DeepMind

GANs: a lesson on distributional divergences and optimisation in games

Mihaela Rosca

Staff Research Engineer at DeepMInd
PhD student at UCL

Mediterranean Machine learning Summer school 2022

Why still care about GANs?

Diffusion models are producing great results, VAEs are catching up.

GANs

Still produce great results on image generation.

GANs

Beyond that, they provide an excellent ground to learn about:

- distributional learning principles (beyond maximum likelihood)
- optimisation in games

This is what we will talk about today.

Summary of today's lecture

- GANs intro
- GANs and distributional divergences and distances
- GANs and optimisation in two-player games

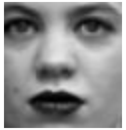
Disclaimer

The field is large and there are many other related views on GANs, related works and applications. This talk presents one view.

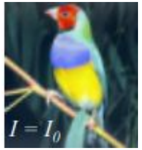
Specifically, this talk focuses more on general principles than specific models. There are many interesting and useful GAN models out there that will not be mentioned here.

Please look at the references at the end of the slides for more related works and check connectedpapers.com for other works.

Generative adversarial networks



Goodfellow, et al. **Generative adversarial networks.** NIPS **(2014)**



Denton, et al. **Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks.** NIPS **(2015)**

Radford et al. **Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks** ICLR **(2015)**

Miyato et al. **Spectral normalization for Generative Adversarial Networks** ICLR **(2018)**

Karras et al. **Large Scale GAN Training for High Fidelity Natural Image Synthesis** ICLR **(2018)**

Brock et al. **Large Scale GAN Training for High Fidelity Natural Image Synthesis** ICLR **(2019)**

Karras et al. **A Style-Based Generator Architecture for Generative Adversarial Networks** CVPR **(2019)**

Generative adversarial networks

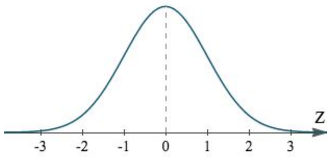
Want to learn more?

Goodfellow, et al. **Generative adversarial networks.** Neurips (2014)

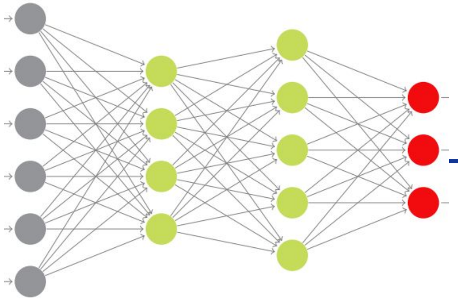
Learning an implicit generative model through a two player game.

Implicit latent variable models (generator)

latent (“noise”) vector
z ~ P(**z**)



generator G

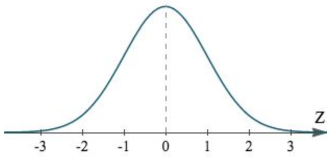


generated data
G(**z**)

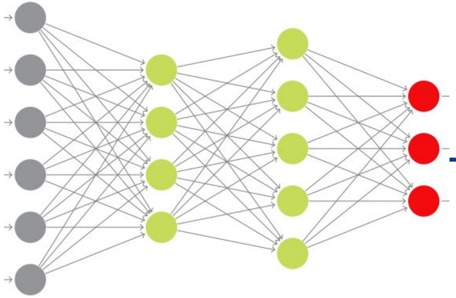
parameters θ

Implicit latent variable models (generator)

latent (“noise”) vector
$\mathbf{z} \sim P(\mathbf{z})$



generator G



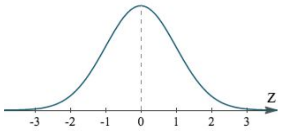
parameters θ

generated data
$G(\mathbf{z})$

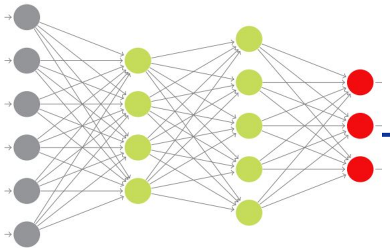
Only samples!
No explicit likelihood!

What objective can we use?

latent (“noise”) vector
z ~ P(**z**)



generator



parameters θ

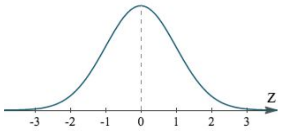
generated data
G(**z**)

$$L(\theta)$$

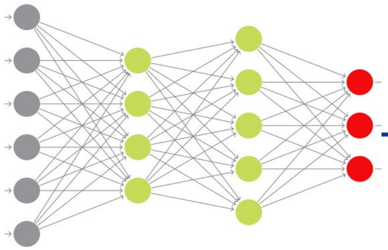
What objective can we use?

$$x_1, \ldots x_n \sim p^*(x)$$

latent ("noise") vector
z ~ P(**z**)



generator



parameters θ

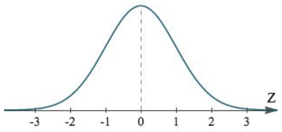
generated data
G(**z**)

$$L(\theta)$$

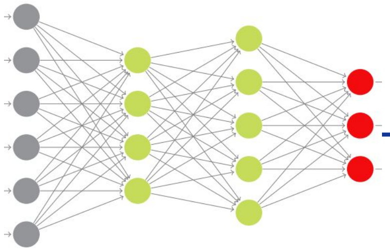
What objective can we use?

$$x_1, \ldots x_n \sim p^*(x)$$

latent ("noise") vector
z ~ P(**z**)



generator



parameters θ

generated data
G(**z**)

$$L(\theta)$$

backprop

How can we learn this model?

Cannot do maximum likelihood, because we need to query the model for the likelihoods of the data.

$$\mathbf{KL}\left[p^* || q_\theta\right] = \int p^*(x) \log \frac{p^*(x)}{q_\theta(x)} dx$$

$$= \int p^*(x) \log p^*(x) - \int p^*(x) \log q_\theta(x) dx$$

$$= C - \mathbb{E}_{p*(x)} \log q_\theta(x)$$

How can we learn this model?

Cannot do maximum likelihood, because we need to query the model for the likelihoods of the data.

$$\mathbf{KL}\left[p^* || q_\theta\right] = \int p^*(x) \log \frac{p^*(x)}{q_\theta(x)} dx$$

$$= \int p^*(x) \log p^*(x) - \int p^*(x) \log q_\theta(x) dx$$

$$= C - \mathbb{E}_{p*(x)} \log \cancel{q_\theta(x)}$$

The types of objectives we are looking for

We have samples from the model and samples from the data.

We are thus looking for objectives which depend on the model and the data distribution via expectations.

$$\mathbf{E}_{p^*(x)} f(x) + \mathbf{E}_{q_\theta(x)} g(x)$$

The types of objectives we are looking for

We have samples from the model and samples from the data.

We are thus looking for objectives which depend on the model and the data distribution via expectations.

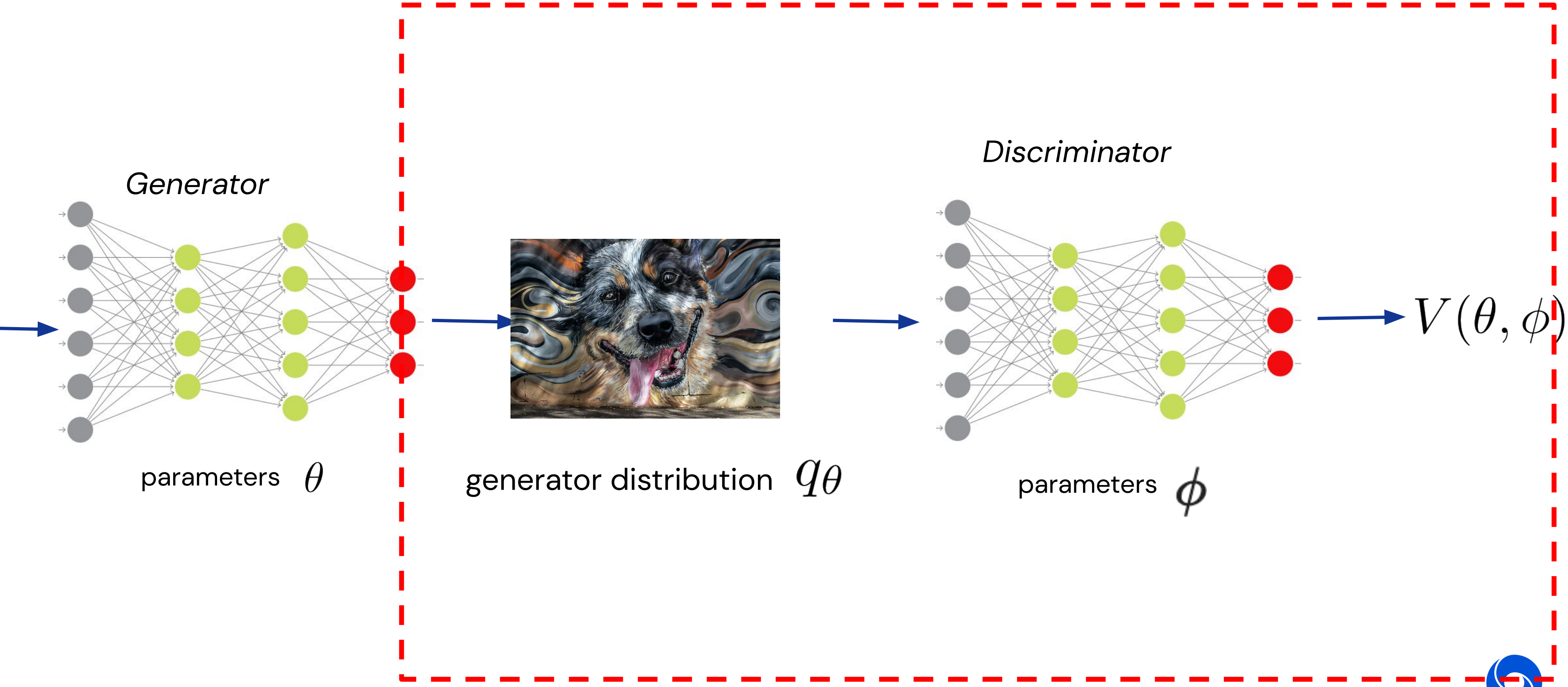
$$\mathbf{E}_{q_\theta(x)} g(x) = \int q_\theta(x) g(x) dx = \int q(z) g(G(z;\theta)) dz = \mathbf{E}_{q(z)} g(G(z;\theta))$$

Want to learn more?

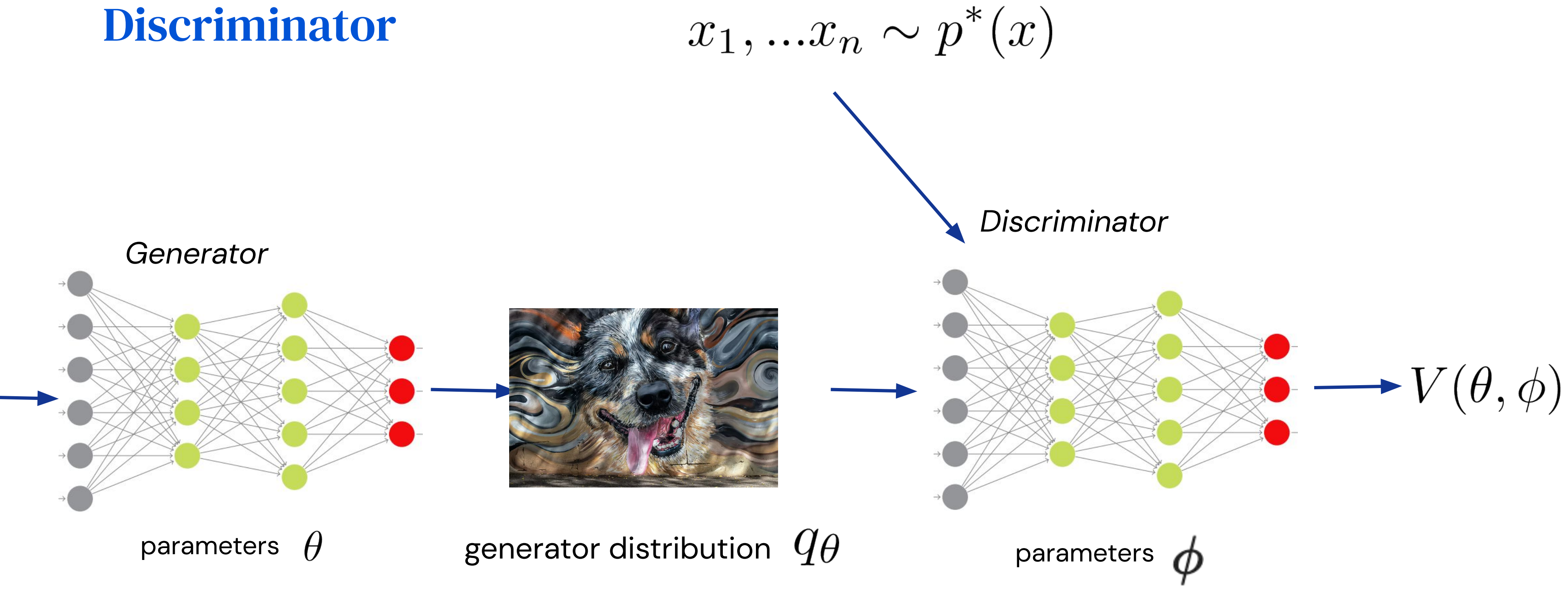
Goodfellow, et al.
Generative adversarial networks. Neurips (2014)

The GAN idea: introduce another model

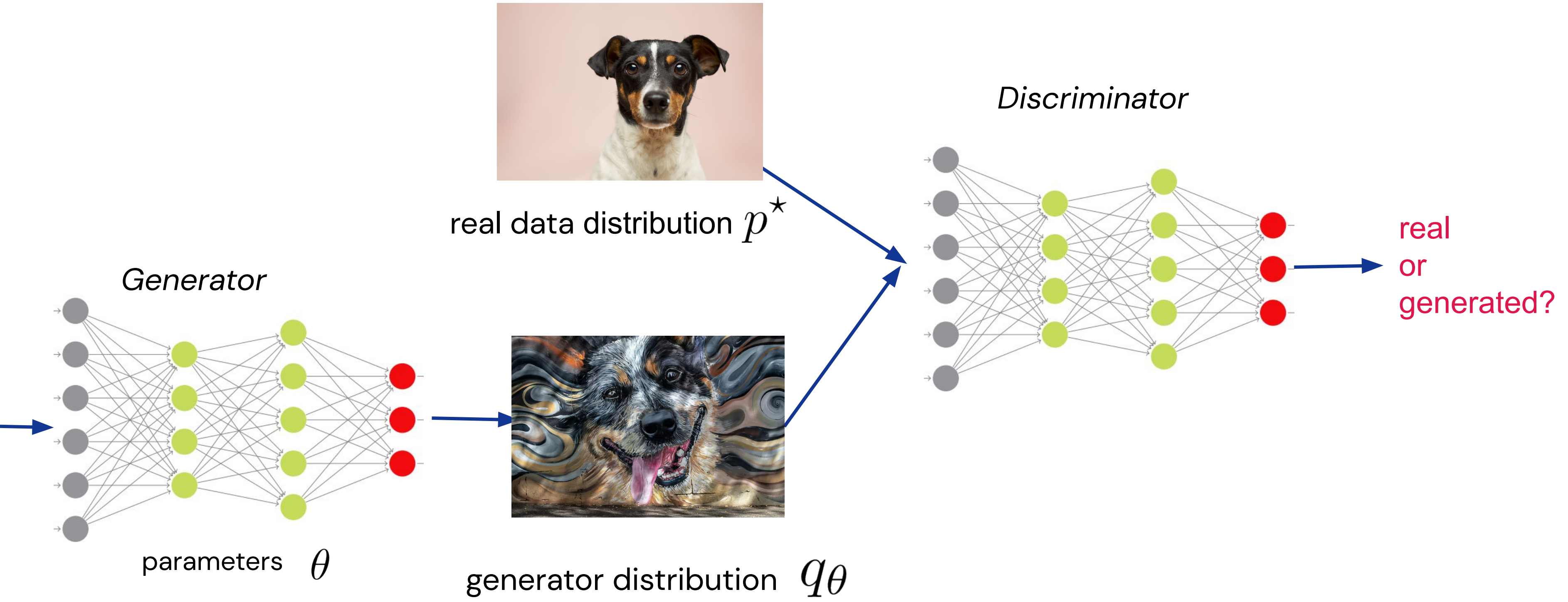
Discriminator



Discriminator



How do we train this model, the discriminator?

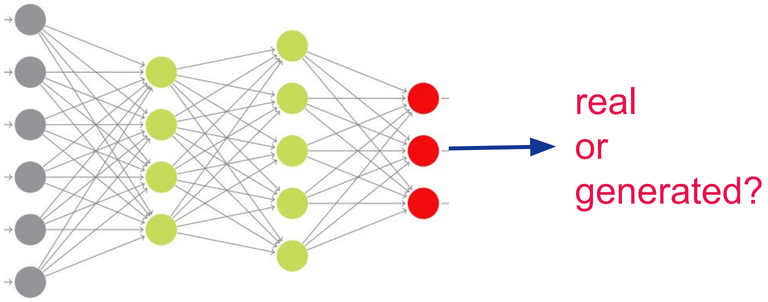


Original Generative Adversarial Network

Want to learn more?

Goodfellow, et al. **Generative adversarial networks..** Neural Information Processing Systems (**2014**)

Discriminator



This can be formalised as a classifier: associate label 1 to real data and label 0 to generated data.

Original Generative Adversarial Network

Want to learn more?

Goodfellow, et al.
Generative adversarial networks. Neurips (2014)

$$V(\theta, \phi) = \mathbb{E}_{p^*(x)} \log D(x;\phi) + \mathbb{E}_{q(z)} \log (1 - D(G(z;\theta);\phi))$$

log-probability that D correctly
predicts real data **x** are real

Original Generative Adversarial Network

Want to learn more?

Goodfellow, et al.
Generative adversarial networks. Neurips (2014)

$$V(\theta, \phi) = \mathbb{E}_{p^*(x)} \log D(x;\phi) + \mathbb{E}_{q(z)} \log\left(1 - D(G(z;\theta);\phi)\right)$$

log-probability that D correctly predicts generated data G(**z**) are generated

Original Generative Adversarial Network

Want to learn more?

Goodfellow, et al.
Generative adversarial networks. Neurips (2014)

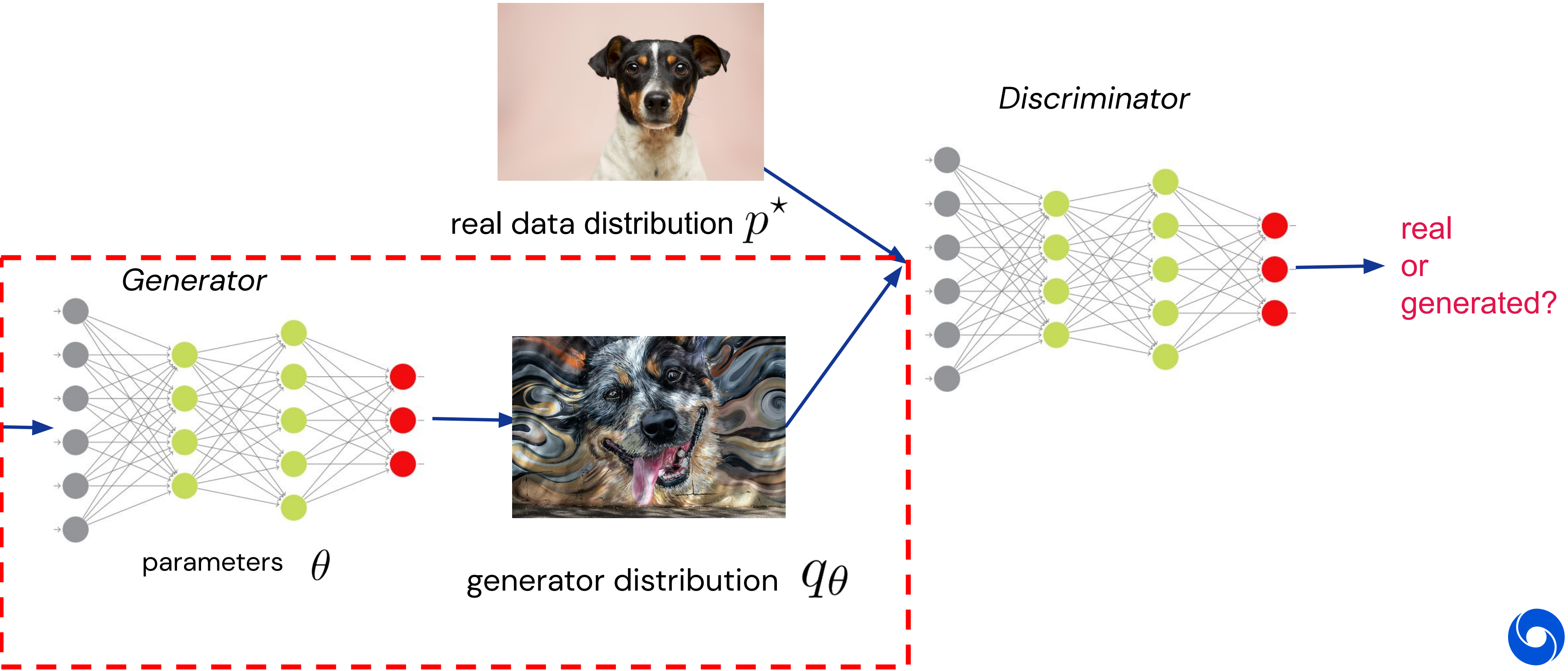
$$V(\theta, \phi) = \mathbb{E}_{p^*(x)} \log D(x;\phi) + \mathbb{E}_{q(z)} \log (1 - D(G(z;\theta);\phi))$$

$$\max_{\phi} V(\theta, \phi)$$

Discriminator's (D) goal: maximize prediction accuracy

(classify real data as real, and generated data as generated)

How do we train the generator?



