Generative adversarial network

Carlos Valle

Departamento de Informática
Universidad Técnica Federico Santa María

cvalle@inf.utfsm.cl

February 7, 2020

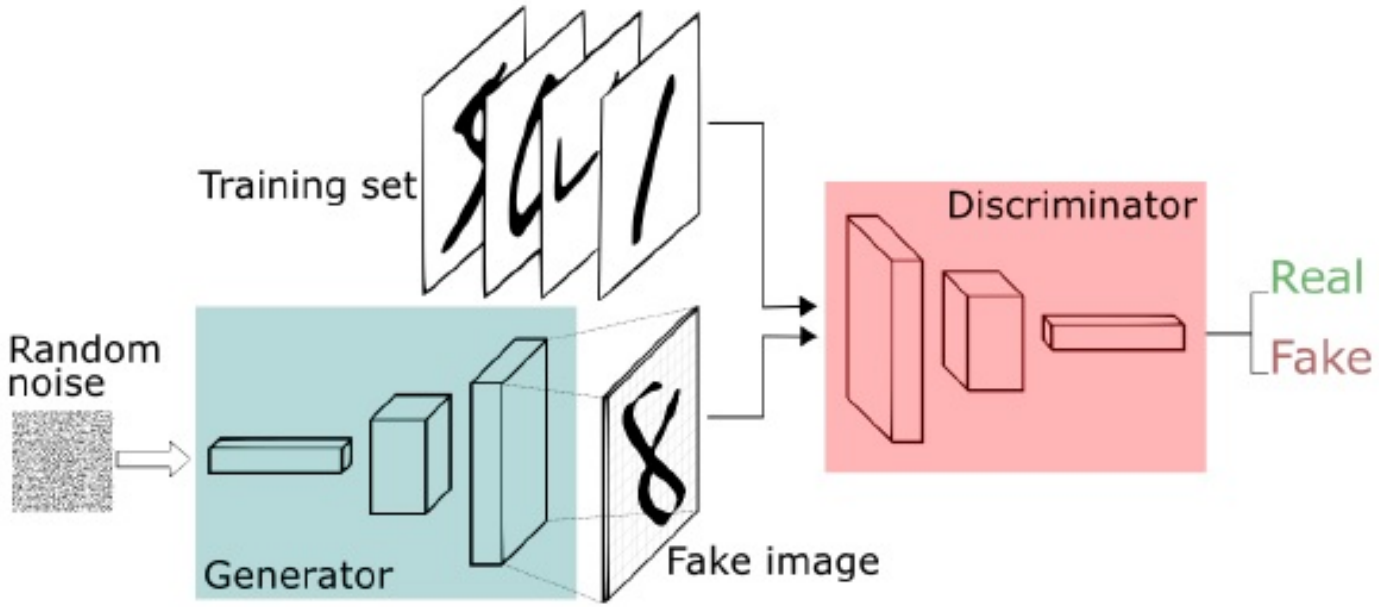
Overview

1. Introduction
2. GANs
3. Main theoretical results of GANs
4. Modal Collapse

Generative adversarial network

- Generative adversarial networks are based on a game theoretic scenario in which the generator network must compete against an adversary.
- The generator G is a generative model whose goal is to approximate the probability distribution $p(x)$ of a set of observations $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M$ such that $\mathbf{x}_m \sim p(\mathbf{x})$.
- The generator attempts to produce artificial examples $\mathbf{x}$ that follow the same law of probability of training observations, this is, $p(x)$.
- The mechanism that the generator uses to produce these artificial examples consists of
  1. Generate a v.a. $\mathbf{z}$ distributed according to a law $p_z(\mathbf{z})$ to be chosen.
  2. Transform $\mathbf{z}$ into $\mathbf{x}$ using a Neural Network $G_\theta(\mathbf{z})$, where θ is a vector with the network's parameters.

Generative adversarial network architecture



How to train a GAN?

