Nonlinear Transform Coding

Johannes Ballé, *Member, IEEE*, Philip A. Chou, *Fellow, IEEE*, David Minnen, Saurabh Singh, Nick Johnston, *Member, IEEE*, Eirikur Agustsson, Sung Jin Hwang, and George Toderici

***Abstract*—We review a class of methods that can be collected under the name nonlinear transform coding (NTC), which over the past few years have become competitive with the best linear transform codecs for images, and have superseded them in terms of rate–distortion performance under established perceptual quality metrics such as MS-SSIM. We assess the empirical rate–distortion performance of NTC with the help of simple example sources, for which the optimal performance of a vector quantizer is easier to estimate than with natural data sources. To this end, we introduce a novel variant of entropy-constrained vector quantization. We provide an analysis of various forms of stochastic optimization techniques for NTC models; review architectures of transforms based on artificial neural networks, as well as learned entropy models; and provide a direct comparison of a number of methods to parameterize the rate–distortion trade-off of nonlinear transforms, introducing a simplified one.**

***Index Terms*—Artificial neural networks, data compression, machine learning, rate-distortion, source coding, transform coding, unsupervised learning.**

I. Introduction

THERE is no end in sight for the world's reliance on multimedia communication. Digital devices have been increasingly permeating our daily lives, and with them comes the need to store, send, and receive images and audio ever more efficiently. Almost universally, transform coding (TC) has been the method of choice for compressing this type of data source.

In his 2001 article for IEEE Signal Processing Magazine [1], Vivek Goyal attributed the success of TC to a divide-and-conquer paradigm: the practical benefit of TC is that it separates the task of decorrelating a source, from coding it. Any source can be optimally compressed in theory using vector quantization (VQ) [2]. However, in general, VQ quickly becomes computationally infeasible for sources of more than a handful dimensions, mainly because the codebook of reproduction vectors, as well as the computational complexity of the search for the best reproduction of the source vector grow exponentially with the number of dimensions. TC simplifies quantization and coding by first mapping the source vector into a latent space via a decorrelating invertible transform, such as the Karhunen–Loève Transform (KLT), and then separately quantizing and coding each of the latent dimensions.

Much of the theory surrounding TC is based on an implicit or explicit assumption that the source is jointly Gaussian, because this assumption allows for closed-form solutions. If the source is Gaussian, all that is needed to make the latent dimensions independent is decorrelation. When speaking of TC, it is almost always assumed that the transforms are linear, even if the source is far from Gaussian. As an example, consider the banana-shaped distribution in Fig. 1: While linear transform coding (LTC) is limited to lattice quantization, nonlinear transform coding (NTC) can more closely adapt to the source, leading to better compression performance.

Until a few years ago, one of the fundamental constraints in designing transform codes was that determining nonlinear transforms with desirable properties, such as improved independence between latent dimensions, is a difficult problem for high-dimensional sources. As a result, not much practical research had been conducted in directly using nonlinear transforms for compression. However, this premise has changed with the recent resurgence of artificial neural networks (ANNs). It is well known that, with the right set of parameters, ANNs can approximate arbitrary functions [3]. It turns out that in combination with stochastic optimization methods, such as stochastic gradient descent (SGD), and massively parallel computational hardware, a nearly universal set of tools for function approximation has emerged. These tools have also been used in the context of data compression [4]–[9]. Even though these methods were developed from scratch, they have rapidly become competitive with modern conventional compression methods such as HEVC [10], which are the culmination of decades of incremental engineering efforts. This demonstrates, as it has in other fields, the flexibility and ease of prototyping that universal function approximation brings over designing methods manually, and the power of developing methods in a data-driven fashion.

This paper reviews some of the recent developments in data-driven lossy compression; in particular, we focus on a class of methods that can be collectively called *nonlinear transform coding* (NTC), providing insights into its capabilities and challenges. We assess the empirical rate–distortion (RD) performance of NTC with the help of simple example sources: the Laplace source and the two-dimensional distribution of Fig. 1. To this end, we introduce a novel variant of entropy-constrained vector quantization (ECVQ) algorithm [11]. Further, we provide insights into various forms of optimization techniques for NTC

Manuscript received July 2, 2020; revised September 29, 2020; accepted October 9, 2020. Date of publication October 28, 2020; date of current version February 22, 2021. The Guest Editor coordinating the review of this manuscript and approving it for publication was Prof. Ahmet Murat Tekalp. *(Corresponding author: Johannes Ballé.)*

The authors are with Google Research, Mountain View, CA 94043 USA (e-mail: jballe@google.com; philchou@google.com; dminnen@google.com; saurabhsingh@google.com; nickj@google.com; eirikur@google.com; sjhwang@google.com; gtoderici@google.com).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/JSTSP.2020.3034501

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

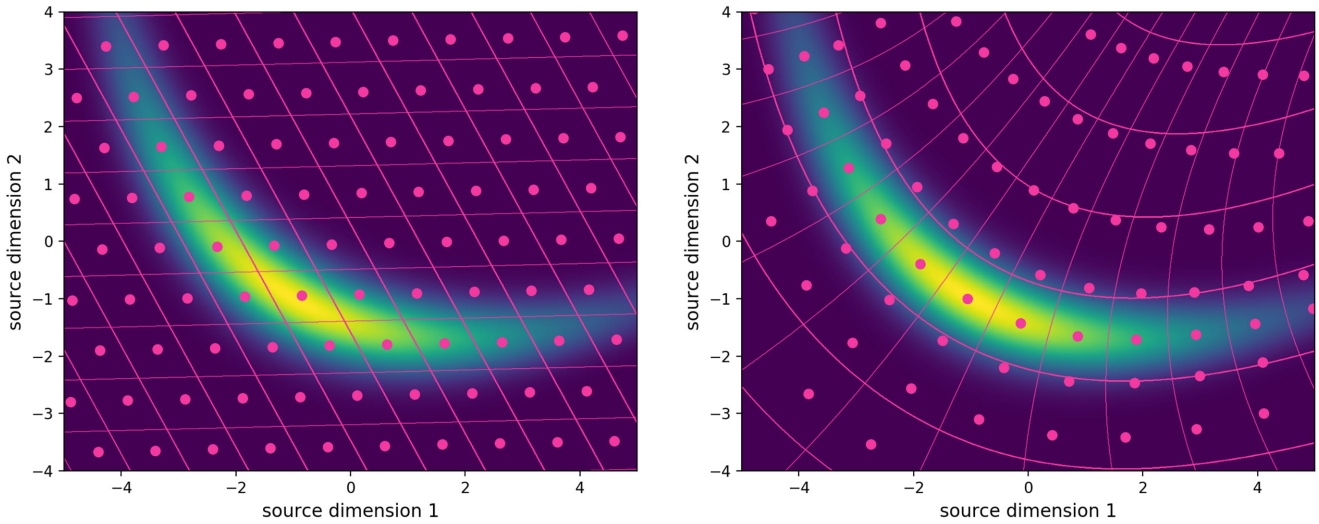

Fig. 1. Linear transform code (left), and nonlinear transform code (right) of a banana-shaped source distribution, both obtained by empirically minimizing the rate–distortion Lagrangian (13). Lines represent quantization bin boundaries, while dots indicate code vectors. While LTC is limited to lattice quantization, NTC can more closely adapt to the source, leading to better compression performance (RD results in Fig. 3; details in Section III).

models and review ANN-based transform architectures, as well as entropy modeling for NTC. A further contribution of this paper is to provide a direct comparison of a number of methods to parameterize the RD trade-off, and to introduce a simplified method.

In the next section, we first review stochastic gradient optimization of the RD Lagrangian, a necessary tool for optimizing ANNs for lossy compression. We introduce *variational* ECVQ, illustrating this type of optimization. VECVQ also serves as a baseline to evaluate NTC in the subsequent section. In that section, we discuss various approaches for approximating the gradient of the RD objective and review ANN architectures. Section IV reviews entropy modeling via learned forward and backward adaptation, and illustrates its performance gains on image compression. Section V compares several ways of parameterizing the transforms to continuously traverse the RD curve with a single set of transforms. The last two sections discuss connections to related work and conclude the paper, respectively.

II. Stochastic Rate–Distortion Optimization

Consider the following lossy compression scenario. Alice is drawing vectors $\boldsymbol{x} \in \mathbb{R}^N$ from some data source, whose probability density function we denote p_{source}. Here, Alice is concerned with compressing each vector into a bit sequence, communicating this sequence to Bob, who then uses the information to reconstruct an approximation to $\boldsymbol{x}$. Each possible vector $\boldsymbol{x}$ is approximated using a codevector $\boldsymbol{c}_k \in C$, where $C = \{\boldsymbol{c}_k \in \mathbb{R}^N \mid 0 \leq k < K\}$ is called the codebook. Once the codevector index $k = e(\boldsymbol{x})$ for a given $\boldsymbol{x}$ is determined using the encoder $e(\cdot)$, Alice subjects it to lossless entropy coding, such as Huffman coding or arithmetic coding, which yields a bit sequence of nominal length $s(k)$. In what follows, we'll assume that the performance of this entropy coding method is optimized to closely approximate the theoretical limit, i.e., that Alice and Bob share an estimate of the marginal probability distribution of k, also called an entropy model, $P(k)$, and that $s(k) \approx -\log P(k)$. To the extent that $P(k)$ approximates $M(k) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{source}}} \delta(k, e(\boldsymbol{x}))$, the true marginal distribution of k (where δ denotes the Kronecker delta function), $s(k)$ is close to optimal, since codes of length $-\log M(k)$ would achieve the lowest possible average rate, the entropy of k. Since Alice and Bob also share knowledge of the codebook, Bob can decode the index k and finally look up the reconstructed vector $\boldsymbol{c}_k$.

To optimize the efficiency of this scheme, Alice and Bob seek to simultaneously minimize the cross entropy of the index under the entropy model (the rate) as well as the distortion between $\boldsymbol{x}$ and the reconstructed vector, quantified by some distortion measure d:

$$L = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{source}}}\big[-\log P(k) + \lambda\, d(\boldsymbol{x}, \boldsymbol{c}_k)\big], \tag{1}$$

with $k = e(\boldsymbol{x})$ as determined by the encoder $e(\cdot)$, choosing a codebook index for each possible source vector. Many authors formulate this as a minimization problem over one of the terms given a hard constraint on the other [12]. In this paper, we consider the Lagrangian relaxation of the distortion-constrained problem, with the Lagrange multiplier λ on the distortion term determining the trade-off between rate and distortion.

The top panel of Fig. 2 illustrates a lossy compression method for a simple, one-dimensional Laplacian source, optimized for squared error distortion (i.e., $d(\boldsymbol{x}, \boldsymbol{c}) = \|\boldsymbol{x} - \boldsymbol{c}\|_2^2$). The source distribution is plotted in blue. The codebook vectors are represented by the horizontal locations of the black stalks, while the height of each stalk is proportional to the likelihood of that code vector under the entropy model P. Dotted lines delineate the quantization bins, i.e., the intervals for which all source values get mapped to a given codebook value (the one within the respective interval). For Laplacian sources, the minimizer of L has been studied by Sullivan [13]. It is characterized by equal-width quantization bins and equidistant code vectors, except for the

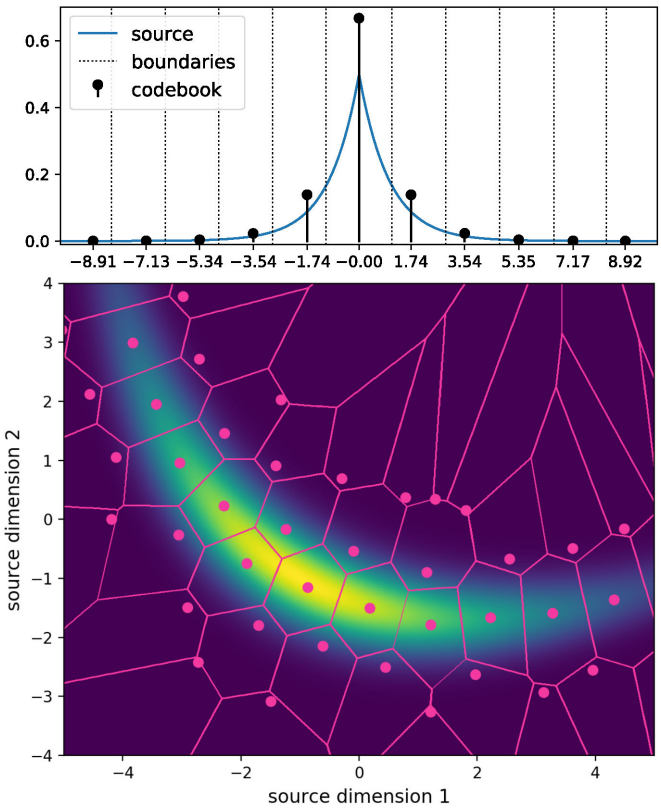

Fig. 2. Top: Near-optimal entropy-constrained scalar quantizer of a standard Laplacian source, found using the VECVQ algorithm (5). Bottom: Entropy-constrained vector quantizer of a banana-shaped source, found using the same algorithm.

center bin (coinciding with the mode of the source distribution); both characteristic features are present in the figure up to small deviations. The bottom panel of the same figure visualizes a vector quantizer for a banana-shaped source distribution. The boundaries between quantization bins are shown as pink lines, while the code vectors are rendered as discs. Note the presence of hexagon-like bins, which are a feature of optimal VQ for squared-error distortions.

