

Embedding Compression Distortion in Video Coding for Machines

Yuxiao Sun¹, Yao Zhao¹, Meiqin Liu^{*1}, Chao Yao², and Weisi Lin³

¹Beijing Jiaotong University, China

²University of Science and Technology Beijing, China

³Nanyang Technological University, Singapore

{yuxiaosun, yzhao, mqliu}@bjtu.edu.cn, yaochao@ustb.edu.cn, wslin@ntu.edu.sg

Abstract—Currently, video transmission serves not only the Human Visual System (HVS) for viewing but also machine perception for analysis. However, existing codecs are primarily optimized for pixel-domain and HVS-perception metrics rather than the needs of machine vision tasks. To address this issue, we propose a Compression Distortion Representation Embedding (CDRE) framework, which extracts machine-perception-related distortion representation and embeds it into downstream models, addressing the information lost during compression and improving task performance. Specifically, to better analyze the machine-perception-related distortion, we design a compression-sensitive extractor that identifies compression degradation in the feature domain. For efficient transmission, a lightweight distortion codec is introduced to compress the distortion information into a compact representation. Subsequently, the representation is progressively embedded into the downstream model, enabling it to be better informed about compression degradation and enhancing performance. Experiments across various codecs and downstream tasks demonstrate that our framework can effectively boost the rate-task performance of existing codecs with minimal overhead in terms of bitrate, execution time, and number of parameters. Our codes and supplementary materials are released in <https://github.com/Ws-Syx/CDRE/>.

Index Terms—Distortion Representation Embedding, Video Compression, Video Coding for Machines

I. INTRODUCTION

Digital video plays a significant role in our daily lives, accounting for a large proportion of data traffic. Increasingly, video transmission serves not only human visual system (HVS) but also downstream machine vision tasks, such as surveillance video analysis and facial recognition. However, most existing codecs focus on pixel-domain and HVS-related metrics [1]–[5] and introduce distortions that harm the accuracy of these downstream tasks, thereby damaging overall rate-task performance. This issue is regarded as a challenge within the field of Video Coding for Machines (VCM).

To address the performance degradation in machine vision tasks caused by compression distortion, a straightforward approach is restoring frames for downstream machine perception with a restoration model [6]–[8]. Another simple yet effective solution is to optimize the machine vision model on compressed datasets, which makes the model better handle

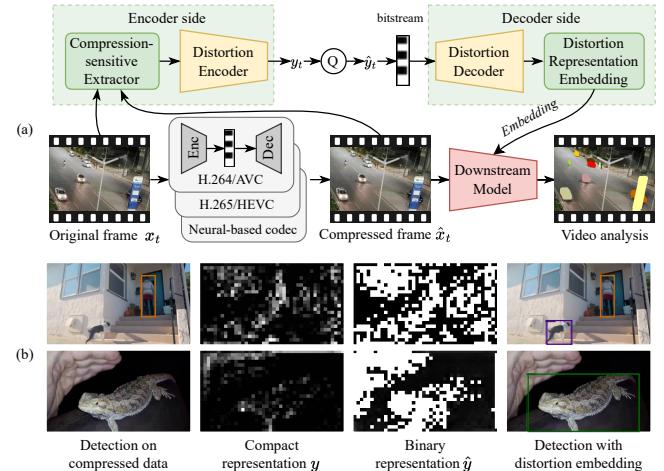


Fig. 1. (a) Overview of our proposed CDRE framework, which extracts, compresses, transmits, and embeds machine-perception-related distortion representations. It helps the downstream model be aware of specific distortions in the input data, improving task accuracy. (b) Visualization of compact and binary representation of machine-perception-related distortion. Detection accuracy increases after distortion representation embedding.

distorted inputs. However, these methods only generalize common distortion patterns and struggle to precisely capture the specific distortion encountered by each input frame. Moreover, introducing additional bitstreams has proven effective, such as maintaining semantic consistency between compressed and original images [9], [10] and enhancing edge details [11]. However, a significant burden is introduced due to the increase in parameters and execution time.

Inspired by prompt learning [12], we reinterpret the prompt as the compression distortion information of the current input, represented through an auxiliary bitstream. The lightweight Compression Distortion Representation Embedding (CDRE) framework is proposed to further improve the rate-task performance of existing codecs without introducing significant computational and bitrate overhead. This framework detects and compresses compression-induced distortion and fuses it into the machine perception process. Due to the down-sampling and extraction in the backbone of machine perception models, pixel-domain texture and HVS-related details are often diminished, making pixel-domain and HVS-related quality less relevant to downstream performance. Instead of focusing on improving the quality of compressed frames, our approach embeds distortion information directly into the downstream

* Corresponding author

This work is supported by the National Natural Science Foundation of China (62120106009, 62372036, U24B20179, U22A2022, 62332017).

