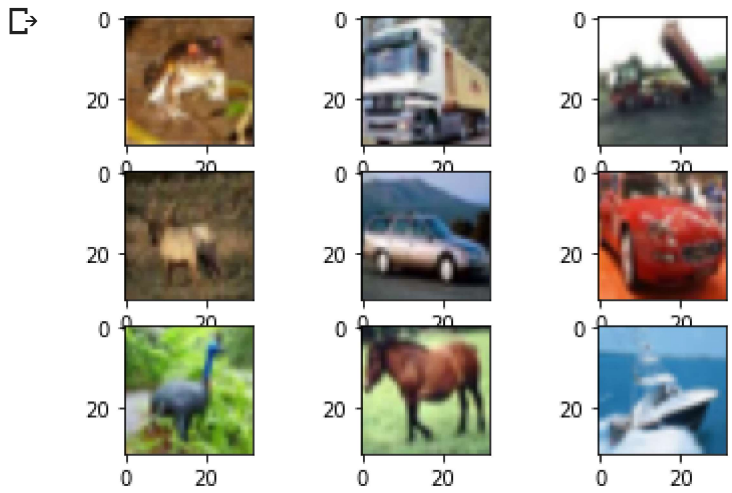
```
#Trần·Nam·Phương·20146470
from keras.datasets import cifar10
import matplotlib.pyplot as plt

(X_train,y_train),(X_test,y_test)=cifar10.load_data()
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170500096/170498071 [==============================] - 3s 0us/step
170508288/170498071 [==============================] - 3s 0us/step
```

```
for i in range(9):
        plt.subplot(330+i+1)
        plt.imshow(X_train[i],cmap=plt.get_cmap('gray'))
plt.show()
```



```
X_train.shape
```

```
(50000, 32, 32, 3)
```

```
X_test.shape
```

```
(10000, 32, 32, 3)
```

```
X_train =X_train.reshape(50000, 3072)
X_test = X_test.reshape(10000, 3072)
```

```
X_train= X_train.astype('float32')
X_test=X_test.astype('float32')
```

```
X_train/=255
```

```
X_test /= 255
```

```
pip install np_utils
```

```
Collecting np_utils
  Downloading np_utils-0.6.0.tar.gz (61 kB)
     |████████████████████████████████| 61 kB 431 kB/s
Requirement already satisfied: numpy>=1.0 in /usr/local/lib/python3.7/dist-packages (fro
Building wheels for collected packages: np-utils
  Building wheel for np-utils (setup.py) ... done
  Created wheel for np-utils: filename=np_utils-0.6.0-py3-none-any.whl size=56459 sha256
  Stored in directory: /root/.cache/pip/wheels/d2/83/71/a781667865955ae7dc18e5a4038401de
Successfully built np-utils
Installing collected packages: np-utils
Successfully installed np-utils-0.6.0
```

```
from tensorflow.keras.utils import to_categorical
y_train=to_categorical(y_train, 10)
y_test=to_categorical(y_test, 10)
```

```
y_train
```

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 1.],
       [0., 0., 0., ..., 0., 0., 1.],
       ...,
       [0., 0., 0., ..., 0., 0., 1.],
       [0., 1., 0., ..., 0., 0., 0.],
       [0., 1., 0., ..., 0., 0., 0.]], dtype=float32)
```

