Test-Time Optimization for Post-Processing of Compressed Videos

Hongil Kim*, Changwoo Han*, Donghyun Kim†, Sung-Chang Lim† and Seung-Won Jung*
* Department of Electrical Engineering, Korea University, Seoul
E-mail: {yourhong1, hcwoo329, swjung83}@korea.ac.kr
† Hyper-Reality Metaverse Research Laboratory, Electronics and Telecommunications Research Institute, Daejeon
E-mail: {kimddng, sclim}@etri.re.kr

***Abstract*— Lossy compression is widely used for video compression, but it often introduces compression artifacts that degrade the visual quality of compressed videos. Consequently, numerous deep learning-based methods have been developed to post-process compressed videos. However, previous post-processing models often encounter difficulties when there is a domain gap between the training and test datasets. Test-time optimization (TTO), a technique that finetunes the model during the test stage, has been considered an effective solution to address the domain gap problem. In this paper, we introduce a novel TTO method specialized for compression artifacts reduction. Specifically, we propose using image pairs available on the decoder-side, *i.e.*, the images before and after the adaptive loop filtering of the versatile video coding standard, as input and target of TTO such that the post-processing model can be adapted to the characteristics of test data. Experimental results on several baseline models and test datasets demonstrate the effectiveness of the proposed method in post-processing compressed videos.**

I. Introduction

With the increasing demand for multimedia content, video traffic occupies the majority of worldwide internet traffic. To address the exponential growth of video traffic, the Joint Video Experts Team (JVET) established the latest video coding standard called Versatile Video Coding (VVC) [1] as a successor to High Efficiency Video Coding (HEVC) [2]. These video coding standards aim to reconstruct high-quality video frames while requiring fewer bits for encoding. However, the block-based coding structure inevitably leads to compression artifacts, such as blocking and ringing artifacts, which significantly degrade image quality.

To alleviate compression artifacts, VVC adopted several filters, including deblocking filter (DBF) [3], sample adaptive offset (SAO) [4], and adaptive loop filter (ALF) [5]. DBF attenuates blocking artifacts across block boundaries primarily caused by block-wise transform and quantization. Furthermore, SAO and ALF perform adaptive adjustments to the decoded frame to reduce the difference between the decoded and uncompressed frames. Although these filters effectively reduce compression artifacts, there is still ample room for improving the quality of decoded frames. Inspired by the recent success of deep learning in low-level vision tasks [6], [7], [8], [9], [10], [11], [12], many researchers have explored convolutional neural network (CNN)-based compression artifacts reduction filters [13], [14], [15], [16], [17], [18]. CNN-based filters have demonstrated higher performance in reducing compression artifacts than handcrafted filters due to their strong ability to capture non-linear degradation in compressed frames.

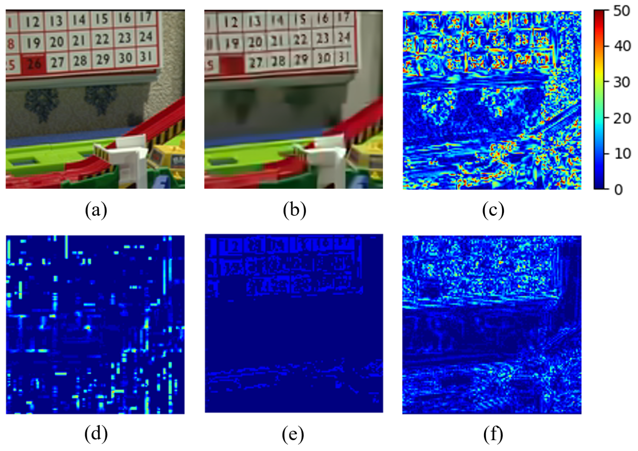

Fig. 1. Comparisons of the difference signals: (a) Original image, (b) decoded image (after ALF), (c) difference between the original and decoded images, (d) difference of before/after DBF, (e) difference of before/after SAO and (f) difference of before/after ALF. Absolute difference values are colored for visualization.

