END-TO-END LEARNED IMAGE COMPRESSION WITH CONDITIONAL LATENT SPACE MODELING FOR ENTROPY CODING

Aziz Berkay Yeşilyurt *Fatih Kamışlı*

Department of Electrical and Electronics Engineering
Middle East Technical University, Ankara, Turkey

ABSTRACT

The use of neural networks in image compression enables transforms and probability models for entropy coding which can process images based on much more complex models than the simple Gauss-Markov models in traditional compression methods. All at the expense of higher computational complexity. In the neural-network based image compression literature, various methods to model the dependencies in the transform domain/latent space are proposed. This work uses an alternative method to exploit the dependencies of the latent representation. The joint density of the latent representation is modeled as a product of conditional densities, which are learned using neural networks. However, each latent variable is not conditioned on all previous latent variables as in the chain rule of factoring joint distributions, but only on a few previous variables, in particular the left, upper and upper-left spatial neighbor variables based on a Markov property assumption for a simpler model and algorthm. The compression performance is comparable with the state- of-the-art compression models, while the conditional densities require a much simpler network and training time due to their simplicity and less number of parameters then its counterparts.

Index Terms— image compression, transform coding, deep learning, conditional modeling

1. INTRODUCTION

Image compression or coding aims to represent an image signal using as few bits as possible for a given reconstruction quality. In lossy compression, the observer is not interested in the exact representation of the original image signal, so the signal can be transmitted with tolerable distortion such that it is possible achieve much lower bit-rates. The introduced distortion allows significant reduction in the bit rate, i.e. very high compression ratios.

In traditional image compression algorithms, linear transforms with fast computation algorithms (such as DCT [1]) are used, which are optimal for simple models of the data (e.g. Gaussian-Markov model of the image pixels [2, 3]). Recent traditional image/video compression algorithms [4, 5] use multiple-mode spatial prediction algorithms, multiple transforms, and conditional probability models for entropy coding to model images better and improve compression performance.

Each of the processing steps require iterative tweaking to adjust the processing depending on the other steps to obtain decent performance. This type of approach requires many iterations and an immense amount of engineering work. The advantage of using a system based on neural network (NN) is that the processing steps such as transformation and probability models for entropy coding can be performed with NN whose parameters are learned jointly from the dataset of images, hence no iterative tweaking is required and the processing parameters are optimized from real images instead of simpler models.

There are some problems regarding the differentiability for the image compression using neural nets. First of all, quantization of transform coefficients is necessary for the lossy compression, but the quantization operation is not differentiable, which is explicitly required by the popular nonlinear optimization algorithms using gradients. The second problem is the quantization of probability densities that is required by entropy coders such as arithmetic coder. There are different methods applied as a solution to these problems.

To overcome the derivative problem of the quantization operation, Ballé et al. [6] uses additive uniform noise as a proxy for quantization during training. The latent variables at the output of the analysis transform are mixed additively with the uniform noise to imitate the effect of quantization. During the training, continuous univariate density model for each channel of latent space is learned, assuming the independence of adjacent pixels in the latent space. The density model and the latent variables are only quantized during the testing phase. This approach in [6] is the basis for the proposed method.

Theis et al. [7] proposes an alternative way to deal with the quantization. They apply the quantization in the forward pass, but the derivative of the quantizer is set to unity for the backward pass. This way the decoder always trains on quantized variables and the quantization operation only affects the training of the analysis transform (encoder of autoencoder).

Mentzer et al. [8] proposes an alternative quantization method such that the quantization centers are learned during

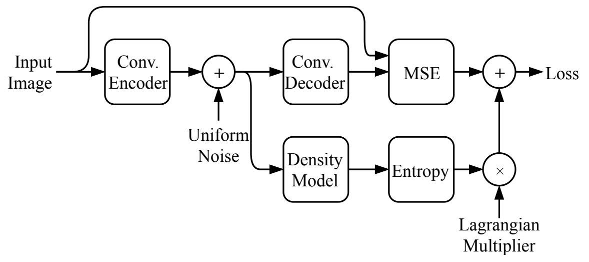

Fig. 1. Graph used in the training of the proposed network.

training. The quantization is applied as it is during the forward pass and the relaxation of the quantization is performed by a smooth distance kernel to each quantization center in the backward pass.

