海康威视 HIKVISION

科技呵护未来

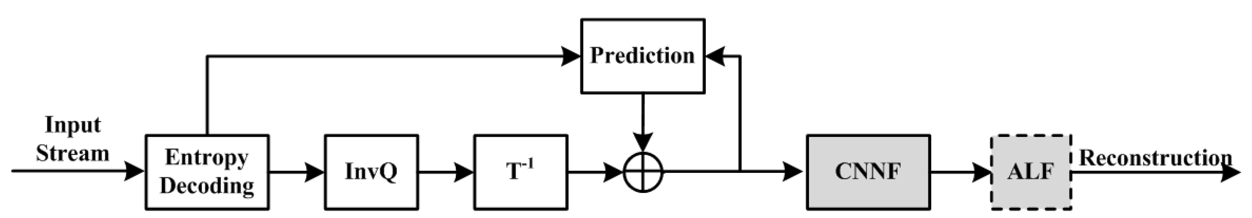
A practical convolutional neural network as loop filter for intra frame

yaojiabao@hikvision.com

Introduction

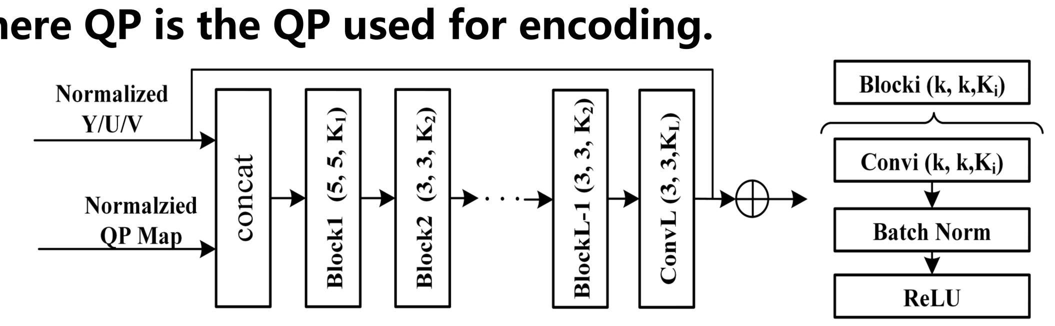
- We propose a practical convolutional neural network filter(CNNF) to replace all traditional filters to improve the subjective and objective quality. Both decoded frame and QP are taken as inputs to CNNF to obtain a QP independent model and adapt to reconstructions with different qualities. After training, the obtained model is compressed for acceleration. Also the compressed model is quantized to ensure the consistency.

CNN Filter in the JEM7.0[1]



Network structure of CNNF

- **CNNF includes two inputs, the reconstruction and QP map. QP map is generated by QPMap(x,y) = QP, where QP is the QP used for encoding.**



- **During training, a simple CNN with 8 layers was taken and all the filter numbers in each layers was set to 64, the kernel size is set to 5*5 in the first layer, others are set to 3*3. By connecting the normalized Y, U or V to summation layer, the network is regularized to learn characteristics of residual between the decoded frame and the original one.**

Model Compression

- **To decrease the calculation of the CNN, two additional regularizers are designed,**

$$\text{Loss} = \underbrace{\frac{1}{2M}\sum_{i=1}^{M}\left\|y^i - f_w(x^i)\right\|^2}_{mean\ square\ error} + \underbrace{\lambda_w \sum_{j=1}^{L}\left\|W_j\right\|_{g1}}_{normal\ regularizer}$$

$$\underbrace{+\ \lambda_S\|S\|_{g2} + \lambda_{lda}\sum_{j=1}^{L-1}\sum_{i=1}^{L-1}\left\|\frac{W_j}{\|W_j\|}-\frac{W_i}{\|W_i\|}\right\|_1}_{additional\ regularizer}$$

$y^i, f_w(x^i)$ are the ground truth and filtered results of x^i

$W_{(\cdot)}$ is the parameters to be learned and S is the scale parameters in BN layer.