- To train the generator, the GAN uses a 'game' in which G tries to produce an artificial example $\mathbf{x}$ and the discriminator D tries to detect if $\mathbf{x}$ is a real example or an artificial example.
- If $p_g(\mathbf{x})$ represents the probability distribution of the examples produced by the generator at a given time, the discriminator tries to detect if $x \sim p(\mathbf{x})$ (real example) or if $x \sim p_g(\mathbf{x})$ (artificial example).
- More precisely, the discriminator assigns to each input $\mathbf{x}$ a probability $D_\phi(\mathbf{x}) \in [0; 1]$ that $\mathbf{x}$ is a real example.
- For this purpose, the discriminator implements a neural network $D_\phi(\cdot)$ with adaptive parameters ϕ.
- To avoid the discriminator always say that an example is artificial, it receives both real examples, obtained from the training set, as examples produced by the generator.

How to train a GAN? (2)

- Thus, the discriminator adopts the following objective function
-

$$\begin{aligned} & \max_D E_{\mathbf{x}\sim p(\mathbf{x})}[\log D(\mathbf{x})] + E_{x\sim p_g(\mathbf{x})}[\log(1 - D(\mathbf{x}))] \\ \iff \quad & E_{\mathbf{x}\sim p(\mathbf{x})}[\log D(\mathbf{x})] + E_{z\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \end{aligned}$$

- And we simultaneously train the generator to minimize $E_{z\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] = \log(1 - D(G(\mathbf{z}))$.
- In other words, D and G play a two-player minimax game with the value function $V(G, D)$ as follows:

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

Training a GAN in practice

- In practice, the above-mentioned optimization is done by parameterizing the hypothesis space $\mathcal{D}$ of models where we extract the discriminator, that is, $\mathcal{D} = \{D_phi : \phi \in \Phi\}$, and parameterizing the family of models $\mathcal{G}$ where the generator is extracted, i.e., when $\mathcal{G} = \{G_theta : \theta \in \Theta\}$
- Thus the problem to solve is:

$$\min_{\theta} \max_{\phi} E_{\mathbf{x} \sim p(\mathbf{x})}[\log D_{\phi}(\mathbf{x})] + E_{z \sim p_z(\mathbf{z})}[\log(1 - D_{\phi}(G_{\theta}(\mathbf{z})))]$$

- If $\mathcal{D}$ and $\mathcal{G}$ are neural networks we can optimize it by gradient descent.
- The original GAN algorithm is (Goodfellow et al., Generative Adversarial Nets, 2014):

Training a GAN in practice (2)

Algorithm 1 GAN algorithm

1: Initialize the parameters of the discriminator $\phi = \phi^{(0)}$ and the generator $\theta = \theta^{(0)}$.
2: **for** $t = 1, 2, \ldots, T$ **do** ▷ epochs
3: **for** $k = 1, 2, \ldots, K$ **do** ▷ Train the discriminator
4: Sample a minibatch of m noise examples $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m\}$ from $p_z(\mathbf{z})$
5: Sample a minibatch of m examples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ from the train set.
6: Update the discriminator parameters ϕ by ascending its stochastic gradient

$$\nabla_{\phi}^{(t)} = \frac{\partial}{\partial \phi} \left(\frac{1}{m} \sum_{i=1}^{m} \log D_{\phi}(\mathbf{x}_i) + \log(1 - D_{\phi}(G_{\theta}(\mathbf{z}_i))) \right)$$

7: **end for** ▷ Train the generator.
8: Sample a minibatch of m noise examples $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m\}$ from $p_z(\mathbf{z})$
9: Update the generator by descending its stochastic gradient

$$\nabla_{\theta}^{(t)} = \frac{\partial}{\partial \theta} \left(\frac{1}{m} \sum_{i=1}^{m} \log(1 - D_{\phi}(G_{\theta}(\mathbf{z}_i))) \right)$$

