Overfitting NN loop-filters in video coding

Ruiying Yang*†, Maria Santamaria*, Francesco Cricri*, Honglei Zhang*,
Jani Lainema*, Ramin G. Youvalari*, Miska M. Hannuksela* and Tapio Elomaa†
*Nokia Technologies, Tampere, Finland
†Tampere University, Tampere, Finland

***Abstract*—Overfitting is usually regarded as a negative condition since it impairs the generalisation power of a model. Nevertheless, overfitting a Neural Network (NN) on test data may be advantageous to improve the compression efficiency of image/video coding tools and systems. Previous research has demonstrated the benefits of NN overfitting for post-processing operations, i.e. post-filters, but not yet for actual decoding tools. Generally, the NN is overfitted on test data at the encoder end, and the weight update is coded and sent to the decoder end along the image/video bitstream. The proposed approach follows this strategy. In particular, the overfitting of the Low Operation Point (LOP) loop-filter in NN-based Video Coding (NNVC) software is studied. The overall approach yields Bjøntegaard Delta rate (BD-rate) of -7.74%, -13.73% and -12.49%, for the Y, U and V components, respectively. Out of these coding gains, 1.21%, 6.43% and 5.52%, for the Y, U and V components, are attributed to the overfitting. The boost in the coding gains comes with only 1.5% more complexity, due to the multiplier parameters introduced during the overfitting.**

***Index Terms*—Loop-filter, neural network, overfitting, video coding.**

I. Introduction

The popularity and effectiveness of Neural Networks (NNs) has spread to a wide range of application fields, including video compression. The Joint Video Experts Team (JVET) is investigating NN technologies to attain video compression efficiency beyond the Versatile Video Coding (VVC) standard [1].

In video compression, as in other domains, a NN is usually trained on a big set of data in order to perform well on average on any new data. That is, the goal is to get a high generalisation power. It is also known that when a NN is overfitted, it learns particularities from the training data and, in general, performs rather well only on such data. This situation is perhaps detrimental for most applications, but in video compression it can be exploited to increase the efficiency. Often at the encoder end, the NN parameters are overfitted on a small set of data. The weight update that results from this process is compressed and delivered to the decoder end, where the overfitted NN is recovered prior to inference [2]–[11]. Other forms of overfitting focus on adapting the latent tensor at test time by typically back-propagating the loss with respect to the latent tensor [12]–[14].

In JVET, NNs are being explored for intra-prediction, inter-prediction, loop-filtering, post-filtering and super resolution [15]. The tools are available in the NN-based Video Coding (NNVC) software, which is built on top of the VVC Test Model (VTM) 11.0. The post-filtering tool uses overfitting as content-adaptive mechanism and achieves higher coding gains compared to its baseline (where no overfitting is done) [16].

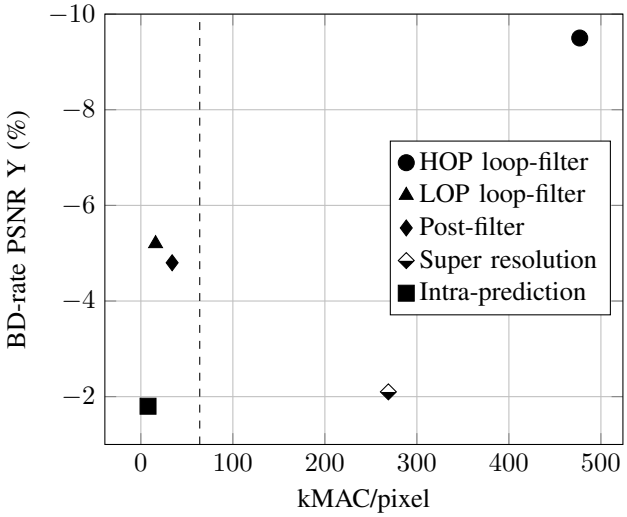

Fig. 1. Complexity performance trade-off of NN tools in NNVC 5.0 (random access configuration). The dashed line represents real-time processing on a GPU (64 kMAC/pixel) [15].

