# EVRNet: Efficient Video Restoration on Edge Devices

Sachin Mehta*
University of Washington
USA

Amit Kumar
Facebook Inc.
USA

Fitsum Reda†
Google
USA

Varun Nasery
Facebook Inc.
USA

Vikram Mulukutla
Facebook Inc.
USA

Rakesh Ranjan
Facebook Inc.
USA

Vikas Chandra
Facebook Inc.
USA

## ABSTRACT

In video transmission applications, video signals are transmitted over lossy channels, resulting in low-quality received signals. To restore videos on recipient edge devices in real-time, we introduce an efficient video restoration network, EVRNet. EVRNet efficiently allocates parameters inside the network using alignment, differential, and fusion modules. With extensive experiments on different video restoration tasks (deblocking, denoising, and super-resolution), we demonstrate that EVRNet delivers competitive performance to existing methods with significantly fewer parameters and MACs. For example, EVRNet has 260× fewer parameters and 958× fewer MACs than enhanced deformable convolution-based video restoration network (EDVR) for 4× video super-resolution while its SSIM score is 0.018 less than EDVR. We also evaluated the performance of EVRNet under multiple distortions on unseen dataset to demonstrate its ability in modeling variable-length sequences under both camera and object motion.

## CCS CONCEPTS

• **Information systems → Mobile information processing systems**.

## KEYWORDS

Video Restoration, Edge Devices, On-Device, Convolutional Neural Network, Video Decompression, Video Denoising, Super-resolution

**ACM Reference Format:**
Sachin Mehta, Amit Kumar, Fitsum Reda, Varun Nasery, Vikram Mulukutla, Rakesh Ranjan, and Vikas Chandra. 2021. EVRNet: Efficient Video Restoration on Edge Devices. In *Proceedings of the 29th ACM Int'l Conference on Multimedia (MM '21), Oct. 20–24, 2021, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3474085.3475477

*Work completed during internship at Facebook Inc..
†Work done while working at Facebook Inc..


*MM '21, October 20–24, 2021, Virtual Event, China.*

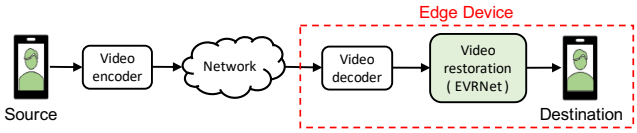
ACM ISBN 978-1-4503-8651-7/21/10…$15.00
https://doi.org/10.1145/3474085.3475477

## 1 INTRODUCTION

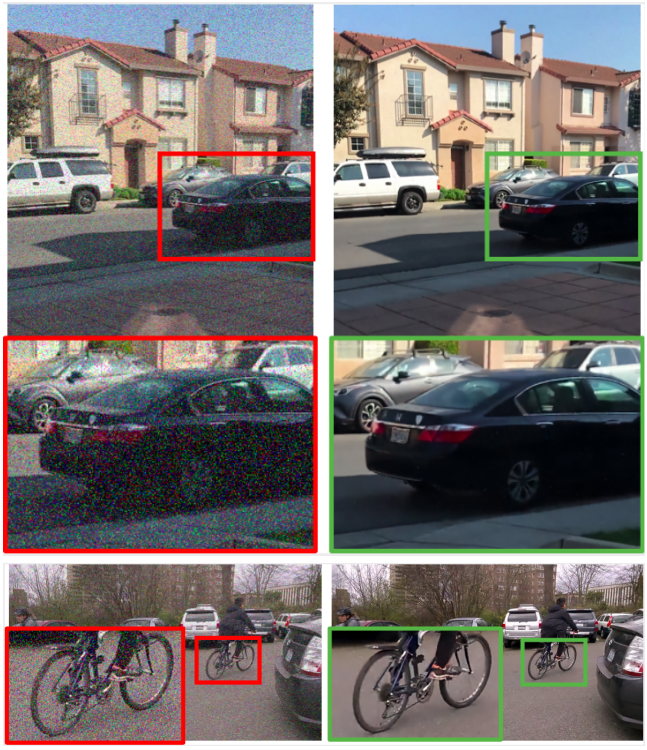
Video restoration aims at recovering the expected quality of videos in recipient devices. Deep neural network-based solutions [11, 27, 54, 58, 62] achieve high accuracy on these tasks, but they are computationally very expensive. For example, a deformable convolution-based video restoration network, EDVR [54], has 21.1 million parameters and requires 9.96 TMACs (multiplication-addition operations) for up-sampling a 360p video frame by a factor of 4. Many video transmission applications (e.g., video streaming and conferencing) run on edge devices, such as smartphones. The trend is likely to continue with the adoption of technologies like augmented and virtual reality. Because edge devices have limited computational resources, memory, and energy, heavy-weight video restoration networks are not suitable for such devices. Additionally, video signals at source often undergo lossy compression for transmission under limited network bandwidth (see Figure 1a). Because of compression and transmission noise, the quality of received video signals is low. In order to be effective, neural networks for these applications should be light-weight and low latency while restoring high quality and temporally stable videos on edge devices.

This work introduces an efficient neural network called Efficient Video Restoration Network (EVRNet) to restore videos with high quality on edge devices in real-time, and is shown in Figure 2. EVRNet is inspired by traditional computer vision methods for motion estimation and image enhancement [5, 33, 40]. Briefly, EVRNet uses an alignment module to align current and previous frames without optical flow. High-frequency components (e.g., object edges) are often lost during compression. To restore such details, EVRNet uses a differential and fusion module. The differential module learns representations corresponding to high-frequency components while the fusion module uses these representations along with the input to produce high-quality output (see Figure 1b). EVRNet more efficiently allocates parameters and operations inside each of these modules using small and light-weight encoder-decoder networks.

EVRNet is refreshingly simple and can be used to restore either a single (e.g., video denoising) or multiple distortions (e.g., video decompression and denoising). To demonstrate the simplicity and effectiveness of EVRNet, it's performance is evaluated on




**(a) EVRNet in video conferencing application.**



**(b) Sample EVRNet results on unseen videos. Left: compressed and noisy frames. Right: Restored frames.**

**Figure 1: EVRNet on edge devices. (a) shows how EVRNet is integrated to an edge device while (b) shows the results of EVRNet on H264 compressed and noisy (Gaussian + salt and pepper) "unseen" videos. EVRNet is able to restore the videos with multiple artifacts. See supplementary material for more results.**

a large scale Vimeo-90K dataset [58] on three independent and standard video restoration tasks: (1) deblocking (Section 4.2), (2) denoising (Section 4.3), and (3) super-resolution (Section 4.4). EVRNet's performance is also studied for a typical low-bandwidth video conferencing system where videos undergoes multiple distortions due to video encoding and noisy transmission network (Section 5). EVRNet delivers competitive performance as state-of-the-art methods but with significantly fewer parameters and MACs. For example, on the task of video deblocking and denoising, EVRNet delivers similar performance to ToFlow [58] but with 46× and 13.63× fewer MACs and parameters, respectively. On the task of 4× video super-resolution, EVRNet has slightly lower SSIM score (0.018) than EDVR [54], but has 260× fewer parameters and 958× fewer MACs.

