Self-Conditioned Probabilistic Learning of Video Rescaling

Yuan Tian[1] Guo Lu[2] Xiongkuo Min[1] Zhaohui Che[1] Guangtao Zhai[1*] Guodong Guo[3] Zhiyong Gao[1]
[1]Shanghai Jiao Tong Unversity [2]Beijing Institute of Technology [3]Baidu
{ee_tianyuan, minxiongkuo, chezhaohui, zhaiguangtao, zhiyong.gao}@sjtu.edu.cn
guo.lu@bit.edu.cn, guoguodong01@baidu.com

Abstract

Bicubic downscaling is a prevalent technique used to reduce the video storage burden or to accelerate the downstream processing speed. However, the inverse upscaling step is non-trivial, and the downscaled video may also deteriorate the performance of downstream tasks. In this paper, we propose a self-conditioned probabilistic framework for video rescaling to learn the paired downscaling and upscaling procedures simultaneously. During the training, we decrease the entropy of the information lost in the downscaling by maximizing its probability conditioned on the strong spatial-temporal prior information within the downscaled video. After optimization, the downscaled video by our framework preserves more meaningful information, which is beneficial for both the upscaling step and the downstream tasks, e.g., video action recognition task. We further extend the framework to a lossy video compression system, in which a gradient estimator for non-differential industrial lossy codecs is proposed for the end-to-end training of the whole system. Extensive experimental results demonstrate the superiority of our approach on video rescaling, video compression, and efficient action recognition tasks.

1. Introduction

High-resolution videos are widely used over various computer vision tasks [44][62][7][35][33][50][64]. However, considering the increased storage burden or the high computational cost, it is usually required to first downscale the high-resolution videos. Then we can either compress the output low-resolution videos for saving storage cost or feed them to the downstream tasks to reduce the computational cost. Despite that this paradigm is prevalent, it has the following two disadvantages. First, it is non-trivial to restore the original high-resolution videos from the (compressed) low-resolution videos, even we use the latest super-resolution methods [30, 63, 48, 57]. Second, it is also a challenge for the downstream tasks to achieve high performance based on these low-resolution videos. Therefore, it raises the question that whether the downscaling operation can facilitate the reconstruction of the high-resolution videos and also preserve the most meaningful information.

Recently, this question has been partially studied as a single image rescaling problem [24, 27, 47, 60], which learns the down/up scaling operators jointly. However, how to adapt these methods from image to video domain and leverage the rich temporal information within videos are still open problems. More importantly, modeling the lost information during downscaling is non-trivial. Current methods either ignore the lost information [24, 27, 47] or assume it as an independent distribution in the latent space [60], while neglecting the *internal relationship* between the downscaled image and the lost information. Besides, all literature above has not explored how to apply the rescaling technique to the lossy image/video compression.

In this paper, we focus on building a video rescaling framework and propose a self-conditioned probabilistic learning approach to learn a pair of video downscaling and upscaling operators by exploiting the information dependency within the video itself. Specifically, we first design a learnable frequency analyzer to decompose the original high-resolution video into its downscaled version and the corresponding high-frequency component. Then, a Gaussian mixture distribution is leveraged to model the high-frequency component by *conditioning* on the downscaled video. For accurate estimation of the distribution parameters, we further introduce the local and global temporal aggregation modules to fuse the spatial information from adjacent downscaled video frames. Finally, the original video can be restored by a frequency synthesizer from the downscaled video and the high-frequency component sampled from the distribution. We integrate the components above as a novel self-conditioned video rescaling framework termed **SelfC** and optimize it by minimizing the negative log-likelihood for the distribution.

Furthermore, we apply our proposed SelfC in two practical applications, *i.e.*, lossy video compression and video action recognition. In particular, to integrate our framework with the existing non-differential video codecs (*e.g.*,

*Corresponding author.

H.264 [59] and H.265 [46]), we propose an efficient and effective one-pass optimization strategy based on the *control variates* method and approximate the gradients of traditional codecs in the back-propagation procedure, which formulates an end-to-end optimization system.

Experimental results demonstrate that the proposed framework achieves state-of-the-art performance on the video rescaling task. More importantly, we further demonstrate the effectiveness of the framework in practical applications. For the lossy video compression task, compared with directly compressing the high-resolution videos, the video compression system based on our SelfC framework cuts the storage cost significantly (up to 50% reduction). For the video action recognition task, our framework reduces more than 60% computational complexity with negligible performance degradation. In summary, our main contributions are:

- We propose a probabilistic learning framework dubbed **SelfC** for the video rescaling task, which models the lost information during downscaling as a dynamic distribution conditioned on the downscaled video.
- Our approach exploits rich temporal information in downscaled videos for an accurate estimation of the distribution parameters by introducing the specified local and global temporal aggregation modules.
- We propose a gradient estimation method for non-differential lossy codecs based on the control variates method and Monte Carlo sampling technique, extending the framework to a video compression system.

2. Related Work

Video Upscaling after Downscaling. Previous SR works [30, 63, 48, 57, 17, 23] mainly leverage a heavy neural network to hallucinate the lost details during the downscaling, only achieving unsatisfactory results. Taking the video downscaling method into consideration may help mitigate the ill-posedness of the video upscaling procedure.

There are already a few works on single image rescaling task following this spirit. For example, Kim *et al*. [24] proposed a task-aware downscaling model based on an auto-encoder framework. Li *et al*. [27] proposed to use a neural network to estimate the downscaled low-resolution images for a given super-resolution method. Yang *et al*. [65] proposed a superpixel-based downsampling/upsampling scheme to effectively preserve object boundaries. Recently, Xiao *et al*. [60] proposed to leverage a deep invertible neural network (INN) to model the problem, which maps the complex distribution of the lost information to an *independent* and *fixed* normal distribution.

However, these methods neither leverage the temporal information between adjacent frames, which is important for video-related tasks, nor consider the fact that the components of different frequencies in natural images or videos are conditionally dependent [53, 45, 38, 54].

Video Compression. Several traditional video compression algorithms have been proposed and widely deployed, such as H.264 [59] and H.265 [46]. Most of them follow the predictive coding architecture and rely on the sophisticated hand-crafted transformations to analyze the redundancy within the videos. Recently, fully end-to-end video codecs [32] [15][6][66][2][29][31][19][34] have been proposed by optimizing the rate-distortion trade-off. They demonstrate promising performance and may be further improved by feeding more ubiquitous videos in the wild. However, they haven't been widely used by industrial due to computational inefficiency. In contrast, our framework can be readily integrated with the best traditional video codecs and further saves the video storage space significantly.

