Enhanced Machine Learning-based Inter Coding for VVC

Martin Benjak, Holger Meuel, Thorsten Laude and Jörn Ostermann
Institut für Informationsverarbeitung
Leibniz Universität Hannover
Hannover, Germany
{benjak, meuel, laude, office}@tnt.uni-hannover.de

***Abstract*—In this paper, we propose an enhanced machine learning-based inter coding algorithm for VVC. Conceptually, the reference pictures from the decoded picture buffer are processed using a recurrent neural network to generate an artificial reference picture at the time instance of the currently coded picture. The network is trained using a SATD cost function to minimize the bit rate cost for the prediction error rather than the pixel-wise difference. By this we achieved average weighted BD-rate gains of 0.94%. The coding time increased about 5% for the encoder and 300% for the decoder due to the use of a neural network.**

***Index Terms*—VVC, inter coding, video coding, machine learning, recurrent neural networks**

I. Introduction

During the past decades, a tremendous improvement of video coding algorithms was observed. In January 2013, the Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T VCEG and ISO/IEC MPEG finished the technical work for the latest video coding standard, High Efficiency Video Coding (HEVC), which is also referred to as H.265 by ITU-T and as MPEG-H Part 2 by ISO/IEC. After the finalization of HEVC, ISO/IEC and ITU-T established the Joint Video Experts Team (JVET) to develop the HEVC successor. The new standard is referred to as Versatile Video Coding (VVC) and was finalized in 2020. Depending on the application and encoder configurations, VVC achieves a bit rate reduction of around 30% at similar quality compared to HEVC [1].

All modern video codecs share the same fundamental working principle: block-based hybrid video coding. It is the combination of motion-compensated prediction with transform coding for the prediction error. The prediction methods can be distinguished into intra and inter coding. Intra coding relies on previously coded parts of the current picture to predict a new block within this picture. Inter coding additionally utilizes temporal redundancy between consecutive pictures to improve the prediction. Conceptually, previously reconstructed pictures are stored in a reference picture buffer and are used to make a prediction for the currently coded block via motion-compensated prediction. The quality of motion compensated prediction highly depends on the available reference pictures. Furthermore, the better motion-compensation performs, the lower the bit rate for the prediction error becomes.

It is worth noting that due to the motion compensation, the quality of the reference pictures does not necessarily correlate with the pixel-wise fidelity between the current picture and the reference pictures. For example, a reference picture which is a translationally shifted version of the current picture would be a good prediction reference despite the low pixel-wise fidelity between these pictures. More problematic are complex motions or occlusions which cannot be handled by the motion model of the video codec.

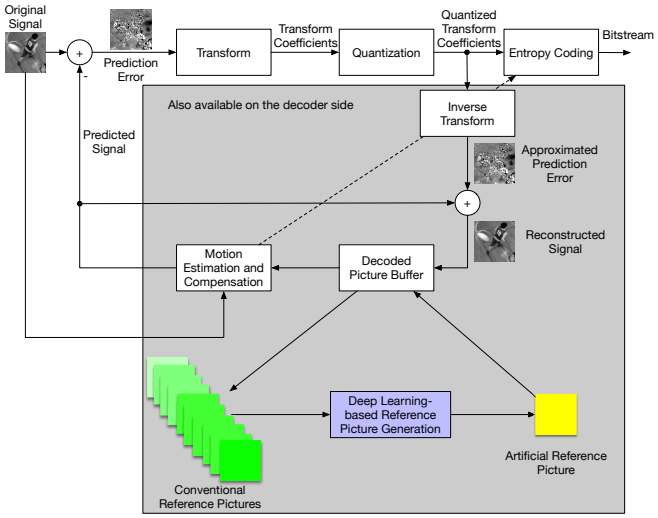

Fig. 1: Codec integration of the proposed method. Conventional reference pictures are stored in a ring buffer.

In [2], Laude *et al.* approached the problem in the context of HEVC by generating *artificial reference pictures* using a recurrent neural network (RNN). They processed the reference pictures in the decoded picture buffer, which they refer to as *conventional reference pictures*, to generate a new reference picture at the time instance of the currently coded picture. This way, they coped with the motion which the translational motion model of this video codec was not able to compensate.