A. *Variational Entropy-Constrained Vector Quantization*

To generate both of the results in Fig. 2, we used a novel algorithm for entropy-constrained vector quantization based on directly minimizing (1). To begin, without loss of generality, we parameterize the entropy model as

$$P(k) = \frac{e^{a_k}}{\sum_{j=0}^{K-1} e^{a_j}}. \tag{2}$$

Then, denoting model parameters $\Theta = \{a_k, \boldsymbol{c}_k \mid 0 \le k < K\}$, we define the sample loss

$$\ell_\Theta(k, \boldsymbol{x}) = -\log P(k) + \lambda\, d(\boldsymbol{x}, \boldsymbol{c}_k) \tag{3}$$

and the encoder function

$$e_\Theta(\boldsymbol{x}) = \arg\min_k \ell_\Theta(k, \boldsymbol{x}), \tag{4}$$

where we have made explicit their dependence on the parameters Θ. We express (1) as

$$L_{\text{VQ}} = \mathbb{E}_{\boldsymbol{x}}\, \ell_\Theta(e_\Theta(\boldsymbol{x}), \boldsymbol{x}) = \mathbb{E}_{\boldsymbol{x}} \min_k \ell_\Theta(k, \boldsymbol{x}), \tag{5}$$

which we now wish to minimize over Θ using stochastic gradient descent (SGD). SGD relies on a Monte Carlo approximation of the expectation, and the fact that expectations and derivatives are both linear operators, whose order can be exchanged. Thus

$$\frac{\partial}{\partial \Theta} L_{\text{VQ}} = \mathbb{E}_{\boldsymbol{x}} \frac{\partial}{\partial \Theta} \min_k \ell_\Theta(k, \boldsymbol{x}), \tag{6}$$

which can be approximated by the sample expectation

$$\frac{\partial}{\partial \Theta} L_{\text{VQ}} \approx \frac{1}{B} \sum_{\{\boldsymbol{x}_b \sim p_{\text{source}} \mid 0 \le b < B\}} \frac{\partial \ell_\Theta(k_b, \boldsymbol{x}_b)}{\partial \Theta}, \tag{7}$$

with $k_b = e_\Theta(\boldsymbol{x}_b)$. This is an unbiased estimator of the derivative of L_{VQ} based on averaging the derivatives of ℓ over a batch of B source vector samples.

Minimization of L_{VQ} will fit the entropy model to the marginal distribution of k, $M(k) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{source}}}\, \delta(k, e(\boldsymbol{x}))$, as well as adjust the codebook vectors to minimize distortion. To see this, add and subtract the expected negative log likelihood of k under the marginal to (5):

$$L_{\text{VQ}} = D_{\text{KL}}[M||P] + \mathbb{E}_{\boldsymbol{x}}\big[-\log M(k) + \lambda\, d(\boldsymbol{x}, \boldsymbol{c}_k)\big]. \tag{8}$$

Since the second term is constant wrt. the parameters of P almost everywhere, minimizing L_{VQ} results in fitting P to M by minimizing their Kullback–Leibler (KL) divergence. Similarly, since the first term is constant wrt. the codebook almost everywhere, each $\boldsymbol{c}_k$ is adjusted to minimize the distortion between it and all source vectors getting mapped to k. Note that since the KL divergence is non-negative, L_{VQ} can be interpreted as an upper bound on the second term, which is the rate–distortion objective for the optimal choice of entropy model. This can be likened to variational Bayesian inference, in which a *variational* approximation (P) to an unobserved true distribution (M) is found by minimizing an upper bound on the true objective. We therefore name this method variational entropy-constrained vector quantization (VECVQ).,[1] Note that, since P as defined in (2) can represent arbitrary distributions, the variational approximation here is capable of recovering the true marginal, i.e., the KL divergence can converge to zero.

In the left panel of Fig. 3, we plot the operational rate–distortion function of the optimal entropy-constrained scalar quantizer due to Sullivan [13], as well as the empirical rate–distortion function of the VECVQ algorithm for the same Laplace source. The plot shows that the algorithm recovers the theoretical optimum. Since it is constrained only by the size of the codebook, we can use the algorithm as an empirical lower bound on the rate–distortion objective of more constrained

[1]The VECVQ algorithm inherits from two methods. The first is the ECVQ algorithm of [11], [14], [15] which minimizes (1) using a clustering algorithm instead of gradient descent. The second is the online K-means algorithm of [16], which minimizes the distortion part of (1), $\mathbb{E}_{\boldsymbol{x}}\big[d(\boldsymbol{x}, \boldsymbol{c}_k)\big]$, using gradient descent. Both ECVQ and the online K-means algorithms derive in turn from the generalized Lloyd algorithm [17]–[20].

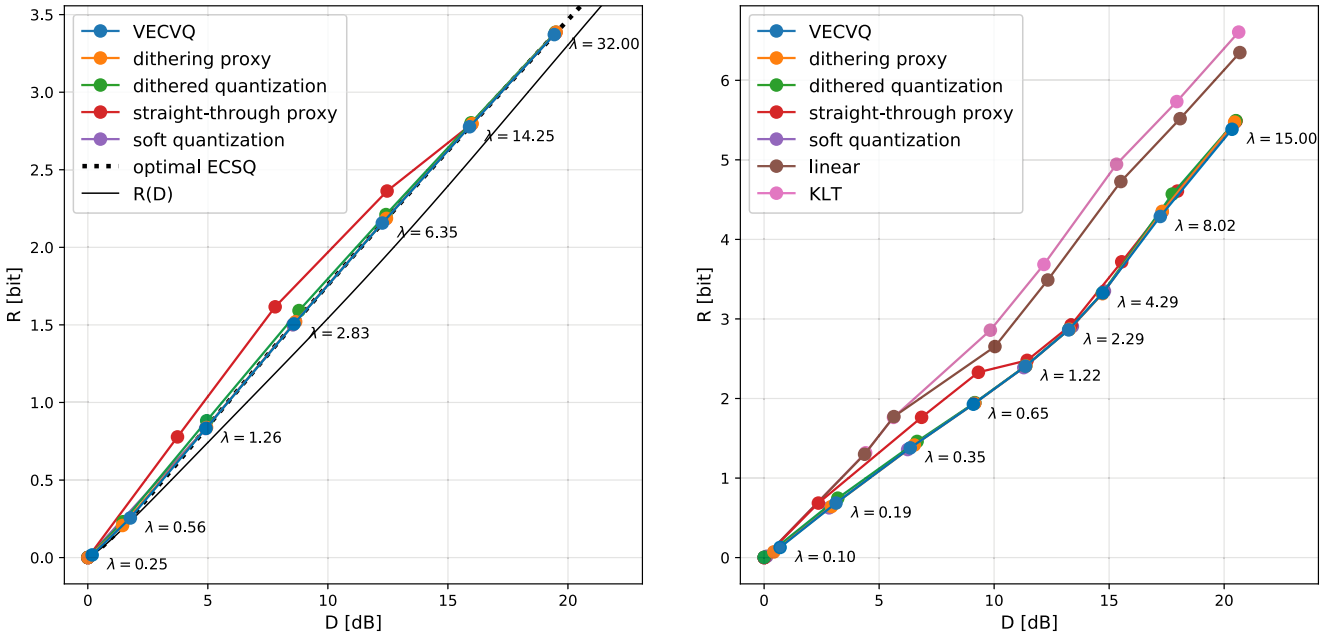

Fig. 3. Left: Rate–distortion performance of different quantizers for standard Laplace source. Both VECVQ and NTC with optimized offset ("dithering proxy") recover the optimal entropy-constrained scalar quantizer established by Sullivan [13]. NTC with randomized offset ("dithered quantization") is slightly suboptimal at lower rates, as predicted by theory. The NTC trained with the straight-through proxy is unstable at low rates. Using the dithering proxy with explicit soft quantization recovers the optimal quantizer as well. $R(D)$ indicates the information-theoretic rate–distortion function, $R(D) = \inf_{p(\hat{x}|x)}\{I(x;\hat{x}) \text{ s.t. } \mathbb{E}[d(x,\hat{x})] \leq D\}$ (achievable only in the limit of large blocksizes, not with a scalar quantizer). Right: Rate–distortion performance of different quantizers for banana source. NTC closely matches the performance of VECVQ; the straight-through variant diverges at low rates. The linear TC trained for the same objective performs significantly worse. Constraining it to the KLT is not necessarily optimal, as pointed out by Goyal [1].

compression methods, such as nonlinear transform coding, even for source distributions for which no theoretical optimum is presently known. As an example, consider the more complex banana distribution in the right panel: the nonlinear transform coders trained with the dithering proxy (to be discussed in the next section) perform ever so slightly worse than VECVQ.

III. Nonlinear Transform Coding

It is easy to modify (1) to accommodate nonlinear transform coding. Rather than explicitly enumerating the codebook vectors, we consider mapping the source vectors into a latent space $\mathbb{R}^M$ and back via a pair of transforms. Quantization and compression takes place in this latent space. Specifically, we define the analysis transform as a parametric function $\boldsymbol{y} = g_a(\boldsymbol{x})$, implemented by a neural network with parameters $\boldsymbol{\phi}$, and the synthesis transform as a function $\tilde{\boldsymbol{x}} = g_s(\tilde{\boldsymbol{y}})$, with parameters $\boldsymbol{\theta}$. We can write the rate–distortion objective as:

$$L_{\text{NTC}} = \mathbb{E}_{\boldsymbol{x}}\big[-\log P\big(\lfloor g_a(\boldsymbol{x})\rceil\big) + \lambda\, d\big(\boldsymbol{x}, g_s(\lfloor g_a(\boldsymbol{x})\rceil)\big)\big], \quad (9)$$

where $\lfloor\cdot\rceil$ denotes uniform scalar quantization (rounding to integers). P is now a probability distribution over a space of integer vectors, which take the role of the codebook index k in (1).

As an example, consider the nonlinear transform code illustrated in Fig. 4. Again, we plot the effective codebook vectors and quantization boundaries on top of the source distribution. However, unlike the example in Fig. 2, this quantization scheme is defined indirectly via the analysis and synthesis transforms, as

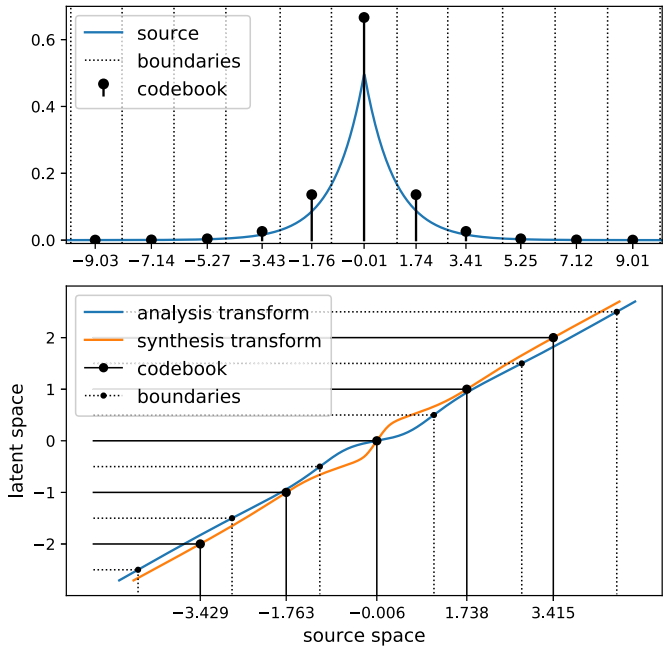

Fig. 4. A near-optimal nonlinear transform code of a standard Laplacian source, obtained by minimizing the dithering rate–distortion proxy (13).

illustrated in the bottom panel of Fig. 4. The analysis transform maps the space of source values to the latent space (blue curve). In this space, uniform quantization is applied, rounding values between half-integers to full integers. These integer values are