Fig. 1 displays the trade-off between coding efficiency and computational complexity of various NN tools in NNVC 5.0. The former is expressed in terms of the Bjøntegaard Delta rate (BD-rate) derived from the Peak Signal-to-Noise Ratio (PSNR) of the luma (Y) component, and the latter in units of a thousand Multiply-ACcumulate (kMAC) operations per pixel. While the High Operation Point (HOP) loop-filter provides around -9.5% BD-rate, it is significantly more complex (477 kMAC/pixel) than other NN filters. On the other hand, the Low Operation Point (LOP) loop-filter [17] strikes a reasonable balance between coding efficiency (-5.2% BD-rate) and computational complexity (16 kMAC/pixel). Therefore, by overfitting the LOP loop-filter on test data, here the aim is to increase its coding efficiency along with keeping the complexity low. While previous works have concentrated on overfitting of NN post-filters, or overfitting non-NN loop-filters [18], this paper is arguably the first to apply overfitting to an NN loop-filter.

The proposed approach is evaluated on NNVC 5.0 under the JVET Random Access (RA) Common Test Conditions (CTC) for NNVC [19]. On top of NNVC 5.0 without any NN tools enabled, when the NN intra tool and the overfitted LOP loop-

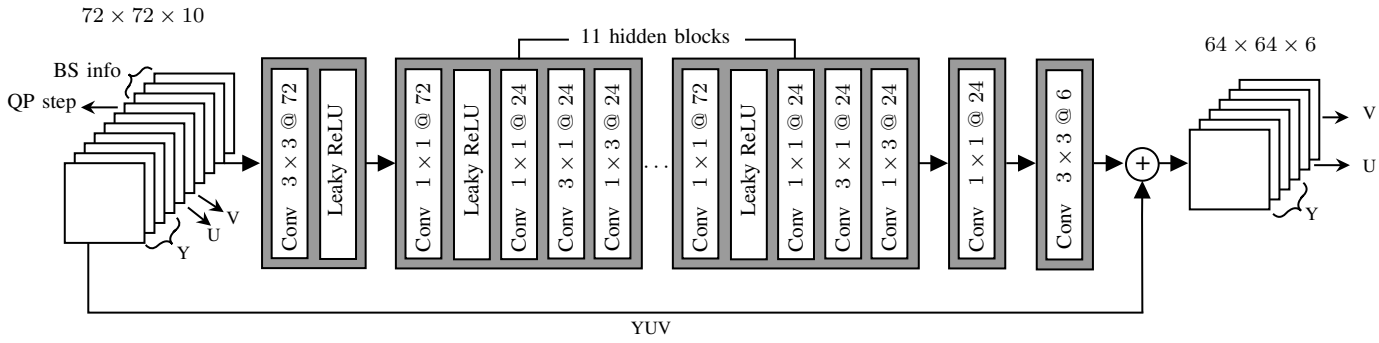

Fig. 2. LOP loop-filter architecture [17]. BS stands for boundary strength.

filter are used, the average coding gains are 7.74%, 13.73% and 12.49% for the Y, U and V components, respectively. When the anchor has enabled the NN intra tool and the base LOP loop-filter, the coding gain improves by 1.21%, 6.43% and 5.52% for the Y, U and V components, and the computational complexity goes from 24 kMAC/pixel to 24.3 kMAC/pixel (1.5% increase) due to introduction of multiplier parameters during the overfitting.

The rest of paper is organised as follows. Section II describes the proposed approach. Section III presents the experimental results. Section IV summarises final remarks.

II. Overfitting NNVC LOP loop-filter

This work aims at increasing the coding efficiency of NNVC 5.0 LOP loop-filter by overfitting it on each test video. Since the filter is known at both the encoder and decoder ends, the weight update is compressed and sent along the video bitstream in order to recover the overfitted filter, before it is deployed.

The LOP filter architecture is illustrated in Fig. 2. The network processes 10-bit YUV 4:2:0 blocks of size 144×144. The input are reconstructed blocks (before any loop-filter is applied), the Quantisation Parameter (QP) step $q = 2^{(\mathrm{QP}-42)/6}$, and the boundary strength information, i.e. a map that contains the boundaries and edges relative to the block partitioning. The output are the filtered blocks. For both input and output, the luma samples are interleaved to create four channels that match the size of the chroma samples.

