A Codec Information Assisted Framework for Efficient Compressed Video Super-Resolution Supplementary File

Hengsheng Zhang[1], Xueyi Zou[2], Jiaming Guo[2], Youliang Yan[2], Rong Xie[1], and Li Song[1,3]✉

[1] Institute of Image Communication and Network Engineering, Shanghai Jiao Tong University
[2] Huawei Noah's Ark Lab
[3] MoE Key Lab of Artifical Intelligence, AI Institute, Shanghai Jiao Tong University
{hs_zhang,xierong,song_li}@sjtu.edu.cn
{zouxueyi,guojiaming5,yanyouliang}@huawei.com

In the supplementary file, we first present the details of the experiments. Then we provide additional experimental results to demonstrate the performance of our framework.

1 Experiments Details

1.1 Training Details

During training, we use the Adam optimizer[3] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. 64×64 patches are randomly cropped from the compressed REDS [5] training set, and the video length is set to 7. We enlarge the dataset with a ratio of 100 for saving time when restarting the data loader after each epoch[10]. And we totally train 60 epochs. The learning rate of VSR models is set to 2×10^{-4}; for SpyNet[6], the learning rate is set to 5×10^{-5}. We use "MultiStepLR" to adjust the learning rates, decaying the learning rates by gamma 0.1 once the number of epoch reaches 40 and 50.

1.2 Encoding and Decoding

In experiments, we encode the Gaussian-downsampled LR raw videos with FFmpeg under the popular video compression standard H.264[7]. The structure of the encoded group of pictures (GOP) is set to "IPP...P" by turning off bidirectional predictive mode($-bf$ 0 ($FFmpeg$)). Because every video of REDS[5] has 100 frames, the minimum distance between I-frames is set to
100 ($-keyint\_min$ 100 ($FFmpeg$)). The number of reference frames for the recurrent frame is set to 1 ($-refs$ 1 ($FFmpeg$)) so that the reference frame of the current frame is the previous encoded frame. When decoding, we modified the JM Reference Software to output Motion Vectors and Residuals while decoding video frames.

1.3 Implement Details

For FRVSR[8], because the spatial sizes of the motion field and the previous HR prediction are different, we first use the space-to-depth operator to reshape HR estimation to LR spatial size and then warp it according to MVs or optical flow. To improve training speed and stability, we add the network's output to the upsampled LR input to obtain the HR output. In every iteration of RLSP[1], the information of the past frame includes the latent features and the HR prediction. For HR prediction, we use the same processing in FRVSR. The RLSP also takes the next LR frame as an input for the current HR iteration. In experiments, we don't align the next frame. For RSDN[2], the past frame's latent features, structure, and detail information are all aligned with MVs or optical flow in experiments.

For sparse processing, we multiply the results of vanilla convolutions with predicted spatial masks during training. When testing, we first select the features that need to process according to the indices generated from the Residual-based spatial mask, then matrix multiplication is executed to produce the output features according to the selected features with size $L \times N$ and convolutional weights with size $C \times L$. N is the number of selected features, $L = C \times K_h \times K_w$ is the size of a selected pixel's feature, C is the number of channels, K_h and K_w are the height and width of the convolutional kernel.

Table 1: **The quantitative comparison (PSNR/ SSIM/ LPIPS)** on Vid4[4]. PSNR is calculated on Y-channel; SSIM and LPIPS are calculated on RGB-channel. Red and blue colors indicate the best and the second-best performance, respectively. 4× upsampling is performed following previous studies.