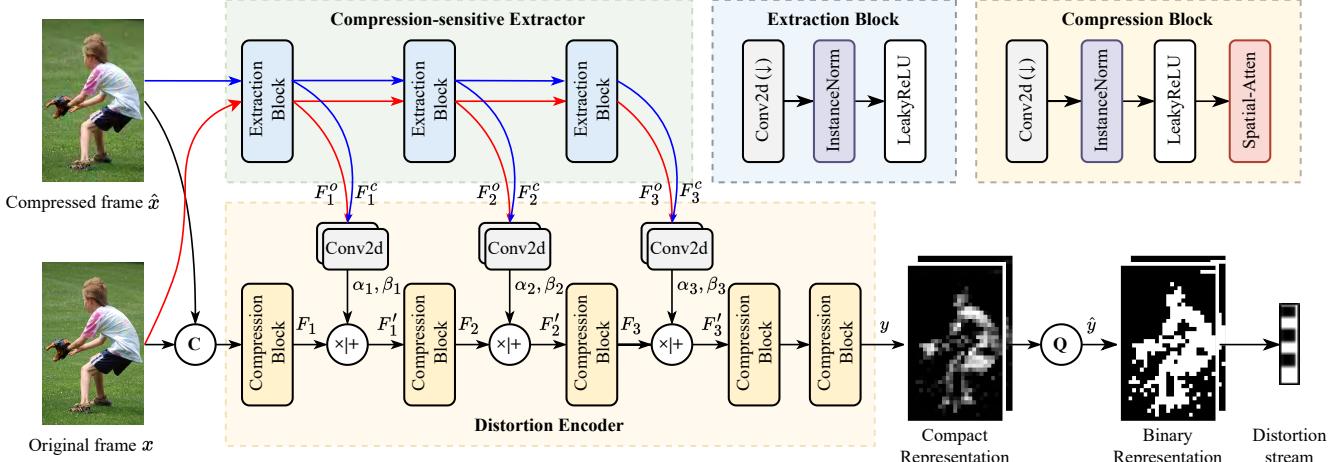


Fig. 2. The process of distortion representation extraction and compression at the encoder side. C denotes channel-wise concatenation, Q denotes binary quantization, and $\times|+$ denotes linear modulation operations. Both the compressed image and the original image are input into the compression-sensitive extractor, yielding multi-level features $F_{1,2,3}^o$ and $F_{1,2,3}^c$ to identify distortions in the feature domain. Subsequently, under the modulation of the aforementioned features, the distortion encoder extracts and compresses the distortions into a compact representation, which is then quantized.

inference process. By effectively incorporating distortion information in the feature domain, downstream models achieve better performance, as illustrated in Fig. 1.

In particular, a compression-sensitive extractor is introduced to identify compression degradation and mining machine-perception-related distortions in the feature domain. To effectively transmit the distortion information to the decoder side without putting extra burden on the encoder side, we employ a lightweight distortion codec to compress and quantize the distortion into a binary representation. After transmission, the representation is gradually decoded and embedded at multiple scales into the backbone of the downstream model by a Distortion Representation Embedding Module, enabling it to perform inference with prior knowledge of the current compression distortion. The effectiveness of the CDRE framework is validated on various existing codecs, including both standard and neural-based codecs. Experimental results indicate that our CDRE effectively enhances rate-task performance in object detection, semantic segmentation, and video instance segmentation. Additionally, only minimal overhead is introduced by CDRE in terms of bitrate, execution time, and number of parameters, making it suitable for scenarios with limited encoder-side resources.

Our contributions are summarized as follows:

- We propose a lightweight Compression Distortion Representation Embedding (CDRE) framework, which enhances rate-task performance in VCM scenarios by extracting and embedding machine-perception-related distortion representation in the feature domain.
- We introduce a compression-sensitive extractor for analyzing the feature-domain distortion and a lightweight distortion codec to compress the distortion representation with minimal computation and bitrate overhead.
- We design a multi-scale distortion representation embedding module to progressively integrate distortion information into downstream models, effectively utilizing this information as a prior during the inference process.