Toderici et al. [9] trains an Recurrent Neural Network (RNN) that outputs a binary sequence of bits each time the network is run. The generated bit-stream is then modeled using an architecture similar to PixelRNN [10] for conditional coding.

The contribution of this work is to train a conditional density model for each channel of the latent representation along with a convolutional autoencoder such that likelihoods of the latent variables are represented with a conditional density, conditioned on just a few causal neighbor variables in the same latent channel. Hence, the joint density of a latent channel is modeled with a product of conditional densities, which gives better compression performance than a product of marginal densities [6] but has similar complexity, and gives similar compression performance to using a hyper-prior based model [11] but has lower complexity.

The rest of this paper is organized as follows. The problem formulation is introduced in Section 2. Then, the model used for the density estimation of latent variables in [6] is explained in detail in Section 3. Our approach based on [12] is presented in Section 4. The performance of our approach is compared in Section 5 and Section 6 concludes the paper.

2. PROBLEM DEFINITION

Training of NN for image compression is formulated in Eq. 3 as minimizing the weighted sum of rate and distortion costs [13], where rate is measured with the entropy, $H(P_{\hat{Y}})$, of the quantized latent variables/transform coefficients $\hat{\mathbf{y}}$, and the distortion is measured with the mean squared error, $\mathbb{E}\|\mathbf{x} - \hat{\mathbf{x}}\|$, between the original image, $\mathbf{x}$ and the compressed/reconstructed image $\hat{\mathbf{x}}$.

$$\min_{\phi,\theta} J = H(P_{\hat{Y}}) + \lambda\mathbb{E}\|\mathbf{x} - \hat{\mathbf{x}}\| \tag{1}$$

$$= H(P_{\hat{Y}}) + \lambda\mathbb{E}\|\mathbf{x} - g(f(\mathbf{x},\phi),\theta)\| \tag{2}$$

$$= \mathbb{E}[-\log_2 p_{\hat{Y}}(\hat{y})] + \lambda\mathbb{E}\|\mathbf{x} - g(f(\mathbf{x},\phi),\theta)\| \tag{3}$$

The neural network for transform/analysis is $f(\mathbf{x},\phi)$, shown as Conv. Encoder in Fig. 1 and 2, and inverse transform/synthesis is $g(\hat{\mathbf{y}},\theta)$, shown as Conv. Decoder in Fig. 1 and 2, where ϕ and θ represent the parameters of these NN.

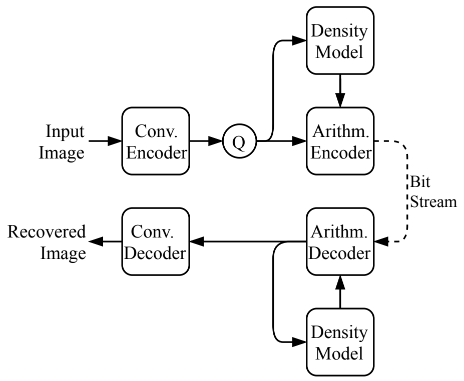

Fig. 2. Graph used in the testing of the proposed network.

This work focuses on the density modeling of the latent variables to achieve higher coding efficiency. While an independent density model is trained in [6] with the assumption that the latent variables are independent, it is apparent that the latent variables are still correlated after the training [11]. Motivated by this observation, a conditional density model, that makes use of the neighboring latent variables, is trained for the joint probability density modeling of latent variables within a latent channel using an architecture similar to Monotonic Neural Density Estimator (MONDE) proposed in [12].

