

Tencent 腾讯

AHG11: neural network based in-loop filter with adaptive model selection

Liqiang Wang, Xiaozhong Xu, Shan Liu
Tencent



Outline

- Introduction
- Proposed method
- Results
- Conclusions

Introduction

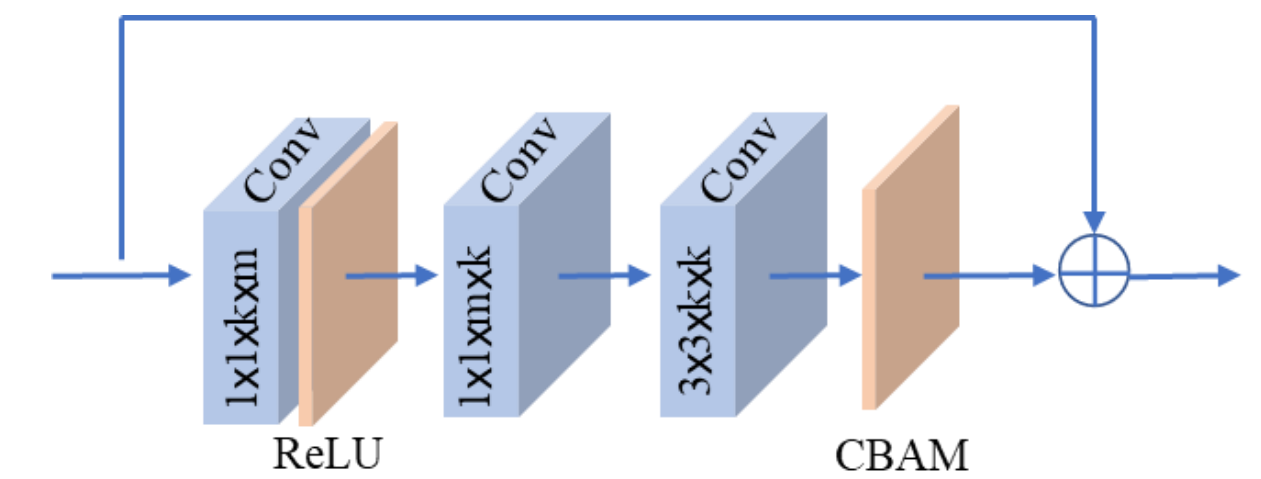
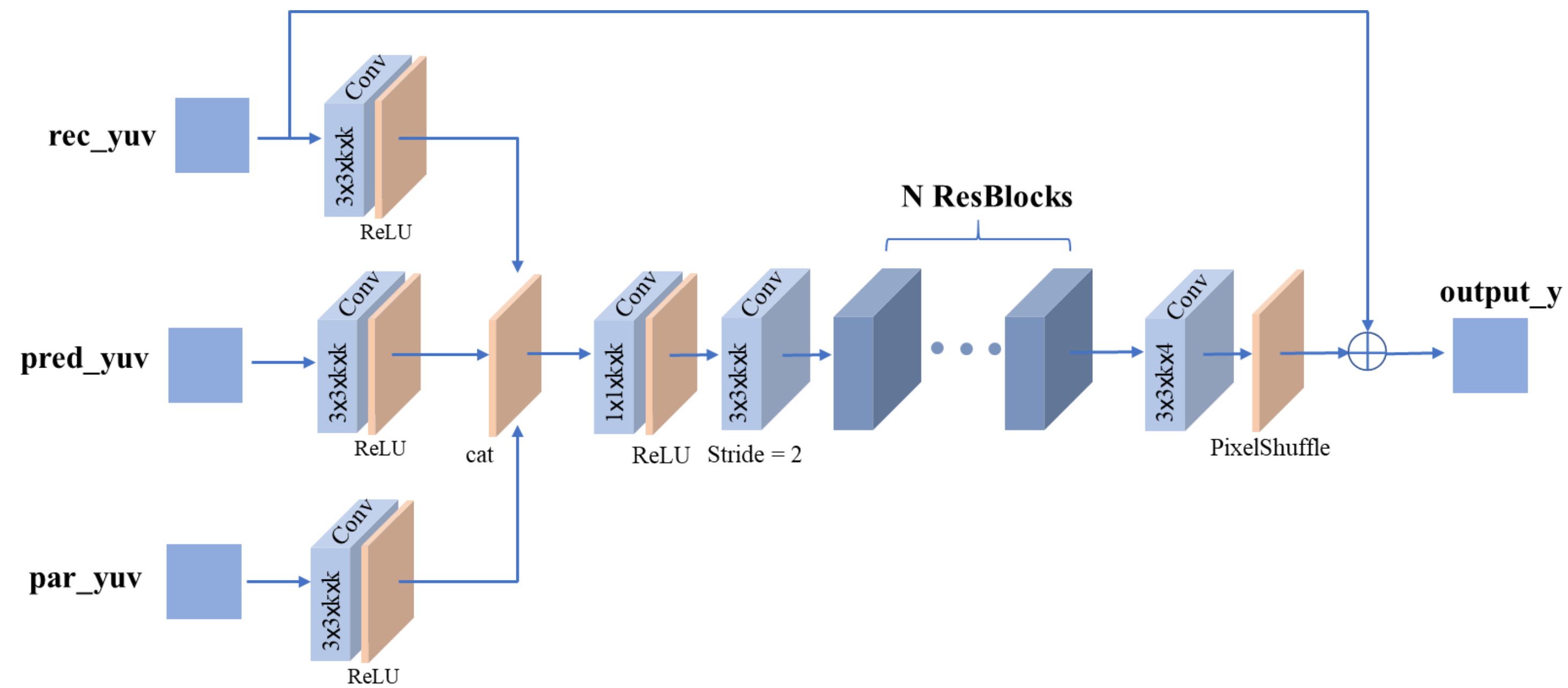
- Based on the network in JVET-W0113.
 - For B slices, the optimal filtered result is decided between the two outputs from the proposed filter and SAO.



