Edge Intelligence

D. Sai Mohith

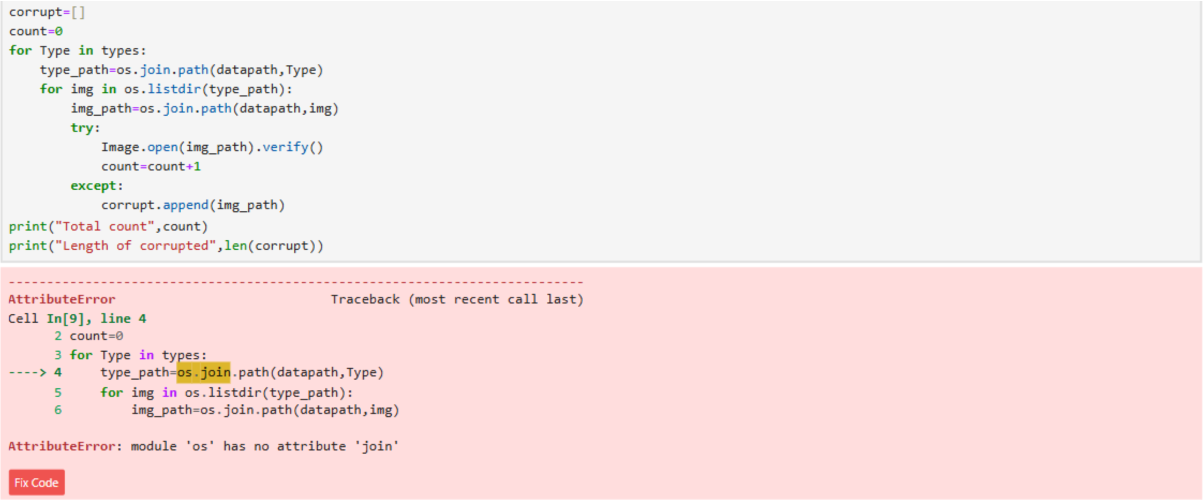
25MML0032

Title: Image Processing of the Animal Dataset

Errors:

Given list of errors include typing mistakes, logical errors or wrong usage of methods

Syntax error-1:

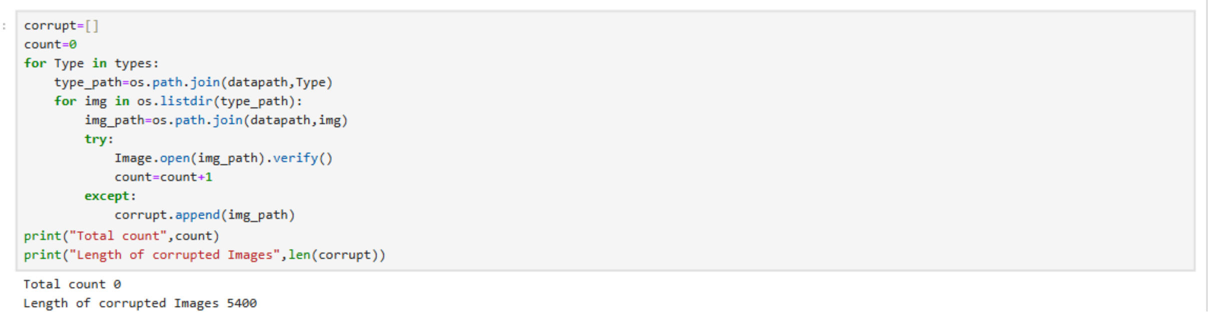


```
corrupt=[]
count=0
for Type in types:
    type_path=os.join.path(datapath,Type)
    for img in os.listdir(type_path):
        img_path=os.join.path(datapath,img)
        try:
            Image.open(img_path).verify()
            count=count+1
        except:
            corrupt.append(img_path)
print("Total count",count)
print("Length of corrupted",len(corrupt))
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[9], line 4
      2 count=0
      3 for Type in types:
----> 4     type_path=os.join.path(datapath,Type)
      5     for img in os.listdir(type_path):
      6         img_path=os.join.path(datapath,img)

AttributeError: module 'os' has no attribute 'join'
```

