Complexity-guided Slimmable Decoder for Efficient Deep Video Compression (Supplementary Materials)

A. Training Details

At the first stage, we train the whole network without complexity penalty. The loss function at this stage is formulated as follows,

$$L = R_m + R_r + \lambda D(\hat{X}_t, X_t) \tag{1}$$

in which the R_m and R_r denote the bit-rate cost for entropy coding the motion and residual information, respectively. $D(\hat{X}_t, X_t)$ denotes the distortion between the output frame $\hat{X}_t$ and the input frame X_t. λ is the hyper-parameter to control the trade-off between the bit-rate cost and the distortion.

For the second stage, we add the complexity penalty for achieving different complexity constraints. N complexity targets are predefined as $C_1^{tar}, C_2^{tar}, ..., C_N^{tar}$. For each training step, we randomly select a complexity target C_i^{tar} and feed the corresponding one-hot complexity vector to the network. For the i^{th} complexity target C_i^{tar}, the optimization function is formulated as follows,

$$L = R_m + R_r + \lambda D(\hat{X}_t, X_t) + \alpha_i C_i \tag{2}$$

in which C_i is the decoder complexity at the current step. However, by simply using the predefined and fixed weight for α_i, we cannot reach the target complexity constraint as we need to learn the optimal selection options to simultaneously satisfy multiple complexity targets. To address this issue, we empirically define the weight α_i as the L2 distance between the current complexity C_i and the target complexity C_i^{tar}, which is formulated as

$$\alpha_i = \beta(C_i - C_i^{tar})^2 \tag{3}$$

where β is a normalization weight to balance the trade-off between the rate-distortion loss and the complexity penalty. In our work, β is set as 0.001 when C_i is larger than C_i^{tar}, otherwise it is empirically set as -0.001 for achieving higher complexity. Note that we use GMACs (Giga multiply-accumulate operations) to calculate the complexity of C_i and C_i^{tar}.

Additionally, the hyper-parameter τ of each Gumbel Softmax layer of the newly proposed mode selection module is gradually reduced from three to zero so the channel width of each layer is first randomly selected and the selection process will gradually become stable. Therefore, our proposed complexity-guided slimmable decoder (cgSlimDecoder) will gradually reach the pre-defined target complexities with different input complexity vectors.

B. Details of Gumbel Softmax

Recently, The Gumbel Softmax strategy [2, 4] is proposed for handling the undifferentiable issue. In this work, we use Soft Gumbel with the reparametrization trick for our complexity-guided channel width selection. When $\mathbf{x} = [x_1, x_2, x_3]$ is the predicted confidence scores of three options (*i.e.,* channel widths) during training, we obtain $\hat{x}_i = x_i + G_i$, where $G_i = -log(-logU_i)$ is the gumbel noise and U_i is uniformly sampled from 0 to 1. Then we obtain the output $\hat{\mathbf{a}} = [\hat{a}_1, \hat{a}_2, \hat{a}_3]$, where $\hat{a}_i = 1$ if $i = \arg\max_j \hat{x}_j$, otherwise $\hat{a}_i = 0$. During back-propagation, we relax $\hat{\mathbf{a}}$ to $\tilde{\mathbf{a}} = [\tilde{a}_1, \tilde{a}_2, \tilde{a}_3]$, in which $\tilde{a}_i = \frac{exp(\hat{x}_i/\tau)}{\sum_{j=1}^{3} exp(\hat{x}_j/\tau)}$.

C. The Difference Between SaEC and [1]

Our SaEC is proposed for accelerating the entropy coding of both motion and residual and uses the predicted mean value for the skipped elements, while [1] only skips the entropy coding of residual information and set the skipped elements as zero. Moreover, Our SaEC performs better than [1]. On the MCL-JCV dataset, the BD-PSNR results of our cgSlimDecoder+SaEC(FVC) are 0.172dB/0.167dB/0.088dB at three complexity levels, while the corresponding results are -0.051dB/-0.066dB/-0.146dB when using the skip method in [1].

