Deep Contextual Video Compression

Anonymous Author(s)
Affiliation
Address
email

Abstract

Although great progress has been recently made in deep video compression, most existing works still adopt the predictive coding framework, which first generates the predicted frame and then encodes its residue with the current frame. However, as for compression ratio, predictive coding is a sub-optimal solution by using hand-crafted coding modules to explore correlations across frames. For this reason, we propose a paradigm shift from traditional predictive coding to direct conditional coding. Different from most existing methods, our method focuses on how to define, learn, and use the context rather than generating an accurate predicted frame. For obtaining reconstructed frame with high quality, this paper proposes using context in the feature domain as the condition. Context with higher dimension is able to carry richer information to help the conditional coding, especially for the high-resolution videos containing many high-frequency textures. Experiments show that our method can significantly outperform the previous state-of-the-art (SOTA) models based on deep learning. When compared with x265 using veryslow preset, we can achieve 26.0% bitrate saving for 1080P standard test videos. In addition, it is noted that the proposed framework is extensible, where the condition can be flexibly designed. More explorations can be conducted upon our framework.

1 Introduction

From H.261 [1] developed in 1988 to the just released H.266 [2] in 2020, all traditional video coding standards are based on a predictive coding paradigm, where the motion compensated predicted frame is first generated by handcrafted modules and then the residue between the current frame and the predicted frame is encoded and decoded. Recent many deep learning (DL)-based video compression methods [3–11] also adopt the predictive coding framework to encode the residue where all modules in traditional video codec are merely replaced by neural network.

The residual coding is a simple yet efficient framework for video compression, considering the strong temporal correlations among frames. However, residual coding is not optimal to encode the current frame x_t given the predicted frame $\tilde{x}_t$. The entropy of residual coding is greater than or equal to that of conditional coding [12]: $H(x_t - \tilde{x}_t) \geq H(x_t|\tilde{x}_t)$, where H represents the Shannon entropy. Theoretically, one pixel in frame x_t correlates to all the pixels in the previous decoded frames and the pixels already been decoded in the current frame. For traditional video codec, it is impossible to use the handcrafted rules to explicitly explore the correlations by taking all of them into consideration due to the huge space. Thus, residual coding is widely adopted as a special extremely simplified case of conditional coding, with the very strong assumption that the current pixel only has the correlation with the predicted pixel. DL opens the door to automatically explore correlations in a huge space. Considering the success of DL in image compression [13, 14] just using autoencoder to explore the correlations among pixels, why not use network to build the conditional coding-based framework for deep video compression rather than restricting our vision into residue coding?

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

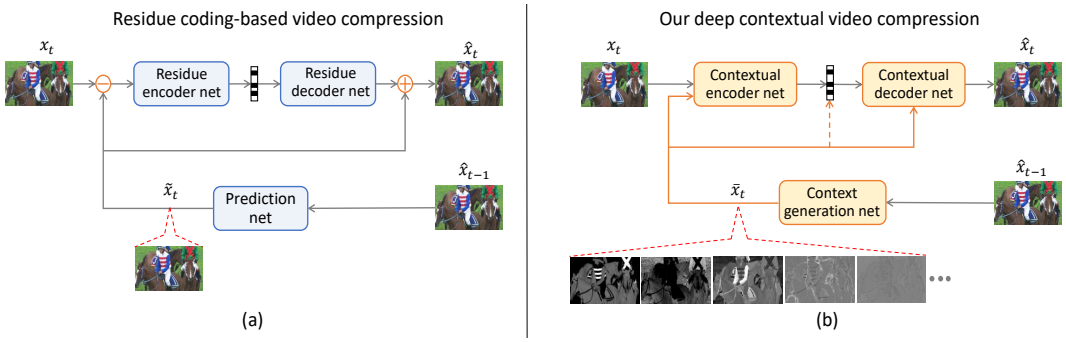

Figure 1: Framework comparison between the commonly-used residue coding-based framework and our framework. The dashed line means that the context is also used for entropy model. x_t is the current frame. $\hat{x}_t$ and $\hat{x}_{t-1}$ are the current and previous decoded frames. In our design, we propose using network automatically learn the condition in feature domain. Context with higher dimension can contain richer information to help the following conditional coding. In particular, different channels in context also have the freedom to provide different kinds of information.

When we design the conditional coding-based framework, a series of questions naturally come up: *What is condition? How to use condition? And how to learn condition?* Technically speaking, condition can be anything that may be helpful to compress the current frame. The predicted frame can be used as condition but it is not necessary to limit the representation of condition. Thus, we define the condition as learnable contextual features with arbitrary dimension. Along this idea, we propose a deep contextual video compression (DCVC) framework to utilize condition in a unified, simple, yet efficient way. The diagram of our DCVC framework is shown in Fig. 1. The contextual information is used as part of the input of contextual encoder, contextual decoder, as well as the entropy model. In particular, benefiting from the temporal prior provided by context, the entropy model itself is temporally adaptive, resulting in a richer and more accurate model. As for how to learn condition, a simple convolutional network is not able to extract adequate helpful feature, especially when encoding video with complex motion. Inspired by the motion estimation (ME) in pixel domain [4], we propose using ME to search the most correlated region in feature domain. Then the motion compensation (MC) is used to align the correlated feature and then generate the final context. The MEMC can guide the model where to extract useful context. Experimental results demonstrate the effectiveness of the proposed DCVC. For 1080p standard test videos, our DCVC can achieve 26.0% bitrate saving over x265 using veryslow preset, and 16.4% bitrate saving over previous SOTA DL-based model DVCPro [4].

