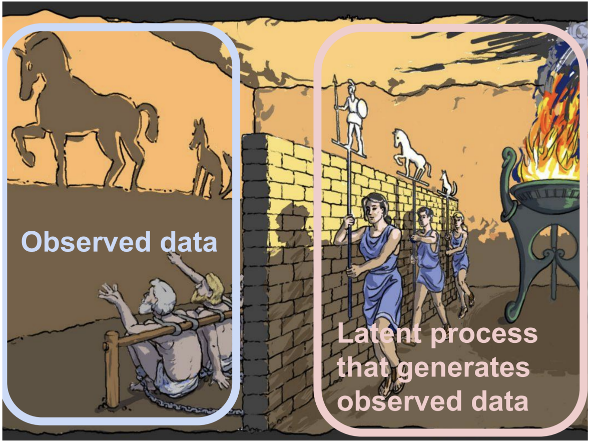
Lectures on Natural Language Processing

13. Latent-Variable Generative Models

Karl Stratos

Review: Introducing Latent Variables in Generative Models

$$p_\theta(x, z) = \underbrace{\kappa_\theta(x|z)}_{\text{conditional likelihood}} \times \underbrace{\pi_\theta(z)}_{\text{prior}}$$



Review: Marginal Log-Likelihood

Since we observe x, maximize the **marginal log-likelihood (MLL)**

$$\log m_\theta(x) = \begin{cases} \log\left(\sum_{z \in \mathcal{Z}} p_\theta(x, z)\right) & \text{if } \mathcal{Z} \text{ is discrete} \\ \log\left(\int_{z \in \mathcal{Z}} p_\theta(x, z) dz\right) & \text{if } \mathcal{Z} \text{ is continuous} \end{cases}$$

Computing MLL exactly is not tractable in all but simplest models. We need combinations of

- Sampling-based approximations
- *Variational* optimization: compute a surrogate objective that's easier to optimize

Warmup: Importance Sampling

Naive sampling. Draw iid $z_1 \dots z_K \sim \pi_\theta$ and estimate

$$m_\theta(x) \approx \frac{1}{K} \sum_{k=1}^{K} \kappa_\theta(x|z_k)$$

Consistent but can be high variance (e.g., $\kappa_\theta(x|z_{\text{rare}}) = 1$)

Warmup: Importance Sampling

Naive sampling. Draw iid $z_1 \dots z_K \sim \pi_\theta$ and estimate

$$m_\theta(x) \approx \frac{1}{K} \sum_{k=1}^{K} \kappa_\theta(x|z_k)$$

Consistent but can be high variance (e.g., $\kappa_\theta(x|z_{\text{rare}}) = 1$)

Importance sampling. Draw iid $z_1 \dots z_K \sim q(\cdot|x)$ and estimate

$$m_\theta(x) \approx \frac{1}{K} \sum_{k=1}^{K} \frac{p_\theta(x, z_k)}{q(z_k|x)}$$

Consistent for all q, can potentially reduce variance (if q is good)

Importance Sampling for MLL

Draw iid $z_1 \dots z_K \sim q(\cdot|x)$ and estimate

$$\log m_\theta(x) \approx \log \left(\frac{1}{K} \sum_{k=1}^{K} \frac{p_\theta(x, z_k)}{q(z_k|x)} \right)$$

The *expected* value

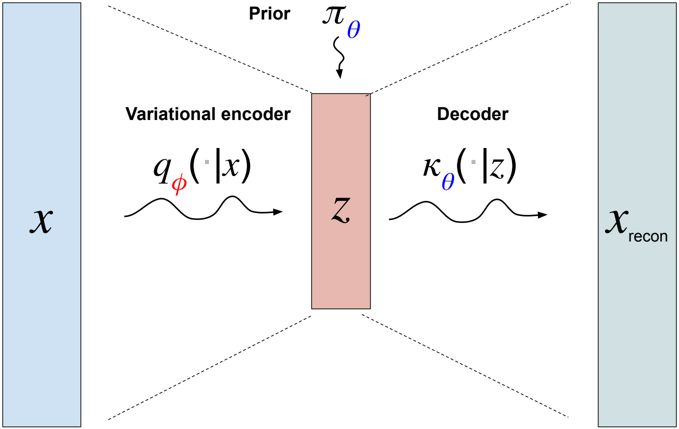
1. Is a lower bound on MLL (Jensen's inequality)

$$\mathbf{E}\left[\log \left(\frac{1}{K} \sum_{k=1}^{K} \frac{p_\theta(x, z_k)}{q(z_k|x)} \right) \right] \leq \log m_\theta(x)$$

2. Monotonically converges to MLL as $K \to \infty$ (Burda et al., 2016)

$$\mathbf{E}\left[\log \left(\frac{p_\theta(x, z)}{q(z|x)} \right) \right] \leq \mathbf{E}\left[\log \left(\frac{p_\theta(x, z_1)}{2q(z_1|x)} + \frac{p_\theta(x, z_2)}{2q(z_2|x)} \right) \right] \leq \cdots \to \log m_\theta(x)$$

Variational Autoencoders (VAEs)



$$\max_{\theta,\,\phi} \underbrace{\mathop{\mathbf{E}}_{z \sim q_\phi(\cdot|x)} [\log \kappa_\theta(x|z)]}_{\text{reconstruction}} - \beta \underbrace{\mathrm{KL}(q_\phi(\cdot|x) || \pi_\theta)}_{\text{regularization}}$$

Gaussian VAEs (Kingma and Welling, 2013)

- While VAEs are not tied to any specific distributions, many works focus on (for computational reasons)
  1. *Continuous* latent variable $z \in \mathbb{R}^d$
  2. *Gaussian* distributions over the latent variable
- The model

$$\begin{aligned} \pi &= \mathcal{N}(0_d, I_{d \times d}) \\ q_\phi(\cdot|x) &= \mathcal{N}(\mu_\phi(x), \mathrm{diag}(\sigma^2_\phi(x))) \\ \kappa_\theta(\cdot|z) &= \text{any conditional model of input} \end{aligned}$$

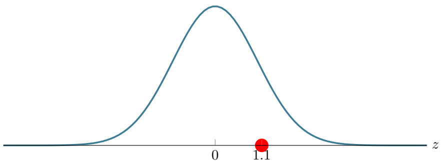
Here, $\mu_\phi, \sigma^2_\phi : \mathcal{X} \to \mathbb{R}^d$ can be any differentiable networks

$$\begin{bmatrix} \mu_\phi(x) \\ \sigma^2_\phi(x) \end{bmatrix} = \textbf{enc}_\phi(x)$$

Generation and Inference

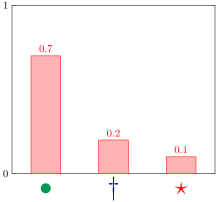
Generation. Latent z is drawn from a standard Gaussian prior π, decoder $\kappa_\theta(x|z)$ then defines some conditional input distribution

$$\pi(z) = \mathcal{N}(0,1)(z)$$



$\rightarrow$

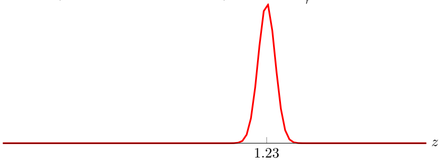
$$\kappa_\theta(x|1.1)$$



Inference. Given $x \in \mathcal{X}$, the variational encoder/inference network $q_\phi(\cdot|x)$ defines a conditional Gaussian distribution over z

$$q_\phi(z|●) = \mathcal{N}(\mu_\phi(●), \sigma^2_\phi(●))(z)$$

$x =$ ● $\rightarrow$



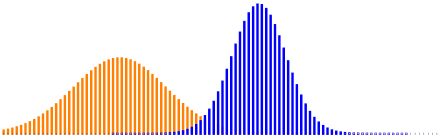
Estimating the Reconstruction Term

Drawing $z \sim \mathcal{N}(\mu, \sigma^2)$ is equivalent to $z = \mu + \epsilon\sigma$, $\epsilon \sim \mathcal{N}(0, 1)$

$$\begin{aligned}
&\mathop{\mathbf{E}}_{z \sim q_{\phi}(\cdot|x)} \left[\log \kappa_{\theta}(x|z)\right] \\
&= \mathop{\mathbf{E}}_{z \sim \mathcal{N}(\mu_{\phi}(x), \mathrm{diag}(\sigma^2_{\phi}(x)))} \left[\log \kappa_{\theta}(x|z)\right] \\
&= \mathop{\mathbf{E}}_{\epsilon \sim \mathcal{N}(0_d, I_{d \times d})} \left[\log \kappa_{\theta}(x|\mu_{\phi}(x) + \epsilon \odot \sigma_{\phi}(x))\right]
\end{aligned}$$

The expectation is no longer with respect to the model parameter (aka. "reparameterization trick"), stable to estimate

Computing the Regularization Term



$$\mathrm{KL}(Q||P) = \mathop{\mathbf{E}}_{z \sim Q}\left[\log \frac{Q(z)}{P(z)}\right]$$

KL divergence between diagonal Gaussians has a closed-form expression.

$$\begin{aligned}\mathrm{KL}(q_\phi(\cdot|x)||\pi_\theta) &= \mathrm{KL}(\underbrace{\mathcal{N}(\mu_\phi(x), \mathrm{diag}(\sigma^2_\phi(x)))}_{\text{diagonal Gaussian}} || \underbrace{\mathcal{N}(0_d, I_{d\times d})}_{\text{diagonal Gaussian}}) \\ &= \frac{1}{2}\left(\sigma^2_\phi(x) + \mu_\phi(x)^2 - 1_d - \log \sigma^2_\phi(x)\right)^\top 1_d\end{aligned}$$

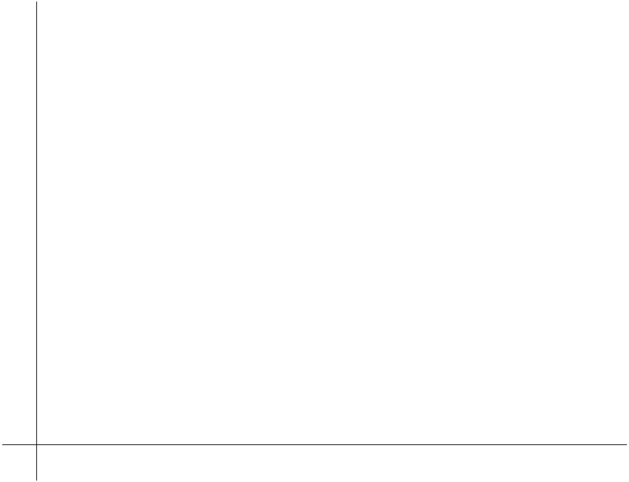
Example: Gaussian VAE for Text Generation (Bowman et al., 2016)

$$
\begin{aligned}
\mathcal{Z} &= \mathbb{R}^d \\
\mathcal{X} &= \text{sentences} \\
\pi &= \mathcal{N}(0_d, I_{d \times d}) \\
q_\phi(\cdot|x) &= \mathcal{N}(\mu_\phi(x), \mathrm{diag}(\sigma_\phi^2(x))) \\
\kappa_\theta(\cdot|z) &= \text{conditional language model}
\end{aligned}
$$

Training: given a sentence $x \in \mathcal{X}$, sample "noise" $\epsilon \sim \mathcal{N}(0_d, I_{d \times d})$ and take a gradient step with

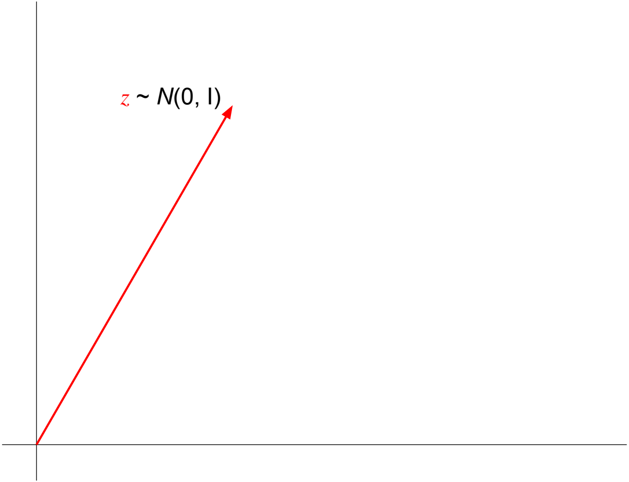
$$
\nabla_{\theta,\phi} \Bigg(\log \kappa_\theta(x|\mu_\phi(x) + \epsilon \odot \sigma_\phi(x)) \\
- \frac{\beta}{2} \left(\sigma_\phi^2(x) + \mu_\phi(x)^2 - 1_d - \log \sigma_\phi^2(x) \right)^\top 1_d \Bigg)
$$

Generating Text From a Gaussian VAE



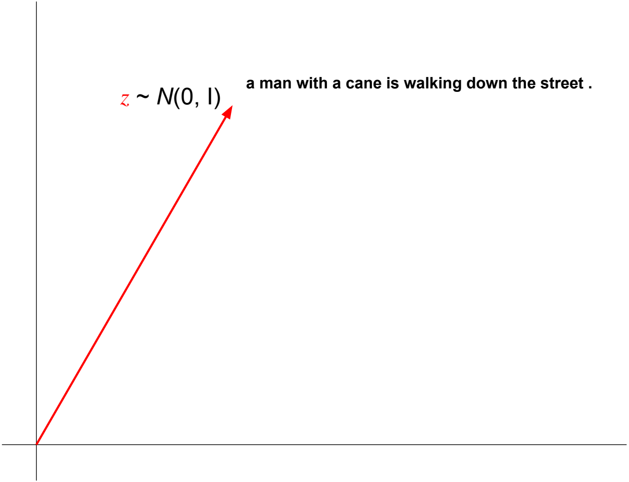
(examples from Li et al., 2019)

Generating Text From a Gaussian VAE



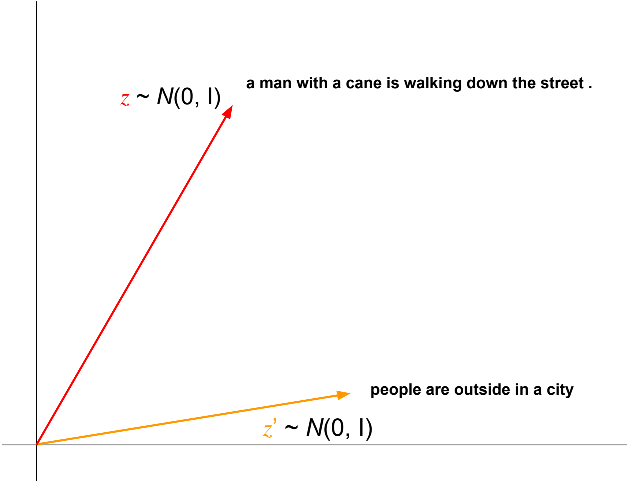
(examples from Li et al., 2019)

Generating Text From a Gaussian VAE



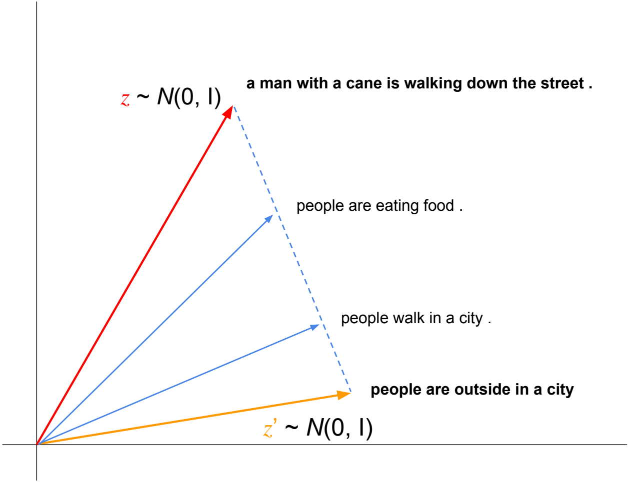
(examples from Li et al., 2019)

Generating Text From a Gaussian VAE



(examples from Li et al., 2019)

Generating Text From a Gaussian VAE



(examples from Li et al., 2019)

VAE Maximizes a Lower Bound on MLL

Evidence lower bound (ELBO). For any choice of

$$\begin{aligned} \pi(z) &= \text{prior} \\ \kappa(x|z) &= \text{conditional likelihood} \\ m(x) &= \mathop{\mathbf{E}}_{z \sim \pi} [\kappa(x|z)] \quad \text{(marginal likelihood)} \\ q(z|x) &= \text{variational posterior} \end{aligned}$$

$$\log m(x) \geq \mathop{\mathbf{E}}_{z \sim q(\cdot|x)} [\log \kappa(x|z)] - \mathrm{KL}(q(\cdot|x)||\pi)$$

The inquality is tight iff $q(z|x) = p(z|x)$ where $p(z|x) = \frac{\pi(z)\kappa(x|z)}{m(x)}$ is the true posterior.

Proof

$$
\begin{aligned}
\log m(x) &= \mathop{\mathbf{E}}_{z \sim q(\cdot|x)} [\log m(x)] \\
&= \mathop{\mathbf{E}}_{z \sim q(\cdot|x)} \left[\log \frac{m(x)p(z|x)q(z|x)}{p(z|x)q(z|x)}\right] \\
&= \mathop{\mathbf{E}}_{z \sim q(\cdot|x)} \left[\log \frac{p(x,z)}{q(z|x)}\right] + \underbrace{\mathop{\mathbf{E}}_{z \sim q(\cdot|x)} \left[\log \frac{q(z|x)}{p(z|x)}\right]}_{\mathrm{KL}(q(\cdot|x)||p(\cdot|x)) \geq 0}
\end{aligned}
$$

The first term is ELBO:

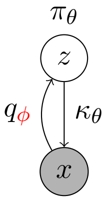
$$
\mathop{\mathbf{E}}_{z \sim q(\cdot|x)} \left[\log \frac{\kappa(x|z)\pi(z)}{q(z|x)}\right] = \mathop{\mathbf{E}}_{z \sim q(\cdot|x)} [\log \kappa(x|z)] - \mathrm{KL}(q(\cdot|x)||\pi)
$$

Variational Thinking

Directly maximizing MLL is difficult:

$$\theta \leftarrow \nabla_\theta \left(\log \mathop{\mathbf{E}}_{z \sim \pi_\theta} \left[\kappa_\theta(x|z) \right] \right)$$

VAE: introduce a variational posterior $q_\phi(z|x)$ and instead maximize a *lower bound* on MLL that's easier to compute



$$\theta \leftarrow \nabla_\theta \left(\mathop{\mathbf{E}}_{z \sim q_\phi(\cdot|x)} \left[\log \kappa_\theta(x|z) \right] - D_{\mathrm{KL}}(q_\phi(\cdot|x) || \pi_\theta) \right)$$

$$\phi \leftarrow \nabla_\phi \left(\mathop{\mathbf{E}}_{z \sim q_\phi(\cdot|x)} \left[\log \kappa_\theta(x|z) \right] - D_{\mathrm{KL}}(q_\phi(\cdot|x) || \pi_\theta) \right)$$

Connection to Expectation Maximization (EM)

EM (Dempster et al., 1977) is a fundamental technique for maximum likelihood estimation from incomplete data

Can be seen as alternating optimization of ELBO

$$\mathrm{ELBO}_x(p, q) = \mathop{\mathbf{E}}_{z \sim q(\cdot|x)} \left[\log \frac{p(x, z)}{q(z|x)} \right]$$

EM algorithm: Until convergence, repeat

$$q^\star \leftarrow \arg\max_q \ \mathrm{ELBO}_x(p, q) \qquad \text{(E Step)}$$

$$p \leftarrow \arg\max_p \ \mathrm{ELBO}_x(p, q^\star) \qquad \text{(M Step)}$$

Key point: E Step corresponds to using the true posterior (EM assumes that this is tractable), which makes ELBO=MLL. Thus every EM iteration monotonically increases MLL until convergence.

Population-Level VAE

In practice, we assume N iid input samples (e.g., sentences, images) $x_1 \ldots x_N \sim \mathbf{pop}$ and optimize:

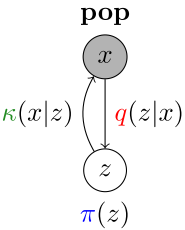
$$\widehat{\mathrm{ELBO}}(\kappa, q, \pi) = \frac{1}{N} \sum_{i=1}^{N} \mathop{\mathbf{E}}_{z \sim q(\cdot|x_i)} [\log \kappa(x_i|z)] - \mathrm{KL}(q(\cdot|x_i)||\pi)$$

An information theoretic view of ELBO: for all κ, q, π:

$$H(\mathbf{pop}) \leq -\mathrm{ELBO}(\kappa, q, \pi)$$

VAE viewed as minimizing the number of bits to encode the behavior of $\mathbf{pop}$, using the chosen latent-variable model class

Proof



"True model" now viewed as

$$p(x,z) = \mathbf{pop}(x)q(z|x)$$

which implies the (intractable to calculate) true marginals $p(z)$ and $p(x|z)$

$$
\begin{aligned}
-\log \mathbf{pop}(x) &= \log \frac{q(z|x)}{p(z)} - \log p(x|z) \\
\mathbf{E}_x[-\log \mathbf{pop}(x)] &= \mathbf{E}_{x,z}\left[\log \frac{q(z|x)}{p(z)}\right] - \mathbf{E}_{x,z}[\log p(x|z)] \\
&\leq \underbrace{\mathbf{E}_{x,z}\left[\log \frac{q(z|x)}{\pi(z)}\right] - \mathbf{E}_{x,z}[\log \kappa(x|z)]}_{-\mathrm{ELBO}(\kappa,q,\pi)}
\end{aligned}
$$

The inequality uses the fact that cross entropy upper bounds entropy

Posterior Collapse in VAEs

- What's one undesirable strategy to maximize

$$\mathrm{ELBO}(\theta, \phi) = \mathop{\mathbf{E}}_{x \sim \mathbf{pop},\ z \sim q_\phi(\cdot|x)} [\log \kappa_\theta(x|z)] - \beta \mathop{\mathbf{E}}_{x \sim \mathbf{pop}} [\mathrm{KL}(q_\phi(\cdot|x) || \pi_\theta)]$$

Posterior Collapse in VAEs

- What's one undesirable strategy to maximize

$$\text{ELBO}(\theta, \phi) = \mathop{\mathbf{E}}_{x \sim \mathbf{pop},\ z \sim q_\phi(\cdot|x)} [\log \kappa_\theta(x|z)] - \beta \mathop{\mathbf{E}}_{x \sim \mathbf{pop}} [\text{KL}(q_\phi(\cdot|x) || \pi_\theta)]$$

- Annihilate the KL term by setting $q_\phi(z|x) = \pi_\theta(z)$!
- Decoder κ_θ will then learn to ignore z
  - Degenerates to unconditional generative model
  - Known as **posterior collapse**

Posterior Collapse in VAEs

- What's one undesirable strategy to maximize

$$\mathrm{ELBO}(\theta, \phi) = \mathop{\mathbf{E}}_{x \sim \mathbf{pop},\ z \sim q_\phi(\cdot|x)} [\log \kappa_\theta(x|z)] - \beta \mathop{\mathbf{E}}_{x \sim \mathbf{pop}} [\mathrm{KL}(q_\phi(\cdot|x) || \pi_\theta)]$$

- Annihilate the KL term by setting $q_\phi(z|x) = \pi_\theta(z)$!
- Decoder κ_θ will then learn to ignore z
  - Degenerates to unconditional generative model
  - Known as **posterior collapse**
- Many strategies to alleviate
  - Annealing: Gradually increase KL weight β during training
  - Free bits (Kingma et al., 2016): Don't penalize KL if it's less than λ bits. Per-dimension version:

$$\sum_{i=1}^{d} \max \left\{ \lambda, \mathrm{KL}(q_\phi^{(i)}(\cdot|x) || \pi_\theta^{(i)}) \right\}$$

- Encoder pretraining (Li et al., 2019): Pretrain without KL term, reset decoder, train with annealed free bits

Quantities to Monitor During VAE Training

1. MLL: not equal to ELBO! But can be naturally estimated by importance sampling (with large K)

$$\log m_\theta(x) \approx \log \left(\frac{1}{K} \sum_{k=1}^{K} \frac{p_\theta(x, z_k)}{q_\phi(z_k|x)} \right)$$

   - Note that ELBO is a special case with $K = 1$
   - Importance weighted autoencoder (IWAE) (Burda et al., 2016) optimizes this directly
2. ELBO, consisting of two terms
   - **Reconstruction term**: Are we doing a good job of reconstructing the input? Always better in pure autoencoding
   - **KL term**: If this is zero, we have posterior collapse
3. Mutual information between $x \sim \mathbf{pop}$ and $z \sim q_\phi(\cdot|x)$
4. Number of active units (Burda et al., 2016): z_i is δ-active if

$$\underset{\substack{x \sim \mathbf{pop} \\ z_i \sim q_\phi^{(i)}(\cdot|x)}}{\mathrm{Var}} (z_i) > \delta \qquad \text{(commonly } \delta = 0.01\text{)}$$

Discrete Latent Variables

- Gaussian reparameterization trick allows us to "backpropagate through sampling" for continuous $z \in \mathbb{R}^d$
- What if z is **discrete**, for instance
$$z \in \{0, 1\}^d$$
  - (Possibly) More interpretable: 1st dim corresponding to sentiment, 2nd dim gender, etc.
  - Compression: Cheaper to store ints than floats.
- Typical inference network: "mean-field approximation"

$$q_\phi(z|x) = \prod_{i=1}^{d} q_\phi(z_i|x, i)$$

Easy to parametrize: $q_\phi(1|x, i) = \sigma(w_i \cdot \textbf{enc}_\phi(x))$

Backpropation Through Discrete Sampling

- Marginalization intractable (2^d possible values of z). This is despite mean-field approx

$$\mathop{\mathbf{E}}_{z \sim q_\phi(\cdot|x)} [\log \kappa_\theta(x|z)] = \sum_{z_1 \in \{0,1\}} q_\phi(z_1|x,1) \cdots \sum_{z_d \in \{0,1\}} q_\phi(z_d|x,d) \log \kappa_\theta(x|z)$$

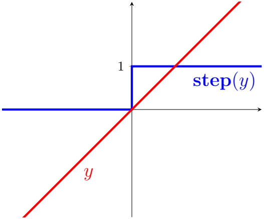
- So we'd like to approximate by sampling z. Here it's fine to sample $z_i \sim q_\phi(\cdot|x,i)$ independently and use $z = (z_1 \ldots z_d)$ to estimate $\log \kappa_\theta(x|z)$
- Exercise: Verify the reparameterization trick

$$z_i \sim q_\phi(\cdot|x,i) \qquad \Leftrightarrow \qquad z_i = \textbf{step}(q_\phi(1|x,i) - \epsilon)$$

  where $\epsilon \sim \mathrm{Unif}(0,1)$ and **step**$(y) = 1$ if $y \geq 0$ and 0 otherwise
- Unfortunately z_i still non-differentiable because of **step**

Straight-Through Gradient Estimator (Hinton, 2012)

- Idea: Approximate $\frac{\partial}{\partial y}\textbf{step}(y) \approx \frac{\partial}{\partial y} y = 1$
  - $f(y) = y$ linearization of $\textbf{step}(y)$ preserving the sign



- "Straight-through" gradient estimation of the step function

$$\frac{\partial \textbf{step}(f(\phi))}{\partial \phi} = \cancel{\frac{\partial \textbf{step}(f(\phi))}{\partial f(\phi)}} \times \frac{\partial f(\phi)}{\partial \phi} \approx \frac{\partial f(\phi)}{\partial \phi}$$

- With this approximation, we can sample $z_i \sim q_\phi(\cdot|x, i)$ in the forward pass and backpropage directly to $q_\phi(1|x, i)$ in the backward pass!

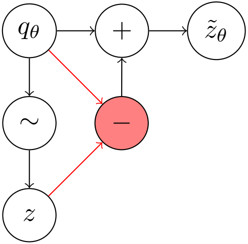
Implemention Trick for Straight-Through

- PyTorch style code

$$z = \texttt{bernoulli}(q_\phi) \qquad \text{\# Not differentiable}$$

$$\tilde{z}_\phi = q_\phi + (z - q_\phi).\texttt{detach}() \qquad \text{\# Differentiable } (\tilde{z}_\phi.\texttt{val} = z.\texttt{val})$$

- Corresponding computation graph



- Generally useful trick. Another example: gradient reversal layer (Ganin and Lempitsky, 2014)

$$\tilde{x} = -x + (x + x).\texttt{detach}()$$

Categorical VAE

- What if $z \in \{1 \dots K\}^d$?
  - Again mean-field approx $q_\phi(z|x) = \prod_{i=1}^d q_\phi(z_i|x, i)$
  - Easy to parameterize: $q_\phi(\cdot|x, i) = \mathrm{softmax}(\textbf{enc}_\phi^{(i)}(x))$
- **Gumbel-max trick**

$$z_i \sim q_\phi(\cdot|x, i) \qquad \Leftrightarrow \qquad z_i = \arg\max_{k=1}^{K} \, [\textbf{enc}_\phi^{(i)}(x)]_k + \epsilon_k$$

where $\epsilon_1 \dots \epsilon_K \overset{iid}{\sim} \mathrm{Gumbel}(0, 1)$. Unfortunately z_i still non-differentiable because of $\arg\max$

- **Differentiable relaxation.** WLOG assume one-hot representation: $z_i = e_k \in \{0, 1\}^K$ means $z_i = k$. Then

$$z_i \sim q_\phi(\cdot|x, i) \quad \overset{\tau \to 0^+}{\Leftrightarrow} \quad z_i = \mathrm{softmax}\left(\frac{\textbf{enc}_\phi^{(i)}(x) + \epsilon}{\tau}\right)$$

where $\epsilon_1 \dots \epsilon_K \overset{iid}{\sim} \mathrm{Gumbel}(0, 1)$. Now z_i differentiable

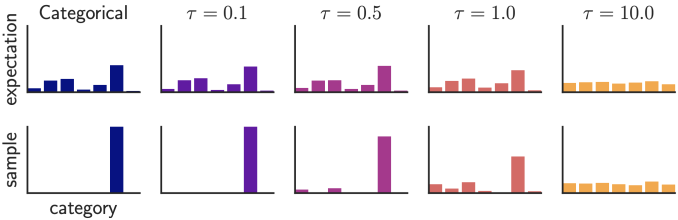
Gumbel-Softmax (Jang et al., 2016)

- Let $d = 1$ for simplicity, $q_\phi(\cdot|x) = \mathrm{softmax}(\mathbf{enc}_\phi(x))$

$$\frac{\partial}{\partial \phi}\left(\mathop{\mathbf{E}}_{z \sim q_\phi(\cdot|x)}\left[\log \kappa_\theta(x|z)\right]\right) \approx \mathop{\mathbf{E}}_{\epsilon \sim \mathrm{Gumbel}^K(0,1)}\left(\frac{\partial}{\partial \phi}\left(\log \kappa_\theta\left(x \,\middle|\, \underbrace{\mathrm{softmax}\left(\frac{\mathbf{enc}_\phi(x)+\epsilon}{\tau}\right)}_{[0,1]^K}\right)\right)\right)$$

  Note κ_θ must handle vector representation of z

- In practice use fixed τ (e.g., 0.9)



Striaght-Through Gumbel-Softmax

- For a fixed temperature the actual input to κ_θ is never sparse
- Idea: Enforce sparsity by sampling and use straight-through
  - More aligned with test time (when we actually sample), some applications need sparse input (reinforcement learning)
- **Forward pass**

$$
\begin{aligned}
\epsilon &\sim \mathrm{Gumbel}^K(0,1) \\
\delta_\tau &= \mathrm{softmax}\left(\frac{\textbf{enc}_\phi(x)+\epsilon}{\tau}\right) \\
z &\sim \mathrm{Cat}(\delta_\tau) \\
\tilde{z} &= \delta_\tau + (\textsf{one-hot}(z) - \delta_\tau).\texttt{detach}() \\
J_{\mathrm{recon}} &= \log \kappa_\theta(x|\tilde{z})
\end{aligned}
$$

- Can be seen as approximating the gradient of a "snap" function with a linearization