Генеративно-состязательные сети

Денис Деркач

Лаборатория Lambda, ФКН ВШЭ, 19 февраля 2020 года

Содержание

Порождающее моделирование

Все модели являются порождающими моделями.

-Эрик Янг

Генеративное и дискриминативное моделирование

Дискриминативная модель

- приближает $\mathbb{P}(y|x)$;
- ищет границу между классами;
- пример: логистическая регрессия, SVM и т. д.

Генеративная модель

- приближает $\mathbb{P}(x|y)$ (а вообще $\mathbb{P}(x)$);
- ищет как классы были сгенерированы (порождены);
- пример: наивный байес, смесь Гауссов и т. д.

Разделение классов

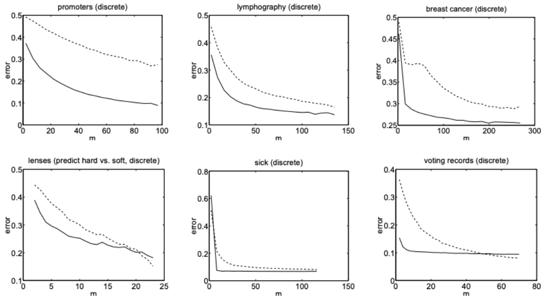


Figure 1: Results of 15 experiments on datasets from the UCI Machine Learning repository. Plots are of generalization error vs. m (averaged over 1000 random train/test splits). Dashed line is logistic regression; solid line is naive Bayes.

- дискримантивные модели выигрывают (почти всегда) для большого количества событий;
- генеративные — для маленького.

From A. Ng et al.

Типизация генеративных моделей

- ”Непараметрические”:
  - гистограмы;
  - ядерное сглаживание;
  - k-ближайших соседей.
- Параметрические с явным правдоподобием:
  - авторегресионные;
  - вариационные автокодировщики;
  - нормализующие потоки.
- параметрические бех явного правдоподобия:
  - Генеративно-состязательные сети.

Интуиция генеративных моделей

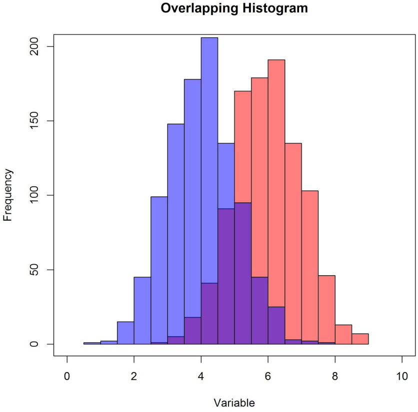
Поиск лучшей метрики

Обычные метрики не годятся, так как нам нужно получить распределение вероятностей.
Необходима метрика такая, что:

- хорошо сравнивает две плотности распределения вероятности;
- дифференциируемая;
- хорошо контролирующая сходимость.

Пусть $p(x)$ - искомое распределение, $q_\theta(x)$ - распределение, которое мы хотим приблизить.

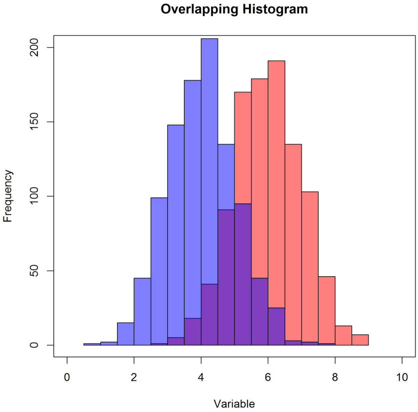
Расстояние между гистограммами: идея



Первая идея:

- Используем первое, что придёт в голову.
- Поточечное расстояние между распределениями.
- Расстояние = синяя область + красная область.

Расстояние полной вариации



Перепишем математическим языком:

$$D(p(x), q_\theta(x)) = \frac{1}{2} \int\limits_{\mathbb{R}^n} |p(x) - q_\theta(x)| dx,$$

для $x \in \mathbb{R}^n$.

