Zhang *et al.*
EURASIP Journal on Image and Video Processing (2025) 2025:3
https://doi.org/10.1186/s13640-024-00661-0

EURASIP Journal on Image and Video Processing

RESEARCH Open Access

Side information-driven image coding for hybrid machine–human vision

Zhongpeng Zhang[1], Ying Liu[1*] and Wen-Hsiao Peng[2]

*Correspondence:
yliu15@scu.edu

[1] Department of Computer Science and Engineering, Santa Clara University, 500 El Camino Real, Santa Clara, California 95053, USA
[2] Department of Computer Science, National Yang Ming Chiao Tung University, 1001 Ta-Hsueh Rd., Hsinchu 30010, Taiwan

Abstract

With the development of machine learning, advanced photography and image transmission systems, images are being processed more and more by machines, so image coding for machines (ICM) came into being. After the image codec compresses and transmits the image, the image will be handed over to machine vision task networks. These vision tasks include image classification, semantic segmentation, and so on. We propose a side information-driven image coding for hybrid machine–human vision (SICMH) framework, not only for machine vision tasks, but also for human vision-oriented image reconstruction. The proposed SICMH framework can perform image classification, semantic segmentation, and coarse image reconstruction by using purely the side information. Moreover, SICMH can perform fine image reconstruction by using the residue information. In particular, we propose a multi-scale feature fusion block to enhance the usage of side information, and a novel semantic segmentation network named modified TrSeg to generate better semantic segmentation maps. The experimental results well demonstrated the effectiveness of our proposed framework. SICMH achieves the same image classification and semantic segmentation accuracy as the existing traditional or learning-based multi-task ICM frameworks using the lowest bitrate. For the image reconstruction task, the proposed SICMH achieved the same PSNR as existing learning-based multi-task hybrid ICM frameworks and the traditional image codec BPG again with the lowest bitrate.

Keywords: Image classification, Image coding for machines, Image compression, Semantic segmentation, Side information

1 Introduction

In recent years, the rise of the Internet of Things and smart cities has spawned the deployment of a large number of monitoring facilities around traffic lights, communities, and shopping malls. People even buy affordable and practical monitoring devices and install them in their rooms to monitor the infants, the elderly or pets remotely in real time in order to prevent emergencies. In addition, vehicle intelligence is beginning to emerge, and vehicle cameras need to take a large number of pictures and transmit them to the backend server. These conditions result in a large amount of image data being transmitted over the network, which consumes large network bandwidth. In order to reduce the amount of transmitted data, we generally perform lossy compression on the image and we will obtain a compressed file with a smaller data volume for

© The Author(s) 2024. **Open Access** This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by-nc-nd/4.0/.

transmission. The above process is completed by an image codec. Traditional image codecs include JPEG [1], J2K [2], BPG [3], etc., which compress images according to a set of fixed algorithms or formulas. With the advancement of machine learning and deep learning, learning-based image codec has emerged, such as the hyperprior codec [4] and coarse-to-fine codec [5].

The processing of decoded images is another problem because they are extremely large in number and require further analysis to complete visual recognition tasks such as image classification and semantic segmentation. The emergence of deep learning-based visual recognition networks has greatly alleviated this problem. For example, ResNet50 [6] proposed the application of residual bottleneck layer in image classification for the first time, and ViT [7] applied the concept of transformer to image classification for the first time. Besides, magnetic resonance imaging (MRI) images in modern medicine increasingly rely on semantic segmentation technology to more accurately identify tumors in the brain or other parts of the body. When a large number of images need to be transmitted for visual recognition tasks, ICM comes into being.

ICM is one kind of frameworks that can compress images specifically for machine vision tasks like image recognition and semantic segmentation. An ICM framework is generally composed of an image codec and a task network. When the image codec is traditional, the codec encoder encodes the image into a binary file. When the codec is learning-based, the codec encoders first compress the image into a high-dimensional but small-volume tensor which we call latent representation or compressed feature, and then the latent representation is further encoded by an arithmetic encoder into a binary file. After the binary file arrives at the decoder side, it will be decoded into the image and be input into the visual recognition network.

Currently, in order to perform multiple machine visual recognition tasks, many ICM frameworks first compress and decompress images, and then send the decompressed images to different task networks. However, the training of these frameworks is often based on the metrics of human eye such as PSNR instead of machine task metrics such as the recognition loss function, which will cause the decoded images not only to lose much key information for visual recognition tasks, but also to contain huge amount of redundancy. Some other frameworks try to use the side information to do the visual recognition task to save the bandwidth, such as SIIC [8]. However, SIIC is single-task and has no ability to do other visual recognition tasks.

In order to solve the above problems, we propose the side information-driven image coding for hybrid machine–human vision (SICMH) framework to maximize the retention of key information useful for image classification and semantic segmentation while satisfying the image reconstruction task. Our contributions are as follows:

- We propose a hybrid machine–human ICM framework SICMH, which can perform image classification, semantic segmentation, coarse image reconstruction, and fine image reconstruction.
- We propose a multi-scale feature fusion block, which is located on the decoder side of SICMH. The multi-scale feature fusion block can make full use of side information to achieve better rate-accuracy performance on image classification and semantic segmentation.

- We propose a new semantic segmentation network Modified TrSeg based on TrSeg [9]. Our Modified TrSeg can achieve higher segmentation accuracy.

The structure of this paper is as follows: Section 2 describes various types of existing ICM frameworks, Section 3 introduces the method we proposed in details, Section 4 shows the experimental results of SICMH on semantic segmentation, image classification, and image reconstruction. Besides, Section 5 provides ablation studies to demonstrate the effectiveness of specific proposed modules. Section 6 discusses the potential performance trade-offs within SICMH and the complexity of SICMH. Section 7 concludes this paper.

2 Related works

In this section, we summarize existing popular ICM frameworks that can perform image classification or semantic segmentation, and divide them into three categories: traditional codec or learned codec conjoining task network, feature compression, and scalable coding.

2.1 Traditional codec or learned codec conjoining task network

Conjoining means the former part is concatenated with the later part. The structure of this category is shown in Fig. 1. The codecs used in this category are traditional image codecs such as BPG [3] or learning-based codecs. After the images are compressed, transmitted and decoded by a traditional or learned codec, the decoded images will be sent to different task networks. For example, J-FT T-FT [10] compresses and transmits the image $\mathbf{x}$ to the decoder and decodes it to $\widehat{\mathbf{x}}$. Then J-FT T-FT [10] allocates $\widehat{\mathbf{x}}$ to different task networks. Its primary task network is object detection, and its secondary task network is image classification on ImageNet1k [11] or semantic segmentation on Cityscapes [12]. In the J-FT T-FT framework, the whole network is jointly fine-tuned to improve the classification accuracy or segmentation accuracy. Similarly, Recognition-Aware Learned Image Compression [13] uses a hierarchical autoencoder [14] improved by Gaussian Mixture Models (GMM) as the codec and EfficientNet-B0 [15] as the image classification network. Another instance is the transformed images method [16], which first removes the task-irrelevant information in the image, then uses a codec to compress and transmit the processed image for final visual recognition tasks.

Within this category, some other frameworks need to first perform a semantic segmentation operation on the image to obtain the semantic map shown as Fig. 2. The codecs use the segmentation map to encode and decode the image. C128 [17], C64 [17], SC, inst., EDG+ [18], RL-ASC [19] and simplified RL-ASC [19] belong to this category. Taking RL-ASC [19] as an example, image $\mathbf{x}$ generates semantic map $\mathbf{s}$ through

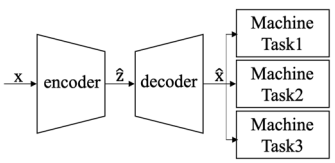

Fig. 1 The structure of traditional codec or learned codec conjoining task network

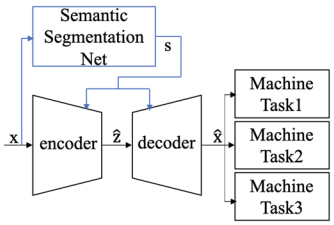

Fig. 2 The structure of traditional codec or learned codec conjoining task network with the assistance of semantic segmentation map

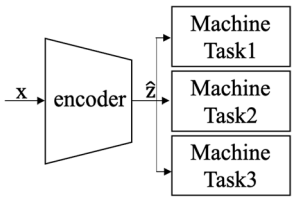

Fig. 3 The structure of feature compression

a pretrained semantic segmentation net, then **s** is split into M semantic sub-maps to make sure each sub-map only contains segmentation information belonging to its own class. At the same time, **x** enters a feature extraction net, which produces feature map **f**. Then **f**, respectively, multiplies with the M semantic sub-maps to obtain M sub-features. Each sub-feature is then assigned an optimal quantization level. At the decoder side, the decoder finally reconstructs the image with the help of **s**. In other words, the map **s** participates in the compression and decompression stages of the codec, which makes the decoded image contain richer semantic segmentation information. The bitrate calculated for this subcategory should include the bitrate of the semantic segmentation map **s** as well as the bitrate of the feature map **f**.

2.2 Feature compression

Figure 3 shows the architecture of this category. The encoder directly compresses image **x** into the semantic feature $\hat{\mathbf{z}}$, which is fed into the subsequent visual recognition tasks, without reconstructing image pixels. The frameworks in this category consists of one encoder and multiple decoders for different visual recognition tasks. For instance, the compressed representation method [20] inputs the compressed feature or latent representation $\hat{\mathbf{z}}$ into different subsequent task networks. This category also contains SPIC-Q [21], Post-SA [22] and SIIC [8].