Video Action Recognition. Simonyan *et al*. [44] first proposed the two-stream framework. Feichtenhofer *et al*. [9] then improved it. Later, wang *et al*. [56] proposed a new sparse frame sampling strategy. Recently, 3D networks [51][5][16][52][39][8][49] also show promising performance. Our work can accelerate the off-the-shelf action CNNs by 3-4 times while reserving the comparable performance. We mainly conduct experiments on light-weight 2D action CNNs (*e.g.*, TSM [28]) for efficiency.

3. Proposed Method

An overview of our proposed SelfC framework is shown in Fig. 1 (a). During the downscaling procedure, given a high-resolution (HR) video, a frequency analyzer (FA) (Section 3.1) first converts it into video features f, where the first 3 channels are low-frequency (LF) component f_l, the last $3 \cdot k^2$ channels are high-frequency (HF) component f_h, and k is the downscaling ratio. Then, f_l is quantized to a LR video x_l for storage. f_h is discarded in this procedure.

During the upscaling procedure, given the LR video x_l, the spatial-temporal prior network (STP-Net) (Section 3.3) predicts the probability density function of the HF component f_h:

$$p(f_h|x_l) = \mathsf{STP\text{-}Net}(x_l). \tag{1}$$

We model $p(f_h|x_l)$ as a continuous mixture of the parametric Gaussian distributions (Section 3.2). Then, a case of the HF component $\hat{f}_h$ related to LR video x_l is drawn from the distribution. Finally, we reconstruct the HR video from the concatenation of HF component $\hat{f}_h$ and LR video x_l by the frequency synthesizer (FS).

3.1. Frequency Analyzer and Synthesizer

As shown in Fig. 1 (b), we first decompose the HR input video x as the k times downscaled low-frequency component $c_l := \mathsf{Down}(x) \in \mathbb{R}^{t\times 3\times \frac{h}{k}\times \frac{w}{k}}$ and the residual high-frequency component $c_h := \mathsf{PixelUnshuffle}(x - \mathsf{Up}(c_l)) \in \mathbb{R}^{t\times 3\cdot k^2\times \frac{h}{k}\times \frac{w}{k}}$, where $h \times w$ denotes the spatial scale of the

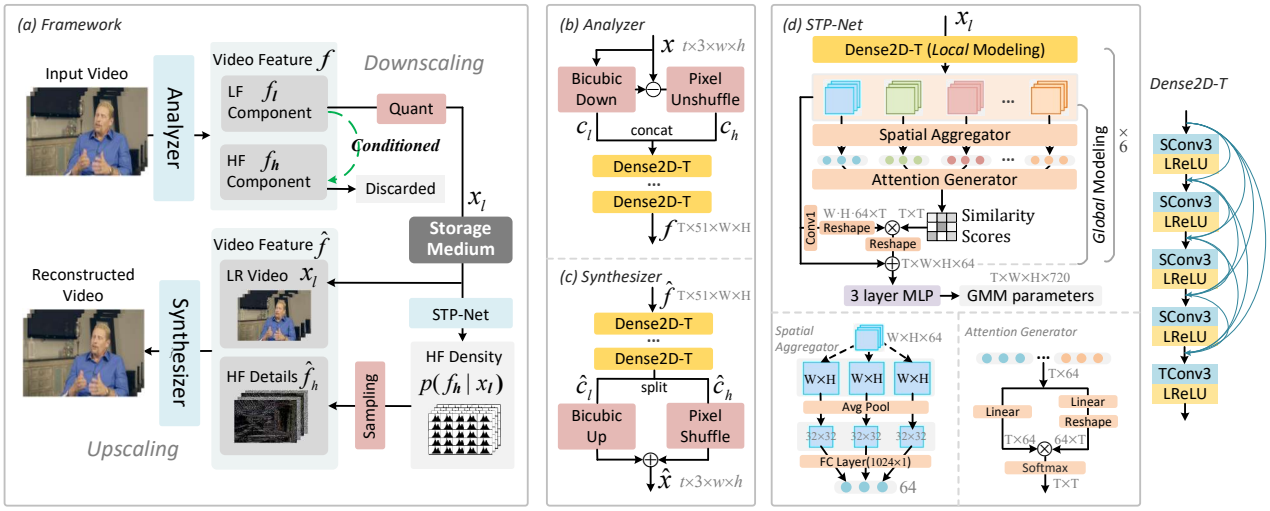

Figure 1: Overview of the proposed SelfC framework. We exploit the conditional relationship between different frequency components within the input video for better learning of video rescaling. During downscaling, the low-frequency (LF) component f_l is quantized to produce the low-resolution (LR) video x_l and then stored. The storage medium can be lossless or lossy for different applications. During upscaling, the probability density $p(f_h|x_l)$ of the high-frequency (HF) component f_h is predicted by a spatial-temporal prior network (STP-Net) from x_l. Then, x_l and the sampled HF details $\hat{f}_h$ are reconstructed to the high-resolution video by the frequency synthesizer. We also introduce some new symbols, W= w/k, H= h/k and T= t, for better indication of the tensor dimensions. Conv1/SConv3/TConv3 denote the 3D convolutions of kernel size $1 \times 1 \times 1/1 \times 3 \times 3/3 \times 1 \times 1$. LReLU denotes the Leaky-ReLU non-linearity [61].

original video and t denotes the video length. Down and Up represent bicubic downscaling and upscaling operations with the scaling ratio k. PixelUnshuffle is the inverse operation of the pixel shuffling operation proposed in [43], where the scaling ratio is also k. Then, we use a learnable operation $\mathcal{T}$ to transform c_l and c_h to the output features $f := \mathcal{T}(c_l © c_h)$, where © denotes the channel concatenation operation. The produced video feature f consists of LF component f_l and HF component f_h. The network architecture for $\mathcal{T}$ is very flexible in our framework and we use multiple stacking Dense2D-T blocks to implement it by default. The Dense2D-T block is modified from the vanilla Dense2D block [20] by replacing the last spatial convolution with the temporal convolution.

The architecture of the frequency synthesizer is symmetric with the analyzer, as shown in Fig. 1 (c). Specifically, we use channel splitting, bicubic upscaling, and pixel shuffling operations to synthesize the final high-resolution videos based on the reconstructed video feature $\hat{f}$.

