Learning-Based Image Compression using Convolutional Autoencoder and Wavelet Decomposition

Pinar Akyazi and Touradj Ebrahimi
Multimedia Signal Processing Group (MMSPG)
Ecole Polytechnique Fédérale de Lausanne
CH 1015, Lausanne, Switzerland
pinar.akyazi@epfl.ch, touradj.ebrahimi@epfl.ch

Abstract

In this paper, a learning-based image compression method that employs wavelet decomposition as a preprocessing step is presented. The proposed convolutional autoencoder is trained end-to-end to yield a target bitrate smaller than 0.15 bits per pixel across the full CLIC2019 test set. Objective results show that the proposed model is able to outperform legacy JPEG compression, as well as a similar convolutional autoencoder that excludes the proposed preprocessing. The presented architecture shows that wavelet decomposition is beneficial in adjusting the frequency characteristics of the compressed image and helps increase the performance of learning-based image compression models.

1. Introduction

Machine learning models have demonstrated efficient solutions to many image processing problems such as object classification and image enhancement. Recent learning-based image compression models have reached and even surpassed the performance of transform-based state-of-the-art image codecs such as JPEG [1], JPEG 2000 [2] and HEVC intra [3]. Instead of using hand-crafted features, learning-based methods rely on a latent representation of the input image through training of similar contents. The latent representation can then be quantized and entropy coded. The compression can become reversible, when the parameters of a decoder are also learned and an end-to-end training of the entire codec is achieved. Models operating at various conditions can be used to regulate bitrate, either directly, or indirectly by making use of a loss function.

Recent learning-based compression methods consist of convolutional neural networks (CNN), recurrent neural networks (RNN) and general adversarial networks (GAN) to construct a latent representation of the image. In [4], authors have used RNN and CNN architectures to implement fully-connected, LSTM-based and convolutional/deconvolutional residual autoencoders. The residual at each iteration can be further encoded to reach higher bitrates and better quality. The models were tested on thumbnail images and then extended to standard resolution images in [5]. Although residual autoencoders are flexible in terms of rate, they have significant computational complexity. GANs were introduced for image compression in [6] where a multiscale adversarial model was trained. Selected regions of images were fully synthesized [7] to achieve good subjective image quality at very low bitrates.

Convolutional autoencoders (CAEs) have been used in [8, 9, 10, 11]. In [9], residual blocks and leaky rectifications were preferred and upsampling was performed using sub-pixel convolutions. Leaky rectifications were also used in [10], and an additional dimensional reduction was performed using principal component analysis. Generalized divisive normalization (GDN) is introduced in [12] and used as the nonlinearity between convolutional layers in [8] and [11]. GDN is a parametric nonlinear transformation that is well-suited for Gaussianizing data from natural images. When followed by uniform scalar quantization, GDN builds a parametric form of vector quantization of the latent representation on the input image vector space. A hyperprior capturing spatial dependencies is introduced in [11] and used in the entropy model for further reduction in rate. This approach is different from the previous methods where rate was estimated using the factorized entropy of the quantized code.

In this paper, a convolutional autoencoder is proposed with a preprocessing step involving a 3-scale wavelet decomposition of all input channels (WCAE). A latent representation of the image is obtained during training using convolutional layers and GDN nonlinearities, and is quantized and entropy coded. WCAE is trained end-to-end, using mean squared error (MSE) in the loss function. The performance of the proposed method is tested on the CLIC2019 validation and test sets, and compared to the performance of JPEG and JPEG 2000, as well as a similar neural network that does not use wavelet decomposition.

The proposed method and network architecture are described in details in the next section, followed by the results and the conclusion in the third and fourth sections, respectively.