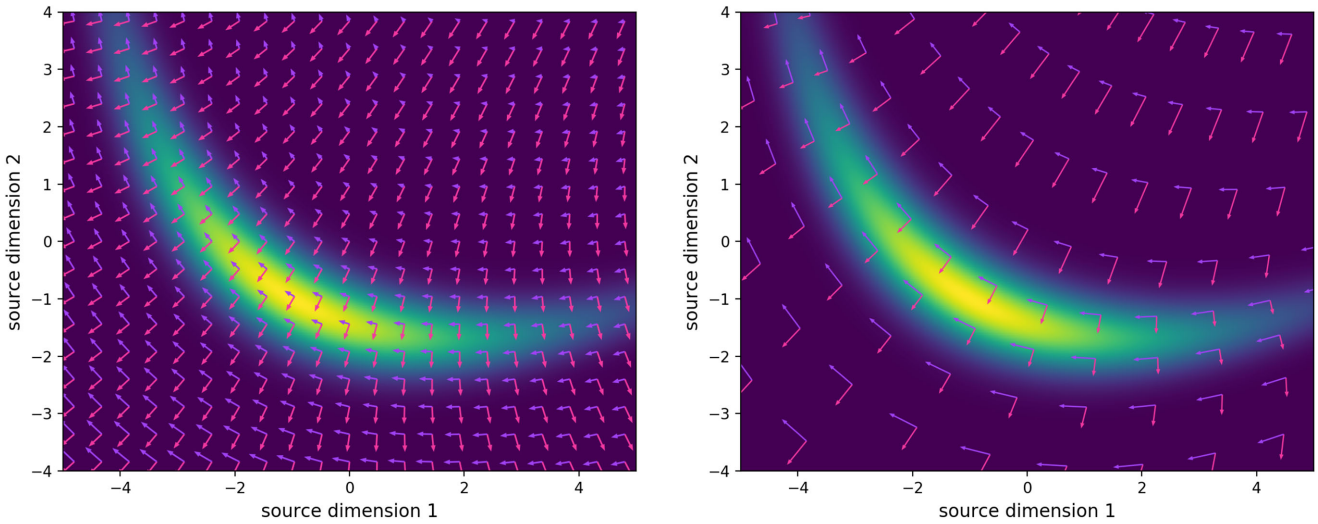

Fig. 5. Location-dependent Jacobian matrices $\nabla g_a(\boldsymbol{x})$ (left; arrows visualize local Jacobian inverse) and $\nabla g_s(\boldsymbol{y})$ (right; arrows visualize local Jacobian) of a model optimized for squared error distortion on the banana source. The transforms form a local orthogonalization of the source density.

then mapped back into the source space using the synthesis transform (orange curve).

There are a few key observations here: First, the analysis transform determines the effective quantization bins. In particular, its intersections with the dotted lines, corresponding to half-integers in the latent space, give rise to quantization bins of varying size in the source space. Second, the synthesis transform determines the effective codebook vectors. Notably, the full behavior of the synthesis transform as determined by the optimization procedure does not matter – only its values at integer locations are relevant. Third, since the transforms are not constrained to be exact inverses of each other, using uniform quantization in the latent space is sufficient to enable codebook vectors to be located anywhere in the corresponding quantization bins (technically, even outside of it). Nonlinear transform coding thus generalizes companding [21], [22], which permits implementing non-uniform quantization using uniform quantizers; with nonlinear transforms, the quantization step size can be fixed to one without loss of generality (parameterizing the model for different rate–distortion trade-offs is discussed in Section V).

Fig. 1 illustrates LTC and NTC of the banana distribution also shown in Fig. 2 (bottom), and evaluated in Fig. 3 (right). Both the linear and nonlinear transform codes are designed to minimize (9) under their respective constraints. Because the linear transform coder is constrained to affine transformations, the method effectively amounts to a lattice quantizer in the source space (left panel). Note that the linear transform is not orthogonal and hence is *not* the KLT, as would be optimal if the source distribution were Gaussian.

The nonlinear transform coder has more flexibility, and can adapt the shapes of its quantization bins to better fit the source distribution (right panel). Note that in both cases, since invertibility of the transforms is not enforced, codebook vectors do not necessarily appear in consistent locations relative to their bins. Their optimal locations, for squared error distortions, are at the conditional mean of their respective cells. Both coders reflect this by shifting the codebook vectors closer to the high-probability regions of the source distribution. This may come at the expense of reconstruction accuracy in low-probability regions – in the nonlinear example, some low-probability codebook vectors even lie outside of their respective bins – because the behavior of the method in these regions often does not contribute much to the overall objective. Another reason for this trade-off may be limitations in the parameterizations of the transforms. This is further examined in Section III-B.

Although the optimized nonlinear analysis and synthesis transforms illustrated in Fig. 1 (right) are globally nonlinear, they are of course differentiable, and hence can be viewed as locally approximately linear. We find that in high-probability regions of the data distribution, they locally resemble KLTs in that they are approximately orthogonalized (Fig. 5). Specifically, at each point $\boldsymbol{x}$, the columns of the inverse Jacobian matrix of the analysis transform appear approximately orthogonal to each other, as shown in the left panel of the figure. Likewise, the columns of the Jacobian matrix of the synthesis transform are approximately orthogonal, as shown in the right panel. In the neighborhood of some point $\boldsymbol{x}_0$, $(\boldsymbol{y} - \boldsymbol{y}_0) \approx \nabla g_a(\boldsymbol{x}_0) \cdot (\boldsymbol{x} - \boldsymbol{x}_0)$ approximates a linear orthogonal analysis transform, and $(\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}_0) \approx \nabla g_s(\boldsymbol{y}_0) \cdot (\boldsymbol{y} - \boldsymbol{y}_0)$ approximates its inverse, where $\boldsymbol{y}_0 = g_a(\boldsymbol{x}_0)$ and $\tilde{\boldsymbol{x}}_0 = g_s(\boldsymbol{y}_0)$.

While more can be said about the local properties of the analysis and synthesis transforms (see supplement), for now, let us consider the transforms as "black boxes" that simply serve to approximate the optimal transforms.

A. *Optimization and Proxy Rate–Distortion Loss*

Note that in the VECVQ loss given in (5), the encoder function is defined by exhaustively minimizing over all possible codes. As such, it can be folded into a minimum over the sample loss ℓ_Θ, which is differentiable with respect to almost all $\boldsymbol{x}$. If we were

to choose another encoder function with trainable parameters of its own, we would not be able to obtain a gradient of the loss function with respect to them that is useful for SGD. The gradient would be zero for almost all $\boldsymbol{x}$, because e is integer valued. This problem also appears in (9) due to the quantizer. Derivatives of the loss with respect to any parameter of the analysis transform are zero almost everywhere. However, when employing dithered quantization (i.e., randomizing the quantization offset) [23], this problem can be avoided [24].

Consider uniformly sampling one random quantization offset per latent dimension $\boldsymbol{o} \in [-\frac{1}{2}, \frac{1}{2})^M$, and formulating the following loss function as an expectation over it:

$$\mathbb{E}_{\boldsymbol{o}}\, L_{\text{NTC},\boldsymbol{o}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{o}}\Big[-\log P\big(\lfloor g_a(\boldsymbol{x}) - \boldsymbol{o}\rceil; \boldsymbol{o}\big) + \lambda\, d\big(\boldsymbol{x}, g_s(\lfloor g_a(\boldsymbol{x}) - \boldsymbol{o}\rceil + \boldsymbol{o})\big)\Big], \quad (10)$$

where $L_{\text{NTC},\boldsymbol{o}}$ is the loss for a given offset $\boldsymbol{o}$, and $P(\cdot; \boldsymbol{o})$ is an entropy model conditioned on $\boldsymbol{o}$. (Note that all else being equal, the marginal distribution of the quantized latents changes with the offset.) This loss function is differentiable with respect to the parameters of g_a. To see this, let us consider both terms separately. For the rate term, we have

$$\begin{aligned}
&-\mathbb{E}_{\boldsymbol{x},\boldsymbol{o}} \log P\big(\lfloor g_a(\boldsymbol{x}) - \boldsymbol{o}\rceil; \boldsymbol{o}\big) \\
&= -\mathbb{E}_{\boldsymbol{x},\boldsymbol{o}} \sum_{\boldsymbol{k}\in\mathbb{Z}^M} \delta\big(\lfloor g_a(\boldsymbol{x}) - \boldsymbol{o}\rceil = \boldsymbol{k}\big) \log P(\boldsymbol{k}; \boldsymbol{o}) \\
&= -\mathbb{E}_{\boldsymbol{x}} \int \cdots \int_{-\frac{1}{2}}^{\frac{1}{2}} \mathrm{d}\boldsymbol{o} \sum_{\boldsymbol{k}\in\mathbb{Z}^M} \delta\big(\|g_a(\boldsymbol{x}) - \boldsymbol{o} - \boldsymbol{k}\|_\infty \le \tfrac{1}{2}\big) \log P(\boldsymbol{k}; \boldsymbol{o}) \\
&= -\mathbb{E}_{\boldsymbol{x}} \int \cdots \int_{-\infty}^{\infty} \mathrm{d}\boldsymbol{v}\, \delta\big(\|g_a(\boldsymbol{x}) - \boldsymbol{v}\|_\infty \le \tfrac{1}{2}\big) \log p(\boldsymbol{v}) \\
&= -\mathbb{E}_{\boldsymbol{x}} \int \cdots \int_{-\frac{1}{2}}^{\frac{1}{2}} \mathrm{d}\boldsymbol{u} \log p\big(g_a(\boldsymbol{x}) + \boldsymbol{u}\big) \\
&= -\mathbb{E}_{\boldsymbol{x},\boldsymbol{u}} \log p\big(g_a(\boldsymbol{x}) + \boldsymbol{u}\big), \qquad (11)
\end{aligned}$$

where δ is the Kronecker delta function, we define $\boldsymbol{v} = \boldsymbol{k} + \boldsymbol{o}$ and $p(\boldsymbol{v}) = P(\lfloor \boldsymbol{v}\rceil; \boldsymbol{v} - \lfloor \boldsymbol{v}\rceil)$, and consider $\boldsymbol{u} \in [-\frac{1}{2}, \frac{1}{2})^M$ uniformly distributed. It is easy to show that p is non-negative and integrates to one, and hence represents a probability density function. We can thus interpret p as a continuous equivalent of an entropy model for the "noisy" latents $g_a(\boldsymbol{x}) + \boldsymbol{u}$. For the distortion term, we have

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{x},\boldsymbol{o}}\, d\big(\boldsymbol{x}, g_s(\lfloor g_a(\boldsymbol{x}) - \boldsymbol{o}\rceil + \boldsymbol{o})\big) \\
&\qquad = \mathbb{E}_{\boldsymbol{x},\boldsymbol{u}}\, d\big(\boldsymbol{x}, g_s(g_a(\boldsymbol{x}) + \boldsymbol{u})\big), \qquad (12)
\end{aligned}$$

since dithered quantization and additive uniform noise have the same marginal distribution (i.e., integrating out $\boldsymbol{o}$ and $\boldsymbol{u}$ is equivalent). For a proof, refer to [23].

Hence, denoting $\tilde{\boldsymbol{y}} = g_a(\boldsymbol{x}) + \boldsymbol{u}$, we can now write

$$\mathbb{E}_{\boldsymbol{o}}\, L_{\text{NTC},\boldsymbol{o}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{u}}\big[-\log p(\tilde{\boldsymbol{y}}) + \lambda\, d\big(\boldsymbol{x}, g_s(\tilde{\boldsymbol{y}})\big)\big], \quad (13)$$

which can be directly minimized via SGD as in (7).

Analogously to (8), we can interpret (13) as a variational upper bound on the true marginal:

$$\mathbb{E}_{\boldsymbol{o}}\, L_{\text{NTC},\boldsymbol{o}} = D_{\text{KL}}[m||p] + \mathbb{E}_{\boldsymbol{x},\boldsymbol{u}}\big[-\log m(\tilde{\boldsymbol{y}}) + \lambda\, d\big(\boldsymbol{x}, g_s(\tilde{\boldsymbol{y}})\big)\big], \quad (14)$$

where $m(\boldsymbol{v}) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{u}}\, \delta(\boldsymbol{v}, g_a(\boldsymbol{x}) + \boldsymbol{u})$ is the marginal distribution of the noisy latents.[2] As such, minimizing (13) results in fitting the continuous entropy model p to the marginal. Note that, unlike in the case of VECVQ, the KL divergence may not converge to zero, as for high-dimensional source distributions such as images, the entropy model will generally not be capable of representing the marginal accurately. Section IV talks about this in more detail.