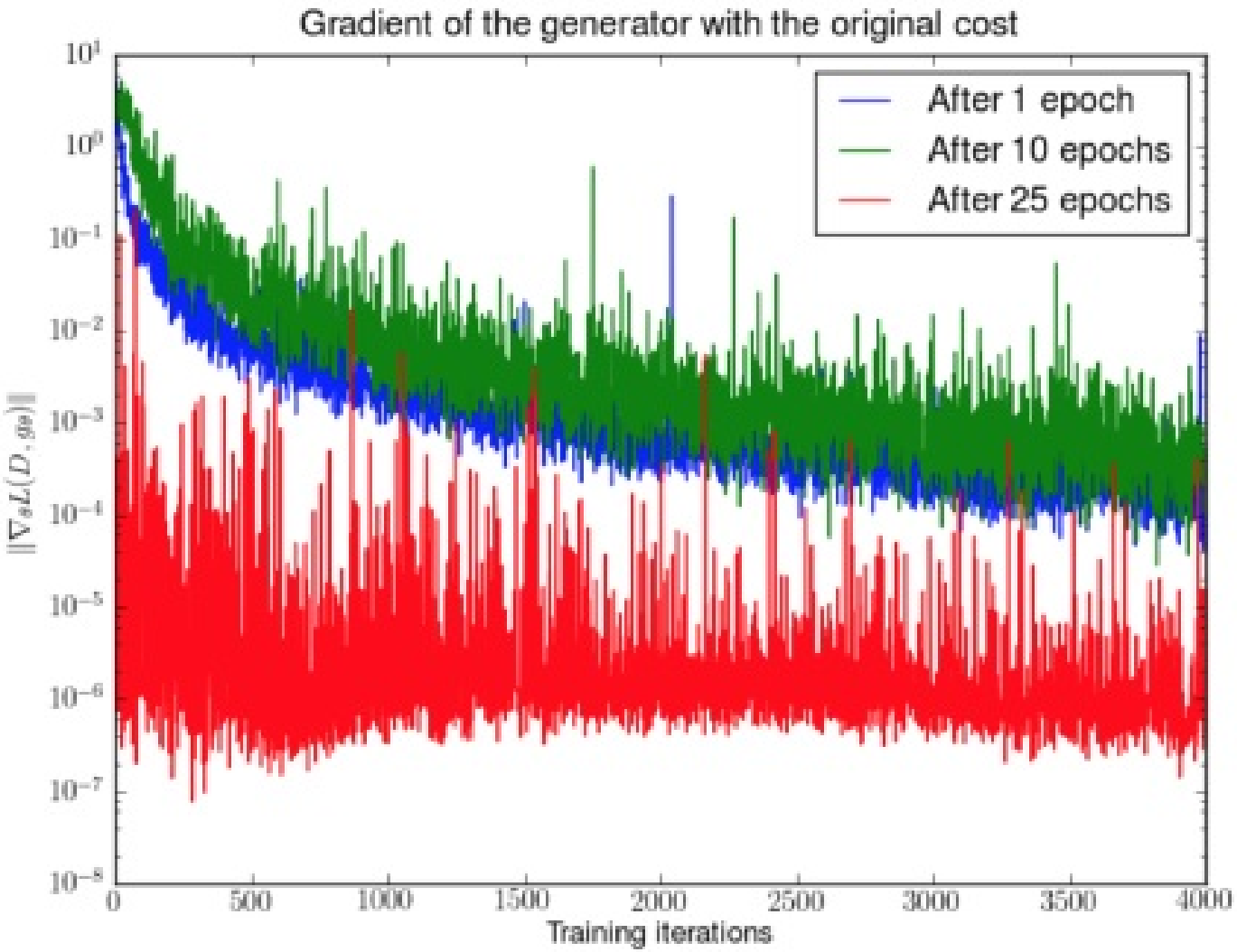
10: **end for**

Vanishing gradient descent problems

- A problem observed in the original GAN's paper has to do with the vanishing gradient descent in the generator.
- If the discriminator is too good to distinguish the instances artificially generated from the real ones (very probable situation in the initial epochs of training, in which the generator is poor), that is, $D(G(\mathbf{z})) \approx 0$
- It happens that the gradient of the generator is

$$\frac{\partial}{\partial G} E_{z \sim p_z(\mathbf{z})} \log(1 - D(G(\mathbf{z}))) = -E_{z \sim p_z(\mathbf{z})} \left[\frac{1}{1 - D(G(\mathbf{z}))} \frac{\partial D(G(\mathbf{z}))}{\partial G} \right]$$

Vanishing gradient descent problems (2)



Vanishing gradient descent problems (3)

- One way to mitigate this situation is replace the original objective function of the generator.
- Instead of minimizing $\log(1 - D(G(\mathbf{z})))$, we can maximize $\log D(G(\mathbf{z}))$.
- Hence, the new gradients are

$$\frac{\partial}{\partial G} E_{z \sim p_z(\mathbf{z})} \log(D(G(\mathbf{z}))) = -E_{z \sim p_z(\mathbf{z})} \left[\frac{1}{D(G(\mathbf{z}))} \frac{\partial D(G(\mathbf{z}))}{\partial G} \right]$$

- Note that in the original objective function the gradients are scaled by $1/\log(1 - D(G(\mathbf{z})))$ which is close to 1 when $D(G(\mathbf{z})) \approx 0$.
- Where as in the new objective function, the gradients are scaled by $D(G(\mathbf{z}))$, which is very large when $D(G(\mathbf{z})) \approx 0$.

