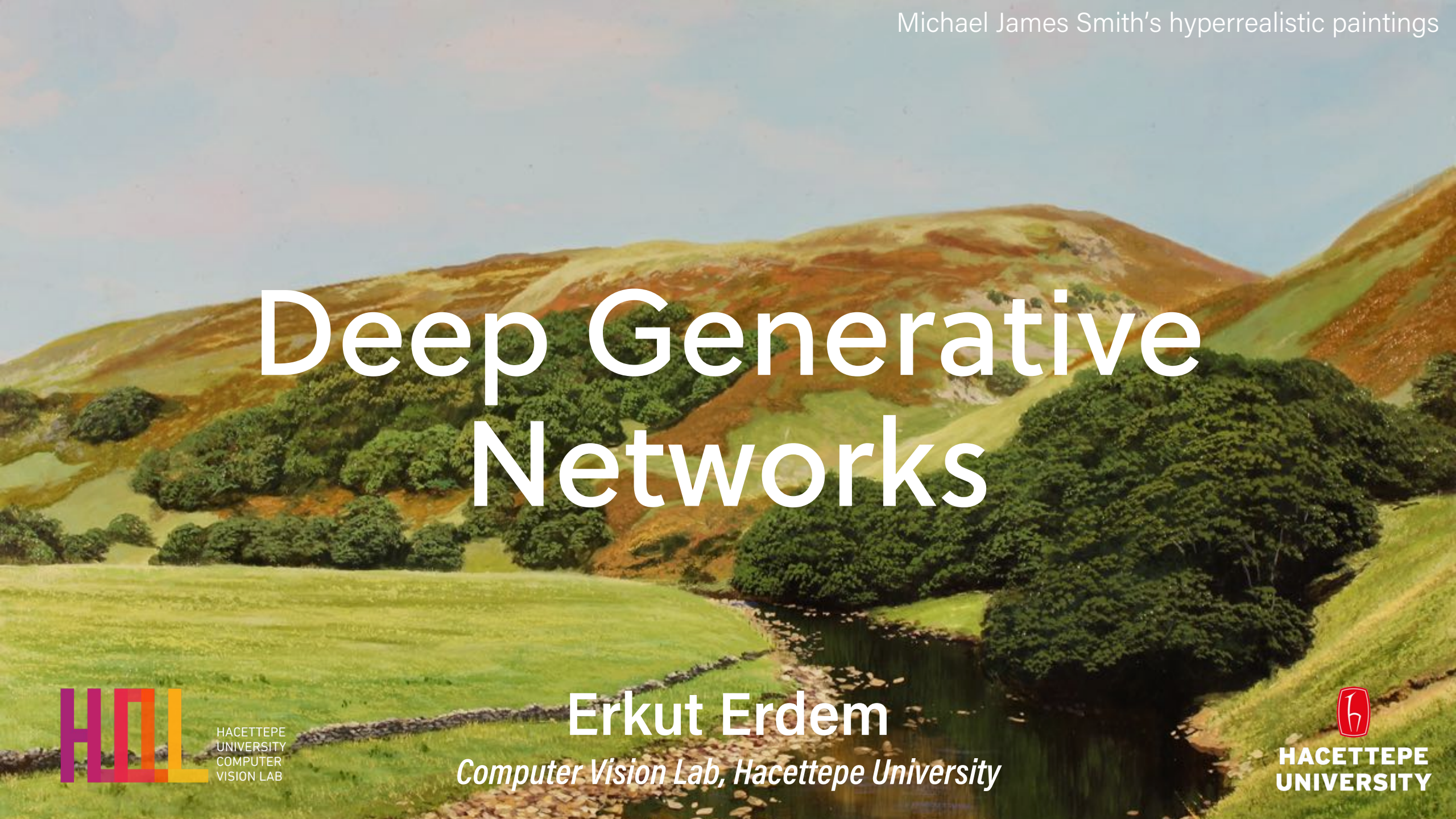


Deep Generative Networks

Erkut Erdem

Computer Vision Lab, Hacettepe University

HACETTEPE UNIVERSITY COMPUTER VISION LAB

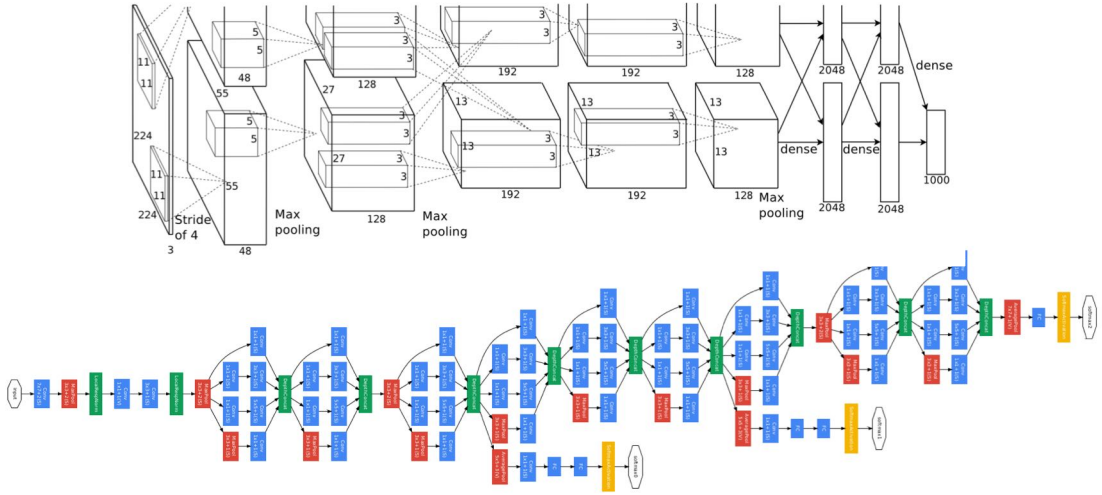
HACETTEPE UNIVERSITY

Lecture overview

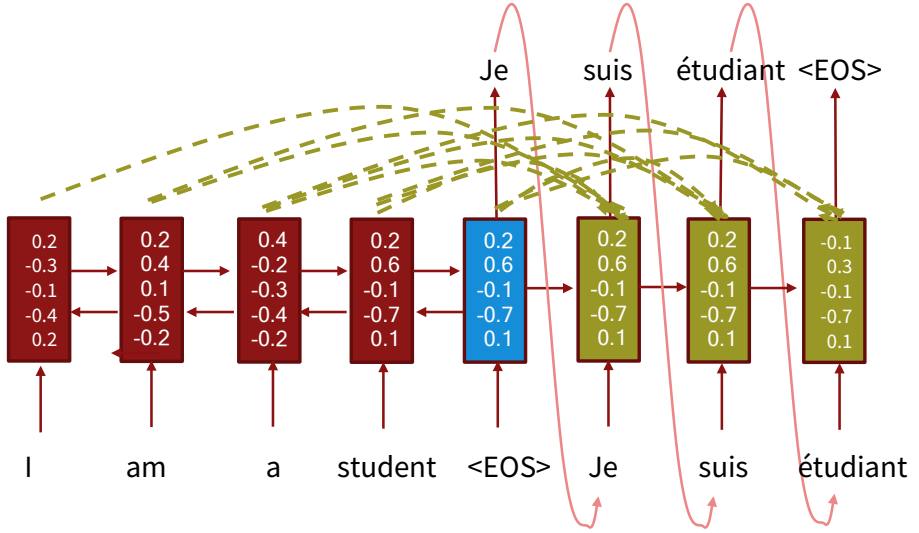
- Why Generative Models
- Types of Generative Models
  - Autoregressive Generative Models
  - Latent Variable Models
  - Transformation Models
- Image Editing with GANs

Deep Supervised Learning: A Success Story

- Obtain lots of input-output examples
- Train a deep neural network



Deep CNN

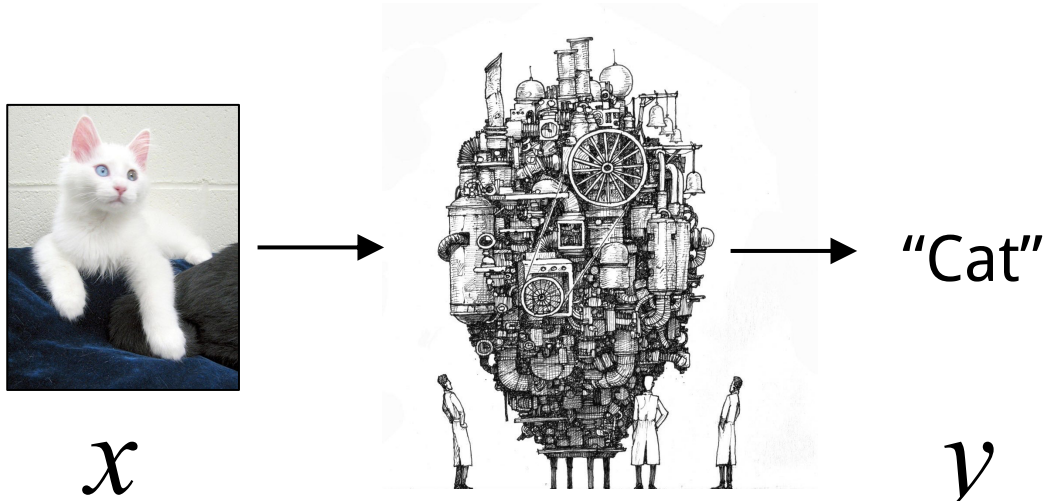


RNN with attention

- Achieve superior results

Discriminative vs. Generative Models

Discriminative models

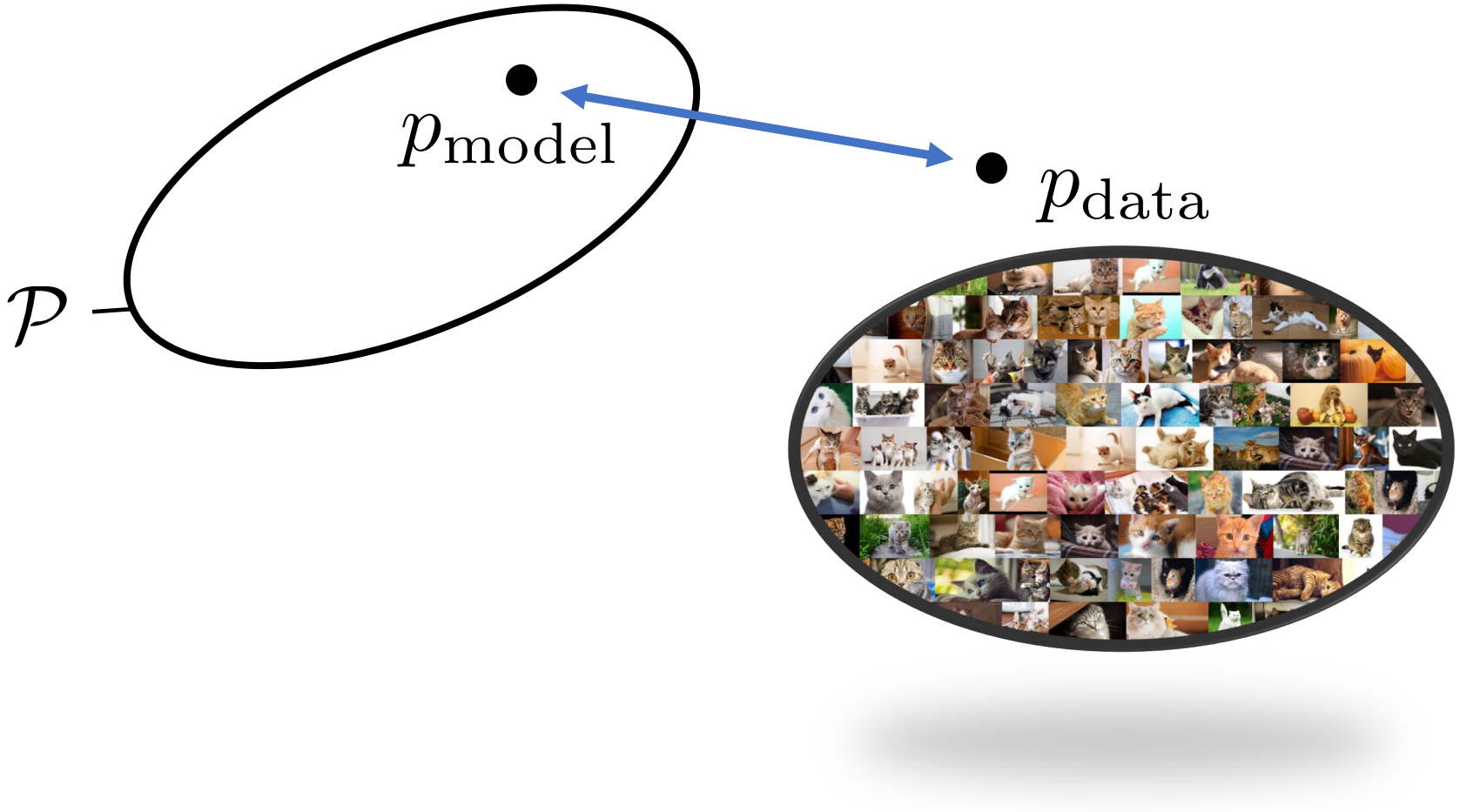


Goal: Learn a function to map x -> y

Generative models

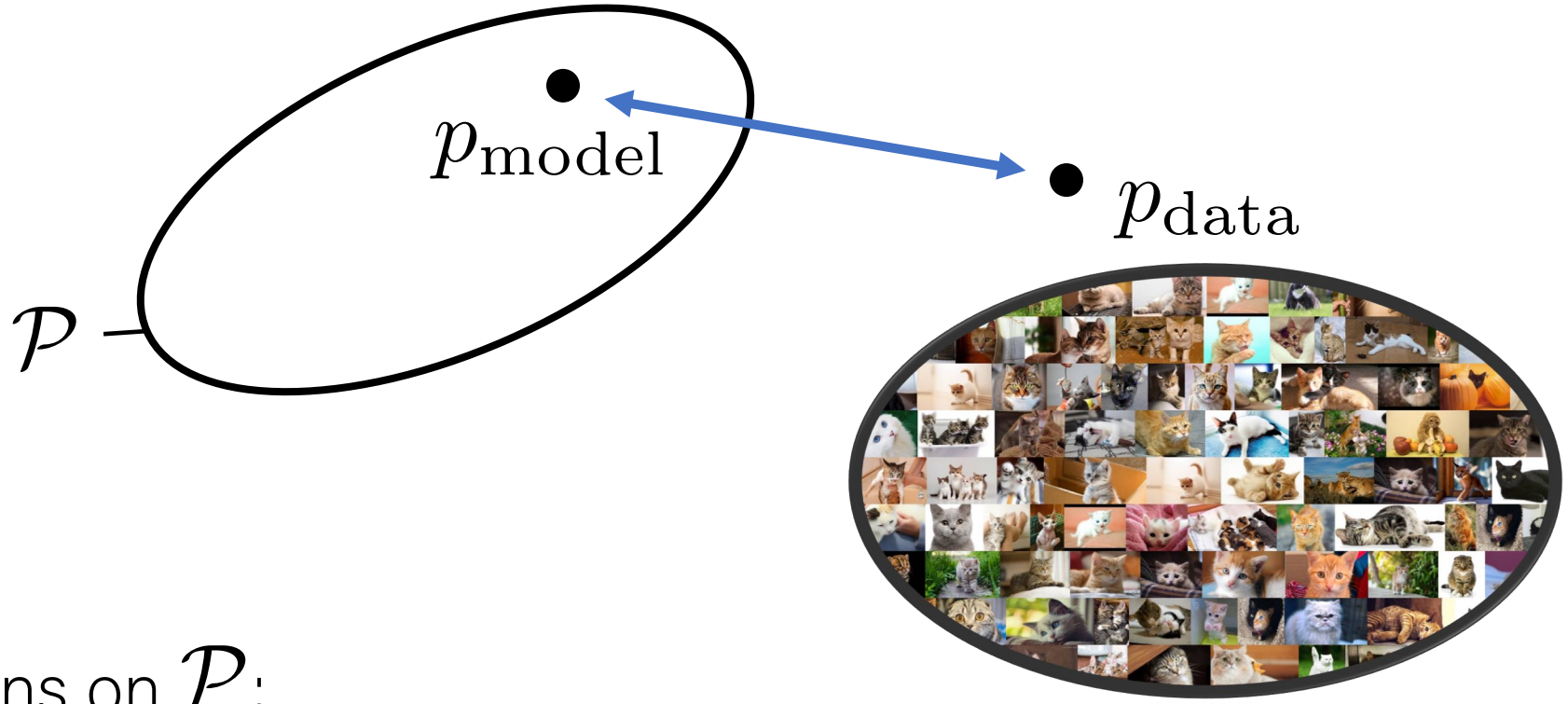
Goal: Learn some underlying hidden structure of the training samples to generate novel samples from same data distribution

Generative Modeling



Slide adapted from Sebastian Nowozin

Generative Modeling



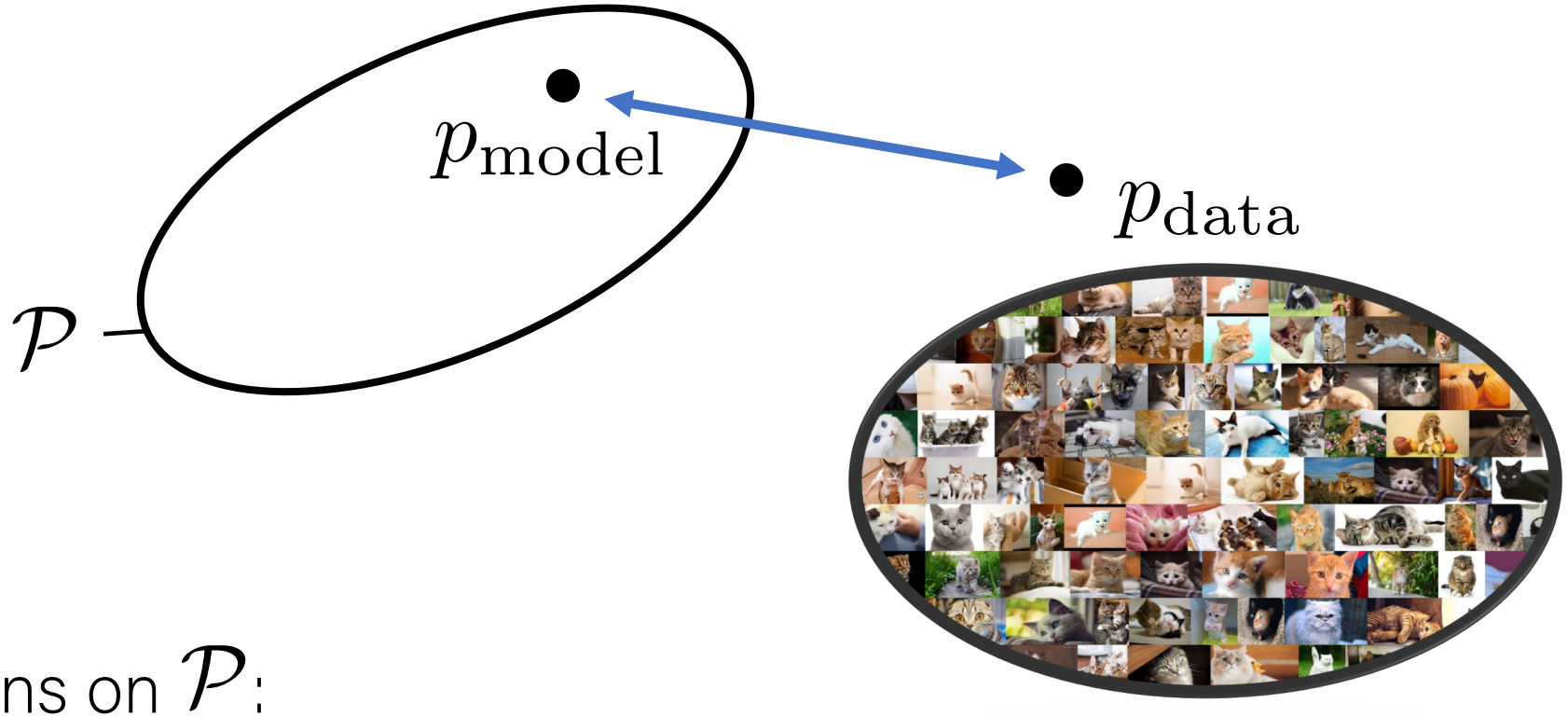
Assumptions on $\mathcal{P}$:

- tractable sampling

Training examples Model samples

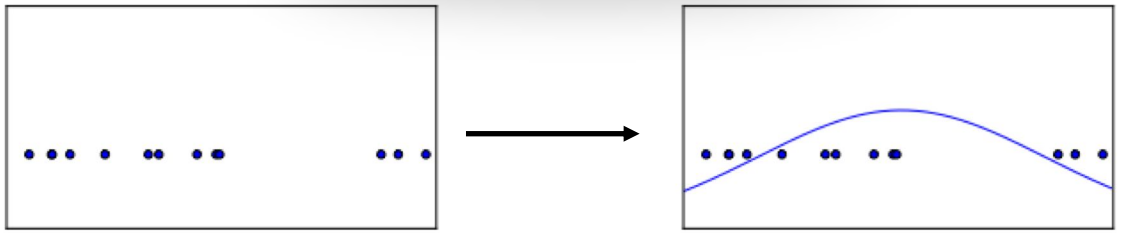
Slide adapted from Sebastian Nowozin

Generative Modeling



Assumptions on $\mathcal{P}$:

- tractable sampling
- tractable likelihood function



Why study deep generative models?