2.3 Scalable coding

In this category, the codecs often consist of at least two transmission layers. The first layer is the base layer, which only transmits the bit stream of the partial compressed feature to complete simple tasks such as image classification or preview image generation. The second layer is the enhancement layer, which transmits the residue between the original image and the base layer information. On the decoder side, the decoded residue is added to the base layer information to handle more complex tasks such as image

reconstruction. For example, HMI-IC [23] is typical in this category. On the encoder side, image $\mathbf{x}$ enters the base layer to get sub-stream $\mathbf{f}_{\text{base}}$, which is fed into preview image generation network to get preview image $\mathbf{x}'$, then the residue between $\mathbf{x}$ and $\mathbf{x}'$ is compressed as enhancement sub-stream $\mathbf{f}_{\text{enhance}}$. $\mathbf{f}_{\text{enhance}}$ is then added to $\mathbf{x}'$ to produce decoded image $\hat{\mathbf{x}}$. DSSLIC [24] has a structure similar to HMI-IC [23], but like RL-ASC [19], it also needs to obtain a semantic segmentation map to assist the codec.

Our proposed SICMH belongs to scalable coding. SICMH's base layer can perform image classification, semantic segmentation, and coarse image reconstruction by purely exploiting the hyperprior information. Besides, SICMH's enhancement layer can transmit the residue information to complete fine image reconstruction.

3 Methods

In this section, we introduce our proposed image coding for hybrid machine–human vision system SICMH. It contains a base layer and an enhancement layer. As shown in Fig. 4, the black, blue and yellow parts above the red dotted line belong to the base layer. The red parts below the red dotted line belong to the enhancement layer. The base layer can achieve both machine tasks and human vision task: image classification, semantic segmentation, and coarse image reconstruction. The enhancement layer can achieve fine image reconstruction for human vision.

3.1 Coding for machine tasks

The base layer in SICMH uses only the side information to perform image classification, semantic segmentation, and coarse image reconstruction.

3.1.1 The encoders

The Encoder, Encoder_h_1 and Encoder_h_2 on the leftmost side of the SICMH framework process image $\mathbf{x}$ to produce the latent representation $\mathbf{y}$, and two layers of hyperprior information $\mathbf{h}_1$ and $\mathbf{h}_2$, which we call side information. Then the quantized side information $\widehat{\mathbf{h}}_1$ and $\widehat{\mathbf{h}}_2$ are transmitted to the decoder side. We use the following formulas to express the relationships among the encoders in Fig. 4:

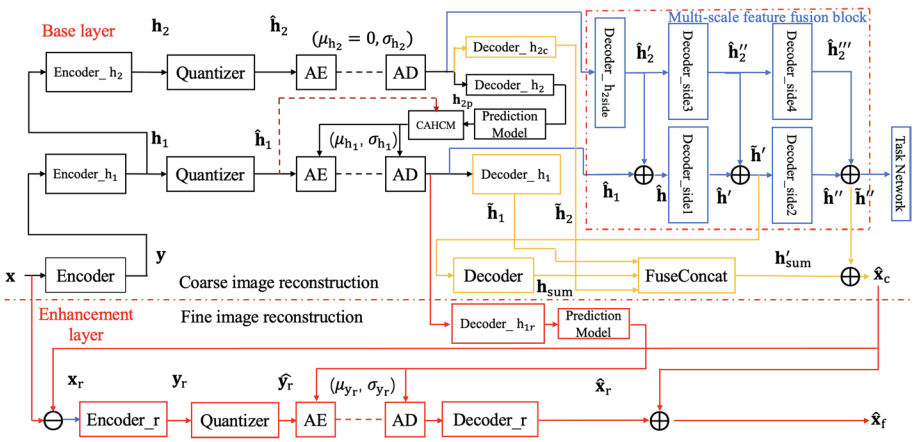

Fig. 4 The proposed SICMH's structure

$$\mathbf{y} = \text{Encoder}(\mathbf{x}), \tag{1}$$

$$\mathbf{h}_1 = \text{Encoder\_h}_1(\mathbf{y}), \tag{2}$$

$$\mathbf{h}_2 = \text{Encoder\_h}_2(\mathbf{h}_1). \tag{3}$$

Figure 5 (a)-(c) shows the detailed structures of the Encoder, Encoder_h_1 and Encoder_h_2. These three structures all contain 4 convolutional layers to extract features. Figure 5 (a) uses GDN for activation among different convolutional layers. Figure 5 (b) and (c) uses ReLU for activation.

3.1.2 Channel-wise auto-regressive hyper-prior context model

As shown in Fig. 4, we adopt a channel-wise auto-regressive hyper-prior context model (CAHCM) for the entropy coding of hyper prior $\widehat{\mathbf{h}}_1$ through better prediction of its distribution parameters. We divide $\widehat{\mathbf{h}}_1$ into several groups in the channel direction. When we code the kth group, we use the previously decoded $k-1$ groups along with the output of the prediction model which is $\mathbf{h_{2p}}$ as the input of this context model. Then CAHCM outputs the final predicted distribution parameters (μ_{h_1},σ_{h_1}) for the k th group.

3.1.3 The decoders and the task networks

At the decoder side, the proposed multi-scale feature fusion block further processes the side information, as shown in Fig. 4 blue part. Compared with SIIC [8], our multi-scale feature fusion block improves the way the decoders utilize $\widehat{\mathbf{h}}_1$ and $\widehat{\mathbf{h}}_2$. Its purpose is to combine the deeper layer side information $\widehat{\mathbf{h}}_2$ and shallower layer side information $\widehat{\mathbf{h}}_1$ to generate richer semantic feature for subsequent machine vision tasks.

The specific process is as follows. After $\widehat{\mathbf{h}}_1$ and $\widehat{\mathbf{h}}_2$ arrive at the decoder side, $\widehat{\mathbf{h}}_2$ is processed by Decoder_h_{2side} to generate $\widehat{\mathbf{h}}_2'$. $\widehat{\mathbf{h}}_2'$ and $\widehat{\mathbf{h}}_1$ are fused to form $\widehat{\mathbf{h}}$, then $\widehat{\mathbf{h}}_2'$ and $\widehat{\mathbf{h}}$ pass through Decoder_side3 and Decoder_side1 to generate $\widehat{\mathbf{h}}_2''$ and $\widehat{\mathbf{h}}'$. The sum of $\widehat{\mathbf{h}}_2''$ and $\widehat{\mathbf{h}}'$ is $\tilde{\mathbf{h}}'$, which is then processed by Decoder_side2 to obtain $\widehat{\mathbf{h}}''$. $\widehat{\mathbf{h}}_2''$ is processed by Decoder_side4 to obtain $\widehat{\mathbf{h}}_2'''$. Finally, the sum of $\widehat{\mathbf{h}}''$ and $\widehat{\mathbf{h}}_2'''$ is fed to the task network.

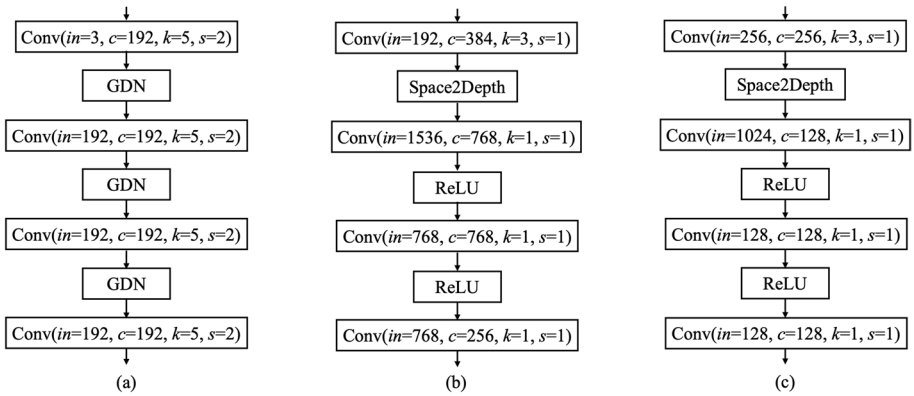

Fig. 5 Different encoder structures: **a** is the structure of Encoder and Encoder_r, **b** is the structure of Encoder_h_1, **c** is the structure of Encoder_h_2. Conv: convolutional layer, *in*: input channel number, *c*: output channel number, *k*: kernel size, *s*: stride. GDN is Generalized Divisive Normalization [25]. Space2Depth [5] increases the channels by a factor of 4 while reducing the height and width by half

This structure extracts multiple scales of features from both $\widehat{\mathbf{h}}_1$ and $\widehat{\mathbf{h}}_2$ with feature fusion at each scale, which allows SICMH to have better performance in all mentioned tasks. We use the following formulas to express the relationships among the decoders in Fig. 4:

$$\widehat{\mathbf{h}}_2' = \text{Decoder\_h}_{2\text{side}}(\widehat{\mathbf{h}}_2), \tag{4}$$

$$\widehat{\mathbf{h}} = \widehat{\mathbf{h}}_1 + \widehat{\mathbf{h}}_2', \tag{5}$$

$$\widehat{\mathbf{h}}_2'' = \text{Decoder\_side3}(\widehat{\mathbf{h}}_2'), \tag{6}$$

$$\widehat{\mathbf{h}}' = \text{Decoder\_side1}(\widehat{\mathbf{h}}), \tag{7}$$

$$\tilde{\mathbf{h}}' = \widehat{\mathbf{h}}_2'' + \widehat{\mathbf{h}}', \tag{8}$$

$$\widehat{\mathbf{h}}'' = \text{Decoder\_side2}(\tilde{\mathbf{h}}'), \tag{9}$$

$$\widehat{\mathbf{h}}_2''' = \text{Decoder\_side4}(\widehat{\mathbf{h}}_2''), \tag{10}$$

$$\tilde{\mathbf{h}}'' = \widehat{\mathbf{h}}_2''' + \widehat{\mathbf{h}}''. \tag{11}$$

After $\tilde{\mathbf{h}}''$ is created, it will be sent to different machine vision task networks. Figure 6 (a) shows Decoder_h$_{2\text{side}}$. Figure 6 (b) shows Decoder_side1 and Decoder_side3. Figure 6 (c) shows Decoder_side2 and Decoder_side4. These three structures all contain 4 transposed convolutional layers. Figure 6 (a) and (b) uses ReLU for activation, while Fig. 6 (c) uses IGDN for activation among different transposed convolutional layers.

