# Supplementary Materials

### A. *Correlation between Pixel-domain and HVS-related Metrics with Machine Perception.*

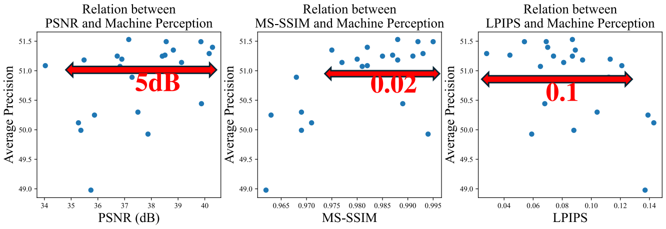

Fig. 1. The correlation between widely used metrics (PSNR, MS-SSIM, and LPIPS) with downstream machine vision performance on Mask2Former [1] and YTVIS-2019 dataset [2].

We evaluated the performance on Video Instance Segmentation (VIS) of the mainstream standard codec H.265/HEVC [3] and neural-based codecs, including DCVC-FM [4], DCVC-DC [5], DCVC-TCM [6]. The pixel-domain metric PSNR, and HVS-related metrics MS-SSIM and LPIPS of compressed videos are recorded. As shown in Fig. 1, there is no strong correlation between the HVS-related distortion metric and the machine perception performance, highlighting the importance of introducing machine-perception-oriented distortion.

### B. *Structure of Distortion Decoder*

TABLE I
DECODING PROCESS OF DISTORTION DECODER FOR DIFFERENT DOWNSTREAM MODELS.

| Layer | For CNN | For Transformer |
|---|---|---|
| $\hat{y}$ | $6, \frac{h}{32}, \frac{w}{32}$ | $6, \frac{h}{32} \times \frac{w}{32}$ |
| 1 | $11, \frac{h}{16}, \frac{w}{16}$ | $51, \frac{h}{32} \times \frac{w}{32}$ |
| 2 | $11, \frac{h}{8}, \frac{w}{8}$ | $51, \frac{h}{32} \times \frac{w}{32}$ |
| 3 | $16, \frac{h}{4}, \frac{w}{4}$ | $96, \frac{h}{32} \times \frac{w}{32}$ |

The distortion decoder contains three stages. Each stage contains a convolutional layer, a BatchNorm layer, and a ReLU layer. 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. The decoding process is shown in TABLE I.

### C. *Structure of Distortion Representation Embedding Module*

To maintain consistency with downstream models, the progressive transformation of the distortion feature is performed through convolution for CNN-based downstream backbones. Each convolutional layer is followed by a BatchNorm layer and a ReLU layer. Meanwhile, MLP with one hidden layer is used for Transformer-based downstream backbones.

TABLE II
STRUCTURE OF TRANSFORMATION MODULE. SYMBOL ↓ MEANS DOWN-SAMPLING WITH STRIDE=2.

| Layer | For CNN | For Transformer |
|---|---|---|
| 1 | Conv2d(in=16, out=16)<br>Conv2d(in=16, out=16)<br>Conv2d(in=16, out=16) | MLP(in=96, hidden=96, out=96) |
| 2 | Conv2d(in=16, out=40)↓<br>Conv2d(in=40, out=40)<br>Conv2d(in=40, out=64) | MLP(in=96, hidden=96, out=192) |
| 3 | Conv2d(in=64, out=96)↓<br>Conv2d(in=96, out=96)<br>Conv2d(in=96, out=128) | MLP(in=192, hidden=192, out=384) |
| 4 | Conv2d(in=128, out=142)↓<br>Conv2d(in=142, out=142)<br>Conv2d(in=142, out=256) | MLP(in=384, hidden=384, out=768) |

### D. *Configurations of Standard Codecs*

H.265/HEVC and H.264/AVC in the experiments in implemented by libx265 and libx264 in FFmpeg [7]. The command for calling libx265 is as follows.

```
FFREPORT=file=ffreport.log:level=56 ffmpeg
-pix_fmt yuv420p
-s <width>x<height>
-i <input_path>
-c:v libx265
-tune zerolatency
-x265-params "crf=<crf>:keyint=16:verbose=1" out.mkv
```

Additionally, the command line of libx264 is as follows.

```
FFREPORT=file=ffreport.log:level=56 ffmpeg
-pix_fmt yuv420p
-s <width>x<height>
-i <input_path>
-c:v libx264
-tune zerolatency
-x264-params "crf=<crf>:keyint=16:verbose=1" out.mkv
```

### E. *Training Details*

For ”Ours”, our proposed CDRE-related modules and downstream models are jointly optimized while codecs are fixed, following configurations of downstream models. (1) For Video Instance Segmentation, we employ Mask2Former-video [1] and follows the configuration in ***video_maskformer2_swin_tiny_bs16_8ep.yaml*** from its released codebase. (2) For Keypoint Detection, we use the Keypoint R-CNN model [8], following the configuration in ***keypoint_rcnn_R_50_FPN_1x.yaml*** from the Detectron2 codebase [9]. (3) For Object Detection, we employ the Faster R-CNN model [10], following the configuration in ***faster_rcnn_R_50_FPN_1x.yaml*** from the Detectron2 codebase.

For ”Ours-FD”, our proposed CDRE-related modules are optimized while codecs and downstream models are fixed. The optimization includes $40,000$ iterations at $lr = 1e-4$ and $20,000$ iterations at $lr = 1e-5$ with $batchsize = 4$.