In contrast to HEVC, VVC includes a more advanced affine motion model. Hence, it is of interest, whether Laude's approach is still beneficial or whether the VVC motion model can compensate enough motion to deprecate their method. In this work, we demonstrate the usefulness of the method of Laude *et al.* in the context of VVC. Furthermore, we propose enhancements to their method. In their work, the training of

their network was focused on a high pixel-wise fidelity of the predicted signal. However, considering that the predicted signal is never displayed to the viewer, this is not of uttermost importance. Instead, the reconstructed signal as sum of the predicted signal and the transmitted prediction error is displayed. The fidelity of the reconstructed signal is not available during training due to the lack of the transform coding scheme. Therefore, we use the sum of absolute transformed differences (SATD) as an approximation for the bit rate of the prediction error as optimization criterion for the training. The SATD is also used as an optimization criterion in the VVC reference software VTM.

Our main contributions in this paper are: 1.) Generation of an extrapolated reference picture using a recurrent neural network with SATD cost function. 2.) Complete coding pipeline with the neural network integrated in the video codec VVC.

The remainder of the paper is organized as follows: In Section II, we discuss related works from the literature. The proposed method is presented in Section III and experimental results are discussed in Section IV. In Section V, we conclude the paper.

II. Related Work

During recent years, machine learning-based approaches were introduced to video codecs, e.g. for faster rate-distortion optimization [3] or for higher coding efficiency [4], [5].

Similar to our approach, Lee *et al.* [4], Liu *et al.* [5] and Laude *et al.* [2] use neural networks for improving motion-compensated prediction. While Lee *et al.* and Liu *et al.* interpolate between one forward and one backward reference picture, we solely extrapolate from backward reference pictures into the future without knowing how the future reference pictures look like. Our extrapolation approach is more challenging, but enables our method to be used in low-delay settings. Like our approach, Liu *et al.* [5] is using a SATD based cost function. While they measure a massive complexity increase with an average of more than doubled complexity for the encoder, our complexity increase is much more moderate with 5% for the encoder. Laude *et al.* extrapolate from backward reference pictures like we do with our method. Unlike our approach, they trained their network using a pixel-wise cost function, which we found to be less suited for video coding.

III. Neural Network-based Inter Coding

As in [2], we use the recurrent neural network architecture from the PredNet model proposed by Lotter *et al.* [6] to predict the picture to be coded from the previously coded pictures. This network consists of four stacked modules, each trying to make predictions for its input. The prediction is generated by a convolutional layer from a recurrent representation. The difference between prediction and actual input (prediction error) is then passed through a convolutional layer and given as input to the next layer. In opposite order, the recurrent representation of each module is generated using a Long Short-term Memory (LSTM) layer with the prediction error of the last time step, the recurrent representation of the last layer and the recurrent representation of the last time step as input. This process is repeated with the next picture of a sequence in each time step. After the last time step, the unknown next picture in the sequence is obtained by inputting an arbitrary input (e.g. a blank picture) into the first module.

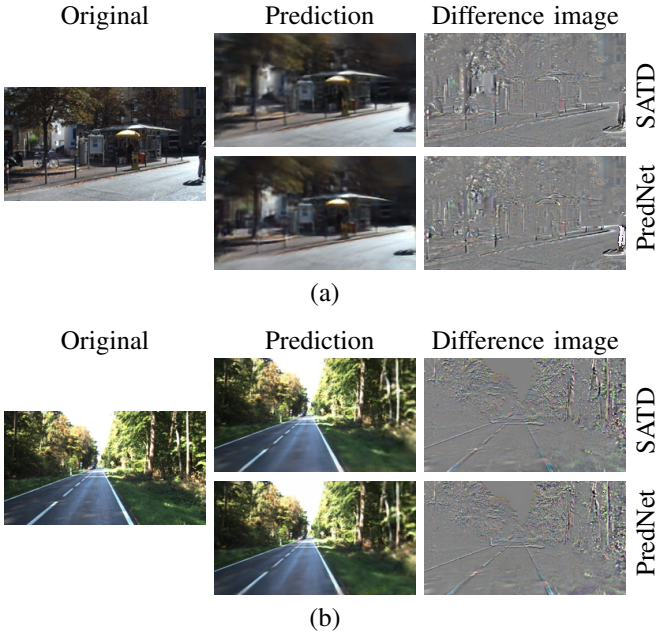

Fig. 2: Exemplary pictures predicted by the neural networks trained optimizing the SATD and PredNet cost function for a curvy sequence (a) and a straight sequence (b). The difference images between the predicted and original pictures show the corresponding prediction error. The predictions for both neural networks are generally less sharp than the original picture, but look similar to each other. It can be observed that the position of the bike driver on the right side of the picture in (a) was predicted wrongly by the neural network trained to optimize the PredNet cost function.

