From GAN to Wasserstein GAN

Peng Xu

Columbia University in the City of New York

February 11, 2020

Introduction

- Given a labeled data-set $\{(y_1, \boldsymbol{X}_1), \ldots, (y_n, \boldsymbol{X}_n)\}$ or an unlabeled data-set $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$, it would be useful to learn the data generating distributions, i.e. $P(\boldsymbol{X} \mid Y = y)$, $P(\boldsymbol{X}, y)$, and simply $P(\boldsymbol{X})$.
- We can use them to make classifications, or even generate new data instances.
- Models that go directly after the distribution of the data is called *generative*. Some of the classical generative learning models includes: Naïve Bayes, LDA, QDA, etc.
- The *Generative Adversarial Network* (GAN, [Goo+14]) is a generative model that utilizes neural networks to learn the data distribution and generate new samples. Due to its promising performance, GAN has became extremely popular in the field of machine learning since its introduction.

Introduction

- The idea of GAN is to train not one, but two models simultaneously: a discriminator D and a generator G. Then let them participate in an *adversarial game*.
- The generator G is trained to generate samples to fool the discriminator, and the discriminator D is trained to distinguish between real data and fake samples generated by G.
- Through the "competition" between these two players, we hope that their performance will improve and eventually becomes satisfactory.
- In the language of game theory, we train D and G until they reach a *Nash equilibrium*.

Introduction

- To be more specific, the generator $G(\boldsymbol{z}; \boldsymbol{\theta})$ is a differentiable function that takes random noises $\boldsymbol{z}$ as input, and produces a sample in the data space $\mathcal{X}$ as output.
- The noises $\boldsymbol{z}$ are the latent variables associated with the data. This setup is analogous to the inverse transform sampling: if we know the CDF F_X for some X, then we can sample $F_X^{-1}(U) \sim X$ with $U \sim \mathcal{U}[0, 1]$.
- The discriminator $D(\boldsymbol{X}; \boldsymbol{\delta})$, on the other hand, takes a sample $\boldsymbol{X}$ and returns the score (probability) that $\boldsymbol{X}$ came from the real data rather than G.
- Both D and G in the original GAN are multilayer perceptrons. A more popular architecture, *deep convolutional generative adversarial network* (DCGAN, [RMC15]), replaced them by deep CNNs.

Objective of GAN

- Hereinafter, we will denote the noise distribution as $\boldsymbol{Z}$, the real data distribution as $\boldsymbol{X}_R$, and the approximated data distribution (by the generator) as $\boldsymbol{X}_G$.
- Formally, the training of GAN is formulated as a *minimax problem*:

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\delta}} V_{\boldsymbol{\delta},\boldsymbol{\theta}}(D, G),$$

where

$$V_{\boldsymbol{\delta},\boldsymbol{\theta}}(D, G) := \mathrm{E}_{\boldsymbol{X}_R}\left[\ln D(\boldsymbol{X}\,;\,\boldsymbol{\delta})\right] + \mathrm{E}_{\boldsymbol{Z}}\left[\ln\left(1 - D\left[G(\boldsymbol{z}\,;\,\boldsymbol{\theta})\,;\,\boldsymbol{\delta}\right]\right)\right].$$

- The next slide will present an alternating algorithm for solving this minimax problem.

Algorithm of GAN

Input: A noise prior $\boldsymbol{Z}$ and a data generating distribution $\boldsymbol{X}_R$;
for $t \leftarrow 1, \ldots, T$ **do**
  for $k \leftarrow 1, \ldots, K$ **do**
    Sample minibatch of noise $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ from $\boldsymbol{Z}$;
    Sample minibatch of data $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_m\}$ from $\boldsymbol{X}_R$;
    Update the discriminator D by stochastic gradient ascend:

$$\boldsymbol{\delta} \leftarrow \boldsymbol{\delta} + \alpha \nabla_{\boldsymbol{\delta}} \frac{1}{m} \sum_{i=1}^{m} \left[\ln D(\boldsymbol{X}_i) + \ln\left(1 - D\left[G(\boldsymbol{z}_i)\right]\right)\right]$$

  end
  Sample minibatch of noise $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ from noise prior $\boldsymbol{Z}$;
  Update the generator G by stochastic gradient descend:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \frac{1}{m} \sum_{i=1}^{m} \ln\left(1 - D\left[G(\boldsymbol{z}_i)\right]\right)$$

end

Algorithm 1: Minibatch training of generative adversarial nets.