*Contributions.* The main contributions of this paper are: (1) A novel efficient video restoration network capable of running at real-time on edge devices. (2) A single neural network, EVRNet, that can be used to restore video under a single or multiple distortions. (3) Qualitative and quantitative results along with comparisons with state-of-the-art methods on three video restoration tasks, demonstrating EVRNet's competitive performance, while having significantly fewer network parameters and MACs.

## 2 RELATED WORK

Designing deep neural networks for video restoration tasks is an active area of research. This section briefly reviews these approaches followed by efforts in improving the efficiency of neural networks.

*Video restoration.* Deblocking (e.g., [6, 32, 35, 58, 62]), denoising (e.g., [27, 35, 58, 61, 62]), and super-resolution (e.g., [3, 7, 21, 25, 28, 31, 44, 51, 53–55]) are three main video restoration tasks that have been studied widely in the literature. Video deblocking aims at removing artifacts that arises due to compression (e.g., checkerboard patterns). Video denoising aims at removing noise-related artifacts that may arise due to noisy transmission channel (e.g., Internet). Super-resolution aims at producing a high-resolution videos from low-resolution videos. Most methods are studied on one of these tasks and are computationally very expensive. For example, ToFlow [58] has about 466 GMACs for denoising (or deblocking) a 360p video. Unlike existing methods, EVRNet can be used to restore videos under either single or multiple distortions.

Also, some video restoration methods use optical flow (e.g., [2, 3, 58]) which is computed using deep flow networks, such as FlowNet [8, 23], PWCNet [46], and SpyNet [41]). Computing optical flow with these networks is expensive and this limits the practical applicability of such approaches, especially on resource-constrained devices (e.g., smartphones). Similar to [24, 52, 54], EVRNet also does implicit alignment between consecutive frames using the pyramidal structure in the alignment module and handles large motion without optical flow. Importantly, EVRNet can restore videos with high-quality in real-time on edge devices.

*Efficient networks.* Designing efficient deep neural networks is an active area in both academic and industrial research, and aims at reducing the network parameters and MACs by designing efficient learnable layers (e.g., depth-wise [4] and dimension-wise [36] convolutions) or quantization or compression or pruning. The most similar to our work are the methods on architecture design (hand-crafted [19, 34, 38, 45] and learned [18, 49, 50, 63]). Similar to these methods, EVRNet also uses depth-wise convolutions for learning representations efficiently. Network compression & pruning (e.g., [10, 14, 29, 39, 56, 60]), quantization (e.g., [1, 22, 42, 57]), and distillation (e.g., [9, 15, 59]) are important complementary efforts that can be further used to improve the efficiency of EVRNet.

## 3 EVRNET

We introduce EVRNet, an **E**fficient **V**ideo **R**estoration **Net**work, to remove artifacts and restore videos in edge devices in real-time (schematic shown in Figure 2). EVRNet takes inspirations from traditional techniques in motion estimation and image enhancement [33, 40]. Specifically, EVRNet uses an alignment module based on a

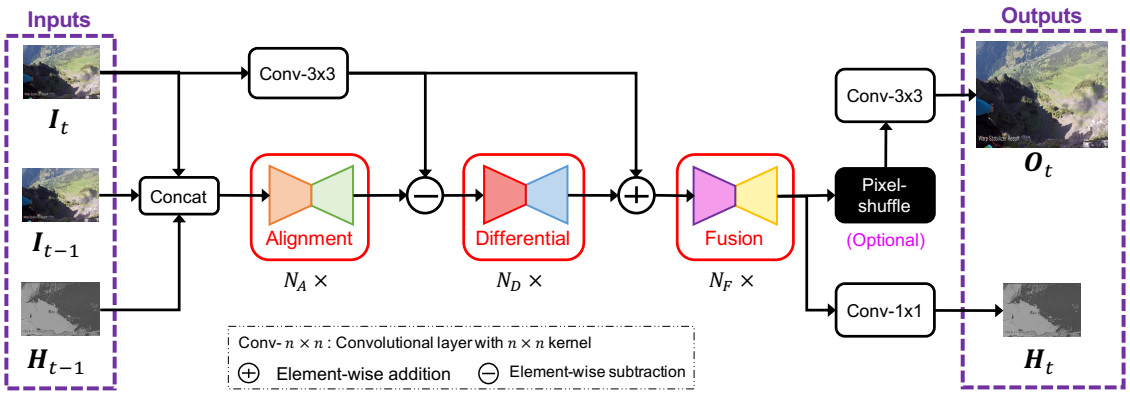
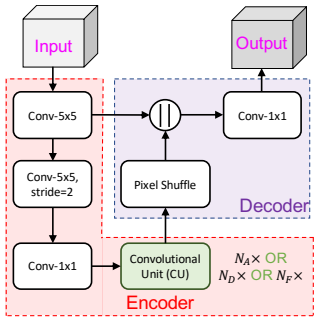
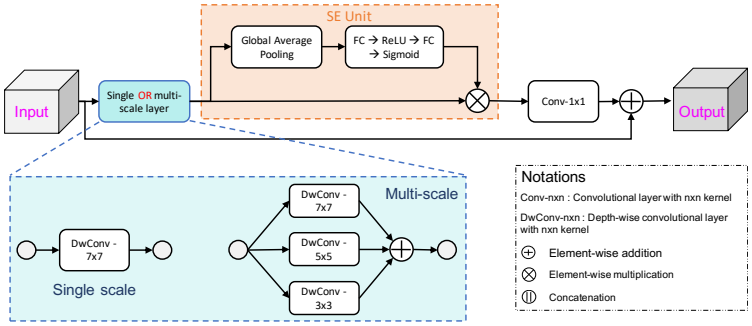



**Figure 2: Overview of the EVRNet architecture for video restoration tasks. The EVRNet architecture takes the current RGB $I_t$, previous RGB $I_{t-1}$, and previous latent $H_{t-1}$ frames as inputs and produces two outputs: restored RGB frame $O_t$ and latent frame $H_t$. The pixel-shuffle operation is optional and is used only for super-resolution tasks. The alignment, differential, and fusion modules are light-weight and efficient encoder-decoder networks (see Figure 3) with $N_A$, $N_D$, and $N_F$ layers, respectively.**




**(a) Small encoder-decoder network**




**(b) Convolutional unit (CU)**

**Figure 3: Overview of alignment, differential, and fusion module. Each of these modules are identical in construction, i.e., they follow an encoder-decoder structure (a), with an exception to the number of convolutional units (b). The alignment, differential, and fusion module stacks $N_A$, $N_D$, and $N_F$ convolutional units (CUs) to learn deep representations, respectively.**

pyramidal structure to model the motion without explicit use of optical flow. To restore high-frequency details (e.g., edges) that may be lost due to distortions (e.g., compression), EVRNet uses differential and fusion module. These modules learn high-frequency components which are then added back to achieve sharp details. Following sub-sections describe the overall architecture of EVRNet in detail.

