Firing Bits Where It Matters: Spiking-Guided Just Recognizable Distortion Modeling for Machine-Centric Video Coding

Wuyuan Xie[1], Zhenming Li[1], Yuwu Lu[2], Di Lin[3], Yun Song[4*], Miaohui Wang[5*]

[1]College of Computer Science & Software Engineering, Shenzhen University
[2]School of Artificial Intelligence, South China Normal University
[3]College of Intelligence and Computing, Tianjing University
[4]School of Computer Science and Technology, Changsha University of Science and Technology
[5]Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University
wuyuan.xie@gmail.com, wang.miaohui@gmail.com

Abstract

Just recognizable distortion (JRD) has emerged as a promising paradigm for machine-centric video coding. However, existing JRD-guided coding methods are limited by coarse annotation granularity and high computational cost, which hinder their deployment. In this paper, we first investigate the impact of different JRD annotation strategies on downstream task performance. By incorporating both instance-level and contextual information, we construct a new JRD dataset with fine-grained annotations compatible with *object detection* and *instance segmentation* tasks. To enhance quantization parameter (QP) map prediction while maintaining computational efficiency, we propose a novel *spiking neural network* (SNN)-based framework that decomposes video frames into spatial structures, channel interactions, and temporal patterns. Furthermore, we introduce a spiking attention mechanism to aggregate task-relevant features and employ adaptive scaling vectors to suppress machine-perceived redundancy, enabling targeted bitrate allocation aligned with task-critical content. Extensive experiments on multiple datasets and backbones demonstrate that our approach consistently outperforms state-of-the-art codec-based and JRD-guided methods in maintaining task performance at ultra-low bitrates, while significantly reducing computational overhead.

1 Introduction

Recent advances in machine vision have accelerated the deployment of applications such as *autonomous driving*, *smart surveillance*, and *industrial automation*. These systems rely on robust visual analysis under constrained computational and transmission conditions. However, existing video coding standards [JVET 2025; Wang, Ngan, and Li 2016], which prioritize perceptual quality for human-centric applications, are misaligned with the needs of automated vision tasks [Yin et al. 2025]. This mismatch has motivated a paradigm shift from coding-for-humans to coding-for-machines, leading to the emergence of video coding for machines (VCM).

Standard codecs such as H.265/HEVC and H.266/VVC are designed to preserve visual fidelity in terms of *texture*, *color*, and *edge sharpness* [Tang et al. 2025; Wang et al. 2021]. However, these human visual cues do not always match the semantic information required by machine vision models [Harell et al. 2025; Sheng et al. 2024]. In practice, allocating bitrate to visually salient but semantically irrelevant regions results in reduced performance for downstream vision tasks like *object detection* or *semantic segmentation*, especially in bandwidth-limited scenarios. This limitation emphasizes the necessity of task-oriented compression, which combines the bitrate allocation with the semantic information needs of machine vision models.

A promising direction in VCM is the concept of "*just recognizable distortion (JRD)*", which defines the maximum tolerable distortion that a machine vision model can accept without significantly degrading task accuracy. JRD-guided compression involves predicting a block-wise quantization parameter (QP) map that reflects the machine's tolerance to information redundancy, enabling more efficient and task-aware bitrate control. The core idea is to preserve semantic features that are important to machine perception while discarding pixel content that is visually critical to the human eye but irrelevant to the specific task.

Recent VCM approaches can be broadly classified into two categories: (1) *Codec-based* methods embed neural network modules into the codec pipeline [Liu et al. 2024; Li et al. 2024; Lu et al. 2025], enabling joint optimization of the compression and task models. While flexible, these methods require custom codec modifications and often involve trade-offs between human and machine objectives. (2) *JRD-guided* methods predict a QP map externally to guide standard codecs [Zhang et al. 2021, 2024b]. These are simpler to deploy and better suited for ultra-low bitrate and task-oriented scenario. Despite their advantages, JRD-guided methods face several limitations:

- **Lack of fine-grained annotations for complex vision tasks** such as *instance segmentation*. Existing datasets focus on detection-level granularity, which is insufficient for tasks requiring detailed object boundaries.
- **Inadequate modeling of spatial and temporal dynamics**, particularly for frame-wise video compression, where understanding subtle shape changes or motion is

*Corresponding author: Yun Song, Miaohui Wang

important for accurate QP allocation.

- **High computational overhead of deep convolutional models**, which limits their use in resource-constrained settings such as edge devices or embedded hardware.

To address these limitations, we propose an efficient and accurate JRD-guided video coding framework based on spiking neural networks (SNNs) which is a biologically inspired architecture characterized by sparse and event-driven computation. Our goal is to build a more practical and scalable solution for task-aware video compression in machine vision pipelines. Specifically:

- We construct a large-scale JRD dataset derived from the COCO benchmark, incorporating both *object detection* and *instance segmentation* annotations. This enables pixel-level supervision of machine-recognizable distortion, facilitating more fine-grained QP prediction.
- We develop an SNN-based model that decomposes visual inputs into spatial structure, channel interactions, and temporal patterns. Leveraging the temporal coding and sparse firing of spiking neurons, our model captures edge-sensitive details and object boundaries more precisely than conventional CNNs, while significantly reducing computation.
- We propose a spiking attention module to selectively aggregate task-relevant features at multiple scales, guided by learned scaling vectors. This enables accurate QP estimation even under challenging conditions such as occlusion, clutter, or fine object structures.
- Extensive experiments on popular benchmarks show that our predicted QP maps lead to better task accuracy at ultra-low bitrates when integrated with H.266/VVC. Our framework outperforms state-of-the-art CNN-based JRD predictors in both detection and segmentation tasks, while offering better efficiency for deployment.

2 Related Work

2.1 Codec-based Methods

Codec-based VCMs optimize compression pipelines by integrating learned neural modules directly into the codec architecture [Choi and Bajić 2022; Lin et al. 2023; Zhang et al. 2024a]. These approaches aim to improve entropy modeling or rate-distortion efficiency through end-to-end training. SegPIC [Liu et al. 2024] introduced a region-adaptive transform module using separable convolutions and affine scaling to handle spatial variability. FTIC [Li et al. 2024] employed a frequency-aware transformer to enhance directional detail preservation. WeConvene [Fu et al. 2024] explicitly modeled frequency-domain correlations using discrete wavelet transforms, improving compression efficiency. AuxTIC [Li et al. 2025] accelerated convergence and improved performance with a wavelet-based linear auxiliary transform that decouples energy compression from fine-grained fitting. DCAE [Lu et al. 2025] incorporated external dictionary priors for latent entropy modeling, enhancing texture detail without added bitrate.