Nonetheless, CNN-based filters are inherently sensitive to the domain gap between training and test datasets. In particular, since compression artifacts depend on video coding parameters, such as the quantization parameter (QP), as well as video characteristics, it is very challenging to handle images with diverse distributions of compression artifacts using CNNs trained on specific training datasets. To this end, several methods have attempted to train individual CNNs for a set of QPs [19], [20], train a CNN using images compressed with different QPs [21], or even combine multiple CNNs [22]. However, such trained CNNs may still require more ability to handle images with different compression artifacts at the test stage.

Test-time optimization (TTO) has been considered one of the solutions for improving the performance of pre-trained CNNs for test data. Since ground-truth (GT) labels do not exist for test data, several novel ideas have been introduced in classification [23], human pose estimation [24], super-resolution [25], video frame interpolation [26], video segmentation [27], video object segmentation [28] tasks to finetune the CNN

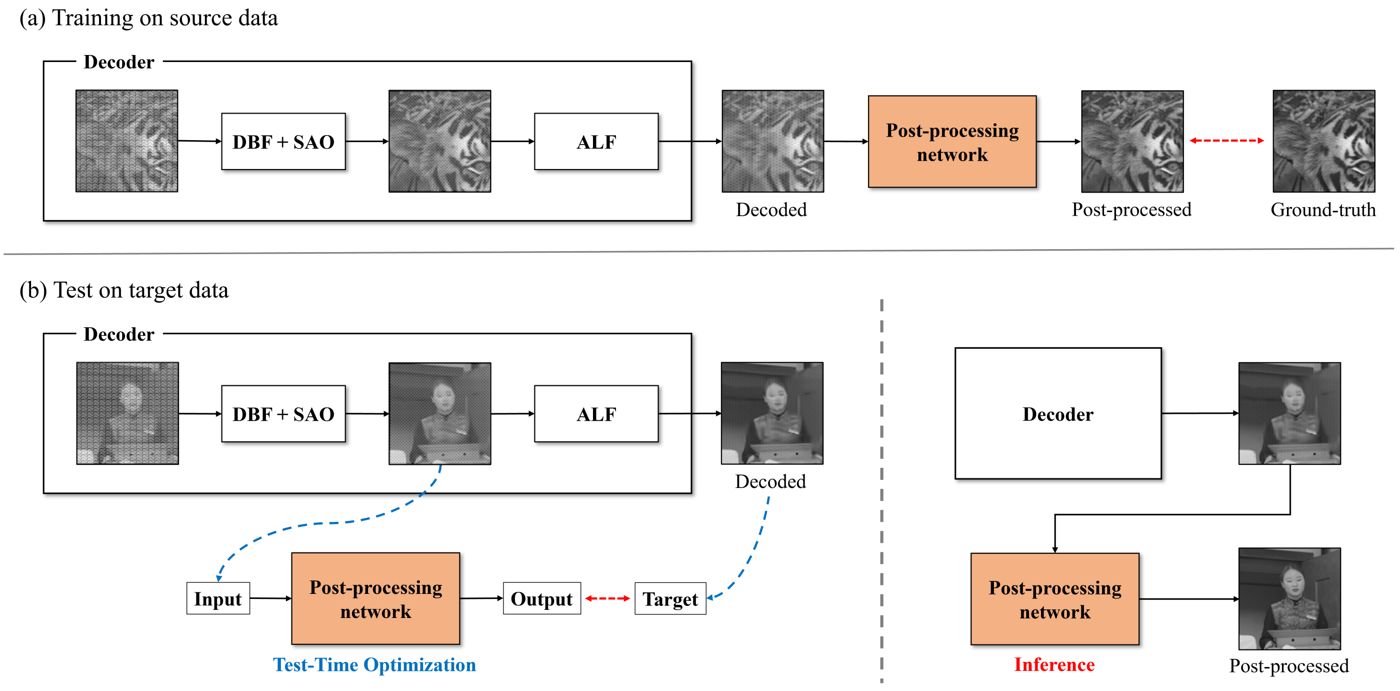

Fig. 2. The overall pipeline of the proposed method: (a) The post-processing network is trained on the source data using GT labels; (b) The post-processing network is first finetuned on the test data (left) and then applied to obtain artifacts-reduced video frames (right).

without GT labels. However, these methods [23], [24], [25], [27], [28] incorporate image generation or rely on self-training and pseudo-labeling, which demand significant computational capacity or suffer from the challenges posed by noisy pseudo-labels [29].

