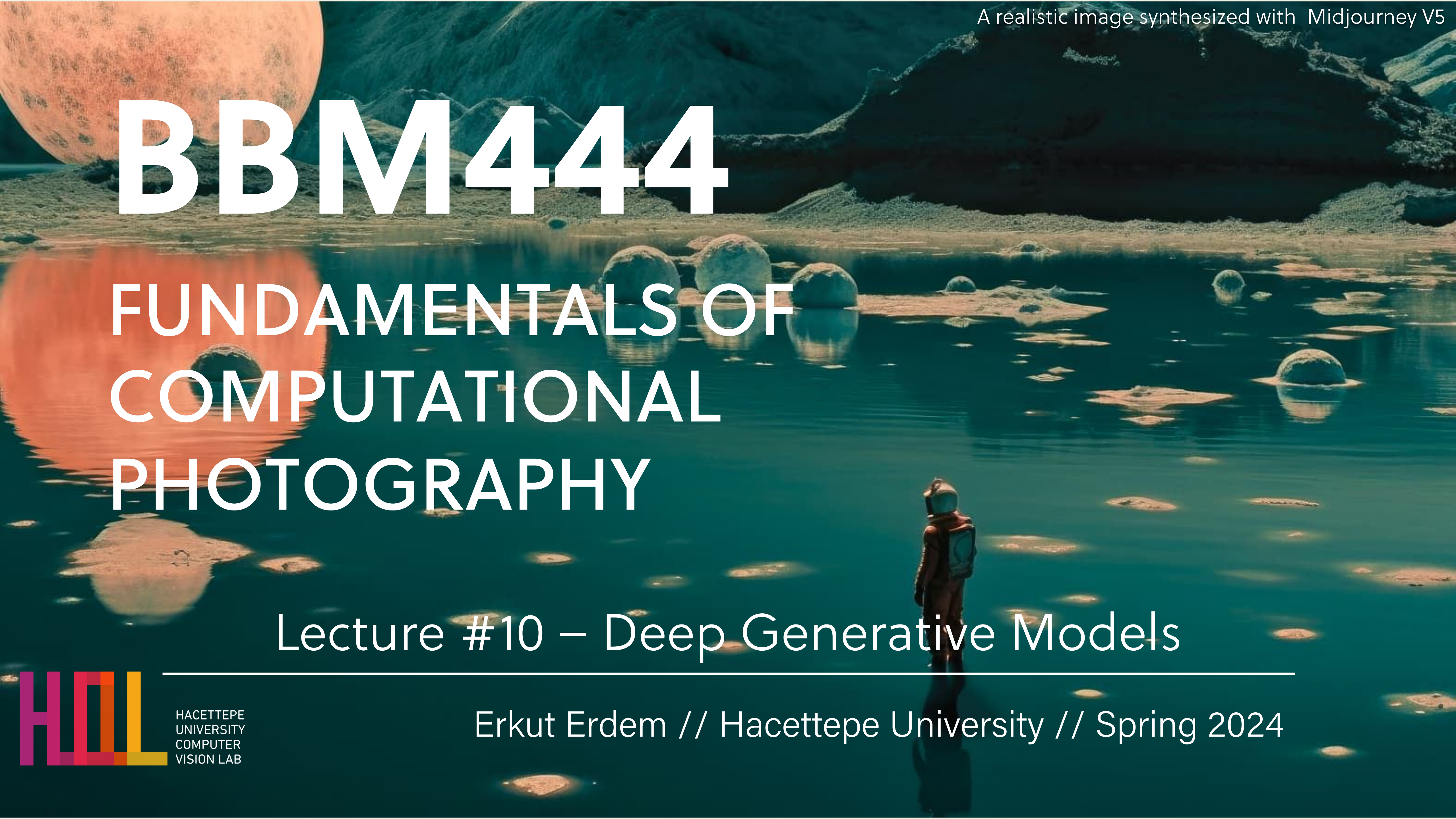


A realistic image synthesized with Midjourney V5

BBM444

FUNDAMENTALS OF COMPUTATIONAL PHOTOGRAPHY

Lecture #10 – Deep Generative Models

HACETTEPE UNIVERSITY COMPUTER VISION LAB

Erkut Erdem // Hacettepe University // Spring 2024

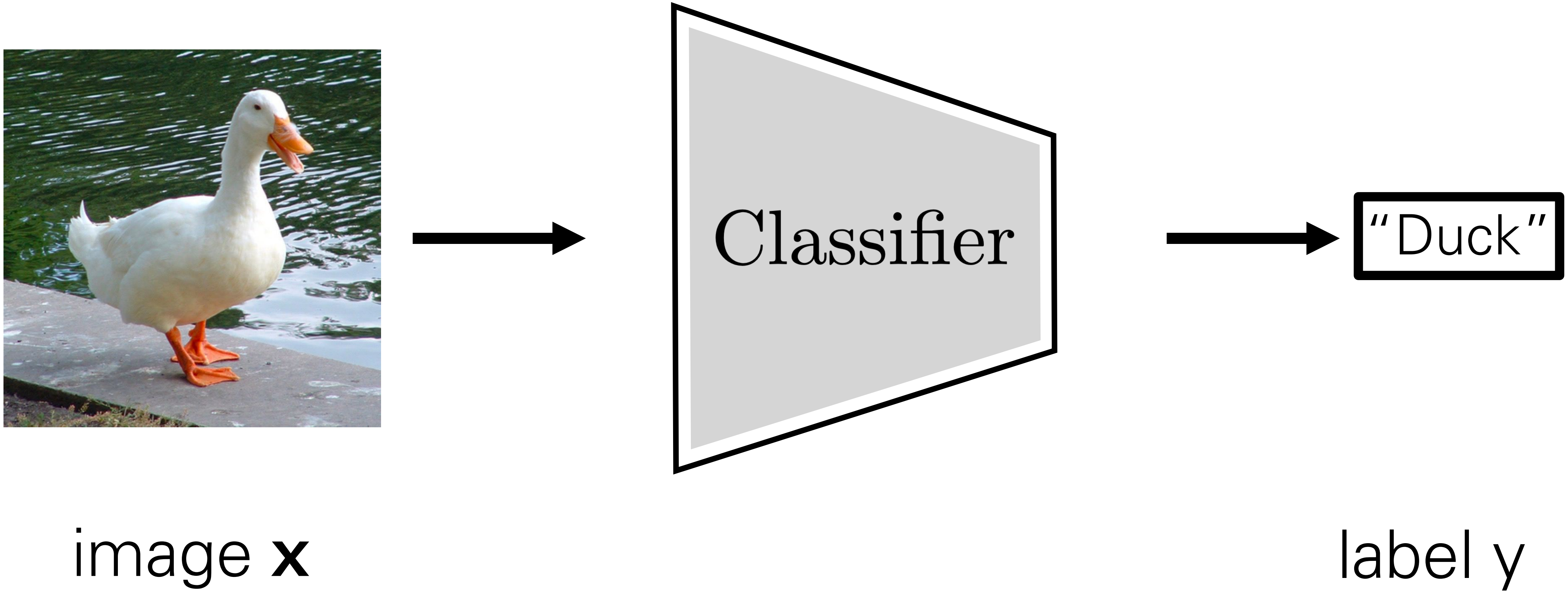
Today’s Lecture

- Discriminative vs. generative models
- Image synthesis
- Representation learning
- Data translation
- Applications in computational photography

Disclaimer: The material and slides for this lecture were borrowed from
— Bill Freeman, Antonio Torralba and Phillip Isola’s MIT 6.869 class
— Philip Isola and Stefanie Jegelka’s MIT 6.S898 class

Discriminative vs. Generative Models

Analysis



Analysis

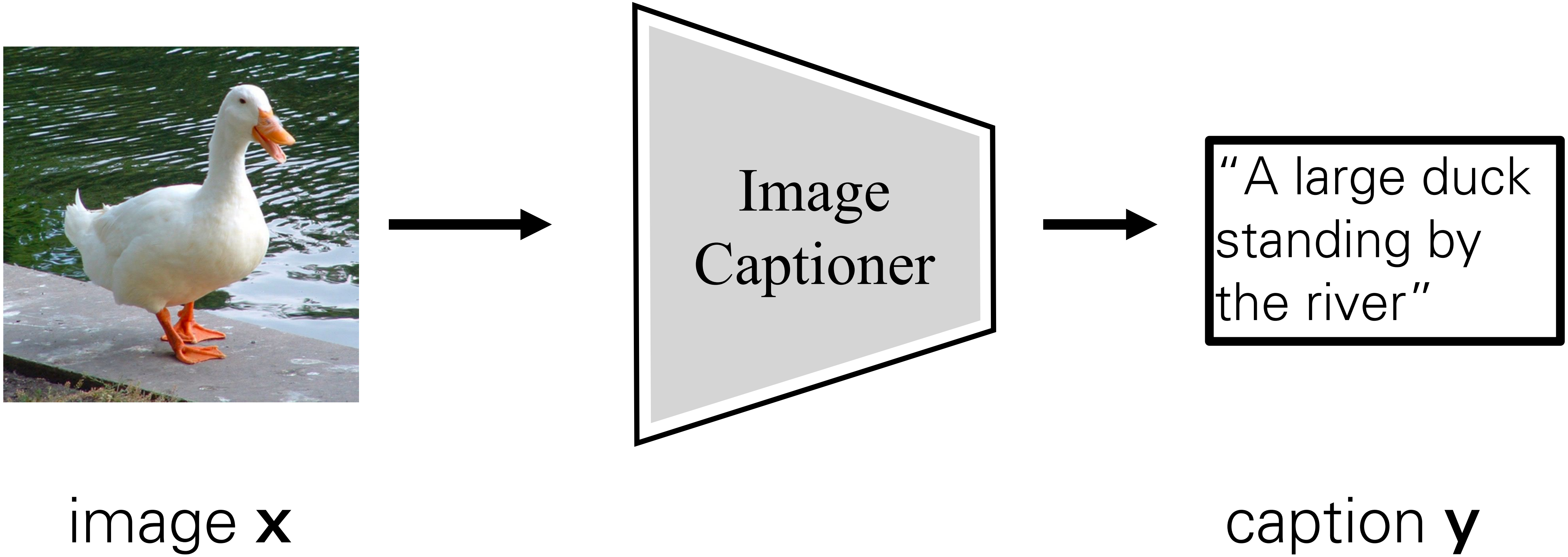
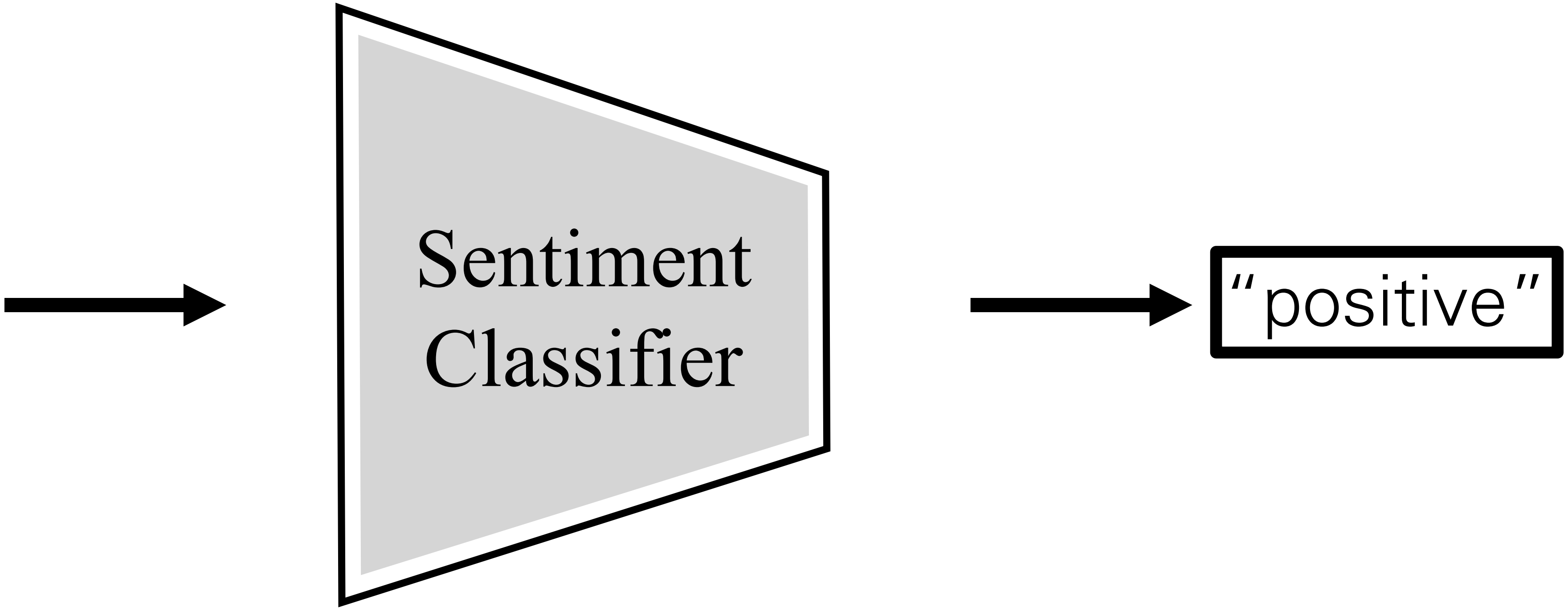


Image Captioner
"A large duck standing by the river"
image **x**
caption **y**

Analysis

“A statuesque duck gazing gracefully over the water”



sentence **x**

sentiment y

Analysis

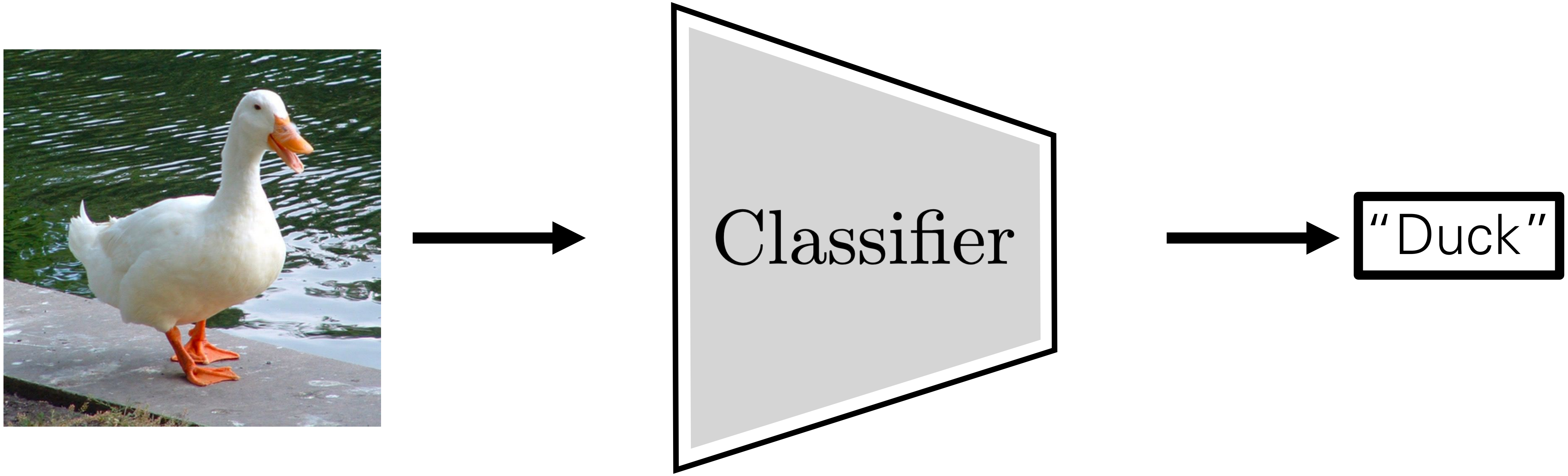
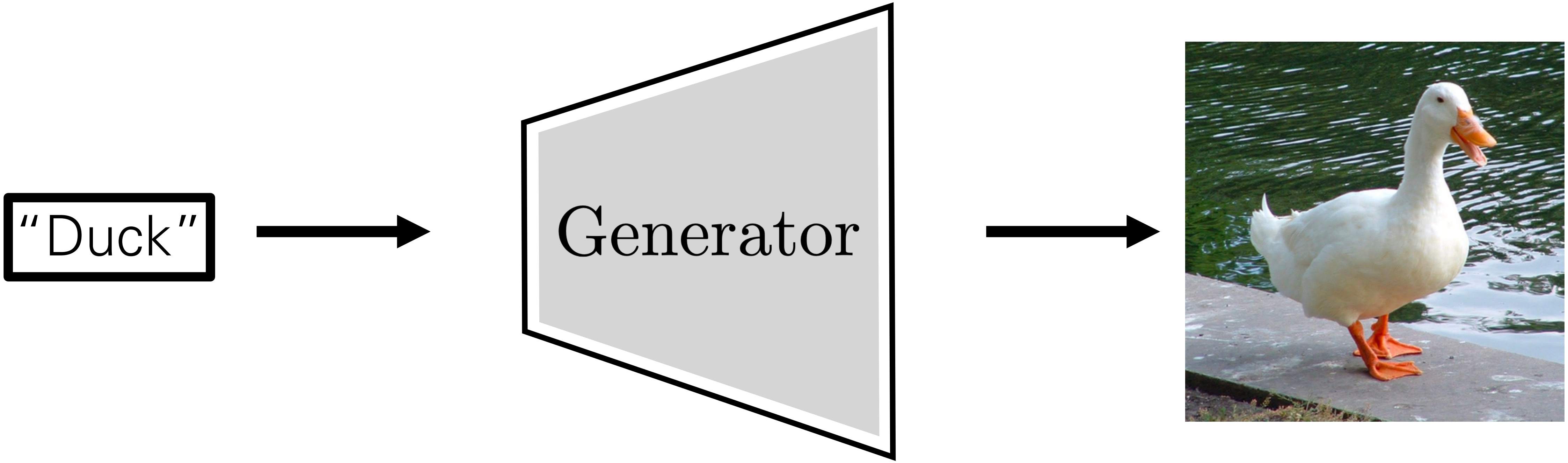


image **x**

label y

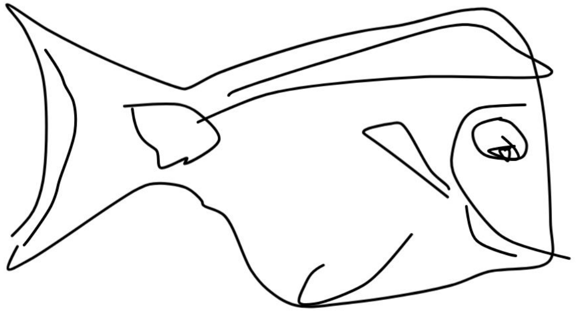
Synthesis

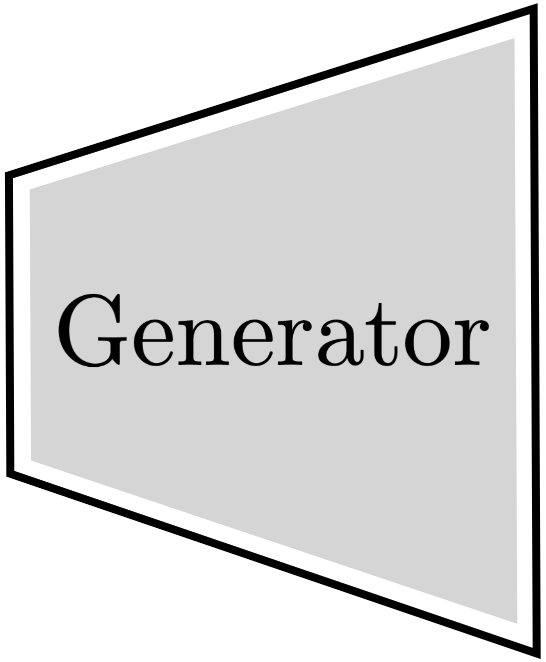


label y

image **x**

Synthesis





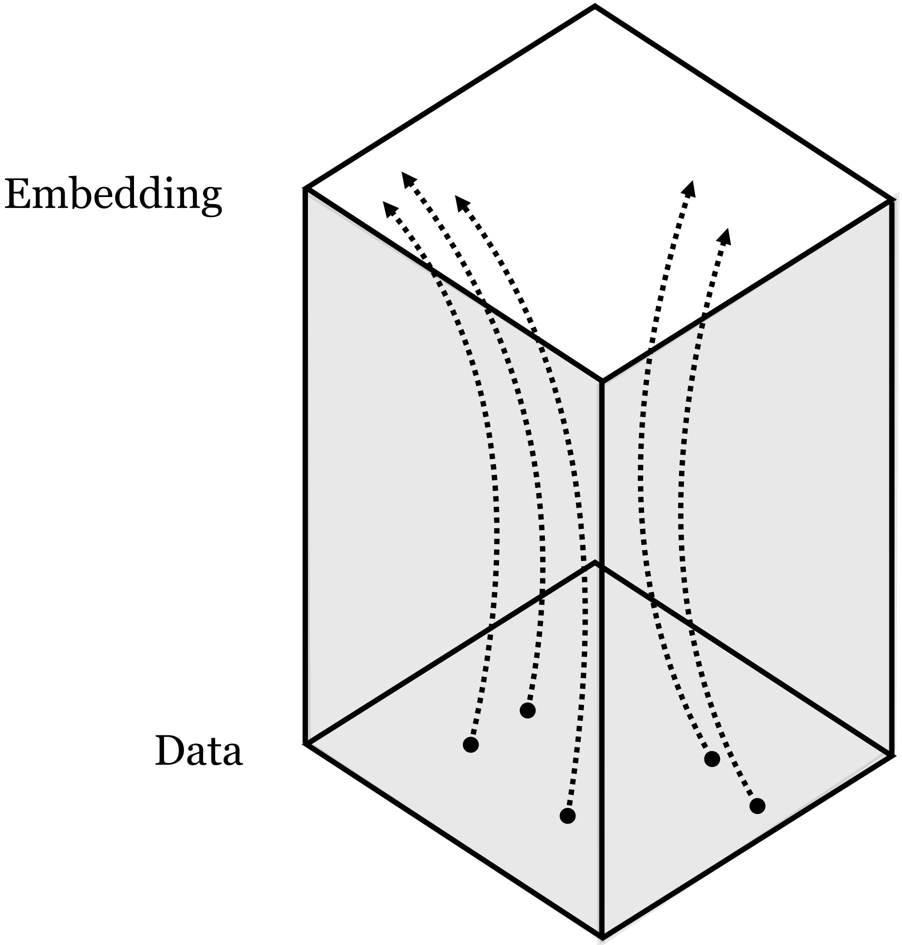
User sketch

Photo

Deep nets are data transformers

- Deep nets transform datapoints, layer by layer

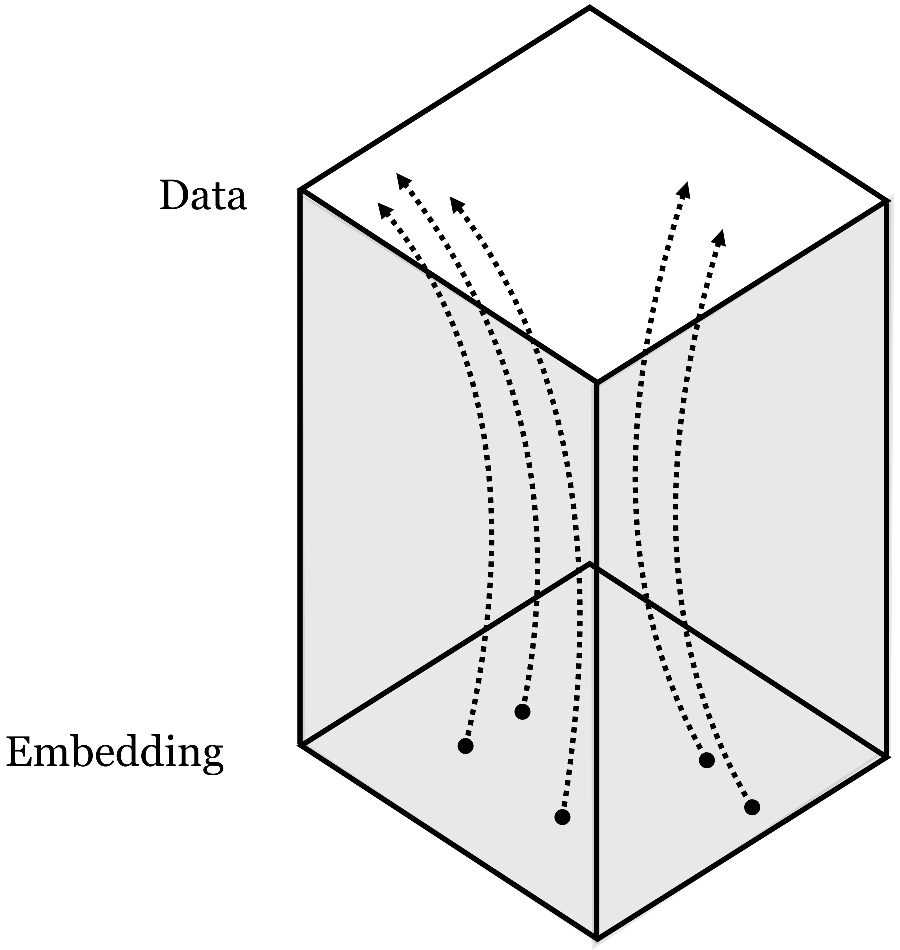
- Each layer is a different representation of the data



Deep nets are data transformers

- Deep nets transform datapoints, layer by layer

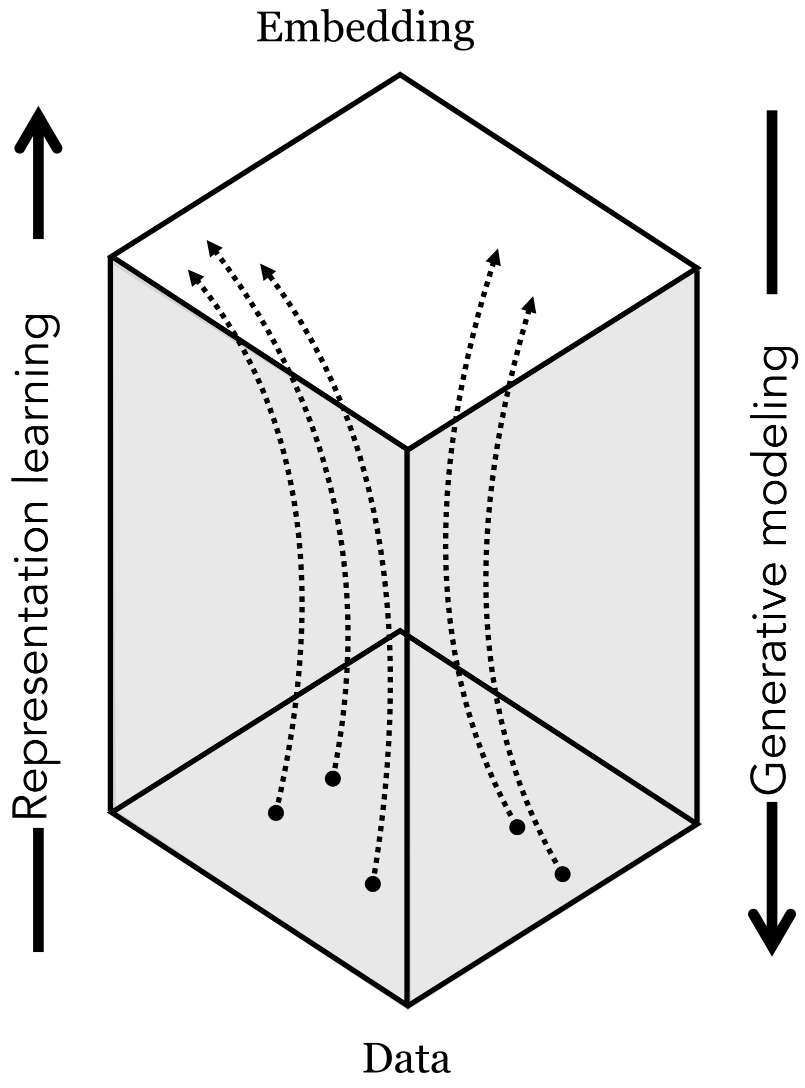
- Each layer is a different representation of the data



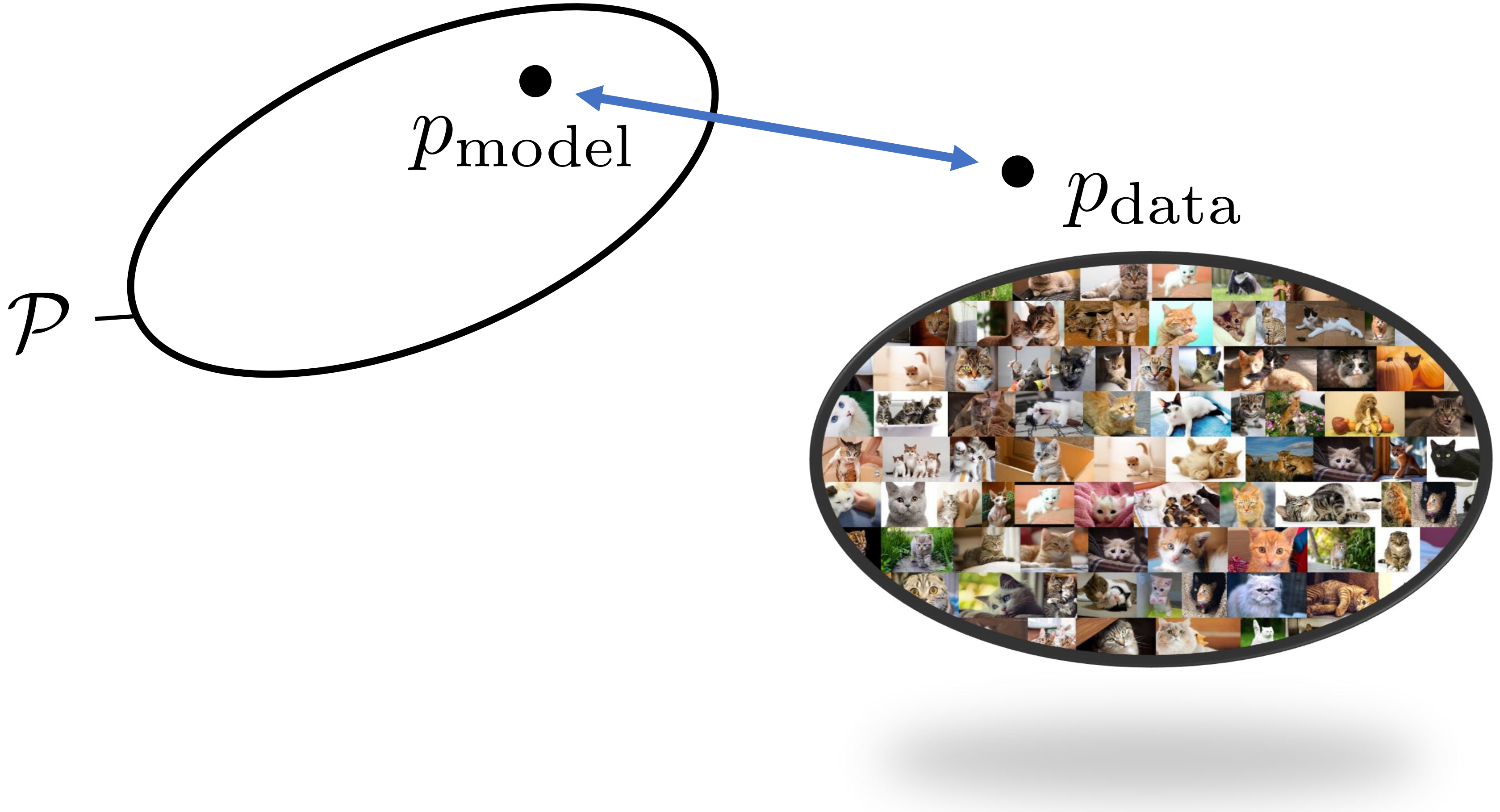
Generative modeling vs Representation learning

Representation learning:
mapping data to abstract representations (analysis)

Generative modeling:
mapping abstract representations to data (synthesis)

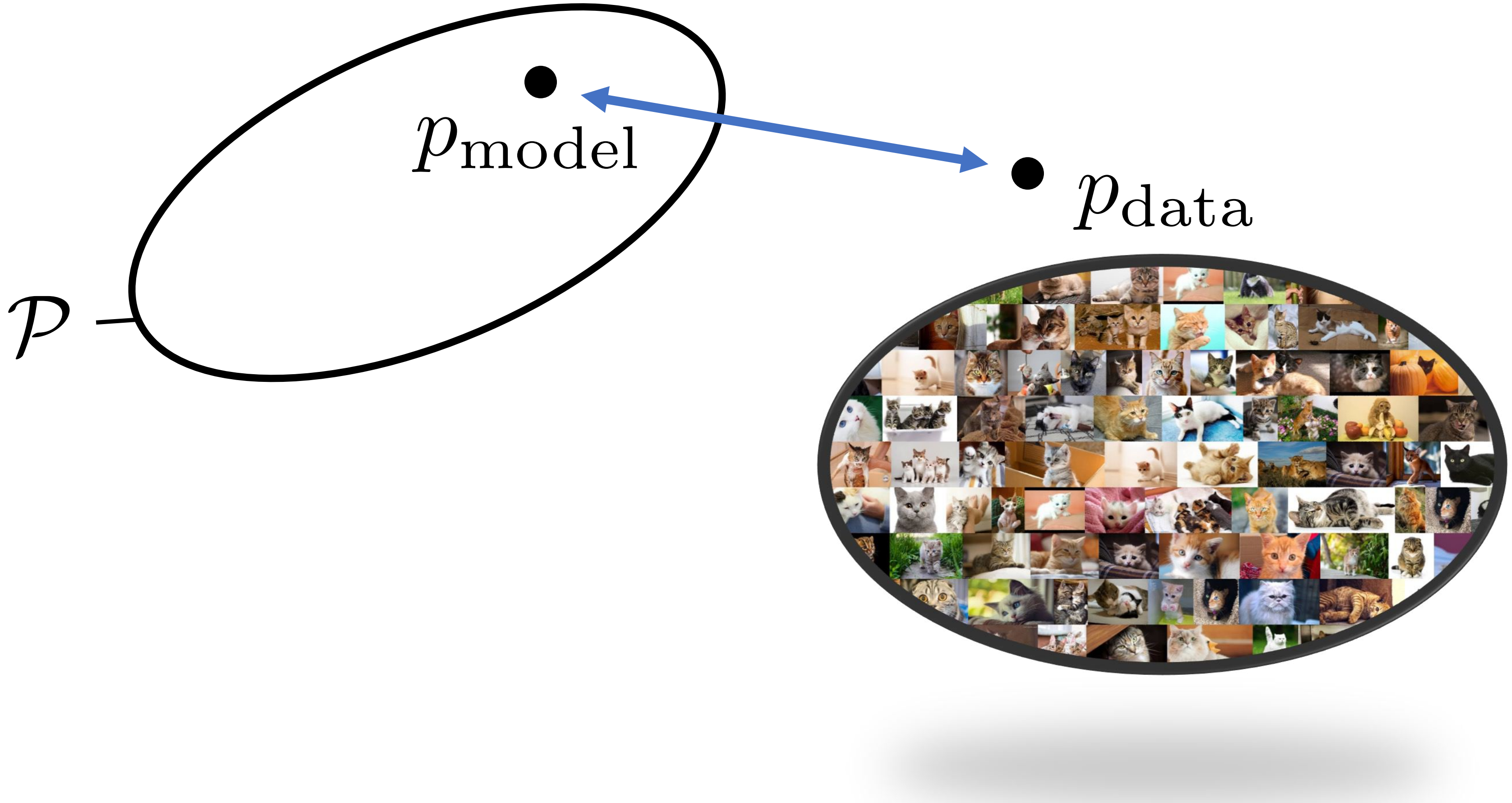


Generative Modeling



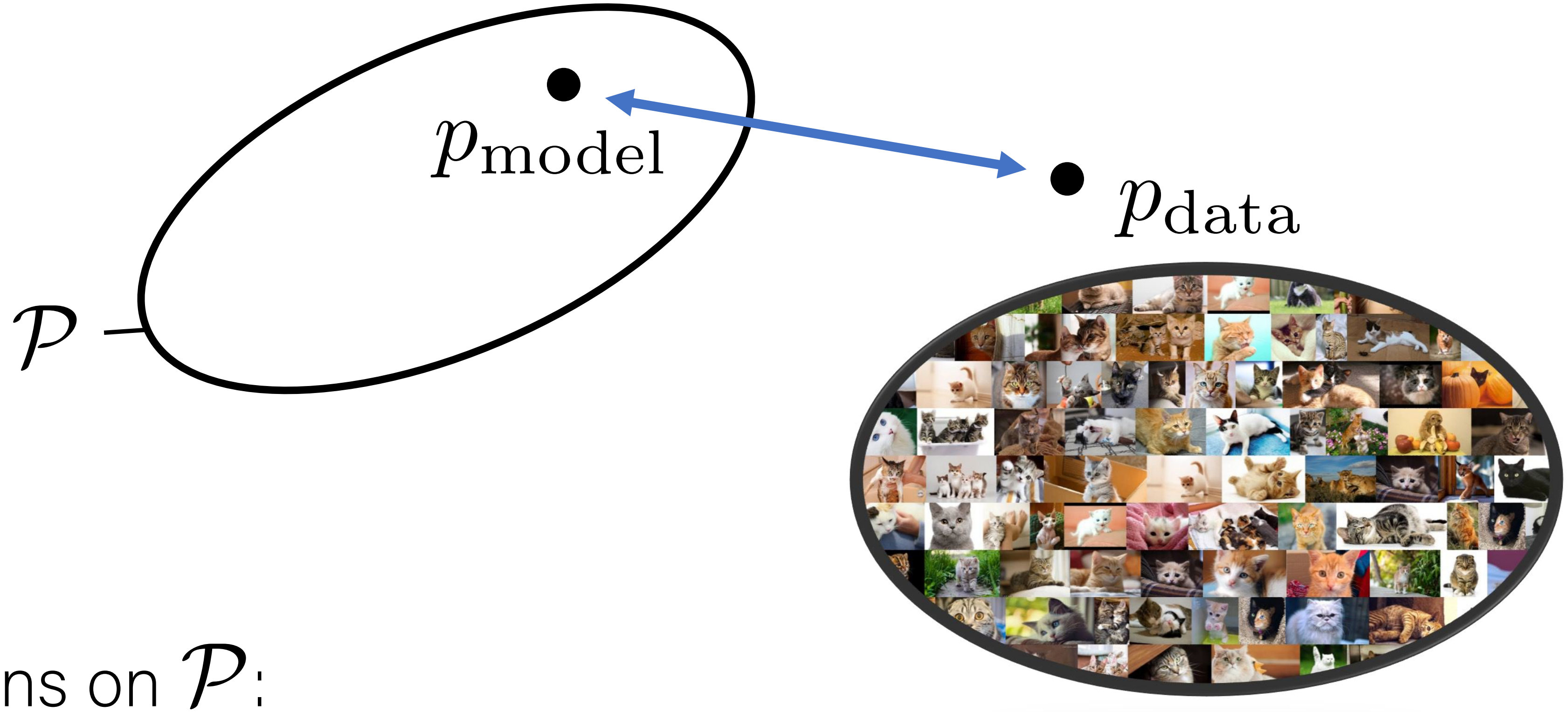
Slide adapted from Sebastian Nowozin

Generative Modeling



- **Goal:** Learn some underlying hidden structure of the training samples to generate novel samples from same data distribution

Generative Modeling



Assumptions on $\mathcal{P}$:

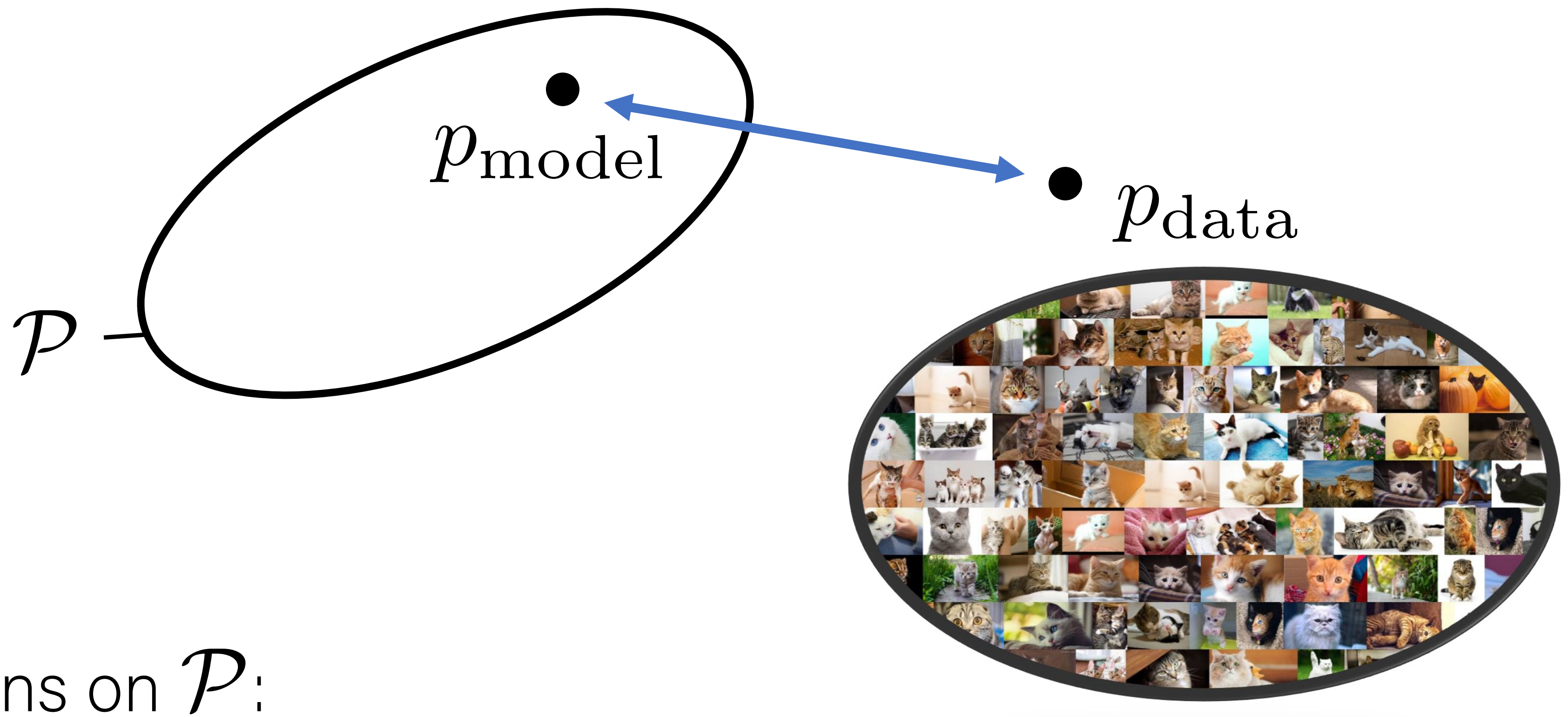
- tractable sampling

Training examples

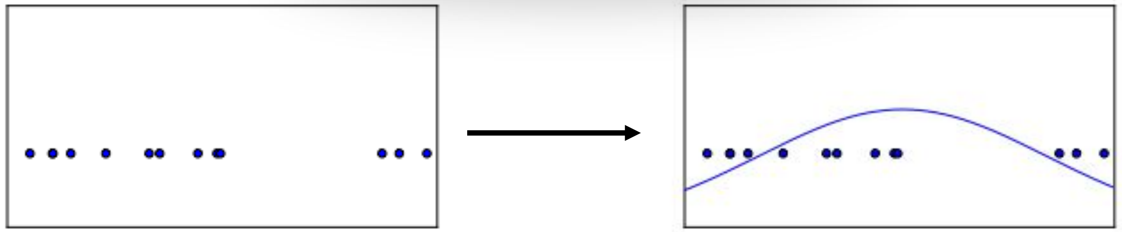
Model samples

Slide adapted from Sebastian Nowozin

Generative Modeling



Assumptions on $\mathcal{P}$:
- tractable sampling
- tractable likelihood function



Slide adapted from Sebastian Nowozin

[Images: https://ganbreeder.app/]

Image synthesis

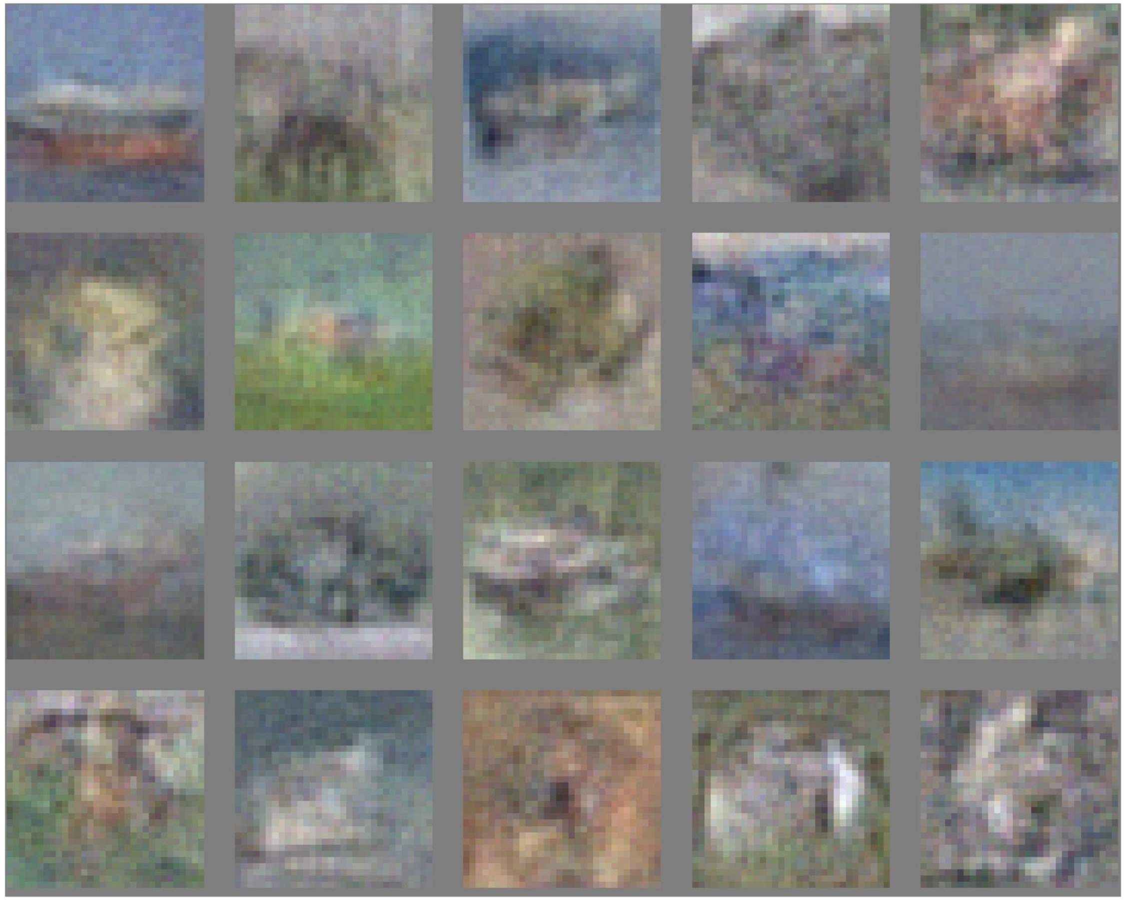
Generate Images

[Deep Belief Nets, Hinton, Osindero, Teh, 2006]

Generate Images

[VAE, Kingma and Welling, 2013]

Generate Images



[GAN, Goodfellow et al. 2014]

Generate Images

[DCGAN, Radford, Metz, Chintala 2015]

Generate Images

[DCGAN, Radford, Metz, Chintala 2015]

Generate Images

[CycleGAN: Zhu, Park, Isola & Efros, 2017]

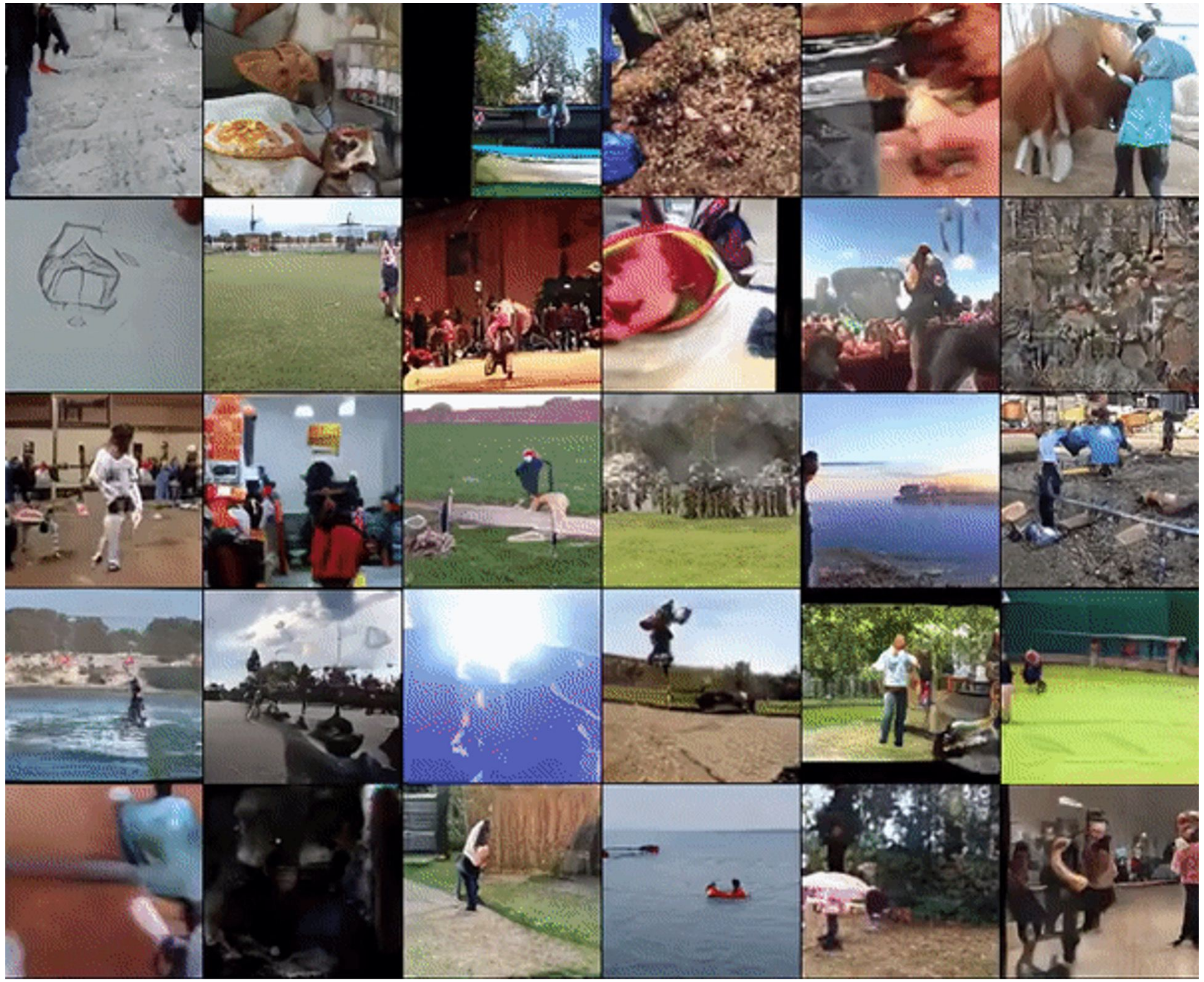
Generate Images

[BigGAN, Brock, Donahue, Simonyan, 2018]

Generate Images

[StyleGAN, Karras, Laine, Aila, 2018]

Generate Video



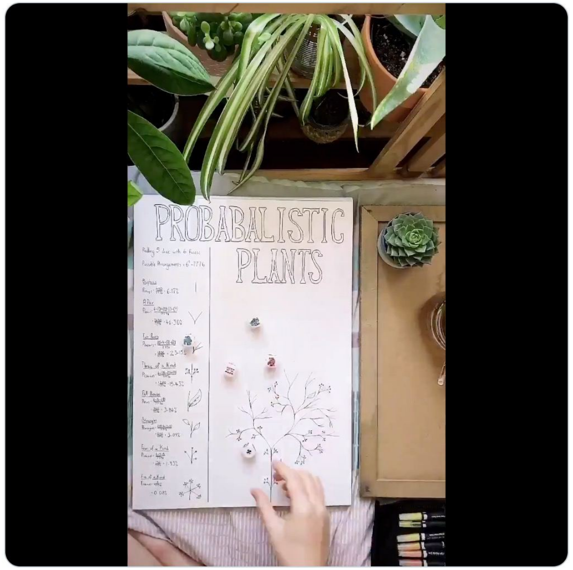
DVD-GAN: Adversarial Video Generation on Complex Datasets, Clark, Donahue, Simonyan, 2019

Procedural graphics

[Anders Scheil]

Ayliean @Ayliean · Nov 17

Made up a set of rules and rolled some dice to decide how this plant would grow. I never did get that five of a kind, as expected, but I was still hopeful! 🌱🍀🌿



52 · 1.1K · 4.5K

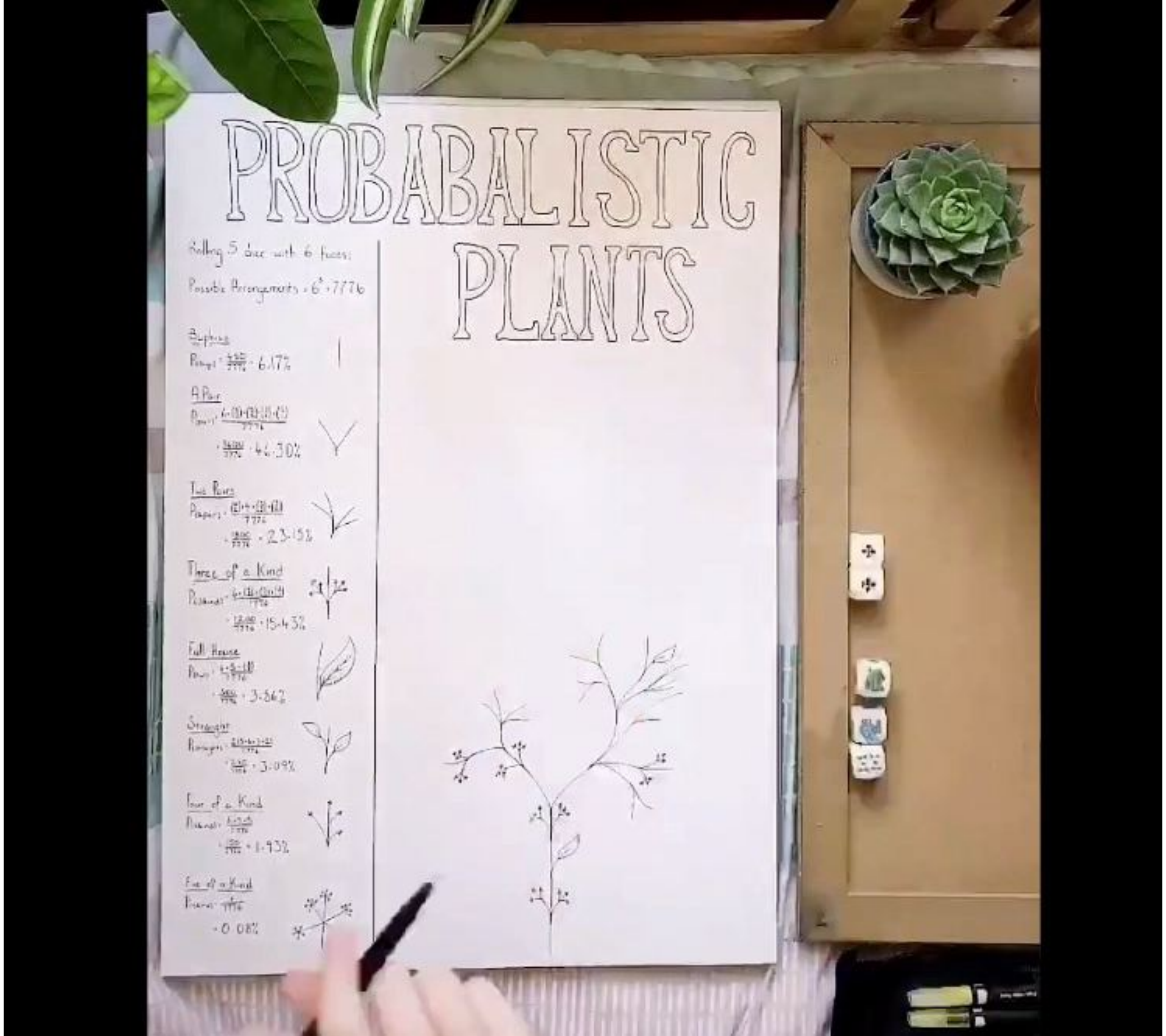
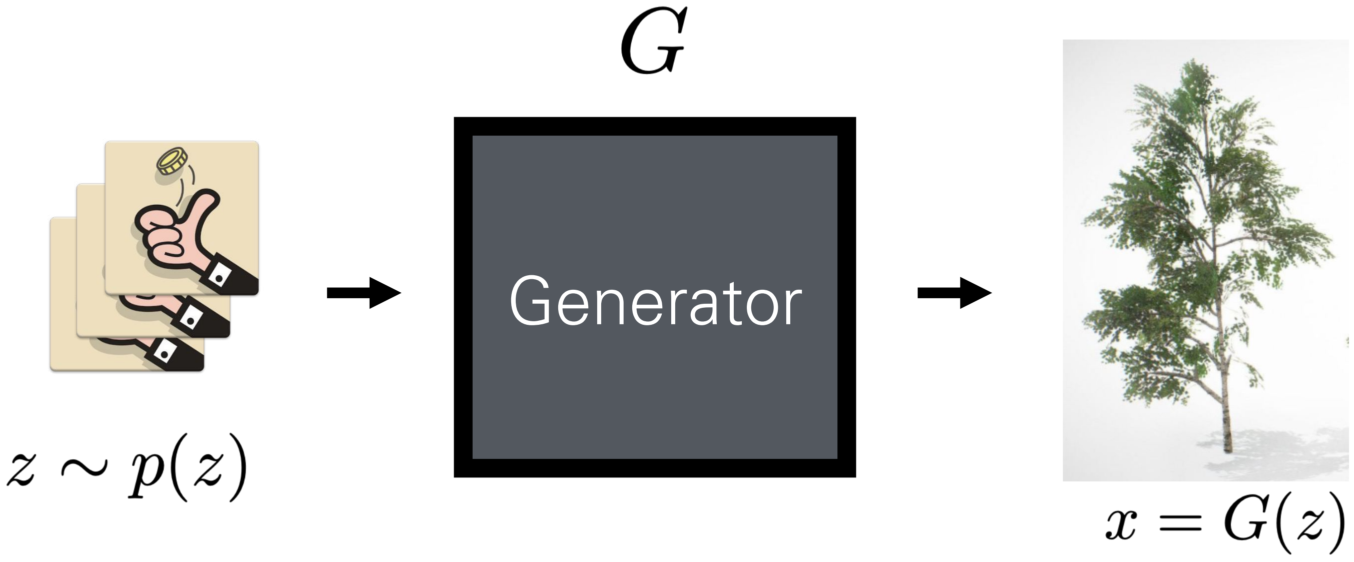


Image synthesis from "noise"



Sampler

$$G : \mathcal{Z} \to \mathcal{X}$$

$$z \sim p(z)$$

$$x = G(z)$$

$\mathbf{z} \sim \texttt{Bernoulli}(0.5)$

for $i = 1, \ldots, N$ **do**

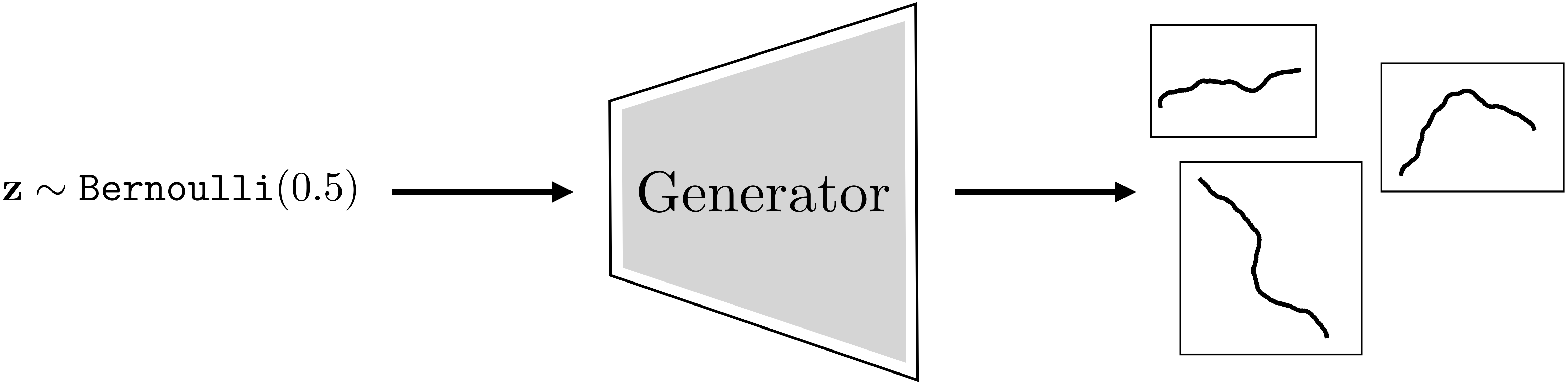
  `extend line 1 unit in current heading direction`

  if $z_i == 1$ **then**

    `rotate heading` 10° `to the right`

  else

    `rotate heading` 10° `to the left`



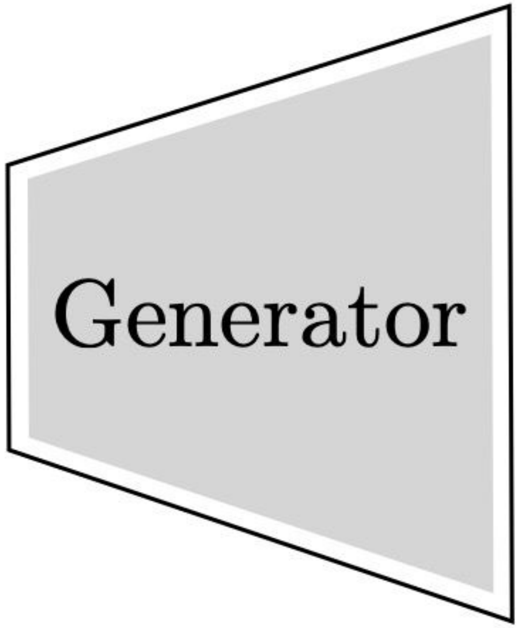
"noise" "latent variables"

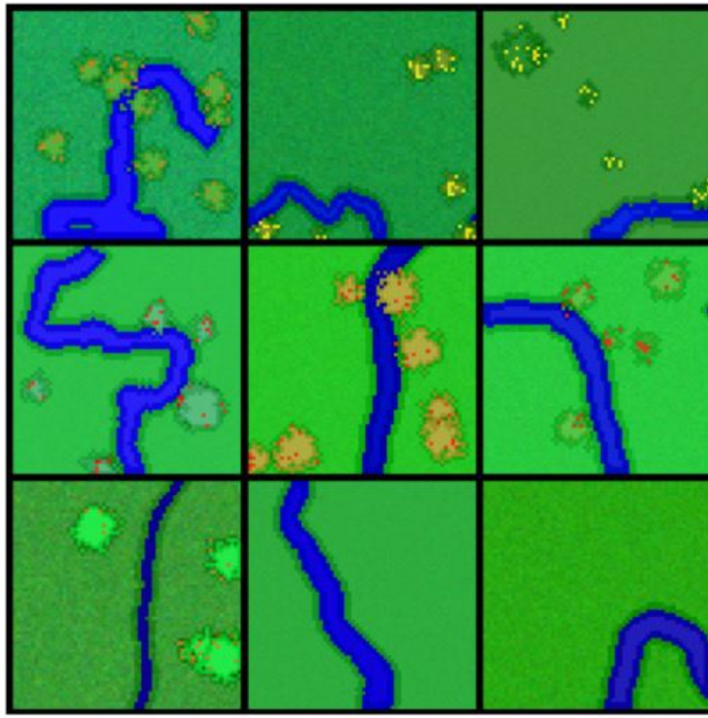
$\mathbf{z}_1 \sim \texttt{Bernoulli}(0.5)$ (River turns)

$\mathbf{z}_2 \sim \texttt{Normal}(\mu_1, \mathbf{\Sigma}_1)$ (Grass color)

$\mathbf{z}_3 \sim \texttt{Unif}(0, 10)$ (Number trees)

$\vdots$





Concept #1: **noise is latent variables**

What's the goal of generative modeling?

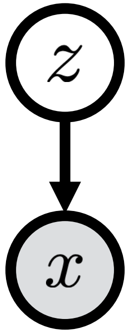
- Make synthetic data that "looks like" real data.
- How to measure "looks like"?
- The main answer in deep generative models is: "has high probability under a density model fit to real data."

What's the goal of generative modeling?