Despite these advances, codec-based methods primarily aim at generic visual quality, often overlooking task-specific machine vision performance, which represents a clear gap in VCMs for intelligent systems.

2.2 JRD-guided Methods

JRD-guided VCMs optimize compression by preserving task-relevant information based on JRD thresholds. EL-JRD [Zhang et al. 2021] first introduced this paradigm, leveraging a learned JRD map to guide encoding for machine vision tasks. Later methods further refined this idea: Fischer *et al.* [Fischer et al. 2021] adopted saliency-driven quantization to differentially encode foreground and background regions. Lee *et al.* [Lee et al. 2023] dynamically allocated QPs to support *object detection*, segmentation, and tracking. Zhang *et al.* [Zhang et al. 2024b] further improved object-wise compression by predicting per-object JRD thresholds and applying aggressive background compression.

While effective, JRD-guided methods typically rely on coarse object-level annotations and heuristic QP assignment. This restricts their applicability to more fine-grained tasks such as *instance segmentation*, which requires pixel-level preservation near boundaries.

2.3 Motivation

Instance segmentation demands pixel-wise accuracy, where perception distortion tolerance is highly sensitive. However, existing JRD-guided VCMs rely on datasets with only object-level annotations and fail to address the fine-grained needs of vision tasks [Zhang et al. 2024a].

Consequently, we investigate a new JRD dataset featuring fine-grained annotations compatible with both *object detection* and *instance segmentation*. This enables more accurate modeling of task-relevant distortion thresholds, especially near edges. Furthermore, recent advances in SNNs provide an energy-efficient and temporally dynamic framework ideally suited for low-power and real-time scenarios [Shen et al. 2025; Zhang et al. 2024c; Zhang and Zhang 2024; Yu et al. 2025]. Compared to traditional deep convolutional networks, SNNs offer: (1) **Temporal coding** via spike trains, capturing fine-grained spatiotemporal variations [Maass 1997]; (2) **Event-driven computation**, significantly reducing energy consumption [Roy, Jaiswal, and Panda 2019]; (3) **Inherent robustness** through spiking thresholds that suppress noises [Sharmin et al. 2020; Ding et al. 2024].

3 Proposed Fine-grained JRD Dataset

3.1 Data Preparation

To quickly build a delicate fine-grained JRD dataset, it is reasonable to adopt existing machine vision-based task datasets. Therefore, we randomly select 10,000 images from COCO2017 with 80 categories [Caesar, Uijlings, and Ferrari 2018], and choose the *instance segmentation* model *Mask R-CNN* [He et al. 2017] with *ResNet-101* as the backbone, and employ a pre-trained model in the widely-used *MMDetection* [Chen et al. 2019] to perform *instance segmentation* to obtain the label result. We employ the reference software *VTM-22.2* [JVET 2025] of the H.266/VVC standard to encode the original images with QPs ranging from 0 to 63, and

decode them to obtain a total of 640,000 distorted images. Finally, these compressed images are fed into the *Mask R-CNN* to obtain the detection results at different compression artifact levels.

3.2 Annotation Rules

Similar to *object detection*, the machine perception redundancy of *instance segmentation* consists of the instance region and the background region. After obtaining 64 compression levels, we adopt separate annotation rules for these two regions. Specifically, for a segmentation model $\mathcal{F}(\cdot)$ and the t-th instance, we can find the maximum compression level $l_{\max} \in [0, 63]$ which does not affect the perception accuracy of the instance, satisfying:

$$\begin{cases} \mathbb{D}\left(\mathcal{F}(R^t_{\text{ori}}), \mathcal{F}(R^t_{\text{com},l})\right) \leq \epsilon, \\ \mathbb{D}\left(\mathcal{F}(R^t_{\text{ori}}), \mathcal{F}(R^t_{\text{com},l+c})\right) > \epsilon, \end{cases}, \tag{1}$$

where R^t_{ori} and $R^t_{\text{com},l}$ refer to the regions of the t-th instance in the original and the distorted images, respectively. $\mathbb{D}(\cdot)$ denotes a difference measure of the prediction results, ϵ denotes the error threshold, and c=1 denotes a small constant.

Instance Annotation For *instance segmentation*, $\mathbb{D}(\cdot)$ needs to be designed by considering four key factors. Specifically, the prediction output P is mainly composed of *segmentation mask* (M), *predicted object box* (B), *confidence score* (S), and *category* (C), defining $P = \{M, B, S, C\}$. Therefore, $\mathbb{D}(\cdot)$ is defined as:

$$\mathbb{D}(\cdot) = \{\mathbb{D}_M(\cdot), \mathbb{D}_B(\cdot), \mathbb{D}_S(\cdot), \mathbb{D}_C(\cdot)\}. \tag{2}$$

Assume that the instance information predicted on R^t_{ori} and $R^t_{\text{com},l}$ are P^t_{ori} and $P^t_{\text{com},l}$, which is defined as:

$$\begin{cases} P^t_{\text{ori}} = \{M^t_{\text{ori}}, B^t_{\text{ori}}, S^t_{\text{ori}}, C^t_{\text{ori}}\} \\ P^t_{\text{com},l} = \{M^t_{\text{com},l}, B^t_{\text{com},l}, S^t_{\text{com},l}, C^t_{\text{com},l}\} \end{cases}. \tag{3}$$

Finally, $\mathbb{D}(\cdot)$ can be summarized as follows:

$$\mathbb{D}(P^t_{\text{ori}}, P^t_{\text{com},l}) = \begin{cases} \mathbb{D}(M^t_{\text{ori}}, M^t_{\text{com},l}) = 1 - IoU(M^t_{\text{ori}}, M^t_{\text{com},l}) \\ \mathbb{D}(B^t_{\text{ori}}, B^t_{\text{com},l}) = 1 - IoU(B^t_{\text{ori}}, B^t_{\text{com},l}) \\ \mathbb{D}(S^t_{\text{ori}}, S^t_{\text{com},l}) = S^t_{\text{ori}} - S^t_{\text{com},l} \\ \mathbb{D}(C^t_{\text{ori}}, C^t_{\text{com},l}) = |C^t_{\text{ori}} - C^t_{\text{com},l}| \end{cases}, \tag{4}$$