Actually, the concept of conditional coding has appeared in [15] and [12, 16]. However, these works are only designed for partial module (e.g. only entropy model) or need hand-crafted operations to filter which content should be conditionally coded. By contrast, our framework is a more comprehensive solution which considering all of encoding, decoding, and entropy model. In addition, to best of our knowledge, the proposed DCVC is the first extensible conditional coding-based framework, where the condition can be flexible designed. Although this paper proposes using feature domain MEMC to generate contextual feature and demonstrates its effectiveness, we still think it is a key and open question worth further investigation to boost the compression ratio.

Our main contributions are three-folded:

- We design a deep contextual video compression framework based on conditional coding. The definition, usage, and learning manner of condition are all innovative. Our method can achieve higher compression ratio than previous residual coding-based methods.
- We propose a simple yet efficient way using context to help the encoding, decoding, as well as the entropy modeling. For entropy modeling, we jointly utilize the spatial and temporal correlation, then more accurate probability estimation can be obtained.
- We define the condition as the context in feature domain. The context with higher channel dimension can provide richer information to help the conditional coding.

2 Related Works

Deep Image Compression Recently there are many works for deep image compression. For example, the compressive auto encoder [17] could get comparable results with JPEG 2000. Subsequently, many works boost the performance by more advanced entropy models and network structures. For example, Ballé *et al.* proposed the factorized [18] and hyper prior [13] entropy models. The method based on hyper prior gets comparable results with H.265 intra coding. In 2018, the entropy model jointly utilizing hyper prior and auto regressive context was proposed and able to outperform H.265 intra coding. The method with Gaussian mixture model [19] gets comparable with H.266 intra coding. For the network structure, some RNN (recurrent neural network)-based methods [20–22] were proposed in the early development stage, but most of recent methods are based on CNN.

Deep Video Compression Existing works for deep video compression can be classified into two categories, i.e. non-delay-constrained and delay-constrained. For the first category, there is no restriction on reference frame location, which means that the reference frame can be from future. For example, Wu *et al.* [11] proposed interpolating the predicted frame from previous and future frames, and then the frame residue is encoded. Djelouah *et al.*[9] also followed this coding structure and introduced the optical flow estimation network to get better predicted frame. Yang *et al.* [6] designed a recurrent enhancement module for this coding structure. In addition, 3D auto encoder was proposed to encode group of picture in [23, 24]. This is a natural extension of deep image compression by increasing the dimension of input. It is noted that this coding manner will bring larger delay and the GPU memory cost will be significantly increased. For the delay-constrained methods, the reference frame is only from the previous frames. For example, Lu *et al.* [3] designed the DVC model, where all modules in traditional hybrid video codec are replaced by network. Then the improved model DVCPro which adopts more advanced entropy model from [14] and deeper network was proposed in [4]. Following the similar framework with DVC, Agustsson *et al.* designed a more advanced optical flow estimation in scale space. Hu *et al.* [25] considered the rate distortion optimization when encoding motion vector (MV). In [7], the single reference frame is extended to multiple reference frames. Recently, Yang *et al.* [8] proposed an RNN-based MV/residue encoder and decoder.

Our research belongs to the delay-constrained method because it can be applied in more scenarios, e.g. real time communication. Different from above works, we design a conditional coding-based framework rather than following the commonly-used residue coding. Actually, recent works in [15] and [12, 16] have some investigations on condition coding. In [15], the conditional coding is only designed for entropy model. However, due to the lack of MEMC, the compression ratio is not high. From the PSNR-Bitrate curve shown in [15] , the method in [15] cannot outperform DVC. By contrast, our conditional coding designed for encoding, decoding, and entropy model can significantly outperform DVCPro. In [12, 16], the input frame needs to be explicitly separated into foreground and background images, where the predicted frame is only used for the conditional coding of foreground image. By contrast, our method does not needs the handcrafted operation to filter the current frame. In addition, the condition in DCVC is the context in feature domain, which has much larger capacity and is able to provide different kinds of information. In summary, when compared with [15] and [12, 16], the definition, usage, and learning manner of the condition in DCVC are all innovative.

3 Proposed Method

In this section, we present the details of the proposed DCVC. We first describe the whole framework of DCVC. Then we introduce the entropy model for compressing the latent codes. The next is the analysis of context and how to learn the context. At last, we give the training strategy.

3.1 The framework of DCVC

In traditional video codec, the inter frame coding adopts the residue coding, formulated as:

$$\hat{x}_t = f_{dec}\big(\lfloor f_{enc}(x_t - \tilde{x}_t)\rceil\big) + \tilde{x}_t \; with \; \tilde{x}_t = f_{predict}(\hat{x}_{t-1}). \tag{1}$$

$\hat{x}_t$ and $\hat{x}_{t-1}$ are the current and previous decoded frames, respectively. f_{enc} and f_{dec} are the residue encoder and decoder. $\lfloor\cdot\rceil$ is the quantization operation. $f_{predict}$ represents the function for generating the predicted frame $\tilde{x}_t$. In traditional video codec, $f_{predict}$ is implemented in the manner of MEMC, which uses handcrafted coding tools to search the best MV and then interpolates the predicted frame.

