Supplementary Material: Nonlinear Vector Transform Coding

Runsen Feng Zongyu Guo Weiping Li Zhibo Chen
University of Science and Technology of China
{fengruns, guozy}@mail.ustc.edu.cn, {wpli, chenzhibo}@ustc.edu.cn

A. On Toy Sources

We investigate the advantages of ECVQ on 11 different source distributions with varied dimensions. Among them, we show the 8 two-dimentional distributions in Figure 2, including Isotropic Gaussian, Banana, Boomerang, Laplace, Gaussian Mixture, Sphere0, Shpere50 and Shpere99.

The details about the architecture of codecs are described in Section A.1. The experimental results and analysis are provided in Section A.2.

A.1. Architecture

We implement three different codecs for comparison: VQ/ECVQ, SQ with nonlinear transform (NTC), and VQ/ECVQ with nonlinear transform (NT-VQ/NT-ECVQ).

VQ/ECVQ. As mentioned in the main paper, the encoder function of **VQ** with dimension k and size N is as follows:

$$i = \arg\min_i d(\boldsymbol{x}, \boldsymbol{c}_i), \tag{1}$$

where d is a distortion metric and $\boldsymbol{c}_i$ is a codeword of the codebook $\boldsymbol{c} = \{\boldsymbol{c}_i \in \mathbb{R}^k | i = 0, 1, ..., N-1\}$. The decoder function is a lookup operation: $\hat{\boldsymbol{x}} = \boldsymbol{c}_i$. The codeword probability $P(i) = p_i$ is parameterized with the Softmax function and unnormalized logits $\boldsymbol{w} = (w_1, w_2, ..., w_N)$:

$$p_i = \frac{e^{-w_i}}{\sum_{j=1}^{N} e^{-w_j}}. \tag{2}$$

The rate loss is $\mathbb{E}_{\boldsymbol{x}}[-\log p_i]$ and the distortion loss is $\mathbb{E}_{\boldsymbol{x}}[d(\boldsymbol{x}, \hat{\boldsymbol{x}})]$. The encoder function of **ECVQ** is improved as follows:

$$i = \arg\min_i \left[-\log p_i + \lambda d(\boldsymbol{x}, \boldsymbol{c}_i)\right], \tag{3}$$

where λ controls the RD trade-off.

NTC. The architecture of NTC for toy sources is shown in Figure 3. Input $\boldsymbol{x}$ is transformed into latents $\boldsymbol{y} = g_a(\boldsymbol{x})$ by a analysis transform g_a, which is then quantized by a uniform scalar quantizer (rounding to integers). The quantized latent $\hat{\boldsymbol{y}} = \lfloor \boldsymbol{y} \rceil$ is then mapped back to the reconstruction $\hat{\boldsymbol{x}} = g_s(\hat{\boldsymbol{y}})$ by a synthesis transform g_s. In training, the rounding operation approximated by adding uniform noise [2,3]. We use the factorized prior [3] for entropy modeling. The overall loss is as follows:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}}[-\log P(\hat{\boldsymbol{y}})] + \lambda \mathbb{E}_{\boldsymbol{x}} d(\boldsymbol{x}, \hat{\boldsymbol{x}}). \tag{4}$$

NT-VQ/NT-ECVQ. To verify the effectiveness of ECVQ in latent space, we replace the uniform scalar quantization of NTC with VQ/ECVQ, named NT-VQ/NT-ECVQ. For **NT-VQ**, the encoding function is as follows:

$$i = \arg\min_i d_1(g_a(\boldsymbol{x}), \boldsymbol{c}_i), \tag{5}$$

and the encoding function for **NT-ECVQ** is:

$$i = \arg\min_i \left[-\log p_i + \lambda d_1(g_a(\boldsymbol{x}), \boldsymbol{c}_i)\right]. \tag{6}$$

We have $\boldsymbol{y} = g_a(\boldsymbol{x})$, $\hat{\boldsymbol{y}} = \boldsymbol{c}_i$ and $\hat{\boldsymbol{x}} = g_s(\hat{\boldsymbol{y}})$. To jointly optimize the transform and VQ codebook, we employ the straight-through estimator (STE) used in previous works [1, 9]:

$$\frac{d\hat{\boldsymbol{y}}}{d\boldsymbol{y}} = 1, \tag{7}$$

and propose a new rate-distortion loss for latent space VQ:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}}[-\log p_i] + \lambda \mathbb{E}_{\boldsymbol{x}} d(\boldsymbol{x}, \hat{\boldsymbol{x}}) + \beta \mathbb{E}_{\boldsymbol{x}} d_1(\boldsymbol{y}, \boldsymbol{c}_i), \tag{8}$$

where d is the source-space distortion metric, and d_1 is the latent-space VQ distortion metric. λ controls the rate-distortion trade-off. β controls the trade-off between d and d_1. In practice, both d and d_1 are measured by MSE, and β is equal to λ.