Algorithm of GAN

- This algorithm alternates between K steps of optimizing D and one step of optimizing G.
- The discriminator D was trained multiple times in the hope that it will be maintained near its optimal solution.
- The number of steps K is a hyper-parameter chosen to prevent prohibitive computation and over-fitting in the inner loop.
- The gradient-based updates can be replaced by any standard gradient-based learning rule, e.g. RMSProp, ADAM. (See [Rud16] for details of these methods.)
- We will next examine the objective of GAN in detail. Before doing so, however, let us review two distance measures for distributions: Kullback-Leibler and Jensen-Shannon.

Objective of GAN

- Let μ and ν be probability measures over $\mathcal{X}$ with $\mu \ll \nu$. The *Kullback-Leibler divergence* between μ and ν is defined as

$$\mathrm{KL}(\mu \parallel \nu) := \int_{\mathcal{X}} \ln \frac{d\mu}{d\nu}\, d\mu = \int_{\mathcal{X}} \ln \frac{f(x)}{g(x)} f(x)\, dx,$$

  where f and g are the respective densities of μ and ν, assuming that they exists. This divergence is asymmetric, it can also be infinite if the two measures do not share a common support.

- A measure closely connected to the Kullback-Leibler is the *Jensen-Shannon divergence*:

$$\mathrm{JS}(\mu \parallel \nu) = \mathrm{KL}\left(\mu \,\middle\|\, \frac{\mu+\nu}{2}\right) + \mathrm{KL}\left(\nu \,\middle\|\, \frac{\mu+\nu}{2}\right).$$

  Unlike the Kullback-Leibler, this divergence is symmetrical and always defined.

Objective of GAN

- Through a change-of-variable, $V(D, G)$ can be expanded as

$$V(D, G) = \int_{\mathcal{X}} \left(f_{\boldsymbol{X}_R}(\boldsymbol{X}) \ln[D(\boldsymbol{X})] + f_{\boldsymbol{X}_G}(\boldsymbol{X}) \ln[1 - D(\boldsymbol{X})] \right) d\boldsymbol{X}.$$

- Taking derivative of the integrand with respect to $D(\boldsymbol{X})$, it's not difficult to see that for a fixed G, $V(D, G)$ attains its maximum at

$$D^*(\boldsymbol{X}) = \frac{f_{\boldsymbol{X}_R}(\boldsymbol{X})}{f_{\boldsymbol{X}_R}(\boldsymbol{X}) + f_{\boldsymbol{X}_G}(\boldsymbol{X})}.$$

- In particular, if the generator does a perfect job so that $P_{\boldsymbol{X}_G} \overset{a.e.}{=} P_{\boldsymbol{X}_R}$, we will have $D^*(\boldsymbol{X}) = 1/2$. Plug this back into $V(D, G)$, we get

$$V^*(D, G) = -2 \ln 2.$$

Objective of GAN

- By holding D at its optimum, we can derive with some algebra that

$$V(D^*, G) = \mathrm{JS}(P_{\boldsymbol{X}_R} \parallel P_{\boldsymbol{X}_G}) - 2\ln 2.$$

- Thus, the generator updating step in the GAN algorithm can be seen as minimizing the JS divergence between $\boldsymbol{X}_G$ and $\boldsymbol{X}_R$.
- This also proves $V^* = -2\ln 2$ is a global minimum, since $\mathrm{JS}(\mu \parallel \nu)$ is non-negative and zero if and only if $\mu \overset{a.e.}{=} \nu$.