II. METHOD

A. Framework Overview

The pipeline of the CDRE framework is shown in Fig. 1 (b). It contains three main processes: video compression by existing codecs, distortion representation extraction and compression, and distortion representation embedding. Firstly, videos are compressed, transmitted, and reconstructed through existing codecs. Once video transmission is completed, the compression distortion is extracted by a compression-sensitive extractor and compressed into a compact representation y using a distortion encoder. Then it is quantized into binary representation \hat{y} and transmitted to the decoder side with a minimal bitstream and computation overhead. On the decoder side, the representation \hat{y} is reconstructed and embedded into the backbone of downstream tasks in a multi-scale manner, enabling downstream models to better counteract compression distortions.

B. Distortion Representation Extraction

Existing standard and neural-based codecs typically reduce pixel-domain and HVS-perception-related distortion using metrics such as PSNR, MS-SSIM, and LPIPS [1]–[5], [13]. However, the pixel-domain metric PSNR and the HVS-related metrics MS-SSIM and LPIPS do not exhibit a strong correlation with machine perception, leading to suboptimal rate-task performance, as detailed in the supplementary materials. This issue stems from that the down-sampling and feature extraction of downstream backbones discard pixel-domain and HVS-related details, reducing the impact of pixel-domain and HVS-related quality on task accuracy. While distortions related to the HVS, such as motion blur and blocking artifacts, are well-defined, machine-perception distortions lack a clear definition. To overcome this limitation, we propose investigating machine-perception-related distortions by shifting the focus from pixel-domain metrics to feature-domain analysis. Specifically, we defined feature-domain distortion as:

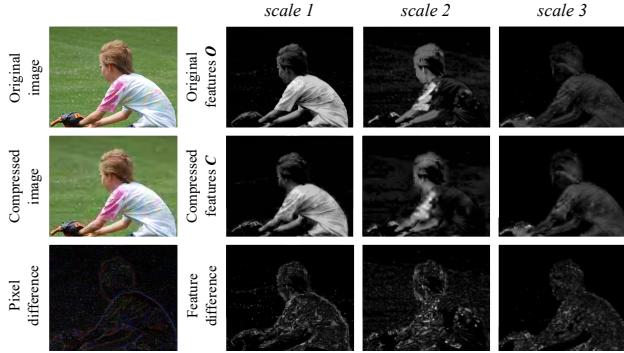


Fig. 3. Visual example from MS-COCO-2017 dataset. White means a higher value in the feature map. Since the feature distance between original and compressed images is amplified at three distinct scales, distortion is more easily discernible in the feature domain than in the pixel domain.

$$D = -\cosine(F^o, F^c) \quad (1)$$

where D denotes machine-perception-related distortion in the feature domain, and $\cosine(\cdot)$ denotes cosine similarity. F^o and F^c are original features and compression-distorted features mentioned later, respectively.

To effectively perceive distortions in the feature domain, we introduce a lightweight compression-sensitive extractor. As shown in Fig. 2, the extractor contains three simple extraction blocks, avoiding computational burdens on the encoder side. As dedicated in Fig. 3, it amplifies the compression degradation in the feature domain between the original and compressed frames, thereby enhancing the focus on areas where feature distortion occurs during the subsequent extraction and compression process. Specifically, the original image x and compressed image \hat{x} are fed into the extractor separately to obtain features F^o and F^c , enabling a hierarchical perception of distortion, as detailed by the following equations.

$$\begin{aligned} F_i^o &= \text{LeakyReLU}(\text{Norm}(\text{Conv}(F_{i-1}^o))) \\ F_i^c &= \text{LeakyReLU}(\text{Norm}(\text{Conv}(F_{i-1}^c))) \end{aligned} \quad (2)$$

where i is the layer-index, $\text{Conv}(\cdot)$ denotes a convolutional layer, $\text{Norm}(\cdot)$ denotes an Instance Normalization layer, and $\text{LeakyReLU}(\cdot)$ denotes a LeakyReLU activation layer.

C. Distortion Representation Compression

To reduce the bitrate of transmitting distortion representation to downstream processes, we employ a VAE-like network [14] for distortion representation compression. It takes the original image x and compressed image \hat{x} as input and considers features F^o and F^c as compression condition. The compression process is dedicated to the following equation.

$$F_d = \mathcal{D}(\mathcal{Q}(\mathcal{E}(x, \hat{x}|F^o, F^c))) \quad (3)$$

where F_d is reconstructed distortion feature, $\mathcal{Q}(\cdot)$ denotes binary quantization mentioned later, $\mathcal{E}(\cdot)$ and $\mathcal{D}(\cdot)$ denote distortion encoder and decoder, respectively.