D. Selected Network Structure of DVC

In Table S1, we provide the detailed selected network structure of our proposed cgSlimDecoder(DVC). We observe that the network structure of the residual decoder is not changed at three complexity levels, which denotes that the residual decoder plays a more important role in video compression than the motion decoder and the motion compensation module so we reduce the complexities of both motion compression and motion compensation modules to prevent the significant drop of rate-distortion performance.

Table S1. Selected network structure of our proposed cgSlimDecoder(DVC) when using different input complexity vectors for achieving different complexity constraints. "C(C_{in},C_{out})" denotes a convolution layer with C_{in} input channels and C_{out} output channels while "D(C_{in},C_{out})" denotes a deconvolution layer with C_{in} input channels and C_{out} output channels. "↓"/"↑" after the convolution/deconvolution layer denotes the current convolution/deconvolution layer has stride 2. "R(C)" denotes a residual block with C channels. "↓"/"↑" after the residual block denotes the current residual block is followed by an average pooling/upsampling layer with stride 2. "Level 1", "Level 2" and "Level 3" indicate the complexity at the highest, the middle and the lowest level, respectively.

Original	Level 1	Level 2	Level 3
Motion Decoder			
D(128,128)↑	D(64,64)↑	D(64,64)↑	D(64,64)↑
C(128,128)	C(64,64)	C(64,64)	C(64,64)
D(128,128)↑	D(64,64)↑	D(64,64)↑	D(64,64)↑
C(128,128)	C(64,64)	C(64,64)	C(64,64)
D(128,128)↑	D(64,64)↑	D(64,64)↑	D(64,64)↑
C(128,128)	C(64,64)	C(64,64)	C(64,64)
D(128,128)↑	D(128,128)↑	D(64,64)↑	D(32,32)↑
C(128,2)	C(32,2)	C(64,2)	C(64,2)
Motion Compensation Network			
C(6,64)	C(64,64)	C(64,64)	C(64,64)
R(64)↓	R(64)↓	R(64)↓	R(32)↓
R(64)↓	R(64)↓	R(64)↓	R(64)↓
R(64)	R(64)	R(64)	R(64)
R(64)↑	R(64)↑	R(64)↑	R(64)↑
R(64)↑	R(64)↑	R(64)↑	R(32)↑
R(64)	R(64)	R(32)	R(32)
D(64,3)	D(16,3)	D(16,3)	D(16,3)
Residual Decoder			
D(96,96)↑	D(96,96)↑	D(96,96)↑	D(96,96)↑
D(96,96)↑	D(96,96)↑	D(96,96)↑	D(96,96)↑
D(96,96)↑	D(96,96)↑	D(96,96)↑	D(96,96)↑
D(96,3)↑	D(96,3)↑	D(96,3)↑	D(96,3)↑

When comparing the lowest complexity level (*i.e.*, level 3) with the higher complexity levels (*i.e.*, level 1 and level 2), it is observed that the channel widths of the last few layers in the motion decoder and the channel widths of the first and the last residual blocks in the motion compensation modules are reduced. Most of these layers are performed in higher resolutions, so reducing the channel widths in these layers can effectively reduce the model complexity without significant performance drop.

E. Selected Network Structure of FVC

In Table S2, we provide the detailed selected network structure of our proposed cgSlimDecoder(FVC). When comparing the network from the highest complexity level (*i.e.*, level 1) with the original network structure, it is observed that the channel widths of some independent convolution layers (*e.g.*, the independent convolution layers in feature extraction, residual decoder and frame reconstruction) are reduced, which denotes that it is inefficient to use large channel widths for these independent layers. When comparing our cgSlimDecoder(FVC) at the middle complexity level (*i.e.*, level 2) with our cgSlimDecoder(FVC) at the highest complexity level, the channel widths of the residual blocks in the motion decoder are further reduced. And when comparing our cgSlimDecoder(FVC) at the lowest complexity level (*i.e.*, level 3) with ours at the middle complexity level, the channel widths of the residual blocks in the residual decoder are reduced. One possible explanation is that reconstructing the residual information is more important than reconstructing the motion information and thus we need to allocate more complexity to the residual decoder than the motion decoder.

