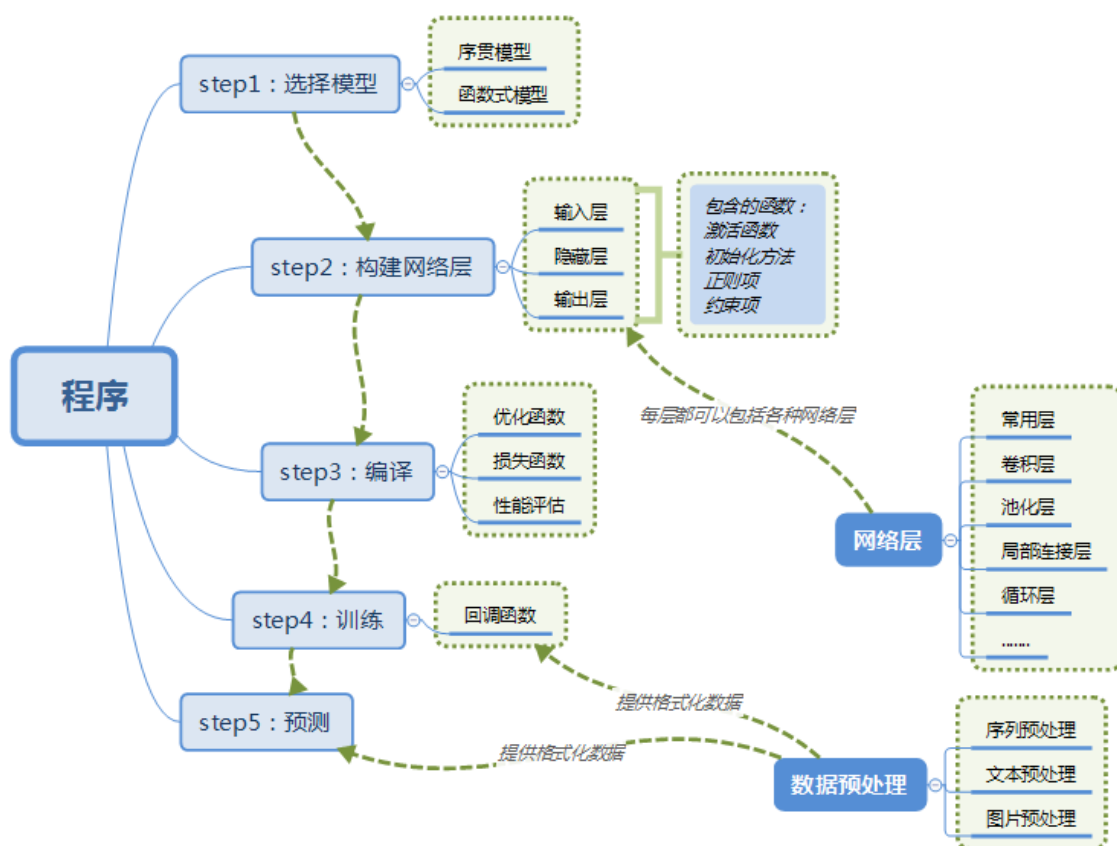
pandas 是基于NumPy 的一种工具，该工具是为了解决数据分析任务而创建的。

2、Keras

Keras的模块结构：



使用Keras搭建一个神经网络:



Keras有两种类型的模型，序贯模型（Sequential）和函数式模型（Model），函数式模型应用更为广泛，序贯模型是函数式模型的一种特殊情况。

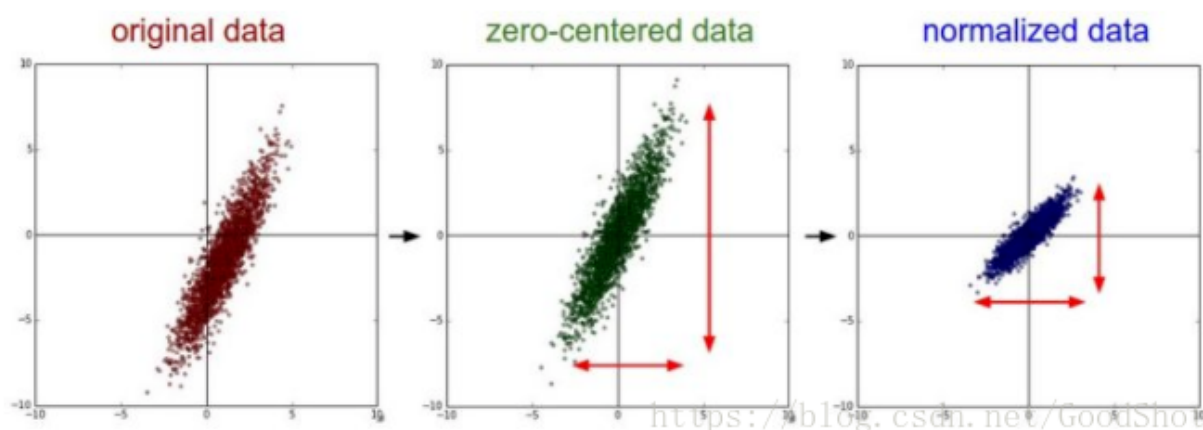
a) **序贯模型 (Sequential)**:单输入单输出，一条路通到底，层与层之间只有相邻关系，没有跨层连接。这种模型编译速度快，操作也比较简单

b) **函数式模型 (Model)**：多输入多输出，层与层之间任意连接。这种模型编译速度慢。

3、规范化数据

通过标准化处理，可以使得不同的特征具有相同的尺度 (Scale)。简言之，当原始数据不同维度上的特征的尺度（单位）不一致时，需要标准化步骤对数据进行预处理。

左图 表示的是原始数据；中间 的是中心化后的数据，数据被移动大原点周围；右图 将中心化后的数据除以标准差，得到为标准化的数据，可以看出每个维度上的尺度是一致的（红色线段的长度表示尺度）。



当使用梯度下降法寻求最优解时，很有可能走“之字型”路线（垂直等高线走），从而导致需要迭代很多次才能收敛；而右图对两个原始特征进行了归一化，其对应的等高线显得很圆，在梯度下降进行求解时能较快的收敛。因此如果机器学习模型使用梯度下降法求最优解时，归一化往往非常有必要，否则很难收敛甚至不能收敛。

以下是两种常用的归一化方法：

1) min-max标准化 (Min-MaxNormalization)

$$x^* = \frac{x - \min}{\max - \min}$$

2) Z-score标准化 (0-1标准化) 方法

给予原始数据的均值 (mean) 和标准差 (standard deviation) 进行数据的标准化。经过处理的数据符合标准正态分布，即均值为0，标准差为1。

$$x^* = \frac{x - \mu}{\sigma}$$

4、Designing Neural Network Architecture

Flatten层:

Flatten层用来将输入“压平”，即把多维的输入一维化，常用在从卷积层到全连接层的过渡。Flatten不影响batch的大小。

```
*model = Sequential()
model.add(Convolution2D(64, 3, 3,
                        border_mode='same',
                        input_shape=(3, 32, 32)))
# now: model.output_shape == (None, 64, 32, 32)
model.add(Flatten())
# now: model.output_shape == (None, 65536)*
```

Dense层:

Dense就是常用的全连接层，所实现的运算是 $\text{output} = \text{activation}(\text{dot}(\text{input}, \text{kernel}) + \text{bias})$ 。其中activation是逐元素计算的激活函数，kernel是本层的权值矩阵，bias为偏置向量，只有当use_bias=True才会添加。

如果本层的输入数据的维度大于2，则会被先压为与kernel相匹配的大小。

```
# as first layer in a sequential model:
# as first layer in a sequential model:
model = Sequential()
model.add(Dense(32, input_shape=(16,)))
# now the model will take as input arrays of shape (*, 16)
# and output arrays of shape (*, 32)

# after the first layer, you don't need to specify
# the size of the input anymore:
model.add(Dense(32))
```