A.2. Experiments

In Figure 4 we provide the RD curves. We show the visualization of quantization results in Figure 5.

Space-filling advantage. As mentioned in the main paper, the space-filling advantage increases with dimension (see the RD results of 2-d, 4-d, 8-d and 16-d isotropic Gaussian distributions in Figure 4).

Memory advantage. The benefits of memory advantage are significant in the distributions of Boomerang, Gaussian Mixture, Sphere0, Sphere50, and Sphere99. All these distributions have highly nonlinear correlations.

ECVQ vs. VQ. ECVQ is better than VQ on all the distributions. An observation is that the gap between VQ and ECVQ is very small on the sphere-like uniform distributions (Sphere0, Sphere50, Sphere99). It is probably because the probabilities of quantization centers on uniform distribution are similar to each other, which decreases the impact of rate bias in ECVQ encoding. Another observation in Figure 5 is that the quantization cells/regions of VQ in high probability area are much smaller than that in low probability area. In contrast, in ECVQ, the quantization regions in different density area have similar sizes. The reason is that the quantization boundaries in ECVQ shift from high probability region to low probability region, enlarging the size of high-probability region.

ECVQ vs. NT-ECVQ. In Figure 5, a major difference between the ECVQ and NT-ECVQ is that NT-ECVQ warps the source space by nonlinear transform, making the quantization boundaries into curves. Moreover, as shown in Figure 4, NT-ECVQ optimized with Equation 8 has a comparable RD performance to ECVQ, demonstrating the effectiveness of the proposed training loss.

NT-ECVQ vs. NT-VQ. We have verified the effectiveness of entropy-constrained quantization in source space. What about the effectiveness in latent space? Can nonlinear transform learn to approximate the shift of quantization boundaries in ECVQ? By comparing the performance of NT-ECVQ and NT-VQ, we show that ECVQ is also important in latent space. Despite nonlinear transform approximate ECVQ well on 1-d distributions [2], it frequently fails on 2-d distributions at most of the rate points. As shown in Figure 5, the shift of quantization boundaries is not observed in the quantization results of NT-VQ.

B. On Neural Images

B.1. Baseline

BPG The RD results of BPG is obtained from CompressAI [4] by running the following command:

```
python -m compressai.utils.bench bpg [dataset]
-q 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45
```

VTM-18.2 The software VTM-18.2 is downloaded from https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM. We first convert the RGB images into YUV444:

```
ffmpeg -i [IMGfile] -pix_fmt yuv444p [YUVfile]
```

We then encode the YUV file into bitstream:

```
EncoderAppStatic -i [YUVfile] -c [CFGfile] -q [QP]
-o [OUTfile] -b [BINfile] -wdt [W] -hgt [H] -fr 1 -f 1
--InputChromaFormat = 444 --InputBitDepth = 8
--ConformanceWindowMode = 1
```

And the decoding command is:

```
DecoderAppStatic -o [OUTfile] -b [BINfile] -d 8
```

Finally, we convert the reconstructed YUVfile into RGB images for evaluation:

```
ffmpeg -s [W]x[H] -pix_fmt yuv444p -i [OUTfile]
[IMGfile]
```

Learning-based methods The results of Cheng-CVPR20 [5], Minnen-NIPS2018 [7] and Balle-ICLR18 [3] are evaluated based on the reimplementation from CompressAI [4]. The complexity results of Zou-CVPR22 [11] and Zhu-CVPR22 [10] are evaluated based on their official implementation. The other results are obtained from the paper or the authors.

B.2. Architecture

Downscale & upscale layers. The detailed architecture of Downscale and Upscale layers are shown in Figure 6. We use Pixel Shuffle [8] instead of strided convolution for upsampling. Downscale-B is enhanced with Resblocks for learning nonlinear vector representation.

