In [1]:

```
!pip install h5py
!pip install keras
!pip install tensorflow

import keras
from keras.layers import Dense, Conv2D, BatchNormalization, Activation, Dropout
from keras.layers import AveragePooling2D, Input, Flatten
from keras.optimizers import adam_v2
from keras.callbacks import ModelCheckpoint, LearningRateScheduler
from keras import backend as K
from keras.models import Model
import numpy as np
import tensorflow as tf
from keras.utils import np_utils
import os
import matplotlib.pyplot as plt
import pandas as pd
import pickle
from pathlib import Path
from skimage import io
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report

# import Sequential from the keras models module
from keras.models import Sequential

# import Dense, Dropout, Flatten, Conv2D, MaxPooling2D from the keras layers module
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPooling2D
```

```
Requirement already satisfied: h5py in c:\users\nehal\anaconda3\lib\site-packages (3.2.1)
Requirement already satisfied: numpy>=1.19.0 in c:\users\nehal\anaconda3\lib\site-packages (from h5
py) (1.20.3)
Requirement already satisfied: keras in c:\users\nehal\anaconda3\lib\site-packages (2.8.0)
Requirement already satisfied: tensorflow in c:\users\nehal\anaconda3\lib\site-packages (2.8.0)
Requirement already satisfied: tf-estimator-nightly==2.8.0.dev2021122109 in c:\users\nehal\anaconda
3\lib\site-packages (from tensorflow) (2.8.0.dev2021122109)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\nehal\anaconda3\lib\site-packages (f
rom tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\nehal\anaconda3\lib\site-packages (from tens
orflow) (3.2.1)
Requirement already satisfied: wrapt>=1.11.0 in c:\users\nehal\anaconda3\lib\site-packages (from te
nsorflow) (1.12.1)
Requirement already satisfied: tensorboard<2.9,>=2.8 in c:\users\nehal\anaconda3\lib\site-packages
(from tensorflow) (2.8.0)
Requirement already satisfied: absl-py>=0.4.0 in c:\users\nehal\anaconda3\lib\site-packages (from t
ensorflow) (1.0.0)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in c:\users\nehal\anaconda3\lib
\site-packages (from tensorflow) (0.25.0)
Requirement already satisfied: numpy>=1.20 in c:\users\nehal\anaconda3\lib\site-packages (from tens
orflow) (1.20.3)
Requirement already satisfied: gast>=0.2.1 in c:\users\nehal\anaconda3\lib\site-packages (from tens
orflow) (0.5.3)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\nehal\anaconda3\lib\site-packages (f
rom tensorflow) (1.44.0)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\nehal\anaconda3\lib\site-packages (fro
m tensorflow) (3.3.0)
Requirement already satisfied: keras<2.9,>=2.8.0rc0 in c:\users\nehal\anaconda3\lib\site-packages
(from tensorflow) (2.8.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\nehal\anaconda3\lib\site-packages (from
tensorflow) (1.1.0)
Requirement already satisfied: protobuf>=3.9.2 in c:\users\nehal\anaconda3\lib\site-packages (from
tensorflow) (3.20.1)
```

```
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\nehal\anaconda3\lib\site-packag
es (from tensorflow) (3.10.0.2)
Requirement already satisfied: flatbuffers>=1.12 in c:\users\nehal\anaconda3\lib\site-packages (fro
m tensorflow) (2.0)
Requirement already satisfied: keras-preprocessing>=1.1.1 in c:\users\nehal\anaconda3\lib\site-pack
ages (from tensorflow) (1.1.2)
Requirement already satisfied: libclang>=9.0.1 in c:\users\nehal\anaconda3\lib\site-packages (from
tensorflow) (13.0.0)
Requirement already satisfied: six>=1.12.0 in c:\users\nehal\anaconda3\lib\site-packages (from tens
orflow) (1.16.0)
Requirement already satisfied: setuptools in c:\users\nehal\anaconda3\lib\site-packages (from tenso
rflow) (58.0.4)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\nehal\anaconda3\lib\site-packages (fro
m tensorflow) (1.6.3)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\nehal\anaconda3\lib\site-packages (fr
om astunparse>=1.6.0->tensorflow) (0.37.0)
Requirement already satisfied: markdown>=2.6.8 in c:\users\nehal\anaconda3\lib\site-packages (from
tensorboard<2.9,>=2.8->tensorflow) (3.3.6)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in c:\users\nehal\anaconda3\lib\sit
e-packages (from tensorboard<2.9,>=2.8->tensorflow) (0.4.6)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in c:\users\nehal\anaconda3\li
b\site-packages (from tensorboard<2.9,>=2.8->tensorflow) (0.6.1)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in c:\users\nehal\anaconda3\lib\site-p
ackages (from tensorboard<2.9,>=2.8->tensorflow) (1.8.1)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\nehal\anaconda3\lib\site-packages (f
rom tensorboard<2.9,>=2.8->tensorflow) (2.26.0)
Requirement already satisfied: werkzeug>=0.11.15 in c:\users\nehal\anaconda3\lib\site-packages (fro
m tensorboard<2.9,>=2.8->tensorflow) (2.0.2)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\nehal\anaconda3\lib\site-packages
(from tensorboard<2.9,>=2.8->tensorflow) (2.6.6)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\nehal\anaconda3\lib\site-packages
(from google-auth<3,>=1.6.3->tensorboard<2.9,>=2.8->tensorflow) (0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\nehal\anaconda3\lib\site-packages (from go
ogle-auth<3,>=1.6.3->tensorboard<2.9,>=2.8->tensorflow) (4.8)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\nehal\anaconda3\lib\site-packages
(from google-auth<3,>=1.6.3->tensorboard<2.9,>=2.8->tensorflow) (5.0.0)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\nehal\anaconda3\lib\site-packag
es (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.9,>=2.8->tensorflow) (1.3.1)
Requirement already satisfied: importlib-metadata>=4.4 in c:\users\nehal\anaconda3\lib\site-package
s (from markdown>=2.6.8->tensorboard<2.9,>=2.8->tensorflow) (4.8.1)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\nehal\anaconda3\lib\site-packa
ges (from requests<3,>=2.21.0->tensorboard<2.9,>=2.8->tensorflow) (2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\nehal\anaconda3\lib\site-packages
(from requests<3,>=2.21.0->tensorboard<2.9,>=2.8->tensorflow) (1.26.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\nehal\anaconda3\lib\site-packages (fr
om requests<3,>=2.21.0->tensorboard<2.9,>=2.8->tensorflow) (2021.10.8)
Requirement already satisfied: idna<4,>=2.5 in c:\users\nehal\anaconda3\lib\site-packages (from req
uests<3,>=2.21.0->tensorboard<2.9,>=2.8->tensorflow) (3.2)
Requirement already satisfied: zipp>=0.5 in c:\users\nehal\anaconda3\lib\site-packages (from import
lib-metadata>=4.4->markdown>=2.6.8->tensorboard<2.9,>=2.8->tensorflow) (3.6.0)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\nehal\anaconda3\lib\site-packages
(from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.9,>=2.8->tensorflow) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\nehal\anaconda3\lib\site-packages (from
requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.9,>=2.8->tensorflow) (3.
2.0)
```

In [79]:

```
batch_size = 20
epochs = 100
num_classes = 2
depth = 20
subtract_pixel_mean = True
```

