```
from keras.models import Sequential
from keras.layers import Activation
from keras.layers.core import Dense, Flatten
from keras.optimizers import Adam
from keras.callbacks import TensorBoard, EarlyStopping
import keras.optimizers
from sklearn.metrics import classification_report
import keras.optimizers
from keras.applications import vgg16
import numpy as np
import random
import os
from tqdm import tqdm
import pickle
import cv2



from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from tensorflow.keras.callbacks import TensorBoard, EarlyStopping
import pickle
import time
import numpy as np
import keras.optimizers
from sklearn.metrics import classification_report


from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
!pip install tensorflow
!pip install keras
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-packages (2.12.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=2.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.5.26)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.56.0)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.8.0)
Requirement already satisfied: jax>=0.3.15 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.4.13)
Requirement already satisfied: keras<2.13,>=2.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (16.0.6)
Requirement already satisfied: numpy<1.24,>=1.22 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.22.4)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.1)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in /usr/local/lib/p
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.16.0)
Requirement already satisfied: tensorboard<2.13,>=2.12 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.3)
Requirement already satisfied: tensorflow-estimator<2.13,>=2.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.1
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.3.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (4.7.1)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->tensorflow) (
Requirement already satisfied: ml-dtypes>=0.1.0 in /usr/local/lib/python3.10/dist-packages (from jax>=0.3.15->tensorflow) (0.2.0)
Requirement already satisfied: scipy>=1.7 in /usr/local/lib/python3.10/dist-packages (from jax>=0.3.15->tensorflow) (1.10.1)
Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2.12->tens
Requirement already satisfied: google-auth-oauthlib<1.1,>=0.5 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2.12->tensorflow
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2.12->tensor
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2.12->tensorflow
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tenso
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensor
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from google-auth-oauthlib<1.1,>
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorbo
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tens
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard<2.13,
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from werkzeug>=1.0.1->tensorboard<2.13
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-au
Requirement already satisfied: keras in /usr/local/lib/python3.10/dist-packages (2.12.0)
```

```
# Define necessary constants
TEST_DIR = '/content/drive/MyDrive/archive (5)/Testing'
TRAIN_DIR = '/content/drive/MyDrive/archive (5)/Training'
IMG_SIZE = 224
CATEGORIES = ["glioma","meningioma","notumor","pituitary"]


# Creating training dataset
training_data = []

def create_training_data():
    for category in CATEGORIES:
        path = os.path.join(TRAIN_DIR,category)
        class_num = CATEGORIES.index(category)
        for img in tqdm(os.listdir(path)):
          img_array = cv2.imread(os.path.join(path,img) ,cv2.IMREAD_COLOR)
          new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))
          training_data.append([new_array, class_num])

    random.shuffle(training_data)

create_training_data()
#np.save('train_data.npy', training_data)
print(len(training_data))

print("train")
print()
X_train = np.array([i[0] for i in training_data]).reshape(-1,IMG_SIZE,IMG_SIZE,3)
Y_train = [i[1] for i in training_data]

pickle_out = open("X_train.pickle","wb")
pickle.dump(X_train, pickle_out)
pickle_out.close()

pickle_out = open("Y_train.pickle","wb")
pickle.dump(Y_train, pickle_out)
pickle_out.close()
```

```
100%|██████████| 1321/1321 [00:14<00:00, 93.61it/s]
100%|██████████| 1339/1339 [00:15<00:00, 88.90it/s]
100%|██████████| 1595/1595 [00:09<00:00, 175.10it/s]
100%|██████████| 1457/1457 [00:12<00:00, 114.84it/s]
5712
train
```

```
# Creating testing dataset
testing_data = []

def create_testing_data():
    for category in CATEGORIES:
        path = os.path.join(TEST_DIR,category)
        class_num = CATEGORIES.index(category)

        for img in tqdm(os.listdir(path)):
          img_array = cv2.imread(os.path.join(path,img) ,cv2.IMREAD_COLOR)
          new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))
          testing_data.append([new_array, class_num])

    random.shuffle(testing_data)

create_testing_data()
#np.save('testing_data.npy', testing_data)
print(len(testing_data))

print("testing")
print()
X_test= np.array([i[0] for i in testing_data]).reshape(-1,IMG_SIZE,IMG_SIZE,3)
Y_test = [i[1] for i in testing_data]

pickle_out = open("X_test.pickle","wb")
pickle.dump(X_test, pickle_out)
pickle_out.close()

pickle_out = open("Y_test.pickle","wb")
pickle.dump(Y_test, pickle_out)
pickle_out.close()
```

```
100%|██████████| 300/300 [00:03<00:00, 84.56it/s]
100%|██████████| 306/306 [00:03<00:00, 91.00it/s]
100%|██████████| 405/405 [00:04<00:00, 96.92it/s]
100%|██████████| 300/300 [00:03<00:00, 82.59it/s]
1311
```

```
testing
```

```
#print(X_train.shape)
print(X_test.shape)
```

```
(1311, 224, 224, 3)
```

```
X_train=X_train[0:2000]
Y_train=Y_train[0:2000]
```

```
#X_train = X_train / 255.0
X_test = X_test / 255.0

#Y_train = np.array(Y_train)
Y_test = np.array(Y_test)
```

```
from keras.models import Sequential
from keras.layers import Activation
from keras.layers.core import Dense, Flatten
from keras.optimizers import Adam
from keras.callbacks import TensorBoard, EarlyStopping
import keras.optimizers
from sklearn.metrics import classification_report
import keras.optimizers

import numpy as np
```

```
dense_layers = [0,1,2]
layer_sizes = [32, 64, 128]
conv_layers = [1, 2, 3]

for dense_layer in dense_layers:
    for layer_size in layer_sizes:
        for conv_layer in conv_layers:
            NAME = "{}-conv-{}-nodes-{}-dense".format(conv_layer, layer_size, dense_layer)
            if NAME == "3-conv-128-nodes-1-dense":
              print(NAME)
```

```
3-conv-128-nodes-1-dense
```

```
tensorboard = TensorBoard(log_dir='./logs', histogram_freq=0,
                          write_graph=True, write_images=False)
es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=5)
```

```
dense_layers = [0,1, 2]
layer_sizes = [32, 64, 128]
conv_layers = [1, 2, 3]
```

```
for dense_layer in dense_layers:
    for layer_size in layer_sizes:
        for conv_layer in conv_layers:
            NAME = "{}-conv-{}-nodes-{}-dense".format(conv_layer, layer_size, dense_layer)
            if NAME == "3-conv-128-nodes-1-dense":
              print(NAME)
              model = Sequential()

              model.add(Conv2D(layer_size, (3, 3), input_shape=X_train.shape[1:]))
              model.add(Activation('relu'))
              model.add(MaxPooling2D(pool_size=(2, 2)))

              for l in range(conv_layer-1):
                  model.add(Conv2D(layer_size, (3, 3)))
                  model.add(Activation('relu'))
                  model.add(MaxPooling2D(pool_size=(2, 2)))

              model.add(Flatten())
              for _ in range(dense_layer):
                  model.add(Dense(layer_size))
                  model.add(Activation('relu'))
                  model.add(Dropout(0.33))

              model.add(Dense(4))
              model.add(Activation('softmax'))
```