- **The first additional regularizer was set to make the learned scale parameters in BN layer tends to be zeros. After training, the corresponding filter will be pruned, hers is the filter numbers after being pruned.**

convL	K_1	K_2	K_3	K_4	K_5	K_6	K_7
# Filter	45	54	58	48	51	40	31

Dynamic Fixed Point Inference

- **To ensure consistency between encoding and decoding across different platforms, DFP[2] are proposed to be used in testing. A value V in dynamic fixed point is described by,**

$$V = (-1)^{s} \cdot 2^{-FL} \sum_{i=0}^{B_f-1} 2^i \cdot x_i$$

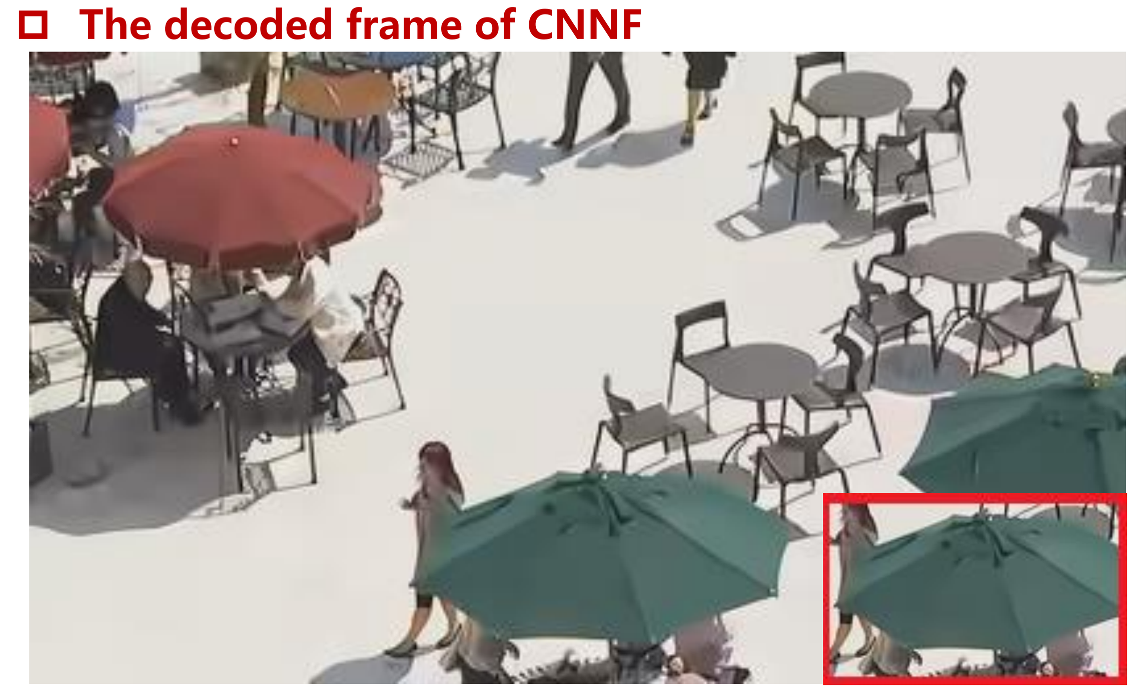
- **Each group in the same layer share one common FL estimated from available training data and layer parameters,**

convL	1	2	3	4	5	6	7	8
FL_w	9	8	8	8	8	8	8	10
FL_b	17	15	14	16	15	13	13	16
FL_o	15	14	14	15	15	15	16	18

