Recurrent Back-Projection Network for Video Super-Resolution

Muhammad Haris[1], Greg Shakhnarovich[2], and Norimichi Ukita[1]
[1]Toyota Technological Institute, Japan [2]Toyota Technological Institute at Chicago, United States
{mharis, ukita}@toyota-ti.ac.jp, greg@ttic.edu

Abstract

We proposed a novel architecture for the problem of video super-resolution. We integrate spatial and temporal contexts from continuous video frames using a recurrent encoder-decoder module, that fuses multi-frame information with the more traditional, single frame super-resolution path for the target frame. In contrast to most prior work where frames are pooled together by stacking or warping, our model, the Recurrent Back-Projection Network (RBPN) treats each context frame as a separate source of information. These sources are combined in an iterative refinement framework inspired by the idea of back-projection in multiple-image super-resolution. This is aided by explicitly representing estimated inter-frame motion with respect to the target, rather than explicitly aligning frames. We propose a new video super-resolution benchmark, allowing evaluation at a larger scale and considering videos in different motion regimes. Experimental results demonstrate that our RBPN is superior to existing methods on several datasets.

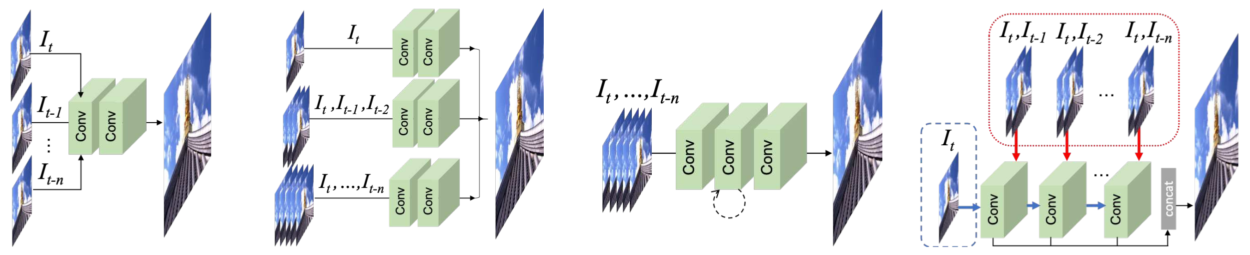
1. Introduction

The goal of super-resolution (SR) is to enhance a low-resolution (LR) image to higher resolution (HR) by filling missing fine details in the LR image. This field can be divided into Single-Image SR (SISR) [4, 8, 9, 19, 21, 29], Multi-Image SR (MISR) [5, 6], and Video SR (VSR) [2, 30, 27, 16, 13, 25], the focus of this paper.

Consider a sequence of LR video frames $I_{t-n}, \ldots, I_t, \ldots, I_{t+n}$, where we super-resolve a *target frame*, I_t. While I_t can be super-resolved independently of other frames as SISR, this is wasteful of missing details available from the other frames. In MISR, the missing details available from the other frames are fused for super-resolving I_t. For extracting these missing details, all frames must be spatially aligned explicitly or implicitly. By separating differences between the aligned frames from missing details observed only in one or some of the frames, the missing details are extracted. This alignment is required to be very precise (e.g., sub-pixel accuracy) for SR. In MISR, however, the frames are aligned independently with no cue given by temporal smoothness, resulting in difficulty in the precise alignment. Yet another approach is to align the frames in temporal smooth order as VSR.

In recent VSR methods using convolutional networks, the frames are concatenated [22, 16] or fed into recurrent networks (RNNs) [13]) in temporal order; no explicit alignment is performed. The frames can be also aligned explicitly, using motion cues between temporal frames with the alignment modules [25, 2, 30, 27]. These latter methods generally produce results superior to those with no explicit spatial alignment [22, 13]. Nonetheless, these VSR methods suffer from a number of problems. In the frame-concatenation approach [2, 16, 25], many frames are processed simultaneously in the network, resulting in difficulty in training the network. In RNNs [30, 27, 13], it is not easy to jointly model subtle and significant changes (e.g., slow and quick motions of foreground objects) observed in all frames of a video even by those designed for maintaining long-term temporal dependencies such as LSTMs [7].

Our method proposed in this paper is inspired by "back-projection" originally introduced in [14, 15] for MISR. Back-projection iteratively calculates residual images as reconstruction error between a target image and a set of its corresponding images. The residuals are back-projected to the target image for improving its resolution. The multiple residuals can represent subtle and significant differences between the target frame and other frames independently. Recently, Deep Back-Projection Networks (DBPN) [8] extended back-projection to Deep SISR under the assumption that only one LR image is given for the target image. In that scenario, DBPN produces a high-resolution feature map, iteratively refined through multiple up- and down-sampling layers. Our method, Recurrent Back-Projection Networks (RBPN), integrates the benefits of the original, MISR back projection and DBPN, for VSR. Here we use other video frames as corresponding LR images for the original MISR back-projection. In addition, we use the idea of iteratively refining HR feature maps representing missing details by up- and down-sampling processes to further improve the



(a) Temporal concatenation (b) Temporal aggregation (c) RNNs (d) Our RBPN

Figure 1. Comparison of Deep VSRs. (a) Input frames are concatenated to preserve temporal information [18, 2, 16, 22]. (b) Temporal aggregation improves (a) to preserve multiple motion regimes [25]. (c) RNNs take a sequence of input frames to produce one SR image at a target frame, I_t [13, 30, 27]. (d) Our recurrent back-projection network accepts I_t, which is enclosed by a blue dashed line, as well as a set of residual features computed from a pairing I_t with other frames (i.e., I_{t-k} for $k \in \{1, \cdots, n\}$), as enclosed by a red dotted line, while previous approaches using RNNs shown in (c) feed all temporal frames one by one along a single path. Residual features computed from the pairs of (I_t, I_{t-k}) (MISR path - the vertical red arrows) are fused with features extracted from variants of I_t (SISR path - the horizontal blue arrows) through RNN.

quality of SR.

