This CVPR Workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

A Compression Objective and a Cycle Loss for Neural Image Compression

Caglar Aytekin, Francesco Cricri, Antti Hallapuro, Jani Lainema, Emre Aksu and Miska Hannuksela
Nokia Technologies
Hatanpaan Valtatie 30, Tampere, Finland
caglar.aytekin@nokia.com

Abstract

In this manuscript we propose two objective terms for neural image compression: a compression objective and a cycle loss. These terms are applied on the encoder output of an autoencoder and are used in combination with reconstruction losses. The compression objective encourages sparsity and low entropy in the activations. The cycle loss term represents the distortion between encoder outputs computed from the original image and from the reconstructed image (code-domain distortion). We train different autoencoders by using the compression objective in combination with different losses: a) MSE, b) MSE and MS-SSIM, c) MSE, MS-SSIM and cycle loss. We observe that images encoded by these differently-trained autoencoders fall into different points of the perception-distortion curve (while having similar bit-rates). In particular, MSE-only training favors low image-domain distortion, whereas cycle loss training favors high perceptual quality.

1. Introduction

Traditional image compression methods are mostly based on transform-coding such as BPG [16] and JPEG [17]. With the recent advances in deep learning, neural networks have been applied to image compression with promising results.

Neural image compression can be applied in a hybrid system comprising a traditional codec and a neural network used either within the traditional codec (e.g. replacing some traditional filters as in [12]) or after it (e.g. a post-processing filter) [8], [7].

Another approach is to design an image codec solely based on neural networks – this is commonly referred to as *end-to-end learned* approach. Recently end-to-end learned approaches have shown considerable success [4], [5], [15], [18], [19]. The main research topics in this field include distortion/perception loss functions [19], activation binarization/quantization [4], [18], rate loss functions [5], [15], spatial/channel importance learning [14].

In this paper, we describe a method for the end-to-end learned approach, and we address two of the research topics above, namely rate and perception loss functions. First, we propose a rate loss based on a sparsity metric, that we refer to as *compression objective*. This loss helps obtaining very sparse codes which are highly compressible. Second, we propose a perception loss which does not require any additional neural network (thus avoiding significant increase in memory and computational complexity at training stage). We refer to this perception loss as *cycle loss*.

We used the image codecs presented in this paper to participate to the 2019 Challenge on Learned Image Compression. In particular, our submission names were NTCodec2019C, NTCodec2019CM, and NTCodec2019CC.

2. Related Work

In [15] a rate loss is introduced which penalizes spatial deviations in the code, thus helping to achieve low bit-rates when a context adaptive entropy coder is used. In [4], a similar loss was introduced for one-dimensional data, as the encoder's output is 1-D. In [5] a differentiable approximation of entropy was used as a rate loss. Our proposed compression term helps obtaining very sparse codes which directly reduce entropy and may indirectly increase the chance of low spatial variance.

This paper proposes a loss term which encourages compressibility of the encoder's output by achieving sparsity. In [2], we proposed to use part of the term proposed in this paper, but applied on neural network's weights. In [3], we proposed a development of the compressibility term, again for compressing neural network's weights.

Regarding losses for achieving high perceptual visual quality, one approach is to use metrics other than mean-squared error (MSE). For example, in [19], metrics such as multi-scale structural similarity (MS-SSIM) and peak signal-to-noise ratio human visual system (PSNR-HVS) were optimized in order to improve visual quality. Another approach is to use an additional neural network to compute a perceptual quality metric. In [15], [1], a generative adversarial network (GAN) was used in order to obtain images with better visual quality. Another common strategy is to use a network pre-trained on a classification task using a big dataset, such as a VGG network on ImageNet. For example, in [13] the authors combined an adversarial loss with the MSE computed on the VGG features extracted from ground-truth and predicted images, for the task of super-resolution. In [20], the authors study different options for obtaining perceptual metrics using deep neural networks, and conclude that even networks pre-trained in unsupervised or in self-supervised way provide similarly performing metrics as those provided by supervised networks such as classifiers.