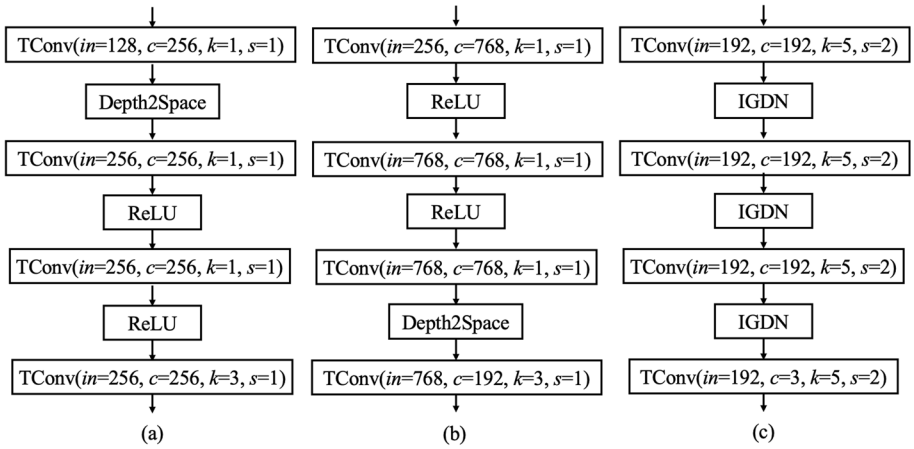

Fig. 6 Different decoder structures: **a** is the structure of Decoder_h$_2$, Decoder_h$_{2c}$ and Decoder_h$_{2\text{side}}$, **b** is the structure of Decoder_h$_1$, Decoder_h$_{1r}$, Decoder_side1, and Decoder_side3, **c** is the structure of Decoder, Decoder_r, Decoder_side2, and Decoder_side4. TConv: transposed convolutional layer, *in*: input channel number, *c*: output channel number, *k*: kernel size, *s*: stride. IGDN is Inverse Generalized Divisive Normalization [25]. Depth2Space [5] reduces the channels by a factor of 4 while doubling the height and width

For the segmentation task, we concatenate SICMH and the Modified TrSeg network, which we call SICMH conjoining Modified TrSeg. In the first training stage, all encoder, decoder and Modified TrSeg parameters participate in the training process. First, the pretrained coarse-to-fine framework [5] provides the pretrained Encoder, Encoder_h_1, Encoder_h_2 and Decoder_h_2. Second, we train the Modified TrSeg as mentioned in Section 5.1. Third, we concatenate the encoders, the proposed multi-scale feature fusion block, and Modified TrSeg to form the ICM framework for semantic segmentation task and train the model end-to-end using the following loss function:

$$loss_{1st\_segmentation} = \lambda \times R_{side} + loss_{trseg}, \tag{12}$$

where $loss_{trseg}$ is the cross-entropy loss between the modified TrSeg predicted segmentation map and the ground-truth map, R_{side} is the sum of the bitrates used to encode $\widehat{\mathbf{h}}_1$ and $\widehat{\mathbf{h}}_2$, and λ is the hyper parameter that controls the balance of these two items. In the second training stage, in order to further improve the mIoU and perform semantic segmentation on multi-scale resolutions of Cityscapes images [12], we freeze the parameters in the black part of Fig. 4 and only fine-tune the multi-scale feature fusion block and the Modified TrSeg net using the following loss function:

$$loss_{2nd\_segmentation} = loss_{trseg}. \tag{13}$$

For the classification task, we concatenate SICMH and the task network ViT, which we call SICMH conjoining ViT. The parameters of Encoder, Encoder_h_1, Encoder_h_2, Decoder_h_2 and Prediction Model are all from the first training stage in SICMH conjoining modified TrSeg, while the pretrained ViT model is from [7]. We freeze the parameters in the black part and only fine-tune the parameters of the multi-scale feature fusion block and ViT in Fig. 4. The loss function is as follows:

$$loss_{classification} = loss_{vit}, \tag{14}$$

where $loss_{vit}$ is the cross-entropy loss between the predicted labels and the ground-truth class labels. The bitrate for this task is R_{side}.

What needs to be emphasized is, for the semantic segmentation task and the image classification task, SICMH does not need to reconstruct the image.

3.1.4 Modified TrSeg

For the segmentation task, we propose a novel semantic segmentation network Modified TrSeg as shown in Fig. 7. The upper part of Fig. 7 is the original TrSeg network, which extracts feature map **f3D** and produces the semantic segmentation map **S**. We modify the TrSeg network by adding extra components as shown in the lower part of Fig. 7 with the red dotted box. Figure 8 shows the details of the extra components: how modified TrSeg multiplies **f3D** and **S** to get a new feature map, which is used to perform the TrSeg operations again.

On the one hand, in the original TrSeg [9], as shown in the upper part of Fig. 7, the image is first processed by backbone ResNet101 v2 to generate **f3D**, and the number of channels is $C = 512$, then **f3D** performs average pooling through the multi-scale pooling module (MSPool). The output of MSPool is flattened as $\mathbf{g_0}$. At the same time, **f3D** is also

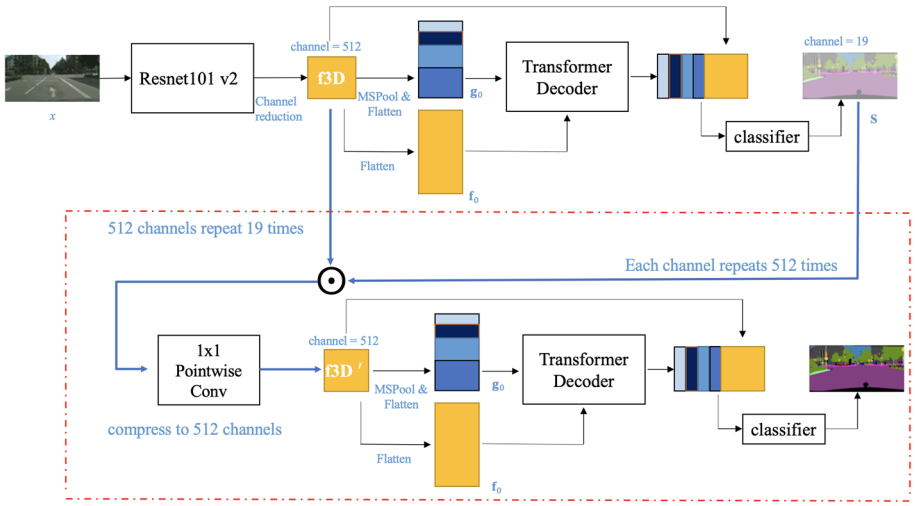

Fig. 7 The modified TrSeg structure

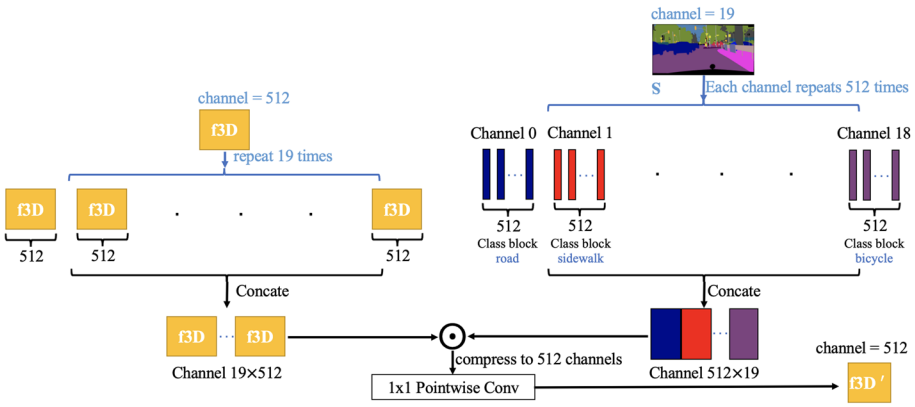

Fig. 8 The details of the point product combination of **f3D** and **S** (red dotted box of Fig. 7)

flattened as $\mathbf{f_0}$, then $\mathbf{g_0}$ and $\mathbf{f_0}$ are used as the source and target inputs of the transformer decoder. Finally, the output of the transformer decoder is stitched together with **f3D** to get the semantic segmentation map **S** through a classifier.

