Abstract

Most recent video super-resolution (SR) methods either adopt an iterative manner to deal with low-resolution (LR) frames from a temporally sliding window, or leverage the previously estimated SR output to help reconstruct the current frame recurrently. A few studies try to combine these two structures to form a hybrid framework but have failed to give full play to it. In this paper, we propose an omniscient framework to not only utilize the preceding SR output, but also leverage the SR outputs from the present and future. The omniscient framework is more generic because the iterative, recurrent and hybrid frameworks can be regarded as its special cases. The proposed omniscient framework enables a generator to behave better than its counterparts under other frameworks. Abundant experiments on public datasets show that our method is superior to the state-of-the-art methods in objective metrics, subjective visual effects and complexity. Our code will be made public.

1. Introduction

Super-Resolution (SR) aims at reconstructing high-resolution (HR) images from the corresponding low-resolution (LR) images. As the most basic problem in SR, single image super-resolution (SISR) has been relatively studied thoroughly, where under a unified framework, the researchers only have to design different kinds of convolutional neural networks (CNNs) [3, 13, 31, 30, 2] to solve this issue. Based on SISR, video super-resolution (VSR) has also been developed, albeit a lot of works [1, 21, 18, 26, 28, 10, 9] have been proposed, there is not a unified framework being dominant in VSR yet. Figure 1 illustrates dozens of state-of-the-art (SOTA) VSR methods in terms of performance and speed.

As SISR requires only one input image, most SISR methods focus on exploring different generator networks to extract features from this one image under a unified framework. Nevertheless, since VSR involves consecutive video frames as input, different schemes for handling the temporal information have emerged. We demonstrate different kinds of frameworks for VSR in Figure 2.

As illustrated in Figure 2(a), most recent VSR methods [1, 21, 24, 29, 11, 23, 28, 22, 10] apply an iterative manner to deal with LR frames from a temporally sliding window, where we only show the case of window size as 3. Given a sequence of video frames, the iterative framework considers the whole VSR processing as multiple independent sub-processes. Theoretically, these sub-processes are not temporally correlated and can be handled simultaneously, which means they enjoy the advantage of parallel computing [28]. However, the iterative framework can only obtain more neighboring LR frames by increasing the window size but omits the previously estimated SR output, which is the exact reason that prevents it from a better performance.

As demonstrated in Figure 2(b), the recurrent framework [18, 9] processes video frames by the order, while it can never exploit the subsequent frames to assist with recov-
Figure 2: Different kinds of frameworks for VSR, where “G” represents the generator network. Red, black and blue arrows denote information from the past, present and future respectively.

Overall, in this paper, we propose a more generic omniscient framework to exploit both LR frames and estimated hidden states from the past, present and future. In fact, as shown in Figure 2, the iterative, recurrent and hybrid frameworks can be regarded as the special counterparts of our proposed omniscient framework. We have explored a same kind of generator network under different frameworks, and we have found the omniscient framework (local and global) superior to the existing iterative, recurrent and hybrid frameworks. Our models surpass other SOTA methods in both performance and complexity, and thus we hope this framework will become a standard framework in VSR, under which researchers are free to design more effective generator networks, explicit or implicit motion information capturing modules, or loss functions to tap its potential.

2. Related Work

2.1. Iterative Video Super-Resolution

With the development of deep learning, a lot of CNN-based SISR methods, e.g. SRCNN [3], VDSR [13], have emerged, from which some early VSR methods [1, 12, 16] are inspired and simply apply the architectures from SISR to VSR. These methods consider VSR as a multi-input counterpart of SISR and employ the iterative framework to solve it. These iterative methods can be further divided into two
categories according to their generator networks: iterative [17, 11, 23, 28, 22, 10, 27] and recurrent [21, 24, 29, 6]. Particularly, RVSR-LTD [17] designs a temporal adaptive network with a pyramid structure, and DUFVSR [11] proposes a 3D CNN with dynamic upsampling filters. EDVR [23] has designed a pyramid, cascading and deformable (PCD) alignment module with a temporal and spatial attention (TSA) fusion module to achieve wonderful results, however, its training resource requirement is prohibitive. PFNL [28] proposes a progressive fusion network with a non-local correlations extraction module, which has obtained an amazing result in terms of both performance and complexity. TDAN [22] devises a temporally-deformable alignment network to avoid the optical flow estimation. TGA [10] divides temporal information into groups, and utilizes a hybrid module with both 2D and 3D residual blocks for inter-group fusion.