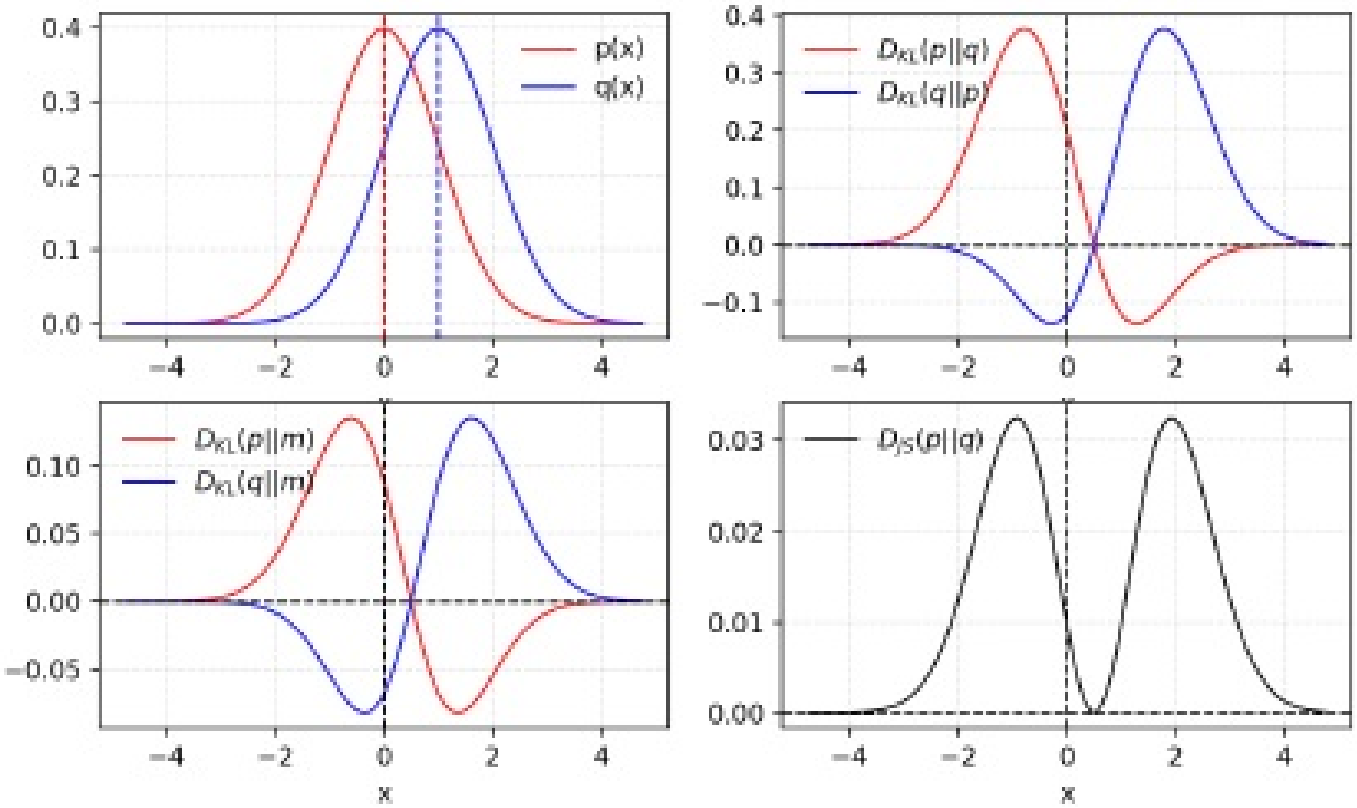
Goal and divergence measure

- As we viewed, this game between G and D makes that G learns the data distribution.
- To measure the quality of G we could use the KL divergence.
- As we saw, KL divergence is not symmetric, and it ignores the differences between $p(x)$ and $q(x)$ for $\mathbf{x}$ when $p(\mathbf{x}) \approx 0$.
- To fix that we use the Jensen-Shannon divergence, which is a symmetric version of KL divergence:

$$JS(p||q) = \frac{1}{2}KL(p||M) + \frac{1}{2}KL(q||M),$$

- where M is $(p+q)/2$.

