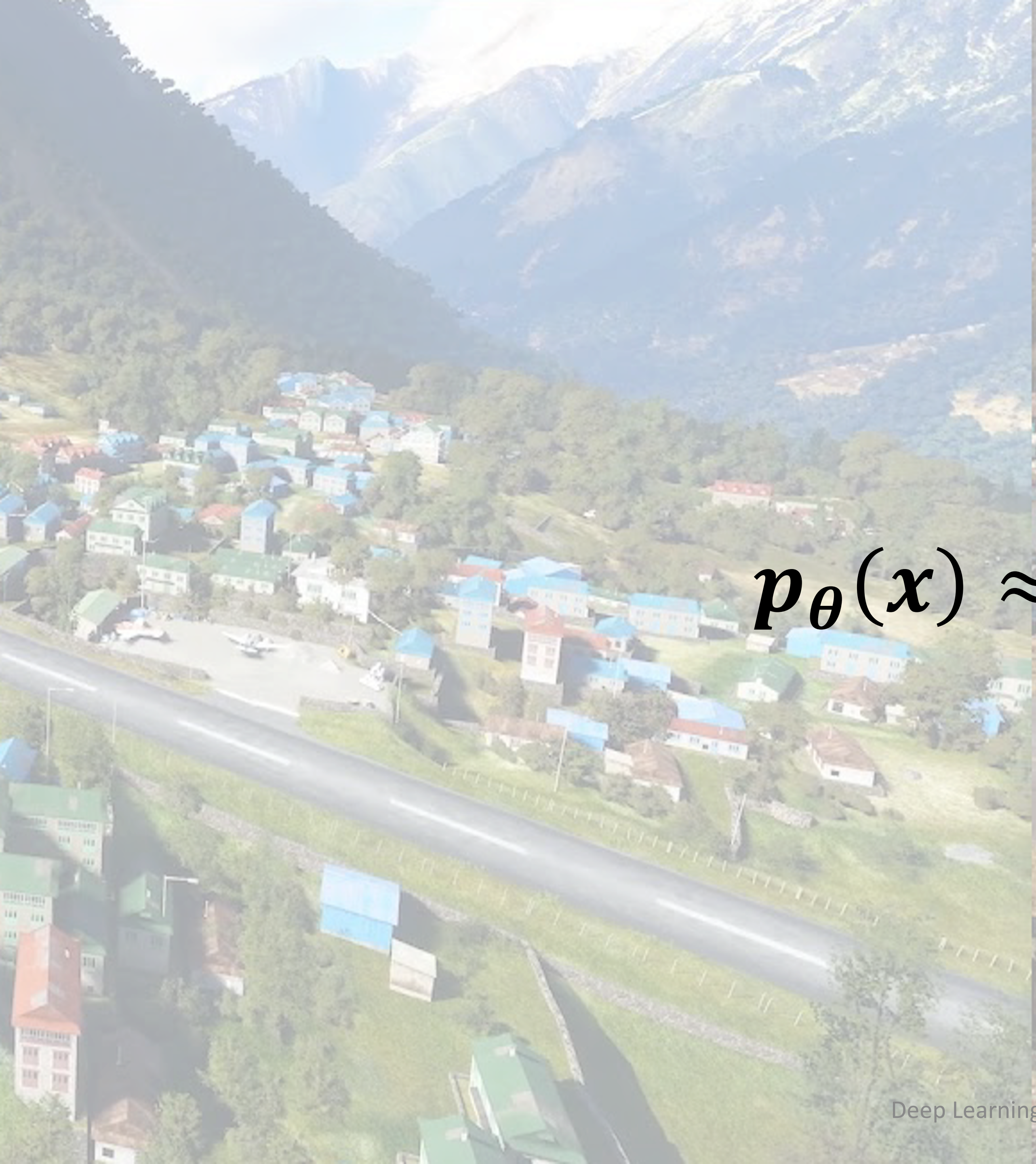
Generative Models

GAN basics

Yingzhen Li (yingzhen.li@imperial.ac.uk)



