GoP-to-Frame Encoder Adaptation for Learned Video Compression

Xiaohan Pan*, Runsen Feng*, Henan Wang, Yixin Gao, Zhibo Chen†
University of Science and Technology of China
{pxh123, fengruns, henanwang, gaoyixin}@mail.ustc.edu.cn, chenzhibo@ustc.edu.cn

***Abstract*—Online adaptation is an important optimization method in learned video compression. The approach improves the rate-distortion performance of the pretrained model by adapting the encoder or the encoded representation to better fit specific content. However, existing methods face significant challenges due to their high complexity, limiting the application. To address this issue, we propose a new approach named GoP-to-Frame Encoder Adaptation (GFEA), which consists of two steps: GoP-level Encoder Adaptation (GEA) and Frame-level Encoder Adaptation (FEA). Unlike previous methods that adapt the model frame-by-frame, GEA learns a shared model update through sequence-level adaptation, largely reducing the adaptation complexity. To further improve the rate-distortion-complexity performance, we introduce the more efficient local frame adaptation mechanism in FEA. The experimental results demonstrate the effectiveness of our approach, showing a reduction in encoding time and memory usage by approximately 80% and 65%, while retaining the overall performance improvement brought by the online adaptation method.**

I. Introduction

Video streams now constitute the majority of Internet traffic, with the volume of video data continuing to grow each year. This underscores the critical role of video compression algorithms in reducing transmission and storage costs.

Learned video compression has gained significant attention in recent years. While early studies [1]–[5] explored various architectures, most modern approaches [3], [6]–[17] still follow the hybrid scheme established by traditional coding methods. However, advancements in network architectures and model design have rapidly improved the performance of learned video compression. As a result, it now offers comparable performance to traditional methods, while also providing the flexibility to adapt to different evaluation metrics and downstream tasks.

Among the various optimization techniques for learned image and video compression, online adaptation emerges as a pivotal and promising direction. While learned compression optimizes based on the statistical distribution of the dataset through empirical risk minimization, discrepancies often exist between individual sample content and the overall dataset distribution. Online adaptation addresses this mismatch by dynamically adjusting the model during the compression process, leading to enhanced performance. Extensive research [18]–[26] have explored the potential of this approach, continuously pushing its boundaries for further improvements. These methods can be broadly classified into two categories: latent adaptation, which adapts the latent representation to the content, and encoder adaptation, which updates the encoder based on the content.

Despite the promising performance gains, these methods face significant challenges due to their high complexity, which hinders practical application. This complexity manifests in two aspects: long encoding time and high memory consumption. The extended encoding time results from slow convergence and the lengthy single-frame update process, a particular drawback of latent-adaptation-based methods that require thousands of iterations to fit a single frame. In contrast, encoder adaptation methods offer better time efficiency. A notable example is Online Encoder Updating (OEU) [25], where the encoder is updated based on frame-level rate-distortion loss. However, this approach suffers from excessive memory consumption. The high memory usage stems from the generation of high-resolution feature maps during frame adaptation, as well as the additional overhead needed to store gradients.

To enhance the practicality of online adaptation, we propose a GoP-to-Frame Encoder Adaptation (GFEA) method to reduce complexity in both memory usage and encoding time. Unlike previous methods adapt model frame-by-frame, GFEA first adapts a shared model across the sequence content, allowing for faster convergence and reduces encoding time. To further improve the rate-distortion-complexity performance, GFEA then introduces the more efficient local frame adaptaion mechanism, avoiding the high-complexity of global frame adaptation in OEU [25].

Our contributions are summarized as follows:

- We propose GoP-level Encoder Adaptation (GEA) that effectively reduces online adaptation time.
- We introduce a local frame adaptation mechanism in Frame-level Encoder Adaptation (FEA) that optimizes computational resource utilization and reduces memory usage.
- Experimental results demonstrate the effectiveness of our approach, showing a reduction in encoding time and memory usage by approximately 80% and 65%, respectively, while maintaining the performance improvements achieved by general online adaptation methods such as OEU [25].