On the other hand, our Modified TrSeg improves the original TrSeg by recombining **S** with **f3D** as shown in Fig. 7, and the details are shown in Fig. 8. We repeat each channel of **S** 512 times, so that every block of 512 channels corresponds to only one class in Cityscapes [12], which we call class block. We then multiply **f3D** with each class block to split **f3D** into sub-feature maps. Similar to RL-ASC [19], each sub-feature map only focuses on the corresponding class feature. The multiply operation is inspired by Mask R-CNN [26], which mentioned that the two-category classification is easier than multi-category classification, because in two-category classification, we only need to distinguish whether the information belongs to the corresponding category, but in multi-category classification, we need to divide the information into different categories, which is more difficult and less accurate. Finally, the sub-feature maps are fed into a point-wise convolution layer to obtain $\mathbf{f3D}'$, which is further processed to generate the new segmentation map.

3.2 Coding for coarse image reconstruction

The base layer can perform not only multiple machine vision tasks, but also coarse image reconstruction by only utilizing the hyperpriors. The base layer for coarse image reconstruction is the yellow part shown in Fig. 4. We leverage several additional modules, namely Decoder, Decoder_h$_1$, Decoder_h$_{2c}$ and FuseConcat. In Fig. 4, $\widetilde{\mathbf{h}}'$ enters the Decoder to produce $\mathbf{h}_{\text{sum}}$, while $\widehat{\mathbf{h}}_1$ and $\widehat{\mathbf{h}}_2$ are processed by Decoder_h$_1$ and Decoder_h$_{2c}$, respectively, to get $\tilde{\mathbf{h}}_1$ and $\tilde{\mathbf{h}}_2$. Then $\mathbf{h}_{\text{sum}}$, $\tilde{\mathbf{h}}_1$ and $\tilde{\mathbf{h}}_2$ are sent to FuseConcat to generate $\mathbf{h}'_{\text{sum}}$. Finally, $\mathbf{h}'_{\text{sum}}$ and $\tilde{\mathbf{h}}''$ are fused to obtain the coarse reconstructed image $\widehat{\mathbf{x}}_c$. The equations below describe how the feature flows for the coarse image reconstruction task:

$$\tilde{\mathbf{h}}_1 = \text{Decoder\_h}_1(\widehat{\mathbf{h}}_1), \tag{15}$$

$$\tilde{\mathbf{h}}_2 = \text{Decoder\_h}_{2c}(\widehat{\mathbf{h}}_2), \tag{16}$$

$$\mathbf{h}_{\text{sum}} = \text{Decoder}(\tilde{\mathbf{h}}'), \tag{17}$$

$$\mathbf{h}'_{\text{sum}} = \text{FuseConcat}(\mathbf{h}_{\text{sum}}, \tilde{\mathbf{h}}_1, \tilde{\mathbf{h}}_2), \tag{18}$$

$$\widehat{\mathbf{x}}_c = \tilde{\mathbf{h}}'' + \mathbf{h}'_{\text{sum}}. \tag{19}$$

Figure 6 (a) shows Decoder_h$_2$ and Decoder_h$_{2c}$. Figure 6 (b) shows Decoder_h$_1$. Figure 6 (c) shows the Decoder. Figure 9 shows the detailed structure of the Prediction Model in Fig. 4. The neural network layers are shown in the black boxes, among them are the output shapes. Figure 10 shows the internal structure of FuseConcat in Fig. 4, where Conv represents convolution, TConv represents transposed convolution.

The side information bits come from SICMH conjoining Modified TrSeg mentioned in Section 3.1.3. The parameters in the black part of SICMH shown in Fig. 4 are exactly the same for three tasks: semantic segmentation, image classification, and coarse image reconstruction. To be more precise, all parameters of Encoder, Encoder_h$_1$, Encoder_h$_2$, Decoder_h$_2$ and Prediction Model remain unchanged when the system switches among these three tasks. We only train the parameters of the multi-scale feature fusion block (Fig. 4 blue part), Decoder, Decoder_h$_1$, Decoder_h$_{2c}$ and FuseConcat (Fig. 4 yellow part). The loss function is as follows:

$$loss_{coarse\_image\_reconstruction} = loss_{MSE}, \tag{20}$$

where $loss_{MSE}$ is the mean squared error between the original image $\mathbf{x}$ and the coarse reconstructed image $\widehat{\mathbf{x}}_c$.

3.3 Coding for fine image reconstruction

The image reconstructed from only the side information could be blurry or unclear, because the information extracted for machine vision tasks provides limited information to reconstruct images for human eyes. In order to bridge the information gap between human eyes and machine vision tasks, and to generate images that better meet the needs of human

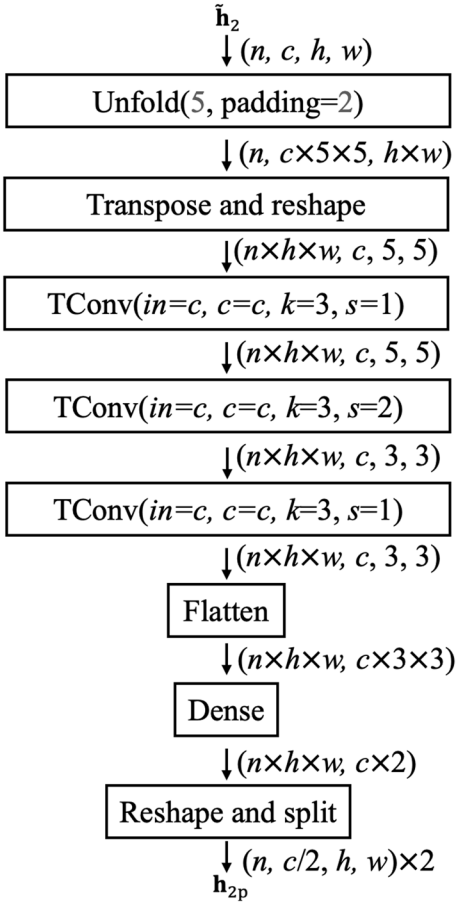

Fig. 9 The internal structure of the Prediction Model in Fig. 4. *n*: batch size, *c*: channel, *h*: height of the tensor, *w*: width of the tensor. Unfold(5) [5] is used to increase the channel by 25 times

eyes, in this section, we propose the enhancement layer, which codes and transmits the residue between the original image and the coarse reconstructed image $\widehat{\mathbf{x}}_c$ for fine image reconstruction.

The proposed enhancement layer is shown in Fig. 4, below the red dot line, where $\mathbf{x}_r$ is the residue between the original image $\mathbf{x}$ and the coarse reconstructed image $\widehat{\mathbf{x}}_c$. It is first encoded by Encoder_r to produce $\mathbf{y}_r$, then the quantized $\widehat{\mathbf{y}}_r$ is transmitted to Decoder_r for further amplification and restoration. Finally, the fine reconstructed image $\widehat{\mathbf{x}}_f$ is completed by summing up $\widehat{\mathbf{x}}_c$ and $\widehat{\mathbf{x}}_r$. The equations below describe how the feature flows in the fine image reconstruction framework:

$$\mathbf{x}_r = \mathbf{x} - \widehat{\mathbf{x}}_c, \tag{21}$$

$$\mathbf{y}_r = \text{Encoder\_r}(\mathbf{x}_r), \tag{22}$$

$$\widehat{\mathbf{x}}_r = \text{Decoder\_r}(\widehat{\mathbf{y}}_r), \tag{23}$$

$$\widehat{\mathbf{x}}_f = \widehat{\mathbf{x}}_r + \widehat{\mathbf{x}}_c. \tag{24}$$

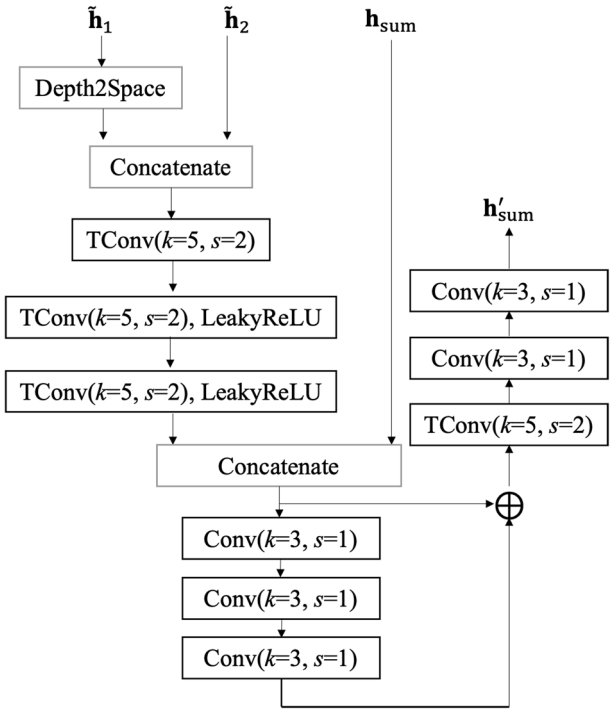

Fig. 10 The internal structure of FuseConcat in Fig. 4. *k*: kernal size and *s*: stride size. Concatenate is to concatenate two tensors in the channel direction

Fig. 5 (a) shows the structure of Encoder_r. Figure 6 (c) shows the structure of Decoder_r. For entropy coding, according to [5] we assume that $\widehat{\mathbf{y}}_{\mathrm{r}}$ follows a normal distribution. The parameters $(\mu_{\mathrm{y_r}},\sigma_{\mathrm{y_r}})$ are predicted by $\widehat{\mathbf{h}}_1$ through Decoder_$\mathrm{h_{1r}}$ and Prediction Model. Figure 6 (b) shows the structure of Decoder_$\mathrm{h_{1r}}$.