The approaches discussed above for measuring the visual quality have several drawbacks. MS-SSIM and PSNR-HVS are hand-crafted metrics, and learning-based approaches require additional neural networks which increase the computational and memory complexity of the training stage. In this manuscript, we propose to use the encoder part of our autoencoder structure as a high level-semantic feature extractor and introduce a cycle loss where we minimize the MSE between original image's semantic features and reconstructed image's semantic features. We realize that this helps us to achieve visually pleasing images.

The concept of cycle loss for training neural networks was introduced in [21] in the context of GANs. However, the authors were using an additional generator network to map back from output to input domain. Instead, we map back to only the code domain, and we already have the mapping function – it is the encoder network. A similar idea was explored also in [11], in the context of disentangling factors of variation using variational autoencoders. However, in that work the backward cycle is applied in order to map two different reconstructed images (obtained from a combination of same sampled unspecified latent embedding and different specified latent variables) to similar unspecified latent embeddings. In our case instead, the backward cycle is applied in order to map a reconstructed image back to the code from which it was generated.

3. Proposed Method

The proposed image compression framework is based on neural autoencoders trained with a compression objective together with a task loss.

3.1. Compression Objective

The compression objective is based on a term that encourages sparsity and another term that encourages small non-zero values. The loss is defined as follows.

$$L_{comp}(x) = \frac{|x|}{||x||} + \alpha \frac{||x||^2}{|x|} \tag{1}$$

The first part of the compression loss in Eq. 1, $\frac{|x|}{||x||}$, is adopted from the work [10] and is a measure of the sparsity in a signal. We call this sparsity term of the compression objective. The sparsity term is independent of the values of non-zeros in the signal. For example a vector $[0, 0, 500, 500]$ and $[0, 0, 0.1, 0.1]$ would have exactly the same sparsity value and large values are not penalized. However, it is usually a good practice to have reasonably small values in machine learning literature to avoid exploding gradients and also to act as a regularization. Because of this, we add another factor to the compression loss which favors small non-zero values in a signal – this is the second part: $\frac{||x||^2}{|x|}$. The weight α in Eq. 1, acts as a regularizer between the sparsity term and the squeezing term.

3.2. Task Loss

We have investigated three different task losses for training neural autoencoders. The first task loss we have used is the mean squared error that is defined as follows.

$$L_{mse}(I, \hat{I}) = \frac{1}{N} \sum_{i}^{N} (I(i) - \hat{I}(i))^2 \tag{2}$$

In Eq. 2, I and $\hat{I}$ are the original and the reconstructed image, respectively. Although the MSE is a direct indicator of the per-pixel distortion measure, it has been observed in the literature ([20]) that lower distortion does not necessarily mean better perceptual quality. Therefore other metrics should be used in order to increase perceptual quality of the reconstructed image. One of these metrics is structural similarity measure (SSIM) defined as follows.

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{3}$$

SSIM in Eq. 3 is calculated on blocks, μ_x and σ_x stand for mean and standard deviation of block x and c_1 and c_2 are variables to stabilize low value denominator. A multi-scale version of SSIM (MS-SSIM) is widely used and computed over multiple scales. Since MS-SSIM is a quality measure (in range $[0, 1]$), we use it as a loss in the following way:

$$L_{ms-ssim}(x, y) = \frac{1 - mssim(x, y)}{2} \tag{4}$$

Other perceptual losses are based on learned networks, such as the VGG-loss or adversarial losses, which however require additional neural networks. We propose a perceptual loss which does not incur in additional networks. We use the encoder part (E) of the autoencoder as feature extraction. In order to obtain the features, we freeze the encoder part (E_f) and calculate the features for the original and the reconstructed image and minimize the MSE between these features as illustrated in Fig. 1. We refer to this as the cycle loss and is formulated as follows.