where $IoU(\cdot)$ represents the intersection-over-union function, and $|\cdot|$ denotes the absolute value function. The thresholds $\epsilon = \{\epsilon_M, \epsilon_B, \epsilon_S, \epsilon_C\}$ of these four difference metrics need to be set separately. When all four thresholds are all less than or equal to the corresponding thresholds (*i.e.*, $\mathbb{D}(M^t_{\text{ori}}, M^t_{\text{com},l}) \leq \epsilon_M, \mathbb{D}(B^t_{\text{ori}}, B^t_{\text{com},l}) \leq \epsilon_B, \mathbb{D}(S^t_{\text{ori}}, S^t_{\text{com},l}) \leq \epsilon_S, \mathbb{D}(C^t_{\text{ori}}, C^t_{\text{com},l}) \leq \epsilon_C$), the current compression level l can be considered to not affect the machine recognition accuracy of the t-th instance. In the experiments, we set ϵ_M, ϵ_B, ϵ_S and ϵ_C to 0.05, 0.1, 0.05 and 0, respectively. It means that 5% difference is allowed for the segmentation mask and confidence, 10% difference is allowed for the bounding box, and ϵ_C is set to 0, indicating that no category difference is allowed for *object detection* and *instance segmentation*.

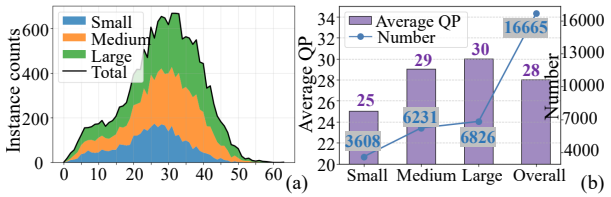


Figure 1: Statistical analysis of the proposed dataset. (a) Instance distribution across different compression levels. (b) The relationship between the average QP and instance size.

Background Annotation Previous studies applied restrictive heuristics to define background regions. For example, [Zhang et al. 2021] considered only the "`person`" category, while [Zhang et al. 2024b] used a high confidence threshold (0.9) to filter detections. However, using such a high threshold discards nearly half the detection boxes (*e.g.*, reducing from 60,000 to 32,000 boxes over 10,000 images), which can misclassify valid object regions as background. This compromises both annotation quality and the model's capacity to suppress redundant information.

To address this, we adopt a lower confidence threshold of 0.3, retaining 92.3% of the original detections. Additionally, we apply an IoU threshold of 0.75, aligned with common evaluation standards (*e.g.*, COCO and VOC), to match predictions with ground truth. Only regions with both IoU $>$0.75 and the confidence $S$$>$0.3 are labeled as instance regions, while all others are considered background region.

3.3 Annotation Generation

With instance and background masks identified, we construct a per-image QP map. An integer matrix initialized to 63 (the maximum QP) is created at the original resolution. Instance masks are overlaid, and overlapping regions take the minimum QP to preserve fine details.

To comply with H.266/VVC, each QP map is then downsampled according to the coding tree unit (CTU) structure. Using a CTU size of 64 yields a grid-level QP map aligned with practical encoder settings.

3.4 Dataset Analysis

Figure 1(a) shows the QP distribution across instance sizes (*small*, *medium*, *large*, *total*). The distribution is roughly normal and spans the full QP range, with instances present even in extreme intervals (0–5 and 50–55), ensuring broad coverage. As shown in Figure 1(b), average QP increases with instance size: from about 25 (small) to 30 (large), indicating reasonable compression allocation. The size ratio (1:1.7:1.9) mirrors real-world distributions, and the dataset's 16,665 instances provide strong statistical support for data-driven analysis.

In summary, our dataset has fine-grained and statistically representative annotations with realistic instance distributions. It is suited for compression analysis, model training, and parameter optimization in JRD-guided VCM.

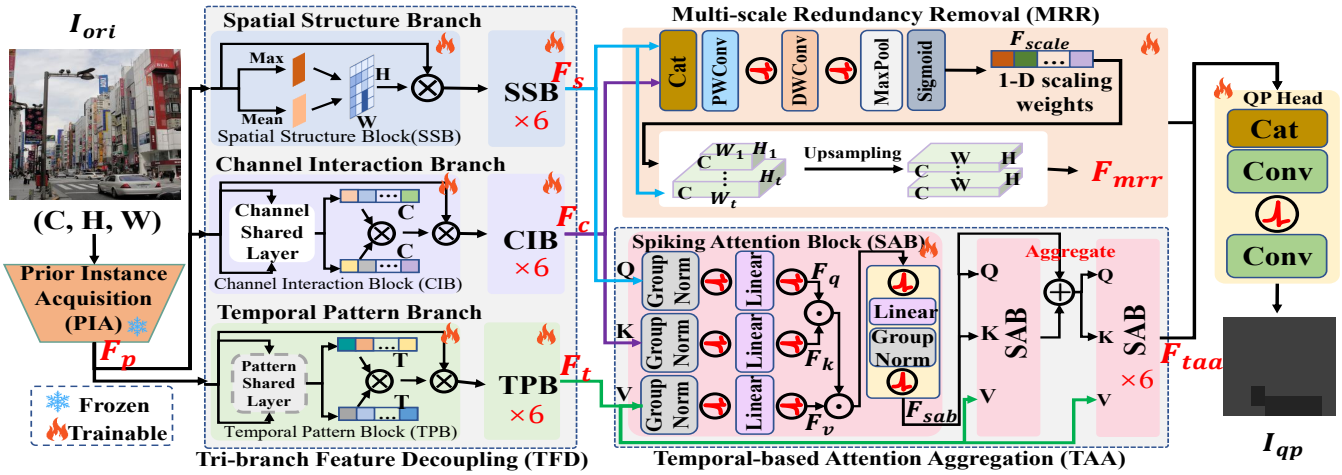


Figure 2: Pipeline of the proposed spiking-guided JRD prediction model. It consists of four key modules: prior instance acquisition (PIA), tri-branch feature decoupling (TFD), multi-scale redundancy removal (MRR), and temporal-based attention aggregation (TAA). PIA extracts the initial instance feature, TFD disentangles feature representations along spatial, channel, and temporal dimensions, MRR suppresses task-irrelevant redundancy using adaptive scaling weights, and TAA enhances temporal feature modeling via spiking attention blocks (SABs).