- Go beyond associating inputs to outputs
- Understand high-dimensional, complex probability distributions
- Discover the “true” structure of the data
  - Detect surprising events in the world (*anomaly detection*)
  - Missing Data (*semi-supervised learning*)
  - Generate models for planning (*model-based reinforcement learning*)

Three Broad Categories

- Autoregressive Models
  - PixelCNN
- Latent Variable Models
  - Variational Autoencoders
- Transformation Models
  - Generative Adversarial Networks (GANs)

Autoregressive Generative Models

Learning the Distribution of Natural Data

$$p(\mathbf{x}) = \prod_i p(x_i|\mathbf{x}_<)$$

1D sequences such as text or sound

$$p(\mathbf{x}) = \prod_j \prod_i p(x_{i,j}|\mathbf{x}_<)$$

2D tensors such as images

$$p(\mathbf{x}) = \prod_k \prod_j \prod_i p(x_{i,j,k}|\mathbf{x}_<)$$

3D tensors such as videos

- Fully visible belief networks — [Frey et al.,1996] [Frey, 1998]
- NADE/MADE — [Larochelle and Murray, 2011] [Germain et al., 2015]
- PixelRNN/PixelCNN (Images) — [van den Oord, Kalchbrenner, Kavukcuoglu, 2016] [van den Oord, Kalchbrenner, Vinyals, et al., 2016]
- Video Pixel Nets (Videos) — [Kalchbrenner, van den Oord, Simonyan, et al., 2016]
- ByteNet (Language/seq2seq) — [Kalchbrenner, Espeholt, Simonyan, et al., 2016]
- WaveNet (Audio) — [van den Oord, Dieleman, Zen, et al., 2016]

Slide adapted from Nal Kalchbrenner

PixelCNN

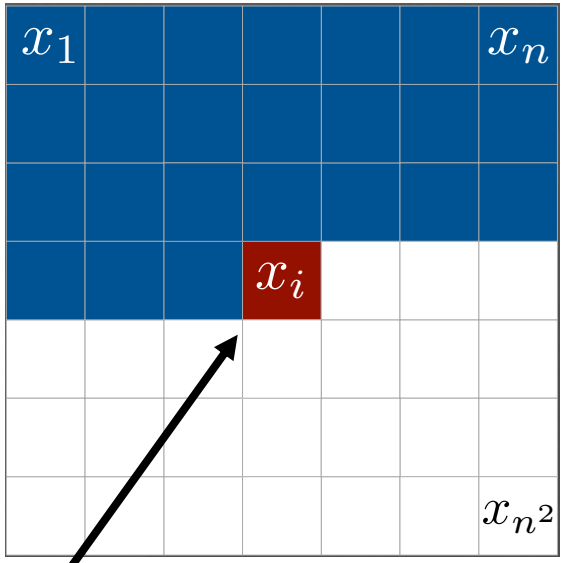
$P($  $)$

- approach the generation process as sequence modeling problem
- an explicit density model

Slide adapted from Nal Kalchbrenner

PixelCNN

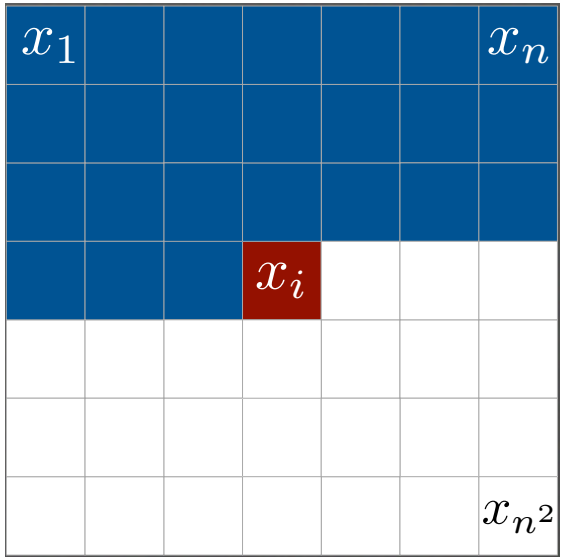
$P($  $)$



Slide adapted from Nal Kalchbrenner

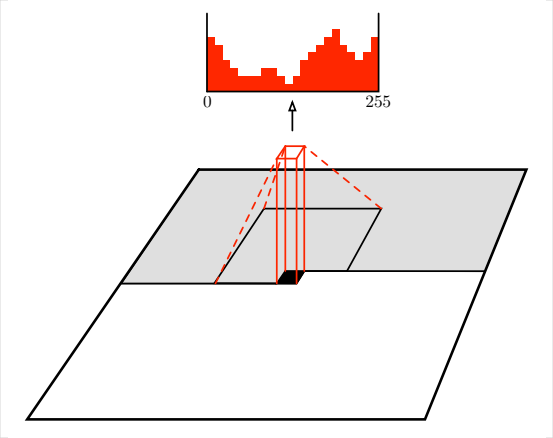
PixelCNN

$P($  $)$



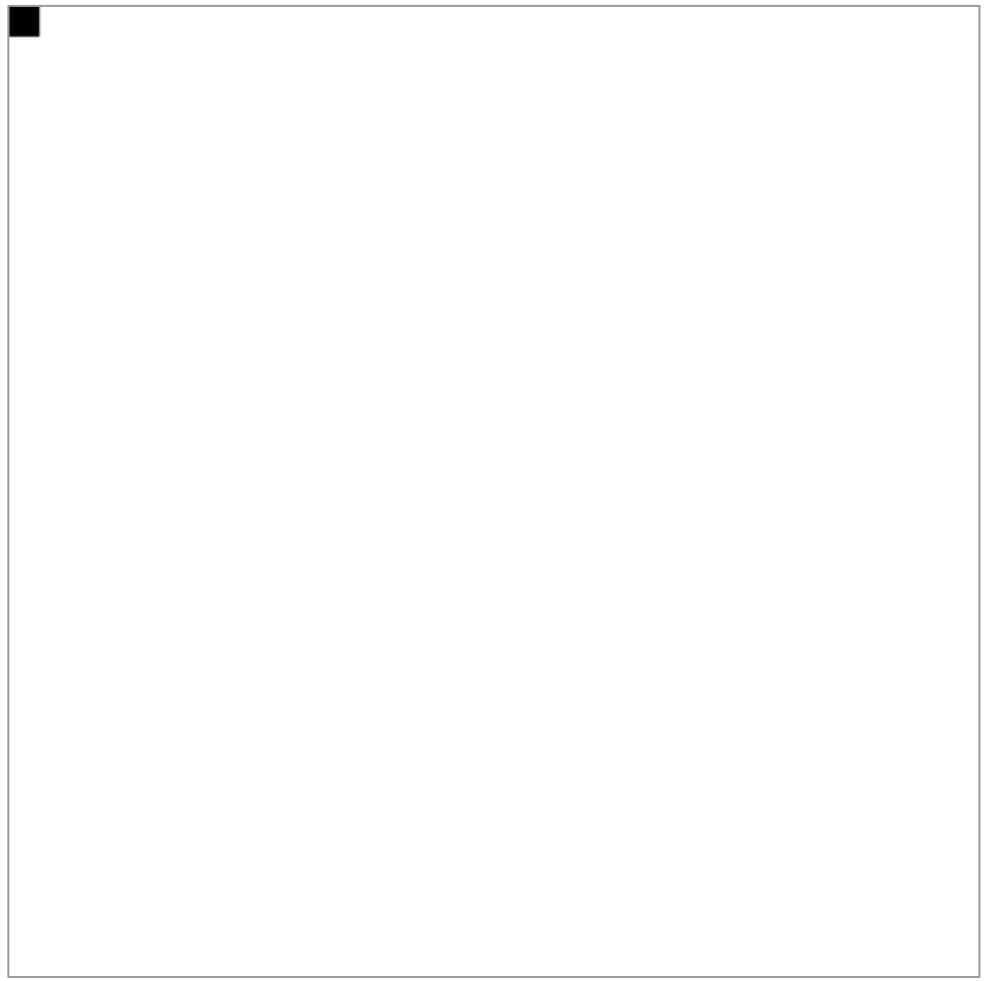
By chain rule and using pixels as variables,

$$P(X) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2)$$



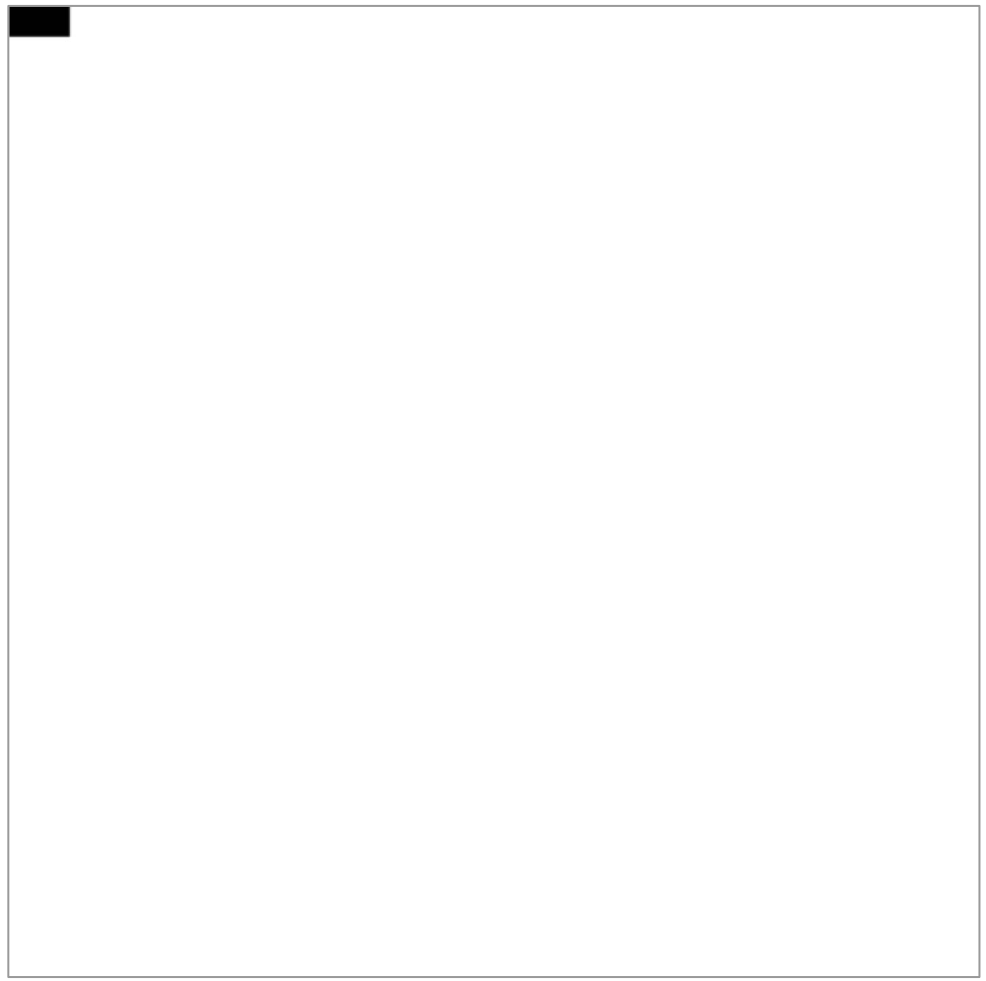
Slide adapted from Nal Kalchbrenner

PixelCNN



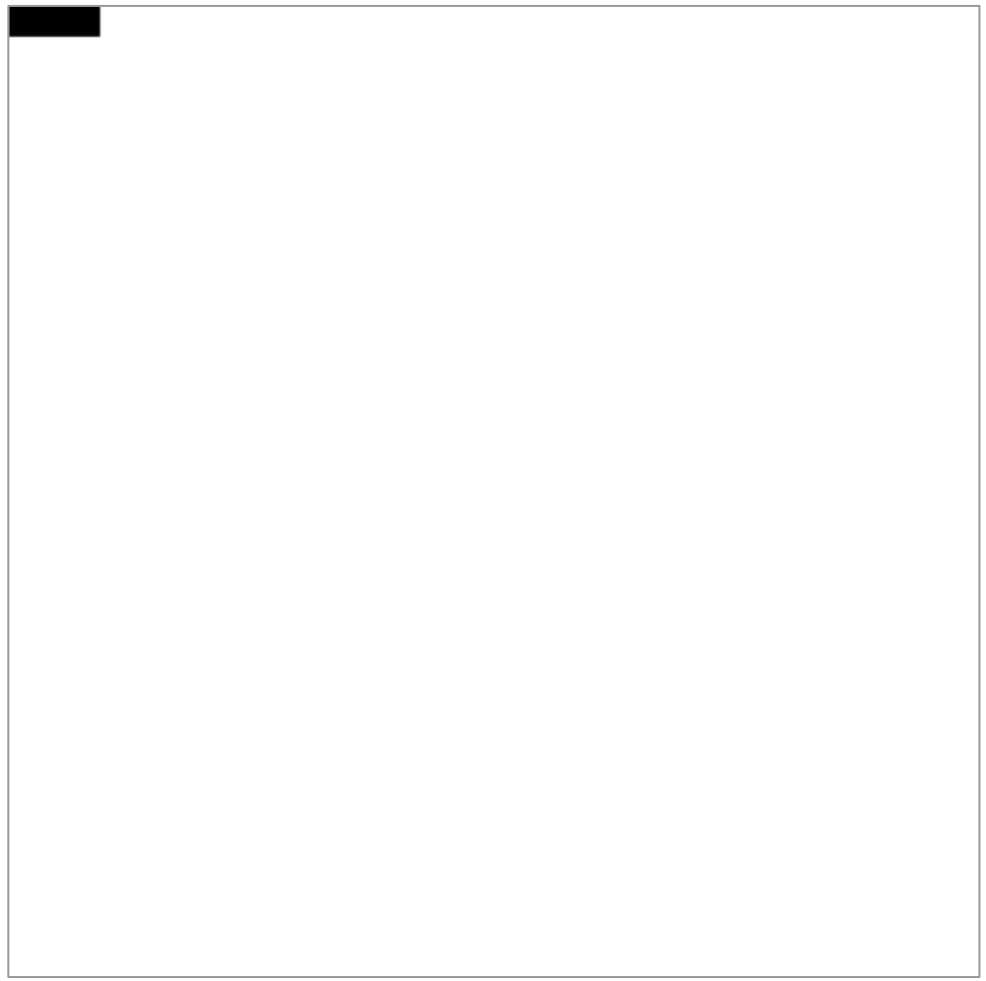
Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN



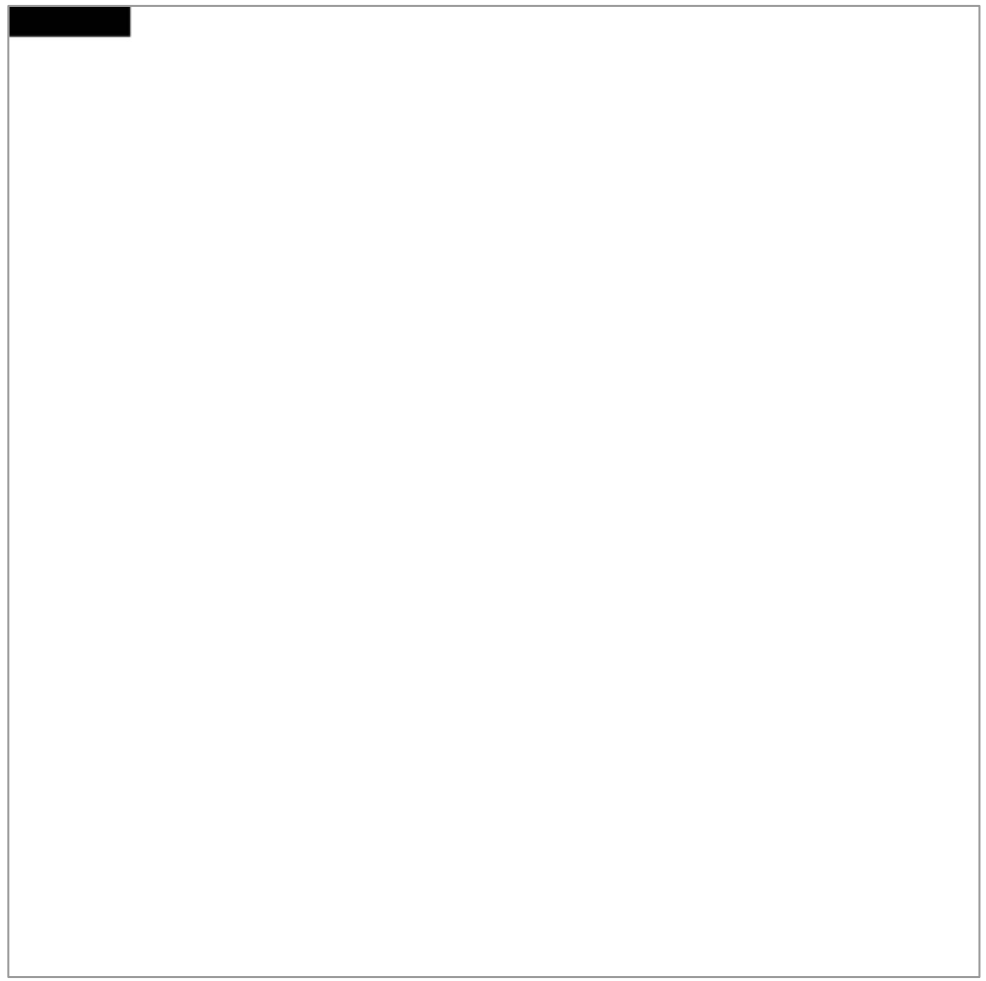
Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN



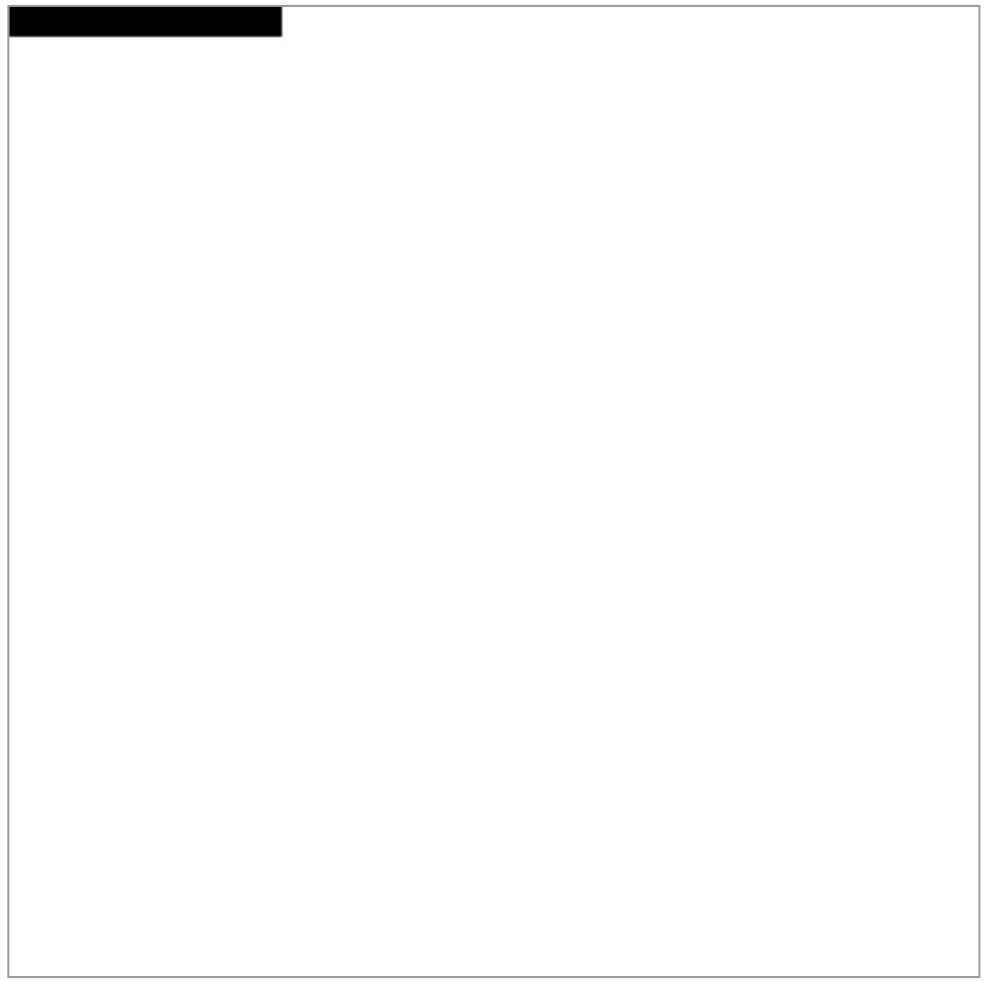
Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN



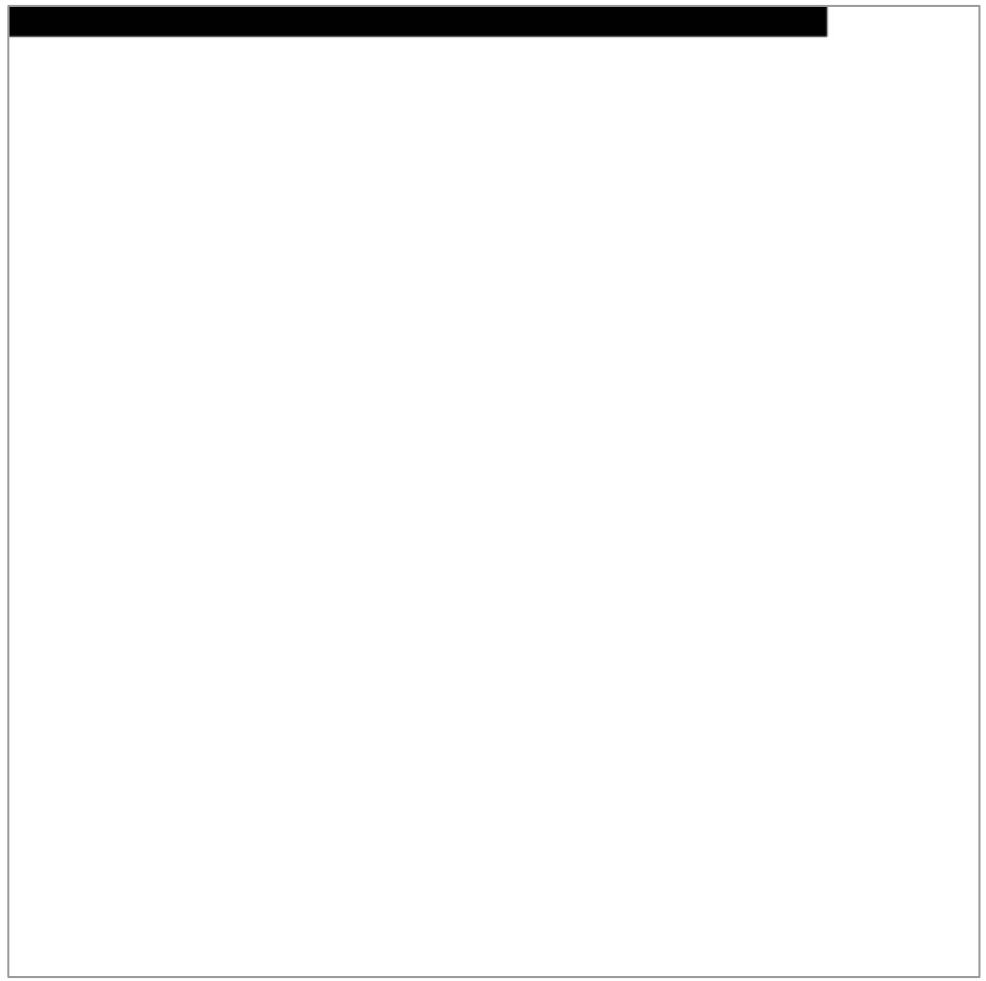
Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN



Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN

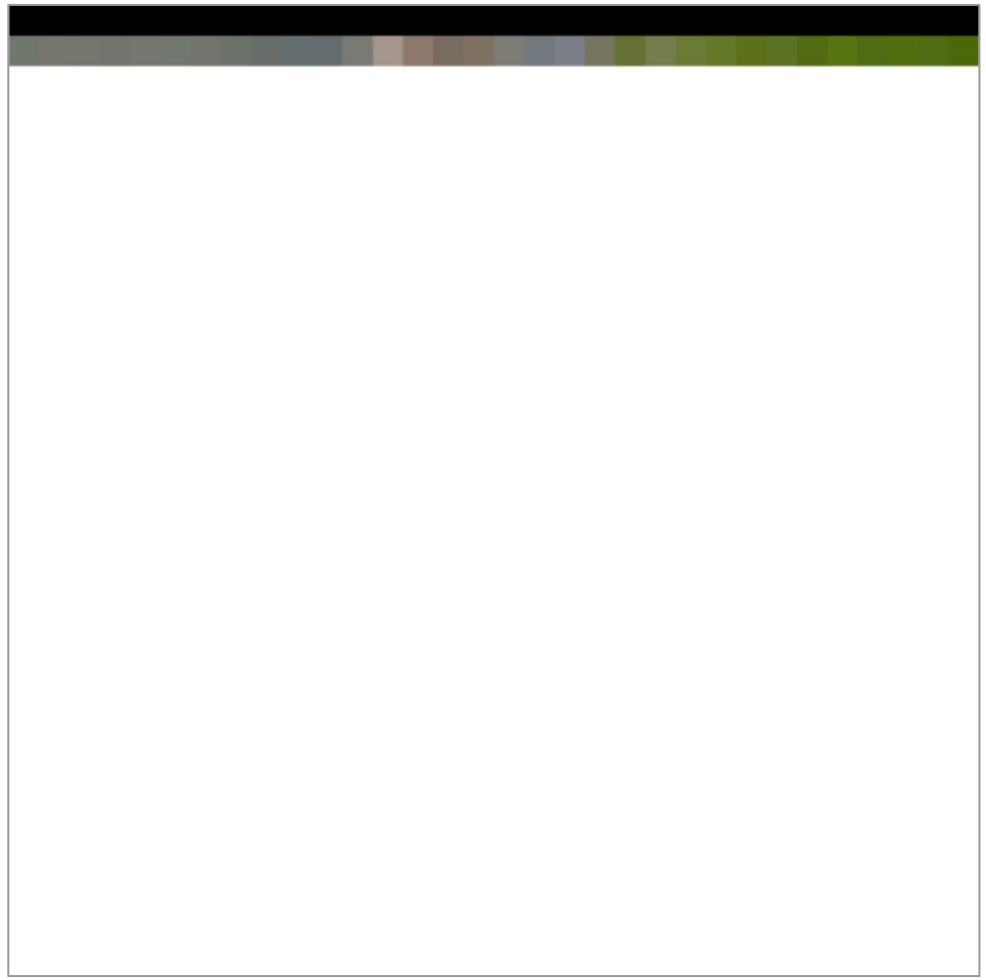


Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN

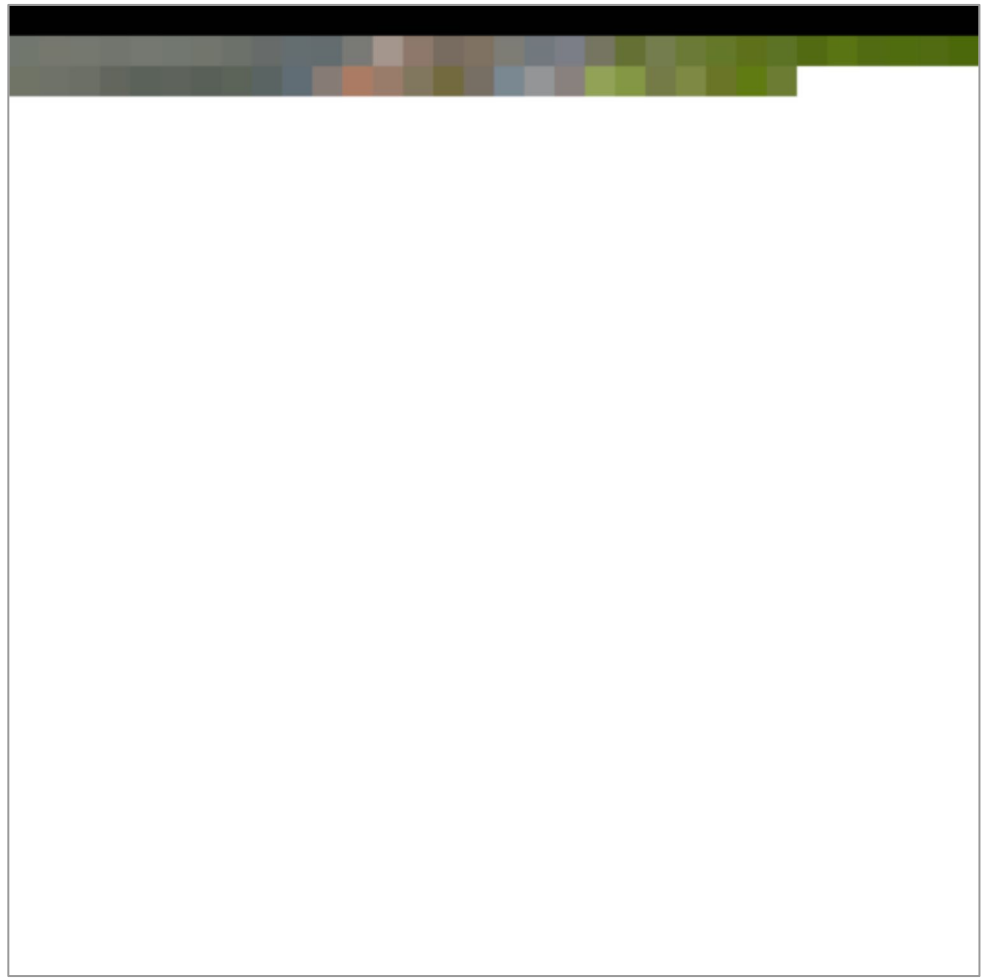
Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN



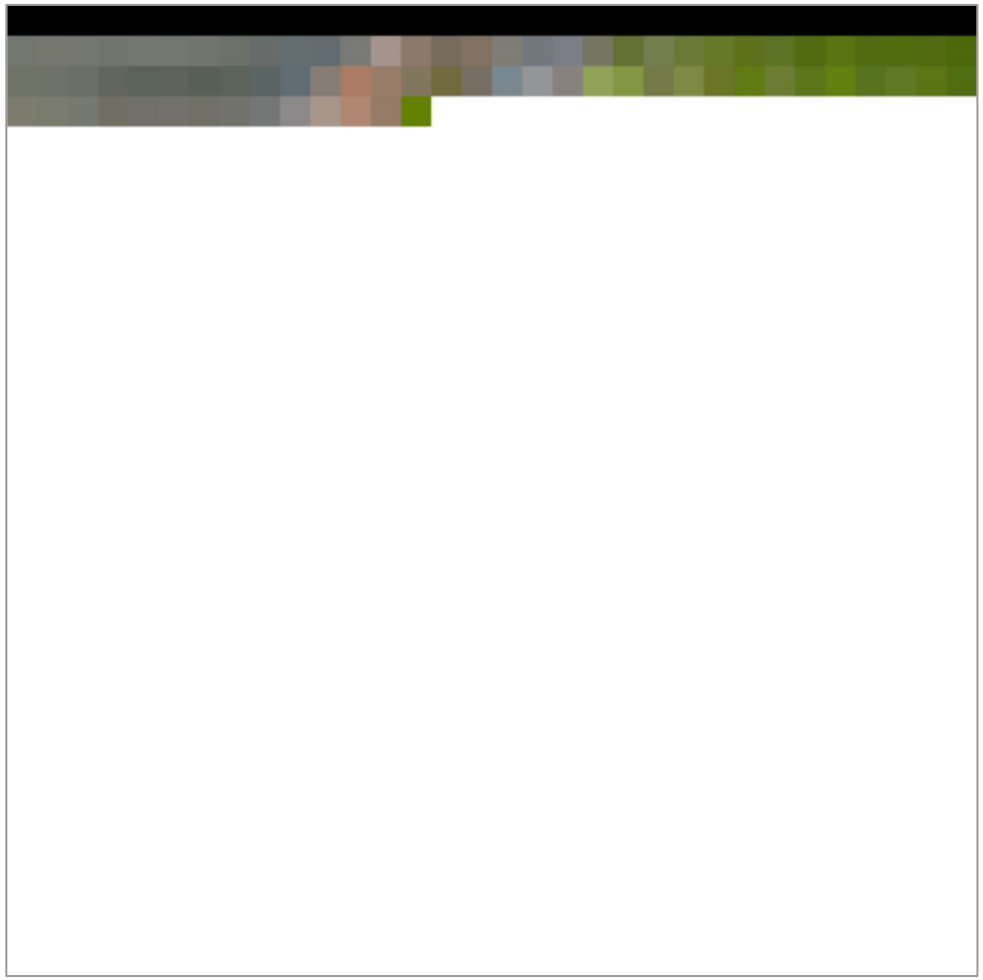
Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN



Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN

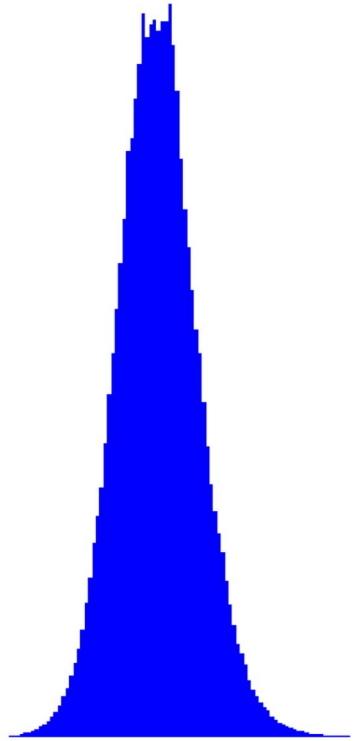


Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN

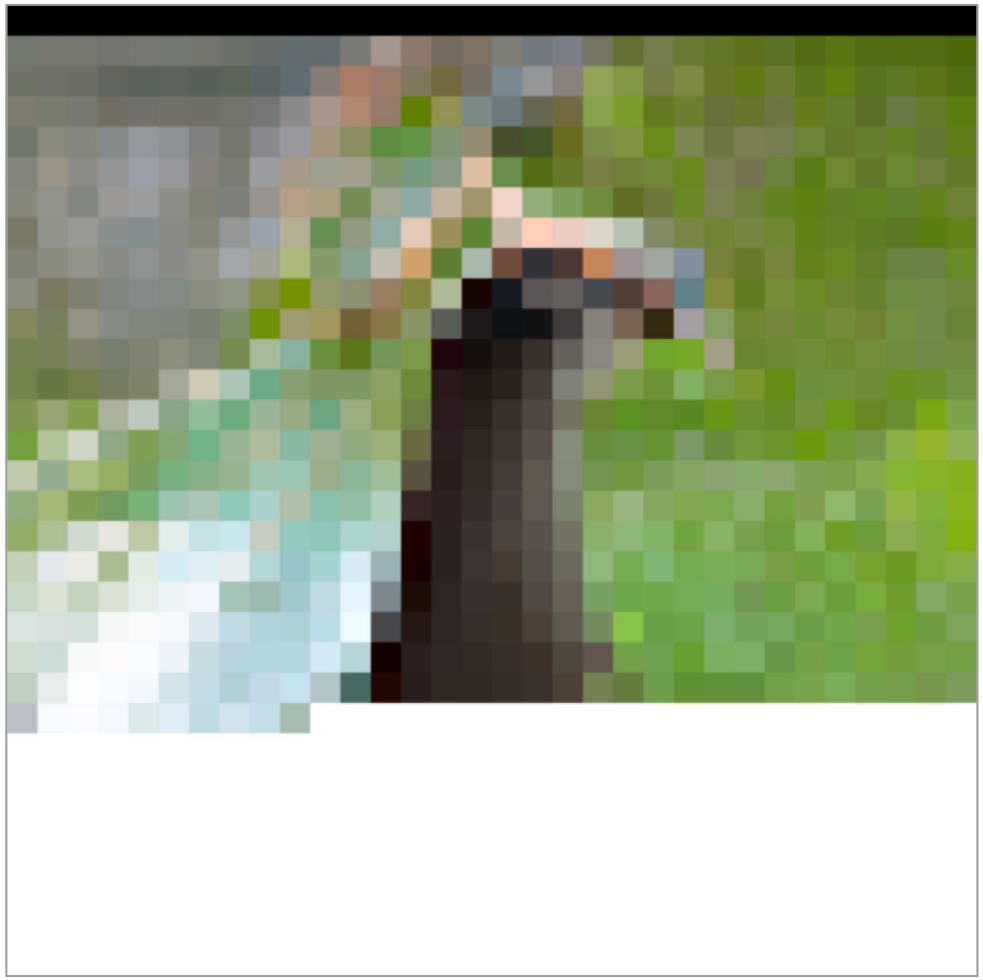
Slide adapted from
Oriol Vinyals and Navdeep Jaitly

