DECODEALIGN: POST-DECODING REPRESENTATION ALIGNMENT FOR VIDEO CODING FOR MACHINES

Anonymous authors
Paper under double-blind review

ABSTRACT

With the increasing use of videos for machine understanding, the objectives of human-oriented video coding differ from those of machine tasks, making the improvement of machine readability for compressed videos an important challenge. However, existing approaches for machine tasks often require modifications to the coding process, new bitstream designs, or joint optimization with machine models, limiting their applicability in existing video coding ecosystems. In this paper, we revisit VCM from a representation perspective and reveal that the machine readability of compressed videos depends not only on the information preserved during encoding, but also on the effectiveness of decoded content representations for downstream machine models, which is not constrained by the original coding process. Based on this observation, we propose DecodeAlign, a post-decoding representation alignment framework for compressed video machine tasks. DecodeAlign keeps the original codecs, bitstream structures, and downstream task models unchanged, and introduces an independent representation alignment network after decoding to reduce the representation discrepancy between decoded and original content representations in the representation space of machine models, thereby improving machine task performance. Experiments demonstrate that DecodeAlign effectively improves machine task performance across eight host codecs, achieving up to 45.6% BD-rate reduction, and generalizes across codecs according to compression distortion distributions.

1 INTRODUCTION

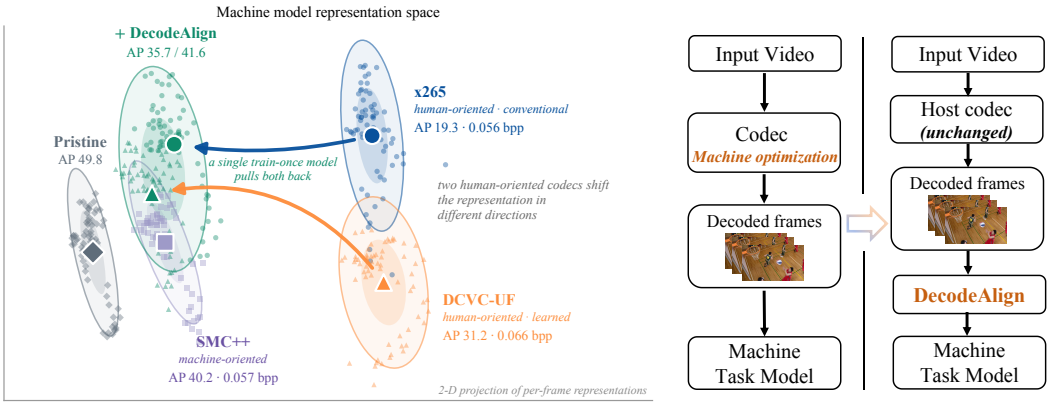
Video compression has traditionally been optimized for human perception, aiming to reconstruct visually faithful frames under limited bitrates. However, an increasing amount of visual data is now directly processed by machine vision systems for downstream tasks such as object detection, tracking, and retrieval. Human-oriented compression objectives do not fully satisfy the requirements of machine tasks, motivating the emergence of video coding for machines (VCM) to bridge this gap and improve the *machine readability* of compressed videos, i.e., how well decoded content supports downstream machine models (Duan et al., 2020).

Over the past decade, VCM has evolved through several design paradigms (Yang et al., 2024). One line redesigns the coding system to directly optimize machine tasks, where end-to-end semantic codecs learn representations tailored to downstream perception (Tian et al., 2026; Sun et al., 2026a). Another retains conventional codecs while introducing task-aware modules, such as resampling, region-of-interest processing, or learned coding tools, to adapt compression toward machine objectives (ISO/IEC JTC 1/SC 29/WG 04, 2025; Ahonen et al., 2023). Some studies further explore scalable coding with separated machine- and human-oriented representations (Hadizadeh & Bajić, 2024) or compress intermediate features for direct machine consumption (Sun et al., 2026b). Despite their differences, these approaches all adapt the coding process to preserve information beneficial for machine inference and incorporate machine objectives into the coding system.

Does machine readability necessarily need to be established within the coding process?

As shown in Fig. 1(a), in the representation space of a downstream machine model, the decoded representations produced by the human-oriented codecs x265 and DCVC-UF deviate from those

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Representation-space visualization (b) Overview of DecodeAlign

Figure 1: Representation-space motivation and overview of DecodeAlign. (a) Per-frame representations from a frozen downstream machine model are jointly projected into 2-D at matched bitrates. The clusters correspond to pristine frames, codec-decoded frames, and the corresponding frames after DecodeAlign. (b) Existing VCM incorporates machine-oriented optimization into the codec, whereas DecodeAlign introduces an independent representation alignment stage after an unchanged host codec and before the machine task model.

of the original content, whereas the machine-oriented codec SMC++ produces representations that remain closer to the original. This observation motivates us to revisit VCM from a representation-space perspective: machine readability depends not only on the amount of task-relevant information preserved in the decoded content (*information availability*), but also on whether the preserved information can be effectively represented and utilized by the downstream machine model (*representation availability*). Since information availability is directly constrained by the coding process, whereas representation availability concerns how the machine model exploits the information already available after decoding, the latter does not necessarily require modifications to the codec. Therefore, machine readability does not have to be fully established within the coding process; the component related to representation availability can be decoupled from coding and optimized independently after decoding.

