Pattern Recognition 171 (2026) 112091

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

TransVFC: A transformable video feature compression framework for machines

Yuxiao Sun [a,b], Yao Zhao [a,b], Meiqin Liu [a,b,*], Chao Yao [c], Huihui Bai [a,b], Chunyu Lin [a,b], Weisi Lin [d]

[a] Institute of Information Science, Beijing Jiaotong University, Beijing, 100044, China
[b] Visual Intelligence +X International Cooperation Joint Laboratory of MOE, Beijing, 100044, China
[c] School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing, 100083, China
[d] School of Computer Science and Engineering, Nanyang Technological University, 639798, Singapore

ARTICLE INFO

Keywords:
Neural video compression
Feature compression
Video coding for machines
Intermediate feature

ABSTRACT

Currently, an increasing number of video transmissions are focusing primarily on downstream machine vision tasks rather than on human vision. While the widely deployed human visual system (HVS)-oriented video coding standards such as H.265/HEVC and H.264/AVC are efficient, they are not the optimal approaches for video coding for machines (VCM) scenarios, leading to unnecessary bitrate expenditures. Academic and technical explorations within the VCM domain have led to the development of several strategies; however, conspicuous limitations remain in their adaptability to multitask scenarios. To address this challenge, we propose a Transformable Video Feature Compression (TransVFC) framework. It offers a compress-then-transfer solution and includes a video feature codec and feature space transform (FST) modules. In particular, the temporal redundancy of video features is squeezed by the codec through a scheme-based inter-prediction module. Then, the codec implements perception-guided conditional coding to minimize spatial redundancy and help the reconstructed features align with the downstream machine perception process. Subsequently, the reconstructed features are transferred to new feature spaces for diverse downstream tasks by the FST modules. To accommodate a new downstream task, only one lightweight FST module needs to be trained, avoiding the need to retrain and redeploy the upstream codec and downstream task networks. Experiments show that TransVFC achieves high rate-task performance for diverse tasks at different granularities. We expect our work to provide valuable insights for video feature compression in multitask scenarios. The codes are available at https://github.com/Ws-Syx/TransVFC.

1. Introduction

Digital videos play a crucial role in our lives, constituting a significant portion of the information consumed daily. For videos aimed at the human visual system (HVS), such as movies and short clips, preserving visual details perceptible to humans during the compression process is essential. Moreover, videos collected for machine vision tasks, such as surveillance [1] and facial recognition [2], do not require all of their visual details to be preserved [3,4]. Recently, neural video compression frameworks for the HVS have evolved significantly and now offer excellent video compression performance [5–7]. However, a comprehensive exploration of neural-based video coding for machines (VCM) remains nascent.

HVS-oriented video codecs, such as H.265/HEVC [9] and H.264/AVC [8], are frequently employed to compress videos for downstream analysis, as shown in Fig. 1(a). However, these approaches encounter two limitations in VCM scenarios. First, these compression frameworks focus on minimizing pixel-domain and HVS-related distortion, such as the peak signal-to-noise ratio (PSNR) and the multi-scale structural similarity index measure (MS-SSIM), rather than meeting the specific needs of machine vision applications, which is a suboptimal approach for machine vision. Second, machine vision tasks usually require only a subset of image content [3,15]. For example, indiscriminately transmitting the background of an image for the downstream image classification task leads to bitrate waste. More tailored approaches are needed in machine-centric scenarios.

* Corresponding author.
E-mail addresses: yuxiaosun@bjtu.edu.cn (Y. Sun), yzhao@bjtu.edu.cn (Y. Zhao), mqliu@bjtu.edu.cn (M. Liu), yaochao@ustb.edu.cn (C. Yao), hhbai@bjtu.edu.cn (H. Bai), cylin@bjtu.edu.cn (C. Lin), wslin@ntu.edu.sg (W. Lin).

https://doi.org/10.1016/j.patcog.2025.112091
Received 17 February 2025; Received in revised form 18 May 2025; Accepted 5 July 2025
Available online 7 July 2025

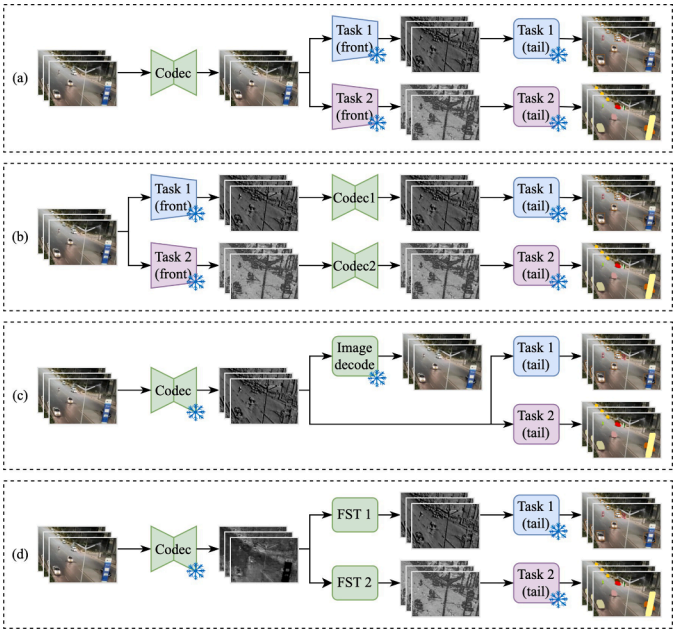

Fig. 1. A comparison among different pipelines in VCM scenarios. "Task(front)" represents the shallow layers of the downstream task network, "Task(tail)" denotes the rest of the downstream task network, and the snowflake symbol represents "module weights are frozen". (a) Videos are compressed by a hybrid or neural-based codec [8–11] and analyzed by downstream networks. (b) Intermediate features are extracted by the shallow layers of the task network and compressed by a specific-optimized video feature codec [12,13]. (c) The intermediate features for frame reconstruction are used to perform machine vision tasks, and the whole downstream task network is optimized [14]. (d) Our framework uses a video feature codec for continuous feature transmission, then transfers the reconstructed features to various downstream tasks via lightweight feature space transform (FST) modules.

Some studies have delved into the analyze-then-compress (ATC) paradigm to solve the above problems. The paradigm begins by extracting features from images, followed by feature compression for specific downstream tasks [16]. To enhance the versatility, some studies [3] focus on mining the generalization of intermediate features across various downstream tasks. Nonetheless, the above advancements only cater to intra-compression, and do not address the temporal redundancy in continuous features. For video feature compression, one strategy [12,13] entails optimizing a video feature codec by the specific downstream loss, as shown in Fig. 1(b). Alternatively, another strategy [14] focuses on freezing the codec while fine-tuning the entire downstream task network, as described in Fig. 1(c). However, these approaches require retraining and redeploying either the upstream codec or the downstream machine vision networks to accommodate new downstream tasks, thus costing more computational resources and limiting their scalability in real-world applications.

To achieve better scalability and versatility in multitask scenarios, we propose a Transformable Video Feature Compression (TransVFC) framework that offers a compress-then-transfer solution. As illustrated in Fig. 1(d), our proposed framework contains an innovative neural-based video feature codec and diverse lightweight feature space transform (FST) modules. Specifically, the codec employs a scheme-based inter-prediction module to squeeze the temporal redundancy of video features and form a coarse compensated feature. Furthermore, it conducts perception-guided conditional coding for fine reconstruction and helps the reconstructed feature align with the downstream machine perception process. Subsequently, the reconstructed features are transferred to other feature spaces of diverse downstream machine vision tasks via the FST modules. For any new downstream task, only one lightweight FST module must be trained instead of retraining and redeploying the upstream codec or the networks of the downstream tasks. Experiments are conducted on three machine vision tasks at different granularities. The results demonstrate that the proposed TransVFC outperforms the state-of-the-art (SOTA) neural codecs on all downstream tasks and outperforms VTM-23.1 [17] on video instance segmentation and object detection. The contributions of this study are as follows.

- We propose a novel Transformable Video Feature Compression (TransVFC) framework. It comprises two components: a video feature codec and diverse feature space transform modules, offering a scalable and deployable VCM solution.
- We introduce an innovative neural-based video feature codec to squeeze the redundancy encountered in the feature domain. It includes a scheme-based inter-prediction module and a perception-guided conditional coding module.
- We design a lightweight feature space transform module that transfers intermediate features to diverse downstream tasks in a highly scalable way. The experimental results validate the scalability and effectiveness of TransVFC across multiple downstream machine vision tasks of varying granularities.

2. Related works

2.1. Neural video compression

Most of the existing neural video compression methods follow the motion-then-residual paradigm [5,6,18] and mainly include inter-prediction and residual (i.e. context) compression. Lu *et al.* [19] proposed the first end-to-end video compression framework called DVC, which uses optical flow for inter-prediction and replaces the DCT transform with an autoencoder. Lu *et al.* [20] proposed FVC to convert videos from the pixel domain to the feature domain and use deformable convolution for motion estimation and motion compensation in the feature domain. In traditional hybrid coding frameworks and above neural video compression frameworks, residuals are calculated based on mathematical subtraction. This method is simple and easy to implement, but it may not be the optimal solution for compression. Li *et al.* [21] redefined the concept of residual and transform subtraction-based residual into conditional residual calculated by the neural codec, named DCVC. Sheng *et al.* [22] proposed the DCVC-TCM with a multi-scale conditional residual, which enhances the ability to remove inter-frame temporal redundancy. Overall, the existing neural video compression methods achieve improved compression efficiency from various perspectives such as inter-prediction, residual compression, and entropy models. Many NVC methods (e.g., the DCVC series [5–7]) demonstrate formidable compression capabilities.

