ヘテロジニアスコンピューティングを活用した圧縮ドメインにおける映像解析

早稲田大学　理工学術院総合研究所　孫鶴鳴

Compressed Domain Video Analysis Using Heterogeneous Computing

Heming Sun
Waseda Research Institute for Science and Engineering, Waseda University, Japan

I. Introduction

There is a growing number of images and are analyzed by machines rather than humans. This enables object detection, semantic segmentation, and other machine vision tasks to be used in artificial intelligence systems and assist our daily life. However, these models present substantial computational challenges because of deep neural networks and large data sets. Therefore, we usually run these networks remotely after compressing and transmitting data to a remote end. However, the decompressed images are distorted, which will affect the accuracy of the following tasks.

Classical image compression standards, such as, JPEG [1], HEVC/H.265-intra [2], FLIF [3] have achieved very good results in human perception. On the other hand, some learned end-to-end image compression methods [4], [5], [6] are proposed. These methods achieve and even surpass the state-of-the-art classical compression standards.

However, many challenges still exist with these approaches. Firstly, these methods are specifically designed for human vision that has a gap with machine vision [7]. Therefore, the decompressed images obtained by these methods may not achieve optimal results on machine vision tasks. Secondly, the image reconstruction may introduce latency and computation. The current commonly used pixel-based scheme is shown in Fig. 1. We input images to an encoder for compression and encoding in the beginning, after that, the bitstream is transmitted to the decoder for decoding to get the decompressed images which are used for finishing machine vision tasks at last. However, decoders in this scheme usually consist of multiple layers of transposed convolution [4] or pixel shuffle [6] which leads to a lot of computation.

Some pixel-based methods [8], [9] that can improve machine vision are proposed. But these methods can not solve the second problem. On the other hand, some related standards, such as Video Coding for Machine (VCM) [7] and JPEG AI [10] which uses the compressed domain to learn are gradually being established to address these problems. However, both

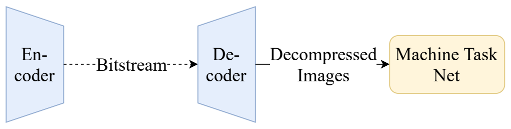

Fig. 1: Traditional pixel domain-based framework.

standards currently provide only a simple framework, which is not effective if used directly. There are still many details that can be modified and improved.

In this paper, we propose an end-to-end multiple tasks scheme of learning in the compressed domain. The main contributions of this paper are as follows:

- We introduce a gate module to adaptively select suitable partial compression domain features and reduce the required bit rate while maintaining the same accuracy of machine vision tasks.
- We improve the accuracy of machine vision tasks by using a knowledge distillation approach.
- We explore a new training strategy that makes our scheme scalable, allowing us to add other machine vision tasks without affecting the existing machine and human vision tasks.
- The results of experiments show that our method is superior to the state-of-the-art pixel-domain method [8] that can take both machine and human vision tasks. In terms of human vision tasks, PSNR can be improved by about 0.5dB at the same bitrate. In terms of machine vision tasks, our method can save up to 80% bit rate with the same accuracy while significantly reducing about the 34% FLOPs for semantic segmentation and foreground extraction, 16% FLOPs for object detection.

II. Method

A. Framework

Our framework is shown in Fig. 2. This framework is mainly composed of four parts: an image compression network, multiple gate modules, transform modules and task networks.

