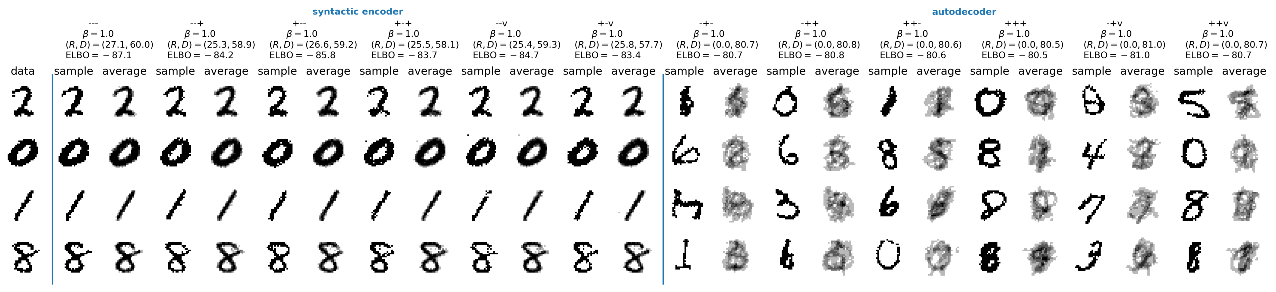
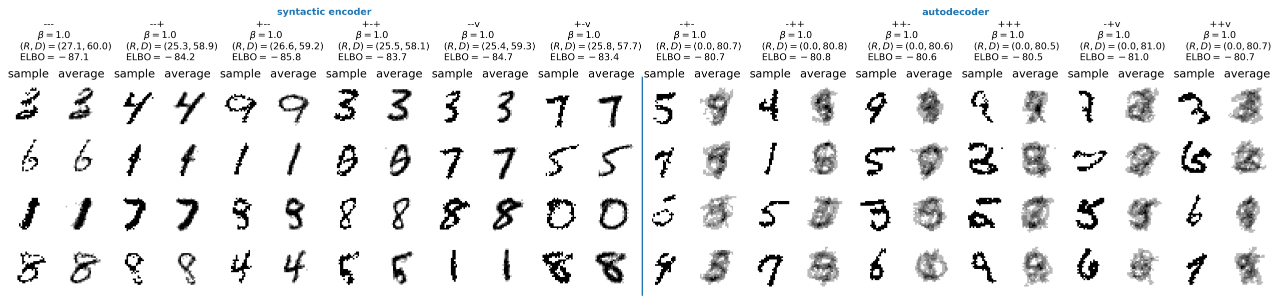
Supplemental Materials: Fixing a Broken ELBO

A. More results on Static MNIST



(a) (reconstructions)



(b) (generations)

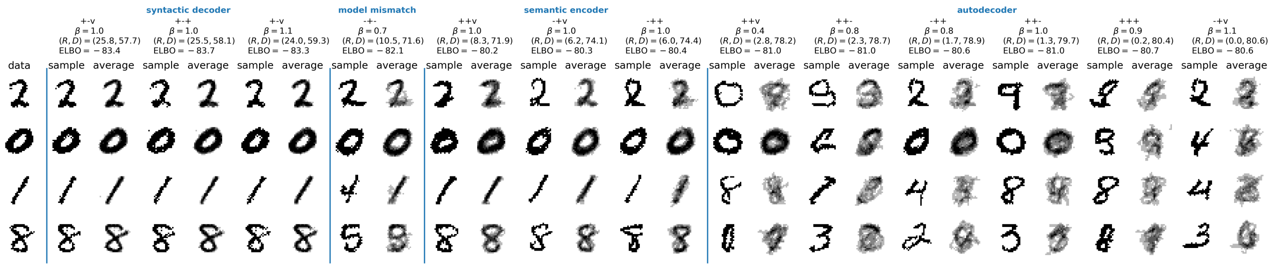
Figure 5. Traditional VAE behaviors of all model families. Note the clear separation between syntactic encoders and autodecoders, both in terms of the rate-distortion tradeoff, and in qualitative terms of sample variance. Also note that none of the 12 VAEs is a semantic encoder. Semantic encoding seems difficult to achieve at $\beta = 1$.

Figure 6 illustrates that many different architectures can participate in the optimal frontier and that we can achieve a smooth variation between the pure autodecoding models and models that encode more and more semantic and syntactic information. On the left, we see three syntactic encoders, which do a good job of capturing both the content of the digit and its style, while having variance in the decodings that seem to capture the sampling noise. On the right, we have six clear autodecoders, with very low rate and very high variance in the reconstructed or generated digit. In between are three semantic encoders, capturing the class of each digit, but showing a wide range of decoded style variation, which corresponds to the syntax of MNIST digits. Finally, between the syntactic encoders and semantic encoders lies a modeling failure, in which a weak encoder and marginal are paired with a strong decoder. The rate is sufficiently high for the decoder to reconstruct a good amount of the semantic and syntactic information, but it appears to have failed to learn to distinguish between the two.

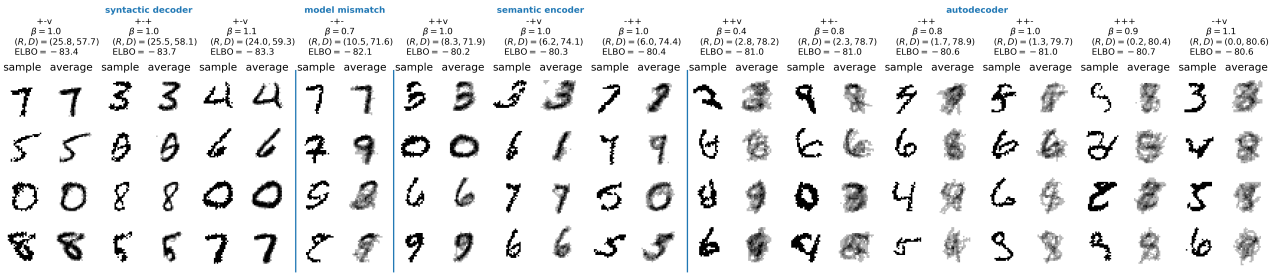
B. Results on OMNIGLOT

Figure 7 plots the RD curve for various models fit to the Omniglot dataset (Lake et al., 2015), in the same form as the MNIST results in Figure 3. Here we explored βs for the powerful decoder models ranging from 1.1 to 0.1, and βs of 0.9, 1.0, and 1.1 for the weaker decoder models. After these runs, it was clear that much of the frontier was missing, so given the already dominating performance of the powerful decoder models, those were additionally run for βs of $0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8$ to fill in the diagram.

On Omniglot, the powerful decoder models dominate over the weaker decoder models. The powerful decoder models with their autoregressive form most naturally sit at very low rates. We were able to obtain finite rates by means of KL annealing. Our best achieved ELBO was at -90.37 nats, set by the ++- model with $\beta = 1.0$ and KL annealing. This model obtains $R = 0.77, D = 89.60, ELBO = -90.37$ and is nearly auto-decoding. We found 14 models with ELBOs below 91.2 nats ranging in rates from 0.0074 nats to 10.92 nats.



(a) (reconstructions)



(b) (generations)

Figure 6. Exploring the frontier. Here we show the reconstructions (a) and generated samples (b) from a collection of runs that all lie on the frontier of realizable rate distortion tradeoffs.

Similar to Figure 4 in Figure 8 we show sample reconstruction and generated images from the same ”-+v” model family trained with KL annealing but at various βs. Just like in the MNIST case, this demonstrates that we can smoothly interpolate between auto-decoding and auto-encoding behavior in a single model family, simply by adjusting the β value.

C. Generative mutual information

Given any four distributions: $p^*(x)$ – a density over some data space X, $e(z|x)$ – a stochastic map from that data to a new representational space Z, $d(x|z)$ – a stochastic map in the reverse direction from Z to X, and $m(z)$ – some density in the Z space; we were able to find an inequality relating three functionals of these densities that must always hold. We found this inequality by deriving upper and lower bounds on the mutual information in the joint density defined by the natural *representational* path through the four distributions, $p_e(x,z) = p^*(x)e(z|x)$. Doing so naturally made us consider the existence of two other distributions $d(x|z)$ and $m(z)$. Let's consider the mutual information along this new *generative* path.

$$p_d(x,z) = m(z)d(x|z) \tag{7}$$

$$\mathrm{I_d}(X;Z) = \iint dx\, dz\, p_d(x,z) \log \frac{p_d(x,z)}{p_d(x)p_d(z)} \tag{8}$$

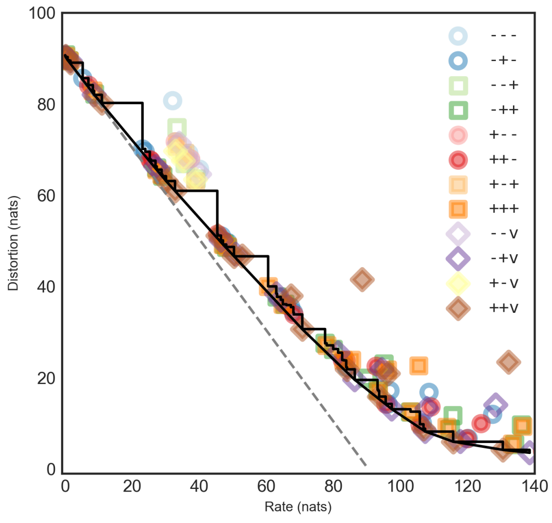
Just as before we can easily establish both a variational lower and upper bound on this mutual information. For the lower bound (proved in Section D.5), we have:

$$E \equiv \int dz\, p(z) \int dx\, p(x|z) \log \frac{q(z|x)}{p(z)} \leq \mathrm{I_d} \tag{9}$$

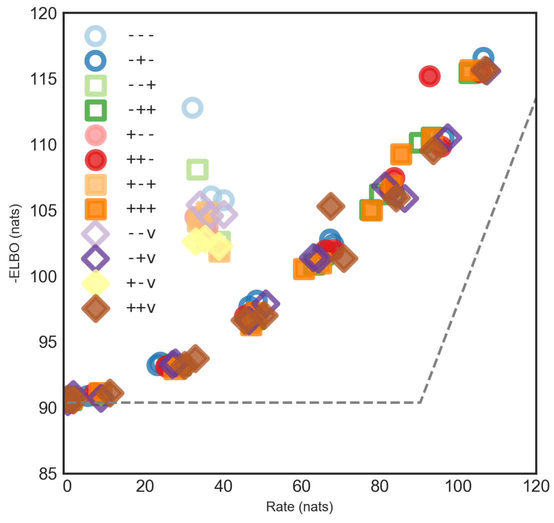
Where we need to make a variational approximation to the decoder posterior, itself a distribution mapping X to Z. Since we already have such a distribution from our other considerations, we can certainly use the encoding distribution $q(z|x)$ for this purpose, and since the bound holds for any choice it will hold with this choice. We will call this bound E since it gives the distortion as measured through the *encoder* as it attempts to encode the generated samples back to their latent representation.

We can also find a variational upper bound on the generative mutual information (proved in Section D.6):

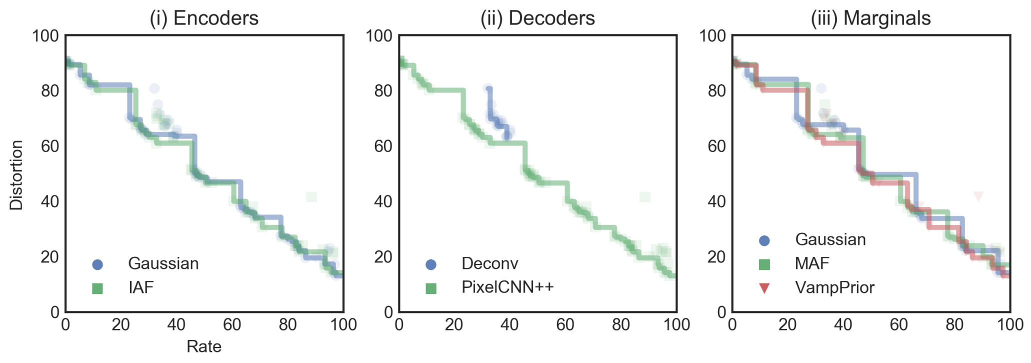
$$G \equiv \int dz\, m(z) \int dx\, d(x|z) \log \frac{d(x|z)}{q(x)} \geq \mathrm{I_d} \tag{10}$$



(a) Distortion vs Rate

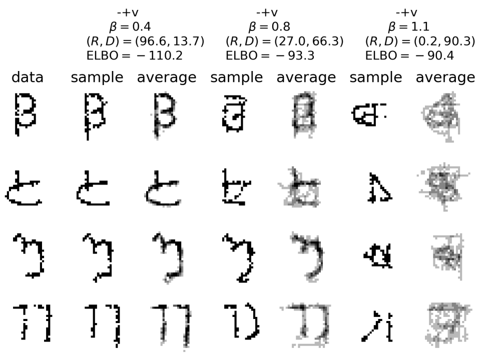


(b) -ELBO ($R + D$) vs Rate

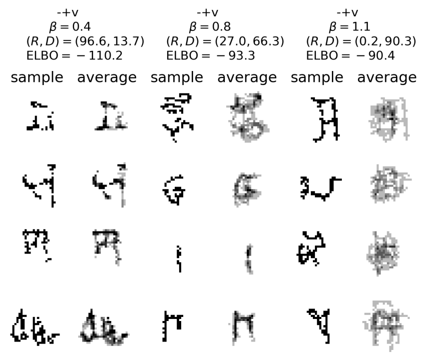


(c) Distortion vs Rate Breakout

Figure 7. Results on Omniglot. Otherwise same description as Figure 3. (a) Rate-distortion curves. (b) The same data, but on the skew axes of -ELBO = $R + D$ versus R. (c) Rate-distortion curves by encoder, decoder, and marginal family.

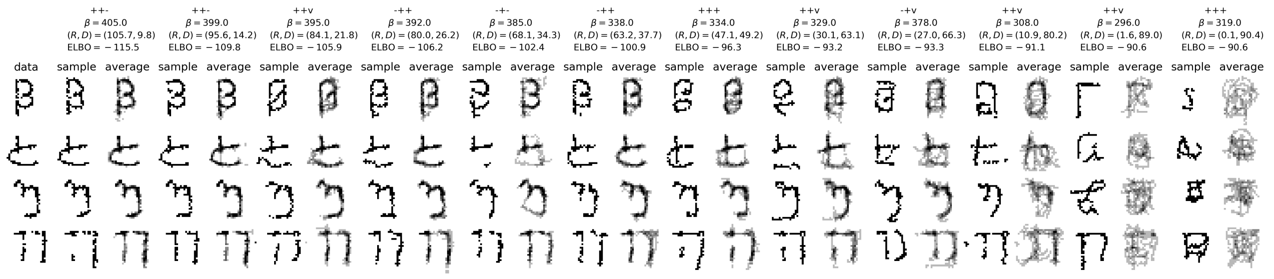


(a) Omniglot Reconstructions: $z \sim e(z|x)$, $\hat{x} \sim d(x|z)$

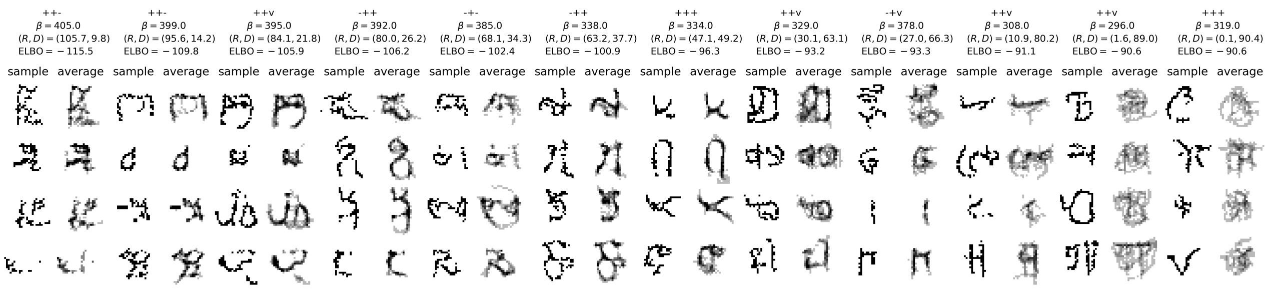


(b) Omniglot Generations: $z \sim m(z)$, $\hat{x} \sim d(x|z)$

Figure 8. We can smoothly move between pure autodecoding and autoencoding behavior in a single model family by tuning β. (a) Sampled reconstructions from the -+v model family trained at given β values. Pairs of columns show a single reconstruction and the mean of 5 reconstructions. The first column shows the input samples. (b) Generated images from the same set of models. The pairs of columns are single samples and the mean of 5 samples. See text for discussion.



(a) (reconstructions)



(b) (generations)

Figure 9. Exploring the Omniglot frontier. Here we show the reconstructions and generated samples from a whole collections of runs that all lie on the frontier of relealizable rate distortion tradeoffs. We do this primarily to illustrate that many different architectures can participate and that we can achieve a smooth variation between the pure generative models and models that encode larger and larger rates.

This time we need a variational approximation to the marginal density of our generative model, which we denote as $q(x)$. We call this bound G for the rate in the *generative* model.

Together these establish both lower and upper bounds on the generative mutual information:

$$E \leq \mathrm{I_d} \leq G. \tag{11}$$

In our early experiments, it appears as though additionally constraining or targeting values for these generative mutual information bounds is important to ensure consistency in the underlying joint distributions. In particular, we notice a tendency of models trained with the β-VAE objective to have loose bounds on the generative mutual information when β varies away from 1.

C.1. Rearranging the Representational Lower Bound

In light of the appearance of a new independent density estimate $q(x)$ in deriving our variational upper bound on the mutual information in the generative model, let's actually use that to rearrange our variational lower bound on the representational mutual information.

$$\int dx\, p^*(x) \int dz\, e(z|x) \log \frac{e(z|x)}{p^*(x)} = \int dx\, p^*(x) \int dz\, e(z|x) \log \frac{e(z|x)}{q(x)} - \int dx\, p^*(x) \log \frac{p^*(x)}{q(x)} \tag{12}$$

Doing this, we can express our lower bound in terms of two reparameterization independent functionals:

$$U \equiv \int dx\, p^*(x) \int dz\, e(z|x) \log \frac{d(x|z)}{q(x)} \tag{13}$$

$$S \equiv \int dx\, p^*(x) \log \frac{p^*(x)}{q(x)} = - \int dx\, p^*(x) \log q(x) - H \tag{14}$$

This new reparameterization couples together the bounds we derived both the representational mutual information and the generative mutual information, using $q(x)$ in both. The new function S we've described is intractable on its own, but when split into the data entropy and a cross entropy term, suggests we set a target cross entropy on our own density estimate $q(x)$ with respect to the empirical data distribution that might be finite in the case of finite data.

Together we have an equivalent way to formulate our original bounds on the representaional mutual information

$$U - S = H - D \leq I_{\text{rep}} \leq R \tag{15}$$

We believe this reparameterization offers and important and potential way to directly control for overfitting. In particular, given that we compute our objectives using a finite sample from the true data distribution, it will generically be true that $\mathrm{KL}[\hat{p}(x) \mid\mid p^*(x)] \geq 0$. In particular, the usual mode we operate in is one in which we only ever observe each example once in the training set, suggesting that in particular an estimate for this divergence would be:

$$\mathrm{KL}[\hat{p}(x) \mid\mid p^*(x)] \sim H(X) - \log N. \tag{16}$$

Early experiments suggest this offers a useful target for S in the reparameterized objective that can prevent overfitting, at least in our toy problems.

D. Proofs

D.1. Lower Bound on Representational Mutual Information

Our lower bound is established by the fact that Kullback-Leibler (KL) divergences are positive semidefinite

$$\mathrm{KL}[q(x|z) \mid\mid p(x|z)] = \int dx\, q(x|z) \log \frac{q(x|z)}{p(x|z)} \geq 0$$

which implies for any distribution $p(x|z)$:

$$\int dx\, q(x|z) \log q(x|z) \geq \int dx\, q(x|z) \log p(x|z)$$

$$
\begin{aligned}
\mathrm{I_e} = \mathrm{I_e}(X;Z) &= \iint dx\,dz\,p_e(x,z)\log\frac{p_e(x,z)}{p^*(x)p_e(z)}\\
&= \int dz\,p_e(z)\int dx\,p_e(x|z)\log\frac{p_e(x|z)}{p^*(x)}\\
&= \int dz\,p_e(z)\left[\int dx\,p_e(x|z)\log p_e(x|z) - \int dx\,p_e(x|z)\log p^*(x)\right]\\
&\geq \int dz\,p_e(z)\left[\int dx\,p_e(x|z)\log d(x|z) - \int dx\,p_e(x|z)\log p^*(x)\right]\\
&= \iint dx\,dz\,p_e(x,z)\log\frac{d(x|z)}{p^*(x)}\\
&= \int dx\,p^*(x)\int dz\,e(z|x)\log\frac{d(x|z)}{p^*(x)}\\
&= \left(-\int dx\,p^*(x)\log p^*(x)\right) - \left(-\int dx\,p^*(x)\int dz\,e(z|x)\log d(x|z)\right)\\
&\equiv H - D
\end{aligned}
$$

D.2. Upper Bound on Representational Mutual Information

The upper bound is established again by the positive semidefinite quality of KL divergence.

$$
\mathrm{KL}[q(z|x)\;||\;p(z)] \geq 0 \implies \int dz\,q(z|x)\log q(z|x) \geq \int dz\,q(z|x)\log p(z)
$$

$$
\begin{aligned}
\mathrm{I_e} = \mathrm{I_e}(X;Z) &= \iint dx\,dz\,p_e(x,z)\log\frac{p_e(x,z)}{p^*(x)p_e(z)}\\
&= \iint dx\,dz\,p_e(x,z)\log\frac{e(z|x)}{p_e(z)}\\
&= \iint dx\,dz\,p_e(x,z)\log e(z|x) - \iint dx\,dz\,p_e(x,z)\log p_e(z)\\
&= \iint dx\,dz\,p_e(x,z)\log e(z|x) - \int dz\,p_e(z)\log p_e(z)\\
&\leq \iint dx\,dz\,p_e(x,z)\log e(z|x) - \int dz\,p_e(z)\log m(z)\\
&= \iint dx\,dz\,p_e(x,z)\log e(z|x) - \iint dx\,dz\,p_e(x,z)\log m(z)\\
&= \iint dx\,dz\,p_e(x,z)\log\frac{e(z|x)}{m(z)}\\
&= \int dx\,p^*(x)\int dz\,e(z|x)\log\frac{e(z|x)}{m(z)} \equiv R
\end{aligned}
$$

D.3. Optimal Marginal for Fixed Encoder

Here we establish that the optimal marginal approximation $p(z)$, is precisely the marginal distribution of the encoder.

$$
R \equiv \int dx\,p^*(x)\int dz\,e(z|x)\log\frac{e(z|x)}{m(z)}
$$

Consider the variational derivative of the rate with respect to the marginal approximation:

$$
m(z) \to m(z) + \delta m(z) \qquad \int dz\,\delta m(z) = 0
$$

$$\begin{aligned}
\delta R &= \int dx\, p^*(x) \int dz\, e(z|x) \log \frac{e(z|x)}{m(z) + \delta m(z)} - R \\
&= \int dx\, p^*(x) \int dz\, e(z|x) \log \left(1 + \frac{\delta m(z)}{m(z)}\right) \\
&\sim \int dx\, p^*(x) \int dz\, e(z|x) \frac{\delta m(z)}{m(z)}
\end{aligned}$$

Where in the last line we have taken the first order variation, which must vanish if the total variation is to vanish. In particular, in order for this variation to vanish, since we are considering an arbitrary $\delta m(z)$, except for the fact that the integral of this variation must vanish, in order for the first order variation in the rate to vanish it must be true that for every value of x, z we have that:

$$m(z) \propto p^*(x) e(z|x),$$

which when normalized gives:

$$m(z) = \int dx\, p^*(x) e(z|x),$$

or that the marginal approximation is the true encoder marginal.

D.4. Optimal Decoder for Fixed Encoder

Next consider the variation in the distortion in terms of the decoding distribution with a fixed encoding distribution.

$$d(x|z) \to d(x|z) + \delta d(x|z) \qquad \int dx\, d(x|z) = 0$$

$$\begin{aligned}
\delta D &= -\int dx\, p^*(x) \int dz\, e(z|x) \log(d(x|z) + \delta d(x|z)) - D \\
&= -\int dx\, p^*(x) \int dz\, e(z|x) \log \left(1 + \frac{\delta d(x|z)}{d(x|z)}\right) \\
&\sim -\int dx\, p^*(x) \int dz\, e(z|x) \frac{\delta d(x|z)}{d(x|z)}
\end{aligned}$$

Similar to the section above, we took only the leading variation into account, which itself must vanish for the full variation to vanish. Since our variation in the decoder must integrate to 0, this term will vanish for every x, z we have that:

$$d(x|z) \propto p^*(x) e(z|x),$$

when normalized this gives:

$$d(x|z) = e(z|x) \frac{p^*(x)}{\int dx\, p^*(x) e(z|x)}$$

which ensures that our decoding distribution is the correct posterior induced by our data and encoder.

D.5. Lower bound on Generative Mutual Information

The lower bound is established as all other bounds have been established, with the positive semidefiniteness of KL divergences.

$$\mathrm{KL}[d(z|x) \mid\mid q(z|x)] = \int dz\, d(z|x) \log \frac{d(z|x)}{q(z|x)} \geq 0$$

which implies for any distribution $q(z|x)$:

$$\int dz\, d(z|x) \log d(z|x) \geq \int dz\, d(z|x) \log q(z|x)$$

$$\begin{aligned}
I_{\text{gen}} = I_{\text{gen}}(X;Z) &= \iint dx\, dz\, p_d(x,z) \log \frac{p_d(x,z)}{p_d(x)p_d(z)} \\
&= \int dx\, p_d(x) \int dz\, p_d(z|x) \log \frac{p_d(z|x)}{m(z)} \\
&= \int dx\, p_d(x) \left[\int dz\, p_d(z|x) \log p_d(z|x) - \int dz\, p_d(z|x) \log m(z) \right] \\
&\geq \int dx\, p_d(x) \left[\int dz\, p_d(z|x) \log e(z|x) - \int dz\, p_d(z|x) \log m(z) \right] \\
&= \iint dx\, dz\, p_d(x,z) \log \frac{e(z|x)}{m(z)} \\
&= \int dz\, m(z) \int dx\, d(x|z) \log \frac{e(z|x)}{m(z)} \\
&\equiv E
\end{aligned}$$

D.6. Upper Bound on Generative Mutual Information

The upper bound is establish again by the positive semidefinite quality of KL divergence.

$$\text{KL}[p(x|z) \mid\mid r(x)] \geq 0 \implies \int dx\, p(x|z) \log p(x|z) \geq \int dx\, p(x|z) \log r(x)$$

$$\begin{aligned}
I_{\text{gen}} = I_{\text{gen}}(X;Z) &= \iint dx\, dz\, p_d(x,z) \log \frac{p_d(x,z)}{p_d(x)m(z)} \\
&= \iint dx\, dz\, p_d(x,z) \log \frac{d(x|z)}{p_d(x)} \\
&= \iint dx\, dz\, p_d(x,z) \log d(x|z) - \iint dx\, dz\, p_d(x,z) \log p_d(x) \\
&= \iint dx\, dz\, p_d(x,z) \log d(x|z) - \int dx\, p_d(x) \log p_d(x) \\
&\leq \iint dx\, dz\, p_d(x,z) \log d(x|z) - \int dx\, p_d(x) \log q(x) \\
&= \iint dx\, dz\, p_d(x,z) \log d(x|z) - \iint dx\, dz\, p_d(x,z) \log q(x) \\
&= \iint dx\, dz\, p_d(x,z) \log \frac{d(x|z)}{q(x)} \\
&= \int dz\, m(z) \int dx\, d(x|z) \log \frac{d(x|z)}{q(x)} \equiv G
\end{aligned}$$

E. Toy Model Details

Data generation. The true data generating distribution is as follows. We first sample a latent binary variable, $z \sim \text{Ber}(0.7)$, then sample a latent 1d continuous value from that variable, $h|z \sim \mathcal{N}(h|\mu_z, \sigma_z)$, and finally we observe a discretized value, $x = \text{discretize}(h; \mathcal{B})$, where $\mathcal{B}$ is a set of 30 equally spaced bins. We set μ_z and σ_z such that $R^* \equiv \text{I}(x; z) = 0.5$ nats, in the true generative process, representing the ideal rate target for a latent variable model.

Model details. We choose to use a discrete latent representation with $K = 30$ values, with an encoder of the form $e(z_i|x_j) \propto -\exp[(w_i^e x_j - b_i^e)^2]$, where z is the one-hot encoding of the latent categorical variable, and x is the one-hot encoding of the observed categorical variable. Thus the encoder has $2K = 60$ parameters. We use a decoder of the same form, but with different parameters: $d(x_j|z_i) \propto -\exp[(w_i^d x_j - b_i^d)^2]$. Finally, we use a variational marginal, $m(z_i) = \pi_i$. Given this, the true joint distribution has the form $p_e(x,z) = p^*(x)e(z|x)$, with marginal $m(z) = \sum_x p_e(x,z)$ and conditional $p_e(x|z) = p_e(x,z)/p_e(z)$.

F. Details for MNIST and Omniglot Experiments

We used the static binary MNIST dataset originally produced for (Larochelle & Murray, 2011)[5], and the Omniglot dataset from Lake et al. (2015); Burda et al. (2015).

As stated in the main text, for our experiments we considered twelve different model families corresponding to a simple and complex choice for the encoder and decoder and three different choices for the marginal.

Unless otherwise specified, all layers used a linearly gated activation function activation function (Dauphin et al., 2017), $h(x) = (W_1 x + b_2)\sigma(W_2 x + b_2)$.

F.1. Encoder architectures

For the encoder, the simple encoder was a convolutional encoder outputting parameters to a diagonal Gaussian distribution. The inputs were first transformed to be between -1 and 1. The architecture contained 5 convolutional layers, summarized in the format Conv (depth, kernel size, stride, padding), followed by a linear layer to read out the mean and a linear layer with softplus nonlinearity to read out the variance of the diagonal Gaussiann distribution.

- Input (28, 28, 1)
- Conv (32, 5, 1, same)
- Conv (32, 5, 2, same)
- Conv (64, 5, 1, same)
- Conv (64, 5, 2, same)
- Conv (256, 7, 1, valid)
- Gauss (Linear (64), Softplus (Linear (64)))

For the more complicated encoder, the same 5 convolutional layer architecture was used, followed by 4 steps of mean-only Gaussian inverse autoregressive flow, with each step's location parameters computed using a 3 layer MADE style masked network with 640 units in the hidden layers and ReLU activations.

F.2. Decoder architectures

The simple decoder was a transposed convolutional network, with 6 layers of transposed convolution, denoted as Deconv (depth, kernel size, stride, padding) followed by a linear convolutional layer parameterizing an independent Bernoulli distribution over all of the pixels:

- Input (1, 1, 64)
- Deconv (64, 7, 1, valid)
- Deconv (64, 5, 1, same)
- Deconv (64, 5, 2, same)
- Deconv (32, 5, 1, same)
- Deconv (32, 5, 2, same)

[5] https://github.com/yburda/iwae/tree/master/datasets/BinaryMNIST

- Deconv (32, 4, 1, same)
- Bernoulli (Linear Conv (1, 5, 1, same))

The complicated decoder was a slightly modified PixelCNN++ style network (Salimans et al., 2017)[6]. However in place of the original RELU activation functions we used linearly gated activation functions and used six blocks (with sizes (28×28) – (14×14) – (7×7) – (7×7) – (14×14) – (28×28)) of two resnet layers in each block. All internal layers had a feature depth of 64. Shortcut connections were used throughout between matching sized featured maps. The 64-dimensional latent representation was sent through a dense lineary gated layer to produce a 784-dimensional representation that was reshaped to $(28 \times 28 \times 1)$ and concatenated with the target image to produce a $(28 \times 28 \times 2)$ dimensional input. The final output (of size $(28 \times 28 \times 64)$) was sent through a (1×1) convolution down to depth 1. These were interpreted as the logits for a Bernoulli distribution defined on each pixel.

F.3. Marginal architectures

We used three different types of marginals. The simplest architecture (denoted (-)), was just a fixed isotropic gaussian distribution in 64 dimensions with means fixed at 0 and variance fixed at 1.

The complicated marginal (+) was created by transforming the isotropic Gaussian base distribution with 4 layers of mean-only Gaussian autoregressive flow, with each steps location parameters computed using a 3 layer MADE style masked network with 640 units in the hidden layers and relu activations. This network resembles the architecture used in Papamakarios et al. (2017).

The last choice of marginal was based on VampPrior and denoted with (v), which uses a mixture of the encoder distributions computed on a set of pseudo-inputs to parameterize the prior (Tomczak & Welling, 2017). We add an additional learned set of weights on the mixture distributions that are constrained to sum to one using a softmax function: $m(z) = \sum_{i=1}^{N} w_i e(z|\phi_i)$ where N are the number of pseudo-inputs, w are the weights, e is the encoder, and ϕ are the pseudo-inputs that have the same dimensionality as the inputs.

F.4. Optimization

The models were all trained using the β-VAE objective (Higgins et al., 2017) at various values of β. No form of explicit regularization was used. The models were trained with Adam (Kingma & Ba, 2015) with normalized gradients (Yu et al., 2017) for 200 epochs to get good convergence on the training set, with a fixed learning rate of 3×10^{-4} for the first 100 epochs and a linearly decreasing learning rate towards 0 at the 200th epoch.

Supplementary References

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance weighted autoencoders. In *ICLR*, 2015.

Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. Language modeling with gated convolutional networks. In *ICML*, 2017.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *ICLR*, 2017.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015. URL `https://arxiv.org/abs/1412.6980`.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Larochelle, H. and Murray, I. The neural autoregressive distribution estimator. In *AI/Statistics*, 2011.

Papamakarios, G., Murray, I., and Pavlakou, T. Masked autoregressive flow for density estimation. In *NIPS*. 2017.

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

[6]Original implmentation available at https://github.com/openai/pixel-cnn

Tomczak, J. M. and Welling, M. VAE with a VampPrior. *ArXiv e-prints*, 2017.

Yu, A. W., Lin, Q., Salakhutdinov, R., and Carbonell, J. Normalized Gradient with Adaptive Stepsize Method for Deep Neural Network Training. *ArXiv e-prints*, 2017.