2. Proposed Method

The proposed architecture is depicted in Figure 1 with the analysis and synthesis blocks details in Figure 2. The input color image X is separated into non-overlapping patches of dimensions $N \times M$. Before the analysis stage of the convolutional autoencoder, each color channel of an RGB input image patch is first normalized to have $[-1, 1]$ range and then undergoes a 3-scale 2D wavelet transform, where Daubechies-1 wavelets are used. 2D wavelet decomposition is known to be effective in various image processing tasks, compression in particular. When compressing an image using convolutional neural networks, although image features are expected to be learned by the network without ideally any preprocessing, using wavelet decomposition has two distinct benefits. First, the image is separated into its high frequency and low frequency components at different scales, allowing more control over the visual characteristics of the compressed image by giving more or less emphasis on particular frequency components. Second, introducing a wavelet decomposition as a preprocessing step is expected to help the network converge faster.

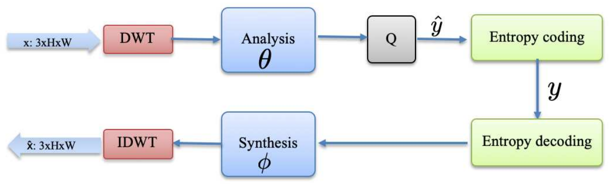

Figure 1: Proposed convolutional autoencoder architecture.

The analysis block is separated into three channels for each scale of the wavelet transform. The coarsest scale has 12 inputs, 4 from each of the 3 color channels. The second and finest scales have 9 inputs each. Convolutional filters of dimensions 3×3 are used at each layer and the number of outputs is doubled once for all scales. The coarsest, second and finest scale inputs are downsampled 2, 3 and 4 times, respectively. Between the convolutional layers, GDN functions provide a nonlinear mapping of the layer outputs. The latent representation is formed by concatenating the 32 outputs of each scale, and has dimensions $32 \times \frac{N \times M}{1024}$. With this representation, the $3 \times N \times M$ input is reduced approximately by a factor of 100, in size.

The latent representation (code) then needs to be quantized. Since quantization is a function with zero gradients

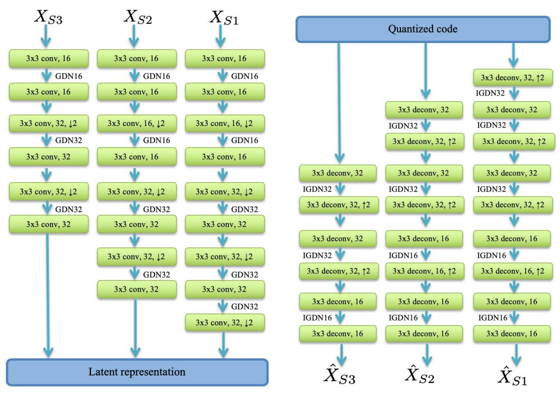

Figure 2: Architecture of the analysis (left) and synthesis (right) blocks. The representation 3x3 conv, [C] depicts a convolutional layer with 3x3 kernels and C outputs. GDN[C] is the generalized divisive normalization function with C inputs. ↑ 2 and ↓ 2 refer to upsampling and downsampling by a factor of 2, respectively. The model has 256606 parameters in total.

almost everywhere, it is replaced by additive uniform noise during training. This is a method preferred at the quantization step of learning-based encoders [8, 10] assuming unit bin size and uncorrelated quantization errors between elements. The quantized values then need to be encoded, where the resulting rate will be a component of the overall loss function. Since the latent representation has been uniformly quantized, an effective entropy coding is expected to reduce the rate optimally. However, the entropy coding also needs to be fully differentiable. The lower bound of the rate, on the other hand, is equal to the entropy of the quantized code [13]. It is therefore sufficient to compute the entropy of the quantized code and use it in the loss function as an estimate of the rate portion.

Once the entropy is computed, the quantized code then goes through a synthesis stage, where deconvolutional filters and upsampling operators are used in parallel with the analysis stage. The quantized code is separated into three equally dimensional components, which represent the coarsest, second and finest scales of the decoded image at the output of the synthesis transform. The three outputs are then merged using an inverse wavelet transform and yield the decoded image. The distortion between the original and decoded image is used in the loss function. The overall loss function of the WCAE is then:

$$J(\theta, \phi; X) = D(X, \hat{X}) + \lambda R \tag{1}$$

where

$$D(X, \hat{X}) = \sum (X - \hat{X})^2 \tag{2}$$

$$R = \sum_i \hat{y}_i \log_2 P_{\hat{y}}(\hat{y}_i) \tag{3}$$

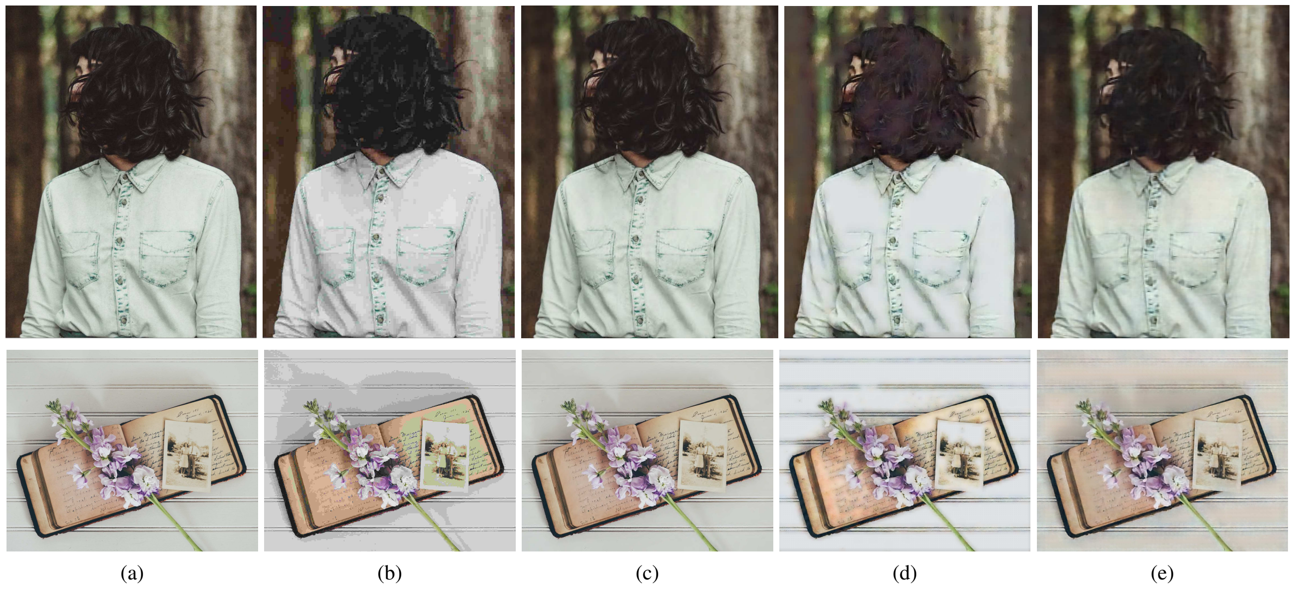
Figure 3: Visual examples from the validation and test image datasets (a) and the decoded images of codecs JPEG (b), JPEG2000 (c), NoWCAE (d) and WCAE (e).

Above, X denotes the input image, $\hat{X}$ is the decoded image, D is the distortion and R is the entropy of the quantized code $\hat{y}$ which has the distribution $P_{\hat{y}}(\hat{y})$. Here, the distribution of $\hat{y}$ is approximated to be Gaussian after the use of multiple GDN nonlinearities and the discrete probability function of $\hat{y}$ is computed as a Gaussian distribution with mean $\mu_{\hat{y}}$ and standard deviation $\sigma_{\hat{y}}$.

After the network is fully trained, the latent representation is quantized by rounding to the nearest integer at test time and entropy coding is performed by the range encoder [9]. The range encoder expects a positive input, therefore the minimum value of the quantized code is subtracted and then passed to the decoder. In addition, the decoder also needs to receive the input image dimensions, as each input channel is padded with zeros to have dimensions that are multiples of 32. Finally, the cumulative distribution function of the quantized code is also passed to the decoder. The total size of the encoded bitstream is then equal to the sum of these additional parameters sent to the decoder and the output of the range encoder. In order to reach a total rate less than 0.15 bits per pixel (bpp) on the CLIC2019 test dataset, the parameter λ needs to be adjusted during training.