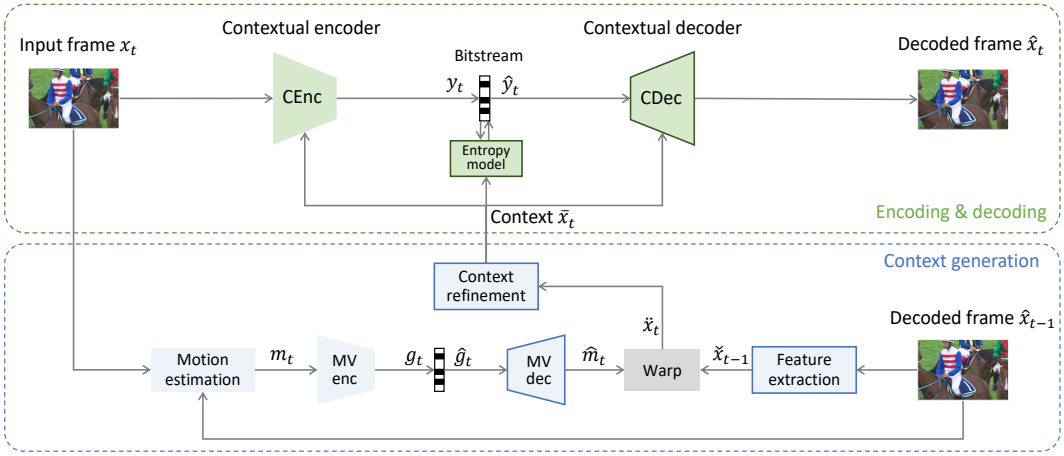

Figure 2: The framework of our DCVC. The modules with blue color are used to generate context. The modules with green color are encoder and decoders given the context. The video decoding process only involves the modules with bold border.

For most existing DL-based video codecs [3–10], $f_{predict}$ is the MEMC which is totally composed of network.

In this paper, we do not adopt the commonly-used residue coding but try to design a conditional coding-based framework for higher compression ratio. Actually, one straightforward conditional coding manner is directly using the predicted frame $\tilde{x}_t$ as the condition:

$$\hat{x}_t = f_{dec}\big(\lfloor f_{enc}(x_t|\tilde{x}_t)\rceil \mid \tilde{x}_t\big) \; with \; \tilde{x}_t = f_{predict}(\hat{x}_{t-1}). \tag{2}$$

However, the condition is still restricted in pixel domain with small channel dimension. This will limit the model capacity. Now that the conditional coding is used, why not let model learn the condition by itself. Thus, this paper proposes a contextual video compression framework, where we use network to generate context rather than predicted frame. Our framework can be formulated as:

$$\hat{x}_t = f_{dec}\big(\lfloor f_{enc}(x_t|\bar{x}_t)\rceil \mid \bar{x}_t\big) \; with \; \bar{x}_t = f_{context}(\hat{x}_{t-1}). \tag{3}$$

$f_{context}$ represents the function of generating context $\bar{x}_t$. To carry more helpful information, the context is in the feature domain with higher dimension. f_{enc} and f_{dec} are the contextual encoder and decoder. The proposed DCVC framework is illustrated in Fig. 2. As shown in this figure, DCVC can be divided into two main parts. The first part is how to encode and decode the current frame conditioned on the context. The other part is the implementation of $f_{context}$.

For the first part, the encoding and decoding of the current frame are both conditioned on the context $\bar{x}_t$. As shown in Fig. 2, x_t is encoded into 16x down-sampled latent codes y_t conditioned on $\bar{x}_t$. $\hat{y}_t$ is the quantized y_t via rounding operation. Through contextual decoder, the reconstructed frame $\hat{x}_t$ is finally obtained. In our design, we use network to automatically learn the correlation between x_t and $\bar{x}_t$ and then remove the redundancy rather than using fixed subtraction operation in residue coding. From another perspective, our method also has the ability to adaptively use the context. As the variety of video content, some regions in a frame have new objects. In this situation, each block can adaptively choose intra coding or inter coding through rate distortion optimization in traditional hybrid framework. However, the DNN-based video codec with residue coding has not such adaptability. The residue can be very large and the inter coding via subtraction operation may be worse than the intra coding of x_t. By contrast, our conditional coding framework has the capacity to adaptively utilize the condition. If the context is less correlated in some regions, the model can adaptively learn the intra coding in x_t.

In addition, we not only use the context $\bar{x}_t$ to generate the latent codes, but also use it to build the entropy model. More details are introduced in the next subsection.

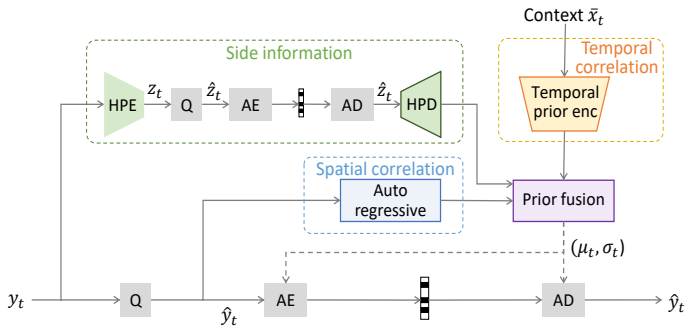

Figure 3: Our entropy model used to encode the quantized latent codes $\hat{y}_t$. HPE and HPD are hyper prior encoder and decoder. Q means quantization. AE and AD are arithmetic encoder and decoder.

