IEEE International Conference on Image Processing

27 - 30 October 2024 | Abu Dhabi, UAE

Redefining Visual Quality: The Impact of Loss Functions on INR-Based Image Compression

Lorenzo Catania, Dario Allegra

Department of Mathematics and Computer Science, University of Catania

lorenzo.catania@phd.unict.it, dario.allegra@unict.it

Implicit Neural Representations

Implicit Neural Representations (INR) is an emerging paradigm for data representation in which a signal is interpreted as a function from coordinates to samples: $I(i_{x,y}) = (R_c, G_c, B_c)$.

A neural network is then overfitted to this function. If the purpose is to compress data, then the parameters are compressed and transmitted. The signal is therefore reconstructed by inference through the neural network.

A fundamental step when defining an INR pipeline is to choose a proper loss function that represents the distortion between the original signal and the one reconstructed by the network. In the case of images, the most common loss is the L2 mean, also known as *Mean Square Error* (MSE), and compression distortion is commonly evaluated by using the traditional *Peak Signal-to-Noise Ratio* (PSNR). However, these simple metrics may not match the perceived quality of decoded images,

Contributions

- The evaluation of five functions as losses in three SotA INR-based image compression, a paradigm in which the choice of the loss function is fundamental yet nearly unexplored. Results are presented in terms of averaged quantitative results, visual fidelity of specific samples and appearance of artifacts.
- We examine in depth the potential of adding structural factors in loss functions when training INRs for image compression, proposing recommendations that consistently improve the state-of-the-art in terms of perceptive metrics while maintaining a high PSNR. Also, decoded images benefit reduced artifacts and better visual fidelity.
- The code used for the experiments and the full results are publicly released to the community on GitHub: https://github.com/INRAnalysis-ICIP24.

Network architectures

NIF [1]: A SIREN architecture which takes positional features as input. A modulation module alters the period of each activation based on the coordinates of the pixel. Also, the number of features on each layer is reduced proportionally to its depth. This technique has been empirically proved to enhance the bitrate/distortion ratio.

COOL-CHIC v1 [2]: A multi-layer perception with ReLU activations which takes latent grid features as input. The purpose of this architecture is to reduce the decoding complexity of the method limiting the amount of the operations needed to decode each pixel. An auto-regressive probability model is added to estimate the parameters' distribution and an entropy factor is added to the loss function to minimize the parameters' entropy.

COOL-CHIC v2 [3]: An evolution of [2] which adds convolutional layers to the original architecture and adaptive upsampling instead of fixed one to upsample grid features.

Methodology

We propose the following five loss functions to be used during training. In the formula, y is the original sample, $\hat{y}$ is the reconstructed sample and N is the number of pixels.

L1: Also known as *Mean Average Error*, it is the absolute difference between two signals.

$$L1(y,\hat{y}) = \frac{\sum |y-\hat{y}|}{N}$$

MSE: The most common loss in INR-based compression.

$$MSE(y,\hat{y}) = \sqrt{\frac{\sum (y-\hat{y})^2}{N}}$$

Compared to L1, this loss function penalizes large errors and is less sensitive to small differences.

LogCosh: In practice, the following approximation of $log(cosh(x))$ is used to avoid infinite growth for large differences:

$$Lc(y,\hat{y}) = \frac{\sum((y-\hat{y}) + sp(2*(y-\hat{y})) - ln(2))}{N}$$

Where sp is the SoftPlus function and is calculated as:

$$sp(x) = ln(1+e^x)$$

It exhibits a shape similar to L1 for large values and MSE for small values, obtaining the best of both worlds.

L1SSIM (L1 + SSIM): A combination of L1 and SSIM:

$$L1SSIM(y,\hat{y}) = (1-\alpha) * L1(y,\hat{y}) + \alpha * (1 - SSIM(y,\hat{y}))$$

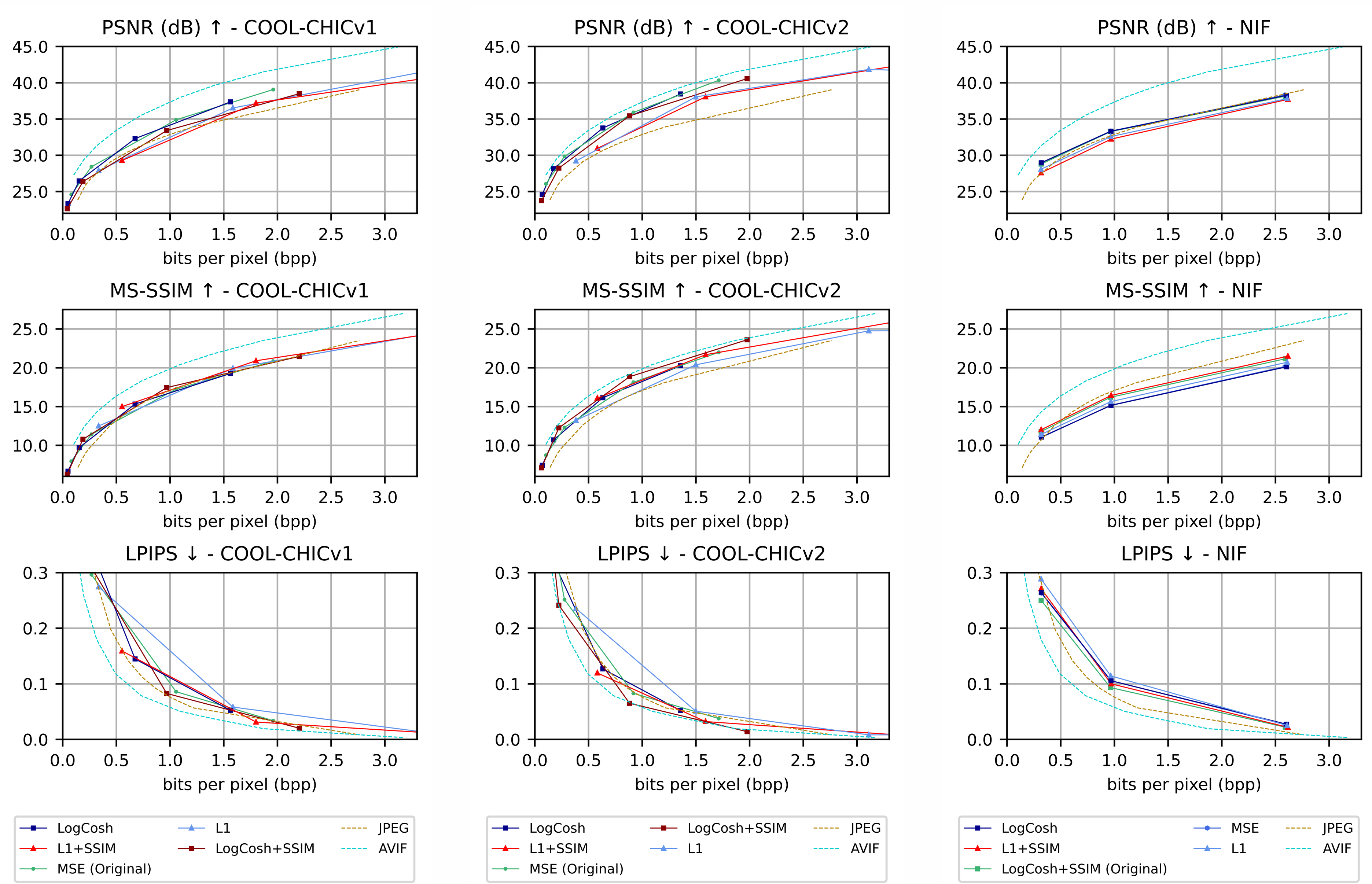
In this case, the α factor increases the influence of SSIM on the values and decreases the influence of L1.

LcSSIM (LogCosh + SSIM): A combination of LogCosh and SSIM [4]:

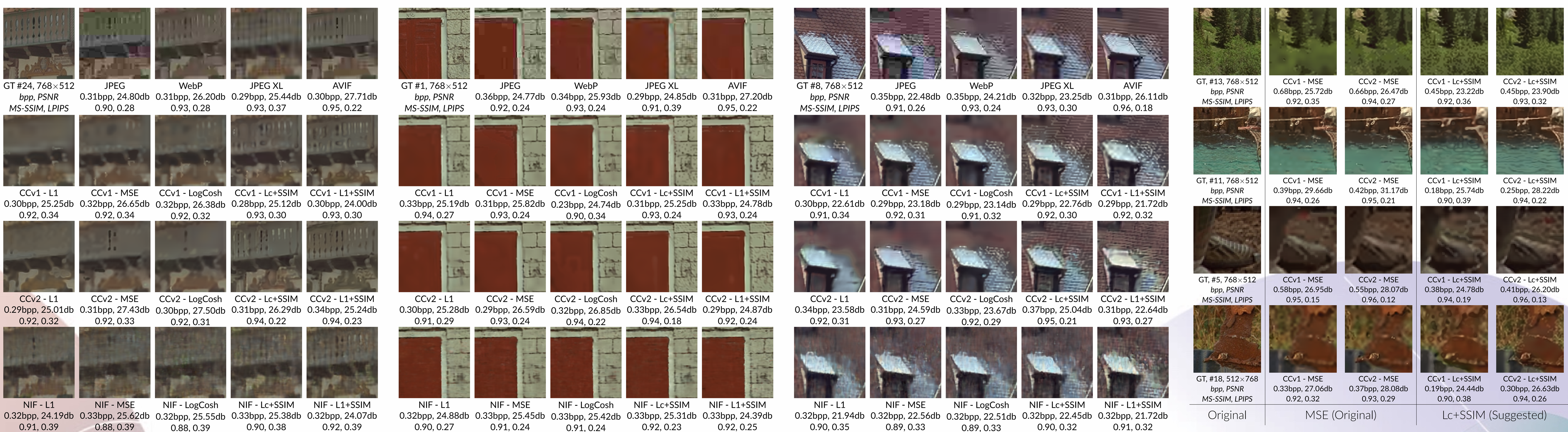
$$LcSSIM(y,\hat{y}) = Lc(y,\hat{y}) + \alpha * (1 - SSIM(y,\hat{y}))$$

Where α is a factor which scales the SSIM, as it is usually much bigger than LogCosh and may dominate the loss value. First proposed in [1], it aids the training process to consider structural information on the image instead of optimizing each point independently

Quantitative results



Visual comparisons



References

[1] Lorenzo Catania and Dario Allegra. NIF: a fast implicit image compression with bottleneck layers and modulated sinusoidal activations. In *ACM International Conference on Multimedia*, 2023.

[2] Théo Ladune, Pierrick Philippe, Félix Henry, Gordon Clare, and Thomas Leguay. COOL-CHIC: Coordinate-based low complexity hierarchical image codec. In *IEEE International Conference on Computer Vision*, 2023.

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

[4] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004.