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

```
Found 80 images belonging to 8 classes.
Found 38 images belonging to 8 classes.
```

```
x_train_data.class_indices
```

```
{'chicken': 0,
 'lam rice': 1,
 'noodle': 2,
 'pancakes': 3,
 'pho': 4,
 'pork_bologna': 5,
 'rib_rice': 6,
 'spring roll': 7}
```

```
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(8,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, 8)                 4104

=================================================================
Total params: 103,577,384
Trainable params: 103,577,384
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= 4, batch_size= 64, verbose= 1,
                    validation_data= x_test_data)
```

```
Epoch 1/4
27/27 [==============================] - 4s 132ms/step - loss: 0.0670 - accuracy: 0.9750 - val_
Epoch 2/4
27/27 [==============================] - 3s 121ms/step - loss: 2.2995e-04 - accuracy: 1.0000 -
Epoch 3/4
27/27 [==============================] - 3s 117ms/step - loss: 9.9109e-04 - accuracy: 1.0000 -
Epoch 4/4
27/27 [==============================] - 4s 159ms/step - loss: 0.0065 - accuracy: 1.0000 - val_
```

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

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

labels = {0: 'Chicken fried', 1: 'Lam rice', 2: 'Noodles', 3: 'Banh Xeo',
          4: 'Pho', 5: 'Pork_bologna', 6: 'Rib rice', 7: 'Spring rolls'}
```

```
from keras.preprocessing.image import load_img, img_to_array
img = load_img('banhxeo.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  Banh Xeo
```