3.2 Entropy Model

According to [26], the cross-entropy between the estimated probability distribution and the actual latent code distribution is a tight lower bound of the actual bitrate, namely

$$R(\hat{y}_t) \geq \mathbb{E}_{\hat{y}_t \sim q_{\hat{y}_t}}[-log_2 p_{\hat{y}_t}(\hat{y}_t)], \tag{4}$$

$p_{\hat{y}_t}(\hat{y}_t)$ and $q_{\hat{y}_t}(\hat{y}_t)$ are estimated and true probability mass functions of $\hat{y}_t$, respectively. Actually, the arithmetic coding almost can encode the latent code at the bitrate of cross-entropy. The gap between actual bitrate $R(\hat{y}_t)$ and the bitrate of cross-entropy is negligible. So our target is designing an entropy model which can accurately estimate the probability distribution of latent code $p_{\hat{y}_t}(\hat{y}_t)$. The framework of our entropy model is illustrated in Fig. 3, First, we use the hyper prior model [13] to learn the hierarchical prior of the latent codes and use auto regressive network [14] to learn the spatial prior. For spatial prior, the previous encoded codes $\hat{y}_{t,<i}$ can be used as condition for encoding the i th code $\hat{y}_{t,i}$. The two priors (hierarchical prior and spatial prior) are commonly-used in deep image compression. However, for video coding, the latent codes also have the temporal correlation. For this reason, we propose using the context $\bar{x}_t$ to generate the temporal prior. As shown in Fig. 3, we design a temporal prior encoder to explore the temporal correlation. The prior fusion network will learn to fuse the three different priors and then estimate the mean and scale of distribution. In this paper, we follow the existing work [27] and assume that $p_{\hat{y}_t}(\hat{y}_t)$ follows the Laplace distribution. The formulation of $p_{\hat{y}_t}(\hat{y}_t)$ is:

$$\begin{aligned} p_{\hat{y}_t}(\hat{y}_t|\bar{x}_t, \hat{z}_t) &= \prod_i \left(\mathcal{L}(\mu_{t,i}, \sigma_{t,i}^2) * \mathcal{U}(-\frac{1}{2}, \frac{1}{2})\right)(\hat{y}_{t,i}) \\ with \;\; \mu_{t,i}, \sigma_{t,i} &= f_{pf}\big(f_{hpd}(\hat{z}_t), f_{ar}(\hat{y}_{t,<i}), f_{tpe}(\bar{x}_t)\big). \end{aligned} \tag{5}$$

f_{hpd} is the hyper prior decoder network. f_{ar} is the auto regressive network. f_{tpe} is the specially designed temporal prior encoder network. f_{pf} denotes the prior fusion network. In our entropy model, $f_{ar}(\hat{y}_{t,<i})$ and $f_{tpe}(\bar{x}_t)$ provide the spatial and temporal priors, respectively. $f_{hpd}(\hat{z}_t)$ provides supplemental side information which cannot be learned from spatial and temporal correlation.

3.3 Context in Feature Domain

For residue coding-based framework, the $\bar{x}_t$ must be predicted frame in pixel domain, and its channel dimension can only be fixed, e.g. 3 for commonly-used RGB or YUV formats. However, in our DCVC framework, $\bar{x}_t$ is context in feature domain with arbitrary dimension. Context with larger channel dimension has larger capacity to carry more useful information, means bringing richer condition for frame encoding and improving the probability estimation of the latent codes. In addition, due to the large capacity of context, different channels have the freedom to extract different kinds of feature to help the compression. Here we give an analysis example in Fig. 4. In the figure, the first column shows the current frame x_t and the traditional predicted frame in pixel domain. Because residue coding peruses the minimization of residue, the predicted frame in pixel domain is similar with x_t. In Fig. 4, the second and third columns show the four channel examples in context learned by DCVC. Looking into the contents in four channels, we can find that different channels have different focuses. For example, the top-left channel focuses on extracting large textures, e.g.

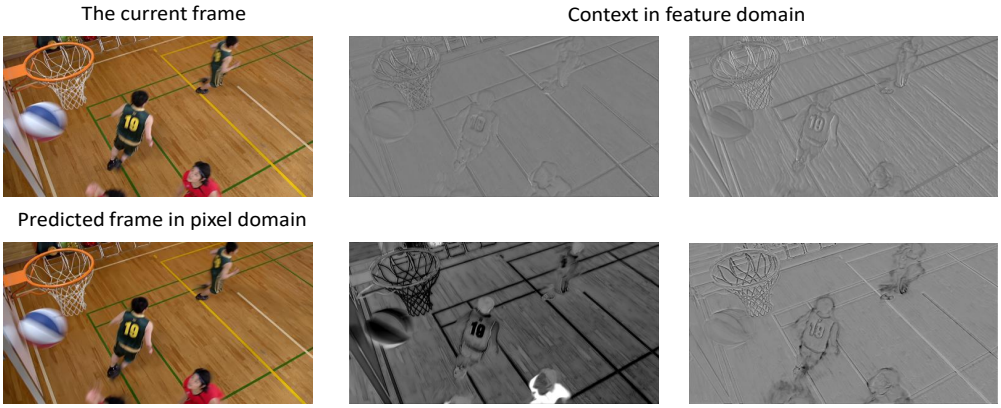

Figure 4: Visual comparison between predicted frame in pixel domain and context in feature domain. In the first column, the above image is the current original frame (the third frame in *BasketballDrill* sequence, HEVC Class C), and the below image the predicted frame in pixel domain. The four images in the second and third column are four channel examples in the feature domain context, where the total channel number is 64.