2.2. Neural-based video coding for machines

The exploration of neural-based VCM reveals two pivotal paradigms: the compress-then-analyze (CTA) paradigm and the analyze-then-compress (ATC) paradigm.

2.2.1. Image and video compression in the CTA paradigm

With the surge in machine vision applications, video compression frameworks are re-envisioned to better cater to downstream machine vision tasks. Some methods [23,24] bridge the image codec and downstream task networks, and then integrate the loss function of the downstream task to guide the optimization process of the compression network, thus tailor-fitting it for attaining enhanced performance on specific tasks. In addition, Tian *et al.* [10,11] proposed maintaining semantic similarities through an additional bitstream, which improves the performance on multiple downstream tasks in an unsupervised way. Furthermore, the introduction of plug-and-play preprocessing modules [15] represents a significant improvement. These approaches achieve better

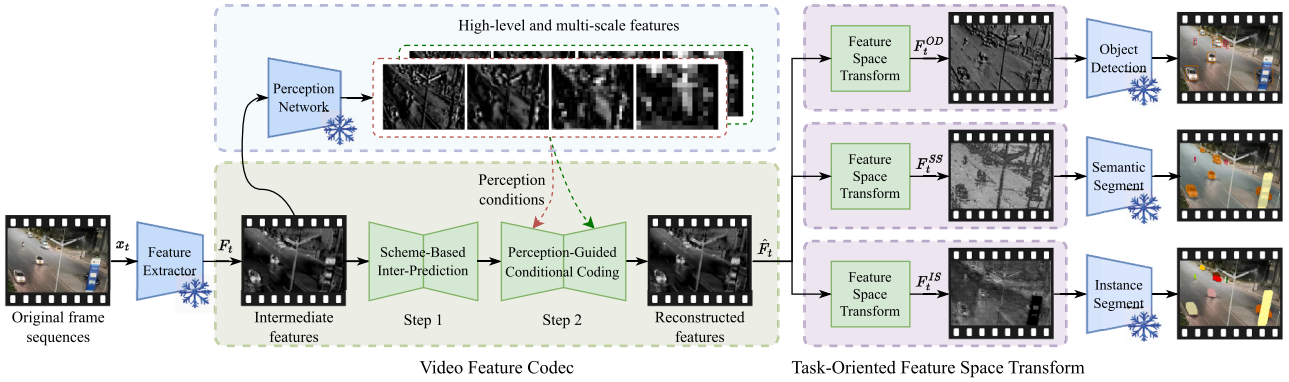

Fig. 2. Overview of the proposed TransVFC framework. F_t^{OD}, F_t^{SS}, and F_t^{IS} denote features for downstream object detection, semantic segmentation, and instance segmentation tasks, respectively. The TransVFC framework employs a compress-then-transfer process. Specifically, the codec conducts scheme-based inter-prediction to form a coarse compensated feature and then performs perception-guided conditional coding for fine reconstruction. The reconstructed features are subsequently transferred to other feature spaces of diverse downstream machine vision tasks via the FST modules. Notably, each downstream task corresponds to a distinct FST module.

downstream performance by enhancing important regions and filtering useless details for downstream analysis. Moreover, VCM is a sub-task of video coding for humans and machines (VCHM). Some studies [3,4] proposed to modify the decoder and use features that are originally dedicated to fully reconstructing images for downstream video analysis.

2.2.2. Feature compression in ATC paradigm

Intermediate feature compression is a widely studied VCM method under the ATC paradigm. Intermediate features contain more general information about images than high-level features do and offer the potential for conducting multitask analysis. Moreover, they preserve the original spatial structure, which enables more effective redundancy removal through neural networks. Unlike shallow features, intermediate features undergo a preliminary extraction process, where irrelevant information is filtered out for machine vision tasks, making it easier to compress. In image feature compression, some approaches adopt traditional hybrid codec [25] or variational autoencoder (VAE)-based networks that are optimized by feature distortion and specific task losses [26] for intra-compression. Moreover, some methods [27,28] change the compressed object from a single intermediate feature to multi-scale features and compress them into a joint bit stream. In the field of video features compression, Misra *et al.* [12] introduced an end-to-end feature compression network. It employs a simple ResBlock-based [29] bidirectional interpolation in the feature domain, and the entire framework is optimized for specific downstream tasks. Sheng *et al.* [14] proposed a framework that conducts pixel-feature-domain inter-compression and supports multiple downstream tasks by freezing the upstream codec and optimizing the downstream networks. However, a limitation is encountered when retraining and redeploying the upstream feature codec or the whole downstream task networks in practical applications. In light of the above challenge, there is a growing need for adaptable and scalable VCM solutions.

3. Methodology

The pipeline of the proposed Transformable Video Feature Compression (TransVFC) framework is shown in Fig. 2. It contains two main components: a neural-based video feature codec and diverse feature space transform (FST) modules. Inspired by [12,20], the intermediate features are extracted by the $res2$ layers of the ResNet50 backbone in Faster R-CNN [30]; then, the 256D features are converted into a 64D representation to squeeze their channel redundancy. The video feature codec follows the motion-then-residual paradigm; it employs the scheme-based inter-prediction module to obtain a coarse motion-compensated feature

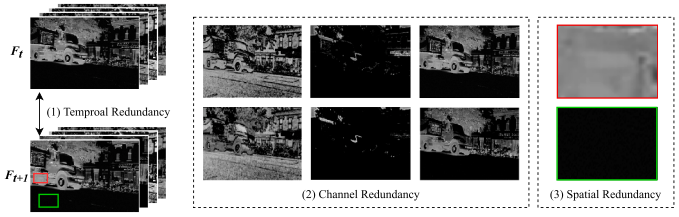

Fig. 3. Temporal, channel, and spatial redundancies among video features. The above redundancies need to be squeezed by the video feature codec.

and then uses the perception-guided conditional coding module for fine feature reconstruction. Afterward, the FST modules transfer the reconstructed intermediate feature $\hat{F}_t$ to different feature spaces, making them suitable for various downstream machine vision tasks. Notably, each downstream task is associated with a dedicated FST module.

3.1. Scheme-based inter-prediction

Temporal redundancy exists among repeated spatial structures, thus highlighting the need for redundancy removal by inter-prediction techniques, as shown in Fig. 3. For conducting inter-prediction in the feature domain, the deformable-convolution-based approach [20] focuses on finding the optimal reference region and recombining existing feature values. To better address complex motion, we depart from this referencing-and-recombination method. Instead, we propose a scheme-based inter-prediction module. It generates a variety of potential motion schemes from the reference frame and selectively combines them to obtain the compensated feature.

In the encoder, the motion estimation module performs a four-step sampling procedure for motion analysis across three distinct scales. The motion representation m_t contains both the global trends and the high-frequency details of motion, as shown in Fig. 4(b). Afterward, the motion encoder compresses m_t into a compact latent representation z with dimensions of $(H/16, W/16, 64)$. Subsequently, the latent representation z is quantized into $\hat{z}$ for entropy coding and transmission. In the decoder, the motion combination matrix $\hat{m}_t$ is reconstructed by the motion decoder module from $\hat{z}$. Leveraging the channel-wise computation by the depthwise separable convolution [31], the motion compensation module generates diverse possible motion schemes based on the reference frame f_{ref}. Then, referring to the motion representation $\hat{m}_t$, schemes are judiciously selected and combined to form the compensated feature $\tilde{f}_t$.

