Overfitting the Data: Compact Neural Video Delivery via Content-aware Feature Modulation

Jiaming Liu[1*], Ming Lu[2*†], Kaixin Chen[1*], Xiaoqi Li[1*], Shizun Wang[1*], Zhaoqing Wang[1], Enhua Wu[3],
Yurong Chen[2], Chuang Zhang[1‡], and Ming Wu[1]

[1]Beijing University of Posts and Telecommunications
[2]Intel Labs China
[3]State Key Lab of Computer Science, IOS, CAS & FST, University of Macau
{liujiaming,zhangchuang}@bupt.edu.cn,lu199192@gmail.com

Abstract

Internet video delivery has undergone a tremendous explosion of growth over the past few years. However, the quality of video delivery system greatly depends on the Internet bandwidth. Deep Neural Networks (DNNs) are utilized to improve the quality of video delivery recently. These methods divide a video into chunks, and stream LR video chunks and corresponding content-aware models to the client. The client runs the inference of models to super-resolve the LR chunks. Consequently, a large number of models are streamed in order to deliver a video. In this paper, we first carefully study the relation between models of different chunks, then we tactfully design a joint training framework along with the Content-aware Feature Modulation (CaFM) layer to compress these models for neural video delivery. **With our method, each video chunk only requires less than 1% of original parameters to be streamed, achieving even better SR performance.** *We conduct extensive experiments across various SR backbones, video time length, and scaling factors to demonstrate the advantages of our method. Besides, our method can be also viewed as a new approach of video coding. Our primary experiments achieve better video quality compared with the commercial H.264 and H.265 standard under the same storage cost, showing the great potential of the proposed method. Code is available at:* `https://github.com/Neural-video-delivery/CaFM-Pytorch-ICCV2021`

1. Introduction

Internet video is achieving explosive growth over the past few years, which brings a huge burden to the video delivery infrastructure. The quality of video heavily depends on the bandwidth between servers and clients. Techniques at both sides evolve over time to handle the scalability challenges at Internet scale. Inspired by the increasing computational power of client/server and recent advances in deep learning, several works are proposed to apply Deep Neural Networks (DNNs) to video delivery system [14, 30]. The core idea of these works is to stream both the low resolution video and content-aware models from servers to clients. The clients run the inference of models to super-resolve the LR videos. In this manner, better user Quality of Experience (QoE) can be obtained under limited Internet bandwidth.

In contrast to current approaches on Single Image Super-Resolution (SISR) [24, 8, 20, 32, 15] and Video Super-Resolution (VSR) [2, 26, 3], content-aware DNNs leverage neural network's overfitting property and use the training accuracy to achieve high performance. Specifically, a video is first divided into several chunks, and then a separate DNN is trained for each chunk. The low resolution chunks and corresponding trained models are delivered to the clients over the Internet. Different backbones [24, 8, 20, 32, 15] can be used as the DNN for each chunk. This kind of DNN-based video delivery system has achieved better performance compared with commercial video delivery techniques like WebRTC [14].

Although it is promising to apply DNNs to video delivery, existing methods still have several limitations [18]. One major limitation is that they need to train one DNN for each chunk, resulting in a large number of separate models for a long video. This brings additional storage and bandwidth

---

cost for the practical video delivery system. In this paper, we first carefully study the relation between models of different chunks. Although these models are trained to overfit different chunks, we observe the relation between their feature maps is linear and can be modelled by a Content-aware Feature Modulation (CaFM) layer. This motivates us to design a method, which allows the models to share the most of parameters and preserve only private CaFM layers for each chunk. However, directly finetuning the private parameter fails to obtain competitive performance compared with separately trained models. Therefore, we further design a tactful joint training framework, which trains the shared parameters and private parameters simultaneously for all chunks. In this way, our method can achieve relative better performance compared with individually trained models.

Apart from video delivery, our method can also be considered as a new approach of video coding. We conduct primary experiments to compare the proposed approach against commercial H.264 and H.265 standards under the same storage cost. Our method can achieve higher PSNR performance thanks to the overfitting property, showing the great potential of the proposed approach.

Our contributions can be concluded as follows:

- We propose a novel joint training framework along with the content-aware feature modulation layer for neural video delivery.

- We conduct extensive experiments across various SR backbones, video time length, and scaling factors to demonstrate the advantages of our method.

- We compare with commercial H.264 and H.265 standard under the same storage cost and show promising results thanks to the overfitting property.

2. Related Work

DNN-based Image Super-Resolution SRCNN [8] is the pioneering work that introduces DNN to the SR task. Their DNN consists of three stages, namely feature extraction, non-linear mapping, and image reconstruction. With the development of DNN, following the pipeline of SRCNN, plenty of DNN-based methods are proposed to improve the performance of SISR. For example, VDSR [15] uses a very deep DNN to learn the image residual instead of HR image. Inspired by ResNet [10], SRResNet [17] introduces Residual Block to SR and improves the capability of DNN. EDSR [20] modifies the structure of SRResNet and removes the Batch Normalization layer [12], further advancing the SR results. RDN [33] proposes to adopt dense connection [11] in order to fully use previous layers' information. RCAN [32] studies the attention mechanism and presents deeper DNs for SR. However, RCAN

is computationally complicated, which limits its practical usage. To reduce the computational cost, many works are proposed for efficient SR. ESPCN [24] uses LR image as input and upsamples the feature map by the PixelShuffle layer at the end of DNN. LapSRN [16] proposes the Laplacian Pyramid network for fast and accurate SR, which progressively reconstructs the sub-band residuals of HR images. FALSR [6] uses Neural Architecture Search (NAS) to search the lightweight and accurate networks for SR. LA-PAR [19] presents a method based on linearly-assembled pixel-adaptive regression network, which learns the pixel-wise filter kernel for SR. All of those methods are external methods, which train the model on large-scale image databases like DIV2K [1] and test on certain benchmark databases. However, external methods fail to explore the overfitting property of DNNs, which are useful for practical video delivery systems.