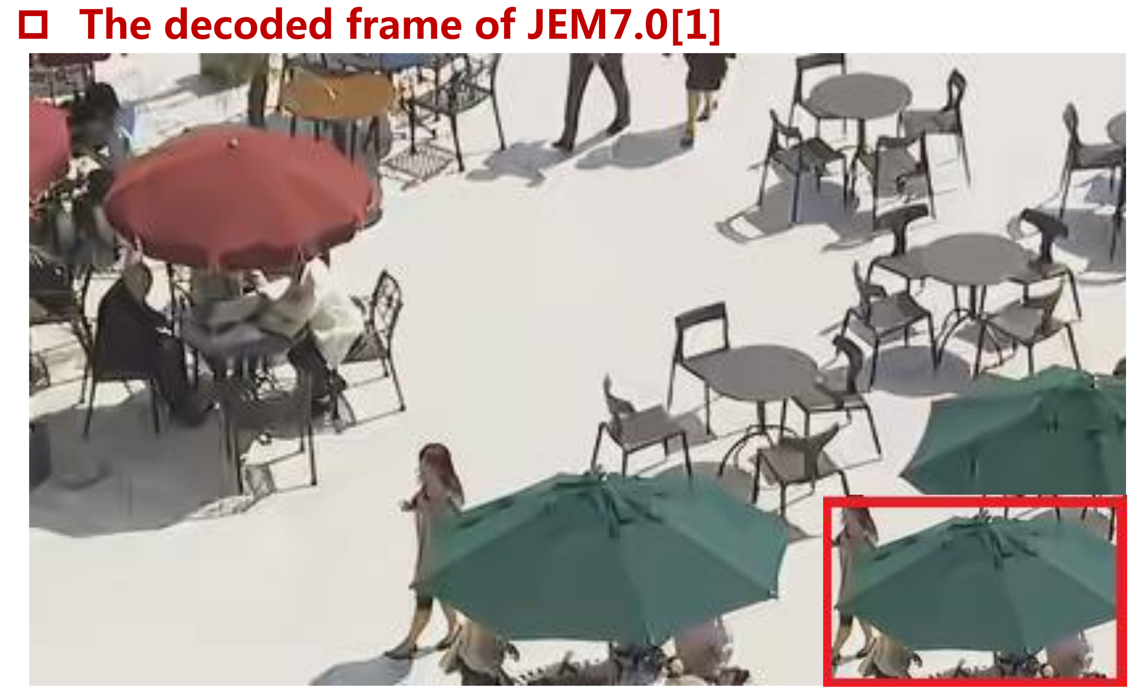
Qualitative Results

The original frame

The decoded frame of CNNF



The decoded frame of JEM7.0[1]



BD RATE on JEM7.0[1]

Test results of AI configuration with ALF off

	Y	U	V
ClassA1	-1.57%	-4.74%	-4.03%
ClassA2	-2.36%	-5.72%	-6.07%
ClassB	-2.71%	-4.58%	-5.99%
ClassC	-3.70%	-6.21%	-8.21%
ClassD	-4.07%	-5.29%	-7.98%
ClassE	-3.97%	-5.64%	-4.81%
Overall	**-3.14%**	**-5.21%**	**-6.28%**

Test results of RA configuration with ALF on

				CPU	
	Y	U	V	EncT	DecT
ClassA1	-0.39%	-1.96%	-1.93%	99%	275%
ClassA2	-1.76%	-3.70%	-4.29%	99%	303%
ClassB	-1.46%	-4.65%	-4.14%	99%	339%
ClassC	-1.28%	-4.40%	-4.75%	99%	289%
ClassD	-1.22%	-3.28%	-4.20%	99%	219%
Overall	**-1.23%**	**-3.65%**	**-3.88%**	**99%**	**284%**

Test results of AI configuration with ALF on

				CPU+GPU		CPU	
	Y	U	V	EncT	DecT	EncT	DecT
ClassA1	-2.26%	-6.21%	-5.05%	93%	157%	109%	15360%
ClassA2	-3.58%	-6.33%	-7.02%	92%	158%	112%	16312%
ClassB	-3.08%	-5.06%	-6.27%	94%	148%	108%	15360%
ClassC	-3.88%	-6.98%	-9.11%	94%	158%	103%	11139%
ClassD	-4.13%	-5.63%	-8.20%	94%	214%	102%	7256%
ClassE	-4.93%	-7.41%	-6.88%	94%	169%	111%	15441%
Overall	**-3.57%**	**-6.17%**	**-7.06%**	**93%**	**157%**	**109%**	**12887%**

Reference

[1] https://jvet.hhi.fraunhofer.de/svn/svn_HMJEMSoftware/branches/HM-16.6-JEM-7.0-dev/, 2018, [Online; accessed5-February-2018].

[2] Philipp Gysel, Mohammad Motamedi, and Soheil Ghiasi, "Hardware-oriented approximation of convolutional neural networks," arXiv preprint arXiv:1604.03168,2016

The open source: https://github.com/Hikvision-Codec/caffe-DFP