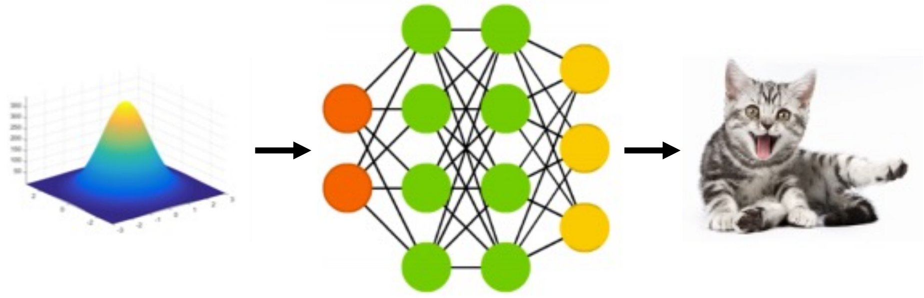
$$p_\theta(x) \approx p_{data}(x)$$

Divergence minimisation

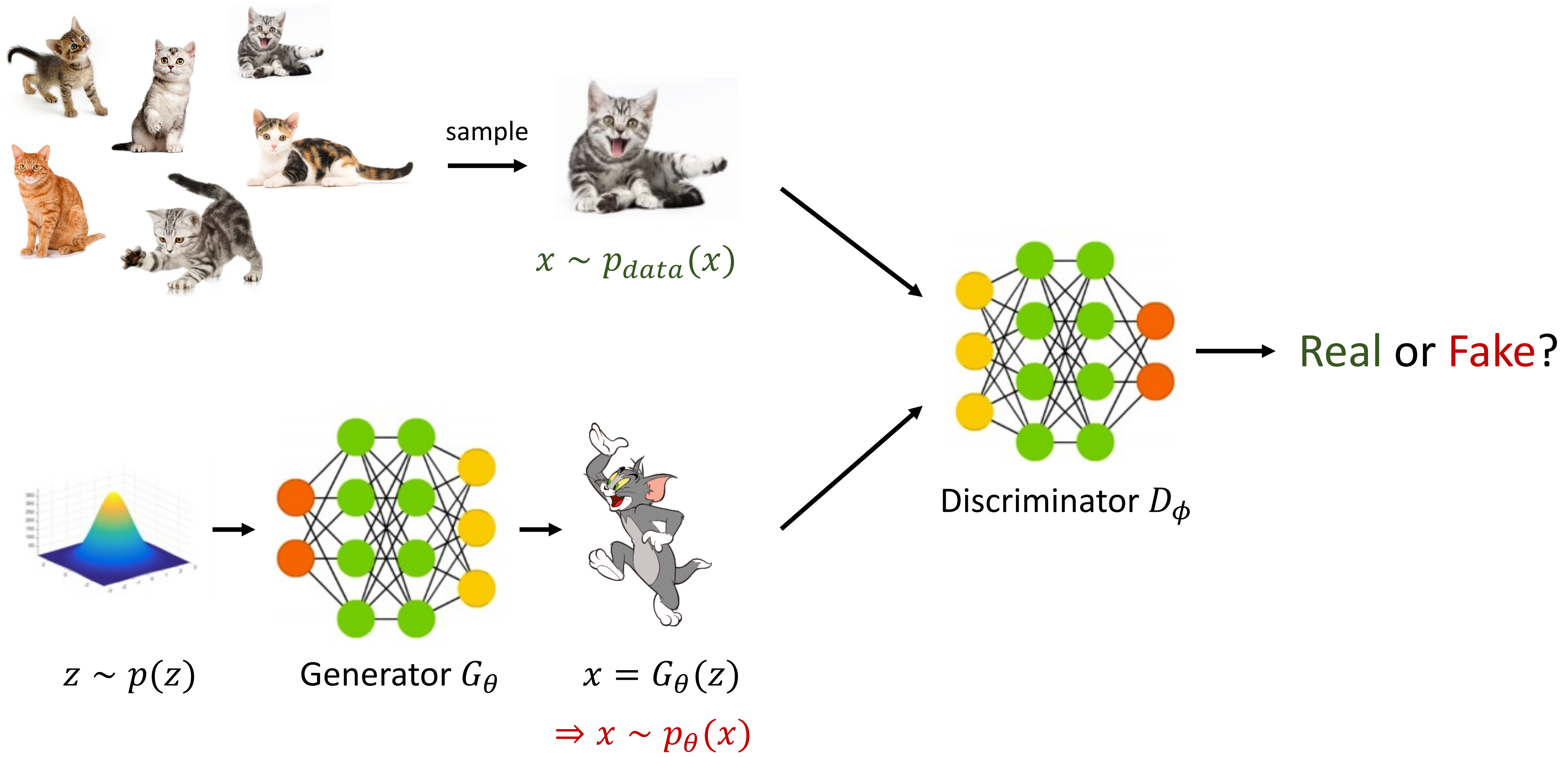
- Fitting the model to the data by divergence minimisation:

$$\theta^* = argmin\ D[p_{data}(x) \,||\, p_\theta(x)]$$

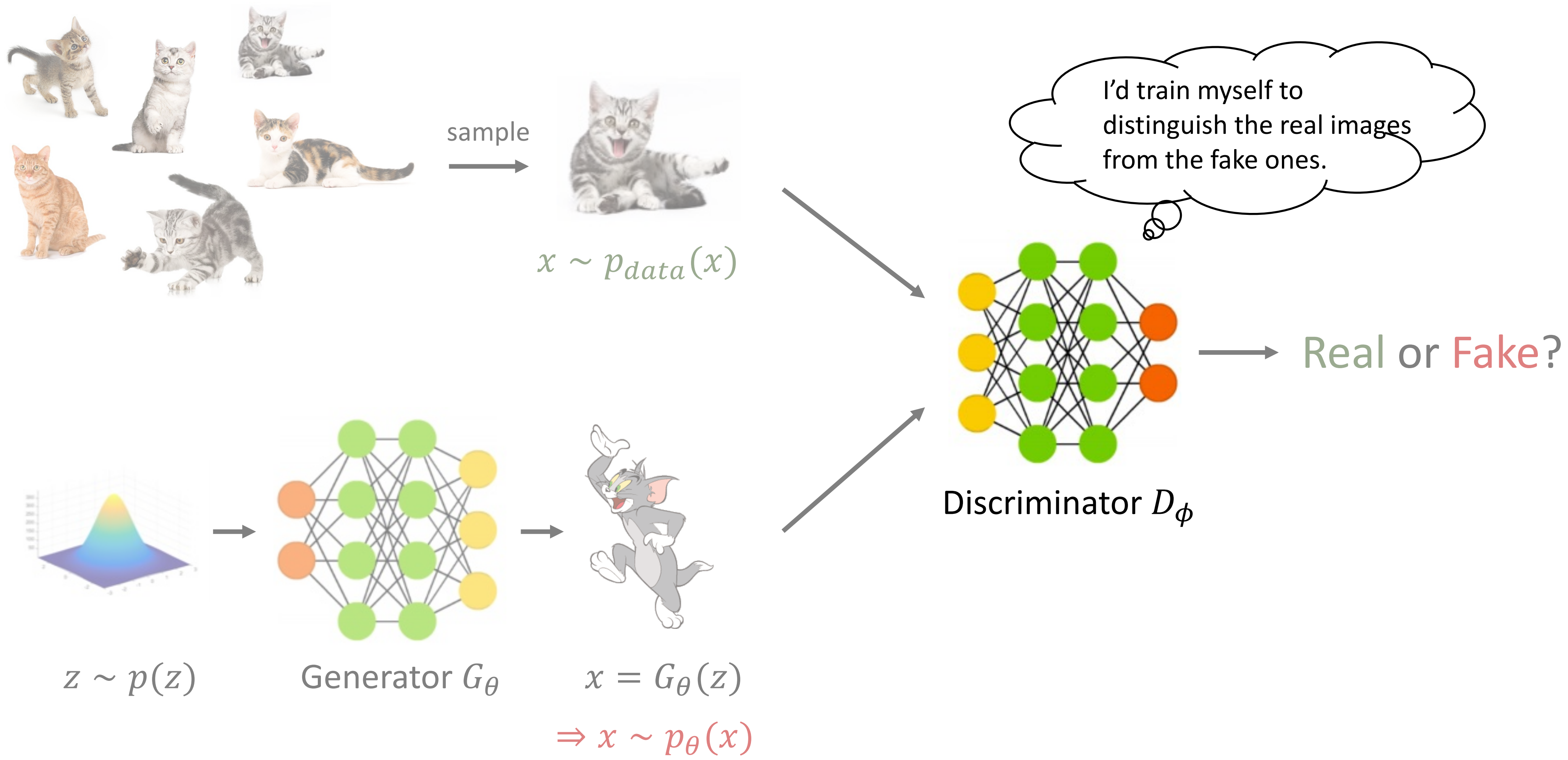
- VAE: variational maximum likelihood training
  - Objective: MLE is equivalent to minimizing $KL[p_{data}(x) \,||\, p_\theta(x)]$
  - For LVMs, $\log p_\theta(x) = \log \int p_\theta(x|z)p(z)dz$ is intractable
    - $\Rightarrow$ variational lower-bound $L(x, \phi, \theta) \leq \log p_\theta(x)$
    - maximise $E_{p_{data}(x)}[L(x, \phi, \theta)]$ instead



