# Bit-Swap: Recursive Bits-Back Coding for Lossless Compression with Hierarchical Latent Variables

**Friso H. Kingma** [1] **Pieter Abbeel** [1] **Jonathan Ho** [1]

## Abstract

The bits-back argument suggests that latent variable models can be turned into lossless compression schemes. Translating the bits-back argument into efficient and practical lossless compression schemes for general latent variable models, however, is still an open problem. Bits-Back with Asymmetric Numeral Systems (BB-ANS), recently proposed by Townsend et al. (2019), makes bits-back coding practically feasible for latent variable models with one latent layer, but it is inefficient for hierarchical latent variable models. In this paper we propose Bit-Swap, a new compression scheme that generalizes BB-ANS and achieves strictly better compression rates for hierarchical latent variable models with Markov chain structure. Through experiments we verify that Bit-Swap results in lossless compression rates that are empirically superior to existing techniques.

## 1. Introduction

Likelihood-based generative models—models of joint probability distributions trained by maximum likelihood—have recently achieved large advances in density estimation performance on complex, high-dimensional data. Variational autoencoders (Kingma & Welling, 2013; Kingma et al., 2016), PixelRNN and PixelCNN and their variants (Oord et al., 2016; van den Oord et al., 2016b; Salimans et al., 2017; Parmar et al., 2018; Chen et al., 2017), and flow-based models like RealNVP (Dinh et al., 2014; 2016; Kingma & Dhariwal, 2018) can successfully model high dimensional image, video, speech, and text data (Karras et al., 2017; Kalchbrenner et al., 2016b; van den Oord et al., 2016a; Kalchbrenner et al., 2016a; 2018; Vaswani et al., 2017).

The excellent density estimation performance of modern likelihood-based models suggests another application: lossless compression. Any probability distribution can in theory be converted into a lossless code, in which each datapoint is encoded into a number of bits equal to its negative log probability assigned by the model. Since the best expected codelength is achieved when the model matches the true data distribution, designing a good lossless compression algorithm is a matter of jointly solving two problems:

1. Approximating the true data distribution $p_{\text{data}}(\mathbf{x})$ as well as possible with a model $p_{\boldsymbol{\theta}}(\mathbf{x})$
2. Developing a practical compression algorithm, called an *entropy encoding* scheme, that is compatible with this model and results in codelengths equal to $-\log p_{\boldsymbol{\theta}}(\mathbf{x})$.

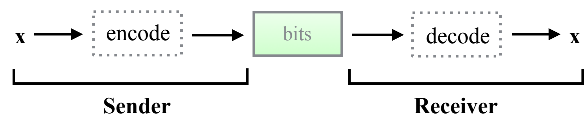
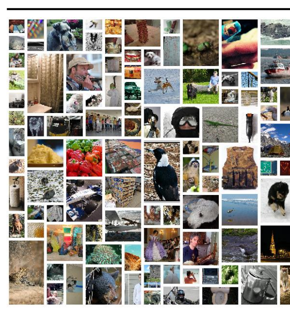



Figure 1: Schematic overview of lossless compression. The sender encodes data $\mathbf{x}$ to a code with the least amount of bits possible without losing information. The receiver decodes the code and must be able to exactly reconstruct $\mathbf{x}$.

Table 1: Lossless compression rates (in bits per dimension) on unscaled and cropped ImageNet of Bit-Swap against other compression schemes. See Section 5 for an explanation of Bit-Swap and Section 6 for detailed results.



| Compression Scheme | Rate |
|---|---|
| Uncompressed | 8.00 |
| GNU Gzip | 5.96 |
| bzip2 | 5.07 |
| LZMA | 5.09 |
| PNG | 4.71 |
| WebP | 3.66 |
| BB-ANS | 3.62 |
| **Bits-Swap (ours)** | **3.51** |

[1]University of California, Berkeley, California, USA. Correspondence to: Friso H. Kingma <fhkingma@gmail.com>, Jonathan Ho <jonathanho@berkeley.edu>.

*Proceedings of the 36th International Conference on Machine Learning*, Long Beach, California, PMLR 97, 2019. 

Unfortunately, it is generally difficult to jointly design a likelihood-based model and an entropy encoding scheme that together achieve a good compression rate while remaining computationally efficient enough for practical use. Any model with tractable density evaluation can be converted into a code using Huffman coding, but building a Huffman tree requires resources that scale exponentially with the dimension of the data. The situation is more tractable, but still practically too inefficient, when autoregressive models are paired with arithmetic coding or asymmetric numeral systems (explained in Section 2.1). The compression rate will be excellent due to to the effectiveness of autoregressive models in density estimation, but the resulting decompression process, which is essentially identical to the sample generation process, will be extremely slow.

Fortunately, fast compression and decompression can be achieved by pairing variational autoencoders with a recently proposed practically efficient coding method called Bits-Back with Asymmetric Numeral Systems (BB-ANS) (Townsend et al., 2019). However, the practical efficiency of BB-ANS rests on two requirements:

1. All involved inference and recognition networks are fully factorized probability distributions
2. There are few latent layers in the variational autoencoder.

The first requirement ensures that encoding and decoding is always fast and parallelizable. The second, as we will discuss later, ensures that BB-ANS achieves an efficient bitrate: it turns out that BB-ANS incurs an overhead that grows with the number of latent variables. But these requirements restrict the capacity of the variational autoencoder and pose difficulties for density estimation performance, and hence the resulting compression rate suffers.

To work toward designing a computationally efficient compression algorithm with a good compression rate, we propose Bit-Swap, which improves BB-ANS's performance on hierarchical latent variable models with Markov chain structure. Compared to latent variables models with only one latent layer, these hierarchical latent variable models allow us to achieve better density estimation performance on complex high-dimensional distributions. Meanwhile, Bit-Swap, as we show theoretically and empirically, yields a better compression rate compared to BB-ANS on these models due to reduced overhead.

## 2. Background

First, we will set the stage by introducing the lossless compression problem. Let $p_{\text{data}}$ be a distribution over discrete data $\mathbf{x} = (x_1, \ldots, x_D)$. Each component $x_1, \ldots, x_D$ of $\mathbf{x}$ is called a *symbol*. Suppose that a *sender* would like to communicate a sample $\mathbf{x}$ to a *receiver* through a *code*, or equivalently a *message*. The goal of lossless compression is to send this message using the minimum amount of bits on average over $p_{\text{data}}$, while ensuring that $\mathbf{x}$ is always fully recoverable by the receiver. See Figure 1 for an illustration.

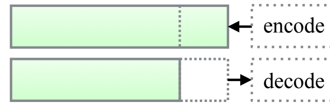



Figure 2: Asymmetric Numeral Systems (ANS) operates on a bitstream in a stack-like manner. Symbols are decoded in opposite order as they were encoded.

*Entropy encoding* schemes use a probabilistic model $p_{\boldsymbol{\theta}}(\mathbf{x})$ to define a code with codelengths $-\log p_{\boldsymbol{\theta}}(\mathbf{x})$. If $-\log p_{\boldsymbol{\theta}}(\mathbf{x})$ matches $-\log p_{\text{data}}(\mathbf{x})$ well, then resulting average codelength $\mathbb{E}[-\log p_{\boldsymbol{\theta}}(\mathbf{x})]$ will be close to the entropy of the data $H(\mathbf{x})$, which is the average codelength of an optimal compression scheme.

### 2.1. Asymmetric Numeral Systems

We will employ a particular entropy encoding scheme called *Asymmetric Numeral Systems* (ANS) (Duda, 2009). Given a univariate probability distribution $p(x)$, ANS encodes a symbol $x$ into a sequence of bits, or *bitstream*, of length approximately $-\log p(x)$ bits. ANS can also code a vector of symbols $\mathbf{x}$ using a fully factorized probability distribution $p(\mathbf{x}) = \prod_i p(x_i)$, resulting in $-\log p(\mathbf{x})$ bits. (It also works with autoregressive $p(\mathbf{x})$, but throughout this work we will only use fully factorized models for parallelizability purposes.)

ANS has an important property: if a sequence of symbols is encoded, then they must be decoded in the opposite order. In other words, the state of the ANS algorithm is a bitstream with a *stack* structure. Every time a symbol is encoded, bits are pushed on to the right of the stack; every time a symbol is decoded, bits are popped from the right of the stack. See Figure 2 for an illustration. This property will become important when ANS is used in BB-ANS and Bit-Swap for coding with latent variable models.

### 2.2. Latent Variable Models

The codelength of an entropy encoding technique depends on how well its underlying model $p_{\boldsymbol{\theta}}(\mathbf{x})$ approximates the true data distribution $p_{\text{data}}(\mathbf{x})$. In this paper, we focus on latent variable models, which approximate $p_{\text{data}}(\mathbf{x})$ with a marginal distribution $p_{\boldsymbol{\theta}}(\mathbf{x})$ defined by

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) d\mathbf{z} = \int p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) p(\mathbf{z}) d\mathbf{z} \qquad (1)$$

where $\mathbf{z}$ is an unobserved latent variable. For continuous $\mathbf{z}$, $p_{\boldsymbol{\theta}}(\mathbf{x})$ can be seen as an infinite mixture, which makes such an implicit distribution over $\mathbf{x}$ potentially highly flexible.

Since exactly evaluating and optimizing the marginal likelihood $p_{\boldsymbol{\theta}}(\mathbf{x})$ is intractable, variational autoencoders introduce an *inference model* $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, which approximates the model posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$. For any choice of $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, we can rewrite the marginal likelihood $p_{\boldsymbol{\theta}}(\mathbf{x})$ as follows:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \underbrace{\mathbb{E}_{q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})} \log \frac{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}{q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})}}_{=\mathcal{L}(\boldsymbol{\theta})\ \text{(ELBO)}} + \underbrace{\mathbb{E}_{q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})} \log \frac{q_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})}}_{=D_{\mathrm{KL}}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})\ \|\ p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}))} \tag{2}$$

As $D_{\mathrm{KL}}\left(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})\right) \geq 0$, the inference model and generative model can be found by jointly optimizing the Evidence Lower BOund (ELBO), a lower bound on $\log p_{\boldsymbol{\theta}}(\mathbf{x})$:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z}) - \log q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})] \tag{3}$$

For continuous $\mathbf{z}$ and a differentiable inference model and generative model, the ELBO can be optimized using the reparameterization trick (Kingma & Welling, 2013).

### 2.3. Bits-Back Coding with ANS

It is not straightforward to use a latent variable model for compression, but it is possible with the help of the inference network $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$. Assume that both the sender and receiver have access to $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$, $p(\mathbf{z})$, $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ and an entropy encoding scheme. Let $\mathbf{x}$ be the datapoint the sender wishes to communicate. The sender can send a latent sample $\mathbf{z} \sim q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ by coding using the prior $p(\mathbf{z})$, along with $\mathbf{x}$, coded with $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$. This scheme is clearly valid and allows the receiver to recover the $\mathbf{x}$, but results in an inefficient total codelength of $\mathbb{E}\left[-\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) - \log p(\mathbf{z})\right]$. Wallace (1990) and Hinton & Van Camp (1993) show in a thought experiment, called the bits-back argument, it is possible to instead transmit $-\log q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ fewer bits in a certain sense, thereby yielding a better net codelength equal to the negative ELBO $-\mathcal{L}(\boldsymbol{\theta})$ of the latent variable model.

BB-ANS (Townsend et al., 2019), illustrated in Figure 3, makes the bits-back argument concrete. BB-ANS operates by starting with ANS initialized with a bitstream of $N_{\text{init}}$ random bits. Then, to encode $\mathbf{x}$, BB-ANS performs the following steps:

1. Decode $\mathbf{z}$ from bitstream using $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, subtracting $-\log q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ bits from the bitstream,

2. Encode $\mathbf{x}$ to bitstream using $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$, adding $-\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ bits to the bitstream,

3. Encode $\mathbf{z}$ to bitstream using $p(\mathbf{z})$, adding $-\log p(\mathbf{z})$ bits to the bistream.

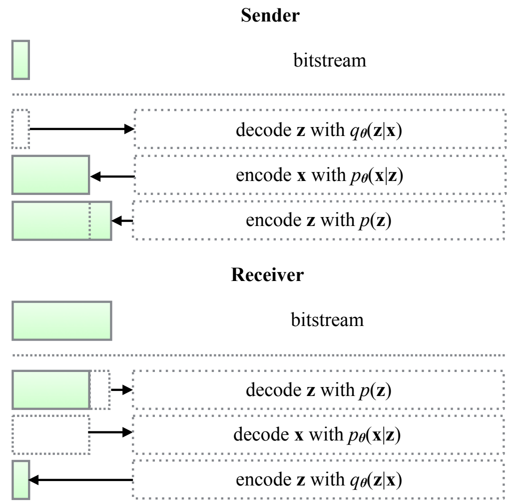



Figure 3: Bits-Back with Asymmetric Numeral Systems (BB-ANS).

The resulting bitstream, which has a length of $N_{\text{total}} \coloneqq N_{\text{init}} + \log q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) - \log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) - \log p(\mathbf{z})$ bits, is then sent to the receiver.

The receiver decodes the data by initializing ANS to the received bitstream, then proceeds in reverse order, with the encode and decode operations swapped: the receiver decodes $\mathbf{z}$ using $p(\mathbf{z})$, decodes $\mathbf{x}$ using $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$, then encodes $\mathbf{z}$ using $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$. The final step of encoding $\mathbf{z}$ will recover the $N_{\text{init}}$ bits that the encoder used to initialize ANS. Thus, the sender will have successfully transmitted $\mathbf{x}$ to the receiver, along with the initial $N_{\text{init}}$ bits—and it will have taken $N_{\text{total}}$ bits to do so.

To summarize, it takes $N_{\text{total}}$ bits to transmit $\mathbf{x}$ plus $N_{\text{init}}$ bits. In this sense, the *net* number of bits sent regarding $\mathbf{x}$ only, ignoring the initial $N_{\text{init}}$ bits, is

$$N_{\text{total}} - N_{\text{init}} = \log q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) - \log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) - \log p(\mathbf{z})$$

which is on average equal to $-\mathcal{L}(\boldsymbol{\theta})$, the negative ELBO.

## 3. Initial Bits in Bits-Back Coding

We now turn to the core issue that our work addresses: the amount of initial bits $N_{\text{init}}$ required for BB-ANS to function properly.

It is crucial for there to be enough initial bits in the ANS state for the sender to decode $\mathbf{z}$ from the initial bitstream. That is, we must have

$$N_{\text{init}} \geq -\log q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \tag{4}$$

in order to guarantee that the receiver can recover the initial $N_{\text{init}}$ bits. If not, then to sample $\mathbf{z}$, the sender must draw

**Algorithm 1** BB-ANS for lossless compression with hierarchical latent variables. The operations below show the procedure for encoding a dataset $\mathcal{D}$ onto a bitstream.

**Input:** data $\mathcal{D}$, depth $L$, $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}_{1:L})$, $q_{\boldsymbol{\theta}}(\mathbf{z}_{1:L}|\mathbf{x})$
**Require:** ANS
**Initialize:** bitstream
**repeat**
  Take $\mathbf{x} \in \mathcal{D}$
  decode $\mathbf{z}_1$ with $q_{\boldsymbol{\theta}}(\mathbf{z}_1|\mathbf{x})$
  **for** $i = 1$ **to** $L-1$ **do**
    decode $\mathbf{z}_{i+1}$ with $q_{\boldsymbol{\theta}}(\mathbf{z}_{i+1}|\mathbf{z}_i)$
  **end for**
  encode $\mathbf{x}$ with $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}_1)$
  **for** $i = 1$ **to** $L-1$ **do**
    encode $\mathbf{z}_i$ with $p_{\boldsymbol{\theta}}(\mathbf{z}_i|\mathbf{z}_{i+1})$
  **end for**
  encode $\mathbf{z}_L$ with $p(\mathbf{z}_L)$
**until** $\mathcal{D} = \varnothing$
**Send:** bitstream

**Algorithm 2** **Bit-Swap (ours)** for lossless compression with hierarchical latent variables. The operations below show the procedure for encoding a dataset $\mathcal{D}$ onto a bitstream.

**Input:** data $\mathcal{D}$, depth $L$, $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}_{1:L})$, $q_{\boldsymbol{\theta}}(\mathbf{z}_{1:L}|\mathbf{x})$
**Require:** ANS
**Initialize:** bitstream
**repeat**
  Take $\mathbf{x} \in \mathcal{D}$
  decode $\mathbf{z}_1$ with $q_{\boldsymbol{\theta}}(\mathbf{z}_1|\mathbf{x})$
  encode $\mathbf{x}$ with $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}_1)$
  **for** $i = 1$ **to** $L-1$ **do**
    decode $\mathbf{z}_{i+1}$ with $q_{\boldsymbol{\theta}}(\mathbf{z}_{i+1}|\mathbf{z}_i)$
    encode $\mathbf{z}_i$ with $p_{\boldsymbol{\theta}}(\mathbf{z}_i|\mathbf{z}_{i+1})$
  **end for**
  encode $\mathbf{z}_L$ with $p(\mathbf{z}_L)$
**until** $\mathcal{D} = \varnothing$
**Send:** bitstream

bits from an auxiliary random source, and those bits will certainly not be recoverable by the receiver. And, if those bits are not recoverable, then the sender will have spent $N_{\text{total}}$ bits to transmit $\mathbf{x}$ *only*, without $N_{\text{init}}$ bits in addition. So, we must commit to sending at least $N_{\text{init}} \geq -\log q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ initial bits to guarantee a short net codelength for $\mathbf{x}$.

Unfortunately, the initial number of bits required can be significant. As an example, if the latent variables are continuous, as is common with variational autoencoders, one must discretize the density $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ into bins of volume $\delta\mathbf{z}$, yielding a probability mass function $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})\delta\mathbf{z}$. But this imposes a requirement on the initial bits: now $N_{\text{init}} \geq -\log q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) - \log \delta\mathbf{z}$ increases as the discretization resolution $1/\delta\mathbf{z}$ increases.

Townsend et al. (2019) remark that initial bits can be avoided by transmitting multiple datapoints in sequence, where every datapoint $\mathbf{x}^i$ (except for the first one $\mathbf{x}^1$) uses the bitstream built up thus far as initial bitstream. This *amortizes* the initial cost when the number of datapoints transmitted is large, but the cost can be significant for few or moderate numbers of datapoints, as we will see in experiments in Section 6.

## 4. Problem Scenario: Hierarchical Latent Variables

Initial bits issues also arise when the model has many latent variables. Models with multiple latent variables are more expressive in practice can more closely model $p_{\text{data}}$, leading to better compression performance. But since $-\log q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ generally grows with the dimension of $\mathbf{z}$, adding more expressive power to the latent variable model via more latent variables will incur a larger initial bitstream for BB-ANS.

We specialize our discussion to the case of hierarchical latent variable models: variational autoencoders with multiple latent variables whose sampling process obeys a Markov chain of the form $\mathbf{z}_L \to \mathbf{z}_{L-1} \to \cdots \to \mathbf{z}_1 \to \mathbf{x}$, shown schematically in Figure 4. (It is well known that such models are better density estimators than shallower models, and we will verify in experiments in Section 6 that these models indeed can model $p_{\text{data}}$ more closely than standard variational autoencoders. A discussion regarding other topologies can be found in Appendix G.) Specifically, we consider a model whose marginal distributions are

$$\begin{aligned}
p_{\boldsymbol{\theta}}(\mathbf{x}) &= \int p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}_1)p_{\boldsymbol{\theta}}(\mathbf{z}_1)\mathrm{d}\mathbf{z}_1 \\
p_{\boldsymbol{\theta}}(\mathbf{z}_1) &= \int p_{\boldsymbol{\theta}}(\mathbf{z}_1|\mathbf{z}_2)p_{\boldsymbol{\theta}}(\mathbf{z}_2)\mathrm{d}\mathbf{z}_2 \\
&\vdots \\
p_{\boldsymbol{\theta}}(\mathbf{z}_{L-1}) &= \int p_{\boldsymbol{\theta}}(\mathbf{z}_{L-1}|\mathbf{z}_L)p(\mathbf{z}_L)\mathrm{d}\mathbf{z}_L,
\end{aligned} \tag{5}$$

and whose marginal distribution over $\mathbf{x}$ is

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}_1)p_{\boldsymbol{\theta}}(\mathbf{z}_1|\mathbf{z}_2)\cdots p(\mathbf{z}_L)\mathrm{d}\mathbf{z}_{1:L}. \tag{6}$$

We define an inference model $q_{\boldsymbol{\theta}}(\mathbf{z}_i|\mathbf{z}_{i-1})$ for every latent layer $\mathbf{z}_i$, so that we can optimize a variational bound on the marginal likelihood $p_{\boldsymbol{\theta}}(\mathbf{x})$. The resulting optimization objective (ELBO) is

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{q_{\boldsymbol{\theta}}(\cdot|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}_{1:L}) - \log q_{\boldsymbol{\theta}}(\mathbf{z}_{1:L}|\mathbf{x})]. \tag{7}$$

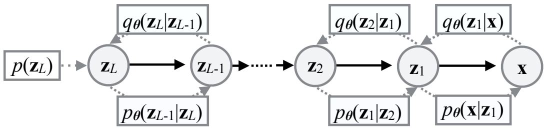



Figure 4: The model class we are targeting: hierarchical latent variable models. Specifically, variational autoencoders whose sampling process obeys a Markov chain.

Now, consider what happens when this model is used with BB-ANS for compression. Figure 5(b) illustrates BB-ANS for such a model with three latent layers $\{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3\}$; the algorithm for arbitrary latent depths $L$ of a hierarchical latent variable model is shown in Algorithm 1.

The first thing the sender must do is decode the latent variables $\mathbf{z}_{1:L}$ from the initial bitstream of ANS. So, the number of bits present in the initial bitstream must be at least

$$N_{\text{init}}^{\text{BBANS}} := -\log q_{\boldsymbol{\theta}}(\mathbf{z}_{1:L}|\mathbf{x}) = \sum_{i=0}^{L-1} -\log q_{\boldsymbol{\theta}}(\mathbf{z}_{i+1}|\mathbf{z}_i) \tag{8}$$

where $\mathbf{z}_0 = \mathbf{x}$. Notice that $N_{\text{init}}^{\text{BBANS}}$ must grow with the depth $L$ of the latent variable model. With $L$ sufficiently large, the required initial bits could make BB-ANS impractical as a compression scheme with hierarchical latent variables.

## 5. Bit-Swap

To mitigate this issue, we propose Bit-Swap (Algorithm 2), an improved compression scheme that makes bits-back coding efficiently compatible with the layered structure of hierarchical latent variable models.

In our proposed model (Equations 5-6), the sampling process of both the generative model and the inference model obeys a Markov chain dependency between the stochastic variables. The data $\mathbf{x}$ is generated conditioned on a latent variable $\mathbf{z}_1$, as in a standard variational autoencoder. However, instead of using a fixed prior for $\mathbf{z}_1$, we assume that $\mathbf{z}_1$ is generated by a second latent variable $\mathbf{z}_2$. Subsequently, instead of using a fixed prior for $\mathbf{z}_2$, we assume that $\mathbf{z}_2$ is generated by third latent variable $\mathbf{z}_3$, and so on.

These nested dependencies enable us to recursively apply the bits-back argument as follows. Suppose we aim to compress one datapoint $\mathbf{x}$ in a lossless manner. With standard BB-ANS, the sender begins by decoding $\mathbf{z}_{1:L}$, which incurs a large cost of initial bits. With Bit-Swap, we notice that we can apply the first two steps of the bits-back argument on the first latent variable: first decode $\mathbf{z}_1$ and directly afterwards encode $\mathbf{x}$. This adds bits to the bitstream, which means that further decoding operations for $\mathbf{z}_{2:L}$ will need fewer initial bits to proceed. Now, we recursively the bits-back argument for the second latent variable $\mathbf{z}_2$ in a similar fashion: first decode $\mathbf{z}_2$ and afterwards encode $\mathbf{z}_1$. Similar operations of encoding and decoding can be performed for the remaining latent variables $\mathbf{z}_{3:L}$: right before decoding $\mathbf{z}_{i+1}$, Bit-Swap always encodes $\mathbf{z}_{i-1}$, and hence at least $-\log p_{\boldsymbol{\theta}}(\mathbf{z}_{i-1}|\mathbf{z}_i)$ are available to decode $\mathbf{z}_{i+1} \sim q_{\boldsymbol{\theta}}(\mathbf{z}_{i+1}|\mathbf{z}_i)$ without an extra cost of initial bits. Therefore, the amount of initial bits that Bit-Swap needs is bounded by $\sum_{i=0}^{L-1} \max\left(0, \log \frac{p_{\boldsymbol{\theta}}(\mathbf{z}_{i-1}|\mathbf{z}_i)}{q_{\boldsymbol{\theta}}(\mathbf{z}_{i+1}|\mathbf{z}_i)}\right)$, where we used the convention $\mathbf{z}_0 = \mathbf{x}$ and $p_{\boldsymbol{\theta}}(\mathbf{z}_{-1}|\mathbf{z}_0) = 1$. We can guarantee that Bit-Swap requires no more initial bits than BB-ANS:

$$N_{\text{init}}^{\text{BitSwap}} \leq \sum_{i=0}^{L-1} \max\left(0, \log \frac{p_{\boldsymbol{\theta}}(\mathbf{z}_{i-1}|\mathbf{z}_i)}{q_{\boldsymbol{\theta}}(\mathbf{z}_{i+1}|\mathbf{z}_i)}\right) \tag{9}$$

$$\leq \sum_{i=0}^{L-1} -\log q_{\boldsymbol{\theta}}(\mathbf{z}_{i+1}|\mathbf{z}_i) = N_{\text{init}}^{\text{BBANS}} \tag{10}$$

See Figure 5(a) for an illustration of Bit-Swap on a model with three latent variables $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3$.

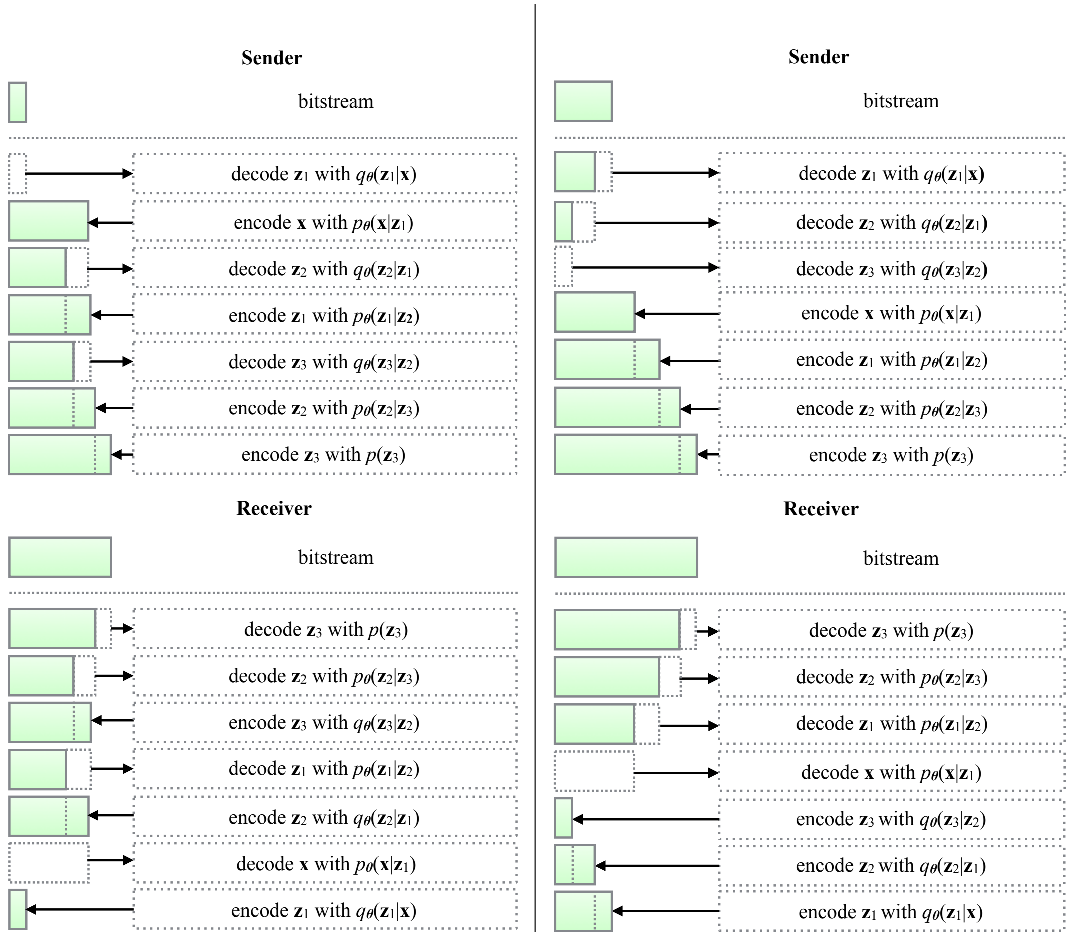
## 6. Experiments

To compare Bit-Swap against BB-ANS, we use the following image datasets: MNIST, CIFAR-10 and ImageNet ($32 \times 32$). Note that the methods are not constrained to this specific type of data. As long as it is feasible to learn a hierarchical latent variable model with Markov chain structure $p_{\boldsymbol{\theta}}(\mathbf{x})$ of the data under the given model assumptions, and the data is discrete, it is possible to execute the compression schemes Bit-Swap and BB-ANS on this data.

Referring back to the introduction, designing a good lossless compression algorithm is a matter of jointly solving two problems: **1)** approximating the true data distribution $p_{\text{data}}(\mathbf{x})$ as well as possible with a model $p_{\boldsymbol{\theta}}(\mathbf{x})$, and **2)** developing a practical compression scheme that is compatible with this model and results in codelengths equal to $-\log p_{\boldsymbol{\theta}}(\mathbf{x})$. We address the first point in Section 6.1. As for the second point, we achieve bitrates that are approximately equal to the $-\mathcal{L}(\boldsymbol{\theta})$, the negative ELBO, which is an upper bound on $-\log p_{\boldsymbol{\theta}}(\mathbf{x})$. We will address this in Section 6.2.

### 6.1. Performance of Hierarchical Latent Variable Models

We begin our experiments by demonstrating how hierarchical latent variable models with Markov chain structure with different latent layer depths compare to a latent variable model with only one latent variable in terms of how well the models are able to approximate a true data distribution




(a) **Bit-Swap (ours)** (b) BB-ANS

Figure 5: **Bit-Swap (ours, left)** vs. BB-ANS (right) on a hierarchical latent variable model with three latent layers. Notice that BB-ANS needs a longer initial bitstream compared to Bit-Swap.

Table 2: **MNIST** model optimization results (columns 2 and 3) and test data compression results (columns 4 to 8) for various depths of the model (column 1). Column 2 shows the ELBO in bits/dim of the trained models evaluated on the test data. Column 3 denotes the number of parameters used (in millions). Using the trained models, we executed Bit-Swap and BB-ANS on the test data. We used $2^{10}$ bins to discretize the latent space (see Appendix F). Column 5 denotes the scheme used; BB-ANS or Bit-Swap. Column 4 denotes the average net bitrate in bits/dim (see Section 6.2), averaged over Bit-Swap and BB-ANS. Columns 6-8 show the cumulative moving average in bits/dim (CMA) (see Section 6.2) at various timesteps (1, 50 and 100 respectively). The reported bitrates are the result of compression of 100 datapoints (timesteps), averaged over 100 experiments. We believe that the small discrepancy between the ELBO and the net bitrate comes from the noise resulting from discretization. Also, Bit-Swap reduces to BB-ANS for $L = 1$.

| Depth ($L$) | ELBO $-\mathcal{L}(\boldsymbol{\theta})$ | # Parameters | Avg. Net Bitrate | Scheme | Initial ($n=1$) | CMA ($n=50$) | CMA ($n=100$) |
|---|---|---|---|---|---|---|---|
| 1 | 1.35 | 2.84M | - | - | - | - | - |
| 2 | 1.28 | 2.75M | $1.28 \pm 0.34$ | BB-ANS | $6.59 \pm 0.30$ | $1.38 \pm 0.05$ | $1.33 \pm 0.03$ |
| | | | | **Bit-Swap** | $3.45 \pm 0.32$ | $1.32 \pm 0.05$ | $1.30 \pm 0.03$ |
| 4 | 1.27 | 2.67M | $1.27 \pm 0.34$ | BB-ANS | $11.63 \pm 0.30$ | $1.47 \pm 0.05$ | $1.37 \pm 0.04$ |
| | | | | **Bit-Swap** | $3.40 \pm 0.31$ | $1.31 \pm 0.05$ | $1.29 \pm 0.04$ |
| 8 | 1.27 | 2.60M | $1.27 \pm 0.33$ | BB-ANS | $21.93 \pm 0.34$ | $1.68 \pm 0.05$ | $1.48 \pm 0.03$ |
| | | | | **Bit-Swap** | $3.34 \pm 0.33$ | $1.31 \pm 0.05$ | $1.29 \pm 0.03$ |

Table 3: **CIFAR-10** model optimization (columns 2 and 3) and test data compression results (columns 4 to 8) for various depths of the model (column 1). Equal comments apply as Table 2.

| Depth ($L$) | ELBO $-\mathcal{L}(\boldsymbol{\theta})$ | # Parameters | Avg. Net Bitrate | Scheme | Initial ($n=1$) | CMA ($n=50$) | CMA ($n=100$) |
|---|---|---|---|---|---|---|---|
| 1 | 4.57 | 45.3M | - | - | - | - | - |
| 2 | 3.83 | 45.0M | $3.85 \pm 0.77$ | BB-ANS | $12.66 \pm 0.61$ | $4.03 \pm 0.11$ | $3.93 \pm 0.08$ |
| | | | | **Bit-Swap** | $6.76 \pm 0.63$ | $3.91 \pm 0.11$ | $3.87 \pm 0.08$ |
| 4 | 3.81 | 44.9M | $3.82 \pm 0.83$ | BB-ANS | $22.30 \pm 0.83$ | $4.19 \pm 0.12$ | $4.00 \pm 0.09$ |
| | | | | **Bit-Swap** | $6.72 \pm 0.67$ | $3.89 \pm 0.12$ | $3.85 \pm 0.09$ |
| 8 | 3.78 | 44.7M | $3.79 \pm 0.80$ | BB-ANS | $44.24 \pm 0.87$ | $4.60 \pm 0.12$ | $4.19 \pm 0.09$ |
| | | | | **Bit-Swap** | $6.53 \pm 0.74$ | $3.86 \pm 0.12$ | $3.82 \pm 0.09$ |

Table 4: **ImageNet** ($32 \times 32$) model optimization (columns 2 and 3) and test data compression results (columns 4 to 8) for various depths of the model (column 1). Equal comments apply as Table 2.

| Depth ($L$) | ELBO $-\mathcal{L}(\boldsymbol{\theta})$ | # Parameters | Avg. Net Bitrate | Scheme | Initial ($n=1$) | CMA ($n=50$) | CMA ($n=100$) |
|---|---|---|---|---|---|---|---|
| 1 | 4.94 | 45.3M | - | - | - | - | - |
| 2 | 4.53 | 45.0M | $4.54 \pm 0.84$ | BB-ANS | $13.39 \pm 0.60$ | $4.71 \pm 0.13$ | $4.63 \pm 0.08$ |
| | | | | **Bit-Swap** | $7.45 \pm 0.62$ | $4.60 \pm 0.13$ | $4.57 \pm 0.08$ |
| 4 | 4.48 | 44.9M | $4.48 \pm 0.85$ | BB-ANS | $22.72 \pm 0.79$ | $4.84 \pm 0.13$ | $4.66 \pm 0.08$ |
| | | | | **Bit-Swap** | $6.97 \pm 0.70$ | $4.53 \pm 0.13$ | $4.50 \pm 0.08$ |

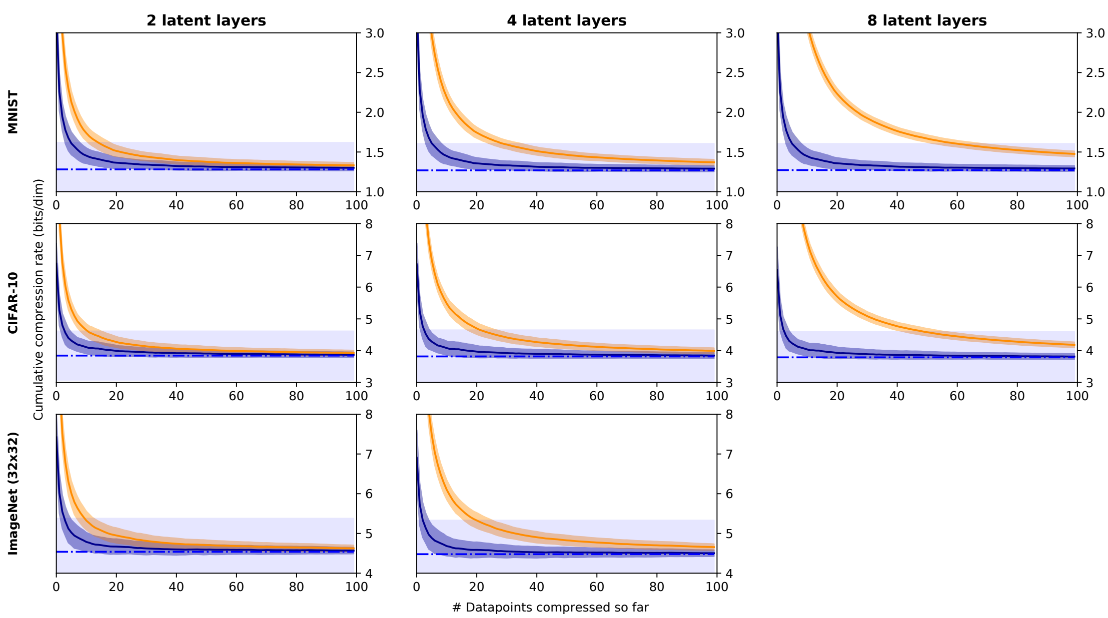



Figure 6: Cumulative moving average of compression rate over time for Bit-Swap (blue) and BB-ANS (orange) for sequences of 100 datapoints, averaged over 100 experiments. The blue dotted line and region represent the average and standard deviation of the net bitrate across the entire test set, without the initial bits (see Section 6.2).

$p_{\text{data}}(\mathbf{x})$. A detailed discussion on the model architecture design can be found in Appendix D.

The results of training of the hierarchical latent variable models are shown in the left three columns of Table 2 (MNIST), 3 (CIFAR-10) and 4 (ImageNet ($32 \times 32$)). One latent layer corresponds to *one* latent variable $\mathbf{z}_i$. The metric we used is bits per dimension (bits/dim) as evaluated by the negative ELBO $-L(\boldsymbol{\theta})$. Note from the resulting ELBO that, as we add more latent layers, the expressive power increases. A discussion on the utility of more latent layers can be found in Appendix E.

### 6.2. Performance of Bit-Swap versus BB-ANS

We now show that Bit-Swap indeed reduces the initial bits required (as discussed in Section 5) and outperforms BB-ANS on hierarchical latent variable models in terms of actual compression rates. To compare the performance of Bit-Swap versus BB-ANS for different depths of the latent layers, we conducted 100 experiments for every model and dataset. In every experiment we compressed 100 datapoints in sequence and calculated the cumulative moving average (CMA) of the resulting lengths of the bitstream after each datapoint. Note that this includes the initial bits necessary for decoding latent layers. In addition, we calculated the *net* number of bits added to the bitstream after every datapoint, as explained in Section 2.3, and averaged them over all datapoints and experiments for one dataset and model. This can be interpreted as a lower bound of the CMA of a particular model and dataset. We discretized the continuous latent variables $\mathbf{z}_{1:L}$ using $2^{10}$ discretization bins for all datasets and experiments, as explained in Appendix F.

The CMA (with the corresponding average net bitrate) over 100 experiments for every model and dataset is shown in Figure 6. Bit-Swap is depicted in blue and BB-ANS in orange. These graphs show two properties of Bit-Swap and BB-ANS: the difference between Bit-Swap and BB-ANS in the need for initial bits, and the fact that the CMA of Bit-Swap and BB-ANS both amortize towards the average net bitrate. The last five columns of Table 2 (MNIST), 3 (CIFAR-10) and 4 (ImageNet ($32 \times 32$)) show the CMA (in bits/dim) after 1, 50 and 100 datapoints for the Bit-Swap versus BB-ANS and the average net bitrate (in bits/dim).

The initial cost is amortized (see Section 3) as the amount of datapoints compressed grows. Also, the CMA converges to the average net bitrate. The relatively high initial cost of both compression schemes comes from the fact that the initial cost increases with the number of discretization bins, discussed in Appendix F. Furthermore, discretizing the latent space adds noise to the distributions. When using BB-ANS, remember that this initial cost also grows linearly with the amount of latent layers $L$. Bit-Swap does not have this problem. This results in a CMA performance gap that grows with the amount of latent layers $L$. The efficiency of Bit-Swap compared to BB-ANS results in much faster amortization, which makes Bit-Swap a more practical algorithm.

Finally, we compared both Bit-Swap and BB-ANS against a number of benchmark lossless compression schemes. For MNIST, CIFAR-10 and Imagenet ($32 \times 32$) we report the bitrates, shown in Table 5, as a result of compressing 100 datapoints in sequence (averaged over 100 experiments) and used the best models reported in Table 2, 3 and 4 to do so. We also compressed 100 single images independently taken from the original unscaled ImageNet, cropped to multiples of 32 pixels on each side, shown in Table 6. First, we trained the same model as used for Imagenet ($32 \times 32$) on random $32 \times 32$ patches of the corresponding train set. Then we executed Bit-Swap and BB-ANS by compressing one $32 \times 32$ block at the time and averaging the bitrates of all the blocks in *one* image. We used the same cropped images for the benchmark schemes. We did not include deep autoregressive models as benchmark, because they are too slow to be practical (see introduction). Bit-Swap clearly outperforms all other benchmark lossless compression schemes.

Table 5: Compression rates (in bits/dim) on MNIST, CIFAR-10, Imagenet ($32\times32$). The experimental set-up is explained in Section 6.2.

| | **MNIST** | **CIFAR-10** | **ImageNet** ($32 \times 32$) |
|---|---|---|---|
| Uncompressed | 8.00 | 8.00 | 8.00 |
| GNU Gzip | 1.65 | 7.37 | 7.31 |
| bzip2 | 1.59 | 6.98 | 7.00 |
| LZMA | 1.49 | 6.09 | 6.15 |
| PNG | 2.80 | 5.87 | 6.39 |
| WebP | 2.10 | 4.61 | 5.29 |
| BB-ANS | 1.48 | 4.19 | 4.66 |
| **Bit-Swap** | **1.29** | **3.82** | **4.50** |

Table 6: Compression rates (in bits/dim) on 100 images taken independently from unscaled and cropped ImageNet. The experimental set-up is explained in Section 6.2.

| | **ImageNet** (unscaled & cropped) |
|---|---|
| Uncompressed | 8.00 |
| GNU Gzip (Gailly & Adler, 2018) | 5.96 |
| bzip2 (Seward, 2010) | 5.07 |
| LZMA (Pavlov, 1996) | 5.09 |
| PNG | 4.71 |
| WebP | 3.66 |
| BB-ANS | 3.62 |
| **Bit-Swap** | **3.51** |

## 7. Conclusion

Bit-Swap advances the line of work on practical compression using latent variable models, starting from the theoretical bits-back argument (Wallace, 1990; Hinton & Van Camp, 1993), and continuing on to practical algorithms based on arithmetic coding (Frey & Hinton, 1996; Frey, 1998) and asymmetric numeral systems (Townsend et al., 2019).

Bit-Swap enables us to efficiently compress using hierarchical latent variable models with a Markov chain structure, as it is able to avoid a significant number of initial bits that BB-ANS requires to compress with the same models. The hierarchical latent variable models function as powerful density estimators, so combined with Bit-Swap, we obtain an efficient, low overhead lossless compression algorithm capable of effectively compressing complex, high-dimensional datasets.

## Acknowledgements

We want to thank Diederik Kingma for helpful comments and fruitful discussions. This work was funded in part by ONR PECASE N000141612723, Huawei, Amazon AWS, and Google Cloud.

## References

Chen, X., Kingma, D. P., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016.

Chen, X., Mishra, N., Rohaninejad, M., and Abbeel, P. Pixelsnail: An improved autoregressive generative model. *arXiv preprint arXiv:1712.09763*, 2017.

Dinh, L., Krueger, D., and Bengio, Y. Nice: non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using Real NVP. *arXiv preprint arXiv:1605.08803*, 2016.

Duda, J. Asymmetric numeral systems, 2009.

Frey, B. J. *Bayesian networks for pattern classification, data compression, and channel coding.* Citeseer, 1998.

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

Gailly, J.-l. and Adler, M. Gnu gzip, Aug 2018. URL https://www.gnu.org/software/gzip/manual/gzip.html.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

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

Kalchbrenner, N., Espheholt, L., Simonyan, K., Oord, A. v. d., Graves, A., and Kavukcuoglu, K. eural machine translation in linear time. *arXiv preprint arXiv:1610.00527*, 2016a.

Kalchbrenner, N., Oord, A. v. d., Simonyan, K., Danihelka, I., Vinyals, O., Graves, A., and Kavukcuoglu, K. Video pixel networks. *arXiv preprint arXiv:1610.00527*, 2016b.

Kalchbrenner, N., Elsen, E., Simonyan, K., Noury, S., Casagrande, N., Lockhart, E., Stimberg, F., Oord, A. v. d., Dieleman, S., and Kavukcuoglu, K. Efficient neural audio synthesis. *arXiv preprint arXiv:1802.08435*, 2018.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. *Proceedings of the International Conference on Learning Representations 2015*, 2015.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pp. 10236–10245, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. *Proceedings of the 2nd International Conference on Learning Representations*, 2013.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pp. 4743–4751, 2016.

MacKay, D. J. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Oord, A. v. d., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, Ł., Shazeer, N., and Ku, A. Image transformer. *arXiv preprint arXiv:1802.05751*, 2018.

Pavlov, I. Lzma, 1996. URL https://www.7-zip.org/sdk.html.

Polyak, B. T. and Juditsky, A. B. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

Salimans, T. and Kingma, D. P. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *arXiv preprint arXiv:1602.07868*, 2016.

Salimans, T., Karpathy, A., Chen, X., and Kingma, D. P. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

Seward, J. bzip2, Sep 2010. URL http://www.sourceware.org/bzip2/.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Townsend, J., Bird, T., and Barber, D. Practical lossless compression with latent variables using bits back coding. 2019.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016a.

van den Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., and Kavukcuoglu, K. Conditional image generation with PixelCNN decoders. *arXiv preprint arXiv:1606.05328*, 2016b.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Wallace, C. S. Classification by minimum-message-length inference. In *International Conference on Computing and Information*, pp. 72–81. Springer, 1990.

Witten, I. H., Neal, R. M., and Cleary, J. G. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987.

Zhao, S., Song, J., and Ermon, S. Learning hierarchical features from generative models. *CoRR*, abs/1702.08396, 2017. URL http://arxiv.org/abs/1702.08396.