In this paper, we make the first attempt to apply TTO for compression artifacts reduction. In our case study of artifacts reduction in videos compressed by the VVC standard [1], we notice that the input and output of the ALF can effectively serve as the input and target for TTO of the CNN, respectively, to overcome the aforementioned challenges. Specifically, using a few input and output image pairs of the ALF, we update the parameters of the CNN for a few steps such that the CNN can better handle test data. From extensive experimental results, we show the effectiveness of the proposed TTO method.

II. Proposed Method

A. Motivation

Assume that we have a post-processing CNN trained on uncompressed images and their corresponding compressed ones as the GT labels and input, respectively. Since uncompressed images do not exist on the decoder side, we need surrogate pairs of input and target images for TTO of the post-processing CNN. Fortunately, the VVC standard, as well as many other off-the-shelf video codecs, perform several in-loop filtering operations during decoding; thus, the input and output images of the in-loop filters are still available on the decoder side. Among the three adopted in-loop filters of the VVC, *i.e.*, DBF [3], SAO [4], and ALF [5], the difference between the input and output images of ALF best mimics the one between the uncompressed and compressed images, as illustrated in Fig. 1. We thus propose to use this valuable input and output image pair of ALF as the input and target images for TTO, respectively, to adapt the post-processing CNN on test data.

B. Test-Time Optimization

Fig. 2(a) shows the training procedure. Let f_θ denote a post-processing CNN parameterized by θ, which is trained to minimize the following:

$$\theta^* = \arg\min_{\theta} \sum_i \left\| f_\theta\left(\hat{I}_i\right) - I_i \right\|, \tag{1}$$

where $\hat{I}_i$ and I_i are the i-th pair of decoded and uncompressed images, and $\|\cdot\|$ measures the distance between the network output and target uncompressed images. Given the training dataset $\mathcal{I} = \left\{\hat{I}_i, I_i\right\}_{i=1}^{N_T}$, where N_T is the number of images, we can obtain a set of optimized parameters θ^* that work well on average for the training images. To make the post-processing CNN more effective on test data with diverse distributions, we propose an algorithm that can adapt network parameters to test data.

Fig. 2(b) illustrates the proposed TTO procedure. Let $\hat{J}_i$ denote the i-th decoded video frame on the decoder side. Since its corresponding GT image J_i is not available, we instead extract an image before ALF, denoted as $\tilde{J}_i$, and use it as input for TTO of the post-processing CNN. Specifically, the network parameter is first initialized as θ^* in (1) and then finetuned on test data as follows:

$$\theta^{**} = \arg\min_{\theta} \sum_{i \in \Upsilon_i} \left\| f_\theta\left(\tilde{J}_i\right) - \hat{J}_i \right\|, \tag{2}$$

where Υ_i is a set of frame indices used for TTO. Since video characteristics change over time, TTO has to be applied continuously during decoding. In our implementation, we apply TTO at the group of pictures (GOP)-level, a set of 32 consecutive frames in VVC [1]. In each frame, ALF is selectively applied according to the rate-distortion cost [1]. Therefore, we define Υ_i as a set of indices of frames belonging to the same GOP with $\hat{J}_i$ that ALF processes. Once a GOP-level TTO is completed, all frames in the GOP are post-processed by $f_{\theta^{**}}$ to have artifacts-reduced video frames, as illustrated in Fig. 2(b). If no frame is processed by ALF in a GOP, TTO is not applied.