*Equal contribution. †Corresponding author.

This work was supported in part by NSFC under Grant 62371434, 62021001.

979-8-3315-7206-8/25/$31.00

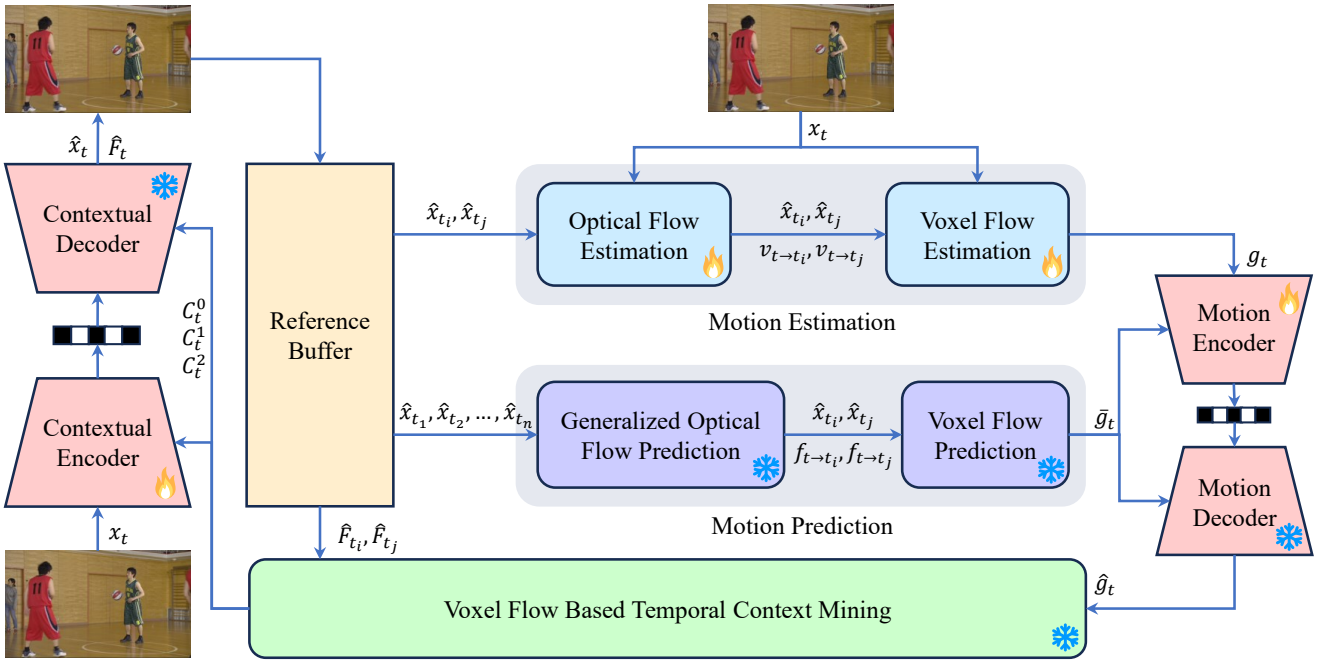


Fig. 1: The framework consists of five key components: Motion Prediction, Motion Estimation, Motion Compression, Temporal Context Mining, and Contextual Compression. During online encoder adaptation, the modules marked with snowflake icons remain frozen, while only those with flame icons undergo updates.

II. Related Work

A. Learned Video Compression

The schemes of learned video compression can be roughly categorized into the following two classes:

1) Predictive Coding based Scheme: Models following this scheme are largely inspired by traditional video compression techniques. Temporal redundancy is handled through motion prediction/compensation and residual coding. DVC [3] is a pioneering framework from the early stages of this approach. Subsequent works have advanced performance by focusing on hierarchical processing [9], [12], improved motion estimation and compensation [6], [7], [11], [13], motion compression [8], and enhanced context mining [5], [10], [14]–[17].

