1 Variational Autoencoder (VAE)

(a) Explain why a standard Autoencoder cannot directly be used to generate new samples.

A: For a standard Autoencoder, its latent space distribution is not explicitly designed (unknown), therefore we cannot directly sample a latent space vector and generate samples from a decoder.

(b) Explain how the KL-divergence term $D_{KL}(q(z|x)||p(z))$ in the loss regularizes the latent space

A: The latent distribution $p(z)$ in VAE is pre-defined (often Gaussian). The KL-divergence term pushes the posterior $q(z|x)$ to align with $p(z)$. This prevents the encoder from learning a single point in the latent space instead of a distribution. If the encoder only learns a single point, the encoder-decoder outputs a low-dimensional copy of the input, rather than learns meaningful latent representations of the input distribution.

(c) We assume a standard Gaussian prior $p(z)$ to compare against the learned latent encoder distribution $q(z|x)$. Why does the encoder not always predict constant $\mu(x) = 0$ and $\Sigma(x) = I$ for all inputs?

A: Although prior $p(z)$, posterior $p(z|x)$ should be standard Gaussian, and likelihood $p(x|z)$ should be some pre-defined distribution (Gaussian, Bernoulli, etc.) as well, the evidence $p(x) = \int p(x|z)p(z)dz$ is then be a mixture of Gaussian or even more stochastic distribution, leading to non-standard-Gaussian means and covariance matrices.

From another angle, the reconstruction term in the VAE loss function pushes output to be as similar to input as possible, rather than to $\mu(x) = 0$ and $\Sigma(x) = I$.

(d) VAEs with a Gaussian likelihood $p(x|z)$ often produce blurry images compared to Generative Adversarial Networks (GANs). Explain why this happens.

A: Since decoder $p(x|z)$ is modeled as Gaussian, the loss function compares the input with the mean and variance of the generated images only. Because of the variance term, VAE loss function penalizes generated sharp images that largely deviate from the Gaussian mean. This encourages the decoder to generate blurry images.

GANs, on the other hand, train a discriminator to identify each image, not in an average sense. The decoder is not constrained by Gaussian distribution and can explore the data space to generate sharp images as long as the image can "fool" the discriminator.

2 Generative Adversarial Network (GAN)

(a) Since true data and generated data are **mutually exclusive**:

$$p_{\text{data}}(x) = 0 \quad \text{if} \quad x \sim p_{\text{gen}}(x)$$

, and

$$p_{\text{gen}}(x) = 0 \quad \text{if} \quad x \sim p_{\text{data}}(x)$$

Therefore, the optimal discriminator

$$D^*(x) = \begin{cases} \frac{0}{0+p_{\text{gen}}(x)} = 0 & \text{if} \quad x \sim p_{\text{gen}}(x) \\ \frac{p_{\text{data}}(x)}{p_{\text{data}}(x)+0} = 1 & \text{if} \quad x \sim p_{\text{data}}(x) \end{cases}$$

$$\implies D^*(x) = \begin{cases} 1, & x \sim p_{\text{data}}(x) \\ 0, & x \sim p_{\text{gen}}(x) \end{cases}$$

So the loss functions

$$L_D = -\mathbb{E}_{x \sim p_{\text{data}}}[\log(1)] - \mathbb{E}_{x \sim p_{\text{gen}}}[\log(1-0)] = 0$$

$$L_G = \mathbb{E}_{x \sim p_{\text{gen}}}[\log(1-0)] = 0$$

- The optimal discriminator perfectly distinguishes generated data from real ones.
- The generator does the worst job possible. (maximum L_G)
- With the current form of L_G, the gradient vanishes. The training fails to converge.

$$L_G = \mathbb{E}_{p_{x \sim \text{gen}}}\left[\log\left(1 - D^*(x)\right)\right]$$

x comes from non-linear transformation of a latent z with parameters θ

$$D^*(x) = D^*(G_\theta(z))$$

$$\nabla_\theta L_G = \mathbb{E}_{x \sim p_{\text{gen}}}\left[\frac{1}{1 - D^*} \cdot (-1) \cdot \frac{\partial D^*}{\partial x} \cdot \frac{\partial x}{\partial \theta}\right]$$

Since

$$D^*(x) = 0 \quad \text{for } x \sim p_{\text{gen}}(x)$$

$$\implies \frac{\partial D^*}{\partial x} = 0 \quad \implies \nabla_\theta L_G = 0$$

(b)
$p_{\text{data}}(x) = p_{\text{gen}}(x)$

$$D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_{\text{data}}(x)} = 0.5$$

$$L_D = -2\mathbb{E}_{x\sim p_{\text{data}}}[\log(0.5)] = 2\log 2$$

$$L_G = \mathbb{E}_{x\sim p_{\text{data}}}[\log(0.5)] = -\log 2$$

- The generator generates exactly the real images.
- The optimal discriminator cannot distinguish generated data from true ones at all, giving a 50/50 guess for each image. (minimum L_G)

3 GAN Training

(a) Define mode collapse in the context of GANs

A: The generator of GANs might learn only one or several modes of the true distribution, rather than the whole distribution. Furthermore, once the discriminator is able to distinguish the generated images from a certain mode, the generator might jump to some other modes to escape. The game becomes hide and seek.

(b) Explain why mode collapse happens during GAN training. Include in your explanation the roles of both the generator and the discriminator

A: During training, in each iteration, the discriminator only "sees" the sample image from one class only, leading to a gradient descent towards one mode only. Then to reduce L_G, the generator refrains from generating images from that mode, jumping to some other modes. The gradients of the discriminator then track the generator's behavior and focus solely on the target modes and so on.

(c) Describe and explain a specific method that can be used to address mode collapse.

A: Minibatch discrimination encodes the mutual distance into training: in an intermediate layer of the discriminator, the model encodes a defined distance between the current image and the rest of the images in the same minibatch, and passes the information foward to the last layer of the discriminator. The discriminator considers the whole relative distance information as *side information*, which encourages the discriminator to increase diversity.