Multi-Stage Progressive Image Restoration

Syed Waqas Zamir* 1 Aditya Arora* 1 Salman Khan 2 Munawar Hayat 3
Fahad Shahbaz Khan 2 Ming-Hsuan Yang 4,5,6 Ling Shao 1,2
1Inception Institute of AI 2Mohamed bin Zayed University of AI 3Monash University
4University of California, Merced 5Yonsei University 6Google Research

Abstract

Image restoration tasks demand a complex balance between spatial details and high-level contextualized information while recovering images. In this paper, we propose a novel synergistic design that can optimally balance these competing goals. Our main proposal is a multi-stage architecture, that progressively learns restoration functions for the degraded inputs, thereby breaking down the overall recovery process into more manageable steps. Specifically, our model first learns the contextualized features using encoder-decoder architectures and later combines them with a high-resolution branch that retains local information. At each stage, we introduce a novel per-pixel adaptive design that leverages in-situ supervised attention to reweight the local features. A key ingredient in such a multi-stage architecture is the information exchange between different stages. To this end, we propose a two-faceted approach where the information is not only exchanged sequentially from early to late stages, but lateral connections between feature processing blocks also exist to avoid any loss of information. The resulting tightly interlinked multi-stage architecture, named as MPRNet, delivers strong performance gains on ten datasets across a range of tasks including image deraining, deblurring, and denoising. The source code and pre-trained models are available at https://github.com/swz30/MPRNet.

1. Introduction

Image restoration is the task of recovering a clean image from its degraded version. Typical examples of degradation include noise, blur, rain, haze, etc. It is a highly ill-posed problem as there exist infinite feasible solutions. In order to restrict the solution space to valid/natural images, existing restoration techniques [19, 29, 39, 59, 66, 67, 100] explicitly use image priors that are handcrafted with empirical observations. However, designing such priors is a challenging task and often not generalizable. To ameliorate this issue, recent state-of-the-art approaches [17, 44, 57, 86, 87, 93, 94, 97] employ convolutional neural networks (CNNs) that implicitly learn more general priors by capturing natural image statistics from large-scale data.

The performance gain of CNN-based methods over the others is primarily attributed to its model design. Numerous network modules and functional units for image restoration have been developed including recursive residual learning [4, 95], dilated convolutions [4, 81], attention mechanisms [17, 86, 96], dense connections [73, 75, 97], encoder-decoders [7, 13, 43, 65], and generative models [44, 62, 90, 92]. Nevertheless, nearly all of these models for low-level vision problems are based on single-stage design. In contrast, multi-stage networks are shown to be more effective than their single-stage counterparts in high-level vision problems such as pose-estimation [14, 46, 54], scene parsing [15] and action segmentation [20, 26, 45].

Recently, few efforts have been made to bring the multi-stage design to image deblurring [70, 71, 88], and image deraining [47, 63]. We analyze these approaches to identify the architectural bottlenecks that hamper their performance. First, existing multi-stage techniques either employ the encoder-decoder architecture [71, 88] which is effective in encoding broad contextual information but unreliable in preserving spatial image details, or use a single-scale pipeline [63] that provides spatially accurate but semanti-
cally less reliable outputs. However, we show that the combination of both design choices in a multi-stage architecture is needed for effective image restoration. Second, we show that naively passing the output of one stage to the next stage yields suboptimal results [53]. Third, unlike in [88], it is important to provide ground-truth supervision at each stage for progressive restoration. Finally, during multi-stage processing, a mechanism to propagate intermediate features from earlier to later stages is required to preserve contextualized features from the encoder-decoder branches.

We propose a multi-stage progressive image restoration architecture, called MPRNet, with several key components. 1). The earlier stages employ an encoder-decoder for learning multi-scale contextual information, while the last stage operates on the original image resolution to preserve fine spatial details. 2). A supervised attention module (SAM) is plugged between every two stages to enable progressive learning. With the guidance of ground-truth image, this module exploits the previous stage prediction to compute attention maps that are in turn used to refine the previous stage features before being passed to the next stage. 3). A mechanism of cross-stage feature fusion (CSFF) is added that helps propagating multi-scale contextualized features from the earlier to later stages. Furthermore, this method eases the information flow among stages, which is effective in stabilizing the multi-stage network optimization. The main contributions of this work are:

• A novel multi-stage approach capable of generating contextually-enriched and spatially accurate outputs. Due to its multi-stage nature, our framework breaks down the challenging image restoration task into sub-tasks to progressively restore a degraded image.

