NTIRE 2019 Challenge on Real Image Denoising: Methods and Results

Abdelrahman Abdelhamed Radu Timofte Michael S. Brown Songhyun Yu
Bumjun Park Jechang Jeong Seung-Won Jung Dong-Wook Kim Jae-Ryun Chung
Jiaming Liu Yuzhi Wang Chi-Hao Wu Qin Xu Yuqian Zhou Chuan Wang
Shaofan Cai Yifan Ding Haoqiang Fan Jue Wang Kai Zhang Wangmeng Zuo
Magauiya Zhussip Dong Won Park Shakarim Soltanayev Se Young Chun
Zhiwei Xiong Chang Chen Muhammad Haris Kazutoshi Akita Tomoki Yoshida
Greg Shakhnarovich Norimichi Ukita Syed Waqas Zamir Aditya Arora
Salman Khan Fahad Shahbaz Khan Ling Shao Sung-Jea Ko Dong-Pan Lim
Seung-Wook Kim Seo-Won Ji Sang-Won Lee Wenyi Tang Yuchen Fan
Yuqian Zhou Ding Liu Thomas S. Huang Deyu Meng Lei Zhang
Hongwei Yong Yiyun Zhao Pengliang Tang Yue Lu Raimondo Schettini
Simone Bianco Simone Zini Chi Li Yang Wang Zhiguo Cao

Abstract

This paper reviews the NTIRE 2019 challenge on real image denoising with focus on the proposed methods and their results. The challenge has two tracks for quantitatively evaluating image denoising performance in (1) the Bayer-pattern raw-RGB and (2) the standard RGB (sRGB) color spaces. The tracks had 216 and 220 registered participants, respectively. A total of 15 teams, proposing 17 methods, competed in the final phase of the challenge. The proposed methods by the 15 teams represent the current state-of-the-art performance in image denoising targeting real noisy images.

1. Introduction

Image denoising is a fundamental and active research area (*e.g.*, [28, 38, 40, 11]) with a long-standing history in computer vision (*e.g.*, [19, 21]). A primary goal of image denoising is to remove or correct for noise in an image, either for aesthetic purposes, or to help improve other downstream tasks. For many years, researchers have primarily relied on synthetic noisy image for developing and evaluating image denoisers, especially the additive white Gaussian noise (AWGN)—for example, [6, 9, 38]. Recently, more focus has been given to evaluating image denoisers on real noisy images [1, 25]. It was shown that the performance of learning-based image denoisers on real noisy images can be limited if trained using only synthetic noise. Also, hand-engineered and statistics-based methods have been shown to perform better on real noisy images. To this end, we have proposed this challenge as a means to evaluate and benchmark image denoisers on real noisy images.

This challenge is based on the recently released Smartphone Image Denoising Dataset (SIDD) [1] that consists of thousands of real noisy images with their estimated ground-truth, in both raw sensor data (raw-RGB) and standard RGB (sRGB) color spaces. Hence, in this challenge, we provide two tracks for benchmarking image denoisers in both raw-RGB and sRGB color spaces. We present more details on both tracks in the next section.

2. The Challenge

The NTIRE 2019 Real Image Denoising Challenge is aimed to gauge and advance the state-of-the-art in image denoising. The focus of the challenge is on evaluating image denoisers on *real*, rather than synthetic, noisy images. In the following, we present some details about the dataset used in the challenge and how the challenge is designed.

A. Abdelhamed (kamel@eecs.yorku.ca, York University), R. Timofte, and M.S. Brown are the NTIRE 2019 challenge organizers, while the other authors participated in the challenge.
Appendix A contains the authors' teams and affiliations.
NTIRE webpage: http://www.vision.ee.ethz.ch/ntire19/

2.1. Dataset

We used the SIDD dataset [1] for providing training, validation, and testing images for the challenge. The SIDD dataset consists of thousands of real noisy images and their corresponding ground truth, from ten different scenes, captured repeatedly with five different smartphone cameras under different lighting conditions and ISO levels. The ISO levels ranged from 50 to 10,000. The images are provided in both raw-RGB and sRGB color spaces. We believe this dataset is a good fit for benchmarking image denoisers on real noisy images, mainly due to the great extent of variety in noise levels and lighting conditions found in the dataset. Also, this dataset is large enough to provide sufficient training data for learning-based methods, especially, convolutional neural networks (CNNs).

2.2. Challenge Design and Tracks

Tracks We provide two tracks to benchmark the proposed image denoisers based on the two different color spaces: the raw-RGB and the sRGB. Images in the raw-RGB format represent minimally processed images obtained directly from the camera's sensor. These images are in a sensor-dependent color space where the R, G, and B values are related to the sensor's color filter array's spectral sensitivity to incoming visible light. Images in the sRGB format represent the cameras raw-RGB image that have been processed by the in-camera image processing pipeline to map the sensor-dependent RGB colors to a device-independent color space, namely standard RGB (i.e., sRGB). Different camera models apply their own proprietary photo-finishing routines, including several nonlinear color manipulations, to modify the raw-RGB values to appear visually appealing (see [16] for more details). We note that the provided sRGB images are not compressed and therefore do not exhibit compression artifacts. Denoising a raw-RGB would typically represent a denoising module applied within the in-camera image processing pipeline. Denoising an sRGB image would represent a denoising module applied after the in-camera color manipulation. As found in recent works [1, 25], image denoisers tend to perform better in the raw-RGB color space than in the sRGB color space. However, raw-RGB images are far less common than sRGB images which are easily saved in common formats, such as JPEG and PNG. Since the SIDD dataset contains both raw-RGB and sRGB versions of the same image, we found it feasible to provide a separate track for denoising in each color space. Both tracks follow similar data preparation, evaluation, and competition timeline, as discussed next.

Data preparation The provided training data was the SIDD-Medium dataset that consists of 320 noisy images in both raw-RGB and sRGB space with corresponding ground truth and metadata. Each noisy or ground truth image is a 2D array of normalized raw-RGB values (mosaiced color filter array) in the range $[0, 1]$ in single-precision floating point format saved as Matlab .mat files. The metadata files contained dictionaries of Tiff tags for the raw-RGB images, saved as .mat files.

The validation data consisted of 1280 noisy image blocks (*i.e.*, croppings) form both raw-RGB and sRGB images, each block is 256×256 pixels. The blocks are taken from 40 images, 32 blocks from each image ($40 \times 32 = 1280$). All image blocks are combined in a single 4D array of shape $[40, 32, 256, 256]$ where the four dimensions represent the image index, the index of the block within the image, the block height, and the block width, respectively. The blocks have the same number format as the training data. The testing data consisted of 1280 noisy image blocks different from the validation block but following the same format as the validation data. Image metadata files were also provided for the 40 images from which the validation/testing data was extracted.

We also provided the simulated camera pipeline used to render raw-RGB images into sRGB for the SIDD dataset [1]. The provided pipeline offers a set of processing sages similar to an on-board camera pipeline. Such stages include: black level subtraction, active area cropping, white balance, color space transformation, and global tone mapping. Additionally, we provided the noise level estimates of the SIDD images. The range of noise level functions (NLFs) is $[1.1841^{-4}, 2.1949^{-2}]$ for β_1 and $[2.0024^{-06}, 1.7506^{-3}]$ for β_2. For Gaussian σ, the estimated range is $[0.242, 11.507]$ in the space of $[0, 255]$.

Evaluation The evaluation is based on the comparison of the restored clean (denoised) images with the ground-truth images. For this we use the standard peak signal-to-noise ratio (PSNR) and, complementary, the structural similarity (SSIM) index [33] as often employed in the literature. Implementations are found in most of the image processing toolboxes. We report the average results over all image blocks provided.

For submitting the results, participants were asked to provide the denoised image blocks in a multidimensional array shaped in the same way as the input data (*i.e.*, $[40, 32, 256, 256]$). In addition, participants were asked to provide additional information: the algorithm's runtime per mega pixel (in seconds); whether the algorithm employs CPU or GPU at runtime; and whether extra metadata is used as inputs to the algorithm.

At the final stage of the challenge, the participants were asked to submit fact sheets to provide information about the teams and to describe their methods.

[1] https://github.com/AbdoKamel/simple-camera-pipeline

Timeline The challenge timeline was performed in two stages. The validation stage started on January 13, 2019, and lasted for approximately 8 weeks. The final testing stage started on March 12, 2019, and lasted for 12 days. Each participant was allowed a maximum of 20 and 6 submissions during the validation and testing phases, respectively. The challenge ended on March 24, 2019.

