Neural Estimation of the Rate-Distortion Function With Applications to Operational Source Coding

Eric Lei, *Graduate Student Member, IEEE*, Hamed Hassani, and Shirin Saeedi Bidokhti

***Abstract*—A fundamental question in designing lossy data compression schemes is how well one can do in comparison with the rate-distortion function, which describes the known theoretical limits of lossy compression. Motivated by the empirical success of deep neural network (DNN) compressors on large, real-world data, we investigate methods to estimate the rate-distortion function on such data, which would allow comparison of DNN compressors with optimality. While one could use the empirical distribution of the data and apply the Blahut-Arimoto algorithm, this approach presents several computational challenges and inaccuracies when the datasets are large and high-dimensional, such as the case of modern image datasets. Instead, we reformulate the rate-distortion objective, and solve the resulting functional optimization problem using neural networks. We apply the resulting rate-distortion estimator, called NERD, on popular image datasets, and provide evidence that NERD can accurately estimate the rate-distortion function. Using our estimate, we show that the rate-distortion achievable by DNN compressors are within several bits of the rate-distortion function for real-world datasets. Additionally, NERD provides access to the rate-distortion achieving channel, as well as samples from its output marginal. Therefore, using recent results in reverse channel coding, we describe how NERD can be used to construct an operational one-shot lossy compression scheme with guarantees on the achievable rate and distortion. Experimental results demonstrate competitive performance with DNN compressors.**

***Index Terms*—Generative models, lossy compression, neural compression, rate-distortion theory, reverse channel coding.**

I. Introduction

DRIVEN by advances in deep neural network (DNN) compression schemes, rapid progress has been made in finding high-performing lossy compression schemes for large, high-dimensional datasets that remain practical [1], [2], [3], [4]. While these methods have empirically shown to outperform classical compression schemes for real-world data (e.g., images), it remains unknown as to how well they perform in comparison to the fundamental limit, which is given by the rate-distortion function. To investigate this question, one approach is to examine a stylized data source with a known probability distribution that is analytically tractable, such as the sawbridge random process, as done in [5]. This allows for a closed-form solution of the rate-distortion function; one can then compare it with empirically achievable rate and distortion of DNN compressors trained on realizations of the source. However, this approach does not evaluate DNN compressors on true sources of interest, such as real-world images, for which architectural choices such as convolutional layers have been engineered [6]. Thus, evaluating the rate-distortion function on these sources is paramount to understanding the efficacy of DNN compressors on real-world data.

Furthermore, a class of information-theoretically designed one-shot lossy source codes with near-optimal rate-distortion guarantees, which fall under the area of reverse channel coding [7], [8], [9], [10], [11], [12], [13], [14], [15], can provide a one-shot benchmark for DNN compressors, which are typically one-shot. However, these schemes require the rate-distortion-achieving conditional distribution (see (1)), which is generally intractable for real-world data, especially when the data distribution is unknown and only samples are available. Having the ability to recover the rate-distortion function's optimizing conditional distribution only from samples, in addition to the rate-distortion function itself, would allow for implementation of reverse channel codes even without access to the full data distribution.

Consider an independent and identically-distributed (i.i.d.) data source $X \sim P_X$, where P_X is a probability distribution supported on alphabet $\mathcal{X}$. Let $\mathcal{Y}$ be the reproduction alphabet, and $\mathsf{d} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$ be a distortion function on the input and output alphabets. The asymptotic limit on the minimum number of bits required to achieve a distortion D is given by the rate-distortion function [16], [17], [18], defined as

$$R(D) := \inf_{P_{Y|X}:\, \mathbb{E}_{P_{X,Y}}[\mathsf{d}(X,Y)] \le D} I(X;Y), \tag{1}$$

Any rate-distortion pair (R, D) satisfying $R > R(D)$ is achievable by some lossy source code, and no code can achieve a rate-distortion less than $R(D)$. It is important to note that $R(D)$ is achievable only under asymptotic blocklengths, whereas DNN compressors are typically one-shot, as compressing i.i.d. blocks for real-world datasets may not be feasible. However, the one-shot achievable region is known to be within $\log(R(D) + 1) + O(1)$ bits of $R(D)$ [7], and thus even in the one-shot setting, $R(D)$ remains an appropriate measure of the fundamental limits.

Manuscript received 1 April 2022; revised 6 October 2022, 31 January 2023, and 3 April 2023; accepted 1 May 2023. Date of publication 12 May 2023; date of current version 13 June 2023. This work was supported in part by the AI Institute for Learning-Enabled Optimization at Scale (TILOS) and in part by the NSF Award under Grant CCF-2112665. The work of Eric Lei was supported by the NSF Graduate Research Fellowship. The work of Hamed Hassani was supported by the NSF Award under Grant CIF-1943064. The work of Shirin Saeedi Bidokhti was supported in part by NSF Award under Grant 1910594, and in part by the NSF CAREER Award under Grant 2047482. *(Corresponding author: Eric Lei.)*

The authors are with the Department of Electrical and Systems Engineering, University of Pennsylvania, Phialdelphia, PA 19104 USA (e-mail: elei@seas.upenn.edu; hassani@seas.upenn.edu; saeedi@seas.upenn.edu).

Digital Object Identifier 10.1109/JSAIT.2023.3273467

There are several immediate challenges when computing $R(D)$ (and its optimizing conditional distribution) for large-scale data. Even when the distribution of P_X is known, the analytical form of the rate-distortion function has been difficult to evaluate, and has been characterized only on specific sources. This prohibits an analytical derivation using a density estimate of P_X (which are also not sample-efficient in high dimensions) in most cases. Computational methods such as the Blahut-Arimoto (BA) algorithm seem to be better suited for our setting; however, as will be shown, BA provides inaccurate estimates and fails to scale with large datasets.

A. Blahut-Arimoto Fails to Scale

Let $D_{\mathsf{KL}}(\mu||\nu)$ be the Kullback-Leibler (KL) divergence, defined as $\mathbb{E}_\mu[\log \frac{d\mu}{d\nu}]$ when the density $\frac{d\mu}{d\nu}$ exists and $+\infty$ otherwise. Due to the convex and strictly decreasing properties [18] of $R(D)$, it suffices to fix some $\beta > 0$, and solve the following double optimization problem.

Lemma 1 (Double-Minimization Form, cf. [18, ch. 10], [19]): The minimizers $P_{Y|X}^{(\beta)}$, $Q_Y^{(\beta)}$ of

$$\mathsf{RD}(\beta) := \inf_{Q_Y} \inf_{P_{Y|X}} D_{\mathsf{KL}}\big(P_{X,Y}||P_X \otimes Q_Y\big) + \beta \mathop{\mathbb{E}}_{P_{X,Y}}\big[\mathsf{d}(X,Y)\big], \tag{2}$$

yield a unique point $R_\beta = D_{\mathsf{KL}}(P_X P_{Y|X}^{(\beta)}||P_X \otimes Q_Y^{(\beta)})$ and $D_\beta = \mathbb{E}_{P_X P_{Y|X}^{(\beta)}}[\mathsf{d}(X,Y)]$ on the positive-rate regime of the rate-distortion curve, i.e., $R(D_\beta) = R_\beta$, such that $D_\beta < D_{\max}$ where $R(D_{\max}) = 0$.

The Blahut-Arimoto (BA) solves (2) by alternating steps on $P_{Y|X}$ and Q_Y until convergence. In discrete settings, the optimizers take the following closed form:

$$p(y|x) = \frac{r(y)e^{-\beta\,\mathsf{d}(x,y)}}{\sum_{\tilde{y}\in\mathcal{Y}} r(\tilde{y})e^{-\beta\,\mathsf{d}(x,\tilde{y})}}, \quad \forall x \in \mathcal{X}, y \in \mathcal{Y} \tag{3}$$

$$r(y) = \sum_{x\in\mathcal{X}} p_X(x)p(y|x), \quad \forall y \in \mathcal{Y} \tag{4}$$

Even though BA requires knowledge of the source distribution, one can use the empirical distribution $\hat{P}_n = \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$ as a proxy. This, however, does not scale in the case of modern datasets. Consider the setting when $\mathcal{X} = \mathbb{R}^m$ is continuous. Applying BA requires discretization of the input and output alphabets. In many cases, this would require acute knowledge of how to discretize $\mathbb{R}^m$ to form an appropriate reconstruction alphabet $\mathcal{Y}$, and even if one could, it might result in computational complexity that grows, potentially exponentially, with m. One would need to store a $n \times |\mathcal{Y}|$ matrix for the conditional PMFs and a $|\mathcal{Y}|$-sized vector for the output marginal PMF, which may not fit in memory depending on the number of data points or the choice of discretization. For example, in image compression, where we assume each $X_i \in \mathbb{R}^m$ to be a single image realization, $\mathcal{Y} \subseteq \mathbb{R}^m$. Even for 8-bit grayscale images, full precision quantization would require $2^8 \cdot m$ points, and although one could provide better discretization schemes, they may still require an intractable number of points.

To demonstrate, we attempt to apply discretized BA to MNIST digits in Fig. 1, and plot its estimated curve in comparison to rate-distortion (with squared-error distortion) achieved by DNN compressors. Specifically, our source is the empirical MNIST distribution $\hat{P}_n$, and we discretize $\mathcal{Y}$ to be exactly the support of our data, i.e., $\mathcal{Y} = \bigcup_{i=1}^n \{X_i\}$. While this scheme should converge to the true rate-distortion function as $n \to \infty$ [20], we see that even for $n = 60,000$, this fails to capture the general trend of the DNN compressors. Finally, the distortion corresponding to $R = 0$, known as $D_{\max}$, that BA estimates is far from the optimal given by $\mathbb{E}_{P_X}\big[\|X - \mu_X\|_2^2\big]$ – see Section IV for more details. This showcases the inaccuracy of BA in estimating the rate-distortion function even with relatively large number of samples. In contrast, our method, which provides the estimate $\widehat{R_\Theta(D)}_n$, does not exhibit these failures and is able to generalize to the true MNIST distribution for a significant portion of the rate-distortion function.

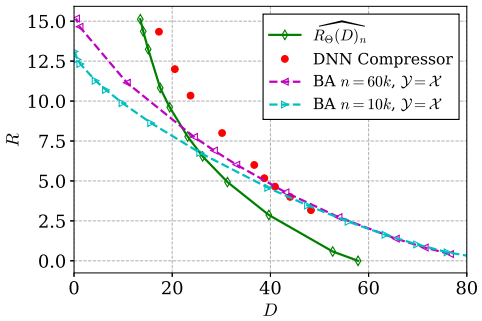


Fig. 1. Inaccuracy of discretized Blahut-Arimoto in comparison to our method, $\widehat{R_\Theta(D)}_n$, for computing the rate distortion curve on the MNIST dataset. DNN compressors provide codes that lie in the achievable region. See text for details.