Different from the publications that performs context modeling in the latent space [11, 8], the proposed network directly learns the distributions, rather than making nonlinear predictions of the values of the coefficients. In addition to that, learning the distributions directly requires a very simple network structure, so that the network complexity is reduced compared to the architectures with similar rate-distortion performance.

3. INDEPENDENT DENSITY MODELING

If the latent variables in a channel of the latent representation (i.e. transform coefficients of a NN based non-linear transform) are assumed to be independent and their distribution does not change with the position of that variable inside the channel, then the entire channel's joint probability distribution can be represented using a single univariate probability distribution. In addition, arithmetic coding of the entire channel can be performed in a simple manner using this single univariate distribution. This approach was used in [6]. However, our empirical observations indicate that latent variables in a channel are not independent. Indeed, the authors of [11] improve their work by accounting for the dependencies using a hyperprior NN that transforms the latent variables to another set of variables, conditioned on which the latent variables become independent.

A univariate probability density function (PDF) $p_X(x) : \mathbb{R} \rightarrow \mathbb{R}^+$ and its cumulative distribution function (CDF) $F_X(x) : \mathbb{R} \rightarrow [0,1]$ should satisfy the following constraints:

$$F_X(-\infty) = 0 \tag{4}$$

$$F_X(\infty) = 1 \tag{5}$$

$$\frac{\partial F_X(x)}{\partial x} = p_X(x) \geq 0 \tag{6}$$

These conditions imply that the CDF should be a monotonically increasing function, starting from 0 raising up to 1. If we want to express the CDF by series of K functions such as:

$$F_X(x) = f_K \circ f_{K-1} \circ ... \circ f_1 \tag{7}$$

then the PDF can be expressed using the chain rule of calculus as the derivative of the functions according to Eq. 6:

$$p_X(x) = \frac{\partial F_X(x)}{\partial x} \tag{8}$$

$$= \frac{\partial}{\partial x}(f_K \circ f_{K-1} \dots f_1) \tag{9}$$

$$= f_K^{'} \cdot f_{K-1}^{'} \dots f_1^{'} \tag{10}$$

where f_k are vector functions such that $f_k : \mathbb{R}^{N_k} \rightarrow \mathbb{R}^{N_{k+1}}$ and $\mathbb{R}^{N_k}$ is the domain of f_k and the range of f_{k-1}. Then, $f_k^{'} = \frac{\partial f_k}{\partial x}$ are Jacobian matrices with entries being derivatives of function outputs with respect to inputs.

Hence the PDF, $p_X(x)$, can be expressed as series of matrix multiplications, which suggests the use of fully connected neural networks. To ensure that $p_X(x)$ is univariate, input and output dimensions of $p_X(x)$ are $N_0 = N_K = 1$.

According to the Eq. 6, $F_X(x)$ should be monotonically increasing since $p_X(x)$ is nonnegative. Although $F_X(x)$ can be represented by NN with several layers, the constraints in Eq. 4, 5 and 6 should be imposed such that the resulting function is a proper probability distribution function. Although imposing the constraints in Eq. 4 and 5 can be as simple as applying the sigmoid activation function at the output of the last layer f_K, the procedure to satisfy Eq. 6 is not straightforward.

To assure that the derivative of $F_X(x)$ is always nonnegative, the analysis of derivatives should be conducted. Let f_k be fully connected layers,

$$f_k(\mathbf{x}_k) = g_k(\mathbf{W}_k\mathbf{x}_k + \mathbf{b}_k) \tag{11}$$

$$f_K(\mathbf{x}_K) = \sigma(\mathbf{W}_K\mathbf{x}_K + \mathbf{b}_K) \tag{12}$$

where $k \in \{1 \dots K-1\}$ are the intermediate layers and K denotes the output layer, σ is the sigmoid nonlinearity, used to constrain the output in the range $[0,1]$, and g_k is a pointwise nonlinearity. The derivatives of the layers are

$$f_k'(\mathbf{x}_k) = \mathbf{W}_k \operatorname{diag} g_k'(\mathbf{W}_k\mathbf{x}_k + \mathbf{b}_k) \tag{13}$$

$$f_K'(\mathbf{x}_K) = \mathbf{W}_K \, \sigma'(\mathbf{W}_K\mathbf{x}_K + \mathbf{b}_K) \tag{14}$$