Table S2. Selected network structure of our proposed cgSlimDecoder(FVC) when using different input complexity vectors for achieving different complexity constraints. "C(C_{in},C_{out})" denotes a convolution layer with C_{in} input channels and C_{out} output channels while "D(C_{in},C_{out})" denotes a deconvolution layer with C_{in} input channels and C_{out} output channels. "↓"/"↑" denotes the current convolution/deconvolution layer has stride 2. "R(C)" denotes a residual block with C channels. "Level 1", "Level 2" and "Level 3" indicate the complexity at the highest, the middle and the lowest level, respectively.

Original	Level 1	Level 2	Level 3
Feature Extraction Module			
C(3,64)↓	C(3,32)↓	C(3,32)↓	C(3,32)↓
R(64)	R(64)	R(64)	R(64)
Motion Decoder			
D(128,128)↑	D(128,64)↑	D(128,64)↑	D(128,64)↑
R(128)	R(128)	R(64)	R(64)
D(128,128)↑	D(128,128)↑	D(64,64)↑	D(64,64)↑
R(128)	R(128)	R(64)	R(64)
D(128,64)↑	D(64,64)↑	D(32,64)↑	D(32,64)↑
Motion Compensation Network			
C(128,64)	C(128,64)	C(128,64)	C(128,16)
C(64,64)	C(64,64)	C(64,64)	C(16,16)
Residual Decoder			
D(128,128)↑	D(128,32)↑	D(64,32)↑	D(64,32)↑
R(128)	R(128)	R(128)	R(64)
D(128,128)↑	D(32,32)↑	D(32,32)↑	D(32,32)↑
R(128)	R(128)	R(128)	R(64)
D(128,64)↑	D(32,64)↑	D(32,64)↑	D(32,64)↑
Frame reconstruction Module			
R(64)	R(64)	R(64)	R(64)
C(64,3)↑	C(32,3)↑	C(32,3)↑	C(32,3)↑

Table S3. Selected network structure of our proposed cgSlimDecoder(DCVC) when using different input complexity vectors for achieving different complexity constraints. "C(C_{in},C_{out})" denotes a convolution layer with C_{in} input channels and C_{out} output channels while "D(C_{in},C_{out})" denotes a deconvolution layer with C_{in} input channels and C_{out} output channels. "↓"/"↑" after the convolution/deconvolution layer denotes the current convolution/deconvolution layer has stride 2. "R(C_{in},C_{out})" denotes a residual block with C_{in} input channels and C_{out} output channels. "Level 1", "Level 2" and "Level 3" indicate the complexity at the highest, the middle and the lowest level, respectively.

Original	Level 1	Level 2	Level 3
Motion Decoder			
D(128,128)↑	D(64,64)↑	D(64,64)↑	D(64,64)↑
D(128,128)↑	D(64,64)↑	D(64,64)↑	D(64,64)↑
D(128,128)↑	D(32,32)↑	D(32,32)↑	D(32,32)↑
D(128,2)↑	D(32,2)↑	D(32,2)↑	D(32,2)↑
Motion Refinement			
C(5,64)	C(5,16)	C(5,16)	C(5,16)
C(64,64)	C(32,32)	C(32,32)	C(32,32)
C(64,64)	C(64,64)	C(32,32)	C(32,32)
C(64,64)	C(64,64)	C(64,64)	C(32,32)
C(64,64)	C(64,64)	C(64,64)	C(32,32)
C(64,64)	C(64,64)	C(64,64)	C(16,16)
C(64,2)	C(64,2)	C(64,2)	C(32,2)
Feature Extraction			
C(3,64)	C(3,32)	C(3,32)	C(3,32)
R(64,64)	R(64,64)	R(32,64)	R(32,64)
Context Refinement			
R(64,64)	R(64,16)	R(64,16)	R(64,16)
C(64,64)	C(32,64)	C(32,64)	C(32,64)
Context Encoder			
C(64,64)↓	C(64,32)↓	C(64,32)↓	C(64,16)↓
C(64,64)↓	C(32,32)↓	C(32,32)↓	C(32,32)↓
C(64,64)↓	C(32,32)↓	C(32,32)↓	C(32,32)↓
C(64,96)↓	C(32,96)↓	C(32,96)↓	C(32,96)↓
Contextual Decoder			
D(96,64)↑	D(96,64)↑	D(96,64)↑	D(96,32)↑
D(64,64)↑	D(64,64)↑	D(64,64)↑	D(32,32)↑
R(64,64)	R(64,64)	R(64,64)	R(32,32)
D(64,64)↑	D(64,64)↑	D(64,64)↑	D(32,32)↑
R(64,64)	R(64,64)	R(64,64)	R(16,16)
D(64,64)↑	D(64,64)↑	D(64,64)↑	D(32,64)↑
C(128,64)	C(128,64)	C(128,32)	C(128,32)
R(64,64)	R(64,64)	R(32,32)	R(32,32)
R(64,64)	R(64,64)	R(32,32)	R(16,16)
C(32,3)	C(32,3)	C(32,3)	C(32,3)

