Rate-Distortion Optimized Learning-Based Image Compression using an Adaptive Hierachical Autoencoder with Conditional Hyperprior

Fabian Brand, Kristian Fischer, André Kaup
Multimedia Communications and Signal Processing
Friedrich-Alexander Universität Erlangen-Nürnberg
Cauerstr. 7, 91058 Erlangen, Germany
fabian.brand@fau.de, kristian.fischer@fau.de, andre.kaup@fau.de

Abstract

Deep-learning-based compressive autoencoders consist of a single non-linear function mapping the image to a latent space which is quantized and transmitted. Afterwards, a second non-linear function transforms the received latent space back to a reconstructed image. This method achieves superior quality than many traditional image coders, which is due to a non-linear generalization of linear transforms used in traditional coders. However, modern image and video coder achieve large coding gains by applying rate-distortion optimization on dynamic block-partitioning. In this paper, we present RDONet, a novel approach to achieve similar effects in compression with full image autoencoders by using different hierarchical levels, which are transmitted adaptively after performing an external rate-distortion optimization. Using our model, we are able to save up to 20% rate over comparable non-hierarchical models while maintaining the same quality.

1. Introduction

Rate-Distortion Optimization (RDO) [18] is an important tool in many image and video codecs. Here, the encoder has the choice between several alternatives during the encoding process and picks the parametrization resulting in the lowest cost. This decision is then transmitted to the decoder such that the decoder is aware of the decision and decodes the bitstream appropriately. Often the best decision is obtained by an exhaustive search, trying out all possible combinations.

One prominent example is the block partitioning in video coders like HEVC [17], VVC [7], VP9 [16], or AV1 [8]. Since all mentioned coders are block-based, the block size plays a crucial role in both the rate and the achieved image quality. In most cases, a large block corresponds to small rate and low quality. These coders therefore mainly use large blocks for low quality settings in areas without many details. Adaptive block-partitioning allows to transmit an image with varying characteristics efficiently by choosing the best block size for each part of the image.

In recent years, much research has been conducted in neural-network-based image compression. The dominant technology is the compressive autoencoder, which extends the autoencoder as proposed by Krizhevsky and Hinton [12] with an entropy bottleneck [3].

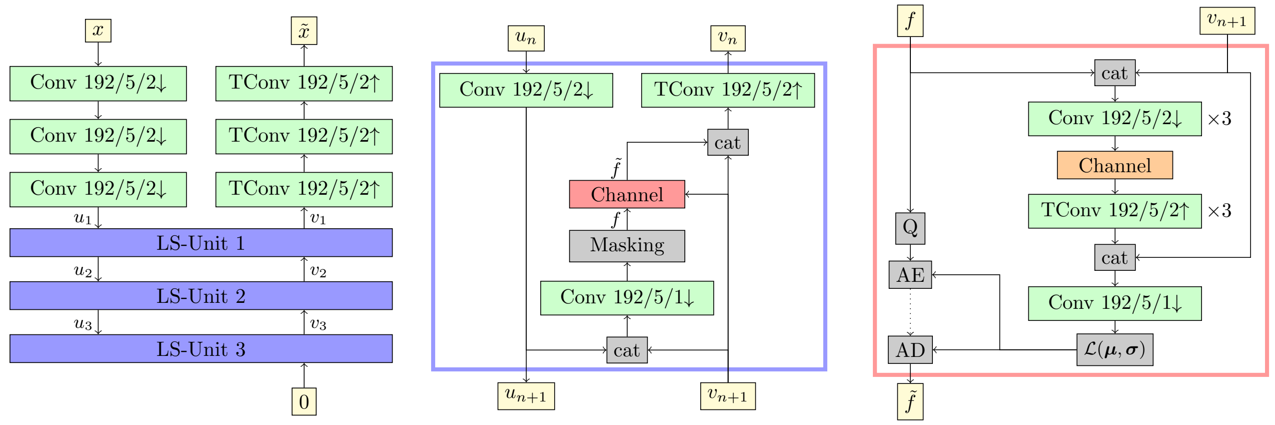
In this paper, we combine the autoencoder and rate distortion optimized hierarchical coding. To the best of our knowledge, we are the first to propose a deep learning-based compression scheme which allows the coder to choose between different hierarchical level in a block-wise manner depending on the image content.