Goal and divergence measure (2)



Main theoretical results of GANs

Theorem 1

For G fixed, the optimal discriminator D is

$$D^*(\mathbf{x}) = \frac{p(\mathbf{x})}{p(\mathbf{x}) + p_g(\mathbf{x})}$$

Main theoretical results of GANs (2)

Proof.

The discriminator maximizes the following function:

$$E_{\mathbf{x}\sim p(\mathbf{x})}[\log D(\mathbf{x})] + E_{x\sim p_g(\mathbf{x})}[\log(1 - D(\mathbf{x}))] \tag{1}$$

$$\sum_{\mathbf{x}} p(\mathbf{x}) \log D(\mathbf{x}) + \sum_{\mathbf{x}} p_g(\mathbf{x}) \log(1 - D(\mathbf{x}))$$

$$\sum_{\mathbf{x}} (p(\mathbf{x}) \log D(\mathbf{x}) + p_g(\mathbf{x}) \log(1 - D(\mathbf{x})))$$

The function $J(y) = a \log y + b \log(1 - y)$ has the following derivatives:

$$\frac{\partial J}{\partial y} = \frac{a}{y} - \frac{b}{(1 - y)}$$

$$\frac{\partial J^2}{\partial y^2} = -\frac{a}{y^2} - \frac{b}{(1 - y)^2}, 0$$

Thus $y^* = \frac{a}{a+b} \in [0, 1]$ is a maximum for J. □

Main theoretical results of GANs (3)

- This means that the discriminator can minimize the sum in 1 by choosing implement the function

$$D^*(\mathbf{x}) = \frac{p(\mathbf{x})}{p(\mathbf{x}) + p_g(\mathbf{x})}$$

- Of course, this will be possible in practice, if the hypothesis space chosen by the discriminator contains this function.
- Therefore we have to train the generator to minimize the objective function $V(D, G)$ is equivalent to minimize $JS(p(\mathbf{x})||p_g(\mathbf{x}))$.

Main theoretical results of GANs (4)

Theorem 2

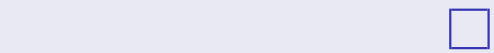
Training the generator to minimize $V(D,G)$ *is equivalent to finding* $p_g(\mathbf{x})$ *so as to minimize* $JS(p(\mathbf{x})||pg(\mathbf{x}))$. *Therefore, the generator is optimal if and only if* $p_g(\mathbf{x}) = p(\mathbf{x})$.

Main theoretical results of GANs (5)

Proof.

Replacing in the objective function the optimal discriminator found in the previous section, we get the generator to train to minimize the next objective function

$$
\begin{aligned}
C(G) &= E_{\mathbf{x}\sim p(\mathbf{x})}\left[\log\left(\frac{p(\mathbf{x})}{p(\mathbf{x})+p_g(\mathbf{x})}\right)\right] + E_{x\sim p_g(\mathbf{x})}\left[\log\left(1-\left(\frac{p(\mathbf{x})}{p(\mathbf{x})+p_g(\mathbf{x})}\right)\right)\right] \\
&= E_{\mathbf{x}\sim p(\mathbf{x})}\left[\log\left(\frac{p(\mathbf{x})}{p(\mathbf{x})+p_g(\mathbf{x})}\right)\right] + E_{x\sim p_g(\mathbf{x})}\left[\log\left(\frac{p_g(\mathbf{x})}{p(\mathbf{x})+p_g(\mathbf{x})}\right)\right] \\
&= E_{\mathbf{x}\sim p(\mathbf{x})}\left[\log\left(\frac{1}{2}\frac{p(\mathbf{x})}{\frac{p(\mathbf{x})+p_g(\mathbf{x})}{2}}\right)\right] + E_{x\sim p_g(\mathbf{x})}\left[\log\left(\frac{1}{2}\frac{p_g(\mathbf{x})}{\frac{p(\mathbf{x})+p_g(\mathbf{x})}{2}}\right)\right] \\
&= E_{\mathbf{x}\sim p(\mathbf{x})}\left[\log\left(\frac{p(\mathbf{x})}{M}\right)\right] + E_{x\sim p_g(\mathbf{x})}\left[\log\left(\frac{p_g(\mathbf{x})}{M}\right)\right] - 2\log(2) \\
&= KL(p(\mathbf{x})||M) + KL(p_g(\mathbf{x})||M) - \log(4) \\
&= 2JS(p(\mathbf{x})||p_g(\mathbf{x})) - \log(4)
\end{aligned}
$$