Original Generative Adversarial Network

Want to learn more?

Goodfellow, et al.
Generative adversarial networks. Neurips (2014)

$$V(\theta, \phi) = \mathbb{E}_{p^*(x)} \log D(x; \phi) + \mathbb{E}_{q(z)} \log (1 - D(G(z; \theta); \phi))$$

$$\min_{\theta} \max_{\phi} V(\theta, \phi)$$

Generator's goal: minimise discriminator prediction accuracy

Original Generative Adversarial Network

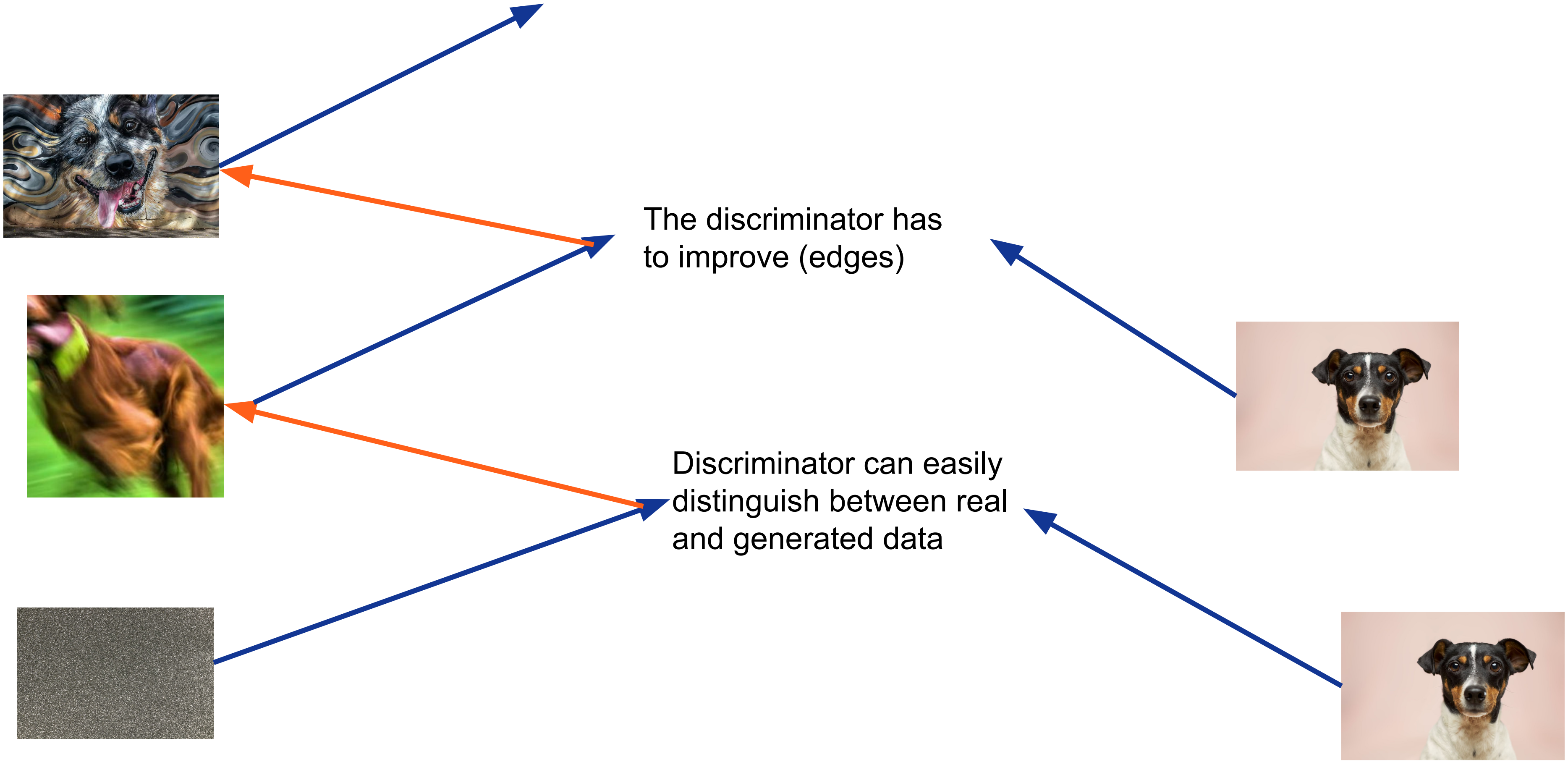
Want to learn more?

Goodfellow, et al.
Generative adversarial networks. Neurips (2014)

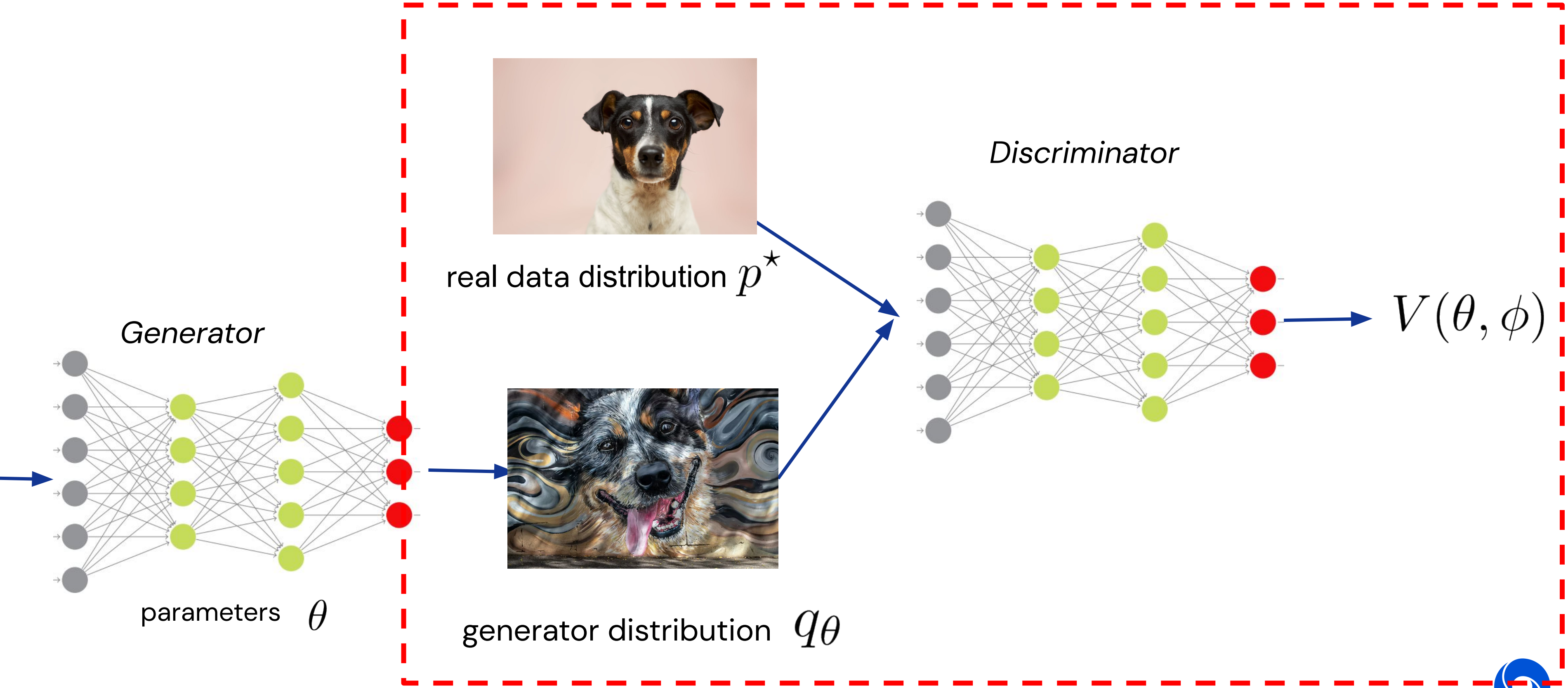
$$V(\theta, \phi) = \mathbb{E}_{p^*(x)} \log D(x; \phi) + \mathbb{E}_{q(z)} \log (1 - D(G(z; \theta); \phi))$$

$$\min_{\theta} \max_{\phi} V(\theta, \phi)$$

Generator's goal: minimise discriminator prediction accuracy

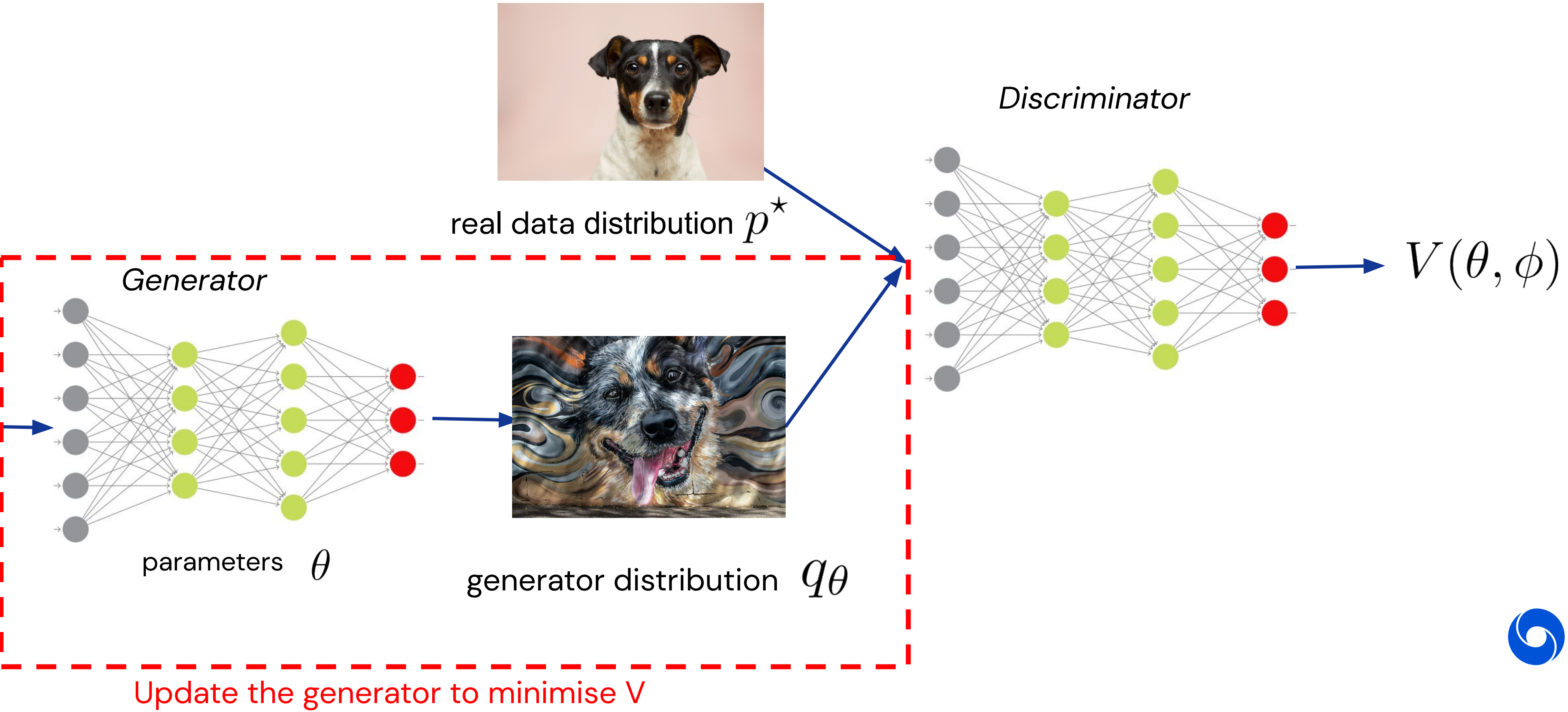

The discriminator has
to improve (edges)
Discriminator can easily
distinguish between real
and generated data

Update the discriminator



Update the discriminator to maximise V

Update the generator



Training GANs

Want to learn more?

Goodfellow, et al.
Generative adversarial networks. Neurips (2014)

```
while training:
    for i in 1... number_discriminator_updates :
        update the discriminator parameters to maximise V

    update the generator using the new discriminator
    parameters
        to minimise V
```

Generative adversarial networks and divergence minimisation

Generative models as divergence or distance minimization

- Generative models **often to minimize a divergence or distance**.
- Most common: Maximum likelihood (KL divergence).

Why divergence/distance minimization?

$$D(p^*, q_\theta) \geq 0$$

$$D(p^*, q_\theta) = 0 \implies p^* = q_\theta$$

Are GANs doing divergence minimization?

Want to learn more?

Goodfellow, et al.
Generative adversarial networks. Neurips (2014)

$$\min_{\theta} \max_{\phi} V(\theta, \phi) = \mathbb{E}_{p^*(x)} \log D(x; \phi) + \mathbb{E}_{q(z)} \log (1 - D(G(z; \theta); \phi))$$

**If the discriminator (D) is optimal:
the generator is minimizing the Jensen Shannon divergence
between the true and generated distributions.**

Are GANs doing divergence minimization?

Want to learn more?

Goodfellow, et al.
Generative adversarial networks. Neurips (2014)

$$\min_\theta \max_\phi V(\theta, \phi) = \mathbb{E}_{p^*(x)} \log D(x; \phi) + \mathbb{E}_{q(z)} \log (1 - D(G(z; \theta); \phi))$$

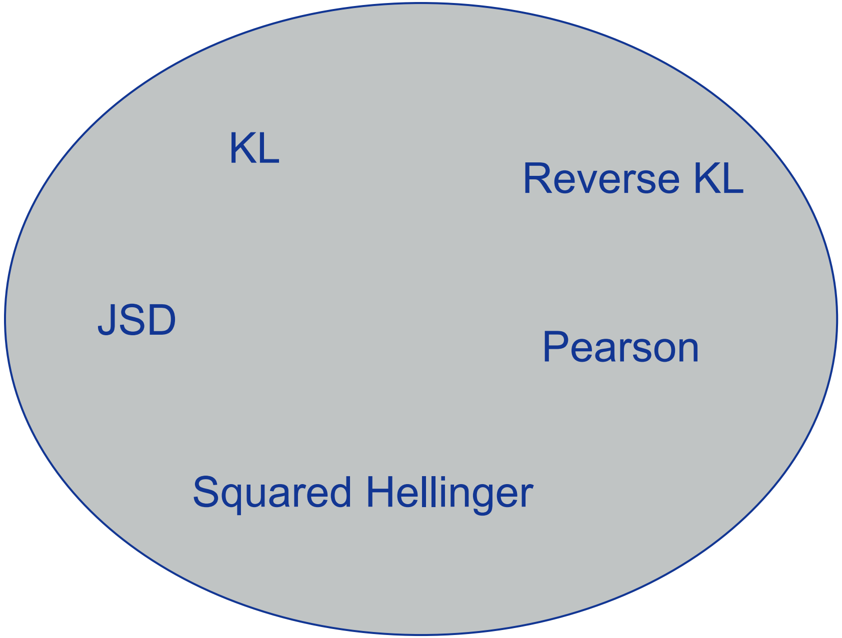
If the discriminator (D) is optimal:
the generator is minimizing the Jensen Shannon divergence between the true and generated distributions.

Connection to optimality:

$$JSD(p^* || q_\theta) = 0 \implies p^* = q_\theta$$

From f-divergences

f-divergences

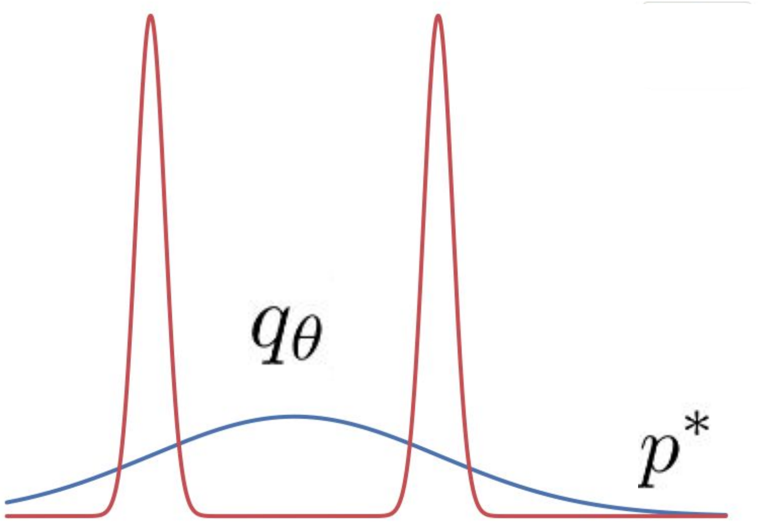


Effects of the choice of divergence

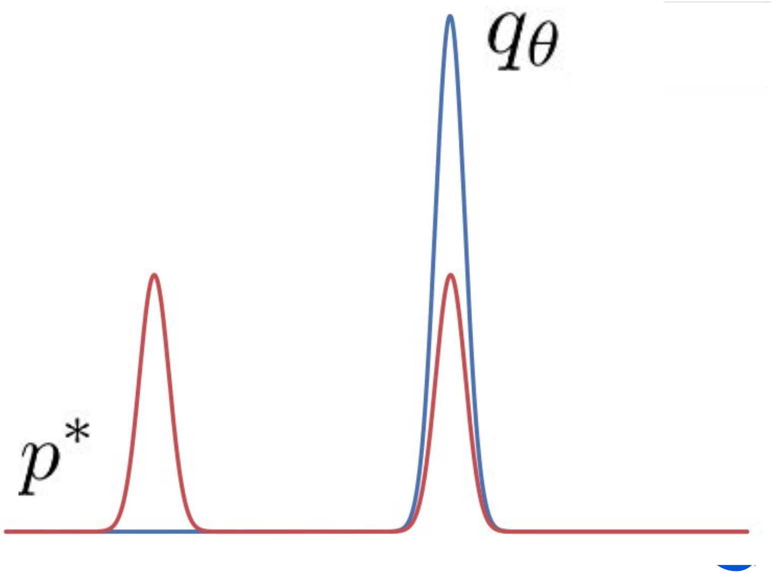
Want to learn more?

Goodfellow, et al. **Neurips 2016 Tutorial: Generative Adversarial Networks.** Arxiv (2016)

$$\mathbf{KL}(p^* || q_\theta)$$



$$\mathbf{KL}(q_\theta || p^*)$$



f-divergences

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips (**2016**)

$$D_f(p^*, q_\theta) = \mathbb{E}_{q_\theta(x)} f\left(\frac{p^*(x)}{q_\theta(x)}\right)$$

f convex, semi continuous and f(1) = 0.