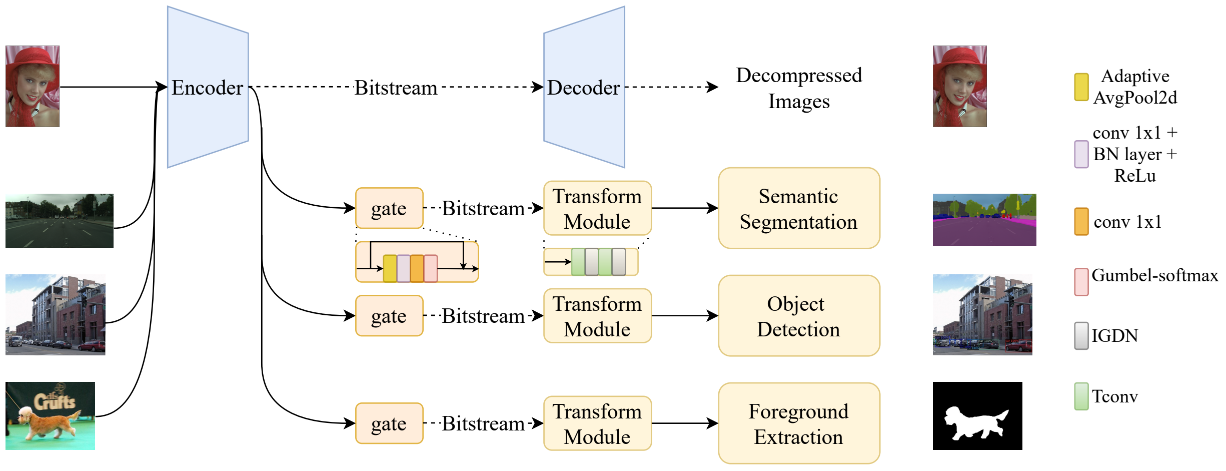

Fig. 2: The framework of our method. The video analysis tasks (semantic segmentation, object detection and foreground extraction) are performed in the compressed domain. In the decoder side, the bitstream is first entropy decoded on CPU, and then the remaining computation is performed on GPU.

Here we use the hyper-prior model in [4] as the image compression network. It can be formulated as the following equations:

$$\begin{aligned} y &= g_a(x;\phi) \\ \hat{y} &= Q(y) \\ \hat{x} &= g_s(\hat{y};\theta) \\ z &= h_a\left(y;\phi_h\right) \\ \hat{z} &= Q(z) \\ \mu_y,\sigma_y &= h_s\left(\hat{z};\theta_h\right) \end{aligned} \tag{1}$$

where the first three equations represent main path and the last three are hyper-prior part. x is an input image while g_a means the analysis transform that convert x to the compressed representation y. Q represents the quantization which is approximated by a noise $\mathcal{U}\left(-\frac{1}{2},\frac{1}{2}\right)$ during training, and is a rounding operator during inference. At last, the decompressed image $\hat{x}$ is generated by the synthesis transform g_s. To capture spatial dependencies, hyper-prior path is introduced. It uses y as input to get side information z by analysis transform h_a. After quantization, the mean μ_y and scale σ_y of $\hat{y}$ are further obtained by synthesis transform h_s, ϕ, θ, ϕ_h, θ_h are learned parameters of network g_a, g_s, h_a, h_s.

We firstly input an image to the encoder of the compression network to get the compressed representation $\hat{y}$. When we need to complete the image compression task, according to the operation mentioned in the previous paragraph, we encode and transmit it to the decoder end for reconstruction. But if our task is a machine vision, we use a gate module for selecting the suitable channels of $\hat{y}$ to get a selected representation $\hat{y}_{selected}$ with less entropy so that we can transmit it with a smaller bit rate. On the other end, to reduce the computation, we don't use the decoder of the image compression network to get a pixel-domain image $\hat{x}$. Instead, we choose to replace the first two layers of the machine vision task network with a transform module, allowing us to learn directly using the representations of the compression domain. The transform module consists of two transpose convolution layers and two IGDN (Inverse Generalized Normalization Transformation). It is used for transforming the representations of the compression domain into the features of machine vision tasks.

B. Gate Module

Here we use a gate module to select useful channels of compressed representation for machine vision tasks. Previous works [13], [18], [22] have proved that channel selection in compressed domain can effectively reduce the bit rate. But some of these methods are only tested in the DCT domain, rather than the compression domain obtained by the neural network. Some other methods try to select channels through some objective metrics such as variance and information entropy. Nonetheless, these selection methods cannot select the optimal channels because different inputs need a different number of channels, but these methods can only choose a fixed number of channels. At the same time, by using these methods, we can only remove some channels with low entropy, which has little effect on the bit rate improvement. And it leaves some channels that are high in entropy but useless for the task still in the compressed representation.

Different from the above methods, in this paper, we use a gate module to select channels in the compressed domain adaptively. Refer to [13], the architecture of the gate module is designed as shown in Fig. 2. A representation of size $H \times W \times C$ is first transformed into a tensor of size $1 \times 1 \times C$ through an adaptive average pooling layer. And then through a 1×1 convolution layer, a batch normalization (BN) layer, a ReLU layer, and a 1×1 convolution layer to get a tensor whose size is $1 \times 1 \times C \times 2$. In this $2 \times C$ number, the first C number represents the probabilities that C channels of compressed representation $\hat{y}$ are sampled as 0 respectively, while the latter means the probabilities that C channels are sampled as 1.