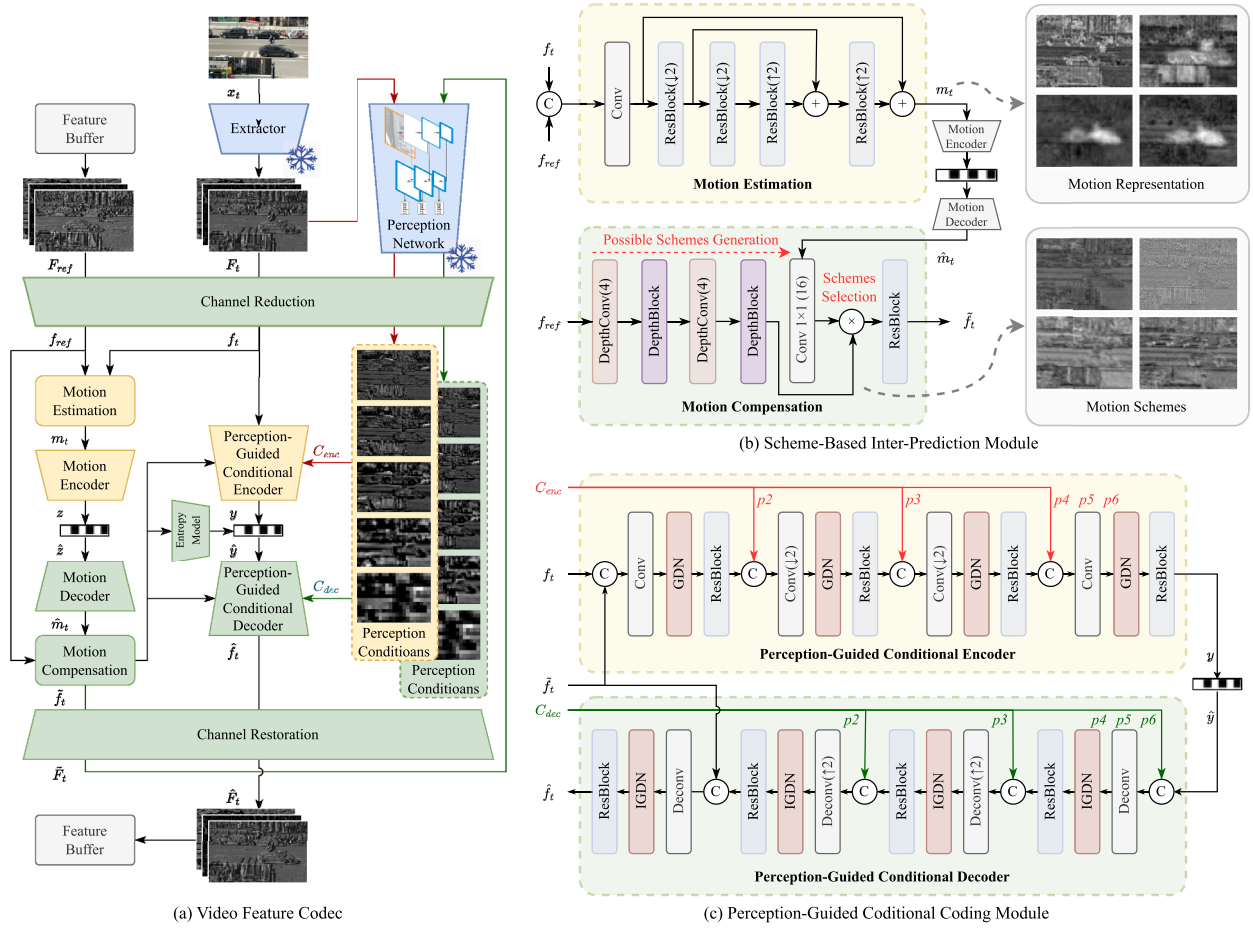

Fig. 4. (a) The overall structure of the proposed neural-based video feature codec. It contains 3 main stages: channel reduction/restoration, scheme-based inter-prediction, and perception-guided conditional coding. The green modules are located on both the encoder and decoder sides, whereas the yellow modules are only used on the encoder side. (b) The structure of the scheme-based inter-prediction module, including a motion estimation module, a motion compensation module, a motion encoder, and a motion decoder. *DepthConv*(*n*) represents a depthwise separable convolution layer with the number of channels increased by *n* times. The structure of *DepthBlock* is similar to *ResBlock* but replaces the convolution layers with the depthwise separable convolution layers. (c) The structure of the perception-guided conditional encoder and decoder. High-level and multi-scale features C_{enc} and C_{dec} are inferred from the Perception Network and used as conditions during the residual compression and reconstruction phases, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

More analyses and visualizations are shown in Section IV. The whole scheme-based inter-prediction process is described as follows:

$$\tilde{f}_t = MC(f_{ref}, D_m(\lfloor \mathcal{E}_m(ME(f_t, f_{ref})) \rceil)) \tag{1}$$

where $ME(\cdot)$ denotes motion estimation, $MC(\cdot)$ denotes motion compensation. $\mathcal{E}_m(\cdot)$ and $D_m(\cdot)$ denote motion encoder and decoder, respectively. $\lfloor \cdot \rceil$ denotes the quantization operation.

3.2. Perception-guided conditional coding

The compensated feature $\tilde{f}_t$ is obtained from the previous scheme-based inter-prediction. However, there is a gap in content detail between $\tilde{f}_t$ and f_t, making it essential to complete the content details using the residual. We employ conditional coding to compress the residual in the feature domain. Since different machine vision tasks share common perceptions [32], we further introduce multi-scale high-level features in Faster R-CNN [30] as perception conditions to help the reconstructed features better align with the downstream machine perception. Furthermore, the perception conditions offer TransVFC more prior knowledge during residual compression and reconstruction phases for achieving lower entropy and better spatial redundancy removal, as follows:

$$H(f - \tilde{f}) > H(f|\tilde{f}) > H(f|\tilde{f}, C_{enc}, C_{dec}) \tag{2}$$

where $H(\cdot)$ represents entropy, $\tilde{f}$ denotes the compensated feature, C_{enc} and C_{dec} denote the perception conditions for encoding and decoding, respectively.

As depicted in Fig. 4(c), the perception-guided conditional encoder comprises a four-step feature extraction process that compresses residuals into a compact and flat representation, while the decoder mirrors this structure symmetrically to reconstruct the intermediate features. Multi-scale perception conditions are strategically inserted into positions that align with their corresponding spatial resolutions (specifically at the 1/4, 1/8, and 1/16 scales), serving as conditions for both encoding and decoding to enhance the overall performance of the codec. In particular, the encoding perception conditions $C_{enc} = \{p2, p3, p4, p5, p6\}$ are inferred from the original intermediate feature F_t via feature pyramid network (FPN) backbone of Faster R-CNN. Due to the invisibility of F_t during decoding, the decoding perception condition C_{dec} is calculated from the compensated feature $\tilde{F}_t$. The whole process of perception-guided conditional coding is described as follows:

$$\hat{f}_t = D_c(\lfloor \mathcal{E}_c(f_t|C_{enc}, \tilde{f}_t) \rceil | C_{dec}, \tilde{f}_t) \tag{3}$$

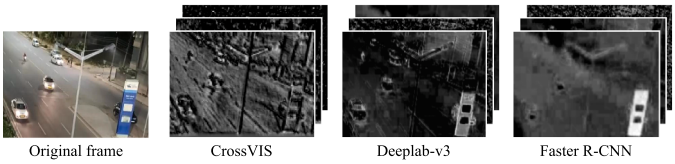

Fig. 5. Visualizations of the first three channels of the intermediate features across various machine vision networks. There are similar spatial structures but distinct feature patterns and textures among different intermediate features.

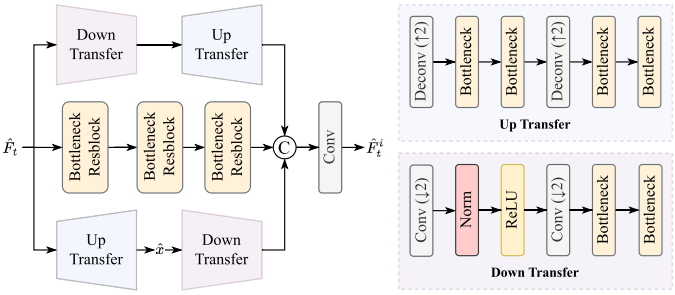

Fig. 6. The structure of feature space transform module. The reconstructed intermediate feature $\hat{F}_t$ is transferred to $\hat{F}_t^i$ in a new feature space for the i-th downstream machine vision task.

where $\mathcal{E}_c(\cdot)$ and $D_c(\cdot)$ denote perception-guided conditional encoder and decoder, respectively.

The latent representation y of the residual with dimensions of $(H/16, W/16, 96)$ is entropy-encoded by an entropy model similar to DCVC-TCM [22]. With respect to computational efficiency, autoregressive or other complex techniques are not employed in TransVFC. Additionally, a detailed explanation of how the perception-guided conditional coding removes redundancy is given in Section IV.

3.3. Task-oriented feature space transform

Owing to the gap between the intermediate features of different neural networks, as shown in Fig. 5, the decoded video features cannot be directly used in diverse downstream tasks. Some studies [3,14] have already shown that intermediate features have the potential to be converted and used in other machine vision tasks. Inspired by [3], we design the multi-scale feature space transform (FST) module that maps the reconstructed features to other feature spaces for different downstream tasks. Different from the existing neural-based VCM strategies [12, 14], our approach does not fine-tune the upstream feature codec and downstream task networks. Instead, it only requires a single lightweight FST module to be trained for a specific downstream task.

As shown in Fig. 6, the FST module is structured with three branches: the up-then-down branch, which coarsely reconstructs the current frame $\hat{x}_t$ for content preservation in pixel domain; the bottleneck-resblock [29] branch, facilitating feature migration at the original shape; and the down-then-up branch, focusing on global information extraction. Additionally, a convolution layer is used to align the channel and spatial shape of the output features to the specific downstream task. The process of feature space transform is described below:

$$\hat{F}_t^i = FST^i(\hat{F}_t) \tag{4}$$

where $FST^i(\cdot)$ denotes the i-th FST module, $\hat{F}_t$ denotes the intermediate feature reconstructed by the video feature codec, and $\hat{F}_t^i$ denotes the transferred feature that is suitable for the i-th downstream task.

3.4. Optimization

Since strong correlations between HVS-oriented pixel-domain metrics and machine vision performance are lacking, as mentioned in Fig. 7.

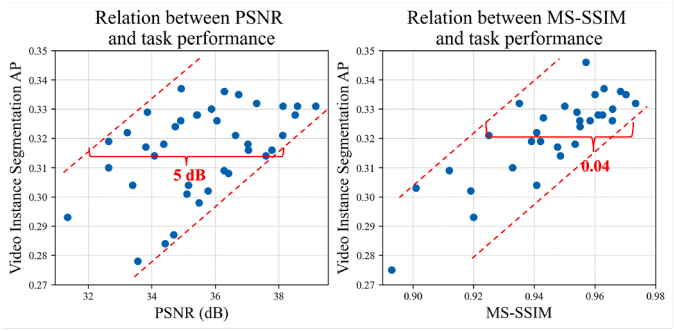

Fig. 7. The correlations between the pixel-domain and HVS-related distortion metric (the PSNR and MS-SSIM) and the downstream machine vision performance (e.g., the average precision of video instance segmentation) are weak. Optimizing the compression network for minimizing pixel-domain HVS distortions is not the best approach for the VCM scenario. The degraded videos are collected from traditional hybrid codecs and neural-based codecs [6,7,9,20–22]. Particularly, the Youtube-VIS 2019 dataset and the CrossVIS [33] model are used.