In [80]:

```
model_type = 'ResNet%d' % (depth)
```

In [81]:
```
# 2 Load image labels
# load labels.csv from datasets folder using pandas
labels = pd.read_csv('Original_data/dataset_1/labels.csv', index_col=0)

# print value counts for genus
print(labels.genus.value_counts())

# assign the genus label values to y
y = labels.genus.values
```

```
1.0    827
0.0    827
Name: genus, dtype: int64
```

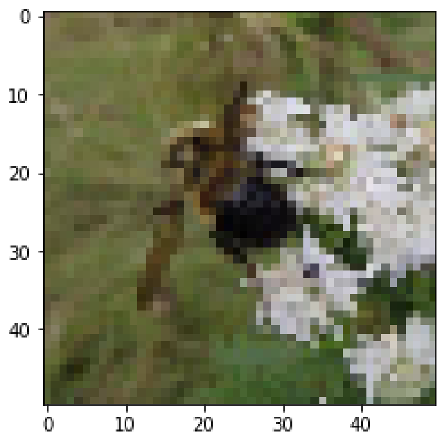
In [83]:
```
# 3. Examine RGB values in an image matrix
# load an image and explore
example_image = io.imread('Original_data/dataset_1/maindataset/{}.jpg'.format(labels.index[0]))

# show image
plt.imshow(example_image)

# print shape
print('Image has shape:', example_image.shape)

# print color channel values for top left pixel
print('RGB values for the top left pixel are:', example_image[0, 0, :])
```

```
Image has shape: (50, 50, 3)
RGB values for the top left pixel are: [127 108  95]
```



In [84]:
```
# 4. Importing the image data
# create empty list
image_list = []

for i in labels.index:
    # load image
    img = io.imread('Original_data/dataset_1/maindataset/{}.jpg'.format(i)).astype(np.float64)

    # append to list of all images
    image_list.append(img)

# convert image list to single array
X = np.array(image_list)

print(X.shape)
```

```
(1654, 50, 50, 3)
```

In [85]:
```
# 5 SPLITTING
# split remaining data into train and test sets
x_train, x_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    test_size=0.3,
                                                    random_state=52)

# examine number of samples in train, test, and validation sets
print('x_train shape:', x_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')
```

```
x_train shape: (1157, 50, 50, 3)
1157 train samples
497 test samples
```

In [87]:

```
input_shape = x_train.shape[1:]
input_shape
```

Out[87]:

```
(50, 50, 3)
```

In [88]:

```
x_train = x_train.astype('float32') / 255
x_test = x_test.astype('float32') / 255
```

In [89]:

```
if subtract_pixel_mean:
  x_train_mean = np.mean(x_train, axis=0)
  x_train -= x_train_mean
  x_test -= x_train_mean
```

In [90]:

```
print('x_train shape:', x_train.shape)
print('y_train shape:', y_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')
```

```
x_train shape: (1157, 50, 50, 3)
y_train shape: (1157,)
1157 train samples
497 test samples
```

In [91]:

```
y_train = keras.utils.np_utils.to_categorical(y_train, num_classes)
y_test = keras.utils.np_utils.to_categorical(y_test, num_classes)
```

In [92]:

```
def lr_schedule(epoch):
  lr = 1e-3
  if epoch > 180:
      lr *= 0.5e-3
  elif epoch > 160:
      lr *= 1e-3
  elif epoch > 120:
      lr *= 1e-2
  elif epoch > 80:
      lr *= 1e-1
  print('Learning rate: ', lr)
  return lr
```

In [93]:

```
def resnet_layer(inputs, num_filters=16, kernel_size=3, strides=1, activation='relu', batch_normali

  conv = Conv2D(num_filters, kernel_size=kernel_size, strides=strides, padding='same')
```

```
    x = inputs
    if conv_first:
        x = conv(x)
        if batch_normalization:
            x = BatchNormalization()(x)
        if activation is not None:
            x = Activation(activation)(x)
    else:
        if batch_normalization:
            x = BatchNormalization()(x)
        if activation is not None:
            x = Activation(activation)(x)
        x = conv(x)
    return x
```

In [94]:

```
def resnet_20(input_shape, depth, num_classes=2):

    if (depth - 2) % 6 != 0:
        raise ValueError('depth should be 6n+2 (eg 20, 32, 44 in [a])')
    # Start model definition.
    num_filters = 16
    num_res_blocks = int((depth - 2) / 6)

    inputs = Input(shape=input_shape)
    x = resnet_layer(inputs=inputs)
    # Instantiate the stack of residual units
    for stack in range(3):
        for res_block in range(num_res_blocks):
            strides = 1
            if stack > 0 and res_block == 0:  # first layer but not first stack
                strides = 2  # downsample
            y = resnet_layer(inputs=x,num_filters=num_filters,strides=strides)
            y = resnet_layer(inputs=y,num_filters=num_filters,activation=None)
            if stack > 0 and res_block == 0:  # first layer but not first stack
                # linear projection residual shortcut connection to match
                # changed dims
                x = resnet_layer(inputs=x,num_filters=num_filters,kernel_size=1,strides=strides,act
            x = keras.layers.add([x, y])
            x = Activation('relu')(x)
            x = Dropout(rate=0.25)(x)
        num_filters *= 2

    # Add classifier on top.
    # v1 does not use BN after last shortcut connection-ReLU
    x = AveragePooling2D(pool_size=8)(x)
    y = Flatten()(x)
    outputs = Dense(num_classes, activation='softmax')(y)

    # Instantiate model.
    model = Model(inputs=inputs, outputs=outputs)
    return model
```

In [95]:

```
model = resnet_20(input_shape=input_shape, depth=depth)
```

In [96]:

```
model.compile(loss='binary_crossentropy',optimizer=adam_v2.Adam(lr=lr_schedule(0)),metrics=['accura
```

```
Learning rate:  0.001
```