Logical error-1:



```
corrupt=[]
count=0
for Type in types:
    type_path=os.path.join(datapath,Type)
    for img in os.listdir(type_path):
        img_path=os.path.join(datapath,img)
        try:
            Image.open(img_path).verify()
            count=count+1
        except:
            corrupt.append(img_path)
print("Total count",count)
print("Length of corrupted Images",len(corrupt))
```

```
Total count 0
Length of corrupted Images 5400
```

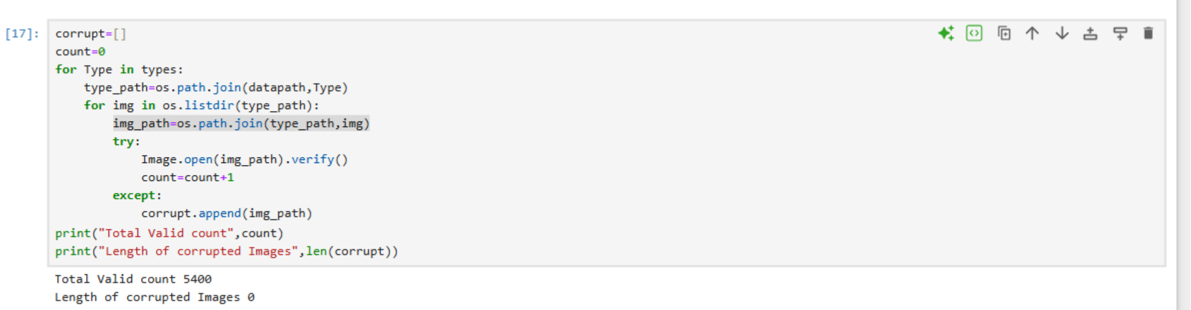
Reason:

img is the name of the file inside **that type folder**, but I have unintentionally joined it with datapath instead of type_path. So it image is not consider and executed except block and append file path in corrupt list

code mistake: img_path=os.path.join(datapath,img)

Corrected:

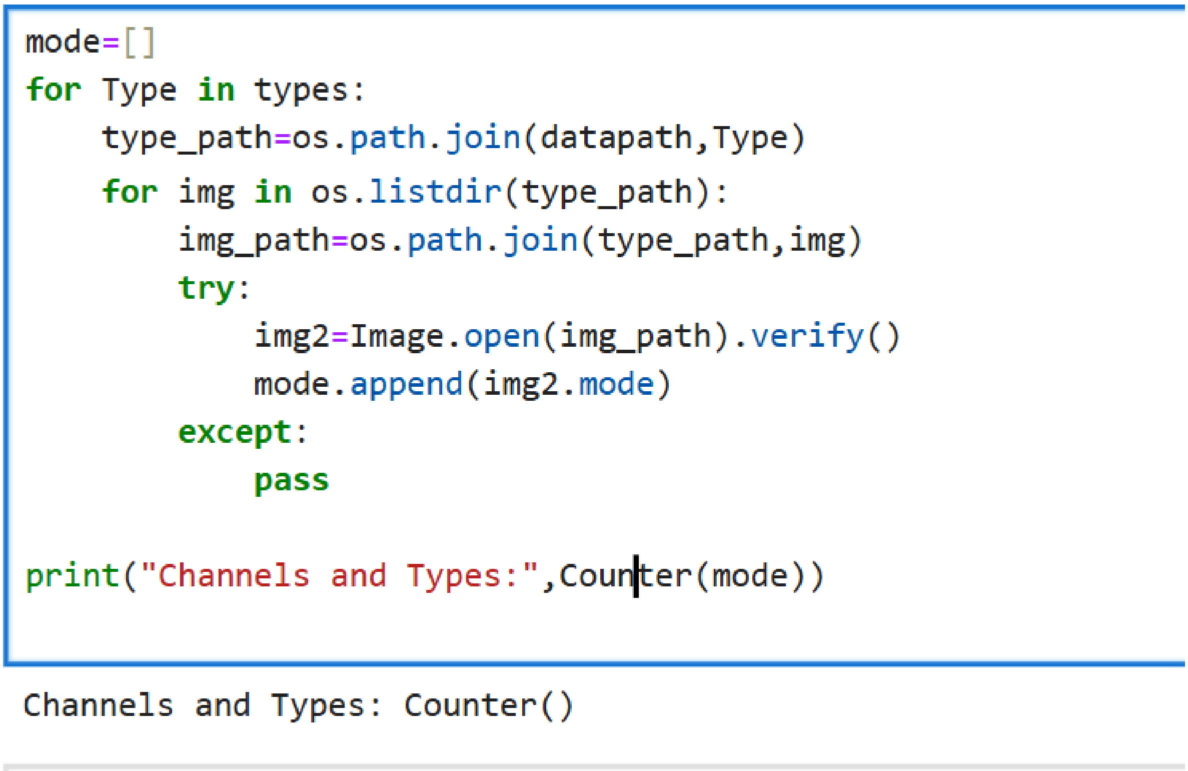
img_path=os.path.join(type_path,img)



```
corrupt=[]
count=0
for Type in types:
    type_path=os.path.join(datapath,Type)
    for img in os.listdir(type_path):
        img_path=os.path.join(type_path,img)
        try:
            Image.open(img_path).verify()
            count=count+1
        except:
            corrupt.append(img_path)
print("Total Valid count",count)
print("Length of corrupted Images",len(corrupt))
```

```
Total Valid count 5400
Length of corrupted Images 0
```