DNN-based Video Super-Resolution In contrast to image super-resolution, video super-resolution can additionally utilize temporal information for SR. Therefore, temporal alignment plays an important role and should be extensively studied. VESPCN [2] estimates the motions between neighboring frames and performs image warping before feeding neighboring frames into SR network. However, a precise estimation of optical flow is often not tractable. TOFlow [28] proposes a task-oriented flow tailored for specific video processing tasks. They jointly train the motion estimation component and video processing component in a self-supervised manner. DUF [13] avoids the accurate explicit motion compensation by training a network to generate dynamic upsampling filters and a residual image. EDVR [26] improves the performance of VSR from two aspects. First, they devise an alignment module to handle large motions, in which frame alignment is performed at the feature level using deformable convolutions in a coarse-to-fine manner. Second, they apply attention mechanism both temporally and spatially, aiming to emphasize important features for subsequent restoration. In order to reduce the computational cost of VSR, FRVSR [23] presents a recurrent framework that uses the previous SR to super-resolve the subsequent frame. Their recurrent frames naturally encourage temprally consistency and reduce the computational cost by warping only one image in each step. TecoGAN [5] explores the temporal self-supervision for GAN-based VSR. They propose a temporal adversarial learning method that achieves temporally coherent solutions without sacrificing spatial detail. BasicVSR [3] reconsider some most essential components for VSR by reusing some existing components added with minimal redesigns. They achieve appealing improvements in terms of speed and restoration quality in comparison to state-of-the-art algorithms. All these VSR approaches also belong to external methods that fail to explore the overfitting property of DNN. Apart from
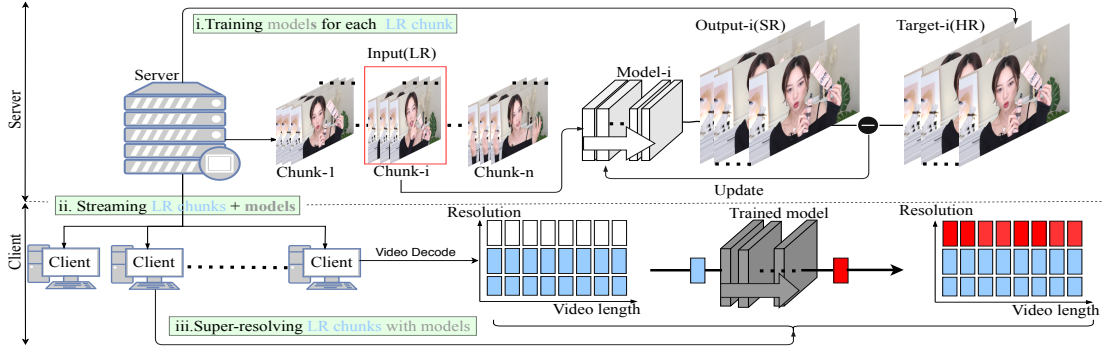
Figure 1. The whole procedure of adopting content-aware DNNs for video delivery. A video is first divided into several chunks and the server trains one model for each chunk. Then the server delivers LR video chunks and models to client. The client runs the inference to super-resolve the LR chunks and obtain the SR video.

this, handling temporal alignment brings huge additional computational and storage cost, which limits their practical applications in resource-limited devices like mobile phone.

Neural Video Delivery NAS [30] is a new and promising practical Internet video delivery framework that integrates DNN based quality enhancement. It can solve the video quality degradation problem when Internet bandwidth becomes congested. NAS can enhance the average Quality of Experience (QoE) by 43.08% using the same bandwidth budget, or saving 17.13% of bandwidth while providing the same user QoE. The core idea is to leverage DNN's overfitting property and use the training accuracy to deliver high enhancement performance. Many following works are proposed to apply the idea of NAS to different applications, like UAV video streaming [27], Live Streaming [14], 360 Video Streaming [7, 4], Volumetric Video Streaming [31], and Mobile Video Streaming [29], etc. In this paper, the proposed method can further reduce the bandwidth budget by sharing most of the parameters over video chunks. Therefore, only a small portion of private parameters are streamed for each video chunk.

3. Proposed Method

Neural video delivery [14, 30] aims at utilizing DNNs to save bandwidth when delivering Internet videos. Different from traditional video delivery systems, they replace HR video with LR video and content-aware models. As shown in Fig. 1, the whole procedure includes three stages: (i) training models for video chunks on the server; (ii) delivering LR chunks along with content-aware models from server to client; (iii) super-resolving LR chunks on client. However, this procedure requires to stream one model for each chunk, resulting in additional bandwidth cost. In this part, we first analyze the relation between models of video chunks in Section 3.1. Then we introduce a Content-aware Feature Modulation (CaFM) layer to model this relation in

section 3.2. However, fine-tuning the CaFM module on each chunk fails to achieve competitive results. Therefore, we propose a joint training framework in Section 3.3.