on the basketball players and court line. Besides the large textures, the top-right channel provides more small texture details (the floor textures). The below-left channel seems to focus on the red color because objects with red color (i.e. the below basketball player and the below edge of basketball) have high intensity. The below-right channel gives more emphases on the three basketball players, especially for the leg parts. Actually, these parts has large motion. For this reason, we think our DCVC model focuses on learning motion context in the below-right channel. In summary, context in feature domain is able to carry richer condition information and each channel can extract different kinds of feature.

For how to design the context extraction function $f_{context}$, one alternative solution is directly using a plain network, where the input is the reference frame $\hat{x}_{t-1}$ and the output is $\bar{x}_t$. However, it is hard for a plain network to extract useful information when there is no supervision on learning. Video often contains various content and there exists a lot of complex motion. For a position in the current frame, the collocated position in the reference frame may have less correlation. In this situation, the collocated position in context $\bar{x}_t$ is also less correlated to the position in x_t, and the less correlated context cannot facilitate the encoding of x_t. For this reason, we also adopt the idea of MEMC. We use MV to guide the network where to extract the context. The context extraction function $f_{context}$ is designed as:

$$f_{context}(\hat{x}_{t-1}) = f_{cg}\big(warp(f_{fe}(\hat{x}_{t-1}), \hat{m}_t)\big) \tag{6}$$

We first design a feature extraction network $\check{x}_{t-1} = f_{fe}(\hat{x}_{t-1})$ to convert the reference frame from pixel domain to feature domain. At the same time, we use the optical flow estimation network [28] to learn the MV between the reference frame $\hat{x}_{t-1}$ and the current frame x_t. The MV m_t is then encoded and decoded. The decoded $\hat{m}_t$ guides the network where to extract the context through warping operation $\ddot{x}_t = warp(\check{x}_{t-1}, \hat{m}_t)$. The $\ddot{x}_t$ is kind of relatively rough context because the warping operation may introduce some spatial discontinuity. Thus, we design a context generation network f_{cg} to obtain the final context $\bar{x}_t = f_{cg}(\ddot{x}_t)$. In addition, the MV also enables network to learn context from a larger region when compared with the solution without MEMC.

For the entropy model of MV latent codes, we just use the entropy model [14] in image compression.

3.4 Training

The target of video compression is using least bitrate cost to get the best reconstruction quality. Thus the training loss consists of two metrics:

$$L = \lambda D + R \tag{7}$$

Method	MCL-JCV	UVG	HEVC Class B	HEVC Class C	HEVC Class D	HEVC Class E
DCVC (proposed)	**-23.9%**	**-25.3%**	**-26.0%**	**-5.8%**	**-17.5%**	**-11.9%**
DVCPro [4]	-4.1%	-7.9%	-9.0%	7.2%	-6.9%	17.2%
x265 (veryslow)	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
DVC [3]	13.3%	17.2%	7.9%	15.1%	7.2%	21.1%
x264 (veryslow)	32.7%	30.3%	35.0%	19.9%	15.5%	50.0%

Table 1: The BD-Bitrate comparison. The anchor is x265 (veryslow). Negative number means bitrate saving and positive number means bitrate increase.

λ controls the trade-off between the distortion D and the bitrate cost R. D can be MSE (mean squared error) or MS-SSIM (multiscale structural similarity) for different targets. During the training, R is calculated as the cross-entropy between the true and estimated probability of the latent codes.

Although we already use the pre-trained optical flow estimation network [28] as the initialization of motion estimation like [4], the training may be still unstable if we directly use L_{all} to begin the training. Sometimes, the MV bitrate cost is very small but the total rate-distortion loss is large. This is because that the model thinks directly learning to generate context without MEMC is easier. Inspired by the existing work [7] where the progressive training strategy is used, we customize a progressive training strategy for DCVC, where more details are presented in supplementary materials.

The proposed progressive training strategy makes the model training more stable. The learning rate is set as 1e-4 from the start and 1e-5 is used in the fine-tuning stages. The training batch size is set as 4. For comparing DCVC with other methods, we training 4 models with different λ {MSE: 256, 512, 1024, 2048; MS-SSIM: 8, 16, 32, 64}.

4 Experimental Results

4.1 Experimental Settings

Training Data We use the training part in Vimeo-90k septuplet dataset [29] as our training data. Each video has 7 frames and the spatial resolution is 448x256. During the training, we will randomly clip videos into 256x256 patch.

Testing Data The testing data includes HEVC Class B (1080P), C (480P), D (240P), E (720P) from the common test conditions [30] used by codec standard community. In addition, The 1080p videos from MCL-JCV[31] and UVG[32] datasets are also tested.