2) Implicit Neural Representation based Scheme: These models represent a video through the weights of a neural network. The primary approach involves overfitting the model to the video and subsequently compressing it, a process that is computationally expensive and time-consuming. However, decoding is much faster than earlier methods, as it involves a single inference step. Chen et al. [27] were the first to propose frame-wise implicit neural representation (INR) instead of pixel-wise representations. Subsequent works have enhanced this framework by introducing spatial-temporal decoupling [28], entropy constraints [29], patch-wise representations [30], and techniques to address temporal redundancy [31], [32]. Further advancements include improvements in coordinate embedding [33], [34].

B. Online Adaptation

Similar to implicit neural representation-based video compression, online adaptation also fine-tunes the video representations or compression model weights specifically to the video. There are two types of adaptation:

1) Latent Adaptation: Several works achieve online adaptation through latent refinement. Campos et al. [18] developed an optimization pipeline within learned image compression. Building on this, subsequent studies [19], [22], [23] have explored optimal quantization techniques and the upper limits of this adaptation method. Xu et al. [24] extended this approach to learned video compression. While the performance gains are substantial, this method is hindered by long update iterations, leading to excessively prolonged encoding times.

2) Encoder Adaptation: Other approaches focus on adapting the encoder network to the content. Lu et al. [25] were the first to introduce an online encoder updating strategy in learned video compression. Subsequent work [26] introduced a parameter-efficient updating strategy, but it still relies on full-frame content as input and suffers from high memory consumption. We explore a more efficient approach to online encoder adaptation, which will be discussed in detail later.

III. Method

A. Learned Video Compression Framework

Our framework builds upon VLVC [11], with the overall pipeline illustrated in Fig. 1. In addition to the reference buffer, the framework is composed of five key components:

Motion Prediction, Motion Estimation, Motion Compression, Temporal Context Mining, and Contextual Compression. Each component is introduced in detail below:

1) Motion Prediction: The motion prediction is a two-step process. First, generalized optical flow $f_{t\to t_k}$ is predicted. To fully leverage information from multiple frames $\hat{x}_{t_1}, \hat{x}_{t_2}, \ldots, \hat{x}_{t_n}$, we model forward flows using polynomial functions. Optical flows between cached frames are estimated using a pre-trained flow network, and polynomial coefficients are obtained through a closed-form solution. A reversal layer then generates the backward flow via softmax splatting.

Next, a dedicated network generates the voxel flow feature based on the predicted optical flows and reference frames. Using the generalized flow expression, required flows (e.g., $f_{t\to t_i}, f_{t\to t_j}$) are derived, and combined with the reference frames $\hat{x}_{t_i}$, $\hat{x}_{t_j}$ to produce the voxel flow feature $\bar{g}_t$, which serves as the condition for the motion coder. The structure of the voxel flow will be introduced later.

2) Motion Estimation: The motion estimation module estimates the voxel flow g_t for encoding. Its structure mirrors the motion prediction module. First, optical flows between the current frame x_t and referred frames (e.g., $\hat{x}_{t_i}, \hat{x}_{t_j}$) are estimated. Then, the flows $v_{t\to t_i}, v_{t\to t_j}$ and reference frames $\hat{x}_{t_i}, \hat{x}_{t_j}$ are processed to generate the voxel flow g_t, a 4-channel field comprising the 3-channel spatial-temporal flow $g^i = (g^i_x, g^i_y, g^i_z)$ and a weight channel g^i_w.

3) Motion Compression: Motion compression is performed using an auto-encoder-based network, which takes the voxel flow g_t as input. The predicted voxel flow feature $\bar{g}_t$ is injected into both the encoder and decoder. The compression network also incorporates a hyperprior and context model.

4) Temporal Context Mining: The temporal context mining module extracts contextual information to compress the current frame. It takes the decoded voxel flow $\hat{g}_t$ and referred frame features $\hat{F}_{t_i}, \hat{F}_{t_j}$ from the reference buffer as inputs, outputting multi-scale contextual features C^0_t, C^1_t, C^2_t.

5) Contextual Compression: Contextual compression, like motion compression, is performed using an auto-encoder-based network. During encoding and decoding, multi-scale contextual features C^0_t, C^1_t, C^2_t are injected at different layers to aid in compressing the current frame x_t. The decoded frame $\hat{x}_t$ and feature $\hat{F}_t$ are then stored in the reference buffer.

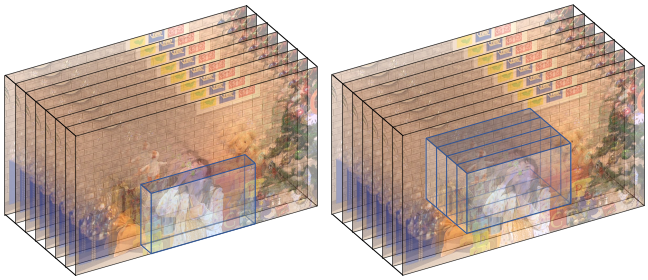
B. GoP-to-Frame Encoder Adaptation

Building on the low-delay mode of our framework, we introduce the GoP-to-Frame Encoder Adaptation method.

1) Frame-Level Encoder Adaptation (FEA): First we describe the adaptation method under the assumption of single frame optimization with global content, which is exactly the OEU strategy [25]. With the frame x_t and the decoded $\hat{x}_t$, rate-distortion loss could be calculated as follows:

$$L_t = R_t + \lambda \cdot D_t = R_t + \lambda \cdot D(x_t, \hat{x}_t) \tag{1}$$

Utilizing the loss L_t, modules of motion estimation are updated through gradient back propagation as well as motion encoder and contextual encoder, while the other modules in the decoding loop are frozen as depicted in Fig. 1. To achieve convergence, the model iterates n times for every frame. The iteration times vary across different models.



(a) Local Frame-level Adaptation (b) Local GoP-level Adaptation

Fig. 2: Illustration of our proposed efficient adaptation methods. (a) The local frame-level adaptation significantly reduces memory usage. (b) The local GoP-level adaptation adapts the model to the sequence, reduces the overall encoding time.

Actually the global frame adaptation described above has some disadvantages. First, a series of high-resolution features need to be maintained during the back propagation process, which leads excessive memory usage; Second, the per frame adaptation is time consuming. Both drawbacks make this adaptation method inefficient.

Experimentally, we found that too much spatial content has little benefit in further improving adaptation performance. So we propose the local frame adaptation method as a natural solution to the memory problem, as shown in Fig. 2(a). The loss can be modified as follows:

$$L_t = R_t + \lambda \cdot D(x_t \odot m_{local}, \hat{x}_t \odot m_{local}) \tag{2}$$

Where $\odot$ represents Hadamard product and m_{local} represents the local content mask picked every iteration.

2) GoP-Level Encoder Adaptation (GEA): In addition, we further explored the adaptation of sequence content. Inspired by the thoughts inside NeRV [27], we also attempt to learn model parameters shared across the sequence. Since the entire sequence is adapted at the same time, this method is much faster than frame-by-frame optimization.

Concretely, during the adaptation, we pick local GoP within a GoP from the sequence, and use the local GoP level loss to update the weights of a shared model. Here, "local" refers to both temporal and spatial dimensions. Assuming the local GoP contains T frames with time indices $\{t_1, t_2, ..., t_T\}$, the loss can be written as follows:

$$L_T = \sum_{t=1}^{T} R_t + \sum_{t=1}^{T} \lambda_t \cdot D(x_t \odot m_{local}, \hat{x}_t \odot m_{local}) \tag{3}$$

Here each frame in a local GoP has its own λ_{t_i}. But unlike local frame adaptation, all frames are encoded with a shared model. Therefore, the shared model are adapted to the whole sequence rather than a single frame.

