Outline: Generative Adversarial Networks (GANs)

Concepts of data synthesis

- Sampling from distributions
  - Discrete distributions
  - Empirical distributions
  - Continuous distributions
- Learning parametric distributions then sampling
  - Learning parametric distributions
  - Sampling from learned parametric distributions
- Learning to sample from an unknown distribution
  - Learning autoencoding generative models
  - Learning adversarially
- Synthesizing images
  - Natural images: Progressive Growing of GANs (2018)
  - Art: Creative Adversarial Networks (2017)
  - Anime character generation: Towards the Automatic Anime Characters Creation (2017)

Adversarial Learning

- Basics and notation
  - Distribution, expectation, generator, discriminator and objective
- Learning adversarially revisited
- Conditional adversarial learning
- Adversarial disentanglement

Generative Adversarial Networks

Lecture 1: Concepts of Data Synthesis

Ricardo Henao
Duke University

Sampling from discrete distributions

Example 1: coin flip

Heads
Probability(Heads)=0.5

Tails
Probability(Tails)=0.5

Equally likely

Assumption: the coin is fair.

Sampling from discrete distributions

Example 2: die rolling

Equally likely

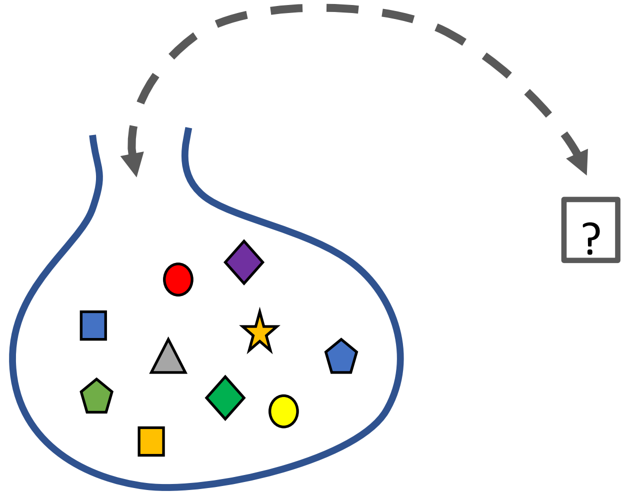
Probability(1)=Probability(2) … =Probability(6)=1/6

Assumption: the dice are fair and independent.

Sampling from empirical distributions

Example 3: bag of objects

Probability( )=1/10
Probability( )=1/10
⋮
Probability( )=1/10



Assumption: the objects in the bag are unique.

Note: we could sample without replacement:

Probability(any 1^{st} draw) = 1/10, Probability(any 2^{nd} draw | 1^{st} draw) = 1/9, …, Probability(any 10^{th} draw | 9^{th} draw, …) = 1

Sampling from empirical distributions

Example 3: handwritten digits (MNIST, N=60,000 images)

Sampling from empirical distributions

Example 3: natural images (ImageNet, N>1M images)

Sampling from a continuous distribution

Example 4: numbers between 0 and 1

From previous example, let the bag have a large collection of objects:

$$1,2, \ldots, 2^d - 1$$

Draw one number from the bag (x) at random:

$$\frac{x}{2^d} \in (0,1)$$

Each number in the bag has the same probability

$$\text{Probability}(x) = \frac{1}{2^d - 1}$$

Computationally, "drawing from the bag" is implemented by generating the sequence 1, 2, … in random order.

Sampling from empirical distributions

Examples 1-4 represent *uniform distributions* (the probability of every outcome is the same).

The uniform distribution between 0 and 1 can be denoted as Uniform(0,1).

Sampling from empirical distributions

Examples 1-4 represent *uniform distributions* (the probability of every outcome is the same).

The uniform distribution between 0 and 1 can be denoted as Uniform(0,1).

Questions:

1. How do we know that the coin/die is fair?
2. How do we know if two dice produce the same results (*i.e.*, have the same probabilities)?
3. How do we know if a bag of objects has more that one type of object?
4. What if we do not know the total number of objects in the bag of objects?
5. What if we want objects like in the bag but that we know for sure are not in the bag of objects?

Sampling from empirical distributions

Examples 1-4 represent *uniform distributions* (the probability of every outcome is the same).

The uniform distribution between 0 and 1 can be denoted as Uniform(0,1).

Questions:

1. How do we know that the coin/die is fair?
2. How do we know if two dice produce the same results (*i.e.*, have the same probabilities)?
3. How do we know if a bag of objects has more that one type of object?
4. What if we do not know the total number of objects in the bag of objects?
5. What if we want objects like in the bag but that we know for sure are not in the bag of objects?

Generative Adversarial Learning

Learning to sample from an unknown distribution

Example 3: handwritten digits (MNIST, N=60,000 images)

Complex object

 $= f(\epsilon)$

Simple object

Generator
MLP (day1) or CNN (day 2)

Learning to sample from an unknown distribution

Example 3: handwritten digits (MNIST, N=60,000 images)

Complex object

 $= f(\epsilon)$

Simple object

Generator
MLP (day1) or CNN (day 2)

How do we learn $f(\epsilon)$?

Learning to sample from an unknown distribution

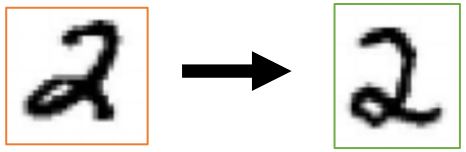
Example 5: handwritten digits (MNIST, N=60,000 images)

Real | Generated

x | $x = f(\epsilon; \theta)$

Generation (forward)

Learn θ so



Learning (backward)

Learning to sample from an unknown distribution

Example 5: handwritten digits (MNIST, N=60,000 images)

Real

x

Generated

$x = f(\epsilon; \theta)$

Generation (forward)

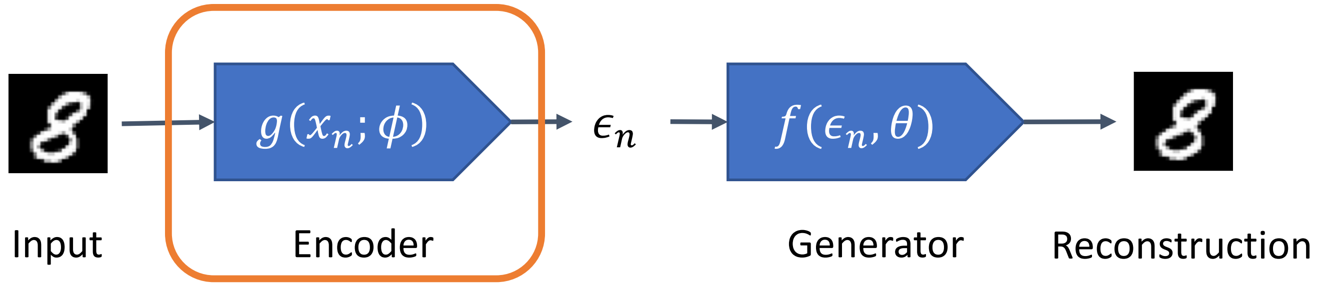
Learn θ so  

Learning (backward)

Which ϵ corresponds to x?

Autoencoding generative models

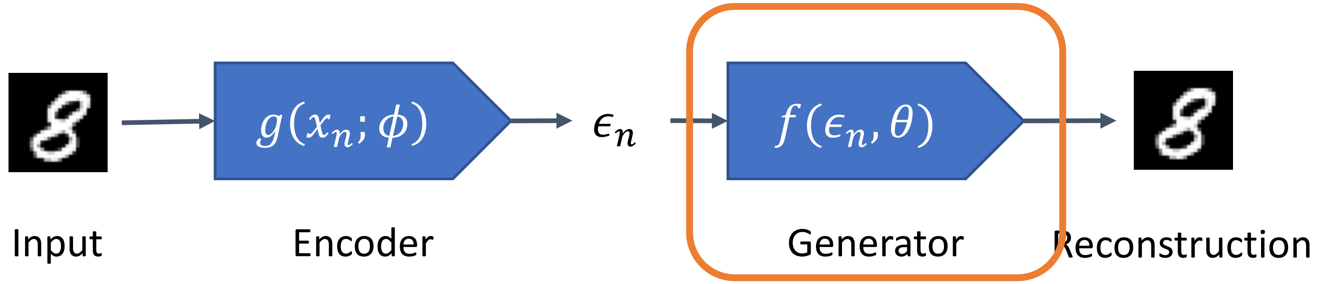
Example 5: handwritten digits (MNIST, N=60,000 images)



Step 1 (Encode): Generate ϵ from input

Autoencoding generative models

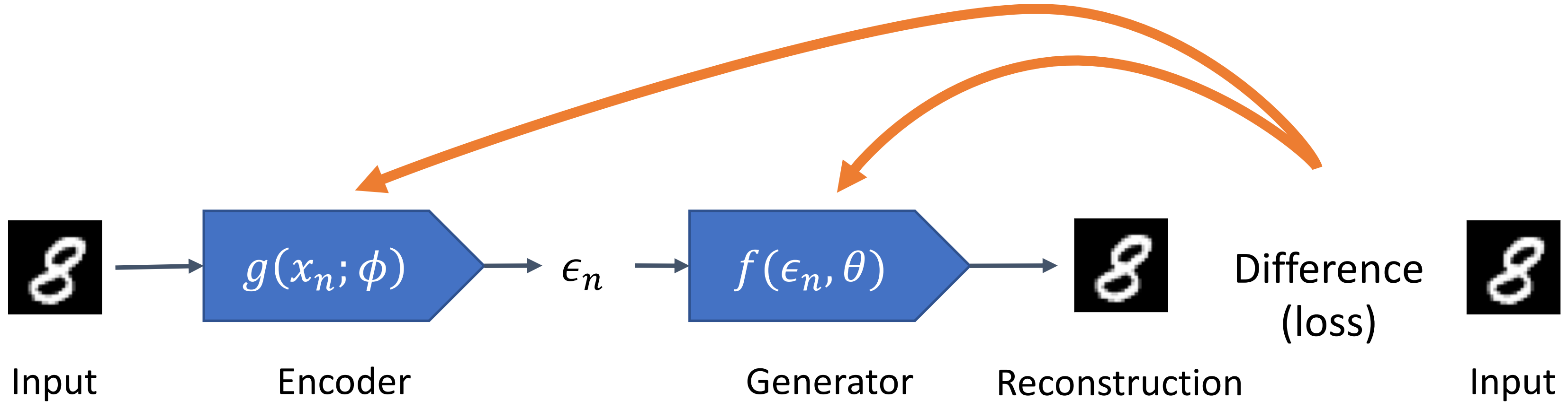
Example 5: handwritten digits (MNIST, N=60,000 images)



Step 2 (Decode): Generate reconstruction from ϵ

Autoencoding generative models

Example 5: handwritten digits (MNIST, N=60,000 images)

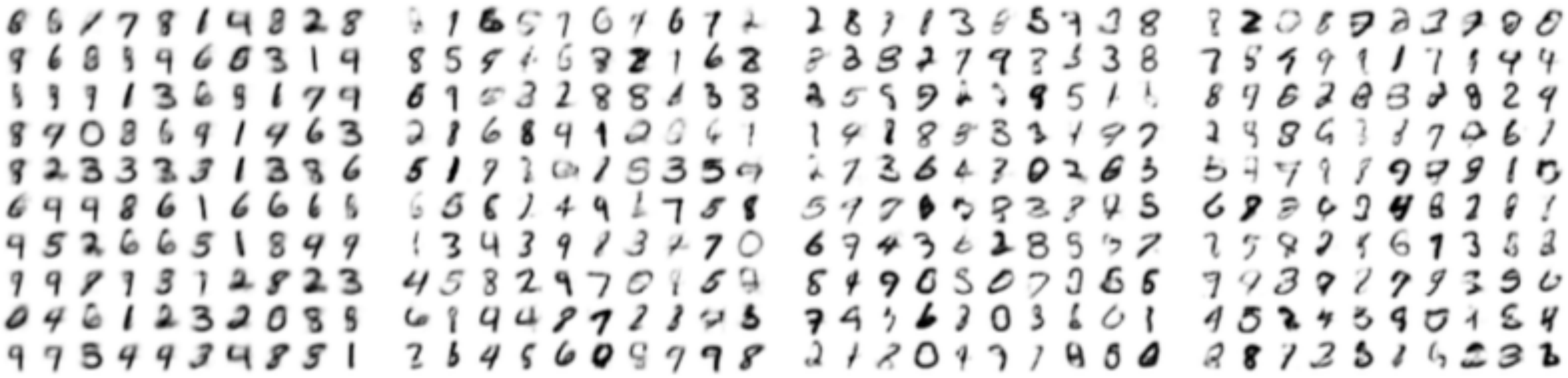


Step 3 (Learn): Minimize reconstruction error

Autoencoding generative models

Example 5: handwritten digits (MNIST, N=60,000 images)

