LOW-COMPLEXITY MULTI-MODEL CNN IN-LOOP FILTER FOR AVS3

Shen Wang[⋆] *Yibing Fu*[⋆] *Chen Zhu*[⋆] *Li Song*[⋆,†] *Wenjun Zhang*[⋆]

[⋆] Institute of Image Communication and Network Engineering, Shanghai Jiao Tong University
[†] MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
E-mail:wangshen22206, song_li@sjtu.edu.cn(correspinding author)

ABSTRACT

Convolutional neural network (CNN) has demonstrated powerful capabilities in many image/video processing tasks. In this paper, a low-complexity multi-model CNN in-loop filtering scheme is proposed for AVS3. Firstly, we carefully choose simplified ResNet as the lightweight single model of our proposed network. Subsequently, based on the selected single model, the multi-model iterative training framework is proposed to train a multi-model filter, where the network depth and the number of multi-models are customized for different ranges of bit rate to achieve the trade-off between model performance and computational complexity. Experimental results show that our method achieves on average 6.06% BD-rate reduction on Y component under all intra configuration. Compared to other CNN filters with comparable performance, our proposed multi-model filter can significantly reduce the decoder complexity, and the experimental results indicate that the decoding time can be saved by 26.6% on average.

Index Terms— Audio Video Coding standard (AVS3), In-Loop Filter, Multi-Model, CNN

1. INTRODUCTION

The third generation of Audio Video Coding Standard (AVS3) [1] as a new video coding standard is developed by the AVS working group in China. Compared with the previous generation, AVS3 has higher compression efficiency. Block-based hybrid coding framework of AVS3 includes a range of tools for affine motion compensation, MMVD etc. However, distortions such as blocking, ringing and blurring artifacts still exist in the images/videos reconstructed by advanced codecs, which are more perceptually obvious and unsatisfactory at low bit rates. In addition to the impact of visual quality, inter-prediction efficiency is also affected by these artifacts.

In order to mitigate the video coding distortion, various in-loop filtering algorithms have been proposed. The main idea of these filtering algorithms is to enhance the quality of each encoded frame. AVS3 standard adopts three types of in-loop filters, including Deblocking filter (DBF) [2], Sample adaptive offset (SAO) [3] filter, and Adaptive loop filter (ALF) [4]. The DBF is firstly used to reduce the block effect in the compressed video; the SAO filter can handle the problem of ringing artifacts by superimposing an adaptive offset on each sample point; the ALF is based on the Wiener filtering algorithm to further reduce the overall pixel-level error.

In recent years, convolutional neural networks (CNN) have achieved great success in computer vision tasks. CNN has also shown great potential in the field of video compression [5]. Among multiple modules in the hybrid framework, In-loop filtering for image recovery is most suitable for CNN modeling. Many deep-learning-based loop filtering methods have been proposed [6] [7], which significantly improves the coding efficiency. These methods generally build a complex large-scale CNN model by learning the spatial correlation in video frames. However, the recovery capability of a single CNN model is not sufficient to accommodate the wide variety of video content and recover all kinds of distortion. It is a challenging task to train a single model on datasets from the realistic production environment because it includes various potential distributions. This type of problem is well suited to being solved by ensemble model [8]. The essential idea of ensemble model is to improve the final performance of it by combining the results of multiple single models effectively. The ensemble model algorithm provides a solution so that we can train these models and make synthetic predictions which achieves better accuracy than each individual model.

In this paper, we proposed a low-complexity multi-model CNN in-loop filter to improve the objective quality of the reconstructed video. Firstly, a variety of lightweight single models are explored in comparison, and simplified ResNet is selected as the single model. Then a multi-model iterative training mechanism is applied to train a multi-model CNN network, and the number of multi-models and network depth are optimized. In addition, we add the design of frame-level flag which helps to achieve overall performance optimization. Finally, The low-complexity multi-model CNN loop filter is incorporated into AVS3 reference software HPM7.1.