The above two improvements for adaptation can be used together as the GoP-to-Frame encoder adaptation method, where GEA is performed prior to FEA.

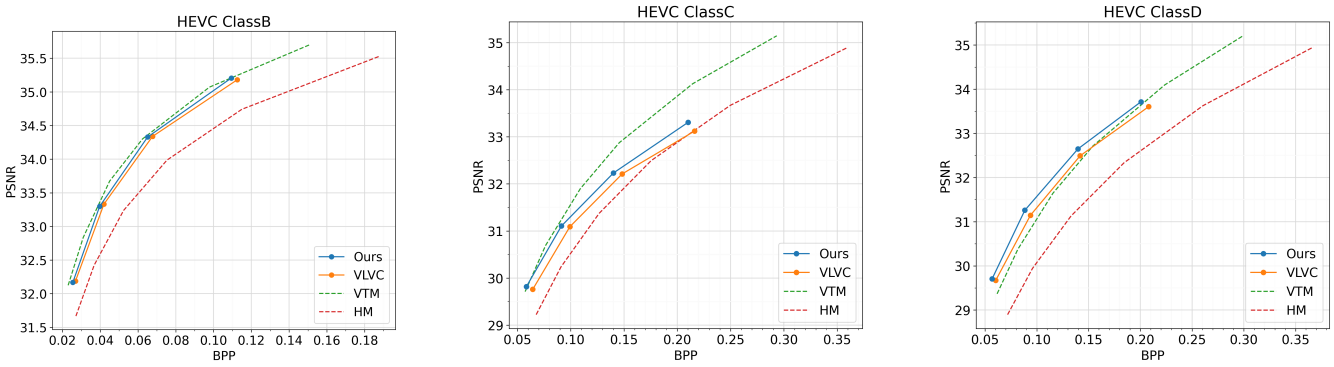


Fig. 3: RD curves on HEVC dataset.

IV. Experiments

A. Experimental Setup

1) Baseline and Dataset: Our baseline codec is VLVC [11] in the lowdelay mode. We also include the reference softwares HM [35] and VTM [36] for evaluation. For baseline training we use 7-frame videos of Vimeo-90k [37]. During training the video clips are randomly cropped into 256×256 patches. For evaluation, we use the Class B, C, D of HEVC dataset [38], which contains videos with different resolution. The first 96 frames of each video is used for evaluation.

2) Implementation Details: In our proposed adaptation method, the input frames are cropped into local patches. The crop size is set to 512×512 for HEVC B dataset, and 256×256 for HEVC C/D dataset. In frame adaptation, the model is optimized for 64 steps per frame. In GoP adaptation, the model is optimized for 128 steps per GoP and the GoP size is set to 32. We use the Adam [39] optimizer, with the learning rate initially set to 5×10^{-5} and reduced to 1×10^{-5} during the last 20% of the optimization steps.

B. Results And Analysis

1) RD Results: We show the RD curves in Fig. 3. VLVC [11] is our baseline codec. With the proposed online encoder adaptation method, the RD performance is significantly improved. In Table I, we also provide the BD-rate comparison. OEU [25] is reimplemented in the VLVC codec, optimized for 64 steps per frame. Compared to OEU, our method (denoted as Ours) achieves comparable RD performance with a reduction in encoding time and memory usage by approximately 80% and 65%.

2) Effect of Local Frame Adaptation and Local GoP Adaptation: To verify the effectiveness of frame-level encoder adaptation (denoted at Ours-FEA) and GoP-level encoder adaptation (denoted at Ours-GEA), we also evaluate the performance of the two methods. In Table I, we show that Ours-GEA and Ours-FEA have different rate-distortion-complexity trade-off. Ours-FEA outperforms Ours-GEA in terms of RD (rate-distortion) performance. This is because the optimized model weight in Ours-GEA is shared across all the frames in the GoP, restricting the adaptation flexibility on each frame. However, the encoding time of Ours-FEA is about 6× that of Ours-GEA. Compared with OEU, both the proposed method have worse performance but lower complexity.