C. Implementation Details

Considering that the residual block (ResBlock) is one of the most common building blocks of recent image restoration models [10], [30], [31], [32], we design a ResBlock-based baseline network (denoted as M1), as shown in Fig. 3(a) for the performance verification of the proposed TTO. M1 contains 24 residual blocks, each composed of two 3×3 convolutional layers and a PReLU activation layer. The global skip connection is also included to ease the training. The network complexity of M1 is 58,600 multiply-accumulate operations per pixel (MAC/pixel). In addition, the UNet architecture is widely used in image restoration tasks, such as deblurring [33], [34] and denoising [35], [36], [37]; thus, we design a baseline network based on the UNet architecture (denoted as M2), as shown in Fig. 3(b). M2 consists of three levels of convolution blocks, down-blocks, and up-blocks. Each of these blocks is composed of 3×3 convolutional layers and PReLU activation functions. Additionally, each down-block and up-block incorporates a 2×2 convolution and transposed convolution layer, respectively. Skip connections are included between each level's down-block and up-block. The complexity of M2 is 104.3 kMAC/pixel, and the numbers of network parameters of M1 and M2 are 1.78M and 4.92M, respectively.

Since the proposed TTO is designed to alleviate the performance degradation of the post-processing CNN caused by the domain gap between training and test data, the required number of parameter update steps of (2) can differ for each video sequence or even for each GOP of the same video sequence. To this end, we devise a simple but effective strategy that terminates the parameter update if the following condition is satisfied:

$$\frac{1}{|\Upsilon_i|} \sum_{i \in \Upsilon_i} \left\| f_{\theta_{t+1}}\left(\tilde{J}_i\right) - f_{\theta_t}\left(\tilde{J}_i\right) \right\| \; < \; T_h \tag{3}$$

where $|\cdot|$ represents the cardinality of a set, θ_t represents a set of network parameters after the t-th update, and T_h is a threshold, empirically chosen as 0.0003. In other words, TTO is terminated if it leads to a marginal change after a certain update step. We also terminate TTO if the number of updates reaches the threshold N_h, empirically chosen as 6.

TABLE I
PERFORMANCE EVALUATION IN BD-RATE (%) UNDER RA CONFIGURATION FOR THE JVET-CTC TEST SEQUENCES.

Class	Sequence	M1	M1+TTO	M2	M2+TTO
B	MarketPlace	-3.09	-3.25	-1.13	-1.23
	RitualDance	-3.93	-4.13	-1.43	-1.49
	Cactus	-2.50	-2.83	-0.28	-0.66
	BasketballDrive	-3.66	-3.88	-0.47	-0.95
	BQTerrace	-1.05	-2.04	+0.62	-0.07
C	BasketballDrill	-3.88	-4.40	-0.70	-1.01
	BQMall	-4.45	-5.00	-1.01	-1.35
	PartyScene	-4.20	-4.48	-0.57	-0.73
	RaceHorsesC	-2.41	-2.75	-0.50	-0.53
D	BasketballPass	-5.89	-6.02	-1.66	-1.83
	BQSquare	-9.44	-9.91	-1.98	-2.33
	BlowingBubbles	-4.47	-4.62	-0.83	-0.99
	RaceHorses	-4.79	-4.87	-1.40	-1.57
Total	B Class	-2.85	-3.23	-0.54	-0.88
	C Class	-3.76	-4.16	-0.69	-0.90
	D Class	-6.15	-6.35	-1.47	-1.68
	Average	**-4.14**	**-4.48**	**-0.82**	**-1.13**

Inspired by [27], [38], [39], [40], we aim to preserve domain-agnostic information of the pre-trained model while updating only domain-specific features. To achieve this, instead of updating the parameters of all layers in the post-processing CNN, we opt to freeze earlier layers to retain low-level features as well as the last-stage image reconstruction layers. Consequently, the proposed TTO aims to learn high-level content-specific features by finetuning only the second half of the ResBlocks in the M1 model, as well as the Up-block section in the M2 model. Fig. 3 indicates the trainable layers by TTO. Last, since ALF obtains filter coefficients by minimizing the L2 loss between the original and restored images [41], we use the L2 loss in (2)-(3).