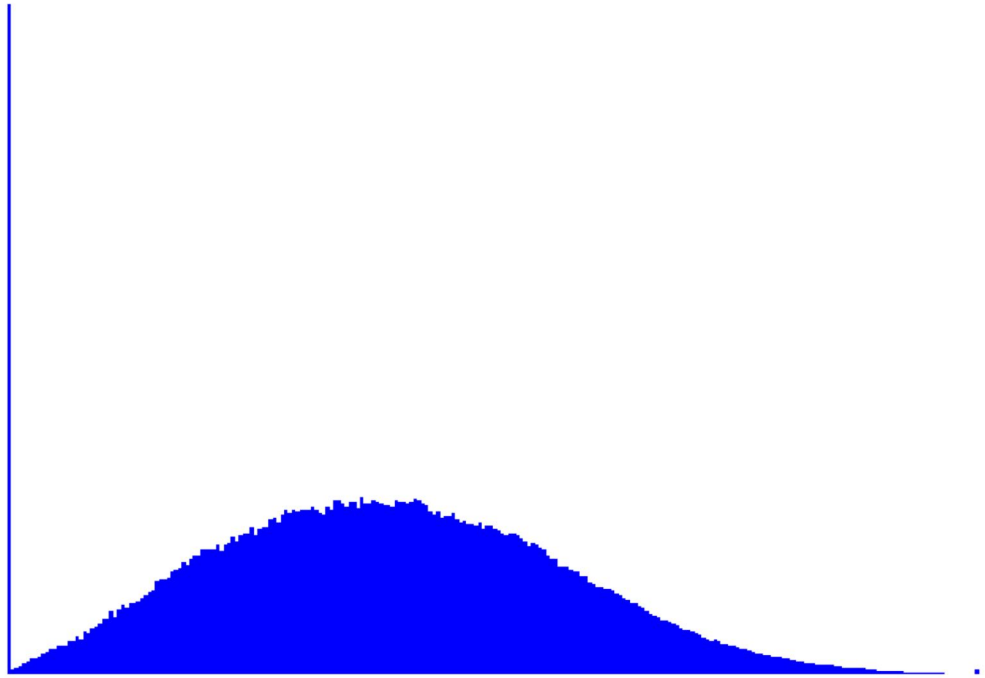
PixelCNN – Softmax Sampling



Slide adapted from
Oriol Vinyals and Navdeep Jaitly

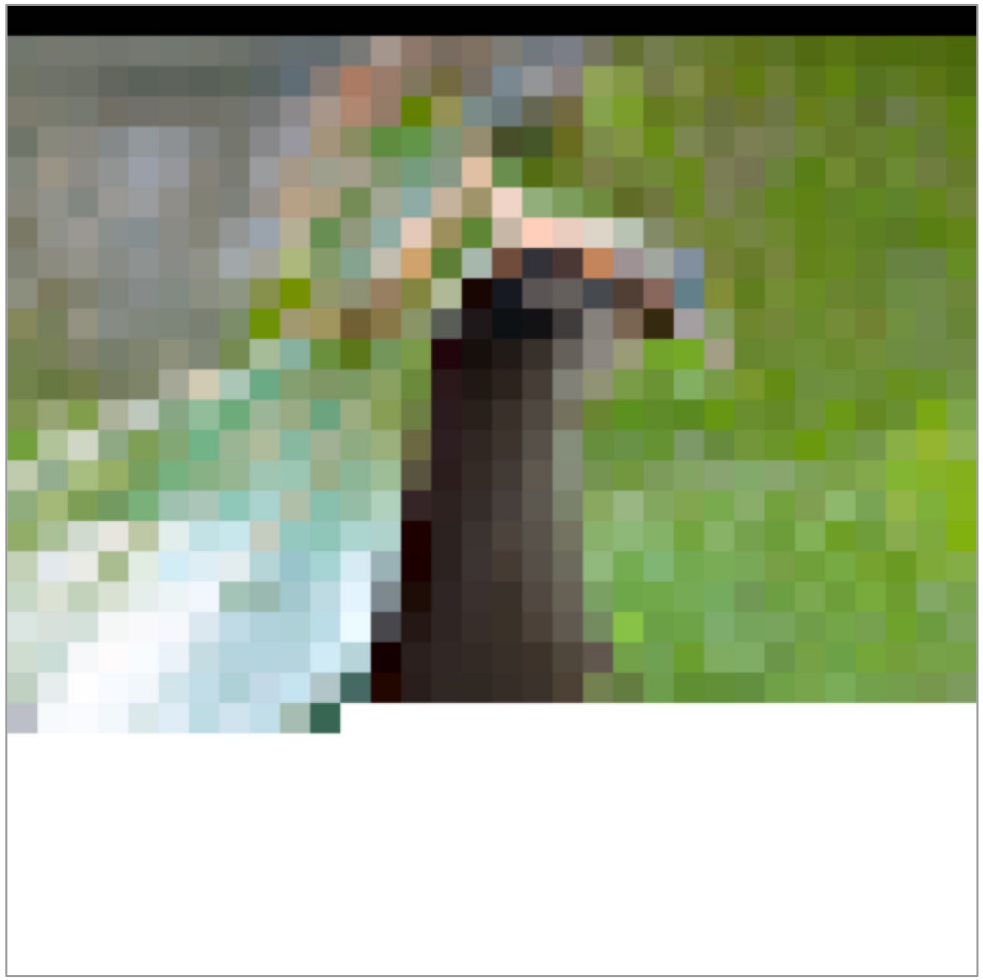
PixelCNN – Softmax Sampling

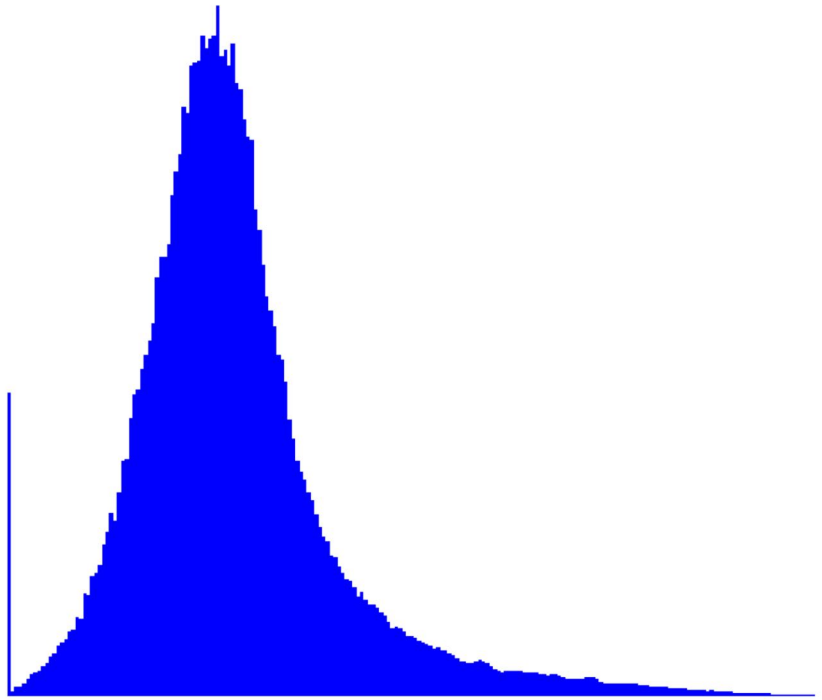




Slide adapted from
Oriol Vinyals and Navdeep Jaitly

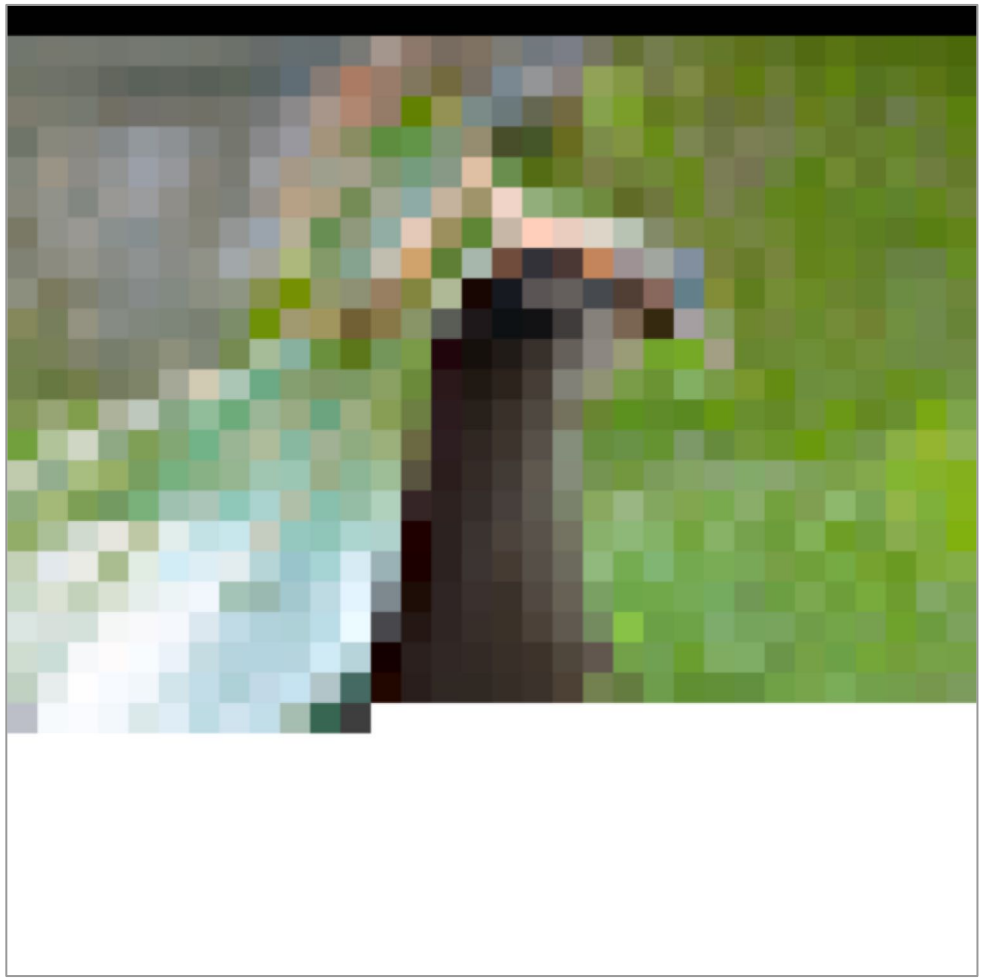
PixelCNN – Softmax Sampling

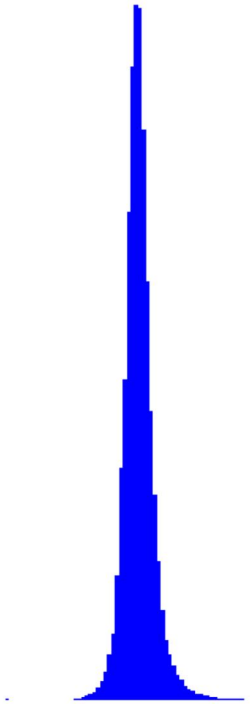




Slide adapted from
Oriol Vinyals and Navdeep Jaitly

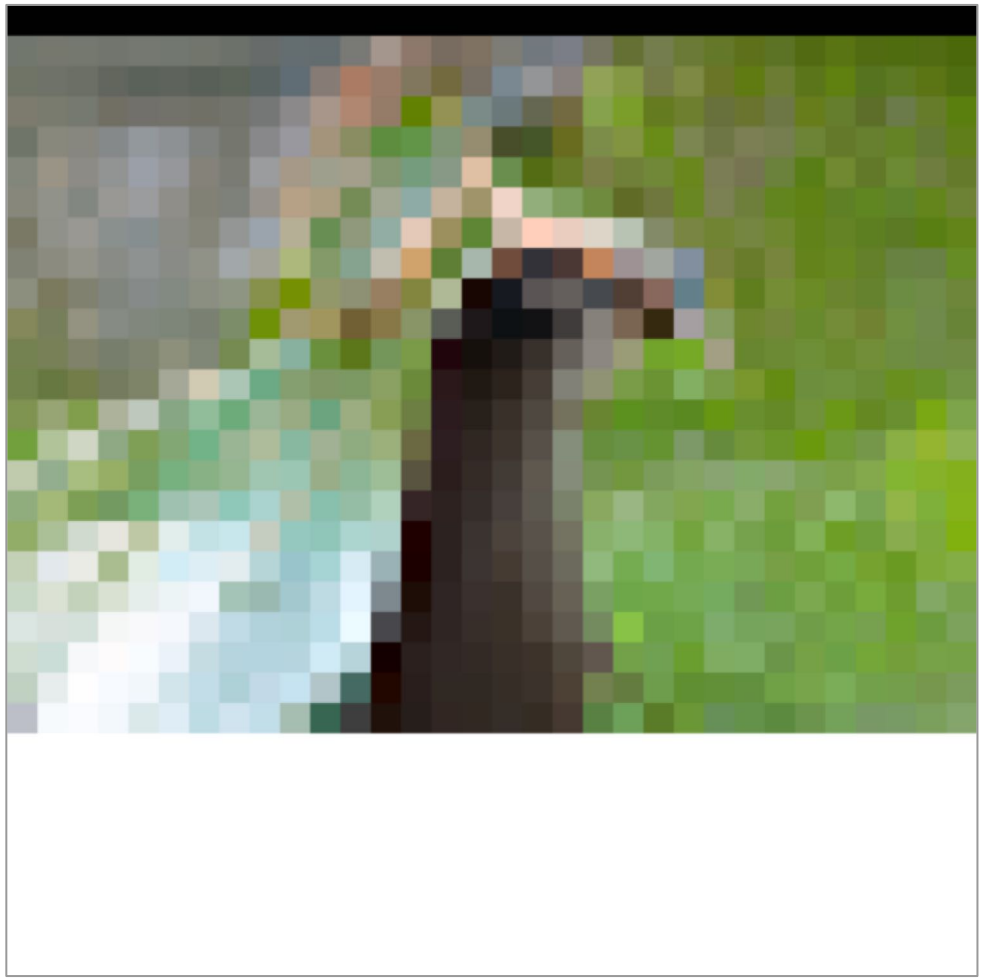
PixelCNN – Softmax Sampling

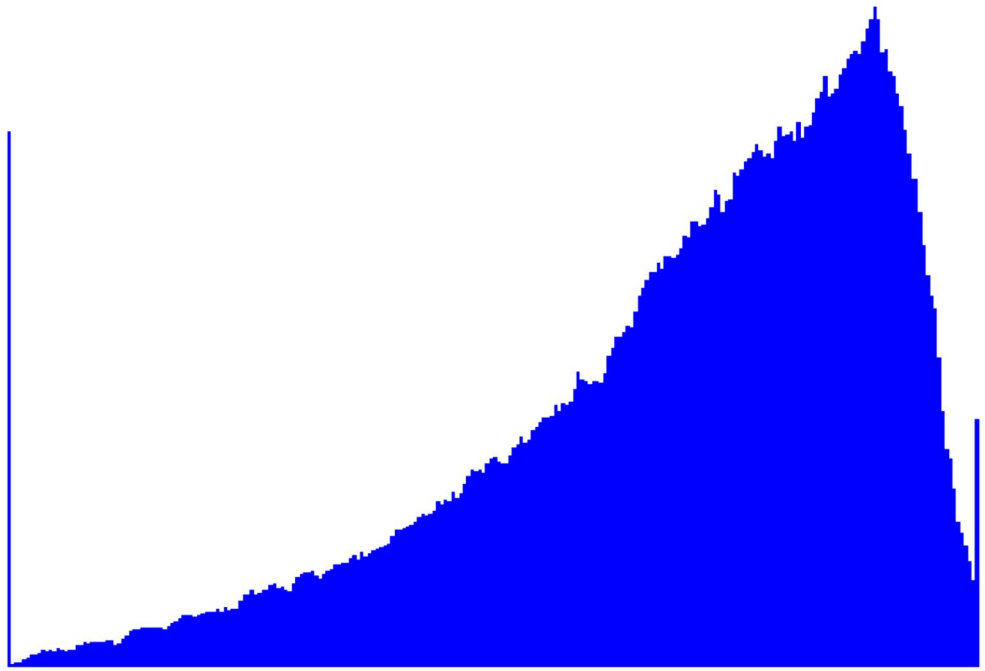




Slide adapted from
Oriol Vinyals and Navdeep Jaitly

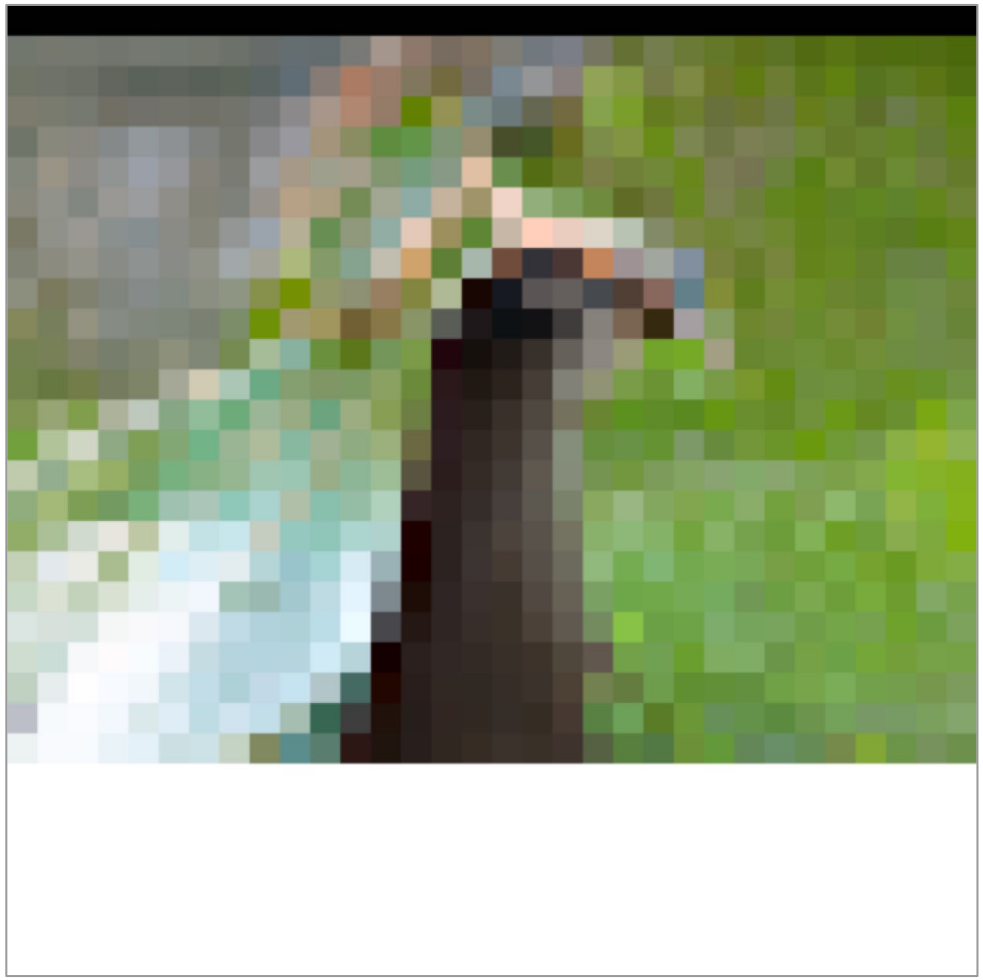
PixelCNN – Softmax Sampling

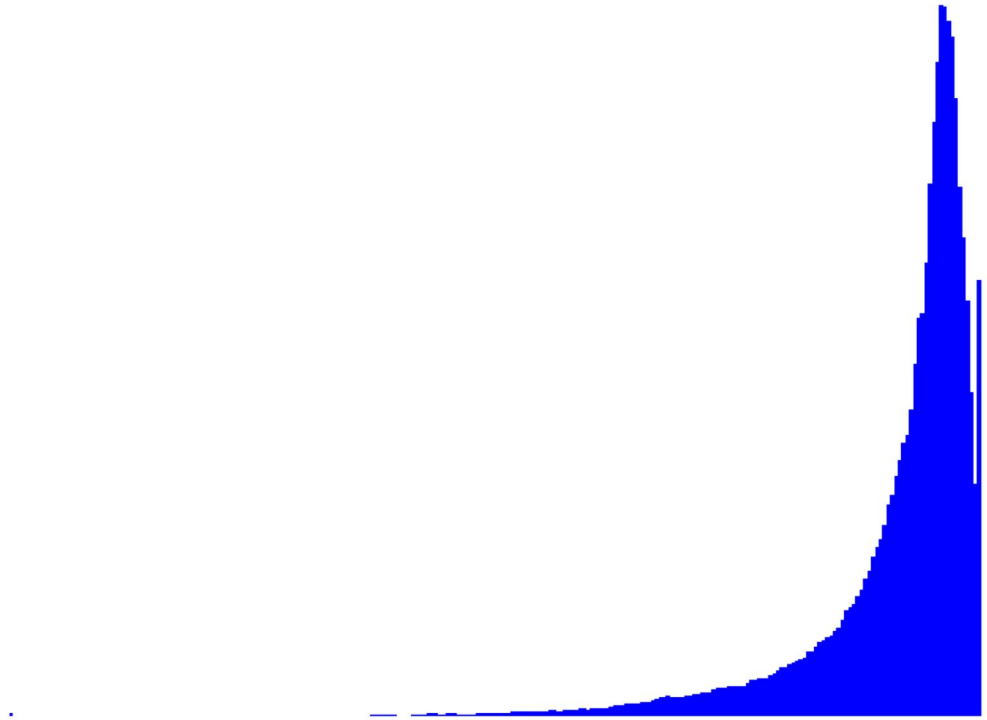




Slide adapted from
Oriol Vinyals and Navdeep Jaitly

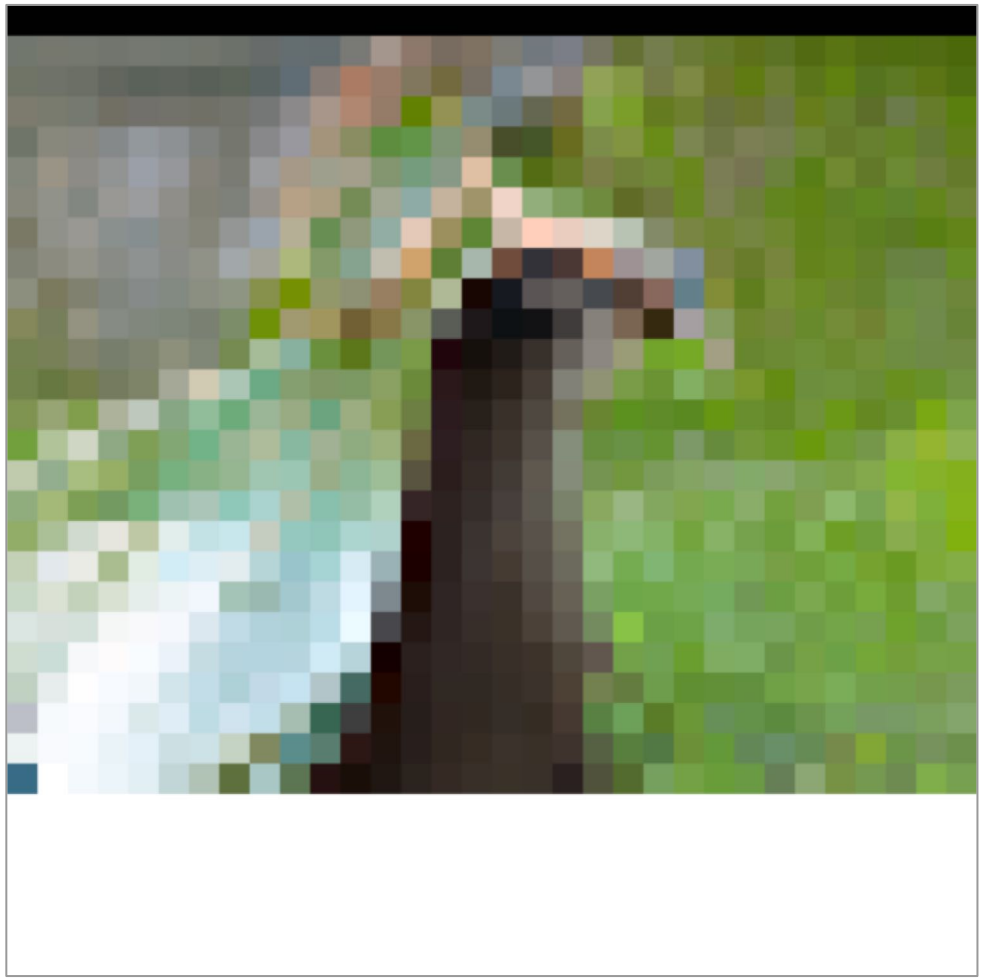
PixelCNN – Softmax Sampling

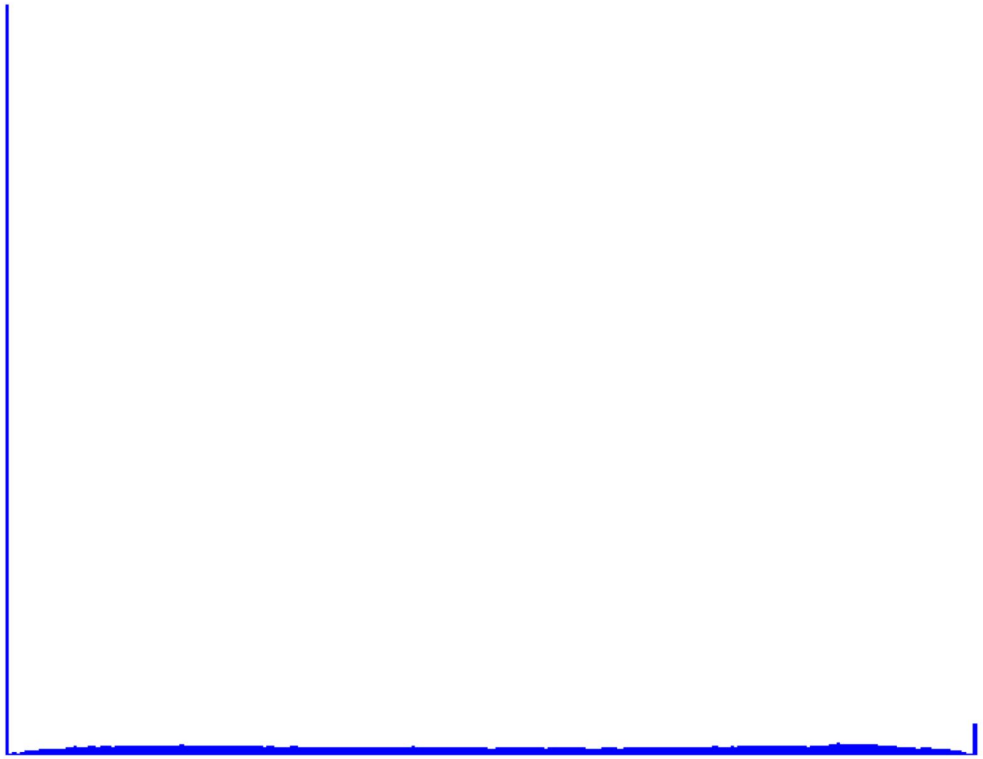




Slide adapted from
Oriol Vinyals and Navdeep Jaitly

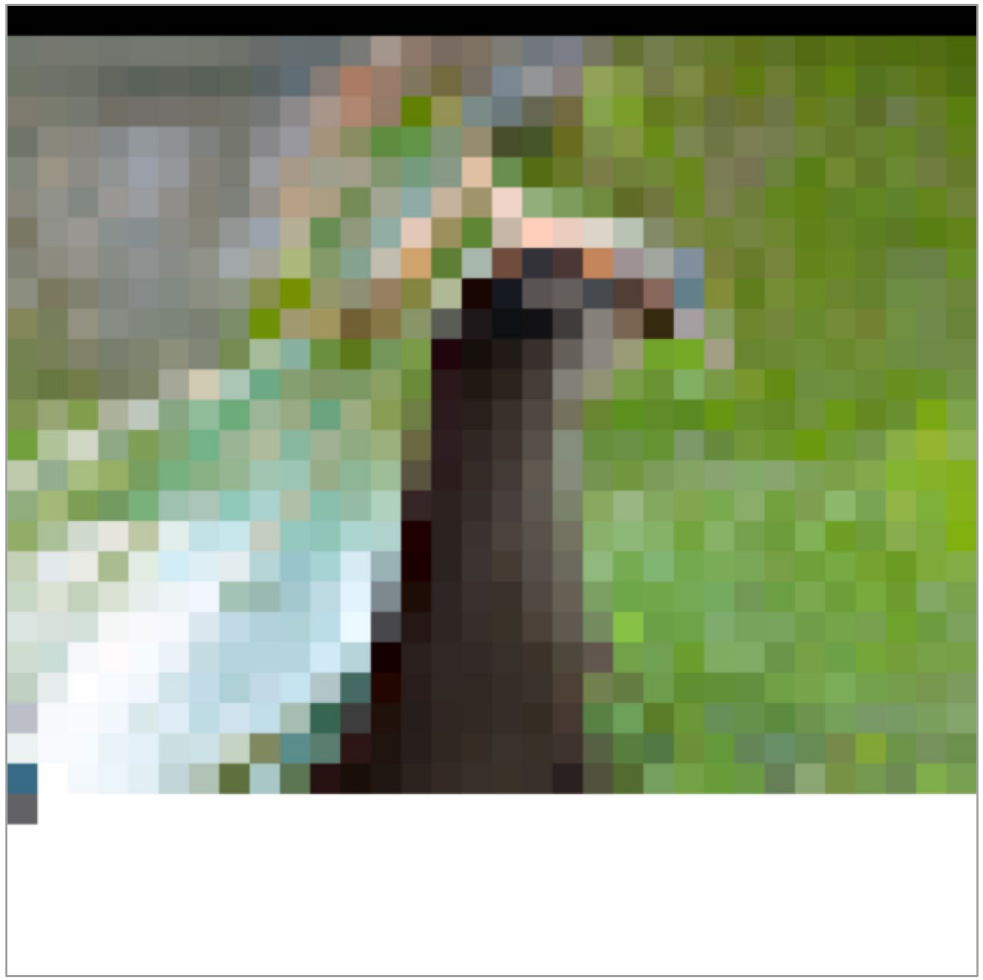
PixelCNN – Softmax Sampling

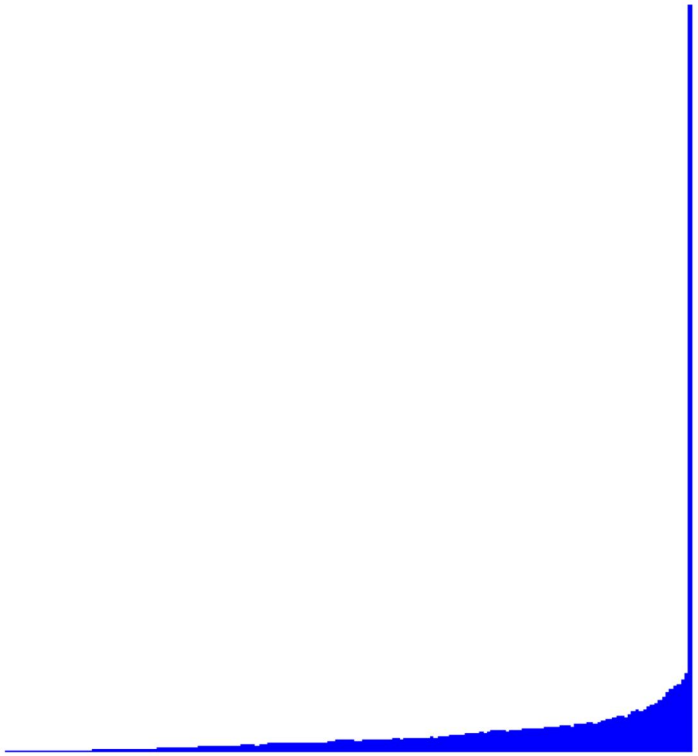




Slide adapted from
Oriol Vinyals and Navdeep Jaitly

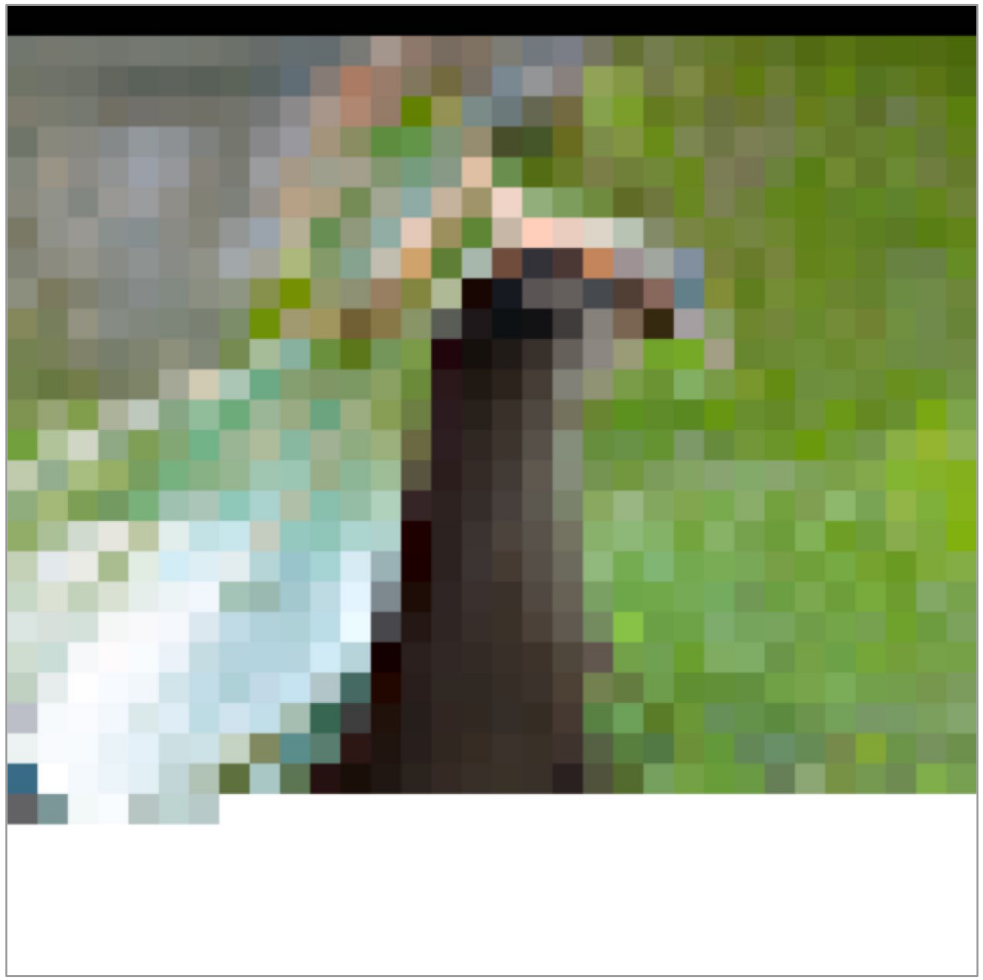
PixelCNN – Softmax Sampling

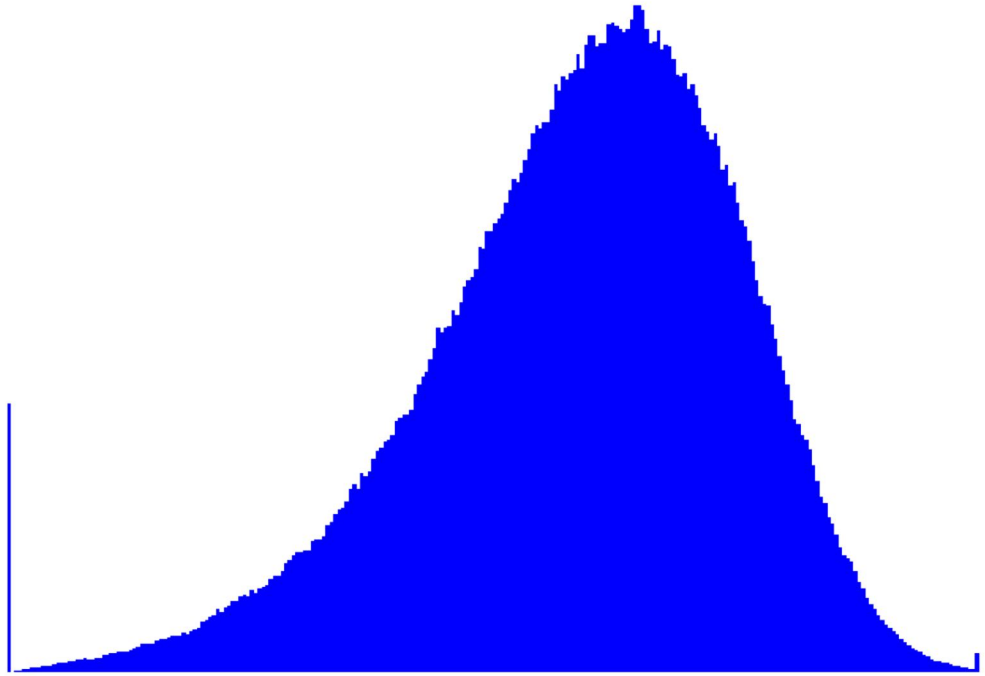