To realize post-decoding adaptation in VCM, we propose **DecodeAlign**, a framework that improves representation availability without modifying the original coding system. As illustrated in Fig. 1(b), DecodeAlign applies a frame-domain mapping to decoded frames and learns this mapping under the supervision of a frozen machine model, encouraging the representations of the adapted frames in its representation space to approach those of the pristine content. The encoder, decoder, bitstream, and downstream machine model remain unchanged throughout the process, requiring no task annotations. As shown in Fig. 1(a), applying DecodeAlign to the decoded outputs of x265 and DCVC-UF shifts their representations in the machine model's representation space toward the pristine cluster while improving detection performance. These results further demonstrate that representation availability is an important factor in machine readability and can be effectively enhanced after decoding. Further investigation reveals that machine-oriented adaptation does not require codec-by-codec design, but exhibits a distortion-family-level generalization pattern. A DecodeAlign model trained with a representative codec generalizes to unseen codecs within the same distortion family, while generalization across distortion families remains weaker.

Our contributions are summarized as follows:

(1) We revisit VCM from a machine representation perspective and characterize machine readability through two dimensions: information availability and representation availability. We reveal that post-compression machine performance is determined not only by information preservation during coding, but also by the effectiveness of representations induced from decoded content, indicating that representation availability does not necessarily need to be established during the coding process.

(2) We propose DecodeAlign, a post-decoding representation alignment framework for machine tasks. Without modifying the existing coding system, DecodeAlign learns a mapping between

decoded-frame representations and the corresponding representations of the original content, and improves the machine readability of compressed content through representation alignment.

(3) We conduct experiments on eight host codecs spanning conventional, learned, and machine-oriented coding systems. DecodeAlign consistently improves machine-task performance, achieving up to 45.6% and 28.0% BD-rate reductions on DCVC-FM and VVenC, respectively, while revealing that post-decoding representation enhancement follows compression distortion families rather than individual codec identities.

2 Related Work

Video Compression Development. Video compression has evolved from handcrafted hybrid coding frameworks to data-driven learned models. Traditional codecs adopt hybrid architectures that combine prediction, transform, quantization, and entropy coding, with compression efficiency improved through rate-distortion optimization and the refinement of coding tools. Representative standards, including H.264/AVC (Wiegand et al., 2003), HEVC (Sullivan et al., 2012), and VVC (Bross et al., 2021), progressively enhance coding performance through advances in intra/inter prediction, motion compensation, transform coding, and entropy modeling. These codecs establish the foundation of modern video compression by optimizing handcrafted components under the rate-distortion framework.

With the development of deep learning, learned compression methods have introduced data-driven optimization into the coding pipeline. Ballé et al. proposed end-to-end learned image compression frameworks that replace handcrafted transforms and entropy models with neural networks and jointly optimize rate-distortion objectives (Ballé et al., 2017; 2018). Following this direction, learned video compression methods further incorporate neural networks into motion estimation, motion compensation, and residual coding. DVC (Lu et al., 2019) explored an end-to-end video compression framework based on neural motion estimation and autoencoders, while subsequent DCVC series methods (Li et al., 2021; 2022; 2023; Jia et al., 2025) further improved compression efficiency through conditional probability modeling and temporal context utilization.

Video Coding for Machines. As visual data are increasingly consumed by machine vision systems, video compression has expanded from human-oriented reconstruction toward supporting machine-oriented analysis, leading to the emergence of Video Coding for Machines (VCM) (Duan et al., 2020; Yang et al., 2024). Existing VCM studies have explored several technical directions to enable downstream machine tasks from compressed visual data.

The first direction focuses on *task-driven video coding*, where machine objectives are incorporated into the coding process. Early approaches mainly extend conventional video coding frameworks with task-aware modules. For example, MPEG VCM reference software (VCM-RS) and NN-VVC investigate the integration of region adaptation, resampling strategies, and neural coding tools into existing codecs to improve machine task performance (ISO/IEC JTC 1/SC 29/WG 04, 2025; Ahonen et al., 2023; Zhang et al., 2024; Sun et al., 2025). Building upon these efforts, end-to-end semantic coding methods further optimize the coding process jointly with downstream task objectives. Representative approaches, including SEC-VCM and SMC++, learn task-oriented latent codes that preserve information beneficial for machine inference during encoding (Sun et al., 2026a; Tian et al., 2026).

The second direction investigates *human-machine scalable video coding*, which aims to support both machine analysis and human viewing within a unified compression framework. These approaches typically employ hierarchical coding structures, where different layers serve different objectives. For example, scalable coding frameworks based on base and enhancement layers allocate machine-oriented information to base layers while using enhancement layers to improve human reconstruction quality (Hadizadeh & Bajić, 2024).

