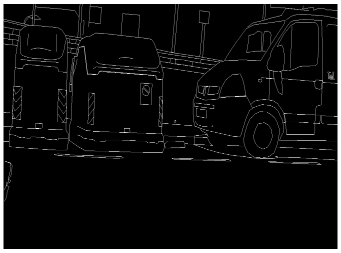
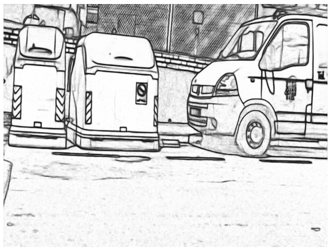
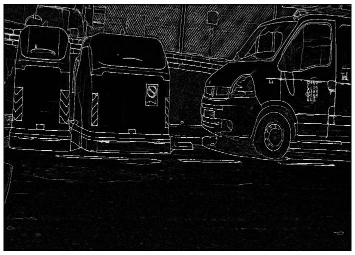
1st_to_2nd_May_Update

1. The following are the Requirements to perform post processing on the predicted edge maps from the DexiNed model before evaluation of error metrics.
    a. Matlab version R2019b
        i. Image Processing Toolbox
        ii. Parallel Processing Toolbox
        iii. Pdollar's Structured Edge Detection Toolbox (source from github)

2. Given below is the basic progression of how the predicted image is post processed before evaluation:

RGB Image	Ground Truth	Predicted edge map	imComplement + nms
			

The RGB image is fed as input to the DexiNed model, which predicts the edge map as seen above.

The next step in the process is to perform imComplement or image complement (to change the edges to white and the rest of the pixels to black) as this is preferable to perform nms operation (Non-maximum Suppression).

Following this edge thinning is performed and the evaluation of the error metric parameters ODS, OIS and AP are performed.

3. Following are the steps taken to prepare the directories for post processing and evaluation:
    a. Clone the pdollar edges repository to get the scripts for edge detection
    b. Clone the pdollar toolbox repo to get the toolbox (note: requires image processing toolbox to be already installed in Matlab)
    c. Create necessary directories
       -> img = contains the RGB images
       -> gt = contains the edge maps (ground truth as provided by BIPED

dataest)
-> pred = contains the predicted edge maps from the inference run on DexiNed model on PyTorch
-> edge_nms = *empty folder - soon will be filled with thinned edge maps after post processing on Matlab*

4. A brief description of files used for error metrics evaluation:
    a. **main_eval.m** -
    -> Adds the necessary paths to the pdollar edge detection toolbox
    -> Performs the nms (Non-maximum Suppression) operation and stores the .png files in edge_nms folder
    -> Function call to edgesEvalDir_x() which returns the computed error metric values

    b. **edgesEvalDir_x.m** -
    -> **Input:** nms_pred_dir, gtDir -> the path to directory containing the post-processed predicted edge maps from the DexiNed model and the ground truth edge maps respectively

    -> **Output:** ODS, OIS, AP

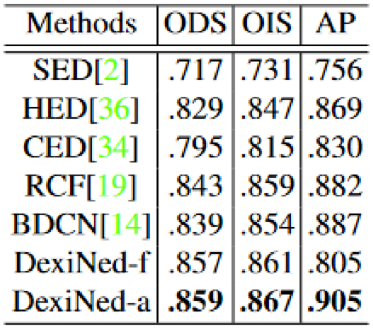
    This function is part of the Pdollar's edge detection toolbox. Some of the lines of code had to be modified as the ground truth values are in .png format instead of .mat format (the latter for which the toolbox was written).

5. Post Processing and Evaluation results on the accurate model

    a. First 2 test images
    -> ODS = 0.8640
    -> OIS = 0.8654
    -> AP = 0.9220

    b. First 5 test images (runtime = ~1hr 30min)
    -> ODS = 0.8706
    -> OIS = 0.8736
    -> AP = 0.9207

    c. The error metrics (over 50 test images of BIPED dataset) as mentioned in the paper for comparison ->



Methods	ODS	OIS	AP
SED[2]	.717	.731	.756
HED[36]	.829	.847	.869
CED[34]	.795	.815	.830
RCF[19]	.843	.859	.882
BDCN[14]	.839	.854	.887
DexiNed-f	.857	.861	.805
DexiNed-a	**.859**	**.867**	**.905**

6. Some useful links:-

a) https://github.com/xavysp/DexiNed - Dexined main repo

b) https://github.com/pdollar/edges - Pdollar edges repository

c) https://github.com/pdollar/toolbox - Pdollar toolbox repository

d) https://github.com/yun-liu/RCF/blob/master/examples/rcf/edge_nms.m - edgenmsmex() function to be used with appropriate parameter values for DexiNed

e) https://github.com/xavysp/DexiNed/issues/30 - code for edge thinning and other useful insights

f) https://github.com/teeso/pytorch_hed/blob/master/eval/bench_bsds500.m

g) https://github.com/xavysp/DexiNed/issues/8

h) https://github.com/xavysp/DexiNed/issues/19

i) https://github.com/teeso/pytorch_hed/tree/master/eval - contains edge_nms.m and bench_bsds.m files with instructions

j) https://github.com/MarkMoHR/Awesome-Edge-Detection-Papers#3-useful-links - Some edge detection papers