3.2. A Self-conditioned Probabilistic Model

Directly optimizing $p(f_h|x_l)$ in Eq. (1) through gradient descent is unstable due to the unsmooth gradient [24] of the quantization module. Thus, we optimize $p(f_h|f_l)$ instead during the training procedure. Specifically, we represent the high-frequency component f_h as a continuous multi-modal probability distribution conditioned on the low-frequency component f_l, which is formulated as:

$$p(f_h|f_l) = \prod_o p(f_h(o)|f_l), \tag{2}$$

where o denotes the spatial-temporal location. We use a continuous Gaussian Mixture Model (GMM) [40] to approximate p with component number $K = 5$. The distributions are defined by the learnable mixture weights w_o^k, means μ_o^k and log variances σ_o^k. With these parameters, the distributions can be accurately determined as:

$$p(f_h(o)|f_l) = \sum_{k=1}^{K} w_o^k p_g(f_h(o)|\mu_o^k, e^{\sigma_o^k}), \tag{3}$$

where

$$p_g(f|\mu, \sigma^2) = \frac{1}{\sqrt{\pi}\sigma} e^{-\frac{(f-\mu)^2}{\sigma^2}}. \tag{4}$$

3.3. Spatial-temporal Prior Network (STP-Net)

As shown in Fig. 1 (d), to estimate the parameters of the distribution above, we propose the STP-Net to model both the local and global temporal information within the downscaled video. We first utilize the Dense2D-T block to extract the short-term spatial-temporal features for each input frame. In this stage, only information from local frames, *i.e.*, the previous or the next frames, are aggregated into the current frame, while the temporally long-range dependencies in videos are neglected. Therefore, we further introduce the attention mechanism for modeling the global temporal information. More specifically, the spatial dimension of the short-term spatial-temporal features is first reduced

by a spatial aggregator, which is implemented as an average pooling operation followed by a full-connected (FC) layer. The output scale of the pooling operation is 32×32. Then we use the dot-producting operation to generate the attention map, which represents the similarity scores between every two frames. Finally, we refine the local spatial-temporal features based on the similarity scores. We repeat the following procedure six times to extract better video features. After that, a three-layer multi-layer perceptron (MLP) is used to estimate the parameters of the GMM distribution.

3.4. Quantization and Storage Medium

We use rounding operation as the quantization module, and store the output LR videos by lossless format, *i.e.*, H.265 lossless mode. The gradient of the module is calculated by the Straight-Through Estimator [3]. We also discuss how to adapt the framework to more practical lossy video formats such as H.264 and H.265 in Section 3.6.

3.5. Training Strategy

Building a learned video rescaling framework is non-trivial, especially the generated low-resolution videos are expected to benefit both the upscaling procedure and the downstream tasks. We consider the following objectives.

Learning the self-conditioned probability. First, to make sure the STP-Net can obtain an accurate estimation for the HF component f_h, we directly minimize the negative log-likelihood of $p(f_h|f_l)$ in Eq. (2):

$$\mathcal{L}_c = -\sum_{i=0}^{N} log(p(f_h^i|f_l^i)), \tag{5}$$

where N is the number of the training samples.

Mimicking Bicubic downscaling. Then the downscaled video is preferred to be similar to the original video, making its deployment for the downstream tasks easier. Therefore, we regularize the downscaled video before quantization, *i.e.*, f_l, to mimic the bicubic downsampled x:

$$\mathcal{L}_{mimic} = ||x_{bicubic} - f_l||_2, x_{bicubic} = \mathsf{Bicubic}(x). \tag{6}$$

Penalizing $\mathcal{T}$. Without any extra constraint, Eq. (5) can be easily minimized by tuning f_h to one constant tensor for any input video. Thus, to avoid the trivial solution, the CNN parts of the frequency analyzer and synthesizer are penalized by the photo-parametric loss (*i.e.*, ℓ_2 loss) between the video directly reconstructed from f and the original input x:

$$\mathcal{L}_{pen} = ||x - \mathsf{FS}(f)||_2. \tag{7}$$

Minimizing reconstruction difference. Finally, the expected difference between the reconstructed video sampled from the model and the original video should be minimized:

$$\mathcal{L}_{recons} = \ell(x, \hat{x}), \hat{x} = \mathsf{FS}(x_l © \hat{f}_h), \tag{8}$$

where ℓ denotes a photo-metric loss (*i.e.*, ℓ_1 loss), © denotes the channel-wise concatenation operation. In each training iteration, $\hat{f}_h$ is sampled from the distribution constructed from the parameters output by STP-Net, conditioning on the LR video x_l. To enable an end-to-end optimization, we apply the "reparametrization trick" [26, 41, 14] to make the sampling procedure differentiable.

The total loss is then given by:

$$\mathcal{L}_{selfc} = \lambda_1\mathcal{L}_c + \lambda_2 \cdot k^2\mathcal{L}_{mimic} + \lambda_3\mathcal{L}_{pen} + \lambda_4\mathcal{L}_{recons}, \tag{9}$$

where λ_1, λ_2, λ_3 and λ_4 are the balancing parameters, and k is the scaling ratio. The loss function of our framework may seem a little bit complicated. However, we want to mention that the performance of our framework is not sensitive to these hyper-parameters, and directly setting all the parameters to 1 already achieves reasonable performance.

3.6. Application I: Video Compression

In this section, we extend the proposed SelfC framework to a lossy video compression system, which aims to demonstrate the effectiveness of our approach in reducing the video storage space. The whole system is shown in Fig. 2. Specifically, we first use the SelfC to generate the downscaled video x_l, which will be compressed by using the off-the-shelf industrial video codecs, *e.g.*, H.265 codec. Then at the decoder side, the compressed videos will be decompressed and upscaled to the full resolution video.

Considering the traditional video codecs are non-differential, we further propose a novel optimization strategy. Specifically, we introduce a differentiable surrogate video perturbator ϕ, which is implemented as a deep neural network (DNN) consisting of 6 Dense2D-T blocks. During the back-propagation stage, the gradient of the codec can be approximated by that of ϕ, which is tractable. During the test stage, the surrogate DNN is removed and we directly use the H.265 codec for compression and decompression.

According to the *control variates* theory [11, 13], ϕ can be a low-variance gradient estimator for the video codec (*i.e.*, η) when (1) the differences between the outputs of the two functions are minimized and (2) the correlation coefficients ρ of the two output distributions are maximized.