There is one caveat with using dithered quantization for compression itself: it is not necessarily optimal, since $\mathbb{E}_{\boldsymbol{o}}\, L_{\text{NTC},\boldsymbol{o}} \ge \min_{\boldsymbol{o}} L_{\text{NTC},\boldsymbol{o}}$ (Fig. 3, left panel, verifies this empirically). If we do not wish to use dithered quantization, we can still use (13) as a proxy loss for transform coding with a fixed quantization offset known to both Alice and Bob. A simple algorithm for stochastic optimization of an NTC model is:

1) Minimize (13).
2) Determine which offsets $\boldsymbol{o}$ minimize $L_{\text{NTC},\boldsymbol{o}}$. If the continuous entropy model p is accurate enough, this can be done without re-estimating the discrete entropy models, since $P(\boldsymbol{k}; \boldsymbol{o}) = p(\boldsymbol{k} + \boldsymbol{o})$.

Note that the offsets themselves cannot be determined via gradient descent on L_{NTC}, for the same reason that g_a cannot be determined in this way. We must therefore use some other method. While Ballé *et al.* [5] explicitly perform a grid search over $\boldsymbol{o}$, some follow-up papers have resorted to a simple heuristic: guided by the result that for Laplacian distributions, it is optimal to pick an offset that aligns the mode of the source distribution with a codebook vector, one can simply pick an offset for each latent dimension such that one of the quantization bins is centered on the mode (or, in case that is computationally intractable, the median) of the entropy model p [27], [28]. For an entropy model with fixed mode, such as a zero-mean Gaussian, this implies that the offset may as well be fixed a priori.

A suboptimal choice of offset for an NTC encoding a Laplace source is illustrated in Fig. 6, along with a plot of $L_{\text{NTC},o}$ as a function of o for the same source. Note that optimizing the dithering proxy loss leads to the transforms becoming increasingly curved around the central quantization bin, to accommodate arbitrary choices of o. Because the analysis transform becomes increasingly flat around the center, and the synthesis transform increasingly steep, the effective code vector and quantization bin around the mode of the distribution is skewed towards the near-optimal quantizer illustrated in Fig. 4. It could be argued that, to minimize the loss function, this should happen in all bins. However, we haven't observed this empirically, presumably due to ANNs naturally favoring smoother functions, and the other bins not contributing enough to the value of the loss function.

[2]This dithering objective of rate–distortion optimized nonlinear transform coding is equivalent to β-variational autoencoders [5], [25], [26] up to choice of parameterized distributions.

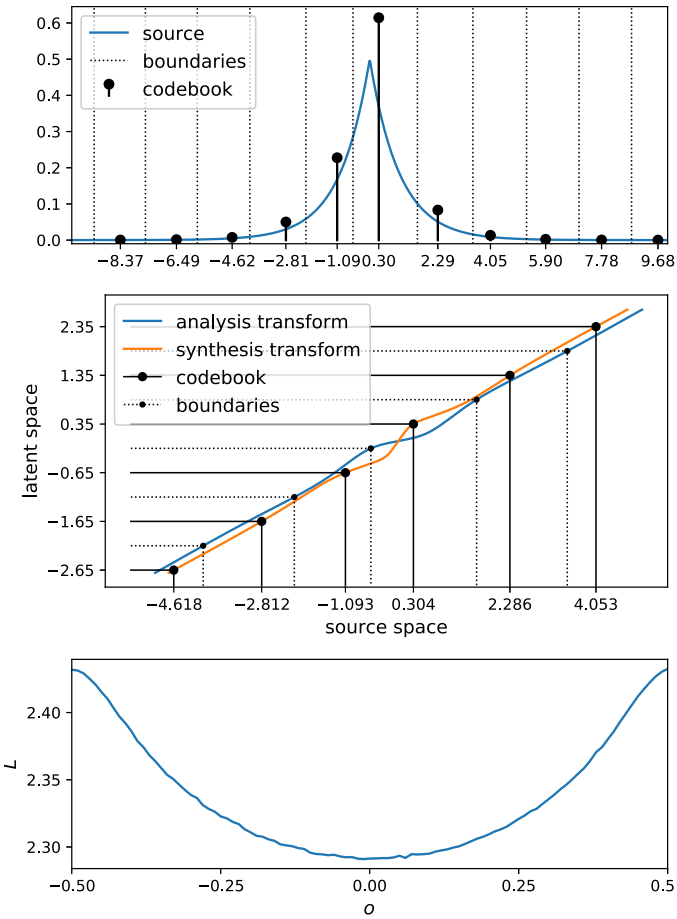

Fig. 6. Same instance of NTC as in Fig. 4, obtained by minimizing $\mathbb{E}_{\boldsymbol{o}}\, L_{\mathrm{NTC},\boldsymbol{o}}$. For this figure, a sub-optimal offset was chosen post hoc. Top: visualization of effective quantizer. Center: analysis and synthesis transforms giving rise to the quantizer. Note that the transforms themselves are identical to the ones in Fig. 4, but the quantization in the latent space is performed with an offset $o = .35$. Bottom: $L_{\mathrm{NTC},o}$ as a function of o.

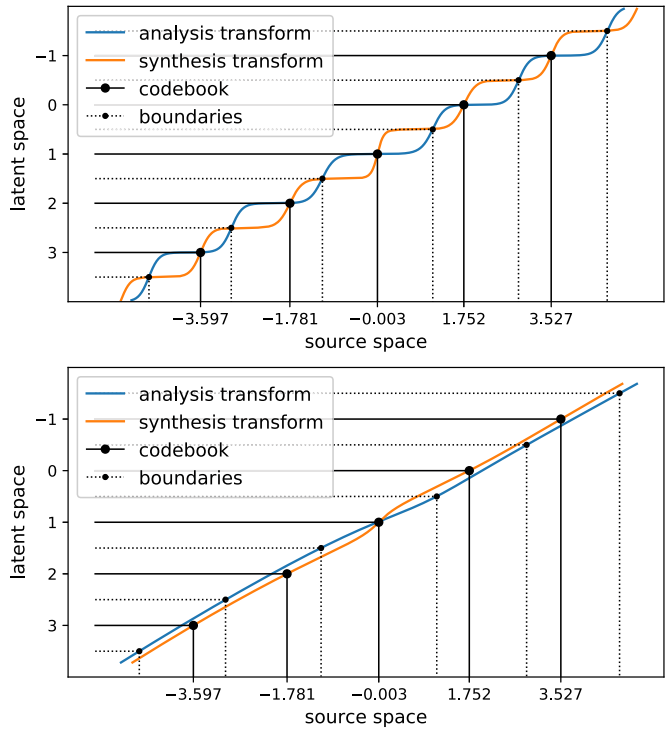

Fig. 7. Transforms for Laplace source with explicit soft quantization due to [29]. Top: ANN transforms including the explicit soft quantization. Bottom: ANN transforms excluding the explicit soft quantization. With this technique, the ANNs themselves can implement smoother (i.e., in some sense, simpler) functions.

Agustsson and Theis [29] discuss augmenting the transforms with a soft quantization function (and making appropriate modifications to the entropy model), which explicitly implements the curvature observed in Fig. 4. The soft quantization function has a *temperature* parameter, interpolating between the identity function and hard quantization. By explicitly modeling this behavior, the technique relieves the ANN itself from implementing it (Fig. 7), and represents a more controlled approach to bridging the gap between quantization and additive uniform noise while retaining near-optimal performance at all rates (Fig. 3). The temperature parameter allows explicitly trading off the bias of the proxy loss with the variance of the gradients. The method also suggests that the hard quantization offset after training can simply be chosen to be consistent with the soft quantization offset during training. However, it requires careful choice of an annealing schedule for the temperature. For simplicity, the experiments in this paper use the mode-centering approach described above.

Other authors choose to retain the dithering proxy for the rate term, but use a *straight-through* gradient estimate for the distortion term (effectively computing the distortion loss with constant-offset quantization during training, but replacing the gradient expression of the quantization operation with the identity function [30]–[33]. We have found this approach to yield reasonable results at higher rates, but at low rates, the ad-hoc nature of this approach leads to problematic behaviors of the transforms (RD performance plotted in Fig. 3, illustration in Fig. 8).

B. Nonlinear Transforms

ANNs are known as universal function approximators, and as such we permitted ourselves to ignore their details in the examples above. However, it is crucial to take into account their limitations, some of which arise as a function of their architecture. This is particularly important for complex or high-dimensional source distributions, such as natural images. With higher complexity and rates (larger values of λ), the optimal transforms generally are more complex and require neural networks with an increasing number of parameters.[3]

In general, neural networks are compositions of layers (parametric functions $\mathbb{R}^A \to \mathbb{R}^B$), wherein each layer typically consists of a linear transformation such as matrix multiplication or convolution, followed by the addition of a bias vector, followed

[3]It could be argued that in the high-rate limit, the transforms should collapse to identity functions. However, we haven't observed this effect for image compression models and practically interesting rate–distortion trade-offs, suggesting that this is only the case for extremely high rates.

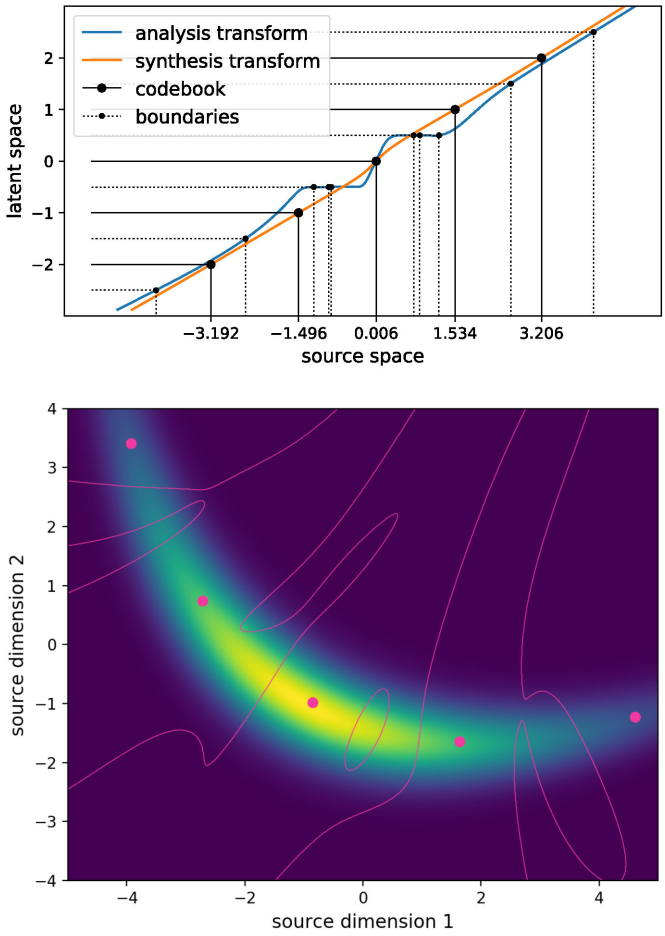

Fig. 8. Instabilities observed with straight-through proxy objective at low rates for the Laplace (top) and banana (bottom) distribution. While the synthesis transform tends to be smoother, the analysis transform begins oscillating around the locations of bin boundaries, leading effectively to discontiguous quantization bins.

in turn by a nonlinear function, which is typically applied separately on each vector dimension:

$$\boldsymbol{v} = g(\boldsymbol{r}), \text{ with } \boldsymbol{r} = \boldsymbol{W}\boldsymbol{u} + \boldsymbol{b}, \tag{15}$$

where $\boldsymbol{u} \in R^A$ is the input vector to the layer, $\boldsymbol{v} \in R^B$ are the layer's outputs or *activations*, and $\boldsymbol{W} \in \mathbb{R}^{B\times A}$ and $\boldsymbol{b} \in \mathbb{R}^B$ are the layer's parameters. For the NTC examples above, we used neural networks with 4 fully connected (i.e., non-convolutional) layers, the first three using the softplus nonlinearity ($g(x) = \ln(1 + e^x)$ elementwise), while the last layer omits the nonlinearity in order not to constrain the range of the transform to positive values. The *approximation capacity*, i.e., the capability of the neural network to approximate increasingly complex functions, grows with the number of units per layer (A, B), as well as the depth of the network (the number of layers). Above, we chose $A = B = 100$ (except that we set $A = N$ for the first, and $B = M = N$ for the last layer in g_a; analogous for g_s), which we found empirically to be large enough for all chosen values of λ.