We only fine-tune the parameters of Decoder_$\mathrm{h_{1r}}$, Encoder_r, and Decoder_r. The rest parameters are all from the coarse image reconstruction task and are frozen. The loss function is as follows:

$$loss_{fine\_image\_reconstruction} = loss_{MSE} + \lambda \times R, \tag{25}$$

where $loss_{MSE}$ is the mean squared error between the original image $\mathbf{x}$ and the fine reconstructed image $\widehat{\mathbf{x}}_{\mathrm{f}}$, R is the sum of the bitrates used to encode $\widehat{\mathbf{h}}_1$, $\widehat{\mathbf{h}}_2$ and $\widehat{\mathbf{x}}_{\mathrm{r}}$, and λ is the hyper parameter that controls the balance of these two items.

4 Results

This section shows the experimental details and results of our proposed framework on semantic segmentation, image classification and image reconstruction.

4.1 Semantic segmentation

For semantic segmentation, the data set is Cityscapes [12], where the training set contains 2,975 images and the verification set contains 500 images which is used as our testing set. There are 19 semantic classes. The performance of semantic segmentation is evaluated by the mean intersection over union (mIoU). The higher the mIoU, the better the performance.

The original image resolution is 1024×2048. In addition, we also experiment on images with resolutions of 512×1024 and 256×512. Only when the resolution is 1024×2048, the proposed SICMH conjoining Modified TrSeg needs to be trained in two stages as described in Section 3.1.3. For the other two image resolutions, our method only fine-tunes the decoders and the Modified TrSeg.

When the resolution is 1024×2048, in the first training stage, the learning rate is 10^{-5} and the model is trained for 60 epochs. Then we change the learning rate to 10^{-6} and continue training for 30 epochs. In the second training stage, we freeze the parameters in the black part in Fig. 4 and only fine-tune the decoders and the Modified TrSeg for 30 epochs with a learning rate of 10^{-6}. When the input image resolution is 512×1024 or 256×512, the model parameters trained for resolution 1024×2048 are used as the initial parameters, then we fine-tune the parameters of the decoders and the Modified TrSeg for 40 epochs with a learning rate of 10^{-6}.

As shown in Fig. 11, when the resolution is 1024×2048, our results are compared with the experimental results of [10, 27–30]. These methods belong to traditional codec or learned codec conjoining task network. In particular, the result of JPEG AI is obtained by compressing the test images using JPEG AI, then feeding the decoded images into the Modified TrSeg trained in Section 5.1 to obtain segmentation results. It can be clearly seen that our proposed method only uses about 28% of the bitrate of J-FT T-FT [10], about 20% of the bitrate of VVC [28], and about 18% of the bitrate of HEVC [29] to achieve the same mIoU (≈ 51%). When the bitrate is around 0.11 bits per pixel (bpp), our method achieves an mIoU of more than 64% while VVC [28] achieves an mIoU of about 53.5%, HEVC [29] achieves an mIoU of about 51%, cheng20 [30] and J-FT T-FT [10] only achieves an mIoU of about 48%. At mIoU=61%, our method uses less than half of the bitrates required by JPEG AI [27].

As shown in Fig. 12, when the resolution is 512×1024, our results are compared with the experimental results of [2, 3, 8, 17, 18, 32]. These methods belong to the category of

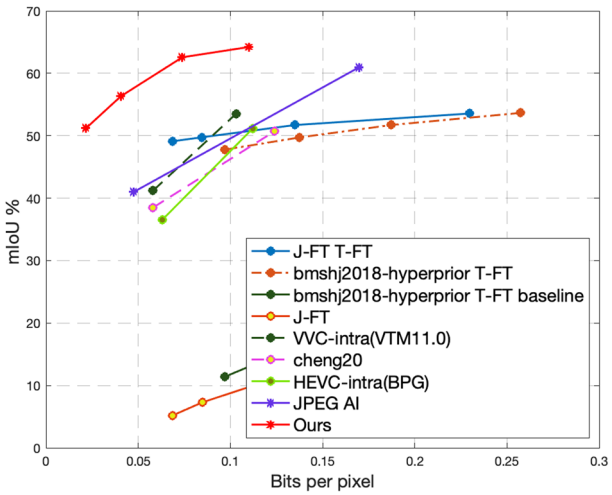

Fig. 11 Comparison of the semantic segmentation results of different methods on the Cityscapes validation set with a resolution of 1024×2048, including JPEG AI [27], J-FT T-FT [10], bmshj2018-hyperprior T-FT [10], bmshj2018-hyperprior T-FT baseline [10], J-FT [10], VVC-intra [28], HEVC-intra [29], cheng20 [30] and ours. The results of VVC-intra [28], HEVC-intra [29], cheng20 [30] are taken from [31]

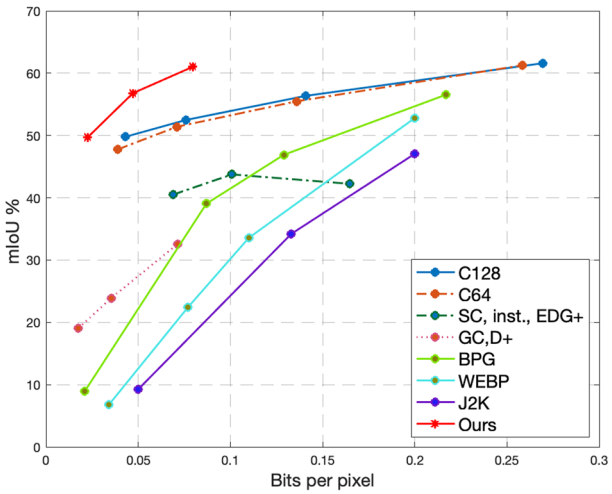

Fig. 12 Comparison of the semantic segmentation results of different methods on the Cityscapes validation set with a resolution of 512×1024, including C128 [17], C64 [17], SC, inst., EDG+ [18], GC,D+ [18], WEBP [32], J2K [2], BPG [3] and ours

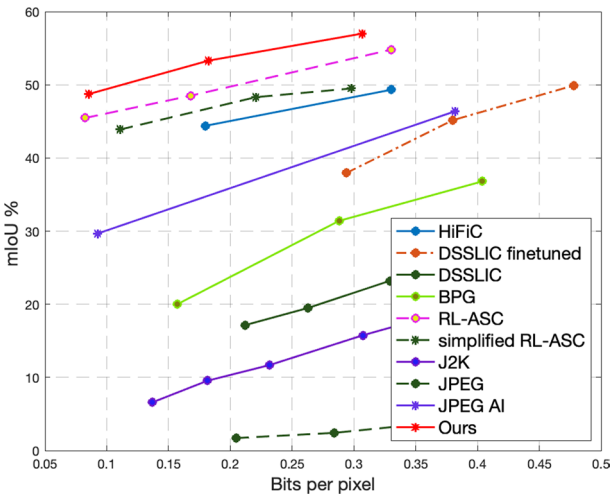

Fig. 13 Comparison of the semantic segmentation results of different methods on the Cityscapes validation set with a resolution of 256×512, including HiFiC [33], DSSLIC fine-tuned [24], DSSLIC [24], BPG [3], RL-ASC [19], simplified RL-ASC [19], J2K [2], JPEG [1], JPEG AI [27] and ours

traditional codec or learned codec conjoining task network. At the same mIoU ≈ 50% level, our method uses about 0.02 bpp, which is 45% less than the second best performing C128 [17], and 60% less than the third best performing C64 [17]. At the same mIoU ≈ 61% level, our method uses 0.08 bpp, which is 68% less than C128 [17], and 67% less than C64 [17].

As shown in Fig 13, when the resolution is 256×512, our results are compared with the experimental results of [3, 8, 19, 24, 27, 33]. These methods belong to the category of traditional codec or learned codec conjoining task network. At the same mIoU ≈ 49% level, we use 0.085 bpp, which is 50% less than the second best performing RL-ASC [19] which is 0.18 bpp, and 74% less than the third best performing simplified RL-ASC [19] which

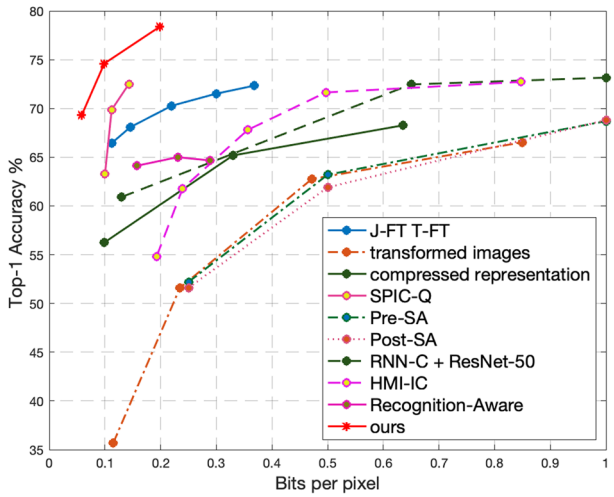

Fig. 14 Comparison of the results of different methods on the ImageNet1K [11] validation set, including SPIC-Q [21], Recognition-Aware [13], J-FT T-FT [10], transformed images [16], compressed representation [20], Pre-SA [22], Post-SA [22], RNN-C conjoining ResNet-50 [34], HMI-IC [23] and ours

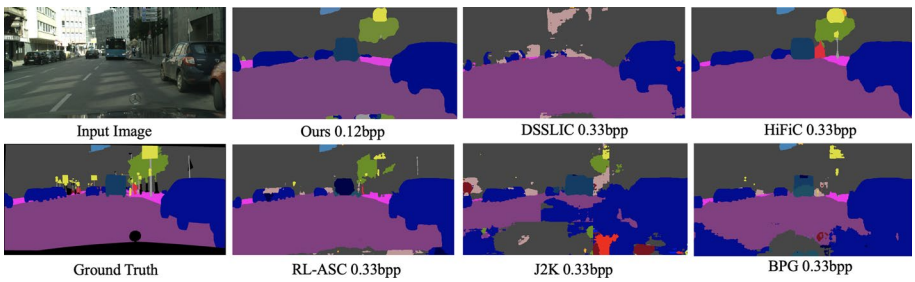

Fig. 15 The semantic maps on Cityscapes [12], the resolution is 256×512. RL-ASC [19], DSSLIC [24], J2K [2], HiFiC [33] and BPG [3] are from the paper [19]. They all use **0.33** bpp, while our SICMH uses **0.12** bpp

is 0.3 bpp. When the bitrate is 0.3 bpp, our method achieves more than 57% mIoU while RL-ASC [19] and HiFiC [33] achieves an mIoU of 55% and 49%, respectively.