Another group of iterative-based methods adopts recurrent generator networks to transfer temporal correlations between frames instead. DRVSR [21] utilizes ConvLSTM [20] module for capturing long-range temporal information, based on which MMCNN [24] designs a multi-memory residual block to enhance the memory ability. MTUDM [29] proposes an ultra-dense memory residual block combined with a multi-temporal information fusion strategy to build a shallower but wider network. RBPN [6] extends [5] to video SR, which sends LR frames into a projection module step by step according to their temporal distances. In all, these iterative methods have been concentrating on designing more and more complicated generator networks but with huge computational costs in the meantime.

2.2. Recurrent and Hybrid Video Super-Resolution

As to the recurrent methods, FRVSR [18] proposes a frame-recurrent network to leverage the last rebuilt SR frame for reconstructing the current frame, which is fast but not robust enough. RSDN [9] designs a dual-channel network to learn the structures and details of frames. Still, these recurrent methods refuse the assist from subsequent LR frames.

FFCVSR [26] and RLSP [4] have tried to form a hybrid framework but could not achieve a wonderful result because 1) the hybrid framework still cannot exploit the estimated hidden states from the present and future, and 2) their naive network designs have turned them against maximizing their potential.

3. Method

3.1. Omniscient Video Super-Resolution

We first introduce the core idea of omniscient video super-resolution (OVSRS). As illustrated in Figure 2, the hybrid VSR combines the idea of iterative VSR and recurrent VSR, which adopts both neighboring LR frames and a previously estimated SR output as source information. Theoretically, the neighboring LR frames provide the most basic spatial-temporal information in LR space, and the estimated SR output can reserve more temporally correlated information connected to the HR space. Thus, it is natural to combine them to fully exploit the spatial-temporal correlations.

Based on the above analysis, we wonder what if we further introduce the estimated SR outputs from the present and future. However, after long deliberation, we reckon that idea is not realizable if only processing through the video frames in a single time like the iterative, recurrent, or hybrid VSR. Eventually, as shown in Figure 2(d) and Figure 2(e), we manage to design two sub-networks: a precursor network Netp and a successor network Nets to form the OVSR framework. The precursor network first flows through the LR frames to generate the SR frames and hidden states of all time steps, and then the successor network reconstructs all SR frames with the assist of the corresponding LR frames and estimated hidden states. We further add the SR frames generated both by the precursor and successor for refinement to reconstruct the final SR output.

According to the direction of the precursor network and successor network, the OVSR can be further categorized as local omniscient VSR (LOVSR) and global omniscient VSR (GOVSR). Both Netp and Nets of LOVSR process video frames in the same direction, which means it can only utilize the past and present information, as well as a limited number of frames (2 in Figure 2(d)) in the future, which is why we call it “local omniscient”. Then, we design the GOVSR by reversing the direction of Netp subtly, through which any LR frame gets access to all other frames in a same video sequence, and thus it is called “global omniscient”. Although the GOVSR is capable of leveraging all frames to assist in rebuilding one frame at any time step, it relies heavily on future information. Nevertheless, frames too far in the future are actually not accessible in some delay-sensitive real-time tasks like live broadcasts and online meetings, where the LOVSR is more appropriate. In all, LOVSR suits online VSR and GOVSR is more appropriate for offline VSR. Still, the omniscient VSR, no matter LOVSR or GOVSR, can exploit both the LR frames and estimated hidden states from the past, present and future.

It is worth mentioning that the proposed omniscient framework does not require a specific structure of the generator network, instead, any kind of existing generator networks can be easily inserted into the architecture illustrated in Figure 3, as long as they satisfy the form shown in Equation (1) and Equation (2).

3.2. Network Design

We then elaborate on the specific network design of our model under the omniscient framework. Although our pro-
posed OVSR manages to leverage more source information theoretically, it still requires a well-designed structure of the generator network to fulfill its potential practically. Fortunately, we learn a progressive fusion residual block (PFRB) from PFNL [28], which is sophisticated and shows to be both effective and efficient.

To combine the PFRB with the omniscient framework, we ameliorate it to embody 3 channels. As illustrated in Figure 3, we show the architecture of our model under GOVSR, and the LOVSR case can be inferred. Because the Figure 3, we show the architecture of our model under the global omniscient framework, and the local omniscient case can be inferred according to Figure 2(d). The “⊕” represents element-wise add and “©” denotes concatenation along the channel axis. Red, black and blue arrows denote information from the past, present and future respectively.