2. Related Work

In 2017, Ballé *et al*. proposed an end-to-end trained image compression method using an autoencoder with an entropy bottle neck in the latent space [3]. That way it was possible to train a network that compresses the image into a low entropy latent space, which can be used to reconstruct the image at the decoder side. This work has served as basis for many subsequent approaches which extended the method in order to achieve better compression of the latent space. Most notable is probably the introduction of a scale hyperprior [4]. This hyperprior was subsequently extended by Minnen *et al*. in [15] to predict not only the variance but also the mean of the latent space distribution, also including spatial correlation derived from a context model.

On the other hand, there has been much research to improve the reconstruction quality of the image by increasing the generative capabilities of the decoder and finding optimal loss functions. In particular, interpreting the decoder as the generative network of a generative adversarial network (GAN) improved the visual quality [19, 14, 9].

There has also been some work to incorporate rate-



(a) RDONet (b) Latent Space Unit (LS-Unit) (c) Channel with conditional hyperprior

Figure 1. Detailed architecture of the proposed RDONet with all novel components. The input image is denoted as x. Conv *c*/*k*/*s*↓ denotes a convolutional layer with c output channels, a $k \times k$ kernel and a subsampling factor of s. TConv denotes a transposed convolution with analogous parameters. Both Conv and TConv include a generalized divisive normalization (GDN) layer except those which are immediately before a Channel or the overall output $\tilde{x}$. cat represents a concatenation along the channels and $\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ denotes a multidimensional Laplace distribution with parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. The channel for the hyperprior in (c) is not given in detail for sake of simplicity. We use a standard autoregressive model with two masked convolutions [10] that generate parameters for a second Laplace distribution.

distortion optimization in end-to-end image compression. In [20], Wang *et al.* proposed to train multiple networks, each specialized on certain characteristics. In the end, the image is divided into large blocks, each of which is encoded with one of the networks. The network ID is transmitted as side information.

3. RDONet

In this section, we propose a novel structure for rate-distortion-optimized image compression using a single network. In the following, let g denote an encoder function and f denote a decoder function. In recent approaches, both g and f are largely fixed during the training process, so there is no way to externally adapt the coder to the content after the training is completed. We start our approach, by recognizing that a deep autoencoder with more spatial subsampling steps is capable of transmitting at lower rates, since there are much less coefficients to be transmitted. However, since not enough information can be passed to the decoder, detailed structures can not be reconstructed properly. We notice that for certain areas a deep autoencoder, which encodes a large portion of the picture in one latent space element, may be desirable.

We therefore propose RDONet, a hierarchical compressive autoencoder. This structure includes a masking layer, which sets certain parts of the latent space to zero, such that they do not have to be transmitted. We show the network structure in Fig 1. The core of our network is the Latent Space Unit (LS-Unit) which handles the transmission. In Fig. 1 (a), we see that we use three consecutive LS-Units. Just looking at the left side of Fig. 1 (b), each LS-Unit computes a new sub-sampled representation u_{n+1} from the input u_n. This works just like the previous layers of the encoder, finding deeper representations for the input signal. The major novelty here is that within each LS-Unit, we can transmit parts of the latent space. Looking at the right side, we see that during decoding, we receive some information v_{n+1} from the lower LS-Unit which is combined with information from the current latent space before obtaining the output v_n using a transposed convolution. The lowest LS-Unit receives a zero tensor as v_{n+1}.

Note, that having only one LS-Unit, which transmits the whole latent space, corresponds to a standard compressive autoencoder similar to the approach of Minnen [15]. The only difference is the additional convolution layer before transmission over the channel and the use of the autoregressive context model in the hyperprior and not in the latent space.

The transmission order is bottom to top, so first LS-Unit 3 transmits parts of the latent space, then LS-Unit 2 and then LS-Unit 1. Different to standard approaches, we do not directly transmit the output of the convolutional layer. Since potentially information was transmitted in deeper layers, we need to take this information into account in order to not transmit the same information twice. A straightforward way would be to subtract the two signals. However, we decided to use a "generalized difference" in form of a concatenation and a convolution, as depicted in the lower part of Fig. 1 (b). It is easily seen that a signal difference is a special case of that structure. However, this way the network has the freedom to take the previously transmitted information into account in any way it sees fit. That way, we obtain our latent space which we need to transmit. Which part of the latent space is transmitted is controlled by the masking layer.