To ensure that $p_X(x) \geq 0$, all the elements of $f_k' \in \{1 \dots K\}$, i.e. all elements of the Jacobian matrices given in Eq. 10 must be nonnegative. Starting with Eq. 13, it is assumed that the nonlinearity g_k is a pointwise nonlinearity, hence its Jacobian is a diagonal matrix, since $\frac{\partial g_{k_i}(\mathbf{x})}{\partial x_j} = 0, \forall i \neq j$. The nonlinearity g_k is chosen as

$$g_k(\mathbf{x}_k) = \mathbf{x}_k + \mathbf{a}_k \odot \tanh(\mathbf{x}_k) \tag{15}$$

where $\mathbf{a}_k$ is a parameter vector and $\odot$ is the element-wise multiplication. This particular choice for the nonlinearity enables the function to model troughs as easy as the peaks of $p_X(x)$ as indicated by Balle et al. [11].

The derivative of the nonlinearity given in Eq. 15 is

$$g_k'(\mathbf{x}_k) = 1 + \mathbf{a}_k \odot \tanh'(\mathbf{x}_k) \tag{16}$$

To ensure that Eq. 16 is nonnegative, all of $\mathbf{a}$'s elements should be greater than -1, since the derivative of $\tanh : \mathbb{R} \rightarrow [-1,1]$ is already nonnegative:

$$\tanh'(x) = 1 - \tanh^2(x) \tag{17}$$

To assure nonnegativity of g_k', $\mathbf{a}$ could be reparametrized to yield values greater than -1,

$$\mathbf{a}_k = \tanh(\hat{\mathbf{a}}_\mathbf{k}) \tag{18}$$

For Eq. 13, the only thing that remains is to assure that $\mathbf{W}_k$ is a nonnegative matrix. For this purpose, a similar reparametrization is applied here such that

$$\mathbf{W}_k = \sigma^+(\hat{\mathbf{W}}_k) \tag{19}$$

where $\sigma^+ : \mathbb{R} \rightarrow \mathbb{R}^+$ is the softplus function. Overall, all elements of $\mathbf{W}_k$ and the diagonal matrix in Eq. 13 are positive. To ensure positiveness of the Jacobian matrix in Eq. 14, a similar parametrization for $\mathbf{W}_K$ can be used and note the derivative of a sigmoid is always positive.

4. CONDITIONAL DENSITY MODELING

This work proposes an alternative way to model the dependencies of latent variables so that the joint density of the latent variables is better modeled. The joint density of the latent representation is modeled as a product of conditional densities. However, only a few neighbor latent variables are taken into account for the conditioning based on Markov property assumption. Hence, each latent variable is only conditioned on the adjacent neighbor latent variables, namely upper, left and upper-left variables. The conditional densities are no longer univariate functions due to the conditions and are learned with NN.

The validity of *conditional* CDF generated by a neural network are analyzed in [12] and following conditions should be satisfied:

$$\lim_{x \to -\infty} F_X(x|\mathbf{y}) = 0 \tag{20}$$

$$\lim_{x \to \infty} F_X(x|\mathbf{y}) = 1 \tag{21}$$

$$\frac{\partial F_X(x|\mathbf{y})}{\partial x} = p_X(x|\mathbf{y}) \geq 0 \tag{22}$$

Eq. 20 and 21 is satisfied using sigmoid nonlinearity at the output layer of the neural network. Eq. 22 can be achieved using a similar parametrization as discussed in Sec. 3. Let $F_X(x|y)$ be expressed with a K layer NN and let f_j denote the j-th layer of it:

$$F_X(x|\mathbf{y}) = f_K \circ f_{K-1} \circ \cdots \circ f_1(x|\mathbf{y}) \tag{23}$$