5、整个设计思路以及分析

1、Load Train and Test data

```
# create the training & test sets, skipping the header row with [1:]
train =
pd.read_csv("/home/weijia.wu/workspace/Kaggle/lib_data/train.csv")
X_train = train.iloc[:,1:].values.astype('float32') #取出所有的数据
y_train = train.iloc[:,0].values.astype('int32') # 取出所有的标签
```

2、Preprocessing the digit images

```
mean_px = X_train.mean().astype(np.float32) #均值
std_px = X_train.std().astype(np.float32) #标准差
```

```
def standardize(x):  
    return (x-mean_px)/std_px      #均零标准化
```

3、One Hot encoding of labels

```
from keras.utils.np_utils import to_categorical  
y_train= to_categorical(y_train)  
num_classes = y_train.shape[1]
```

One Hot encoding的意义与作用:

One-Hot编码，又称为一位有效编码，主要是采用位状态寄存器来对个状态进行编码，每个状态都由他独立的寄存器位，并且在任意时候只有一位有效。

1.使用one-hot编码，将离散特征的取值扩展到了欧式空间，离散特征的某个取值就对应欧式空间的某个点。

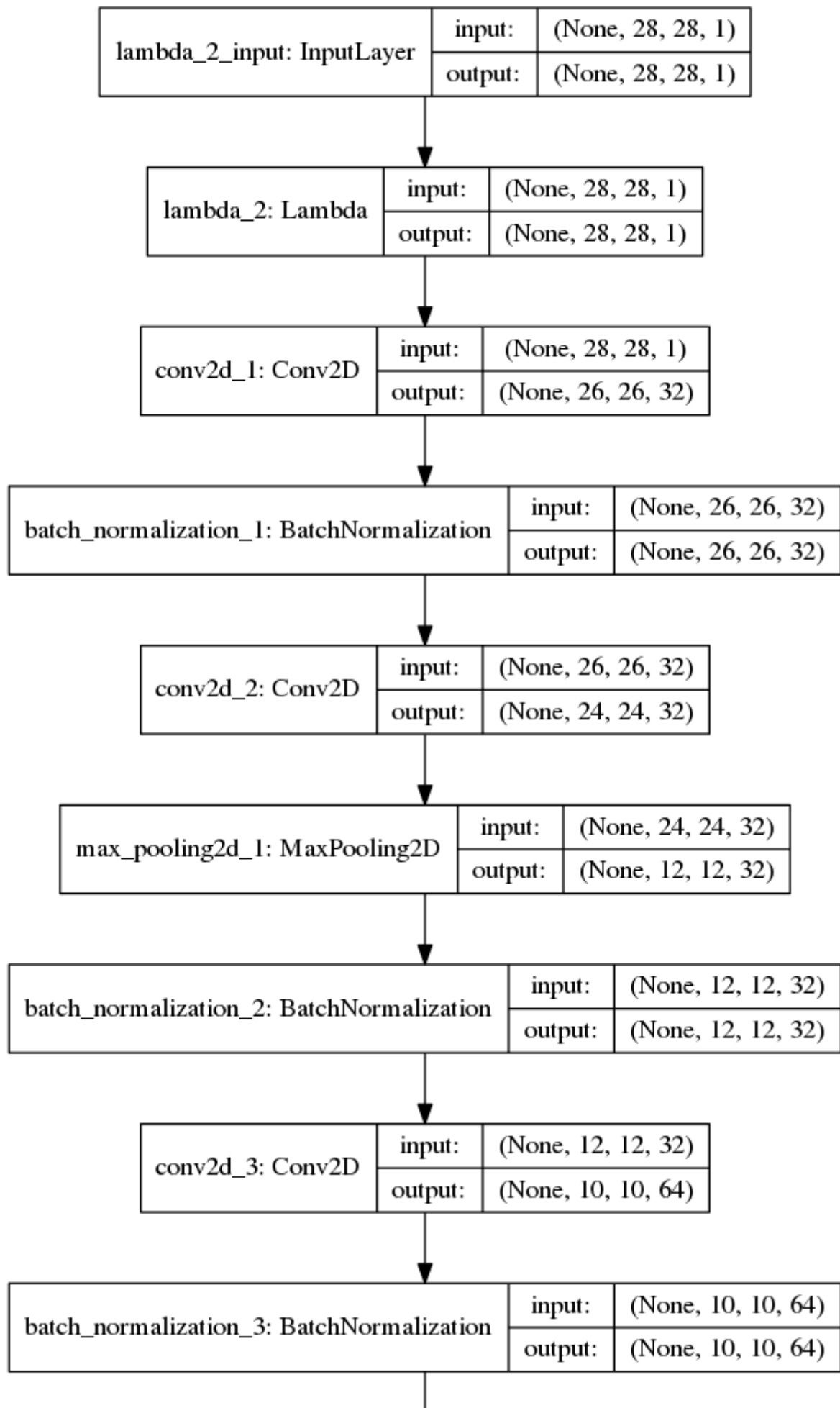
2.将离散特征通过one-hot编码映射到欧式空间，是因为，在回归，分类，聚类等机器学习算法中，特征之间距离的计算或相似度的计算是非常重要的，而我们常用的距离或相似度的计算都是在欧式空间的相似度计算，计算余弦相似性，基于的就是欧式空间。