3.1. Motivation

In this part, we adopt the popular architecture EDSR [20] and analyze the models trained on video chunks. Following former works [30, 14], we split a video into n chunks and train n EDSR models $S_1, S_2...S_n$ for these chunks accordingly. Then we empirically analyze $S_1, S_2...S_n$ by feeding a randomly selected image of DIV2K [1]. We visualize the feature maps of these n EDSR models in Fig. 2. Each image represents the feature map of a certain channel and we only visualize one layer of EDSR for simplicity. Specifically, we denote the feature map as $f_{i,j,k} \in R^{H \times W}$, where i means the ith model, j means j^{th} channel, and k means the kth layer of EDSR. For the randomly selected image, we can calculate the cosine distance between $f_{i_1,j_1,k}$ and $f_{i_2,j_2,k}$, which measure the similarity between these two feature maps. For the feature maps in Fig. 2, we calculate the cosine distance matrix among $f_{1,:,k}$, $f_{2,:,k}$ and $f_{3,:,k}$. As shown in Fig. 3, we observe that although $S_1, S_2...S_n$ are trained on different chunks, the cosine distance between corresponding channel is very small according to the diagonal values of the matrices in Fig. 3. We calculate the average of cosine distance among S_1, S_2, and S_3 across all layers, and the results are about 0.16 and 0.04 separately. **This indicates that although different models are trained on different chunks, the relation between $f_{1,j,k}$ and $f_{2,j,k}$ can be approximately modelled by a linear function.** The aforementioned observation motivates us to share most of the parameters for $S_1, S_2...S_n$ and privatize each DNN with Content-aware Feature Modulation (CaFM), which will be elaborated in the next part.
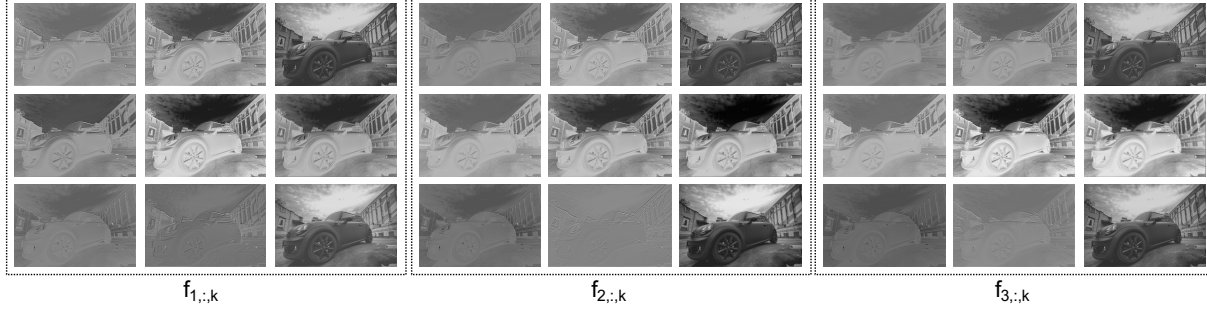
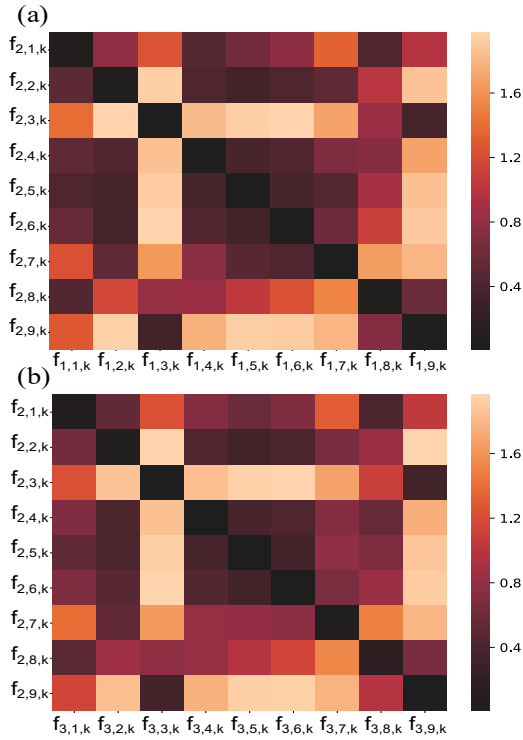Figure 2. The feature map visualization of different models S_{1-n}.

(a)

(b)

Figure 3. The cosine distance matrices of $f_{1,:,k}$, $f_{2,:,k}$ and $f_{3,:,k}$ in Fig. 2

3.2. Content-aware Feature Modulation

In this part, we introduce the Content-aware Feature Modulation (CaFM) layer into baseline models to privatize the DNN for each video chunk. The overall framework is shown in Fig. 4. As mentioned in the motivation, CaFM aims to manipulate the feature map and adapt the model to different chunks. Therefore, models of different chunks can share most of the parameters. We represent CaFM as a channel-wise linear function:

$$CaFM(x_j) = a_j * x_j + b_j, 0 < j \leq C \qquad (1)$$

where x_j is the jth input feature map, C is the number of feature channels, a_j and b_j are the channel-wise scaling and bias parameters respectively. We add CaFM to modulate the output feature of each filter for the baseline models. Taking EDSR as the example, the parameter of CaFM takes up about **0.6%** of EDSR. Therefore, for a video with n chunks, we can reduce the size of models from n EDSR to 1 shared EDSR plus n private CaFM. Thus our method can significantly reduce the band-width and storage cost compared with baseline methods.