The precursor network can be described as

\[ I_t^{SR_p}, H_t^p = Net_p(\{I_{t-1}^{LR}, I_t^{LR}, I_{t+1}^{LR}\}, H_{t+1}^p) \]  

(1)

where \( \{I_{t-1}^{LR}, I_t^{LR}, I_{t+1}^{LR}\} \) stand for the neighboring LR frames, \( I_t^{SR_p} \) and \( H_t^p \) denote the SR frame and hidden state generated by the precursor. Note that \( H_{t+1}^p \) is for GOVSR, which should be \( H_{t-1}^p \) for LOVSR instead. The successor network can be described as

\[ I_t^{SR_s}, H_t^s = Net_s(\{I_{t-1}, I_t^{LR}, I_{t+1}^{LR}\}, \{H_{t-1}^s, H_t^p, H_{t+1}^p\}) \]  

(2)

where \( I_t^{SR_s} \) and \( H_t^s \) denote the SR frame and hidden state generated by the successor. We add the SR frames generated by \( Net_p \) and \( Net_s \) to form the final output.

\[ I_t^{SR} = I_t^{SR_p} + I_t^{SR_s} \]  

(3)

because \( Net_s \) inherits \( Net_p \), we naturally restrict \( Net_s \) to mainly learn the low-frequency structures and \( Net_s \) to study the high-frequency details within frames. We adopt the Charbonnier loss function [15] to form the loss function:

\[ L = \sqrt{(I_t^{HR} - I_t^{SR})^2 + \varepsilon^2} + \alpha \sqrt{(I_t^{HR} - I_t^{SR_p})^2 + \varepsilon^2} \]  

(4)

where \( I_t^{HR} \) denotes the original HR frame, \( \alpha \) is set to adjust the weight of the precursor network, and \( \varepsilon \) is empirically set to \( 10^{-3} \).

In all, our model fully exploits the spatio-temporal correlations contained in the LR frames and estimated hidden states from the past, present and future.
4. Experiments

4.1. Implementation Details

We first adopt a public MM522 dataset [24] for training, which contains 522 32-frame sequences with various scenes. During training, we employ 20 video sequences from [28] for evaluation. When trained on the MM522 dataset, we test the models on Vid4 [1] and UDM10 [29] testing datasets. We also conduct experiments on another public training dataset Vimeo-90K [25], and test the models on its testing dataset Vimeo-90K-T (please see the appendix). Like in [11, 23, 28, 9], we adopt Gaussian blur with $\sigma = 1.6$ and then $4 \times$ down-sampling scheme.

We use Adam [14] optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We set the batch size as 16 and input LR size as $64 \times 64$. We only feed 7 consecutive frames for the ablation study, and to train our final models as well as rebuild other SOTA methods, we set 2 additional frames for feeding the $I_{t-1}^{LR}$, $H_{t-1}$ in the beginning and $I_{t+1}^{LR}$, $H_{t+1}$ at the end [4]. The initial learning rate is $1 \times 10^{-3}$ and decays linearly to $1 \times 10^{-4}$ after 120 K iterations, which keeps the same until 200 K iterations. Then, the learning rate is further decayed to $5 \times 10^{-5}$ and $1 \times 10^{-5}$ until converged. The whole training process lasts about 300 K - 400 K iterations. We conduct experiments on Python 3.6, PyTorch 1.6 and NVIDIA RTX 2080Ti GPUs.

Table 1: Performances, parameters, FLOPs and testing time costs of different models. The PSNR is evaluated on 20 sequences from [28], while FLOPs and testing time costs are evaluated with respect to 1280 $\times$ 720 HR frame under 4$\times$ SR.

| Model | IYSR | RVSR | HVSR | LOYSR | GOYSR |
|-------|------|------|------|-------|-------|
| Parameter (M) | 1.864 | 1.866 | 1.868 | 1.897 | 1.897 |
| FLOPs (G) | 107.780 | 107.696 | 107.796 | 109.746 | 109.746 |
| Testing time (ms) | 23.06 | 23.33 | 23.29 | 25.35 | 25.35 |
| PSNR (dB) | 30.66 | 30.60 | 31.10 | 31.24 | 31.27 |

4.2. Omniscient vs Iterative, Recurrent and Hybrid

We compare the same generator network under the iterative, recurrent, hybrid and omniscient frameworks (denoted as IVSR, RVSR, HVSR and OVSR), where only necessary adjustments are made to satisfy the form of input/output of these frameworks. Note that IVSR, RVSR and HVSR only embody one generator network, and thus we have to add the Bicubic amplified frame $I_{t}^{Bic}$ for residual learning [7, 13]. For LOYSR and GOYSR, we also replace $I_{t}^{SR_p}$ with $I_{t}^{Bic}$ in Equation (3), and we set $\alpha = 0$ in Equation (4) for a fair comparison. We integrate the multi-channel PFRBs into these frameworks, where we set 5 PFRBs with the filter number as 64 for IVSR, RVSR and HVSR.