Testing Settings The GOP (group of pictures) size is same with [4], namely 10 for HEVC videos and 12 for non-HEVC videos. As this paper only focuses inter frame coding, for intra coding, we directly use existing deep image compression models, where the model parameters are provided by CompressAI [33]. For MSE target, we use *cheng2020-anchor* [19]. For MS-SSIM target, we use *hyperprior* [13] because CompressAI has not the *cheng2020-anchor* models for MS-SSIM target.

According to the performance comparison in [34, 35], DVCPro[4] is the previous SOTA DL-based codec among recent works [9, 6, 10, 5, 25]. Thus, we use DVCPro as our main baseline. In addition, its predecessor DVC[3] is also tested. It is noted that, for fair comparison, DVC and DVCPro are retested using the same intra-coding with DCVC, and their results are better than that reported in [4]. For traditional hybrid video codecs, x264 and x265 encoders [36] are tested. The settings of these two encoders are same with [4] except that we use the veryslow preset rather than veryfast preset. Veryslow preset can achieve higher compression ratio than veryfast preset.

4.2 Performance Comparison

Fig .5 and Fig .6 show the rate and distortion comparison between these methods, where the distortion in Fig .5 is measured by PSNR and the distortion in Fig .5 is measured by MS-SSIM. From these figure, we can find that our DCVC model can outperform DVCPro for all bitrate ranges. Table 1 gives the corresponding BD-Bitrate [37] results in terms of PSNR. When compared with x265 using veryslow preset, DVCPro achieves 4.1%, 7.9%, 9.0%, and 6.9% bitrate saving on MCL-JCV, UVG, HEVC Class B, and D, respectively. However, for HEVC Class C and E, DVCPro still performs worse, and there are 7.2% and 17.2% bitrate increases. By contrast, our DCVC can outperform x265

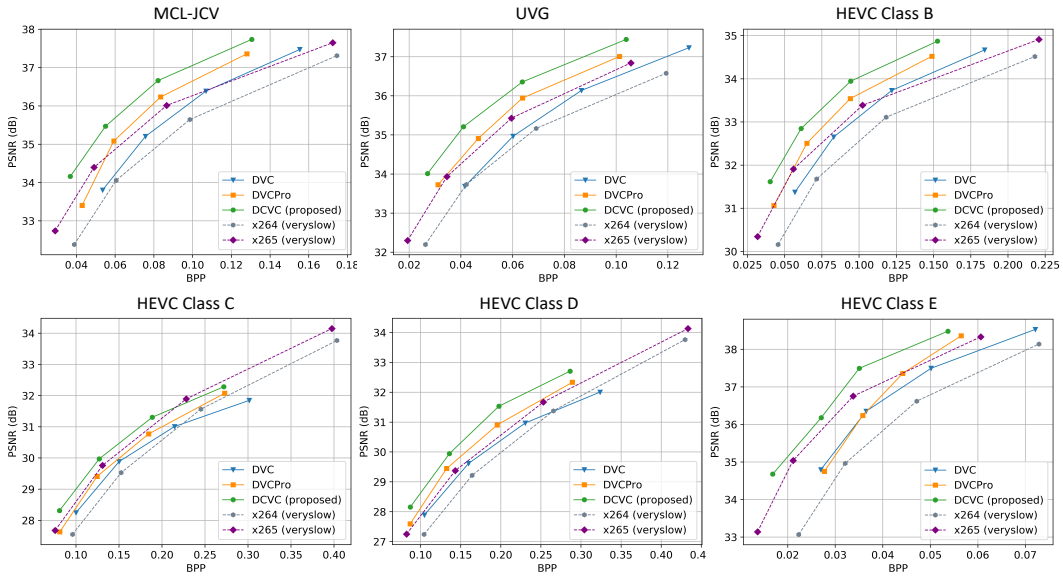

Figure 5: PSNR and bitrate comparison. The horizontal axis means BPP (bits per pixel) representing bitrate cost and vertical axis is PSNR representing reconstruction quality.

on all datasets. For three 1080p datasets (MCL-JCV, UVG, HEVC Class B), the bitrate savings are 23.9%, 25.3%, 26.0%, respectively. For low resolution videos HEVC Class C (480P) and D (240P), the improvements are also adequate, i.e. 5.8% and 17.5%. For HEVC Class E with relatively has small motion, the bitrate saving 11.9%. From these comparisons, we can find that our DCVC can significantly outperform DCVC and x265 for videos with different resolutions and different context characteristics.

In addition, we can find that DCVC can achieve larger improvement on high resolution video. This is because that high resolution video contains more textures with high frequency. For this kind of video, the feature domain context with higher dimension is more helpful and able to carry richer information to encode the high frequency content.

4.3 Ablation Study

Conditional Coding and Context in Feature Domain Our DCVC is a conditional coding-based framework, where the condition is the context in feature domain. To demonstrate the effectiveness of these two ideas, we test the performance of conditional coding manner and the context in feature domain separately. The performance comparison is shown in Fig. 7. We first test our designed conditional coding manner but the condition is still the predicted frame in pixel domain, i.e. DCVC (predicted frame in pixel domain). It is with the brown dashed curve in Fig. 7. When compared with DVCPro, the bitrate saving is 12.7%. When we use the context in feature domain as condition, the performance can be further boosted, especially for the bitrate range from middle to high. The additional bitrate saving is 5.1%. These results show the advantage of our ideas.

