
RealSR



Real-World Super-Resolution via Kernel Estimation and Noise Injection (CVPR 2020 Workshops)

Xiaozhong Ji, Yun Cao, Ying Tai, Chengjie Wang, Jilin Li, and Feiyue Huang

Tencent Youtu Lab

Our solution is the **winner of CVPR NTIRE 2020 Challenge on Real-World Super-Resolution** in both tracks.

(Official PyTorch Implementation)

Update - September 1, 2020

- Release training code at Github/Tencent.

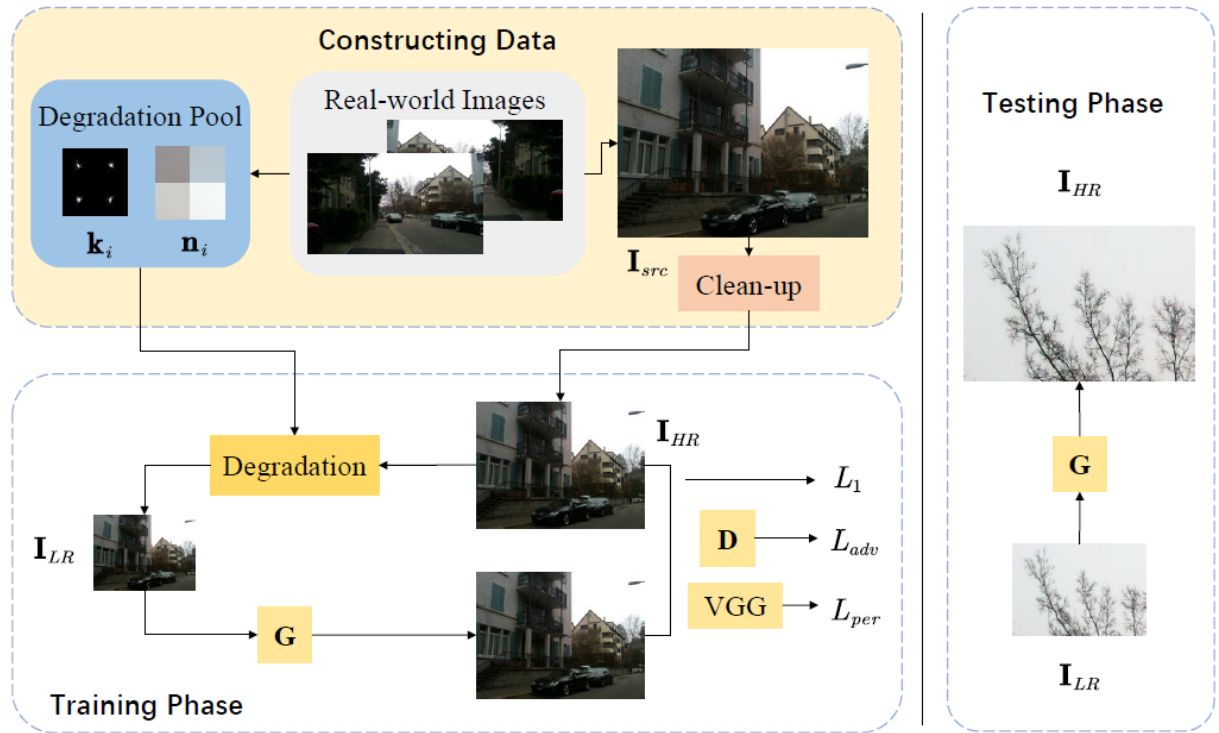
Update - May 26, 2020

- Add DF2K-JPEG Model.
- Executable files based on ncnn are available. Test your own images on windows/linux/macOS. More details refer to realsr-ncnn-vulkan

- Usage - `./realsr-ncnn-vulkan -i in.jpg -o out.png`
- `-x` - use ensemble
- `-g 0` - select gpu id.

Introduction

Recent state-of-the-art super-resolution methods have achieved impressive performance on ideal datasets regardless of blur and noise. However, these methods always fail in real-world image super-resolution, since most of them adopt simple bicubic downsampling from high-quality images to construct Low-Resolution (LR) and High-Resolution (HR) pairs for training which may lose track of frequency-related details. To address this issue, we focus on designing a novel degradation framework for real-world images by estimating various blur kernels as well as real noise distributions. Based on our novel degradation framework, we can acquire LR images sharing a common domain with real-world images. Then, we propose a real-world super-resolution model aiming at better perception. Extensive experiments on synthetic noise data and real-world images demonstrate that our method outperforms the state-of-the-art methods, resulting in lower noise and better visual quality. In addition, our method is the winner of NTIRE 2020 Challenge on both tracks of Real-World Super-Resolution, which significantly outperforms other competitors by large margins.