In [97]:

```
model.summary()
print(model_type)
```

```
Model: "model_2"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to                     
==================================================================================================
 input_3 (InputLayer)           [(None, 50, 50, 3)]  0           []                               
                                                                                                  
 conv2d_42 (Conv2D)             (None, 50, 50, 16)   448         ['input_3[0][0]']                
                                                                                                  
 batch_normalization_38 (BatchN  (None, 50, 50, 16)  64          ['conv2d_42[0][0]']              
 ormalization)                                                                                    
                                                                                                  
 activation_38 (Activation)     (None, 50, 50, 16)   0           ['batch_normalization_38[0][0]'] 
                                                                                                  
 conv2d_43 (Conv2D)             (None, 50, 50, 16)   2320        ['activation_38[0][0]']          
                                                                                                  
 batch_normalization_39 (BatchN  (None, 50, 50, 16)  64          ['conv2d_43[0][0]']              
 ormalization)                                                                                    
                                                                                                  
 activation_39 (Activation)     (None, 50, 50, 16)   0           ['batch_normalization_39[0][0]'] 
                                                                                                  
 conv2d_44 (Conv2D)             (None, 50, 50, 16)   2320        ['activation_39[0][0]']          
                                                                                                  
 batch_normalization_40 (BatchN  (None, 50, 50, 16)  64          ['conv2d_44[0][0]']              
 ormalization)                                                                                    
                                                                                                  
 add_18 (Add)                   (None, 50, 50, 16)   0           ['activation_38[0][0]',          
                                                                  'batch_normalization_40[0][0]'] 
                                                                                                  
 activation_40 (Activation)     (None, 50, 50, 16)   0           ['add_18[0][0]']                 
                                                                                                  
 dropout_18 (Dropout)           (None, 50, 50, 16)   0           ['activation_40[0][0]']          
                                                                                                  
 conv2d_45 (Conv2D)             (None, 50, 50, 16)   2320        ['dropout_18[0][0]']             
                                                                                                  
 batch_normalization_41 (BatchN  (None, 50, 50, 16)  64          ['conv2d_45[0][0]']              
 ormalization)                                                                                    
                                                                                                  
 activation_41 (Activation)     (None, 50, 50, 16)   0           ['batch_normalization_41[0][0]'] 
                                                                                                  
 conv2d_46 (Conv2D)             (None, 50, 50, 16)   2320        ['activation_41[0][0]']          
                                                                                                  
 batch_normalization_42 (BatchN  (None, 50, 50, 16)  64          ['conv2d_46[0][0]']              
 ormalization)                                                                                    
                                                                                                  
 add_19 (Add)                   (None, 50, 50, 16)   0           ['dropout_18[0][0]',             
                                                                  'batch_normalization_42[0][0]'] 
                                                                                                  
 activation_42 (Activation)     (None, 50, 50, 16)   0           ['add_19[0][0]']                 
                                                                                                  
 dropout_19 (Dropout)           (None, 50, 50, 16)   0           ['activation_42[0][0]']          
                                                                                                  
 conv2d_47 (Conv2D)             (None, 50, 50, 16)   2320        ['dropout_19[0][0]']             
                                                                                                  
 batch_normalization_43 (BatchN  (None, 50, 50, 16)  64          ['conv2d_47[0][0]']              
 ormalization)                                                                                    
                                                                                                  
 activation_43 (Activation)     (None, 50, 50, 16)   0           ['batch_normalization_43[0][0]'] 
                                                                                                  
 conv2d_48 (Conv2D)             (None, 50, 50, 16)   2320        ['activation_43[0][0]']          
                                                                                                  
 batch_normalization_44 (BatchN  (None, 50, 50, 16)  64          ['conv2d_48[0][0]']              
 ormalization)                                                                                    
                                                                                                  
 add_20 (Add)                   (None, 50, 50, 16)   0           ['dropout_19[0][0]',             
                                                                  'batch_normalization_44[0][0]'] 
 activation_44 (Activation)     (None, 50, 50, 16)   0           ['add_20[0][0]']

 dropout_20 (Dropout)           (None, 50, 50, 16)   0           ['activation_44[0][0]']

 conv2d_49 (Conv2D)             (None, 25, 25, 32)   4640        ['dropout_20[0][0]']

 batch_normalization_45 (BatchN  (None, 25, 25, 32)  128         ['conv2d_49[0][0]']
 ormalization)

 activation_45 (Activation)     (None, 25, 25, 32)   0           ['batch_normalization_45[0][0]']

 conv2d_50 (Conv2D)             (None, 25, 25, 32)   9248        ['activation_45[0][0]']

 conv2d_51 (Conv2D)             (None, 25, 25, 32)   544         ['dropout_20[0][0]']

 batch_normalization_46 (BatchN  (None, 25, 25, 32)  128         ['conv2d_50[0][0]']
 ormalization)

 add_21 (Add)                   (None, 25, 25, 32)   0           ['conv2d_51[0][0]',
                                                                  'batch_normalization_46[0][0]']

 activation_46 (Activation)     (None, 25, 25, 32)   0           ['add_21[0][0]']

 dropout_21 (Dropout)           (None, 25, 25, 32)   0           ['activation_46[0][0]']

 conv2d_52 (Conv2D)             (None, 25, 25, 32)   9248        ['dropout_21[0][0]']

 batch_normalization_47 (BatchN  (None, 25, 25, 32)  128         ['conv2d_52[0][0]']
 ormalization)

 activation_47 (Activation)     (None, 25, 25, 32)   0           ['batch_normalization_47[0][0]']

 conv2d_53 (Conv2D)             (None, 25, 25, 32)   9248        ['activation_47[0][0]']

 batch_normalization_48 (BatchN  (None, 25, 25, 32)  128         ['conv2d_53[0][0]']
 ormalization)

 add_22 (Add)                   (None, 25, 25, 32)   0           ['dropout_21[0][0]',
                                                                  'batch_normalization_48[0][0]']

 activation_48 (Activation)     (None, 25, 25, 32)   0           ['add_22[0][0]']

 dropout_22 (Dropout)           (None, 25, 25, 32)   0           ['activation_48[0][0]']

 conv2d_54 (Conv2D)             (None, 25, 25, 32)   9248        ['dropout_22[0][0]']

 batch_normalization_49 (BatchN  (None, 25, 25, 32)  128         ['conv2d_54[0][0]']
 ormalization)

 activation_49 (Activation)     (None, 25, 25, 32)   0           ['batch_normalization_49[0][0]']

 conv2d_55 (Conv2D)             (None, 25, 25, 32)   9248        ['activation_49[0][0]']

 batch_normalization_50 (BatchN  (None, 25, 25, 32)  128         ['conv2d_55[0][0]']
 ormalization)

 add_23 (Add)                   (None, 25, 25, 32)   0           ['dropout_22[0][0]',
                                                                  'batch_normalization_50[0][0]']

 activation_50 (Activation)     (None, 25, 25, 32)   0           ['add_23[0][0]']

 dropout_23 (Dropout)           (None, 25, 25, 32)   0           ['activation_50[0][0]']

 conv2d_56 (Conv2D)             (None, 13, 13, 64)   18496       ['dropout_23[0][0]']

 batch_normalization_51 (BatchN  (None, 13, 13, 64)  256         ['conv2d_56[0][0]']
 ormalization)                                                                                   
                                                                                                  
 activation_51 (Activation)     (None, 13, 13, 64)   0           ['batch_normalization_51[0][0]'] 
                                                                                                  
 conv2d_57 (Conv2D)             (None, 13, 13, 64)   36928       ['activation_51[0][0]']          
                                                                                                  
 conv2d_58 (Conv2D)             (None, 13, 13, 64)   2112        ['dropout_23[0][0]']             
                                                                                                  
 batch_normalization_52 (BatchN  (None, 13, 13, 64)  256         ['conv2d_57[0][0]']              
 ormalization)                                                                                   
                                                                                                  
 add_24 (Add)                   (None, 13, 13, 64)   0           ['conv2d_58[0][0]',              
                                                                  'batch_normalization_52[0][0]'] 
                                                                                                  
 activation_52 (Activation)     (None, 13, 13, 64)   0           ['add_24[0][0]']                 
                                                                                                  
 dropout_24 (Dropout)           (None, 13, 13, 64)   0           ['activation_52[0][0]']          
                                                                                                  
 conv2d_59 (Conv2D)             (None, 13, 13, 64)   36928       ['dropout_24[0][0]']             
                                                                                                  
 batch_normalization_53 (BatchN  (None, 13, 13, 64)  256         ['conv2d_59[0][0]']              
 ormalization)                                                                                   
                                                                                                  
 activation_53 (Activation)     (None, 13, 13, 64)   0           ['batch_normalization_53[0][0]'] 
                                                                                                  
 conv2d_60 (Conv2D)             (None, 13, 13, 64)   36928       ['activation_53[0][0]']          
                                                                                                  
 batch_normalization_54 (BatchN  (None, 13, 13, 64)  256         ['conv2d_60[0][0]']              
 ormalization)                                                                                   
                                                                                                  
 add_25 (Add)                   (None, 13, 13, 64)   0           ['dropout_24[0][0]',             
                                                                  'batch_normalization_54[0][0]'] 
                                                                                                  
 activation_54 (Activation)     (None, 13, 13, 64)   0           ['add_25[0][0]']                 
                                                                                                  
 dropout_25 (Dropout)           (None, 13, 13, 64)   0           ['activation_54[0][0]']          
                                                                                                  
 conv2d_61 (Conv2D)             (None, 13, 13, 64)   36928       ['dropout_25[0][0]']             
                                                                                                  
 batch_normalization_55 (BatchN  (None, 13, 13, 64)  256         ['conv2d_61[0][0]']              
 ormalization)                                                                                   
                                                                                                  
 activation_55 (Activation)     (None, 13, 13, 64)   0           ['batch_normalization_55[0][0]'] 
                                                                                                  
 conv2d_62 (Conv2D)             (None, 13, 13, 64)   36928       ['activation_55[0][0]']          
                                                                                                  
 batch_normalization_56 (BatchN  (None, 13, 13, 64)  256         ['conv2d_62[0][0]']              
 ormalization)                                                                                   
                                                                                                  
 add_26 (Add)                   (None, 13, 13, 64)   0           ['dropout_25[0][0]',             
                                                                  'batch_normalization_56[0][0]'] 
                                                                                                  
 activation_56 (Activation)     (None, 13, 13, 64)   0           ['add_26[0][0]']                 
                                                                                                  
 dropout_26 (Dropout)           (None, 13, 13, 64)   0           ['activation_56[0][0]']          
                                                                                                  
 average_pooling2d_2 (AveragePo  (None, 1, 1, 64)    0           ['dropout_26[0][0]']             
 oling2D)                                                                                        
                                                                                                  
 flatten_2 (Flatten)            (None, 64)           0           ['average_pooling2d_2[0][0]']    
                                                                                                  
 dense_2 (Dense)                (None, 2)            130         ['flatten_2[0][0]']              
                                                                                                  
==================================================================================================
Total params: 273,922
Trainable params: 272,546
```

```
Non-trainable params: 1,376
_________________________________________________________________________________________________
ResNet20
```