Samples generated by an autoencoder:



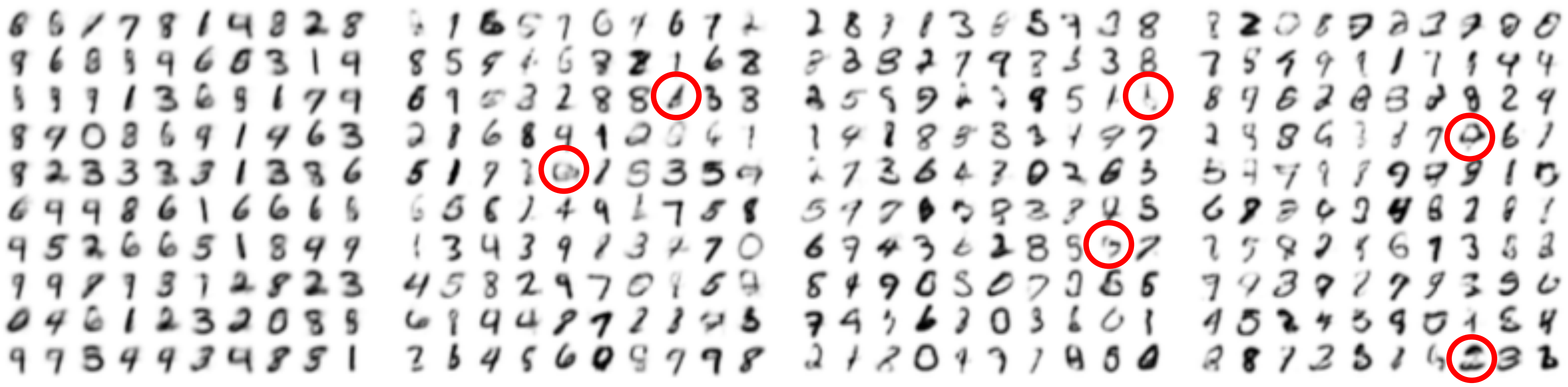
(a) 2-D latent space (b) 5-D latent space (c) 10-D latent space (d) 20-D latent space

Kingma and Welling, ICLR 2014

Autoencoding generative models

Example 5: handwritten digits (MNIST, N=60,000 images)

Samples generated by an autoencoder:



Problem: some samples do not look like digits.

Learning adversarially (analogy)

What if we had a *critic* judging the quality of the synthesized samples?

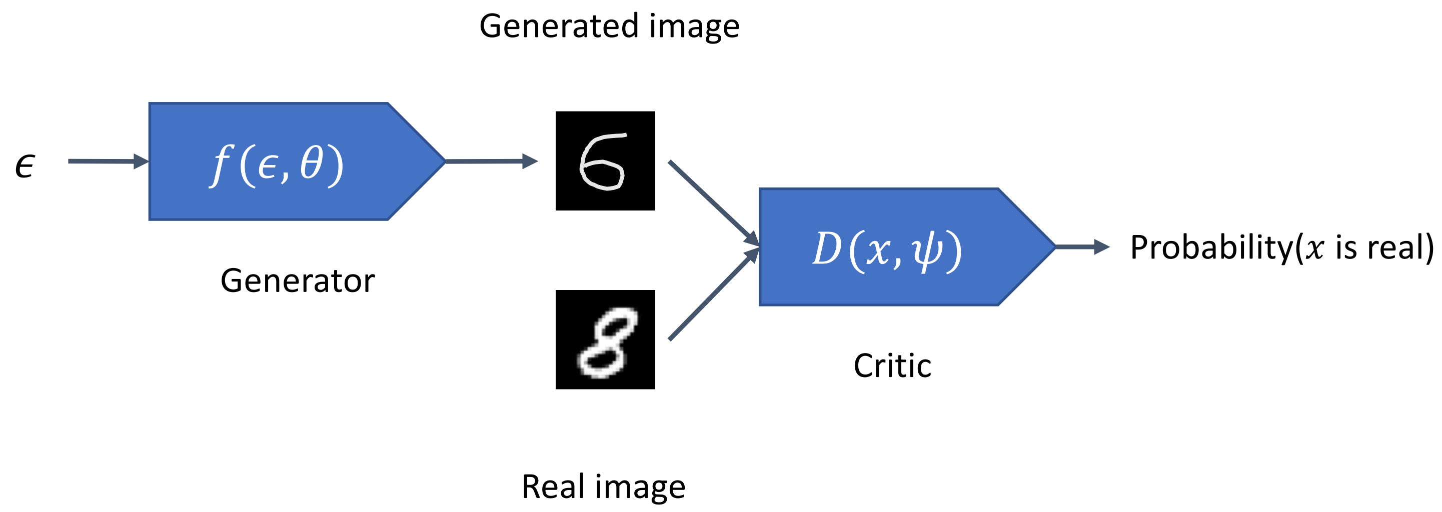
Wine critic

Wine collection

Wine generator

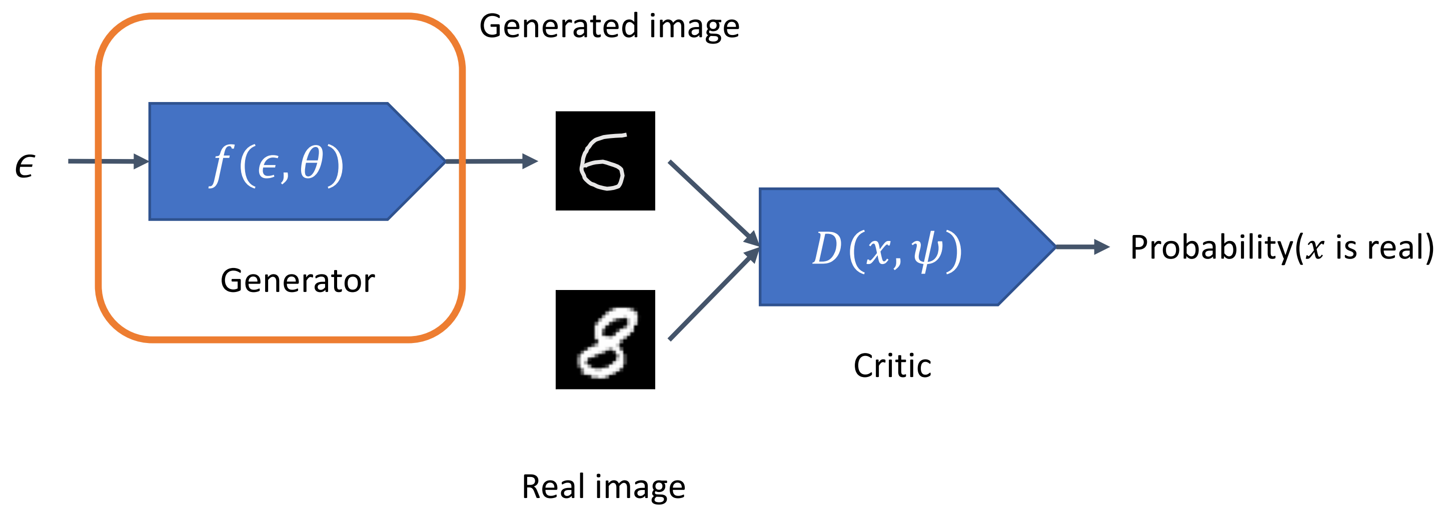
Learning adversarially

Example 5: handwritten digits (MNIST, N=60,000 images)



Learning adversarially

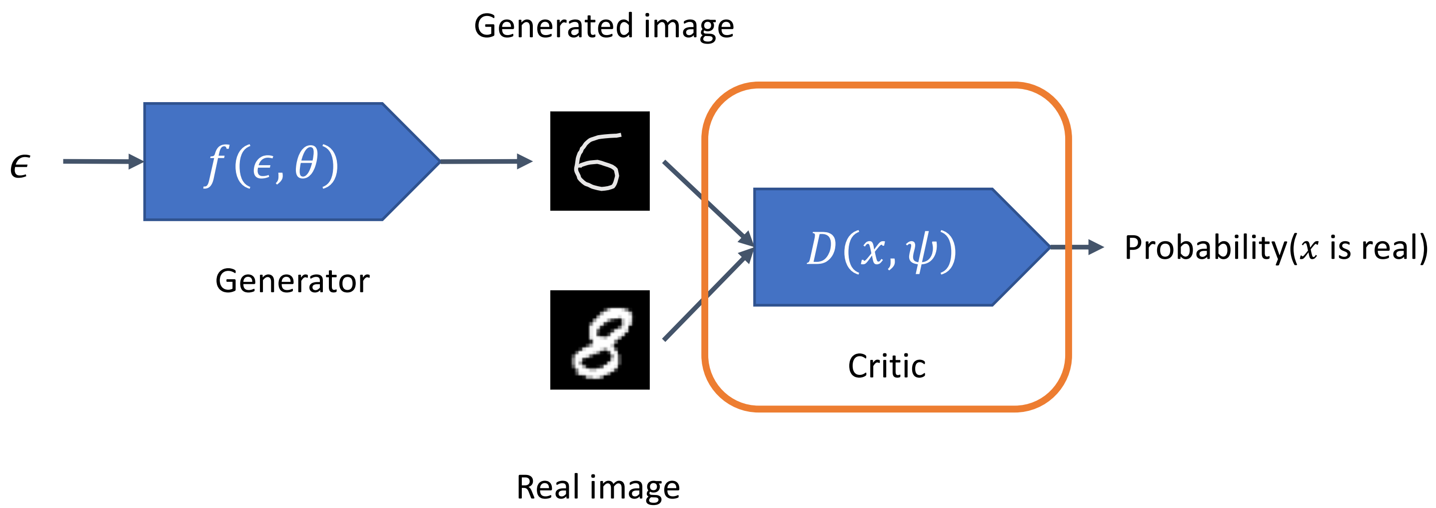
Example 5: handwritten digits (MNIST, N=60,000 images)



Step 1: Learn θ (generator) so the generator misleads critic.