Examples of f-divergences

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips (**2016**)

Name	$D_f(P\|Q)$	$f(u)$
Kullback-Leibler	$\int p(x) \log \frac{p(x)}{q(x)} \, \mathrm{d}x$	$u \log u$
Reverse KL	$\int q(x) \log \frac{q(x)}{p(x)} \, \mathrm{d}x$	$-\log u$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)} \, \mathrm{d}x$	$(u-1)^2$
Squared Hellinger	$\int \left(\sqrt{p(x)} - \sqrt{q(x)}\right)^2 \, \mathrm{d}x$	$(\sqrt{u}-1)^2$
Jensen-Shannon	$\frac{1}{2} \int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} \, \mathrm{d}x$	$-(u+1) \log \frac{1+u}{2} + u \log u$

Challenge with f-divergences

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips **(2016)**

unknown!

$$D_f(p^*, q_\theta) = \mathbb{E}_{q_\theta(x)} f\left(\frac{p^*(x)}{q_\theta(x)}\right)$$

Variational bound on *f*-divergences

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips (**2016**)

$$D_f(p^*, q_\theta) = \mathbb{E}_{q_\theta(x)} f\left(\frac{p^*(x)}{q_\theta(x)}\right)$$

f convex:

$$f(x) = \sup_t tx - f^\dagger(t)$$

Variational bound on *f*-divergences

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips (**2016**)

$$
\begin{aligned}
D_f(p^*, q_\theta) &= \int p(x) f\left(\frac{p^*(x)}{q_\theta(x)}\right) dx \\
&= \int p(x) \sup_t \left[t\frac{p^*(x)}{q_\theta(x)} - f^\dagger(t)\right] dx \\
&= \int \sup_{t(x)} p(x) \left[t(x)\frac{p^*(x)}{q_\theta(x)} - f^\dagger(t(x))\right] dx \\
&= \int \sup_{t(x)} t(x)p^*(x) - q_\theta(x) f^\dagger(t(x)) dx \\
&= \sup_{t:\mathcal{X}\to\mathbb{R}} \int t(x)p^*(x) - q_\theta(x) f^\dagger(t(x)) dx \\
&= \sup_{t:\mathcal{X}\to\mathbb{R}} \mathbb{E}_{p^*(x)} t(x) - \mathbb{E}_{q_\theta(x)} f^\dagger(t(x)) dx
\end{aligned}
$$

Variational bound on *f*-divergences

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips (**2016**)

$$D_f(p^*, q_\theta) = \sup_{t:\mathcal{X}\to\mathbb{R}} \mathbb{E}_{p^*(x)} t(x) - \mathbb{E}_{q_\theta(x)} f^\dagger(t(x))$$

Variational bound on f-divergences

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips (**2016**)

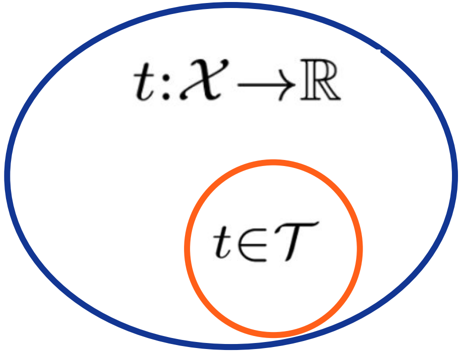
$$
\begin{aligned}
D_f(p^*, q_\theta) &= \sup_{t:\mathcal{X}\to\mathbb{R}} \mathbb{E}_{p^*(x)} t(x) - \mathbb{E}_{q_\theta(x)} f^\dagger(t(x)) \\
&\geq \sup_{t\in\mathcal{T}} \mathbb{E}_{p^*(x)} t(x) - \mathbb{E}_{q_\theta(x)} f^\dagger(t(x))
\end{aligned}
$$

Variational bound on *f*-divergences

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips (**2016**)

$$
\begin{aligned}
D_f(p^*, q_\theta) &= \sup_{t:\mathcal{X}\to\mathbb{R}} \mathbb{E}_{p^*(x)} t(x) - \mathbb{E}_{q_\theta(x)} f^\dagger(t(x)) \\
&\geq \sup_{t\in\mathcal{T}} \mathbb{E}_{p^*(x)} t(x) - \mathbb{E}_{q_\theta(x)} f^\dagger(t(x))
\end{aligned}
$$



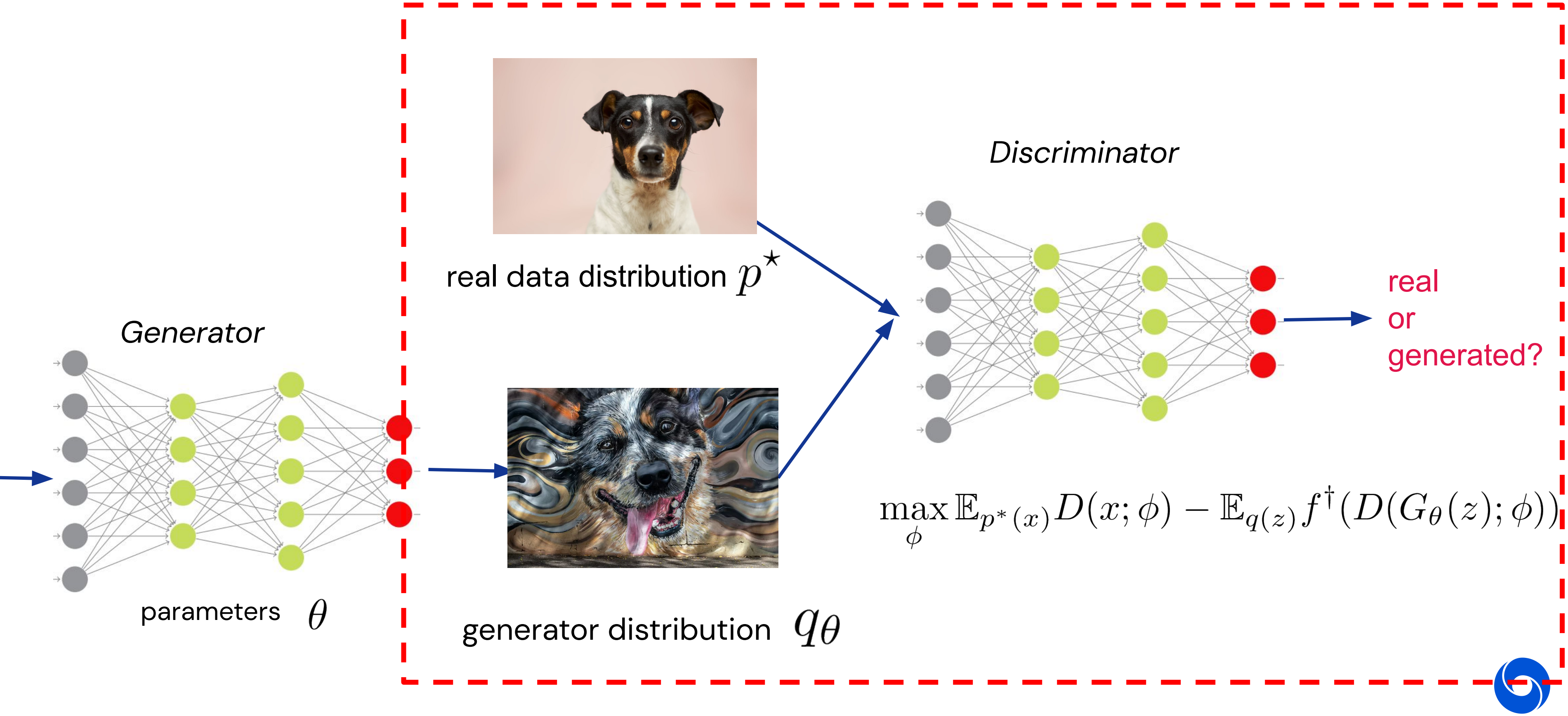
So far… evaluating *f*-divergences

$$D_f(p^*, q_\theta) = \mathbb{E}_{q_\theta(x)} f\left(\frac{p^*(x)}{q_\theta(x)}\right)$$

$$\sup_{t \in \mathcal{T}} \mathbb{E}_{p^*(x)} t(x) - \mathbb{E}_{q_\theta(x)} f^\dagger(t(x))$$

$$\max_{\phi} \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z)} f^\dagger(D(G(z;\theta);\phi))$$

learning a *discriminator* to distinguish between samples from two distributions


Discriminator
real data distribution $p^\star$
real
or
generated?
Generator
$\max_{\phi} \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z)} f^\dagger(D(G_\theta(z);\phi))$
parameters θ
generator distribution q_θ

From *f*-divergences to *f*-GAN

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips (**2016**)

evaluation

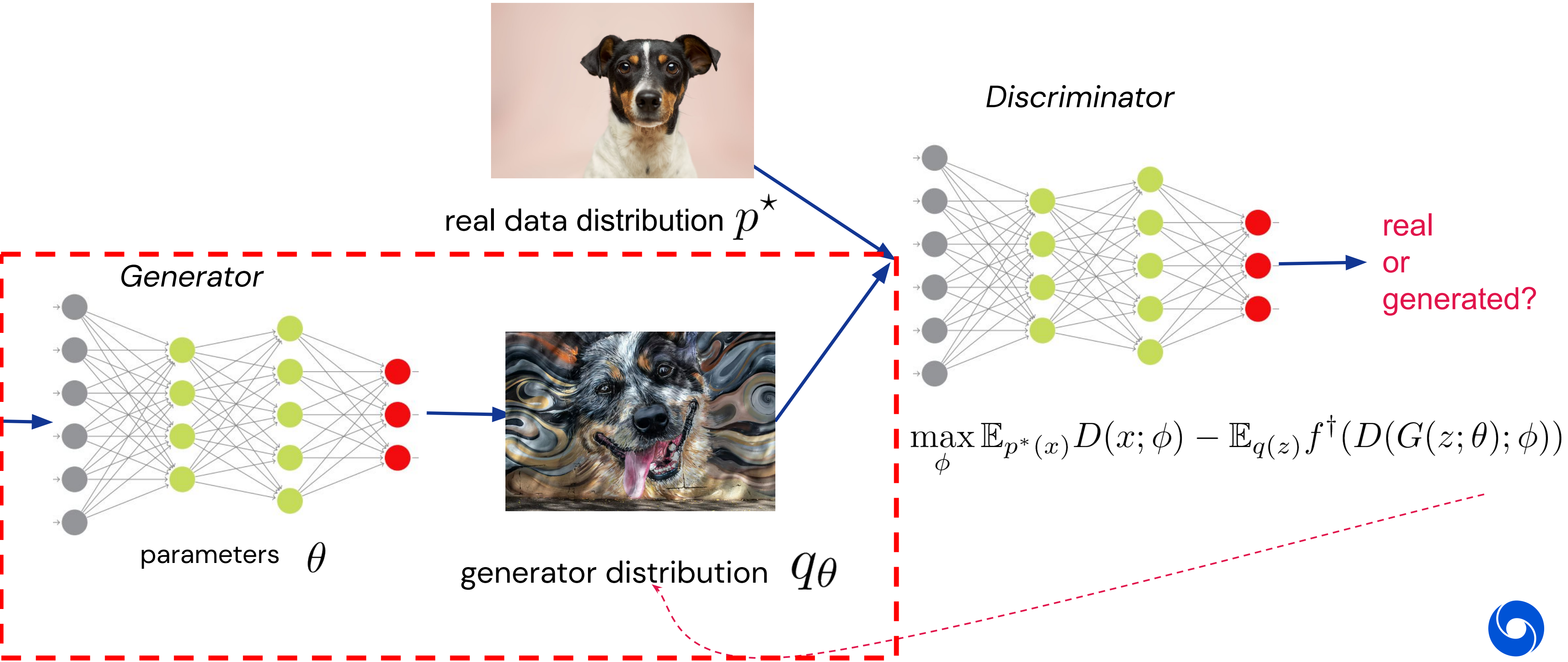
$$D_f(p^*, q_\theta) = \mathbb{E}_{q_\theta(x)} f\left(\frac{p^*(x)}{q_\theta(x)}\right)$$

approximation

$$\max_\phi \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z)} f^\dagger(D(G(z;\theta);\phi))$$

learning

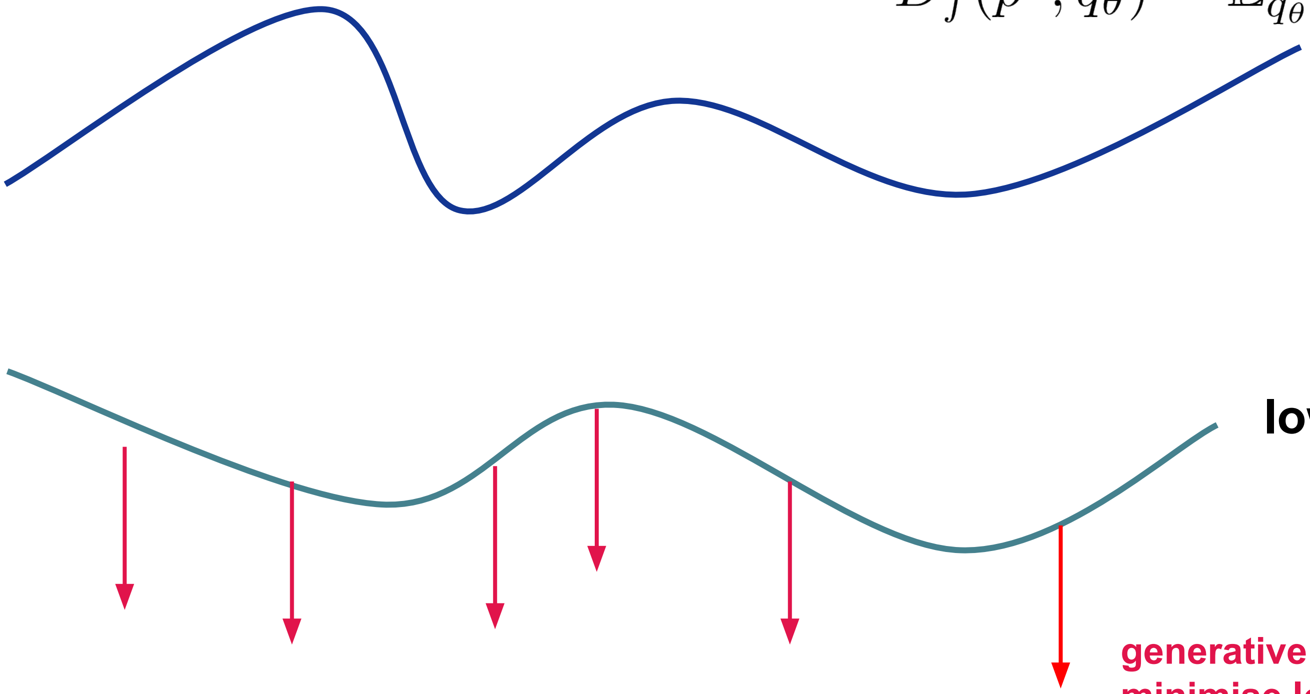
$$\min_\theta \max_\phi \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z)} f^\dagger(D(G(z;\theta);\phi))$$



real data distribution $p^\star$
Discriminator
real
or
generated?
Generator
parameters θ
generator distribution q_θ
$\max_{\phi} \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z)} f^\dagger(D(G(z;\theta);\phi))$

Challenge: minising a lower bound

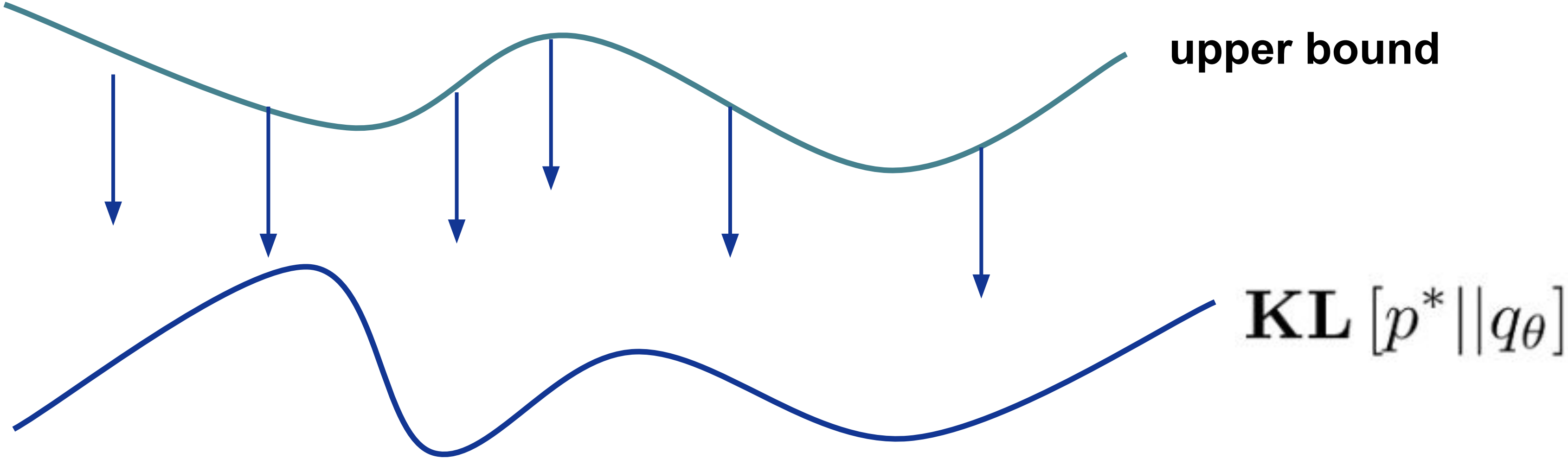
$$D_f(p^*, q_\theta) = \mathbb{E}_{q_\theta(x)} f\left(\frac{p^*(x)}{q_\theta(x)}\right)$$



lower bound

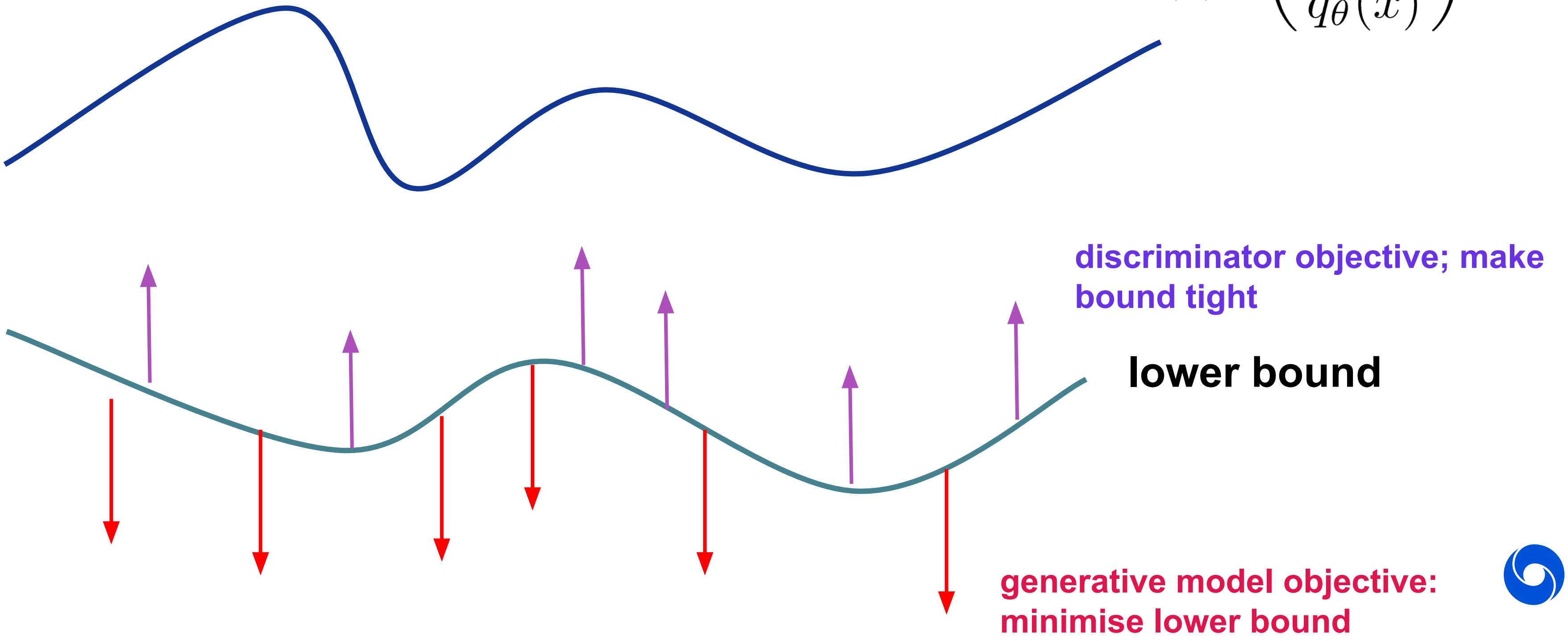
**generative model objective:
minimise lower bound**

Contrast with VAEs

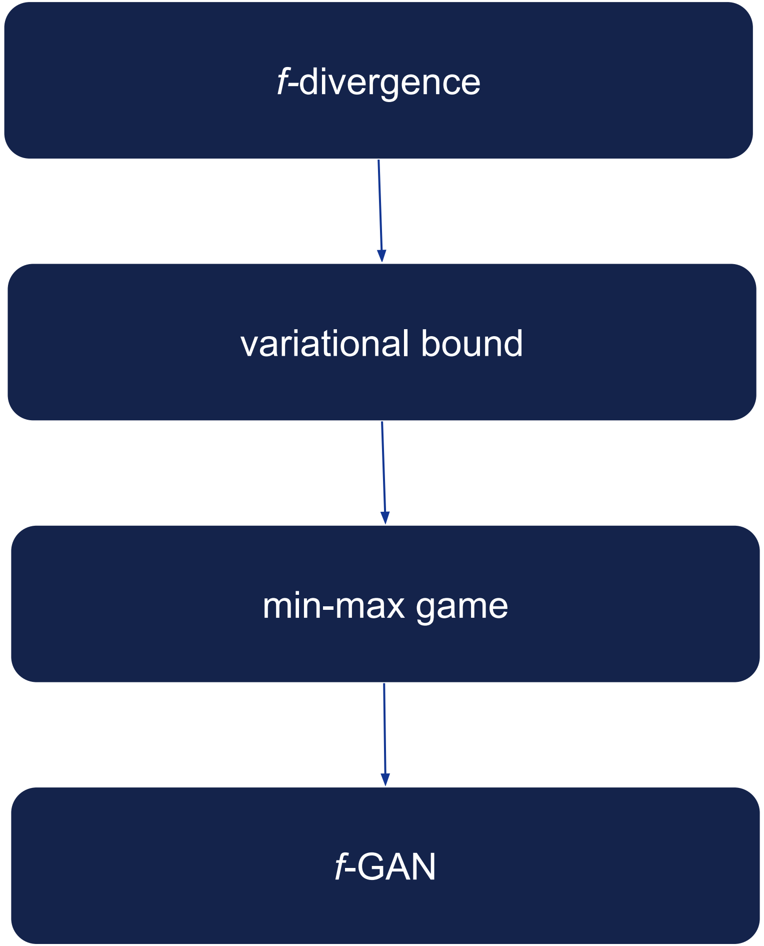


Still works well in practice!

$$D_f(p^*, q_\theta) = \mathbb{E}_{q_\theta(x)} f\left(\frac{p^*(x)}{q_\theta(x)}\right)$$



Recipe so far



From Integral Probability Metrics

Integral probability metrics are distances, not divergences

Divergence

$$D(p^*, q_\theta) \geq 0$$

$$D(p^*, q_\theta) = 0 \implies p^* = q_\theta$$

Distance

$$D(p^*, q_\theta) \geq 0$$

$$D(p^*, q_\theta) = 0 \implies p^* = q_\theta$$

$$D(p^*, q_\theta) = D(q_\theta, p^*)$$

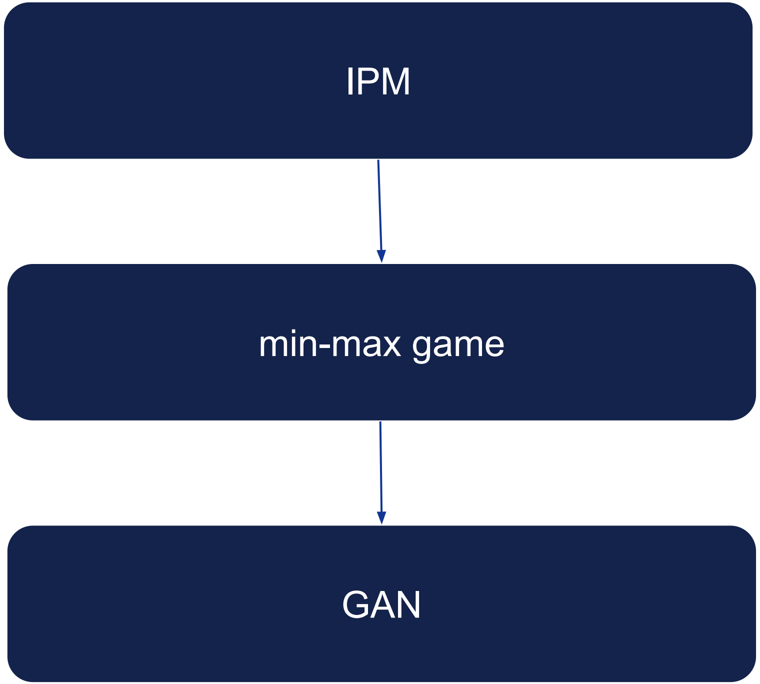
$$D(p^*, q_\theta) \leq D(p, q_\theta) + D(p, p^*)$$

Integral Probability Metrics

$$D_{\mathcal{F}}(p^*, q_\theta) = \sup_{f \in \mathcal{F}} \left| \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x) \right|$$

Different IPM instatiations given by different family of functions.