Logical Error-2:



```
mode=[]
for Type in types:
    type_path=os.path.join(datapath,Type)
    for img in os.listdir(type_path):
        img_path=os.path.join(type_path,img)
        try:
            img2=Image.open(img_path).verify()
            mode.append(img2.mode)
        except:
            pass

print("Channels and Types:",Counter(mode))
```

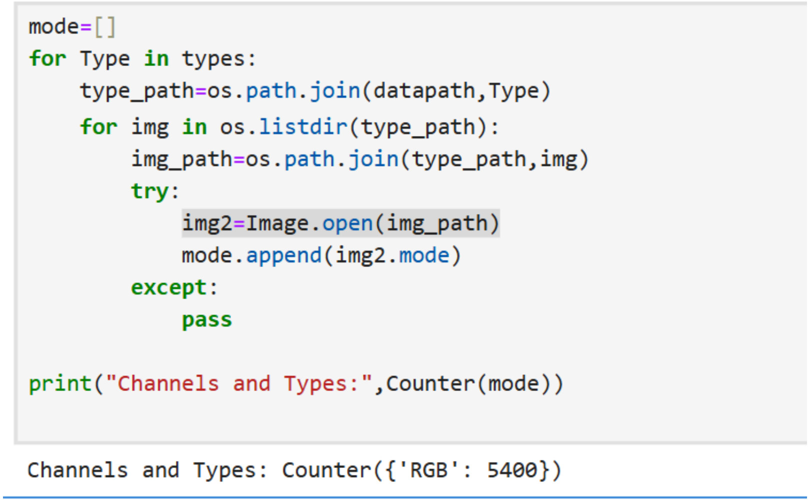
```
Channels and Types: Counter()
```

In this .verify() just verifies whether image is corrupted or not. If not corrupted then it closes image and returns null that’s the reason for the output “Counter()” which is not clear.

img2 = Image.open(img_path).verify()- Wrong

Corrected:

img2=Image.open(img_path)