Learning adversarially

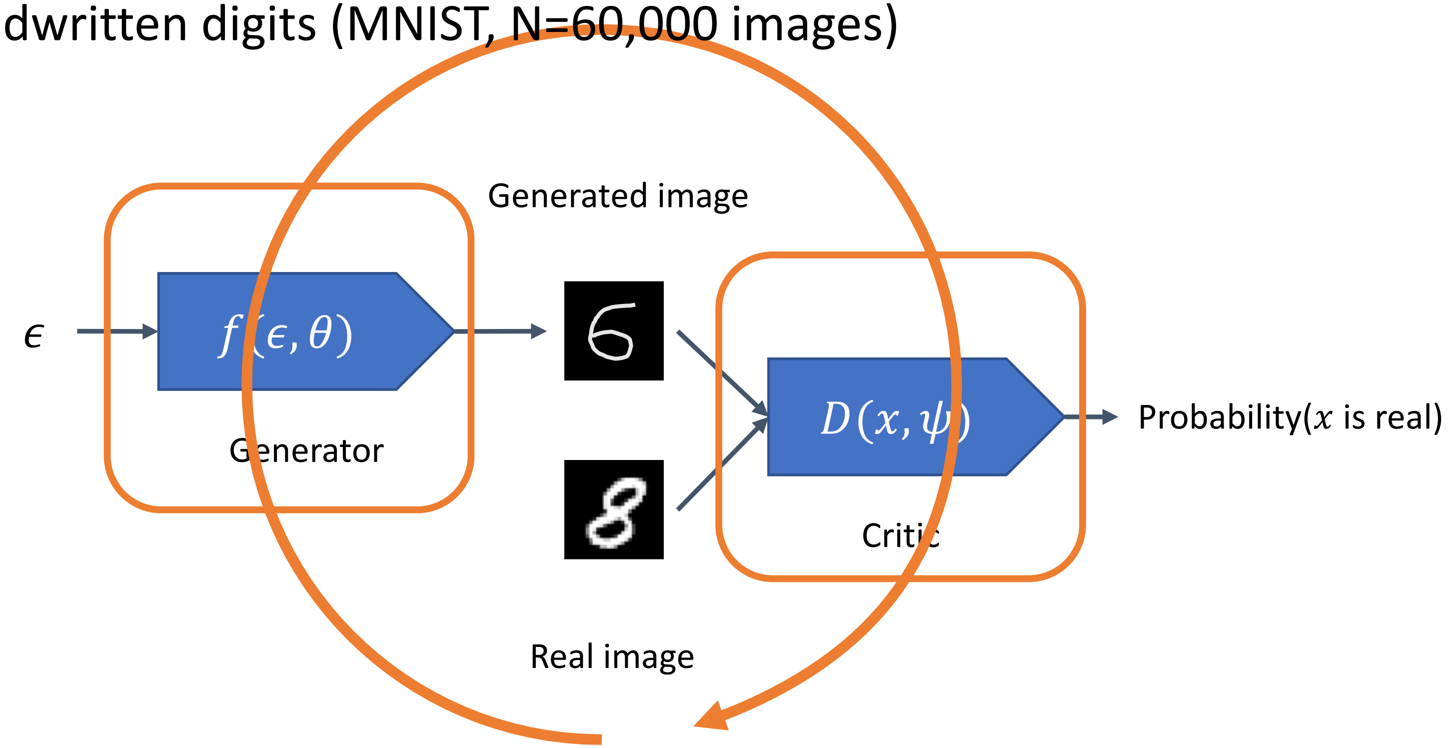
Example 5: handwritten digits (MNIST, N=60,000 images)



Step 2: Learn ψ (critic) so critic doesn't get mislead.

Learning adversarially

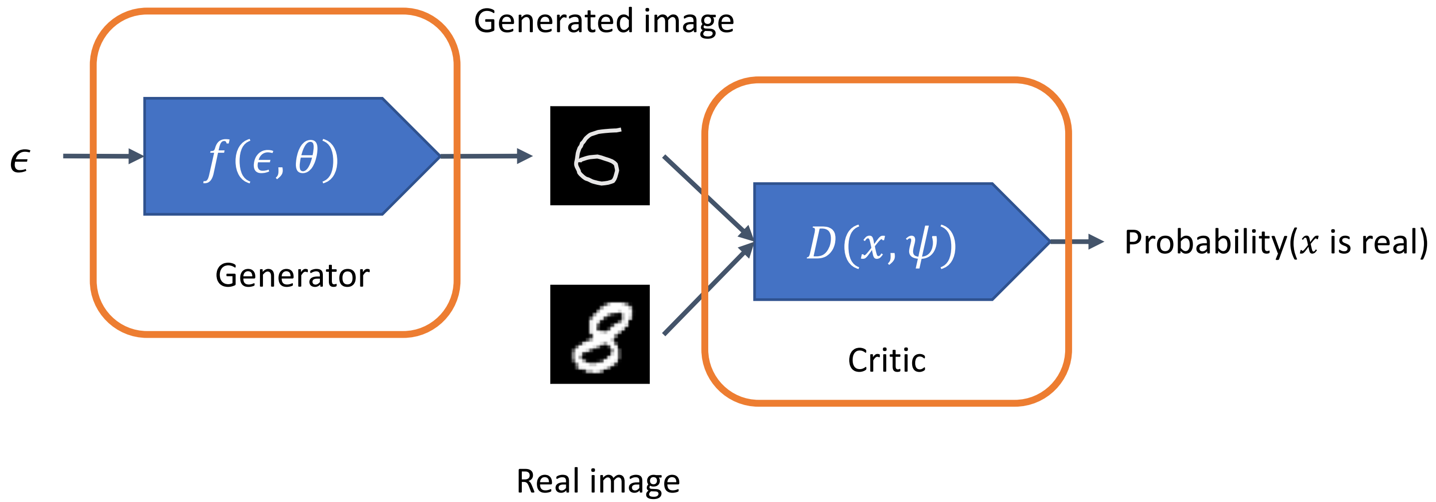
Example 5: handwritten digits (MNIST, N=60,000 images)



Step 3: Repeat, generator and critic get better.

Learning adversarially

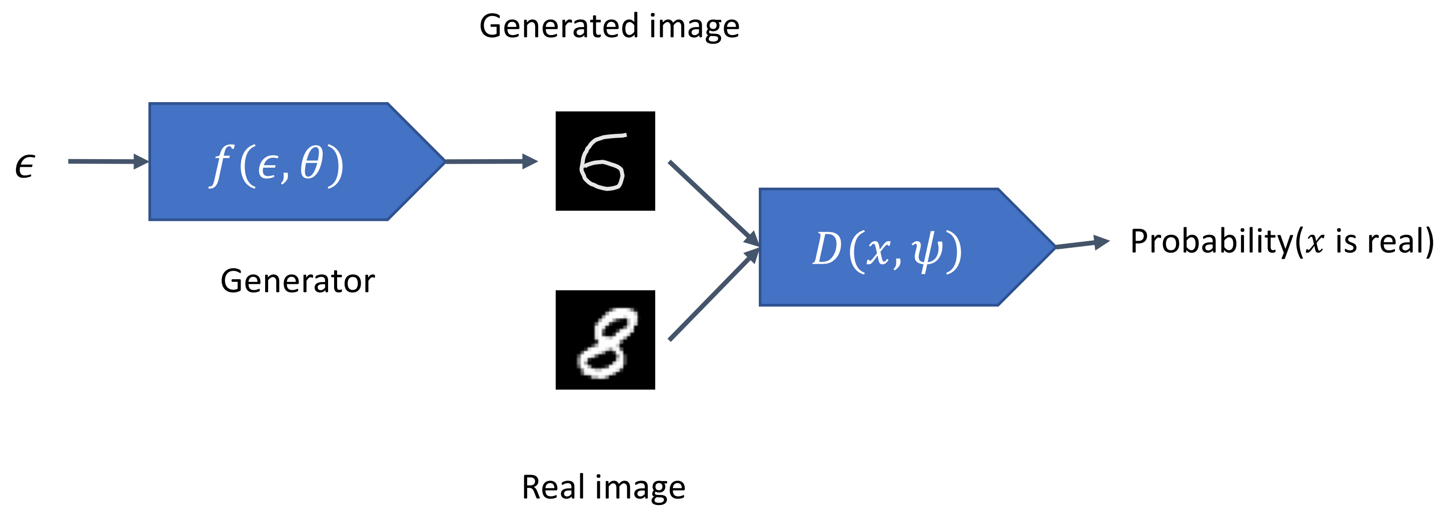
Example 5: handwritten digits (MNIST, N=60,000 images)



This framework is a game between two *adversaries*.

Learning adversarially

Example 5: handwritten digits (MNIST, N=60,000 images)



Formally:

- We learn θ by <u>minimizing</u> the critic's ability to identify samples from the generator as not real.
- We learn ψ by <u>maximizing</u> the chances for the critic to correctly identify real samples.
- This framework, *Generative Adversarial Networks* (GANs), corresponds to a <u>minimax 2-player game</u>.

Learning adversarially

Example 5: handwritten digits (MNIST, N=60,000 images)

Samples generated by a generator learned adversarially:

Closest image from MNIST set

Generated samples

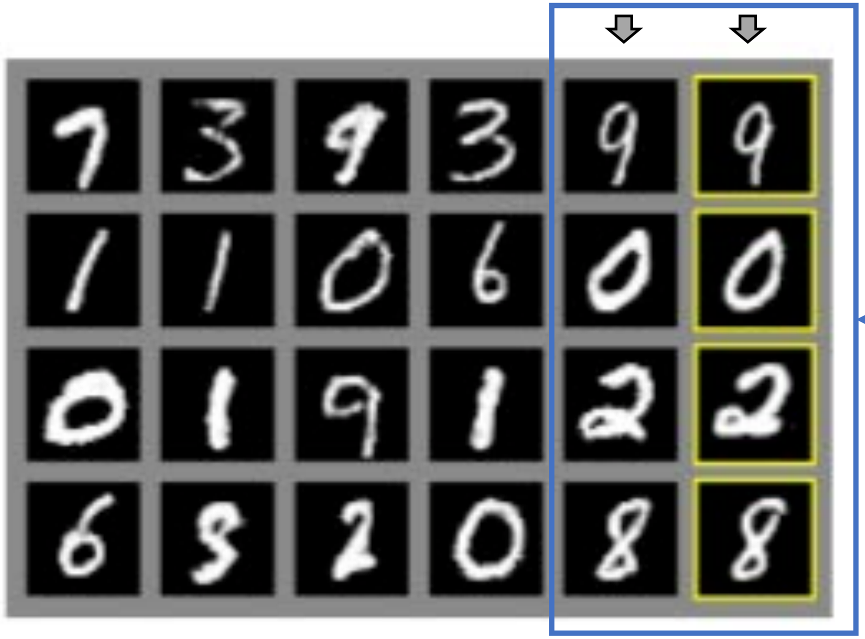
Goodfellow *et al*, NIPS 2014

Learning adversarially

Example 5: handwritten digits (MNIST, N=60,000 images)

Samples generated by a generator learned adversarially:

Closest image from MNIST set



The model generalizes

Generated samples

Goodfellow *et al*, NIPS 2014

Synthesizing images: Natural images

Progressive growing of GANs

1024x1024 images using CelebA-HQ dataset

256x256 images using LSUN dataset

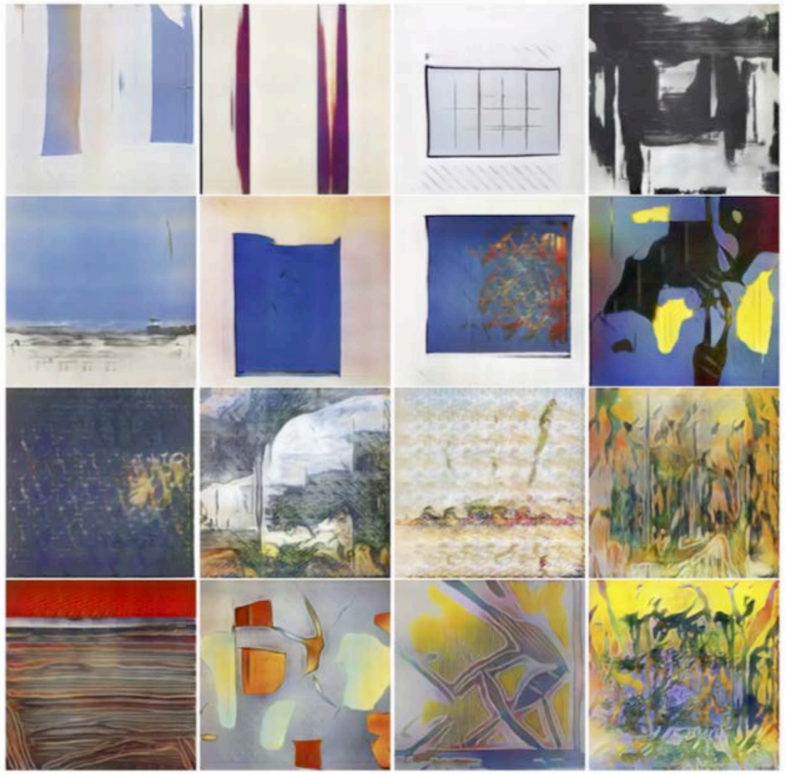
Karras *et al*, NIPS 2018

Synthesizing images: Natural images

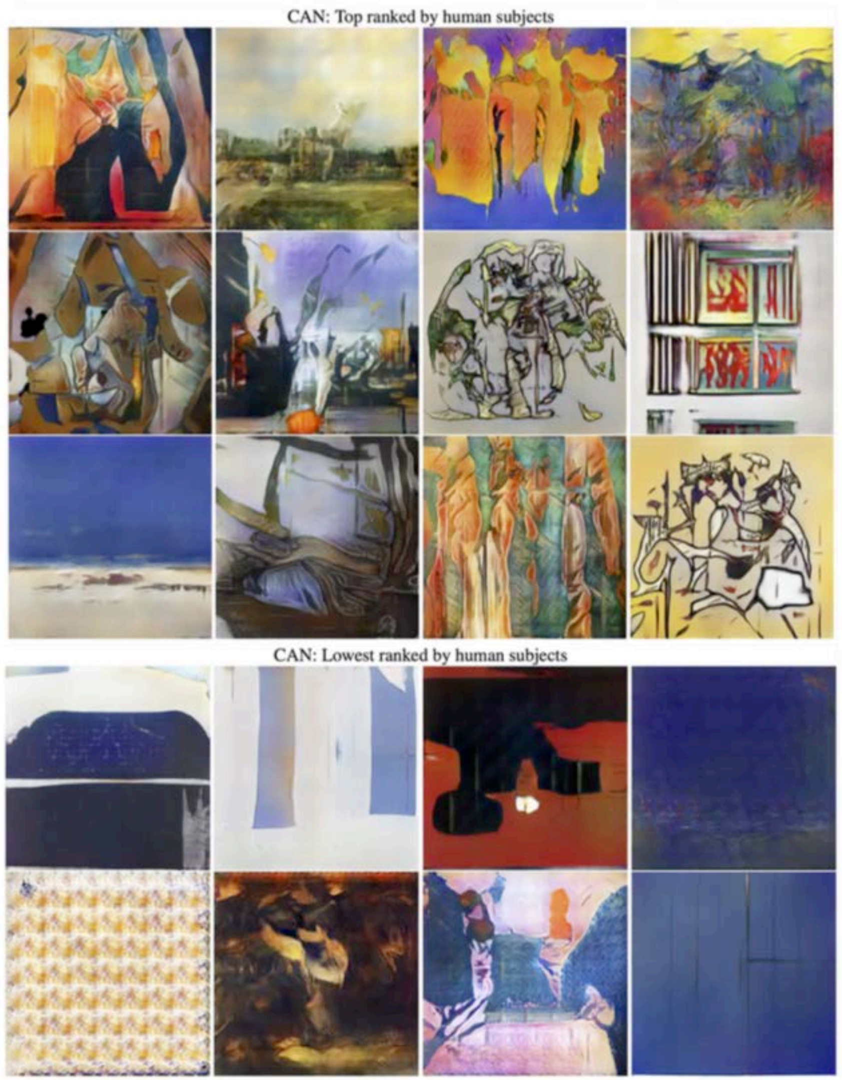
Quality progression of face generators

Brundage *et al*, arXiv 2018

Creative adversarial networks

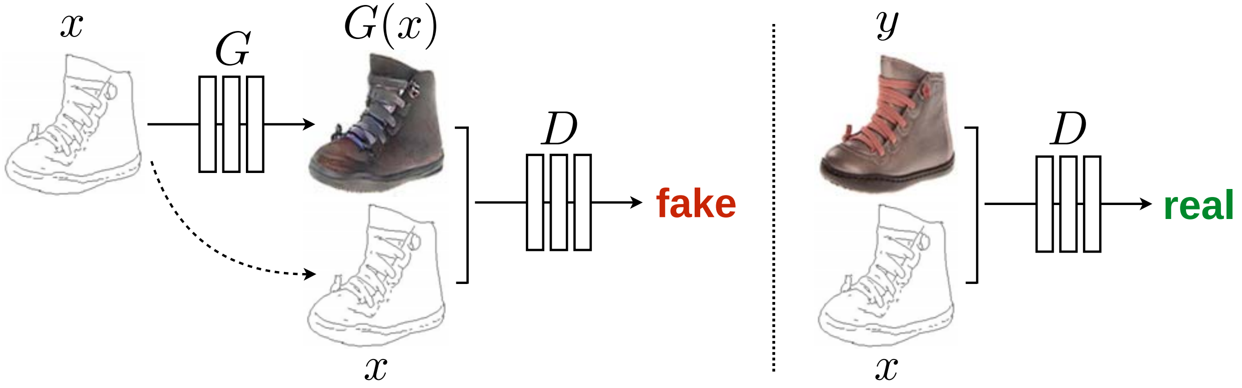


256x256 image samples using WikiArt dataset



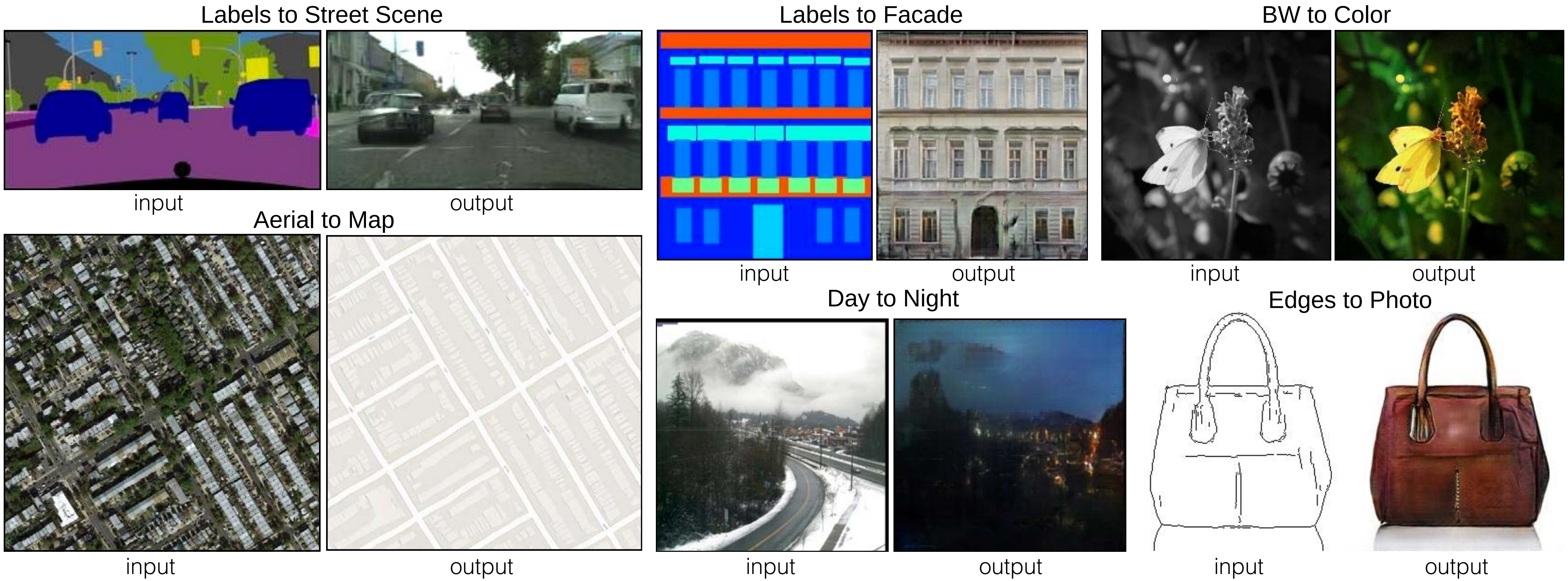
Elgammal *et al*, ICCC 2017

Image to image translation



Isola *et al*, CVPR 2017

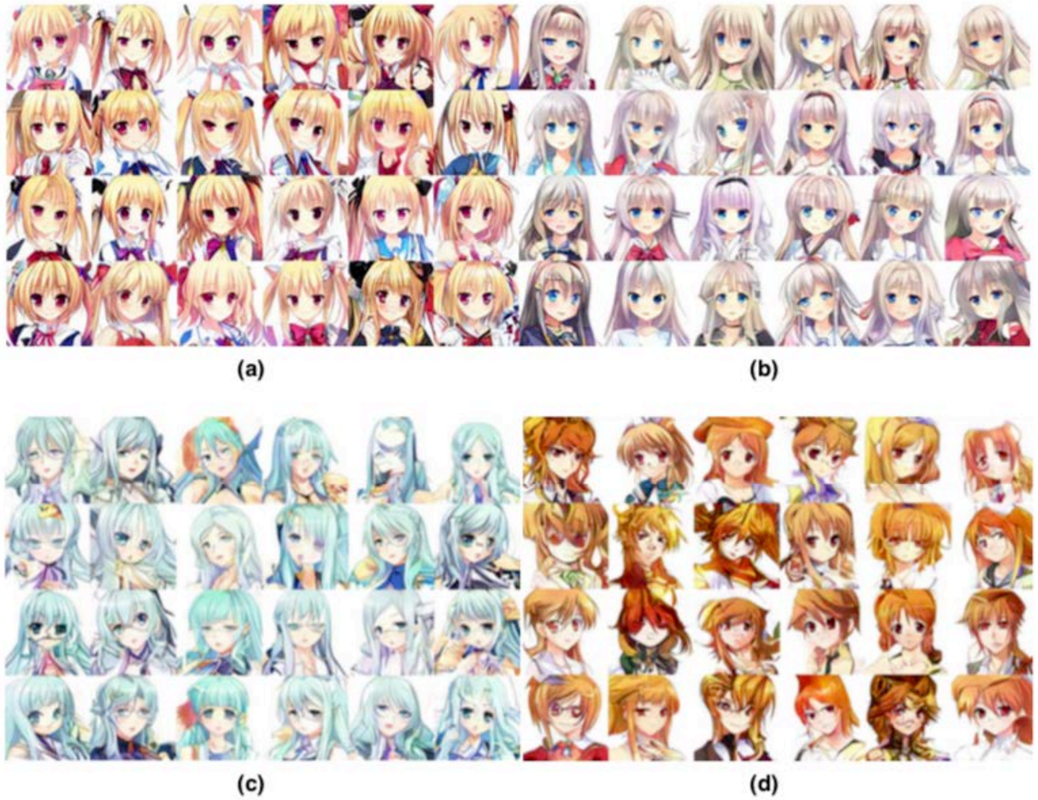
Image to image translation



Isola *et al*, CVPR 2017

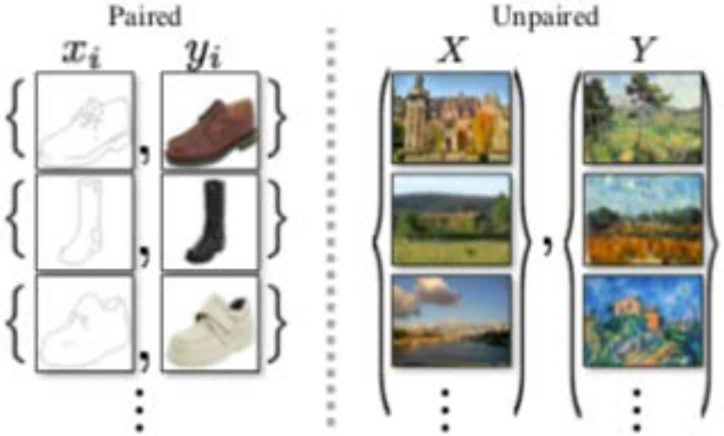
Automatic Anime Characters Creation

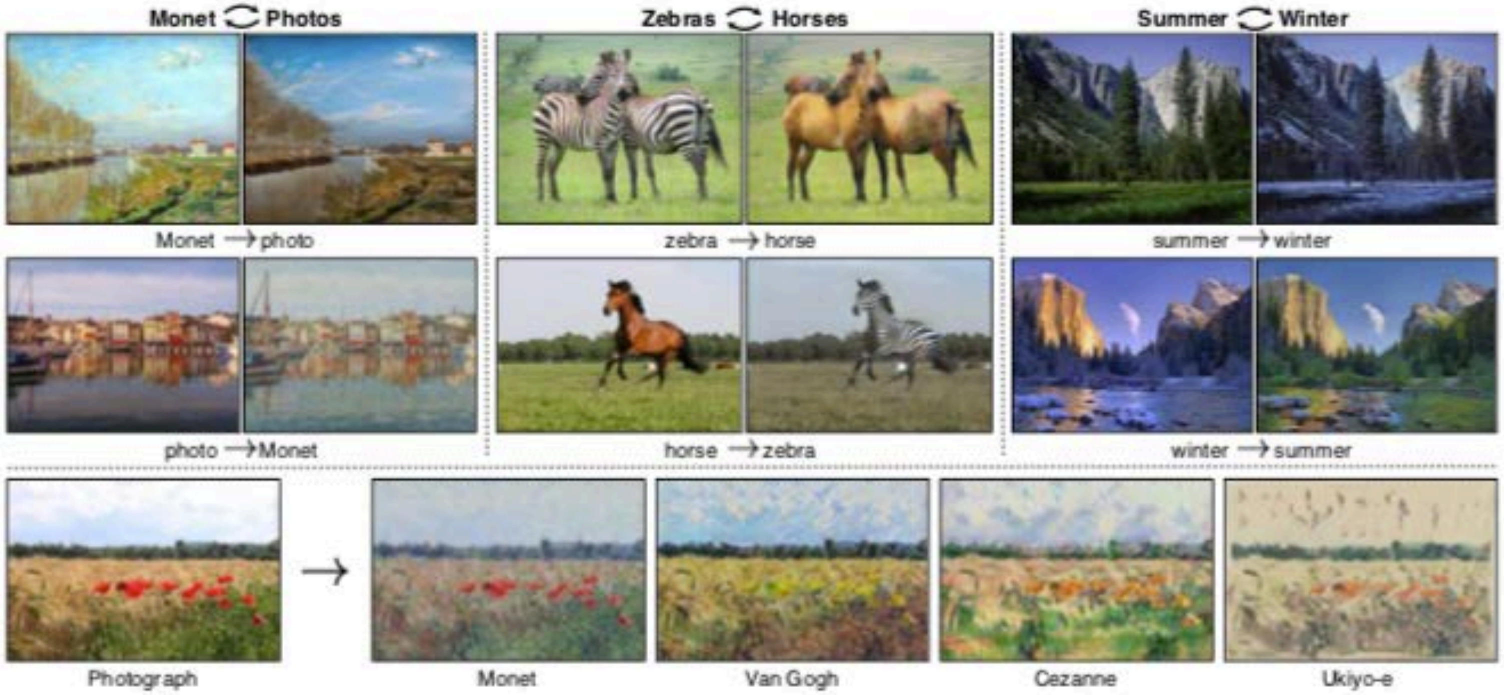
128x128 image samples



Jin *et al*, Comiket 2017

Unpaired Image to image translation



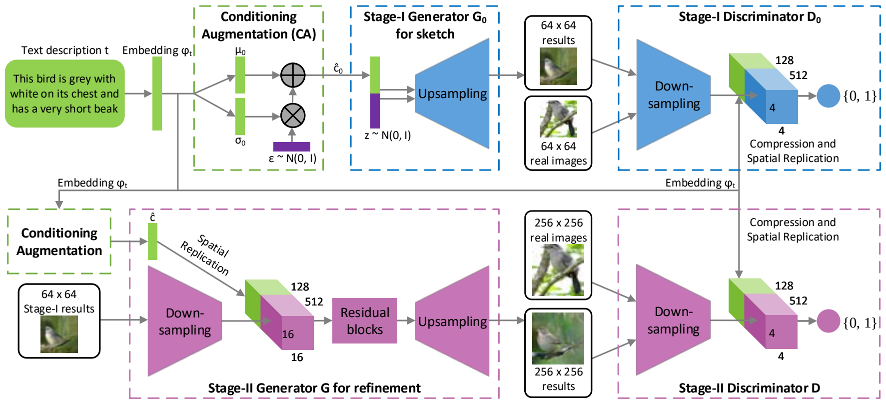


Zhu *et al*, ICCV 2017

Unpaired Image to image translation (style transfer)

Zhu *et al*, ICCV 2017

Text to image translation



Zhang *et al*, ICCV 2017

Text to image translation

Zhang *et al*, ICCV 2017

Text to image translation

Zhang *et al*, ICCV 2017

Generative Adversarial Networks

Lecture 2: Adversarial Learning

Ricardo Henao
Duke University

Basics and notation

We denote the distribution of x (usually unknown) as $p(x)$ and $x \sim p(x)$ as a sample x of $p(x)$.

Basics and notation

We denote the distribution of x (usually unknown) as $p(x)$ and $x \sim p(x)$ as a sample x of $p(x)$.

We denote the expected value of x as $\mathbb{E}_{p(x)}[x]$, in practice estimated as an average: $\frac{1}{M}\sum_{m=1}^{M} x_m$, for $x \sim p(x)$.

Basics and notation

We denote the distribution of x (usually unknown) as $p(x)$ and $x \sim p(x)$ as a sample x of $p(x)$.

We denote the expected value of x as $\mathbb{E}_{p(x)}[x]$, in practice estimated as an average: $\frac{1}{M}\sum_{m=1}^{M} x_m$, for $x \sim p(x)$.

We use $x = f(\epsilon, \theta)$ to denote a generator for x parameterized by θ and $\epsilon \sim p(\epsilon)$ are samples from a simple distribution.

Basics and notation

We denote the distribution of x (usually unknown) as $p(x)$ and $x \sim p(x)$ as a sample x of $p(x)$.