Integral Probability Metrics



Wasserstein Distance

Want to learn more?

Arjovsky, et al
Wasserstein GAN
ICML (2017)

$$W(p^*, q_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$

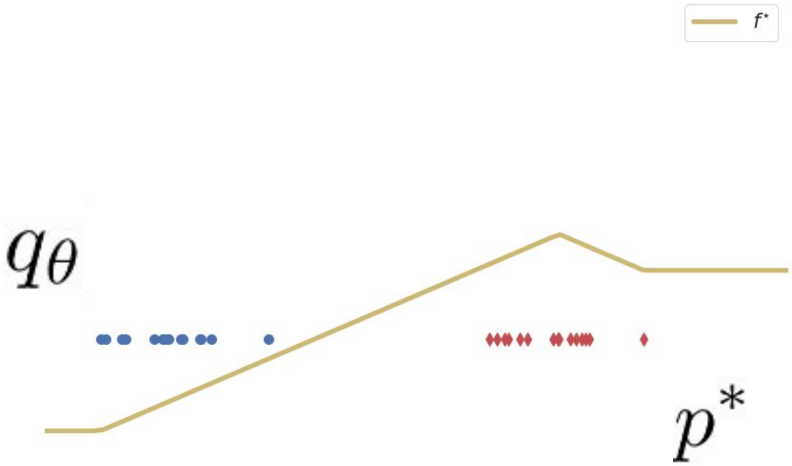
$$|f(x) - f(y)| \leq |x - y|$$

Wasserstein Distance

Want to learn more?

Arjovsky, et al
Wasserstein GAN
ICML (2017)

$$W(p^*, q_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$



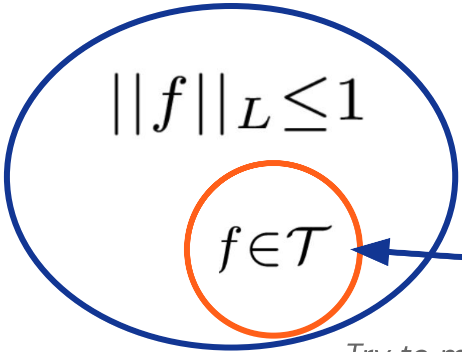
Estimating the Wasserstein Distance

Want to learn more?

Arjovsky, et al
Wasserstein GAN
ICML (**2017**)

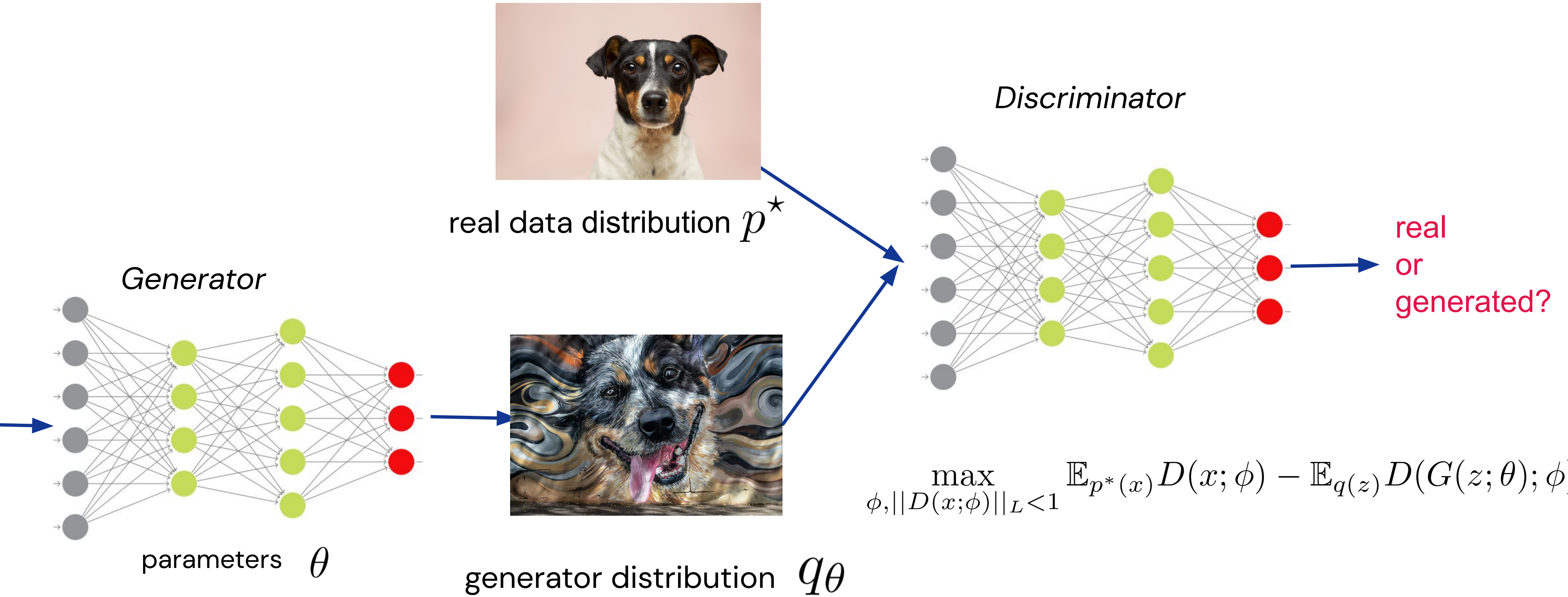
$$W(p^*, q_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$

$$\geq \sup_{f \in \mathcal{T}} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$



Neural network family of functions

Try to make D is 1-Lipschitz via gradient penalties, spectral normalization, weight clipping.



$$\max_{\phi, ||D(x;\phi)||_L < 1} \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z)} D(G(z;\theta);\phi)$$

Wasserstein GAN

Want to learn more?

Arjovsky, et al
Wasserstein GAN
ICML (2017)

$$W(p^*, q_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$

$$\geq \sup_{f \in \mathcal{T}} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$

Model Learning

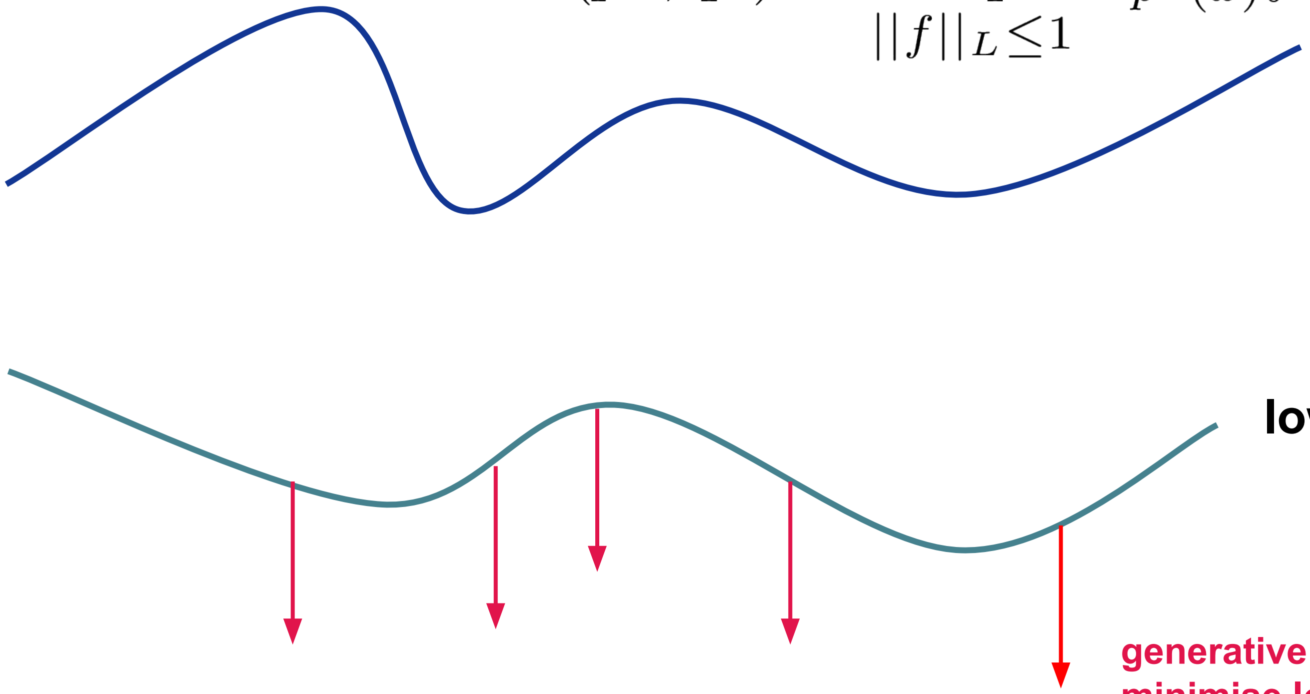
Wasserstein GAN

$$\min_\theta \max_{\phi, ||D(x;\phi)||_L < 1} \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z)} D(G(z;\theta);\phi)$$

Try to make D is 1-Lipschitz via gradient penalties, spectral normalization, weight clipping.

Still minising a lower bound

$$W(p^*, q_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$



lower bound

generative model objective: minimise lower bound

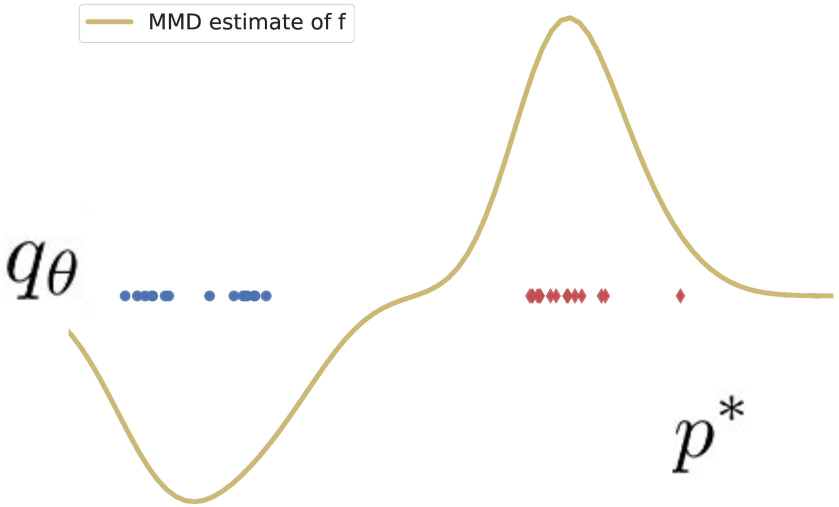
MMD

Want to learn more?

Li, et al **MMD GAN: Towards Deeper Understanding of Moment Matching Network.** Neurips (2017)

$$\mathrm{MMD}(p^*, q_\theta) = \sup_{||f||_{\mathcal{H}} \leq 1} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$

$\mathcal{H}$ *is a RKHS.*



MMD

Want to learn more?

Li, et al **MMD GAN: Towards Deeper Understanding of Moment Matching Network.** Neurips (2017)

$$\mathrm{MMD}(p^*, q_\theta) = \sup_{||f||_{\mathcal{H}} \leq 1} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$

Kernel choice (feature learning)

$$\mathrm{MMD}(p^*, q_\theta) = \sup_{||f||_{\mathcal{H}_\phi} \leq 1} \mathbb{E}_{p^*(x)} f(\phi(x)) - \mathbb{E}_{q_\theta(x)} f(\phi(x))$$

Choose kernel with learned features (via D)

$$K_\phi(x, x') = K(\phi(x), \phi(x'))$$

MMD

Want to learn more?

Li, et al **MMD GAN: Towards Deeper Understanding of Moment Matching Network.** Neurips (2017)

$$\mathrm{MMD}(p^*, p) = \sup_{||f||_{\mathcal{H}_\phi} \leq 1} \mathbb{E}_{p^*(x)} f(\phi(x)) - \mathbb{E}_{p(x)} f(\phi(x))$$

Model learning

MMD-GAN

$$\min_{\theta} \max_{\phi, ||D(x;\phi)||_H < 1} \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z))} D(G(z;\theta);\phi)$$

Choose kernel with learned features (via D)

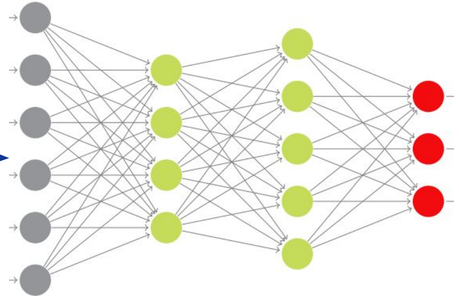
$$K_\phi(x, x') = K(\phi(x), \phi(x'))$$

Distributional view of GANs

real data distribution $p^\star$

Distributional divergences and distances

Generator



parameters θ

generator distribution q_θ

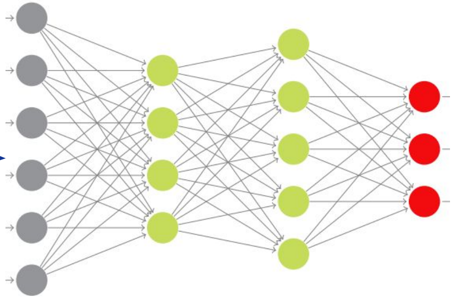
***Approximated or estimated using a learned discriminator.*

Distributional view of GANs

real data distribution $p^\star$

Distributional divergences and distances

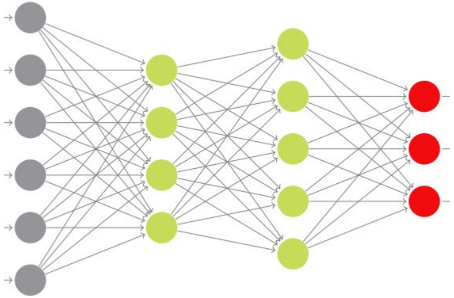
Generator



parameters θ

generator distribution q_θ

***Approximated or estimated using a learned discriminator.*



Why train a GAN instead of doing divergence minimization?

- Model type
- Computational Intractability
- Smooth learning signal
- Learned “divergence“

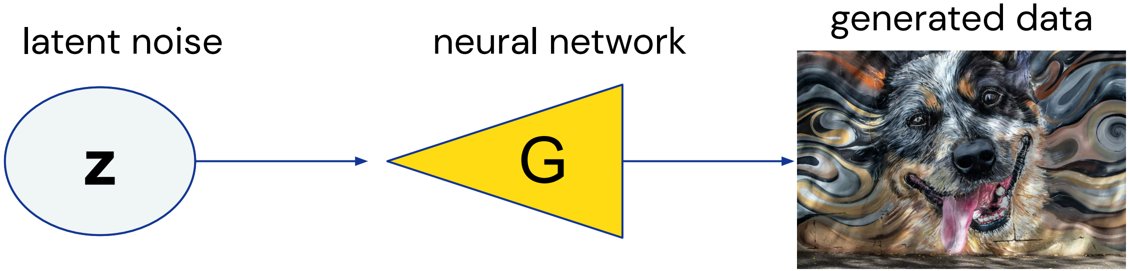
Implicit models and KL divergence

Want to learn more?

Nowozin, et al **f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization.** Neurips (**2016**)

$$\mathbf{KL}\left[p^*||q_\theta\right] = \int p^*(x)\log\frac{p^*(x)}{q_\theta(x)}dx$$

we do not have access to the explicit distribution p(x).



f-GAN $\min_\theta \max_\phi \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z)} f^\dagger(D(G(z;\theta);\phi))$

Wasserstein distance & computational intractability

Want to learn more?

Arjovsky, et al
Wasserstein GAN
ICML (**2017**)

$$W(p^*, q_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{p^*(x)} f(x) - \mathbb{E}_{q_\theta(x)} f(x)$$

Computationally intractable for complex cases.

Wasserstein GAN

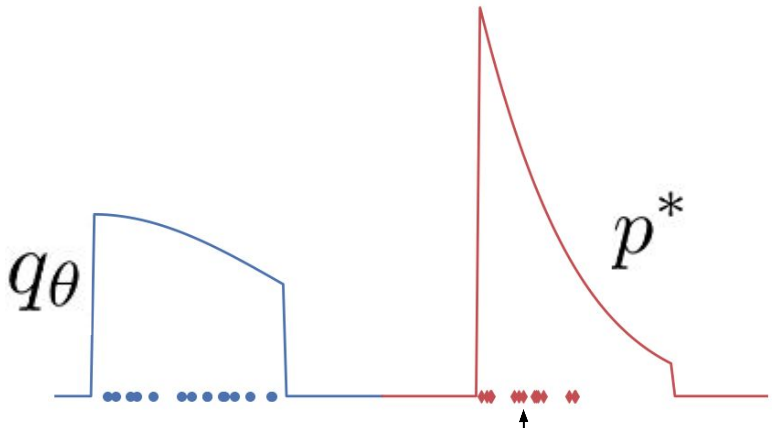
$$\min_{\theta} \max_{\phi, ||D(x;\phi)||_L < 1} \mathbb{E}_{p^*(x)} D(x;\phi) - \mathbb{E}_{q(z)} D(G(z;\theta);\phi)$$

Smooth learning signal

Want to learn more
Gretton, et al
Interpretable comparison of distributions and models.
Neurips Tutorial (**2019**)

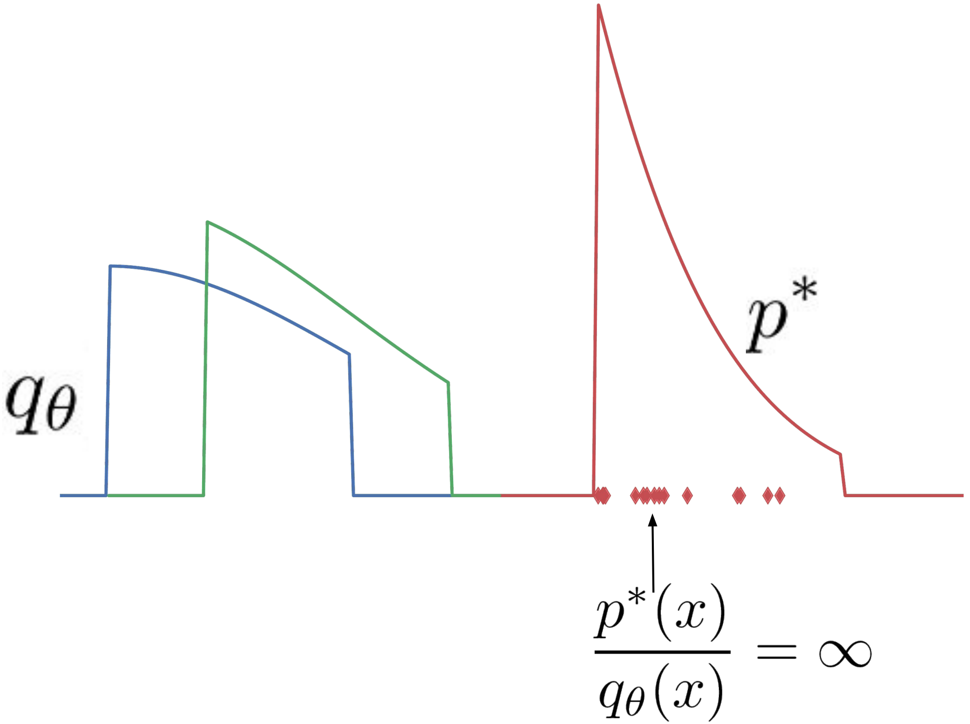
No learning signal from KL/JSD divergence if non-overlapping support between the data and the model.

$$\mathbf{KL}\left[p^* || q_\theta\right] = \infty \qquad \mathbf{JSD}\left[p^* || q_\theta\right] = \log 2$$



$$\frac{p^*(x)}{q_\theta(x)} = \infty$$

Smooth learning signal



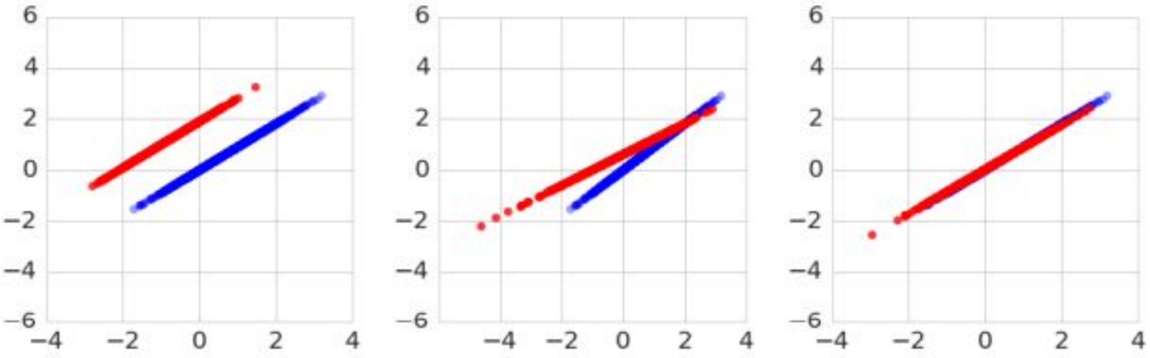
The density ratio jumps to infinity at the data distribution.