If you are interested in this work, please cite our paper

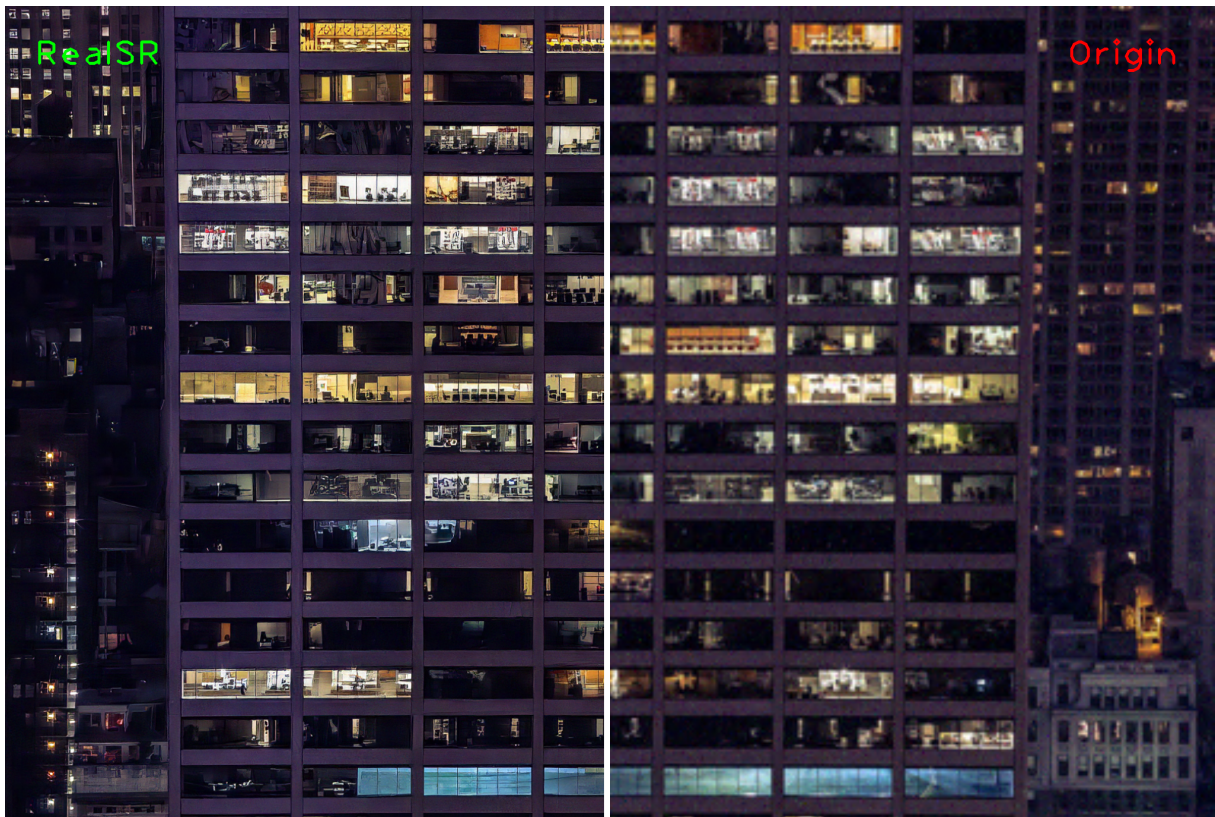
```
1 @InProceedings{Ji_2020_CVPR_Workshops,  
2   author = {Ji, Xiaozhong and Cao, Yun and Tai, Ying and  
3   Wang, Chengjie and Li, Jilin and Huang, Feiyue},  
4   title = {Real-World Super-Resolution via Kernel  
   Estimation and Noise Injection},  
   booktitle = {The IEEE/CVF Conference on Computer Vision
```

```
5         and Pattern Recognition (CVPR) Workshops},
6         month = {June},
7         year = {2020}
    }
```