3. Challenge Results

From approximately 220 registered participants in each track, 15 teams entered in the final phase and submitted results, codes/executables, and factsheets. Tables 1 and 2 report the final test results, in terms of peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) index [33], for the raw-RGB and sRGB tracks, respectively. The tables show the method ranks based on each measure in subscripts. We present the self-reported runtimes and major details provided in the factsheets submitted by participants. Figures 1 and 2 show a 2D visualization of PSNR and SSIM values for all methods in both raw-RGB and sRGB tracks, respectively. For combined visualization, both figures are overlaid in Figure 3. The methods are briefly described in section 4 and team members are listed in Appendix A.

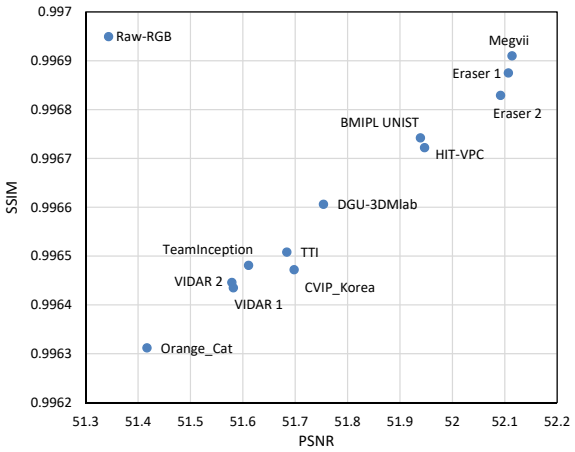

Figure 1: Combined PSNR and SSIM values of method from the raw-RGB track.

Main ideas All of the proposed method are based on deep learning. Specifically, all methods employ convolutional neural networks (CNNs) based on various architectures. Most of adapted architectures are based on widely-used networks, such as U-Net [26], ResNet [14], and DenseNet [15]. The main ideas included re-structuring existing networks, introducing skip connections, introducing residual connections, and using densely connected components.

Most teams used L_2 or L_1 loss as the optimization function while some teams (BMIPL UNIST and Team Inception) adopted a mixed loss between L_1 and multi-scale

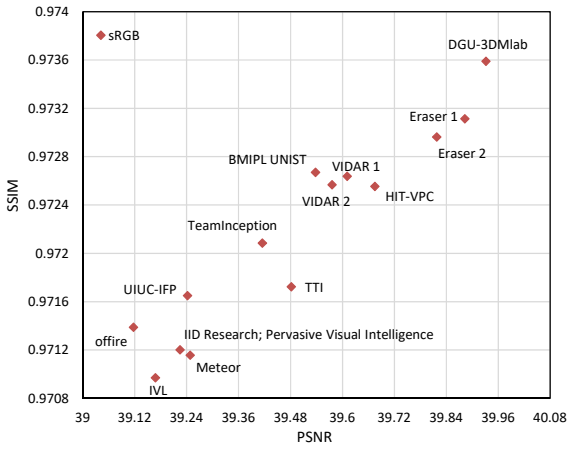

Figure 2: Combined PSNR and SSIM values of method from the sRGB track.

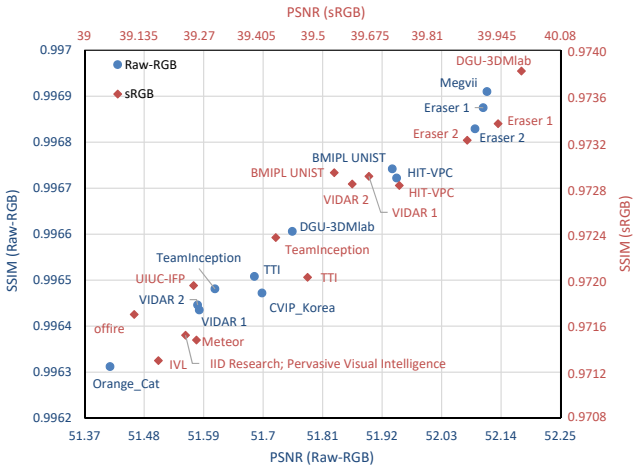

Figure 3: Combined PSNR and SSIM values of all methods from both raw-RGB (in blue) and sRGB (in red) tracks. Note the different axes and scales for each track.

structural similarity (MS-SSIM) [34]. Team "IID Research; Pervasive Visual Intelligence Team" used KL divergence as the optimization function to infer the mean value of a pixel as well as the noise variance at that pixel's location.

Top results The top three methods achieved very close performances, in terms of PSNR and SSIM, with less than 0.02 dB differences in raw-RGB space, as shown in Figure 1, and with less than 0.06 dB differences in the sRGB, as shown in Figure 2. The differences in SSIM values were similarly close, with less than 1×10^{-4} in raw-RGB space and less than 5×10^{-4} in sRGB space. The best performing method for raw-RGB denoising (team Megvii) achieved 52.114 dB PSNR while the best method for sRGB denoising (team DGU-3DMlab) achieved 39.932 dB PSNR. The next best two methods in both tracks were proposed by team Eraser, as shown in Figure 3.

Team	Username	PSNR	SSIM	Runtime (s/Mpixel)	CPU/GPU (at runtime)	Platform	Ensemble	Loss
Megvii	memono11	$52.114_{(1)}$	$0.9969_{(1)}$	0.169	RTX 2080Ti	PyTorch	models ensemble ($\times 8$)	L_1
Eraser	Songsaris	$52.107_{(2)}$	$0.9969_{(2)}$	3.381	RTX 2080Ti	PyTorch	models ($\times 2$), flip/rotate ($\times 8$)	L_1
Eraser	kkbbbj	$52.092_{(3)}$	$0.9968_{(3)}$	~ 2	RTX 2080Ti	PyTorch	models ($\times 2$), flip/rotate ($\times 8$)	L_1
HIT-VPC	opt	$51.947_{(4)}$	$0.9967_{(5)}$	?	GTX 1080Ti	PyTorch	flip/rotate ($\times 8$)	L_1
BMIPL UNIST	BMIPL_denoiser	$51.939_{(5)}$	$0.9967_{(4)}$	3.132	Titan X	TensorFlow/ PyTorch	models ($\times 3$), flip/rotate ($\times 8$)	Mixed (L_1 and MS-SSIM)
DGU-3DMlab	DGU-3DMlab1	$51.754_{(6)}$	$0.9966_{(6)}$	0.8965	Titan Xp	PyTorch	None	L_1
CVIP_Korea	DP_Lim	$51.698_{(7)}$	$0.9965_{(9)}$	1.983	Titan Xp	TensorFlow	None	L_2
TTI	iim_lab	$51.684_{(8)}$	$0.9965_{(7)}$	2	Titan X	PyTorch	flip/rotate ($\times 8$)	L_1
TeamInception	swz30	$51.611_{(9)}$	$0.9965_{(8)}$	0.48	Titan Xp	PyTorch	flip/rotate ($\times 8$)	MSE
VIDAR	ChangC	$51.582_{(10)}$	$0.9964_{(11)}$	0.665	Tesla V100	PyTorch	models ($\times 3$), flip/rotate ($\times 8$)	MSE
VIDAR	eubear	$51.579_{(11)}$	$0.9964_{(10)}$	0.665	Tesla V100	PyTorch	models ($\times 3$), flip/rotate ($\times 8$)	MSE
Orange_Cat	orange_cat	$51.417_{(12)}$	$0.9963_{(12)}$	0.064	GTX 1080Ti	TensorFlow	models ($\times 11$)	L_1

Table 1: Results and rankings of methods submitted to the raw-RGB denoising track.