2. PROPOSED METHOD

In this section, we compare the performance of different single models and determine the backbone of the single model and propose a multi-model iterative training framework, while further investigating the effect of network depth

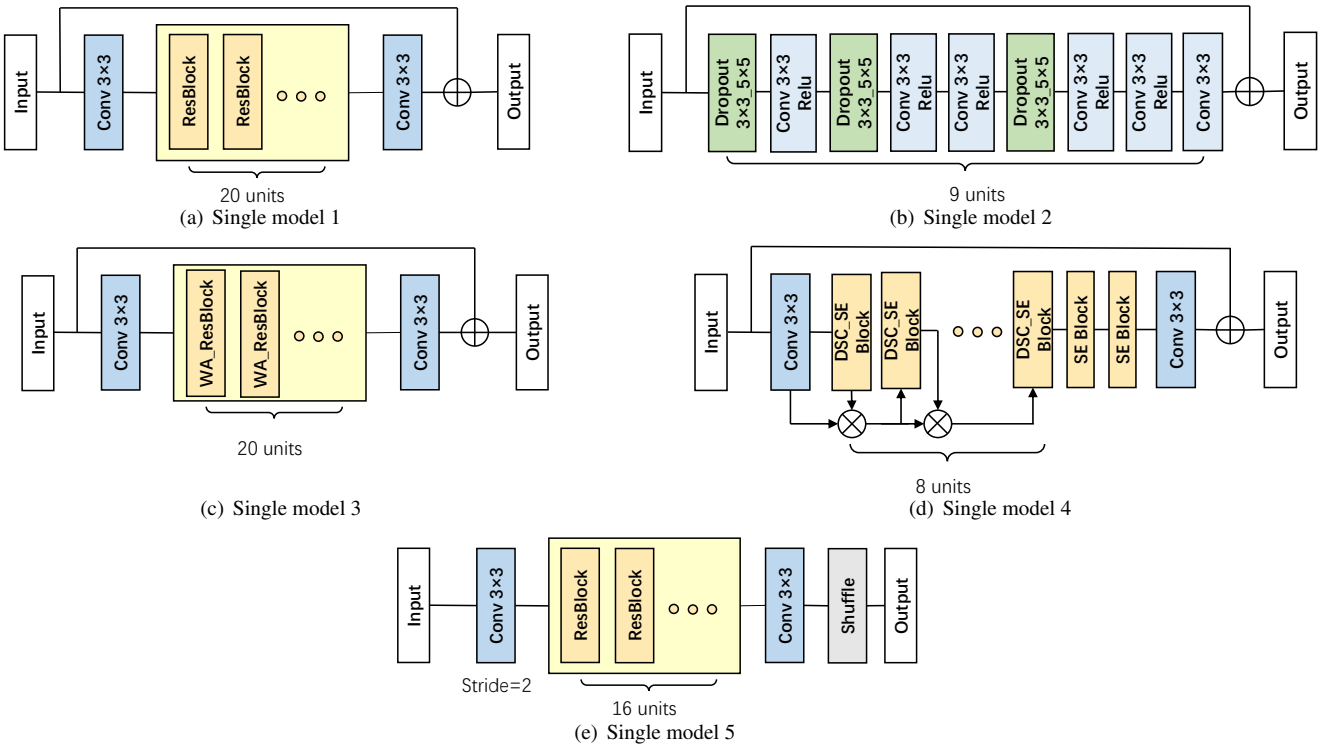

Fig. 1: Illustration of five single model structures

and the number of multi-models.

2.1. Single CNN model of Multi-model network

There are diverse CNN architectures to be applied for loop filtering. Here five lightweight models are chosen as the candidates of single model. We investigate the main features of each architecture and compare their performance.

These model structures are briefly described here. Motivated by AVS3 proposal M5129 [6], we design the single model 1, which consists of 20 stacked residual blocks with 32 channels. The number of channels is half that of the proposal M5129. The single model 2 is characterized by convolution kernels of different sizes and the addition of a dropout layer, which is from content-aware CNN single model [9]. The Wide-Activated mechanism is added to residual block in the single model 3 [7]. In the single model 4, the basic residual block is replaced by the depthwise separable convolution (DSC) [10] while SE-Block is added to assign different weights to each channel. The single model 5 uses a downsampling convolutional layer with a step size of 2, and finally upsampling by shuffle operation. The single model 4 and 5 refer to the JVET proposal T0069 [11] and U0068 [12]. The five single model structures are shown in the figure 1.

To make a fair comparison, five lightweight single models were trained under the same training conditions and dataset DIV2K [13]. The pre-trained models are used to filter the first ten reconstructed images with QP37 in the DIV2K validation set. The training details and testing conditions are shown in Chapter III.A. The PSNR results after filtering and model complexities are shown in Table 1. We can see that the highest PSNR in 5 single models is from the single model 1, while MACs/pixel (Multiply–Accumulate Operations per pixel) of the single model 1 is only a quarter of that by the M5129 proposal scheme, which indicates the computational complexity of the model. So the single model 1 is chosen as the single CNN model structure of the multi-model network.