Что на самом деле, равно расстоянию полной вариации (с учётом теоремы Шеффе).

Расстояние полной вариации

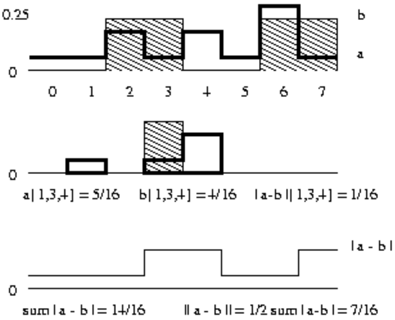
Расстояние полной вариации:

$$D(p(x), q_\theta(x)) = \sup_A \left| \int_A p(x)dx - \int_A q_\theta(x)dx \right|,$$

where $\sup$ берётся по всем измеримым A.

Расстояние полной вариации: 1D пример

- Распределения вероятностей a (сплошная) и b (затемнённая).
- Лучшее подмножество $x = \{1, 3, 4\}$. $P_a(x) = 5/16$, а $P_b(x) = 4/16$.
- $||a - b||$ даёт самую большую дистанцию из всех 256 возможных разбиений.
- $||a - b|| = 1/2 \sum |a - b|$



From GECCO-2002

Свойства

- Симметрична: $D(P,Q) = D(Q,P)$.
- Связана с тестированием гипотез: $1 - D(P,Q)$ – сумма ошибок.
- При росте количества экспериментов $D(f_{X^n}, g_{Y^n}) \to 1$.

Проблема: может игнорировать некоторые сходимости.
Например, при $X_1, \ldots, X_n \sim \pm 1$, $S_n = \sum_n X_i$, имеем

$$S_n/\sqrt{n} \to \mathcal{N}(0,1),$$

но $D(S_n, Z) = 1$ для любого n.

Дивергенция Кульбака-Лейблера: определение

Для двух распределений вероятности $p(x)$ и $q(x)$

$$KL(P||Q) = \int_{\mathbb{R}^n} p(x) \log\left(\frac{p(x)}{q(x)}\right) dx.$$

NB: хотя иногда мы говорим расстояние, оно не симметрично, так что не является настоящей метрикой.

Свойства КЛ дивергенции

- не симметрична $KL(P||Q) \neq KL(Q||P)$;
- инварианта при преобразованиях переменных;
- складывается для независимых случайных переменных;.
- подчиняется цепное правило $KL(p_{X^n}, p_{Y^n}) = nKL(p_X, p_Y)$;
- Оптимальная точка КЛ дивергенции совпадает с оценкой максимального правдоподобия.

Кросс-Энтропия и КЛ дивергенция

Для двух распределений $p(x)$ и $q(x)$, кросс-энтропия определена:

$$H(p, q) = \mathbb{E}_p(\log q)$$

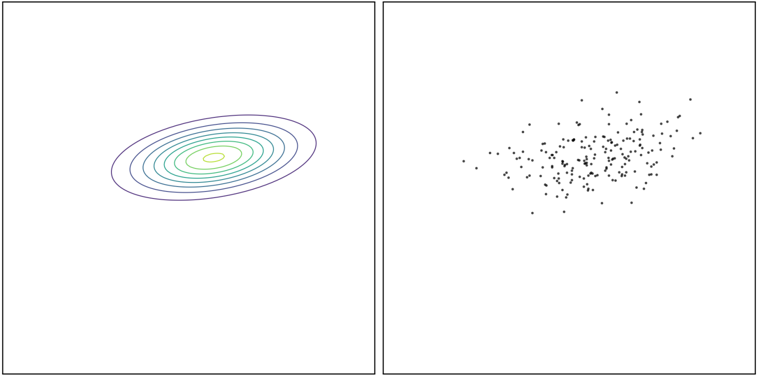
КЛ дивергенция связана с ней напрямую:

$$KL(p, q) = H(p, p) + H(p, q).$$

Так как обычно мы оптмтимизируем функцию правдоподобия $L(\theta) = H(p_{data}, q(x))$, то оптимизация $KL \leftrightarrow$ оптимизациия H.