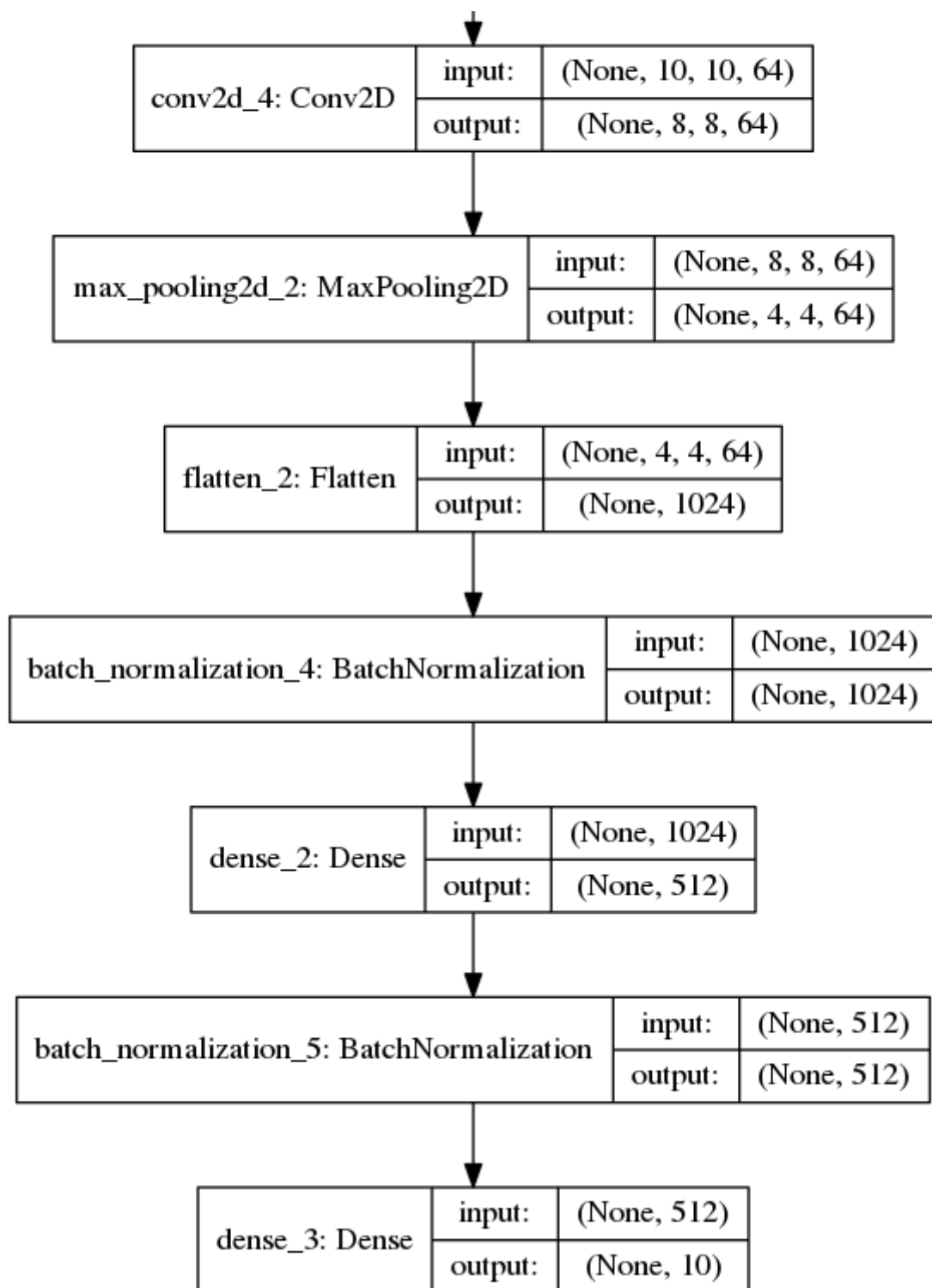
3.将离散型特征使用one-hot编码，确实会让特征之间的距离计算更加合理。

4、Convolutional Neural Network

```
from keras.layers.normalization import BatchNormalization  
  
def get_bn_model():  
    model = Sequential([  
        Lambda(standardize, input_shape=(28,28,1)),    ##输入  
        Convolution2D(32,(3,3), activation='relu'),    ##卷积层  
        BatchNormalization(axis=1),                    ##批量标准化  
        Convolution2D(32,(3,3), activation='relu'),  
        MaxPooling2D(),                                  ##池化  
        BatchNormalization(axis=1),  
        Convolution2D(64,(3,3), activation='relu'),  
        BatchNormalization(axis=1),  
        Convolution2D(64,(3,3), activation='relu'),  
        MaxPooling2D(),  
        Flatten(),  
        BatchNormalization(),  
        Dense(512, activation='relu'),                  ##全连接  
        BatchNormalization(),  
        Dense(10, activation='softmax')  
    ])  
    model.compile(Adam(), loss='categorical_crossentropy', metrics=['accuracy'])  
    return model
```

模型拓扑:





Lambda层：本函数用以对上一层的输出施以任何Theano/TensorFlow表达式

Conv2D层：二维卷积层，即对图像的空域卷积。该层对二维输入进行滑动窗卷积，当使用该层作为第一层时，应提供input_shape参数。

```

keras.layers.convolutional.Conv2D(filters, kernel_size, strides=(1, 1), padding='valid', data_format=None, dilation_rate=(1, 1), activation=None, use_bias=True, kernel_initializer='glorot_uniform', bias_initializer='zeros', kernel_regularizer=None, bias_regularizer=None, activity_regularizer=None, kernel_constraint=None, bias_constraint=None)

```


- **filters**: 卷积核的数目（即输出的维度）
- **kernel_size**: 单个整数或由两个整数构成的list/tuple，卷积核的宽度和长度。如为单个整数，则表示在各个空间维度的相同长-度。

BatchNormalization层：该层在每个batch上将前一层的激活值重新规范化，即使得其输出数据的均值接近0，其标准差接近1

- **axis**: **整数**，指定要规范化的轴，通常为特征轴。例如在进行data_format="channels_first"的2D卷积后，一般会设axis=1。

MaxPooling2D层：为空域信号施加最大值池化。

5、模型可视化

首先先安装Keras中神经网络可视化模块keras.utils.visualize_util:

- 1、pip install graphviz
- 2、pip install pydot
- 3、pip install pydot_ng

然后输入：

```
# 使用下面的简单方法，在model.compile()之后调用，即可绘图
from keras.utils import plot_model
plot_model(semantic_model, to_file='model.png', show_shapes='True')
```

反馈与建议

- 微博：[@柏林designer](#)
- 邮箱：wwj123@zju.edu.cn