Our contributions include the following key innovations.
Integrating SISR and MISR in a unified VSR framework: SISR and MISR extract missing details from different sources. Iterative SISR [8] extracts various feature maps representing the details of a target frame. MISR provides multiple sets of feature maps from other frames. These different sources are iteratively updated in temporal order through RNN for VSR.
Back-projection modules for RBPN: We develop a recurrent encoder-decoder mechanism for incorporating details extracted in SISR and MISR paths through the back-projection. While the SISR path accepts only I_t, the MISR path also accepts I_{t-k} where $k \in [n]$. A gap between I_t and I_{t-k} is larger than the one in other VSRs using RNN (i.e., gap only between I_t and I_{t-1}). Here, the network is able to understand this large gap since each context is calculated separately, rather than jointly as in previous work, this separate context plays an important role in RBPN.
Extended evaluation protocol: We report extensive experiments to evaluate VSR. In addition to previously-standard datasets, Vid4 [24] and SPMCS [30], that lack significant motions, a dataset containing various types of motion (Vimeo-90k [34]) is used in our evaluation. This allows us to conduct a more detailed evaluation of strengths and weaknesses of VSR methods, depending on the type of the input video.

2. Related Work

While SR has an extensive history, our discussion in this section focuses on *deep SR* – SR methods that involve deep neural network components, trained end-to-end.

2.1. Deep Image Super-resolution

Deep SISR is first introduced by SRCNN [4] that requires a predefined upsampling operator. Further improvements include better up-sampling layers [28], residual learning [19, 29], back-projection [8], recursive layers [20], and progressive upsampling [21]. See NTIRE2018 [31] and PIRM2018 [1] for comprehensive comparison.

2.2. Recurrent Networks

Recurrent neural networks (RNNs) deal with sequential inputs and/or outputs, and have been employed for video captioning [17, 26, 35], video summarization [3, 32], and VSR [30, 13, 27]. Two types of RNN have been used for VSR. A many-to-one architecture is used in [30, 13] where a sequence of frames is mapped to a single target HR frame. A synchronous many-to-many RNN has recently been used in VSR by [27], to map a sequence of LR frames to a sequence of HR frames.

2.3. Deep Video Super-resolution

Deep VSR can be primarily divided into three types based on the approach to preserving temporal information as shown in Fig. 1 (a), (b), and (c).
(a) **Temporal Concatenation**. The most popular approach to retain temporal information in VSR is by concatenating the frames as in [18, 2, 16, 22]. This approach can be seen as an extension of SISR to accept multiple input images. VSR-DUF [16] proposed a mechanism to construct up-sampling filters and residual images. However, this approach fails to represent the multiple motion regimes on a sequence because input frames are concatenated together.
(b) **Temporal Aggregation.** To address the dynamic motion problem in VSR, [25] proposed multiple SR inferences which work on different motion regimes. The final layer aggregates the outputs of all branches to construct SR frame. However, this approach basically still concatenates many input frames, resulting in difficulty in global optimization.
(c) **RNNs.** This approach is first proposed by [13] using bidirectional RNNs. However, the network has a small net-

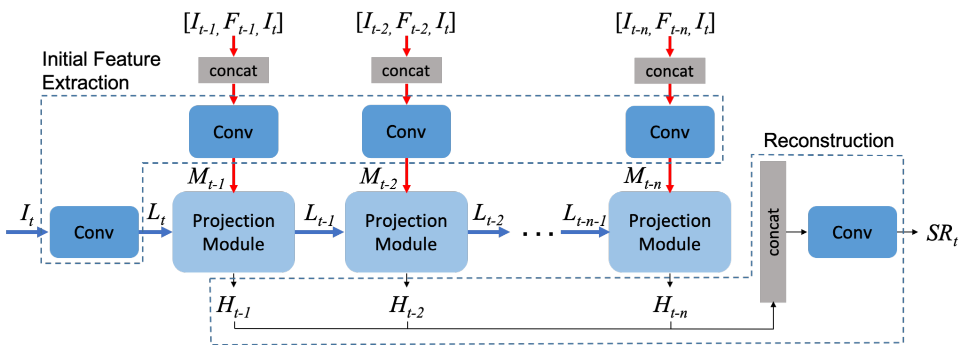


Figure 2. Overview of RBPN. The network has two approaches. The horizontal blue line enlarges I_t using SISR. The vertical red line is based on MISR to compute the residual features from a pair of I_t to neighbor frames ($I_{t-1}, ..., I_{t-n}$) and the precomputed dense motion flow maps ($F_{t-1}, ..., F_{t-n}$). Each step is connected to add the temporal connection. On each projection step, RBPN observes the missing details on I_t and extract the residual features from each neighbor frame to recover the missing details.

work capacity and has no frame alignment step. Further improvement is proposed by [30] using a motion compensation module and a convLSTM layer [33]. Recently, [27] proposed an efficient many-to-many RNN that uses the previous HR estimate to super-resolve the next frames. While recurrent feedback connections utilize temporal smoothness between neighbor frames in a video for improving the performance, it is not easy to jointly model subtle and significant changes observed in all frames.

3. Recurrent Back-Projection Networks

3.1. Network Architecture

Our proposed network is illustrated in Fig. 2. Let I be LR frame with size of $(M^l \times N^l)$. The input is sequence of $n+1$ LR frames $\{I_{t-n}, \dots, I_{t-1}, I_t\}$ where I_t is the target frame. The goal of VSR is to output HR version of I_t, denoted by $\texttt{SR}_t$ with size of $(M^h \times N^h)$ where $M^l < M^h$ and $N^l < N^h$. The operation of RBPN can be divided into three stages: initial feature extraction, multiple projections, and reconstruction. Note that we train the entire network jointly, end-to-end.

Initial feature extraction. Before entering projection modules, I_t is mapped to LR feature tensor L_t. For each neighbor frame among I_{t-k}, $k \in [n]$, we concatenate the precomputed dense motion flow map F_{t-k} (describing a 2D vector per pixel) between I_{t-k} and I_t with the target frame I_t and I_{t-k}. The motion flow map encourages the projection module to extract missing details between a pair of I_t and I_{t-k}. This stacked 8-channel "image" is mapped to a *neighbor feature* tensor M_{t-k} .

Multiple Projections. Here, we extract the missing details in the target frame by integrating SISR and MISR paths, then produce refined HR feature tensor. This stage receives L_{t-k-1} and M_{t-k}, and outputs HR feature tensor H_{t-k}.

