$$\ell(\theta) = \sum_i \log p(x_i;\theta) = \sum_i \log \sum_{z_i \in \Omega(z_i)} p(x_i, z_i;\theta)$$

$$= \sum_i \log \sum_{z_i} q_i(z_i;\phi_i) \frac{p(x_i, z_i;\theta)}{q_i(z_i;\phi_i)}$$

$$\begin{cases} \geq \sum_i \sum_{z_i} q_i(z_i;\phi_i) \cdot \log \frac{p(x_i, z_i;\theta)}{q_i(z_i;\phi_i)} = \sum_i \mathcal{L}(x_i;\theta,\phi_i) \\ = \sum_i \left[\sum_{z_i} q_i(z_i;\phi_i) \cdot \log p(x_i|z_i;\theta) + H(q_i) + KL\left(q_i(z_i;\phi_i) \| p(z_i|x_i;\theta)\right) \right] \end{cases}$$

$\nabla_\theta \mathcal{L}(x_i;\theta,\phi_i) = \nabla_\theta \sum_{z_i} q_i(z_i;\phi_i) \cdot \log p(x_i, z_i;\theta) \approx \frac{1}{k} \sum_k \nabla_\theta \log p(x_i, z_i^{(k)};\theta)$

where $z_i^{(k)}$'s are iid sampled from q_i

$\nabla_{\phi_i} \sum_{z_i} q_i(z_i;\phi_i) \cdot \log q_i(z_i;\phi_i)$ is difficult to computation

- Reparameterization

want to compute $\nabla_\phi E_{q(z;\phi)}[r(z)] = \nabla_\phi \int q(z;\phi)\, r(z)\, dz$ where z is continuous

suppose $q(z;\phi) = \mathcal{N}(u, \sigma^2 I)$, $\phi = (u, \sigma)$

two equivalent way to sample from q: $\begin{cases} z \sim \mathcal{N}(u, \sigma^2 I) \\ \epsilon \sim \mathcal{N}(0, I) \quad z = u + \sigma\epsilon = g(\epsilon;\phi) \end{cases}$

$E_{z\sim q(z;\phi)}[r(z)] = E_{\epsilon\sim\mathcal{N}(0,I)}[r(g(\epsilon,\phi))] = \int p(\epsilon) \cdot r(u+\sigma\epsilon)\, d\epsilon$

$\nabla_\phi E_{z\sim q(z;\phi)}[r(z)] = \nabla_\phi E_{\epsilon\sim\mathcal{N}(0,I)}[r(g(\epsilon,\phi))] = E_{\epsilon\sim\mathcal{N}(0,I)}[\nabla_\phi r(g(\epsilon,\phi))] \approx \frac{1}{k} \sum_k \nabla_\phi r(g(\epsilon^k;\phi))$ where $\epsilon^1 \cdots \epsilon^k$ are iid $\mathcal{N}(0,I)$

Want to compute $\nabla_{\phi_i} -\sum_{z_i} q_i(z_i;\phi_i)\cdot \log q_i(z_i;\phi_i) = \nabla_{\phi_i} E_{z_i\sim q_i}\left[\log q_i(z_i;\phi_i)\right]$

assume $z_i = u + \theta\epsilon = g(\epsilon;\phi)$

$$\nabla_{\phi_i} E_{z_i\sim q_i}\left[\log q_i(z_i;\phi_i)\right] = \nabla_{\phi_i} E_{\epsilon}\left[\log q_i\left(g(\epsilon;\phi_i);\phi_i\right)\right]$$

· Amortized Inference

$$\ell(\theta) \geq \sum_i \sum_{z_i} q_i(z_i;\phi_i) \log \frac{p(x_i,z_i;\theta)}{q_i(z_i;\phi_i)} = \sum_i \mathcal{L}(x_i;\theta;\phi_i)$$

let $\theta^* = \arg\max \ell(\theta)$ $\quad (\hat\theta,\hat\phi) = \arg\max_{\theta,\phi} \sum_i \sum_{z_i} q_i(z_i;\phi_i) \log \frac{p(x_i,z_i;\theta)}{q_i(z_i;\phi_i)} = \arg\max_{\theta,\phi} \sum_i \mathcal{L}(x_i;\theta,\phi_i)$

$$\sum_i \mathcal{L}(x_i;\hat\theta,\hat\phi_i) \leq \sum_i \log \sum_{z_i} p(x_i,z_i;\hat\theta) = \ell(\hat\theta) \leq \ell(\theta^*)$$

$$\therefore \max_\theta \ell(\theta) \geq \max_{\theta,\phi} \sum_i \mathcal{L}(x_i;\theta,\phi_i)$$

So far we apply variational inference for each x_i, this does not scale for large dataset

amortization: learn a single function $f: x_i \to \phi_i^*$

try to learn $q_i(z_i;\phi) \approx p(z_i|x_i;\theta)$

Instead of learing ϕ_i for each $q_i(z_i;\phi_i)$

we learn $q(z;f_\lambda(x_i))$

$$\max_{\phi_i} \sum_{z_i} q_i(z_i;\phi_i)\cdot \log \frac{p(x_i,z_i;\theta)}{q_i(z_i;\phi_i)}$$
$$= \max_{\phi_i} \sum_{z_i} q_i(z_i;\phi_i)\cdot \log \frac{p(z_i|x_i;\theta)\cdot p(x_i;\theta)}{q_i(z_i;\phi_i)}$$
$$= \max_{\phi_i} \sum_{z_i} q_i(z_i;\phi_i)\cdot \log p(x_i;\theta) - \sum_{z_i} q_i(z_i;\phi_i) \frac{q_i(z_i;\phi_i)}{p(z_i|x_i;\theta)}$$
$$= \min_{\phi_i} KL\left(q_i(z_i;\phi_i) \,\|\, p(z_i|x_i;\theta)\right)$$

$\max \ell(\theta) = \sum_i \log p(x_i;\theta)$

E-Step: $Q_i(z_i) = p(z_i|x_i;\theta)$

M-Step: $\theta = \arg\max_\theta \sum_{i=n} \sum_{z_i} Q_i(z_i)\cdot \log \frac{p(x_i,z_i;\theta)}{Q_i(z_i)}$

· Learning with Amortized Inference

$$
\begin{aligned}
\ell &= \sum_i \log \sum_{z_i} q_i(z_i;\phi_i) \frac{p(x_i,z_i;\theta)}{q_i(z_i;\phi_i)} \\
&\geq \sum_i \sum_{z_i} q_i(z_i;\phi_i) \log \frac{p(x_i,z_i;\theta)}{q_i(z_i;\phi_i)} \\
&= \sum_i \sum_{z_i} q_i(z_i;f_\lambda(x_i)) \cdot \log \frac{p(x_i,z_i;\theta)}{q_i(z_i;f_\lambda(x_i))} \\
&= \sum_i \mathbb{E}_{z_i \sim q(z_i;f_\lambda(x_i))} \left[\log p(x_i,z_i;\theta) - \log q_i(z_i;f_\lambda(x_i)) \right] \\
&= \sum_i \mathcal{L}(x_i;\theta,\lambda)
\end{aligned}
$$

initialize θ, λ

randomly sample a data x_i from dataset

compute $\nabla_\theta \mathcal{L}(x_i;\theta,\lambda)$ with monte carlo

compute $\nabla_\lambda \mathcal{L}(x_i;\theta,\lambda)$ with reparameterization

update θ, ϕ in the gradient direction

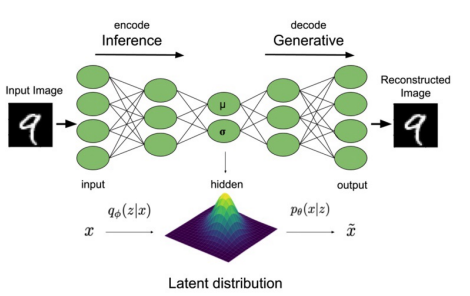
· Variational Auto Encoder

$$
\begin{aligned}
\mathcal{L}(x_i,\theta,\lambda) &= \mathbb{E}_{z_i \sim q_i(z_i;f_\lambda(x_i))} \left[\log p(x_i,z_i;\theta) - \log q_i(z_i;f_\lambda(x_i)) \right] \\
&= \mathbb{E}_{z_i \sim q_i(z_i;f_\lambda(x_i))} \left[\log p(x_i,z_i;\theta) + \log p(z_i;\theta) - \log p(z_i;\theta) - \log q_i(z_i;f_\lambda(x_i)) \right] \\
&= \underbrace{\mathbb{E}_{z_i \sim q_i(z_i;f_\lambda(x_i))} \left[\log p(x_i|z_i;\theta) \right]}_{\text{encouraging } \hat{x} \approx x_i} - \underbrace{KL\left(q_i(z_i;f_\lambda(x_i)) \,\|\, p(z_i;\theta) \right)}_{\text{encouraging } \hat{z} \text{ to be likely under prior } p(z)}
\end{aligned}
$$

take a sample x_i

sample $\hat{z} \sim q(z_i;f_\lambda(x_i))$ (encoder)

reconstruct $\hat{x}$ by sampling from $p(x|\hat{z};\theta)$ (decoder)



We can map simple distributions to complex distributions using change of variable

· change of variable

if $X = f(z)$ and f is monotone with $z = f^{-1}(x) = h(x)$
then $p_X(x) = p_Z(h(x)) \cdot |h'(x)|$

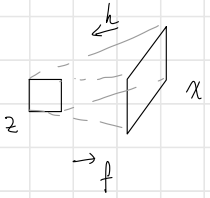
$= h'(x)$

$$\int_{-\infty}^{+\infty} p_Z(z)\,dz = \int_{-\infty}^{+\infty} p_Z(h(x))\,dh(x) = \int_{\infty}^{+\infty} p_Z(h(x)) \cdot h'(x)\,dx$$

$f: \mathbb{R}^n \mapsto \mathbb{R}^n$ is an invertible function $X = f(z)$ $z = h(x)$

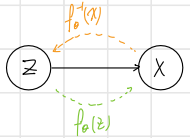
$$p_X(x) = p_Z(h(x)) \cdot \left|\det\left(\nabla_x h(x)\right)\right|$$

$$= p_Z(z) \cdot \left|\det\left(\nabla_z f(z)\right)\right|^{-1}$$



x and z need to be continuous and have same dimension

· Normalizing Flow Models



in a normalizing flow model, the mapping from z to X $f_\theta: \mathbb{R}^n \to \mathbb{R}^n$ is deterministic and invertible
$X = f_\theta(z)$ and $z = f_\theta^{-1}(x)$

$$p_X(x;\theta) = p_Z(z;\theta) \cdot \left|\det\left(\nabla f_\theta^{-1}(x)\right)\right|$$

$$x = f_\theta(z) = f_\theta^{M} f_\theta^{M-1} \cdots f_\theta^{1}(z)$$

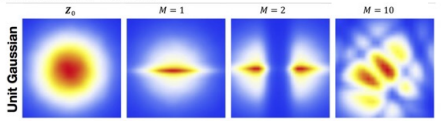
can compose multiple invertible transformation into one large invertible transformation

Start with a simple distribution for z_0 (eg. Gaussian)
Apply a sequence of invertible transformation $f_i(\,)$

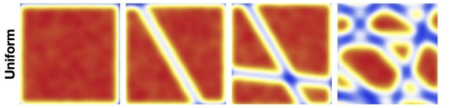
$$z_0 \xrightarrow{f_1} z_1 \xrightarrow{f_2} z_2 \cdots \longrightarrow x$$

$$P_X(x;\theta) = P_Z\left(f_\theta^{-1}(x)\right) \cdot \prod_{i=1}^{n} \left|\det\left[\nabla f_i^{-1}(z_i)\right]\right|$$

- Base distribution: Gaussian



- Base distribution: Uniform



- Learning and Inference

$$\max_\theta \log P_X(D) = \sum_{x \in D} P_Z\left(f_\theta^{-1}(x)\right) + \log\left|\det\left[\nabla f_i^{-1}(z_i)\right]\right|$$

Sampling: sample from $P_Z(z)$, and perform forward transformation $x = f_\theta(z)$
Latent representation: $z = f_\theta^{-1}(x)$