The selection of the cost function determines for which criterion the network configuration is optimized during the training. Only if the cost function is suitable for the problem to be solved, the training can lead to meaningful results. In [6] and [2], PredNet is trained with the pixel-wise sum of the prediction errors as cost function. From a coding point of view, it is desirable to take into account that the prediction leads to a prediction error that can be transmitted at a low bit rate. The error is coded in the frequency domain. The probability density function $p_{\text{coeff}}(\mathcal{C})$ of the coefficients $\mathcal{C}$ calculated by the transformation are approximately mean-free and Laplace-distributed [7]:

$$p_{\text{coeff}}(\mathcal{C}) = \Phi e^{\Psi|\mathcal{C}|} \tag{1}$$

with constants Φ and Ψ. The data rate r required for coding a coefficient can be approximated by the information content

I and is proportional to the absolute value of the coefficient:

$$r \sim I\left(p_{\text{coeff}}\left(\mathcal{C}\right)\right) = -\log_2\left(\Phi e^{\Psi|\mathcal{C}|}\right) \sim \Xi|\mathcal{C}| \tag{2}$$

with a constant Ξ. The entropy of the prediction error, i.e. the average information content and an approximation of the data rate required for the transmission of the same under the condition of a sufficiently good entropy coding, is thus proportional to the SATD. Therefore, the SATD is chosen as the cost function over the pixel-wise sum of the prediction errors.

We calculate the SATD block-wise by dividing the prediction error $\mathbf{P} = \mathbf{I_t} - \mathbf{I_p}$ between target picture $\mathbf{I_t}$ and predicted picture $\mathbf{I_p}$ into N non-overlapping 8x8 blocks $\mathbf{P_i}$. According to Liu *et al.* [5], the SATD $\ell_{\mathrm{S}}(\mathbf{P})$ over all N blocks is obtained by

$$\ell_{\mathrm{S}}(\mathbf{P}) = \sum_{i=1}^{N}\sum_{x=1}^{8}\sum_{y=1}^{8}\left|\mathrm{HT}(\mathbf{P_i})(x,y)\right| \tag{3}$$

using the Hadamard transform

$$\mathrm{HT}(\mathbf{P_i}) = \mathbf{H}\cdot\mathbf{P_i}\cdot\mathbf{H^T} \tag{4}$$

with the 8x8 Hadamard matrix

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}. \tag{5}$$

Since the model is trained with a sequence of T pictures in the YCbCr color space, the final cost is calculated by

$$\ell_{\text{train}} = \sum_{t=1}^{T}\lambda_t\frac{6\ell_{\mathrm{S}}(\mathbf{P_Y}) + \ell_{\mathrm{S}}(\mathbf{P_{Cb}}) + \ell_{\mathrm{S}}(\mathbf{P_{Cr}})}{8}, \tag{6}$$

where the prediction errors $\mathbf{P_Y}$, $\mathbf{P_{Cb}}$ and $\mathbf{P_{Cr}}$ are weighted in the ratio 6/1/1 following [8]. Analog to [6], the time step weight λ_t is set to zero for the first time step, since it is used to initialize the recurrent representation. For all following time steps, λ_t is set to one. Compared to setting only λ_T to one, the influence of preceding time steps is also considered which shortens the training time.

To train our neural network, we used 13 sequences of train and subway drives in HD resolution filmed with a front-viewing camera downloaded from Youtube. The sequences were down-sampled to a frame rate of 10 frames per second and a resolution of 256x144. After down-sampling, the data set consist of 487020 pictures. Each sequence was split into 1000 frames long snippets.

As model parameters, we used a layer channel size of (3, 48, 96, 192) following [6]. We trained two models over 4 epochs with the full training data set. The first model was trained minimizing the pixel-wise prediction error as in [6], [2]. The second model was training using the SATD cost function as formulated in Equation (6). Both models were trained with Adam as optimizer, $\beta_1 = 0.1$, $\beta_2 = 0.999$ and an initial learning rate of 0.001. After 2 epochs, we started to linearly decrease the learning rate down to 0.0008 at the end of the training.

The trained models are used to generate artificial reference pictures for the motion-compensated prediction within the VVC encoding and decoding process. We modified the VVC reference software VTM 7.0 such that after the encoding of a picture, the last coded picture is added to a ring buffer with ten entries. The size of this ring buffer is independent of the size of the reference picture list. The reason for not increasing the size of the reference picture list is the increased signaling cost it would cause. Considering that the reference picture selection for the network is fixed, i.e. it does not imply any signalling overhead, it is reasonable to use ten reference pictures as input. The content of this ring buffer is then fed into the trained neural network. As output of the neural network, we get a new artificial reference picture, which replaces the picture in the reference picture list with the highest temporal difference to the next picture to be coded following [2]. The integration of our neural network into VTM is depicted in Fig. 1. No other changes to the encoding and decoding process or to the signaling are necessary, because the motion-compensated prediction can deal with the artificial reference pictures in the same way as with conventional reference pictures.