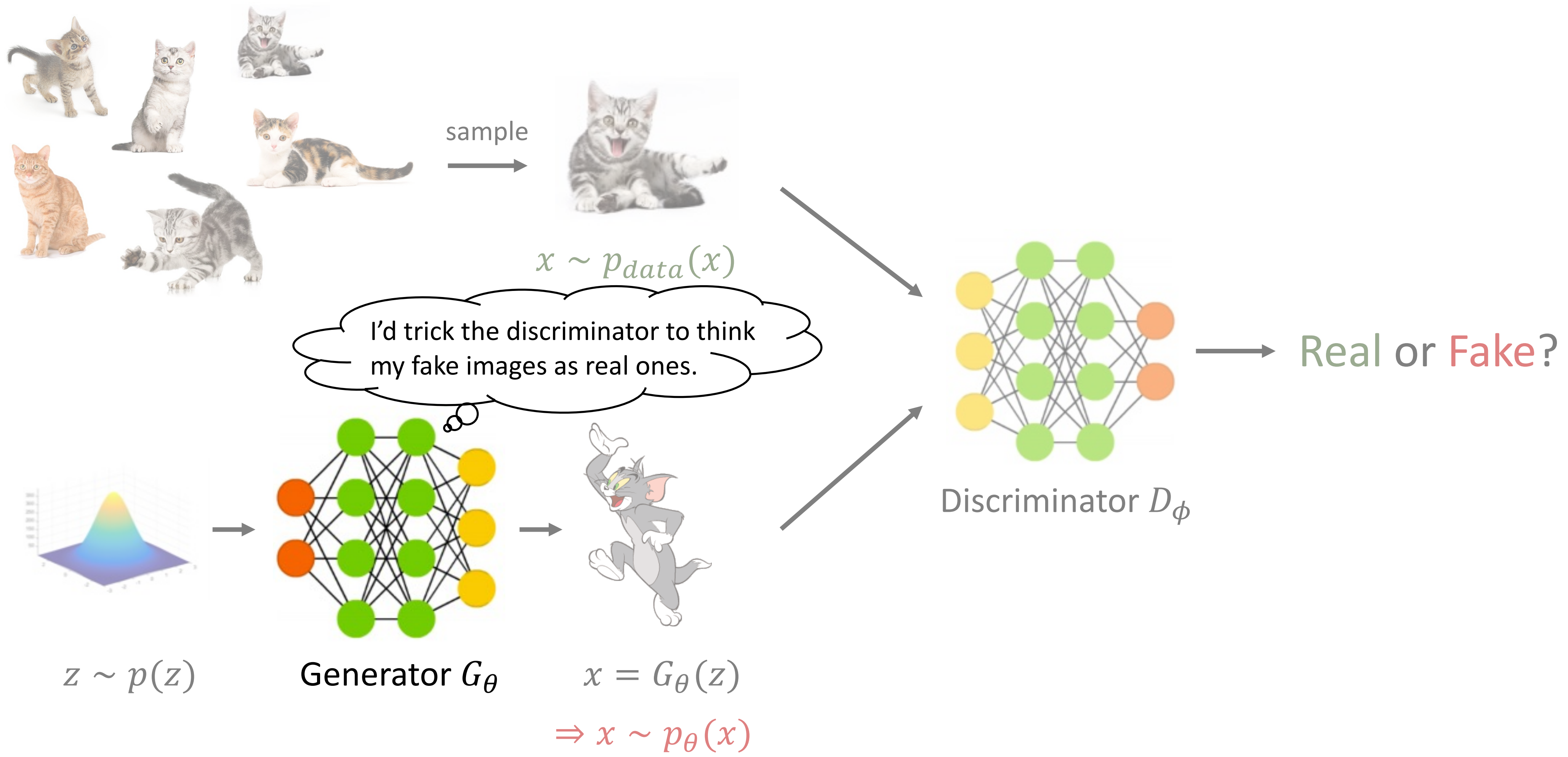
Generative adversarial networks (GANs)



Generative adversarial networks (GANs)



Generative adversarial networks (GANs)



Generative adversarial networks (GANs)

- Two-player game objective:

$$\min_{\theta} \max_{\phi} L(\theta, \phi) \coloneqq E_{p_{data}(x)}\left[\log D_{\phi}(x)\right] + E_{p_{\theta}(x)}[\log(1 - D_{\phi}(x))]$$

$$D_{\phi}(x) \coloneqq P(x \text{ } is \text{ } real), \quad 1 - D_{\phi}(x) = P(x \text{ } is \text{ } fake)$$

  - With fixed θ: training D_{ϕ} as the classifier of the following binary classification task with maximum likelihood (i.e. negative cross-entropy):

$$y = 1 \text{ if } x \sim p_{data}(x), \quad \text{else} \quad y = 0 \text{ if } x \sim p_{\theta}(x)$$