Table 1: PSNR results after five single models filtering and model complexity

model number	PSNR	params	MACs/pixel
recon	36.23	/	/
single model 1	**36.79**	**370,529**	**0.37M**
single model 2	36.72	362,753	0.36M
single model 3	36.76	368,469	0.37M
single model 4	36.62	227,625	0.22M
single model 5	36.77	4,728,596	1.18M
M5129	/	1,515,265	1.51M

2.2. Multi-model CNN In-loop filtering

Due to the different distortion levels at different ranges of bit rate and the diversity in video contents, it is common in many pieces of research to use a large number of parameters to train a generic model, which may lead to high complexity in the decoder. Motivated by ensemble model, it is reasonable to replace complex models with multiple lightweight single models. After determining the single CNN model structure, we proposed the multi-model iterative training mechanism to train multiple single models. Different single models are targeted for different kinds of video content. The iterative pro-

Algorithm 1 Mechanism of Multi-model iterative training

Input:

pre-trained Single CNN model, training samples (x_i, y_i) generated from DIV2K, multi-models number N_i, iteration time K.

Output:

N_i fine-tuned models;

1: **for** QP in {27,32,37,45} **do**
2: Divide the training samples (x_i, y_i) generated from DIV2K into N_i equal parts randomly;
3: Training the Single CNN models using N_i aliquots of the dataset respectively, then yields N_i initialized Single CNN models;
4: Filter each sample using the N_i single models, and label each sample with index $Idx(x_i)$ to represent single model with the best filtering effect (highest PSNR);
5: Generate N_i folds of training samples by clustering samples with same index $Idx(x_i)$;
6: Fine-tune and update N_i models with corresponding folds of clustered training samples;
7: Repeat steps four to six K times to finally generate the multi-model CNN filter we proposed.
8: **end for**

cess is shown in Algorithm 1.

Since the degree of image distortion varies at different Quantizer Parameter (QP), we trained the corresponding multi-model networks on four QP {27,32,37,45}. Assume that network of each QP contains N_1, N_2, N_3 and N_4 single models respectively. The multi-model CNN network structure is shown in figure 2. At the encoder, for a CTU of size 128×128, the corresponding multi-model network model is first selected according to the QP, then each of N_i single models is used to filter and Rate Distortion Optimization (RDO) is performed to select the single model with the highest PSNR. The selected single model index is written into the bitstream. At the decoder, according to the transmitted model index, the corresponding single model is selected to filter the current CTU.

$$J_i = D_i + \lambda * R_i \tag{1}$$

In addition, we design a frame-level flag to control the filter on/off by equation (1). If the frame-level flag is off, each CTU is not filtered by the model, and no more model indexes are transmitted. Conversely, a syntax element is required to represent the model index for each CTU with frame-level flags enabled. D_1 and D_2 indicate the distortion with the scheme proposed enabled and disabled respectively while R_1 and R_2 denote bits transmitted to the decoder. R_1 is set to the number of CTUs filtered by the model and R_2 is equal to zero. Frame flag is turned on if $J_1 < J_2$ and vice versa.

Initially, the initial single models of four QPs are all composed of 20 stacked residual blocks with 32 channels and the multi-model structure of each QP consists of 8 single models ($N_1 = N_2 = N_3 = N_4 = 8$). Iteration time K is set

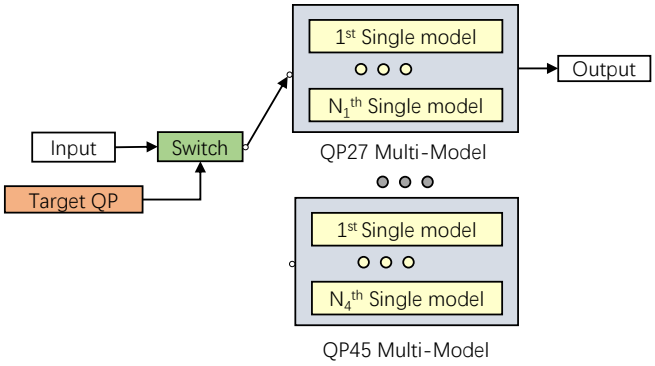

Fig. 2: Illustration of Multi-model CNN structure

to be 5. Considering that the coded frames are close to the ground truth at small QP, which makes the network easy to learn, the network depth and the number of multi-models can be reduced correspondingly. Through experiments, Multi4_Res12_32 and Multi4_Res16_32 are determined as the multi-model of QP27 and QP32 respectively, the model parameters of other QPs remain unchanged. The results of filter response compared with the initialized model are shown in the Table 2. In the Table 2, ResM_N represents a stack of M residual blocks with N channels, and MultiX_ResM_N represents a multi-model filter consisting of X ResM_N single models. The details are described below.