The goal is not to replicate the training data but to make **new** data that is **realistic** (captures the essential properties of real data)

(A model that memorizes the training data is overfit in exactly the same sense as a classifier can be overfit)

Learning a generative model



latent variables

Input samples

$\rightarrow$

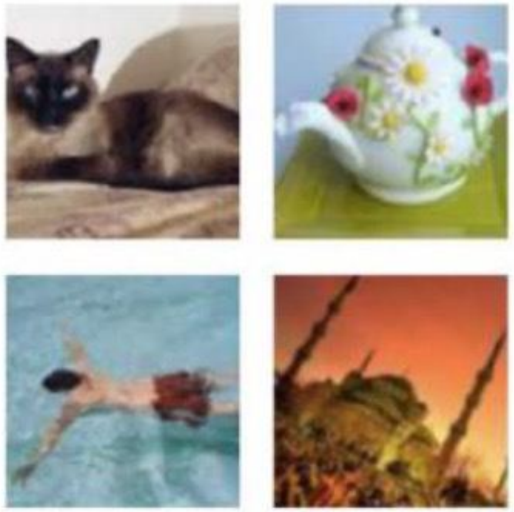
Learner

Objective

Hypothesis space

Optimizer

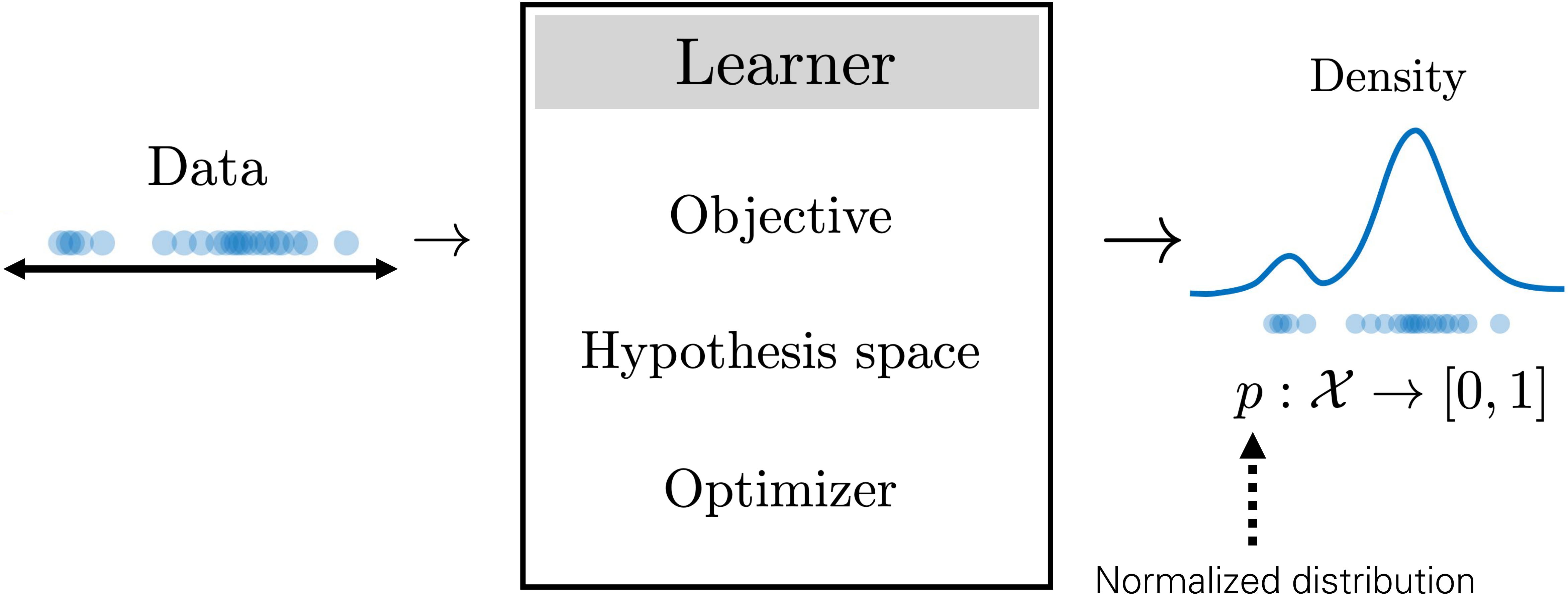
$\rightarrow$ z $\rightarrow$



Generated samples

[figs modified from: http://introtodeeplearning.com/materials/2019_6S191_L4.pdf]

Learning a density model



$$p : \mathcal{X} \to [0, 1]$$

Normalized distribution
(some models output
unormalized *energy functions*)

[figs modified from: http://introtodeeplearning.com/materials/2019_6S191_L4.pdf]

Case study #1: Fitting a Gaussian to data

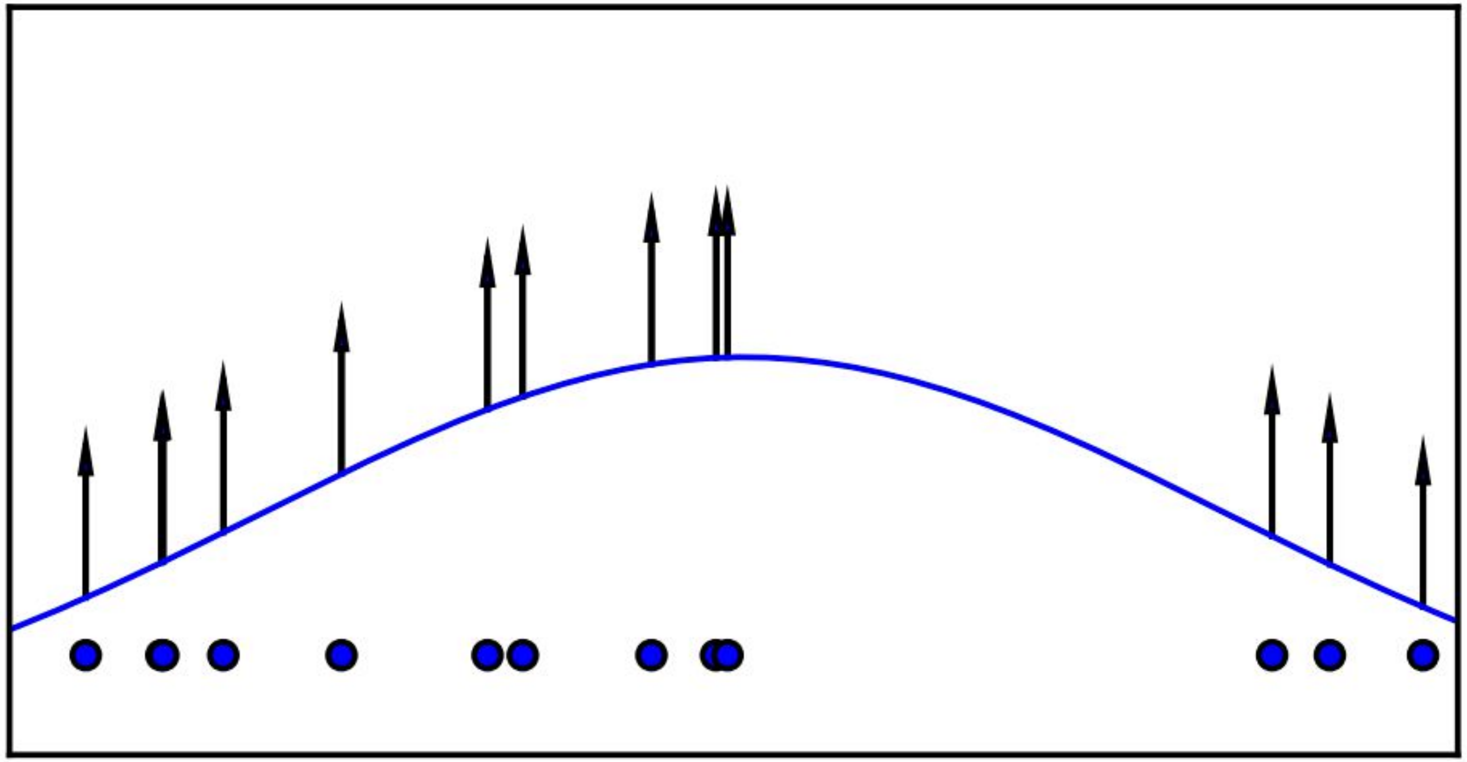


fig from [Goodfellow, 2016]

Max likelihood objective

$$\max_{\theta} \mathbb{E}_{x \sim p_{\texttt{data}}}[\log p_\theta(x)]$$

Considering only Gaussian fits

$$p_\theta(x) = \mathcal{N}(x; \mu, \sigma)$$
$$\theta = [\mu, \sigma]$$

Closed form optimum:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x^{(i)} \quad \sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x^{(i)} - \mu)^2$$

Case study #1: Fitting a Gaussian to data

"max likelihood"

Data $\rightarrow$

Learner

Objective

$$\max_{\theta} \mathbb{E}_{x \sim p_{\texttt{data}}}[\log p_\theta(x)]$$

Hypothesis space

$$p(x) = \mathcal{N}(x; \mu, \sigma)$$

Optimizer

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x^{(i)} \quad \sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x^{(i)} - \mu)^2$$

$\rightarrow$

Density

$$p : \mathcal{X} \rightarrow [0, 1]$$

Case study #2: learning a deep generative model

Data $\rightarrow$

Learner

Objective
Usually max likelihood

Hypothesis space
Deep net

Optimizer
SGD

$\rightarrow$ Density $p : \mathcal{X} \rightarrow [0, 1]$

Case study #2: learning a deep generative model

$\text{Data} \quad \rightarrow$

Learner

Objective
Usually max likelihood

Hypothesis space
Deep net

Optimizer
SGD

$\rightarrow$

Density
$p : \mathcal{X} \rightarrow [0, 1]$

Sampler
$G : \mathcal{Z} \rightarrow \mathcal{X}$
$z \sim p(z)$
$x = G(z)$

Models that provide a sampler but no density are called **implicit generative models**

Learning data generator

Two approaches:

confusingly, sometimes called an "implicit generative model"

1. **Direct approach:** learn a function that generates data directly $G : \mathcal{Z} \rightarrow \mathcal{X}$

2. **Indirect approach:** learn a function that scores data; generate data by finding points that score highly under this function $E : \mathcal{X} \rightarrow \mathbb{R}$

Direct approach

Training

Data

$\{\mathbf{x}^{(i)}\}_{i=1}^{N}$

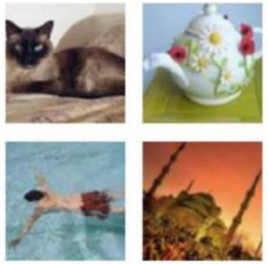
$\rightarrow$ Learner $\rightarrow$ Generator

Sampling

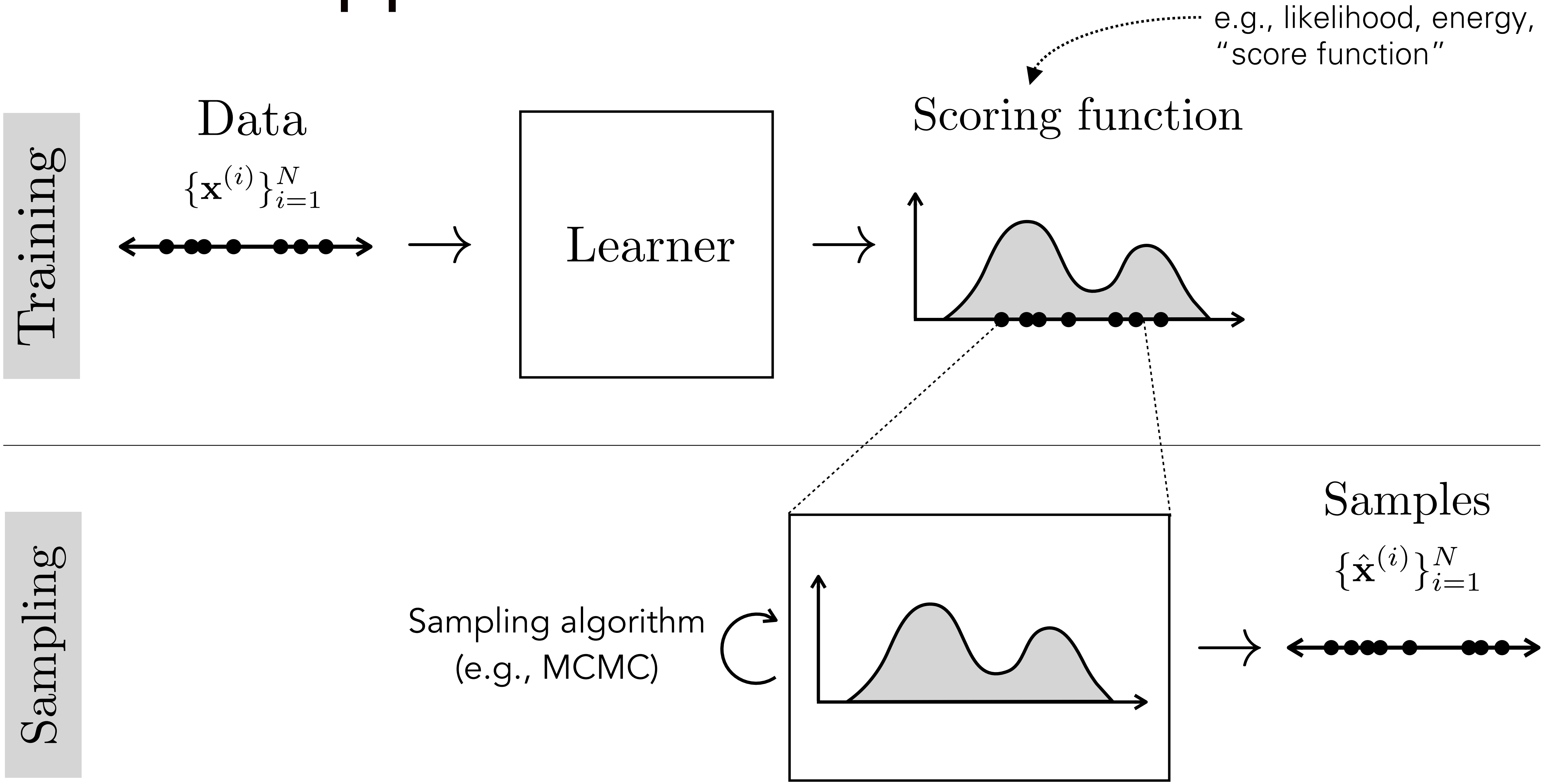
$\mathbf{z} \rightarrow$ Generator $\rightarrow$

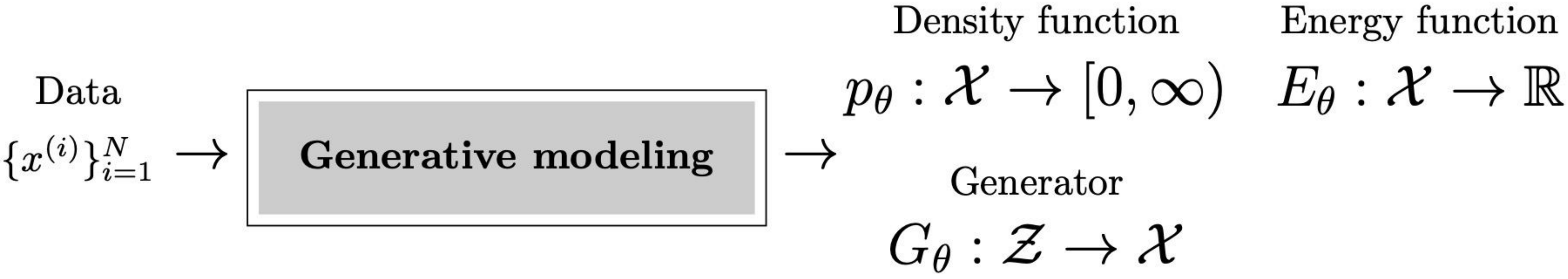
Samples

$\{\hat{\mathbf{x}}^{(i)}\}_{i=1}^{N}$



Indirect approach



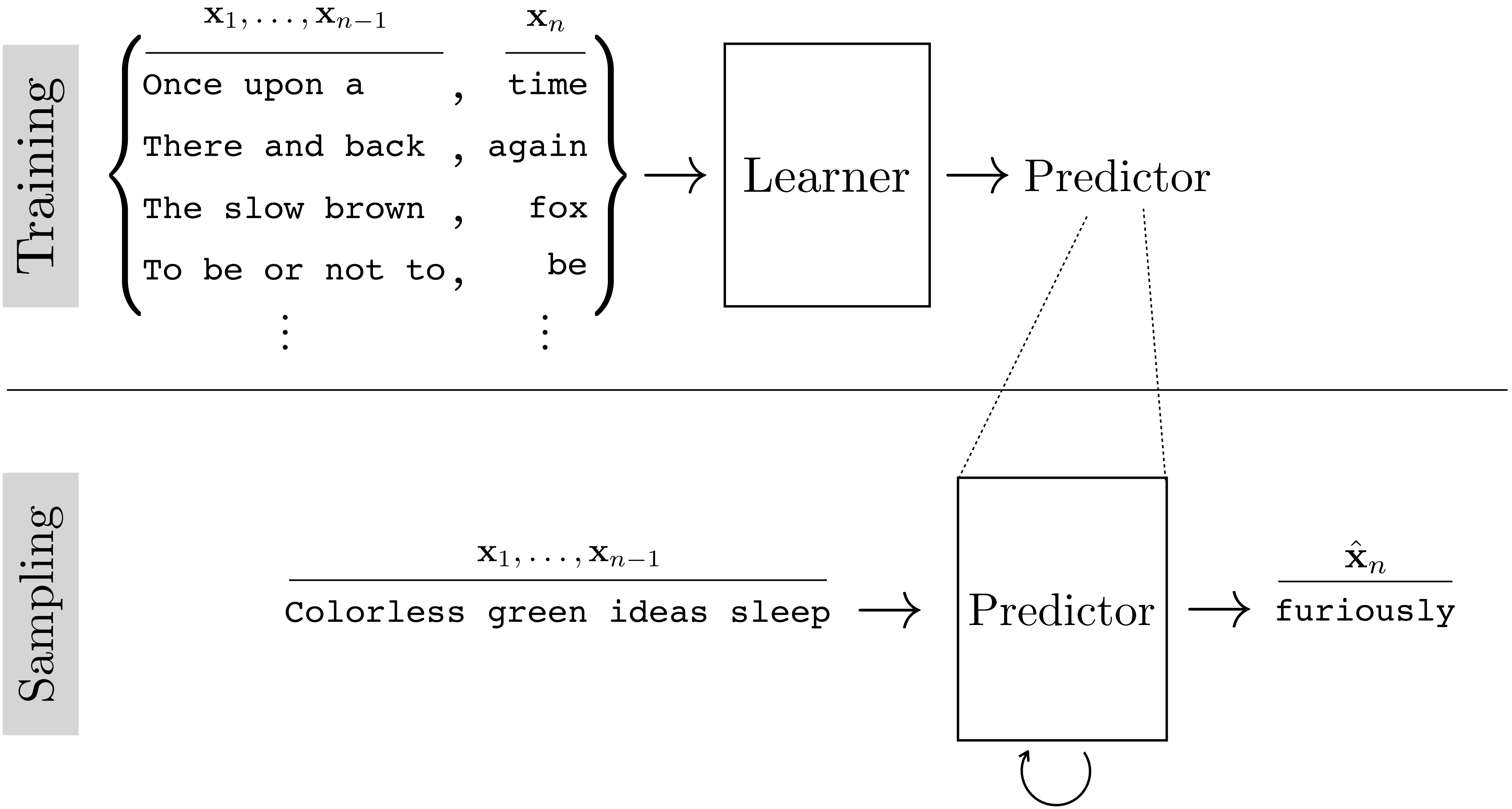


Concept #2: **you can represent the data generating process directly or indireclty**

Autoregressive models

`Once upon ___` → Predictor → `time`

`Once ___ a time` → Predictor → `Upon`


Training
$\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}$
$\mathbf{x}_n$
Once upon a , time
There and back , again
The slow brown , fox
To be or not to, be
Learner
Predictor
Sampling
$\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}$
Colorless green ideas sleep
Predictor
$\hat{\mathbf{x}}_n$
furiously

Autoregressive probability models

$$p(\mathbf{X}) = p(\mathbf{x_n}|\mathbf{x_1}, \ldots, \mathbf{x}_{n-1})p(\mathbf{x}_{n-1}|\mathbf{x}_1, \ldots, \mathbf{x}_{n-2}) \quad \ldots \quad p(\mathbf{x}_2|\mathbf{x}_1)p(\mathbf{x}_1)$$

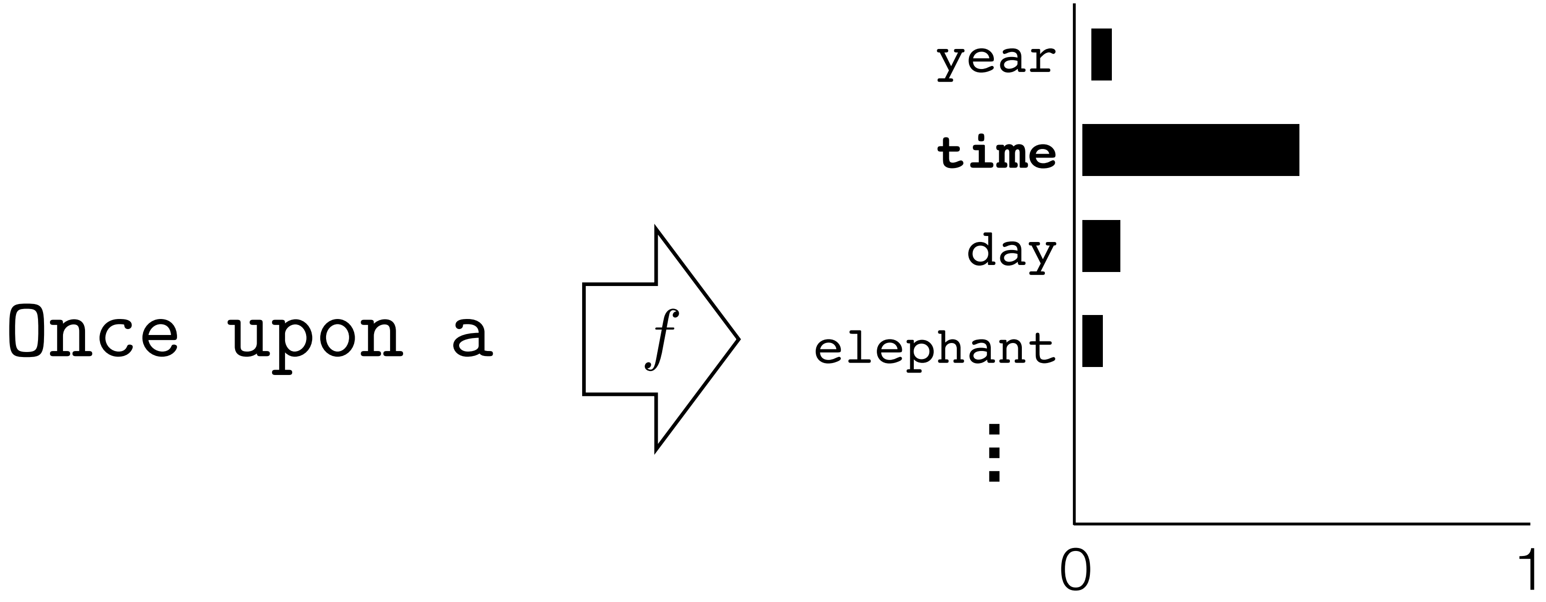
$$p(\mathbf{X}) = \prod_{i=1}^{n} p(\mathbf{x}_i|\mathbf{x}_1, \ldots, \mathbf{x}_{i-1})$$

$$p(\overbrace{\overbrace{\underbrace{\underbrace{\texttt{Once}}_{p(\texttt{Once})}\ \texttt{upon}}_{p(\texttt{upon}|\texttt{Once})}\ \texttt{a}}^{p(\texttt{a}|\texttt{Once},\texttt{upon})}\ \texttt{time}}^{p(\texttt{time}|\texttt{Once},\texttt{upon},\texttt{a})})$$

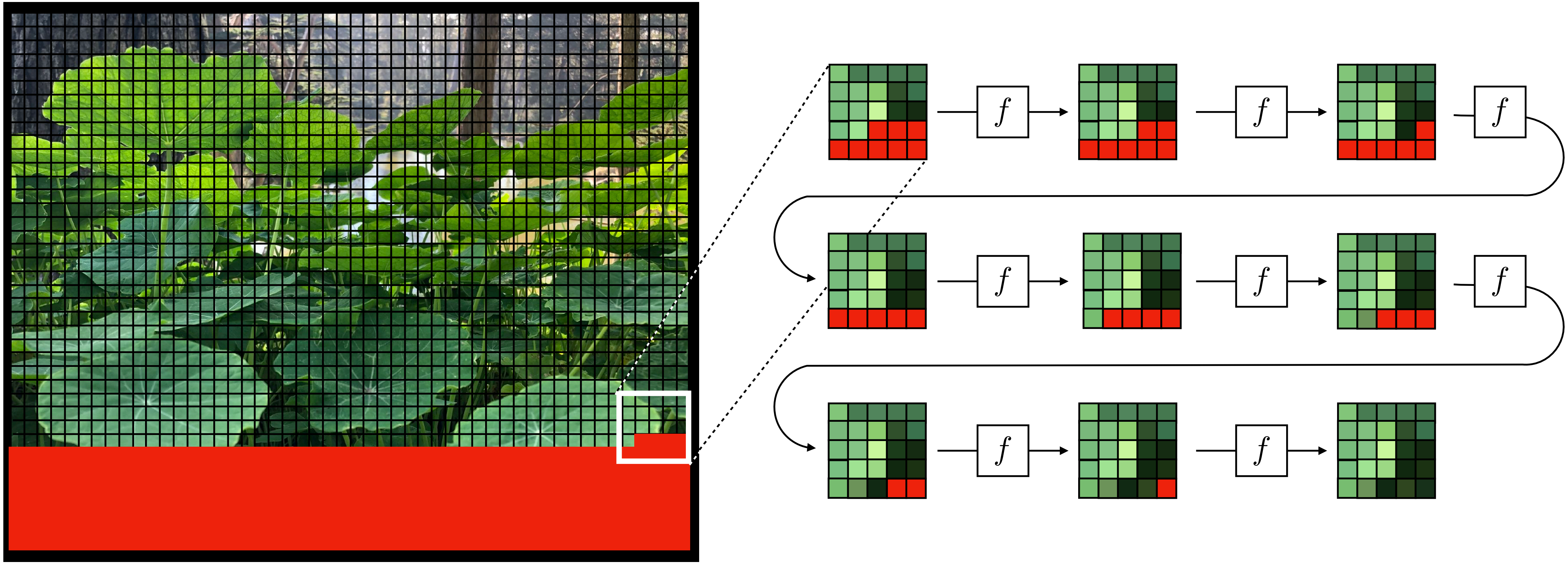
Modeling a sequence of words

How to model $p(\texttt{time}|\texttt{Once}, \texttt{upon}, \texttt{a})$?

Just treat it as a next word classfier!



Autoregressive model of pixels



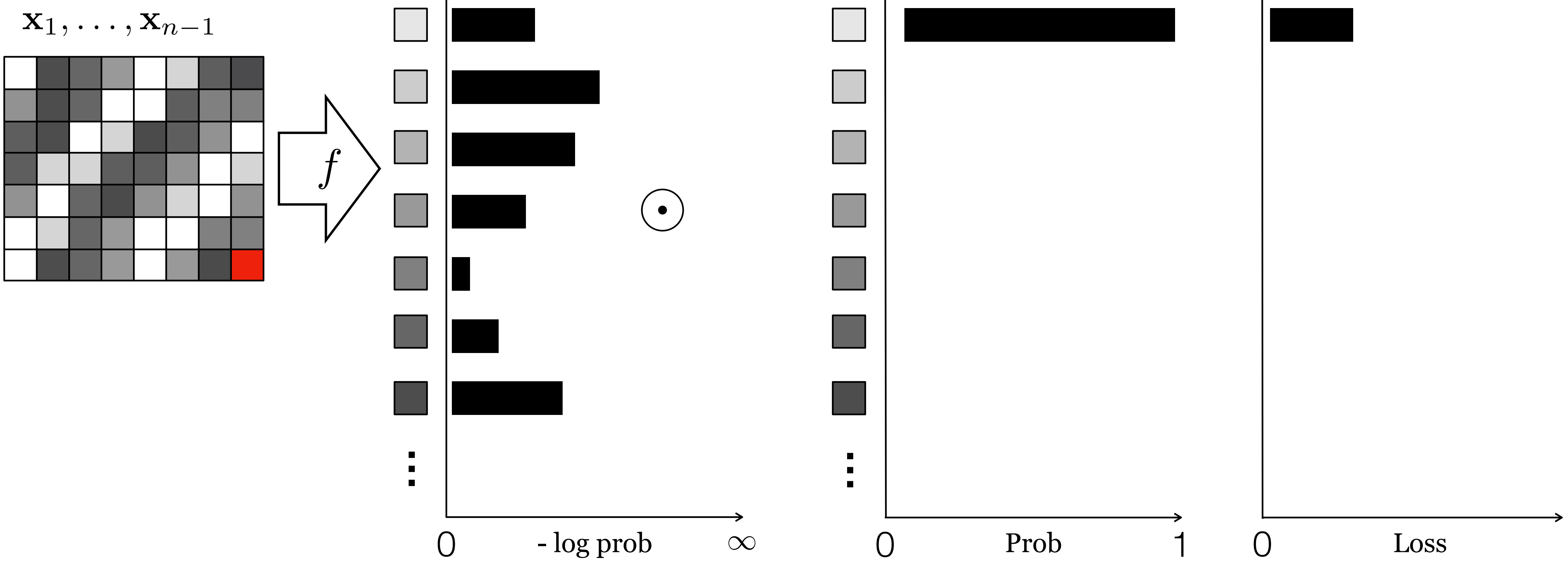
Prediction $\hat{\mathbf{x}}_n$

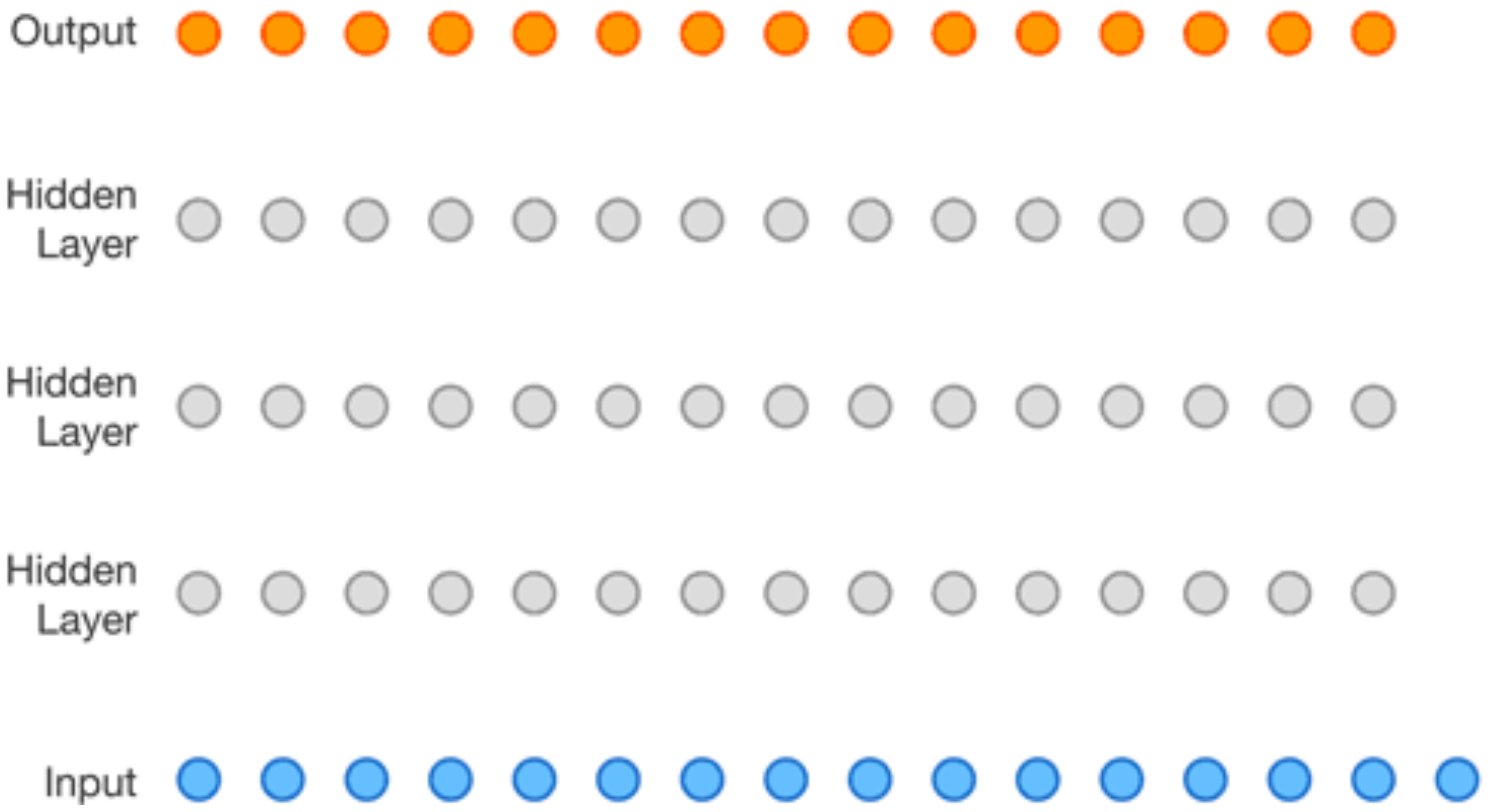
$f : \mathcal{X} \to \mathbb{R}^{256}$

Ground truth label $\mathbf{x}_n$

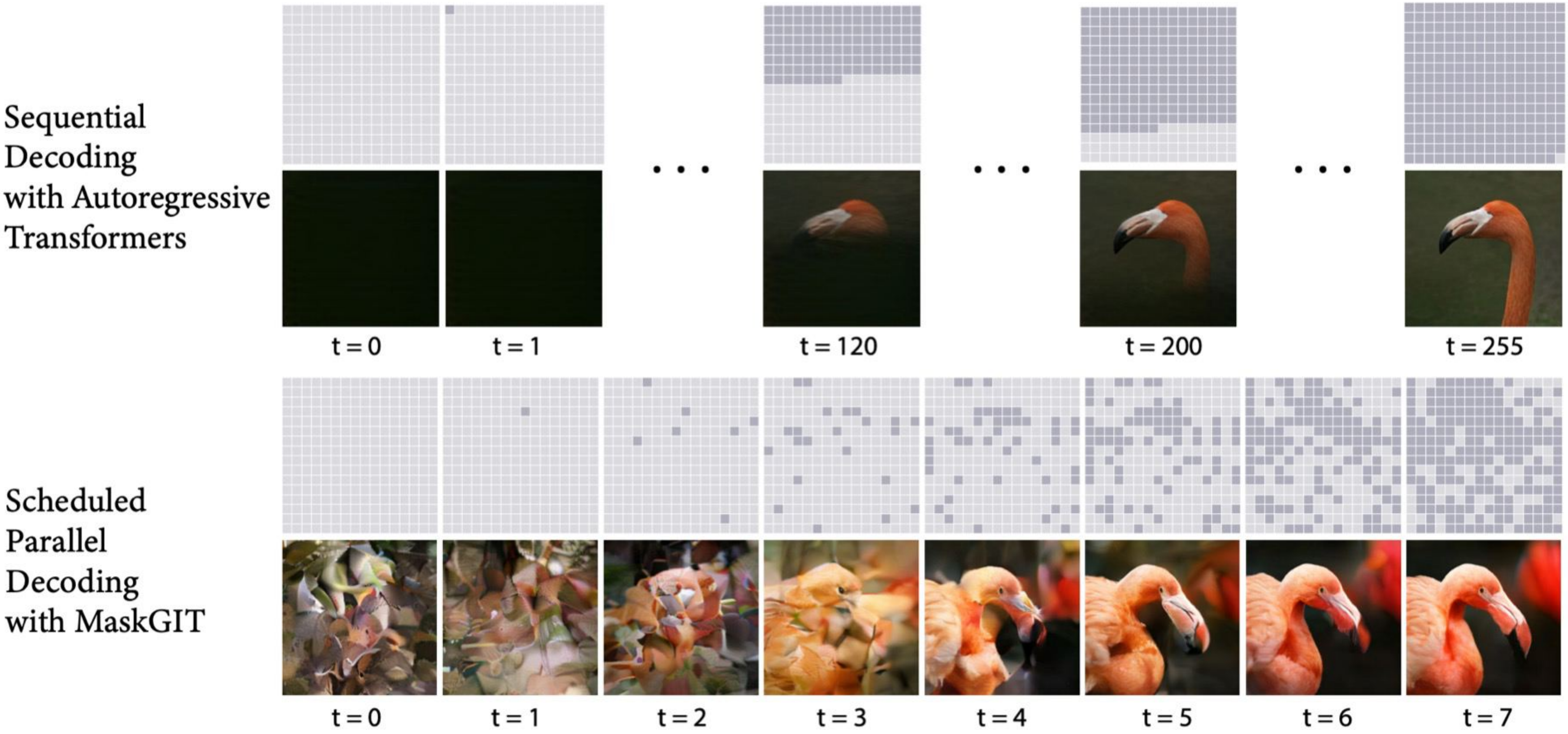
Loss

$$H(\mathbf{x}_n, \hat{\mathbf{x}}_n) = -\sum_{k=1}^{256} x_{n_k} \log \hat{x}_{n_k}$$



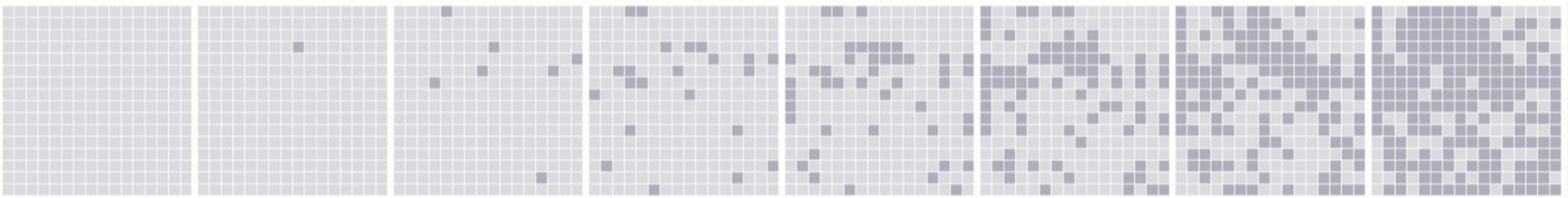


[**Wavenet**, https://www.deepmind.com/blog/wavenet-a-generative-model-for-raw-audio/]

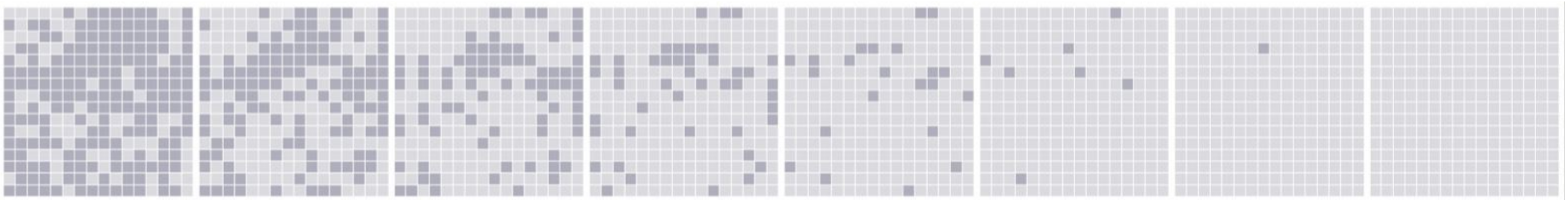


["MaskGIT", Chang et al. 2022]

Diffusion models



Diffusion models

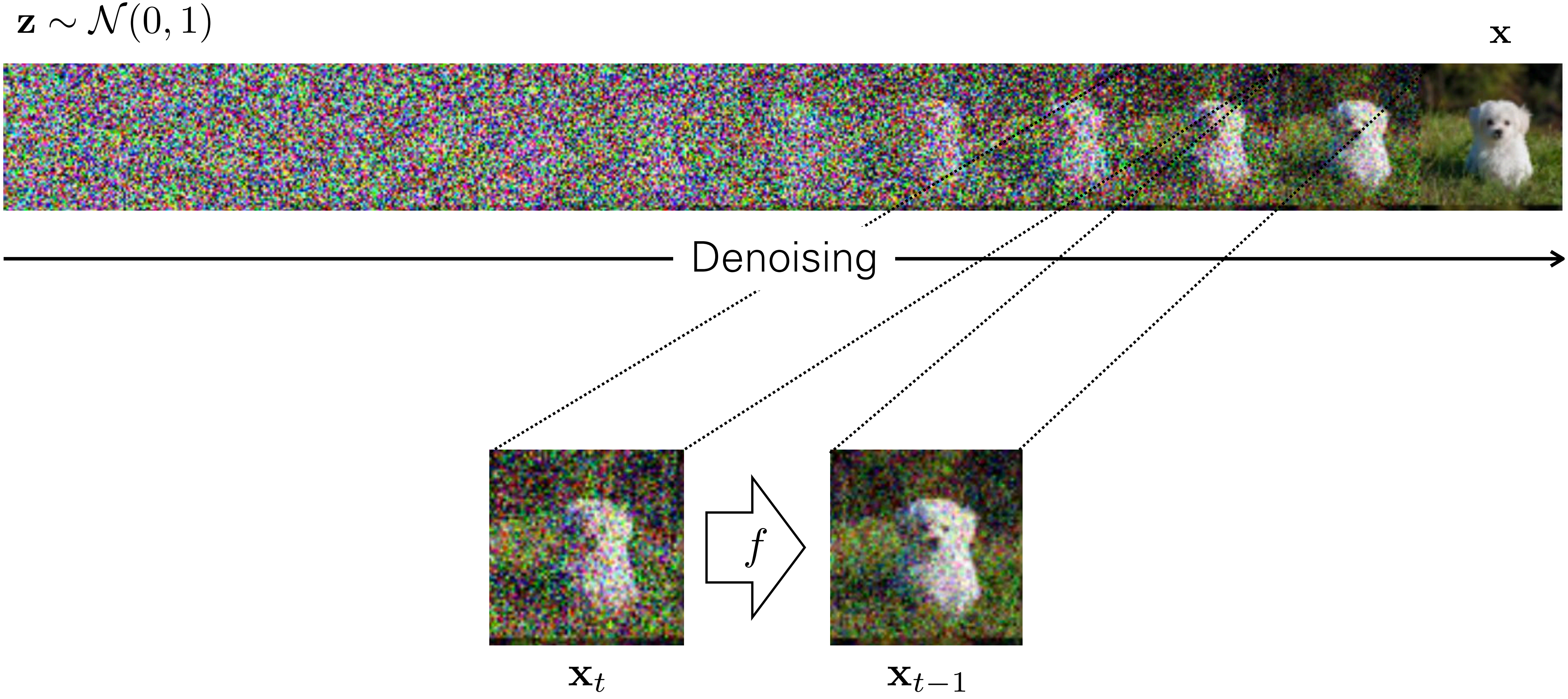


Corrupting the input

Corrupting the input

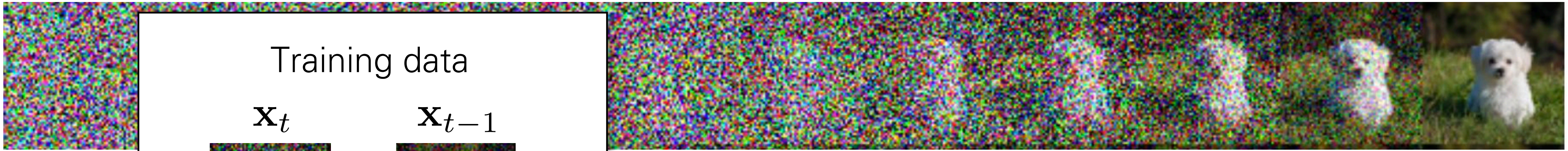
[Image from Nichol & Dhariwal 2021]

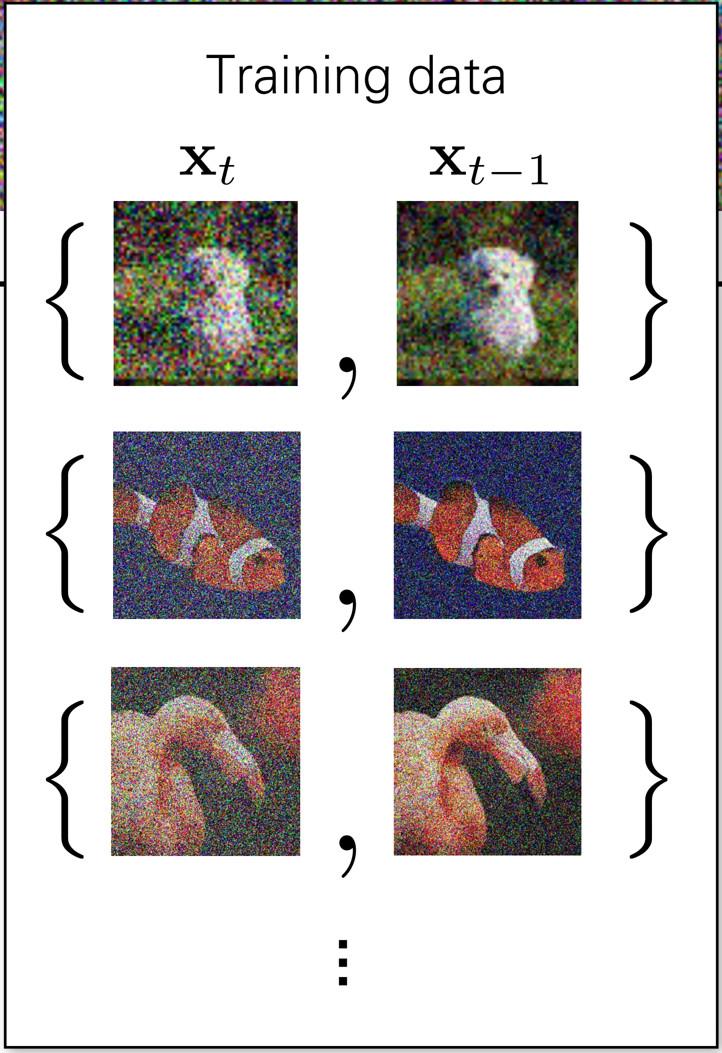
Diffusion models



Diffusion models

$\mathbf{z} \sim \mathcal{N}(0,1)$ $\mathbf{x}$



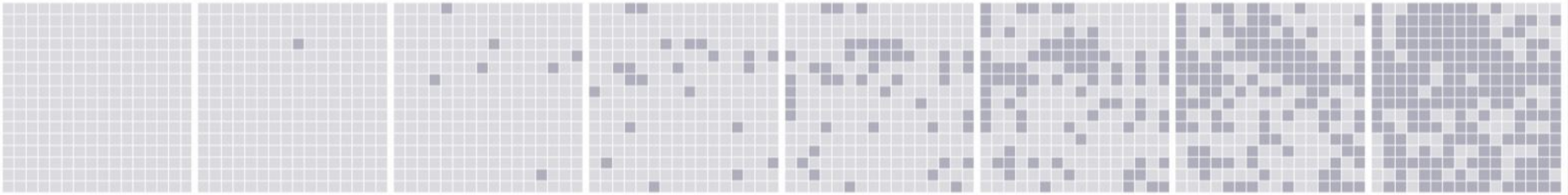


Denoising

$$\arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} \mathcal{L}(f(\mathbf{x}_t), \mathbf{x}_{t-1})$$

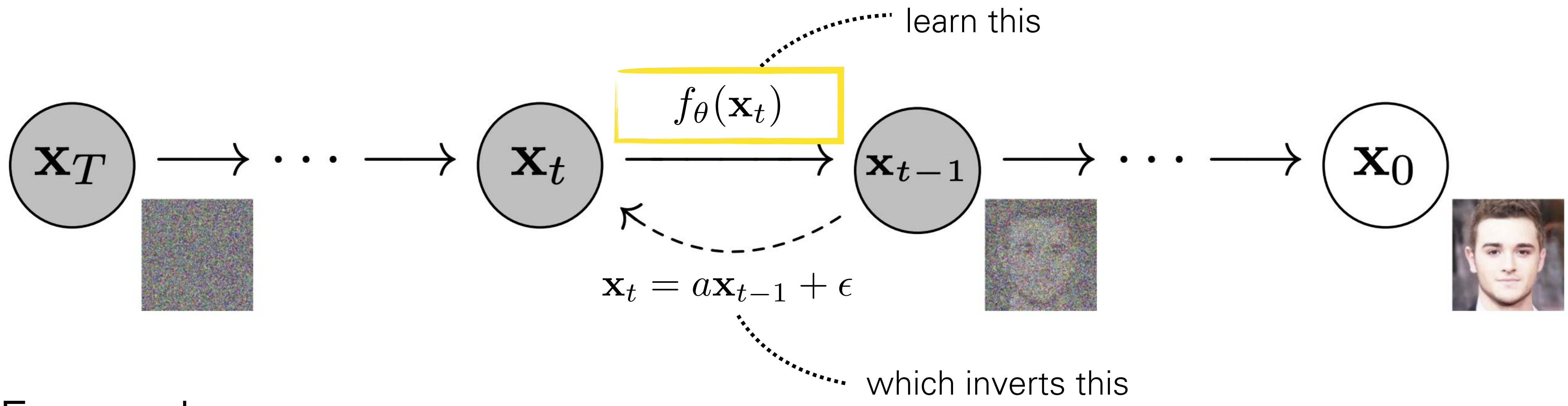
Converts generative modeling into a bunch of supervised prediction problems

Diffusion models



Concept #3: **A common strategy is to turn generative modeling into a sequence of supervised learning problems**

Gaussian diffusion models



Forward process:

$$\epsilon \sim \mathcal{N}(0, b) \qquad \mathbf{x}_t = a\mathbf{x}_{t-1} + \epsilon$$

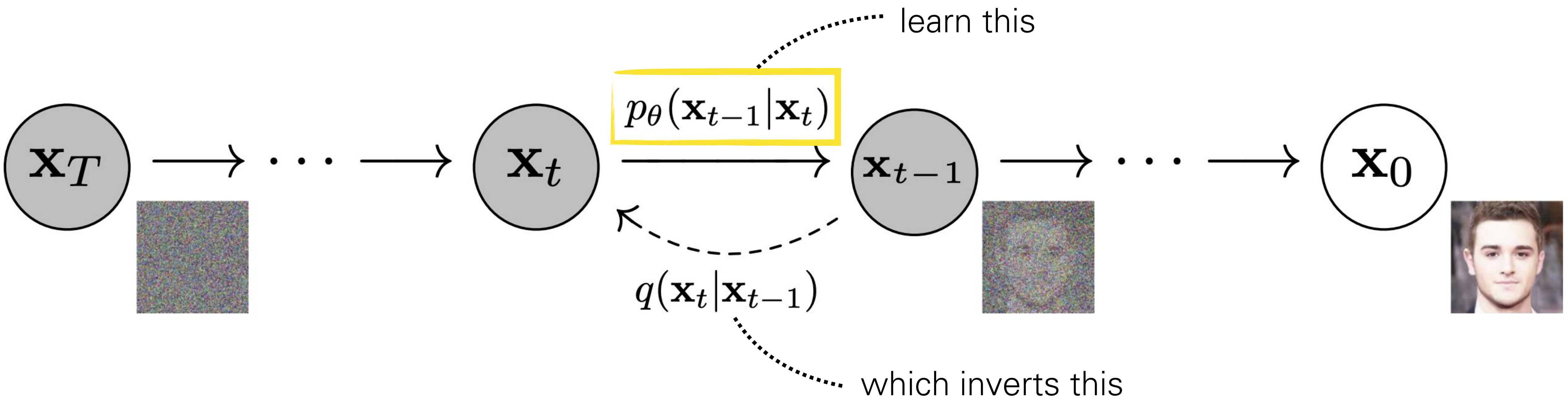
Reverse process:

$$\mu = f_\theta(\mathbf{x}_t) \qquad \mathbf{x}_{t-1} \sim \mathcal{N}(\mu, c)$$

a, b, and c are hyperparameters (see Ho, Jain, Abbeel for one way to set them)

[Image from Ho, Jain, Abbeel, 2020]

Gaussian diffusion models



Forward process:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(a\mathbf{x}_{t-1}, b)$$

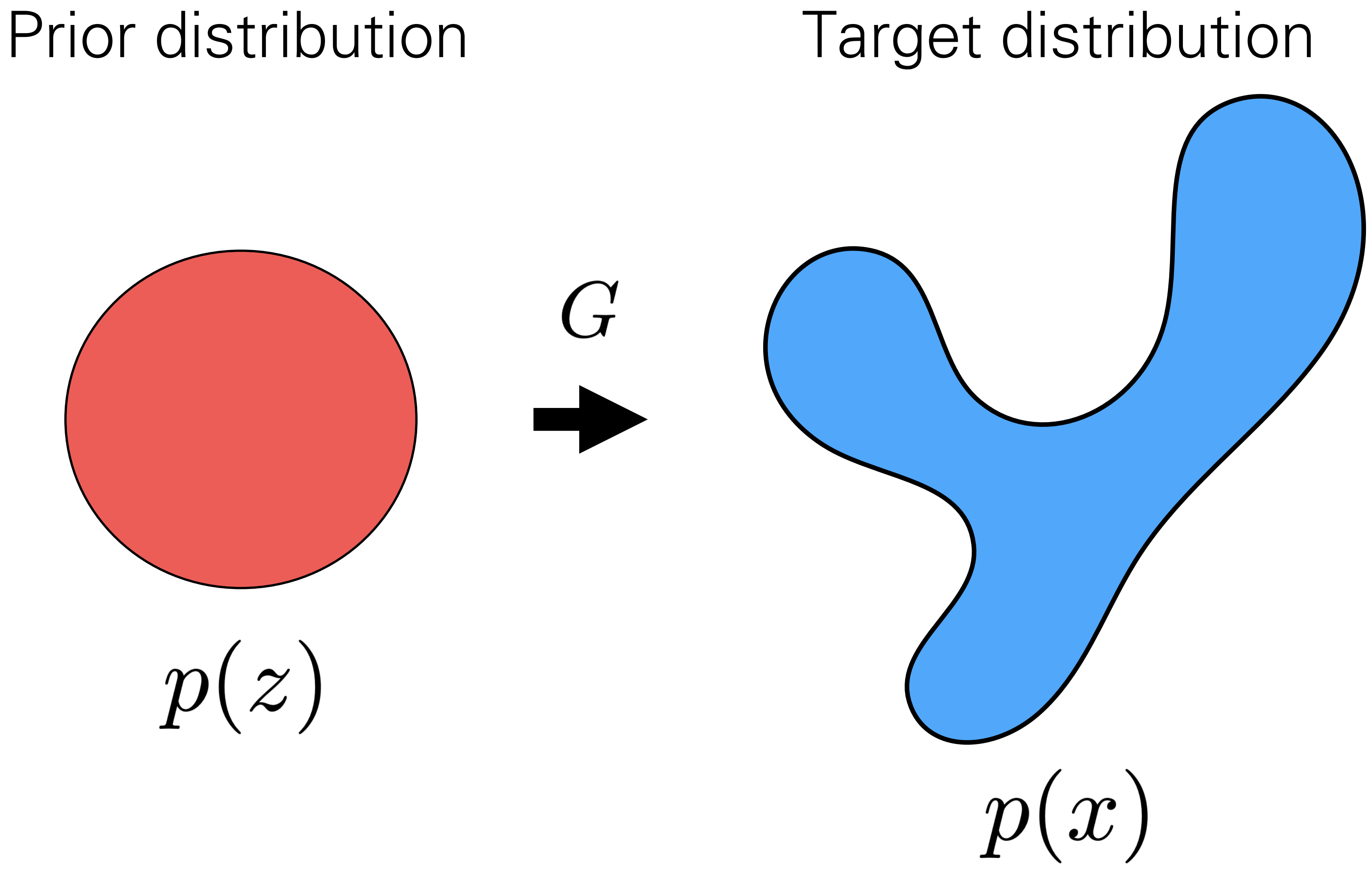
Reverse process:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(f_\theta(\mathbf{x}_t), c)$$

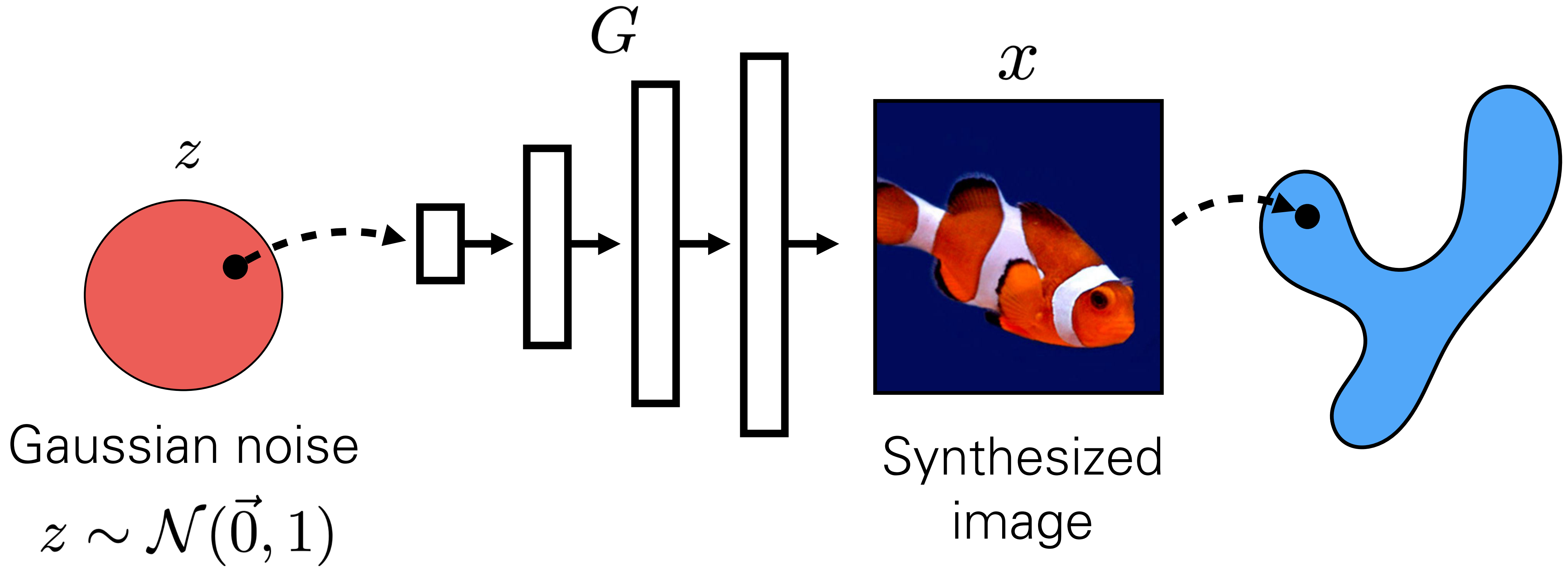
a, b, and c are hyperparameters (see Ho, Jain, Abbeel for one way to set them)

[Image from Ho, Jain, Abbeel, 2020]

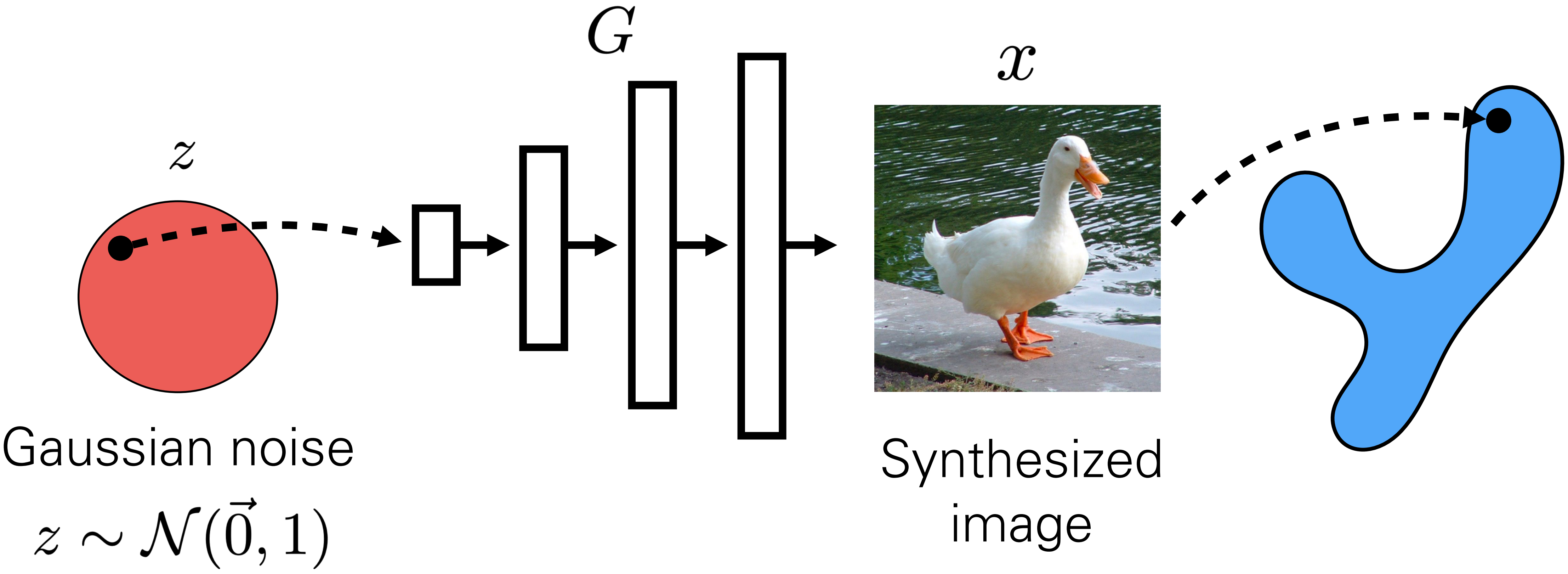
Deep generative models are distribution transformers



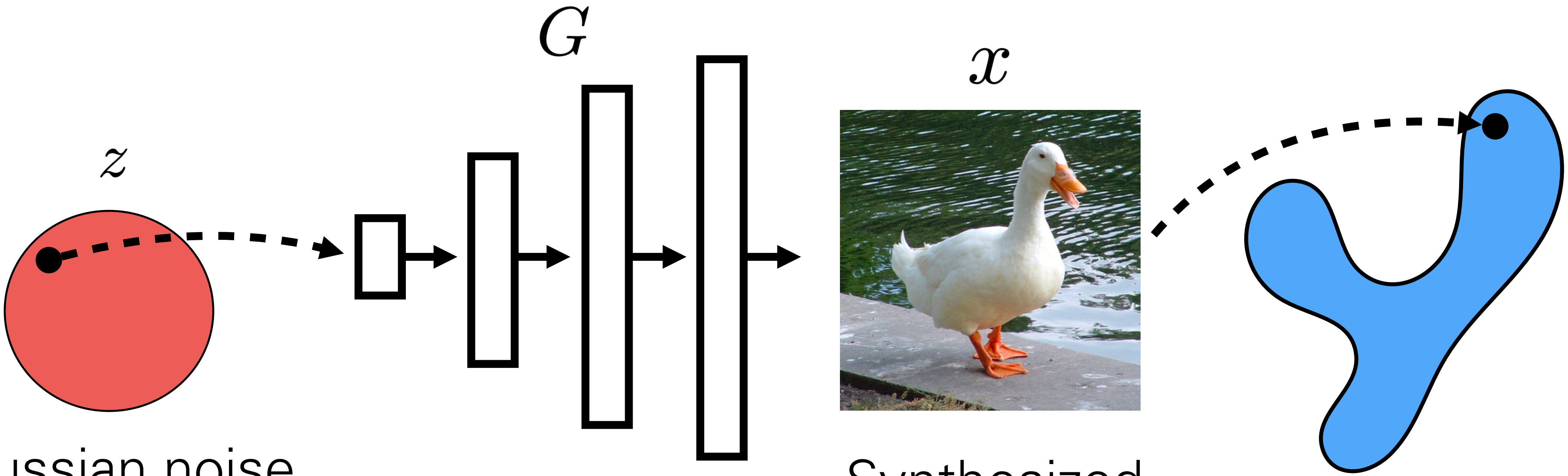
Deep generative models are distribution transformers



Deep generative models are distribution transformers



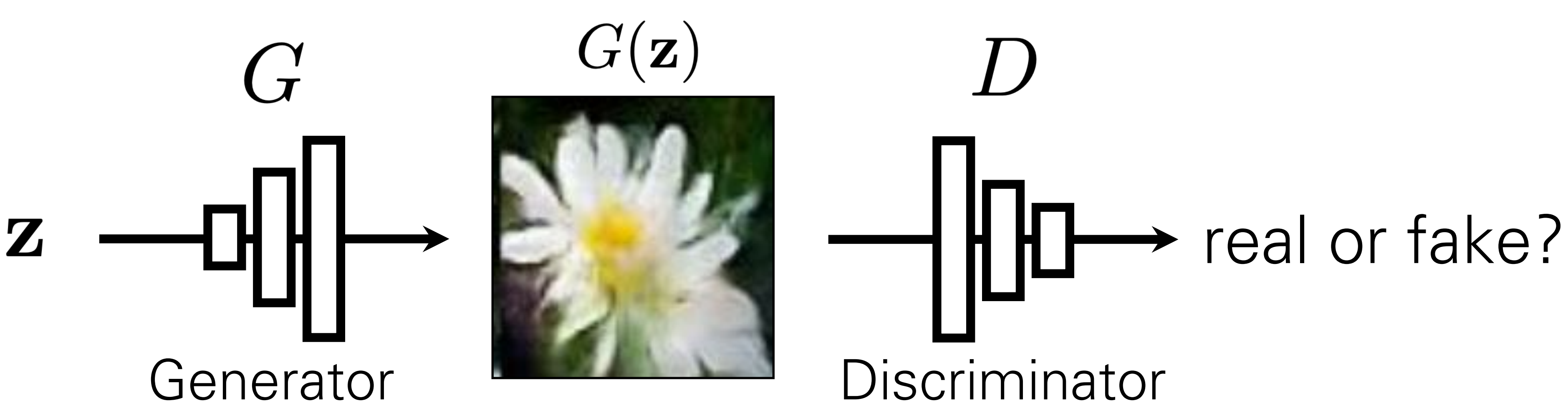
Generative Adversarial Networks (GANs)



Gaussian noise

$z \sim \mathcal{N}(\vec{0}, 1)$

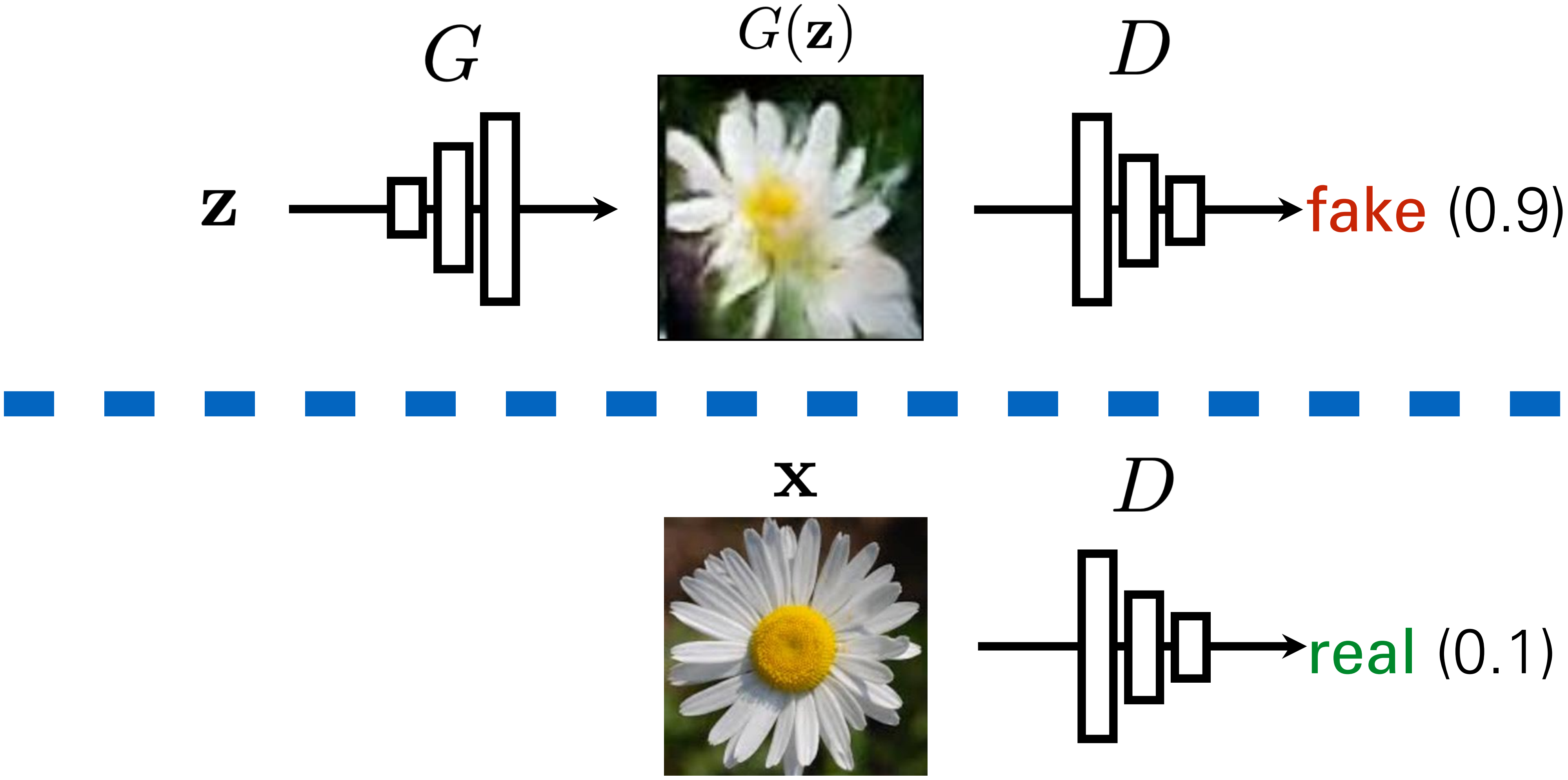
Synthesized image



G tries to synthesize fake images that fool D

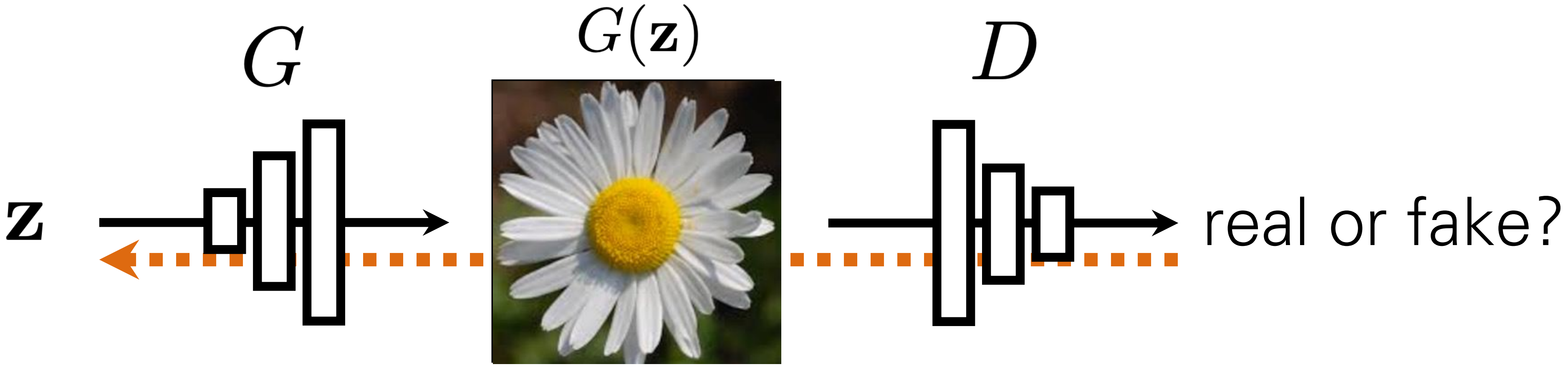
D tries to identify the fakes

[Goodfellow et al., 2014]



$$\arg\max_{D} \mathbb{E}_{\mathbf{z},\mathbf{x}}[\ \boxed{\log D(G(\mathbf{z}))} + \boxed{\log(1 - D(\mathbf{x}))}\ ]$$

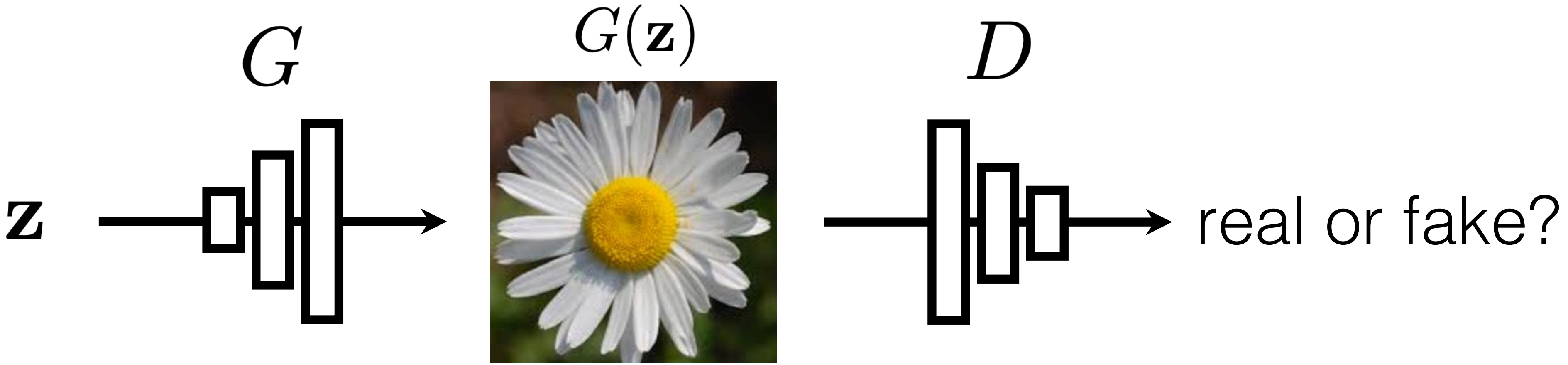
[Goodfellow et al., 2014]



G tries to synthesize fake images that *fool* D:

$$\arg\boxed{\min_{G}}\ \mathbb{E}_{\mathbf{z},\mathbf{x}}\left[\ \log D(G(\mathbf{z})) \ + \ \log\left(1 - D(\mathbf{x})\right)\ \right]$$

[Goodfellow et al., 2014]

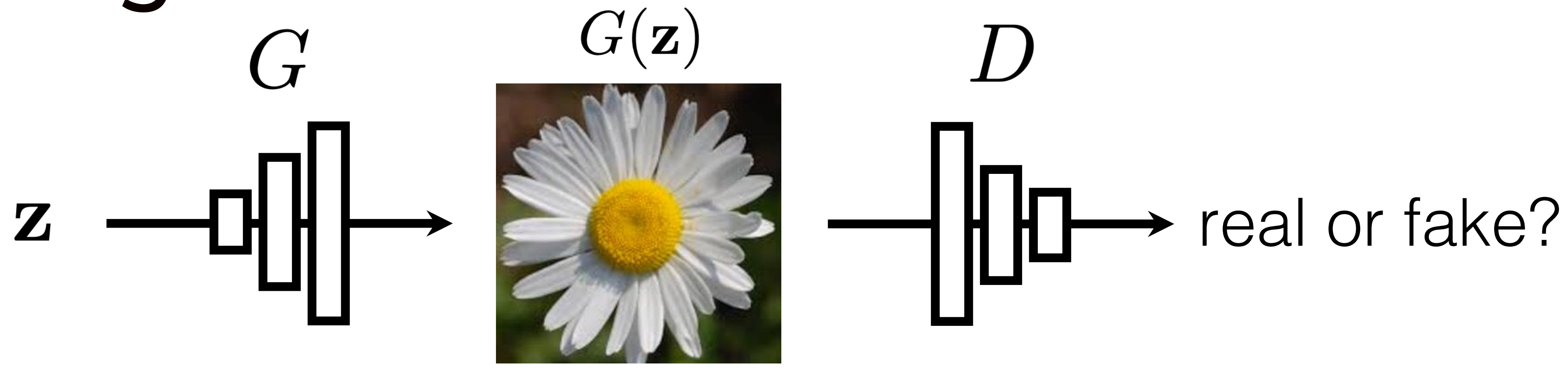


G tries to synthesize fake images that *fool* the *best* D:

$$\arg \boxed{\min_G} \boxed{\max_D} \; \mathbb{E}_{\mathbf{z},\mathbf{x}}[\; \log D(G(\mathbf{z})) \; + \; \log(1 - D(\mathbf{x})) \;]$$

[Goodfellow et al., 2014]

Training



G tries to synthesize fake images that fool **D**

D tries to identify the fakes

- Training: iterate between training **D** and **G** with backprop.
- Global optimum when **G** reproduces data distribution.

[Goodfellow et al., 2014]

GAN Training: Minimax Game (Goodfellow et al., 2014)

$$\min_{\theta} \max_{\omega} \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}} \left[\log D_\omega(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}} \left[\log\left(1 - D_\omega(G_\theta(\boldsymbol{z}))\right)\right]$$

Real data

Noise vector used to generate data

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

Cross-entropy loss for binary classification

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D\left(G(\boldsymbol{z})\right)$$

Generator maximizes the log-probability of the discriminator being mistaken

- Equilibrium of the game
- Minimizes the Jensen-Shannon divergence

GAN Training: Minimax Game (Goodfellow et al., 2014)

$$\min_{\theta} \max_{\omega} \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}} [\log D_{\omega}(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}} [\log (1 - D_{\omega}(G_{\theta}(\boldsymbol{z})))]$$

Real data

Noise vector used to

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim}$$

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D(G(\boldsymbol{z}))$$

ross-entropy
ss for binary
assification

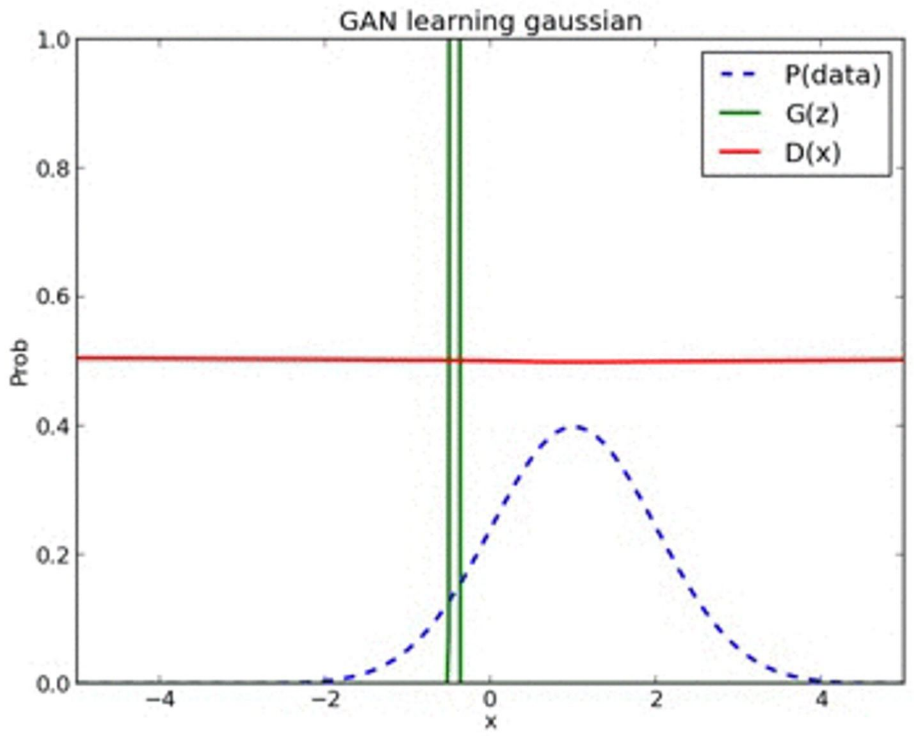
probability
of the discriminator being mistaken

Important question is "Does this converge??"