Smooth learning signal

Want to learn more?

Fedus, et al **Many paths to equilibrium: GANs do not need to decrease a divergence at every step** . ICLR (2018)

GANs still learn!



(a) Step 0 (b) Step 5000 (c) Step 12500

Red = data
Blue = model (changes in training)

Want to learn more

Gretton, et al
Interpretable comparison of distributions and models.
Neurips Tutorial (2019)

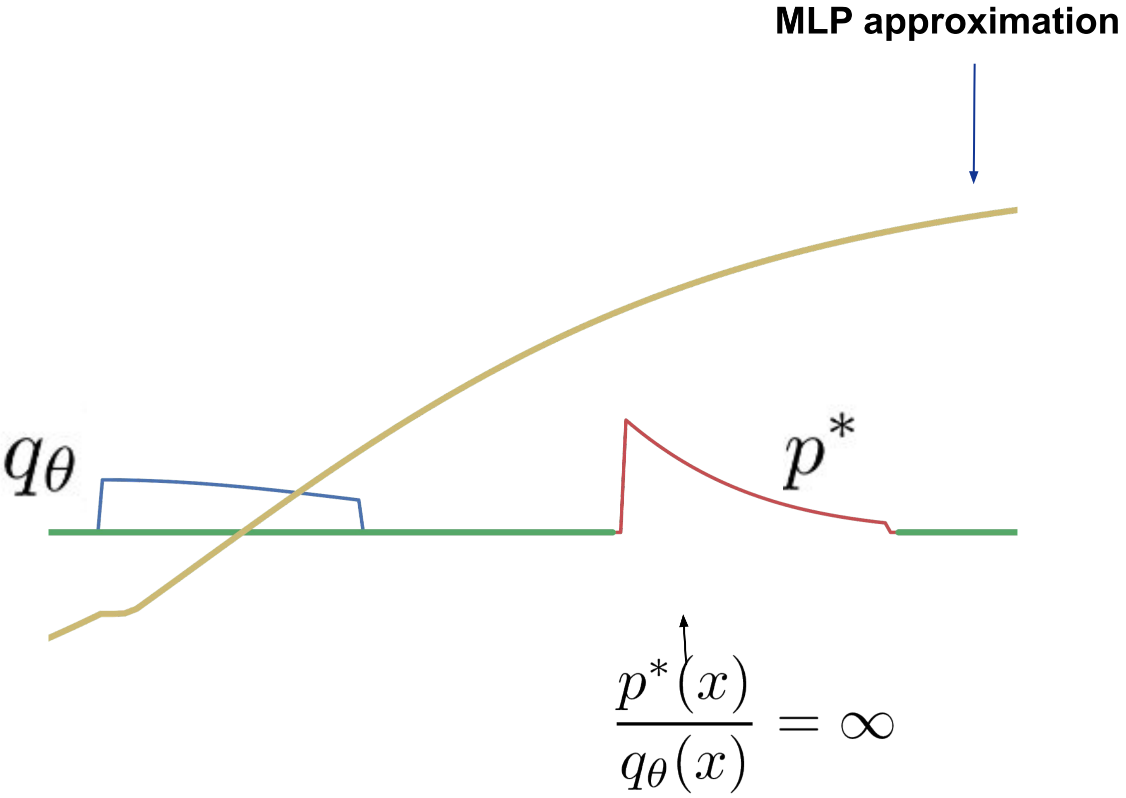
true ratio

$$D_f(p^*, q_\theta) = \int q_\theta(x) f\left(\frac{p^*(x)}{q_\theta(x)}\right)$$

$$\geq \sup_{t \in \mathcal{T}} \mathbb{E}_{p^*(x)} t(x) - \mathbb{E}_{q_\theta(x)} f^\dagger(t(x))$$

ratio approximation (smooth)

Smooth learning signal



Smooth approximation of the density ratio does not go to infinity.

Discriminators as learned “distances”

$$\min_{\theta} \boxed{\max_{\phi} V(\theta, \phi)}$$

D provides a learned distance between the data and sample distributions, using **learned neural network features.**

Discriminators as learned "distances"

Want to learn more?

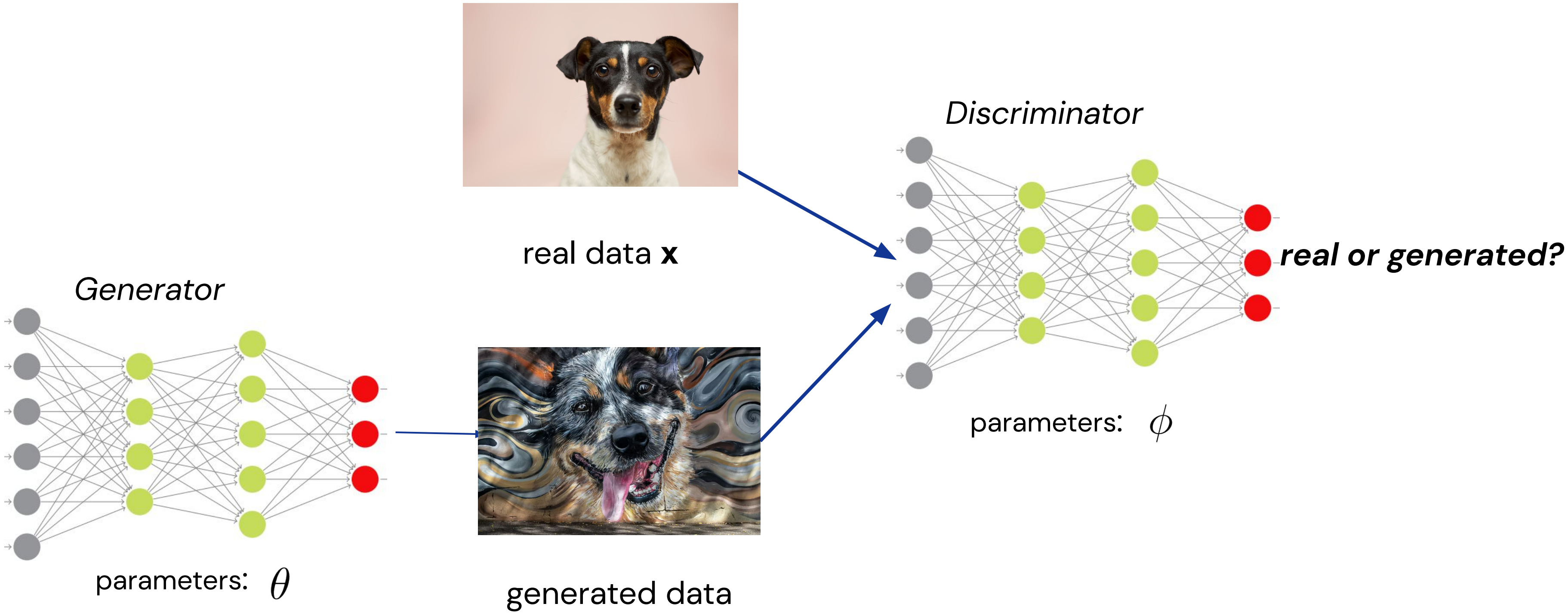
Arora, et al **Generalization and Equilibrium in Generative Adversarial Nets.** ICML(**2017**)

We can think of D (the teacher) as learning a "distance" between the data and model distribution that can provide useful gradients to the model.

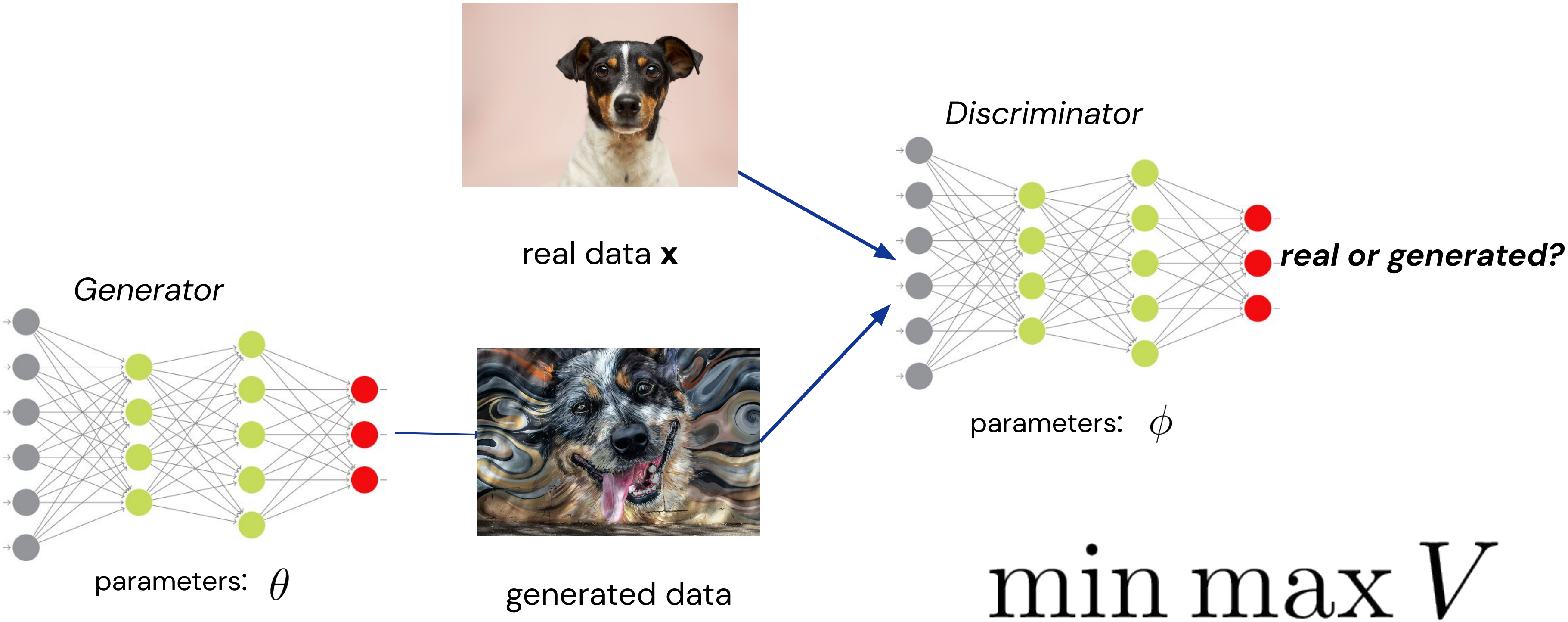
Questions and breathing time!

Generative adversarial networks and optimisation in two player games

Generative adversarial networks



Generative adversarial networks



$$\min_{\theta} \max_{\phi} V$$

Generative Adversarial Networks as zero sum game

$$\min_G \max_D V(D, G)$$

The challenge of optimisation in adversarial games

$$\min_G \max_D V(D, G)$$

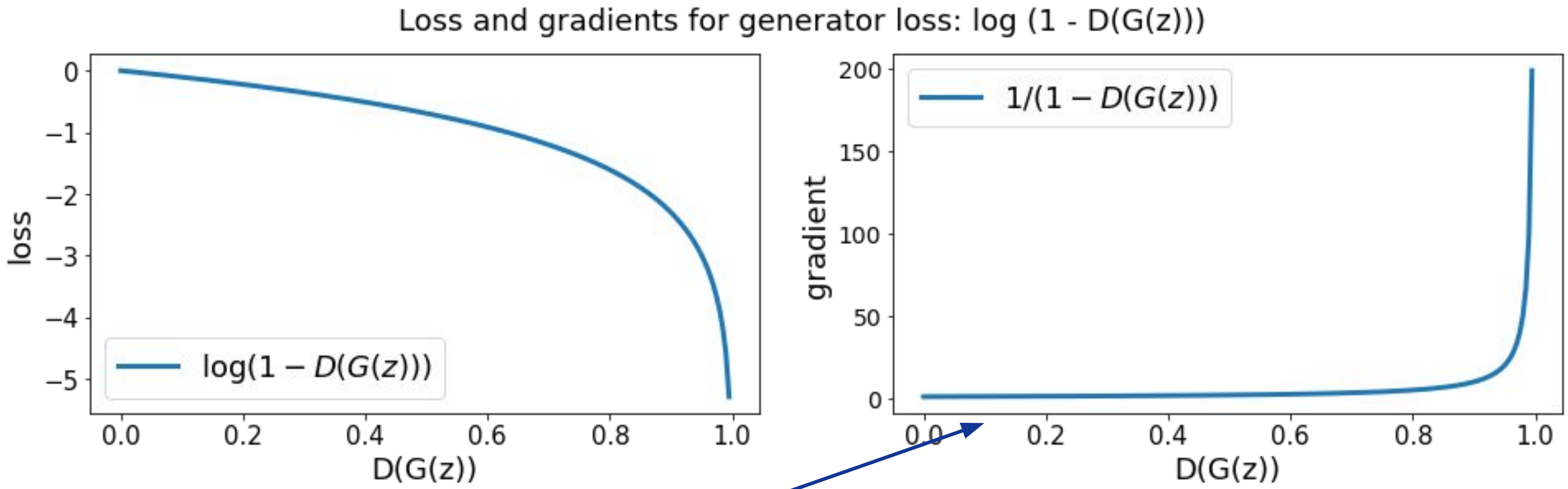
Fully optimising D is not tractable - we are thus not doing divergence minimisation, but this can also introduce optimisation challenges.

Alternating updates

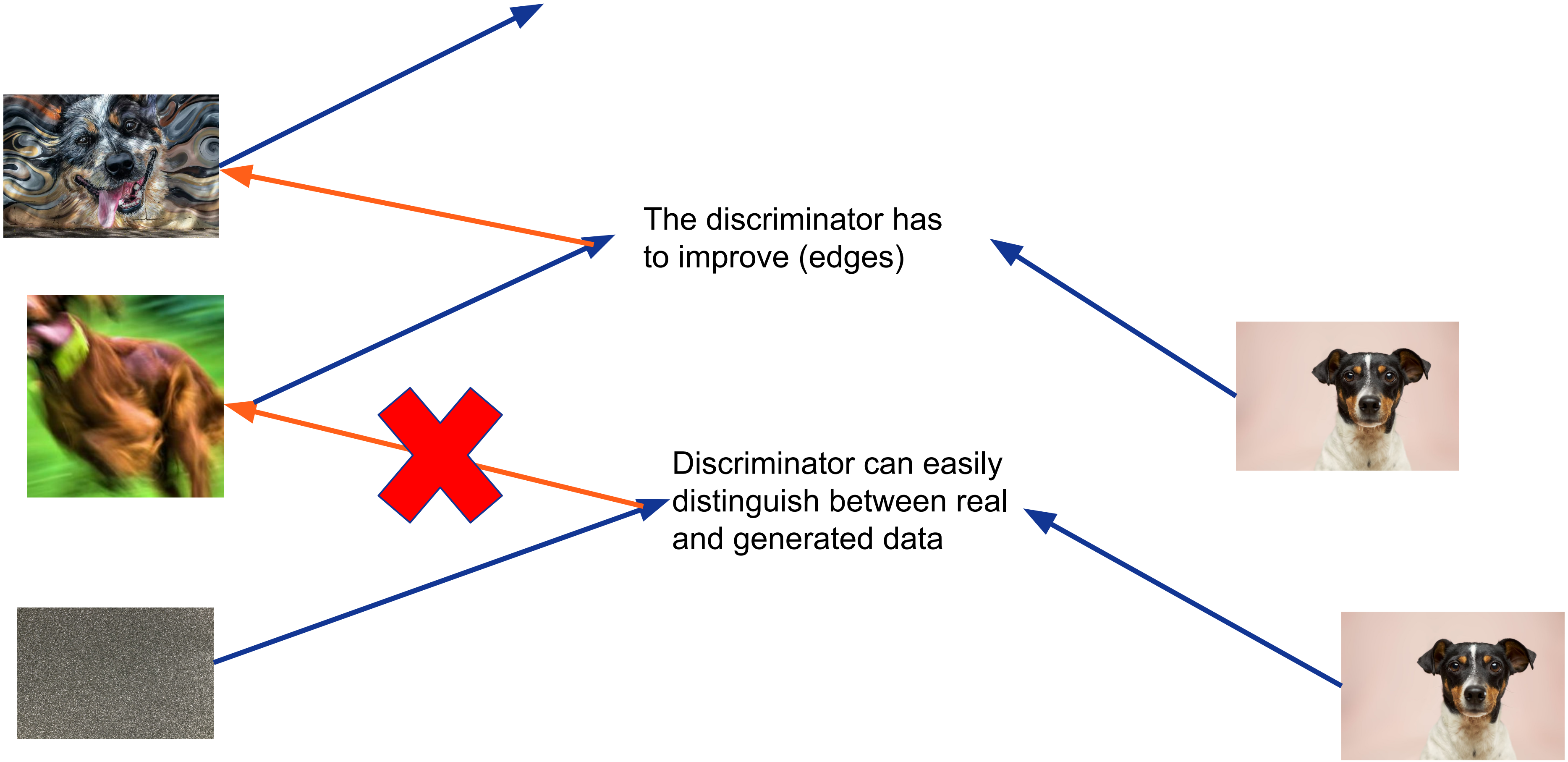
```
while training:
    for i in 1... number_discriminator_updates :
        update the discriminator
    update the generator using the new discriminator
    parameters
```

Gradients matter: the original GAN

$$V(\theta, \phi) = \mathbb{E}_{p^*(x)} \log D(x;\phi) + \boxed{\mathbb{E}_{q(z)} \log(1 - D(G(z;\theta);\phi))}$$



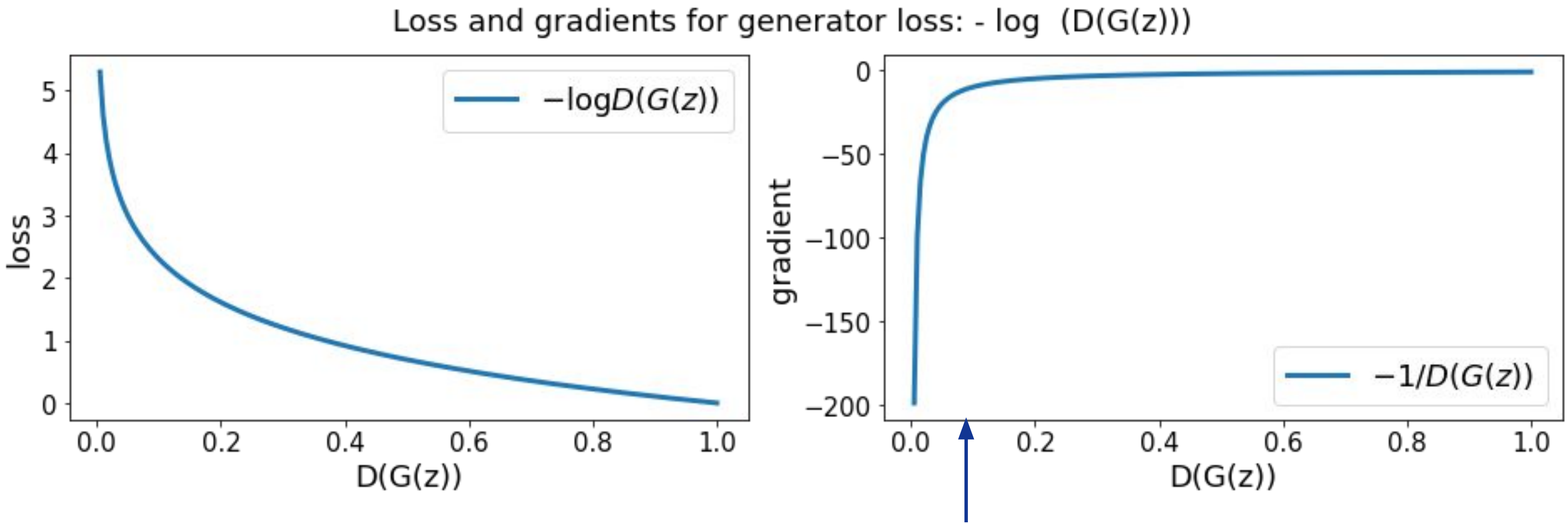
The generator is doing badly: the discriminator is able to confidently detect generated data as generated.
But it gets very little learning signal (gradient is 0)!


The discriminator has
to improve (edges)
Discriminator can easily
distinguish between real
and generated data

Gradients matter: non-saturating loss

Want to learn more?

