End-to-end Optimized Video Compression with MV-Residual Prediction

XiangJi Wu[1], Ziwen Zhang[1], Jie Feng[1], Lei Zhou[1], Junmin Wu[1]
[1]Tucodec Inc
{wuxiangji}@tucodec.com

Abstract

We present an end-to-end trainable framework for P-frame compression in this paper. A joint motion vector (MV) and residual prediction network MV-Residual is designed to extract the ensembled features of motion representations and residual information by treating the two successive frames as inputs. The prior probability of the latent representations is modeled by a hyperprior autoencoder and trained jointly with the MV-Residual network. Specially, the spatially-displaced convolution is applied for video frame prediction, in which a motion kernel for each pixel is learned to generate predicted pixel by applying the kernel at a displaced location in the source image. Finally, novel rate allocation and post-processing strategies are used to produce the final compressed bits, considering the bits constraint of the challenge. The experimental results on validation set show that the proposed optimized framework can generate the highest MS-SSIM for P-frame compression competition.

1. Introduction

Recently, artificial neural networks (ANNs) have been applied to solve the image and video compression problem and a number of works have been proposed [11, 6, 2, 3, 5, 9, 18]. Recent studies in deep learning based image compression methods have achieved significant performance improvement and they focus on designing end-to-end optimized frameworks [1, 15, 11, 6], in which the modules such as transformation, quantization and entropy estimation are optimized jointly. It is therefore not surprising to see that Deep Neural Networks (DNN) have attracted attention for solving video compression tasks. Many approaches [17, 8, 7] were proposed to replace the components in traditional video codecs by DNNs. For example, Liu *et al* [8] utilized a DNN in the fractional interpolation of motion compensation, and [7] applied DNNs to improve performance of the in-loop filter. End-to-end video compression frameworks have also been studied widely and various approaches have been proposed. For example Wu *et al.* presented a framework for predicting frames by interpolation from reference frames, and then the image compression network of was applied to compress the residual. In 2019, Lu *et al.* [9] proposed the Deep Video Compression (DVC) method, in which optical flow was used to predict the temporal motion, and then two compression subnetworks were designed to compress the motion and residual. In order to realize spatial-temporal energy compaction in learning image and video compression, a spatial-temporal energy compaction was incorporated into the loss function to improve the video compression performance in [3]. Meanwhile, Habibian *et al.* [5] firstly employed a model that consisted of a 3D autoencoder with a discrete latent space and an autoregressive prior for video compression. In [2], the concept of Pixel-MotionCNN (PMCNN) which includes motion extension and hybrid prediction networks was proposed to design more robust motion prediction moduel. PMCNN can model spatiotemporal coherence to effectively perform predictive coding inside the learning network. Different from these methods which are trained with one loss function applied on all frames, Yang *et al.* [18] proposed a Hierarchical Learned Video Compression (HLVC) method with three hierarchical quality layers and a recurrent enhancement network. The video frames are compressed in the hierarchical layers 1, 2 and 3 with decreasing quality, using an image compression method for the first layer and the proposed BDDC and SMDC networks for the second and third layers, respectively.

In CLIC 2020 P-frame compression challenge, we propose a novel video compression framework which consists of a MV-Residual prediction network for video framework prediction and a post-processing module for visual quality enhancement. The MV-Residual prediction network is capable of estimating motion vectors and residual information simultaneously. Moreover, the techniques such as hyperprior base rate estimation, soft quantization and resource allocation which were proposed by Balle et al. and Mentzer et al. [1, 10] have also been utilized to improve the compression performance.

