DL4CVhw03

1、运行本课提供的ipny文件，截图结果；

In [1]:

```
"""
基础_模型迁移_CBIR_augmentation
by jsxyhelu
注意，本模型最好不要在GPU下执行
"""
import numpy as np
import cv2
import os
import math
import h5py as h5py

#!apt-get -qq install -y graphviz && pip install -q pydot
import pydot
import matplotlib.pyplot as plt
from keras.utils.vis_utils import plot_model
from IPython.display import Image

from keras.utils.data_utils import get_file
from keras.models import Sequential, Model
from keras.layers import Input, Dense, Dropout, Flatten
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.applications.vgg16 import VGG16
from keras.optimizers import SGD

from keras.preprocessing.image import ImageDataGenerator
from keras.preprocessing import image
```

```
/home/ian/installed/anaconda3/lib/python3.6/site-packages/h5py/__init
__.py:36: FutureWarning: Conversion of the second argument of issubdt
ype from `float` to `np.floating` is deprecated. In future, it will b
e treated as `np.float64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
```

运行下面的代码时报错：AttributeError: 'Node' object has no attribute 'output_masks'

解决方法：pip install --upgrade keras_applications

In [2]:

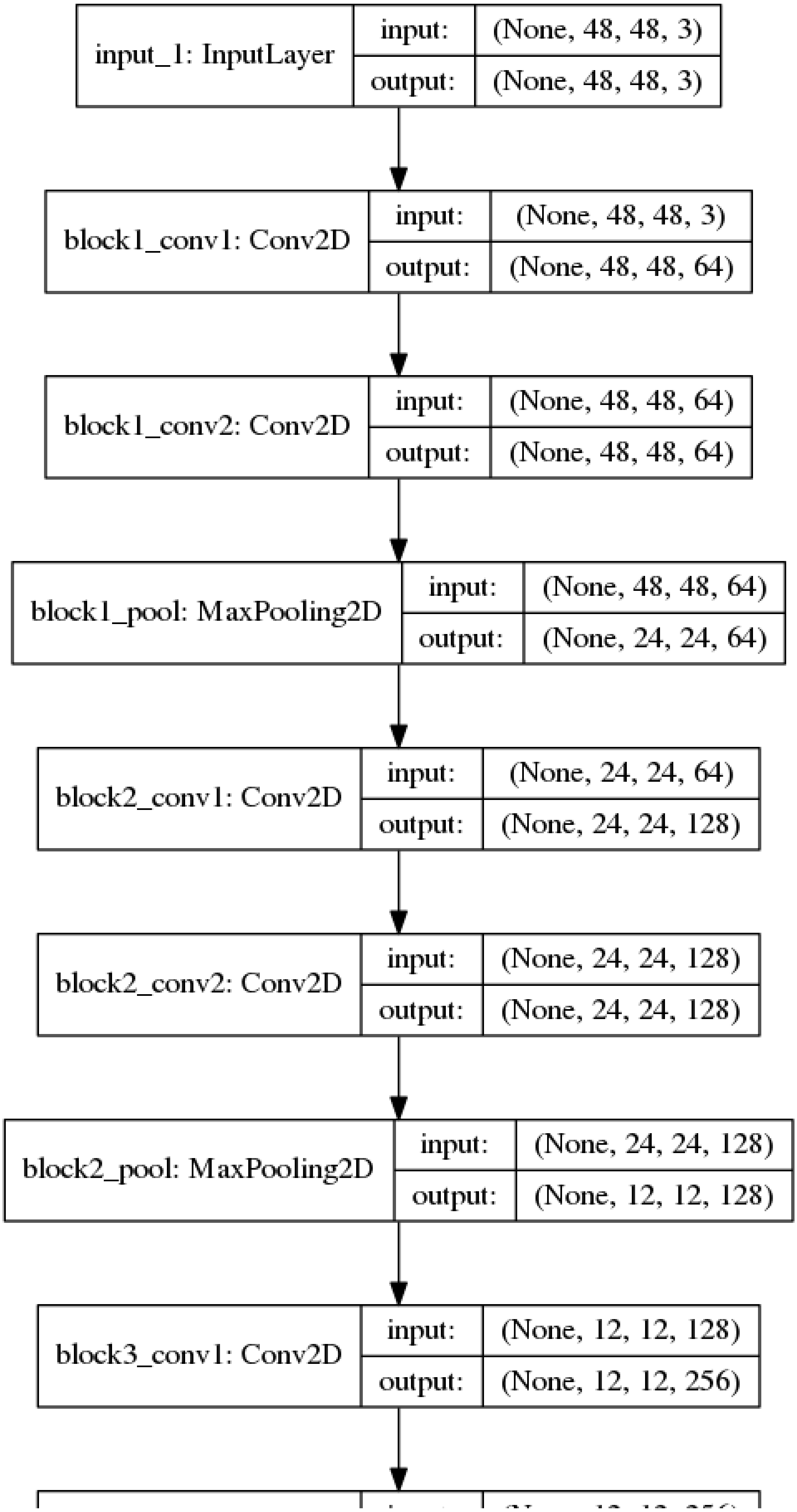
```
#训练集和验证集比率
RATIO = 0.2
#根据分类总数确定one-hot总类
NUM_DENSE = 5
#训练总数。经过实验发现epoch = 7 的时候，模型收敛，准确率98
#TODOepochs = 7
epochs = 1
#默认图片大小:48*48
ishape=48
#one hot TODO给改掉
def tran_y(y):
    y_ohe = np.zeros(NUM_DENSE)
    y_ohe[y] = 1
    return y_ohe

print('导入vgg模型')
#导入vgg模型
model_vgg = VGG16(include_top = False, weights = 'imagenet', input_shape = (ishape,
#将fc层失活，并且重新迁移训练
for layer in model_vgg.layers:
        layer.trainable = False
model = Flatten()(model_vgg.output)
model = Dense(4096, activation='relu', name='fc1')(model)
model = Dense(4096, activation='relu', name='fc2')(model)
model = Dropout(0.5)(model)
model = Dense(NUM_DENSE, activation = 'softmax', name='prediction')(model)
model_vgg_pretrain = Model(model_vgg.input, model, name = 'vgg16_pretrain')

print('模型编译')
#模型编译
sgd = SGD(lr = 0.05, decay = 1e-5)
model_vgg_pretrain.compile(loss = 'categorical_crossentropy', optimizer = sgd, metr
```

```
导入vgg模型
模型编译
```