- Equilibrium of the game
- Minimizes the Jensen-Shannon divergence

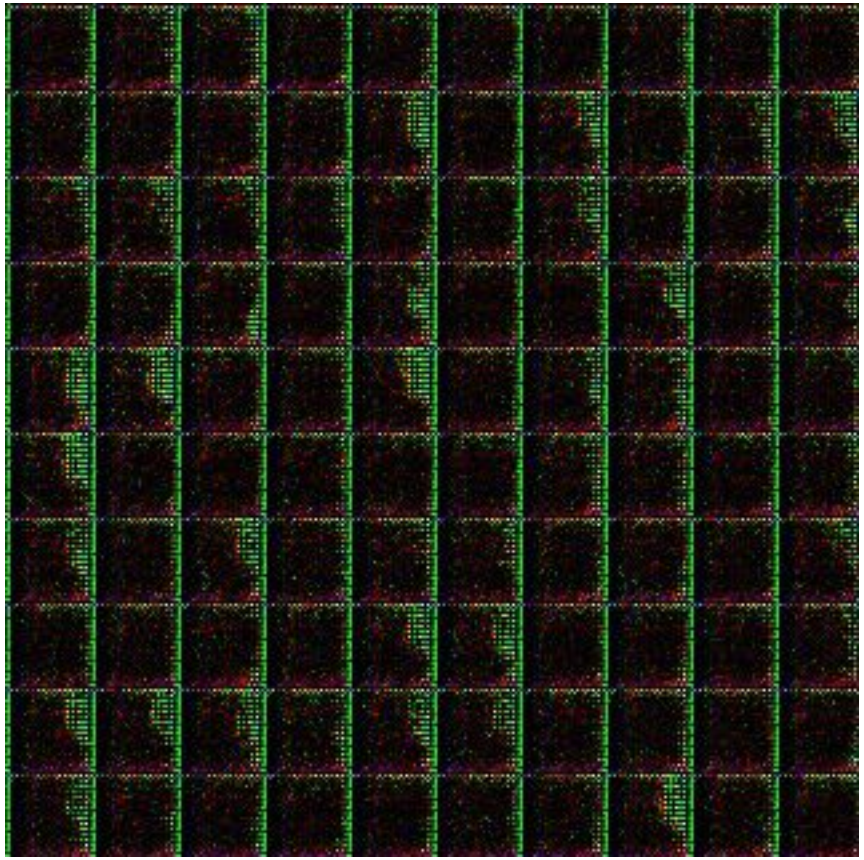
Training Procedure

(Goodfellow et al., 2014)



Source: Alec Radford

Generating 1D points



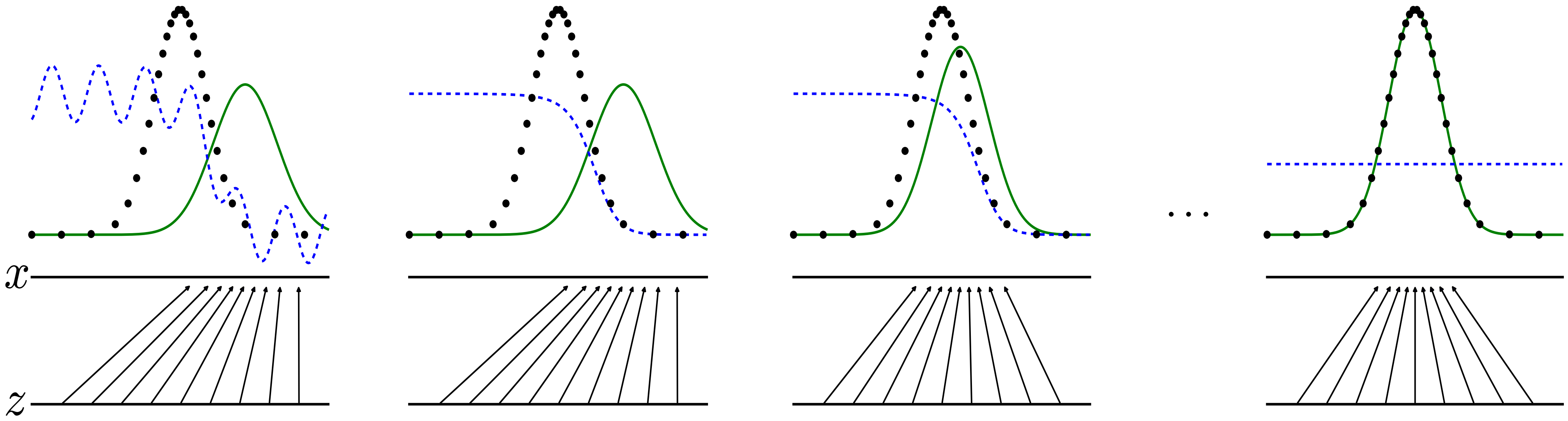
Source: OpenAI blog

Generating images

Training Procedure

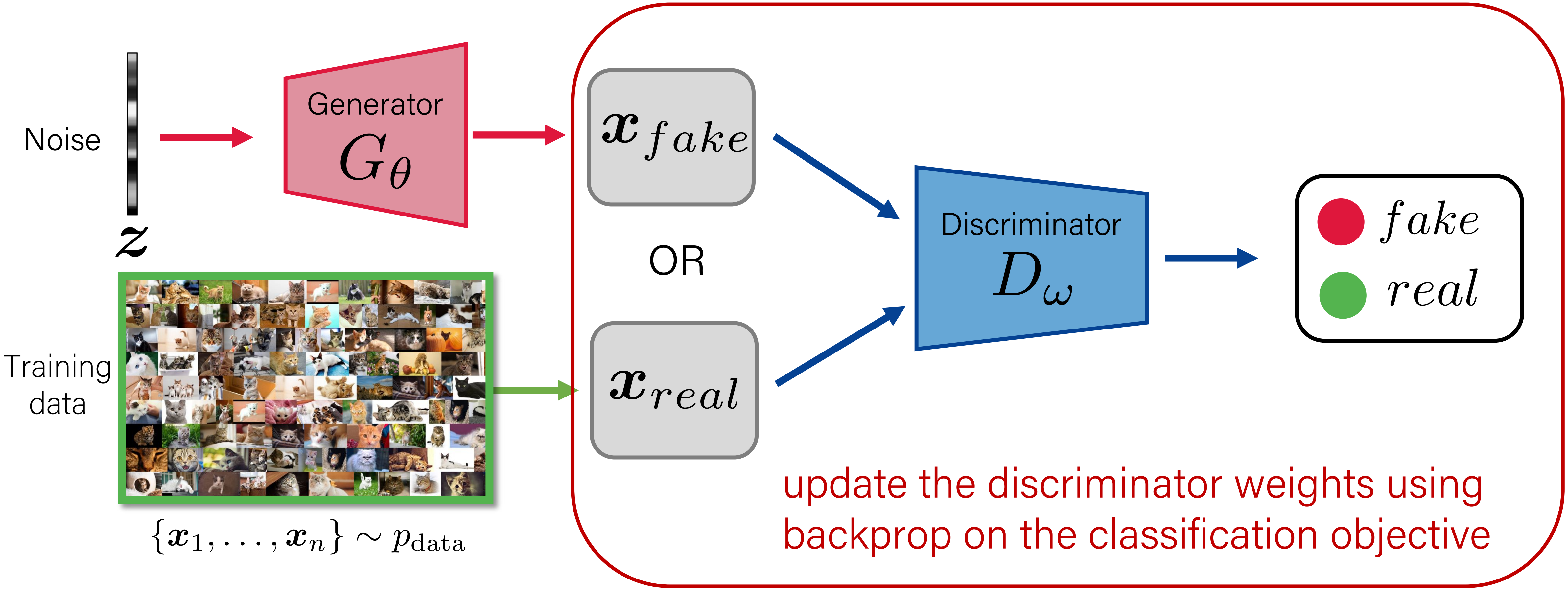
(Goodfellow et al., 2014)

- Use SGD on two minibatches simultaneously:
  - A minibatch of training examples
  - A minibatch of generated samples



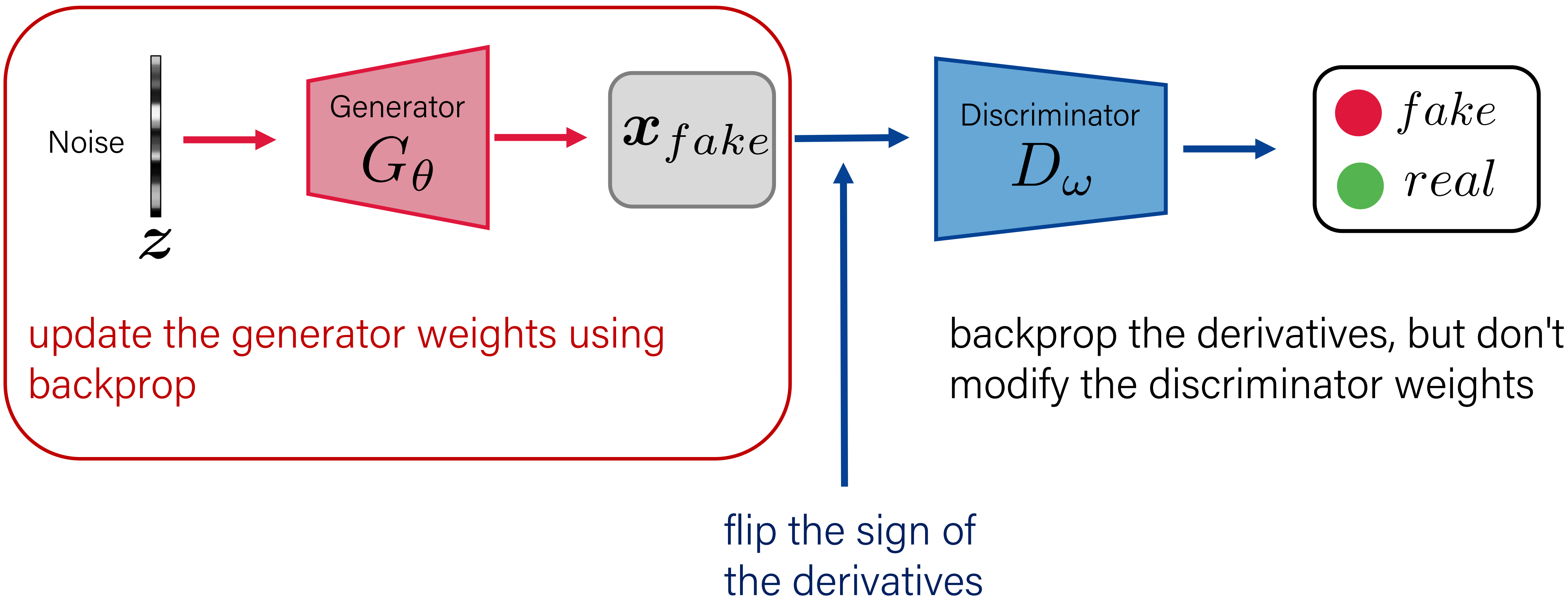
Training Procedure

- Updating the discriminator:



Training Procedure

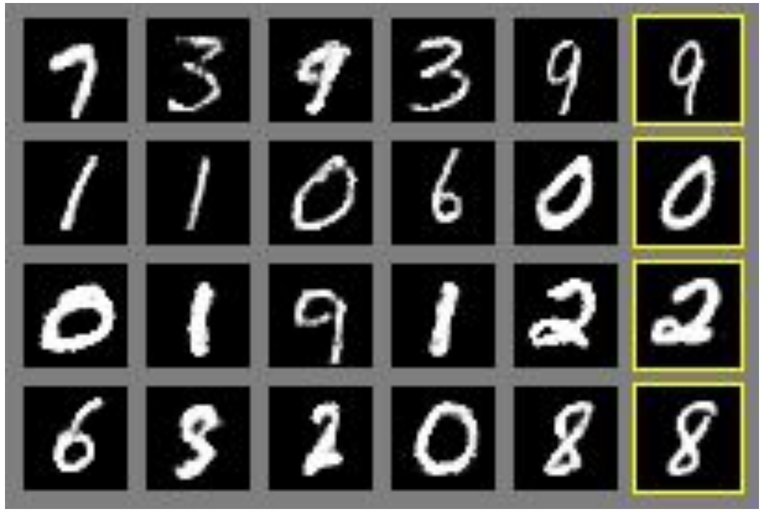
- Updating the generator:



Early Results

(Goodfellow et al., 2014)

- The generator uses a mixture of rectifier linear activations and/or sigmoid activations
- The discriminator net used maxout activations.



MNIST samples

TFD samples

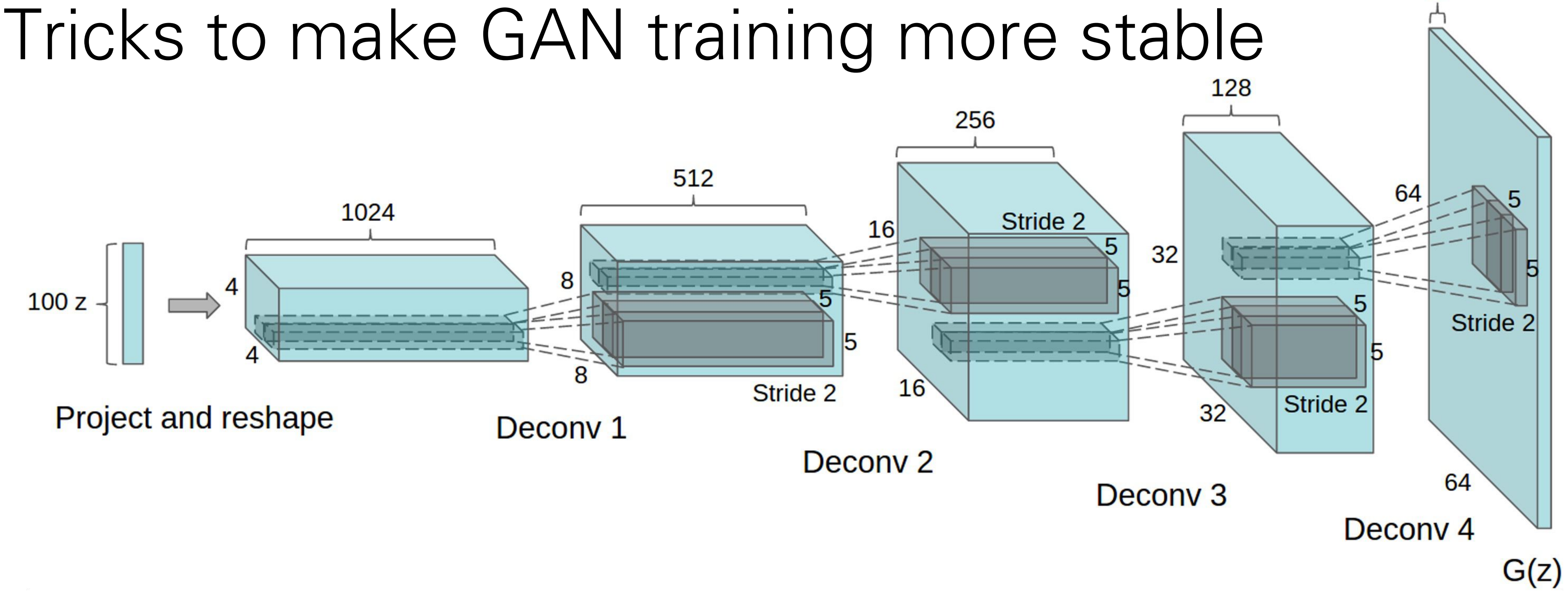
CIFAR10 samples
(*fully-connected model*)

CIFAR10 samples
(*convolutional discriminator, deconvolutional generator*)

Deep Convolutional GANs (DCGAN)

(Radford et al., 2015)

- **Idea:** Tricks to make GAN training more stable



- No fully connected layers
- Batch Normalization (Ioffe and Szegedy, 2015)
- Leaky Rectifier in D
- Use Adam (Kingma and Ba, 2015)
- Tweak Adam hyperparameters a bit (lr=0.0002, b1=0.5)

DCGAN for LSUN Bedrooms

64×64 pixels
~3M images

(Radford et al., 2015)

Walking over the latent space

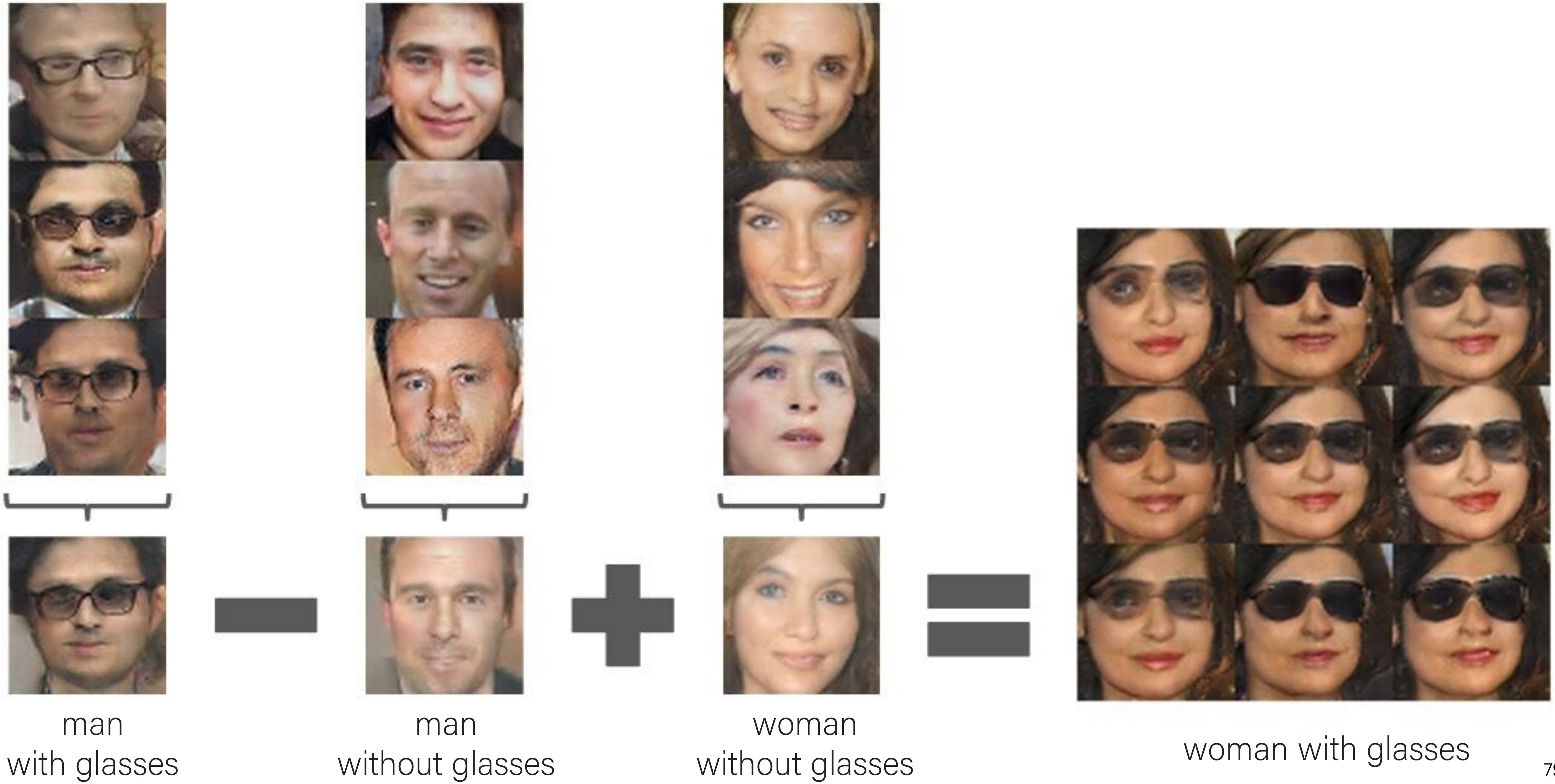
(Radford et al., 2015)

- Interpolation suggests non-overfitting behavior

Walking over the latent space (Radford et al., 2015)

Vector Space Arithmetic

(Radford et al., 2015)



Vector Space Arithmetic

(Radford et al., 2015)

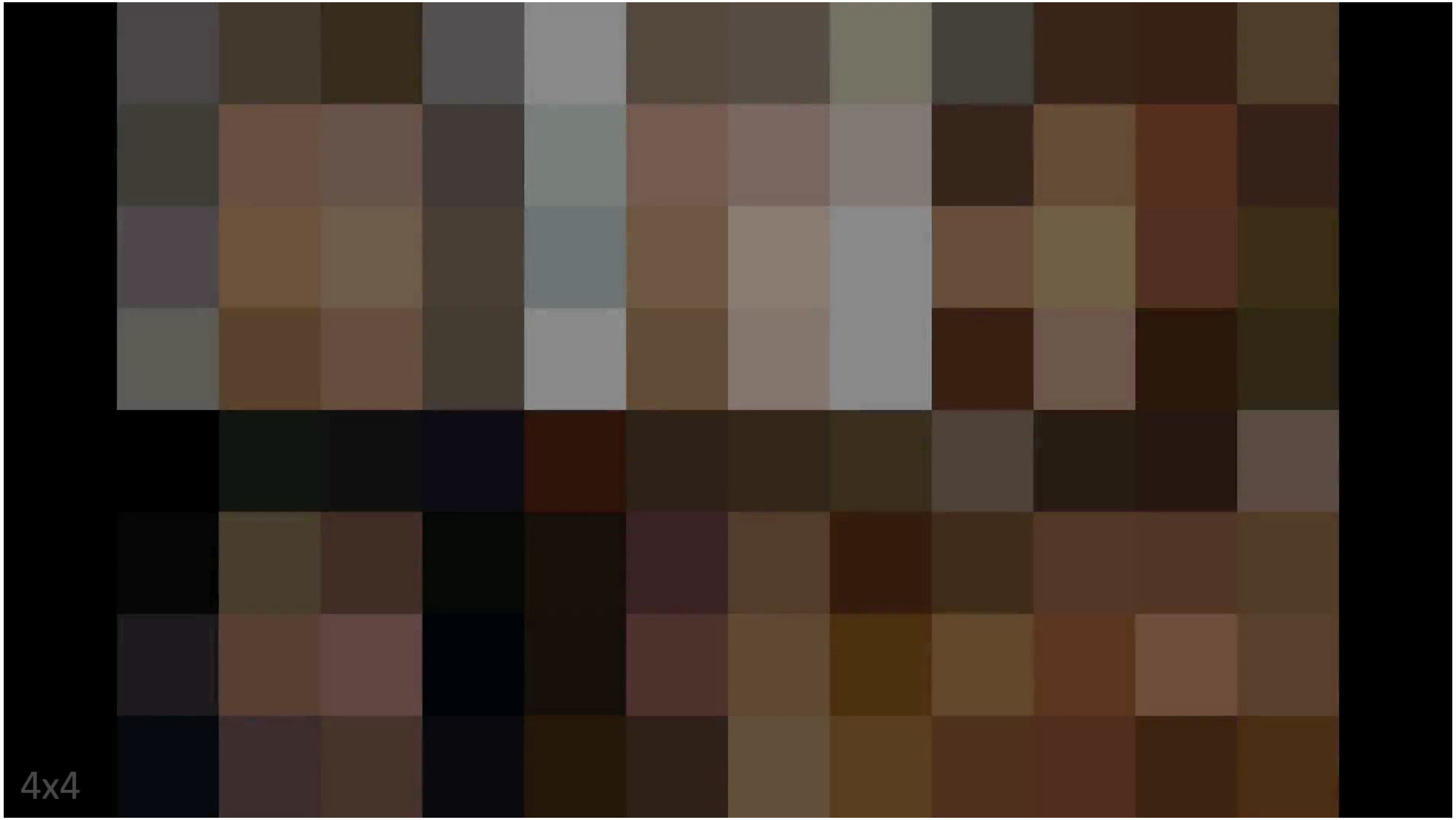
smiling woman

neutral woman

neutral man

smiling man

Progressive GANs(Karras et al., 2018)



Progressive GANs (Karras et al., 2018)

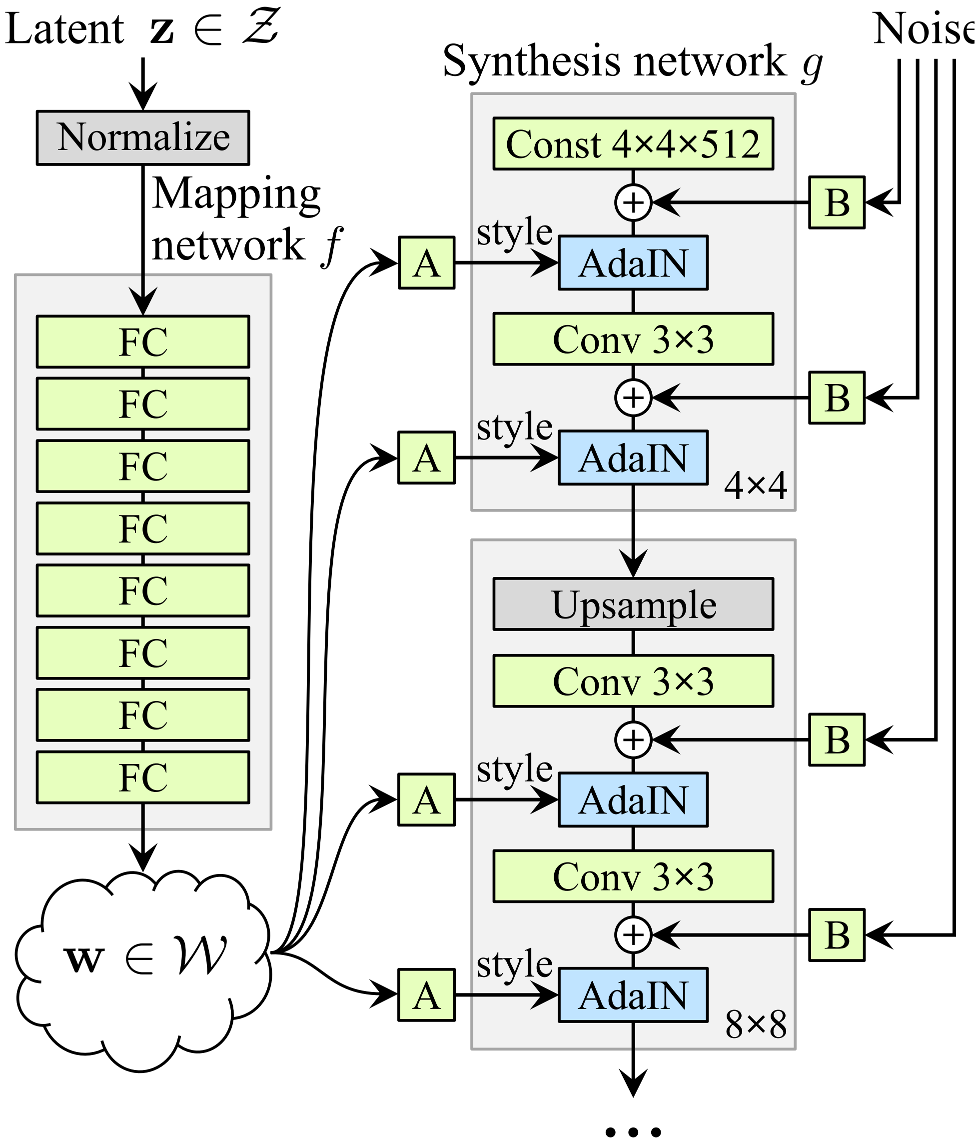
CelebA-HQ
random interpolations

Samples from BigGAN

[Brock et al. 2018]

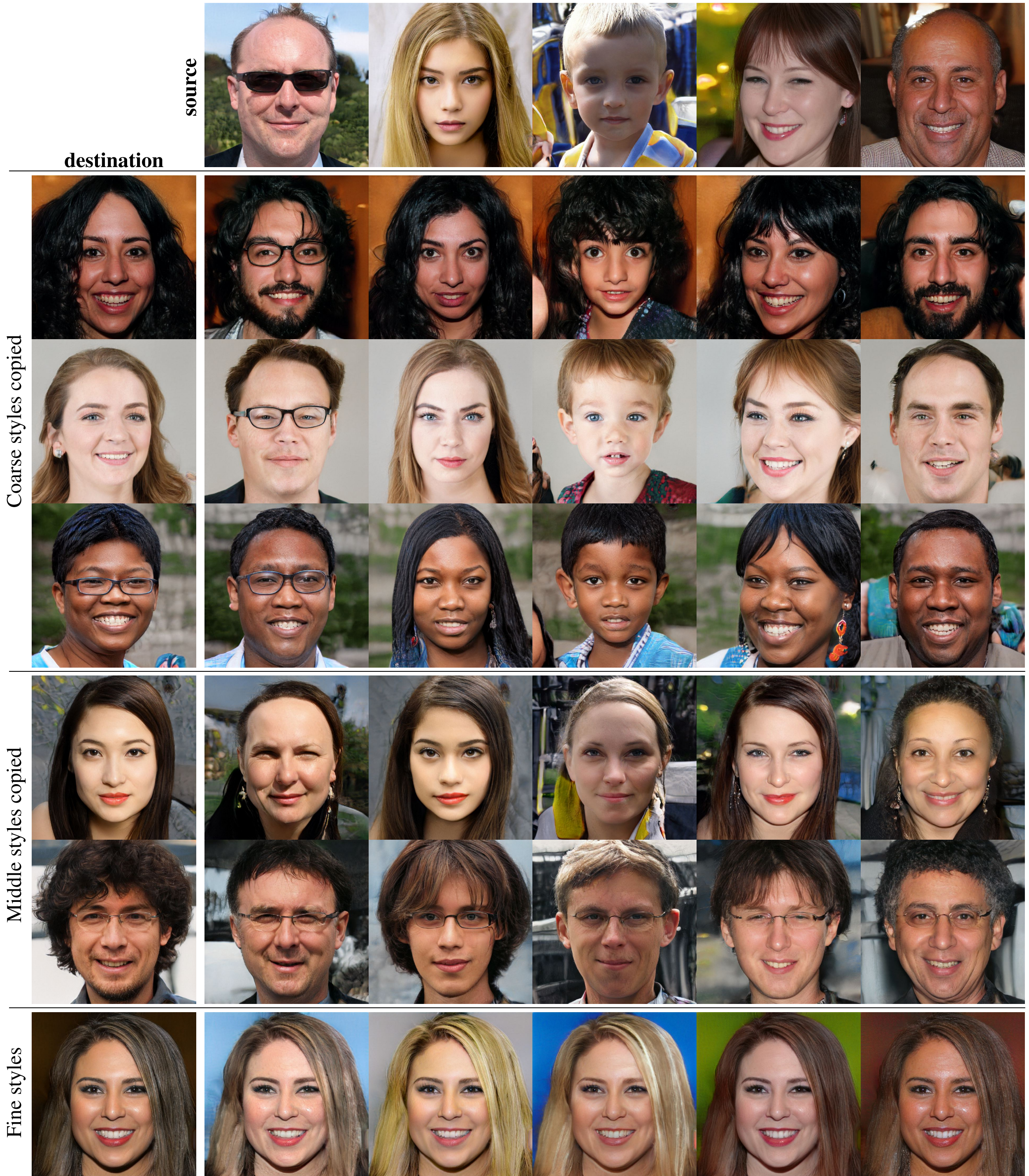
StyleGANs (Karras et al., 2019)

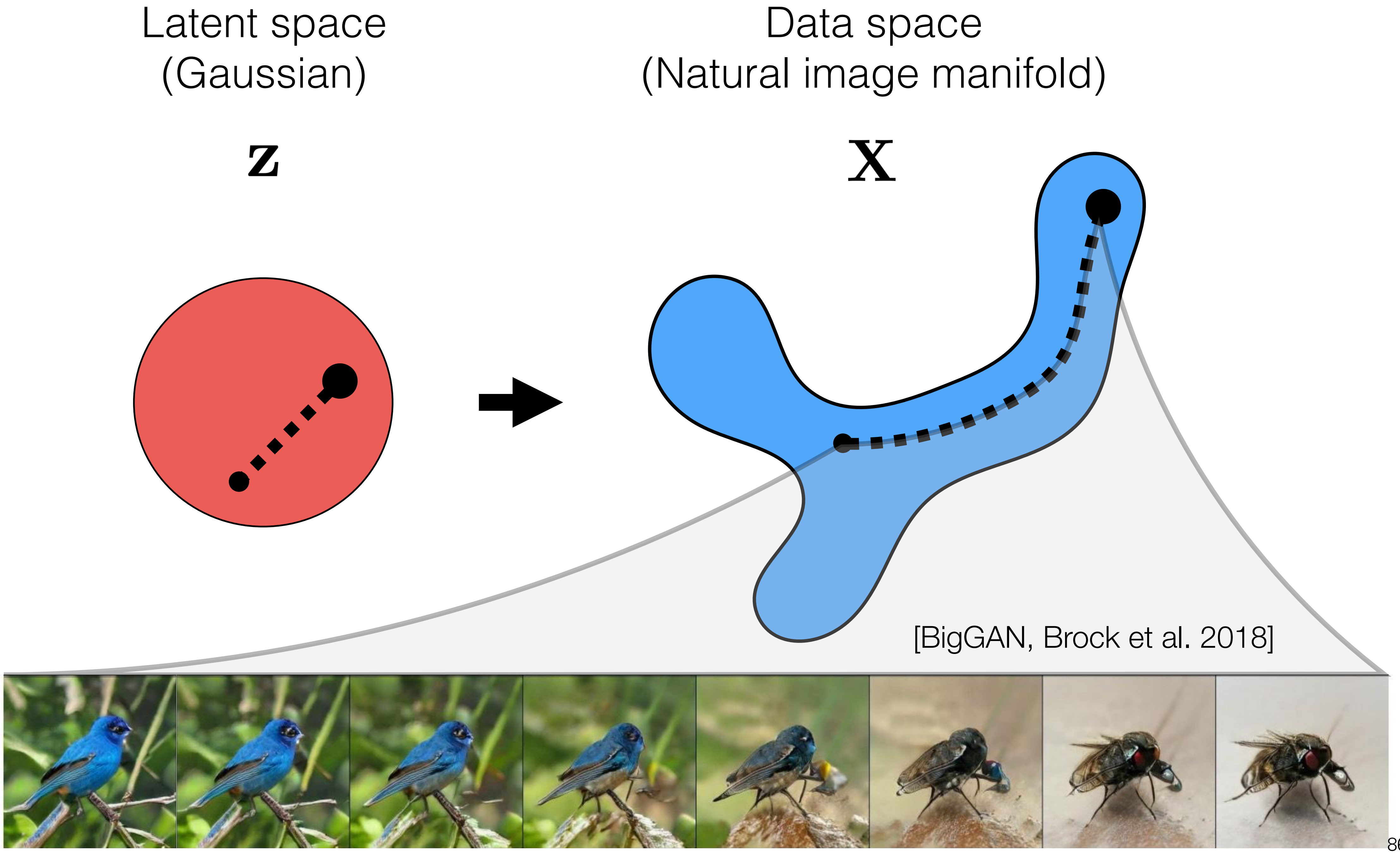
- An architecture motivated by the style transfer networks
- allows unsupervised separation of high-level attributes and stochastic variation in the generated images



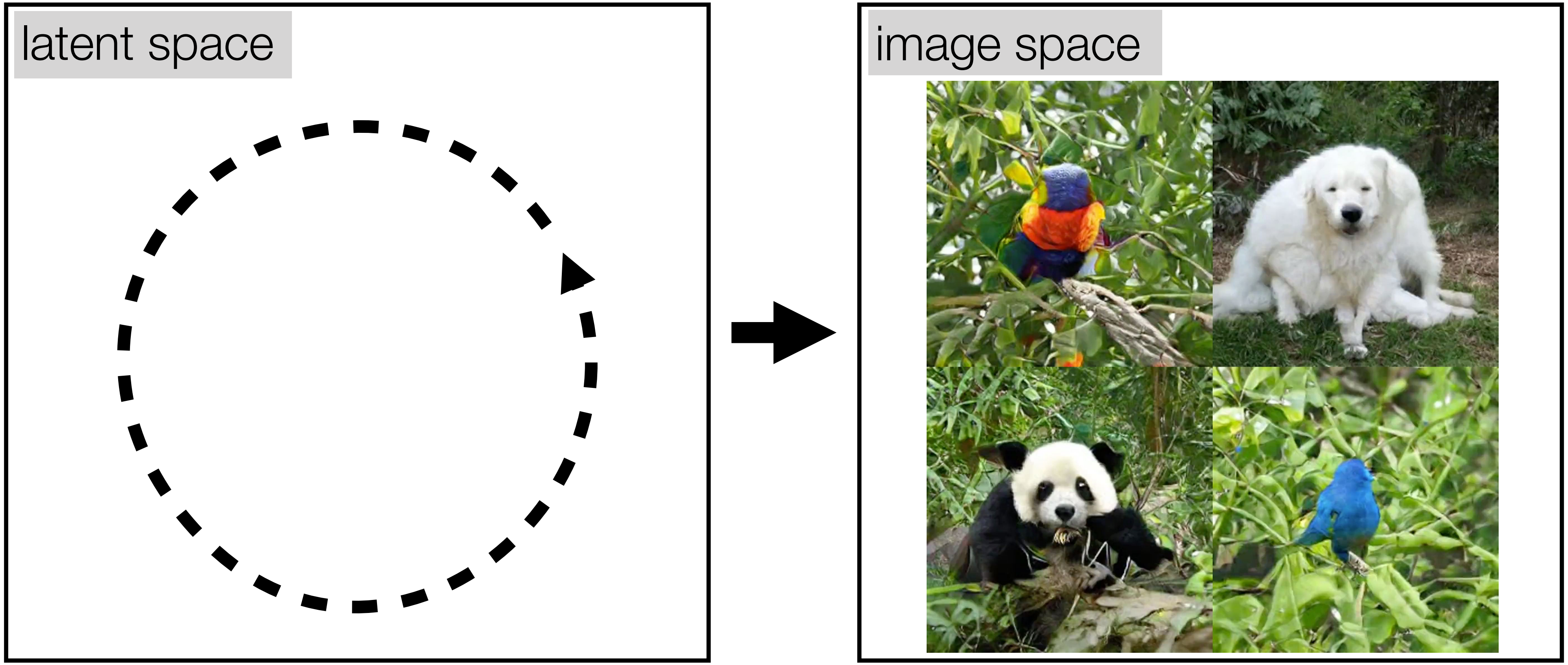
["StyleGAN", Karras, Laine, Aila, 2018]

StyleGANs (Karras et al., 2018)

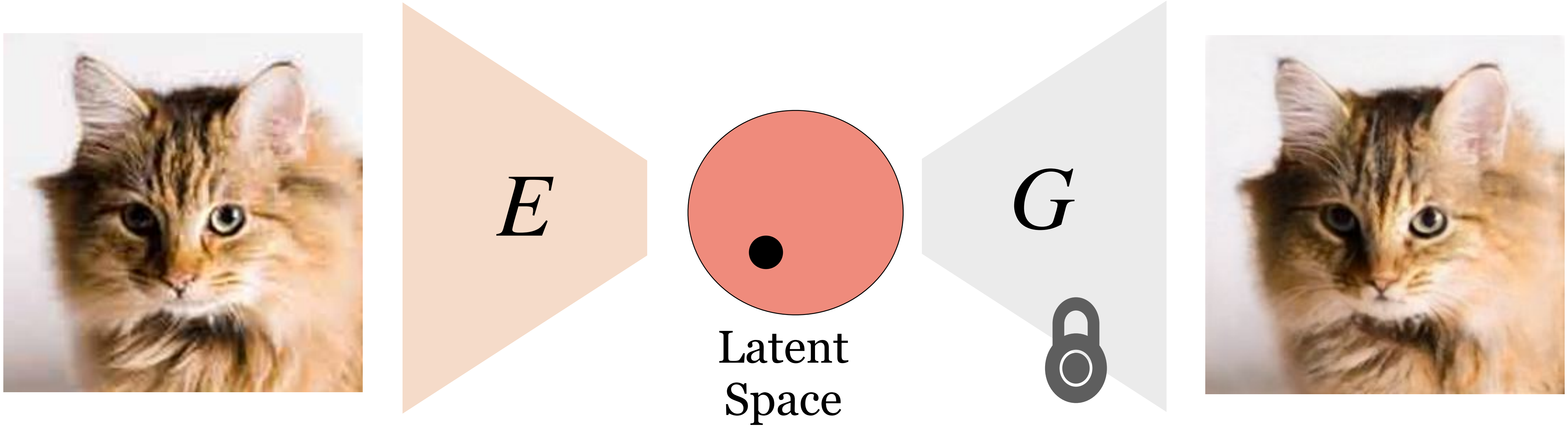



Latent space
(Gaussian)
$\mathbf{z}$
Data space
(Natural image manifold)
$\mathbf{x}$
[BigGAN, Brock et al. 2018]

Generative models organize the manifold of natural images



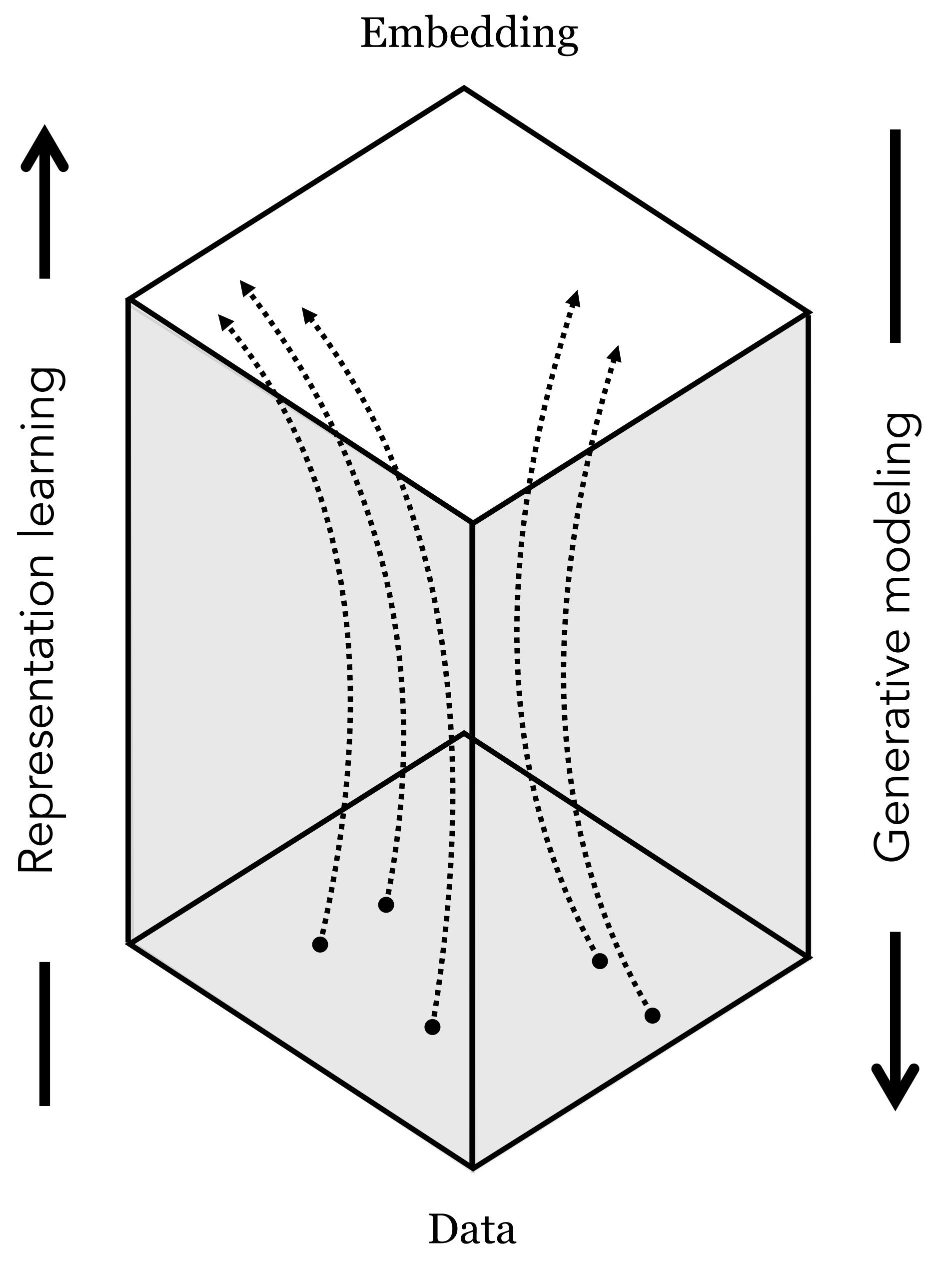
Projecting images into GAN latent space



$$w^* = \arg\min_{w} L_{\text{img}}(x, G(w)) + \lambda L_{\text{latent}}(w, E(x))$$

iGAN. Zhu et al. 2016; GAN Inversion: A Survey. Xia et al. 2021

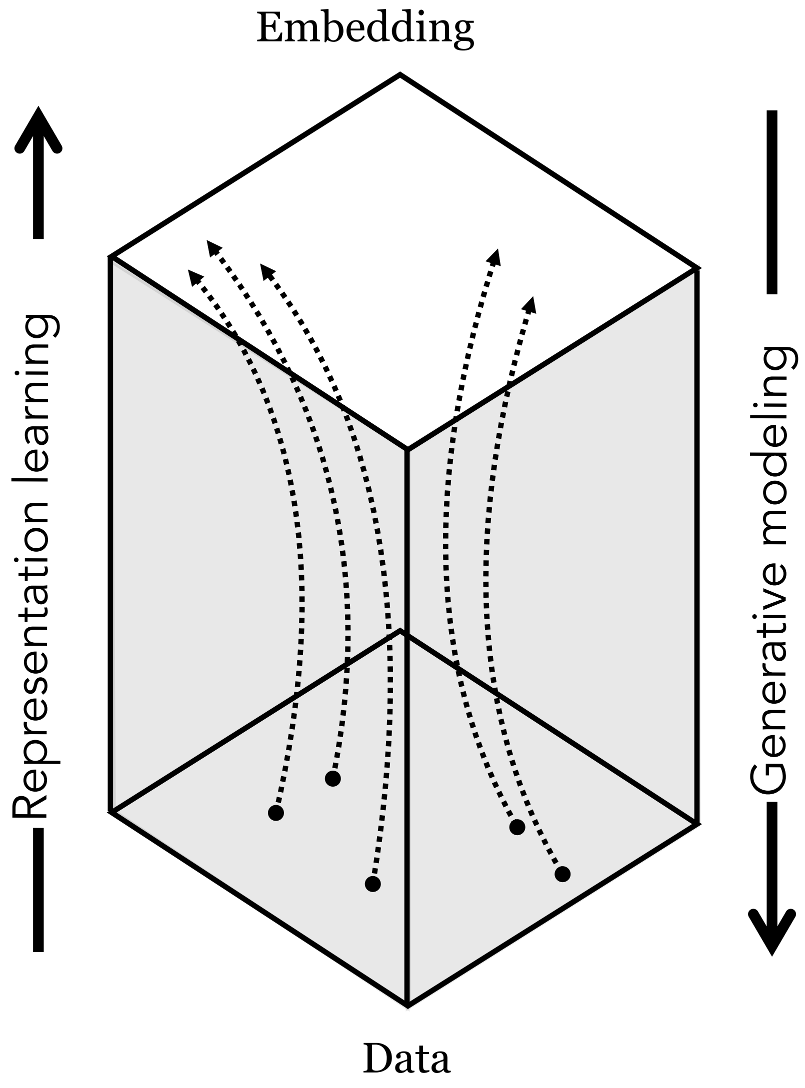
Representation Learning



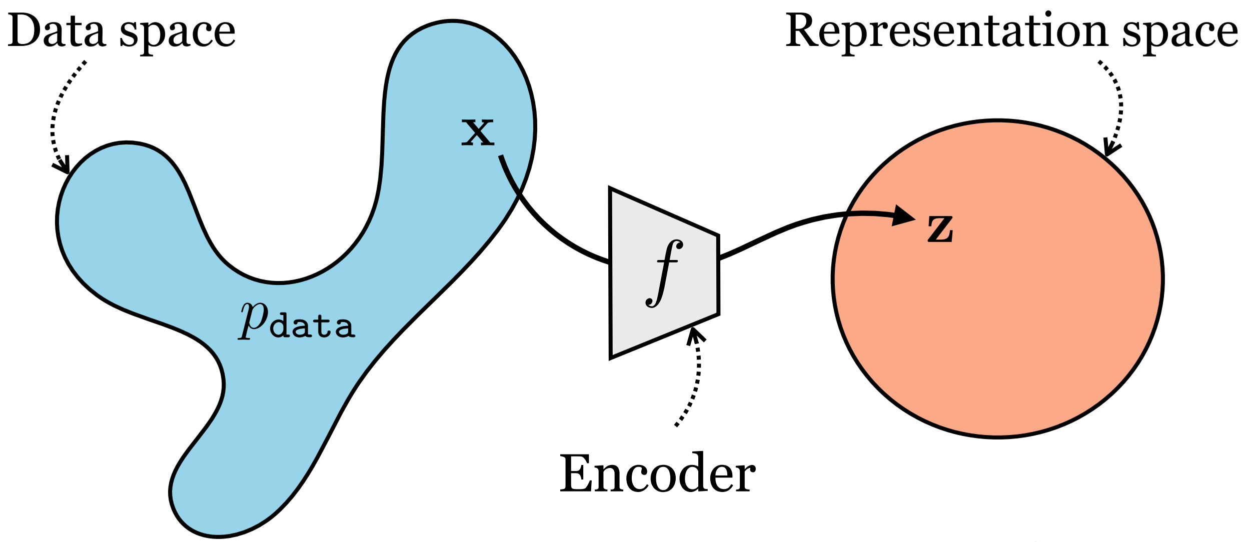
Generative modeling vs Representation learning

Representation learning:
mapping data to abstract representations (analysis)

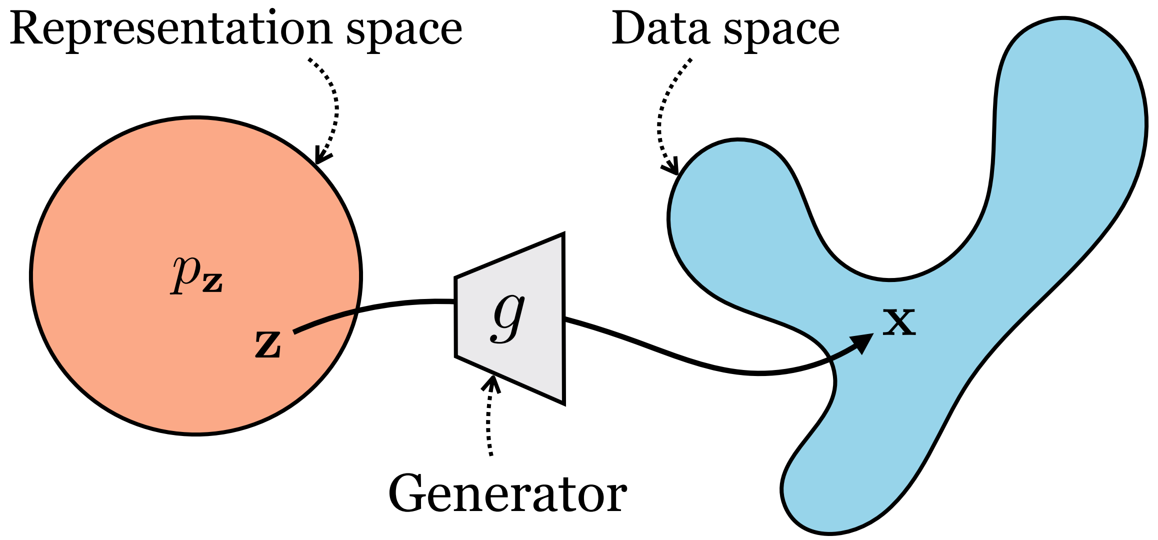
Generative modeling:
mapping abstract representations to data (synthesis)

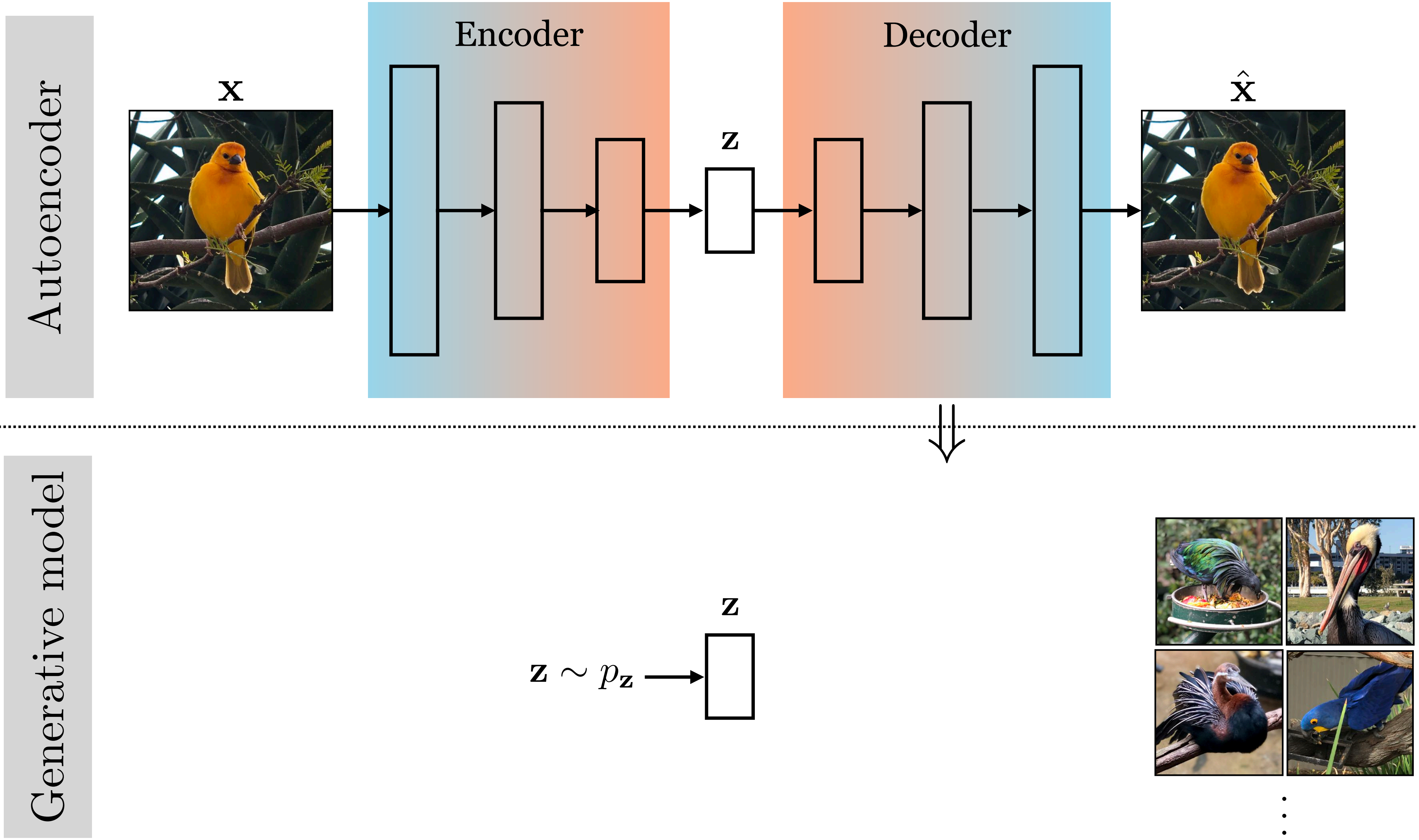


Representation learning

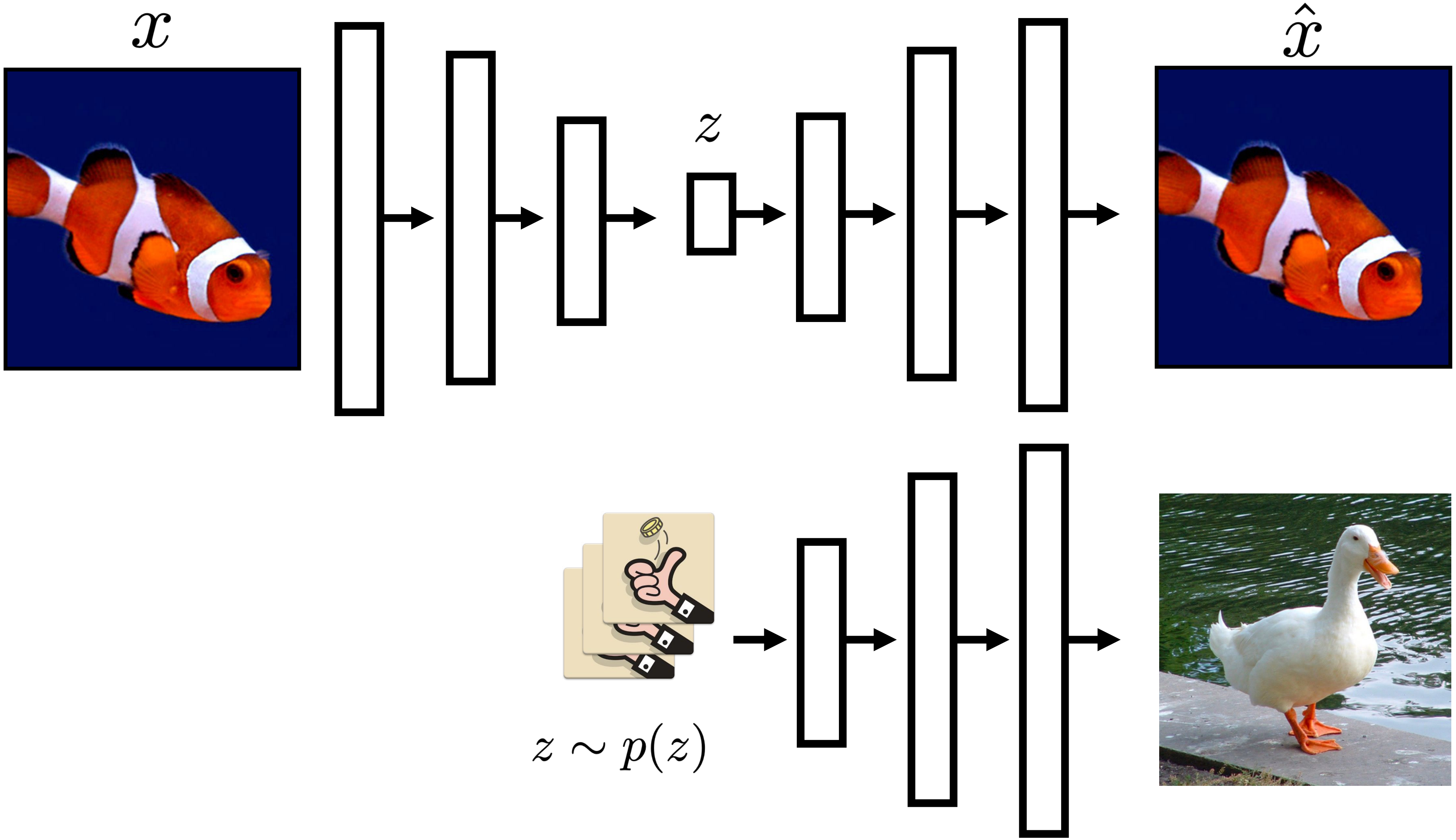


Generative modeling



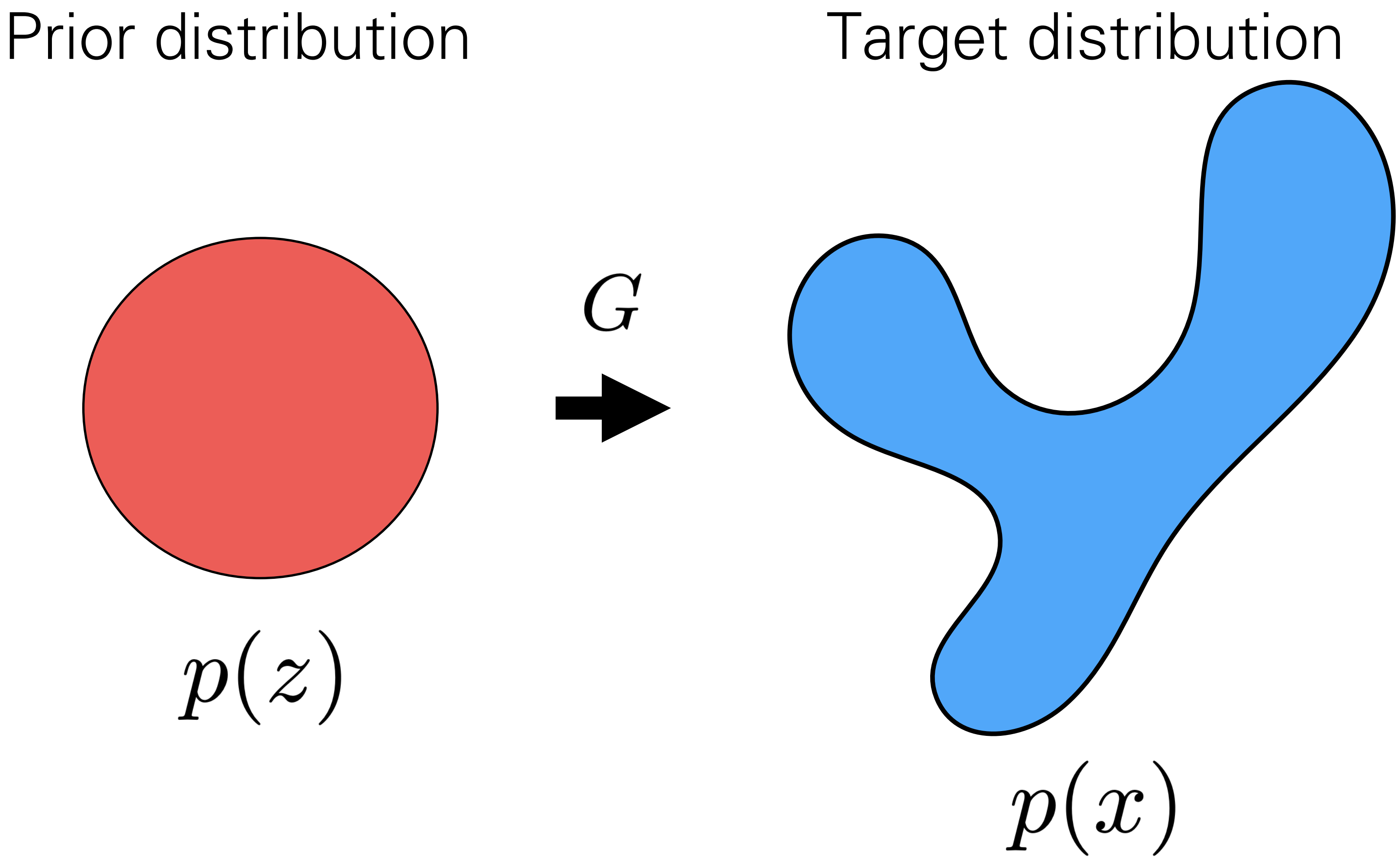

Autoencoder
Encoder
Decoder
x
z
x̂
Generative model
z
z ∼ pz

Autoencoder → Generative model

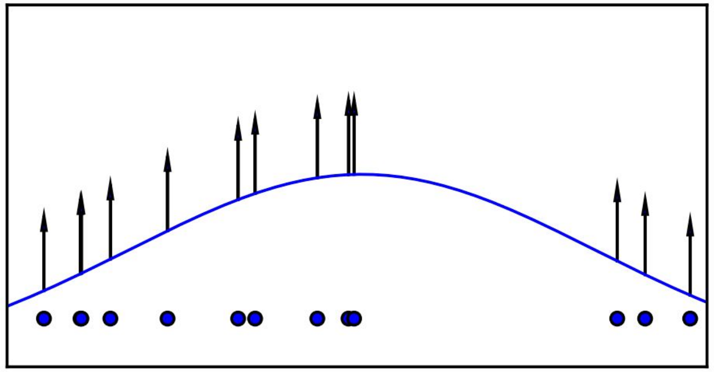


Variational Autoencoder (VAE)

[Kingma & Welling, 2014; Rezende, Mohamed, Wierstra 2014]

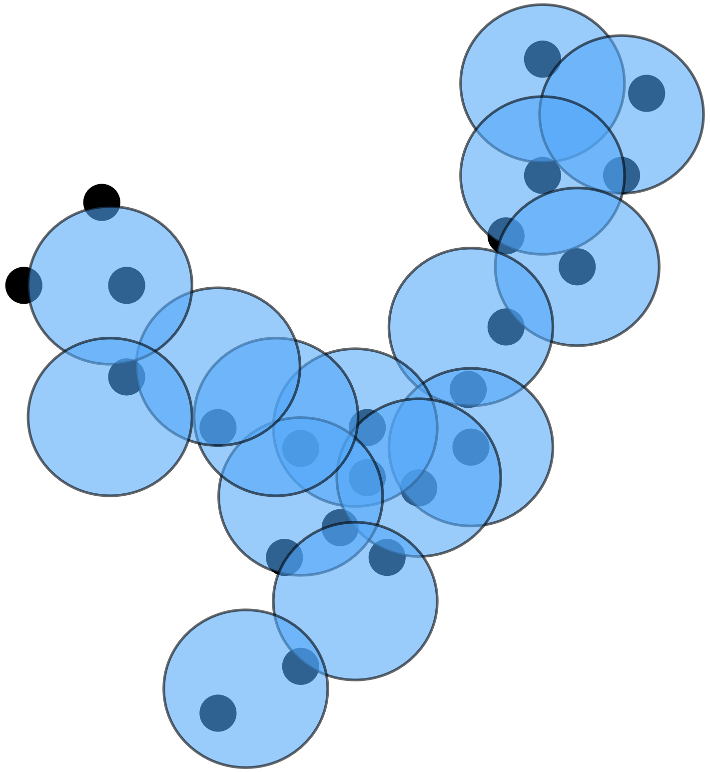


Mixture of Gaussians



$$p_\theta(x) = \sum_{i=1}^{k} w_i \mathcal{N}(x; u_i, \Sigma_i)$$

Target distribution

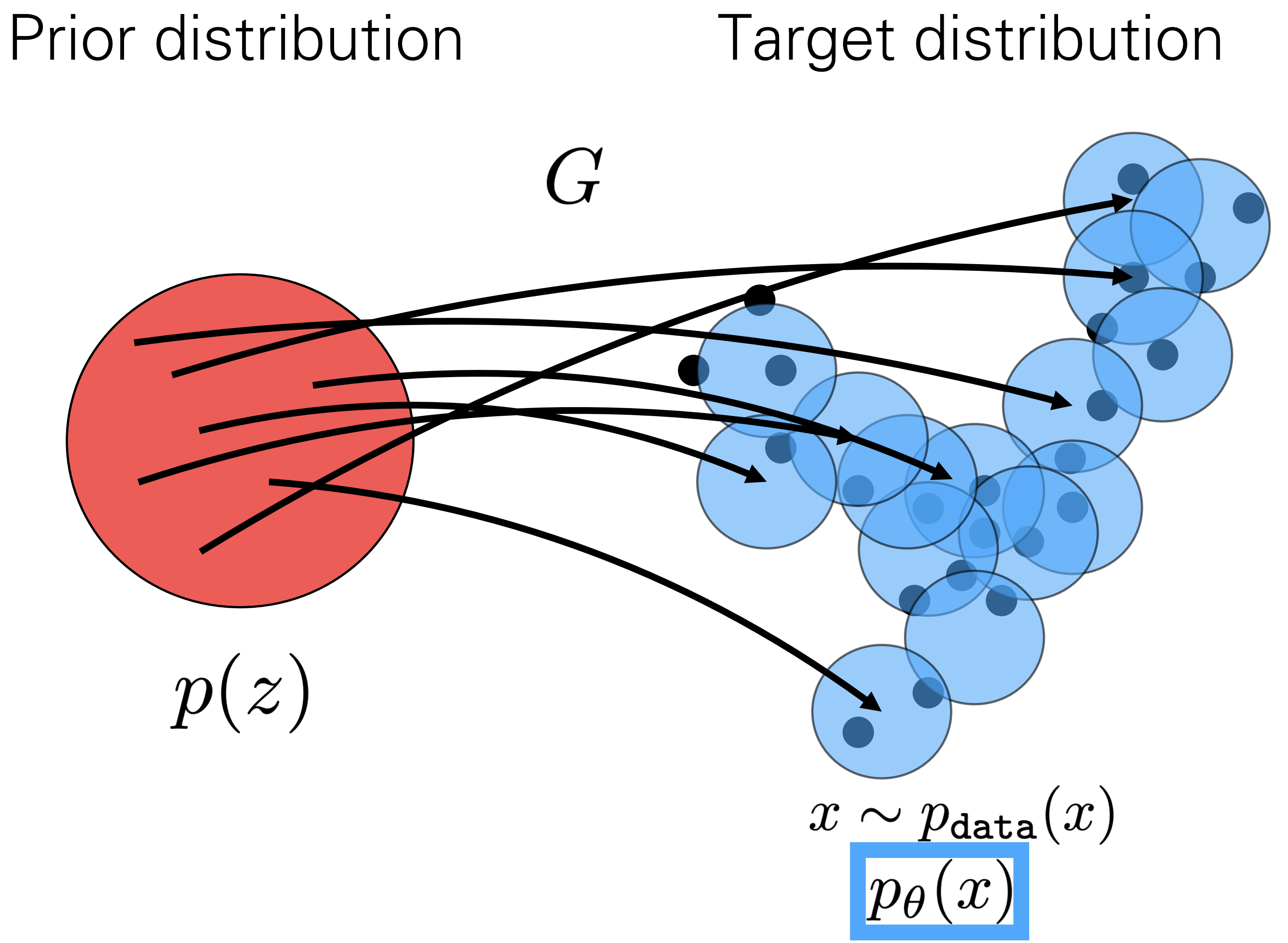


$x \sim p_{\texttt{data}}(x)$

$\boxed{p_\theta(x)}$

Variational Autoencoder (VAE)

[Kingma & Welling, 2014; Rezende, Mohamed, Wierstra 2014]



Density model:

$$p_\theta(x) = \int p(x|z;\theta)p(z)dz$$

$$p(x|z;\theta) \sim \mathcal{N}(x; G^{\mu}_{\theta}(x), G^{\sigma}_{\theta}(x))$$

Sampling:

$$z \sim p(z) \quad \epsilon \sim \mathcal{N}(0,1)$$

$$x = G^{\mu}_{\theta}(z) + G^{\sigma}_{\theta}(z)\epsilon$$

Variational Autoencoder (VAE)