In [98]:

```
save_dir = os.path.join(os.getcwd(), 'saved_models_with_dropout')
model_name = 'cifar10_%s_model.{epoch:03d}.h5' % model_type
if not os.path.isdir(save_dir):
    os.makedirs(save_dir)
filepath = os.path.join(save_dir, model_name)
```

In [99]:

```
checkpoint = ModelCheckpoint(filepath=filepath,monitor='val_acc',verbose=1,save_best_only=True)
```

In [100…

```
lr_scheduler = LearningRateScheduler(lr_schedule)
```

In [101…

```
callbacks = [checkpoint, lr_scheduler]
```

In [76]:

```
tf.config.run_functions_eagerly(True)
```

In [103…

```
history = model.fit(x_train, y_train,batch_size=batch_size,epochs=epochs,validation_data=(x_test, y
```

```
Learning rate:  0.001
Epoch 1/100
232/232 [==============================] - ETA: 0s - loss: 0.7296 - accuracy: 0.5324WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 261ms/step - loss: 0.7296 - accuracy: 0.5324 - val_l
oss: 0.7549 - val_accuracy: 0.5171 - lr: 0.0010
Learning rate:  0.001
Epoch 2/100
232/232 [==============================] - ETA: 0s - loss: 0.6680 - accuracy: 0.6059WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.6680 - accuracy: 0.6059 - val_l
oss: 0.6280 - val_accuracy: 0.6499 - lr: 0.0010
Learning rate:  0.001
Epoch 3/100
232/232 [==============================] - ETA: 0s - loss: 0.6150 - accuracy: 0.6724WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.6150 - accuracy: 0.6724 - val_l
oss: 0.7065 - val_accuracy: 0.6016 - lr: 0.0010
Learning rate:  0.001
Epoch 4/100
232/232 [==============================] - ETA: 0s - loss: 0.5962 - accuracy: 0.6845WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.5962 - accuracy: 0.6845 - val_l
oss: 0.5846 - val_accuracy: 0.6801 - lr: 0.0010
Learning rate:  0.001
Epoch 5/100
232/232 [==============================] - ETA: 0s - loss: 0.5607 - accuracy: 0.7252WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 261ms/step - loss: 0.5607 - accuracy: 0.7252 - val_l
oss: 0.7053 - val_accuracy: 0.6922 - lr: 0.0010
Learning rate:  0.001
Epoch 6/100
232/232 [==============================] - ETA: 0s - loss: 0.5318 - accuracy: 0.7442WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.5318 - accuracy: 0.7442 - val_l
oss: 0.4798 - val_accuracy: 0.7746 - lr: 0.0010
Learning rate:  0.001
Epoch 7/100
232/232 [==============================] - ETA: 0s - loss: 0.5221 - accuracy: 0.7373WARNING:tensorf
```

```
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 258ms/step - loss: 0.5221 - accuracy: 0.7373 - val_l
oss: 0.5333 - val_accuracy: 0.7384 - lr: 0.0010
Learning rate:  0.001
Epoch 8/100
232/232 [==============================] - ETA: 0s - loss: 0.4957 - accuracy: 0.7623WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 262ms/step - loss: 0.4957 - accuracy: 0.7623 - val_l
oss: 0.5855 - val_accuracy: 0.6922 - lr: 0.0010
Learning rate:  0.001
Epoch 9/100
232/232 [==============================] - ETA: 0s - loss: 0.5010 - accuracy: 0.7770WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 263ms/step - loss: 0.5010 - accuracy: 0.7770 - val_l
oss: 0.5017 - val_accuracy: 0.7666 - lr: 0.0010
Learning rate:  0.001
Epoch 10/100
232/232 [==============================] - ETA: 0s - loss: 0.4912 - accuracy: 0.7632WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 62s 265ms/step - loss: 0.4912 - accuracy: 0.7632 - val_l
oss: 0.5745 - val_accuracy: 0.7022 - lr: 0.0010
Learning rate:  0.001
Epoch 11/100
232/232 [==============================] - ETA: 0s - loss: 0.5000 - accuracy: 0.7563WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 259ms/step - loss: 0.5000 - accuracy: 0.7563 - val_l
oss: 1.4309 - val_accuracy: 0.6781 - lr: 0.0010
Learning rate:  0.001
Epoch 12/100
232/232 [==============================] - ETA: 0s - loss: 0.4660 - accuracy: 0.7839WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.4660 - accuracy: 0.7839 - val_l
oss: 0.6148 - val_accuracy: 0.6962 - lr: 0.0010
Learning rate:  0.001
Epoch 13/100
232/232 [==============================] - ETA: 0s - loss: 0.4693 - accuracy: 0.7822WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 255ms/step - loss: 0.4693 - accuracy: 0.7822 - val_l
oss: 0.5106 - val_accuracy: 0.7606 - lr: 0.0010
Learning rate:  0.001
Epoch 14/100
232/232 [==============================] - ETA: 0s - loss: 0.4620 - accuracy: 0.7831WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 252ms/step - loss: 0.4620 - accuracy: 0.7831 - val_l
oss: 0.8899 - val_accuracy: 0.5714 - lr: 0.0010
Learning rate:  0.001
Epoch 15/100
232/232 [==============================] - ETA: 0s - loss: 0.4532 - accuracy: 0.7926WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.4532 - accuracy: 0.7926 - val_l
oss: 0.5001 - val_accuracy: 0.7807 - lr: 0.0010
Learning rate:  0.001
Epoch 16/100
232/232 [==============================] - ETA: 0s - loss: 0.4552 - accuracy: 0.7960WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.4552 - accuracy: 0.7960 - val_l
oss: 0.5851 - val_accuracy: 0.7042 - lr: 0.0010
Learning rate:  0.001
Epoch 17/100
232/232 [==============================] - ETA: 0s - loss: 0.4315 - accuracy: 0.8021WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.4315 - accuracy: 0.8021 - val_l
oss: 0.8108 - val_accuracy: 0.5211 - lr: 0.0010
Learning rate:  0.001
Epoch 18/100
232/232 [==============================] - ETA: 0s - loss: 0.4350 - accuracy: 0.8047WARNING:tensorf
```

```
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 255ms/step - loss: 0.4350 - accuracy: 0.8047 - val_l
oss: 0.7195 - val_accuracy: 0.6439 - lr: 0.0010
Learning rate:  0.001
Epoch 19/100
232/232 [==============================] - ETA: 0s - loss: 0.4288 - accuracy: 0.8099WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 255ms/step - loss: 0.4288 - accuracy: 0.8099 - val_l
oss: 0.5868 - val_accuracy: 0.7545 - lr: 0.0010
Learning rate:  0.001
Epoch 20/100
232/232 [==============================] - ETA: 0s - loss: 0.4251 - accuracy: 0.8029WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.4251 - accuracy: 0.8029 - val_l
oss: 0.4086 - val_accuracy: 0.8068 - lr: 0.0010
Learning rate:  0.001
Epoch 21/100
232/232 [==============================] - ETA: 0s - loss: 0.4027 - accuracy: 0.8194WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 259ms/step - loss: 0.4027 - accuracy: 0.8194 - val_l
oss: 0.5289 - val_accuracy: 0.7827 - lr: 0.0010
Learning rate:  0.001
Epoch 22/100
232/232 [==============================] - ETA: 0s - loss: 0.4186 - accuracy: 0.8142WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 255ms/step - loss: 0.4186 - accuracy: 0.8142 - val_l
oss: 0.4372 - val_accuracy: 0.8048 - lr: 0.0010
Learning rate:  0.001
Epoch 23/100
232/232 [==============================] - ETA: 0s - loss: 0.3895 - accuracy: 0.8384WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.3895 - accuracy: 0.8384 - val_l
oss: 0.5580 - val_accuracy: 0.7425 - lr: 0.0010
Learning rate:  0.001
Epoch 24/100
232/232 [==============================] - ETA: 0s - loss: 0.4033 - accuracy: 0.8271WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.4033 - accuracy: 0.8271 - val_l
oss: 0.4148 - val_accuracy: 0.7928 - lr: 0.0010