3.3. Joint training framework

As mentioned in the above section, we can privatize the DNN of each chunk with CaFM. However, finetuning the added CaFM module cannot achieve competitive results compared with separately trained models. Therefore, we further propose a joint training framework, which simultaneously trains the models of video chunks. Given a LR image $I_{LR}^{i,s}$, where i denotes the ith video chunk and s denotes the sth sample in this chunk, we can obtain the SR image:

$$I_{SR}^{i,s} = g(I_{LR}^{i,s}; W_s, W_i) \qquad (2)$$

where W_s is the shared parameter and W_i is the parameter of CaFM for i^{th} video chunk. For each video chunk, we can define the reconstruction loss:

$$L_i = \frac{\sum_{s=1}^{S} \left\| I_{HR}^{i,s} - I_{SR}^{i,s} \right\|_1}{S} \qquad (3)$$

Therefore, the loss function of joint training can be represented as:

$$\mathcal{L} = \sum_{i=1}^{n} L_i \qquad (4)$$

During training, we uniformly sample the images from video chunks to construct the training data. All images are used to update the shared parameter W_s, while images of ith video chunk are used to update the corresponding CaFM parameter W_i.
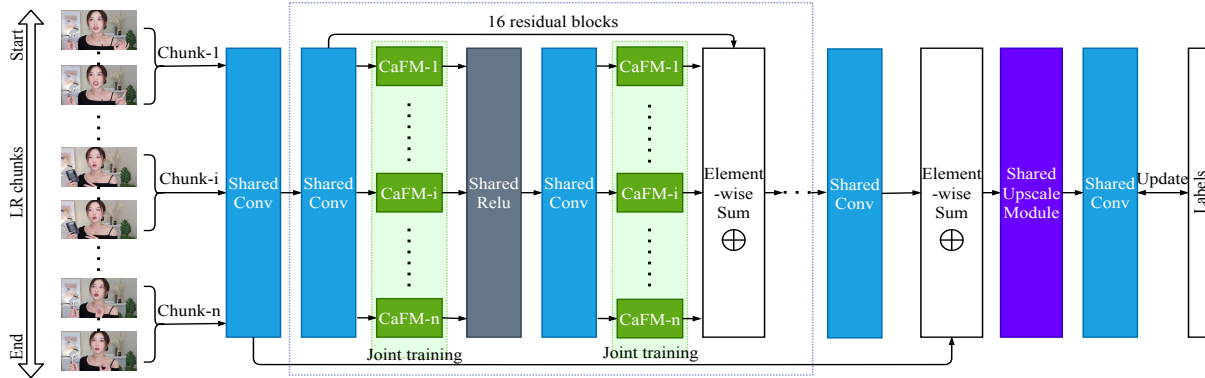
Figure 4. The joint training framework along with CaFM.

4. Experiments

In this section, we conduct extensive experiments to show the advantages of our method. To evaluate the proposed approach, we construct a Video Streaming Dataset with 4K videos (VSD4K). The details of VSD4K, training protocol, and model structure are given in Section 4.1. In Section 4.2, we compare the performance of external learning and content-aware learning based on EDSR [20] and EDVR [26]. In section 4.3, we report the results of our method across different video lengths and scaling factors. We also conduct comprehensive ablation study in Section 4.4 to evaluate the contribution of each component. In order to demonstrate the generalization ability of our work, we report results with various popular SR architectures in Section 4.5. Finally, since our method can be also considered as a video coding method, we compare it with the commercial H.264 and H.265 standard in Section 4.6.

4.1. Experimental Details

Video Streaming Dataset 4K Public video datasets like Vimeo-90K [28] and REDS [21] only consist of neighboring frame sequences, which are not suitable for video delivery. Therefore, we collect several 4K videos from YouTube to simulate practical video delivery scenario. We use bicubic interpolation to generate the LR videos following former works [20, 32]. We select 6 popular video categories to construct VSD4K. Each category consists of various video lengths including: 15s, 30s, 45s, 1min, 2min, 5min. Detailed information of VSD4K can be found in the supplementary material.

Training details Since neural video delivery relies on the overfitting property of DNN, we train and test on the same video. In order to reduce the computational cost, we sample 1 frame out of 10 frames for testing. We use 48×48 HR patches with corresponding LR patches for training. We adopt Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. L1 loss is adopted as the loss function. The learning rate is set as 10^{-4} and decay at different iterations depends on video length. Besides, we set different mini-batch sizes for different video lengths to maintain the computational cost same as the sum of S_{1-n}.

Model structure Since our method can be applied to different SR architectures, we conduct extensive experiments with various popular networks (EDSR, VDSR, ESPCN, and SRCNN). We report most of the ablation study results on EDSR for the consideration of simplicity. We add the proposed CaFM module to the output of each convolutional layer in EDSR. The CaFM module is implemented as a 1×1 depth-wise convolution. Other SR architectures can be modified accordingly.