Model	CRF23				Compressed Results		
	calendar	walk	city	foliage	CRF18	CRF23	CRF28
FRVSR	20.91/0.5860/0.4490	26.27/0.7554/0.3047	25.17/0.5743/0.4747	23.43/0.5264/0.5304	24.74/0.6705/0.3767	23.95/0.6105/0.4397	22.84/0.5357/0.5257
FRVSR+MV	20.99/0.5979/0.4434	26.56/0.7634/0.2935	25.32/0.5879/0.4576	23.56/0.5365/0.5363	24.91/0.6817/0.3753	24.11/0.6214/0.4327	22.94/0.5431/0.5184
FRVSR+Flow	21.06/0.5991/0.4471	26.53/0.7634/0.2926	25.45/0.5955/0.4433	23.53/0.5387/0.5273	24.98/0.6859/0.3691	24.14/0.6242/0.4276	22.93/0.5436/0.5126
RLSP	20.75/0.5751/0.4524	26.21/0.7525/0.3093	25.08/0.5710/0.4685	23.44/0.5240/0.5408	24.57/0.6583/0.3849	23.87/0.6056/0.4427	22.84/0.5361/0.5232
RLSP+MV	21.12/0.6113/0.4203	26.68/0.7656/0.2780	25.58/0.6053/0.4334	23.64/0.5456/0.5255	25.15/0.6948/0.3548	24.26/0.6319/0.4143	23.02/0.5504/0.5002
RLSP+Flow	21.25/0.6154/0.4166	26.71/0.7677/0.2846	25.56/0.6058/0.4264	23.71/0.5515/0.5108	25.25/0.7008/0.3486	24.31/0.6351/0.4096	23.02/0.5511/0.4955
RSDN	21.00/0.5944/0.4231	26.32/0.7624/0.2891	25.12/0.5801/0.4443	23.51/0.5315/0.5327	24.79/0.6727/0.3638	23.99/0.6171/0.4223	22.88/0.5443/0.5033
RSDN+MV	21.35/0.6265/0.3965	26.75/0.7742/0.2699	25.42/0.6087/0.4151	23.54/0.5440/0.5151	25.22/0.7028/0.3382	24.27/0.6383/0.3992	23.06/0.5574/0.4826
RSDN+Flow	21.58/0.6416/0.3843	26.77/0.7731/0.2720	25.74/0.6217/0.4032	23.68/0.5580/0.5017	25.48/0.7166/0.3283	24.44/0.6486/0.3903	23.09/0.5609/0.4777

2 Additional Experiment Results

2.1 Effect of our MV-based Alignment

In this section, we provide additional comparisons on REDS4[9] and Vid4[4]. Tab 1 is the quantitative comparison on Vid4. Models with our MV-based alignment obviously outperform the models without alignment, even achieve comparable

performance with optical flow-based models. Fig 1 and Fig 2 are the qualitative results on the REDS4 and Vid4. Our MV-based alignment can significantly boost the visual results.

2.2 Residual Informed Sparse Processing

In this section, we provide additional qualitative results of our Residual informed sparse processing in Fig. 3. Models with our Residual informed sparse processing achieve comparable even superior visual results over baseline.

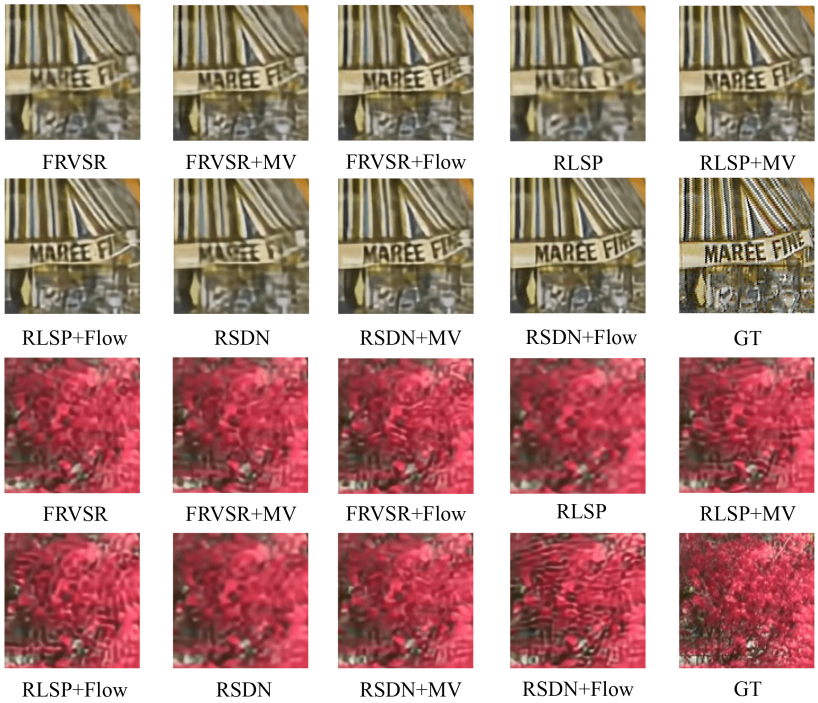

Fig. 1: Visual results on Vid4[4]

Fig. 2: Visual results on REDS4[9]