It should be noted that the above-mentioned C128 [17], DSSLIC [24], inst., EDG+ [18] and RL-ASC [19] all need the prior semantic segmentation map for both encoding and decoding process, but our proposed framework does not need to send any kind of semantic segmentation map information to the decoder.

In order to demonstrate the effect clearly, we visually compare our semantic segmentation results with other methods, as shown in Fig. 15. It can be clearly seen that even though our framework uses less than half of the bitrate of other methods, our segmentation result is no worse than learning-based frameworks RL-ASC [19], DSSLIC [24] and traditional frameworks such as J2K [2] and BPG [3].

4.2 Image classification

For image classification, we use ImageNet1K [11] as the dataset. The ImageNet dataset has 1,000 classes, the training set contains 1.28M images, and the validation part contains 50,000 images, which is used as our testing set. The ViT version we choose has a patch window size of 24×24 and an input size of 384×384. The batch size is set

as 64. The learning rate is 10^{-4} for the first 3 epochs, 10^{-5} for the next 3 epochs and 10^{-6} for the last 3 epochs. The optimizer is the Adam optimizer [35].

As shown in Fig. 14, we compare the top-1 image classification accuracy of our proposed method with [8, 10, 13, 16, 20–23, 34]. These methods belong to the category of learned codec conjoining task network. In one word, our method can use less bitrate than others to achieve better classification accuracy. Our method is able to achieve a top-1 accuracy of around 75% using 0.1 bpp.

4.3 Coarse image reconstruction

As shown in Fig. 4, for coarse image reconstruction on Cityscapes [12] of resolution 256×512, we initialize the base layer network with trained SICMH conjoining Modified TrSeg model and fine-tune the multi-scale feature fusion block, Decoder, Decoder_h_1, Decoder_h_{2c} and FuseConcat. The learning rate is 10^{-4} for the first 600 epochs, 5^{-5} for the next 600 epochs and 10^{-6} for the final 600 epochs. The optimizer is the Adam optimizer [35].

As shown in Fig. 16, we compare our PSNR performance on Cityscapes [12] with [1–3, 19, 24, 33]. Among these methods, JPEG [1], J2K [2] and BPG [3] are traditional codecs and the rest of them are learning-based codecs. When the PSNR is 26 dB, our coarse reconstruction method only uses 0.08 bpp, which is 40% of DSSLIC [24] and JPEG [1], and 25% of RL-ASC [19].

Since the coarse image reconstruction uses side information specifically designed for image classification and semantic segmentation, the decoded image quality is limited. In the next subsection, we will demonstrate that the proposed fine image reconstruction module further improves image quality with the assistance of the coarse image reconstruction result.

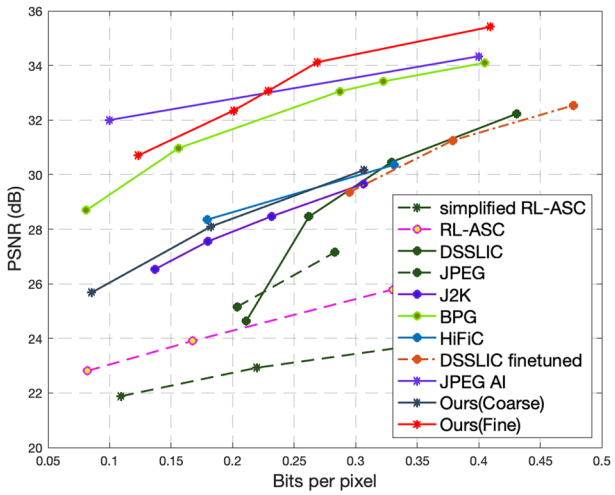

Fig. 16 Comparison of the PSNR results of different methods on the Cityscapes validation set with resolution of 256×512, including HiFiC [33], DSSLIC fine-tuned [24], DSSLIC [24], RL-ASC [19], simplified RL-ASC [19], JPEG [1], BPG [3], J2K [2], JPEG AI [27] and our coarse and fine image reconstruction methods

4.4 Fine image reconstruction

As shown in Fig. 4, for fine image reconstruction on Cityscapes [12] of resolution 256× 512, we need not only the bits of $\widehat{\mathbf{x}}_c$ from the base layer, but also the residue $\widehat{\mathbf{x}}_r$ from the enhancement layer. In fine image reconstruction, the required bitrates for side information are fixed at 0.08 bpp. At this rate, SICHM achieves a semantic segmentation mIoU of 49% on the Cityscapes images (256×512 resolution), and a top-1 accuracy of 69% in image classification. We freeze the parameters in the black, blue and yellow parts in Fig. 4 and only train the enhancement layer which is the red part below the red dotted line in Fig. 4. The learning rate is 10^{-4} for the first 200 epochs and 10^{-5} for another 200 epochs. The optimizer is the Adam optimizer [35].

We compare the PSNR performance of different methods on Cityscapes [12]. As shown in Fig. 16, at 0.4 bpp, our fine reconstruction method could achieve a PSNR of over 35 dB, which is more than 1 dB higher than BPG [3] and JPEG AI [27]. Comparing with the rest of the methods shown in Fig. 16, our method could achieve the highest PSNR with the same bitrate, except that in the lower bitrate range (0.1 bpp $\sim$ 0.23 bpp), our PSNR performance is not as good as JPEG AI [27]. This is because, to ensure that SICMH's performance in image classification and semantic segmentation is not degraded, we need to allocate 0.08 bpp to the base layer. Consequently, SICMH has very limited bitrates available for the enhancement layer in this lower bitrate range, resulting in reconstructed image quality that is worse than JPEG AI [27].

As shown in Figs. 17 and 18, in order to more intuitively observe the difference in human vision between our method and other frameworks, we selected two sample images from Cityscapes [12] for comparison. At the same bitrate of 0.3 bpp, in Fig. 17, our proposed fine image reconstruction achieves a PSNR of 33.48 dB, which is 0.5 dB higher than BPG [3]. In Fig. 18, our method achieves a PSNR of 33.71 dB, which is 0.74 dB higher than BPG [3]. As we can see, in both figures, the images

Fig. 17 At a bitrate of 0.3 bpp, the reconstructed images of Cityscapes [12]. The PSNR values are beneath the images

Fig. 18 At a bitrate of 0.3 bpp, the reconstructed images of Cityscapes [12]. The PSNR values are beneath the images

Table 1 Semantic segmentation mIoU of TrSeg and modified TrSeg

Method	mIoU
TreSeg	74.30%
Modified TreSeg	**76.44%**

decoded by our proposed fine image reconstruction have sharper edges. The colors are closer to the original images than BPG [3], and the decoded images are clearer and less blurry than BPG [3].

5 Ablation studies

5.1 Experiments on the modified TrSeg and the original TrSeg

To demonstrate the effectiveness of our proposed modified TrSeg for semantic segmentation, we conducted ablation experiments comparing the modified TrSeg and the original TrSeg [9].

Following the training procedure of TrSeg [9], we train TrSeg and modified TrSeg from scratch on Cityscapes [12] with a resolution of 1024×2048. The Cityscapes training split is used as the training set, and we evaluate the mIoU on the Cityscapes validation split. The batch size is 4. The initial learning rate is 5×10^{-5}. The weight decay is 10^{-4}. TrSeg [9] is trained for 200 epochs. The upper part of the modified TrSeg shown in the upper part of Fig. 7 is first trained alone for 110 epochs, then the whole modified TrSeg shown in Fig. 7 is trained for another 90 epochs. The results are shown in Table 1. We observe that modified TreSeg improved the mIoU by 2.14%, indicating that recombining $\mathbf{S}$ with $\mathbf{f3D}$ as shown in Fig. 8 is helpful for semantic segmentation.