Performance of GAN

- GAN's success at generating realistically samples, it seems, can largely be contributed to the use of JS divergence instead of KL.
- To see why, let us first review a fundamental assumption in manifold learning.

Performance of GAN

- Known as the *manifold hypothesis*, this assumption states that high dimensional data, in plenty cases of interest, lie in the vicinity of a low dimensional manifold.
- Evidences suggest this assumption is at least approximately correct for many tasks involving images, sounds, or text. [GBC16]
- An heuristic argument for the hypothesis is that if we samples images uniformly at random, then we will almost always observe noisy images, and never encounter an image with discernible features such as an human face. Therefore, the set of interesting images must be negligible compare to the entire image space.

Performance of GAN

- Classical generative models often fit by maximizing likelihood, which asymptotically is equivalent to minimizing $\mathrm{KL}(P_{\boldsymbol{X}_R} \parallel P_{\boldsymbol{X}_G})$.
- However, if the manifold hypothesis is true, then the intersection of the supports of $\boldsymbol{X}_R$ and $\boldsymbol{X}_G$ is negligible, in which case the KL divergence will become $+\infty$.
- The typical remedy is to add a noise term to the model distribution, thus forcing the supports to overlap. But this in turn degrades the quality of the generated samples.

Performance of GAN

- Furthermore, imagine trying to model a multi-modal $P_{\boldsymbol{X}_R}$ with a uni-modal $P_{\boldsymbol{X}_G}$.
- Minimizing $\mathrm{KL}(P_{\boldsymbol{X}_R} \parallel P_{\boldsymbol{X}_G})$ corresponds to moment matching, which has a tendency to overgeneralize and generate implausible samples.
- Minimizing $\mathrm{KL}(P_{\boldsymbol{X}_G} \parallel P_{\boldsymbol{X}_R})$, on the other hand, leads to a mode-seeking behavior. The optimal $P_{\boldsymbol{X}_G}$ will typically concentrate around the largest mode of $P_{\boldsymbol{X}_R}$ while ignoring the rest.
- The JS divergence can be regarded as an interpolation between the two KL divergences. This fact likely alleviates many of the issues caused by the latter. [Hus15]
- The JS divergence is also smoother, and defined even in the case of non-overlapping support. These properties implies it is nicer to handle in computation.

Problems in GAN

- Even though the JS divergence has appealing properties and GAN has been proven to be successful in practice. They are far from perfect.
- The training of GAN is often difficult and unstable. Some of its most prominent problems include:
  - Vanishing gradient;
  - Mode collapse;
  - Non-convergence.

Problems in GAN: Vanishing Gradient

- JS divergence is not free from the problem caused by the manifold hypothesis. If the supports of $\boldsymbol{X}_R$ and $\boldsymbol{X}_G$ are disjoint or lie on low dimensional manifolds, then (almost surely)

$$\mathrm{JS}(P_{\boldsymbol{X}_R} \parallel P_{\boldsymbol{X}_G}) = 2 \ln 2$$

  even if the manifolds lie arbitrarily close. [AB17]
- The loss function of GAN is therefore discontinuous near its optimality, and minimize it at this point via gradient is impossible.

Problems in GAN: Vanishing Gradient

- In addition, if the manifold hypothesis is true, then (again almost surely) there exists an optimal smooth discriminator D^* that discerns $\boldsymbol{X}_R$ and $\boldsymbol{X}_G$ perfectly. It is not difficult to prove that as $D \to D^*$,

$$\left\| \nabla_{\boldsymbol{\theta}} \mathrm{E}_{\boldsymbol{Z}} \left[\ln\left(1 - D\left[G(\boldsymbol{z})\right]\right)\right] \right\|_2 \to 0.$$

- Simply put, as the discriminator becomes better and better after several epochs of training, the gradient for the generator becomes smaller and smaller. The model updates are therefore increasingly negligible.