  - With fixed ϕ: training G_{θ} to minimize the log-probability of $x \sim p_{\theta}(x)$ being classified as "fake data" by D_{ϕ}

Goodfellow et al. Generative Adversarial Nets. NeurIPS 2014

Generative adversarial networks (GANs)

- Solving the two-player game objective:

$$\min_{\theta} \max_{\phi} L(\theta, \phi) \; := E_{p_{data}(x)}[\log D_{\phi}(x)] + E_{p_{\theta}(x)}[\log(1 - D_{\phi}(x))]$$

  - Assume the discriminator network D_{ϕ} has infinite capacity: with fixed θ

$$\phi^* \; := \max_{\phi} L(\theta, \phi) \;\text{ satisfies }\; D_{\phi^*}(x) = \frac{p_{data}(x)}{p_{data}(x) + p_{\theta}(x)}$$

  - Plug-in the optimal discriminator (θ dependant) to the objective:

$$L(\theta, \phi^*(\theta)) = E_{p_{data}(x)}\left[\log \frac{p_{data}(x)}{p_{data}(x) + p_{\theta}(x)}\right] + E_{p_{\theta}(x)}\left[\log \frac{p_{\theta}(x)}{p_{data}(x) + p_{\theta}(x)}\right]$$

$$= KL[p_{data}(x) \,||\, \tilde{p}(x)] + KL[p_{\theta}(x) \,||\, \tilde{p}(x)] - 2\log 2$$

$$\tilde{p}(x) := \frac{1}{2} p_{data}(x) + \frac{1}{2} p_{\theta}(x)$$

$$= 2\, JS[p_{data}(x) \,||\, p_{\theta}(x)] \; - 2\log 2$$

Jensen-Shannon divergence between $p_{data}(x)$ and $p_{\theta}(x)$

$JS[p_{data} \,||\, p_{\theta}] = 0 \;\Leftrightarrow\; p_{\theta}(x) = p_{data}(x)$

Goodfellow et al. Generative Adversarial Nets. NeurIPS 2014

Generative adversarial networks (GANs)

- Optimising GANs in practice: a double-loop algorithm
  - Inner loop: with fixed θ, optimise ϕ for a few gradient ascent iterations:

$$\max_{\phi} E_{p_{data}(x)}[\log D_{\phi}(x)] + E_{p_{\theta}(x)}[\log(1 - D_{\phi}(x))]$$

  - Outer loop: with fixed ϕ from the inner loop, optimize θ by ONE gradient descent step:

$$\min_{\theta} E_{p_{\theta}(x)}[\log(1 - D_{\phi}(x))]$$

  - In practice the expectations $E_{p_{data}(x)}[\cdot]$ and $E_{p_{\theta}(x)}[\cdot]$ are estimated by mini-batches:

$$E_{p_{data}(x)}[\log D_{\phi}(x)] \approx \frac{1}{M}\sum_{m=1}^{M} \log D_{\phi}(x_m), x_m \sim p_{data}(x)$$

$$E_{p_{\theta}(x)}\left[\log\left(1 - D_{\phi}(x)\right)\right] \approx \frac{1}{K}\sum_{k=1}^{K} \log\left(1 - D_{\phi}\left(G_{\theta}(z_k)\right)\right), z_k \sim p(z)$$

Loop over until convergence

Goodfellow et al. Generative Adversarial Nets. NeurIPS 2014

Generative adversarial networks (GANs)

Practical implementation for solving $\min_{\theta} \max_{\phi} E_{p_{data}(x)}[\log D_\phi(x)] + E_{p_\theta(x)}[\log(1 - D_\phi(x))]$ (pseudo code):

- Initialise θ, ϕ, learning rates γ_D, γ_G, SGD outer-/inner-loop iterations T, K
- For $t = 1, \ldots, T$

  # update discriminator

  - For $i = 1, \ldots, K$
    - $z_1, \ldots, z_M \sim p(z)$
    - $x_1, \ldots, x_M \sim p_{data}(x)$
    - $\phi \leftarrow \phi + \gamma_D \nabla_\phi [\frac{1}{M} \sum_{m=1}^{M} \log D_\phi(x_m) + \frac{1}{M} \sum_{m=1}^{M} \log(1 - D_\phi(G_\theta(z_m)))]$