We denote the expected value of x as $\mathbb{E}_{p(x)}[x]$, in practice estimated as an average: $\frac{1}{M}\sum_{m=1}^{M} x_m$, for $x \sim p(x)$.

We use $x = f(\epsilon, \theta)$ to denote a generator for x parameterized by θ and $\epsilon \sim p(\epsilon)$ are samples from a simple distribution.

We use $p(x \text{ is real}) = D(x, \psi)$ to denote the critic for x parameterized by ψ.

Basics and notation

We denote the distribution of x (usually unknown) as $p(x)$ and $x{\sim}p(x)$ as a sample x of $p(x)$.

We denote the expected value of x as $\mathbb{E}_{p(x)}[x]$, in practice estimated as an average: $\frac{1}{M}\sum_{m=1}^{M} x_m$, for $x{\sim}p(x)$.

We use $x = f(\epsilon, \theta)$ to denote a generator for x parameterized by θ and $\epsilon{\sim}p(\epsilon)$ are samples from a simple distribution.

We use $p(x \text{ is real}) = D(x, \psi)$ to denote the critic for x parameterized by ψ.

The objective is the expression we wish to optimize (minimize, maximize or both).

Basics and notation

We denote the distribution of x (usually unknown) as $p(x)$ and $x \sim p(x)$ as a sample x of $p(x)$.

We denote the expected value of x as $\mathbb{E}_{p(x)}[x]$, in practice estimated as an average: $\frac{1}{M}\sum_{m=1}^{M} x_m$, for $x \sim p(x)$.

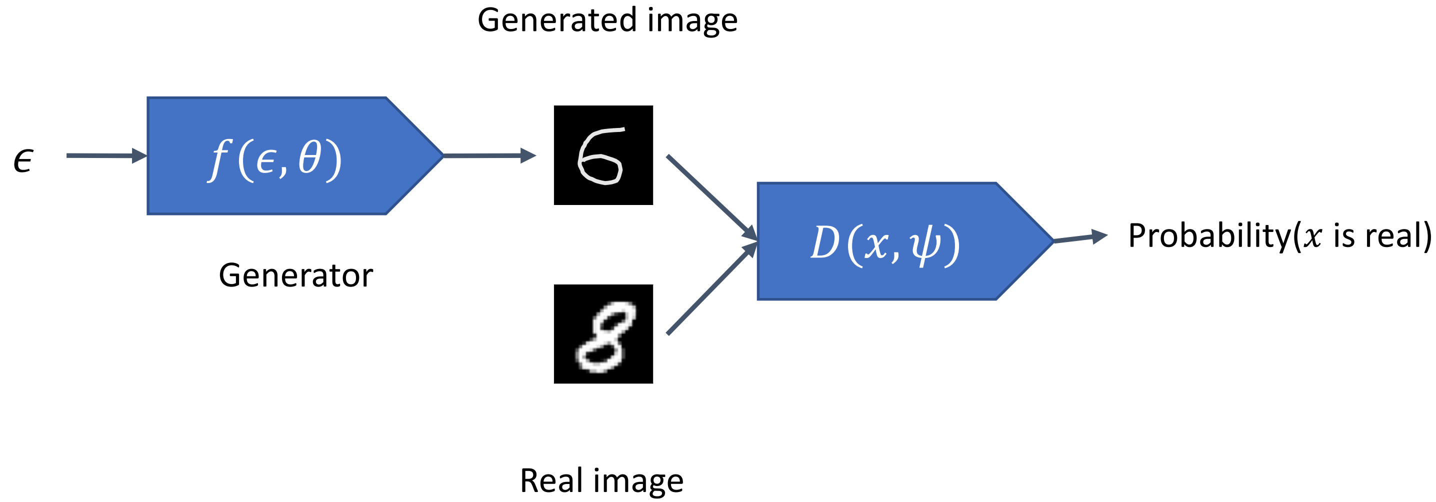
We use $x = f(\epsilon, \theta)$ to denote a generator for x parameterized by θ and $\epsilon \sim p(\epsilon)$ are samples from a simple distribution.

We use $p(x \text{ is real}) = D(x, \psi)$ to denote the critic for x parameterized by ψ.

The objective is the expression we wish to optimize (minimize, maximize or both).

Parameters (θ and ψ) are estimated via backpropagation of the objective function.

Learning adversarially revisited

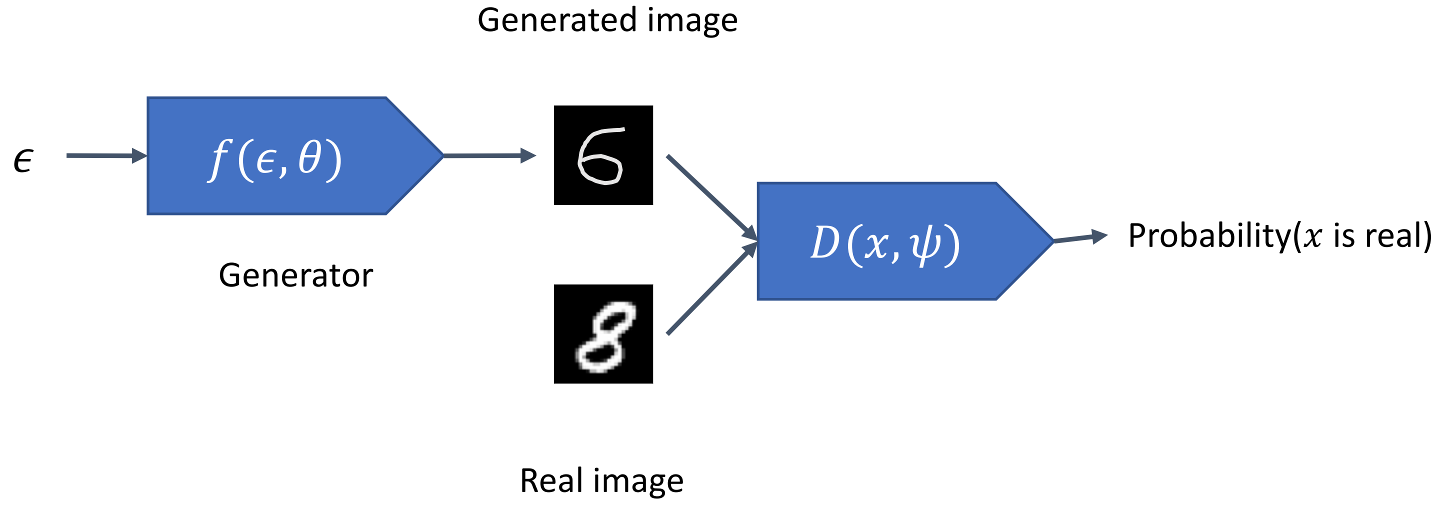


- We learn θ by minimizing the critic's ability to identify samples from the generator as not real.
- We learn ψ by maximizing the chances for the critic to correctly identify real samples.

Formally, the objective $V(\theta, \psi)$ is written as:

$$\text{argmin}_\theta \, \text{argmax}_\psi \quad V(\theta, \psi) = \mathbb{E}_{p(x)}[\log D(x; \psi)] + \mathbb{E}_{p(\epsilon)}[\log(1 - D(f(\epsilon; \theta); \psi)]$$

Learning adversarially revisited



- We learn θ by minimizing the critic's ability to identify samples from the generator as not real.
- We learn ψ by maximizing the chances for the critic to correctly identify real samples.