Data $\rightarrow$

Learner

Objective

$$\max_{\theta} \mathbb{E}_{x \sim p_{\texttt{data}}} [\log p_\theta(x)]$$

Hypothesis space

$$\boxed{p_\theta(x) = \int p(x|z;\theta)p(z)dz}$$

$$x = G_\theta^\mu(z) + G_\theta^\sigma(z)\epsilon$$

$\longrightarrow$

Density

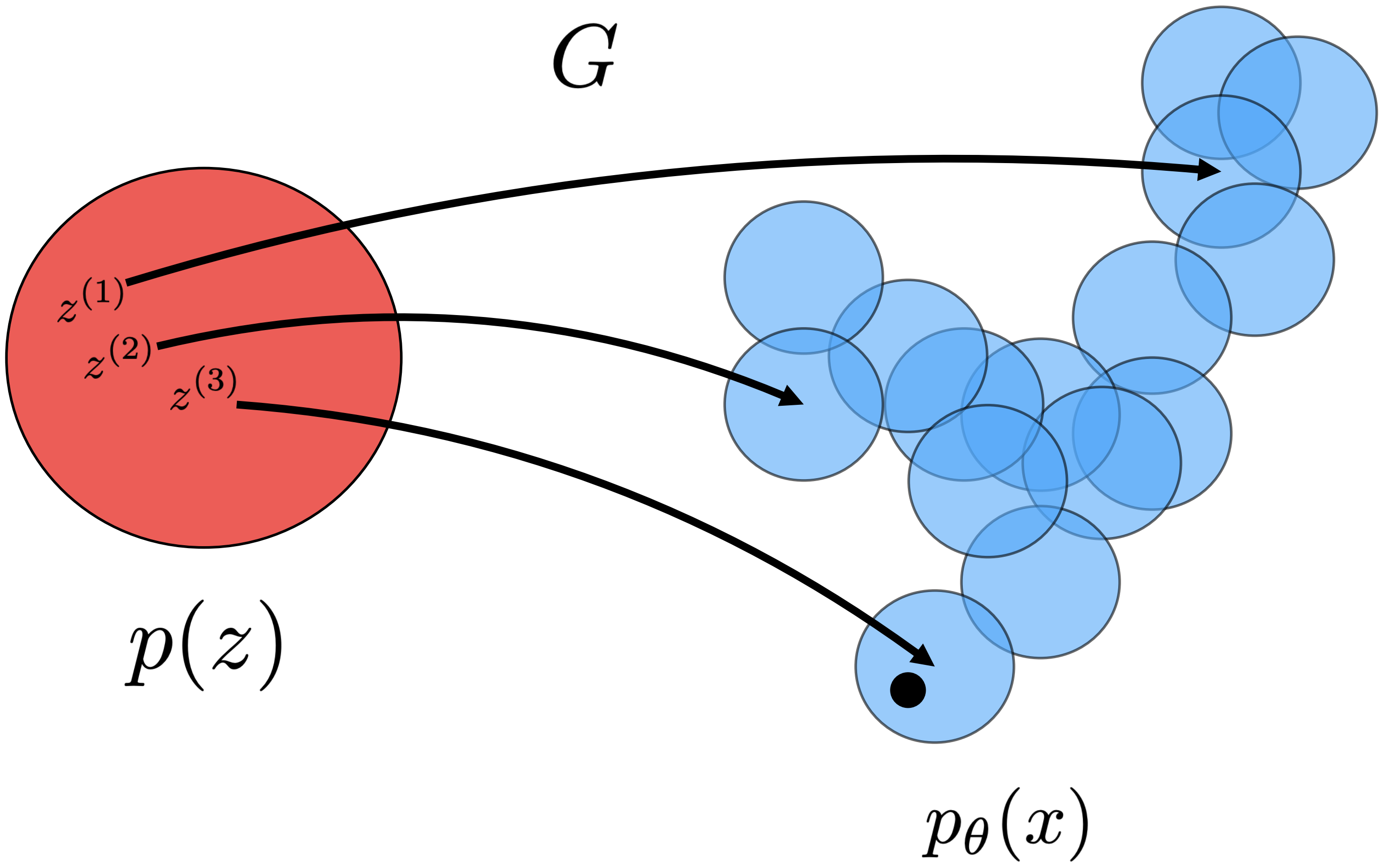
$$p_\theta : \mathcal{X} \to [0, 1]$$

Sampler

$$G_\theta : \mathcal{Z} \to \mathcal{X}$$

Prior distribution

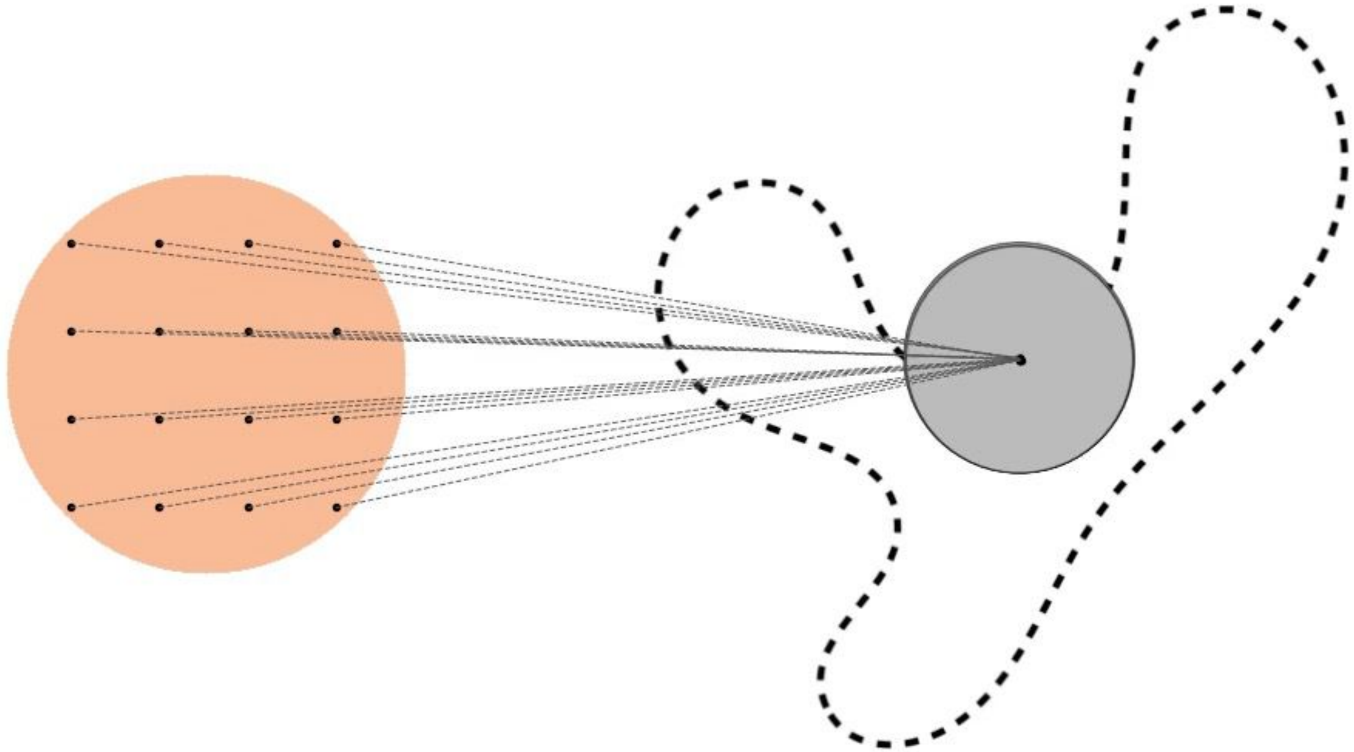
Current model of target distribution

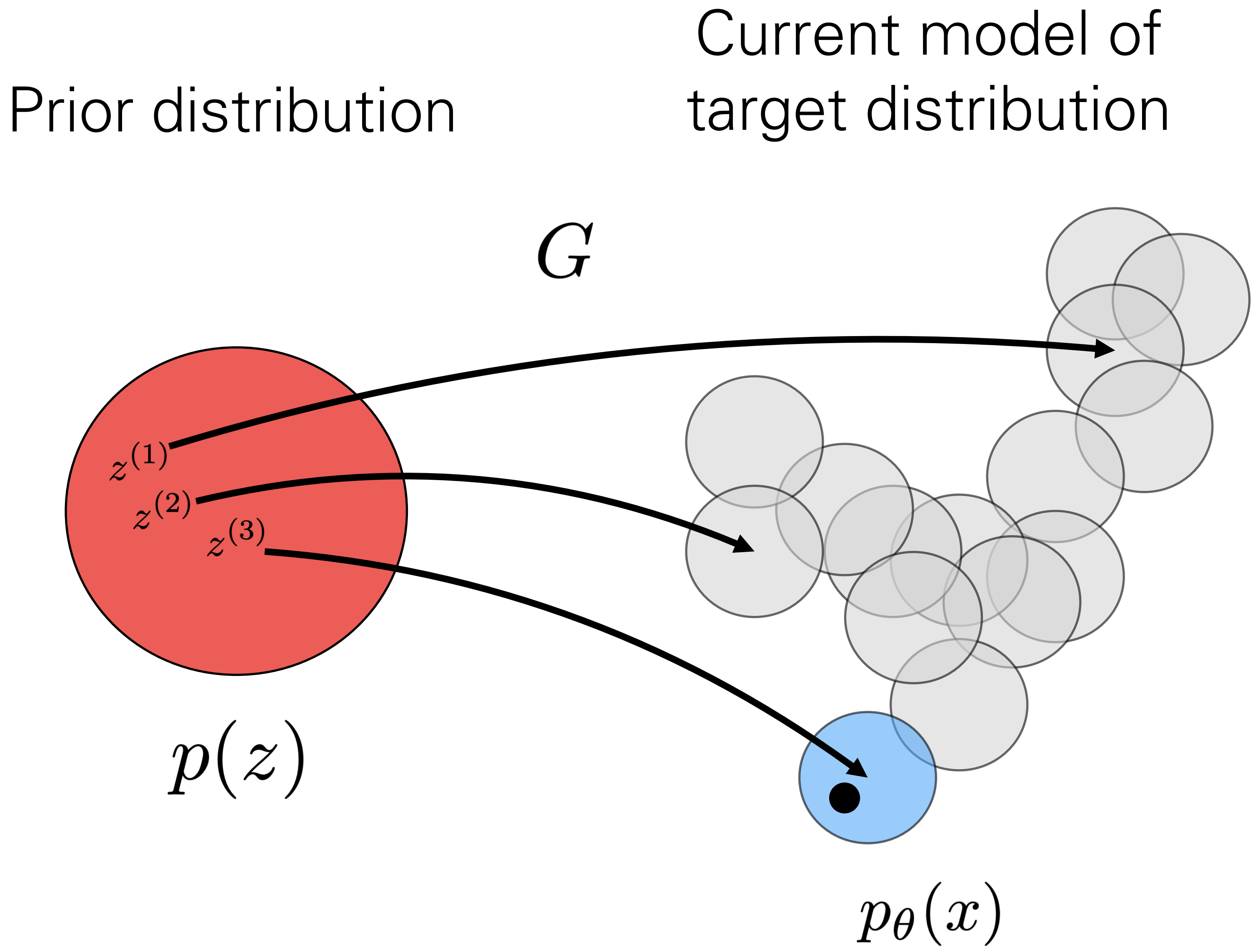


In order to optimize our model, we need to measure the likelihood it assigns to each datapoint x

$$p_\theta(x) = \int p(x|z;\theta)p(z)dz$$

$$= p(x|z^{(1)})p(z^{(1)})dz + p(x|z^{(2)})p(z^{(2)})dz + p(x|z^{(3)})p(z^{(3)})dz + \ldots$$





In order to optimize our model, we need to measure the likelihood it assigns to each datapoint x

$$p_\theta(x) = \int p(x|z;\theta)p(z)dz$$

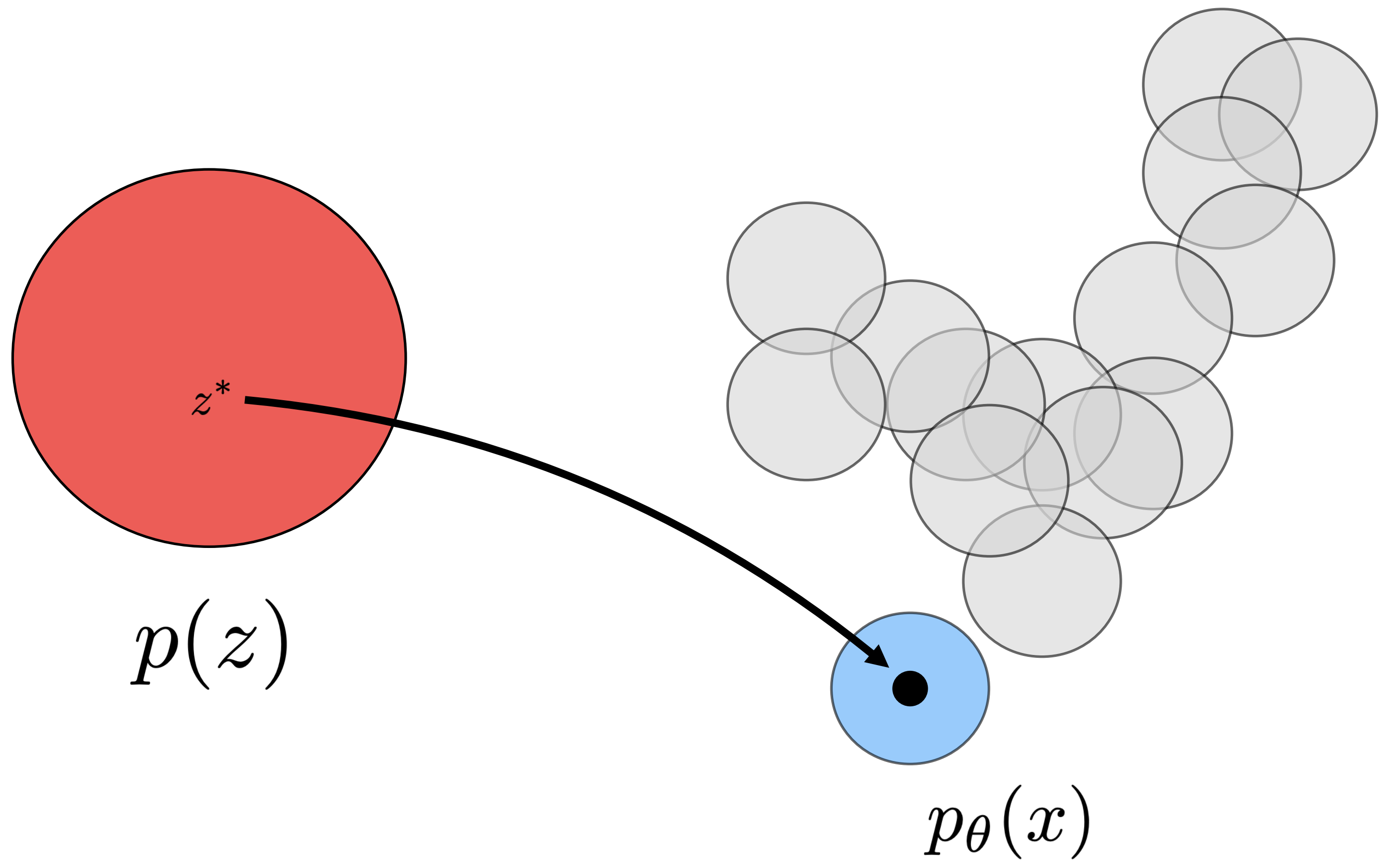
$$= \sim 0+$$

$$\sim 0+$$

$$p(x|z^{(3)})p(z^{(3)})dz + ...$$

Current model of target distribution

Prior distribution

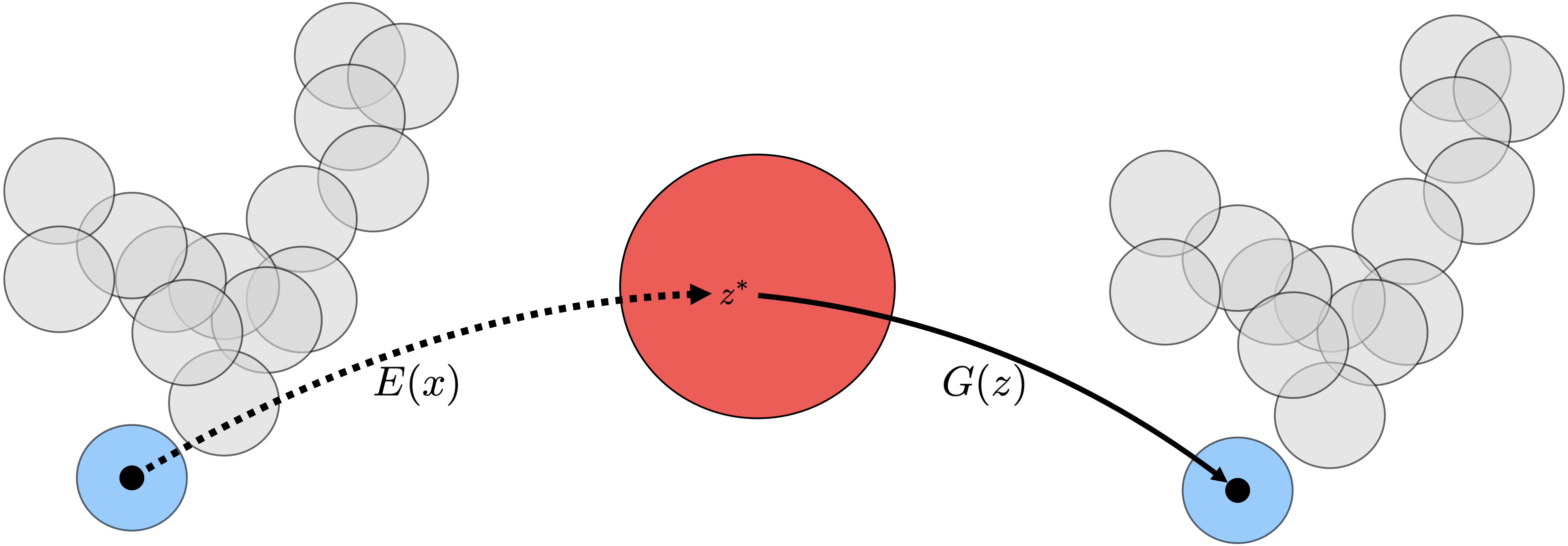


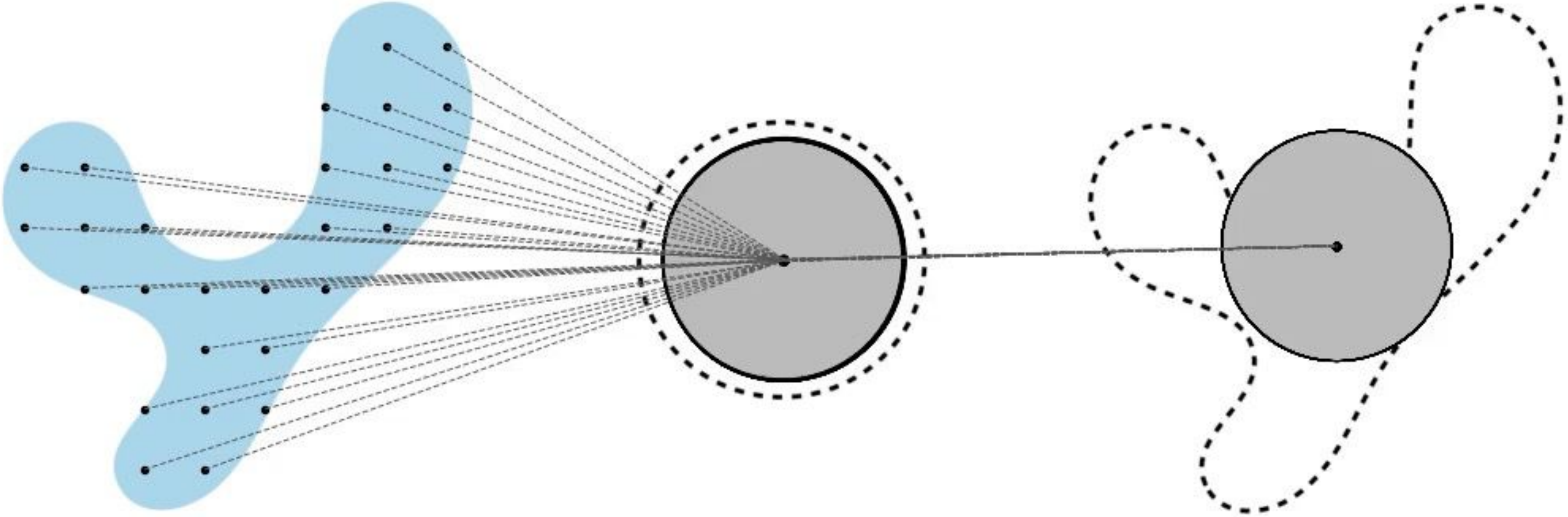
If only we knew z*, we wouldn't need the integral…

$$p_\theta(x) = \int p(x|z;\theta)p(z)dz$$

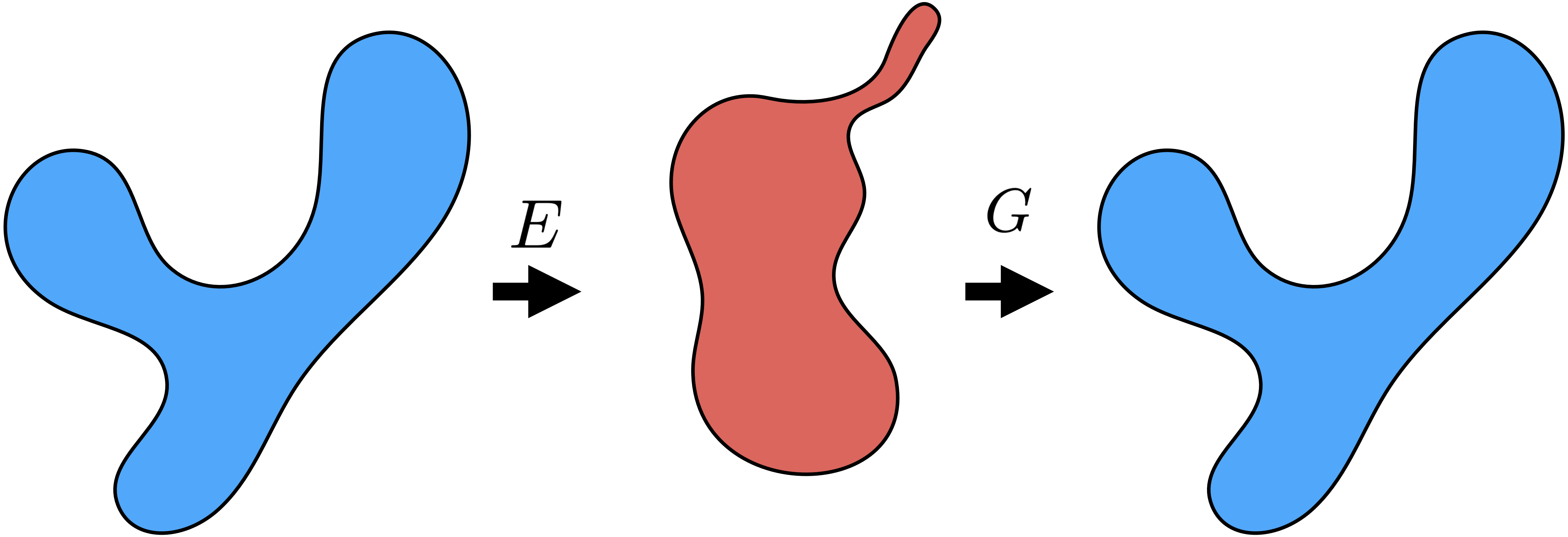
$$\approx p(x|z^*;\theta)p(z^*)$$

Autoencoder!



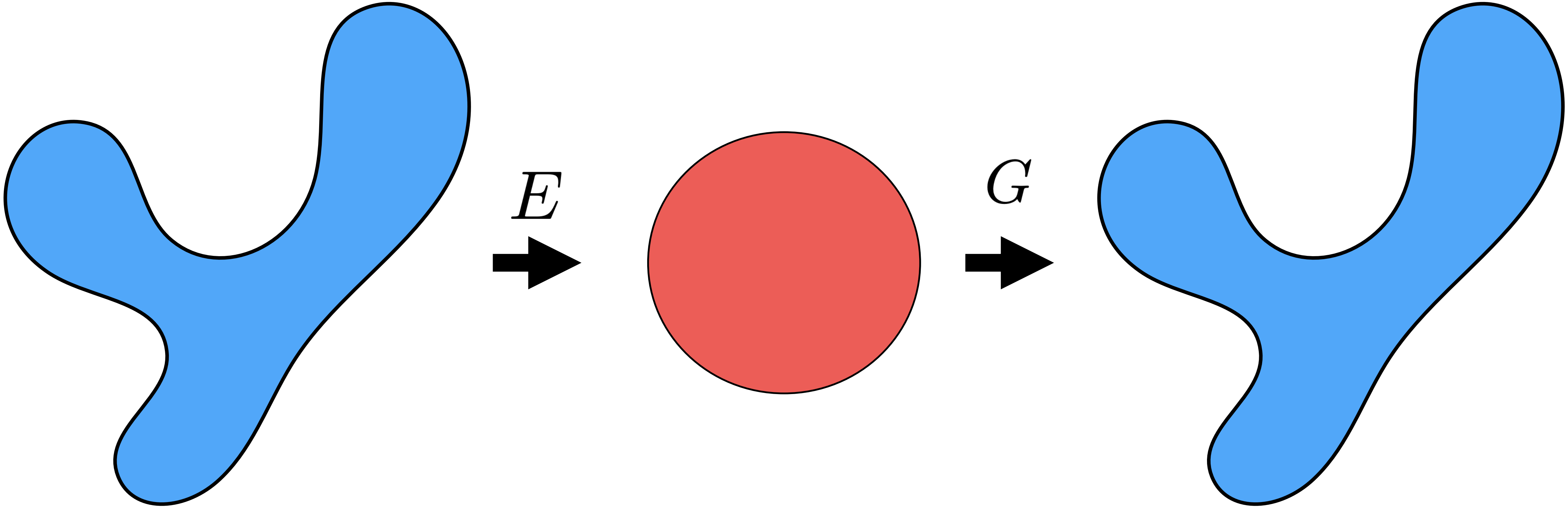


Classical Autoencoder



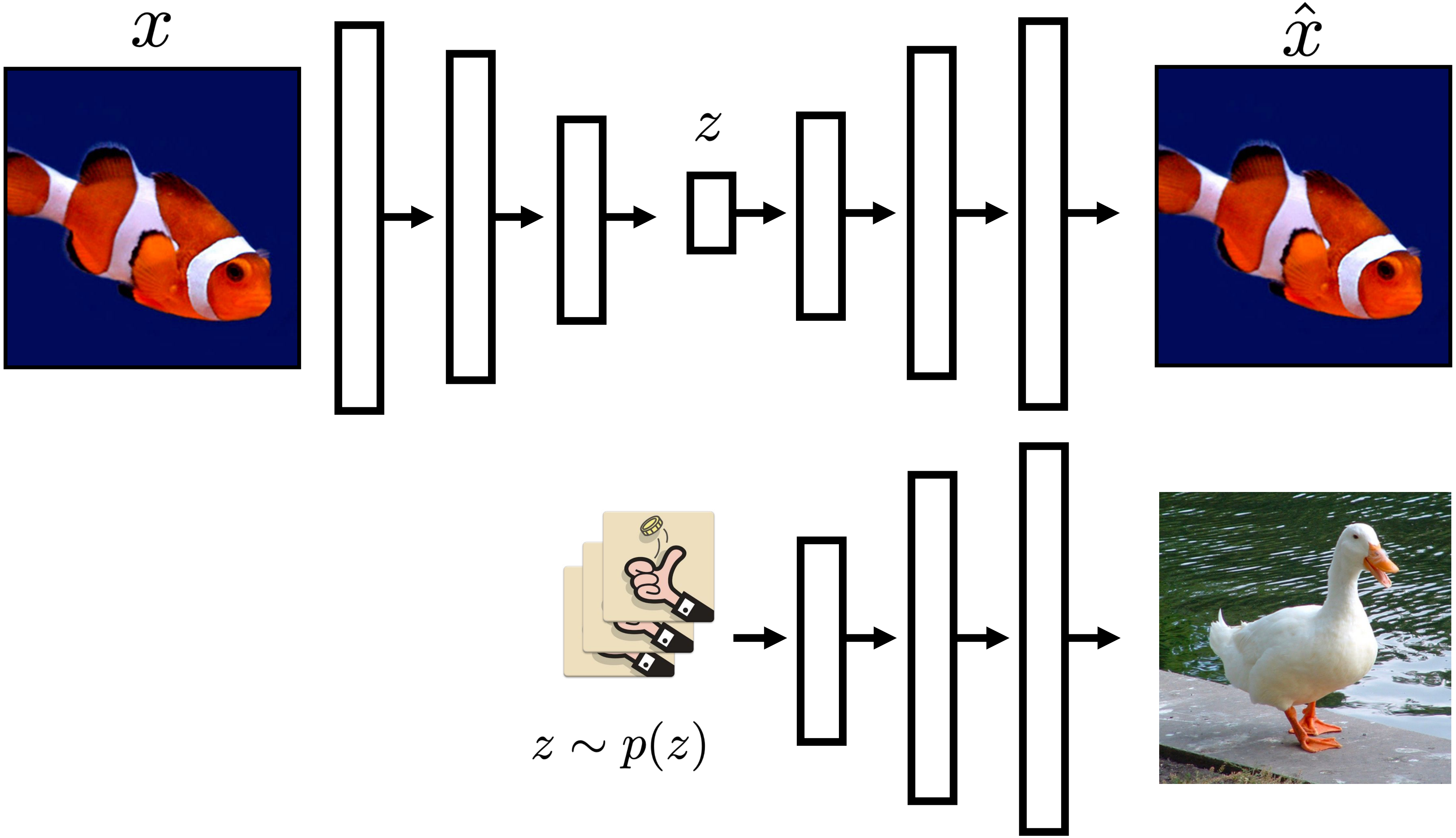
$$\arg\min_{G,E} \mathbb{E}_x[\|G(E(x)) - x\|_2^2]$$

Variational Autoencoder



$$\underset{G,E}{\arg\min}\, \mathbb{E}_{x,\epsilon}[\|G(E(x+\epsilon)) - x\|_2^2 + \|E(x+\epsilon)\|_2^2]$$

Variational Autoencoder



Gaussian VAEs 2013

Sample $z \sim \mathcal{N}(0, I)$ and compute $y_\Phi(z)$

[Alec Radford]

Vector Quantized VAEs (VQ-VAE) 2019

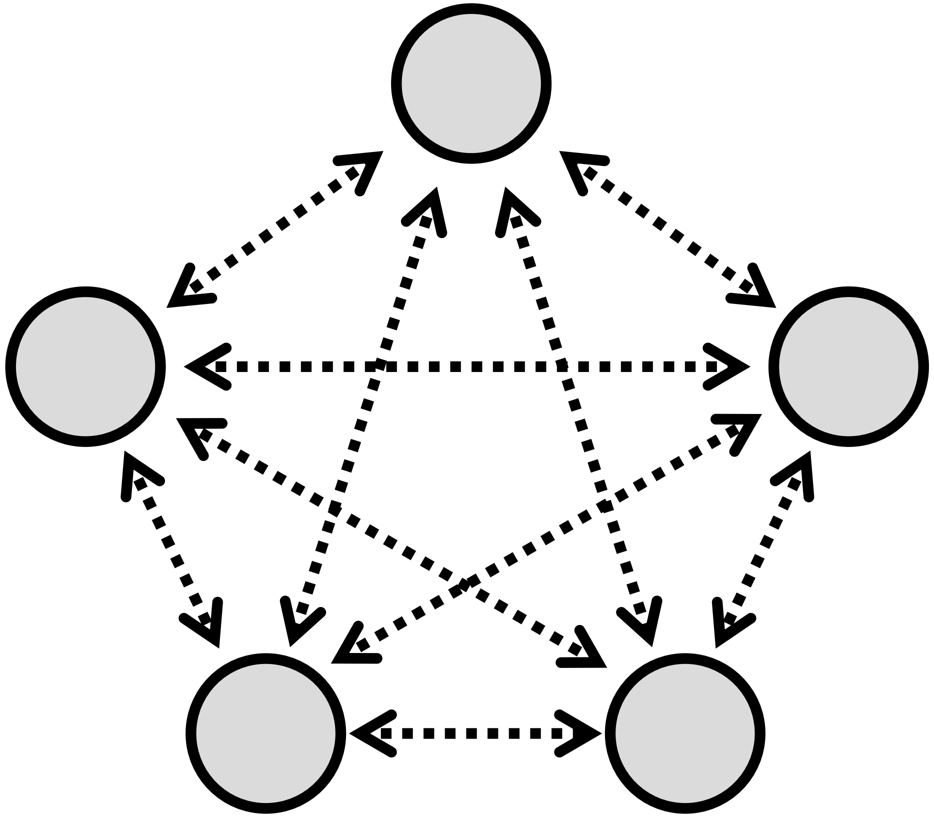
VQ-VAE-2, Razavi et al., NeurIPS 2019

Vector Quantized VAEs (VQ-VAE) 2019

Figure 1: Class-conditional 256x256 image samples from a two-level model trained on ImageNet.

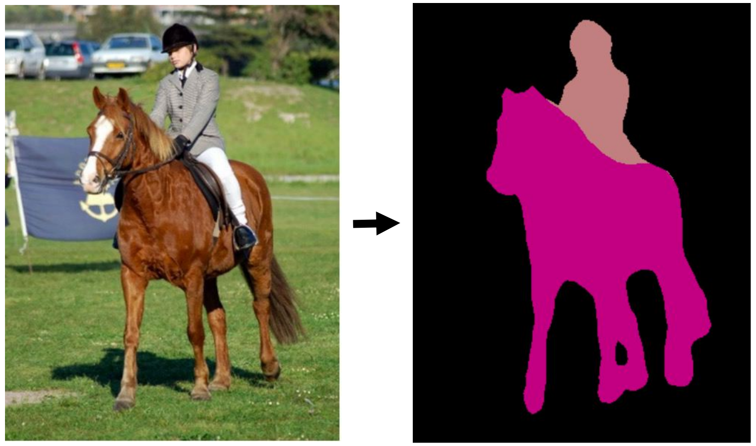
VQ-VAE-2, Razavi et al., NeurIPS 2019

Data Translation



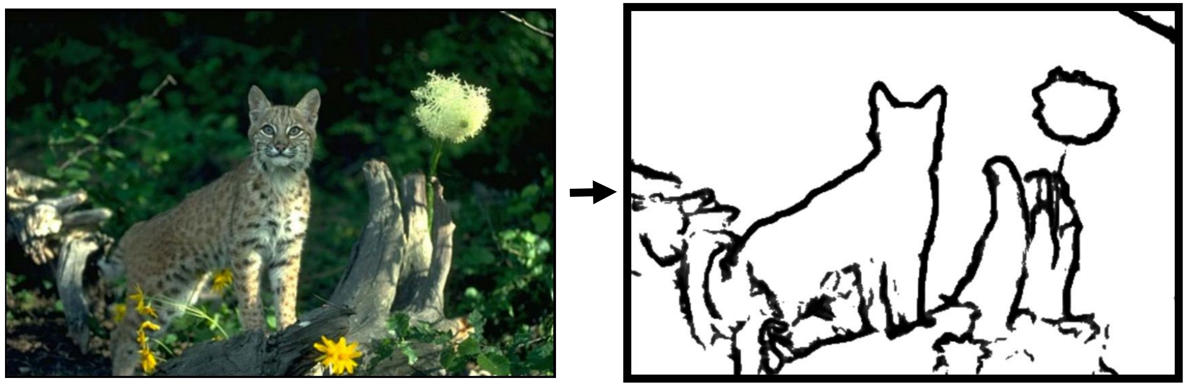
Data translation problems ("structured prediction")

Semantic segmentation



[Long et al. 2015, …]

Edge detection



[Xie et al. 2015, …]

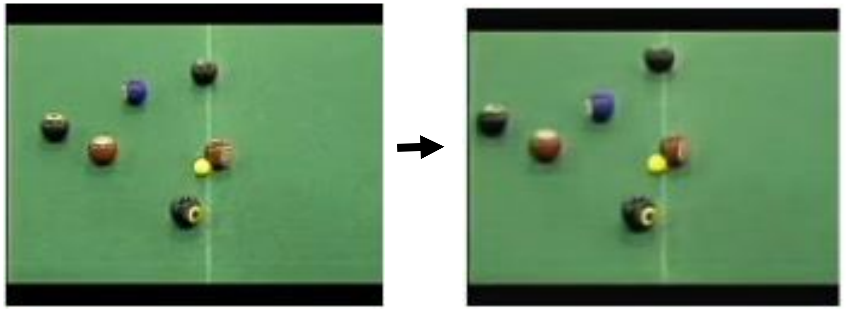
Text-to-image

"this small bird has a pink

breast and crown…"

[Reed et al. 2014, …]

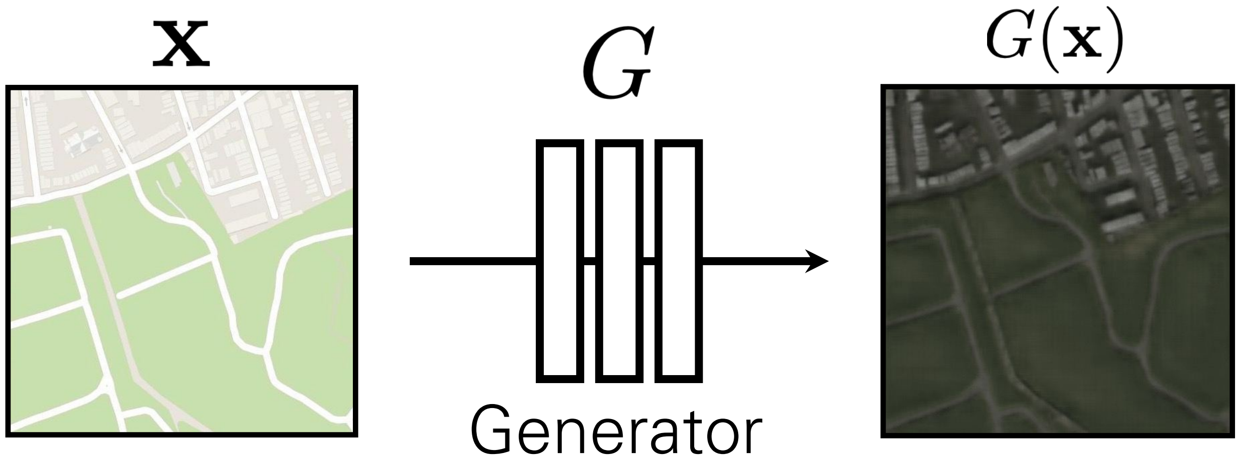
Future frame prediction



[Mathieu et al. 2016, …]

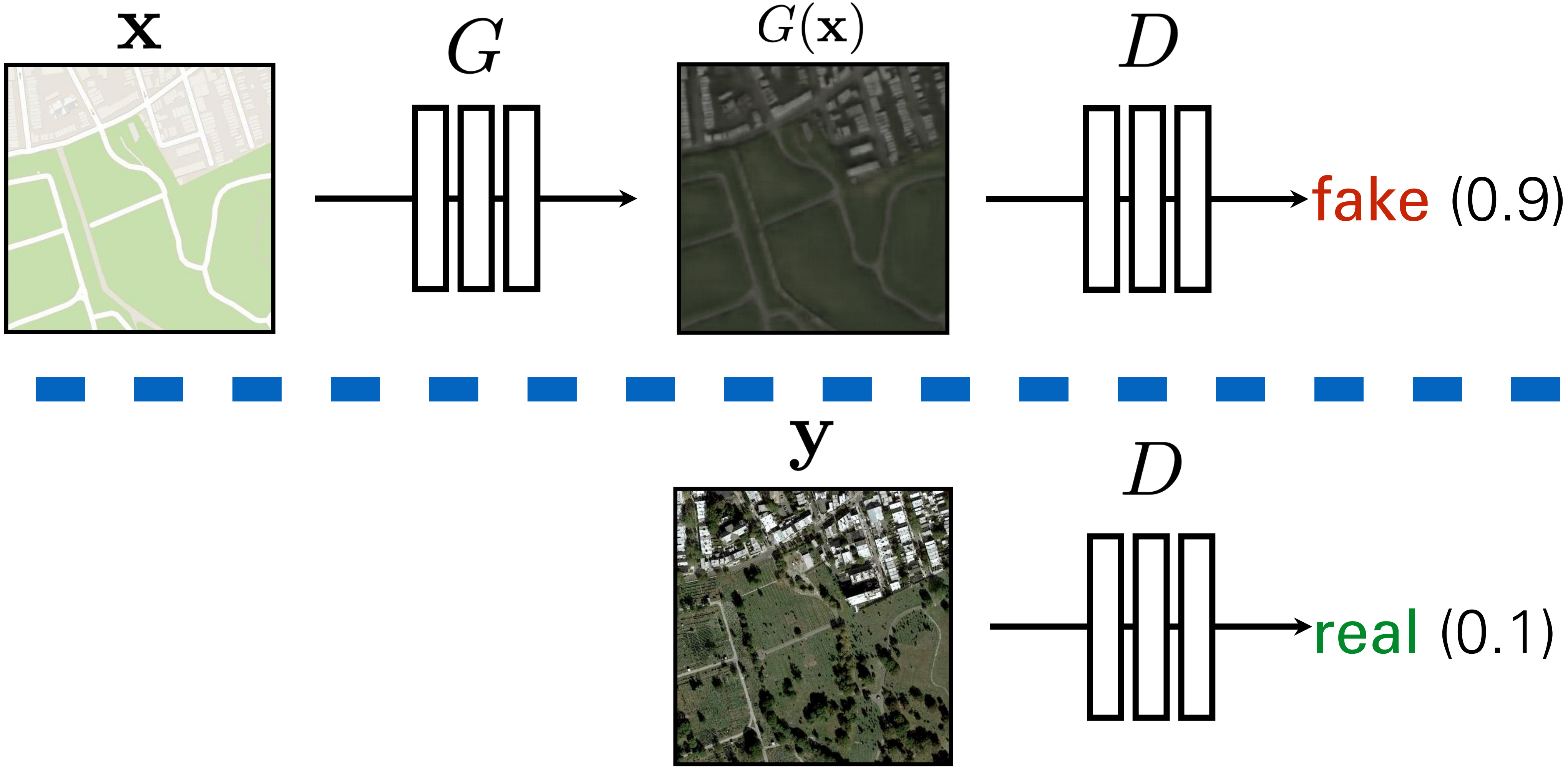
Input

Deep net output

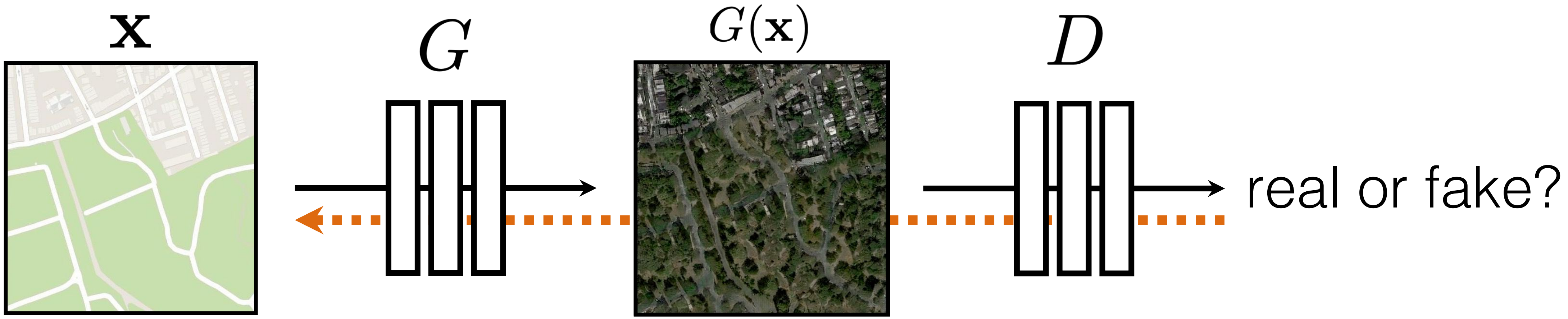

$\mathbf{x}$
G
Generator
$G(\mathbf{x})$

G tries to synthesize fake images that fool D

D tries to identify the fakes

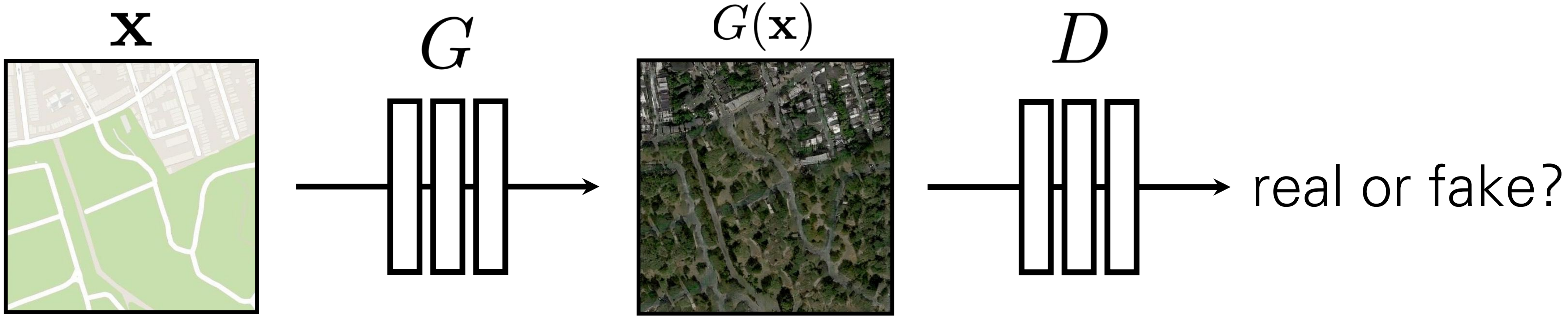


$$\arg\max_{D} \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \boxed{\log D(G(\mathbf{x}))} \;+\; \boxed{\log(1 - D(\mathbf{y}))} \;]$$



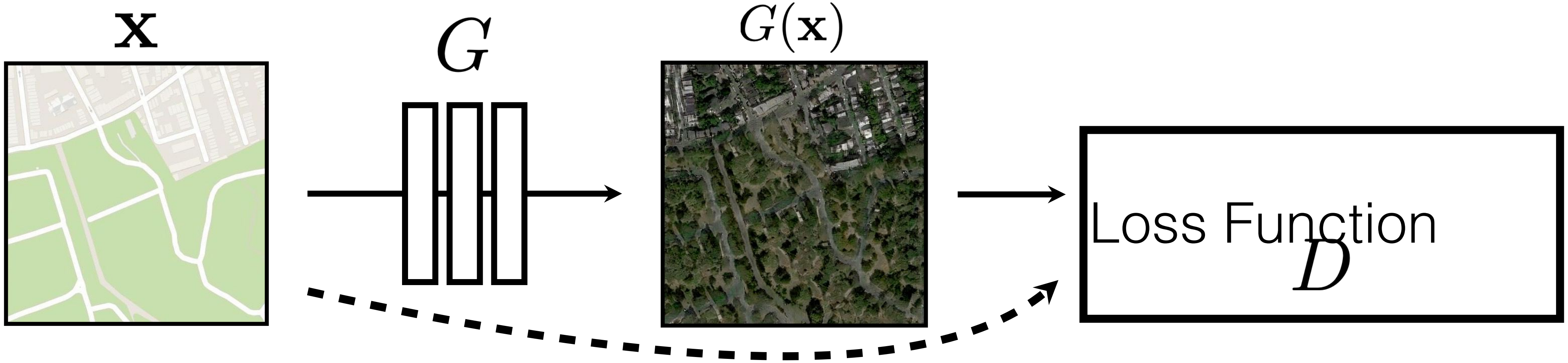
G tries to synthesize fake images that *fool* D:

$$\arg \boxed{\min_{G}} \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(G(\mathbf{x})) \quad + \quad \log(1 - D(\mathbf{y})) \;]$$



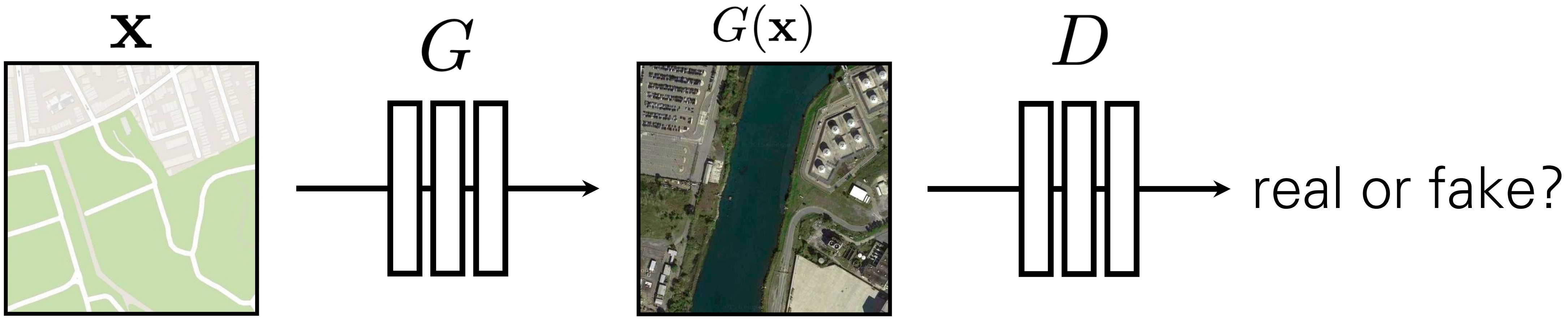
G tries to synthesize fake images that *fool* the *best* D:

$$\arg \boxed{\min_G} \boxed{\max_D} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) \quad + \quad \log(1 - D(\mathbf{y}))\ ]$$

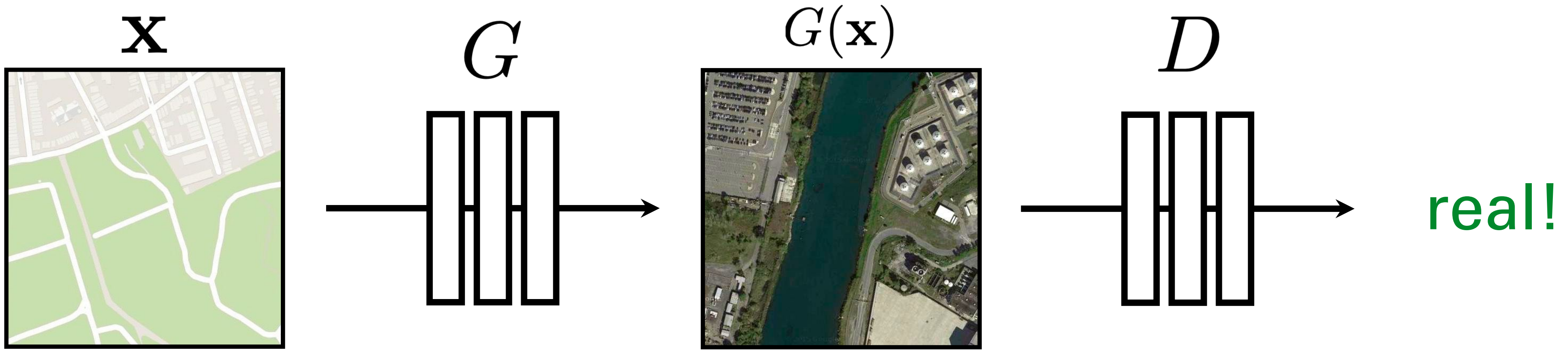


G′s perspective: **D** is a loss function.

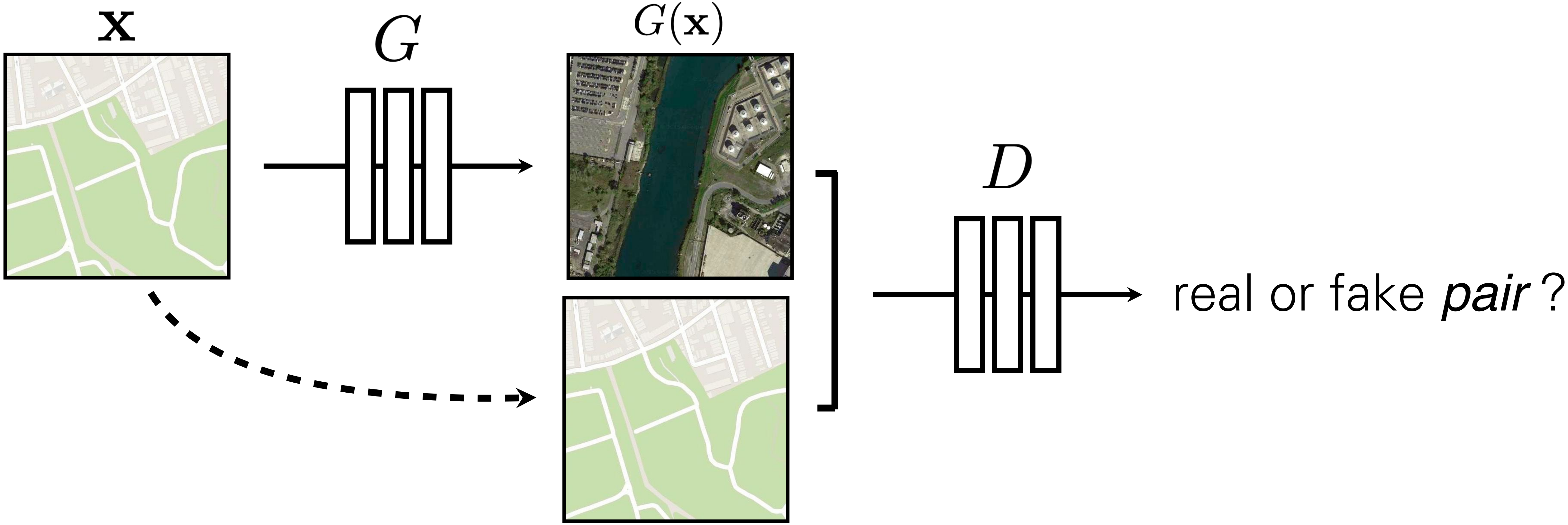
Rather than being hand-designed, it is *learned* and *highly structured*.



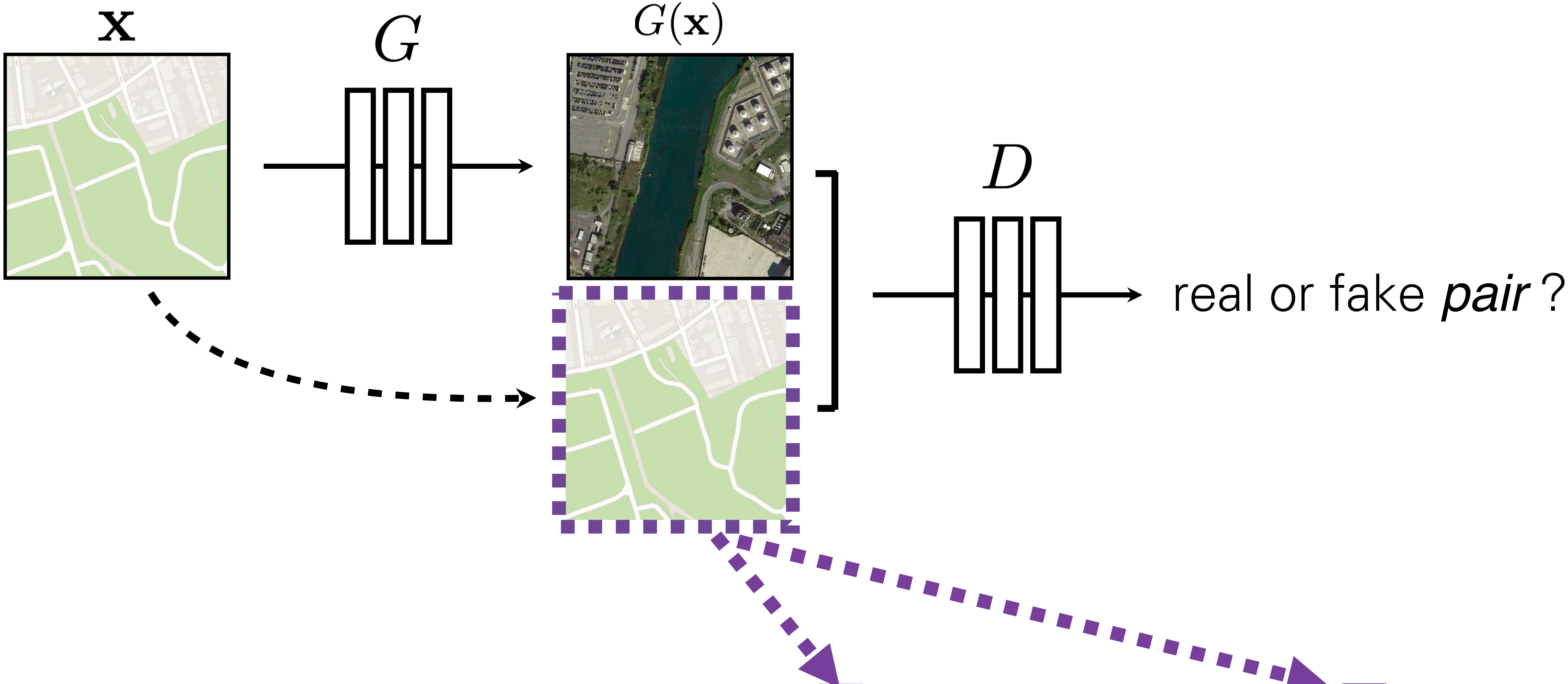
$$\arg\min_{G}\max_{D}\ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) \quad + \quad \log(1 - D(\mathbf{y}))\ \ ]$$



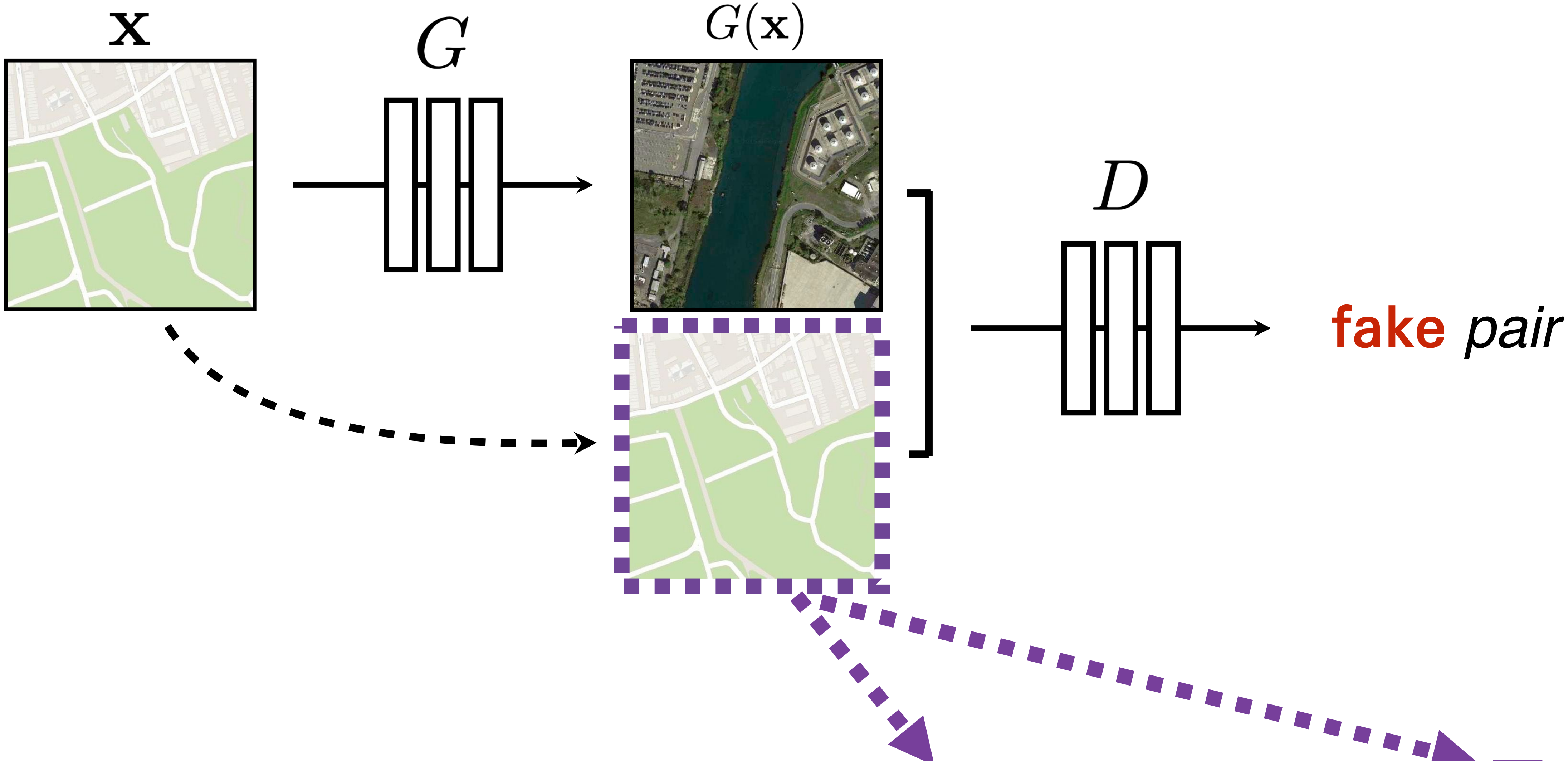
$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) \quad + \quad \log(1 - D(\mathbf{y})) \ ]$$



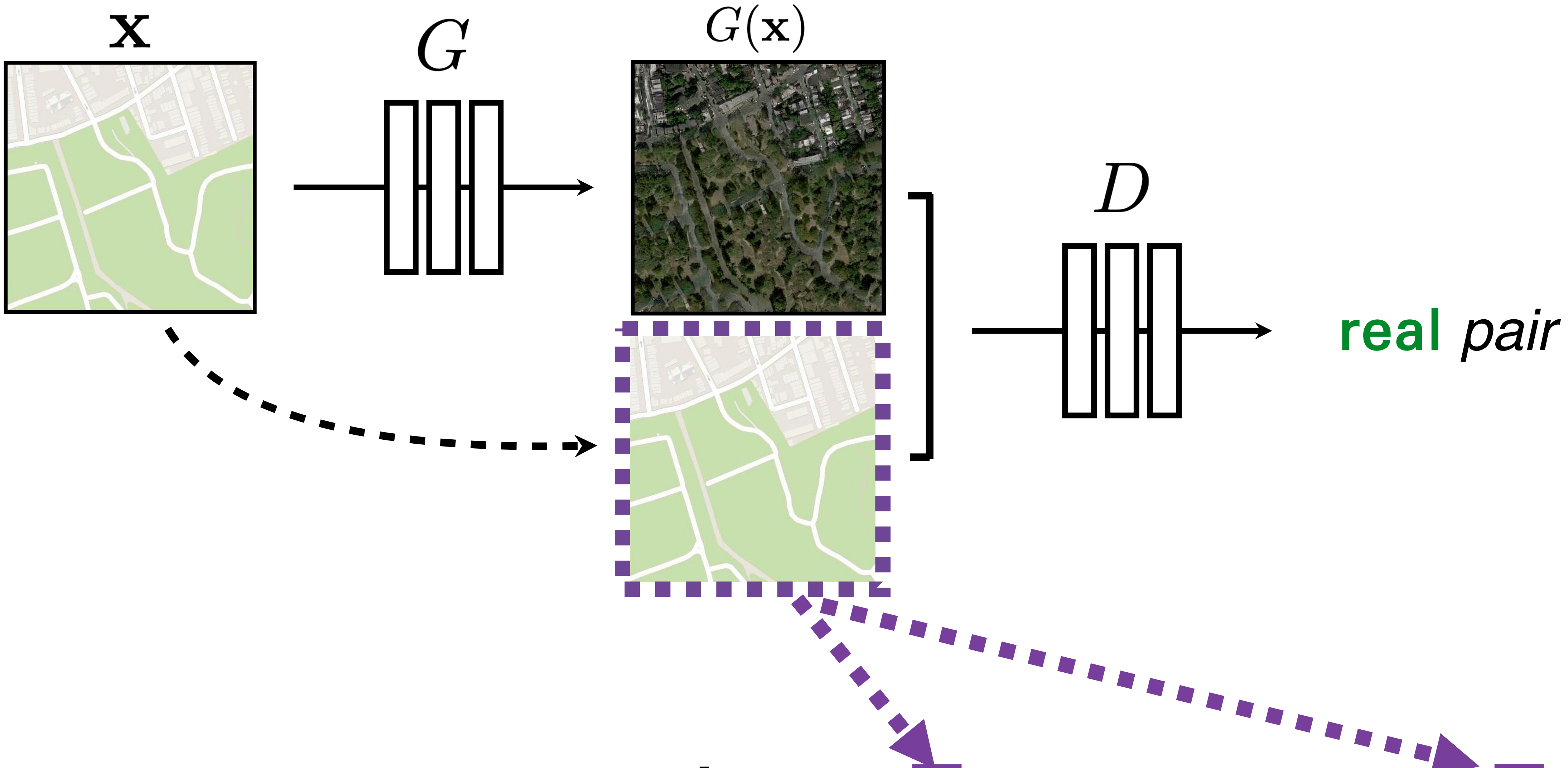
$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) \quad + \quad \log(1 - D(\mathbf{y}))\ ]$$



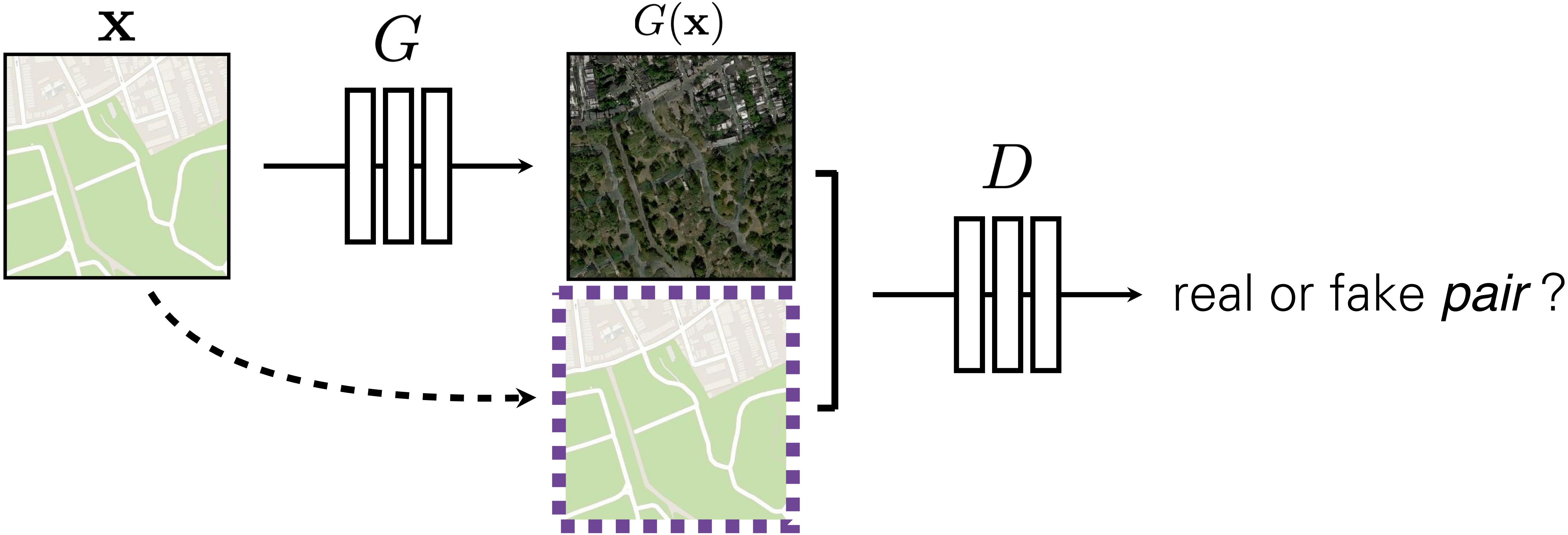
$$\arg\min_{G}\max_{D}\ \mathbb{E}_{\mathbf{x},\mathbf{y}}\left[\ \log D(\boxed{\mathbf{x}}, G(\mathbf{x})) + \log(1 - D(\boxed{\mathbf{x}}, \mathbf{y}))\ \right]$$



$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(\boxed{\mathbf{x}}, G(\mathbf{x})) + \log(1 - D(\boxed{\mathbf{x}}, \mathbf{y})) \;]$$



$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}\left[\ \log D(\boxed{\mathbf{x}}, G(\mathbf{x})) + \log(1 - D(\boxed{\mathbf{x}}, \mathbf{y}))\ \right]$$



$$\boxed{\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\;\log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y}))\;]}$$

Training Details: Loss function

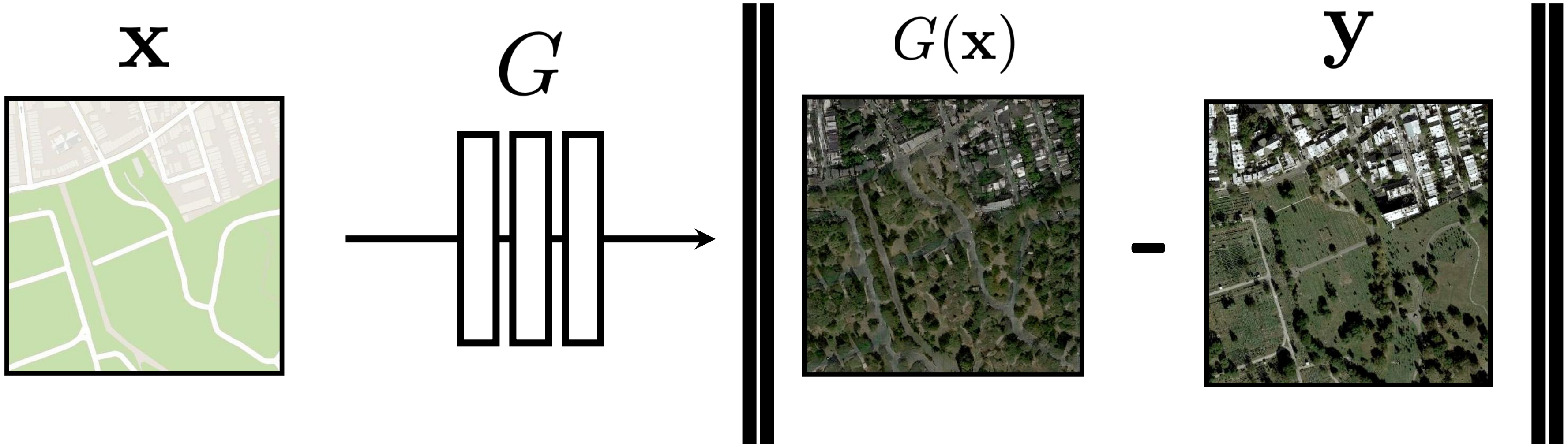
Conditional GAN

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G).$$

Training Details: Loss function

Conditional GAN

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G).$$



Stable training + fast convergence

[c.f. Pathak et al. CVPR 2016]

Input

Unstructured prediction (L1)

Input

Structured Prediction (cGAN)

Input

Output

Groundtruth

Data from
[maps.google.com]