In [3]:

```
#模型结构打印
plot_model(model_vgg_pretrain, to_file="model.png", show_shapes=True)
Image('model.png')
```

Out[3]:



input_1: InputLayer | input: (None, 48, 48, 3) | output: (None, 48, 48, 3)

block1_conv1: Conv2D | input: (None, 48, 48, 3) | output: (None, 48, 48, 64)

block1_conv2: Conv2D | input: (None, 48, 48, 64) | output: (None, 48, 48, 64)

block1_pool: MaxPooling2D | input: (None, 48, 48, 64) | output: (None, 24, 24, 64)

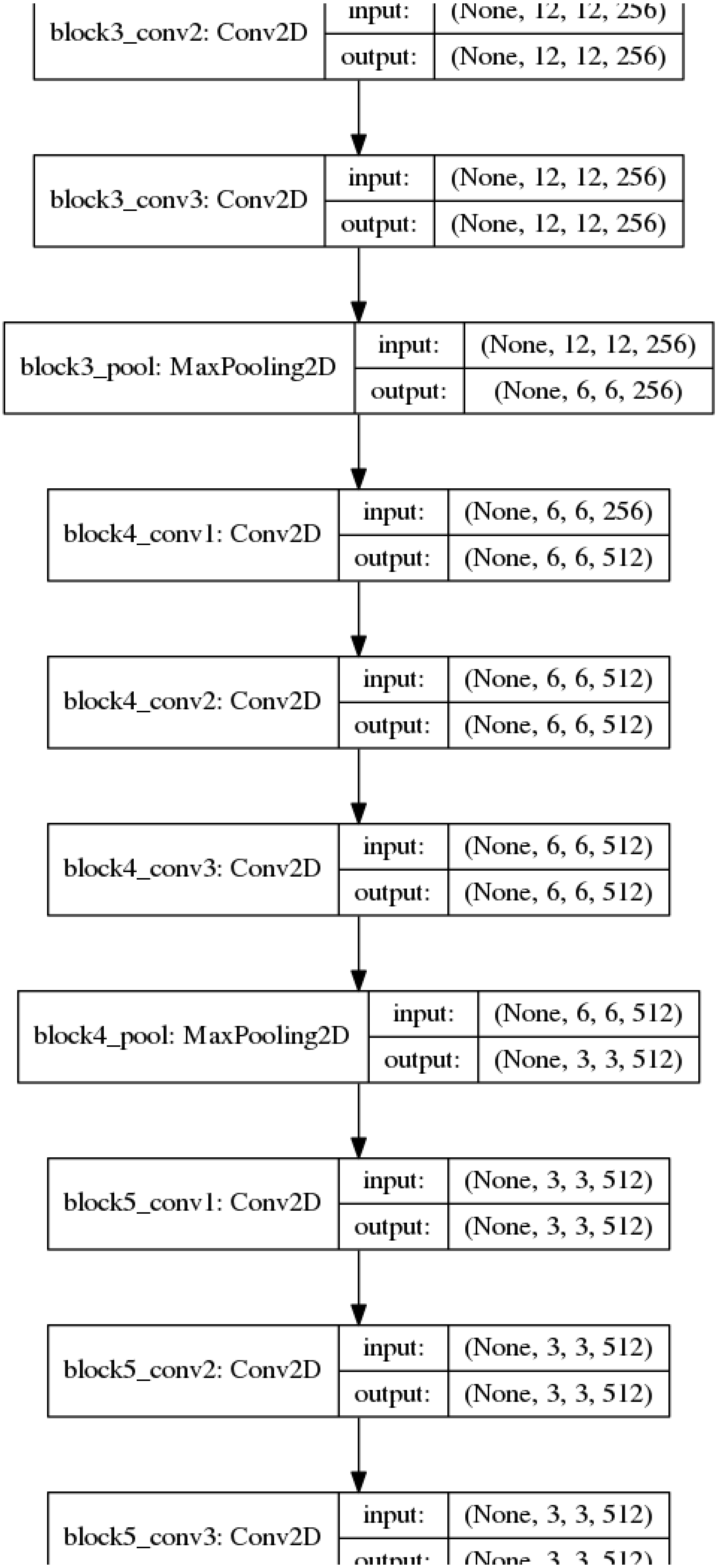
block2_conv1: Conv2D | input: (None, 24, 24, 64) | output: (None, 24, 24, 128)