For practical sources such as images, video, or audio, imposing special structure in the transforms may have significant benefits in terms of computational complexity, training data

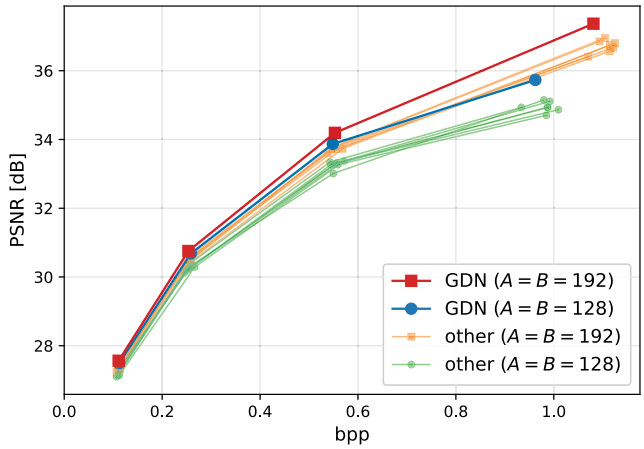

Fig. 9. Rate–distortion performance of an NTC model with GDN vs. a collection of pointwise nonlinearities on the Kodak testset [34] (after [35], with more nonlinearities: ReLU, leaky ReLU, tanh, softplus, ELU [36], SELU [37], Swish [38]). We compare networks with a different number of hidden units A, B per layer. At low rates, the approximation capacity of all networks is sufficient, and performance converges. At high rates, the capacity of smaller networks saturates earlier; in this regime, the performance benefit of GDN vs. other activation functions becomes measurable.

efficiency, or both. Generally, NTC models for this type of data use combinations of architectural constraints, most commonly *convolutionality* in g_a and g_s, as well as *downsampling* in g_a and *upsampling* in g_s, making the transforms share certain characteristics with multi-scale filterbanks, and leading to latent vectors with a tensor structure, consisting of one or more spatial/temporal dimensions, as well as one *channel* dimension (akin to subbands). A detailed example of such an architecture is described by Ballé *et al.* [5].

It has been observed that spatially local normalization as a nonlinearity is beneficial in terms of the trade-off between number of units and RD performance in image compression. In particular, a computationally optimized version of generalized divisive normalization (GDN) [39] as used in recent models is defined as

$$v_i = \frac{r_i}{\beta_i + \sum_j \gamma_{ij} |r_j|}, \tag{16}$$

where $\boldsymbol{r}$ are the linear responses of the layer, $\boldsymbol{v}$ represents the vector of normalized responses (the activations), and the vector $\boldsymbol{\beta}$ and matrix $\boldsymbol{\gamma}$ represent parameters of the transformation (both non-negative). The computation is typically replicated across spatial dimensions, as linear filtering is in convolutions, and i, j only index the channel dimension. Johnston *et al.* [40] show that the originally more complex form of GDN can be simplified to resemble a weighted ℓ^1-norm (plus a constant), as in (16), with negligible RD performance loss, but minimizing computationally costly exponentiations. Since ANNs can be understood as universal function approximators, the benefit of a particular architectural constraint may only become evident when the network is at its approximation capacity. Ballé [35] finds that this is the case at higher rates; i.e., for a constant network architecture, the superiority of GDN vs. pointwise nonlinearities disappears at lower rates (Fig. 9). Johnston *et al.* [40] also discuss the trade-off between computational complexity and RD performance resulting from other architectural choices in further

detail, such as the number of channels per layer or the number of decoder layers, and provide an algorithm to semi-automatically determine these hyperparameters.

IV. Learned Entropy Models

In linear transform coding with a Gaussian source assumption, the probabilistic model P in (9) is typically considered to be a distribution factorized over each latent dimension, since the KLT factorizes the source. However, as pointed out by Goyal [1], this is not necessarily a good model for real-world sources, and decorrelating the source is not generally RD optimal; this is also illustrated in Fig. 3.

In NTC, any differentiable density model p can be used to minimize (13) in principle; after determining a quantization offset, it can be translated into a corresponding probability mass function $P(\cdot; \boldsymbol{o})$, and used for entropy coding. However, to fully benefit from the function approximation capabilities of ANNs, techniques have been developed that allow jointly optimizing the transforms with an ANN-based density model, such that both components of the model are adapted to each other as best as possible. Key to this approach is conditioning. Generally, entropy coding methods such as arithmetic coding process one dimension at a time. Thus, we must be able to write the entropy model as a chain of conditionals:

$$P(\hat{\boldsymbol{y}} \mid \hat{\boldsymbol{z}}) = \prod_i P(\hat{y}_i \mid \hat{\boldsymbol{y}}_{:i}, \hat{\boldsymbol{z}}), \tag{17}$$

where $\hat{\boldsymbol{y}}$ is the quantized latent representation, $\hat{\boldsymbol{y}}_{:i}$ indicates the vector comprising the dimensions of $\hat{\boldsymbol{y}}$ preceding the ith (according to some predetermined ordering), and $\hat{\boldsymbol{z}}$ is another (optional) vector that must be known to both Alice and Bob. In the simplest case, P is assumed factorized, and the chain collapses to a product of independent scalar densities. Conditioning on some other vector $\hat{\boldsymbol{z}}$ typically requires transmitting the vector as side information, and thus corresponds to forward adaptation (FA) of the density model. Conditioning on the preceding dimensions of $\hat{\boldsymbol{y}}$ can be done without additional side information, but requires interleaving the computation of the probabilities with the decoding of $\hat{\boldsymbol{y}}$; this is backward adaptation (BA). FA and BA have long been used in conventional image compression. Context-adaptive arithmetic coding is an example of BA; mode selection and signaling is an example of FA [41].

Learned forward adaptation was first described in the context of image compression [27], inspired by the observation that the magnitudes of spatially nearby elements of the latent tensor in a convolutional NTC for images tend to be correlated. As illustrated in Fig. 10, $\boldsymbol{y} = g_a(\boldsymbol{x})$ is further processed by an ANN h_a to produce a side information vector $\boldsymbol{z}$. The entropy model for $\hat{\boldsymbol{y}}$ is conditioned on the decoded $\hat{\boldsymbol{z}}$. Ballé *et al.* [27] assume elements of $\boldsymbol{y}$ to be zero-mean Gaussians, conditionally independent wrt. $\hat{\boldsymbol{z}}$; a non-zero mean model is introduced by Minnen *et al.* [28]. The entropy of $\hat{\boldsymbol{z}}$ is small enough to warrant the improved fit of $P(\hat{\boldsymbol{y}} \mid \hat{\boldsymbol{z}})$, effectively lowering the rate. Fig. 11 compares the rate–distortion performance of several learned image compression models with JPEG [42] and BPG, a variant of HEVC [10], in terms of peak signal-to-noise ratio (PSNR) as well as MS-SSIM, a popular perceptual image quality metric [43]. The introduction of FA into NTC leads to a significant improvement of RD performance with respect to both metrics.

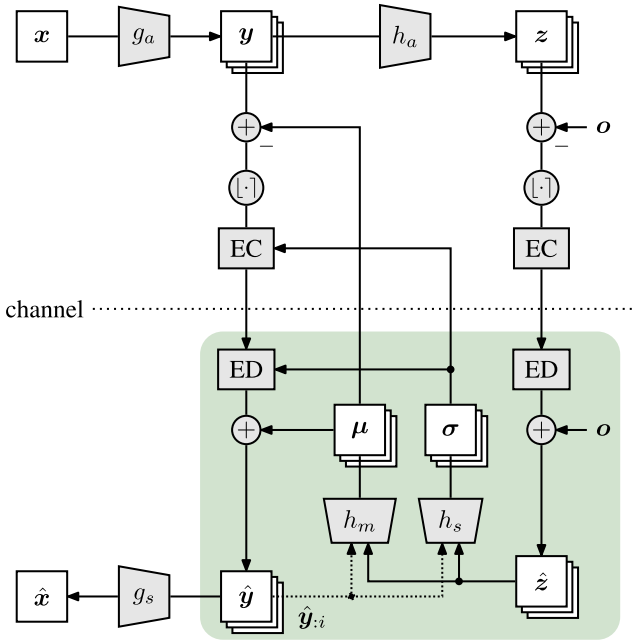

Fig. 10. Illustration of a nonlinear transform coder using both learned forward and backward adaptation. $\boldsymbol{x}$ is the source vector, $\hat{\boldsymbol{x}}$ the reconstruction. $\boldsymbol{y}$ is a latent representation tensor, and $\hat{\boldsymbol{y}}$ its uniformly quantized counterpart. $\boldsymbol{z}$ and $\hat{\boldsymbol{z}}$ are an analogous hierarchical latent representation computed via a transform h_a, representing side information. While the entropy model on $\hat{\boldsymbol{z}}$ is predetermined, the entropy model on $\hat{\boldsymbol{y}}$ is here assumed conditionally independent Gaussian with mean tensor $\boldsymbol{\mu}$ and standard deviation tensor $\boldsymbol{\sigma}$. Both tensors are computed as functions of previously decoded values of $\hat{\boldsymbol{y}}$ (backward adaptation) and the side information $\hat{\boldsymbol{z}}$ (forward adaptation) using the ANNs h_m and h_s. While Bob begins with entropy decoding $\hat{\boldsymbol{z}}$ (ED), and then uses it to decode $\hat{\boldsymbol{y}}$, Alice must have access to the entropy model on $\hat{\boldsymbol{y}}$ to entropy encode it (EC). Thus, in addition to computing the upper half of the diagram, she must also compute the section in the shaded box. The quantization offset of $\boldsymbol{y}$ is assumed aligned with the conditional mean of the Gaussian, making the entropy model only dependent on $\boldsymbol{\sigma}$. To enable reliable cross-platform decoding of $\hat{\boldsymbol{y}}$, h_s may be computed using a learned integer transform and translated into an arithmetic code via lookup tables.

Minnen *et al.* [28] are among the first authors to discuss learned backward adaptation. They introduce a spatially autoregressive model, where $\hat{\boldsymbol{y}}$ is processed one spatial location at a time, producing a distribution for each channel vector conditioned on previously decoded spatial locations. Combined with FA, the model produces further RD gains over the FA-only model (Fig. 11), and outperforms BPG.

A downside of BA compared to FA is that it impedes computational parallelism: the system must alternate between computing conditional probabilities and entropy decoding. On the other hand, with FA, Alice may effectively convey more information to Bob than necessary.[4] To alleviate the computational bottleneck

[4]We can write the excessive information as $H(\hat{\boldsymbol{z}} \mid \hat{\boldsymbol{y}})$, i.e. the conditional entropy of the side information $\hat{\boldsymbol{z}}$ given the latents $\hat{\boldsymbol{y}}$. It is also referred to as *bits-back* cost [44], [45]; practical algorithms to asymptotically get these "bits back" have been proposed in [46] and more recently in [47]. Note that in all

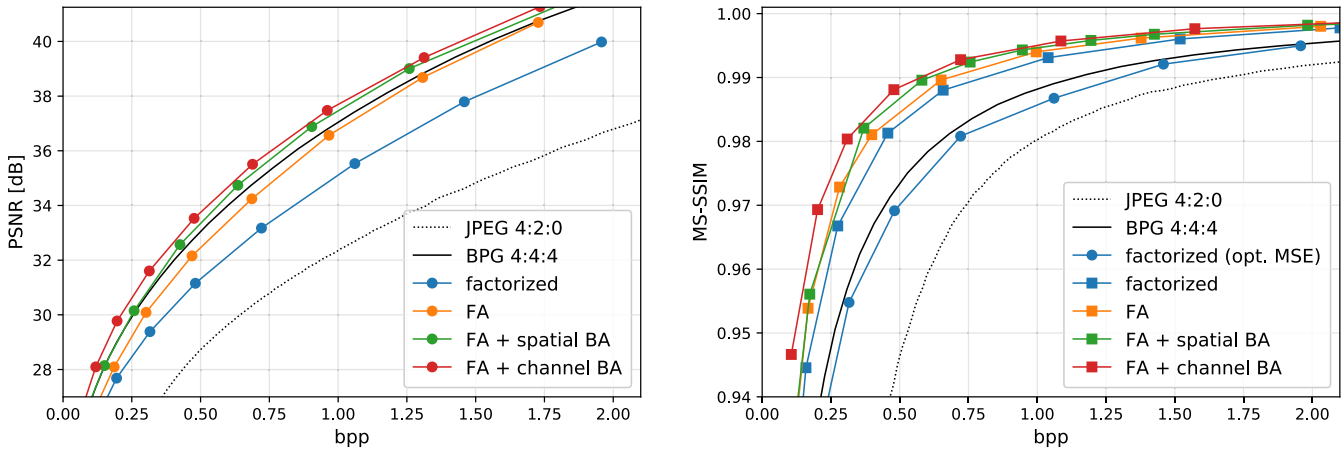

Fig. 11. Image compression performance of NTC on the Kodak testset [34], comparing different learned entropy models to JPEG and BPG, which is a popular variant of HEVC, a relatively recent and popular commercial method. The *factorized* and *FA* models replicate [27]; the *FA + spatial BA* model is due to [28]; the *FA + channel BA* model is provided by [33]. We compare models that are approximately equivalent in terms of optimization procedure and architecture of analysis and synthesis transforms. Conditional entropy models lead to significant improvements over factorized models. With sophisticated entropy modeling, learned image compression compares favorably to BPG in terms of PSNR, which BPG is optimized for. Regarding MS-SSIM, note that even the factorized model optimized for mean squared error (MSE) performs relatively closely to BPG, and when optimized for MS-SSIM, far outperforms it. We attribute this to the fact that regardless of the distortion measure, nonlinear transforms are better suited to model the source distribution (Fig. 1), and that MS-SSIM captures certain characteristics of the source that are also relevant perceptually (Fig. 14).

in BA, Minnen and Singh [33] introduce a model that iterates over channel slices rather than spatial locations, which is more amenable to parallelization using GPUs and, along with further modeling improvements, presents a significant improvement over traditional methods.