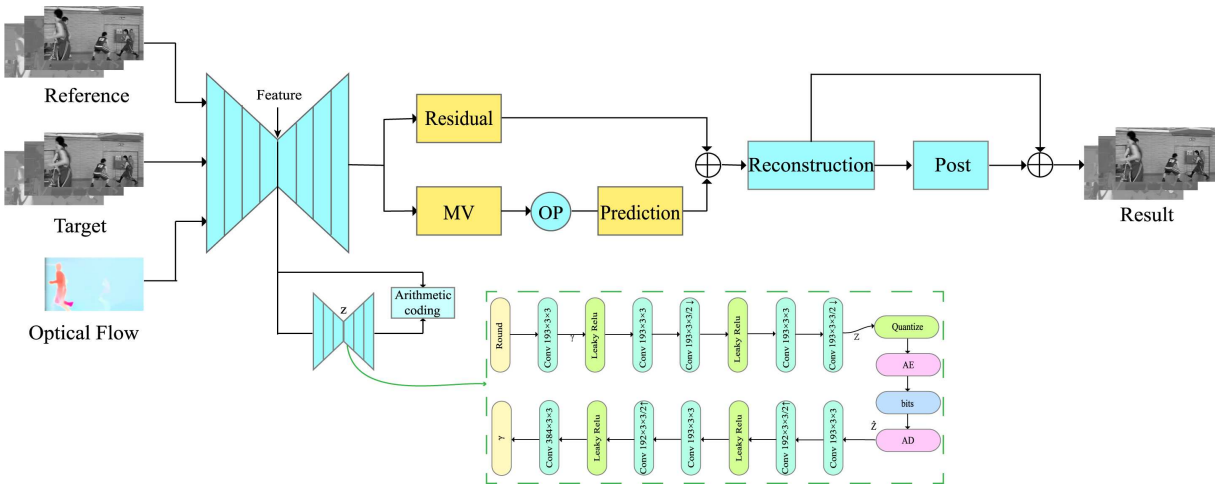

Figure 1. Illustration of the proposed architecture. Input data is fed into the MV Encoder-decoder network and the hyperprior network works after encoding. Details for hyperprior autoencoder is shown in the dotted frame, convolution parameters are denoted as number of filter kernel height kernel width / down or upsampling stride, where ↓ indicates downsampling and ↑ indicates upsampling. AE, AD represent arithmetic encoder and arithmetic decoder. OP operation stands for SDC-Net. [12, 19]

2. End-to-end Optimized Video Compression with MV-Residual Prediction

2.1. Overview of the Proposed Model

The proposed model is based on our CVPR 2019 CLIC framework in Low-rate compression [19]. Fig 1 provides an overview of our end-to-end video compression architecture, which can be optimized in an end-to-end manner. The brief summarization on the working process is introduced as follows:

Step1. Data preprocessing. In this year's competition, the validation and test sets are subsets of the training set. Thus, any unnecessary modifications to the training data would not benefit the final result. To gain better performance and get rid of any unnecessary loss, we simply up-sampled the given pictures with format of YUV420 to YUV444 as reference frame and target frame. The optical flow is generated with pretrained PWC-Net [14].

Step2. MV Encoder-decoder network. In order to encode the motion information, we design an auto-encoder style CNN for better encoding. In this step, after a series of convolution operations and nonlinear transformations, representations of the motion will be generated. Then, the latent feature is quantized and fed into a hyperprior autoencoder to obtain prior probability, which is fully discussed in Section 2.3.

Step3. Motion estimation. An OP operation is designed to obtain the prediction based on the motion vector calculated by the previous network. To avoid blurry reconstruction, we utilized SDC-net [12] as the OP operation, which is fully differentiable and thus allows our model to train end-to-end. More information is provided in Section 2.2.

Step4. Post processing. With reconstruction obtained by summing residual and prediction, we employ a ResNet style network to optimize our result to get final output.

Step5. Variable rate. In order to make full use of every bit space available, a rate control module [4] is utilized to fit our model to variable compression rates with a single set of weights. A rate control parameter, Lagrange multiplier is used as a conditional input for our end-to-end model, and contributes to our loss function for better optimization.

2.2. Spatially-displaced Convolution

Given the reference frame and decoded motion vector produced by previous stage, a simple way to estimate motion is to calculate vector-based transformation. However, as discussed in [12], such operation will lead to insufficient representation of the motion and end in blurry results. Following [12], we use SDC-net to estimate motion:

$$I_{t+1} = \mathcal{T}(\mathcal{G}(I_{1:t}, F_{2:t}), I_t), \tag{1}$$

where $\mathcal{T}$ is realized with SDC operating on the reference frame I_t and F_i refers to decoded optical flow. $\mathcal{G}$ is a fully convolutional network which takes in a sequence of past frames $I_{1:t}$ and outputs pixel-wise separable kernels K_u, K_v In this way, predictions of multiple frames will be extended naturally by recirculating new inputs.

2.3. Rate Estimation Module

We model each latent $\hat{y}_i$ as a Laplacian distribution with mean and scale parameters μ_i, σ_i convolved with a unit uni-

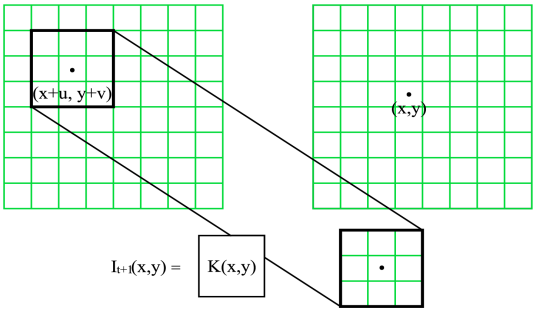

Figure 2. Illustration of Spatial Displaced Convolution.