4.2. Evaluation on content-aware learning

In this part, we demonstrate the benefit of utilizing DNN's overfitting property for video delivery. Methods like EDSR and EDVR train the DNN on a large-scale dataset and use the trained DNN to super-resolve the LR input. We call this kind of training external learning. However, for neural video delivery, the video is known beforehand. Therefore, we can train the models to overfit each video and achieve better SR performance. This kind of training can be named content-aware learning. Apart from this, a video can be further divided into video chunks and a DNN is trained for each chunk separately. We compare the performance of external learning and content-aware learning in Tab. 1. As we can see, content-aware learning can achieve much better results compared with external learning. In particular, EDSR with content-aware learning outperforms EDVR with external learning by significant margins. These results prove that content-aware learning is more suitable for video delivery compared with external learning.

4.3. Evaluation on VSD4K datasets

In this section, we report the results on VSD4K. We mainly present the results of 15s, 30s, 45s, 1min, 2min,

			game-45s			vlog-45s			inter-45s		
Method	Model	Dataset	x2	x3	x4	x2	x3	x4	x2	x3	x4
External learning	EDSR-M	DIV2K	36.02	31.20	29.00	46.85	42.70	40.41	38.13	32.00	28.96
	EDVR-M	REDS	-	-	28.72	-	-	41.28	-	-	28.90
	EDVR-L	REDS	-	-	29.45	-	-	40.42	-	-	27.30
	EDVR-L	Vimeo-90K	-	-	30.01	-	-	42.04	-	-	30.06
Content-aware learning	EDSR-M	VSD4K	42.11	35.75	33.33	47.98	43.58	41.53	42.73	34.49	31.34
Content-aware learning*	EDSR-M	VSD4K	43.22	36.72	34.32	48.48	44.12	42.12	43.31	35.80	32.67
			sport-45s			dance-45s			city-45s		
			x2	x3	x4	x2	x3	x4	x2	x3	x4
External learning	EDSR-M	DIV2K	46.25	41.23	38.07	40.55	37.72	35.51	37.32	32.20	29.73
	EDVR-M	REDS	-	-	40.10	-	-	35.93	-	-	31.52
	EDVR-L	REDS	-	-	40.31	-	-	36.84	-	-	31.91
	EDVR-L	Vimeo-90K	-	-	41.38	-	-	35.74	-	-	32.28
Content-aware learning	EDSR-M	VSD4K	47.81	42.28	39.16	45.49	38.26	37.32	38.67	33.44	30.86
Content-aware learning*	EDSR-M	VSD4K	48.34	43.01	40.34	45.71	38.61	37.67	39.87	34.62	31.97

Table 1. Comparison of content-aware learning versus external training. EDVR-M, EDVR-L, EDSR-M has 10, 40, 16 resblocks respectively. * denotes training a content-aware DNN for each video chunk. Red and blue indicates the best and the second best results.

	game-15s			game-30s			game-45s		
Scale	x2	x3	x4	x2	x3	x4	x2	x3	x4
M0	42.24	35.88	33.44	41.84	35.54	33.05	42.11	35.75	33.33
S_{1-n}	42.82	36.42	34.00	43.07	36.73	34.17	43.22	36.72	34.32
Ours	43.13	37.04	34.47	43.08	36.94	34.22	43.32	37.19	34.61
Margin	0.31	0.62	0.47	0.01	0.21	0.05	0.10	0.47	0.29
	vlog-15s			vlog-30s			vlog-45s		
Scale	x2	x3	x4	x2	x3	x4	x2	x3	x4
M0	48.87	44.51	42.58	47.79	43.38	41.24	47.98	43.58	41.53
S_{1-n}	49.10	44.80	42.83	48.20	43.68	41.55	48.48	44.12	42.12
Ours	49.30	45.03	43.11	48.35	43.94	41.90	48.45	44.11	42.16
Margin	0.20	0.23	0.28	0.15	0.26	0.35	-0.03	-0.01	0.04
	inter-15s			inter-30s			inter-45s		
Scale	x2	x3	x4	x2	x3	x4	x2	x3	x4
M0	44.85	37.89	34.94	43.06	35.37	32.30	42.73	34.49	31.34
S_{1-n}	45.06	38.38	35.47	43.50	36.48	33.42	43.31	35.80	32.67
Ours	45.35	38.66	35.70	43.65	36.30	33.28	43.37	35.62	32.35
Margin	0.29	0.28	0.23	0.15	-0.18	-0.14	0.06	-0.18	-0.32
	sport-15s			sport-30s			sport-45s		
Scale	x2	x3	x4	x2	x3	x4	x2	x3	x4
M0	48.20	42.56	39.66	50.36	44.72	41.86	47.81	42.28	39.16
S_{1-n}	48.43	43.04	40.38	50.67	45.45	42.94	48.34	43.01	40.34
Ours	48.48	43.06	40.43	50.74	45.31	42.73	48.24	42.93	40.21
Margin	0.05	0.02	0.05	0.07	-0.14	-0.21	-0.10	-0.08	-0.13
	dance-15s			dance-30s			dance-45s		
Scale	x2	x3	x4	x2	x3	x4	x2	x3	x4
M0	44.35	37.57	36.18	44.85	37.99	36.67	45.49	38.26	37.32
S_{1-n}	44.48	37.69	36.40	44.99	38.13	36.93	45.71	38.61	37.67
Ours	44.71	37.82	36.61	45.30	39.66	37.22	45.97	39.96	37.84
Margin	0.23	0.13	0.21	0.31	1.53	0.29	0.26	1.35	0.17
	city-15s			city-30s			city-45s		
Scale	x2	x3	x4	x2	x3	x4	x2	x3	x4
M0	37.89	32.32	29.36	38.90	33.15	30.35	38.67	33.44	30.86
S_{1-n}	38.14	32.61	29.67	39.85	34.15	31.30	39.87	34.62	31.97
Ours	38.17	32.65	29.68	39.70	34.01	31.18	39.75	34.48	31.87
Margin	0.03	0.04	0.01	-0.15	-0.14	-0.12	-0.12	-0.14	-0.10
	game-1min			game-2min			game-5min		
Scale	x2	x3	x4	x2	x3	x4	x2	x3	x4
M0	41.82	35.25	32.61	41.89	35.72	33.27	40.62	34.59	32.14
S_{1-n}	43.24	36.56	33.52	43.20	37.00	34.47	42.47	36.08	33.53
Ours	43.49	37.18	34.33	43.49	37.47	34.80	43.01	36.65	34.07
Margin	0.25	0.62	0.81	0.29	0.47	0.33	0.54	0.57	0.54