As shown in Figure 4(a) and Table 1, HVSR outperforms IVSR and RVSR a lot in PSNR, which confirms the effec-
Table 2: PSNR (dB) evaluated by adjusting the weight of $I_{t_{SR}}$ through changing $\alpha$ in Equation (4).

| Setting | +Bicubic | $\alpha = 0$ | $\alpha = 0.01$ | $\alpha = 0.1$ | $\alpha = 1$ |
|---------|----------|---------------|-----------------|----------------|---------------|
| LOVSR   | 31.24    | 31.25         | 31.23           | 31.26          | 31.17         |
| GOVSR   | 31.27    | 31.28         | 31.36           | 31.32          | 31.30         |

Table 3: PSNR (dB) evaluated by adjusting the residual blocks in the precursor and successor networks. “4+2” denotes setting 4 residual blocks in $Net_p$ and 2 in $Net_s$.

| Setting | 0+6 | 1+5 | 2+4 | 3+3 | 4+2 | 5+1 | 6+0 |
|---------|-----|-----|-----|-----|-----|-----|-----|
| LOVSR   | 31.12 | 31.26 | 31.20 | 31.23 | 31.28 | 31.26 | 31.24 |
| GOVSR   | 31.20 | 31.36 | 31.40 | 31.46 | 31.52 | 31.47 | 31.43 |

4.3. Reconstruction Refinement

In Section 4.2, because IVSR, RVSR and HVSR do not own a precursor network, we adopt the Bicubic amplified results to replace the SR frames given by the precursor for OVSR. However, the Bicubic amplification is not optimal for our omniscient framework. Since OVSR consists of a precursor network and a successor network, we naturally consider making the precursor and successor learn the structures and details of frames respectively, and then add them for refinement. Thus, we change $\alpha$ in Equation (4) during training to explore the best option.

Training curves are shown in Figure 4(b) and Figure 4(c), while the specific numbers are demonstrated Table 2. For LOVSR, we find $\alpha = 0.1$ to be the optimal option, which surpasses the “+Bicubic” 0.02 dB. For GOVSR, setting $\alpha = 0.01$ achieves 31.36 dB in PSNR, which surpasses the “+Bicubic” 0.09 dB. We reckon that $Net_p$ and $Net_s$ in GOVSR cooperate better due to the global information utilization.

It is worth mentioning that in some case, especially when imposing no restrictions on the precursor ($\alpha = 0$), as illustrated in Figure 4(b) and Figure 4(c), models could suffer sudden drops during training, but they will still converge eventually.

4.4. Proportion of Precursor and Successor

In Section 4.2 and Section 4.3, we only set 1 PFRB in the precursor and 5 PFRBs in the successor for OVSR. Nevertheless, the optimal proportion of $Net_p$ and $Net_s$ still needs to be explored. Thus, we keep the total number of residual blocks fixed (6), and adjust the residual blocks in $Net_p$ (from 0 to 6) and $Net_s$ (from 6 to 0) to find an appropriate proportion. Training curves are shown in Figure 4(d) and Figure 4(e), while the specific numbers are demonstrated Table 3. Obviously, neither “0+6” or “6+0” is the best or second-best option, which verifies the necessity of designing both the precursor and successor networks. Both LOVSR and GOVSR achieve their best performances with 4 residual blocks in the precursor and 2 in the successor, but GOVSR achieves 31.52 dB under the optimal setting, which again proves the advantage of utilizing global information. Note that “4:2” is the optimal proportion under 6 residual blocks in total, however, there could be a better proportion if with more residual blocks.

After optimization, our models LOVSR and GOVSR achieve 31.28 dB and 31.52 dB in PSNR, which surpass the HVSR 0.18 dB and 0.42 dB respectively. With similar parameters and calculation costs, our OVSR outperforms the existing IVSR, RVSR and HVSR enough to prove the effectiveness of further involving estimated hidden states from the present and future.

4.5. Comparisons with SOTA Methods

Because most methods train their models on different training datasets with different down-sampling kernels, it is unfair to compare these methods only according to their papers. Thus, we have reimplemented dozens of SOTA VSR methods on the same training dataset with the same down-sampling kernels. Since PFNL [28] has already rebuilt DRVSR [21], FRVSR [18] and DUF_52L [11] under the same training conditions, we adopt their public codes. Based on the public code of PFNL, we reimplement MMD-CNN [24] and MTUDM [29] on the TensorFlow platform. Moreover, We rebuild RBPN [6], EDVR [23], FFCVSR [26], TDAN [22], RSDN [9] and RLSP [4] on the PyTorch platform. We could not retrain TGA because it requires too much GPU memory, and we have to report the results in its paper.