The optimization of our proposed TransVFC is mainly conducted in the feature domain and divided into two stages.

3.4.1. Optimization of the video feature codec

Rate-distortion optimization (RDO) is performed for the proposed video feature codec in the feature domain. The loss function $\mathcal{L}_{codec}$ is defined as follows:

$$\mathcal{L}_{codec} = \lambda_R(R_r + R_m) + \lambda_f D_f + \lambda_c D_c + \lambda_p D_p \tag{5}$$

where λ_R, λ_f, λ_c, and λ_p represent coefficients for balancing different loss terms. R_r and R_m represent the bitrates of the residual and motion representation, respectively. D_f denotes the mean square error (MSE) between the original intermediate feature F_t and the reconstructed feature $\hat{F}_t$. D_c denotes the MSE between F_t and compensated feature $\bar{F}_t$. D_p denotes the distortion in the perception space and is defined as follows:

$$D_p = \frac{1}{N}\sum_{j=1}^{N=5} MSE(PN_j(F_t), PN_j(\hat{F}_t)) \tag{6}$$

where $PN(\cdot)$ denotes the perception network contained in TransVFC, and N denotes the number of high-level output features derived from the perception network.

3.4.2. Optimization of feature space transform module

Since the FST module mainly transforms the reconstructed intermediate features to other spaces for downstream networks. It is optimized for minimizing the downstream task loss and feature distortion in the new feature space. All other modules are frozen in this training stage. The total loss of the FST modules is defined as follows:

$$\mathcal{L}_{FST} = \lambda_{task}\mathcal{L}_{task} + \lambda_x D_x + \lambda_{mid} D_{mid} + \lambda_{high} D_{high} \tag{7}$$

where λ_{task}, λ_x, λ_{mid} and λ_{high} represent coefficients for balancing different loss terms. $\mathcal{L}_{task}$ denotes the loss of downstream task network. D_{mid} denotes the MSE between the transferred feature $\hat{F}_t^i$ and the original feature F_t^i for the i-th downstream task, D_x denotes the MSE between the coarsely reconstructed frame $\hat{x}_t$ and the original frame x_t, and the definition of D_{high} is defined as follows:

$$D_{high} = \frac{1}{N}\sum_{j=1}^{N} MSE(TASK_j^i(F_t^i), TASK_j^i(\hat{F}_t^i)) \tag{8}$$

where $TASK^i(\cdot)$ represents the backbone of the i-th downstream task network and N denotes the number of output high-level features from the i-th downstream backbone.

Table 1
Training strategy for video feature codec.

Stages	$\mathcal{L}_{codec}$	Learning rate
1	$\frac{1}{2}\lambda_R R_y + \lambda_f D_f + \lambda_c D_c + \lambda_p D_p$	1×10^{-4}
2	$\frac{1}{2}\lambda_R (R_y + R_z) + \lambda_f D_f + \lambda_c D_c + \lambda_p D_p$	1×10^{-4}
3	$\lambda_R (R_y + R_z) + \lambda_f D_f + \lambda_c D_c + \lambda_p D_p$	1×10^{-4}
4	$\lambda_R (R_y + R_z) + \lambda_f D_f + \lambda_c D_c + \lambda_p D_p$	5×10^{-5}
5	$\lambda_R (R_y + R_z) + \lambda_f D_f + \lambda_c D_c + \lambda_p D_p$	1×10^{-5}

Table 2
Training hyperparameters λ for feature space transform module.

Downstream tasks	λ_{mid}	λ_{high}	λ_x	λ_{task}
Object detection	16	4	1024	10
Instance segmentation	8	64	1024	1
Semantic segmentation	16	64	1024	10

4. Experiment

4.1. Experimental settings

4.1.1. Downstream machine vision tasks

The performance of TransVFC is verified on three downstream tasks at different granularities. We employ the CrossVIS [33] framework for video instance segmentation, DeepLab-v3 [34] for semantic segmentation, and Faster R-CNN [30] for object detection. The parameters of all the downstream networks are frozen throughout the experiments.

4.1.2. Datasets

Experiments are conducted on the YoutubeVIS2019 (YTVIS2019) [33] and Video Scene Parsing in the Wild (VSPW) [35] datasets. The YTVIS2019 dataset is a large video dataset that includes 2883 videos with frame-level annotations of 40 categories for video instance segmentation. The resolutions of the videos range from 1080P to 360P, and the data preproces stages follow [33]. The VSPW dataset is a large video dataset that includes 3536 videos in 480P resolution across 231 scenarios. It has frame-level annotations of 124 categories for video semantic segmentation.

4.1.3. Compared methods

The proposed TransVFC framework is compared with traditional hybrid codecs VTM-23.1 (lowdelay-P) [17], HM-18.0 (lowdelay-P) [1] [36] and x265 (FFmpeg-4.2.7, zerolatency) [2] [37], and open-sourced neural video compression (NVC) frameworks, such as DCVC-DC [6], DCVC-HEM [7], DCVC-TCM [22], DCVC [21], and FVC [20]. For compared NVC methods, all available pre-trained models are evaluated across different metrics (PSNR, MS-SSIM, YUV), showcasing only the model with the highest rate-task performance. In addition, VCM-oriented video codec SMC++ [11] is used as a comparison method.

4.1.4. Implementation details

In the first stage, we optimize the video feature compression framework at different bitrates with $\lambda_R = 16, 32, 128, 256$, $\lambda_f = 16$, $\lambda_c = 0.1\lambda_f$, and $\lambda_p = 4$. The training strategy is shown in Table 1. The input features during training are cropped to 128×128. The neural-based video feature codec is optimized on the YTVIS2019-train.

[1] The command of VTM and HM is `./bin/TAppEncoderStatic -c ./cfg/encoder_lowdelay_P_main.cfg -i {input_path} -b {output_binary_path} -o {output_path} -wdt {width} -hgt {height} -q {QP} -fr {frame_rate} -InputChromaFormat=420 --IntraPeriod=12`

[2] The command of x265 is `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=12:verbose=1" out.mkv`

Table 3
BD-Rate (%) ↓ comparison. The anchor is VTM-23.1. **Bold** indicates the best results.

	Object detection	Semantic segmentation	Instance segmentation
VTM-23.1 (low-delay) [17]	0.00	0.00	0.00
HM-18.0 (low-delay) [36]	7.82	**-11.40**	5.60
x265 (zero-latency) [37]	−1.16	36.34	3.91
FVC (CVPR'20) [20]	97.15	130.03	368.56
DCVC (NerulPS'21) [21]	50.34	286.38	109.43
DCVC-TCM (TMM'22) [22]	7.84	204.46	32.69
DCVC-HEM (ACMMM'22) [7]	−3.92	183.80	46.34
DCVC-DC (CVPR'23) [6]	−4.53	145.53	26.56
SMC++ (arXiv'24) [11]	−4.28	74.61	−6.77
TransVFC (Ours)	**-15.21**	63.60	**-27.67**

In the second stage, different weights λ are used to train FST modules for each downstream task, as shown in Table 2. Owing to the impracticality of exhaustively tuning the λ weights under computational constraints, we determine them empirically. Inspired by the existing approach [34], we follow a simple rule: each loss term is scaled such that its magnitude and the gradient it contributes to the FST module are comparable to those of others terms. This strategy facilitates stable training and balanced learning across multiple objectives. The number of training iteration for the FST module is $100k$, and the learning rate is set to 1×10^{-5}. Notably, the FST modules for different downstream tasks are trained separately.

The implementation of TransVFC is based on PyTorch 1.9.0. The whole framework is optimized on a single NVIDIA RTX 3090 24GB with $batchsize = 4$.

4.2. Evaluation metrics

The number of bits per pixel (bpp) is used to represent bitrate cost, where a lower bpp value indicates a higher compression ratio. For downstream tasks, average precision (AP) ↑ is used to evaluate the performance of object detection and instance segmentation, referring to [30,33]. While the mean intersection over union (mIoU) ↑ is used to assess semantic segmentation performance, referring to [34]. The PSNR ↑ and MS-SSIM ↑ are employed to evaluate the quality of the reconstructed frames. The Bjøntegaard Delta Rate (BD-Rate) ↓ is used to quantify the overall rate-task performance. It reflects the percentage of bitrate change while achieving the same task performance. A lower BD-Rate represents greater bitrate savings. VTM-23.1 serves as the anchor for calculating the BD-Rate.

4.3. Rate-task performance

4.3.1. Object detection

The implementation of the Faster R-CNN [30] is based on Detectron2 [38], which is an extensively used and efficient framework for keypoint detection, object detection, and segmentation. The task performance across various bitrates is displayed in Fig. 8(a). In terms of rate-task performance, TransVFC achieves a 15.21 % bitrate reduction relative to VTM, as shown in Table 3. From a rate-time perspective, TransVFC achieves a better speed-performance balance than the other neural-based methods do. Visualization examples of the object detection results are shown in Fig. 9. TransVFC tends to produce fewer false detections under various bitrates and scenarios.