Table 2. The comprehensive results of our method on VSD4K. We conduct experiments with different video length and scaling factors. We also show the margins between our method and S_{1-n} [30].

and 5min across scaling factors $\times 2, \times 3$, and $\times 4$. We compare our results with two other methods. The first method trains a content-aware DNN for the whole video, which is denoted as M0. The second method divides a video into n chunks and trains one DNN for each chunk [30]. We denote the second method as S_{1-n}. Compared with S_{1-n} [30], our methods significantly reduce the parameters of models. As shown in Tab. 2, our methods can consistently obtain
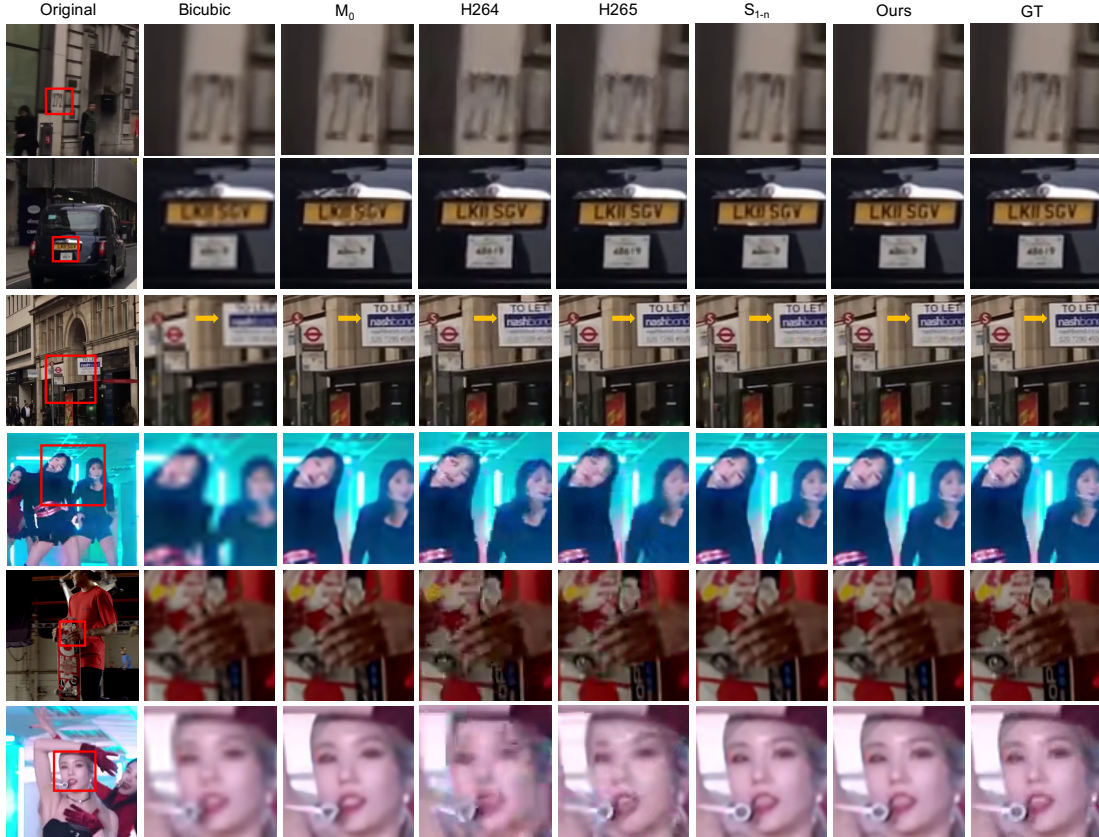
Figure 5. Qualitative comparisons on the VSD4K dataset. Best viewed by zooming x4.