III. EXPERIMENTAL RESULTS

For performance evaluation, we used videos compressed by the VVC reference software VTM-11.0 using the random access (RA) configuration and five QPs = $\{22, 27, 32, 37, 42\}$. The BVI-DVC dataset [42] and the class B-D JVET-CTC test sequences [43] were used as training and test data, respectively. The BD-rate [44] of the Y channel was used as a performance metric, which represents the percentage of the average bit-saving over the VTM-11.0 baseline.

During the training stage, the baseline network was trained for a total of 100 epochs via (1) using the Adam optimizer with an initial learning rate of 0.0001. The learning rate decayed at every 20 epochs after 30 epochs with a decaying rate of 0.1. We trained five networks separately for the five chosen QPs. In the test stage, TTO was applied at the GOP-level using the ALF-applied frames. From the input and output image pairs of ALF, we randomly cropped three patches with a size of

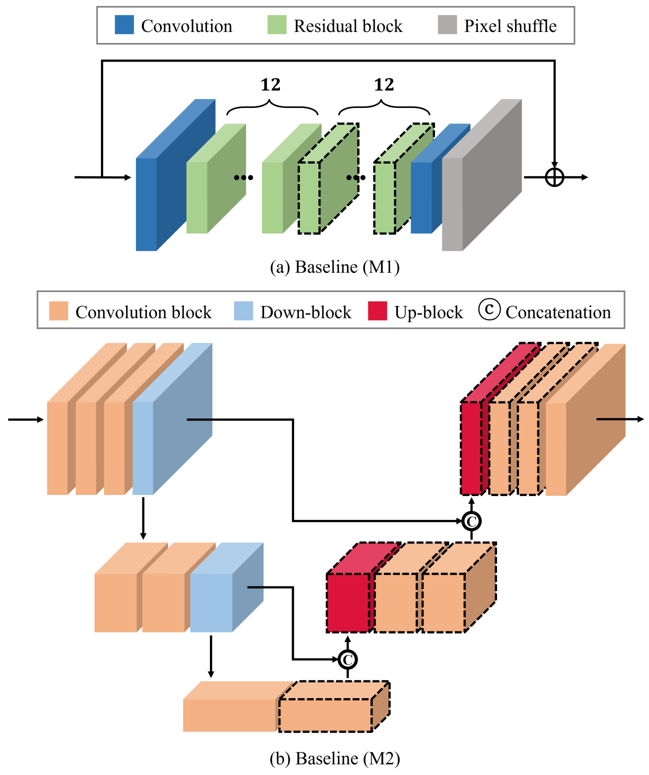

Fig. 3. Post-processing network architectures used in our experiments: (a) ResBlock-based baseline (M1) and (b) UNet-based baseline (M2). Only the dotted blocks are updated during TTO.

TABLE II
EXPERIMENTAL RESULTS ON THE NUMBER OF NETWORK UPDATE STEPS. THE PERFORMANCE WAS EVALUATED AS THE AVERAGE BD-RATE AND THE AVERAGE PROCESSING TIME PER GOP FOR THE CLASS B-D TEST SEQUENCES.

M1	N_h					
	0	2	4	6	8	10
BD-rate (%)	-4.14	-4.30	-4.41	-4.48	-4.49	-4.49
Time (s)	4.3	6.0	8.5	10.8	13.2	15.7

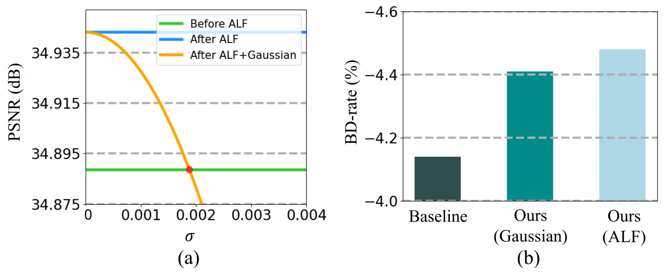

Fig. 4. Comparison of the TTO results obtained using different training datasets: (a) The PSNRs between the degraded and uncompressed (original) frames, where we add the Gaussian noise to the decoded (after ALF) frames of the JVET-CTC test sequences and obtain the noise standard deviation that results in the similar average PSNR of the frames before ALF, which is 0.0018 at QP=32; (b) Experimental results in BD-rates, where the Gaussian noise-added frames (Ours, Gaussian) and the frames before ALF (Ours, ALF) are used as input for TTO, respectively. The noise standard deviations are chosen to match the average PSNRs of the frames before ALF for all five tested QPs. Both TTO results show better performance than M1 while using the input and output image pairs of ALF for TTO yields a higher BD-rate gain.