4.3.2. Semantic segmentation

We implement the DeepLab-v3 [34] via TorchVision-0.9.0. Following [34], mean intersection over Union (mIoU) is used to evaluate the semantic segmentation performance of the tested methods. As demonstrated in Table 3, TransVFC outperforms the best neural-based method, SMC++ [11], in terms of rate-task performance. Additionally, TransVFC achieves the best speed-performance balance among the

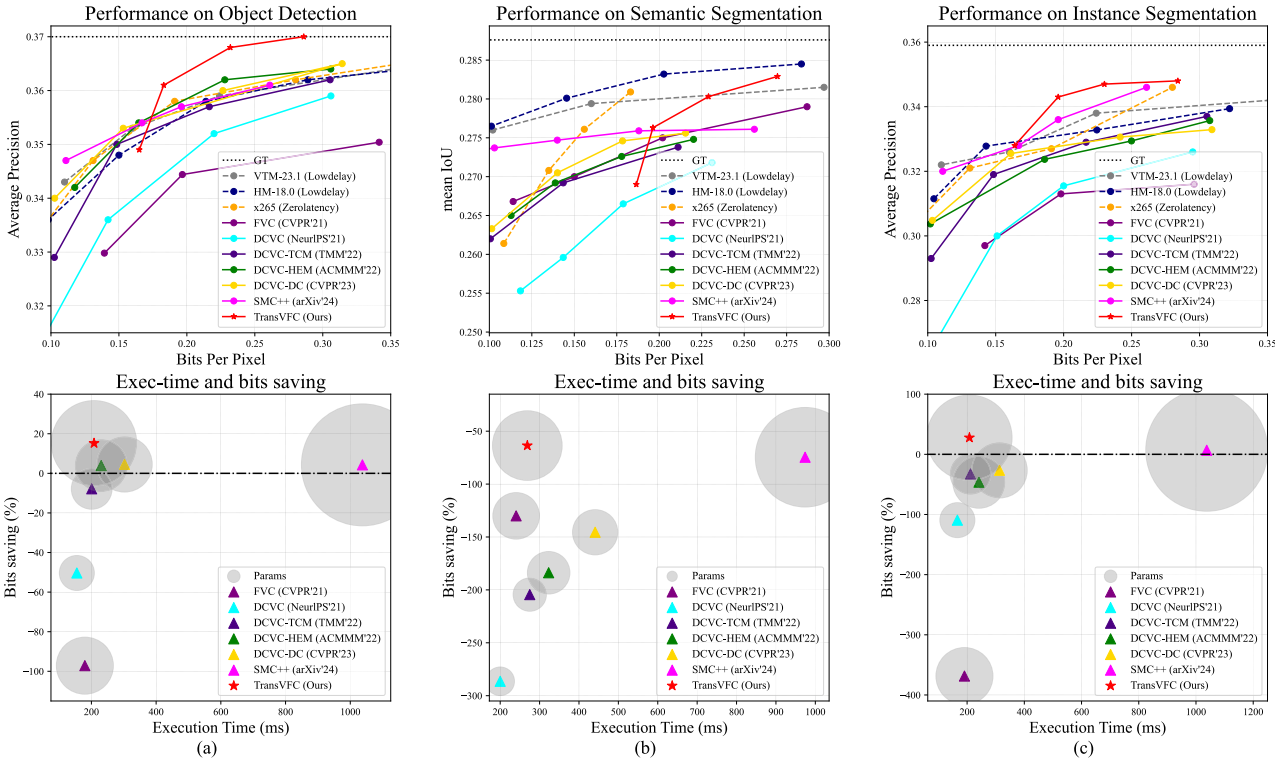

Fig. 8. Rate-task performance of all the compared methods (in the upper row) and execution times of neural-based methods (in the lower row) on object detection, semantic segmentation, and instance segmentation tasks. The execution time, including compression and downstream analysis, is evaluated with $batchsize = 1$ on a single NVIDIA RTX 3090 24GB, excluding the time of file I/O.

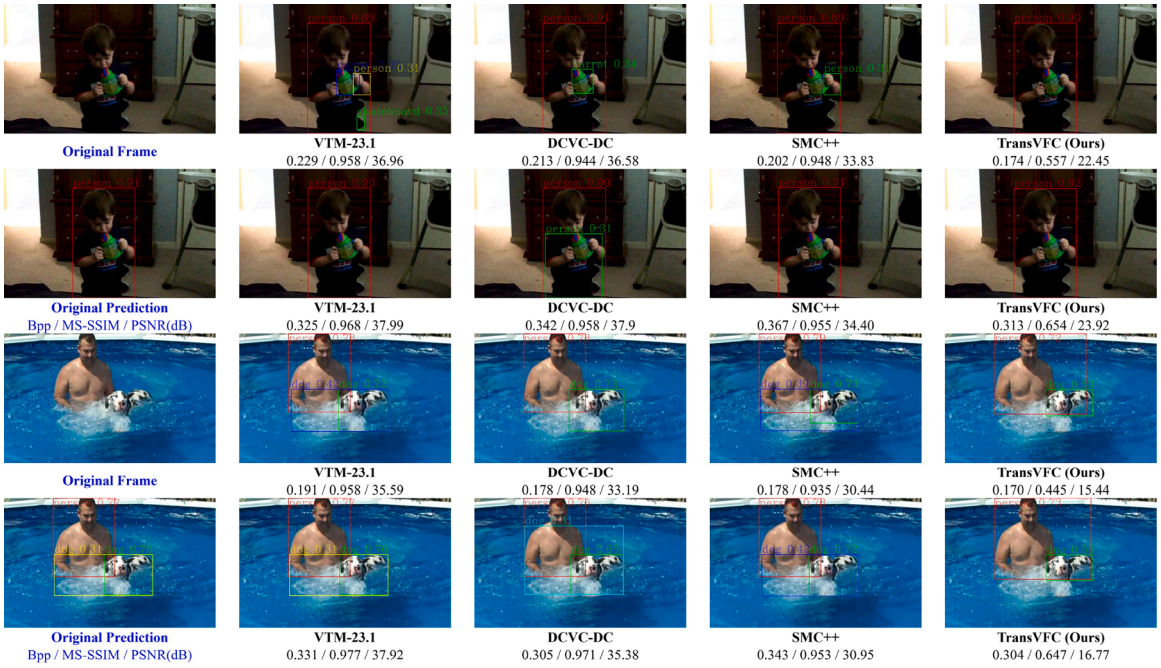

Fig. 9. Visualization of object detection results at different bitrates, along with the corresponding bpp, MS-SSIM, and PSNR values. The proposed TransVFC tends to produce fewer false detections. In contrast, other methods exhibit false positive detections, despite achieving high reconstruction quality.

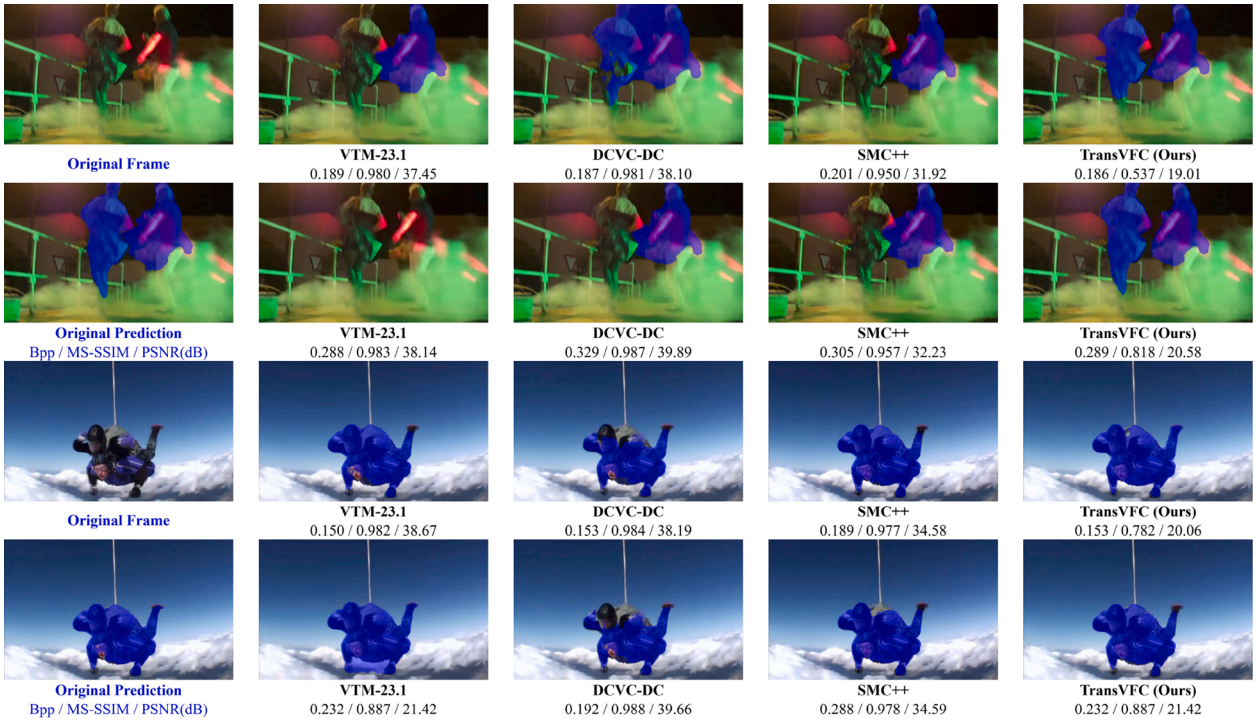

Fig. 10. Visualization of semantic segmentation, bpp, MS-SSIM, and PSNR at different bitrates. TransVFC better preserves object contours and performs segmentation more accurately under challenging conditions (e.g., fog, low-light, and intense motion) compared to other methods.