The third direction explores *feature-domain coding*, where the transmitted content is shifted from pixel-domain videos to intermediate machine features. Instead of reconstructing videos at the receiver, these methods compress and transmit compact visual features or tokens for direct machine consumption. For example, TransVFC investigates feature-domain video compression by encoding intermediate features and introducing feature transformation mechanisms to support downstream machine tasks (Sun et al., 2026b).

3 METHOD

We revisit machine readability from the perspective of a machine model's representation space, focusing not only on the task-relevant information preserved after compression but also on how decoded content deviates from the original input in representation space. Based on this perspective, we propose **DecodeAlign**, a post-decoding representation alignment framework that improves machine readability by reducing compression-induced representation mismatch.

3.1 REPRESENTATION-SPACE FORMULATION

Let $X \in \mathcal{X}$ denote a source video frame, and let $Y_R = C_R(X)$ denote the corresponding decoded frame produced by a codec operating at rate R, where C_R abstracts the complete encoding–decoding process, including temporal prediction when present. Let B_R denote the corresponding bitstream and $\ell(B_R)$ its length. Conventional human-oriented compression is typically optimized under a rate–distortion objective,

$$\min_{C_R} \mathbb{E}[d_H(X, Y_R)] \quad \text{s.t.} \quad \mathbb{E}[\ell(B_R)] \le R, \tag{1}$$

where d_H denotes a reconstruction distortion.

Machine perception introduces another space in which the decoded content is evaluated. Let $F : \mathcal{X} \to \mathcal{Z}$ be a frozen machine representation extractor, and define the representation of the original frame as $Z = F(X)$. The same machine model maps the decoded frame to $F(Y_R)$. We therefore characterize compression-induced machine degradation through the representation distortion

$$\mathcal{D}_F(R) = \mathbb{E}\left[\|F(Y_R) - Z\|_2^2\right]. \tag{2}$$

Equation (1) places no constraint on $F(Y_R)$, so a small $d_H(X, Y_R)$ does not bound $\mathcal{D}_F(R)$: a codec may attain a visually plausible reconstruction while inducing a substantial deviation in representation space. Conversely, a large $\mathcal{D}_F(R)$ does not imply that the corresponding machine-relevant information has been irreversibly removed from Y_R.

3.2 IRRECOVERABLE DEGRADATION AND REPRESENTATION MISMATCH

For a decoded observation Y_R, define the Bayes-optimal estimate of the original machine representation as

$$Z_R^*(Y_R) = \mathbb{E}[Z \mid Y_R]. \tag{3}$$

Under squared error, $Z_R^*(Y_R)$ is the best representation estimate attainable from the information contained in Y_R, without restricting the estimator to a particular architecture.

Proposition 1 (Representation-error decomposition). Assume $\mathbb{E}\|Z\|_2^2 < \infty$. For any measurable representation estimator $h(Y_R)$ with $\mathbb{E}\|h(Y_R)\|_2^2 < \infty$,

$$\begin{aligned} \mathbb{E}\,\|Z - h(Y_R)\|_2^2 = &\underbrace{\mathbb{E}\,\|Z - Z_R^*(Y_R)\|_2^2}_{\mathcal{L}_{\mathrm{irr}}(R)} \\ &+ \underbrace{\mathbb{E}\,\|Z_R^*(Y_R) - h(Y_R)\|_2^2}_{\mathcal{L}_{\mathrm{mis}}(h;R)}. \end{aligned} \tag{4}$$

Here, $\mathcal{L}_{\mathrm{irr}}(R)$ is an observation-limited irreducible component determined by the information available in Y_R, whereas $\mathcal{L}_{\mathrm{mis}}(h; R)$ measures the excess representation error of a particular estimator relative to the Bayes estimate. The proof is provided in Appendix A.

For the unprocessed decoded frame, $h(Y_R) = F(Y_R)$, and Eq. (4) becomes

$$\mathcal{D}_F(R) = \mathcal{L}_{\mathrm{irr}}(R) + \mathbb{E}\left[\|Z_R^*(Y_R) - F(Y_R)\|_2^2\right]. \tag{5}$$

The two terms of Eq. (5) have distinct origins. $\mathcal{L}_{\mathrm{irr}}(R)$ is the uncertainty about Z that cannot be resolved from Y_R. The second term measures the extent to which applying the fixed machine model

directly to the decoded frame fails to realize the best representation still supported by that observation; we call it *representation mismatch*. Only the second term depends on how Y_R is processed, and is therefore reducible after decoding. The two terms correspond to the *information* and *representation availability* of Sec. 1.

3.3 Post-Decoding Representation Alignment

Since representation mismatch is reducible after decoding, we keep the codec unchanged and introduce a post-decoding mapping $\widetilde{Y}_R = G_\theta(Y_R)$, where $G_\theta : \mathcal{X} \to \mathcal{X}$ maps a decoded frame to a video frame of the same spatial resolution. We refer to this stage as **DecodeAlign**.