and challenge report NTIRE 2020 Challenge on Real-World Image Super-Resolution: Methods and Results

```
1 @article{Lugmayr2020ntire,
2     title={NTIRE 2020 Challenge on Real-World Image Super-
3         Resolution: Methods and Results},
4     author={Andreas Lugmayr, Martin Danelljan, Radu Timofte,
5         Namhyuk Ahn, Dongwoon Bai, Jie Cai, Yun Cao, Junyang Chen,
6         Kaihua Cheng, SeYoung Chun, Wei Deng, Mostafa El-Khamy Chiu,
7         Man Ho, Xiaozhong Ji, Amin Kheradmand, Gwantae Kim, Hanseok
8         Ko, Kanghyu Lee, Jungwon Lee, Hao Li, Ziluan Liu, Zhi-Song
9         Liu, Shuai Liu, Yunhua Lu, Zibo Meng, Pablo Navarrete,
10        Micheline Christian, Micheloni Kalpesh, Prajapati Haoyu, Ren
11        Yong, Hyeok Seo, Wan-Chi Siu, Kyung-Ah Sohn, Ying Tai, Rao
12        Muhammad Umer, Shuangquan Wang, Huibing Wang, Timothy
13        Haoning Wu, Haoning Wu, Biao Yang, Fuzhi Yang, Jaejun Yoo,
14        Tongtong Zhao, Yuanbo Zhou, Haijie Zhuo, Ziyao Zong, Xueyi
15        Zou},
16     journal={CVPR Workshops},
17     year={2020},
18 }
```

Visual Results





Quantitative Results Compared with Other Participating Methods

'Impressionism' is our team. Note that the final decision is based on MOS (Mean Opinion Score) and MOR (Mean Opinion Rank).

Team	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	MOS \downarrow
Impressionism (ours), <i>winner</i>	24.67 (16)	0.683 (13)	0.232 (1)	2.195
Samsung-SLSI-MSL	25.59 (12)	0.727 (9)	0.252 (2)	2.425
BOE-IOT-AIBD	26.71 (4)	0.761 (4)	0.280 (4)	2.495
MSMers	23.20 (18)	0.651 (17)	0.272 (3)	2.530
KU-ISPL	26.23 (6)	0.747 (7)	0.327 (8)	2.695
InnoPeak-SR	26.54 (5)	0.746 (8)	0.302 (5)	2.740
ITS425	27.08 (2)	0.779 (1)	0.325 (6)	2.770
MLP-SR	24.87 (15)	0.681 (14)	0.325 (7)	2.905
Webbzhou	26.10 (9)	0.764 (3)	0.341 (9)	-
SR-DL	25.67 (11)	0.718 (10)	0.364 (10)	-
TeamAY	27.09 (1)	0.773 (2)	0.369 (11)	-
BIGFEATURE-CAMERA	26.18 (7)	0.750 (6)	0.372 (12)	-
BMIPL-UNIST-YH-1	26.73 (3)	0.752 (5)	0.379 (13)	-
SVNIT1-A	21.22 (19)	0.576 (19)	0.397 (14)	-
KU-ISPL2	25.27 (14)	0.680 (15)	0.460 (15)	-
SuperT	25.79 (10)	0.699 (12)	0.469 (16)	-
GDUT-wp	26.11 (8)	0.706 (11)	0.496 (17)	-
SVNIT1-B	24.21 (17)	0.617 (18)	0.562 (18)	-
SVNIT2	25.39 (13)	0.674 (16)	0.615 (19)	-
Bicubic	25.48 (-)	0.680 (-)	0.612 (-)	3.050
ESRGAN Supervised	24.74 (-)	0.695 (-)	0.207 (-)	2.300