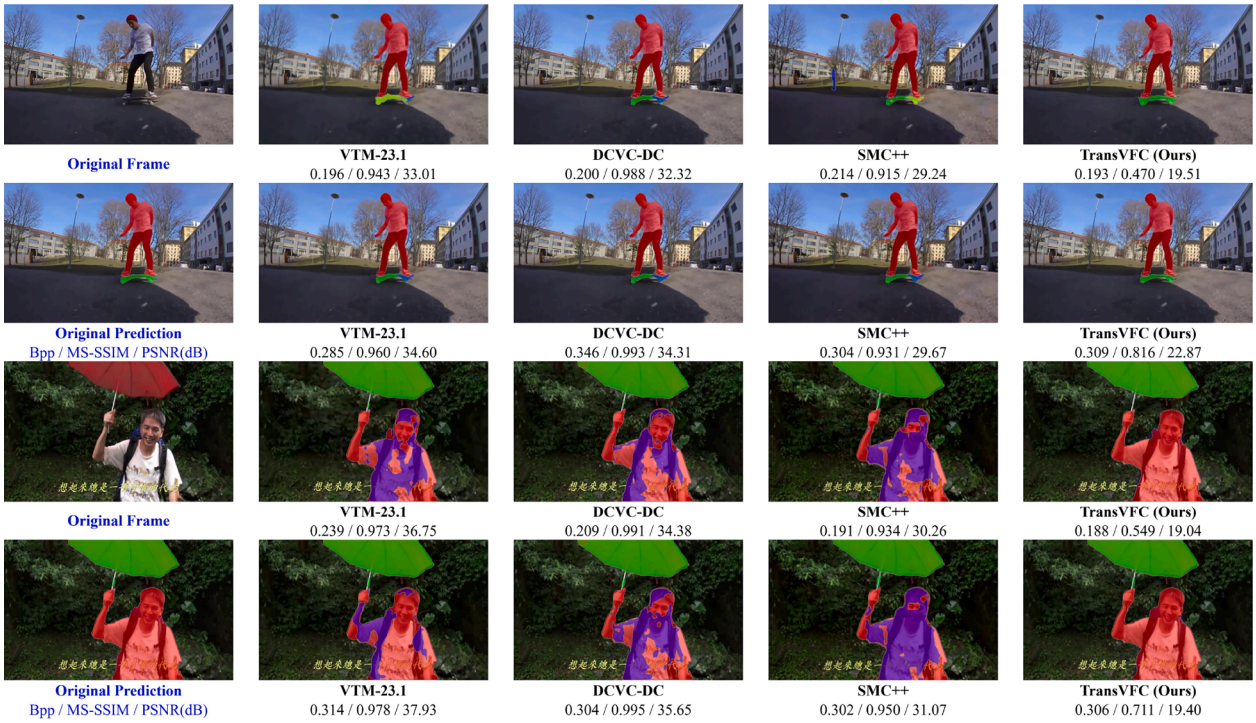

Fig. 11. Visualization of video instance segmentation, bpp, MS-SSIM, and PSNR at different bitrates. In videos with intense motion (in the first two rows) and tiny movement (in the last two rows), TransVFC maintains the original prediction and ensures consistency of the instance. The visualization shows that even if the reconstructed frames have high reconstruction quality, they may still underperform in downstream tasks.

tested neural-based methods, as shown in Fig. 8(b). Visualization of semantic segmentation is provided in Fig. 10, where TransVFC demonstrates better preservation of object contours and more accurate segmentation under challenging conditions, such as fog and low-light, compared to other methods.

4.3.3. *Instance segmentation*

CrossVIS [33] is implemented on its officially released code. As demonstrated in Table 3, in terms of rate-task performance, TransVFC achieves the highest compression ratio, achieving a 27.67 % bitrate reduction compared to VTM. In terms of execution speed, compared with

Table 4
Execution time (ms) on 720P frame, number of model parameters, and MACs per pixel of neural-based methods. Notably, TransVFC has 22.4 optimized parameters and 26.7M frozen parameters.

	Non-stream inference	With bitstream		Model params	MACs per pixel
		Encoding	Decoding		
FVC (CVPR'20) [20]	165.8	/	/	21.0M	/
DCVC (NerulPS'21) [21]	129.1	1818.9	4738.3	7.9M	1.09M
DCVC-TCM (TMM'22) [22]	197.6	232.8	121.4	10.7M	1.40M
DCVC-HEM (ACMMM'22) [7]	240.6	250.3	124.6	17.5M	1.58M
DCVC-DC (CVPR'23) [6]	347.5	285.5	243.8	19.8M	1.27M
SMC++ (arXiv'24) [11]	830.1	/	/	96.2M	
TransVFC (Ours)	191.2	234.5	122.5	49.1M	1.16M

the high-performance NVC method DCVC-DC [6], TransVFC has a 34 % faster execution speed, as shown in Fig. 8(c). As shown in Fig. 11, TransVFC produces better subjective segmentation results at different bitrates. Despite the high quality of reconstructed frames, the downstream task network CrossVIS struggles with maintaining the segmentation consistency of the main objects (e.g., the skateboard and the man holding an umbrella), often incorrectly segmenting them into multiple instances. In contrast, our framework better maintains the consistency of the instance and maximally retains the original segmentation results.

4.4. Analysis

4.4.1. Complexity of video features compression

We compare the execution time, number of parameters, and MACs of our proposed video feature codec with other neural-based compression methods [6,7,11,20–22], as shown in Table 4. Our proposed video feature codec consists of an optimized codec with 22.4M parameters and a frozen perception network with 26.7M parameters. The inference time reflects the computational complexity of all the neural-based modules on the GPU without including arithmetic coding. The encoding and decoding times include the arithmetic coding operation time but exclude file I/O time. Although our codec has more parameters than other neural compression methods, it has fewer MACs per pixel than high-performance codecs like DCVC-DC and DCVC-HEM. TransVFC has a better complexity-performance balance than other high-performance NVC approaches, with efficiency gains stemming from three aspects. First, the intermediate features have a 1/4 spatial size of the original image, which helps TransVFC use fewer convolutions and down/upsampling operations than neural video compression frameworks. Second, to improve encoding and decoding speed, TransVFC uses a simple entropy model including a mean-scale hyperprior module and a temporal prior module [22], which is better parallelized. Third, introducing depthwise convolution reduces the computational complexity of the model [6,31], resulting in lower MACs.

4.4.2. Complexity of feature space transform

The FST module in the TransVFC framework is lightweight, with a parameter size of 4.3M. It is significantly smaller than the networks used for downstream visual analysis (e.g., the CrossVIS-ResNet50 version has 37.4 M parameters), adding less additional training overhead. The execution time of FST under a 720P resolution is 11.7 ms, which accounts for only 3.3 % of the total time. The MACs per pixel of the FST module are 0.16M. The above results indicate that the FST module is highly effective during both the training and inference stages.

4.4.3. Visualization of the scheme-based inter-prediction module

A visualization of our proposed scheme-based inter-prediction module is shown in Fig. 12. This module generates potential pattern schemes and then combines them through motion representation. The motion representation captures motion information, including local edge movements (e.g., channels 0 and 8) and large-scale motion (e.g., the rapid movement of the vehicle in channels 4 and 6). Moreover, motion schemes illustrate the potential components of the compensated feature,

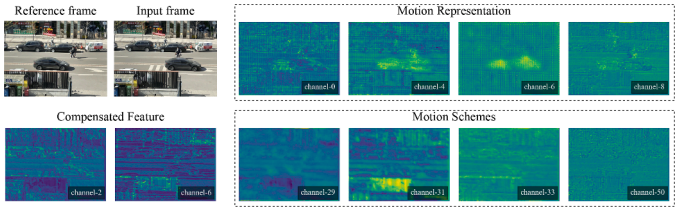

Fig. 12. Visualization of the compensated features, motion representations, and motion schemes.

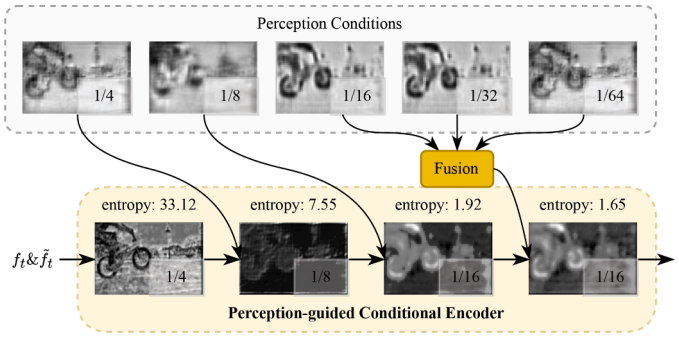

Fig. 13. The process of perception-guided conditional encoding. The feature is compressed into a compact representation with lower entropy with the help of perception conditions as prior knowledge.

incorporating various types of pattern schemes. These schemes are subsequently synthesized into the compensated feature under the guidance of the motion representation.

The compensated feature is a coarsely reconstructed feature obtained by inter-prediction and is similar to the current feature. The feature of the car is already moved to a new position in the compensated frame, as shown in Fig. 12. Since the compensated feature is merely a coarse-version feature of the current frame, the feature details are fulfilled through following perception-guided conditional coding process.