Learning rate:  0.001
Epoch 25/100
232/232 [==============================] - ETA: 0s - loss: 0.3901 - accuracy: 0.8220WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 263ms/step - loss: 0.3901 - accuracy: 0.8220 - val_l
oss: 0.8379 - val_accuracy: 0.7425 - lr: 0.0010
Learning rate:  0.001
Epoch 26/100
232/232 [==============================] - ETA: 0s - loss: 0.3813 - accuracy: 0.8323WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 259ms/step - loss: 0.3813 - accuracy: 0.8323 - val_l
oss: 0.6985 - val_accuracy: 0.6861 - lr: 0.0010
Learning rate:  0.001
Epoch 27/100
232/232 [==============================] - ETA: 0s - loss: 0.3932 - accuracy: 0.8306WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 260ms/step - loss: 0.3932 - accuracy: 0.8306 - val_l
oss: 0.6177 - val_accuracy: 0.6861 - lr: 0.0010
Learning rate:  0.001
Epoch 28/100
232/232 [==============================] - ETA: 0s - loss: 0.3941 - accuracy: 0.8332WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 260ms/step - loss: 0.3941 - accuracy: 0.8332 - val_l
oss: 0.4519 - val_accuracy: 0.7887 - lr: 0.0010
Learning rate:  0.001
Epoch 29/100
232/232 [==============================] - ETA: 0s - loss: 0.3686 - accuracy: 0.8505WARNING:tensorf
```

```
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.3686 - accuracy: 0.8505 - val_l
oss: 0.5433 - val_accuracy: 0.7445 - lr: 0.0010
Learning rate:  0.001
Epoch 30/100
232/232 [==============================] - ETA: 0s - loss: 0.3695 - accuracy: 0.8462WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 254ms/step - loss: 0.3695 - accuracy: 0.8462 - val_l
oss: 0.5710 - val_accuracy: 0.7425 - lr: 0.0010
Learning rate:  0.001
Epoch 31/100
232/232 [==============================] - ETA: 0s - loss: 0.3539 - accuracy: 0.8557WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 258ms/step - loss: 0.3539 - accuracy: 0.8557 - val_l
oss: 0.4546 - val_accuracy: 0.7968 - lr: 0.0010
Learning rate:  0.001
Epoch 32/100
232/232 [==============================] - ETA: 0s - loss: 0.3477 - accuracy: 0.8548WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 62s 269ms/step - loss: 0.3477 - accuracy: 0.8548 - val_l
oss: 1.4921 - val_accuracy: 0.5573 - lr: 0.0010
Learning rate:  0.001
Epoch 33/100
232/232 [==============================] - ETA: 0s - loss: 0.3738 - accuracy: 0.8289WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 260ms/step - loss: 0.3738 - accuracy: 0.8289 - val_l
oss: 0.7093 - val_accuracy: 0.6942 - lr: 0.0010
Learning rate:  0.001
Epoch 34/100
232/232 [==============================] - ETA: 0s - loss: 0.3088 - accuracy: 0.8634WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 260ms/step - loss: 0.3088 - accuracy: 0.8634 - val_l
oss: 0.5564 - val_accuracy: 0.7384 - lr: 0.0010
Learning rate:  0.001
Epoch 35/100
232/232 [==============================] - ETA: 0s - loss: 0.3741 - accuracy: 0.8470WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.3741 - accuracy: 0.8470 - val_l
oss: 0.5161 - val_accuracy: 0.7988 - lr: 0.0010
Learning rate:  0.001
Epoch 36/100
232/232 [==============================] - ETA: 0s - loss: 0.3553 - accuracy: 0.8479WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 261ms/step - loss: 0.3553 - accuracy: 0.8479 - val_l
oss: 0.4473 - val_accuracy: 0.8068 - lr: 0.0010
Learning rate:  0.001
Epoch 37/100
232/232 [==============================] - ETA: 0s - loss: 0.3096 - accuracy: 0.8781WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 261ms/step - loss: 0.3096 - accuracy: 0.8781 - val_l
oss: 0.6862 - val_accuracy: 0.6962 - lr: 0.0010
Learning rate:  0.001
Epoch 38/100
232/232 [==============================] - ETA: 0s - loss: 0.3437 - accuracy: 0.8505WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 253ms/step - loss: 0.3437 - accuracy: 0.8505 - val_l
oss: 0.4817 - val_accuracy: 0.7867 - lr: 0.0010
Learning rate:  0.001
Epoch 39/100
232/232 [==============================] - ETA: 0s - loss: 0.3645 - accuracy: 0.8557WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 251ms/step - loss: 0.3645 - accuracy: 0.8557 - val_l
oss: 0.3847 - val_accuracy: 0.8249 - lr: 0.0010
Learning rate:  0.001
Epoch 40/100
232/232 [==============================] - ETA: 0s - loss: 0.3361 - accuracy: 0.8522WARNING:tensorf
```

```
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 62s 266ms/step - loss: 0.3361 - accuracy: 0.8522 - val_l
oss: 0.5450 - val_accuracy: 0.7505 - lr: 0.0010
Learning rate:  0.001
Epoch 41/100
232/232 [==============================] - ETA: 0s - loss: 0.3224 - accuracy: 0.8643WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.3224 - accuracy: 0.8643 - val_l
oss: 0.4221 - val_accuracy: 0.8310 - lr: 0.0010
Learning rate:  0.001
Epoch 42/100
232/232 [==============================] - ETA: 0s - loss: 0.3014 - accuracy: 0.8747WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 260ms/step - loss: 0.3014 - accuracy: 0.8747 - val_l
oss: 0.4247 - val_accuracy: 0.7988 - lr: 0.0010
Learning rate:  0.001
Epoch 43/100
232/232 [==============================] - ETA: 0s - loss: 0.2994 - accuracy: 0.8747WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.2994 - accuracy: 0.8747 - val_l
oss: 0.5559 - val_accuracy: 0.7404 - lr: 0.0010
Learning rate:  0.001
Epoch 44/100
232/232 [==============================] - ETA: 0s - loss: 0.3120 - accuracy: 0.8678WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 255ms/step - loss: 0.3120 - accuracy: 0.8678 - val_l
oss: 0.4328 - val_accuracy: 0.8089 - lr: 0.0010
Learning rate:  0.001
Epoch 45/100
232/232 [==============================] - ETA: 0s - loss: 0.2893 - accuracy: 0.8876WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 259ms/step - loss: 0.2893 - accuracy: 0.8876 - val_l
oss: 0.5966 - val_accuracy: 0.7726 - lr: 0.0010
Learning rate:  0.001
Epoch 46/100
232/232 [==============================] - ETA: 0s - loss: 0.3096 - accuracy: 0.8712WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 261ms/step - loss: 0.3096 - accuracy: 0.8712 - val_l
oss: 0.4563 - val_accuracy: 0.8028 - lr: 0.0010
Learning rate:  0.001
Epoch 47/100
232/232 [==============================] - ETA: 0s - loss: 0.2715 - accuracy: 0.8825WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.2715 - accuracy: 0.8825 - val_l
oss: 0.4962 - val_accuracy: 0.8089 - lr: 0.0010
Learning rate:  0.001
Epoch 48/100
232/232 [==============================] - ETA: 0s - loss: 0.2901 - accuracy: 0.8704WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.2901 - accuracy: 0.8704 - val_l
oss: 0.5337 - val_accuracy: 0.7586 - lr: 0.0010
Learning rate:  0.001
Epoch 49/100
232/232 [==============================] - ETA: 0s - loss: 0.2802 - accuracy: 0.9015WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 259ms/step - loss: 0.2802 - accuracy: 0.9015 - val_l
oss: 0.5831 - val_accuracy: 0.7626 - lr: 0.0010
Learning rate:  0.001
Epoch 50/100
232/232 [==============================] - ETA: 0s - loss: 0.2947 - accuracy: 0.8695WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.2947 - accuracy: 0.8695 - val_l
oss: 0.5038 - val_accuracy: 0.7787 - lr: 0.0010