Reconstruction. The final SR output is obtained by feeding concatenated HR feature maps for all frames into a reconstruction module, similarly to [8]: $\texttt{SR}_t = f_{rec}([H_{t-1}, H_{t-2}, ..., H_{t-n}])$. In our experiments, f_{rec} is a single convolutional layer.

3.2. Multiple Projection

The multiple projection stage of RBPN uses a recurrent chain of encoder-decoder modules, as shown in Fig. 3. The projection module, shared across time frames, takes two inputs: $L_{t-n-1} \in \mathbb{R}^{M^l \times N^l \times c^l}$ and $M_{t-n} \in \mathbb{R}^{M^l \times N^l \times c^m}$, then produces two outputs: L_{t-n} and $H_{t-n} \in \mathbb{R}^{M^h \times N^h \times c^h}$ where c^l, c^m, c^h are the number of channels for particular map accordingly.

The encoder produces a hidden state of estimated HR features from the projection to a particular neighbor frame. The decoder deciphers the respective hidden state as the next input for the encoder module as shown in Fig. 4 which are defined as follows:

$$\text{Encoder:} \quad H_{t-n} = \texttt{Net}_E(L_{t-n-1}, M_{t-n}; \theta_E) \tag{1}$$

$$\text{Decoder:} \quad L_{t-n} = \texttt{Net}_D(H_{t-n}; \theta_D) \tag{2}$$

The encoder module $\texttt{Net}_E$ is defined as follows:

$$\text{SISR upscale:} \quad H^l_{t-n-1} = \texttt{Net}_{sisr}(L_{t-n-1}; \theta_{sisr}) \tag{3}$$

$$\text{MISR upscale:} \quad H^m_{t-n} = \texttt{Net}_{misr}(M_{t-n}; \theta_{misr}) \tag{4}$$

$$\text{Residual:} \quad e_{t-n} = \texttt{Net}_{res}(H^l_{t-n-1} - H^m_{t-n}; \theta_{res}) \tag{5}$$

$$\text{Output:} \quad H_{t-n} = H^l_{t-n-1} + e_{t-n} \tag{6}$$

3.3. Interpretation

Figure 5 illustrates the RBPN pipeline, for a 3-frame video. In the encoder, we can see RBPN as the combination of SISR and MISR networks. First, target frame is enlarged by $\texttt{Net}_{sisr}$ to produce H^l_{t-k-1}. Then, for each combination of concatenation from neighbor frames and target frame, $\texttt{Net}_{misr}$ performs implicit frame alignment and absorbs the motion from neighbor frames to produce warping

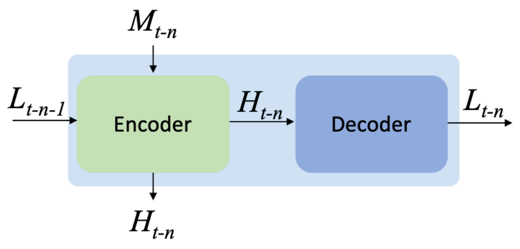
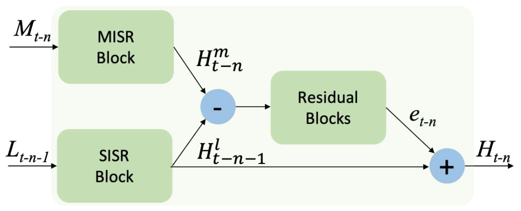
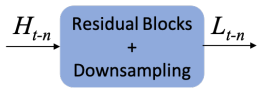


Figure 3. The proposed projection module. The target features (L_{t-n-1}) is projected to neighbor features (M_{t-n}) to construct better HR features (H_{t-n}) and produce next LR features (L_{t-n}) for the next step.



(a) Encoder (the back-projection)



(b) Decoder

Figure 4. Detailed illustration of encoder and decoder. The encoder performs back-projection from L_{t-n-1} to M_{t-n} to produce the residual e_{t-n}.

features H^m_{t-k} which may capture missing details in the target frame. Finally, the residual features e_{t-k} from H^l_{t-k-1} and H^m_{t-k} are fused back to H^l_{t-k-1} to refine the HR features and produce hidden state H_{t-k}. The decoder "deciphers" the hidden state H_{t-k} to be the next input for the encoder L_{t-k}. This process is repeated iteratively until the target frame is projected to all neighbor frames.

The optimal scenario for this architecture is when each frame can contribute to filling in some missing details in the target frame. Then H_{t-k} generated in each step k produce unique features. In the generate case when $n = 0$ (no other frames) or the video is completely static (identical frames) RBPN will effectively ignore the $\texttt{Net}_{misr}$ module, and fall back to a recurrent SISR operation.

4. Experimental Results

In all our experiments, we focus on $4\times$ SR factor.

4.1. Implementation and training details

We use DBPN [8] for $\texttt{Net}_{sisr}$, and Resnet [10] for $\texttt{Net}_{misr}$, $\texttt{Net}_{res}$, and $\texttt{Net}_{D}$. For $\texttt{Net}_{sisr}$, we construct three stages using 8×8 kernel with stride = 4 and pad by 2 pixels. For $\texttt{Net}_{misr}$, $\texttt{Net}_{res}$, and $\texttt{Net}_{D}$, we construct five

Bicubic 1 Frame	DBPN 1 Frame	DBPN-MISR 5 Frames	RBPN-MISR 5 Frames	RBPN 5 Frames
27.13/0.749	29.85/0.837	30.64/0.859	30.89/0.866	31.40/0.877

Table 1. Baseline comparison on SPMCS-32. Red here and in the other tables indicates the best performance (PSNR/SSIM).

blocks where each block consists of two convolutional layers with 3×3 kernel with stride = 1 and pad by 1 pixel. The up-sampling layer in $\texttt{Net}_{misr}$ and down-sampling layer in $\texttt{Net}_{D}$ use 8×8 kernel with stride = 4 and pad by 2 pixels. Our final network uses $c^l = 256$, $c^m = 256$, and $c^h = 64$.