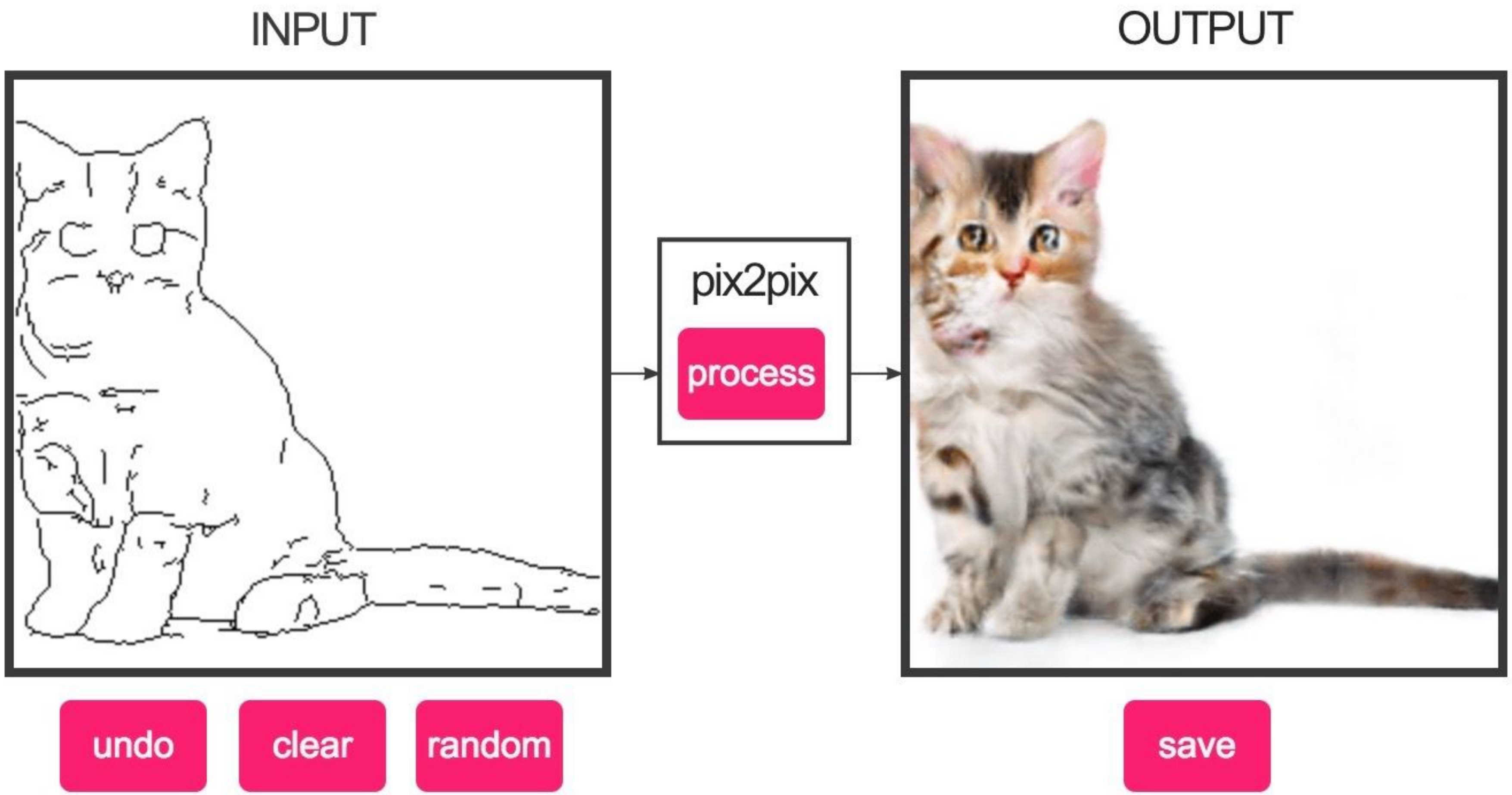
Input

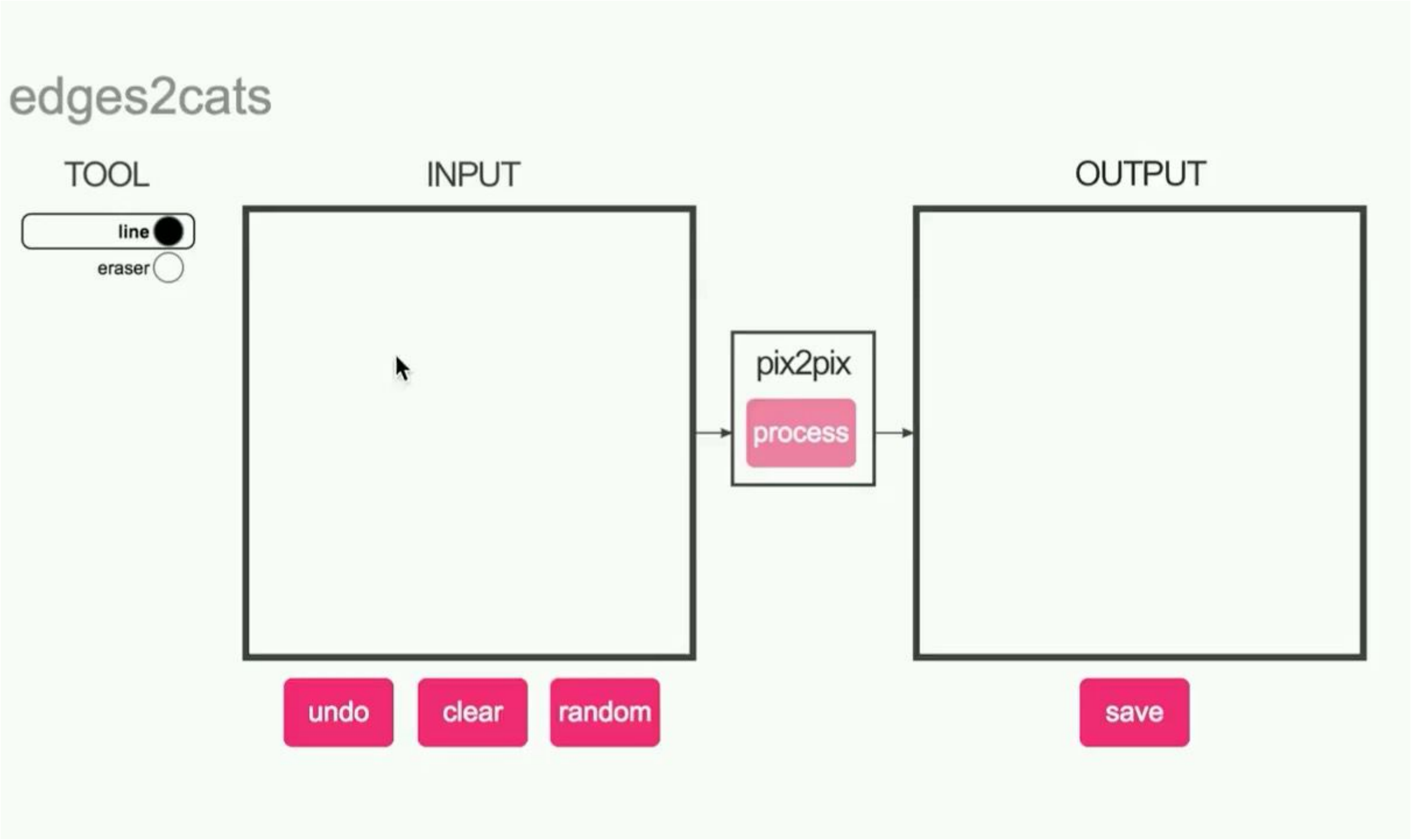
Output

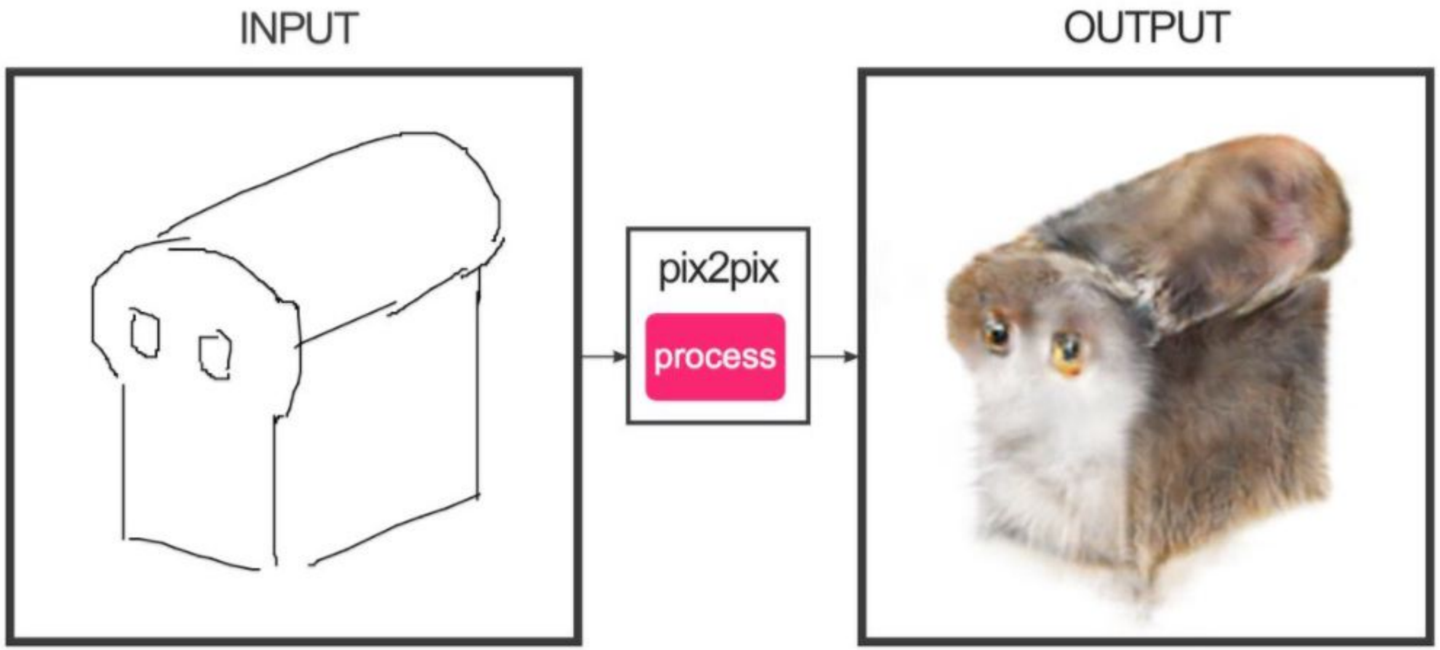
Groundtruth

Data from [maps.google.com]

#edges2cats [Chris Hesse]



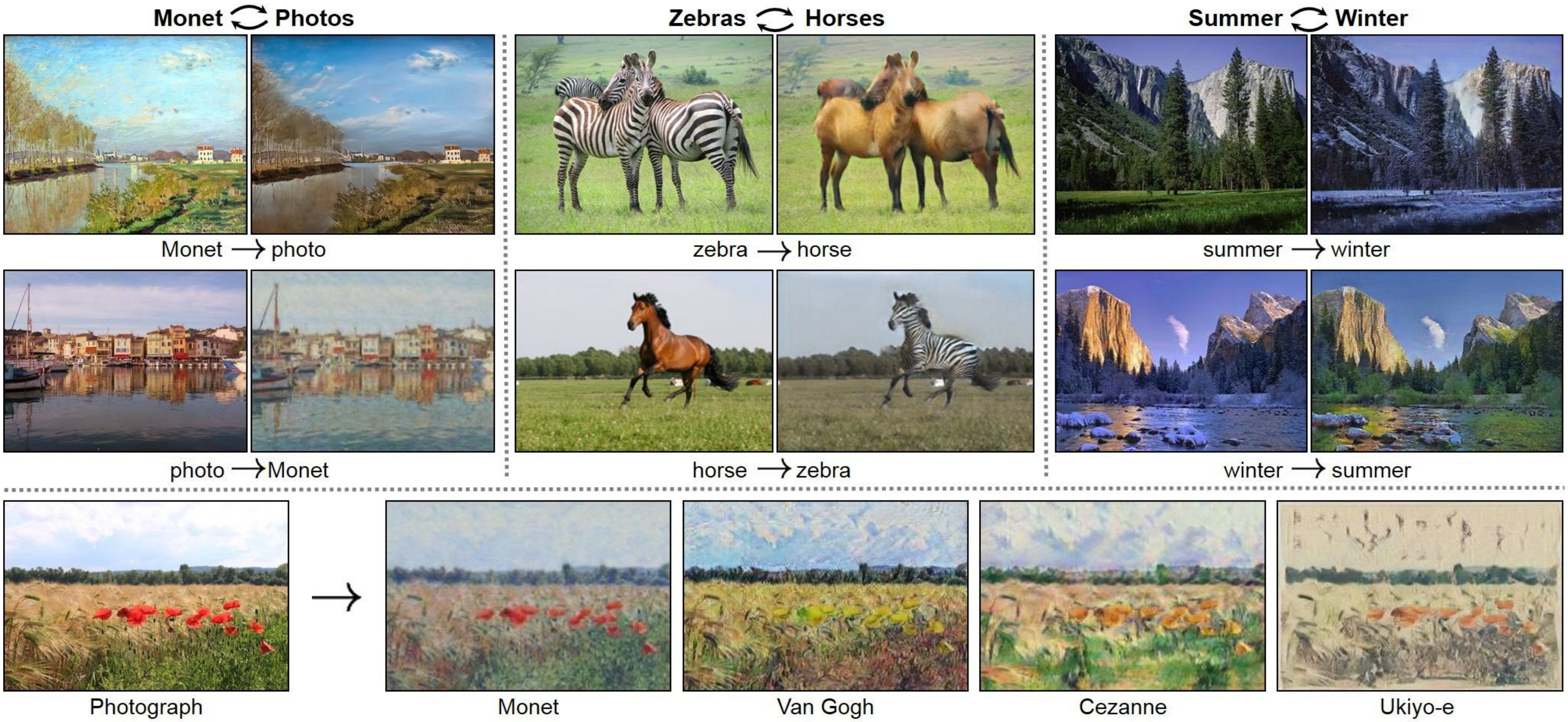

edges2cats
TOOL
line
eraser
INPUT
pix2pix
process
OUTPUT
undo
clear
random
save



Ivy Tasi @ivymyt

Vitaly Vidmirov @vvid

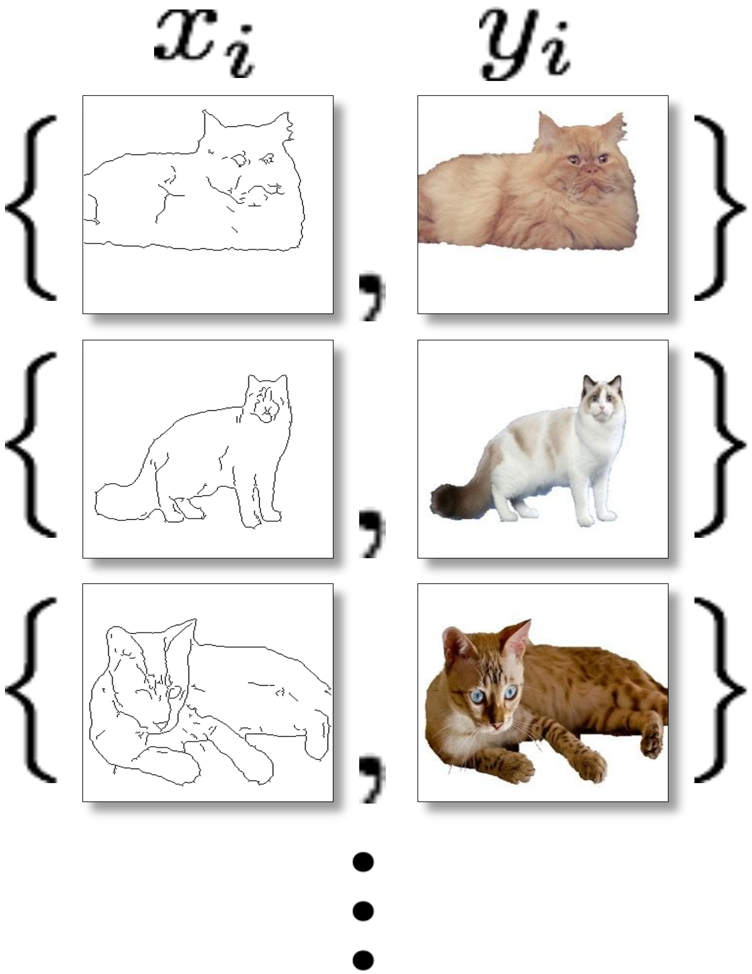
CycleGAN: Pix2Pix w/o input-output pairs



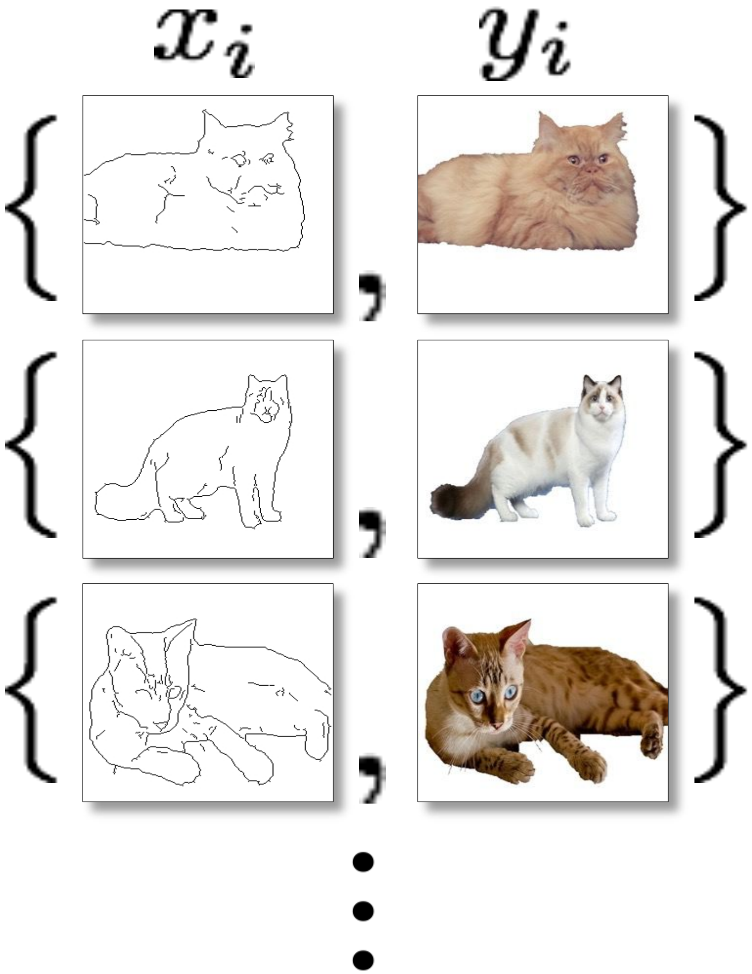
(Zhu et al. 2017)

Slide Credit: Taesung Park and Jun-Yan Zhu

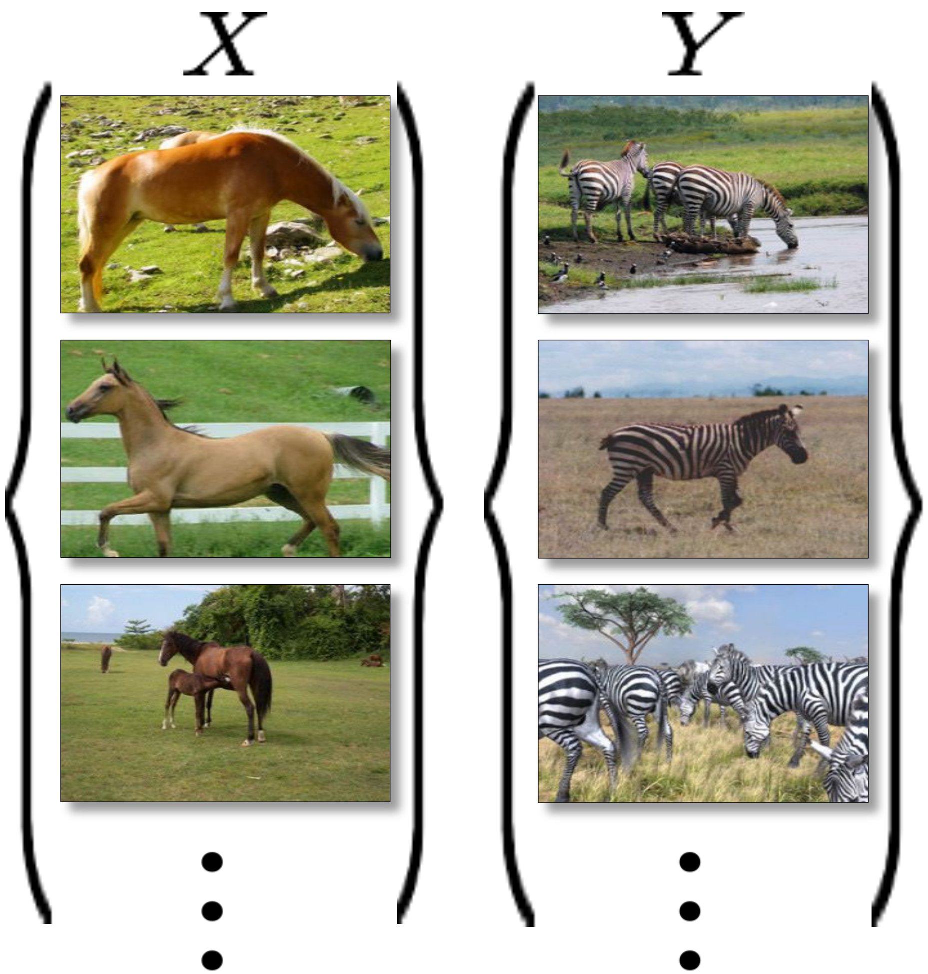
Paired data

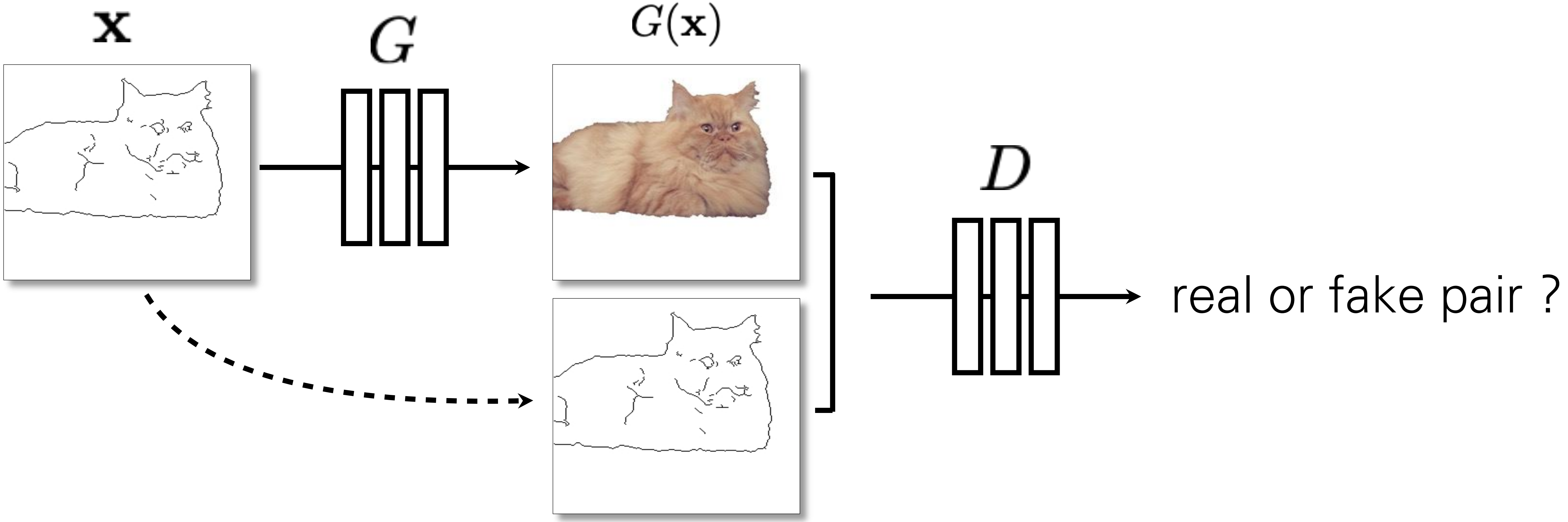


Paired data

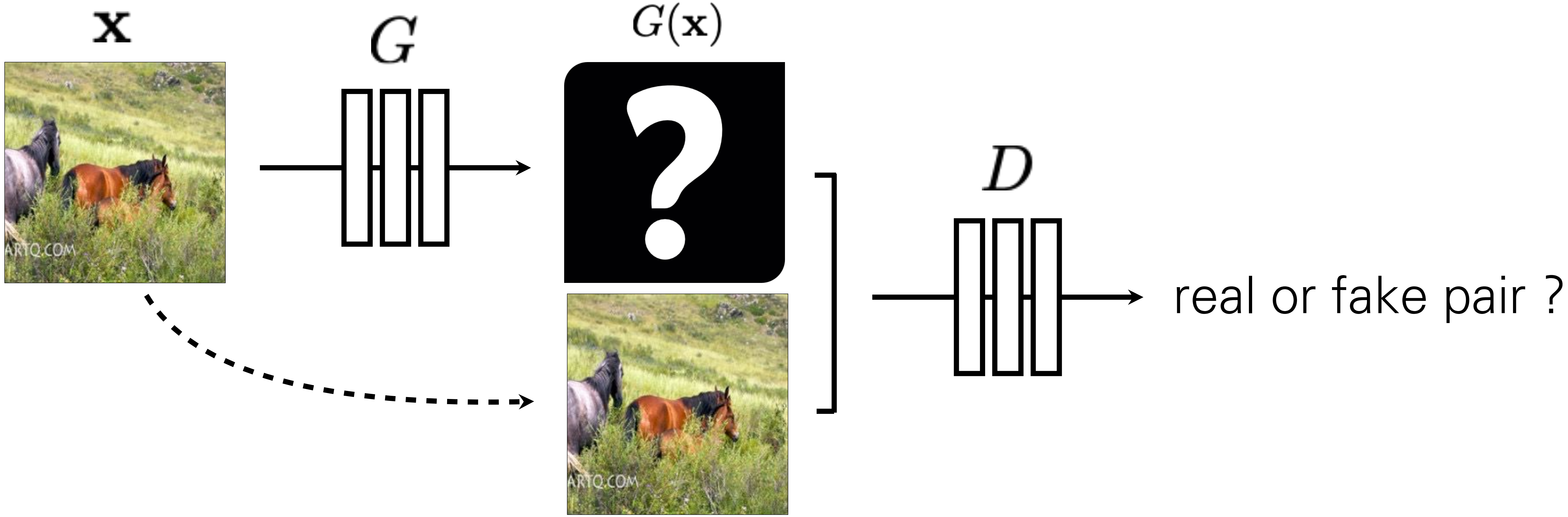


Unpaired data





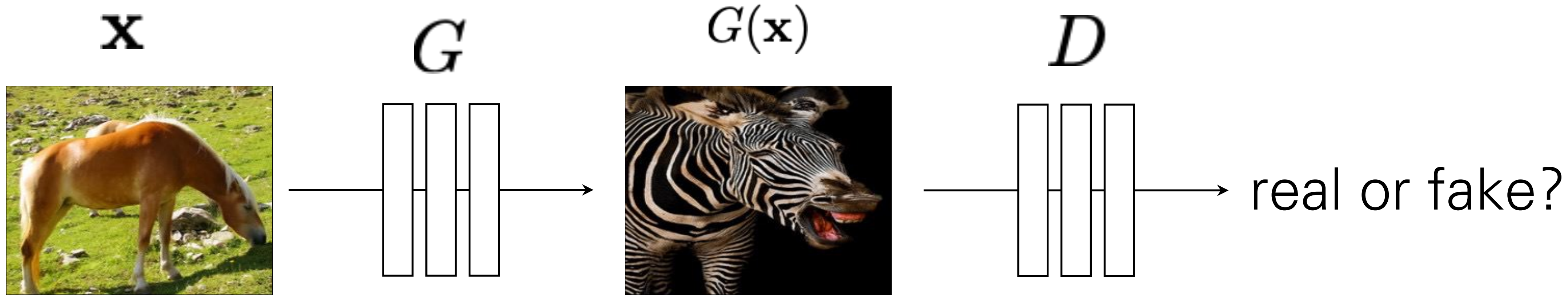
$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y}))\ ]$$



$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y}))\ ]$$

No input-output pairs!

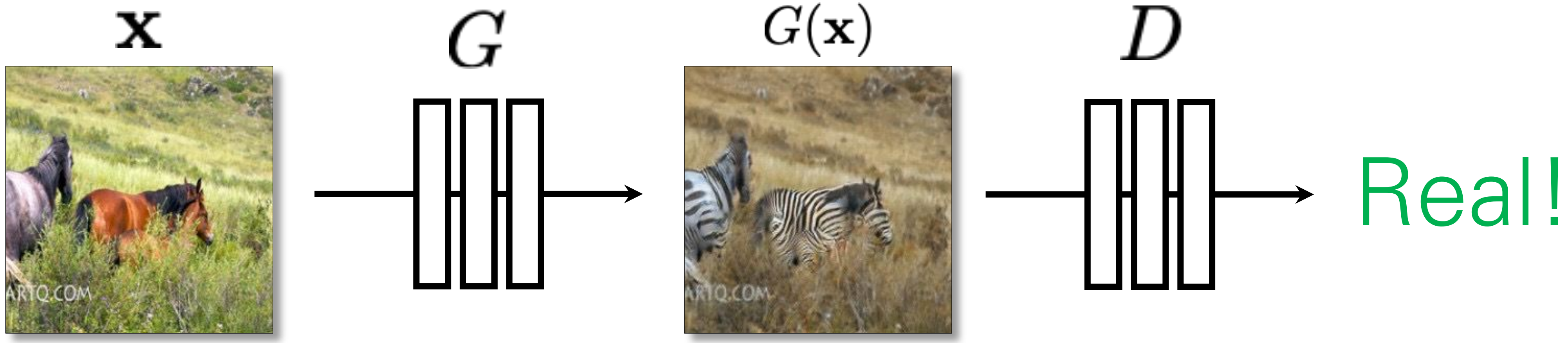
Slide Credit: Taesung Park and Jun-Yan Zhu



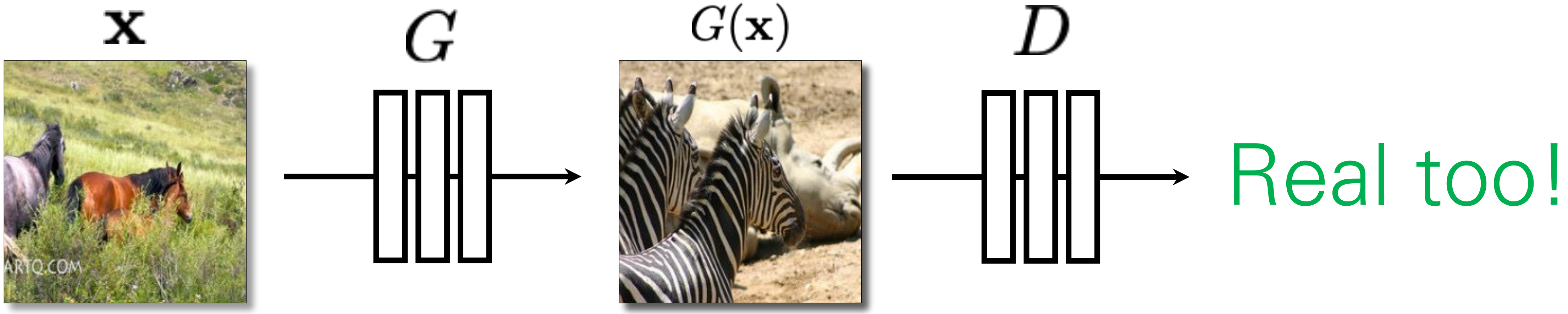
$$\arg\min_{G}\max_{D}\ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) \ + \ \log(1 - D(\mathbf{y}))\ ]$$

Usually loss functions check if output matches a target instance

GAN loss checks if output is part of an admissible set

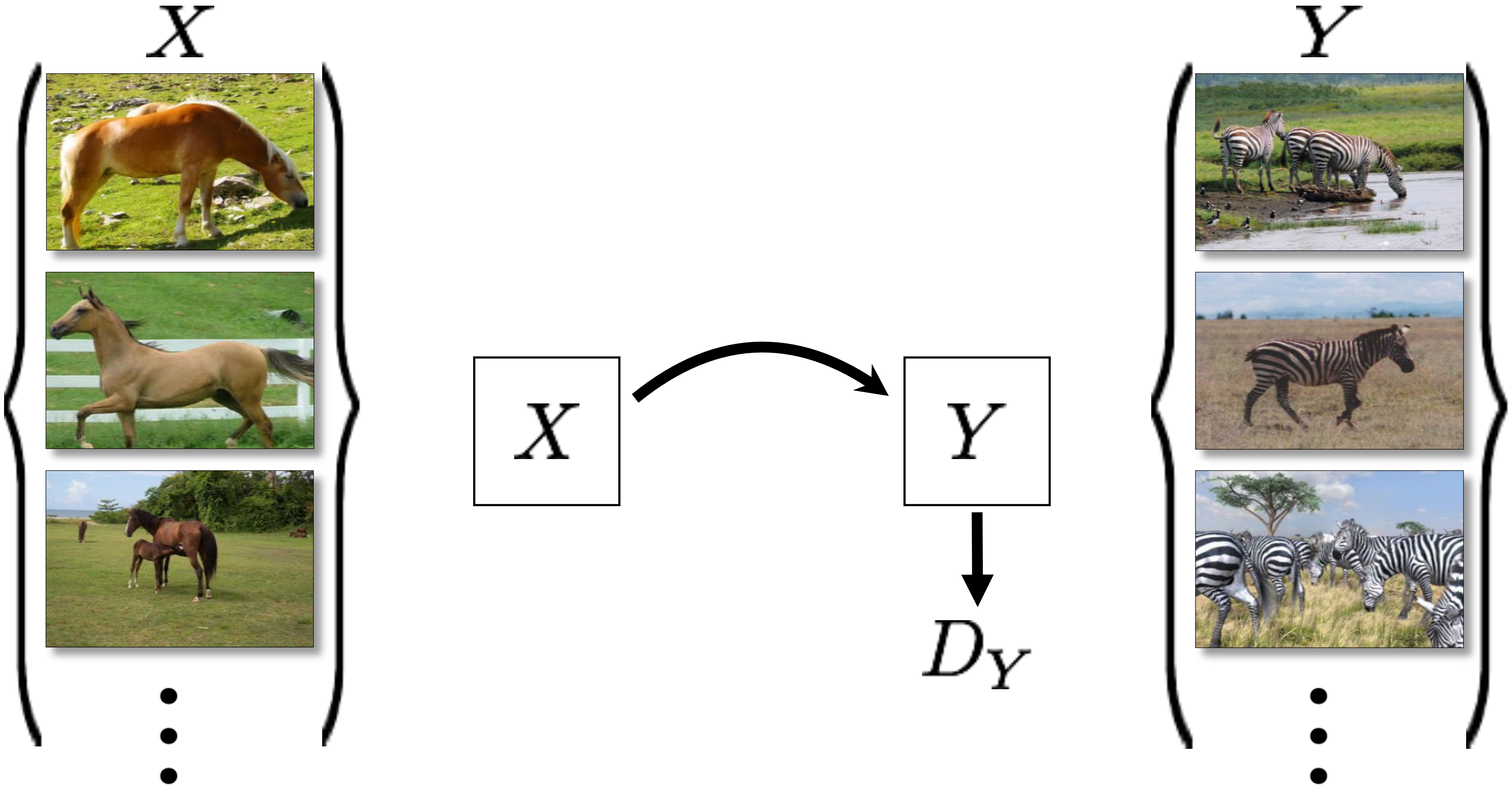


Slide Credit: Jun-Yan Zhu



Nothing to force output to correspond to input

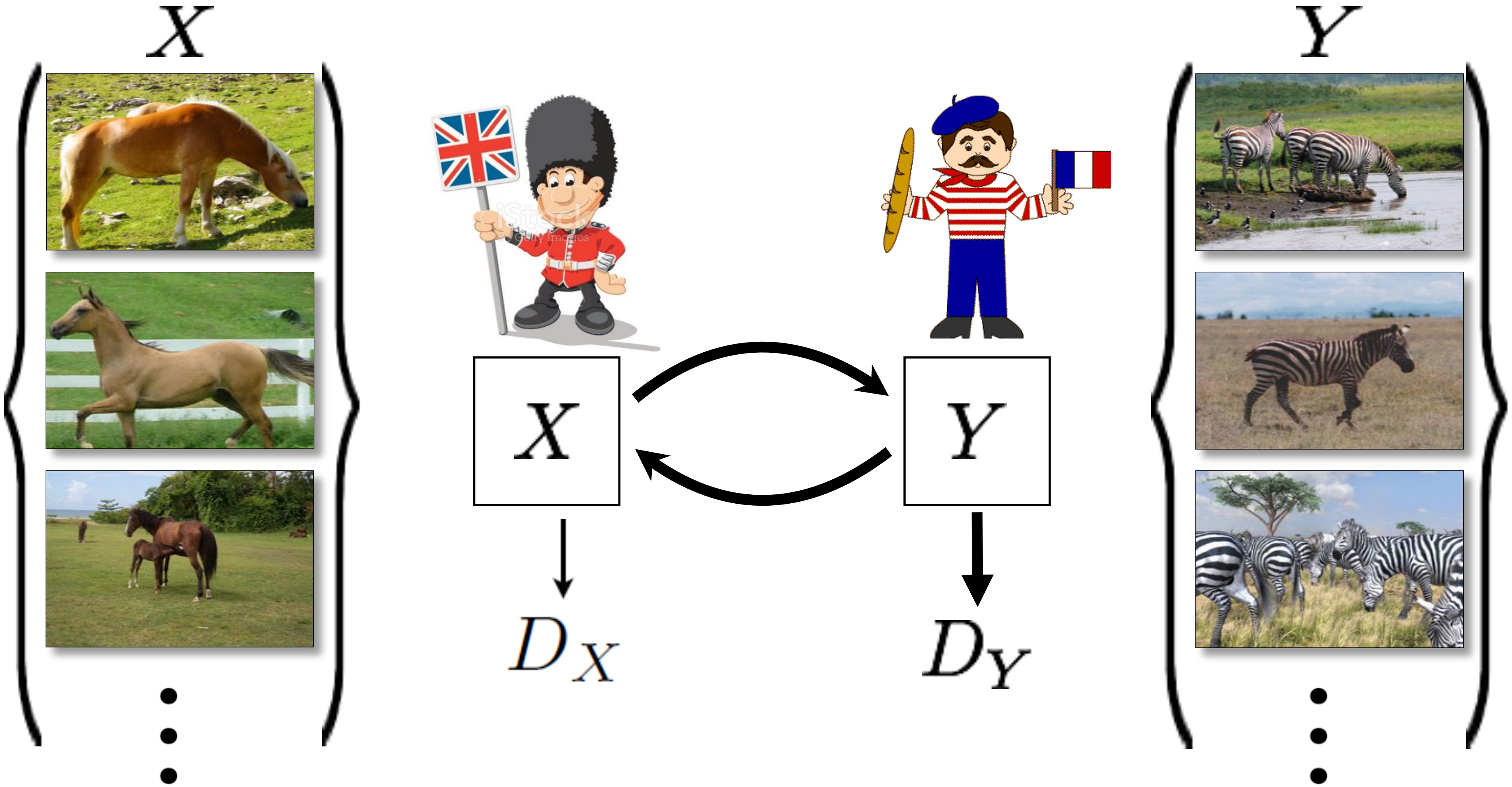
Cycle-Consistent Adversarial Networks



[Zhu et al. 2017], [Yi et al. 2017], [Kim et al. 2017]

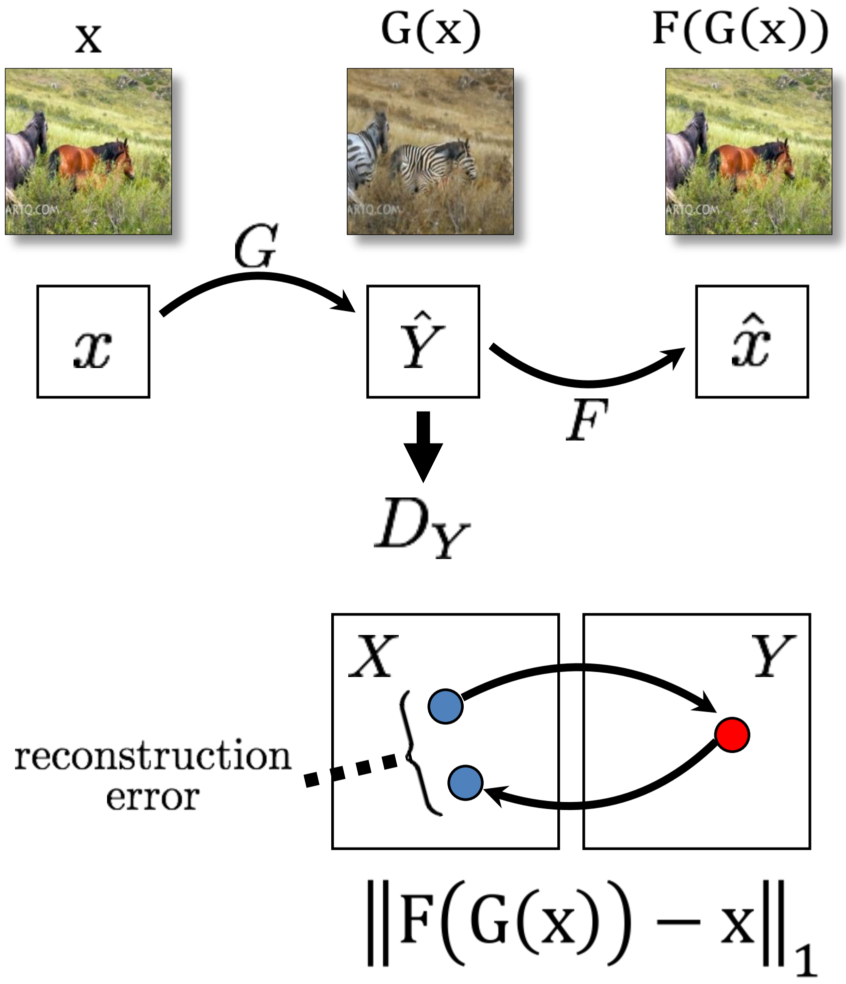
Slide Credit: Taesung Park and Jun-Yan Zhu

Cycle-Consistent Adversarial Networks



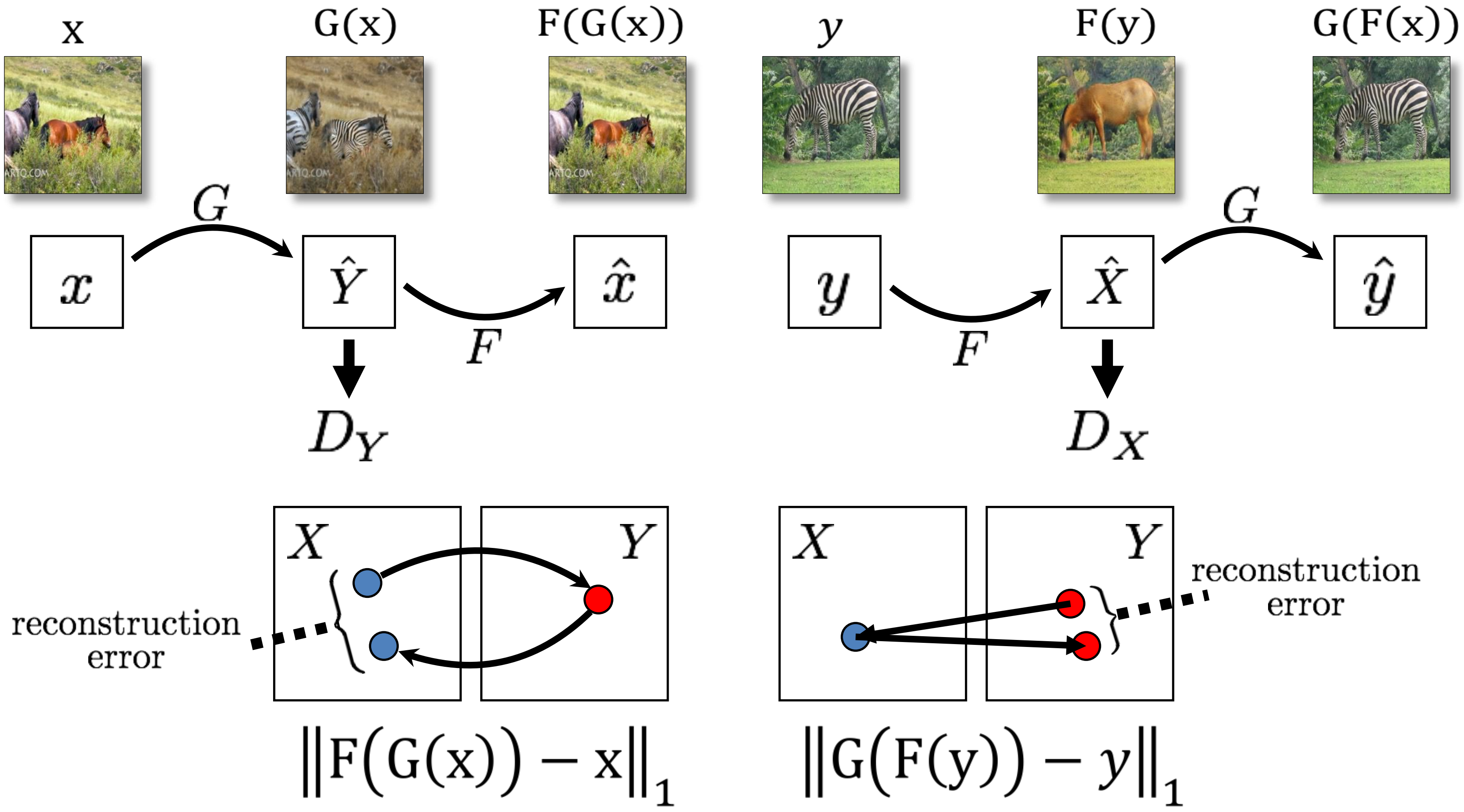
Slide Credit: Taesung Park and Jun-Yan Zhu

Cycle Consistency Loss

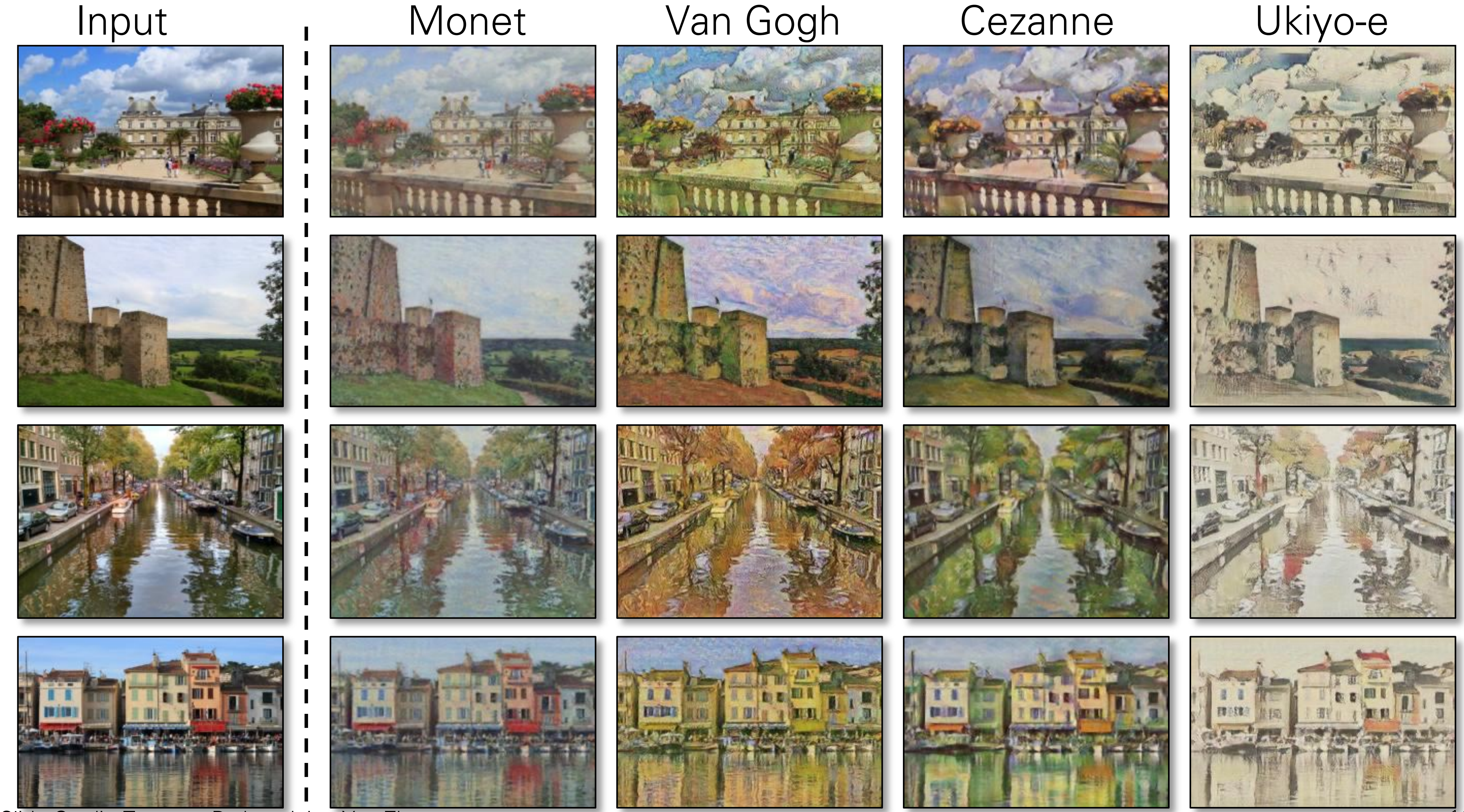


Slide Credit: Taesung Park and Jun-Yan Zhu

Cycle Consistency Loss



Slide Credit: Taesung Park and Jun-Yan Zhu



Slide Credit: Taesung Park and Jun-Yan Zhu

Monet's paintings → photos


Slide Credit: Taesung Park and Jun-Yan Zhu

Monet's paintings → photos

Slide Credit: Taesung Park and Jun-Yan Zhu

Leveraging pretrained models for efficient data translation

The point of deep learning is to enable learning with little data

Deep learning

Representations
(encoders)

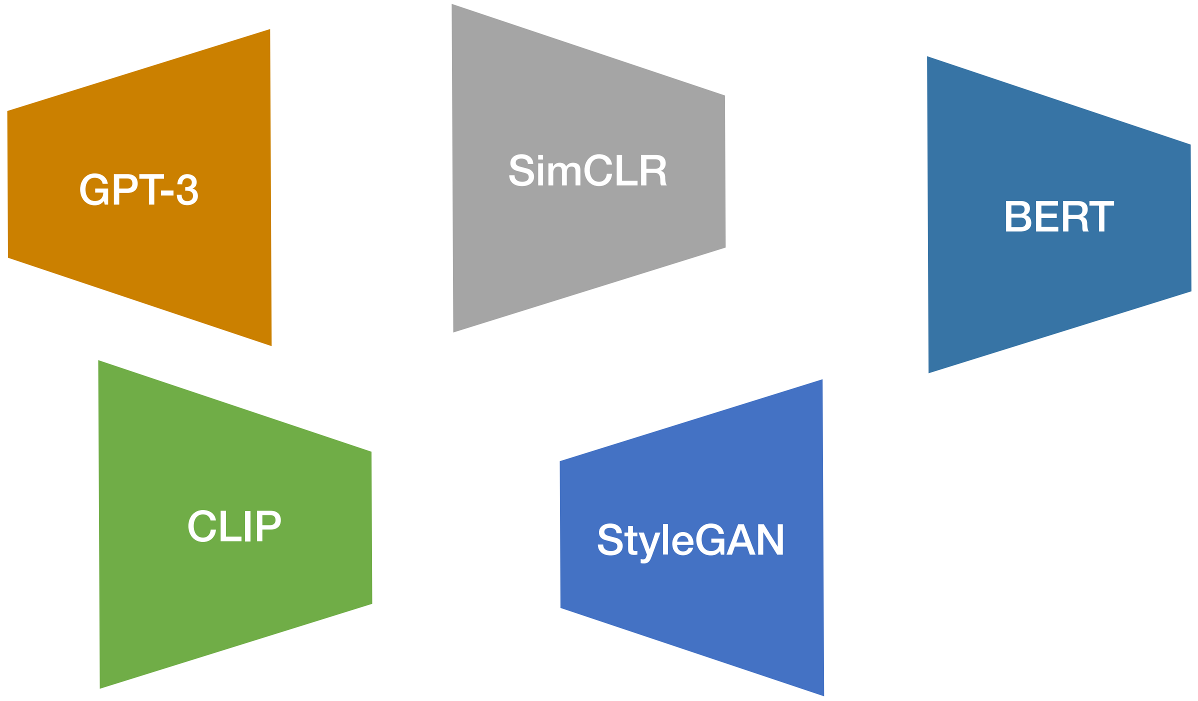
Models
(decoders)

Foundation models [Bommasani et al. 2021] https://arxiv.org/pdf/2108.07258.pdf

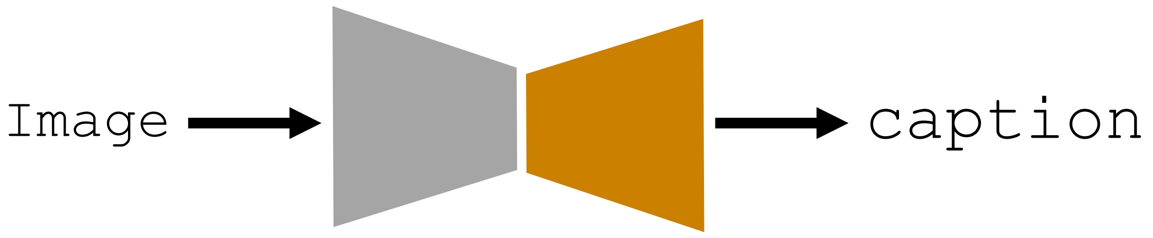
"If I have seen further it is by standing on the shoulders of Giants"
— Newton

[Blind Orion Searching for the Rising Sun by Nicolas Poussin, 1658]

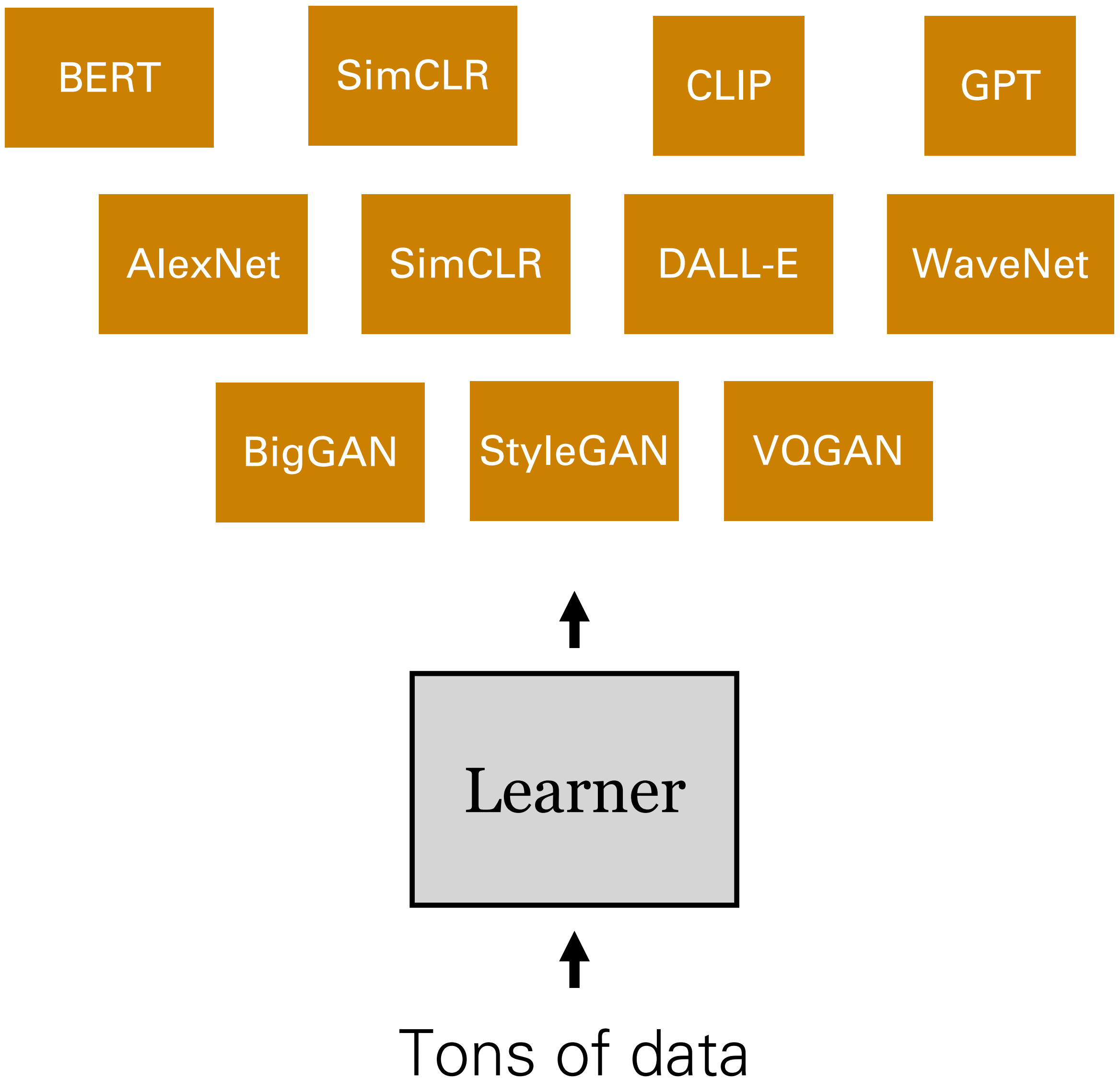
1. Learn foundation model **encoders** and **decoders** for each domain



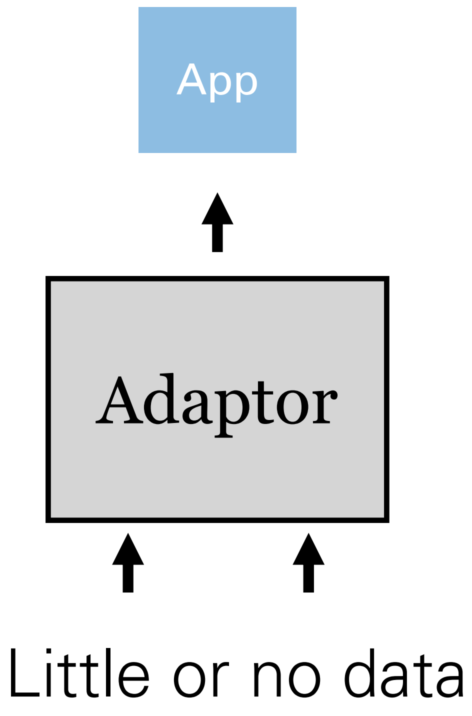
2. Plug them together to translate between modalities (may require finetuning)



Learn foundation models

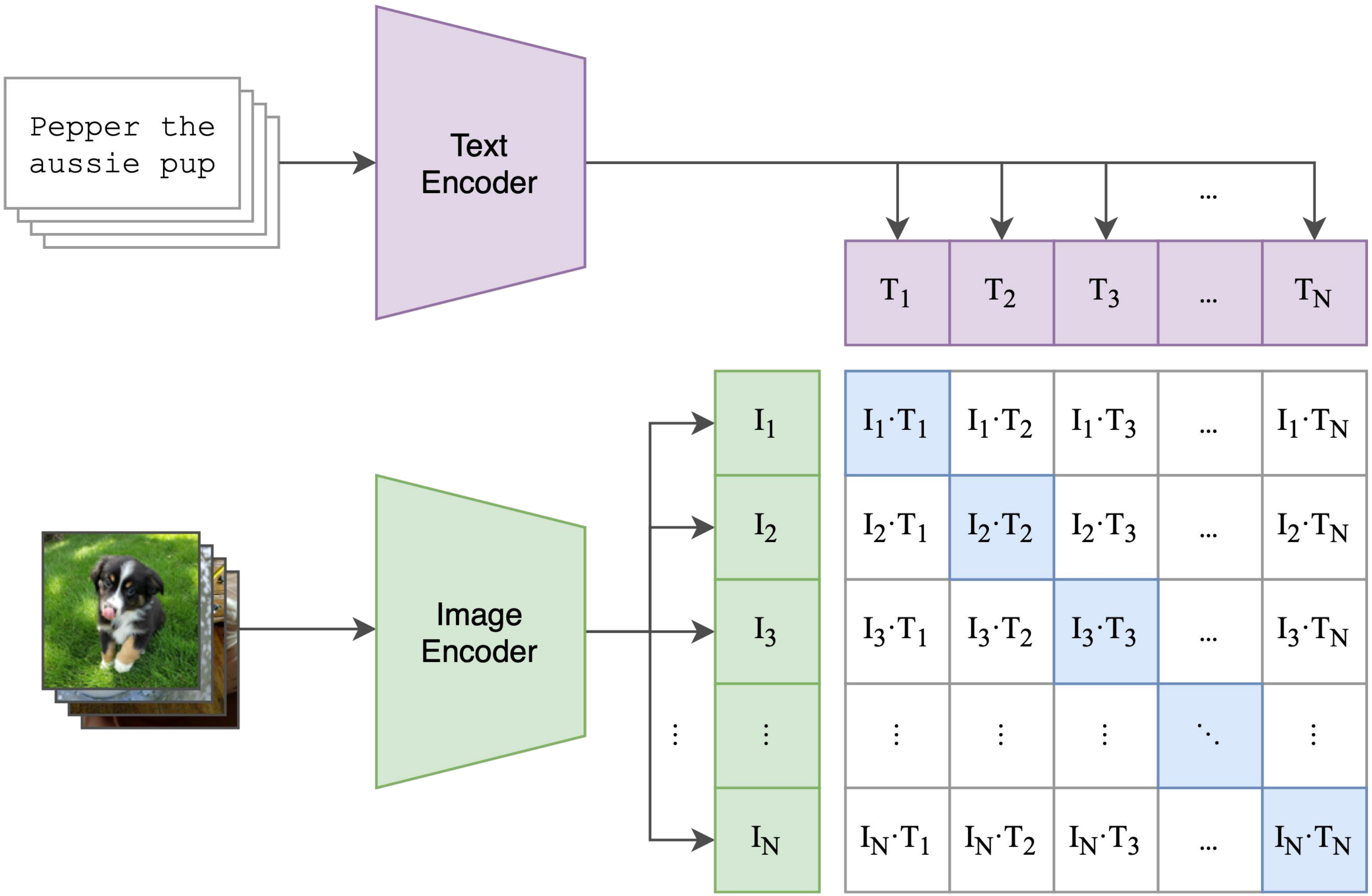


Use/adapt foundations to solve new problems



CLIP

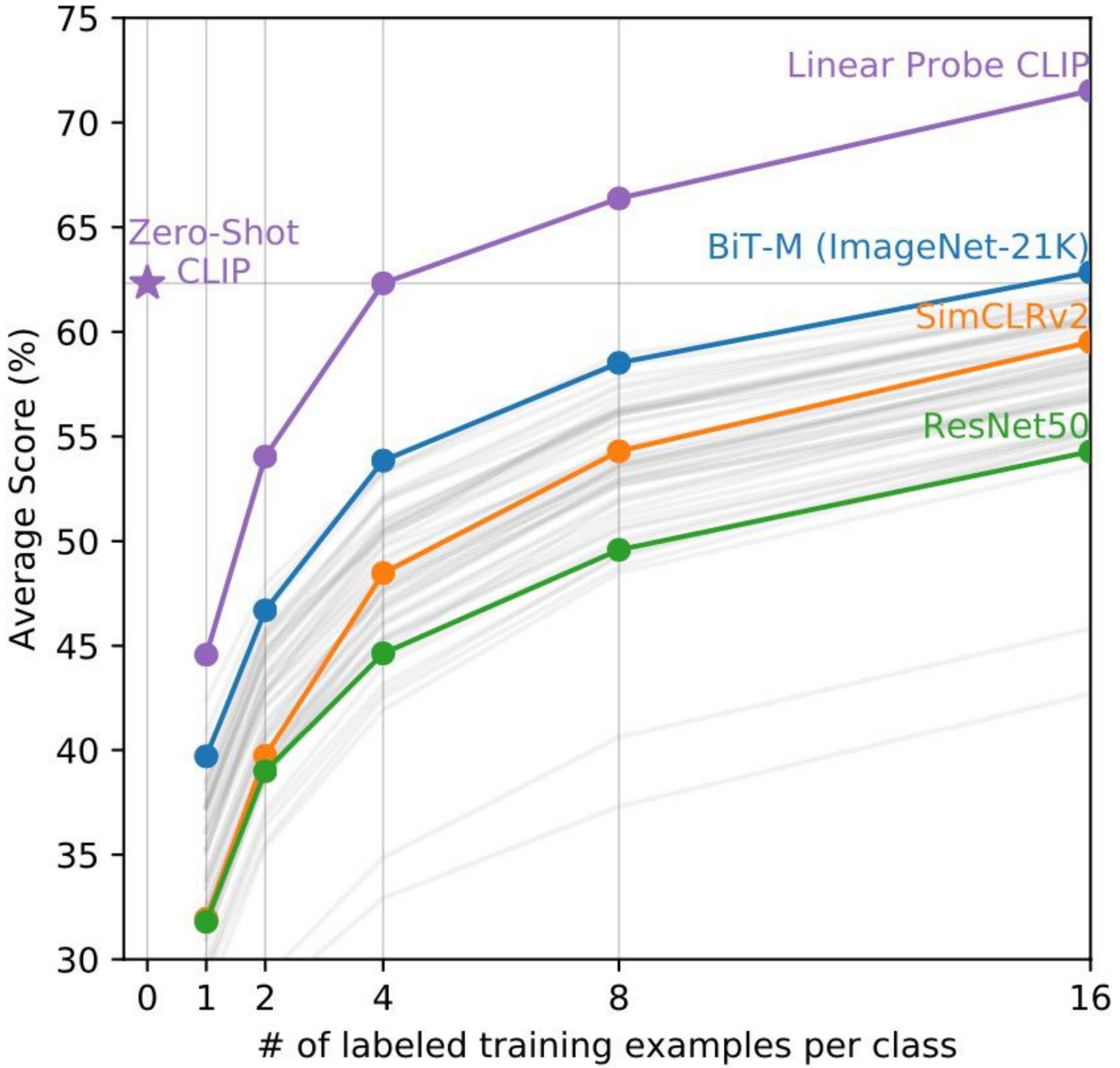
[Radford et al., 2021] https://arxiv.org/pdf/2103.00020.pdf



[https://openai.com/blog/clip/]

CLIP [Radford et al., 2021] https://arxiv.org/pdf/2103.00020.pdf

2. **Adaptor:**
Linear classifer on top of image encodings

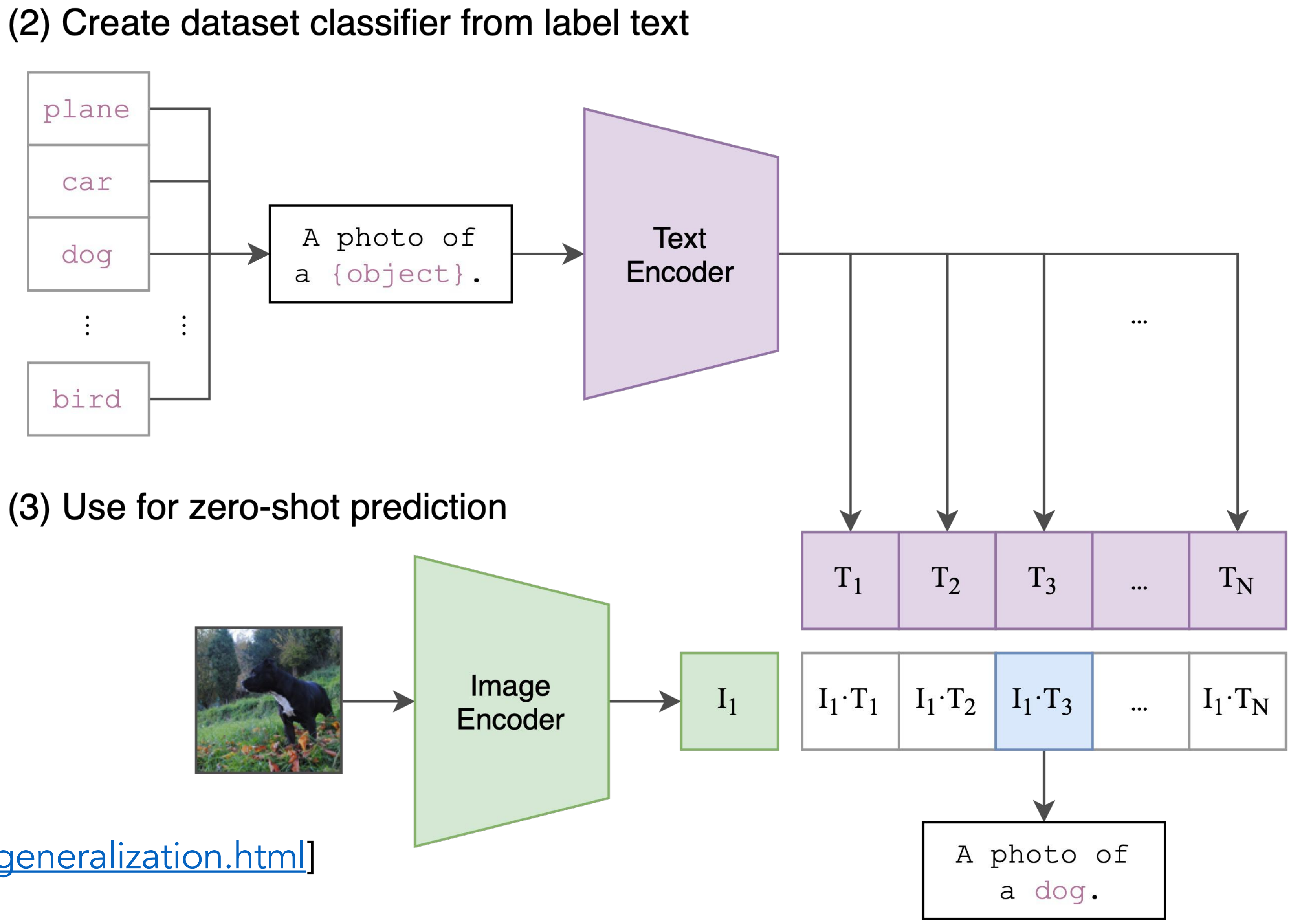


[https://openai.com/blog/clip/]

CLIP

[Radford et al., 2021] https://arxiv.org/pdf/2103.00020.pdf

2. Adaptor: Just ask



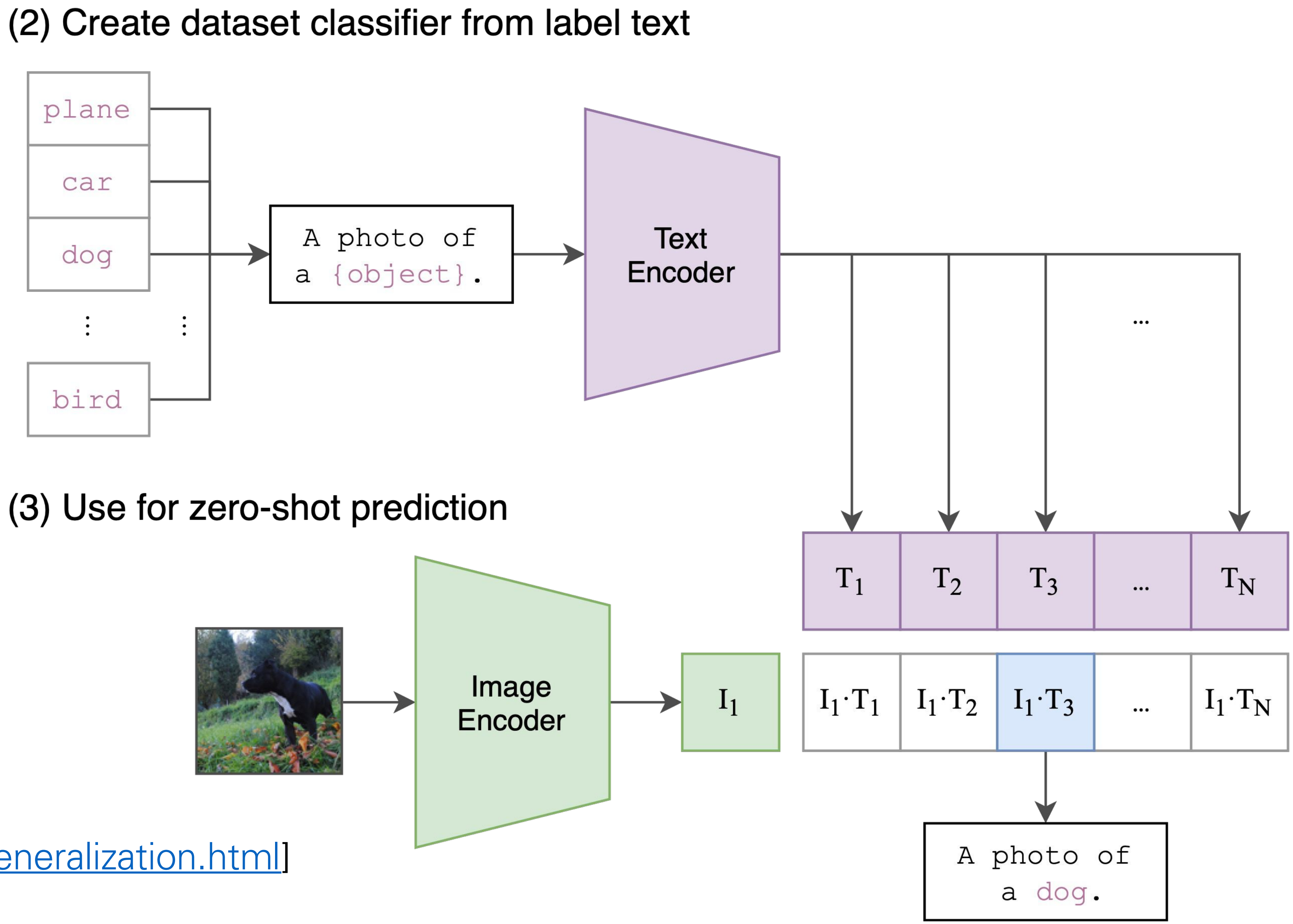
[https://evjang.com/2021/10/23/generalization.html]

[https://openai.com/blog/clip/]

