MAC5921 – Deep Learning

Aula 18 – 26/10/2023

Generative models

Nina S. T. Hirata

$$\boxed{P(\mathbf{y} \mid \mathbf{x})} = \boxed{\frac{P(\mathbf{x} \mid \mathbf{y})\, P(\mathbf{y})}{P(\mathbf{x})}} = \frac{P(\mathbf{x}, \mathbf{y})}{P(\mathbf{x})}$$

Discriminativa — Gerativa

Discriminativa: aprende $P(\mathbf{y} \mid \mathbf{x})$

Generativa: aprende $P(\mathbf{x}, \mathbf{y})$

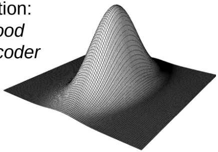
Generative Models

Approximate data distribution P_r with another distribution P_θ

θ = distribution parameters

P_r

Learn prior distribution:
Maximizing Likelihood
Variational Autoencoder



Learn to generate samples following P_θ
Without using directly P_θ
Train a generator only → *GANs*

Generative Models

Gaussian Mixture Models (GMMs)
Hidden Markov Models (HMMs)
Recurrent Neural Networks (RNNs)

2014: Variational Autoencoders (VAEs) https://arxiv.org/abs/1312.6114

2014: Generative Adversarial Network (GAN),
https://arxiv.org/abs/1406.2661

2015: Flow-based models / Diffusion models

2017: Transformers

NeRFs (2020) – Neural Radiance Fields, a technique for generating 3D content from 2D images

Modelagem explícita de $p(x)$ / $p(x, y)$

Gaussian Mixture Models (GMMs)

Probabilistic Graphical models

Curso no IME: MAC6916 – Probabilistic Graphical Models

https://www.ime.usp.br/~ddm/courses/mac6916/

Livro sobre ML com viés mais probabilístico:

- Probabilistic Machine Learning: An Introduction
  Kevin P. Murphy

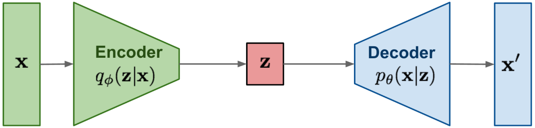
  https://probml.github.io/pml-book/book1.html

- Probabilistic Machine Learning: Advanced Topics
  Kevin P. Murphy

  https://probml.github.io/pml-book/book2.html

VAE (Variational auto-encoder)

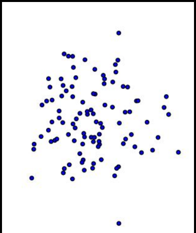
VAE: maximize variational lower bound



https://www.rootstrap.com/blog/how-to-generate-images-with-ai/

Variational Auto-encoder (VAE)

Intuition: given a bunch of random variables that can be sampled easily, we can generate random samples following other distributions, through a complicated non-linear mapping x = f(z)



$$f(z) = z/10 + z/\|z\|$$

→

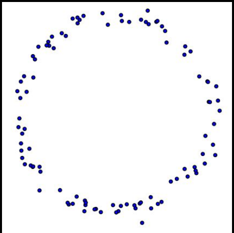
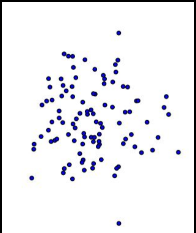


Image Credit: Doersch 2016

Variational Auto-encoder (VAE)

Intuition: given a bunch of random variables that can be sampled easily, we can generate some new random samples through a complicated non-linear mapping x = f(z)



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

$$X \sim \mathcal{N}(f(z;\theta), \sigma^2 I)$$

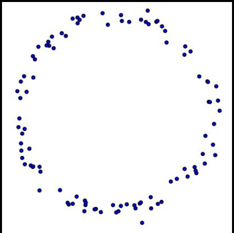
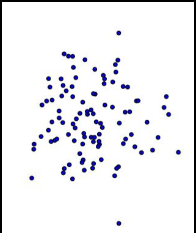


Image Credit: Doersch 2016

Variational Auto-encoder (VAE)

Intuition: given a bunch of random variables, we can generate some new random samples through a complicated non-linear mapping x = f(z)



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

$$X \sim \mathcal{N}(f(z;\theta), \sigma^2 I)$$

Gaussian

NN

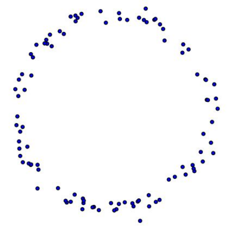
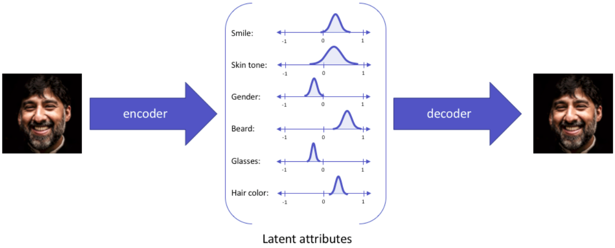


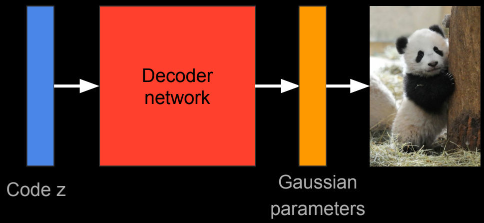
Image Credit: Doersch 2016



https://www.v7labs.com/blog/autoencoders-guide

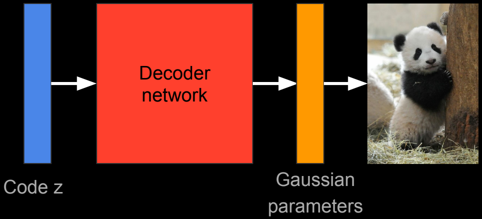
Variational Auto-encoder (VAE)

You can consider it as a decoder!



Variational Auto-encoder (VAE)

How do we learn the parameters of decoder network?



Variational Auto-encoder (VAE)

Review: Marginalization

$$p(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})d\mathbf{z}$$

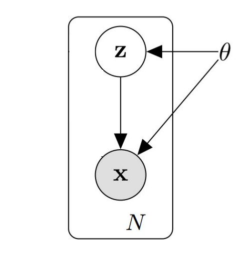


Image Credit: Doersch 2016

Variational Auto-encoder (VAE)

Review: Marginalization

$$p(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})d\mathbf{z}$$

[0, 0.2, -0.5]

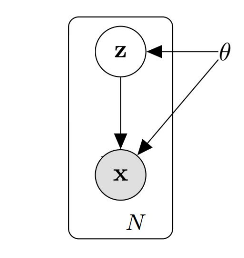


Image Credit: Doersch 2016

Variational Auto-encoder (VAE)

Learning objective: maximize the log-probability

$$\max_{\theta} \sum_{i} \log p_{\theta}(\mathbf{x}_i)$$

Training images should have high probability

$$p_{\theta}(\mathbf{x}) = \int p_{\theta}(\mathbf{x}|\mathbf{z}) p_{\theta}(\mathbf{z}) d\mathbf{z}$$

Image Credit: Doersch 2016

Variational Auto-encoder (VAE)

Learning objective: maximize the log-probability

$$\max_\theta \sum_i \log p_\theta(\mathbf{x}_i)$$

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})d\mathbf{z}$$

Integration over a neural network. Difficult!

Image Credit: Doersch 2016

Variational Auto-encoder (VAE)

Learning objective: maximize the log-probability

$$\max_{\theta} \sum_{i} \log p_{\theta}(\mathbf{x}_i)$$

$$p_{\theta}(\mathbf{x}) = \int p_{\theta}(\mathbf{x}|\mathbf{z}) p_{\theta}(\mathbf{z}) d\mathbf{z}$$

Integration over a neural network. Difficult!

Quiz: Why not do this? $\log p_{\theta}(\mathbf{x}) \approx \log \frac{1}{N} \sum_{j} p_{\theta}(\mathbf{x}|\mathbf{z}_j)$

Image Credit: Doersch 2016

Variational Auto-encoder (VAE)

Learning objective: maximize the log-probability

$$\max_\theta \sum_i \log p_\theta(\mathbf{x}_i)$$

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z}) p_\theta(\mathbf{z}) d\mathbf{z}$$

many sampled z will have a close-to-zero p(x|z)

Quiz: Why not do this? $$\log p_\theta(\mathbf{x}) \approx \log \frac{1}{N} \sum_j p_\theta(\mathbf{x}|\mathbf{z}_j)$$

Image Credit: Doersch 2016

Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

$$\log p_\theta(\mathbf{x}_i) \geq \underline{\mathbb{E}_{q(\mathbf{z})}[\log p_\theta(\mathbf{x}_i|\mathbf{z})] - KL[q(\mathbf{z})||p_\theta(\mathbf{z})]}$$

Variational lower-bound

Quiz: How to choose a good proposal distribution?

Proposal distribution

Variational Auto-encoder (VAE)

Why it is the variational lower-bound?

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x}|\mathbf{z}) p_\theta(\mathbf{z}) d\mathbf{z}$$

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x}|\mathbf{z}) \frac{p_\theta(\mathbf{z})}{q(\mathbf{z})} q(\mathbf{z}) d\mathbf{z}$$

Jenson inequality

$$\log \int p(\mathbf{x}) g(\mathbf{x}) d\mathbf{x} \geq \int p(\mathbf{x}) \log g(\mathbf{x}) d\mathbf{x}$$

$$\log p_\theta(\mathbf{x}) \geq \int q(\mathbf{z}) \log \left(p_\theta(\mathbf{x}|\mathbf{z}) \frac{p_\theta(\mathbf{z})}{q(\mathbf{z})} \right) d\mathbf{z}$$

$$\log p_\theta(\mathbf{x}) \geq \int q(\mathbf{z}) \log p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z} - \int q(\mathbf{z}) \log \frac{p_\theta(\mathbf{z})}{q(\mathbf{z})} d\mathbf{z}$$

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - KL[q(\mathbf{z}) || p_\theta(\mathbf{z})]$$

Kingma et al. 2014

Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

$$\log p_\theta(\mathbf{x}_i) \geq \mathbb{E}_{q(\mathbf{z})}[\log p_\theta(\mathbf{x}_i|\mathbf{z})] - KL[q(\mathbf{z})||p_\theta(\mathbf{z})]$$

Variational lower-bound

Proposal distribution

Quiz: How to choose a good proposal distribution?

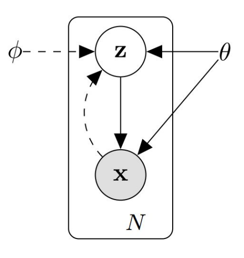
- Easy to sample
- Differentiable wrt parameters
- Given a training sample X, the sampled z is likely to have a non-zero p(x|z)

Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

$$\log p_\theta(\mathbf{x}_i) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i|\mathbf{z})] - KL[q_\phi(\mathbf{z}|\mathbf{x}_i)||p_\theta(\mathbf{z})]$$

Answer: Another neural network + Gaussian to approximate the posterior!



Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

$$\log p_\theta(\mathbf{x}_i) \geq \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i|\mathbf{z})]} - \underbrace{KL[q_\phi(\mathbf{z}|\mathbf{x}_i)||p_\theta(\mathbf{z})]}$$

Reconstruction error:

- Training samples have higher probability

Prior:

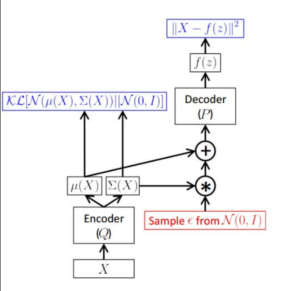
- Proposal distribution should be like Gaussian

Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

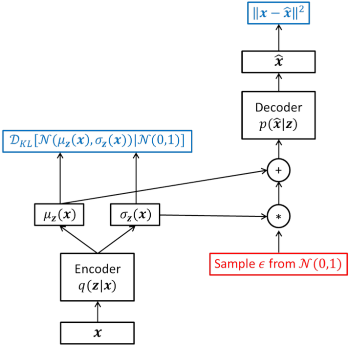
$$\log p_\theta(\mathbf{x}_i) \geq \underline{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i|\mathbf{z})]} - \underline{KL[q_\phi(\mathbf{z}|\mathbf{x}_i)||p_\theta(\mathbf{z})]}$$

- KL-Divergence: closed-form and differentiable if both are Gaussians
- Reconstruction error: approximate by just sampling one z



Computation graph
Credit: Doersch

Variational Autoencoder



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

- Encoder $q(\mathbf{z}|\mathbf{x})$ learns latent parameters $\mu_\mathbf{z}(\mathbf{x}), \sigma_\mathbf{z}(\mathbf{x})$ of Gaussian distribution
- Re-parametrization $z = \mu_\mathbf{z}(\mathbf{x}) + \sigma_\mathbf{z}(\mathbf{x})\epsilon$ with $\epsilon \in \mathcal{N}(0,1)$
- Example: 2D Gaussian
  - Only two independently normal distributed parameters in hidden layer for each input
- $\mathcal{D}_{KL}[\mathcal{N}(\mu_z(\mathbf{x}), \sigma_z(\mathbf{x}))|\mathcal{N}(0,1)]$

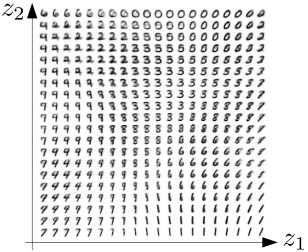
$$= \frac{1}{N}\sum_\mathbf{x}\int_\mathbf{z} dz\ \mathcal{N}(\mu_\mathbf{z}, \sigma_\mathbf{z}) \log \frac{\mathcal{N}(0,1)}{\mathcal{N}(\mu_\mathbf{z}(\mathbf{x}), \sigma_\mathbf{z}(\mathbf{x}))}$$

$$= \frac{1}{N}\sum_\mathbf{x}\frac{1}{2}(1 + \log\sigma^2_\mathbf{z}(\mathbf{x}) - \mu^2_\mathbf{z}(\mathbf{x}) - \sigma^2_\mathbf{z}(\mathbf{x}))$$

Variational Autoencoder

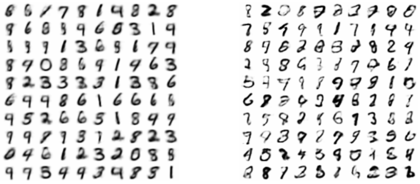
Objective: $\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}, \mathbf{z}) = MSE(\mathbf{x}, \hat{\mathbf{x}}) + \mathcal{D}_{KL}[q(\mathbf{z}|\mathbf{x})|p(\mathbf{z})]$

- Mean-squared-error: → How accurate input can be reconstructed
- KL-divergence: → How close the latent variables match (unit Gaussian)
- Allows walk in latent space



VAE trained on MNIST using 2D Gaussian prior

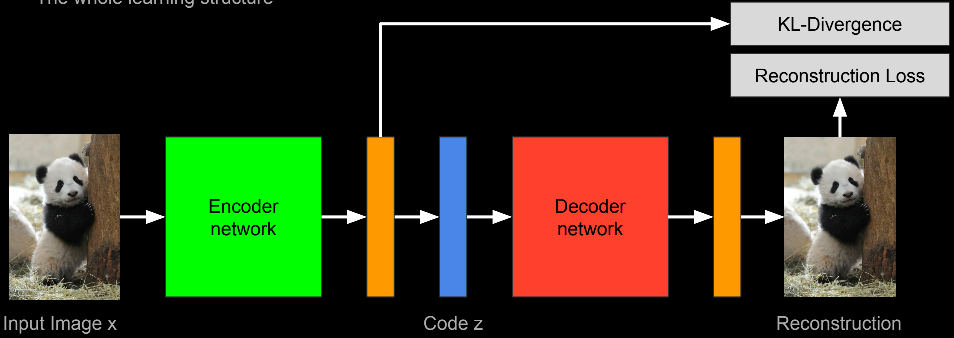
Improved quality for increased size of latent space



(a) 2-D latent space (d) 20-D latent space

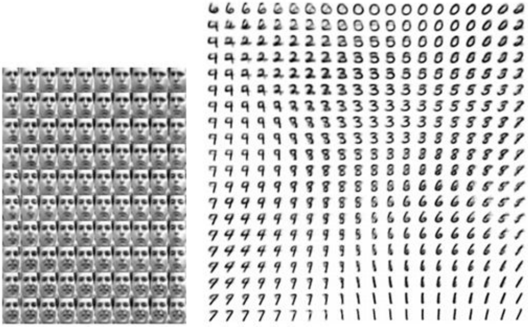
Variational Auto-encoder (VAE)

The whole learning structure



Variational Auto-encoder (VAE)

Results



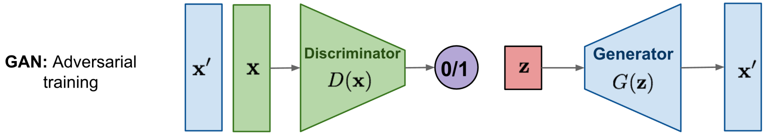
(a) Learned Frey Face manifold (b) Learned MNIST manifold

Kingma et al. 2014

VAE do ponto de vista de NN e de probabilidade

https://jaan.io/what-is-variational-autoencoder-vae-tutorial/

GAN – Generative Adversarial Networks



https://www.rootstrap.com/blog/how-to-generate-images-with-ai/

$\mathbf{x}$: entrada real

$\mathbf{x}'$: entrada fake / imagem gerada por G

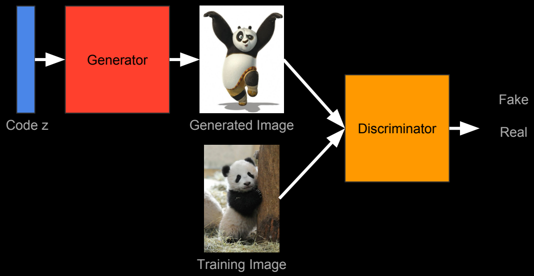
$\mathbf{z}$: vetor latente

Generative Adversarial Network (GAN)

Intuitions:

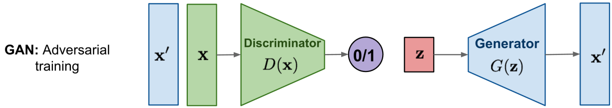
- Generator tries the best to cheat the discriminator by generating more realistic images

- Discriminator tries the best to distinguish whether the image is generated by computers or not



Função de perda adversarial:

$$\min_G \max_D \mathcal{L}(D, G) = E_{\mathbf{x}}\big[\log D(\mathbf{x})\big] + E_{\mathbf{z}}\big[\log(1 - D(\mathbf{z}))\big]$$



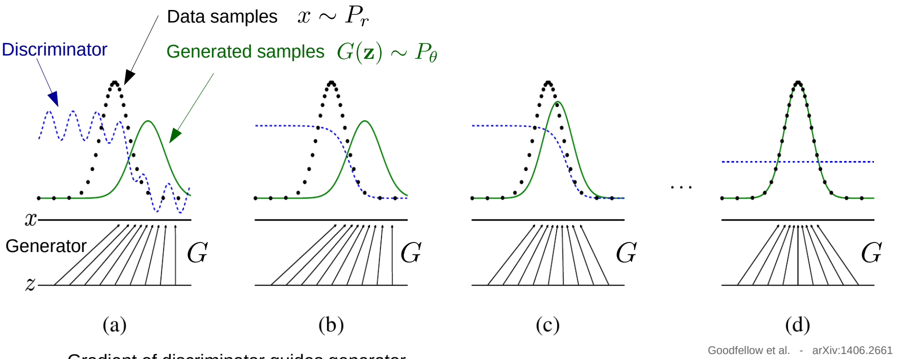
- *data space* $\mathbf{x}$, com distribuição de probabilidade p_{data}
- espaço de entradas ruidosas $\mathbf{z}$, com distribuição $p_z(\mathbf{z})$
- $G(\mathbf{z}; \theta_g)$: mapeamento do espaço $\mathbf{z}$ para espaço $\mathbf{x}$
- $D(\mathbf{x}; \theta_d)$: mapeamento do espaço $\mathbf{x}$ para $[0, 1]$, probabilidade de uma instância $\mathbf{x}$ ser um sample de p_g

GAN – treinamento

Tomar mini-batch de imagens fakes e reais
Fazer update dos pesos de G
Fazer update dos pesos de D

Problema Min-max: encontrar um ponto de sela em vez de um ótimo global, não é muito estável

Optimal Evolution of GAN Training

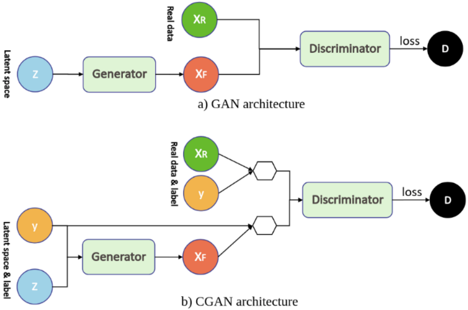


Gradient of discriminator guides generator
→ G generates samples which are more likely identified as data

Epochs

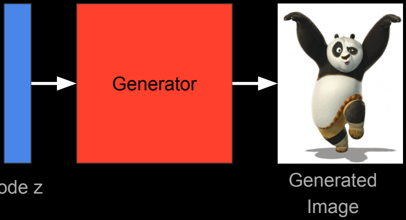
c-GAN

$$\min_G \max_D \mathcal{L}(D, G) = E_x\left[\log D(x|y)\right] + E_z\left[\log(1 - D(z|y))\right]$$



https://mc.ai/a-tutorial-on-conditional-generative-adversarial-nets-keras-implementation/

Generative Adversarial Network (GAN)



GANs for face and bedroom

Credit: Denton

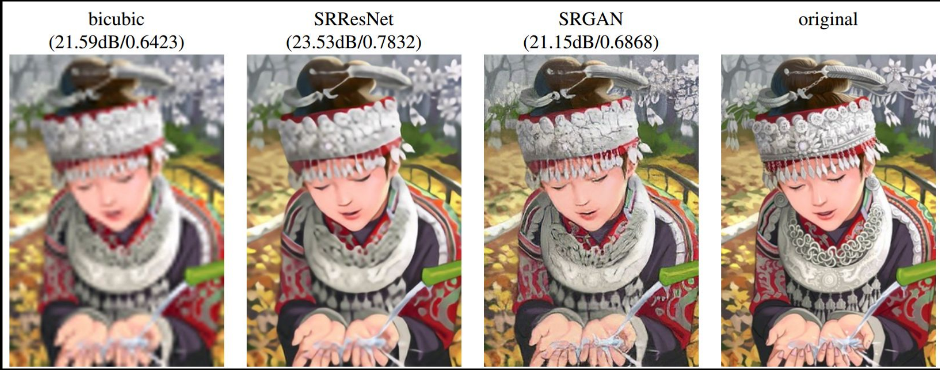
GANs for Japanese Anime

Credit: Radford

GANs for Videos

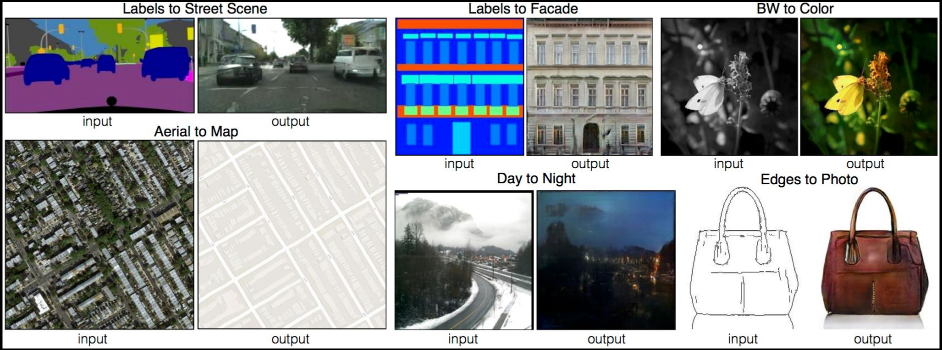
Credit: Vondrick

GANs for Image Upsampling



Credit: Ledig

Conditional GAN



Credit: Zhu et al.

Generative Adversarial Network (GAN)

Extensions:

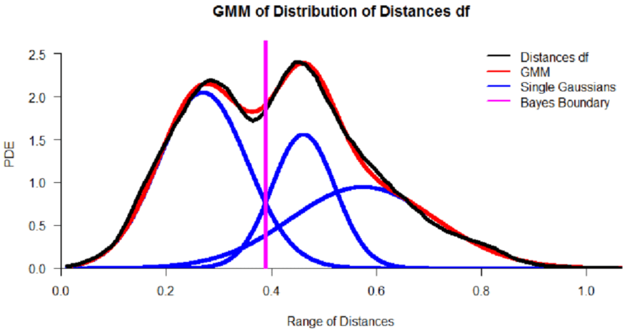
- DCGANs: some hacks that work well
- LAPGANs: coarse-to-fine conditional generation through Laplacian pyramids
- f-GANs: more general GANs with different loss other than cross-entropy
- infoGANs: additional objective that maximize mutual-information between the latent and the sample
- EBGANs: Discriminative as energy functions
- GVMs: using GANs as an energy term for interactive image manipulation
- Conditional GANs: not random z, instead z is some data from other domain
- ...

Generative Adversarial Network (GAN)

Hacks:

- How to train a GAN?
- 17 hacks that make the training work.
- https://github.com/soumith/ganhacks

GMM (Gaussian mixture models)



Aproximar fdp por um histograma?

$$P(\mathbf{x} \,|\, y_j)$$

Uma forma de estimar a densidade de probabilidade seria fazer uma aproximação por meio de um histograma

Quais são os problemas de se usar histograma para tal cálculo aproximado?

Estimação de fdp via janelas de Parzen (ou kernels)

Considere a função auxiliar definida por, para todo $\mathbf{u} = (u_1, u_2, \ldots, u_d) \in \mathbb{R}^d$,

$$\varphi(\mathbf{u}) = \begin{cases} 1, & \text{se } |u_j| \leqslant 1/2, \quad j = 1, 2, \ldots, d \\ 0, & \text{caso contrário.} \end{cases}$$

O que essa função define ?

Estimação de fdp via janelas de Parzen (ou kernels)

Considere a função auxiliar definida por, para todo $\mathbf{u} = (u_1, u_2, \ldots, u_d) \in \mathbb{R}^d$,

$$\varphi(\mathbf{u}) = \begin{cases} 1, & \text{se } |u_j| \leqslant 1/2, \quad j = 1, 2, \ldots, d \\ 0, & \text{caso contrário.} \end{cases}$$

O que essa função define ?

Hipercubo unitário centrado na origem

Estimação de fdp via janelas de Parzen (ou kernels)

Considere a função auxiliar definida por, para todo $\mathbf{u} = (u_1, u_2, \ldots, u_d) \in \mathbb{R}^d$,

$$\varphi(\mathbf{u}) = \begin{cases} 1, & \text{se } |u_j| \leqslant 1/2, \quad j = 1, 2, \ldots, d \\ 0, & \text{caso contrário.} \end{cases}$$

O que essa função define ?

Hipercubo unitário centrado na origem

Fixado $\mathbf{x}$, o que é então $\varphi(\mathbf{u} - \mathbf{x})$?

Estimação de fdp via janelas de Parzen (ou kernels)

Considere a função auxiliar definida por, para todo $\mathbf{u} = (u_1, u_2, \ldots, u_d) \in \mathbb{R}^d$,

$$\varphi(\mathbf{u}) = \begin{cases} 1, & \text{se } |u_j| \leqslant 1/2, \quad j = 1, 2, \ldots, d \\ 0, & \text{caso contrário.} \end{cases}$$

O que essa função define ?

Hipercubo unitário centrado na origem

Fixado $\mathbf{x}$, o que é então $\varphi(\mathbf{u} - \mathbf{x})$?

Hipercubo unitário centrado em $\mathbf{x}$

Estimação de fdp via janelas de Parzen (ou kernels)

Fixe $\mathbf{x} \in \mathbb{R}^d$ e $h > 0$.

O que é então $\varphi\left(\frac{\mathbf{y} - \mathbf{x}}{h}\right)$, $\mathbf{y} \in \mathbb{R}^d$??

Estimação de fdp via janelas de Parzen (ou kernels)

Fixe $\mathbf{x} \in \mathbb{R}^d$ e $h > 0$.

O que é então $\varphi\left(\frac{\mathbf{y} - \mathbf{x}}{h}\right)$, $\mathbf{y} \in \mathbb{R}^d$??

Define um hipercubo de lados com tamanho h, com centro em $\mathbf{x}$

Estimação de fdp via janelas de Parzen (ou kernels)

Fixe $\mathbf{x} \in \mathbb{R}^d$ e $h > 0$.

O que é então $\varphi\left(\frac{\mathbf{y} - \mathbf{x}}{h}\right)$, $\mathbf{y} \in \mathbb{R}^d$??

Define um hipercubo de lados com tamanho h, com centro em $\mathbf{x}$

Densidade em $\mathbf{x}$: ??

Estimação de fdp via janelas de Parzen (ou kernels)

Fixe $\mathbf{x} \in \mathbb{R}^d$ e $h > 0$.

O que é então $\varphi\left(\frac{\mathbf{y} - \mathbf{x}}{h}\right)$, $\mathbf{y} \in \mathbb{R}^d$??

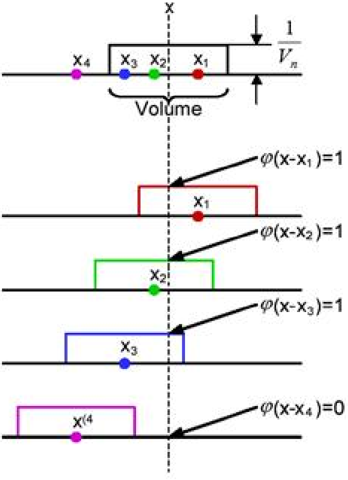
Define um hipercubo de lados com tamanho h, com centro em $\mathbf{x}$

Densidade em $\mathbf{x}$:
podemos usar um *bin* centrado em $\mathbf{x}$, em vez de *bins* fixos

Estimação de fdp via janelas de Parzen (ou kernels)

Amostras: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$

$\hat{P}(\mathbf{x}) = ??$



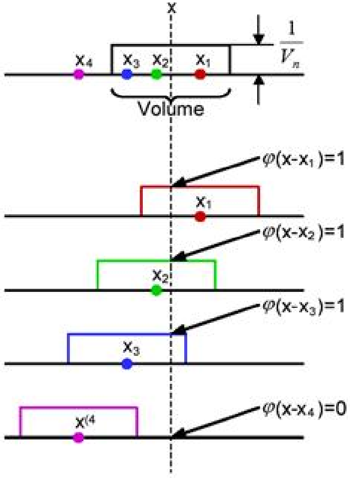
https://www.byclb.com/TR/Tutorials/neural_networks/ch11_1.htm

Estimação de fdp via janelas de Parzen (ou kernels)

Amostras: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$

$$\hat{P}(\mathbf{x}) = \sum_{n=1}^{N} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right)$$

$\varphi\left(\frac{\mathbf{x}-\mathbf{x}_n}{h}\right)$ funciona como um contador de amostras que estão no *bin* de largura h centrado em $\mathbf{x}$



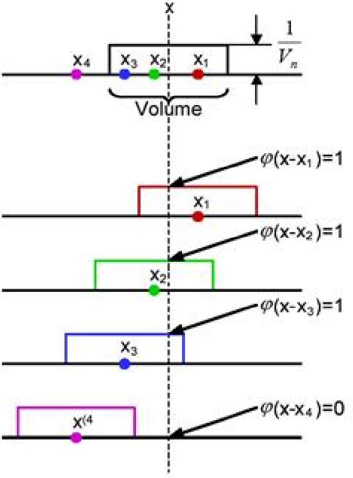
https://www.byclb.com/TR/Tutorials/neural_networks/ch11_1.htm

Estimação de fdp via janelas de Parzen (ou kernels)

Amostras: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$

$$\hat{P}(\mathbf{x}) = \frac{1}{N} \sum_{n=1}^{N} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right)$$

$\varphi\left(\frac{\mathbf{x}-\mathbf{x}_n}{h}\right)$ funciona como um contador de amostras que estão no *bin* de largura h centrado em $\mathbf{x}$



https://www.byclb.com/TR/Tutorials/neural_networks/ch11_1.htm

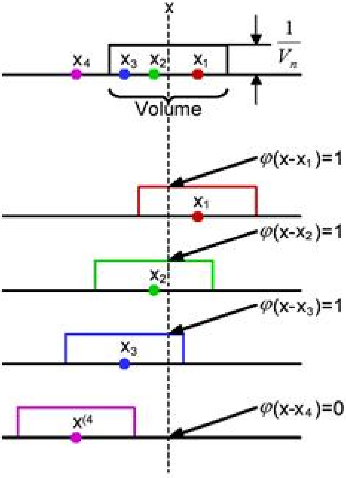
Estimação de fdp via janelas de Parzen (ou kernels)

Amostras: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$

$$\hat{P}(\mathbf{x}) = \frac{1}{V}\Big[\frac{1}{N}\sum_{n=1}^{N} \varphi\Big(\frac{\mathbf{x}-\mathbf{x}_n}{h}\Big)\Big]$$

$\varphi\big(\frac{\mathbf{x}-\mathbf{x}_n}{h}\big)$ funciona como um contador de amostras que estão no *bin* de largura h centrado em $\mathbf{x}$

V: volume do hipercubo



https://www.byclb.com/TR/Tutorials/neural_networks/ch11_1.htm

Estimação de fdp via janelas de Parzen (ou kernels)

Amostras: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$

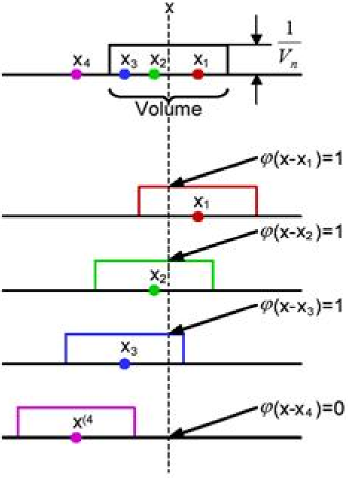
$$\hat{P}(\mathbf{x}) = \frac{1}{V}\left[\frac{1}{N}\sum_{n=1}^{N}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_n}{h}\right)\right]$$

$$= \frac{1}{V}\frac{k}{N}$$

$\varphi\left(\frac{\mathbf{x}-\mathbf{x}_n}{h}\right)$ funciona como um contador de amostras que estão no *bin* de largura h centrado em $\mathbf{x}$

V: volume do hipercubo

k: número de amostras no *bin* de $\mathbf{x}$



https://www.byclb.com/TR/Tutorials/neural_networks/ch11_1.htm

Parzen mostrou que se φ satisfaz

$$\varphi(\mathbf{x}) \geqslant 0$$

e

$$\int \varphi(\mathbf{u}) d\mathbf{u} = 1$$

então

$$\hat{P}(\mathbf{x}) = \frac{1}{V}\Big[\frac{1}{N}\sum_{n=1}^{N} \varphi\Big(\frac{\mathbf{x} - \mathbf{x}_n}{h}\Big)\Big]$$

é uma fdp (função densidade de probabilidade)

Exemplos de kernel unidimensional

- Retangular

$$\phi(u) = \begin{cases} \frac{1}{2}, & \text{se } |u| < 1, \\ 0, & \text{c.c.} \end{cases}$$

- Triangular

$$\phi(u) = \begin{cases} 1 - |u|, & \text{se } |u| < 1, \\ 0, & \text{c.c.} \end{cases}$$

- Biweight

$$\phi(u) = \begin{cases} \frac{15}{16}(1 - u^2)^2, & \text{se } |u| < 1, \\ 0, & \text{c.c.} \end{cases}$$

- Normal (um dos mais usados)

$$\phi(u) = \frac{1}{\sqrt{2\pi}} e^{\frac{-u^2}{2}}$$

- Bartlett-Epanechnenko

$$\begin{cases} \frac{3}{4}\left(1 - \frac{u^2}{\sqrt{5}}\right), & \text{se } |u| < \sqrt{5}, \\ 0, & \text{c.c.} \end{cases}$$

Voltando ao classificador de Bayes

$$P(y_j \,|\, \mathbf{x}) = \frac{P(y_j)\, P(\mathbf{x} \,|\, y_j)}{P(\mathbf{x})}$$

Voltando ao classificador de Bayes

$$P(y_j \,|\, \mathbf{x}) = \frac{P(y_j)\, P(\mathbf{x} \,|\, y_j)}{P(\mathbf{x})} = \frac{P(y_j, \mathbf{x})}{P(\mathbf{x})}$$

$\hat{P}(\mathbf{x}) = ??$ $\hat{P}(y_j, \mathbf{x}) = ??$

Voltando ao classificador de Bayes

$$P(y_j \mid \mathbf{x}) = \frac{P(y_j)\, P(\mathbf{x} \mid y_j)}{P(\mathbf{x})} = \frac{P(y_j, \mathbf{x})}{P(\mathbf{x})}$$

$$\hat{P}(\mathbf{x}) = \frac{k}{NV} \qquad\qquad \hat{P}(y_j, \mathbf{x}) = \frac{k_j}{NV}$$

k: número de amostras no *bin* de $\mathbf{x}$

k_j: número de amostras da classe j no *bin* de $\mathbf{x}$

Voltando ao classificador de Bayes

$$P(y_j \mid \mathbf{x}) = \frac{P(y_j)\, P(\mathbf{x} \mid y_j)}{P(\mathbf{x})} = \frac{P(y_j, \mathbf{x})}{P(\mathbf{x})}$$

$$\hat{P}(\mathbf{x}) = \frac{k}{NV} \qquad \hat{P}(y_j, \mathbf{x}) = \frac{k_j}{NV}$$

k: número de amostras no *bin* de $\mathbf{x}$

k_j: número de amostras da classe j no *bin* de $\mathbf{x}$

$$\hat{P}(y_j|\mathbf{x}) = ??$$

Voltando ao classificador de Bayes

$$P(y_j \mid \mathbf{x}) = \frac{P(y_j)\, P(\mathbf{x} \mid y_j)}{P(\mathbf{x})} = \frac{P(y_j, \mathbf{x})}{P(\mathbf{x})}$$

$$\hat{P}(\mathbf{x}) = \frac{k}{NV} \qquad\qquad \hat{P}(y_j, \mathbf{x}) = \frac{k_j}{NV}$$

k: número de amostras no *bin* de $\mathbf{x}$

k_j: número de amostras da classe j no *bin* de $\mathbf{x}$

$$\hat{P}(y_j|\mathbf{x}) = \frac{\hat{P}(y_j, \mathbf{x})}{\hat{P}(\mathbf{x})} = \frac{k_j}{k}$$

A expressão

$$\hat{P}(y_j|\mathbf{x}) = \frac{1}{N}\frac{\hat{P}(y_j, \mathbf{x})}{\hat{P}(\mathbf{x})} = \frac{k_j}{k}$$

leva-nos ao k-NN !

Algoritmo k-NN (*nearest neighbors*):

- seja $\mathbf{x}$ a nova amostra a ser classificada
- determina-se os k pontos em D mais próximos de $\mathbf{x}$
- atribui-se a $\mathbf{x}$ o rótulo mais frequente dentre esses k pontos

No caso de k-NN, fixa-se k em vez do volume V !

Vizinhos mais próximos e estimação de fdp por kernel

Vantagens:

Desvantagens:

Vizinhos mais próximos e estimação de fdp por kernel

Vantagens:

- simples
- ??

Desvantagens:

Vizinhos mais próximos e estimação de fdp por kernel

Vantagens:

- simples
- ??

Desvantagens:

- complexidade computacional, memória
- depende da métrica de similaridade / kernel
- em espaços de dimensão alta pode não haver vizinhos ...

Vizinhos mais próximos e estimação de fdp por kernel

Vantagens:

- simples
- ??

Desvantagens:

- complexidade computacional, memória
- depende da métrica de similaridade / kernel
- em espaços de dimensão alta pode não haver vizinhos ...

As the amount of data approaches infinity, the two-class k-NN algorithm is guaranteed to yield an error rate no worse than twice the Bayes error rate