- More side information is utilized, such as the prediction image, the partition image.
- For a given base QP, more models are trained to adapt to different coding situations.

Proposed Method

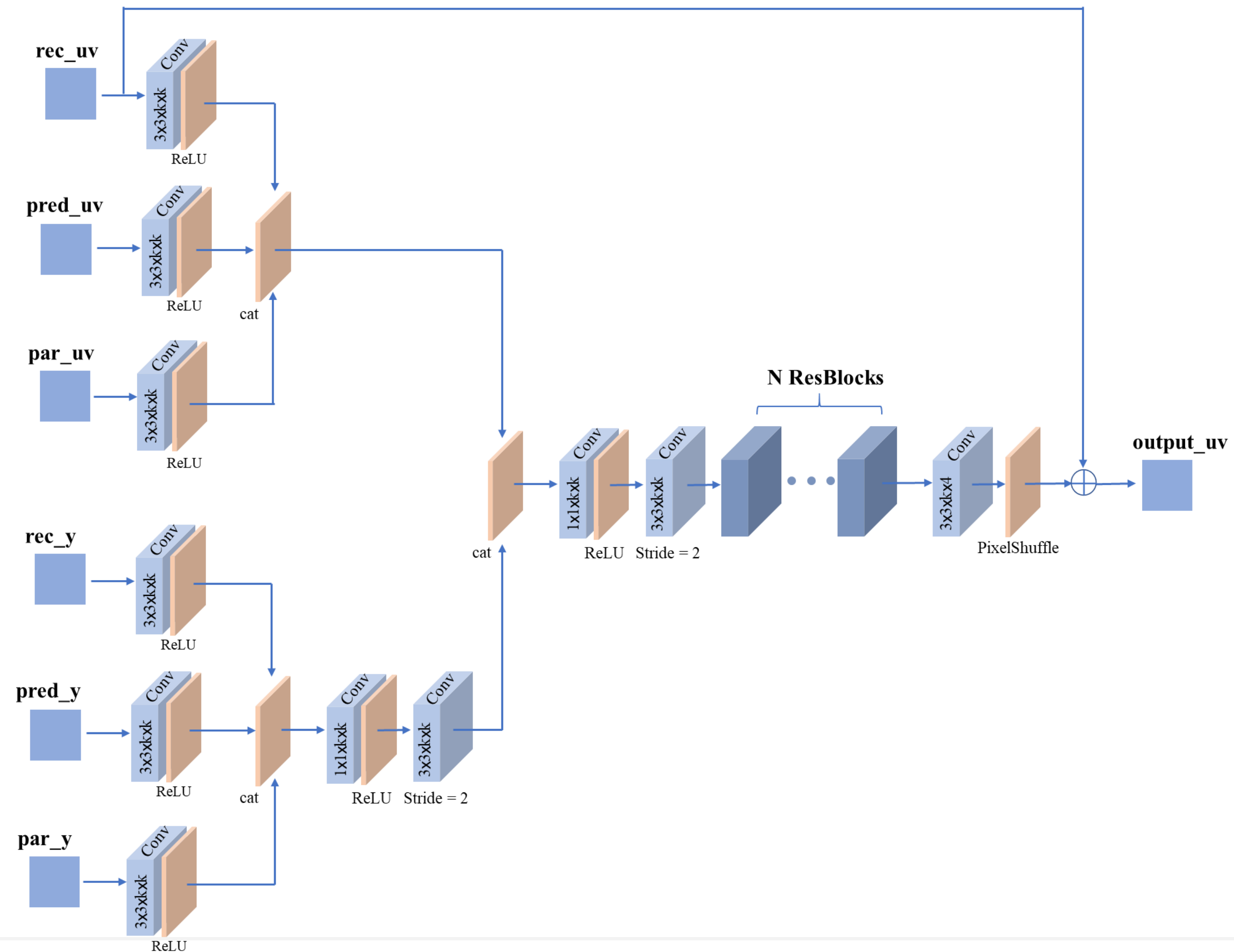
- The network architecture for the luma component:



The Resblock architecture

Proposed Method

- The network architecture for the chroma components:



Proposed Method

- Network information in training and inference stage.

Network Information in Training Stage		
Mandatory	GPU Type	Tesla V100 SXM2 32GB
	Framework:	Torch v1.4.0
	Number of GPUs per Task	1
	Epoch:	~400
	Batch size:	64
	Loss function:	L1
	Training time:	~200h
	Training data information:	DIV2K, TVD, BVI-DVC
	Training configurations for generating compressed training data (if different to VTM CTC):	additional QP17 with RA configuration
Optional		
	Number of iterations	1200
	Patch size	Luma: 144x144, Chroma: 72x72
	Learning rate:	1e-4
	Optimizer:	ADAM
	Preprocessing:	flipped, rotated
	Mini-batch selection process:	random cropped
	Other information:	

Network Information in Inference Stage		
Mandatory	HW environment:	
	GPU Type	CPU only
	Framework:	Torch v1.4.0
	Number of GPUs per Task	0
	Number of Parameters (Each Model)	~2M
	Total Number of Parameters (All Models)	42.78M
	Parameter Precision (Bits)	32
	Memory Parameter (MB)	~8MB / model, 22 models in all
	Multiplay Accumulate (MAC)/pixel	795K
Optional		
	Total Conv. Layers	267
	Total FC Layers	0
	Total Memory (MB)	
	Batch size:	1
	Patch size	Luma: 144x144, Chroma: 72x72
	Changes to network configuration or weights required to generate rate points	
	Peak Memory Usage (Total)	
	Peak Memory Usage (per Model)	
	Border handling	
	Other information:	





Proposed Method

- Implementation
 - For I slices, Deblock and SAO are disabled, and the proposed filter is placed before ALF.
 - For B slices, Deblock and SAO are enabled. The optimal filtered result is decided between the two outputs from the proposed filter and SAO.
 - Multiple results are inferred by three models for B slices, and these models are indexed by $\{qp, qp-5, qp-10\}$, where the qp is the base QP.
 - The proposed filter can be turned on/off at the CTU level and slice level.