• An effective supervised attention module that takes full advantage of the restored image at every stage in refining incoming features before propagating them further.

• A strategy to aggregate multi-scale features across stages.

• We demonstrate the effectiveness of our MPRNet by setting new state-of-the-art on ten synthetic and real-world datasets for various restoration tasks including image deraining, deblurring, and denoising while maintaining a low complexity (see Fig. 1). Further, we provide detailed ablations, qualitative results, and generalization tests.

2. Related Work

Recent years have witnessed a paradigm shift from high-end DSLR cameras to smartphone cameras. However, capturing high-quality images with smartphone cameras is challenging. Image degradations are often present in images either due to the limitations of cameras and/or adverse ambient conditions. Early restoration approaches are based on total variation [10, 67], sparse coding [3, 51, 52], self-similarity [8, 16], gradient prior [68, 80], etc. Recently, CNN-based restoration methods have achieved state-of-the-art results [57, 70, 86, 93, 97]. In terms of architectural design, these methods can be broadly categorized as single-stage and multi-stage.

Single-Stage Approaches. Currently, the majority of image restoration methods are based on a single-stage design, and the architectural components are usually based on those developed for high-level vision tasks. For example, residual learning [30] has been used to perform image denoising [2, 72, 93], image deblurring [42, 43] and image deraining [37]. Similarly, to extract multi-scale information, the encoder-decoder [65] and dilated convolution [83] models are often used [4, 28, 43]. Other single-stage approaches [5, 89, 97] incorporate dense connections [34].

Multi-Stage Approaches. These methods [24, 47, 53, 63, 70, 71, 88, 99] aim to recover clean image in a progressive manner by employing a light-weight subnetwork at each stage. Such a design is effective since it decomposes the challenging image restoration task into smaller easier sub-tasks. However, a common practice is to use the identical subnetwork for each stage which may yield suboptimal results, as shown in our experiments (Section 4).

Attention. Driven by its success in high-level tasks such as image classification [31, 32, 79], segmentation [21, 35] and detection [74, 79], attention modules have been used in low-level vision tasks [38]. Examples abound, including methods for image deraining [37, 47], deblurring [61, 70], super-resolution [17, 95], and denoising [4, 86]. The main idea is to capture long-range inter-dependencies along spatial dimensions [98], channel dimensions [32], or both [79].

3. Multi-Stage Progressive Restoration

The proposed framework for image restoration, shown in Fig. 2, consists of three stages to progressively restore images. The first two stages are based on encoder-decoder subnetworks that learn the broad contextual information due to large receptive fields. Since image restoration is a position-sensitive task (which requires pixel-to-pixel correspondence from the input to output), the last stage employs a subnetwork that operates on the original input image resolution (without any downsampling operation), thereby preserving the desired fine texture in the final output image.

Instead of simply cascading multiple stages, we incorporate a supervised attention module between every two stages. With the supervision of ground-truth images, our module rescales the feature maps of the previous stage before passing them to the next stage. Furthermore, we introduce a cross-stage feature fusion mechanism where the intermediate multi-scale contextualized features of the earlier subnetwork help consolidating the intermediate features of the latter subnetwork.

Although MPRNet stacks multiple stages, each stage has
3.1. Complementary Feature Processing

Existing single-stage CNNs for image restoration typically use one of the following architecture designs: 1) An encoder-decoder, or 2) A single-scale feature pipeline. The encoder-decoder networks [7, 13, 43, 65] first gradually map the input to low-resolution representations, and then progressively apply reverse mapping to recover the original resolution. While these models effectively encode multi-scale information, they are prone to sacrificing spatial details due to the repeated use of downsampling operation. In contrast, the approaches that operate on single-scale feature pipeline are reliable in generating images with fine spatial details [6, 18, 93, 97]. However, their outputs are semantically less robust due to the limited receptive field. This indicates the inherent limitations of the aforementioned architecture design choices that are capable of generating either spatially accurate or contextually reliable outputs, but not both. To exploit the merits of both designs, we propose a multi-stage framework where earlier stages incorporate the encoder-decoder networks, and the final stage employs a network that operates on the original input resolution.

