Online Streaming Video Super-Resolution With Convolutional Look-Up Table

Guanghao Yin, Zefan Qu, Xinyang Jiang, Shan Jiang, Zhenhua Han, Ningxin Zheng, Huan Yang, Xiaohong Liu, *Member, IEEE*, Yuqing Yang, Dongsheng Li, *Senior Member, IEEE*, and Lili Qiu

***Abstract*— Online video streaming has fundamental limitations on the transmission bandwidth and computational capacity and super-resolution is a promising potential solution. However, applying existing video super-resolution methods to online streaming is non-trivial. Existing video codecs and streaming protocols (*e.g.*, WebRTC) dynamically change the video quality both spatially and temporally, which leads to diverse and dynamic degradations. Furthermore, online streaming has a strict requirement for latency that most existing methods are less applicable. As a result, this paper focuses on the rarely exploited problem setting of online streaming video super resolution. To facilitate the research on this problem, a new benchmark dataset named LDV-WebRTC is constructed based on a real-world online streaming system. Leveraging the new benchmark dataset, we propose a novel method specifically for online video streaming, which contains a convolution and Look-Up Table (LUT) hybrid model to achieve better performance-latency trade-off. To tackle the changing degradations, we propose a mixture-of-expert-LUT module, where a set of LUT specialized in different degradations are built and adaptively combined to handle different degradations. Experiments show our method achieves 720P video SR around 100 FPS, while significantly outperforms existing LUT-based methods and offers competitive performance compared to efficient CNN-based methods. Code is available at https://github.com/quzefan/ConvLUT.**

***Index Terms*— Adaptive online bitstream, online video super-resolution, look-up table.**

Manuscript received 18 June 2023; revised 10 November 2023 and 28 December 2023; accepted 15 February 2024. Date of publication 18 March 2024; date of current version 25 March 2024. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Nam Ik Cho. *(Guanghao Yin and Zefan Qu contributed equally to this work.) (Corresponding author: Xinyang Jiang.)*

Guanghao Yin was with Microsoft Research Asia, Shanghai 200232, China. He is now with the Key Laboratory of Design Intelligence and Digital Creativity of Zhejiang Province, Zhejiang University, Hangzhou 310027, China (e-mail: ygh_zju@zju.edu.cn).

Zefan Qu was with Microsoft Research Asia, Shanghai 200232, China. He is now with the Department of Computer Science and Technology, Tongji University, Shanghai 200092, China (e-mail: qzf@tongji.edu.cn).

Xinyang Jiang, Zhenhua Han, Ningxin Zheng, Huan Yang, Yuqing Yang, Dongsheng Li, and Lili Qiu are with Shanghai AI/ML Group, Microsoft Research Asia, Shanghai 200232, China (e-mail: xinyangjiang@microsoft.com; zhenhua.han@microsoft.com; ningxin.zheng@microsoft.com; huan.yang@microsoft.com; yuqing.yang@microsoft.com; dongsheng.li@microsoft.com; liliqiu@microsoft.com).

Shan Jiang was with Microsoft Research Asia, Shanghai 200232, China. He is now with the School of Computer Science and Technology, University of Science and Technology of China (USTC), Hefei 230026, China (e-mail: jiangshan@ustc.edu.cn).

Xiaohong Liu is with the John Hopcroft Center (JHC) for Computer Science, Shanghai Jiao Tong University (SJTU), Shanghai 200240, China (e-mail: xiaohongliu@sjtu.edu.cn).

Digital Object Identifier 10.1109/TIP.2024.3374104

I. Introduction

With the fast development of network infrastructure, video delivery techniques, and the growing demand of users, video streaming has become the "killer" application of the Internet in the past two decades [1]. Due to users' steep expectations for quality, delivering high-definition (HD) video to end users is important. However, the quality of streamed video heavily depends on the network bandwidth between servers and clients. Streaming 4K videos require over 40 Mbps bandwidth per user [2], which is difficult to achieve in many areas. With the ever-increasing computational power of client devices and advances in deep learning, super-resolution (SR), which aims to restore high-resolution (HR) frames by adding the missing details from low-resolution (LR) frames, has been considered as a promising direction to reduce the bandwidth requirement of streaming HD videos [2], [3], [4], [5], [6], [7], [8], [9].

While previous works have achieved great progress [2], [5], [6], [8], [9], [10], [11], existing super-resolution methods still face great challenges in real-world video streaming. Adapting existing video super resolution (VSR) methods to streaming data is non-trivial for two reasons. First, the VSR task requires the inference speed of the SR method to be fast (*i.e.*, low-latency), especially for online scenarios involving real-time user interaction (*e.g.*, video conference or cloud gaming), where slight latency will significantly harm the user experience. Most state-of-the-art VSR methods involve high complexity models and need to cache future frames for super-resolving the current frame, inevitably introducing high latency. Second, videos transmitted by streaming system suffer dynamically changing degradations both temporally and spatially. In temporal domain, in order to adapt the time-varying network conditions, existing video streaming protocols (*e.g.*, WebRTC [12]) adopt Adaptive Bitrate Streaming (ABR), which adaptively changes the quality of the video frame at each time step, as shown in Fig. 1. In spatial domain, the codec used in the streaming system applies different compression configurations to each macro-block within a frame, resulting in degradation variations across different macro-blocks. Thus, it is essential to develop VSR models capable of dealing with spatial and temporal changing degradations.

As a result, this paper focuses on a rarely studied problem setting, which aims at super-resolving videos transmitted by real-world online streaming system, named online streaming VSR. Since existing VSR datasets lack videos produced

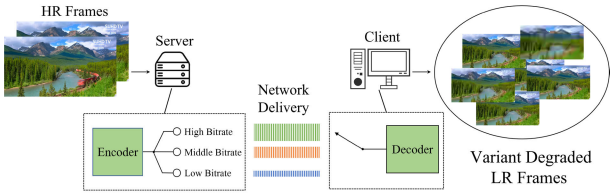


Fig. 1. Pipeline of online video streaming. A server uses the Adaptive Bitrate streaming (ABR) to encode frames at multiple bitrates for delivery, and the clients select one of the streams to decode. During this adaptive delivery, the spatial-temporal dynamic degradations are inevitably introduced.

from real-world streaming system, a new online streaming VSR dataset named LDV-WebRTC is proposed. Specifically, we built a real-world online video streaming prototype based on Web Real-Time Communication (WebRTC) [12], which is used to transmit videos from LDV 2.0 dataset [13] with ABR under different network conditions. As shown in Fig. 2, compared to un-compressed videos or videos compressed with fixed quantization parameter (QP), the quality and compression configurations of the videos in this dataset vary significantly through time, which further verifies the drastically changing degradations. We have developed a benchmark based on this dataset to assess both SR performance and model latency of various VSR baselines.

Leveraging the proposed dataset, this paper proposes a novel method tackling the aforementioned challenges of online streaming VSR, named ConvLUT. To meet the low-latency requirement, we propose a novel hybrid network structure combining convolution and Look-Up Table (LUT) [14], [15], [16], [17], [18], [19], [20], resulting in a balanced trade-off between the high inference efficiency of LUT and the strong computational capacity of neural networks. To tackle the spatial-temporal dynamic degradations, ConvLUT adopts a novel mixture-of-expert-LUT module. This module contains a set of expert LUTs specialized in specific degradations, which are built from state-of-the-art (SOTA) SR networks trained on a pool of sampled degradations. The expert LUTs are adaptively combined to address the diverse degradations across different macro-blocks at each time step. Furthermore, since the proposed Convolution-LUT hybrid structure is unfriendly to parallel computing accelerators, such as GPU, NPU, and FPGA [21], [22], [23], we further propose an efficient interpolation algorithm to support LUT inference in parallel.

In conclusion, our work contributes to threefold:

(1) We delve into the largely unexplored yet challenging realm of online streaming video super-resolution. Our study stands out as it specifically tackles the need for high-speed inference and the ability to adapt to fluctuating video degradation modes associated with changing network states.

(2) To facilitate research in online video streaming super-resolution, we introduce the first real-world video streaming SR dataset, LDV-WebRTC. This unique dataset, built upon Web Real-Time Communication (WebRTC) and LDV 2.0, provides a diverse range of network conditions.

(3) A novel ConvLUT hybrid VSR framework, is designed to handle real-time process latency and manage spatial-temporal degradation variations using a mixture of expert LUTs. Our extensive qualitative and quantitative experiments, conducted on the proposed LDV-WebRTC dataset, demonstrate the effectiveness and efficiency of our novel approach.

II. Related Work

A. Adaptive Online Bitstream

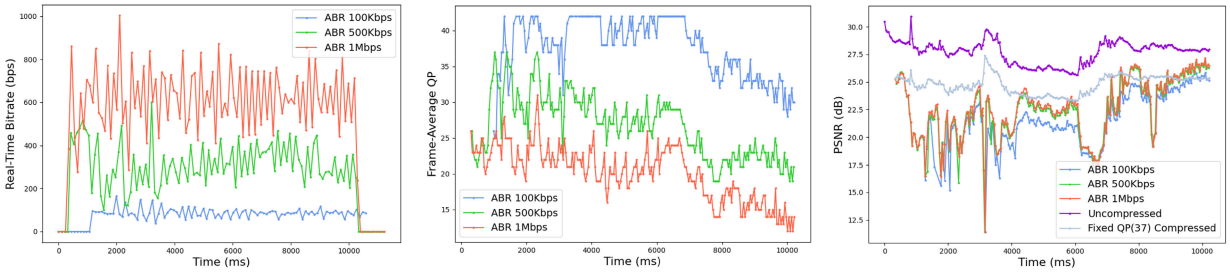
Adaptive online streaming aims to handle unpredictable bandwidth variations for high-quality video delivery. In online streaming, a server first utilizes Video Coding Protocols, such as H.264/AVC [24] or HEVC [25] to encode image frames at multiple bitrates by adjusting QP in both spatial and temporal domains. Then the client uses an ABR algorithm to select suitable video quality and decode the received frames [26]. During this adaptive delivery, coding artifacts are inevitably introduced [27]. Therefore, one of the major challenges for high-quality online streaming is to handle the spatial-temporal dynamic degradations adaptively.

B. Super-Resolution

Since the pioneering SRCNN [3], deep learning based approaches [3], [4], [6] have exhibited impressive performance in single-image SR (SISR) tasks. By considering the potential dependency in consecutive frames, various VSR models [2], [5], [7], [8], [9], [28], [29] have achieved great success, many of which adopt computational intensive modules such as optical flow alignment, deformable convolution and transformer. Recently, there has been an increasing interest in efficient SR. For example, CARN [5] replaces the conventional convolutions with group convolutions, which reduce the parameters of its original big model. VESPCN [7] uses lightweight motion estimation and pixel-shuffle modules to conduct spatial-temporal upscaling. RRN [2] removes the optical flow based alignment, but directly uses hidden states of recurrent proceedings to involve temporal information.

C. Look-Up Table

Look-Up Table is an efficient tool for classic image processing because it can replace complex computations with direct query operations. The pre-defined LUT has been widely used as the template to adjust the pixel distribution in photo editing and camera imaging [14]. Recent deep models have also extended LUTs to low-level vision tasks [14], [15], [16], [17], [18]. For color enhancement, Zeng et al. [14] propose learnable 3D LUT to achieve image-level LUT adaption. Yang et al. [30] propose a more flexible sampling point allocation to adaptively learn the non-uniform sampling intervals in 3D color space. Liu et al. [16] propose a learnable context-aware 4D LUT to achieve content-dependent enhancement. Recently, some new attempts have also been proposed for super-resolution. SR-LUT [17] first use a single 4D LUT to transfer the LR-HR mappings from a pretrained SR model with small receptive field (RF). SPLUT [18] uses the parallel cascaded LUTs to process the high and low 4-bit components of 8-bit LR images. Meanwhile, the padding aggregations



(a) Variation of Real-Time Bitrate (b) Variation of Frame Average QP (c) Variation of PSNR

Fig. 2. The variations of real-time bitrates, frame-average QP, and PSNR of streamed test video 005 using our LDV-WebRTC testbed under 100Kbps, 500Kbps and 1Mbps bandwidths.

are also applied to enlarge the receptive field of LUT. Nevertheless, the fixed LUT mapping from the simple-designed network structures still limits their performance for dynamic degradations.

III. Online Streaming VSR and Dataset

Online Streaming VSR is a rarely studied problem setting, and since existing VSR datasets either contain uncompressed raw video frames [13], [31], [32] or frames compressed with fixed presets [13], they do not reflect the spatial-temporal changing degradations of online video streaming. Hence, to better tackle real-world challenges and facilitate research on online streaming VSR, we collect a new video SR dataset under real-world online streaming settings, named LDV-WebRTC.

WebRTC [12] is a real-time communication protocol that is widely used to stream real-time videos to browsers or mobile devices. We build a video streaming prototype based on WebRTC, which uses a server to stream low-resolution videos to a laptop client via a router. The router uses Linux `tc` to control the network bandwidth between the server and the client to emulate the diverse bandwidth settings of real-world applications. We collect all 335 high-resolution videos in the LDV 2.0 dataset [13], containing a rich diversity of content scenes whose frame resolution is 960×512. The high-resolution frames are downsampled by $4\times$ in bicubic mode to get 240×128 low-resolution frames, which are encoded via FFmpeg-H.264 [33] and then transmitted from the servers. The WebRTC server enables Adaptive Bitrate streaming (ABR) that adjusts encoding quality with QP values in a range of $[0, 50]$, according to factors including network bandwidth, encoding latency, and decoding latency. A larger QP leads to worse quality of encoded frames. After receiving an encoding video stream, the client decodes it into a sequence of decoded low-resolution frames with timestamps. The target of video streaming SR is to restore those received LR frames. To involve different network conditions, three representative types of networks are emulated in the router with an average bandwidth of 100kbps, 500kbps, and 1Mbps. Note that, when bandwidth is limited, not all frames are successfully received due to frame drop. Thus we align the decoded frames at the client with the original high-resolution frames using encoding timestamps. In addition to the decoded frames, we also collect the motion vector priors extracted by the video codec of the streaming system.

TABLE I
Statistical Results of Our LDV-WebRTC Dataset

Bandwidth	Dataset	Bitrate (Kbps)	QP	PSNR (dB)	Frame Number
100Kbps	Training	85.15	30.79	24.37	63792
	Validation	84.79	29.75	23.72	4415
	Test	82.26	29.87	23.31	5558
500Kbps	Training	334.56	20.90	25.04	74207
	Validation	340.05	18.80	24.68	5926
	Test	322.05	20.42	24.27	5980
1Mbps	Training	497.22	17.73	25.21	73603
	Validation	476.10	16.21	24.79	5939
	Test	486.48	17.54	24.35	6078

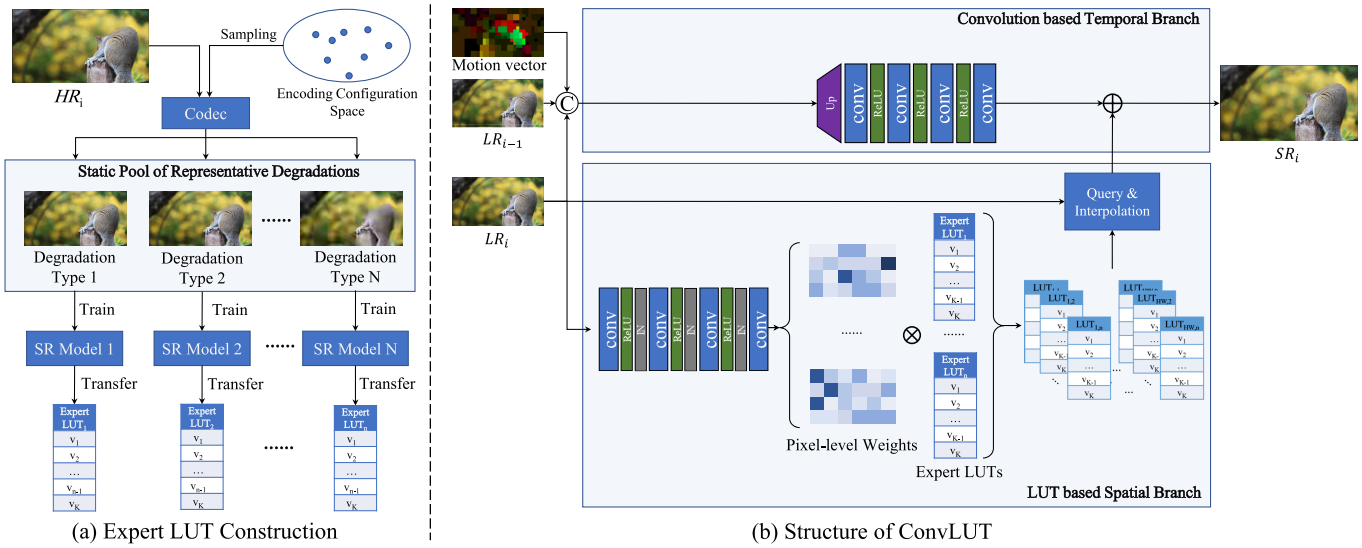
Fig. 2 illustrates the statistics of real-time bitrates, QP and PSNR of streamed frames using our WebRTC testbed under different network bandwidths. It is clear that the PSNR of real streamed video fluctuated more severely due to the real-time encoding-decoding pipeline and the variation in bitrates. The QP values of encoded frames also vary greatly and sometimes even trigger resolution changes (*i.e.*WebRTC's default strategy degrades resolution when the frame-average QP is quite large). Moreover, the frame drop also happens frequently for online streaming. All those observations demonstrate the necessity of building a more realistic dataset that reflects the diverse and time-varying degradation of real-world online video streaming.

In conclusion, our dataset consists of the aligned LR frames of the client, their original HR versions of the server, and the bitstream priors at 100Kbps, 500Kbps, and 1Mbps bandwidths. Table I illustrates the statistical results of average real-time bitrates, QP and PSNR of streamed frames on the training, validation, and test sets, which are collected by our WebRTC testbed under different network bandwidths. When the bandwidth setting decreases from 1Mbps to 100Kbps, the real bitrate decreases and the online streaming system uses higher QP to compress frames, which causes lower image quality. Moreover, frame drops occur more frequently as the bandwidth decreases. All those observations reveal the online streaming degradations are dynamic and challenging.

IV. Method

A. ConvLUT Hybrid Network Architecture

As shown in Fig. 3, we design a hybrid VSR network combining the convolution and LUT for online streaming,



(a) Expert LUT Construction (b) Structure of ConvLUT

Fig. 3. The overall scheme of our proposed ConvLUT. (a) First, a set of n expert LUTs are built from N SR networks pre-trained on N static representative degradations sampling from the configuration space. (b) Then, the spatial and temporal branches of our ConvLUT conduct adaptive LUT fusion and multiple-frame fusion in parallel, and the outputs of the two branches are added to get the final SR results.

which contains two parallel branches. The LUT based spatial branch learns a combination of a group of expert LUTs to handle the dynamic degradation in the spatial dimension, and the convolution based temporal branch further refines the LUT outputs with temporal information from history frames and the priors of codec (i.e. motion vectors).

B. Mixture of Expert LUT

As explained above, the challenge of online streaming VSR is to handle the dynamic degradations at real-time speed and with low latency. Despite LUT's fast inference speed and small parameter requirements, existing LUT-based super-resolution (SR) methods [17], [18] fail to deliver promising results in online streaming scenarios. This is due to the fact that their LUTs are transferred from a single SR model with a simple network structure, which constrains their ability to adapt to various or complex forms of degradation. Additionally, these methods apply the LUTs for each pixel solely within a small query patch, which results in limited texture and structural information due to their small receptive fields. This, in turn, creates a significant challenge for online streaming VSR. To address it, we propose to create a group of expert LUTs, each of which specializes in a particular type of degradation. Specifically, we propose transferring each expert LUT from a state-of-the-art SR network that has been trained on a particular static degradation. During inference, the expert LUTs are adaptively fused to handle macro-blocks in each video frame with different degradations.

In this section, we elaborate on how to construct the expert LUTs (IV.B.1) and how to dynamically fuse the expert LUTs for degraded macro-blocks (IV.B.2). Moreover, we present an efficient LUT query and interpolation method that makes LUT more compatible with parallel acceleration (IV.B.3).

1) Transferring SR Networks to Expert LUTs: Here, we introduce how to build expert LUTs specialized for different degradations. The degradation variation of video streaming systems is primarily caused by changes in the configuration of video compression. In addition to the down-sampling operation, the quantization process used in video compression is the main source of degradation, leading to a loss of details and introducing artifacts. The combination of various compression parameters leads to an extensive configuration space, making it impractical to explore all possible options. As a result, we select a static pool of representative degradations, sampling compression parameter settings from the configuration space. We then train SR networks on the videos corresponding to each degradation in the pool, obtaining SR models specialized in addressing specific type of degradation. Following [17], [18], and [19], we treat the RGB channels equally with the same SR networks and LUTs to reduce the LUT storage space and query time consumption.

As shown in Fig. 3(a), in our experiments, the configuration space is defined by quantization parameter (QP), which is one of the most critical parameters to control the quantization process in video compression. Higher QP indicates higher compression ratio and lower video quality. We uniformly sample N different QP values from $D_{qp\in[0,m]}$ corresponding to N different degradations $\{D_i\}_{i=0}^{N}$, and use them to generate N video SR training subsets $\{\mathcal{X}^{D_i}\}_{i=1}^{N}$. As a result, N SR models $\{f_{sr_i}\}_{i=1}^{N}$ are trained on those N subsets. It should be noted that the structure of the SISR model has no restrictions and any SOTA methods can be applied for training.

The pre-trained SR networks are then transferred to expert LUTs specialized in different degradations, which are later dynamically fused to handle degradation variation. Following previous LUT works [14], [16], [17], [18], each expert LUT stores a mapping between a LR patch and the patch super-resolved by a SR network. Specifically, we create a full value permutation of a 2×2 patches, which are 256^4 patches in total ranging from [0,0,0,0] to [255,255,255,255]. The SR model f_{sr_i} takes each 2×2 patch as input, and we store all super-resolved $r \times r$ patches at the left-upper place of

the $2r \times 2r$ SR results into the LUT, which up-samples the left-upper pixel of the low-resolution patch by scale factor r. Eventually, when all permutations of input patches are processed by SR model f_{sr_i}, we get the transferred LUT_i with the size of $[256, 256, 256, 256, r, r]$. Moreover, we follow SR-LUT [17] to compress LUT_i by uniformly sampling the original LUT with the interval size of 16, resulting in the compressed LUT_i with the size of $[17, 17, 17, 17, r, r]$.

2) Adaptive LUT Fusion: Given that the degradation of an online streaming video varies both temporally and spatially, it is crucial to obtain a specialized LUT for each macro-block within each frame corresponding to its particular degradation. Inspired by the concept of mixture of experts [34], [35], [36], we can create a spatially and temporally variant look-up table by combining expert LUTs with different weight combinations at each pixel position, allowing LUTs to effectively adapt to any type of degradation. Specifically, a lightweight predictor is proposed to output the LUT combination weights for each pixel based on the content of the input frames. The weight predictor, denoted as f_w, consists of 4 convolution layers with Instance Normalization [37] and LeakyReLU [38] operations. For the input LR frame $X \in \mathbb{R}^{h \times w \times 3}$, f_w outputs a weight tensor $W \in \mathbb{R}^{h \times w \times N}$, where N is the number of expert LUTs. To obtain the SR result of a specific pixel in the frame $X_{i,j,k}$, the weighted LUT used for query is:

$$\hat{LUT}_{X_{i,j,k}} = W_{i,j,1} \times LUT_1 + \ldots + W_{i,j,N} \times LUT_N. \quad (1)$$

It should be noted that the fusion operation is only conducted on the 4D lattice surrounding the input pixel value, not the whole LUT. Moreover, since our weight predictor takes the entire frame as input, the weighted fusion is obtained based on a much larger receptive field, providing more spatial information than previous LUT-based methods [17], [18].

3) Efficient LUT Interpolation: In order to make the proposed Conv-LUT hybrid structure more friendly to parallel computing, we introduce an efficient method to query the mixture of expert LUTs.

As illustrated in Section IV-B, the LUT takes 4 pixel values in a patch as input (4D input).Given the 4D input values, the output values of an anchor pixel $X_{i,j,k}$ are generated by querying and interpolating the nearest sampled points in LUT. Specifically, for the input (x, y, z, u), we first conduct the look-up query operation to find its location in the 4D LUT lattice. As explained in previous LUT works [14], [16], [17], [18], the most significant bits (MSBs) of the input pixel value can be used for LUT location, and the least significant bits (LSBs) are used for interpolation.

ConvLUT uses tetrahedral interpolation [39], which needs only 5 multiplications with the values of 5 bounding vertices of 4-simplex geometry. However, in practice, finding the 5 vertices among total 24 neighboring vertices is implemented with 24 control flow instructions, which is unfriendly to parallel accelerators, resulting in a low inference speed. For example, CUDA operators do not support such flow control operation for parallel acceleration. Hence, we accelerate the tetrahedral interpolation by replacing the complicated control flow with the mapping table query. As shown in Table III, the 24 logical statements (x, y, z, u) in tetrahedral interpolation are equivalent to sorting the order of 4 input pixels from small to large. Since all input permutations are countable, we can use an additional table to store the sorted 4 values (x', y', z', u'). The indexes of 5 corresponding neighboring vertices $(O_1, O_2, O_3, O_4, O_5)$ can also be store according to the 24 control flow instructions in Table III. It should be noted that the relative indexes of O_1 and O_5 are kept fixed at 0000 and 1111, and we only need to store the indexes of (O_2, O_3, O_4). Moreover, only the least significant 4-bits determine the weights $(w_1, w_2, w_3, w_4, w_5)$, our additional table only needs to save 16^4 permutations.

An example of our efficient LUT inference is presented in Fig. 4. For the input (x, y, z, u), we first separate the MSBs (H_x, H_y, H_z, H_u) and the LSBs (L_x, L_y, L_z, L_u). Specifically, we separate the 8bit input pixel values to high 4bit integers (H_x, H_y, H_z, H_u) as:

$$H_x = \left\lfloor \frac{x}{W} \right\rfloor, H_y = \left\lfloor \frac{y}{W} \right\rfloor, H_z = \left\lfloor \frac{z}{W} \right\rfloor, H_u = \left\lfloor \frac{u}{W} \right\rfloor, \quad (2)$$

TABLE II

THE NUMBER OF OPERATIONS OF DIFFERENT INTERPOLATIONS. DUE TO THE COMPLICATED IF-ELSE CONTROL FLOW, TETRAHEDRAL INTERPOLATION IS UNFRIENDLY TO ACCELERATORS LIKE BASIC CUDA OPERATORS. OUR INTERPOLATION NEEDS FEWER OPERATIONS AND CAN BE ACCELERATED BY CUDA

Interpolation	Query	Multiplication	If-else Control Flow	Parallel Accelerator
Tetralinear	16	16	0	✓
Tetrahedral	5	5	24	✗
Ours	9	5	0	✓

TABLE III

THE 24 CONTROL FLOWS OF TETRAHEDRAL INTERPOLATION EQUIVALENT FOR 4D SPACE, ALSO PRESENTED IN SR-LUT [17]. SINCE THE CONTROL FLOWS ARE UNFRIENDLY FOR PARALLEL ACCELERATORS LIKE GPU, WE USE AN ADDITIONAL TABLE TO REPLACE THE IF-ELSE LOGICAL OPERATIONS AND UNIFORMLY CONDUCT THE INTERPOLATION

Condition	w_1	w_2	w_3	w_4	w_5	O_2	O_3	O_4
$L_x>L_y>L_z>L_u$	$W-L_x$	L_x-L_y	L_y-L_z	L_z-L_u	L_u	P_{1000}	P_{1100}	P_{1110}
$L_x>L_y>L_u>L_z$	$W-L_x$	L_x-L_y	L_y-L_u	L_u-L_z	L_z	P_{1000}	P_{1100}	P_{1101}
$L_x>L_u>L_y>L_z$	$W-L_x$	L_x-L_u	L_u-L_y	L_y-L_z	L_z	P_{1000}	P_{1001}	P_{1101}
$L_u>L_x>L_y>L_z$	$W-L_u$	L_u-L_x	L_x-L_y	L_y-L_z	L_z	P_{0001}	P_{1001}	P_{1101}
$L_x>L_z>L_y>L_u$	$W-L_x$	L_x-L_z	L_z-L_y	L_y-L_u	L_u	P_{1000}	P_{1010}	P_{1110}
$L_x>L_z>L_u>L_y$	$W-L_x$	L_x-L_z	L_z-L_u	L_u-L_y	L_y	P_{1000}	P_{1010}	P_{1011}
$L_x>L_u>L_z>L_y$	$W-L_x$	L_x-L_u	L_u-L_z	L_z-L_y	L_y	P_{1000}	P_{1001}	P_{1011}
$L_u>L_x>L_z>L_y$	$W-L_u$	L_u-L_x	L_x-L_z	L_z-L_y	L_y	P_{0001}	P_{1001}	P_{1011}
$L_z>L_x>L_y>L_u$	$W-L_z$	L_z-L_x	L_x-L_y	L_y-L_u	L_u	P_{0010}	P_{1010}	P_{1110}
$L_z>L_x>L_u>L_y$	$W-L_z$	L_z-L_x	L_x-L_u	L_u-L_y	L_y	P_{0010}	P_{1010}	P_{1011}
$L_z>L_u>L_x>L_y$	$W-L_z$	L_z-L_u	L_u-L_x	L_x-L_y	L_y	P_{0010}	P_{0011}	P_{1011}
$L_u>L_z>L_x>L_y$	$W-L_u$	L_u-L_z	L_z-L_x	L_x-L_y	L_y	P_{0001}	P_{0011}	P_{1011}
$L_y>L_x>L_z>L_u$	$W-L_y$	L_y-L_x	L_x-L_z	L_z-L_u	L_u	P_{0100}	P_{1100}	P_{1110}
$L_y>L_x>L_u>L_z$	$W-L_y$	L_y-L_x	L_x-L_u	L_u-L_z	L_z	P_{0100}	P_{1100}	P_{1101}
$L_y>L_u>L_x>L_z$	$W-L_y$	L_y-L_u	L_u-L_x	L_x-L_z	L_z	P_{0100}	P_{0101}	P_{1101}
$L_u>L_y>L_x>L_z$	$W-L_u$	L_u-L_y	L_y-L_x	L_x-L_z	L_z	P_{0001}	P_{0101}	P_{1101}
$L_y>L_z>L_x>L_u$	$W-L_y$	L_y-L_z	L_z-L_x	L_x-L_u	L_u	P_{0100}	P_{0110}	P_{1110}
$L_y>L_z>L_u>L_x$	$W-L_y$	L_y-L_z	L_z-L_u	L_u-L_x	L_x	P_{0100}	P_{0110}	P_{0111}
$L_y>L_u>L_z>L_x$	$W-L_y$	L_y-L_u	L_u-L_z	L_z-L_x	L_x	P_{0100}	P_{0101}	P_{0111}
$L_u>L_y>L_z>L_x$	$W-L_u$	L_u-L_y	L_y-L_z	L_z-L_x	L_x	P_{0001}	P_{0101}	P_{0111}
$L_z>L_y>L_x>L_u$	$W-L_z$	L_z-L_y	L_y-L_x	L_x-L_u	L_u	P_{0010}	P_{0110}	P_{1110}
$L_z>L_y>L_u>L_x$	$W-L_z$	L_z-L_y	L_y-L_u	L_u-L_x	L_x	P_{0010}	P_{0110}	P_{0111}
$L_z>L_u>L_y>L_x$	$W-L_z$	L_z-L_u	L_u-L_y	L_y-L_x	L_x	P_{0010}	P_{0011}	P_{0111}
else	$W-L_u$	L_u-L_z	L_z-L_y	L_y-L_x	L_x	P_{0001}	P_{0011}	P_{0111}

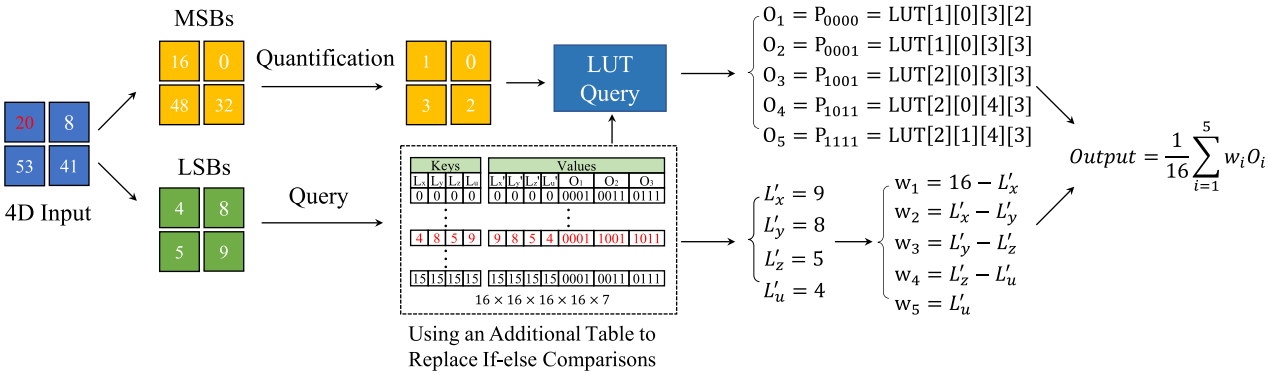


Fig. 4. An example of the implementation of our efficient LUT inference. Here, the quantization value is set as 16 to compress LUT.

and low 4bit decimals (L_x, L_y, L_z, L_u) as:

$$L_x = x - H_x \times W, L_y = y - H_y \times W,$$
$$L_z = z - H_z \times W, L_u = u - H_u \times W, \quad (3)$$

where $\lfloor \cdot \rfloor$ is the floor function, and W represents the quantization value to compress the LUT, which is set as 16 in our paper. Then, we query the pre-defined additional table to get the sorted LSBs (L'_x, L'_y, L'_z, L'_u) and the binary indexes of (O_2, O_3, O_4). When we get the sorted LSBs (L'_x, L'_y, L'_z, L'_u), the calculation of interpolation weights $(w_1, w_2, w_3, w_4, w_5)$ dose not need the if-else judgement in Table III, but can be uniformly defined as:

$$w_1 = W - L'_x, w_2 = L'_x - L'_y, w_3 = L'_y - L'_z,$$
$$w_4 = L'_z - L'_u, w_5 = L'_u. \quad (4)$$

When we get the binary indexes of (O_2, O_3, O_4), the values of $(O_1, O_2, O_3, O_4, O_5)$ can be accessed by using their binary indexes and MSBs (H_x, H_y, H_z, H_u) to conduct LUT query. For example, if the index of O_2 is 0001, the value of O_2 is $P_{0001} = LUT[H_x][H_y][H_z][H_u + 1]$. Finally, the output of weighted interpolation are calculated as:

$$Output = \frac{1}{W} \sum_{i=1}^{5} w_i * O_i. \quad (5)$$

During the LUT interpolation, the weighted combination of N LUT bases can also be conducted in parallel.

Since only the sorted LSBs determine the interpolation weights, the size of the mapping table can be efficiently compressed to [16, 16, 16, 16, 8] by only storing 4-bit interpolation weights $(w_1, w_2, w_3, w_4, w_5)$ instead of 256^4 8-bit 4D-pixel permutations. Moreover, the relative indexes of O_1 and O_5 are kept fixed at 0000 and 1111, and we only need to store the index of (O_2, O_3, O_4). Once the order table is pre-defined, the comparison and flow control operation can be replaced by query, and the LUT inference can be accelerated in parallel.

Therefore, our accelerated LUT inference can be defined as three steps: (1) For the input (x, y, z, u), we first separate the MSBs (h_x, h_y, h_z, h_u) and LSBs (l_x, l_y, l_z, l_u); (2) we query the pre-defined order table to get the interpolation weights $(w_1, w_2, w_3, w_4, w_5)$ and the binary index of (O_1, O_2, O_3); (3) we conduct the unified tetrahedral interpolation with the additional order table. As shown in Table II, the number of operations of our accelerated interpolation is smaller and can be easily deployed to accelerators. In our work, we use CUDA accelerator to conduct parallel computation.

C. Temporal Branch for Multi-Frame Processing

To fully utilize the temporal information, our proposed model contains a temporal branch responsible for refining the result of the LUT branch with history frames and object motions. While the LUT branch utilizes a look-up table to handle spatial degradation, the temporal branch uses convolutional networks with better computational capacity to handle more complicated temporal and motion related information.

To achieve fast speed and low latency for online streaming, commonly used optical flow alignment, deformable convolution and transformer [2], [5], [8], [9] are not suitable for our task. Moreover, due to the latency restriction of online VSR, the future frames cannot be utilized.

As shown in Fig. 3(b), our temporal branch only uses 4 convolutional layers with LeakyReLU [38] to fuse the previous and current frames. To incorporate motion-related information, we leverage a readily available video streaming prior: motion vectors. Similar to optical flow, these vectors provide a coarse approximation of patch-level correspondence and alignment between two frames. However, unlike optical flow, they require no extra calculation since they are part of the streaming system's prior knowledge. To avoid additional computational cost, we simply add the motion vector between two frames as an additional feature map.

V. Experiment

A. Experimental Setting

1) Datasets: The experiments are conducted on the proposed real-world online streaming VSR dataset, LDV-WebRTC. We focus on the scale factor $r = 4$. To assess the model ability to deal with degradations under different bandwidths, we use the LR-HR pairs under 1Mbps for training, and evaluate SR models on 100Kbps, 500Kbps, and 1Mbps testsets respectively.

2) Evaluation Metrics: We evaluate the SR performance for online streaming from three perspectives: the number of model

TABLE IV

×4 SR MODEL COMPARISONS ON LDV-WEBRTC TESTSETS UNDER 100KBPS, 500KBPS, AND 1MBPS. THE LATENCY LEVELS REQUIRED BY DIFFERENT METHODS ARE SORTED FROM HIGH TO LOW. SIZE DENOTES THE STORAGE SPACE OR THE PARAMETER NUMBER OF EACH MODEL. THE ROW HIGHLIGHTED IN GRAY MEANS THE SR METHOD HAS UNBEARABLE HIGH LATENCY, AND THUS CANNOT BE APPLIED FOR ONLINE STREAMING. FOR ONLINE PRACTICAL SR METHODS, BEST AND SECOND BEST RESULTS ARE HIGHLIGHTED IN RED AND BLUE. RUNTIME IS MEASURED WITH 2080TI GPU FOR GENERATING 1280 × 720 RESULTS. *: THE LUT-BASED METHODS ARE ACCELERATED BY OUR INTERPOLATION. †: THE STORAGE SPACE OF THE LUT-BASED METHOD

Latency Level	Model	100Kbps PSNR / SSIM	500Kbps PSNR / SSIM	1Mbps PSNR / SSIM	Size	Runtime	FPS	Video (30FPS)	Gaming (60FPS)
High	BasicVSR++ [9]	23.95 / 0.6210	24.70 / 0.6754	25.11 / 0.6997	9.54M	418.5ms	2.34	✗	✗
	TTVSR [8]	23.87 / 0.6246	24.73 / 0.6742	25.24 / 0.7012	6.72M	244.3ms	4.09		
	RCAN [6]	23.80 / 0.6165	24.64 / 0.6729	24.85 / 0.6960	15.6M	205.4ms	4.89		
	SRResNet [4]	23.76 / 0.6141	24.53 / 0.6663	24.62 / 0.6878	1.52M	75.80ms	13.19		
Middle	RRN [2]	23.48 / 0.6121	24.48 / 0.6521	24.84 / 0.6811	3.36M	27.00ms	37.03	✓	✗
	CARN [5]	23.77 / 0.6137	24.50 / 0.6565	24.74 / 0.6927	1.59M	25.30ms	39.53		
	VESPCN [7]	23.25 / 0.6063	24.34 / 0.6427	24.50 / 0.6774	0.88M	22.84ms	43.78		
	MuLUT* [19]	23.53 / 0.6102	24.44 / 0.6474	24.69 / 0.6795	8.16MB†	20.44ms	48.92		
	SPLUT* [18]	23.34 / 0.6097	24.41 / 0.6333	24.55 / 0.6764	18.12MB†	21.81ms	45.85		
Low	BI	23.31 / 0.6031	24.27 / 0.6274	24.35 / 0.6724	-	-	-	✓	✓
	PAN [40]	23.55 / 0.6123	24.50 / 0.6332	24.66 / 0.6742	0.27M	16.12ms	62.50		
	SR-LUT* [17]	23.45 / 0.5901	24.32 / 0.6198	24.42 / 0.6451	5.37MB†	11.52ms	86.81		
	ConvLUT-CARN*	23.73 / 0.6123	24.46 / 0.6540	24.80 / 0.6803	12.75MB†	10.15ms	98.52		
	ConvLUT-SRResNet*	23.78 / 0.6140	24.54 / 0.6558	24.87 / 0.6822					
	ConvLUT-RCAN*	23.82 / 0.6163	24.52 / 0.6576	24.84 / 0.6816					

parameters, runtime, and the distortion quality of the generated results. Specifically, Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) [24] are adopted for evaluation. To compare the running speed, we measure and report the runtime of super-resolving 320 × 180 LR images on one NVIDIA RTX 2080Ti GPU.

3) Implementation Details: As explained in Sec. IV-B.1, we uniformly select 6 QP values ($QP = 0, 10, 20, 30, 40, 50$) and use these QP values to encode 6 degraded video subsets $\{D_1, \ldots, D_6\}$. Three state-of-the-art SR models, SRResNet [4], CARN [5], and RCAN [6] are trained on the 6 datasets and transferred to 3 groups of expert LUTs. Finally, ConvLUT is trained with the expert LUTs, resulting in three models, denoted as ConvLUT-SRResNet, ConvLUT-CARN, and ConvLUT-RCAN.

The number of channels of spatial and temporal branches is set to 64. The number of output channels of pixel-level weight predictor is set to 6, matching the number of expert LUTs. In training configurations, the image patch is randomly cropped with the size of 48 × 48, and the batch size is set to 16. The whole ConvLUT is jointly trained by imposing Charbonnier loss on the final SR outputs. We use Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ to update model parameters. The initial learning rate is 10^{-4}. We conducted the model training with NVIDIA Tesla V100 GPUs.

B. Experiments on LDV-WebRTC Dataset

We compare our method with various SOTAs. Among these methods, SRResNet [4], CARN [5], and RCAN [6] are the base networks for expert LUT construction. PAN [40] is a widely used single-image SR structure with low model complexity and fast inference speed. We also compare with two fast VSR methods VESPCN [7] and RRN [2]. Due to low latency requirement in online streaming scenario, it is infeasible to cache future frames for VSR models. As a result, we modify VESPCN by removing the next frame branch and only extract the spatial-temporal information with the previous and current frames. BasicVSR++ [9] and TTVSR [8] are much larger bi-directional VSR models. Although it is infeasible to apply them to online scenarios, we still involve them as the performance upper bound. Moreover, we further compare two LUT-based SISR models, SR-LUT [17] and SPLUT [18]. Note that the GPU inference speed of the current SR-LUT implementation is quite slow due to the large portion of if-else control flow operations and serial for-loop processing for each pixel. For a fair comparison, we also accelerate those LUT-based models with our efficient interpolation method in parallel, as explained in Sec. IV-B.3. We used the open-source codes provided by the authors to implement the compared methods. All methods use the same train-test set partition.

All the evaluation results are reported in Table IV. Due to the strict requirement of latency for online streaming, we further measure runtime and FPS to evaluate the model efficiency. Based on FPS, We categorize the compared SR methods into three types. We categorize methods with FPS lower than 30 as High Latency methods, which are hard to support online streaming applications. Low Latency group refers to methods with FPS higher than 60, fast enough to support high-frame-rate gaming. The rest methods in the range of 30 FPS and 60 FPS are marked as Middle Latency. Benefited from spatial-temporal feature extraction and deep network structure, VSR methods BasicVSR++ [9] and TTVSR [8], as well as SISR models RCAN [6] and SRResNet [4] achieve high PSNR/SSIM performance, in exchange of very high latency impractical to online streaming. Compared with methods in the Middle Latency group, our model outperforms them with a marginal improvement in terms of PSNR and SSIM, and in the meantime achieves a much better latency and FPS. In low latency scenarios, our method can significantly

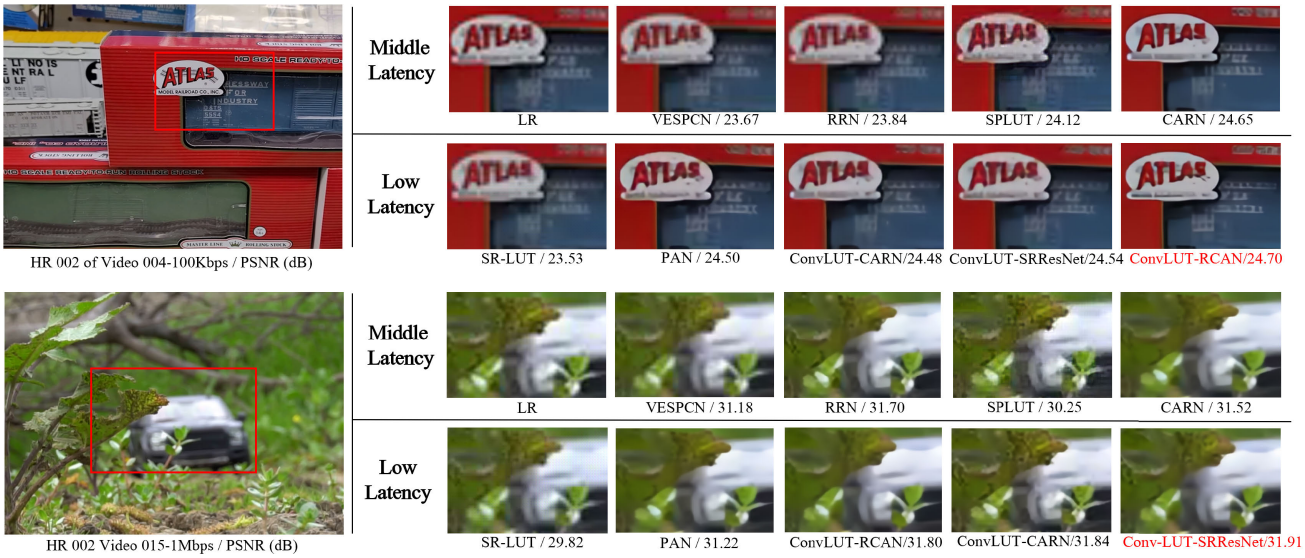


Fig. 5. SR perceptual results (×4) of images selected from 100Kbps and 1Mbps testsets. Best results are highlighted in red.

outperform other lightweight models such as PAN [40] and LUT-based SR-LUT [17] in terms of PSNR/SSIM values and meanwhile achieves the lowest latency. The comparisons between our models using 3 types of expert LUTs can also give some interesting findings. When the quality of LR frames decreases (*i.e.*100Kbps), the expert LUTs transferred from better SR structures (*i.e.*RCAN) produce better results. Moreover, by comparing model sizes, we can see that our method only brings a linear increase in storage costs because of multiple expert LUTs. We believe the model size of our ConvLUT is acceptable for current devices. All those results verify the effectiveness of our ConvLUT in the online streaming scenario.

Visual comparisons are shown in Fig. 5. Here, we choose SR methods relatively practical for streaming in the Middle and Low Latency groups for comparisons. It can be seen that previous LUT-based methods, including SR-LUT [17] and SPLUT [18], fail to present natural details and produce more artifacts like blocking effect. Limited by fewer conventional layers and network parameters, the lightweight CNN-based SR models, such as PAN [40] and VESPCN [7], generate results with fewer high-frequency details. And our models produce fewer artifacts than previous LUT-based methods and present a similar level of sharpness to CARN [5] and RRN [2] at faster speed. In addition to the detailed comparison in Fig. 5, we also demonstrate the advantages of our approach over some typical rapid super-resolution networks with more samples in Fig. 6.

C. Experiments on Static Degradation

The expert LUTs of our method are determined by the structure of SISR model and the corresponding training degradations of those SR models. Here, we evaluate the effectiveness of the fusion of expert LUTs with different SR network structures, which are trained with the same static degradation. Specifically, we follow the NTIRE 2022 challenge [13] to conduct experiments on the static $QP = 37$ compression degradation. Three SISR model SRResNet [4], CARN [5], and RCAN [6] are used as the base networks for LUT construction, and our model is denoted as ConvLUT-RCAN+SRResNet+CARN. All those 3 SR networks are trained with the same training set and three corresponding transferred LUTs are grouped as the bases for LUT fusion. And we follow the NTIRE 2022 challenge [13] to train and test SR models on LDV 2.0 dataset.

TABLE V

ABLATION STUDIES OF THE COMPONENTS OF CONVLUT ON 1MBPS TESTSET. THE EXPERT LUTS ARE TRANSFERRED FROM RCAN [6]. EACH COMPONENT BRINGS IMPROVEMENTS IN TERMS OF PSNR

	(A)	(B)	(C)	(D)	**Our**
Single-LUT	✓				
Multi-LUTs Adaptive Fusion		✓		✓	✓
Previous Frame Information			✓	✓	✓
Motion Vector information					✓
Size (MB)	5.37	12.62	0.12	12.74	12.75
Runtime (ms)	2.82	4.00	8.20	10.13	10.15
PSNR (dB)	24.21	24.47	24.65	24.80	**24.84**

The SOTA comparisons are presented in Table VI. Except the 3 mentioned SR models above, we further add 3 novel methods presented in NTIRE 2022 Report [13]. Although cannot outperform the deep VSR methods, such as the 1st winner model GY-Lab, our model still outperforms both LUT-based models and three SR base networks. Those results proves that the combination of LUT bases can also efficiently fuse the capabilities of different SR structures. In practice, the LUT fusion can be considered in both the training degradation and the SR structure.

D. Experiments of GPU Memory Occupancy

Table VII shows the average GPU memory occupancy of different methods. All the gpu memory consumption is calculated when processing video in frame-to-frame mode, except for BasicVSR++. Since BasicVSR++ model is a

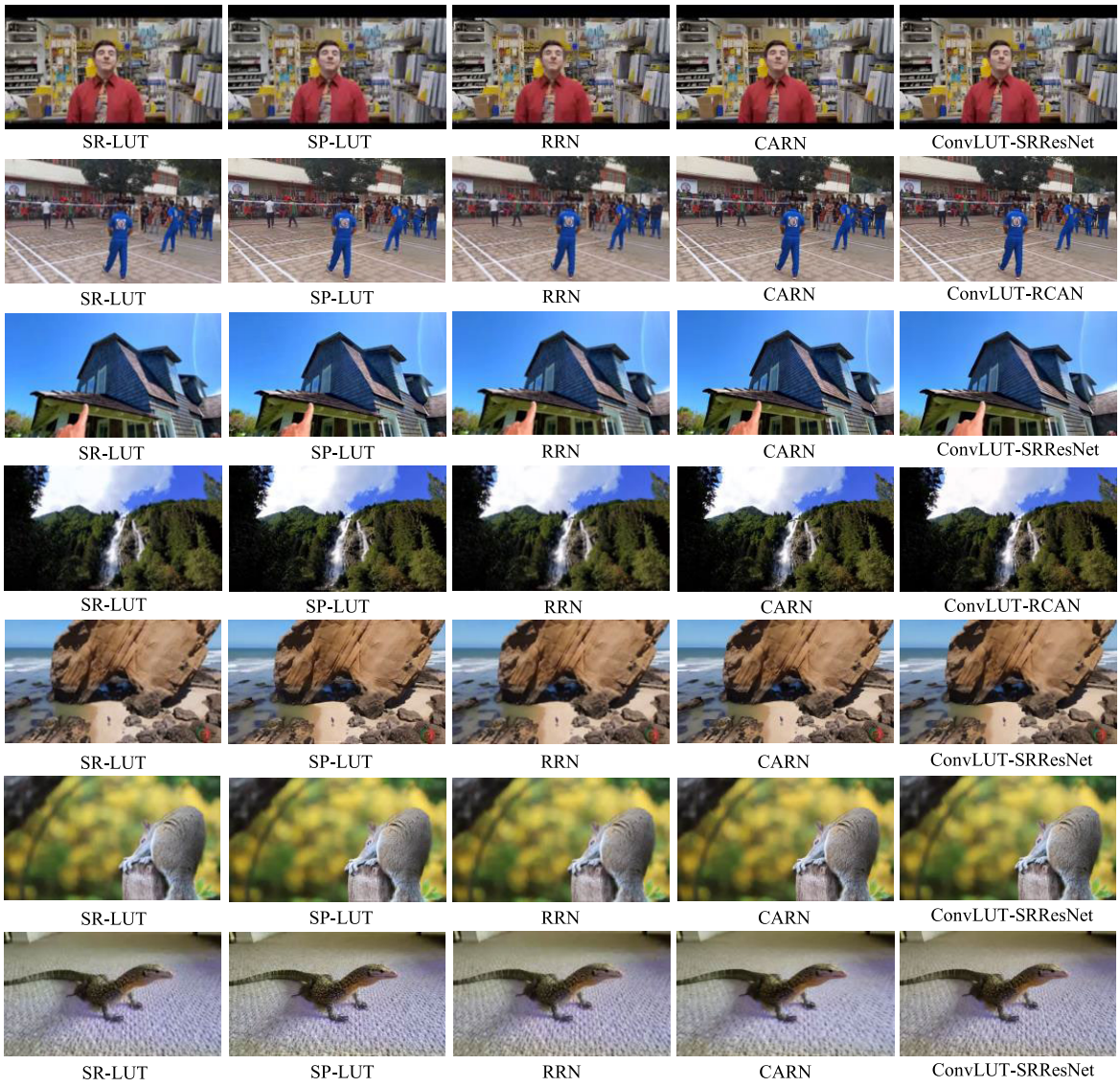


Fig. 6. More qualitative comparisons of SR-LUT [17], SP-LUT [18], RRN [2], and CARN [5] on our LDV-WebRTC testset. The frames listed are from video 007, 013, 010, 011, 012, 001 and 002.

TABLE VI

×4 SR MODEL COMPARISONS OF NTIRE 2022 CHALLENGE TRACK3 ON SUPER-RESOLUTION AND QUALITY ENHANCEMENT OF COMPRESSED VIDEO. THE RESULTS FROM THE 1ST ROW TO 3RD ROW ARE DIRECTLY EXTRACTED FROM NTIRE 2022 REPORT [13]. *: THE LUT-BASED METHODS ARE ACCELERATED BY OUR INTERPOLATION

Model	PSNR	Runtime	Hardware
GY-Lab (BasicVSR++ [9] based) 1st [13]	24.23	11.5s	V100
AVRT (VSRTransformer [41] based) 9th [13]	23.52	2s	A100
Modern_SR (EDVR [42] based) 12th [13]	23.03	860ms	3080
RCAN [6]	22.96	205.4ms	2080ti
SRResNet [4]	22.93	75.80ms	2080ti
CARN [5]	22.89	25.30ms	2080ti
SP-LUT* [18]	22.87	21.81ms	2080ti
SR-LUT* [17]	22.61	11.52ms	2080ti
ConvLUT-RCAN+SRResNet+CARN*	23.32	10.15ms	2080ti

recurrent video SR approach, which needs the complete video to process each frame. The memory of the BasicVSR++ is obtained from the statistics of a 100-frames video.

Since we have made improvements to the LUT interpolation method, ConvLUT consumes more GPU memory compared to some SISR methods [4], [5] for it uses more intermediate variables during inference. However, the memory consumption of 60-70MB is much smaller than the upper memory of current mainstream GPUs, so we consider that the memory consumption of ConvLUT is acceptable. The recurrent video SR methods [2], [9] consume large amounts of GPU memory, which is more demanding on the GPU device and not suitable for online environments.

E. Ablation Analysis

1) Effectiveness of Different Modules: Table V shows the effectiveness of different modules in ConvLUT. The baseline model (A), where only one LUT is applied, is quite similar to SR-LUT [17], and it produces poor PSNR result, which proves the small receptive field and single neural networks cannot handle the dynamic degradations of online streaming. By adding adaptive LUT fusion, the PSNR has increased

TABLE VII
GPU MEMORY OCCUPANCY COMPARISONS OF CONVLUT WITH DIFFERENT METHODS. THE MEMORY IS MEASURED WITH 2080TI GPU FOR GENERATING 1280 × 720 RESULTS IN A FRAME-TO-FRAME MANNER

Method	GPU Memory	PSNR
SRResNet [4]	27.67 MB	24.62
RCAN [6]	81.48 MB	24.85
BasicVSR++ [9]	15.6 GB(100 Frames)	25.11
CARN [5]	27.84 MB	24.74
SR-LUT [17]	67.14 MB	24.42
RRN [2]	447.51 MB	24.84
ConvLUT(Ours)	68.05 MB	24.84

TABLE VIII
PERFORMANCE COMPARISON OF CONVLUT WITH DIFFERENT EXPERT LUT NUMBERS

LUT Number	Storage	Runtime	1Mbps PSNR / SSIM
1	6.42 MB	8.63 ms	24.21 / 0.6231
3	8.94 MB	9.29 ms	24.60 / 0.6694
6	12.75 MB	10.15 ms	24.84 / **0.6816**
12	20.37 MB	11.85 ms	**24.89** / 0.6813

TABLE IX
PERFORMANCE COMPARISON OF CONVLUT WITH SINGLE MODEL LUTS AND MULTI MODELS LUTS ON 1MBPS LDV-WEBRTC TESTSETS

Model	Storage	Runtime	PSNR/SSIM
ConvLUT-RCAN	12.75 MB	10.15ms	24.84/0.6816
ConvLUT-CARN			24.80/0.6803
ConvLUT-SRResNet			24.87/0.6822
ConvLUT-18LUTs	27.93 MB	14.37ms	24.87/0.6821

by 0.26dB, showing that the pixel-level weight predictor effectively fuses the expert LUTs to adaptively handle dynamic degradations. The weight predictor cost more storage and computation, but the 12.62MB increase should be acceptable in practice.

Video SR task has more temporal information compared to single image SR task, which can be extracted and utilized to efficiently recover the degraded details of the video. When retaining only the temporal branch, the shallow convolutional layers predicted the super-resolution results by both the video spatial and temporal information and achieved the result of 24.65 dB. When co-optimizing the two branches, since the multi-LUTs part has fully utilized the spatial information in the video to process the SR results, then the temporal branch can concentrate more on the temporal information of the video and utilize it to complement the results after the LUT fusion. When both branches are used simultaneously, the performance of the network is improved by 0.15 dB compared to utilizing only the temporal branch, which proves that the spatial and temporal branches can promote each other to fully utilize the spatial and temporal information in the video SR task.

To better extract the temporal information, the reference frame needs to be spatially aligned. The commonly used temporal fusion modules such as optical flow estimation or deformable convolution are too slow to be applied to online streaming, and hence we adopt the motion vector information to efficiently and effectly process the reference frame and obtain 0.04 dB performance improvement.

2) Runtime of Different Modules: Table V also shows the runtime of different modules in ConvLUT. When only using the baseline model, we obtain the SR results by querying a single table, and our CUDA acceleration of the interpolation process enables the runtime of the table query to be only 2.82ms, but it has performance degradation compared to the BI interpolation results due to the limited receptive field and information storage capacity of a single table. When multiple LUTs are used, the network can efficiently handle multiple video degradations, but the runtime increases 1.18ms due to the multiple query process.

With the introduction of temporal branch, the network performs an additional 4 layers of convolutional computation, which increases the runtime by 6.14ms but brings 0.33dB performance improvement to the network. While temporal branch increases the runtime substantially, this branch fully utilizes the temporal information of the video SR task, and meanwhile it has a mutually promoting effect with the spatial branch, which greatly increases the performance of the network. The introduction of the temporal branch is one of the reasons why ConvLUT achieves a significant performance improvement over SR-LUT [17] with less inference time. The Motion Vector comes from the codec information obtained during video decoding, which does not increase the network's inference time, but can solve the network's problem of misalignment of the previous and subsequent frames in the video, thus slightly improving the performance of the temporal branch.

3) Configurations of Expert LUT: The Expert LUT of ConvLUT can be constructed from different SR networks. The last 3 rows of Table IV show how Expert LUT transferred from different SR networks affect the performance of ConvLUT, where better performed SISR network can help the corresponding Expert LUT to get better results in severe degradations. We also analyze how the number of Expert LUTs affects the performance of ConvLUT. We evenly selected N QP values from the range of 0 to 50, where the corresponding videos are used to construct N expert LUTs. As shown in Table VIII, increasing the number of expert LUTs constantly improves the SR performance, but more LUTs result in more storage and computational overhead. When the number of LUTs is larger than 6, our model only has a minor improvement, and hence achieves the best trade-off between SR performance and efficiency.

Besides generating multiple LUTs for a single model, we also attempted to fusion the experts LUTs of the 3 models, as shown in Table IX. Specifically, in ConvLUT-18LUTs we take all 18 LUTs of the 3 models as experts LUTs and generate an 18-channel weights as pixel-level weights for each LUT with spatial branch. The performance improvement achieved by the obtained model is very marginal, and the obtained images are similar in quality to those generated by ConvSRResNet. This is due to computing the weights of 18 LUTs is overly difficult for the spatial branch with only 4 layers of convolution layers, so the model does not significantly improve with multi-model LUT fusion. However, the storage and runtime metrics of the model have increased substantially

TABLE X

RUNTIME OF LUT INTERPOLATION INFERENCE. AFTER USING OUR ACCELERATED TETRAHEDRAL INTERPOLATION, THE PARALLEL ACCELERATION CAN BE APPLIED WITHOUT IF-ELSE CONTROL FLOW INSTRUCTIONS. THE INFERENCE SPEEDS OF BOTH OUR MODEL AND SR-LUT GET SIGNIFICANTLY IMPROVED

Model	Our Acceleration	If-else Control Flow	Runtime
SR-LUT [17]	✗	✓	381.89 ms
	✓	✗	**11.52 ms**
ConvLUT	✗	✓	516.21 ms
	✓	✗	**10.15 ms**

TABLE XI

COMPARISON OF OUR CONVLUT-RCAN WITH DIFFERENT SAMPLING INTERVAL SIZES. WE SET THE SAMPLING INTERVAL SIZE AS 16 FOR OUR MODEL TO REDUCE THE LUT SIZE, MINIMIZING THE DROP OF THE ORIGINAL PERFORMANCE

Sampling	LUT Storage	1Mbps PSNR / SSIM
2^0 (Full LUT)	384 GB	24.91 / 0.6875
2^2	1632 MB	24.90 / 0.6871
2^3	108 MB	24.87 / 0.6843
2^4 (Our)	7.644 MB	24.84 / 0.6816
2^5	612 KB	24.61 / 0.6740
2^6	59.35 KB	24.34 / 0.6538
2^8	2.304 KB	23.15 / 0.6459

TABLE XII

THE PERFORMANCE UPPER BOUND OF THE THREE MODELS IN OUR CONVLUT FRAMEWORK. UPPER BOUND MEANS DIRECTLY USING THE PRE-TRAINED NETWORK INSTEAD OF THE LUT IN CONVLUT

Model	100Kbps		500Kbps		1Mbps	
	ConvLUT	Upper Bound	ConvLUT	Upper Bound	ConvLUT	Upper Bound
CARN	23.73/0.6123	23.81/0.6127	24.46/0.6540	24.62/0.6683	24.80/0.6803	24.80/0.6955
SRResNet	23.78/0.6140	23.83/0.6177	24.54/0.6558	24.59/0.6700	24.87/0.6822	24.88/0.6964
RCAN	23.82/0.6163	23.84/0.6211	24.52/0.6576	24.65/0.6730	24.84/0.6816	24.97/0.6998

when using more LUTs, so we can obtain the same conclusion as in Table VIII, that the optimal number of LUTs in our model is 6.

4) Effectiveness of Interpolation Acceleration: As shown in Table X, with the proposed the efficient interpolation method, both SR-LUT [17] and the proposed ConvLUT achieve more than 35 times inference acceleration on GPU devices. Since ConvLUT only needs to query one fused LUT while SR-LUT needs to repeatedly query one LUT for 4 times, our methods outperform SR-LUT in terms of runtime after acceleration.

5) Analysis of LUT Sampling: For LUT-based SR methods [17], [18], the original LUT is commonly sampled with a quantization value to compress the size of LUTs. For our method, we also uniformly sample the LUT. In table XI, we present the comparisons of our ConvLUT models with different quantization values. The uncompressed LUT bases (2^0) produces the best results but have unbearable storage (384GB). When the sampling size increases from 2^2 to 2^4, the size of LUT significantly decreases from 1632MB to 7.644 MB while getting acceptable performance drop. Therefore, we choose the quantization value 16 as our default setting. If the LUT size matters, sampling sizes 2^5 and 2^6 could also be considered. For practical implementation, the sampling size should be considered as the tradeoff between the storage cost and the performance.

6) The Performance Upper Bound of ConvLUT: As shown in Table XII, we test the upper bound performance of the three SR models in ConvLUT framework. Specifically, after pre-training the model on the six QPs compressed video, instead of converting it to a LUT, we directly weight the results generated by the six networks to obtain the output of the spatial branch. It can be concluded that all three models have experienced a certain performance degradation in the ConvLUT framework due to the great reduction of the receptive field by storing and replacing the network with the LUT structure. To further minimize the impact of this degradation on the network is one of the feasible directions to continue to improve the SR capability of the ConvLUT framework, which we will explore in our future work.

VI. CONCLUSION

Online video streaming presents unique challenges for super-resolution due to dynamically changing degradations and strict latency requirements. This paper addresses this problem with a new benchmark dataset, LDV-WebRTC, produced with real-world online streaming system. A novel hybrid network that combines convolution and Look-Up Table (LUT) is proposed to achieve a better performance-latency tradeoff. Our proposed mixture-of-expert-LUT module builds a set of LUTs specialized in different degradations and adaptively combines them to handle changing degradations. Experiment results show that our method achieves 720P video SR at around 100 FPS, outperforming existing LUT-based methods and offering competitive performance compared to efficient CNN-based methods.

ACKNOWLEDGMENT

A preprint version of this research work was put on arXiv. They declare that their manuscript is original and has not been previously published. It is not currently being considered for publication elsewhere.

REFERENCES

[1] Z. Lu, H. Xia, S. Heo, and D. Wigdor, "You watch, you give, and you engage: A study of live streaming practices in China," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, Apr. 2018, pp. 1–13.

[2] T. Isobe, F. Zhu, X. Jia, and S. Wang, "Revisiting temporal modeling for video super-resolution," 2020, *arXiv:2008.05765*.

[3] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, Feb. 2015.

[4] C. Ledig et al., "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4681–4690.

[5] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 252–268.

[6] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 286–301.

[7] J. Caballero et al., "Real-time video super-resolution with spatio-temporal networks and motion compensation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 4778–4787.

[8] C. Liu, H. Yang, J. Fu, and X. Qian, "Learning trajectory-aware transformer for video super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 5687–5696.

[9] K. C. K. Chan, S. Zhou, X. Xu, and C. C. Loy, "BasicVSR++: Improving video super-resolution with enhanced propagation and alignment," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 5972–5981.

[10] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo, "Learning texture transformer network for image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5791–5800.

[11] Z. Qiu et al., "Learning spatiotemporal frequency-transformer for compressed video super-resolution," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 257–273.

[12] A. B. Johnston and D. C. Burnett, *WebRTC: APIs and RTCWEB Protocols of the HTML5 Real-Time Web*. Lilburn, GA, USA: Digital Codex LLC, 2012.

[13] R. Yang et al., "NTIRE 2022 challenge on super-resolution and quality enhancement of compressed video: Dataset, methods and results," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2022, pp. 1220–1237.

[14] H. Zeng, J. Cai, L. Li, Z. Cao, and L. Zhang, "Learning image-adaptive 3D lookup tables for high performance photo enhancement in real-time," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 4, pp. 2058–2073, Apr. 2022.

[15] T. Wang et al., "Real-time image enhancer via learnable spatial-aware 3D lookup tables," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2021, pp. 2471–2480.

[16] C. Liu, H. Yang, J. Fu, and X. Qian, "4D LUT: Learnable context-aware 4D lookup table for image enhancement," 2022, *arXiv:2209.01749*.

[17] Y. Jo and S. Joo Kim, "Practical single-image super-resolution using look-up table," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 691–700.

[18] C. Ma, J. Zhang, J. Zhou, and J. Lu, "Learning series-parallel lookup tables for efficient image super-resolution," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 305–321.

[19] J. Li, C. Chen, Z. Cheng, and Z. Xiong, "MuLUT: Cooperating multiple look-up tables for efficient image super-resolution," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 238–256.

[20] F. Zhang, H. Zeng, T. Zhang, and L. Zhang, "CLUT-Net: Learning adaptively compressed representations of 3DLUTs for lightweight image enhancement," in *Proc. 30th ACM Int. Conf. Multimedia*, Oct. 2022, pp. 6493–6501.

[21] C. Nvidia, "CUDA toolkit documentation," NVIDIA Corp. (NVIDIA), Santa Clara, CA, USA, Tech. Rep. 12.4, 2018.

[22] Z. Wang, B. He, W. Zhang, and S. Jiang, "A performance analysis framework for optimizing OpenCL applications on FPGAs," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Mar. 2016, pp. 114–125.

[23] F. Khorasani, R. Gupta, and L. N. Bhuyan, "Efficient warp execution in presence of divergence with collaborative context collection," in *Proc. 48th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Dec. 2015, pp. 204–215.

[24] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[25] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[26] H. Yeo, C. J. Chong, Y. Jung, J. Ye, and D. Han, "NEMO: Enabling neural-enhanced video streaming on commodity mobile devices," in *Proc. 26th Annu. Int. Conf. Mobile Comput. Netw.*, 2020, pp. 1–14.

[27] P. Chen, W. Yang, L. Sun, and S. Wang, "When bitstream prior meets deep prior: Compressed video super-resolution with learning from decoding," in *Proc. 28th ACM Int. Conf. Multimedia (ACM MM)*, Oct. 2020, pp. 1000–1008.

[28] K. C. K. Chan, X. Wang, K. Yu, C. Dong, and C. C. Loy, "Understanding deformable alignment in video super-resolution," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 973–981.

[29] K. C. K. Chan, X. Wang, K. Yu, C. Dong, and C. C. Loy, "BasicVSR: The search for essential components in video super-resolution and beyond," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 4947–4956.

[30] C. Yang, M. Jin, X. Jia, Y. Xu, and Y. Chen, "AdaInt: Learning adaptive intervals for 3D lookup tables on real-time image enhancement," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jul. 2022, pp. 17522–17531.

[31] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *Int. J. Comput. Vis.*, vol. 127, no. 8, pp. 1106–1125, Aug. 2019.

[32] X. Yang, W. Xiang, H. Zeng, and L. Zhang, "Real-world video super-resolution: A benchmark dataset and a decomposition based learning scheme," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 4761–4770.

[33] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H. 264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[34] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Comput.*, vol. 3, no. 1, pp. 79–87, 1991.

[35] N. Shazeer et al., "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," 2017, *arXiv:1701.06538*.

[36] M. Emad, M. Peemen, and H. Corporaal, "MoESR: Blind super-resolution using kernel-aware mixture of experts," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2022, pp. 4009–4018.

[37] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," 2016, *arXiv:1607.08022*.

[38] A. L. Maas et al., "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, Atlanta, GA, USA, 2013, vol. 30, no. 1, p. 3.

[39] J. M. Kasson, S. I. Nin, W. Plouffe, and J. L. Hafner, "Performing color space conversions with three-dimensional linear interpolation," *J. Electron. Imag.*, vol. 4, no. 3, pp. 226–250, 1995.

[40] H. Zhao, X. Kong, J. He, Y. Qiao, and C. Dong, "Efficient image super-resolution using pixel attention," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2020, pp. 56–72.

[41] J. Cao, Y. Li, K. Zhang, and L. Van Gool, "Video super-resolution transformer," 2021, *arXiv:2106.06847*.

[42] X. Wang, K. C. K. Chan, K. Yu, C. Dong, and C. C. Loy, "EDVR: Video restoration with enhanced deformable convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 1954–1963.

Guanghao Yin received the B.S. degree from the College of Automation, Huazhong University of Science and Technology, Wuhan, China, in 2018. He is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Zhejiang University, Hangzhou, China. His current research interests include low-level image processing, image quality assessment, affective computing, and human–machine interface.

Zefan Qu received the B.S. degree from the College of Software Engineering, Dalian University of Technology, Dalian, China, in 2021. He is currently pursuing the master's degree with the College of Electronics and Information Engineering, Tongji University, Shanghai, China. His research interests include computer vision, deep learning, low-level vision, and person re-identification.

Xinyang Jiang received the Ph.D. degree in computer science and technology from Zhejiang University in 2017. He was a Senior Researcher with the Tencent Youtu Laboratory. He is currently a Researcher with Microsoft Research Asia. He has published more than ten papers in CVPR, ECCV, AAAI, ACMMM, IEEE TRANSACTIONS ON IMAGE PROCESSING, and other top conferences and journals on computer vision and artificial intelligence. His current research interests include cross-modal retrieval, computer vision, and pedestrian re-identification. He is a Program Member of AAAI, CVPR, MM, and other conferences, and a reviewer for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON IMAGE PROCESSING, and other journals.

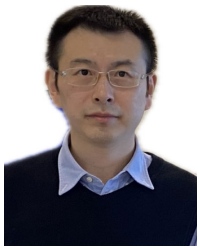
Shan Jiang received the B.S. degree in computer science and technology from China University of Mining and Technology in 2021. He is currently pursuing the M.S. degree with the School of Computer Science and Technology, University of Science and Technology of China. From July 2022 to July 2023, he was a full-time Research Intern with MSRA. His research interests include computer vision and machine learning systems.

Zhenhua Han received the B.Eng. degree in electronic and information engineering from the University of Electronic Science and Technology of China in 2014 and the Ph.D. degree from The University of Hong Kong (HKU) in 2020. He is currently a Senior Researcher with Microsoft Research Asia, Shanghai. His research interests include resource management, systems for machine learning, and cloud computing.

Ningxin Zheng received the B.S. degree from Huazhong University of Science and Technology in 2017 and the M.S. degree from Shanghai Jiao Tong University in 2020. He is currently a Research Software Development Engineer II with Microsoft Research Shanghai. His research interests include AI systems, cloud computing, and model compression.

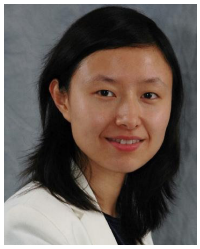
Huan Yang received the B.S. and Ph.D. degrees in computer science from Shanghai Jiao Tong University in 2014 and 2019, respectively. He is currently a Senior Researcher with the Multi-Modal Computing Group, Microsoft Research Asia. He has published about 20 papers at the top international CV/AI conferences, such as CVPR, ICCV, ECCV, and NeurIPS. His research interests include GAN/diffusion-based content creation, image/video restoration, and enhancement.

Xiaohong Liu (Member, IEEE) received the B.E. degree in communication engineering from Southwest Jiaotong University, Chengdu, China, in 2014, the M.A.Sc. degree in electrical and computer engineering from the University of Ottawa, Ottawa, ON, Canada, in 2016, and the Ph.D. degree in electrical and computer engineering from McMaster University, Hamilton, ON, Canada, in 2021. He is currently a Tenure-Track Assistant Professor with the John Hopcroft Center, Shanghai Jiao Tong University, Shanghai, China. His research interests include image/video restoration and image segmentation. He was a recipient of the Ontario Graduate Scholarship in 2019, the NSERC Alexander Graham Bell Canada Graduate Scholarship-Doctoral, and the Borealis AI Global Fellowship Award in 2020. He is a reviewer of several IEEE journals, including IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.



Yuqing Yang received the B.S. and Ph.D. degrees in microelectronics from Fudan University, Shanghai, China, in 2006 and 2011, respectively. He is currently with Microsoft Research Asia, with a focus on efficient computing systems for emerging scenarios, such as deep learning and video streaming. His current research interests include execution optimization for dynamic and sparse neural networks, efficient computing for cloud and edge, and neural-enhanced video streaming.

Dongsheng Li (Senior Member, IEEE) received the B.E. degree from the Department of Computer Science and Technology, University of Science and Technology of China (USTC), in 2007, and the Ph.D. degree from the School of Computer Science, Fudan University, in 2012. He is currently a Principal Research Manager with Microsoft Research Asia (MSRA), Shanghai, China. He is also an Adjunct Professor with the School of Computer Science, Fudan University, Shanghai. Before joining MSRA, he was a Research Staff Member (RSM) with IBM Research, Shanghai. His research interests include machine learning, AI for health, and recommender systems. He is a Senior Member of ACM. He has served as the Program Committee Member of several top conferences, such as NIPS, ICML, and ICLR. In 2018, his work on the Cognitive Recommendation Engine won the IBM Corporate Award.



Lili Qiu received the M.S. and Ph.D. degrees in computer science from Cornell University. She is currently an Assistant Managing Director of Microsoft Research Asia, Shanghai, where she is mainly responsible for overseeing research and collaboration with industries, universities, and research institutes. She is an expert in internet and wireless networking and was with the System and Networking Group, Microsoft Research Redmond, as a Researcher, from 2001 to 2004. In 2005, she joined The University of Texas at Austin as an Assistant Professor with the Department of Computer Science. Later, she was promoted to a tenured Professor and a Doctoral Advisor, in view of her outstanding achievements in the internet and wireless networks fields. She is an ACM Fellow. She serves as the ACM SIGMOBILE Chair. She was named an ACM Distinguished Scientist and was a recipient of the NSF CAREER Award, among many other honors.