# -\*- coding: utf-8 -\*-

"""

Created on Thu Jan 13 03:33:29 2022

@author: WAEL

"""

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten, Conv2D, MaxPooling2D

import numpy as np

import pandas as pd

from pathlib import Path

import os.path

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

import tensorflow as tf

from sklearn.metrics import confusion\_matrix, classification\_report

image\_dir = Path('../content/drive/MyDrive/X-ray (2)/train')

filepaths = list(image\_dir.glob(r'\*\*/\*.png'))

labels = list(map(lambda x: os.path.split(os.path.split(x)[0])[1], filepaths))

filepaths = pd.Series(filepaths, name='Filepath').astype(str)

labels = pd.Series(labels, name='Label')

image\_df = pd.concat([filepaths, labels], axis=1)

train\_df, test\_df = train\_test\_split(image\_df, train\_size=0.7, shuffle=True, random\_state=1)

train\_generator = tf.keras.preprocessing.image.ImageDataGenerator(

    rescale=1./255,

    horizontal\_flip=True,

    width\_shift\_range=0.2,

    height\_shift\_range=0.2,

    validation\_split=0.2

)

test\_generator = tf.keras.preprocessing.image.ImageDataGenerator(

    rescale=1./255

)

train\_images = train\_generator.flow\_from\_dataframe(

    dataframe=train\_df,

    x\_col='Filepath',

    y\_col='Label',

    target\_size=(224, 224),

    color\_mode='rgb',

    class\_mode='binary',

    batch\_size=32,

    shuffle=True,

    seed=42,

    subset='training'

)

val\_images = train\_generator.flow\_from\_dataframe(

    dataframe=train\_df,

    x\_col='Filepath',

    y\_col='Label',

    target\_size=(224, 224),

    color\_mode='rgb',

    class\_mode='binary',

    batch\_size=32,

    shuffle=True,

    seed=42,

    subset='validation'

)

test\_images = test\_generator.flow\_from\_dataframe(

    dataframe=test\_df,

    x\_col='Filepath',

    y\_col='Label',

    target\_size=(224, 224),

    color\_mode='rgb',

    class\_mode='binary',

    batch\_size=32,

    shuffle=False

)

inputs = tf.keras.Input(shape=(224, 224, 3))

x = tf.keras.layers.Conv2D(filters=32, kernel\_size=(3, 3), activation='relu')(inputs)

x = tf.keras.layers.MaxPool2D()(x)

x = tf.keras.layers.Conv2D(filters=32, kernel\_size=(3, 3), activation='relu')(x)

x = tf.keras.layers.MaxPool2D()(x)

x = tf.keras.layers.Dropout(0.2)(x)

x =tf.keras.layers.Flatten()(x)

x = tf.keras.layers.Dense(32, activation='relu')(x)

x = tf.keras.layers.Flatten()(x)

outputs = tf.keras.layers.Dense(1, activation='sigmoid')(x)

model = tf.keras.Model(inputs=inputs, outputs=outputs)

model.compile(

    optimizer='adam',

    loss='binary\_crossentropy',

    metrics=['accuracy']

)

history = model.fit(train\_images,

    validation\_data=val\_images,

    epochs=100,

    callbacks=[

        tf.keras.callbacks.EarlyStopping(

            monitor='val\_loss',

            patience=5,

            restore\_best\_weights=True

        ),

        tf.keras.callbacks.ReduceLROnPlateau(

            monitor='val\_loss',

            patience=3

        )

    ]

)

results = model.evaluate(test\_images, verbose=0)

print("    Test Loss: {:.5f}".format(results[0]))

print("Test Accuracy: {:.2f}%".format(results[1] \* 100))

predictions = (model.predict(test\_images) >= 0.5).astype(np.int)

cm = confusion\_matrix(test\_images.labels, predictions, labels=[0, 1])

clr = classification\_report(test\_images.labels, predictions, labels=[0, 1], target\_names=["covid", "normal"])

plt.figure(figsize=(6, 6))

sns.heatmap(cm, annot=True, fmt='g', vmin=0, cmap='Blues', cbar=False)

plt.xticks(ticks=[0.5, 1.5], labels=["covid", "normal"])

plt.yticks(ticks=[0.5, 1.5], labels=["covid", "normal"])

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.title("Confusion Matrix")

plt.show()

print("Classification Report:\n----------------------\n", clr)

Found 316 validated image filenames belonging to 2 classes.

Found 78 validated image filenames belonging to 2 classes.

Found 169 validated image filenames belonging to 2 classes.