We first reimplement these VSR methods on MM522 [24] training dataset and then test them on Vid4 [1] and UDM10 [29] testing dataset. The PSNR and SSIM values are calculated only on the luminance channel of YCbCr colorspace, skipping the first and last two frames and elimi-
but they still outperform most other SOTA methods. 8+4-56 achieves the second-best performance. LOVSR-8+4-56 achieves the best performance, while GOVSR-8+4-56 achieves the second-best performance. LOVSR-8+4-56 (ours) 38.26/0.9718 43.57/0.9794 31.23/0.9376 35.40/0.9698 41.79/0.9804 49.40/0.9947 38.26/0.9765 32.20/0.8904 38.79/0.9765 40.04/0.9793 39.60/0.9698

Table 5: PSNR (dB) / SSIM of different video SR models on UDM10 testing dataset [29] by the upscaling factor of 4. Red and blue respectively indicate the best and second-best results. The * denotes the results reported in the original papers.

| Methods         | calendar    | city        | foliage      | walk        | average    | average*   |
|-----------------|-------------|-------------|--------------|-------------|------------|------------|
| DRVSR [21]      | 24.47 / 0.8387 | 28.75 / 0.8554 | 27.00 / 0.8013 | 30.85 / 0.9183 | 27.83 / 0.8534 | 27.92 / 0.8505 |
| LOVSR-4+2-56 (ours) | 24.71 / 0.8378 | 28.47 / 0.8502 | 26.94 / 0.7969 | 30.97 / 0.9204 | 27.78 / 0.8513 | 27.88 / 0.8513 |
| LOVSR-4+4-56 (ours) | 24.93 / 0.8439 | 29.08 / 0.8616 | 27.11 / 0.8073 | 31.23 / 0.9299 | 27.89 / 0.9092 | 28.09 / 0.8952 |
| LOVSR-8+4-80 (ours) | 25.10 / 0.8515 | 28.97 / 0.8666 | 27.25 / 0.8124 | 31.47 / 0.9273 | 28.20 / 0.8644 | 28.20 / 0.8644 |
| GOVSR+4-2-56 (ours) | 24.88 / 0.8463 | 28.74 / 0.8614 | 27.06 / 0.8075 | 31.27 / 0.9245 | 27.99 / 0.8599 | 27.99 / 0.8599 |
| GOVSR+4-4-56 (ours) | 25.16 / 0.8556 | 28.76 / 0.8683 | 27.36 / 0.8190 | 30.96 / 0.9181 | 27.59 / 0.8419 | 27.59 / 0.8419 |
| GOVSR+4-8-80 (ours) | 25.28 / 0.8581 | 29.10 / 0.8769 | 27.49 / 0.8230 | 31.79 / 0.9314 | 27.81 / 0.8724 | 27.81 / 0.8724 |

Because ViD4 dataset contains only 4 scenes at a low resolution, we further test these methods on a bigger testing dataset UDM10. As many methods have not conducted experiments on UDM10 dataset, we only report the results rebuilt by us in Table 5. Our heavy model GOVSR-8+4-80 achieves the best performance, while GOVSR-4+2-56 and LOVSR-8+4-80 achieve comparable performances. To make a comprehensive comparison, we show more details of these methods, e.g. type of framework and generator.
Table 6: Comprehensive comparisons of different video SR methods.