The masking layer can be influenced externally. We define three masks $m_{1,2,3}$, which are associated with the corresponding LS-Unit and share the spatial dimensions with its latent space. When we encode images, the mask is optimized such that the rate-distortion measure is optimal for the image. There are certain constraints to the masks which are most easily understood when we assign a portion of the image to each mask entry. So each entry of the deepest mask m_3 corresponds to a 64×64 block. Every part of the image has to be transmitted exactly once. So if $m_3(0,0) = \text{True}$, i.e. we transmit the latent space at position $(0,0)$ in the lowest layer, the corresponding entries in m_2 and m_1 are always set to False. This serves two purposes: First, this limits the amount of side-information and second, during training it is encouraged that each latent space can stand for itself. Note that this also comes close to the intuition of including adaptive block-partitioning to end-to-end trained image coders. Also note that this constraint implies that we do not need to transmit m_1 at all, since there all remaining positions have to be transmitted.

Finally, we will have a look at the way we transmit each latent space. Here, we use a hyperprior, similar to [15]. The only difference is that we use a conditional autoencoder to transmit the hyperprior. Conditional autoencoders were first used for coding in [5, 6] in the context of intra prediction and more recently in [13] for residual coding. As a condition, which is known to both encoder and decoder, we again use the previously decoded signal. This allows for a very efficient transmission of higher layers, when the previously decoded data contains enough information to predict the current layer. The exact structure of the conditional autoencoder we use is shown in Fig. 1 (c).

4. Experiments and Results

4.1. Training

We implemented the network as shown and explained above using the PyTorch framework. We trained the network using the following loss function:

$$L_{\text{train}} = 0.1 \cdot D_{\text{mse}}(x, \tilde{x}) + D_{\text{ms-ssim}}(x, \tilde{x}) + \lambda_{\text{t}} \cdot r \quad (1)$$

Here, x and $\tilde{x}$ denote the original and reconstructed image respectively and D_{mse} and $D_{\text{ms-ssim}}$ denote the MSE and MS-SSIM loss, respectively. Furthermore, r represents the total rate in bit per pixel (bpp), we need to transmit all latent spaces and hyperpriors. This takes into account that we do not transmit all symbols of each latent space. λ_{t} is the Lagrangian multiplier to set the importance of the rate during training.

Ideally, we need to perform an RDO for each training step to obtain the best masks, however, this is computationally infeasible. We therefore pick random masks obeying the constraints from above during the training process.

We train the model on the CLIC21 intra challenge training set, the TECNICK dataset [2], and the DIV2K dataset [1]. In total these are 1585 images with varying resolution. We train the network using the ADAM optimizer [11] and an initial learning rate of 0.00001. We train the model for 2000 epochs and divide the learning rate by ten after the first 1000 epochs.

4.2. Rate Distortion Optimization

After training the model, we encode the images using rate distortion optimization. During RDO, we measure the performance of the coder using the rate-distortion loss

$$L_{\text{RDO}} = D_{\text{ms-ssim}}(x, \tilde{x}) + \lambda_{\text{e}} \cdot r \quad (2)$$

To that end, we initialize the masks such that m_3 is True for all positions, which automatically sets the remaining masks to False. We then test for each entry in m_3 if the coder performs better when we transmit the corresponding image section using LS-Unit 2 by setting the masks accordingly. In line with the concepts in HEVC we call this a split. We then check for each of the four corresponding positions in m_2 if another split is advantageous (leading to transmission in LS-Unit 1) before moving on to the next element of m_3. Preliminary experiments indicated that this procedure does not converge to the global optimum, since parts of the image are still set to the initial value when the decision for the first mask elements is made. We therefore use a two-pass RDO, by repeating the procedure above initialized with the result of the first pass. Here, the coder is also allowed to reverse splits. This improved the results. In theory, even more passes are possible, however we found that the additional gain is too small to justify the additional effort beyond two passes.

4.3. Experiments

At first, we want to give a visual demonstration of the effect of the RDO. To that end, we show a reconstructed image and the corresponding depth information in Fig. 2. We can clearly observe the expected behavior. In areas with many fine structures, like the camera and the hair, the RDO chooses to use the highest possible level for encoding. On the other hand, in the slightly blurry background, we see that often the lower levels down to the third LS-Unit are used.

Next, we want to compare our model against one without hierarchical latent space unit. For a fair comparison, we use a model with the same parameters having only one LS-Unit which we train in exactly the same way. As explained in the previous section, this model is similar to current compressive autoencoder such as by Minnen *et al.* [15]. We show the rate-distortion curve for one image in Fig. 3. At first, we recognize that $\lambda_{\text{t}} = 0.04$ constitutes an outlier, being