Свойства сходимости

У нас нет доступа к $p(x)$, потому мы насемплируем оттуда:



Здесь мы будем использовать 2D Гаусс как q_θ.

Пример мотивирован блогом Колина Раффеля.

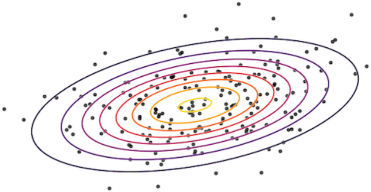
Оптимизация параметров

Необходимо найти оптимальный θ^*. Сделаем это минимизируя КЛ дивергенцию

$$\begin{aligned}
\theta^* &= \arg\min_\theta KL(p(x)||q_\theta(x)) = \\
&= \arg\min_\theta (\mathbb{E}_{x\sim p}[\log p(x)] - \mathbb{E}_{x\sim p}[\log q_\theta(x)]) \\
&\quad \text{since } p(x) \text{ does not depend on } \theta = \\
&= \arg\min_\theta -\mathbb{E}_{x\sim p}[\log q_\theta(x)] = \\
&= \arg\max_\theta \mathbb{E}_{x\sim p}[\log q_\theta(x)].
\end{aligned}$$

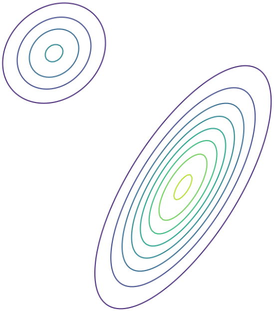
то есть мы хотим найти θ^* которая даёт $p(x)$ самую высокую $\log$ вероятность при $q_{\theta^*}(x)$.

Сходимость с КЛ



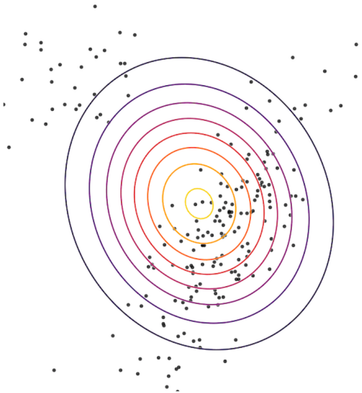
- Всё сошлось!
- Означает ли это, что все проблемы решены?

Сходимость с КЛ: мультимодальный случай



- Всё сошлось!
- Означает ли это, что все проблемы решены?
- Нет, у нам могут быть проблемы с мультимодальностью.

Converging with KL: multimodal case intuition



- Неожиданности нет, так как мы оптимизируем:

$$\arg\max_{\theta} \mathbb{E}_{x\sim p}[\log q_\theta(x)]$$

- Если носитель $q_\theta(x)$ не находится там, где есть $x \sim p(x)$ то функция будет ∞.
- Автоматически, мы получаем ненулевую $q_\theta(x)$ в "белых пятнах", где нет $x \sim p(x)$.

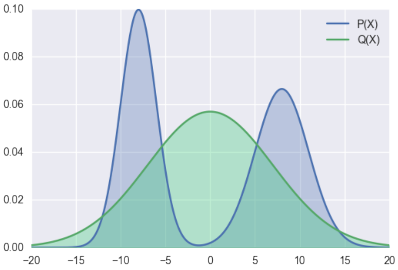
Обратная КЛ дивергенция

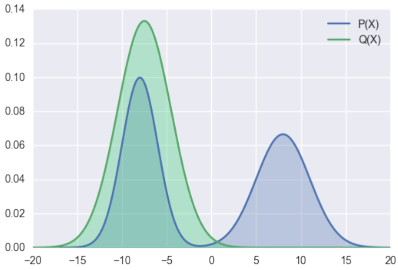
Чтобы преодалеть проблему, мы можем инвертировать КЛ дивергенцию:

$$rKL(q_\theta||p) = \int_{\mathbb{R}^n} p(x) \log\left(\frac{p(x)}{q_\theta(x)}\right) dx.$$

Интуиция

$$KL = \int p(X) \log \frac{p(x)}{q_\theta(x)} dx \qquad rKL = \int q(x) \log \frac{q_\theta(x)}{p(x)} dx$$





Прямая КЛ дивергенция избегает мест, где $q(x) = 0$, если $P(x) > 0$.