CLIP

[Radford et al., 2021] https://arxiv.org/pdf/2103.00020.pdf

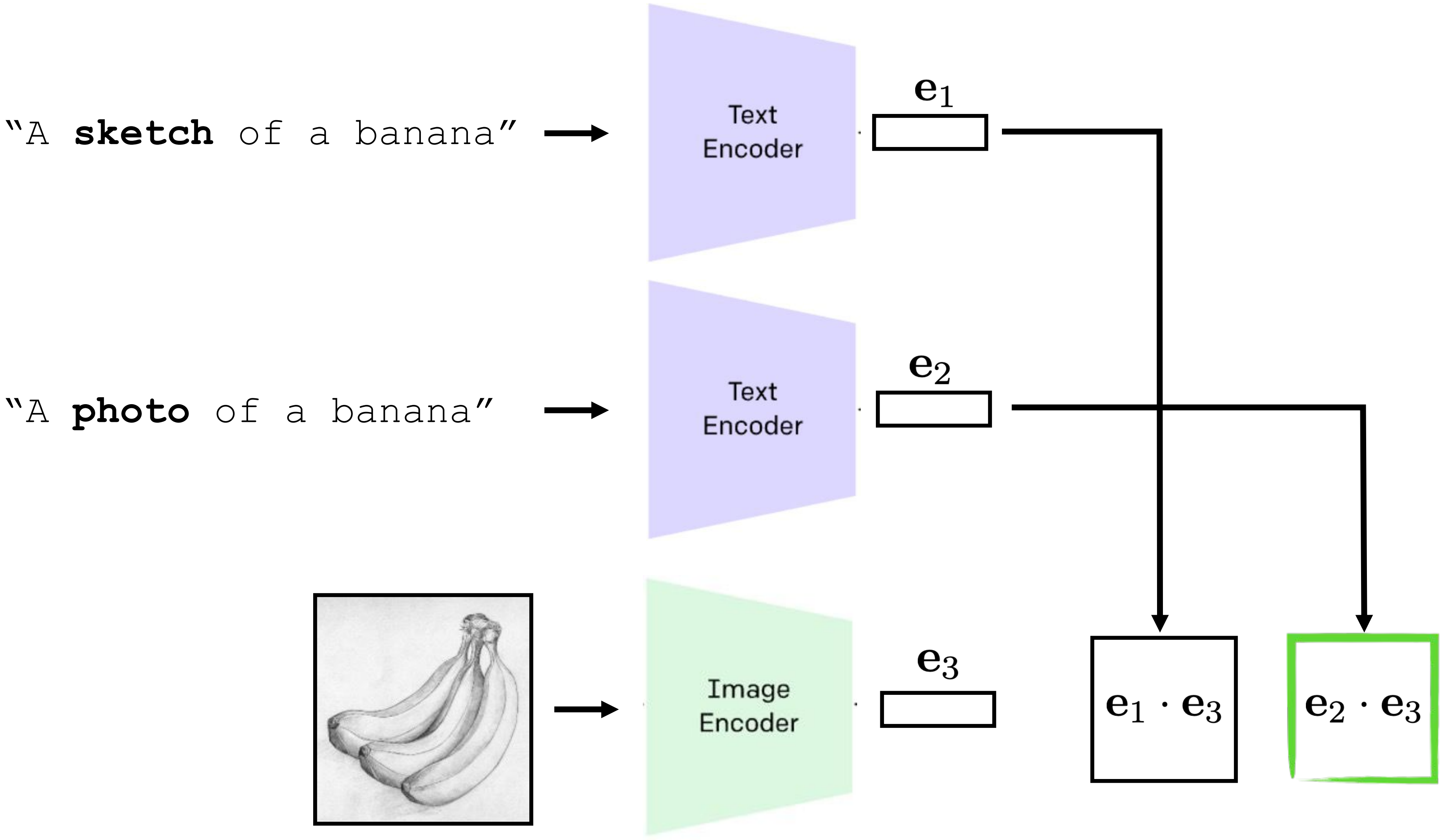
2. **Adaptor**: Just ask



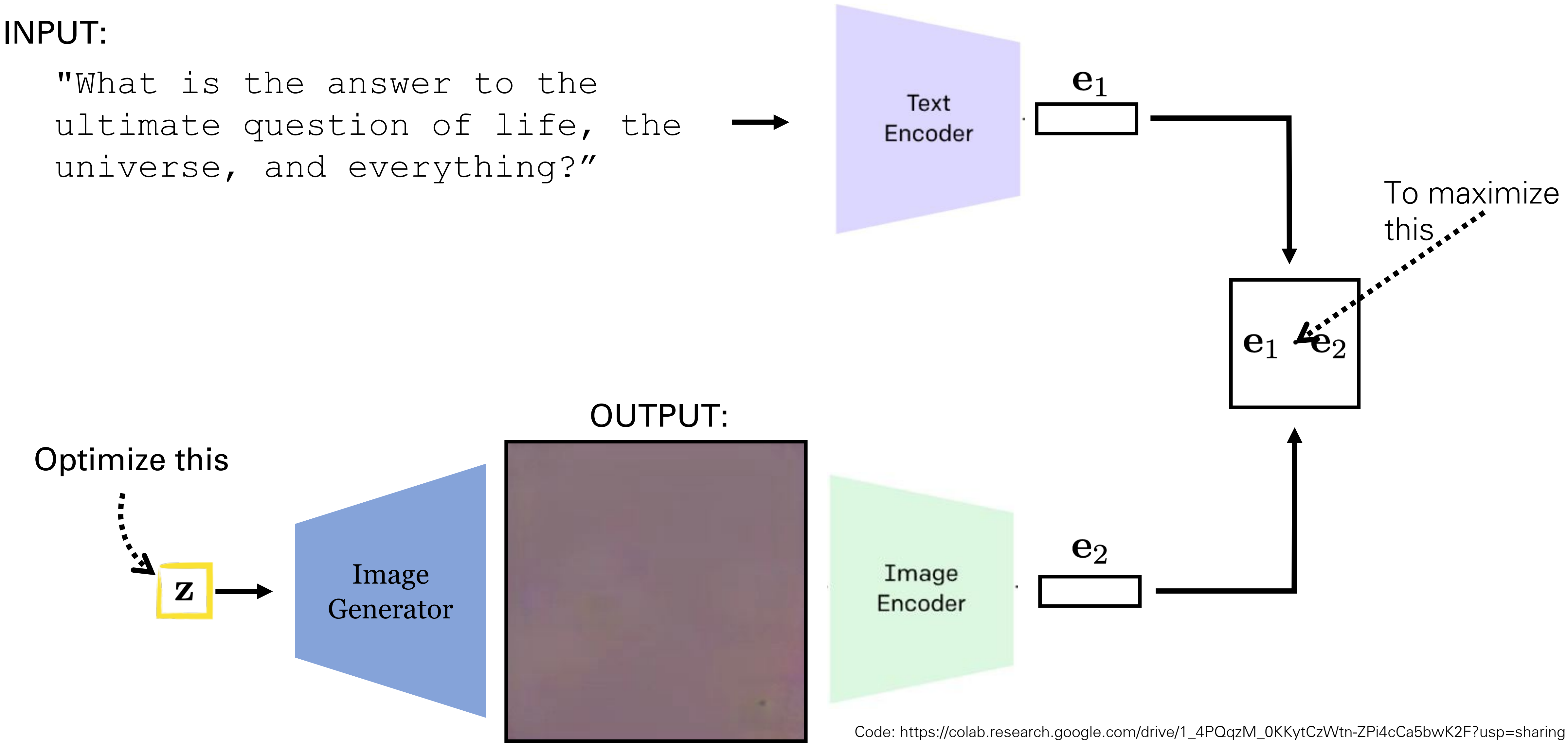
[https://evjang.com/2021/10/23/generalization.html]

[https://openai.com/blog/clip/]

New capabilities by just asking

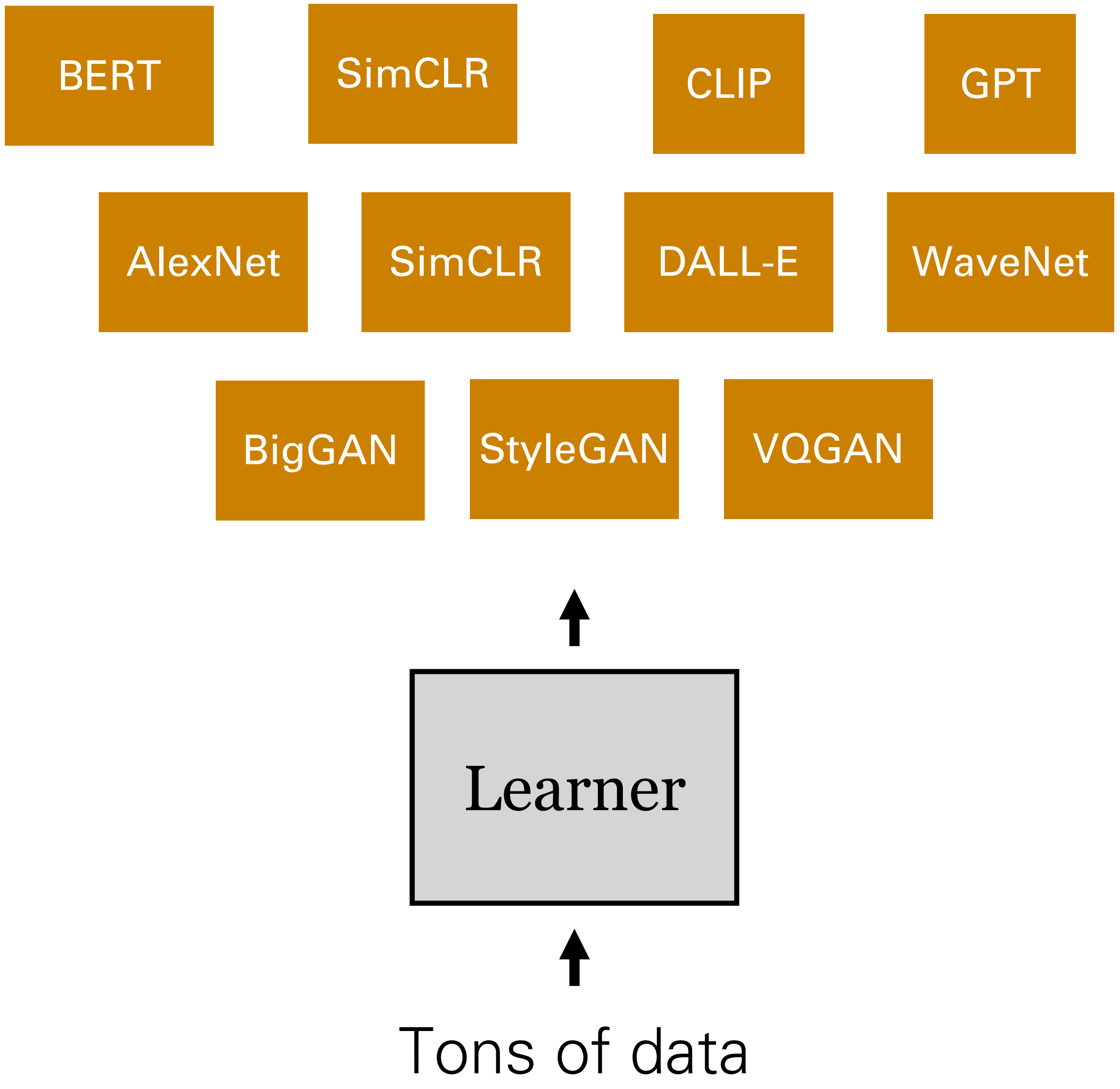


New capabilities by plugging pretrained models together: CLIP+GAN

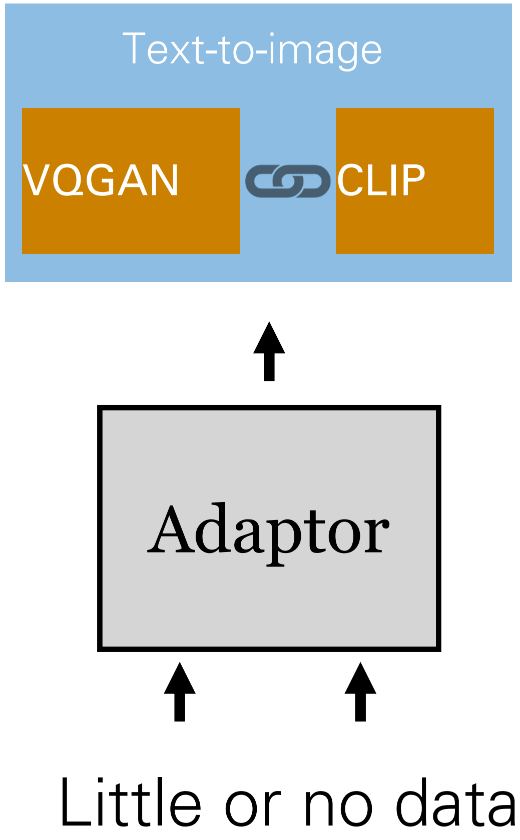


Code: https://colab.research.google.com/drive/1_4PQqzM_0KKytCzWtn-ZPi4cCa5bwK2F?usp=sharing

Learn foundation models

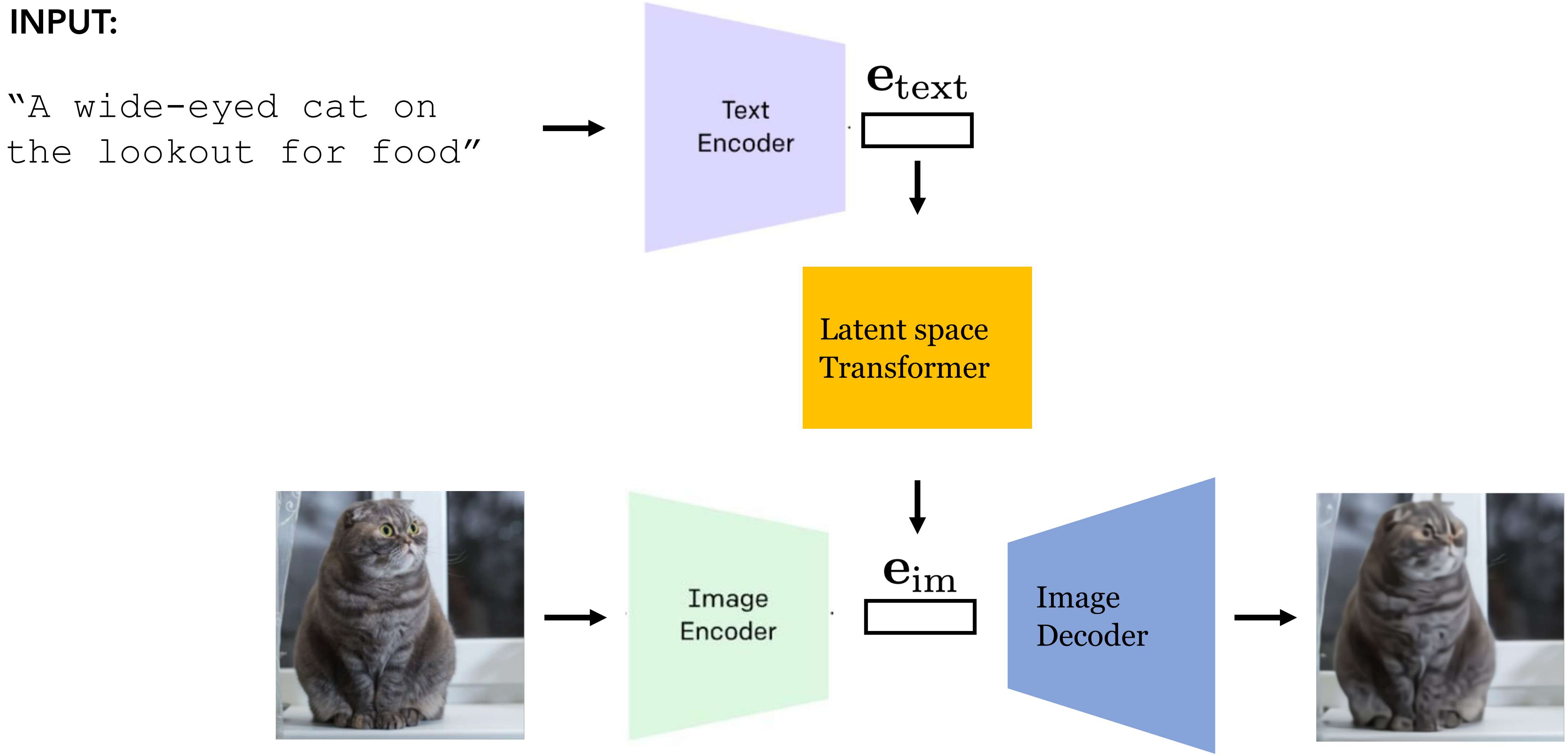


Use/adapt foundations to solve new problems



DALL-E

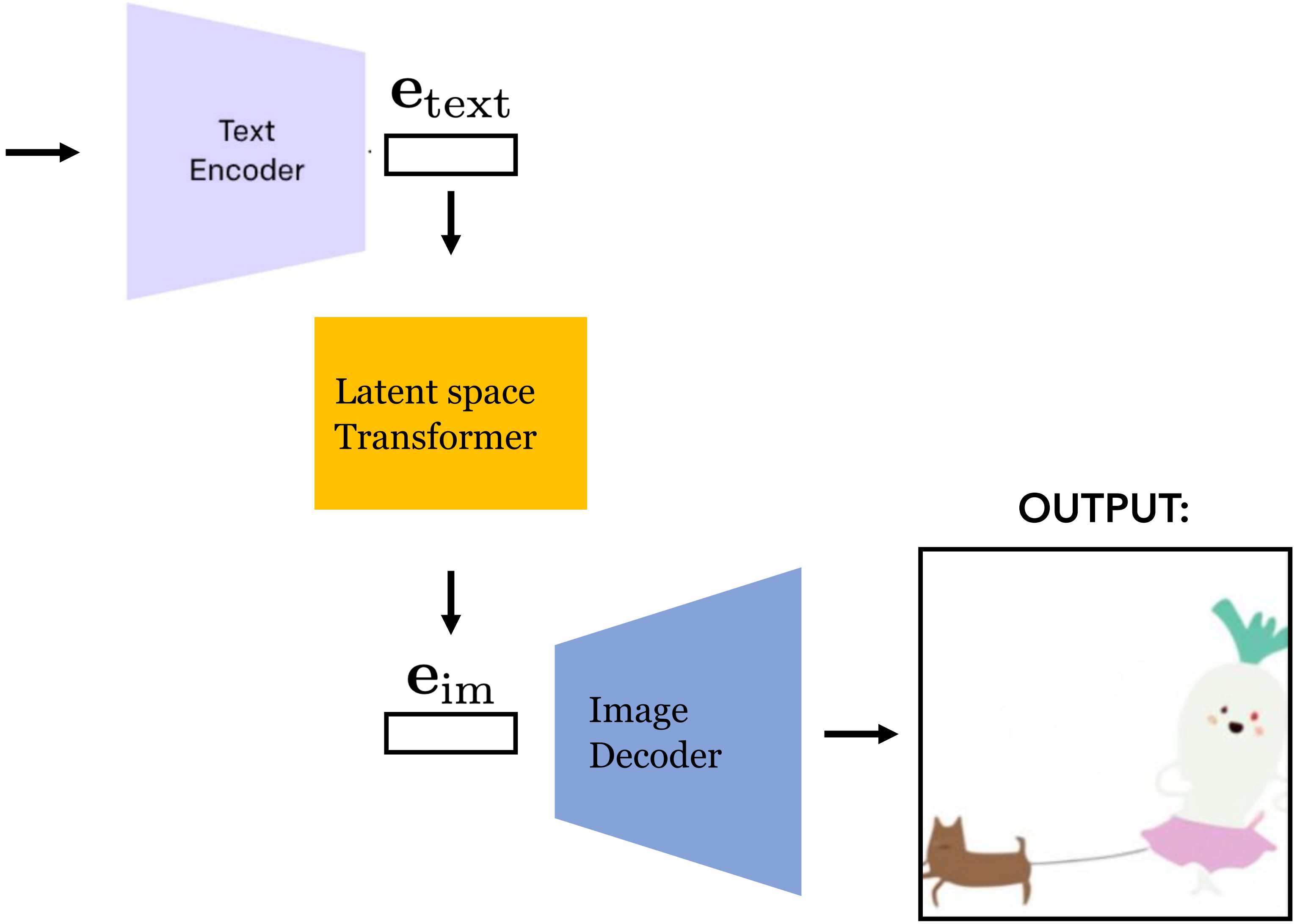
[Ramesh et al. 2021] https://arxiv.org/pdf/2102.12092.pdf https://openai.com/blog/dall-e/



Text-to-image translation

INPUT:

```
“An illustration of a
baby daikon radish in
a tutu walking a dog”
```



New capabilities by just asking: product design

TEXT PROMPT an armchair in the shape of an avocado. an armchair imitating an avocado.

AI-GENERATED IMAGES

New capabilities by just asking: image translation

TEXT PROMPT the exact same cat on the top as a sketch on the bottom

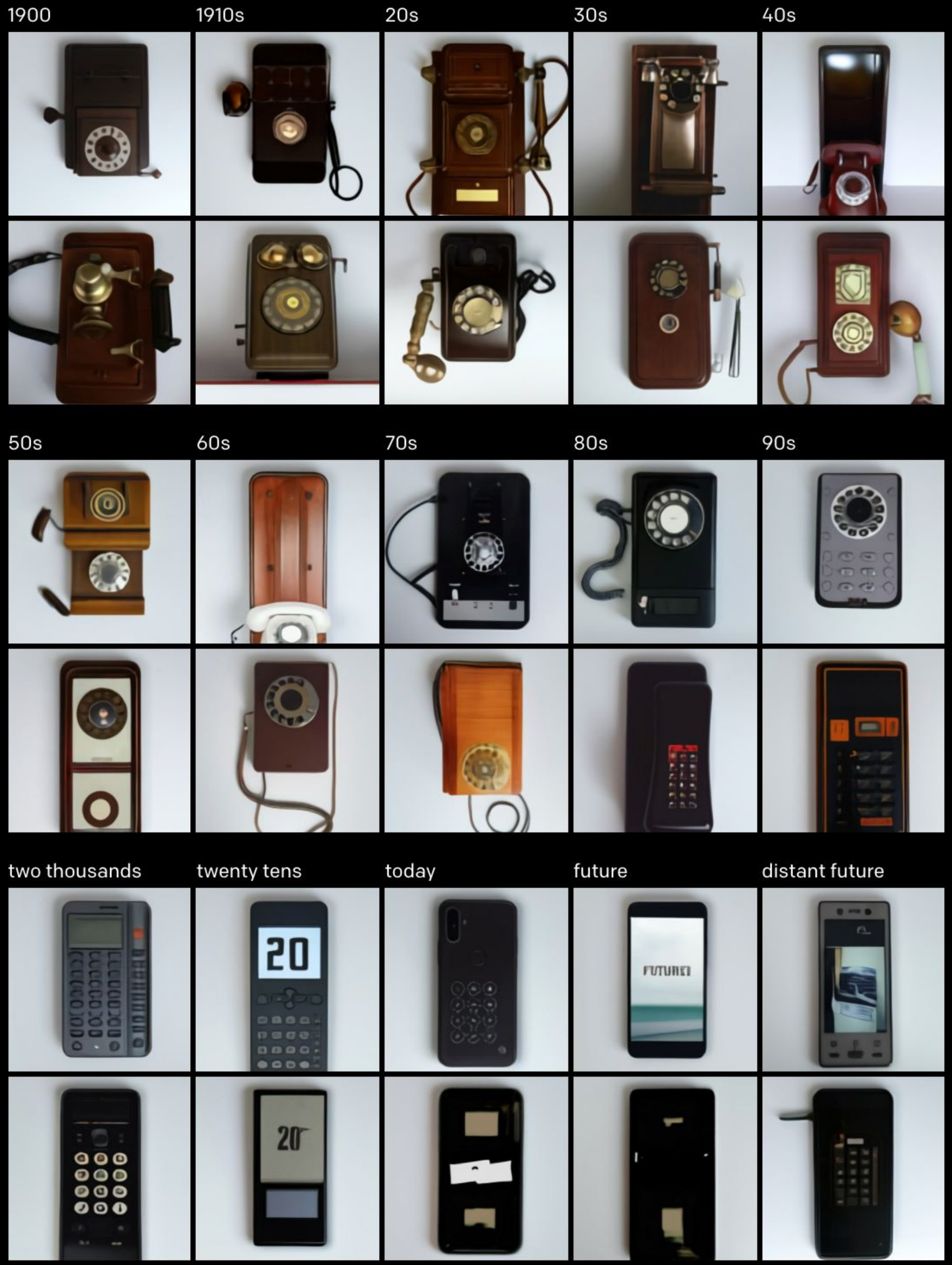
AI-GENERATED IMAGES

AI-GENERATED IMAGES

TEXT PROMPT

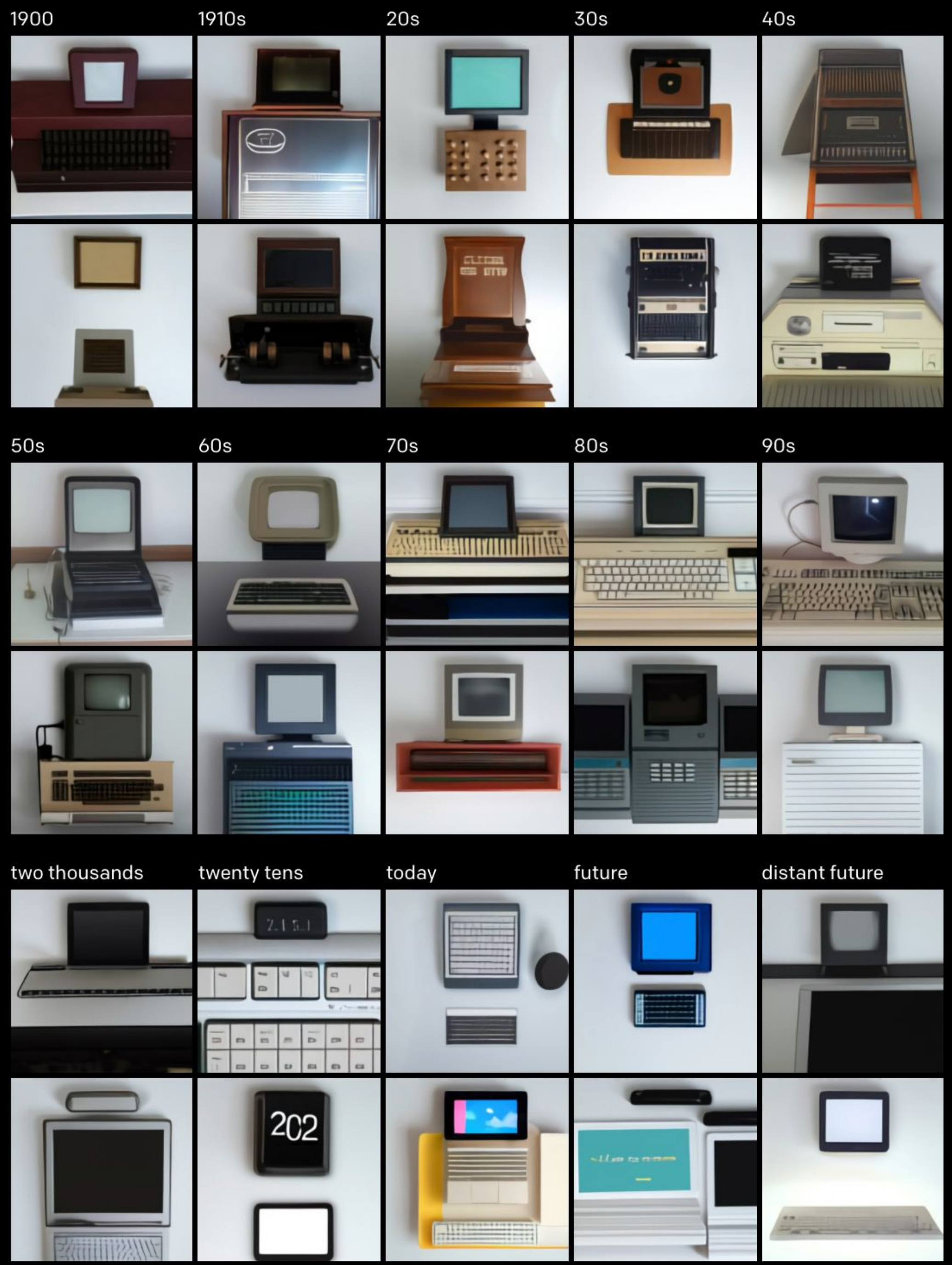
a photo of a phone from the ...

AI-GENERATED IMAGES



TEXT PROMPT a photo of a computer from the ...

AI-GENERATED IMAGES

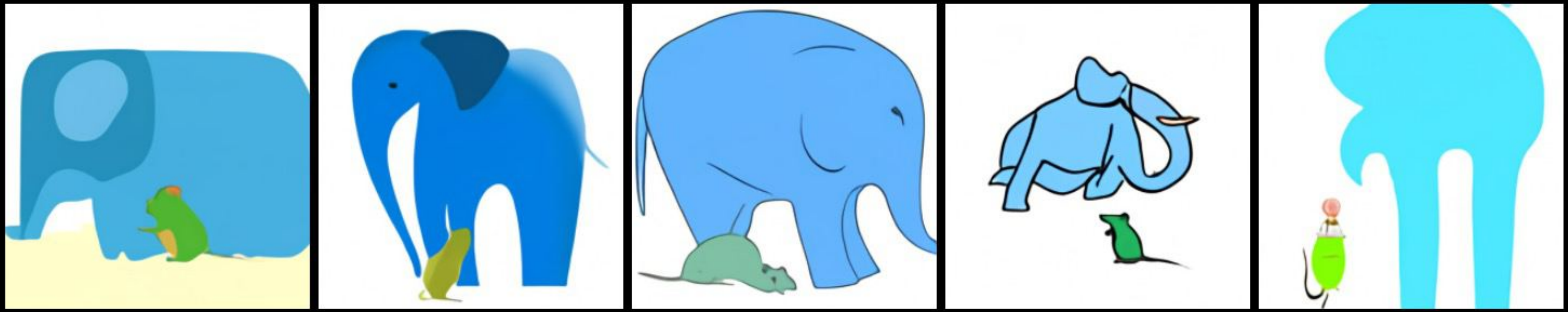


DALL-E

[Ramesh et al. 2021] https://arxiv.org/pdf/2102.12092.pdf https://openai.com/blog/dall-e/

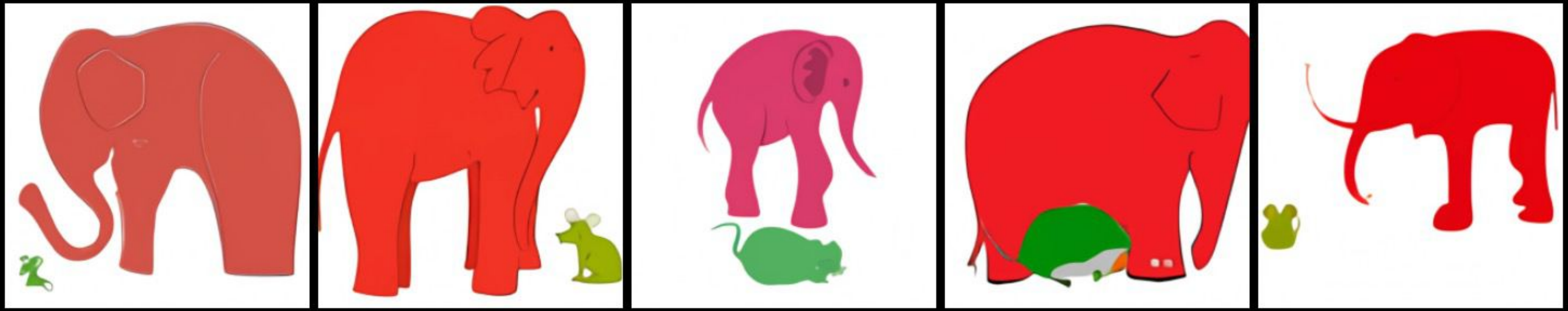
TEXT PROMPT an illustration of a small green mouse sitting below a large blue elephant

AI-GENERATED IMAGES

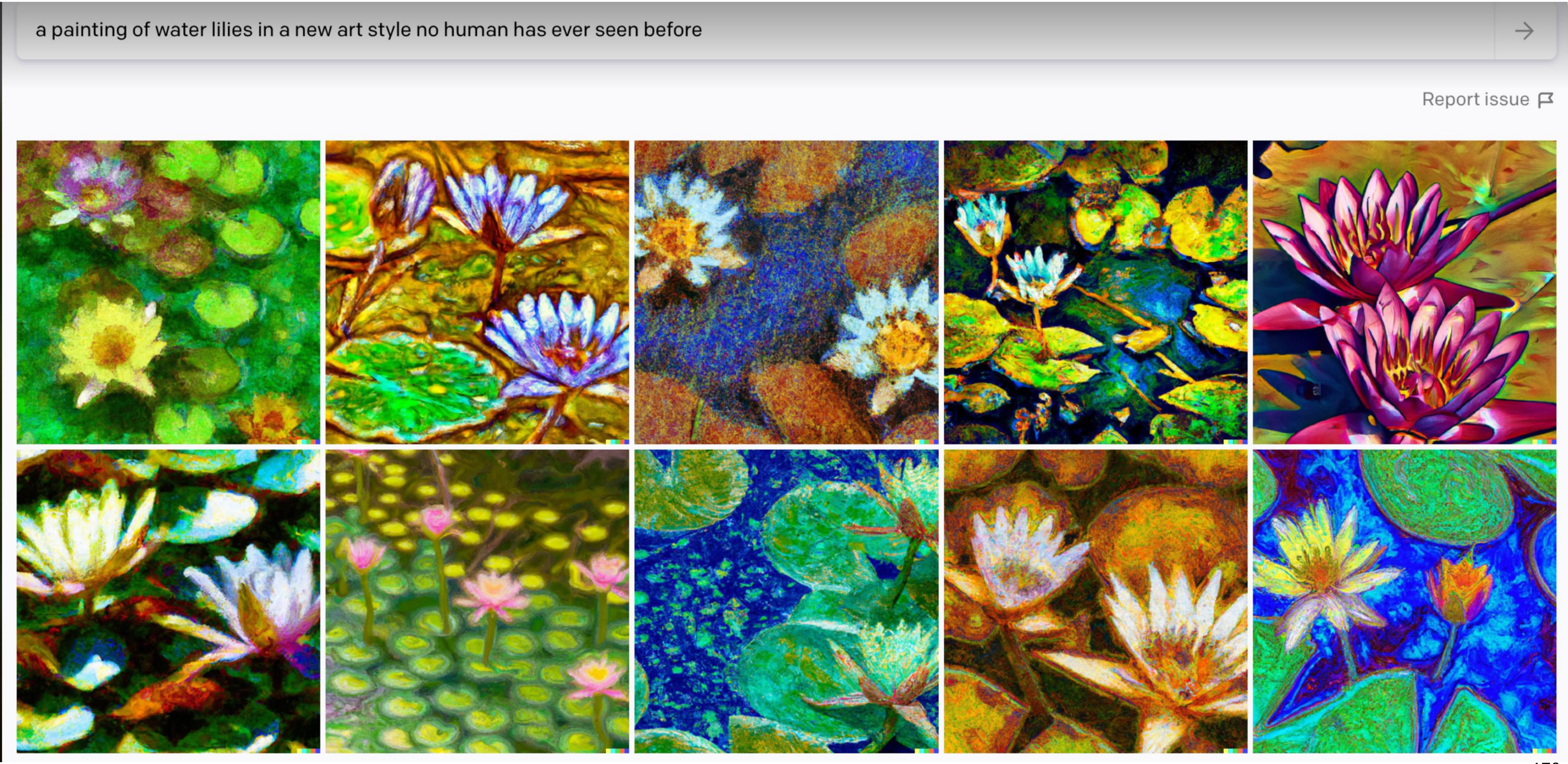


TEXT PROMPT an illustration of a small green mouse sitting below a large red elephant

AI-GENERATED IMAGES



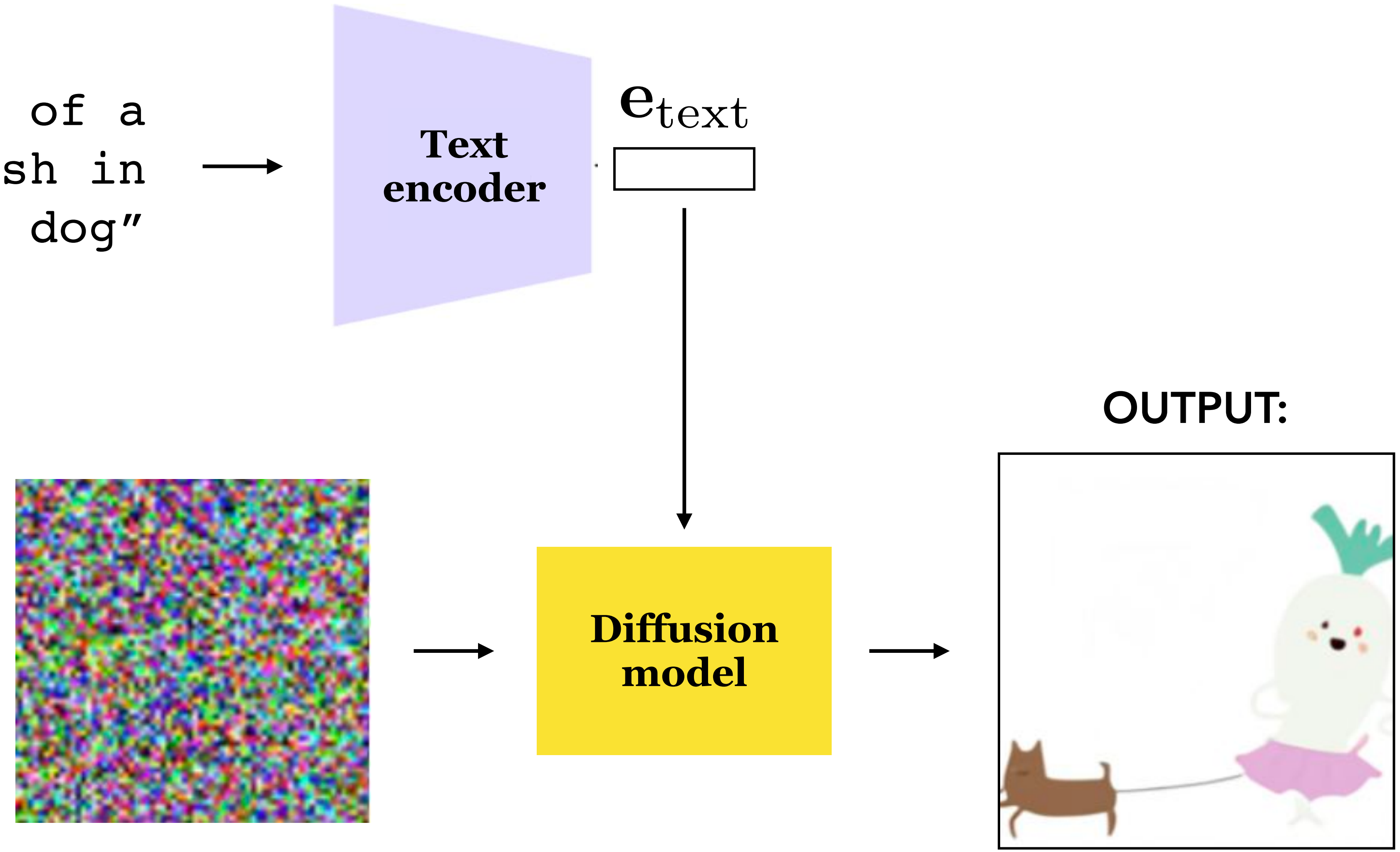
DALL-E 2


a painting of water lilies in a new art style no human has ever seen before
Report issue

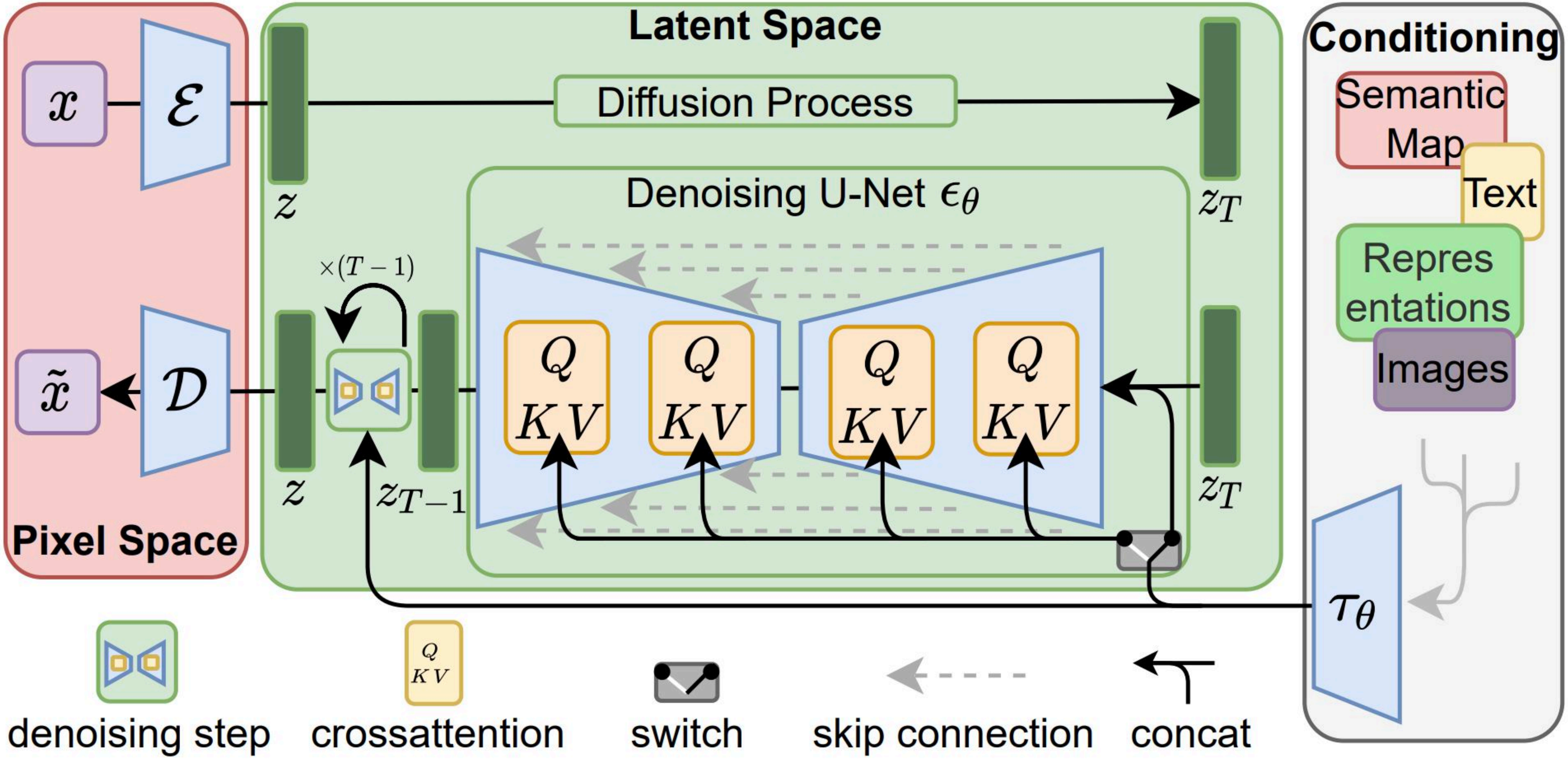
DALL-E 2 [Ramesh et al. 2022] https://cdn.openai.com/papers/dall-e-2.pdf

INPUT:

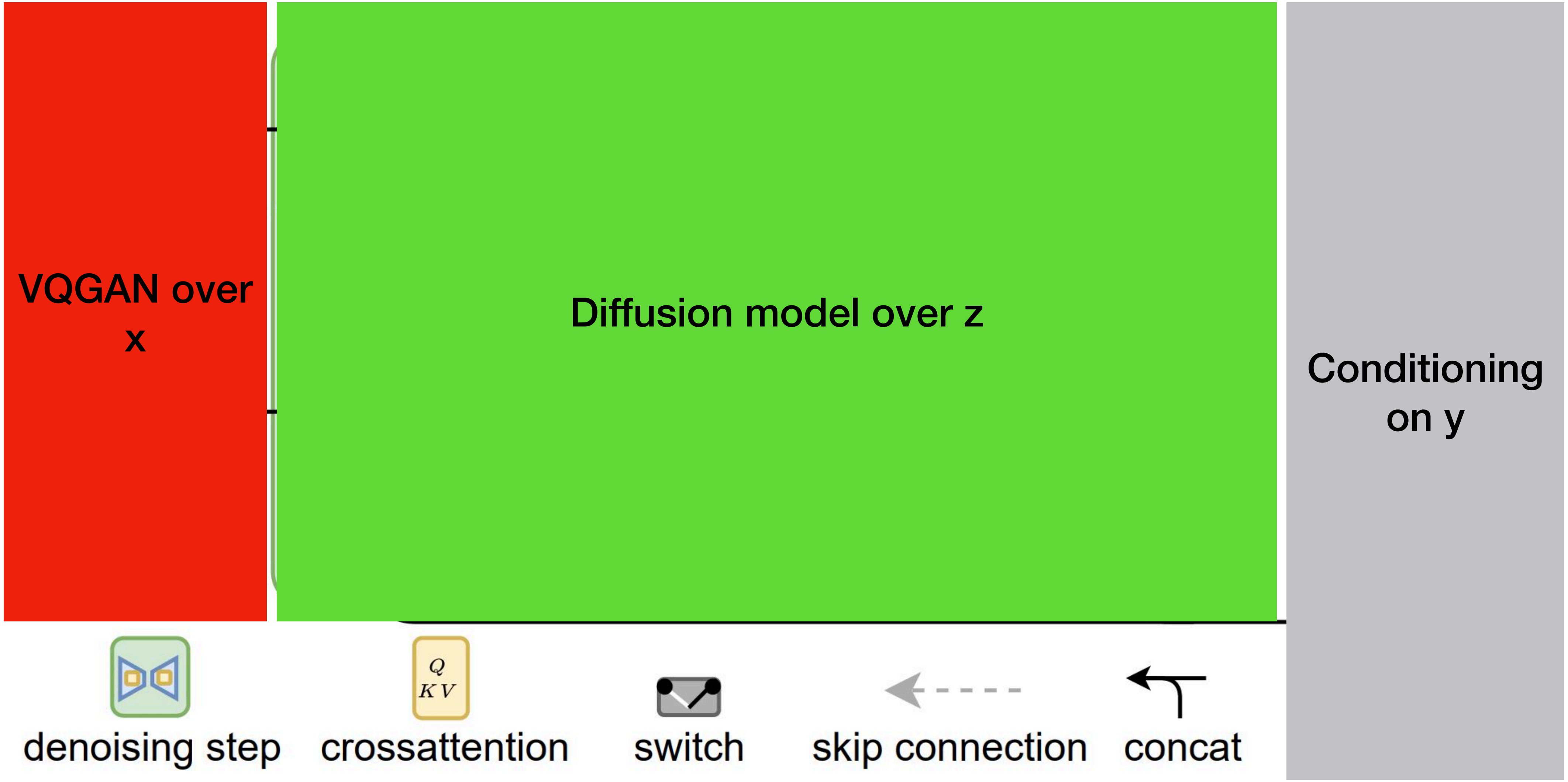
“An illustration of a baby daikon radish in a tutu walking a dog”



Latent diffusion [Rombach*, Blattman* et al. 2022] https://arxiv.org/abs/2112.10752



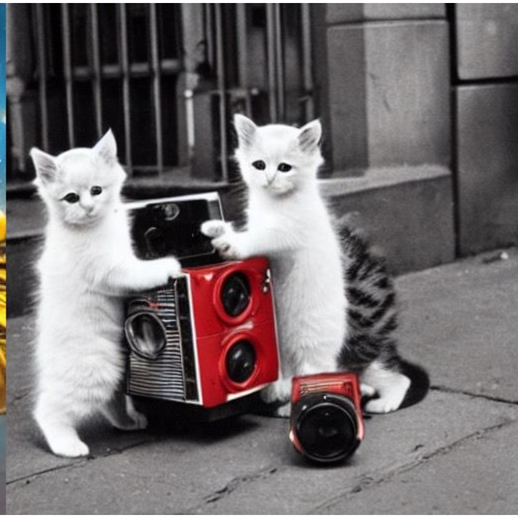
Latent diffusion [Rombach*, Blattman* et al. 2022] https://arxiv.org/abs/2112.10752



Stable Diffusion [Rombach*, Blattman* et al. 2022] https://arxiv.org/abs/2112.10752







Stable Diffusion Applications: Twitter Mega Thread

https://twitter.com/daniel_eckler/status/1572210382944538624

slide adapted from Roni Sengupta

IMAGEN

[Saharia∗, Chan∗, Saxena† et al. 2022] https://arxiv.org/pdf/2205.11487.pdf

Sprouts in the shape of text 'Imagen' coming out of a fairytale book.

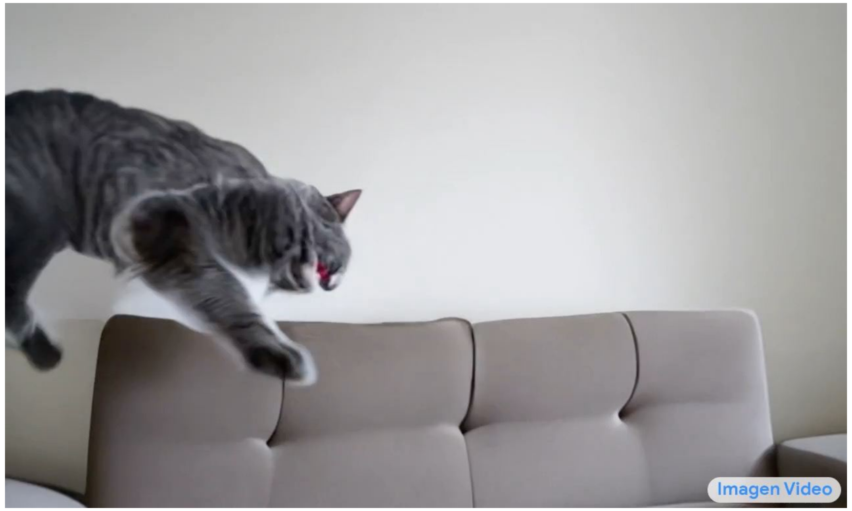
A photo of a Shiba Inu dog with a backpack riding a bike. It is wearing sunglasses and a beach hat.

A high contrast portrait of a very happy fuzzy panda dressed as a chef in a high end kitchen making dough. There is a painting of flowers on the wall behind him.

IMAGEN Video

[Ho et al., "Imagen Video", 2022] https://arxiv.org/abs/2210.02303

A teddy bear
running in New York City



A british shorthair
jumping over a coach

A swarm of bees
flying around their hive

Generating Videos from Text - Sora

A stylish woman walks down a Tokyo street filled with warm glowing neon and animated city signage. She wears a black leather jacket, a long red dress, and black boots, and carries a black purse. She wears sunglasses and red lipstick. She walks confidently and casually. The street is damp and reflective, creating a mirror effect of the colorful lights. Many pedestrians walk about.

Video credit: OpenAI

Generating Videos from Text - Sora

A young man at his 20s is sitting on a piece of cloud in the sky, reading a book.

Video credit: OpenAI

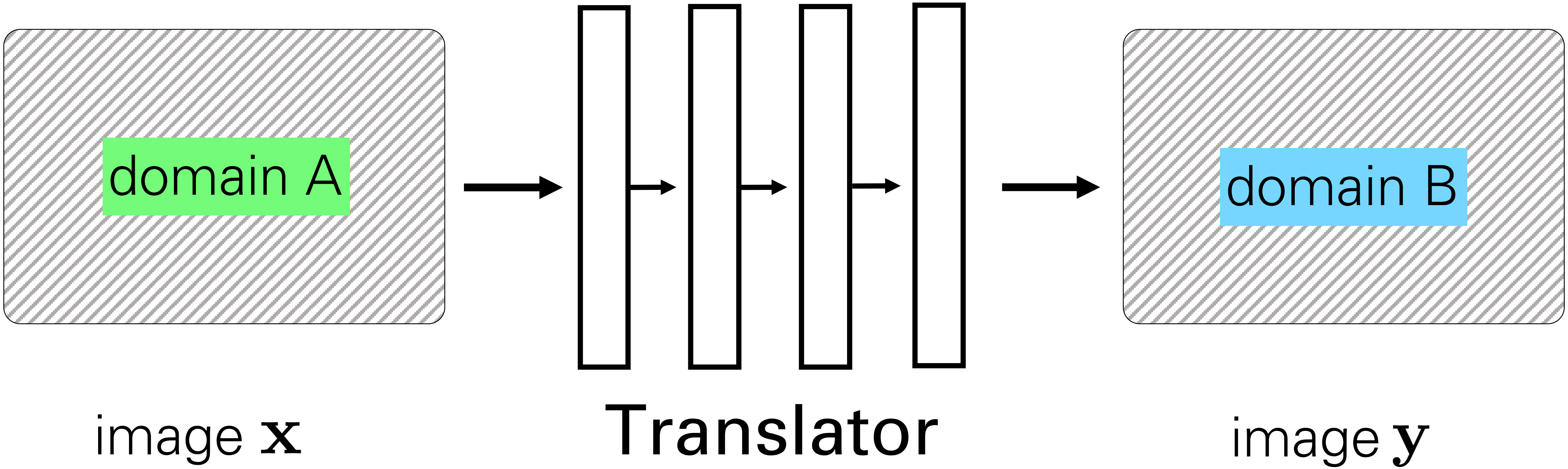
Generating Videos from Text - Sora

The camera directly faces colorful buildings in burano italy. An adorable dalmation looks through a window on a building on the ground floor. Many people are walking and cycling along the canal streets in front of the buildings.

Video credit: OpenAI

Applications in Computational Photography

Image-to-Image Translation



- Learning to map images from one domain into another
- A general framework for both low and high-level image processing

Super-resolution

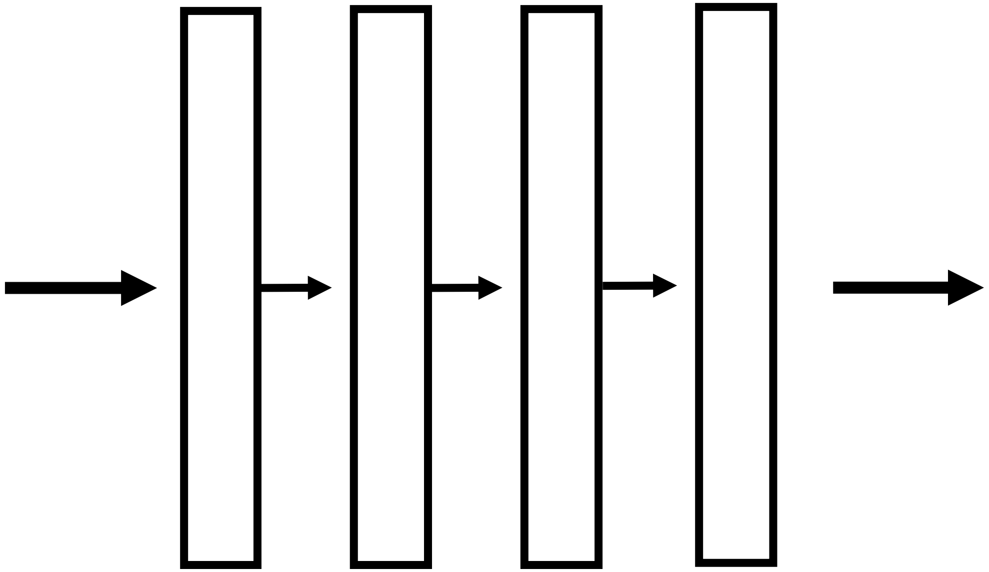


image $\mathbf{x}$ Translator image $\mathbf{y}$

- **Input:** low-res image
  Output: high-res image

C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, W. Shi. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. CVPR 2017.

Colorization

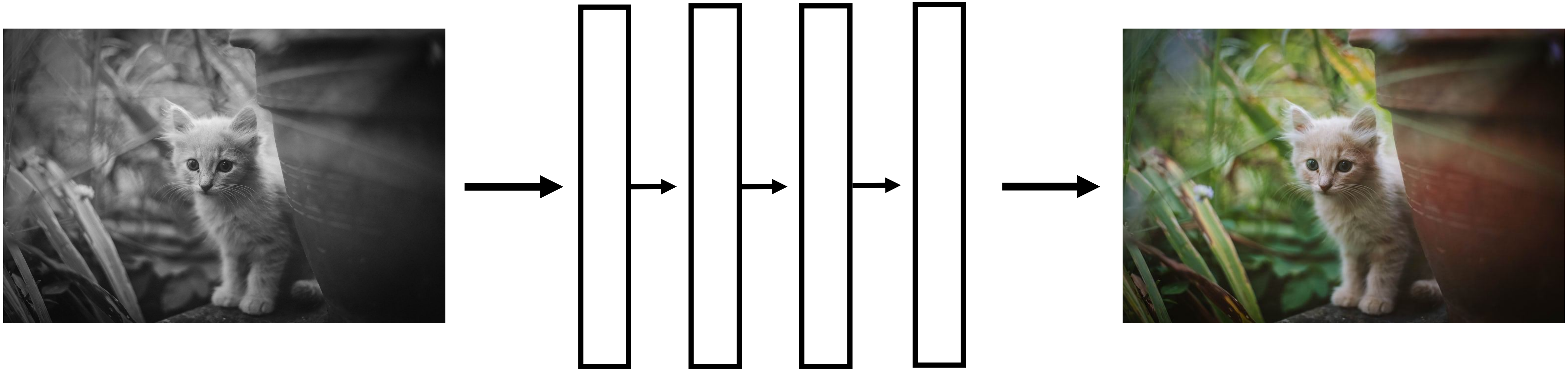


image $\mathbf{x}$ Translator image $\mathbf{y}$

- **Input:** black & white image
  Output: color image

Jason Antic. DeOldify: **A Deep Learning based project for colorizing and restoring old images.** 2018

BW → Color

Input Output Input Output Input Output

Data from [Russakovsky et al. 2015]

Day to Night

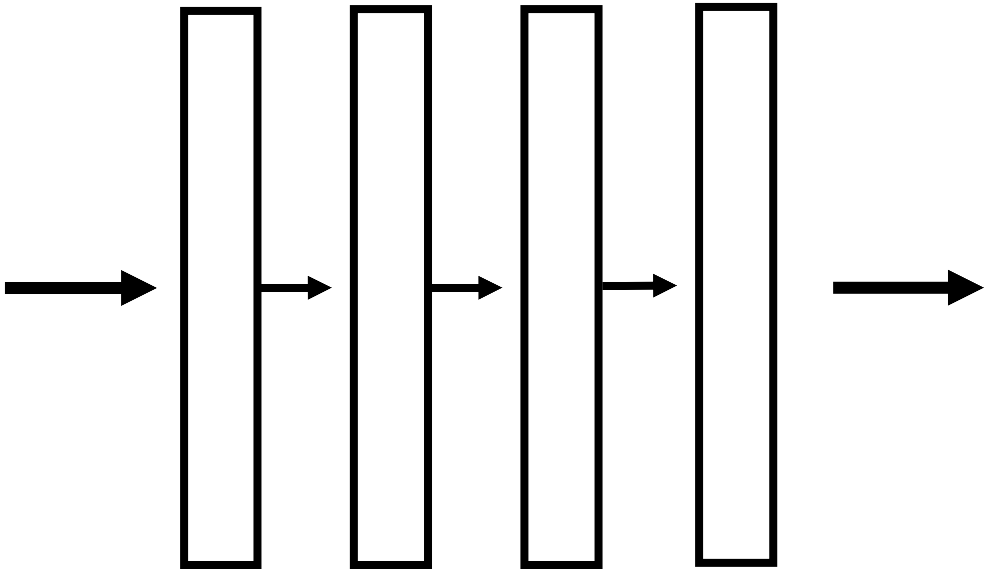


image $\mathbf{x}$

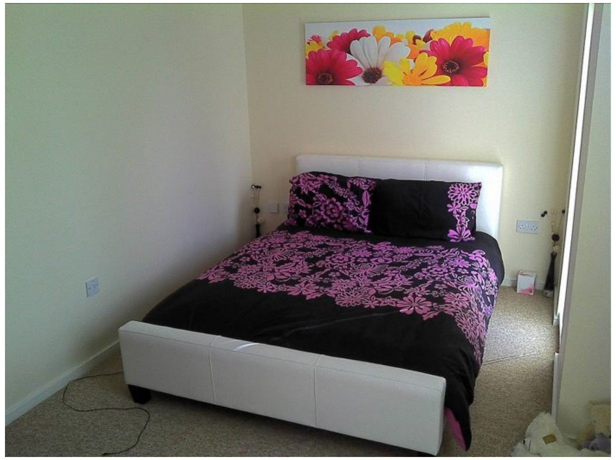
Translator

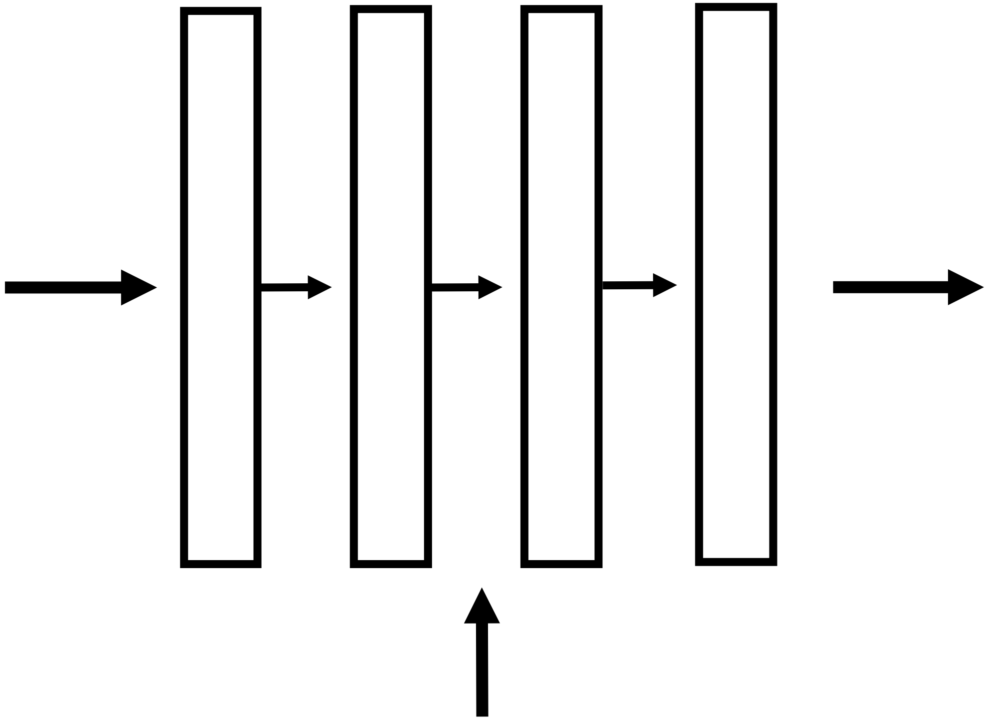
image $\mathbf{y}$

- **Input:** day image
  Output: night image

Ming-Yu Liu, Thomas Breuel, Jan Kautz. Unsupervised Image-to-Image Translation Networks. NIPS 2017.

Photo Style Transfer





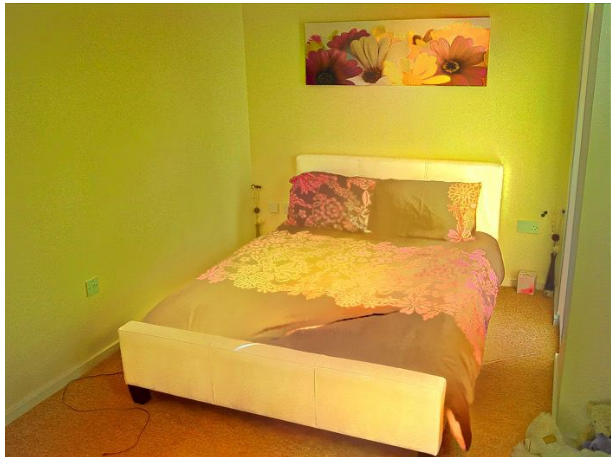
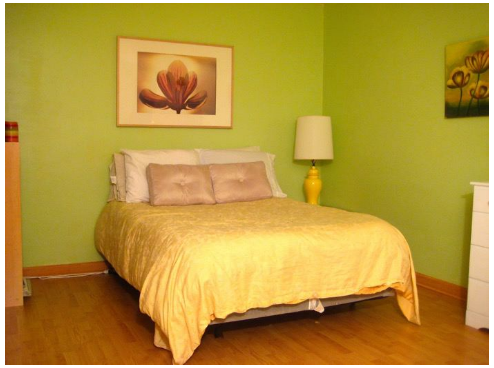


image $\mathbf{x}$

image $\mathbf{y}$

- **Input:** input image
  + target style image

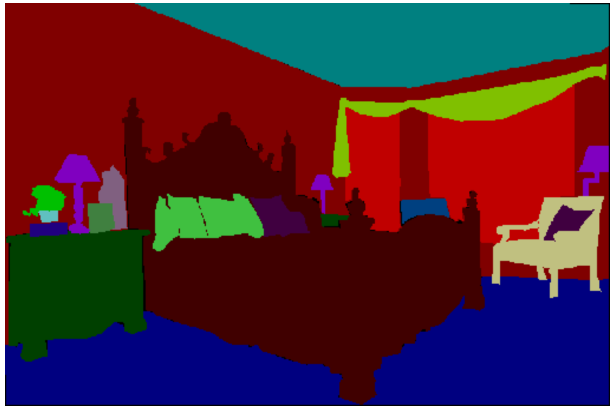
  Output: synthesized image



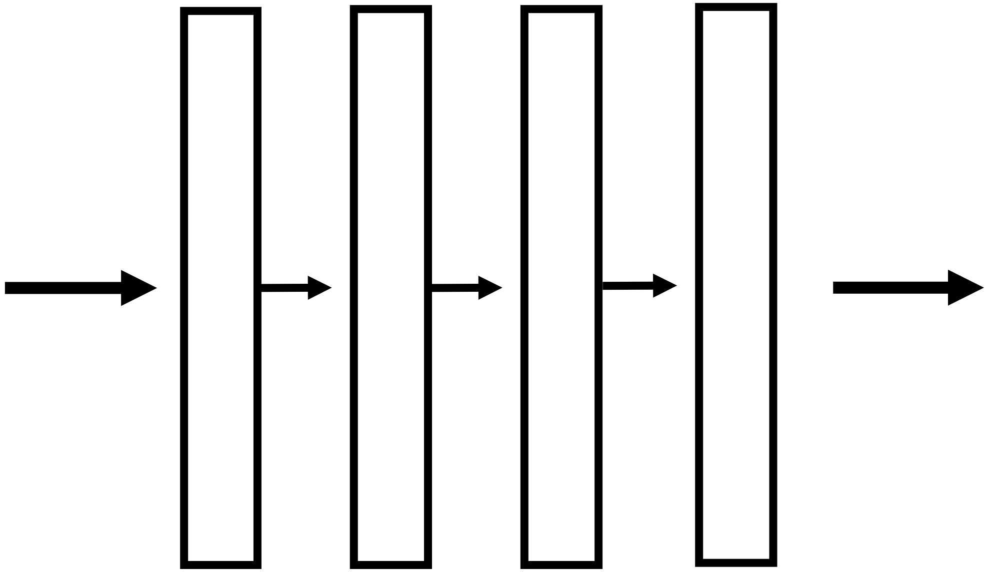
Target style image $\mathbf{t}$

Fujun Luan, Sylvain Paris, Eli Shechtman, Kavita Bala. **Deep Photo Style Transfer**. CVPR 2017.

Semantic Image Synthesis



semantic
layout $\mathbf{x}$

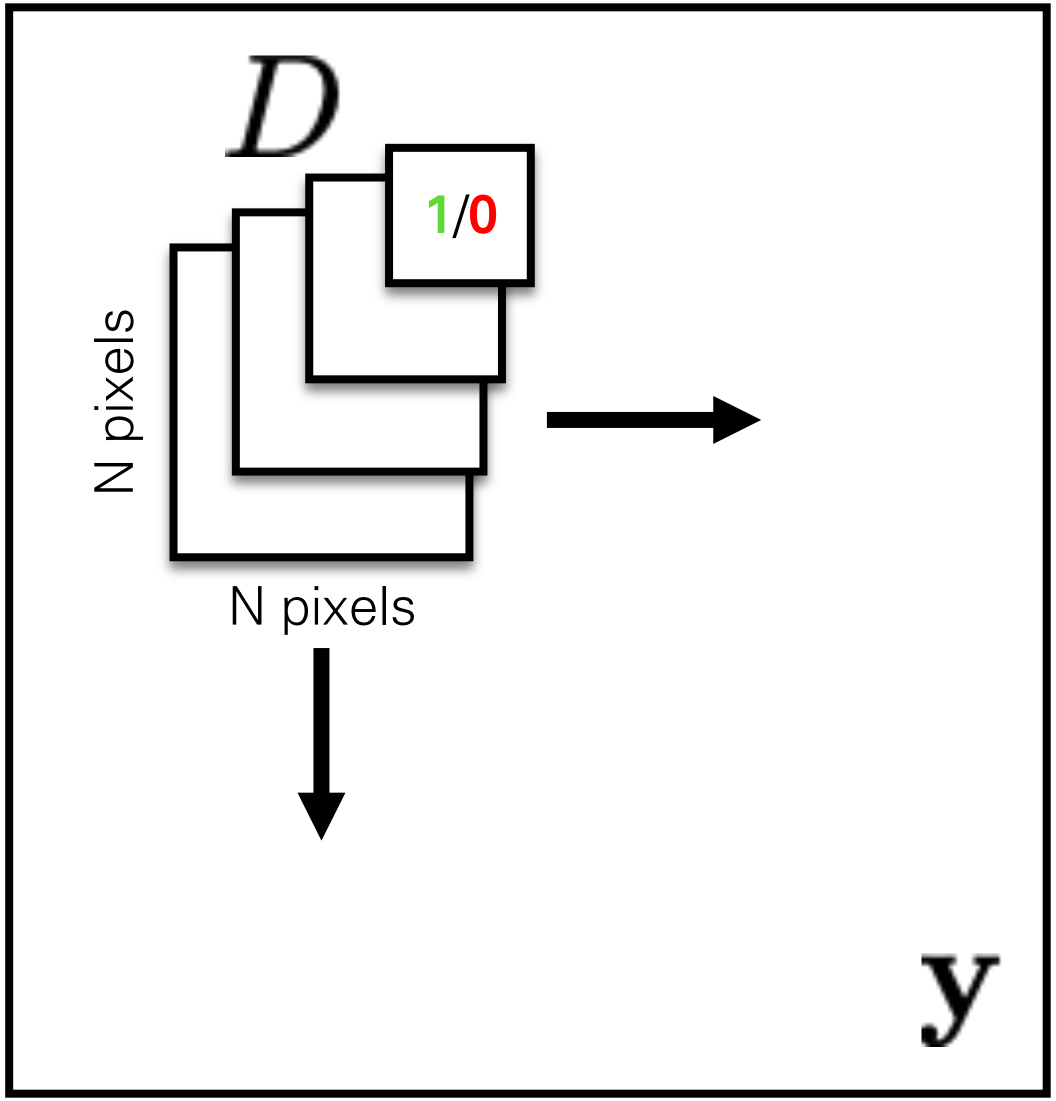


synthesized
image $\mathbf{y}$

- **Input:** input layout
  Output: synthesized image

T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, B. Catanzaro. **High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs**. CVPR 2018.