Problems in GAN: Vanishing Gradient

- The original GAN paper has noticed this issue and proposed replacing $+\mathrm{E}_{\boldsymbol{z}}\left[\ln\left(1-D\left[G(\boldsymbol{z})\right]\right)\right]$ by $-\mathrm{E}_{\boldsymbol{z}}\left[\ln D\left[G(\boldsymbol{z})\right]\right]$ in $V(D,G)$.
- However, it has been shown in [AB17] that, under some strong assumptions, coordinates of $\mathrm{E}_{\boldsymbol{z}}\left[-\nabla_{\boldsymbol{\theta}}\ln D\left[G(\boldsymbol{z})\right]\right]$ are Cauchy, which has undefined mean and variance, implying instability.
- While the assumptions for this result are a bit unrealistic, empirical studies have shown that this alternative gradient indeed becomes increasingly unstable as the discriminator gets better.

Problems in GAN: Mode Collapse

- Another serious issue faced by GAN is the so-called mode collapse. It means the generator fails to output diverse samples, e.g. the true distribution $\boldsymbol{X}_R$ is multi-modal, but $\boldsymbol{X}_G$ concentrates only around few of the modes, or even a single point.
- At mode collapse, the generator can still produces good quality samples, and the discriminator will not classify them as fake. This prevents the generator from improving further.

Problems in GAN: Mode Collapse

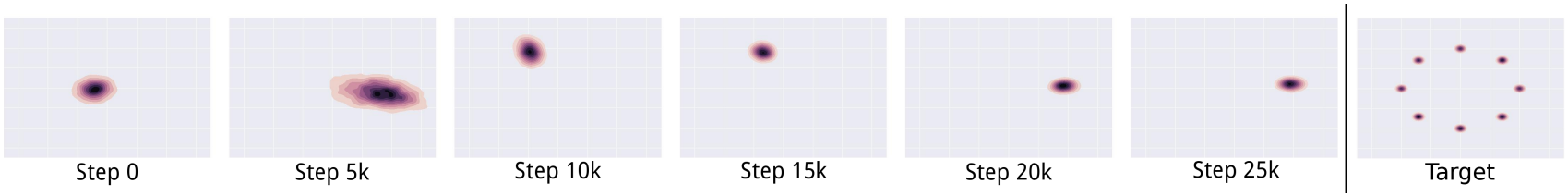

Figure 1: A demonstration of mode collapse from [Met+16]. Columns show a heatmap of the generator distribution after increasing numbers of training steps. The generator rotates through the modes of the data distribution. It never converges to a fixed distribution, and only ever assigns significant probability mass to a single data mode at once.

Problems in GAN: Mode Collapse

- As mentioned earlier, the $\mathrm{KL}(P_{\boldsymbol{X}_G} \parallel P_{\boldsymbol{X}_R})$ component in the GAN objective may be a main culprit.
- Heuristic techniques, e.g. *mini-batch discrimination* and *historical averaging*, have been introduced to deal with the problem. [Sal+16]
- Extended models, such as the unrolled GAN [Met+16], are shown to be effective against mode collapse.
- Certain non-GAN models, such as the Implicit Maximum Likelihood Estimation [LM18], also provide insight and potential solutions to the mode collapsing issue. They may be helpful in directing future studies.

Problems in GAN: Non-convergence

- Recall that the algorithm of GAN can be interpreted as finding a Nash equilibrium for a minimax problem through stochastic gradient ascend/decent.
- Unfortunately, stochastic gradient methods are not designed for this kind of problems. There is no theoretical guarantee that the algorithm will ever converge.
- After all, the training of GAN is a highly heuristic process and further studies are still in need.

Wasserstein GAN

- In an attempt to cope with the aforementioned difficulties, the *Wasserstein Generative Adversarial Network* (WGAN, [ACB17]) was introduced.
- As the name implied, it replaced the JS divergence in the GAN objective with the Wasserstein distance.
- Wasserstein distance is deeply connected to the problem of optimal transport. Both of which have become of great interest in recent years.