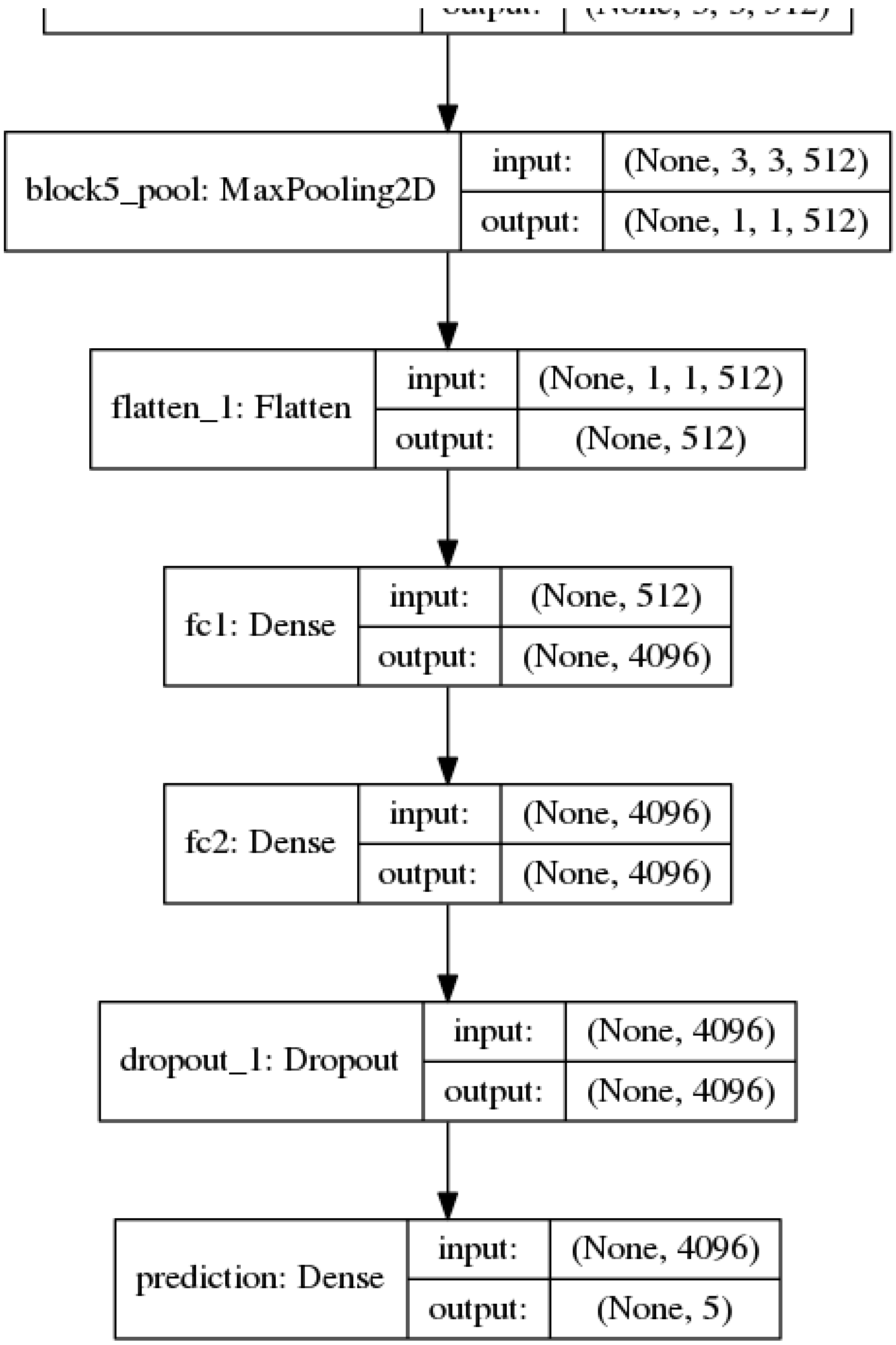
block2_conv2: Conv2D | input: (None, 24, 24, 128) | output: (None, 24, 24, 128)

block2_pool: MaxPooling2D | input: (None, 24, 24, 128) | output: (None, 12, 12, 128)

block3_conv1: Conv2D | input: (None, 12, 12, 128) | output: (None, 12, 12, 256)

input: (None, 12, 12, 256)