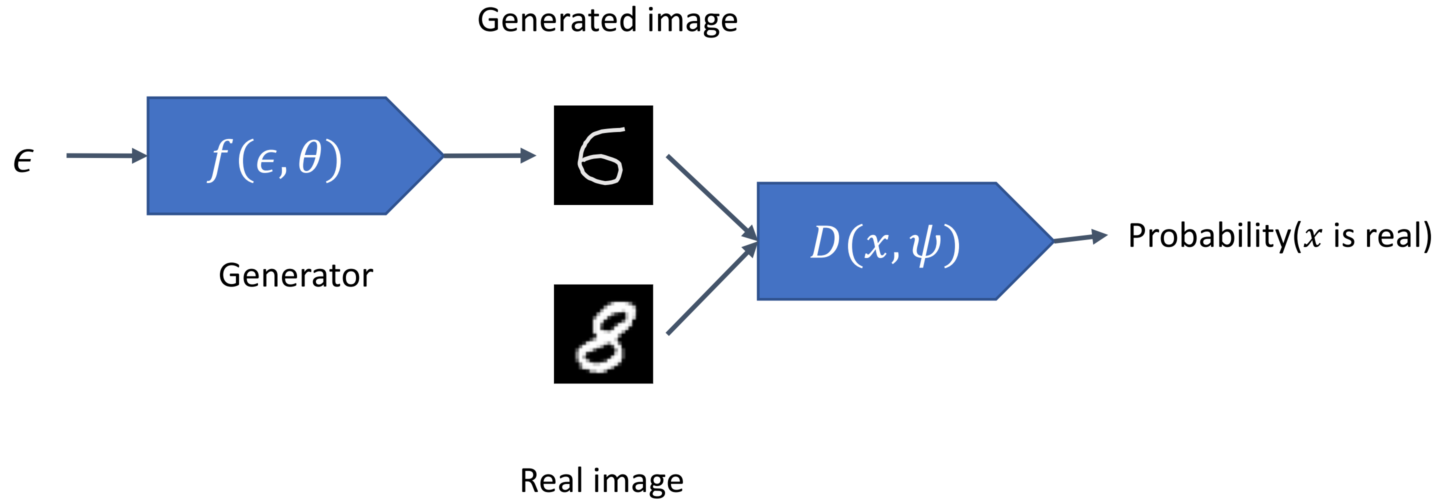
Formally, the objective $V(\theta, \psi)$ is written as:

$$\operatorname{argmin}_\theta \operatorname{argmax}_\psi \quad V(\theta, \psi) = \mathbb{E}_{p(x)}[\log D(x; \psi)] + \mathbb{E}_{p(\epsilon)}[\log(1 - D(f(\epsilon; \theta); \psi)]$$

Sample from empirical distribution (bag of objects)

Sample from Uniform(0,1)

Learning adversarially revisited



- We learn θ by minimizing the critic's ability to identify samples from the generator as not real.
- We learn ψ by maximizing the chances for the critic to correctly identify real samples.

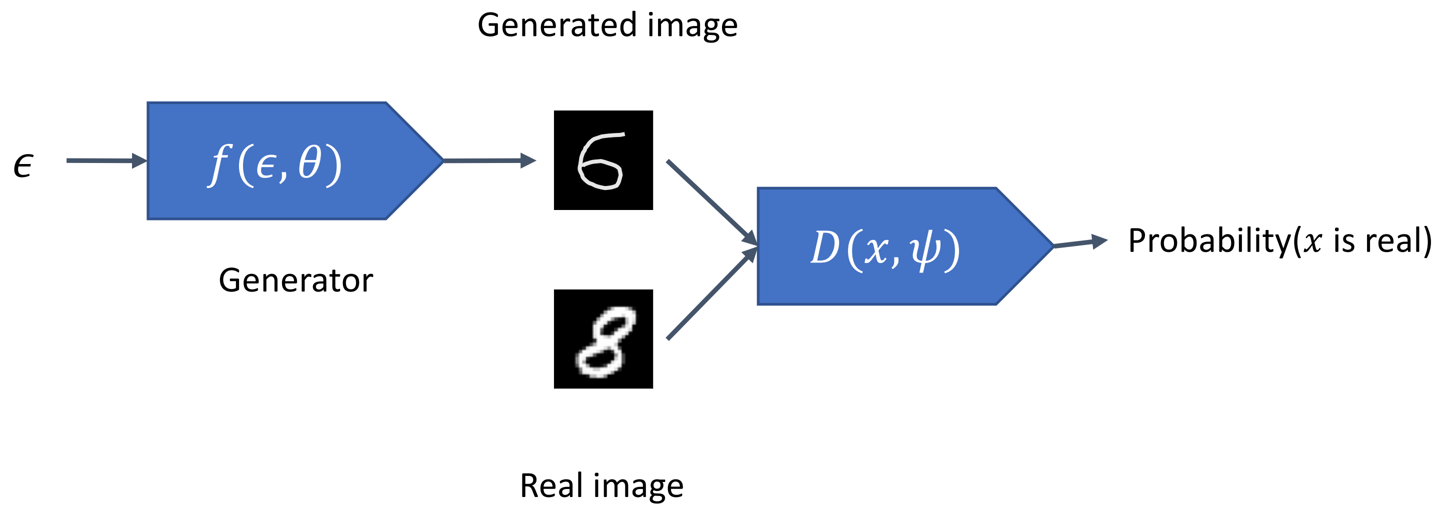
Formally, the objective $V(\theta, \psi)$ is written as:

$$\text{argmin}_\theta \, \text{argmax}_\psi \; V(\theta, \psi) = \mathbb{E}_{p(x)}[\log D(x; \psi)] + \mathbb{E}_{p(\epsilon)}[\log(1 - D(f(\epsilon; \theta); \psi)]$$

The critic is right

The critic cannot be fooled

Learning adversarially revisited



- We learn θ by minimizing the critic's ability to identify samples from the generator as not real.
- We learn ψ by maximizing the chances for the critic to correctly identify real samples.

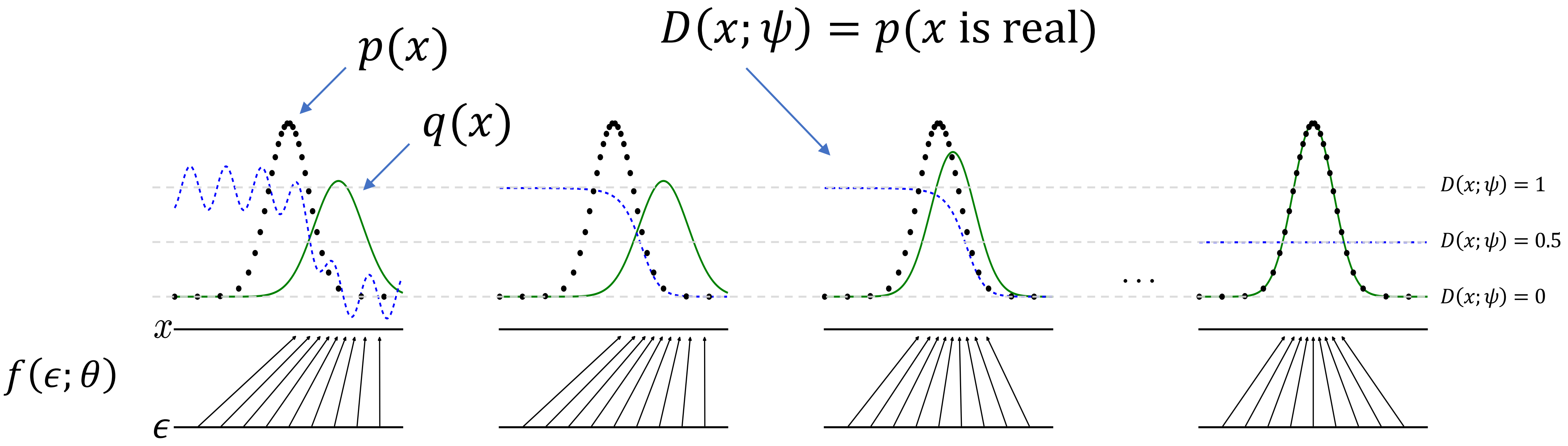
Formally, the objective $V(\theta, \psi)$ is written as:

$$\text{argmin}_\theta \ \text{argmax}_\psi \quad V(\theta, \psi) = \mathbb{E}_{p(x)}[\cancel{\log D(x; \psi)}] + \mathbb{E}_{p(\epsilon)}[\log(1 - D(f(\epsilon; \theta); \psi)]$$