F. Selected Network Structure of DCVC

In Table S3, we provide the detailed selected network structure of our proposed cgSlimDecoder(DCVC). DCVC [3] is the recently proposed deep video compression method and contains more complex network structure, which makes it extremely hard to manually design the network structure to satisfy different complexity constraints. In our proposed cgSlimDecoder(DCVC), our method can automatically allocate the total complexity to different modules. For example, the motion refinement network is less important than the motion decoder. Therefore, the channel widths of the motion refinement network are reduced from higher complexity levels to lower complexity levels, while the network structure of motion decoder is not changed. The selected network structures of our cgSlimDecoder(DCVC) demonstrate the effectiveness of our proposed cgSlimDecoder.

G. Results on the HEVC Datasets

In our main paper, we only provide the experimental results on the UVG [5] and MCL-JCV [6] datasets due to space limitation. We have also mentioned that the results on the HEVC datasets will be shown in the *supplementary material*. Therefore, in this section, We provide the average results of our method on the HEVC datasets including HEVC Class B, C, D, and E datasets.

In Figure S1, we provide the BD-PSNR results at different complexity levels on the HEVC datasets. In terms of the average performance over the HEVC Class B, C, D and E datasets, we observe that our proposed cgSlimDecoder(DVC) drops about 0.3 dB when compared with the original DVC at the lowest complexity level. Additionally, our cgSlimDecoder+SaEC(DVC) not only improves 0.2 dB BD-PSNR performance at the highest complexity level when compared with cgSlimDecoder(DVC) but also improves the efficiency of entropy decoding. The experiments also demonstrate the effectiveness of our newly proposed cgSlimDecoder and saEC.

In Figure S2, we also provide BD-MSSSIM(dB) results at different complexity levels on the HEVC datasets. In terms of the average performance over the HEVC Class B, C, D and E datasets, our cgSlimDecoder(DVC) only drops less than 0.2dB at the lowest complexity level. and our cgSlimDecoder+SaEC(DVC) can also further improve the BD-MSSSIM(dB) results with more efficient entropy decoding.

H. Visualization of Skip-adaptive Entropy Coding

In Figure S3, we provide the visualization of our skip-adaptive entropy coding (SaEC) in our cgSlimDecoder+SaEC(FVC). The second frame of the first video from the HEVC Class C dataset is taken as an example. In Figure S3(a2,a3,a4,a5), we provide the predicted mean and sigma values of the motion and residual information from the hyperprior network. It is observed that the mean values