TABLE I: BD-rates and coding time ratios relative to the VTM 7.0 anchor for six sequences from the KITTI [9] data set split into two classes. Sequences in curve class contain scenes with lane changes, tight curves and road crossings, while sequences in straight class contain predominantly scenes driven on a straight track. Negative BD-rates indicate increased coding efficiency. Coding time ratios > 1 indicate increased complexity.

Cost	Class	KITTI drive	BD-Rates				Time ratios	
			Y	Cb	Cr	Weighted	Enc.	Dec.
PredNet	Curve	5	−0.10%	0.71%	−0.05%	−0.04%	1.01	3.74
		35	−0.97%	2.79%	0.93%	−0.52%	1.00	4.31
		46	0.17%	1.47%	0.22%	0.28%	1.10	4.39
		Mean	−0.30%	1.66%	0.36%	−0.10%	1.06	4.15
	Straight	13	−0.35%	−2.20%	0.22%	−0.45%	1.01	3.49
		27	−0.68%	−2.03%	1.09%	−0.63%	1.07	3.41
		16	−0.49%	−1.60%	0.40%	−0.50%	1.03	4.18
		Mean	−0.49%	−1.94%	0.57%	−0.52%	1.04	3.69
SATD	Curve	5	−0.20%	0.26%	−2.14%	−0.30%	1.06	3.85
		35	−1.06%	3.23%	−1.65%	−0.78%	1.00	4.34
		46	0.12%	2.51%	0.89%	0.34%	1.10	4.60
		Mean	−0.38%	2.00%	−0.97%	−0.25%	1.05	4.26
	Straight	13	−0.58%	−3.27%	0.80%	−0.70%	1.03	3.64
		27	−0.99%	−5.82%	0.29%	−1.32%	1.08	3.43
		16	−0.74%	−2.93%	0.28%	−0.82%	1.04	3.99
		Mean	−0.77%	−4.01%	0.46%	−0.94%	1.05	3.69

IV. Evaluation

In this section, we discuss the results of our modified VVC reference software VTM 7.0 in low-delay configuration with two trained neural networks. One network was trained to minimize the SATD between predicted and original picture.

Fig. 3: Usage of the artificial reference picture generated by the neural network trained to optimize the SATD in VTM as a heat map. Linearly scaled from no usage (white) to highest usage (purple).

The other network was trained with the original PredNet cost function as described in [6]. Exemplary pictures generated by both networks are shown in Fig. 2. While both networks were trained using sequences of front-viewing train and subway drives, for the evaluation we used sequences from the KITTI data set [9] containing front-viewing car drives. We chose these sequences, because they are available as uncoded raw data. The affine motion model of VVC's motion-compensated prediction is not optimal for the KITTI sequences showing scenes with a camera moving towards a vanishing-point. Thus, we use these sequences to demonstrate the capability of our method. Since our neural networks were trained with a different data set than we evaluate on, we can show that our method is capable to generalize to unseen content.

We defined two classes (curve and straight) which each contain 3 sequences from the KITTI data set. The curve class contains the KITTI sequences 5, 35 and 46, which mainly contain scenes with lane changes, tight curves and road crossings. The straight class contains the KITTI sequences 13, 27 and 16, which predominately depict scenes driven on a straight track. Since the aspect ratio of these sequences differs from the aspect ratio of the videos our networks were trained on, we cropped a central 16:9 window of maximal size and scaled these down to a resolution of 256x144.

To calculate the coding efficiency, we encoded and decoded each of our 6 evaluation sequences with our modified VTM using both neural network variants with QP values 22, 27, 32, 37 following the common test conditions [10]. BD-rates were calculated according to [11]. Additionally, weighted BD-rates were calculated with weighting factors 6/1/1 for the three PSNR values of Y/Cb/Cr following [8]. The results are shown in Table I. BD-rate gains up to 1.06% for luma and 5.82% for chroma are measured. On average the weighted BD-rate for the neural network trained with the original PredNet cost function is -0.10% for curve class and -0.52% for straight class. The neural network trained with our SATD cost function has average weighted BD-rates of -0.25% for curve class and -0.94% for straight class. For both cost functions, the sequences from straight class lead to lower, thus better, BD-rates than sequences from curve class. This is expected, since both neural networks were trained using train sequences consisting of mostly straight tracks. Furthermore, the motion of a straight movement is easier to model than a curved movement. The mean encoding complexity of our method is only 5% higher than the unmodified VTM anchor. The mean decoding complexity increased by 298%. It should be noted that the neural network adds the same absolute coding complexity to the encoder and decoder. This leads to a higher increase in relative coding complexity for the decoder compared to the encoder, since the absolute decoding complexity is significantly lower than the absolute encoding complexity.