Channel Dimension of Context In DCVC, the channel dimension of context is set as 64 in the implementation. We also conduct the experiment when different dimensions are used. The cases (3, 16, 256-Dim.) are also tested. The corresponding bitrate saving comparison is shown in Fig. 8, where the anchor is the 3-Dim. model. From this figure, we can observe that the 16-Dim. model can improve the performance in some degree, and the 64-Dim. model can further boost the performance in a larger degree for most datasets. However, the improvement brought by 256-Dim. model is relatively small. The HEVC Class E even has performance loss. The reason may be that the model training is not stable if there is no extra supervision for training the context. For this reason, we adopt the 64-Dim. model at present.

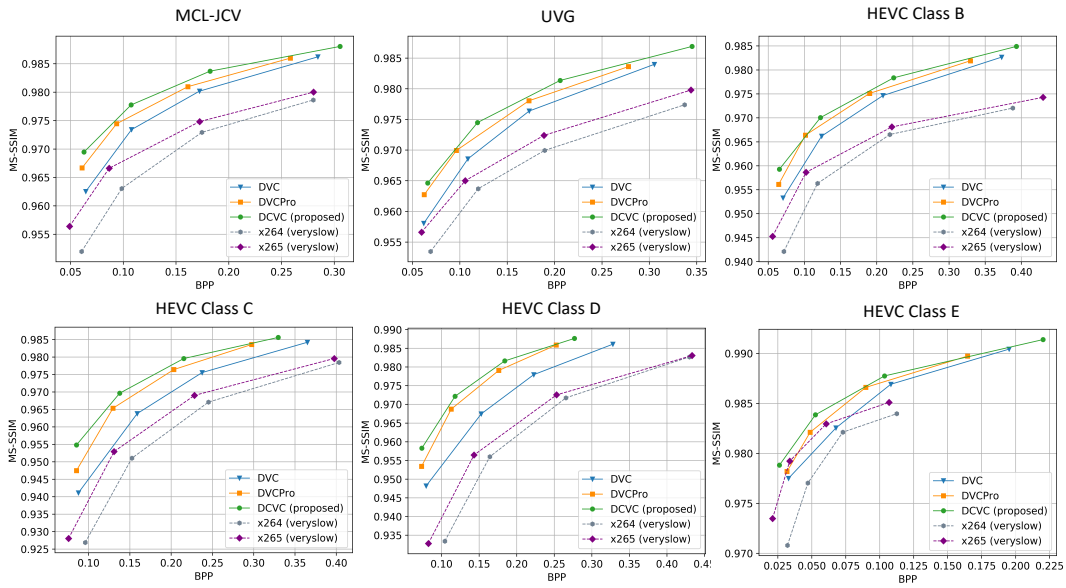

Figure 6: MS-SSIM and bitrate comparison. The DL-based codecs are retrained for MS-SSIM.

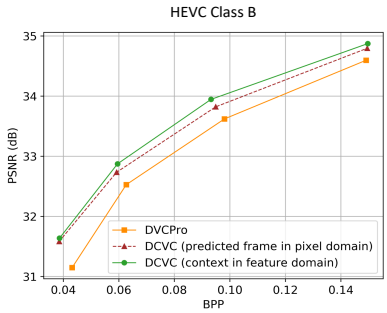

Figure 7: Conditional coding using context or predicted frame.

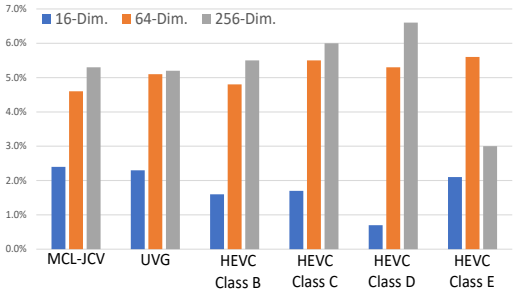

Figure 8: Bitrate saving when using different channel dimensions for context. The anchor is 3-Dim. (dimension is 3) model.

5 Conclusion

In this paper, we make efforts on designing a conditional coding-based deep video compression framework which has a lower entropy bound than the commonly-used residue coding-based framework. We provide an solution called DCVC, which answer these questions: *What is condition? How to use condition? And how to learn condition?* Experimental results demonstrate the effectiveness of our DCVC solution. When compared with previous SOTA model and x265 using veryslow preset, DCVC can achieve significant bitrate saving. More experiments and analyses can be seen in the supplementary materials. DCVC is the beginning of conditional coding-based framework. There exists huge extensible potential in designing a more efficient solution by better defining, using and learning the condition.

References

[1] B. Girod, E. G. Steinbach, and N. Faerber, "Comparison of the h. 263 and h. 261 video compression standards," in *Standards and Common Interfaces for Video Information Systems: A Critical Review*, 1995.

[2] B. Bross, J. Chen, J.-R. Ohm, G. J. Sullivan, and Y.-K. Wang, "Developments in international video coding standardization after avc, with an overview of versatile video coding (vvc)," *Proceedings of the IEEE*, 2021.

[3] 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/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11006–11015, 2019.

[4] G. Lu, X. Zhang, W. Ouyang, L. Chen, Z. Gao, and D. Xu, "An end-to-end learning framework for video compression," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[5] G. Lu, C. Cai, X. Zhang, L. Chen, W. Ouyang, D. Xu, and Z. Gao, "Content adaptive and error propagation aware deep video compression," in *European Conference on Computer Vision*, pp. 456–472, Springer, 2020.