We trained our networks using Vimeo-90k [34], with a training set of 64,612 7-frame sequences, with fixed resolution 448×256. Furthermore, we also apply augmentation, such as rotation, flipping, and random cropping. To produce LR images, we downscale the HR images $4\times$ with bicubic interpolation.

All modules are trained end-to-end using per-pixel L1 loss per-pixel between the predicted frame and the ground truth HR frame. We use batch size of 8 with size 64×64 which is cropped randomly from 112×64 LR image. The learning rate is initialized to $1e-4$ for all layers and decrease by a factor of 10 for half of total 150 epochs. We initialize the weights based on [11]. For optimization, we used Adam with momentum to 0.9. All experiments were conducted using Python 3.5.2 and PyTorch 1.0 on NVIDIA TITAN X GPUs. Following the evaluation from previous approaches [2, 30, 27], we crop 8 pixels near image boundary and remove first six frames and last three frames. All measurements use only the luminance channel (Y).

4.2. Ablation studies

Baselines We consider three baselines, that retain some components of RBPN while removing others. First, we remove all components by $\texttt{Net}_{sisr}$ (DBPN); this ignores the video context. Second, we use DBPN with temporal concatenation (DBPN-MISR). Third, we remove the decoder, thus severing temporal connections, so that our model is reduced to applying back-projection $\texttt{Net}_{misr}$ with each neighboring frame, and concatenating the results; we call this baseline RBPN-MISR. The results are shown in Table 1. Our intuition suggests, and the results confirm, that such an approach would be weaker than RBPN, since it does not have the ability to separately handle changes of different magnitude that RBPN has. As expected, SISR suffers from ignoring extra information in other frames. RBPN-MISR and DBPN-MISR does manage to leverage multiple frames to improve performance, but the best results are obtained by the full RBPN model.

Network setup. The modular design of our approach allows easy replacement of modules; in particular we consider choices of DBPN or ResNet for $\texttt{Net}_{sisr}$, $\texttt{Net}_{misr}$, or both. In Table 2, we evaluate three combinations: RBPN

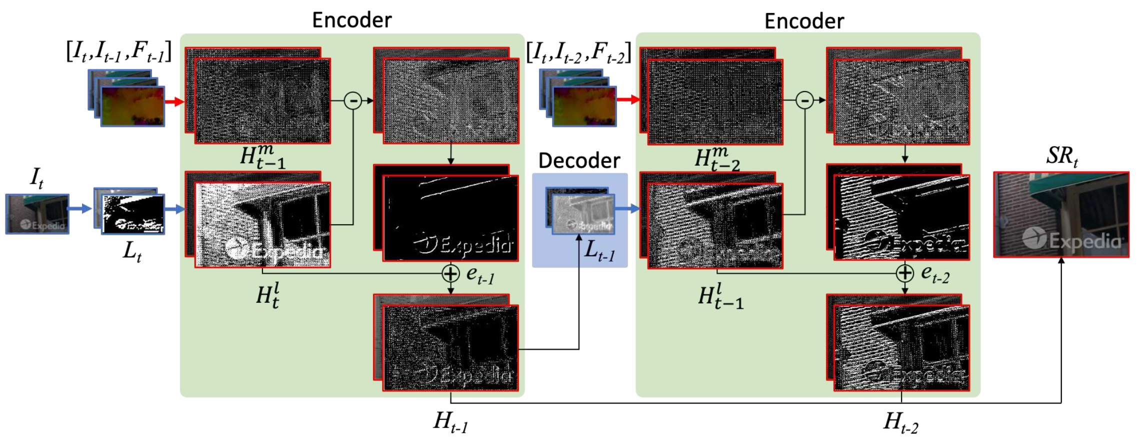


Figure 5. The illustration of each operation in RBPN ($n+1=3$). Zoom in to see better visualization.

DBPN	RESNET	$\mathtt{Net}_{sisr}$=DBPN, $\mathtt{Net}_{misr}$=RESNET
30.54/0.856	30.74/0.862	30.96/0.866

Table 2. Network analysis using RBPN/2 on SPMCS-32.

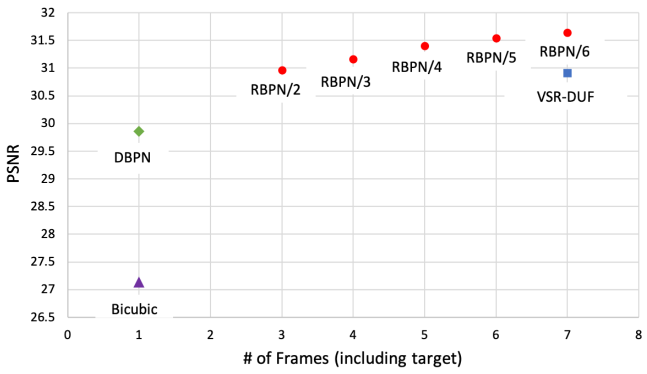


Figure 6. Effect of context (past) length, 4× SR on SPMCS-32. RBPN/$\langle k \rangle$: RBPN trained/tested with k past frames. Note: DBPN is equivalent to RBPN/0.

with DBPN, RBPN with Resnet, and RBPN with the combination of DBPN as $\mathtt{Net}_{sisr}$ and Resnet as $\mathtt{Net}_{misr}$. The latter produces the best results, but the difference are minor, showing stability of RBPN w.r.t. choice of components.

Context length We evaluated RBPN with different lengths of video context, i.e., different number of past frames $n \in \{2, \ldots, 6\}$. Figure 6 shows that performance (measured on (on SPMCS-32 test set) improves with longer context. The performance of RBPN/3 is even better than VSR-DUF as one of state-of-the-art VSR which uses six neighbor frames. It also shows that by adding more frames, the performance of RBPN increase by roughly 0.2 dB.

Fig. 7 provides an illustration of the underlying performance gains. Here, VSR-DUF fails to reconstruct the brick pattern, while RBPN/3 reconstructs it well, even with fewer frames in the context; increasing context length leads to further improvements.

	RBPN/2		RBPN/6	
	RBPN	Last	w/ LSTM	RBPN
PSNR/SSIM	30.96/0.866	30.89/0.864	31.46/0.880	31.64/0.883

Table 3. Comparison of temporal integration strategies on SPMCS-32.