Therefore, we introduce these two constraints to the optimization procedure of the proposed SelfC based video compression system. The loss function for the surrogate video

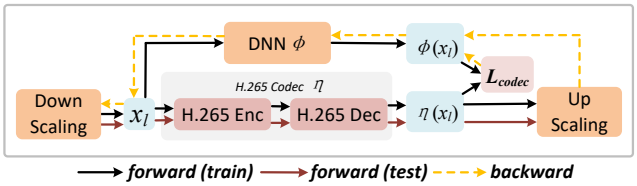

Figure 2: We introduce a surrogate DNN to calculate the gradient of the non-differential codec. We take the H.265 codec as an example.

perturbator is formulated as:

$$\mathcal{L}_{codec} = ||\eta(x_l) - \phi(x_l)||_2 - \lambda_\rho \rho(\eta, \phi), \quad (10)$$

where λ_ρ is set to a small value, *i.e.*, 10^{-5}, and ρ is estimated within each batch by Monte Carlo sampling:

$$\rho(\eta, \phi) = \frac{\sum_{k=1}^{N}(\eta(x_l^k) - \mathbb{E}[\eta])(\phi(x_l^k) - \mathbb{E}[\phi])}{\sqrt{\sum_{k=1}^{N}(\eta(x_l^k) - \mathbb{E}[\eta])^2}\sqrt{\sum_{k=1}^{N}(\phi(x_l^k) - \mathbb{E}[\phi])^2}}, \quad (11)$$

where

$$\mathbb{E}[\eta] = \frac{1}{N}\sum_{k=1}^{N}\eta(x_l^k),\ \mathbb{E}[\phi] = \frac{1}{N}\sum_{k=1}^{N}\phi(x_l^k), \quad (12)$$

and N denotes the batch size. Note that ρ is separately computed on each spatial-temporal location and then averaged.

Finally, the total loss function for the SelfC based video compression system is given by:

$$\mathcal{L}_{compression} = \mathcal{L}_{selfc} + \lambda_{codec}\mathcal{L}_{codec}, \quad (13)$$

where λ_{codec} is the balancing weight.

3.7. Application II: Efficient Action Recognition

We further apply the proposed SelfC framework to the video action recognition task. Specifically, we adopt the LR videos (*i.e.*, x_l) downscaled by our framework as the input of action recognition CNNs for efficient action recognition. Considering the downscaler of our approach can preserve meaningful information for the downstream tasks and the complexity of itself can be rather low, inserting the downscaler before the off-the-shelf action CNNs can reduce the huge computational complexity of them with negligible performance drop. Moreover, the light-weightiness of the rescaling framework makes the joint optimization tractable. In fact, compared with bicubic downscaling operation, our downscaler in SelfC framework can still generate more informative low-resolution videos for the action recognition task even without the joint training procedure. Please see Section 4.5 for more experimental results.

4. Experiments

4.1. Dataset

We use the videos in Vimeo90K dataset [63] as our training data, which are also adopted by the recent video super resolution methods [21, 22, 57] and learnable video compression methods [32, 18, 31]. *For video rescaling task*, the evaluation datasets are the test set of Vimeo90K (denoted by Vimeo90K-T), the widely-used Vid4 benchmark [30] and SPMCs-30 dataset [48]. *For video compression task*, the evaluation datasets include UVG [1], MCL-JCV [55], and HEVC Class B [46], which contain high-quality videos with resolution 1920×1080. *For video recognition task*, we train and evaluate it on two large scale datasets requiring temporal relation reasoning, *i.e.*, Something V1&V2 [12][36].

4.2. Implementation Details

Video rescaling. λ_1, λ_2, λ_3 and λ_4 are set as 0.01, 1, 1 and 1, respectively. λ_1 is decayed to 0 in the last 200K iterations. Each training clip consists of 7 consecutive RGB patches of size 224×224. The batch size is set as 8. We augment the training data with random horizontal flips and 90° rotations. We train our model with Adam optimizer [25] by setting β_1 as 0.9, β_2 as 0.999, and learning rate as 10^{-4}. The total training iteration number is about 400K. The learning rate is divided by 2 every 100K iterations. We implement the models with the PyTorch framework [37] and train them with 2 NVIDIA 2080Ti GPUs. We draw 5 times from the generated distribution for each evaluation and report the averaged performance. We leverage the invertible neural network (INN) architecture to implement the CNN parts of the paired frequency analyzer and synthesizer for a fair comparison with IRN because INN will cut the number of parameters by 50%. We propose the two following models: the SelfC-*small* and SelfC-*large*, which consist of 2 and 8 invertible Dense2D-T blocks. The invertible architecture follows [60]. Training the SelfC-*large* model takes ∼6days.

Video compression. The rescaling ratio of the SelfC is set to 2. We adopt H.265 as the embedded codec. λ_4 is set as 0.1 to make sure the statistical distribution of the downscaled videos is more closed to the natural images, which stabilizes the performance of the whole system. λ_{codec} is empirically set as 4. The CRF value of the H.265 codec is randomly selected from {11,13,17,21} during training while it is set to a fixed value during evaluation. The input video clip length is set as 3. The other details follow that of video rescaling task. The models are initialized from SelfC-*large* model but the number of invertible Dense2D-T blocks is reduced to 4. The surrogate CNN is randomly initialized, and is jointly optimized with other components of SelfC online. It takes about 5days to train the model.

Action recognition. We insert the downscaler of our framework before the action recognition CNN, *i.e.*, TSM [28]. The data augmentation pipeline also follows it. The downscaling ratio of SelfC is 2. At inference time, we used just 1 clip per video and each clip contains 8 frames. We adopt 2 plain Dense2D-T blocks as the CNN part of the frequency analyzer. Note that the downscaler is first pretrained on Vimeo90K dataset with the rescaling task.

4.3. Results of Video Rescaling

As shown in Tab. 1 and Tab. 2, our method outperforms the recent state-of-the-art video super resolution methods on Vid4, SPMCs-30 and Vimeo90K-T by a large margin in terms of both PSNR and SSIM. For example, the result on the Vid4 dataset for our SelfC-large model is 32.85dB, while the state-of-the-art video super resolution approach RSDN only achieves 27.92dB. Furthermore, we also provide the results of the image rescaling method, *i.e.*, IRN, in

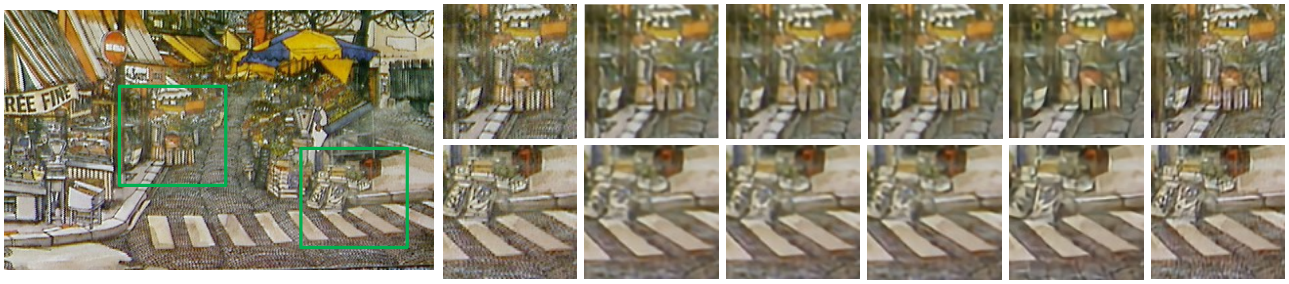

Figure 3: Qualitative comparison on the reconstruction of 4× downscaled *calendar* clip. *Best to view by zooming in.*