A problem frequently encountered with conditional entropy models is numerical determinism. To make image compression models practically relevant, they need to be implemented on a wide variety of hardware platforms. However, when probabilities are computed using floating-point arithmetic, numerical round-off errors can lead to catastrophic decoding failures due to the sensitivity of entropy coding with respect to discrepancies in the probability model between sender and receiver. The exact numerical round-off at each layer of an ANN depends on the hardware representation of floating point numbers, as well as the mode of parallelism, because round-off errors are not associative. This problem is typically handled in linear transform coders by using lookup tables to model probabilities, e.g. [41]. Ballé *et al.* [48] provide a solution for ANN-based entropy modeling, where ANNs are trained using floating-point arithmetic, but use integer arithmetic when deployed. This enables reliable decoding on arbitrary hardware platforms for the above-mentioned class of entropy models.

V. RD Traversal With λ-Parameterization

The loss function in (13) is optimized in expectation over the source distribution. The resulting transform thus jointly minimizes the rate and the expected distortion d between the source and the reconstruction, for a fixed trade-off predetermined by the choice of λ. In many linear transform coders, the system is parameterized by the quantization step size, such that only one set of transforms is needed to continuously traverse a range of RD trade-offs. Using this approach with a single set of trained nonlinear transforms was first explored by Dumas *et al.* [49]. However, this is not generally optimal.

With NTC, the transforms and/or the entropy model can be made more general functions of λ. One method to "condition" an ANN, first introduced in the context of stylization tasks [50], is to insert additional computations between layers, such as affine transformations:

$$\boldsymbol{w} = h_f(\lambda) \odot \boldsymbol{v} + h_b(\lambda), \tag{18}$$

where $\boldsymbol{v}$ are the outputs of a layer, $\boldsymbol{w}$ are the inputs to the next layer, and $\odot$ represents elementwise multiplication. In this context, h_f and h_b are parametric functions of λ that can be computed themselves via ANNs. The parameters of these ANNs in turn are optimized for the RD objective (13) as well. Parameterizing the entropy model and transforms this way was proposed earlier by Choi *et al.* [51], and Dosovitskiy and Djolonga [52]. We simplify this approach here by noting that, since h_f and h_b are functions of a scalar, they may be conveniently defined via first-order splines (i.e., piecewise linear functions). Additionally, we propose to remove the λ-parameterization of the entropy model, and for transforms using GDN, to simply treat its parameters ($\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$) as functions of λ instead of using affine transformations.

We carried out experiments with an NTC model with FA following the experimental setup of Ballé *et al.* [27], but with 160 channels per layer. When implementing h_f and h_b with ANNs, we used two-layer networks with 128 hidden units for each scalar element produced by the functions. For the spline implementation, we used a first-order spline with 25 parameters. First, we found that the optimization of ANN-based parameterization is numerically more difficult than first-order splines (Fig. 13).

models compared in Fig. 11, the side information only amounts to a fraction of the total bit rate. Thus, since $H(\hat{\boldsymbol{z}}) \geq H(\hat{\boldsymbol{z}} \mid \hat{\boldsymbol{y}})$, the bits-back cost in these models is negligible.

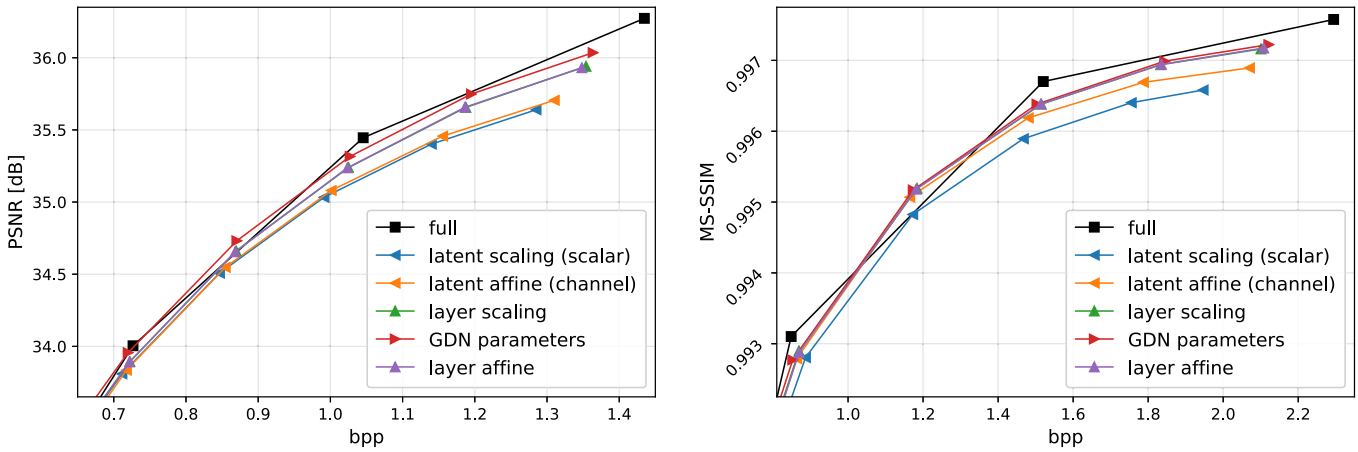

Fig. 12. Performance comparison of λ-parameterization techniques implementing (18) using first-order splines, and parameterizing only the transforms, not the entropy model. All models have the same architecture and were optimized using the dithering proxy either for MSE (left panel) or MS-SSIM (right panel). Results are shown for the Kodak testset [34]. As in Fig. 9, differences between models become more evident at high rates. Optimization using the straight-through proxy gives consistent results (not shown). *Full:* RD performance of separate models for each λ; *latent scaling:* only the output of the g_a and the input to g_s are scaled with a single scalar each (h_b is zero); *latent affine:* each output channel of g_a and each input channel of g_s are subjected to a scalar affine transformation, as in (18); *layer affine:* each channel of each layer of g_a and g_s is subjected to a scalar affine transformation; *layer scaling:* ditto, except that h_b is zero; *GDN parameters:* rather than adding a scaling between layers, the parameters of each instance of GDN in the transforms ($\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$) are represented as first-order splines dependent on λ. We note that a simple affine transformation of the latent space, corresponding to varying the quantization interval, is not sufficient to maintain comparable performance with the full model. Scaling the activations of each layer appears sufficient, while reparameterizing GDN as a function of λ yields slightly better performance.

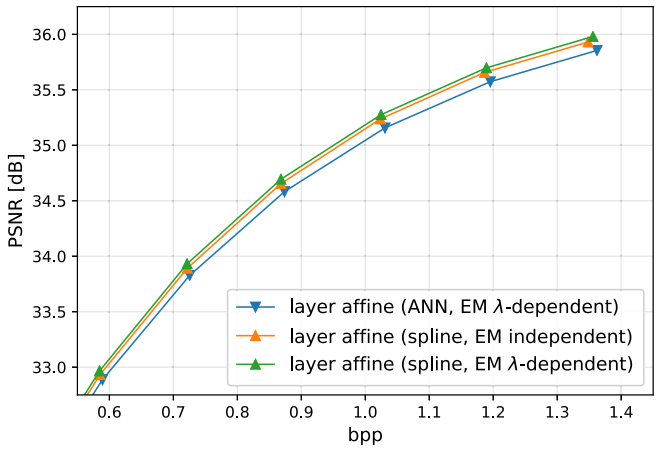

Fig. 13. Performance comparison of λ-parameterization using splines vs. ANNs, and of optionally applying it to the entropy model. As in Fig. 9, differences between models become more evident at high rates. Comparing models implementing (18) using ANNs vs. first-order splines, we find that the performance of ANNs is consistently worse than that of splines, despite a larger number of parameters, suggesting that optimization of ANNs may be numerically more difficult in this context. Comparing models using splines and either a λ-parameterized entropy model, or a forward-adaptive entropy model independent of λ, we find that the benefit of parameterizing the EM is rather small. Results shown are for the Kodak testset [34] and are optimized using the dithering proxy; the straight-through proxy yields consistent results.

Furthermore, we removed the λ-parameterization of the entropy model and noted that it is not crucial to RD performance (same figure). Along with the practical requirement that h_s needs to be implemented with integer arithmetic for cross-platform stability, this suggests that making the entropy model explicitly dependent on λ may not be worth the complexity of implementation.

Fig. 12 compares λ-parameterizations of only the transforms g_a and g_s, and using splines, in terms of RD performance. As for the experiments with different nonlinearities, differences between the parameterizations emerge at high rates, since the network capacities saturate in that regime. We find that a scaling or affine transformation of the latent space alone, roughly equivalent to parameterizing the quantization step size and offset, are not sufficient to achieve an RD performance equivalent to the family of full, non-parameterized models. However, any layer-wise parameterization appears close enough. This is explainable by the fact that the RD family of optimal entropy-constrained scalar quantizers cannot in general be parameterized by a scaling of the quantization offset (a notable exception being the Laplace source discussed above). GDN reparameterization performs the best in an RD sense, but also requires slightly more model parameters compared to the other methods, since (18) requires two length-B vectors, but $\boldsymbol{\gamma}$ is a $B \times B$ matrix.

VI. Related Work

Due to the resurgence of ANNs and data-driven computing in recent years, the field of data compression has received an influx of new ideas. While transform coding as a concept has been around for decades [53], one could observe a recent convergence of it with the idea of autoencoders [54]. Autoencoders, likewise, have been discussed for decades, but largely in a separate community. One notable step towards this convergence was the fusion of variational Bayesian methods with autoencoders, which introduced a probabilistic interpretation, making the connection to information-theoretic quantities such as entropy [55], [56]; another was the use of a dithering-based loss for optimization of nonlinear transform codes [24].

Fig. 14. Reconstructions of kodim15 [34] compressed by BPG (left, 0.0738 bpp) and by a learned NTC model [33] optimized for MS-SSIM (right, 0.0713 bpp). The combination of NTC and an MS-SSIM loss function, which is designed to model texture masking effects in the human visual system, leads to significantly better texture retention in the sweater and fewer geometric distortions on the person's face.

As is often the case in lossy compression, the field of nonlinear transform coding has been driven forward by the need to compress digital images. Early image compression models using ANNs include the work of Toderici *et al.* [7], who do not use entropy modeling; Rippel and Bourdev [8], who use a context-based adaptive entropy coder not jointly optimized with the transforms; and Ballé *et al.* [5] and Theis *et al.* [6], who jointly optimize the transforms with continuous entropy models, the latter with a different formulation than what we use here. Agustsson *et al.* [9] combine an autoencoder with VQ in the latent space over small blocks of coefficients, utilizing a soft quantization proxy. More recent work using soft notions of quantization includes the work of Agustsson and Theis [29], and Alexandre *et al.* [57].

Beyond the use of convolutional filtering, up- or downsampling, and special nonlinearities [5] as discussed earlier, many authors exploit properties of the image distribution by way of introducing special structure into the transforms, such as multi-scale architectures [8], [58], [59]; non-local, or "attention"-based network architectures [60], [61]; or iteration built into the transforms [62], [63]. Recently, the topic of extending nonlinear transform codes to video signals has received much attention, and the space of possible network architectures suitable for this application has been explored, including spatiotemporal convolutions, optical flow networks, as well as multi-scale linear filtering [30], [64]–[74]. Ballé *et al.* [48] develop integer architectures for learned entropy models, in order to guarantee reliable decoding on arbitrary hardware platforms, and Johnston *et al.* [40] discuss selecting architecture parameters, such as the number of layers, or number of channels per layer, while taking into account the RD performance.

Notable work in the space of learned entropy models includes Minnen *et al.* [75], which use block-based forward adaptation, and several other concurrent publications performing learned backward adaptation [28], [76], [77]. More recent work on learned backward adaptation includes the papers by M. Li *et al.* [78], Guo *et al.* [79].

