Deep Learning Code Explanation

เทรนโมเดล VGG16

```
import tensorflow as tf
import numpy as np
from tensorflow.keras.preprocessing import image
from sklearn.metrics import accuracy_score, f1_score, precision_score, confusion_matrix, roc_curve, auc, plot_roc_curve
```

```
path_train = "dataset/train"

img_train = tf.keras.preprocessing.image_dataset_from_directory(
    path_train,
    validation_split=0.2,
    subset = "training",
    seed = 125,
    image_size = (224,224),
    batch_size = 32
)

img_validation = tf.keras.preprocessing.image_dataset_from_directory(
    path_train,
    validation_split=0.2,
    subset = "validation",
    seed = 125,
    image_size = (224,224),
    batch_size = 32
)
```

link

ทำการ import library และ ทำการ import รูปภาพที่จะใช้ train image โดยแบ่งเป็น img_train กับ img_validation

```
VGG16_MODEL=tf.keras.applications.VGG16(input_shape=(224,224,3),
                                        include_top=False,
                                        weights='imagenet')
VGG16_MODEL.trainable=False
global_average_layer = tf.keras.layers.GlobalAveragePooling2D()
prediction_layer = tf.keras.layers.Dense(3,activation='softmax')
```

```
model = tf.keras.Sequential([
  VGG16_MODEL,
  global_average_layer,
  prediction_layer
])

model.compile(optimizer="adam",
              loss=tf.keras.losses.sparse_categorical_crossentropy,
              metrics=["accuracy"])
```

```
model.summary()
```

link

ทำการโหลด model VGG16 มาใช้และทำการกำหนด parameter ที่ VGG16 ต้องใช้ จากนั้นก็ใช้ summary() เพื่อดู layer

```
history = model.fit(img_train,
                    epochs=18,
                    steps_per_epoch=30,
                    validation_steps=2,
                    validation_data=img_validation)
model.save("deeplearningVGG16.h5")
```

```
#import Test img

path_test = "dataset/test"

img_test = tf.keras.preprocessing.image_dataset_from_directory(
    path_test,
    image_size = (224,224)
)


label = ["glaucoma","normal","other"]

loadmodel = tf.keras.models.load_model("deeplearningVGG16.h5")

predict = loadmodel.predict(img_test)
prediction = np.argmax(predict)
```

link

จากนั้นก็ทำการ train model VGG16 ด้วย img_train และนำโมเดลที่ผ่านการ train แล้วมาทดสอบด้วย validation ก่อน (ในที่นี้ไม่ได้ใส่ไว้ในโค้ด) ค่อยนำไปทดสอบกับ img_test และให้แสดงค่าที่ predict สูงที่สุดของทั้ง 3 class ในแต่ละรูป

พลอตกราฟ ROC

```python
import tensorflow as tf
from keras.models import load_model
from keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt
import numpy as np
from sklearn import metrics
from sklearn.metrics import roc_curve, auc, confusion_matrix
import seaborn as sns
```

```python
path_test = "dataset/test"
model = load_model('deeplearningVGG16.h5')

img_test = ImageDataGenerator().flow_from_directory(
    path_test,
    target_size = (224,224),
    shuffle = False
)
```

```python
result = img_test.classes
```

```python
predict = model.predict(img_test)
```

```python
print(result)
print(predict)
```

link

ทำการโหลด img_test และ model ของ VGG16 มาเพื่อเอามาทำนายค่าที่ได้ และเก็บค่าที่ทำนายของทั้ง 3 class เก็บไว้ใน predict และค่าที่บอกว่าเป็น class ไหนไว้ในตัวแปร result

```
fprG , tprG , thres = metrics.roc_curve(result,predict[:,0],pos_label=0)
fprN , tprN , thres = metrics.roc_curve(result,predict[:,1],pos_label=1)
fprO , tprO , thres = metrics.roc_curve(result,predict[:,2],pos_label=2)

roc_aucG = auc(fprG, tprG)
roc_aucN = auc(fprN, tprN)
roc_aucO = auc(fprO, tprO)

lw = 2
plt.plot(fprG,tprG,lw=2,label='ROC curve Glaucoma (area = %.2f)'%roc_aucG)
plt.plot(fprN,tprN,lw=lw,label='ROC curve Normal (area = %.2f)'%roc_aucN)
plt.plot(fprO,tprO,lw=2,label='ROC curve Other (area = %.2f)'%roc_aucO)
plt.plot([0,1],[0,1],color ='navy',lw=lw,linestyle='--')
plt.xlim([0.0,1.0])
plt.ylim([0.0,1.1])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC')
plt.legend(loc="lower right")
plt.show()
```

```
predictResult = predict.argmax(axis = 1)

print(metrics.classification_report(result,predictResult,digits=5))
```

```
cf_matrix = confusion_matrix(result, predictResult)
print(cf_matrix)
```

link

ทำการพลอตกราฟด้วย parameter ที่ต้องใช้ในการพลอต และทำการแสดงค่า accuracy,callback,f1-score และ precision

```
conf_mat = confusion_matrix(result, predictResult)
print(np.sum(cf_matrix))
conf_mat_normalized = np.array([conf_mat[0] / (np.sum(cf_matrix[0])),conf_mat[1] / (np.sum(cf_matrix[1])),conf_mat[2] / (np.sum(cf_matrix[2]))])

print(conf_mat_normalized)
sns.heatmap(conf_mat_normalized,annot=True,fmt='.2%',cmap='Blues')
plt.ylabel('True label')
plt.xlabel('Predicted label')
```

link

ทำการหาค่า confusion matrix ของแต่ละ class โดยใช้ seaborn ในการแสดงค่า