3) Effect of Crop Size in Frame Adaptation: In Table II, we show the BD-rate and complexity performance when changing crop size in frame adaptation. A smaller crop size results in worse BD-rate performance but reduces the overall complexity. As results, we choose 512×512 for HEVC Class B dataset for better rate-distortion-complexity performance.

TABLE I: BD-rate(%) comparison in terms of PSNR. Time is the average encoding time per frame and memory is the peak GPU memory. Time and memory are evaluated on 1080p videos.

	Class B	Class C	Class D	Time (s)	Memory (GB)
VLVC	0.0	0.0	0.0	0.5	5.6
+ OEU	**-4.32**	**-8.81**	-7.56	∼42	∼30
+ Ours-FEA	-3.72	-7.94	-7.31	8.2	**7.3**
+ Ours-GEA	-0.75	-3.14	-2.78	**1.4**	10.3
+ Ours	-3.92	-8.20	**-7.66**	9.1	10.3

TABLE II: The effect of crop size in frame adaptation on HEVC Class B dataset. The anchor of BD-rate is VLVC.

Crop size	BD-rate	Time (s)	Memory (GB)
256x256	2.93	5.9	6.1
512x512	-3.72	8.2	7.3
1024x1024	-4.18	22.6	15.4
Full Frame	-4.32	∼42	∼30

V. Conclusion

In this paper, we propose a new online encoder adaptation method to address the high complexity issue in previous method. To address the issue, we first propose a GoP-level adaptation method which adapts the model to a GoP instead of a single frame. The updated model weight is shared across all the frames in a GoP. Secondly, we propose to adapt model to the local frame content, largely reducing the encoding time and memory of global frame adaptation. Experimental results demonstrate the effectiveness of our method.

REFERENCES

[1] C.-Y. Wu, N. Singhal, and P. Krahenbuhl, "Video compression through image interpolation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 416–431.

[2] O. Rippel, S. Nair, C. Lew, S. Branson, A. G. Anderson, and L. Bourdev, "Learned video compression," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3454–3463.

[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*, 2019, pp. 11 006–11 015.

[4] 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*, 2019, pp. 7033–7042.

[5] O. Rippel, A. G. Anderson, K. Tatwawadi, S. Nair, C. Lytle, and L. Bourdev, "Elf-vc: Efficient learned flexible-rate video coding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 479–14 488.

[6] 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*, 2020, pp. 8503–8512.

[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*, 2020, pp. 3546–3554.

[8] Z. Hu, Z. Chen, D. Xu, G. Lu, W. Ouyang, and S. Gu, "Improving deep video compression by resolution-adaptive flow coding," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 193–209.

[9] Z. Hu, G. Lu, and D. Xu, "FVC: A new framework towards deep video compression in feature space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1502–1511.

[10] J. Li, B. Li, and Y. Lu, "Deep contextual video compression," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 114–18 125, 2021.

[11] R. Feng, Z. Guo, Z. Zhang, and Z. Chen, "Versatile learned video compression," *arXiv preprint arXiv:2111.03386*, 2021.

[12] Z. Hu, G. Lu, J. Guo, S. Liu, W. Jiang, and D. Xu, "Coarse-to-fine deep video coding with hyperprior-guided mode prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5921–5930.

[13] Z. Guo, R. Feng, Z. Zhang, X. Jin, and Z. Chen, "Learning cross-scale weighted prediction for efficient neural video compression," *IEEE Transactions on Image Processing*, vol. 32, pp. 3567–3579, 2023.

[14] X. Sheng, J. Li, B. Li, L. Li, D. Liu, and Y. Lu, "Temporal context mining for learned video compression," *IEEE Transactions on Multimedia*, vol. 25, pp. 7311–7322, 2022.

[15] J. Li, B. Li, and Y. Lu, "Hybrid spatial-temporal entropy modelling for neural video compression," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 1503–1511.