Обратная KL дивергенция форсирует $q(X) = 0$ в некоторых местах, даже если $P(X) > 0$.

Картинка credit:https://wiseodd.github.io/techblog/2016/12/21/forward-reverse-kl/

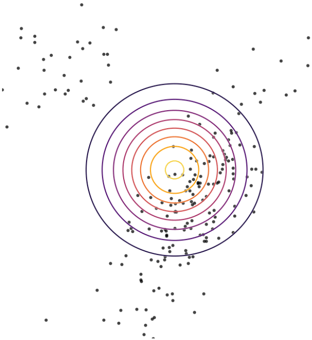
Обратная КЛ: оптимизация

Мы получим очень похожую вещь:

$$\begin{aligned}
\theta^* &= \arg\min_\theta KL(q_\theta(x)||p(x)) = \\
&= \arg\min_\theta (\mathbb{E}_{\tilde{x}\sim q_\theta}[\log q_\theta(x)] - \mathbb{E}_{\tilde{x}\sim q_\theta}[\log p(x)]) = \\
&= \arg\max_\theta (-\mathbb{E}_{\tilde{x}\sim q_\theta}[\log q_\theta(x)] + \mathbb{E}_{\tilde{x}\sim q_\theta}[\log p(x)])
\end{aligned}$$

- Решена проблема функции правдоподобия, которая уходит в бесконечность в ненужных местах.
- Первое слагаемое похоже на энтропию порождающей модели.
- Второе слагаемое штрафует распределения, которые не похожи на искомое.

Converging with rKL



- $q_\theta(x)$ больше нет в местах без $x \sim p(x)$.
- Распределение выглядит хорошо для одного решения.

Основная проблема, в оптимизируемую величину входит $p(x)$

$$\arg\max_\theta(-\mathbb{E}_{\tilde{x}\sim q_\theta}[\log q_\theta(x)] + \mathbb{E}_{\tilde{x}\sim q_\theta}[\log p(x)]),$$

о котором мы ничего не знаем.

Дивергенция Йенсена-Шеннона

Мы можем поменять дивергенции, в (безуспешных) попытках избежать проблем с мультимодальностью. Одно из интересных попыток является смесь прямой и обратной КЛ дивергенций, назаывемых дивергенцией Йенсена-Шеннона

$$
\begin{aligned}
JS(p(x)||q_\theta(x)) = & \quad \tfrac{1}{2} \quad KL(p(x)||\frac{p(x)+q_\theta(x)}{2}) + \\
+ & \quad \tfrac{1}{2} \quad KL(q_\theta(x)||\frac{p(x)+q_\theta(x)}{2}),
\end{aligned}
$$

It is symmetric and does not ignore zeroes like KL and does not ignore x like rKL.

ЙШ: свойства

- симметрична;
- $JS(p(x)||q_\theta(x)) \leq \mathrm{const}$;
- $\sqrt{JS}$ - настоящая метрика.

К сожалению, мы не можем напрямую её оптимизировать так как нам понадобится доступ к $p(x)$.

Генеративно-состязательные сети

Идея

Можем ли мы построить алгоритм, который выполнял предыдущее упражнение, но без прямого доступа к $p(x)$? Возможная оптимизация выглядела бы вот так:

$$\theta^* = \arg\min_\theta \max_\phi \mathbb{E}_{x \sim p, \tilde{x} \sim q_\theta} V(f_\phi(x), f_\phi(\tilde{x}))$$

То есть, вместо того, чтобы минимизировать аналитическую дивергенцию, мы могли бы оптимизировать ”выученную дивергенцию”.

