Fitted Neural Lossless Image Compression

Supplementary Material

A. Method and Experimental Details

Pre-fitter The ARM pre-fitter consists of three sub-modules designed to predict the distribution at sub-images 2, 3, and 4 (cf. Figure 2). The image is rearranged in the order of sub-image autoregression, transforming a 3 $\times$ H $\times$ W image into a 12 $\times$ H/2 $\times$ W/2 format. For each sub-module, a 3x3 convolution is first applied to extract 32-channel features, followed by a residual block [1] with 32 channels. Finally, a 3$\times$3 convolution layer is used to predict the distribution parameters. ReLU activations are employed throughout the process.

Overfitter The architecture of the overfitter is fixed for all test images. We use a four-level hierarchy of latent variables. We use the learned upsampler proposed by [5] with a kernel size of 8. Following the Cool-Chic series [4], we model the distribution of latent variables using a Laplace distribution.

For the latent ARM, we employ two linear layers with ReLU activations and 24 channels, along with a skip connection. The synthesizer consists of a 3$\times$3 convolution layer followed by two residual blocks, with each block containing a single convolution layer, unlike the two layers used in the ARM pre-fitter. The hidden layers have 24 channels with GELU [2] activations.

To train the overfitter, we use the same strategy proposed by C3 [3] and the implementation available at https://github.com/Orange-OpenSource/Cool-Chic. For entropy coding of the latent variables, we use the entropy coder from FSAR [6], available at https://github.com/alipay/Finite_State_Autoregressive_Entropy_Coding/tree/main/cbench. Additionally, we found that varying the initialization of a leads to different results for each image. Consequently, we performed rate optimization by training twice with initial values of $a = 1$ and $a = 0.01$, selecting the best model for each case, which will increase the encoding time without influencing decoding speed. For a fair comparison, all other methods in Table 2 and Table 3 employ a similar rate optimization strategy. They are also overfitted twice and the best result for each image is selected.

B. Compared Methods

- PNG: We use the Pillow library, version 10.3.0. https://github.com/python-pillow/Pillow
- JPEG-LS: We use the pillow-jpls library, version 1.3.2. https://github.com/planetmarshall/pillow-jpls

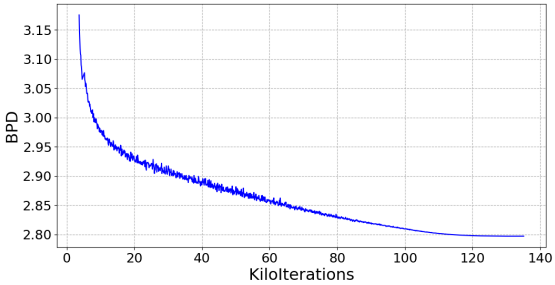

Figure 4. Part of the training loss on the Kodak dataset.

- JPEG2000: We use the open-jpeg library, version v2.4.0. https://www.openjpeg.org/
- WebP: We use libwebp, version 0.6.1. https://github.com/webmproject/libwebp We use the slowest preset with -z 9.
- FLIF: We use flif, version 0.3. https://github.com/FLIF-hub/FLIF We use the slowest preset with -E 100.
- JPEG-XL: We use libjxl, version 0.9.0. https://github.com/libjxl/libjxl We use the slowest preset with -e 9.
- L3C: We use the official implementation from https://github.com/fab-jul/L3C-PyTorch.
- LLICTI: We use the official implementation from https://github.com/kamisli-icpl/LLICTI.
- LC-FDNet: We use the official implementation from https://github.com/myideaisgood/LC-FDNet.
- DLPR: We use the official implementation from https://github.com/BYchao100/Deep-Lossy-Plus-Residual-Coding.
- ArIB-BPS: We use the official implementation from https://github.com/ZZ022/ArIB-BPS.

C. Additional Results

We visualize the training loss on the Kodak dataset with $a = 1$ initialization in Figure 4. FNLIC could achieve similar performance with fewer training steps.

D. Discussion on Encoding Time

Due to the overfitting process, FNLIC requires significantly longer encoding time compared to pre-fit-based methods - a common limitation shared by INR-based methods. For instance, ArIB-BPS [7] requires only 14 seconds to encode a

768×512 image, which is substantially faster than FNLIC. However, as demonstrated in Tables 1 and 2, this overfitting process enables remarkable improvements in decoding efficiency. When considering FNLIC's compression ratio of 3.36 without overfitting on the Kodak dataset, we observe a consistent trade-off between compression ratio and decoding complexity, where higher compression ratios typically demand greater decoding complexity.

Table 2 reveals that our overfitting strategies achieve a significant BPD gain of 0.48 while introducing negligible decoding complexity overhead. This enables FNLIC to match the compression ratios of LC-FDNet, while achieving more than 45$\times$ faster decoding inference time. The increased encoding time directly contributes to this substantial improvement in decoding efficiency. Notably, in many practical scenarios where an image is encoded once but decoded multiple times by numerous users, the imbalance between encoding and decoding usage makes the increased encoding time an acceptable trade-off for improved compression ratios. This aligns with common practices in traditional codecs where complex encoders are routinely employed to enhance compression performance.

Furthermore, as illustrated in Figure 4, the majority of compression ratio gains occur during the early stages of overfitting, allowing for flexible trade-offs between encoding time and compression ratios. This characteristic enables practical optimization based on specific application requirements.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1

[2] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 1

[3] Hyunjik Kim, Matthias Bauer, Lucas Theis, Jonathan Richard Schwarz, and Emilien Dupont. C3: High-performance and low-complexity neural compression from a single image or video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9347–9358, 2024. 1

[4] Théo Ladune, Pierrick Philippe, Félix Henry, Gordon Clare, and Thomas Leguay. COOL-CHIC: Coordinate-based low complexity hierarchical image codec. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13515–13522, 2023. 1

[5] Thomas Leguay, Théo Ladune, Pierrick Philippe, Gordon Clare, Félix Henry, and Olivier Déforges. Low-complexity overfitted neural image codec. In *IEEE 25th International Workshop on Multimedia Signal Processing*, pages 1–6. IEEE, 2023. 1

[6] Yufeng Zhang, Hang Yu, Jianguo Li, and Weiyao Lin. Finite-state autoregressive entropy coding for efficient learned lossless compression. In *The Twelfth International Conference on Learning Representations*, 2024. 1

[7] Zhe Zhang, Huairui Wang, Zhenzhong Chen, and Shan Liu. Learned lossless image compression based on bit plane slicing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27579–27588, 2024. 1