Epoch 1/100

10/10 [==============================] - 32s 3s/step - loss: 1.2384 - accuracy: 0.5570 - val\_loss: 0.6298 - val\_accuracy: 0.6667 - lr: 0.0010

Epoch 2/100

10/10 [==============================] - 30s 3s/step - loss: 0.5640 - accuracy: 0.7342 - val\_loss: 0.6258 - val\_accuracy: 0.6667 - lr: 0.0010

Epoch 3/100

10/10 [==============================] - 30s 3s/step - loss: 0.5367 - accuracy: 0.7342 - val\_loss: 0.6125 - val\_accuracy: 0.6667 - lr: 0.0010

Epoch 4/100

10/10 [==============================] - 30s 3s/step - loss: 0.5482 - accuracy: 0.7437 - val\_loss: 0.7388 - val\_accuracy: 0.6667 - lr: 0.0010

Epoch 5/100

10/10 [==============================] - 30s 3s/step - loss: 0.5248 - accuracy: 0.7437 - val\_loss: 0.5012 - val\_accuracy: 0.6795 - lr: 0.0010

Epoch 6/100

10/10 [==============================] - 30s 3s/step - loss: 0.4920 - accuracy: 0.7437 - val\_loss: 0.5068 - val\_accuracy: 0.8846 - lr: 0.0010

Epoch 7/100

10/10 [==============================] - 30s 3s/step - loss: 0.4594 - accuracy: 0.7753 - val\_loss: 0.4805 - val\_accuracy: 0.6667 - lr: 0.0010

Epoch 8/100

10/10 [==============================] - 30s 3s/step - loss: 0.4032 - accuracy: 0.8133 - val\_loss: 0.4326 - val\_accuracy: 0.7436 - lr: 0.0010

Epoch 9/100

10/10 [==============================] - 31s 3s/step - loss: 0.5024 - accuracy: 0.7468 - val\_loss: 0.4471 - val\_accuracy: 0.7692 - lr: 0.0010

Epoch 10/100

10/10 [==============================] - 31s 3s/step - loss: 0.3962 - accuracy: 0.8323 - val\_loss: 0.4713 - val\_accuracy: 0.7308 - lr: 0.0010

Epoch 11/100

10/10 [==============================] - 30s 3s/step - loss: 0.3721 - accuracy: 0.8228 - val\_loss: 0.4136 - val\_accuracy: 0.7821 - lr: 0.0010

Epoch 12/100

10/10 [==============================] - 30s 3s/step - loss: 0.2996 - accuracy: 0.8766 - val\_loss: 0.3114 - val\_accuracy: 0.8846 - lr: 0.0010

Epoch 13/100

10/10 [==============================] - 30s 3s/step - loss: 0.3155 - accuracy: 0.8608 - val\_loss: 0.2677 - val\_accuracy: 0.8846 - lr: 0.0010

Epoch 14/100

10/10 [==============================] - 31s 3s/step - loss: 0.2476 - accuracy: 0.9146 - val\_loss: 0.2014 - val\_accuracy: 0.9487 - lr: 0.0010

Epoch 15/100

10/10 [==============================] - 31s 3s/step - loss: 0.2447 - accuracy: 0.8861 - val\_loss: 0.1707 - val\_accuracy: 0.9231 - lr: 0.0010

Epoch 16/100

10/10 [==============================] - 31s 3s/step - loss: 0.2691 - accuracy: 0.8956 - val\_loss: 0.1891 - val\_accuracy: 0.9231 - lr: 0.0010

Epoch 17/100

10/10 [==============================] - 30s 3s/step - loss: 0.2846 - accuracy: 0.8829 - val\_loss: 0.2410 - val\_accuracy: 0.9231 - lr: 0.0010

Epoch 18/100

10/10 [==============================] - 31s 3s/step - loss: 0.2795 - accuracy: 0.8892 - val\_loss: 0.2717 - val\_accuracy: 0.8846 - lr: 0.0010

Epoch 19/100

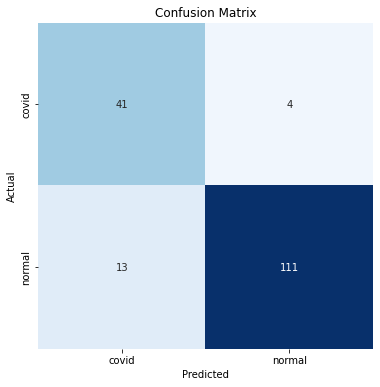
10/10 [==============================] - 30s 3s/step - loss: 0.2374 - accuracy: 0.8956 - val\_loss: 0.1859 - val\_accuracy: 0.9615 - lr: 1.0000e-04

Epoch 20/100

10/10 [==============================] - 30s 3s/step - loss: 0.2158 - accuracy: 0.9304 - val\_loss: 0.2079 - val\_accuracy: 0.9231 - lr: 1.0000e-04

Test Loss: 0.26332

Test Accuracy: 89.94%



Classification Report:

----------------------

precision recall f1-score support

covid 0.76 0.91 0.83 45

normal 0.97 0.90 0.93 124

accuracy 0.90 169

macro avg 0.86 0.90 0.88 169

weighted avg 0.91 0.90 0.90 169