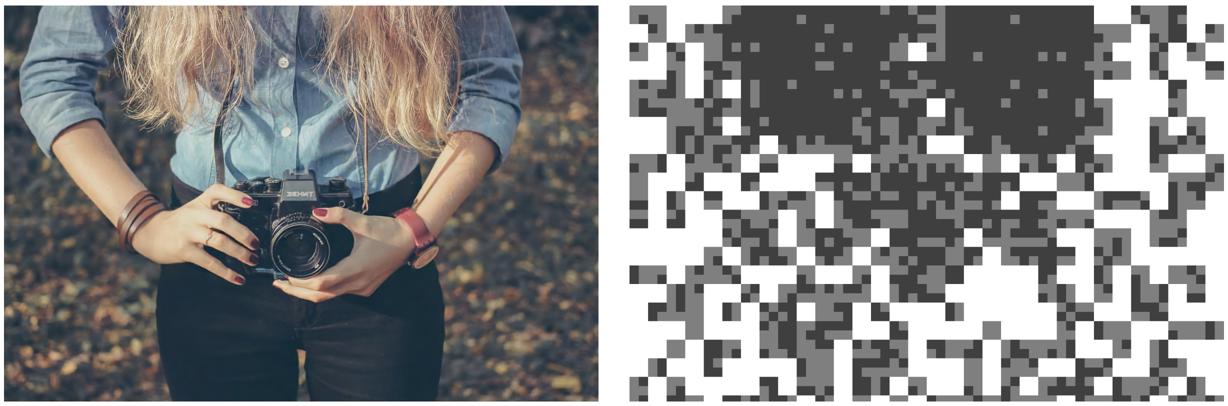


Figure 2. Reconstructed image and coder depth for the image "sergey-zolkin-1045" from the CLIC validation set for $\lambda_t/\lambda_e = 0.02/0.0625$. The right image shows the depth of the latent space transmitted at each position, where white indicates the highest depth.

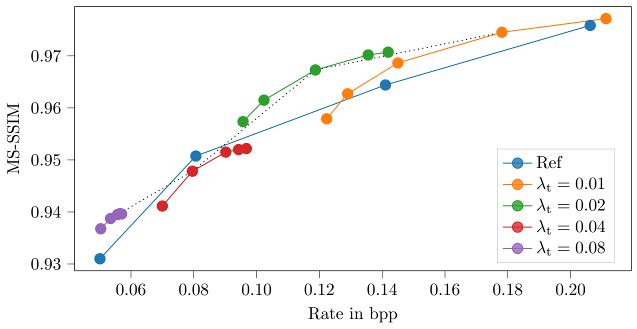


Figure 3. Rate-distortion curve for the image "sergey-zolkin-1045". We show one curve for the non-hierarchical model in which we vary λ_t. For the hierarchical model with 2-pass RDO, we show one curve for each λ_t in which we vary λ_e.The dotted line shows the curve we use to compute the Bjøntegaard Delta Rate.

the only value for which the non-hierarchical model is better. This tendency can be observed throughout the entire dataset. We assume that this is due to unfavorable initialization in the training and that another training might solve the problem. For reasons of fairness and to not overfit on our test set, however, we decided not to repeat the training. Second, we notice that one value of λ_t can produce a wide range of rate points by varying λ_e. Our model is therefore suitable to rate-variable compression, which is not obvious for deep-learning-based image compression. For the plot, we varied λ_e from 0.0625 to 1. For most measurements, the curves are located well above the reference.

Finally, we present the average rate savings of RDONet compared to the reference. For these computations, we used the curve of the reference as shown in Fig. 3 and picked one rate-point for each λ_t to create an RD-curve for RDONet. We chose to pick $\lambda_e = [0.5, 0.5, 0.25, 0.0125]$ for $\lambda_t = [0.08, 0.04, 0.02, 0.01]$, respectively, following the intuition that a larger λ_e fits best to a larger λ_t. As visualization, we included the dotted line in Fig. 3. We first see that 2-pass RDO in fact works better than 1-pass RDO. In some cases, the reference performs better, but only by a maximum of 3.5% in the worst case. On the other hand, in the best case, we can save more than 22% for equal MS-SSIM. On average, we can save 7.4% of bitrate over the entire dataset.

		1-pass	2-pass
Worst Case	jeremy-cai-1174	+7.5%	+3.5%
Best Case	casey-fyfe-999	-18.8%	-22.5%
Average		-4.1%	-7.7%

Table 1. Bjøntegaard Delta Rate for the CLIC validation set. We present the worst, best, and average performance for a 1-pass RDO and a 2-pass RDO. Negative values denote rate savings.