The first work exploring the RD trade-off with a single set of nonlinear transforms is due to Dumas *et al.* [49], who explore varying the quantization step size, as in linear TC (corresponding most closely to "latent affine" in Fig. 12). Guarda *et al.* [80] use the same method for coding point-cloud geometries. Layer-wise parameterization as in (18) was introduced for image compression by Choi *et al.* [51]. It was generalized by Dosovitskiy and Djolonga [52] for a range of other tasks including compression. Our technical contribution is to simplify the parameterization and to adapt it for GDN.

While many authors have explored ANN-based compression in the context of existing, commercially viable image and video compression methods [81]–[87], other authors begin with nonlinear transforms, and explore incorporating concepts traditionally used in linear TC, such as energy compaction [88], wavelets [89], [90], or trellis coded quantization [91]. Lossless image compression based on learned entropy models has been explored by, e.g., Mentzer *et al.* [77], [92]. Still others explore the intersection between learned image compression and other vision tasks such as content and semantic analysis [93]–[96], inpainting [97], super-resolution [98], quality enhancement [99], or encryption [100].

Another topic of active research is the question of more "perceptual" image compression. Ballé *et al.* [27] discuss optimization of NTC models for squared error vs.

MS-SSIM. Ding *et al.* [101] provide a more in-depth discussion, with an even larger set of different perceptual distortion measures. Interestingly, Chen *et al.* [102] provide a method to optimize NTC models for non-differentiable perceptual metrics. Valenzise *et al.* [103], Cheng *et al.* [104], and Ascenso *et al.* [105] study the perceived image quality of learned image compression models optimized for metrics such as squared error and MS-SSIM with the help of human rating experiments. Other authors have explored augmenting the fixed distortion measure with ANN-based losses that have shown visually convincing results in image generation tasks, most notably generative adversarial networks (GANs) [106]–[110]. Blau and Michaeli [111] formulate theoretical limits for the three-way trade-off between the rate, the reconstruction quality of an image compression method, as well as divergence measures between the source distribution and the marginal distribution of image reconstructions, the latter of which are related to adversarial losses.

While image compression dominates the literature on NTC, other applications have emerged as well, such as compression of point clouds [80], [112], volumetric data [113], ANN features for tasks like large-scale image retrieval [32], and compression of ANN parameters themselves [31]. Further reviews of the current state of the literature, specifically with respect to image and video compression, are given by Liu *et al.* [91], and Ma *et al.* [114].

VII. Conclusion

We have presented an overview of nonlinear transform coding, which heavily relies on ANNs as universal function approximators and stochastic optimization of the rate–distortion Lagrangian. We introduced the VECVQ algorithm, a stochastically optimized version of entropy-constrained VQ, as well as a novel method of λ-parameterization. Furthermore, we provide the first direct comparison of different λ-parameterizations for image compression models.

Most of the desirable properties of NTC stem from the use of stochastic RD optimization and ANNs. Given enough computing power, NTC models are quickly adaptible to arbitrary data sources, including domain-specific imagery (e.g., medical, astronomical) or new modalities of multimedia, since many parameters of the system can be found using end-to-end optimization rather than manual experimentation. This can lower prototyping times from years to weeks.

NTC models can also be directly optimized for any differentiable distortion measure (or, more generally, distortion loss). Fig. 14 illustrates the visual difference of a model optimized for MS-SSIM vs. squared error, which has long been noted as perceptually flawed [115]. This adaptability of NTC will dovetail with the development of better and more general perceptual losses, of which hybrid adversarial losses are an example [106]–[110].

With the rapidly increasing availability of parallelized computation, we believe NTC will fundamentally change the landscape of practical data compression.

Acknowledgment

The authors would like to thank the anonymous reviewers, as well as Lucas Theis and Aaron Wagner for their helpful comments on the draft and insightful discussions.

References

[1] V. K. Goyal, "Theoretical foundations of transform coding," *IEEE Signal Process. Mag.*, vol. 18, no. 5, pp. 9–21, Sep. 2001, doi: 10.1109/79.952802.

[2] A. Gersho and R. M. Gray, *Vector Quantization Signal Compression*. Norwell, MA, USA: Kluwer, 1992.

[3] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural Netw.*, vol. 6, no. 6, 1993, doi: 10.1016/S0893-6080(05)80131-5.

[4] G. Toderici *et al.*, "Variable rate image compression with recurrent neural networks," in *Proc. 4th Int. Conf. Learn. Representations*, 2016, *arXiv: 1511.06085*.

[5] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[6] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[7] G. Toderici *et al.*, "Full resolution image compression with recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5435–5443, doi: 10.1109/CVPR.2017.577.

[8] O. Rippel and L. Bourdev, "Real-time adaptive image compression," in *Proc. 34th Int. Conf. Mach. Learn.*, ser. Proc. of Machine Learning Research, 2017, vol. 70, pp. 2922–2930.

[9] E. Agustsson *et al.*, "Soft-to-hard vector quantization for end-to-end learning compressible representations," in *Proc. Adv. Neural Inform. Process. Syst.*, 2017, pp. 1141–1151.

[10] "ITU-R rec. H.265 & ISO/IEC 23008-2: High efficiency video coding," 2013.

[11] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy-constrained vector quantization," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, no. 1, pp. 31–42, Jan. 1989, doi: 10.1109/29.17498.

[12] T. M. Cover and J. A. Thomas, *Elements Inf. Theory*, 2nd ed. Wiley, 2006.

[13] G. J. Sullivan, "Efficient scalar quantization of exponential and Laplacian random variables," *IEEE Trans. Inf. Theory*, vol. 42, no. 5, pp. 1365–1374, Sep. 1996, doi: 10.1109/18.532878.

[14] T. Berger, "Optimum quantizers and permutation codes," *IEEE Trans. Inf. Theory*, vol. IT-18, no. 6, pp. 759–765, Nov. 1972, doi: 10.1109/TIT.1972.1054906.

[15] N. Farvardin and J. Modestino, "Optimum quantizer performance for a class of non-Gaussian memoryless sources," *IEEE Trans. Inf. Theory*, vol. IT-30, no. 3, pp. 485–497, May 1984, doi: 10.1109/TIT.1984.1056920.

[16] L. Bottou and Y. Bengio, "Convergence properties of the k-means algorithms," in *Proc. Adv. Neural Inf. Process. Syst. 7*, 1994, pp. 585–592.

[17] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982, (Previously an unpublished Bell Lab tech. note, 1957.), doi: 10.1109/TIT.1982.1056489.

[18] J. Max, "Quantizing for minimum distortion," *IRE Trans. Inf. Theory*, vol. 6, no. 1, 1960, doi: 10.1109/TIT.1960.1057548.

[19] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probability*, 1967, pp. 281–297.

[20] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COMM-28, no. 1, pp. 84–95, Jan. 1980, doi: 10.1109/TCOM.1980.1094577.

[21] W. R. Bennett, "Spectra of quantized signals," *Bell Syst. Tech. J.*, vol. 27, no. 3, 1948, doi: 10.1002/j.1538-7305.1948.tb01340.x.

[22] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inf. Theory*, vol. IT-25, no. 4, pp. 373–380, Jul. 1979, doi: 10.1109/TIT.1979.1056067.

[23] L. Schuchman, "Dither signals and their effect on quantization noise," *IEEE Trans. Commun. Technol.*, vol. COM-12, no. 4, pp. 162–165, Dec. 1964, doi: 10.1109/TCOM.1964.1088973.

[24] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimization of nonlinear transform codes for perceptual quality," in *Proc. Picture Coding Symp.*, 2016, pp. 1–5, doi: 10.1109/PCS.2016.7906310.

[25] I. Higgins *et al.*, "β-VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[26] A. A. Alemi, B. Poole, I. Fischer, J. V. Dillon, R. A. Saurous, and K. Murphy, "Fixing a broken ELBO," in *Proc. 35th Int. Conf. Mach. Learning*, ser. Proc. of Machine Learning Research, 2018, vol. 80, pp. 159–168.

[27] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *Proc. 6th Int. Conf. Learn. Representations*, 2018.

[28] D. Minnen, J. Ballé, and G. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Proc. Adv. Neural Inform. Process. Syst.*, 2018, pp. 10771–10780.

[29] E. Agustsson and L. Theis, "Universally quantized neural compression," 2020, accepted at NeurIPS, *arXiv: 2006.09952*.

[30] E. Agustsson, D. Minnen, N. Johnston, J. Ballé, S. J. Hwang, and G. Toderici, "Scale-space flow for end-to-end optimized video compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8500–8509, doi: 10.1109/CVPR42600.2020.00853.

[31] D. Oktay, J. Ballé, S. Singh, and A. Shrivastava, "Scalable model compression by entropy penalized reparameterization," in *Proc. 8th Int. Conf. Learn. Representations*, 2020.

[32] S. Singh, S. Abu-El-Haija, N. Johnston, J. Ballé, A. Shrivastava, and G. Toderici, "End-to-end learning of compressible features," *IEEE Int. Conf. Image Process. (ICIP)*, Abu Dhabi, United Arab Emirates, 2020, pp. 3349–3353, doi: 10.1109/ICIP40778.2020.9190860.

[33] D. Minnen and S. Singh, "Channel-wise autoregressive entropy models for learned image compression," *IEEE Int. Conf. Image Process. (ICIP)*, Abu Dhabi, United Arab Emirates, 2020, pp. 3339–3343, doi: 10.1109/ICIP40778.2020.9190935.

[34] Eastman Kodak, "Kodak lossless true color image suite (PhotoCD PCD0992)," 1993. [Online]. Available: http://r0k.us/graphics/kodak/

[35] J. Ballé, "Efficient nonlinear transforms for lossy image compression," in *Proc. Picture Coding Symp.*, 2018, pp. 248–252.

[36] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," in *Proc. 4th Int. Conf. Learn. Representations*, 2016, *arXiv: 1511.07289*.

[37] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Proc. Adv. Neural Inform. Process. Syst.*, 2017.

[38] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," 2017, *arXiv: 1710.05941*.

[39] J. Ballé, V. Laparra, and E. P. Simoncelli, "Density modeling of images using a generalized normalization transformation," in *Proc. 4th Int. Conf. Learn. Representations*, 2016, *arXiv: 1511.06281*.

[40] N. Johnston, E. Eban, A. Gordon, and J. Ballé, "Computationally efficient neural image compression," 2019, *arXiv: 1912.08771*.

[41] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 620–636, Jul. 2003.

[42] "ITU-R rec. T.81 & ISO/IEC 10918-1: Digital compression and coding of continuous-tone still images," 1992.

[43] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *Proc. 37th Asilomar Conf. Signals, Syst. Comput.*, 2003, vol. 2, pp. 1398–1402, doi: 10.1109/ACSSC.2003.1292216.

[44] C. S. Wallace, "Classification by minimum-message-length inference," in *Proc. Adv. Neural Inform. Process. Syst.*, 1990, pp. 72–81, doi: 10.1007/3-540-53504-7_63.

[45] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length and Helmholtz free energy," in *Proc. Adv. Neural Inform. Process. Syst.*, 1993, pp. 3–10.

[46] B. J. Frey and G. E. Hinton, "Efficient stochastic source coding and an application to a Bayesian network source model," *The Comput. J.*, vol. 40, no. 2/3, 1997, doi: 10.1093/comjnl/40.2_and_3.157.

[47] J. Townsend, T. Bird, and D. Barber, "Practical lossless compression with latent variables using bits back coding," in *Proc. 7th Int. Conf. Learn. Representations*, 2019.

[48] J. Ballé, D. Minnen, and N. Johnston, "Integer networks for data compression with latent-variable models," in *Proc. 6th Int. Conf. Learn. Representations*, 2018.

[49] T. Dumas, A. Roumy, and C. Guillemot, "Autoencoder based image compression: Can the learning be quantization independent?," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 1188–1192, doi: 10.1109/ICASSP.2018.8462263.

[50] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[51] Y. Choi, M. El-Khamy, and J. Lee, "Variable rate deep image compression with a conditional autoencoder," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3146–3154, doi: 10.1109/ICCV.2019.00324.

[52] A. Dosovitskiy and J. Djolonga, "You only train once: Loss-conditional training of deep networks," in *Proc. 8th Int. Conf. Learn. Representations*, 2020.

[53] N. Ahmed and K. R. Rao, *Orthogonal Transforms for Digital Signal Process.* Springer, 1975.

[54] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Sci.*, vol. 313, no. 5786, 2006.

[55] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. 2nd Int. Conf. Learn. Representations*, 2014, *arXiv: 1312.6114*.

[56] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximative inference in deep generative models," in *Proc. 31st Int. Conf. Mach. Learn.*, ser. Proc. of Machine Learning Research, vol. 32, 2014, pp. 1278–1286.