To show in which areas of a picture our method has the highest impact on the coding efficiency, we overlayed a heat map of the block-wise average usage of the artificial reference picture for the sequence KITTI drive 13 encoded with our modified VTM using the neural network trained to optimize the SATD over an example picture from that sequence in Fig. 3. Since this sequence is depicting a drive on a straight track, the positions of the road, road markings and forest do not significantly change over time. It can be observed, that the highest usage is on the road section and in the areas perpendicular to the road. If a front-viewing camera is moved in a straight line parallel to a plane, temporally different picture blocks depicting the same position on that plane can only be transformed into each other using a perspective transformation. Since VVC only supports an affine motion model, our method is expected to outperform the non-modified VTM implementation in these cases. In our example, the road is planar and thus fits this model, which explains the high usage of the artificial reference picture. The trees perpendicular to the road are inhomogeneous and thus harder to predict.

In [2], Laude *et al.* achieved average weighted BD-rate gains up to 1.54% using a similar approach implemented in the HEVC reference software HM 16.18. With our aproach, an average weighted BD-rate gain of up to 0.94% was measured for the straight class. Our lower BD-rate gain compared to Laude *et al.* is mainly due to technical improvements of VVC over HEVC, especially the new affine motion model of VVC's motion-compensated prediction. Although the affine model is not capable of describing the motion in our evaluation scenes accurately, it may be a relatively good approximation.

V. Conclusions

In this paper, we propose an enhanced machine learning-based inter coding algorithm for VVC. Conceptually, the reference pictures from the decoded picture buffer are processed using a recurrent neural network to generate an artificial reference picture at the time instance of the currently coded picture. We selected a SATD cost function specifically suited for video coding. By this, we outperform traditional cost functions commonly employed for the training of neural networks. BD-rate gains of up to 1.06% for luma and up to 5.82% for chroma are measured (0.94% and 0.25% average weighted BD-Rate gains for straight and curve class, respectively). The increase in coding time is moderate with 5% for the encoder and 298% for the decoder.

REFERENCES

[1] T. Laude, Y. G. Adhisantoso, J. Voges, M. Munderloh, and J. Ostermann, "A comprehensive video codec comparison," *APSIPA Transactions on Signal and Information Processing*, vol. 8, p. 30, Nov. 2019.

[2] T. Laude, F. Haub, and J. Ostermann, "HEVC inter coding using deep recurrent neural networks and artificial reference pictures," in *Picture Coding Symposium (PCS)*, Nov. 2019.

[3] T. Laude and J. Ostermann, "Deep learning-based intra prediction mode decision for HEVC," in *Proceedings of 32nd Picture Coding Symposium (PCS)*. Nuremberg, Germany: IEEE, 2016.

[4] J. K. Lee, N. Kim, S. Cho, and J.-W. Kang, "Convolution Neural Network based Video Coding Technique using Reference Video Synthesis," in *Proceedings of APSIPA Annual Summit and Conference*. Hawaii, US: APSIA, 2018.

[5] J. Liu, S. Xia, and W. Yang, "Deep Reference Generation with Multi-Domain Hierarchical Constraints for Inter Prediction," *IEEE Transactions on Multimedia*, 2019.

[6] W. Lotter, G. Kreiman, and D. Cox, "Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning," in *International Conference on Learning Representations*, 2017.

[7] Y. Li and X. Mou, "Transform coefficients distribution of the future versatile video coding (VVC) standard," in *Optoelectronic Imaging and Multimedia Technology V*. International Society for Optics and Photonics, 2018.

[8] J. Ström, K. Andersson, R. Sjöberg, A. Segall, F. Bossen, G. J. Sullivan, and J.-R. Ohm, "JVET-Q2016-v4: Summary information on BD-rate experiment evaluation practices. 17th Meeting of the Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11. Brussels, BE," 2020.

[9] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[10] F. Bossen, J. Boyce, K. Suehring, X. Li, and V. Seregin, "JVET-N1010-v1: JVET common test conditions and software reference configurations for SDR video. 14th Meeting of the Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11. Geneva, CH," 2019.

[11] G. Bjøntegaard, "VCEG-AI11: Improvements of the BD-PSNR model. ITU-T Study Group 16 Question 6. 35th Meeting, Berlin, Germany," 2008.