Team	Username	PSNR	SSIM	Runtime (s/Mpixel)	CPU/GPU (at runtime)	Platform	Ensemble	Loss
DGU-3DMlab	DGU-3DMlab	$39.932_{(1)}$	$0.9736_{(1)}$	0.5577	Titan Xp	PyTorch	None	L_1
Eraser	kkbbbj	$39.883_{(2)}$	$0.9731_{(2)}$	~ 2	GTX 1080Ti	PyTorch	models ($\times 2$), flip/rotate ($\times 8$)	L_1
Eraser	Songsaris	$39.818_{(3)}$	$0.973_{(3)}$	3.416	GTX 1080Ti	PyTorch	models ($\times 2$), flip/rotate ($\times 8$)	L_1
HIT-VPC	opt	$39.675_{(4)}$	$0.9726_{(7)}$	?	GTX 1080Ti	PyTorch	flip/rotate ($\times 8$)	L_1
VIDAR	eubear	$39.611_{(5)}$	$0.9726_{(5)}$	0.903	Tesla V100	PyTorch	models ($\times 3$), flip/rotate ($\times 8$)	MSE
VIDAR	ChangC	$39.576_{(6)}$	$0.9726_{(6)}$	0.903	Tesla V100	PyTorch	models ($\times 3$), flip/rotate ($\times 8$)	MSE
BMIPL UNIST	BMIPL_denoiser	$39.538_{(7)}$	$0.9727_{(4)}$	3.132	Titan X	TensorFlow/ PyTorch	models ($\times s$), flip/rotate ($\times 8$)	Mixed (L_1 and MS-SSIM)
TTI	iim_lab	$39.482_{(8)}$	$0.9717_{(9)}$	~ 2	Titan X	PyTorch	flip/rotate ($\times 8$)	L_1
TeamInception	swz30	$39.415_{(9)}$	$0.9721_{(8)}$	1.136	Titan Xp	PyTorch	flip/rotate ($\times 8$)	MSE
Meteor	loseall	$39.248_{(10)}$	$0.9712_{(13)}$	0.13	Titan X	TensorFlow/ PyTorch	flip/rotate ($\times 8$)	Multi-level L_1
UIUC-IFP	fyc0624	$39.242_{(11)}$	$0.9717_{(10)}$	10.73	GPU	TensorFlow	flip/rotate ($\times 8$)	?
IID Research; Pervasive Visual Intelligence	zsyue	$39.225_{(12)}$	$0.9712_{(12)}$	0.0283	RTX 2080Ti	PyTorch	flip/rotate ($\times 8$)	KL divergence
IVL	Zino	$39.168_{(13)}$	$0.971_{(14)}$	0.02	Titan V	PyTorch	model snapshots ($\times 3$ epochs)	L_1
offire	of-fire	$39.117_{(14)}$	$0.9714_{(11)}$	3.83	Titan Xp	PyTorch	None	L_1

Table 2: Results and rankings of methods submitted to the sRGB denoising track.

Ensembles To boost performance, most of the methods applied different flavours of ensemble techniques. Specifically, most teams used a self-ensemble [30] technique where the results from eight flipped/rotated versions of the same image are averaged together. Some teams applied additional model-ensemble techniques.

Training data Most of the teams relied solely on the training data provided by the SIDD dataset while applying usual data augmentation strategies, such as flipping and rotating images. However, some teams (*e.g.*, Meteor) used additional training data from other datasets, such as DIV2K dataset [29, 2], DSB500 dataset [4], and Waterloo Exploration Database [23].

Conclusions From the analysis of the presented results, we can conclude that the proposed methods achieve state-of-the-art performance in real image denoising on the SIDD benchmark. The methods proposed by the top ranking teams (*i.e.*, DGU-3DMlab, Megvii, and Eraser) achieve

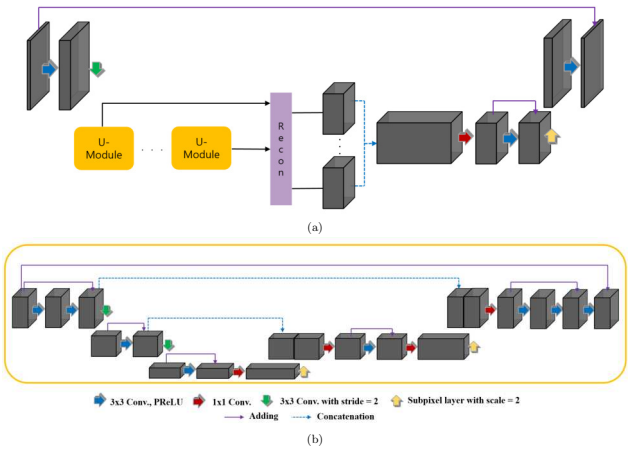

Figure 4: Eraser's DIDN architecture. (a) Overall architecture. (b) U-Module.

consistent performance across both color spaces—that is, raw-RGB and sRGB (see Figure 3).

4. Methods and Teams

4.1. Eraser Team

The Eraser team proposed two methods for image denoising, each has been applied to both raw-RGB and sRGB tracks. Both methods are discussed in the following.

Deep Iterative Down-Up Network (DIDN) for Image Denoising DIDN [37] is a hierarchical network inspired by U-Net [26]. A modified U-Net architecture is used as a module, and multiple modules are connected in a sequential manner. In U-Net module, convolution with stride of 2 is used for down-sampling, and subpixel layer is used for up-sampling. Local and global residual learning are used in the network. The same network structures are used for both of raw-RGB and sRGB tracks, however, the number of modules is different: 9 and 10 modules were used in the raw-RGB track while 8 modules were used for the sRGB track. The proposed network architecture is shown in Figure 4.

The main contributions of this solution are as follows: (1) a modified U-Net architecture for image denoising using local residual learning and changed down-up scaling layers; (2) U-Net modules are sequentially connected at a small resolution, and this repetitive up-down sampling is shown to be effective for denoising; (3) instead of using max-pooling and deconvolution layer, convolution with stride of 2 and subpixel layers are used for down-up sampling stage, therefore minimizing the loss caused by the resolution changes; (4) by reducing the resolution of feature maps by three steps, the computation and memory usage are reduced, which results in the model with a lot of parameters.

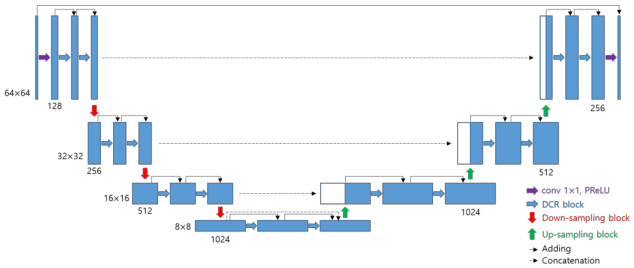

Figure 5: DHDN architecture by Eraser team.

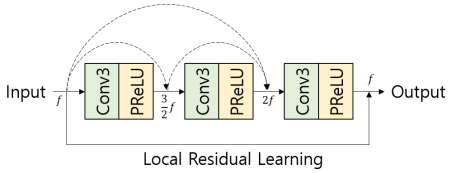

Figure 6: DCR block architecture.

Densely Connected Hierarchical Network for Image Denoising (DHDN) DHDN [24] is inspired by U-Net [26], DenseNet [15], ResNet [14], and Residual Dense Network [43]. Figure 5 shows the architecture of the DHDN. There are two Densely Connected Residual blocks (DCR blocks) in each step. Figure 6 shows the architecture of the DCR block. Feature maps of DCR block are densely connected where growth rate is set to half of the number of the input feature maps. The last convolution layer restores the number of the feature map to apply local residual learning. The input of a down-sampling block is connected to the output of the up-sampling block with dense connectivity. Final convolution layer generates 3 feature maps for sRGB data, and 1 feature map for raw-RGB data. Global residual learning was applied by connecting input image with the output of the final convolution layer.

4.2. DGU-3DMlab Team

The DGU-3DMlab team proposed a **Grouped Residual Dense Network (GRDN) for Image Denoising** [17]. This method uses grouped residual dense units. The network architecture is mainly motivated by the residual dense network (RDN) [43]. The architectures for both raw-RGB and sRGB tacks are shown in Figures 7 and 8, respectively. We made the RDN as a component (denoted as GRDB in Figure 9) and built a network by cascading GRDBs. The RDB Residual connections are applied in three or four different levels (global residual connection, semi-global residual connection, semi-global residual connection in GRDB, and local residual connection in each RDB). Down-sampling and up-sampling layers are included to enable a deeper and wider architecture and the convolutional block attention module (CBAM) [35] is also applied.

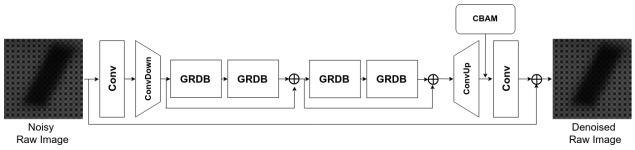

Figure 7: GRDN architecture for raw-RGB track.