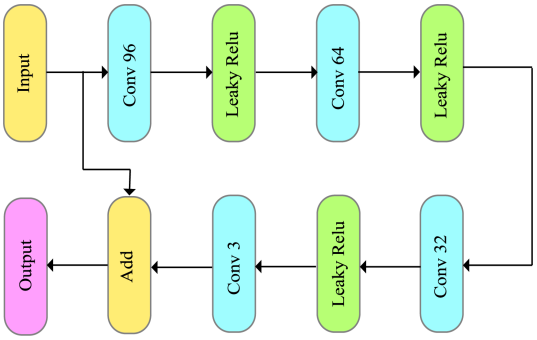

Figure 3. Illustration of Post Processing. Conv n represents the convolution operation with the output channel of n, input is reconstructed pictures.

form distribution. This ensures a good match between encoder and decoder distributions of both the quantized latents. Based on our solution in [19], both the hyperprior as well as the causal context of each latent $\hat{y}_i$ are designed to predict the Laplacian parameters. Furthermore, the rate estimation module is a hyperprior network H with parameter Θ_h. The predicted Laplacian parameters are functions of learned parameters Θ_h:

$$p_{\hat{y}}(\hat{y}|\hat{z},\Theta_h) = \prod_i (Lap(\mu_i,\sigma_i^2) * U(-\frac{1}{2},\frac{1}{2}))(\hat{y}_i), \quad (2)$$

where $\mu_i, \sigma = h_d(\hat{z};\Theta_h)$ is the output of hyperprior network.

Hyperprior Network H: The subsampled feature y is fed into the hyperprior encoder which summarizes the distribution of standard deviations in $z = h_e(y)$. z is then quantized $\hat{z} = Q(z)$, compressed and transmitted as side information. The final layer of hyperprior network must have exactly twice as many channels as the bottleneck, so as to predict two values: the mean and scale of a Laplacian distribution for each latent. As to the distribution of $\hat{z}$, we model it as a non-parametric and fully factorized density model because there doesn't exist prior knowledge for $\hat{z}$, , similar to the strategy used in [1]:

$$p_{\hat{z}|\psi}(\hat{z}|\psi) = \prod_i (P_{z_i|\psi_i}(\psi_i) * \mu(-\frac{1}{2},\frac{1}{2}))(\hat{z}_i), \quad (3)$$

where the vector ψ_i represents the parameters of each univariate distribution $P_{z_i|\psi_i}$.

Finally, the compression rates are composed of two part: rate R_y of compressed representation $\hat{y}$ and rate R_z of compressed side information $\hat{z}$. These rates are defined as follows:

$$\begin{aligned} R_y &= \sum_i -log_2(p_{\hat{y}}(\hat{y}|\hat{z},\Theta_h,\Theta_{cm},\Theta_{ep})), \\ R_z &= \sum_i -log_2(p_{\hat{z}_i|\psi}(\hat{z}|\psi)) \end{aligned} \quad (4)$$

2.4. Post Processing

In order to further improve the performance of our frame, a CNN model is used as our post processing after reconstruction finished. As illustrated in Figure 3, we design the CNN with four layers and the final output with three channels responding to three dimensions of YUV444, the same as the input.

2.5. Optimized Rate Control

Rate-Distortion optimization is a common strategy in compression algorithms. The rate-control strategy is similar to our CLIC 2019 solution used [19]. Considering the bits constraint, a rate control optimization problem is defined to allocate the bits more effectively for each frame:

$$max_{j\in M} \sum_{i=1}^{N} MS_j(x_i,\hat{x}_i) \;\; st. \sum_i R_j^i < R_{max}, \quad (5)$$

where MS represents the MS-SSIM calculated between original frame x_i and the reconstructed frame $\hat{x}_i$. M is the vector set which contains all possible quality configurations for the set of frames. N is the frame number. MS_j and R_j are the performances and rates under configuration j. The best quality configuration is selected for each image via optimizing Eq (5) in our implementation. The rate control problem is optimized using dynamic programming algorithm.

3. Experimental Results

For training, 463686 image pairs are selected. where each training sample consists of two consecutive frames with the last frame serving as the ground truth. These images are random sampled to 256 × 256 pixels to train the network. Our team have submitted three solutions: Tucodec, TUCODEC_SSIM, and TucodecVideo. The results

Table 1. Evaluation results on CLIC 2020 P-frame validation datasets.