Table 2: PSNR increment of multi-model number and network depth optimization

Model\QP	27	32	37	45
Res20_32	0.232	0.358	0.441	0.500
Res12_32	0.267	/	/	/
Res16_32	/	0.345	/	/
Multi8_Res20_32	0.316	0.401	0.491	0.564
Multi4_Res12_32	0.307	/	/	/
Multi4_Res16_32	/	0.403	/	/

- The multi-model's filtering effect is significantly better than the single model's, indicating that the multi-model structure is effective;
- For QP 27, the filtering effect of Res12_32 is better than that of Res20_32, indicating that when the quantization distortion is slight, using a network model of more layers may on the contrary cause a worse filtering effect. If we further reduce the number of multi-models and use Multi4_Res12_32 structure, there is a slight decrease in performance compared with Multi8_Res20_32, but the complexity is greatly reduced. Therefore for QP 27, we select Multi4_Res12_32 as the multi-model structure.
- For QP 32, the use of Multi4_Res16_32 structure can significantly reduce network complexity and the storage space for model, while performance can be considered unchanged. For QP 32, Multi4_Res16_32 is selected as the multi-model structure.
- For QP 37 and 45, using Multi8_Res20_32 structure as the Multi-model structure.

Table 3: BD-rate saving and decoding time saving under AI configration on HPM7.1

Class	Sequence	BD-rate		T_{dec} / $T_{dec(HPM7.1)}$		ΔT_{dec}
		M5129	Proposed	M5129	Proposed	(Propsed vs. M5129)
UHD4K	Campfire	-4.78%	-5.04%	6775.75%	4596.02%	-32.17%
	DaylightRoad2	-7.73%	-7.43%	7533.69%	5336.96%	-29.16%
	ParkRunning3	-4.49%	-4.48%	6961.78%	4649.33%	-33.22%
	Tango2	-6.04%	-5.72%	7266.46%	4872.40%	-32.95%
	Average	**-5.76%**	**-5.67%**	7122.32%	4852.53%	**-31.87%**
1080p	BasketballDrive	-5.97%	-5.71%	8490.19%	6774.21%	-20.21%
	Cactus	-6.00%	-5.70%	6132.10%	4631.87%	-24.47%
	MarketPlace	-4.78%	-4.66%	8524.85%	6078.61%	-28.70%
	RitualDance	-10.46%	-10.25%	9615.68%	6955.78%	-27.66%
	Average	**-6.80%**	**-6.58%**	8009.19%	5978.30%	**-25.36%**
720p	City	-3.87%	-3.62%	6485.23%	4708.44%	-27.40%
	Crew	-3.88%	-3.73%	8536.95%	6192.96%	-27.46%
	Vidyo1	-9.50%	-9.27%	7303.61%	6184.04%	-15.33%
	Vidyo3	-7.47%	-7.11%	10683.80%	8046.13%	-24.69%
	Average	**-6.18%**	**-5.93%**	7997.27%	6066.95%	**-24.14%**
Average		**-6.25%**	**-6.06%**	7709.59%	5632.59%	**-26.94%**

3. EXPERIMENT RESULT

3.1. Training Details And Testing Conditions

The training dataset includes 800 high-resolution images from DIV2K [13]. All these video sequences are compressed by HPM7.1 under AI configuration using QP values of {27,32,37,45}. All our models are implemented based on Python3.6, using pytorch1.2 backend.

We utilize the Adam optimizer to train our network. The loss function of our proposed network is mean squared error (MSE) between the reconstructed image and the original image. Learning rate is 1e-4 for all models initially and is divided by half every 4 epochs. Original and reconstructed images from training dataset are both divided into 128 × 128 non-overlapping patches, and sent into the network.

To fully evaluate the compression efficiency, we test the BD-rate performance of our proposed multi-model CNN filter with ALF on in the AI configuration. Other filters DBF and SAO are set off. Anchor here refers to HPM7.1 with DBF, SAO, and ALF on. The proposal M5129 is used to compare.

3.2. Quality Evaluations

The experimental results of the common test sequence under AI configuration are described in Table 3. The proposed method achieves BD-rate savings of 6.06% on average compared with the anchor. It is worth noting that our proposed scheme achieves high compression efficiency on different resolution videos. BD-rate reductions are 5.67%, 6.58%, 5.93% for UHD4K, 1080p and 720p video sequences. For UHD4K video sequences, BD-RATE saving of multi-model scheme is close to that of the proposal M5129, and for $Campfire$ sequence, our proposed scheme performance exceeds the proposal M5129. This indicates that the distortion is diverse for high-resolution video sequences, and the multi-model scheme is able to process for different distortions effectively.