Downscaling	Upscaling	#Frame	FLOPs	#Param.	Calendar-Y	City-Y	Foliage-Y	Walk-Y	Average-Y	Average-RGB
Bicubic	Bicubic	1	N/A	N/A	18.83/0.4936	23.84/0.5234	21.52/0.4438	23.01/0.7096	21.80/0.5426	20.37/0.5106
Bicubic	SPMC [48]	3	-	-	-/-	-/-	-/-	-/-	25.52/0.76	-/-
Bicubic	Liu [30]	5	-	-	21.61/-	26.29/-	24.99/-	28.06/-	25.23/-	-/-
Bicubic	TOFlow [63]	7	0.81T	1.41M	22.29/0.7273	26.79/0.7446	25.31/0.7118	29.02/0.8799	25.85/0.7659	24.39/0.7438
Bicubic	FRVSR [42]	2	0.14T	5.05M	22.67/0.7844	27.70/0.8063	25.83/0.7541	29.72/0.8971	26.48/0.8104	25.01/0.7917
Bicubic	DUF-52L [23]	7	0.62T	5.82M	24.17/0.8161	28.05/0.8235	26.42/0.7758	30.91/0.9165	27.38/0.8329	25.91/0.8166
Bicubic	RBPN [17]	7	9.30T	12.2M	24.02/0.8088	27.83/0.8045	26.21/0.7579	30.62/0.9111	27.17/0.8205	25.65/0.7997
Bicubic	EDVR-L [57]	7	0.93T	20.6M	24.05/0.8147	28.00/0.8122	26.34/0.7635	31.02/0.9152	27.35/0.8264	25.83/0.8077
Bicubic	PFNL [67]	7	0.70T	3.00M	23.56/0.8232	28.11/0.8366	26.42/0.7761	30.55/0.9103	27.16/0.8365	25.67/0.8189
Bicubic	RLSP [10]	3	0.09T	4.21M	24.36/0.8235	28.22/0.8362	26.66/0.7821	30.71/0.9134	27.48/0.8388	25.69/0.8153
Bicubic	TGA [22]	7	0.23T	5.87M	24.50/0.8285	28.50/0.8442	26.59/0.7795	30.96/0.9171	27.63/0.8423	26.14/0.8258
Bicubic	RSDN 9-128 [21]	2	0.13T	6.19M	24.60/0.8355	29.20/0.8527	26.84/0.7931	31.04/0.9210	27.92/0.8505	26.43/0.8349
IRN	IRN [60]	1	0.24T	4.36M	26.62/0.8850	33.48/0.9337	29.71/0.8871	35.36/0.9696	31.29/0.9188	29.21/0.8990
SelfC-*small*	SelfC-*small*	7	0.043T	1.76M	27.10/0.9020	33.49/0.9379	30.46/0.9138	35.40/0.9730	31.61/0.9317	29.64/0.9151
SelfC-*large*	**SelfC-*large***	7	0.084T	3.37M	**27.82/0.9167**	**34.14/0.9454**	**30.95/0.9210**	**36.28/0.9778**	**32.30/0.9402**	**30.27/0.9245**

Table 1: Quantitative comparison (PSNR(dB) and SSIM) on Vid4 for 4× video rescaling. Y indicates the luminance channel. FLOPs (MAC) are calculated by upscaling an LR frame of size 180×120.

Downscaling	Bicubic	Bicubic	Bicubic	Bicubic	Bicubic	Bicubic	Bicubic	IRN [60]	SelfC-*small*	**SelfC-*large***
Upscaling	Bicubic	TOFlow [63]	FRVSR [42]	DUF-52L [23]	RBPN [17]	PFNL [67]	RSDN 9-128 [21]	IRN [60]	SelfC-*small*	**SelfC-*large***
SPMCs-30	23.29/0.6385	27.86/0.8237	28.16/0.8421	29.63/0.8719	29.73/0.8663	29.74/0.8792	-/-	36.24/0.9559	37.20/0.9681	**38.32/0.9744**
Vimeo90K-T	31.30/0.8687	34.62/0.9212	35.64/0.9319	36.87/0.9447	37.20/0.9458	-/-	37.23/0.9471	40.83/0.9734	40.68/0.9756	**41.53/0.9786**

Table 2: Quantitative comparison (PSNR-Y(dB) and SSIM-Y) on SPMCs-30 and Vimeo90K-T for 4× video rescaling.

Tab. 1 and Tab. 2. It is obvious that our method outperforms IRN significantly while also reduces the computational complexity by ∼3× (SelfC-*large*) or 6× (SelfC-*small*). This result clearly proves that it is necessary to exploit the temporal relationship for video rescaling task, while the existing image rescaling methods like IRN ignored this temporal cue. We also show the qualitative comparison with other methods on Vid4 in Fig. 3. Our method demonstrates much better details than both video super-resolution methods and image rescaling methods, proving the superiority of the video rescaling paradigm.

4.4. Results of Video Compression

In our framework, we adopt the H.265 codec in *default*[1] or *zerolatency*[2] modes. The first mode is for the offline storage of video data while the other one is oriented to the online low-latency video streaming services. The evaluation metrics are PSNR and MS-SSIM [58].

[1] *ffmpeg -pix_fmt yuv444p -s WxH -r 50 -i video.yuv -c:v libx265 -preset veryfast -x265-params "qp=Q"*

[2] *ffmpeg -pix_fmt yuv444p -s WxH -r 50 -i video.yuv -c:v libx265 -preset veryfast -tune zerolatency -x265-params "qp=Q"*

Fig. 4 shows the experimental results. It is obvious that our method outperforms both the traditional methods and learning-based methods (DVC [32], Yang *et al*. [66], Hu *et al*. [18], and Lu *et al*. [31]) on video compression task by a large margin. Although our method is only optimized by ℓ_1 loss, it demonstrates strong performances in terms of both PSNR and MS-SSIM metrics. It should also be mentioned that our models generalize well to *default* mode although only trained with the *zerolatency* mode of H.265 codec. We also evaluate the Bjntegaard Delta Bit-Rate (BDBR) [4] by using H.265 as the anchor method. As shown in Tab. 3, our method saves the bit cost or the storage space by over 30% averagely under the same MS-SSIM, compared with the vanilla H.265 codec. Notably, we reduce the bit cost by

H.265 mode	BDBR-SSIM			BDBR-PSNR
	UVG	MCL-JCV	HEVC B	UVG
default	-50.84	-36.48	-30.61	-49.01
zerolatency	-33.98	-28.16	-18.04	-19.41

Table 3: BDBR results using H.265 as the anchor. The lower value, the more bit cost reduced.