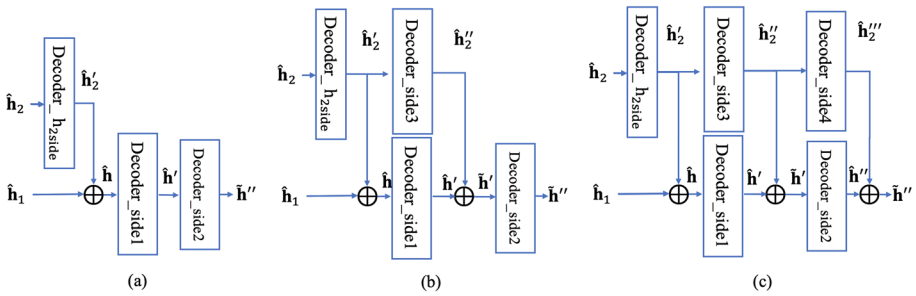

Fig. 19 Structure **a** is the adapter structure in SIIC [8], it only contains one fusion step. **b** Is the structure with two fusion steps by adding one extra decoder Decoder_side3 to structure (**a**). Structure **c** is the proposed multi-scale feature fusion block, which contains three fusion steps

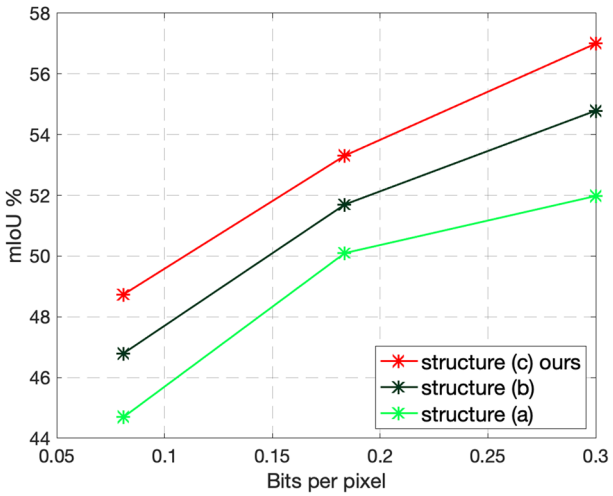

Fig. 20 The comparison of mIoU on Cityscapes 256×512 resolution images among structure (**a**), (**b**), and (**c**) ours in Fig. 19

5.2 The effectiveness of the multi-scale feature fusion block

To demonstrate the effectiveness of our proposed multi-scale feature fusion block in Section 3.1.3, in this section we conduct an ablation study. We compare three structures as shown in Fig. 19 (a)-(c). Each structure takes the entropy decoded side information $\widehat{\mathbf{h}}_1$ and $\widehat{\mathbf{h}}_2$ as the input, and outputs $\widetilde{\mathbf{h}}''$, which is fed into the task network. Three structures differ in the way they fuse the information from $\widehat{\mathbf{h}}_1$ and $\widehat{\mathbf{h}}_2$. We start from the adapter structure in SIIC [8], shown in Fig. 19 (a), which is the baseline to be compared. It scales up $\widehat{\mathbf{h}}_2$ by Decoder_h$_{\mathbf{2side}}$ which is then added to $\widehat{\mathbf{h}}_1$. This structure contains only one fusion step, and we call it as the one-scale feature fusion module. In order to better utilize $\widehat{\mathbf{h}}_1$ and $\widehat{\mathbf{h}}_2$, in Fig. 19 (b), we create the second structure using Decoder_side3 to further upscale $\widehat{\mathbf{h}}_2'$ and using one more fusion step. We call this structure as the two-scale feature fusion module. Similarly in Fig. 19 (c), we use Decoder_side4 to upscale $\widehat{\mathbf{h}}_2''$, and followed by one more fusion step to get our proposed multi-scale feature fusion module. It contains three fusion steps, which enables a strong utilization of the side information.

The comparison results are shown in Fig. 20 and Fig. 21. It can be observed that with each additional fusion step, the semantic segmentation mIoU on the Cityscapes dataset

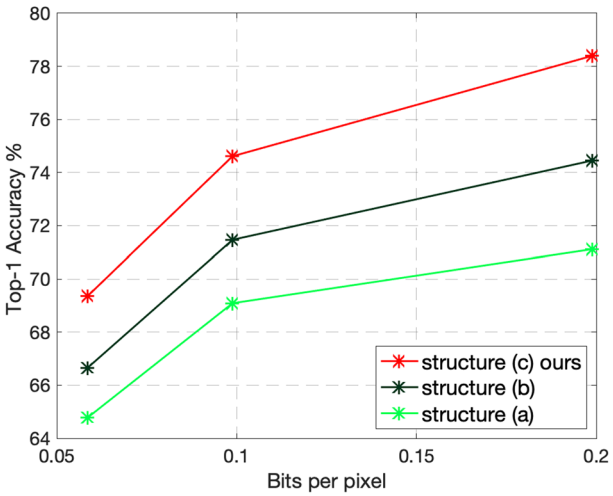

Fig. 21 The comparison of top-1 classification accuracy on ImageNet1K among structure (**a**), (**b**), and (**c**) ours in Fig. 19

and the top-1 image classification accuracy on the ImageNet1K dataset increase, indicating that the proposed multi-scale feature fusion block is essential for the effective utilization of side information.

6 Discussion

6.1 Performance trade-off between image reconstruction and machine tasks

Since our proposed framework is a scalable coding based hybrid machine human vision task architecture, it can perform multiple tasks. Inevitably, when the total bitrate is fixed, there will be a performance trade-off between the machine task and the human vision-oriented fine image reconstruction task. Generally speaking, the more bitrate allocated to the machine task, the better the machine task is performed. Correspondingly, this leaves less bitrate for the fine image reconstruction task, also known as the enhancement layer, resulting in its performance degradation.

In this section, we analyze the performance trade-off between image reconstruction and machine tasks. Under the same total bitrate budget, we allocate increasing bitrate to the base layer, accordingly the enhancement layer for fine image reconstruction has decreasing bitrate. With this decreasing enhancement layer bitrate, we evaluate the fine image reconstruction PSNR values.

As shown in Fig. 22, bits per pixel refers to the total bitrates, including both the base layer bitrate and the enhancement layer bitrate. We conducted three sets of experiments: (1) side=0.08, acc=69%, mIoU=49%; (2) side=0.18, acc=75%, mIoU=53%; and (3) side=0.30, acc=78%, mIoU=57%. We use set(side=0.08), set(side=0.18), and set(side=0.30) to represent them. Side refers to the bitrates allocated to side information, acc refers to the corresponding image classification accuracy on the ImageNet1K dataset, and mIoU indicates the semantic segmentation accuracy on the Cityscapes dataset with a resolution of 256×512.

It can be observed that when the total bitrates are fixed, the more bitrates allocated to side information, the better the performance in image classification and semantic

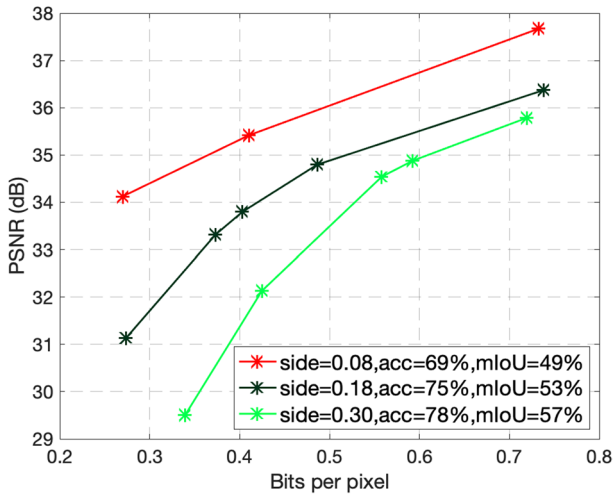

Fig. 22 Comparison of the fine image reconstruction PSNR results of SICMH with different bitrate of the side information, on the Cityscapes validation set with a resolution of 256×512

segmentation, but the worse the performance in image reconstruction. This phenomenon is more pronounced when the total bitrates are relatively low, such as when the total bitrate is 0.35 bpp. At this low bitrate, the enhancement layer bitrate for set(side=0.08), set(side=0.18) and set(side=0.30) are 0.27 bpp, 0.17 bpp and 0.05 bpp. The difference between these three enhancement layer bitrates is large, 0.27 bpp is 1.6 times of 0.17 bpp and 5.4 times of 0.05 bpp, therefore when the total bitrates are low the difference between the PSNR values is significant. In contrast, when the total bitrates are higher, the PSNR difference among these three sets becomes much smaller. For instance, when the total bitrate is 0.75 bpp, the enhancement layer bitrates for set(side=0.08), set(side=0.18) and set(side=0.30) are 0.67 bpp, 0.57 bpp, and 0.45 bpp. The difference between these three enhancement layer bitrates is small, 0.67 bpp is only 1.17 times of 0.57 bpp and 1.49 times of 0.45 bpp, therefore when the total bitrates are high the difference between the PSNR values is small.

6.2 Complexity analysis

This section discusses the model complexity of our proposed SICMH framework, and compare it with non-scalable coding approaches.

The proposed SICMH is a scalable coding architecture for hybrid machine–human vision tasks. It consists of four modules A, B, C, and D as shown in Fig. 23. The base layer includes A, B and C. It can perform semantic segmentation task and image classification task using modules A and B, and coarse image reconstruction task using modules A, B and C. The fine image reconstruction task requires modules A, B, C and D, and module D is the enhancement layer.

The training process is as follows. First, we train modules A and B for the semantic segmentation task, after which A is frozen. For image classification, module B is trained with module A frozen. Coarse image reconstruction requires training modules B and C, still with A frozen. For fine image reconstruction, module D is trained with the base layer modules A, B, and C frozen.