  # update generator

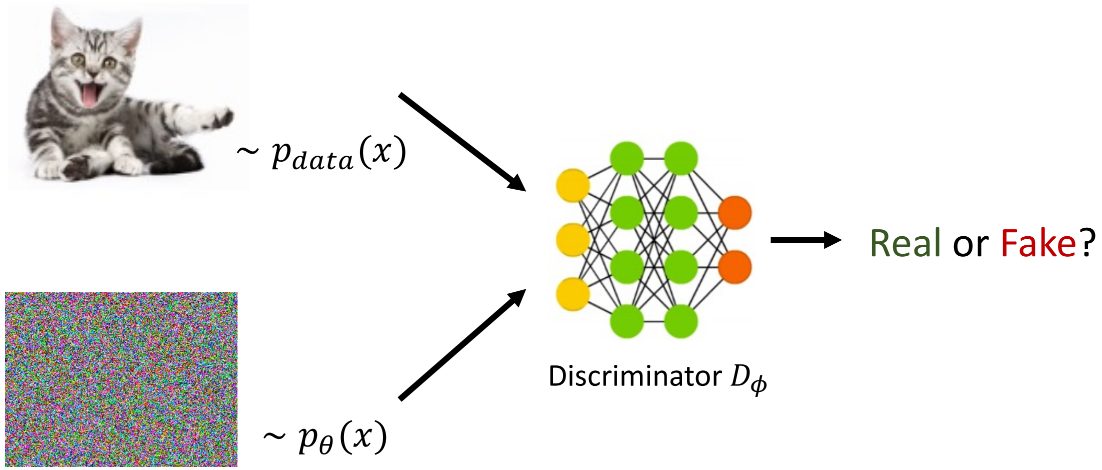
  - $z_1, \ldots, z_J \sim p(z)$
  - $\tilde{x}_j = G_\theta(z_j), j = 1, \ldots, J$
  - $\theta \leftarrow \theta - \gamma_G \nabla_\theta \frac{1}{J} \sum_j^J \log\left(1 - D_\phi(\tilde{x}_j)\right)$

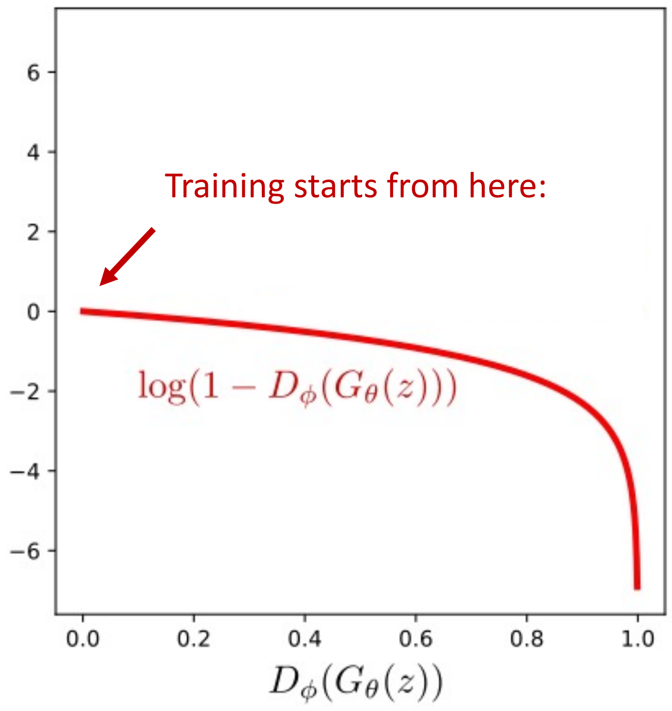
Learning rates γ_D, γ_G & inner-loop iterations K need to be chosen carefully!
(otherwise training may be unstable)

Goodfellow et al. Generative Adversarial Nets. NeurIPS 2014

Generative adversarial networks (GANs)

- Practical strategy for training the generator G_θ:
  - At the beginning, generated image quality is bad



