import argparse

def args():

"""

Define args that is used in project

"""

parser = argparse.ArgumentParser(description="Pose guided image generation usign deformable skip layers")

parser.add\_argument("--output\_dir", default='output/displayed\_samples', help="Directory with generated sample images")

parser.add\_argument("--batch\_size", default=4, type=int, help='Size of the batch')

parser.add\_argument("--training\_ratio", default=1, type=int,

help="The training ratio is the number of discriminator updates per generator update.")

parser.add\_argument("--l1\_penalty\_weight", default=100, type=float, help='Weight of l1 loss')

parser.add\_argument('--gan\_penalty\_weight', default=1, type=float, help='Weight of GAN loss')

parser.add\_argument('--tv\_penalty\_weight', default=0, type=float, help='Weight of total variation loss')

parser.add\_argument('--lstruct\_penalty\_weight', default=0, type=float, help="Weight of lstruct")

parser.add\_argument("--number\_of\_epochs", default=500, type=int, help="Number of training epochs")

parser.add\_argument("--content\_loss\_layer", default='none', help='Name of content layer (vgg19)'

' e.g. block4\_conv1 or none')

parser.add\_argument("--checkpoints\_dir", default="output/checkpoints", help="Folder with checkpoints")

parser.add\_argument("--checkpoint\_ratio", default=30, type=int, help="Number of epochs between consecutive checkpoints")

parser.add\_argument("--generator\_checkpoint", default=None, help="Previosly saved model of generator")

parser.add\_argument("--discriminator\_checkpoint", default=None, help="Previosly saved model of discriminator")

parser.add\_argument("--nn\_loss\_area\_size", default=1, type=int, help="Use nearest neighbour loss")

parser.add\_argument("--use\_validation", default=1, type=int, help="Use validation")

parser.add\_argument('--dataset', default='market', choices=['market', 'fasion', 'prw', 'fasion128', 'fasion128128'],

help='Market, fasion or prw')

parser.add\_argument("--display\_ratio", default=1, type=int, help='Number of epochs between ploting')

parser.add\_argument("--start\_epoch", default=0, type=int, help='Start epoch for starting from checkpoint')

parser.add\_argument("--pose\_estimator", default='pose\_estimator.h5',

help='Pretrained model for cao pose estimator')

parser.add\_argument("--images\_for\_test", default=12000, type=int, help="Number of images for testing")

parser.add\_argument("--use\_input\_pose", default=True, type=int, help='Feed to generator input pose')

parser.add\_argument("--warp\_skip", default='stn', choices=['none', 'full', 'mask', 'stn'],

help="Type of warping skip layers to use.")

parser.add\_argument("--warp\_agg", default='max', choices=['max', 'avg'],

help="Type of aggregation.")

parser.add\_argument("--disc\_type", default='call', choices=['call', 'sim', 'warp'],

help="Type of discriminator call - concat all, sim - siamease, sharewarp - warp.")

parser.add\_argument("--use\_bg", default=0, type=int, help='Use background images only for prw dataset')

parser.add\_argument("--pose\_rep\_type", default='hm', choices=['hm', 'stickman'],

help='Representation of the pose, hm - heatmap, stickman - like in vunet (https://github.com/CompVis/vunet).')

parser.add\_argument("--cache\_pose\_rep", default=1, type=int, help="Cache pose representation on disk.")

parser.add\_argument("--generated\_images\_dir", default='output/generated\_images',

help='Folder with generated images from training dataset')

parser.add\_argument('--load\_generated\_images', default=0, type=int,

help='Load images from generated\_images\_dir or generate')

parser.add\_argument('--use\_dropout\_test', default=0, type=int,

help='To use dropout when generate images')

args = parser.parse\_args()

args.images\_dir\_train = 'data/' + args.dataset + '-dataset/train'

args.images\_dir\_test = 'data/' + args.dataset + '-dataset/test'

args.annotations\_file\_train = 'data/' + args.dataset + '-annotation-train.csv'

args.annotations\_file\_test = 'data/' + args.dataset + '-annotation-test.csv'

args.pairs\_file\_train = 'data/' + args.dataset + '-pairs-train.csv'

args.pairs\_file\_test = 'data/' + args.dataset + '-pairs-test.csv'

args.bg\_images\_dir\_train = 'data/' + args.dataset + '-dataset/train-bg'

args.bg\_images\_dir\_test = 'data/' + args.dataset + '-dataset/test-bg'

if args.dataset == 'fasion':

args.image\_size = (256, 256)

elif args.dataset == 'fasion128128':

args.image\_size = (128, 128)

else:

args.image\_size = (128, 64)

args.tmp\_pose\_dir = 'tmp/' + args.dataset + '/'

del args.dataset

return args