[6] R. Yang, F. Mentzer, L. V. Gool, and R. Timofte, "Learning for video compression with hierarchical quality and recurrent enhancement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[7] J. Lin, D. Liu, H. Li, and F. Wu, "M-lvc: multiple frames prediction for learned video compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3546–3554, 2020.

[8] R. Yang, F. Mentzer, L. V. Gool, and R. Timofte, "Learning for video compression with hierarchical quality and recurrent enhancement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6628–6637, 2020.

[9] A. Djelouah, J. Campos, S. Schaub-Meyer, and C. Schroers, "Neural inter-frame compression for video coding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[10] E. Agustsson, D. Minnen, N. Johnston, J. Balle, S. J. Hwang, and G. Toderici, "Scale-space flow for end-to-end optimized video compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8503–8512, 2020.

[11] C.-Y. Wu, N. Singhal, and P. Krähenbühl, "Video compression through image interpolation," in *ECCV*, 2018.

[12] T. Ladune, P. Philippe, W. Hamidouche, L. Zhang, and O. Déforges, "Optical flow and mode selection for learning-based video coding," in *22nd IEEE International Workshop on Multimedia Signal Processing, MMSP 2020, Tampere, Finland, September 21-24, 2020*, pp. 1–6, IEEE, 2020.

[13] 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.

[14] D. Minnen, J. Ballé, and G. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," *arXiv preprint arXiv:1809.02736*, 2018.

[15] J. Liu, S. Wang, W.-C. Ma, M. Shah, R. Hu, P. Dhawan, and R. Urtasun, "Conditional entropy coding for efficient video compression," *arXiv preprint arXiv:2008.09180*, 2020.

[16] T. Ladune, P. Philippe, W. Hamidouche, L. Zhang, and O. Déforges, "Conditional coding for flexible learned video compression," in *Neural Compression: From Information Theory to Applications – Workshop @ ICLR*, 2021.

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

[18] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," *arXiv preprint arXiv:1611.01704*, 2017.

[19] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7939–7948, 2020.

[20] G. Toderici, S. M. O'Malley, S. J. Hwang, D. Vincent, D. Minnen, S. Baluja, M. Covell, and R. Sukthankar, "Variable rate image compression with recurrent neural networks," *arXiv preprint arXiv:1511.06085*, 2015.

[21] G. Toderici, D. Vincent, N. Johnston, S. Jin Hwang, D. Minnen, J. Shor, and M. Covell, "Full resolution image compression with recurrent neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5306–5314, 2017.

[22] N. Johnston, D. Vincent, D. Minnen, M. Covell, S. Singh, T. Chinen, S. J. Hwang, J. Shor, and G. Toderici, "Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4385–4393, 2018.

[23] J. Pessoa, H. Aidos, P. Tomás, and M. A. Figueiredo, "End-to-end learning of video compression using spatio-temporal autoencoders," in *2020 IEEE Workshop on Signal Processing Systems (SiPS)*, pp. 1–6, IEEE, 2020.

[24] A. Habibian, T. v. Rozendaal, J. M. Tomczak, and T. S. Cohen, "Video compression with rate-distortion autoencoders," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7033–7042, 2019.

[25] Z. Hu, Z. Chen, D. Xu, G. Lu, W. Ouyang, and S. Gu, "Improving deep video compression by resolution-adaptive flow coding," in *European Conference on Computer Vision*, pp. 193–209, Springer, 2020.

[26] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE mobile computing and communications review*, vol. 5, no. 1, pp. 3–55, 2001.

[27] "Pytorchvideocompression." https://github.com/ZhihaoHu/PyTorchVideoCompression.

[28] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4161–4170, 2017.

[29] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *International Journal of Computer Vision (IJCV)*, vol. 127, no. 8, pp. 1106–1125, 2019.

[30] F. Bossen *et al.*, "Common test conditions and software reference configurations," in *JCTVC-L1100*, vol. 12, 2013.

[31] H. Wang, W. Gan, S. Hu, J. Y. Lin, L. Jin, L. Song, P. Wang, I. Katsavounidis, A. Aaron, and C.-C. J. Kuo, "Mcl-jcv: a jnd-based h. 264/avc video quality assessment dataset," in *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 1509–1513, IEEE, 2016.

[32] "Ultra video group test sequences." http://ultravideo.cs.tut.fi. Online; accessed 12 April 2021.

[33] J. Bégaint, F. Racapé, S. Feltman, and A. Pushparaja, "Compressai: a pytorch library and evaluation platform for end-to-end compression research," *arXiv preprint arXiv:2011.03029*, 2020.

[34] D. Xu, G. Lu, R. Yang, and R. Timofte, "Learned image and video compression with deep neural networks," in *2020 IEEE International Conference on Visual Communications and Image Processing, VCIP 2020, Macau, China, December 1-4, 2020*, pp. 1–3, IEEE, 2020.

[35] "Tutorial: Learned image and video compression with deep neural networks." https://drive.google.com/file/d/162omgk0CmHPBj4J7vWsNr8N9SPn5j97F/view, 2020. Online; accessed 12 April 2021.

[36] "Ffmpeg." https://www.ffmpeg.org/.

[37] G. Bjontegaard, "Calculation of average psnr differences between rd-curves," *VCEG-M33*, 2001.