Goodfellow, et al. **Generative adversarial networks. Neurips** (2014)

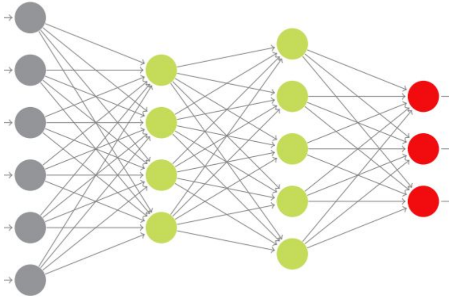


The generator is doing badly: the discriminator is able to confidently detect generated data as generated. Strong learning signal!

Generative adversarial networks

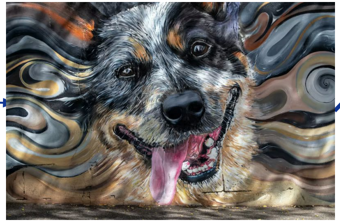
real data **x**

Generator



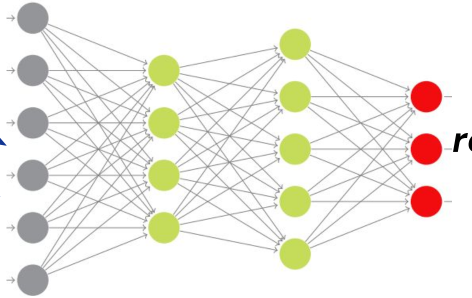
parameters: θ

$$\min_{\theta} L_g(\phi, \theta)$$



generated data

Discriminator



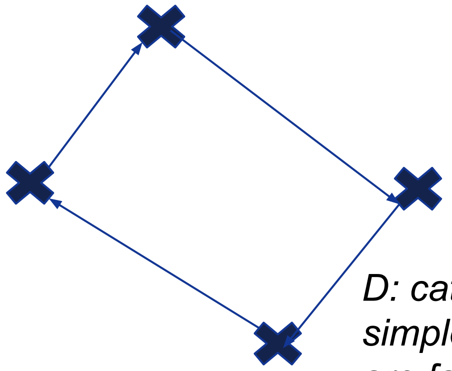
real or generated?

parameters: ϕ

$$\min_{\phi} L_f(\phi, \theta)$$

The challenge of optimisation in adversarial games - sketch

Data: images of cats and dogs



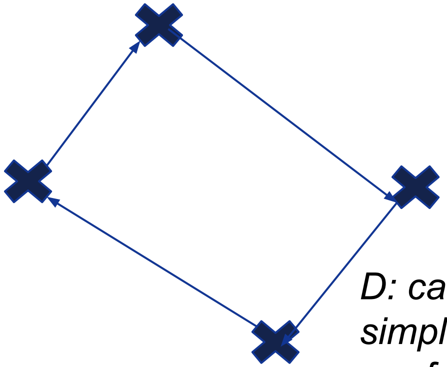
The challenge of optimisation in adversarial games - sketch

Mode hopping

D: catches up and takes simple solution that cats are real and dogs are fake

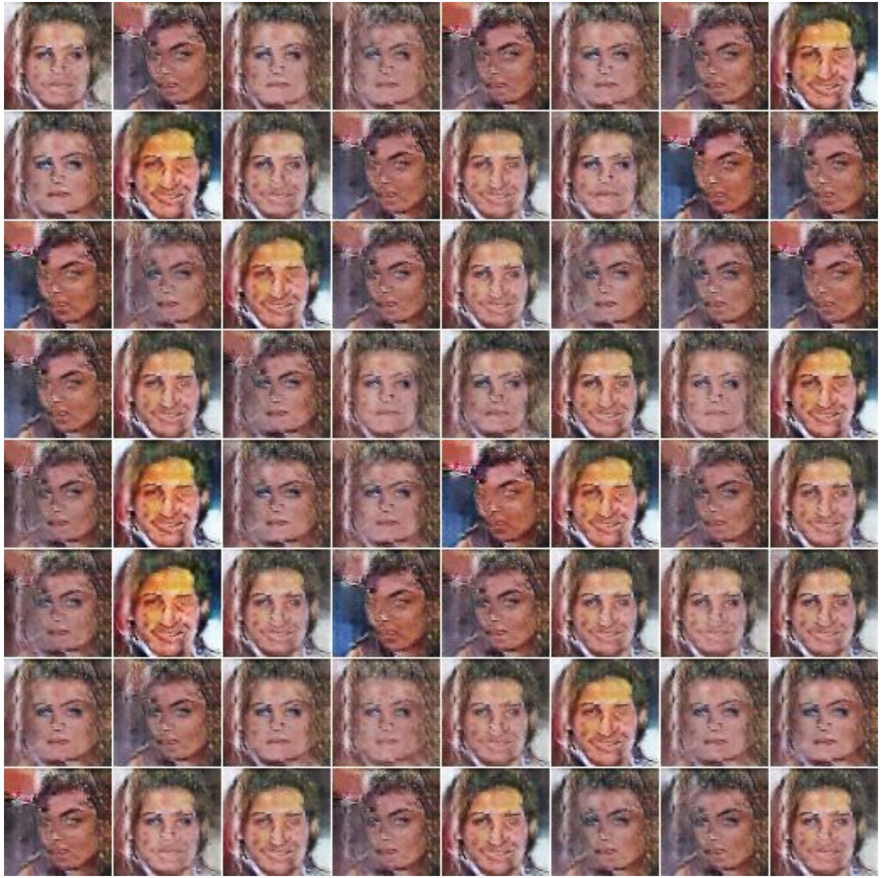
G: catches up and generates only cats

G: generates only dogs



D: catches up and takes simple solution that cats are fake and dogs are real

Mode collapse

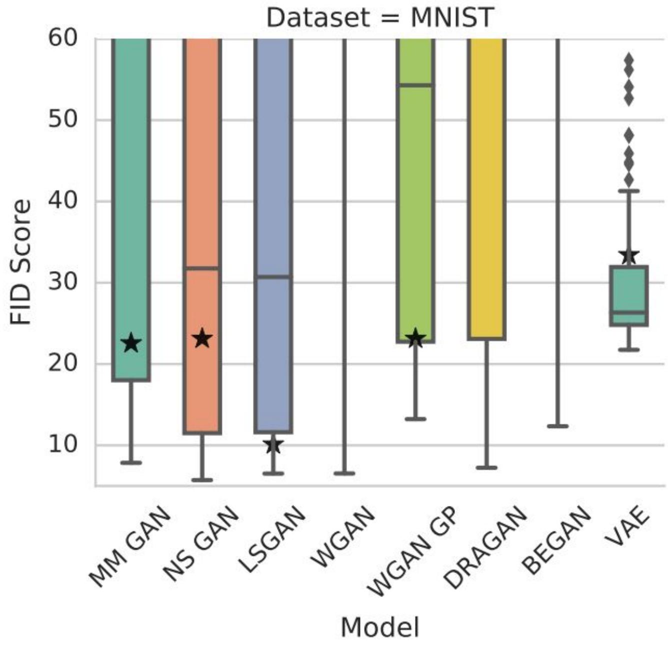


GANs can suffer from mode collapse, where the generator misses modes from the data distribution.

Hyperparameter sensitivity

Want to learn more?

Lucic, et al. **Are GANs Created Equal? A Large-Scale Study. Neurips** (2018)



GANs have been known to suffer from hyperparameter sensitivity.

Figure from Lucic et al, Are GANs Created Equal? A Large-Scale Study.

Mitigation strategies which help with the above issues

Optimisation changes:

- large batch sizes
- low momentum

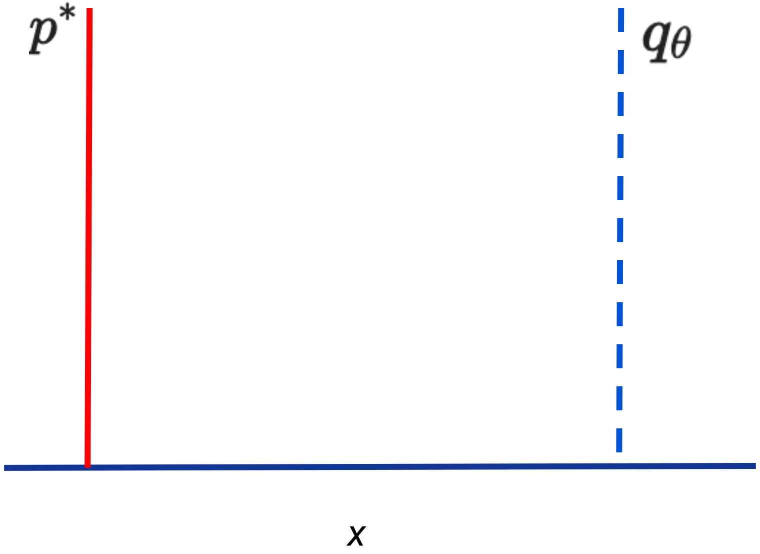
Other changes (optimisation related):

- BatchNorm, Resnets
  - easier to optimise
- spectral normalisation

DiracGAN: a simple example

Want to learn more?

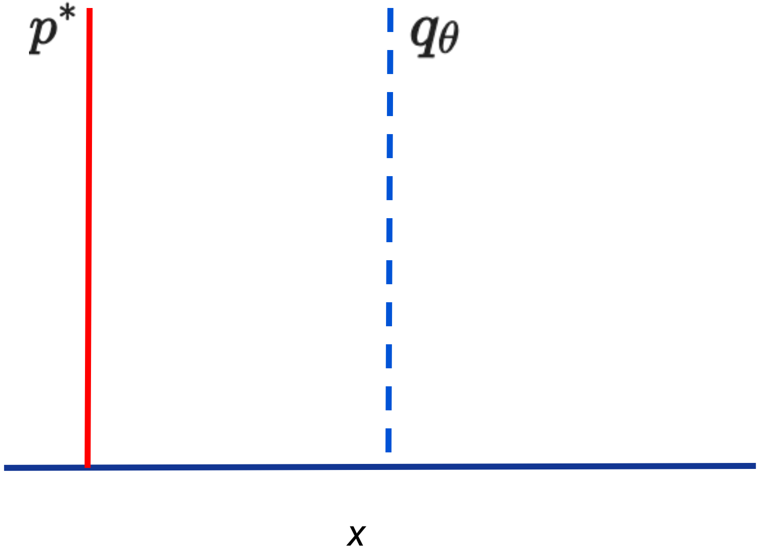
Mescheder, et al. **Which Training Methods for GANs do actually Converge?** ICML (2018)



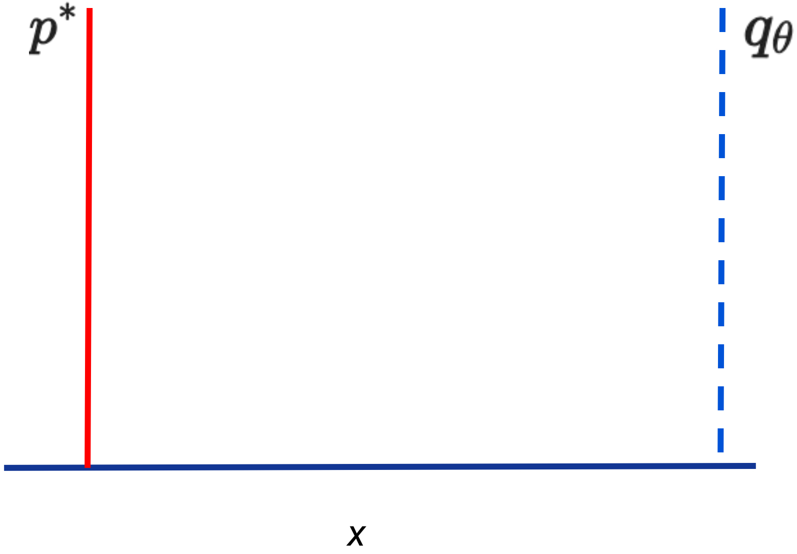
DiracGAN

Want to learn more?

Mescheder, et al. **Which Training Methods for GANs do actually Converge?** ICML (2018)



DiracGAN



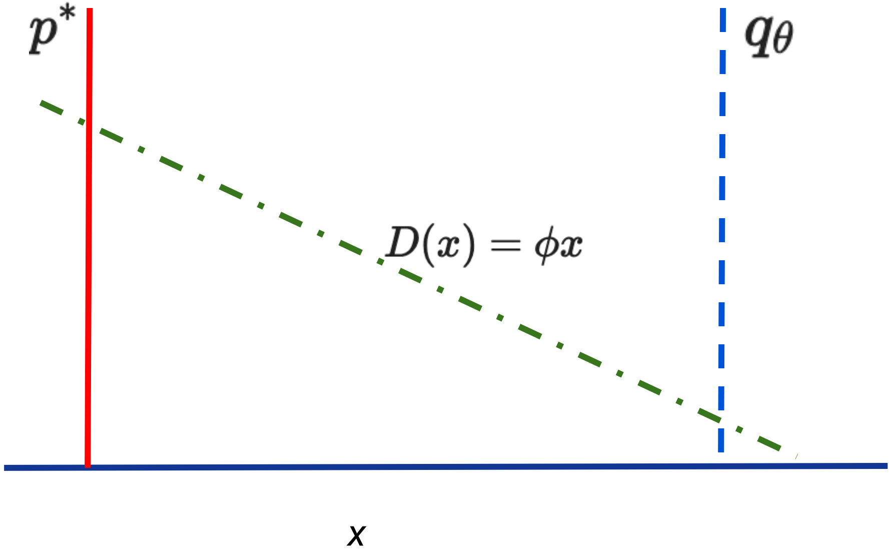
Want to learn more?

Mescheder, et al. **Which Training Methods for GANs do actually Converge?** ICML (2018)

DiracGAN

Want to learn more?

Mescheder, et al. **Which Training Methods for GANs do actually Converge?** ICML (2018)

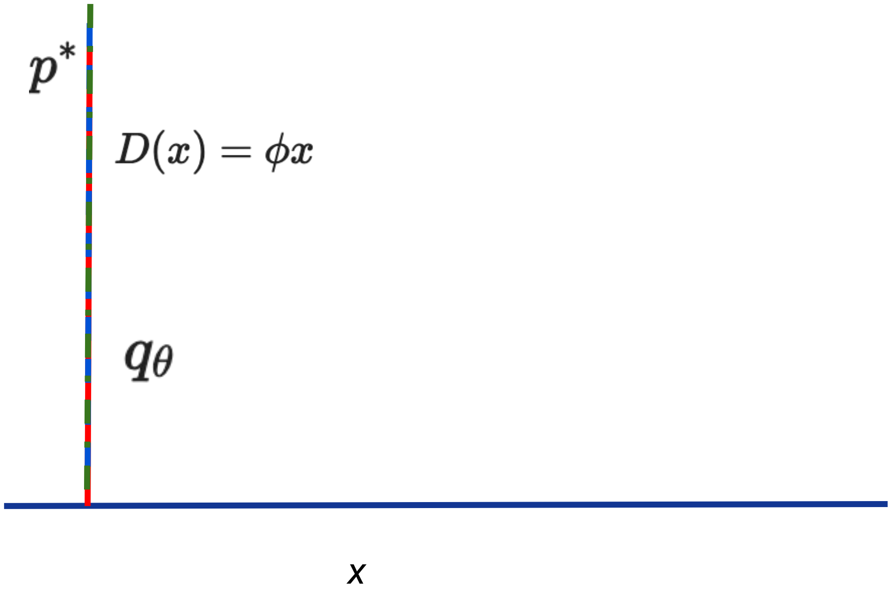


linear discriminator:
areas of space determined more as real

Solution to DiracGAN

Want to learn more?

Mescheder, et al. **Which Training Methods for GANs do actually Converge?** ICML (2018)

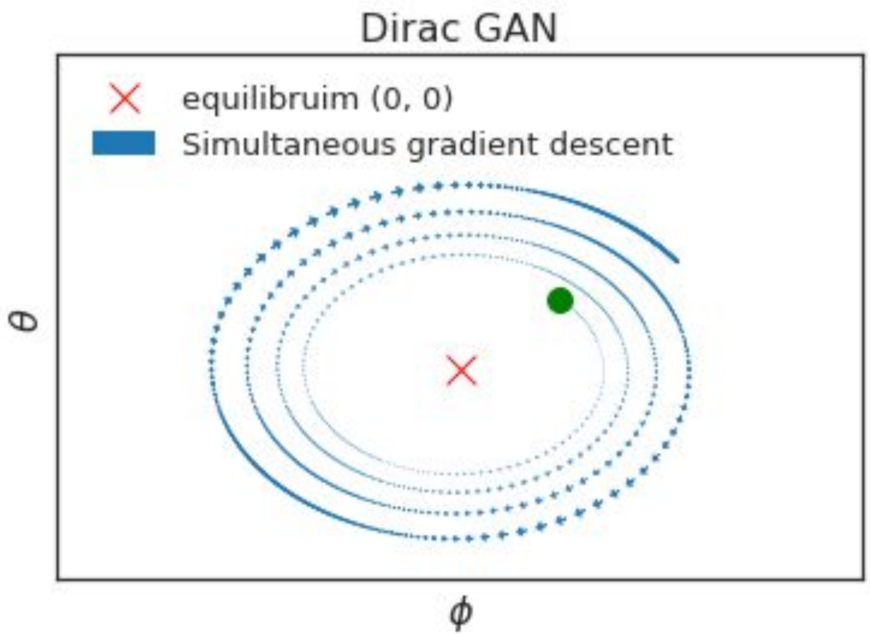


ϕ = 0 θ = 0

Rotational forces in DiracGAN

Want to learn more?

Mescheder, et al. **Which Training Methods for GANs do actually Converge?** ICML (2018)



The authors show that many GANs do not converge on this simple problem!

What does convergence mean for games?

Convergence in games

Nash equilibria: a game has reached a Nash equilibrium if no player can perform better by moving to another part of the space.

$$\phi^\star \in \arg\min_{\mathbb{R}^m} L_f(\cdot, \theta^\star)$$

$$\theta^\star \in \arg\min_{\mathbb{R}^n} L_g(\phi^\star, \cdot)$$

Convergence in GANs: global Nash equilibrium

$$V(\theta, \phi) = \mathbb{E}_{p^*(x)} \log D(x; \phi) + \mathbb{E}_{q(z)} \log\left(1 - D(G(z; \theta); \phi)\right)$$

Looking for the global optimum for the discriminator and generator above leads to:

$$D(x) = \frac{2p^*(x)}{p^*(x) + q_\theta(x)}$$

$$p^*(x) = q_\theta(x)$$

This does not account for optimisation or neural network capacity.

Convergence in games

Local Nash equilibrium:

$$\phi^\star \in \arg\min_{V_f} L_f(\cdot, \theta^\star)$$

$$\theta^\star \in \arg\min_{V_g} L_g(\phi^\star, \cdot)$$

$$\nabla_\phi L_f(\phi, \theta) = 0$$
$$\nabla_\theta L_g(\phi, \theta) = 0$$

$$\begin{bmatrix} \boxed{\nabla_\phi \nabla_\phi L_f(\phi, \theta)} & \nabla_\theta \nabla_\phi L_f(\phi, \theta) \\ \nabla_\phi \nabla_\theta L_g(\phi, \theta) & \boxed{\nabla_\theta \nabla_\theta L_g(\phi, \theta)} \end{bmatrix}$$

positive semi-definite

Do GANs reach a Nash equilibrium?

Want to learn more?

Farina, et al. **Do GANs always have Nash equilibria?** ICML (2020)

Local Nash equilibria might not exist for the GAN game.

The authors find small problems for which given a discriminator and a generator as well as a GAN formulation, one can prove a Nash equilibrium does not exist.

Empirically, they also show that many GANs we train do not reach a Nash equilibrium.

Other ways of measuring convergence

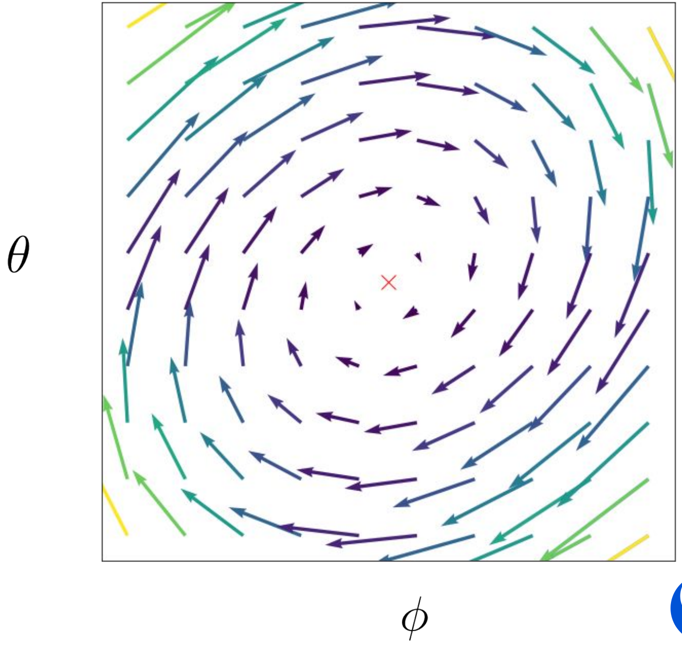
Stationarity: important as GD will stop at stationary points.

$$\nabla_\phi L_f(\phi, \theta) = 0$$
$$\nabla_\theta L_g(\phi, \theta) = 0$$

Locally stable stationary point:

$$\begin{bmatrix} \nabla_\phi \nabla_\phi L_f(\phi, \theta) & \nabla_\theta \nabla_\phi L_f(\phi, \theta) \\ \nabla_\phi \nabla_\theta L_g(\phi, \theta) & \nabla_\theta \nabla_\theta L_g(\phi, \theta) \end{bmatrix}$$

real part of the eigenvalues of the Hessian > 0



Other: such as *Local minmax, Stackelberg equilibrium*.