Encoder-Decoder Subnetwork. Figure 3a shows our encoder-decoder subnetwork, which is based on the standard U-Net [65], with the following components. First, we add channel attention blocks (CABs) [95] to extract features at each scale (See Fig. 3b for CABs). Second, the feature maps at U-Net skip connections are also processed with the CAB. Finally, instead of using Transposed convolution for increasing spatial resolution of features in the decoder, we use bilinear upsampling followed by a convolution layer. This helps reduce checkerboard artifacts in the output image that often arise due to the Transposed convolution [55].

Original Resolution Subnetwork. In order to preserve fine details from the input image to the output image, we introduce the original-resolution subnetwork (ORSNet) in the last stage (see Fig. 2). ORSNet does not employ any downsampling operation and generates spatially-enriched high-resolution features. It consists of multiple original-resolution blocks (ORBs), each of which further contains CABs. The schematic of ORB is illustrated in Fig. 3b.

3.2. Cross-stage Feature Fusion

In our framework, we introduce the CSFF module between two encoder-decoders (see Fig. 3c), and between encoder-decoder and ORSNet (see Fig. 3d). Note that the features from one stage are first refined with 1 × 1 convolutions before propagating them to the next stage for aggregation. The proposed CSFF has several merits. First, it makes the network less vulnerable by the information loss due to repeated use of up- and down-sampling operations in the encoder-decoder. Second, the multi-scale features of one stage help enriching the features of the next stage. Third, the network optimization procedure becomes more stable as it eases the flow of information, thereby allowing us to add several stages in the overall architecture.

Figure 2: Proposed multi-stage architecture for progressive image restoration. Earlier stages employ encoder-decoders to extract multi-scale contextualized features, while the last stage operates at the original image resolution to generate spatially accurate outputs. A supervised attention module is added between every two stages that learns to refine features of one stage before passing them to the next stage. Dotted pink arrows represent the cross-stage feature fusion mechanism.
Figure 3: (a) Encoder-decoder subnetwork. (b) Illustration of the original resolution block (ORB) in our ORSNet subnetwork. Each ORB contains multiple channel attention blocks. GAP represents global average pooling [49]. (c) Cross-stage feature fusion between stage 1 and stage 2. (d) CSFF between stage 2 and the last stage.

3.3. Supervised Attention Module

Recent multi-stage networks for image restoration [70, 88] directly predict an image at each stage, which is then passed to the next consecutive stage. Instead, we introduce a supervised attention module between every two stages, which facilitates achieving significant performance gain. The schematic diagram of SAM is shown in Fig. 4, and its contributions are two-fold. First, it provides ground-truth supervisory signals useful for the progressive image restoration at each stage. Second, with the help of locally supervised predictions, we generate attention maps to suppress the less informative features at the current stage and only allow the useful ones to propagate to the next stage.

As illustrated in Fig. 4, SAM takes the incoming features $F_{in} \in \mathbb{R}^{H \times W \times C}$ of the earlier stage and first generates a residual image $R_S \in \mathbb{R}^{H \times W \times 3}$ with a simple $1 \times 1$ convolution, where $H \times W$ denotes the spatial dimension and $C$ is the number of channels. The residual image is added to the degraded input image $I$ to obtain the restored image $X_S \in \mathbb{R}^{H \times W \times 3}$. To this predicted image $X_S$, we provide explicit supervision with the ground-truth image. Next, per-pixel attention masks $M \in \mathbb{R}^{H \times W \times C}$ are generated from the image $X_S$ using a $1 \times 1$ convolution followed by the sigmoid activation. These masks are then used to re-calibrate the transformed local features $F_{in}$ (obtained after $1 \times 1$ convolution), resulting in attention-guided features which are added to the identity mapping path. Finally, the attention-augmented feature representation $F_{out}$, produced by SAM, is passed to the next stage for further processing.

4. Experiments and Analysis

We evaluate our method for several image restoration tasks, including (a) image deraining, (b) image deblurring, and (c) image denoising on 10 different datasets.

4.1. Datasets and Evaluation Protocol

Quantitative comparisons are performed using the PSNR and SSIM [76] metrics. As in [7], we report (in parenthesis) the reduction in error for each method relative to the best performing method by translating PSNR to RMSE ($\text{RMSE} = \sqrt{10 - \text{PSNR}/10}$) and SSIM to DSSIM ($\text{DSSIM} = (1 - \text{SSIM})/2$). The datasets used for training and testing are summarized in Table 1 and described next.