Temporal integration Once the initial feature extraction and the projection modules have produced a sequence of HR feature maps H_{t-k}, $k = 1, \ldots, n$, we can use these maps in multiple ways to reconstruct the HR target. The proposed DBPN concatenates the maps; We also consider an alternative where only the H_{t-n} is fed to $\mathtt{Net}_{rec}$ (referred to as Last). Furthermore, instead of concatenating the maps, we can feed them to a convolutional LSTM [33], the output of which is then fed to $\mathtt{Net}_{rec}$. The results are shown in Table 3. Dropping the concatenation and only using last feature map harms the performance (albeit moderately). Replacing concatenation with an LSTM also reduces the performance (while increasing computational cost). We conclude that the RBPN design depicted in Fig. 2 is better than the alternatives.

Temporal order When selecting frames to serve as context for a target frame t, we have a choice of how to choose and order it: use only past frames (P; for instance, with $n = 6$, this means $I_t, I_{t-1}, \ldots, I_{t-6}$), use both past and future (PF, $I_{t-3}, \ldots, I_t, \ldots, I_{t+3}$), or consider the past frames in random order (PR; we can do this since the motion flow is computed independently for each context frame w.r.t. the target). Table 4 shows that PF is better than P by 0.1 dB; presumably this is due to the increased, more symmetric representation of motion occurring in frame t. Interestingly, when the network is trained on PF, then tested on P (PF$\rightarrow$P), the performance is decreased (-0.17dB), but when RBPN is trained on P then tested on PF (P$\rightarrow$PF), the performance remains almost the same.

The results of comparing order P to random ordering PR are shown in Table 5. Interestingly, RBPN performance is

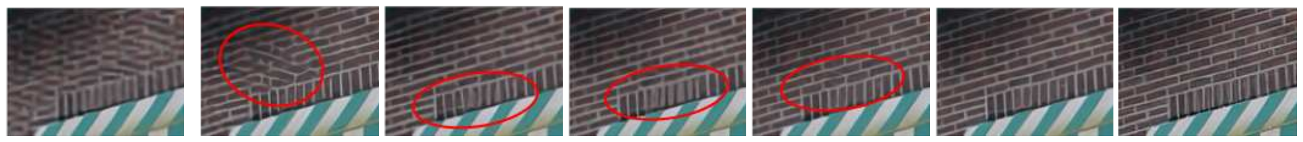

(a) VSR-DUF/6 [16] (b) RBPN/2 (c) RBPN/3 (d) RBPN/4 (e) RBPN/5 (f) RBPN/6 (g) GT

Figure 7. Visual results on different frame length (SPMCS-32). Zoom in to see better visualization.

	P	PF	P $\rightarrow$ PF	PF $\rightarrow$ P
PSNR/SSIM	31.64/0.883	31.74/0.884	31.66/0.884	31.57/0.881

Table 4. Effect of temporal order of context, RBPN/6 on SPMCS-32.

	P	PR	P $\rightarrow$ PR	PR $\rightarrow$ P
PSNR/SSIM	31.40/0.877	31.39/0.876	31.39/0.877	31.35/0.875

Table 5. Effect of temporal order (RBPN/4) on SPMCS-32.

	RBPN/5	
	w/	w/o
PSNR/SSIM	31.54/0.881	31.36/0.878

Table 6. Optical flow (OF) importance on SPMCS-32.

not significantly affected by the choice of order. We attribute this robustness to the decision to associate each context frame with the choice of order.

Optical flow Finally, we can remove the optical flow component of M_{t-k}, feeding the projection modules only the concatenated frame pairs. As Table 6 shows, explicit optical flow representation is somewhat, but not substantially, beneficial. We compute the flow using an implementation of [23].

4.3. Comparison with the-state-of-the-arts

We compare our network with eight state-of-the-art SR algorithms: DBPN [8], BRCN [13], VESPCN [2], $B_{123} + T$ [25], VSR-TOFLOW [34], DRDVSR [30], FRVSR [27], and VSR-DUF [16]. Note: only VSR-DUF and DBPN provide full testing code without restrictions, and most of the previous methods use different training sets. Other methods provide only the estimated SR frames. For RBPN, we use $n = 6$ with PF (past+future) order, which achieves the best results, denoted as RBPN/6-PF.

We carry out extensive experiments using three datasets: Vid4 [24], SPMCS [30], and Vimeo-90k [34]. Each dataset has different characteristics. We found that evaluating on Vid4, commonly reported in literature, has limited ability to assess relative merits of competing approaches; the sequences in this set have visual artifacts, very little inter-frame variation, and fairly limited motion. Most notably, it only consists of *four* video sequences. SPMCS data exhibit more variation, but still lack significant motion. Therefore, in addition to the aforementioned data sets, we consider Vimeo-90k, a much larger and diverse data set, with high-quality frames, and a range of motion types. We stratify the Vimeo-90k sequences according to estimated motion velocities into slow, medium and fast "tiers", as shown in Fig. 8, and report results for these tiers separately.

Table 7 shows the results on Vid4 test set. We also provide the average flow magnitude (pixel/frame) on Vid4. It shows that Vid4 does not contain significant motion. The results also show that RBPN/6-PF is better than the previous methods, except for VSR-DUF. Figure 9 shows some qualitative results on Vid4. (on "Calendar"). The "MAREE" text reconstructed with RBPN/6-PF has sharper images than previous methods. However, here we see that the ground truth (GT) itself suffers from artifacts and aliasing, perhaps due to JPEG compression. This apparently leads in some cases to penalizing sharper SR predictions, like those made by our network, as illustrated in Fig. 9.

Table 8 shows the detailed results on SPMCS-11. RBPN/6-PF has better performance of 0.68 dB and 1.28 dB than VSR-DUF and DRDVSR, respectively. Even with fewer frames in the context, RBPN/4-P has better average performance than VSR-DUF and DRDVSR by 0.33 dB and 0.93 dB, respectively. Qualitative results on SPMCS are shown in Fig. 10. In the first row, we see that RBPN reproduces a well-defined pattern, especially on the stairs area. In the second row, RBPN recovers sharper details and produces better brown lines from the building pattern.