A. Overview

Most of the complexity of the proposed approach is at the encoder end given that two-pass encoding is required.

In the first-pass the test video is encoded, the training data is extracted and the overfitting takes place. The resulting weight update (parameter difference between base NN and overfitted NN) is then lossy coded with the Moving Picture Experts Group (MPEG) Neural Network Compression and representation (NNC) standard [20]. Finally, the decoded overfitted NN is generated and used in the second-pass encoding, replacing the corresponding base LOP loop-filter model.

Once the second-pass encoding is completed, the video bitstream and the NNC bitstream are transmitted to the decoder end.

At the decoder, first the NNC bitstream is decoded and then the overfitted NN is recovered. In this way, the overfitted NN can be used when needed.

B. Overfitting

The LOP loop-filter consists of four different models. However, for each test sequence and coding configuration only one of these is overfitted. The selected model is the one that at inference time provides the higher gain in terms of PSNR. In order to restrict the overhead from the weight update, multiplier parameters are introduced, initialised in 1 and overfitted for each non-separable convolutional block:

$$\mathbf{y} = (\mathbf{K} * \mathbf{x} + \mathbf{b}) \cdot \mathbf{m}, \tag{1}$$

where $\mathbf{x}$ is the input, $\mathbf{K}$ is the kernel, $*$ is the convolution operator, $\mathbf{b}$ the bias and $\mathbf{m}$ the multiplier.

The overfitting involves several training iterations where the training data and the test data are the same. The loss function is the weighted Mean Squared Error (MSE), with weights 4, 1 and 1 for the Y (de-interleaved), U and V components, respectively. The optimiser is Adam [21], the learning rate is 0.001 and the batch size is 64.

Given this process occurs at the encoder side, and that only the base LOP models are part of the video encoder and decoder, the weight update that results from the overfitting must be delivered to the decoder somehow.

C. Weight update compression

For transmission, the weight update is coded with the NNC standard. The encoding makes use of Inference-Optimised Quantisation (IOQ) [22] in order to find a trade-off between ΔPSNR (difference between the PSNR of the filtered video and the PSNR of the unfiltered video) and the NNC bitstream size.

IOQ starts from a low QP, which is increased at each iteration. Each iteration the NN weights are quantised with the current QP and encoded with Deep Context-Adaptive Binary Arithmetic Coding (DeepCABAC) [23]. The ΔPSNR of the

TABLE I
CODING PERFORMANCE. ANCHOR: NN TOOLS DISABLED. TEST: NN INTRA TOOL AND LOP LOOP-FILTER ENABLED (3 BASE AND 1 OVERFITTED MODELS).

Class	BD-rate Y	BD-rate U	BD-rate V
A1	−8.69%	−8.66%	−13.54%
A2	−6.75%	−13.29%	−11.76%
B	−7.88%	−16.07%	−12.30%
C	−7.61%	−14.93%	−12.50%
Overall	−7.74%	−13.73%	−12.49%
D	−8.29%	−16.82%	−16.31%
F	−4.09%	−13.17%	−9.94%

resulting NN is evaluated. When the performance falls below a certain threshold, the QP of the previous iteration is chosen as the best and the final NNC bitstream is generated.

D. Decoded overfitted model

The overfitting is done in TensorFlow 2.8.0 using 32-bit floating-point precision, although, the inference within the video codec uses 16-bit fixed-point precision. Hence, before the overfitted model is ready to be used, it is quantised. The NN parameters are normalised to reduce large values and avoid overflows [17].

III. EXPERIMENTAL RESULTS

The video codec utilised is NNVC 5.0, where for NN inference uses the Small Adhoc Deep-Learning (SADL) library [24]. The video codec is ran on Intel Xeon Gold 6240 CPUs only, whereas the overfitting ran on Nvidia Tesla A100 GPUs.

The LOP loop-filter signalling mechanism remains the same: index of the filter to be used, enable/disabled filter usage at frame or block level, and residual scaling factors.