3. Results

For training the network, the mobile and professional training datasets of CLIC2019 were used and each image was divided into 256×256 non-overlapping patches. A total of 16750 distinct patches were used during training. With a selection of $\lambda = 0.0025$, the total rate of CLIC2019 validation set was fixed at 4514814 bytes, which is equal to 0.148bpp on average. The network was trained iteratively using back propagation [14, 15] and the Adam [16] optimizer with a batch size of 8 and learning rate of 10^{-4} for a total of 100 epochs, where the loss had converged to a stable value. The model with the lowest validation error was selected for testing. The results of the proposed method WCAE were compared to three different codecs, of which two are JPEG and JPEG 2000. To analyze the impact of performing wavelet decomposition in the proposed method, a network with similar architecture was built and trained without the wavelet decomposition. This network is referred to as No Wavelet CAE (NoWCAE), and the analysis and synthesis blocks have the same architecture as that of the finest scale of WCAE. The results in terms of PSNR(dB) and MSSSIM are given in Tables 1 and 2 on the validation and test sets of CLIC2019, respectively. All results have been averaged on the complete validation and test databases, which contained 102 and 330 different images, respectively. None of the images in the validation and test sets were used during training updates.

	JPEG	JPEG 2000	NoWCAE	WCAE
PSNR(dB)	30.27	33.29	23.82	25.61
MSSSIM	0.8208	0.9404	0.8983	0.8965
Rate (bpp)	0.147	0.149	0.206	0.143

Table 1: PSNR(dB) and MSSSIM on the validation set for codecs JPEG, JPEG 2000, NoWCAE and WCAE.

The objective results indicate that although WCAE was trained using MSE loss, MSSSIM values averaged

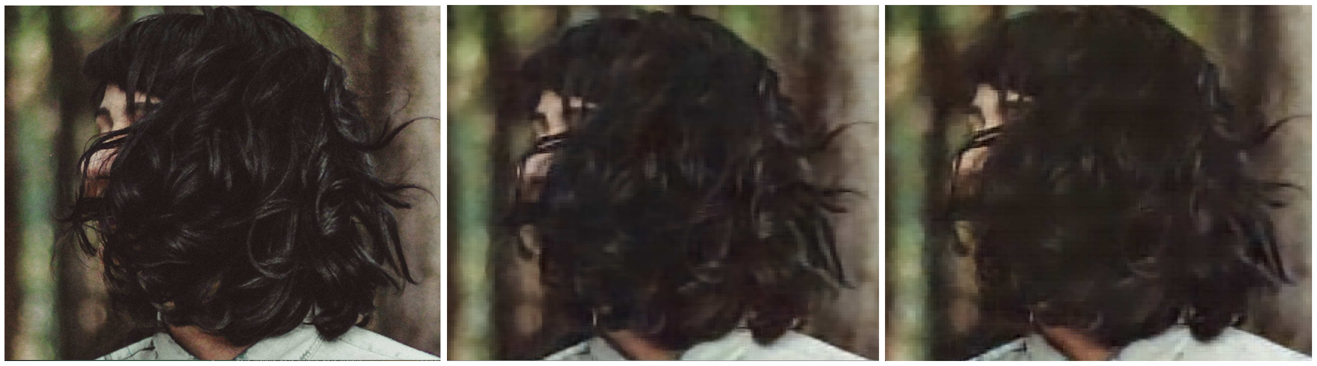
(a) (b) (c)

Figure 4: Section of an example image from the validation set (a), WCAE outputs at target bitrate 0.15bpp using 32 outputs at all wavelet scales with PSNR = 31.12dB and MSSSIM = 0.9264 (b) and 64 outputs instead of 32 at the coarsest scale with PSNR = 29.99dB and MSSSIM = 0.9093 (c).

over both validation and test sets are higher compared to those of JPEG. Visual examples from validation and test databases are presented in Figure 3, where WCAE clearly has higher subjective quality compared to JPEG and NoWCAE. WCAE also outperforms NoWCAE in terms of PSNR, however the MSSSIM of NoWCAE is slightly higher than WCAE in the validation database. It must be taken into consideration here that NoWCAE has a higher actual bitrate on both databases, which explains the higher MSSSIM of NoWCAE on the validation set despite its lower visual quality. Overall, distributions of images compressed with WCAE are more faithful to the distributions of their respective original images. Distinct color changes in Figure 3 (d) with respect to Figure 3 (e) compared to the originals in Figure 3 (a) highlight this advantage of using wavelet transform as a preprocessing step in the proposed network. WCAE has much less low frequency errors compared to JPEG, however the high frequency artifacts are eminent compared to the artifacts in JPEG 2000 results.