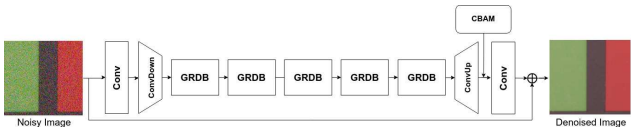

Figure 8: GRDN architecture for sRGB track.

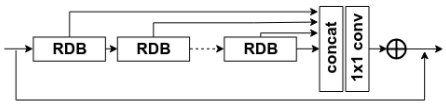

Figure 9: Architecture of GRDB. The architecture of RDB is shown in Figure 10.

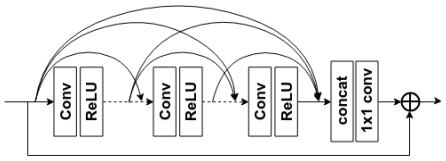

Figure 10: Architecture of RDB.

4.3. Megvii Team

The Megvii team proposed the method **Bayer Pattern Normalization and Bayer Preserving Augmentation** [22] for the raw-RGB track only.The method is based no the U-Net architecture [26]. To preserve the consistency of the network input despite the discrepancy between different input images, the team designed a crafted method of data pre-processing. It enabled the application of the same network to inputs with different Bayer patterns, and thereby to train the network using a larger set of training images without dampening the performance. Also, this method allowed to incorporate data augmentation methods like rotation and flipping, which would alter Bayer patterns if applied directly. These advantage led to a better performance of the obtained network with only a subtle portion of extra running time. The team applied three different strategies: (1) using the given validation set to monitor the training; (2) using only a part of the training set without applying the data augmentation method; (3) applying a mixture of both strategies (1) and (2). The last strategy achieved better results than the other two.

4.4. HIT-VPC Team

The HIT-VPC team proposed the method **Deep Residual U-Net for Image Denoising**. This method is based on the U-Net architecture [26] and ResNet [14]. In order to take advantage of both U-Net and ResNet, a deep residual U-Net (ResUNet) which plugs residual blocks into U-Net is proposed for image denoising. Note that similar idea of combining U-Net and ResNet can also be found in [32, 45]. The architecture of the proposed ResUNet is illustrated in Figure 11. The network has five scales with identity skip connections between encoding and decoding of the same scale. Specifically, an inverse pixel-shuffle downsampling (PixelUnshuffler) [40] is directly applied after the input, and correspondingly, the pixel-shuffle upsampling (PixelShuffler) [27] is adopted before the final denoised output. Such a simple strategy is expected to speed up the inference. For the other downsamplings and upsamplings, 2×2 strided convolution (SConv) and 2×2 transposed convolution (TConv) are respectively used. To exploit the merits of ResNet, a group of 10 residual blocks are adopted in the encoding and decoding of all the scales except the first one. Following [20], each residual block is composed of two 3×3 convolutional layers with ReLU activation in the middle and an identity skip connection summed to its output. The channel number of feature maps in each layers is fixed to 128. Geometric self-ensemble strategy based on flipping and rotation [30] is adopted during testing.

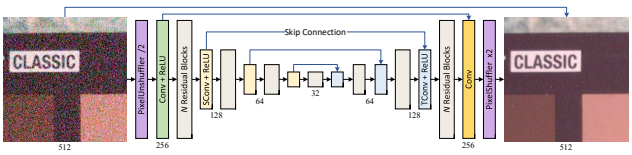

Figure 11: The ResUNet architecture.

4.5. BMIPL UNIST Team

The BMIPL UNIST team presented a study and a method under the title: **Investigation on Deep Neural Network based Denoising Methods with Real Photographs**. In their ensemble strategy, the outputs from different networks were summed together and averaged by the number of networks in the ensemble to get the final denoised image. In this way, they take the advantage of the powerful denoising capabilities of various network architectures. The results revealed that ensemble of networks was always better than individual networks. They used three networks that have significant differences in their architectures and used different combinations of them (ensembles) to get the best performance in raw-RGB and sRGB denoising tracks. These networks are: (1) U-Net [26, 5]; (2) Dual-Domain Denoising Network (D3Net); (3) Down-Up Scaling Network (RCAN-DU).

Dual-Domain Denoising Network (D3Net) The team proposed D3Net to denoise images in spatial and Fourier

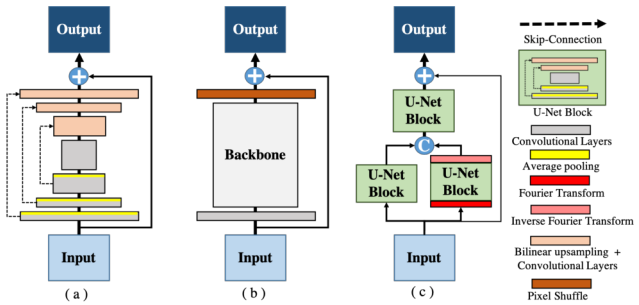

Figure 12: Different Encoder-Decoder structure based CNNs for image denoising. (a) Classic U-Net network (b) Proposed RCAN Down-Up scaling network (RCAN-DU) (c) Proposed Dual-Domain Denoising network (D3Net).

domains. Inspired by the [18], the D3Net strives to preserve high-contrast features such as edges by optimizing convolutional kernels in a spatial domain and reinforces the detail features like textures by learning prior information in a Fourier domain. The D3Net consists of three U-Net blocks with three convolutional layers with non-linearity function PReLU [13] followed by average pooling in encoder part and bilinear upsampling; followed by the three convolutional layers with PReLU in decoding part. The outputs of spatial and transform domain U-Net blocks are concatenated and feed the third U-Net block to obtain residual image. See Figure 12c.

Down-Up Scaling Network (RCAN-DU) The team also proposed another network, shown in Figure 12, that is based on the Residual Channel Attention Network (RCAN) [41]. RCAN has shown a great performance in single image super-resolution problem due to its large receptive field and various skip connections such as long skip connection (LSC) and short skip connection (SSC). The team adopted RCAN as a backbone to the encoder-decoder based network to deal with real noisy sRGB images. The input image firstly downscaled by convolutional layers with stride 2 and then goes to the backbone network. The backbone network consists of a Long Skip Connection and 10 Residual Groups (RG) with the same structure as the existing RCAN. See Figure 13. Each Residual Group consists of a 20 Residual Channel Attention Blocks (RCAB). Unlike the existing residual blocks [14, 20], RCAB does not use batch normalization and uses a channel attention (CA) mechanism to learn global context information. As an up-scaling operator, the team chose Pixel-shuffle method, because of its low computational complexity. In addition, proposed RCAN-DU utilizes skip connection for faster training.

Loss function For raw-RGB denoising, the team used L_1 loss as a cost function. However, for sRGB denoising they used a recently proposed mix loss [46]. The mix loss is

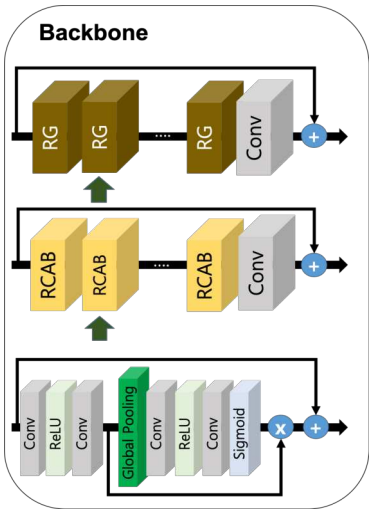

Figure 13: Backbone structure of the proposed RCAN-DU network.

a combination of L_1 and a multi-scale structural similarity (MS-SSIM) losses and can be formulated as:

$$\mathcal{L}_{\text{Mix}}(\theta) = \alpha\, \mathcal{L}_{\text{MS-SSIM}} + (1-\alpha)\, \mathcal{L}_{\text{L}_1} \qquad (1)$$

where θ is a set of network parameters and $\alpha = 0.78$.

4.6. VIDAR Team

The VIDAR team proposed two methods for blind and non-blind image denoising and applied both methods to both raw-RGB and sRGB tracks.

The method **Blind Real-World Image Denoising with Deep Boosting** is a deep boosting framework for blind real-world image denoising based on the team's previous publication [7]. A series of ensemble methods, such as model average, self-ensemble, and model-ensemble, have been adopted to promote the overall performance. An overview of the framework is shown in Figure 14.

