# import numpy as np

# import imageio.v2 as imageio

# import os

# import matplotlib.pyplot as plt

# from PIL import Image

# # image = imageio.imread(".ppm")

# # pil\_image = Image.fromarray(image)

# # pil\_image.show()

# data=[]

# labels=[]

# classes=5

# cur\_path=os.getcwd()

# print(cur\_path)

#

# for i in range(classes):

# path=os.path.join(cur\_path,'Training',str(i))

# images=os.listdir(path)

# for x in images:

# try:

# image=Image.open(path+'\\'+x)

# image=image.resize((30,30))

# image=np.array(image)

# data.append(image)

# labels.append(i)

# except:

# print("err")

# data=np.array(data)

# labels=np.array(labels)

# print(data.shape)

# print(labels.shape)

# for i in range(classes):

# axs[i].imshow(data[i])

# axs[i].axis('off')

# axs[i].set\_title(f'Class {i}')

#

# plt.show()

import numpy as np

import os

from PIL import Image

import matplotlib.pyplot as plt

# data = []

# labels = []

# classes = 5

# cur\_path = os.getcwd()

# print(cur\_path)

#

# # Create a subplot to display images

# fig, axs = plt.subplots(1, classes, figsize=(15, 3))

#

# for i in range(classes):

# path = os.path.join(cur\_path, 'Training','0000'+ str(i))

# images = os.listdir(path)

#

# for x in images:

# try:

# image = Image.open(os.path.join(path, x))

# image = image.resize((30, 30))

# image = np.array(image)

# data.append(image)

# labels.append(i)

# except:

# print("err")

#

# # Convert lists to NumPy arrays

# data = np.array(data)

# labels = np.array(labels)

#

# print("Data shape:", data.shape)

# print("Labels shape:", labels.shape)

#

# # Display images in a grid

# for i in range(5):

# axs[i].imshow(data[i])

# axs[i].axis('off')

# axs[i].set\_title(f'Class {i}')

#

# plt.show()

# import imageio.v2 as imageio

# import os

#

# # Replace 'path\_to\_directory' with the path to the directory containing PPM images

# directory\_path = 'Training/00000'

#

# # List all files in the directory

# files = os.listdir(directory\_path)

#

# # Initialize an empty list to store image data

# image\_data = []

#

# # Loop through each file in the directory

# for file\_name in files:

# # Construct the full path to the file

# file\_path = os.path.join(directory\_path, file\_name)

#

# # Check if the file is a PPM image

# if file\_name.lower().endswith('.ppm'):

# # Read the PPM image using imageio

# image = imageio.imread(file\_path)

#

# # Append the image data to the list

# image\_data.append(image)

#

# # Convert the list of images to a NumPy array

# image\_data = np.array(image\_data)

#

# # Display some information about the imported data

# print(f"Number of images: {len(image\_data)}")

# print(f"Shape of each image: {image\_data[0].shape} (assuming all images have the same shape)")

# import imageio.v2 as imageio

# import os

# import numpy as np

# from skimage.transform import resize

# import matplotlib.pyplot as plt

#

# # Replace 'path\_to\_directory' with the path to the directory containing PPM images

# directory\_path = 'Training/00000'

#

# # List all files in the directory

# files = os.listdir(directory\_path)

#

# # Initialize an empty list to store image data

# image\_data = []

#

# # Loop through each file in the directory

# for file\_name in files:

# # Construct the full path to the file

# file\_path = os.path.join(directory\_path, file\_name)

#

# # Check if the file is a PPM image

# if file\_name.lower().endswith('.ppm'):

# # Read the PPM image using imageio

# image = imageio.imread(file\_path)

#

# # Resize the image to a common shape (e.g., 32x32 pixels)

# image = resize(image, (32, 32), anti\_aliasing=True)

#

# # Append the resized image data to the list

# image\_data.append(image)

#

# # Convert the list of images to a NumPy array

# image\_data = np.array(image\_data)

#

# # Display some information about the imported data

# # print(f"Number of images: {}")

# print(f"Shape of each image: {image\_data[0].shape} (assuming all images have the same shape)")

# num\_images\_to\_display = len(image\_data) # Adjust the number of images to display as needed

#

# for i in range(num\_images\_to\_display):

# plt.imshow(image\_data[i])

# plt.title(f"Image {i}")

# plt.show()

# import os

# import cv2

# import numpy as np

# from skimage import io

# root\_directory = 'Training'

# classes = os.listdir(root\_directory)

#

# images = []

# labels = []

#

# for class\_label in classes:

# class\_path = os.path.join(root\_directory, class\_label)

# for image\_file in os.listdir(class\_path):

# image\_path = os.path.join(class\_path, image\_file)

# image = cv2.imread(image\_path)

# if image is None:

# print(f"Error: Unable to read image {image\_path}")

# continue

# image = cv2.resize(image, (30, 30)) # Resize images as needed

# images.append(image)

# labels.append(int(class\_label)) # Assuming class labels are represented as integers

#

# images = np.array(images)

# labels = np.array(labels)

# root\_directory = 'Training'

# classes = os.listdir(root\_directory)

#

# images = []

# labels = []

#

# for class\_label in classes:

# class\_path = os.path.join(root\_directory, class\_label)

# for image\_file in os.listdir(class\_path):

# if image\_file.lower().endswith(('.ppm', '.png', '.jpg', '.jpeg', '.gif')):

# image\_path = os.path.join(class\_path, image\_file)

# image = io.imread(image\_path)

# images.append(image)

# labels.append(int(class\_label)) # Assuming class labels are represented as integers

# else:

# print(f"Skipped non-image file: {image\_file}")

# images = np.array(images)

# labels = np.array(labels)

# import os

# import numpy as np

# from skimage import io

#

# root\_directory = 'Training'

# classes = [class\_label for class\_label in os.listdir(root\_directory)

# if os.path.isdir(os.path.join(root\_directory, class\_label))]

#

# images = []

# labels = []

#

# for class\_label in classes:

# class\_path = os.path.join(root\_directory, class\_label)

#

# # Add a check to ensure class\_path is a directory

# if not os.path.isdir(class\_path):

# print(f"Skipped non-directory: {class\_path}")

# continue

#

# for image\_file in os.listdir(class\_path):

# image\_path = os.path.join(class\_path, image\_file)

#

# # Skip non-image files

# if image\_file.lower().endswith(('.ppm', '.png', '.jpg', '.jpeg', '.gif')):

# image = io.imread(image\_path)

# images.append(image)

# labels.append(int(class\_label)) # Assuming class labels are represented as integers

# else:

# print(f"Skipped non-image file: {image\_file}")

#

# images = np.array(images)

# labels = np.array(labels)