3.3. Complexity Analysis

Compared with the proposal M5129, which has approximately the same BD-rate performance, our proposed multi-model CNN filtering scheme offers significant time savings in the decoder. Compared with the proposal M5129, the decoding time savings of our proposed scheme are shown in Table 3. It can be seen that our proposed scheme has a time saving of over 20% at all different resolutions. It is worth noting that up to 31.41% time savings can be achieved at 4K resolution. With the improvement of video resolution, the time saving of our proposed multi-model CNN scheme keeps increasing.

4. CONCLUSION

In this paper, we propose a low-complexity multi-model CNN in-loop filter for AVS3. Motivated by ensemble model, we first select stacked residual blocks as the lightweight single model structure after the comparative research, then train multiple single models to replace the complex model through a multi-model iterative training mechanism. Subsequently, we optimize the multi-models number and depth of the network and add the design of frame-level flag to improve the flexibility of our scheme. Extensive experiments show that our proposed multi-model CNN in-loop filtering scheme can achieve great compression performance. Meanwhile, our scheme can significantly reduce the decoding time compared with other schemes with comparable performance.

5. ACKNOWLEDGEMENTS

This work was supported in part the National Key Research and Development Project of China under Grant 2019YFB18 02701, MoE-China Mobile Research Fund Project under Grant MCM20180702, Chinese National Science Funding 62132006, and Shanghai Key Laboratory of Digital Media Processing and Transmissions.

6. REFERENCES

[1] Jiaqi Zhang, Chuanmin Jia, Meng Lei, Shanshe Wang, Siwei Ma, and Wen Gao, "Recent development of avs video coding standard: Avs3," in *2019 Picture Coding Symposium (PCS)*. IEEE, 2019, pp. 1–5.

[2] Jianqiang He and Siwei Ma, "Improvement of deblocking filter in avs2," *AVS-Doc, M3013*, 2012.12.

[3] Jie Chen, Sunil Lee, Elena Alshina, Chanyul Kim, Chih-Ming Fu, Yu-Wen Huang, and Shawmin Lei, "Sample adaptive offset for avs2," *AVS-Doc, M3197*, 2013.09.

[4] Xinfeng Zhang, Si Junjun, Shanshe Wang, Siwei Ma, Jiayang Cai, Qinghua Chen, Yu-Wen Huang, and Shawmin Lei, "Adaptive loop filter for avs2," *AVS-Doc, M3292*, 2014.04.

[5] Siwei Ma, Xinfeng Zhang, Chuanmin Jia, Zhenghui Zhao, Shiqi Wang, and Shanshe Wang, "Image and video compression with neural networks: A review," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 6, pp. 1683–1698, 2019.

[6] Kai Lin, Chuanmin Jia, Zhenghui Zhao, Shanshe Wang, Siwei Ma, Dezhao Wang, Yueyu Hu, Jiaying Liu, Li Wang, and Shiliang Pu, "Neural network filtering based on residual network," *AVS-Doc, M5129*, 2020.01.

[7] Jiangyue Xia and Jiangtao Wen, "Asymmetric convolutional residual network for av1 intra in-loop filtering," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 1291–1295.

[8] Xibin Dong, Zhiwen Yu, Wenming Cao, Yifan Shi, and Qianli Ma, "A survey on ensemble learning," *Frontiers of Computer Science*, vol. 14, no. 2, pp. 241–258, 2020.

[9] Chuanmin Jia, Shiqi Wang, Xinfeng Zhang, Shanshe Wang, Jiaying Liu, Shiliang Pu, and Siwei Ma, "Content-aware convolutional neural network for in-loop filtering in high efficiency video coding," *IEEE Transactions on Image Processing*, vol. 28, no. 7, pp. 3343–3356, 2019.

[10] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[11] Ouyang Tong, Feiyang Liu, Han Zhu, Zhenzhong Chen, Xiaozhong Xu, and Shan Liu, "Ssim based cnn model for in-loop filtering," *Document: JVET-T0069-v2, Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29*, 2020.10.

[12] Yue Li and Li Zhang, "Convolutional neural network-based in-loop filter with adaptive model selection," *Document: JVET-U0068-v2, Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29*, 2021.01.

[13] Eirikur Agustsson and Radu Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 126–135.