Image Deraining. Using the same experimental setups of the recent best method on image deraining [37], we train our model on 13,712 clean-rain image pairs gathered from multiple datasets [23, 48, 81, 89, 90], as shown in Table 1. With this single trained model, we perform evaluation on various test sets, including Rain100H [81], Rain100L [81], Test100 [90], Test2800 [23], and Test1200 [89].

Image Deblurring. As in [70, 88, 43, 71], we use the GoPro [53] dataset that contains 2,103 image pairs for training and 1,111 pairs for evaluation. Furthermore, to demonstrate generalizability, we take our GoPro trained model and directly apply it on the test images of the HIDE [69] and RealBlur [64] datasets. The HIDE dataset is specifically collected for human-aware motion deblurring and its test set contains 2,025 images. While the GoPro and HIDE datasets are synthetically generated, the image pairs of RealBlur dataset are captured in real-world conditions. The RealBlur dataset has two subsets: (1) RealBlur-J is formed with the camera JPEG outputs, and (2) RealBlur-R is generated offline by applying white balance, demosaicking, and denoising operations to the RAW images.

Image Denoising. To train our model for image denoising task, we use 320 high-resolution images of the SIDD dataset [1]. Evaluation is conducted on 1,280 validation patches from the SIDD dataset [1] and 1,000 patches from the DND benchmark dataset [60]. These test patches are extracted from the full resolution images by the original au-
Table 1: Dataset description for various image restoration tasks.

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Deraining</th>
<th>Deblurring</th>
<th>Denoising</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train Samples</td>
<td>11200</td>
<td>1800</td>
<td>700</td>
</tr>
<tr>
<td>Test Samples</td>
<td>2800</td>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>

Testset Rename Test2800 - Test100 Rain100H Rain100L Test1200 -

Table 2: Image deraining results. Best and second best scores are highlighted and underlined. For each method, reduction in error relative to the best-performing algorithm is reported in parenthesis (see Section 4.1 for error calculation technique). Our MPRNet achieves ~20% relative improvement in PSNR over the previous best method MSPFN [37].

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR ↑</td>
<td>SSIM ↑</td>
<td>PSNR ↑</td>
<td>SSIM ↑</td>
<td>PSNR ↑</td>
<td>SSIM ↑</td>
</tr>
<tr>
<td>DerainNet [22]</td>
<td>22.77</td>
<td>0.810</td>
<td>14.92</td>
<td>0.592</td>
<td>27.03</td>
<td>0.884</td>
</tr>
<tr>
<td>SEMI [77]</td>
<td>22.35</td>
<td>0.788</td>
<td>16.56</td>
<td>0.482</td>
<td>25.03</td>
<td>0.842</td>
</tr>
<tr>
<td>DIDMDN [89]</td>
<td>22.56</td>
<td>0.818</td>
<td>17.35</td>
<td>0.524</td>
<td>25.23</td>
<td>0.741</td>
</tr>
<tr>
<td>UMRL [82]</td>
<td>24.41</td>
<td>0.829</td>
<td>26.01</td>
<td>0.832</td>
<td>29.18</td>
<td>0.923</td>
</tr>
<tr>
<td>RESCAN [47]</td>
<td>25.00</td>
<td>0.835</td>
<td>26.36</td>
<td>0.786</td>
<td>29.80</td>
<td>0.881</td>
</tr>
<tr>
<td>PreNet [65]</td>
<td>24.81</td>
<td>0.851</td>
<td>26.77</td>
<td>0.858</td>
<td>32.44</td>
<td>0.950</td>
</tr>
<tr>
<td>MSPFN [37]</td>
<td>27.50</td>
<td>0.876</td>
<td>28.66</td>
<td>0.860</td>
<td>32.40</td>
<td>0.933</td>
</tr>
</tbody>
</table>

MPRNet (Ours) | 30.27 | 0.897 | 30.41 | 0.890 | 36.40 | 0.965 | 33.64 | 0.938 | 32.91 | 0.916 | 32.73 | (0.0%) | 0.921 | (0.0%) |

4.2. Implementation Details

Our MPRNet is end-to-end trainable and requires no pre-training. We train separate models for three different tasks. We employ 2 CABs at each scale of the encoder-decoder, and for downsampling we use $2 \times 2$ max-pooling with stride 2. In the last stage, we employ ORSNet that contains 3 ORBs, each of which further uses 8 CABs. Depending on the task complexity, we scale the network width by setting the number of channels to 40 for deraining, 80 for denoising, and 96 for deblurring. The networks are trained on 256×256 patches with a batch size of 16 for $4 \times 10^{5}$ iterations. For data augmentation, horizontal and vertical flips are randomly applied. We use Adam optimizer [41] with the initial learning rate of $2 \times 10^{-4}$, which is steadily decreased to $1 \times 10^{-6}$ using the cosine annealing strategy [50].