[16] ——, "Neural video compression with diverse contexts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 22 616–22 626.

[17] ——, "Neural video compression with feature modulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 26 099–26 108.

[18] 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 Workshops*, 2019, pp. 0–0.

[19] Y. Yang, R. Bamler, and S. Mandt, "Improving inference for neural image compression," *Advances in Neural Information Processing Systems*, vol. 33, pp. 573–584, 2020.

[20] T. van Rozendaal, I. Huijben, and T. S. Cohen, "Overfitting for fun and profit: Instance-adaptive data compression," in *9th International Conference on Learning Representations, ICLR 2021*. ICLR, 2021.

[21] G. Pan, G. Lu, Z. Hu, and D. Xu, "Content adaptive latents and decoder for neural image compression," in *European Conference on Computer Vision*. Springer, 2022, pp. 556–573.

[22] C. Gao, T. Xu, D. He, Y. Wang, and H. Qin, "Flexible neural image compression via code editing," *Advances in Neural Information Processing Systems*, vol. 35, pp. 12 184–12 196, 2022.

[23] Y. Lv, J. Xiang, J. Zhang, W. Yang, X. Han, and W. Yang, "Dynamic low-rank instance adaptation for universal neural image compression," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 632–642.

[24] T. Xu, H. Gao, C. Gao, Y. Wang, D. He, J. Pi, J. Luo, Z. Zhu, M. Ye, H. Qin *et al.*, "Bit allocation using optimization," in *International Conference on Machine Learning*. PMLR, 2023, pp. 38 377–38 399.

[25] 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 *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 456–472.

[26] H. Yang, S. Oh, and E. Park, "Parameter-efficient instance-adaptive neural video compression," *arXiv preprint arXiv:2405.08530*, 2024.

[27] H. Chen, B. He, H. Wang, Y. Ren, S. N. Lim, and A. Shrivastava, "Nerv: Neural representations for videos," *Advances in Neural Information Processing Systems*, vol. 34, pp. 21 557–21 568, 2021.

[28] Z. Li, M. Wang, H. Pi, K. Xu, J. Mei, and Y. Liu, "E-nerv: Expedite neural video representation with disentangled spatial-temporal context," in *European Conference on Computer Vision*. Springer, 2022, pp. 267–284.

[29] C. Gomes, R. Azevedo, and C. Schroers, "Video compression with entropy-constrained neural representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18 497–18 506.

[30] S. R. Maiya, S. Girish, M. Ehrlich, H. Wang, K. S. Lee, P. Poirson, P. Wu, C. Wang, and A. Shrivastava, "Nirvana: Neural implicit representations of videos with adaptive networks and autoregressive patch-wise modeling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14 378–14 387.

[31] B. He, X. Yang, H. Wang, Z. Wu, H. Chen, S. Huang, Y. Ren, S.-N. Lim, and A. Shrivastava, "Towards scalable neural representation for diverse videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 6132–6142.

[32] J. C. Lee, D. Rho, J. H. Ko, and E. Park, "Ffnerv: Flow-guided frame-wise neural representations for videos," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 7859–7870.

[33] H. Chen, M. Gwilliam, S.-N. Lim, and A. Shrivastava, "Hnerv: A hybrid neural representation for videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 10 270–10 279.

[34] Q. Zhao, M. S. Asif, and Z. Ma, "Dnerv: Modeling inherent dynamics via difference neural representation for videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2031–2040.

[35] HM, "Hevc offical test model," https://hevc.hhi.fraunhofer.de, 2021.

[36] VTM, "Vvc offical test model," https://jvet.hhi.fraunhofer.de, 2021.

[37] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *International Journal of Computer Vision*, vol. 127, pp. 1106–1125, 2019.

[38] F. Bossen *et al.*, "Common test conditions and software reference configurations," *JCTVC-L1100*, vol. 12, no. 7, p. 1, 2013.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.