```
from google.colab import drive
drive.mount('/content/drive')
```

```
     Mounted at /content/drive
```

```
import numpy as np
import os

from keras.models import Sequential
from keras.layers import Dense, Conv2D, Dropout, Flatten
from keras.constraints import maxnorm
from tensorflow.keras.optimizers import Adam

from keras.layers.convolutional import Convolution2D, MaxPooling2D
from keras.callbacks import ModelCheckpoint, LearningRateScheduler
from keras.callbacks import ReduceLROnPlateau, EarlyStopping

from keras.utils import np_utils
import matplotlib.pyplot as plt
from keras.preprocessing.image import ImageDataGenerator
```

```
x_train = '/content/drive/MyDrive/tropical_fruits/train'
x_test  = '/content/drive/MyDrive/tropical_fruits/test'
```

```
x_train = ImageDataGenerator(rescale=1/255)
x_test  = ImageDataGenerator(rescale=1/255)
```

```
x_train_data = x_train.flow_from_directory(
    directory= r"/content/drive/MyDrive/tropical_fruits/train",
    target_size=(224,224),
    batch_size=3,
    class_mode='categorical'
)
x_test_data = x_test.flow_from_directory(
    directory= r"/content/drive/MyDrive/tropical_fruits/test",
    target_size=(224,224),
    batch_size=3,
    class_mode='categorical'
)
```

```
     Found 66 images belonging to 10 classes.
     Found 28 images belonging to 10 classes.
```

```
x_train_data.class_indices
```

```
     {'apple': 0,
      'banana': 1,
      'cherry': 2,
      'coconut': 3,
      'durian': 4,
      'kiwi': 5,
      'mango': 6,
      'orange': 7,
      'pomelo': 8,
      'water melon': 9}
```

```
model = Sequential()

model.add(Conv2D(32,(3,3),input_shape=(224,224,3),padding='same',activation='relu'))
model.add(Dropout(0.2))

model.add(Conv2D(32,(3,3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64,(3,3),activation='relu',padding='same'))
model.add(Dropout(0.2))

model.add(Conv2D(64,(3,3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(128,(3,3),activation='relu',padding='same'))
model.add(Dropout(0.2))

model.add(Conv2D(128,(3,3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dropout(0.2))

model.add(Dense(1024,activation='relu'))
model.add(Dropout(0.2))

model.add(Dense(512,activation='relu'))
model.add(Dropout(0.2))

model.add(Dense(10,activation='softmax'))
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d (Conv2D)             (None, 224, 224, 32)      896       
                                                                 
 dropout (Dropout)           (None, 224, 224, 32)      0         
                                                                 
 conv2d_1 (Conv2D)           (None, 224, 224, 32)      9248      
                                                                 
 max_pooling2d (MaxPooling2D  (None, 112, 112, 32)     0         
 )                                                               
                                                                 
 conv2d_2 (Conv2D)           (None, 112, 112, 64)      18496     
                                                                 
 dropout_1 (Dropout)         (None, 112, 112, 64)      0         
                                                                 
 conv2d_3 (Conv2D)           (None, 112, 112, 64)      36928     
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 56, 56, 64)       0         
 2D)                                                             
                                                                 
 conv2d_4 (Conv2D)           (None, 56, 56, 128)       73856     
                                                                 
 dropout_2 (Dropout)         (None, 56, 56, 128)       0         
                                                                 
 conv2d_5 (Conv2D)           (None, 56, 56, 128)       147584    
                                                                 
 max_pooling2d_2 (MaxPooling  (None, 28, 28, 128)      0         
 2D)                                                             
 flatten (Flatten)           (None, 100352)            0

 dropout_3 (Dropout)         (None, 100352)            0

 dense (Dense)               (None, 1024)              102761472

 dropout_4 (Dropout)         (None, 1024)              0

 dense_1 (Dense)             (None, 512)               524800