Conditional entropy model (CEM). The architecture of CEM at layer l is shown in Figure 7. CEM consists of three parts: 1) an entropy parameter module that generates the prior parameters $\boldsymbol{e}_l$, 2) a vector quantizer that maps $\boldsymbol{e}_l$ into a finite set, and 3) conditional logit prior that outputs discrete probability distribution given the prior parameters $\boldsymbol{e}_l$ or $\hat{\boldsymbol{e}}_l$. The use of VQ in CEM ensures that all possible probability distributions in a layer can be indexed by a distribution table. This distribution table is known to both encoder and decoder after training. Instead of generating probability distributions dynamically, we simply lookup the distribution

table with the VQ index for a much faster entropy coding process (about 6x faster). Currently, the VQ in CEM brings about 0.15dB drop, which can be further optimized in the future. In practice, we first train the model without the VQ in CEM, and then finetune the full model.

Model configurations We provide the detailed model configurations in Table 1. For layer l, $H_l \times W_l \times C_l$ is the feature size. b_l is the block size used in the VT units and product VQ. N_l and k_l are the VQ codebook size and VQ dimension, respectively. T_l is the iteration milestones for progressive initialization. Besides, δT is set to 10k.

Layer	H_l	W_l	C_l	b_l	N_l	k_l	T_l
1	$\frac{H}{16}$	$\frac{W}{16}$	192	4	512	16	0.0M
2							
3							
4							
5							
6							
7	$\frac{H}{8}$	$\frac{W}{8}$			256	8	0.2M
8							
9							
10							
11							
12							
13	$\frac{H}{4}$	$\frac{W}{4}$			128/256	4	0.4M
14							
15							
16							

Table 1. Detailed configurations of different quantization layers.

B.3. Experiments

In this section, we provide additional ablation studies, model properties and visualization of reconstruction. For ablation studies, we provide the BD-rate in Table 2 and the RD curves in Figure 1. BD-rate is calculated using the software from https://github.com/Anserw/Bjontegaard_metric. All models are optimized with 128 $\times$ 128 image patches.

The number of quantization layers. As mentioned in the main paper, the number of quantization layers influence the rate-distortion-complexity trade-off. The full model consists of 16 quantization layers with $(L_1, L_2, L_3) = (6, 6, 4)$. We change (L_1, L_2, L_3) to $(3, 3, 2)$, $(0, 6, 4)$ and $(0, 0, 4)$ and train 3 model variants, which is named "A4: L=3+3+2", "A5: L=0+6+4" and "A6: L=0+0+4", respectively. It can be observed that 1) the models with fewer quantization layers perform worse at high rate points, and

	BD-rate	Parameters (M)
Full model	0.0%	12.8
A4: L=3+3+2	7.7%	8.7
A5: L=0+6+4	6.8%	8.9
A6: L=0+0+4	27.1%	5.8
A7: w/o depth-wise	29.4%	9.5
A8: w/o CEM in EC	39.6%	12.8

Table 2. Ablation study on Kodak. BD-rate is computed over the full models (smaller is better).

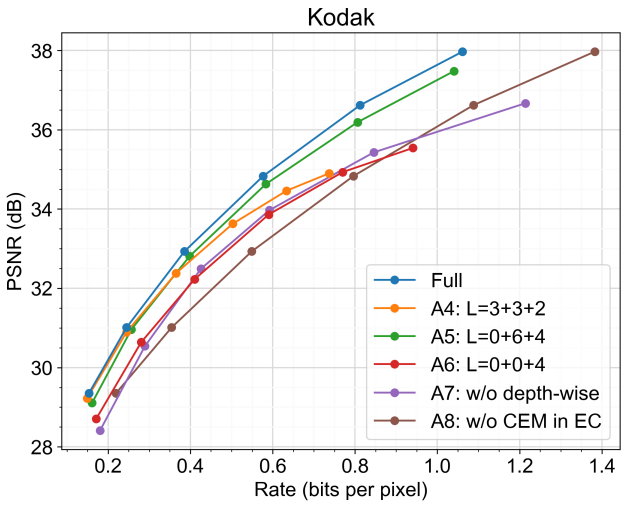


Figure 1. Ablation study on Kodak dataset.

2) the low-resolution quantization layers (numbered by L_1 or L_2) which capture global correlation have a significant impact at all rate points.

Depth-wise Block FC vs. Block FC. We investigate the impact of the proposed Depth-wise Block FC layer. We build a variant named Block FC where all channels use the same transformation matrix. Block FC is similar to Block-DCT with a learnable transformation matrix. The performance of Block FC (noted as "A7: w/o depth-wise") is much worse than the full model.