	game-45s			vlog-45s			inter-45s		
Method	x2	x3	x4	x2	x3	x4	x2	x3	x4
H.264	39.26	36.99	35.52	43.45	42.07	41.31	37.99	36.07	35.06
H.265	39.77	37.71	36.42	44.24	43.09	42.31	38.31	36.51	35.58
Ours	43.32	37.19	34.61	48.45	44.11	42.16	43.37	35.62	32.35
Storage(MB)	19.9	12.6	9.8	19.5	12.5	9.8	19.5	12.4	9.8
	sport-45s			dance-45s			city-45s		
Method	x2	x3	x4	x2	x3	x4	x2	x3	x4
H.264	40.30	38.09	36.83	31.11	28.32	26.76	36.60	34.18	32.89
H.265	41.35	39.66	38.67	32.62	30.40	29.18	37.17	35.10	34.03
Ours	48.24	42.93	40.21	45.97	39.96	37.84	39.75	34.48	31.87
Storage(MB)	19.8	12.6	9.9	19.6	12.6	9.8	20.4	12.9	10.1

Table 3. Comparisons with H.264/H.265. Red and blue indicates the best and the second best results.

slightly better performance with less parameters compared with S_{1-n} [30]. We also show the qualitative comparison in Fig. 5.

4.4. Ablation study

Variants of CaFM Since we implement CaFM as 1×1 depth-wise convolution, we also study the effect of different kernel sizes for depth-wise convolution. We compare the results of 1×1, 3×3, 5×5, and 7×7 as shown in Tab. 5. We also show the percentage of CaFM's parameter compared with EDSR in Tab. 5. In general, larger kernel can achieve slightly better performance but will result in more

parameters. Nevertheless, CaFM with 1×1 kernel already achieves competitive performance.

Benefit of Joint Training As described above, joint training is important for our method to achieve good performance. To evaluate the benefit of joint training, we first train the DNN on a specific video from scratch, denoting M_0. Then we freeze the parameters and add CaFM for each chunk. We finetune the parameters of CaFM to overfit each chunk. This result is denoted as FT in Tab. 6. The results of training separate DNN for each chunk are also reported as S_{1-n} [30]. Our results outperform the results of other methods, demonstrating the benefit of joint training.

Model	Data Scale	game-45s			dance-45s			inter-45s			vlog-45s		
		x2	x3	x4	x2	x3	x4	x2	x3	x4	x2	x3	x4
ESPCN	M0	35.42	30.63	28.65	43.12	36.62	34.95	38.64	31.97	28.32	45.71	41.40	39.20
	S_{1-n}	35.55	30.67	28.74	43.27	36.72	35.09	38.81	32.14	28.61	45.81	41.52	39.29
	Ours	36.09	31.06	29.05	43.56	36.89	35.30	38.88	32.22	28.75	46.19	41.72	39.52
SRCNN	M0	35.05	30.50	28.59	42.67	36.79	34.60	38.66	31.78	28.25	45.87	41.58	39.29
	S_{1-n}	35.15	30.55	28.61	42.69	36.90	34.69	38.79	31.93	28.38	45.95	41.66	39.36
	Ours	35.49	30.63	28.66	43.06	37.01	34.86	38.88	32.02	28.48	46.18	41.85	39.52
VDSR	M0	40.29	34.53	31.28	45.03	37.95	36.57	41.99	33.80	30.34	47.61	42.92	40.94
	S_{1-n}	41.37	34.92	32.42	45.18	38.05	36.82	42.40	34.53	31.10	47.88	43.33	41.23
	Ours	41.92	35.56	33.16	45.41	38.24	37.16	42.86	34.49	30.95	48.00	43.50	41.38
EDSR	M0	42.11	35.75	33.33	45.49	38.26	37.32	42.73	34.49	31.34	47.98	43.58	41.53
	S_{1-n}	43.22	36.72	34.32	45.71	38.61	37.67	43.31	35.80	32.67	48.48	44.12	42.12
	Ours	43.32	37.19	34.61	46.00	39.96	37.84	43.37	35.62	32.35	48.45	44.11	42.16

Table 4. The generalization ability of our method on various SR architecture.

Dataset:	game-15s	1x1	3x3	5x5	7x7
PSNR:	x2	43.13	43.16	43.17	43.23
	x3	37.04	37.09	37.10	37.18
	x4	34.47	34.45	34.50	34.54
Percentage:	x2	0.66%	2.20%	4.62%	8.19%
	x3	0.58%	1.96%	3.74%	7.23%
	x4	0.60%	1.67%	3.83%	7.40%

Table 5. The above part shows the PSNR of different kernel size in 15s game video. The below part demonstrates the parameter percentage of CaFM compared with EDSR.

		M0	S_{1-n}	FT	Ours
game-15s	x2	42.24	42.82	42.30	43.13
	x3	35.88	36.42	35.93	37.04
	x4	33.44	34.00	33.49	34.47
dance-15s	x2	44.35	44.48	44.37	44.71
	x3	37.57	37.69	37.59	37.82
	x4	36.18	36.40	36.22	36.61

Table 6. Benefit of joint training.

4.5. The generalization of our method

As shown in Tab. 4, we present the results of our method applied to different SR backbones. We select four 45s videos and adopt 4 popular SR backbones: SRCNN [8], ES-PCN [24], VDSR [15] and EDSR [20]. These experiments aim to study the generalization ability of our method. The conclusion is consistent with the results shown above. The joint training framework along with CaFM can generalize well to other SR architectures, validating the generalization ability of our method.