4 Proposed Spiking-Guided JRD Framework

4.1 Problem Formulation

Our objective is to minimize the pixel-wise difference between the predicted QP map I_{qp} and the ground-truth JRD label I_{jrd}. In practice, existing methods optimize this task using global or locally weighted loss functions across the entire image. However, optimizing based on a global loss often fails to yield accurate pixel-level QP predictions, particularly in regions with high distortion sensitivity. To address this, we introduce an error-aware masking strategy. Specifically, we compute the absolute error matrix $I_{\text{diff}} = |I_{\text{qp}} - I_{\text{jrd}}|$ and apply a threshold θ. Pixels in I_{diff} with values greater than θ are assigned a value of 1 in a corresponding binary mask matrix I_{mask}, and 0 otherwise. We then perform element-wise multiplication between I_{diff} and I_{mask} to obtain a focused pixel-wise difference matrix, which highlights the regions the model should prioritize. In addition, we compute the ℓ_1 loss between I_{qp} and I_{jrd}, capturing global content-level consistency. The final loss $\mathcal{L}_{total}$ is a weighted combination of the pixel-wise masked loss and the ℓ_1 loss:

$$\mathcal{L}_{total} = \alpha ||I_{\text{qp}} - I_{\text{jrd}}||_1 + (1-\alpha)\frac{SUM(I_{\text{diff}} \otimes I_{\text{mask}})}{SUM(I_{\text{mask}})}, \quad \text{s.t.} \quad I_{\text{diff}} = |I_{\text{qp}} - I_{\text{jrd}}|, I_{\text{mask}} = (I_{\text{diff}} > \theta), \tag{5}$$

where $\otimes$ represents the element-wise multiplication, and $SUM(\cdot)$ represents the summation operation. The weighting factor α is set to 0.3, and the threshold θ is empirically determined as 3 based on extensive experiments.

4.2 Prior Instance Acquisition

It is critical to obtain instance feature before decoupling it into spatial structure, channel interaction, and temporal pattern. In the PIA module $\mathcal{F}_{\text{PIA}}$, we choose the pretrained *Mask R-CNN* [He et al. 2017]. Each image $I_{\text{ori}} \in \mathbb{R}^{C\times H\times W}$ is corresponded to a processed result $I_p \in \mathbb{R}^{N_{\text{ins}}\times 1\times H\times W}$, where N_{ins} represents the number of instances. The maximum reduction is performed to integrate all instance features into an overall matrix $I_{whole} \in \mathbb{R}^{1\times H\times W}$. Then the tensor concatenation operation is performed on the overall matrix, and the output is denoted as $\mathbf{F}_p \in \mathbb{R}^{3\times H\times W}$.

$$\mathbf{F}_p = \mathcal{F}_{\text{PIA}}(I_{\text{ori}}). \tag{6}$$

4.3 Tri-branch Feature Decoupling

The TFD module $\mathcal{F}_{\text{TFD}}$ consists of three branches, and outputs the spatial structure feature $\mathbf{F}_s$, the channel interaction feature $\mathbf{F}_c$, and the temporal pattern feature $\mathbf{F}_t$:

$$\mathbf{F}_s, \mathbf{F}_c, \mathbf{F}_t = \mathcal{F}_{\text{TFD}}(\mathbf{F}_p), \tag{7}$$

Spatial Structure Branch. The spatial structure branch is composed of multiple spatial structure blocks (SSBs) $\mathcal{F}_{\text{ssb}}$, which aims to capture the global and local structural features. Specifically, $\mathbf{F}_p$ is first reshaped, and the maximum feature $\mathbf{F}_p^{max}$ and the mean feature $\mathbf{F}_p^{mean}$ are concatenated (`Concat`). A spatial feature weight matrix $\mathbf{F}_{sw} \in \mathbb{R}^{1\times H\times W}$ is obtained by passing through a convolutional layer (`Conv`), leaky integrate-and-fire spiking neuron (SN), and `Softmax` function as shown in Figure 3:

$$\mathbf{F}_{sw}^1 = \texttt{Softmax}(SN(\texttt{Conv}(\texttt{Concat}(\mathbf{F}_p^{max}, \mathbf{F}_p^{mean})))), \tag{8}$$

Subsequently, $\mathbf{F}_{sw}^1$ is used to guide $\mathbf{F}_p$ to obtain the first SSB feature $\mathbf{F}_{ssb}^1$, which can be expressed as:

$$\mathbf{F}_{\text{ssb}}^1 = \mathcal{F}_{\text{ssb}}(\mathbf{F}_p) = \texttt{Reshape}\left(SN(\texttt{Conv}(\mathbf{F}_{sw}^1 \otimes \mathbf{F}_p))\right), \tag{9}$$

Finally, the spatial structure feature $\mathbf{F}_s$ can be obtained by $\mathcal{F}_{\text{ssb}}^i(\cdots \mathcal{F}_{\text{ssb}}^1(\mathbf{F}_p))$ in a stacked manner.

Channel Interaction Branch. The channel interaction branch is composed of multiple channel interaction blocks (CIBs) $\mathcal{F}_{\text{cib}}$, which aims to capture dimensional correlation between redundant machine channel features. Specifically, $\mathbf{F}_p$ is passed to the two ordered modules $\{\texttt{DWConv}, GN, SN\}$ and $\{\texttt{PWConv}, GN, SN\}$ to obtain the latent features $\{\mathbf{F}_p^{c1}, \mathbf{F}_p^{c2}\}$, where DWConv represents depth-wise convolution, PWConv represents point-wise convolution, and GN represents group normalization. $\mathbf{F}_p^{c1}$ is passed to a channel shared layer (CSL) $\mathcal{F}_{\text{csl}}$ to obtain one-dimensional feature vector $\mathbf{F}_{cw}^{1,1} = \texttt{AvgPool}(SN(\texttt{Conv}(\mathbf{F}_p^{c1})))$. Similarly, we obtain $\mathbf{F}_{cw}^{1,2}$.

Let $\mathbf{F}_{cw}^{i,j}$ represent the j-th channel weight vector in the i-th CIB block. $\mathbf{F}_{cw}^{1,1}$ and $\mathbf{F}_{cw}^{1,2}$ are used to guide $\mathbf{F}_p$ to obtain the first CIB feature $\mathbf{F}_{cib}^1$, which can be expressed as:

$$\mathbf{F}_{cib}^1 = \mathcal{F}_{\text{cib}}(\mathbf{F}_{cw}^{1,1}, \mathbf{F}_{cw}^{1,2}, \mathbf{F}_p)) = \texttt{Reshape}(SN(\texttt{Conv}(\texttt{Softmax}\left(\mathbf{F}_{cw}^{1,1} \otimes \mathbf{F}_{cw}^{1,2}\right) \otimes \mathbf{F}_p))), \tag{10}$$

Finally, the channel interaction feature $\mathbf{F}_c$ can be obtained by $\mathcal{F}_{\text{cib}}^i(\cdots \mathcal{F}_{\text{cib}}^1(\mathbf{F}_{cw}^{1,1}, \mathbf{F}_{cw}^{1,2}, \mathbf{F}_p))$ in a stacked manner.