| Methods          | Framework | Generator | Frame | Parameter (M) | FLOPs (T) | Time (ms) / FPS | Time (ms) / FPS |
|------------------|-----------|-----------|-------|---------------|-----------|----------------|----------------|
| DRVSRC [21]      | iterative | recurrent | 3     | 1.722         | 0.415     | 123.6 / 8.09   | 276.2 / 3.62   |
| FRVSR [18]       | recurrent | recurrent | 5     | 10.582        | 3.347     | 696.7 / 1.44   | 1548.9 / 0.65  |
| MMCNN [24]       | iterative | recurrent | 5     | 5.919         | 1.672     | 506.9 / 1.97   | 1127.6 / 0.89  |
| MTUDM [29]       | iterative | iterative | 7     | 5.824         | 2.348     | 1108.0 / 0.90  | 2499.8 / 0.40  |
| DUF_52L [11]     | iterative | recurrent | 7     | 12.772        | 8.516     | 6555.7 / 0.15  | 14935.8 / 0.07 |
| RBPN [6]         | iterative | iterative | 7     | 20.699        | 2.954     | 436.7 / 2.29   | 979.2 / 1.02   |
| FFCVSR [26]      | hybrid    | hybrid    | 7     | 3.003         | 0.940     | 231.0 / 4.33   | 567.0 / 1.76   |
| RLSP7-256 [4]    | hybrid    | hybrid    | 3     | 5.581         | 0.322     | 50.8 / 19.69   | 112.7 / 8.87   |
| TDAN [22]        | iterative | iterative | 5     | 2.285         | 0.558     | 102.3 / 9.78   | 225.4 / 4.44   |
| TGA [10]         | iterative | iterative | 7     | 7.058         | 0.700     | 383.5 / 2.61   | 869.7 / 1.15   |
| RSDN9-128 [9]    | recurrent | recurrent | 2     | 6.180         | 0.356     | 59.3 / 16.86   | 132.0 / 7.58   |
| OVSR-4+2-56 (ours) | omniscient | omniscient | 3     | 1.897         | 0.110     | 25.4 / 39.37   | 56.1 / 17.83   |
| OVSR-8+4-56 (ours) | omniscient | omniscient | 3     | 3.480         | 0.201     | 46.5 / 21.51   | 101.9 / 9.81   |
| OVSR-8+4-80 (ours) | omniscient | omniscient | 3     | 7.062         | 0.407     | 81.2 / 12.32   | 178.5 / 5.60   |

Figure 5: Results of different methods on calendar from Vid4 [1] dataset.

In this paper, we propose an omniscient framework for VSR, which can be divided into the local omniscient VSR and global omniscient VSR. LOVSR suits online VSR and GOVSR fits offline VSR, and they both leverage LR frames and estimated hidden states from the past, present and future. We have conducted abundant experiments to prove the robustness of the omniscient framework, which is superior to other frameworks under similar complexity. Combined with a sophisticated generator network, the omniscient framework makes it possible to conduct VSR in real-time but still with an amazing performance. Due to the overwhelming advantages of this framework, we hope it will become a standard framework in VSR, based on which the researchers are welcome to design more effective generators, motion information capturing modules, as well as loss functions to further unleash its potential.
References

[1] Jose Caballero, Christian Ledig, Andrew Peter Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-time video super-resolution with spatio-temporal networks and motion compensation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2848–2857, 2017.

[2] Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. Second-order attention network for single image super-resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11065–11074, June 2019.

[3] Chao Dong, Change Loy Chen, 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.

[4] Dario Fuoli, Shuhang Gu, and Radu Timofte. Efficient video super-resolution through recurrent latent space propagation. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 3476–3485, 2019.

[5] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1664–1673, 2018.

[6] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Recurrent back-projection network for video super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2016.

[9] Takashi Isobe, Xu Jia, Shuhang Gu, Songjiang Li, Shengjin Wang, and Qi Tian. Video super-resolution with recurrent structure-detail network. In *European Conference on Computer Vision (ECCV)*, pages 645–660, 2020.

[10] Takashi Isobe, Songjiang Li, Xu Jia, Shansin Yuan, Gregory Slabaugh, Chunjing Xu, Ya-Li Li, Shengjin Wang, and Qi Tian. Video super-resolution with temporal group attention. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8008–8017, June 2020.

[11] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3224–3232, 2018.

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

[13] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1646–1654, 2016.

[14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2014.

[15] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5835–5843, 2017.

[16] Dingyi Li and Zengfu Wang. Video super-resolution via motion compensation and deep residual learning. *IEEE Transactions on Computational Imaging*, 3(4):749–762, 2017.

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

[18] Mehdi S. M Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6626–6634, 2018.

[19] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, 2016.

[20] Xingjian Shi, Zhourong Chen, Hao Wang, Wang Chun Woo, Wang Chun Woo, and Wang Chun Woo. Convolutional Istm network: a machine learning approach for precipitation nowcasting. In *International Conference on Neural Information Processing Systems (NIPS)*, pages 802–810, 2015.

[21] Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. Detail-revealing deep video super-resolution. In *IEEE International Conference on Computer Vision (ICCV)*, pages 4482–4490, 2017.

[22] Yapeng Tian, Yulun Zhang, Yun Fu, and Chenliang Xu. Tdan: Temporally-deformable alignment network for video super-resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3360–3369, June 2020.