Генератор

Мы семплировали события из $q(x)$. В реальности, мы использовали обратный семплинг. Можно его переписать так:

$$z_j \sim \mathcal{N}(0;1),$$
$$\hat{x}_j = G_\theta(z_j)$$

где $G_\theta(z_j) : z_j \mapsto x_j$ может быть любой, но мы рассмотрим только нейронные сети.
Мы получим:

$$\{\tilde{x}_j\} \sim q_\theta(x).$$

Дискриминатор

У нас теперь есть два семпла, потому мы можем построить другую сеть, дискриминатор D_ϕ, который различает между настоящими и сгенерированными семплами:

$$\max_\phi \left(\mathbb{E}_{x \sim p(x)} (\log(D_\phi(x)) + \mathbb{E}_{\tilde{x} \sim q_\theta(x)} (1 - \log(D_\phi(\tilde{x}))) \right).$$

Первое слагаемое служит для распознавания настоящих изображений. Второе для характеризации сгенерированных.

G+D

Давайте сложим вместе генератор и дискриминатор.

- цель дискриминатора:

$$\max_{\phi} \Big(\mathbb{E}_{x\sim p(x)}(\log(D_\phi(x)) + \mathbb{E}_{z\sim\mathcal{N}(0;1)}(1 - \log(D_\phi(G_\theta(z)))\Big) .$$

- цель генератора:

$$\min_{\theta} \mathbb{E}_{z\sim\mathcal{N}(0;1)}(1 - \log(D_\phi(G_\theta(z))$$

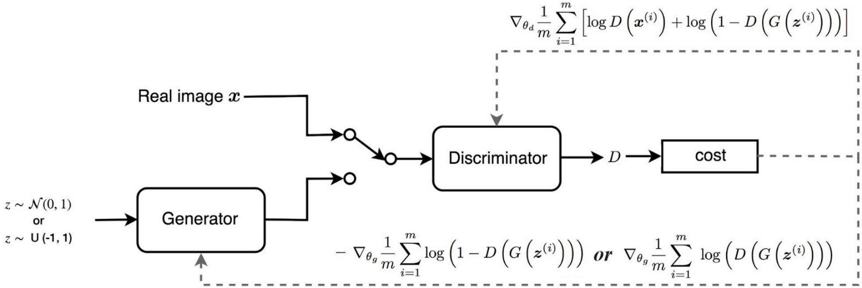
Мы получили mimimax игру:

$$\min_{\theta} \max_{\phi} \mathbb{E}_{x\sim p, \tilde{x}\sim q_\theta} V(f_\phi(x), f_\phi(\tilde{x})),$$

именно то, чего мы хотели.

Графическое отображение

Так как мы использовали нейронные сети и для генератора и для дискриминатора:



мы сможем использовать обратное распространение ошибки.

Оптимальное решение

- Если мы зафиксируем генератор, оптимальный дискриминатор выглядит так:

$$D^*_\phi(G) = \frac{p(x)}{p(x) + q_\theta(x)}.$$

- Подставим в minimax игру:

$$\begin{aligned} C(G) \quad &= \max_D V(G, D) = \\ = \quad &\mathbb{E}_{x\sim p(x)}(\log(D^*_\phi(x))) + \mathbb{E}_{x\sim q_\theta}(1 - \log(D^*_\phi(G_\theta(z)))) = \\ = \quad &\mathbb{E}_{x\sim p(x)} \log \frac{p(x)}{p(x) + q_\theta(x)} + \mathbb{E}_{x\sim q_\theta} \log \frac{q_\theta(x)}{p(x) + q_\theta(x)} \end{aligned}$$