CEM in entropy coding. CEM plays an important role in removing inter-vector redundancy across different quantization layers. We investigate rate savings when disabling CEM in entropy coding. Instead of retraining the whole model, we simply replace the CEM in a trained model with the unconditional entropy model (UEM) shown in Equation 2. Then UEM is optimized using the rate loss only for entropy coding. The results ("A8: w/o CEM in EC") demonstrate the significant rate savings of CEM, which is not considered in previous works [1, 10] with VQ.

Subjective comparison. In Figure 8 and Figure 9, we provide the subjective comparison between our method and VVC. Our reconstruction has a slightly better subjective quality with a smaller bpp.

References

[1] Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc V Gool. Soft-to-hard vector quantization for end-to-end learning compressible representations. *Advances in neural information processing systems*, 30, 2017. 1, 3

[2] Johannes Ballé, Philip A Chou, David Minnen, Saurabh Singh, Nick Johnston, Eirikur Agustsson, Sung Jin Hwang, and George Toderici. Nonlinear transform coding. *IEEE Journal of Selected Topics in Signal Processing*, 15(2):339–353, 2020. 1, 2

[3] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018. 1, 2

[4] Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja. Compressai: a pytorch library and evaluation platform for end-to-end compression research. *arXiv preprint arXiv:2011.03029*, 2020. 2

[5] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro 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*, pages 7939–7948, 2020. 2

[6] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 5

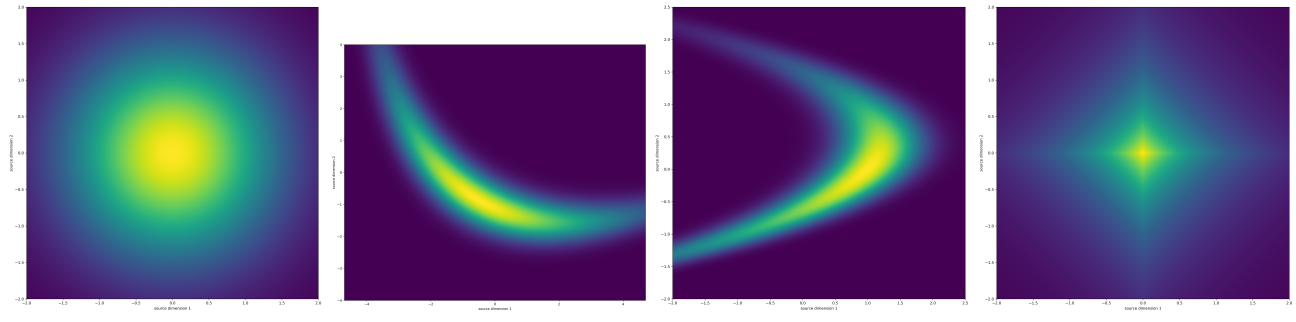
[7] David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. *Advances in neural information processing systems*, 31, 2018. 2

[8] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016. 2, 8

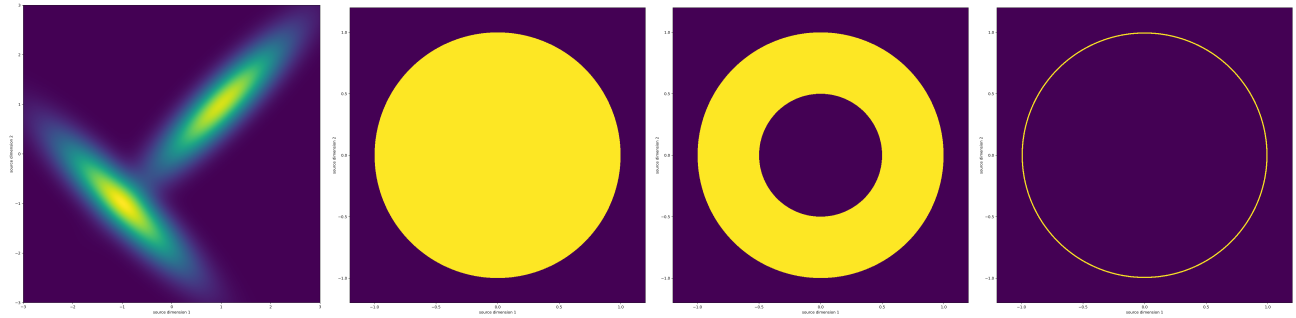
[9] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 1

[10] Xiaosu Zhu, Jingkuan Song, Lianli Gao, Feng Zheng, and Heng Tao Shen. Unified multivariate gaussian mixture for efficient neural image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17612–17621, 2022. 2, 3

[11] Renjie Zou, Chunfeng Song, and Zhaoxiang Zhang. The devil is in the details: Window-based attention for image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17492–17501, 2022. 2



(a) Isotropic Gaussian (b) Banana (c) Boomerang (d) Laplace



(e) Gaussian Mixture (f) Shpere0 (g) Shpere50 (h) Shpere99

Figure 2. Visualization of 2-d sources. Yellow means high probability density and purple means low probability density.