The evaluation was done according to the JVET RA CTC for NNVC [19], using as test content the mandatory video sequences, which have spatial resolutions ranging from 240p to 2160p. The PSNR measures the quality of the reconstructed videos, and the BD-rate [25] indicates the coding performance.

In the first-pass encoding, used to generate the overfitting data, the NN intra tool and the base LOP loop-filter are enabled. During the second-pass encoding, the overfitted filter replaces the corresponding base filter.

Table I shows the BD-rate when the NN intra tool and the overfitted LOP loop-filter are enabled, and the anchor has all the NN tools disabled. Table II shows the performance of the same test, but in this case the anchor has the NN intra tool and the base LOP loop-filter enabled. These results demonstrate the benefits of the overfitting strategy in conjunction with the intra tool: reference (and output) frames are of higher quality, allowing for more temporal and spatial compression.

Fig. 3 depicts a comparison of the trade-off (coding efficiency and complexity) between the proposed approach and various NN tools in NNVC 5.0. It can be observed that overfitting the LOP loop-filter increases the coding gains, reducing the gap between LOP and HOP filters and maintaining a low complexity below to 64 kMAC/pixel.

In a different test, during the second-pass encoding the overfitted filter is considered an additional LOP model candidate, meaning five filters are evaluated. The results are in

TABLE II
CODING PERFORMANCE. ANCHOR: NN INTRA TOOL AND LOP LOOP-FILTER ENABLED (BASE MODELS). TEST: NN INTRA TOOL AND LOP LOOP-FILTER ENABLED (3 BASE AND 1 OVERFITTED MODELS).

Class	BD-rate Y	BD-rate U	BD-rate V
A1	−1.59%	−3.96%	−7.97%
A2	−0.22%	−7.77%	−6.84%
B	−1.70%	−8.42%	−5.13%
C	−1.06%	−4.80%	−3.17%
Overall	−1.21%	−6.43%	−5.52%
D	−0.80%	−7.86%	−7.23%
F	−1.07%	−7.51%	−5.16%

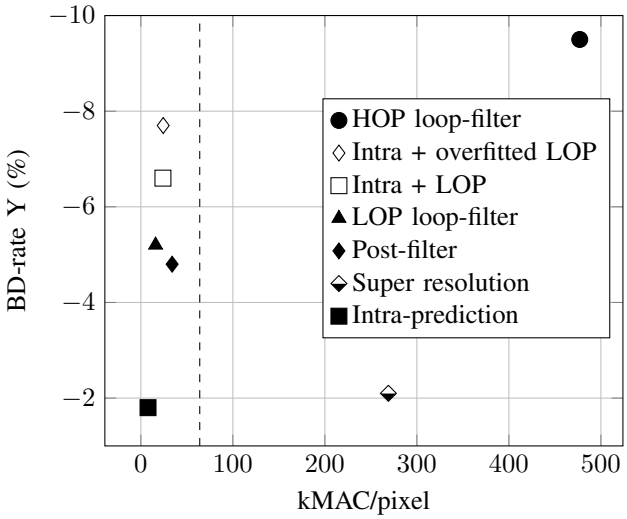

Fig. 3. Complexity performance trade-off of the proposed approach and NN tools in NNVC 5.0 in RA configuration. The dashed line marks 64 kMAC/pixel.

TABLE III
CODING PERFORMANCE. ANCHOR: NN INTRA TOOL AND LOP LOOP-FILTER ENABLED (BASE MODELS). TEST: NN INTRA TOOL AND LOP LOOP-FILTER ENABLED (4 BASE AND 1 OVERFITTED MODELS).

Class	BD-rate Y	BD-rate U	BD-rate V
C	−1.12%	−5.14%	−3.30%
D	−0.84%	−8.08%	−7.35%

TABLE IV
CODING PERFORMANCE. ANCHOR: NN TOOLS DISABLED. TEST: LOP LOOP-FILTER ENABLED (3 BASE AND 1 OVERFITTED MODELS).

Class	BD-rate Y	BD-rate U	BD-rate V
C	−6.17%	−14.12%	−11.49%
D	−7.14%	−15.87%	−15.65%

TABLE V
CODING PERFORMANCE. ANCHOR: LOP LOOP-FILTER ENABLED (BASE MODELS). TEST: LOP LOOP-FILTER ENABLED (3 BASE AND 1 OVERFITTED MODELS).

Class	BD-rate Y	BD-rate U	BD-rate V
C	-1.07%	-4.80%	-3.27%
D	-0.83%	-7.57%	-7.42%

Table III, which shows a slight boosts in the BD-rate compared to the scenario where four LOP models (three base and one overfitted) are evaluated (Table II).

In order to remove the influence of the NN intra tool on the overfitting strategy, the NN intra tool is disabled during the first- and second-pass encodings, meaning the overfitting data is different too. Tables IV and V show the coding gains on top of NNVC 5.0 with all NN tools disabled and when only the base LOP loop-filter is enabled.

A. Encoding and decoding times

When using Intel Xeon Gold 6240 CPUs, a single-pass video encoding of the proposed approach takes nearly the same elapsed time than the anchor in Table I (101%). A similar trend was observed for the video decoding, where the elapsed time of the proposed approach is on average 105% of the elapsed time for the anchor. That is owning to the fact that the same amount of models are deployed, and that only one out of four models requires additional kMAC/pixel associated with the multiplier parameters. Since the proposed approach uses multi-pass encoding, the actual video encoding takes roughly double the elapsed time of the anchor.

Furthermore, overfitting in Nvidia Tesla A100 GPU, takes from 1 hour for a 420p video sequence, up to 11 hours for a 4K video sequence. In addition, the weight update compression ranges from 2 minutes for a 420p video sequence to 1 hour for a 4K video sequence.

Finally, decoding the weight update and reconstructing the overfitted model have a negligible time impact, as they take 5 seconds on average.

IV. CONCLUSION

This paper studied the overfitting LOP loop-filter in NNVC 5.0 on test data. The strategy makes use of two-pass encoding. After the first-pass, the overfitting data is extracted and the training is done. The weight update is NNC-coded, and the decoded overfitted NN is generated. In the second-pass encoding, this NN is deployed as loop-filter. Finally, both the video and the NNC bitstream are transmitted. For decoding, first the NNC bitstream is decoded and then the overfitted NN is obtained. Next, the video decoding takes place, with the overfitted NN ready to be used as loop-filter. The results showed that the overfitting a low complexity NN tool on test data boost the coding gains, while introducing a minimal increase in computing complexity.

REFERENCES

[1] Recommendation ITU-T H.266 | ISO/IEC 23090-3, "Versatile video coding," 2020.

[2] G. He, C. Wu, L. Li, J. Zhou, X. Wang, Y. Zheng, B. Yu, and W. Xie, "A video compression framework using an overfitted restoration neural network," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 593–597.

[3] N. Zou, H. Zhang, F. Cricri, H. R. Tavakoli, J. Lainema, M. Hannuksela, E. Aksu, and E. Rahtu, "L^2C – learning to learn to compress," in *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, 2020.

[4] T. van Rozendaal, I. A. Huijben, and T. Cohen, "Overfitting for fun and profit: Instance-adaptive data compression," in *International Conference on Learning Representations*, 2021.

[5] M. Santamaria, Y.-H. Lam, F. Cricri, J. Lainema, R. G. Youvalari, H. Zhang, M. M. Hannuksela, E. Rahtu, and M. Gaubbuj, "Content-adaptive convolutional neural network post-processing filter," in *2021 IEEE International Symposium on Multimedia (ISM)*, 2021, pp. 99–106.

[6] H. Zhang, F. Cricri, H. R. Tavakoli, M. Santamaria, Y.-H. Lam, and M. M. Hannuksela, "Learn to overfit better: finding the important parameters for learned image compression," in *2021 International Conference on Visual Communications and Image Processing (VCIP)*, 2021.

[7] W. Jiang, W. Wang, S. Li, and S. Liu, "Online meta adaptation for variable-rate learned image compression," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2022, pp. 497–505.