Temporal Pattern Branch. The temporal pattern branch is composed of multiple temporal pattern blocks (TPBs) $\mathcal{F}_{\text{tpb}}$, which aims to capture the dynamic visual changes using the spiking sequence properties. Specifically, $\mathbf{F}_p$ is passed to the two ordered modules $\{\texttt{Reshape}, \texttt{AvgPool}\}$ and $\{\texttt{Reshape}, \texttt{MaxPool}\}$ to obtain the latent features $\{\mathbf{F}_p^{t1}, \mathbf{F}_p^{t2}\}$. $\mathbf{F}_p^{t1}$ is passed to a pattern shared layer (PSL) $\mathcal{F}_{\text{psl}}$ to obtain one-dimensional feature vector $\mathbf{F}_{tw}^{1,1} = \texttt{Conv}(SN(\texttt{Conv}(\mathbf{F}_p^{t1})))$. Similarly, we obtain $\mathbf{F}_{tw}^{1,2}$.

Subsequently, the first TPB feature $\mathbf{F}_{tpb}^1$ is obtained by replacing $\{\mathbf{F}_{cw}^{1,1}, \mathbf{F}_{cw}^{1,2}\}$ in Eq. (10) with $\{\mathbf{F}_{tw}^{1,1}, \mathbf{F}_{tw}^{1,2}\}$, where the definition of $\mathcal{F}_{\text{tpb}}$ is the same as $\mathcal{F}_{\text{cib}}$. Finally, the temporal pattern feature $\mathbf{F}_t$ can be obtained by $\mathcal{F}_{\text{tpb}}^i(\cdots \mathcal{F}_{\text{tpb}}^1(\mathbf{F}_{tw}^{1,1}, \mathbf{F}_{tw}^{1,2}, \mathbf{F}_p))$ in a stacked manner.

4.4 Multi-scale Redundancy Removal

To adapt the model to the redundant features of instances of different sizes in images of different resolutions, we further obtain the corresponding adaptive scaling weight $\mathbf{F}_{scale}$. Specifically, the spatial structure feature $\mathbf{F}_s$ and the channel interaction feature $\mathbf{F}_c$ are passed into the MRR module $\mathcal{F}_{\text{mrr}}$ to extract the one-dimensional scaling vector $\mathbf{F}_{scale}$, which is defined as follows:

$$\mathbf{F}_{scale} = \texttt{Sigmoid}(\texttt{MaxPool}(SN(\texttt{DWConv}(SN(\texttt{PWConv}(\texttt{Concat}(\mathbf{F}_s, \mathbf{F}_c))))))), \tag{11}$$

The scaling vector $\mathbf{F}_{scale}$ is used to guide the multi-scale sampling operation of the spatial structure feature $\mathbf{F}_s$. Specifically, after the multi-scale downsampling operation (Downsample), the spatial features of different scales is obtained, and then the upsampling operation (Upsample) is performed to restore to the previous feature map size.

$$\mathbf{F}_{mrr} = \mathcal{F}_{\text{mrr}}(\mathbf{F}_s, \mathbf{F}_c) = \texttt{Upsample}\left(\texttt{Downsample}(\mathbf{F}_s, \mathbf{F}_{scale})\right), \tag{12}$$

where F_{mrr} represents the less redundant features obtained through the MRR module.

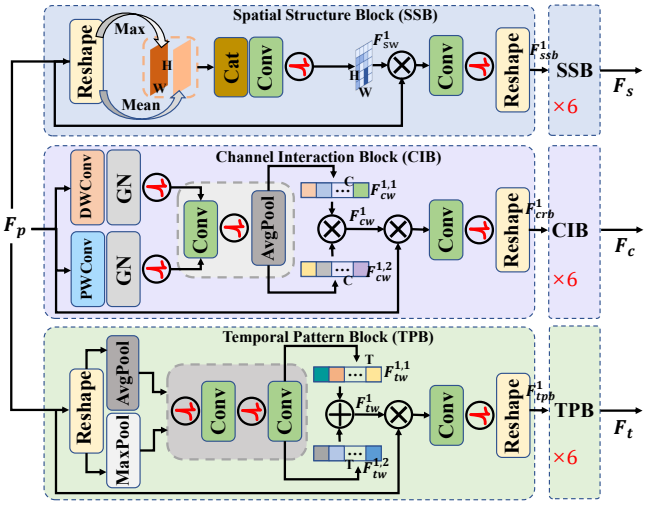


Figure 3: Tri-branch feature decoupling (TFD) module. Three specialized branches are composed of spatial structure blocks (SSBs), channel interaction blocks (CIBs), and temporal pattern blocks (TPBs), respectively.

4.5 Temporal-based Attention Aggregation

The TAA module $\mathcal{F}_{\text{TAA}}$ is composed of multiple spiking attention blocks (SABs), where each SAB uses spiking neurons to implement attention weight calculation. It aims to obtain the temporal attention feature $\mathbf{F}_{taa}$ based on the tri-branch decoupling features:

$$\mathbf{F}_{taa} = \mathcal{F}_{\text{TAA}}(\mathbf{F}_s, \mathbf{F}_c, \mathbf{F}_t). \tag{13}$$

Specifically, $\mathbf{F}_s$, $\mathbf{F}_c$, and $\mathbf{F}_t$ are used as the query (Q), key (K), and value (V) in the SAB, respectively. Let $\mathbf{F}_{q,k,v}^1 = SN\left(\texttt{Linear}(SN(GN(\mathbf{F}_{s,c,t})))\right)$ represent the Q, K, or V values in the first SAB, respectively. They are used in the following pulse attention process to obtain the output $\mathbf{F}_{sab}^1$ of the first SAB:

$$\mathbf{F}_{sab}^1 = \mathcal{F}_{\text{sab}}(\mathbf{F}_q^1, \mathbf{F}_k^1, \mathbf{F}_v^1) = SN(GN(\texttt{Linear}(SN(\frac{\mathbf{F}_q^1 \odot (\mathbf{F}_k^1)^T}{\sqrt{S}} \odot \mathbf{F}_v^1)))), \tag{14}$$

where $\odot$ represents the matrix multiplication, and S is a constant and is set to 0.125.

