1.1

① : $E_{q(z|x)}\left[\log\left(p_\theta(x|z)\,p(z)\right)\right]$

② $\log p_\theta(x) - D_{KL}\left(q(z|x) \,\|\, p_\theta(z|x)\right)$

WTS $\arg\max_\theta$ ② $=$ $\arg\max_\theta$ ①

$$\arg\max_\theta \log(p_\theta(x)) - D_{KL}(q(z|x) \,\|\, p_\theta(z|x))$$

$$= \arg\max_\theta \log(p_\theta(x)) - E_{q(z|x)}\left[\log(q(z|x)) - \log(p_\theta(z|x))\right]$$

$$= \arg\max_\theta \log(p_\theta(x) - E_{q(z|x)}\left[\log(q(z|x)) - \log\left(\frac{p_\theta(x|z)\,p(z)}{p_\theta(x)}\right)\right]$$

Bayes rule

$$= \underset{\theta}{\arg\max} \; \log(p_\theta(x)) - E_{q(z|x)}\left[\log(q(z|x)) + \log(p_\theta(x)) - \log p_\theta(x|z) - \log p(z)\right]$$

$$= \underset{\theta}{\arg\max} \; - E_{q(z|x)}\left[\log(q(z|x)) - \log p_\theta(x|z) - \log(p(z))\right]$$

$$= \underset{\theta}{\arg\max} \; E_{q(z|x)}\left[\log(p_\theta(x|z)) + \log p(z) - \log(q(z|x))\right]$$

$$= \underset{\theta}{\arg\max} \; E_{q(z|x)}\left[\log(p_\theta(x|z)) + \log(p(z))\right]$$

$$= \underset{\theta}{\arg\max} \; E_{q(z|x)}\left[\log(p_\theta(x|z) \cdot p(z))\right]$$

$$= \underset{\theta}{\arg\max} \; \textcircled{1} \qquad \boxtimes$$

1.2

We know that

$$\log p(x) = \mathcal{L}(q, p) + KL(q(z|x) \,\|\, p(z|x))$$

using q_{ϕ^*} and q_i^*, we get equations

① $\log(p_\theta(x_i)) = \mathcal{L}(q_{\phi^*}, p_\theta) + D_{KL}(q_{\phi^*}(z|x_i) \,\|\, p_\theta(z|x_i))$

② $\log(p_\theta(x_i) = \mathcal{L}(q_i^*, p_\theta) + D_{KL}(q_i^*(z|x_i) \,\|\, p_\theta(z|x_i))$

Then

① = ②

$\mathcal{L}(q_{\phi^*}, p_\theta) + D_{KL}(q_{\phi}(z|x_i) \,\|\, p_\theta(z|x_i) =$

$\mathcal{L}(q_i^*, p_\theta) + D_{KL}(q_i^*(z|x_i) \,\|\, p_\theta(z|x_i))$

$$\Rightarrow D_{KL}(q_{f_\phi} \| p_\theta) = D_{KL}(q_{f_i}^* \| p_\theta) + \overbrace{\mathcal{L}(q_{f_i}^*, p_\theta) - \mathcal{L}(q_{f_{\phi^*}}, p_\theta)}^{>0}$$

fact: $\mathcal{L}(q_{f_i}^*, p_\theta) > \mathcal{L}(q_{f_{\phi^*}}, p_\theta)$, since

$q_{f_i}^*$ is optimized for specific x_i

$$\Rightarrow \mathcal{L}(q_{f_i}^*, p_\theta) - \mathcal{L}(q_{f_{\phi^*}}, p_\theta) > 0$$

Consequently,

$$D_{KL}(q_{f_\phi} \| p_\theta) > D_{KL}(q_{f_i}^* | p_\theta)$$

1.3

a) $p_\theta(x) = \mathcal{L}(q, p) + D_{KL}(q(z|x) \| p(z|x))$

$$p_\theta(x) - \mathcal{L}(q, p) = \underbrace{D_{KL}(q(z|x) \| p(z|x))}_{\text{bias}}$$

Since we showed in 1.2 that

$$D_{KL}(q_{\phi^*} \| p_\theta) > D_{KL}(q_*^i \| p_\theta),$$

the per-instance approach is less biased than amortized approach

b) per-instance is less computationally efficient since you must optimize for every x_i.

In amortized, you train once over data points and then only have to compute a single forward pass for new data points.

c) In terms of memory, amortized is less costly since you have shared paramaters ϕ^* across datapoints,

whereas in per-instance you have a set of paramaters for every q_i^*.

2.1

When $\beta = 0$, the loss is

$$\mathcal{L}^{\beta\text{-VAE}}_{\theta,\phi,\beta}(x) = -\mathbb{E}_{z \sim q_\phi(z|x)}\left[\log p_\theta(x|z)\right]$$

The objective here is to minimize the negative log-likelihood of the data.

The optimal solution will be for the encoder to give $p(x_i) = 1$ for $x_i \in D$. This will result in very "peaky" distributions in latent space that will likely not at all resemble the prior $p_\theta(z)$.

2.2 The optimal solution would be to make $q_\phi(z|x) = p_\theta(z)$, as this would give a KL of zero.

This is bad because encoder would ignore information about x.

2.4

New loss:

$$\mathcal{L}^{\beta\text{-VAE}}_{\theta,\phi,\beta,\alpha,\Delta}(x) = \alpha \cdot \mathbb{E}_{z \sim q_\phi(z|x)} \left[\| x - f_\Delta(z) \|_2 \right] +$$

$$\beta \cdot KL(q_\phi(z|x) \| p_\theta(z))$$

Here the decoder f_Δ is trained to reconstruct x with a similarity metric, Euclidian distance. A hyper-parameter $\alpha \in \mathbb{R}$ is used to tune the reconstruction.

2.5 a)

Let $x := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$

$$f_a(x) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x$$

$$f_b(x) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} x$$

then $z_a = f_a(x) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$

$$z_b = f_b(x) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$$

$$z_a \neq z_b$$

2.5 b)

It suggests there are rarely unique latent representations of datapoints x, meaning you will likely have different encoder and decoder functions that are equally as good at reconstruction loss.

$$S := \{ z \in \mathbb{R}^d \mid \exists f \text{ where } f(x) = z \}$$

3.1 $V(d, g) = d \cdot g$, then

$$\partial_g V = d$$

$$\partial_d V = g$$

We want to maximize V with respect to d (gradient ascent) and minimize V with respect to g (gradient descent), the update rules are

$$d_{k+1} = d_k + \alpha \cdot g_k$$

$$g_{k+1} = g_k - \alpha \cdot d_k$$

In matrix format:

$$\begin{bmatrix} d_{k+1} \\ g_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & \alpha \\ -\alpha & 1 \end{bmatrix} \begin{bmatrix} d_k \\ g_k \end{bmatrix},$$

$$A = \begin{bmatrix} 1 & \alpha \\ -\alpha & 1 \end{bmatrix}$$

3.2

A stationary point is a point that remains unchanged after gradient descent up date, since it's partial derivatives are zero. The equation to solve is then

$$\begin{pmatrix} d \\ g \end{pmatrix} = A \begin{pmatrix} d \\ g \end{pmatrix} = \begin{pmatrix} 1 & \alpha \\ -\alpha & 1 \end{pmatrix} \begin{pmatrix} d \\ g \end{pmatrix}$$

$$\begin{pmatrix} d \\ g \end{pmatrix} = \begin{pmatrix} d + \alpha g \\ g - \alpha d \end{pmatrix}$$

$$\begin{aligned} d &= d + \alpha g \\ g &= g - \alpha d \end{aligned} \Rightarrow \begin{aligned} 0 &= \alpha g \\ 0 &= -\alpha d \end{aligned}$$

This means $(d, g) = (0, 0)$ is a stationery point.

3.3 To find eigenvalues of A, we solve

$$\det\left(\begin{pmatrix} 1-\lambda & \alpha \\ -\alpha & 1-\lambda \end{pmatrix}\right) = 0$$

$$(1-\lambda)^2 + \alpha^2 = 0$$

$$(1-\lambda)^2 = -\alpha^2$$

$$1-\lambda = \pm i\alpha$$

$$\lambda = 1 \pm i\alpha$$

$$\Rightarrow \quad \lambda_1 = 1 + i\alpha \quad , \quad \lambda_2 = 1 - i\alpha$$

We can calculate the magnitude of these complex numbers λ_1, λ_2 as

$$|\lambda_1| = \sqrt{1^2 + (\alpha)^2} = \sqrt{1+\alpha^2}$$

$$|\lambda_2| = \sqrt{1^2 + (-\alpha)^2} = \sqrt{1+\alpha^2} = |\lambda_1|$$

For whatever learning rate $\alpha > 0$ chosen, we have $|\lambda_1| = |\lambda_2| = \sqrt{1+\alpha^2} > 0$

This means training diverges, since the eigenvalues of the update matrix have magnitudes larger than 1, the values of d_k and g_k grow at each iteration, unbounded as $k \to \infty$.

4.1

$$\mathcal{L}_{NCE}(E_\theta, p_d, p_n) = \mathop{E}_{x\sim p_d}\left[\log(\sigma(E_\theta(x))\right] + \mathop{E}_{z\sim p_n}\left[\log(1-\sigma(E_\theta(z))\right]$$

To optimize $\mathcal{L}_{NCE}$, we need to optimize the loss contribution from a single data point, call it $\mathcal{L}_x$.

$$\mathcal{L}_x(E(x)) = p_d(x)\log(\sigma(E(x)) + p_n(x)\log(1-\sigma(E(x))$$

where $p_d(x)$ is the probability x is from data and $p_n(x)$ is the probability x is noise

let $a := p_d(x)$, $b := p_n(x)$, $u := E(x)$

$$\mathcal{L}_x(u) = a\cdot\log(\sigma(u)) + b\cdot\log(1-\sigma(u))$$

$$\frac{\partial}{\partial u} \mathcal{L}(u) = \frac{\partial}{\partial u}\left[a \log(\sigma(u)) + b \cdot \log(1-\sigma(u))\right] = 0$$

$$= a \frac{1}{\sigma(u)} \cdot \sigma'(u) + b \frac{1}{1-\sigma(u)} \cdot (-1)\, \sigma'(u) = 0$$

$$= \frac{a \cdot \cancel{\sigma(u)}(1-\sigma(u))}{\cancel{\sigma(u)}} - \frac{b}{\cancel{1-\sigma(u)}} \sigma(u) \cdot \cancel{(1-\sigma(u))} = 0$$

$$= a(1-\sigma(u)) - b\,\sigma(u) = 0$$

$$= a - a\sigma(u) - b\sigma(u) = 0$$

$$a = \sigma(u)(a+b)$$

$$\sigma^*(u) = \frac{a}{a+b} = \frac{p_d(x)}{p_d(x) + p_n(x)}$$

the optimal solution is when

$$\sigma(E(x)) = \frac{p_d}{p_d + p_n}$$

$$\frac{1}{1+e^{-E(x)}} = \frac{p_d}{p_d + p_n}$$

$$p_d + p_n = p_d + p_d e^{-E(x)}$$

$$\frac{p_n}{p_d} = e^{-E(x)}$$

$$-E(x) = \log\left(\frac{p_n}{p_d}\right)$$

$$E_{\theta^*}(x) = \log\left(\frac{p_d}{p_n}\right)$$

4.2 $$L_{contr}(f_\theta, g_\phi, D) = \frac{1}{N}\sum_{i=1}^{N}\left[-\log \frac{e^{z_i^T \cdot y_i}}{\sum_{j=1}^{M} e^{z_i^T y_j}}\right]$$

$$-\log\left(\frac{e^{z_i^T \cdot y_i}}{\sum_{j=1}^{M} e^{z_i^T y_j}}\right) = -\left[\log\left(e^{z_i^T \cdot y_i}\right) - \log\left(\frac{M}{M}\sum_{j=1}^{M} e^{z_i^T y_j}\right)\right]$$

$$= -\left[\log\left(e^{z_i^T \cdot y_i}\right) - \left(\log \frac{1}{M}\sum_{j=1}^{M} e^{z_i^T y_j} + \log(M)\right)\right]$$

$$= -\log\left(e^{z_i^T y_i}\right) + \log\left(\frac{1}{M}\sum_{j=1}^{M} e^{z_i^T y_j}\right) + \log(M)$$

- Replace this expression in $L_{contr}(f_\theta, g_\phi, D)$

$$L_{contr} = \frac{1}{N}\sum_{i=1}^{N}\left(-\log\left(e^{z_i^T y_i}\right) + \log\left(\frac{1}{M}\sum_{j=1}^{M} e^{z_i^T y_j}\right) + \log(M)\right)$$

$$= \frac{1}{N}\sum_{i=1}^{N} -\log\left(e^{z_i^T y_i}\right) + \frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{M}\sum_{j=1}^{M} e^{z_i^T y_j}\right) + \frac{1}{N}\sum_{i=1}^{N}\log(M)$$

$$= -\frac{1}{N}\sum_{i=1}^{N} z_i^T \cdot y_i + \frac{1}{N}\sum_{i=1}^{N} \log \frac{1}{M}\sum_{j=1}^{M} e^{z_i^T y_j} + \log(M)$$

$$= -\mathcal{L}_{align}(f_\theta, g_\theta, D) + \mathcal{L}_{unif}(f_\theta, g_\theta, D) + \log(M)$$

What does each term represent?

$\underline{\mathcal{L}_{align}}$: This computes the dot product between the positive pair representations, which means it is computing the average similarity of positive pairs. A smaller negative similarity "aligns" the positive pairs.

$\mathcal{L}_{unif}$: This term promotes uniformity in the distribution of negative embeddings y_j. Essentially, for a fixed z_i, the term penalizes many y_j's being overly similar to z_i, thus promoting uniformity in the distribution of y_j.

$\log(M)$: Fixed offset that does not effect optimization.

Relating this to information maximization principal, we can look at the mutual information between X and the encoded value of X, Z:

$$I(X, Z) = H(Z) - H(Z|X)$$

We want to miximize the mutual

information between X and it's encoded representations Z.

The $\mathcal{L}_{unif}$ term promotes large $H(Z)$, as uniform distributions have large entropy.

The $\mathcal{L}_{align}$ term promotes small value for $H(Z|X)$, since aligning the representations of the positive samples makes $H(Z|X)$ small. Essentially, aligning the different "views" of the same datapoint (x_i, x_i') means the conditional variability $H(Z|X)$ is reduced.

What happens as $M \to \infty$?

$$L_{unif} = \frac{1}{N} \sum_{i=1}^{N} \log\left(\frac{1}{M} \sum_{j=1}^{M} e^{z_i^T y_j} \right)$$

converges to:

$$\frac{1}{N} \sum_{i=1}^{N} \log\left(\mathop{E}_{y \sim p(y)} \left[e^{z_i^T y} \right] \right)$$

as $M \to \infty$

4.3 $$\mathcal{L}_*(h_\psi, g_\phi, D) = \frac{1}{N}\sum_{i=1}^{N} -\log\left(\frac{\exp\left(h_\psi(x_i, x_i')^T \cdot g_\phi(x_i')\right)}{\frac{1}{M}\sum_{j=1}^{M}\exp\left(h_\psi(x_i, x_i')^T \cdot g_\phi(x_j)\right)}\right)$$

In this formulation, we recover similar terms:

$$\underline{\mathcal{L}_{align}} = -\frac{1}{N}\sum_{i=1}^{N} h_\psi(x_i, x_i') \cdot g_\phi(x_i')$$

which aims to align positive representations.

$$\underline{\mathcal{L}_{unif}} = \frac{1}{N}\sum_{i=1}^{N} \log \frac{1}{M}\sum_{j=1}^{M} e^{h_\psi(x_i, x_i')^T \cdot g_\phi(x_j)}$$

which promotes uniform distribution of negative representations.

$\log(M)$: rescaling constant.

4.3

EBM with L_{contr}:

This design is straight forward, as the model produces embeddings independently. However, representations may not be as rich since they both use one data modality.

EBM with L^*:

This version has the advantage of potentially capturing richer representations with h_ψ, since it uses two different modalities and may capture features f_θ and g_ϕ can't. However, the trade off is that h_ψ will likely need higher model capacity and more data in order to do so.