Оптимальное решение

- В оптимальном случае, $p = q_\theta$, у нас получится $C(G) = -\log(4)$.
- То есть мы можем написать:

$$C(G) = -\log(4) + \frac{1}{2}KL(p(x)||\frac{p(x)+q_\theta(x)}{2}) + \\ +\frac{1}{2}KL(q_\theta(x)||\frac{p(x)+q_\theta(x)}{2}).$$

- Другими словами, подобной игрой мы оптимизируем ЙШ дивергенцию:

$$C(G) = -\log(4) + JS(p(x)||q_\theta(x)).$$

- Напоминаю: мы это сделали без доступа к $p(x)$.
- Вообще, мы можем специально коснтруировать игру под любую дивергенцию.

Алгоритм GAN

1. Семплируем мини-батч из данных ($x_1, .., x_m \sim D$).
2. Семплируем мини-батч из генератора ($z_1, .., z_m \sim q_\theta$).
3. Уточняем параметры генератора по стохастическому градиентному спуску:

$$\nabla_\theta V(G_\theta, D_\phi) = \frac{1}{m} \sum_{i=1}^{m} \log(1 - D_\phi(G_\theta(z_i))).$$

4. Уточняем параметры дискриминатора по стохастическому градиентному подъёму:

$$\nabla_\phi V(G_\theta, D_\phi) = \frac{1}{m} \sum_{i=1}^{m} \left(\log D_\phi(x_i) + \log(1 - D_\phi(G_\theta(z_i)))\right).$$

5. Делаем несколько эпох обучения.

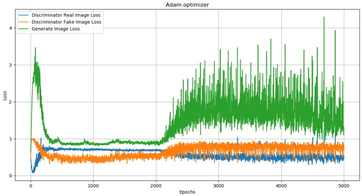
GAN: плюсы

- Плюсы:
  - Можем использовать обратное распространение ошибки.
  - Не занимаемся интегрированием.
  - Не нужно Марковских цепей.
- Минусы:
  - Нет явной формы $q_\theta(x)$.
  - Трудно тренировать.
  - Каждый раз новый дискриминатор, то есть метрика качества.
  - Легко попасть в локальный минимум.
  - Трудно обратить.

Но главные проблемы приходят из плюсов.

Трудности тренировки

- Вообще, сходимость гарантируется, если обновление генератора производится в функциональном пространстве, а дискриминатор оптимален на каждом шаге
- Далеко не так.
- Лосс-функция обычно довольно шумная.

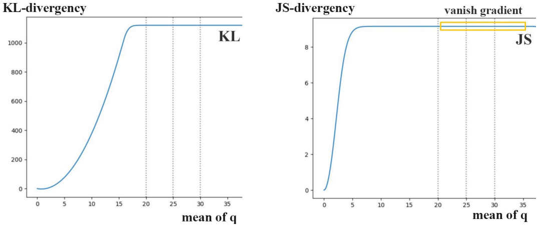


Нет чёткого правила остановки.

Больше проблем

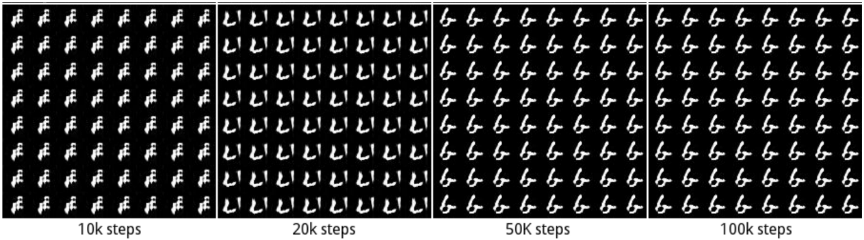
Мы оптимизируем дивергенцию ЙШ, что порождает свои трудности:

- Коллапс мод: как мы видели ранее на примере КЛ дивергенции.
- Исчезающие градиенты: если мы начнём слишком далеко, то никогда не придём к решению.



Всё из-за формы дивергенции, которую мы оптимизируем. Может быть, мы можем найти другую?