Slide adapted from
Oriol Vinyals and Navdeep Jaitly

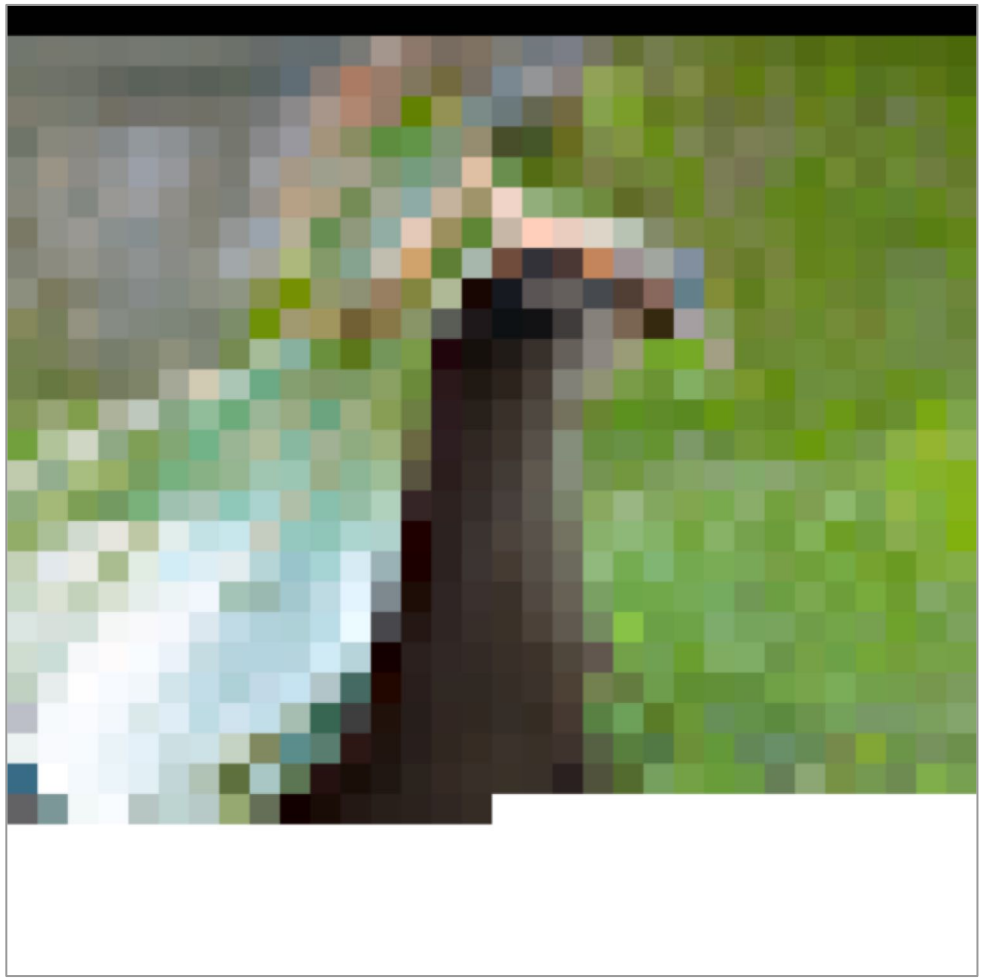
PixelCNN – Softmax Sampling

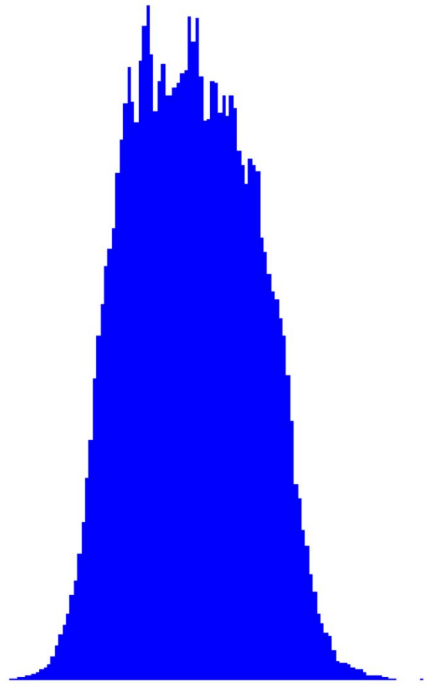




Slide adapted from
Oriol Vinyals and Navdeep Jaitly

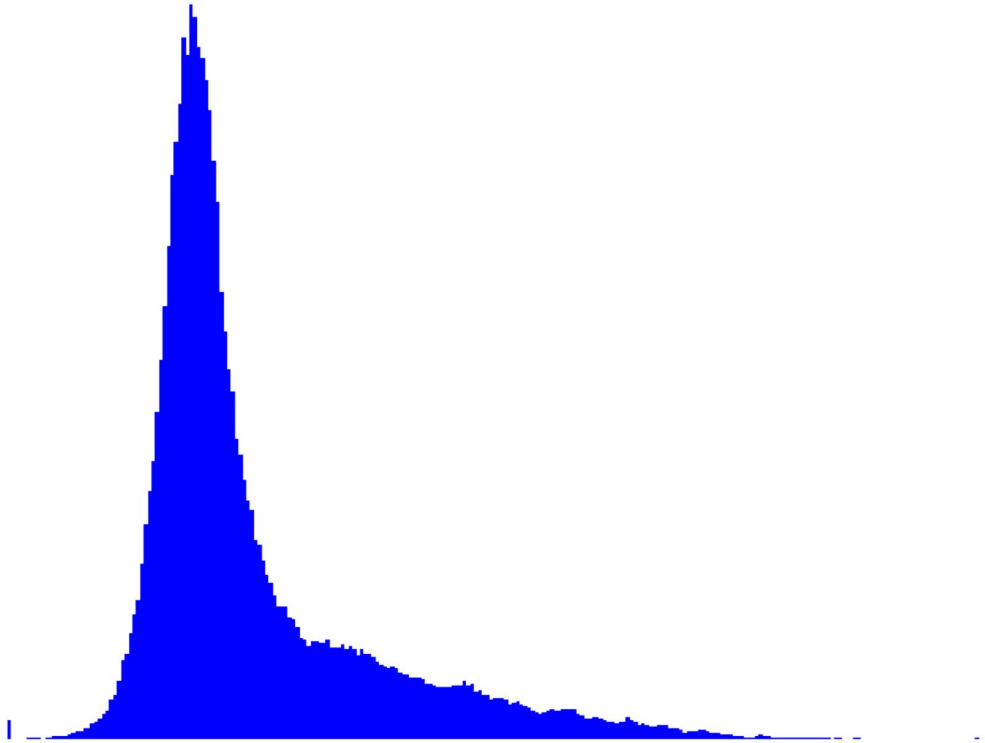
PixelCNN – Softmax Sampling





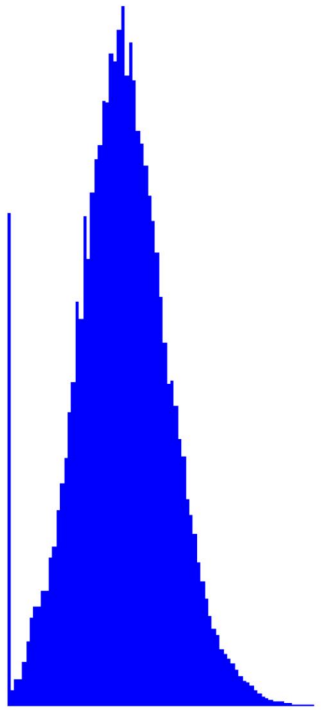
Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN – Softmax Sampling



Slide adapted from
Oriol Vinyals and Navdeep Jaitly

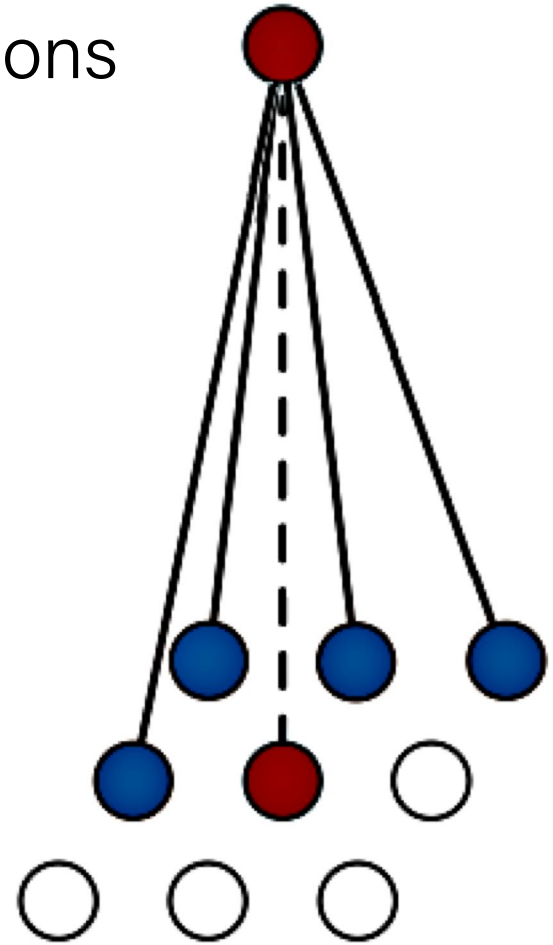
PixelCNN – Softmax Sampling

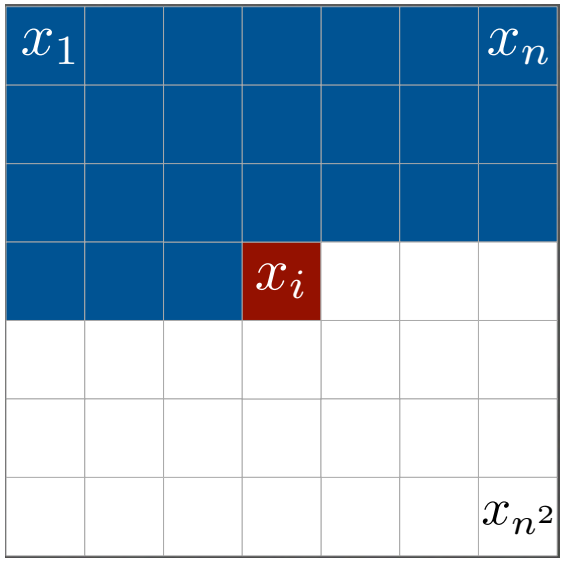


Slide adapted from
Oriol Vinyals and Navdeep Jaitly

PixelCNN

use masked convolutions to enforce the autoregressive relationship



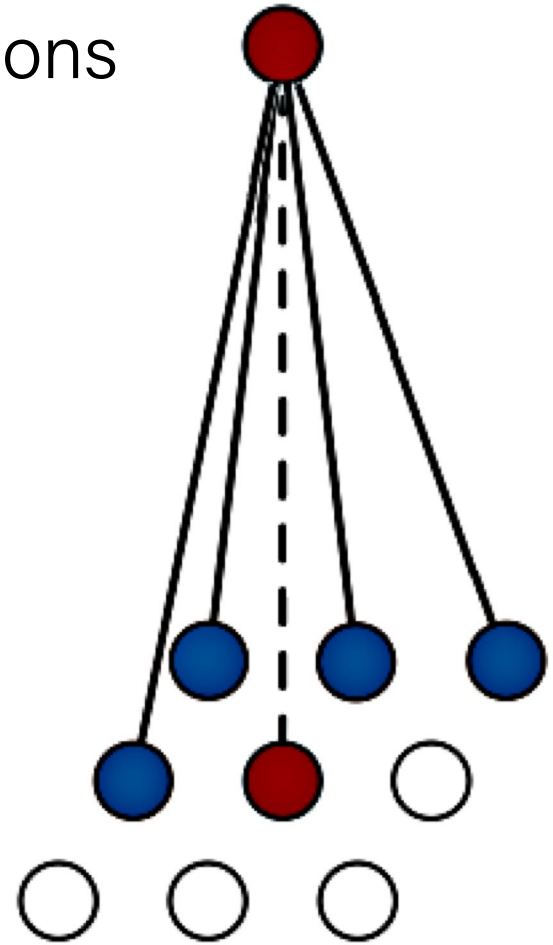


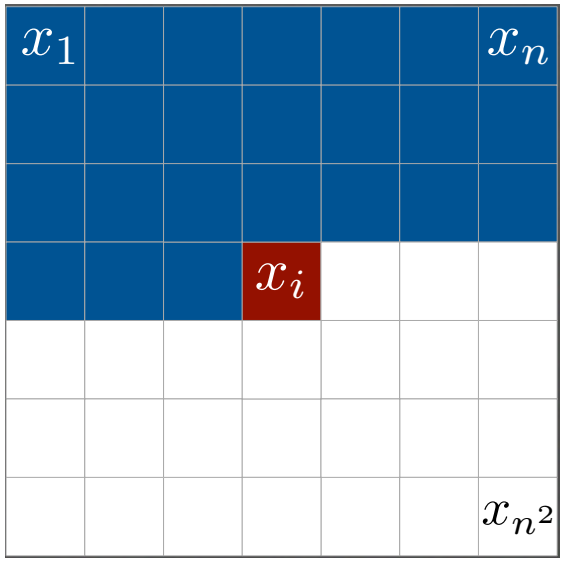
1	1	1	1	1
1	1	1	1	1
1	1	0	0	0
0	0	0	0	0
0	0	0	0	0

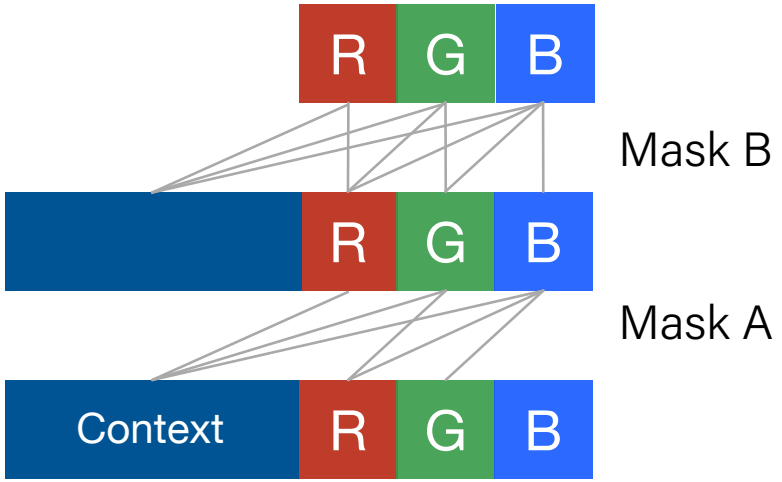
Slide adapted from Nal Kalchbrenner

PixelCNN

use masked convolutions to enforce the autoregressive relationship







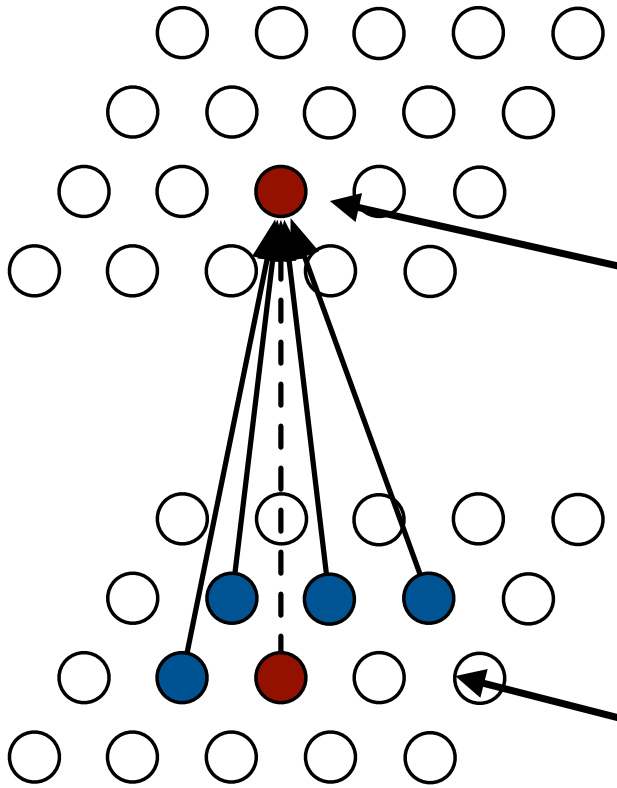
autoregressive over color channels

$$p(x_i \mid \mathbf{x}_{<i}) = p(x_{i,R} \mid \mathbf{x}_{<i})p(x_{i,G} \mid x_{i,R}, \mathbf{x}_{<i})p(x_{i,B} \mid x_{i,R}, x_{i,G}, \mathbf{x}_{<i})$$

