



Parcial IA

Autoencoder

Codificador: Mapea la entrada \mathbf{x} a una representación latente \mathbf{z}

Decodificador: Mapea la representación latente \mathbf{z} a una reconstrucción de la entrada $\hat{\mathbf{x}}$

Modelo: $f(\cdot)$

$$Eo : \mathbf{x} \longrightarrow \mathbf{z}$$

$$Do : \mathbf{z} \longrightarrow \hat{\mathbf{x}}$$

Codificador:

$$\mathbf{z} = f(\mathbf{x}; \mathbf{W}_e, \mathbf{b}_e) = \varphi(\mathbf{W}_e \mathbf{x} + \mathbf{b}_e)$$

\mathbf{W}_e : Matriz de pesos del encoder φ : Función de activ.

\mathbf{b}_e : Vector de sesgos del encoder \mathbf{z} : Latent represent.

Decodificador:

$$\hat{\mathbf{x}} = g(\mathbf{z}; \mathbf{W}_d, b_d) = \varphi(\mathbf{W}_d + b_d)$$

\mathbf{W}_d : Matriz de pesos del Decodificador φ : Función de activ.

b_d : Vector de sesgos del Decodificador $\hat{\mathbf{x}}$: Reconstrucción de la entrada

Función de costo (loss):

MSE

$$L(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{N} \sum_{i=1}^N \| \mathbf{x}_i - \hat{\mathbf{x}}_i \|^2$$
$$= \frac{1}{N} \sum_{i=1}^N \sum_{j=1}^d (\mathbf{x}_{ij} - \hat{\mathbf{x}}_{ij})^2$$

Función de optimización:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} L(\mathbf{x}, \hat{\mathbf{x}}, \theta) ;$$

$$\theta = \{\mathbf{W}_d, \mathbf{W}_e, b_d, b_e\}$$

$$L(X, \hat{X}, \theta) = E_x \left\{ L(X, g_\theta(f_\theta(X))) \right\}$$

Autoencoder regularizado

Codificador:

$$f(X; \theta_e) = \varphi(W_e X + b_e) = Z$$

Decodificador:

$$g(Z; \theta_d) = \varphi(W_d Z + b_d) = \hat{X}$$

Función de costo (loss):

$$L(X, \hat{X}, \theta) = E_x \left\{ L(X, g(f_\theta(X); \theta_e); \theta_d) \right\}$$

$$+ \lambda L(\theta)$$

$$\Theta = \{\theta_s, \theta_d\} = \{W_f, b_e, W_d, b_d\}$$

$$L(X, \hat{X}) = - \sum_{i=1}^n (X_i \log(\hat{X}_i) + (1-X_i) \log(1-\hat{X}_i))$$

$$J(\theta) = \frac{1}{2} (\|W_f\|^2 + \|W_d\|^2)$$

Función de optimización

$$\hat{\theta} = \arg \min_{\theta} E_g L(X, g(f(X; \theta_s); \theta_d))$$

$$J \lambda L(\theta)$$

Gradiente descendente

$$\theta^{(t+1)} = \theta^{(t)} - \eta \nabla_{\theta} L(\theta)$$

t: Iteración actual

n: tasa de aprendizaje

$\nabla_{\theta} L(\theta)$: Gradiente

Autoencoder Variacional

Un autoencoder Variacional (VAE) es una extensión del ya conocido autoencoder que introduce una interpretación probabilística en el encoder permitiendo la generación de nuevas muestras a partir de una distribución latente.

Modelo

$$f(\cdot)$$

$$E_\theta : X \longrightarrow Z$$

$$D_\phi : Z \longrightarrow \tilde{X}$$

$$g(\cdot)$$

Codificador:

Transformamos los datos de \mathbf{X} a una distribución latente \mathbf{Z} :

$$q_{\theta}(\mathbf{Z} | \mathbf{X})$$

tal que:

$$\mu, \log \sigma^2 = f(\mathbf{x}; \theta_f)$$

asumimos un prior, en este caso gaussiano!

$$p(\mathbf{z}) \sim N(\mathbf{z}; 0I)$$

$$q_{\theta}(\mathbf{z} | \mathbf{x}_n; \theta) \sim N(\mathbf{z}; \mu, G^2 I)$$

reparametrización:

$$\xi \sim N(0, 1)$$

$$\mathbf{z} = \mu(\mathbf{x}) + G(\mathbf{x}) \odot \xi$$

$$f(\mathbf{x}) = q_{\theta}(\mathbf{z} | \mathbf{x}; \theta) = z$$

Decodificador:

$$g(\mathbf{z}; \theta_d) = g(u_s(\mathbf{x}) + G_s(\mathbf{x}) \odot \xi; \theta_d) = \hat{\mathbf{x}}$$

$$\hat{\mathbf{x}} = g(f(\mathbf{x}; \theta_e); \theta_d)$$

$$= \psi(\sqrt{\lambda_d} \mathbf{z} + b_d)$$

Función de costo:

Prior

$$\lambda_1 D_{KL}\left(g_\theta(\mathbf{z} | \mathbf{x}_n; \theta_e) \| \overbrace{P(\mathbf{z} | \mathbf{x}_n; \theta)}^{\text{aprox. posterior}}\right) + \lambda_2 \underbrace{J(\mathbf{x}_n, \hat{\mathbf{x}}_n)}_{\text{reconstrucción}}$$

Loss de
reconstrucción

$$D_{KL}\left(g_\theta(\mathbf{z} | \mathbf{x}_n; \theta_e) \| P(\mathbf{z} | \mathbf{x}_n; \theta)\right)$$

$$\Rightarrow E_{\theta_e} \left\{ \log \frac{g_\theta(\mathbf{z} | \mathbf{x}_n; \theta_e)}{P(\mathbf{z} | \mathbf{x}_n; \theta)} \right\}$$

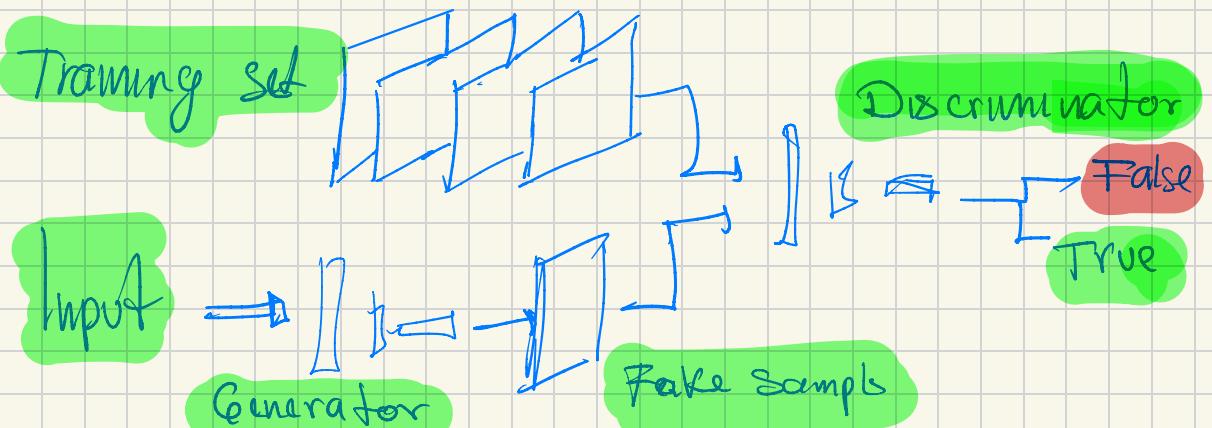
$$= \sum g_\theta(\mathbf{z} | \mathbf{x}) \log \frac{g_\theta(\mathbf{z} | \mathbf{x})}{P(\mathbf{z} | \mathbf{x})} = E_{\theta_e} \left(\log \frac{g(\mathbf{z} | \mathbf{x})}{P(\mathbf{z} | \mathbf{x})} \right)$$

Optimizador

$$\hat{\theta} = \underset{\theta}{\operatorname{arg\,min}} \lambda_1 DKL + \lambda_2 L_{\text{reconstruc}}(X, \hat{X})$$

Redes Generativas adversarias (GAN's)

Como su nombre lo indica las gans se conforman de un generador y un discriminador, uno que genera muestras nuevas y el otro que discrimina si son falsas o verdaderas.



Generador:

$$X = f(Z) = \varphi(W_g Z + b_g)$$

Z : Input (noise) φ : Activation function

X : Fake sample W_g, b_g : Parameters

Discriminador:

$$\hat{X} = g(X) = \varphi(W_d X + b_d)$$

donde $\hat{X} \in [0, 1]$

Funciones de Costo:

Generador:

$$J_g = -\frac{1}{Z - p(X)} \left\{ \log g(f(Z)) \right\}$$

Discriminador

$$d_0 = -\frac{1}{N} \sum_{i=1}^N \log \left(g(x_i) \right)$$

$$+ E_{\epsilon} \left\{ \log \left(1 - \left(g(f(\epsilon)) \right) \right) \right\}$$

Optimizador

$$\hat{\theta}_0, \hat{\theta}_G = \min_{\theta_0} \max_{\theta_G} L(\theta_0) - d_G(\theta_G)$$