[23] Xintao Wang, Kelvin C.K. Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[24] Zhongyuan Wang, Peng Yi, Kui Jiang, Junjun Jiang, Zhen Han, Tao Lu, and Jiayi Ma. Multi-memory convolutional neural network for video super-resolution. *IEEE Transactions on Image Processing*, 28(5):2530–2544, 2019.

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

[26] Bo Yan, Chuming Lin, and Weimin Tan. Frame and feature-context video super-resolution. In *AAAI Conference on Artificial Intelligence*, pages 5597–5604, 2019.
[27] Peng Yi, Zhongyuan Wang, Kui Jiang, Junjun Jiang, Tao Lu, and Jiayi Ma. A progressive fusion generative adversarial network for realistic and consistent video super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[28] Peng Yi, Zhongyuan Wang, Kui Jiang, Junjun Jiang, and Jiayi Ma. Progressive fusion video super-resolution network via exploiting non-local spatio-temporal correlations. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3106–3115, 2019.

[29] Peng Yi, Zhongyuan Wang, Kui Jiang, Zhenfeng Shao, and Jiayi Ma. Multi-temporal ultra dense memory network for video super-resolution. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(8):2503–2516, 2020.

[30] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *The European Conference on Computer Vision (ECCV)*, pages 294–310, September 2018.

[31] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2472–2481, 2018.
Appendix

We conduct more ablation studies in Section A. We show the result on Vimeo-90K [25] in Section B and more visual comparisons in Section C.

A. Ablation Studies

A.1. HVSR with 4 LR frames

Our LOVSR actually utilizes 4 LR frames through the future hidden states $H_{t+1}$, and thus we have conducted an additional experiment of HVSR adopting 4 LR frames ($I_{t-1}, I_t, I_{t+1}, I_{t+2}$). As shown in Figure 6 and Table 7, HVSR$^3$ and HVSR$^4$ denote HVSR with 3 and 4 input LR frames respectively, where HVSR$^4$ surpasses HVSR$^3$ about 0.05 dB. Still, with the same 4 LR frames, our LOVSR outperforms HVSR$^4$ about 0.13 dB, and our GOVSR surpasses HVSR$^4$ about 0.37 dB with the assist of all LR frames in a video sequence. Note that these models have a similar number of parameters and calculation costs, which again proves the robustness of our OVSR framework.

![Figure 6: Training curves of different models.](image)

Table 7: PSNR (dB) of different models.

| Model      | HVSR$^3$ | HVSR$^4$ | LOVSR-4+2 | GOVSR-4+2 |
|------------|----------|----------|-----------|-----------|
| PSNR (dB)  | 31.10    | 31.15    | 31.28     | 31.52     |

A.2. Influences of Input Information

We further investigate the influence of each input information on the performance, as demonstrated in Table 8, we present the performances of different models by removing part of the input information.

Obviously, all models drop seriously without the center LR frame $I_t$, which accords with common sense because $I_t$ contains the most basic and significant source information. Interestingly, IVSR, RVSR, and HVSR models all achieve 26.61 dB without $I_t$, and this is almost the same as the Bicubic magnified center frame $I_{t}^{Bic}$ actually. Because generating the Bicubic magnified center frame $I_{t}^{Bic}$ does not require learnable parameters, the IVSR, RVSR, and HVSR models all drop to $I_{t}^{Bic}$ without $I_t$. Nevertheless, our OVSR models add the SR frames by $Net_s$ and $Net_f$ for reconstruction refinement (discussed in Equation (3) and Section 4.3 in the original paper), which requires the cooperation of two learnable networks and thus is more unstable than the Bicubic magnification, where they basically drop more than 10 dB by removing the center frame $I_t$.

IVSR drops nearly the same by removing $I_{t-1}$ or $I_{t+1}$, which means $I_{t-1}$ and $I_{t+1}$ contribute almost equally to the result, and thus it is unwise for RVSR to overlook the subsequent frame $I_{t+1}$.

Excluding $I_{t-1}$ or $H_{t-1}$, RVSR drops a lot, which proves that the hidden states can provide some beneficial information.

Moreover, HVSR drops more by removing $I_{t+1}$ compared to removing $I_{t-1}$, and we reckon that this is due to the assist from $H_{t-1}$. This phenomenon further confirms that the hidden states contribute to the VSR indeed, and thus it makes sense to further adopt the hidden states from the present and future to help VSR. Similar phenomena can also be observed in our models LOVSR and GOVSR.