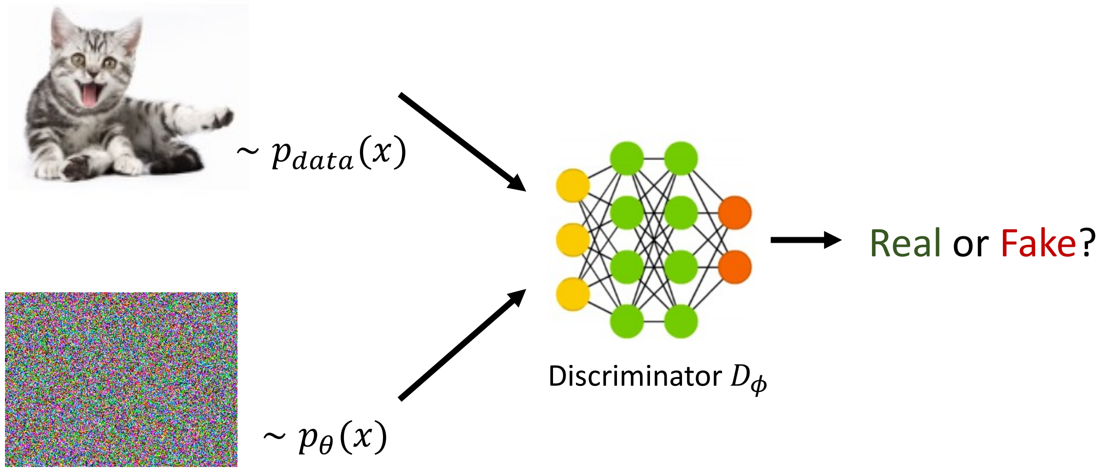


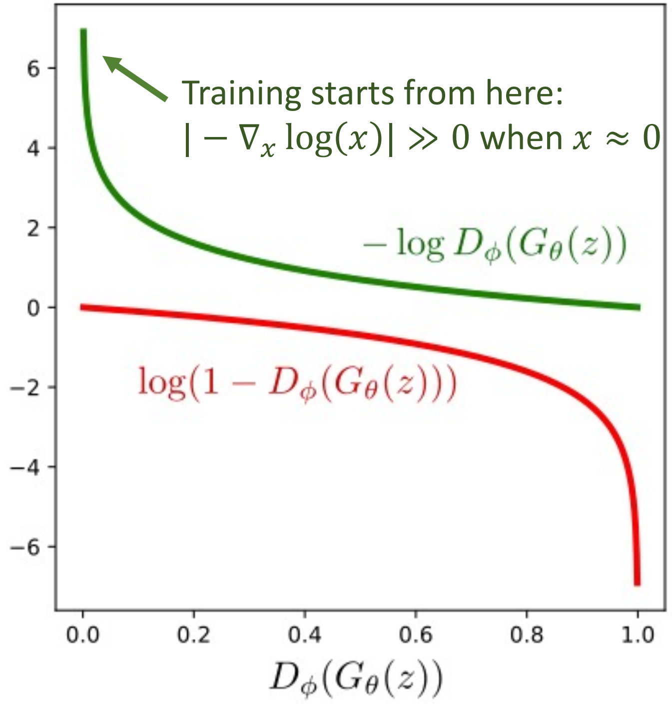
$\Rightarrow$ Discriminator can classify fake images correctly with high confidence: $D_\phi(G_\theta(z)) \approx 0$

Goodfellow et al. Generative Adversarial Nets. NeurIPS 2014

Generative adversarial networks (GANs)

- Practical strategy for training the generator G_θ:
  - At the beginning, generated image quality is bad





$\Rightarrow$ Use an alternative "non-saturate" loss:

$$\min_\theta -E_{p_\theta(x)}[\log D_\phi(x)]$$

"maximizing the probability of making wrong decisions on fake data"

Goodfellow et al. Generative Adversarial Nets. NeurIPS 2014

Wasserstein GAN

- Discriminator can be used to score the provided inputs

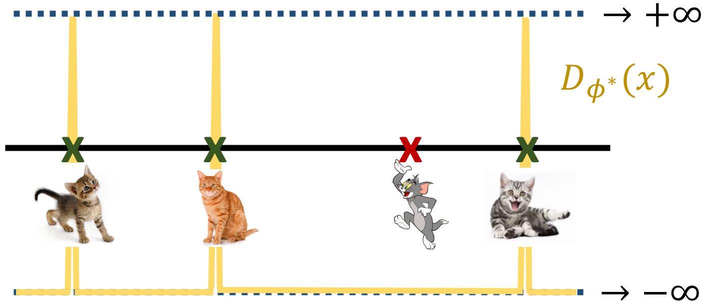
$$\min_{\theta} \max_{\phi} \underline{E_{p_{data}(x)}[D_\phi(x)]} - \underline{E_{p_\theta(x)}[D_\phi(x)]}$$

Discriminator should assign high scores to data inputs and low scores to fake inputs

- Assume the discriminator network D_ϕ has infinite capacity: a trivial solution

$$D_{\phi^*}(x) = +\infty \text{ if } x \sim p_{data}(x) \text{ else } D_{\phi^*}(x) = -\infty$$

No useful gradient info for generator learning!