Slide adapted from Nal Kalchbrenner

PixelCNN

Multiple layers of masked convolutions



composing multiple layers increases the context size

only depends on pixel above and to the left

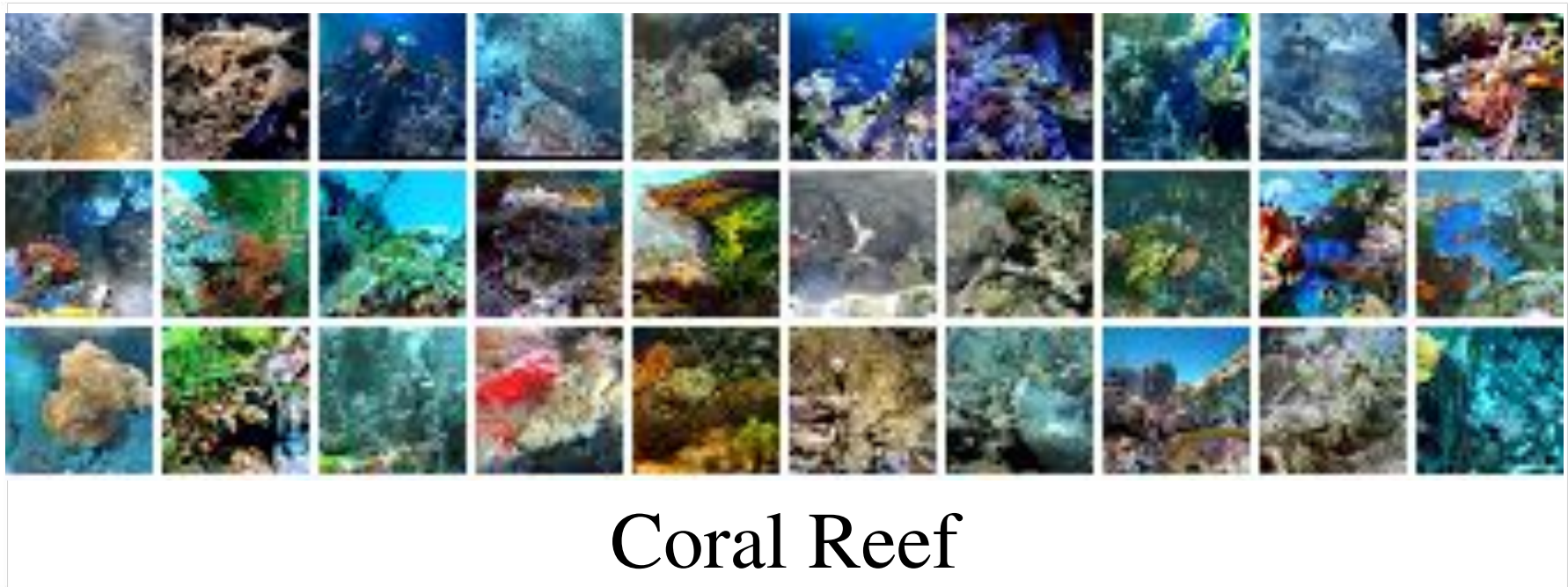
masked convolution

Slide adapted from Aaron Courville

Samples from PixelCNN

Topics: CIFAR-10

- Samples from a class-conditioned PixelCNN

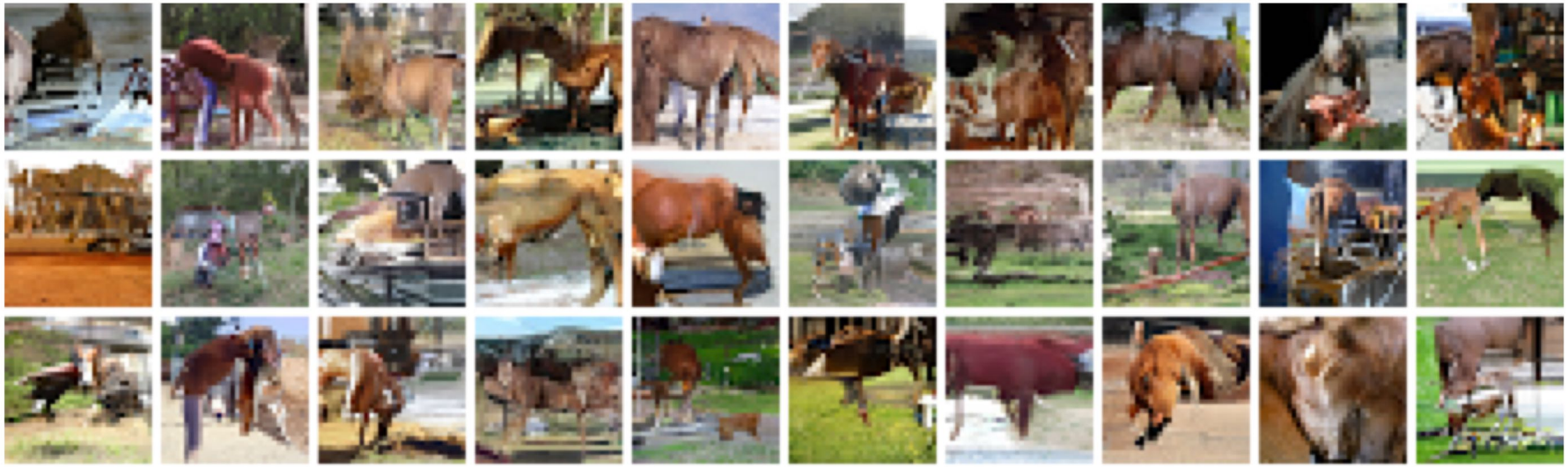


Slide credit: Nal Kalchbrenner

Samples from PixelCNN

Topics: CIFAR-10

- Samples from a class-conditioned PixelCNN



Sorrel horse

Slide credit: Nal Kalchbrenner

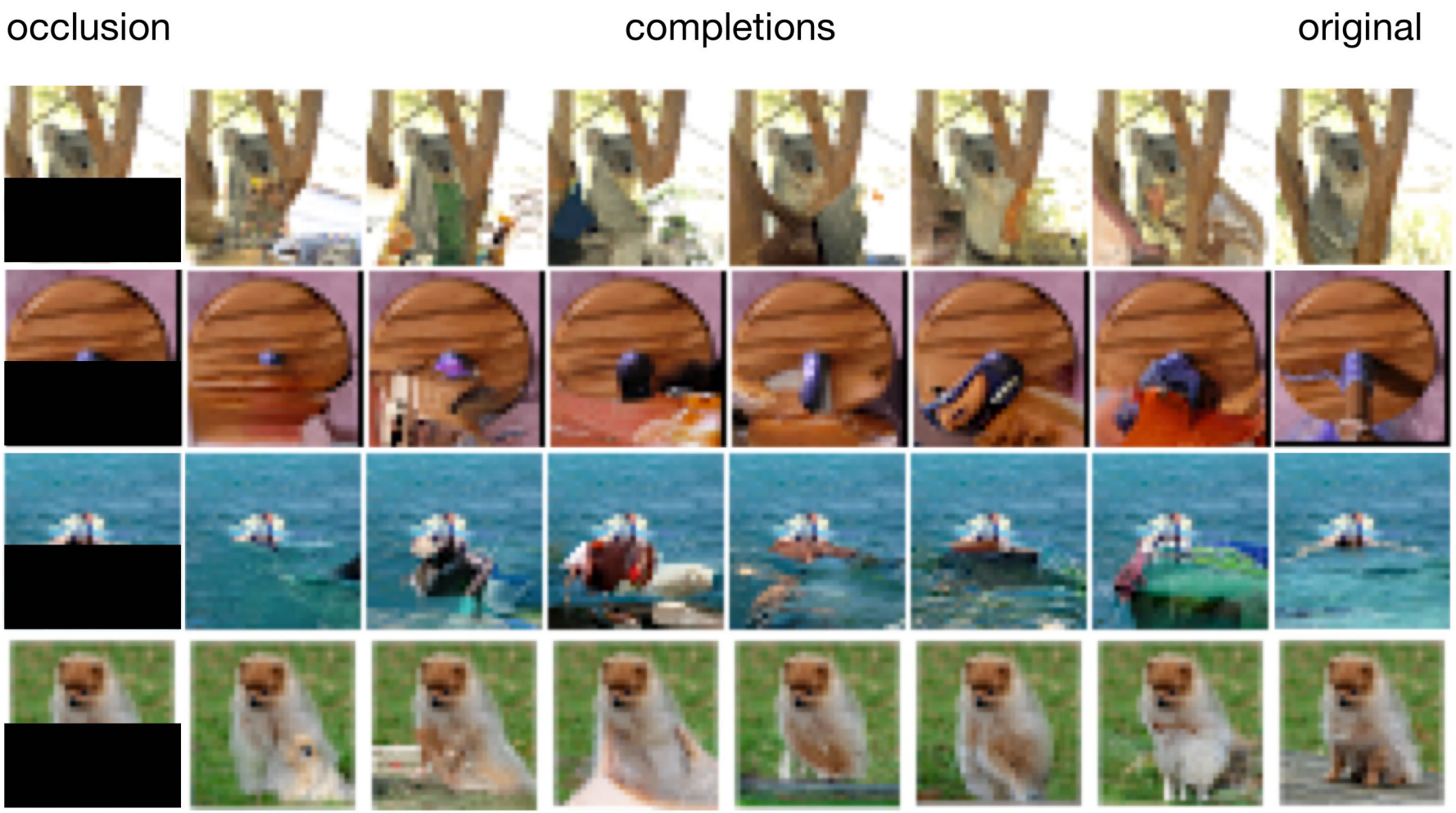
Samples from PixelCNN

Topics: CIFAR-10

- Samples from a class-conditioned PixelCNN

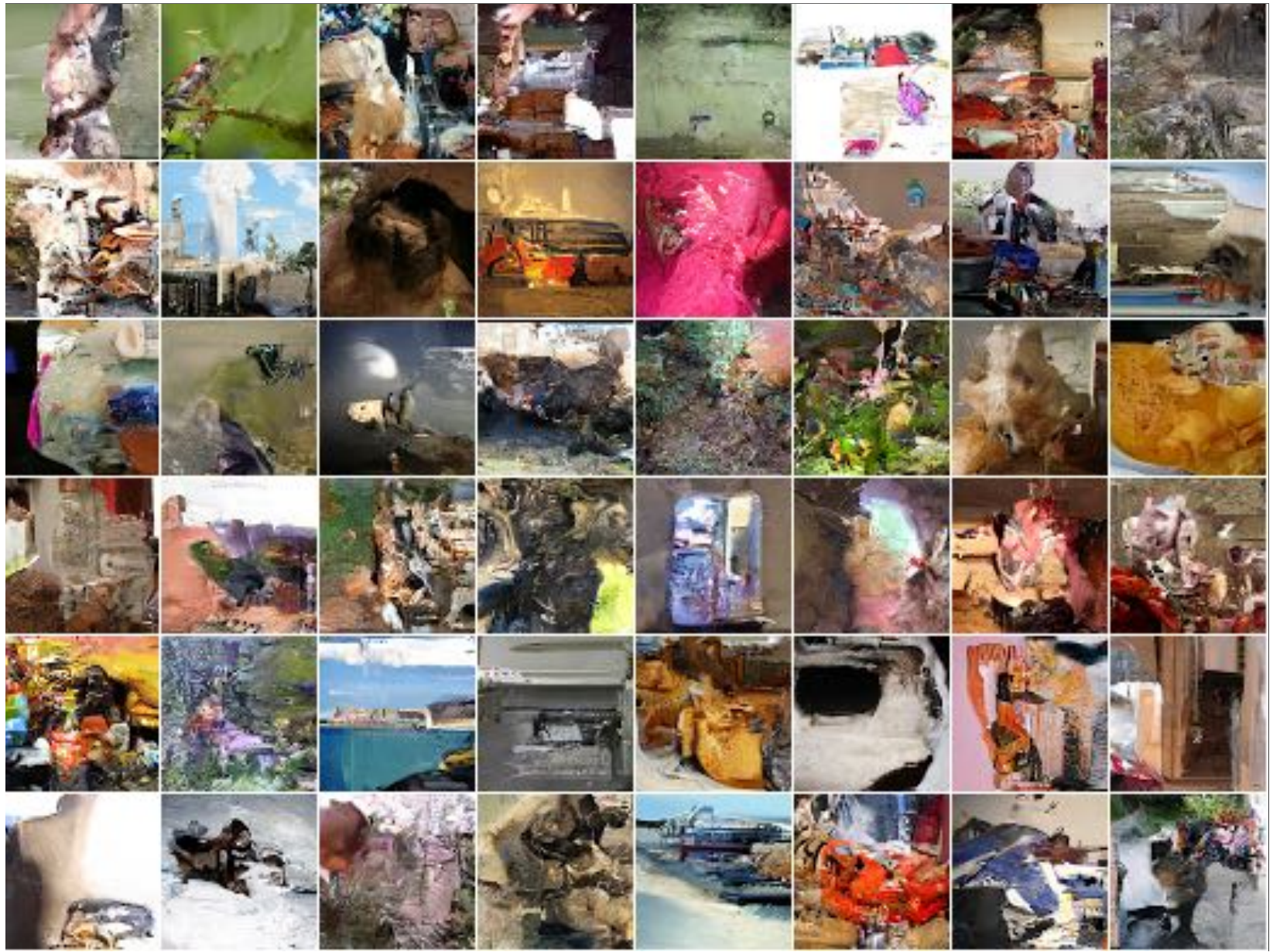
Slide credit: Nal Kalchbrenner

Samples from PixelRNN



Slide credit: Nal Kalchbrenner

Samples from PixelRNN

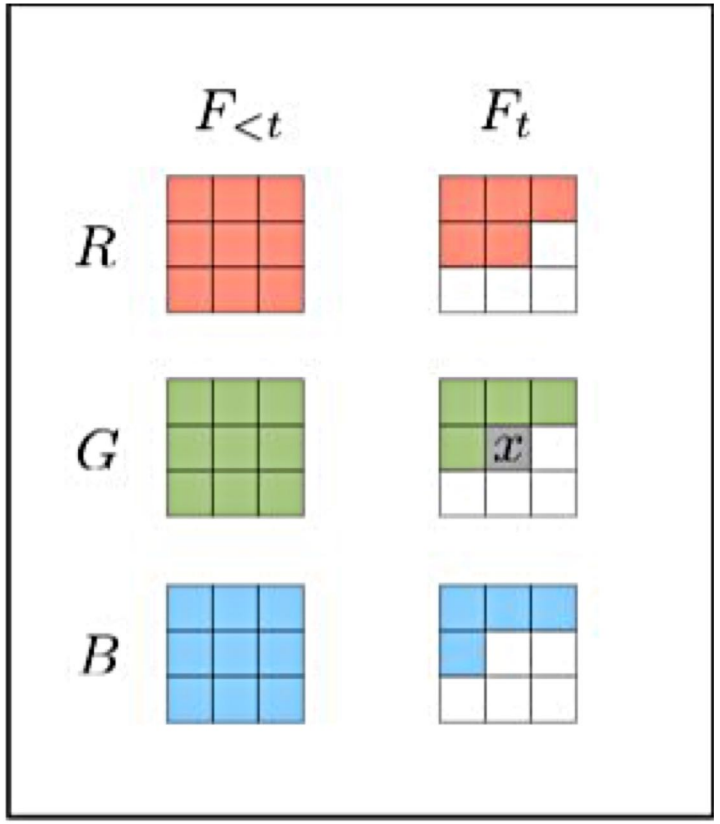


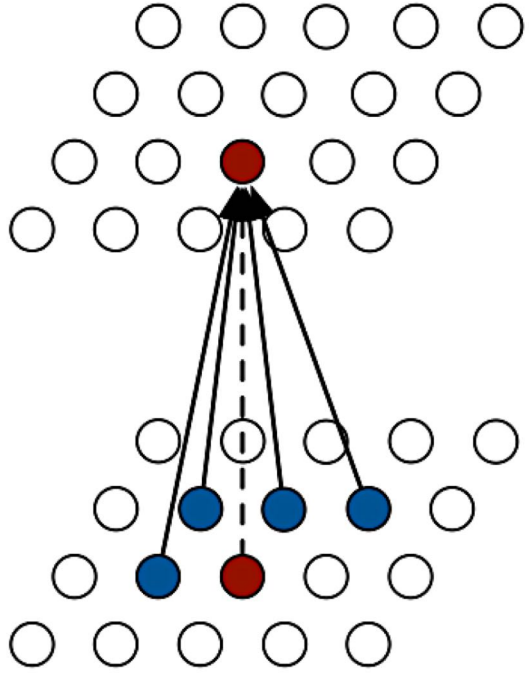
Slide credit:
Nal Kalchbrenner

Samples from PixelRNN

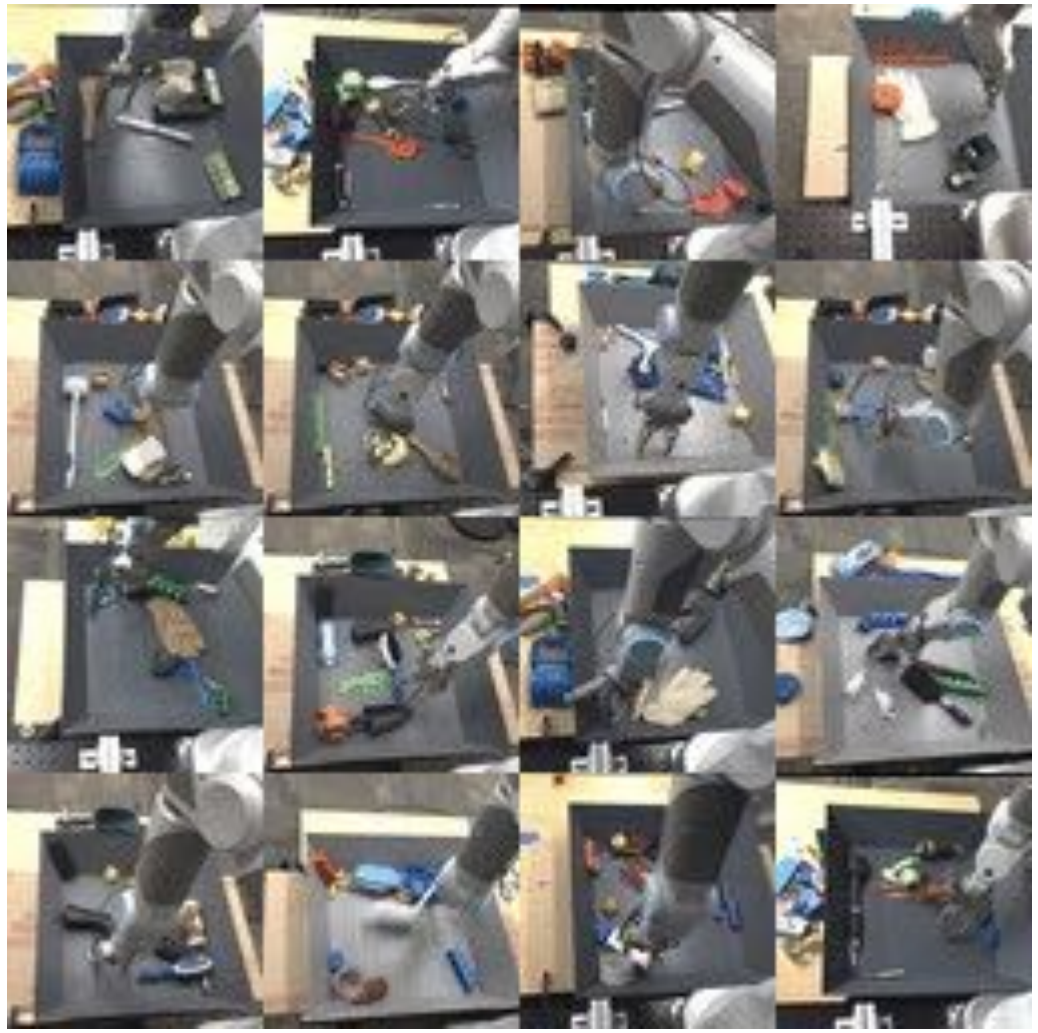
Slide credit: Nal Kalchbrenner

Video Pixel Net (VPN)



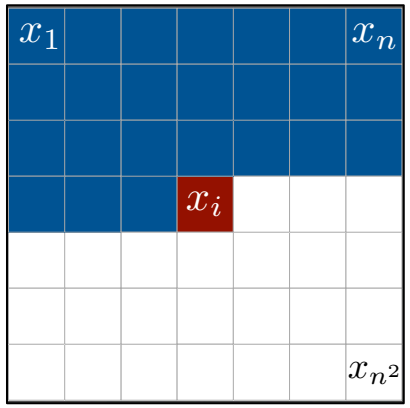


masked convolution



VPN Samples for Robotic Pushing

Autoregressive Models



- Explicitly model conditional probabilities:

$$p_{\text{model}}(\boldsymbol{x}) = p_{\text{model}}(x_1) \prod_{i=2}^{n} p_{\text{model}}(x_i \mid x_1, \ldots, x_{i-1})$$

Each conditional can be a complicated neural net

Advantages:

- $p_{\text{model}}(x)$ is tractable (easy to train and sample)

Disadvantages:

- Generation can be too costly
- Generation can not be controlled by a latent code

PixelCNN elephants
(van den Ord et al. 2016)

VAEs

Variational Autoencoders

Latent Variable Models

- Variational Autoencoder (VAE) model [Kingma and Welling, 2014] [Rezende et al., 2014]

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

z: latent variable

x: observed data

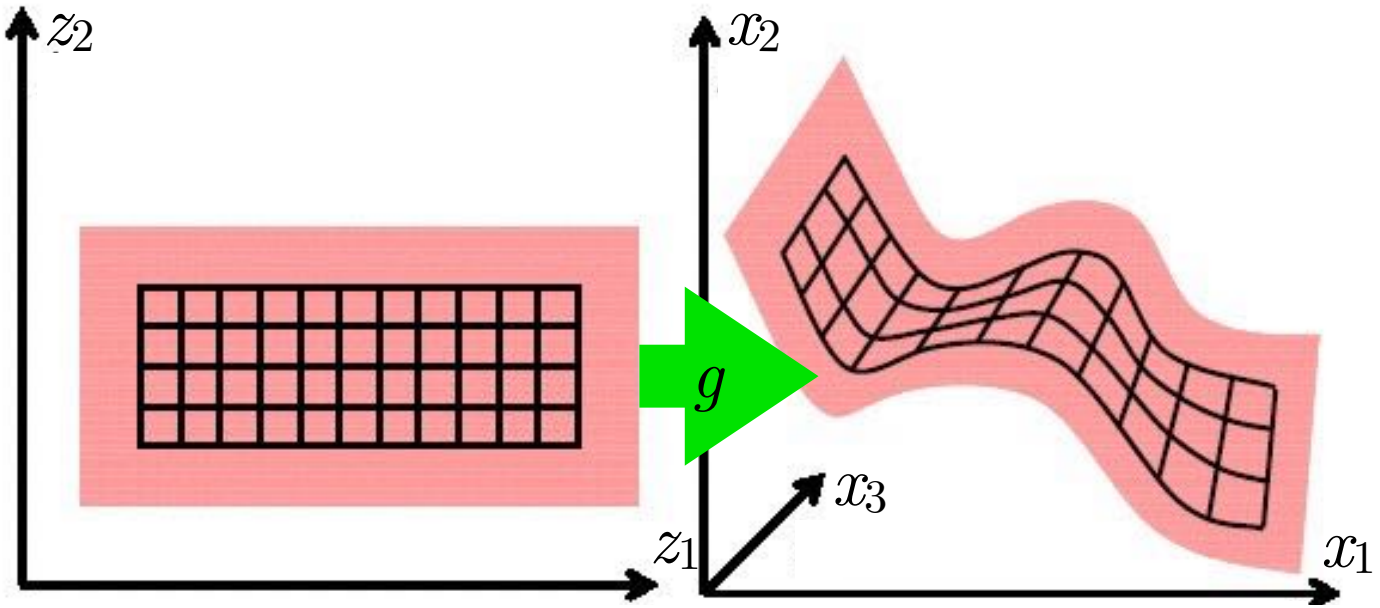
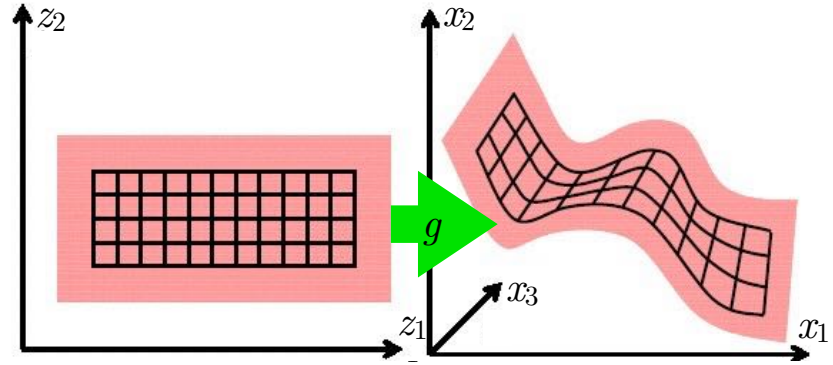


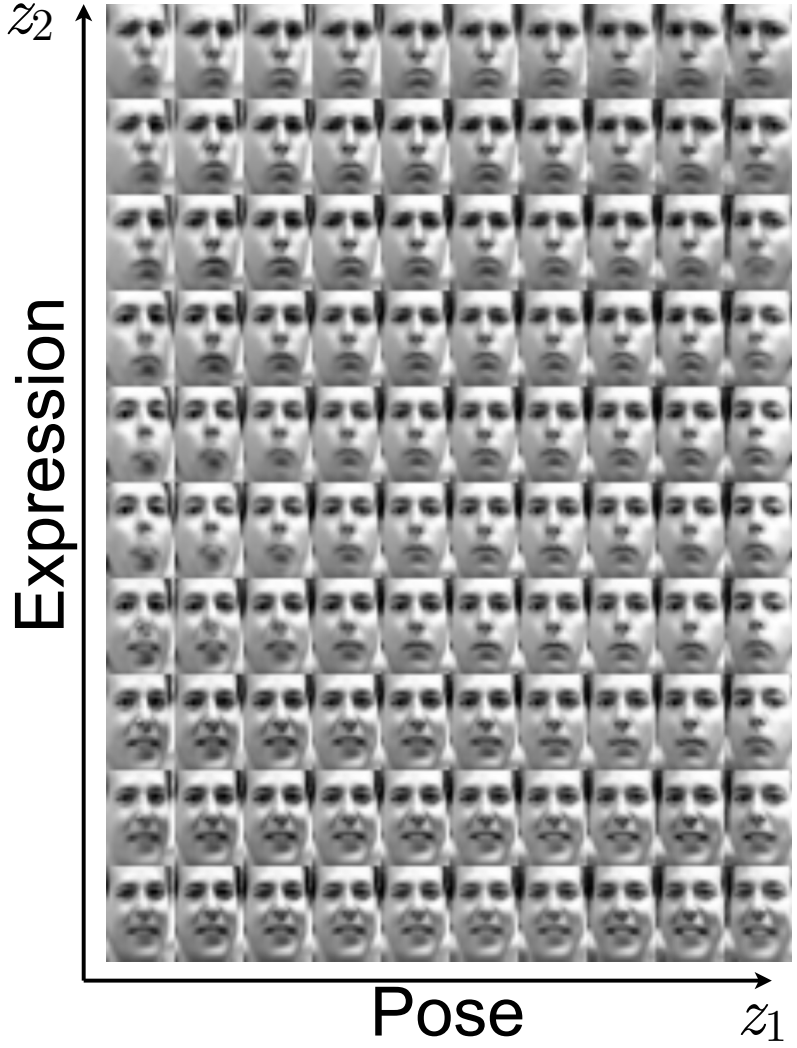
Image credit: Ward and Hamarneh

Slide credit: Aaron Courville

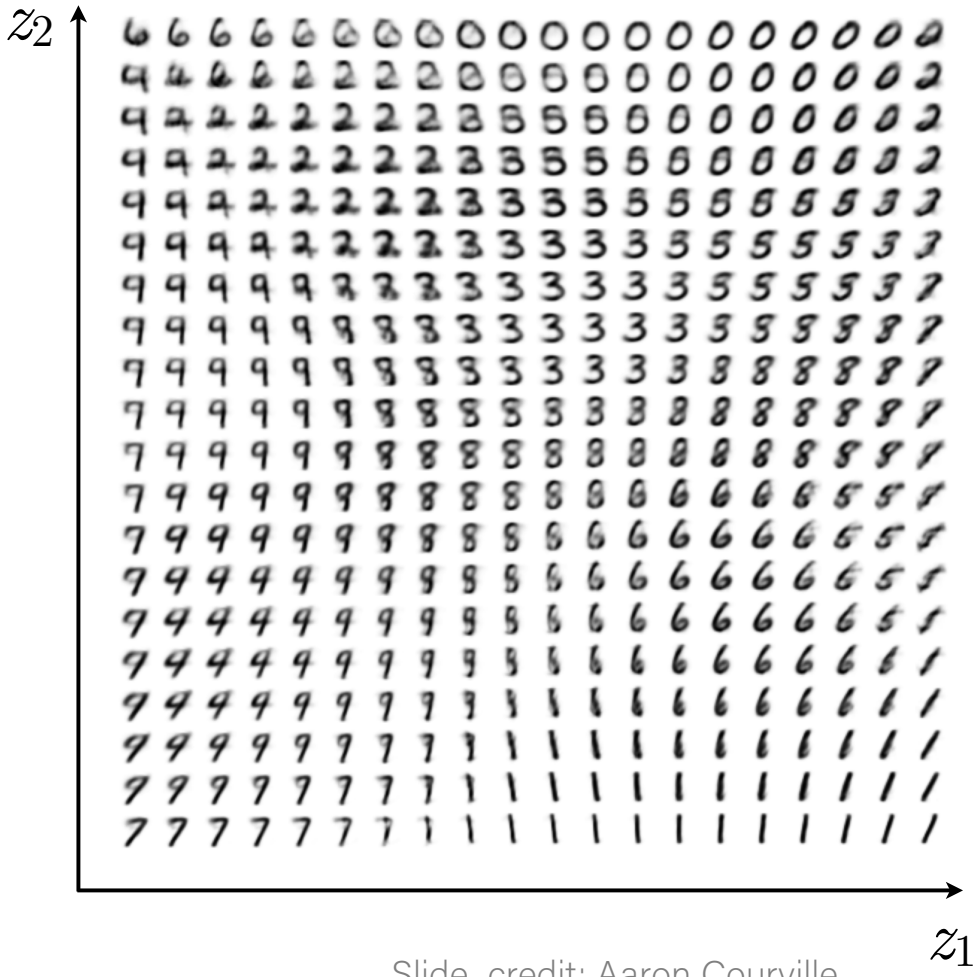
Latent Variable Models



Frey Face dataset:



MNIST:



Slide credit: Aaron Courville

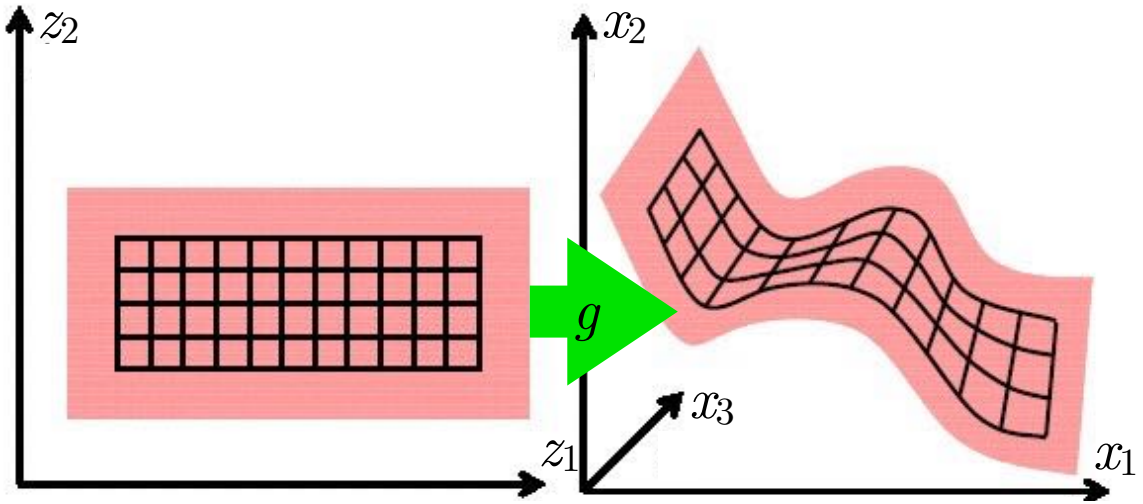
Latent Variable Models

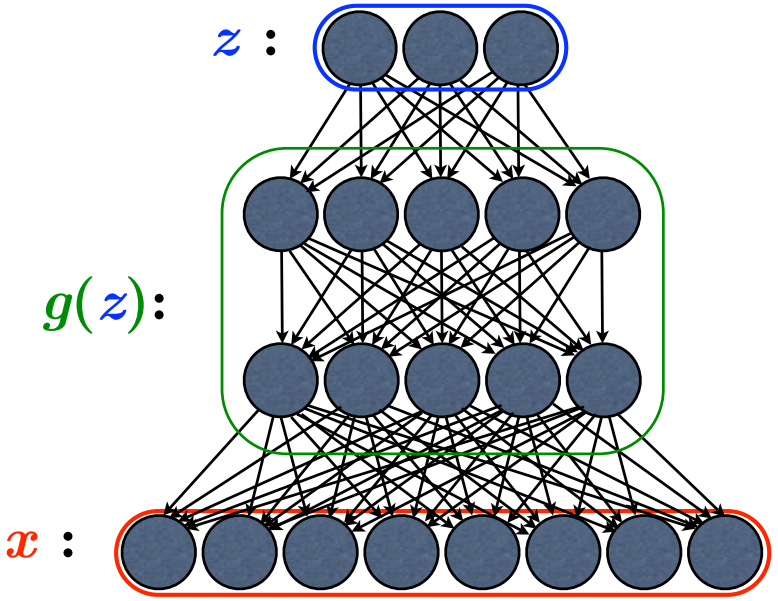
- learn a mapping from some latent variable z to a complicated distribution on x.

$$p(x) = \int p(x, z)\, dz \qquad \text{where} \quad p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x} \mid \boldsymbol{z}) p(\boldsymbol{z})$$

$$p(\boldsymbol{z}) = \text{something simple} \qquad p(\boldsymbol{x} \mid \boldsymbol{z}) = g(\boldsymbol{z})$$

- Can we learn to decouple the true explanatory factors underlying the data distribution?
  e.g. separate identity and expression in face images





Slide credit:
Aaron Courville

Image credit:
Ward and Hamarneh

Variational Autoencoder

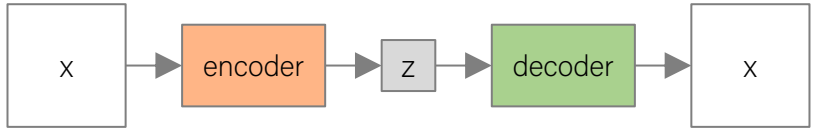
- **Where does z come from?** — The classic directed model dilemma.
- The VAE approach: introduce an inference machine $q_\phi(z \mid x)$ that learns to approximate the posterior $p_\theta(z \mid x)$.
  - Define a variational lower bound on the data likelihood: $p_\theta(x) \geq \mathcal{L}(\theta, \phi, x)$

$$\begin{aligned}\mathcal{L}(\theta, \phi, x) &= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x, z) - \log q_\phi(z \mid x)\right] \\ &= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z) + \log p_\theta(z) - \log q_\phi(z \mid x)\right] \\ &= -D_{\mathrm{KL}}\left(q_\phi(z \mid x) \| p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right]\end{aligned}$$

***regularization* term** ***reconstruction* term**

- What is $q_\phi(z \mid x)$?

VAE Inference model



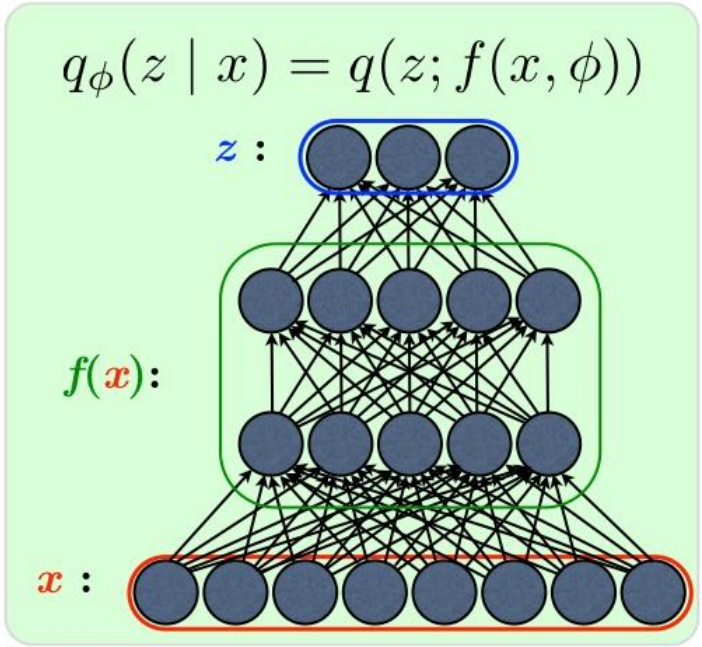
- The VAE approach: introduce an inference model $q_\phi(z \mid x)$ that learns to approximate the intractable posterior $p_\theta(z \mid x)$ by optimizing the variational lower bound:

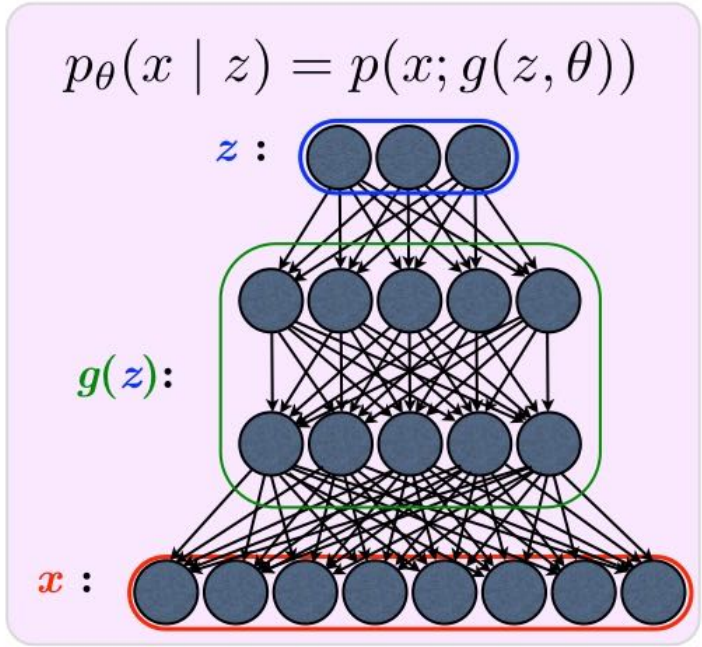
$$\mathcal{L}(\theta, \phi, x) = -D_{\mathrm{KL}}\left(q_\phi(z \mid x) \| p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right]$$

- We parameterize $q_\phi(z \mid x)$ with another neural network:

encoder network

inference model



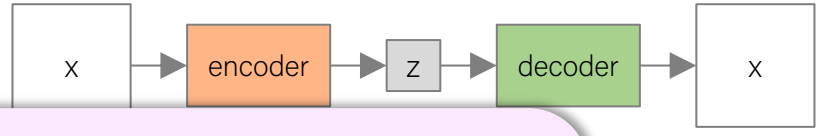


decoder network

generative model

Slide credit: Aaron Courville

VAE Inference model



- The VAE approach: introd $q_\phi(z \mid$ ximate the intractable posterior p ne vari

Make approximate posterior distribution close to prior

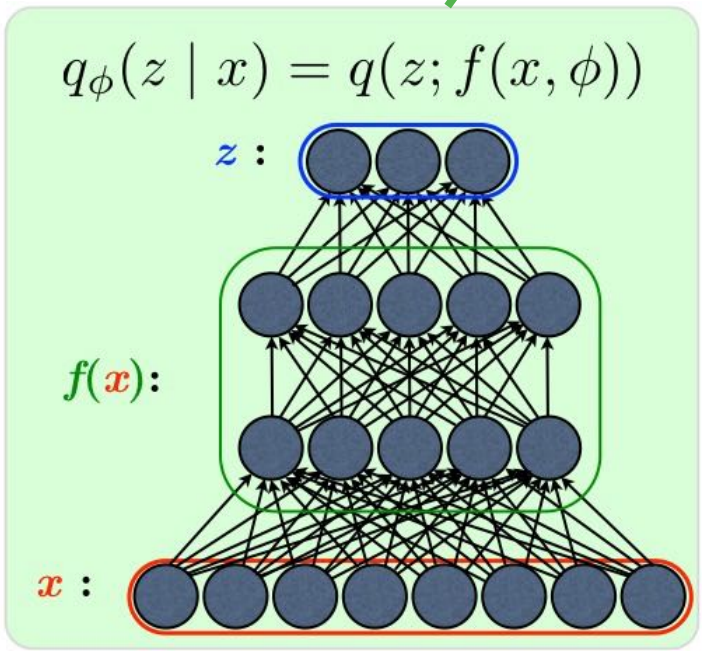
Maximize likelihood of original input being reconstructed

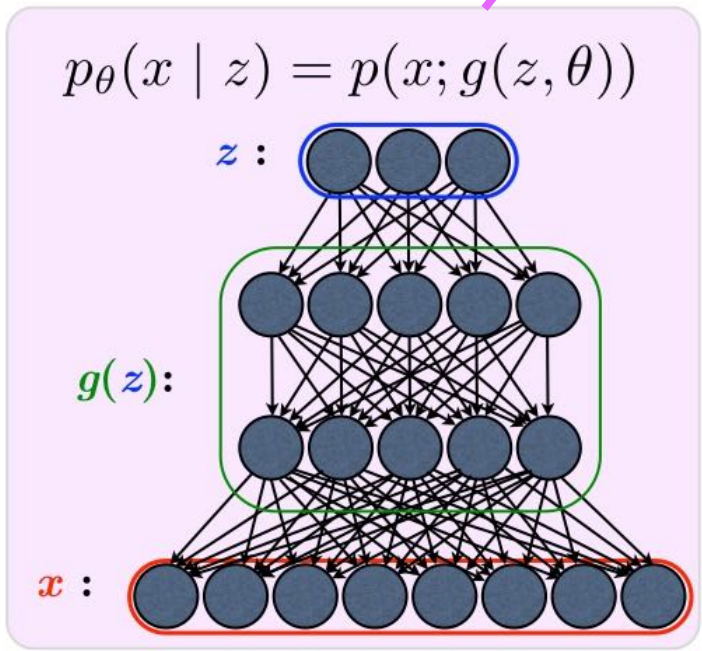
$$\mathcal{L}(\theta, \phi, x) = -D_{\mathrm{KL}}\left(q_\phi(z \mid x) \| p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right]$$

- We parameterize $q_\phi(z \mid x)$ with another neural network:

encoder network

inference model





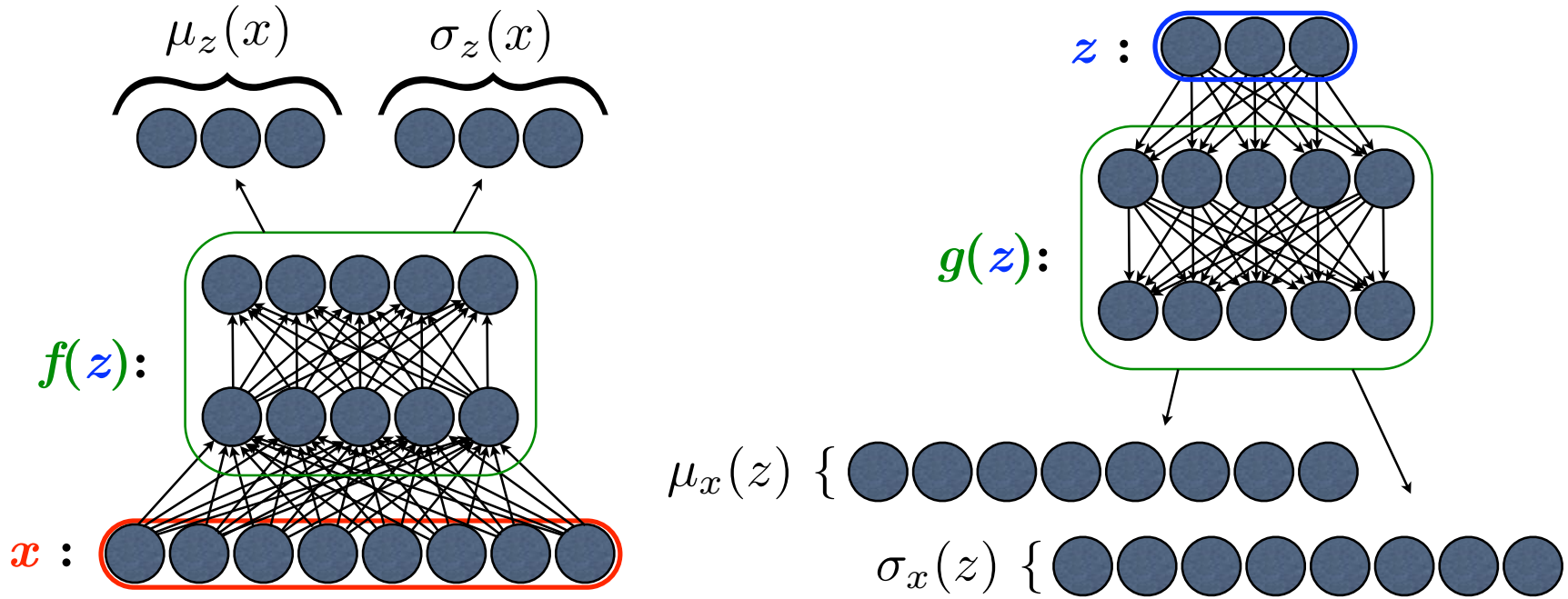
decoder network

generative model

Slide credit: Aaron Courville

Reparametrization trick

- Adding a few details + one really important trick
- Let's consider z to be real and $q_\phi(z \mid x) = \mathcal{N}(z; \mu_z(x), \sigma_z(x))$
- Parametrize z as $z = \mu_z(x) + \sigma_z(x)\epsilon_z$ where $\epsilon_z = \mathcal{N}(0, 1)$
- (optional) Parametrize x as $x = \mu_x(z) + \sigma_x(z)\epsilon_x$ where $\epsilon_x = \mathcal{N}(0, 1)$



Reparametrization trick

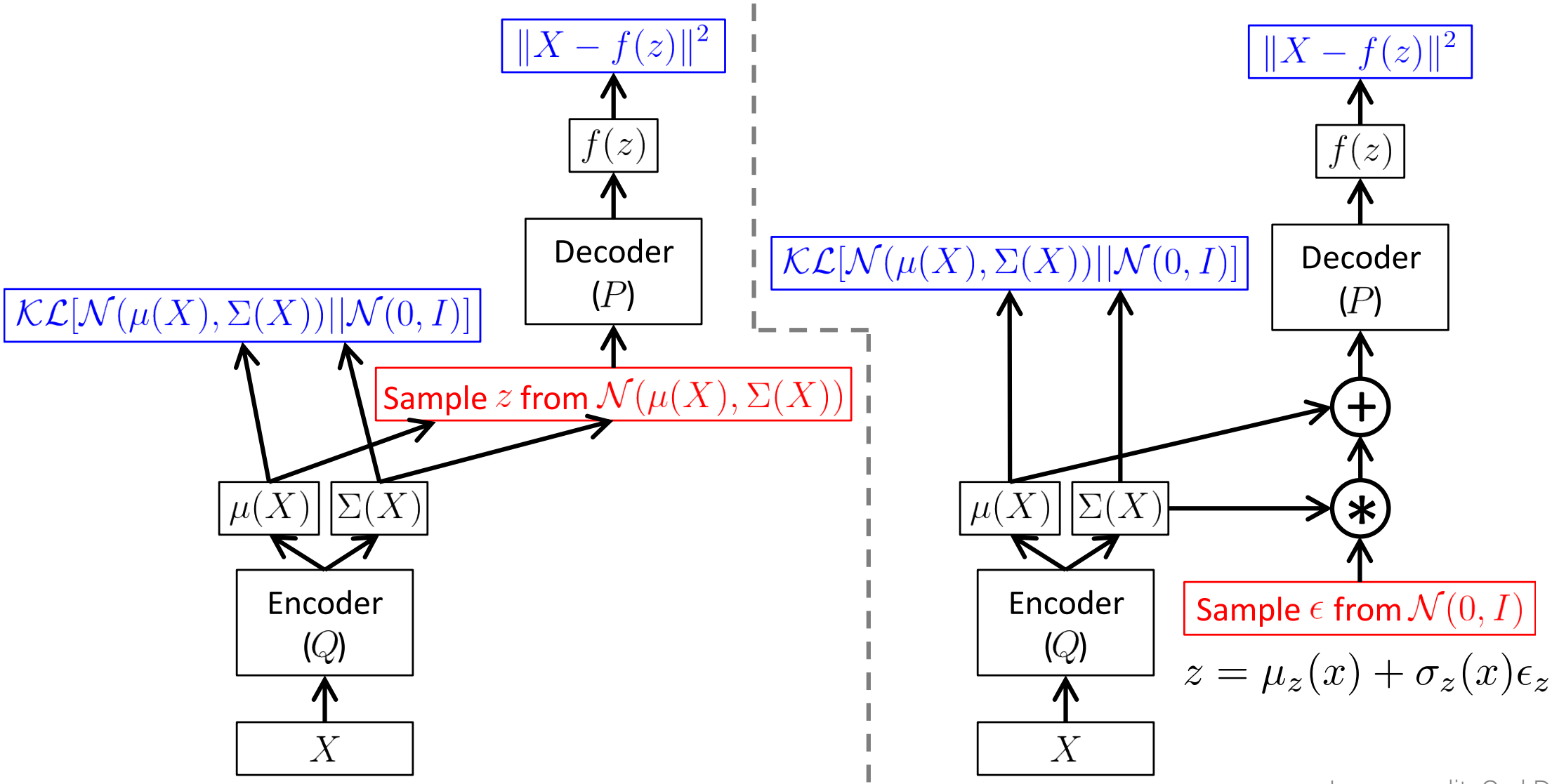
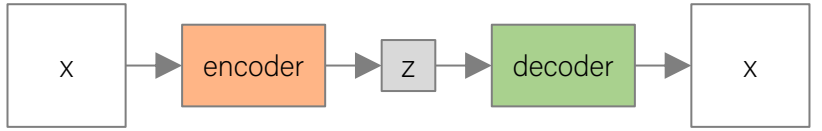
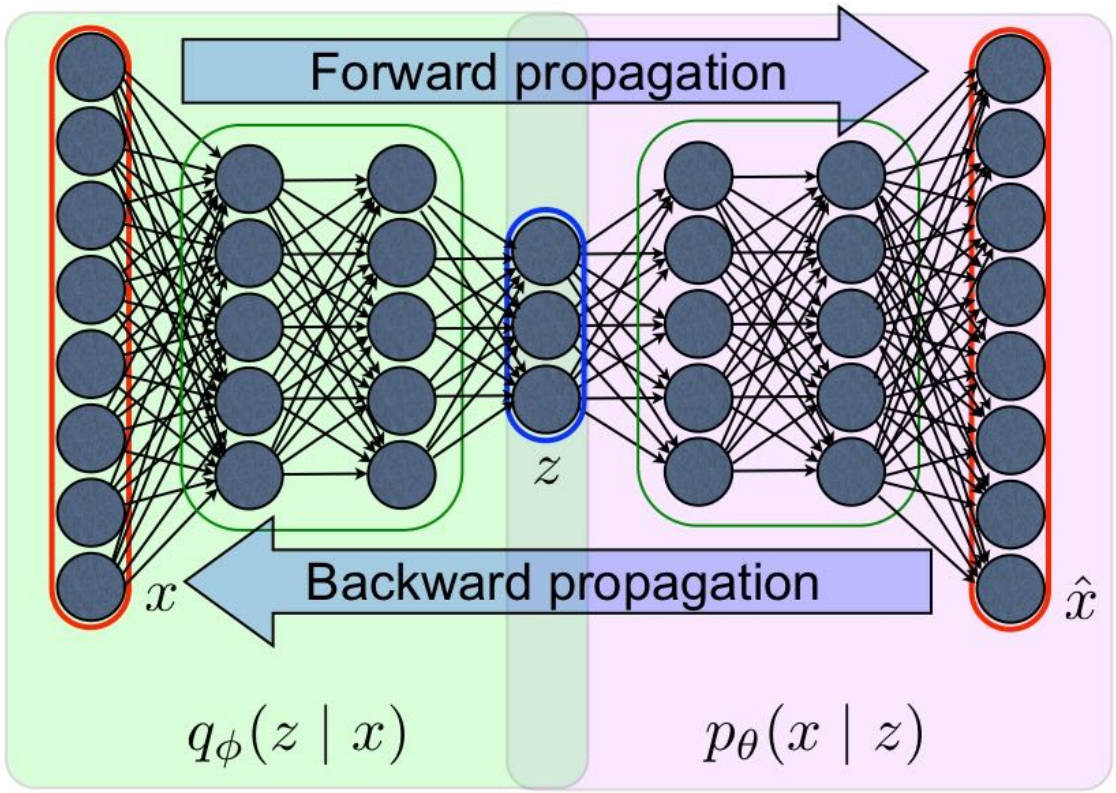