The distortion encoder down-samples the distortion information and squeezes it to a compact representation y with

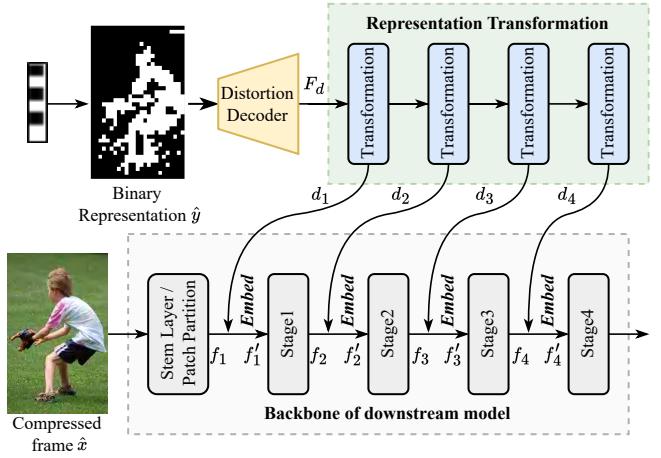


Fig. 4. The process of distortion representation reconstruction and embedding at the decoder side. The representation is reconstructed by the distortion decoder. Then it is transferred and embedded into the backbone of the downstream model for better performance. The structure of the distortion decoder and distortion transformation module are detailed in supplementary materials.

lower channel and spatial dimensions. To better focus on the regions that are severely distorted and beneficial for downstream machine perception, features F^o and F^c are used for linear modulation during distortion representation encoding, as shown in equation 4.

$$\begin{aligned} \alpha_i &= \text{Conv}(F_i^o, F_i^c), \beta_i = \text{Conv}(F_i^o, F_i^c) \\ F'_i &= \alpha_i \times F_i + \beta_i \end{aligned} \quad (4)$$

where α_i and β_i are i -th scaling factor and shifting factor. F_i is the output feature of i -th compression block in distortion encoder and F'_i is distortion-modulated feature.

Similar to the distortion encoder, the decoder contains convolutional layers for up-sampling and restoring the distortion representation. Notably, for CNN-based downstream backbones, the decoder reconstructs the binary representation \hat{y} to the original spatial dimensions. In contrast, for Transformer-based backbones, the decoder reconstructs \hat{y} to match the channel dimension of tokens in the downstream backbone. Details of the decoder are in supplementary materials.

For lower bitrate consumption, the compact representation y is quantized. Existing neural-based codecs typically quantize the floating-point matrix into integers and then use entropy models for entropy coding [1], [3], [5], [15]. However, entropy models introduce heavy computation overhead and require extensive iterations for training. Instead, we introduce a simpler method. The value range of the compact representation is rescaled to between 0 and 1 by the Sigmoid function. Then the existing quantization is simplified to binary quantization, as shown in Equation 5.

$$\hat{y} = \lceil \text{Sigmoid}(y) \rceil \quad (5)$$

where y and \hat{y} are compact representation and binary representation of distortion, respectively. $\lceil \cdot \rceil$ denotes round operation.

D. Distortion Representation Embedding

Inspired by prompt learning [12], we embed distortion features into the backbone of the downstream task model.

This integration allows the model to remain aware of distortion information in the current input, enhancing its performance on downstream tasks. To fully utilize the distortion information, the distortion feature F_d is further transformed into $d_{1,2,3,4}$ and embedded at four distinct scales, as shown in Fig. 4. Notably, the module ensures that the dimensions of the output feature $d_{1,2,3,4}$ align with each stage in the downstream backbone.

The distortion representation embedding operation integrates distortion information with the inference process. Given that CNN and Transformers are the most widely used backbones in machine vision, especially ResNet [16] and Swin-transformer [17], we propose two distinct embedding methods. For the CNN-based backbone, the process of distortion feature transformation and embedding is described by the following equation.

$$\begin{aligned} d_i &= \text{ReLU}(\text{Norm}(\text{Conv}(d_{i-1}))) \\ f'_i &= f_i + \text{CA}(\text{SA}(f_i, d_i)) \end{aligned} \quad (6)$$

where $\text{CA}(\cdot)$ is channel-attention operation, $\text{SA}(\cdot)$ is spatial-attention operation, f_i represents i -th intermediate result in backbone, and f'_i represents embedded feature.

