初识对抗生成网络

Generative Adversarial Network

主讲人：董豪　讲义：董豪

内容提要

- 引入：生成式模型 Generative Models
- 朴素GAN Vanilla GAN
- 对抗损失函数 VS 均方差 Adversarial Loss vs. MSE
- GAN面临的挑战 Challenges of GAN

生成式模型

- 如何用计算机生成数据（比如图像）

Computer Graphics →

Ball(color=yellow, position=(50, 100),…)
Ball(color=red, position=(30, 75),…)
Ball(color=blue, position=(30, 125),…)
…
…

← Computer Vision

"描述"
Description

"实测数据"
Observation

生成式模型

- 统计生成模型是**数据驱动**的方法

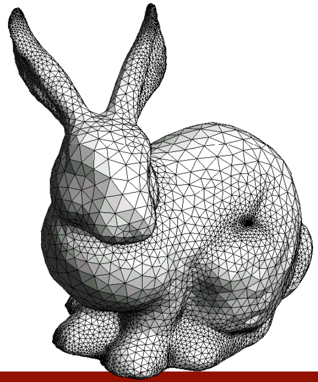
Computer Graphics

Statistical Generative Models

先验知识
Prior Knowledge

Data

Material, Physical Modeling, Lighting ....



生成式模型

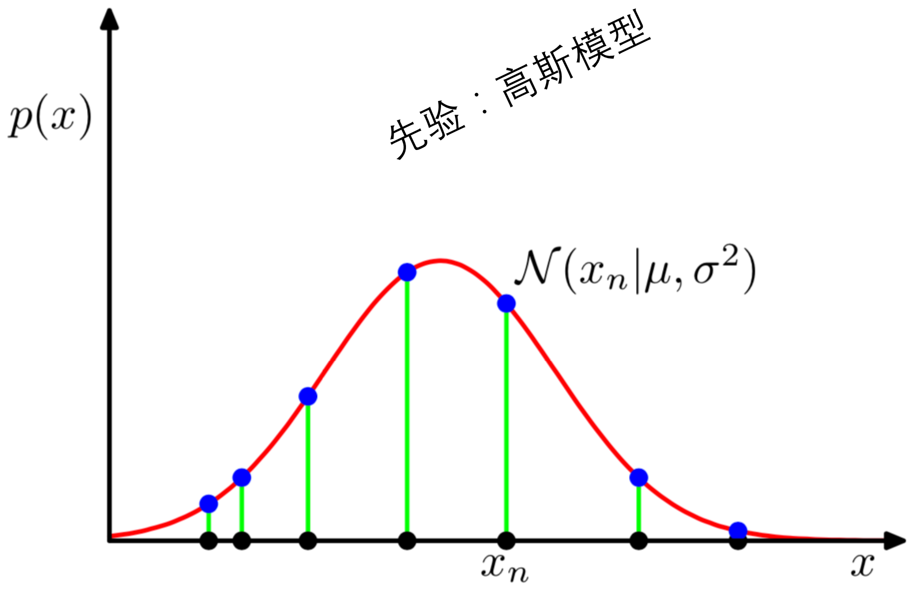
- 计算机图形学
  - 完全基于先验知识
  - 难以扩展和泛化
  - 开发耗时

- 机器学习/深度学习
  - 减少先验知识的需求
  - 从数据中学习

- 统计/深度生成模型
  - 仍需要一些先验知识…
  - 损失函数、学习方法、架构、先验分布（例如，高斯）

生成式模型

- 统计/深度生成模型



- 给定数据样本 学习概率分布 $p(x)$
- 因此它是生成性的，因为可以从$p(x)$中采样新的数据样本

$$x_{new} \sim p_x$$

生成式模型

- 统计/深度生成模型
  - 给定数据样本
  - 学习概率分布 $p(x)$

  - 该算法是生成性的，因为可以从$p(x)$中采样新的数据样本
  - $x_{new} \sim p(x)$

数据分布可能是高维的，比如图像

生成式模型

- **数据表示**
- 我们希望学习一个关于$\mathbf{x}$的概率分布$p(\mathrm{x})$
  1. 生成（采样）：$\mathbf{x}_{new} \sim p(\mathrm{x})$
  2. 密度估计：如果$\mathbf{x}$看起来像猫，那么$p(\mathrm{x})$会很高
  3. 无监督表示学习：

  从数据分布中发现潜在结构（如耳朵、鼻子、眼睛等）

p_{data}

$$\mathbf{x}^j \sim p_{data}$$
$$j = 1,2, \ldots |\mathcal{D}|$$

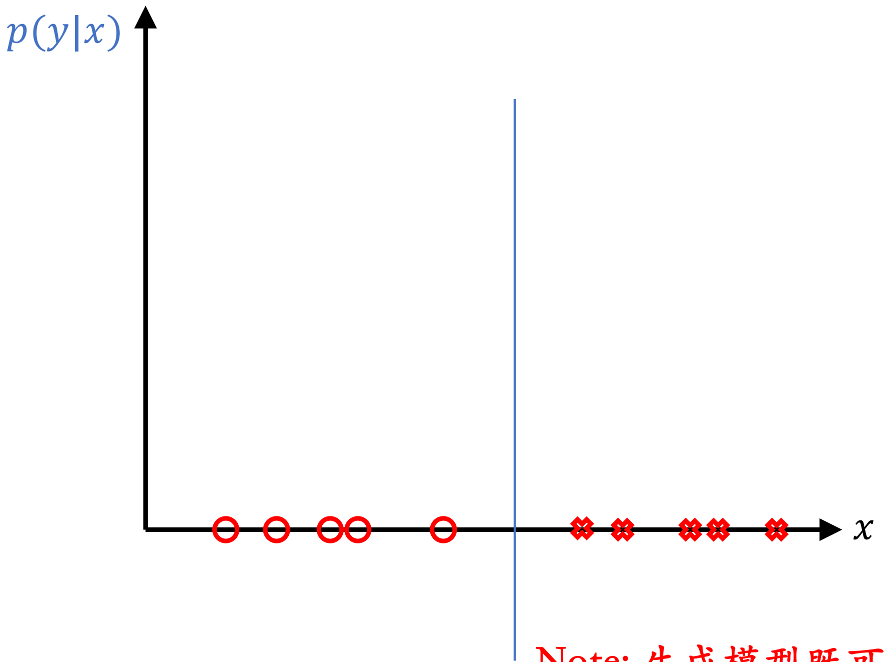
- 数据集 $\mathcal{D}$
- 数据分布 p_{data}
- 模型参数 $\theta \in \mathcal{M}$

生成式模型

判别式 vs. 生成式

判别式模型： 分类数据

寻找**决策边界** $P(Y \mid X)$

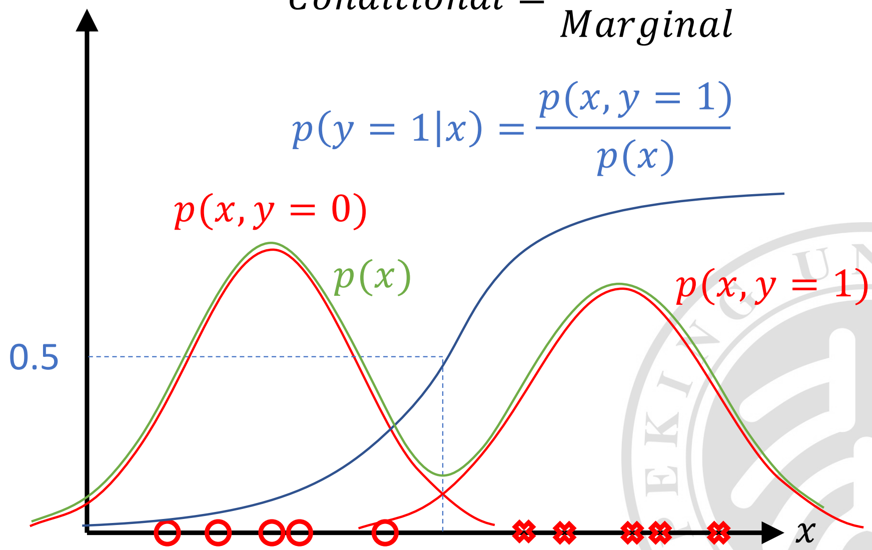


生成式模型： 生成数据

寻找**联合分布** $P(Y,X)$

$$Conditional = \frac{Joint}{Marginal}$$

$$p(y=1|x) = \frac{p(x,y=1)}{p(x)}$$

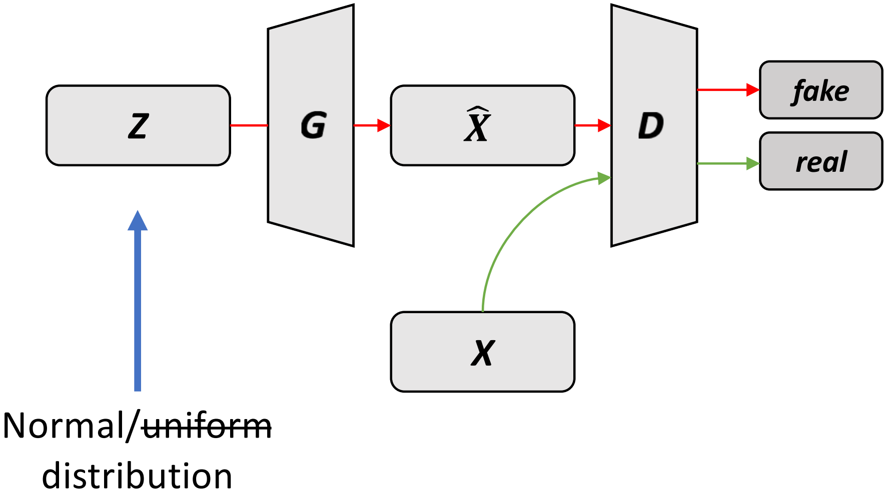


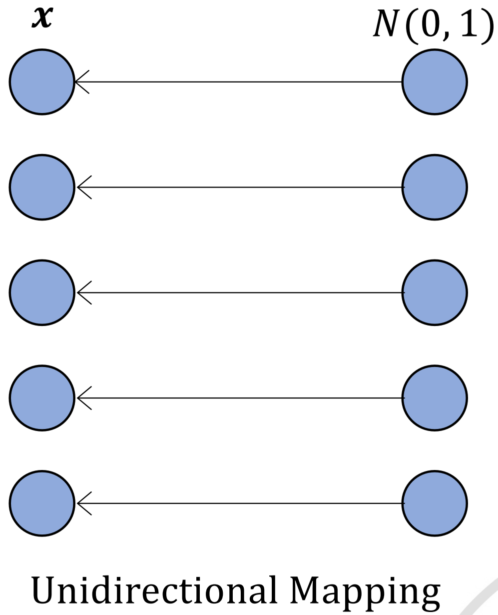
Note: 生成模型既可以执行生成任务，也可以执行判别任务

Vanilla GAN

朴素GAN

- 朴素GAN Vanilla GAN





GAN: map a distribution to another distribution

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_z}[\log(1-D(G(\boldsymbol{z}))]$$

$$\mathcal{L}_D = -\boxed{\mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D(\boldsymbol{x})]} - \boxed{\mathbb{E}_{\boldsymbol{z}\sim p_z}[\log(1-D(G(\boldsymbol{z}))]}$$

$$\mathcal{L}_G = -\boxed{\mathbb{E}_{\boldsymbol{z}\sim p_z}[\log D(G(\boldsymbol{z}))]}$$

朴素GAN

$$G^{*} = \min_{G} \mathbb{E}_{\boldsymbol{z} \sim p_z}[\log(1 - D(G(\boldsymbol{z})))]$$

$$D^{*} = \max_{D} \mathbb{E}_{\boldsymbol{x} \sim p_{data}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_z}[\log(1 - D(G(\boldsymbol{z})))]$$

$$\boxed{\min_{G} \max_{D} V(D, G) = \min_{G} \max_{D} \mathbb{E}_{\boldsymbol{x} \sim p_{data}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_z}\left[\log\left(1 - D\left(G(\boldsymbol{z})\right)\right)\right]}$$

为什么优化这个目标函数可以起作用?

朴素GAN

- Vanilla GAN - Theoretical Results
- 我们认为这个**最小-最大博弈（ min-max game ）**有一个全局最优解，即

$$p_g = p_{data}$$

  - 首先，我们认为当生成器G固定时，最优判别器D*满足:

$$D^* = \frac{p_{data}}{p_g + p_{data}} = 0.5 \quad (p_g = p_{data})$$

后面会有推导

  - 接着，当判别器D*固定时,

$$V(G,D) = -log4 + JS(p_{data}||p_g) = -log4 = -2log2 \quad (p_g = p_{data})$$

  - 因此，当判别器D和生成器G都达到最优时,

$$p_{data} = p_g$$

朴素GAN

- 深入理解目标函数
  - $\min_G \max_D V(D,G) = \min_G \max_D \mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_z}\left[\log(1 - D\big(G(\boldsymbol{z})\big))\right]$
  - When $z \sim p(z)$, let p_g denote distribution of $G(z)$
  - $\boxed{\min_G \max_D V(D,G) = \min_G \max_D \mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1 - D(\boldsymbol{x}))]}$

朴素GAN

- 首先假设G固定

$$\min_G \max_D V(D,G) = \min_G \max_D \mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1 - D(\boldsymbol{x})]$$

$$\begin{aligned} V(G,D) &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x}))dx + \int_z p_{\boldsymbol{z}}(\boldsymbol{z}) \log(1 - D(g(\boldsymbol{z})))dz \\ &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x})) + p_g(\boldsymbol{x}) \log(1 - D(\boldsymbol{x}))dx \end{aligned}$$

(logA)’ = $\frac{1}{A}$

$$V(G,D)' = \frac{p_{data}}{D(x)} - \frac{p_g}{1 - D(x)} = 0$$

- The optimum $\boxed{D^* = \frac{p_{data}}{p_g + p_{data}}}$

朴素GAN

Vanilla GAN - Theoretical Results

- Then for D^* fixed

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}}[\log(1 - D_G^*(G(\boldsymbol{z})))] \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1 - D_G^*(\boldsymbol{x}))] \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{P_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] + \mathbb{E}_{\boldsymbol{x}\sim p_g}\left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right]
\end{aligned}
$$

- Recap $JS(P||Q) = \frac{1}{2}KL(P||\frac{P+Q}{2}) + \frac{1}{2}KL(Q||\frac{P+Q}{2})$

- Then $\boxed{V(G,D) = -log4 + 2 * JS(p_{data}||p_g)}$

0 ~ log2

$V(G,D) = -2log2 + 2log2 = 0$

朴素GAN

训练算法

- 循环：采样一批 {z_i} 和 {x_i} -> 更新判别器D -> 更新生成器G

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

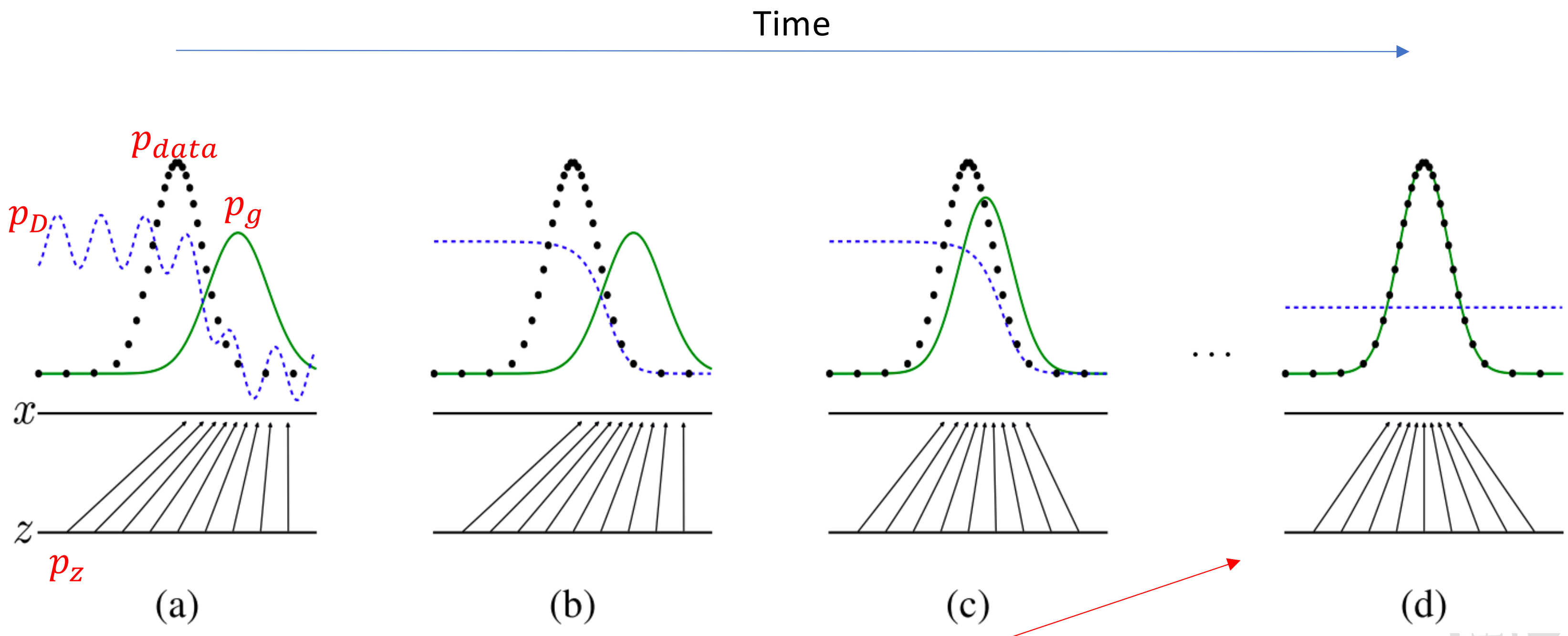
end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

为什么不先将判别器D更新至其最优状态?

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

Vanilla GAN - Theoretical Results



判别器无法区分这两个分布，即 D(x) = 0.5

朴素GAN

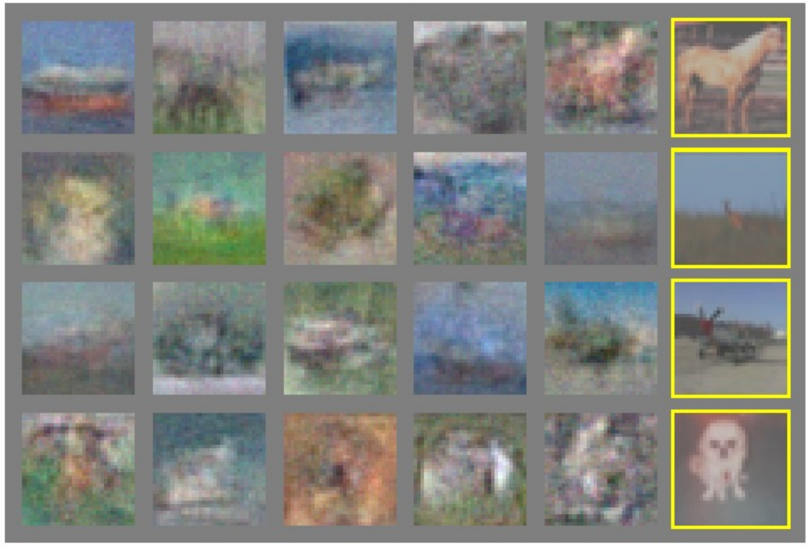
- Vanilla GAN的实验
- MNIST and TFD
  - 随机采样 Random sample

  - 潜码插值 Interpolation on Latent Codes

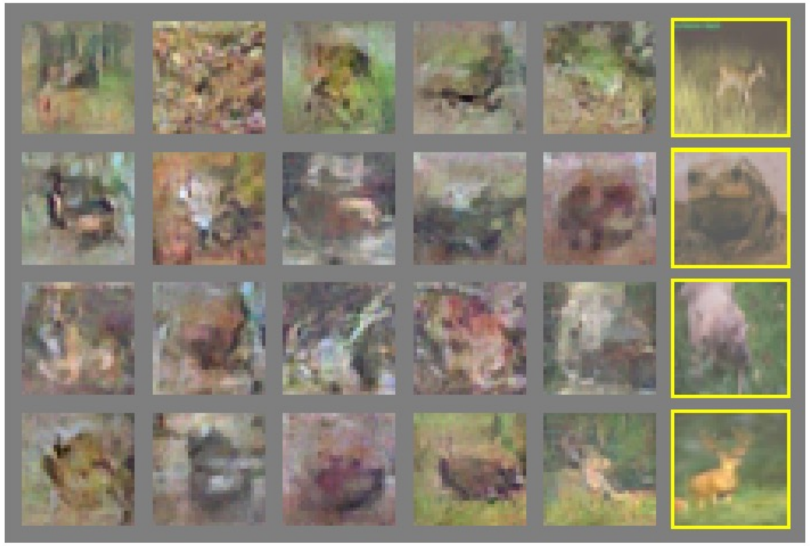
朴素GAN

- Vanilla GAN的实验
- CIFAR10

Fully connected model



Convolutional model



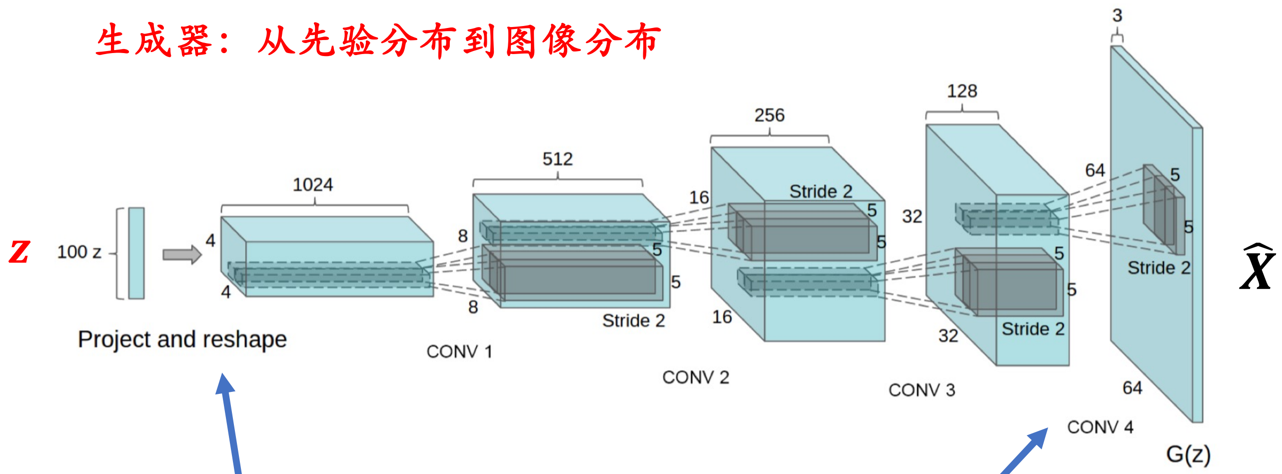
如何获得更好的表现?

DCGAN: Deep Convolutional GAN

DCGAN

- 发挥卷积神经网络的威力

生成器：从先验分布到图像分布



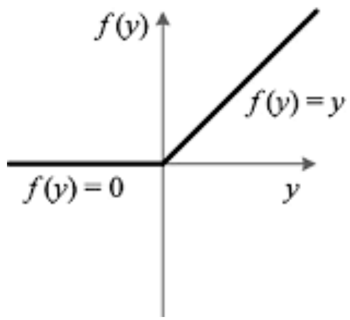
Normal distribution as the latent distribution
z = 100 values

Data distribution
x = 64x64x3 values

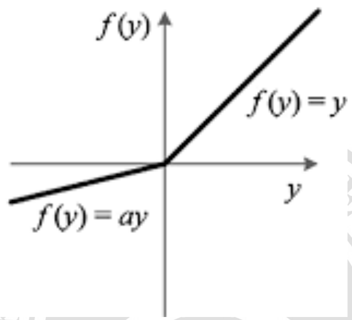
DCGAN

- DCGAN技巧:

1. 对所有层使用批量归一化，除了G的最后一层和D的输入层，衰减系数为0.9（默认值为0.99）。
2. 使用Adam优化器，一阶动量（beta1）为0.5（默认值为0.9）。
3. 带0.2α的Leaky ReLU（默认值为ReLU）。
4. 使用跨步卷积（默认值为最大池化）。
5. 学习速率为0.0002（默认值为0.0001）。



ReLU



Leaky ReLU

DCGAN

- Latent Representation

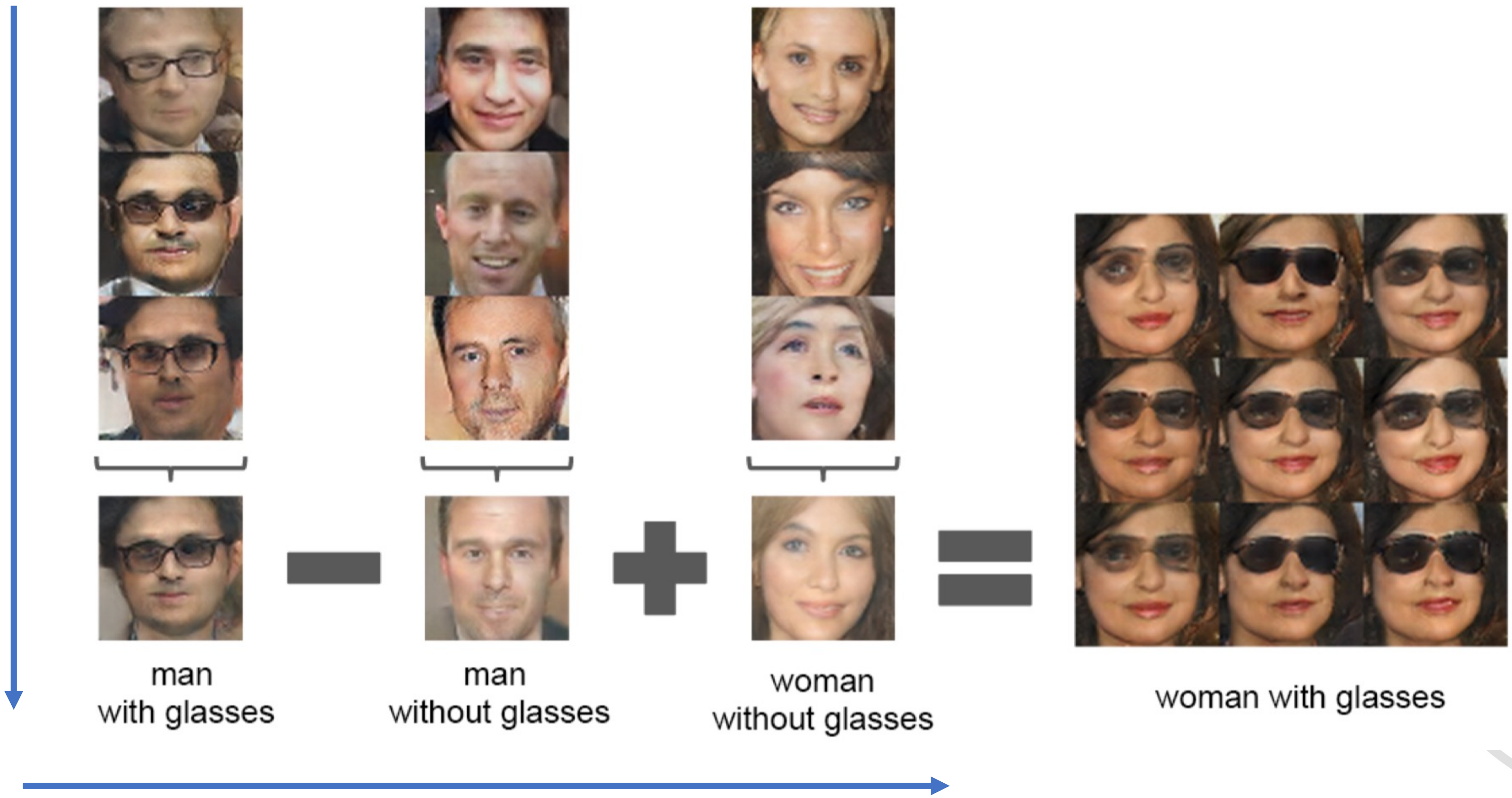
Interpolation: $(\mathbf{1}-\alpha)\boldsymbol{z}_{\mathbf{1}}+\alpha\boldsymbol{z}_{\mathbf{2}}$

Noise vector z_1 → **Noise vector z_2**

$\alpha = 0$ $\alpha = 0.5$ $\alpha = 1$

DCGAN

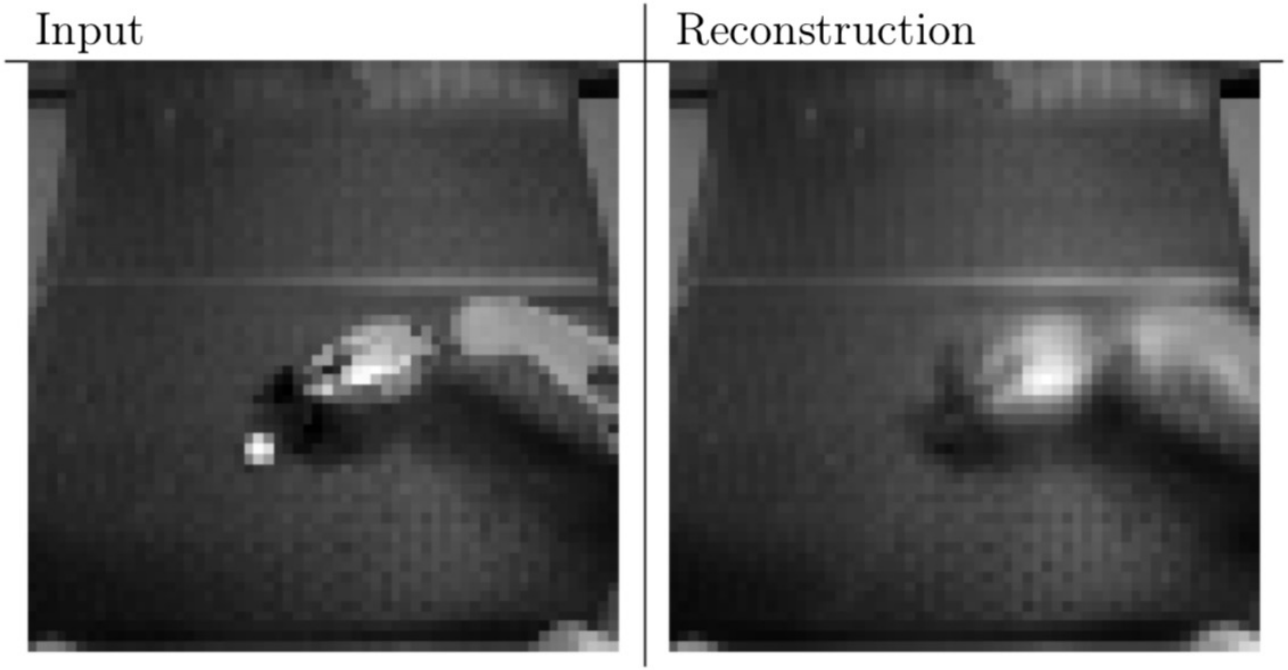
- Latent Representation



Adversarial Loss vs. MSE

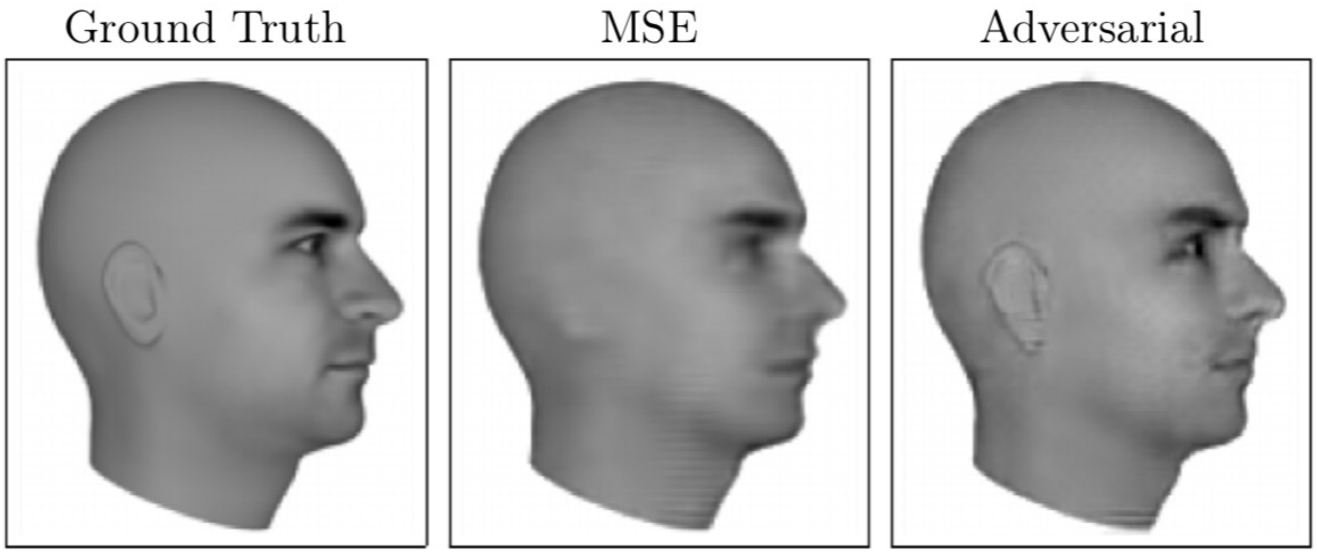
Adversarial Loss

- MSE的缺陷



使用均方误差训练的自编码器在机器人任务中未能重构乒乓球。

Adversarial Loss



（中）仅使用均方误差训练的预测生成网络生成的图像。由于耳朵与相邻肌肤之间的亮度差异不是很明显，所以它们不足以成为模型学习表达它们的足够显著特征。

（右）通过使用均方误差和对抗损失的组合训练模型生成的图像。使用这个学习到的代价函数，耳朵变得显著，因为它们遵循可预测的模式。

Adversarial Loss

- 一些GAN模型生成的效果惊艳的样本，与其他生成模型进行比较：

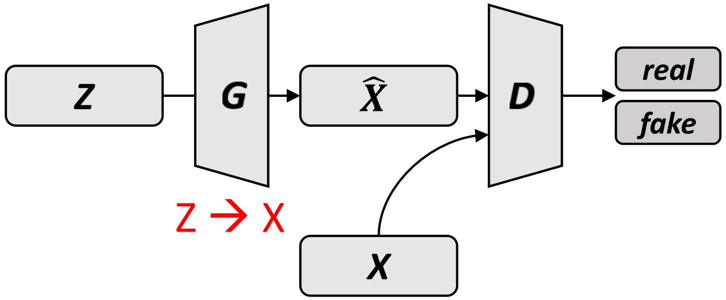
Style-GAN 2019

DFC-VAE

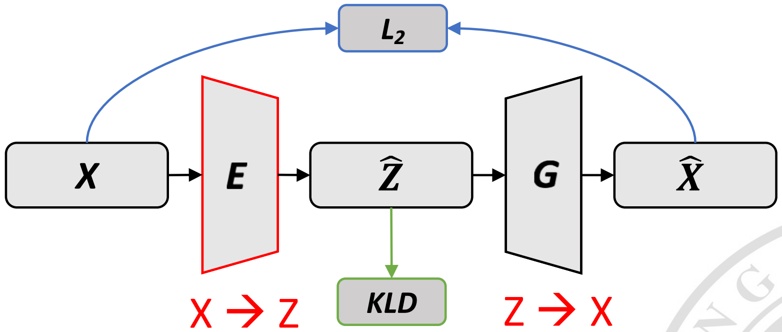
Challenges of GAN

Vanilla GAN vs Variational Autoencoder

Vanilla GAN

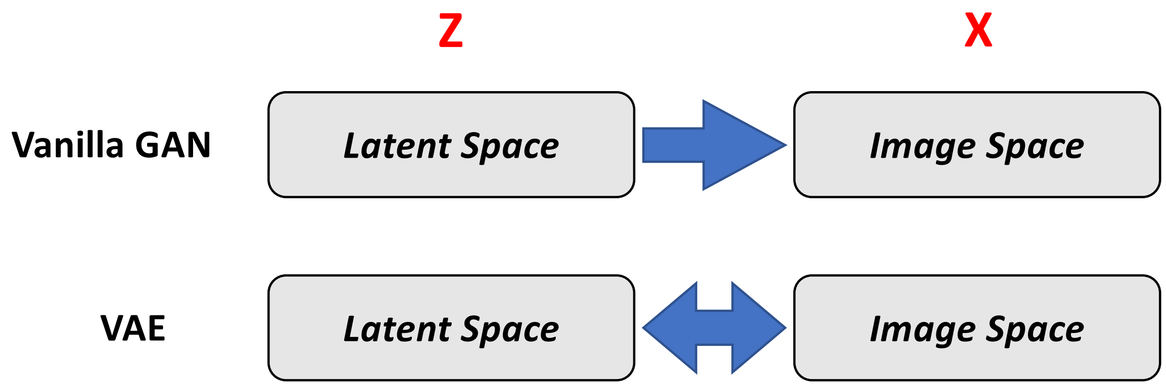


VAE variational autoencoder



VAE has an Encoder that can map x to z

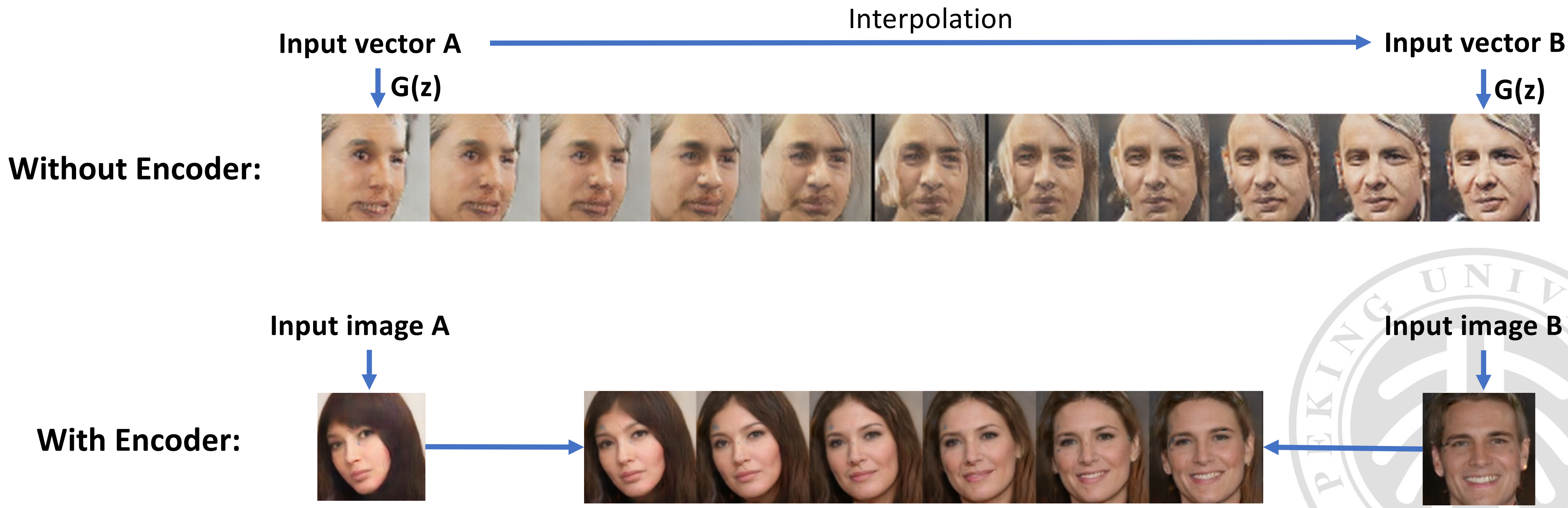
Vanilla GAN vs Variational Autoencoder



- VAE = **G**enerator + **E**ncoder
- Vanilla GAN = **G**enerator + **D**iscriminator
- Better GAN = **G**enerator + **D**iscriminator + **Encoder**

Vanilla GAN vs Variational Autoencoder

- Encoder allows GAN to receive images == More applications



作业

- 第四课作业
  - 用DCGAN在 celebA 数据集上实现人脸生成实验
- 要求：
  1. 阅读DCGAN原论文
  2. 根据论文复现实验，下载数据集，使用PyTorch搭建模型和训练
  3. 调整网络结构、损失函数、训练流程，观察对训练效果的影响
  4. 总结实验报告

谢谢