**《深度学习-keras模型的保存和加载》**

在上一篇关于深度学习的推文中，我们介绍了使用Keras来实现机器学习中的线性回归和非线性回归算法。不过在那期的推文当中我们没有介绍如何保存我们的训练的模型和加载我们训练的模型，所以在本次的推文中我们将向大家介绍一下Keras中如何进行模型的保存、加载和接着训练。

**一．Keras模型的保存**

我们将使用手写字体识别的案例来进行说明。

我们先看一下之前的代码：

from keras.datasets import mnist  
from keras.utils import np\_utils  
from keras.models import Sequential  
from keras.layers import Dense,Activation  
path = **r'F:\kerasdataset\mnist.npz'**(X\_train, y\_train), (X\_test, y\_test) = mnist.load\_data(path)  
X\_train = X\_train.reshape(len(X\_train),-1)  
X\_test = X\_test.reshape(len(X\_test), -1)  
X\_train = X\_train.astype(**'float32'**)/255  
X\_test = X\_test.astype(**'float32'**)/255  
y\_train = np\_utils.to\_categorical(y\_train)  
y\_test = np\_utils.to\_categorical(y\_test)  
model = Sequential()  
model.add(Dense(512, input\_shape=(28\*28,),activation=**'relu'**))  
model.add(Dense(10,activation=**'softmax'**))  
model.compile(optimizer=**'adam'**, loss=**'categorical\_crossentropy'**, metrics=[**'accuracy'**])  
model.fit(X\_train, y\_train, epochs=10, batch\_size=64, verbose=1, validation\_split=0.05)  
loss, accuracy = model.evaluate(X\_test, y\_test)  
Testloss, Testaccuracy = model.evaluate(X\_test, y\_test)  
print(**'Testloss:'**, Testloss)  
print(**'Testaccuracy:'**, Testaccuracy)

在Keras中，我们常用到的保存模型的方式有四种：

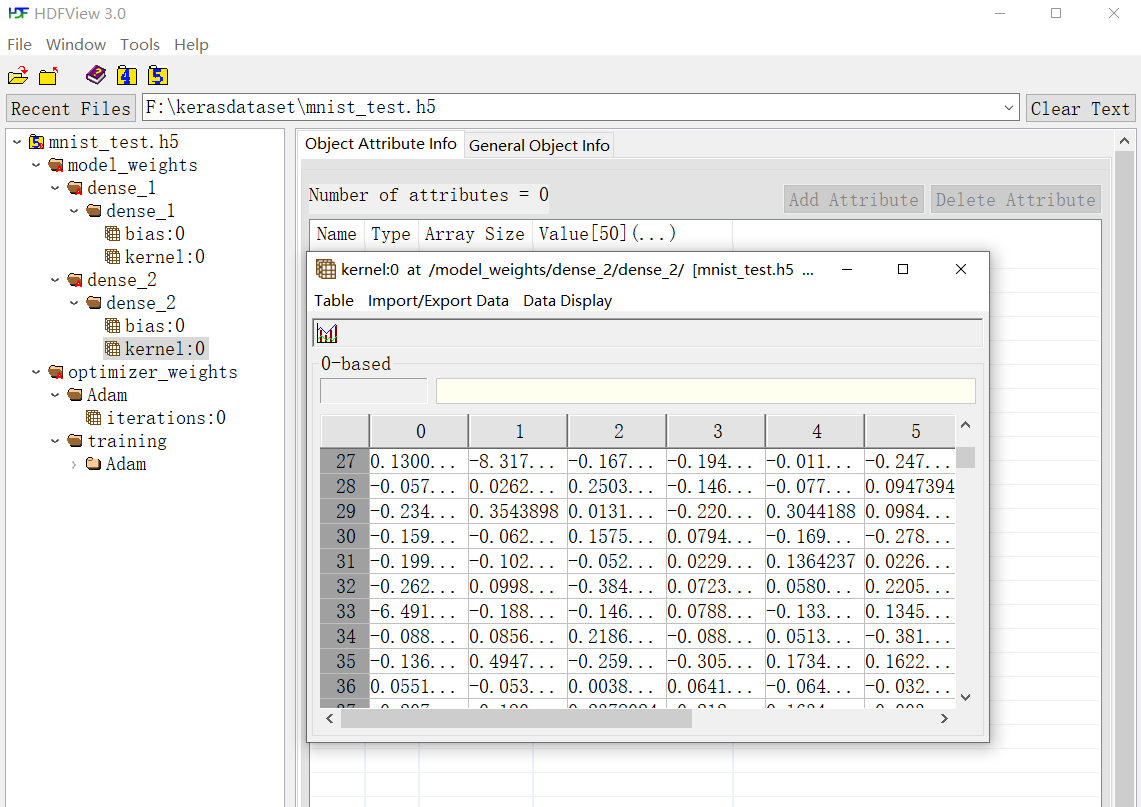
model.save()   
model.save\_weights()  
model.to\_json()  
model.to\_yaml()

* 1. **model.save()**

这种方法是将Keras模型和权重保存在一个HDF5文件中，具体的方法：

save\_path = **r'F:\kerasdataset\mnist\_test.h5'**model.save(save\_path)