Team	NIQE \downarrow	BRISQUE \downarrow	PIQE \downarrow	NRQM \uparrow	PI \downarrow	IQA-Rank \downarrow	MOR \downarrow
Impressionism (ours), <i>winner</i>	5.00 (1)	24.4 (1)	17.6 (2)	6.50 (1)	4.25 (1)	3.958	1.54 (1)
AITA-Noah-A	5.63 (4)	33.8 (5)	29.7 (8)	4.23 (8)	5.70 (6)	7.720	3.04 (2)
ITS425	8.95 (18)	52.5 (18)	88.6 (18)	3.08 (18)	7.94 (18)	14.984	3.30 (3)
AITA-Noah-B	8.18 (17)	50.1 (12)	88.0 (17)	3.23 (15)	7.47 (17)	13.386	3.57 (4)
Webbzhou	7.88 (15)	51.1 (15)	87.8 (16)	3.27 (14)	7.30 (15)	12.612	4.44 (5)
Relbmag-Eht	5.58 (3)	33.1 (3)	12.5 (1)	6.22 (2)	4.68 (2)	4.060	-
MSMers	5.43 (2)	38.2 (7)	20.5 (3)	5.22 (5)	5.10 (3)	5.420	-
MLP-SR	6.45 (8)	30.6 (2)	29.0 (6)	6.12 (3)	5.17 (4)	5.926	-
SR-DL	6.11 (5)	33.5 (4)	29.4 (7)	5.24 (4)	5.43 (5)	6.272	-
InnoPeak-SR	7.42 (13)	39.3 (8)	21.5 (4)	5.12 (6)	6.15 (9)	7.716	-
QCAM	6.21 (6)	44.2 (9)	49.6 (9)	4.10 (10)	6.05 (8)	8.304	-
SuperT	6.94 (10)	50.2 (13)	75.1 (11)	4.23 (9)	6.35 (10)	9.612	-
KU-ISPL	6.79 (9)	45.1 (10)	61.6 (10)	3.60 (13)	6.59 (12)	10.152	-
BMIPL-UNIST-YH-1	7.03 (12)	50.2 (14)	81.5 (13)	3.70 (12)	6.66 (13)	12.218	-
BIGFEATURE-CAMERA	7.45 (14)	49.2 (11)	87.1 (14)	3.23 (16)	7.11 (14)	13.784	-
Bicubic	7.97 (16)	52.0 (17)	87.2 (15)	3.16 (17)	7.40 (16)	14.532	6.04 (6)
RRDB	7.01 (11)	51.3 (16)	76.0 (12)	4.06 (11)	6.48 (11)	10.042	6.06 (7)

Qualitative Results Compared with Other Participating Methods

‘Impressionism’ is our team.