4.4.4. Relation between perception-guided conditional coding and spatial redundancy removal

Spatial redundancy is prevalent in intermediate features, as adjacent regions often exhibit similar textures and high-frequency details, leading to overlapping or repetitive information. The perception-guided conditional coding module addresses this redundancy through two key perspectives: First, as depicted in Fig. 13, the original features are downsampled multiple times and become smaller, more compact, and flatter. For a better understanding of the decrease in the amount of information, we take one frame as an example and calculate its entropy per pixel, as shown in Eq. 9. Fig. 13 shows that the entropy of the feature decreases during the encoding process; then, the feature is compressed into a latent representation with a lower entropy that is suitable for entropy coding and transmission. Second, the intermediate features to be compressed have significant spatial structural correlations and repetition with the perception condition. Since the perception condition (acting as prior knowledge) is already accessible on both the encoder and decoder sides, there is no need to redundantly transmit this content from the encoder to the decoder. Instead, the decoder can effectively reconstruct the original content using the available perception information, further squeezing the spatial redundancy and enhancing the efficiency of the coding process.

$$entropy = \sum_{i}^{N} p(f_i) log(p(f_i))/(H \times W) \tag{9}$$

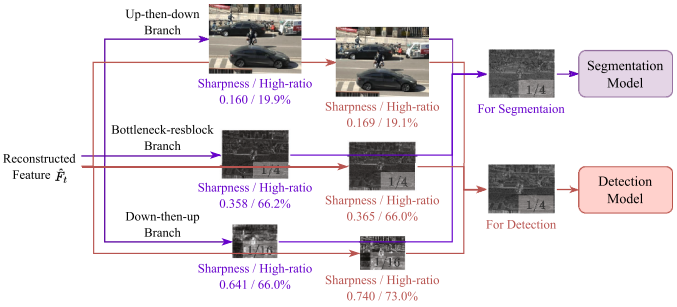

Fig. 14. Visualization of the intermediate results produced by each branch in the FST module for video instance segmentation and object detection tasks.

Fig. 15. Visualization of the upsampled results. The "up-then-down" branch can coarsely reconstruct the original content in the pixel domain, which helps the feature transfer process gain knowledge of pixel-domain content.

where N denotes the number of values in f_i, $p(\cdot)$ represents the probability of each value. H and W represent the height and width of the current frame, respectively. The feature f undergoes 8-bit quantization for probability statistics.

4.4.5. How the multi-branch architecture of FST works

To better understand the behavior of the three branches contained in the FST module, intermediate results produced by each branch are visualized in Fig. 14. Since each FST module is optimized for a specific task, different branches contribute differently depending on the given task. We use two metrics to evaluate the intermediate results: Sobel-based sharpness and the high-frequency ratio, defined as the proportion of FFT energy beyond the central 1/4 area. First, in terms of the high-frequency ratio, the down-then-up branch tends to produce global feature maps with richer content for the coarse-grained detection task. In contrast, the up-then-down and bottleneck-resblock branches generate richer low-level details for the fine-grained instance segmentation task. This aligns with the task needs: segmentation relies more on local details to support pixel-level classification and fine boundary extraction. Second, from the perspective of sharpness, all three branches tend to produce smoother intermediate results for downstream instance segmentation tasks. This also matches the task requirements, as segmentation prefers spatial consistency and avoids discontinuities or jagged edges. Furthermore, the up-then-down branch enhances the feature-domain transformation by coarsely reconstructing the original frame, making the FST module aware of pixel-domain content, as illustrated in Fig. 15.

4.4.6. Robustness analysis

The proposed TransVFC framework is evaluated on the video instance segmentation task under two types of degradation: low-light and Gaussian noise. For the low-light scenario, following the existing approach [39], we map the videos to the YCbCr space, reduce the Y channel (0.3×), and then convert them back to the RGB space. For the Gaussian noise scenario, we follow [40] and add Gaussian noise with $\sigma = 25$. As shown by the experimental results in Fig. 16, our method maintains SOTA rate-task performance under low-light conditions. When Gaussian noise is introduced in the pixel domain, the bitrate of all methods increases significantly. Notably, HVS-oriented VTM and DCVC-DC

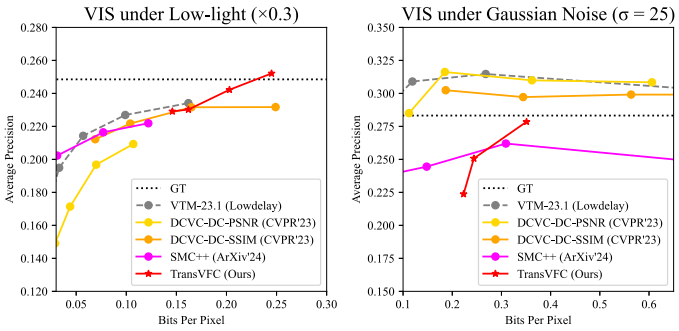

Fig. 16. Rate-task performance of video instance segmentation in low-light and Gaussian noise scenarios. The proposed TransVFC performs well under low-light scenarios but is less effective when facing pixel-domain noise.

Table 5
Ablation study on the proposed components in the video feature codec.

Models	Scheme-based inter-prediction	Perception condition	Perception loss	BD-Rate (%)↓
Model 1	✗	✓	✓	+11.37
Model 2	✓	✗	✓	+13.85
Model 3	✓	✓	✗	+36.92
Model 4	✓	✗	✗	+40.18
Model 5	✗	✗	✗	+46.71

can effectively suppress and filter such pixel-domain noise, and the reconstructed frames are mostly denoised during compression, resulting in task performance that even surpasses that of the uncompressed and noisy input. In contrast, our feature-compression-based method performs worse in this case, as pixel-domain noise harms the efficiency of feature compression. Addressing this limitation will be one of the directions of our future work.

4.5. Ablation study

Ablation experiments are conducted on the video instance segmentation task and the CrossVIS [33] model.

4.5.1. Ablation on video feature Codec

To verify the effectiveness of the proposed scheme-based inter-prediction, the proposed motion estimation and motion compensation modules are replaced with the existing deformable-convolution-based approach [20]; this model, which is represented as Model 1 in Table 5, results in an 11.37 % average bitrate increase. To verify the effectiveness of the perception conditions, we retain the framework structure but do not use C_{enc} and C_{dec} as conditions, named Model 2. It is demonstrated that reconstructing video features without the conditions causes a 13.85 % bitrate increase. Furthermore, the high-level perception loss D_p is removed in Model 3, resulting in a 36.92 % bitrate increase. The result of Model 4 indicates that introducing high-level perception in both the conditional coding process and the loss function can significantly increase the rate-task performance (40.18 % in total). Additionally, when both scheme-based inter-prediction and perception-guided conditional coding modules are removed (Model 5), simplifying the codec to a structure similar to FVC [20] with deformable-convolution-based inter-prediction and residual coding, the bitrate increases by 46.71 %.

4.5.2. Ablation on feature space transform module

To verify the function of each branch contained in the FST module, we remove the up-then-down branch (Model 6), the down-then-up branch (Model 7), and both branches (Model 8), as shown in Table 6. The experimental results demonstrate that each branch plays a significant role in the quality of the feature space transformation.

Table 6
Ablation study on the proposed FST module.

Models	Bottleneck-resblock	Down-then-up	Up-then-down	BD-Rate(%)↓
Model 6	✓	✓	✗	+2.83
Model 7	✓	✗	✓	+2.69
Model 8	✓	✗	✗	+5.69

Table 7
Ablation study on different approaches in ATC paradigm. "✓" means optimized and "✗" means frozen.

Models	Codec	Task	BD-Rate(%)↓	Optimzed params	GPU mem (GiB)	Training time per step (s)
Model 9	✓	✗	−6.33	22.4M	19.3(+12.9%)	1.430(+14.1%)
Model 10	✗	✓	−7.16	37.4M	18.6(+8.8%)	1.374(+9.7%)
Ours	✗	✗	0	4.3M	17.1	1.253

Additionally, we conduct an ablation study on the complexity of the FST module. We roughly double the number of parameters of the FST. The number of parameters in FST increases from 4.30M to 5.92M (+37%), and the MACs per pixel rise from 0.14M to 0.26M (+86%). The FST further reduces the BD-Rate by only 2.42%. We also roughly reduce the number of res-blocks. The number of parameters is reduced to 3.53M and MACs per pixel reduce to 0.12M. The BD-Rate increases by 5.60%. This finding shows that the current structure is appropriate since greater complexity results in only a limited BD-Rate reduction.

4.5.3. Comparison among different approaches in ATC paradigm

Other ATC-based VCM pipelines are implemented based on TransVFC, as detailed in Table 7. Referring to [12–14], we fine-tune either the upstream video feature codec (Model 9) or the downstream task network (Model 10) instead of the FST module. The experimental results indicate that training either the upstream or the downstream network leads to additional bitrate savings. However, this comes at the cost of needing to optimize more parameters, consuming more computational resources and training time. Benefiting from the FST module, our approach uses fewer computational resources and avoids redeploying the upstream video feature codec or downstream task networks, offering better scalability.

4.5.4. Influence on I-frame Codec

The proposed TransVFC directly uses the feature of the first lossless frame and calculates the bpp value of its original I-frame jpeg file. Additionally, our experiments show that introducing x265 for I-frame compression causes a 5.10% bitrate increase.

