Untitled41.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk

```
[ ]
!pip install rarfile
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting rarfile
  Downloading rarfile-4.0-py3-none-any.whl (28 kB)
Installing collected packages: rarfile
Successfully installed rarfile-4.0
```

```
[ ]
import rarfile

with rarfile.RarFile('/content/drive/MyDrive/confident-unconfident.rar') as rf:
    rf.extractall()
```

2m

```
# Import necessary libraries
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten
from tensorflow.keras.applications import VGG19
import os
import numpy as np
import cv2
from sklearn.metrics import confusion_matrix, f1_score, accuracy_score, precision_score, roc_curve, roc_auc_score
import matplotlib.pyplot as plt

# Declared paths for train, val and test sets
train_data_dir = '/content/confident-unconfident/train'
validation_data_dir = '/content/confident-unconfident/val'
test_data_dir = '/content/confident-unconfident/test'

# Declared image dimensions and batch size
img_height = 128
img_width = 128
batch_size = 32
num_classes=2

# Loading train, validation and test sets

# Creating an ImageDataGenerator for train,test and validation data
train_datagen = ImageDataGenerator(
    rescale=1./255)
validation_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)

# Loading the train,test and validation data from directory using flow_from_directory method
train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    color_mode='grayscale',
    class_mode='binary')

validation_generator = validation_datagen.flow_from_directory(
    validation_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    color_mode='grayscale',
    class_mode='binary')

test_generator = test_datagen.flow_from_directory(
    test_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    color_mode='grayscale',
    class_mode='binary')


# Defining the model architecture
model = tf.keras.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu' ,padding="same", input_shape=(img_height, img_width, 1)),
```

```
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu' ,padding="same"),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu',padding="same"),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])

# printing model summary
print(model.summary())

'''
# Define the model architecture
base_model = VGG19(weights='imagenet', include_top=False, input_shape=(img_width, img_height, 3))

for layer in base_model.layers:
    layer.trainable = False

model = Sequential()
model.add(base_model)
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))  #my inference remove dropout layer because may be my model required more feature
model.add(Dense(num_classes, activation='softmax'))
'''
# Defining the optimizer with a specific learning rate
#Optimizer = tf.keras.optimizers.Adam(learning_rate=0.0001)

# Compiling the model
model.compile(optimizer='adam',
              loss=tf.losses.BinaryCrossentropy(),
              metrics=['accuracy'])


# Defining callbacks
logdir = 'logs'
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=logdir)
early_stopping_callback = tf.keras.callbacks.EarlyStopping(
    monitor='val_loss',
    patience=3,
    restore_best_weights=True
)

# Training the model
history = model.fit(train_generator,
                    epochs=8,
                    validation_data=validation_generator,
                    callbacks=[tensorboard_callback, early_stopping_callback])


# Saving the model
model.save('/content/final_model.h5')
```

```
Found 22354 images belonging to 2 classes.
Found 6014 images belonging to 2 classes.
Found 2000 images belonging to 2 classes.
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 128, 128, 32)      320

 max_pooling2d_3 (MaxPooling  (None, 64, 64, 32)       0
 2D)

 conv2d_4 (Conv2D)           (None, 64, 64, 64)        18496

 max_pooling2d_4 (MaxPooling  (None, 32, 32, 64)       0
 2D)

 conv2d_5 (Conv2D)           (None, 32, 32, 32)        18464

 max_pooling2d_5 (MaxPooling  (None, 16, 16, 32)       0
 2D)

 flatten_1 (Flatten)         (None, 8192)              0

 dense_2 (Dense)             (None, 256)               2097408
```

```
 dense_3 (Dense)             (None, 1)                 257

=================================================================
Total params: 2,134,945
Trainable params: 2,134,945
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/8
699/699 [==============================] - 20s 25ms/step - loss: 0.6603 - accuracy: 0.5949 - val_loss: 0.6252 - val_accuracy: 0.6475
Epoch 2/8
699/699 [==============================] - 17s 24ms/step - loss: 0.5998 - accuracy: 0.6657 - val_loss: 0.5766 - val_accuracy: 0.6872
Epoch 3/8
699/699 [==============================] - 19s 28ms/step - loss: 0.5610 - accuracy: 0.7034 - val_loss: 0.5465 - val_accuracy: 0.7142
Epoch 4/8
699/699 [==============================] - 17s 24ms/step - loss: 0.5085 - accuracy: 0.7397 - val_loss: 0.5402 - val_accuracy: 0.7245
Epoch 5/8
699/699 [==============================] - 17s 24ms/step - loss: 0.4418 - accuracy: 0.7865 - val_loss: 0.5374 - val_accuracy: 0.7358
Epoch 6/8
699/699 [==============================] - 17s 24ms/step - loss: 0.3428 - accuracy: 0.8452 - val_loss: 0.5972 - val_accuracy: 0.7403
Epoch 7/8
699/699 [==============================] - 17s 24ms/step - loss: 0.2176 - accuracy: 0.9114 - val_loss: 0.7453 - val_accuracy: 0.7226
Epoch 8/8
699/699 [==============================] - 17s 25ms/step - loss: 0.1174 - accuracy: 0.9567 - val_loss: 0.9374 - val_accuracy: 0.7333
```