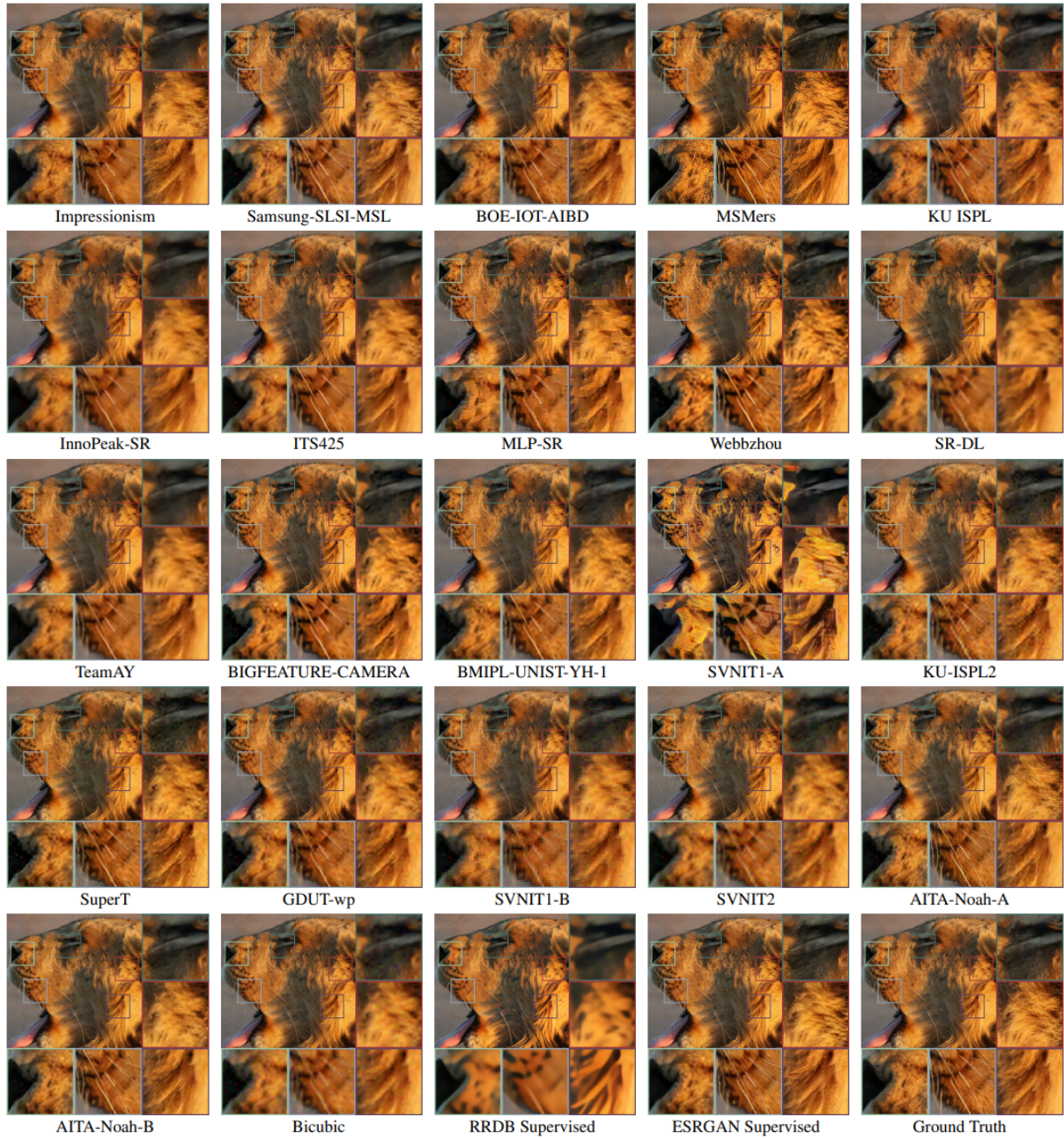


Figure 2. Qualitative comparison between the participating approaches for Track 1. (4× super-resolution)

