Generative Adversarial Network
生成式对抗网络

王宗威

2019/06/30

采样？抽样？
(sample)

张帅

采样的定义

- 采样（Sampling）是指从目标总体中抽取一部分个体作为样本，通过观察样本的某一或某些属性，依据所获得的数据对总体的数量特征得出具有一定可靠性的估计判断，从而到达对总体的认识。

抽样类型

1. 简单随机采样（sample random sampling）
2. 系统采样（systematic sampling）
3. 分层采样（stratified sampling）
4. 整群采样（cluster sampling）
5. 图像采样（image sampling）

简单随机采样

- 也叫纯随机采样。从总体N个单位中随机地抽取n个单位作为样本，使得每个样本单位被抽中的概率相等。要求每个单位完全独立，彼此间无一定的关联性和排斥性。
- 可以用抽签法或随机数法

系统采样

- 也称为等距采样。将总体中的所有单位按一定顺序排列，在规定的范围内随机地抽取一个单位作为初始单位，然后按事先规定好的规则确定其他样本单位。
- 个体数目较大且无较大差异、将总体均分成若干部分、分段间隔相等、在第一段内用简单随机抽样抽取，其余依次加上间隔的整数倍、等可能抽样.

系统采样举例

- 某校有学生1200人，为了调查某种情况打算抽取一个样本容量为50的样本。
- ⑴随机地将这1200名学生编号为1，2，3，…，1200.
- ⑵将总体按编号顺序均分成50部分，每部分包括24个个体.
- ⑶在第一部分的个体编号1，2，3，…，24中，利用简单随机抽样抽取一个号码，比如是18.
- ⑷以18为起始号码，每间隔24抽取一个号码，这样得到一个容量为50的样本：18，42，66，…，982，1002

分层采样

- 将样本单位按某种特征或某种规则划分为不同的层，然后从不同的层中独立、随机地抽取样本。从而保证样本的结构与总体的结构比较相近，从而提高估计的精度。

分层采样举例

- 有A、B、C三类数据各100个，需要从中取n个样本，为了保证数据分布的一致性，应分布从A、B、C中取n/3个样本。

整群采样

- 将总体中若干个单位合并为组，抽样时直接抽取群，然后对中选群中的所有单位全部实施调查，抽样时只需群的抽样框，可简化工作量，缺点是估计的精度较差。

整群采样举例

- 检验某种零件的质量时，不是逐个抽取零件，而是随机抽若干盒（每盒装有若干个零件），对所抽各盒零件进行全面检验。

图像采样

- 在图像区域的一些特定位置上取出图像的亮度值（或色度值），以此作为原图像的一种替代，这一过程就称作图像的采样，而每一个采样的位置称为采样点，该点的亮度值（或色度值）就是抽样值。

抽样数据方式

1. 留出法（hold-out）
2. 交叉验证法（cross validation）
3. 自助法（bootstrapping）

留出法

- 留出法直接将数据集D划分为两个互斥的集合，其中一个集合作为训练集S，另一个作为测试集T，在S上训练出模型后，用T来评估其测试误差，作为对泛化误差的估计。
- 需要注意的是，训练/测试集的划分要尽可能保持数据分布的一致性。如果从采样的角度来看待数据集的划分过程，则保留类别比例的采样方式通常称为“分层采样”。

交叉验证法

- 交叉验证法先将数据集D划分为k个大小相似的互斥子集，每个子集D_i都尽可能保持数据分布的一致性，即从D中通过分层采样得到。然后，每次用k-1个子集的并集作为训练集，余下的那个子集作为测试集。

自助法

- 在留出法和交叉验证法中，由于保留了一部分样本用于测试，因此实际评估的模型所使用的数据集比D小，这必然会引入一些因训练样本规模不同而导致的估计偏差。留一法受训练样本规模变化的影响较小，但计算复杂度又太高了。

自助法

- 给定包含m个样本的数据集，我们对它采样产生数据集D’：从D进行m次有放回的随机抽样。显然，D中有一部分样本会在D’中多次出现，而另一部分样本不出现。样本在m次采样中始终不被采到的概率是

$$\left(1-\frac{1}{m}\right)^{m}$$

自助法

- 取极限得到

$$\lim_{m\to\infty}\left(1-\frac{1}{m}\right)^{m}=\frac{1}{e}\approx 0.368$$

- 即有36.8%的样本未出现在D’中。
- 我们用未出现的样本作为测试样本。这样，实际评估的模型在期望评估的模型都使用m个训练样本。
- 自助法在数据集较小、难以有效划分训练/测试集时很有用。

预备知识

信息熵，交叉熵，KL散度，JS散度

钞美玲　2019/07/07

信息熵

信息熵，是香农从热力学中借用过来的。热力学中的热熵是表示分子状态混乱程度的物理量，香农用信息熵的概念来描述信源的不确定度。通常，一个信源发送出什么符号是不确定的，衡量它可以根据其出现的概率来度量。概率大，出现机会多，不确定性小；反之不确定性就大。

香浓给出了信息熵的三个性质：

1. 单调性，发生概率越高的事件，其携带的信息量越低；
2. 非负性，信息熵可以看作为一种广度量，非负性是一种合理的必然；
3. 累加性，即多随机事件同时发生存在的总不确定性的量度是可以表示为各事件不确定性的量度的和，这也是广度量的一种体现。

信息熵的定义公式：

$$H(X) = -\sum_{x \in \chi} p(x) \log p(x)$$

对信息熵三条性质的理解：

单调性说的是，事件发生的概率越低，其发生时所能给出的信息量越大。举一个极端的例子，“太阳从西边升起”所携带的信息量就远大于“太阳从东边升起”，因为后者是一个万年不变的事实，不用特意述说大家都知道；而前者是一个相当不可能发生的事情，如果发生了，那代表了太多的可能性，可能太阳系有重大变故，可能物理法则发生了变化，等等。对累加性的解释，考虑到信息熵的定义涉及到了事件发生的概率，我们可以假设信息熵是事件发生概率的函数：

$$H(X) = H(p(x))$$

对于两个相互独立的事件 X=A,Y=B来说，其同时发生的概率：

$$p(X = A, Y = B) = p(X = A) \cdot (Y = B)$$

其同时发生的信息熵，根据累加性可知：

$$H\left(p(X = A, Y = B)\right) = H\left(p(X = A) \cdot (Y = B)\right) = H\left(p(X = A)\right) + H\left(p(Y = B)\right)$$

一种函数形式，满足两个变量乘积函数值等于两个变量函数值的和，那么这种函数形式应该是对数函数。再考虑到概率都是小于等于 1 的，取对数之后小于 0，考虑到信息熵的第二条性质，所以需要在前边加上负号。

KL散度

在机器学习中，P往往用来表示样本的真实分布，比如[1,0,0]表示当前样本属于第一类。Q用来表示模型所预测的分布，比如[0.7,0.2,0.1]。直观的理解就是如果用P来描述样本，那么就非常完美。而用Q来描述样本，虽然可以大致描述，但是不是那么的完美，信息量不足，此时需要一个量来衡量近似分布Q比原分布P损失了多少信息量。这就是KL散度起作用的地方。

KL散度的计算公式其实是熵计算公式的简单变形, 在原有概率分布 p 上，加入我们的近似概率分布q，计算他们的每个取值对应对数的差：

$$D_KL(p||q) = \sum_{i=1}^{N} p(x_i) \cdot (\log p(x_i) - \log q(x_i))$$

换句话说，KL散度计算的就是数据的原分布与近似分布的概率的对数差的期望值。

在对数以2为底时，$\log 2$ 可以理解为“我们损失了多少位的信息”，写成期望形式：

$$D_K L(p||q) = E[\log p(x) - \log (q(x)]$$

更常见的是以下形式：

$$D_K L(p||q) = \sum_{i=1}^{N} p(x_i) \cdot \log \frac{p(x_i)}{q(x_i)}$$

注意：散度不是距离，并不对称。

$$D_K L(p||q) \neq D_K L(q||p)$$

因为KL散度不具有交换性，所以不能理解为“距离”的概念，衡量的并不是两个分布在空间中的远近，更准确的理解还是衡量一个分布相比另一个分布的信息损失(infomation lost)

交叉熵

对KL散度中的公式进行变形操作

$$D_{KL}(p||q) = \sum_{i=1}^{n} p(x_i) log(p(x_i)) - \sum_{i=1}^{n} p(x_i) log(q(x_i))$$

$$= -H(p(x)) + [-\sum_{i=1}^{n} p(x_i) log(q(x_i))]$$

等式的前一部分恰巧就是p的熵，等式的后一部分，就是交叉熵：

$$H(p,q) = -\sum_{i=1}^{n} p(x_i) log(q(x_i))$$

在机器学习中，我们需要评估真实值和预测值之间的差距，使用KL散度刚刚好，即 $D_{KL}(y \parallel y)$ ，由于KL散度中的前一部分−H(y)不变，故在优化过程中，只需要关注交叉熵就可以了。

应用

交叉熵在单分类问题上基本是标配的方法：$$loss = -\sum_{i=1}^{n} y_i log(\hat{y}_i)$$

例如有如下样本：

对应的标签和预测值：

	猫	青蛙	狗
Label	0	1	0
Pred	0.3	0.6	0.1

那么 $$\begin{aligned} loss &= -(0 \times log(0.3) + 1 \times log(0.6) + 0 \times log(0.1) \\ &= -log(0.6) \end{aligned}$$

对应一个batch的loss就是 $$loss = -\frac{1}{m}\sum_{j=1}^{m}\sum_{i=1}^{n} y_{ji} log(\hat{y_{ji}})$$

m为当前batch的样本数

JS散度

KL散度可以用来表示两个概率分布之间的差异，但有个不大好的地方是它并不是对称的，因此有时用它来训练神经网络会有顺序不同造成不一样的训练结果的情况。为了克服这个问题，有人就提出了一个新的衡量公式，叫做JS散度，式子如下：

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

JS散度是利用KL散度来得到的，JS是对称的而且值是有界的[0,1]。显然，如果P1，P2完全相同，那么JS =0，如果完全不相同，那么就是1。

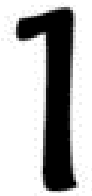
数据生成（Data generation）

- 数字的生成

生成数字1：极有可能学习到一个分布使得像素落在图像中轴线上的概率大大增加。

输入：x→P(x)，x可以是坐标点

输出：在该坐标处有像素点的概率，最终根据概率生成数字。



0	0	0	0	0	0	0	0	0
0	0.1	0.2	0.9	0.8	0.2	0.1	0	0
0	0.1	0.2	0.9	0.8	0.2	0.1	0	0
0	0.1	0.2	0.8	0.9	0.2	0.1	0	0
0	0.1	0.2	0.8	0.9	0.2	0.1	0	0
0	0.1	0.2	0.9	0.8	0.2	0.1	0	0
0	0.1	0.2	0.9	0.8	0.2	0.1	0	0
0	0.1	0.2	0.9	0.8	0.2	0.1	0	0
0	0	0	0	0	0	0	0	0

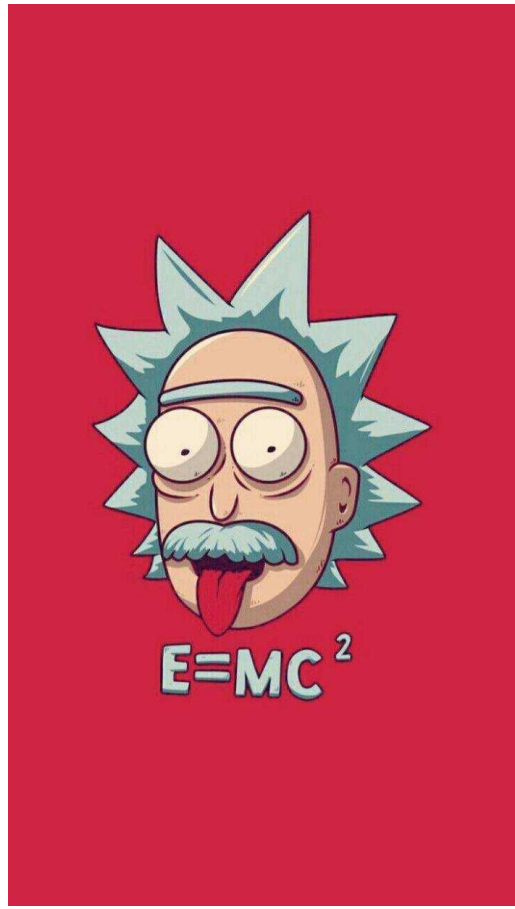
数据生成（Data generation）

- 绘画

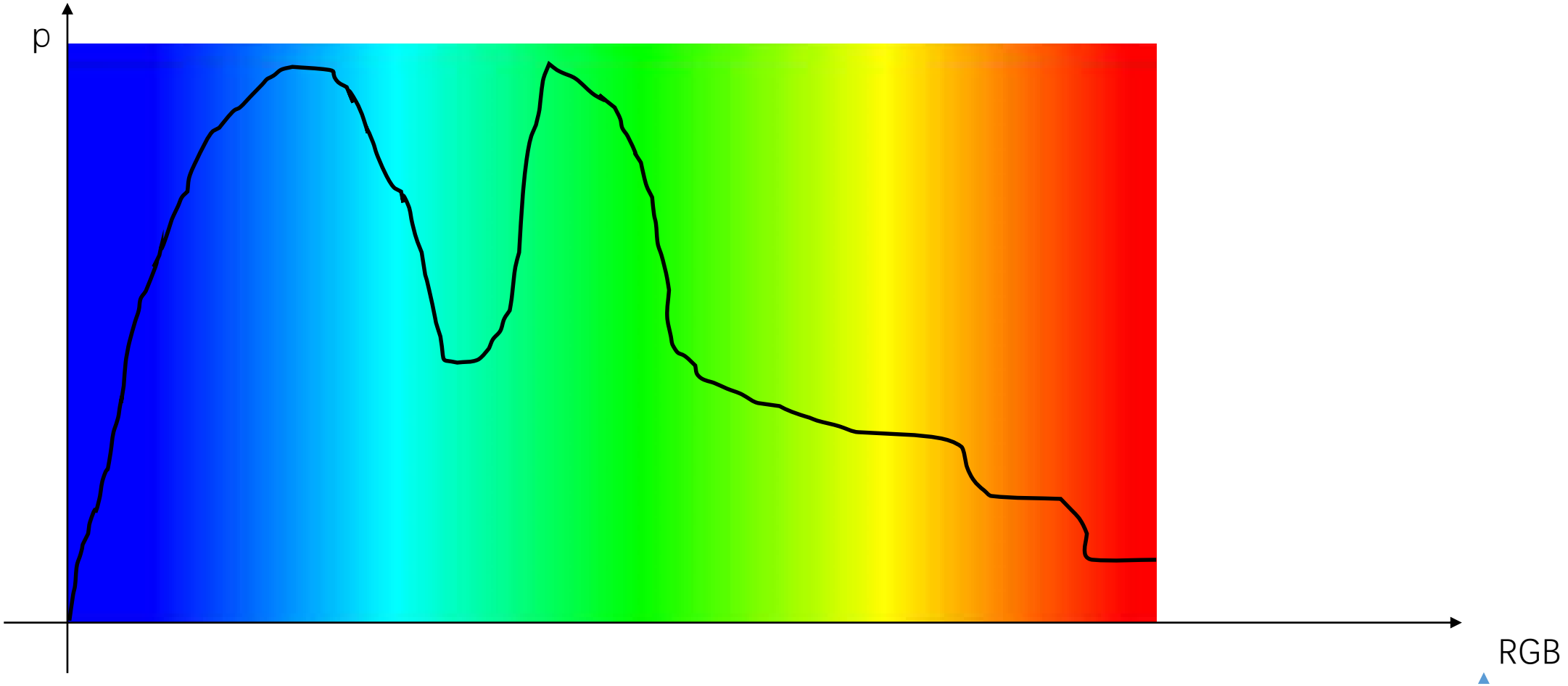
生成一幅画：可以通过当前要画的像素点的坐标和附近像素点的颜色，画出下一个点。

输入：x→P(x|pre_pixel)

输出：该坐标处下一个像素颜色的概率



绘画



数据生成（Data generation）

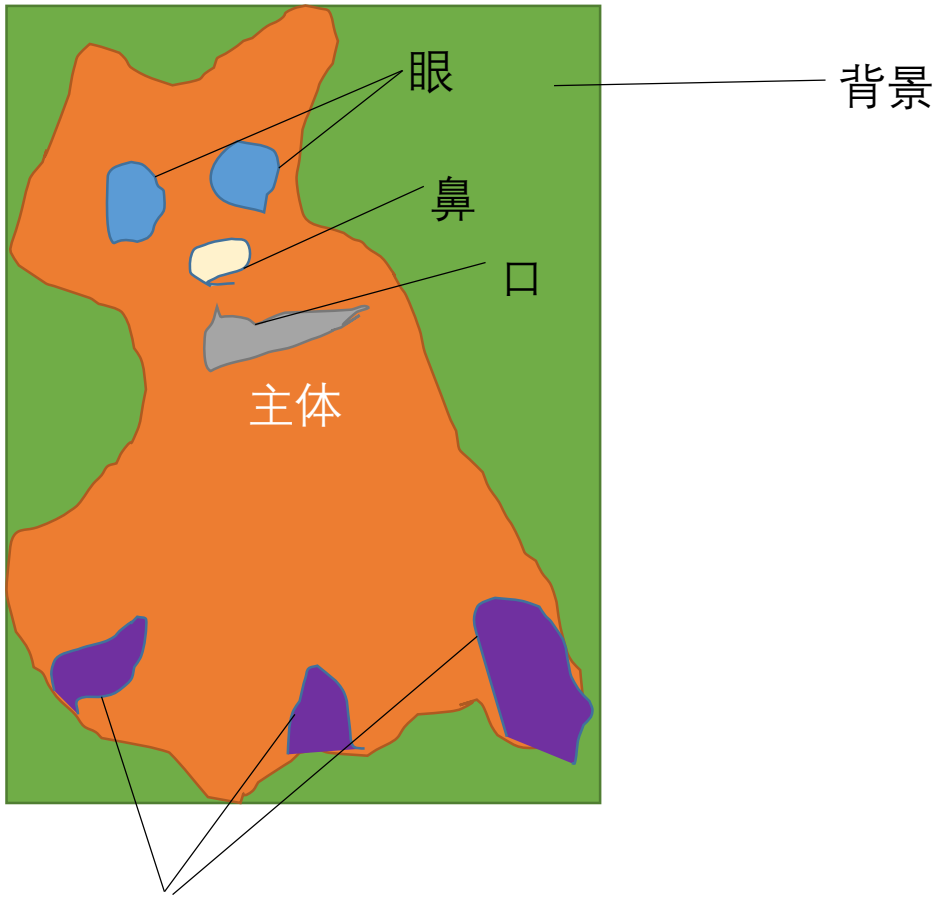
- 真实图像

生成一张狗的图像：学习到狗的外貌分布，以及图片的背景分布

输入：x→P(x)，x可以是从其他分布采样得到的

输出：一张狗的图像

真实图像



背景知识

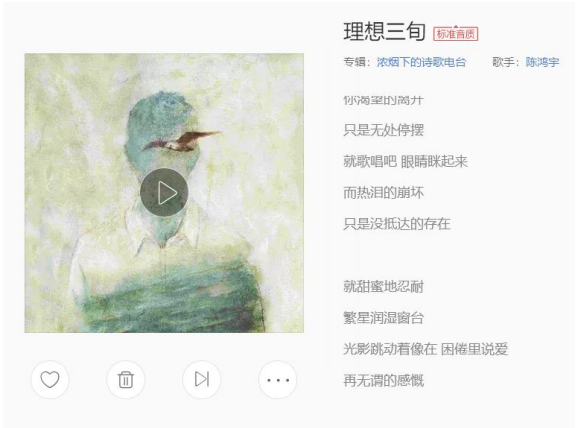
大家想一想，你们一想到生成能够想到什么？

文字

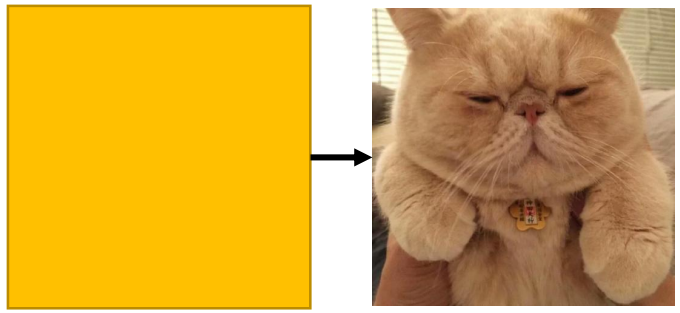
歌曲

图片

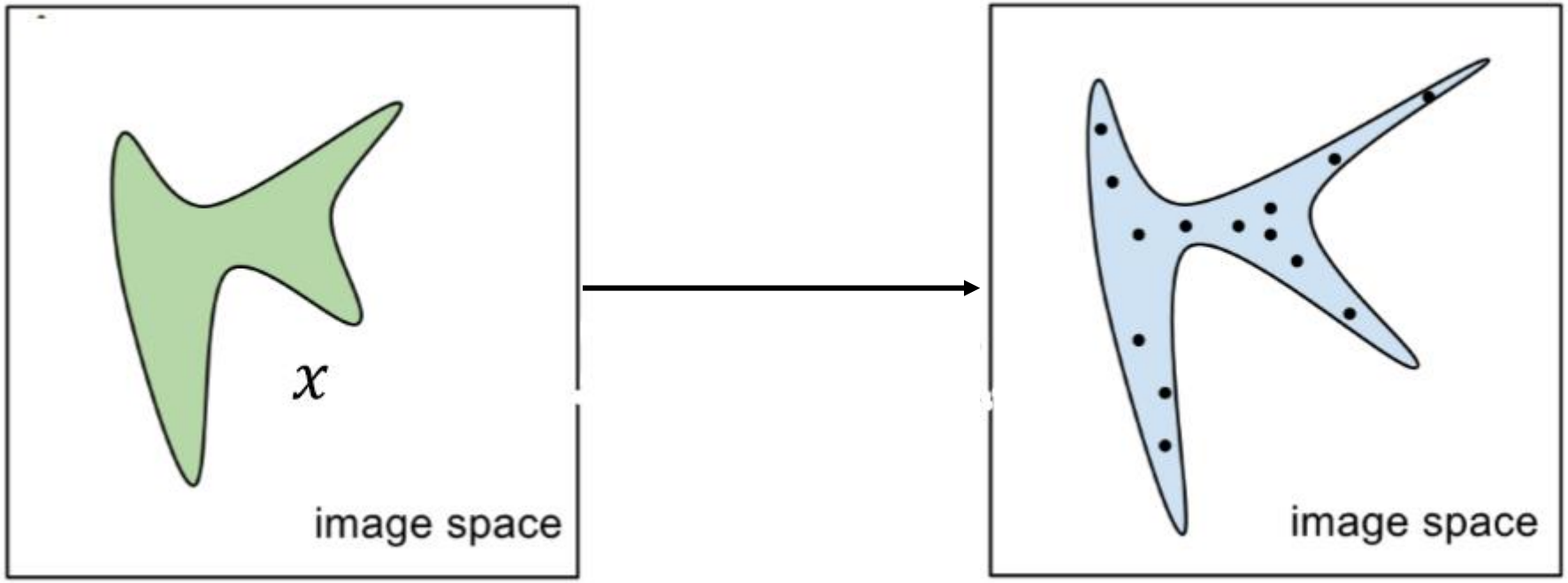
重庆大学
Chongqing University



生成的实质是什么？



1	0	1	0	1
0	1	1	0	0
1	0	0	1	1
1	0	1	1	0
1	1	0	0	0



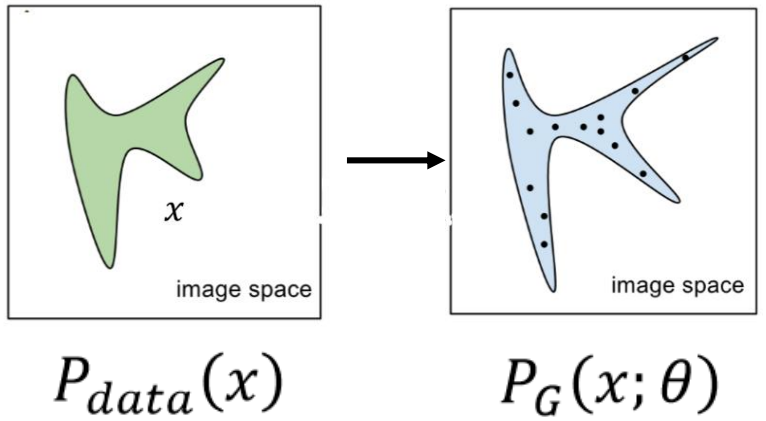
最大似然估计（Maximum Likelihood Estimation）

给定一个数据分布 $P_{data}(x)$

找到一个由参数 θ 决定的另一个数据分布 $P_G(x;\theta)$

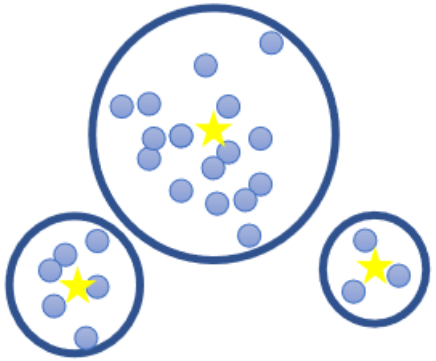
我们需要达到怎样的目的呢？

$$P_{data}(x) \quad = \quad P_G(x;\theta)$$



最大似然估计（Maximum Likelihood Estimation）

How to do it?



1	0	1	0	1
0	1	1	0	0
1	0	0	1	1
1	0	1	1	0
1	1	0	0	0

1. 采样

Sample $\{x^1, x^2, \ldots, x^m\}$ from $P_{data}(x)$

2. 计算样本的数据分布相似性

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

3. 找到最优的模型参数 θ^* 去最大化损失函数L

最大似然估计（Maximum Likelihood Estimation）

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

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

$\{x^1, x^2, \ldots, x^m\}$ from $P_{data}(x)$

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

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

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

How to have a very general $P_G(x;\theta)$?

来源：李宏毅《2017深度学习》GANv11.pdf　Page23

问题和难点

1. 请大家思考一下，这个生成的分布是怎么来的？

答：一般来说，是我们假设的分布，比如正太分布，我们所学的参数就是均值和方差。

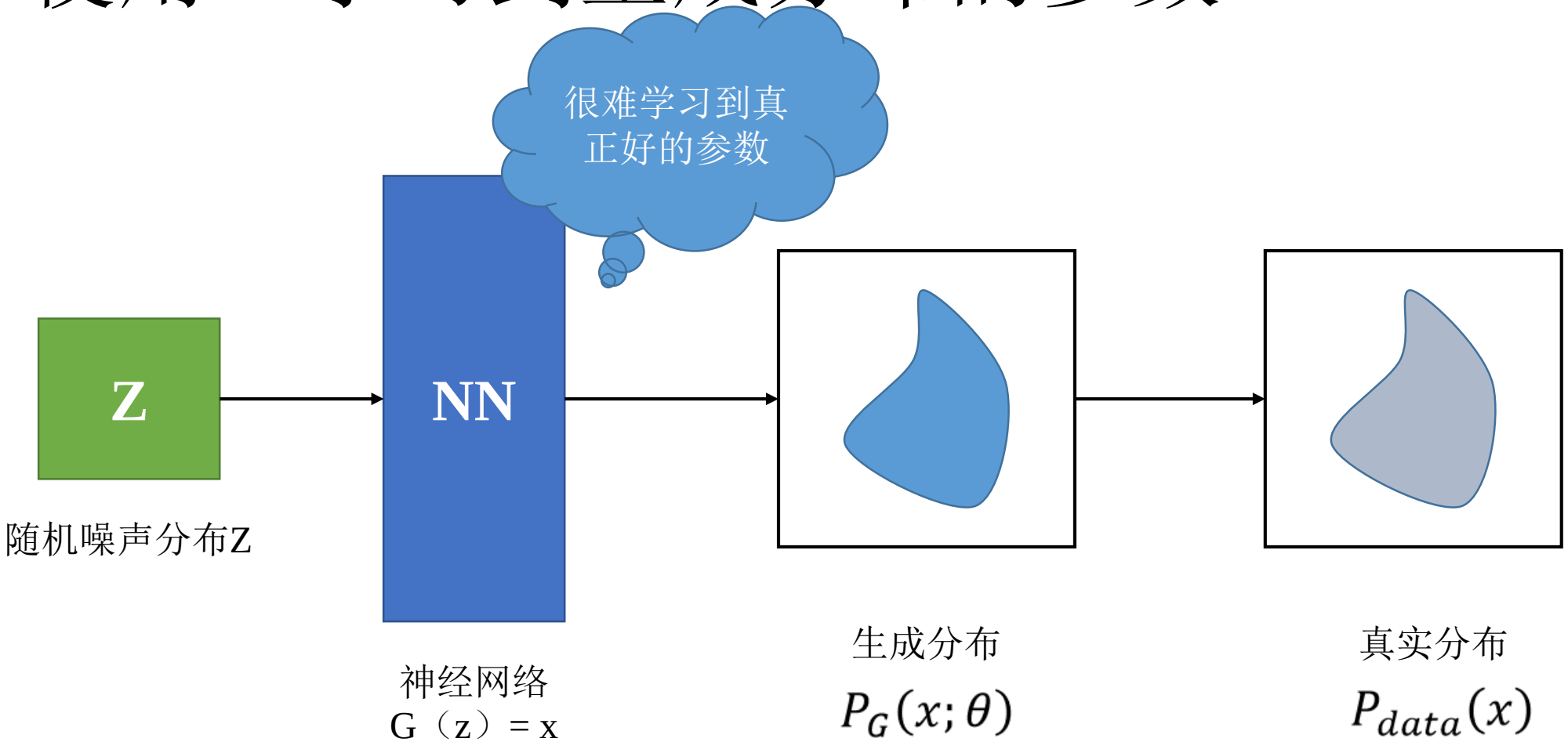
2. 真实的分布真的就和正态分布很类似吗？

答：不会，真实情况往往比这个复杂得多。

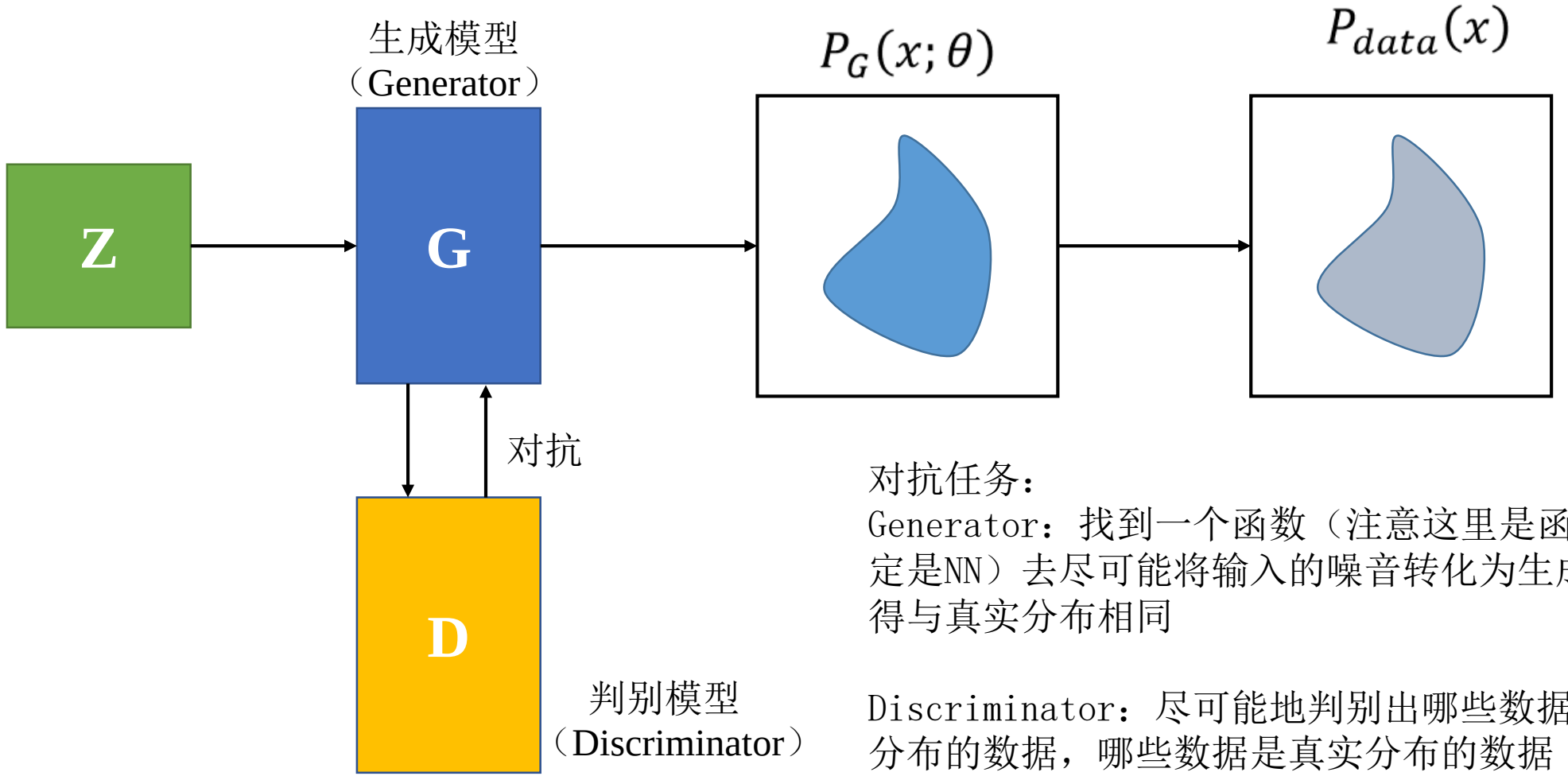
3. 如果假设的分布特别复杂怎么办？

答：使用NN。

使用NN学习到生成分布的参数



生成式对抗网络的提出



对抗任务：
Generator：找到一个函数（注意这里是函数不一定是NN）去尽可能将输入的噪音转化为生成分布使得与真实分布相同

Discriminator：尽可能地判别出哪些数据是生成分布的数据，哪些数据是真实分布的数据

具体的数学表示

生成器

Input:

随机噪音Z

（取自于一个噪音先验分布 $P_{prior}(z)$）

一个生成器的概率分布 $P_G(x)$

Output:

生成的图像X

判别器

Input：

图像X（真实的和生成的图像）

Output:

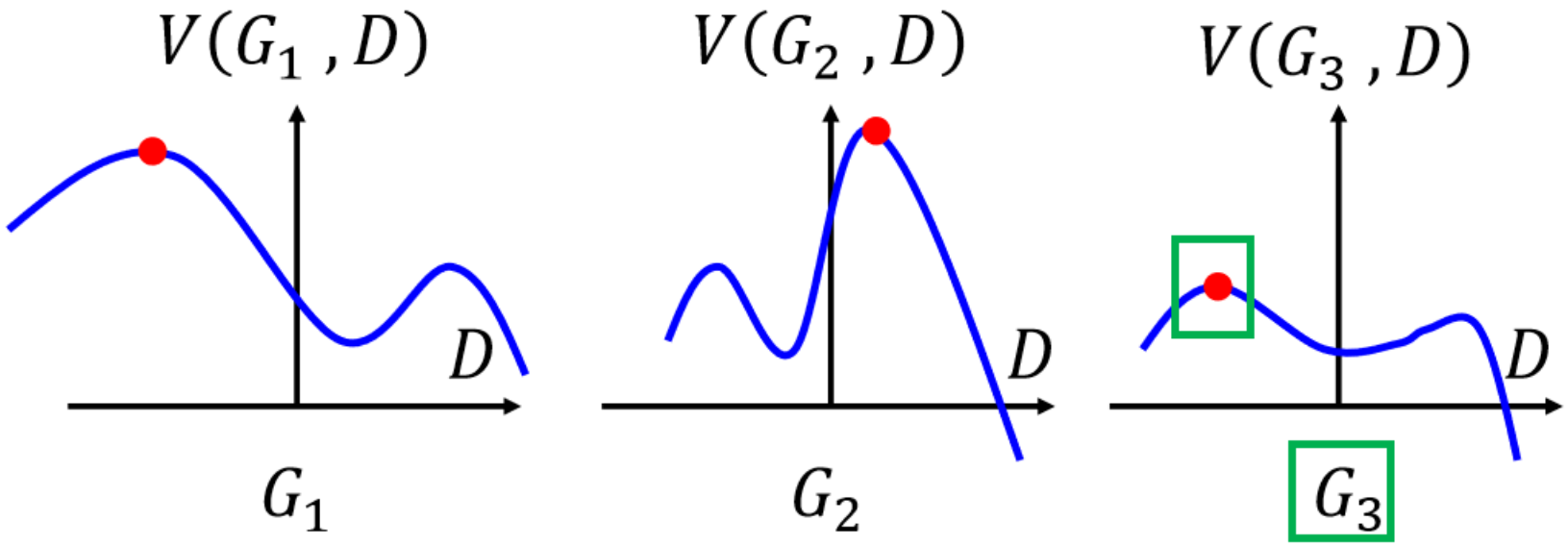
一个判别结果，标量Scalar

任务：

$$G^* = arg \min_G \max_D V(G,D)$$

$$V = E_{x \sim P_{data}}[logD(x)] + E_{x \sim P_G}[log(1 - D(x))]$$

$$G^* = arg \min_G \max_D V(G, D)$$



来源：李宏毅《2017深度学习》GANv11.pdf　Page27

$$V = E_{x\sim P_{data}}[logD(x)] + E_{x\sim P_G}[log(1-D(x))]$$

$$G^* = arg\min_G \max_D V(G,D)$$

给定一个固定的G：

$$V = E_{x\sim P_{data}}[logD(x)] + E_{x\sim P_G}[log(1-D(x))]$$

$$= \int_x P_{data}(x)logD(x)\,dx + \int_x P_G(x)log(1-D(x))\,dx$$

$$= \int_x [P_{data}(x)logD(x) + P_G(x)log(1-D(x))]\,dx$$

这个时候如果x固定，那么问题就变成了关注框里的部分

$$P_{data}(x)logD(x) + P_G(x)log\big(1 - D(x)\big)$$

a　　D　　b　　D

问题转化成了一个很简单的问题：

令　$\mathrm{f}(D) = \mathrm{a}log(D) + blog(1 - D)$

$$\frac{d\mathrm{f}(D)}{dD} = a \times \frac{1}{D} + b \times \frac{1}{1 - D} \times (-1) \ = 0$$

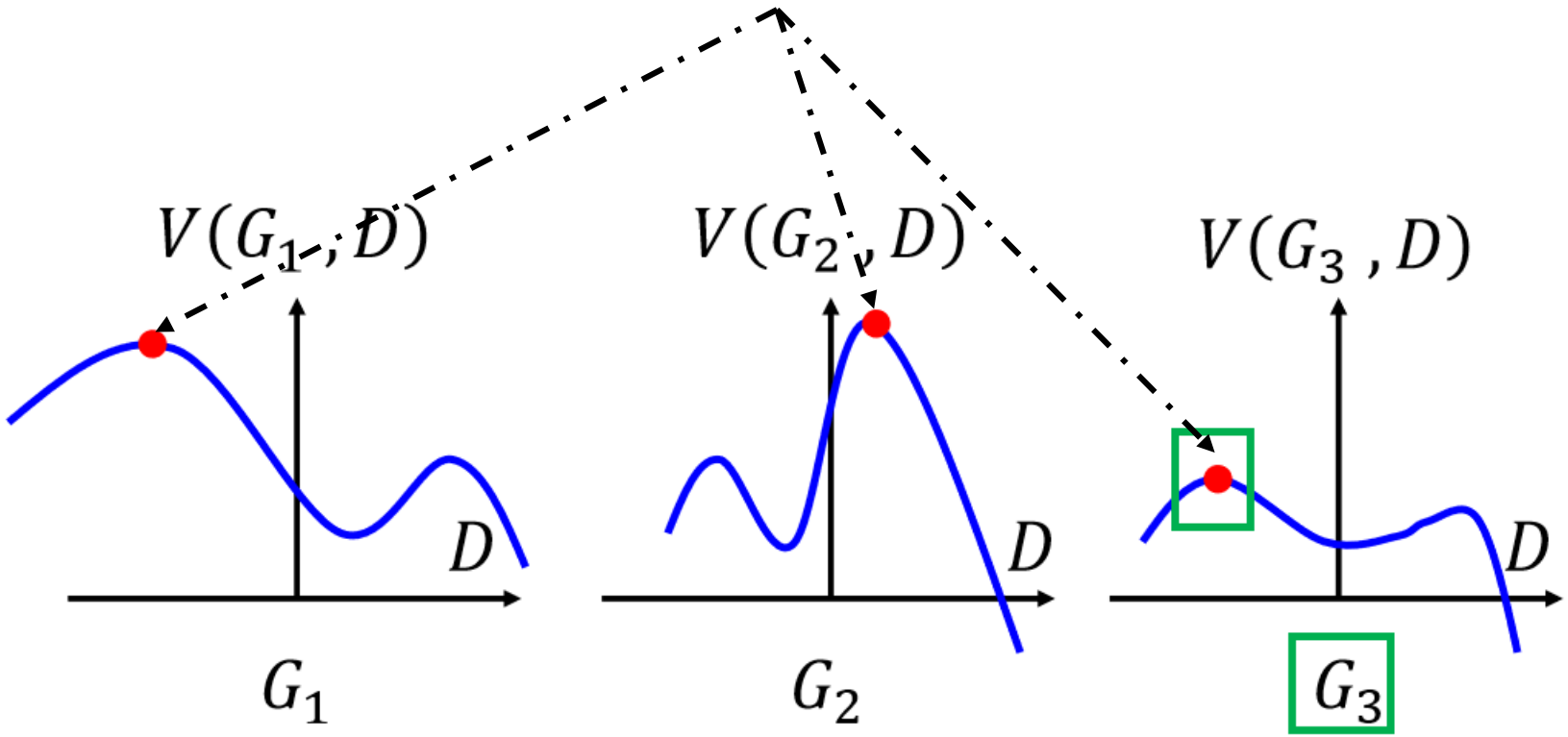
$$a \times \frac{1}{D^*} = b \times \frac{1}{1 - D^*} \qquad a \times (1 - D^*) = b \times D^*$$

$$a - aD^* = bD^*$$

$$D^* = \frac{a}{a + b} \Rightarrow \qquad D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)}$$

0 <　　< 1

$$D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)}$$



$$V = E_{x\sim P_{data}}[logD(x)] + E_{x\sim P_G}[log(1-D(x))]$$

$$= E_{x\sim P_{data}}\left[log\frac{P_{data}(x)}{P_{data}(x)+P_G(x)}\right] + E_{x\sim P_G}\left[log\frac{P_G(x)}{P_{data}(x)+P_G(x)}\right]$$

$$= \int_x P_{data}(x) log\frac{\frac{1}{2}P_{data}(x)}{\frac{P_{data}(x)+P_G(x)}{2}}dx + \int_x P_G(x) log\frac{\frac{1}{2}P_G(x)}{\frac{P_{data}(x)+P_G(x)}{2}}dx$$

$$= -2log2 + \int_x P_{data}(x) log\frac{P_{data}(x)}{(P_{data}(x)+P_G(x))/2}dx + \int_x P_G(x) log\frac{P_G(x)}{(P_{data}(x)+P_G(x))/2}dx$$

$$= -2log2 + \int_x P_{data}(x) log \frac{P_{data}(x)}{\big(P_{data}(x) + P_G(x)\big)/2} dx$$

$$+ \int_x P_G(x) log \frac{P_G(x)}{\big(P_{data}(x) + P_G(x)\big)/2} dx$$

$$= -2\log2 + \mathrm{KL}\left(\mathrm{P_{data}(x)} || \frac{\mathrm{P_{data}(x) + P_G(x)}}{2}\right)$$

$$+\mathrm{KL}\left(\mathrm{P_G(x)} || \frac{\mathrm{P_{data}(x) + P_G(x)}}{2}\right)$$

$$= -2log2 + 2JSD\big(P_{data}(x) || P_G(x)\big)$$

到这里，为什么这么设置目标函数的原因终于出现了！问题就转化成了一个求JS散度的问题

$$\max_D V(G,D) = -2log2 + 2JSD\big(P_{data}(x) || P_G(x)\big)$$

$$\max_D V(G,D) = -2log2 + 2JSD\left(P_{data}(x)||P_G(x)\right)$$

0 <　　< log 2

接下来就是一个简单的最小化JS散度的问题了

$$G^* = arg\min_G \max_D V(G,D)$$

实践中如何实现

最主要的问题就是怎么算V

$$V = E_{x\sim P_{data}}[logD(x)] + E_{x\sim P_G}[log(1-D(x))]$$

Sample $\{x^1, x^2, \dots, x^m\}$ from $P_{data}(x)$, sample $\{\tilde{x}^1, \tilde{x}^2, \dots, \tilde{x}^m\}$ from generator $P_G(x)$

Maximize $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} logD(x^i) + \frac{1}{m}\sum_{i=1}^{m} log\left(1-D(\tilde{x}^i)\right)$

Algorithm

Initialize θ_d for D and θ_g for G

Can only find lower found of $\max_D V(G,D)$

- In each training iteration:

Learning D — Repeat k times

- Sample m examples $\{x^1, x^2, \dots, x^m\}$ from data distribution $P_{data}(x)$
- Sample m noise samples $\{z^1, z^2, \dots, z^m\}$ from the prior $P_{prior}(z)$
- Obtaining generated data $\{\tilde{x}^1, \tilde{x}^2, \dots, \tilde{x}^m\}$, $\tilde{x}^i = G(z^i)$
- Update discriminator parameters θ_d to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} logD(x^i) + \frac{1}{m}\sum_{i=1}^{m} log\left(1 - D(\tilde{x}^i)\right)$
  - $\theta_d \leftarrow \theta_d + \eta \nabla \tilde{V}(\theta_d)$

Learning G — Only Once

- Sample another m noise samples $\{z^1, z^2, \dots, z^m\}$ from the prior $P_{prior}(z)$
- Update generator parameters θ_g to minimize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} logD(x^i) + \frac{1}{m}\sum_{i=1}^{m} log\left(1 - D\left(G(z^i)\right)\right)$
  - $\theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)$

真正在实现的时候有一个小Trick

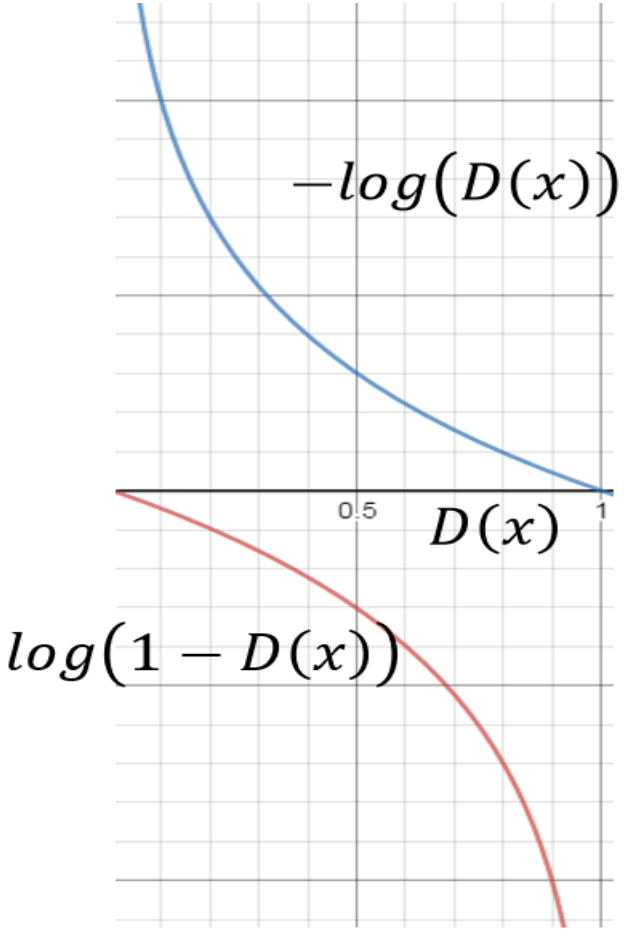
$$V = E_{x \sim P_{data}}[logD(x)]$$

$$+E_{x \sim P_G}[log(1 - D(x))]$$

Slow at the beginning

$$V = E_{x \sim P_G}[-log(D(x))]$$

Real implementation:
label x from P_G as positive



建议完成的事情

1.用 Tensorflow 实现一个CNN完成任何一个任务

谢谢！