Shrinking the capacity: Patch Discriminator



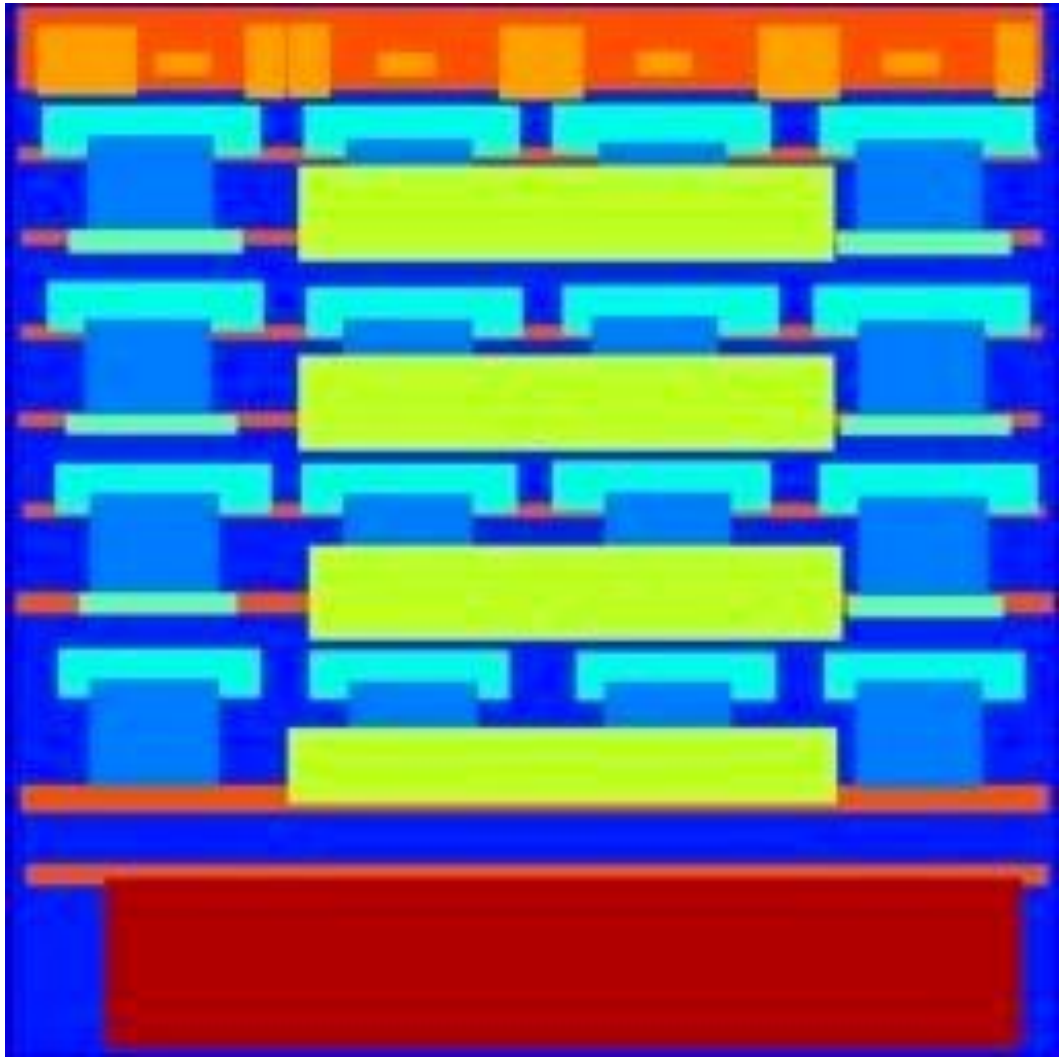
Rather than penalizing if output image looks fake, penalize if each overlapping patch in output looks fake

- Faster, fewer parameters
- More supervised observations
- Applies to arbitrarily large images

[Li & Wand 2016]
[Shrivastava et al. 2017]
[Isola et al. 2017]

Slide Credit: Philip Isola

Labels → Facades

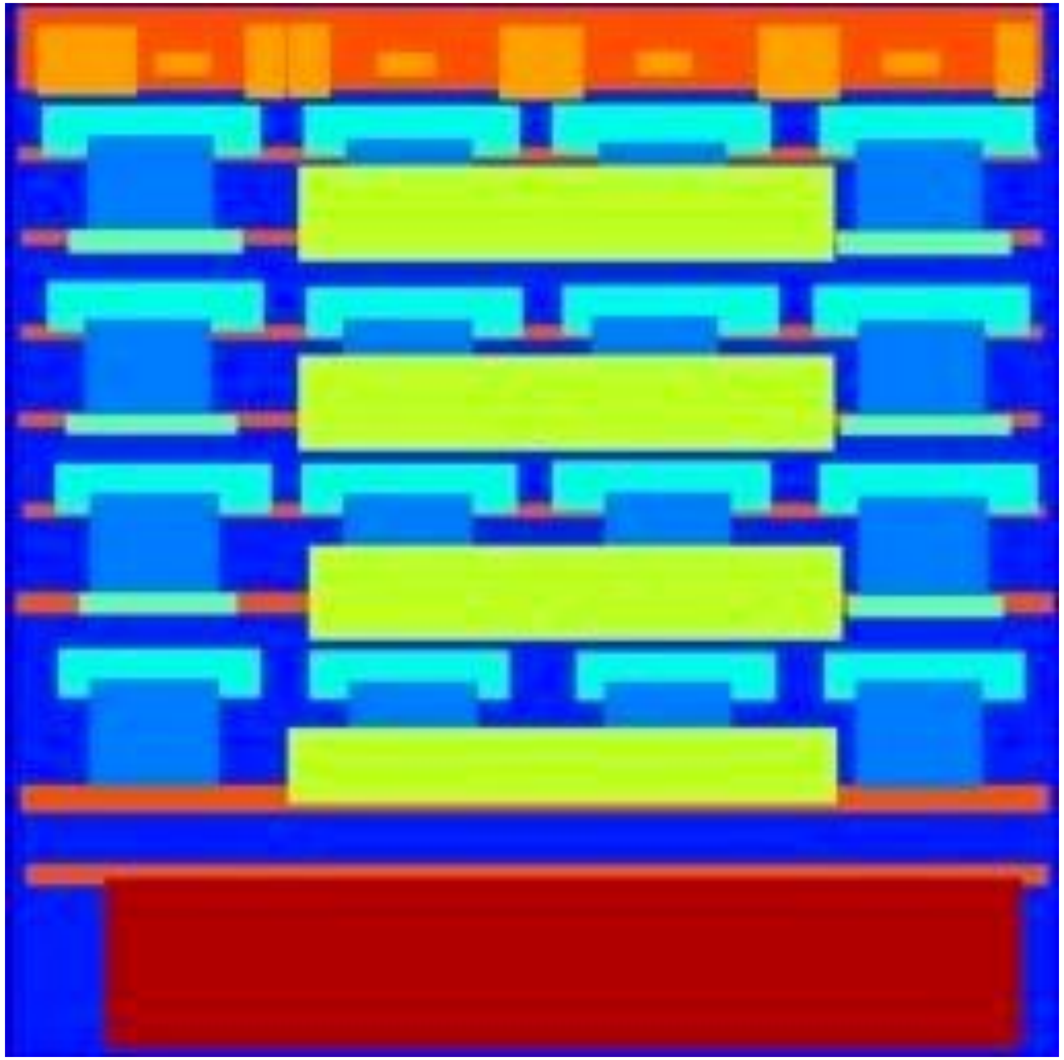


Data from [Tylecek, 2013]

Slide Credit: Philip Isola

Labels → Facades

Input



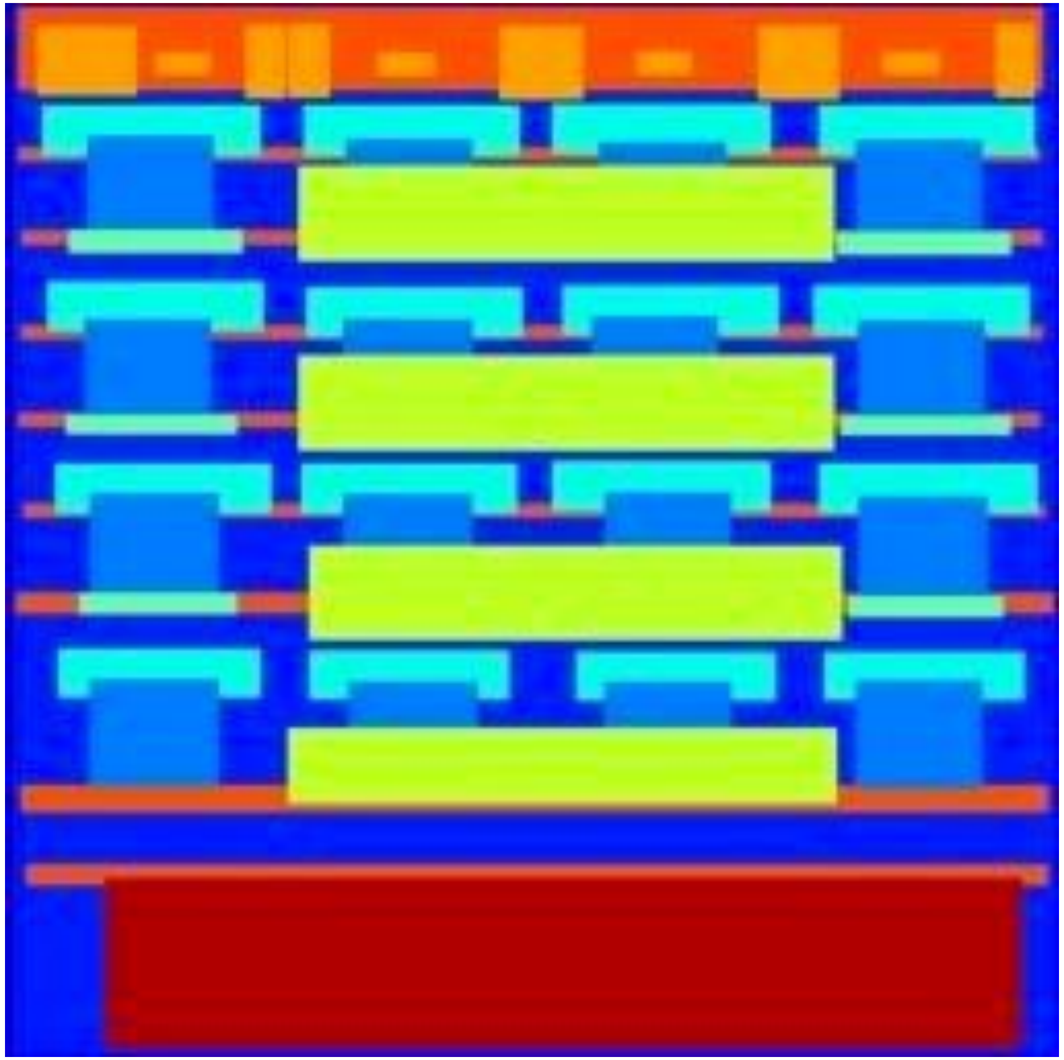
16x16 Discriminator

Data from [Tylecek, 2013]

Slide Credit: Philip Isola

Labels → Facades

Input



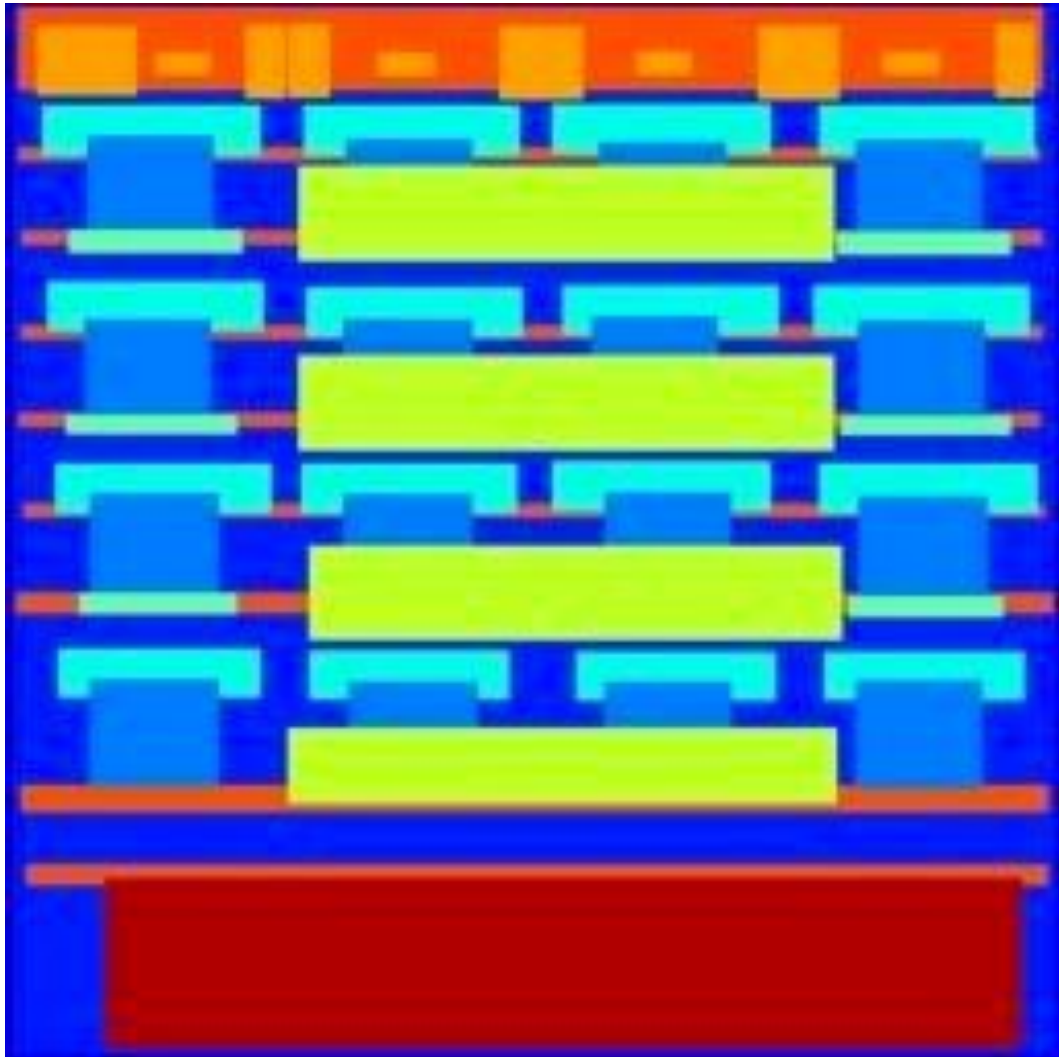
70x70 Discriminator

Data from [Tylecek, 2013]

Slide Credit: Philip Isola

Labels → Facades

Input



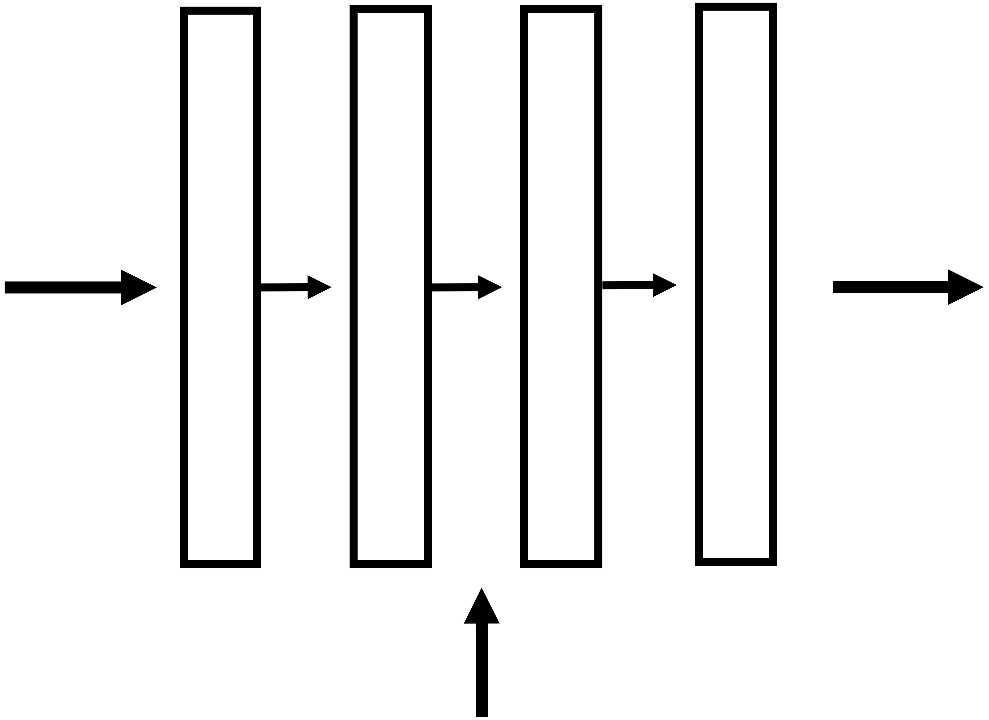
Full image Discriminator

Data from [Tylecek, 2013]

Slide Credit: Philip Isola

Semantic Image Synthesis

semantic
layout $\mathbf{x}$



synthesized
image $\mathbf{y}$

- **Input:** input layout
  + target style image

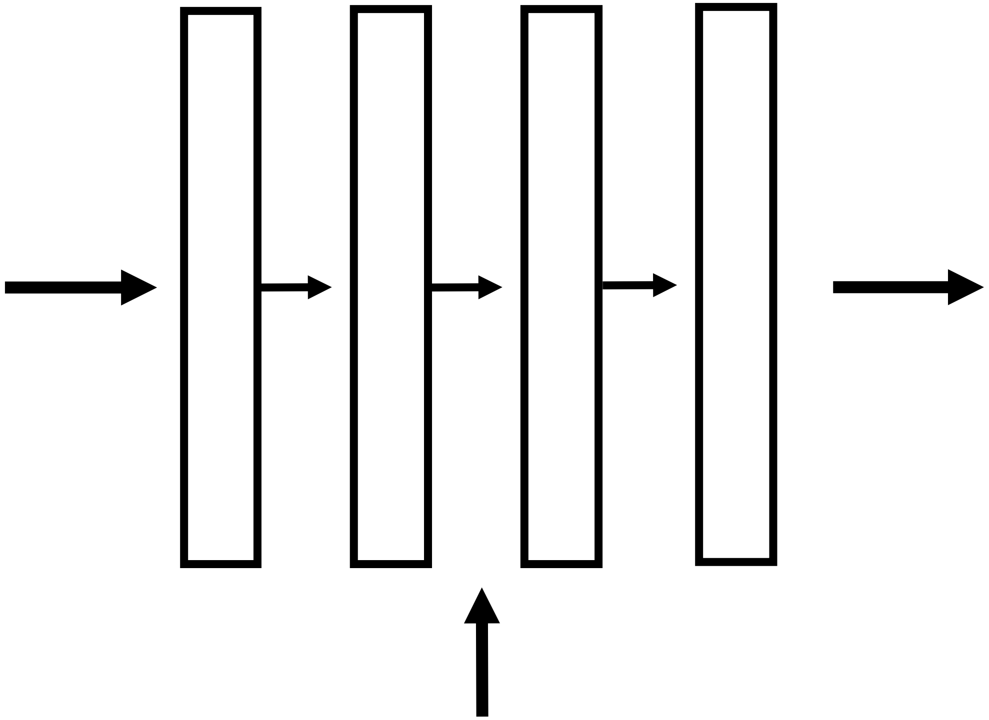
  Output: synthesized image

Target style
image $\mathbf{t}$

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, Jun-Yan Zhu. **Semantic Image Synthesis with Spatially-Adaptive Normalization**. CVPR 2020.

Semantic Image Editing

image $\mathbf{x}$



“more flowers”
“more cloudy”

Target scene attribute(s) $\mathbf{t}$

manipulated image $\mathbf{y}$

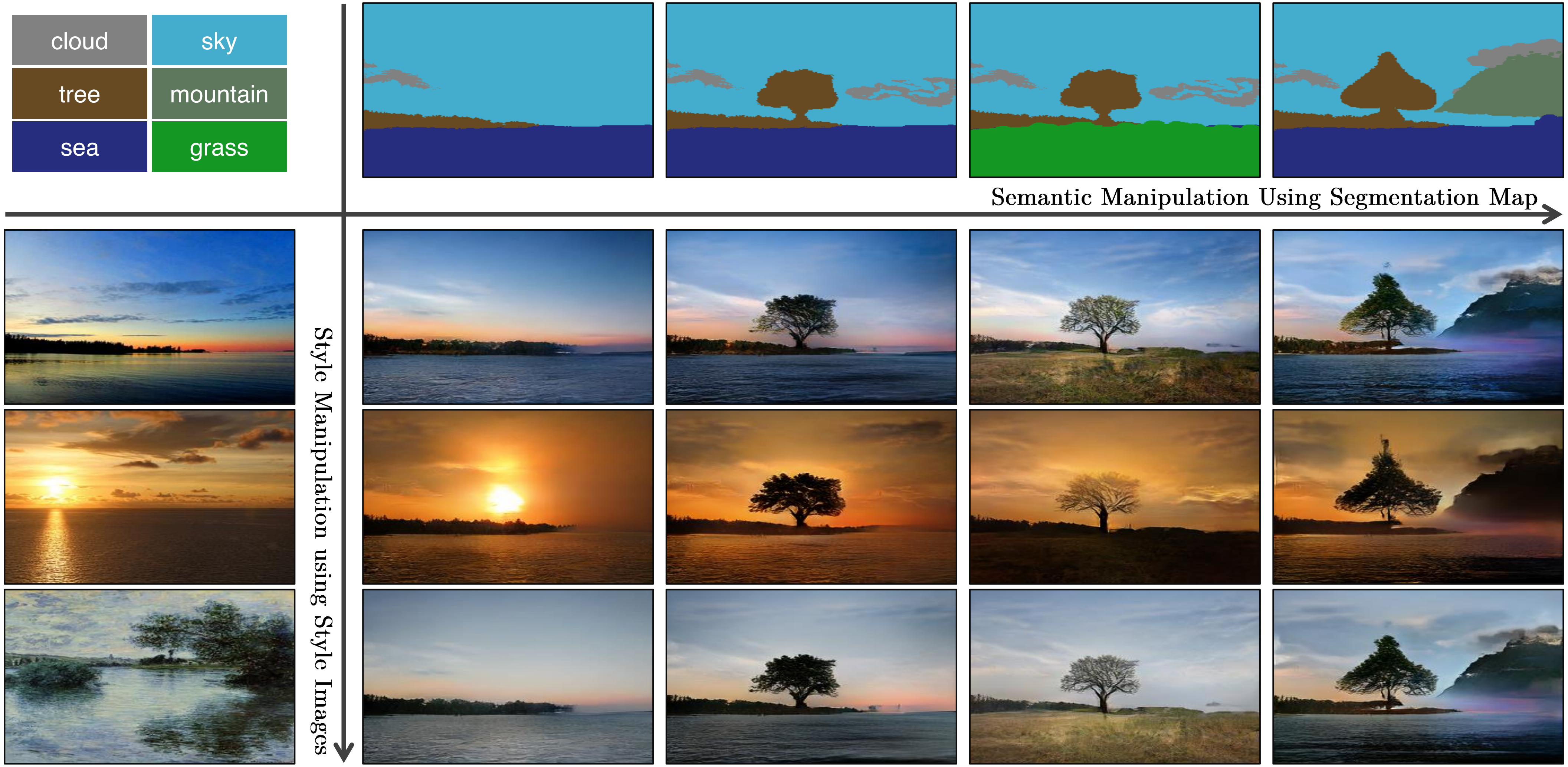
- **Input:** input image
  (+ semantic layout)
  + target attribute

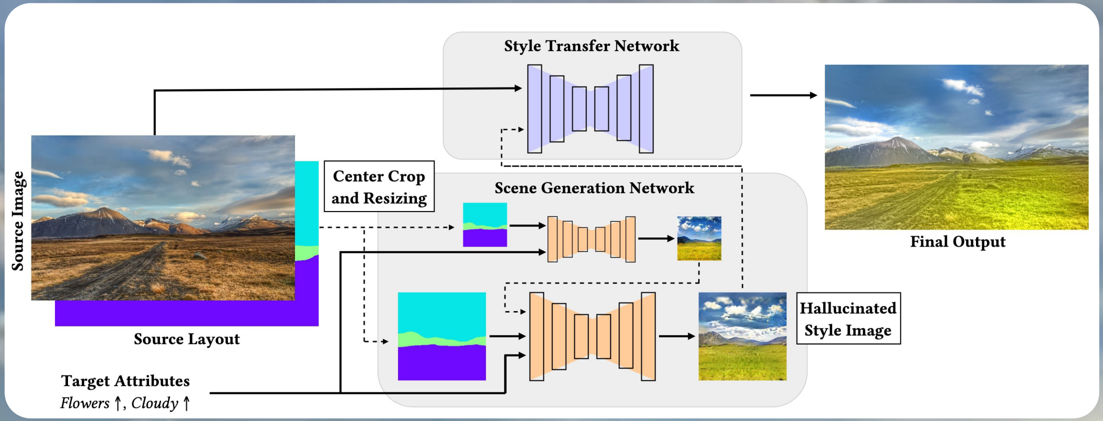
  Output: manipulated image

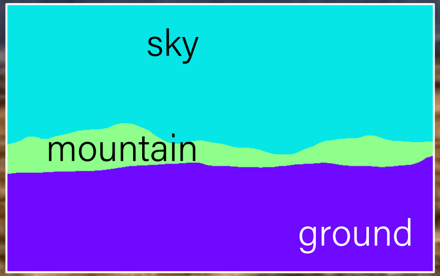
Levent Karacan, Zeynep Akata, Aykut Erdem, Erkut Erdem. **Manipulating Attributes of Natural Scenes via Hallucination.** ACM TOG 2020.

Semantic Image Synthesis (SPADE) (Park et al., 2019)

- Image generation conditioned on semantic layouts







Semantic layout

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

winter

prediction

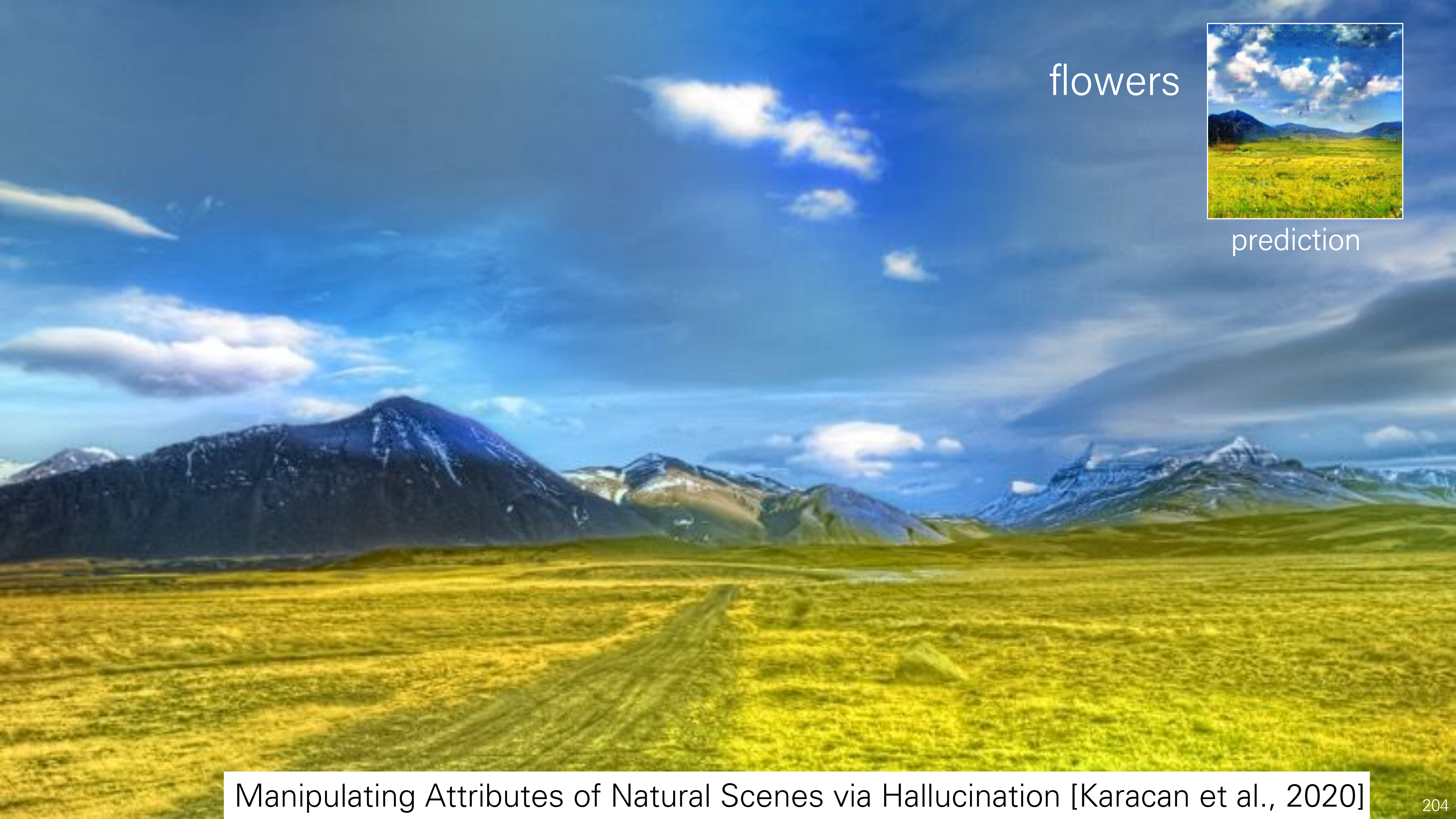
Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Moist, rain and fog

prediction

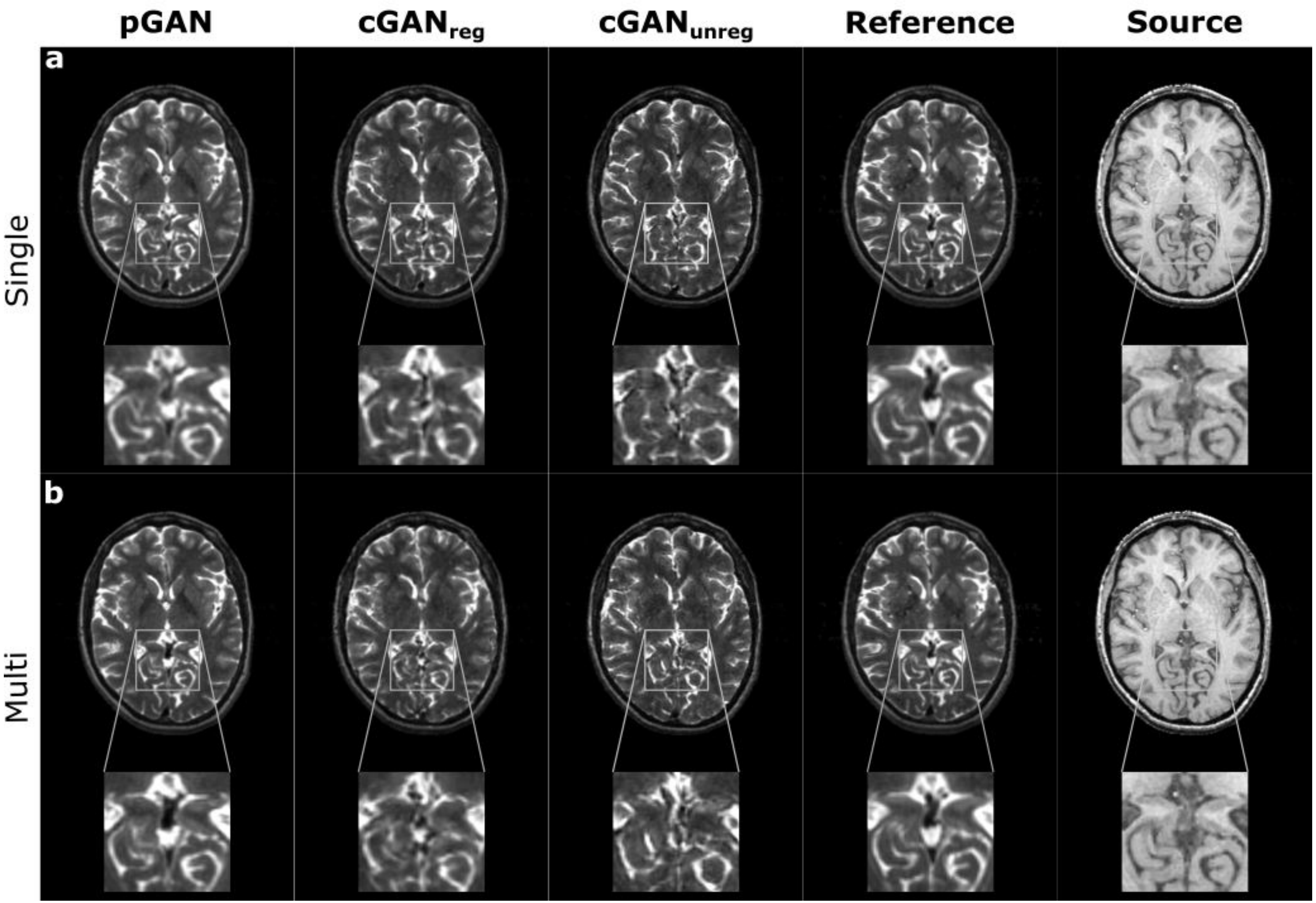
Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

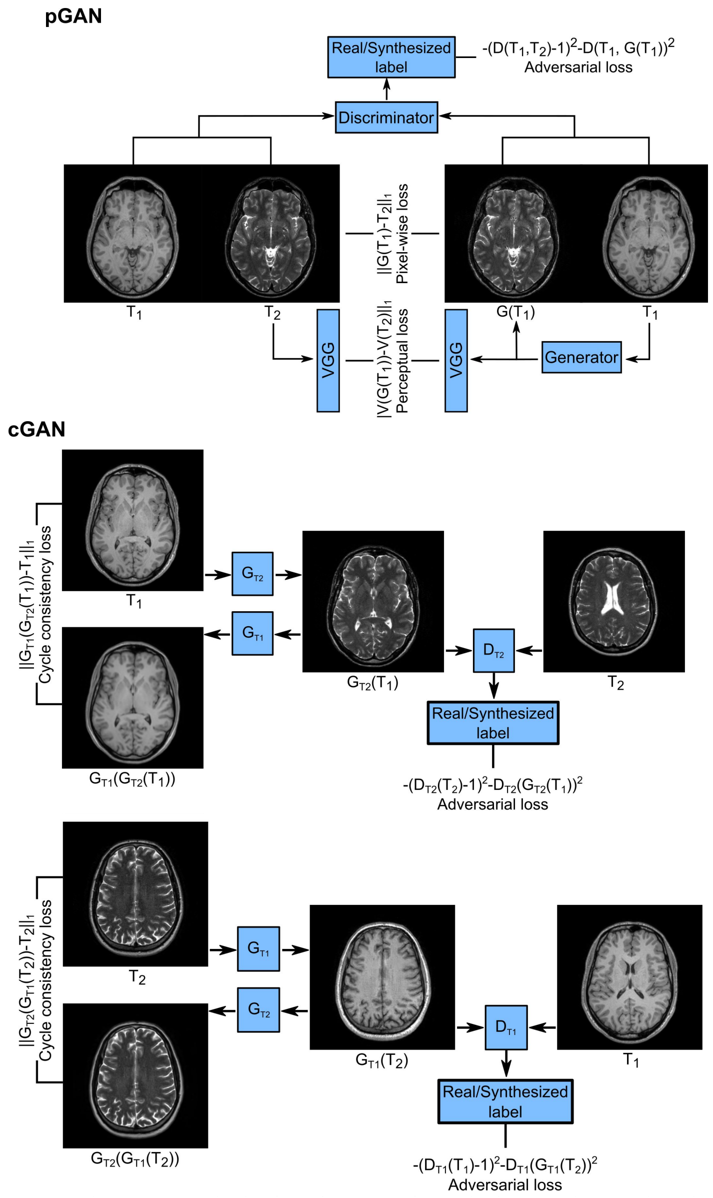


flowers

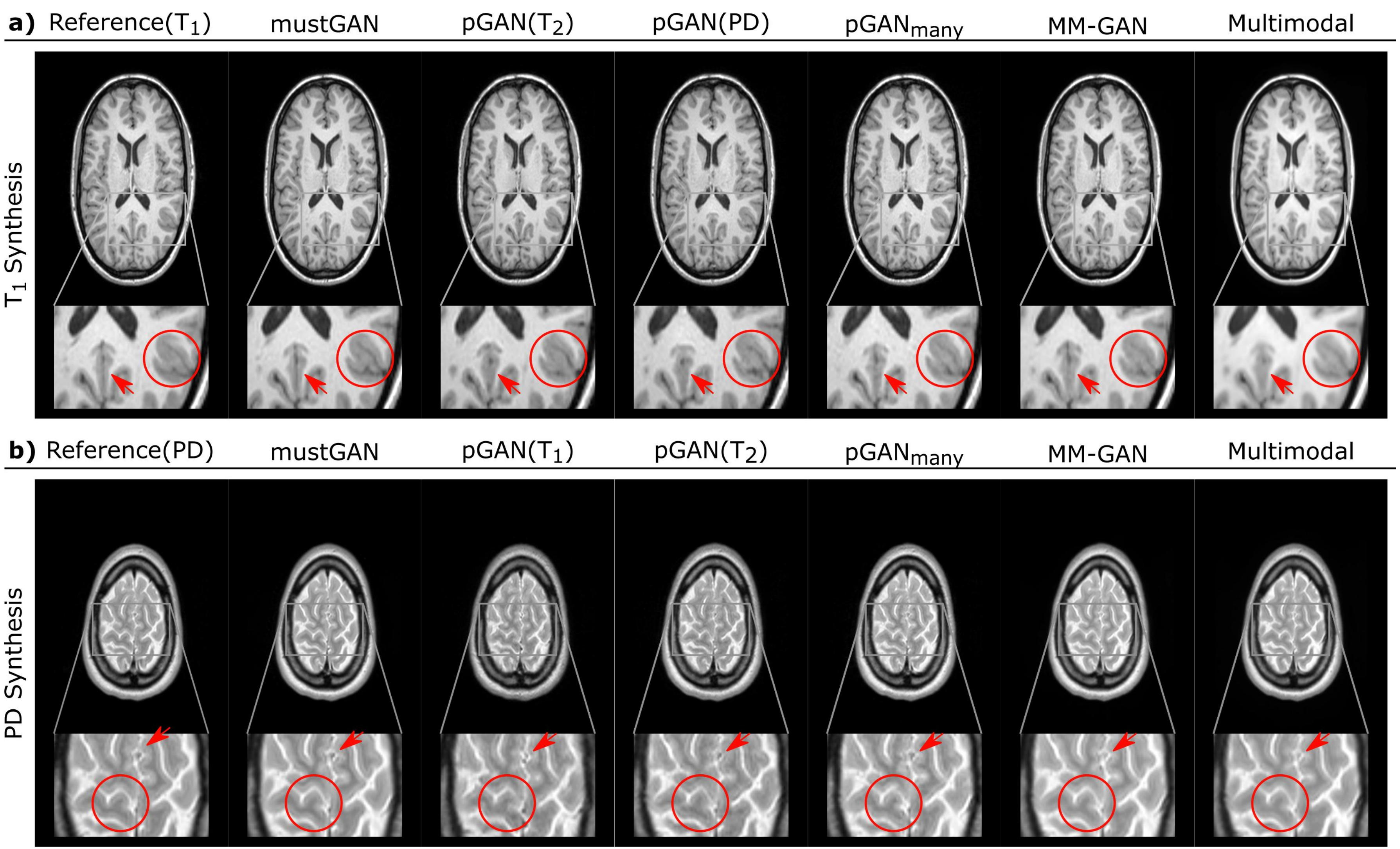
prediction

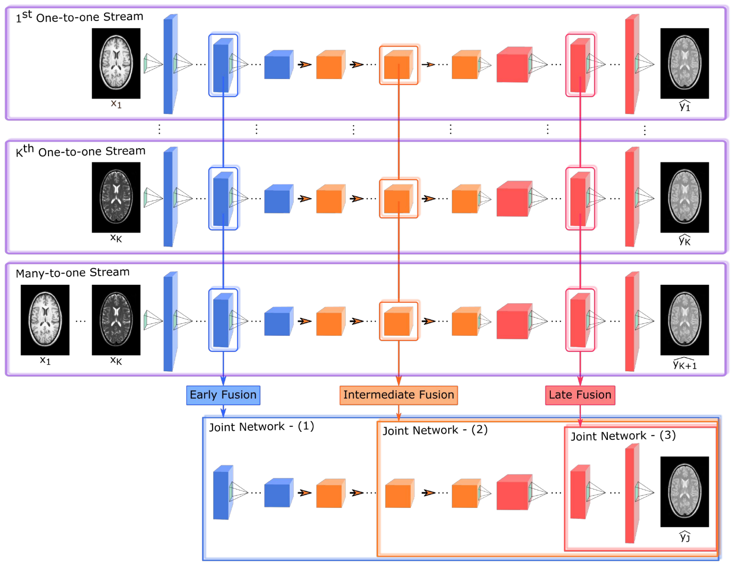
Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]





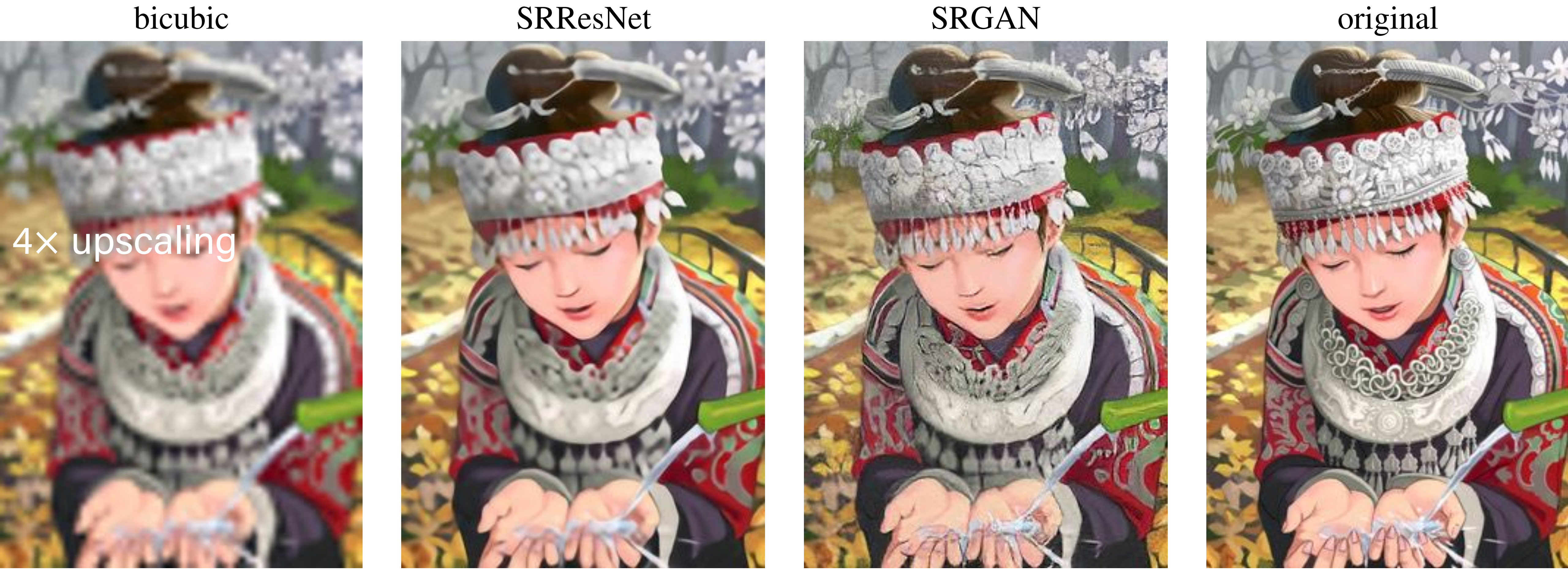
- Image Synthesis in Multi-Contrast MRI [Ul Hassan Dar et al. 2019]





- Image Synthesis in Multi-Contrast MRI [Mahmut Yurt et al. 2021]

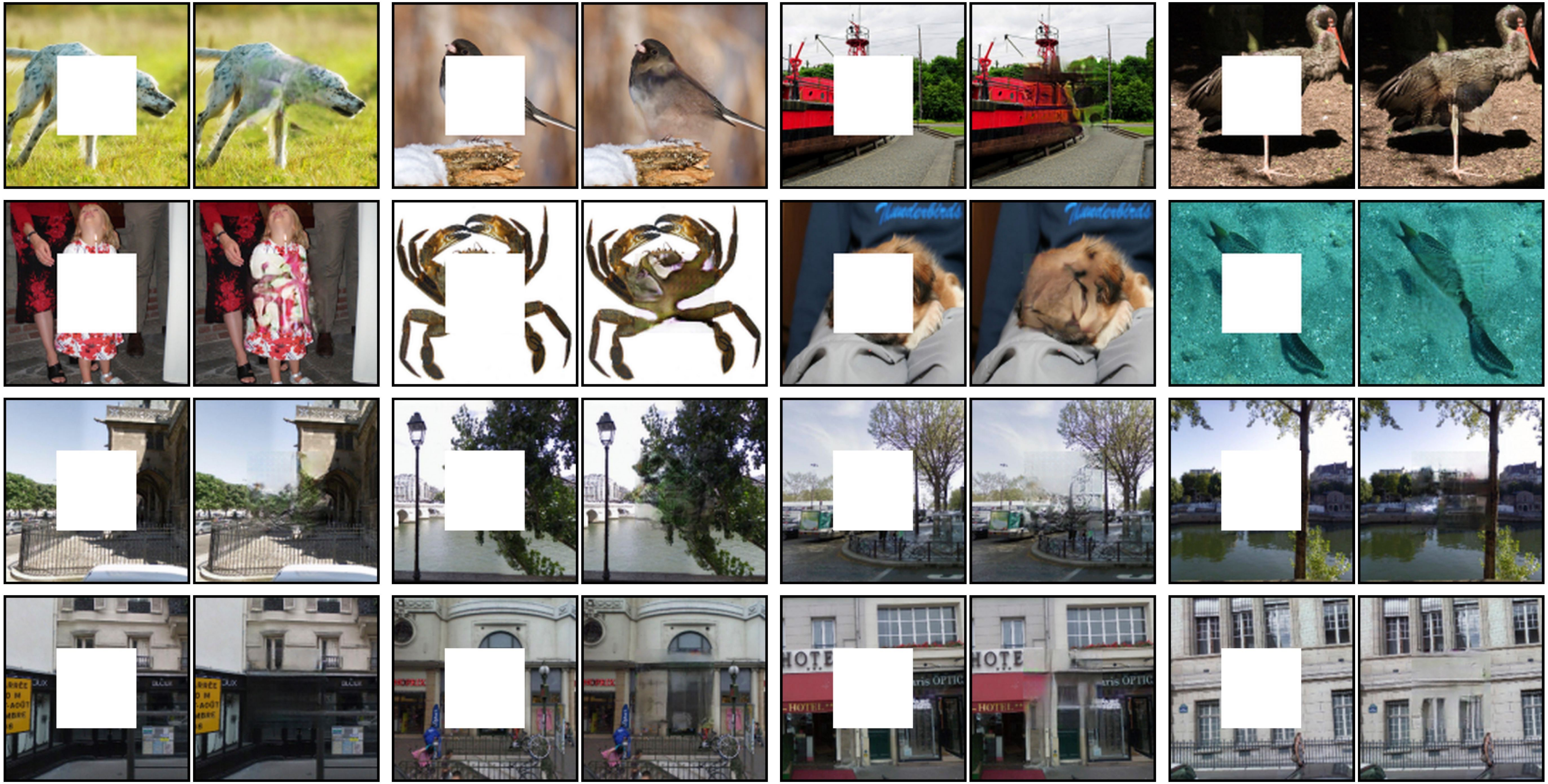
Single Image Super-Resolution



- Key idea: Combine content loss with adversarial loss

(Ledig et al., CVPR 2016)

Image Inpainting (Pathak et al., 2016)



- Key idea: Combine content loss with adversarial loss

(Ledig et al., CVPR 2016)

Image Deblurring

- non-uniform motion blur from a single blurry image.
- **Key idea:** Use multiscale CNNs to restore sharp images in an end-to-end manner

$$\mathcal{L}_{cont} = \frac{1}{2K}\sum_{k=1}^{K}\frac{1}{c_k w_k h_k}\|L_k - S_k\|^2$$

- can be interpreted as a kind of image to image translation

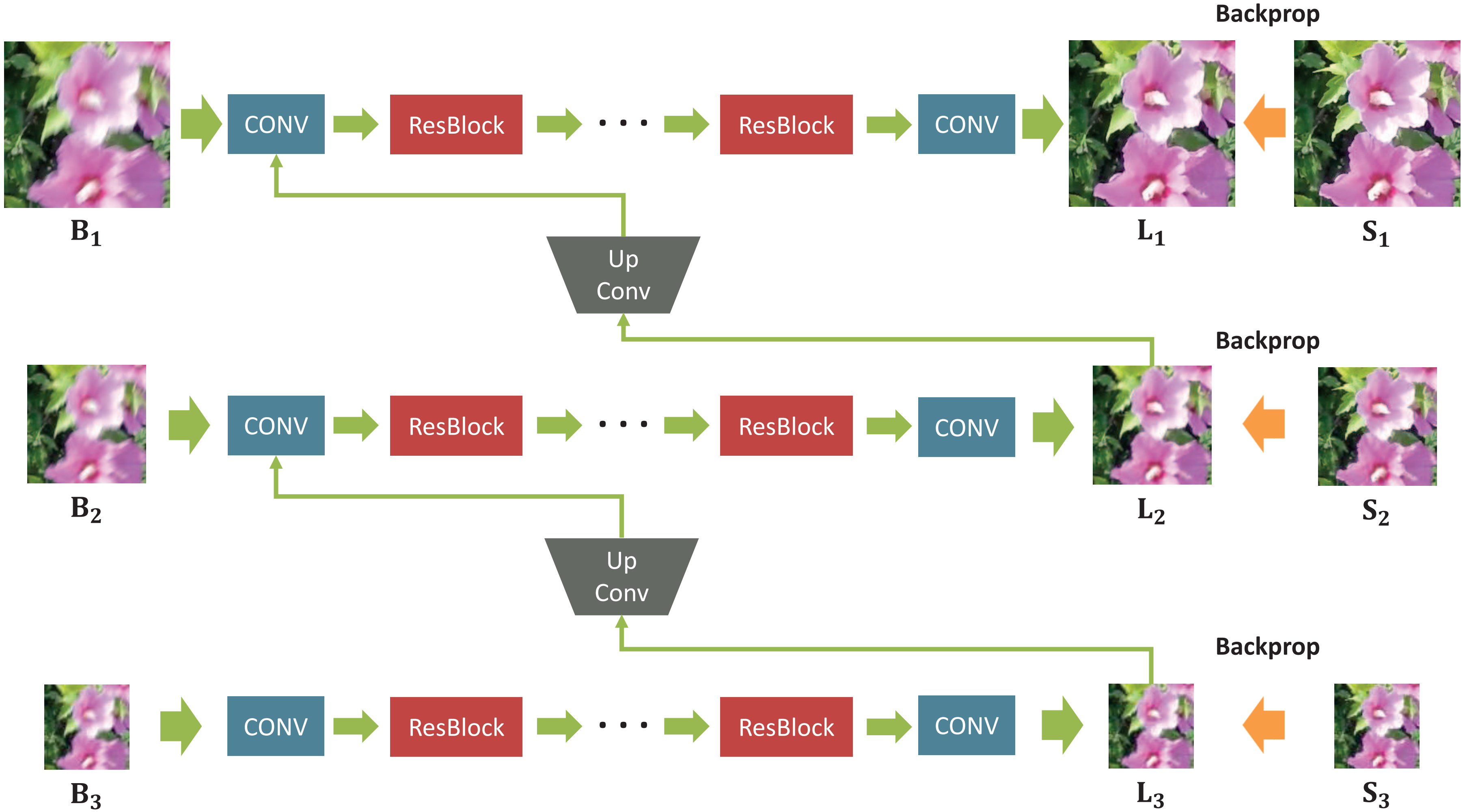
209

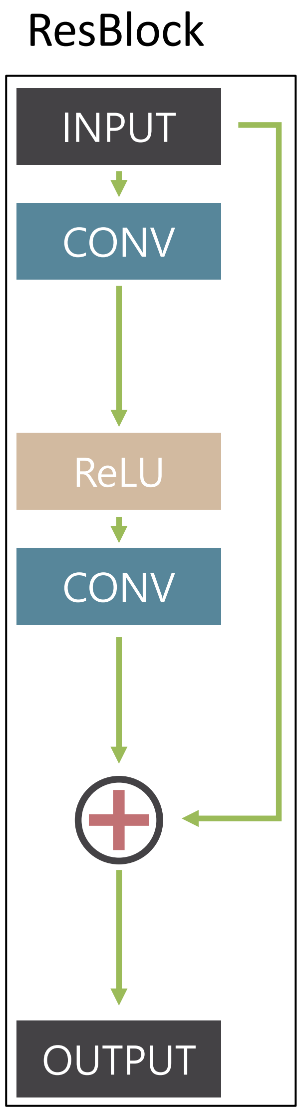
- An additional adversarial loss

$$\mathcal{L}_{adv} = \underset{S\sim p_{sharp}(S)}{\mathbb{E}}[\log D(S)] + \underset{B\sim p_{blurry}(B)}{\mathbb{E}}[\log(1 - D(G(B)))]$$

(Nah et al., CVPR 2017)

Image Deblurring

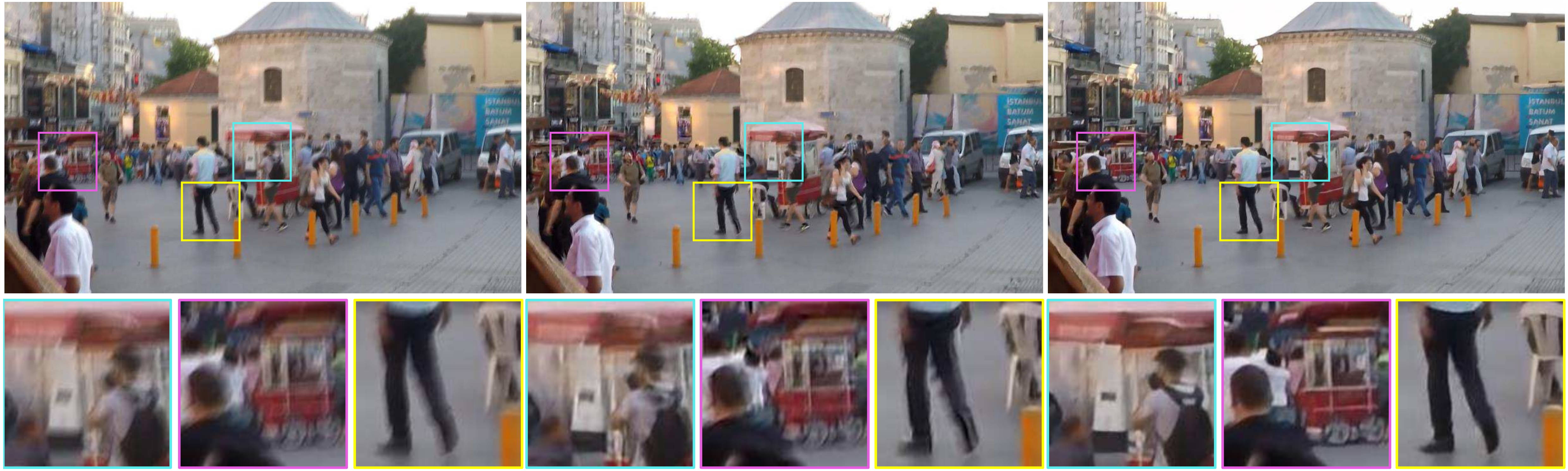




- Coarser scale features aid finer scale image deblurring

(Nah et al., CVPR 2017)

Image Deblurring



Blurred images | Sun et al., CVPR 2015 | Nah et al., CVPR 2017

(Nah et al., CVPR 2017)

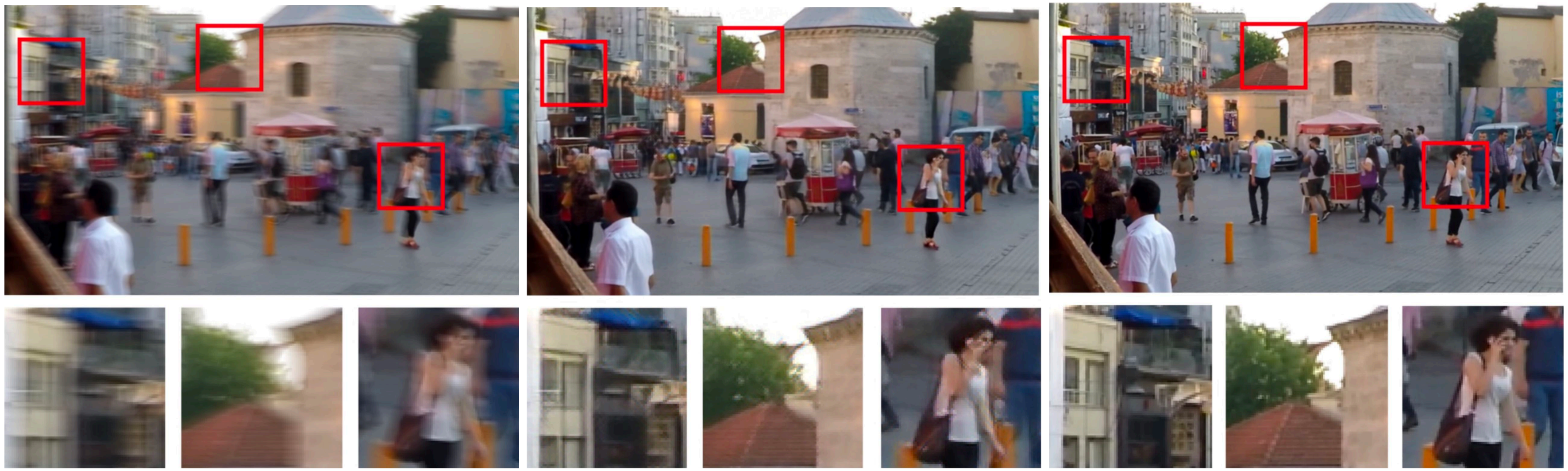
Image Deblurring

- non-uniform motion blur from a single blurry image.
- **Key idea:** Use a conditional GAN and content loss

$$\mathcal{L} = \underbrace{\underbrace{\mathcal{L}_{GAN}}_{adv\ loss} + \underbrace{\lambda \cdot \mathcal{L}_X}_{content\ loss}}_{total\ loss}$$

$$\mathcal{L}_{GAN} = \sum_{n=1}^{N} -D_{\theta_D}(G_{\theta_G}(I^B))$$

$$\mathcal{L}_X = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^S)_{x,y} - \phi_{i,j}(G_{\theta_G}(I^B))_{x,y})^2$$

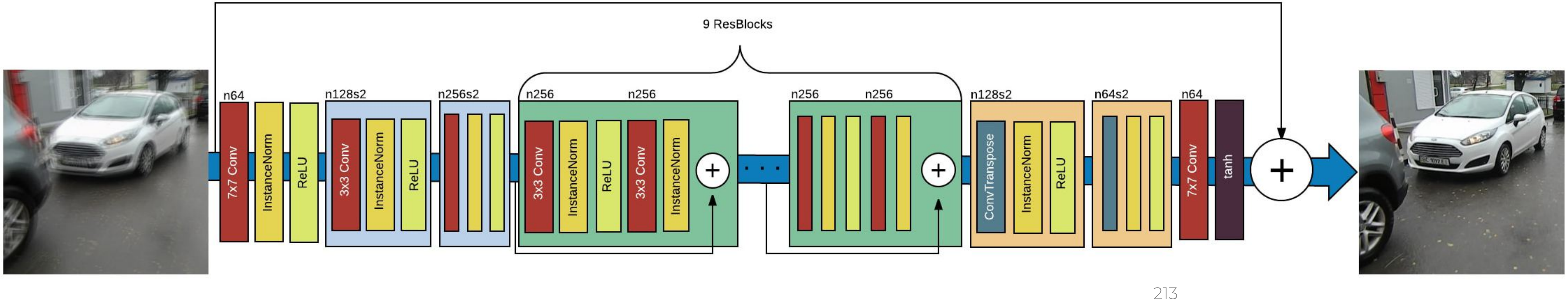


Blurred images | Groundtruth | Predicted

(Kupyn et al., CVPR 2018)

Image Deblurring

- Key idea: An image to image translation model that learns the residual to sharpen the blurred image



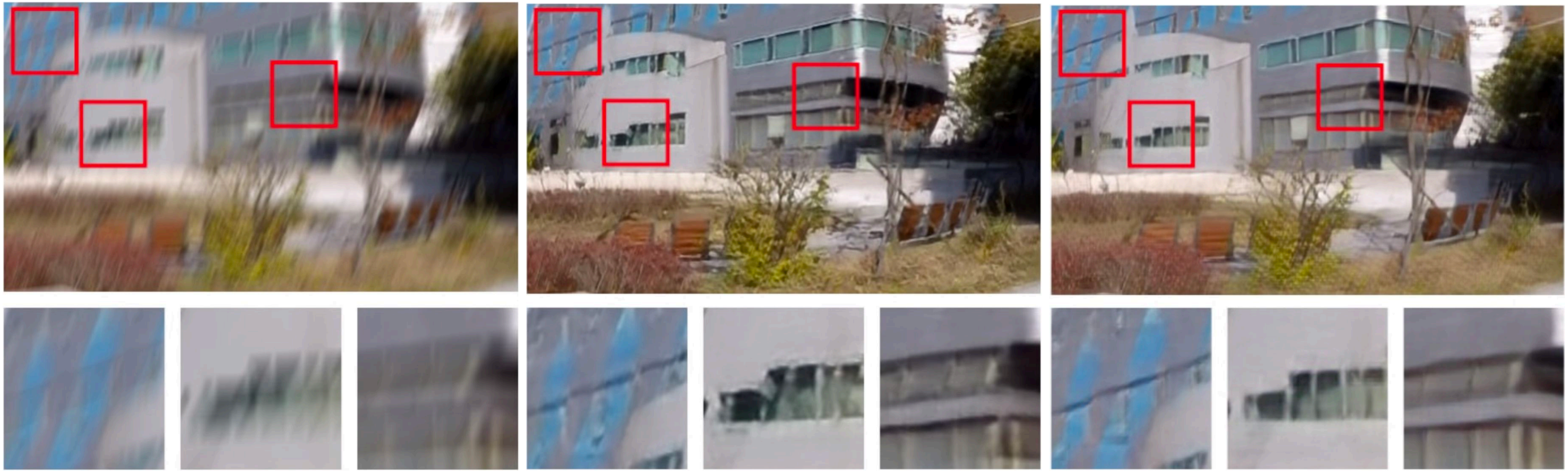
(Kupyn et al., CVPR 2018)

Image Deblurring

Blurred images Nah et al., CVPR 2017 Kupyn et al., CVPR 2018

(Kupyn et al., CVPR 2018)

Image Deblurring



Blurred images | Nah et al., CVPR 2017 | Kupyn et al., CVPR 2018

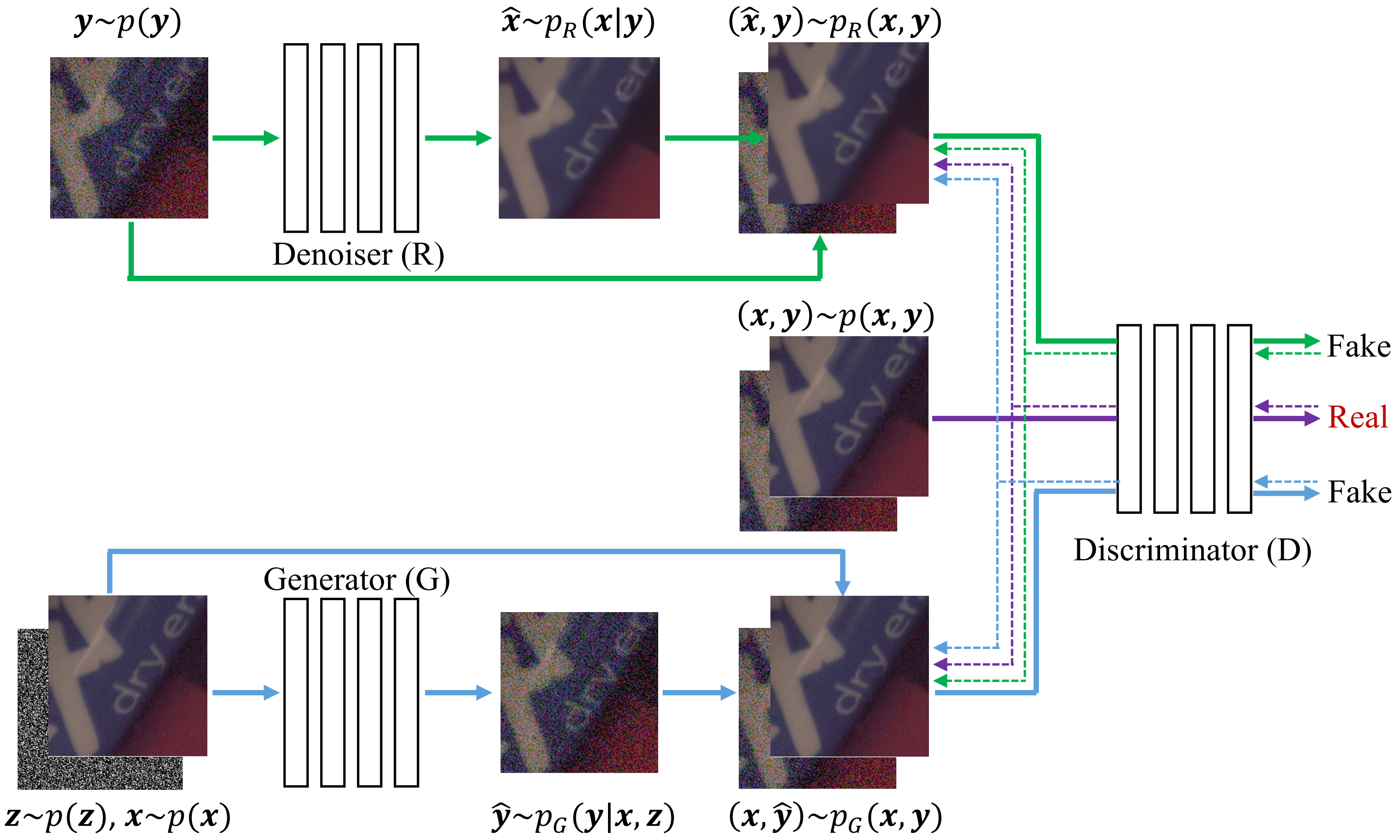
(Kupyn et al., CVPR 2018)

Image Deblurring

Blurred images | Nah et al., CVPR 2017 | Kupyn et al., CVPR 2018

(Kupyn et al., CVPR 2018)

Image Denoising

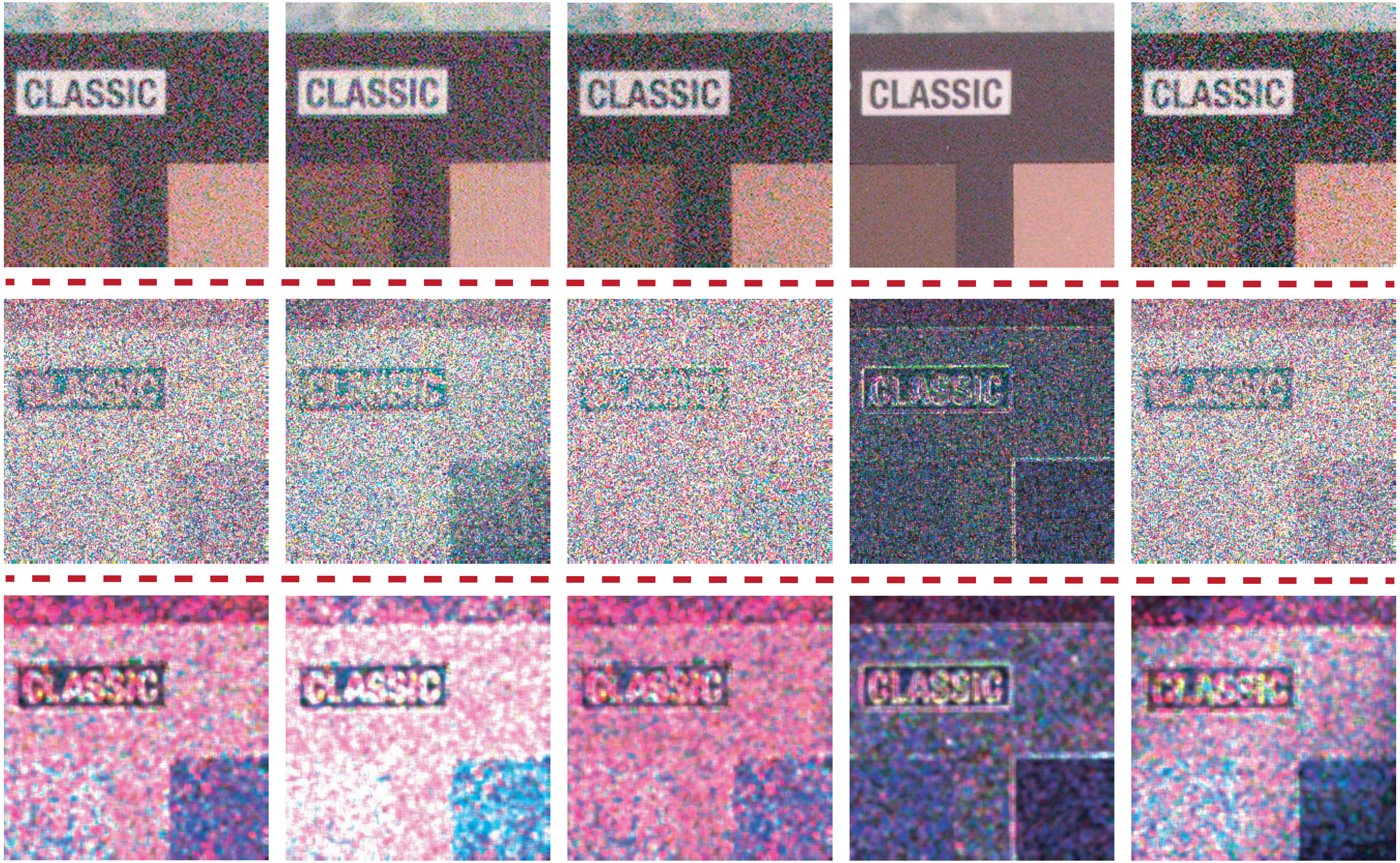


- Key idea: simultaneously deal with the noise removal and noise generation tasks.

(Yue et al., ECCV 2020)

Image Denoising

Generated Noisy Images

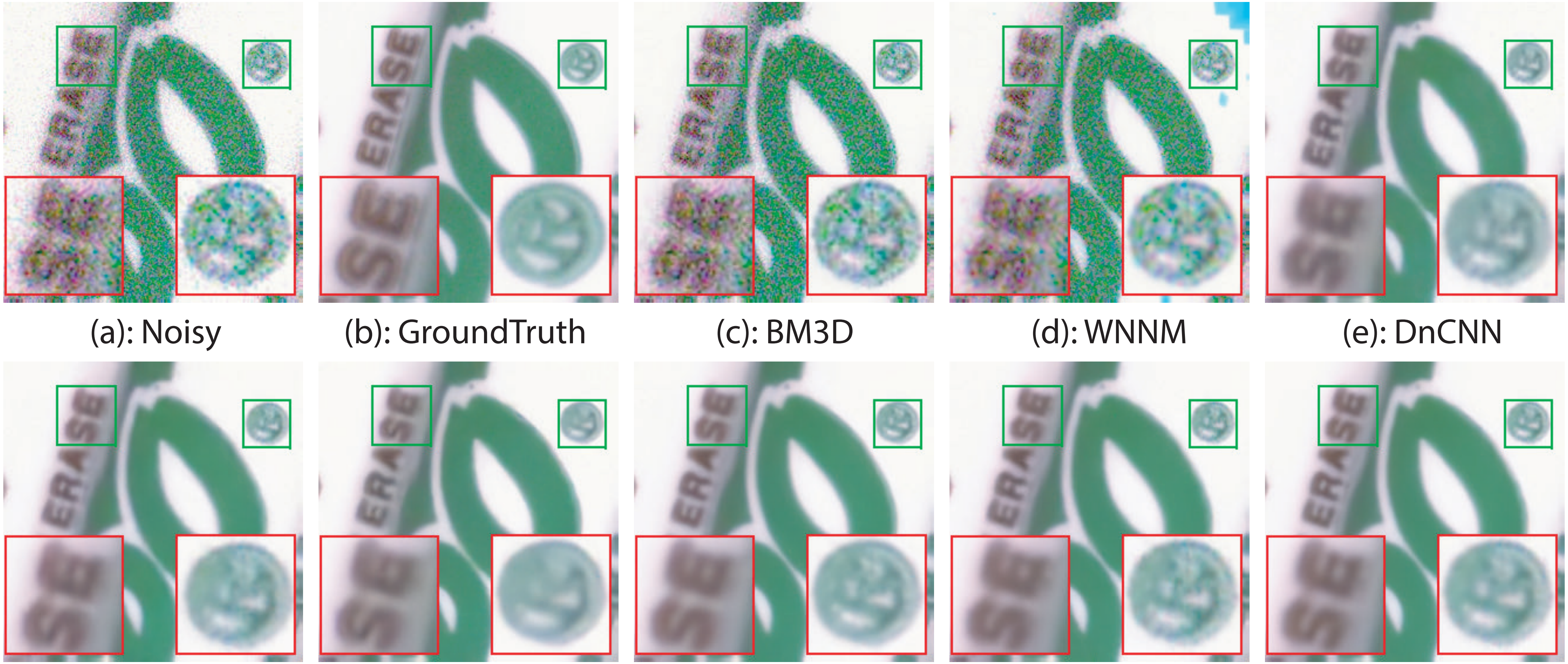


(a) Real (b) CBDNet (c) ULRD (d) GRDN (e) DANet

(Yue et al., ECCV 2020)

Image Denoising

Denoising results

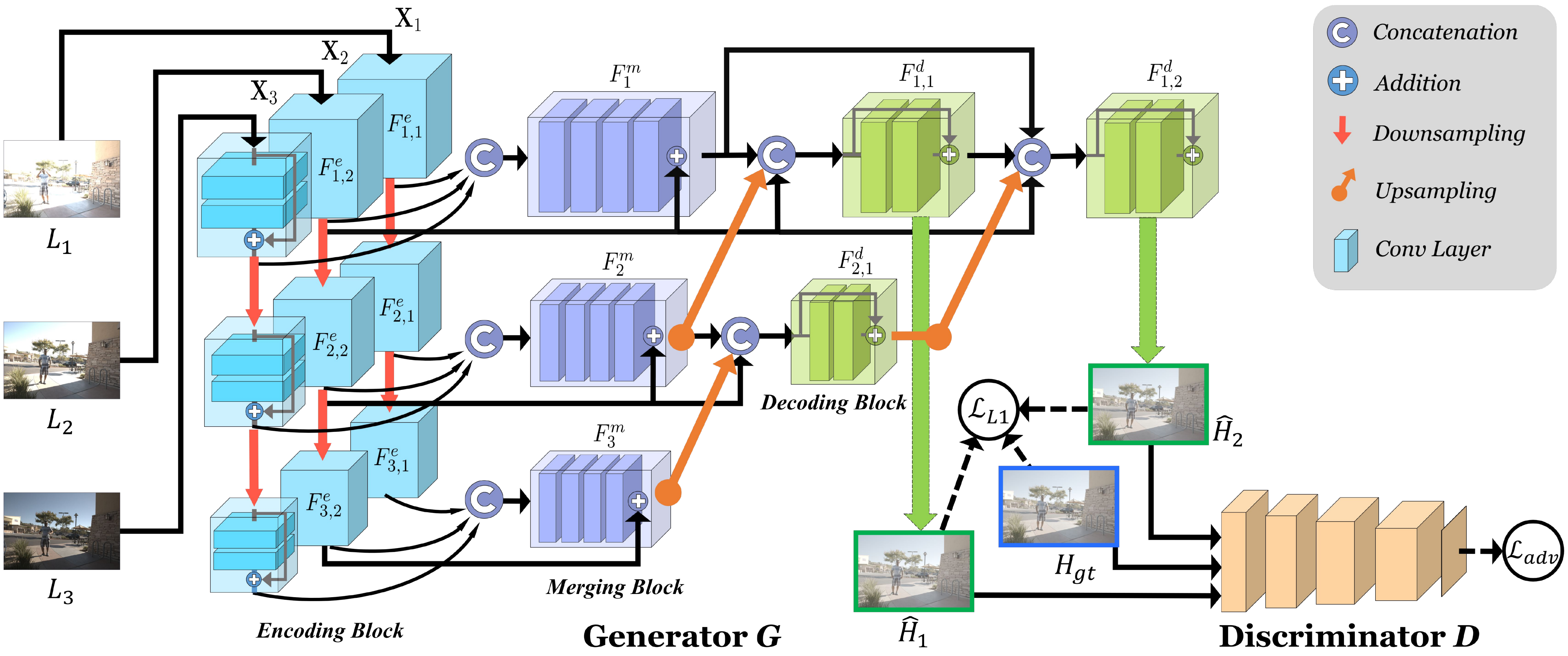


(a): Noisy (b): GroundTruth (c): BM3D (d): WNNM (e): DnCNN

(f): CBDNet (g): RIDNet (h): VDN (i): DANet (j): DANet+

(Yue et al., ECCV 2020)

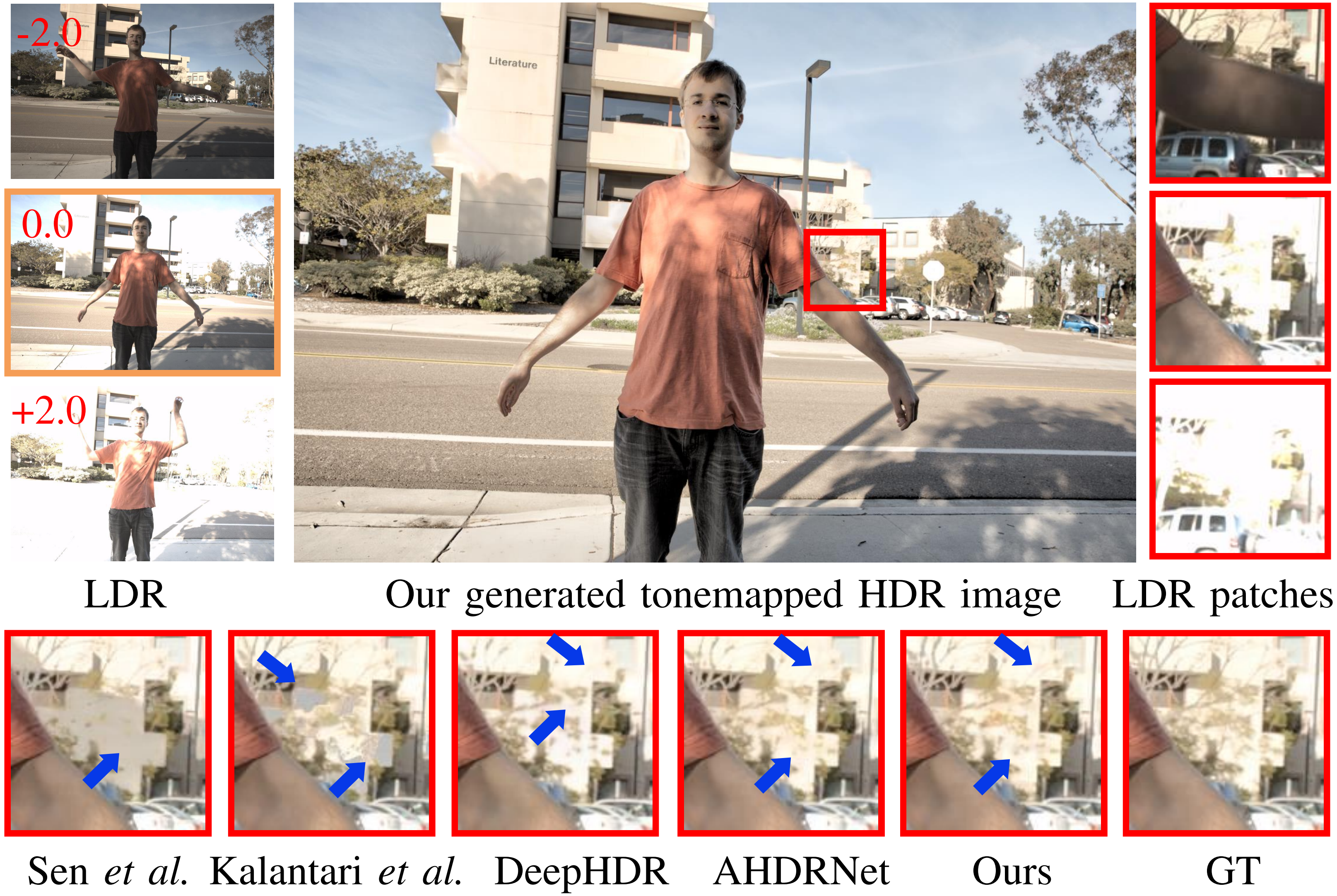
High Dynamic Range Imaging



- Key idea: Incorporate adversarial learning to be able to produce faithful information in the regions with missing content.

