

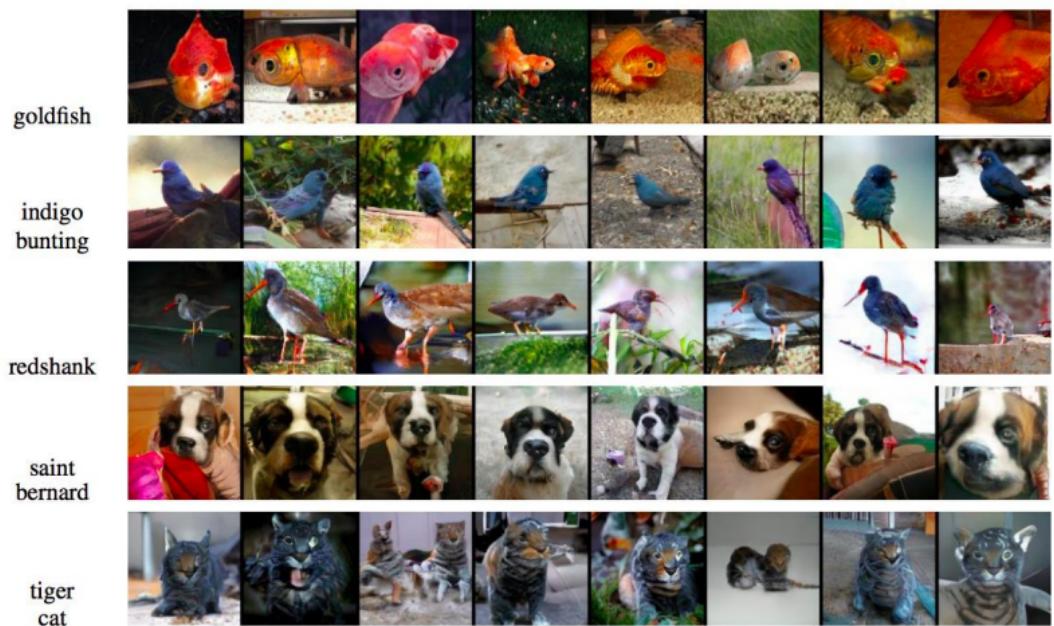
Lecture9 Generative Adversarial Networks

GAN

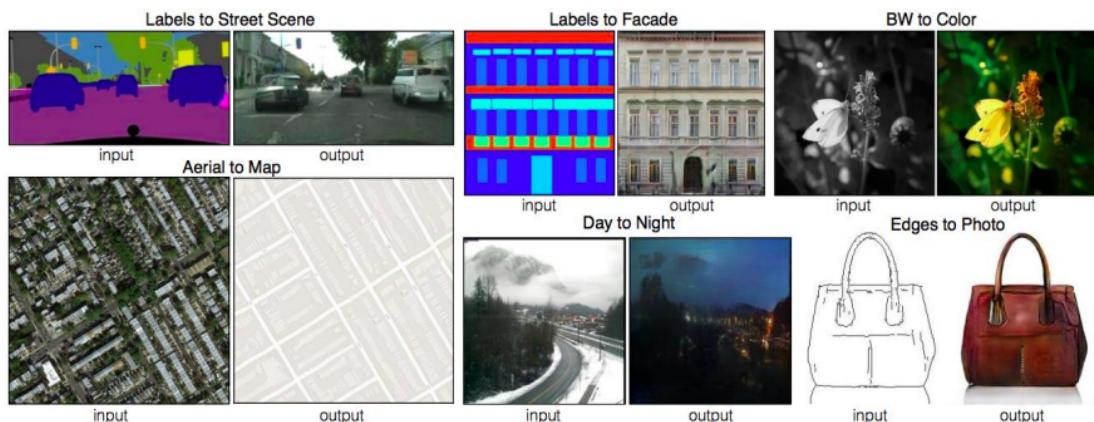
1. 介绍

生成性任务 Generative task

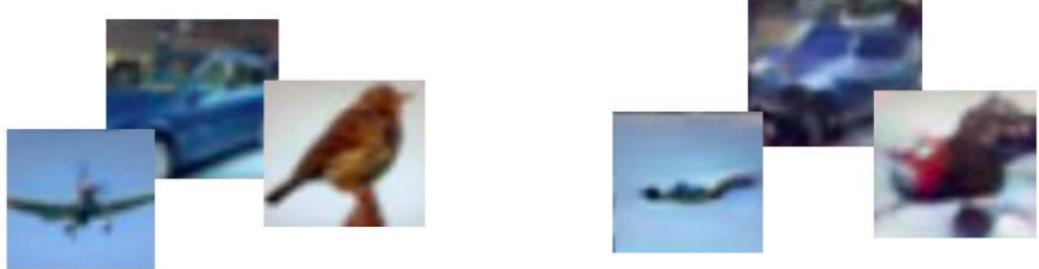
- 生成（从零开始）：学习从训练集表示的分布中采样
 - 非监督学习任务
- 有条件生成



- 图片到图片翻译



学习采样



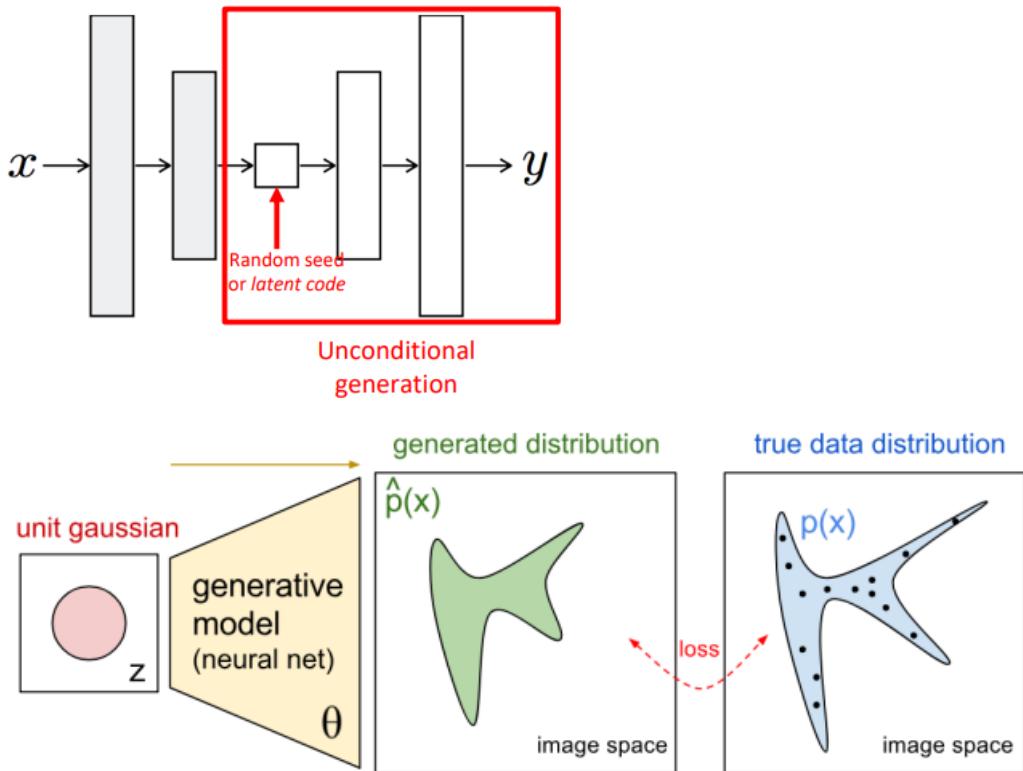
Training data $x \sim p_{\text{data}}$

Generated samples $x \sim p_{\text{model}}$

- 我们想要学习 p_{model} 使其匹配 p_{data}

为生成任务设计一个网络 -- 从 VAE 到 GAN

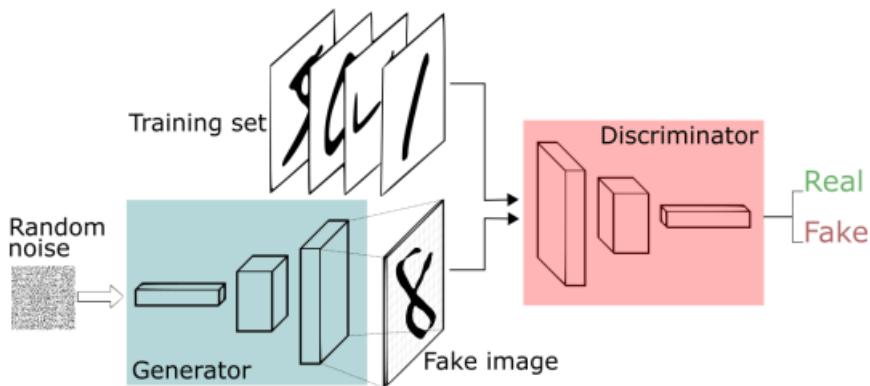
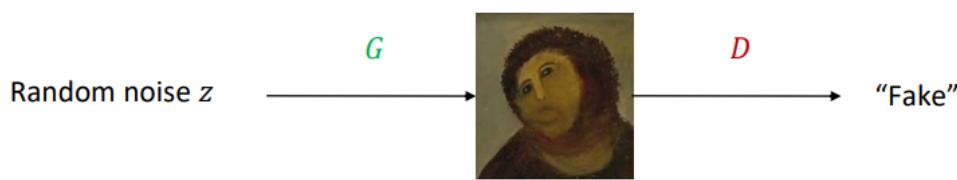
- 我们看到，VAE 可以通过训练编码器和解码器（生成器）来学习生成数据
- 我们能只学习生成器（decoder）吗？



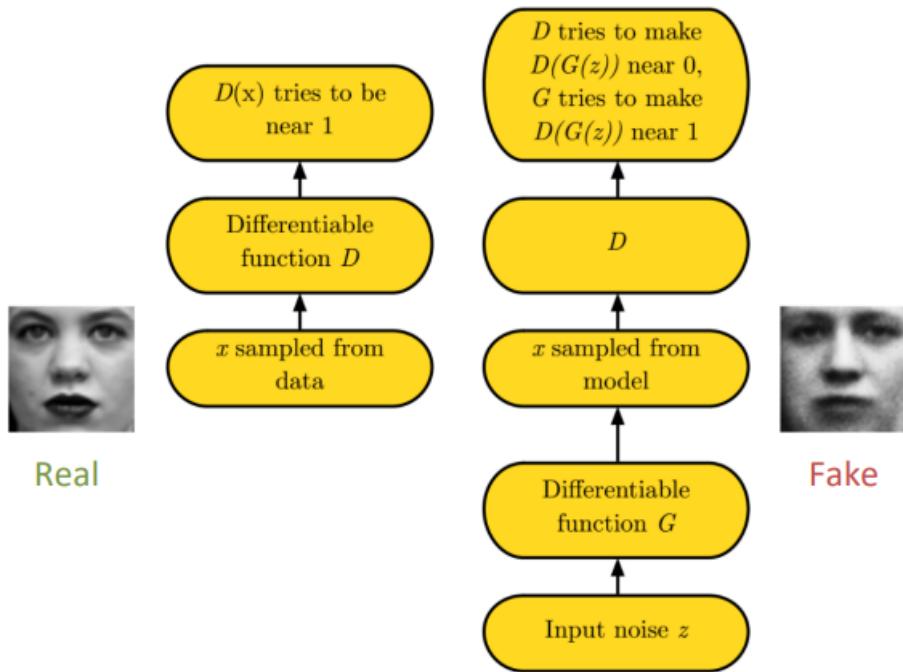
- 在没有编码器的情况下，我们如何生成器的质量呢？
- 换句话说，什么是损失函数？
 - GAN 提出学习损失函数
 - 训练过程是两个网络之间的游戏

2. 生成对抗网络 GAN

组成



- 训练两个目标相反的网络
 - **生成器 Generator:** 学习生成样本
 - $G(z)$ 以随机噪声 z 为输入，输出一幅（伪）图像
 - **鉴别器 Discriminator:** 学会区分生成的样本和真实的样本
 - $D(x)$ 接收到一个输入图像 x ，不管是真实的还是虚假的，并估计它是真实的概率
 - 鉴别器可以使用真实的数据训练好
 - 是一个很传统的二分类神经网络
- **对抗训练 Adversarial training:** 生成器试图欺骗鉴别器，而鉴别器则试图更好地分辨真伪图像
 - 当鉴别器发现假数据时，发生器调整其参数，直到最后发生器再现真实数据分布，而鉴别器无法发现差异
 - 注意：生成器和鉴别器都需要**可微**
 - 通常两者都实现为（深度）神经网络，所以我们可以使用反向传播
 - 这就像是说，最后的鉴别器无法区分生成的数据分布和真实的数据分布（训练数据来自哪里）



极大极小和零和博弈

- G 和 D 遵循极大极小策略

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

- 鉴别器 D 是一个具有单标量输出的可微函数(例如，MLP)，即 x 来自真实数据分布的概率， D 被训练成最大限度地使来自真实数据分布的样本 (标签应该是1) 和来自 G 的样本 (标签应该是0) 得到正确标签的概率最大化
 - 当鉴别器认为 x 是真实的图像时， $D(x) = 1$
 - 当鉴别器认为 $G(z)$ 是真实的图像时， $D(G(z)) = 1$
 - $\mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))]$: 鉴别器希望能够尽可能鉴别生成器生成的假的数据，所以希望 $D(G(z))$ 尽可能小，反之 $1 - D(G(z))$ 尽可能大
 - $\mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)]$: 鉴别器希望能够鉴别真实的数据，所以希望 $D(x)$ 尽可能大
- 同理，作为博弈，生成器希望上面的情况尽可能反着来，生成器是一个可微函数(例如，MLP)，将噪声映射到数据空间
 - 训练 G 最小化 $\log(1 - D(G(z)))$ ，即欺骗鉴别器
- 对于一个固定的 G ，损失函数是有效的二元交叉熵
- 极大极小策略用于零和博弈，即一方的损失=对手的收益
- 极大极小解就是**纳什均衡**
 - 在纳什均衡中， D 和 G 没有任何改变的动机因为这样做会使目标函数的值变差
 - 在机器学习术语中，这意味着我们达到了联合优化问题的梯度下降收敛

训练 GAN

训练鉴别器

对于鉴别器，做梯度上升

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

训练生成器

对于生成器，梯度下降（最小化鉴别器正确的对数概率）

$$G^* = \arg \min_G V(G, D) = \arg \min_G \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))]$$

但在实际训练中，在发生器上做梯度上升（最大化鉴别器出错的对数概率）

$$G^* = \arg \max_G \mathbb{E}_{z \sim p_z(z)} \log (D(G(z)))$$

我们将在后面的内容中说明为什么要这么做

训练伪代码

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

```
for number of training iterations do
    for  $k$  steps do
        • Sample minibatch of  $m$  noise samples  $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$  from noise prior  $p_g(\mathbf{z})$ .
        • Sample minibatch of  $m$  examples  $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$  from data generating distribution  $p_{\text{data}}(\mathbf{x})$ .
        • Update the discriminator by ascending its stochastic gradient:
```

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

```
end for
• Sample minibatch of  $m$  noise samples  $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$  from noise prior  $p_g(\mathbf{z})$ .
• Update the generator by descending its stochastic gradient:
```

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

```
end for
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.
```

- 在实践中，这在一开始就饱和了，因为 D 拒绝了高置信度的样本，因为它们与训练数据明显不同

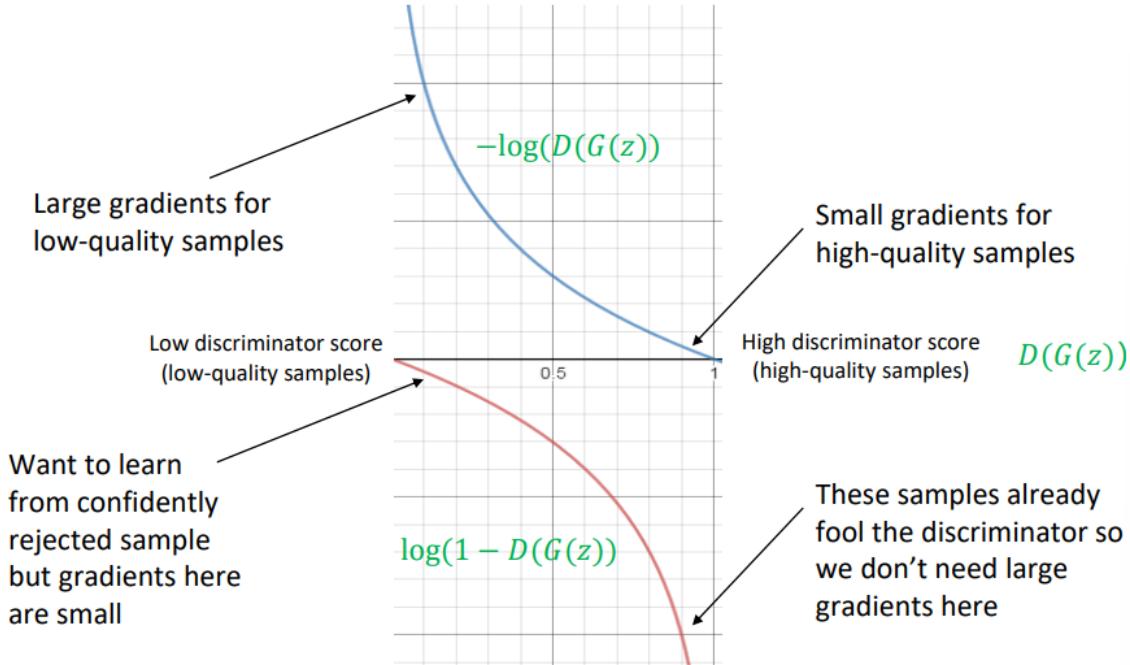
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log \left(1 - D \left(G \left(z^{(i)} \right) \right) \right).$$

这里 $D(G(z^{(i)}))$ 接近为 0，几乎是饱和，所以很难进行梯度下降

- 因此，我们用下面这个损失函数代替生成器的函数，即生成器器最大化了鉴别器错误的概率

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

$$\min_{w_G} \mathbb{E}_{z \sim p} \log(1 - D(G(z))) \text{ vs. } \max_{w_G} \mathbb{E}_{z \sim p} \log(D(G(z)))$$

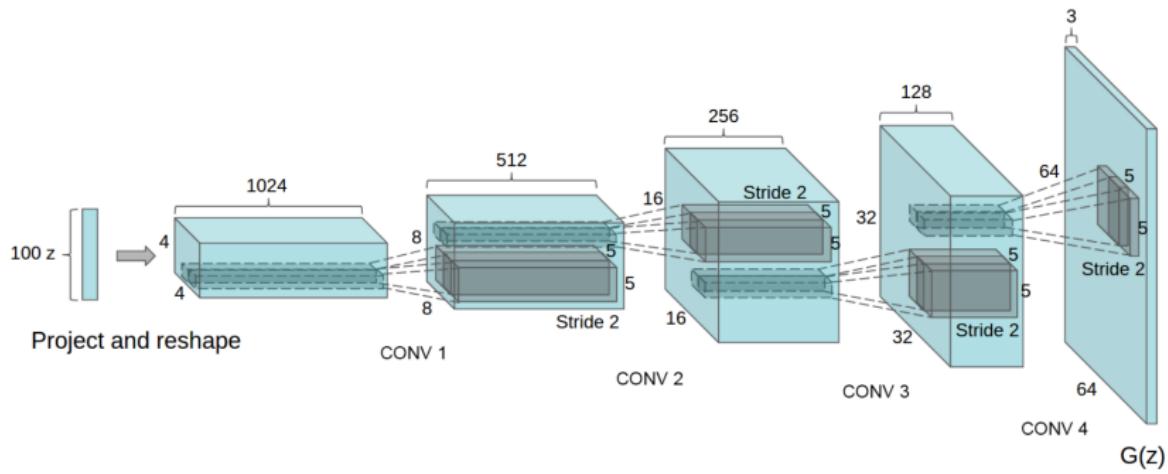


GAN 存在的问题

- 在现实世界中，我们在训练 GAN 时面临多种问题
 - 样本容量有限**: 训练集是有限的，不是完全分布的
 - 容量有限**: 生成器的容量有限，不能很好地代表任何分布
 - 优化错误**: 优化器可能会陷入局部最优或者永远不会收敛到全局最优
 - 鞍点问题**: 比找到最大值或最小值更难
 - 平衡更新**: D 太弱/强意味着 G 没有办法去更新梯度

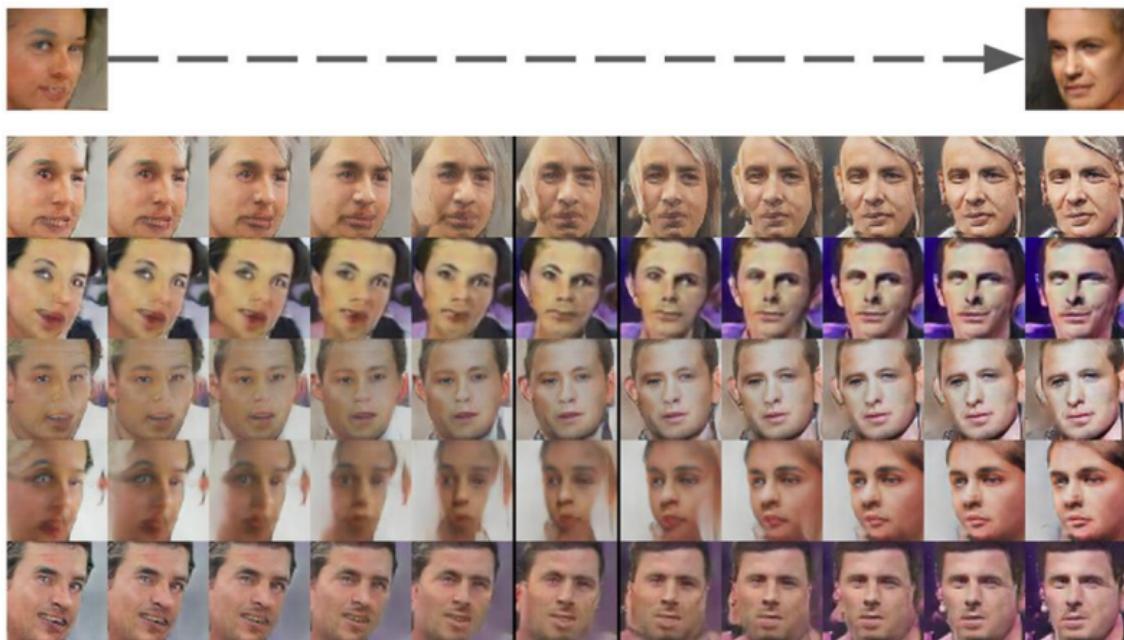
3. GAN 示例

DCGAN



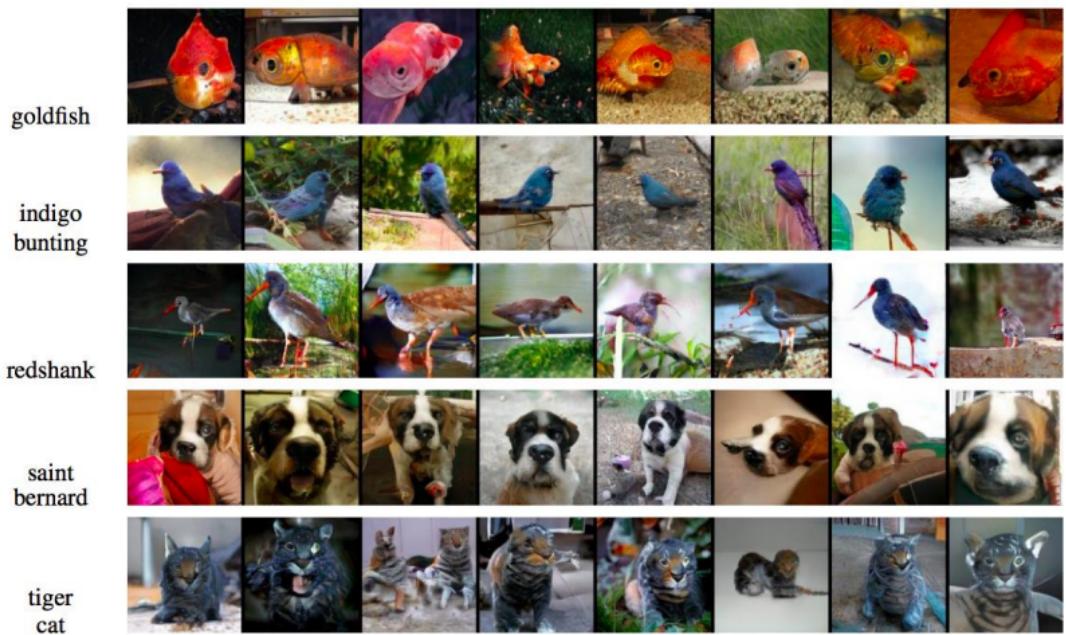
用于 LSUN 场景建模的 DCGAN 生成器，将一个 100 维均匀分布的 Z 投影到一个小空间范围的卷积表示上，并具有许多特征映射

一系列的四小步卷积（在最近的一些论文中，这些被错误地称为反卷积）然后将这种高级表示转换为 64×64 像素的图像，值得注意的是，没有使用全连接层或池化层



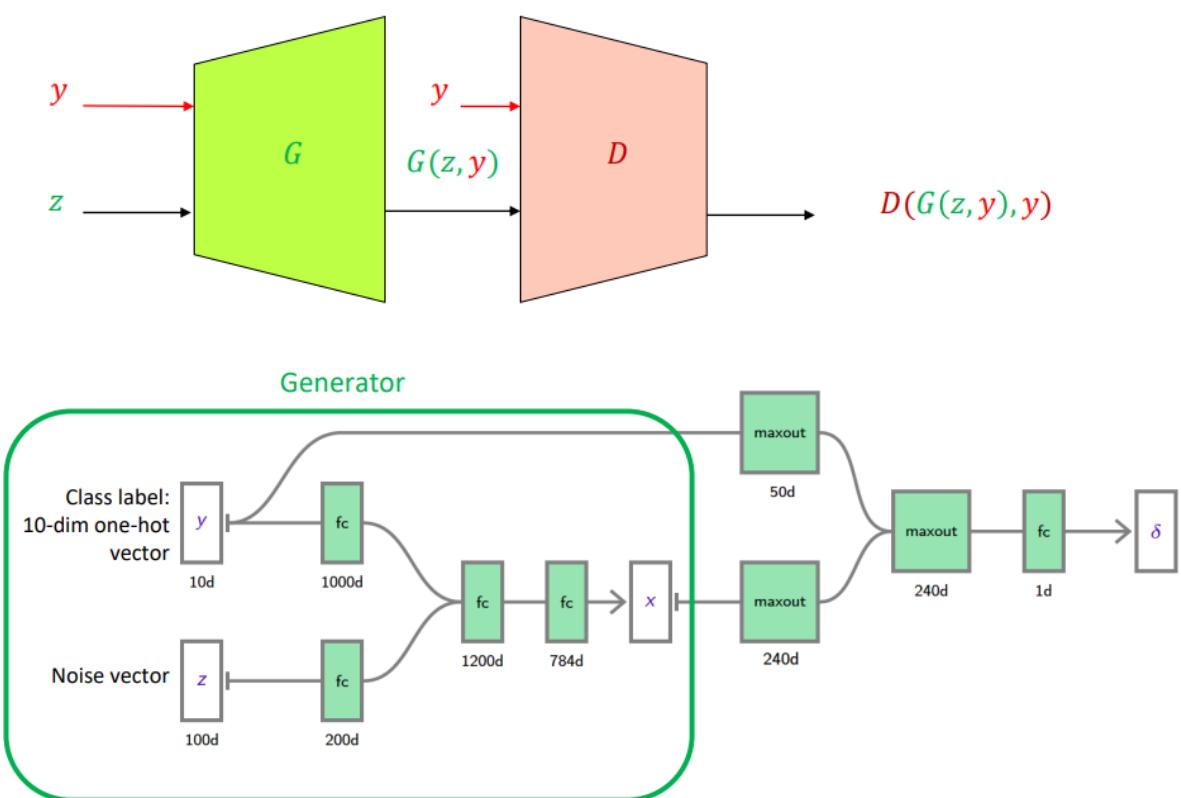
- 通过添加一个“转弯”矢量来进行姿态变换

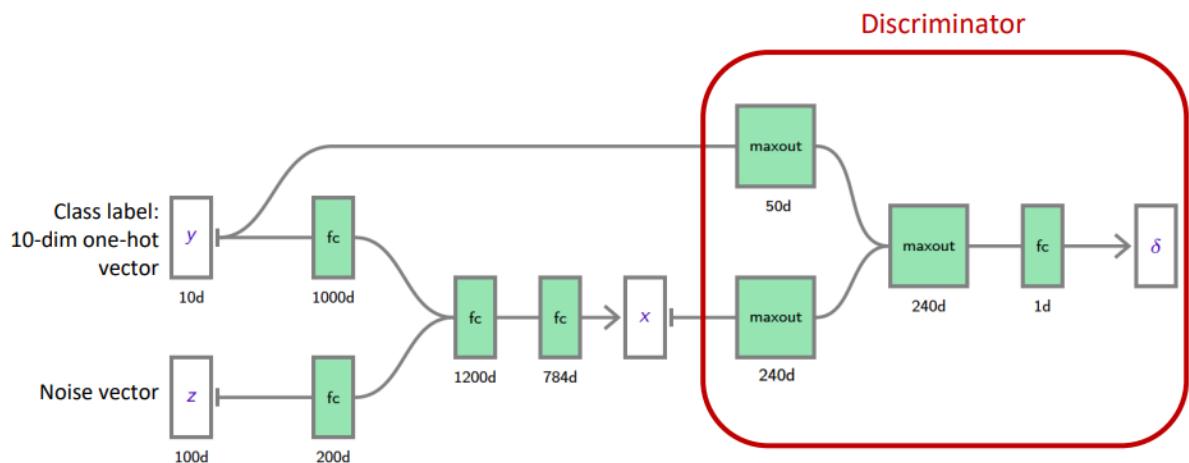
cGAN 有条件生成



- 假设我们想将样本的生成条件设置为离散类信息(label) y

- 我们如何将 y 添加到基本的 GAN 框架中?

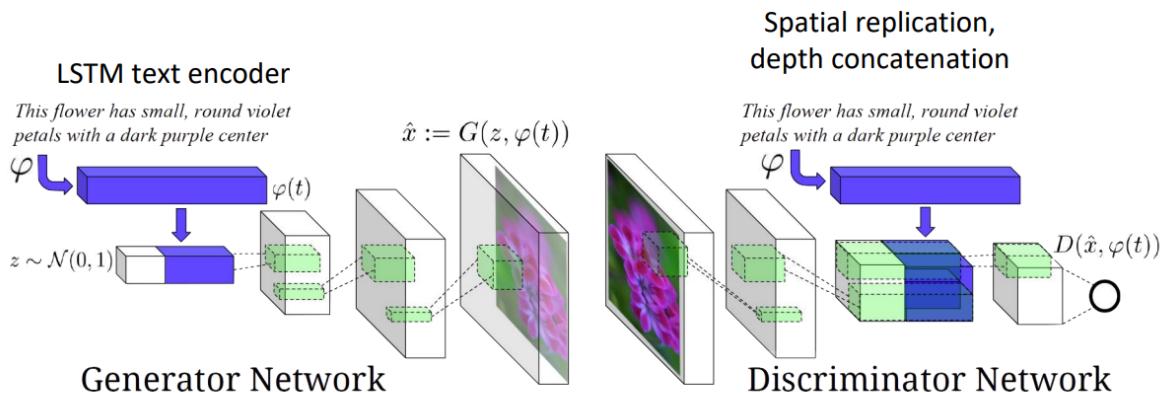




在 MNIST 手写训练集上训练效果如下



另外一个示例：文本->图片合成



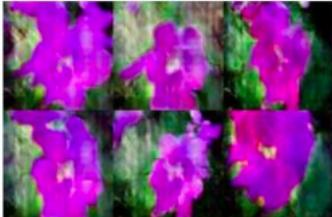
this small bird has a pink breast and crown, and black primaries and secondaries.



this magnificent fellow is almost all black with a red crest, and white cheek patch.



the flower has petals that are bright pinkish purple with white stigma



this white and yellow flower have thin white petals and a round yellow stamen

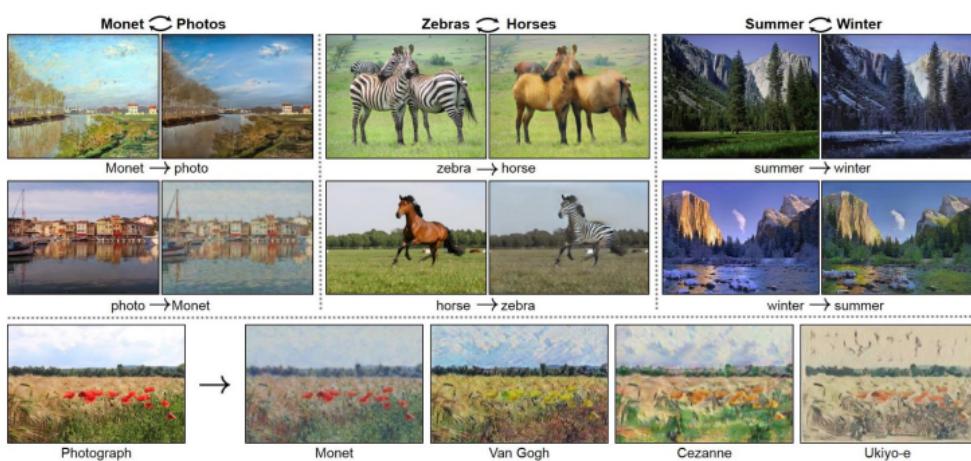


应用

人脸生成



风格转换



图像转移