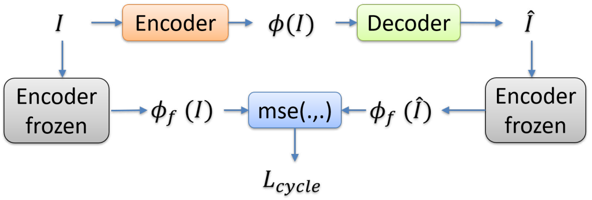

Figure 1. Cycle Loss for Perceptual Quality.

$$L_{cycle}(I,\hat{I}) = L_{mse}(E_f(I), E_f(\hat{I})) \tag{5}$$

In particular, following the common parlance in the context of cycle losses, our cycle loss ensures the so-called backward cycle consistency ($c \rightarrow \hat{I} \rightarrow \hat{c}$), whereas the forward cycle consistency is ensured by the MSE on the image domain ($I \rightarrow c \rightarrow \hat{I}$).

We train three different autoencoders by using the compression objective in combination with the following losses: a) MSE, b) MSE and MS-SSIM, and c) MSE and MS-SSIM and cycle loss.

3.3. Neural Network Architecture

We use a neural autoencoder for image compression. The encoder consists of three blocks where each block consists of a strided convolution layer followed by a residual block as illustrated in Fig. 2. Finally there is a 1x1 convolutional layer followed by a sigmoid to map the values between 0 and 1. The compression loss is applied to the output of this sigmoid activation in order to drive most of the activations close to zero.

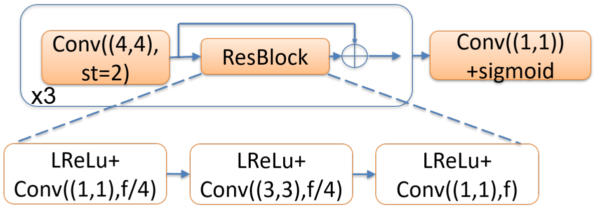

Figure 2. Encoder Structure.

The decoder consists of three blocks where each block consists of an up-sampling deconvolution layer followed by a residual block as illustrated in Fig. 3. Finally there is a 1x1 convolutional layer followed by a sigmoid. Note that the input to the CNN is also normalized to have values between 0 and 1.

Each residual block consists of 3 sub-blocks consisting of a leaky ReLU activation and a convolutional layer. Convolutional layers are 1x1, 3x3 and 1x1 respectively, following the approach of [9]. Filter numbers of each convolutional layer are one fourth, same and one-fourth of the input channel number to the residual block. We do not use any batch-normalization layers.

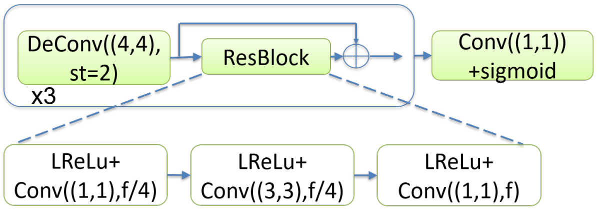

Figure 3. Decoder Structure.

3.4. Activation Binarization

In order to make the most out of the compression, we binarize the output of the encoder. Note that the output of the encoder is in the interval $[0,1]$ already. For the forward pass, we use simple rounding operation and for the backward pass we use the straight-through estimator [14].

3.5. Post-Training Encoder Optimization

After the training, post-training encoder optimization is utilized where the encoder is optimized for each test image by simply fine-tuning the pre-trained autoencoder while keeping the decoder frozen.

3.6. Lossless Coding

After the post-training encoder optimization, the encoder outputs are lossless-encoded by context adaptive binary entropy codec (CABAC).

4. Experimental Results

4.1. Implementation Details

We trained the autoencoder on the CLIC training dataset. In particular, we used 128x128 half-overlapping crops, on which we applied random horizontal flipping as data augmentation. The neural networks were trained with a combination of task loss and compression loss.

For MSE based training, we have used the following loss function

$$L(x) = L_{mse}(I,\hat{I}) + \gamma L_{comp}(c) \tag{6}$$

where c is the encoder output and $\gamma = 1e{-}04$. This loss was used for the CLIC submission NTCodec2019C. For MSE and MS-SSIM based training, we have used the following loss function.

$$L(x) = L_{mse}(I,\hat{I}) + \lambda L_{ms-ssim}(I,\hat{I}) + \gamma L_{comp}(c) \tag{7}$$

where $\lambda = 0.1$ and $\gamma = 2.5e{-}04$. This loss was used for the CLIC submission NTCodec2019CM. For MSE, MS-SSIM and cycle based training, we have used the following loss function.