4.3. Image Deraining Results

For the image deraining task, consistent with prior work [37], we compute image quality scores using the Y channel (in YCbCr color space). Table 2 shows that our method significantly advances state-of-the-art by consistently achieving better PSNR/SSIM scores on all five datasets. Compared to the recent best algorithm MSPFN [37], we obtain a performance gain of 1.98 dB (average across all datasets), indicating 20% error reduction. The improvements on some datasets are as large as 4 dB, e.g., Rain100L [81]. Further, our model has 3.7× fewer parameters than MSPFN [37], while being 2.4× faster.

Figure 5 shows visual comparisons on challenging images. Our MPRNet is effective in removing rain streaks of different orientations and magnitudes, and generates images that are visually pleasant and faithful to the ground-truth. In contrast, other approaches compromise structural content (first row), introduce artifacts (second row), and do not completely remove rain streaks (third row).

4.4. Image Deblurring Results

We report the performance of evaluated image deblurring approaches on the synthetic GoPro [53] and HIDE [69] datasets in Table 3. Overall, our model performs favorably against other algorithms. Compared to the previous best performing technique [70], our method achieves 9% improvement in PSNR and 21% in SSIM on the GoPro [53] dataset, and a 11% and 13% reduction in error on the HIDE dataset [69]. It is worth noticing that our network is trained only on the GoPro dataset, but achieves the state-of-the-art results (+0.98 dB) on the HIDE dataset, thereby demonstrating its strong generalization capability.

We evaluate our MPRNet on the real-world images of a recent RealBlur [64] dataset under two experimental settings: 1). apply the GoPro trained model directly on RealBlur (to test generalization to real images), and 2). train and test on RealBlur data. Table 4 shows the experimental results. For setting 1, our MPRNet obtains performance gains of 0.29 dB on the RealBlur-R subset and 0.28 dB on the RealBlur-J subset over the DMFPHN algorithm [88]. A similar trend is observed for setting 2, where our gains over SRN [71] are 0.66 dB and 0.38 dB on RealBlur-R and RealBlur-J, respectively.

Figure 6 shows some deblurred images by the evaluated approaches. Overall, the images restored by our model are sharper and closer to the ground-truth than those by others.
4.5. Image Denoising Results

In Table 5, we report PSNR/SSIM scores of several image denoising methods on the SIDD [1] and DND [60] datasets. Our method obtains considerable gains over the state-of-the-art approaches, i.e., 0.19 dB over CycleISP [86] on SIDD and 0.21 dB over SADNet [11] on DND. Note that the DND dataset does not contain any training images, i.e., the complete publicly released dataset is just a test set. Experimental results on the DND benchmark with our SIDD trained model demonstrates our model generalizes well to different image domains.

Fig. 7 illustrates visual results. Our method is able to remove real noise, while preserving the structural and textural image details. In contrast, the images restored by other methods are taken from the RealBlur benchmark [64]. The PSNR/SSIM scores for other evaluated approaches are taken from the RealBlur benchmark [64].

Table 4: Deblurring comparisons on the RealBlur dataset [64] under two different settings: 1). applying our GoPro trained model directly on the RealBlur set (to evaluate generalization to real images), 2). Training and testing on RealBlur data where methods are denoted with symbol †. The PSNR/SSIM scores for other evaluated approaches are taken from the RealBlur benchmark [64].

Table 3: Deblurring results. Our method is trained only on the GoPro dataset [53] and directly applied to the HIDE dataset [69].
Table 5: Denoising comparisons on SIDD [1] and DND [60] datasets. * denotes the methods that use additional training data. Whereas our MPRNet is only trained on the SIDD images and directly tested on DND.