[8] M. Santamaria, R. Yang, F. Cricri, H. Zhang, J. Lainema, R. G. Youvalari, H. R. Tavakoli, and M. M. Hannuksela, "Overfitting multiplier parameters for content-adaptive post-filtering in video coding," in *2022 10th European Workshop on Visual Information Processing (EUVIP)*, 2022.

[9] R. Yang, M. Santamaria, F. Cricri, H. Zhang, J. Lainema, R. G. Youvalari, and M. M. Hannuksela, "Low-precision post-filtering in video coding," in *2022 IEEE International Symposium on Multimedia (ISM)*, 2022, pp. 137–140.

[10] N. Zou, F. Cricri, H. Zhang, H. R. Tavakoli, M. M. Hannuksela, and E. Rahtu, "The lottery ticket adaptation for neural video coding," in *2022 IEEE International Symposium on Multimedia (ISM)*, 2022, pp. 141–145.

[11] K. Tsubota, H. Akutsu, and K. Aizawa, "Universal deep image compression via content-adaptive optimization with adapters," in *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023, pp. 2528–2537.

[12] J. Campos, S. Meierhans, A. Djelouah, and C. Schroers, "Content adaptive optimization for neural image compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2019.

[13] N. Le, H. Zhang, F. Cricri, R. Ghaznavi-Youvalari, H. R. Tavakoli, and E. Rahtu, "Learned image coding for machines: A content-adaptive approach," in *2021 IEEE International Conference on Multimedia and Expo (ICME)*, 2021.

[14] M. Abdoli, G. Clare, and F. Henry, "Gop-based latent refinement for learned video coding," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[15] E. Alshina, F. Galpin, Y. Li, D. Rusanovskyy, M. Santamaria, J. Ström, L. Wang, and Z. Xie, "EE1: Summary report of exploration experiment on neural network-based video coding," *JVET 31st Meeting, document JVET-AE0023*, Jul. 2023.

[16] M. Santamaria, R. Yang, F. Cricri, J. Lainema, H. Zhang, R. G. Youvalari, and M. M. Hannuksela, "EE1-1.11: Content-adaptive post-filter," *JVET 29th Meeting, document JVET-AD0055*, Jan. 2023.

[17] J. N. Shingala, A. Shyam, A. Suneja, S. P. Badya, T. Shao, A. Arora, P. Ying, P. Pu, T. Lu, and S. McCarthy, "EE1-1.1: Complexity reduction on neural-network loop filter," *JVET 30th Meeting, document JVET-AD0156*, Apr. 2023.

[18] X. Meng, C. Jia, X. Zhang, M. Lei, S. Wang, L. Li, and S. Ma, "Parametric non-local in-loop filter for future video coding," in *2022 Data Compression Conference (DCC)*, 2022, pp. 474–474.

[19] E. Alshina, R.-L. Liao, S. Liu, and A. Segall, "Common test conditions and evaluation procedures for neural network-based video coding technology," *JVET 30th Meeting, document JVET-AD2016*, Apr. 2023.

[20] ISO/IEC FDIS 15938-17, "Compression of neural networks for multimedia content description and analysis," 2020.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, Y. Bengio and Y. LeCun, Eds., 2015.

[22] P. Haase, D. Becking, H. Kirchhoffer, K. Müller, H. Schwarz, H. Samek, D. Marpe, and T. Wiegand, "Encoder optimizations for the NNR standard on neural network compression," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 3522–3526.

[23] S. Wiedemann, H. Kirchhoffer, S. Matlage, P. Haase, A. Marban, T. Marinč, D. Neumann, T. Nguyen, H. Schwarz, T. Wiegand, D. Marpe, and W. Samek, "DeepCABAC: A universal compression algorithm for deep neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 4, pp. 700–714, 2020.

[24] F. Galpin, P. Nikitin, T. Dumas, and P. Bordes, "SADL small adhoc deep-learning library," *JVET 23rd Meeting, document JVET-W0181*, Apr. 2021.

[25] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," *VCEG 13th Meeting, document VCEG-M33*, Apr. 2001.