Finally, SAB blocks are cascaded, and the output is $\mathbf{F}_{taa}$. The V value input of each SAB is $\mathbf{F}_t$, but the Q and K values vary depending on the SAB number, which is defined as:

$$\mathbf{F}_{sab}^i = \mathbf{F}_{sab}^{i-1} \oplus \mathcal{F}_{\text{sab}}(\mathbf{F}_{sab}^{i-1}, \mathbf{F}_{sab}^{i-1}, \mathbf{F}_t). \tag{15}$$

4.6 QP Head

To integrate the less redundant features $\mathbf{F}_{mrr}$ and the temporal attention features $\mathbf{F}_{taa}$ to generate the QP map $I_{\text{qp}} \in \mathbb{R}^{\lceil H/64 \rceil \times \lceil W/64 \rceil}$, we design the QP head with the help of SN and Conv layers to achieve low-cost computation, which is defined as:

$$I_{\text{qp}} = \mathcal{F}_{\text{qp}}(\mathbf{F}_{mrr}, \mathbf{F}_{taa}) = \texttt{Conv}(SN(\texttt{Conv}(\texttt{Concat}(\mathbf{F}_{mrr}, \mathbf{F}_{taa})))). \tag{16}$$

TFD	MRR	TAA	E_s	E_m	E_l	E_{total}
-	-	-	12.36	11.25	10.17	10.40
✓	-	-	11.10	10.42	9.32	9.56
-	✓	-	10.59	9.27	9.01	9.11
-	-	✓	10.78	9.39	9.20	9.43
✓	✓	-	9.71	9.25	8.33	8.72
✓	-	✓	9.86	9.42	8.67	8.93
-	✓	✓	9.81	9.33	8.66	8.83
✓	✓	✓	9.17	8.73	8.17	8.53

Table 1: Ablation experiments of TFD, MRR, and TAA modules. '-' indicates that the current module is disabled, while '✓' indicates that it is enabled.

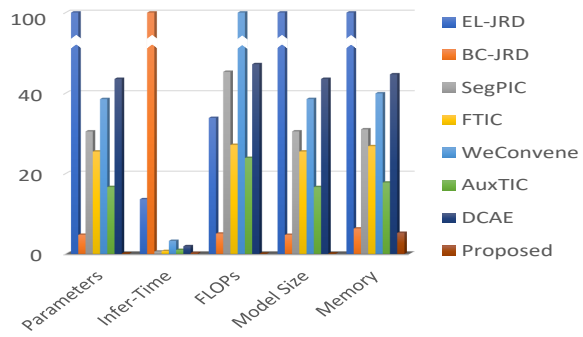


Figure 4: Model complexity across eight methods using five metrics: parameter count, memory usage, model size, FLOPs, and inference time.

5 Experimental Validations

5.1 Datasets

In the experiments, we conduct experiments on three benchmark datasets, including two image datasets COCO2017 [Caesar, Uijlings, and Ferrari 2018] and VOC2012 [Everingham et al. 2015] and one video dataset TVD2022 [Gao et al. 2022]. Specifically, the details for three datasets are

- **COCO**: Used for our fine-grained JRD annotation and end-to-end training. Our annotated dataset is split into 8:1:1 for training, validation, and testing.
- **VOC and TVD**: Used for the cross-dataset testing to evaluate the generalization to new domains.

5.2 Comparison Methods

We have conducted the comparison experiments on both codec-based and JRD-guided baselines:

- **Codec-based** methods: SegPIC [Liu et al. 2024], FTIC [Li et al. 2024], WeConvene [Fu et al. 2024], AuxTIC [Li et al. 2025], and DCAE [Lu et al. 2025],.
- **JRD-guided** methods: EL-JRD [Zhang et al. 2021], and BC-JRD [Zhang et al. 2024b].

Since EL-JRD does not provide official code or model weights, we reproduce its predictor using a binary discriminator trained on eight distortion levels (32, 38, 41, 43, 45, 47, 49, 51). We use a cross-entropy loss function with the class-balanced weights, which is consistent with the class adjustment strategy in [Zhang et al. 2021]. For all other methods, we use publicly available implementations and fine-tune them on our fine-grained dataset following their official training protocols.

5.3 Evaluation Metrics

Based on the requirements of machine vision tasks, we adopt the following three metrics to evaluate performance. (1) **Bitrate**: Average bits-per-pixel (BPP). (2) **Visual quality**: Peak Signal-to-Noise Ratio (PSNR). (3) **Task accuracy**: Mean Average Precision (mAP@0.50) for *object detection* and *instance segmentation*.

For codec-based methods, we test multiple compression levels by using pre-trained checkpoints optimized for the MSE distortion as suggested in [Lu et al. 2025]. For JRD-guided methods, we follow the QP-offset adjustment protocol from [Zhang et al. 2024b], which simulates compression variation by adjusting predicted QP values. It is noted that a better JRD prediction model should maintain consistent task performance under such adjustments.

5.4 Implementation Details

All experiments are conducted on a server with an *Intel Xeon Silver 4210R CPU* and an *NVIDIA GeForce RTX 3090 GPU*. We use the *MMDetection* benchmark [Chen et al. 2019] to evaluate task performance using five models: *Faster R-CNN*, *Mask R-CNN*, *YOLO*, *DETR*, and *SOLO*.

In the TFD module, the total numbers of SSB, CIB, and TPB are set to 6. Their output dimensions are {3, 3, 6, 6, 16, 64}, and the time step T in *SN* is set to 10. Images are normalized but not resized or cropped to preserve structural information. Due to variable training image sizes, the batch size is set to 1. Training is performed for 50 epochs using the *Adam* optimizer with a learning rate of 4*e*-4.