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

#Fit the model
model.fit(X_train, Y_train,
batch_size=32,
epochs=20,
validation_data=(X_test,Y_test),
callbacks=[tensorboard,es])


#Save model
model.save("{}-model.h5".format(NAME))
```

```
3-conv-128-nodes-1-dense
Epoch 1/20
63/63 [==============================] - 27s 277ms/step - loss: 0.9988 - accuracy: 0.5835 - val_loss: 0.7281 - val_accuracy: 0.7231
Epoch 2/20
63/63 [==============================] - 10s 165ms/step - loss: 0.5694 - accuracy: 0.7735 - val_loss: 0.6830 - val_accuracy: 0.7338
Epoch 3/20
63/63 [==============================] - 9s 147ms/step - loss: 0.3877 - accuracy: 0.8545 - val_loss: 0.5326 - val_accuracy: 0.7918
Epoch 4/20
63/63 [==============================] - 9s 147ms/step - loss: 0.2871 - accuracy: 0.8925 - val_loss: 0.4378 - val_accuracy: 0.8429
Epoch 5/20
63/63 [==============================] - 10s 165ms/step - loss: 0.1868 - accuracy: 0.9280 - val_loss: 0.5797 - val_accuracy: 0.8391
Epoch 6/20
63/63 [==============================] - 10s 165ms/step - loss: 0.1291 - accuracy: 0.9535 - val_loss: 0.4561 - val_accuracy: 0.8719
Epoch 7/20
63/63 [==============================] - 10s 163ms/step - loss: 0.0855 - accuracy: 0.9670 - val_loss: 0.5011 - val_accuracy: 0.8764
Epoch 8/20
63/63 [==============================] - 10s 164ms/step - loss: 0.0634 - accuracy: 0.9760 - val_loss: 0.5877 - val_accuracy: 0.8627
Epoch 9/20
63/63 [==============================] - 10s 166ms/step - loss: 0.0745 - accuracy: 0.9700 - val_loss: 0.4996 - val_accuracy: 0.8772
Epoch 9: early stopping
```

```
from keras.models import load_model
model = load_model('/content/drive/MyDrive/3-conv-128-nodes-1-dense-model.h5',compile=True)
```

```
import cv2
from PIL import Image
import numpy as np

image = Image.open('/content/Te-pi_0013.jpg').convert('RGB')  # Convert to RGB mode
image = image.resize((224, 224))
x = np.array(image)
x = x.reshape(1,224, 224, 3)
x=x/255.0
print(x.shape)
```

```
(1, 224, 224, 3)
```

```
y_ = model.predict(x)
```

```
1/1 [==============================] - 0s 22ms/step
```

```
print(y_)
y_bool = np.argmax(y_, axis=1)
z=y_bool[0]
if z==2:
  print("Yesss")
```

```
[[0. 0. 1. 0.]]
Yesss
```

```
scores = model.evaluate(X_test, Y_test, verbose=1)
print('Test loss:', scores[0])
print('Test accuracy:', scores[1])
```

```
41/41 [==============================] - 3s 42ms/step - loss: 0.4996 - accuracy: 0.8772
Test loss: 0.4995712637901306
Test accuracy: 0.8771929740905762
```

```
y_pred = model.predict(X_test, batch_size=64, verbose=1)
y_pred_bool = np.argmax(y_pred, axis=1)
print(classification_report(Y_test, y_pred_bool))
```

```
21/21 [==============================] - 12s 126ms/step
              precision    recall  f1-score   support

           0       0.82      0.93      0.87       300
           1       0.83      0.67      0.74       306
           2       0.91      0.96      0.93       405
           3       0.94      0.94      0.94       300

    accuracy                           0.88      1311
   macro avg       0.87      0.87      0.87      1311
weighted avg       0.88      0.88      0.87      1311
```