The method **Non-blind Real-world Image Denoising with Deep Boosting** is also based on the framework in [7]. This method is a deep boosting framework for non-blind real-world image denoising. Generally, when the training data is sufficient for a learning-based method, a non-blind denoising model would perform better than a blind one. However, due to the lack of enough training data in the real-world condition, it is difficult to directly train separate models for each camera setting. To address this issue, the team first trains a blind model from scratch. Then, based on the ISO and smartphone type information in metadata, they select specific training data to fine-tune the pre-trained model for each setting, which performs a non-blind transfer for better performance. More implementation details, ablation results, and reproducible codes are available at https://github.com/ngchc/deepBoosting.

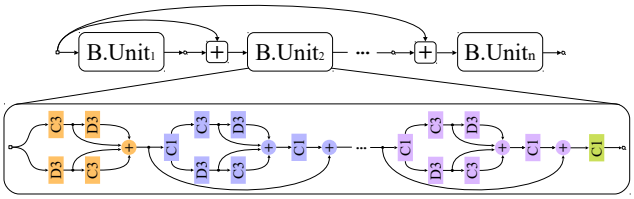

Figure 14: CNN-based deep boosting framework. B.Unit$_n$ denotes the n^{th} boosting unit in the framework, which is implemented by Dilated Dense Fusion Network (DDFN). "+" in a rectangle denotes element-wise summation. For each boosting unit, "C" and "D" denote convolution and its dilated variant, respectively. "1" and "3" denote the kernel size. "+" in a circle denotes channel-wise concatenation. Each layer, except the last, adopts ReLU activation (omitted here for simplicity).

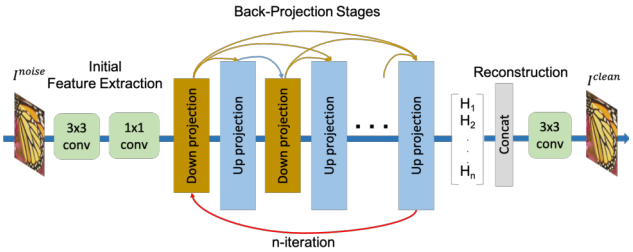

Figure 15: Architecture of DBPN implemented by TTI team.

4.7. TTI Team

The TTI team proposed a method inspired by their prior work on **Deep Back-Projection Networks (DBPN)** [12]. The team constructed iterative down-up projection units. This idea is based on the assumption that down-projection unit can be used to remove the noise by downscaled the feature-maps. Then, the up-projection unit is used to upscale the feature-maps back to the original resolution. We use error feedbacks from the up- and down-scaling steps to guide the network to achieve optimal result as shown in Figure 15.

4.8. TeamInception

TeamInception proposed a deep residual network with spatial and depth-wise attention. The complete framework is shown in Figure 16. The method has three main building blocks: (1) encoder (2) detail decomposition module (DDM), and (3) decoder. Inspired from the work of [42] on super resolution, the proposed method is recursive in nature. The main idea is to gradually remove the noise signal from the image signal while preserving edges, texture and colors. At the entry point of the RDAN network, an encoder is employed to extract features at multiple scales. Then, the DDM progressively separate the information related to the desired clean image from the input noisy image.

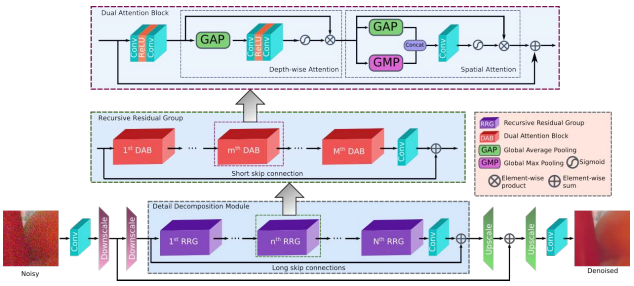

Figure 16: Architecture of RDAN proposed by TeamInception.

Features that are less important get suppressed at the DAB, and only useful information is propagated onward. To discern such features, the team applied two types of attention mechanisms in DAB: depth-wise attention, and spatial attention. Finally, the decoder receives deep features from the last RRG, applies upscaling and pixel shuffling, and yields the final image with the same resolution as of the input noisy image.

Loss Function The team used MSE loss in the raw-RGB track and a mixed loss function, similar to Equation 1, in the sRGB track.

4.9. CVIP_Korea Team

The CVIP_Korea team proposed a **Deep Factorized Network for image denoising (DeFNet)**. The proposed network mainly consists of two parts: the main denoiser and post processing unit which are composed of several factorized sub-networks. Unlike conventional image processing networks based on deep learning [8, 38, 39], these factorized residual sub-networks are expected to act ensemble-like and learn different kinds of residual data for the final goal [31]. The main denoiser consists of a series of 10 sub-denoisers. Each sub-denoiser, as shown in Figure 17, estimates residual information (*i.e.*, noise component) in the clean image inferred on the current step and accumulates the estimated residuals so far, and transfer it to the following sub-denoiser. In addition, similar to recurrent neural network utilizing the hidden states to infer some meaningful information acquired from previous step to next step, our proposed architecture also uses hidden state to transfer information acquired from current sub-denoiser to next sub-denoiser. The final residual information is estimated by accumulating entire residual information estimated from each sub-denoiser. Likewise, the post processing unit, shown in Figure 19, consists of three compensators which also consist of series of 10 sub-compensators. Each sub-compensator, as shown in Figure 18, has similar but lightweight structure with sub-denoiser and has an additional input, an acquired latent clean image from main

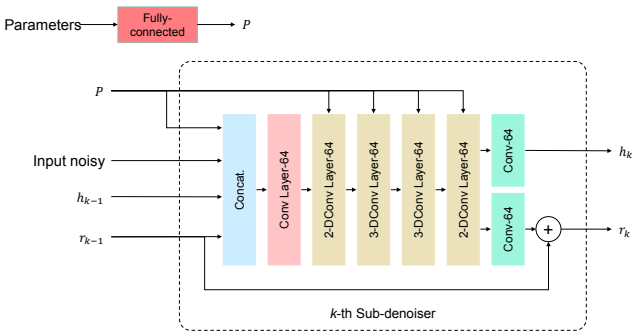

Figure 17: Structure of a sub-denoiser used in DeFNet.

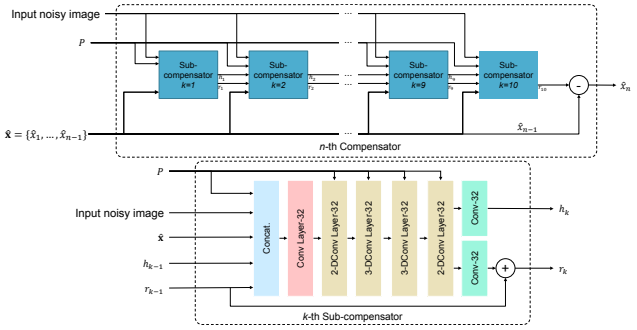

Figure 18: Structure of a compensator and a sub-compensator used in DeFNet.

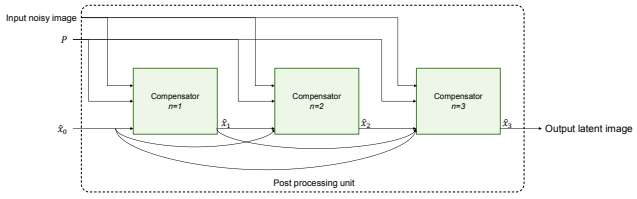

Figure 19: Structure of a post-processing unit used in DeFNet.

denoiser. We found that each sub-network (*i.e.*, the sub-denoiser and the sub-compensator) estimates different type of data for the final goal and transfers it to next sub-network while corresponding each other via hidden state. Therefore, the entire network can robustly acquire desired results by accumulating the data estimated from each sub-network. The network architecture is illustrated in Figure 20. The team used the same described architecture for both raw-RGB and sRGB tracks, only the input data was different.