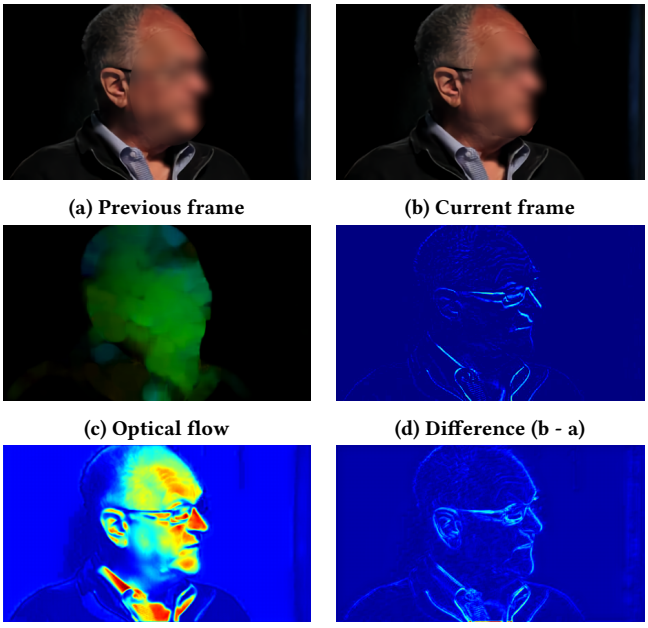
## 3.1 EVRNet Architecture

EVRNet is an auto-regressive network that efficiently models the relationships between a current frame $\mathbf{I}_t \in \mathbb{R}^{3\times H\times W}$, a previous frame $\mathbf{I}_{t-1} \in \mathbb{R}^{3\times H\times W}$, and a previous latent frame $\mathbf{H}_{t-1} \in \mathbb{R}^{2\times H\times W}$. Mathematically, EVRNet takes the form:

$$\mathbf{O}_t, \mathbf{H}_t = \mathcal{F}(\mathbf{I}_t, \mathbf{I}_{t-1}, \mathbf{H}_{t-1}) \tag{1}$$

where $\mathcal{F}$ is our learned network, EVRNet, that efficiently synthesizes restored frame $\mathbf{O}_t \in \mathbb{R}^{3\times H\times W}$ and a latent frame $\mathbf{H}_t \in \mathbb{R}^{2\times H\times W}$, conditioned on inputs ($\mathbf{I}_t, \mathbf{I}_{t-1}$ and $\mathbf{H}_{t-1}$). The latent frame $\mathbf{H}_t$ is similar to cell state in LSTMs [16] and allows information flow between different time steps. Overall, EVRNet has three main modules: (1) alignment module, (2) differential module, and (3) fusion module.

*Alignment module* The alignment module takes a concatenation of the inputs ($\mathbf{I}_t, \mathbf{I}_{t-1}$ and $\mathbf{H}_{t-1}$) and produces aligned representations $\mathbf{A}_t \in \mathbb{R}^{d\times H\times W}$ using an efficient and light-weight encoder-decoder network (Figure 3a). The alignment module first learns pyramidal representations using the encoder network. These representations are then combined by the decoder to produce aligned representations. Compared to existing methods that learns very deep pyramidal representations for motion estimation [8, 23, 33, 41, 46], EVRNet is very light-weight and shallow. To demonstrate the ability of EVRNet in modeling the motion, an example is shown in Figure 4 where person moves his head during a conversation. The most salient regions between consecutive frames are near the nose, spectacles, and shirt as depicted by the optical flow and difference image in Figure 4c and 4d, respectively. The alignment module in the EVRNet also pays attention to these salient regions (red color



(a) Previous frame (b) Current frame

(c) Optical flow (d) Difference (b - a)

(e) Alignment module output (f) Differential module output

**Figure 4: This example visualizes outputs of two EVRNet modules (alignment and differential). The alignment module pays attention to areas corresponding to motion, i.e., nose and spectacles (c, d vs. e) while the differential module pays attention to high frequency components (e.g., spectacle edges in (f)) in region corresponding to motion. Face in (a) and (b) is blurred for de-identification.**

regions in Figure 4f), illustrating EVRNet's ability to model the motion implicitly.

Specifically, the encoder in the alignment module consists of (1) a standard $5 \times 5$ convolutional layer, (2) a standard $5 \times 5$ convolutional layer with a stride of two, (3) a point-wise convolutional layer, and (4) $N_A$ convolutional units (CUs; Section 3.2), where $N_A$ controls the depth of alignment module. The decoder follows a simplified UNet-like architecture [43]. The output of the last CU is first upsampled and then concatenated with the output of the first $5 \times 5$ convolutional layer. The resultant output is then fused using a point-wise convolution to produce aligned representations $\mathbf{A}_t$.

*Differential module* The differential module aims at learning high-frequency components in an image such as object edges. To do so, the input $\mathbf{I}_t$ is first projected to the same dimensionality as $\mathbf{A}_t$ using a $3 \times 3$ convolutional layer to produce a projected output $\mathbf{P}_t \in \mathbb{R}^{d \times H \times W}$. An element-wise difference is then computed between $\mathbf{P}_t$ and $\mathbf{A}_t$. The resultant output is then fed to differential module to further refine these representations and produce high-frequency representations $\mathbf{D}_t \in \mathbb{R}^{d \times H \times W}$. Figure 4f shows an example where EVRNet pays attention to high-frequency components (e.g., spectacle and ear edges). Similar to the alignment module, the differential module also takes the form of small and light-weight encoder-decoder network, with an exception to number of CUs. In the differential module, we stack $N_D$ CUs.

*Fusion module.* The fusion module combines high-frequency representations obtained from the differential module $\mathbf{D}_t$ with projected input representations $\mathbf{P}_t$ and produces restored frame $\mathbf{O}_t$ and latent frame $\mathbf{H}_t$. We first add $\mathbf{D}_t$ with $\mathbf{P}_t$ to enhance high-frequency components and then feed the resultant tensor to a fusion module. If the spatial dimensions of $\mathbf{O}_t$ are not the same as $\mathbf{I}_t$ (e.g., in super-resolution), the output of fusion module is up-sampled using a pixel-shuffle operation. Otherwise, an identity operation is performed. The resultant output is then convolved with a $3 \times 3$ convolutional layer to produce $\mathbf{O}_t$. In parallel, the output of fusion layer is also projected using a point-wise convolutional layer to produce latent frame $\mathbf{H}_t$, which allows to share information between the current and the next time step (Eq. 1). Similar to the alignment and differential module, the fusion module is also an efficient and light-weight encoder-decoder network with $N_F$ CUs.

The operation of differential and fusion module is similar to traditional image enhancement methods (e.g., unsharp mask) [5, 40]. In such approaches, the input image is first smoothed to suppress high-frequency components. Then, a difference between smoothed image and input image is computed to identify high-frequency components, which are then added back to the input to enhance it.

## 3.2 Convolutional Unit (CU)

CNN-based methods for different visual recognition tasks learns representations using either a single branch (e.g., ResNet [13] and MobileNets [19, 45]) or multiple branches (e.g., InceptionNets [47, 48] and ESPNets [37, 38]) convolutional units. This work studies these two methods for learning representations. For learning representations at a single scale, we use a depth-wise convolutional layer with $7 \times 7$ kernel while for learning representations at multiple scales, we apply three depth-wise convolutional layers simultaneously ($3 \times 3$, $5 \times 5$, and $7 \times 7$). In both of these methods, the effective receptive field is the same, i.e., $7 \times 7$. Following recent efficient architectures (e.g., MobileNetv3 [18]), we also adopt squeeze-excitation unit (SE unit) [20] to model channel inter-dependencies. Figure 3b sketches the single and multi-scale CUs.

# 4 EXPERIMENTAL RESULTS

To demonstrate the effectiveness of EVRNet on video restoration tasks, we evaluate its performance on three video restoration tasks: (1) deblocking (Section 4.2), (2) denoising (Section 4.3), and (3) super-resolution (Section 4.4). In this section, we first describe the experimental set-up and then evaluate the performance of EVRNet on each of these tasks.

## 4.1 Experimental Set-up

*Tasks.* We study three video restoration tasks: (1) **Video deblocking** aims at removing artifacts that may arise due to video compression, (2) **Video denoising** aims at removing noise (e.g., adaptive white gaussian noise (AWGN)) which may be induced during video transmission, and (3) **Video super-resolution** which aims at upsampling low-resolution video to high-resolution at receiver's end.

*Dataset.* To evaluate the performance of EVRNet, we use large-scale Vimeo-90K dataset [58] which consists of about 90K independent and diverse video shots with both indoor and outdoor lighting

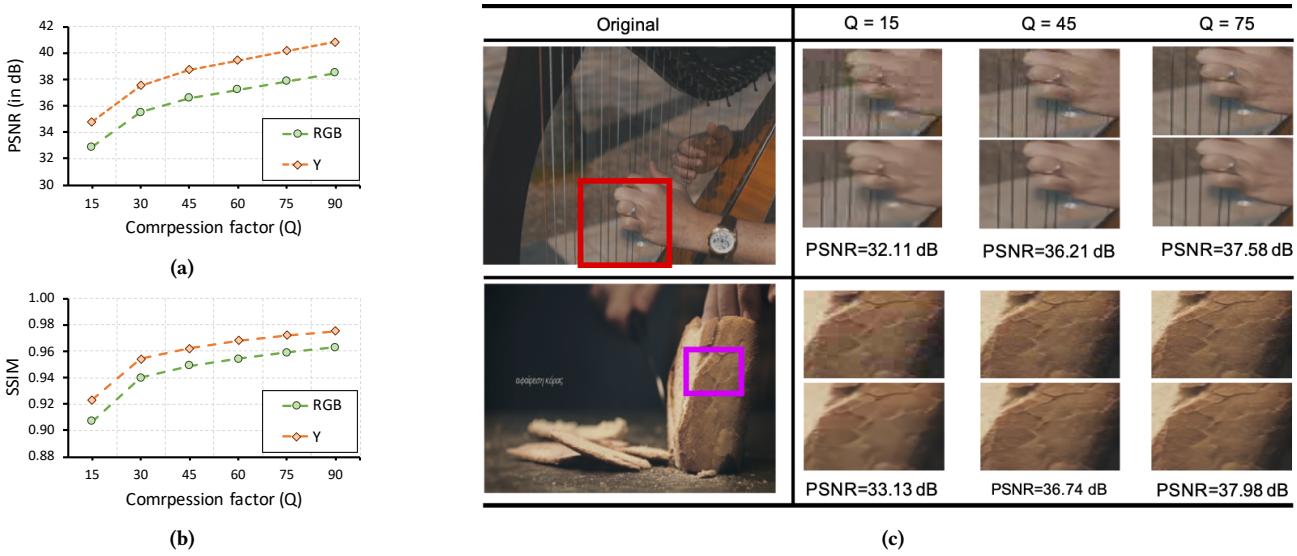



**Figure 5: Performance of EVRNet under compression artifacts. In (a, b), performance in terms of PSNR and SSIM is measured as a function of compression factor $Q$ on both RGB and Y frames, respectively. Lower value of $Q$ means higher compression. In (c), qualitative results for two sample images are shown at different value of $Q$. The top and bottom panels corresponds to the compressed frame and restored frames, respectively. Here, PSNR values are computed on RGB frames.**

scenarios. We use official training and test splits. Note that, for monitoring the training process, we split the training set randomly into two subsets using 90:10 ratio. The first subset is used for training while the second subset is used for validation.

*Training.* EVRNet models are trained by minimizing L1 loss using ADAM optimizer [26] for 50 epochs (or about 50K iterations) using PyTorch. Based on our ablation experiments in Section 6, we set $N_A = 5$, $N_D = 2$, and $N_F = 2$. The learning rate is increased linearly from $1e^{-7}$ to $1e^{-3}$ in first 100 iterations and is then annealed by half at 15-, 25-, 35-, and 45-th epochs. We train EVRNet with an effective batch size of 64 (8 sequences per GPU x 8 GPUs) and use a L2 weight decay of $1e^{-6}$. All our convolutional layers are followed by a PReLU activation [12], except the activation in multi-scale block is after the addition operation. Standard augmentation methods, such as random crop, random flipping, random gamma correction, and random rotation, are used during training. Task-specific augmentation methods are included in respective sub-sections. For comparison with existing methods, we use official splits for deblocking, denoising, and super-resolution while for sensitivity studies, we use functions from OpenCV and Skimage libraries.

*Evaluation metrics.* We use two standard quantitative metrics: (1) peak signal-to-noise ratio (PSNR) and (2) structural similarity index (SSIM). Higher value of PSNR and SSIM indicates better performance. Following previous methods, we report these metrics on RGB and Y channel (YCbCr color space).

## 4.2 Video Deblocking

*Sensitivity study.* We train and evaluate the EVRNet on the task of deblocking artifacts. Similar to state-of-the-art methods (e.g., [32, 58]), we compress frames using JPEG2000 compression. During training, we randomly select the compression or quality factor ($Q$) between 10 and 40. During evaluation, we vary the value of $Q$ from 15 to 90 using OpenCV. Smaller value of $Q$ indicates higher compression or more blocking artifacts. Note that the same EVRNet network is evaluated at different values of $Q$.

Figure 5 shows quantitative and qualitative results under different values of $Q$. The quantitative results in Figure 5a and Figure 5b for both RGB and Y-channel shows that EVRNet is robust to compression. For example, at $Q = 15$, EVRNet is able to achieve PSNR and SSIM values (RGB space) of 33 dB and 0.91, respectively, indicating that it can generate good quality frames even under high compression. These quantitative results are further strengthened with the qualitative results in Figure 5c. The compression artifacts around the hand and strings of harp in the first row and bread loaf in the second row of Figure 5c are completely removed by EVRNet, even under high compression.

*Comparison with state-of-the-art methods* Table 1 compares the performance of EVRNet with state-of-the-art deblocking methods (ARCNN [6], DnCNN [62], V-BM4D [35], ToFlow [58], and DKFN [32]) on the official Vimeo-90K test set. EVRNet delivers similar or better performance than existing methods while having significantly fewer network parameters and multiplication-addition operations (MACs). For example, EVRNet delivers the similar performance as ToFlow [58], but has 46× fewer MACs and 13.64× fewer parameters.

## 4.3 Video Denoising

*Sensitivity study.* Following state-of-the-art methods, we train and evaluate EVRNet under three noise types: (1) Additive White Gaussian Noise (AWGN), (2) Salt and Pepper noise (S&P), and (3) mixture of AWGN and S&P. During training, we randomly select the variance of AWGN noise $\sigma^2$ between 0.05 and 0.4 and the density of

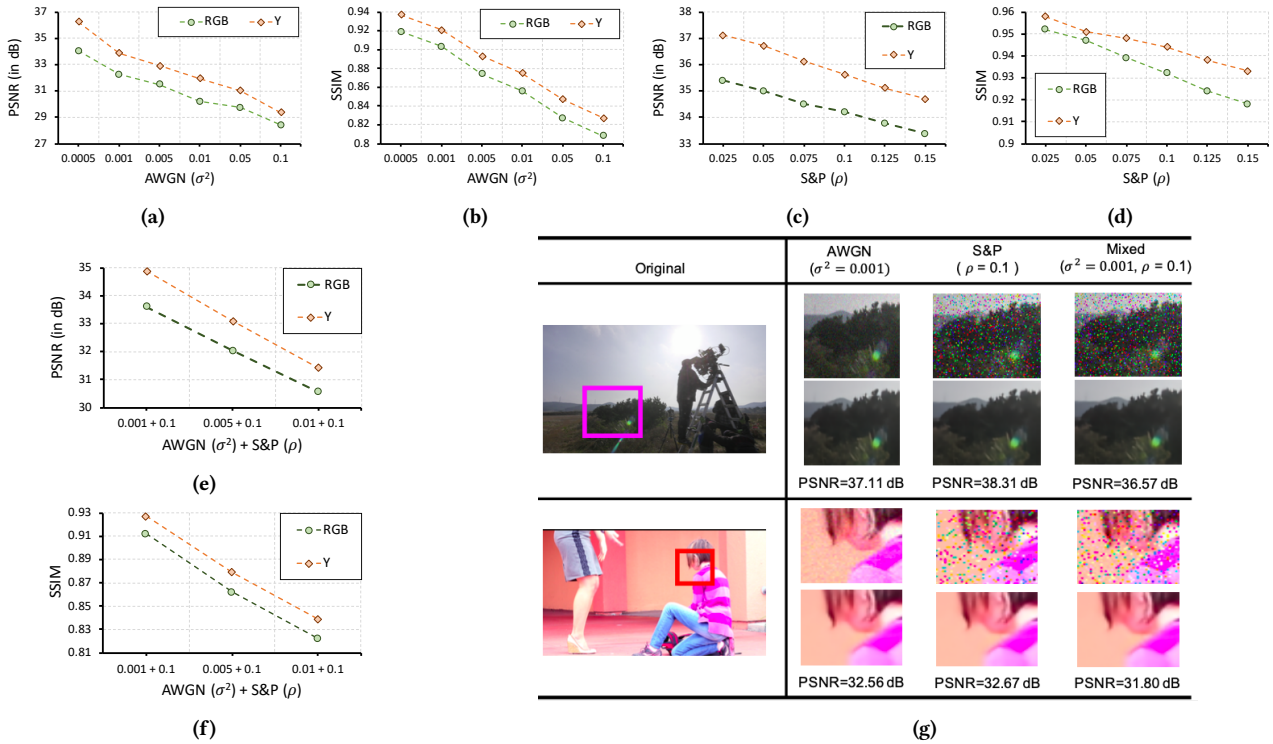



**Figure 6: Performance of EVRNet under noise artifacts. In (a, b), performance in terms of PSNR and SSIM is measured as a function of AWGN noise varience $\sigma^2$ on both RGB and Y frames, respectively. Similarly, in (c, d) and (e, f), performance curves are drawn for salt and pepper noise (S&P) density $\rho$ and mixed noise (AWGN + S&P). Lower value of $\sigma^2$ and $\rho$ means less noise. In (g), qualitative results for two sample images are shown for different types of noise. The top and bottom panels corresponds to the noisy and restored frames, respectively. Here, PSNR values are for RGB frames.**

| Method | MACs | # Params | PSNR↑ | SSIM↑ |
|---|---|---|---|---|
| ARCNN† [6] | 27.73 G | 117.73 K | 36.11 | 0.960 |
| DnCNN† [62] | 128.64 G | 558.34 K | 37.26 | 0.967 |
| V-BM4D [35] | – | – | 35.75 | 0.959 |
| ToFlow [58] | 466.83 G | 1073.48 K | 36.92 | 0.966 |
| DKFN [32] | – | – | 37.93 | 0.971 |
| EVRNet (Ours) | 10.13 G | 78.71 K | 36.65 | 0.967 |

**Table 1: Comparison with existing methods on the task of video deblocking. EVRNet delivers similar or better performance, but with significantly fewer parameters and multiplication-addition operations (MACs). Similar to previous works, we report results in RGB colorspace on the official Vimeo-90K compressed test set where frames are compressed using FFMPEG [58]. The results of methods marked with † are reported in [32] while V-BM4D's performance is reported in [58]. However, MACs and # params of [35] and [32] are not available because code is not open sourced. MACs are measured for $640 \times 360$ RGB frame.**

S&P noise $\rho$ between 0.05 and 0.3. Here, $\sigma$ represents the standard deviation and the value of $\rho$ measures the percentage of pixels randomly replaced with noise. For example, $\rho = 0.3$ indicates that 30% of pixels in a frame are randomly replaced with S&P noise. During evaluation, we first study the effect of AWGN (Figure 6a and 6b) and S&P (Figure 6c and 6d) independently. For AWGN, we vary $\sigma^2$ between 0.0005 and 0.1 while for S&P, we vary $\rho$ between 0.025 and 0.15. We then study the effect of mixture of AWGN and S&P noise (Figure 6e and 6f). In these experiments, we set $\rho = 0.1$ and vary $\sigma^2$ between 0.001 and 0.1. Note that we train only one EVRNet network for video denoising and then evaluate it at different settings of AWGN, S&P, and mixed noise. The quantitative results in Figure 6 shows that EVRNet is robust to different types and amounts of noise. For example, the RGB PSNR values of EVRNet with AWGN noise ($\sigma^2 = 0.001$; Figure 6a), S&P noise ($\rho = 0.1$; Figure 6c), and mixed noise ($\sigma^2 = 0.001$ and $\rho = 0.1$; Figure 6e) are around 33 dB, showing the robustness of EVRNet to different types of noise. This is further demonstrated qualitatively in Figure 6g. In the first and second row of Figure 6g, we can see that EVRNet is able to remove noise and also, restore fine details (e.g., hairs in the second row) under different types of noise.

*Comparison with state-of-the-art methods.* Most state-of-the-art methods train denoising models on Vimeo-90K dataset and evaluate on Vid4 dataset [30]. Following these works, we adopt the

| Method | MACs | # Params | PSNR↑ | SSIM↑ |
|---|---|---|---|---|
| ToFlow [58] | 466.83 G | 1073.48 K | 33.51 | 0.939 |
| EVRNet (Ours) | 10.13 G | 78.71 K | 32.37 | 0.921 |

**(a) Vimeo-90K official test set**

| Method | MACs | # Params | PSNR↑ |
|---|---|---|---|
| V-BM4D$^{\dagger}$ [35] | – | – | 26.31 |
| DnCNN$^{\dagger}$ [62] | 128.64 G | 588.34 K | 26.64 |
| N2V $^{\star\dagger}$ [27] | 140.61 G | 27.90 M | 25.17 |
| N2N+F2F [61] | – | – | 26.56 |
| EVRNet (Ours) | 10.13 G | 78.71 K | 25.79 |

**(b) Vid4 dataset**


**(c) Qualitative denoising results using EVRNet on Vid4 dataset.**

**Table 2: Comparison with state-of-the-art methods on the task of video denoising. EVRNet is able to denoise videos efficiently. Similar to previous works, we report results in RGB colorspace. Here, $^{\dagger}$ represents results are from [61] and $^{\star}$ represents that the number of MACs and parameters are computed for U-Net [43] as N2V is built on top of U-Net. On Vid4 dataset, previous works have not reported SSIM. Therefore, we do not report SSIM on Vid4 dataset.**

| Method | Up-sampling | MACs | # Params | PSNR↑ | SSIM↑ |
|---|---|---|---|---|---|
| ToFlow [58] | 4× | 466.83 G | 1073.48 K | 34.83 | 0.922 |
| DUF [24] | 4× | – | – | 36.37 | 0.939 |
| RBPN [11] | 4× | 29.62 T | 12.77 M | 37.07 | 0.944 |
| EDVR [54] | 4× | 9.96 T | 20.10 M | 37.61 | 0.949 |
| EVRNet (Ours) | 4× | 10.39 G | 79.55 K | 35.98 | 0.931 |
| EVRNet (Ours) | 2× | 10.13 G | 78.71 K | 37.86 | 0.965 |

**Table 3: Comparison with state-of-the-art methods on the task of super-resolution. EVRNet delivers competitive performance to existing methods, but with significantly fewer multiplication-addition operations (MACs) and network parameters. Similar to previous works, we report results in Y-channel on the official Vimeo-90K test set.**

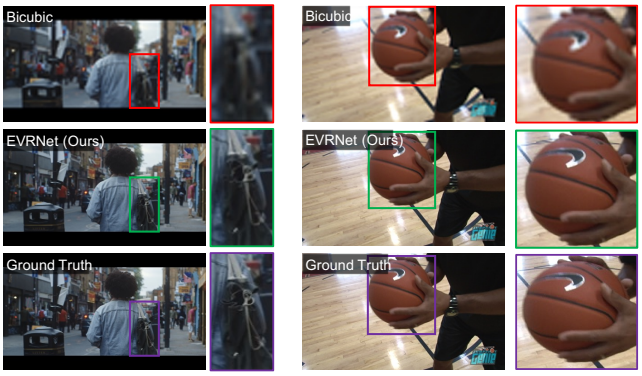



**Figure 7: Qualitative comparison of EVRNet with bicubic up-sampling on the task of 4× video super-resolution. EVRNet is able to restore fine details (e.g., bag pack) which are hard to restore with bicubic interpolation.**

same strategy and evaluate on Vid4 dataset. We also compare EVRNet with ToFlow on the official Vimeo-90K denoising dataset. Results are shown in Table 2. EVRNet delivers competitive performance to state-of-the-art methods, but with significantly fewer MACs and parameters. It is worth mentioning that some existing methods (e.g., ToFlow [58] and N2N + F2F [61]) use optical flow, which is either computationally expensive or requires specialized accelerators. Unlike these methods, EVRNet does not requires any flow estimation and is suitable for edge devices.

### 4.4 Video Super-resolution

We train and evaluate EVRNet on video super-resolution (2× and 4×) task. For training EVRNet that upsamples the input by 2×, we randomly crop a patch whose size lies in the range: {128, 144, 160, 176, 192 }. For 4× model, we finetune 2× model and select random patch size in the range: {64, 72, 80, 88, 96}.

Table 3 shows that EVRNet delivers competitive performance as compared to existing methods, but with significantly fewer parameters and MACs. For example, the SSIM score of EVRNet is 0.018 lower than the EDVR, but has 260× and 958× fewer parameters and MACs, respectively. We note that EVRNet has slightly lower PSNR (about 1.6 dB) as compared to EDVR, however, it is robust to structural changes that occurs due to distortion (as reflected by high SSIM score and qualitative results in Figure 7).[1]

## 5 DISCUSSION

*Generalization to unseen dataset.* A video transmission system, shown in Figure 1a, compresses the video stream before transmitting to the destination in order to reduce network bandwidth. At the destination, the decoded video stream is of low quality due to compression and transmission noise, and is restored using the video restoration methods. To demonstrate the effectiveness of EVRNet in real-world applications (e.g., real-time video conferencing), we trained "multi-task" EVRNet model that is capable of denoising and deblocking on edge devices (see Figure 1a). To train this model, we used the same training and validation sets as mentioned in Section 4, with an exception to inputs to the model. During training, the input sequences were randomly compressed ($Q \in [10, 40]$). After that, mixed noise ($\sigma^2 \in [0.001, 0.01]$ and $\rho \in [0.025, 0.15]$) is added to synthesize transmission noise. Each sequence in Vimeo-90K dataset comprises of 8 frames, has a fixed spatial resolution of

[1]PSNR, though a widely used metric for image quality assessment, does not account for structural changes, which SSIM accounts for. Therefore, for holistic evaluation, both PSNR and SSIM should be considered [17].

| Seq. Id | # Frames | File Size | | RGB | | Y-Channel | |
|---|---|---|---|---|---|---|---|
| | | Original | Compressed | PSNR↑ | SSIM↑ | PSNR↑ | SSIM↑ |
| Seq-1 | 200 | 10.70 MB | 1.43 MB | 37.930 | 0.966 | 39.405 | 0.973 |
| Seq-2 | 200 | 35.54 MB | 4.60 MB | 35.662 | 0.963 | 36.730 | 0.971 |
| Seq-3 | 200 | 36.07 MB | 4.74 MB | 35.880 | 0.962 | 36.713 | 0.967 |
| Seq-4 | 915 | 56.66 MB | 9.28 MB | 38.320 | 0.976 | 39.656 | 0.981 |
| Seq-5 | 366 | 11.40 MB | 8.05 MB | 40.386 | 0.978 | 42.536 | 0.984 |
| Seq-6 | 821 | 10.57 MB | 7.24 MB | 38.775 | 0.974 | 40.903 | 0.979 |
| **Avg.** | | | | 37.826 | 0.970 | 39.324 | 0.976 |

**Table 4: Quantitative results on unseen videos. For generating videos with artifacts, videos are first compressed using H264 compression method. A mixed noise (AWGN with $\sigma^2 = 0.001$ and S&P with $\rho = 0.1$) is then added to synthesize transmission noise. EVRNet is able to remove these artifacts, as is evident in Figure 1b. For more results, see supplementary material.**

| Input size | 240p | | 360p | | 480p | |
|---|---|---|---|---|---|---|
| Output size | 240p | 480p | 360p | 720p | 480p | 960p |
| **iPhone XS** | 12.7 | 12.8 | 7.2 | 7.8 | 4.2 | 4.2 |
| **iPhone 11** | 20.6 | 20.4 | 9.2 | 9.1 | 5.6 | 5.7 |

**Table 5: EVRNet's speed (in FPS) on edge devices. Each data point is an average across 100 iterations and is measured with background applications running on smartphones.**

$448 \times 256$, and are compressed frame-by-frame. Therefore, to test the ability of EVRNet in modeling variable-length sequences under both camera and object motion, we evaluated its performance on six high-definition and diverse video sequences that are captured using different mobile devices (see Table 4). For evaluation, we first compressed these videos using H264 encoding and then added a mixed noise (AWGN with $\sigma^2 = 0.001$ and S&P with $\rho = 0.1$). Both quantitative (Table 4) and qualitative (Figure 1b) results shows that EVRNet (1) can model variable-length sequences and (2) generalizes to unseen videos.

*Run-time on edge devices.* Typically, video conference applications for edge devices, such as Facebook messenger, processes 240p and 360p videos at 10-15 frames per second (FPS) because most of these devices are battery-driven and with higher frame rates, battery would drain out quickly, posing practical implications. To demonstrate the applicability of EVRNet on edge devices, we measured it's inference time on two iOS devices: (1) iPhone XS and (2) iPhone 11. Table 5 shows that EVRNet runs in real-time. We would like to highlight that CoreML (Apple's ML engine) does not support PixelShuffle on the accelerator. To do that operation, we used a solution that uses reshape and transpose operations. These operations are performed on iPhone's CPU (23% CPU occupancy), which resulted in drop in speed. Also, EVRNet is faster on iPhone 11 in comparison to iPhone XS. We believe that accelerator-specific implementations of PixelShuffle along with advancements in hardware technology would further improve the speed of EVRNet on edge devices.

| CU Type | SE Unit | MACs | # Params | RGB | | Y-Channel | |
|---|---|---|---|---|---|---|---|
| | | | | PSNR↑ | SSIM↑ | PSNR↑ | SSIM↑ |
| Single | ✗ | 9.85 G | 68.15 K | 31.207 | 0.868 | 32.650 | 0.886 |
| Single | ✓ | 9.85 G | 72.95 K | 32.006 | 0.896 | 33.365 | 0.914 |
| Multi | ✗ | 10.79 G | 73.91 K | 29.026 | 0.875 | 30.247 | 0.895 |
| Multi | ✓ | 10.79 G | 78.71 K | 32.370 | 0.900 | 33.679 | 0.916 |

**(a) Effect of different CU units**

| Module depth | | | MACs | # Params | RGB | | Y-Channel | |
|---|---|---|---|---|---|---|---|---|
| $N_A$ | $N_D$ | $N_F$ | | | PSNR↑ | SSIM↑ | PSNR↑ | SSIM↑ |
| 1 | 1 | 7 | 11.44 G | 78.71 K | 31.605 | 0.887 | 32.913 | 0.905 |
| 1 | 7 | 1 | 11.44 G | 78.71 K | 31.753 | 0.884 | 32.951 | 0.901 |
| 7 | 1 | 1 | 9.47 G | 78.71 K | 30.859 | 0.871 | 32.139 | 0.890 |
| 2 | 2 | 5 | 11.11 G | 78.71 K | 32.139 | 0.901 | 33.477 | 0.919 |
| 2 | 5 | 2 | 11.11 G | 78.71 K | 32.057 | 0.891 | 33.445 | 0.908 |
| 5 | 2 | 2 | 10.13 G | 78.71 K | 32.403 | 0.903 | 33.884 | 0.921 |
| 3 | 2 | 4 | 10.77 G | 78.71 K | 31.690 | 0.890 | 33.047 | 0.908 |
| 3 | 4 | 2 | 10.77 G | 78.71 K | 30.785 | 0.874 | 32.193 | 0.896 |
| 4 | 3 | 2 | 10.46 G | 78.71 K | 31.416 | 0.877 | 32.690 | 0.895 |
| 3 | 3 | 3 | 10.79 G | 78.71 K | 32.370 | 0.900 | 33.679 | 0.916 |

**(b) Effect of depth of alignment, differential, and fusion modules**

**Table 6: Ablation studies on the task of AWGN denoising ($\sigma^2 = 0.001$). Overall, EVRNet with multi-scale CUs + SE unit and deeper alignment modules provides the best trade-off between performance and MACs.**

## 6 ABLATIONS

*Effect of different CUs.* Table 6a studies the effect of single- and multi-scale convolutional units (CUs) with and without SE unit on the task of AWGN denoising. Multi-scale CU units with SE improves the performance. We hypothesize that this is because AWGN noise is identically distributed in the frames and kernels at different scales helps learn better representations and remove noisy artifacts (see gray color row in Table 6a).

*Effect of the depth of alignment, differential, and fusion modules.* Table 6b studies EVRNet with different values of $N_A$, $N_D$, and $N_F$. We are interested in efficient networks for edge devices, therefore, we studied only those combinations that satisfies this criteria: $N_A + N_D + N_F = 9$. We found that deeper alignment modules delivers the best trade-off between performance and MACs. Therefore, in our main experiments, we used $N_A = 5$, $N_D = 2$, and $N_F = 2$ (see gray color row in Table 6b).