Arjovsky et al. Wasserstein Generative Adversarial Networks. ICML 2017
Gulrajani et al. Improvedtraining of Wasserstein GANs. NeurIPS 2017

Wasserstein GAN

- Regularised discriminator can be used to score the provided inputs

$$\min_{\theta} \max_{\phi} E_{p_{data}(x)}[D_\phi(x)] - E_{p_\theta(x)}[D_\phi(x)] \text{ subject to } \| D_\phi(\cdot) \|_L \leq 1$$

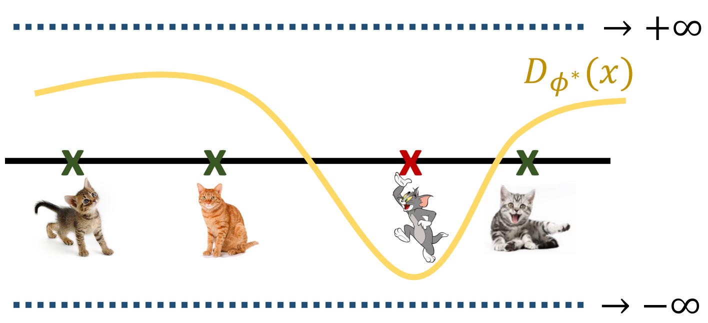
Discriminator should assign high scores to data inputs and low scores to fake inputs
At the same time, discriminator should be smooth to provide useful gradient for learning G_θ

- $\| D_\phi(\cdot) \|_L \leq 1$ is the Lipschitz continuity constraint

$$\|\nabla_x D_\phi(x)\|_2 \leq 1 \text{ for all } x$$

- Equivalent to minimising the Wasserstein distance :

$$W_2(p_{data}(x), p_\theta(x)) := \sup_{\phi: \| D_\phi(\cdot) \|_L \leq 1} E_{p_{data}(x)}[D_\phi(x)] - E_{p_\theta(x)}[D_\phi(x)]$$



Arjovsky et al. Wasserstein Generative Adversarial Networks. ICML 2017
Gulrajani et al. Improvedtraining of Wasserstein GANs. NeurIPS 2017

Wasserstein GAN

- Practical implementation: WGAN-GP

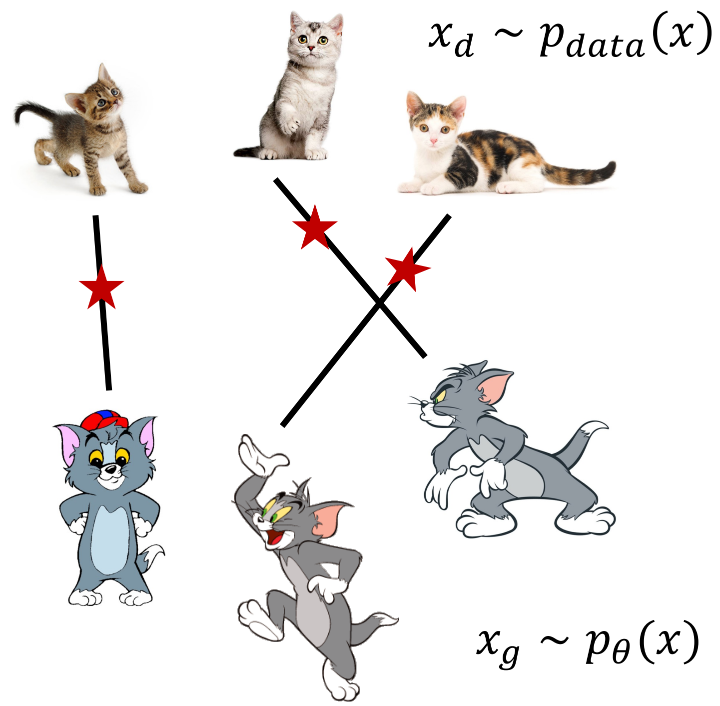
Regulariser to enforce the Lipschitz continuity constraint

$$\min_{\theta} \max_{\phi} E_{p_{data}(x)}[D_\phi(x)] - E_{p_\theta(x)}[D_\phi(x)] + \boxed{\lambda E_{\hat{p}(x)}[(\|\nabla_x D_\phi(x)\|_2 - 1)^2]}$$

- $\hat{p}(x)$ is defined by the following sampling procedure:

$$x_d \sim p_{data}(x)$$
$$x_g \sim p_\theta(x)$$
$$\alpha \sim Uniform([0,1])$$
$$x = \alpha x_d + (1-\alpha) x_g$$

- Training strategy is similar to the original GAN
  - Double-loop algorithm
  - Minibatch sampling



Arjovsky et al. Wasserstein Generative Adversarial Networks. ICML 2017
Gulrajani et al. Improvedtraining of Wasserstein GANs. NeurIPS 2017