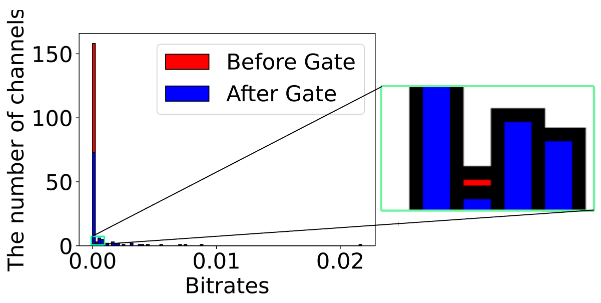

Fig. 3: The histogram of assigned bitrates of a compressed representation before and after channel selection (The enlarged region indicates that our method is different from the channel selection method based on variance. We can directly remove some high-bit-rate but useless channels, instead of removing low-bit-rate channels in bit-rate order).

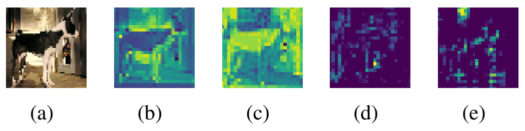

Fig. 4: (a) The original images come from DUTS dataset. (b)(c) Two channels of the compressed representation of (a). (d)(e) The output of the second layer of ResNet-50.

We can use Softmax to evaluate which channel is most likely to be selected and then sample according to this probability. However, since Softmax results are approximately one-hot which are not suitable for sampling, so Gumbel Softmax [13] is used here to solve this problem.

Fig. 3 is the histogram of assigned bitrates of a compressed representation before and after channel selection. The red parts represent channels that have been removed. If we choose according to the method of information entropy or variance, it means that we can only remove the channels with the lowest bit rate according to the order of bit rate. Bins of the histogram resulting from this method will only turn red if all the bins on the left turn red. However, in our method, we can choose the most suitable channels adaptively rather than in accordance with the order of bit rate, which means that we can remove some useless channels with high bit rate, as shown in the enlarged area in the figure. When the channels with small bit rate on the left bins are not completely removed, it can also remove some channels with larger bit rates on the right.

C. *Knowledge Distillation*

As be shown in Fig. 4, there is a gap between the representation of the compressed domain and the feature of the middle layer of machine vision. The former contains obvious structural features, while the latter pays more attention to texture information. A transform module is added to help the compressed representation to convert to the feature of the middle layer in the machine vision networks. Its details are shown in Fig. 2. It consists of only two transposed convolution layers and two IGDNs, which make its output size the same as the input size required by machine vision networks. For example, if the original image's size is $512 \times 768 \times 3$, after the encoder and gate, we can get $\hat{y}_{selected}$ whose size is $32 \times 48 \times C_{selected}$, where $C_{selected}$ means the number of selected channels. After passing through this transform module, its size increases by 4 times, and channels are changed to 64, becoming $128 \times 192 \times 64$ which is equal to the input size of the machine vision task. Because of the neural network of machine vision here we have removed the first two layers, therefore, we input from the third layer, which means the input size of the task net has changed from $512 \times 768 \times 3$ to $128 \times 192 \times 64$.

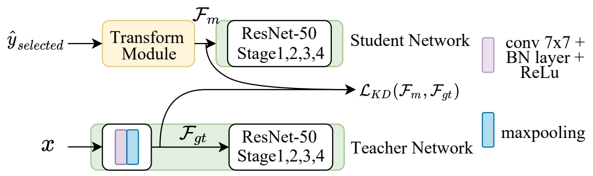

Fig. 5: The knowledge distillation approach on ResNet-50.