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>DnCNN [93]</td>
<td>23.66</td>
<td>(84.2%)</td>
<td>0.583</td>
<td>(89.9%)</td>
</tr>
<tr>
<td>MLP [9]</td>
<td>24.71</td>
<td>(82.2%)</td>
<td>0.641</td>
<td>(88.3%)</td>
</tr>
<tr>
<td>BM3D [16]</td>
<td>25.65</td>
<td>(80.2%)</td>
<td>0.685</td>
<td>(86.7%)</td>
</tr>
<tr>
<td>CBIDNet+ [28]</td>
<td>30.78</td>
<td>(64.2%)</td>
<td>0.801</td>
<td>(78.9%)</td>
</tr>
<tr>
<td>RDNNet+ [4]</td>
<td>38.71</td>
<td>(10.9%)</td>
<td>0.951</td>
<td>(14.3%)</td>
</tr>
<tr>
<td>AINDNet* [40]</td>
<td>38.95</td>
<td>(8.4%)</td>
<td>0.952</td>
<td>(12.5%)</td>
</tr>
<tr>
<td>VDN [84]</td>
<td>39.28</td>
<td>(4.8%)</td>
<td>0.956</td>
<td>(4.6%)</td>
</tr>
<tr>
<td>SADNet+ [11]</td>
<td>39.46</td>
<td>(2.8%)</td>
<td>0.957</td>
<td>(2.3%)</td>
</tr>
<tr>
<td>DANet+ [85]</td>
<td>39.47</td>
<td>(2.7%)</td>
<td>0.957</td>
<td>(2.3%)</td>
</tr>
<tr>
<td>CycleISP* [86]</td>
<td>39.52</td>
<td>(2.2%)</td>
<td>0.957</td>
<td>(2.3%)</td>
</tr>
<tr>
<td>MPRNet (Ours)</td>
<td>39.71 (0.0%)</td>
<td>0.958 (0.0%)</td>
<td>39.80 (0.0%)</td>
<td>0.954 (4.4%)</td>
</tr>
</tbody>
</table>

Table 6: Ablation study on individual components of the proposed MPRNet.

<table>
<thead>
<tr>
<th>#Stages</th>
<th>Stage Combination</th>
<th>SAM</th>
<th>CSFF</th>
<th>PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>U-Net (baseline)</td>
<td>-</td>
<td>-</td>
<td>28.94</td>
</tr>
<tr>
<td>1</td>
<td>ORSNet (baseline)</td>
<td>-</td>
<td>-</td>
<td>28.91</td>
</tr>
<tr>
<td>2</td>
<td>U-Net + U-Net</td>
<td>×</td>
<td>×</td>
<td>29.40</td>
</tr>
<tr>
<td>2</td>
<td>ORSNet + ORSNet</td>
<td>×</td>
<td>×</td>
<td>29.53</td>
</tr>
<tr>
<td>2</td>
<td>U-Net + ORSNet</td>
<td>×</td>
<td>×</td>
<td>29.70</td>
</tr>
<tr>
<td>3</td>
<td>U-Nets + ORSNet</td>
<td>×</td>
<td>×</td>
<td>29.96</td>
</tr>
<tr>
<td>3</td>
<td>U-Nets + ORSNet</td>
<td>×</td>
<td>✓</td>
<td>30.07</td>
</tr>
<tr>
<td>3</td>
<td>U-Nets + ORSNet</td>
<td>✓</td>
<td>×</td>
<td>30.31</td>
</tr>
<tr>
<td>3</td>
<td>U-Nets + ORSNet</td>
<td>✓</td>
<td>✓</td>
<td>30.49</td>
</tr>
</tbody>
</table>

4.6. Ablation Studies

Here we present ablation experiments to analyze the contribution of each component of our model. Evaluation is performed on the GoPro dataset [53] with the deblurring models trained on image patches of size 128×128 for 10⁵ iterations, and the results are shown in Table 6.

Number of stages. Our model yields better performance as the number of stages increases, which validates the effectiveness of our multi-stage design.

Choices of subnetworks. Since each stage of our model could employ different subnetwork design, we test different options. We show that using the encoder-decoder in the earlier stage(s) and the ORSNet in the last stage leads to improved performance (29.7 dB) as compared to employing the same design for all the stages (29.4 dB with U-Net+U-Net, and 29.53 dB with ORSNet+ORSNet).