256×256 and trained the network using (2) with the Adam optimizer and a fixed learning rate of 10e-6.

Table I shows the BD-rate results. Compared to the VTM-11.0 anchor, our post-processing networks, *i.e.*, M1 and M2, led to an improvement in the BD-rate of 4.14% and 0.82%, respectively. The proposed TTO resulted in 4.48% (M1+TTO) and 1.13% (M2+TTO), indicating its effectiveness. Note that TTO decreased the BD-rates for all test sequences, with a maximum gain of 0.99% for BQTerrace and a minimum gain of 0.08% for RaceHorses in M1 and a maximum gain of 0.69% for BQTerrace and a minimum gain of 0.03% for RaceHorsesC in M2.

Next, we obtained the experimental results using different values of N_h, as shown in Table II. As N_h increased, the average processing time of TTO per GOP increased approximately linearly. When N_h = 0, the pretrained post-processing model was applied without TTO. When $N_h \neq 0$, the processing time for TTO and the inference time after TTO were added together. Based on our experiments, with N_h = 6, the processing time ranged from a minimum of 4.3 seconds to a maximum of 10.8 seconds, depending on the test sequence characteristics. Note that the BD-rates did not continuously increase as the number of network updates increased. Since not a real image pair was used for TTO, but a surrogate image pair, we found performance saturation after certain update steps and thus chose N_h = 6 as the default value for both baselines, considering the increased processing time by TTO.

Lastly, to verify the effectiveness of using the image pairs before and after ALF as training images for TTO, we added Gaussian noise to the decoded image $\hat{J}_i$ and obtained its noisy version, denoted as $\overline{J}_i$. For a fair comparison, the standard deviation of the Gaussian noise was determined to make the average PSNR between $\hat{J}_i$ and $\overline{J}_i$ and that between $\hat{J}_i$ and $\tilde{J}_i$ similar, as shown in Fig. 4(a). For this experiment, the proposed TTO was applied while replacing $\tilde{J}_i$ by $\overline{J}_i$ in (2). As shown in Fig. 4(b), the proposed TTO enables improved performance in both cases compared to M1, indicating the effectiveness of using test-time data for finetuning. In addition, the use of the frames before and after ALF was found to be better than the use of noisier-to-noisy image pairs obtained by Gaussian noise addition [45], supporting the effectiveness of using compression-specific distortion for TTO purposes. More results can be found on the project page[1].

IV. Conclusion

This paper introduced a new use case of TTO for compression artifacts reduction. The proposed TTO method uses the signals before and after the in-loop filter of VVC, which are also available on the decoder side. By finetuning the post-processing model at the test stage using these signals as training data, we showed that compression artifacts could be

[1]https://yourhong1.github.io/PP-TTO/

reduced for test video sequences with different characteristics. We believe that our method provides an effective way of optimizing pre-trained models during the test stage, paving the way for further developments in artifacts reduction and quality improvement of compressed videos. Moreover, while our approach has focused on post-processing filters, the potential to extend TTO to other stages within the video codec, such as the in-loop filter or scaling techniques, presents a valuable direction for future research. By applying TTO to these components, we could further enhance overall BD-rate performance. This flexibility suggests that TTO could serve as a versatile tool for solving various challenges in video compression.

Acknowledgment

This work was supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2017-0-00072, Development of Audio/Video Coding and Light Field Media Fundamental Technologies for Ultra Realistic Tera-media

References

[1] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, "Overview of the versatile video coding (VVC) standard and its applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3736–3764, 2021.

[2] 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, 2012.

[3] P. List, A. Joch, J. Lainema, G. Bjontegaard, and M. Karczewicz, "Adaptive deblocking filter," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 614–619, 2003.

[4] C.-M. Fu, E. Alshina, A. Alshin, Y.-W. Huang, C.-Y. Chen, C.-Y. Tsai, C.-W. Hsu, S.-M. Lei, J.-H. Park, and W.-J. Han, "Sample adaptive offset in the hevc standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1755–1764, 2012.

[5] C.-Y. Tsai, C.-Y. Chen, T. Yamakage, I. S. Chong, Y.-W. Huang, C.-M. Fu, T. Itoh, T. Watanabe, T. Chujoh, M. Karczewicz, and S.-M. Lei, "Adaptive loop filtering for video coding," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 934–945, 2013.

[6] 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, 2016.

[7] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, July 2017.

[8] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. pattern Recognit.*, June 2018, pp. 2472–2481.

[9] 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. Vision*. Springer, 2018, pp. 286–301.

[10] D.-W. Kim, J. R. Chung, and S.-W. Jung, "GRDN: Grouped residual dense network for real image denoising and GAN-based real-world noise modeling," in *Proc. IEEE/CVF Conf. Comput. Vis. pattern Recognit. Workshops*, 2019, pp. 2086–2094.

[11] M. Zhao, G. Cao, X. Huang, and L. Yang, "Hybrid transformer-cnn for real image denoising," *IEEE Signal Process. Lett.*, vol. 29, pp. 1252–1256, 2022.

[12] J. Wu, Y. Wang, and X. Zhang, "Lightweight asymmetric convolutional distillation network for single image super-resolution," *IEEE Signal Process. Lett.*, vol. 30, pp. 733–737, 2023.

[13] F. Zhang, C. Feng, and D. R. Bull, "Enhancing VVC through CNN-based post-processing," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2020, pp. 1–6.

[14] F. Zhang, D. Ma, C. Feng, and D. R. Bull, "Video compression with CNN-based postprocessing," *IEEE MultiMedia*, vol. 28, no. 4, pp. 74–83, 2021.

[15] W.-S. Park and M. Kim, "CNN-based in-loop filtering for coding efficiency improvement," in *Proc. IEEE Image, Video, and Multidimensional Signal Processing Workshop*, 2016, pp. 1–5.

[16] Y. Zhang, T. Shen, X. Ji, Y. Zhang, R. Xiong, and Q. Dai, "Residual highway convolutional neural networks for in-loop filtering in HEVC," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 3827–3841, 2018.

[17] D. Luo, M. Ye, S. Li, and X. Li, "Coarse-to-fine spatio-temporal information fusion for compressed video quality enhancement," *IEEE Signal Process. Lett.*, vol. 29, pp. 543–547, 2022.

[18] Z. Wang, M. Ye, S. Li, and X. Li, "Multi-frame compressed video quality enhancement by spatio-temporal information balance," *IEEE Signal Process. Lett.*, vol. 30, pp. 105–109, 2023.

[19] C. Liu, H. Sun, J. Katto, X. Zeng, and Y. Fan, "A convolutional neural network-based low complexity filter," *arXiv preprint arXiv:2009.02733*, 2020.

[20] J. Deng, L. Wang, S. Pu, and C. Zhuo, "Spatio-temporal deformable convolution for compressed video quality enhancement," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 07, 2020, pp. 10 696–10 703.

[21] P. Svoboda, M. Hradis, D. Barina, and P. Zemcik, "Compression artifacts removal using convolutional neural networks," *arXiv preprint arXiv:1605.00366*, 2016.

[22] S.-J. Cho, J. R. Chung, S.-W. Kim, S.-W. Jung, and S.-J. Ko, "Compression artifacts reduction using fusion of multiple restoration networks," *IEEE Access*, vol. 9, pp. 66 176–66 187, 2021.

[23] W. Ma, C. Chen, S. Zheng, J. Qin, H. Zhang, and Q. Dou, "Test-time adaptation with calibration of medical image classification nets for label distribution shift," in *Proc. Int. Conf. Med. image Comput. Comput.-Assist. Intervention*, 2022, pp. 313–323.

[24] J. Zhang, X. Nie, and J. Feng, "Inference stage optimization for cross-scenario 3d human pose estimation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 2408–2419.

[25] M. S. Rad, T. Yu, B. Bozorgtabar, and J.-P. Thiran, "Test-time adaptation for super-resolution: You only need to overfit on a few more images," in *Proc. Int. Conf. Comput. Vision Workshops*, October 2021, pp. 1845–1854.

[26] M. Choi, J. Choi, S. Baik, T. H. Kim, and K. M. Lee, "Test-time adaptation for video frame interpolation via meta-learning," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 44, no. 12, pp. 9615–9628, 2021.

[27] Q. Wang, O. Fink, L. Van Gool, and D. Dai, "Continual test-time domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. pattern Recognit.*, 2022, pp. 7201–7211.

[28] F. Azimi, S. Palacio, F. Raue, J. Hees, L. Bertinetto, and A. Dengel, "Self-supervised test-time adaptation on video data," in *Proc. Winter Conf. Appl. Comput. Vision*, 2022, pp. 3439–3448.

[29] D. Chen, D. Wang, T. Darrell, and S. Ebrahimi, "Contrastive test-time adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 295–305.

[30] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image restoration," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 43, no. 7, pp. 2480–2495, 2020.

[31] Y. Zhao, Y. Xu, Q. Yan, D. Yang, X. Wang, and L.-M. Po, "D2hnet: Joint denoising and deblurring with hierarchical network for robust night image restoration," in *Proc. Eur. Conf. Comput. Vision*. Springer, 2022, pp. 91–110.

[32] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, M.-H. Yang, and L. Shao, "Learning enriched features for fast image restoration and enhancement," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 45, no. 2, pp. 1934–1948, 2022.

[33] S.-J. Cho, S.-W. Ji, J.-P. Hong, S.-W. Jung, and S.-J. Ko, "Rethinking coarse-to-fine approach in single image deblurring," in *Proc. Int. Conf. Comput. Vision*, 2021, pp. 4641–4650.

[34] L. Chen, X. Chu, X. Zhang, and J. Sun, "Simple baselines for image restoration," in *Proc. Eur. Conf. Comput. Vision*. Springer, 2022, pp. 17–33.

[35] J. Gurrola-Ramos, O. Dalmau, and T. E. Alarcón, "A residual dense u-net neural network for image denoising," *IEEE Access*, vol. 9, pp. 31 742–31 754, 2021.

[36] F. Jia, W. H. Wong, and T. Zeng, "DDUNet: Dense dense u-net with applications in image denoising," in *Proc. Int. Conf. Comput. Vision*, 2021, pp. 354–364.

[37] C.-M. Fan, T.-J. Liu, and K.-H. Liu, “SUNet: Swin transformer UNet for image denoising,” in *Proc. IEEE Int. Symp. Circuits Syst.*, 2022, pp. 2333–2337.

[38] B. Neyshabur, H. Sedghi, and C. Zhang, “What is being transferred in transfer learning?” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 512–523.

[39] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Proc. Eur. Conf. Comput. Vision*. Springer, 2014, pp. 818–833.

[40] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.

[41] F. Jin, P. Fieguth, L. Winger, and E. Jernigan, “Adaptive wiener filtering of noisy images and image sequences,” in *Proc. IEEE Int. Conf. Image Process.*, vol. 3, 2003, pp. 349–352.

[42] D. Ma, F. Zhang, and D. R. Bull, “Bvi-dvc: A training database for deep video compression,” *IEEE Trans. Multimedia*, vol. 24, pp. 3847–3858, 2021.

[43] F. Bossen, J. Boyce, X. Li, V. Seregin, and K. Sühring, “VTM common test conditions and software reference configurations for sdr video (JVET-T2010),” *Joint Video Experts Team*, 2020.

[44] G. Bjontegaard, “Calculation of average PSNR differences between RD-curves,” *ITU SG16 Doc. VCEG-M33*, 2001.

[45] N. Moran, D. Schmidt, Y. Zhong, and P. Coady, “Noisier2noise: Learning to denoise from unpaired noisy data,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12 064–12 072.