For the Transformer-based backbone, MLP is used for distortion feature transformation and cross-attention is used for distortion representation embedding, as detailed in the following equation.

$$\begin{aligned} d_i &= \text{MLP}(d_{i-1}) \\ Q_i &= f_i W^Q, K_i = d_i W^K, V_i = d_i W^V \\ f'_i &= f_i + \text{Softmax}(Q_i K_i^T / \sqrt{d}) V_i \end{aligned} \quad (7)$$

where $\text{MLP}(\cdot)$ represents MLP with one hidden layer, W^Q , W^K and W^V represent the linear projections, and d is the dimension of projected matrices.

More details about distortion representation embedding is described in supplementary materials.

E. Loss Function

Since the entire framework is designed for downstream machine perception, its primary optimization target is the performance of downstream tasks. Additionally, for better mining machine-perception-related distortion information, feature distance is amplified by cosine similarity. In summary, the loss function \mathcal{L} of the framework is as follows.

$$\mathcal{L} = \mathcal{L}_{task} + \lambda \sum_{i=1}^3 (1 + \text{cosine}(F_i^o, F_i^c)) \quad (8)$$

where \mathcal{L}_{task} is downstream task loss and λ is balance weight.

III. EXPERIMENTAL RESULTS

A. Implementation Settings

1) Downstream Machine Vision Tasks: The proposed VCED framework is validated across three tasks with different granularities: video instance segmentation (Mask2Former [17]), person keypoints detection (Faster R-CNN [18]), and object detection (Keypoint R-CNN [16]), using YoutubEVIS-2019 (YTVIS2019) [17] and MS-COCO-2017 [16] datasets.

TABLE I
BD-RATE(%)↓ OF COMPARED METHODS ON DIFFERENT DOWNSTREAM TASKS. **BOLD** INDICATES THE BEST RESULT.

Methods	Object detection	Video instance segmentation	Keypoints detection
Direct test	0.0	0.0	0.0
Head finetune	-8.89	-32.54	-6.84
Full finetune	-58.53	-38.96	-29.18
PromptIR [21]	+11.57	+6.27	-1.39
PromptCIR [22]	+3.33	-3.05	-3.86
CascadeNet [6]	-14.39	-7.00	-5.18
Ours-FD	-21.70	-34.95	-9.83
Ours	-66.88	-53.68	-30.34

2) Compression Methods: Seven compression methods are used. For video compression, we employ standard codecs H.265/HEVC [13], H.264/AVC [19], and neural-based high-performance video codecs DCVC [3] and DCVC-DC [1]. For image compression, we use standard codecs H.265/HEVC-Intra, H.264/AVC-Intra, JPEG-2000 [20], and neural-based codec Cheng-2020 [5]. Furthermore, the VCM-related video codec SMC++ [10] is included in our experiment.

3) Compared Methods: We used two types of comparison methods. One method involves directly fine-tuning the downstream task model on compressed datasets, enabling models to fit on degraded images and videos. Another approach employs restoration-based models to improve the quality of compressed images and videos. Specifically, we choose the “all-in-one” restoration model PromptIR [21], the compression restoration model PromptCIR [22], and the recognition-oriented restoration model Cascaded Network (CascadeNet) [6].

4) Training Details: The framework includes codecs, task models, and CDRE-related modules. Video and image codecs remain fixed during training. In “Ours,” CDRE-related and task models are jointly optimized with the default configuration of downstream models; in “Ours-FD,” downstream models are frozen while CDRE-related modules are trained for 60k iterations. (More training details are described in supplementary materials.) Values of λ are 4.0 (video segmentation) and 0.1 (object detection and keypoint detection). More details are shown in the supplementary materials.

B. Rate-task Performance

The BD-rate (%) metric [1] represents the percentage of bitrate changing while achieving the same task performance compared to the anchor. The anchor is established by directly employing the downstream model on the compressed data.

1) With HVS-oriented Codec: Fig. 5 presents the results of three tasks across various bitrates and codecs. Our method significantly enhances rate-task performance with a minimal increase in bits per pixel (bpp). As shown in Table I, it reduces the BD-rate by 8.35% (object detection), 14.72% (instance segmentation), and 1.16% (keypoint detection) while fine-tuning downstream models. It saves 21.70% (object detection), 34.95% (instance segmentation), and 9.83% (keypoint detection) bitrates when downstream models are fixed. Due to the page limitation, the rate-task curve of keypoint detection is shown in supplementary materials. Visualization on video instance segmentation is shown in Fig. 8.