5. Conclusion

We propose a TransVFC framework. It offers a scalable solution for multitask VCM scenarios and eliminates the need for fine-tuning the upstream codec and the downstream machine vision tasks. We devise a novel neural-based video feature codec to achieve continuous feature compression; this method incorporates a scheme-based inter-prediction module for feature-domain temporal redundancy squeezing and employs perception-guided conditional coding to make the features better align with machine perception. We designe an FST module to effectively transfer the intermediate features to multiple downstream tasks. Experiments are conducted on three downstream machine vision tasks at different granularities, demonstrating that TransVFC delivers promising compression efficiency and scalability.

Despite these promising results, our approach has limitations. Its performance tends to decrease in low-bitrate scenarios. This decrease may stem from inter-prediction challenges when addressing low-quality features, which introduces cumulative error in the feature domain and affects the overall rate-task performance. In addition, our method exhibits limited robustness when confronted with Gaussian noise in the pixel domain. Moreover, a gap remains between our framework and real-time systems such as FFmpeg [37]. In future work, we plan to increase the performance of our method under low-bitrate constraints, reduce its coding latency, explore new prior utilization and feature fusion strategies, improve robustness on degraded input video, and incorporate a variable-bitrate mechanism.

We hope that our approach can inspire advancements in video feature compression for multitask scenarios and contribute to the development of ATC-based VCM methods.

Declaration of Generative AI and AI-assisted Technologies in the Writing Process

During the preparation of this work, the author(s) used ChatGPT to polish the manuscript and enhance its readability. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the published article.

CRediT authorship contribution statement

Yuxiao Sun: Writing – original draft, Writing – review & editing, Visualization, Methodology, Validation; **Yao Zhao:** Writing – review & editing, Project administration, Supervision, Writing – original draft, Methodology; **Meiqin Liu:** Writing – review & editing, Methodology, Writing – original draft; **Chao Yao:** Writing – review & editing, Methodology, Writing – original draft; **Huihui Bai:** Writing – review & editing, Methodology, Writing – original draft; **Chunyu Lin:** Writing – original draft, Methodology, Writing – review & editing; **Weisi Lin:** Writing – review & editing, Methodology, Writing – original draft.

Data availability

I have attached my github url in manuscript.

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

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

References

[1] R.M.A. Pandeeswari, G. Rajakumar, Deep intelligent technique for person re-identification system in surveillance images, Pattern Recognit. 162 (2025) 111349. https://doi.org/10.1016/j.patcog.2025.111349
[2] X. Liu, L. Jin, X. Han, J. You, Mutual information regularized identity-aware facial expression recognition in compressed video, Pattern Recognit. 119 (2021) 108105.
[3] H. Choi, I.V. Bajic, Scalable image coding for humans and machines, IEEE Trans. Image Process. 31 (2022) 2739–2754. https://doi.org/10.1109/TIP.2022.3160602
[4] E. Özyılkan, M. Ulhaq, H. Choi, F. Racapé, Learned disentangled latent representations for scalable image coding for humans and machines, in: Proceedings of Data Compression Conference (DCC), 2023, pp. 42–51.
[5] J. Li, B. Li, Y. Lu, Neural video compression with feature modulation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024, pp. 26099–26108.
[6] J. Li, B. Li, Y. Lu, Neural video compression with diverse contexts, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 22616–22626.
[7] J. Li, B. Li, Y. Lu, Hybrid spatial-temporal entropy modeling for neural video compression, in: Proceedings of the ACM International Conference on Multimedia, 2022, pp. 1503–1511.
[8] T. Wiegand, G.J. Sullivan, G. Bjontegaard, A. Luthra, Overview of the H.264/AVC video coding standard, IEEE Trans. Circuits Syst. Video Technol. 13 (7) (2003) 560–576.

[9] A. Abramowski, Towards H.265 video coding standard, in: Proceedings of Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments, 8008, SPIE, 2011, pp. 387–393.
[10] Y. Tian, G. Lu, G. Zhai, Z. Gao, Non-semantics suppressed mask learning for unsupervised video semantic compression, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2023, pp. 13610–13622.
[11] Y. Tian, G. Lu, G. Zhai, SMC + +: masked Learning of Unsupervised Video Semantic Compression, arXiv preprint arXiv:2406.04765 (2024).
[12] K. Misra, T. Ji, A. Segall, F. Bossen, Video feature compression for machine tasks, in: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), 2022, pp. 1–6.
[13] J. Shao, J. Zhang, BottleNet + +: an end-to-end approach for feature compression in device-edge co-inference systems, in: Proceedings of IEEE International Conference on Communications Workshops (ICC Workshops), 2020, pp. 1–6.
[14] X. Sheng, L. Li, D. Liu, H. Li, VNVC: a versatile neural video coding framework for efficient human-machine vision, IEEE Trans. Pattern Anal. Mach. Intell. (2024).
[15] M. Yang, F. Yang, L. Murn, M.G. Blanch, J. Sock, S. Wan, F. Yang, L. Herranz, Task-switchable pre-processor for image compression for multiple machine vision tasks, IEEE Trans. Circuits Syst. Video Technol. (2023). https://doi.org/10.1109/TCSVT.2023.3348995
[16] M. Yamazaki, Y. Kora, T. Nakao, X. Lei, K. Yokoo, Deep feature compression using rate-distortion optimization guided autoencoder, in: Proceedings of the IEEE International Conference on Image Processing (ICIP), 2022, pp. 1216–1220.
[17] VTM-23.1, 2024, (https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM). Accessed 2025-01-31.
[18] C. Xu, M. Liu, C. Yao, W. Lin, Y. Zhao, IBVC: Interpolation-driven B-frame video compression, Pattern Recognit. 153 (2024) 110465.
[19] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, Z. Gao, DVC: an end-to-end deep video compression framework, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 11006–11015.
[20] Z. Hu, G. Lu, 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 (CVPR), 2021, pp. 1502–1511.
[21] J. Li, B. Li, Y. Lu, Deep contextual video compression, Adv. Neur. Inform. Proces. Syst. (NeurIPS) 34 (2021) 18114–18125.
[22] X. Sheng, J. Li, B. Li, L. Li, D. Liu, Y. Lu, Temporal context mining for learned video compression, IEEE Trans. Multimedia 25 (2023) 7311–7322.
[23] L.D. Chamain, F. Racapé, J. Bégaint, A. Pushparaja, S. Feltman, End-to-end optimized image compression for machines, a study, in: Proceedings of Data Compression Conference (DCC), 2021, pp. 163–172.
[24] C. Gao, D. Liu, L. Li, F. Wu, Towards task-generic image compression: a study of semantics-oriented metrics, IEEE Trans. Multimedia 25 (2023) 721–735. https://doi.org/10.1109/TMM.2021.3130754
[25] Z. Chen, K. Fan, S. Wang, L.-Y. Duan, W. Lin, A. Kot, Lossy intermediate deep learning feature compression and evaluation, in: Proceedings of the ACM International Conference on Multimedia, 2019, pp. 2414–2422.
[26] W. Yang, H. Huang, Y. Hu, L. Duan, J. Liu, Video coding for machines: compact visual representation compression for intelligent collaborative analytics, IEEE Trans. Pattern Anal. Mach. Intell. (01) (2024) 1–18.
[27] Z. Zhang, M. Wang, M. Ma, J. Li, X. Fan, MSFC: Deep feature compression in multi-task network, in: Proceedings IEEE International Conference on Multimedia and Expo (ICME), 2021, pp. 1–6.
[28] Y. Kim, H. Jeong, J. Yu, Y. Kim, J. Lee, S.Y. Jeong, H.Y. Kim, End-to-end learnable multi-scale feature compression for VCM, IEEE Trans. Circuits Syst. Video Technol. (2023).
[29] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.
[30] R. Gavrilescu, C. Zet, C. Fosalau, M. Skoczylas, D. Cotovanu, Faster R-CNN: an approach to real-time object detection, in: Proceedings of the International Conference and Exposition on Electrical And Power Engineering (EPE), 2018, pp. 0165–0168. https://doi.org/10.1109/ICEPE.2018.8559776
[31] F. Chollet, Xception: deep learning with depthwise separable convolutions, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1251–1258.
[32] X. Zou, Z.-Y. Dou, J. Yang, Z. Gan, L. Li, C. Li, X. Dai, H. Behl, J. Wang, L. Yuan, et al., Generalized decoding for pixel, image, and language, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 15116–15127.
[33] S. Yang, Y. Fang, X. Wang, Y. Li, C. Fang, Y. Shan, B. Feng, W. Liu, Crossover learning for fast online video instance segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 8043–8052.
[34] S.C. Yurtkulu, Y.H. Şahin, G. Unal, Semantic segmentation with extended DeepLabv3 architecture, in: Proceedings of the IEEE Conference on Signal Processing and Communications Applications (SIU), 2019, pp. 1–4.
[35] J. Miao, Y. Wei, Y. Wu, C. Liang, G. Li, Y. Yang, VSPW: A large-scale dataset for video scene parsing in the Wild, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 4133–4143.
[36] HM-18.0, 2023, (https://vcgit.hhi.fraunhofer.de/jvet/HM). Accessed 2025-01-31.
[37] FFmpeg, 2023, (https://github.com/FFmpeg/). Accessed 2025-01-31.
[38] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, R. Girshick, DetectRon2, 2019, (https://github.com/facebookresearch/detectron2).
[39] C. Wei, W. Wang, W. Yang, J. Liu, Deep retinex decomposition for low-light enhancement, arXiv preprint arXiv:1808.04560 (2018).
[40] S.W. Zamir, A. Arora, S. Khan, M. Hayat, F.S. Khan, M.-H. Yang, Restormer: efficient transformer for high-resolution image restoration, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 5728–5739.