$$\begin{aligned} L(x) = L_{mse}(I,\hat{I}) + \lambda_1 L_{ms-ssim}(I,\hat{I}) + \\ \lambda_2 L_{cycle}(I,\hat{I}) + \gamma L_{comp}(c) \end{aligned} \tag{8}$$

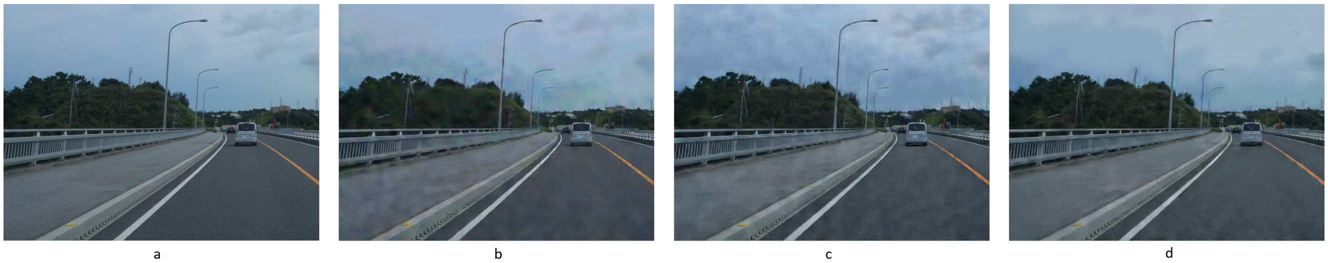

Figure 4. (a) Original Image, Reconstructed images by using neural networks that are trained with losses in Eq. 6 (b), Eq. 7 (c), Eq. 8 (d).

where $\lambda_1 = 0.1$, $\lambda_2 = 0.01$ and $\gamma = 3e-04$. This loss was used for the CLIC submission NTCodec2019CC.

The hyperparameters were empirically selected in order to satisfy the 0.15 bpp (bits per pixel) and to obtain reasonable performance at this bit rate.

For all the trainings we use Adam optimizer with learning rate $2e-04$, we halve the learning rate every 10 epochs. We stop halving the learning rate after epoch 50 and train the neural networks for 200 epochs in total. Each epoch consists of 900 iterations and we have used batch size 64. The training takes about 10 hours in a single GPU in NVIDIA DGX-1 computing cluster. The autoencoder size is 28 MB, where the decoder part is about 14 MB – this is reasonably small for efficient inference.

4.2. Results

In Table 1, we share the results on the CLIC validation set (PSNR, MS-SSIM and bpp) for neural autoencoders trained with the losses in Equations 6, 7 and 8.

Table 1. Performance of neural networks trained with losses in Equations 6, 7 and 8 on CLIC validation set, corresponding to submission names NTCodec2019C, NTCodec2019CM and NTCodec2019CC.

Loss	**PSNR**	**MS-SSIM**	**bpp**
Eq. 6	27.90	0.915	0.145
Eq. 7	27.43	0.921	0.145
Eq. 8	26.98	0.921	0.148

It can be observed from Table 1 that at similar bit-rates the network trained with only MSE loss obtains the best PSNR, the network trained with MSE and MS-SSIM jointly results into nearly half dB loss in PSNR while increasing the MS-SSIM. When cycle loss is added to MSE and MS-SSIM, although this results into a further reduction in PSNR, MS-SSIM stays the same.

Next, we compare the visual quality of decoded images by each method. In Fig. 4, we share an image that is encoded/decoded by different methods. We see clearly that the image encoded/decoded with the neural network trained by cycle loss (d) has better visual quality than others. Referring to the images in Fig. 4 b, c, d, the corresponding PSNR values are 32.10 dB, 31.19 dB and 30.80 dB, respectively, whereas the obtained bpp values are 0.093, 0.093 and 0.091, respectively. Clearly the network trained only with MSE obtains better PSNR performance and as we introduce more losses, PSNR is reduced. However, although the worst PSNR comes from the model that is trained also with cycle loss, we see a superior perceptual quality from this image.