In order to enable the transform module to better learn how to convert compressed representation to the appropriate features for machine vision tasks, we introduce a knowledge distillation approach. The process is shown in Fig. 5. To accomplish this knowledge distillation, we have two neural networks here, a teacher network and a student network. The former is a network pre-trained by the original image and the weights are frozen. The latter network is a backbone whose first two layers are replaced with a transform module. During the training, the selected compressed domain representation $\hat{y}_{selected}$ is input into the student network, and the middle-layer features $\mathcal{F}_m$ are obtained through the transform module. The original image x is input to the teacher network, after it passes the first two layers, we can get the middle-layer features $\mathcal{F}_{gt}$. Here, we take this feature as a ground truth, and get the loss $\mathcal{L}_{KD}$ by calculating the MSE between $\mathcal{F}_m$ and the ground truth $\mathcal{F}_{gt}$. By backpropagating this loss, the student network can learn more effective features. The loss $\mathcal{L}_{KD}$ is expressed as the following equation:

$$\mathcal{L}_{KD} = MSE\left(\mathcal{F}_m, \mathcal{F}_{gt}\right) \tag{2}$$

Finally, the loss $\mathcal{L}_m$ of the machine vision task becomes:

$$\mathcal{L}_m = \lambda_1 \mathcal{L}_{m_{original}} + \lambda_2 \mathcal{L}_{KD} \tag{3}$$

where $\mathcal{L}_{m_{original}}$ represents the original loss of the machine vision task, such as MSE loss, IoU loss and so on. λ_1, λ_2 are used for balancing the two losses.

D. *Training Strategy*

It has been proved that combined training compression network and task network can achieve better machine vision effect [17]. However, some machine vision tasks require data augmentations to achieve better results. For example, during the training of object detection, images are scaled up or down to make them more robust when facing images of different sizes during inference. Nevertheless, scaling up the image has a negative effect on the training of the image compression task. Here, we want to implement a framework in JPEG AI [10] that can simultaneously implement multiple machine task vision tasks as well as human vision tasks. Therefore, we use a training strategy here to resolve the contradiction between the two tasks, so as to achieve better PSNR of human vision task while improving the accuracy of machine vision tasks.

In order to maintain the effect of both the machine vision task and the human vision task, similar to [8], [17], we divided the machine vision tasks we needed to implement into primary and secondary tasks. The primary tasks are usually those tasks whose training methods are similar to image compression, such as semantic segmentation, foreground extraction and so on. For the primary task, we jointly train it with the compression network, and the loss can be expressed as:

$$\mathcal{L}_{primary} = \mathcal{L}_m + \lambda_3 MSE(x, \hat{x}) + \lambda_4 R \tag{4}$$

where

$$\begin{aligned} R =& \mathbb{E}\left[-\log_2\left(p_{\hat{\boldsymbol{y}}\mid\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}\mid\hat{\boldsymbol{z}})\right)\right] + \\ & \mathbb{E}\left[-\log_2\left(p_{\hat{\boldsymbol{z}}\mid\boldsymbol{\psi}}(\hat{\boldsymbol{z}}\mid\boldsymbol{\psi})\right)\right] \end{aligned} \tag{5}$$

where the last two items in (4) are the loss of image compression, where the former represents the distortion of decompressed images while the latter R represents the bit rates of $\hat{y}$ and $\hat{z}$. Since there is no prior for $\hat{z}$, a factorized density model ψ is used to encode $\hat{z}$.

After completing the joint training of the primary task and the image compression task, we fixed the weights of these networks. The following machine vision task we added to this scheme are called secondary tasks. Their input comes from the compressed domain representation, which is the output of the encoder that has been trained by the primary task and the image compression task. We can add any number and any type of secondary tasks, because the weight of the encoder part has been fixed, and the training of the secondary task will not affect the effect of the image compression task and the primary task. The loss $\mathcal{L}_{secondary,i}$ of secondary tasks i can be expressed as:

$$\mathcal{L}_{secondary,i} = \mathbb{SG}(\mathcal{L}_{m,i}; \phi, \phi_h) \tag{6}$$

where $\mathbb{SG}$ is the stop-gradient operator so that the encoder parameters ϕ and ϕ_h are not updated during the training of secondary tasks.

III. Experiments

A. Experiment settings

For the image compression network, we use the hyperprior model in [4] here. For machine vision tasks in order to verify the effectiveness of our framework, we carried out experiments on three different machine vision tasks, namely semantic segmentation, object detection and foreground extraction. For all of these three cases, we use ResNet-50 as the backbone, and then add the specific layers for each task.