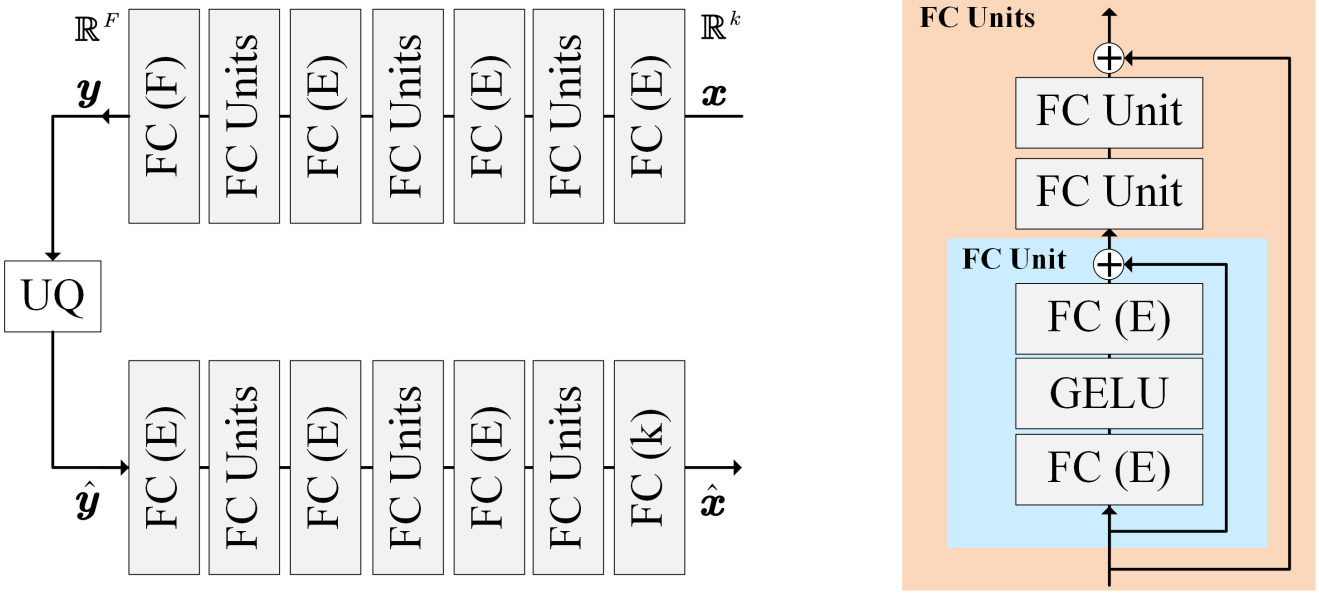


Figure 3. Architecture of NTC on toy sources. "FC (E)" refers to a fully-connected layer with E output channels. F is equal to the data dimension k. $E = 128$ for 2-d distributions, and $E = 384$ for 4-d, 8-d and 16-d distributions. "UQ" is the uniform scalar quantization. "GELU" is the Gaussian Error Linear Units [6].