	Methods	**PSNR**	**MS-SSIM**	**Data Size**	**Decoder Size**	**Decoding Time**
Validation	TUCODEC_SSIM	37.028	0.9969	37870015	112854058	6749
	TucodecVideo	37.026	0.9969	37870015	84778538	N/A
	Tucodec	37.022	0.9968	37870012	112847016	6954

of the validation sets are reported in Table 1. In our implementations, the cluster number is set as 200 for the soft quantization and only one distortion measures perceptional loss are used to train the autoencoder.

$$L = \lambda D + R_y + R_z, \tag{6}$$

In TUCODEC_SSIM the loss $D = 1 - L_{msssim}$ is defined for the perceptional loss where L_{msssim} is as defined in [16]. Then the perceptional loss is combined with the same GAN setup defined in [13] for network optimization. Then five rates with λ=20/22/24/26/28 are trained for 5 rate control. Once the resource allocation is done, MS-SSIM of 0.9969 can be achieved for validation under the constraint of less than 3,900,000,000 bytes with the model size of 112,854,058 bytes. Since there is not enough time, we first submitted a version Tucodec, which is only iteration numbers different from the TUCODEC_SSIM. Further, we quantize the model with 16 bits, it can compress the model with almost no reduction in accuracy. However, this model have not decoded successfully on clic server, but it can run perfectly on our local docker. This model achieve MS-SSIM 0.9969 for validation with the model size of 84,778,538 bytes.

4. Conclusion

In this paper, a novel deep learning based video compression framework which contains a MV-Residual prediction network and a post-processing module is designed for CLIC 2020 challenge. In the MV-Residual prediction network, the motion vectors and residual information are predicted simultaneously. The motion kernels can be learnt by spatially-displaced convolutions to predict pixels in the P-frame by applying the kernels at a displaced locations in the source image. The experiments show that the MV-Residual prediction network can improve the compression performance by modeling the spatial correlation between frames accurately. As shown in the results of the challenges on the validation set, our approaches TUCODEC_SSIM and Tucodec rank the $1st$ and $2nd$ place in P-frame compression challenge for best MS-SSIM.

References

[1] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018. 1, 3

[2] Z. Chen, T. He, X. Jin, and F. Wu. Learning for video compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 2019. 1

[3] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto. Learning image and video compression through spatial-temporal energy compaction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10071–10080, 2019. 1

[4] Y. Choi, M. El-Khamy, and J. Lee. Variable rate deep image compression with a conditional autoencoder. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3146–3154, 2019. 2

[5] A. Habibian, T. v. Rozendaal, J. M. Tomczak, and T. S. Cohen. Video compression with rate-distortion autoencoders. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7033–7042, 2019. 1

[6] J. Lee, S. Cho, and S.-K. Beack. Context-adaptive entropy model for end-to-end optimized image compression. *arXiv preprint arXiv:1809.10452*, 2018. 1

[7] T. Li, M. Xu, R. Yang, and X. Tao. A densenet based approach for multi-frame in-loop filter in hevc. In *2019 Data Compression Conference (DCC)*, pages 270–279. IEEE, 2019. 1

[8] J. Liu, S. Xia, W. Yang, M. Li, and D. Liu. One-for-all: Grouped variation network-based fractional interpolation in video coding. *IEEE Transactions on Image Processing*, 28(5):2140–2151, 2018. 1

[9] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, and Z. Gao. Dvc: An end-to-end deep video compression framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11006–11015, 2019. 1

[10] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. Van Gool. Conditional probability models for deep image compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4394–4402, 2018. 1

[11] D. Minnen, J. Ballé, and G. D. Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, pages 10771–10780, 2018. 1

[12] F. A. Reda, G. Liu, K. J. Shih, R. Kirby, J. Barker, D. Tarjan, A. Tao, and B. Catanzaro. Sdcnet: Video prediction using spatially-displaced convolution. *CoRR*, abs/1811.00684, 2018. 2

[13] O. Rippel and L. Bourdev. Real-time adaptive image compression. *arXiv preprint arXiv:1705.05823*, 2017. 4

[14] D. Sun, X. Yang, M. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. *CoRR*, abs/1709.02371, 2017. 2

[15] L. Theis, W. Shi, A. Cunningham, and F. Huszár. Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*, 2017. 1

[16] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 2, pages 1398–1402. Ieee, 2003. 4

[17] M. Xu, T. Li, Z. Wang, X. Deng, R. Yang, and Z. Guan. Reducing complexity of hevc: A deep learning approach. *IEEE Transactions on Image Processing*, 27(10):5044–5059, 2018. 1

[18] R. Yang, F. Mentzer, L. Van Gool, and R. Timofte. Learning for video compression with hierarchical quality and recurrent enhancement. *arXiv preprint arXiv:2003.01966*, 2020. 1

[19] L. Zhou, Z. Sun, X. Wu, and J. Wu. End-to-end optimized image compression with attention mechanism. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 2, 3