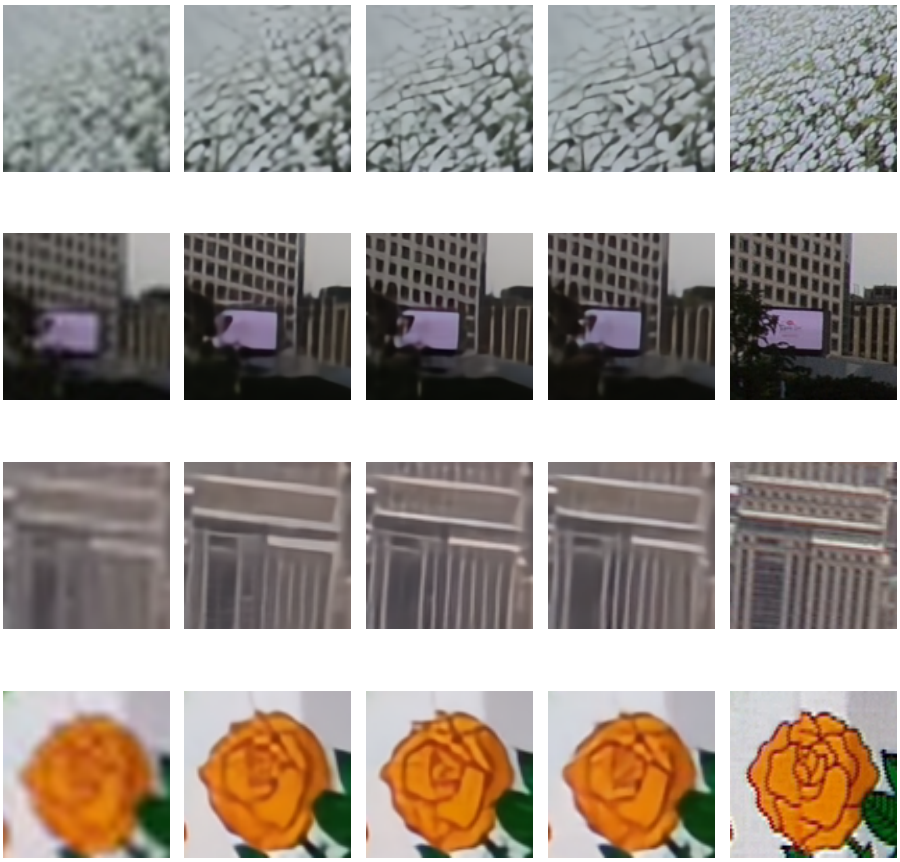

Fig. 3: Visual results of the Residual informed sparse processing on REDS4[9] and Vid4[4]

References

1. Fuoli, D., Gu, S., Timofte, R.: Efficient video super-resolution through recurrent latent space propagation. In: ICCV Workshops. pp. 3476–3485. IEEE (2019)
2. Isobe, T., Jia, X., Gu, S., Li, S., Wang, S., Tian, Q.: Video super-resolution with recurrent structure-detail network. In: ECCV (12). Lecture Notes in Computer Science, vol. 12357, pp. 645–660. Springer (2020)
3. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (Poster) (2015)
4. Liu, C., Sun, D.: A bayesian approach to adaptive video super resolution. In: CVPR. pp. 209–216. IEEE Computer Society (2011)
5. Nah, S., Timofte, R., Gu, S., Baik, S., Huo, X.: Ntire 2019 challenge on video super-resolution: Methods and results. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2019)
6. Ranjan, A., Black, M.J.: Optical flow estimation using a spatial pyramid network. In: CVPR. pp. 2720–2729. IEEE Computer Society (2017)
7. Rec, B.I.: H.264, "advanced video coding for generic audiovisual services (2005)
8. Sajjadi, M.S.M., Vemulapalli, R., Brown, M.: Frame-recurrent video super-resolution. In: CVPR. pp. 6626–6634. Computer Vision Foundation / IEEE Computer Society (2018)
9. Wang, X., Chan, K.C.K., Yu, K., Dong, C., Loy, C.C.: EDVR: video restoration with enhanced deformable convolutional networks. In: CVPR Workshops. pp. 1954–1963. Computer Vision Foundation / IEEE (2019)
10. Wang, X., Yu, K., Chan, K.C., Dong, C., Loy, C.C.: BasicSR: Open source image and video restoration toolbox. https://github.com/xinntao/BasicSR (2020)