It is interesting to see that VSR-DUF tends to do better on SSIM than on PSNR. It has been suggested that PSNR is more sensitive to Gaussian noise, while SSIM is more sensitive to compression artifacts [12]. VSR-DUF generates up-sampling filter to enlarge the target frame. The use of up-sampling filter can keep overall structure of target frame which tends to have higher SSIM. However, since the residual image produced by VSR-DUF fails to generate the missing details, PSNR tends to be lower. In contrast with VSR-DUF, our focus is to fuse the missing details to the target frame. However, if in some cases we generate sharper pattern than GT, this causes lower SSIM. This phenomenon mainly can be observed in the Vid4 test set.

Table 9 shows the results on Vimeo-90k. RBPN/6-PF outperforms VSR-DUF by a large margin. RBPN/6-PF gets higher PSNR by 1.22 dB, 1.44 dB, and 2.54 dB than VSR-DUF on, respectively, slow, medium, and fast motion. It can be seen that RBPN is able to preserve different temporal scale. RBPN achieves the highest gap relative to prior work

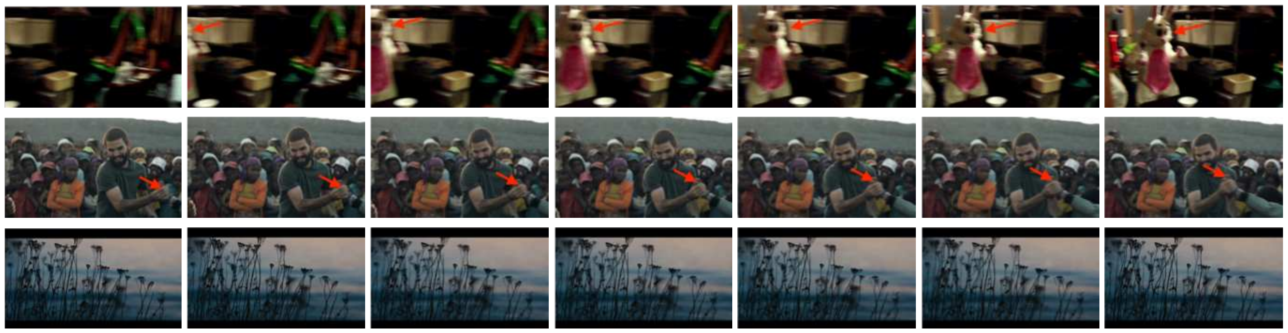


Figure 8. Examples from Vimeo-90k [34]. Top row: fast camera motion; new object appears in the third frame. Middle row: medium motion, little camer movement but some scene movement (e.g., person's arm in the foreground). Bottom row: slow motion only.

Clip Name	Flow Magnitude	Bicubic	DBPN [8]	BRCN [13]	VESPCN [2]	$B_{123}+T$ [25]	DRDVSR [30]	FRVSR [27]	VSR-DUF [16]	RBPN/6-PF
Calendar	1.14	19.82/0.554	22.19/0.714	-	-	21.66/0.704	22.18/0.746	-	(24.09/0.813*)	23.99/0.807 (23.93/0.803*)
City	1.63	24.93/0.586	26.01/0.684	-	-	26.45/0.720	26.98/0.755	-	(28.26/0.833*)	27.73/0.803 (27.64/0.802*)
Foliage	1.48	23.42/0.575	24.67/0.662	-	-	24.98/0.698	25.42/0.720	-	(26.38/0.771*)	26.22/0.757 (26.27/0.757*)
Walk	1.44	26.03/0.802	28.61/0.870	-	-	28.26/0.859	28.92/0.875	-	(30.50/0.912*)	30.70/0.909 (30.65/0.911*)
Average	1.42	23.53/0.629	25.37/0.737	24.43/0.662	25.35/0.756	25.34/0.745	25.88/0.774	26.69/0.822	(27.31/0.832*)	27.12/0.818 (27.16/0.819*)

Table 7. Quantitative evaluation of state-of-the-art SR algorithms on Vid4 for $4\times$. Red indicates the best and blue indicates the second best performance (PSNR/SSIM). The calculation is computed without crop any pixels border and remove first and last two frames. For $B_{123}+T$ and DRDVSR, we use results provided by the authors on their webpage. For BRCN, VESPCN, and FRVSR, the values taken from their publications. *The output is cropped 8-pixels near image boundary.

Clip Name	Flow Magnitude	Bicubic	DBPN [8] (1 Frame)	DRDVSR [30] (7 Frames)	VSR-DUF [16] (7 Frames)	RBPN/4-P (5 Frames)	RBPN/6-P (7 Frames)	RBPN/6-PF (7 Frames)
car05_001	6.21	27.62	29.58	32.07	30.77	31.51	31.65	31.92
hdclub_003_001	0.70	19.38	20.22	21.03	22.07	21.62	21.91	21.88
hitachi_isee5_001	3.01	19.59	23.47	23.83	25.73	25.80	26.14	26.40
hk004_001	0.49	28.46	31.59	32.14	32.96	32.99	33.25	33.31
HKVTG_004	0.11	27.37	28.67	28.71	29.15	29.28	29.39	29.43
jvc_009_001	1.24	25.31	27.89	28.15	29.26	29.81	30.17	30.26
NYVTG_006	0.10	28.46	30.13	31.46	32.29	32.83	33.09	33.25
PRVTG_012	0.12	25.54	26.36	26.95	27.47	27.33	27.52	27.60
RMVTG_011	0.18	24.00	25.77	26.49	27.63	27.33	27.64	27.69
veni3_011	0.36	29.32	34.54	34.66	34.51	36.28	36.14	36.53
veni5_015	0.36	27.30	30.89	31.51	31.75	32.45	32.66	32.82
Average	1.17	25.67/0.726	28.10/0.820	28.82/0.841	29.42/0.867	29.75/0.866	29.96/0.873	30.10/0.874

Table 8. Quantitative evaluation of state-of-the-art SR algorithms on SPMCS-11 for $4\times$. Red indicates the best and blue indicates the second best performance (PSNR/SSIM).