4.10. Meteor Team

The Meteor team proposed the method **A Multi-Level Network for Real Image Denoising (MLDN)**, motivated by the strategy that noise can be smoothed and easier to remove when images are down-scaling to a smaller size. The team applied a multi-level architecture, which begins to denoise down-sampled images and progressively grows

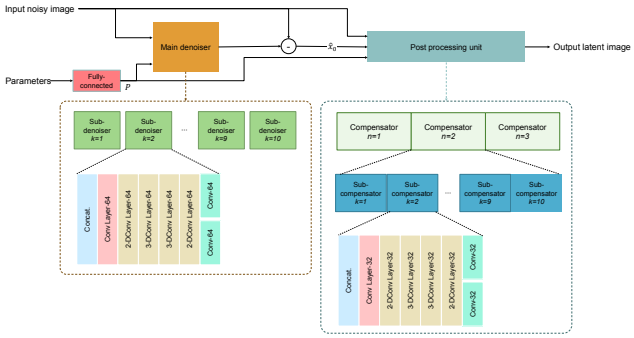

Figure 20: Architecture of DeFNet.

up to denoise the full-size image. The model diagram is illustrated in the Figure 22. The team built a 4-level architecture, each level contains a noise-remover and an up-sampler, except for the last level which replaces the up-sampler by a single convolution layer. The input images are down-sampled by averaging pooling layer at the power of 2: 1/8, 1/4, and 1/2 for level 1–3, respectively. At the 1st level, the team used a stacked CNN-BN-ReLU architecture [38] as the noise extractor (NE) to estimate noise for the 1st level. The extracted noise is then concatenated to the 1/8 down-sampled noisy images and fed into the 1st level. Outputs from the previous level are concatenated to the next level's inputs. The team used outputs of the 4th level as the final denoised images.

Each level has a local loss function $\text{Loss}_i = \|y_i - \hat{y}_i\|_1$. The loss for noise extractor is L_2 between estimated noise and down-sampled ground-truth noise. Eventually the total loss is expressed as:

$$\text{Loss} = \|n - \hat{n}\|_2 + \sum_{i=1}^{L} \lambda_i \|y_i - \hat{y}_i\|_1 , \qquad (2)$$

where $\|x\|_p$ is the L_p norm for x and L is the total number of levels. λ_i is the weight for level i, and we empirically set all λ_i to 1.0. The detail structure of noise-remover is shown in the Figure 21. The team adapted modules from CARN [3] and RCAN [42]. Two cascade blocks are stacked for each level. The cascade block is composed of three residual dense blocks and one channel attention layer. A pixel shuffle layer in levels 1–3 is responsible for up-sampling images by a factor of 2. A block diagram of the MLDN architecture is shown in Figure 22.

4.11. UIUC-IFP Team

The UIUC-IFP team proposed the method **Image Denoising with Kernel and Residual Prediction Networks**. The method aims to predicts both kernels and residuals. In the networks, the team used multiple wide activated residual blocks proposed by [36, 10], then the feature maps are branched to kernels and residuals. The predicted kernels

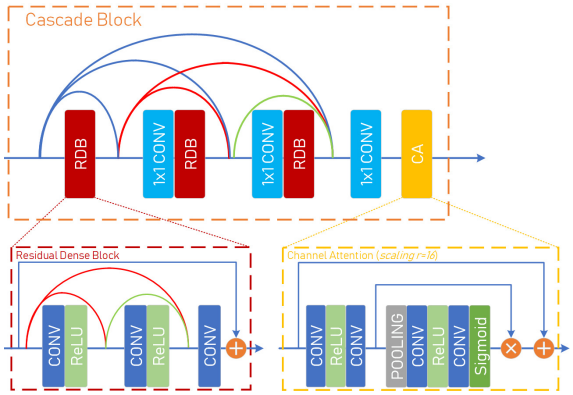

Figure 21: The structure of a cascade block used in the MLDN architecture.

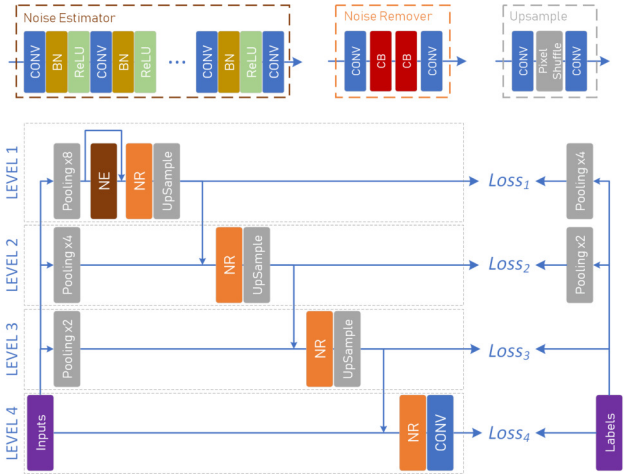

Figure 22: A block diagram of the MLDN architecture proposed by Meteor team.

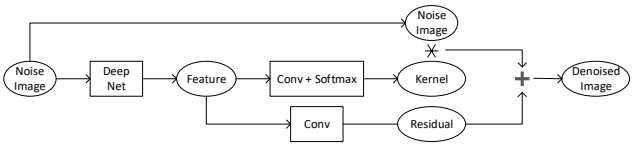

Figure 23: Kernel and Residual Prediction Network proposed by UIUC-IFP team.

will apply to the original noisy image and added to residuals. The network structure is illustrated in Figure 23.

4.12. IID Research; Pervasive Visual Intelligence Team

The team proposed a method for **Fast and Felxible Blind sRGB Image Denosing**. The team denote the real ground-truth image as $\mathbf{Z}$ and the estimated ground-truth image (*i.e.*, the almost noise-free image) as $\mathbf{X}$, The team assume the ground-truth $\mathbf{Z}$ follows a Gaussian distribution with mean $\mathbf{X}$ and a small variance—that is, $z_{ij} \sim \mathcal{N}(x_{ij}, \epsilon_0^2)$. The goal is to learn this distribution from large amounts of observed noisy images $\mathbf{Y}$ under the KL divergence measurement through neural network. The Gaussian distribution's variance is set to 10^{-1} in all of the experiments since the almost noise-free image $\mathbf{X}$ is very close to the real ground-truth $\mathbf{Z}$.

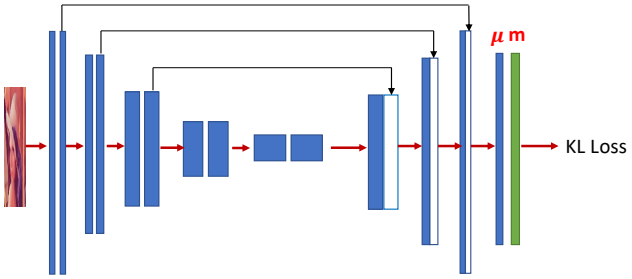

Figure 24: A block diagram of the network proposed by IID Research team.

The team participated in the sRGB denoising track. Instead of directly predicting the denoised image as most of other methods, they regard the real ground-truth image as an stochastic variable with Gaussian distribution, and learn its mean and variance parameters employing a deep neural network. Thus, for one noisy image, instead of outputting the mean value of a Gaussian distribution as the final denoised result, the neural network also outputs a predicted variance (noise level) pixel-wisely. The proposed network is shown in Figure 24. The team adopted the KL divergence as a loss function

$$\arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \sum_{i,j} D_{\mathrm{KL}}(\mathcal{N}(\mu_{ij}, \sigma_{ij}^2) \,||\, \mathcal{N}(x_{ij}, \epsilon_0^2)), \qquad (3)$$

which is equal to

$$\arg\min_{\boldsymbol{\theta}} \frac{(\mu_{ij}(\boldsymbol{\theta}) - x_{ij})}{2\epsilon_0^2} + \frac{1}{2}\left(\frac{m_{ij}^2(\boldsymbol{\theta})}{\epsilon_0^2} - 1 - \log\frac{m_{ij}^2(\boldsymbol{\theta})}{\epsilon_0^2}\right), \qquad (4)$$

where $\boldsymbol{\theta}$ is the network parameters, $\mu_{ij}(\boldsymbol{\theta})$ and $m_{ij}^2(\boldsymbol{\theta})$ are the outputs of the network.