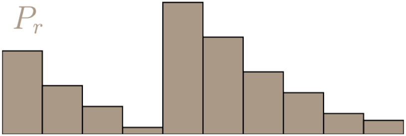
Пример: коллапс мод

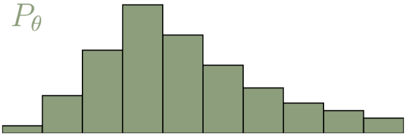


Дистанция Васерштейна

Мотивация

Представьте, что мы хотим перенести события из P_r в P_θ. Мы также хотим сэкономить усилия, то есть не перемещать большие части на большие расстояния.

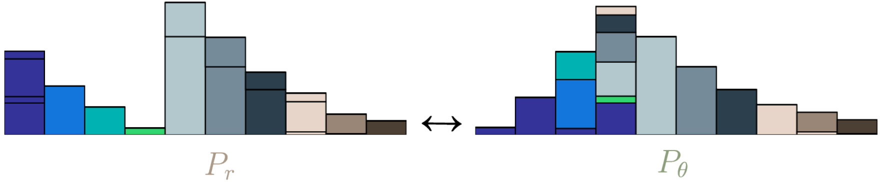




Это связано с проблемой, которая ежедневно решается многими землекопами. Фактически, это задача оптимального транспорта из P_r в P_θ.

Picture credit:https://vincentherrmann.github.io/blog/wasserstein/

Earth Mover's Distance



$$EMD(P_r, P_\theta) = \inf_{\gamma \in \Pi} \sum_{x,y} ||x - y|| \gamma(x, y) = \inf_{\gamma \in \Pi} \mathbb{E}_{(x,y)\sim\gamma} ||x - y||,$$

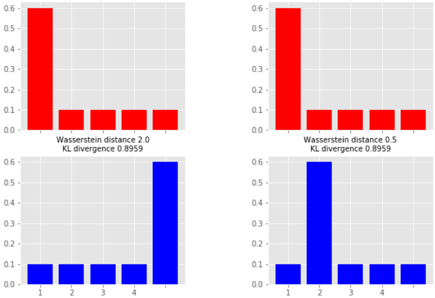
где $\gamma(x, y)$ усилия по перемещению x в y, Π — все возможные перемещения из P_r в P_θ, $\gamma \in \Pi$.

Мы легко можем переписать на случай непрерывных переменных (построим метрику Васерштейна).

Picture credit:https://vincentherrmann.github.io/blog/wasserstein/

W vs KL

EMD также учитывает расстояние, на котором находятся различия в распредеелниях.
Это как раз то, что нужно учитывать при мультимодальности!



Picture credit:https://goo.gl/ncx3gt

Дуальность Канторовича-Рубинштейна

Метрику EMD (или расстояние Вассерштейна), приведенную выше, довольно сложно вычислить на практике, к счастью, у нас есть двойственность этой метрики:

$$W(p_r, p_\theta) = \sup_{f \in \mathrm{Lip}_1(\mathrm{X})} \left(\mathbb{E}_{x \sim p_r}[f(x)] - \mathbb{E}_{x \sim p_\theta}[f(x)] \right),$$

где $\mathrm{Lip}_1(\mathrm{X})$: $|f(x) - f(y)| \le d(x, y)$.
Мы можем ограничить веса нейронной сети (clipped weights) чтобы она удовлетворяла $\mathrm{Lip}_1(\mathrm{X})$ условию.
То есть мы можем сконструировать GAN, который оптимизирует W.

WGAN

Лосс-функция для Wasserstein-GAN

$$\mathcal{L}(p, q) = W(p, q) = \max_{w \in W} \mathbb{E}_{x \sim p}[f_w(x)] - \mathbb{E}_{z \sim q(z)} f_w(g_\theta(z)).$$

- Дискриминатор теперь помогает считать метрику W.
- На каждом шаге его можно тренировать до сходимости.