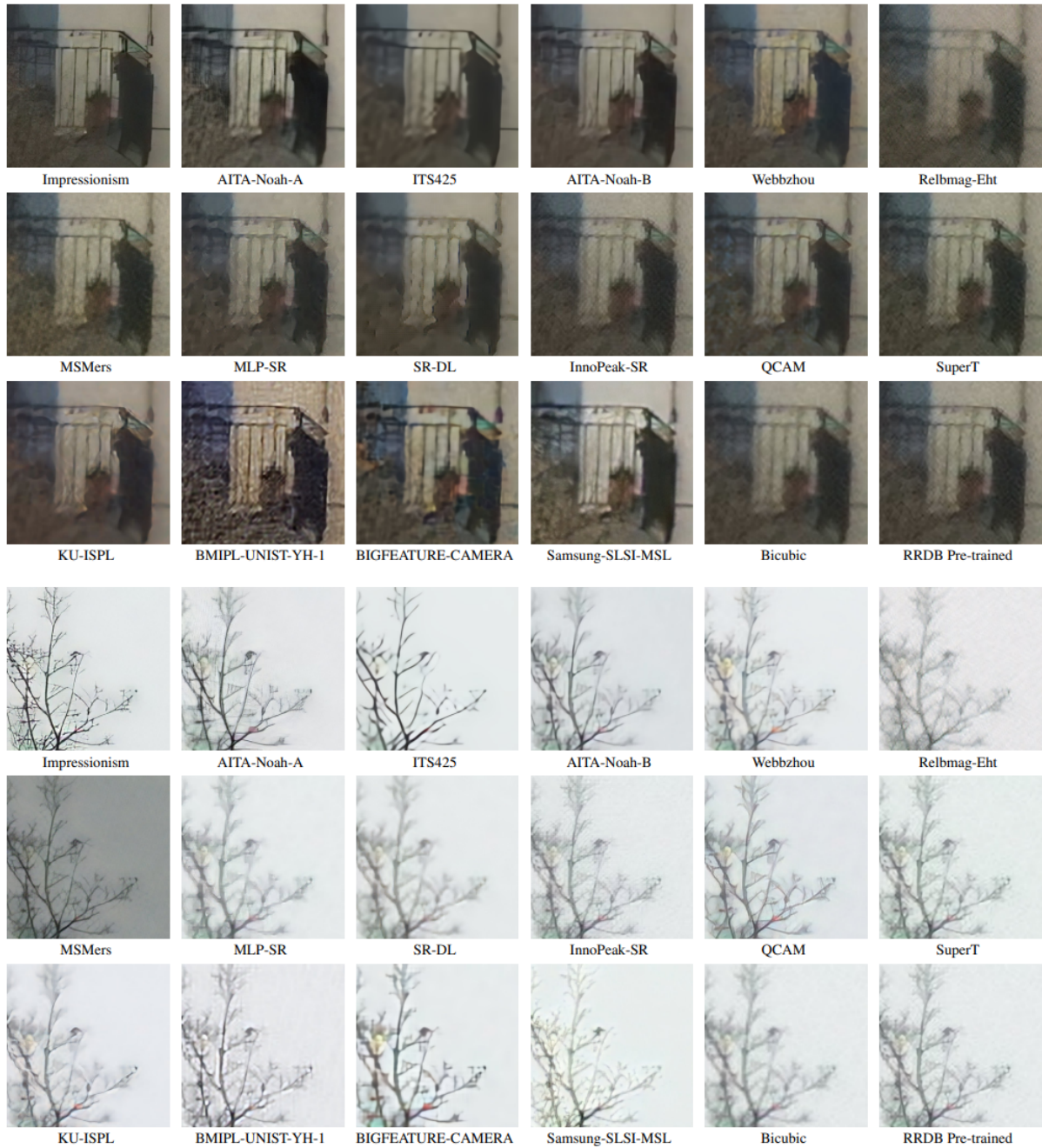


Figure 3. Qualitative comparison between the participating approaches for Track 2. (4 \times super-resolution)

Dependencies and Installation

This code is based on BasicSR.

- Python 3 (Recommend to use Anaconda)

-
- PyTorch ≥ 1.0
 - NVIDIA GPU + CUDA
 - Python packages: `pip install numpy opencv-python lmdb pyyaml`
 - TensorBoard:
 - PyTorch ≥ 1.1 : `pip install tb-nightly future`
 - PyTorch ≤ 1.0 : `pip install tensorboardX`

Pre-trained models

- Models for challenge results
 - DF2K for corrupted images with processing noise.
 - DPED for real images taken by cell phone camera.
- Extended models
 - DF2K-JPEG for compressed jpeg image.

Testing

Download dataset from NTIRE 2020 RWSR and unzip it to your path.

For convenient, we provide Corrupted-te-x and DPEDiphone-crop-te-x.

`cd ./codes`

DF2K: Image processing artifacts

1. Modify the configuration file `options/df2k/test_df2k.yml`
 - line 1 : 'name' – dir name for saving the testing results
 - line 13 : 'dataroot_LR' – test images dir
 - line 26 : 'pretrain_model_G' – pre-trained model for testing
2. Run command : `CUDA_VISIBLE_DEVICES=X python3 test.py -opt options/df2k/test_df2k.yml`
3. The output images is saved in './results/'

DPED: Smartphone images

1. Modify the configuration file options/dped/test_dped.yml
 - line 1 : 'name' – dir name for saving the testing results
 - line 13 : 'dataroot_LR' – test images dir
 - line 26 : 'pretrain_model_G' – pre-trained model for testing
2. Run command : `CUDA_VISIBLE_DEVICES=X python3 test.py -opt options/dped/test_dped.yml`
3. The output images is saved in './results/'

Training

Track 1

1. prepare training data
 - specify dataset paths in './preprocess/path.yml' and create bicubic dataset : `python3 ./preprocess/create_bicubic_dataset.py --dataset df2k --artifacts tdsr`
 - run the below command to collect high frequency noise from Source : `python3 ./preprocess/collect_noise.py --dataset df2k --artifacts tdsr`
2. train SR model
 - Modify the configuration file options/df2k/train_bicubic_noise.yml
 - Run command : `CUDA_VISIBLE_DEVICES=4,5,6,7 python3 train.py -opt options/df2k/train_bicubic_noise.yml`
 - checkpoint dir is in './experiments'

Track 2

1. prepare training data
 - Use KernelGAN to generate kernels from source images. Clone the repo here. Replace SOURCE_PATH with specific path and run : `cd KernelGAN CUDA_VISIBLE_DEVICES=4,5,6,7 python3 train.py --X4 --input-dir SOURCE_PATH`
 - specify dataset paths in './preprocess/path.yml' and generated KERNEL_PATH to kernel create kernel dataset: `python3 ./preprocess/create_kernel_dataset.py --dataset dped --artifacts clean --kernel_path KERNEL_PATH`

-
- run the below command to collect high frequency noise from Source: `python3 ./preprocess/collect_noise.py --dataset dped --artifacts clean`

2. train SR model

- Modify the configuration file `options/dped/train_kernel_noise.yml`
- run command : `CUDA_VISIBLE_DEVICES=4,5,6,7 python3 train.py -opt options/dped/train_kernel_noise.yml`
- checkpoint dir is in `'../experiments'`