The representation distortion after alignment is

$$\mathcal{D}_F^G(R) = \mathbb{E}\left[\|Z - F(G_\theta(Y_R))\|_2^2\right]. \tag{6}$$

Substituting $h(Y_R) = F(G_\theta(Y_R))$ into Eq. (4) gives

$$\mathcal{D}_F^G(R) = \mathcal{L}_{\mathrm{irr}}(R) + \mathbb{E}\left[\|Z_R^*(Y_R) - F(G_\theta(Y_R))\|_2^2\right]. \tag{7}$$

Equations (5) and (7) read $\mathcal{L}_{\mathrm{irr}}(R)+\mathcal{L}_{\mathrm{mis}}(F;R)$ and $\mathcal{L}_{\mathrm{irr}}(R)+\mathcal{L}_{\mathrm{mis}}(F\circ G_\theta;R)$: only the mismatch term depends on the post-decoding mapping. By the data-processing inequality, $I(Z;G_\theta(Y_R)) \le I(Z;Y_R)$, so DecodeAlign cannot recover information no longer inferable from Y_R; it can only reshape $F(G_\theta(Y_R))$ relative to $Z_R^*(Y_R)$ within the space induced by the fixed F, and under the frame-domain restriction this term need not be eliminable.

Machine-oriented adaptation is thus performed outside the codec,

$$X \xrightarrow{C_R} Y_R \xrightarrow{G_\theta} \widetilde{Y}_R \xrightarrow{F} F(\widetilde{Y}_R). \tag{8}$$

The bitstream syntax, encoder, decoder, and rate accounting therefore remain unchanged, and $\widetilde{Y}_R$ is consumed by an existing machine-vision pipeline without a codec-specific representation interface.

3.4 Frame-Domain Representation Alignment

We instantiate G_θ with an unmodified NAFNet (Chen et al., 2022), which takes and returns RGB frames in $[0,1]$ of identical size. We use width 32, 12 middle blocks, encoder depths $[2,2,4,8]$, and decoder depths $[2,2,2,2]$, resulting in 29.2M parameters. The original global skip connection of NAFNet is retained, so the mapping can be written as

$$G_\theta(Y) = \mathrm{clip}\left(Y + \Delta_\theta(Y),\, 0,\, 1\right). \tag{9}$$

Each decoded frame is mapped independently; no codec-specific component and no temporal input are introduced.

Machine-oriented feature alignment. For practical learning, we instantiate F using the frozen bottom-up backbone of a Faster R-CNN ResNeXt-101-32×8d-FPN detector and extract the pre-FPN features

$$\mathcal{S} = \{\mathrm{res2}, \mathrm{res3}, \mathrm{res4}, \mathrm{res5}\}. \tag{10}$$

Only the frozen backbone is used; no detection head is involved and no detection annotations are required during DecodeAlign training. Task-specific supervision is provided solely through the frozen detector features, and inputs follow the detector's own preprocessing exactly (BGR ordering, $[0,255]$ scaling, per-channel mean and standard-deviation normalization).

For an input frame A, let

$$f_l(A) \in \mathbb{R}^{H_l \times W_l \times C_l} \tag{11}$$

denote the feature map at level l. We compute cosine distance along the channel dimension at each spatial location and average over all spatial locations and the four feature levels,

$$\mathcal{L}_{\mathrm{feat}} = \frac{1}{|\mathcal{S}|}\sum_{l\in\mathcal{S}} \frac{1}{H_l W_l}\sum_p \left[1 - \frac{\left\langle f_l(\widetilde{Y})_p, f_l(X)_p \right\rangle}{\left\| f_l(\widetilde{Y})_p \right\|_2 \left\| f_l(X)_p \right\|_2}\right]. \tag{12}$$

No global pooling is applied; the ℓ_2 normalization enters only through the cosine distance, not as a separate preprocessing step.

Since $\|\hat{u} - \hat{v}\|_2^2 = 2(1 - \cos(u, v))$, each term of Eq. (12) is half a squared Euclidean distance between ℓ_2-normalized features. Applying Proposition 1 to each term and averaging over positions and levels decomposes the expected feature loss into a θ-independent irreducible part and a θ-dependent mismatch part; an ε-stabilized denominator handles near-zero norms, where the identity is approximate.

Pixel-domain anchor. We additionally use a Charbonnier reconstruction loss,

$$\mathcal{L}_{\text{pix}} = \text{mean}\sqrt{\left(\widetilde{Y} - X\right)^2 + 10^{-6}}. \tag{13}$$

The final objective is

$$\mathcal{L} = 0.7\,\mathcal{L}_{\text{pix}} + 0.3\,\mathcal{L}_{\text{feat}}. \tag{14}$$

Here $\mathcal{L}_{\text{feat}}$ drives machine-oriented representation alignment, while $\mathcal{L}_{\text{pix}}$ anchors the output to the original visual content. Training-pair construction, optimization details, and the deployment protocol across codecs follow this objective and are described in Sec. 4.1.

4 Experiments

4.1 Experimental Setup

Datasets and task. We evaluate DecodeAlign on object detection using the 14 SFU-HW test sequences (Choi et al., 2021). Following the VCM setting, detection performance is used as the measure of machine readability. DecodeAlign is trained on 6,000 Vimeo-90k septuplets (Xue et al., 2019) generated from x265 and DCVC-UF compressed pairs with a 60:40 ratio. SFU-HW sequences and detection annotations are excluded from training, and other host codecs are evaluated zero-shot. The main experiments use a frozen Faster R-CNN with a ResNeXt-101-FPN backbone (Wu et al., 2019), while generalization to an unseen YOLOv8m detector (Jocher et al., 2023) is studied in the ablation.