SAM and CSFF. We demonstrate the effectiveness of the proposed supervised attention module and cross-stage feature fusion mechanism by removing them from our final model. Table 6 shows a substantial drop in PSNR from 30.49 dB to 30.07 dB when SAM is removed, and from 30.49 dB to 30.31 dB when we take out CSFF. Removing both of these components degrades the performance by a large margin from 30.49 dB to 29.86 dB.

5. Resource Efficient Image Restoration

CNN models generally exhibit a trade-off between accuracy and computational efficiency. In the pursuit of achieving higher accuracy, deeper and complex models are often developed. Although large models tend to perform better than their smaller counterparts, the computational cost can
be prohibitively high. As such, it is of great interest to develop resource-efficient image restoration models. One solution is to train the same network by adjusting its capacity every time the target system is changed. However, it is tedious and oftentimes infeasible. A more desirable approach is to have a single network that can make (a) early predictions for compute efficient systems and (b) latter predictions to obtain high accuracy. A multi-stage restoration model naturally offers such functionalities.

Table 7 reports the stage-wise results of our multi-stage approach. Our MPRNet demonstrates competitive restoration performance at each stage. Notably, our stage-1 model is light, fast, and yields better results than other sophisticated algorithms such as SRN [71] and DeblurGAN-v2 [43]. Similarly, when compared to a recent method DMPHN [88], our stage-2 model shows the PSNR gain of 0.51 dB while being more resource-efficient (∼2× fewer parameters and 13× faster).

6. Conclusion

In this work, we propose a multi-stage architecture for image restoration that progressively improves degraded inputs by injecting supervision at each stage. We develop guiding principles for our design that demand complementary feature processing in multiple stages and a flexible information exchange between them. To this end, we propose contextually-enriched and spatially accurate stages that encode a diverse set of features in unison. To ensure synergy between reciprocal stages, we propose feature fusion across stages and an attention guided output exchange from earlier stages to the later ones. Our model achieves significant performance gains on numerous benchmark datasets. In addition, our model is light-weighted in terms of model size and efficient in terms of runtime, which are of great interest for devices with limited resources.

Acknowledgments. M.-H. Yang is supported in part by the NSF CAREER Grant 1149783. Special thanks to Kui Jiang for providing image deraining results.
References


[4] Saeed Anwar and Nick Barnes. Real image deconvolution with feature attention. ICCV, 2019. 1, 2, 7, 8


[22] Xueyang Fu, Jiabin Huang, Xinghao Ding, Yinghao Liao, and John Paisley. Clearing the skies: A deep network architecture for single-image rain removal. TIP, 2017. 5, 6

[23] Xueyang Fu, Jiabin Huang, Delu Zeng, Yue Huang, Xinghao Ding, and John Paisley. Removing rain from single images via a deep detail network. In CVPR, 2017. 4, 5

[24] Xueyang Fu, Borong Liang, Yue Huang, Xinghao Ding, and John Paisley. Lightweight dynamic scene deblurring with parameter selective sharing and nested skip connections. In CVPR, 2019. 6, 7


[34] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. CCNet: Criss-cross attention for semantic segmentation. In ICCV, 2019. 2


[37] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak
[40] Yoonsik Kim, Jae Woong Soh, Gu Yong Park, and Nam Ik Cho. Transfer learning from synthetic to real-noise denoising with adaptive instance normalization. In CVPR, 2020. 7, 8
[45] Shi-Jie Li, Yazan AbuFarha, Yun Liu, Ming-Ming Cheng, and Juergen Gall. MS-TCN++: Multi-stage temporal convolutional network for action segmentation. TPAMI, 2020. 1
[47] Xia Li, Jianlong Wu, Gu Yong Park, and Nam Ik Cho. Transfer learning from synthetic to real-noise denoising with adaptive instance normalization. In CVPR, 2020. 7, 8
[55] Tobias Plotz and Stefan Roth. Benchmarking denoising algorithms with real photographs. In CVPR, 2017. 4, 5, 6, 7, 8
[66] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiayua Jia. Scale-recurrent network for deep image deblurring. In CVPR, 2018. 1, 2, 4, 5, 6, 7, 8
[68] Tong Tong, Gen Li, Xiejie Liu, and Qinquan Gao. Image super-resolution using dense skip connections. In ECCV, 2017. 1
[70] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. ESRGAN:


