Improving Object Representations in Low Light Conditions Through Unpaired GAN Based Image Enhancement

This code accompanies the final assignment report for the ComputerVision-KEN4255 course and provides reproducible instructions and data.

Layout

First the layout of the codebase is described which contains the following folders,

```
EnlightenGAN
SSD-Pytorch
Tools
```

Each of these folders contain their own README explaining the process associated with them. This top-level README refers to the README in these folders as and when required.

`EnlightenGAN` contains code, instructions and walkthrough for the GAN based enhancement and shows changes made to accomodate different approaches.

`SSD-Pytorch` contains code used to generate the *source* dataset for the object-context paired training and also contains processing and evaluation scripts to benchmark object detection on different versions of the ExDark dataset (different enhancements can be considered different versions of this dataset.)

`Tools` contains utility scripts and processing scripts to help with converting data to VOC format, transferring data between EnlightenGAN, SSD-Pytorch and ExDark formats and some additional download scripts.

For all the instructions mentioned henceforth, if exact running instructions are not mentioned, refer to the script pointed out by the instruction. Documentation is added within the script on how to use it. Moreover, argparse integration is also provided so `--help` flag can be used on the script for a quick check. (Refer code walkthrough section at the end of README)

Requirements

All code is run and tested using python3.6

To avoid installing requirements for all the subfolders, install global requirements provided here using,

```
pip install -r requirements
```

Preprocessing the ExDark Dataset to PASCAL VOC Format

The Exclusively Dark Dataset was obtained from https://github.com/cs-chan/Exclusively-Dark-Image-Dataset. Images in the dataset are categorized into different folders by object class and each folder then contains all the images corresponding to those objects. The annotations are provided separately with a similar structure. txt files are used to store the annotations.

This dataset first needed to be processed to a format that could easily be used across different tasks. Since object detection was to be performed on the dataset, it was converted to the PASCAL VOC format. The SSD implementation used in this project provides evaluation utilites for the PASCAL VOC format.

Steps to convert the dataset to PASCAL VOC format,

1. The data was downloaded from http://web.fsktm.um.edu.my/~cschan/source/CVIU/ExDark.zip with annotations from https://drive.google.com/file/d/1goqzN0Eg7YqClZfP3cQ9QjENFrEhildz/view?usp=sharing.
2. `batch_copy()` from `Tools/util_fns.py` was used to move all the images and annotations into a flat folder structure as found in VOC. All the images were converted to jpg format using bash commands as the original data contained png and unstreamlined extensions. The annotations are still in txt format but with a flat structure.
3. `Tools/convert_to_voc_annotation.py` was used to generate xml files from ExDark txt annotations for each image with information about objects and bboxes contained in them.
4. A list of trainval and test images was compiled from metadata: https://github.com/cs-chan/Exclusively-Dark-Image-Dataset/blob/master/Groundtruth/imageclasslist.txt provided by the ExDark authors using `create_file_list()` in `Tools/util_fns.py`.

The data is now processed to the VOC format as

```
JPEGImages #Contains all images in jpg format
Annotations #Contains all annotations in xml format same as VOC
ImageSets
    -Main
        -trainval.txt #List of images in trainval
        -test.txt #List of images in test set
```

The processed data in VOC format is made available here: https://drive.google.com/file/d/151j-h_abejzPNglEt3ml-kNrnYvgbm2F/view

The annotations in the new format were tested using https://github.com/penny4860/Pascal-VOC-annotation-viewer to make sure no mistakes were made during the processing. Images from this viewer are presented in the report in Figure 2.

Creating the EnlightenGAN Dataset

Next, *source and target* data was generated to be used in the object-context paired training as described in the report.

The source data was generated from ExDark dataset using the SSD-Pytorch codebase. `SSD-Pytorch/gan_dataset_creator_exdark.py` was used to perform detections over images in the trainval set. Basic detection code was taken from `SSD-Pytorch/detect.py` and adapted to run over images in the ExDark trainval list and then save images with a particular object detected in them to a folder corresponding to that object. (Please refer to the README.md file in the SSD-Pytorch directory on how to setup the code before running these)

The target data was generated from the PASCAL VOC dataset. To download the PASCAL VOC data, the following script can be used: `Tools/download_voc.py`.

Once the VOC dataset was downloaded and extracted, the `Tools/gan_dataset_creator_voc.py` was used to generate the target data.

At the end of this process, the processed GAN training directory has the following format,

```
train_A
    - car
    - bicycle
    ...
train_B
    -car
    -bicycle
    ...
test_A
    ...
test_B
    ...
```

The A and B suffixes refer to source and target pairing.

Training and Testing the EnlightenGAN

Once this dataset is prepared, the EnlightenGAN training process can begin. Refer to the README.md file in the EnlightenGAN directory for more details about different aspects of the training and testing process.

In this project, all models are qualitatively and quantitatively evaluated on the test set of ExDark. The test set is compiled to EnlightenGAN required format using `copy_files_from_voc_list()` from `Tools/utils_fn.py`. This compiles images from test.txt of ExDarkVOC into the folder structure required to test EnlightenGAN. (See the README in the EnlightenGAN folder for more details about this.)

Quantitative Benchmarking using SSD-Pytorch

After having the enhanced images ready, benchmarks are run on them using the evaluation script `SSD-Pytorch/exdark_eval.py`. This is a heavily modified version of the original script. Refer to Running evaluation with SSD section of the README.md in SSD-Pytorch for more information.

Code walkthrough

All the scripts mentioned throughout the README contain some basic documentation on how to use them and have all parameters loaded using argparse. argparse integration was added to all the scripts to make it easy to reproduce. These scripts can be checked for input parameters using `python name_of_script.py --help`. Each argparse argument is provided with help statements to convey their purpose.

Please refer to README within each of the subfolders for more information and detailed walkthrough.

Reproducible Data

All the data used/generated from the experiments are uploaded on Google Drive and can be accessed with the following links, (The instructions throughout describe where and when to use them)

ExDark in VOC Format: https://drive.google.com/open?id=151j-h_abejzPNglEt3ml-kNrnYvgbm2F

EnlightenGAN Object-Context trainval data: https://drive.google.com/open?id=1ghNO6R8UNEoyWy9ugjEQ2pXgZbyVPe-y

Trained GAN Model Weights: https://drive.google.com/open?id=1N4faPXW3OVfUnkSoQ2YLrGtut6aXMfq1

Enhancement comparisons on the ExDark testset: https://drive.google.com/open?id=1A3uj_AGUck3EtKWAxNInkv6wEDzIuPci

Comparisons on DICM Data: https://drive.google.com/open?id=1YbeENPqyYUmjpYXafvYlbMDWviPUFalI

Quantitative Benchmark results: https://drive.google.com/open?id=1RMJmy7_SOta-tpbsamh74vWVzqKhqfLP

Object Detection Samples: https://drive.google.com/open?id=1XxXAUMkGI5dJZodVI4YJMJHje-3ui-CQ