	JPEG	JPEG 2000	NoWCAE	WCAE
PSNR(dB)	30.05	32.83	22.25	23.85
MS-SSIM	0.8034	0.9335	0.8743	0.8817
Rate (bpp)	0.149	0.148	0.184	0.138

Table 2: PSNR(dB) and MSSSIM on the test set for codecs JPEG, JPEG 2000, NoWCAE and WCAE.

WCAE introduces apparent high frequency artifacts and is therefore unable to outperform JPEG 2000 at the targeted rate 0.15bpp neither subjectively nor objectively. In the analysis block, the number of outputs from each wavelet scale is 32. It is possible to attenuate the high frequency artifacts by changing the contribution of outputs from coarse to fine scales. When the outputs of the coarsest scale are doubled to 64 at the fifth convolutional layer, the high frequency artifacts become weaker, however the decoded images have increased low frequency noise. This results in lower subjective and objective quality averaged over the validation and test images, depicted on the example in Figure 4. Despite the reduced quality, such changes demonstrate how the use of wavelets can be beneficial in order to adjust the frequency characteristics of the output image. Optimization of the contribution from different scales to the latent representation can adjust the decoded images to have better subjective and objective quality. Optimization of the bit allocation from these scales at the quantization step can increase the quality even further.

4. Conclusion

A convolutional autoencoder involving a 3-scale 2D wavelet decomposition is proposed. The model is trained end-to-end, using mean squared error (MSE) in the loss function. Objective results indicate that WCAE outperforms JPEG and NoWCAE, a similar method that excludes the wavelet decomposition step, in terms of MSSSIM at bitrates lower than 0.15bpp across the full CLIC2019 test set. Visual results of the proposed method also indicate better performance compared to JPEG and NoWCAE. Although WCAE is not able to outperform its transform based counterpart JPEG 2000, we show that the introduction of wavelet decomposition as a preprocessing step improves the performance of learning-based image compression. Future work involves optimization of wavelet scales at the analysis/synthesis and quantization stages of the WCAE.

Acknowledgements

This paper reports a research performed under the framework of project Digital Eye: Deep Learning Video Quality Assessment Technology, funded by The Swiss Commission for Technology and Innovation (CTI) under the grant 27403.1 PFES-ES.

References

[1] Gregory K Wallace. The jpeg still picture compression standard. *IEEE transactions on consumer electronics*, 38(1):xviii–xxxiv, 1992. 1

[2] David Taubman and Michael Marcellin. *JPEG2000 image compression fundamentals, standards and practice: image compression fundamentals, standards and practice*, volume 642. Springer Science & Business Media, 2012. 1

[3] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012. 1

[4] George Toderici, Sean M O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar. Variable rate image compression with recurrent neural networks. *arXiv preprint arXiv:1511.06085*, 2015. 1

[5] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. Full resolution image compression with recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5306–5314, 2017. 1

[6] Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2922–2930. JMLR. org, 2017. 1

[7] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. *arXiv preprint arXiv:1804.02958*, 2018. 1

[8] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. *arXiv preprint arXiv:1611.01704*, 2016. 1, 2

[9] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*, 2017. 1, 3

[10] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Deep convolutional autoencoder-based lossy image compression. In *2018 Picture Coding Symposium (PCS)*, pages 253–257. IEEE, 2018. 1, 2

[11] 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

[12] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. Density modeling of images using a generalized normalization transformation. *arXiv preprint arXiv:1511.06281*, 2015. 1

[13] Claude Elwood Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948. 2

[14] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 3

[15] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012. 3

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3