How to ensure GANs reach convergence?

By analysing conditions for convergence, methods to encourage convergence can be constructed.

Often they take the form of explicit regularisation.

$$L_f(\phi, \theta) \to L_f(\phi, \theta) + \lambda_f R_f(\phi, \theta)$$
$$L_g(\phi, \theta) \to L_g(\phi, \theta) + \lambda_g R_g(\phi, \theta)$$

Examples of ensuring GAN convergence

Common form of regularisers include:

Gradient norm with respect to data

$$R_f(\phi, \theta) = ||\nabla_x D(x)||^2$$

Connection to Lipschitz smoothness.

Connection to convergence.

Gradient norm with respect to parameters

$$R_f(\phi, \theta) = ||\nabla_\phi L_f(\phi, \theta)||^2$$
$$R_f(\phi, \theta) = ||\nabla_\theta L_g(\phi, \theta)||^2$$
$$R_f(\phi, \theta) = ||\nabla_\phi L_f(\phi, \theta)||^2 + ||\nabla_\theta L_f(\phi, \theta)||^2$$

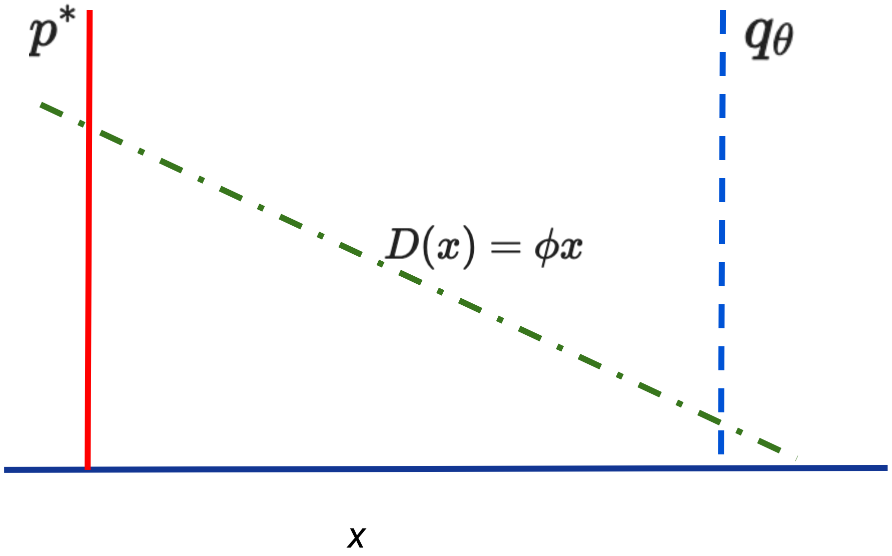
Stabilising effects.

Connection to convergence.

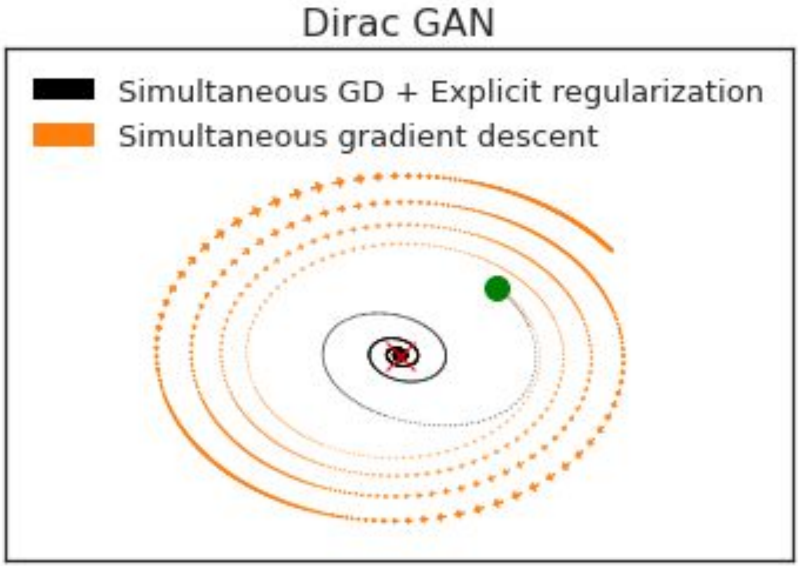
Rotational forces in DiracGAN: explicit regularisation

Want to learn more?

Mescheder, et al. **Which Training Methods for GANs do actually Converge?** ICML (2018)



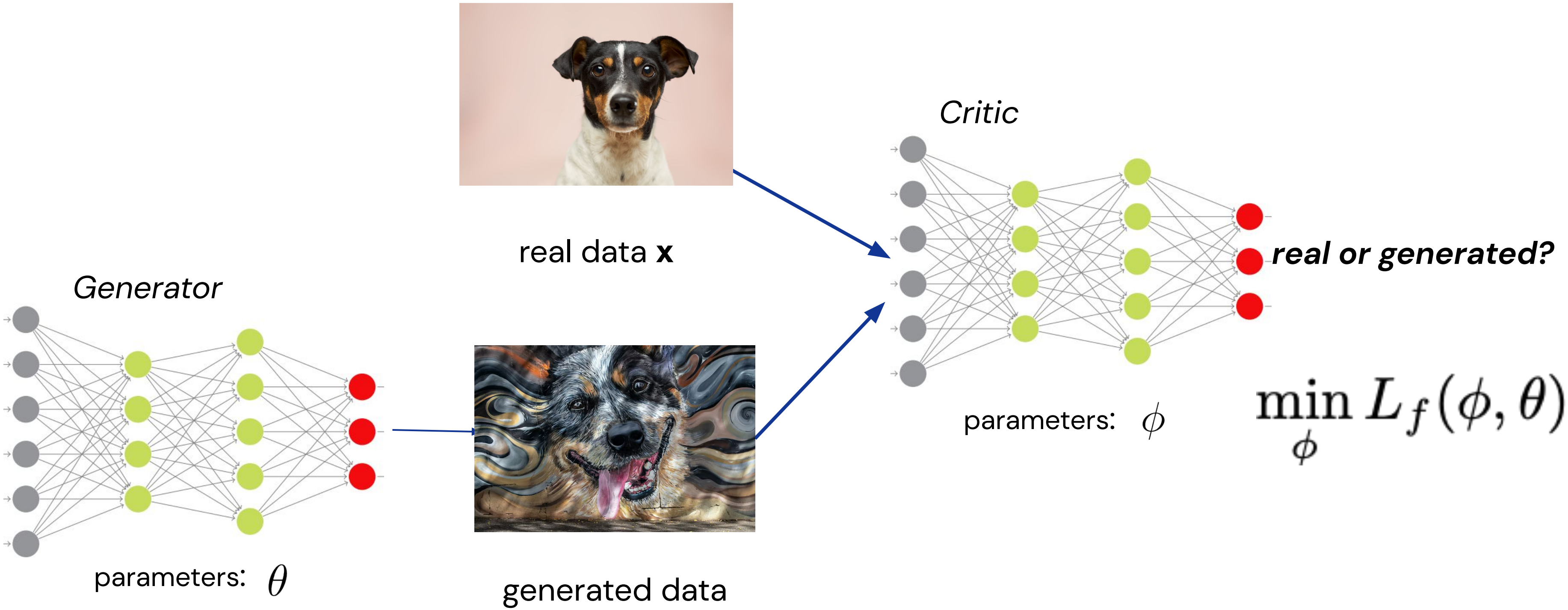
linear discriminator:
areas of space determined more as real



Sometimes we have to change the game in order to ensure convergence.

Are all instabilities inherent in the game or due to gradient descent?

Generative adversarial networks



$$\min_{\phi} L_f(\phi, \theta)$$

$$\min_{\theta} L_g(\phi, \theta)$$

GAN dynamics

$$\dot{\phi} = -\nabla_{\phi} L_f(\phi, \theta) \qquad \text{(Discriminator)}$$

$$\dot{\theta} = -\nabla_{\theta} L_g(\phi, \theta) \qquad \text{(Generator)}$$

GAN dynamics

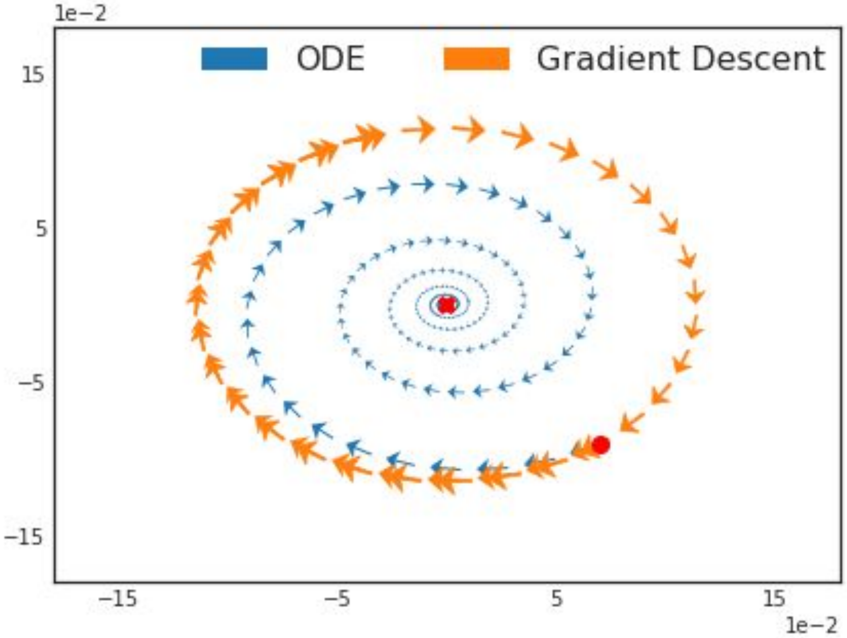
$$\dot{\phi} = -\nabla_{\phi} L_f(\phi, \theta) \qquad \textbf{(Discriminator)}$$

$$\dot{\theta} = -\nabla_{\theta} L_g(\phi, \theta) \qquad \textbf{(Generator)}$$

$$\Big\downarrow \text{discretization}$$

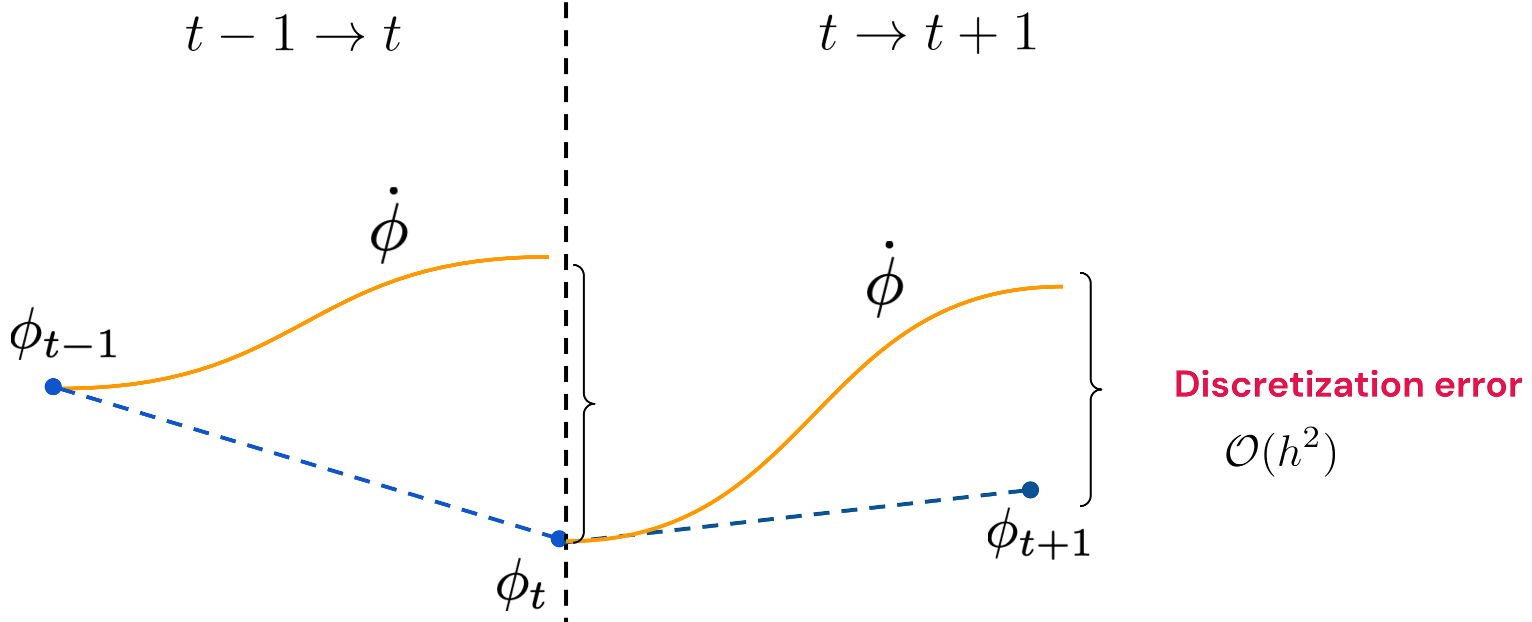
Gradient descent

θ



ϕ

Discretization error for gradient descent

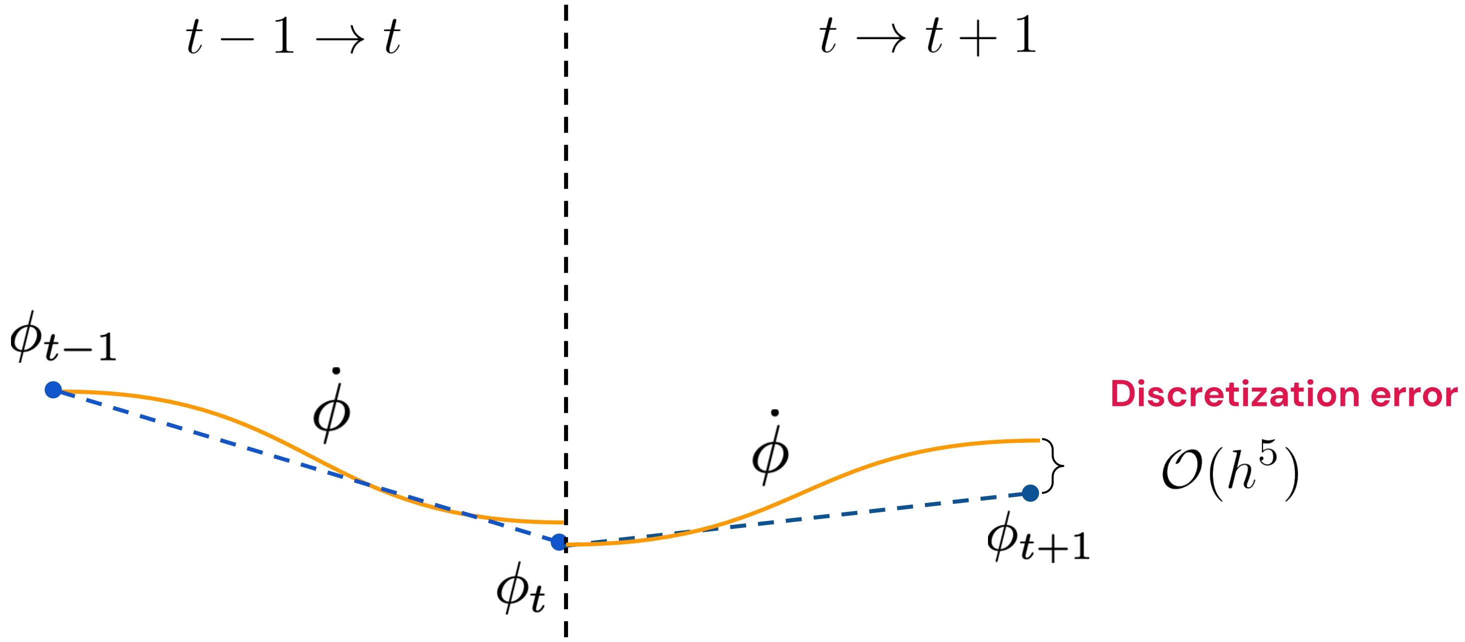


$$\phi_t = \phi_{t-1} - h\nabla_\phi E(\phi_{t-1})$$

Discretization error for Runge Kutta 4 updates

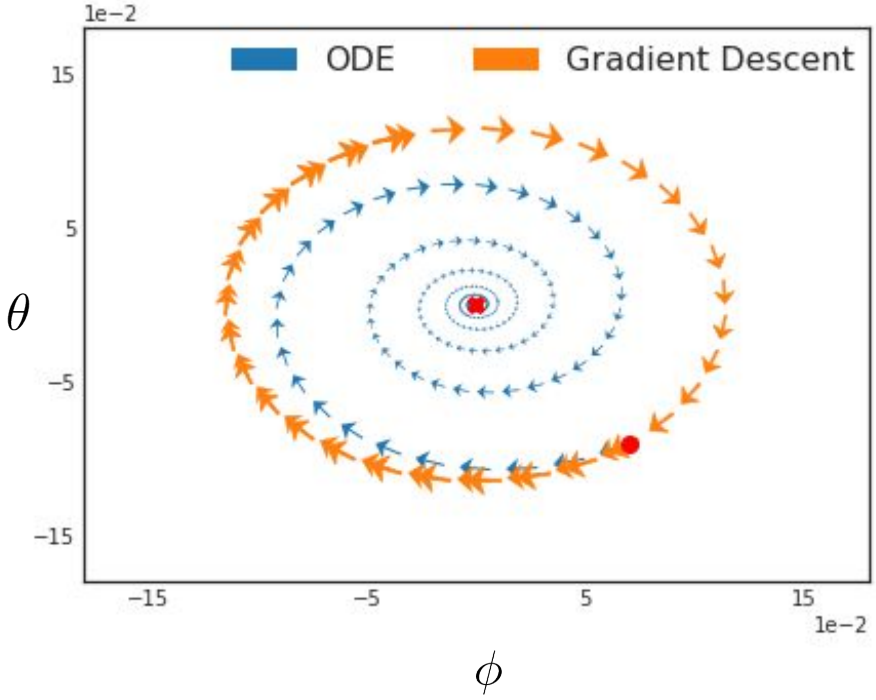
Want to learn more?

Qin, et al. **Training generative adversarial networks by solving ordinary differential equations** Neurips **(2020)**



$$\phi_t = \phi_{t-1} + h\mathrm{RK\_step}(f, \phi_{t-1}, h)$$

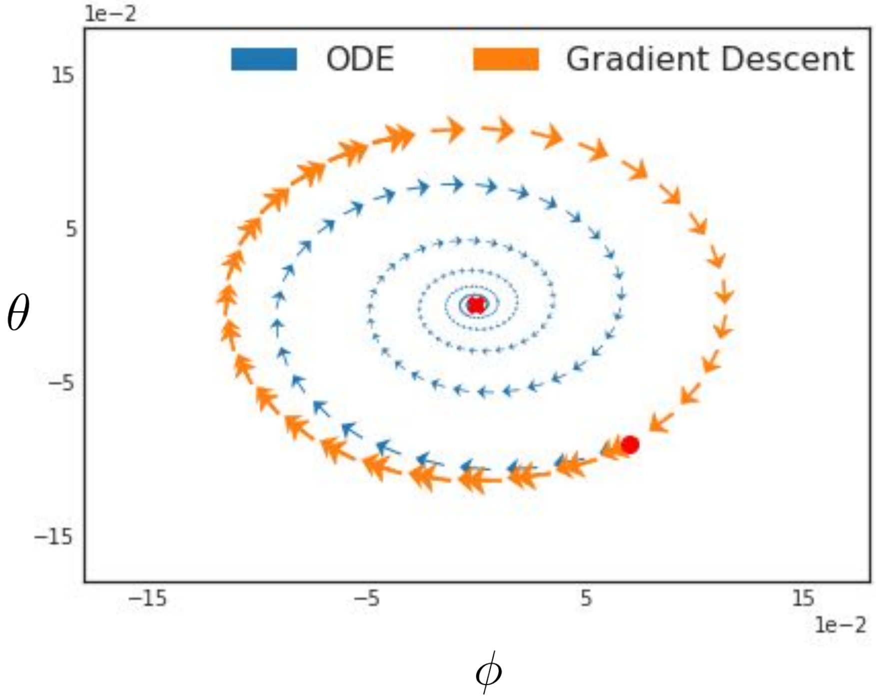
Loss of Stability Due to Discretisation

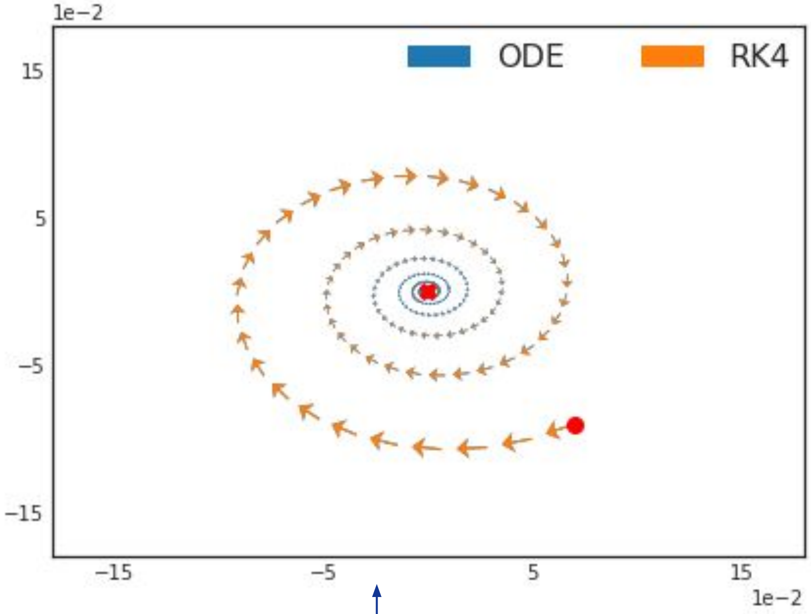


Loss of Stability Due to Discretisation

Want to learn more?