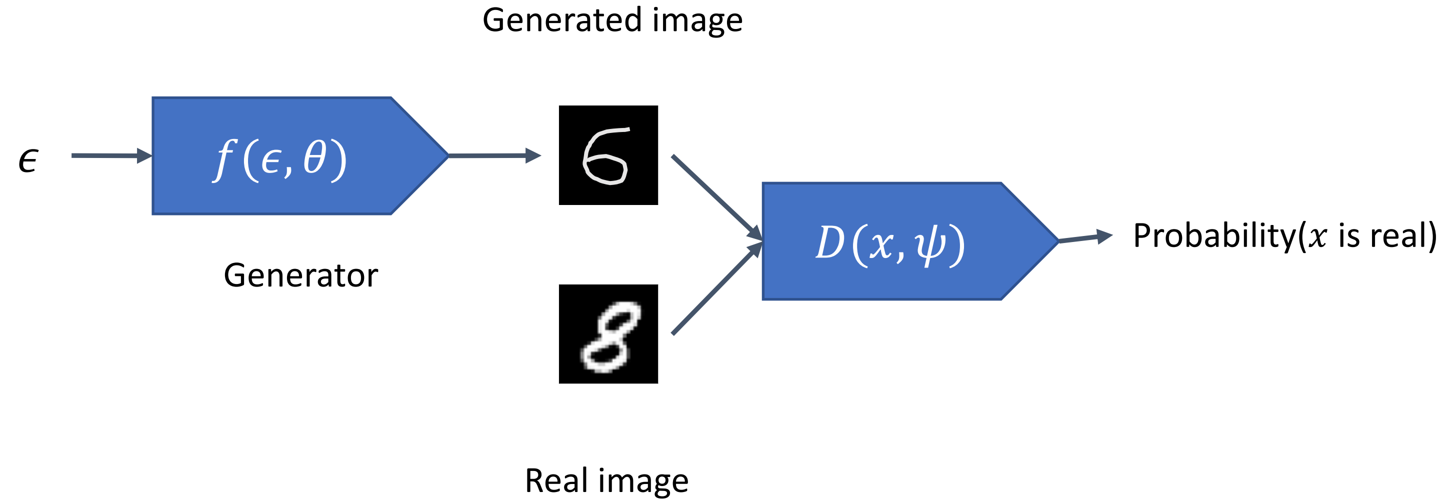
Not dependent of θ

The critic is fooled

Learning adversarially revisited



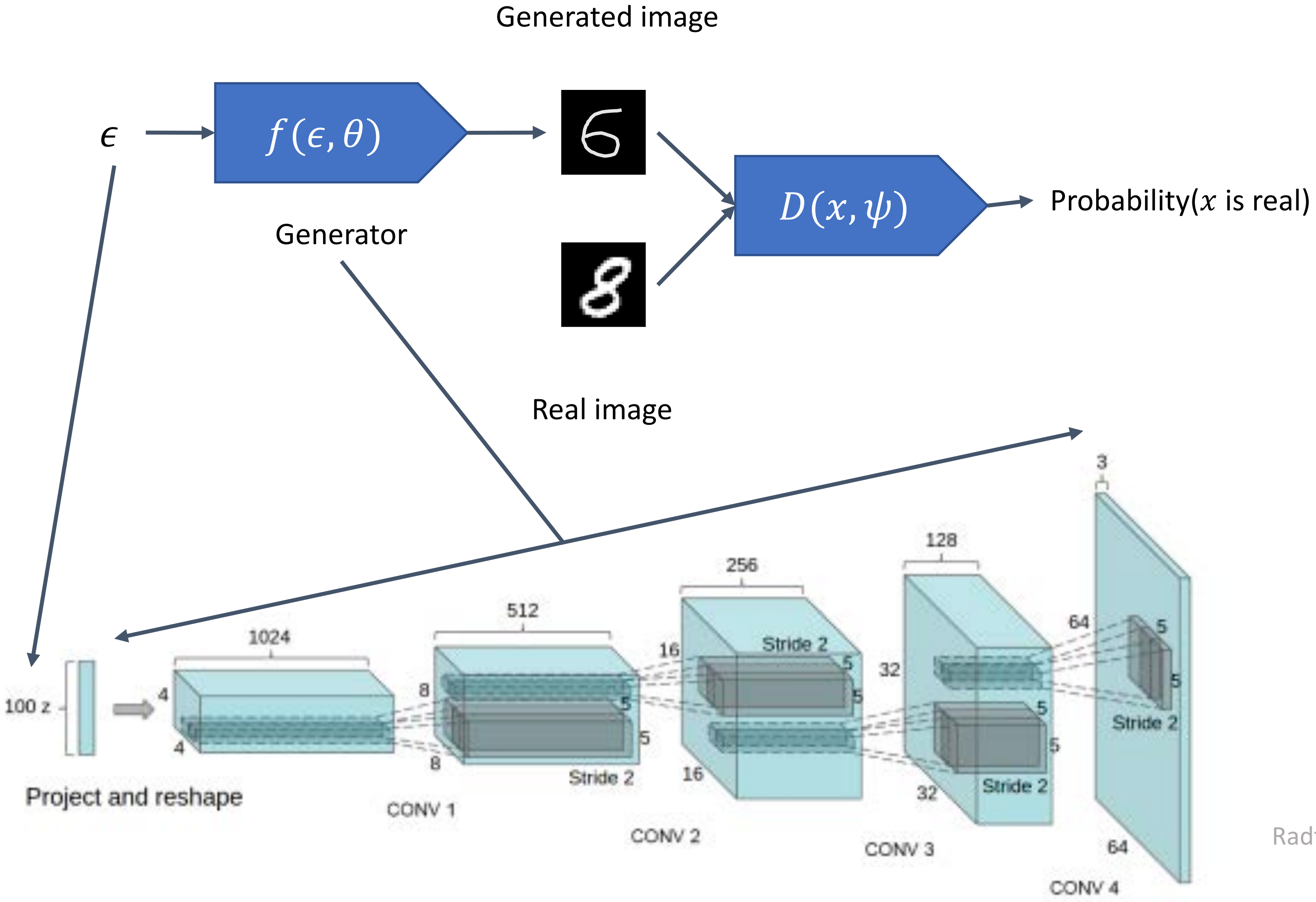
Learning adversarially revisited



For images:

- Specify $f(\epsilon, \theta)$ as a deep convolutional neural network (Day 2).
- Specify $D(x, \psi)$ as a deep convolutional network classifier (Day 2).

Learning adversarially revisited



Radford *et al*, ICLR 2016

Learning adversarially revisited

Algorithm (GAN):

For a number of training iterations do:

Sample minibatch of noise samples $\epsilon_1, \dots, \epsilon_M \sim p(\epsilon)$.
Sample minibatch of real (image) samples $x_1, \dots, x_M \sim p(x)$.

Update critic by stochastic gradient ascent

$$\nabla_\psi \frac{1}{M} \sum_{m=1}^{M} [\log D(x_m; \psi) + \log(1 - D(f(\epsilon_m; \theta); \psi))]$$

Update generator by stochastic gradient descent

$$\nabla_\theta \frac{1}{M} \sum_{m=1}^{M} \log(1 - D(f(\epsilon_m; \theta); \psi))$$

Learning adversarially revisited

Algorithm (GAN):

For a number of training iterations do:

Sample minibatch of noise samples $\epsilon_1, \dots, \epsilon_M \sim p(\epsilon)$.
Sample minibatch of real (image) samples $x_1, \dots, x_M \sim p(x)$.

Sample from Uniform(0,1)

Sample from empirical distribution
(bag of objects)

Update critic by stochastic gradient ascent

$$\nabla_\psi \frac{1}{M} \sum_{m=1}^{M} [\log D(x_m; \psi) + \log(1 - D(f(\epsilon_m; \theta); \psi))]$$

Update generator by stochastic gradient descent

$$\nabla_\theta \frac{1}{M} \sum_{m=1}^{M} \log(1 - D(f(\epsilon_m; \theta); \psi))$$

Learning adversarially revisited

Algorithm (GAN):

For a number of training iterations do:

Sample minibatch of noise samples $\epsilon_1, \dots, \epsilon_M \sim p(\epsilon)$.
Sample minibatch of real (image) samples $x_1, \dots, x_M \sim p(x)$.

Sample from Uniform(0,1)

Sample from empirical distribution (bag of objects)

Update critic by stochastic gradient ascent

Expectation replaced by averages

$$\nabla_\psi \boxed{\frac{1}{M} \sum_{m=1}^{M}} [\log D(x_m; \psi) + \log(1 - D(f(\epsilon_m; \theta); \psi))]$$

Update generator by stochastic gradient descent

$$\nabla_\theta \frac{1}{M} \sum_{m=1}^{M} \log(1 - D(f(\epsilon_m; \theta); \psi))$$

Learning adversarially revisited

Algorithm (GAN):

For a number of training iterations do:

Sample from Uniform(0,1)

Sample from empirical distribution (bag of objects)

Sample minibatch of noise samples $\epsilon_1, \ldots, \epsilon_M \sim p(\epsilon)$.
Sample minibatch of real (image) samples $x_1, \ldots, x_M \sim p(x)$.

Update critic by stochastic gradient ascent

Expectation replaced by averages

$$\nabla_\psi \frac{1}{M} \boxed{\sum_{m=1}^{M}} [\log D(x_m; \psi) + \log(1 - D(f(\epsilon_m; \theta); \psi))]$$

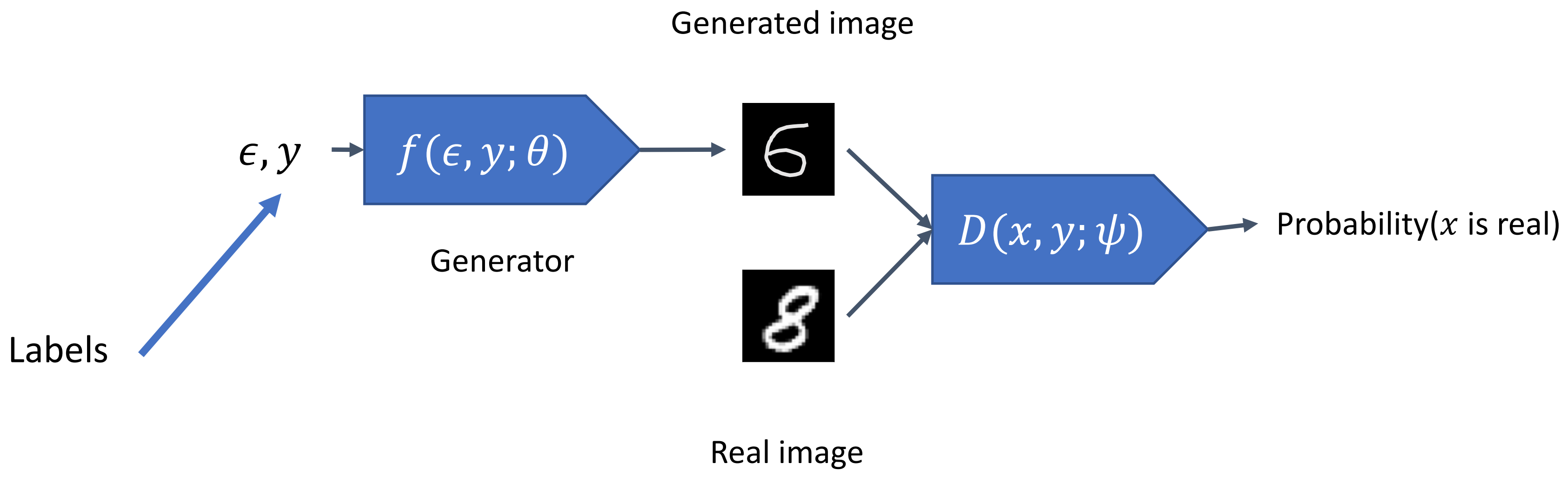
Fixed generator parameters θ in critic update

Update generator by stochastic gradient descent

Fixed critic parameters ψ in generator update

$$\nabla_\theta \frac{1}{M} \sum_{m=1}^{M} \log(1 - D(f(\epsilon_m; \theta); \psi))$$

Conditional adversarial learning

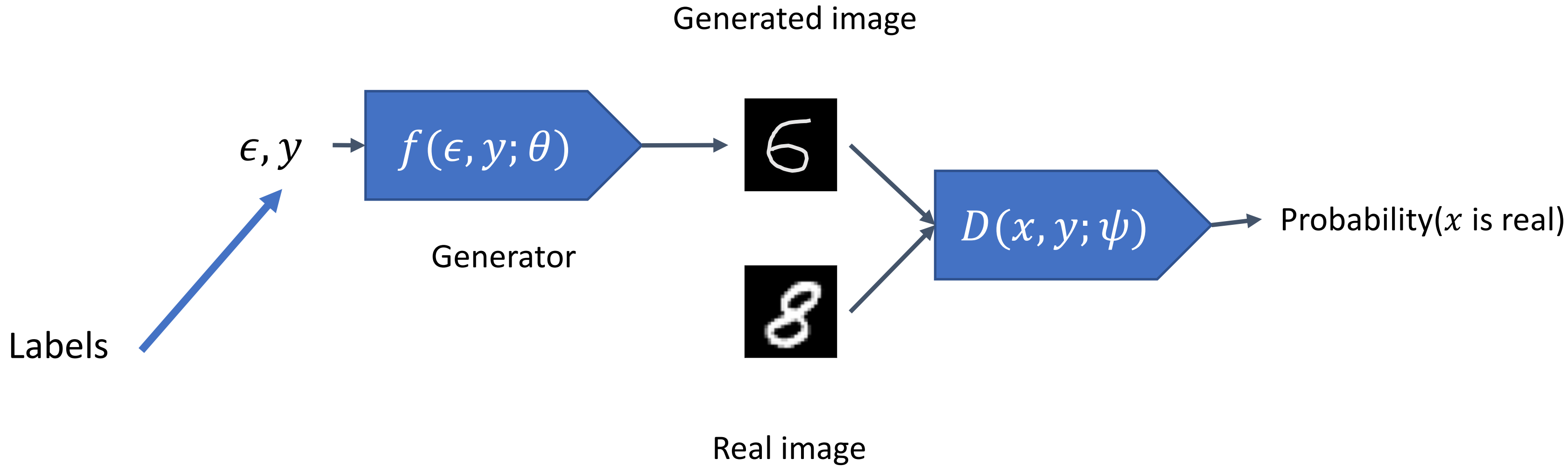


- We learn θ by minimizing the critic's ability to identify samples from the generator as not real.
- We learn ψ by maximizing the chances for the critic to correctly identify real samples.

Formally, the objective $V(\theta, \psi)$ is written as:

$$\text{argmin}_{\theta}\ \text{argmax}_{\psi} \quad V(\theta, \psi) = \mathbb{E}_{p(x)}[\log D(x, y; \psi)] + \mathbb{E}_{p(\epsilon)}[\log(1 - D(f(\epsilon, y; \theta), y; \psi)]$$

Conditional adversarial learning



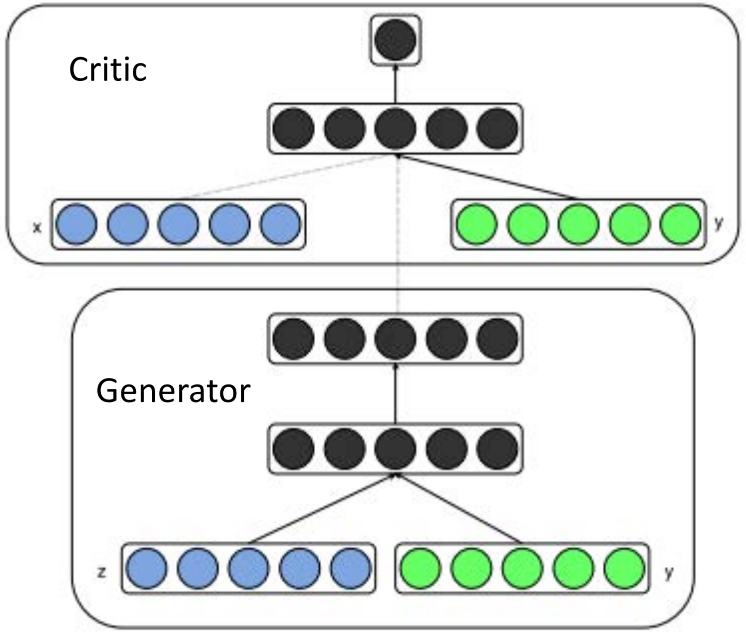
- We learn θ by minimizing the critic's ability to identify samples from the generator as not real.
- We learn ψ by maximizing the chances for the critic to correctly identify real samples.