Learning rate:  0.001
Epoch 51/100
232/232 [==============================] - ETA: 0s - loss: 0.2563 - accuracy: 0.8971WARNING:tensorf
```

```
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.2563 - accuracy: 0.8971 - val_l
oss: 0.4073 - val_accuracy: 0.8149 - lr: 0.0010
Learning rate:  0.001
Epoch 52/100
232/232 [==============================] - ETA: 0s - loss: 0.2557 - accuracy: 0.8937WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 259ms/step - loss: 0.2557 - accuracy: 0.8937 - val_l
oss: 0.7022 - val_accuracy: 0.6901 - lr: 0.0010
Learning rate:  0.001
Epoch 53/100
232/232 [==============================] - ETA: 0s - loss: 0.2704 - accuracy: 0.8920WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 255ms/step - loss: 0.2704 - accuracy: 0.8920 - val_l
oss: 0.6647 - val_accuracy: 0.7203 - lr: 0.0010
Learning rate:  0.001
Epoch 54/100
232/232 [==============================] - ETA: 0s - loss: 0.2557 - accuracy: 0.8920WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.2557 - accuracy: 0.8920 - val_l
oss: 0.4523 - val_accuracy: 0.7887 - lr: 0.0010
Learning rate:  0.001
Epoch 55/100
232/232 [==============================] - ETA: 0s - loss: 0.2516 - accuracy: 0.8928WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 260ms/step - loss: 0.2516 - accuracy: 0.8928 - val_l
oss: 0.6059 - val_accuracy: 0.7505 - lr: 0.0010
Learning rate:  0.001
Epoch 56/100
232/232 [==============================] - ETA: 0s - loss: 0.2267 - accuracy: 0.9170WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 262ms/step - loss: 0.2267 - accuracy: 0.9170 - val_l
oss: 0.4527 - val_accuracy: 0.8068 - lr: 0.0010
Learning rate:  0.001
Epoch 57/100
232/232 [==============================] - ETA: 0s - loss: 0.2395 - accuracy: 0.8989WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 261ms/step - loss: 0.2395 - accuracy: 0.8989 - val_l
oss: 0.5172 - val_accuracy: 0.7827 - lr: 0.0010
Learning rate:  0.001
Epoch 58/100
232/232 [==============================] - ETA: 0s - loss: 0.2244 - accuracy: 0.9041WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 255ms/step - loss: 0.2244 - accuracy: 0.9041 - val_l
oss: 0.8085 - val_accuracy: 0.7485 - lr: 0.0010
Learning rate:  0.001
Epoch 59/100
232/232 [==============================] - ETA: 0s - loss: 0.2434 - accuracy: 0.8971WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 254ms/step - loss: 0.2434 - accuracy: 0.8971 - val_l
oss: 0.6916 - val_accuracy: 0.7062 - lr: 0.0010
Learning rate:  0.001
Epoch 60/100
232/232 [==============================] - ETA: 0s - loss: 0.2277 - accuracy: 0.9101WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.2277 - accuracy: 0.9101 - val_l
oss: 0.5765 - val_accuracy: 0.7404 - lr: 0.0010
Learning rate:  0.001
Epoch 61/100
232/232 [==============================] - ETA: 0s - loss: 0.2582 - accuracy: 0.8928WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 255ms/step - loss: 0.2582 - accuracy: 0.8928 - val_l
oss: 0.5100 - val_accuracy: 0.7887 - lr: 0.0010
Learning rate:  0.001
Epoch 62/100
232/232 [==============================] - ETA: 0s - loss: 0.2334 - accuracy: 0.9015WARNING:tensorf
```

```
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 255ms/step - loss: 0.2334 - accuracy: 0.9015 - val_l
oss: 0.5933 - val_accuracy: 0.7384 - lr: 0.0010
Learning rate:  0.001
Epoch 63/100
232/232 [==============================] - ETA: 0s - loss: 0.2189 - accuracy: 0.9188WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.2189 - accuracy: 0.9188 - val_l
oss: 0.6642 - val_accuracy: 0.7404 - lr: 0.0010
Learning rate:  0.001
Epoch 64/100
232/232 [==============================] - ETA: 0s - loss: 0.1903 - accuracy: 0.9274WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 260ms/step - loss: 0.1903 - accuracy: 0.9274 - val_l
oss: 0.4814 - val_accuracy: 0.7887 - lr: 0.0010
Learning rate:  0.001
Epoch 65/100
232/232 [==============================] - ETA: 0s - loss: 0.1845 - accuracy: 0.9257WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 263ms/step - loss: 0.1845 - accuracy: 0.9257 - val_l
oss: 0.7371 - val_accuracy: 0.7183 - lr: 0.0010
Learning rate:  0.001
Epoch 66/100
232/232 [==============================] - ETA: 0s - loss: 0.2176 - accuracy: 0.9049WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 254ms/step - loss: 0.2176 - accuracy: 0.9049 - val_l
oss: 0.5407 - val_accuracy: 0.7968 - lr: 0.0010
Learning rate:  0.001
Epoch 67/100
232/232 [==============================] - ETA: 0s - loss: 0.1926 - accuracy: 0.9222WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 256ms/step - loss: 0.1926 - accuracy: 0.9222 - val_l
oss: 0.5381 - val_accuracy: 0.7867 - lr: 0.0010
Learning rate:  0.001
Epoch 68/100
232/232 [==============================] - ETA: 0s - loss: 0.2188 - accuracy: 0.9170WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 258ms/step - loss: 0.2188 - accuracy: 0.9170 - val_l
oss: 0.5283 - val_accuracy: 0.7968 - lr: 0.0010
Learning rate:  0.001
Epoch 69/100
232/232 [==============================] - ETA: 0s - loss: 0.2213 - accuracy: 0.9118WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 259ms/step - loss: 0.2213 - accuracy: 0.9118 - val_l
oss: 0.5511 - val_accuracy: 0.7606 - lr: 0.0010
Learning rate:  0.001
Epoch 70/100
232/232 [==============================] - ETA: 0s - loss: 0.1862 - accuracy: 0.9300WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 63s 270ms/step - loss: 0.1862 - accuracy: 0.9300 - val_l
oss: 0.5305 - val_accuracy: 0.7907 - lr: 0.0010
Learning rate:  0.001
Epoch 71/100
232/232 [==============================] - ETA: 0s - loss: 0.1841 - accuracy: 0.9222WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 254ms/step - loss: 0.1841 - accuracy: 0.9222 - val_l
oss: 0.5022 - val_accuracy: 0.8028 - lr: 0.0010
Learning rate:  0.001
Epoch 72/100
232/232 [==============================] - ETA: 0s - loss: 0.1724 - accuracy: 0.9352WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 252ms/step - loss: 0.1724 - accuracy: 0.9352 - val_l
oss: 0.5536 - val_accuracy: 0.7626 - lr: 0.0010
Learning rate:  0.001
Epoch 73/100
232/232 [==============================] - ETA: 0s - loss: 0.2035 - accuracy: 0.9205WARNING:tensorf
```

```
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 251ms/step - loss: 0.2035 - accuracy: 0.9205 - val_l
oss: 0.5326 - val_accuracy: 0.7968 - lr: 0.0010
Learning rate:  0.001
Epoch 74/100
232/232 [==============================] - ETA: 0s - loss: 0.1872 - accuracy: 0.9274WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 252ms/step - loss: 0.1872 - accuracy: 0.9274 - val_l
oss: 0.6409 - val_accuracy: 0.7586 - lr: 0.0010
Learning rate:  0.001
Epoch 75/100
232/232 [==============================] - ETA: 0s - loss: 0.1586 - accuracy: 0.9343WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 250ms/step - loss: 0.1586 - accuracy: 0.9343 - val_l
oss: 0.4919 - val_accuracy: 0.7968 - lr: 0.0010
Learning rate:  0.001
Epoch 76/100
232/232 [==============================] - ETA: 0s - loss: 0.2030 - accuracy: 0.9162WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 259ms/step - loss: 0.2030 - accuracy: 0.9162 - val_l