Main theoretical results of GANs (6)

- As we discused, $JS(p(\mathbf{x})||p_g(\mathbf{x})) = 0$ if only if $p(\mathbf{x}) = p_g(\mathbf{x})$.
- So, the global minimum for $C(G)$ is $-\log(4)$.
- It is important to note that in order to establish the previous result, the discriminator is not fixed.
- Instead, it is optimized in conjunction with the generator (to obtain the result we have replaced $D(\mathbf{x})$ by the optimal function corresponding to each possible generator, which is what we are looking for to optimize).
- If we consider that D is fixed we can demonstrate the following theorem

Main theoretical results of GANs (7)

Theorem 3

For any fixed discriminator $D^*(\mathbf{x})$ *, the objective function of the generator side* $\min_G C_{D^*}(G)$ *with*

$$C_{D^*}(G) = E_{\mathbf{x}\sim p(\mathbf{x})}[\log D^*(\mathbf{x})] + E_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D^*(G(\mathbf{z})))]$$

is such that any function $G(\cdot)$ *that maps* $\mathbf{z} \sim p_{\mathbf{z}}$ *to some (possibly only 1) of the modes of* $D^*(\mathbf{x})$ *be optimal.*

Main theoretical results of GANs (8)

Proof.

Optimizing $\min_G C_{D^*}(G)$ in G is equivalent to solve

$$\min_G C'_{D^*}(G) = E_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D^*(G(\mathbf{z})))]$$

$$\Leftrightarrow \quad \sum_{\mathbf{z}} p_{\mathbf{z}}(\mathbf{z}) \log(1 - D^*(G(\mathbf{z}))$$

Let G^* be the function that $G^*(\mathbf{z}) = s, \forall \mathbf{z}$, where $s \in \text{argmax}_{\mathbf{z}} D(\mathbf{x}$. We have that for any alternative generator G the following property is met

$$\forall \mathbf{z} D^*(G(\mathbf{z})) \leq D^*(G^*(\mathbf{z})) \Leftrightarrow \forall \mathbf{z}(1 - D^*(G(\mathbf{z}))) \geq (1 - D^*(G^*(\mathbf{z}))$$

Since $\log$ is a monotonic function, we have that

$$\forall \mathbf{z} \log(1 - D^*(G(\mathbf{z}))) \geq \log(1 - D^*(G^*(\mathbf{z}))$$

Proof cont.

Proof.

Therefore, for any $p_{\mathbf{z}}$

$$\begin{aligned} & \sum_{\mathbf{z}} \log(1 - D^*(G(\mathbf{z}))) \geq \sum_{\mathbf{z}} \log(1 - D^*(G^*(\mathbf{z})) \\ \Leftrightarrow \quad & E_{\mathbf{z} \sim p(\mathbf{z})} \log(1 - D^*(G(\mathbf{z}))) \geq E_{\mathbf{z} \sim p(\mathbf{z})} \log(1 - D^*(G^*(\mathbf{z})) \\ \Leftrightarrow \quad & C'_{D^*}(G) \geq C'_{D^*}(G^*) \end{aligned}$$