Wasserstein GAN

- Let μ and ν be probability measures over $\mathcal{X}$, and let $\mathbf{\Gamma}(\mu, \nu)$ be the set of all measures over $\mathcal{X} \times \mathcal{X}$ whose respective marginals are μ and ν. The Wasserstein-p distance between μ and ν is defined as

$$W_p(\mu, \nu) := \left(\inf_{\gamma \in \mathbf{\Gamma}(\mu,\nu)} \int_{\mathcal{X}\times\mathcal{X}} \|x - y\|^p \, d\gamma(x, y) \right)^{1/p}.$$

- Intuitively, it is the minimum "cost" of transporting one measure to another.
- The WGAN utilized specifically the Wasserstein-1 distance (also known as the earth-mover distance). It can be expressed as

$$W_1(\mu, \nu) = \inf_{\gamma \in \mathbf{\Gamma}(\mu,\nu)} E_\gamma \left[\|X - Y\| \right],$$

where $(X, Y) \sim \gamma$.

Wasserstein GAN

- In the context of GAN, the choice of W_1 offers several benefits over the JS divergence.
- An important property of W_1 is that it induces the weak topology. That is, $W_1(\mu_n, \mu) \to 0$ if and only if $\mu_n \xrightarrow{D} \mu$. This is comparatively weaker than JS convergence, which is equivalent to the convergence in total variation.
- Intuitively, the easier the convergence, the easier it is to have a continuous mapping between $\boldsymbol{\theta}$ and $P_{\boldsymbol{X}_G}$.

Wasserstein GAN

- Indeed, if $G(\boldsymbol{z}; \boldsymbol{\theta})$ is continuous in $\boldsymbol{\theta}$, so is $\mathrm{W}_1(P_{\boldsymbol{X}_R}, P_{\boldsymbol{X}_G})$.
- Moreover, if $G(\boldsymbol{z}; \boldsymbol{\theta})$ is locally Lipschitz and satisfies a mild regularity condition, then $\mathrm{W}_1(P_{\boldsymbol{X}_R}, P_{\boldsymbol{X}_G})$ is continuous everywhere and differentiable almost everywhere.
- The mild condition we mentioned is automatically satisfied by any feed-forward neural networks, which are function composed by affine transformations and pointwise nonlinearities that are smooth Lipschitz functions.
- As a result, W_1 as an objective in GAN is expected to behave much more nicely than the JS divergence.

Wasserstein GAN

- Unfortunately, computing the W_p distance is very difficult. It is usually done in the Kantorovich-Rubinstein dual form:

$$\frac{W_p^p(\mu,\nu)}{p} = \sup_{\psi\in\Psi_p(\mathcal{X})} \int_{\mathcal{X}} \psi \, d\mu + \int_{\mathcal{X}} \psi^{c_p} \, d\nu,$$

- ψ^{c_p} here denotes the c_p-conjugate of ψ:

$$\psi^{c_p}(y) = \inf_{x\in\mathcal{X}} c_p(x,y) - \psi(x), \qquad c_p(x,y) = \frac{\|x-y\|^p}{p}.$$

- $\Psi_p(\mathcal{X})$ is the set of all c_p-concave functions, i.e. functions that can be written as c_p-conjugate of some other functions.

Wasserstein GAN

- Incidentally, the dual form of W_1 distance is quite tame, namely:

$$W_1(\mu, \nu) = \sup_{\psi \in \Psi_1(\mathcal{X})} E_\mu[\psi] - E_\nu[\psi],$$

where the c_1-concave set is in fact the set of all 1-Lipschitz function:

$$\Psi_1(\mathcal{X}) = \{\psi \colon \mathcal{X} \to \overline{\mathbb{R}} \mid \|\psi(x) - \psi(y)\| \leq \|x - y\| \ \forall x, y \in \mathcal{X}\}.$$