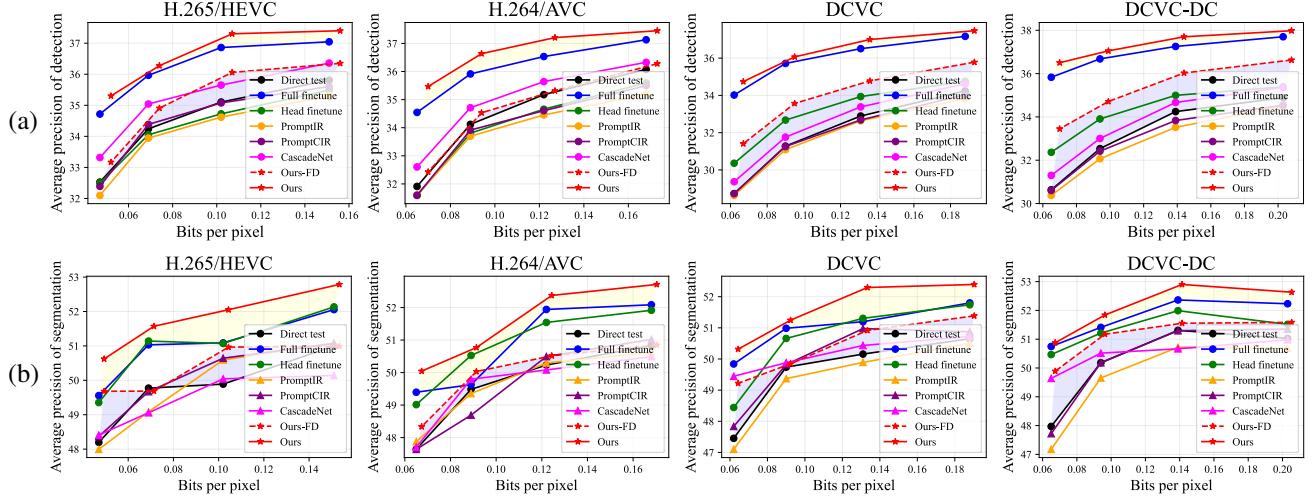


Fig. 5. The rate-task performance of (a) object detection and (b) Video instance segmentation across various existing HVS-oriented codecs. “Ours-FD” represents CDRE modules are optimized but the downstream model is frozen, “Ours” represents CDRE modules and the downstream model are jointly optimized. The average precision on uncompressed data is 37.3% (object detection) and 51.5% (instance segmentation).

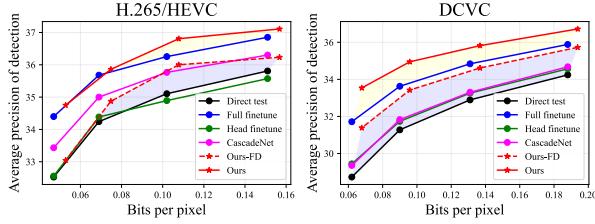


Fig. 6. Rate-task performance of the CDRE framework trained on H.264/AVC-compressed data but tested on video compressed by H.265/HEVC and DCVC.

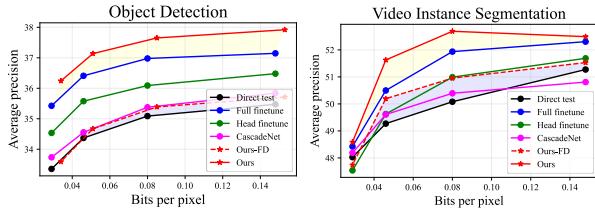


Fig. 7. Rate-task performance of CDRE framework with VCM-related codec SMC++ on object detection and video instance segmentation.

To validate the transferability of CDRE, it is trained on H.264/AVC-compressed videos but is tested on videos compressed by H.265/HEVC and DCVC. Fig. 6 shows that CDRE has strong transferability across different codecs and achieves the best performance.

2) *With VCM-related Codec:* The CDRE is also tested on VCM-related codec SMC++ [10]. As shown in Fig. 7, the CDRE further reduces the bitrate by 26.70% (object section) and 12.89% (instance segmentation) with fine-tuning downstream models. It saves 10.48% and 34.95% bitrates when downstream models are frozen.

The above experiments demonstrate that the CDRE effectively improves the rate-task performance of existing codecs. In contrast, training the model on compressed data and restoration-based methods struggles to achieve satisfactory overall performance. Notably, image restoration for HVS metrics (PromptIR and PromptCIR) offers few benefits to downstream task performance.