Host codecs. We compare DecodeAlign with host-decoded outputs from diverse codecs, including human-oriented conventional codecs (x265 (MulticoreWare, 2023), VTM-24.0 (JVET, 2024), and VVenC (Wieckowski et al., 2021)), human-oriented learned codecs (DCVC-UF (Li et al., 2026), DCVC-FM (Li et al., 2024), and DCVC-HEM (Li et al., 2022)), and machine-oriented codecs (Learned-SVC (Hadizadeh & Bajić, 2024) and SMC++ (Tian et al., 2026)). DecodeAlign is applied after decoding without modifying the original codec, bitstream, or machine model.

Evaluation metrics. All results are reported as mAP@[.5:.95] averaged over 14 sequences. The uncompressed videos achieve 51.17 mAP as the reference. We evaluate performance using rate–accuracy curves, BD-rate (Bjøntegaard, 2001), and ΔmAP, where ΔmAP measures the mAP gain of DecodeAlign-enhanced outputs over host-decoded outputs at matched bitrates.

Implementation details. DecodeAlign is based on a NAFNet (Chen et al., 2022) backbone with 29.2M parameters. It is trained with AdamW for 60K iterations using a joint objective of pixel-level reconstruction and feature-level alignment losses. During inference, DecodeAlign operates as an independent post-decoding module without introducing additional transmission overhead.

4.2 Main comparison

Figure 2(a)–(c) present the rate–accuracy curves of different host codecs before and after applying DecodeAlign, grouped by codec type, while Fig. 2(d) summarizes the corresponding gains at matched bitrates. Overall, DecodeAlign consistently improves machine task performance across diverse host codecs, shifting the rate–accuracy curves toward higher accuracy regions. These results show that post-decoding representation alignment can enhance machine readability without modifying the existing coding system. Specifically, DecodeAlign achieves maximum BD-rate reductions

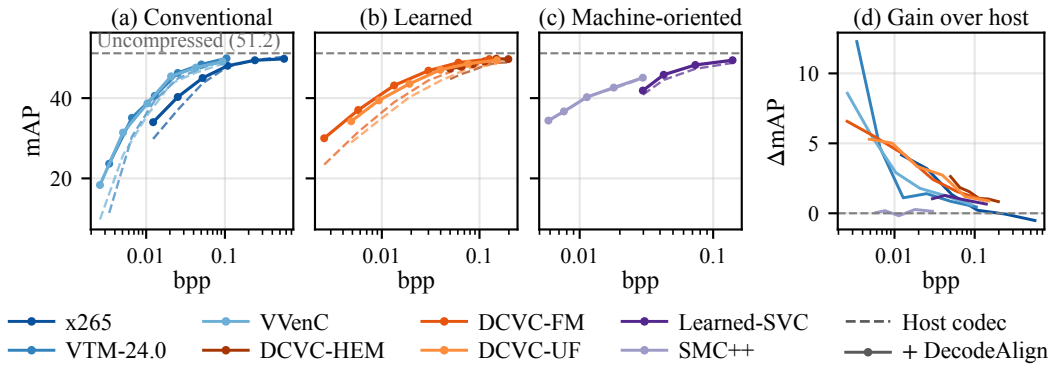


Figure 2: Average mAP–rate curves on SFU-HW. Panels (a)–(c) show human-oriented conventional, human-oriented learned, and machine-oriented codecs, respectively. Panel (d) reports the mAP gain of applying DecodeAlign to host codecs at matched bitrates. Dashed and solid curves denote host codecs and host codecs with DecodeAlign, respectively; the gray dashed line indicates the uncompressed reference.

