# IMPORTANCE WEIGHTED COMPRESSION

**Lucas Theis**[*]
Google Research
theis@google.com

**Jonathan Ho**[*]
Google Research
jonathanho@google.com

## ABSTRACT

The connection between variational autoencoders (VAEs) and compression is well established and they have been used for both lossless and lossy compression. Compared to VAEs, importance-weighted autoencoders (IWAEs) achieve a larger bound on the log-likelihood. However, it is not well understood whether a similar connection between IWAEs and compression exists and whether the improved loss corresponds to better compression performance. Here we show that the loss of IWAEs can indeed be interpreted as the cost of lossless or lossy compression schemes, and using IWAEs for compression can lead to small improvements in performance.

## 1 INTRODUCTION

*Variational autoencoders* (VAEs; Kingma and Welling, 2014; Rezende et al., 2014) are generative models of the form $p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x} \mid \mathbf{z})p(\mathbf{z})$ together with an approximate posterior $q(\mathbf{z} \mid \mathbf{x})$. Using *bits-back coding*, they can be used to losslessly compress data with an expected coding cost corresponding to the negative *evidence lower bound* (ELBO) of the VAE (Hinton and van Camp, 1993; Frey and Hinton, 1996; Townsend et al., 2018):

$$\underbrace{D_{\mathrm{KL}}[q(\mathbf{z} \mid \mathbf{x}) \,\|\, p(\mathbf{z})]}_{\text{Rate}} + \underbrace{\mathbb{E}_q[-\log p(\mathbf{x} \mid \mathbf{z})]}_{\text{Distortion}} = \underbrace{-\overbrace{\mathbb{E}_q\left[\log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z} \mid \mathbf{x})}\right]}^{\text{ELBO}}}_{\text{Coding cost}} \geq -\log p(\mathbf{x}). \tag{1}$$

In lossy compression, VAEs have been used to approximate rate-distortion trade-offs during training (Ballé et al., 2017; Theis et al., 2017). The relationship between the ELBO and a rate-distortion trade-off has further been shown to be exact for additive uniform noise (Agustsson and Theis, 2020).

In this paper we will draw connections between compression and *importance weighted autoencoders* (IWAEs; Burda et al., 2016). The loss of an IWAE is given by

$$\underbrace{\mathbb{E}_q\left[-\log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z} \mid \mathbf{x})}\right]}_{\text{VAE}} \geq \underbrace{\mathbb{E}_q\left[-\log \frac{1}{N}\sum_{n=1}^{N} \frac{p(\mathbf{x}, \mathbf{z}_n)}{q(\mathbf{z}_n \mid \mathbf{x})}\right]}_{\text{IWAE}} \geq -\log p(\mathbf{x}). \tag{2}$$

Here, the expectation is over $N$ independent samples from $q$. Given that the IWAE loss is less or equal that of a VAE, it is natural to ask whether the IWAE can lead to an improvement in compression performance. In the following, we answer this question positively by showing that the IWAE loss can be interpreted as the cost of a compression scheme, for both the lossless and lossy case.

## 2 RELATED WORK

The compression viewpoint on VAEs has yielded insights into VAE behavior. For example, posterior collapse is undesirable and a non-trivial property from a representation learning view (e.g., Phuong et al., 2018), but it is expected and even desired from a compression point of view since matching the prior is saving bits (Chen et al., 2017). The compression viewpoint on VAEs has also brought

[*]Equal contribution

insights from generative modeling to compression. For example, Yang et al. (2020) recently demonstrated improvements in lossy compression by applying various inference techniques. In particular, iterative inference normally used to reduce the amortization gap (Campos et al., 2019) was shown to also find better codes in a lossy neural compression scheme.

Similarly, our goal is to provide a compression viewpoint on IWAEs. This is in contrast to prior work that also reframes IWAEs but mainly studies them for their ability to better approximate the posterior over latents in a generative model (e.g., Bachman and Precup, 2015; Rainforth et al., 2018; Yang et al., 2020; Finke and Thiery, 2019; Lawson et al., 2019), not their ability to compress data.

Our work reflects the theme of tailoring coding algorithms for various types of generative models. Arithmetic coding (Rissanen, 1976) and asymmetric numeral systems (Duda, 2013) solve the coding problem for autoregressive models. Bits-back coding (Wallace and Boulton, 1968; Hinton and van Camp, 1993; Frey and Hinton, 1996) is suited for VAEs (Townsend et al., 2018; Kingma et al., 2019; Townsend et al., 2018) and has been adapted for flow models (Ho et al., 2019). Reverse channel coding techniques are also suitable for VAEs (Havasi et al., 2019; Flamich et al., 2020). In this paper, we follow this theme by proposing coding algorithms for IWAEs.

## 3 Lossless compression

**Bits-back coding** Bits-back coding attains a coding cost given by the negative ELBO of a VAE, provided access to a coding procedure for $p(\mathbf{x} \mid \mathbf{z})$, $p(\mathbf{z})$, and $q(\mathbf{z} \mid \mathbf{x})$. Assuming that one is willing to transmit extra random *auxiliary bits*, the sender follows these steps to encode $\mathbf{x}$:

1. Decode $\mathbf{z} \sim q(\mathbf{z} \mid \mathbf{x})$ from the source of auxiliary bits
2. Encode $\mathbf{x}$ using $p(\mathbf{x} \mid \mathbf{z})$
3. Encode $\mathbf{z}$ using $p(\mathbf{z})$

The receiver decodes $(\mathbf{z}, \mathbf{x})$ using $p(\mathbf{z})p(\mathbf{x} \mid \mathbf{z})$, then it re-encodes $\mathbf{z}$ using $q(\mathbf{z} \mid \mathbf{x})$. Re-encoding recovers the auxiliary bits that the sender used to sample $\mathbf{z}$. Not counting the auxiliary bits, the net number of bits needed to encode $\mathbf{x}$ is the negative ELBO of the VAE (e.g., Townsend et al., 2018). The coding cost savings due to the entropy of $q$ are only possible if auxiliary bits are available, for example if multiple data points are sent in sequence.

**Bits-back coding for importance weighted autoencoders** We now show how to apply bits-back coding to the IWAE. We first expand the IWAE objective:

$$\mathbb{E}\left[-\log \frac{1}{N}\sum_{n=1}^{N}\frac{p(\mathbf{x}, \mathbf{z}_n)}{q(\mathbf{z}_n \mid \mathbf{x})}\right] = \mathbb{E}\left[-\log \sum_{n} \frac{1}{N} p(\mathbf{x}, \mathbf{z}_n) \prod_{m \neq n} q(\mathbf{z}_m \mid \mathbf{x}) + \sum_{n} \log q(\mathbf{z}_n \mid \mathbf{x})\right] \quad (3)$$

This suggests a coding scheme where $-\sum_n \log q(\mathbf{z}_n \mid \mathbf{x})$ bits are saved by decoding $\mathbf{z}_{1:N}$ independently from auxiliary bits. We only have to show that we can encode $(\mathbf{x}, \mathbf{z}_{1:N})$ into

$$l(\mathbf{x}, \mathbf{z}_{1:N}) = -\log \sum_{n} \frac{1}{N} p(\mathbf{x}, \mathbf{z}_n) \prod_{m \neq n} q(\mathbf{z}_m \mid \mathbf{x}) \quad (4)$$

bits to achieve the IWAE objective. If we define the notation

$$P(n) = 1/N \qquad \text{and} \qquad \tilde{p}(\mathbf{x}, \mathbf{z}_{1:N} \mid n) = p(\mathbf{x} \mid \mathbf{z}_n) p(\mathbf{z}_n) \prod_{m \neq n} q(\mathbf{z}_m \mid \mathbf{x}), \quad (5)$$

then we see that

$$l(\mathbf{x}, \mathbf{z}_{1:N}) = -\log \sum_{n=1}^{N} P(n) \tilde{p}(\mathbf{x}, \mathbf{z}_{1:N} \mid n) \quad (6)$$

is the codelength of what appears to be a mixture model over $N$ components. Now we can introduce our idea: we can achieve this codelength by treating $n$ as a latent variable to be transmitted stochastically using bits-back. The optimal way of doing so is with the true posterior,

$$Q(n \mid \mathbf{x}, \mathbf{z}_{1:N}) = \frac{\tilde{p}(\mathbf{x}, \mathbf{z}_{1:N} \mid n)}{\sum_{m=1}^{N} \tilde{p}(\mathbf{x}, \mathbf{z}_{1:N} \mid m)} = \frac{p(\mathbf{x} \mid \mathbf{z}_n) p(\mathbf{z}_n) / q(\mathbf{z}_n \mid \mathbf{x})}{\sum_{m=1}^{N} p(\mathbf{x} \mid \mathbf{z}_m) p(\mathbf{z}_m) / q(\mathbf{z}_m \mid \mathbf{x})}, \quad (7)$$

which is tractable to calculate since $N$ is generally not too large. The complete bits-back coding scheme for the sender is as follows:

1. Decode $\mathbf{z}_1, \ldots, \mathbf{z}_N \sim q(\mathbf{z} \mid \mathbf{x})$. (Saves $\sum_n - \log q(\mathbf{z}_n \mid \mathbf{x})$ bits.)
2. Decode $n \sim Q(n \mid \mathbf{x}, \mathbf{z}_{1:N})$. (Saves $- \log Q(n \mid \mathbf{x}, \mathbf{z}_{1:N})$ bits.)
3. Encode $(\mathbf{x}, \mathbf{z}_{1:N})$ using $\tilde{p}(\mathbf{x}, \mathbf{z}_{1:N} \mid n) = p(\mathbf{x} \mid \mathbf{z}_n) p(\mathbf{z}_n) \prod_{m \neq n} q(\mathbf{z}_m \mid \mathbf{x})$:
   (a) Encode $\mathbf{z}_{1:N \setminus n}$ using $q(\mathbf{z} \mid \mathbf{x})$. (Costs $\sum_{m \neq n} - \log q(\mathbf{z}_m \mid \mathbf{x})$ bits.)
   (b) Encode $\mathbf{x}$ using $p(\mathbf{x} \mid \mathbf{z}_n)$. (Costs $- \log p(\mathbf{x} \mid \mathbf{z}_n)$ bits.)
   (c) Encode $\mathbf{z}_n$ using $p(\mathbf{z})$. (Costs $- \log p(\mathbf{z}_n)$ bits.)
4. Encode $n$ using $P(n)$. (Costs $- \log P(n) = \log N$ bits.)

The decoding steps save bits because the receiver is able to recover the auxiliary bits needed to sample the latents $(\mathbf{z}_1, \ldots, \mathbf{z}_N, n)$, and one can verify that summing the coding cost contributions above yields the IWAE objective. Interestingly, the first two steps of the sender resemble the sampling-importance-resampling algorithm of Cremer et al. (2017), who devised a method to modify a given VAE's approximate posterior to make its ELBO equal to or better than the IWAE objective.

Our coding procedure allows the user to bring the coding cost arbitrarily close to the theoretical optimum $- \log p(\mathbf{x})$ by increasing the number of IWAE samples $N$. The tradeoff is that increasing $N$ increases the number of auxiliary bits required to attain the IWAE objective, because auxiliary bits are needed to decode all of the $N$ latents. Thus our scheme may only be practical if $N$ is chosen adaptively over the course of sending a long sequence of data.

## 4 Lossy compression

**Softmin coding** Consider a lossy compression scheme where we first try to encode the same input multiple times. We do this either using different encoders or, if our encoder is stochastic, by running the same encoder multiple times. For each of the $N$ attempts, we can measure a rate-distortion loss:

$$\#_1 + \lambda d_1, \quad \#_2 + \lambda d_2, \quad \ldots, \quad \#_N + \lambda d_N.$$

In the following discussion it will be convenient to assume that the coding cost $\#_n$ is measured in *nats*. $d_n$ is the distortion achieved by the $n$-th code and $\lambda$ controls how much we care about distortion relative to the coding cost.

We could now chose to use the code with smallest rate-distortion cost. Importantly, we then also have to communicate to the decoder the index $n^*$ of the encoding used. If we uniformly encode $n^*$, then the additional cost is $\ln N$ nats. But in general it could be $- \ln P(n^*)$ for some probability distribution $P(n)$. To minimize the overall cost, we should pick

$$n^* = \operatorname{argmin}_n \left( \#_n + \lambda d_n - \ln P(n) \right). \tag{8}$$

Since we are free to choose $P(n)$, we could randomly generate the probabilities every time we encode data. As long as our probabilities do not depend on the data, we are able to generate them both on the encoder and the decoder side using, for example, a pseudorandom number generator with a common seed. We choose to generate the probabilities as follows:

$$P(n) = \frac{S_n^{-1}}{\sum_{m=1}^N S_m^{-1}}, \quad S_n \sim \text{Exp}(1). \tag{9}$$

For reasons that will become clear in a moment, we dub this randomized coding strategy *softmin coding*. Now consider the expected cost of doing softmin coding. Averaging over $S_n$, the expected rate-distortion loss is (Appendix A)

$$\mathbb{E}_P[\#_{n^*} + \lambda d_{n^*} - \ln P_{n^*}] = - \ln \tfrac{1}{N} \textstyle\sum_n \exp\left(-\#_n - \lambda d_n\right) + \psi_N. \tag{10}$$

Here, $\Psi_N$ is the extra cost incurred for encoding the data $N$ times instead of once (Fig. 1, left),

$$\psi_N = \mathbb{E}_S \left[ \ln \tfrac{1}{N} \textstyle\sum_n S_n^{-1} \right] - \gamma \geq 0, \tag{11}$$

and $\gamma$ is the Euler-Mascheroni constant. The left-hand term in Equation 10 is also called a soft-minimum, hence the name.

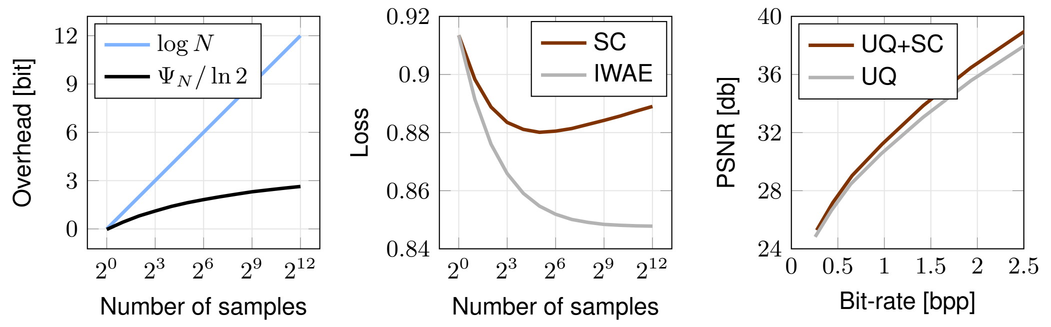

Figure 1: *Left:* Overhead of softmin coding compared to the overhead incurred when assigning uniform probabilities to the samples. *Middle:* Rate-distortion loss (Eq. 10, $\lambda = 0.003125$) of a linear model with (SC) and without (IWAE) taking into account the overhead $\Psi_N$. *Right:* Rate-distortion curves for a linear model using universal quantization with and without softmin coding with up to 4096 samples.

**Softmin coding for importance weighted autoencoders** Now consider applying softmin coding to *universally quantized neural compression* (Agustsson and Theis, 2020). Given some input $\mathbf{x}$, the encoder generates a uniformly distributed representation $\mathbf{z}$ using universal quantization (Ziv, 1985),

$$\mathbf{k} = \lfloor f(\mathbf{x}) - \mathbf{u}' \rceil, \quad \mathbf{z} = \mathbf{k} + \mathbf{u}', \tag{12}$$

where $\mathbf{u}' \sim U([-0.5, 0.5)^M)$ is a vector of independent uniform noise and the rounding function is applied pointwise. Importantly, $\mathbf{z}$ has the same uniform distribution as if we added uniform noise directly, $f(\mathbf{x}) + \mathbf{u}$ (Roberts, 1962; Schuchman, 1964). The uniform noise is assumed to be available on both the encoder and the decoder side. The encoder transmits $\mathbf{k}$ and the decoder reconstructs $\mathbf{z} = \mathbf{k} + \mathbf{u}$. The cost of encoding $\mathbf{k}$ is

$$-\ln p_z(\mathbf{k} + \mathbf{u}) = -\ln p_z(\mathbf{z})/q(\mathbf{z} \mid \mathbf{x}), \tag{13}$$

where $q(\mathbf{z} \mid \mathbf{x}) = 1$ if $\mathbf{z} - f(\mathbf{x}) \in [-0.5, 0.5)^M$ and $p_z(\mathbf{z})$ is an appropriately chosen model density.

Next, let us assume that the distortion is a cross-entropy, i.e. $\lambda d(\mathbf{x}, \mathbf{z}) = -\ln p(\mathbf{x} \mid \mathbf{z})$. If we now generate $N$ samples $\mathbf{z}_n$ and encode one of them using softmin encoding, the expected coding cost is

$$-\ln \frac{1}{N} \sum_n \frac{p(\mathbf{x}, \mathbf{z}_n)}{q(\mathbf{z}_n \mid \mathbf{x})} + \Psi_N \tag{14}$$

which is the loss of an importance weighted autoencoder up to a constant.

## 5 Discussion and results

We tested the feasibility of softmin coding in practice. Following Agustsson and Theis (2020), we trained a linear model with a factorial prior and a deep model with a hyperprior (Ballé et al., 2018). The linear model uses linear transforms to encode 8x8 image patches independently. Each encoder produces a stochastic encoding by adding uniform noise to the output of a neural network (more details are provided in Appendix C).

An important consideration is how much information to encode at once. Using softmin coding incurs an overhead which becomes less significant as we encode more information. On the other hand, picking 1 out of $N$ samples can communicate at most $\log N$ bits of extra information and so we should expect to need more samples to make a meaningful contribution for larger images.

Among other factors, the number of bits is controlled by the trade-off parameter $\lambda$ (Eq. 10) and the image size. We used the linear model to independently encode 8x8 patches of an image (which does not degrade its performance) and applied the deep model to 64x64 image patches (which degraded its performance on the Kodak dataset only slightly).

Results for the linear model are provided in Fig. 1. Note that unlike for the loss of IWAEs, the optimal number of samples with softmin coding is finite. We find that softmin coding slightly improves the performance of the linear model. For the deep model, results are provided in Appendix B. In this case the optimal number of samples is too large to be practically feasible. We found that using 4096 samples only slightly improved its performance.

### Acknowledgments

The authors would like to thank Eirikur Agustsson for help with setting up the experiments.

## References

E. Agustsson and L. Theis. Universally Quantized Neural Compression. In *Advances in Neural Information Processing Systems 33*, 2020.

P. Bachman and D. Precup. Training Deep Generative Models: Variations on a Theme. In *NIPS 2015 Workshop: Advances in Approximate Bayesian Inference*, 2015.

J. Ballé, V. Laparra, and E. P. Simoncelli. End-to-end Optimized Image Compression. In *International Conference on Learning Representations*, 2017.

J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston. Variational Image Compression with a Scale Hyperprior. In *International Conference on Learned Representations*, 2018.

J. Ballé, V. Laparra, and E. P. Simoncelli. Density Modeling of Images using a Generalized Normalization Transformation. In *International Conference on Learning Representations*, 2016.

Y. Burda, R. Grosse, and R. Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations*, 2016.

J. Campos, S. Meierhans, A. Djelouah, and C. Schroers. Content adaptive optimization for neural image compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel. Variational lossy autoencoder. In *International Conference on Learning Representations*, 2017.

C. Cremer, Q. Morris, and D. Duvenaud. Reinterpreting importance-weighted autoencoders. In *ICLR Workshop*, 2017.

J. Duda. Asymmetric numeral systems: entropy coding combining speed of huffman coding with compression rate of arithmetic coding. *arXiv preprint arXiv:1311.2540*, 2013.

A. Finke and A. H. Thiery. On importance-weighted autoencoders, 2019.

G. Flamich, M. Havasi, and J. M. Hernández-Lobato. Compressing images by encoding their latent representations with relative entropy coding. In *Advances in Neural Information Processing Systems 33*, 2020.

B. J. Frey and G. E. Hinton. Free energy coding. In *Proceedings of Data Compression Conference-DCC'96*, pages 73–81. IEEE, 1996.

M. Havasi, R. Peharz, and J. M. Hernández-Lobato. Minimal random code learning: Getting bits back from compressed model parameters. In *International Conference on Learning Representations*, 2019.

G. E. Hinton and D. van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13, 1993.

J. Ho, E. Lohn, and P. Abbeel. Compression with flows via local bits-back coding. In *Advances in Neural Information Processing Systems 32*, 2019.

D. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

F. H. Kingma, P. Abbeel, and J. Ho. Bit-swap: Recursive bits-back coding for lossless compression with hierarchical latent variables. In *International Conference on Machine Learning*, 2019.

J. Lawson, G. Tucker, B. Dai, and R. Ranganath. Energy-inspired models: Learning with sampler-induced distributions. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

D. Minnen, J. Ballé, and G. D. Toderici. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. In *Advances in Neural Information Processing Systems 31*, 2018.

M. Phuong, M. Welling, N. Kushman, R. Tomioka, and S. Nowozin. The mutual autoencoder: Controlling information in latent code representations, 2018.

T. Rainforth, A. Kosiorek, T. A. Le, C. Maddison, M. Igl, F. Wood, and Y. W. Teh. Tighter Variational Bounds are Not Necessarily Better. In *Proceedings of the 35 th International Conference on Machine Learning*, 2018.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

J. J. Rissanen. Generalized Kraft inequality and arithmetic coding. *IBM Journal of research and development*, 20(3):198–203, 1976.

L. G. Roberts. Picture Coding Using Pseudo-Random Noise. *IRE Transactions on Information Theory*, 1962.

L. Schuchman. Dither signals and their effect on quantization noise. *IEEE Transactions on Communication Technology*, 12(4):162–165, 1964.

L. Theis, W. Shi, A. Cunningham, and F. Huszár. Lossy Image Compression with Compressive Autoencoders. In *International Conference on Learning Representations*, 2017.

J. Townsend, T. Bird, and D. Barber. Practical lossless compression with latent variables using bits back coding. In *International Conference on Learning Representations*, 2018.

C. S. Wallace and D. M. Boulton. An information measure for classification. *The Computer Journal*, 11(2):185–194, 1968.

Y. Yang, R. Bamler, and S. Mandt. Improving Inference for Neural Image Compression. In *Advances in Neural Information Processing Systems 33*, 2020.

J. Ziv. On universal quantization. *IEEE Transactions on Information Theory*, 31(3):344–347, 1985.

## Appendix A

$$\begin{aligned}
\mathbb{E}_P[\#_{n^*} + \lambda d_{n^*} - \ln P_{n^*}] &= \mathbb{E}_P\left[\min_n \left(\#_n + \lambda d_n - \ln P_n\right)\right] && (15)\\
&= \mathbb{E}_S\left[\min_n \left(\#_n + \lambda d_n - \ln S_n^{-1}\right) + \ln \sum_n S_n^{-1}\right] && (16)\\
&= \mathbb{E}_G\left[\min_n \left(\#_n + \lambda d_n - G_n\right)\right] + \mathbb{E}_S\left[\ln \sum_n S_n^{-1}\right] && (17)\\
&= \mathbb{E}_G\left[-\max_n \left(-\#_n - \lambda d_n + G_n\right)\right] + \mathbb{E}_S\left[\ln \sum_n S_n^{-1}\right] && (18)\\
&= -\ln \sum_n \exp\left(-\#_n - \lambda d_n\right) - \gamma + \mathbb{E}_S\left[\ln \sum_n S_n^{-1}\right] && (19)\\
&= -\ln \frac{1}{N} \sum_n \exp\left(-\#_n - \lambda d_n\right) - \gamma + \mathbb{E}_S\left[\ln \frac{1}{N} \sum_n S_n^{-1}\right] && (20)\\
&= -\ln \frac{1}{N} \sum_n \exp\left(-\#_n - \lambda d_n\right) + \psi_N && (21)
\end{aligned}$$

Here, $\gamma$ is the Euler-Mascheroni constant, $G_n = \ln S_n^{-1} \sim \text{Gumbel}(0, 1)$ are Gumbel distributed random variables, and we used the fact that the maximum of identically scaled Gumbels is again Gumbel distributed,

$$\max_n(\mu_n + G_n) \sim \text{Gumbel}\left(\log \sum_n \exp(\mu_n), 1\right). \tag{22}$$

## Appendix B

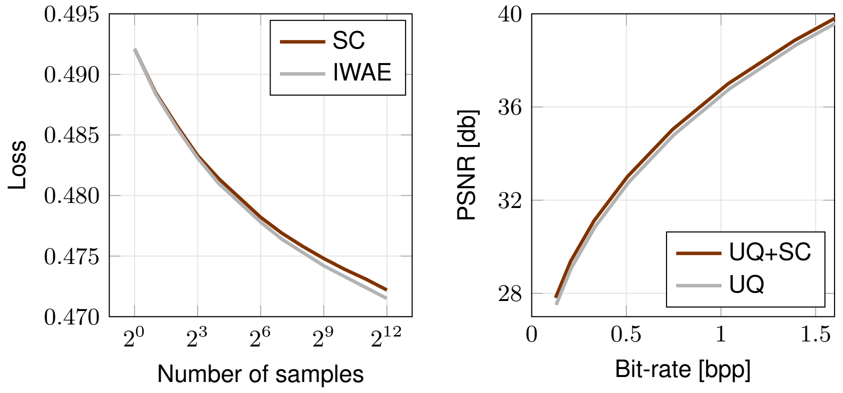

Figure 2: *Left:* Rate-distortion loss (Eq. 10, $\lambda = 0.003125$) of a deep model using a hyperprior with (SC) and without (IWAE) taking into account the overhead $\Psi_N$. *Right:* Rate-distortion curves for a deep model using universal quantization with and without softmin coding with up to $4096$ samples.

## Appendix C

Following Agustsson and Theis (2020), we trained a linear and a deep model with uniform encoders for a VAE loss. The loss corresponds to the coding cost if universal quantization (UQ) is used.

In addition, we trained the same models using an IWAE loss. After adding a term for the overhead ($\Psi_N$), this loss to the coding cost if universal quantization is combined with softmin coding (UQ+SC). Linear models were trained with 32 samples, deep models were trained with 16 samples. We used batch sizes of $512/N$ to make comparisons between UQ and UQ+SC fair in terms of computational resources. Models were trained on images of size 256 by 256 pixels.

The linear model independently transforms 8x8 pixel blocks into 192 coefficients each. The marginal distribution of each coefficient is modeled independently by a flexible univariate distribution.

The deep model's encoder uses 4 strided convolutional layers with a kernel sizes of 5. Each convolution is followed with downsampling by a factor of 2 and a generalized divisive normalization (GDN; Ballé et al., 2016). The decoder similarly uses 4 transposed convolutions and upsampling by a factor of 2, each followed by an inverse GDN operation. Like the linear encoder, the deep encoder outputs 192 channels. The distribution of these coefficients is modeled with a "mean and scale hyperprior" (Ballé et al., 2018; Minnen et al., 2018).

The linear model was applied to 8x8 images patches before applying softmin coding. The deep model was used to independently encode 64x64 image patches of an image using softmin coding. For each target $\lambda$, we used the number of samples $N$ which minimized the coding cost (Fig. 1, middle), up to a number of samples of 4096.