Expressive Variational Autoencoders

John Thickstun

The Gaussian VAE parameterizes the prior $r(z)$, conditional likelihood $p(x|z)$, and posterior approximation $q(x|z)$ with with Gaussian distributions. The in-expressivity of these Gaussian models can make it difficult to capture the distribution $p(x)$; complaints about the "blurriness" of the VAE may be attributable to these assumptions. Note that many papers visualize the mean $g_\theta(\tilde{z})$ of the decoder network, rather than samples $g_\theta(\tilde{z}) + \eta$, which coupled with a Gaussian noise model on $\mathcal{X}$ could exacerbate blurriness.

PixelCNN and PixelVAE

One way to increase the expressivity of the VAE is to remove the conditional-independence assumption from the decoder distribution $p(x|z)$. In the standard Gaussian VAE, the components x_i of x are conditionally independent given the latent code z:

$$p(x|z) = \prod_{i=1}^{|\mathcal{X}|} p(x_i|z) = \prod_{i=1}^{|\mathcal{X}|} \mathcal{N}(x_i|\mu_i(z), \sigma^2). \tag{1}$$

We can remove this assumption by building a fully-autoregressive model of the decoder distribution over observations x, i.e.

$$p(x|z) = \prod_{i=1}^{|\mathcal{X}|} p(x_i|x_{<i}, z). \tag{2}$$

An auto-regressive parameterization of the conditional likelihood called PixelVAE is explored by Gulrajani et al. [2017], based on a line of work building autoregressive models called PixelCNN [van den Oord et al., 2016b,a, Salimans et al., 2017] that extends the NADE modeling perspective to images. One oddity of these models is that, in order to construct an autoregressive factorization of the like distribution over images, we need to fix a (somewhat arbitrary) ordering over pixels; the standard choice is to order the pixels from left to right, top-to-bottom, starting with the pixel in the upper-left corner of the image.

One might question whether the order matters; while any order leads to a valid factorization of the joint distribution, perhaps some factorizations would be easier to learn than others? This question was asked in the original NADE work, and the answer. There is followup work on orderless NADE [Uria et al., 2014] that learns an ensemble of factored autoregressive models, one for each possible ordering of pixels; by ensembling these models, it may be possible to construct a better model than using any particular ordering. But in practice, just picking an arbitrary ordering doesn't seem to cause too much trouble.

Two serious problems with using autoregressive likelihoods $p(x|z)$ are posterior collapse (discussed in the next section) and the computational expense of sampling from an autoregressive likelihood. Recall that sampling from an autoregressive model is slow: sampling an object of length n (e.g. $n = d \times d$ pixels of a square image) requires $O(n)$ serial calls to the model. For generating data like text, this is not so bad. But generating a high-resolution $1,024 \times 1,024$ image or $44,100$kHz audio could take hours! In contrast, a very nice property of the Gaussian VAE is that the conditional independence assumption on the likelihood means that we can generate each output $p(x_i|z)$ in parallel, given the code z. For these reasons, the PixelVAE is not the favored means through which we will improve the expressivity of the VAE.

Posterior Collapse

A serious trouble using an expressive likelihoods $p(x|z)$ in the VAE (such as an autoregressive likelihood) is a problem known as posterior collapse, wherein we learn a model that completely ignores the latent codes z. If all we wanted to do was learn a generative model, then this isn't necessarily a problem. But if we had a reason to learn a latent variable model that extracts semantically meaningful codes, then we need to worry about this.

As a simple example of this phenomenon, suppose $\mathcal{X} = \{0, 1\}$ and $p = \text{Bernoulli}(.5)$. Using the standard Gaussian VAE, we define $\mathcal{Z} = \mathbb{R}$, $\rho_Z(z) = \mathcal{N}(0, 1)$, and $q(z|x) = \mathcal{N}(f_\varphi(x), \sigma_\varphi^2(x))$. Consider a family of Bernoulli likelihoods $p_\theta(x|z) = \text{Bernoulli}(g_\theta(z))$ where $g_\theta : \mathcal{Z} \to \mathcal{X}$. Suppose we set $p_\theta(x|z) = \text{Bernoulli}(.5)$ (independent of z). Then the posterior $p_\theta(z|x)$ is just the prior ρ_Z, which is realizable by our family of posterior candidates q. Setting $q(z|x) = \mathcal{N}(0, 1)$ yields no slop in the ELBO, and we precisely model $p(x)$ while observing "posterior collapse" in the sense that the latent code z is completely ignored.

We can generalize this to categorical distributions. $\mathcal{X}$ is discrete and $p = \text{Categorical}(|\mathcal{X}|)$ is a categorical distribution over $\mathcal{X}$. Using the standard Gaussian VAE, we define $\rho_Z(z) = \mathcal{N}(0, I)$, and $q_\varphi(z|x) = \mathcal{N}(f_\varphi(x), \Sigma_\varphi(x))$. Because $\mathcal{X}$ is discrete, we can parameterize $p_\theta(x|z)$ with all possible categorical likelihoods, denoted by $p_\theta(x|z) = \text{Categorical}_{\theta,z}(|\mathcal{X}|)$. One global minimizer of the variational bound is just $p_\theta(x|z) = p(x)$, ignoring the latent variable z. Furthermore, suppose $p_\theta(x|z)$ is a global minimizer of the elbo; then the variational bound must be tight, and therefore

$$\rho_Z(z) = \sum_{x \in \mathcal{X}} p(x) p_\theta(z|x) = \sum_{x \in \mathcal{X}} p(x) q_\varphi(z|x). \tag{3}$$

Observe that the left-hand side is a Gaussian, and the right-hand side is a mixture of Gaussians. Equality holds iff all the Gaussians on the right-hand side are identical; i.e. $q_\varphi(z|x)$ is independent of x. But the posterior is independent of x iff the likelihood is independent of z; i.e. the latent codes are completely ignored.

Imposing a restrictive assumption on the likelihood, such as the conditional independence assumption, prevents collapse; because a conditionally independent family of distributions cannot model expressive dependencies between components, these dependencies must be captured by the latent codes z.

Inference Suboptimality

Another source of in-expressivity in the Gaussian VAE is our use of a Gaussian family to parameterize the posterior approximation $q_\varphi(z|x)$. We can quantify the approximation error of a variational posterior estimate $q_\varphi(z|x)$ by looking at the difference between the evidence lower bound and the marginal likelihood. The bound is tight iff $q_\varphi(z|x) = p_\theta(z|x)$, so this gap measures the cost of our approximation $q_\varphi(z|x)$. Assuming we have effectively optimized q_φ, we can view this gap as a measure of the cost of the Gaussian family approximation. Cremer et al. [2018] attempt to quantify the approximation error of the Gaussian parameterization of the posterior by measuring this gap. They find the error to be small, and conclude that the Gaussian approximation isn't a significant simplification.

But this analysis seems a bit off the mark. Because we jointly optimize the likelihood and the posterior approximation, in principle we shouldn't get hurt by limiting the family of posterior approximations $q_\varphi(z|x)$ to Gaussians: the optimization will just pick a likelihood $p_\theta(x|z)$ such that the true posterior $p_\theta(z|x)$ is well-approximated by the parametric family $q_\varphi(z|x)$. In practice, it might be difficult to find a likelihood $p_\theta(x|z)$ that satisfies both:

- The marginal $p_\theta(x) = \int_Z p_\theta(x|z)\rho_Z(z)\,dz$ is close to the true distribution $p(x)$,
- The posterior $p_\theta(z|x)$ of the chosen likelihood is approximately Gaussian.

There is tension between these criteria, so we may learn a likelihood that achieves a relatively small posterior approximation gap at the expense of a suboptimal estimate of the marginal distribution.

Normalizing Flows

One way to make the posterior approximations more expressive makes use of the normalizing flow construct [Rezende and Mohamed, 2015]. Normalizing flows are based on the pushforward principle. Suppose that $\mathbf{z}_0 \sim q_0$, where q has a tractable density, e.g. $q_0 = \mathcal{N}(0, I)$. If we define $\mathbf{z}_s = g_s(\mathbf{z}_{t-1})$ where $g_s : \mathcal{Z} \to \mathcal{Z}$, then the log-density of the distribution of $\mathbf{z}_t = g_t \circ \cdots \circ g_1(\mathbf{z}_0)$ is given by

$$\log q_t(\mathbf{z}_t) = \log q_0(\mathbf{z}_0) - \sum_{s=1}^{t} \operatorname{logdet}\left(\frac{\partial g_s(\mathbf{z}_{t-1})}{\partial \mathbf{z}_{t-1}}\right). \tag{4}$$

So long as we can compute the log-determinant of the Jacobians of g_s, we can compute the log-likelihood of observations $\mathbf{z}_t$. In general these determinants will not be tractable, and the onus is on us to design functions g_s for which this quantity is efficiently computable.

We have already seen (e.g. inverse transform sampling) that we can use pushforwards to turn simple distributions into arbitrarily rich distributions; our goal is to use leverage this capacity to construct a rich family of distributions with which to approximate the posterior $p_\theta(z|x)$. Therefore, it is in our interest to make the pushforward functions g_s as rich and flexible as possible, while also ensuring efficient calculation of the Jacobian determinants.

Inverse Autoregressive Flows

One class of functions that admits efficient calculation of Jacobian determinants is the family of inverse autoregressive functions. Recall that we can parameterize a (scalar) autoregressive model over $\mathbf{y} \in \mathbb{R}^p$ by $\mathbf{y}_k = \mu_k(\mathbf{y}_{<k}) + \sigma_k(\mathbf{y}_{<k})\boldsymbol{\varepsilon}_t$ where $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, I)$ and μ_k, σ_k are functions of elements of $\mathbf{y}$ with indices $j < k$; this the RNADE autoregressive model [Uria et al., 2013], which can be viewed as a generalization of NADE [Larochelle and Murray, 2011] to continuous-valued sequences. Alternatively RNADE can be viewed as a generalization of the classical AR(p) model to non-linear functions μ_k, σ_k of the history $\mathbf{y}_{<k}$.

An autoregressive pushforward operation can be inverted, so long as $\sigma_k > 0$:

$$\varepsilon_k = \frac{y_k - \mu_k(\mathbf{y}_{<k})}{\sigma_k(\mathbf{y}_{<k})}. \tag{5}$$

Whereas sampling from an autoregressive model costs $O(p)$ sequential sampling operations, the inverse autoregressive transformation can be computed as a parallel, vectorized operation: $\boldsymbol{\varepsilon} = (\mathbf{y} - \boldsymbol{\mu}(\mathbf{y}))/\boldsymbol{\sigma}(\mathbf{y})$. Futhermore, due to the autoregressive structure, the Jacobian of this operation is lower diagonal and

$$\operatorname{logdet}\left(\frac{\partial \boldsymbol{\varepsilon}}{\partial \mathbf{y}}\right) = -\sum_{k=1}^{p} \log \sigma_k(\mathbf{y}_{<k}). \tag{6}$$

The idea of inverse autoregressive flows [Kingma et al., 2016] is to use a sequence of inverse autoregressive functions to construct an expressive family of pushforward distributions to parameterize the posterior approximation in a VAE. Concretely, we take $\mathbf{y} = \mathbf{z}_{t-1}$, $\boldsymbol{\varepsilon} = \mathbf{z}_t$ and write

$$\mathbf{z}_t = \frac{\mathbf{z}_{t-1} - \boldsymbol{\mu}_t(\mathbf{z}_{t-1})}{\boldsymbol{\sigma}_t(\mathbf{z}_{t-1})} = \frac{\mathbf{z}_{t-1}}{\boldsymbol{\sigma}_t(\mathbf{z}_{t-1})} - \frac{\boldsymbol{\mu}_t(\mathbf{z}_{t-1})}{\boldsymbol{\sigma}_t(\mathbf{z}_{t-1})}. \tag{7}$$

Rather than directly parameterizing $\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t$, it's more convenient and numerically stable to parameterize this transformation with a pair of autoregressive neural networks $\mathbf{m}_t, \mathbf{s}_t$, with which we define $\mathbf{v}_t = \operatorname{sigmoid}(\mathbf{s}_t)$, $\boldsymbol{\sigma}_t = 1/\mathbf{v}_t$, and $\boldsymbol{\mu}_t = -\boldsymbol{\sigma}_t \odot (1 - \mathbf{v}_t) \odot \mathbf{m}_t$; it follows that

$$\mathbf{z}_t = \mathbf{v}_t \mathbf{z}_{t-1} + (1 - \mathbf{v}_t) \odot \mathbf{m}_t. \tag{8}$$

Another way to think about this construction is that using an expressive posterior approximation $\mathbf{z}_T$ parameterized by an inverse autoregressive flow is equivalent to using an autoregressive prior over the latent space using an autoregressive pushforward of $\mathbf{z}_0 \sim \mathcal{N}(0, I)$; the advantage of using the *inverse* flow is that we can take advantage of parallelism in the inverse operations. Furthermore, the additional computational costs of the autoregressive operation are pushed into the encoder network, while sampling $p_\theta(x, z)$ remains untouched from the original formulation of the VAE; not only does an autoregressive prior require serial computations, but these computations are incurred in the forward (decoder) sampling operation.

References

Chris Cremer, Xuechen Li, and David Duvenaud. Inference suboptimality in variational autoencoders. *arXiv preprint arXiv:1801.03558*, 2018. (document)

Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. Pixelvae: A latent variable model for natural images. *International Conference on Learning Representations*, 2017. (document)

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in neural information processing systems*, 2016. (document)

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *International Conference on Artificial Intelligence and Statistics*, 2011. (document)

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *International Conference on Machine Learning*, 2015. (document)

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *International Conference on Learning Representations*, 2017. (document)

Benigno Uria, Iain Murray, and Hugo Larochelle. Rnade: The real-valued neural autoregressive density-estimator. In *Advances in Neural Information Processing Systems*, 2013. (document)

Benigno Uria, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *International Conference on Machine Learning*, 2014. (document)

Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*, 2016a. (document)

Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *International Conference on Learning Representations*, 2016b. (document)