Formally, the objective $V(\theta, \psi)$ is written as:

$$\operatorname{argmin}_\theta \operatorname{argmax}_\psi \quad V(\theta, \psi) = \mathbb{E}_{p(x)}[\log D(x, y; \psi)] + \mathbb{E}_{p(\epsilon)}[\log(1 - D(f(\epsilon, y; \theta), y; \psi)]$$

Labels

Conditional adversarial learning

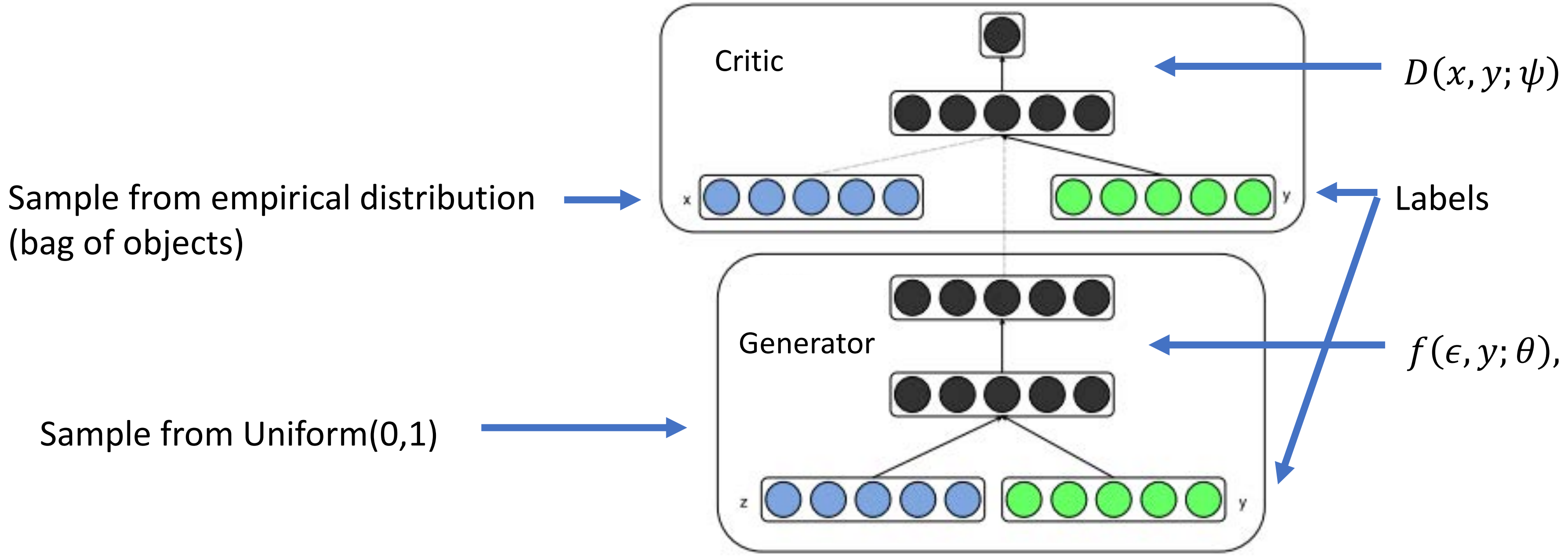


Formally, the objective $V(\theta, \psi)$ is written as:

$$\operatorname{argmin}_\theta \operatorname{argmax}_\psi \quad V(\theta, \psi) = \mathbb{E}_{p(x)}[\log D(x, y; \psi)] + \mathbb{E}_{p(\epsilon)}[\log(1 - D(f(\epsilon, y; \theta), y; \psi)]$$

Note: the conditioning variable y can be a label, but it could also contain side covariates, attributes, captions, *etc.*

Conditional adversarial learning

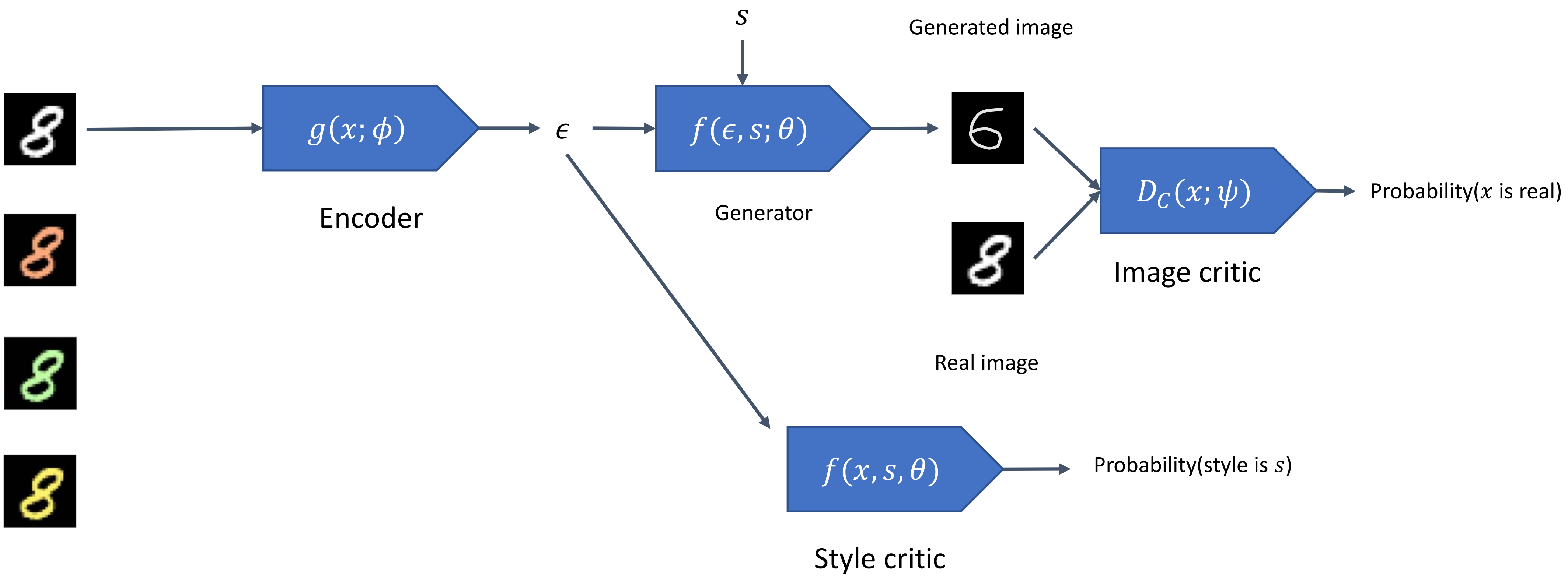


Formally, the objective $V(\theta, \psi)$ is written as:

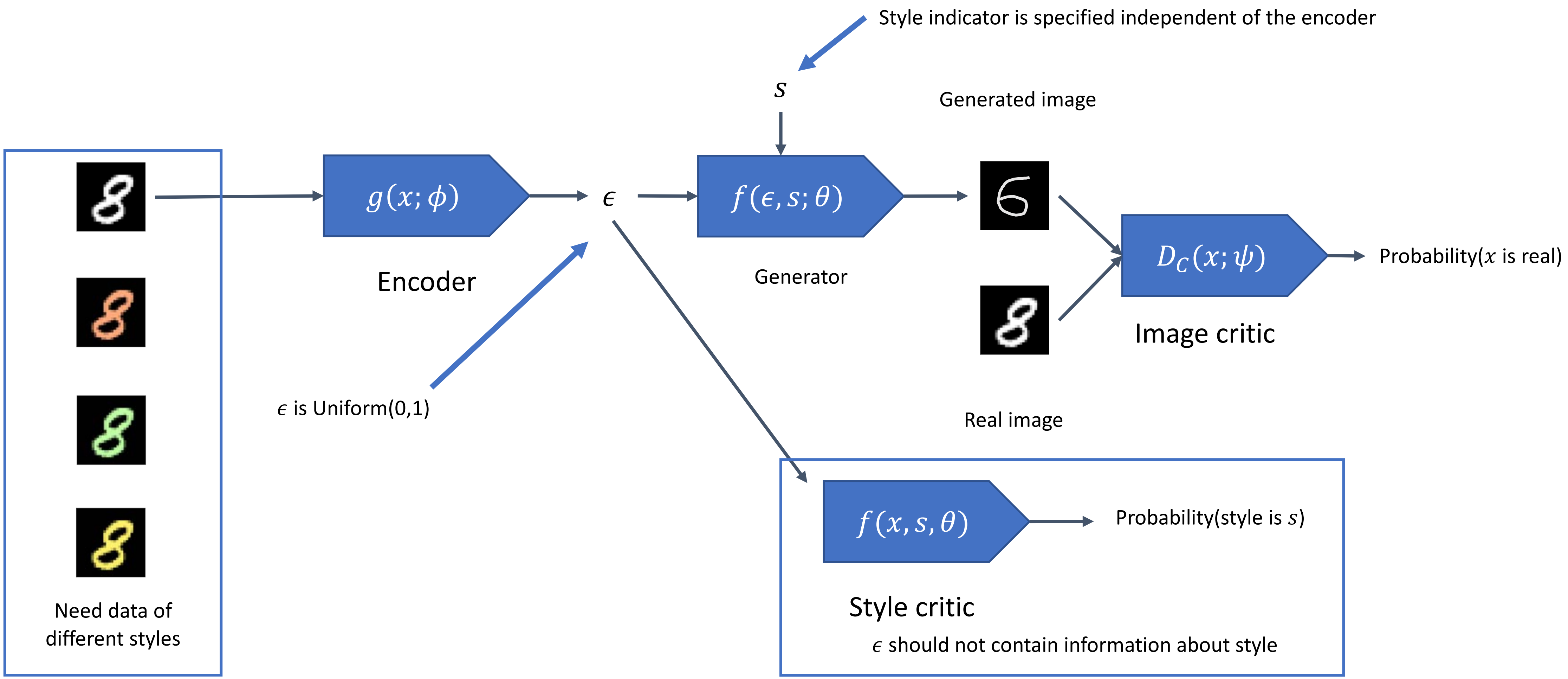
$$\text{argmin}_\theta \; \text{argmax}_\psi \quad V(\theta, \psi) = \mathbb{E}_{p(x)}[\log D(x, y; \psi)] + \mathbb{E}_{p(\epsilon)}[\log(1 - D(f(\epsilon, y; \theta), y; \psi)]$$

Note: the conditioning variable y can be a label, but it could also contain side covariates, attributes, captions, *etc.*

Adversarial disentanglement



Adversarial disentanglement



Background Papers

- Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. Generative adversarial nets. In Advances in Neural Information Processing Systems (NIPS) 2014. http://papers.nips.cc/paper/5423-generative-adversarial-nets
- Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks. In International Conference on Learning Representations (ICLR) 2016. https://arxiv.org/abs/1511.06434

Application Papers

- Karras T, Aila T, Laine S, Lehtinen J. Progressive growing of GANs for improved quality, stability, and variation. In International Conference on Learning Representations (ICLR) 2018. https://arxiv.org/abs/1710.10196. Source: https://github.com/tkarras/progressive_growing_of_gans
- Elgammal A, Liu B, Elhoseiny M, Mazzone M. CAN: Creative Adversarial Networks, Generating "Art" by Learning About Styles and Deviating from Style Norms. In International Conference on Computational Creativity (ICCC) 2017. https://arxiv.org/abs/1706.07068. Source: https://github.com/mlberkeley/Creative-Adversarial-Networks
- Jin Y, Zhang J, Li M, Tian Y, Zhu H, Fang Z. Towards the Automatic Anime Characters Creation with Generative Adversarial Networks. In Comiket 92 2017. https://arxiv.org/abs/1708.05509. Source: https://github.com/ctwxdd/Tensorflow-ACGAN-Anime-Generation. Demo: https://make.girls.moe/#/