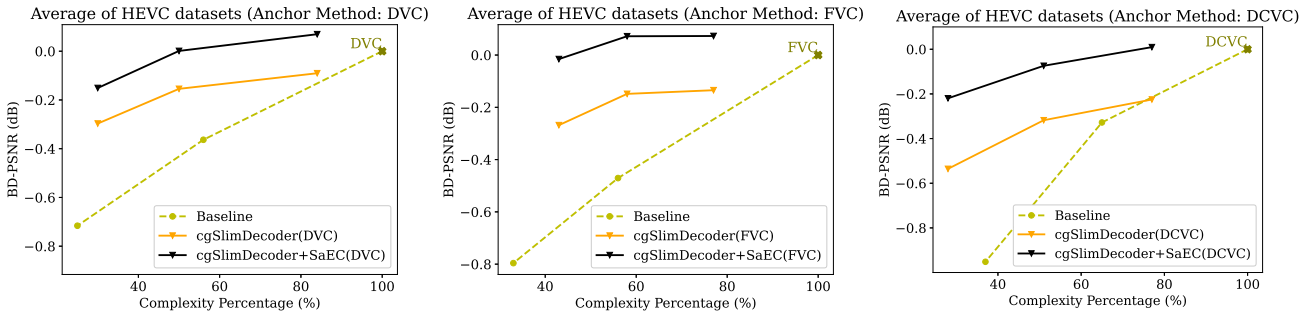

Figure S1. The experimental results (*i.e.*, the average performance over the HEVC Class B, C, D and E datasets) of different methods when using PSNR as the distortion metric.

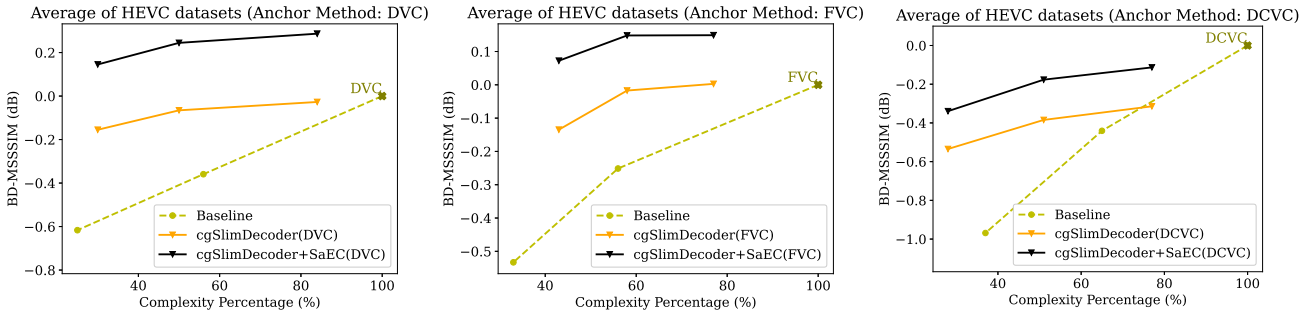

Figure S2. The experimental results (*i.e.*, the average performance over the HEVC Class B, C, D and E datasets) of different methods when using MS-SSIM as the distortion metric.

of motion features can be well predicted by the hyperprior network and the background areas are smooth and can be easily predicted, so we can skip the entropy coding of such areas for motion features. In Figure S3(b2,b3,c2,c3,c5), we provide the predicted 0/1 masks of our SaEC based on the motion and residual features. It is observed that our SaEC still needs to entropy code the areas containing moving objects, while for the background areas, our SaEC skips the entropy coding procedure for more efficient entropy coding. In Figure S3(b4,b5,c4), we also observe that in some channels, the entropy coding process is directly skipped. One possible explanation is that the elements from these channels can be reliably predicted by the hyperprior networks, so we can simply skip the entropy coding process of such channels, which can significantly improve the entropy encoding and decoding efficiency. The visualization results demonstrate that our proposed SaEC can automatically predict the meaningful areas that should be entropy coded and directly skip the entropy coding process of the less meaningful areas and channels to accelerate the entropy encoding and decoding procedures for both motion and residual features.

