VISUALIZATION

add Codeadd Markdown

plt.style.use("dark\_background")

add Codeadd Markdown

GENERAL

add Codeadd Markdown

sns.countplot(Main\_Train\_Data["CATEGORY"])

plt.show()

add Codeadd Markdown

Main\_Train\_Data['CATEGORY'].value\_counts().plot.pie(figsize=(5,5))

plt.show()

add Codeadd Markdown

IMAGES

add Codeadd Markdown

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

x = cv2.imread(Main\_Train\_Data["PNG"][0])

plt.imshow(x)

plt.xlabel(x.shape)

plt.title(Main\_Train\_Data["CATEGORY"][0])

add Codeadd Markdown

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

x = cv2.imread(Main\_Train\_Data["PNG"][993])

plt.imshow(x)

plt.xlabel(x.shape)

plt.title(Main\_Train\_Data["CATEGORY"][993])

add Codeadd Markdown

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

x = cv2.imread(Main\_Train\_Data["PNG"][20])

plt.imshow(x)

plt.xlabel(x.shape)

plt.title(Main\_Train\_Data["CATEGORY"][20])

add Codeadd Markdown

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

x = cv2.imread(Main\_Train\_Data["PNG"][48])

plt.imshow(x)

plt.xlabel(x.shape)

plt.title(Main\_Train\_Data["CATEGORY"][48])

add Codeadd Markdown

fig, axes = plt.subplots(nrows=5,

ncols=5,

figsize=(10,10),

subplot\_kw={"xticks":[],"yticks":[]})

​

for i,ax in enumerate(axes.flat):

ax.imshow(cv2.imread(Main\_Train\_Data["PNG"][i]))

ax.set\_title(Main\_Train\_Data["CATEGORY"][i])

plt.tight\_layout()

plt.show()

add Codeadd Markdown

fig, axes = plt.subplots(nrows=5,

ncols=5,

figsize=(10,10),

subplot\_kw={"xticks":[],"yticks":[]})

​

for i,ax in enumerate(axes.flat):

x = cv2.imread(Main\_Train\_Data["PNG"][i])

x = cv2.cvtColor(x,cv2.COLOR\_RGB2BGR)

ax.imshow(x)

ax.set\_title(Main\_Train\_Data["CATEGORY"][i])

plt.tight\_layout()

plt.show()

add Codeadd Markdown

DETERMINATION TRAIN AND TEST DATA

add Codeadd Markdown

IMAGE GENERATOR

add Codeadd Markdown

Train\_Generator = ImageDataGenerator(rescale=1./255,

shear\_range=0.3,

zoom\_range=0.2,

brightness\_range=[0.2,0.9],

rotation\_range=30,

horizontal\_flip=True,

vertical\_flip=True,

fill\_mode="nearest",

validation\_split=0.1)

add Codeadd Markdown

Test\_Generator = ImageDataGenerator(rescale=1./255)

add Codeadd Markdown

SPLITTING TRAIN AND TEST

add Codeadd Markdown

Train\_Data,Test\_Data = train\_test\_split(Main\_Train\_Data,train\_size=0.9,random\_state=42,shuffle=True)

add Codeadd Markdown

print("TRAIN SHAPE: ",Train\_Data.shape)

print("TEST SHAPE: ",Test\_Data.shape)

add Codeadd Markdown

print(Train\_Data.head(-1))

print("----"\*20)

print(Test\_Data.head(-1))

add Codeadd Markdown

print(Test\_Data["CATEGORY"].value\_counts())

add Codeadd Markdown

encode = LabelEncoder()

add Codeadd Markdown

For\_Prediction\_Class = encode.fit\_transform(Test\_Data["CATEGORY"])

add Codeadd Markdown

How Generator Applied Image Look Like

add Codeadd Markdown

example\_Image = Train\_Data["PNG"][99]

Load\_Image = image.load\_img(example\_Image,target\_size=(200,200))

Array\_Image = image.img\_to\_array(Load\_Image)

Array\_Image = Array\_Image.reshape((1,) + Array\_Image.shape)

​

i = 0

for batch in Train\_Generator.flow(Array\_Image,batch\_size=1):

plt.figure(i)

IMG = plt.imshow(image.array\_to\_img(batch[0]))

i += 1

if i % 4 == 0:

break

plt.show()

add Codeadd Markdown

APPLYING GENERATOR AND TRANSFORMATION TO TENSOR

add Codeadd Markdown

Train\_IMG\_Set = Train\_Generator.flow\_from\_dataframe(dataframe=Train\_Data,

x\_col="PNG",

y\_col="CATEGORY",

color\_mode="rgb",

class\_mode="categorical",

batch\_size=32,

subset="training")

add Codeadd Markdown

Validation\_IMG\_Set = Train\_Generator.flow\_from\_dataframe(dataframe=Train\_Data,

x\_col="PNG",

y\_col="CATEGORY",

color\_mode="rgb",

class\_mode="categorical",

batch\_size=32,

subset="validation")

add Codeadd Markdown

Test\_IMG\_Set = Test\_Generator.flow\_from\_dataframe(dataframe=Test\_Data,

x\_col="PNG",

y\_col="CATEGORY",

color\_mode="rgb",

class\_mode="categorical",

batch\_size=32)

add Codeadd Markdown

CHECKING

add Codeadd Markdown

for data\_batch,label\_batch in Train\_IMG\_Set:

print("DATA SHAPE: ",data\_batch.shape)

print("LABEL SHAPE: ",label\_batch.shape)

break

add Codeadd Markdown

for data\_batch,label\_batch in Validation\_IMG\_Set:

print("DATA SHAPE: ",data\_batch.shape)

print("LABEL SHAPE: ",label\_batch.shape)

break

add Codeadd Markdown

for data\_batch,label\_batch in Test\_IMG\_Set:

print("DATA SHAPE: ",data\_batch.shape)

print("LABEL SHAPE: ",label\_batch.shape)

break

add Codeadd Markdown

print("TRAIN: ")

print(Train\_IMG\_Set.class\_indices)

print(Train\_IMG\_Set.classes[0:5])

print(Train\_IMG\_Set.image\_shape)

print("---"\*20)

print("VALIDATION: ")

print(Validation\_IMG\_Set.class\_indices)

print(Validation\_IMG\_Set.classes[0:5])

print(Validation\_IMG\_Set.image\_shape)

print("---"\*20)

print("TEST: ")

print(Test\_IMG\_Set.class\_indices)

print(Test\_IMG\_Set.classes[0:5])

print(Test\_IMG\_Set.image\_shape)