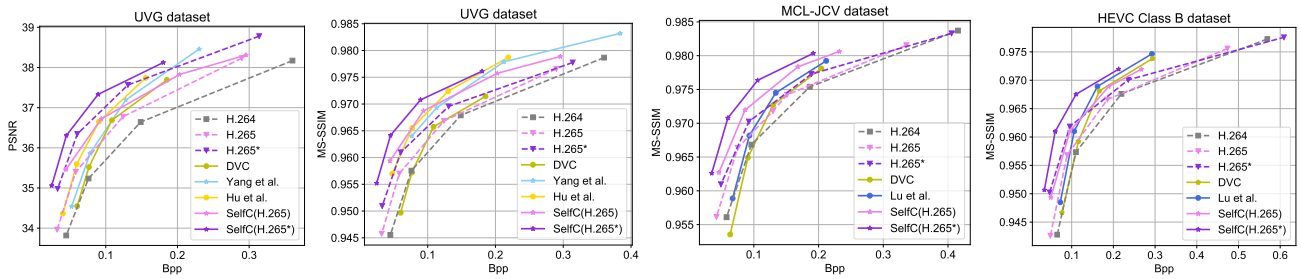

Figure 4: Comparison between the proposed method with H.265, H.264, and learnable video codecs. H.265 and H.265* denote the H.265 codec in *zerolatency* and *default* modes, respectively.

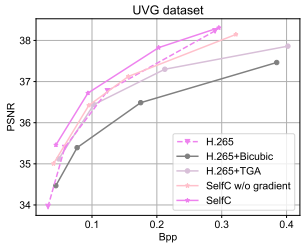

Figure 5: Comparison of our "Video rescaling+Codec" scheme with other paradigms. SelfC w/o gradient model is trained by neglecting the H.265 codec gradients [3].

over 45% on the UVG dataset. This proves that the video rescaling technique is a novel and effective way to improve video compression performance, without considering much about the complicated details of the industrial lossy codecs.

We perform more analysis to verify the effectiveness of the "Video rescaling+Codec" paradigm and the proposed gradient estimation method. As shown in Fig. 5, it is observed that using Bicubic as the downscaler and upscaler in the video compression system (*i.e.,* H.265+Bicubic) leads to a much inferior result than the baseline. We also try to improve the result by using a state-of-the-art video super resolution method, *i.e.*, TGA [22]. The performance is indeed improved though still lower than the baseline method H.265. Considering the network parameters of TGA are 5.87M while ours are only 0.88M, this result further demonstrates the effectiveness of our SelfC framework. Finally, we provide experimental results (*i.e.,* SelfC w/o gradient) when directly using the biased Straight-Through Estimator [3] for approximating the gradient of H.265. The results show that the proposed gradient estimation method in Section 3.6 can bring nearly 0.3dB improvements.

Finally, We give the complexity analysis of the proposed video compression system. Although it seems like that our method adds extra computation costs upon the H.265 codec, our system is indeed more efficient because the input videos to codec are downscaled. Concretely, under the *zerolantency* mode, for one frame with resolution 1920×1080, the average encoding time of our method is 108ms, including 21ms for 2×downscaling and 87ms consumed by the embedded H.265 codec. Our system improves the efficiency of the vanilla H.265 codec (116ms) and is also ∼5× faster than the learnable codec DVC (522ms).

4.5. Results of Efficient Video Action Recognition

We show the video action recognition results in Tab. 4. In the first group, when directly testing the action model pretrained on full resolution videos, we observe the performances of low-resolution videos downscaled by Bicubic and ours are both dropped drastically, because the classification networks are rather sensitive to the absolute scale of the input. However, our downscaler still performs better (30.4% vs. 32.7% on Something V1 dataset).

In the second group, we provide the experimental results when the action recognition CNN is fine-tuned on the low-resolution videos downsampled by Bicubic and our downscaler. It is obvious that our method clearly outperforms Bicubic downsampler by about 1.5% in terms of Top1 accuracy on the two datasets. Notably, our downscaler is learnable. Therefore, we then jointly fine-tune action recognition CNNs and our downscaler. The results in the last group show that the end-to-end joint training strategy further improves the performance by an obvious margin. On Something V2, the ultimate performances of our method nearly achieve that of performing recognition directly on HR videos, and our method improves the efficiency by over 3 times. The downscaler of IRN can not improve the efficiency of this task because its computation cost is even larger than the HR setting. We try to decrease the layer number of IRN but it no longer converges.

Method	Action CNN	FLOPs	Params	V1		V2	
				Top1 (%)	Top5 (%)	Top1 (%)	Top5 (%)
HR	TSM	33.0 G	24.31M	45.6	74.2	58.8	85.4
Bicubic	TSM	8.6G	24.31M	30.4	57.5	40.1	68.7
Ours	TSM	10.8G	24.35M	32.7	59.5	41.8	71.5
Bicubic (FT)	TSM	8.6G	24.31M	42.1	72.1	56.0	83.9
Ours (FT)	TSM	10.8G	24.35M	43.5	72.8	57.4	84.8
Ours (E2E)	TSM	10.8G	24.35M	**44.6**	**73.3**	**58.3**	**85.5**

Table 4: Comparison between our method and Bicubic downscaler. FT denotes only fine-tuning the action recognition CNN. E2E denotes also fine-tuning the dowscaler.

