Flows and Discrete VAEs

Instructor: John Thickstun

Discussion Board: Available on Ed

Zoom Link: Available on Canvas

Instructor Contact: thickstn@cs.washington.edu

Course Webpage: https://courses.cs.washington.edu/courses/cse599i/20au/

Variational Autoencoders

- Generative model $p_\theta(x, z) = p_\theta(x|z)r(z)$. Learn $p_\theta(x) \approx p(x)$, where

$$p_\theta(x) = \mathbb{E}_{z\sim r}[p_\theta(x|z)] = \int_{\mathcal{Z}} p_\theta(x|z)r(z)\, dz.$$

- Estimate the MLE using the ELBO:

$$\hat{\theta}_{\text{mle}} \equiv \arg\max_\theta \mathop{\mathbb{E}}_{x\sim p} \log p_\theta(x) = \arg\max_\theta \sup_{q_\varphi} \mathop{\mathbb{E}}_{\substack{x\sim p \\ z\sim q_\varphi(\cdot|x)}} \log \frac{p_\theta(x, z)}{q_\varphi(z|x)}.$$

- Modeling choices: prior $r(z)$, likelihood $p_\theta(x|z)$, and proposal $q_\varphi(z|x)$.

- Construct an expressive family proposals so that the ELBO is tight.

Normalizing Flows

- Build a series of transformations of our initial proposal $\mathbf{z}_0 \sim q_\phi(\cdot|x)$.
- Let $g_s : \mathcal{Z} \to \mathcal{Z}$ and define $\mathbf{z}_t = g_t \circ \cdots \circ g_1(\mathbf{z}_0)$.
- The log-density of the pushforward distribution on $\mathbf{z}_t$ is given by

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

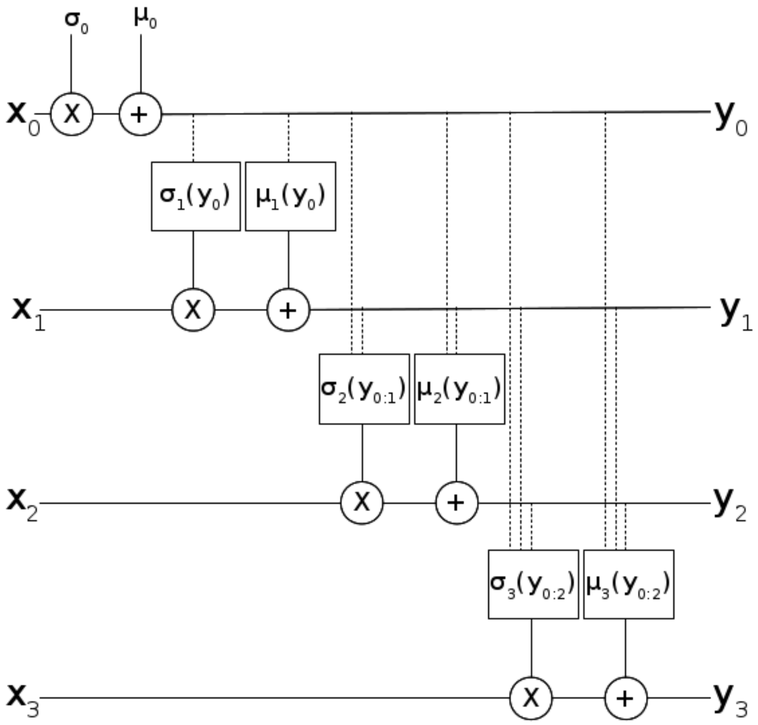
- Choose functions g_s so that $\operatorname{logdet} \left(\frac{\partial g_s(\mathbf{z}_{s-1})}{\partial \mathbf{z}_{s-1}} \right)$ is easy to calculate.

Autoregressive Model?

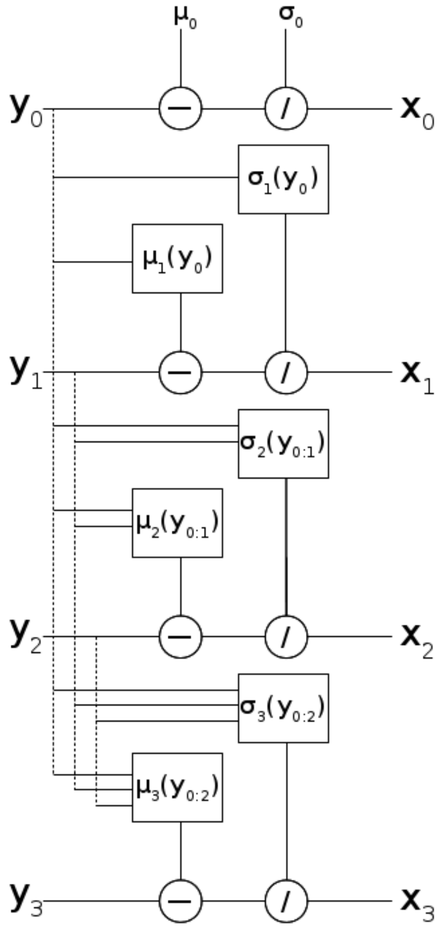
- The density of the pushforward distribution on $\mathbf{z}_t$ is directly parameterized.
- No need to accumulate densities of pushfoward distributions.
- But sampling is slow: O(p) where p is the dimensionality of the latent space.

Inverse Autoregressive Model

- Autoregressive model: serially transform $\mathbf{x} \sim \mathcal{N}(0, I)$ into sample $\mathbf{y} \sim q$.
- Inverse autoregressive model: parallel transform $\mathbf{y} \sim q$ into samples $\mathbf{x} \sim \mathcal{N}(0, I)$.
- IAF whitens AR samples.
- Figure credit: Brian Keng.



Autoregressive Transform



Inverse Autoregressive Transform

Inverse Autoregressive Flow

- Inverse autoregressive transformation: $\mathbf{z}_t = \dfrac{\mathbf{z}_{t-1} - \boldsymbol{\mu}_t(\mathbf{z}_{t-1})}{\boldsymbol{\sigma}_t(\mathbf{z}_{t-1})}$.

- Claim: $\log\det\left(\dfrac{\partial \mathbf{z}_t}{\partial \mathbf{z}_{t-1}}\right) = -\sum_{k=1}^{p} \log \boldsymbol{\sigma}_{t,k}(\mathbf{z}_{t-1,<k})$. Proof:

$$\frac{\partial \mathbf{z_t}}{\partial \mathbf{z_{t-1}}} = \begin{bmatrix} \frac{\partial z_{t,0}}{\partial z_{t-1,0}} & 0 & \cdots & 0 \\ \frac{\partial z_{t,0}}{\partial z_{t-1,1}} & \frac{\partial z_{t,1}}{\partial z_{t-1,1}} & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \frac{\partial z_{t,0}}{\partial z_{t-1,p}} & \cdots & \frac{\partial z_{t,p-1}}{\partial z_{t-1,p}} & \frac{\partial z_{t,p}}{\partial z_{t-1,p}} \end{bmatrix} = \begin{bmatrix} \frac{1}{\boldsymbol{\sigma}_{t,0}} & 0 & \cdots & 0 \\ \frac{\partial z_{t,0}}{\partial z_{t-1,1}} & \frac{1}{\boldsymbol{\sigma}_{t,1}} & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \frac{\partial z_{t,0}}{\partial z_{t-1,p}} & \cdots & \frac{\partial z_{t,p-1}}{\partial z_{t-1,p}} & \frac{1}{\boldsymbol{\sigma}_{t,p}} \end{bmatrix}.$$

Re-parameterizing IAF

- Re-write the inverse autoregressive transformation:

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

- Don't parameterize mean and deviation; instead parameterize $m_t, s_t : \mathcal{Z} \to \mathcal{Z}$:

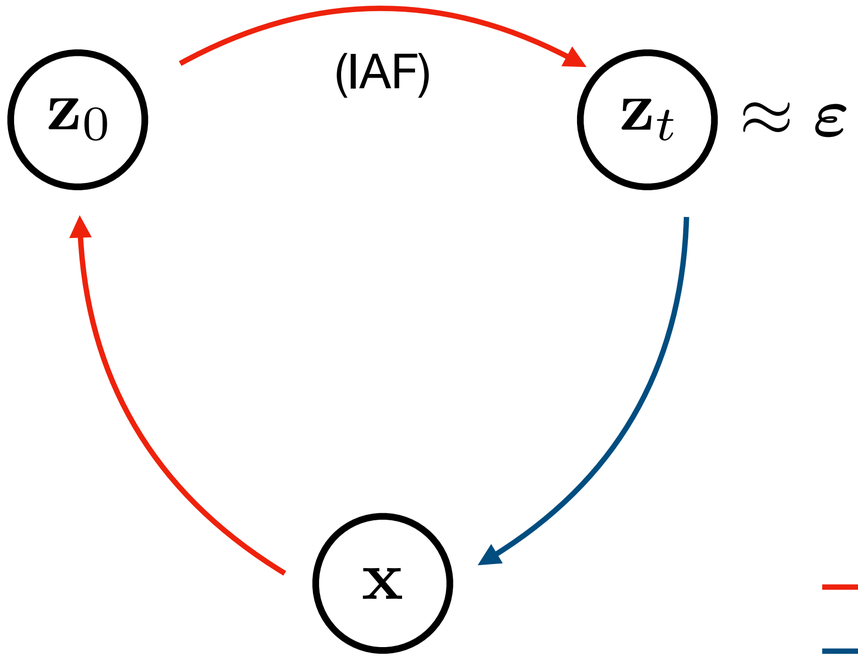
$$v_t = \text{sigmoid}(s_t), \boldsymbol{\sigma}_t = 1/v_t, \boldsymbol{\mu}_t = -\boldsymbol{\sigma}_t \odot (1 - v_t) \odot m_t.$$

- Then the IAF looks like:

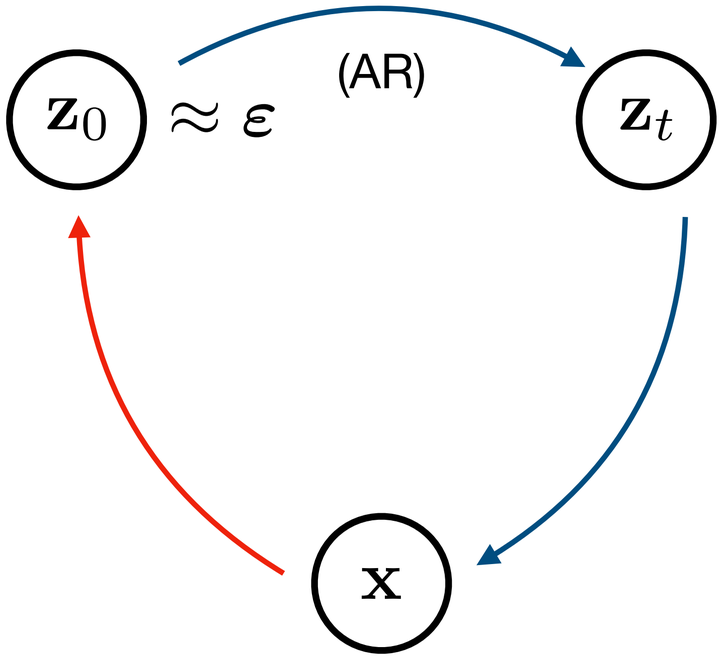
$$\mathbf{z}_t = v_t(\mathbf{z}_{t-1})\mathbf{z}_{t-1} + (1 - v_t(\mathbf{z}_{t-1})) \odot m_t(\mathbf{z}_{t-1}).$$

IAF Posterior vs AR Prior

IAF Posterior



Autoregressive Prior



Encoder

Decoder

5-Minute Break

Variational Autoencoders

- Generative model $p_\theta(x, z) = p_\theta(x|z)r(z)$. Learn $p_\theta(x) \approx p(x)$, where

$$p_\theta(x) = \mathbb{E}_{z\sim r}[p_\theta(x|z)] = \int_{\mathcal{Z}} p_\theta(x|z)r(z)\,dz.$$

- Estimate the MLE using the ELBO:

$$\hat{\theta}_{\text{mle}} \equiv \arg\max_\theta \mathop{\mathbb{E}}_{x\sim p} \log p_\theta(x) = \arg\max_\theta \sup_{q_\varphi} \mathop{\mathbb{E}}_{\substack{x\sim p \\ z\sim q_\varphi(\cdot|x)}} \log \frac{p_\theta(x, z)}{q_\varphi(z|x)}.$$

- Modeling choices: prior $r(z)$, likelihood $p_\theta(x|z)$, and proposal $q_\varphi(z|x)$.

- What if $\mathcal{Z}$ is finite, but large?

A Discrete ELBO

- The general form of the ELBO optimization:

$$\hat{\theta}_{\text{mle}} = \arg\max_{\theta} \sup_{q} \mathop{\mathbb{E}}_{\substack{x \sim p \\ z \sim q_\phi(\cdot|x)}} \Big[\log p_\theta(x|z) - D(q_\phi(z|x) \parallel r(z)) \Big].$$

- For discrete latent spaces, use a uniform prior $r(z) = 1/K$ where $K = |\mathcal{Z}|$.

- The divergence term becomes: $D(q_\phi(z|x) \parallel r(z)) = \log(K) - H(q_\phi(z|x))$.

- Likelihood term is unchanged. Combined variational objective is

$$\mathcal{L}(x, \theta, \phi) = \mathop{\mathbb{E}}_{z \sim q_\phi(\cdot|x)} [\log p_\theta(x|z)] + H(q_\phi(z|x)) - \log(K).$$

How to Get a Gradient?

- Want to estimate a gradient with respect to ϕ of the discrete ELBO:

$$\mathcal{L}(x, \theta, \phi) = \mathop{\mathbb{E}}_{z \sim q_\phi(\cdot|x)} [\log p_\theta(x|z)] + H(q_\phi(z|x)) - \log(K).$$

- Can't just use the re-parameterization trick: gradients are zero!
- No closed form for the entropy term; we'll need to estimate that.
- Need to estimate the gradient of the entropy too.

The Policy Gradient Theorem

- Also known as REINFORCE:

$$\begin{aligned}
\nabla_\phi \mathop{\mathbb{E}}_{z\sim q_\phi(\cdot|x)} \left[\log p_\theta(x|z)\right] &= \sum_{z\in\mathcal{Z}} \log p_\theta(x|z)\nabla_\phi q_\phi(z|x) \\
&= \sum_{z\in\mathcal{Z}} \log p_\theta(x|z) q_\phi(z|x)\nabla_\phi \log q_\phi(z|x) \\
&= \mathop{\mathbb{E}}_{z\sim q_\phi(\cdot|x)} \left[\log p_\theta(x|z)\nabla_\phi \log q_\phi(z|x)\right] \\
&= \mathop{\mathbb{E}}_{z\sim q_\phi(\cdot|x)} \left[\frac{\log p_\theta(x|z)}{q_\phi(z|x)}\nabla_\phi q_\phi(z|x)\right].
\end{aligned}$$

- A similar trick applies to the entropy term. But the variance is high!

Gumbel Re-parameterization

- Suppose $\mathcal{Z}$ has some factorized structure, e.g. $\mathcal{Z} = \{1, \ldots, d\}^k (K = d^k)$.
- Homework 1: if $g_{i,1}, \ldots, g_{i,d} \sim \text{Gumbel}(0, 1)$ then $z \sim q$ where

$$z_i = \arg\max_u \left[\log q_i(u) + g_{i,u} \right].$$

- We can use this to re-parameterize the ELBO. More generally,

$$\mathop{\mathbb{E}}_{z \sim q_\phi} \left[f(z) \right] = \mathop{\mathbb{E}}_{g_u \sim \text{Gumbel}(0,1)} \left[f\left(\arg\max_u \left[\log q_\phi(u) + g_u \right] \right) \right].$$

- Gradients can come inside the expectation now.
- But gradient of an argmax operation is zero…

Gumbel Softmax

- Re-parameterized expectation:

$$\mathop{\mathbb{E}}_{z \sim q_\phi}\left[f(z)\right] = \mathop{\mathbb{E}}_{g_u \sim \text{Gumbel}(0,1)}\left[f\left(\arg\max_u \left[\log q_\phi(u) + g_u\right]\right)\right].$$

- What if we relax the argmax operation to a softmax?

$$\mathop{\mathbb{E}}_{z \sim q_\phi}\left[f(z)\right] = \mathop{\mathbb{E}}_{g_u \sim \text{Gumbel}(0,1)}\left[f\left(\operatorname{softmax}_u \left[\log q_\phi(u) + g_u; \tau\right]\right)\right].$$

- Softmax at temperature τ defined by:

$$\operatorname{softmax}_u(h(u); \tau) = \frac{\exp(h(u)/\tau)}{\sum_v \exp(h(v)/\tau)}.$$

Gumbel Softmax Critiqued

- Gumbel-softmax distribution also called the Concrete distribution.
- It is a *biased* estimator of the gradient.
- Bias goes away as $\tau \to 0$, but gradients blow up.
- Sometimes it works!
- Doesn't seem to work well when d is large.

Straight-Through Estimation?

- Want to take a gradient of the form $\nabla_\phi \mathop{\mathbb{E}}_{z \sim q_\phi} \big[f(z) \big]$.
- The straight-through estimator: $\tilde{\nabla}_\phi \mathop{\mathbb{E}}_{z \sim q_\phi} [f(z)] \equiv \mathop{\mathbb{E}}_{z \sim q_\phi} \left[(\nabla_\phi f(z)) \nabla_\phi q_\phi(z) \right]$.
- Where does this come from? Ignore sampling & define $\tilde{\nabla}_\phi z \equiv \nabla_\phi q_\phi(z)$.
- Use this to compute gradients of expressions with samples z:

$$\tilde{\nabla}_\phi f(z) = (\nabla_\phi f(z)) \tilde{\nabla}_\phi z = (\nabla_\phi f(z)) \nabla_\phi q_\phi(z).$$

- Unclear to me why this sometimes works.