import numpy as np

import pandas as pd

import numpy as np

import cv2

import os

import PIL

from tensorflow import keras

from keras import layers

from keras.models import Sequential

import pathlib

data\_dir=pathlib.Path('/kaggle/input/melenoma-cnn-dataset/Skin cancer ISIC The International Skin Imaging Collaboration')

def laod\_images\_opencv\_extisions(path):

    ext = [".jpg",".gif",".png",".tga",".webp"]   # Add image formats here

    files = []

    images = []

    [files.extend(data\_dir.glob(path + '/\*' + e)) for e in ext]

    return files

classes= [i.split('/')[-1] for i in [x[0] for x in os.walk(str(data\_dir)+'/Train')][1:]]

classes

image\_dir={}

for i in classes:

    image\_dir[i]=laod\_images\_opencv\_extisions('Train/'+i)

test\_dir={}

for i in classes:

    test\_dir[i]=laod\_images\_opencv\_extisions('Test/'+i)

import pandas as pd

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

def get\_dataFrame(dic):

    x,y=[],[]

    for imageName, imagePaths in dic.items():

        for img in imagePaths:

            try:

                x.append(img)

                y.append(imageName)

            except:

                continue

    dataFrame = pd.DataFrame({

        'filename': x,

        'category': y

    })

    dataFrame['category'] = dataFrame['category']

    dataFrame['category']=le.fit\_transform(dataFrame['category'])

    dataFrame['category'] = dataFrame['category'].astype('string')

    dataFrame['filename'] = dataFrame['filename'].astype('string')

    dataFrame=dataFrame.sample(frac=1)

    #dataFrame= dataFrame.reset\_index()

    return dataFrame

train\_data\_frame=pd.DataFrame(get\_dataFrame(image\_dir))

train\_data\_frame=train\_data\_frame.reset\_index()

train\_data\_frame.head()

test\_data\_frame=pd.DataFrame(get\_dataFrame(test\_dir))

test\_data\_frame=test\_data\_frame.reset\_index()

test\_data\_frame.head()

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

train\_df, validate\_df = train\_test\_split(train\_data\_frame,random\_state=10, test\_size=0.2)

train\_df = train\_df.reset\_index()

validate\_df = validate\_df.reset\_index()

# validate\_df = validate\_df.sample(n=100).reset\_index() # use for fast testing code purpose

# train\_df = train\_df.sample(n=1800).reset\_index() # use for fast testing code purpose

total\_train = train\_df.shape[0]

total\_validate = validate\_df.shape[0]

image\_size = 224

input\_shape = (image\_size, image\_size, 3)

epochs = 5

batch\_size = 16

from keras.preprocessing.image import ImageDataGenerator

train\_datagen = ImageDataGenerator(

    rotation\_range=15,

    rescale=1./255,

    shear\_range=0.2,

    zoom\_range=0.2,

    horizontal\_flip=True,

    fill\_mode='nearest',

    width\_shift\_range=0.2,

    height\_shift\_range=0.2

)

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

    train\_df,

    x\_col='filename',

    y\_col='category',

    target\_size=(image\_size, image\_size),

    batch\_size=batch\_size

)

validation\_datagen = ImageDataGenerator(rescale=1./255)

validation\_generator = validation\_datagen.flow\_from\_dataframe(

    validate\_df,

    x\_col='filename',

    y\_col='category',

    target\_size=(image\_size, image\_size),

    batch\_size=batch\_size

)

import matplotlib.pyplot as plt

def data\_argumentation\_show(n, grid\_size):

    sample\_data = train\_datagen.flow\_from\_dataframe(

            (train\_df.sample(n)),

            x\_col = 'filename',

            y\_col = 'category',

            target\_size = (224, 244),

            class\_mode = 'categorical')

  #subplot grid

    fig = plt.figure(figsize = (10,10))

    fig.patch.set\_facecolor('#f5f6f6')

    for i in range(0,grid\_size\*grid\_size):

        plt.subplot(grid\_size,grid\_size, i+1)

        for x,y in sample\_data:

            img = x[0]

            plt.imshow(img)

            plt.axis('off')

            break

            plt.tight\_layout()

            del img

    fig.show()

data\_argumentation\_show(1,4)