Algorithm	Vimeo-90k Slow	Medium	Fast
Bicubic	29.33/0.829	31.28/0.867	34.05/0.902
DBPN [8]	32.98/0.901	35.39/0.925	37.46/0.944
TOFLOW [34]	32.16/0.889	35.02/0.925	37.64/0.942
VSR-DUF/6 [16]	32.96/0.909	35.84/0.943	37.49/0.949
RBPN/3-P	33.73/0.914	36.66/0.941	39.49/0.955
RBPN/6-PF	34.18/0.920	37.28/0.947	40.03/0.960
# of clips	1,616	4,983	1,225
Avg. Flow Mag.	0.6	2.5	8.3

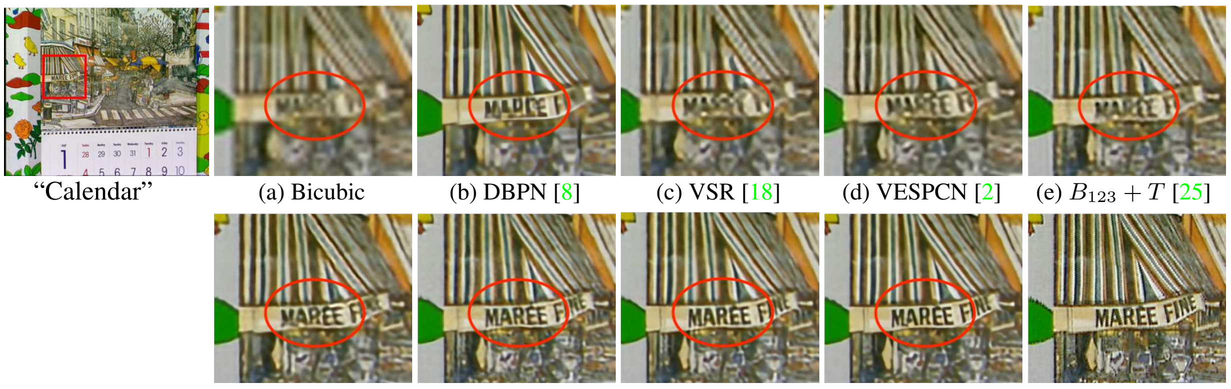
Table 9. Quantitative evaluation of state-of-the-art SR algorithms on Vimeo-90k [34] for $4\times$.

on fast motion. Even with reduced amount of temporal context available, RBPN/3-P (using only 3 extra frames) does better than previous methods like VSR-DUF using the full 6-extra frame context. RBPN/3-P get higher PSNR by 0.77 dB, 0.82 dB, and 2 dB than VSR-DUF on slow, medium, and fast motion, respectively.

Figure 11 shows qualitative results on Vimeo-90k. RBPN/6-PF obtains reconstruction that appears most similar to the GT, more pleasing and sharper than reconstructions with other methods. We have highlighted regions in which this is particularly notable.

5. Conclusion

We have proposed a novel approach to video super-resolution (VSR) called Recurrent Back-Projection Network (RBPN). It's a modular architecture, in which temporal and spatial information is collected from video frames surrounding the target frame, combining ideas from single- and multiple-frame super resolution. Temporal context is organized by a recurrent process using the idea of (back)projection, yielding gradual refinement of the high-


"Calendar" (a) Bicubic (b) DBPN [8] (c) VSR [18] (d) VESPCN [2] (e) $B_{123} + T$ [25]

(f) DRDVSR [30] (g) FRVSR [27] (h) VSR-DUF [16] (i) RBPN/6-PF (j) GT

Figure 9. Visual results on Vid4 for $4\times$ scaling factor. Zoom in to see better visualization.

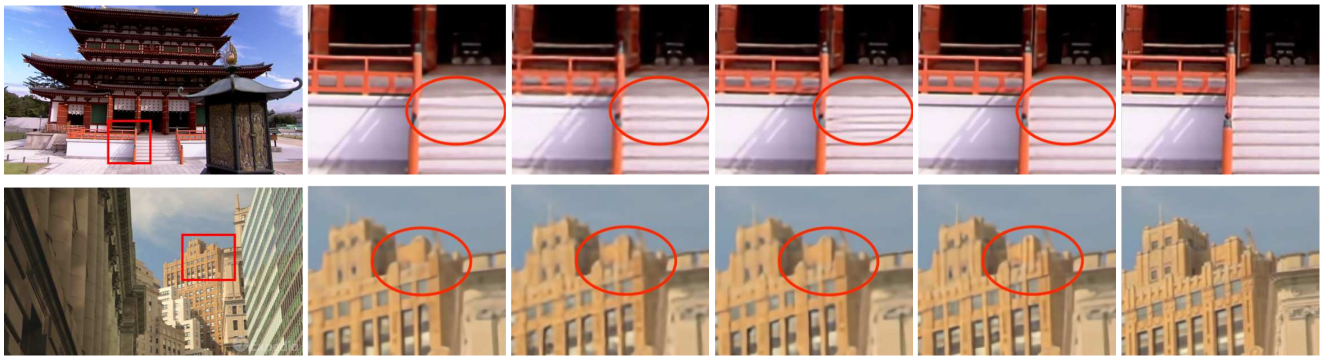

(a) DBPN [8] (b) DRDVSR [30] (c) VSR-DUF [16] (d) RBPN/6-PF (e) GT

Figure 10. Visual results on SPMCS for $4\times$ scaling factor. Zoom in to see better visualization.