1) Semantic Segmentation: We utilize Deep Lab V3 [23] which is a classical network for segmentation tasks. The Cityscapes dataset [24] is used for training and evaluation. We use cross entropy as a loss during training. Mean intersection over union (mIoU) is used as the metric of effectiveness.

2) Foreground Extraction: The Deep Lab V3 is also used for foreground extraction. Since this task only needs to distinguish a pixel in the foreground or background, and don't need to distinguish which object the pixel is, we can use 0,1 to represent either background and foreground. Therefore, a sigmoid layer is added at the last layer of Deep Lab V3. This task is trained and evaluated on DUTS dataset [25]. Dice loss is used in the training process. And the results are measured by Mean Absolute Error (MAE).

3) Object Detection: Faster RCNN [26] is used for object detection. And the Pascal VOC 07+12 train sets [27] are utilized to train, while the Pascal VOC 07 test set is used for testing. The shortest edge is scaled up to 640 in the test. we measure the results by using Average precision (AP).

4) Training Setup: In order to accelerate convergence, we use two training tricks. Firstly, all training is based on the pre-training models, in which the pre-training model of the compression network is come from CompressAI [28], and the pre-training of the machine vision task network is completed separately based on the corresponding data sets. The other point is, in the early stages of training, about 10% of the total number of iterations, we only train with the loss of knowledge distillation $\mathcal{L}_{KD}$. For all the training tasks here, we chose Adam as the optimizer. The initial learning rate is 1e-4, and drops to 1e-5 and 1e-6 at 70% and 90% of the total epochs, respectively. For semantic segmentation, object detection and foreground extraction, we trained 800, 100 and 300 epochs on a GeForce RTX 3090 GPU with 24 GB RAM respectively. And all of these tasks' batch sizes are set as 4. λ_1, λ_2, λ_4 are fixed as 1, 1, 1 while λ_3 is used for adjusting bitrate.

In the following experiments, "Informed from the primary task" is used for indicating what the primary task is if the current task is not trained as the primary one. For example, "Informed from Seg" indicates that the primary task is semantic segmentation, and "Informed from Foreground" indicates that the primary task is foreground extraction.

B. Human Vision Task

Refer to [18], [29], we also test the PSNR on machine vison dataset. The results is shown in Fig. 6. Since our compression model is based on the hyperprior model, we compare the original hyperprior model which is pretrained by CompressAI with our hyperprior model jointly trained by a machine vision task. It can be seen that the PSNR after joint training is comparable and even superior to the original model, especially in the cases of low bit rate. At the same bit rate, PSNR can be improved by about 0.5dB in the cases of low bit rate. The results are similar whether we make foreground extraction or semantic segmentation as the primary task, which proves the robustness of our model. This also suggests that the introduction of machine vision tasks, if trained in the right

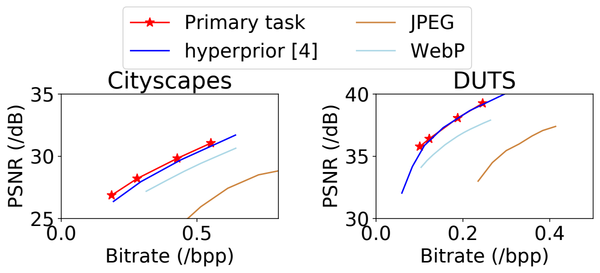

Fig. 6: The PSNR comparison (The left one is based on Cityscapes dataset when the primary task is semantic segmentation, the right one is based on DUTS test dataset when the primary task is foreground extraction.)

way, can improve human vision tasks to some extent, because the labels of machine vision tasks give us more information. For example, if we jointly train semantic segmentation task and image compression task, due to the label of semantic segmentation, we can clearly know the edge information of each image, which can help us improve the image compression task. On the other hand, we also compare with some classical compression methods, such as JPEG and WebP. The results show that our method is superior to these two methods.

