Nguyễn Thái Bình - 19146050 - Nhóm 02CLC - 11_Money

```
import numpy as np
from tensorflow import keras
from tensorflow.keras.models import load_model
from tensorflow.keras.utils import load_img,img_to_array
from tensorflow.keras.preprocessing import image
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt
import os
import matplotlib.pyplot as plt
from skimage import io
from keras.models import Sequential
from keras.utils import np_utils
from keras.layers import Dense,Activation,Dropout,LSTM,BatchNormalization
from keras.layers import Flatten
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras.utils import to_categorical
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
```

```
trainset='/content/drive/MyDrive/11_Money_data/train'
validationset='/content/drive/MyDrive/11_Money_data/validation'
train=ImageDataGenerator(rescale=1/255.0,validation_split=0.1)
validation=ImageDataGenerator(rescale=1/255.0,validation_split=0.9)
```

```
train_data=train.flow_from_directory(trainset,target_size=(150,150),batch_size=10,class_mode=
validation_set=validation.flow_from_directory(validationset,target_size=(150,150),batch_size=
```

```
Found 312 images belonging to 11 classes.
Found 80 images belonging to 11 classes.
```

```
print(train_data.class_indices)
print(validation_set.class_indices)
```

```
{'0.2k': 0, '0.5k': 1, '100k': 2, '10k': 3, '1k': 4, '200k': 5, '20k': 6, '2k': 7, '500k
{'0.2k': 0, '0.5k': 1, '100k': 2, '10k': 3, '1k': 4, '200k': 5, '20k': 6, '2k': 7, '500k
```

```
model=Sequential()
model.add(Conv2D(32,(3,3),activation='relu',input_shape=(150,150,3)))
model.add(MaxPooling2D((2,2)))
model.add(Conv2D(64,(3,3),activation='relu'))
model.add(MaxPooling2D((2,2)))
model.add(Conv2D(128,(3,3),activation='relu'))
model.add(MaxPooling2D((2,2)))
```

