人工智能原理与技术

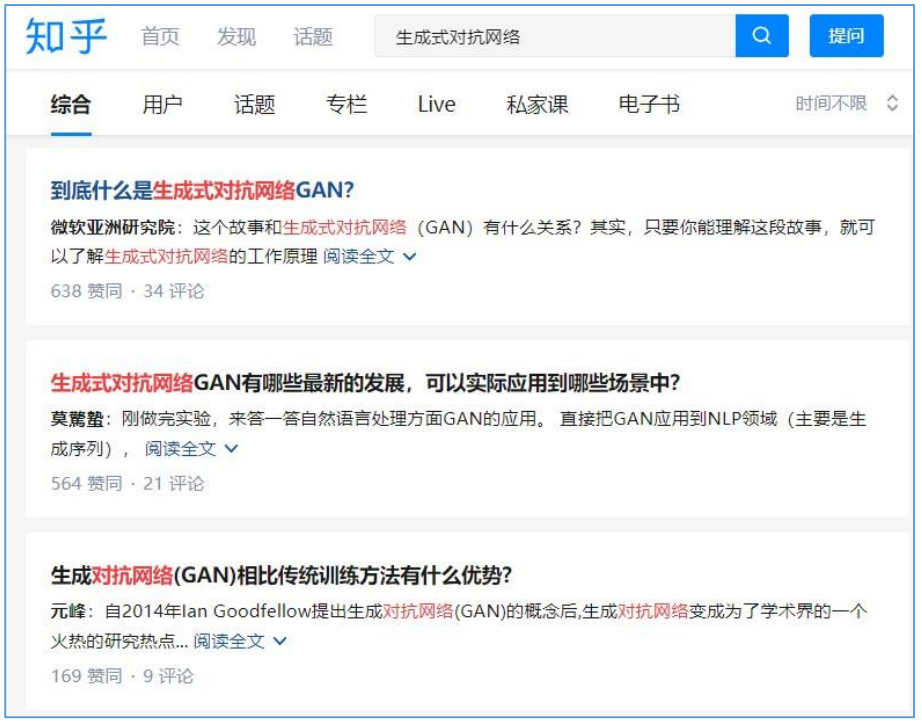
生成式对抗网络


新智元
2019年1月7日 10:33
7纳米！华为鲲鹏920芯片问世，最强性能面向智能计算
【图深度学习时代降临】清华朱文武组一文综述GraphDL五类...
机器学习新隐写术：伪装训练集，隐藏任务意图
他评出2018年GAN论文榜TOP 3，你同意吗？
科研新手写论文常犯的15个错误，请收好这份写作指南
2019年1月8日 10:30
合作投稿
文章搜索
头条号


机器之心
2019年1月2日 12:06
裤子换裙子，就问你GAN的这波操作秀不秀
AI用你的脸生成假视频，你却不能删除，斯嘉丽·约翰逊：这是...
我不是机器人：谷歌最新版验证码系统ReCaptcha破解已开源
只有条件GAN才能稳定训练？对抗+自监督的无监督方法了解一下
2019年1月3日 12:24
关于我们
搜索文章
精选内容


GAN生成式对抗网络
【学界】GANs最新综述论文：生成式对抗网络及其变种如何有用
最近一期的计算机顶级期刊ACM Computing Surveys (CSUR)出版，涵盖最新的GANs综述论文,146篇参考文献，本文的作者来自首尔大学数据科学与人工智能实验室的师生，研究方向为深度学习和机器学习。
2019年1月6日 23:30


GAN生成式对抗网络
星期五 21:19
【学界】效果惊人：上古卷轴III等经典游戏也能使用超分辨率GAN重制了
最近有研究者使用超分辨率技术「高清化」曾经的经典游戏，他们从去年 12 月开始陆续推出几款「高清重制版」游戏，包括《重返德军总部》、《毁灭战士》和《上古卷轴 III：晨风》，而这几天更是放出《马克思·佩恩》的高清版。
星期六 22:19


知乎
首页
发现
话题
生成式对抗网络
提问
综合
用户
话题
专栏
Live
私家课
电子书
时间不限
到底什么是生成式对抗网络GAN?
微软亚洲研究院：这个故事和生成式对抗网络（GAN）有什么关系？其实，只要你能理解这段故事，就可以了解生成式对抗网络的工作原理 阅读全文
638 赞同 · 34 评论
生成式对抗网络GAN有哪些最新的发展，可以实际应用到哪些场景中?
莫鹦鹉：刚做完实验，来答一答自然语言处理方面GAN的应用。 直接把GAN应用到NLP领域（主要是生成序列）， 阅读全文
564 赞同 · 21 评论
生成对抗网络(GAN)相比传统训练方法有什么优势?
元峰：自2014年Ian Goodfellow提出生成对抗网络(GAN)的概念后,生成对抗网络变成为了学术界的一个火热的研究热点... 阅读全文
169 赞同 · 9 评论


全部
博客
下载
论坛
学院
问答
图文课
生成式对抗网络
搜一搜
5天搞定深度学习进阶系列教程 - CSDN学院
学院
讲师：董秉丰
2391人在学
深度学习 — 进阶系列
这门课程是"深度学习入门系列培训教程"的后续内容，深度学习进阶系列培训课程概况： 在深度学习入门系列的课程中，我们基本掌握了一些基础知识，在这门课程中我们将进一步学习深度学习...理、核心内容。包括深度置信网络DBN学习，卷积神经网络CNN入门到精通，深度残差网络RES和
https://edu.csdn.net/course/detail/5462
生成对抗网络(GAN)简单梳理 - 时光杂货店 - CSDN博客
博客
无需定义明确的概率密度函数,代表方法包括马尔科夫链、生成对抗式网络,该系列方法无需定义数据分布的描述函数...
https://blog.csdn.net/xg123321123/article/details/78034859
生成式对抗网络(GAN)如何快速理解?这里有一篇最直观的..._CSDN博客
博客
原文来源:freeCodeCamp作者:Thalles Silva「雷克世界」编译:嗯~阿童木呀、KABUDA让我们假设这样一种情景:你的...
https://blog.csdn.net/Ksf3kg7dU95rn0XL/article/details/79015258

【GAN这4.5年】IanGoodfellow谈论生成对抗网络在人脸生成进展，5篇经典论文，从黑白模糊到惟妙惟肖

原创：专知 专知 今天

【导读】生成式对抗网络(Generative Adversarial Networks， GAN）自在2014年被Ian Goodfellow提出后，取得巨大的进展，在理论算法和应用方面有着丰富成果。近日，GAN之父Ian Goodfellow在twitter谈论了关于GAN在人脸生成的4年半进展，包含5篇代表性文章，值得关注，人脸生成从早起的模糊阶段进化到现在逼真的程度。

GAN在人脸生成这4.5年进展

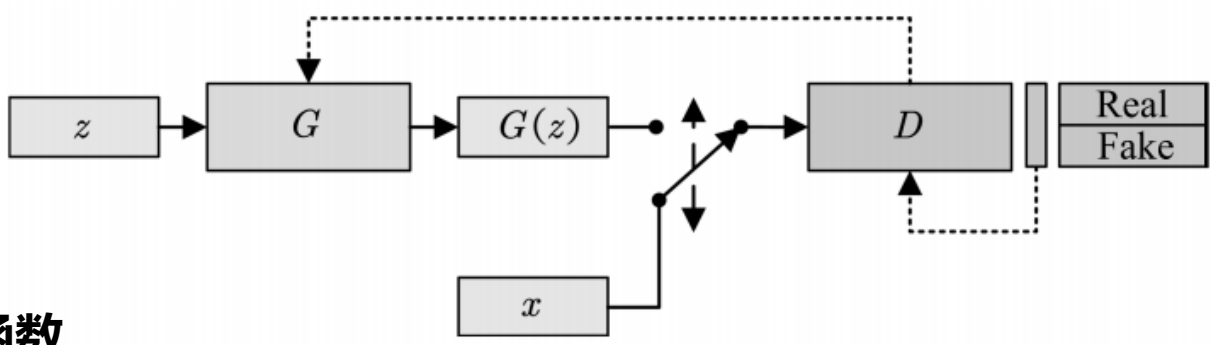
目录

CONTENTS

背景

Ian Goodfellow

模型



目标函数

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

作者 Ian Goodfellow 于2014年NIPS首次提出了GAN的概念

背景

Yann LeCun

What are some recent and potentially upcoming breakthroughs in deep learning?

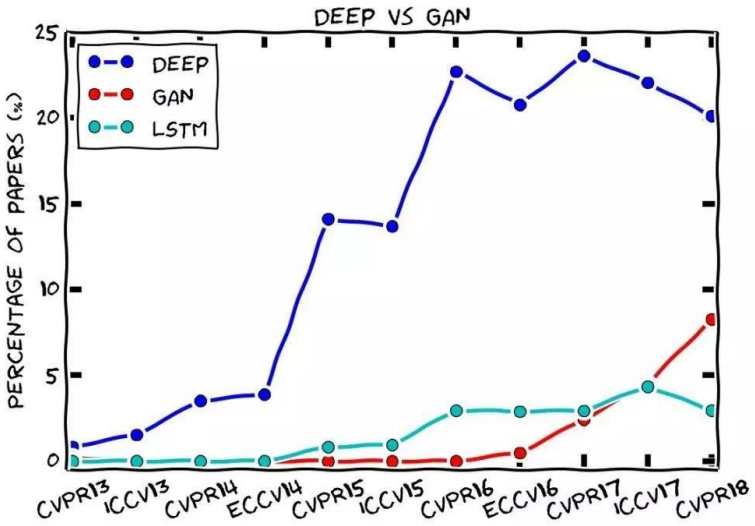
Yann LeCun, Director of AI Research at Facebook and Professor at NYU
Answered Jul 29, 2016 · Upvoted by Joaquin Quiñonero Candela, studied Machine Learning and Gokul Krishnan, M.Sc Computer Science & Machine Learning, ETH Zurich (2018)

The most important one, in my opinion, is adversarial training (also called GAN for Generative Adversarial Networks). This is an idea that was originally proposed by Ian Goodfellow when he was a student with Yoshua Bengio at the University of Montreal (he since moved to Google Brain and recently to OpenAI). This, and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion.

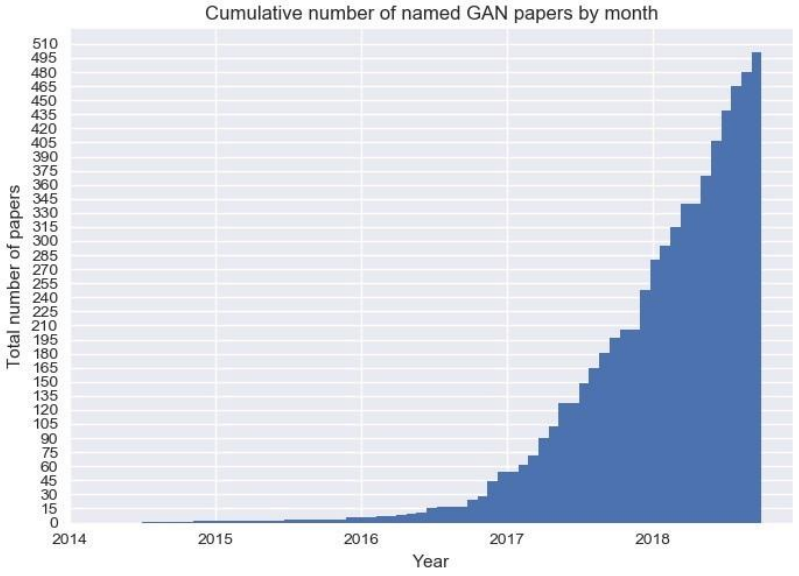
Yann LeCun 是美国工程院院士
Facebook人工智能研究院院长
纽约大学Sliver教授
创立卷积神经网络
人工智能三巨头之一

背景

论文数量



计算机视觉顶级会议（CVPR, ICCV, ECCV）上，GAN已经超过LSTM，成为重要关键词。



ArXiv上，每个月和GAN相关的论文数量

https://github.com/hindupuravinash/the-gan-zoo

目录

CONTENTS

GAN案例

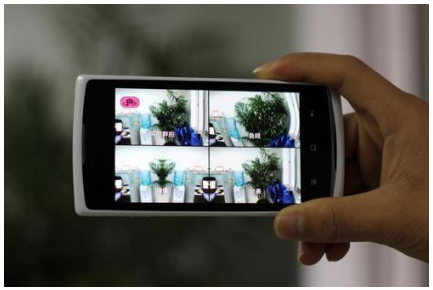
图像着色

用途：旧图像修复

GAN案例

图像超像素

https://github.com/JustinhoCHN/SRGAN_Wasserstein

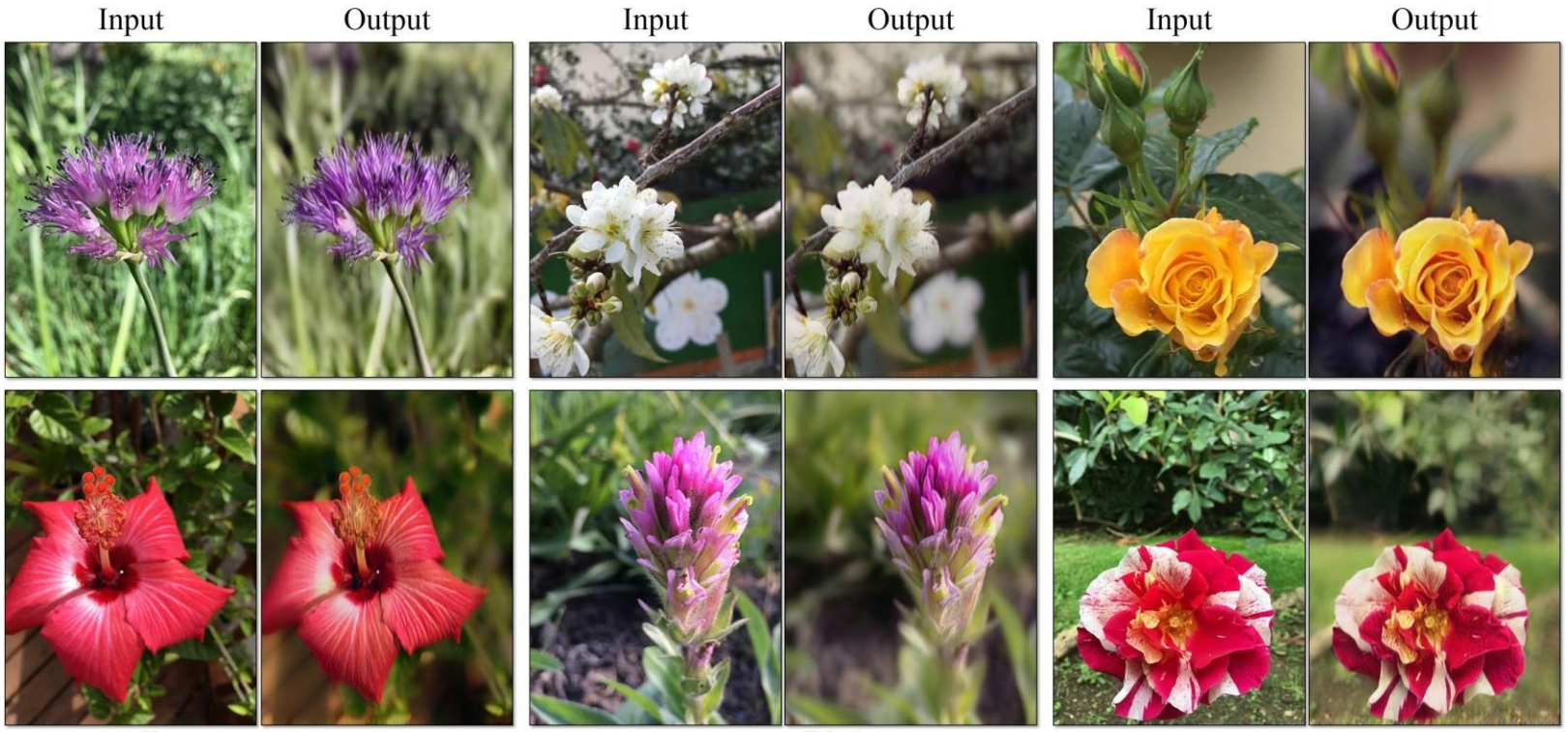


用途：手机相机

超分辨率算法已经广泛的应用于我们的智能手机中

GAN案例

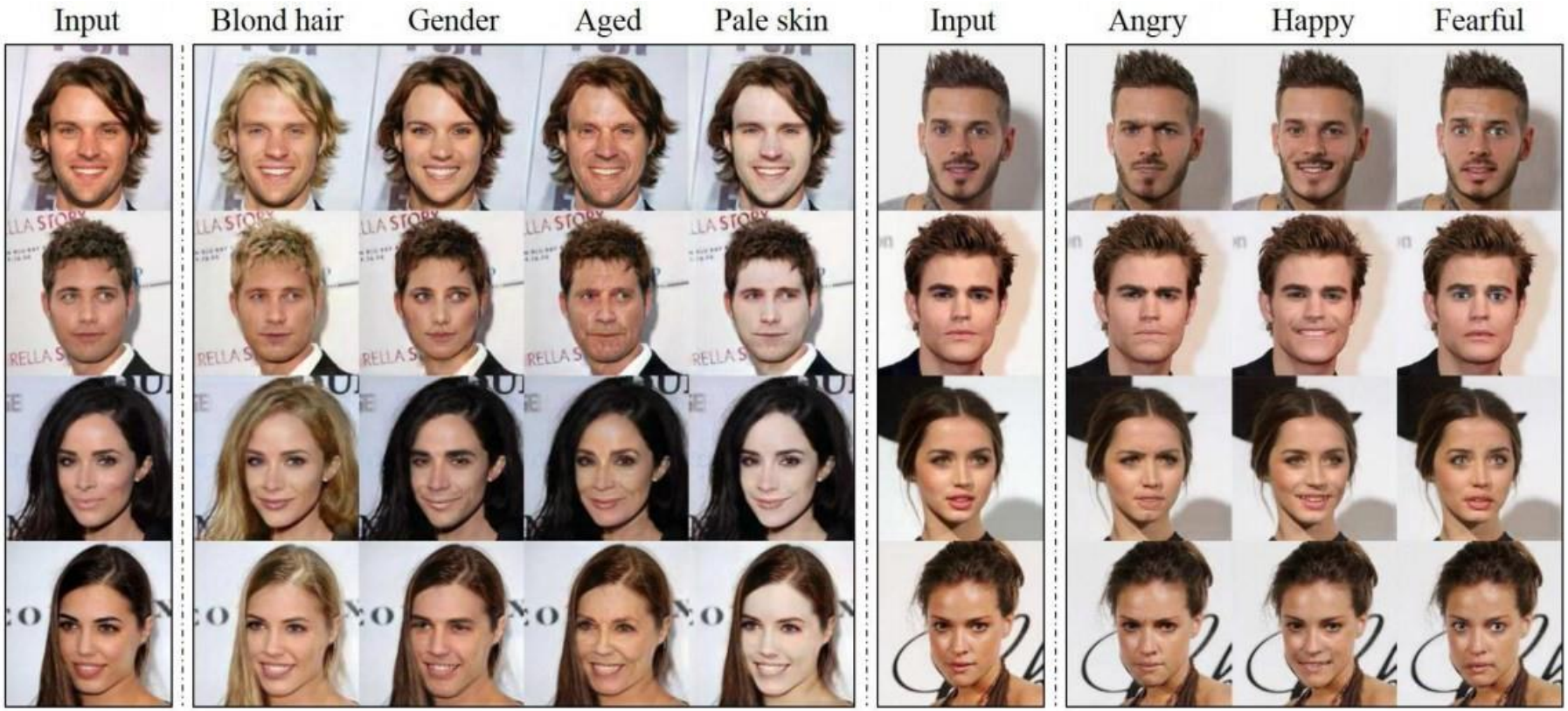
背景模糊



https://github.com/junyanz/CycleGAN

GAN案例

人脸生成

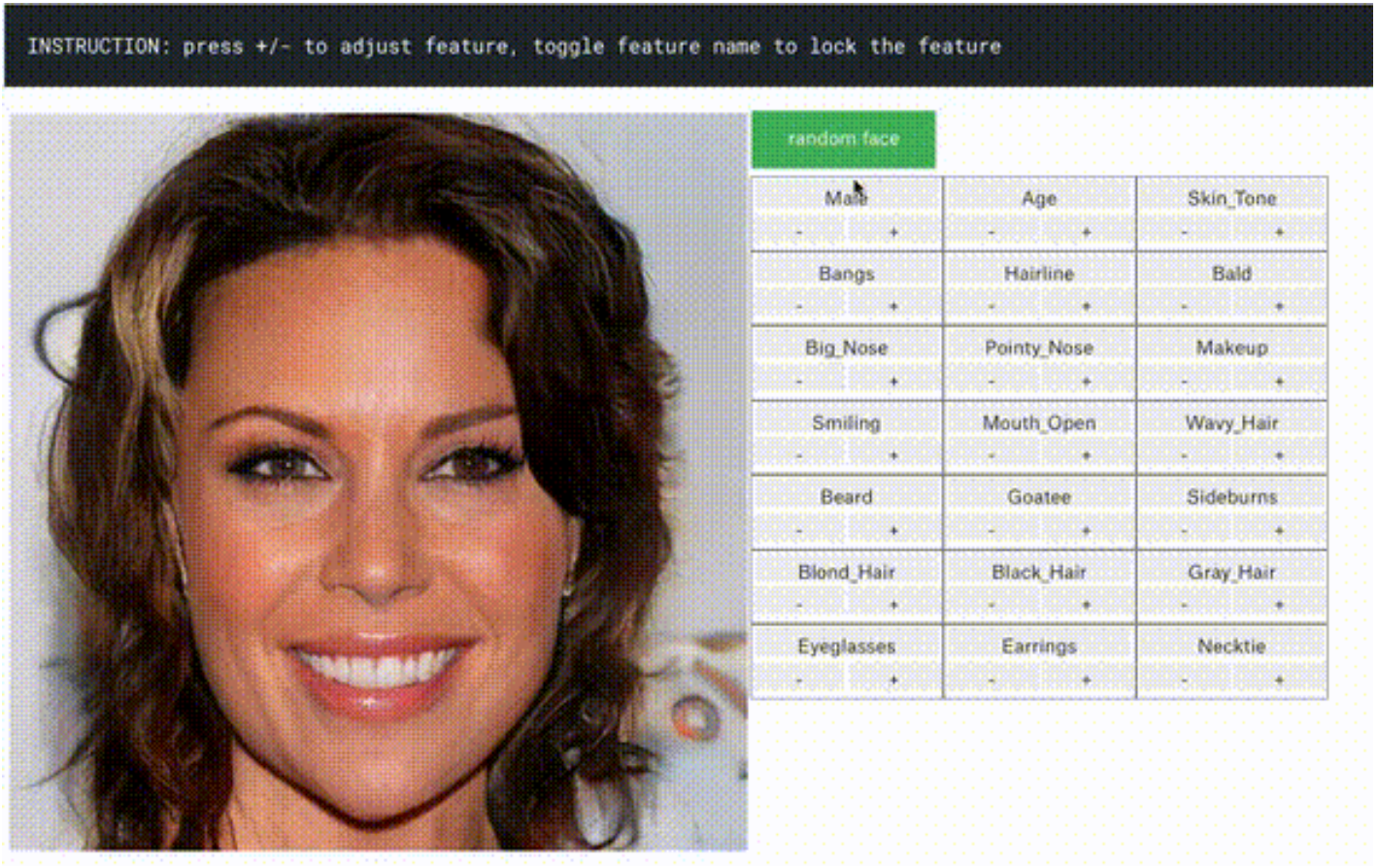


https://github.com/yunjey/stargan

用途：娱乐软件，抖音，美图等

GAN案例

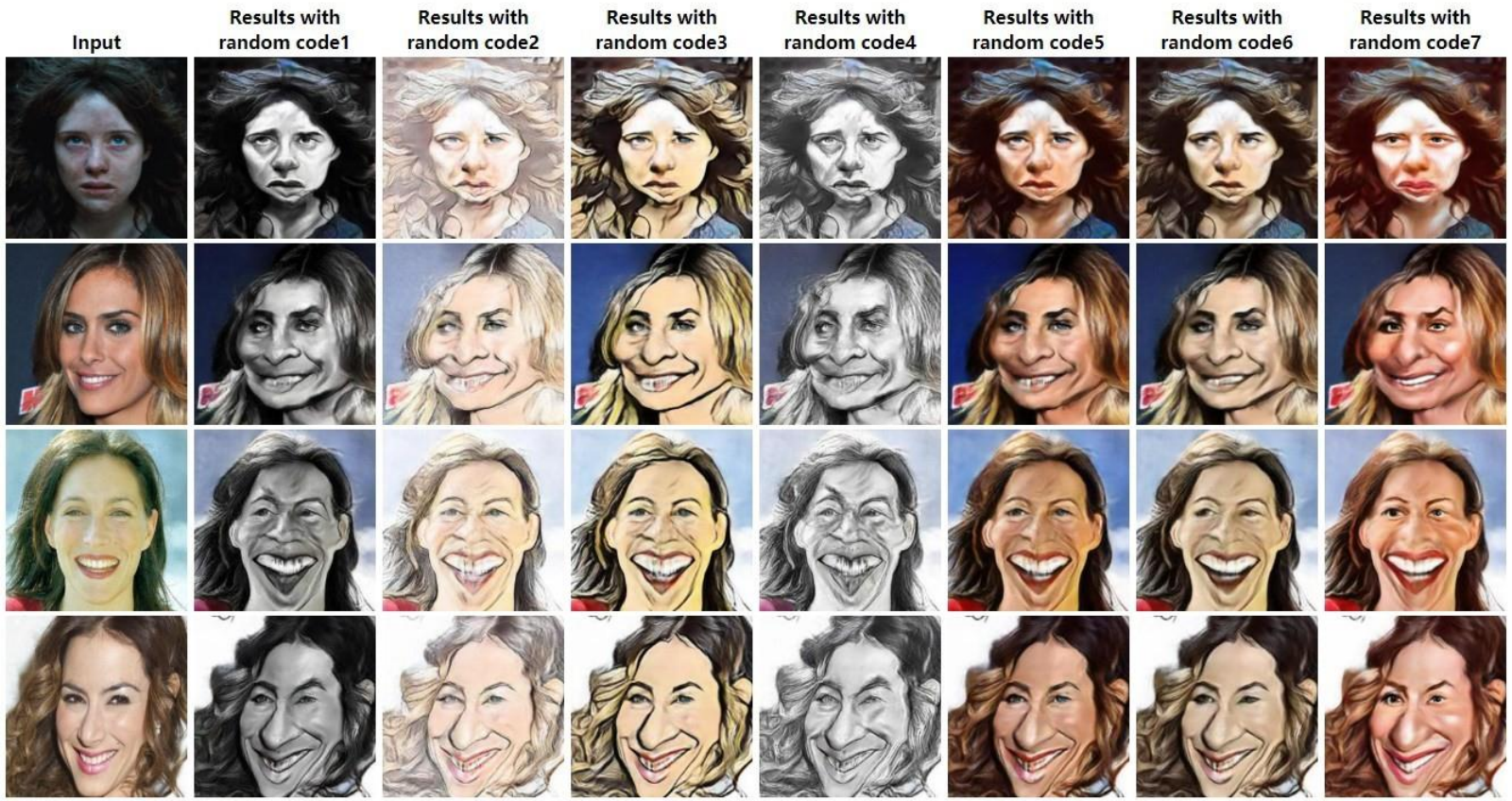
人脸定制



https://github.com/SummitKwan/transparent_latent_gan

GAN案例

卡通头像生成



CariGANs: Unpaired Photo-to-Caricature Translation

GAN案例

文本生成图片

https://github.com/hanzhanggit/StackGAN

GAN案例

字体变换

https://github.com/kaonashi-tyc/zi2zi

GAN案例

风格变换

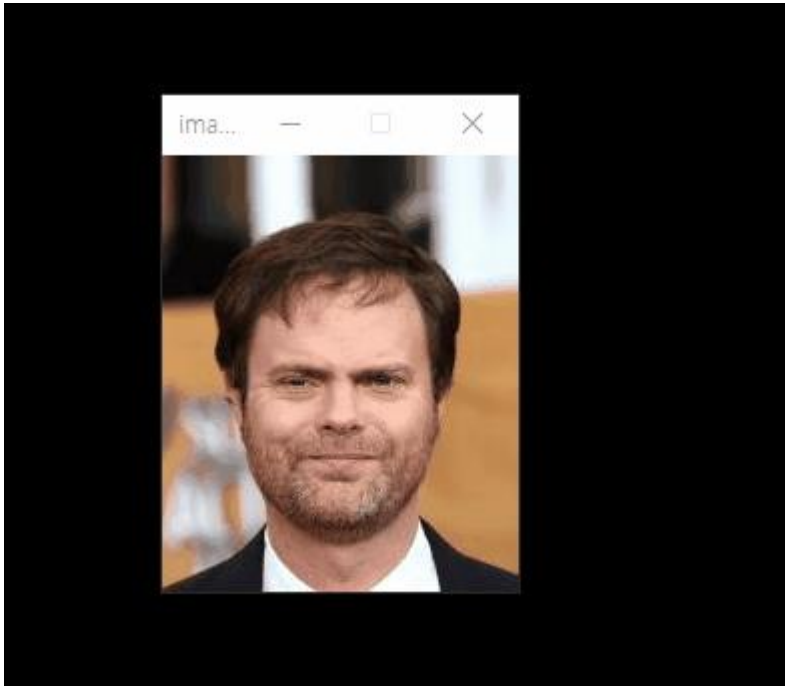
https://github.com/junyanz/CycleGAN

winter Yosemite → summer Yosemite

summer Yosemite → winter Yosemite

GAN案例

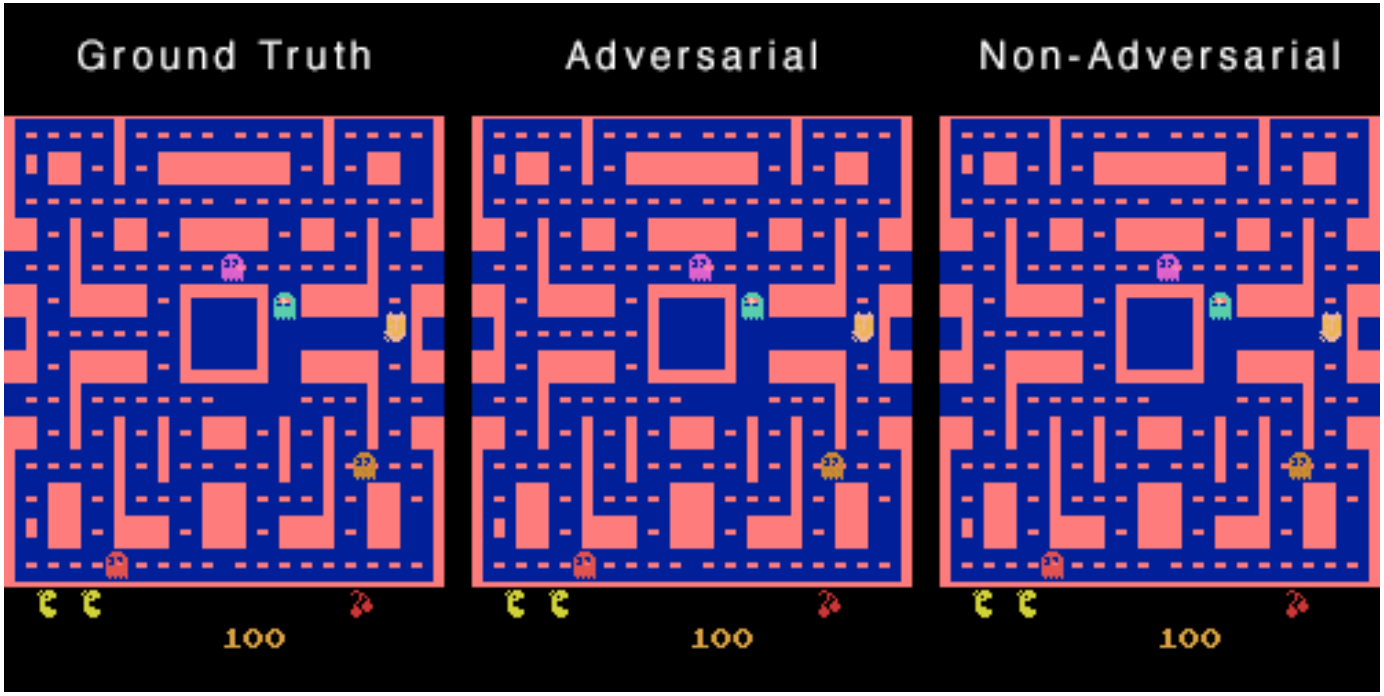
图像修复



https://github.com/shinseung428/GlobalLocalImageCompletion_TF

GAN案例

帧预测



https://github.com/dyelax/Adversarial_Video_Generation

目录

CONTENTS

GAN

生成模型和GAN

低维向量

高维数据
（图片、文本、语音）

不严谨的定义：一个能够生成我们想要的数据的模型(图模型、函数、神经网络)。

GAN

生成模型和GAN

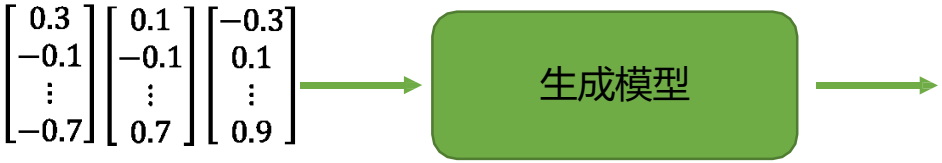
低维向量

高维数据
（图片、文本、语音）

不严谨的定义：一个能够生成我们想要的数据的模型(图模型、函数、神经网络)。

图像生成

文本生成



生成式对抗网络（GAN）的目的是训练这样一个生成模型，生成我们想要的数据。

生成式对抗网络中**生成**一词的由来。

目录

简介

GAN

$$\begin{bmatrix}0.1\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.5\\0.3\\0.4\\0.2\end{bmatrix}$$

→ 生成器（全连接神经网络） →

生成随机数

cGAN

$$\begin{bmatrix}0.1\\0.3\\0.4\\1\end{bmatrix}\begin{bmatrix}0.2\\0.3\\0.4\\3\end{bmatrix}\begin{bmatrix}0.3\\0.3\\0.4\\5\end{bmatrix}\begin{bmatrix}0.4\\0.3\\0.4\\7\end{bmatrix}\begin{bmatrix}0.5\\0.3\\0.4\\9\end{bmatrix}$$

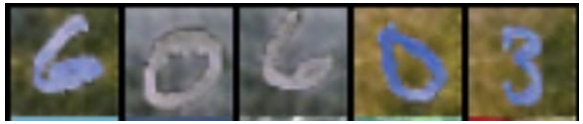
→ 生成器（全连接神经网络） →

生成可控数据

DCGAN

$$\begin{bmatrix}0.1\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.5\\0.3\\0.4\\0.2\end{bmatrix}$$

→ 生成器（卷积神经网络） →



使用卷积神经网络
更适合图片生成

WGAN
WGAN-GP

$$\begin{bmatrix}0.1\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.5\\0.3\\0.4\\0.2\end{bmatrix}$$

→ 生成器（WGAN） →

重新设计目标函数
训练更稳定，生成数据质量更棒

目录

CONTENTS

GAN

零和博弈

为什么罪犯制造的假币越来越逼真？
为什么GAN可以生成数据？

罪犯

使用假币，购买商品
目标：制造逼真假币



？？？

？？？

商贩

收到真币，出售商品
收到假币，不出售商品
目标：准确区分真假币

正版人民币
是红色的！

正版人民币有
精美的花纹！

正版人民币有
毛爷爷头像！

正版人民币

GAN

零和博弈




？？？

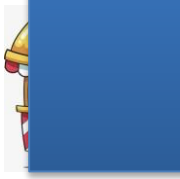
罪犯

使用假币，购买商品
目标：制造逼真假币

？？？

有

大家后面学习时请思考两个问题：
为什么罪犯制造的假币越来越真？
为什么GAN可以生成数据？



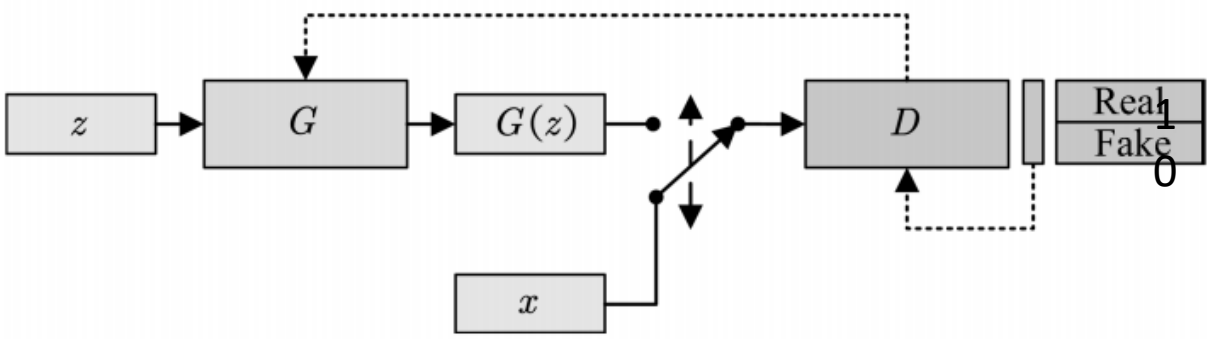
收到真币，出售商品
收到假币，不出售商品
目标：准确区分真假币

正版人民币

GAN

生成式对抗网络

GAN框架



GAN工作原理

随机噪声z：从一个先验分布（人为定义，一般是**均匀分布**或者**正态分布**）中随机采样的向量

$$\begin{bmatrix} 0.1 \\ 0.3 \\ 0.4 \\ 1 \end{bmatrix}$$

真实样本x：从数据库中采样的样本； 合成样本G(z)：生成模型G输出的样本。

判别器(Discriminator)：**区分真实(real)样本和虚假(fake)样本**。对于真实样本，尽可能给出高的评分1；对于虚假数据，尽可能给出低个评分0.

生成器(Generator)：**欺骗判别器。**生成虚假数据，使得判别器D能够尽可能给出高的评分1。

GAN目标函数

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

等价于

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

GAN

生成器

生成器：生成数据

低维向量　　(可以是一个**NN**，或者其他函数)　　数据（图片、文本、语音）

$$\begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.5\\0.3\\0.4\\0.2\end{bmatrix}$$

GAN

判别器

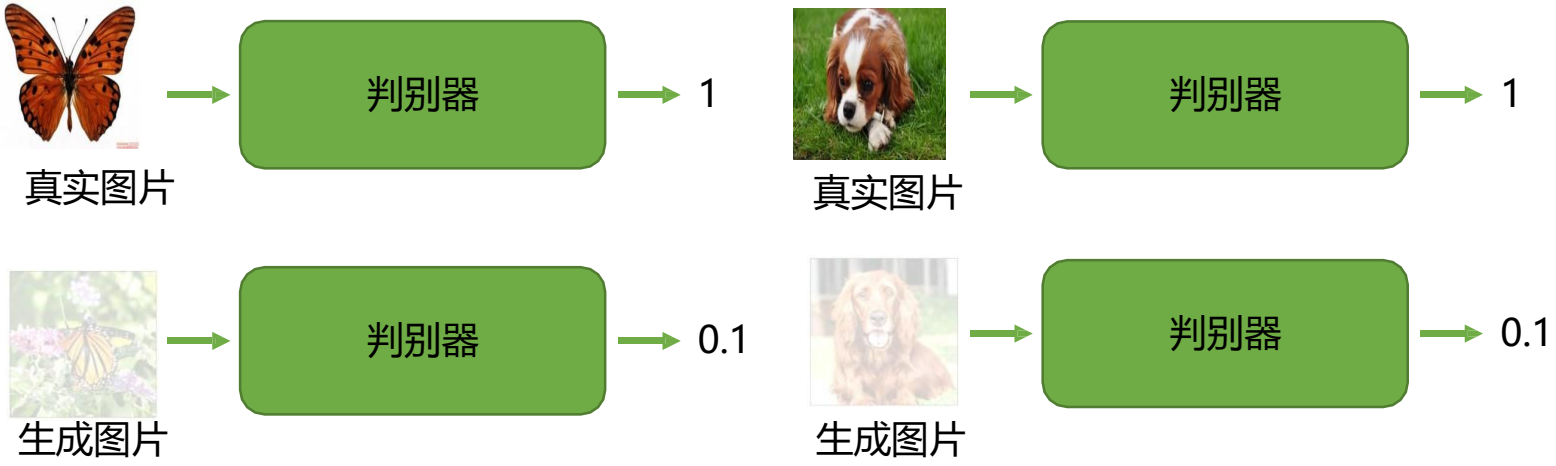
判别器：区分真样本和假样本

数据
（图片、文本、语音）

（可以是一个NN，
或者其他函数）

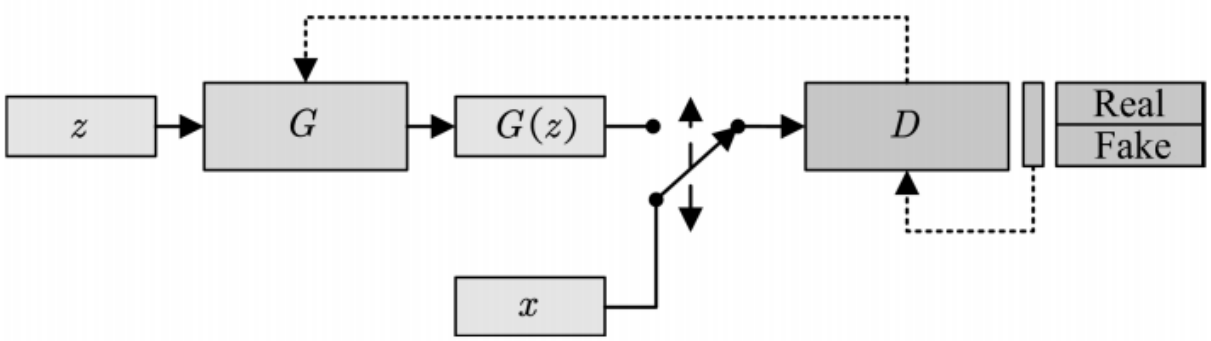
输入数据属于
真实数据的概率

以图像为例



GAN

GAN



【核心思想】判别器：区分真假样本；生成器：欺骗判别器

目标函数

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

等价于

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

GAN

博弈

生成器

罪犯

使用假币，购买商品
目标：制造逼真假币

欺骗判别器

对抗训练

判别器

商贩

收到真币，出售商品
收到假币，不出售商品
目标：准确区分真假币

区分真假数据

更新生成器

？？？

合成数据

？？？

正版人民币
是红色的！

正版人民币有
精美的花纹！

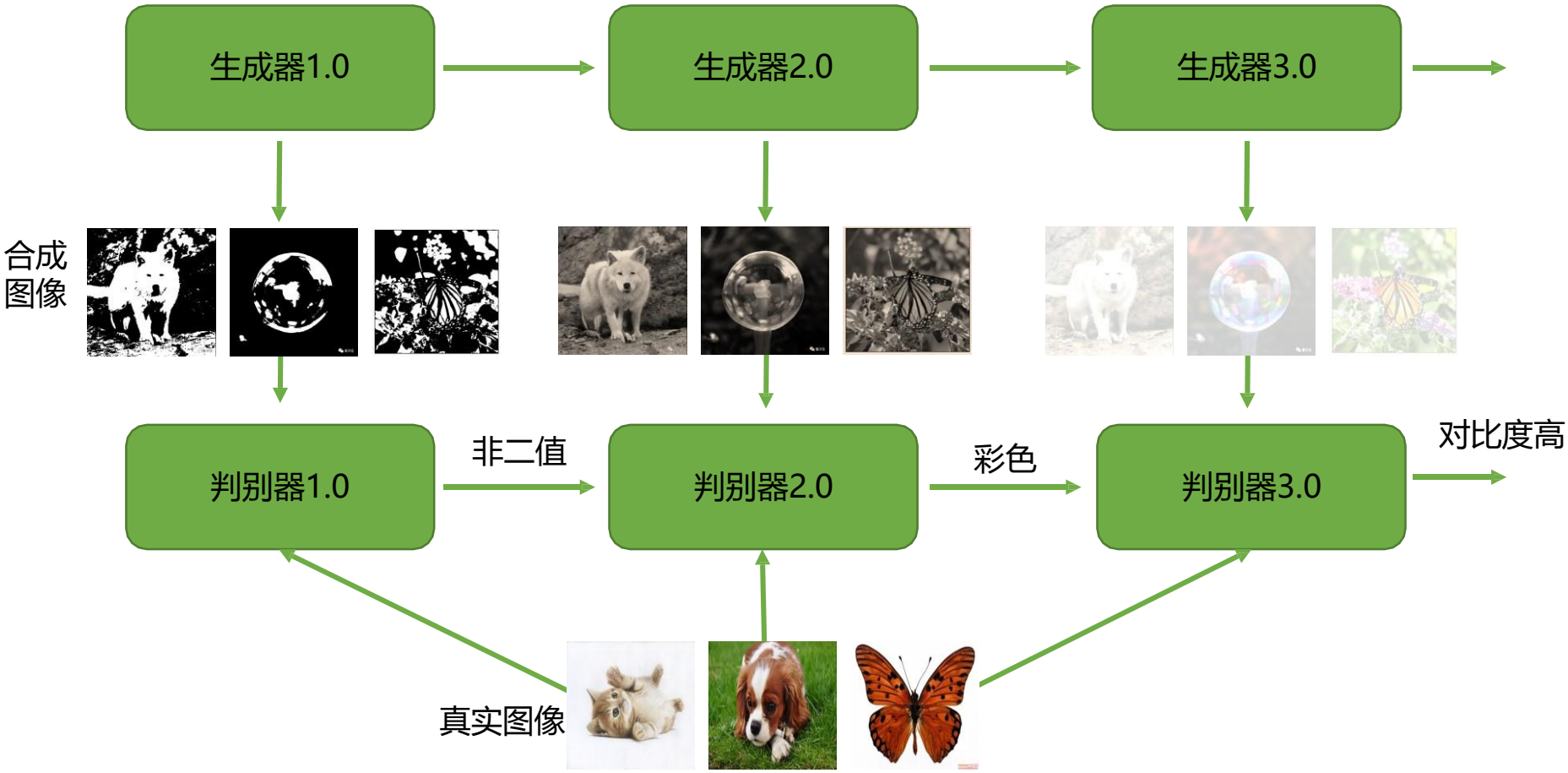
正版人民币有
毛爷爷头像！

更新判别器

正版人民币 **真实数据**

GAN

对抗训练



生成器和判别器之间存在对抗的关系，这也是生成式对抗网络中**对抗**一词的由来

GAN

训练算法

【核心思想】判别器：区分真假样本；生成器：欺骗判别器

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

GAN

【核心思想】判别器：区分真假样本；生成器：欺骗判别器

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

训练算法

1. 随机初始化生成器和判别器

生成器　判别器

2. 交替训练判别器D 和 生成器 G，直到收敛

步骤1：**固定生成器G，训练判别器D区分真实图像与合成图像。**

赋予真实图像高分，赋予合成图像低分

GAN

【核心思想】判别器：区分真假样本；生成器：欺骗判别器

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

训练算法

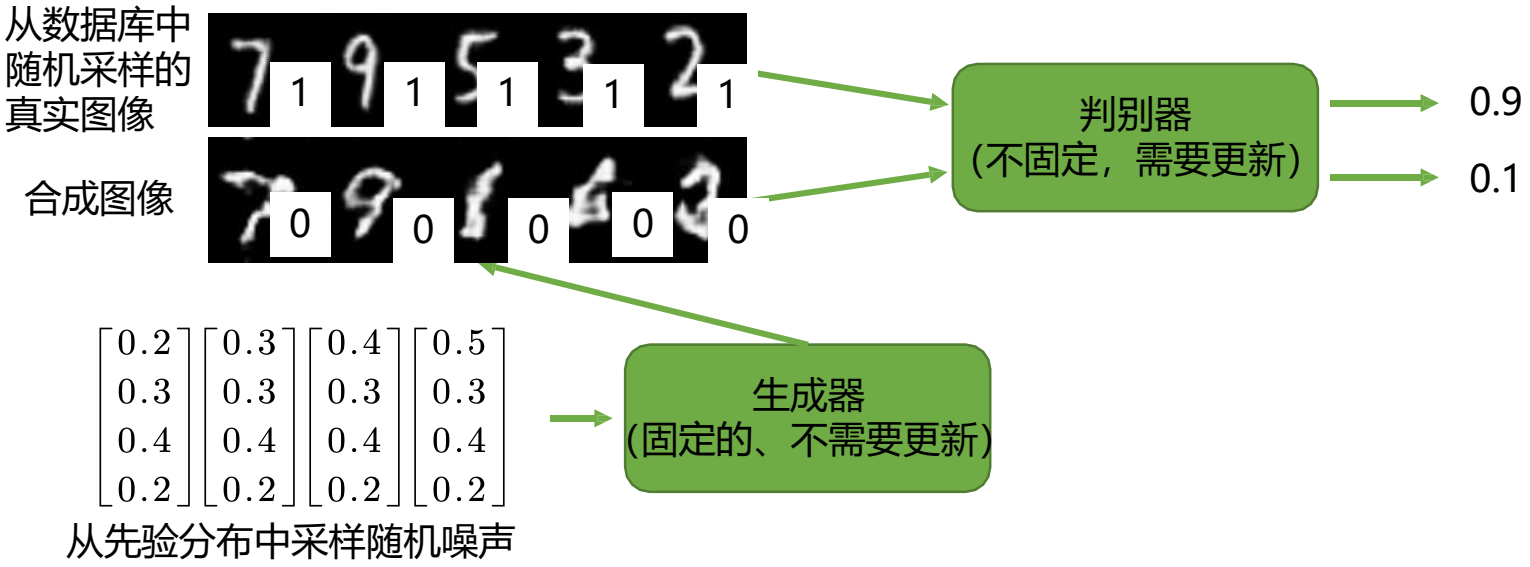
1. 随机初始化生成器和判别器

生成器　判别器

2. 交替训练判别器D 和 生成器 G，直到收敛

步骤1：**固定生成器G，训练判别器D区分真实图像与合成图像。**

赋予真实图像高分，赋予合成图像低分



$$\begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.5\\0.3\\0.4\\0.2\end{bmatrix}$$

请思考：如何训练呢?

GAN

【核心思想】判别器：区分真假样本；生成器：欺骗判别器

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

训练算法

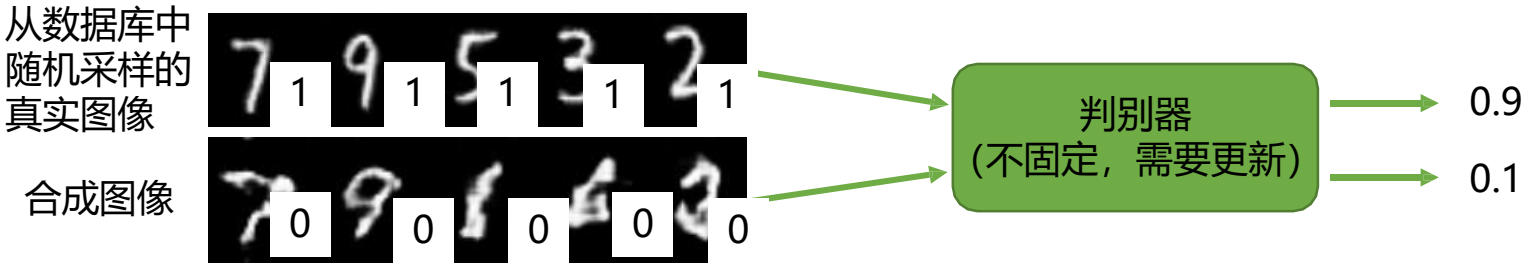
1. 随机初始化生成器和判别器

生成器　判别器

2. 交替训练判别器D 和 生成器 G，直到收敛

步骤1：**固定生成器G，训练判别器D区分真实图像与合成图像。**

赋予真实图像高分，赋予合成图像低分



生成器不需要更新，只要不断生成数据就行

只需要用正常的方式训练一个两类别的分类器！

GAN

【核心思想】判别器：区分真假样本；生成器：欺骗判别器

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

训练算法

1. 随机初始化生成器和判别器

生成器　判别器

2. 交替训练判别器D 和 生成器 G，直到收敛

步骤2：固定判别器D，**训练生成器G欺骗判别器D。**
更新生成器的参数，使其合成的图片被生成器D赋予高分。

GAN

【核心思想】判别器：区分真假样本；生成器：欺骗判别器

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

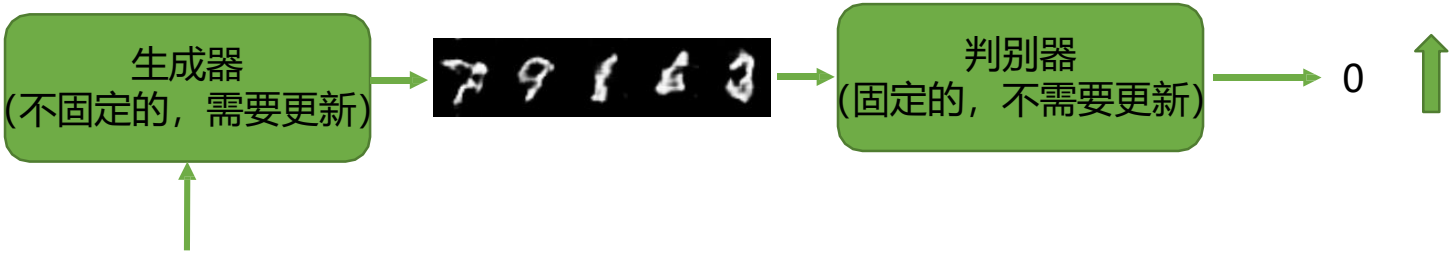
训练算法

1. 随机初始化生成器和判别器

生成器　判别器

2. 交替训练判别器D 和 生成器 G，直到收敛

步骤2：固定判别器D，**训练生成器G欺骗判别器D**。
更新生成器的参数，使其合成的图片被生成器D赋予高分。



$$\begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.5\\0.3\\0.4\\0.2\end{bmatrix}$$

从先验分布中随机采样噪声

请思考：如何优化生成器呢?

GAN

【核心思想】判别器：区分真假样本；生成器：欺骗判别器

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

训练算法

1. 随机初始化生成器和判别器

生成器　判别器

2. 交替训练判别器D 和 生成器 G，直到收敛

步骤2：固定判别器D，**训练生成器G欺骗判别器D**。
更新生成器的参数，使其合成的图片被生成器D赋予高分。

可以反传梯度

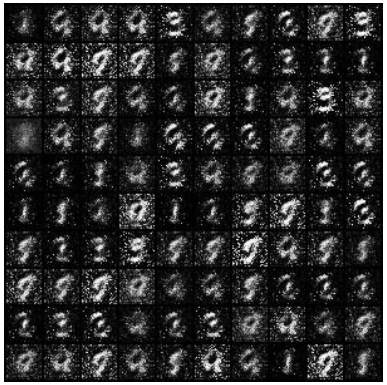
$$\begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.5\\0.3\\0.4\\0.2\end{bmatrix}$$

从先验分布中随机采样噪声

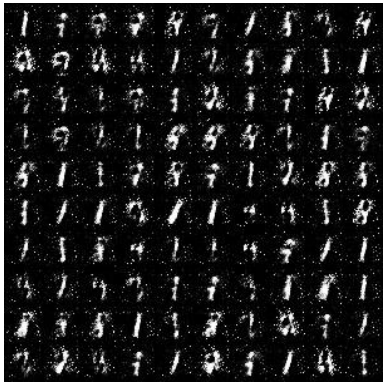
看成输入数据

GAN

实验结果



1 EPOCH



5 EPOCH

10 EPOCH

20 EPOCH

50 EPOCH

200 EPOCH

代码演示

GAN

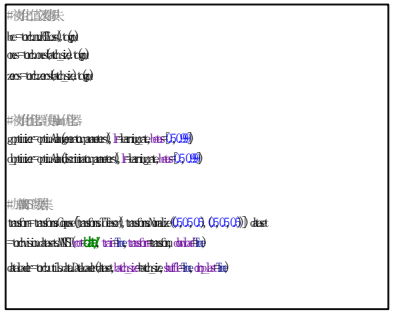
Pytorch实现GAN

准备工作

定义网络结构：判别器和生成器的网络结构
定义目标函数：根据预测标签和真实标签，计算损失
定义优化器：优化器，自动计算梯度并优化网络权重
数据加载：加载数据集

判别器

生成器



其他准备工作

训练直到收敛

加载数据
  提取一个batch的真实数据
  生成一个batch的虚假数据

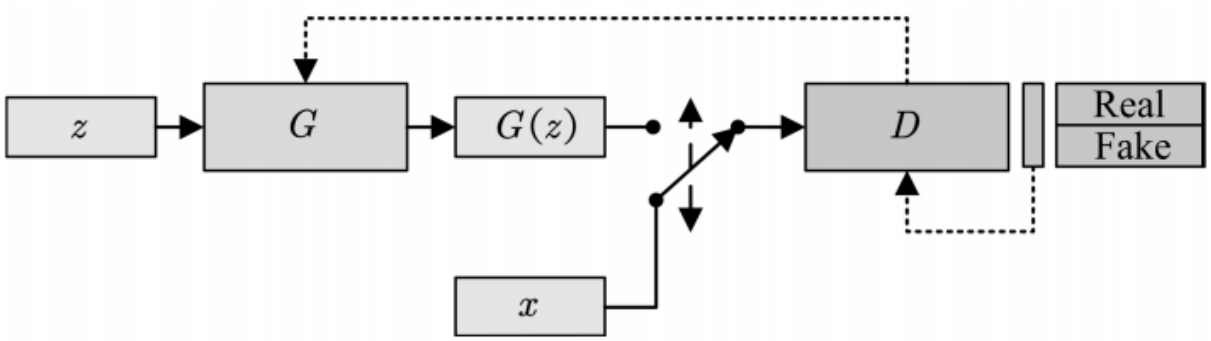
计算损失并优化模型
  计算判别器损失，并优化判别器
  计算生成器损失，并优化生成器

训练过程

GAN

生成式对抗网络

GAN框架



GAN工作原理

判别器(Discriminator)：区分真实(real)数据和虚假(fake)数据。对于真实数据，尽可能给出高的评分1；对于虚假数据，尽可能给出低个评分0.

生成器(Generator)：生成虚假数据，使得判别器D能够尽可能给出高的评分1。

GAN目标函数

$$\min_G \max_D = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

GAN

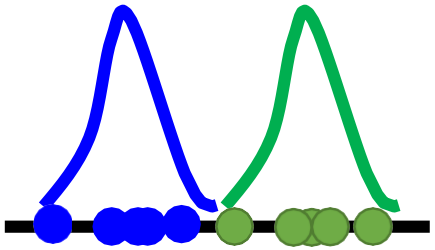
KL散度和JS散度

KL散度：一种衡量两个概率分布的匹配程度的指标

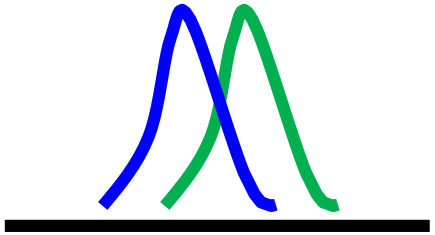
$$KL(P_1||P_2) = \mathbb{E}_{x\sim P_1}[\log \frac{P_1}{P_2}]$$

对于**连续**样本 $KL(P_1||P_2) = \int_x P_1(x) log \frac{P_1(x)}{P_2(x)} dx$

对于**离散**样本 $KL(P_1||P_2) = \sum_x P_1(x) log \frac{P_1(x)}{P_2(x)}$



数据分布：可以简单理解为点的密度



数据分布更近

当且$P_1=P_2$的时候，KL散度为零

KL散度具有**非负性（吉布斯不等式）**：

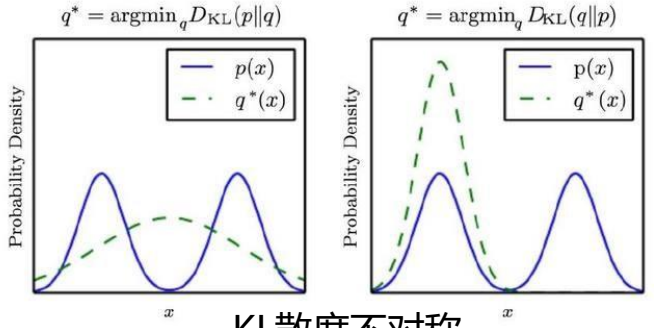
已知 $log(x) \leq x - 1$

$$-KL(P_1||P_2) = \sum_x P_1(x) log \frac{P_2(x)}{P_1(x)} \leq \sum_x P_1(x)(\frac{P_2(x)}{P_1(x)} - 1) = \sum_x P_2(x) - \sum_x P_1(x) = 0$$

JS散度：

$$JS(P_1||P_2) = \frac{1}{2} KL(P_1||\frac{P_1+P_2}{2}) + \frac{1}{2} KL(P_2||\frac{P_1+P_2}{2})$$

JS散度具有 非负性、以及对称性



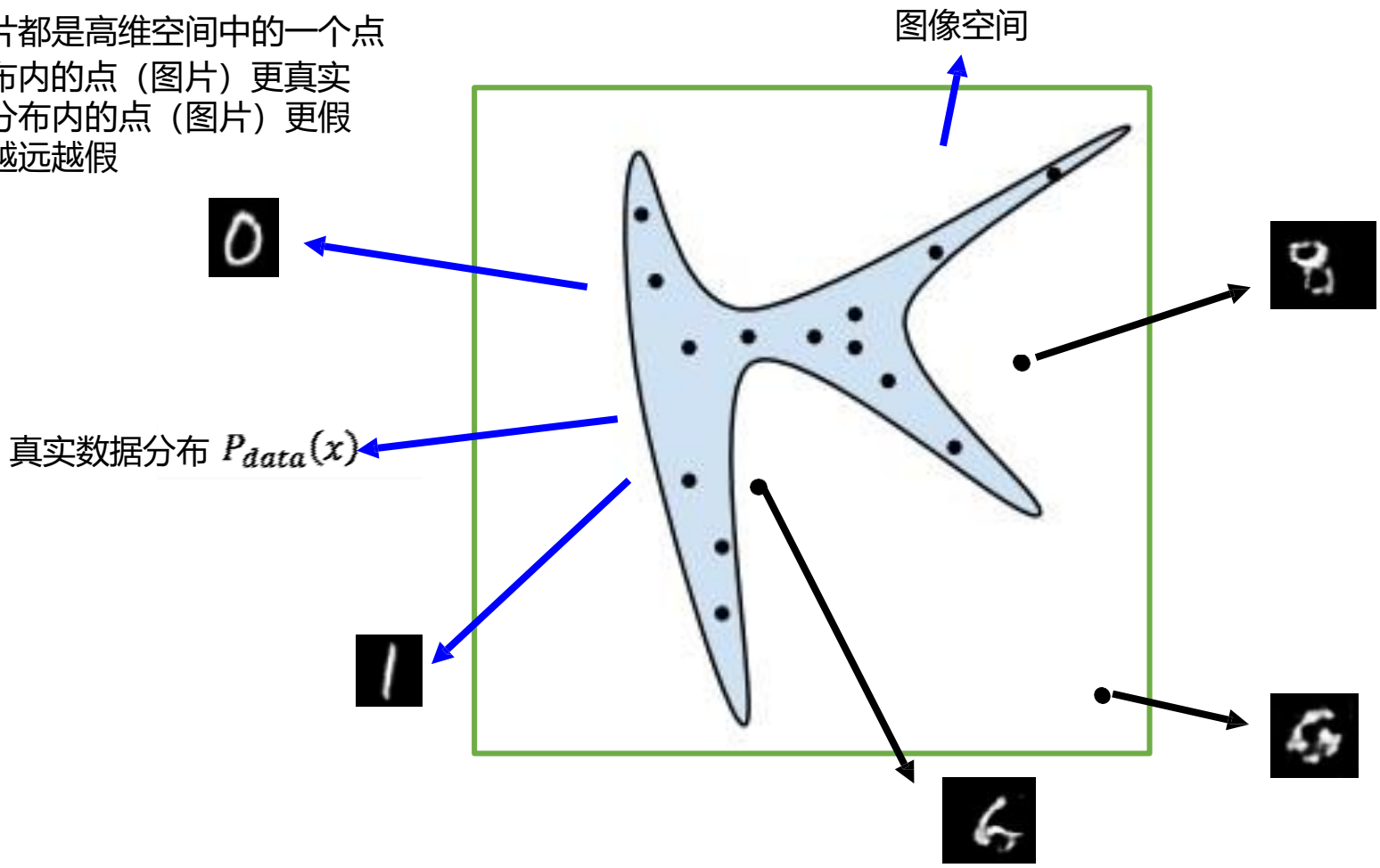
KL散度不对称

GAN

数据分布

生成模型其实就是在学习（找到）数据的分布

每一个图片都是高维空间中的一个点
在真实分布内的点（图片）更真实
不在真实分布内的点（图片）更假
而且距离越远越假



GAN

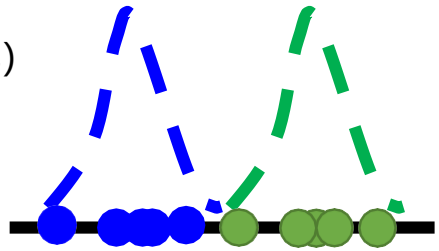
极大似然估计

我们有一个真实数据分布$P_{data}(x)$（不知道他的表达形式，但是可以从中采样）

我们还有一个由参数θ定义的数据分布$P_G(x,\theta)$

我们的目的是，找到这样的参数，使得$P_G(x,\theta)$和$P_{data}(x)$接近

例如$P_G(x,\theta)$是高斯混合模型，那么θ就是均值和方差



$$N(x|u,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} exp[-\frac{1}{2\sigma^2}(x-u)^2]$$

极大似然估计

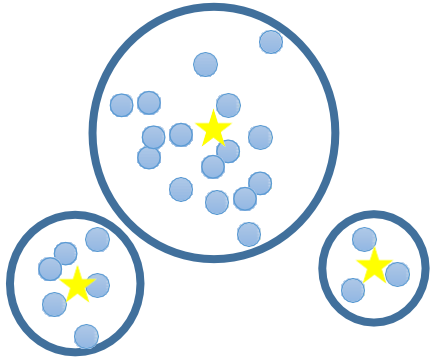
从数据集中采样m的真实样本$\{x^1, x^2, ..., x^m\}$

我们可以计算$P_G(x^i,\theta)$

$$L = \prod_{i=1}^{m} P_G(x^i;\theta)$$

通过最大化L，找到最优的θ*

$P_G(x,\theta^*)$就是我们学习到的数据分布



GAN

极大似然估计=最小化KL

$$\theta^* = arg\max_\theta \prod_{i=1}^{m} P_G(x^i;\theta) = arg\max_\theta log \prod_{i=1}^{m} P_G(x^i;\theta)$$

$$= arg\max_\theta \sum_{i=1}^{m} logP_G(x^i;\theta)$$

$$\approx arg\max_\theta E_{x\sim P_{data}}[logP_G(x;\theta)] = arg\max_\theta \int_x P_{data}(x)logP_G(x;\theta)d$$

$$= arg\max_\theta \int_x P_{data}(x)logP_G(x;\theta)d - \int_x P_{data}(x)logP_{data}(x)dx$$

$$= arg\min_\theta KL(P_{data}||P_G)$$

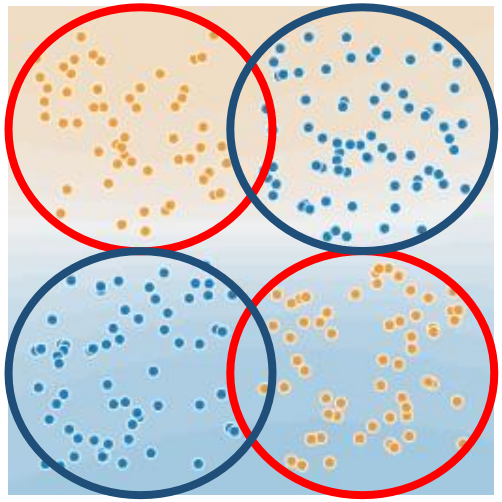
极大似然估计 等价于 **最小化生成数据分布和真实分布的KL散度。**
其中KL散度是两个分布的距离度量。

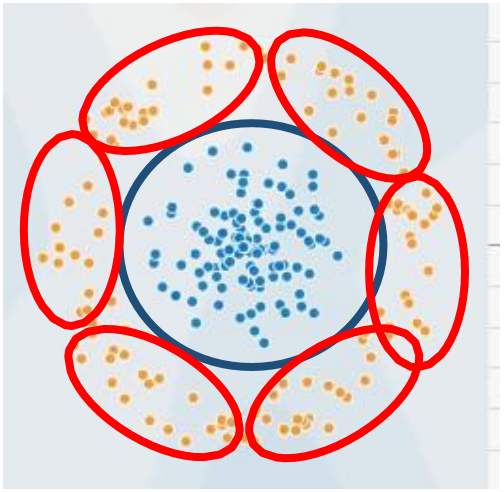
很多数据分布并不符合高斯分布的假设。

我们需要一个更通用的生成模型

GAN

极大似然估计=最小化KL





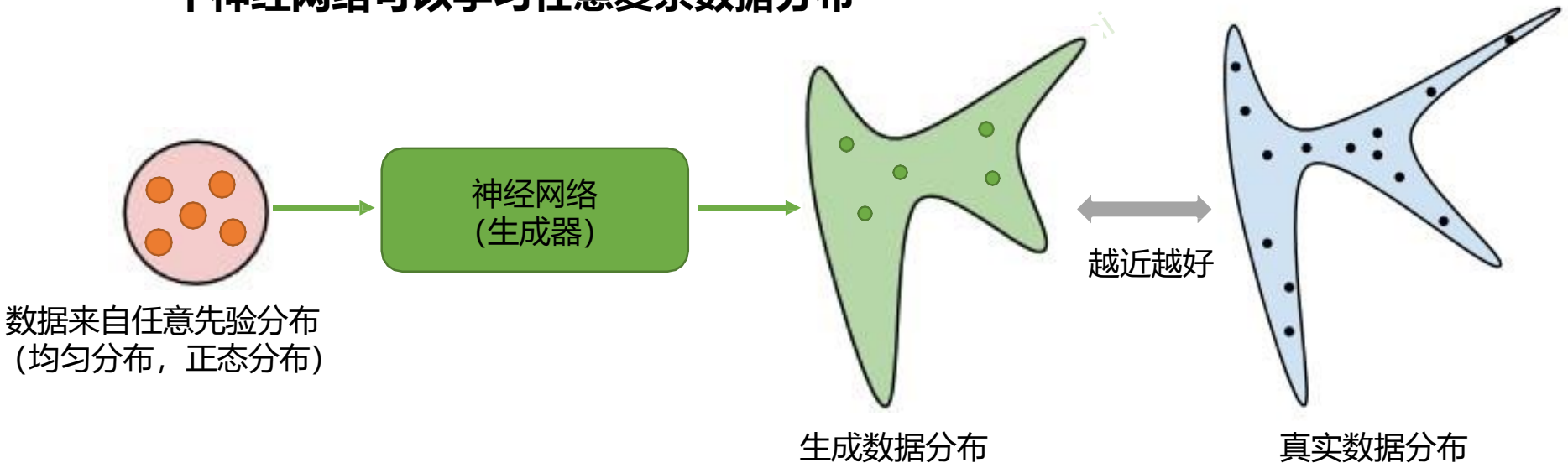
很多情况下，假设数据符合高斯分布是不合理的，数据分布是无法用公式显示的写出来的
因此用高斯模型去拟合数据分布，生成的结果总是不令人们以

我们需要一个更通用的生成模型，可以拟合任意数据分布

GAN

生成器

一个神经网络可以学习任意复杂数据分布



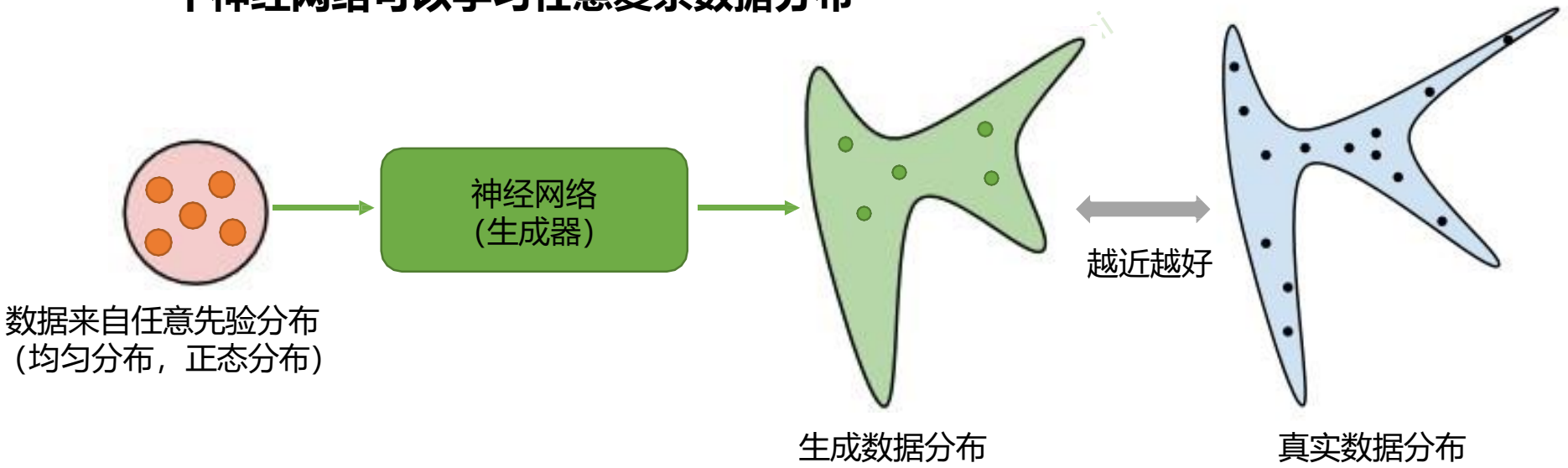
通过最小化生成数据分布P_G 和 真实数据分布P_{data} 的KL散度，就可以求出最优的生成器

$$G^* = arg\min_G Div(P_{data}||P_G)$$

GAN

生成器

一个神经网络可以学习任意复杂数据分布



通过最小化生成数据分布P_G 和 真实数据分布P_{data} 的KL散度，就可以求出最优的生成器

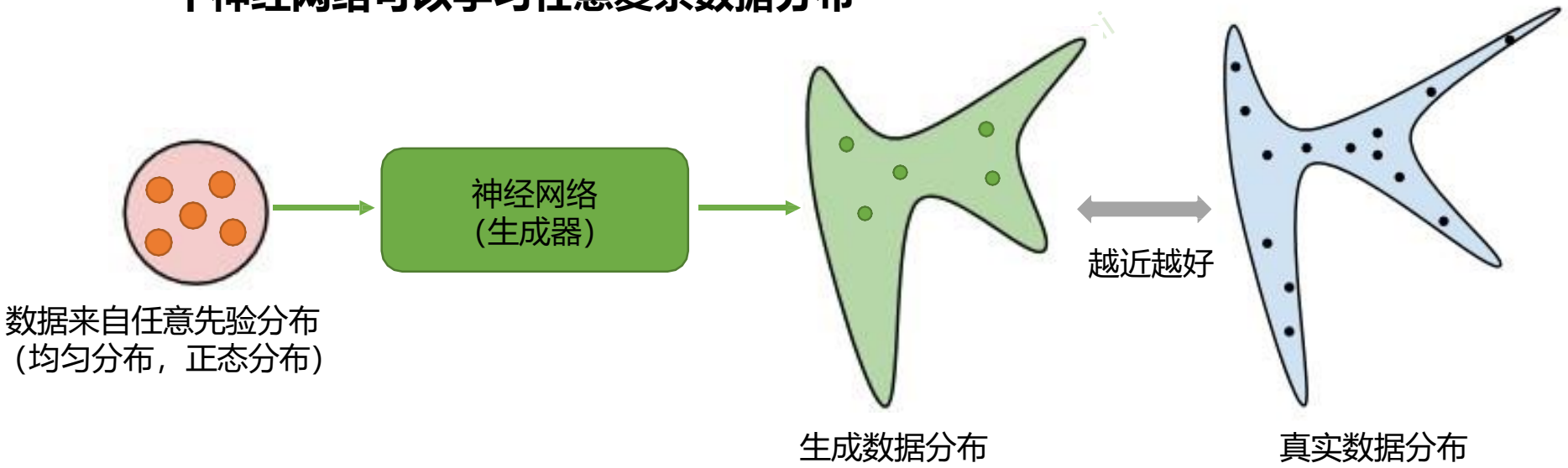
$$G^* = arg\min_G Div(P_{data}||P_G)$$

但是，生成数据和真实数据分布的形式我们不知道，
无法计算散度，无法优化生成器！

GAN

生成器

一个神经网络可以学习任意复杂数据分布



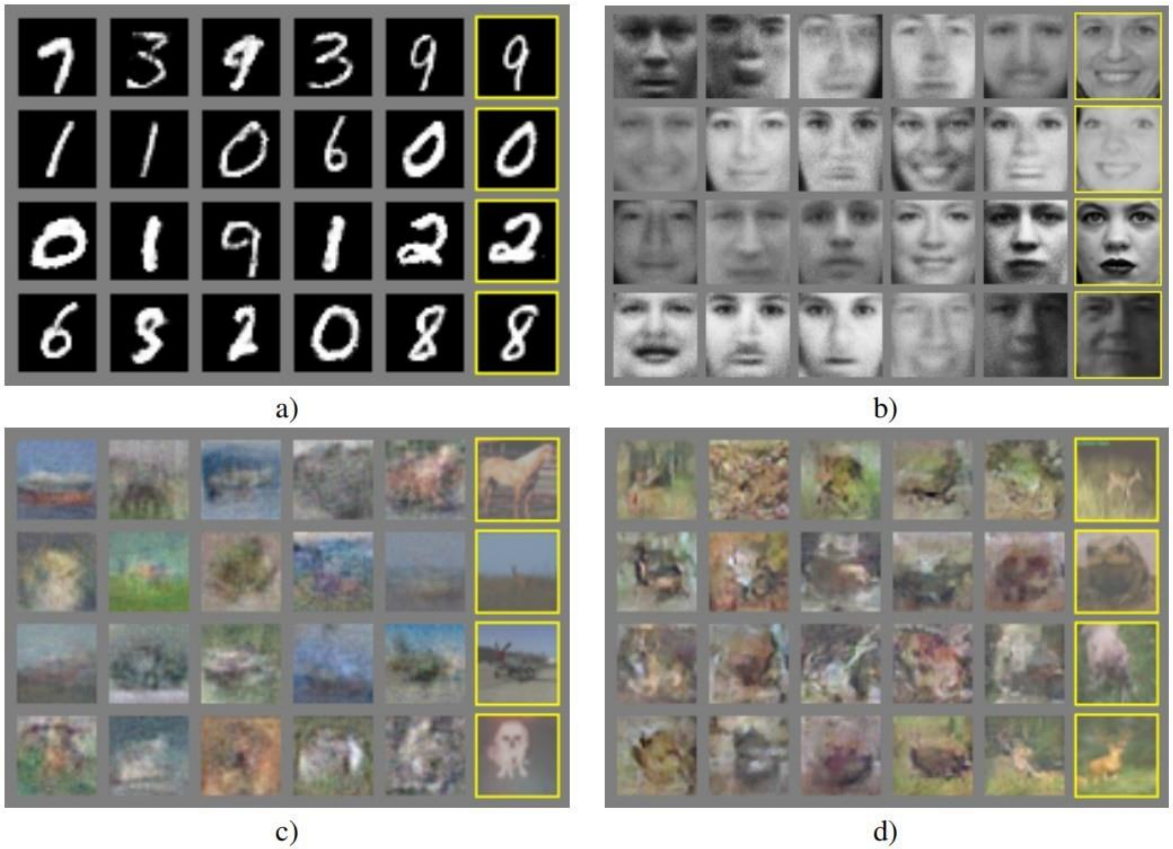
通过最小化生成数据分布P_G 和 真实数据分布P_{data} 的KL散度，就可以求出最优的生成器

$$G^* = arg\min_G Div(P_{data}||P_G)$$

GAN通过对抗训练，间接计算出JS散度，使得模型可以优化！

GAN

实验结果



生成图片比较逼真

生成图片并非简单记忆数据库中的图片

目录

CONTENTS

cGAN

条件生成

我们有一堆真实图片

用GAN学习之后可以生成逼真的图片

$$\begin{bmatrix}0.1\\0.3\\0.4\\0.2\end{bmatrix}\ \begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\ \begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\ \begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}$$

生成器

实际应用中的问题：无法控制生成的类别！

cGAN

条件生成

我们有一堆真实图片

用GAN学习之后可以生成逼真的图片

$$\begin{bmatrix}0.1\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}$$

生成器

为什么要控制生成数据的类别呢?

很多有意义的任务都需要学习条件生成模型!

条件 ✓ X

条件 ✓ X

条件生成非常重要!!

cGAN

条件生成

首先要有类别标签

cGAN

条件生成

首先要有类别标签

除了输入随机向量，还要数据类别

$$\begin{bmatrix}0.1\\0.3\\0.4\\0.2\end{bmatrix}\ \begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\ \begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\ \begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}$$

狐狸　气泡　蝴蝶　狗

→ 生成器 →

cGAN

条件生成

首先要有类别标签

除了输入随机向量，还要数据类别

$$\begin{bmatrix}0.1\\0.3\\0.4\\0.2\end{bmatrix}\ \begin{bmatrix}0.2\\0.3\\0.4\\0.2\end{bmatrix}\ \begin{bmatrix}0.3\\0.3\\0.4\\0.2\end{bmatrix}\ \begin{bmatrix}0.4\\0.3\\0.4\\0.2\end{bmatrix}$$

狐狸　气泡　蝴蝶　狗

→ 生成器 →

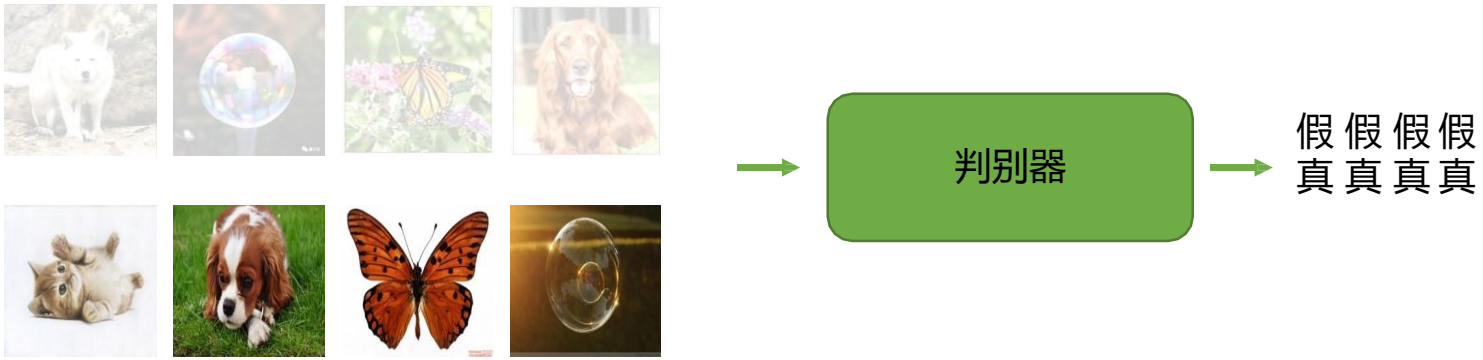
很难刚好生成对应类别的数据！

很可能类别完全不对应！

cGAN

条件生成

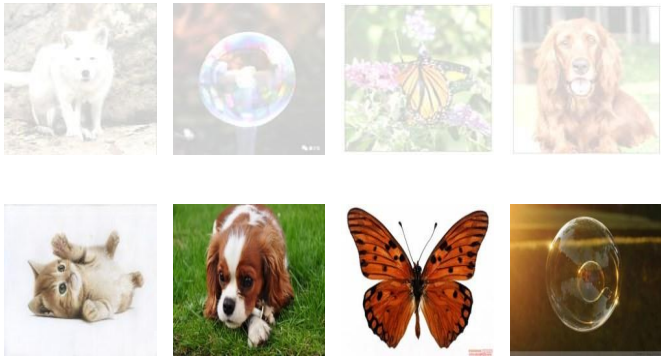
判别器：区分真假样本



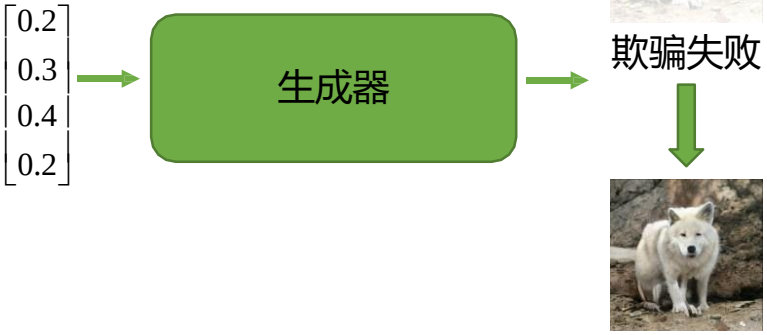
cGAN

条件生成

判别器：区分真假样本



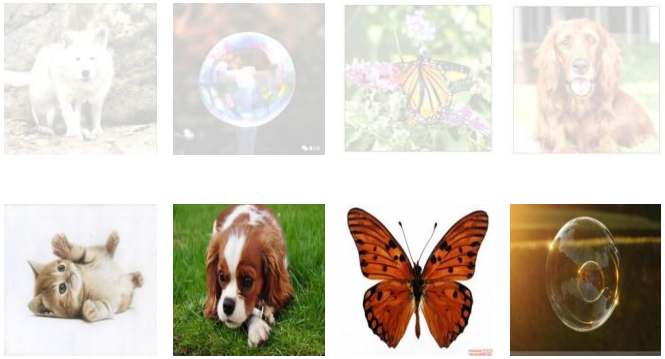
生成器：**欺骗**判别器



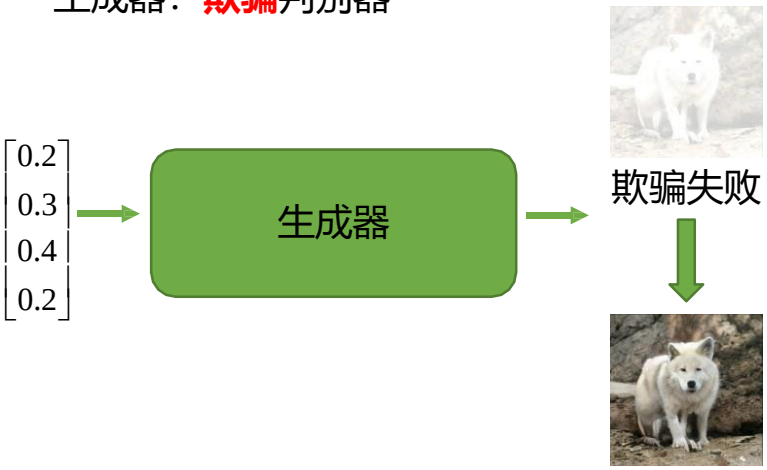
cGAN

条件生成

判别器：区分真假样本



生成器：**欺骗**判别器

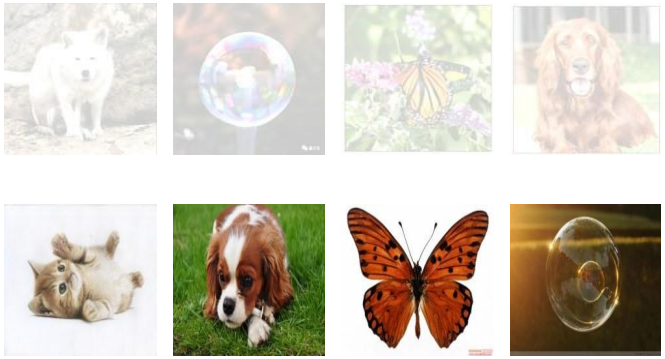


条件生成器：**欺骗**判别器

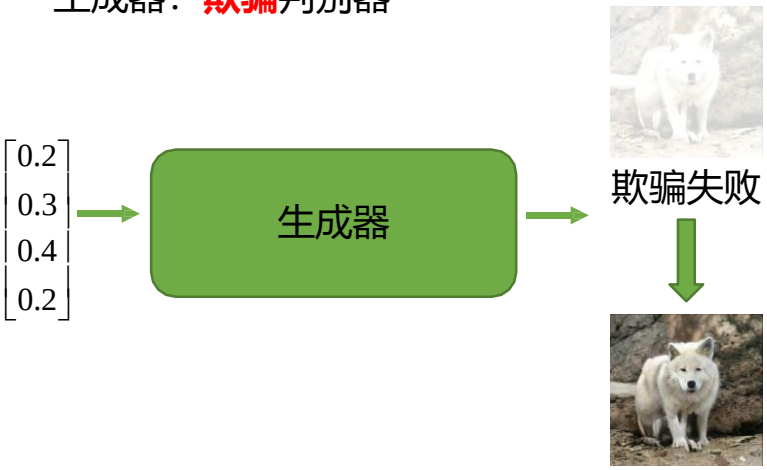
cGAN

条件生成

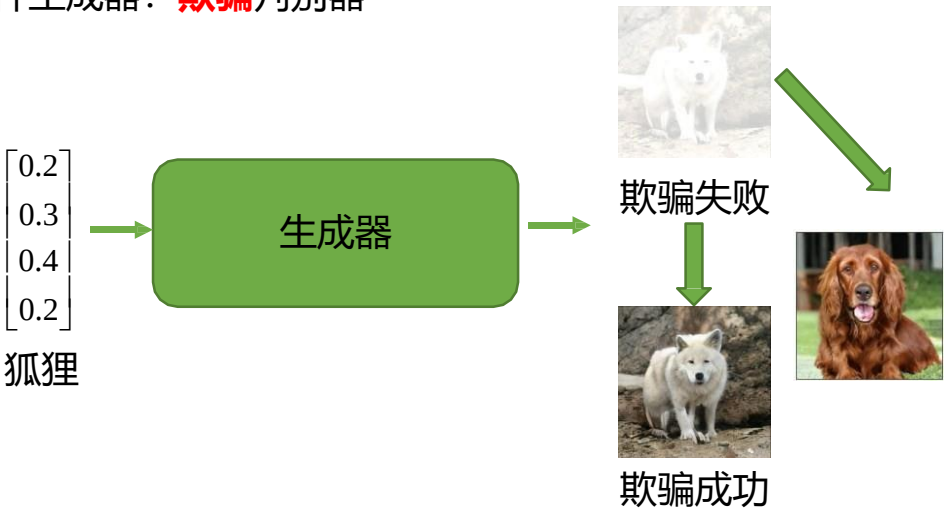
判别器：区分真假样本



生成器：**欺骗**判别器



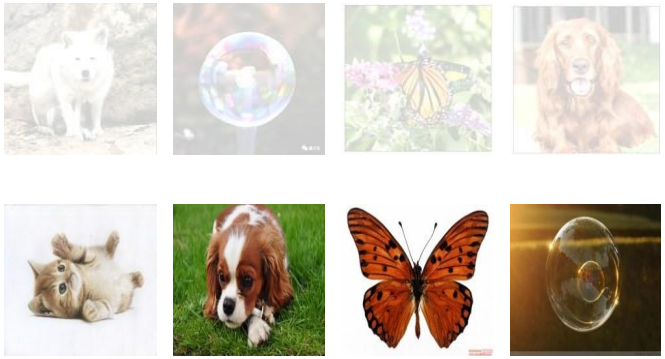
条件生成器：**欺骗**判别器



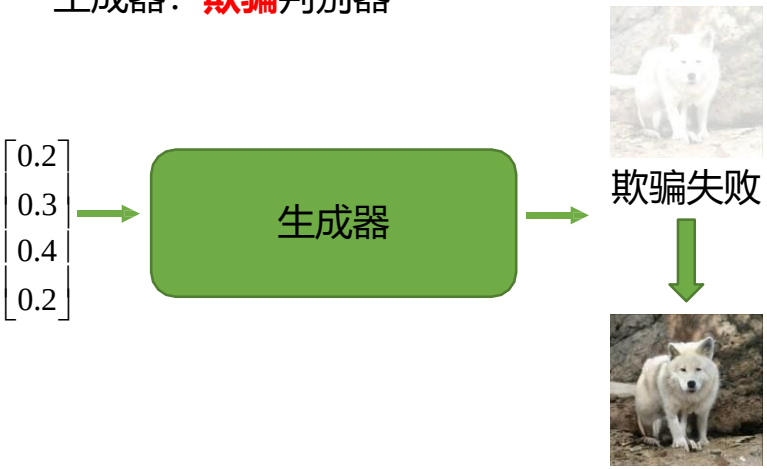
cGAN

条件生成

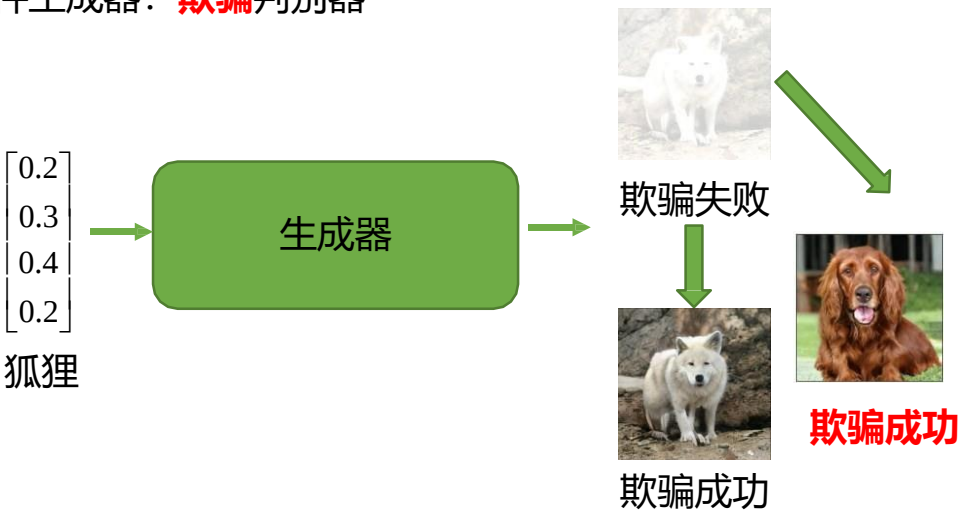
判别器：区分真假样本



生成器：**欺骗**判别器



条件生成器：**欺骗**判别器



大家请思考，问题出在哪呢?

cGAN

改进判别器

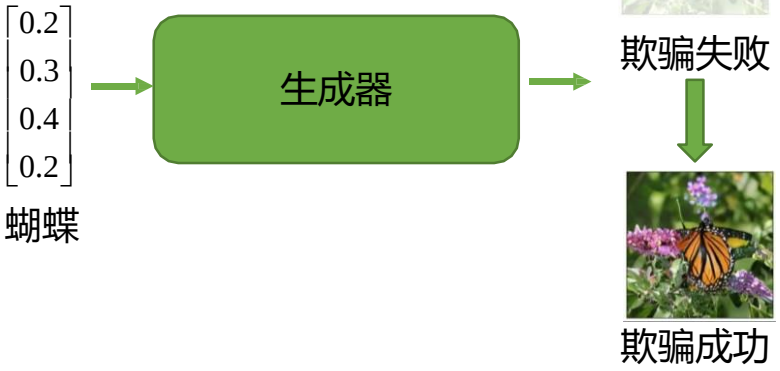
判别器：区分真假样本

cGAN

改进判别器

判别器：区分真假样本

生成器：**欺骗**判别器

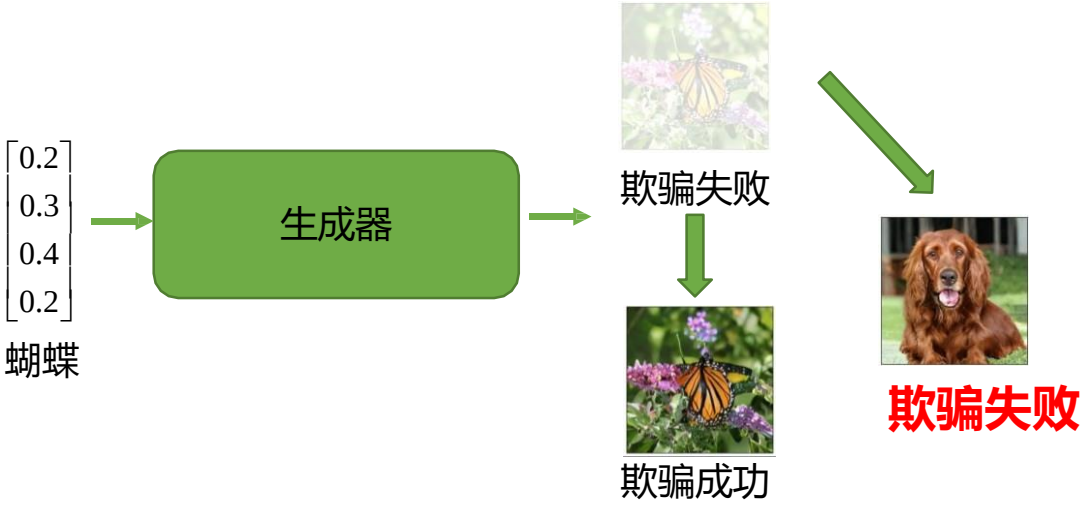


cGAN

改进判别器

判别器：区分真假样本

生成器：**欺骗**判别器

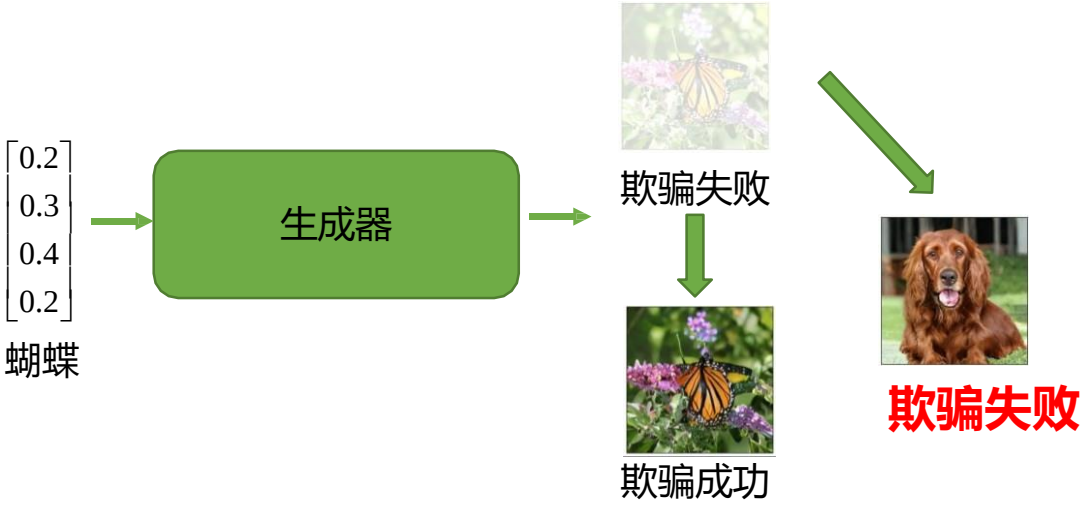


cGAN

改进判别器

判别器：区分真假样本

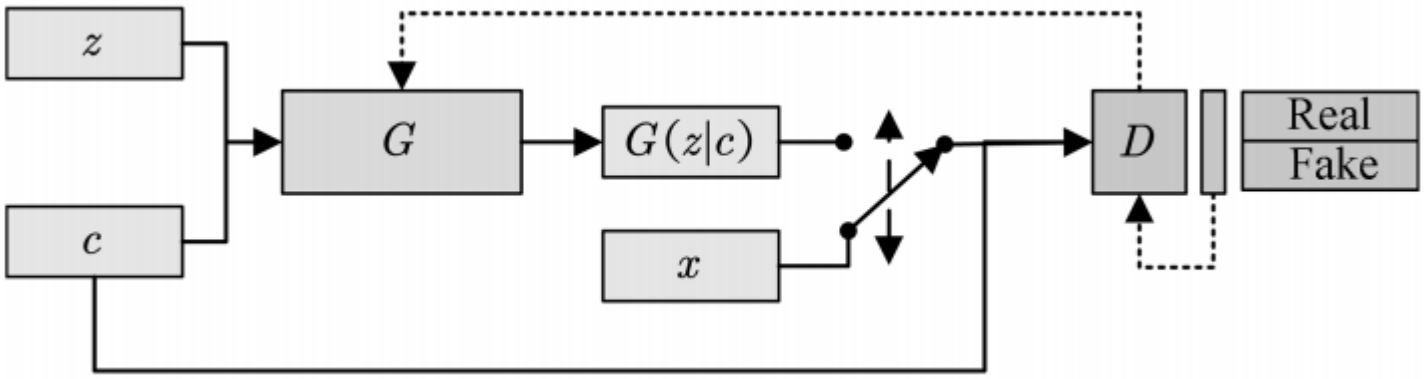
生成器：**欺骗**判别器



cGAN

cGAN

网络结构



Improved CGAN 目标函数：

$$\min_G \max_D V(G,D) = E_{(x,y)\sim p_{data}}[logD(x|y) + log(1 - D(x|\hat{y}))] + E_{z\sim p_z}[log(1 - D(G(z|y)))]$$

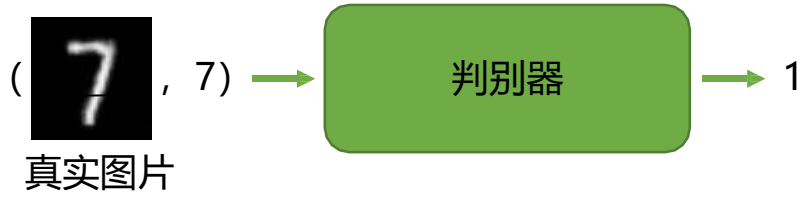
CGAN 目标函数：

$$\min_G \max_D V(G,D) = E_{(x,y)\sim p_{data}}[logD(x|y)] + E_{z\sim p_z}[log(1 - D(G(z|y)))]$$

GAN

cGAN

优化判别器

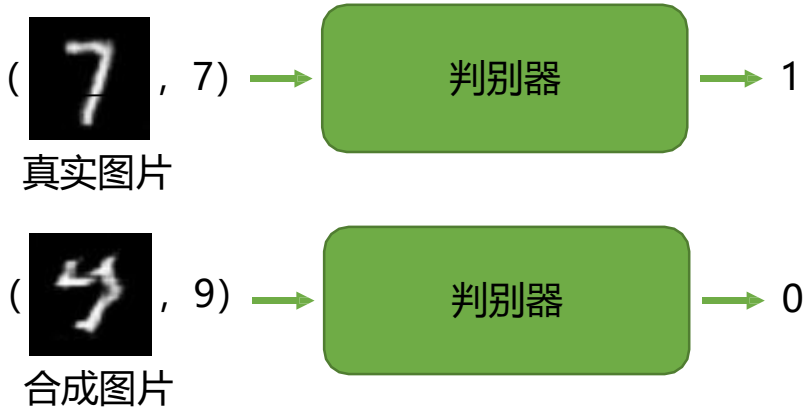


优化生成器

GAN

cGAN

优化判别器

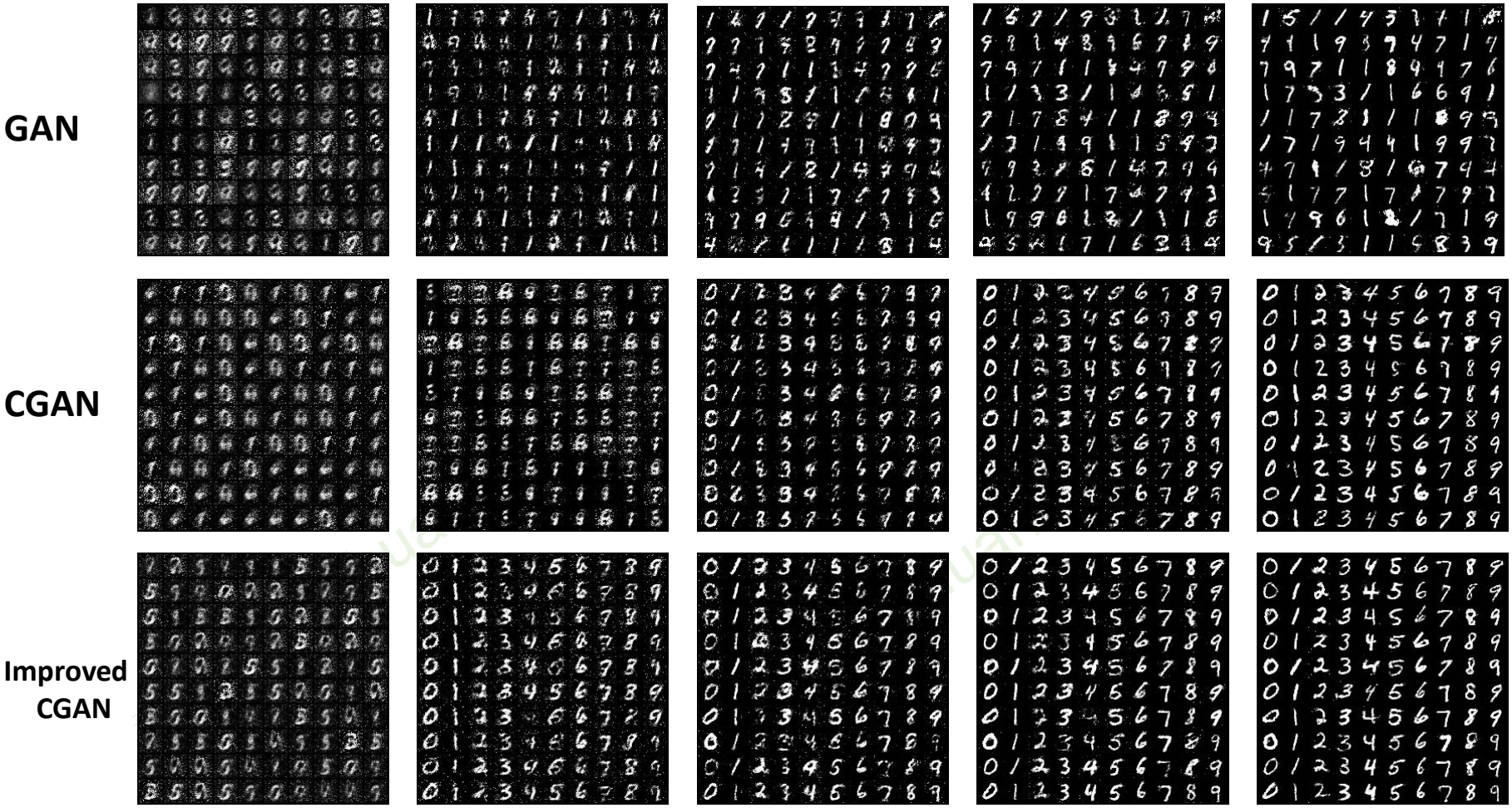


优化生成器

cGAN

演示代码

实验结果



cGAN

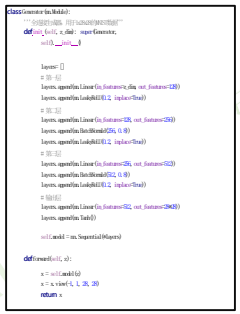
Pytorch实现cGAN

准备工作

定义网络结构：判别器和生成器的网络结构
定义目标函数：根据预测标签和真实标签，计算损失
定义优化器：优化器，自动计算梯度并优化网络权重
数据加载：加载数据集

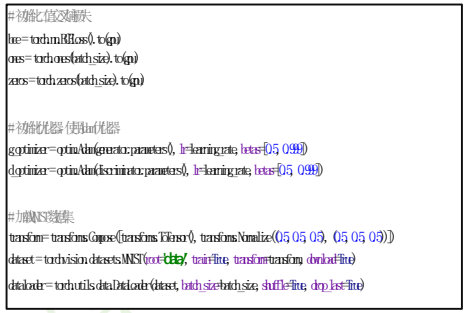
判别器GAN

判别器cGAN



生成器GAN

生成器cGAN



其他准备工作

训练直到收敛

加载数据
- 提取一个batch的真实数据
- 生成一个bathc的虚假数据

计算损失并优化模型
- 计算判别器损失，并优化判别器
- 计算生成器损失，并优化生成器

训练过程

目录

CONTENTS

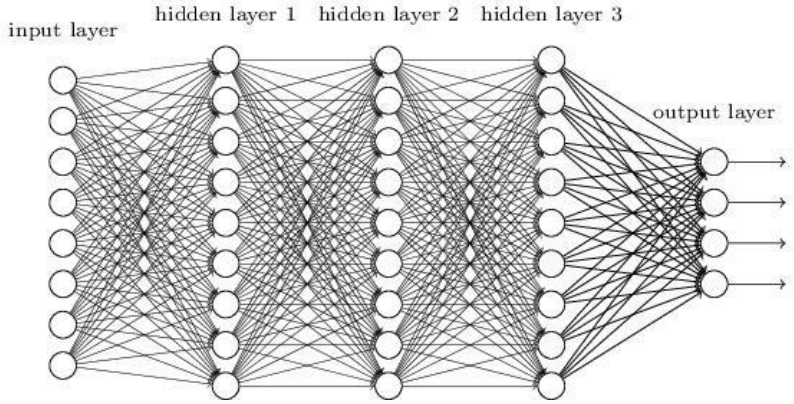
DCGAN

DCGAN

原始GAN，使用全连接网络作为判别器和生成器

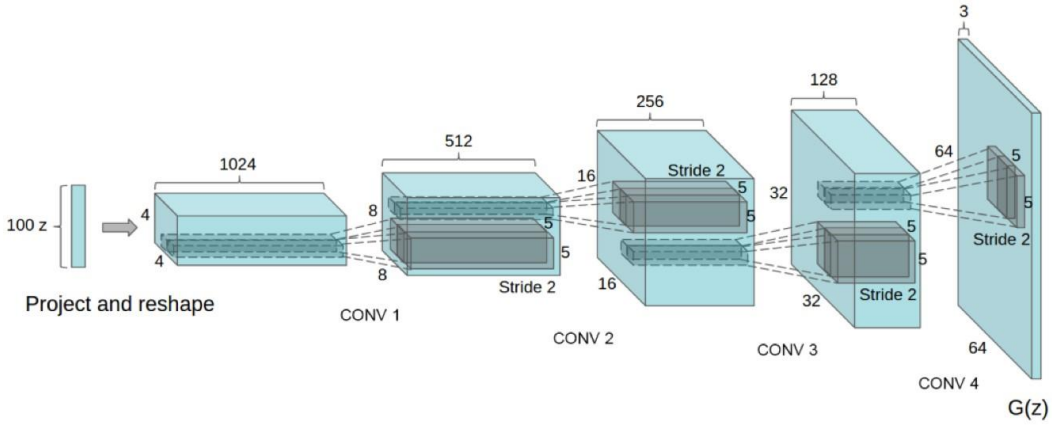
不利于建模图像信息

参数量大，需要大量的计算资源，难以优化



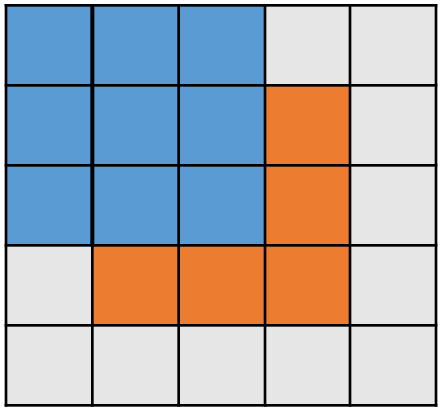
DCGAN，使用卷积神经网络作为判别器和生成器

通过大量的工程实践，经验性地提出一系列的网络结构和优化策略，来有效的建模图像数据

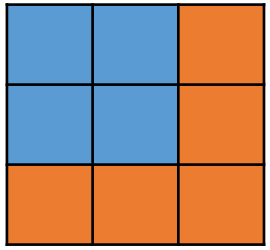


DCGAN

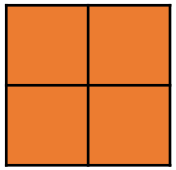
判别器：滑动卷积



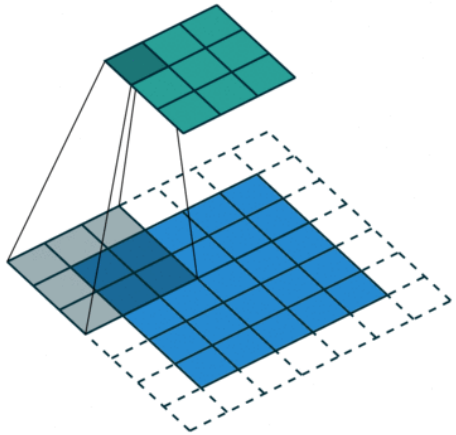
灰色：Padding
橘色：Feature Map
蓝色：Kernel



橘色：Feature Map
蓝色：Pooling



橘色：Feature Map

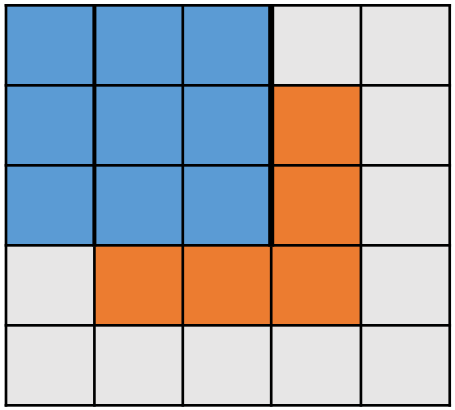


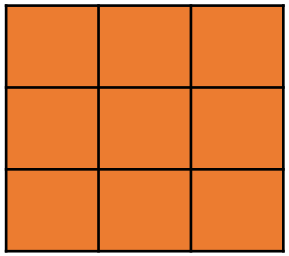
通过Pooling下采样，Pooling是不可学习的，
这可能造成GAN训练困难

使用滑动卷积（Stride Convolution）
所有参数都可学习
实验发现效果更好

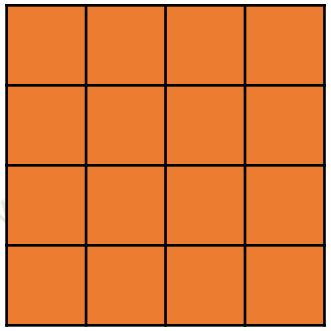
DCGAN

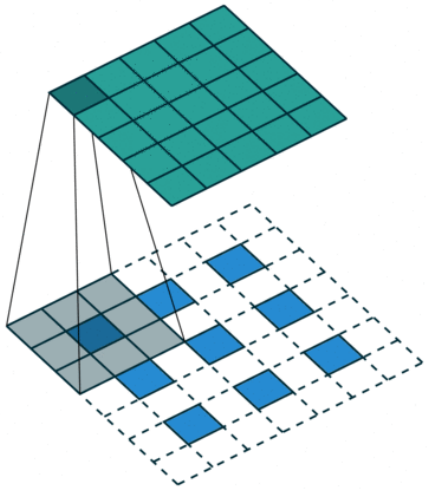
生成器：滑动反卷积





通过插值
上采样





比如：线性插值

灰色：Padding
橘色：Feature Map
蓝色：Kernel

通过插值法上采样，
插值方法是固定的，不可学习的，
这可能给训练造成困难

使用滑动反卷积所有参数都可学习
实验发现效果更好

DCGAN

批归一化

IID独立同分布假设，就是假设训练数据和测试数据是满足相同分布的，这 是通过训练数据获得的模型能够在测试集获得好的效果的一个基本保障

BatchNorm就是在深度神经网络训练过程中使得每一层神经网络的输入保持相同分布的

优势：

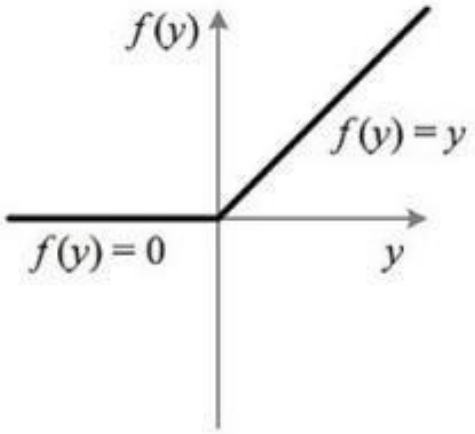
加速神经网络收敛

减小神经网络参数对于初始化的依赖

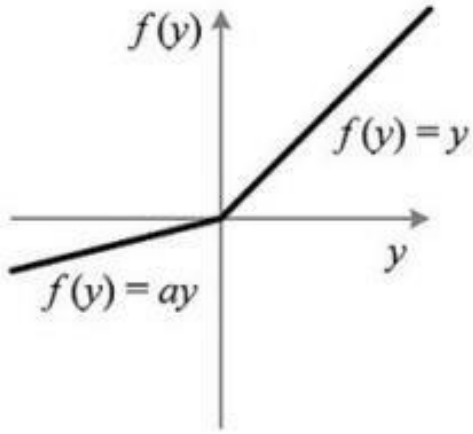
实验发现效果更好~

DCGAN

激活函数



$$ReLU(x) = \begin{cases} x & if\, x > 0 \\ 0 & if\, x \le 0 \end{cases}$$



$$PReLU(x_i) = \begin{cases} x_i & if\, x_i > 0 \\ a_i x_i & if\, x_i \le 0 \end{cases}$$

i 表示不同的通道

Relu可以缓解梯度消失
PRelu是Relu的改进版本，当X<0时仍然可以提供梯度

DCGAN

DCGAN

网络结构（判别器）

- 使用滑动卷积（strided convolution）
- 除了输入层，全部使用批归一化
- 使用 Leaky ReLu 激活函数
- 除了最后一层，不使用全连接层

网络结构（生成器）

- 使用 fractional strided convolution
- 除了输出层，全部使用批归一化
- 使用 ReLu 激活函数，最后一层使用tanh激活函数

滑动卷积、滑动翻卷机：使得判别器和生成器可以学习自己的上采样和下采样策略

批归一化（Batch Normalization）：训练更稳定

Tanh激活函数：更快的学习到真实数据的颜色空间

训练策略

- 数据预处理：所有输入数据归一化到[-1,1]
- 激活函数：Leaky Relu的斜率设置为0.2
- 初始化：使用均值为0，标准差为0.02的正态分布初始化网络参数
- 优化器：使用Adam优化器，学习率为0.0002，beta1=0.5，beta2=0.999

DCGAN

实验结果

DCGAN

1 Epoch　2 Epoch　5 Epoch　10 Epoch　20 Epoch

GAN

1 Epoch　5 Epoch　10 Epoch　20 Epoch　50 Epoch

DCGAN

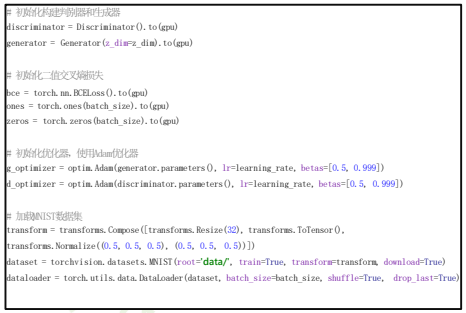
Pytorch实现DCGAN

准备工作

定义网络结构：判别器和生成器的网络结构
定义目标函数：根据预测标签和真实标签，计算损失
定义优化器：优化器，自动计算梯度并优化网络权重
数据加载：加载数据集

判别器

生成器



其他准备工作

训练直到收敛

加载数据
提取一个batch的真实数据
生成一个bathc的虚假数据

计算损失并优化模型
计算判别器损失，并优化判别器
计算生成器损失，并优化生成器

训练过程

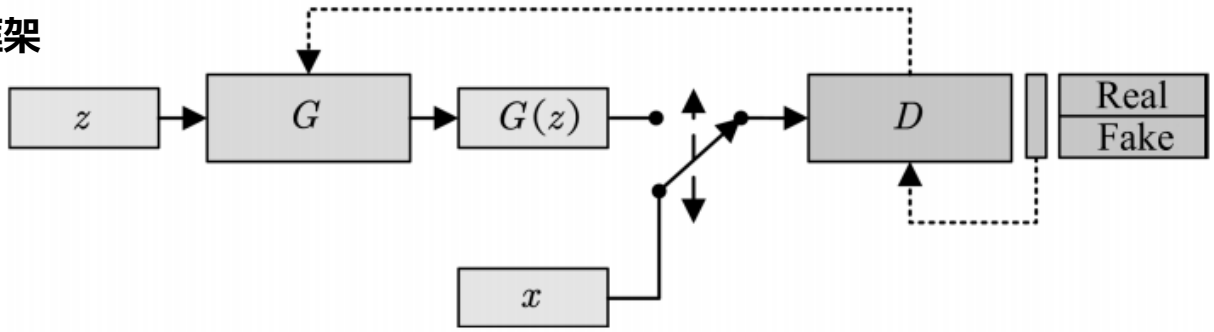
目录

CONTENTS

WGAN

回顾原始 GAN

GAN框架



生成式对抗网络的目标函数

$$\max_D V(G,D) = E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$$

$$\min_G V(G,D) = E_{z\sim p_z}[log(1-D(G(z)))]$$

等价于

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

博弈角度：判别器D 区分真假样本；生成器G 合成真实样本；（**罪犯和商贩的例子**）

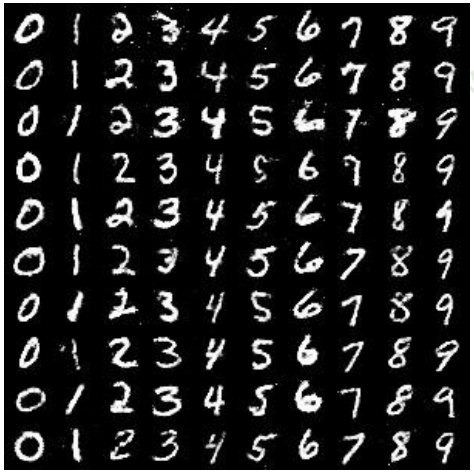
理论分析：优化GAN等于最小化JS散度。

$$D^*(x) = \frac{p_{data}(x)}{p_{data}(x)+p_g(x)} \qquad \min_G V(G,D^*) = 2JS(p_{data}||p_g) + 2log\frac{1}{2}$$

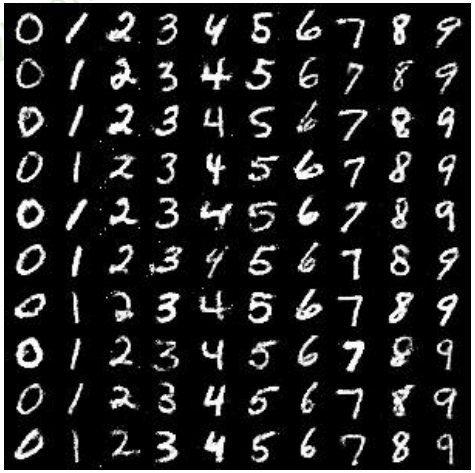
WGAN

GAN、cGAN

GAN



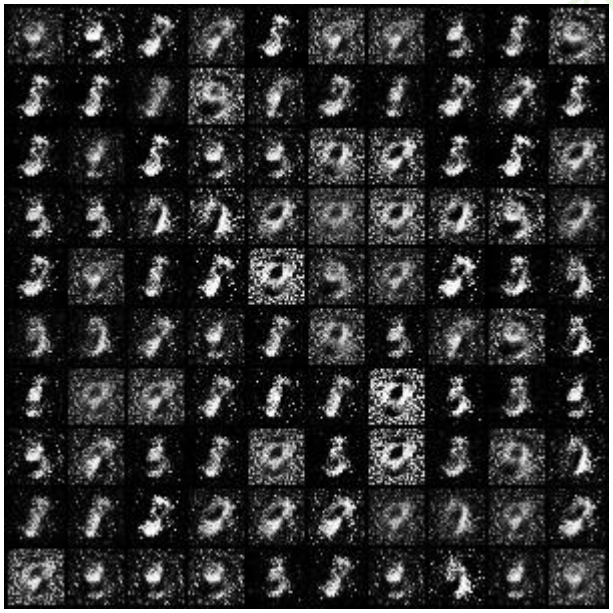
CGAN



Improved CGAN

WGAN

原始 GAN 存在的问题



训练困难：生成器无法生成想要的数据

模式崩塌：生成器无法学习到完整的数据分布

WGAN

JS散度

生成式对抗网络： $\min_G V(G, D^*) = 2JS(p_{data}||p_g) + 2log\frac{1}{2}$

JS散度： $JS(P_1||P_2) = \frac{1}{2}KL(P_1||\frac{P_1+P_2}{2}) + \frac{1}{2}KL(P_2||\frac{P_1+P_2}{2})$

$JS(P_1||P_2) = \frac{1}{2}\int_x P_1(x)log(\frac{P_1(x)}{\frac{1}{2}[P_1(x)+P_2(x)]})dx + \frac{1}{2}\int_x P_2(x)log(\frac{P_2(x)}{\frac{1}{2}[P_1(x)+P_2(x)]})dx$

分析任意一个点x对JS散度的贡献：

当 $P_1(x) = 0 \ and \ P_2(x) = 0$ **时：**

$JS(P_1||P_2)=0$, 对计算JS散度无贡献

当 $P_1(x) \neq 0 \ and \ P_2(x) = 0$ **或** $P_1(x) = 0 \ and \ P_2(x) \neq 0$ **时：**

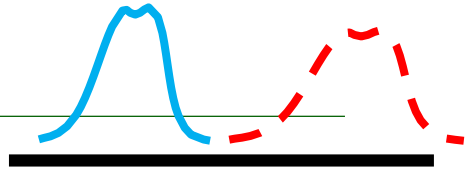
贡献等于常数 log2，但是梯度等于零

当 $P_1(x) \neq 0 \ and \ P_2(x) \neq 0$ **时：**

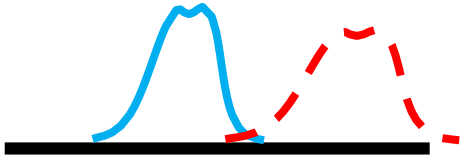
对计算JS散度有贡献且不为常数，因此梯度不为零。

WGAN

不重叠或重叠部分可忽略概率



$P_1(x) = 0 \ and \ P_2(x) \neq 0$
$P_1(x) \neq 0 \ and \ P_2(x) = 0$



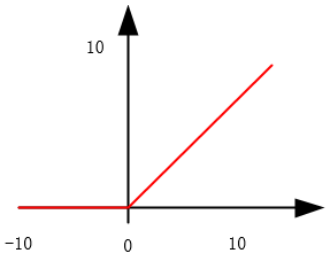
$P_1(x) \neq 0 \ and \ P_2(x) \neq 0$

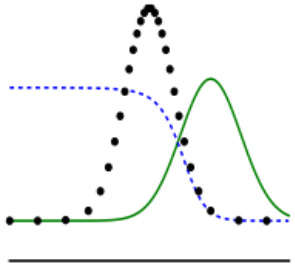
分析$P_{data}(x)$与$P_g(x)$ 不重叠或重叠部分可忽略的可能性有多大?

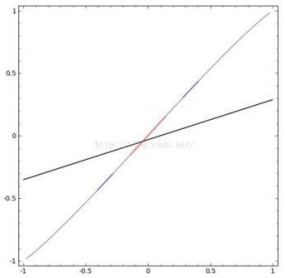
答：非常大!

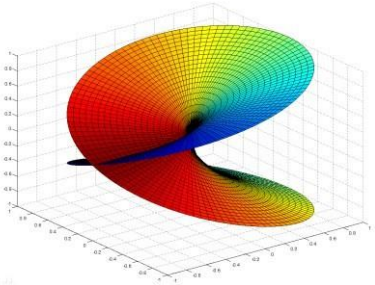
原因：$P_{data}(x)$与$P_g(x)$的支撑集（support）是**高维空间中的低维流形**（manifold）时，$P_{data}(x)$与$P_g(x)$**重叠部分测度（measure）为0的概率为1**。

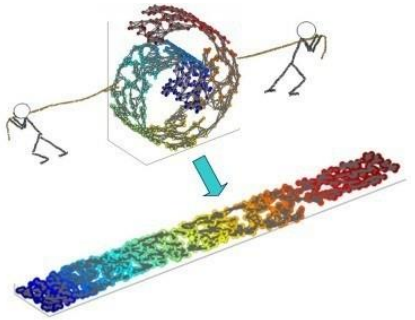
支撑集（support）：函数的非零部分子集。ReLU函数的支撑集就是零到正无穷，概率分布的支撑集就是所有概率密度非零部分的集合









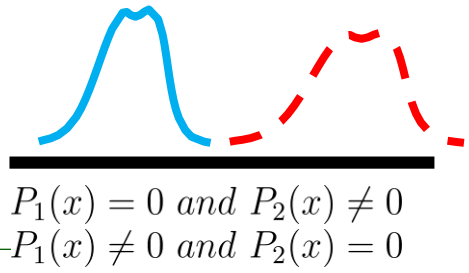


流形（manifold）：是高维空间中曲线、曲面概念的拓广。二维空间中的曲线是一个一维流形；三维空间中的一个曲面是一个二维流形。因为它的本质维度（intrinsic dimension）只有2，一个点在这个二维流形上移动只有两个方向的自由度。

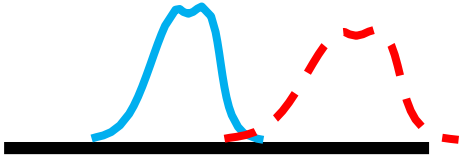
测度（measure）：高维空间中长度、面积、体积概念的拓广。

WGAN

不重叠或重叠部分可忽略概率

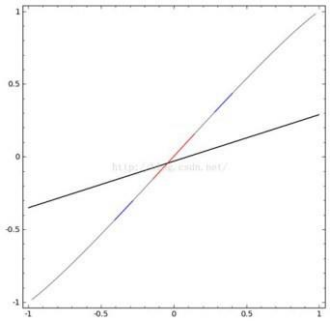


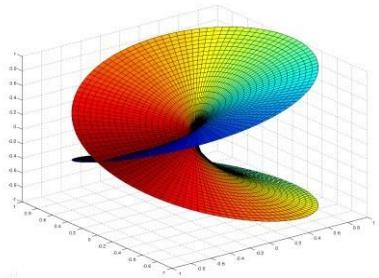
答2：$P_1(x)$与$P_2(x)$的支撑集（support）是**高维空间中的低维流形**（manifold）时，$P_1(x)$与$P_2(x)$**重叠部分测度（measure）为0的概率为1**。

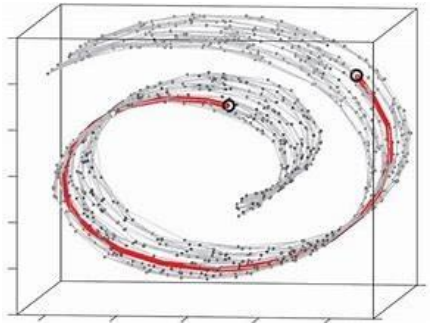


P_{data}与P_g重叠部分测度为0的概率为1：成立！

“撑不满”就会导致真实分布与生成分布难以“碰到面”，这很容易在二维空间中理解。一方面，二维平面中随机取两条曲线，它们之间刚好存在重叠线段的概率为0。另一方面，虽然它们很大可能会存在交叉点，但是相比于两条曲线而言，交叉点比曲线低一个维度，长度（测度）为0，可以忽略。







当P_{data}与P_g的支撑集是高维空间中的低维流形时：成立！

如果生成图片大小是32，那么它在一个32*32=1024维的高维空间中。生成图片的输入一般来自低维（比如100）随机噪声，即使经过非线性映射，它的维度依然被限制在100维。所以生成图片是高维（1024）空间中的低维（100）流形!

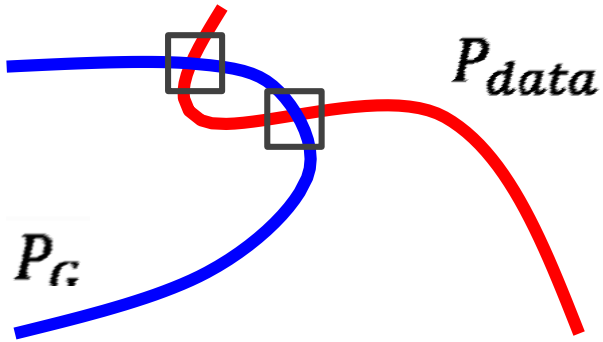
WGAN

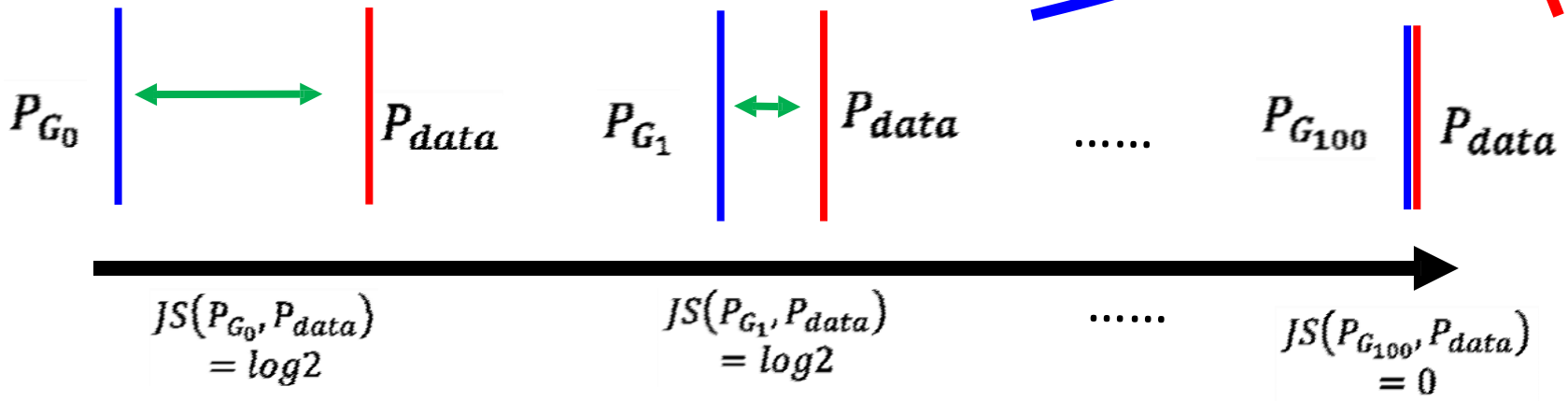
总结JS散度

GAN：真实数据分布P_{data}和生成数据分布P_G是高维空间中的维度流形，它们重叠的区域可以忽略不计

能够提供梯度信息的数据可以忽略不计！

GAN优化困难！





结论：

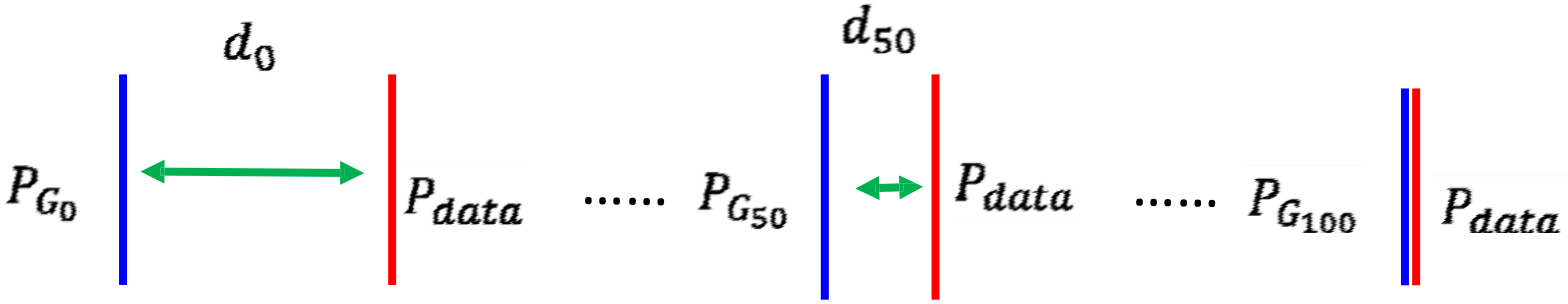
P_G 和 P_{data} 是高维空间的低维流形。判别器（近似）最优，最小化生成器的目标函数等价于最小化真实分布P_{data}与生成分布P_g与之间的JS散度，而由于**真实分布P_{data}与生成分布P_g几乎不可能有不可忽略的重叠**，所以无论它们相距多远，JS散度都是常数 。最终**导致生成器的梯度（近似）为0**，造成梯度消失，造成**GAN难以优化。**

WGAN

Wasserstein 距离

KL散度、JS散度：当两个分布距离较远、或重叠部分可忽略不计的时候，两个分布之间的JS散度为常数，无法反应分布的远近，也就无法进行优化。

解决方案：找到一个距离度量，即使在两个分布距离较远、或重叠部分可忽略不计的时候，也能充分反应两个分布的远近。



X $JS(P_{G_0}, P_{data}) = log2$

$JS(P_{G_{50}}, P_{data}) = log2$

$JS(P_{G_{100}}, P_{data}) = 0$

✓ $W(P_{G_0}, P_{data}) = d_0$

$W(P_{G_{50}}, P_{data}) = d_{50}$

$W(P_{G_{100}}, P_{data}) = 0$

WGAN

Wasserstein 距离

Wasserstein 距离：即使两个分布没有重叠或可忽略不计，Wasserstein距离仍然能够反映它们的远近。

$$W(P_1, P_2) = \inf_{\gamma \sim \Pi(P_1, P_2)} \mathbb{E}_{(x,y)\sim\gamma}[||x - y||]$$

$\Pi(P_1, P_2)$:P_1 和P_2 组合起来的所有可能的联合分布的集合

||x-y||：样本x和y的距离L1距离

inf: 所有可能的下界

分布	C	D
A	0	1
B	1	0

联合分布1

分布	C	D
A	1	0
B	0	1

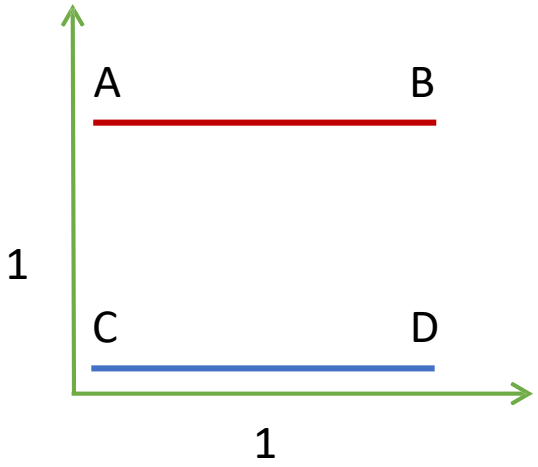
联合分布2

距离	C	D
A	1	$\sqrt{2}$
B	$\sqrt{2}$	1

距离||x-y||

$\sqrt{2}$ 2

2

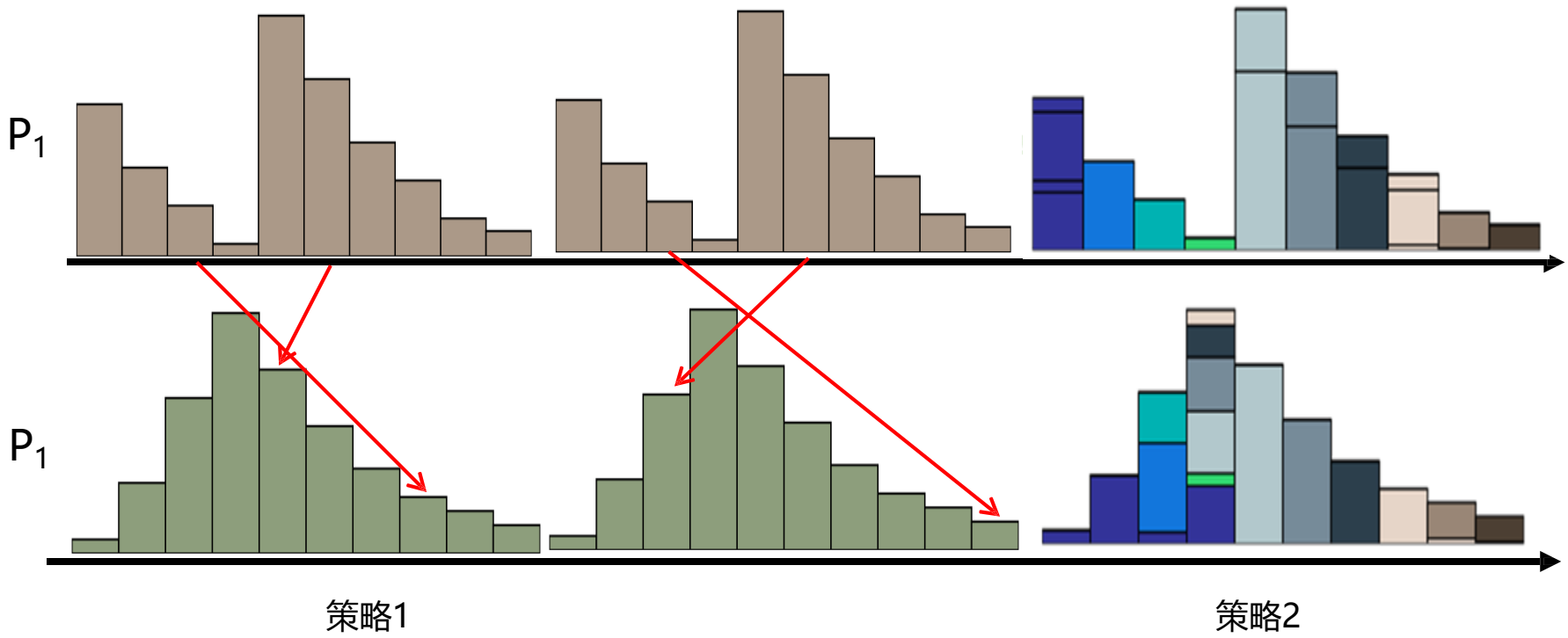


二维空间中的两个离散分布AB、CD
如何求AB和CD的Wasserstein距离?

理解：把C全部移动到A，把B全部移动到D

WGAN

Wasserstein 距离



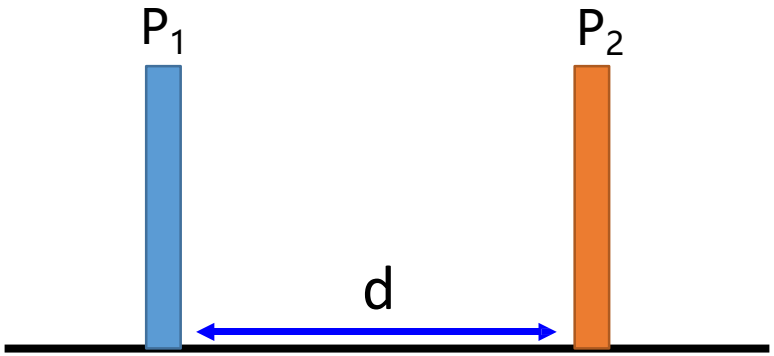
WGAN

Wasserstein 距离

Wasserstein 距离：

$$W(P_1, P_2) = \inf_{\gamma \sim \Pi(P_1, P_2)} \mathbb{E}_{(x,y)\sim\gamma}[||x - y||]$$

理解：在**最优**的**路径规划γ**下，把土堆从**P_1移动到P_2**所需的路径
Wasserstein 距离又称 Earth Mover's（推土机） 距离
可以理解为：把P_1这堆“沙土”移动到P_2处所需最短距离消耗。



两个离散分布P1、P2的Wasserstein 距离

$$W(P_1, P_2) = d$$

WGAN

Wasserstein 距离

KL散度：当两个分布重叠时为零，不重叠是为正无穷

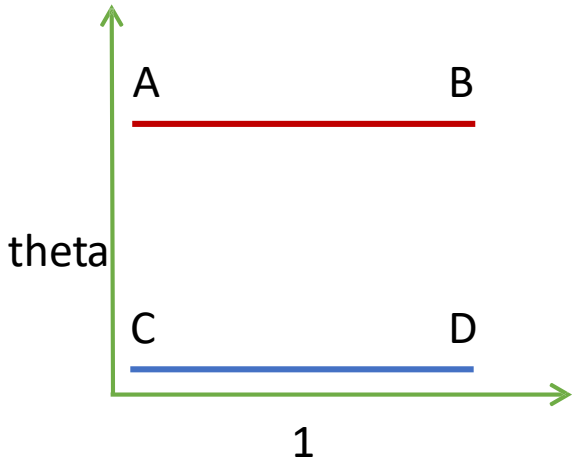
$$KL(P_1||P_2) = \mathbb{E}_{x\sim P_1}[\log \frac{P_1}{P_2}]$$

$$KL(P_1||P_2) = KL(P_1||P_2) = \begin{cases} +\infty & \text{if } \theta \neq 0 \\ 0 & \text{if } \theta = 0 \end{cases}$$

JS散度：当两个分布重叠时为零，不重叠时为常数log2

$$JS(P_1||P_2) = \frac{1}{2}KL(P_1||\frac{P_1+P_2}{2}) + \frac{1}{2}KL(P_2||\frac{P_1+P_2}{2})$$

$$JS(P_1||P_2) = \begin{cases} \log 2 & \text{if } \theta \neq 0 \\ 0 & \text{if } \theta - 0 \end{cases}$$



Wasserstein距离W：即使两个分布没有重叠或可忽略不计，Wasserstein距离仍然能够反映它们的远近。

$$W(P_1, P_2) = \inf_{\gamma\sim\Pi(P_1,P_2)} \mathbb{E}_{(x,y)\sim\gamma}[||x-y||]$$

$$W(P_0, P_1) = |\theta|$$

分布	C	D
A	0.5	0
B	0	0.5

距离	C	D
A	θ	$\sqrt{1+\theta^2}$
B	$\sqrt{1+\theta^2}$	θ

优势：KL散度和JS散度是突变的，要么最大要么最小，Wasserstein距离却是平滑的。如果我们用梯度下降法优化这个参数，前两者根本提供不了梯度，Wasserstein距离却可以。类似地，在高维空间中如果两个分布不重叠或者重叠部分可忽略，则KL和JS既反映不了远近，也提供不了梯度，但是Wasserstein却可以提供有意义的梯度。

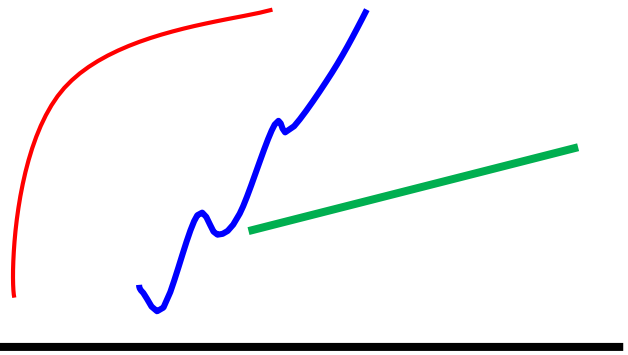
WGAN

Wasserstein 距离到损失

Wasserstein 距离：

$$W(P_1, P_2) = \inf_{\gamma \sim \Pi(P_1, P_2)} \mathbb{E}_{(x,y)\sim\gamma}[||x - y||]$$

inf （穷举所有可能，取下界）项**无法直接求解！**



Wasserstein 损失（Kantorovich-Rubinstein二元性）：

$$W(P_1, P_2) = \frac{1}{K} \sup_{||f||_L \le K} \mathbb{E}_{x\sim P_1}[f(x)] - \mathbb{E}_{x\sim P_2}[f(x)]$$

sup: 所有可能的上界

$||f||_L \le K \quad |f(x_1) - f(x_2)| \le K|x_1 - x_2|$ 函数f的利普希茨（Lipschitz）常数 $||f||_L$x小于K

Lipschitz连续：一个函数f(x)， 对于x任意小的波动，f的变化小于一个常数k。此时k成为函数f的Lipschitz常数。Lipschitz连续条件限制了一个连续函数的最大局部变动幅度。

f任意函数->神经网络！

期望->采样求和求平均！

SUP->近似等于最大化神经网络f！

WGAN损失：

$$\min_G \max_{D,||D||_L \le K} W(P_{data}, P_g) = \mathbb{E}_{x\sim P_{data}}[D(x)] - \mathbb{E}_{x\sim P_g}[D(x)]$$

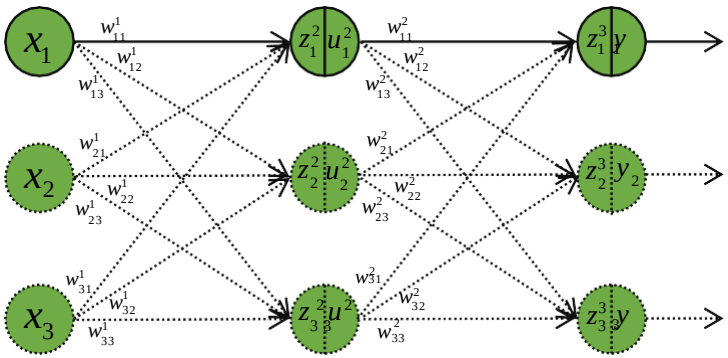
WGAN

利普希茨连续-权重截断

权重截断：在每次优化判别器D之后，把D的权重的绝对值限制在某个范围。

$$w = \begin{cases} w & \text{if } |w| \le c \\ +c & \text{if } w > +c \\ -c & \text{if } w < -c \end{cases}$$

3层全连接神经网络，每层3个节点



公式表示

$$\begin{aligned}
z_1^2 &= w_{11}^1 x_1 + w_{21}^1 x_2 + w_{21}^1 x_3 & u_1^2 &= \sigma(z_1^2) \\
z_2^2 &= w_{12}^1 x_1 + w_{22}^1 x_2 + w_{32}^1 x_3 & u_2^2 &= \sigma(z_2^2) \\
z_3^2 &= w_{13}^1 x_1 + w_{23}^1 x_2 + w_{33}^1 x_3 & u_3^2 &= \sigma(z_3^2) \\
z_1^3 &= w_{11}^2 u_1^2 + w_{21}^2 u_2^2 + w_{21}^2 u_3^2 & y_1 &= \sigma(z_1^3) \\
z_2^3 &= w_{12}^2 u_1^2 + w_{22}^2 u_2^2 + w_{32}^2 u_3^2 & y_2 &= \sigma(z_2^3) \\
z_3^3 &= w_{13}^2 u_1^2 + w_{23}^2 u_2^2 + w_{33}^2 u_3^2 & y_3 &= \sigma(z_3^3)
\end{aligned}$$

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

根据链式法则求导发现，导数正比于权重，所以截断权重w可以使网络满足利普希茨连续

$$\frac{dy_1}{dx_1} = \frac{dy_1}{dz_1^3} \cdot \frac{dz_1^3}{du_1^2} \cdot \frac{du_1^2}{dz_1^2} \cdot \frac{dz_1^2}{x_1} = y_1(1-y_1) \cdot w_{11}^2 \cdot u_1^2(1-u_1^2) \cdot w_{11}^1$$

$$\frac{dy_2}{dx_2} = \frac{dy_2}{dz_2^3} \cdot \frac{dz_2^3}{du_2^2} \cdot \frac{du_2^2}{dz_2^2} \cdot \frac{dz_2^2}{x_2} = y_2(1-y_2) \cdot w_{11}^2 \cdot u_2^2(1-u_2^2) \cdot w_{22}^1$$

……

WGAN

WGAN算法实现

GAN目标函数: $\min_G \max_D V(G,D) E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1-D(G(z)))]$

WGAN目标函数: $\min_G \max_{D,||D||_L \le K} W(P_{data}, P_g) = \mathbb{E}_{x\sim P_{data}}[D(x)] - \mathbb{E}_{x\sim P_g}[D(x)]$

WGAN实现:

1.判别器最后一层去掉 sigmoid 激活函数：原始GAN中，最优的D是一个概率形式，属于[0,1]，因此用sigmoid激活函数可以加速D的训练。WGAN中，D是任意满足利普斯次连续的函数，因此不需要用Sigmoid激活。

GAN $D^*(x) = \frac{p_{data}(x)}{p_{data}(x)+p_g(x)}$

2. 生成器G 和判别器D 的损失不取对数：根据Wasserstein距离的定义可知，不需要取对数。

3.每次更新判别器D 的参数之后，权重绝对值截断到不超过一个固定常数c：使得判别器D 满足利普希茨连续。

4.不基于动量的优化算法（包括momentum和Adam），推荐RMSProp或者SGD：经验。

WGAN

WGAN算法流程图

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

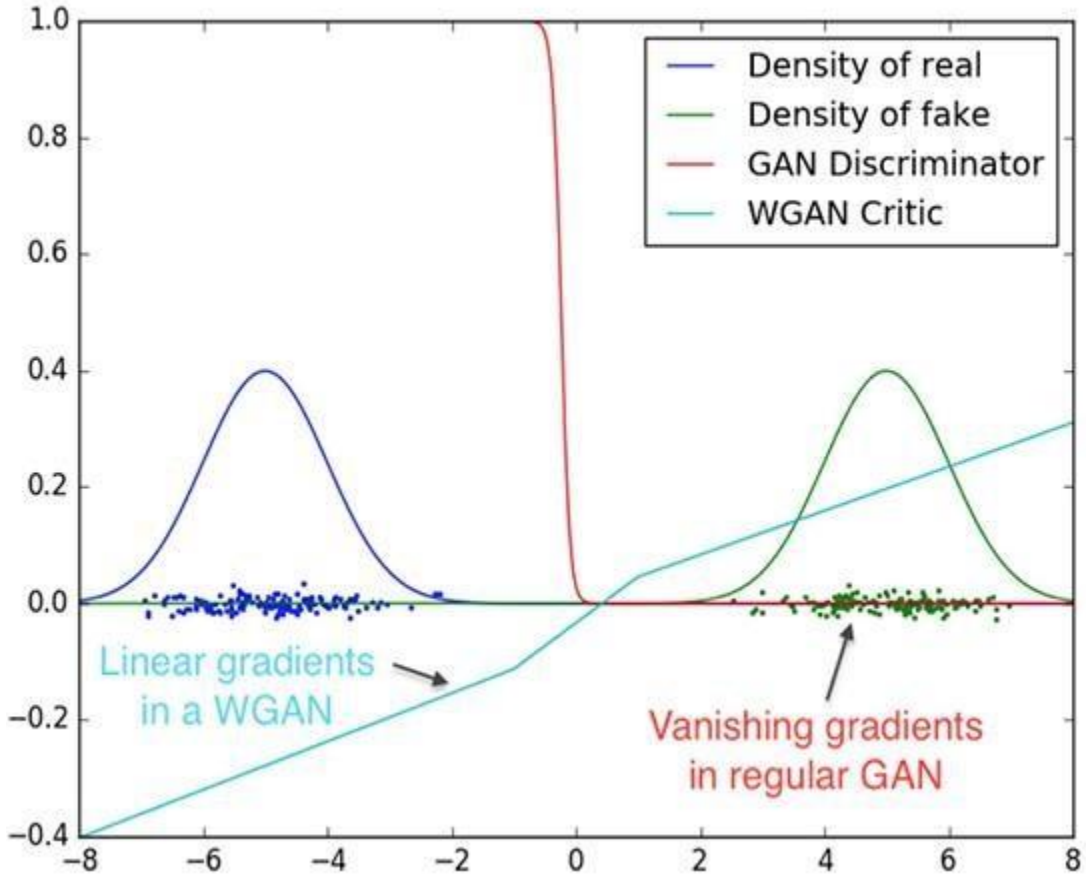
Require: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.

Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

while θ has not converged **do**
  for $t = 0, ..., n_{\text{critic}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_r$ a batch from the real data.
    Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
    $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))\right]$
    $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
    $w \leftarrow \text{clip}(w, -c, c)$
  end for
  Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
  $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
end while

WGAN

原始GAN v.s. WGAN



原始GAN梯度消失严重，WGAN仍然可以提供梯度

WGAN

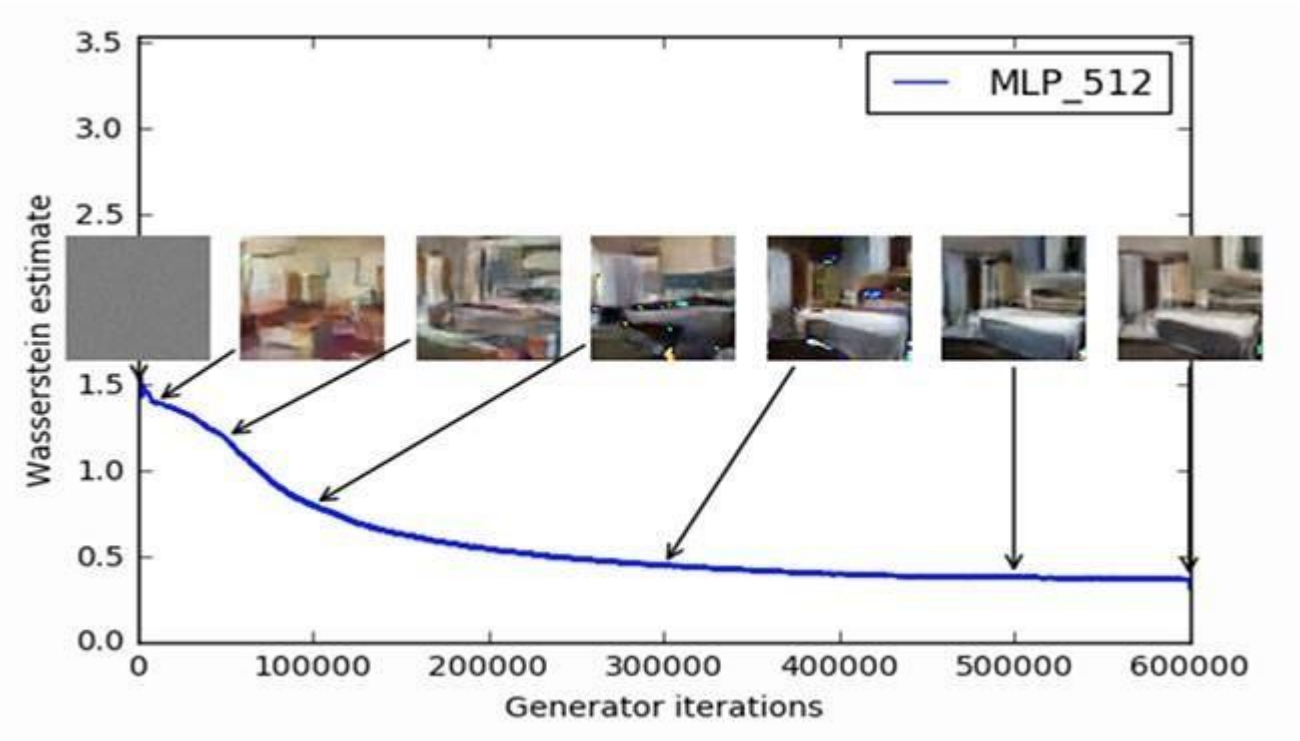
实验结果

上图：WGAN
下图：DCGAN
结论：WGAN的视觉效果比DCGAN略胜一筹

上图：不用BN的 WGAN
下图：不用BN的 DCGAN
结论：DCGAN完全崩溃，WGAN仍可以稳定生成

WGAN

W距离指示优化进度



原始GAN的损失函数无法指示：无论两个分布远近，JS散度都是常数。
WGAN损失可以指示训练进度：W距离可以表示两个分布的远近。

目录

CONTENTS

WGAN-GP

回顾WGAN

WGAN目标函数: $\min_G \max_{D,||D||_L \leq K} W(P_{data}, P_g) = \mathbb{E}_{x \sim P_{data}}[D(x)] - \mathbb{E}_{x \sim P_g}[D(x)]$

Lipschitz连续：一个函数f(x)， 对于x任意小的波动，f的变化小于一个常数k。此时k成为函数f的Lipschitz常数。

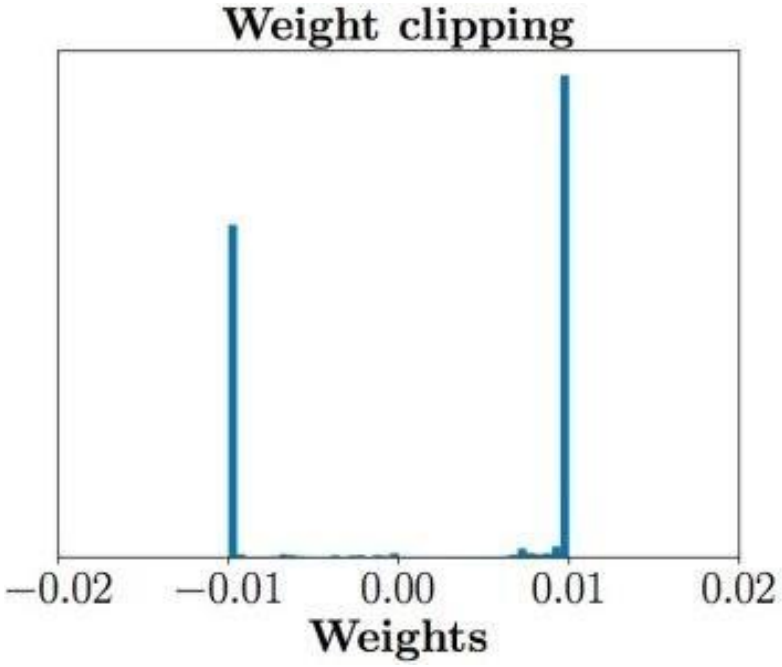
$$|f(x_1) - f(x_2)| \leq K|x_1 - x_2|$$

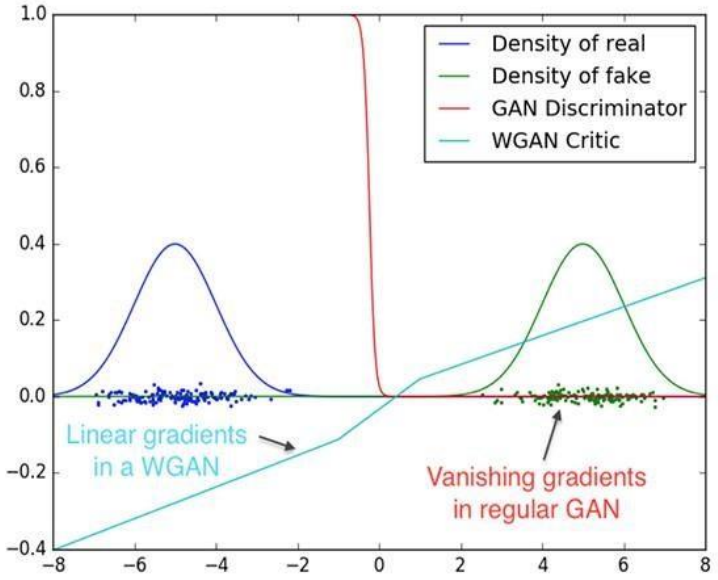
权重截断：为了使得判别器D 满足Lipschitz连续，强制D 所有权重的绝对值小于一个常数c。

$$w \begin{cases} w & \text{if } |w| \leq c \\ +c & \text{if } w > +c \\ -c & \text{if } w < -c \end{cases}$$

WGAN-GP

缺陷1：权重二值化





WGAN几乎所有参数都是正负0.01：
浪费神经网络的拟合能力。

判别器越好，分界面越抖，梯度绝对值越大，权重绝对值越大，就有越多的权重被截断到边界。

WGAN-GP

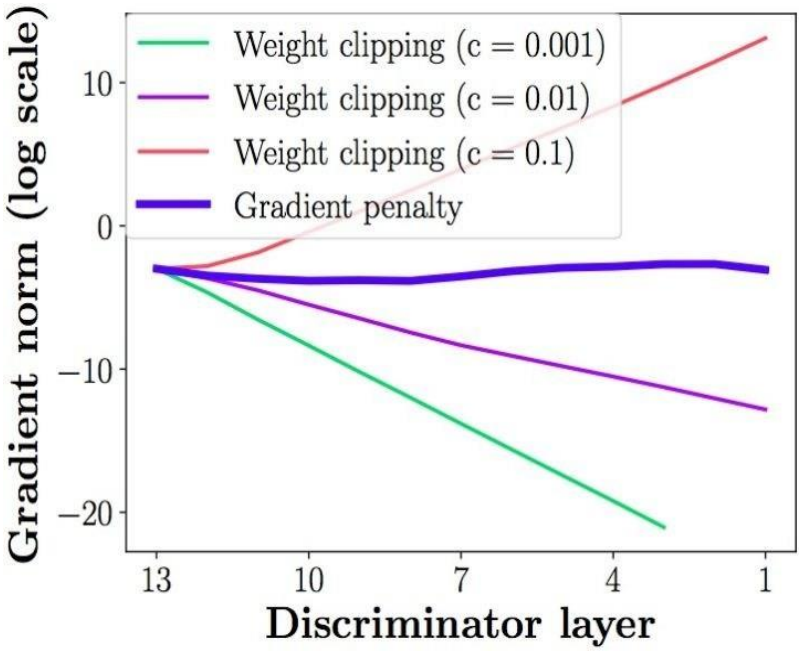
缺陷2：梯度衰减、爆炸

参数截断阈值

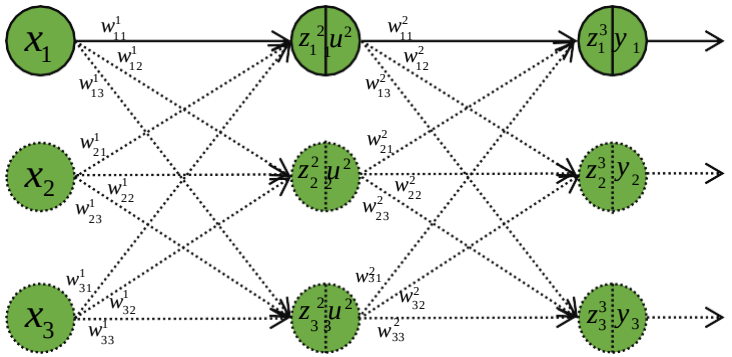
小：根据求导的链式法则，经过多层反向传播，梯度会变得非常小，造成梯度消失

大：根据求导的链式法则，经过多层反向传播，梯度会变得非常大，造成梯度爆炸

适中：实际中适中区域很小，给调参增加了难度



3层全连接神经网络，每层3个节点



$$\frac{dy_1}{dw_{11}^1} = y_1(1-y_1)\cdot w_{11}^2 \cdot u_1^2(1-u_1^2)\cdot x_1$$

$$\frac{dy_2}{dw_{11}^2} = y_2(1-y_2)\cdot u_1^2$$

WGAN-GP

梯度惩罚项

WGAN：梯度截断太“武断”，对于所有参数采取一刀切的方式。尽管满足了Lipschitz连续，但是却“滥杀无辜”。如何让模型既满足Lipschitz限制，又更加“人性化”地学习参数。

WGAN-GP：把Lipschitz限制作为一个正则项加到Wasserstein损失上，在优化GAN损失的同时，尽可能满足Lipschitz限制。

WGAN 目标函数

$$\min_{G} \max_{D,||D||_L \leq K} W(P_{data}, P_g) = \mathbb{E}_{x \sim P_{data}}[D(x)] - \mathbb{E}_{x \sim P_g}[D(x)]$$

WGAN使用权重截断来近似满足利普希茨连续条件：

$$w = \begin{cases} w & \text{if } |w| \leq c \\ +c & \text{if } w > +c \\ -c & \text{if } w < -c \end{cases}$$

WGAN-GP使用一个正则项来近似满足利普希茨连续条件：

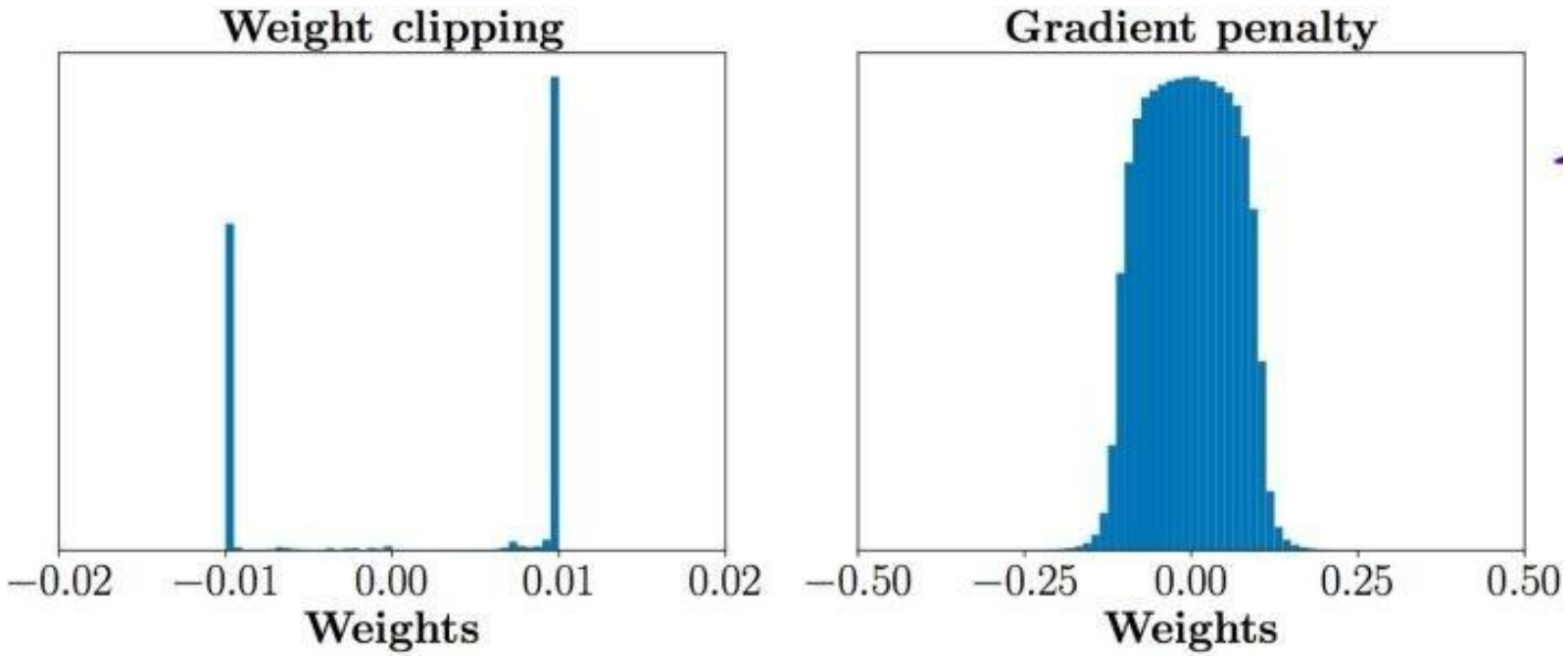
$$\min_{G} \max_{D} W(P_{data}, P_g) = \mathbb{E}_{x \sim P_{data}}[D(x)] + \mathbb{E}_{x \sim P_g}[D(x)] + \lambda \mathbb{E}_{x \sim \mathcal{X}}[||\nabla_x D(x)||_p - K]^2$$

$$x_{data} \sim P_{data}, x_g \sim P_g, \epsilon \sim Uniform[0,1]$$

$$\hat{x} = \epsilon x_{data} + (1-\epsilon) x_g$$

WGAN-GP

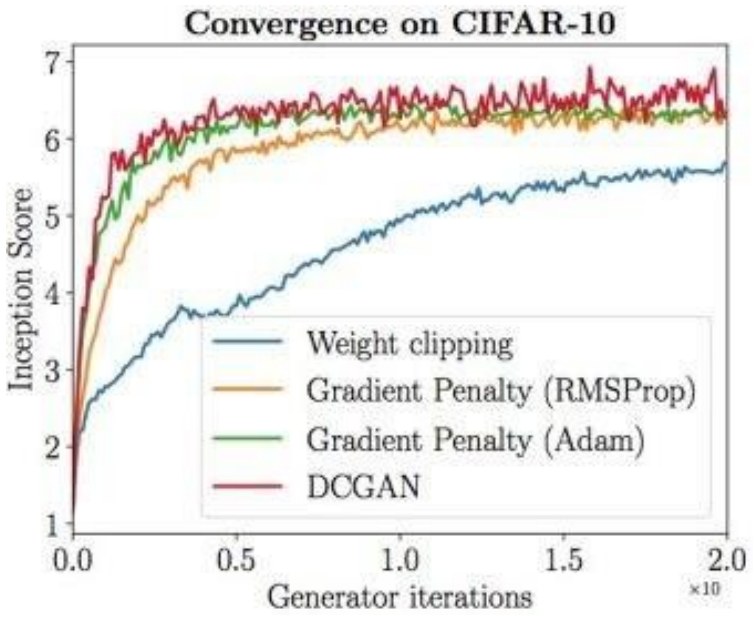
梯度更加均匀

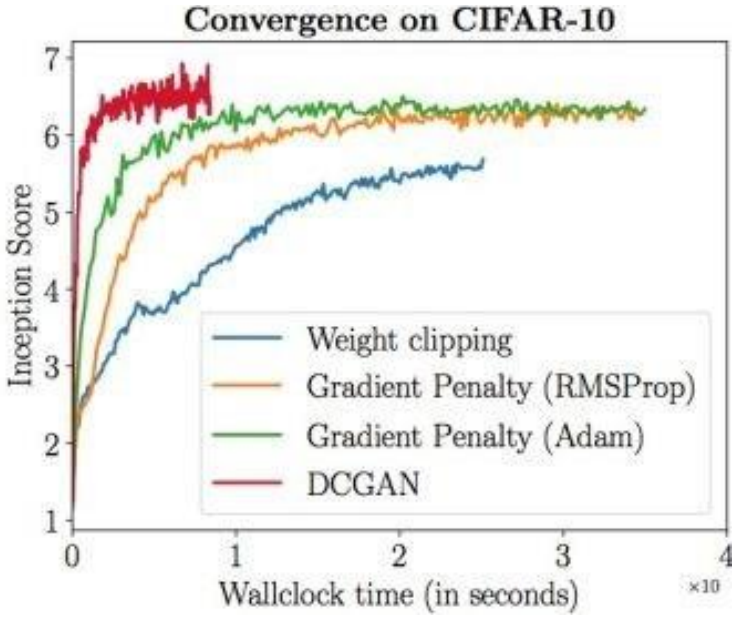


参数分布更加均匀！

WGAN-GP

收敛速度更快



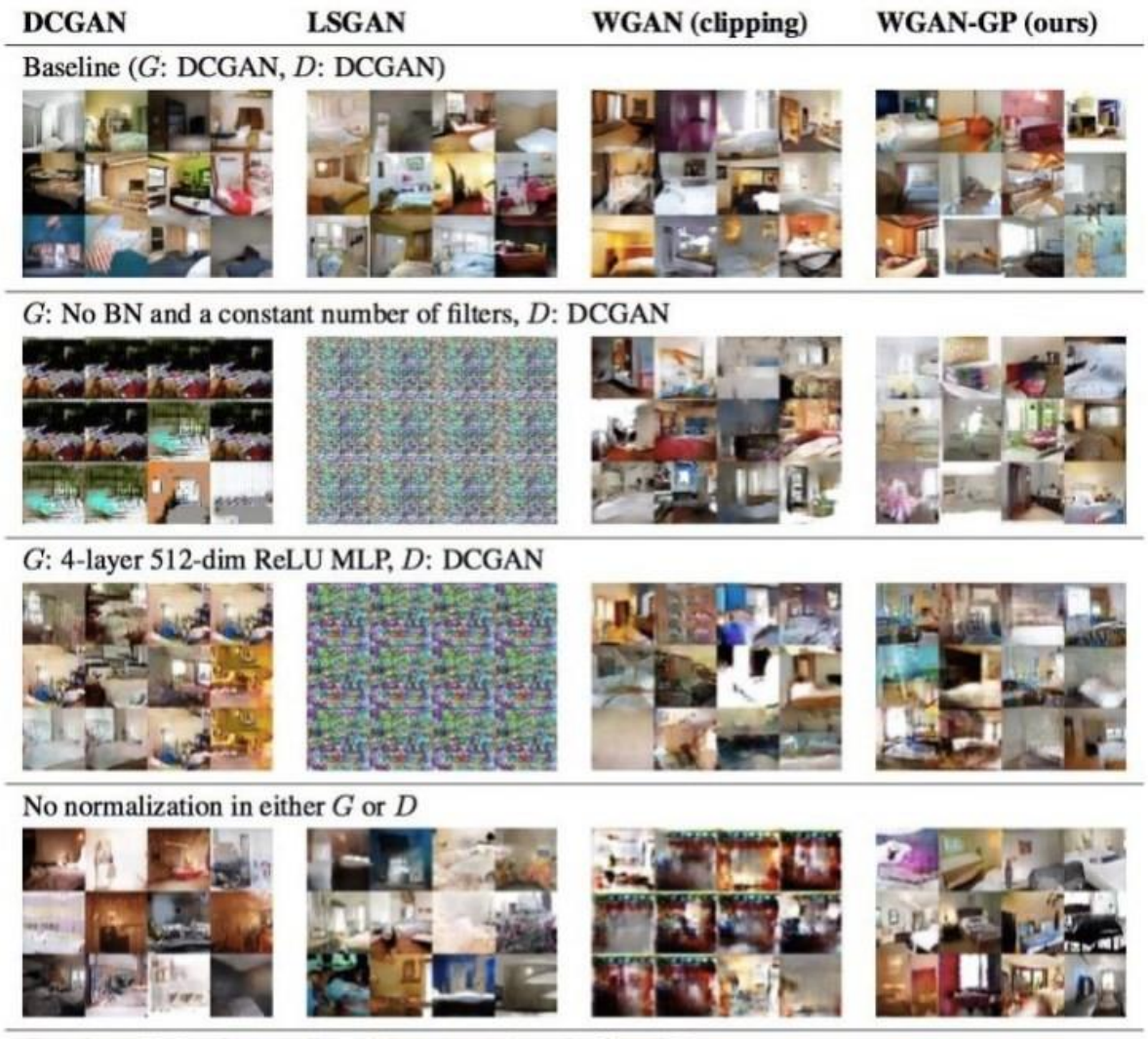


收敛速度显著高于WGAN！
但是仍然慢于DCGAN。

WGAN-GP

对模型更鲁棒

在各种配置下，WGAN-GP均能稳定发挥！
稳定性远远高于DCGAN！



WGAN-GP

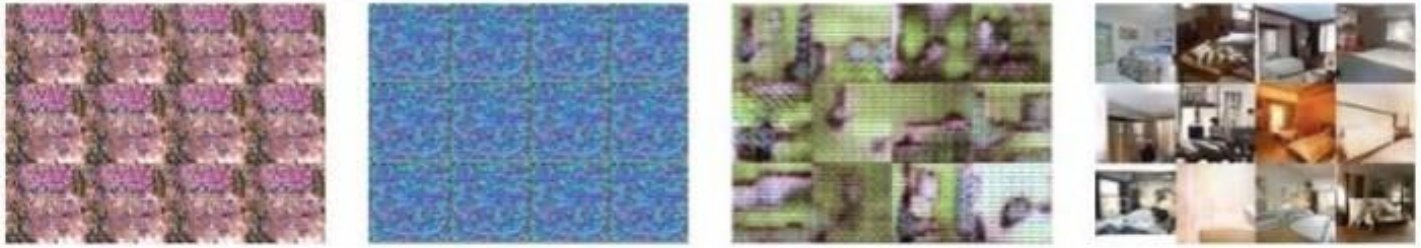
对模型更鲁棒

在各种配置下，WGAN-GP均能稳定发挥！
稳定性远远高于DCGAN！

Gated multiplicative nonlinearities everywhere in G and D

tanh nonlinearities everywhere in G and D

101-layer ResNet G and D



THANKS