Wasserstein GAN

- Of course, optimize $\mathrm{W}_1(P_{\boldsymbol{X}_R}, P_{\boldsymbol{X}_G})$ over all 1-Lipschitz functions is no easy task. We can however discretize it in the following way: suppose $\{\psi_{\boldsymbol{w}}\}_{\boldsymbol{w}\in\mathcal{W}}$, for some $\mathcal{W}$ compact, is a parameterized family of functions that are all K-Lipschitz for some K, then solving

$$\max_{\boldsymbol{w}\in\mathcal{W}} \mathrm{E}_{\boldsymbol{X}_R}[\psi_{\boldsymbol{w}}] - \mathrm{E}_{\boldsymbol{X}_G}[\psi_{\boldsymbol{w}}]$$

  yield $\mathrm{W}_1(P_{\boldsymbol{X}_R}, P_{\boldsymbol{X}_G})$ up to a multiplicative constant.
- Notably, we also have

$$\nabla_{\boldsymbol{\theta}} \mathrm{W}_1(P_{\boldsymbol{X}_R}, P_{\boldsymbol{X}_G}) = -\mathrm{E}_{\boldsymbol{Z}}\left[\nabla_{\boldsymbol{\theta}}\psi_{\boldsymbol{w}}\left(G(\boldsymbol{z};\boldsymbol{\theta})\right)\right],$$

  which make gradient methods possible.
- This naturally is a task for neural networks, since their strength lies in parametrically approximating functions. The next slide will present an algorithm for fitting WGAN.

Algorithm of WGAN

Input: A noise prior $\boldsymbol{Z}$ and a data generating distribution $\boldsymbol{X}_R$;
for $t \leftarrow 1, \ldots, T$ **do**
  for $k \leftarrow 1, \ldots, K$ **do**
    Sample minibatch of noise $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ from $\boldsymbol{Z}$;
    Sample minibatch of data $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_m\}$ from $\boldsymbol{X}_R$;
    Update $\psi_{\boldsymbol{w}}$ by stochastic gradient ascend:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha \nabla_{\boldsymbol{w}} \frac{1}{m} \sum_{i=1}^{m} \left[\psi_{\boldsymbol{w}}(\boldsymbol{X}_i) - \psi_{\boldsymbol{w}}\left(G(\boldsymbol{z}; \boldsymbol{\theta})\right) \right]$$

    Clip $\boldsymbol{w}$ into the range $[-c, c]$.
  end
  Sample minibatch of noise $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ from noise prior $\boldsymbol{Z}$;
  Update the generator G by stochastic gradient descend:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \frac{1}{m} \sum_{i=1}^{m} \psi_{\boldsymbol{w}}\left(G(\boldsymbol{z}; \boldsymbol{\theta})\right)$$

end

Algorithm 2: Minibatch training of Wasserstein generative adversarial nets.

Algorithm of WGAN

- Akin to GAN, this algorithm alternates between updating $\psi_{\boldsymbol{w}}$ and optimizing G.
- To ensure the parameters $\boldsymbol{w}$ lies in a compact space, it is clipped to a fixed hypercube $[-c, c]^d$ (d is the dimension of parameters) after each gradient update.
- This unfortunately weakens the stability of WGAN: if the clipping parameter is large, then $\psi_{\boldsymbol{w}}$ will take longer to reach its optimality; if the parameter is small, then the problem of vanishing gradients will again appear.

Performance of WGAN

- The use of Wasserstein distance significantly alleviates the problem of vanish gradients, as demonstrated by the following figure:

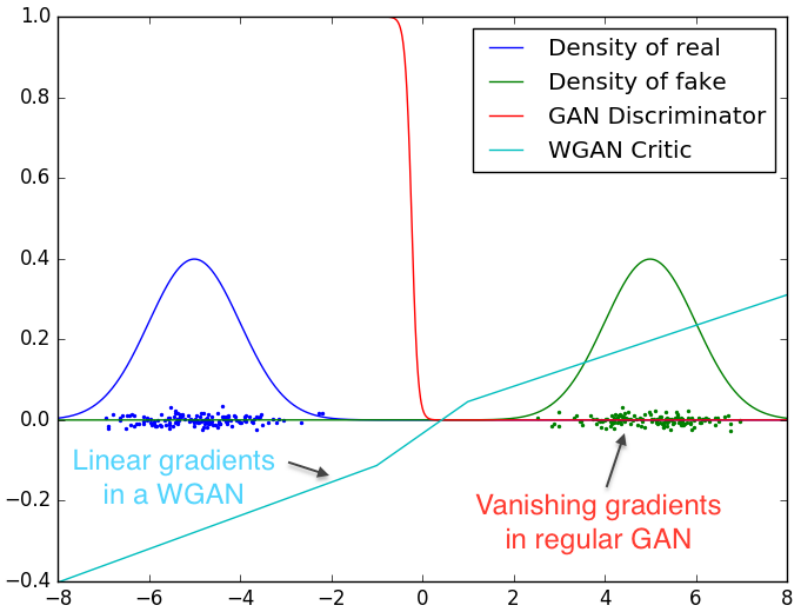

Figure 2: Optimal discriminator and critic when learning to differentiate two Gaussians. The GAN discriminator of saturates and results in vanishing gradients. The WGAN critic provides very clean gradients on all parts of the space.

Performance of WGAN

- This means that the discriminator can be trained till optimality, which in turn helps the mode collapse issue.
- However, the choice of clipping parameter c remains an issue. Several alternative methods have been proposed, the most notable one replace the clipping by a gradient penalty (WGAN-GP, [Gul+17]). Specifically, the following term was added to the objective of WGAN:

$$\lambda \, \mathrm{E}_{\hat{\boldsymbol{X}}} \left[\left(\|\nabla \psi_{\boldsymbol{w}}\|_2 - 1 \right)^2 \right],$$

where $\hat{\boldsymbol{X}}$ is defined, implicitly, by sampling uniformly along straight lines between pairs of points from $\boldsymbol{X}_R$ and $\boldsymbol{X}_G$.

Discussion

- We have introduced the basic of GAN and WGAN, and discussed why the latter is more attractive on a mathematical ground.
- A large-scale simulation study ([Luc+18]) was conducted to compare the performance of several GAN variants (including WGAN and WGAN-GP). It reported that, given enough computation budget and thorough hyper-parameters tuning, variants of GAN were unable to beat each other consistently.
- We may consider, in the future, investigating the implications of this observation.

References I

Martin Arjovsky and Léon Bottou. *Towards Principled Methods for Training Generative Adversarial Networks*. 2017. arXiv: 1701.04862 [stat.ML].

Martin Arjovsky, Soumith Chintala, and Léon Bottou. *Wasserstein GAN*. 2017. arXiv: 1701.07875 [stat.ML].

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

Ian Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., 2014, pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

References II

Ishaan Gulrajani et al. *Improved Training of Wasserstein GANs*. 2017. arXiv: 1704.00028 [cs.LG].

Ferenc Huszár. *How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary?* 2015. arXiv: 1511.05101 [stat.ML].

Ke Li and Jitendra Malik. *Implicit Maximum Likelihood Estimation*. 2018. arXiv: 1809.09087 [cs.LG].

Mario Lucic et al. "Are GANs Created Equal? A Large-Scale Study". In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio et al. Curran Associates, Inc., 2018, pp. 700–709. URL: http://papers.nips.cc/paper/7350-are-gans-created-equal-a-large-scale-study.pdf.

Luke Metz et al. *Unrolled Generative Adversarial Networks*. 2016. arXiv: 1611.02163 [cs.LG].

References III

Alec Radford, Luke Metz, and Soumith Chintala. *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*. 2015. arXiv: 1511.06434 [cs.LG].

Sebastian Ruder. *An overview of gradient descent optimization algorithms*. 2016. arXiv: 1609.04747 [cs.LG].

Tim Salimans et al. *Improved Techniques for Training GANs*. 2016. arXiv: 1606.03498 [cs.LG].