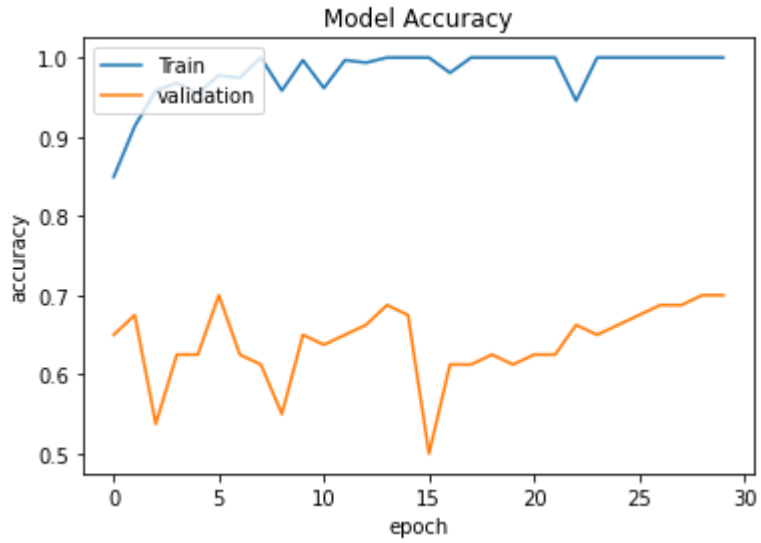
```
model.add(Flatten())
model.add(Dense(128,activation='relu'))
model.add(Dense(11,activation='softmax'))


model.compile(loss='categorical_crossentropy',optimizer='rmsprop',metrics=['accuracy'])
history=model.fit(train_data,batch_size=10,epochs=30,verbose=1,validation_data=validation_set
```

```
Epoch 2/30
32/32 [==============================] - 21s 641ms/step - loss: 0.2256 - accuracy: 0.
Epoch 3/30
32/32 [==============================] - 21s 646ms/step - loss: 0.1396 - accuracy: 0.
Epoch 4/30
32/32 [==============================] - 21s 646ms/step - loss: 0.0995 - accuracy: 0.
Epoch 5/30
32/32 [==============================] - 21s 645ms/step - loss: 0.1982 - accuracy: 0.
Epoch 6/30
32/32 [==============================] - 21s 646ms/step - loss: 0.0940 - accuracy: 0.
Epoch 7/30
32/32 [==============================] - 21s 647ms/step - loss: 0.0928 - accuracy: 0.
Epoch 8/30
32/32 [==============================] - 21s 648ms/step - loss: 0.0013 - accuracy: 1.
Epoch 9/30
32/32 [==============================] - 21s 656ms/step - loss: 0.1658 - accuracy: 0.
Epoch 10/30
32/32 [==============================] - 21s 648ms/step - loss: 0.0163 - accuracy: 0.
Epoch 11/30
32/32 [==============================] - 21s 647ms/step - loss: 0.2282 - accuracy: 0.
Epoch 12/30
32/32 [==============================] - 26s 800ms/step - loss: 0.0155 - accuracy: 0.
Epoch 13/30
32/32 [==============================] - 21s 646ms/step - loss: 0.0196 - accuracy: 0.
Epoch 14/30
32/32 [==============================] - 21s 652ms/step - loss: 4.4539e-05 - accuracy
Epoch 15/30
32/32 [==============================] - 21s 641ms/step - loss: 1.1086e-05 - accuracy
Epoch 16/30
32/32 [==============================] - 22s 684ms/step - loss: 4.4828e-04 - accuracy
Epoch 17/30
32/32 [==============================] - 21s 644ms/step - loss: 0.2246 - accuracy: 0.
Epoch 18/30
32/32 [==============================] - 21s 662ms/step - loss: 2.4110e-04 - accuracy
Epoch 19/30
32/32 [==============================] - 21s 639ms/step - loss: 3.0178e-05 - accuracy
Epoch 20/30
32/32 [==============================] - 21s 645ms/step - loss: 6.2269e-06 - accuracy
Epoch 21/30
32/32 [==============================] - 21s 643ms/step - loss: 1.1967e-06 - accuracy
Epoch 22/30
32/32 [==============================] - 21s 647ms/step - loss: 3.9431e-07 - accuracy
Epoch 23/30
32/32 [==============================] - 21s 645ms/step - loss: 1.6086 - accuracy: 0.
Epoch 24/30
32/32 [==============================] - 21s 646ms/step - loss: 4.2312e-04 - accuracy
Epoch 25/30
```

```
32/32 [==============================] - 21s 647ms/step - loss: 5.4705e-05 - accuracy
Epoch 26/30
32/32 [==============================] - 21s 650ms/step - loss: 1.7031e-05 - accuracy
Epoch 27/30
32/32 [==============================] - 21s 645ms/step - loss: 3.7440e-06 - accuracy
Epoch 28/30
32/32 [==============================] - 21s 649ms/step - loss: 8.6388e-07 - accuracy
Epoch 29/30
32/32 [==============================] - 21s 647ms/step - loss: 1.7385e-07 - accuracy
Epoch 30/30
32/32 [==============================] - 21s 648ms/step - loss: 4.0119e-08 - 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','validation'],loc='upper left')
plt.show()
```




```
model.save('/content/drive/MyDrive/BT AI/11_Money.h5')
```

```
load_model('/content/drive/MyDrive/BT AI/11_Money.h5')
```

```
<keras.engine.sequential.Sequential at 0x7fb799cc3990>
```

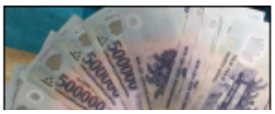
```
generator= ImageDataGenerator(rescale=1./255)
generator_data=generator.flow_from_directory('/content/drive/MyDrive/Test money',batch_size=5

money={0:'200',
       1:'500',
       2:'100k',
       3:'10k',
       4:'1k',
```

```
      5:'200k',
      6:'20k',
      7:'2k',
      8:'500k',
      9:'50k',
      10:'5k'}

plt.figure(figsize=(15,15))
for i in range(len(generator_data.filenames)):
    plt.subplot(5,5,i+1)
    plt.imshow(io.imread(os.path.join(generator_data.directory,generator_data.filenames[i])))
    plt.xticks([])
    plt.yticks([])
    img=load_img('/content/drive/MyDrive/Test money/'+generator_data.filenames[i],target_size
    img=img_to_array(img)
    img=img.reshape(1,150,150,3)
    img=img.astype('float')
    img=img/255
    plt.xlabel(money[np.argmax(model.predict(img))])
plt.show()
```

```
Found 17 images belonging to 1 classes.
```



500k

200k

200k

200k

200

500k

50k

50k

100k

5k

✓ 4 giây hoàn thành lúc 01:28