B. Related Work

There have been several recent works that attempt to estimate $R(D)$ on real-world image data. Aside from the classical works of Arimoto [21] and Blahut [22], the first work that uses neural networks to estimate rate-distortion is [23], who parameterize the $Q_{Y|X}$ channel using restricted Boltzmann machines and study small synthetic sources. More recently, there have been works such as [24], [25], [26], in which the authors bound the rate-distortion function on real-world data. The authors of [24] provide sandwich bounds on $R(D)$, where the upper bound is variational and proved to be tight. In contrast, this work, which was independently developed around the same time, provides an estimate of $R(D)$ by replacing a class of distributions that $R(D)$ minimizes over with a parameterized set of distributions, leading to a natural upper bound. Additionally, the works in [8], [24], [27] discuss the potential of reverse channel coding applied to image compression; our work directly implements reverse channel coding using the rate-distortion achieving $Q_{Y|X}$ channel learned from our $R(D)$ estimate. In [20], the authors analyze theoretical properties of the plug-in estimator for $R(D)$, but do not provide a method that can be applied to real-world datasets.

A related area of work lies in the generative modeling literature, where the rate-distortion trade-off is often used to evaluate generative models and unsupervised learning algorithms. The most relevant work is [28], where the authors take a rate-distortion perspective to evaluate the performance

of generative adversarial networks (GANs) and variational autoencoders (VAEs). In their formulation, they assume the trained generative model is the output $\mathcal{Y}$-marginal of the rate-distortion objective, and find an upper bound on the rate-distortion needed to reproduce the generative model, not the true rate-distortion function of the *source*. Much of the other work in this area [29], [30], [31] use variational bounds on the rate-distortion for the purposes of representation learning, and lack a direct connection to fundamental limits of lossy compression.

C. Contributions

As opposed to the aforementioned approaches in Section I, we take a step back and reformulate the rate-distortion objective into a min-max objective using duality, building on results from [32]. As will be shown, this alleviates many of the issues plaguing previous methods, and allows for practical implementation of lossy compression schemes based on reverse channel coding, solely from samples. Our contributions are as follows.

- We propose an estimator of $R(D)$ based on neural networks, called NERD, which we show is strongly consistent, and provide a corresponding algorithm to compute the estimate from samples for a broad class of distortion measures.
- We empirically show that these methods provide accurate estimates of $R(D)$ on synthetic as well as real-world datasets, and that DNN autoencoder compressors achieve a rate-distortion within a few bits of our estimate.
- We demonstrate how the optimal conditional distribution (or channel) of $R(D)$ can be approximately recovered from NERD, and applied to reverse channel coding schemes, which result in an operational one-shot lossy compression scheme.
- We experimentally show that on real-world data, this scheme performs competitively with DNN compressors while also satisfying guarantees on the achievable rate and distortion.
- We provide evidence that the gap between the one-shot DNN compressors and the estimated rate-distortion function could be minimized by using DNN compressors that perform block coding.

D. Notation

We use $\mathbb{E}[\cdot]$ and $\mathbb{P}(\cdot)$ to denote expectation and probability, respectively. In general, we use subscript letters to denote a probability measure's respective space, e.g., Q_Y for a distribution supported on $\mathcal{Y}$. The distribution P_X refers to the source (or data) distribution, supported on $\mathcal{X}$. For a measure μ, we use $f_*\mu$ to denote the pushforward measure of μ through a function f. We use $\otimes$ to denote product measures, e.g., $\mu \otimes \nu$. We assume logarithms to be taken base 2. $\mathsf{d}(\cdot,\cdot)$ represent a distortion measure, $D_{\mathsf{KL}}(\cdot||\cdot)$ is the Kullback-Leibler divergence, and $W_p(\cdot,\cdot)$ is the p-Wasserstein distance.

II. Problem Formulation

Our goal is to estimate the rate-distortion function $R(D)$ of some source P_X. However, we only have access to n samples $X_1, \dots, X_n$ drawn i.i.d. from P_X, and do not assume any other knowledge about its distribution.

As opposed to BA, which solves the double minimization problem (2) in closed form, we use a dual form of the rate-distortion function. We first note that the constrained version of the inner minimization problem in (2) is known as the *rate function* in the literature [20], [32], i.e.,

$$R(Q_Y, D) := \inf_{\substack{P_{Y|X}: \\ \mathbb{E}_{P_{X,Y}}[\mathsf{d}(X,Y)] \le D}} D_{\mathsf{KL}}\big(P_{X,Y} || P_X \otimes Q_Y\big), \quad (5)$$

which exhibits the following dual characterization.

Lemma 2 (Rate Function Duality, [32, Sec. 2]): The rate function can be equivalently expressed as follows.

$$R(Q_Y, D) = \sup_{\tilde{\beta} \le 0} \tilde{\beta} D - \mathbb{E}_{P_X}\Big[\log \mathbb{E}_{Q_Y}\Big[e^{\tilde{\beta}\,\mathsf{d}(X,Y)}\Big]\Big] \quad (6)$$

Therefore, $R(D)$ is equivalent to $\inf_{Q_Y} R(Q_Y, D)$ and can be expressed as a min-max problem,

$$R(D) = \inf_{Q_Y} \sup_{\tilde{\beta} \le 0} \tilde{\beta} D - \mathbb{E}_{P_X}\Big[\log \mathbb{E}_{Q_Y}\Big[e^{\tilde{\beta}\,\mathsf{d}(X,Y)}\Big]\Big], \quad (7)$$

which can be estimated from samples, since we can approximate expectations with empirical averages for both P_X, Q_Y independently. Furthermore, the inner problem is concave, scalar, and has a unique solution. To solve the inner max, first-order stationary conditions yield [32, Sec. 2.2]

$$D = \mathbb{E}_{P_X \otimes Q_Y}\left[\mathsf{d}(X,Y) \frac{\exp\big(\tilde{\beta}\,\mathsf{d}(X,Y)\big)}{\mathbb{E}_{Y' \sim Q_Y}\big[\exp\big(\tilde{\beta}\,\mathsf{d}(X,Y')\big)\big]}\right], \quad (8)$$

which can be solved for $\tilde{\beta}^*$ via the bisection method. In the next section, we use the dual formulation to derive an estimator that uses neural networks to parametrize Q_Y.

III. Neural Estimation of the Rate-Distortion Function

We propose to parametrize the output marginal distribution Q_Y using architectural choices similar to those used in the GAN literature [33]. Specifically, let P_Z be some simple base distribution over $\mathcal{Z}$ and let $G : \mathcal{Z} \to \mathbb{R}^m$ be a function belonging to a function class $\mathcal{G}$. Then, representing distributions Q_Y with the pushforward $G_* P_Z$, we can optimize over functions in $\mathcal{G}$, and arrive at

$$R_{\mathcal{G}}(D) := \inf_{G \in \mathcal{G}} \sup_{\tilde{\beta} \le 0} \tilde{\beta} D - \mathbb{E}_{P_X}\Big[\log \mathbb{E}_{P_Z}\, e^{\tilde{\beta}\,\mathsf{d}(X, G(Z))}\Big]. \quad (9)$$

The equivalence of this (under certain assumptions on P_Z) with $R(D)$ is justified in [34]. In practice, we only have access to samples $X_1, \dots, X_n$ drawn i.i.d. from P_X, and must estimate (9) from the empirical distribution $\hat{P}_X^{(n)} := \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$. Leveraging the expressive power of neural networks, we choose $\mathcal{G}$ to be the class of functions parametrized by neural networks, and arrive at the following estimator (NERD).

Definition 1 (Neural Estimator of the Rate-Distortion Function (NERD)): Let $\mathcal{G} := \{G_\theta\}_{\theta \in \Theta}$ be a class of functions

Algorithm 1 Neural Estimator of the Rate-Distortion Function (NERD)

Input: Distortion constraint D, batch size B, number of steps T, learning rate η

Initialize generator neural network $G_\theta : \mathcal{Z} \to \mathcal{Y}$

for $t = 1, 2, \ldots, T$ **do**

Sample $\{x_i\}_{i=1}^B \overset{\text{i.i.d.}}{\sim} P_X$

Sample $\{z_j\}_{j=1}^B \overset{\text{i.i.d.}}{\sim} P_Z$

Define $\kappa_{i,j}(\tilde{\beta}, \theta_t) := \exp\Big(\tilde{\beta}\, \mathsf{d}(x_i, G_{\theta_t}(z_j))\Big)$

Solve $D = \frac{1}{B}\sum_{i=1}^B \mathsf{d}(x_i, G(z_i)) \frac{B \cdot \kappa_{i,i}(\tilde{\beta}, \theta_t)}{\sum_{j=1}^B \kappa_{i,j}(\tilde{\beta}, \theta_t)}$ for $\tilde{\beta}^*$

$\theta_{t+1} \leftarrow \theta_t - \eta \nabla_\theta \Big(-\frac{1}{B}\sum_{i=1}^B \log \frac{1}{B}\sum_{j=1}^B \kappa_{i,j}(\tilde{\beta}^*, \theta_t) \Big)$

end for

parametrized by a neural network. NERD is given by

$$\widehat{R_\Theta(D)}_n := \inf_{\theta\in\Theta} \sup_{\tilde{\beta}\le 0} \tilde{\beta} D - \mathop{\mathbb{E}}_{\hat{P}_X^{(n)}}\left[\log \mathop{\mathbb{E}}_{P_Z}\left[e^{\tilde{\beta}\, \mathsf{d}(X, G_\theta(Z))}\right]\right] \tag{10}$$

$$= \inf_{\theta\in\Theta} \sup_{\tilde{\beta}\le 0} \tilde{\beta} D - \frac{1}{n}\sum_{i=1}^n \log \mathop{\mathbb{E}}_{P_Z}\left[e^{\tilde{\beta}\, \mathsf{d}(X_i, G_\theta(Z))}\right] \tag{11}$$

The next theorem shows that NERD is a strongly consistent estimator for the rate distortion function. The proof is provided in Appendix A.

Theorem 1 (Strong consistency of NERD): Suppose the alphabets are $\mathcal{X} = \mathcal{Y} = \mathbb{R}^m$, P_Z is supported on $\mathcal{Z} \in \mathbb{R}^l$, and P_Z is absolutely continuous with respect to Lebesgue measure. Also, suppose that the distortion measure d is L_{d}-Lipschitz in both arguments. Then the NERD estimator in (11) is a strongly consistent estimator of $R(D)$, i.e.,

$$\lim_{n\to\infty} \widehat{R_\Theta(D)}_n = R(D) \text{ almost surely.} \tag{12}$$

Note that while NERD satisfies strong consistency, this may not necessarily apply to settings encountered in practice. In practice, we use stochastic gradient descent to search over the function class Θ which may not necessarily find the minimum, and the expectation over P_Z is estimated using Monte-Carlo methods.

To use NERD, following (11), one can simply sample batches from P_X and P_Z, solve the inner max of (7) by solving (8) for $\tilde{\beta}^*$, and take a gradient step over the DNN parameters. The full algorithm is given in Algorithm 1. The code is available online.[1]

A. Numerical Estimation Challenges

Note that although NERD is strongly consistent, the method suffers from estimation inaccuracies for large rates, which mirror similar estimation challenges of mutual information from samples [35]. The issues stem from the log expectation over P_Z, $\log \mathbb{E}_{P_Z}\left[e^{\tilde{\beta}\, \mathsf{d}(X_i, G_\theta(Z))}\right]$, requiring at least 2^R samples to estimate accurately with a sample mean at rate R. To see this, note that the inner minimization in the min-max form in (7) is equivalent to the Donsker-Varadhan (DV) lower bound [36].

From [32], if $\tilde{\beta}^*$ is the inner problem's maximizer, then

$$\sup_{\tilde{\beta}\le 0} \tilde{\beta} D - \mathbb{E}_{P_X}\left[\log \mathbb{E}_{Q_Y}\left[e^{\tilde{\beta}\, \mathsf{d}(X,Y)}\right]\right] \tag{13}$$

$$= \mathop{\mathbb{E}}_{P_X}\left[\mathop{\mathbb{E}}_{Q^*_{Y|X}(\cdot|X)}\left[\tilde{\beta}^*\, \mathsf{d}(X, Y)\right] - \log \mathop{\mathbb{E}}_{Q_Y}\left[e^{\tilde{\beta}^*\, \mathsf{d}(X,Y)}\right]\right] \tag{14}$$

$$= \mathop{\mathbb{E}}_{P_X}\left[\sup_{f:\mathcal{Y}\to\mathbb{R}} \mathop{\mathbb{E}}_{Q^*_{Y|X}(\cdot|X)}\left[f(Y)\right] - \log \mathbb{E}_{Q_Y}\left[e^{f(Y)}\right]\right] \tag{15}$$

$$= \mathop{\mathbb{E}}_{P_X}\left[D_{\mathsf{KL}}\left(Q^*_{Y|X}(\cdot|X)||Q_Y\right)\right] \tag{16}$$

$$= I^*(X; Y) + D_{\mathsf{KL}}\left(\bar{Q}_Y||Q_Y\right), \tag{17}$$

where $\frac{dQ^*_{Y|X=x}}{dQ_Y}(x, y) = \frac{e^{\tilde{\beta}^*\, \mathsf{d}(x,y)}}{\mathbb{E}_{Y'\sim Q_Y}\left[e^{\tilde{\beta}^*\, \mathsf{d}(x,y)}\right]}$, the optimizing f is given by $f^*(y) = \tilde{\beta}^*\, \mathsf{d}(x, y)$, $I^*(X; Y)$ is the mutual information of the joint $P_X Q^*_{Y|X}$, and $\bar{Q}_Y$ is the Y-marginal[2] of the joint $P_X Q^*_{Y|X}$. Hence, if we use an empirical distribution for Q_Y, say with M samples, then

$$I^*(X; Y) + D_{\mathsf{KL}}\left(\bar{Q}_Y||Q_Y\right) \le H\left(\bar{Q}_Y\right) + D_{\mathsf{KL}}\left(\bar{Q}_Y||Q_Y\right) \tag{18}$$

$$= -\mathbb{E}_{\bar{Q}_Y}\left[\log dQ_Y(y)\right] \tag{19}$$

$$= -\mathbb{E}_{\bar{Q}_Y}\left[\log \frac{1}{M}\right] \tag{20}$$

$$= \log M \tag{21}$$

Additionally, the authors in [35] show that the DV lower bound as well as any other M-sample estimate of a distribution-free high-confidence lower bound of mutual information is at most $O(\log M)$. Therefore, we can expect NERD to estimate $R(D)$ for rates up to $\log_2 M$. In practice, we set $M = 40,000$ for 32×32 images.

On a separate note, the authors in [24] provide an empirical lower bound of $R(D)$ which can be estimated from samples. They note that the lower bound always struggles to achieve rates greater than $O(\log M)$. The above explanation on estimating lower bounds of mutual information may help explain that phenomenon. Since NERD provides an exact estimate of the mutual information of the rate-distortion achieving joint distribution, it is estimating the *tightest* lower bound.

IV. Experimental Results: NERD

In our experiments, we use synthetic data (i.e., Gaussian), MNIST digits, and Fashion MNIST (FMNIST) images to represent our source $X \sim P_X$. We use squared-error distortion for all cases, i.e., $\mathsf{d}(x, y) = \|x - y\|_2^2$. In all cases, we have $n = 60,000$ i.i.d. samples from P_X. We set the base distribution as $P_Z = \mathcal{N}(0, I_{m_z})$ and parametrize $G_\theta : \mathbb{R}^{m_z} \to \mathbb{R}^m$ with a fully connected neural network for the Gaussian case, and a deep convolutional architecture similar to the generator architecture used in DCGAN [37] for the image data. For the DNN compressors, we use the nonlinear transform coding framework outlined in [4]. More architectural details are provided in Appendix C.

[1] https://github.com/leieric/NERD-RCC

[2] Note that from the Blahut-Arimoto equations (3) and (4), the Y-marginal $\bar{Q}_Y$ coincides with Q_Y only at optimality, when they both equal the optimal reproduction distribution Q^*_Y. In other words, $P_X Q^*_{Y|X}$ is a coupling between P_X and Q_Y only at optimality.

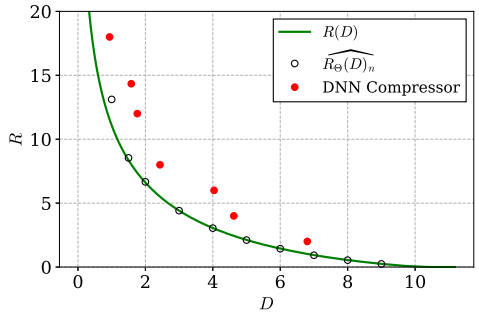


Fig. 2. Estimated $\widehat{R_\Theta(D)}_n$ (NERD) on Gaussian data ($m = 20$, $\sigma_k^2 = 4e^{-\frac{1}{16}k}$) compared with DNN compressors.

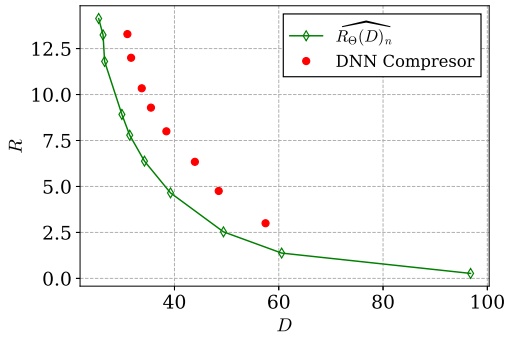


Fig. 3. Estimated $\widehat{R_\Theta(D)}_n$ (NERD) of SVHN images vs. DNN compressors.

A. Synthetic Data

We first verify the NERD estimator on Gaussian data. Specifically, $P_X = \mathcal{N}(0, \Sigma)$ is a multivariate Gaussian with $m = 20$ dimensions. Let $\Sigma = V\text{diag}(\sigma_1^2, \ldots, \sigma_m^2)V^\top$ be the eigendecomposition of Σ, with V orthogonal. In this case, the rate-distortion function of P_X has an analytical form and is given by [18]

$$R(D) = \begin{cases} \sum_{\substack{i\in[m]:\\ \sigma_i^2>\lambda}} \frac{1}{2}\log\frac{\sigma_i^2}{\lambda}, & D \le \sum_{i=1}^m \sigma_i^2 \\ 0, & \text{o.w.} \end{cases} \tag{22}$$

where $\lambda > 0$ satisfies the reverse water-filling condition

$$\lambda\left|\left\{i : \sigma_i^2 > \lambda\right\}\right| + \sum_{i\in[m]:\sigma_i^2\le\lambda} \sigma_i^2 = D. \tag{23}$$

To evaluate NERD, we choose $\sigma_k^2 = 4e^{-\frac{1}{16}k}$, and pick an orthogonal matrix V to form Σ. We generate samples from $\mathcal{N}(0, \Sigma)$ and apply Alg. 1. As shown in Fig. 2, for this choice of P_X, NERD can accurately estimate $R(D)$ for rates below 10 bits. For higher rates, the estimate suffers from numerical instability as discussed in the previous section. We also provide a second Gaussian example in the Appendix, shown in Fig. 15.

B. Real-World Data

We apply NERD to MNIST and FMNIST datasets, which are single-channel images, as well as SVHN (Street View House Numbers) [38] which contain color RGB images. We plot the estimated rate-distortion curves in Fig. 1 for MNIST, Fig. 16 for FMNIST, and Fig. 3 for SVHN. In all cases, the curve appears to satisfy the convex and strictly decreasing properties of the rate-distortion function. Using our estimate of $R(D)$, we can use DNN compressors of the autoencoder type [1], [2], [3] with quantization to see how they perform compared to the fundamental limits. Additionally, we see that DNN compressors closely follow the same trend as the estimated rate-distortion function, and are within several bits of optimality inside the achievable region. However, it remains difficult to conclude in this case whether or not DNN compressors are optimal on these datasets. The gap of several bits could be potentially attributed to the fact that the DNN compressors are one-shot, whereas $R(D)$ is achievable only under asymptotic blocklengths. While one-shot achievable regions of $R > R(D) + \log(R(D) + 1) + 5$ are known [7], lower bounds tighter than $R(D)$ for general sources P_X are not as clear. Either way, it remains to be seen whether other computationally feasible source codes could be designed to perform closer to the rate-distortion limit. In later sections, we discuss learning-based block codes on real-world data that empirically perform closer to $R(D)$ for certain regimes of the rate-distortion tradeoff.

C. Comparison With Blahut-Arimoto

We compare solving (11) to a baseline scheme that uses Blahut-Arimoto on discretized input and output alphabets. On low-dimensional input alphabets, Blahut-Arimoto will perform accurately. But for high-dimensional alphabets, it is unclear how to discretize the continuous space. For image datasets, which are high-dimensional, we have $\mathcal{X} = \{X_1, \ldots, X_n \in [0, 1]^m\}$ and let the source PMF be $\frac{1}{n}\sum_{i=1}^n \delta_{X_i}(x)$ and choose a discretization for $\mathcal{Y} \subseteq [0, 1]^m$ to define an output marginal PMF for Blahut-Arimoto. We attempt to choose the discretization for $\mathcal{Y}$ to be the same as the source, i.e., $\mathcal{Y} = \{X_1, \ldots, X_n \in [0, 1]^m\}$ is exactly the support of the samples. Such a scheme should converge to the true rate-distortion function as $n \to \infty$ assuming the true continuous alphabets are both $[0, 1]^m$. However, we demonstrate that Blahut-Arimoto fails when the number of samples is finite. Firstly, we are limited by the number of samples (60,000 at most with both datasets). Even with a large number of samples, we see that, given in Fig. 1 and 16, doing so does not work particularly well, and the trend is completely off compared to NERD and the DNN codes. It fails to extrapolate to the true rate-distortion function of the true source, and traces the rate-distortion curve for the discrete uniform empirical distribution which we see achieves zero distortion at $R = H(\hat{P}_X^{(n)}) = \log_2 n$. As n grows larger, we would expect the curve traced by this scheme to "rotate" clockwise to the true rate-distortion curve (which requires infinite rate at zero-distortion for continuous sources), but this scheme can only rotate to where the zero-distortion rate reaches $\log_2(60,000) \approx 15.87$. In contrast, NERD is able to follow the same trend of the operational rate-distortion curve estimated by DNN compressors, and matches known characteristics of $R(D)$ as described in the next section.

D. Comparison With Empirical Sandwich Bounds

This section provides comparisons with other neural network-based bounds on $R(D)$; namely, those provided

(a) $R = 0.28$ bits, $D = 54.95$.

(b) $R = 16.82$ bits, $D = 10$.

Fig. 4. Samples from trained Y-marginal Q_Y^* (MNIST). As $R \to 0$, Q_Y^* generates the mean image, achieving $D = D_{\max}$.

(a) $R = 1.37$ bits, $D = 60.55$.

(b) $R = 11.80$ bits, $D = 26.60$.

Fig. 5. Samples from trained Y-marginal Q_Y^* (SVHN). As $R \to 0$, Q_Y^* generates the mean image, achieving $D = D_{\max}$.

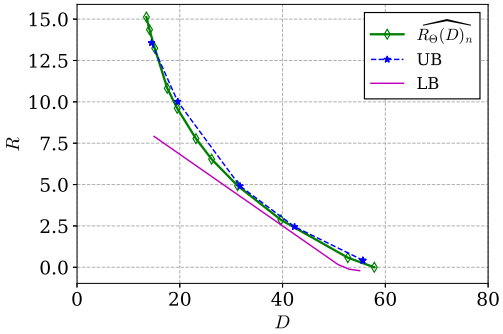


Fig. 6. NERD vs. empirical sandwich bounds [24] on MNIST.

in [24]. It is important to note that NERD itself is an upper bound of $R(D)$. This can be seen since the inner maximization simply computes the mutual information between Q_Y and $Q_{Y|X}$ in closed form. The outer minimization restricts the search over distributions Q_Y to those parameterized by the neural network function class. Thus, any fixed Q_Y parameterized by a neural network directly yields an upper bound of $R(D)$. In Fig. 6 we show how the sandwich bounds provided in [24] compare to NERD on the MNIST dataset. Indeed, it can be seen that on certain parts of the rate-distortion curve, NERD matches up with the upper bound from [24], indicating that the tightness of their bound is likely achieved for these rate points. For the lower bounds, there is a gap when compared to the NERD estimate. However, we encountered instability from the implementation provided from [24], when computing the lower bound. It is interesting to note that NERD's formulation is very similar to the lower bound [24, eq. (4)], which suggests two things. First, NERD chooses a simpler parameterization to solve $R(D)$. Second, our discussion from Section III-A explained why any lower bound of $R(D)$ requires sample complexity exponential in the rate, which corroborates the analysis and results of the lower bound in [24], who struggled to bring the lower bound above log of the number of samples used.

E. Samples From the Optimal Reproduction Distribution

We now illustrate generated samples from the (approximately) optimal reproduction distribution Q_Y^*, parametrized by the trained neural network G_{θ^*}, where θ^* neural network parameters that are the minimizers of NERD. In other words, $Q_Y^* = G_{\theta^**}P_Z$, the pushfoward of P_Z through G_{θ^*}. We show that it indeed aligns with the behavior of the rate-distortion function. Let $D_{\max} := \min_{y\in\mathcal{Y}} \mathbb{E}_{P_X}[\mathsf{d}(X, y)]$ be the distortion achievable at zero rate [19, ch. 9], i.e., $R(D_{\max}) = 0$. This is the best distortion that can be possibly achieved when there is no information about the source, where the reproduction is simply the best constant estimate of X. When d is squared-error, and $\mathcal{X} = \mathcal{Y}$, the best constant estimate is the mean $\mu_X = \mathbb{E}_{P_X}[X]$, with $D_{\max} = \mathbb{E}_{P_X}\left[\|X - \mu_X\|_2^2\right]$. In the MNIST case, the samples generated from the generator neural network at $R = 0.28$, shown in Fig. 4(a), consistently generate the "mean image". Computing $D_{\max}$ with an empirical average turns out to be ≈ 55 (under mild preprocessing of the MNIST dataset), which matches the zero-rate point in Fig. 1. In contrast, samples generated from a trained generator at higher rate, shown in Fig. 4(b), appear more similar to the original MNIST images and produce more modes of the distribution. A similar phenomenon occurs with SVHN and FMNIST as well, shown in Fig. 5 and 18. In comparison, Blahut-Arimoto does not produce this phenomenon and fails to intersect the D-axis at $D_{\max}$.

F. Generality of Distortion Function

In this section we demonstrate that the method reliably works for general distortion functions. In order for the back-propagation operation to function correctly, the only requirement is that the distortion function be differentiable in its arguments. To demonstrate this, we estimate the rate-distortion function of the same Gaussian source as described previously, but instead of squared-error distortion, we now apply general squared ℓ^p distances $\mathsf{d}_p(x, y) = \|x - y\|_p^2$. Due to the fact that $\|v\|_p \le \|v\|_q$ when $0 < q \le p < \infty$, we have that

$$\inf_{\substack{P_{Y|X}:\\ \mathbb{E}_{P_{X,Y}}[\mathsf{d}_p(X,Y)]\le D}} I(X;Y) \le \inf_{\substack{P_{Y|X}:\\ \mathbb{E}_{P_{X,Y}}[\mathsf{d}_q(X,Y)]\le D}} I(X;Y). \quad (24)$$

We show the rate-distortion function of the Gaussian source using $\mathsf{d}_1(x, y)$, $\mathsf{d}_2(x, y)$, and $\mathsf{d}_3(x, y)$. Since an analytical form of the Gaussian rate-distortion function is not known for d_1 and d_3, we only plot the NERD-estimated rate-distortion functions here. As observed in Fig. 7, the inequality in (24) is reflected in the 3 rate-distortion curves.

V. One-Shot Operational Lossy Source Codes

Now that we have the capability to calculate the fundamental limit of lossy source coding, and also recover an approximately optimal reproduction distribution Q_Y^* parametrized by a neural network, a natural question to ask is whether it is possible to use Q_Y^* to construct an operational compressor. Indeed, we will see in this section that Q_Y^* can be used to construct a compression scheme with guarantees on the achievable rate and distortion, and empirically perform similar to those of

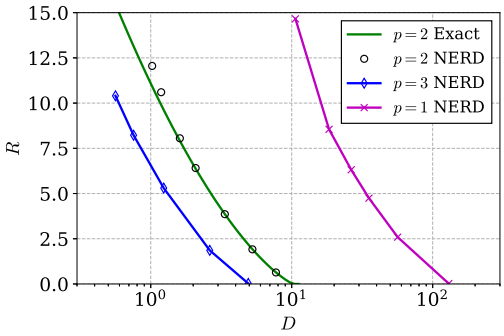


Fig. 7. Gaussian rate-distortion function for $\mathsf{d}_p(x,y) = \|x-y\|_p^2$ for $p = 1, 2, 3$.

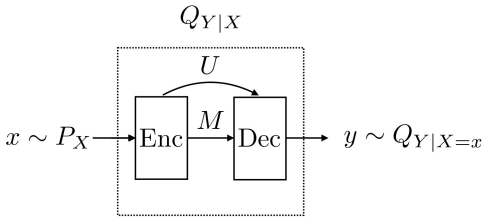


Fig. 8. Diagram of reverse channel coding, or channel simulation. Given $x \sim P_X$, the goal is to generate $y \sim Q_{Y|X=x}$, effectively simulating the channel $Q_{Y|X}$, by transmitting some information M with minimal description length $\mathbb{E}[L(M)]$. Common randomness U between the encoder and decoder is assumed.

DNN compressors. We first discuss reverse channel coding, which is the technique that underlies the lossy compression scheme, unlike DNN compressors which directly model the encoder and decoder with neural networks.

A. Reverse Channel Coding

The one-shot reverse channel coding (RCC) problem, described in [8], consists of reproducing a sample $x \sim P_X$ but allowing it to be corrupted by some noise. It is also known as the channel simulation problem and has been investigated previously in [7], [9], [10], [11], [12], [13], [14], [15] and references therein. Precisely, given a realization $x \sim P_X$, the sender wishes to reproduce a sample y at the receiver such that it follows a prespecified conditional distribution $Q_{Y|X=x}$, as if the realization x had gone through a channel $Q_{Y|X}$ (see Fig. 8). The sender communicates a message M which the decoder uses to reproduce y, with the goal of ensuring $y \sim Q_{Y|X=x}$ and that the expected description length $\mathbb{E}[L(M)]$ per source symbol, known as rate, communicated from the sender to receiver is minimal. In our setting, we assume that the sender and receiver share unlimited common randomness, denoted as U.

The relationship to lossy source coding is as follows. In lossy source coding, we wish to approximately represent some sample $x \sim P_X$ with y such that both expected description length and expected distortion are minimized. While RCC is not explicitly concerned with distortion, it is clear that if one has a RCC scheme for channel $Q_{Y|X}$ with rate $\mathbb{E}[L(M)] \le R$, then the expected distortion incurred is $\mathbb{E}_{P_X Q_{Y|X}}[\mathsf{d}(X,Y)]$. Hence, such a RCC scheme would yield a lossy one-shot code that achieves a rate of R and distortion $\mathbb{E}_{P_X Q_{Y|X}}[\mathsf{d}(X,Y)]$. Furthermore, suppose that $Q^*_{Y|X}$ is the channel that minimizes the rate-distortion function (1) at distortion level D. Any RCC scheme on $Q^*_{Y|X}$ achieves an expected distortion $\mathbb{E}_{P_X Q^*_{Y|X}}[\mathsf{d}(X,Y)] \le D$. What about the rate? It turns out that there are RCC schemes [7], [8] on $Q^*_{Y|X}$ that are guaranteed to achieve rates

$$\mathbb{E}[L(M)] \le R(D) + \log(R(D)+1) + 5, \tag{25}$$

which we now describe.

1) PFR: Suppose we have a channel $Q_{Y|X}$. The Poisson functional representation (PFR), proposed by Li and Gamal [7], is one such RCC scheme with guarantees on the rate required. In particular, in order to transmit a sample $x \sim P_X$, both the encoder and decoder (assuming shared common randomness U) first sample a marked Poisson process $\{(T_i, Y_i)\}_{i=1}^{\infty}$, such that $T_i - T_{i-1} \overset{\text{i.i.d.}}{\sim} \text{Exp}(1)$ and $Y_i \overset{\text{i.i.d.}}{\sim} Q_Y$, where Q_Y is the marginal of Y over the joint distribution $(X,Y) \sim P_X Q_{Y|X}$. The encoder then computes an index

$$K = \arg\min_{i \in \mathbb{N}} T_i \cdot \frac{dQ_Y}{dQ_{Y|X}(\cdot|x)}(Y_i), \tag{26}$$

and encodes it using a lossless source code. The decoder recovers K, and outputs Y_K, which is distributed with $Q_{Y|X=x}$. This scheme has the guarantee that

$$H(K) \le I(X;Y) + \log(I(X;Y)+1) + 4. \tag{27}$$

In practice, one can encode K using a Huffman codebook designed for the Zipf distribution with PMF $q(k) \propto k^{-(1+1/(I(X;Y)+e^{-1}\log e+1))}$, which guarantees a rate equivalent to the rate in (27) plus 1 bit [7]. Applying PFR to the rate-distortion optimal channel $Q^*_{Y|X}$ thus achieves the rate guarantee in (25).

One potential issue in the practical implementation of the PFR is solving (26), which requires solving a discrete and infinite optimization problem. To practically implement this, one must use a finite number of samples, but as described in [8], the guarantee on the rate in (27) does not necessarily hold when a finite number of samples is used.

2) ORC: To alleviate this issue, Theis and Yosri [8] propose ordered random coding (ORC). Rather than weighting the density ratios in (26) by Poisson arrival times, ORC weights them with sorted exponential random variables. The two RCC methods, PFR and ORC, can be summarized as follows [8, Th. 3]. Fix some number of samples N. Sample $\{X_i\}_{i=1}^N \overset{\text{i.i.d.}}{\sim} \text{Exp}(1)$, and $\{Y_i\}_{i=1}^N \overset{\text{i.i.d.}}{\sim} Q_Y$, where Q_Y is defined as above. Then, the encoder generates the cumulative weights $\{W_i\}_{i=1}^N$, defined to be

$$W_i = \begin{cases} \sum_{j=1}^i X_j, & \text{if PFR} \\ \sum_{j=1}^i \frac{N}{N-j+1} X_j, & \text{if ORC} \end{cases} \tag{28}$$

The encoder sends

$$K = \arg\min_{1 \le i \le N} W_i \cdot \frac{dQ_Y}{dQ_{Y|X}(\cdot|x)}(Y_i), \tag{29}$$

and the decoder outputs Y_K. In contrast with PFR, ORC can be shown to achieve the same rate as PFR (given in (27)) [8, Corollory 1], but the rate guarantee still holds for some

finite N [8, Th. 3]. This guarantees the rate in (27) even in practical usage, and so applying ORC to $Q^*_{Y|X}$ will again achieve the rate in (25).

B. Lossy Compression Scheme via Reverse Channel Coding

For any expected distortion tolerance D, let $Q^*_{Y|X}$ be the rate-distortion achieving channel, and Q^*_Y be the rate-distortion achieving reproduction distribution (simply the Y-marginal of the joint $Q^*_{Y|X}P_X$). Applying PFR or ORC to $Q^*_{Y|X}$ achieves a rate no more than $R(D) + \log(R(D) + 1) + 5$ and expected distortion no more than D. The following proposition exemplifies a nice interpretation when the rate-distortion achieving distributions are used for RCC.

Proposition 1: When using $Q^*_{Y|X}$ and Q^*_Y for RCC to compress $x \sim P_X$, the encoder computes the index

$$K = \arg\min_i \left\{ \mathsf{d}(x, Y_i) - \left(\beta^*\right)^{-1} \ln W_i \right\} \tag{30}$$

where $\beta^* < 0$ is the slope of $R(D)$ at the point $D = \mathbb{E}_{P_X Q^*_{Y|X}}[\mathsf{d}(X, Y)]$, and W_i's are generated via (28).

Remark 1: Prop. 1 can be interpreted as the encoder searching for the sample Y_i that is closest to x under distortion measure d, while simultaneously regularizing the size of the index used (and therefore its entropy). The amount of regularization is proportional to the tradeoff between rate and distortion in the rate-distortion function $R(D)$.

Proof: We have that $K = \arg\min_i W_i \frac{dQ^*_Y}{dQ^*_{Y|X}(\cdot|x)}(Y_i)$. Let $P_{X,Y} = Q^*_{Y|X}P_X$ be the joint measure. The optimal density ratio is then given by [32] as

$$\frac{dQ^*_{Y|X}(\cdot|x)}{dQ^*_Y}(y) = \frac{dP_{X,Y}}{dQ^*_Y dP_X}(y) \tag{31}$$

$$= \frac{e^{\beta^* \mathsf{d}(x,y)}}{\mathbb{E}_{Y' \sim Q^*_Y}\left[e^{\beta^* \mathsf{d}(x,Y')}\right]} \tag{32}$$

where $\beta^* = \arg\max_{\tilde{\beta} \le 0} \tilde{\beta} D - \mathbb{E}_{P_X}\left[\log \mathbb{E}_{Q^*_Y}\left[e^{\tilde{\beta}\, \mathsf{d}(X,Y)}\right]\right]$. Hence, using (29), we have that

$$K = \arg\min_i W_i \frac{dQ^*_Y}{dQ^*_{Y|X}(\cdot|x)}(Y_i) \tag{33}$$

$$= \arg\min_i W_i \frac{\mathbb{E}_{Y' \sim Q^*_Y}\left[e^{\beta^* \mathsf{d}(x,Y')}\right]}{e^{\beta^* \mathsf{d}(x,Y_i)}} \tag{34}$$

$$= \arg\min_i \frac{W_i}{e^{\beta^* \mathsf{d}(x,Y_i)}} = \arg\min_i \ln \frac{W_i}{e^{\beta^* \mathsf{d}(x,Y_i)}} \tag{35}$$

$$= \arg\min_i \left\{ \ln W_i - \beta^* \mathsf{d}(x, Y_i) \right\} \tag{36}$$

$$= \arg\min_i \left\{ \mathsf{d}(x, Y_i) - \left(\beta^*\right)^{-1} \ln W_i \right\} \tag{37}$$

where in the last step we have rescaled by $-\frac{1}{\beta^*} \ge 0$. ∎

Therefore, given n samples from P_X, one can design a lossy one-shot code with (approximately) rate (25) and distortion D as follows:

1) Apply NERD to the samples from P_X, which returns a trained neural network G_{θ^*} and slope β^* as solutions of the min-max problem in (11). Form an approximation to Q^*_Y as the pushforward of P_Z through G_{θ^*}, i.e., $G_{\theta^*}{}_*P_Z$. Sampling from this distribution can be done by first sampling $Z \sim P_Z$, and outputting $G_{\theta^*}(Z)$.
2) To compress $x \sim P_X$, apply Alg. 2 with $G_{\theta^*}{}_*P_Z$, slope parameter β^*, rate parameter as the estimate of the rate-distortion function $\widehat{R_\Theta(D)}_n$, and some random seed r. This returns a bit string b.
3) To decompress, apply Alg. 3 with the same parameters as the previous step.

We will refer to the above learned compression procedure as *NERD-RCC*.

Algorithm 2 Lossy Encoding Scheme via RCC

Input: Optimal $\mathcal{Y}$-marginal distribution Q^*_Y, rate-distortion slope parameter $\tilde{\beta} < 0$, rate parameter C, number of samples N, random seed r, sample to compress x
Generate weights $\{W_i\}_{i=1}^N$ according to (28)
Sample $\{Y_i\}_{i=1}^N \overset{\text{i.i.d.}}{\sim} Q^*_Y$ using r
Let $K = \arg\min_{i=1,\ldots,N}\{\mathsf{d}(x, Y_i) - \tilde{\beta}^{-1} \ln W_i\}$
Encode K using a Huffman code for the distribution with mass $q(k) \propto k^{-(1+1/(C+e^{-1}\log e+1))}$, into a bit string b
return b

Algorithm 3 Lossy Decoding Scheme via RCC

Input: Optimal $\mathcal{Y}$-marginal distribution Q^*_Y, rate parameter C, number of samples N, random seed r, bit string b
Decode K using the Huffman code for the distribution with mass $q(k) \propto k^{-(1+1/(C+e^{-1}\log e+1))}$
Sample $\{Y_i\}_{i=1}^N \overset{\text{i.i.d.}}{\sim} Q^*_Y$ using r
return Y_K

C. Experimental Results: NERD-RCC

We evaluate the RCC schemes on synthetic Gaussian, MNIST, and FMNIST data.

In the m-dimensional Gaussian case, we let the source be $P_X = \mathcal{N}(0, \mathrm{diag}(\sigma_1^2, \ldots, \sigma_m^2))$ where $\sigma_k^2 = 4e^{-\frac{1}{16}k^2}$. We can find the rate-distortion achieving channel $Q^*_{Y|X}$ and output marginal distribution Q^*_Y in closed form [18]. Let $A = \{k : \sigma_k^2 > \lambda\}$, with λ defined in (23). Then the channel and output marginal are both factorized as $Q^*_{Y|X} = \prod_{k=1}^m Q_{Y_k|X_k}$ and $Q^*_Y = \prod_{k=1}^m Q^*_{Y_k}$, where for $k \in A$, $Q_{Y_k|X_k=x} = \mathcal{N}(x, \lambda)$ and $Q_{Y_k} = \mathcal{N}(0, \sigma_k^2 + \lambda)$, and for $k \notin A$, Q_{Y_k} and $Q_{Y_k|X_k=x}$ assign probability 1 to 0. We compare the RCC schemes (implemented using these known Q^*_Y and $Q^*_{Y|X}$) with a one-shot DNN compressor trained on realizations from P_X, and show the results in Fig. 9. As can be seen, PFR and ORC achieve comparable rate-distortion to each other, and are several bits below the rate guarantee (25). DNN compressors, interestingly, are also several bits within (25) and seem to perform better than the RCC methods for lower rates but worse at higher rates. In Fig. 19, we show the same comparison with the rate-distortion achieved by RCC methods via the Q^*_Y learned from NERD. It can be seen that they closely mirror the performance of the RCC methods achieved using the exact Q^*_Y.

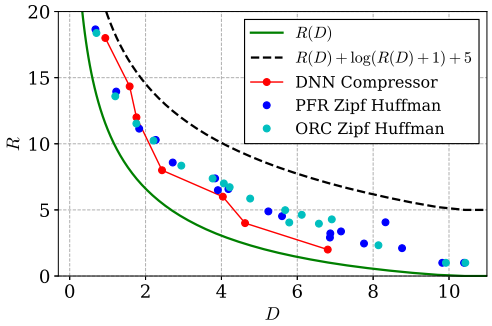


Fig. 9. Gaussian source.

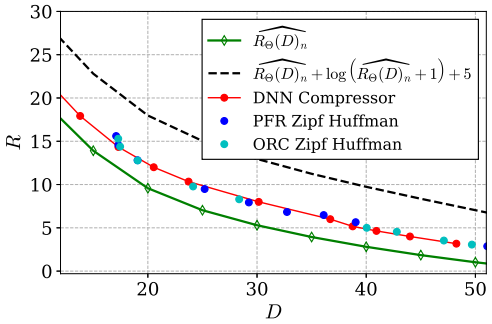


Fig. 10. PFR and ORC on MNIST images.

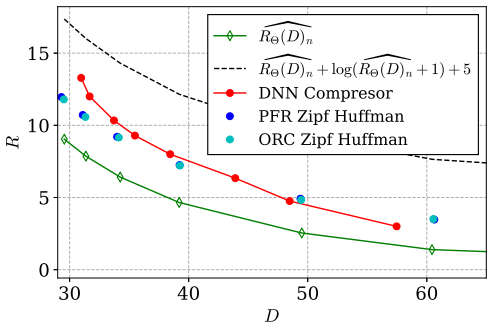


Fig. 11. PFR and ORC on SVHN images.

(a) Original MNIST images.

(b) Decompressed MNIST images via PFR.

Fig. 12. Visualization of Alg. 2, 3 on MNIST images.

The SVHN, MNIST, and FMNIST results are shown in Fig. 11, 10, and 17 respectively. For these datasets, we apply NERD-RCC by using the optimizers Q_Y^* and β^* from NERD since the rate-distortion optimizing channel and output marginal cannot be found in closed form. In both cases, NERD-RCC performs very similarly to the DNN one-shot code. As in the Gaussian case, both DNN compressors and NERD-RCC methods are within the rate guarantee of (25). In the Gaussian case in Fig. 9, we have access to the true rate-distortion optimizing channel and output marginal, so there is no potential for reduced performance at higher rates due to these factors. Nevertheless, the benefit of NERD-RCC is that the rate achieved at distortion D is always guaranteed to be within the upper bound $\widehat{R_\Theta(D)}_n + \log\left(\widehat{R_\Theta(D)}_n + 1\right) + 5$, no matter what dataset or source samples that NERD uses. In contrast, DNN compressors provide no such performance guarantee. In Fig. 11(b), we demonstrate several realizations of the source P_X and the output of NERD-RCC. As can be seen, the reverse channel coding scheme outputs a noisy, but faithful reconstruction of the original source.

VI. Discussion and Open Problems

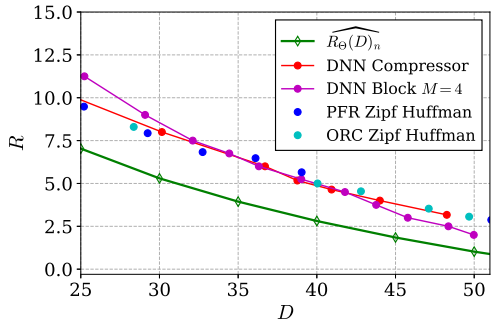
A. Lower Bounds and Block Coding for DNNs

The results in the previous two sections demonstrate that one-shot lossy DNN compressors are close to the rate-distortion function of real-world datasets, and competitive with RCC schemes. However, it is difficult to conclude whether or not DNN compressors are optimal, since a characterization of the one-shot fundamental limits are not as clear for general source distributions. While (25) provides an achievability result, lower bounds are not as clear for general sources. This precludes any definitive answer to optimality of one-shot DNN compressors for real-world datasets.

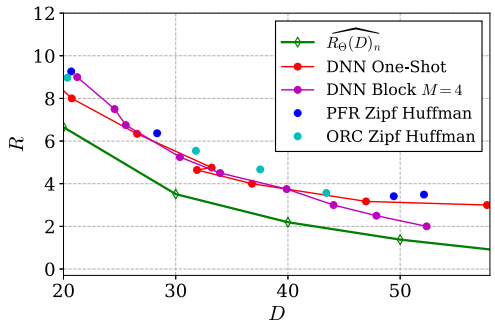
However, one might ask if perhaps this gap between the estimated rate-distortion function and rate-distortion achieved by the one-shot DNN compressors are due to the one-shot nature of the DNNs, and if DNN compressors that compress blocks of M realizations at a time can help close this gap. To test this, we use a DNN compressor with the same architecture as the one-shot DNN compressor, but apply it to blocks of $M = 4$ images combined together to form a larger image, in a 2×2 configuration. We plot the average rate and distortion per image sample in Figs. 13(a) and 13(b). As we can see, for both MNIST and FMNIST, the rate-distortion achieved is consistently better than one-shot DNN compressors as well as the RCC algorithms for smaller rates, demonstrating that block DNN codes on i.i.d. sources can indeed provide performance gains. At higher rates, however, the block DNN compressor performs worse. This can potentially be attributed to limitations of the DNN architecture used, which worked well with single images, but may not be the best for 4 images that have been stitched together. One avenue for future work is finding DNN architectures that work well with compressing blocks of images, and to see how they compare to the rate-distortion curves.

B. Computational Scaling of NERD and RCC

As noted in Section III, NERD requires a number of samples exponential in the rate required. While this is fine for many real-world datasets such as MNIST and SVHN, this limitation does not allow one to estimate large regimes of the rate-distortion function for "higher entropy" datasets such as ImageNet. Designing methods that can accurately estimate $R(D)$ at scale on such datasets is left for future work.



(a) MNIST.



(b) FMNIST.

Fig. 13. DNN block code versus RCC algorithms and DNN one-shot codes.

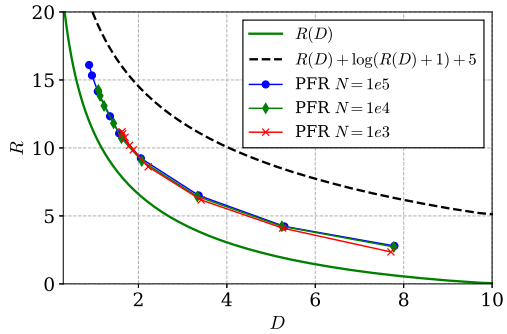


Fig. 14. Effect of N on RCC methods. Example here shows PFR on Gaussian source.

As shown in the previous section, RCC offers an alternative method for learning-based lossy source code design that does not explicitly use quantization, and has guarantees on the achievable rate and distortion which can be estimated directly from data using NERD. However, one drawback of this approach is that the sample complexity of many RCC algorithms is known to scale exponentially with the amount of information communicated from the sender to the receiver [8], [39]. Indeed, when running RCC with varying N, the number of samples generated, the estimated rates seem to scale logarithmic with N. An example of this is shown in Fig. 14. In comparison, DNN compressors do not suffer from such a runtime complexity. In order to scale to larger rates, one potential avenue is to adapt the recently proposed A* coding method in [39] for the similar relative entropy coding (REC) problem to the RCC setting. The authors of [39] use A* coding to achieve sample complexity linear in the rate rather than exponential. Applying these methods to the RCC schemes is left to future work.

VII. Conclusion

In this paper, we propose a new algorithm for computing the rate-distortion function for real-world data. We use an alternative formulation of the rate-distortion objective which is amenable to parametrization with neural networks and provide an estimator, NERD, that is sample and computationally efficient. We empirically show that it accurately estimates the rate-distortion function for synthetic and real-world datasets. We show how NERD can be used to implement near-optimal compression schemes via reverse channel coding on real-world data, with performance guarantees, and demonstrate the potential of DNN block codes to achieve rate-distortion performance closer to the rate-distortion function.

Appendix A
Proofs

Theorem 2 (Theorem 1 in Text): Suppose the alphabets are $\mathcal{X} = \mathcal{Y} = \mathbb{R}^m$, P_Z is supported on $\mathcal{Z} \in \mathbb{R}^l$, and P_Z is absolutely continuous with respect to Lebesgue measure. Also, suppose that the distortion measure d is L_{d}-Lipschitz in both arguments. Then the NERD estimator in (11) is a strongly consistent estimator of $R(D)$, i.e.,

$$\lim_{n\to\infty} \widehat{R_\Theta(D)}_n = R(D) \text{ almost surely.} \tag{38}$$

Proof: Fix $\epsilon > 0$. Define

$$\tilde{R}(Q,\beta) \coloneqq \beta D - \mathbb{E}_{P_X}\Big[\log \mathbb{E}_Q\Big[e^{\beta\, \mathsf{d}(X,Y)}\Big]\Big] \tag{39}$$

where $\beta \le 0$ is nonpositive, and

$$\tilde{R}(Q) \coloneqq \max_{\beta\le 0} R(Q,\beta) \tag{40}$$

since the inner sup has a unique maximum [32]. By definition $R(D) = \inf_{Q_Y} \tilde{R}(Q_Y)$, there exists Q_Y^* such that $\tilde{R}(Q_Y^*) < R(D) + \epsilon/4$. We would like to find a distribution $Q_\theta \sim G_\theta(Z)$, $Z \sim P_Z$, parametrized by a neural network G_θ, that has $\tilde{R}(Q_\theta)$ close to $\tilde{R}(Q_Y^*)$.

For fixed x, the function $y \mapsto e^{\beta\, \mathsf{d}(x,y)}$ is $L_{\mathsf{d}}|\beta|$-Lipschitz, since $\beta \le 0$. Hence, by Lemma 3,

$$\Big|\mathbb{E}_{Q_Y^*}\Big[e^{\beta d(x,Y)}\Big] - \mathbb{E}_{Q_\theta}\Big[e^{\beta d(x,Y)}\Big]\Big| \le L_{\mathsf{d}}|\beta| W_1\big(Q_\theta, Q_Y^*\big) \tag{41}$$

Let β^*, β_θ be the maximizers of $\tilde{R}(Q_Y^*)$ and $\tilde{R}(Q_\theta)$ respectively. Since log is a continuous function, there exists $\delta_1 > 0$ such that if $W_1(Q_\theta, Q_Y^*) < \frac{\delta_1}{L_{\mathsf{d}}|\beta^*|}$, then

$$\Big|\log \mathbb{E}_{Q_Y^*}\Big[e^{\beta^* d(x,Y)}\Big] - \log \mathbb{E}_{Q_\theta}\Big[e^{\beta^* d(x,Y)}\Big]\Big| < \epsilon/8 \tag{42}$$

Therefore, choosing Q_θ such that $W_1(Q_\theta, Q_Y^*) < \frac{\delta_1}{L_{\mathsf{d}}|\beta^*|}$ yields

$$\begin{aligned} &\big|\tilde{R}\big(Q_Y^*, \beta^*\big) - \tilde{R}\big(Q_\theta, \beta^*\big)\big| && (43)\\ &\le \mathbb{E}_{P_X}\Big[\Big|\log \mathbb{E}_{Q_Y^*}\Big[e^{\beta^* d(X,Y)}\Big] - \log \mathbb{E}_{Q_\theta}\Big[e^{\beta^* d(X,Y)}\Big]\Big|\Big] && (44)\\ &\le \mathbb{E}_{P_X}\big[\epsilon/8\big] = \epsilon/8 && (45)\end{aligned}$$

where the first inequality holds due to Jensen's inequality.

We now wish to bound $|\tilde{R}(Q_\theta, \beta^*) - \tilde{R}(Q_\theta, \beta_\theta)|$. Since the map $\beta \mapsto \tilde{R}(Q_\theta, \beta)$ is continuous in β, there exists $\delta_2 > 0$ such that if $|\beta^* - \beta_\theta| < \delta_2$, then $|\tilde{R}(Q_\theta, \beta^*) - \tilde{R}(Q_\theta, \beta_\theta)| < \epsilon/8$. Define

$$f(Q, \beta) := \mathbb{E}_{P_X}\left[\frac{\mathbb{E}_Q\left[\mathsf{d}(X, Y)e^{\beta\, \mathsf{d}(X,Y)}\right]}{\mathbb{E}_Q\left[e^{\beta\, \mathsf{d}(X,Y)}\right]}\right] - D \tag{46}$$

whose root provides the solution to $\tilde{R}(Q)$. We can apply Lemma 3 to the numerator and denominator inside the P_X expectation to conclude that with Q_θ satisfying $W_1(Q_Y^*, Q_\theta) < C$. Since the function $x_1, x_2 \mapsto \frac{x_1}{x_2}$ is continuous, we can conclude that $f(Q, \beta)$ is continuous with respect to the W_1 distance in Q since composition of continuous functions are continuous. Hence there is some $\delta_3 > 0$ such that if $W_1(Q_\theta, Q_Y^*) < \delta_3$, the maximizers β_θ and β^* must satisfy $|\beta^* - \beta_\theta| < \delta_2$, by the implicit function theorem.

We can thus find a neural network $G_{\theta'}$ with sufficient complexity[3] via [40] such that $Q_{\theta'}$ satisfies $W_1(Q_{\theta'}, Q_Y^*) < \min\left(\frac{\delta_1}{L_{\mathsf{d}}|\beta^*|}, \delta_3\right)$, yielding

$$\begin{aligned}&\left|\tilde{R}\left(Q_Y^*\right) - \tilde{R}(Q_{\theta'})\right| \\ &< \left|\tilde{R}\left(Q_Y^*, \beta^*\right) - \tilde{R}\left(Q_{\theta'}, \beta^*\right)\right| + \left|\tilde{R}\left(Q_{\theta'}, \beta^*\right) - \tilde{R}(Q_{\theta'}, \beta_{\theta'})\right| \\ &< \epsilon/4\end{aligned} \tag{47}$$

and hence $|R(D) - \tilde{R}(Q_{\theta'})| < \epsilon/2$. Since $\tilde{R}(Q_{\theta'})$ is an upper bound of $R(D)$, we have that

$$|R(D) - R_\Theta(D)| = \left|R(D) - \inf_{\theta\in\Theta} \tilde{R}(Q_\theta)\right| < \epsilon/2 \tag{48}$$

Applying the strong consistency result for the parametric problem in [20], $\exists N \in \mathbb{N}$ such that for all $n \geq N$, $|\widehat{R_\Theta(D)}_n - R_\Theta(D)| < \epsilon/2$ almost surely, and so again by triangle inequality, we have that $|\widehat{R_\Theta(D)}_n - R(D)| < \epsilon$ for all $n \geq N$, almost surely. ∎

Lemma 3: Let $g : \mathcal{X} \to \mathbb{R}$ be L-Lipschitz. Then for any distributions $P, Q \in \mathcal{P}(\mathcal{X})$,

$$\left|\mathbb{E}_{X\sim P}\left[g(X)\right] - \mathbb{E}_{X\sim Q}\left[g(X)\right]\right| \leq L \cdot W_1(P, Q) \tag{49}$$

where $W_1(P, Q) := \inf_{\pi\in\Pi(P,Q)} \mathbb{E}_{X,X'\sim\pi}\left[\|X - X'\|\right]$ is the 1-Wasserstein distance.

Proof: Let $g'(x) = \frac{g(x)}{L}$, so that g' is 1-Lipschitz.

$$\left|\mathbb{E}_{X\sim P}\left[g'(X)\right] - \mathbb{E}_{X\sim Q}\left[g'(X)\right]\right| \tag{50}$$

$$\leq \sup_{f:\|f\|_{\text{Lip}}\leq 1} \mathbb{E}_{X\sim P}\left[f(X)\right] - \mathbb{E}_{X\sim Q}\left[f(X)\right] \tag{51}$$

$$= W_1(P, Q) \tag{52}$$

where the last step is by Kantorovich-Rubinstein duality [41], [42] and $\|f\|_{\text{Lip}} := \sup_{\substack{x_1,x_2\in\mathcal{X}\\ x_1\neq x_2}} \frac{|f(x_2)-f(x_1)|}{\|x_1-x_2\|}$ is the Lipschitz norm of f. ∎

APPENDIX B
ADDITIONAL EXPERIMENTS

In this section, we provide additional experiments to support the main text.

[3]Assuming the function class Θ contains fully-connected networks with ReLU activations, with sufficient width and depth.

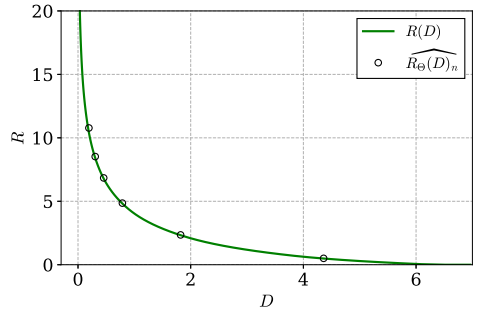


Fig. 15. Gaussian rate-distortion with 40 dimensions and $\sigma_k^2 = 4e^{-\frac{1}{4}k}$.

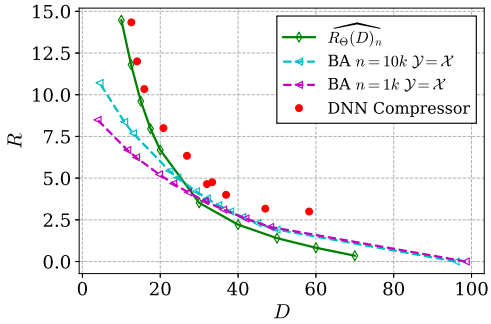


Fig. 16. Estimated $\widehat{R_\Theta(D)_n}$ (NERD) of FMNIST images vs. DNN compressors and BA.

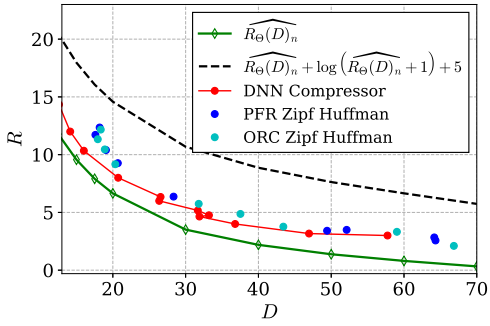
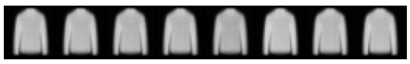


Fig. 17. PFR and ORC on FMNIST images.



(a) $R = 0.33$ bits, $D = 70$.

(b) $R = 3.54$ bits, $D = 30$.

Fig. 18. Samples from trained Y-marginal Q_Y^* (FMNIST). As $R \to 0$, Q_Y^* generates the mean image, achieving $D = D_{\max}$.

A. Additional Results on $R(D)$ Estimation and RCC

Fig. 15 provides an additional example of NERD on a different Gaussian source. Figs. 16, 17, and 18 provides FMNIST results. Fig. 19 provides NERD-estimated Q_Y^* in addition to exact Q_Y^* for RCC methods on the Gaussian source.

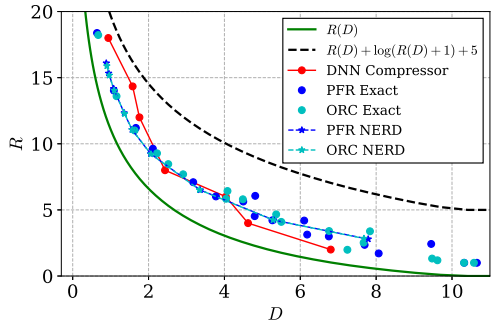


Fig. 19. RCC methods on Gaussian source with exact and NERD-estimated Q_Y^*.

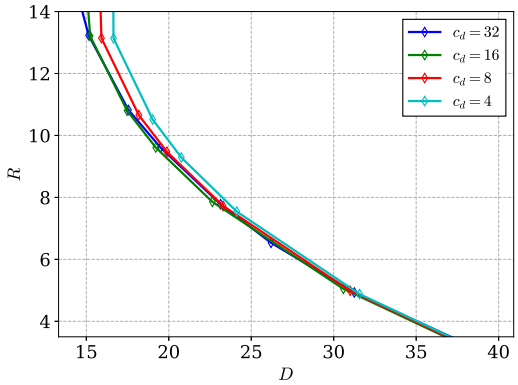


Fig. 20. Ablation study on generator capacity. Estimated rate-distortion curves via NERD for different generator sizes by varying c_d, which controls the number of channel dimensions throughout the model.

B. Ablation Study on Generator Capacity

In this section, we examine the effect of the generator capacity on the rate-distortion curves generated by NERD. In order to do so, we restrict capacity by reducing the number of channel dimensions in the hidden layers of the generator G. In particular, let c_d be the channel dimension parameter we adjust. The input tensors to the four convolution transpose layers of the generator architecture (further explained in Section C) are of size $8c_d, 4c_d, 2c_d$, and c_d. In the main text, all figures are generator using generators with $c_d = 32$. In this ablation study, we vary $c_d \in \{32, 16, 8, 4\}$, and show the results in Fig. 20. In all cases, the latent randomness passed to the generator remains at 128 dimensions (before projection to a $32c_d$-dimensional space). Thus a minimal c_d value of 4 ensures that the amount of latent randomness passed to the generator is not affected by the initial projection. We see that for small c_d (i.e., limited capacity), the rate-distortion estimates are higher at higher rates, and this decreases as c_d increases. The estimates seem to converge (at least for this rate range) for values of c_d at least 16.

APPENDIX C
IMPLEMENTATION DETAILS

A. NERD Generator Architecture

The generator G uses different architectures depending on the source type. For images, we use the DCGAN [37] architecture which is a fully convolutional architecture with 4 layers. It alternates between convolutional transpose layers with ReLU nonlinearities, with sigmoid nonlinearity at the output layer to scale between 0 and 1. Thus the total spatial upsampling is 2^4. The input to the model is a m_z-dimensional vector (with $m_z = 128$) which is first linearly projected to a $8c_d \times 2 \times 2$ space. This is reshaped to a tensor containing $8c_d$ channels and a 2×2 spatial resolution. Each layer increases the spatial dimension by a factor of 2 and decreases the channel dimension by a factor of 2, with the last channel dimension equalling 1 for grayscale images and 3 for RGB images. In the main text, all NERD estimates are given using $c_d = 32$.

For Gaussian sources, we us a fully-connected parameterization for G, with 2 hidden layers of dimension equally the dimensionality of the Gaussian source, and ReLU nonlinearities after each layer except for the last.

B. DNN Compressor Architecture

The DNN compressors used follow the nonlinear transform coding (NTC) framework [4]. In this paper, we use the soft-quantization approach where the latent elements are independently quantized. During training, a soft quantizer with a temperature parameter is used to back-propagate the gradients, with hard quantization at inference time. We found that the dithering-based quantization with factorized prior performed similarly on the image datasets. Similar to the NERD generator architectures, the encoder and decoder neural networks are fully convolution-based for the image datasets, and fully-connected for the synthetic sources.

REFERENCES

[1] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *Proc. 5th Int. Conf. Learn. Represent.*, Toulon, France, 2017.
[2] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," in *Proc. Int. Conf. Learn. Represent.*, 2017.
[3] E. Agustsson et al., "Soft-to-hard vector quantization for end-to-end learning compressible representations," in *Proc. NIPS*, 2017, pp. 1141–1151.
[4] J. Ballé et al., "Nonlinear transform coding," *IEEE Trans. Special Topics Signal Process.*, vol. 15, no. 2, pp. 339–353, Feb. 2021. [Online]. Available: https://arxiv.org/pdf/2007.03034.
[5] A. B. Wagner and J. Ballé, "Neural networks optimally compress the sawbridge," in *Proc. Data Compress. Conf. (DCC)*, 2021, pp. 143–152.
[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.
[7] C. T. Li and A. E. Gamal, "Strong functional representation lemma and applications to coding theorems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2017, pp. 589–593.
[8] L. Theis and N. Yosri, "Algorithms for the communication of samples," 2021, *arXiv:2110.12805*.
[9] P. Harsha, R. Jain, D. McAllester, and J. Radhakrishnan, "The communication complexity of correlation," *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 438–449, Jan. 2010.
[10] C. Bennett, P. Shor, J. Smolin, and A. Thapliyal, "Entanglement-assisted capacity of a quantum channel and the reverse Shannon theorem," *IEEE Trans. Inf. Theory*, vol. 48, no. 10, pp. 2637–2655, Oct. 2002.
[11] P. Cuff, "Distributed channel synthesis," *IEEE Trans. Inf. Theory*, vol. 59, no. 11, pp. 7071–7096, Nov. 2013.
[12] C. H. Bennett, I. Devetak, A. W. Harrow, P. W. Shor, and A. Winter, "The quantum reverse Shannon theorem and resource tradeoffs for simulating quantum channels," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2926–2959, May 2014.
[13] M. Braverman and A. Garg, "Public vs private coin in bounded-round information," in *Proc. 41st Int. Coll. Automata Lang. Program. (ICALP)*, 2014, pp. 502–513.

[14] P. Cuff, "Communication requirements for generating correlated random variables," in *Proc. IEEE Int. Symp. Inf. Theory*, 2008, pp. 1393–1397.
[15] C. T. Li and V. Anantharam, "A unified framework for one-shot achievability via the Poisson matching lemma," *IEEE Trans. Inf. Theory*, vol. 67, no. 5, pp. 2624–2651, May 2021.
[16] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948.
[17] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," in *Proc. IRE Nat. Conv. Rec.*, 1959, pp. 142–163.
[18] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. New York, NY, USA: Wiley, 2006.
[19] R. W. Yeung, *A First Course in Information Theory*. Dordrecht, The Netherlands: Kluwer Academic, 2002.
[20] M. T. Harrison and I. Kontoyiannis, "Estimation of the rate–distortion function," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3757–3762, Aug. 2008.
[21] S. Arimoto, "An algorithm for computing the capacity of arbitrary discrete memoryless channels," *IEEE Trans. Inf. Theory*, vol. IT-18, no. 1, pp. 14–20, Jan. 1972.
[22] R. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Trans. Inf. Theory*, vol. IT-18, no. 4, pp. 460–473, Jul. 1972.
[23] Q. Li and Y. Chen, "Rate distortion via deep learning," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 456–465, Jan. 2020.
[24] Y. Yang and S. Mandt, "Towards empirical sandwich bounds on the rate-distortion function," in *Proc. Int. Conf. Learn. Represent.*, 2022. [Online]. Available: https://openreview.net/forum?id=H4PmOqSZDY
[25] Y. Yang and S. Mandt, "Lower bounding rate-distortion from samples," in *Proc. Neural Compress. Inf. Theory Appl. Workshop (ICLR)*, 2021. [Online]. Available: https://openreview.net/forum?id=0CqQp8_6GH8
[26] J. Gibson, "Rate distortion functions and rate distortion function lower bounds for real-world sources," *Entropy*, vol. 19, no. 11, p. 604, 2017. [Online]. Available: https://www.mdpi.com/1099-4300/19/11/604
[27] G. Flamich, M. Havasi, and J. M. Hernández-Lobato, "Compressing images by encoding their latent representations with relative entropy coding," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 16131–16141.
[28] S. Huang, A. Makhzani, Y. Cao, and R. Grosse, "Evaluating lossy compression rates of deep generative models," in *Proc. 37th Int. Conf. Mach. Learn.*, Jul. 2020, pp. 4444–4454. [Online]. Available: https://proceedings.mlr.press/v119/huang20c.html
[29] C. P. Burgess et al., "Understanding disentangling in β-VAE," 2018, *arXiv:1804.03599*.
[30] R. Brekelmans, D. Moyer, A. Galstyan, and G. Ver Steeg, "Exact rate-distortion in autoencoders via echo noise," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 3884–3895.
[31] A. Alemi, B. Poole, I. Fischer, J. Dillon, R. A. Saurous, and K. Murphy, "Fixing a broken ELBO," in *Proc. 35th Int. Conf. Mach. Learn.*, Jul. 2018, pp. 159–168. [Online]. Available: https://proceedings.mlr.press/v80/alemi18a.html
[32] A. Dembo and L. Kontoyiannis, "Source coding, large deviations, and approximate pattern matching," *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 1590–1615, Jun. 2002.
[33] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.
[34] K. Rose, "A mapping approach to rate-distortion computation and analysis," *IEEE Trans. Inf. Theory*, vol. 40, no. 6, pp. 1939–1952, Nov. 1994.
[35] D. McAllester and K. Stratos, "Formal limitations on the measurement of mutual information," in *Proc. 23rd Int. Conf. Artif. Intell. Statist.*, Aug. 2020, pp. 875–884. [Online]. Available: https://proceedings.mlr.press/v108/mcallester20a.html
[36] M. D. Donsker and S. R. S. Varadhan, "Asymptotic evaluation of certain markov process expectations for large time. IV," *Commun. Pure Appl. Math.*, vol. 36, no. 2, pp. 183–212, 1983. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.3160360204
[37] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2016, *arXiv:1511.06434*.
[38] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, 2011. [Online]. Available: http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf
[39] G. Flamich, S. Markou, and J. M. Hernández-Lobato, "Fast relative entropy coding with A* coding," 2022, *arXiv:2201.12857*.
[40] Y. Lu and J. Lu, "A universal approximation theorem of deep neural networks for expressing probability distributions," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 3094–3105. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/2000f6325dfc4fc3201fc45ed01c7a5d-Paper.pdf
[41] L. Kantorovich and G. S. Rubinstein, "On a space of totally additive functions," *Vestnik Leningrad. Univ*, vol. 13, pp. 52–59, 1958.
[42] G. Peyré and M. Cuturi, "Computational optimal transport: With applications to data science," *Found. Trends Mach. Learn.*, vol. 11, nos. 5–6, pp. 355–607, 2019. [Online]. Available: http://dx.doi.org/10.1561/2200000073

Eric Lei (Graduate Student Member, IEEE) received the bachelor's degree in electrical and computer engineering from Cornell University. He is currently pursuing the Ph.D. degree with the Electrical and Systems Engineering Department, University of Pennsylvania, supported by a NSF Graduate Research Fellowship, under the supervision of S. S. Bidokhti and H. Hassani. His research interests are in neural compression, information theory, and deep generative models.

Hamed Hassani received the Ph.D. degree in computer and communication sciences from EPFL, Lausanne. He is currently an Assistant Professor of Electrical and Systems Engineering Department as well as the Computer and Information Sciences Department, and the Statistics Department with the University of Pennsylvania. Prior to that, he was a Research Fellow with Simons Institute for the Theory of Computing, UC Berkeley affiliated with the program of Foundations of Machine Learning, and a Postdoctoral Researcher with the Institute of Machine Learning, ETH Zürich. He is the recipient of the 2014 IEEE Information Theory Society Thomas M. Cover Dissertation Award, the 2015 IEEE International Symposium on Information Theory Student Paper Award, the 2017 Simons-Berkeley Fellowship, the 2018 NSF-CRII Research Initiative Award, the 2020 Air Force Office of Scientific Research Young Investigator Award, the 2020 National Science Foundation CAREER Award, the 2020 Intel Rising Star Award, and the 2023 IEEE Communications Society & Information Theory Society Joint Paper Award. He has also been selected as the Distinguished Lecturer of the IEEE Information Theory Society from 2022 to 2023.

Shirin Saeedi Bidokhti received the M.Sc. and Ph.D. degrees in computer and communication sciences from the Swiss Federal Institute of Technology. She is an Assistant Professor with the Department of Electrical and Systems Engineering as well as the Department of Computer and Information Science with the University of Pennsylvania (UPenn). Prior to joining UPenn, she was a Postdoctoral Scholar with Stanford University and the Technical University of Munich. She has also held short-term visiting positions with ETH Zürich, University of California at Los Angeles, and The Pennsylvania State University. Her research interests broadly include the design and analysis of network strategies that are scalable, practical, and efficient for use in Internet of Things applications, information transfer on networks, as well as data compression techniques for big data. She is a recipient of the 2023 IEEE Communications Society and Information Theory Society Joint Paper Award, the 2022 IT Society Goldsmith Lecturer Award, the 2021 NSF-CAREER Award, the 2019 NSF-CRII Research Initiative Award, and the prospective researcher and advanced postdoctoral fellowships from the Swiss National Science Foundation.