BLG561E FALL 2021

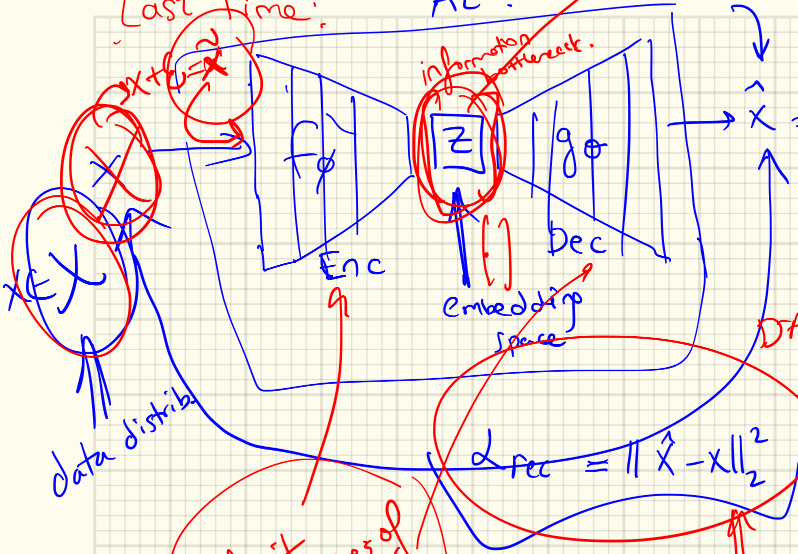
Deep Learning

07.12.2021

Gözde ÜNAL

Last Time:

AE :



$\hat{x} = h_{\theta,\phi}(x) = g_\theta(f_\phi(x))$

① Autoencoder hypothesis

$\mathcal{L}_{rec} = \|\hat{x} - x\|_2^2$

limit capacities of NNs

Enc

Dec.

$\mathcal{L}_{rec} = \|h_{\theta,\phi}(\tilde{x}) - x\|_2^2$

DAE Inference

Test

$x_{test} \rightarrow$ $z + \Delta z$ $\rightarrow x_{new}$.

② Loss

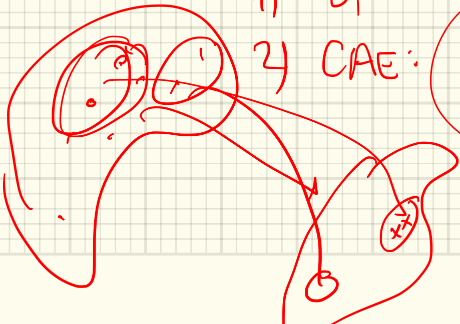
$+ \lambda$ Regularizers.

1) Sparse AE $\Rightarrow \|z\|_1$

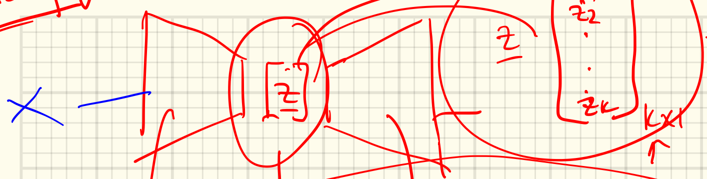
2) CAE: $\left\|\frac{\partial z}{\partial x}\right\|_F^2$ small.

$\frac{\partial f}{\partial x} \rightarrow$ Jacobian matrix

3) Denoising AE



VAE: Variational Autoencoder:



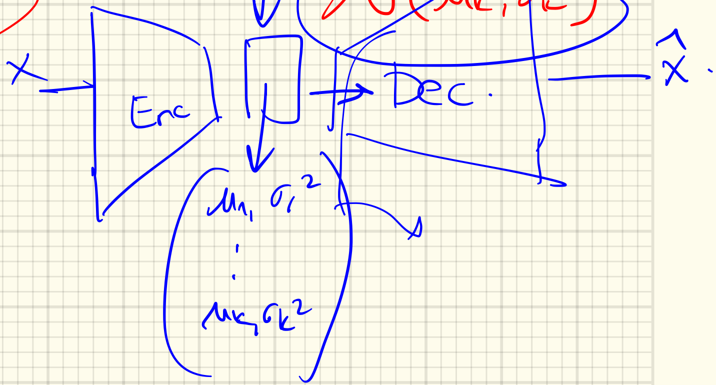
Smooth but parametric latent representation

$p_{encoder}(z|x)$ — Inference

$p_{decoder}(x|z)$

VAE $\begin{bmatrix} \mu_1, \sigma_1^2 \\ \mu_2, \sigma_2^2 \\ \vdots \\ \mu_k, \sigma_k^2 \end{bmatrix}_{2 \times k}$ → sampling from Gaussian. latent space

$\mathcal{N}(\mu_k, \sigma_k^2)$



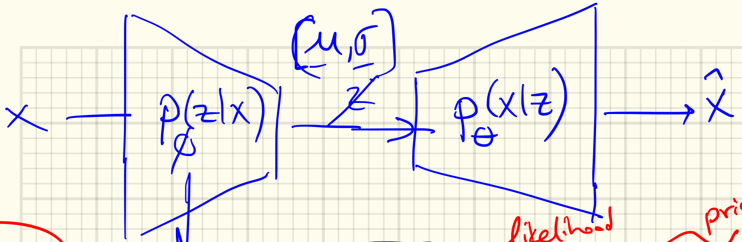
$P_{z|x}$ (inference) distrib. posterior "

x → Z: hidden / latent

$P_{x|z}$ (generation). likelihood. distrib.

Stochastic AEs

→ Variational Inference.



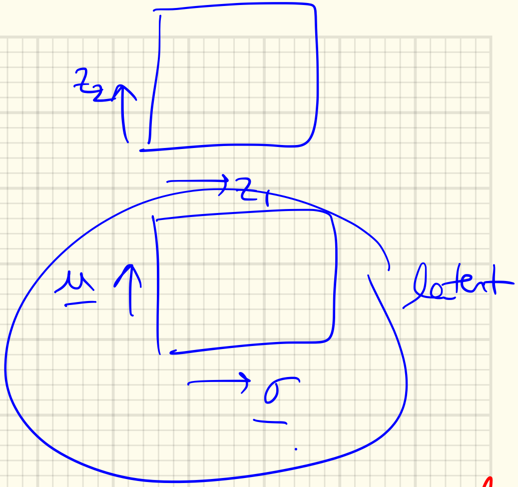
posterior distrib

$$p(z|x) = \frac{p(x,z)}{p(x)} = \frac{\overbrace{p(x|z)}^{\text{likelihood}}\,\overbrace{p(z)}^{\text{prior}}}{\underbrace{p(x)}_{\text{Evidence.}}}$$

$$p(x) = \underbrace{\int\int\int\int\int\int\int}_{\text{huge \# dimensions}} p(x|z)\,p(z)\,dz$$

z unobserved for each input x.

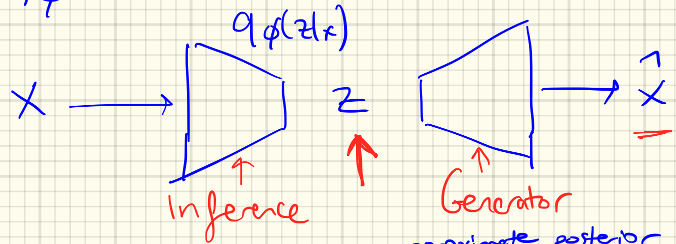
→ Intractable integral.



Intractable both $p(z|x)$ & $p(x)$

→ Variational Inference tries to approximate $p(z|x)$ by an approximate posterior distribution.

→ $q_\phi(z|x)$: approximate posterior distribution :



approximate ... true

$q_\phi(z|x) \approx p(z|x)$

$$KL(p_1 || p_2) = \sum p_1 \log \frac{p_1}{p_2}$$

☆ **VAE** wants to make q (approximate posterior) as "close" as possible to the true $p(z|x)$ — posterior distrib (intractable)

$q_\phi(z|x)$: tractable posterior distribution

e.g. a Gaussian distribution

$$\phi = (\mu_i, \Sigma_i)$$

$$\begin{bmatrix} \sigma_1^2 & & 0 \\ & \sigma_2^2 & \\ 0 & & \ddots \; \sigma_k^2 \end{bmatrix}$$

: uncorrelated across dimensions of the hidden code.

Q. How do I measure closeness of 2 probability distributions ?

A. Resort to Information-theoretic Divergence measures like $\boxed{KL}$ or Jensen-Shannon Divergence. ⟹

$\Rightarrow$ Minimize $\underbrace{KL}_{\text{Kullback-Leibler div. between } q \text{ \& } p}(q \| p) \triangleq \sum q \log \frac{q}{p} = -\sum q \log \frac{p}{q}$

$$\min \; KL\left(q_\phi(z|x) \,\|\, p(z|x)\right)$$

$$KL = -\sum_z q(z|x) \log \frac{p(z|x)}{q(z|x)} = -\sum_z q(z|x) \log \frac{p(z,x)}{q(z|x)} \cdot \frac{1}{p(x)}$$

$$= -\sum_z q(z|x)\left[\log \frac{p(x,z)}{q(z|x)} - \log p(x)\right]$$

$$KL = -\sum_z q(z|x) \log \frac{p(x,z)}{q(z|x)} + \underbrace{\sum_z q(z|x)}_{=1} \boxed{\log p(x)}$$

Any pdf $p(x)$ satisfies
i) $p(x) \geq 0$
ii) $\sum_x p(x) = 1$

$$\to \underbrace{\log p(x)}_{\text{const.}} = \underbrace{KL\left(q(z|x) \,\|\, p(z|x)\right)}_{\text{KL div. properties } \geq 0} + \underbrace{\sum_z q(z|x) \log \frac{p(x,z)}{q(z|x)}}_{\mathcal{L}_{VB}}$$

$\mathcal{L}_{VB}$: called the Variational Lower Bound for $\log p(x)$.

$$\to \boxed{\log p(x) \geq \mathcal{L}_{VB}}$$

want to

$\log p(x) \geq \mathcal{L}_{VB}$

$\max \log p(x) \longrightarrow$ instead $\equiv \max \mathcal{L}_{VB}$: maximize the variational lower bound

$$\rightarrow \mathcal{L}_{VB} = \sum_z q(z|x) \log \frac{\overbrace{p(x|z)\, p(z)}^{p(x,z)}}{p(z|x)}$$

$$\mathcal{L}_{VB} = \underbrace{\sum_z q(z|x) \log p(x|z)}_{} + \underbrace{\sum_z q(z|x) \log \frac{p(z)}{q(z|x)}}_{-KL\left(q(z|x)\,\|\,p(z)\right)}$$

$p(z)$: model distribution $\sim \mathcal{N}(0,1)$

$$\max \mathcal{L}_{VB} = \underbrace{\mathbb{E}_{q(z|x)}\left[\log p(x|z)\right]}_{\text{log likelihood of } x \text{ given } z} \underbrace{- KL\left(q(z|x)\,\|\,p(z)\right)}_{}$$

Data Term : $\boxed{\mathcal{L}_{Data} + \lambda\, \mathcal{L}_{Reg}}$.

eg. Reconstruction Loss

Regularizer Term : minimizes the KL btw approx posterior $q(z|x)$ & $p(z)$

→ or minimize $-\mathcal{L}_{VB}$: $\boxed{\phi}$ $\boxed{\theta}$

Given $\{x^i\}_{i=1}^m$

$$\min \mathcal{L}(\theta, \phi) = \sum_{i=1}^{m} \underbrace{\mathcal{L}_{rec}}_{-E_{z\sim q_\phi(z|x^i)}[\log p_\theta(x^i|z)]} + \overset{\uparrow}{\lambda} \cdot KL(q_\phi(z|x^i) \,||\, \underbrace{p(z)}_{\mathcal{N}(0,1)})$$

(data)

$$+ \lambda \sum_j KL(q_j(z|x^i) \,||\, \underbrace{\mathcal{N}(0,1)}_{\mathcal{N}(\mu_j, \sigma_j^2)})$$

j (for every dimension of the latent space

$\mathcal{N}(\mu_j, \sigma_j^2)$

VAE paper : (Kigma, Welling, 2014)

latent space is parametric w/ 2 parameters ϕ, θ — μ_j, σ_j for each dimension z_j.

$$x^i \longrightarrow q_\phi \longrightarrow z \longrightarrow p_\theta \longrightarrow \hat{x}^i$$

b/c $\exists$ sampling, we need smt. special for backpropping.

$$\boxed{z_j \sim \mathcal{N}(\mu_j, \sigma_j^2)},$$

we need **Reparameterization Trick**:

$$\underset{\text{Input}}{X} \xrightarrow{\text{MLP 1}} h \xrightarrow{\text{MLP 2}} (\mu_j, \sigma_j)$$

Say for example, we have a 2-layer MLP Encoder:

$$h = \tanh(\underbrace{W^1 X + b^1})$$: activation at 1st layer:

$$\to \mu_j = W^2 h^{(x)} + b^2$$

$$\to \boxed{\log \sigma_j^2} = W^3 h^{(x)} + b^3 \longrightarrow \sigma_j^2 = \exp(\underbrace{\log \sigma_j^2})$$

ensures σ^2 is +ve.

to keep the variance > 0

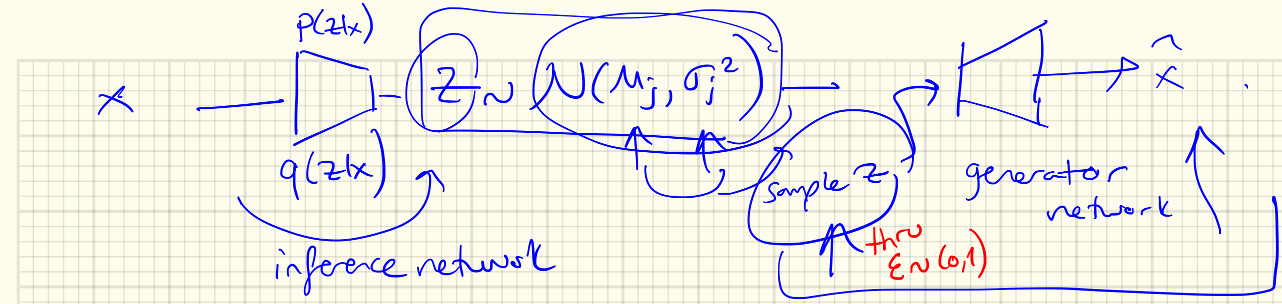
now, we have (μ_j, σ_j^2) from the encoder q_ϕ

$$\mathcal{N}(\mu_j, \sigma_j^2) \longrightarrow z_j$$: sampled by

$$z_j = \mu_j + \sigma_j \cdot \varepsilon, \quad \underbrace{\varepsilon \sim \mathcal{N}(0,1)}$$

ε : sampled from standard normal.

Helps us to be able to differentiate w.r.t. μ_j's & σ_j's during backprop.



Summarize VAE

Loss terms is based on Variational Lower Bound:

1) Data Likelihood $\approx$ Reconst loss

2) Regularizer: KL loss btw $\rightarrow$ approximate inference distrib & a Gaussian distrib.

Generative Adversarial Networks (GANs) / Implicit Generative Models

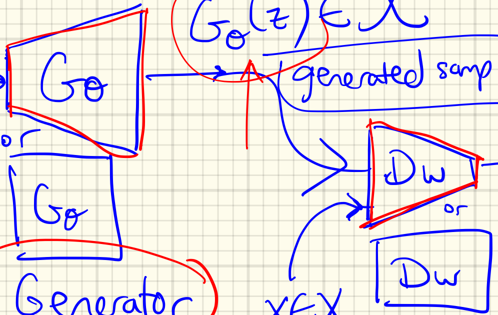
Problem: To learn a generative model of a distribution of data points $P_{\theta\,data}(x)$, implicitly.

Given a set of data sample x from X, $x \in X$, learn a data distribution in the form of a deep NN $G_\theta(\cdot)$, w/ parameters θ.

G takes as input a sample Noise $z \in F$

$z \sim \mathcal{N}(0,1)$

$z \sim U(0,1)$



Generator

$G_\theta(z) \in X$ generated samples

$x \in X$ true data samples

Critic / Discriminator

$\{0/1\}$ real / fake

pushes the GAN towards realistic looking outputs.

GAN: creates a ZERO_SUM Game between the Generator & Discriminator

Generator tries to maximize error of the D network → zero-sum game.

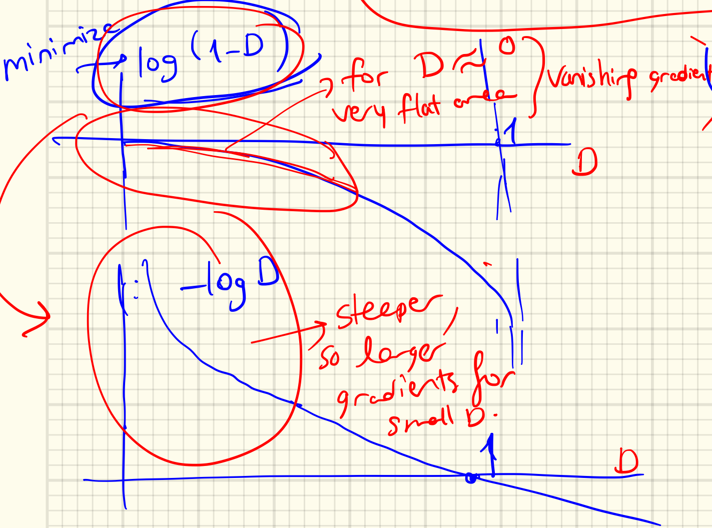
Vanilla GAN : $D_w(x) \in [0,1]$ w/ BCE loss.

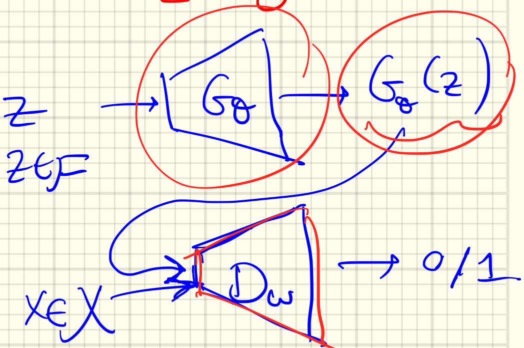
(Goodfellow 2014)

Optimization problem of the vanilla GAN

$$\min_{\theta} \max_{w} \; \underbrace{\mathbb{E}_{x \sim p_{data}(\cdot)}\left[\log D_w(x)\right] + \mathbb{E}_{z \sim F}\left[\log\left(1 - D_w(G_\theta(z))\right)\right]}_{\mathcal{L}(\theta, w)}$$

$$\mathbb{E}\left[-\log D_w(G_\theta(z))\right]$$





Discriminator : max $\mathcal{L}^D = \mathcal{L}(\theta, \omega)$

Generator : min $\mathcal{L}^G = -\mathcal{L}^D$

$\mathcal{L}^D + \mathcal{L}^G = 0$

zero-sum game.

GAN Loss used in practice

$$\min \ \log(1 - D(G(z))) \rightarrow \min \ -\log D(G(z))$$

Loss

$$\min_\theta \max_\omega E_x[\log D_\omega(x)] - E_z[\log D(G(z))]$$

(Goodfellow 2014) : Generator minimizes the Jensen-Shannon (JS) divergence between the true distribution & the generator distribution

Reading assignment

WGAN :

Wasserstein GAN : [Arjovsky et al] WGAN

Instead JS divergence, they optimize Earth_mover distance (Emd) (Wasserstein - 1 metric.) between the true distrib & generator distrib.

WGAN leads to the loss function

WGAN loss:

2

$$\boxed{\min_{\theta}} \; \boxed{\max_{w}} \; \underbrace{E_{x \sim p_{data}}\left[D_w(x)\right] - E_{z \sim F}\left[D_w(G_\theta(z))\right]}_{L(\theta, w)}$$

Note: Generator only operates on this term.

3 $\boxed{\text{Optimization}}$? ≡ Training of GANs ?

→ Is $\mathcal{L}(\theta, w)$ a $\underbrace{\text{convex}}_{\text{in } \theta}$ - $\underbrace{\text{concave}}_{\text{in } w}$ function? No!

Optimization: Via SGD (or its variants) Train parameters θ & w in an alternate fashion.

$$\min_{\theta} \max_{w} \mathcal{L}(\theta, w) \longrightarrow \frac{\text{Iterative Soln.}}{\text{SGD.}}$$

SGD

$$\nabla_w \mathcal{L}$$

$$\nabla_\theta \mathcal{L}$$

for $t \in \{1, \dots, T-1\}$; with $\nabla_{\omega,t} \mathcal{L}(\overset{\text{fixed}}{\theta_t}, \omega_t)$

$\nabla_{\theta,t} \mathcal{L}(\theta_t, \underset{\text{fixed.}}{\omega_t})$

Gradient Ascent for $\underline{\omega}$:

$$\omega_{t+1} = \omega_t + \eta \, \nabla_{\omega,t} \mathcal{L}$$

Gradient Descent for $\underline{\theta}$:

$$\theta_{t+1} = \theta_t - \eta \, \nabla_{\theta,t} \mathcal{L}$$

Q. Are there any guarantees for an optimal solution to this problem?

(Other than in a convex-concave setting) No guarantees for an optimal solution.

☆ GANs are difficult to train → & even unstable!

LS-GAN:

(Least-Squares GAN)

Rather than cross-entropy loss in the vanilla GAN;

Loss functions:

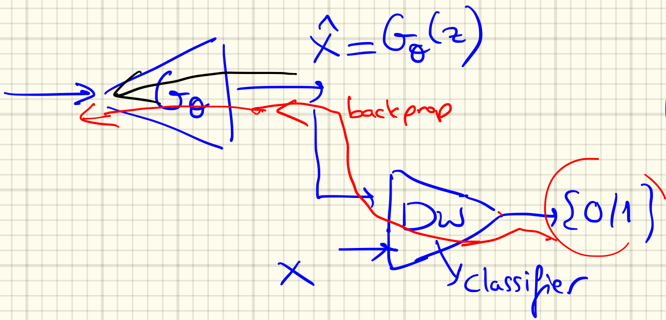
$$D: \min_{w} \mathbb{E}_{x \sim p_{data}}\Big[\big(\underbrace{D_w(x)}_{\text{true data}} - \overset{b}{1}\big)^2\Big] + \mathbb{E}_{z \sim \underbrace{p(z)}_{\mathcal{N}(0,1)}}\Big[\big(D_w(\underbrace{G_\theta(z)}_{\text{Generated}}) - \overset{a}{1}\big)^2\Big]$$

$$G: \min_{\theta} \mathbb{E}_{z \sim p(z)}\Big[\big(D_w(G_\theta(z)) - 1\big)^2\Big]$$

Note: ~~A~~ LSGAN: variant of GAN $\rightarrow$ $\exists$ tons of them

GAN Training Stage:

$z \in \mathcal{N}(0,1)$



Training estimates w & θ.

Now w, & θ are learned:

GAN Inference Stage:

want to Generate new Samples:

No ~~D_w network~~

Sample a $z \sim \mathcal{N}(0,1)$