4.13. Orange_Cat Team

The Orange_Cat team proposed a **Pyramid Image Denoising Network (PIDNet)** for track 1, raw-RGB Denoising. The whole network is comprised of three modules, including Noise Estimation Module, Pyramid Pooling Denoising Module and Channel Attention Module. In the Noise Estimation Module, the noise level map of the input image can be estimated with a fully convolutional network, which is learned with the constraint of the smoothness of noise level map. Based on the noise level map and the input image, a 5-level pyramid module is utilized to get global and local contextual information, then five different U-Nets,

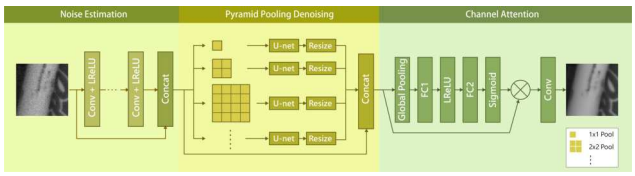

Figure 25: Orange_Cat team's PIDNet architecture.

in parallel, process noisy image blocks, then the results are upsampled to the same size as the original input image and concatenated together. After that, the last stage uses channel attention mechanism to adaptively recalibrate the weight of each channel for better fusion, and finally outputs the clean image blocks. The specific structure of the PIDNet is shown in Figure 25.

To constrain the smoothness of the estimated noise level map, the team adopted a total variation (TV) regularizer:

$$\mathcal{L}_{\mathrm{TV}} = \left\|\nabla_h \phi(x)\right\|^2 + \left\|\nabla_v \phi(x)\right\|^2, \tag{5}$$

where ∇_h and ∇_v represent the gradient operator along the horizontal and vertical directions, respectively, x denotes the input noisy image, ϕ is the weight of the Noise Estimation Module. For the image denoising loss, the team used L_1 loss to supervise the restoration of degraded image content, for the output $\hat{y}$ of the whole network:

$$\mathcal{L}_{\mathrm{denoise}} = \left\|y - \hat{y}\right\|. \tag{6}$$

The overall cost function for training the Pyramid Image Denoising Network is given by

$$\mathcal{L} = \mathcal{L}_{\mathrm{denoise}} + \lambda_{\mathrm{TV}} \mathcal{L}_{\mathrm{TV}}, \tag{7}$$

where λ_{TV} is the tradeoff parameter for the TV regularizer.

4.14. IVL Team

The IVL team proposed a **Deep Residual Autoencoder for Image Denoising**. The proposed model for the image denoising is based on [47]. The proposed method works on YCbCr noisy images and gives as output restored YCbCr images, so the pre-processing step converts the input RGB images into YCbCr color space, and the post-processing step converts the result back into the RGB color space. The entire solution is made of two autoencoder neural networks: the first one is used for the restoration of the luma channel (Y channel), while the second one restores the chroma components (Cb and Cr channels) of the images, using the restored luma channel as a "structure map" to guide the reconstruction. Differently from the original method that was designed for JPEG restoration [47], for the denoising task both the first and second networks consist of B = 5 Residual-in-Residual Dense Blocks (RRDBs), the input mini-batches size equal to 8, made of 100×100 pixel crops taken from

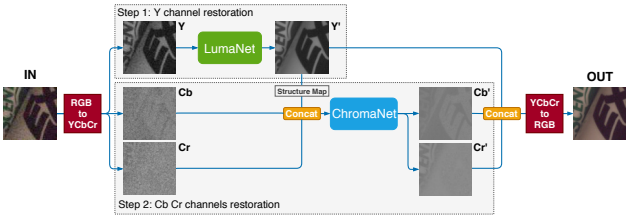

Figure 26: IVL team's Deep Residual Autoencoder.

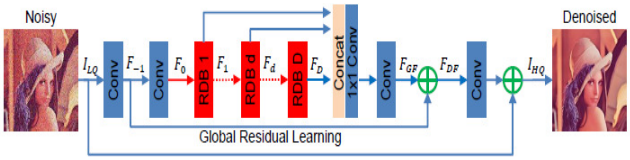

Figure 27: offire team's residual dense network (RDN).

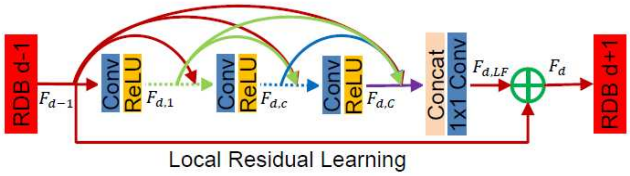

Figure 28: offire team' residual dense block (RDB).

the training dataset. To increase the number of structures and textures seen by the network, online data augmentation (random flipping and rotation) has been applied to the input training crops, during training phase. The model architecture is shown in Figure 26.

4.15. offire Team

The offire team proposed a **Residual Dense Network for sRGB Denoising**. The team utilizes the similar network structure as residual dense networks (RDN) [44], which takes advantage of local and global feature fusion to obtain noise-free images. The implemented network consists of 16 residual dense blocks (RDB), where 8 convolutional layers are densely connected to extract abundant local features. Further more, global residual fusion strategy is involved to jointly and adaptively learn global hierarchical features. The whole network is optimized with L_1 loss function, which has been demonstrated to be more powerful for performance and convergence in image restoration tasks. The network architecture is shown in Figure 27 and the RDB structure is shown in Figure 28. The team participated in the sRGB track.

Acknowledgements

We thank the NTIRE 2019 sponsors: Nvidia, Huawei, Samsung, Oppo, MediaTek, Amazon and ETH Zurich. Abdelrahman Abdelhamed is partially supported by an AdeptMind Scholarship.

A. Teams and Affiliations

NTIRE 2019 Team

Title: NTIRE 2019 Challenge on Real Image Denoising
Members:
Abdelrahman Abdelhamed[1] (kamel@eecs.yorku.ca),
Radu Timofte[2] (radu.timofte@vision.ee.ethz.ch),
Michael S. Brown[1] (mbrown@eecs.yorku.ca)
Affiliations:
[1] York University, Canada
[2] ETH Zurich, Switzerland

Eraser Team

Title: Deep Sequential U-Net for Single Image Denoising, Densely Connected U-Net
Members: Songhyun Yu (3069song@naver.com), Bumjun Park, Jechang Jeong
Affiliations: Hanyang University, Seoul, Korea.

DGU-3DMlab

Title: GRDN: Grouped Residual Dense Network for Image Denoising
Members: Seung-Won Jung (swjung83@gmail.com), Dong-Wook Kim, Jae-Ryun Chung
Affiliations: Dongguk University, South Korea

Megvii

Title: Bayer Pattern Normalization and Bayer Pattern Persevering Augmentation
Members: Jiaming Liu (liujiaming@megvii.com), Yuzhi Wang, Chi-Hao Wu, Qin Xu, Yuqian Zhou, Chuan Wang, Shaofan Cai, Yifan Ding, Haoqiang Fan, Jue Wang
Affiliations: Megvii

HIT-VPC

Title: Deep Residual U-Net for Image Denoising
Members: Kai Zhang (cskaizhang@gmail.com), Wangmeng Zuo
Affiliations: Harbin Institute of Technology, China

BMIPL UNIST

Title: Investigation on Deep Neural Network based Denoising Methods with Real Photographs
Members: Magauiya Zhussip (mzhussip@unist.ac.kr), Dong Won Park, Shakarim Soltanayev, Se Young Chun
Affiliations: Ulsan National Institute of Science and Technology (UNIST), South Korea

VIDAR Team

Title: Real-World Image Denoising with Deep Boosting
Members: Zhiwei Xiong (zwxiong@ustc.edu.cn), Chang Chen
Affiliations: University of Science and Technology of China

TTI

Title: Deep Back-Projection Networks
Members: Muhammad Haris (mharis@toyota-ti.ac.jp), Kazutoshi Akita, Tomoki Yoshida, Greg Shakhnarovich, Norimichi Ukita
Affiliations: Toyota Technological Institute (TTI) and Toyota Technological Institute at Chicago (TTIC)

TeamInception

Title: Image Denoising via Recursive Residual Network with Dual Attention
Members: Syed Waqas Zamir (waqas.zamir@inceptioniai.org), Aditya Arora, Salman Khan, Fahad Shahbaz Khan, Ling Shao
Affiliations: Inception Institute of Artificial Intelligence (IIAI), UAE

CVIP_Korea

Title: Deep Factorized Network for image denoising (DeFNet)
Members: Sung-Jea Ko (sjko@korea.ac.kr), Dong-Pan Lim, Seung-Wook Kim, Seo-Won Ji, Sang-Won Lee
Affiliations: Korea University