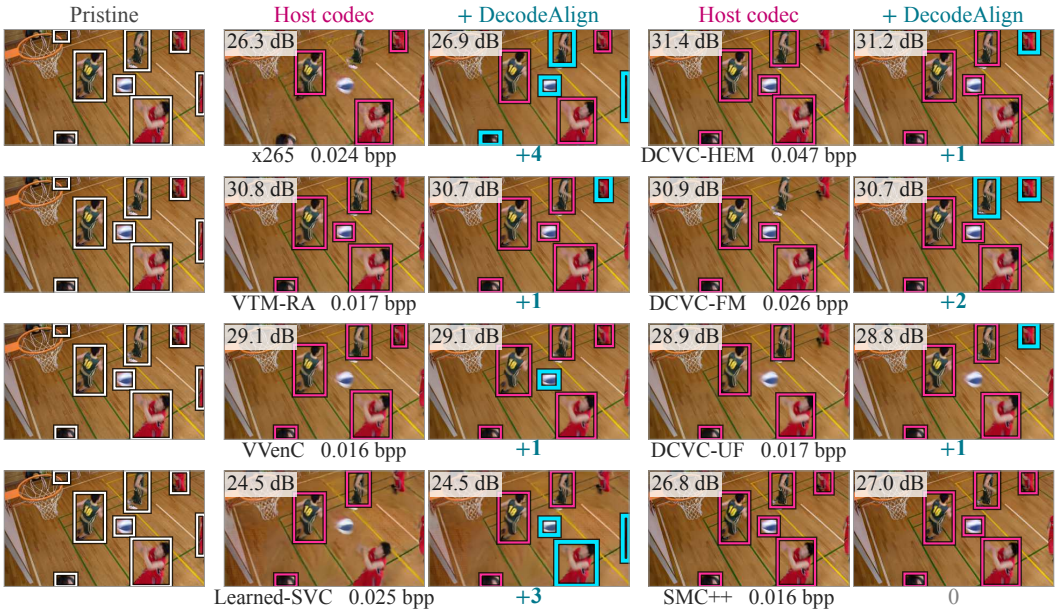


Figure 3: Detection results behind eight host codecs. The leftmost column shows the pristine frame, and the remaining columns are grouped in pairs, each showing a host-decoded frame and the corresponding frame after DecodeAlign. Boxes in each panel are the detector's correct detections on that frame, and cyan highlights the objects additionally detected relative to the host-decoded frame.

of 28.0% on VVenC and 45.6% on DCVC-FM among human-oriented conventional and learned codecs, respectively. For machine-oriented codecs (Learned-SVC and SMC++), DecodeAlign further improves performance, achieving an 18.8% BD-rate reduction on Learned-SVC, while the gain on SMC++ is relatively limited (1.7%) because its decoded representations are already close to those of the original content (Fig. 1).

Figure 3 visualizes the detection results of different host codecs before and after applying DecodeAlign. Compared with the pristine frames, host-decoded outputs exhibit missed detections for several objects. After applying DecodeAlign, some previously missed objects are successfully detected (cyan boxes), bringing the detection results closer to those of the pristine frames.

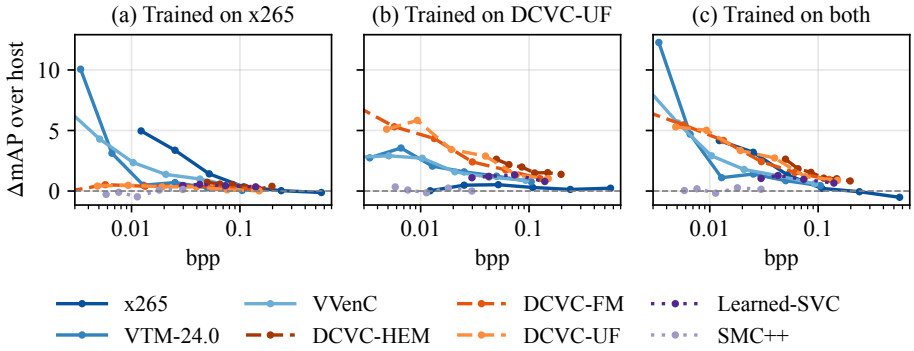


Figure 4: Generalization across compression distortion families. DecodeAlign is trained on three distortion pools and evaluated on host-decoded outputs from all host codecs. Each curve reports ΔmAP relative to the host-decoded output of the same codec at the identical bitrate, averaged over 14 SFU-HW sequences. Colors indicate host codecs, and line styles denote codec types: solid for conventional, dashed for learned, and dotted for machine-oriented codecs.

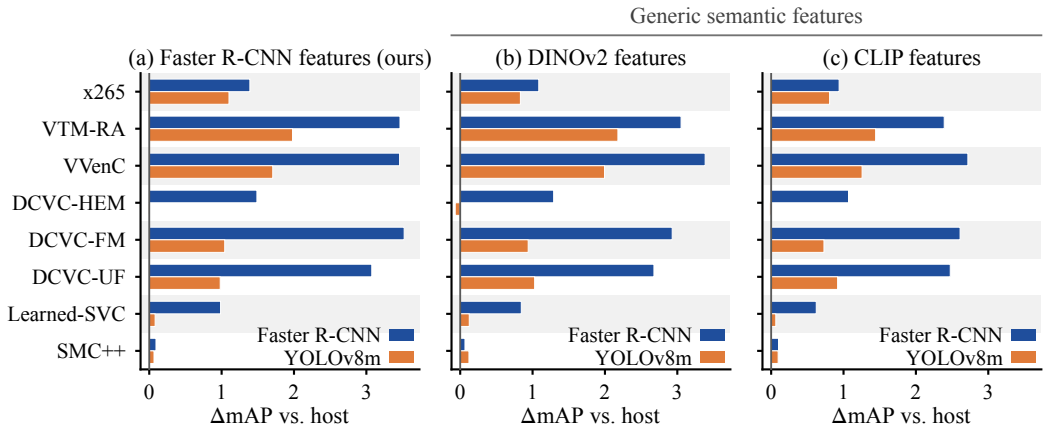


Figure 5: Generalization across alignment targets. DecodeAlign is trained with **(a)** Faster R-CNN features and **(b)–(c)** generic DINOv2/CLIP features. Bars show ΔmAP over host-decoded results, averaged over 14 SFU-HW sequences and evaluated on Faster R-CNN (blue) and YOLOv8m (orange) detectors.