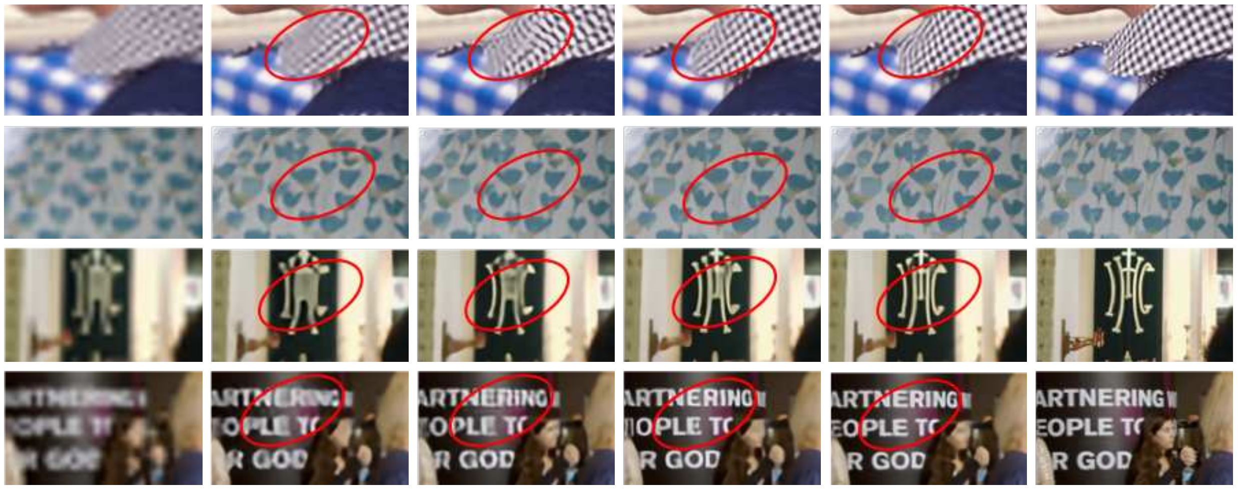

(a) Bicubic (b) TOFlow [34] (c) VSR-DUF [16] (d) RBPN/3-P (e) RBPN/6-PF (f) GT

Figure 11. Visual results on Vimeo-90k for $4\times$ scaling factor. Zoom in to see better visualization.

resolution features used, eventually, to reconstruct the high-resolution target frame. In addition to our technical innovations, we propose a new evaluation protocol for video SR. This protocol allows to differentiate performance of video SR based on magnitude of motion in the input videos. In extensive experiments, we assess the role played by various design choices in the ultimate performance of our approach, and demonstrate that, on a vast majority of thousands of test video sequences, RBPN obtains significantly better performance than existing VSR methods.

References

[1] Yochai Blau, Roey Mechrez, Radu Timofte, Tomer Michaeli, and Lihi Zelnik-Manor. 2018 pirm challenge on perceptual image super-resolution. *arXiv preprint arXiv:1809.07517*, 2018. 2

[2] Jose Caballero, Christian Ledig, Andrew P Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-time video super-resolution with spatio-temporal networks and motion compensation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1, 2, 4, 6, 7, 8

[3] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015. 2

[4] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2016. 1, 2

[5] Esmaeil Faramarzi, Dinesh Rajan, and Marc P Christensen. Unified blind method for multi-image super-resolution and single/multi-image blur deconvolution. *IEEE Transactions on Image Processing*, 22(6):2101–2114, 2013. 1

[6] Diogo C Garcia, Camilo Dorea, and Ricardo L de Queiroz. Super resolution for multiview images using depth information. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(9):1249–1256, 2012. 1

[7] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471, 2000. 1

[8] Muhammad Haris, Greg Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2, 3, 4, 6, 7, 8

[9] Muhammad Haris, M. Rahmat Widyanto, and Hajime Nobuhara. Inception learning super-resolution. *Appl. Opt.*, 56(22):6043–6048, Aug 2017. 1

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 4

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015. 4

[12] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *Pattern recognition (icpr), 2010 20th international conference on*, pages 2366–2369. IEEE, 2010. 6

[13] Yan Huang, Wei Wang, and Liang Wang. Bidirectional recurrent convolutional networks for multi-frame super-resolution. In *Advances in Neural Information Processing Systems*, pages 235–243, 2015. 1, 2, 6, 7

[14] Michal Irani and Shmuel Peleg. Improving resolution by image registration. *CVGIP: Graphical models and image processing*, 53(3):231–239, 1991. 1

[15] Michal Irani and Shmuel Peleg. Motion analysis for image enhancement: Resolution, occlusion, and transparency. *Journal of Visual Communication and Image Representation*, 4(4):324–335, 1993. 1

[16] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. 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*, pages 3224–3232, 2018. 1, 2, 6, 7, 8

[17] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4565–4574, 2016. 2

[18] Armin Kappeler, Seunghwan Yoo, Qiqin Dai, and Aggelos K Katsaggelos. Video super-resolution with convolutional neural networks. *IEEE Transactions on Computational Imaging*, 2(2):109–122, 2016. 2, 8

[19] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654, June 2016. 1, 2

[20] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1637–1645, 2016. 2

[21] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *IEEE Conferene on Computer Vision and Pattern Recognition*, 2017. 1, 2

[22] Renjie Liao, Xin Tao, Ruiyu Li, Ziyang Ma, and Jiaya Jia. Video super-resolution via deep draft-ensemble learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 531–539, 2015. 1, 2

[23] Ce Liu et al. *Beyond pixels: exploring new representations and applications for motion analysis*. PhD thesis, Massachusetts Institute of Technology, 2009. 6

[24] Ce Liu and Deqing Sun. A bayesian approach to adaptive video super resolution. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 209–216. IEEE, 2011. 2, 6

[25] Ding Liu, Zhaowen Wang, Yuchen Fan, Xianming Liu, Zhangyang Wang, Shiyu Chang, and Thomas Huang. Robust video super-resolution with learned temporal dynamics. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2526–2534. IEEE, 2017. 1, 2, 6, 7, 8

[26] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). In *ICLR*, 2015. 2

[27] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018. 1, 2, 3, 4, 6, 7, 8

[28] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution

using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016. 2

[29] Ying Tai, Jian Yang, and Xiaoming Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1, 2

[30] Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. Detail-revealing deep video super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy*, pages 22–29, 2017. 1, 2, 3, 4, 6, 7, 8

[31] Radu Timofte, Shuhang Gu, Jiqing Wu, Luc Van Gool, Lei Zhang, Ming-Hsuan Yang, Muhammad Haris, Greg Shakhnarovich, Norimichi Ukita, et al. Ntire 2018 challenge on single image super-resolution: Methods and results. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2018 IEEE Conference on*, 2018. 2

[32] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, 2014. 2

[33] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015. 3, 5

[34] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *arXiv preprint arXiv:1711.09078*, 2017. 2, 4, 6, 7, 8

[35] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4584–4593, 2016. 2