General Deep Learning Steps for Classification, segmentation and object detection

1. Dataset and DataLoader
2. Transforms
3. Build Model
4. Training and validation
5. Save and Load model
6. Testing

Dataset and DataLoader

Dataset preprocessing

Data preprocessing for 2D images

- Normally we have png, jpeg, tiff dicom for 2D natural and medical images
- Different python libraries used to read these images, OpenCV, skimage, pydicom etc

Data preprocessing for 3D images or volumes

- Normally we have 3D medical images in different format such as .nifiti, .mha,.nrrd
- We have different libraries to read 3D medical images such as SimpleITK, Nibabel etc

Dataset preprocessing

Data preprocessing for 2D images

- Normally we have png, jpeg, tiff dicom for 2D natural and medical images
- Different python libraries used to read these images, OpenCV, skimage, pydicom etc

```
# install cv2 using this command pip install opencv-contrib-python
import cv2
# check open cv version
print(cv2.__version__)
path='C:/Users/aq22/Desktop/kcl2022/Pytorch_basic_tutorials/workshop_Isblue/Test_images/Desmodora sp1_GY_8S1D1.jpg'
img=cv2.imread(path)
############### read gray image
img_grey=cv2.imread(path,0)
img_rgb=cv2.imread(path,1)
```

Data preprocessing

Data preprocessing for 3D images or volumes

Read 3D images or volumes using Nibabel library

```
################################ read dataset using nibabel in nifiti 
format
# install nibabel pip install nibabel
import os
import nibabel as nib
path='C:/Users/aq22/Desktop/kcl2022/Pytorch_basic_tutorials/test_dataset
/001_saed_0000.nii.gz'
imgobj=nib.load(path)
img_array=imgobj.get_fdata()
```

Data preprocessing

Data preprocessing for 3D images or volumes

Read 3D images or volumes using SimpleITK library

```
########### read dataset using SimpleITK in nifti format
## pip install SimpleITK
import SimpleITK as sitk
path='C:/Users/aq22/Desktop/kcl2022/Pytorch_basic_tutorials/test_dataset/001_sa
ed_0000.nii.gz'
img_obj=sitk.ReadImage(path)
img_array=sitk.GetArrayFromImage(img_obj)
```

1. Dataset and DataLoader

Data Loader

All deep learning models require data in batches. We need to create datalaoder to load the dataset in batches.

We have specific object or class function to process the dataset in batches in pytorch and tensorflow.

The dataloader class has main three object or methods inside the dataloader class. These methods are __init__(), __getitem__(),__len()__

The init, getitem and len are special methods that used in dataloader class to get the dataset in batches.

DataLoader

```
import torch
from torch.utils.data import Dataset
class dataset(Dataset):
    def __init__(self,pathdata,transform=None):
        self.pathdata=pathdata
        # initialize dataset here,
        ##### dataset samples are stored in different way
        ## read dataset, get dataset paths in a list

    def __getitem__(self, index):

        ### read dataset from paths one by one
        ########## and convert into tensors
        #### return images and labels in classification,
        ####### return images and segmentation masks in segmentation

        return

    def __len__(self):
        ###### return total length of samples you have to read
        return
```

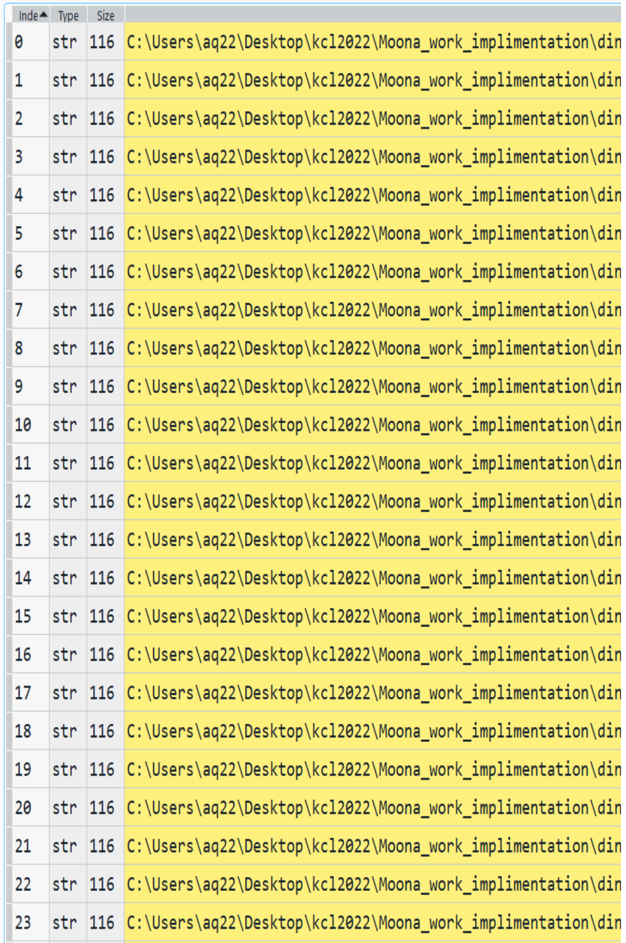
```python
from torch.utils.data import Dataset
import torch
import numpy as np
from skimage import io
import natsort
from glob import glob
import os
class SegmentationDataset(Dataset):
  def __init__(self, dataset, transform):
    self.dataset = dataset
    self.transform = transform
    self.num_classes=4
    self.patht1=natsort.natsorted(glob(os.path.join(self.dataset,'T1','*.png')))
    self.pathseg=natsort.natsorted(glob(os.path.join(self.dataset,'segmentation','*.png')))

  def __len__(self):
    return len(self.pathseg)

  def __getitem__(self, idx):
    patht1= self.patht1[idx]
    pathseg= self.pathseg[idx]
    imgt1=np.array(io.imread(patht1))
    seg=np.array(io.imread(pathseg))
    return imgt1, seg
```

Read the paths in list

Read the image samples one by one from the list (idx start from 0 to until length of your data)



Inde▲	Type	Size	
0	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
1	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
2	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
3	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
4	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
5	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
6	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
7	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
8	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
9	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
10	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
11	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
12	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
13	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
14	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
15	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
16	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
17	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
18	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
19	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
20	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
21	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
22	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din
23	str	116	C:\Users\aq22\Desktop\kcl2022\Moona_work_implimentation\din