C. Machine Vision Task

Here, we take foreground extraction and semantic segmentation as the primary task, while foreground extraction, semantic segmentation and object detection as the secondary task. The results are shown in Fig. 7. The three tasks were evaluated by MAE, mIoU and AP respectively. The lower value of the first metric is, the better effect of the task is, while the higher value of the latter two is, the better effect is. [8] proposed the a pixel-based method that can improve machine vision while maintaining the effect of the human vision task. They name their method as task-finetune (T-FT). Here we use this approach to improve the hyperprior model as a comparison for our approach. We can see that our method achieves better results than T-FT at any bit rate when the task is the primary task. When the task is a secondary task, it also outperforms T-FT at most bit rates, especially at low bit rates. In another word, our method can save up to 80% bit rate with the same accuracy. In addition, we make a comparison with the inference made directly on the original image compression model (the inference method in Fig. 1) without using any improving method like [8]. Our approach is a huge improvement over the original hyperprior [4]. Because the training of the compression network has been aimed at human vision, its performance on machine vision tasks is not good. It can also be seen that, even the Cheng's network [6] which is more advanced than hyperprior is inferior to that of our method based on hyperprior. This also illustrates the previous point, there is a gap between human vision and machine vision. And our approach can bridge the gap between human vision and machine vision.

On the other hand, for the object detection task, we obtain better results than T-FT in both cases informed from foreground and segmentation, which indicates that this framework is robust, different primary tasks can make the secondary tasks achieve similar effects. Another point worth noting is that Cheng's method outperforms hyperprior using T-FT for this task, a result that is reversed from the semantic segmentation task, suggesting that there may also be large differences between two different machine vision tasks. But our model works well for different tasks.

D. Ablation Study

To verify the validity of the gate module and knowledge distillation loss, we conduct ablation studies on the semantic segmentation task when primary is foreground extraction, and the results are shown in Fig. 8.

It can be seen that the model performs worst without knowledge distillation loss $\mathcal{L}_{KD}$ and gate module. When we combine the gate and knowledge distillation loss, we can get the best results.

E. Computational Cost Evaluation

In Table I, we evaluate the computation cost of the three tasks in the pixel domain method and in our method. The input of foreground extraction, semantic segmentation and object detection are RGB images with sizes of 448×448, 1024×2048, 1024×1462, respectively. It can be seen that our method significantly reduces the computational cost by about 16%$\sim$34% of the model compared with the pixel domain-based method [8].

Acknowledgment

This paper is supported by Kenjiro Takayanagi Foundation.

References

[1] G. K. Wallace, "The jpeg still picture compression standard," *IEEE transactions on consumer electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[2] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[3] J. Sneyers and P. Wuille, "Flif: Free lossless image format based on maniac compression," in *2016 IEEE international conference on image processing (ICIP)*. IEEE, 2016, pp. 66–70.

[4] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *International Conference on Learning Representations*, 2018.

[5] D. Minnen, J. Ballé, and G. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *NeurIPS*, 2018.

[6] Z. Cheng, H. Sun, M. Takeuchi, and 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.

[7] L. Duan, J. Liu, W. Yang, T. Huang, and W. Gao, "Video coding for machines: A paradigm of collaborative compression and intelligent analytics," *IEEE Transactions on Image Processing*, vol. 29, pp. 8680–8695, 2020.

[8] L. D. Chamain, F. Racapé, J. Bégaint, A. Pushparaja, and S. Feltman, "End-to-end optimized image compression for machines, a study," in *2021 Data Compression Conference (DCC)*. IEEE, 2021, pp. 163–172.

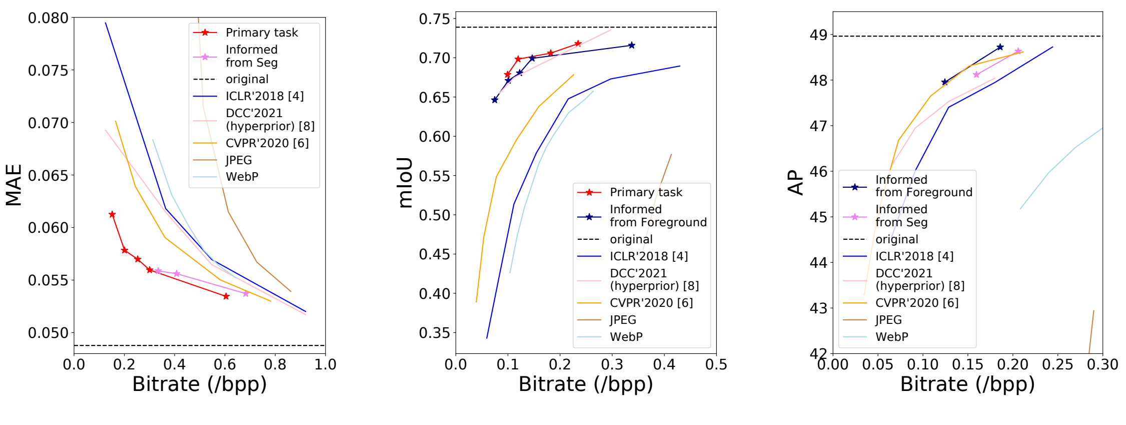

Fig. 7: The metrics of machine vision tasks comparison based on (a) DUTS, (b) Cityscapes, (c) PASCAL VOC 07.