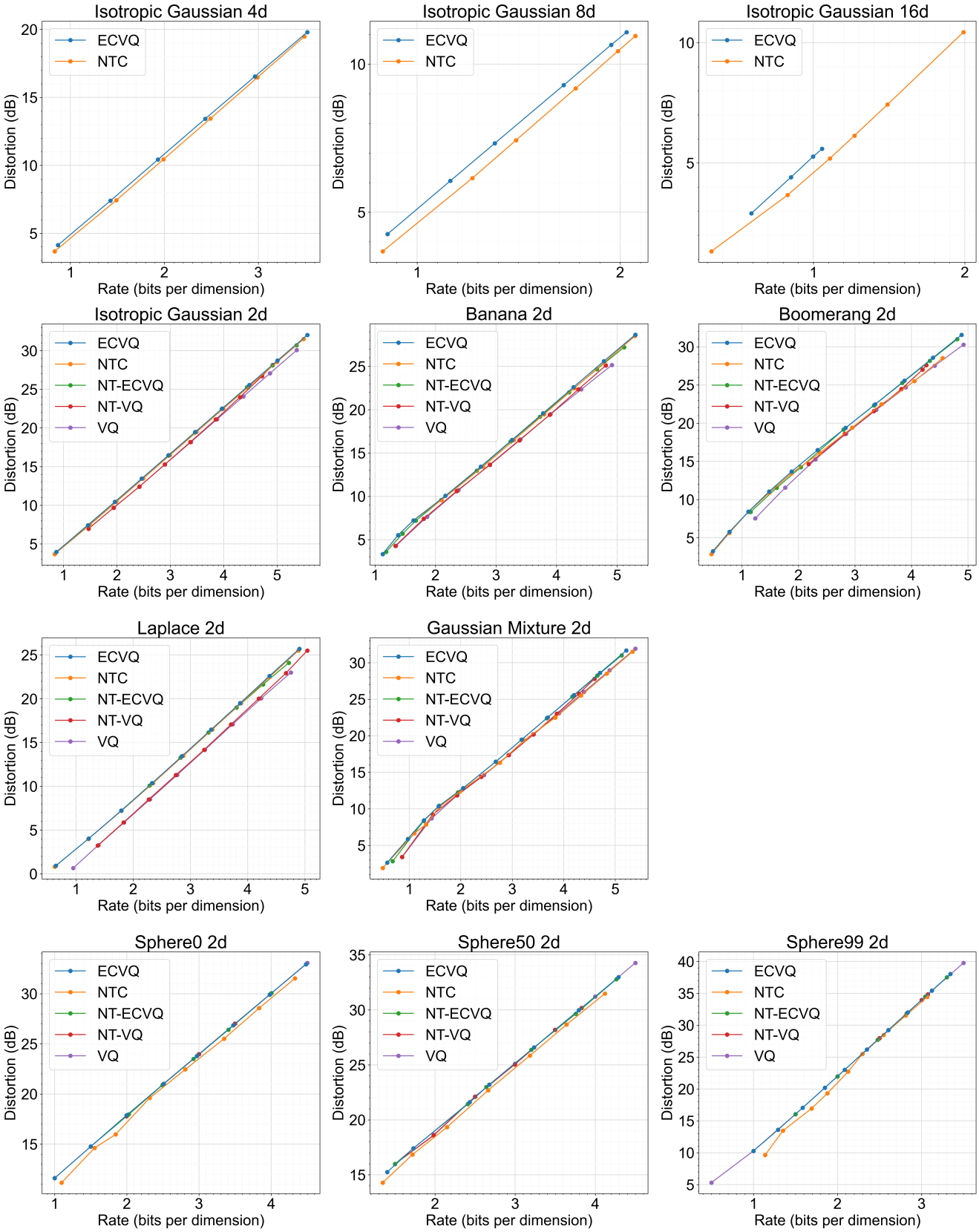


Figure 4. RD curves on toy sources.