4.6. Comparisons with H.264/H.265

Our method can be also viewed as a video coding approach. Therefore, we conduct primary experiments to compare our method with commercial H.264 and H.265 standard. For H.264 and H.265, we decrease the bitrate of the video while maintaining the resolution to obtain a video with the same size as our method (LR video and models). We compare our SR video with low bitrate videos of H.264 and H.265. Six videos are randomly selected from

VSD4K to conduct the comparison with H.264 and H.265. The quantitative results are shown in Tab. 3. Our results outperform H.264 and H.265 in most cases. We also show the qualitative comparison in Fig. 5.

This primary experiment shows the great potential of our method. Furthermore, since the proposed CaFM takes less than 1% additional parameters for each chunk, we believe our method can achieve better results by dividing the video into more chunks. However, this will result in more training time accordingly.

5. Future Work

Although our method utilizes the overfitting property of DNN to achieve significant SR performance, it needs to train one network for each video chunk. This brings additional cost for network training. In order to further improve the video delivery systems, we believe reducing the training time is a promising and important future work. Learning methods like MAML [9] can be used to accelerate the training process of SR networks. Recently, MAML has been applied to zero-shot image SR [22, 25], aiming to reduce the computational cost of DNN training.

6. Conclusion

In this paper, we study the problem of applying DNN to video delivery. Neural video delivery adopts the overfitting property of DNN, by training one model for each video chunk. In order to avoid streaming one model for each video chunk, we propose a novel joint training framework along with the CaFM module. With our method, each video chunk only requires less than 1% of original parameters to be streamed and achieve better SR performance. We conduct comprehensive analysis of our method to demonstrate its advantages. Besides, we also compare with the commercial H.264 and H.265 standards, showing the potential of our approach. We hope our work can inspire future work on applying DNN to video delivery.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[2] Jose Caballero, Christian Ledig, Andrew Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-time video super-resolution with spatio-temporal networks and motion compensation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4778–4787, 2017.

[3] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Basicvsr: The search for essential components in video super-resolution and beyond. *arXiv preprint arXiv:2012.02181*, 2020.

[4] Jiawen Chen, Miao Hu, Zhenxiao Luo, Zelong Wang, and Di Wu. Sr360: boosting 360-degree video streaming with super-resolution. In *Proceedings of the 30th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 1–6, 2020.

[5] Mengyu Chu, You Xie, Jonas Mayer, Laura Leal-Taixé, and Nils Thuerey. Learning temporal coherence via self-supervision for gan-based video generation. *ACM Transactions on Graphics (TOG)*, 39(4):75–1, 2020.

[6] Xiangxiang Chu, Bo Zhang, Hailong Ma, Ruijun Xu, and Qingyuan Li. Fast, accurate and lightweight super-resolution with neural architecture search. *arXiv preprint arXiv:1901.07261*, 2019.

[7] Mallesham Dasari, Arani Bhattacharya, Santiago Vargas, Pranjal Sahu, Aruna Balasubramanian, and Samir R Das. Streaming 360-degree videos using super-resolution. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 1977–1986. IEEE, 2020.

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

[9] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

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

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal co-variate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[13] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. Deep video super-resolution network using dynamic upsampling filters without explicit motion compen-

sation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3224–3232, 2018.

[14] Jaehong Kim, Youngmok Jung, Hyunho Yeo, Juncheol Ye, and Dongsu Han. Neural-enhanced live streaming: Improving live video ingest via online learning. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, pages 107–125, 2020.

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

[16] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Fast and accurate image super-resolution with deep laplacian pyramid networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2599–2613, 2018.

[17] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[18] Royson Lee, Stylianos I Venieris, and Nicholas D Lane. Neural enhancement in content delivery systems: The state-of-the-art and future directions. In *Proceedings of the 1st Workshop on Distributed Machine Learning*, pages 34–41, 2020.

[19] Wenbo Li, Kun Zhou, Lu Qi, Nianjuan Jiang, Jiangbo Lu, and Jiaya Jia. Lapar: Linearly-assembled pixel-adaptive regression network for single image super-resolution and beyond. *Advances in Neural Information Processing Systems*, 33, 2020.

[20] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[21] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[22] Seobin Park, Jinsu Yoo, Donghyeon Cho, Jiwon Kim, and Tae Hyun Kim. Fast adaptation to super-resolution networks via meta-learning. *arXiv preprint arXiv:2001.02905*, 5, 2020.

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

[24] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In

Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1874–1883, 2016.

[25] Jae Woong Soh, Sunwoo Cho, and Nam Ik Cho. Meta-transfer learning for zero-shot super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3516–3525, 2020.

[26] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[27] Xuedou Xiao, Wei Wang, Taobin Chen, Yang Cao, Tao Jiang, and Qian Zhang. Sensor-augmented neural adaptive bitrate video streaming on uavs. *IEEE Transactions on Multimedia*, 22(6):1567–1576, 2019.

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

[29] Hyunho Yeo, Chan Ju Chong, Youngmok Jung, Juncheol Ye, and Dongsu Han. Nemo: enabling neural-enhanced video streaming on commodity mobile devices. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–14, 2020.

[30] Hyunho Yeo, Youngmok Jung, Jaehong Kim, Jinwoo Shin, and Dongsu Han. Neural adaptive content-aware internet video delivery. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, pages 645–661, 2018.

[31] Anlan Zhang, Chendong Wang, Xing Liu, Bo Han, and Feng Qian. Mobile volumetric video streaming enhanced by super resolution. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, pages 462–463, 2020.

[32] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 286–301, 2018.

[33] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018.