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

```
     Found 27 images belonging to 3 classes.
     Found 11 images belonging to 3 classes.
```

```
x_train_data.class_indices
```

```
     {'son': 0, 'thai': 1, 'trang': 2}
```

```
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(3,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, 3)                 1539

=================================================================
Total params: 103,574,819
Trainable params: 103,574,819
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= 10, batch_size= 64, verbose= 1,
                    validation_data= x_test_data)
```

```
Epoch 1/10
9/9 [==============================] - 3s 210ms/step - loss: 0.9802 - accuracy: 0.4444 - val_lo
Epoch 2/10
9/9 [==============================] - 2s 179ms/step - loss: 0.5771 - accuracy: 0.7778 - val_lo
Epoch 3/10
9/9 [==============================] - 2s 182ms/step - loss: 0.4747 - accuracy: 0.7778 - val_lo
Epoch 4/10
9/9 [==============================] - 2s 190ms/step - loss: 0.4710 - accuracy: 0.6296 - val_lo
Epoch 5/10
9/9 [==============================] - 2s 185ms/step - loss: 0.2854 - accuracy: 0.8148 - val_lo
Epoch 6/10
9/9 [==============================] - 2s 218ms/step - loss: 0.0753 - accuracy: 1.0000 - val_lo
Epoch 7/10
9/9 [==============================] - 2s 188ms/step - loss: 0.0133 - accuracy: 1.0000 - val_lo
Epoch 8/10
9/9 [==============================] - 2s 179ms/step - loss: 0.5424 - accuracy: 0.8148 - val_lo
Epoch 9/10
9/9 [==============================] - 2s 183ms/step - loss: 0.3001 - accuracy: 0.9259 - val_lo
Epoch 10/10
9/9 [==============================] - 2s 183ms/step - loss: 0.0351 - accuracy: 1.0000 - val_lo
```

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

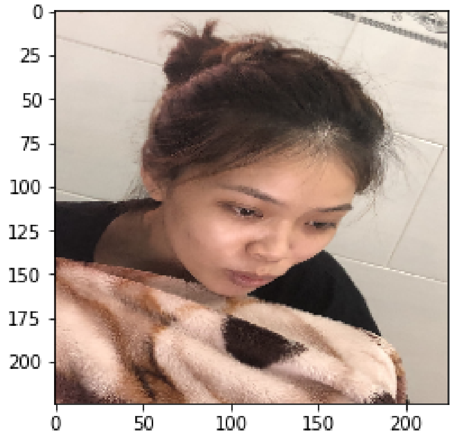
```
from keras.models import load_model
recognition = load_model('face_id.h5')

labels = {0: 'Son', 1: 'Thai', 2: 'Trang'}
```

```
from keras.preprocessing.image import load_img, img_to_array
img = load_img('/content/drive/MyDrive/face_id/train/trang/z3435894541634_445ed013efadd16590420ec8fc6
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  Trang
```



✓ 1s completed at 8:43 AM