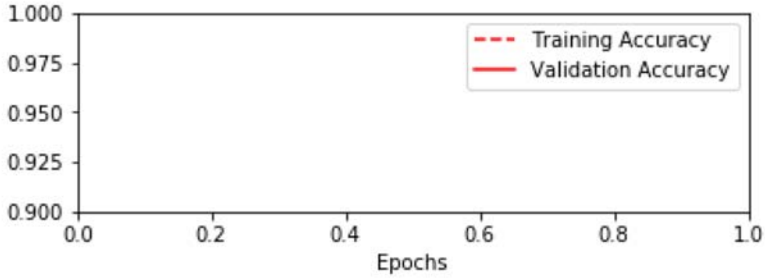

block3_conv2: Conv2D
input: (None, 12, 12, 256)
output: (None, 12, 12, 256)
block3_conv3: Conv2D
input: (None, 12, 12, 256)
output: (None, 12, 12, 256)
block3_pool: MaxPooling2D
input: (None, 12, 12, 256)
output: (None, 6, 6, 256)
block4_conv1: Conv2D
input: (None, 6, 6, 256)
output: (None, 6, 6, 512)
block4_conv2: Conv2D
input: (None, 6, 6, 512)
output: (None, 6, 6, 512)
block4_conv3: Conv2D
input: (None, 6, 6, 512)
output: (None, 6, 6, 512)
block4_pool: MaxPooling2D
input: (None, 6, 6, 512)
output: (None, 3, 3, 512)
block5_conv1: Conv2D
input: (None, 3, 3, 512)
output: (None, 3, 3, 512)
block5_conv2: Conv2D
input: (None, 3, 3, 512)
output: (None, 3, 3, 512)
block5_conv3: Conv2D
input: (None, 3, 3, 512)
output: (None, 3, 3, 512)

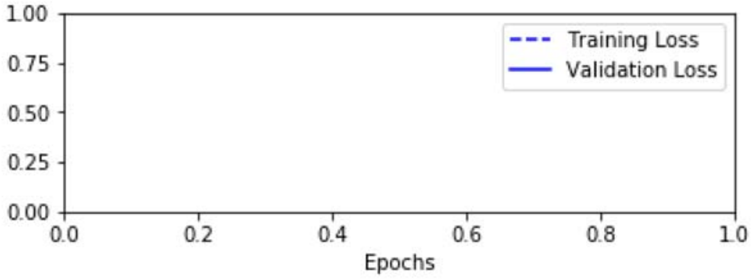

output:
(None, 3, 3, 512)
block5_pool: MaxPooling2D
input:
(None, 3, 3, 512)
output:
(None, 1, 1, 512)
flatten_1: Flatten
input:
(None, 1, 1, 512)
output:
(None, 512)
fc1: Dense
input:
(None, 512)
output:
(None, 4096)
fc2: Dense
input:
(None, 4096)
output:
(None, 4096)
dropout_1: Dropout
input:
(None, 4096)
output:
(None, 4096)
prediction: Dense
input:
(None, 4096)
output:
(None, 5)

