GEN AI ASSIGNMENT 1 – 22K-4080

TASK 1:

Video focuses primarily on the standard Variational Autoencoder (VAE) and then introduces the concept of **Disentangled Variational Autoencoders**, noting that the primary change to achieve this disentanglement is the addition of a hyperparameter to the loss function.

Difference between VAE and β-VAE (Disentangled VAE)

Objective:

- Standard VAE Prioritizes accurate **data reconstruction** while ensuring the latent distribution roughly follows a standard normal prior.
- While β-VAE, prioritizes **disentanglement** and **interpretability** of the latent factors by imposing a stronger constraint on the latent distribution.

Latent Space:

- In Standard VAE, the latent variables might be highly correlated and difficult to interpret (not "disentangled").
- While in β-VAE, the latent variables are forced to become **uncorrelated** and interpretable, with each variable corresponding to a single, distinct generative factor (e.g., rotation, scale, or color).

Both the standard VAE and the β-VAE loss functions are based on the same two fundamental components: the **Reconstruction Loss** and the **KL Divergence**.

1. Total VAE Loss (ELBO)

The total loss for a VAE (which the video covers from **[06:16]**) is the sum of the reconstruction loss and the KL divergence term.

$$\mathcal{L}_{\mathrm{VAE}}(\theta, \phi) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]}_{\text{Reconstruction Loss}} - \underbrace{\mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z}))}_{\text{KL Divergence}}$$

2. Total β-VAE Loss (Disentangled VAE)

The β-VAE modifies the standard VAE loss by introducing the β hyperparameter, which explicitly weights the KL divergence term (as mentioned around **[09:31]**).

$$\mathcal{L}_{\beta\text{-VAE}}(\theta,\phi) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]}_{\text{Reconstruction Loss}} - \underbrace{\beta \cdot \mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))}_{\text{KL Divergence Term}}$$

× If β=1, the formula reverts to the standard VAE loss.
× If β>1 (which is typical for β-VAE), the importance of the KL divergence is amplified, leading to stronger pressure for disentanglement.

TASK 2:

Given Inputs: x1=0.34 and x2=0.55. **Activation Function:** ReLU(z)=max(0,z).

Assumed Weights (W(1)):

1.0, 0.5, 1.0, −0.5, 1.0, 1.0

W(2):

0.5, 0.5, 0.5

W(3):

1.0, 0.8, 1.2

W(4):

0.4, 0.3, 0.3, 0.4, 0.3, 0.3

Forward Pass:

Input Layer (L0)

Initial State: a(0)=[0.34,0.55]

1.Hidden Layer L1

z1(1)=(1.0·0.34)+(−0.5·0.55)=0.34−0.275=0.065

z2(1)=(0.5·0.34)+(1.0·0.55)=0.17+0.55=0.72

z3(1)=(1.0·0.34)+(1.0·0.55)=0.34+0.55=0.89

- ReLU(z)=a(1)

a1(1)=max(0,0.065)=0.065

a2(1)=max(0,0.72)=0.72

a3(1)=max(0,0.89)=0.89

2.Bottle-Neck Layer L2 (Node 4)

- **Weighted Sum (z4(2)):**

  $z4(2)=(0.5 \cdot 0.065)+(0.5 \cdot 0.72)+(0.5 \cdot 0.89)$

  $z4(2)=0.0325+0.36+0.445=0.8375$

- **Activation (a4(2)):**
  - $a4(2)=ReLU(0.8375)=0.8375$

3. Hidden Layer L3 (Nodes 5, 6, 7):

$z5(3)=1.0 \cdot 0.8375=0.8375$

$z6(3)=0.8 \cdot 0.8375=0.670$

$z7(3)=1.2 \cdot 0.8375=1.005$

- $ReLU(z)=a(3)$

$a5(3)=0.8375$

$a6(3)=0.670$

$a7(3)=1.005$

4. Output Layer L^N (Nodes 8, 9):

(x^1) -> $z8(4)=(0.4 \cdot 0.8375)+(0.3 \cdot 0.67)+(0.3 \cdot 1.005)$ -> $z8(4)=0.335+0.201+0.3015=0.8375$

(x^2) -> $z9(4)=(0.3 \cdot 0.8375)+(0.4 \cdot 0.67)+(0.3 \cdot 1.005)$ -> $z9(4) = 0.25125+0.268+0.3015=0.82075$

- ReLU(z)=x^

x^1=ReLU(0.8375) = 0.8375

x^2=ReLU(0.82075) ≈ 0.8208

Final Answer:

x^≈(0.8375,0.8208)

Task 3:

The architecture is now modified at the L2 Bottleneck Layer (Node 4) to output the Mean (μ) and Log Variance (logσ2) for the latent distribution.

Assumed Weights and Inputs:

- **Inputs:** $a(0)=[0.34,0.55]$
- **L1 Activations (from previous calculation using W(1)):** $a(1)=[0.065,0.72,0.89]$
- **$W\mu(2)$ and $W\log\sigma 2(2)$ (Weights L1→L2):** All values are fixed at 0.5 as requested.

1.Encoder Output: μ and logσ2:

$z=0.5\cdot a1(1)+0.5\cdot a2(1)+0.5\cdot a3(1)$

$z=0.5\cdot(0.065+0.72+0.89)=0.5\cdot(1.675)=0.8375$

$\mu=[0.8375]$

$\log\sigma 2=[0.8375]$

2. Standard Deviation (σ):

$\sigma=\text{sqrt}(\exp(\log\sigma 2) = \text{sqrt}(\exp(0.8375)) \approx \text{sqrt}(2.3094) \approx 1.5197$

3. Reparameterization Trick and Sampling

Let $\epsilon=[0.1]$

$z=0.8375+1.5197\cdot 0.1\approx 0.8375+0.1520=0.9895$

$a4(2)\approx 0.9895$

4. Decoder Layers L3 and LN:

- **Layer L3 Activations a(3):**
  - $a5(3)=\text{ReLU}(1.0\cdot 0.9895)=0.9895$
  - $a6(3)=\text{ReLU}(0.8\cdot 0.9895)\approx 0.7916$
  - $a7(3)=\text{ReLU}(1.2\cdot 0.9895)\approx 1.1874$
- **Layer LN Output x^:**
  - $x\text{^}1$: $(0.4\cdot 0.9895)+(0.3\cdot 0.7916)+(0.3\cdot 1.1874)\approx 0.9895$
  - $x\text{^}2$: $(0.3\cdot 0.9895)+(0.4\cdot 0.7916)+(0.3\cdot 1.1874)\approx 0.9697$

2) Backward Pass Requirements

The primary challenge in the VAE backward pass is calculating the gradient through the sampling operation. The **Reparameterization Trick** solves this by expressing the stochastic node z as a deterministic function of μ, σ, and the random variable ϵ.

The overall VAE loss (LVAE) has two components: LRec (Reconstruction Loss) and LKL (KL Divergence Loss). The goal is to find the change in weights ΔW(2) for the mean (Wμ(2)) and log variance (Wlogσ2(2)) layers.

1) The total gradient for the mean (μ) and log variance (logσ2) is the sum of contributions from both the Reconstruction and KL terms:

$$\frac{\partial \mathcal{L}_{\text{VAE}}}{\partial \mu} = \frac{\partial \mathcal{L}_{\text{Rec}}}{\partial \mathbf{z}} \cdot \frac{\partial \mathbf{z}}{\partial \mu} + \frac{\partial \mathcal{L}_{\text{KL}}}{\partial \mu}$$

$$\frac{\partial \mathcal{L}_{\text{VAE}}}{\partial \log \sigma^2} = \frac{\partial \mathcal{L}_{\text{Rec}}}{\partial \mathbf{z}} \cdot \frac{\partial \mathbf{z}}{\partial \log \sigma^2} + \frac{\partial \mathcal{L}_{\text{KL}}}{\partial \log \sigma^2}$$

2) Once ∂LVAE/∂μ and ∂LVAE/∂logσ2 are found using the derivatives above, the gradient is propagated back to the weights W(2) using the output activations of the L1 layer (a(1)):

$$\frac{\partial \mathcal{L}_{\text{VAE}}}{\partial W_{\mu}^{(2)}} = \frac{\partial \mathcal{L}_{\text{VAE}}}{\partial \mu} \cdot a^{(1)}$$

$$\frac{\partial \mathcal{L}_{\text{VAE}}}{\partial W_{\log \sigma^2}^{(2)}} = \frac{\partial \mathcal{L}_{\text{VAE}}}{\partial \log \sigma^2} \cdot a^{(1)}$$

3) These calculated gradients are then used by the optimizer (e.g., Gradient Descent) to determine the weight update.