4.3 Ablations

Compression distortion families. To investigate how DecodeAlign generalizes across compression distortions, we train three DecodeAlign models using only x265 decoded frames, only DCVC-UF decoded frames, and a mixture of both, respectively, while keeping all other training settings unchanged. Each model is evaluated on all host codecs. As shown in Fig. 4(a,b), single-distortion training generalizes effectively to unseen codecs within the same distortion family. Specifically, the x265-only model achieves peak gains of $+10.06$ and $+6.68$ mAP on unseen VTM-24.0 and VVenC, while the DCVC-UF-only model achieves $+7.00$ and $+2.62$ mAP on unseen DCVC-FM and DCVC-HEM. In contrast, cross-family generalization is substantially weaker. As shown in Fig. 4(c), mixed-distortion training covers both conventional and learned compression distortions, achieving positive gains at 42 of 45 tested rate points with a maximum gap of 0.96 mAP from specialist models. These results indicate that DecodeAlign generalizes according to observed distortion distributions rather than individual codec identities.

Alignment targets. We further investigate how the choice of alignment targets affects DecodeAlign. Specifically, we consider two types of alignment targets: task-specific features and

Table 1: Ablation of training losses. Rows denote the loss configurations used to train DecodeAlign, and columns denote host codecs and rate points for the input decoded frames. The *host* row reports the host-decoded mAP and PSNR, while other rows show changes relative to this reference averaged over 14 SFU-HW sequences. **Bold** indicates the best value per column, and gray entries indicate degradation relative to the host-decoded case.

Loss		mAP					PSNR (dB)				
		x265 host			DCVC-UF host		x265 host			DCVC-UF host	
	bpp	0.012	0.025	0.052	0.019	0.040	0.012	0.025	0.052	0.019	0.040
host		29.84	37.08	43.68	40.22	44.31	24.22	26.69	29.07	29.14	31.01
pixel		+0.11	−0.56	−1.10	−1.02	−0.41	**+0.61**	**+0.63**	**+0.61**	**+0.08**	**+0.11**
feature		+3.55	+2.60	+1.02	**+3.42**	**+2.76**	−0.31	−0.52	−0.72	−1.20	−1.26
pixel + feature		**+4.17**	**+3.20**	**+1.33**	+3.34	+2.73	+0.39	+0.37	+0.32	−0.17	−0.15

generic semantic features (Oquab et al., 2024; Radford et al., 2021). For task-specific alignment, we use intermediate features from an object detection model as supervision to examine whether detection-oriented representations can improve the machine readability of compressed video content and generalize across different detectors. As shown in Fig. 5(a), DecodeAlign aligned with Faster R-CNN features achieves the best performance on the corresponding detector, with an average gain of $+2.28$ mAP across eight host codecs. Meanwhile, it generalizes to the unseen YOLOv8m detector, achieving an average gain of $+0.92$ mAP.

For generic alignment, we adopt semantic features independent of specific machine models as alignment targets to examine whether model-agnostic representations can provide effective alignment signals. As shown in Fig. 5(b,c), generic semantic features also provide effective supervision. Specifically, DINOv2 achieves average gains of $+2.00$ and $+0.95$ mAP on Faster R-CNN and YOLOv8m, respectively. These results demonstrate that task-specific features provide stronger alignment signals, while generic semantic features offer an effective model-agnostic alignment strategy and support generalization across different detectors.

Training objectives. We further investigate how different training objectives affect DecodeAlign. Under the same training configuration, we train DecodeAlign with pixel loss, feature alignment loss, and their combination, and evaluate their effects on both mAP and PSNR. As shown in Table 1, the feature alignment loss is the primary driver of machine task performance, with feature-only training achieving gains of $+1.02$ to $+3.55$ mAP across different settings. In contrast, the pixel loss mainly improves reconstruction quality, where pixel-only training achieves the highest PSNR gains at all tested points but does not consistently improve task performance. Combining the two losses further enhances DecodeAlign by jointly improving machine task performance and reconstruction quality, achieving a better balance between machine-oriented inference and human-oriented reconstruction across different host codecs.

5 Conclusion

This paper revisits video coding for machines from a representation perspective and, through an analysis of adaptation constraints in compression systems, reveals that improving machine readability does not necessarily require embedding machine-oriented adaptation into the encoding process. Based on this observation, we propose DecodeAlign, a post-decoding representation alignment framework for compressed video machine tasks. Without modifying the original codecs, bitstream structures, or downstream task models, DecodeAlign improves machine task performance by enhancing the effectiveness of decoded content in the representation space of machine models. Extensive experiments demonstrate that DecodeAlign effectively improves machine readability across diverse host codecs and exhibits a generalization pattern governed by compression distortion distributions. In the future, we will further explore new paradigms for machine video coding from a representation perspective, investigating adaptive representation alignment mechanisms for diverse machine understanding tasks and compression environments, and extending machine readability optimization from the encoding stage to post-decoding representation processing.