 dropout_5 (Dropout)         (None, 512)               0

 dense_2 (Dense)             (None, 10)                5130

=================================================================
Total params: 103,578,410
Trainable params: 103,578,410
Non-trainable params: 0
_________________________________________________________________
```

```
from tensorflow.keras.optimizers import SGD

model.compile(optimizer=Adam(learning_rate=0.0005),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
history = model.fit(x_train_data,epochs= 30, batch_size= 32, verbose= 1,
                    validation_data= x_test_data)
```

```
Epoch 1/30
22/22 [==============================] - 3s 85ms/step - loss: 2.2984 - accuracy: 0.0303 - val
Epoch 2/30
22/22 [==============================] - 2s 77ms/step - loss: 2.2971 - accuracy: 0.0909 - val
Epoch 3/30
22/22 [==============================] - 2s 77ms/step - loss: 2.2982 - accuracy: 0.0758 - val
Epoch 4/30
22/22 [==============================] - 2s 78ms/step - loss: 2.2986 - accuracy: 0.1515 - val
Epoch 5/30
22/22 [==============================] - 2s 75ms/step - loss: 2.2957 - accuracy: 0.0758 - val
Epoch 6/30
22/22 [==============================] - 2s 78ms/step - loss: 2.2972 - accuracy: 0.0606 - val
Epoch 7/30
22/22 [==============================] - 2s 76ms/step - loss: 2.2966 - accuracy: 0.0909 - val
Epoch 8/30
22/22 [==============================] - 2s 76ms/step - loss: 2.2995 - accuracy: 0.1061 - val
Epoch 9/30
22/22 [==============================] - 2s 76ms/step - loss: 2.2993 - accuracy: 0.1061 - val
Epoch 10/30
22/22 [==============================] - 2s 71ms/step - loss: 2.2967 - accuracy: 0.1515 - val
Epoch 11/30
22/22 [==============================] - 2s 78ms/step - loss: 2.2984 - accuracy: 0.1364 - val
Epoch 12/30
22/22 [==============================] - 2s 76ms/step - loss: 2.2989 - accuracy: 0.0909 - val
Epoch 13/30
22/22 [==============================] - 2s 79ms/step - loss: 2.2977 - accuracy: 0.0909 - val
Epoch 14/30
22/22 [==============================] - 2s 79ms/step - loss: 2.2992 - accuracy: 0.0606 - val
Epoch 15/30
22/22 [==============================] - 2s 76ms/step - loss: 2.2978 - accuracy: 0.0909 - val
Epoch 16/30
22/22 [==============================] - 2s 79ms/step - loss: 2.2974 - accuracy: 0.0606 - val
Epoch 17/30
22/22 [==============================] - 2s 77ms/step - loss: 2.2990 - accuracy: 0.0909 - val
Epoch 18/30
22/22 [==============================] - 2s 79ms/step - loss: 2.2982 - accuracy: 0.0909 - val
Epoch 19/30
```

```
22/22 [==============================] - 2s 78ms/step - loss: 2.2963 - accuracy: 0.1061 - val
Epoch 20/30
22/22 [==============================] - 2s 79ms/step - loss: 2.2988 - accuracy: 0.1212 - val
Epoch 21/30
22/22 [==============================] - 2s 74ms/step - loss: 2.2966 - accuracy: 0.1515 - val
Epoch 22/30
22/22 [==============================] - 2s 79ms/step - loss: 2.2963 - accuracy: 0.0606 - val
Epoch 23/30
22/22 [==============================] - 2s 77ms/step - loss: 2.2960 - accuracy: 0.1364 - val
Epoch 24/30
22/22 [==============================] - 2s 78ms/step - loss: 2.2991 - accuracy: 0.1061 - val
Epoch 25/30
22/22 [==============================] - 2s 75ms/step - loss: 2.2975 - accuracy: 0.1212 - val
Epoch 26/30
22/22 [==============================] - 2s 78ms/step - loss: 2.2990 - accuracy: 0.0909 - val
Epoch 27/30
22/22 [==============================] - 2s 75ms/step - loss: 2.2952 - accuracy: 0.1818 - val
Epoch 28/30
22/22 [==============================] - 2s 78ms/step - loss: 2.2970 - accuracy: 0.1364 - val
Epoch 29/30
```

```
model.save('fruit_recognition.h5')
```

```
from keras.models import load_model
recognition = load_model('fruit_recognition.h5')

labels = {0: 'apple', 1: 'banana', 2: 'cherry', 3: 'coconut',
          4: 'durian', 5: 'kiwi', 6: 'mango', 7: 'orange',
          8: 'pomelo', 9: 'water melon'}
```

```
from keras.preprocessing.image import load_img, img_to_array
img = load_img('mango.jpg', target_size = (224,224))
plt.imshow(img)
img = img_to_array(img)
img = img.reshape(1,224,224,3)
img = img.astype('float32')
img = img/255
img.shape

val = recognition.predict(img)
np.argmax(val, axis=1)
print('This is ', labels[np.argmax(val)])
```

```
This is  mango
```