More generally, the conditional variables y can to be transformed using extra layers, to model the condition space better, before the first layer $f_1(x|y)$:

$$F_X(x|\mathbf{y}) = f_K \circ f_{K-1} \circ ... \circ f_1(x|H(\mathbf{y})) \tag{24}$$

$$H(\mathbf{y}) = h_L \circ h_{L-1} \circ \cdots \circ h_1(\mathbf{y}) \tag{25}$$

where $H()$ denotes the composition of L layers to transform only $\mathbf{y}$.

The analysis to achieve $\frac{\partial F_X(x|\mathbf{y})}{\partial x} \geq 0$ does not change for the layers following x. Every layer following x should be constrained such that the weight matrix should be non-negative, $\mathbf{W}_j \geq 0$, which can be achieved using a similar procedure discussed in Sec. 3. The weights forming the conditional terms, namely the weights of h_j, are not constrained to be positive since there are no constraints for the conditional terms [12]. During the experiments, the hyperparameters of the conditional model are set to K = 3 and L = 2.

5. EXPERIMENTAL RESULTS

Fig. 1 shows the overview of the training graph. An input image is fed to the convolutional encoder for the analysis transform to obtain the latent space coefficients. The output of the transform is subject to additive uniform noise of unit range and zero mean during training. The distorted latent variables are fed to convolutional decoder for the reconstruction of the input image. Mean squared error (MSE) is used for the evaluation of distortion by comparing the reconstructed image from the distorted latent variables and the input image. In the meantime, the distorted latent variables are used in the density model for the conditional distribution estimation for each channel in the latent space. Each channel in the latent space is fed to a separate fully connected neural network to model the dependencies in that channel. The conditional density model is conditioned on three neighboring coefficients. In this work, only three neighbors (top, left, top left) are used for conditional density modeling. The coefficients that are on the boundaries of the channels and do not have the necessary neighbors, are assumed to have neighbors with zero value, such as the first latent variable in a channel which is missing all three neighbors. The likelihoods from the output of the density model is used to estimate the entropy. Finally, the rate and distortion losses are combined into a single loss using a scalar multiplier. All the compared networks [1] [6, 11] are trained on DIV2K dataset [15].

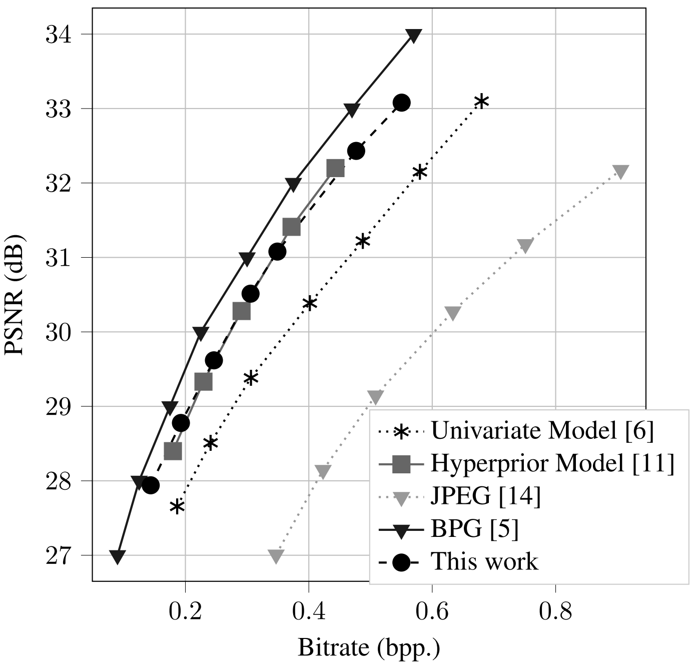

Fig. 3. Comparison of the rate distortion performance of the proposed network with several compression algorithms.