oss: 0.6685 - val_accuracy: 0.7606 - lr: 0.0010
Learning rate:  0.001
Epoch 77/100
232/232 [==============================] - ETA: 0s - loss: 0.1536 - accuracy: 0.9386WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.1536 - accuracy: 0.9386 - val_l
oss: 0.7809 - val_accuracy: 0.7505 - lr: 0.0010
Learning rate:  0.001
Epoch 78/100
232/232 [==============================] - ETA: 0s - loss: 0.1783 - accuracy: 0.9334WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 60s 257ms/step - loss: 0.1783 - accuracy: 0.9334 - val_l
oss: 0.5274 - val_accuracy: 0.8209 - lr: 0.0010
Learning rate:  0.001
Epoch 79/100
232/232 [==============================] - ETA: 0s - loss: 0.1630 - accuracy: 0.9326WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 253ms/step - loss: 0.1630 - accuracy: 0.9326 - val_l
oss: 0.5392 - val_accuracy: 0.8068 - lr: 0.0010
Learning rate:  0.001
Epoch 80/100
232/232 [==============================] - ETA: 0s - loss: 0.1556 - accuracy: 0.9404WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 59s 253ms/step - loss: 0.1556 - accuracy: 0.9404 - val_l
oss: 0.6513 - val_accuracy: 0.7666 - lr: 0.0010
Learning rate:  0.001
Epoch 81/100
232/232 [==============================] - ETA: 0s - loss: 0.1815 - accuracy: 0.9360WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 252ms/step - loss: 0.1815 - accuracy: 0.9360 - val_l
oss: 0.5603 - val_accuracy: 0.8068 - lr: 0.0010
Learning rate:  0.0001
Epoch 82/100
232/232 [==============================] - ETA: 0s - loss: 0.1360 - accuracy: 0.9464WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 246ms/step - loss: 0.1360 - accuracy: 0.9464 - val_l
oss: 0.6001 - val_accuracy: 0.7948 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 83/100
232/232 [==============================] - ETA: 0s - loss: 0.1100 - accuracy: 0.9594WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 252ms/step - loss: 0.1100 - accuracy: 0.9594 - val_l
oss: 0.6247 - val_accuracy: 0.7928 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 84/100
232/232 [==============================] - ETA: 0s - loss: 0.1032 - accuracy: 0.9646WARNING:tensorf
```

```
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 244ms/step - loss: 0.1032 - accuracy: 0.9646 - val_l
oss: 0.5479 - val_accuracy: 0.8048 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 85/100
232/232 [==============================] - ETA: 0s - loss: 0.1136 - accuracy: 0.9637WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 248ms/step - loss: 0.1136 - accuracy: 0.9637 - val_l
oss: 0.6345 - val_accuracy: 0.7787 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 86/100
232/232 [==============================] - ETA: 0s - loss: 0.1026 - accuracy: 0.9602WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 61s 262ms/step - loss: 0.1026 - accuracy: 0.9602 - val_l
oss: 0.6035 - val_accuracy: 0.7968 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 87/100
232/232 [==============================] - ETA: 0s - loss: 0.0985 - accuracy: 0.9697WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 251ms/step - loss: 0.0985 - accuracy: 0.9697 - val_l
oss: 0.5906 - val_accuracy: 0.7907 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 88/100
232/232 [==============================] - ETA: 0s - loss: 0.0919 - accuracy: 0.9706WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 245ms/step - loss: 0.0919 - accuracy: 0.9706 - val_l
oss: 0.6040 - val_accuracy: 0.7948 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 89/100
232/232 [==============================] - ETA: 0s - loss: 0.1049 - accuracy: 0.9594WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 245ms/step - loss: 0.1049 - accuracy: 0.9594 - val_l
oss: 0.5955 - val_accuracy: 0.7907 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 90/100
232/232 [==============================] - ETA: 0s - loss: 0.1059 - accuracy: 0.9594WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 55s 238ms/step - loss: 0.1059 - accuracy: 0.9594 - val_l
oss: 0.5582 - val_accuracy: 0.7907 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 91/100
232/232 [==============================] - ETA: 0s - loss: 0.0879 - accuracy: 0.9663WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 245ms/step - loss: 0.0879 - accuracy: 0.9663 - val_l
oss: 0.6656 - val_accuracy: 0.7746 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 92/100
232/232 [==============================] - ETA: 0s - loss: 0.0942 - accuracy: 0.9646WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 247ms/step - loss: 0.0942 - accuracy: 0.9646 - val_l
oss: 0.6162 - val_accuracy: 0.7867 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 93/100
232/232 [==============================] - ETA: 0s - loss: 0.1040 - accuracy: 0.9576WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 247ms/step - loss: 0.1040 - accuracy: 0.9576 - val_l
oss: 0.6793 - val_accuracy: 0.7726 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 94/100
232/232 [==============================] - ETA: 0s - loss: 0.0871 - accuracy: 0.9706WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 55s 238ms/step - loss: 0.0871 - accuracy: 0.9706 - val_l
oss: 0.6131 - val_accuracy: 0.7907 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 95/100
232/232 [==============================] - ETA: 0s - loss: 0.0771 - accuracy: 0.9715WARNING:tensorf
```

```
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 55s 236ms/step - loss: 0.0771 - accuracy: 0.9715 - val_l
oss: 0.6256 - val_accuracy: 0.7847 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 96/100
232/232 [==============================] - ETA: 0s - loss: 0.0783 - accuracy: 0.9732WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 245ms/step - loss: 0.0783 - accuracy: 0.9732 - val_l
oss: 0.6481 - val_accuracy: 0.7887 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 97/100
232/232 [==============================] - ETA: 0s - loss: 0.0659 - accuracy: 0.9784WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 56s 242ms/step - loss: 0.0659 - accuracy: 0.9784 - val_l
oss: 0.6052 - val_accuracy: 0.7968 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 98/100
232/232 [==============================] - ETA: 0s - loss: 0.0722 - accuracy: 0.9697WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 244ms/step - loss: 0.0722 - accuracy: 0.9697 - val_l
oss: 0.6052 - val_accuracy: 0.7968 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 99/100
232/232 [==============================] - ETA: 0s - loss: 0.0739 - accuracy: 0.9767WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 58s 249ms/step - loss: 0.0739 - accuracy: 0.9767 - val_l
oss: 0.6788 - val_accuracy: 0.7887 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 100/100
232/232 [==============================] - ETA: 0s - loss: 0.0650 - accuracy: 0.9775WARNING:tensorf
low:Can save best model only with val_acc available, skipping.
232/232 [==============================] - 57s 247ms/step - loss: 0.0650 - accuracy: 0.9775 - val_l
oss: 0.6741 - val_accuracy: 0.7847 - lr: 1.0000e-04
```