我们只要在手写字体识别的代码最后加上上述代码即可。在上述的两行代码中我们设置了“mnist\_test.h5”文件的保存路径，并且使用model.save()进行模型的保存。我们可以使用HDFView软件来打开这个保存的文件，如下所示。我们可以使用这个软件来查看模型中的dense和相应的权重信息：



当然了，除了这种方法以外，我们也可以使用代码来进行读取：

import h5py  
*# 模型地址*MODEL\_PATH = **r'F:\kerasdataset\mnist\_test.h5'***# 获取每一层的连接权重及偏重*print(**"读取模型中..."**)  
with h5py.File(MODEL\_PATH, **'r'**) as f:  
 dense\_1 = f[**'/model\_weights/dense\_1/dense\_1'**]  
 dense\_1\_bias = dense\_1[**'bias:0'**][:]  
 dense\_1\_kernel = dense\_1[**'kernel:0'**][:]

dense\_2 = f[**'/model\_weights/dense\_2/dense\_2'**]  
 dense\_2\_bias = dense\_2[**'bias:0'**][:]  
 dense\_2\_kernel = dense\_2[**'kernel:0'**][:]

print(**"第一层的连接权重矩阵：**\n**%s**\n**"**%dense\_1\_kernel)  
print(**"第一层的连接偏重矩阵：**\n**%s**\n**"**%dense\_1\_bias)  
print(**"第二层的连接权重矩阵：**\n**%s**\n**"**%dense\_2\_kernel)  
print(**"第二层的连接偏重矩阵：**\n**%s**\n**"**%dense\_2\_bias)

* 1. **model.save\_weights()**

这种也是获取网络权重的方法，其使用方法如下：

model\_save\_path =**r'F:\kerasdataset\mnist\_test.h5'**model.save\_weights(model\_save\_path, by\_name=True)

by\_name这个参数有两种情况，默认为False:

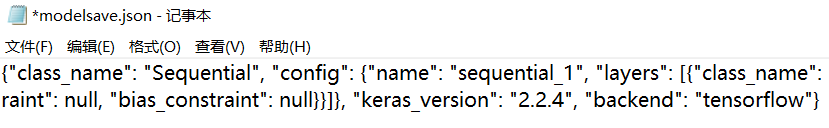
by\_name=False 的时候按照网络的拓扑结构来进行加载权重的。

by\_name=True 的时候就是按照网络层名称进行加载权重的。

* 1. **model.to\_json()**

这种方法只保存了模型结构，没有包含其权重信息，其使用方法如下：json\_save = model.to\_json()  
with open(**"modelsave.json"**, **"w"**) as f:  
 f.write(json\_save)

JSON中的字符内容如下所示：



* 1. **model.to\_yaml()**

这种方法和model.to\_json()一样，其用法如下：

yaml\_save = model.to\_yaml()  
with open(**"modelsave.yaml"**, **"w"**) as f:  
 f.write(yaml\_save)

上述方法中，最常用的还是model.save() 和model.save\_weights()两种方式。

**二．Keras模型的加载**

模型的加载很简单，我们对不同的保存方式进行不同的加载即可：

**2.1. model.save()方式模型的加载**

from keras.models import load\_model  
save\_path = **r'F:\kerasdataset\mnist\_test.h5'**model = load\_model(save\_path)

**2.2. model.save\_weights()方式模型的加载**

model\_save\_path =**r'F:\kerasdataset\mnist\_test.h5'**model = model.load\_weights(model\_save\_path)

**2.3. model.to\_json()和model.to\_yaml()方式模型的加载**

from keras.models import model\_from\_json  
from keras.models import model\_from\_yaml  
model\_json = model\_from\_json(json\_save)  
model\_yaml = model\_from\_yaml(yaml\_save)

我们来总结一下他们的区别：



**三．Keras模型的继续训练**

在实际的项目中，如果一个模型的训练时间很长(或者某种情况先要暂停)，我们可以训练到某一个周期以后暂停模型的训练，等到需要的时候接着训练。那么这样的方案该怎么解决：

假设我们在模型的训练结尾使用model.save(r'F:\kerasdataset\mnist\_test.h5')

保存模型的操作。

接下来我们来看怎么继续加载这个模型进行训练，我们只需load后进行fit:

*# 载入模型*model = load\_model(**r'F:\kerasdataset\mnist\_test.h5'**)  
*# 继续训练模型2个epochs*model.fit(X\_train, y\_train, batch\_size=64, epochs=2)

以上就是模型的加载和接着训练，下面如果要进行模型的评价和预测的话，可以参照手写字体识别的代码。

图片备份：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **保存方式** | **是否保存**  **模型结构** | **是否保存**  **模型权重** | **是否能继续**  **训练网络** | **是否能进行**  **模型预测** |
| model.save() | 是 | 是 | 是 | 是 |
| model.save\_weights() | 否 | 是 | 否 | 是 |
| model.to\_json() | 是 | 否 | 否 | 加载权重后能进行正常预测 |
| model.to\_yaml() | 是 | 否 | 否 | 加载权重后能进行正常预测 |