This result is interesting, yet follows the previous findings in the literature. For example in [6] it was discussed that for non-invertible problems, a perception-distortion curve is evident which defines a boundary between a region that is possible to obtain and a region that is impossible to obtain. An impossible point is for example the perfect reconstruction. This is clearly impossible to obtain if the problem is non-invertible (i.e., if there is a loss of information that cannot be recovered in any way). Therefore, the points on the separating curve in [6] shows a trend where the perceptual error is inversely proportional to distortion. In our case this means that perceptual quality is inversely proportional to PSNR.

Another observation from the above experiment is that the model trained with MS-SSIM and MSE does not obtain clearly observable higher perceptual quality compared to the model that is only trained with MSE. Therefore, this also leads to re-thinking the common belief that MS-SSIM is more perception-friendly loss than MSE. At least it can be deduced from the above experiments that MS-SSIM may not be enough to obtain a good perceptual quality on its own, whereas adding our cycle loss leads to clear perceptual improvements.

4.3. Conclusion

We propose a compression loss which helps obtaining very sparse codes. As another contribution, we propose cycle loss which helps achieving images with better perceptual quality without introducing any additional neural network than the autoencoder itself to calculate the perceptual loss.

References

[1] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, and L. V. Gool. Generative adversarial networks for extreme learned image compression. In *arXiv:1804.02958*, 2018.

[2] C. Aytekin, F. Cricri, and E. Aksu. Compressibility loss for neural network weights. In *arXiv:1905.01044*, 2019.

[3] C. Aytekin, F. Cricri, T. Wang, and E. Aksu. Response to the call for proposals on neural network compression: Training highly compressible neural networks. ISO/IEC JTC1/SC29/WG11 MPEG2019/m47379, Mar. 2019. Input contribution to MPEG Neural Network Representations.

[4] C. Aytekin, X. Ni, F. Cricri, J. Lainema, E. Aksu, and M. Hannuksela. Block-optimized variable bit rate neural image compression. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop and Challenge on Learned Image Compression*, 2018.

[5] J. Ball, V. Laparra, and E. P. Simoncelli. End-to-end optimized image compression. In *International Conference on Learning Representations*, 2017.

[6] Y. Blau and T. Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[7] L. Cavigelli, P. Hager, and L. Benini. Cas-cnn: A deep convolutional neural network for image compression artifact suppression. In *International Joint Conference on Neural Networks (IJCNN)*, 2017.

[8] C. Dong, Y. Deng, C. C. Loy, and X. Tang. Compression artifacts reduction by a deep convolutional network. In *International Conference on Computer Vision (ICCV)*, 2015.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, 2018.

[10] O. P. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, pages 1457–1469, 2004.

[11] A. Jha, S. Anand, M. Singh, and V. Veeravasarapu. Disentangling factors of variation with cycle-consistent variational auto-encoders, 04 2018.

[12] C. Jia, S. Wang, X. Zhang, S. Wang, J. Liu, S. Pu, and S. Ma. Content-aware convolutional neural network for in-loop filtering in high efficiency video coding. *IEEE Transactions on Image Processing*, pages 1–1, 2019.

[13] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. pages 105–114, 07 2017.

[14] M. Li, W. Zuo, S. Gu, D. Zhao, and D. Zhang. Learning convolutional networks for content-weighted image compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3214–3223, 2018.

[15] O. Rippel and L.Bourdev. Real-time adaptive image compression. In *nternational Conference on Machine Learning-*, pages 2922–2930, 2017.

[16] 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*, 22(12):1649–1668, Dec 2012.

[17] D. Taubman and M. Marcellin. *JPEG2000 Image Compression Fundamentals, Standards and Practice*. Springer Publishing Company, Incorporated, 2013.

[18] L. Theis, W. Shi, A. Cunningham, and F. Huszr. Lossy image compression with compressive autoencoders. In *International Conference on Learning Representations*, 03 2017.

[19] G. Toderici, D. Vincent, N. Johnston, S. J. Hwang, D. Minnen, J. Shor, and M. Covell. Full resolution image compression with recurrent neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5306–5314, 2017.

[20] R. Zhang, P. Isola, A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. pages 586–595, 06 2018.

[21] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.