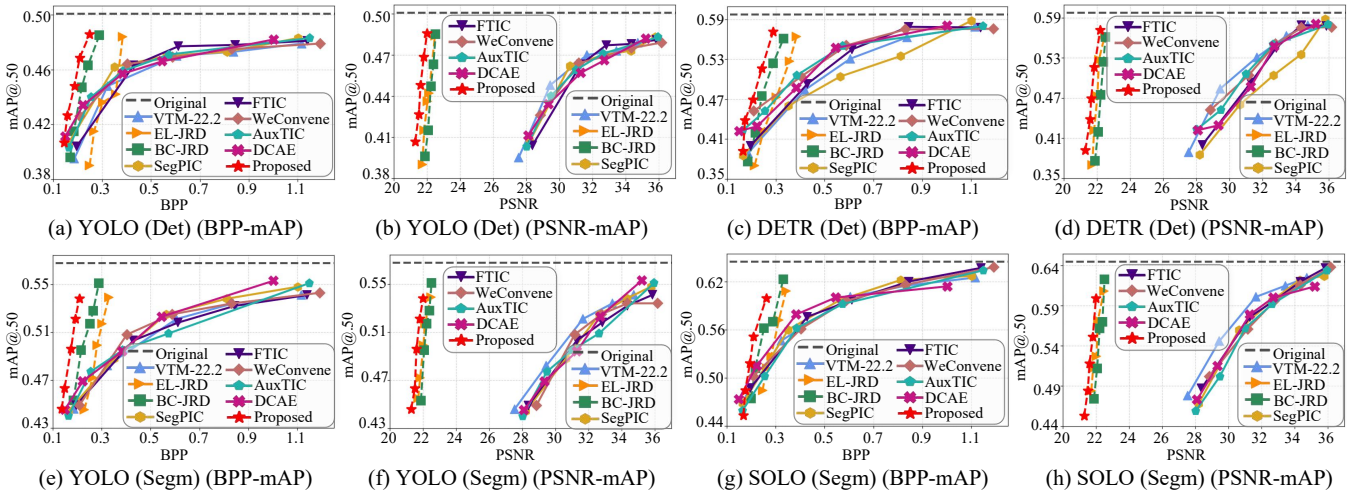
5.5 Ablation Study

To evaluate the contribution of each module, we perform ablation experiments on the three trainable modules: TFD, MRR, and TAA. Specifically, TFD and MRR are replaced by standard 3×3 convolution layers. TAA is replaced with a simpler attention mechanism, where the Q, K, and V features are concatenated along the channel dimension and processed by a convolution layer.

Table 1 demonstrates that removing any single module or any combination of two modules leads to increased QP prediction errors, thereby validating the contribution of each component in our spiking-guided JRD framework.

5.6 Performance Comparison

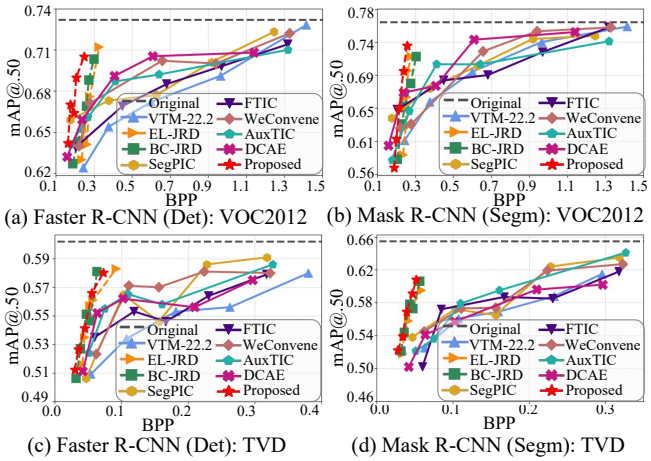
Model complexity. The results are summarized in Figure 4. As seen, the proposed spiking-guided JRD model demonstrates a promising average performance. Specifically, it exhibits significantly lower overhead in terms of parameters, inference time, memory, model size, and FLOPs, and it is 400× faster than BC-JRD. These advantages are attributed



(a) YOLO (Det) (BPP-mAP) (b) YOLO (Det) (PSNR-mAP) (c) DETR (Det) (BPP-mAP) (d) DETR (Det) (PSNR-mAP)

(e) YOLO (Segm) (BPP-mAP) (f) YOLO (Segm) (PSNR-mAP) (g) SOLO (Segm) (BPP-mAP) (h) SOLO (Segm) (PSNR-mAP)

Figure 5: Cross-model comparison of BPP-mAP and PSNR-mAP curves on both *object detection* (Det) and *instance segmentation* (Segm) tasks. (a)–(b) and (c)–(d) show the BPP-mAP@.50 and PSNR-mAP@.50 curves using YOLO and DETR as detection, respectively. (e)–(f) and (g)–(h) show the corresponding curves using YOLO and SOLO as segmentation.



(a) Faster R-CNN (Det): VOC2012 (b) Mask R-CNN (Segm): VOC2012

(c) Faster R-CNN (Det): TVD (d) Mask R-CNN (Segm): TVD

Figure 6: Cross-dataset comparison of BPP-mAP curves for detection (Det) and segmentation (Segm) tasks. (a)–(b) present the results on the VOC dataset, while (c)–(d) show results on the TVD dataset.

to the sparse activation and event-driven computation characteristics of SNNs, which reduce unnecessary processing and improve efficiency. Overall, our method exhibits strong performance in computational efficiency, making it particularly well-suited for deployment in resource-limited scenarios.

Cross-model comparison. To test generalization across different tasks, we conduct the experiments on three detection and segmentation models on our COCO testing set.

Figures 5(a) to 5(d) show the BPP versus mAP@.50 for *object detection*, while Figures 5(e) to 5(h) show the same for *instance segmentation*. As seen, our method consistently achieves better task accuracy compared to all baseline methods. Although our method sacrifices some visual quality (lower PSNR) compared to codec-based methods, it preserves significantly more task-relevant features. Compared to JRD-guided methods, our method produces comparable or better task performance, but our spiking-guided JRD method achieves significantly low computational complexity (see Figure 4).

Cross-dataset experiment. To evaluate the robustness, we conduct the cross-dataset experiments on the VOC and TVD datasets. *Faster R-CNN* and *Mask R-CNN* are used as the detection and segmentation benchmark models.

Figure 6 shows that our method outperforms all seven baselines in most cases across both datasets. These results demonstrate that our method generalizes well to unseen data and remains effective across varied domains, confirming the reliability and accuracy of its JRD prediction.

6 Conclusion

In this paper, we present a new spiking-guided just recognizable distortion (JRD) framework for machine-centric video coding. First, we construct a fine-grained JRD annotation dataset that supports both *instance segmentation* and *object detection*, addressing the limitations of coarse annotations in previous benchmarks. In addition, we introduce a novel *spiking neural network* (SNN)-based architecture that enables end-to-end QP map prediction through tri-branch temporal-spatial-channel decomposition and multi-scale feature guidance, effectively eliminating reliance on reference images and adaptively adjusting to instance-level characteristics. Extensive experiments across multiple vision backbones and datasets demonstrate that our method consistently achieves superior task performance preservation and low complexity at ultra-low bitrates. We believe this approach will further advance the practical deployment of task-oriented compression, particularly in resource-constrained environments.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 62472290 and 62372306, and in part by the Natural Science Foundation of Guangdong Province under Grants 2024A1515011972 and 2023A1515011197.

References

Caesar, H.; Uijlings, J.; and Ferrari, V. 2018. Coco-stuff: Thing and stuff classes in context. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1209–1218.