Qin, et al. **Training generative adversarial networks by solving ordinary differential equations** Neurips (2020)

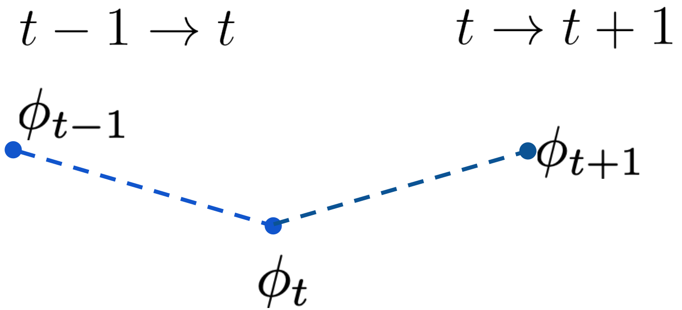




Still might need to be combined with explicit regularisation in practice fo best performance

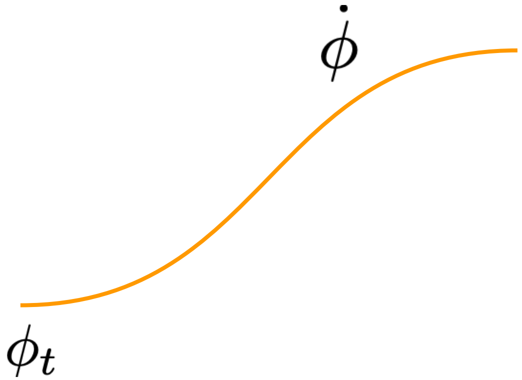
Ways of thinking about GAN optimisation

Discrete view



$$\phi_t = \phi_{t-1} - h\nabla_\phi E(\phi_{t-1})$$

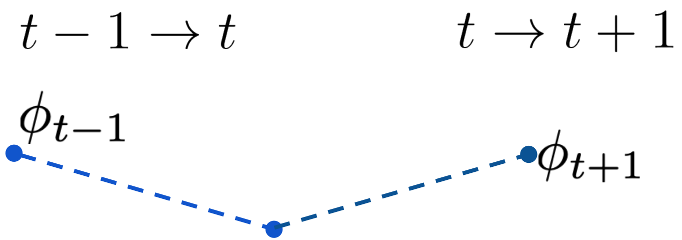
Continuous view



Ways of thinking about GAN optimisation

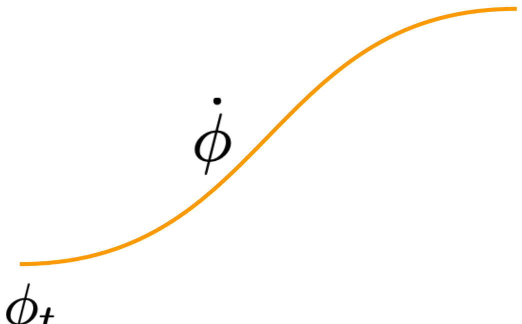
Discrete view

- analyse updates as used in practice (though using updates such as Adam is more complex)
- directly accounts for the learning rate



Continuous view

- analyse the underlying continuous system
- tends to be easier analytically
- the original ODEs do not account for learning rates
    - so there can be a gap between continuous analysis results and what happens in practice



Incorporating the game structure into the optimisation procedure

$$\min_{\theta} \max_{\phi} V$$

Account for the game structure when adapting other optimisation algorithms or creating new ones.

Does a (local) Nash equilibrium exist for this GAN game?

(if it does, it will be locally attractive in continuous time**)

Do our discrete optimisation methods reach this equilibrium?

*** not the case for general games, but true for most GANs*

Optimisation in GANs

Optimisation is an important aspect of GAN training:

- big improvements in GAN results have come from improving optimisation
- defining convergence is not easy
- there is a difference between understanding what the discrete updates do and what the underlying ODE system does

GANs also provide a useful testing ground for the intersection between two-player games and deep learning.

How can we think about GANs?

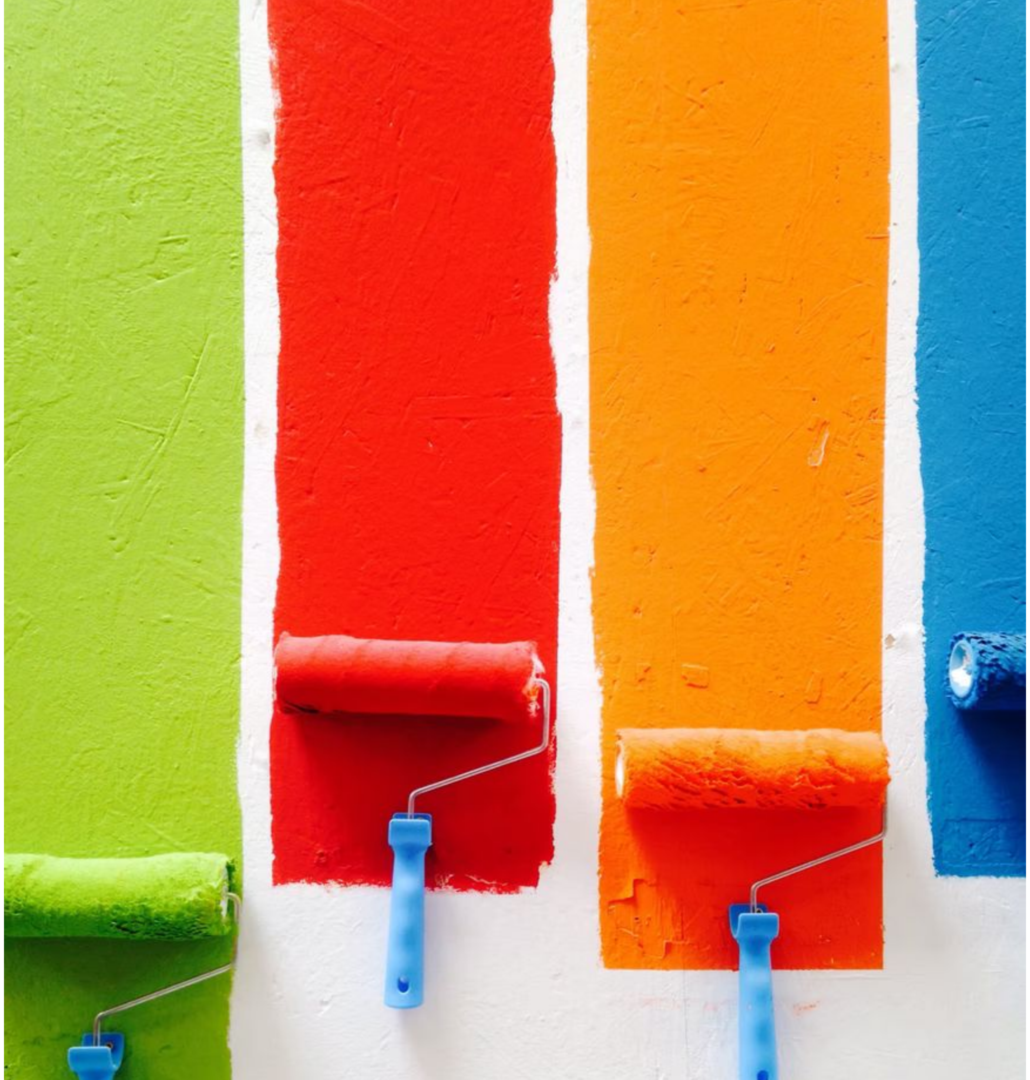
Distributional view

How to construct objectives which ensure the model can learn the data distribution.

Games view

How to construct optimisation methods which ensure convergence.

So far



A more accurate picture

Distributional view

divergence/distance

↓

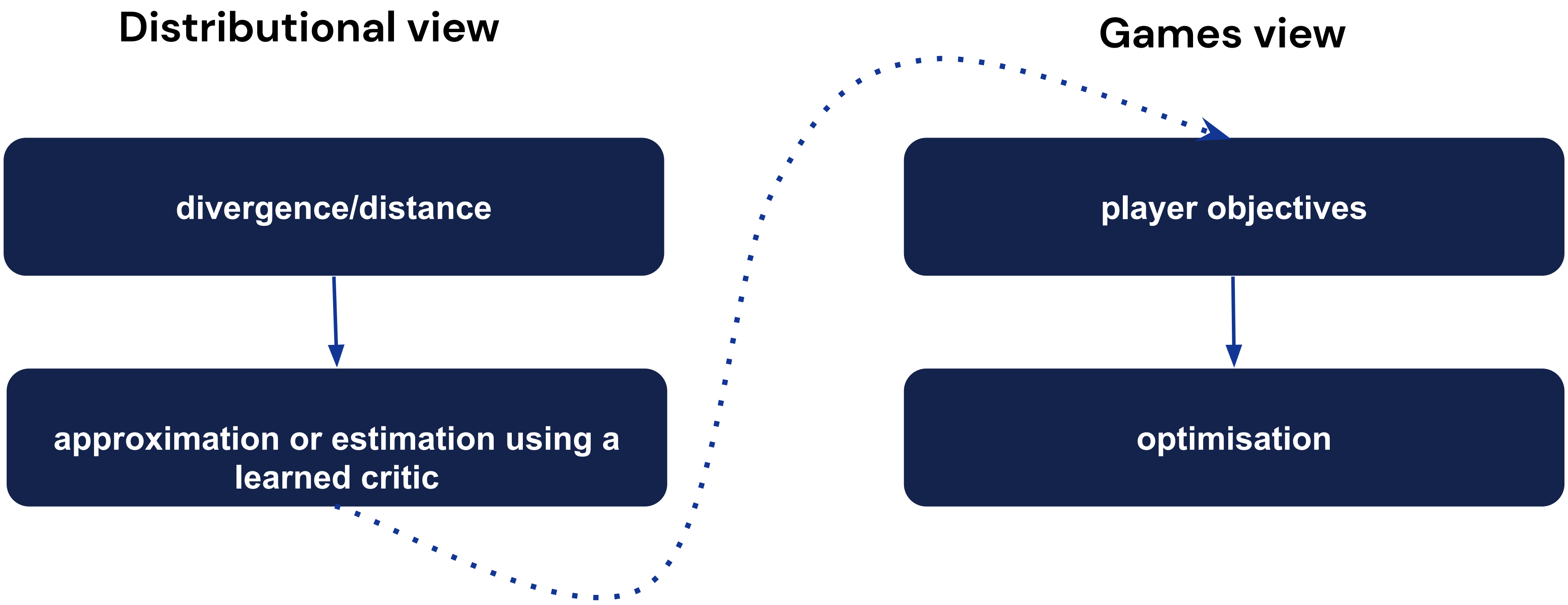
approximation or estimation using a learned discriminator

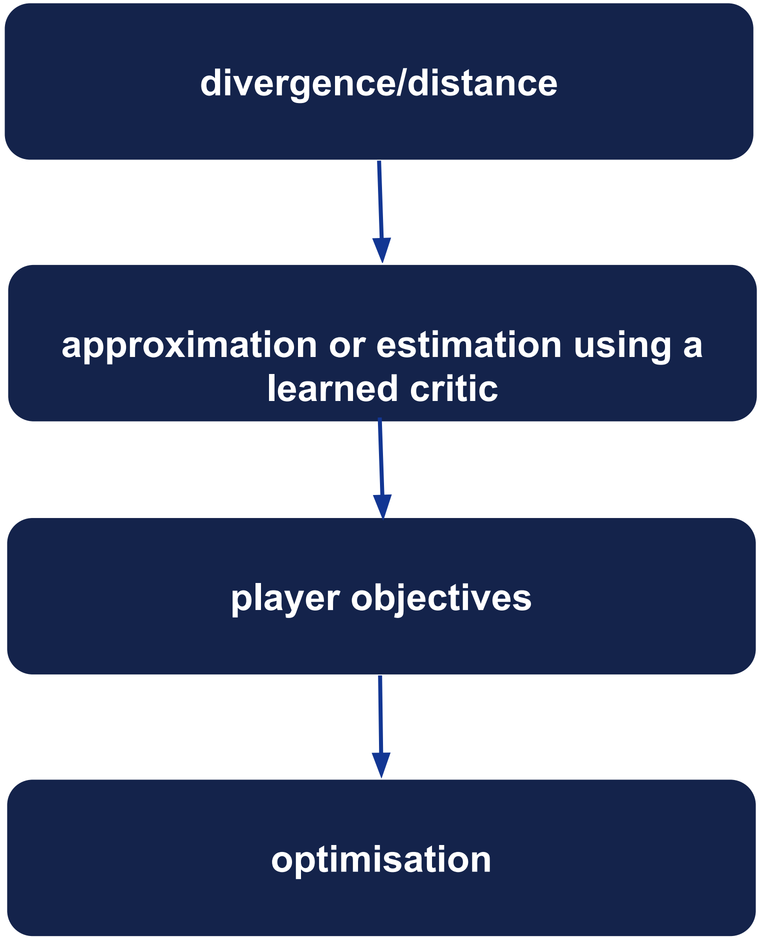
Games view

player objectives

↓

optimisation


Distributional view
Games view
divergence/distance
approximation or estimation using a learned critic
player objectives
optimisation


divergence/distance
approximation or estimation using a learned critic
player objectives
optimisation

Questions to ask

What are the connections between optimisation convergence and the quality of the learned distribution?

Reaching a local Nash equilibrium does not tell us we have learned a good model of the distribution.

$$\nabla_{\phi} L_f(\phi, \theta) = 0$$
$$\nabla_{\theta} L_g(\phi, \theta) = 0$$

$$\begin{bmatrix} \nabla_{\phi}\nabla_{\phi} L_f(\phi, \theta) & \nabla_{\theta}\nabla_{\phi} L_f(\phi, \theta) \\ \nabla_{\phi}\nabla_{\theta} L_g(\phi, \theta) & \nabla_{\theta}\nabla_{\theta} L_g(\phi, \theta) \end{bmatrix}$$

Questions to ask

How can we find the trade-off between optimisation stability and distributional learning performance?

Multiple regularisation methods (explicit gradient regularisation, gradient penalties, dropout) can increase stability but decrease performance of the model.

The cohesive view of supervised learning

Local minima are less of an issue than originally thought.

Connection between optimisation and performance.

Using the implicit regularisation work to make the connection between optimisation and generalisation.

Can the same be done for GANs?

Theoretical analysis might be challenging but perhaps we can start with empirical studies.

Challenges

No clear evaluation metric for how well the model is learning the data distribution.

Many more factors to account for: two players with two architectures and two different optimisation schedules.

What do GANs teach us

- We can estimate different distributional divergences and distances using deep learning and use them to train implicit generative models.

- GANs are a useful testing ground for optimisation ideas for games.

Thank you!

This talk focused on obtaining GAN losses from distributional distances and divergences. There are other ways to change GAN losses, through regularisation or other approaches, including:

- Gradient penalties wrt to inputs
    - *Improved training for Wasserstein GAN*, Gulrajani et al, Neurips, 2017
    - *Which methods of GANs actually converge?* Mescheder et al, ICML 2018
- Gradient regularization wrt to parameters
    - *The numerics of GANs*, Mescheder et al, Neurips, 2017
    - *The Mechanics of n-Player Differentiable Games,* Balduzzi et al, ICML 2018
- Entropy regularization
    - *Prescribed Generative Adversarial Networks*, Dieng et al, 2019
- and many others...

References - Distributional learning

GANs introduced based on divergences and distances:

Generative adversarial nets, Goodfellow et al, Neurips 2014

f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization, Nowozin, et al, Neurips (2016)

Wasserstein GAN, Arjovsky, et al, ICML 2017

MMD GAN: Towards Deeper Understanding of Moment Matching Network, Li, et al, Neurips 2017

Improved Training of Wasserstein GANs, Gulrajani et al, Neurips 2017

Demystifying mmd gans, Bińkowski et al, ICL 2018

Learning in implicit generative models, Mohamed, et al arxiv (2016)

References - Distributional learning

Which divergence and distance to use?

Towards Principled Methods for Training Generative Adversarial Networks, Arjovsky et al, ICLR 2017

Many Paths to Equilibrium: GANs Do Not Need to Decrease a Divergence At Every Step, Fedus et al, ICLR 2018

References - Distributional learning

How to scale other distances and divergences to large scale problems?

Generative Modeling using the Sliced Wasserstein Distance, Deshpande et al, CVPR 2018

Distributional Sliced-Wasserstein and Applications to Generative Modeling, ICLR 2021

Learning Implicit Generative Models with the Method of Learned Moments, Ravuri et al, ICML 2018

Architectures and model regularisation are a core ingredient of GAN training:

- Self attention
  - *Self-Attention Generative Adversarial Networks, Zhang et al, ICML 2019*
- Discriminator regularisation
  - *Spectral Normalization for Generative Adversarial Networks*, Miyato et al, ICLR 2018
- BatchNormalisation is often used for the generator.

Optimisation in games and GANs

Understanding the effect of discretisation

ODE-GAN: Training Generative Adversarial Networks by Solving Ordinary Differential Equations, Qin et al, Neurips 2020

Implicit competitive regularization in gans, Schäfer et al, ICML 2021

Discretization Drift in Two-Player Games, Rosca et al, ICML 2021

The limit points of (optimistic) gradient descent in min-max optimization, Daskalakis et al, Neurips 2018

References - games

What can be done to encourage convergence?

The Numerics of GANs, Mescheder et al, Neurips 2017

Which Training Methods for GANs do actually Converge, Mescheder et al, ICML 2018

ODE-GAN: Training Generative Adversarial Networks by Solving Ordinary Differential Equations, Qin et al, Neurips 2020

Gradient descent GAN optimization is locally stable, Nagarajan et al, Neurips 2017

The Mechanics of n-Player Differentiable Games, Balduzzi et al, ICML 2018

On Solving Minimax Optimization Locally: A Follow-the-Ridge Approach, Wang et al, ICLR 2020

Incorporating the game structure into the optimisation procedure

$$\min_{\theta} \max_{\phi} V$$

Account for the game structure when adapting other optimisation algorithms or creating new ones.

References - games

How can the game structure be incorporated into the optimisation procedure?

Unrolled Generative Adversarial Networks, Metz et al, ICLR 2017

Taming GANs with Lookahead-Minmax, Chavdarova et al, ICLR 2021

Reducing Noise in GAN Training with Variance Reduced Extragradient, Chavdarova et al, Neurips 2019

Competitive Gradient Descent, Schäfer et al, Neurips 2019

On Solving Minimax Optimization Locally: A Follow-the-Ridge Approach, Wang et al, ICLR 2020

Mitigation strategies which help with the above issues

Optimisation changes:

- large batch sizes
- low momentum

Other changes (optimisation related):

- BatchNorm
- Resnets
  - easier to optimise
- spectral normalisation
  - this has been connected to optimisation (both in GANs and more widely, in RL)

Mitigation strategies which help with the above issues

- *Large Scale GAN Training for High Fidelity Natural Image Synthesis, Brock et al, ICLR 2019*
- *Spectral Normalization for Generative Adversarial Networks, Miyato et al, ICLR 2018*
- *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, Radford et al, ICLR 2016*
- *Improved Training of Wasserstein GANs, Gulrajani et al, Neurips 2018*
- *Self-Attention Generative Adversarial Networks, Zhang et al, ICML 2019*

Examples of ensuring GAN convergence

Common form of regularisers include:

Gradient norm with respect to data

$$R_f(\phi, \theta) = ||\nabla_x D(x)||^2$$

Connection to Lipschitz smoothness.

Connection to convergence.

Gradient norm with respect to parameters

$$R_f(\phi, \theta) = ||\nabla_\phi L_f(\phi, \theta)||^2$$
$$R_f(\phi, \theta) = ||\nabla_\theta L_g(\phi, \theta)||^2$$
$$R_f(\phi, \theta) = ||\nabla_\phi L_f(\phi, \theta)||^2 + ||\nabla_\theta L_f(\phi, \theta)||^2$$

Stabilising effects.

Connection to convergence.

Examples of ensuring GAN convergence

- Gradient penalties with respect to input
    - *Which Training Methods for GANs do actually Converge, Mescheder et al, ICML 2018*
    - *On gradient regularizers for MMD GANs, Arbel et al, Neurips 2018*
- Gradient regularison with respect to parameters
    - *The Numerics of GANs, Mescheder et al, Neurips 2017*
    - *Gradient descent GAN optimization is locally stable, Nagarajan et al, Neurips 2017*
    - *The Mechanics of n-Player Differentiable Games, Balduzzi et al, ICML 2018*

Evaluating GANs:

- Inception Score
  - *Improved Techniques for Training GANs*, Salimans et al, Neurips 2016
- Frechet Inception Distance
  - *GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium,* Heusel et al, Neurips 2017
- Kernel Inception Distance
  - *Demystifying MMD GANs*, Binkowski et al, ICLR 2018
- Precision and recall metrics
  - Improved Precision and Recall Metric for Assessing Generative Models, s Kynkäänniemi et al, Neurips 2019
- Training classifiers with data generated from GANs
  - *Classification Accuracy Score for Conditional Generative Models*, Ravuri et al, Neurips 2019

And much more...

You can find more related work at conectedpapers.com

Thank you