In [104…

```
with open("./model.json", "w") as json_file:
  json_file.write(model.to_json())
  model.save_weights("./model_weights.h5")
  print("saved model to disk")
```

```
saved model to disk
```

In [105…

```
loss, acc = model.evaluate(x_test, y_test)
```

```
16/16 [==============================] - 2s 117ms/step - loss: 0.6741 - accuracy: 0.7847
```

In [106…

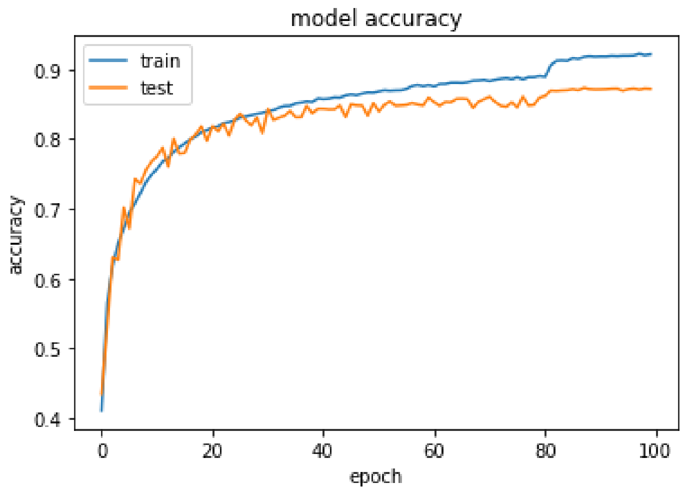
```
print("loss: ",loss,", Accuracy: ", acc)
```

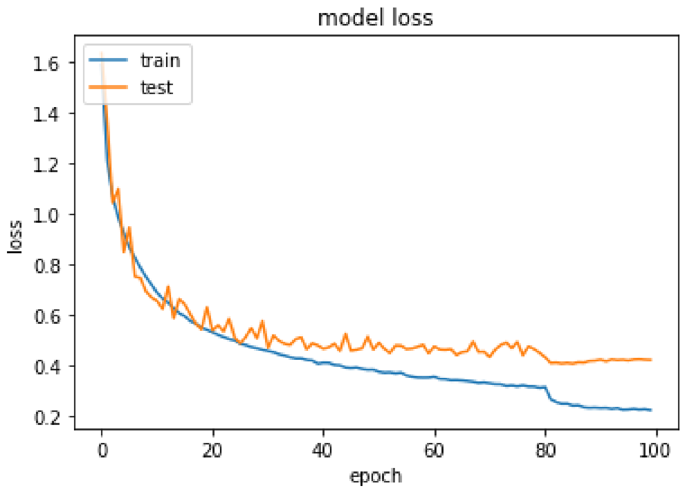
```
loss:  0.6740607619285583 , Accuracy:  0.7847082614898682
```

In [58]:

```
print(history.history.keys())
# summarize history for accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy', 'lr'])
```





In [107…

```
from keras.models import model_from_json
```

In [108…

```
# load json and create model
json_file = open('model.json', 'r')
loaded_model_json = json_file.read()
json_file.close()
loaded_model = model_from_json(loaded_model_json)
# load weights into new model
loaded_model.load_weights("model_weights.h5")
print("Loaded model from disk")

# evaluate loaded model on test data
loaded_model.compile(loss='binary_crossentropy', optimizer='rmsprop', metrics=['accuracy'])
score = loaded_model.evaluate(x_test, y_test, verbose=0)
print('Testing '"%s: %.2f%%" % (loaded_model.metrics_names[1], score[1]*100))

# evaluate loaded model on train data
score = loaded_model.evaluate(x_train, y_train, verbose=0)
print('Training '"%s: %.2f%%" % (loaded_model.metrics_names[1], score[1]*100))
```

```
Loaded model from disk
Testing accuracy: 78.47%
Training accuracy: 99.14%
```

In [ ]: