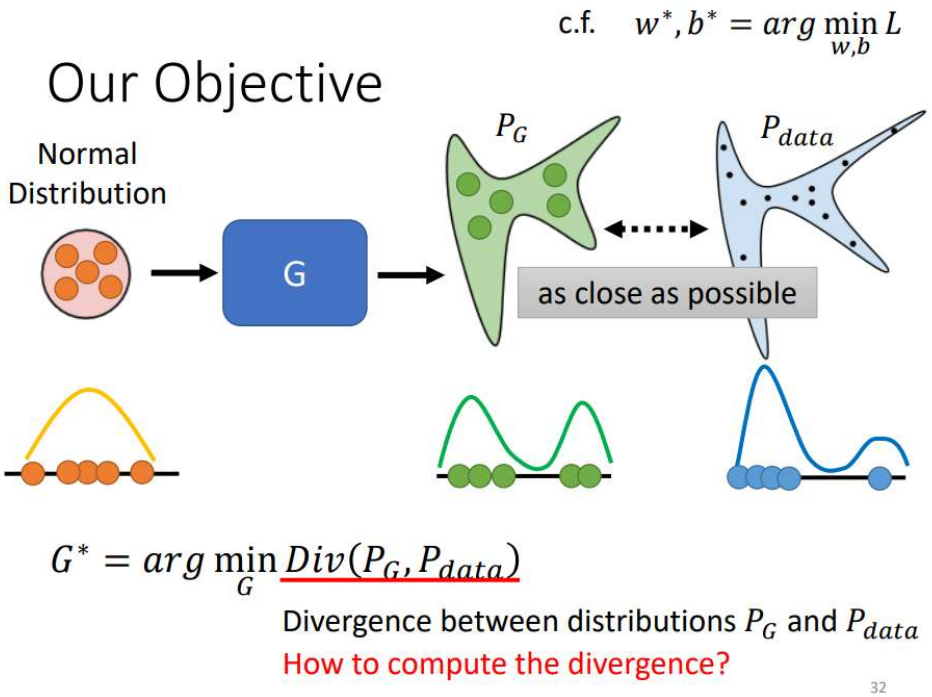
生成式對抗網路 (Generative Adversarial Network, GAN) (二) – 理論介紹與 WGAN

Create at 2022/06/22

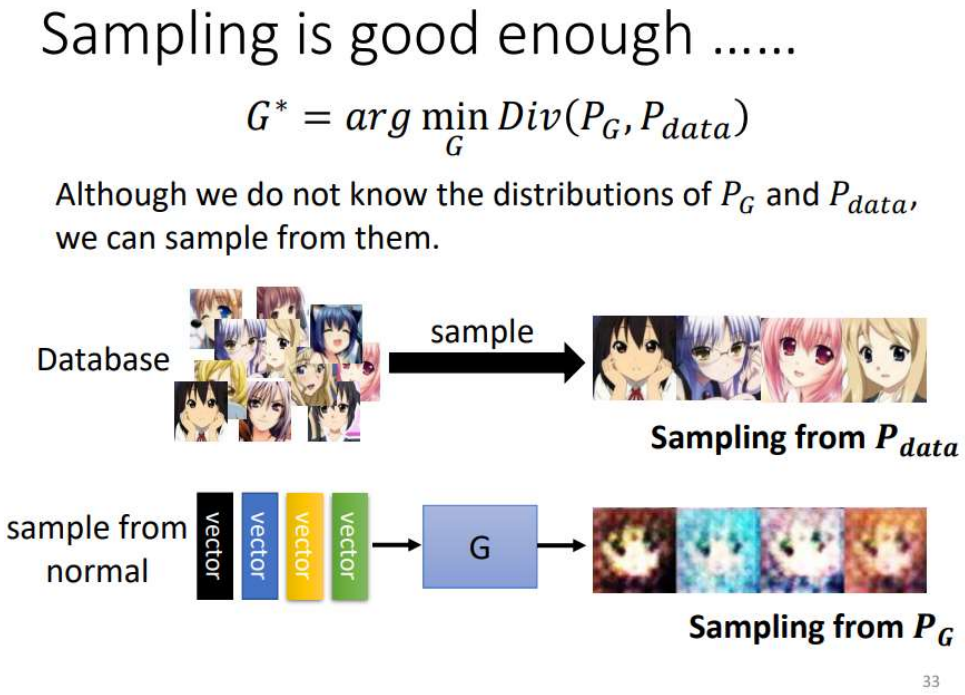
- 上課資源：
  1. 生成式對抗網路 (Generative Adversarial Network, GAN) (二) – 理論介紹與 WGAN (https://www.youtube.com/watch?v=jNY1WBb8I4U)

GAN 理論介紹

- 在訓練 network 的時候，要定一個 Loss Function
- 定完之後用 Gradient Descent 去調參數
- 去 minimize Loss Function 之後結束



- 在 Generation 問題裡面，我們要 minimize 或 maximize 的是什麼？
  - 有一個 Generator 給它一堆 vector (從 normal distribution sample 出來的東西)，丟進 Generator
  - 會產生一個比較複雜的 Distribution 稱為 P_G
  - 真正的 data 也形成另外一個 Distribution 稱為 P_{data}
  - 我們希望 P_G 跟 P_{data} 越接近越好
- 寫成式子的話 $Div(P_G, P_{data})$
  - 就是 P_G 跟 P_{data} 這兩個 distribution 之間的 Divergence
- Divergence : 兩個 distribution 之間的某種距離
  - 如果越大，代表這兩個 distribution 越不像
  - 如果越小，代表這兩個 distribution 越相近
- 目標 :
  - 找一個 Generator 裡面的參數，因為 Generator 也是一個 network，裡面有一堆的 weight 跟 bias，可以讓我們產生出的 P_G 跟 P_{data} 之間的 Divergence 越小越好

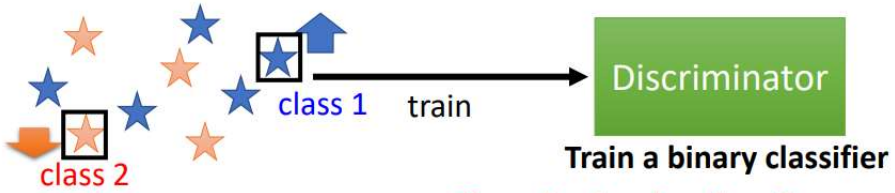


- Divergence 要怎麼算？
  - GAN 告訴我們，只要知道怎麼從 P_G 跟 P_{data} 這兩個 distribution sample 東西出來，就有辦法算 divergence
  - 而 P_G 跟 P_{data} 是可以 sample 的
  - 如何從真正的 data 裡面 sample 出東西來呢？
    - 把 database 拿出來，從 database 裡面隨機產生 sample 一些圖片出來，就得到 P_{data}
  - 如何從 generator 裡面產生東西出來？
    - 把 generator
      - 輸入：從 normal distribution sample 出來的 vector
      - 輸出：產生一堆圖片出來 (這些圖片就是從 P_G sample 出來的結果)
- 有辦法從 P_G 做 sample，就有辦法從 P_{data} 做 sample

https://arxiv.org/abs/1406.2661

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

★ : data sampled from P_{data} ★ : data sampled from P_G



Train a binary classifier

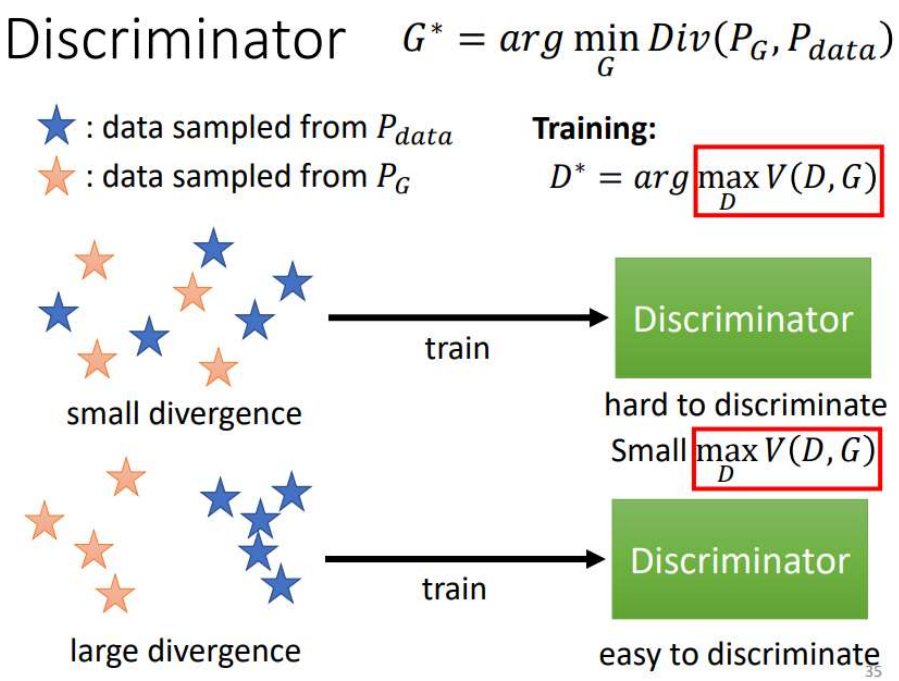
Training: $D^* = arg \max_D V(D,G)$ The value is related to JS divergence.

Objective Function for D

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

$D^* = arg \max_D V(D,G)$ negative cross entropy = Training classifier: minimize cross entropy

- 要靠 discriminator
  - 有一堆 Real data : 一堆從 P_{data} sample 出來的結果
  - 有一堆 Generative data : 一堆從 P_G sample 出來的結果
  - 根據 Real data 跟 Generative data 去訓練一個 discriminator
  - 訓練目標
    - 看到 real data 給比較高的分數
    - 看到 generative data 給比較低的分數
- 希望 Objective Function V 越大越好
  - 意味希望 P_{data} 的 $D(y)$ 越大越好
  - 希望 P_G 的 $D(y)$ 越小越好
- 事實上這個 Objective Function 就是 Cross Entropy 乘上一個負號
  - 在訓練 classifier 的時候，要 minimize cross entropy
  - 當 maximize objective function (maximize cross entropy 乘一個負號) = minimize cross entropy
  - 等於訓練一個 classifier
- Discriminator
  - 把 P_{data} sample 出來的 real image 分類為 class 1
  - 把 P_G sample 出來的假的 image 分類為 class 2
  - 訓練一個 binary classifier，等於解一個 optimization 問題
- 紅框的 Objective function 最大值，它跟 divergence 是有關的



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

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

The maximum objective value is related to JS divergence.

- Initialize generator and discriminator
- In each training iteration:

  Step 1: Fix generator G, and update discriminator D

  Step 2: Fix discriminator D, and update generator G

36

- 原本的目標是要找一個 Generator 去 minimize P_G 跟 P_{data} 的 Divergence
- 但我們卡在不知道怎麼計算 Divergence
- 我們現在知道，只要訓練一個 Discriminator，訓練完之後 objective function V 的最大值，就是 divergence
- 所以考慮把 紅框 跟 Divergence 做替換
- 找一個 G 讓紅框的值最小，這個 G 就是我們要的 Generator

Can we use other divergence?

Name	$D_f(P\|Q)$	Generator $f(u)$
Total variation	$\frac{1}{2}\int \lvert p(x)-q(x)\rvert \,\mathrm{d}x$	$\frac{1}{2}\lvert u-1\rvert$
Kullback-Leibler	$\int p(x)\log\frac{p(x)}{q(x)}\,\mathrm{d}x$	$u\log u$
Reverse Kullback-Leibler	$\int q(x)\log\frac{q(x)}{p(x)}\,\mathrm{d}x$	$-\log u$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)}\,\mathrm{d}x$	$(u-1)^2$
Neyman χ^2	$\int \frac{(p(x)-q(x))^2}{q(x)}\,\mathrm{d}x$	$\frac{(1-u)^2}{u}$
Squared Hellinger	$\int \left(\sqrt{p(x)}-\sqrt{q(x)}\right)^2\,\mathrm{d}x$	$(\sqrt{u}-1)^2$
Jeffrey	$\int (p(x)-q(x))\log\left(\frac{p(x)}{q(x)}\right)\,\mathrm{d}x$	$(u-1)\log u$
Jensen-Shannon	$\frac{1}{2}\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x$	$-(u+1)\log\frac{1+u}{2}+u\log u$
Jensen-Shannon-weighted	$\int p(x)\pi\log\frac{p(x)}{\pi p(x)+(1-\pi)q(x)}+(1-\pi)q(x)\log\frac{q(x)}{\pi p(x)+(1-\pi)q(x)}\,\mathrm{d}x$	$\pi u\log u-(1-\pi+\pi u)\log(1-\pi+\pi u)$
GAN	$\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x-\log(4)$	$u\log u-(u+1)\log(u+1)$

Using the divergence you like ☺

https://arxiv.org/abs/1606.00709

Name	Conjugate $f^*(t)$
Total variation	t
Kullback-Leibler (KL)	$\exp(t-1)$
Reverse KL	$-1-\log(-t)$
Pearson χ^2	$\frac{1}{4}t^2+t$
Neyman χ^2	$2-2\sqrt{1-t}$
Squared Hellinger	$\frac{t}{1-t}$
Jeffrey	$W(e^{1-t})+\frac{1}{W(e^{1-t})}+t-2$
Jensen-Shannon	$-\log(2-\exp(t))$
Jensen-Shannon-weighted	$(1-\pi)\log\frac{1-\pi}{1-\pi e^{t/\pi}}$
GAN	$-\log(1-\exp(t))$

37

- 可以用其他的 divergence

- GAN 以不好 train 而聞名

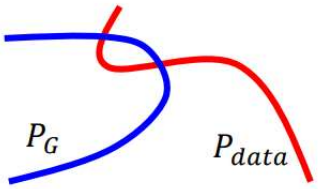
訓練 GAN 的小技巧

JS divergence is not suitable

- In most cases, P_G and P_{data} are not overlapped.
- 1. The nature of data

  Both P_{data} and P_G are low-dim manifold in high-dim space.

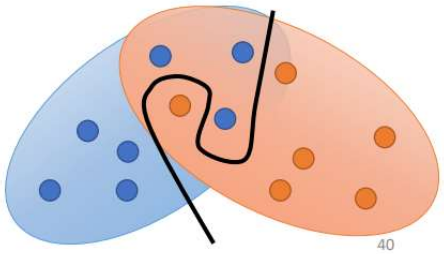
  The overlap can be ignored.



- 2. Sampling

  Even though P_{data} and P_G have overlap.

  If you do not have enough sampling .....

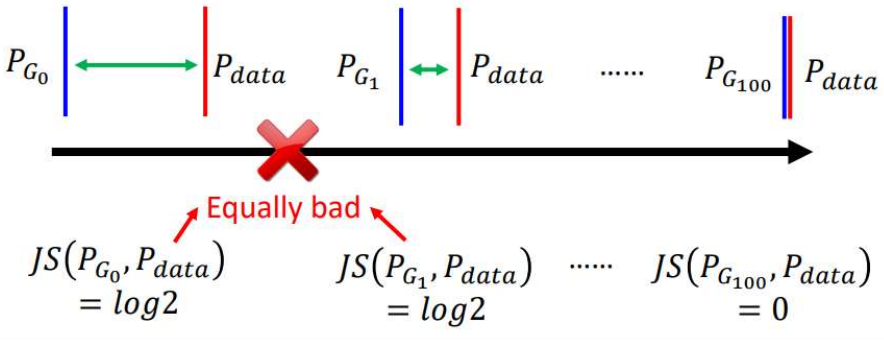


40

- 最知名的 WGAN
- JS divergence 有什麼問題？
  - 先看 P_G 跟 P_{data} 有什麼特性？
    - P_G 跟 P_{data} 重疊的部分非常少
      1. 來自於 data 本身的特性，P_G 跟 P_{data} 都是要產生圖片，圖片是高維空間裡面的一個低維的 manifold (因為在高維空間隨便 sample 一個點，通常都沒辦法構成一個二次元人物的頭像)，所以二次元人物的頭像分佈在高維空間中是非常狹窄的
      2. 我們從來不知道 P_G 跟 P_{data} 長什麼樣子，如果我們沒有 sample 很多的點或是 sample 的點不夠密，可以畫一條線當作楚河漢界

What is the problem of JS divergence?

JS divergence is always log2 if two distributions do not overlap.



Intuition: If two distributions do not overlap, binary classifier achieves 100% accuracy.

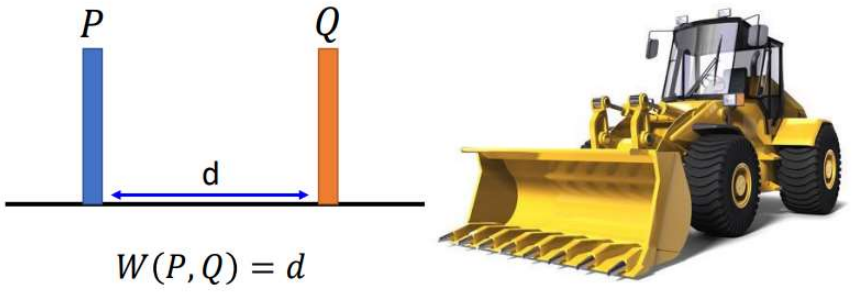
The accuracy (or loss) means nothing during GAN training.

41.

- 幾乎沒有重疊會對 JS divergence 造成什麼問題？
  - JS divergence 的特性，兩個沒有重疊的分佈，JS divergence 算出來會永遠都是 log2
  - 不管距離多遠，只要沒有相交，JS divergence 算出來就是 log2
- 實際直觀的角度說明
  - 當你是用 JS divergence，假設今天在 train binary 的 classifier 去分辨 real 的 Image 跟 Generated Image
  - 會發現實際 train 完之後，正確率幾乎都是 100 %
  - 在實際操作的時候觀察不到這個現象，所以在 update 幾次 generator 之後，要把圖片 print 出來看

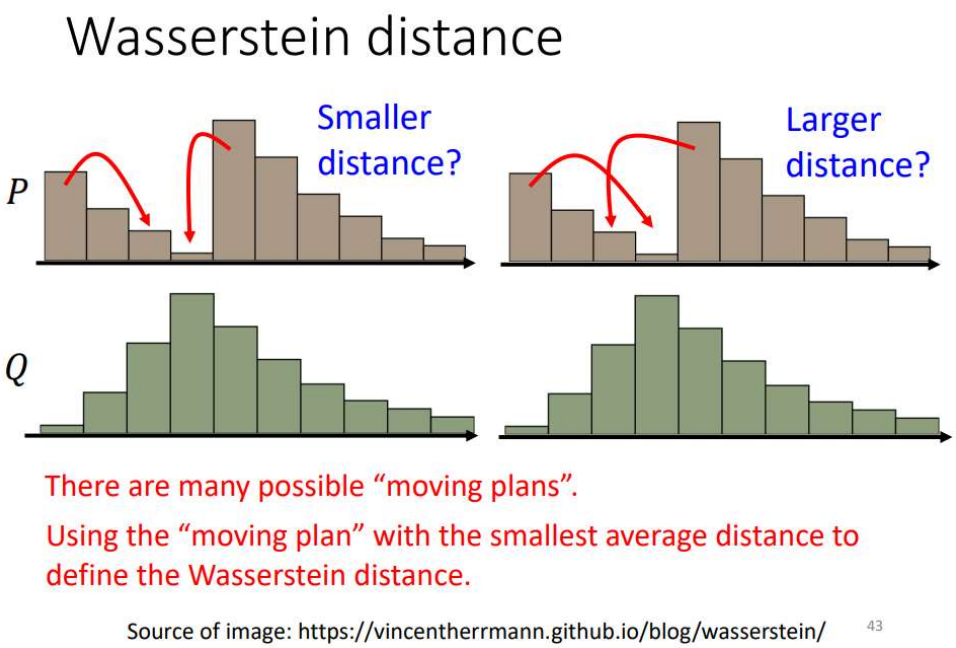
Wasserstein distance

- Considering one distribution P as a pile of earth, and another distribution Q as the target
- The average distance the earth mover has to move the earth.

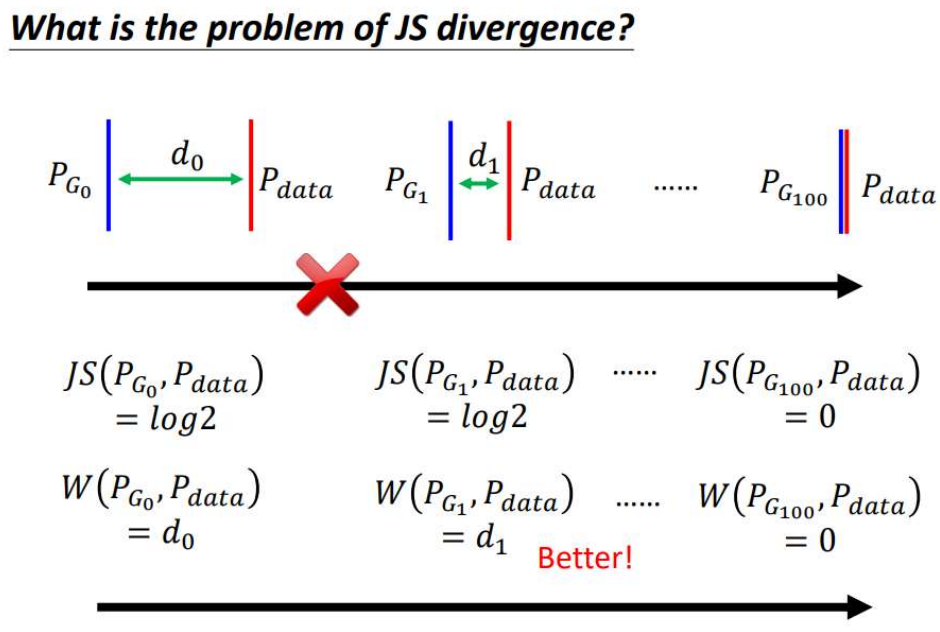


42

- 換一個衡量兩個 distribution 相似程度的方式，換一種 divergence 就可以解決這個問題？
  - **Wasserstein distance**
    - 假設有兩個 distribution P、Q
    - 把 P 移動到 Q 的平均距離 D，就是 wasserstein distance



- 開一台推土機，想把 P 塑造成與 Q 接近的形狀
- 把 P 塑造成與 Q 形狀的方法有無窮多種
- 但不同的 moving plan 移動的距離不同
- 為了讓 wasserstein distance 只有一個固定的值
- 所以這裡 wasserstein distance 的定義是窮舉所有的 moving plan，看哪一個 moving plan 讓移動的距離最小，最小的值才是 wasserstein distance



- 假設能夠計算 wasserstein distance 能帶給我們什麼好處？
  - 所以換一個 divergence 的計算方式，就能解決 JS divergence 帶來的問題

What is the problem of JS divergence?

P_{G_0} d_0 P_{data} P_{G_1} d_1 P_{data} ..... $P_{G_{100}}$ P_{data}

https://www.pnas.org/content/104/suppl_1/8567.figures-only

45

- 演化
- 當使用 wasserstein distance 來計算 divergence 會得到類似的效果
  - 本來 P_{G_0} 跟 P_{data} 距離很遙遠
  - 但只要每次稍微靠近一點點 wasserstein distance 就會有變化，所以才有辦法 train generator
- 對 JS divergence 而言，要從 P_{G_0} 到 $P_{G_{100}}$，它的 loss 才會有差異
- 這就是當我們從 JS divergence 換成 Wasserstein distance 帶來的好處

WGAN

https://arxiv.org/abs/1701.07875

WGAN

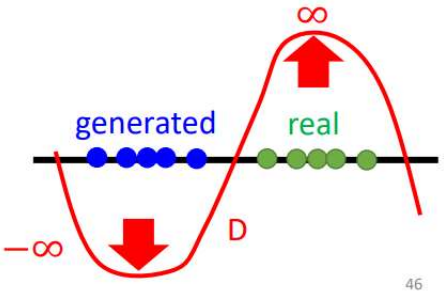
Evaluate Wasserstein distance between P_{data} and P_G

$$\max_{D \in 1-Lipschitz} \{E_{y \sim P_{data}}[D(y)] - E_{y \sim P_G}[D(y)]\}$$

D has to be smooth enough. How to fulfill this constraint?

Without the constraint, the training of D will not converge.

Keeping the D smooth forces D(y) become ∞ and −∞



46

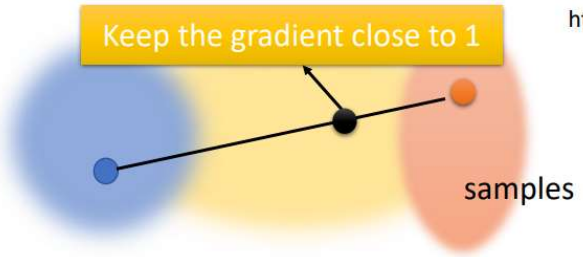
- P_G 跟 P_{data} 之間的 wasserstein distance 要怎麼計算？
  - $E_{y\sim P_{data}}[D(x)]$: y 如果是從 P_{data} 來的，計算它的 $D(x)$ 期望值
  - $-E_{y\sim P_G}[D(x)]$: y 如果是從 P_G 來的，計算它的 $D(x)$ 期望值，但前面加上一個負號
- 所以如果要 maximum objective function 會達成什麼效果
  - 如果 y 是從 P_{data} sample 出來的，discriminator output 越大越好
  - 如果 y 是從 P_G sample 出來的，discriminator output 越小越好
  - D 不能是一個隨便的 function，必須是一個 1-Lipschitz 的 function (足夠平滑的 function，不可以是變動很劇烈的 function)
    - 這個限制是要求 discriminator 不可以太變化劇烈，real data 的值跟 generated data 的值就沒辦法差很多，算出來的 wasserstein distance 就會比較小

$$\max_{D\in 1-Lipschitz}\{E_{y\sim P_{data}}[D(y)]-E_{y\sim P_G}[D(y)]\}$$

- Original WGAN → Weight

  Force the parameters w between c and -c

  After parameter update, if w > c, w = c; if w < -c, w = -c
- Improved WGAN → Gradient Penalty



https://arxiv.org/abs/1704.00028

- Spectral Normalization → Keep gradient norm smaller than 1 everywhere

https://arxiv.org/abs/1802.05957

47

- 實作 WGAN 有很多不同的做法
- 目前看起來效果最好的是 Spectral Normalization (SNGAN)

tags: **2022 李宏毅_機器學習**