TABLE I: The Multiply–Accumulate Operations (MACs) evaluation of the three machine vision tasks.

	Segmentation (Pixel Domain [8])	Segmentation (Ours)	Detection (Pixel Domain [8])	Detection (Ours)	Foreground (Pixel Domain [8])	Foreground (Ours)
MACs(/G)	783.26	516.45 (34.06%↓)	1165.77	977.89 (16.12%↓)	74.49	48.96 (34.27%↓)

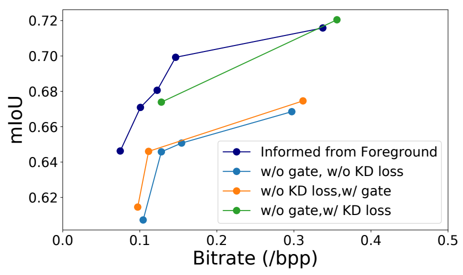

Fig. 8: The ablation studies on the semantic segmentation task.

[9] L. D. Chamain, F. Racapé, J. Bégaint, A. Pushparaja, and S. Feltman, "End-to-end optimized image compression for multiple machine tasks," *arXiv preprint arXiv:2103.04178*, 2021.

[10] J. Ascenso, "Jpeg ai use cases and requirements," in *ISO/IEC JTC1/SC29/WG1 M90014*, 2021.

[11] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[12] M. Rabbani, "Jpeg2000: Image compression fundamentals, standards and practice," *Journal of Electronic Imaging*, vol. 11, no. 2, p. 286, 2002.

[13] K. Xu, M. Qin, F. Sun, Y. Wang, Y.-K. Chen, and F. Ren, "Learning in the frequency domain," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1740–1749.

[14] X. Shen, J. Yang, C. Wei, B. Deng, J. Huang, X.-S. Hua, X. Cheng, and K. Liang, "Dct-mask: Discrete cosine transform mask representation for instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8720–8729.

[15] R. Torfason, F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. Van Gool, "Towards image understanding from deep compression without decoding," in *International Conference on Learning Representations*, 2018.

[16] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," in *International Conference on Learning Representations*, 2017.

[17] F. Codevilla, J. G. Simard, R. Goroshin, and C. Pal, "Learned image compression for machine perception," *arXiv preprint arXiv:2111.02249*, 2021.

[18] Z. Wang, M. Qin, and Y.-K. Chen, "Learning from the cnn-based compressed domain," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 3582–3590.

[19] H. Choi and I. V. Bajic, "Scalable image coding for humans and machines," *arXiv preprint arXiv:2107.08373*, 2021.

[20] F. Yang, Q. Zhang, M. Wang, and G. Qiu, "Quality classified image analysis with application to face detection and recognition," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 2863–2868.

[21] L. Galteri, M. Bertini, L. Seidenari, and A. Del Bimbo, "Video compression for object detection algorithms," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 3007–3012.

[22] J. Liu, H. Sun, and J. Katto, "Learning in compressed domain for faster machine vision tasks," in *Visual Communications and Image Processing, VCIP 2021*, 2021.

[23] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[24] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[25] L. Wang, H. Lu, Y. Wang, M. Feng, D. Wang, B. Yin, and X. Ruan, "Learning to detect salient objects with image-level supervision," in *CVPR*, 2017.

[26] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[27] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[28] J. Bégaint, F. Racapé, S. Feltman, and A. Pushparaja, "Compressai: a pytorch library and evaluation platform for end-to-end compression research," *arXiv preprint arXiv:2011.03029*, 2020.

[29] Y. Bai, X. Yang, X. Liu, J. Jiang, Y. Wang, X. Ji, and W. Gao, "Towards end-to-end image compression and analysis with transformers," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.