Results



	Random access Main10								
	BD-rate Over VTM-11.0_nnv-1.0								
	YUV-PSNR	Y-PSNR	U-PSNR	V-PSNR	Y-MSIM	U-MSIM	V-MSIM	EncT	DecT CPU
Class A1	-10.92%	-8.05%	-18.07%	-21.01%	-8.98%	-20.28%	-22.24%	239%	59604%
Class A2	-13.32%	-8.98%	-25.95%	-26.69%	-8.89%	-25.81%	-24.41%	229%	53356%
Class B	-12.73%	-7.99%	-28.62%	-25.30%	-7.50%	-27.31%	-25.33%	237%	46487%
Class C	-11.86%	-7.96%	-23.35%	-23.79%	-7.19%	-20.46%	-21.58%	191%	45950%
Class E									
Overall	-12.26%	-8.19%	-24.57%	-24.32%	-7.99%	-23.78%	-23.53%	223%	50066%
Class D	-13.67%	-9.73%	-25.11%	-25.88%	-7.09%	-22.51%	-21.52%	187%	44870%
Class F	-5.66%	-3.12%	-13.58%	-12.98%	-3.57%	-14.72%	-14.18%	323%	12310%

	Low delay B Main10								
	BD-rate Over VTM-11.0_nnv-1.0								
	YUV-PSNR	Y-PSNR	U-PSNR	V-PSNR	Y-MSIM	U-MSIM	V-MSIM	EncT	DecT CPU
Class A1	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#NUM!	#NUM!
Class A2	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#NUM!	#NUM!
Class B	-11.80%	-6.58%	-26.89%	-27.97%	-6.37%	-25.12%	-29.13%	225%	41630%
Class C	-11.03%	-6.65%	-23.67%	-24.70%	-5.90%	-20.95%	-21.61%	190%	44449%
Class E	-11.85%	-7.60%	-24.44%	-24.73%	-7.89%	-23.21%	-24.47%	425%	16740%
Overall	-11.64%	-6.86%	-25.46%	-26.51%	-6.59%	-23.27%	-25.78%	250%	33883%
Class D	-12.22%	-7.86%	-24.23%	-26.36%	-5.80%	-22.23%	-21.82%	184%	45061%
Class F	-6.35%	-3.54%	-15.54%	-14.05%	-4.04%	-18.01%	-16.79%	327%	16030%

	All Intra Main10								
	BD-rate Over VTM-11.0_nnv-1.0								
	YUV-PSNR	Y-PSNR	U-PSNR	V-PSNR	Y-MSIM	U-MSIM	V-MSIM	EncT	DecT CPU
Class A1	-9.31%	-6.33%	-17.19%	-19.34%	-7.98%	-19.92%	-21.03%	205%	45534%
Class A2	-10.00%	-6.49%	-20.65%	-20.43%	-6.94%	-22.39%	-18.47%	157%	40414%
Class B	-9.93%	-6.26%	-22.23%	-19.61%	-6.21%	-23.38%	-22.02%	147%	38665%
Class C	-9.78%	-7.31%	-16.35%	-18.05%	-6.81%	-18.58%	-19.91%	125%	28838%
Class E	-12.30%	-9.27%	-21.96%	-20.84%	-9.51%	-22.03%	-22.59%	157%	45090%
Overall	-10.20%	-7.05%	-19.77%	-19.56%	-7.31%	-21.35%	-20.89%	153%	38475%
Class D	-9.79%	-7.31%	-16.51%	-17.95%	-6.38%	-20.26%	-20.05%	121%	26223%
Class F	-5.05%	-3.30%	-10.63%	-9.97%	-3.32%	-13.28%	-13.88%	124%	15460%





Conclusions

- The neural network based in-loop filter is presented in this contribution, and adaptive model selection is used for B slices.
- Good coding performance can be achieved by applying the proposed method.

Thanks