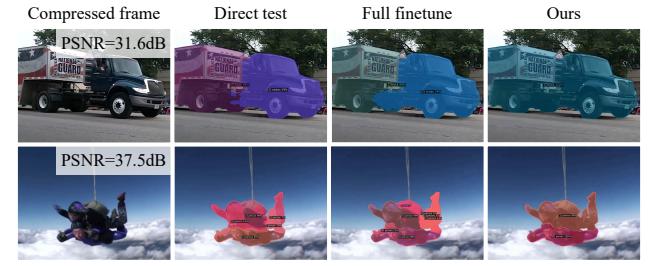


Fig. 8. Visualization of video instance segmentation. The above frames are compressed by H.264/AVC with $crf = 35$. Our proposed CDRE framework achieves better results after lossy compression.

TABLE II
ABLATION STUDY ON COMPRESSION-SENSITIVE EXTRACTOR. THE ANCHOR OF BITRATE IS OURS.

Multi-scale	Cosine Similarity	Modulation	Bitrate increase (%)
✓	✗	✓	+5.77
✗	✓	✓	+5.13
✗	✗	✓	+8.14
✓	✓	✗	+3.71

C. Overhead Analysis

The proposed CDRE framework is lightweight. The CNN-based version of the CDRE contains 1.9M parameters, while the Transformer-based version introduces 3.7M parameters. Notably, both versions share the same structure on the encoder side. The encoder part consists of only 18K parameters. For a 720P frame, the CDRE incurs only an additional 6 ms of computation time on NVIDIA RTX 3090. For 720P 30fps video, the additional distortion information bitstream requires about 20KBps, allowing for more bitrate savings of video streams while maintaining machine perception performance.

The MACs per pixel for each component are 0.89K on the encoder side, 0.33K for the Transformer-version decoder, and 1.61K for the CNN-version decoder. The MACs per pixel of CDRE is less than 2% of the video compression process (e.g., DCVC-DC [1] has 1.27M, DCVC [3] has 1.09M).

TABLE III

ABALATION STUDY ON PROGRESSIVE DISTORTION REPRESENTATION TRANSFORMATION AND EMBEDDING. BOLD INDICATES THE BEST RESULT. THE ANCHOR OF BD-RATE IS DIRECT TEST.				
Embedding layers	1	2	3	4
BD-rate (%)↓	-62.66	-62.86	-63.11	-66.88

TABLE IV

ABALATION STUDY ON CHANNELS OF COMPACT REPRESENTATION. BOLD INDICATES THE BEST RESULT. THE ANCHOR OF BD-RATE IS DIRECT TEST					
Channel count	1	3	6	10	16
BD-rate (%)↓	-64.14	-62.53	-66.88	-59.66	-55.49

D. Ablation Study

1) *Compression-sensitive Extractor*: As shown in Fig. II, to verify the effectiveness of amplifying distortion in the feature domain, cosine similarity is removed, leading to a 5.77% bitrate increase. Moreover, to assess the impact of multi-scale distortion analysis, we simplified the compression-sensitive extractor to a single scale, resulting in a 5.13% bitrate increase. When both ablations were applied, the bitrate increased by 8.14%. Additionally, linear modulation enables the distortion encoder to better leverage feature distortion, reducing the bitrate by 3.71% compared to simple concatenation. All ablation studies are conducted on object detection.

2) *Distortion Representation Embedding*: To explore the effect of progressively distortion representation transformation and embedding, we change the number of embedded layers. As shown in Table III, introducing more layers of transformation and embedding enables the downstream models to better leverage distortion information.

3) *Influence on the Dimension of Compact Representation*: Although transmitting and embedding distortion representation can improve task accuracy, it also introduces bitrate overhead. The dimension of compact representation must strike a balance between accuracy gain and bitrate cost. To address this, different numbers of channels are tested. As shown in table IV, the best rate-task performance is achieved with 6 channels.

IV. CONCLUSION

In this paper, we propose a Compression Distortion Representation Embedding (CDRE) framework to enhance the rate-task performance of existing codecs. We develop a compression-sensitive extractor to amplify and identify distortion in the feature domain. Moreover, a lightweight distortion codec is introduced to compress distortion information into a compact representation for effective transmission. Additionally, we introduce a distortion representation embedding module that progressively transforms and embeds the distortion feature into downstream models, making them aware of compression degradation of the current input. Our framework is evaluated on three downstream tasks—object detection, video instance segmentation, and person keypoint detection—across seven codecs. Experimental results show that the proposed CDRE framework significantly improves the rate-task performance with minimal overhead in terms of bitrate, execution time, and number of parameters.