Amazingly, by removing $I_{t-1}$ in $Net_s$, our GOVSR stays the same but LOVSR increases 0.01 dB in PSNR instead, and we owe it to the OVSR framework, which leverages hidden states from the past, present and future to help VSR, through which it basically does not need $I_{t-1}$ in $Net_s$ anymore. Last but not least, compared to removing the LR frames $I_t$, our models LOVSR and GOVSR deteriorate more seriously or comparably by removing the corresponding hidden states $H$ in $Net_s$, which again confirms that our models indeed make good use of the hidden states from the past, present and future.

B. Result on Vimeo-90K Dataset

We further conduct experiments on another public training dataset Vimeo-90K [25], and test the models on its testing dataset Vimeo-90K-T. According to the average motion flow magnitude, Vimeo-90K-T is divided into 3 categories: slow, medium, and fast [6], where there are 1616, 4983, and 1225 sequences in each category. Albeit Vimeo-90K contains tens of thousands of video sequences, each of which consists of only 7 frames, and the HR frames are at the fixed resolution $448 \times 256$, which is quite low. Besides, during testing, other methods only calculate the PSNR/SSIM of the center SR frame, and consequently, we do not think this dataset is suitable for VSR. Still, we retrain as many methods as we can on Vimeo-90K in a limited time for a more comprehensive comparison, under the same training settings. The PSNR and SSIM values are calculated only on the luminance channel of YCbCr colorspace, focusing on the center frame and eliminating 8 pixels on four bor-
Table 8: PSNR (dB) of different models, where ‘Full’ denotes the original model with all input information, and ‘w/o’ means without.

| Model      | IVSR | RVSR | HVSR | LOVSR-4+2 | GOVSR-4+2 |
|------------|------|------|------|-----------|-----------|
| Network    | G    | G    | G    | Both      | Both      |
| Full       | 30.66| 30.60| 31.10| 31.28     | 31.52     |
| w/o $I_{t-1}$ | 29.07 (-1.59) | 28.95 (-1.65) | 29.18 (-1.92) | 29.27 (-2.01) | 29.42 (-2.16) |
| w/o $I_t$   | 26.61 (-4.05) | 26.61 (-3.99) | 26.61 (-4.49) | 21.18 (-10.10) | 16.78 (-14.50) |
| w/o $I_{t+1}$ | 29.10 (-1.56) | - | 28.91 (-2.19) | 28.94 (-2.34) | 30.66 (-0.62) |
| w/o $H_{t-1}$ | - | 29.35 (-1.25) | 29.81 (-1.29) | 29.74 (-1.54) | 31.01 (-0.27) |
| w/o $H_t$   | - | - | - | 11.18 (-20.10) | - |
| w/o $H_{t+1}$ | - | - | - | 30.52 (-0.76) | - |

Table 9: PSNR (dB) / SSIM of different video SR methods on Vimeo-90K testing dataset [25] by the upscaling factor of 4. Red and blue respectively indicate the best and second-best results. The * denotes the results reported in the original papers.

| Methods      | Vimeo-Slow | Vimeo-Medium | Vimeo-Fast | Vimeo-All |
|--------------|------------|--------------|------------|-----------|
| RBPN* [6]    | 34.18 / 0.9200 | 37.28 / 0.9470 | 40.03 / 0.9600 | 37.07 / 0.9435 |
| EDVR* [23]   | -          | -            | -          | 37.61 / 0.9489 |
| FFCVSR [26]  | 33.59 / 0.9130 | 36.51 / 0.9416 | 38.68 / 0.9481 | 36.24 / 0.9367 |
| RLS7-256 [4] | 33.86 / 0.9173 | 36.97 / 0.9463 | 39.20 / 0.9535 | 36.67 / 0.9415 |
| RSDN9-128 [9]| 33.91 / 0.9179 | 37.03 / 0.9466 | 39.41 / 0.9555 | 36.76 / 0.9421 |
| LOVSR-8+4-80 (ours) | 34.51 / 0.9256 | 37.84 / 0.9538 | 40.23 / 0.9615 | 37.53 / 0.9492 |
| GOVSR-8+4-80 (ours) | 34.60 / 0.9270 | 37.95 / 0.9548 | 40.32 / 0.9624 | 37.63 / 0.9503 |

As shown in Table 9, our model GOVSR-8+4-80 still achieves the best performance. It is worth mentioning that compared to GOVSR-8+4-80, LOVSR-8+4-80 can not uti-
lize all 7 frames to rebuild the center frame, and thus it behaves a little worse.

C. Visual Comparisons

We show more visual comparisons, as illustrated in Figure 7 and Figure 8, most methods can only recover the low-frequency contour of the objects, which seems smooth and blurry. Our models can recover the right textures with more realistic details.