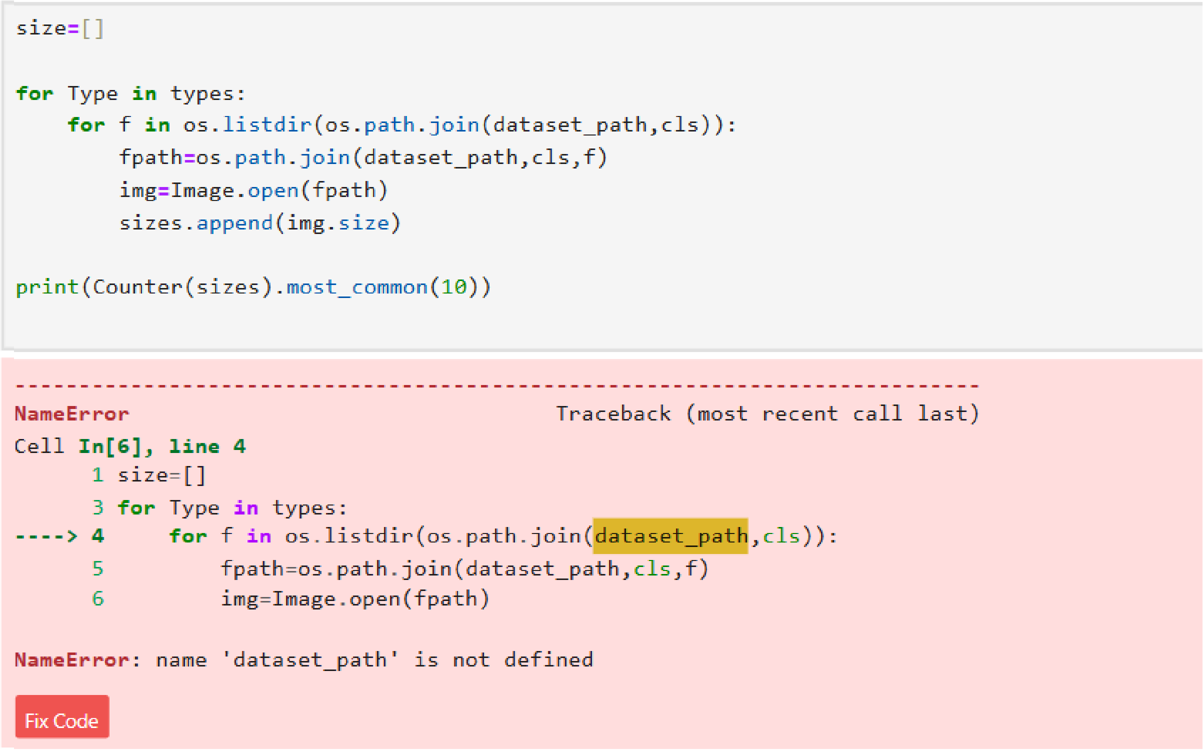


```
mode=[]
for Type in types:
    type_path=os.path.join(datapath,Type)
    for img in os.listdir(type_path):
        img_path=os.path.join(type_path,img)
        try:
            img2=Image.open(img_path)
            mode.append(img2.mode)
        except:
            pass

print("Channels and Types:",Counter(mode))
```

```
Channels and Types: Counter({'RGB': 5400})
```

Syntax-2 error:



```
size=[]

for Type in types:
    for f in os.listdir(os.path.join(dataset_path,cls)):
        fpath=os.path.join(dataset_path,cls,f)
        img=Image.open(fpath)
        sizes.append(img.size)

print(Counter(sizes).most_common(10))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[6], line 4
      1 size=[]
      3 for Type in types:
----> 4     for f in os.listdir(os.path.join(dataset_path,cls)):
      5         fpath=os.path.join(dataset_path,cls,f)
      6         img=Image.open(fpath)

NameError: name 'dataset_path' is not defined
```

Reason: Basic syntax error in using correct the dataset path

Summary:

The purpose of the preprocessing code is to get an image dataset ready for dependable usage in machine learning models. Inconsistencies including corrupted files, inconsistent colour modes, and variable picture sizes are frequently seen in raw image databases.

Using **os.listdir (datapath)**, all of the class folders are first listed. The dataset's animal categories are represented by each folder. The creation of the mapping between folder names and class labels—which will subsequently be transformed into numerical form for training—makes this step crucial.

The code then looks for damaged photos. by using **Image.open(path)** to try to open each file within each class folder in a loop. The programs **verify()** function finds unreadable files. Images that cannot be read are appended to a corrupt list. In order to prevent runtime problems during model training, this step makes sure the dataset is free of any pictures. The algorithm examines the photos' colour modes once the dataset has been cleaned. There are several formats for images, including RGB, RGBA, and greyscale (L). We are able to comprehend the channel distribution by gathering **img.mode** from every file. Finding these variations is crucial before transforming everything into a uniform format since machine learning algorithms require consistent input forms.

Standardisation is the following phase. Depending on the method selected, each image is opened and transformed to either greyscale or RGB. While greyscale simplifies ML models, RGB conversion guarantees three channels. To ensure that every input has the same form, each image is then downsized to a predetermined dimension, such 128 by 128 pixels. Because picture dimensions across datasets are typically quite variable, resizing is one of the most crucial procedures.

Pixel values are normalised from the **0–255 range to 0–1** once the photos have been standardised and transformed into NumPy arrays. Normalisation enhances stability and speeds up model training. All things considered, the preparation code turns an untidy, inconsistent image folder into a clean, consistent dataset that is prepared for machine learning algorithms.