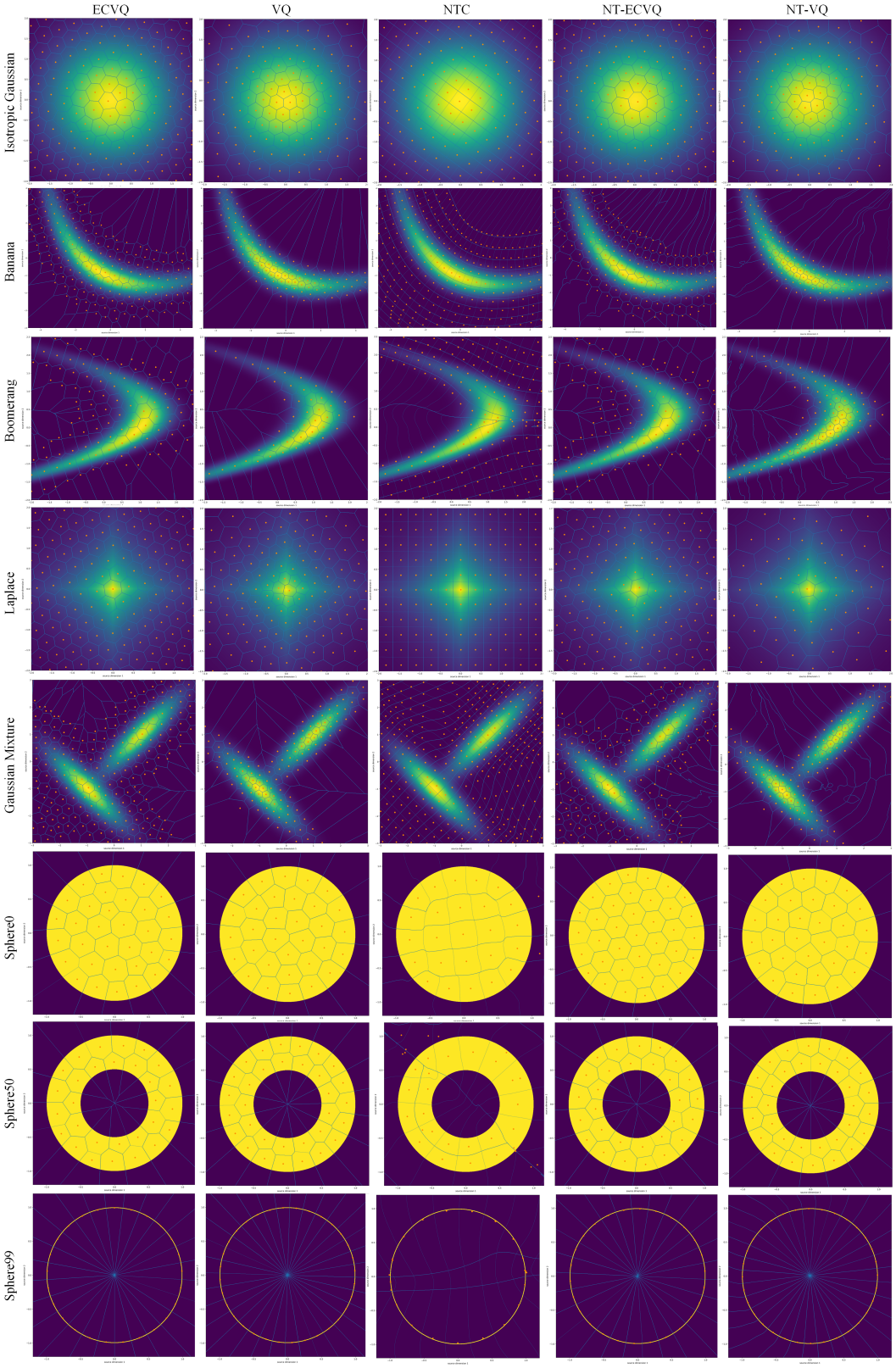


Figure 5. Visualization of quantization results.