In [4]:

```
print('下载数据')
#下载已经打包好的数据集,本例先验3**汽车、4**恐龙、5**大象、6**花、7**马
path='littleCBIR_none.npz'
path = get_file(path,origin='https://github.com/jsxyhelu/DateSets/raw/master/little
f = np.load(path)
X_train, y_train = f['X_train'], f['y_train']
X_test, y_test = f['X_test'], f['y_test']
#下载的图片进行格式转换
X_train = [cv2.cvtColor(cv2.resize(i, (ishape, ishape)), cv2.COLOR_GRAY2BGR) for i
X_train = np.concatenate([arr[np.newaxis] for arr in X_train]).astype('float32')
X_train /= 255.0

X_test = [cv2.cvtColor(cv2.resize(i, (ishape, ishape)), cv2.COLOR_GRAY2BGR) for i i
X_test = np.concatenate([arr[np.newaxis] for arr in X_test]).astype('float32')
X_test /= 255.0

y_train_ohe = np.array([tran_y(y_train[i]) for i in range(len(y_train))])
y_test_ohe = np.array([tran_y(y_test[i]) for i in range(len(y_test))])
y_train_ohe = y_train_ohe.astype('float32')
y_test_ohe = y_test_ohe.astype('float32')


#agumentation
# 设置生成参数
img_generator = ImageDataGenerator(
        featurewise_center=False,  # set input mean to 0 over the dataset
        samplewise_center=False,  # set each sample mean to 0
        featurewise_std_normalization=False,  # divide inputs by std of the dataset
        samplewise_std_normalization=False,  # divide each input by its std
        zca_whitening=False,  # apply ZCA whitening
        rotation_range=0,  # randomly rotate images in the range (degrees, 0 to 180
        width_shift_range=0.1,  # randomly shift images horizontally (fraction of t
        height_shift_range=0.1,  # randomly shift images vertically (fraction of to
        horizontal_flip=True,  # randomly flip images
        vertical_flip=False)  # randomly flip images

print('模型训练')
#模型训练
#TODOsteps_per_epoch = 400
img_generator.fit(X_train)
log = model_vgg_pretrain.fit_generator(img_generator.flow(X_train,y_train_ohe, batc
score = model_vgg_pretrain.evaluate(X_test, y_test_ohe, verbose=0)
#打印显示结果
print('Test loss:', score[0])
print('Test accuracy:', score[1])
#绘制loss和acc曲线
plt.figure('acc')
plt.subplot(2, 1, 1)
plt.plot(log.history['acc'],'r--',label='Training Accuracy')
plt.plot(log.history['val_acc'],'r-',label='Validation Accuracy')
plt.legend(loc='best')
plt.xlabel('Epochs')
plt.axis([0, epochs, 0.9, 1])
plt.figure('loss')
plt.subplot(2, 1, 2)
plt.plot(log.history['loss'],'b--',label='Training Loss')
plt.plot(log.history['val_loss'],'b-',label='Validation Loss')
plt.legend(loc='best')
plt.xlabel('Epochs')
plt.axis([0, epochs, 0, 1])

plt.show()

model_vgg_pretrain.save('5type4cbirMODEL.h5')
```

```
下载数据
Downloading data from https://github.com/jsxyhelu/DateSets/raw/maste
r/littleCBIR_none.npz (https://github.com/jsxyhelu/DateSets/raw/maste
r/littleCBIR_none.npz)
68108288/68101466 [==============================] - 2s 0us/step
模型训练
Epoch 1/1
10/10 [==============================] - 6s 579ms/step - loss: 1.5097
 - acc: 0.4596 - val_loss: 0.7885 - val_acc: 0.8300
Test loss: 0.7884670257568359
Test accuracy: 0.83
```





In [9]:

```
# Install the PyDrive wrapper & import libraries.
# This only needs to be done once in a notebook.
#!pip install -U -q PyDrive
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials

# Authenticate and create the PyDrive client.
# This only needs to be done once in a notebook.
auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)

# Create & upload a text file.
uploaded = drive.CreateFile()
uploaded.SetContentFile('5type4cbirMODEL.h5')
uploaded.Upload()
print('Uploaded file with ID {}'.format(uploaded.get('id')))
```

```
---------------------------------------------------------------------
------
NameError                                 Traceback (most recent call
 last)
<ipython-input-9-9374e5e4a6b1> in <module>()
      9 # Authenticate and create the PyDrive client.
     10 # This only needs to be done once in a notebook.
---> 11 auth.authenticate_user()
     12 gauth = GoogleAuth()
     13 gauth.credentials =
GoogleCredentials.get_application_default()

NameError: name 'auth' is not defined
```

In [ ]:

```
# Install the PyDrive wrapper & import libraries.
# This only needs to be done once per notebook.
#!pip install -U -q PyDrive
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials

# Authenticate and create the PyDrive client.
# This only needs to be done once per notebook.
auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)
#根据文件名进行下载
file_id = '1qjxAm_QiXdSqBmyIoPl3bfnyLNJxwKo9'
downloaded = drive.CreateFile({'id': file_id})
print('Downloaded content "{}"'.format(downloaded.GetContentString()))
```

In [ ]:

```

```