5. Conclusion

In this paper, we presented a novel method to apply concepts known from adaptive block partitioning in traditional image and video coders to end-to-end trained image coders. With our proposed RDONet, we are able to adaptively choose the depth of the autoencoder to fit the image characteristics. We furthermore presented a concrete algorithm how to employ such an RDO using a 2-pass procedure. It is notable that employing RDO at the encoder-side does not influence the decoder complexity.

Since this work was the first of this kind, adding novel components to classical autoencoders, we only trained the network using a MS-SSIM and MSE loss function. However, in principle the concepts are also applicable to any other kind of loss, in particular to adversarial losses as known from GANs, which have the potential to increase the visual quality further. Currently, the models suffers from encoder-decoder drifts if two different GPUs are used for encoding and decoding, which prevented a participation in this year's CLIC challenge. In future research, we aim to make our model more robust against these effects.

In conclusion, these initial results achieved by RDONet over non-hierarchical networks show the great potential of this approach. By optimally switching the depth of the coder in a content adaptive way, we are able to save an average of 7.4% rate with peak savings of over 22%.

References

[1] Eirikur Agustsson and Radu Timofte. NTIRE 2017 challenge on single image super-resolution: Dataset and study. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1122–1131, July 2017. 3

[2] Nicola Asuni and Andrea Giachetti. TESTIMAGES: A large data archive for display and algorithm testing. *Journal of Graphics Tools*, 17(4):113–125, 2013. 3

[3] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. In *Proc. International Conference on Learning Representations (ICLR)*, pages 1 – 27, Toulon, France, Apr 2017. 1

[4] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *Proc. International Conference on Learning Representations (ICLR)*, pages 1–47, 2018. 1

[5] Fabian Brand, Jürgen Seiler, and André Kaup. Intra frame prediction for video coding using a conditional autoencoder approach. In *Proc. Picture Coding Symposium (PCS)*, Nov 2019. 3

[6] Fabian Brand, Jurgen Seiler, and Andre Kaup. Intra-frame coding using a conditional autoencoder. *IEEE Journal of Selected Topics in Signal Processing*, pages 1–12, 2020. 3

[7] Benjamin Bross, Jianle Chen, Shan Liu, and Ye-Kui Wang. Versatile video coding (draft 10), JVET-S2001. *19th Meeting of the Joint Video Exploration Team (JVET)*, pages 1–292, Jan 2020. 1

[8] Jingning Han, Bohan Li, Debargha Mukherjee, Ching-Han Chiang, Adrian Grange, Cheng Chen, Hui Su, Sarah Parker, Sai Deng, Urvang Joshi, Yue Chen, Yunqing Wang, Paul Wilkins, Yaowu Xu, and James Bankoski. A technical overview of AV1. *Proceedings of the IEEE*, pages 1–28, 2021. 1

[9] Chao Huang, Haojie Liu, Tong Chen, Qiu Shen, and Zhan Ma. Extreme image coding via multiscale autoencoders with generative adversarial optimization. In *2019 IEEE Visual Communications and Image Processing (VCIP)*. IEEE, dec 2019. 1

[10] Ajay Jain, Pieter Abbeel, and Deepak Pathak. Locally masked convolution for autoregressive models. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2020. 2

[11] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. International Conference on Learning Representations (ICLR)*, May 2015. 3

[12] Alex Krizhevsky and Geoffrey E Hinton. Using very deep autoencoders for content-based image retrieval. In *Proc. European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, pages 489–494, 2011. 1

[13] Théo Ladune, Pierrick Philippe, Wassim Hamidouche, Lu Zhang, and Oliviér Deforges. Modenet: Mode selection network for learned video coding. In *Proc. IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, Sept. 2020. 3

[14] Fabian Mentzer, George D Toderici, Michael Tschannen, and Eirikur Agustsson. High-fidelity generative image compression. *Advances in Neural Information Processing Systems*, 33, 2020. 1

[15] David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 1, 2, 3

[16] Debargha Mukherjee, Jim Bankoski, Adrian Grange, Jingning Han, John Koleszar, Paul Wilkins, Yaowu Xu, and Ronald Bultje. The latest open-source video codec VP9 - an overview and preliminary results. In *2013 Picture Coding Symposium (PCS)*. IEEE, dec 2013. 1

[17] 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, Dec 2012. 1

[18] G. J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74–90, Nov 1998. 1

[19] Vijay Veerabadran, Reza Pourreza, Amirhossein Habibian, and Taco S. Cohen. Adversarial distortion for learned video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 1

[20] Yefei Wang, Dong Liu, Siwei Ma, Feng Wu, and Wen Gao. Ensemble learning-based rate-distortion optimization for end-to-end image compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(3):1193–1207, mar 2021. 1