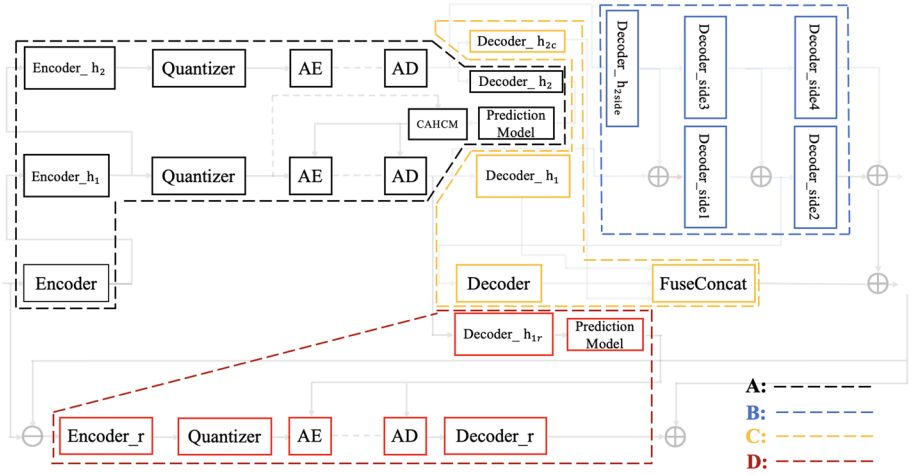

Fig. 23 Modules A, B, C and D in SICMH

Let the model parameter quantities or floating point operations (FLOPs) of A, B, C, and D be a, b, c, and d, respectively. If we train one separate model for each of the four tasks without module sharing of our proposed SICMH, then the parameters or FLOPs required to complete the machine tasks and image reconstruction tasks would be: $a + b$ for semantic segmentation, $a + b$ for image classification, $a + b + c$ for coarse image reconstruction, and $a + b + c + d$ for fine image reconstruction. This totals to $4a + 4b + 2c + d$. However, with our SICMH which uses module sharing, the corresponding parameters or FLOPs are: $a + b$, b, $b + c$, and d, totaling $a + 3b + c + d$. This reduces the overall model size and computational load by $3a + b + c$, greatly saving resources.

7 Conclusion

We proposed a scalable hybrid machine–human vision ICM framework SICMH. Its base layer is able to perform semantic segmentation, image classification and coarse image reconstruction tasks, and its enhancement layer is able to perform fine image reconstruction task. Our proposed framework SICMH adopts multi-scale feature fusion block to efficiently extract semantic information from hyperpriors to obtain better performance on semantic segmentation mIoU, image classification accuracy and coarse image reconstruction PSNR. As for fine image reconstruction, our SICMH transmits both the side information from the base layer and the residue information from the enhancement layer to make the restoration more in line with human eyes. In the future, we will investigate more advanced methods to address the trade-off between machine vision tasks and image reconstruction, especially for low bitrate range.

Acknowledgements
Not applicable.

Author contributions
ZZ: conceptualization, algorithm design, experiments, paper writing, paper revision; YL: conceptualization, paper revision, funding acquisition, project supervision; WP: conceptualization, paper revision. All authors took part in the discussion of the work described in this paper. All authors read and approved the final manuscript.

Funding
This work was supported in part by the National Science Foundation under Grant ECCS-2138635 and in part by the NVIDIA Academic Hardware Grant.

Availability of data and materials
The datasets generated and/or analyzed during the current study are available in the ImageNet1K repository, http://www.image-net.org/ [11], and in the Cityscapes repository, https://www.cityscapes-dataset.com/ [12].

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 29 February 2024 Accepted: 6 December 2024
Published online: 28 January 2025

References

1. G.K. Wallace, The jpeg still picture compression standard. IEEE transactions on consumer electronics **38**(1), xviii–xxxiv (1992)
2. D.S. Taubman, M.W. Marcellin, M. Rabbani, Jpeg 2000: image compression fundamentals, standards and practice. J. Electron. Imaging **11**(2), 286–287 (2002)
3. Bpg image format. https://bellard.org/bpg/. Accessed 02 october 2023
4. J. Ballé, D. Minnen, S. Singh, S.J. Hwang, N. Johnston, Variational image compression with a scale hyperprior. arXiv preprint arXiv:1802.01436 (2018)
5. Y. Hu, W. Yang, J. Liu, Coarse-to-fine hyper-prior modeling for learned image compression, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34 (2020), pp. 11013–11020
6. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in Proceedings of the IEEE conference on computer vision and pattern recognition (2016), pp. 770–778
7. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
8. Z. Zhang, Y. Liu, Side Information Driven Image Coding for Machines, in 2022 Picture Coding Symposium (PCS) (IEEE, 2022), pp. 193–197
9. Y. Jin, D. Han, H. Ko, Trseg: transformer for semantic segmentation. Pattern Recogn. Lett. **148**, 29–35 (2021)
10. L.D. Chamain, F. Racapé, J. Bégaint, A. Pushparaja, S. Feltman, End-to-end optimized image compression for multiple machine tasks. arXiv preprint arXiv:2103.04178 (2021)
11. J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in 2009 IEEE conference on computer vision and pattern recognition (IEEE, 2009), pp. 248–255
12. M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset for semantic urban scene understanding, in Proceedings of the IEEE conference on computer vision and pattern recognition (2016), pp. 3213–3223
13. M. Kawawa-Beaudan, R. Roggenkemper, A. Zakhor, Recognition-aware learned image compression. arXiv preprint arXiv:2202.00198 (2022)
14. J. Liu, G. Lu, Z. Hu, D. Xu, A unified end-to-end framework for efficient deep image compression. arXiv preprint arXiv:2002.03370 (2020)
15. M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in International conference on machine learning (PMLR, 2019), pp. 6105–6114
16. S. Suzuki, M. Takagi, K. Hayase, T. Onishi, A. Shimizu, Image pre-transformation for recognition-aware image compression, in 2019 IEEE International Conference on Image Processing (ICIP) (IEEE, 2019), pp. 2686–2690
17. D. Huang, X. Tao, F. Gao, J. Lu, Deep learning-based image semantic coding for semantic communications, in 2021 IEEE Global Communications Conference (GLOBECOM) (IEEE, 2021), pp. 1–6
18. E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, L.V. Gool, Generative adversarial networks for extreme learned image compression, in Proceedings of the IEEE/CVF International Conference on Computer Vision (2019), pp. 221–231
19. D. Huang, F. Gao, X. Tao, Q. Du, J. Lu, Toward semantic communications: deep learning-based image semantic coding. IEEE J. Selected Areas in Commun. **41**(1), 55–71 (2022)
20. R. Torfason, F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, L. Van Gool, Towards image understanding from deep compression without decoding. arXiv preprint arXiv:1803.06131 (2018)
21. N. Patwa, N. Ahuja, S. Somayazulu, O. Tickoo, S. Varadarajan, S. Koolagudi, Semantic-preserving image compression, in 2020 IEEE International Conference on Image Processing (ICIP) (IEEE, 2020), pp. 1281–1285
22. S. Luo, Y. Yang, Y. Yin, C. Shen, Y. Zhao, M. Song, DeepSIC: Deep semantic image compression, in Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25 (Springer, 2018), pp. 96–106
23. Z. Wang, F. Li, J. Xu, P.C. Cosman, Human-machine interaction-oriented image coding for resource-constrained visual monitoring in IOT. IEEE Int. Things J. **9**(17), 16181–16195 (2022)
24. M. Akbari, J. Liang, J. Han, DSSLIC: Deep semantic segmentation-based layered image compression, in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (IEEE, 2019), pp. 2042–2046
25. Ballé, Johannes and Laparra, Valero and Simoncelli, Eero P, End-to-end optimized image compression. arXiv preprint arXiv:1611.01704 (2016)
26. K. He, G. Gkioxari, P. Dollar, R. Girshick, Mask R-CNN, in Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)

27. JPEG AI. https://jpeg.org/jpegai. Accessed 15 June 2024
28. B. Bross, Y.K. Wang, Y. Ye, S. Liu, J. Chen, G.J. Sullivan, J.R. Ohm, Overview of the versatile video coding (VVC) standard and its applications. IEEE Trans. Circ. Syst. Video Technol. **31**(10), 3736–3764 (2021)
29. G.J. Sullivan, J.R. Ohm, W.J. Han, T. Wiegand, Overview of the high efficiency video coding (HEVC) standard. IEEE Trans. Circ. Syst. Video Technol. **22**(12), 1649–1668 (2012)
30. Z. Cheng, H. Sun, M. Takeuchi, J. Katto, Learned image compression with discretized gaussian mixture likelihoods and attention modules, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2020), pp. 7939–7948
31. R. Feng, X. Jin, Z. Guo, R. Feng, Y. Gao, T. He, Z. Zhang, S. Sun, Z. Chen, Image Coding for Machines with Omnipotent Feature Learning, in Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVII (Springer, 2022), pp. 510–528
32. An image format for the web. https://developers.google.com/speed/webp/. Accessed 02 october 2023
33. F. Mentzer, G.D. Toderici, M. Tschannen, E. Agustsson, High-fidelity generative image compression. Adv. Neural Inf. Processing Syst. **33**, 11913–11924 (2020)
34. M. Weber, C. Renggli, H. Grabner, C. Zhang, Observer dependent lossy image compression, in Pattern Recognition: 42nd DAGM German Conference, DAGM GCPR 2020, Tübingen, Germany, September 28–October 1, 2020, Proceedings 42 (Springer, 2021), pp. 130–144
35. D.P. Kingma, J. Ba, Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.