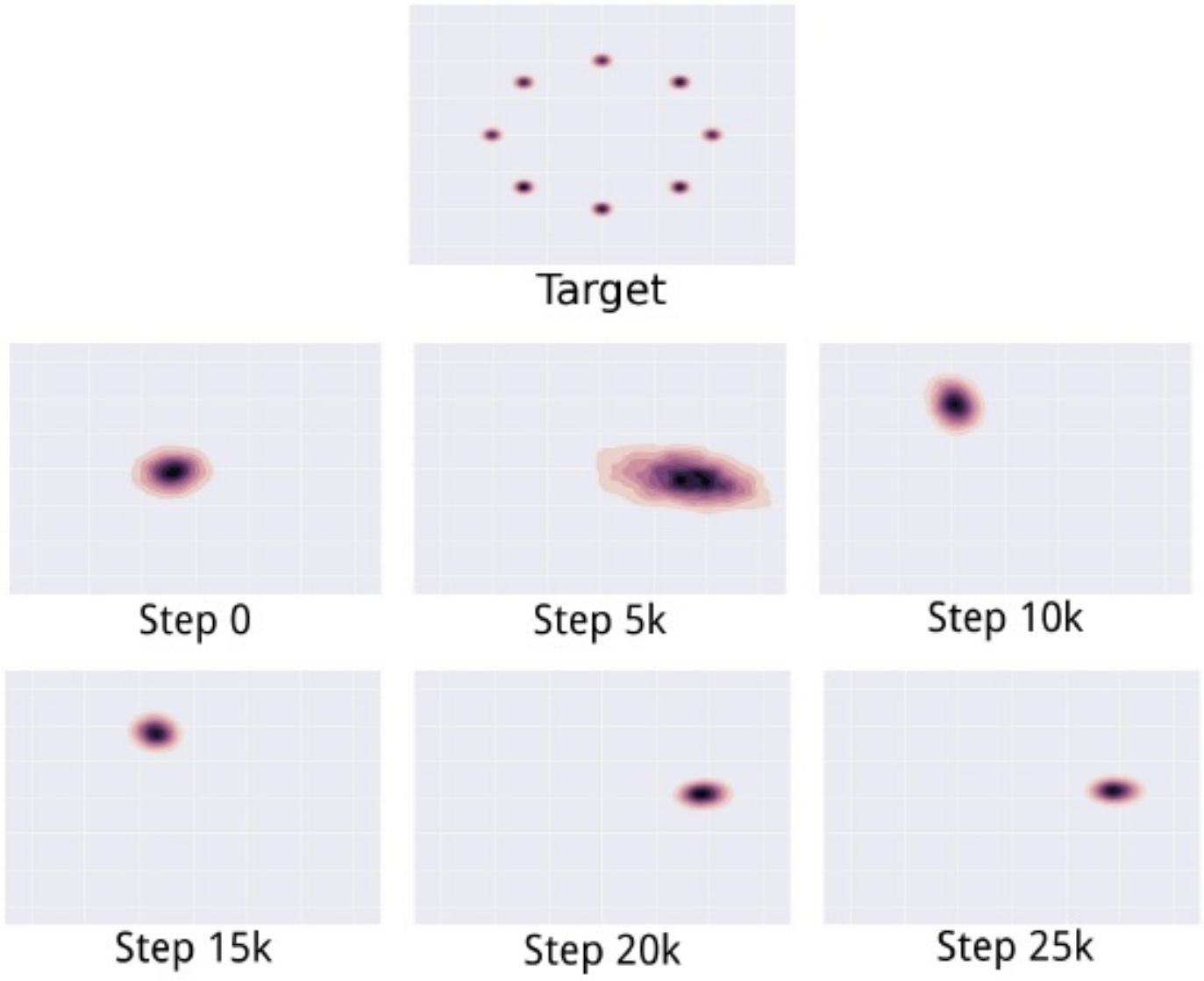
This proofs that G^* is a minimum of the objective function $C_{D^*}(G)$. □

Modal Collapse

- The above problem explains what is known in the GANs literature as the Modal Collapse.
- It means that the GAN generator ends up producing very little varied examples, which correspond to one of the modes of the distribution represented by the discriminator, which in turn coincides with one of the modes of the real distribution $p(\mathbf{x})$.
- The previous theorem shows that the problem of modal collapse is due to the optimization of the generator with a fixed discriminator.
- This suggests a way to avoid the modal collapse could be generate greater interaction between both optimizations, which is exactly what the unrolled GAN do, proposed by Metz in 2016.

Modal Collapse (2)

- Metz et al. Unrolled Generative Adversarial Networks, 2016.



Modal Collapse (3)

- Arjovsky et al. Wasserstein GAN, 2017.

Any questions?