```
y_test
```

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 0., 1., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 1., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 1., 0., 0.]], dtype=float32)
```

```
from keras.models import Sequential
from keras.layers import Dense,Activation,Dropout
model = Sequential()
model.add(Dense(2000,activation='relu',input_shape=(3072,)))
model.add(Dropout(0.2))
model.add(Dense(1272,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(10,activation = 'softmax'))
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 2000)              6146000

 dropout (Dropout)           (None, 2000)              0

 dense_1 (Dense)             (None, 1272)              2545272

 dropout_1 (Dropout)         (None, 1272)              0

 dense_2 (Dense)             (None, 10)                12730

=================================================================
Total params: 8,704,002
Trainable params: 8,704,002
Non-trainable params: 0
_________________________________________________________________
```

```
from tensorflow.keras.optimizers import RMSprop
model.compile(loss='categorical_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])
```

```
from keras.callbacks import EarlyStopping
history = model.fit(X_train,y_train,batch_size = 128,epochs=20,verbose=1,validation_split=0.2
```

```
Epoch 1/20
313/313 [==============================] - 55s 171ms/step - loss: 2.5719 - accuracy: 0.2
Epoch 2/20
313/313 [==============================] - 54s 171ms/step - loss: 1.8836 - accuracy: 0.3
Epoch 3/20
313/313 [==============================] - 55s 174ms/step - loss: 1.7987 - accuracy: 0.3
Epoch 4/20
313/313 [==============================] - 54s 173ms/step - loss: 1.7437 - accuracy: 0.3
Epoch 5/20
313/313 [==============================] - 54s 172ms/step - loss: 1.6996 - accuracy: 0.3
Epoch 6/20
313/313 [==============================] - 54s 171ms/step - loss: 1.6739 - accuracy: 0.4
Epoch 7/20
313/313 [==============================] - 54s 172ms/step - loss: 1.6449 - accuracy: 0.4
313/313 [==============================] - 53s 171ms/step - loss: 1.6219 - accuracy: 0.4
Epoch 9/20
313/313 [==============================] - 56s 179ms/step - loss: 1.6059 - accuracy: 0.4
Epoch 10/20
313/313 [==============================] - 55s 175ms/step - loss: 1.5918 - accuracy: 0.4
Epoch 11/20
313/313 [==============================] - 54s 174ms/step - loss: 1.5720 - accuracy: 0.4
Epoch 12/20
313/313 [==============================] - 54s 172ms/step - loss: 1.5613 - accuracy: 0.4
Epoch 13/20
313/313 [==============================] - 54s 171ms/step - loss: 1.5517 - accuracy: 0.4
Epoch 14/20
313/313 [==============================] - 53s 171ms/step - loss: 1.5444 - accuracy: 0.4
Epoch 15/20
```

```
313/313 [==============================] - 53s 171ms/step - loss: 1.5364 - accuracy: 0.4
Epoch 16/20
313/313 [==============================] - 54s 171ms/step - loss: 1.5177 - accuracy: 0.4
Epoch 17/20
313/313 [==============================] - 53s 171ms/step - loss: 1.5181 - accuracy: 0.4
Epoch 18/20
313/313 [==============================] - 54s 172ms/step - loss: 1.5033 - accuracy: 0.4
Epoch 19/20
313/313 [==============================] - 54s 171ms/step - loss: 1.5083 - accuracy: 0.4
Epoch 20/20
313/313 [==============================] - 57s 182ms/step - loss: 1.4960 - accuracy: 0.4
```

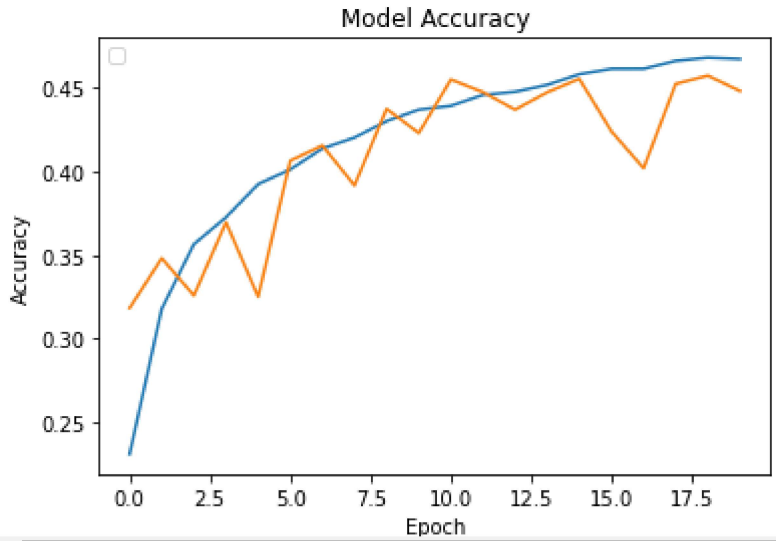
```
model.save('B2.h5')
```

```
score=model.evaluate(X_test,y_test,verbose=0)
print('Test loss: ',score[0])
print('Test Accuracy: ',score[1])
```

```
Test loss:  1.541363000869751
Test Accuracy:  0.4544000029563904
```

```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train'],['Validation'],loc='upper left')
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: UserWarning: Legend does
A proxy artist may be used instead.
See: http://matplotlib.org/users/legend_guide.html#creating-artists-specifically-for-add
```



```
y_pred=model.predict(X_test)
print(y_pred)
```

```
[[1.2301471e-01 1.4106187e-02 7.6187037e-02 ... 5.3512054e-03
  2.1373500e-01 1.2120246e-02]
 [7.6309137e-02 3.5069847e-01 6.7098171e-04 ... 1.8572294e-04
  4.4934779e-01 1.2236102e-01]
 [7.5353019e-02 5.5849932e-02 6.6245473e-03 ... 1.9589837e-03
  8.2112700e-01 3.2437377e-02]
 ...
 [1.3929274e-03 1.9471996e-05 3.0727324e-01 ... 1.1021401e-03
  2.4549854e-03 9.0179739e-05]
 [5.2717224e-02 5.9501339e-02 1.4562559e-01 ... 1.3214591e-01
  5.2540783e-02 7.8842141e-02]
 [1.5285189e-01 2.7881444e-02 1.5701732e-01 ... 1.6057406e-01
  9.3715318e-02 4.1292217e-02]]
```

✓ 5 giây hoàn thành lúc 21:16