REFERENCES

- [1] Jiahao Li, Bin Li, and Yan Lu, “Neural video compression with diverse contexts,” in *Proceedings of CVPR*, 2023, pp. 22616–22626.
- [2] Xihua Sheng, Li Li, Dong Liu, and Houqiang Li, “Spatial decomposition and temporal fusion based inter prediction for learned video compression,” *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.
- [3] Jiahao Li, Bin Li, and Yan Lu, “Deep contextual video compression,” in *Proceedings of NeurIPS*, 2021, pp. 18114–18125.
- [4] Yifan Zhang, Meiqin Liu, Chenming Xu, Qi Tang, Chao Yao, and Yao Zhao, “TLVC: Temporal bit-rate allocation for learned video compression,” in *Proceedings of ICME*. IEEE, 2024, pp. 1–6.
- [5] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto, “Learned image compression with discretized gaussian mixture likelihoods and attention modules,” in *Proceedings of CVPR*, 2020, pp. 7939–7948.
- [6] Ding Liu, Bihan Wen, Xianming Liu, Zhangyang Wang, and Thomas S Huang, “When image denoising meets high-level vision tasks: A deep learning approach,” *arXiv preprint arXiv:1706.04284*, 2017.
- [7] Chao Dong, Yubin Deng, Chen Change Loy, and Xiaoou Tang, “Compression artifacts reduction by a deep convolutional network,” in *Proceedings of ICCV*, 2015, pp. 576–584.
- [8] Nanfeng Jiang, Weiling Chen, Jielian Lin, Tiesong Zhao, and Chia-Wen Lin, “Video compression artifacts removal with spatial-temporal attention-guided enhancement,” *IEEE Transactions on Multimedia*, vol. 26, pp. 5657–5669, 2024.
- [9] Yuan Tian, Guo Lu, Guangtao Zhai, and Zhiyong Gao, “Non-semantics suppressed mask learning for unsupervised video semantic compression,” in *Proceedings of ICCV*, 2023, pp. 13610–13622.
- [10] Yuan Tian, Guo Lu, and Guangtao Zhai, “SMC++: Masked learning of unsupervised video semantic compression,” *arXiv preprint arXiv:2406.04765*, 2024.
- [11] Yuan Tian, Guo Lu, Yichao Yan, Guangtao Zhai, Li Chen, and Zhiyong Gao, “A coding framework and benchmark towards compressed video understanding,” *arXiv preprint arXiv:2202.02813*, 2022.
- [12] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim, “Visual prompt tuning,” in *Proceedings of ECCV*. Springer, 2022, pp. 709–727.
- [13] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand, “Overview of the High Efficiency Video Coding (HEVC) standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.
- [14] Johannes Ballé, Valero Laparra, and Eero P Simoncelli, “End-to-end optimized image compression,” *arXiv preprint arXiv:1611.01704*, 2016.
- [15] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao, “DVC: An end-to-end deep video compression framework,” in *Proceedings of CVPR*, 2019, pp. 11006–11015.
- [16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.
- [17] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar, “Masked-attention mask transformer for universal image segmentation,” in *Proceedings of CVPR*, 2022, pp. 1290–1299.
- [18] Yushe Cao, Xin Niu, and Yong Dou, “Region-based convolutional neural networks for object detection in very high resolution remote sensing images,” in *Proceedings of International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*. IEEE, 2016, pp. 548–554.
- [19] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra, “Overview of the H.264/AVC video coding standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.
- [20] Michael W Marcellin, Michael J Gormish, Ali Bilgin, and Martin P Bolek, “An overview of JPEG-2000,” in *Proceedings of DCC*, 2000, pp. 523–541.
- [21] Vaishnav Potlapalli, Syed Waqas Zamir, Salman Khan, and Fahad Khan, “PromptIR: Prompting for all-in-one image restoration,” in *Proceedings of NeurIPS*, 2023.
- [22] Bingchen Li, Xin Li, Yiting Lu, Ruoyu Feng, Mengxi Guo, Shijie Zhao, Li Zhang, and Zhibo Chen, “PromptCIR: Blind compressed image restoration with prompt learning.,” *arXiv preprint arXiv:2404.17433*, 2024.