References

[1] Zhihao Hu, Guo Lu, Jinyang Guo, Shan Liu, Wei Jiang, and Dong Xu. Coarse-to-fine deep video coding with hyperprior-guided mode prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1704–1713, 2022. 1

[2] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *International Conference on Learning Representations (ICLR)*, 2017. 1

[3] Jiahao Li, Bin Li, and Yan Lu. Deep contextual video compression. *Advances in Neural Information Processing Systems*, 34, 2021. 3

[4] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016. 1

[5] A. Mercat, Marko Viitanen, and J. Vanne. UVG dataset: 50/120fps 4k sequences for video codec analysis and development. *Proceedings of the 11th ACM Multimedia Systems Conference*, 2020. 3

[6] Haiqiang Wang, Weihao Gan, Sudeng Hu, Joe Yuchieh Lin, Lina Jin, Longguang Song, Ping Wang, Ioannis Katsavounidis, Anne Aaron, and C-C Jay Kuo. MCL-JCV: a jnd-based H.264/AVC video quality assessment dataset. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 1509–1513. IEEE, 2016. 3

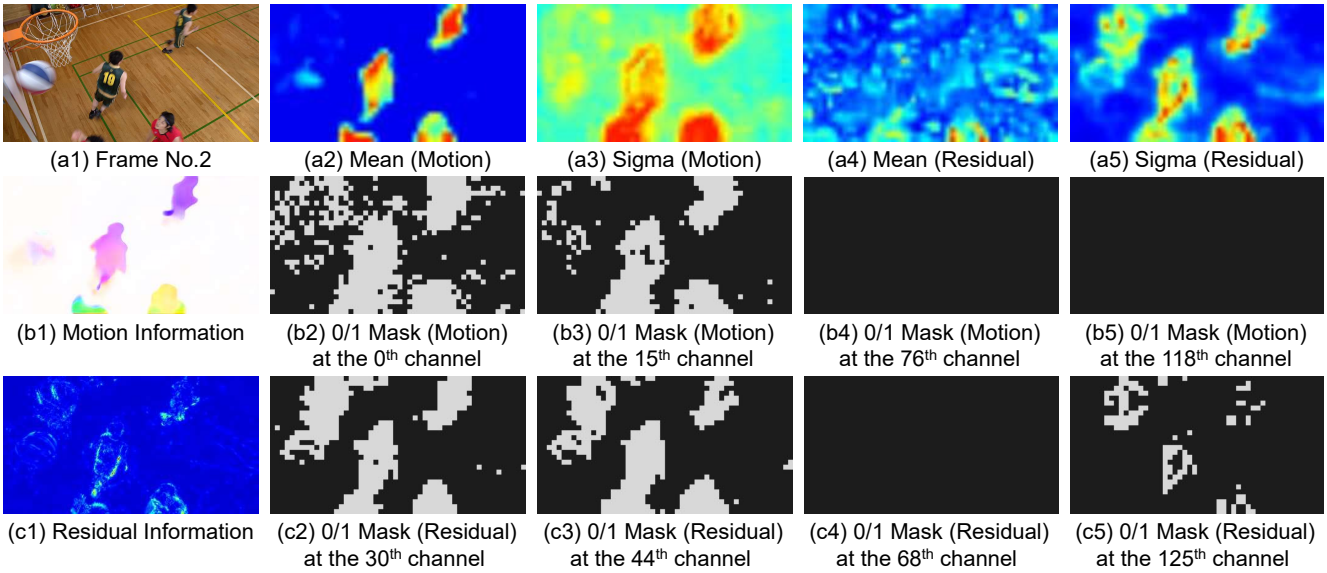

Figure S3. Visualization of our skip-adaptive entropy coding (SaEC) in our cgSlimDecoder+SaEC(FVC). We take the second frame (a1) of the first video on the HEVC Class C dataset as an example. We also provide the predicted mean and sigma of motion features (see (a2) and (a3)) and residual features (see (a4) and (a5)) from the hyperprior network. In the second row, we provide the visualization results of the reconstructed motion information (b1) and the predicted 0/1 mask at different channels (see (b2), (b3), (b4), (b5)) from the mode prediction net in our SaEC. In the third row, we provide the visualization results of the reconstructed residual information (c1) and the predicted 0/1 mask at different channels (see (c2), (c3), (c4), (c5)) from the mode prediction net in our SaEC. Note the white areas in the 0/1 masks denote the corresponding areas will be entropy coded while the black areas in the 0/1 masks denote the corresponding areas will be skipped during the entropy coding procedure for these elements and we directly use the predicted mean value from the hyperprior network during the decoding process.