4.6. Ablation Studies on the Framework

In this section, we conduct experiments on video rescaling task to verify the effectiveness of the components in

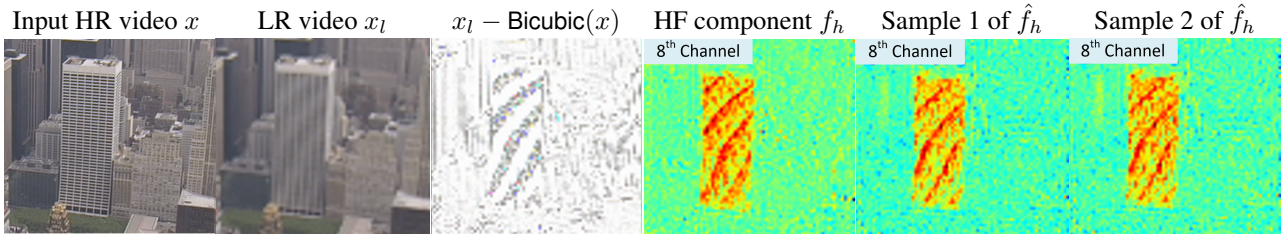

Figure 6: Visualization of the high-frequency component $\hat{f}_h$ sampled from the learned distribution $p(f_h|x_l)$. We also compare the difference of the downscaled video by our learnable downscaler and the Bicubic downsampler. Since the magnitude of the difference is small, we amplify it by 10 times for better visualization.

our framework. We first define the following 2 baselines: (1) IRN [60], which is the most recent state-of-the-art image rescaling method. For fair comparison, we re-train it on Vimeo90K dataset using the codes open-sourced by the authors. (2) Auto-Enc, which is a simple auto encoder-decoder architecture by removing the STP-Net of our model. The experimental results are shown in Tab. 5.

Methods	Backbone	Probability model	Param(M)	Vid4-Y	
				PSNR(dB)	SSIM
Auto-Enc	16×Dense2D-T	-	3.63	28.91	0.8797
IRN*	16×Dense2D	Normal	4.36	30.68	0.9067
IRN-T	16×Dense2D-T	Normal	3.63	30.42	0.9004
SelfC-*basic*	2×Dense2D	GMM(K=1)	1.77	30.62	0.9214
SelfC-*basicT*	2×Dense2D-T	GMM(K=1)	1.61	31.29	0.9268
SelfC-*small*	2×Dense2D-T	GMM(K=5)	1.76	31.61	0.9317
SelfC-*large*	8×Dense2D-T	GMM(K=5)	3.37	**32.30**	**0.9402**

Table 5: Ablation studies on 4× video rescaling. All blocks adopt the INN architecture for a fair comparison with IRN. * indicates the model is re-trained on Vimeo90K.

First, the Auto-Enc baseline shows more inferior performance than both IRN and our method. This proves that explicitly modeling the lost information is important. IRN is inferior to SelfC-*small* model although IRN leverages an 8 times heavier backbone. We also tried to equip IRN with the temporal modeling ability by replacing its backbone from Dense2D to Dense2D-T. Surprisingly, the performance of the resulted model IRN-T decreases by 0.26dB. The reason is that IRN relies on the complex non-linear transformation to transform the real distribution of the lost information to the normal distribution while the transformation ability of the Dense2D-T is weaker (missing 0.73M parameters).

For our method, we start from the most simple model denoted by SelfC-*basic*, where the backbone consists of only spatial convolutions, and the STP-Net only outputs a simple Gaussian distribution. The performance of this model is comparable with IRN but with 2× fewer parameters. This proves the efficiency and superiority of the proposed self-conditioned distribution modeling scheme. Then, we introduce an improved model denoted by SelfC-*basicT*. The temporal modeling ability of the model is enhanced by changing the basic block from Dense2D to Dense2D-T. This leads to 0.67dB improvement while reducing the parameters, proving the effectiveness of the Dense2D-T block for video tasks. Further, we increase the mixture number of the GMM model to 5. The resulted SelfC-*small* model outperforms all the baselines by a large margin with only 1.76M parameters. Our model is also scalable with larger backbone network. Enlarging the backbone by 4 times further improves the performance by 0.69dB.

4.7. Visualization Results

While the previous quantitative results validate the superiority of the proposed self-conditioned modeling scheme on several tasks, it is interesting to investigate the intermediate components output by our model, especially the distribution of the high-frequency (HF) component predicted by STP-Net. Note that the distribution is a mixture of Gaussian and includes multiple channels, we draw two samples of $\hat{f}_h$ from $p(f_h|x_l)$ and randomly select 1 channel of them for visualization. The f_h output from the frequency analyzer is adopted as the ground-truth sample.

As shown in Fig. 6, we first see that the LR video x_l downscaled by our method is modulated into some mandatory information for reconstructing the HF components more easily, compared to Bicubic. Also, the sampled HF components can restore the ground-truth of that accurately in terms of key structures, *i.e.*, the windows of the building, while retaining a certain degree of randomness. This is consistent with our learning objectives.

5. Conclusion

We have proposed a video-rescaling framework to learn a pair of downscaling and upscaling operations. Extensive experiments demonstrated that our method can outperform the previous methods with a large margin while with much fewer parameters and computational costs. Moreover, the learned operators facilitates the tasks of video compression and efficient action recognition significantly.

Acknowledgement This work was supported by the National Science Foundation of China (61831015, 61527804 and U1908210).

References

[1] Ultra video group test sequences. http://ultravideo.cs.tut.fi, accessed: 2019- 11-06. 5

[2] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Balle, Sung Jin Hwang, and George Toderici. Scale-space flow for end-to-end optimized video compression. In *CVPR*, 2020. 2

[3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv*, 2013. 4, 7

[4] Gisle Bjontegaard. Calculation of average psnr differences between rd-curves. *VCEG-M33*, 2001. 6

[5] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 2

[6] Abdelaziz Djelouah, Joaquim Campos, Simone Schaub-Meyer, and Christopher Schroers. Neural inter-frame compression for video coding. In *ICCV*, 2019. 2

[7] Yin Fan, Xiangju Lu, Dian Li, and Yuanliu Liu. Video-based emotion recognition using cnn-rnn and c3d hybrid networks. In *ACM MI*, 2016. 1

[8] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, 2019. 2

[9] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016. 2

[10] Dario Fuoli, Shuhang Gu, and Radu Timofte. Efficient video super-resolution through recurrent latent space propagation. In *ICCVW*, 2019. 6

[11] Peter W Glynn and Roberto Szechtman. Some new perspectives on the method of control variates. In *Monte Carlo and Quasi-Monte Carlo Methods 2000*. 2002. 4

[12] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, 2017. 5

[13] Will Grathwohl, Dami Choi, Yuhuai Wu, Geoffrey Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. *arXiv*, 2017. 4

[14] Alex Graves. Stochastic backpropagation through mixture density distributions. *arXiv*, 2016. 4

[15] Amirhossein Habibian, Ties van Rozendaal, Jakub M Tomczak, and Taco S Cohen. Video compression with rate-distortion autoencoders. In *ICCV*, 2019. 2

[16] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. In *ICCVW*, 2017. 2

[17] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Recurrent back-projection network for video super-resolution. In *CVPR*, 2019. 2, 6