AI USE STATEMENT

In this work, we used generative AI tools to assist with method implementation, experiment execution, and paper writing and editing. We have reviewed all AI-assisted work, and all reported results were verified against the raw experimental outputs. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Jukka I. Ahonen, Nam Le, Honglei Zhang, Antti Hallapuro, Francesco Cricri, Hamed Rezazadegan Tavakoli, Miska M. Hannuksela, and Esa Rahtu. NN-VVC: Versatile video coding boosted by self-supervisedly learned image coding for machines. In *IEEE International Symposium on Multimedia (ISM)*, 2023.

Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. In *International Conference on Learning Representations (ICLR)*, 2017.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations (ICLR)*, 2018.

Gisle Bjøntegaard. Calculation of average PSNR differences between RD-curves. Technical report, ITU-T SG16/Q6 VCEG, Doc. VCEG-M33, 2001.

Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J. Sullivan, and Jens-Rainer Ohm. Overview of the Versatile Video Coding (VVC) standard and its applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(10):3736–3764, 2021.

Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *European Conference on Computer Vision (ECCV)*, 2022.

Hyomin Choi, Ehsan Hosseini, Saeed Ranjbar Alvar, Robert A. Cohen, and Ivan V. Bajić. SFU-HW-Objects-v1: Object labelled dataset on raw video sequences. *Data in Brief*, 34:106701, 2021.

Ling-Yu Duan, Jiaying Liu, Wenhan Yang, Tiejun Huang, and Wen Gao. Video coding for machines: A paradigm of collaborative compression and intelligent analytics. *IEEE Transactions on Image Processing*, 29:8680–8695, 2020.

Hadi Hadizadeh and Ivan V. Bajić. Learned scalable video coding for humans and machines. *EURASIP Journal on Image and Video Processing*, 2024(1):41, 2024. doi: 10.1186/s13640-024-00657-w.

ISO/IEC JTC 1/SC 29/WG 04. Video coding for machines reference software (vcm-rs). MPEG Video Coding for Machines standardization activity, 2025.

Zhaoyang Jia, Bin Li, Jiahao Li, Wenxuan Xie, Linfeng Qi, Houqiang Li, and Yan Lu. Towards practical real-time neural video compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12543–12552, 2025.

Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLOv8. `https://github.com/ultralytics/ultralytics`, 2023.

JVET. VVC test model (VTM) reference software. `https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM`, 2024.

Jiahao Li, Bin Li, and Yan Lu. Deep contextual video compression. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Jiahao Li, Bin Li, and Yan Lu. Hybrid spatial-temporal entropy modelling for neural video compression. In *ACM International Conference on Multimedia (ACM MM)*, 2022.

Jiahao Li, Bin Li, and Yan Lu. Neural video compression with diverse contexts. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22616–22626, 2023.

Jiahao Li, Bin Li, and Yan Lu. Neural video compression with feature modulation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Jiahao Li, Wenxuan Xie, Zhaoyang Jia, Bin Li, Zongyu Guo, Xiaoyi Zhang, and Yan Lu. Ultra-fast neural video compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. DVC: An end-to-end deep video compression framework. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

MulticoreWare. x265 HEVC encoder. `https://www.videolan.org/developers/x265.html`, 2023.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021.

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*, 22(12):1649–1668, 2012.

Yuxiao Sun, Yao Zhao, Meiqin Liu, Chao Yao, and Weisi Lin. Embedding compression distortion in video coding for machines. In *IEEE International Conference on Multimedia and Expo (ICME)*, 2025.

Yuxiao Sun, Meiqin Liu, Chao Yao, Qi Tang, Jian Jin, Weisi Lin, Frederic Dufaux, and Yao Zhao. Symmetric entropy-constrained video coding for machines. *IEEE Transactions on Image Processing*, 2026a. doi: 10.1109/TIP.2026.3705185.

Yuxiao Sun, Yao Zhao, Meiqin Liu, Chao Yao, Huihui Bai, Chunyu Lin, and Weisi Lin. TransVFC: A transformable video feature compression framework for machines. *Pattern Recognition*, 171: 112091, 2026b.

Yuan Tian, Xiaoyue Ling, Cong Geng, Qiang Hu, Guo Lu, and Guangtao Zhai. Smc++: Masked learning of unsupervised video semantic compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

Adam Wieckowski, Jens Brandenburg, Tobias Hinz, Christian Bartnik, Valeri George, Gabriel Hege, Christian Helmrich, Anastasia Henkel, Christian Lehmann, Christian Stoffers, Ivan Zupancic, Benjamin Bross, and Detlev Marpe. VVenC: An open and optimized VVC encoder implementation. In *IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, 2021.

Thomas Wiegand, Gary J. Sullivan, Gisle Bjøntegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, 2003.

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

Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T. Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019.

Wenhan Yang, Haofeng Huang, Yueyu Hu, Ling-Yu Duan, and Jiaying Liu. Video coding for machines: Compact visual representation compression for intelligent collaborative analytics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Honglei Zhang, Jukka I. Ahonen, Nam Le, Ruiying Yang, and Francesco Cricri. Competitive learning for achieving content-specific filters in video coding for machines. In *IEEE International Conference on Image Processing (ICIP)*, 2024.