[57] D. Alexandre, C.-P. Chang, W.-H. Peng, and H.-M. Hang, "Learned image compression with soft bit-based rate-distortion optimization," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 1715–1719, doi: 10.1109/ICIP.2019.8803120.

[58] K. M. Nakanishi, S.-i. Maeda, T. Miyato, and D. Okanohara, "Neural multi-scale image compression," in *Proc. Comput. Vis. – ACCV*, 2018, pp. 718–732, doi: 10.1007/978-3-030-20876-9_45.

[59] C. Cai, L. Chen, X. Zhang, and Z. Gao, "Efficient variable rate image compression with multi-scale decomposition network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 12, pp. 3687–3700, Dec. 2019.

[60] H. Liu, T. Chen, Q. Shen, and Z. Ma, "Practical stacked non-local attention modules for image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2019.

[61] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 7936–7945, doi: 10.1109/CVPR42600.2020.00796.

[62] J. Cai and L. Zhang, "Deep image compression with iterative non-uniform quantization," in *Proc. 25th IEEE Int. Conf. Image Process.*, 2018, doi: 10.1109/ICIP.2018.8451411.

[63] A. G. Ororbia *et al.*, "Learned neural iterative decoding for lossy image compression systems," in *Proc. Data Compression Conf.*, 2019, pp. 3–12, doi: 10.1109/DCC.2019.00008.

[64] C.-Y. Wu, N. Singhal, and P. Krähenbühl, "Video compression through image interpolation," in *Proc. Comput. Vis. – ECCV*, 2018, pp. 425–440, doi: 10.1007/978-3-030-01237-3_26.

[65] J. Han, S. Lombardo, C. Schroers, and S. Mandt, "Deep probabilistic video compression," in *Proc. 7th Int. Conf. Learn. Representat.*, 2019.

[66] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, and Z. Gao, "DVC: An end-to-end deep video compression framework," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10998–11007, doi: 10.1109/CVPR.2019.01126.

[67] Z. Chen, T. He, X. Jin, and F. Wu, "Learning for video compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 2, pp. 566–576, Feb. 2020, doi: 10.1109/TCSVT.2019.2892608.

[68] O. Rippel, S. Nair, C. Lew, S. Branson, A. G. Anderson, and L. Bourdev, "Learned video compression," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3453–3462, doi: 10.1109/ICCV.2019.00355.

[69] A. Habibian, T. van Rozendaal, J. M. Tomczak, and T. S. Cohen, "Video compression with rate-distortion autoencoders," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 7032–7041, doi: 10.1109/ICCV.2019.00713.

[70] A. Djelouah, J. Campos, S. Schaub-Meyer, and C. Schroers, "Neural inter-frame compression for video coding," in *Proc. IEEE/CVF Int. Conf. Comput. Vis*, 2019, pp. 6420–6428, doi: 10.1109/ICCV.2019.00652.

[71] S. Lombardo, J. Han, C. Schroers, and S. Mandt, "Deep generative video compression," in *Proc. Adv. Neural Inform. Process. Syst.*, 2019, pp. 9287–9298.

[72] A. Golinski, R. Pourreza, Y. Yang, G. Sautiere, and T. S. Cohen, "Feedback recurrent autoencoder for video compression," 2020, *arXiv: 2004.04342*.

[73] R. Yang, F. Mentzer, L. Van Gool, and R. Timofte, "Learning for video compression with recurrent auto-encoder and recurrent probability model," 2020, *arXiv: 2006.13560*.

[74] G. Lu, X. Zhang, W. Ouyang, L. Chen, Z. Gao, and D. Xu, "An end-to-end learning framework for video compression," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020, to be published, doi: 10.1109/TPAMI.2020.2988453.

[75] D. Minnen, G. Toderici, S. Singh, S. J. Hwang, and M. Covell, "Image-dependent local entropy models for learned image compression," in *Proc. 25th IEEE Int. Conf. Image Process.*, 2018, pp. 430–434, doi: 10.1109/ICIP.2018.8451502.

[76] J. Lee, S. Cho, and S.-K. Beack, "Context-adaptive entropy model for end-to-end optimized image compression," in *Proc. 7th Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=HyxKIiAqYQ

[77] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. Van Gool, "Conditional probability models for deep image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4394–4402, doi: 10.1109/CVPR.2018.00462.

[78] M. Li, K. Ma, J. You, D. Zhang, and W. Zuo, "Efficient and effective context-based convolutional entropy modeling for image compression," *IEEE Trans. Image Process.*, vol. 29, pp. 5900–5911, 2020, doi: 10.1109/TIP.2020.2985225.

[79] Z. Guo, Y. Wu, R. Feng, Z. Zhang, and Z. Chen, "3-D context entropy model for improved practical image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2020, doi: 10.1109/CVPRW50498.2020.00066.

[80] A. F. R. Guarda, N. M. M. Rodrigues, and F. Pereira, "Deep learning-based point cloud geometry coding: RD control through implicit and explicit quantization," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops*, 2020, pp. 1–6, doi: 10.1109/ICMEW46912.2020.9106022.

[81] T. Laude and J. Ostermann, "Deep learning-based intra prediction mode decision for HEVC," in *Proc. Picture Coding Symp.*, 2016, pp. 1–5, doi: 10.1109/PCS.2016.7906399.

[82] F. Jiang, W. Tao, S. Liu, J. Ren, X. Guo, and D. Zhao, "An end-to-end compression framework based on convolutional neural networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 10, pp. 3007–3018, Oct. 2018, doi: 10.1109/TCSVT.2017.2734838.

[83] J. Li, B. Li, J. Xu, R. Xiong, and W. Gao, "Fully connected network-based intra prediction for image coding," *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3236–3247, Jul. 2018, doi: 10.1109/TIP.2018.2817044.

[84] C. Jia *et al.*, "Content-aware convolutional neural network for in-loop filtering in high efficiency video coding," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3343–3356, Jul. 2019, doi: 10.1109/TIP.2019.2896489.

[85] D. Liu, H. Ma, Z. Xiong, and F. Wu, "CNN-based DCT-like transform for image compression," in *Proc. MMM 2018: MultiMedia Model.*, 2018, pp. 61–72, doi: 10.1007/978-3-319-73600-6_6.

[86] Y. Hu, W. Yang, M. Li, and J. Liu, "Progressive spatial recurrent neural network for intra prediction," *IEEE Trans. Multimedia*, vol. 21, no. 12, pp. 3024–3037, Dec. 2019, doi: 10.1109/TMM.2019.2920603.

[87] P. Helle *et al.*, "Intra picture prediction for video coding with neural networks," in *Proc. Data Compression Conf.*, 2019, pp. 448–457, doi: 10.1109/DCC.2019.00053.

[88] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Energy compaction-based image compression using convolutional autoencoder," *IEEE Trans. Multimedia*, vol. 22, no. 4, pp. 860–873, Apr. 2020.

[89] P. Akyazi and T. Ebrahimi, "Learning-based image compression using convolutional autoencoder and wavelet decomposition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2019.

[90] C. Yang, Y. Zhao, and S. Wang, "Deep image compression in the wavelet transform domain based on high frequency sub-band prediction," *IEEE Access*, vol. 7, pp. 52484–52497, 2019, doi: 10.1109/ACCESS.2019.2911403.

[91] D. Liu, Y. Li, J. Lin, H. Li, and F. Wu, "Deep learning-based video coding: A review and a case study," *ACM Comput. Surv.*, vol. 53, no. 1, 2020, doi: 10.1145/3368405.

[92] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. V. Gool, "Practical full resolution learned lossless image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10621–10630, doi: 10.1109/CVPR.2019.01088.

[93] M. Li, W. Zuo, S. Gu, D. Zhao, and D. Zhang, "Learning convolutional networks for content-weighted image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3214–3223, doi: 10.1109/CVPR.2018.00339.

[94] J. Campos, S. Meierhans, A. Djelouah, and C. Schroers, "Content adaptive optimization for neural image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2019.

[95] M. Akbari, J. Liang, and J. Han, "DSSLIC: Deep semantic segmentation-based layered image compression," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 2042–2046, doi: 10.1109/ICASSP.2019.8683541.

[96] S. Luo, Y. Yang, Y. Yin, C. Shen, Y. Zhao, and M. Song, "DeepSIC: Deep semantic image compression," in *Proc. ICONIP 2018: Neural Inf. Process.*, 2018, pp. 96–106, doi: 10.1007/978-3-030-04167-0_9.

[97] M. H. Baig, V. Koltun, and L. Torresani, "Learning to inpaint for image compression," in *Proc. Adv Neural Inf. Process. Syst. 30*, 2017, pp. 1246–1255.

[98] S. Cao, C.-Y. Wu, and P. Krähenbühl, "Lossless image compression through super-resolution," 2020, *arXiv: 2004.02872*.

[99] J. Lee, S. Cho, and M. Kim, "A hybrid architecture of jointly learning image compression and quality enhancement with improved entropy minimization," 2019, *arXiv: 1912.12817*.

[100] X. Duan, J. Liu, and E. Zhang, "Efficient image encryption and compression based on a VAE generative model," *J. Real-Time Image Process.*, vol. 16, 2019, doi: 10.1007/s11554-018-0826-4.

[101] K. Ding, K. Ma, S. Wang, and E. P. Simoncelli, "Comparison of image quality models for optimization of image processing systems," 2020, *arXiv: 2005.01338*.

[102] L.-H. Chen, C. G. Bampis, Z. Li, A. Norkin, and A. C. Bovik, "ProxIQA: A proxy approach to perceptual optimization of learned image compression," *IEEE Trans. Image Process.*, vol. 30, pp. 360–373, 2021, doi: 10.1109/TIP.2020.3036752.

[103] G. Valenzise, A. Purica, V. Hulusic, and M. Cagnazzo, "Quality assessment of deep-learning-based image compression," in *Proc. IEEE 20th Int. Workshop Multimedia Signal Process.*, 2018, pp. 1–6.

[104] Z. Cheng, P. Akyazi, H. Sun, J. Katto, and T. Ebrahimi, "Perceptual quality study on deep learning based image compression," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 719–723, doi: 10.1109/ICIP.2019.8803824.

[105] J. Ascenso, P. Akyazi, F. Pereira, and T. Ebrahimi, "Learning-based image coding: Early solutions reviewing and subjective quality evaluation," in *Opt., Photon. Digital Technol. Imag. Appl. VI*, ser. Proc. SPIE, vol. 11353, 2020, doi: 10.1117/12.2555368.

[106] S. Santurkar, D. Budden, and N. Shavit, "Generative compression," in *Proc. Picture Coding Symp.*, 2018, pp. 258–262, doi: 10.1109/PCS.2018.8456298.

[107] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, and L. van Gool, "Generative adversarial networks for extreme learned image compression," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 221–231, doi: 10.1109/ICCV.2019.00031.

[108] M. Tschannen, E. Agustsson, and M. Lucic, "Deep generative models for distribution-preserving lossy compression," in *Proc. Adv. Neural Inform. Process. Syst.*, 2018, pp. 5929–5940.

[109] S. Kudo, S. Orihashi, R. Tanida, and A. Shimizu, "GAN-based image compression using mutual information maximizing regularization," in *Proc. Picture Coding Symp.*, 2019, pp. 1–5, doi: 10.1109/PCS48520.2019.8954548.

[110] F. Mentzer, G. Toderici, M. Tschannen, and E. Agustsson, "High-fidelity generative image compression," 2020, accepted at NeurIPS, *arXiv: 2006.09965*.

[111] Y. Blau and T. Michaeli, "Rethinking lossy compression: The rate-distortion-perception tradeoff," in *Proc. 36th Int. Conf. Mach. Learn.*, ser. Proc. of Machine Learning Research, vol. 97, 2019, pp. 675–685.

[112] M. Quach, G. Valenzise, and F. Dufaux, "Learning convolutional transforms for lossy point cloud geometry compression," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 4320–4324, doi: 10.1109/ICIP.2019.8803413.

[113] D. Tang *et al.*, "Deep implicit volume compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1290–1300, doi: 10.1109/CVPR42600.2020.00137.

[114] S. Ma, X. Zhang, C. Jia, Z. Zhao, S. Wang, and S. Wanga, "Image and video compression with neural networks: A review," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 6, pp. 1683–1698, Jun. 2020, doi: 10.1109/TCSVT.2019.2910119.

[115] B. Girod, "What's wrong with mean-squared error?," in *Digital Images and Human Vision*, A. B. Watson, Ed. Cambridge, MA, USA: MIT Press, 1993, pp. 207–220.

[116] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2325–2383, Oct. 1998, doi: 10.1109/18.720541.