Chen, K.; Wang, J.; Pang, J.; Cao, Y.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Xu, J.; Zhang, Z.; Cheng, D.; Zhu, C.; Cheng, T.; Zhao, Q.; Li, B.; Lu, X.; Zhu, R.; Wu, Y.; Dai, J.; Wang, J.; Shi, J.; Ouyang, W.; Loy, C. C.; and Lin, D. 2019. MMDetection: Open MMLab Detection Toolbox and Benchmark. *arXiv preprint arXiv:1906.07155*.

Choi, H.; and Bajić, I. V. 2022. Scalable image coding for humans and machines. *IEEE Transactions on Image Processing*, 31: 2739–2754.

Ding, J.; Yu, Z.; Huang, T.; and Liu, J. K. 2024. Enhancing the robustness of spiking neural networks with stochastic gating mechanisms. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, 492–502.

Everingham, M.; Eslami, S. A.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2015. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1): 98–136.

Fischer, K.; Fleckenstein, F.; Herglotz, C.; and Kaup, A. 2021. Saliency-driven versatile video coding for neural object detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1505–1509.

Fu, H.; Liang, J.; Fang, Z.; Han, J.; Liang, F.; and Zhang, G. 2024. Weconvene: Learned image compression with wavelet-domain convolution and entropy model. In *European Conference on Computer Vision (ECCV)*, 37–53.

Gao, W.; Xu, X.; Qin, M.; and Liu, S. 2022. An open dataset for video coding for machines standardization. In *IEEE International Conference on Image Processing (ICIP)*, 4008–4012.

Harell, A.; Foroutan, Y.; Ahuja, N.; Datta, P.; Kanzariya, B.; Somayazulu, V. S.; Tickoo, O.; de Andrade, A.; and Bajić, I. V. 2025. Rate-distortion theory in coding for machines and its applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(7): 5501–5519.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, 2961–2969.

JVET. 2025. VTM reference software for H.266/VVC, Joint Video Exploration Team (JVET). *[Online]. Available:* https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM.

Lee, Y.; Kim, S.; Yoon, K.; Lim, H.; Kwak, S.; and Choo, H.-G. 2023. Machine-Attention-based Video Coding for Machines. In *IEEE International Conference on Image Processing (ICIP)*, 2700–2704.

Li, H.; Li, S.; Dai, W.; Cao, M.; Kan, N.; Li, C.; Zou, J.; and Xiong, H. 2025. On disentangled training for nonlinear transform in learned image compression. In *International Conference on Learning Representations (ICLR)*.

Li, H.; Li, S.; Dai, W.; Li, C.; Zou, J.; and Xiong, H. 2024. Frequency-aware transformer for learned image compression. In *International Conference on Learning Representations (ICLR)*.

Lin, H.; Chen, B.; Zhang, Z.; Lin, J.; Wang, X.; and Zhao, T. 2023. DeepSVC: Deep scalable video coding for both machine and human vision. In *Proceedings of the 31st ACM International Conference on Multimedia (ACM MM)*, 9205–9214.

Liu, Y.; Yang, W.; Bai, H.; Wei, Y.; and Zhao, Y. 2024. Region-adaptive transform with segmentation prior for image compression. In *European Conference on Computer Vision (ECCV)*, 181–197.

Lu, J.; Zhang, L.; Zhou, X.; Li, M.; Li, W.; and Gu, S. 2025. Learned Image Compression with Dictionary-based Entropy Model. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, 12850–12859.

Maass, W. 1997. Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 10(9): 1659–1671.

Roy, K.; Jaiswal, A.; and Panda, P. 2019. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784): 607–617.

Sharmin, S.; Rathi, N.; Panda, P.; and Roy, K. 2020. Inherent adversarial robustness of deep spiking neural networks: Effects of discrete input encoding and non-linear activations. In *European Conference on Computer Vision (ECCV)*, 399–414.

Shen, S.; Wang, C.; Huang, R.; Zhong, Y.; Guo, Q.; Lu, Z.; Zhang, J.; and Leng, L. 2025. Spikingssms: Learning long sequences with sparse and parallel spiking state space models. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 39, 20380–20388.

Sheng, X.; Li, L.; Liu, D.; and Li, H. 2024. VNVC: A versatile neural video coding framework for efficient human-machine vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(7): 4579–4596.

Tang, C.; Li, Z.; Bian, Y.; Li, L.; and Liu, D. 2025. Neural Video Compression with Context Modulation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 12553–12563.

Wang, M.; Ngan, K. N.; and Li, H. 2016. Low-delay rate control for consistent quality using distortion-based Lagrange multiplier. *IEEE Transactions on Image Processing*, 25(7): 2943–2955.

Wang, M.; Zhang, J.; Huang, L.; and Xiong, J. 2021. Machine learning-based rate distortion modeling for VVC/H.266 intra-frame. In *IEEE International Conference on Multimedia and Expo (ICME)*, 1–6.

Yin, K.; Liu, Q.; Shen, X.; He, Y.; Yang, W.; and Wang, S. 2025. Unified Coding for Both Human Perception and Generalized Machine Analytics with CLIP Supervision. In *AAAI*

Conference on Artificial Intelligence (AAAI), volume 39, 9517–9525.

Yu, K.; Zhang, T.; Wang, H.; and Xu, Q. 2025. FSTA-SNN: Frequency-Based Spatial-Temporal Attention Module for Spiking Neural Networks. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 39, 22227–22235.

Zhang, H.; and Zhang, Y. 2024. Memory-efficient reversible spiking neural networks. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, 16759–16767.

Zhang, Q.; Wang, S.; Zhang, X.; Jia, C.; Wang, Z.; Ma, S.; and Gao, W. 2024a. Perceptual video coding for machines via satisfied machine ratio modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12): 7651–7668.

Zhang, Q.; Wang, S.; Zhang, X.; Ma, S.; and Gao, W. 2021. Just recognizable distortion for machine vision oriented image and video coding. *International Journal of Computer Vision*, 129(10): 2889–2906.

Zhang, Y.; Lin, H.; Sun, J.; Zhu, L.; and Kwong, S. 2024b. Learning to predict object-wise just recognizable distortion for image and video compression. *IEEE Transactions on Multimedia*, 26: 5925–5938.

Zhang, Y.; Liu, X.; Chen, Y.; Peng, W.; Guo, Y.; Huang, X.; and Ma, Z. 2024c. Enhancing representation of spiking neural networks via similarity-sensitive contrastive learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, 16926–16934.