Training Deep Generative Models: Variations on a Theme

Philip Bachman
McGill University, School of Computer Science
phil.bachman@gmail.com

Doina Precup
McGill University, School of Computer Science
dprecup@cs.mcgill.ca

Abstract

Recent techniques for training deep generative models are based on coaxing pairs of sample generating systems into agreement. Methods such as stochastic variational inference (as used in variational auto-encoders), denoising (as used in denoising auto-encoders), and contrastive divergence (as used to train Restricted Boltzmann Machines) all fit nicely under this interpretation. We formally develop this point of view, which provides a unified framework in which to compare and contrast many approaches to training deep generative models. We hope our effort might help other researchers compress their understanding of methods in this domain, and thus avoid getting overwhelmed as they proliferate.

1 Minimizing $\mathrm{KL}(q \,||\, p)$ – a useful thing to do

The KL-divergence $\mathrm{KL}(q \,||\, p)$ can provide a convenient upper bound on the expected negative log-likelihood of any variable x in any generative model p for which we can compute the joint log-density $\log p(x, \text{etc})$, where "etc" indicates an arbitrary collection of additional variables. The expectation can be taken w.r.t. any distribution, e.g. $\mathcal{D}(x)$, over x. That is, $\mathrm{KL}(q \,||\, p)$ can be used to upper bound:

$$\mathop{\mathbb{E}}_{\mathcal{D}(x)} \left[-\log \left(\mathop{\mathbb{E}}_{\text{etc} \sim p(\text{etc})} \left[p(x|\text{etc}) \right] \right) \right] \tag{1}$$

This bound lets us perform maximum likelihood inference in a wide range of settings. The bound depends on an auxiliary distribution $q(x, \text{etc})$. It requires independent samples from $q(x, \text{etc})$ and requires computing $\log q(\text{etc}|x)$. For practical purposes, we also need to compute gradients of $\log p(x, \text{etc})$ and $\log q(\text{etc}|x)$ with respect to their parameters. These are the only requirements for p and q.

Computations for this bound are equivalent to those for the variational free energy. However, our derivation differs from typical derivations of the variational free energy, particularly in our focus on behavior of the full joint distributions $p(x, \text{etc})$ and $q(x, \text{etc})$. While standard interpretations of the variational free energy readily suggest methods like the variational auto-encoders in [10, 11], they are less immediately helpful in understanding other methods, such as those in [13] and [12]. Our derivation brings these methods into a unified framework, and makes it easy to interpret denoising auto-encoders, GSNs, etc. as members of the same family of methods. Work in [5, 1, 6] can be interpreted as applications of our KL-based bound, using a "meta" auxiliary distribution formed by applying a normalized importance sampling correction to q.[1]

[1] Normalized importance sampling – in the context we consider – can be interpreted as sampling from an infinite mixture of discrete, finite-cardinality "resampling distributions". See Section 3 for further discussion.

1.1 Deriving the KL-based log-likelihood bound

To begin, assume distributions $q(x_0, ..., x_n)$ and $p(x_0, ..., x_n)$ over variables $\{x_0, ..., x_n\}$. We can say:

$$\begin{aligned}
\mathrm{KL}(q(x_0,...,x_n) \,||\, p(x_0, ..., x_n)) &= \mathop{\mathbb{E}}_{q(x_0,...,x_n)} \left[\log \frac{q(x_0, ..., x_n)}{p(x_0, ..., x_n)}\right] \\
&= \mathop{\mathbb{E}}_{q(x_0)} \left[\mathop{\mathbb{E}}_{q(x_1,...,x_n|x_0)} \left[\log \frac{q(x_0)q(x_1, ..., x_n|x_0)}{p(x_0)p(x_1, ..., x_n|x_0)}\right]\right] \\
&= \mathop{\mathbb{E}}_{q(x_0)} \left[\log \frac{q(x_0)}{p(x_0)} + \mathop{\mathbb{E}}_{q(x_1,...,x_n|x_0)} \left[\log \frac{q(x_1, ..., x_n|x_0)}{p(x_1, ..., x_n|x_0)}\right]\right] \\
&= \mathrm{KL}(q(x_0) \,||\, p(x_0)) + \mathop{\mathbb{E}}_{q(x_0)} \left[\mathop{\mathbb{E}}_{q(x_1,...,x_n|x_0)} \left[\log \frac{q(x_1, ..., x_n|x_0)}{p(x_1, ..., x_n|x_0)}\right]\right] \\
&= \mathrm{KL}(q(x_0) \,||\, p(x_0)) + \mathop{\mathbb{E}}_{q(x_0)} \Big[\mathrm{KL}(q(x_1, ..., x_n|x_0) \,||\, p(x_1, ..., x_n|x_0))\Big] \qquad (2) \\
&\geq \mathrm{KL}(q(x_0) \,||\, p(x_0)) \qquad (3)
\end{aligned}$$

Note that we chose our subscripts arbitrarily, so we can also say:

$$\mathrm{KL}(q(x_0, ..., x_n) \,||\, p(x_0, ..., x_n)) \geq \arg\max_i \Big[\mathrm{KL}(q(x_i) \,||\, p(x_i))\Big] \qquad (4)$$

1.2 Bounding expected log-likelihood for $x \sim \mathcal{D}$

Now, consider relabelling the distributions q and p to look like: $q(x, z_0, ..., z_n)$ and $p(x, z_0, ..., z_n)$. If the marginal $q(x)$ matches a particular distribution $\mathcal{D}$, i.e. $\forall x : \; q(x) = \mathcal{D}(x)$, then Eq. 2 and Eq. 4 let us say:

$$\begin{aligned}
\mathrm{KL}(q(x, z_0,..., z_n) \,||\, p(x, z_0, ..., z_n)) = \; &... \\
&= \mathrm{KL}(\mathcal{D}(x) \,||\, p(x)) + \mathop{\mathbb{E}}_{\mathcal{D}(x)} \Big[\mathrm{KL}(q(z_0, ..., z_n|x) \,||\, p(z_0, ..., z_n|x))\Big] \\
&\geq \mathop{\mathbb{E}}_{\mathcal{D}(x)} \big[\log \mathcal{D}(x)\big] + \mathop{\mathbb{E}}_{\mathcal{D}(x)} \big[-\log p(x)\big] \qquad (5)
\end{aligned}$$

Thus, if we can draw independent samples from $q(x, z_0, ..., z_n)$, and if the log-densities $\log q(z_0, ..., z_n|x)$ and $\log p(x, z_0, ..., z_n)$ are tractable – we won't optimize or evaluate $q(x) \triangleq \mathcal{D}(x)$, then we can minimize a bound on $\mathbb{E}_{\mathcal{D}(x)} \big[-\log p(x)\big]$ by minimizing $\mathrm{KL}(q(x, z_0, ..., z_n) \,||\, p(x, z_0, ..., z_n)) - \mathbb{E}_{\mathcal{D}(x)} \big[\log \mathcal{D}(x)\big]$.

Monte Carlo minimization of this KL-based bound on $\mathbb{E}_{\mathcal{D}(x)} \big[-\log p(x)\big]$ simply repeats two steps:

1. Sample $(x, z_0, ..., z_n)$ from $q(x, z_0, ..., z_n) \triangleq \mathcal{D}(x)q(z_0, ..., z_n|x)$.
2. Do a descent step to reduce: $\log q(z_0, ..., z_n|x) - \log p(x, z_0, ..., z_n)$.

Many approaches to training deep generative models are based on variants of this objective.

2 Denoising auto-encoders and GSNs

To bring denoising auto-encoders [4] and GSNs [3] into the KL bound framework, we first modify p and q from the previous subsection by incorporating additional latent variables $\bar{z}$ into q, and by defining:

$$p(\bar{z}) \triangleq \mathop{\mathbb{E}}_{q(x,z_1,...,z_n)} \big[q(\bar{z}|x, z_0, ..., z_n)\big] = q(\bar{z}) \qquad (6)$$

We also make some basic assumptions about p and q:

- We can draw independent samples from $q(x, z_1, ..., z_n, \bar{z})$.
- We can compute $\log q(z_1, ..., z_n|x)$ and $\log p(x, z_1, ..., z_n|\bar{z})$.

The key changes from the previous section are that we require $\log p(x, z_1, ..., z_n|\bar{z})$ rather than the full joint $\log p(x, z_1, ..., z_n, \bar{z})$, and that we require $\log q(z_1, ..., z_n|x)$ rather than $\log q(z_1, ..., z_n, \bar{z}|x)$.

Putting this all together, we can say:

$$\begin{aligned}
&\mathrm{KL}(q(x, z_0,..., z_n, \bar{z}) \,||\, p(x, z_0, ..., z_n, \bar{z})) = \ ... \\
&\quad = \mathop{\mathbb{E}}_{q(x,z_0,...,z_n,\bar{z})} \left[\log \frac{q(x, z_0, ..., z_n|\bar{z})q(\bar{z})}{p(x, z_0, ..., z_n|\bar{z})p(\bar{z})} \right] \\
&\quad = \mathop{\mathbb{E}}_{q(x,z_0,...,z_n,\bar{z})} \left[\log \frac{\mathcal{D}(x)q(z_0, ..., z_n, \bar{z}|x)}{p(x, z_0, ..., z_n|\bar{z})q(\bar{z})} \right] \\
&\quad = \mathop{\mathbb{E}}_{q(x,z_0,...,z_n,\bar{z})} \left[\log \frac{\mathcal{D}(x)q(z_0, ..., z_n, \bar{z}|x)}{p(x, z_0, ..., z_n|\bar{z})} - \log \left(\mathop{\mathbb{E}}_{q} \left[q(\bar{z}|\hat{x}, \hat{z}_0, ..., \hat{z}_n) \right] \right) \right] \qquad (7) \\
&\quad \geq \mathop{\mathbb{E}}_{\mathcal{D}(x)} \left[\log \mathcal{D}(x) \right] + \mathop{\mathbb{E}}_{\mathcal{D}(x)} \left[-\log p(x) \right] \qquad (8)
\end{aligned}$$

It's not immediately clear how the $-\log q(\bar{z})$ term in Eq. 7 will affect optimization of this bound. However, with some algebra (given in the supplementary material), we can see that:

$$\begin{aligned}
&\nabla_q \mathrm{KL}(q(x, z_0, ..., z_n, \bar{z}) \,||\, p(x, z_0, ..., z_n, \bar{z})) = \ ... \\
&\quad = \mathop{\mathbb{E}}_{q(x,z_0,...,z_n,\bar{z})} \left[\nabla_q \log \frac{\mathcal{D}(x)q(z_0, ..., z_n, \bar{z}|x)}{p(x, z_0, ..., z_n|\bar{z})} - \nabla_q \log \left(\mathop{\mathbb{E}}_{q} \left[q(\bar{z}|\hat{x}, \hat{z}_0, ..., \hat{z}_n) \right] \right) \right] \\
&\quad = \mathop{\mathbb{E}}_{q(x,z_0,...,z_n,\bar{z})} \left[\nabla_q \log \frac{q(z_0, ..., z_n|x)}{p(x, z_0, ..., z_n|\bar{z})} \right] \qquad (9)
\end{aligned}$$

Here, we use ∇_q to indicate differentiation w.r.t. the parameters of q, and we assume that the "reparametrization trick" is used to construct $q(x, z_0, ..., z_n, \bar{z})$ using $\mathcal{D}(x)$ followed by a product of conditionals.[2]

Thus, we can minimize a KL-based bound on $\mathbb{E}_{\mathcal{D}(x)}[-\log p(x)]$ by repeating two steps:

1. Sample $(x, z_0, ..., z_n, \bar{z})$ from $q(x, z_0, ..., z_n, \bar{z}) \triangleq \mathcal{D}(x)q(z_0, ..., z_n|x)q(\bar{z}|z_0, ..., z_n, x)$.
2. Take a step to reduce: $\log q(z_0, ..., z_n|x) - \log p(x, z_0, ..., z_n|\bar{z})$.

Most importantly, the "non-parametric" definition of $q(\bar{z})$ and $p(\bar{z})$ in Eq. 6 causes gradients from distributions over $\bar{z}$ to disappear from the optimization process. However, due to this "non-parametric" definition, the bound may become degenerate.[3] Nonetheless, we can still generate samples from $\mathcal{D}$ by starting with a single $x \sim \mathcal{D}$ and then alternating between sampling $\bar{z} \sim q(z_0, ..., z_n, \bar{z}|x)$ and $x \sim p(x, z_0, ..., z_n|\bar{z})$. If $\mathrm{KL}(q \,||\, p) = 0$, this process will sample from $p(x) = \mathcal{D}(x)$ because the marginals over x and $\bar{z}$, and the conditionals $x|\bar{z}$ and $\bar{z}|x$, must be the same in p and q for $\mathrm{KL}(q \,||\, p) = 0$ to hold.[4]

From this point of view, when training a denoising auto-encoder, one is minimizing the divergence $\mathrm{KL}(q \,||\, p)$ between the encoder distribution q and decoder distribution p. The prior $p(\bar{z})$ in the decoder is defined non-parametrically by convolving the encoder with the data distribution $\mathcal{D}$, which also ensures that the encoder joint $q(x, z_0, ..., z_n, \bar{z})$ has marginal $q(x) = \mathcal{D}(x)$. The required entropy maximization in the encoder (see gradients in Eq. 9) is obtained by construction, by adding noise to the encoder. Using a noisy encoder can also help to avoid degenerate posteriors $p(\bar{z}|x)$, which are *not* prevented by the non-parametric prior $p(\bar{z})$.

[2] The reparametrization trick conveniently routes q's gradients around the sampling required for the expectations.

[3] The prior $p(\bar{z})$ can partition into islands of mass, with each island generating a particular point in the training set.

[4] Additional assumptions – minor compared to $\mathrm{KL}(q \,||\, p) = 0$ – are required for this to be entirely correct [3, 4].

3 Reweighted Wake-Sleep and IWAEs use NIS Estimates of $\mathrm{KL}(q \,||\, p)$

We now show that the Importance Weighted Auto-encoder objective described in [6] is equivalent to stochastic variational inference with a proposal distribution corrected towards the true posterior via normalized importance sampling. This objective has also appeared as a reweighted form of Wake-Sleep [8] – first in [5] and later in [1].[5] We assume distributions $q(x, z)$ and $p(x, z)$ over variables x and z, where we can easily sample and evaluate $q(z|x)$, $p(x|z)$, and $p(z)$. We procedurally define a distribution $q_p^k(z|x)$ by drawing k independent samples $\{z_1, ..., z_k\}$ from $q(z|x)$, and then resampling from $\{z_1, ..., z_k\}$ in proportion to the normalized importance sampling weights $\{w_1, ..., w_k\}$ for sampling $p(z|x)$ via $q(z|x)$. These weights are:

$$w_i = \frac{\frac{p(z_i|x)}{q(z_i|x)}}{\sum_{j=1}^k \frac{p(z_j|x)}{q(z_j|x)}} = \frac{\frac{p(x|z_i)p(z_i)}{q(z_i|x)}}{\sum_{j=1}^k \frac{p(x|z_j)p(z_j)}{q(z_j|x)}} \tag{10}$$

We can now write a KL-based bound on $\log p(x)$, using samples from the meta distribution $q_p^k(z|x)$:

$$\log p(x) \geq \mathop{\mathbb{E}}_{(z_i, w_i) \sim q_p^k(z_i|x)} \left[\log \frac{p(x|z_i)p(z_i)}{w_i q(z_i|x)} \right] \tag{11}$$

Because sampling w_i from q_p^k requires sampling $k-1$ other zs from $q(z|x)$, we might as well sample k zs at a time and then analytically marginalize over the resampling from $\{z_1, ..., z_k\}$. This produces the bound:

$$\log p(x) \geq \mathop{\mathbb{E}}_{\{z_1, ..., z_k\} \sim q(z|x)} \left[\sum_{i=1}^k w_i \log \frac{p(x|z_i)p(z_i)}{w_i q(z_i|x)} \right] \tag{12}$$

where $\{z_1, ..., z_k\} \sim q(z|x)$ indicates independently sampling k zs from $q(z|x)$, and the w_i are computed according to Eq. 10. Now, consider the log-ratio in Eq. 12. For any x_i/w_i, we can simplify this to:

$$\log \frac{p(x|z_i)p(z_i)}{w_i q(z_i|x)} = \log \frac{p(x|z_i)p(z_i)}{\left(\frac{\frac{p(x|z_i)p(z_i)}{q(z_i|x)}}{\sum_{j=1}^k \frac{p(x|z_j)p(z_j)}{q(z_j|x)}} \right) q(z_i|x)} = \log \left(\sum_{j=1}^k \frac{p(x|z_j)p(z_j)}{q(z_j|x)} \right) \tag{13}$$

where the rightmost term in Eq. 13 is the same for all z_i in the sampled set $\{z_1, ..., z_k\}$. Because we use normalized importance sampling, we know $\sum_{i=1}^k w_i = 1$, which lets us rewrite the bound in Eq. 12 as:

$$\log p(x) \geq \mathop{\mathbb{E}}_{\{z_1, ..., z_k\} \sim q(z|x)} \left[\log \left(\sum_{z_i} \frac{p(x|z_i)p(z_i)}{q(z_i|x)} \right) \right] \tag{14}$$

Thus, by optimizing the standard variational bound in Eq. 12 using samples from q_p^k – which repeatedly performs k-sample normalized importance sampling from $p(z|x)$ via $q(z|x)$ – we reproduce the optimizations described in [5, 1, 6]. Useful properties of this bound, e.g. how its tightness and variance change with k, stem directly from known properties of normalized importance sampling and the variational free-energy.

4 Discussion

In this short paper, we developed a point of view which provides a conceptually parsimonious framework in which to compare and contrast many recent approaches to structuring and training deep generative models. Describing models and training methods within this framework should make it easier to see where they truly differ, and where their apparent differences are merely superficial. Due to space constraints, we place discussion of RBMs and Contrastive Divergence [7, 9] in the supplementary material.

[5] Here, we temporarily disregard the "sleep phase q-update" described in [5]. This aspect of reweighted wake-sleep was not considered in [1]. We discuss this point further in the supplementary material.

References

[1] Jimmy Ba, Roger Grosse, Ruslan Salakhutdinov, and Brendan Frey. Learning wake-sleep recurrent attention models. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[2] Philip Bachman and Doina Precup. Data generation as sequential decision making. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[3] Yoshua Bengio, Éric Thibodeau-Laufer, Guillaume Alain, and Jason Yosinski. Deep generative stochastic networks trainable by backprop. In *International Conference on Machine Learning (ICML)*, 2014.

[4] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[5] Jörg Bornschein and Yoshua Bengio. Reweighted wake-sleep. In *International Conference on Learning Representations (ICLR)*, 2015.

[6] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted auto-encoders. *arXiv:1509.00519v1 [cs.LG]*, 2015.

[7] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.

[8] Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. The wake-sleep algorithm for unsupervised neural networks. *Science*, 268:1158–1161, 1995.

[9] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[10] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[11] Danilo Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning (ICML)*, 2014.

[12] Tim Salimans, Diederik P Kingma, and Max Welling. Markov chain monte carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning (ICML)*, 2015.

[13] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015.

5 Supplementary Material

5.1 Additional Material for Section 1 (General KL bounds)

Some discussion of how, e.g., the time-reversible stochastic process investigated in [13] fits into the KL-based bound framework is available in [2]. In the future, we plan to combine the material from [2] with material in the current paper, to present a comprehensive view of our ideas.

5.2 Additional Material for Section 2 (DAEs and GSNs)

From Section 2, we know that:

$$\begin{aligned}
&\mathrm{KL}(q(x, z_0,..., z_n, \bar{z}) \,||\, p(x, z_0, ..., z_n, \bar{z})) = \ ... \\
&\quad = \mathop{\mathbb{E}}_{q(x,z_0,...,z_n,\bar{z})} \left[\log \frac{\mathcal{D}(x) q(z_0, ..., z_n, \bar{z}|x)}{p(x, z_0, ..., z_n|\bar{z})} - \log \left(\mathop{\mathbb{E}}_{q} \left[q(\bar{z}|\hat{x}, \hat{z}_0, ..., \hat{z}_n) \right] \right) \right] \qquad (15) \\
&\quad \geq \mathop{\mathbb{E}}_{\mathcal{D}(x)} \left[\log \mathcal{D}(x) \right] + \mathop{\mathbb{E}}_{\mathcal{D}(x)} \left[-\log p(x) \right]
\end{aligned}$$

where Eq. 15 comes from our definition of:

$$\log p(\bar{z}) \triangleq \log \left(\mathop{\mathbb{E}}_{q(x,z_1,...,z_n)} \left[q(\bar{z}|x, z_0, ..., z_n) \right] \right) = \log q(\bar{z}) \qquad (16)$$

We can directly compute the gradient of $-\log \left(\mathbb{E}_q \left[q(\bar{z}|\hat{x}, \hat{z}_0, ..., \hat{z}_n) \right] \right)$ in Eq. 15 with respect to (parameters of) the distribution q. We begin by defining $y \triangleq \{z_0, ..., z_n\}$, which lets us write:

$$\begin{aligned}
\log \left(\mathop{\mathbb{E}}_{q(x,y)} \left[q(\bar{z}|x, y) \right] \right) &= \log \left(\sum_{(x,y)} q(x, y) q(\bar{z}|x, y) \right) \\
\nabla_q \log \left(\mathop{\mathbb{E}}_{q(x,y)} \left[q(\bar{z}|x, y) \right] \right) &= \nabla_q \log \left(\sum_{(x,y)} q(x, y) q(\bar{z}|x, y) \right) \\
&\quad ** \textbf{define}: f(x, y) \triangleq \log q(x, y) q(\bar{z}|x, y) \ ** \\
\nabla_q \log \left(\mathop{\mathbb{E}}_{q(x,y)} \left[q(\bar{z}|x, y) \right] \right) &= \nabla_q \log \left(\sum_{(x,y)} e^{f(x,y)} \right) \\
&= \sum_{(x,y)} \left(\frac{e^{f(x,y)}}{\sum_{(\hat{x},\hat{y})} e^{f(\hat{x},\hat{y})}} \nabla_q f(x, y) \right) \\
&= \sum_{(x,y)} \left(\frac{q(\bar{z}|x, y) q(x, y)}{\sum_{(\hat{x},\hat{y})} q(\bar{z}|\hat{x}, \hat{y}) q(\hat{x}, \hat{y})} \nabla_q \log q(\bar{z}|x, y) q(x, y) \right) \\
&= \mathop{\mathbb{E}}_{q(x,y|\bar{z})} \left[\nabla_q \log q(\bar{z}|x, y) + \nabla_q \log q(x, y) \right] \qquad (17)
\end{aligned}$$

Now, we can write the gradient of the full objective in Eq. 15 with respect to q as:

$$
\begin{aligned}
&\nabla_q \,\mathrm{KL}(q(x,y,\bar{z}) \,||\, p(x,y,\bar{z})) = \ ... \\
&= \mathop{\mathbb{E}}_{q(x,y,\bar{z})}\left[\nabla_q \log \frac{q(y,\bar{z}|x)}{p(x,y|\bar{z})} - \nabla_q \log\left(\mathop{\mathbb{E}}_{q(\hat{x},\hat{y})}\left[q(\bar{z}|\hat{x},\hat{y})\right]\right)\right] \\
&= \mathop{\mathbb{E}}_{q(x,y,\bar{z})}\left[\nabla_q \log \frac{q(y|x)q(\bar{z}|x,y)}{p(x,y|\bar{z})} - \mathop{\mathbb{E}}_{q(\hat{x},\hat{y}|\bar{z})}\left[\nabla_q \log q(\bar{z}|\hat{x},\hat{y}) + \nabla_q \log q(\hat{x},\hat{y})\right]\right] \\
&= \mathop{\mathbb{E}}_{q(\bar{z})}\left[\mathop{\mathbb{E}}_{q(x,y|\bar{z})}\left[\nabla_q \log q(y|x) + \nabla_q q(\bar{z}|x,y) - \nabla_q \log p(x,y|\bar{z})\right]\right] - \qquad (18) \\
&\qquad \mathop{\mathbb{E}}_{q(\bar{z})}\left[\mathop{\mathbb{E}}_{q(x,y|\bar{z})}\left[\nabla_q \log q(\bar{z}|x,y) + \nabla_q \log q(x,y)\right]\right] \qquad (19) \\
&= \mathop{\mathbb{E}}_{q(\bar{z})}\left[\mathop{\mathbb{E}}_{q(x,y|\bar{z})}\left[\nabla_q \log q(y|x) - \nabla_q \log p(x,y|\bar{z})\right]\right] \qquad (20)
\end{aligned}
$$

Here, the $\nabla_q \log q(\bar{z}|x,y)$ terms in Eqns. 18 and 19 cancel each other, and the $\nabla_q \log q(x,y)$ term in Eq. 19 cancels itself because it appears in an expectation over $q(x,y)$. The term $\nabla_q \log p(x,y|\bar{z})$ in Eq. 20 is relevant due to the influence of q on the values of y and z (we use $q(x) \triangleq \mathcal{D}(x)$, so $\partial x/\partial q = 0$).

5.3 Additional Material for Section 3 (NIS and IWAEs)

For practical reasons, one may prefer to optimize an alternative objective to Eq. 14:

$$
\log p(x) \geq \mathop{\mathbb{E}}_{\{z_1,...,z_k\}\sim q(z|x)}\left[\sum_{i=1}^{k} \frac{\frac{p(x|z_i)p(z_i)}{q(z_i|x)}}{\sum_{j=1}^{k}\frac{p(x|z_j)p(z_j)}{q(z_j|x)}} \log\left(\frac{p(x|z_i)p(z_i)}{q(z_i|x)}\right)\right] \qquad (21)
$$

which has the same gradient w.r.t. p and q as Eq. 14 when we treat the normalized importance weights as constant w.r.t. p and q. The difference between the values of Eq. 14 and Eq. 21 is actually just the entropy of the discrete resampling distribution over the set $\{x_1, ..., x_k\}$. We now verify these properties.

Gradient equivalence – with $f_i \triangleq \log \frac{p(x|z_i)p(z_i)}{q(z_i|x)}$:

$$
\begin{aligned}
\text{** Eq. 14 **} \quad \nabla_q \log\left(\sum_{i=1}^{k} e^{f_i}\right) &= \sum_{i=1}^{k}\left(\frac{e^{f_i}}{\sum_{j=1}^{k} e^{f_j}} \nabla_q \log\left(e^{f_i}\right)\right) \quad \text{** Eq. 21 **} \\
\frac{1}{\sum_{j=1}^{k} e^{f_j}} \nabla_q \left(\sum_{i=1}^{k} e^{f_i}\right) &= \sum_{i=1}^{k}\left(\frac{e^{f_i}}{\sum_{j=1}^{k} e^{f_j}} \nabla_q \log\left(e^{f_i}\right)\right) \\
\sum_{i=1}^{k} \frac{\nabla_q e^{f_i}}{\sum_{j=1}^{k} e^{f_j}} &= \sum_{i=1}^{k}\left(\frac{e^{f_i}}{\sum_{j=1}^{k} e^{f_j}} \frac{1}{e^{f_i}} \nabla_q e^{f_i}\right) \\
\sum_{i=1}^{k} \frac{\nabla_q e^{f_i}}{\sum_{j=1}^{k} e^{f_j}} &= \sum_{i=1}^{k} \frac{\nabla_q e^{f_i}}{\sum_{j=1}^{k} e^{f_j}} = \sum_{i=1}^{k}\left(\frac{e^{f_i}}{\sum_{j=1}^{k} e^{f_j}} \nabla_q f_i\right) \qquad (22)
\end{aligned}
$$

Entropy difference – with $f_i \triangleq \log \frac{p(x|z_i)p(z_i)}{q(z_i|x)}$ and with $E(w_1, ..., w_k)$ as the *resampling entropy*:

$$\log\left(\sum_{i=1}^{k} e^{f_i}\right) - E(w_1, ..., w_k) = \sum_{i=1}^{k}\left(\frac{e^{f_i}}{\sum_{j=1}^{k} e^{f_j}} \log\left(e^{f_i}\right)\right)$$

$$-E(w_1, ..., w_k) = \sum_{i=1}^{k}\left(\frac{e^{f_i}}{\sum_{j=1}^{k} e^{f_j}}\left(\log\left(e^{f_i}\right) - \log\left(\sum_{j=1}^{k} e^{f_j}\right)\right)\right)$$

$$E(w_1, ..., w_k) = -\sum_{i=1}^{k}\left(\frac{e^{f_i}}{\sum_{j=1}^{k} e^{f_j}} \log \frac{e^{f_i}}{\sum_{j=1}^{k} e^{f_j}}\right) \tag{23}$$

By *resampling entropy*, we mean the entropy of the resampling distribution over $\{z_1, ..., z_k\}$.

5.4 A Discussion of Contrastive Divergence and RBMs

Given an RBM with parameters w, define q as the RBM's Gibbs chain initialized from distribution $\mathcal{D}(x_0)$ over x_0 and run for k steps, and define p as the RBM's Gibbs chain initialized from distribution $p(x_k)$ over x_k and run for k steps. Also define the k-step trajectory $\tau \triangleq \{x_0, z_0, ..., x_k\}$ generated by running q starting from $x_0 \sim \mathcal{D}(x_0)$. With these definitions we write:

$$\mathrm{KL}(q\,||\,p) = \mathop{\mathbb{E}}_{q(\tau)}\left[\log \frac{q(\tau)}{p(\tau)}\right] \tag{24}$$

$$= \sum_{\tau} q(\tau) \log \frac{q(\tau)}{p(\tau)} \tag{25}$$

$$\nabla_w \mathrm{KL}(q\,||\,p) = \sum_{\tau}\left(\nabla_w q(\tau) \log \frac{q(\tau)}{p(\tau)} + q(\tau)\nabla_w \log \frac{q(\tau)}{p(\tau)}\right) \tag{26}$$

$$= \sum_{\tau}\left(q(\tau)\nabla_w \log q(\tau) \log \frac{q(\tau)}{p(\tau)} + q(\tau)\nabla_w \log \frac{q(\tau)}{p(\tau)}\right) \tag{27}$$

$$= \sum_{\tau} q(\tau)\left(\nabla_w \log q(\tau) \log \frac{q(\tau)}{p(\tau)} + \nabla_w \log \frac{q(\tau)}{p(\tau)}\right) \tag{28}$$

$$= \mathop{\mathbb{E}}_{q(\tau)}\left[\nabla_w \log q(\tau) \log \frac{q(\tau)}{p(\tau)} + \nabla_w \log \frac{q(\tau)}{p(\tau)}\right] \tag{29}$$

$$= \mathop{\mathbb{E}}_{\mathcal{D}(x)}\left[\mathop{\mathbb{E}}_{q(\tau_x|x)}\left[\nabla_w \log q(\tau_x|x)\left(\log \frac{q(\tau_x|x)}{p(\tau_x|x)} + \log \frac{\mathcal{D}(x)}{p(x)}\right)\right]\right] +$$

$$\mathop{\mathbb{E}}_{q(\tau)}\left[\nabla_w \log \frac{q(\tau)}{p(\tau)}\right] \quad \text{** **note:** } \log \frac{\mathcal{D}(x)}{p(x)} \text{ is like a baseline in RL **} \tag{30}$$

$$= \mathop{\mathbb{E}}_{\mathcal{D}(x)}\left[\mathop{\mathbb{E}}_{q(\tau_x|x)}\left[\nabla_w \log q(\tau_x|x) \log \frac{q(\tau_x|x)}{p(\tau_x|x)}\right]\right] + \mathop{\mathbb{E}}_{q(\tau)}\left[\nabla_w \log \frac{q(\tau)}{p(\tau)}\right] \tag{31}$$

We get from Eq. 29 to Eq. 30 by the definition of $q(\tau) \triangleq \mathcal{D}(x)q(\tau_x|x)$, and by the assumption that $\nabla_w \log \mathcal{D}(x) = 0$. Note that the first expectation in Eq. 31 goes to 0 as q runs for steps $k \to \infty$. I.e. for an infinitely deep encoder-decoder pair q/p with shared weights between models and layers, the encoder samples from the exact posterior of the decoder, when the decoder's prior $p(x_k)$ is defined as the marginal

over x in the RBM's Gibbs chain. A proof of this statement is given in [9], by Hinton et al. We use w to indicate the RBM's parameter matrix, and we elide bias terms to reduce notational clutter.

Analogous to our treatment of denoising auto-encoders, we consider a prior given by:[6]

$$p(x_k) \triangleq \sum_{\tau_{:k}} \mathcal{D}(x_0) q(z_0|x_0) q(x_1|z_0) q(z_1|x_1) \dots q(x_k|z_{k-1}) \tag{32}$$

where $\tau_{:k}$ indicates the truncated trajectory $\tau_{:k} \triangleq \{x_0, z_0, ..., x_{k-1}, z_{k-1}\}$. This is just the marginal $q(x_k)$ in the trajectory distribution $q(\tau)$. With this definition of $p(x_k)$, we can see that:

$$\log \frac{q(\tau)}{p(\tau)} = \log \frac{q(x_k) q(x_0, z_0, ..., x_{k-1}, z_{k-1}|x_k)}{p(x_k) p(z_{k-1}|x_k) p(x_{k-1}|z_{k-1}) \dots p(x_0|z_0)} = \log \frac{q(x_k) q(x_0, z_0, ..., x_{k-1}, z_{k-1}|x_k)}{p(x_k) p(x_0, z_0, ..., z_{k-1}|x_k)}$$

From our work with denoising auto-encoders in Section 2, we know that this becomes:

$$\log \frac{\mathcal{D}(x_0) q(z_0, ..., x_{k-1}, z_{k-1}, x_k|x_0)}{q(x_k) p(x_0, z_0, ..., z_{k-1}|x_k)} = \log \frac{\mathcal{D}(x_0)}{q(x_k)} + \log \frac{q(z_0, ..., x_{k-1}, z_{k-1}, x_k|x_0)}{p(x_0, z_0, ..., z_{k-1}|x_k)} \tag{33}$$

We can thus write:

$$\begin{aligned} \nabla_w \log \frac{q(\tau)}{p(\tau)} &= \nabla_w \log \frac{\mathcal{D}(x_0)}{q(x_k)} + \nabla_w \log \frac{q(z_0, ..., x_{k-1}, z_{k-1}, x_k|x_0)}{p(x_0, z_0, ..., z_{k-1}|x_k)} \\ &= -\nabla_w \log q(x_k) + \nabla_w \log \frac{q(z_0, ..., x_{k-1}, z_{k-1}, x_k|x_0)}{p(x_0, z_0, ..., z_{k-1}|x_k)} \end{aligned} \tag{34}$$

Using the gradients in Eq. 34, we can rewrite the full KL objective's gradients from Eq. 31 as:

$$\begin{aligned} &\nabla_w \mathrm{KL}(q \,||\, p) = ... \\ &\mathop{\mathbb{E}}_{\mathcal{D}(x)} \left[\mathop{\mathbb{E}}_{q(\tau_x|x)} \left[(\nabla_w \log q(\tau_x|x)) \log \frac{q(\tau_x|x)}{p(\tau_x|x)} \right] \right] + \mathop{\mathbb{E}}_{q(\tau)} \left[\nabla_w \log \frac{q(z_0, x_1, ..., , x_k|x_0)}{p(x_0, z_0, ..., z_{k-1}|x_k)} \right] \end{aligned} \tag{35}$$

where the gradient term $\mathbb{E}_{q(\tau)}[-\nabla_w \log q(x_k)]$ from Eq. 34 has disappeared through self-cancellation. This cancellation occurs because $x_k \in \{x_0, z_0, ..., x_k\} \sim q(\tau)$ is sampled according to the marginal $q(x_k)$.

We can now interpret contrastive divergence and basic MCMC maximum-likelihood training for RBMs as methods based on minimizing $\mathrm{KL}(q \,||\, p)$ using the gradients in Eq. 35. We begin by looking at the term:

$$\mathop{\mathbb{E}}_{q(\tau)} \left[\nabla_w \log \frac{q(z_0, x_1, ..., , x_k|x_0)}{p(x_0, z_0, ..., z_{k-1}|x_k)} \right] \tag{36}$$

With a bit of algebra – see note at the end of this section – one can show that:

$$\log \frac{q(z_0, x_1, ..., , x_k|x_0)}{p(x_0, z_0, ..., z_{k-1}|x_k)} = \log Z_{x_k} - \log Z_{x_0} \tag{37}$$

where $\log Z_{x_i}$ is the RBM *marginal* log-partition function for $x_i \in \tau \triangleq \{x_0, z_0, ..., x_k\}$. I.e.:

$$\log Z_{x_i} \triangleq \log \left(\sum_z \exp\left(-E(x_i, z)\right) \right) \tag{38}$$

[6] This distribution converges to the RBM's marginal over x_k as $k \to \infty$, which bypasses the issues with "complementary priors" faced in [9]. We will assume the RBM Gibbs chain is ergodic, which is trivial to enforce when x and z are both binary vectors.

where $E(x_i, z)$ is the RBM's energy function (parametrized by w). This quantity is minus the "free energy" for x_i in the RBM. We define the RBM energy $E(x, z)$ as:

$$E(x, z) \triangleq -x^\top w z \tag{39}$$

where x and z are binary vectors and w is an appropriately-sized matrix. Further, we can write:

$$\begin{aligned} \log Z_{x_k} - \log Z_{x_0} &= F(x_0) - F(x_k) && (40) \\ &= -\sum_{j=1}^{|z|} \log\left(1 + \exp(\tilde{z}_0^j)\right) + \sum_{j=1}^{|z|} \log\left(1 + \exp(\tilde{z}_k^j)\right) && (41) \\ \text{where} \quad \tilde{z}_i &\triangleq x_i^\top w && (42) \end{aligned}$$

and where we define $\tilde{z}_i^j$ as the j^{th} element of the "partial energy" vector $\tilde{z}_i$ in Eq. 42. Notice that Eq. 41 can be computed directly from τ, without further sampling or approximation.

Finally, taking gradients of the terms in Eq. 40, as required to compute the gradient in Eq. 36, we get:

$$\begin{aligned} \nabla_w \log Z_{x_k} - \nabla_w \log Z_{x_0} =& -\sum_{j=1}^{|z|} \nabla_w \log\left(1 + \exp(\tilde{z}_0^j)\right) + \sum_{j=1}^{|z|} \nabla_w \log\left(1 + \exp(\tilde{z}_k^j)\right) \\ =& x_k \sigma(\tilde{z}_k)^\top - x_0 \sigma(\tilde{z}_0)^\top \end{aligned} \tag{43}$$

where $\sigma(\cdot)$ indicates element-wise sigmoid and $\tilde{z}_i$ are defined as in Eq. 42. Note that we assume all vectors are columns which become rows when transposed. Eq. 43 gives a single Monte Carlo sample of the gradient used in CD-k, i.e. contrastive divergence with k-step roll-outs. Practical applications of CD-k typically perform partial marginalization of the expectation which gets written like:

$$\nabla_w \log p(x) \approx \langle v_0 h_0^\top \rangle_0 - \langle v_k h_k^\top \rangle_k \tag{44}$$

in the contrastive divergence literature. This partial marginalization, which corresponds to the gradients in Eq. 43, arises naturally from our formulation.

From the gradients in Eqns. 35, 36, and 44, we see that both contrastive divergence and full maximum likelihood optimize an RBM's parameters by performing (approximate) gradient descent on the objective $\mathrm{KL}(q \,||\, p)$. For full maximum likelihood, the gradient term in Eq. 35 given by:

$$\mathop{\mathbb{E}}_{\mathcal{D}(x)} \left[\mathop{\mathbb{E}}_{q(\tau_x|x)} \left[\nabla_w \log q(\tau_x|x) \log \frac{q(\tau_x|x)}{p(\tau_x|x)} \right] \right] \tag{45}$$

disappears, because the encoder's forward distribution $q(\tau_x|x)$ becomes equal to the decoder's posterior distribution $p(\tau_x|x)$ as $k \to \infty$ (see [9]). Contrastive divergence simply ignores this term, though it may be appreciably non-zero for typical values of k. One could potentially add the gradients (or an approximation thereof) from Eq. 45 into the standard CD-k optimization. It would be interesting to compare behavior of the training process with and without these additional gradients. The gradients themselves have a nice interpretation as a policy-gradient term that pushes the policy (i.e. $q(\tau_x|x)$) away from regions of τ_x-space that it is visiting too frequently (i.e. where $\log \frac{q(\tau_x|x)}{p(\tau_x|x)}$ is large).

End note: The equality in Eq. 37 can be found by brute force algebra on the forwards and backwards trajectory distributions in the log-ratio $\log \frac{q(\tau)}{p(\tau)}$. Briefly, all of the energies on edges in the overlapping stochastic computation graphs for the forwards/backwards trajectories cancel, and all log-partition functions on the (shared) x_i/z_i nodes in these computation graphs also cancel, except partition functions on the first and last nodes. The non-cancelling partition functions leave us with $\log \frac{q(\tau)}{p(\tau)} = \log Z_{x_k} - \log Z_{x_0}$.