[18] Zhihao Hu, Zhenghao Chen, Dong Xu, Guo Lu, Wanli Ouyang, and Shuhang Gu. Improving deep video compression by resolution-adaptive flow coding. In *ECCV*, 2020. 5, 6

[19] Zhihao Hu, Guo Lu, and Dong Xu. Fvc: A new framework towards deep video compression in feature space. In *CVPR*, 2021. 2

[20] Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. In *arXiv*, 2014. 3

[21] Takashi Isobe, Xu Jia, Shuhang Gu, Songjiang Li, Shengjin Wang, and Qi Tian. Video super-resolution with recurrent structure-detail network. In *ECCV*, 2020. 5, 6

[22] Takashi Isobe, Songjiang Li, Xu Jia, Shanxin Yuan, Gregory Slabaugh, Chunjing Xu, Ya-Li Li, Shengjin Wang, and Qi Tian. Video super-resolution with temporal group attention. In *CVPR*, 2020. 5, 6, 7

[23] 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 *CVPR*, 2018. 2, 6

[24] Heewon Kim, Myungsub Choi, Bee Lim, and Kyoung Mu Lee. Task-aware image downscaling. In *ECCV*, 2018. 1, 2, 3

[25] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *arXiv*, 2014. 5

[26] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv*, 2013. 4

[27] Yue Li, Dong Liu, Houqiang Li, Li Li, Zhu Li, and Feng Wu. Learning a convolutional neural network for image compact-resolution. In *TIP*, 2018. 1, 2

[28] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *ICCV*, 2019. 2, 5

[29] Jianping Lin, Dong Liu, Houqiang Li, and Feng Wu. M-lvc: multiple frames prediction for learned video compression. In *CVPR*, 2020. 2

[30] Ce Liu and Deqing Sun. On bayesian adaptive video super resolution. In *TPAMI*, 2013. 1, 2, 5, 6

[31] Guo Lu, Chunlei Cai, Xiaoyun Zhang, Li Chen, Wanli Ouyang, Dong Xu, and Zhiyong Gao. Content adaptive and error propagation aware deep video compression. In *ECCV*, 2020. 2, 5, 6

[32] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. Dvc: An end-to-end deep video compression framework. In *CVPR*, 2019. 2, 5, 6

[33] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Zhiyong Gao, and Ming-Ting Sun. Deep kalman filtering network for video compression artifact reduction. In *ECCV*, 2018. 1

[34] Guo Lu, Xiaoyun Zhang, Wanli Ouyang, Li Chen, Zhiyong Gao, and Dong Xu. An end-to-end learning framework for video compression. *TPAMI*, 2020. 2

[35] Guo Lu, Xiaoyun Zhang, Wanli Ouyang, Dong Xu, Li Chen, and Zhiyong Gao. Deep non-local kalman network for video compression artifact reduction. *TIP*, 2019. 1

[36] Farzaneh Mahdisoltani, Guillaume Berger, Waseem Gharbieh, David Fleet, and Roland Memisevic. Fine-grained video classification and captioning. In *arXiv*, 2018. 5

[37] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 5

[38] Javier Portilla, Vasily Strela, Martin J Wainwright, and Eero P Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. In *TIP*, 2003. 2

[39] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *ICCV*, 2017. 2

[40] Douglas A Reynolds. Gaussian mixture models. In *EOB*, 2009. 3

[41] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014. 4

[42] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *CVPR*, 2018. 6

[43] 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 *CVPR*, 2016. 3

[44] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, 2014. 1, 2

[45] Vasily Strela, Javier Portilla, and Eero P Simoncelli. Image denoising using a local gaussian scale mixture model in the wavelet domain. In *Wavelet Applications in Signal and Image Processing VIII*, 2000. 2

[46] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *TCSVT*, 2012. 2, 5

[47] Wanjie Sun and Zhenzhong Chen. Learned image downscaling for upscaling using content adaptive resampler. In *TIP*, 2020. 1

[48] Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. Detail-revealing deep video super-resolution. In *ICCV*, 2017. 1, 2, 5, 6

[49] Yuan Tian, Zhaohui Che, Wenbo Bao, Guangtao Zhai, and Zhiyong Gao. Self-supervised motion representation via scattering local motion cues. In *ECCV*, 2020. 2

[50] Yuan Tian, Xiongkuo Min, Guangtao Zhai, and Zhiyong Gao. Video-based early asd detection via temporal pyramid networks. In *ICME*, 2019. 1

[51] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015. 2

[52] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, 2018. 2

[53] Martin J Wainwright and Eero Simoncelli. Scale mixtures of gaussians and the statistics of natural images. In *NeurIPS*, 1999. 2

[54] Martin J Wainwright, Eero P Simoncelli, and Alan S Willsky. Random cascades on wavelet trees and their use in analyzing and modeling natural images. In *ACHA*, 2001. 2

[55] Haiqiang Wang, Weihao Gan, Sudeng Hu, Joe Yuchieh Lin, Lina Jin, Longguang Song, Ping Wang, Ioannis Katsavounidis, Anne Aaron, and C-C Jay Kuo. Mcl-jcv: a jnd-based h. 264/avc video quality assessment dataset. In *ICIP*, 2016. 5

[56] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. In *TPAMI*, 2018. 2

[57] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *CVPRW*, 2019. 1, 2, 5, 6

[58] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *ACSSC*, 2003. 6

[59] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the h. 264/avc video coding standard. *TCSVT*, 2003. 2

[60] Mingqing Xiao, Shuxin Zheng, Chang Liu, Yaolong Wang, Di He, Guolin Ke, Jiang Bian, Zhouchen Lin, and Tie-Yan Liu. Invertible image rescaling. In *ECCV*, 2020. 1, 2, 5, 6, 8

[61] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. In *arXiv*, 2015. 3

[62] Zhongwen Xu, Yi Yang, and Alex G Hauptmann. A discriminative cnn video representation for event detection. In *CVPR*, 2015. 1

[63] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. In *IJCV*, 2019. 1, 2, 5, 6

[64] Zhicong Yan, Gaolei Li, Yuan TIan, Jun Wu, Shenghong Li, Mingzhe Chen, and H Vincent Poor. Dehib: Deep hidden backdoor attack on semi-supervised learning via adversarial perturbation. In *AAAI*, 2021. 1

[65] Fengting Yang, Qian Sun, Hailin Jin, and Zihan Zhou. Superpixel segmentation with fully convolutional networks. In *CVPR*, 2020. 2

[66] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for video compression with hierarchical quality and recurrent enhancement. In *CVPR*, 2020. 2, 6

[67] 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 *ICCV*, 2019. 6