### *F. Rate-task Performance on Keypoint-detection*

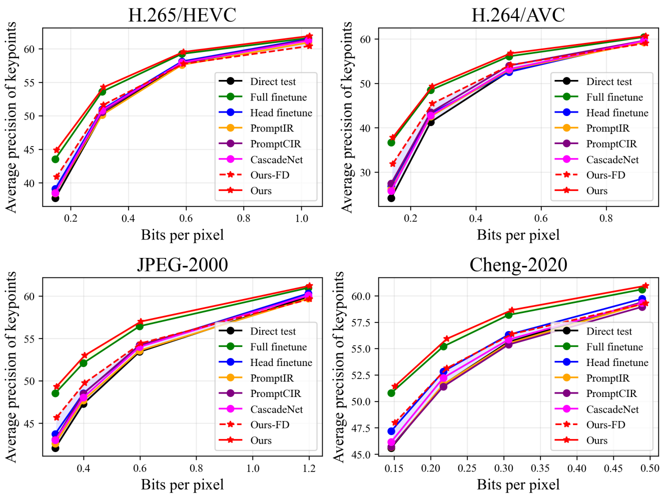

Fig. 2. Rate-task curve of Keypoint Detection on MS-COCO-2017 dataset. The average precision on uncompressed data is 64.0%.

Due to the page limitation of the main PDF, we provide the rate-task curve of the CDRE framework for the Keypoint Detection task in Fig. 2.

### *G. Detailed Complexity Analysis*

Taking the video instance segmentation model Mask2Former-video-tiny [1] as an example, we record the number of introduced parameters and optimized parameters, as well as corresponding BD-rate across different compared methods, as shown in Table III. Our method only introduced a small number of parameters, 3.8M. Especially in the “Ours-FD” case, where all parameters of the downstream model are fixed and only the CDRE-related modules are optimized, our method achieved an effective BD-rate drop by training just a few parameters.

### *H. Analysis for Rate-task Performance Difference*

The proposed CDRE framework achieves effective bitrate saving, but the rate-task performance improvement varies across different tasks. When the downstream network is fixed, our method can save bitrate by $34.95\%$ (video instance segmentation), $21.70\%$ (object detection), and $9.83\%$ (keypoint detection). When the downstream task is optimized, compared to full fine-tuning, our method can save bitrate by $14.72\%$ (video instance segmentation), $8.35\%$ (object detection), and $1.16\%$ (keypoint detection). A clear trend shows that fine-grained tasks are more sensitive to compression distortion, making CDRE more effective in improving rate-task performance for these tasks

TABLE III
INTRODUCED PARAMS, OPTIMIZED PARAMS, AND BD-RATE(%)↓. THE ANCHOR FOR CALCULATING BD-RATE IS “DIRECT TEST”.

| Method | Introduced params | Optimized params | BD-rate(%)↓ |
|---|---|---|---|
| Direct test | 0 | 0 | 0 |
| Head finetune | 0 | 16.1M | -32.54 |
| Full finetune | 0 | 47.4M | -38.96 |
| PromptIR | 36.5M | 0 | +6.27 |
| PromptCIR | 34.8M | 0 | -3.05 |
| CascadeNet | 1.0M | 1.0M | -7.00 |
| Ours-FD | 3.8M | 3.8M | -34.95 |
| Ours | 3.8M | 51.2M | -53.68 |

TABLE IV
BD-RATE(%)↓ BASED ON AP, AP50, AND AP75 OF OBJECT DETECTION. THE ANCHOR FOR CALCULATING BD-RATE IS “DIRECT TEST”. **BOLD** INDICATES THE BEST.

| Method | AP | AP50 | AP75 |
|---|---|---|---|
| Direct test | 0 | 0 | 0 |
| Head finetune | -8.89 | -7.85 | -19.10 |
| Full finetune | -58.53 | -49.90 | -64.03 |
| PromptIR | +11.57 | +14.54 | +16.18 |
| PromptCIR | +3.33 | +4.47 | +4.42 |
| CascadeNet | -14.39 | -16.40 | -12.42 |
| Ours-FD | -21.70 | -27.32 | -21.22 |
| Ours | **-66.88** | **-56.54** | **-66.78** |

### *I. Further Report Rate-task Performance*

In the submitted manuscript, BD-rate is calculated based on Average Precision (AP), since AP is the most commonly used metric for evaluating segmentation and detection accuracy. Considering the valuable comments from the reviewers, we also calculated the BD-rate based on AP50 and AP75 to show that our CDRE framework can improve detection performance under different IoU thresholds, as shown in Table IV.

## REFERENCES

[1] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar, “Masked-attention mask transformer for universal image segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 1290–1299.

[2] Linjie Yang, Yuchen Fan, and Ning Xu, “Video instance segmentation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 5188–5197.

[3] 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.

[4] Jiahao Li, Bin Li, and Yan Lu, "Neural video compression with feature modulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[5] Jiahao Li, Bin Li, and Yan Lu, "Neural video compression with diverse contexts," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[6] Xihua Sheng, Jiahao Li, Bin Li, Li Li, Dong Liu, and Yan Lu, "Temporal context mining for learned video compression," *IEEE Transactions on Multimedia*, vol. 25, pp. 7311–7322, 2022.

[7] FFmpeg, ," https://github.com/FFmpeg/, 2024, Accessed 2024-1-31.

[8] 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.

[9] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[10] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2015.