WGAN vs JSGAN

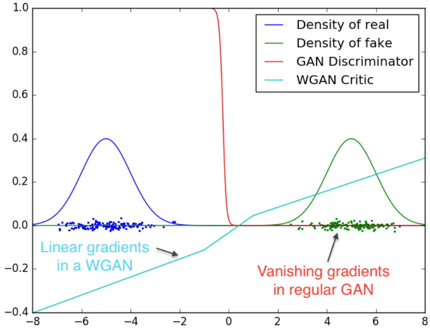
	Discriminator/Critic	Generator
GAN	$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right]$	$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$
WGAN	$\nabla_w \frac{1}{m} \sum_{i=1}^{m} \left[f(x^{(i)}) - f(G(z^{(i)}))\right]$	$\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} f(G(z^{(i)}))$

- WGAN проще тренировать.
- оптимизируется W, которая имеет лучшие свойства, чем ЙШ.

WGAN: решённые проблемы

- проблема коллапса мод минимализирована;
- исчезающие градиенты решены.
- вместо фиксирования весов, мы можем вводить штраф на градиенты (WGAN-GP):

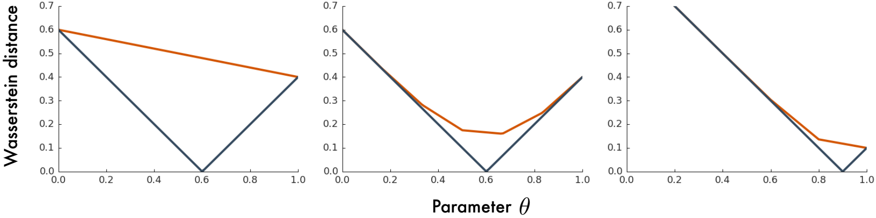
$$GP = \lambda \mathbb{E}\left[(||\nabla f|| - 1)^2\right]$$



From: arXiv:1701.07875

WGAN: новые проблемы

- оценка градиентов EMD может отличаться от настоящего направления на миминимум.
- эта проблема видна даже при оптимиации данных из распределения Бернулли.
- Решение: вместо W использовать энергетические расстояния.



Красный - ожидание градиента в семпле, синий - настоящие градиенты. Слева направо $\theta^* = 0.6; 0.6; 0.9$.

From: arXiv:1705.10743

Истрия GAN

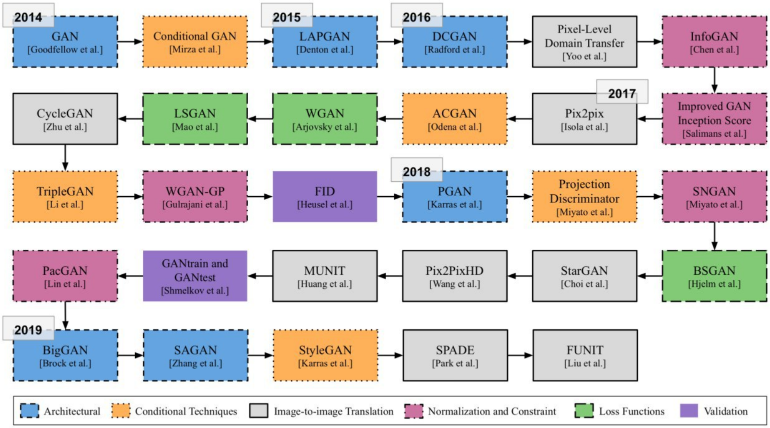


Fig. 1: Timeline of the GANs covered in this paper. Just like our text, we split it in six fronts (architectural, conditional techniques, normalization and constraint, loss functions, image-to-image translation and validation metrics), each represented by a different color and a different line/border style.

From: arXiv:1910.13076

Саммари

- GAN использует игру Генератор-дискриминатор для оценки расстояния от сгенерированного распределения до истинного.
- Wasserstein GAN - полезная методика, позволяющая использовать действительно глубокие сети для генерации данных.
- История ещё не закончилась.