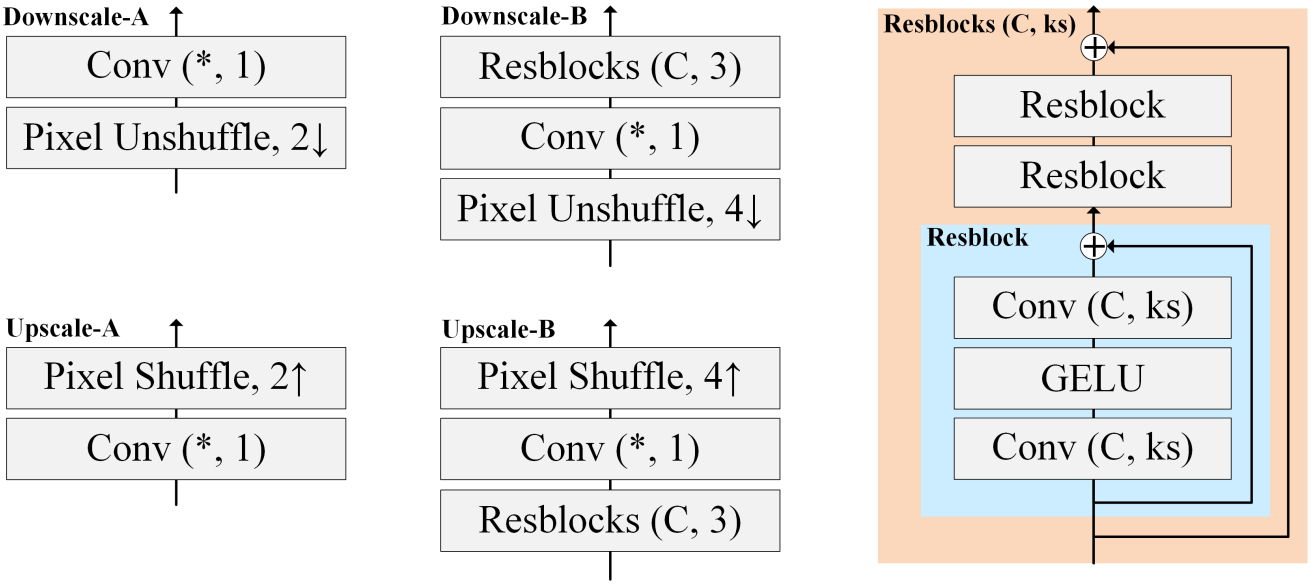


Figure 6. Architecture of Downscale and Upscale layers. "Conv (C, ks)" refers to a convolutional layer with C output channels and a kernel size of ks. "Conv (*, 1)" refers to a 1×1 convolutional layer with "*" output channels which adaptively changes for requirements. Pixel Unshuffle is a space-to-depth layer (for downsampling) and Pixel Shuffle is a depth-to-space layer (for upsampling) [8]. (Right) "Resblocks (C, ks)" refers to the Resblock layer consisting of multiple "Conv (C, ks)". C is set to 192 for all rate points.

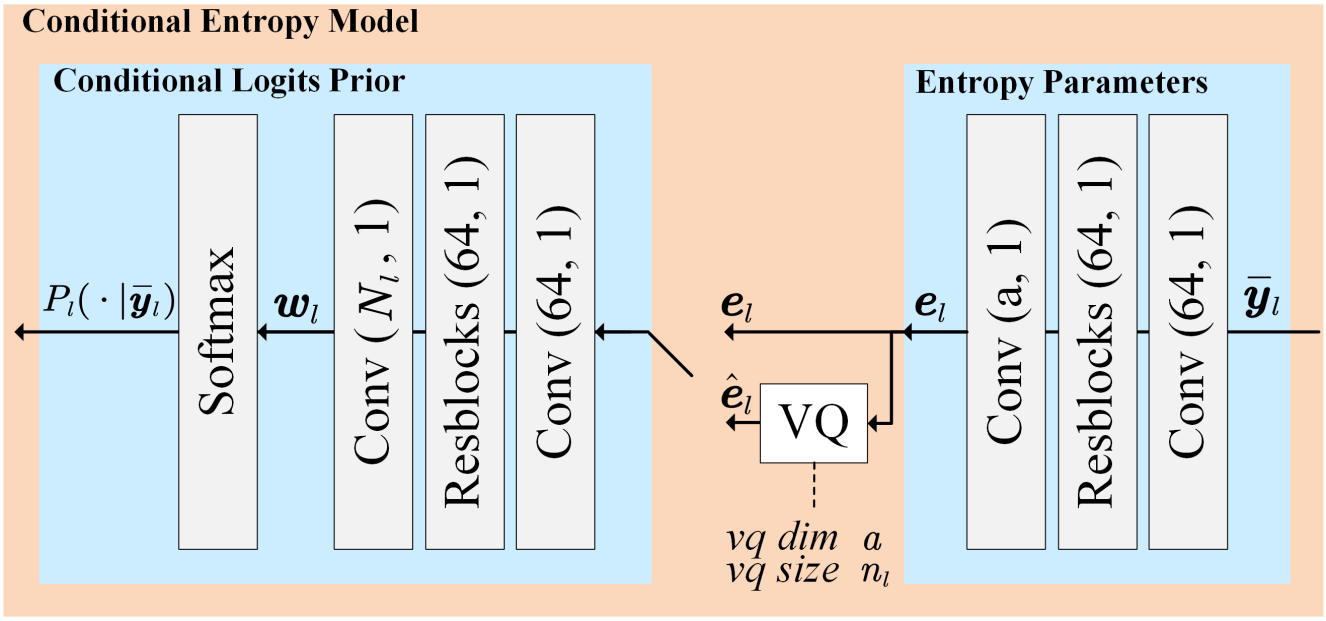


Figure 7. Architecture of Conditional Entropy Model (CEM).

(a) Ours. Bpp=0.2626, PSNR=35.23dB.

(b) VVC. Bpp=0.2761, PSNR=35.32dB.

Figure 8. Comparison between the proposed method and VVC on “Kodim07.png”.

(a) Ours. Bpp=0.6574, PSNR=31.45dB.

(b) VVC. Bpp=0.7051, PSNR=31.60dB.

Figure 9. Comparison between the proposed method and VVC on “Kodim05.png”.