We perform similar studies for deblocking and super-resolution tasks (see supplementary material). We do not observe much gains with different CUs as well as varying the depth of alignment, differential, and fusion modules.

## 7 CONCLUSION

This work introduces EVRNet, a simple neural network that can be used for different video restoration tasks on edge devices, such as deblocking, denoising, and super-resolution. Compared to state-of-the-art video restoration models, EVRNet is more efficient and runs in real-time on edge devices while delivering competitive performance across different tasks.

# REFERENCES


[1] Renzo Andri, Lukas Cavigelli, Davide Rossi, and Luca Benini. 2018. Yodann: An architecture for ultralow power binary-weight cnn acceleration. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2018).

[2] Wenbo Bao et al. 2019. Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement. *IEEE transactions on pattern analysis and machine intelligence* (2019).

[3] Jose Caballero et al. 2017. Real-time video super-resolution with spatio-temporal networks and motion compensation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4778–4787.

[4] François Chollet. 2017. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1251–1258.

[5] Guang Deng. 2010. A generalized unsharp masking algorithm. *IEEE transactions on Image Processing* 20, 5 (2010), 1249–1261.

[6] Chao Dong, Yubin Deng, Chen Change Loy, and Xiaoou Tang. 2015. Compression artifacts reduction by a deep convolutional network. In *Proceedings of the IEEE International Conference on Computer Vision*. 576–584.

[7] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2014. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*. Springer, 184–199.

[8] Alexey Dosovitskiy et al. 2015. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*. 2758–2766.

[9] Saurabh Gupta, Judy Hoffman, and Jitendra Malik. 2016. Cross modal distillation for supervision transfer. In *CVPR*. 2827–2836.

[10] Song Han, Huizi Mao, and William J Dally. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *ICLR*.

[11] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. 2019. Recurrent back-projection network for video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3897–3906.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*. 1026–1034.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[14] Yihui He et al. 2018. Amc: Automl for model compression and acceleration on mobile devices. In *ECCV*.

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780.

[17] Alain Hore and Djemel Ziou. 2010. Image quality metrics: PSNR vs. SSIM. In *2010 20th international conference on pattern recognition*. IEEE, 2366–2369.

[18] Andrew Howard et al. 2019. Searching for mobilenetv3. In *Proceedings of the IEEE International Conference on Computer Vision*. 1314–1324.

[19] Andrew G Howard et al. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).

[20] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7132–7141.

[21] Yan Huang, Wei Wang, and Liang Wang. 2015. Bidirectional recurrent convolutional networks for multi-frame super-resolution. In *Advances in Neural Information Processing Systems*. 235–243.

[22] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2016. Binarized neural networks. In *NIPS*.

[23] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. 2017. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[24] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. 2018. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3224–3232.

[25] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. 2016. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1646–1654.

[26] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[27] Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. 2019. Noise2void-learning denoising from single noisy images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2129–2137.

[28] Christian Ledig et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4681–4690.

[29] Chong Li and CJ Richard Shi. 2018. Constrained Optimization Based Low-Rank Approximation of Deep Neural Networks. In *ECCV*.

[30] Ce Liu and Deqing Sun. 2011. A bayesian approach to adaptive video super resolution. In *CVPR 2011*. IEEE, 209–216.

[31] Ding Liu, Zhaowen Wang, Yuchen Fan, Xianming Liu, Zhangyang Wang, Shiyu Chang, and Thomas Huang. 2017. Robust video super-resolution with learned temporal dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*. 2507–2515.

[32] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Zhiyong Gao, and Ming-Ting Sun. 2018. Deep kalman filtering network for video compression artifact reduction. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 568–584.

[33] Bruce D Lucas, Takeo Kanade, et al. 1981. An iterative image registration technique with an application to stereo vision. (1981).

[34] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. 2018. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*. 116–131.

[35] Matteo Maggioni, Giacomo Boracchi, Alessandro Foi, and Karen Egiazarian. 2012. Video denoising, deblocking, and enhancement through separable 4-D nonlocal spatiotemporal transforms. *IEEE Transactions on image processing* 21, 9 (2012), 3952–3966.

[36] Sachin Mehta, Hannaneh Hajishirzi, and Mohammad Rastegari. 2020. DiCENet: Dimension-wise Convolutions for Efficient Networks. *IEEE transactions on pattern analysis and machine intelligence* (December 2020).

[37] Sachin Mehta, Mohammad Rastegari, Anat Caspi, Linda Shapiro, and Hannaneh Hajishirzi. 2018. Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In *Proceedings of the european conference on computer vision (ECCV)*. 552–568.

[38] Sachin Mehta, Mohammad Rastegari, Linda Shapiro, and Hannaneh Hajishirzi. 2019. Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 9190–9200.

[39] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. 2019. Importance estimation for neural network pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 11264–11272.

[40] Andrea Polesel, Giovanni Ramponi, and V John Mathews. 2000. Image enhancement via adaptive unsharp masking. *IEEE transactions on image processing* 9, 3 (2000), 505–510.

[41] Anurag Ranjan and Michael J Black. 2017. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4161–4170.

[42] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*.

[43] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 234–241.

[44] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. 2018. Frame-recurrent video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6626–6634.

[45] Mark Sandler et al. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4510–4520.

[46] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. 2018. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[47] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. 2016. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261* (2016).

[48] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[49] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. 2019. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2820–2828.

[50] Mingxing Tan and Quoc V Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946* (2019).

[51] Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. 2017. Detail-revealing deep video super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*. 4472–4480.

[52] Yapeng Tian, Yulun Zhang, Yun Fu, and Chenliang Xu. 2020. TDAN: Temporally-Deformable Alignment Network for Video Super-Resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3360–3369.

[53] Tong Tong, Gen Li, Xiejie Liu, and Qinquan Gao. 2017. Image super-resolution using dense skip connections. In *Proceedings of the IEEE International Conference on Computer Vision*. 4799–4807.

[54] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. 2019. Edvr: Video restoration with enhanced deformable convolutional networks. In

*Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 0–0.

[55] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. 2018. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 0–0.

[56] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. In *NIPS*.

[57] Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Jian Cheng. 2016. Quantized convolutional neural networks for mobile devices. In *CVPR*.

[58] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. 2019. Video enhancement with task-oriented flow. *International Journal of Computer Vision* 127, 8 (2019), 1106–1125.

[59] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. 2017. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *CVPR*. 4133–4141.

[60] Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. 2018. Nisp: Pruning networks using neuron importance score propagation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 9194–9203.

[61] Songhyun Yu, Bumjun Park, Junwoo Park, and Jechang Jeong. 2020. Joint Learning of Blind Video Denoising and Optical Flow Estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 500–501.

[62] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. 2017. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing* 26, 7 (2017), 3142–3155.

[63] Barret Zoph and Quoc V Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578* (2016).