Fig. 3 shows the rate-distortion curves for several compression system as well as the proposed network in this work. The worst compression is performed by JPEG [14]. The next is the neural network with univariate density model [6] where the indepence assumption hinders its performance. Followed by the hyperprior model [11] and the conditional model which are built upon the previous network, perform on par. The hyperprior approach models the density for each latent variable as a Gaussian density and estimates its mean and scale parameter. Our density model assumes Markov property for the latent representation and models the densities conditionally using adjacent latent variables. Hence, the conditional modeling improves the univariate model [6] by 30% and have almost the same performance with the hyperprior model [11] as shown in Fig. 3. Finally, the best performing algorithm is the BPG, which is a heavily engineered state-of-the-art compression algorithm used mainly for the coding of intra-frames in HEVC [5]. The encoding and decoding times of the compared algorithms are given in Table 1 measured on a system with Ubuntu 18.04 with 16GB of RAM and Intel i7-6700HQ

[1]Implementations are available on github.com/tensorflow/compression

CPU. More detailed results and further insights can be found in [16].

	Enc. Time (sec.)	Dec. Time (sec.)
JPEG [14]	0.01	0.01
BPG [5]	0.38	0.17
Univ. Mdl. [6]	7.80	7.60
Hypr. Mdl. [11]	9.10	8.50
This Work	8.10	7.90

Table 1. Encoding and decoding times of compression algorithms.

6. CONCLUSION

An end-to-end learning framework is presented in [6] jointly learns the transform and entropy coding operations using neural networks, while minimizing the rate-distortion cost estimated from a dataset of images by coding the latent variables with an independent density model.

In this work, the joint density of the latent representation is modeled as a product of conditional densities based on Markov property. In other words, each latent variable is conditioned only on a few previous variables. The analysis for conditional density model is conducted similar to the monotonic neural density estimator proposed in [12].

The compression performance of the proposed network is compared to other classical compression algorithms, as well as learning-based compression algorithms on Kodak Image Dataset for different rate-distortion performance. The proposed network improves the coding efficiency of the univariate density model in [6] by 30% and is on par with the hyperprior model in [11].

7. REFERENCES

[1] Nasir Ahmed, T. Natarajan, and Kamisetty R Rao, "Discrete cosine transform," *IEEE transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[2] Fatih Kamisli, "Block-based spatial prediction and transforms based on 2d markov processes for image and video compression," *IEEE Transactions on Image Processing*, vol. 24, no. 4, pp. 1247–1260, 2015.

[3] Jingning Han, Ankur Saxena, Vinay Melkote, and Kenneth Rose, "Jointly optimized spatial prediction and block transform for video and image coding," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1874–1884, 2011.

[4] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.

[5] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, Dec 2012.

[6] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli, "End-to-end optimized image compression," *CoRR*, vol. abs/1611.01704, 2016.

[7] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár, "Lossy image compression with compressive autoencoders," 03 2017.

[8] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool, "Conditional probability models for deep image compression," *CoRR*, vol. abs/1801.04260, 2018.

[9] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell, "Full resolution image compression with recurrent neural networks," *CoRR*, vol. abs/1608.05148, 2016.

[10] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu, "Pixel recurrent neural networks," *CoRR*, vol. abs/1601.06759, 2016.

[11] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston, "Variational image compression with a scale hyperprior," in *International Conference on Learning Representations*, 2018.

[12] Pawel Chilinski and Ricardo Silva, "Neural likelihoods via cumulative distribution functions," 11 2018.

[13] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli, "End-to-end optimization of nonlinear transform codes for perceptual quality," *CoRR*, vol. abs/1607.05006, 2016.

[14] Gregory K Wallace, "The jpeg still picture compression standard," *IEEE transactions on consumer electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[15] Eirikur Agustsson and Radu Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[16] Aziz Berkay Yesilyurt, "End-to-end learned image compression with conditional latent space modelling for entropy coding," M.S. thesis, Middle East Technical University, 2019.