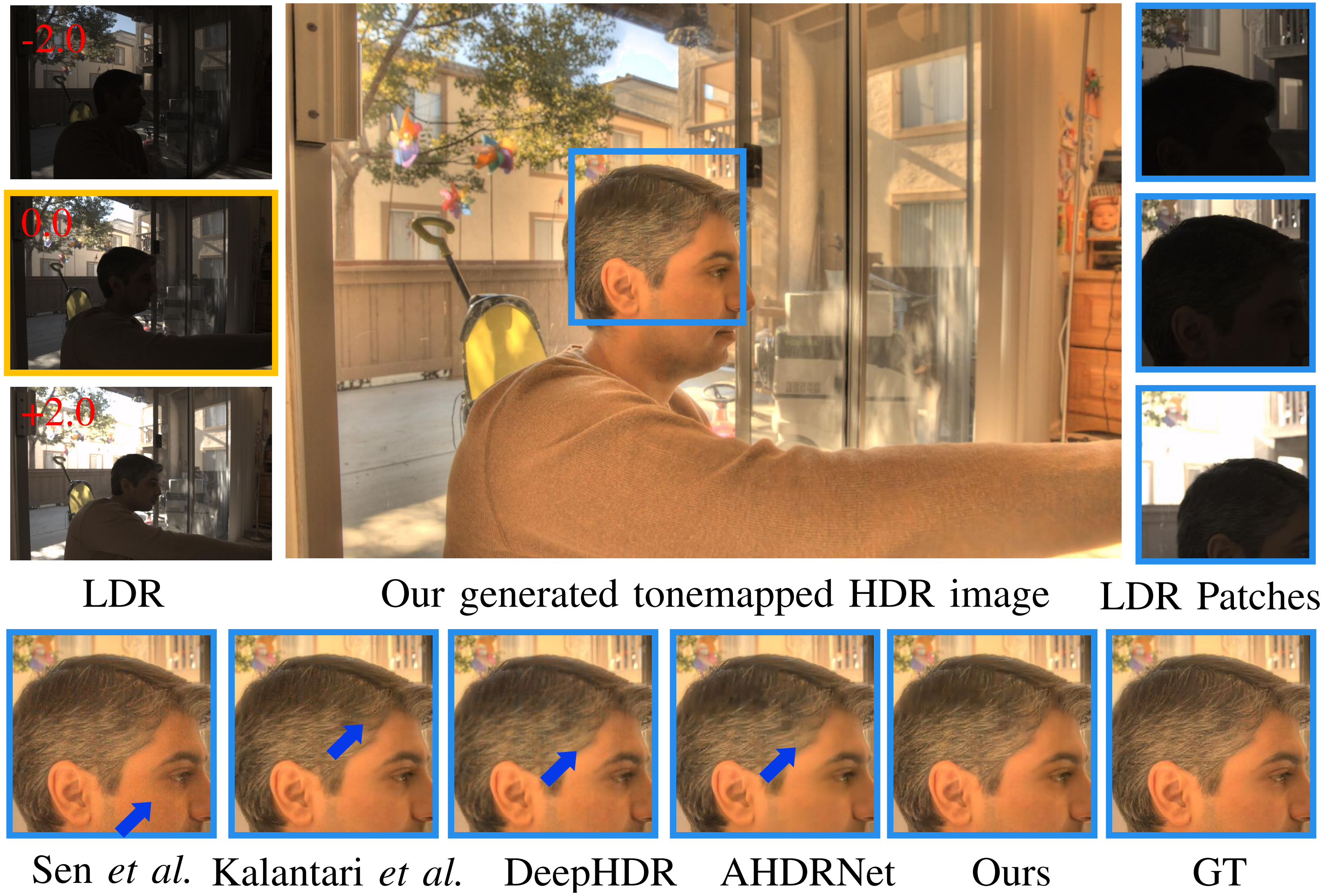
(Niu et al., IEEE TIP 2021)

High Dynamic Range Imaging



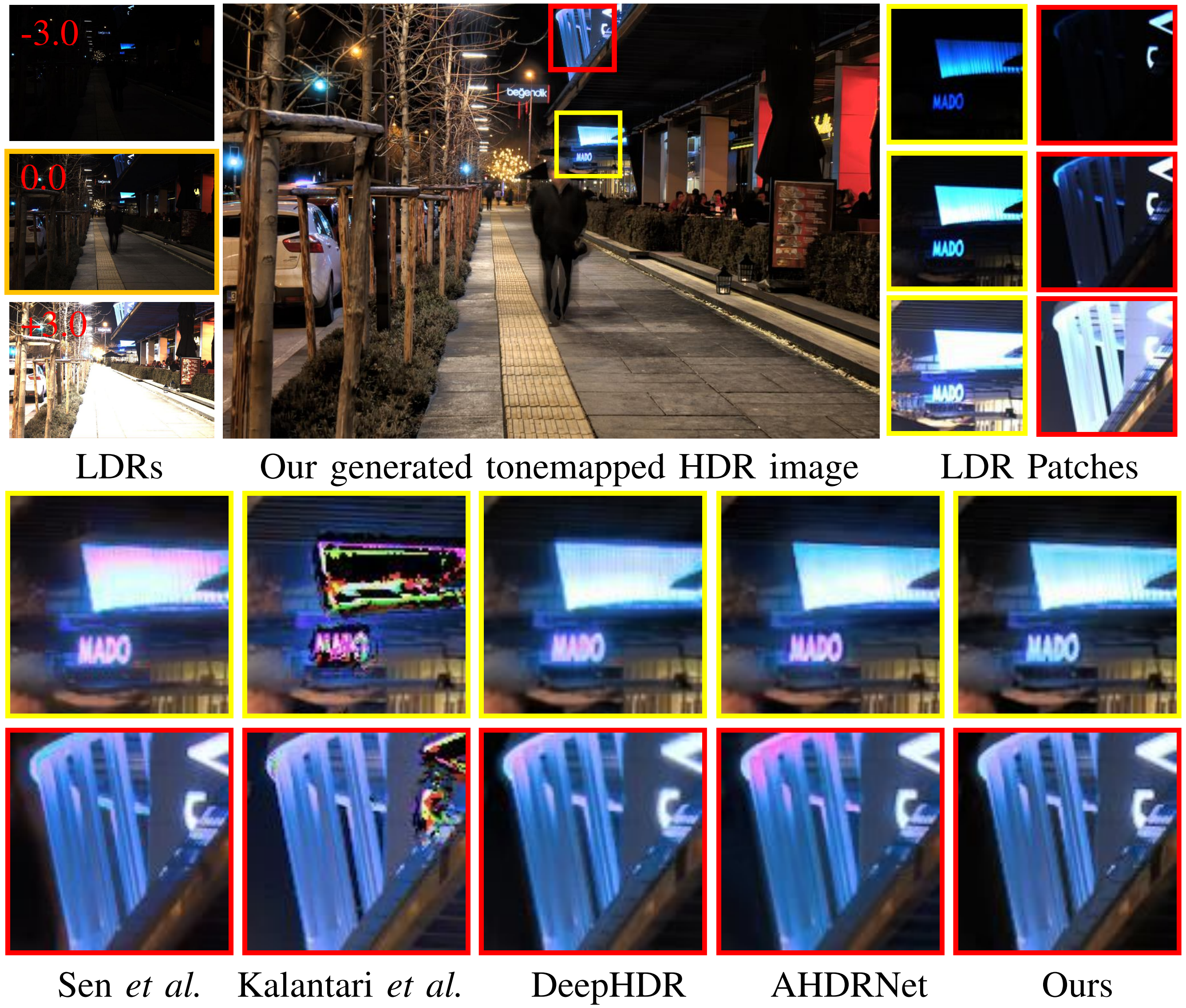
(Niu et al., IEEE TIP 2021)

High Dynamic Range Imaging



(Niu et al., IEEE TIP 2021)

High Dynamic Range Imaging

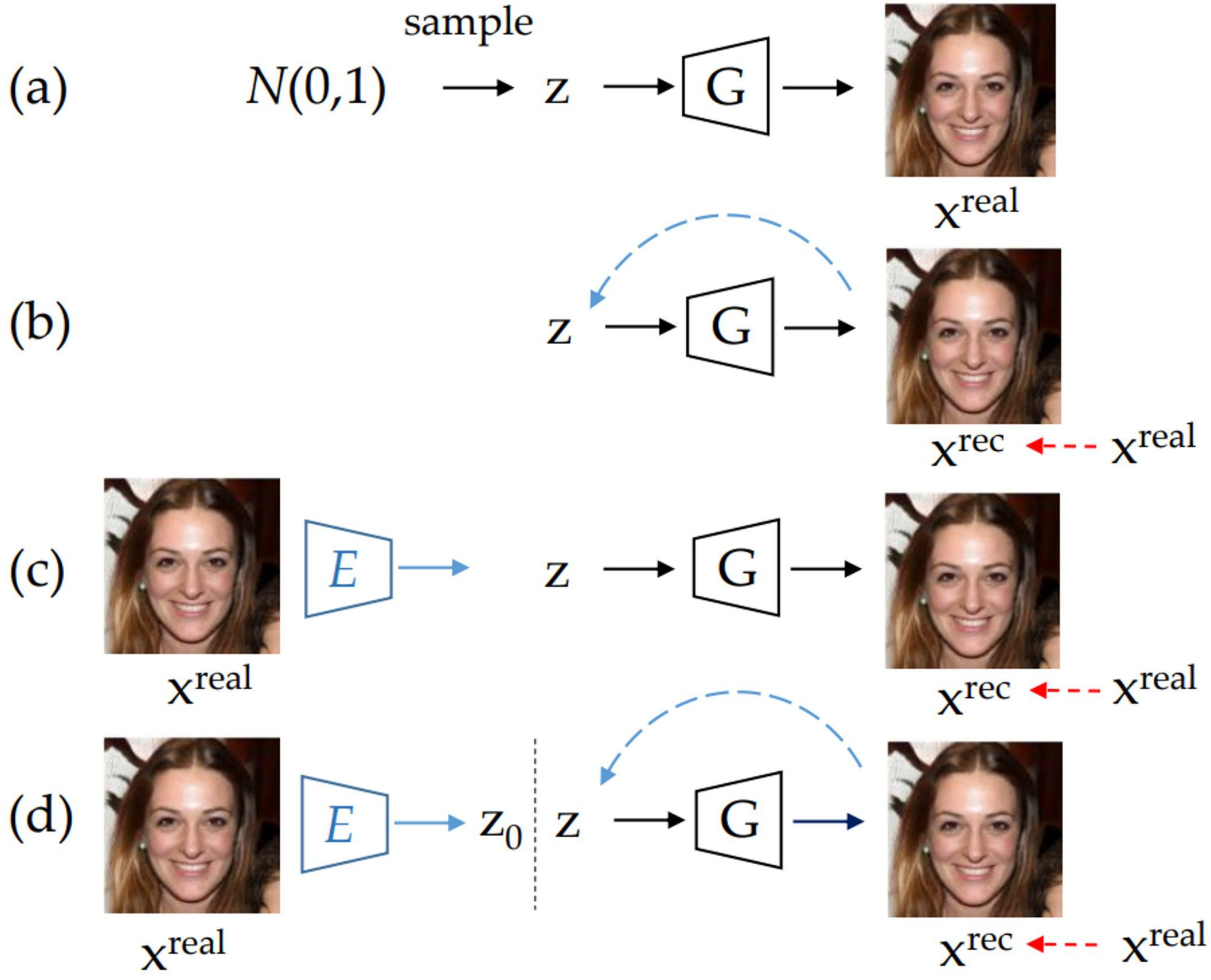


(Niu et al., IEEE TIP 2021)

GAN Inversion

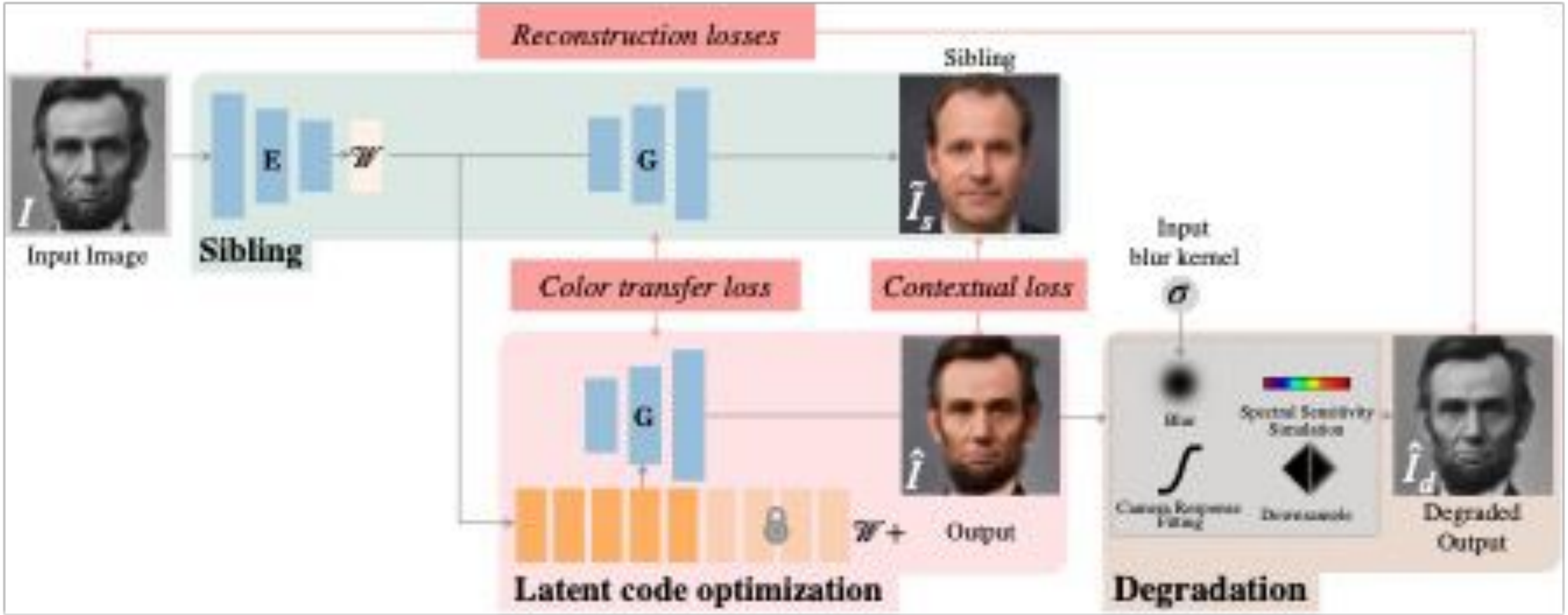
3 methods of inversion:

- Optimization-based (b)
- Learning-based (c)
- Hybrid (d)



(Xia et al., arXiv 2021)

Time-travel Rephotography



- Key idea: Use the StyleGAN2 framework to project old photos into the space of modern high-resolution photos for enhancing their quality.

(Luo et al., SIGGRAPH Asia 2021)

Time-travel Rephotography

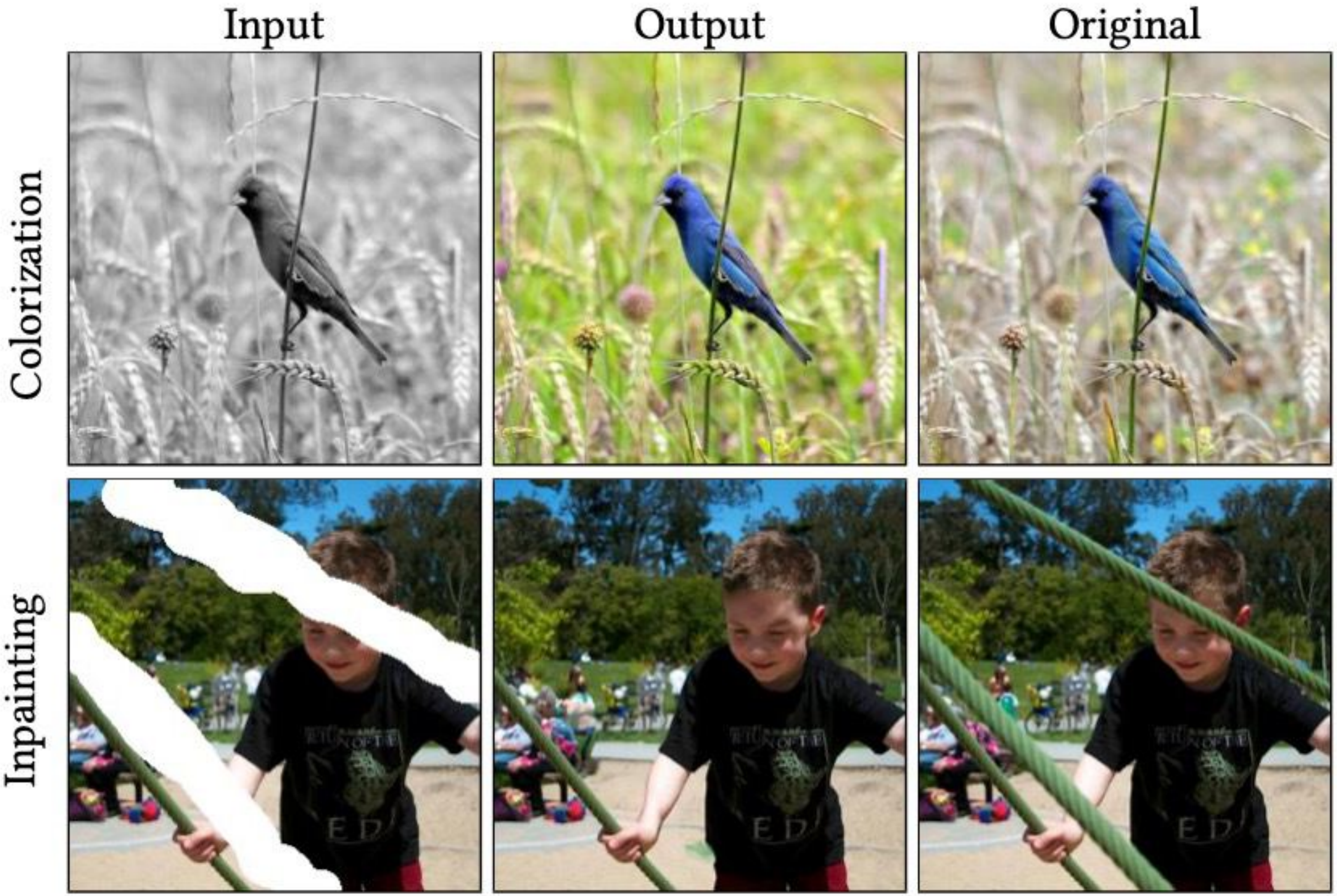
(Luo et al., SIGGRAPH Asia 2021)

Time-travel Rephotography

Input Ours DeOldify InstColorization Zhang Zhang (FFHQ)

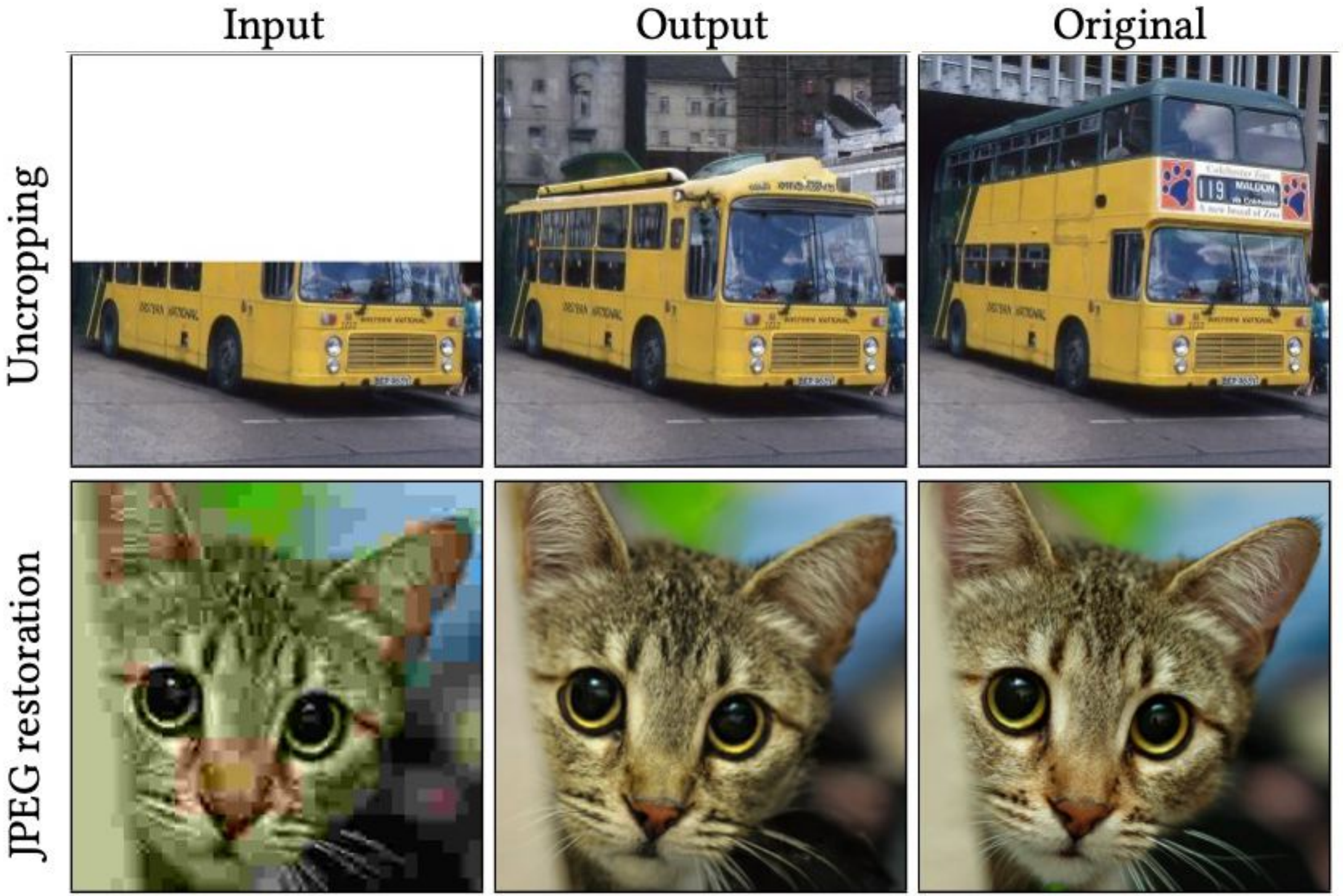
(Luo et al., SIGGRAPH Asia 2021)

Palette: Image-to-Image Diffusion Models



[Saharia et al. 2022]

Palette: Image-to-Image Diffusion Models



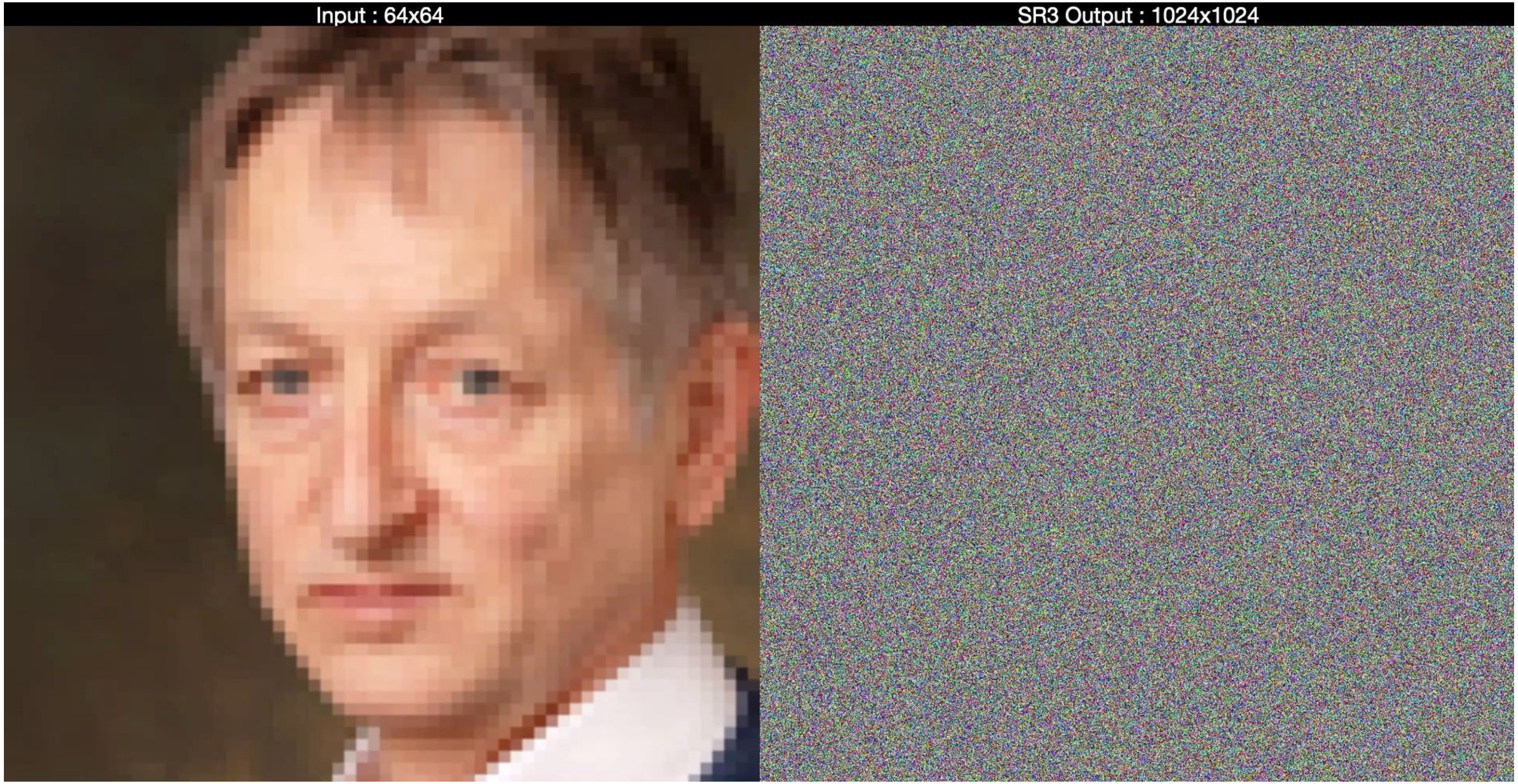
[Saharia et al. 2022]

Palette: Image-to-Image Diffusion Models

Panorama Generation

[Saharia et al. 2022]

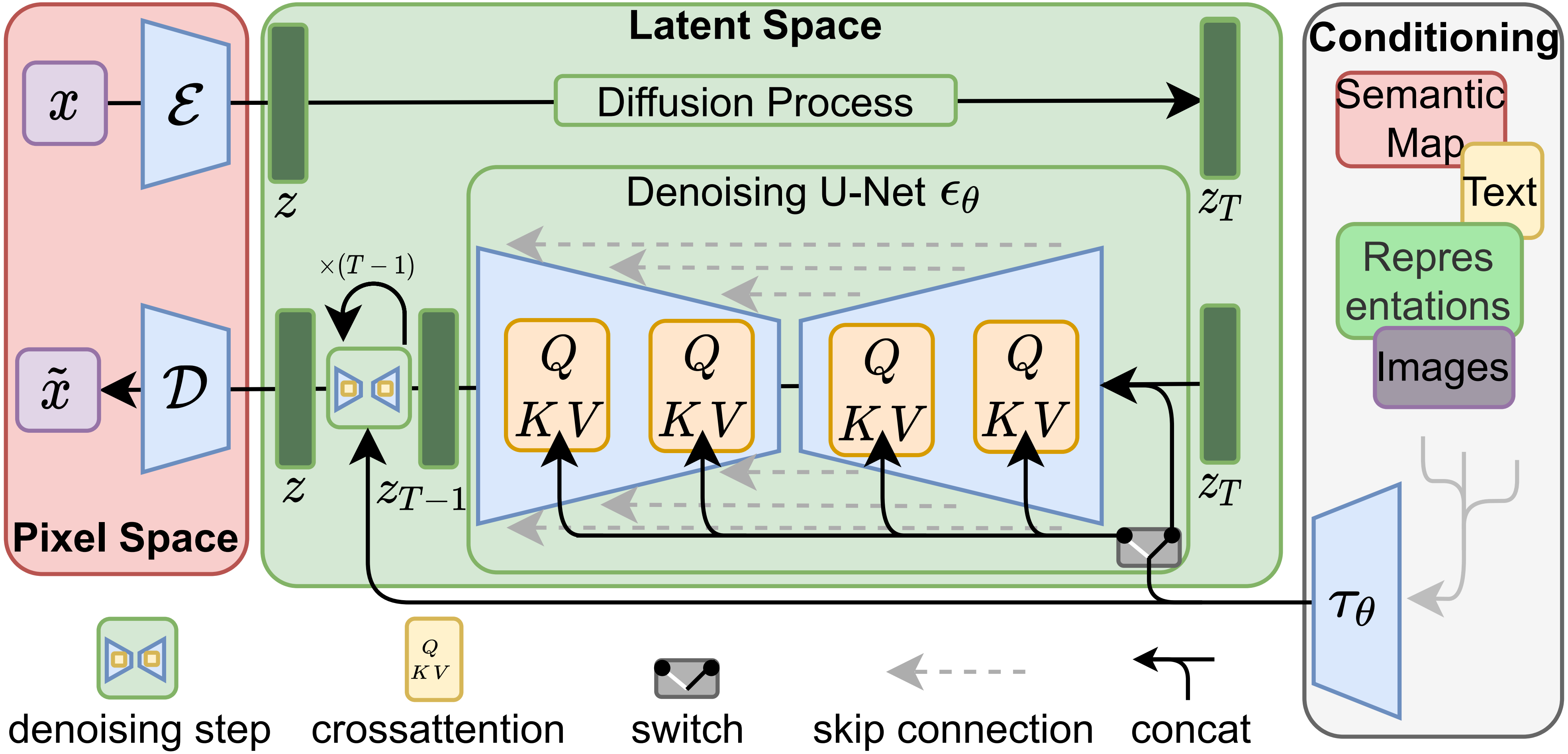
Super Resolution



Results of a SR3 model (64×64 → 512×512), trained on FFHQ, and applied to images outside of the training set.

Latent Diffusion

Autoencoder with KL or VQ regularization



[Rombach et al., "High-Resolution Image Synthesis with Latent Diffusion Models", CVPR 2022]

Image Inpainting

[“LDM”, Rombach et al., 2022]

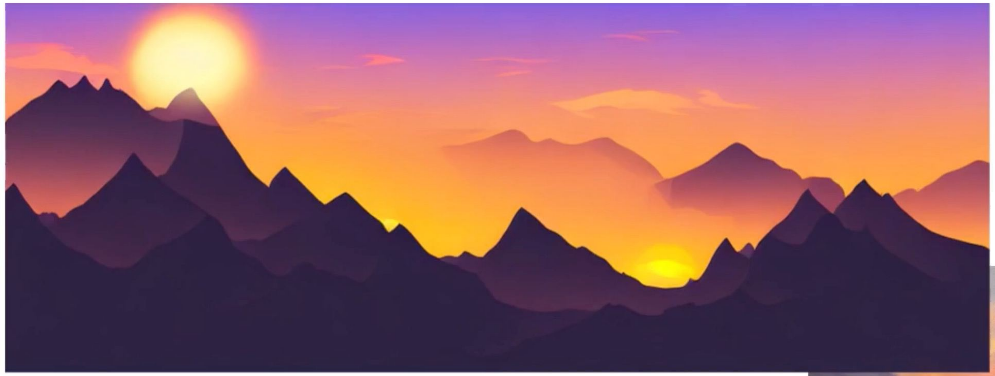
Semantic Image Synthesis

["LDM", Rombach et al., 2022]

Generating Images from Text

"a portrait of a cyberpunk rabbit, trending on artstation"

"A sunset over a mountain, vector image"

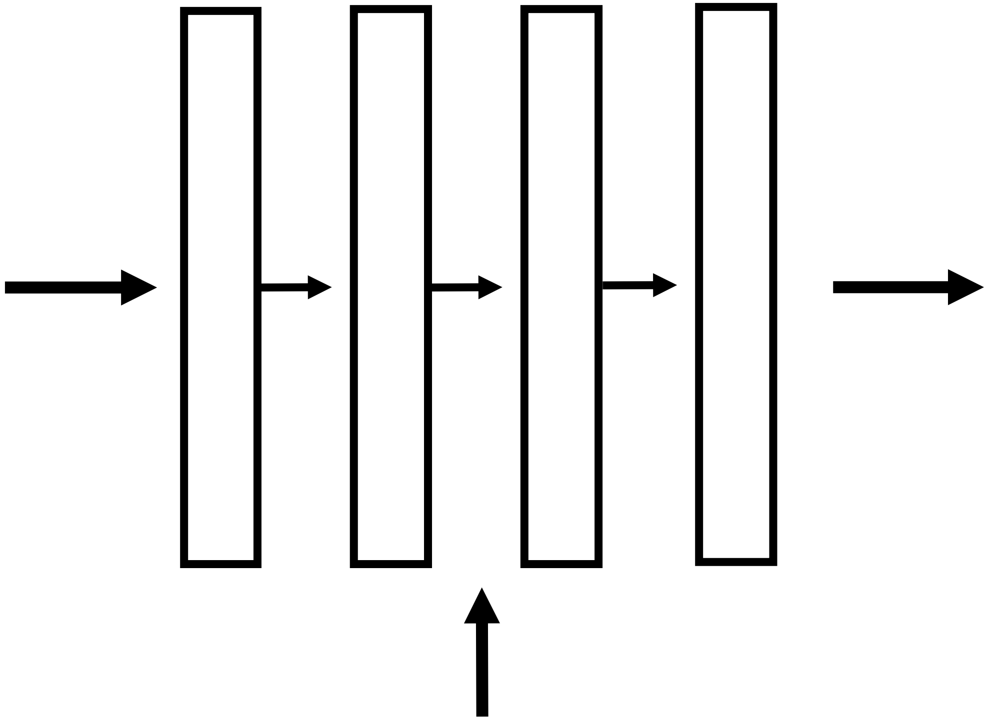


"A sunset over a mountain, oil on canvas"

["LDM", Rombach et al., 2022]

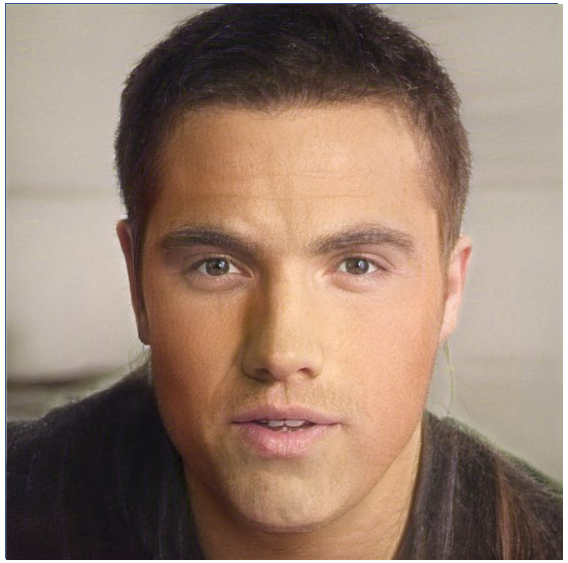
Semantic Image Editing

image $\mathbf{x}$



"he is tanned"

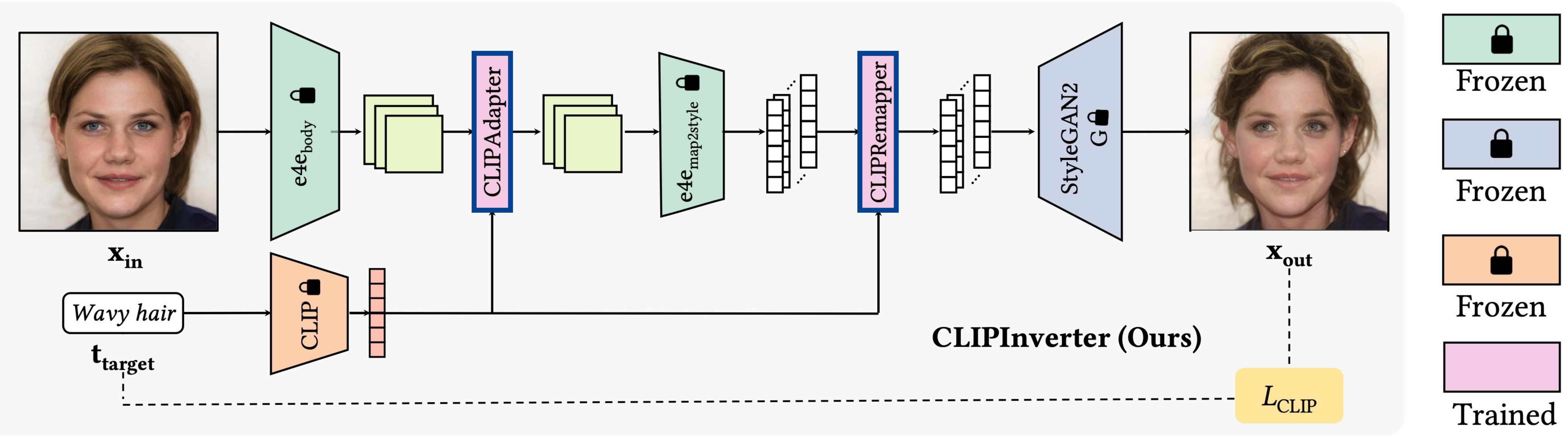
Target description $\mathbf{t}$



manipulated
image $\mathbf{y}$

- **Input:** input image
  + target description
  Output: manipulated image

CLIPInverter

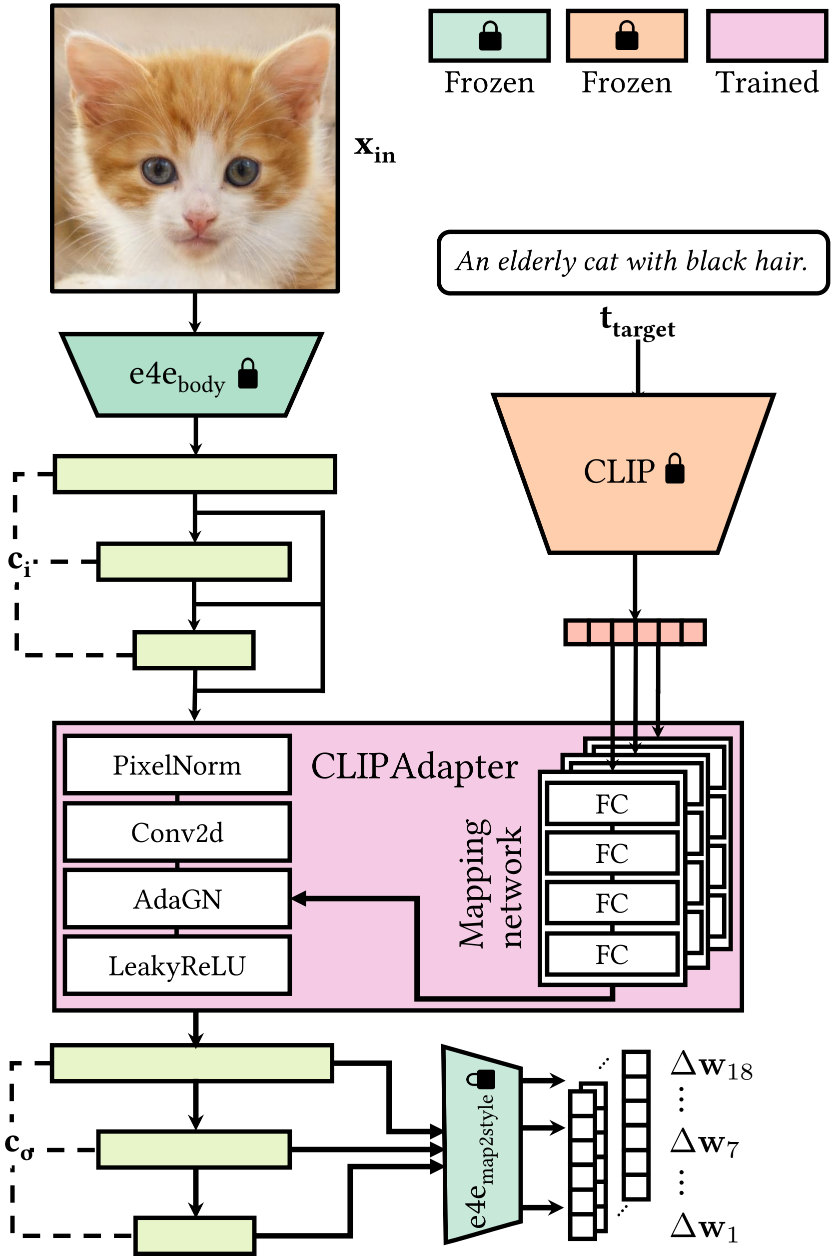


- **CLIPAdapter:** Finds and follows semantic paths in latent space for initial image edits.
- **CLIPRemapper:** Refines edits by aligning the latent code with text prompt embeddings.

CLIPAdapter

- Extract feature maps from the image.
- Modulate these features with CLIP text embeddings.
- Input modulated features into the encoder to produce residual latents.

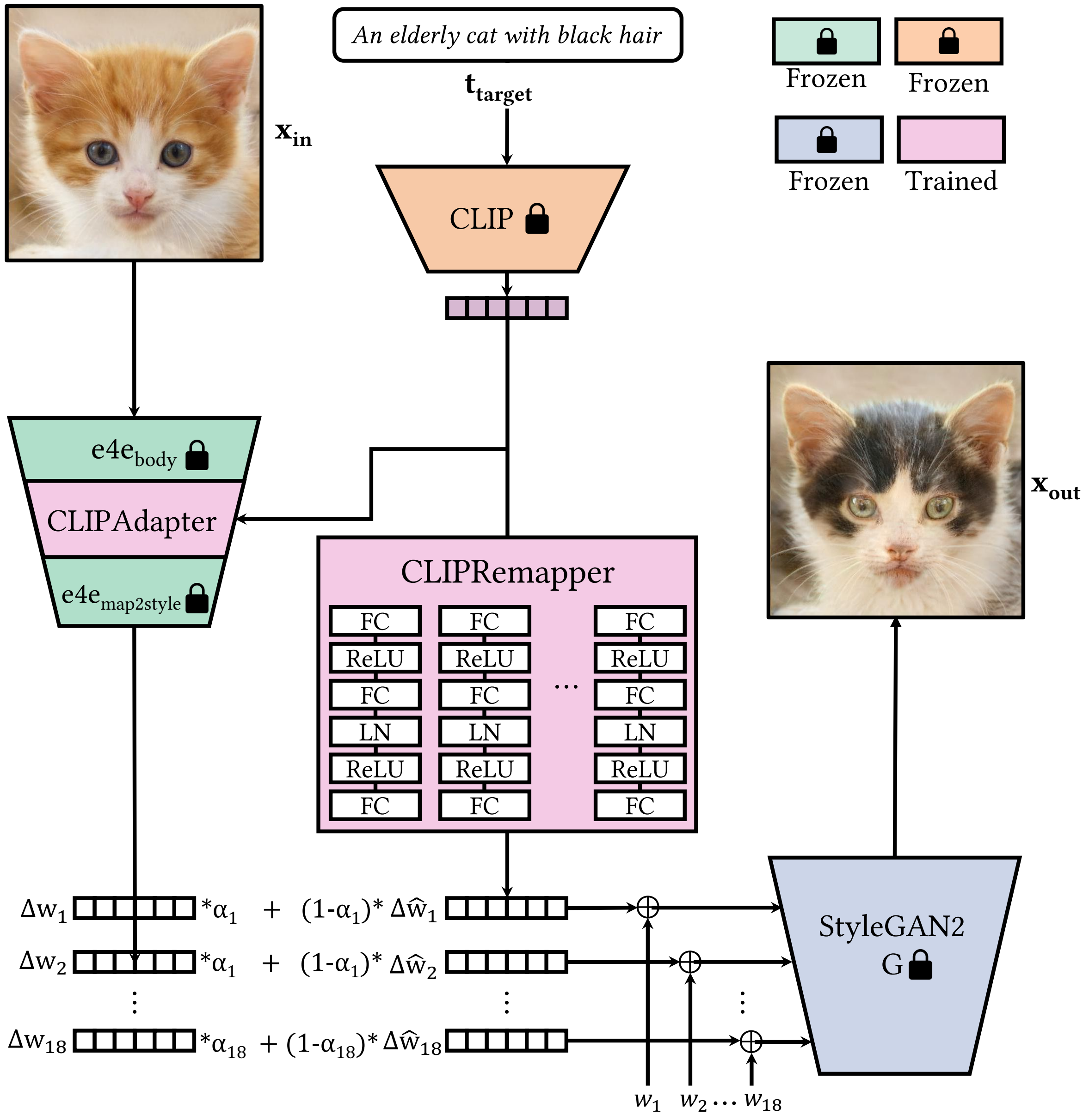
We will show that this improves editability!

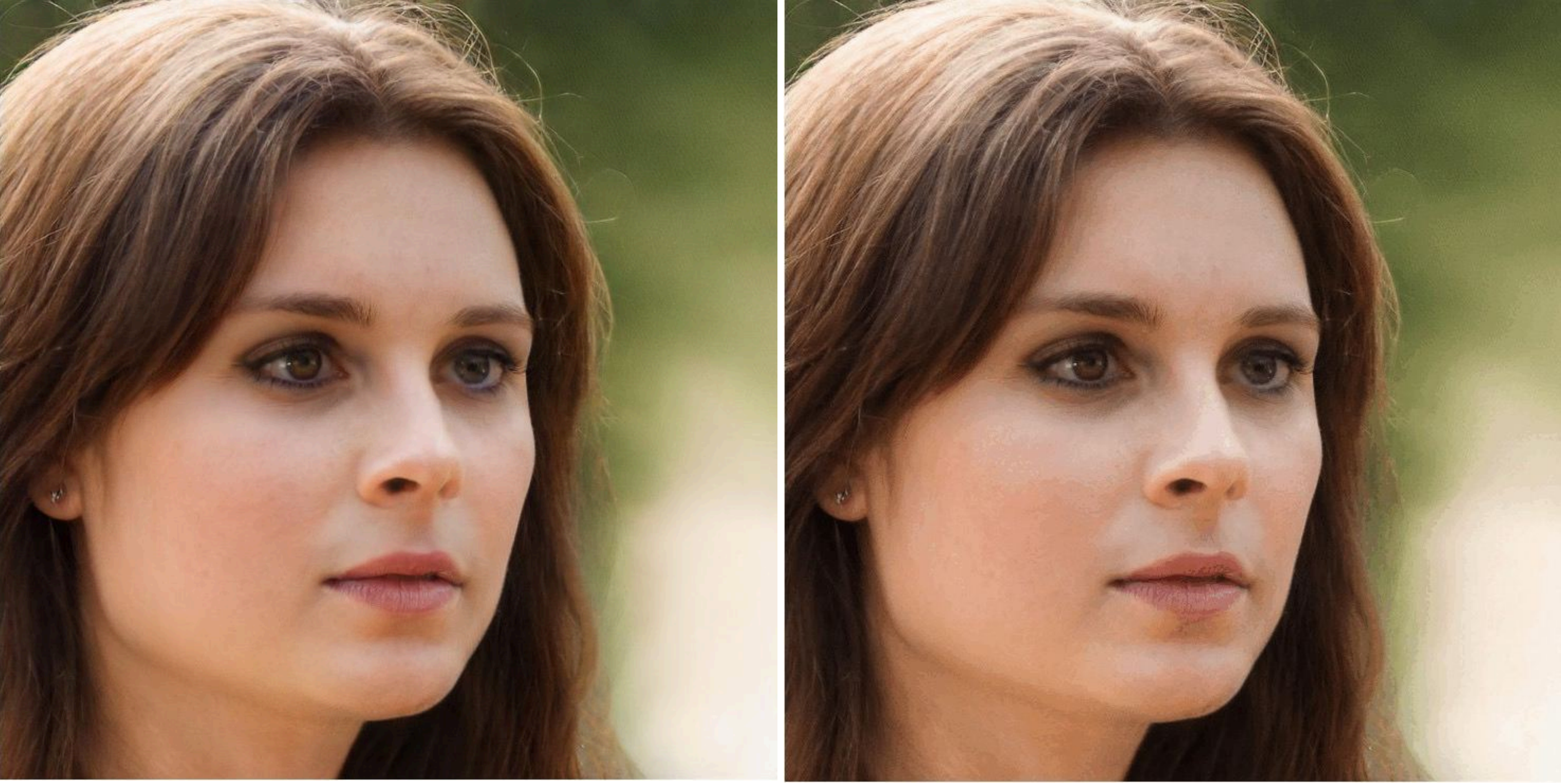


CLIPRemapper

- CLIPAdapter provides residual latents.
- Remapper calculates corrections using text prompts.
- Combines residuals with corrections to input into the generator for final image output.

This further improves editing accuracy!



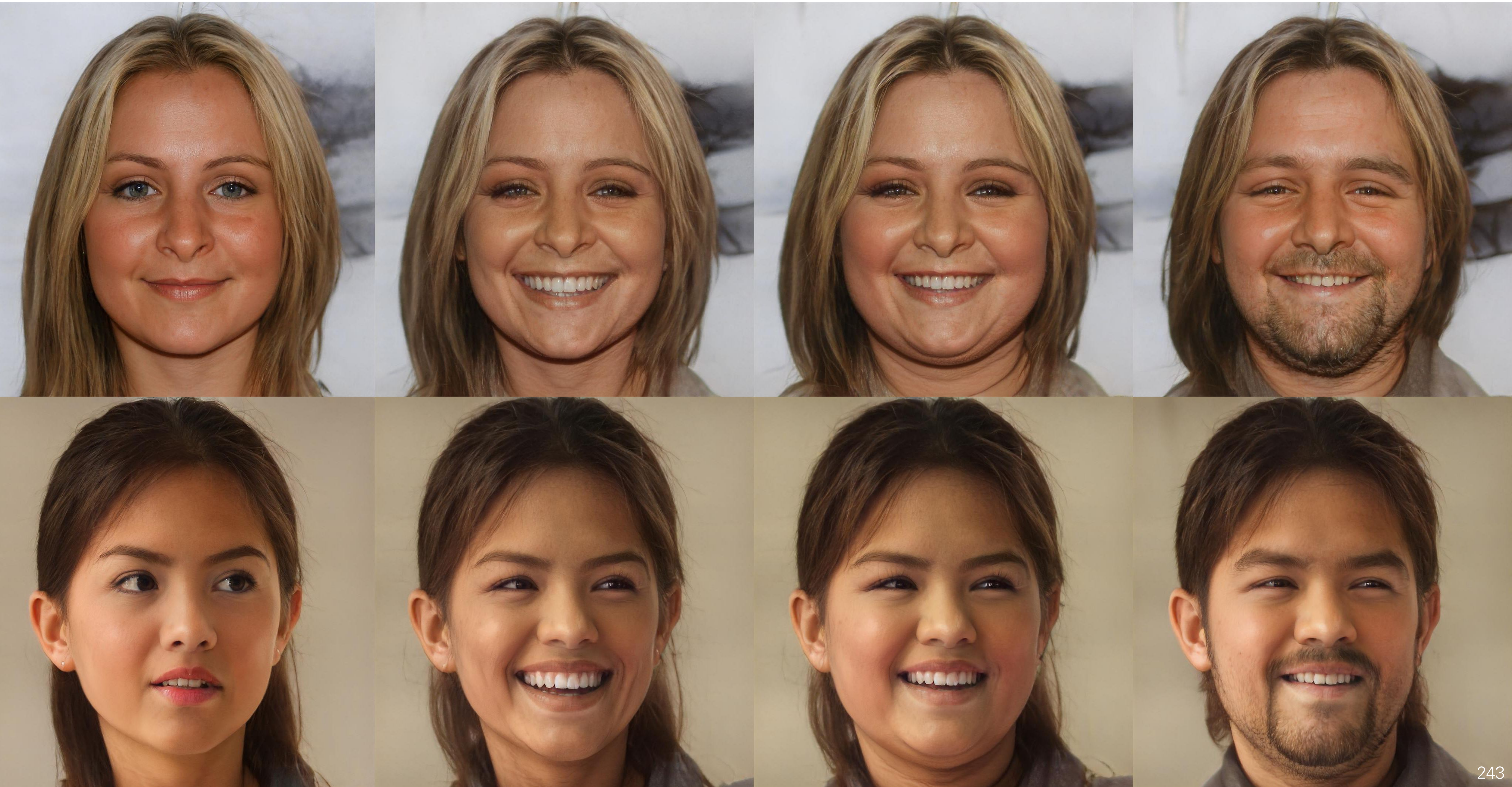


Target: This person has mustache

Target: A cat with ginger hair

Target: This bird has wings that are blue and has a white belly

Original	Smiling	Smiling+Chubby	Smiling+Chubby+Beard



Original | TediGAN-B | CLIPInverter (Ours)

Target: He wears necktie. He has bushy eyebrows, mouth slightly open, bags under eyes, big nose, and high cheekbones. He is smiling.

Xia et al. **TediGAN: Text-Guided Diverse Face Image Generation and Manipulation.** CVPR 2021.

Original | StyleCLIP-LO | CLIPInverter (Ours)

Target: He wears necktie. He has bushy eyebrows, mouth slightly open, bags under eyes, big nose, and high cheekbones. He is smiling.

Patashnik et al. **StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery**. ICCV 2021.

Original | StyleCLIP-LO | CLIPInverter (Ours)

Target: `He wears necktie. He has bushy eyebrows, mouth slightly open, bags under eyes, big nose, and high cheekbones. He is smiling.`

Patashnik et al. **StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery**. ICCV 2021.

Original | StyleMC | CLIPInverter (Ours)

Target: He wears necktie. He has bushy eyebrows, mouth slightly open, bags under eyes, big nose, and high cheekbones. He is smiling.

Kocasari et al. **StyleMC: Multi-Channel Based Fast Text-Guided Image Generation and Manipulation.** WACV 2022.

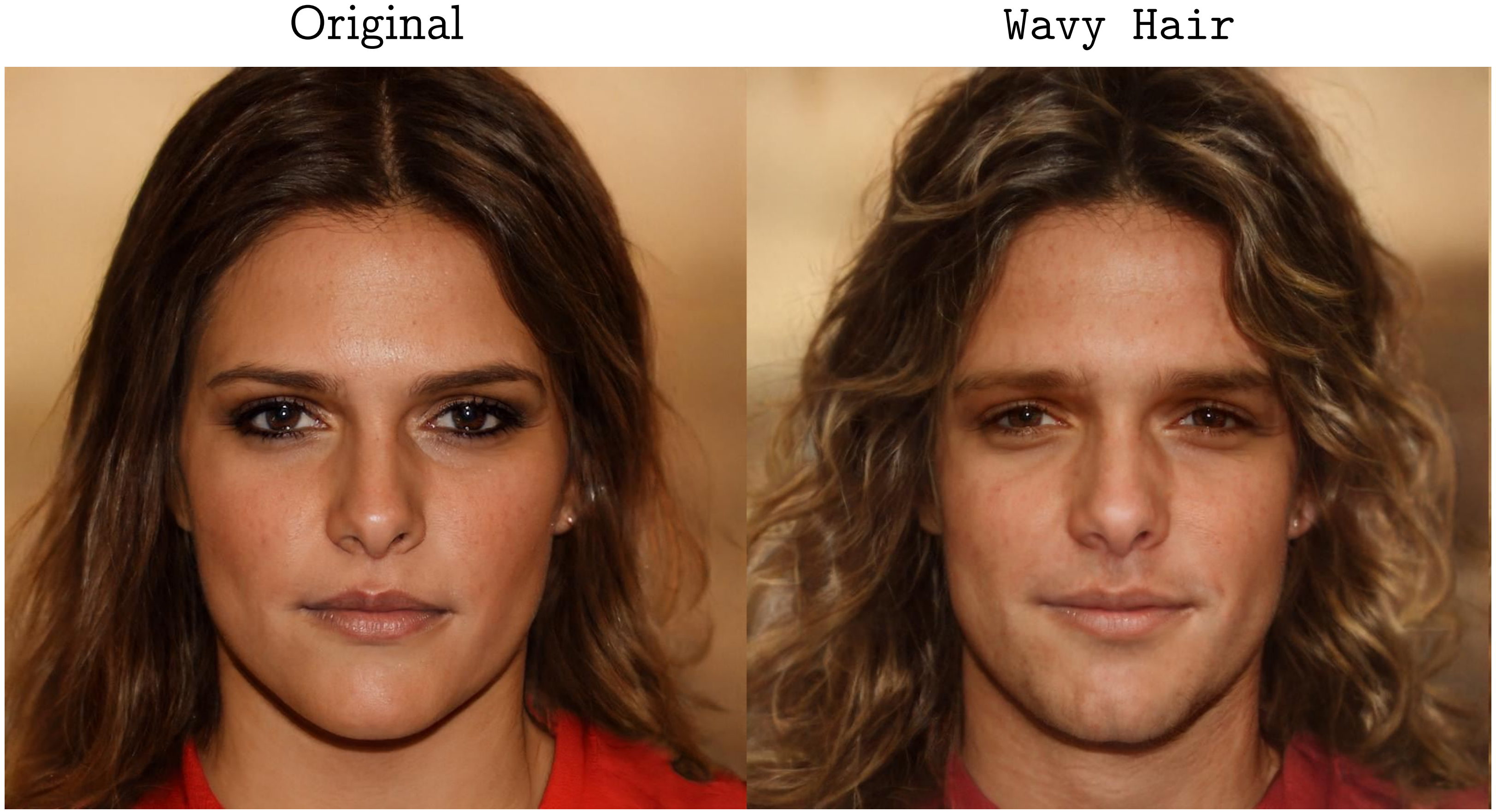
Original | HairCLIP | CLIPInverter (Ours)

Target: He wears necktie. He has bushy eyebrows, mouth slightly open, bags under eyes, big nose, and high cheekbones. He is smiling.

Wei et al. HairCLIP: Design your hair by text and reference image. CVPR 2022.

Limitations

- Directional CLIP loss may unintentionally alter gender representation.

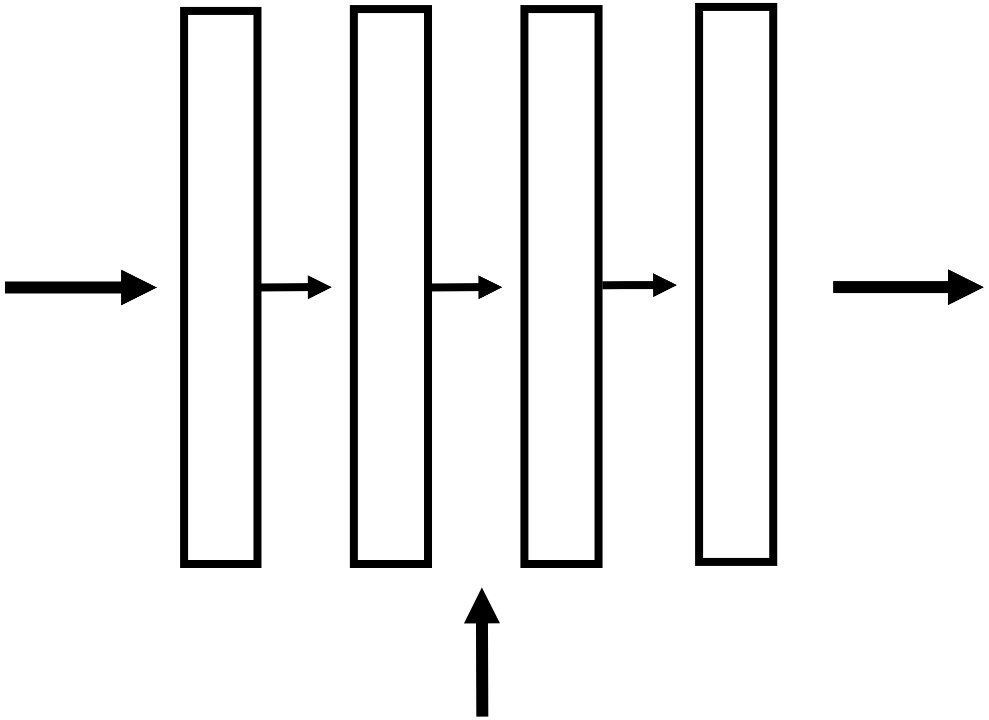


Limitations

- Directional CLIP loss may unintentionally alter gender representation.
- Adding a specific pronoun to the target text description helps prevent this.

Instruction-Based Object Removal

image $\mathbf{x}$



“remove the lamp at the left of the large headboard”

Instruction $\mathbf{t}$

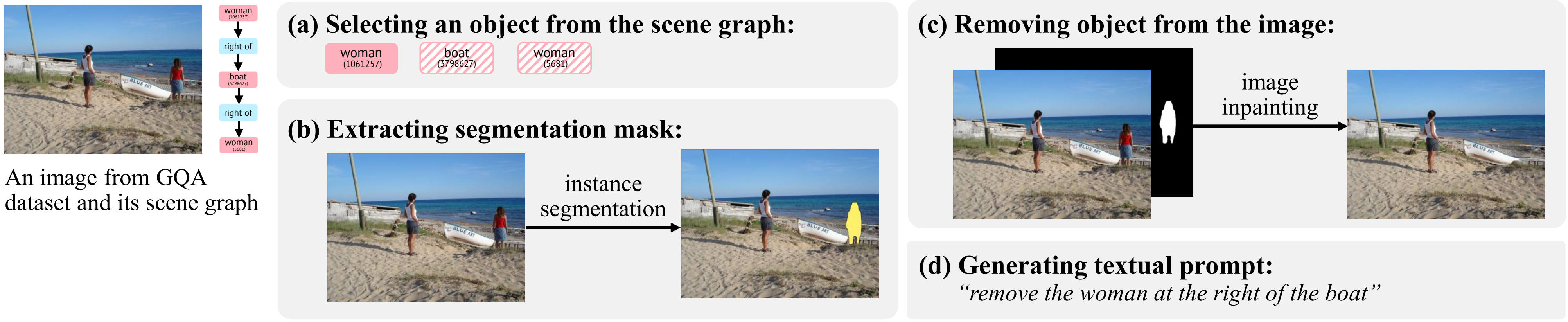
inpainted image $\mathbf{y}$

- **Input:** input image + instruction
  Output: inpainted image

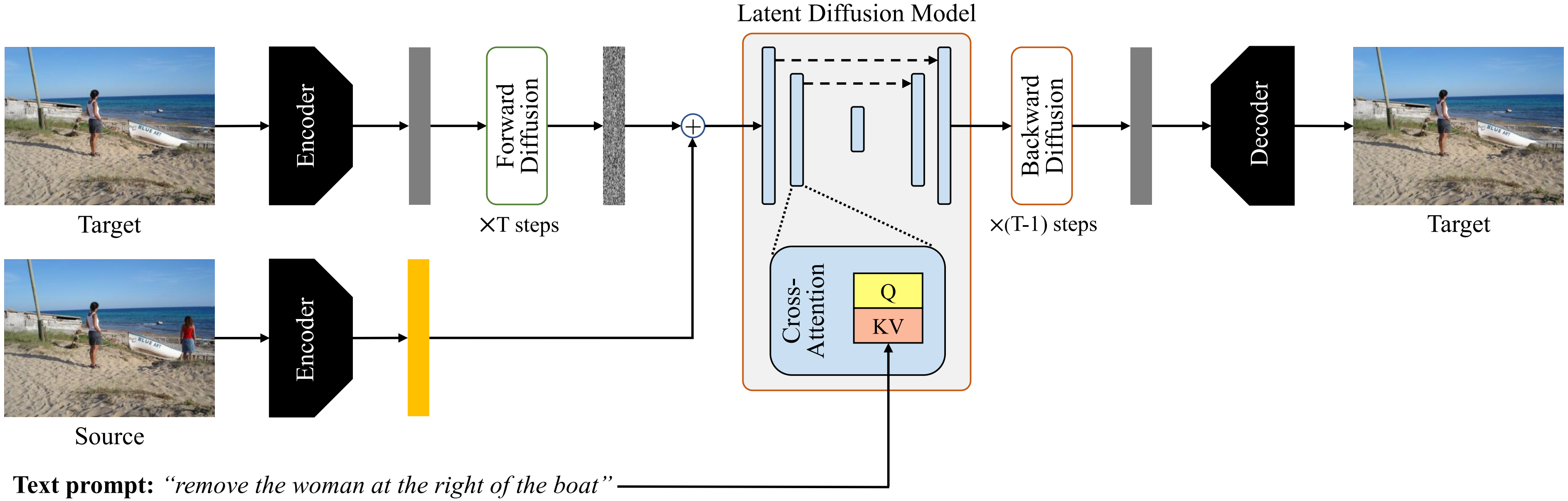
Yildirim et al., “Inst-Inpaint: Instructing to Remove Objects with Diffusion Models”, under review

Instruction-Based Object Removal

Training Data Generation for GQA-Inpaint



Training Inst-Inpaint for Instructional Image Inpainting



Instruction-Based Object Removal

remove the gray kite at the left

remove the street light at the left

remove the man at the right of the man

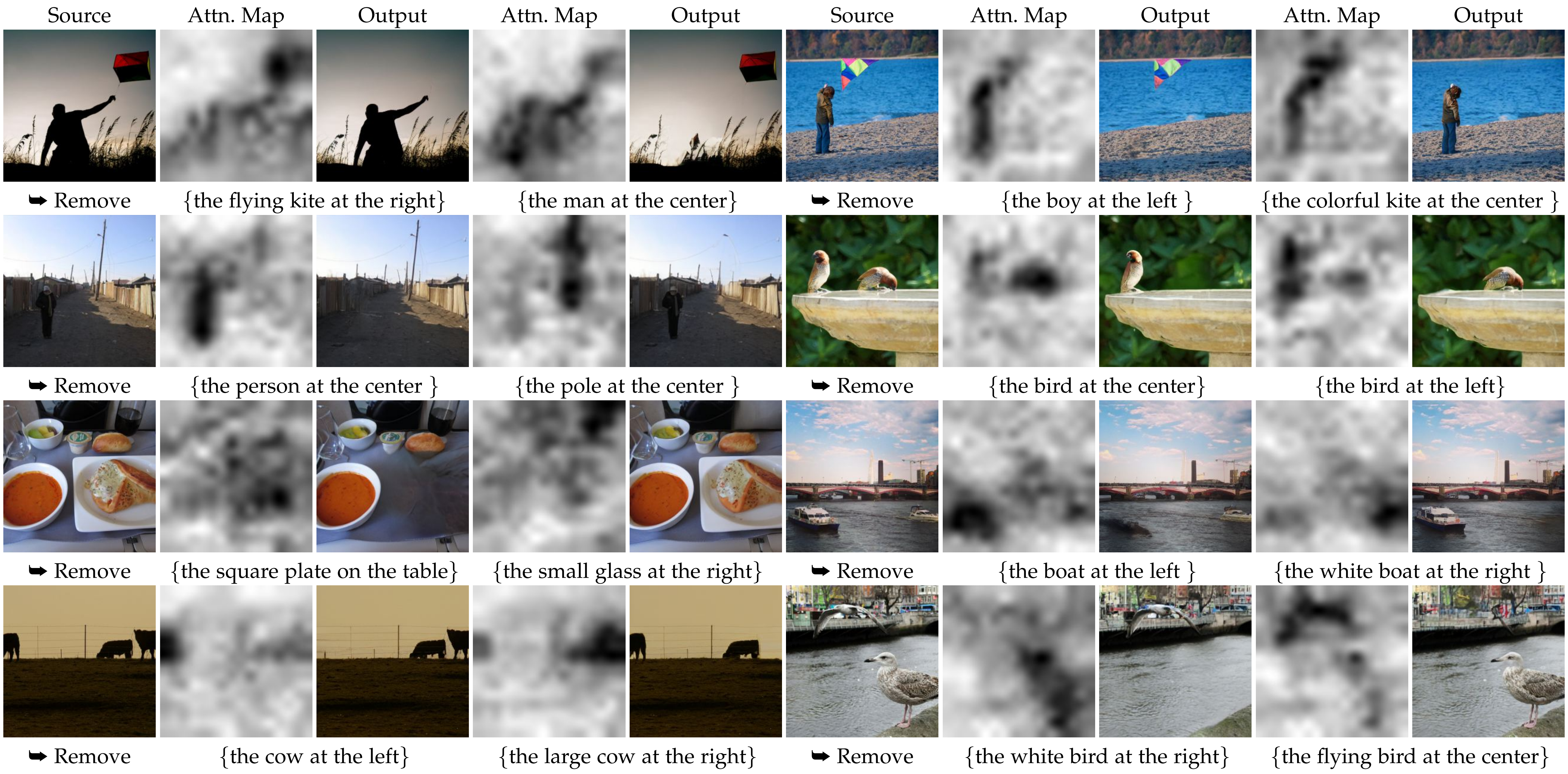
remove the red car at the left of the tall ladder

remove the colorful train at the right

remove the boat at the right of the small boat

Yildirim et al., "Inst-Inpaint: Instructing to Remove Objects with Diffusion Models", under review

Instruction-Based Object Removal



Yildirim et al., "Inst-Inpaint: Instructing to Remove Objects with Diffusion Models", under review

Next Lecture:

Visual Quality Assessment