Meteor

Title: A Multi-Level Network for Real Image Denoising
Members: Wenyi Tang (wenyitang@outlook.com)

UIUC-IFP

Title: Image Denoising with Kernel and Residual Prediction Networks
Members: Yuchen Fan (yuchenf4@illinois.edu), Yuqian Zhou, Ding Liu, Thomas S. Huang
Affiliations: University of Illinois at Urbana-Champaign

IID Research; Pervasive Visual Intelligence

Title: Fast and Felxible Blind sRGB Image Denosing
Members: Deyu Meng (dymeng@mail.xjtu.edu.cn), Lei Zhang (cslzhang@comp.polyu.edu.hk), Kai Zhang, Hongwei Yong
Affiliations: Xi'an Jiaotong Univeersity
Alibaba group Demo AcademyAI center

Orange_Cat

Title: Pyramid Image Denoising Network
Members: Yiyun Zhao (yiyunzhao@bupt.edu.cn), Pengliang Tang, Yue Lu
Affiliations: Beijing University of Posts and Telecommunications

IVL

Title: Deep Residual Autoencoder for Image Denoising
Members: Raimondo Schettini (schettini@disco.unimib.it), Simone Bianco, Simone Zini
Affiliations: University of Milano-Bicocca, Italy

offire

Title: Residual Dense Network for sRGB Denoising
Members: Chi Li (li_chi@hust.edu.cn), Yang Wang, Zhiguo Cao
Affiliations: Huazhong University of Science and Technology, China

References

[1] A. Abdelhamed, S. Lin, and M. S. Brown. A high-quality denoising dataset for smartphone cameras. In *CVPR*, 2018. 1, 2

[2] E. Agustsson and R. Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *CVPR Workshops*, July 2017. 4

[3] N. Ahn, B. Kang, and K. Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *ECCV*, 2018. 9

[4] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *TPAMI*, 33(5):898–916, May 2011. 4

[5] T. Brooks, B. Mildenhall, T. Xue, J. Chen, D. Sharlet, and J. T. Barron. Unprocessing images for learned raw denoising. *arXiv preprint arXiv:1811.11127*, 2018. 6

[6] A. Buades, B. Coll, and J. Morel. A non-local algorithm for image denoising. In *CVPR*, 2005. 1

[7] C. Chen, Z. Xiong, X. Tian, and F. Wu. Deep boosting for image denoising. In *ECCV*, pages 3–18, 2018. 7

[8] Q. Chen, J. Xu, and V. Koltun. Fast image processing with fully-convolutional networks. In *ICCV*, pages 2497–2506, 2017. 8

[9] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3D transform-domain collaborative filtering. *IEEE TIP*, 16(8):2080–2095, 2007. 1

[10] Y. Fan, J. Yu, and T. S. Huang. Wide-activated deep residual networks based restoration for BPG-compressed images. In *CVPR*, June 2018. 9

[11] S. Gu and R. Timofte. A brief review of image denoising algorithms and beyond. In *Springer series on Challenges in Machine Learning*, 2019. 1

[12] M. Haris, G. Shakhnarovich, and N. Ukita. Deep back-projection networks for super-resolution. In *CVPR*, June 2018. 8

[13] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, pages 1026–1034, 2015. 7

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 3, 5, 6, 7

[15] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *CVPR*, pages 4700–4708, 2017. 3, 5

[16] H. Karaimer and M. S. Brown. A software platform for manipulating the camera imaging pipeline. In *ECCV*, 2016. 2

[17] D.-W. Kim, J. R. Chung, and S.-W. a. Jung. GRDN: Grouped residual dense network for real image denoising and GAN-based real-world noise modeling. In *CVPR Workshops*, 2019. 5

[18] C. Knaus and M. Zwicker. Dual-domain image denoising. In *International Conference on Image Processing (ICIP)*, pages 440–444. IEEE, 2013. 7

[19] D. T. Kuan, A. A. Sawchuk, T. C. Strand, and P. Chavel. Adaptive noise smoothing filter for images with signal-dependent noise. *IEEE TPAMI*, (2):165–177, 1985. 1

[20] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPR Workshops*, pages 136–144, 2017. 6, 7

[21] C. Liu, R. Szeliski, S. B. Kang, C. L. Zitnick, and W. T. Freeman. Automatic estimation and removal of noise from a single image. *IEEE TPAMI*, 30(2):299–314, 2008. 1

[22] J. Liu, C.-H. Wu, Y. Wang, Q. Xu, Y. Zhou, H. Huang, C. Wang, S. Cai, Y. Ding, H. Fan, and J. Wang. Learning raw image denoising with bayer pattern normalization and bayer preserving augmentation. In *CVPR Workshops*, 2019. 6

[23] K. Ma, Z. Duanmu, Q. Wu, Z. Wang, H. Yong, H. Li, and L. Zhang. Waterloo Exploration Database: New challenges for image quality assessment models. *TIP*, 26(2):1004–1016, Feb. 2017. 4

[24] B. Park, S. Yu, and J. Jeong. Densely connected hierarchical network for image denoising. In *CVPR Workshops*, 2019. 5

[25] T. Plötz and S. Roth. Benchmarking denoising algorithms with real photographs. In *CVPR*, 2017. 1, 2

[26] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015. 3, 5, 6

[27] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, pages 1874–1883, 2016. 6

[28] Y. Tai, J. Yang, X. Liu, and C. Xu. Memnet: A persistent memory network for image restoration. In *CVPR*, 2017. 1

[29] R. Timofte, S. Gu, J. Wu, L. Van Gool, L. Zhang, M.-H. Yang, M. Haris, et al. Ntire 2018 challenge on single image super-resolution: Methods and results. In *CVPR Workshops*, June 2018. 4

[30] R. Timofte, R. Rothe, and L. Van Gool. Seven ways to improve example-based single image super resolution. In *CVPR*, pages 1865–1873, 2016. 4, 6

[31] A. Veit, M. J. Wilber, and S. Belongie. Residual networks behave like ensembles of relatively shallow networks. In *NIPS*, pages 550–558, 2016. 8

[32] G. Venkatesh, Y. Naresh, S. Little, and N. E. OConnor. A deep residual architecture for skin lesion segmentation. In *OR 2.0 Context-Aware Operating Theaters, Computer Assisted Robotic Endoscopy, Clinical Image-Based Procedures, and Skin Image Analysis*, pages 277–284. Springer, 2018. 6

[33] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE TIP*, 13(4):600–612, 2004. 2, 3

[34] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *Asilomar Conference on Signals, Systems & Computers*, volume 2, pages 1398–1402. IEEE, 2003. 3

[35] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon. CBAM: Convolutional block attention module. In *ECCV*, pages 3–19, 2018. 5

[36] J. Yu, Y. Fan, J. Yang, N. Xu, Z. Wang, X. Wang, and T. Huang. Wide activation for efficient and accurate image super-resolution. *arXiv preprint arXiv:1808.08718*, 2018. 9

[37] S. Yu, B. Park, and J. Jeong. Deep iterative down-up CNN for image denoising. In *CVPR Workshops*, 2019. 5

[38] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE TIP*, 2017. 1, 8, 9

[39] K. Zhang, W. Zuo, S. Gu, and L. Zhang. Learning deep cnn denoiser prior for image restoration. In *CVPR*, pages 3929–3938, 2017. 8

[40] K. Zhang, W. Zuo, and L. Zhang. FFDNet: Toward a fast and flexible solution for CNN-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018. 1, 6

[41] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu. Image super-resolution using very deep residual channel attention networks. In *ECCV*, pages 286–301, 2018. 7

[42] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu. Image super-resolution using very deep residual channel attention networks. In *ECCV*, September 2018. 8, 9

[43] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu. Residual dense network for image super-resolution. In *CVPR*, pages 2472–2481, 2018. 5

[44] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu. Residual dense network for image restoration. *arXiv preprint arXiv:1812.10477*, 2019. 11

[45] Z. Zhang, Q. Liu, and Y. Wang. Road extraction by deep residual U-Net. *IEEE Geoscience and Remote Sensing Letters*, 15(5):749–753, 2018. 6

[46] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on Computational Imaging*, 3(1):47–57, 2017. 7

[47] S. Zini, S. Bianco, and R. Schettini. Deep residual autoencoder for quality independent JPEG restoration. *arXiv preprint arXiv:1903.06117*, 2019. 11