Image credit: Carl Doersch

Training with backpropagation



- Due to a reparametrization trick, we can simultaneously train both the generative model $p_\theta(z \mid x)$ and the inference model $q_\phi(z \mid x)$ by optimizing the variational bound using gradient backpropagation.
- Objective function: $\mathcal{L}(\theta, \phi, x) = -D_{\mathrm{KL}}\left(q_\phi(z \mid x) \| p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right]$



Slide credit: Aaron Courville

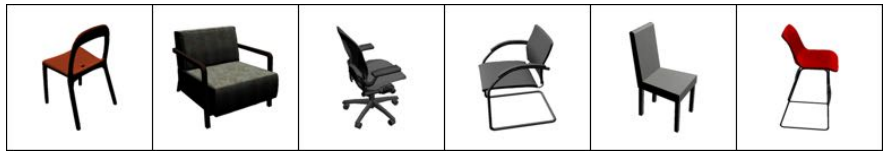
Vanilla VAE samples

Labelled Faces in the Wild (LFW)

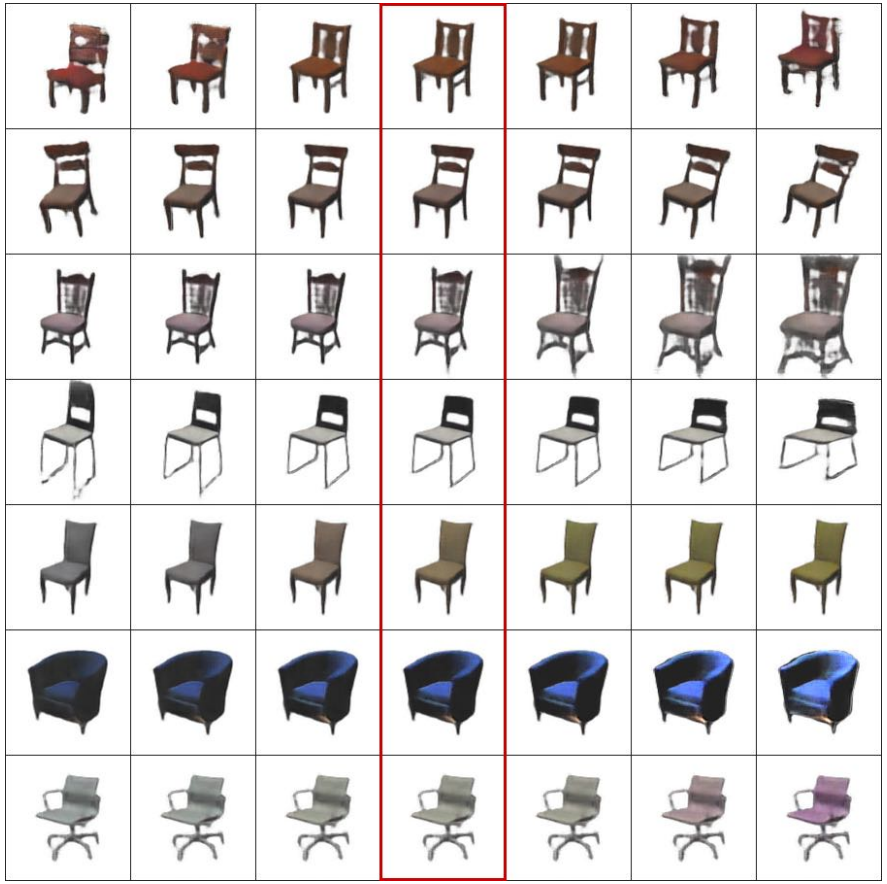
ImageNet (small)

Slide credit: Aaron Courville

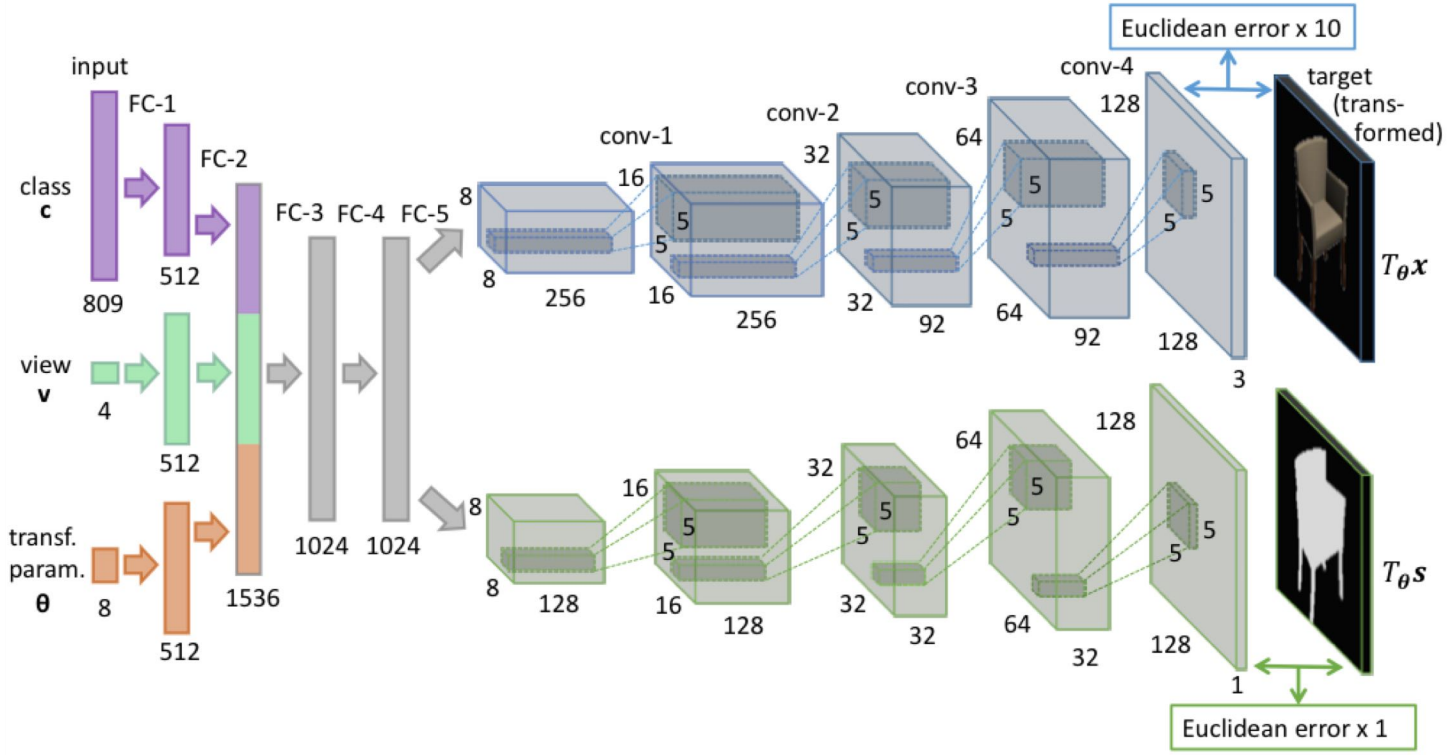
Learning To Generate 3D Models(Dosovitskiyet al., 2015)



Several representative chair images used for training the network.



Generation of chair images while activating various transformations. Each row shows one transformation: translation, rotation, zoom, stretch, saturation, brightness, color. The middle column shows the reconstruction without any transformation.



- 4 convolutional neural nets
  - one for foreground and the other for background for both recognition and generation networks

Attribute-driven Image Generation (Yan et al., 2016)

Attributes

Male, No eyewear, Frowning, Receding hairline, Bushy eyebrow, Eyes open, Pointy nose, Teeth not visible, Rosy cheeks, Flushed face

- 4 convolutional neural nets
  - one for foreground and the other for background for both recognition and generation networks

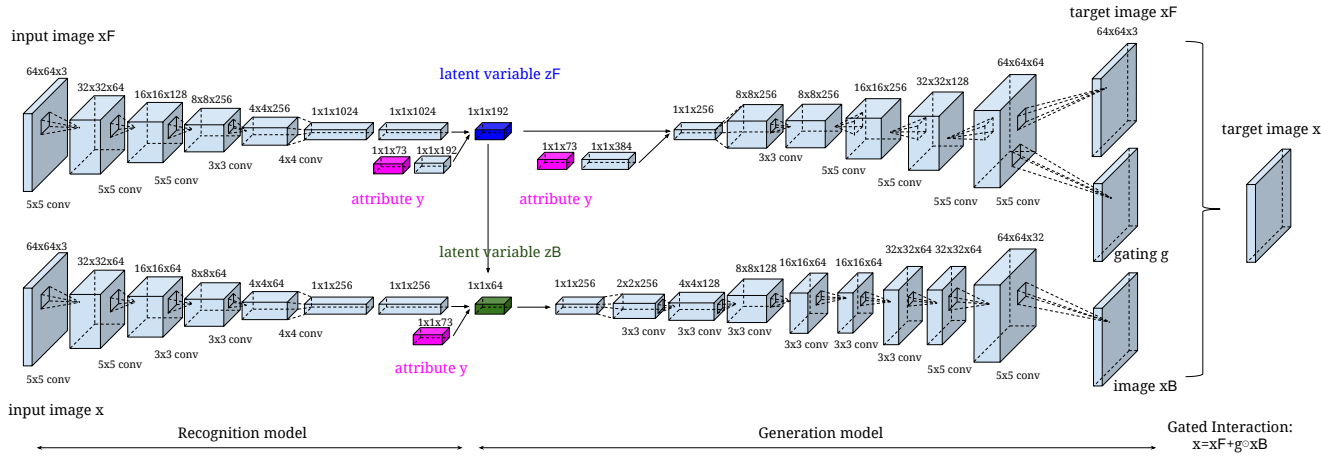
Nearest Neighbor

Vanilla CVAE

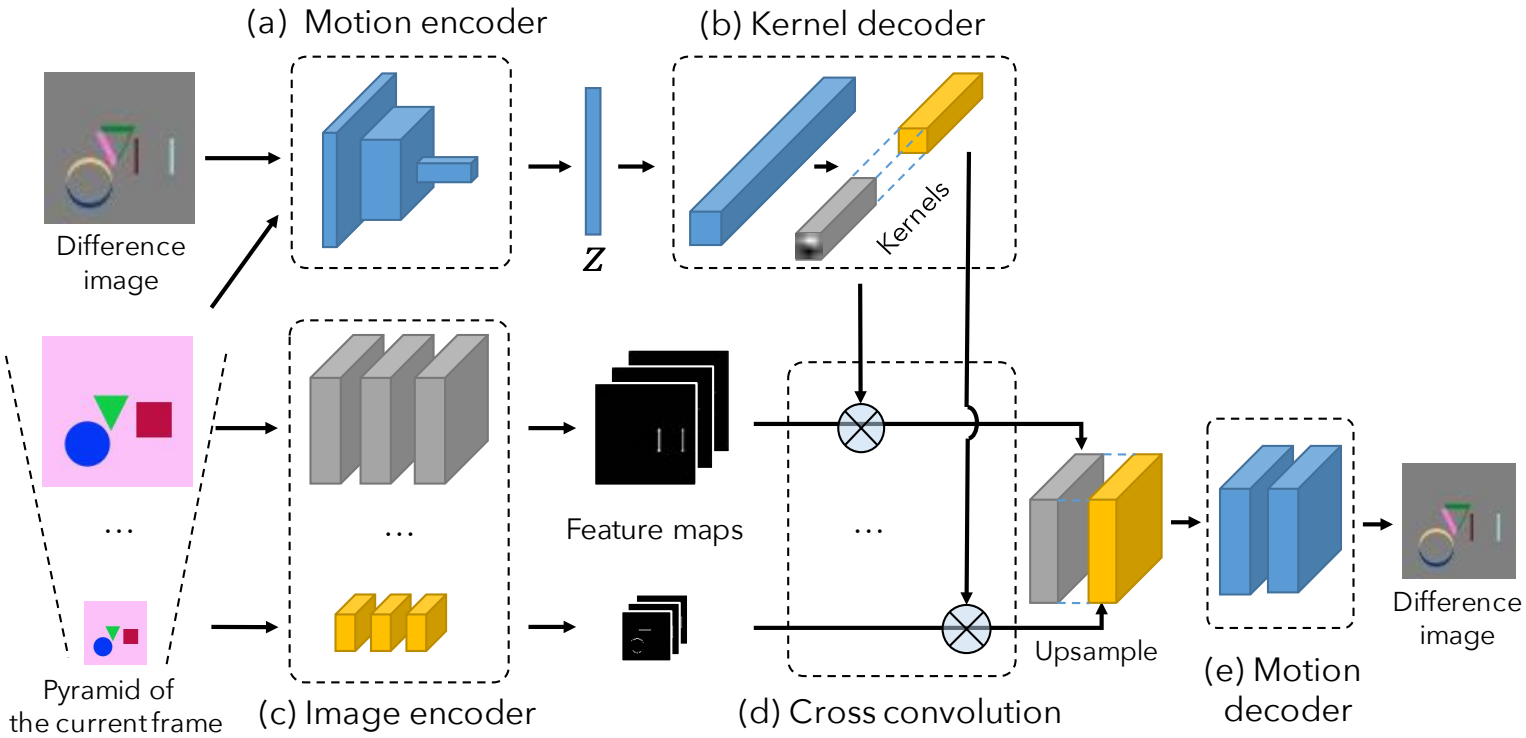
disCVAE (foreground)

disCVAE (full)

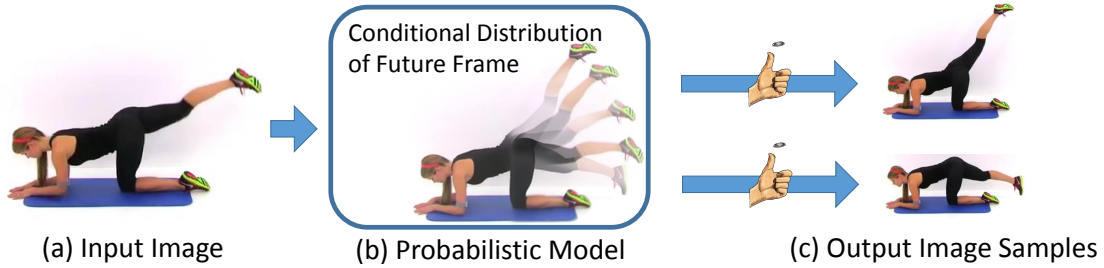
Reference



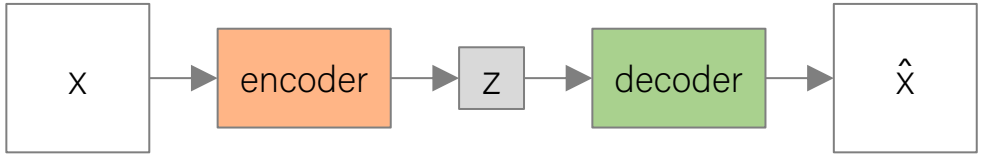
Future Frame Prediction (Xue et al., 2016)



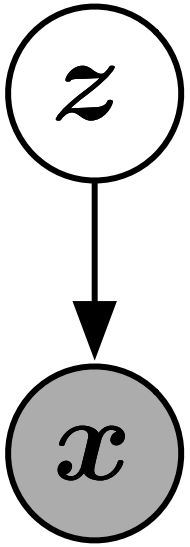
- Conditional VAE
- 5 components
  - motion encoder
  - kernel decoder
  - image encoder
  - cross convolution layer
  - a motion decoder.



Recap: VAEs



- Maximizes a variational lower bound on log-likelihood of $\boldsymbol{x}$



$$\log p(\boldsymbol{x}) \geq \log p(\boldsymbol{x}) - D_{\mathrm{KL}}\left(q(\boldsymbol{z}) \| p(\boldsymbol{z} \mid \boldsymbol{x})\right)$$
$$= \mathbb{E}_{\boldsymbol{z} \sim q} \log p(\boldsymbol{x}, \boldsymbol{z}) + H(q)$$

Face samples for
Labeled Faces in the Wild (LFW)
(Alec Radford)

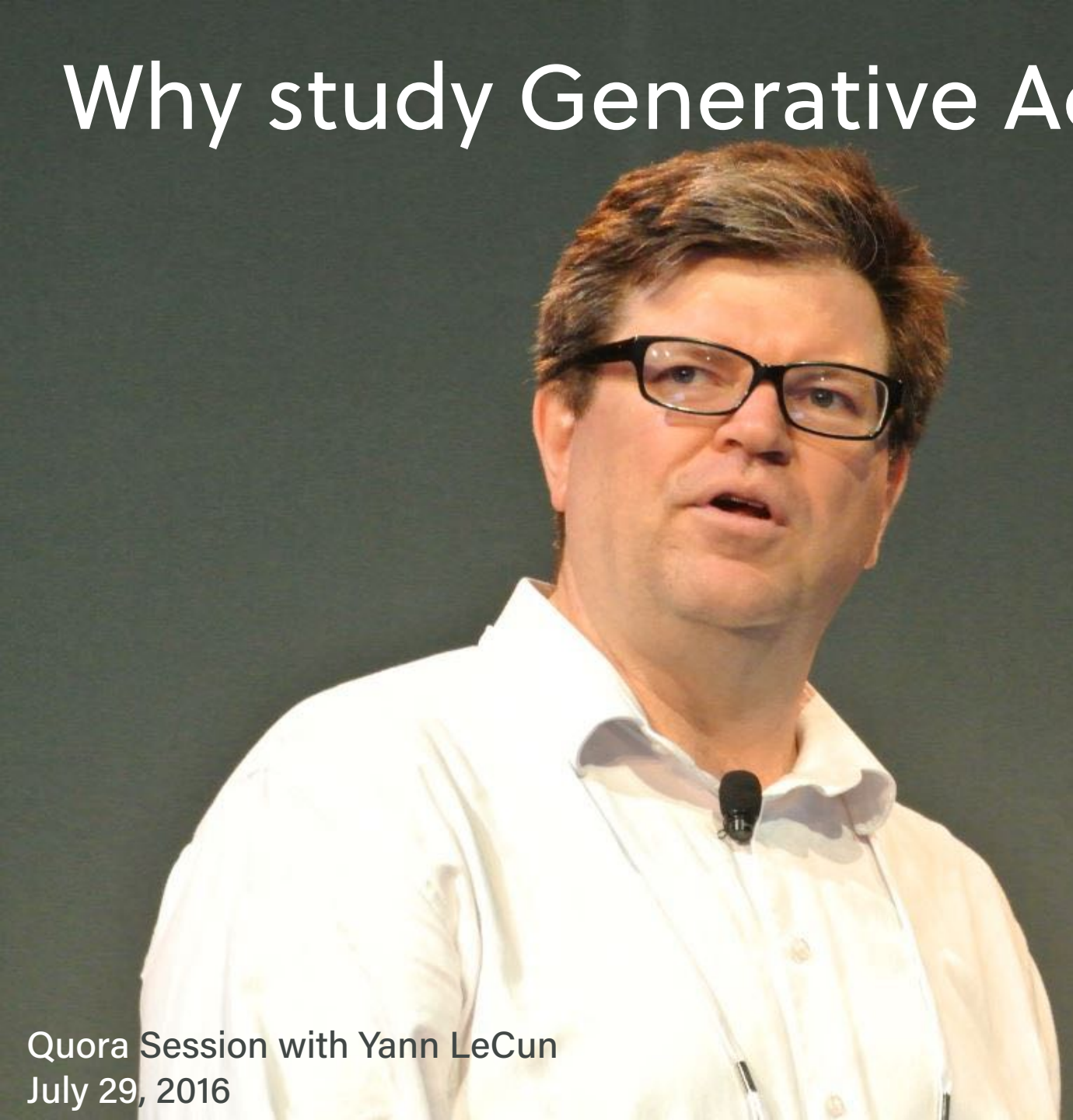
Disadvantages:

- Not asymptotically consistent unless q is perfect
- Tends to produce blurry samples

GANs

Generative
Adversarial
Networks

Why study Generative Adversarial Networks?

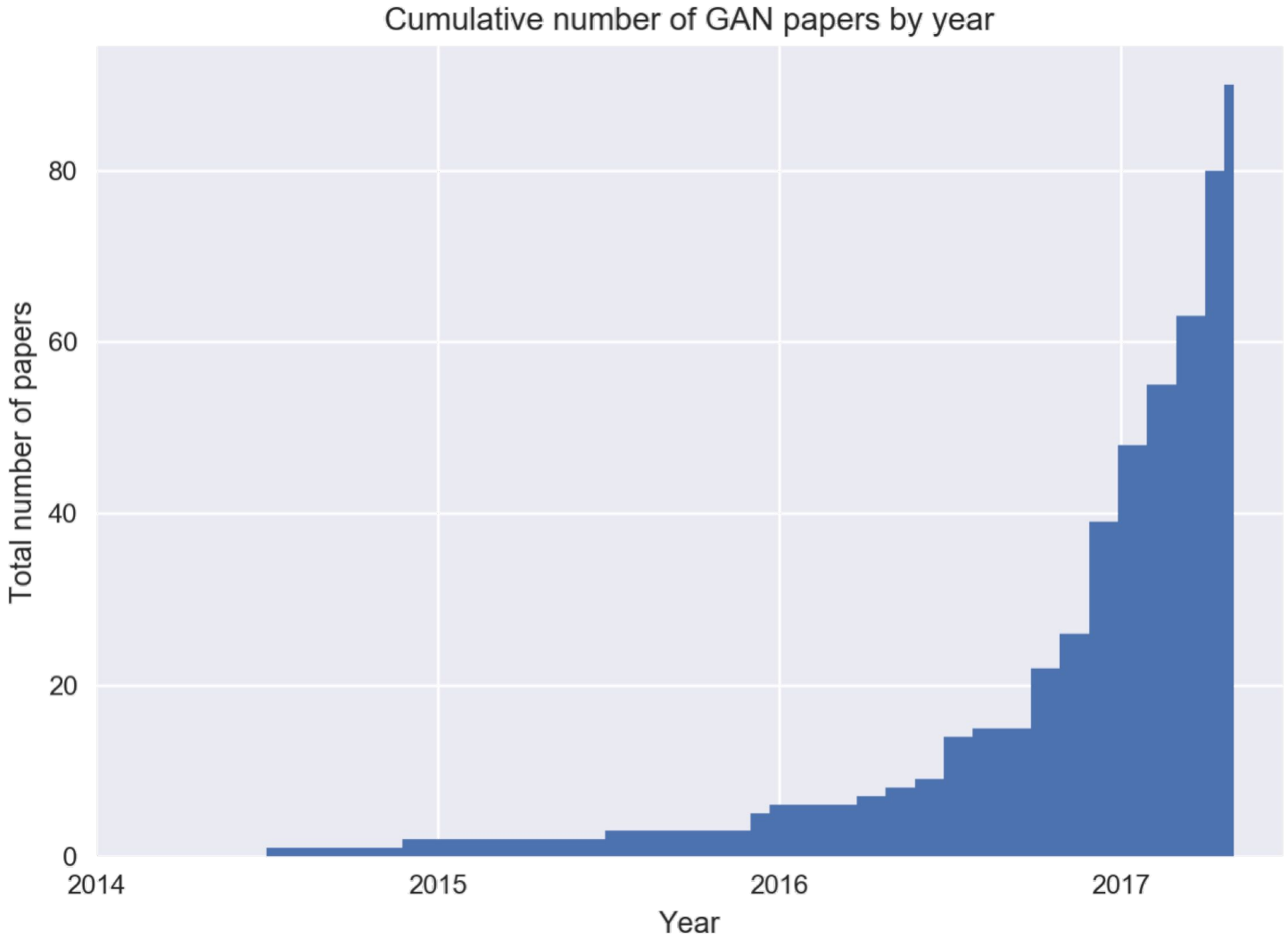


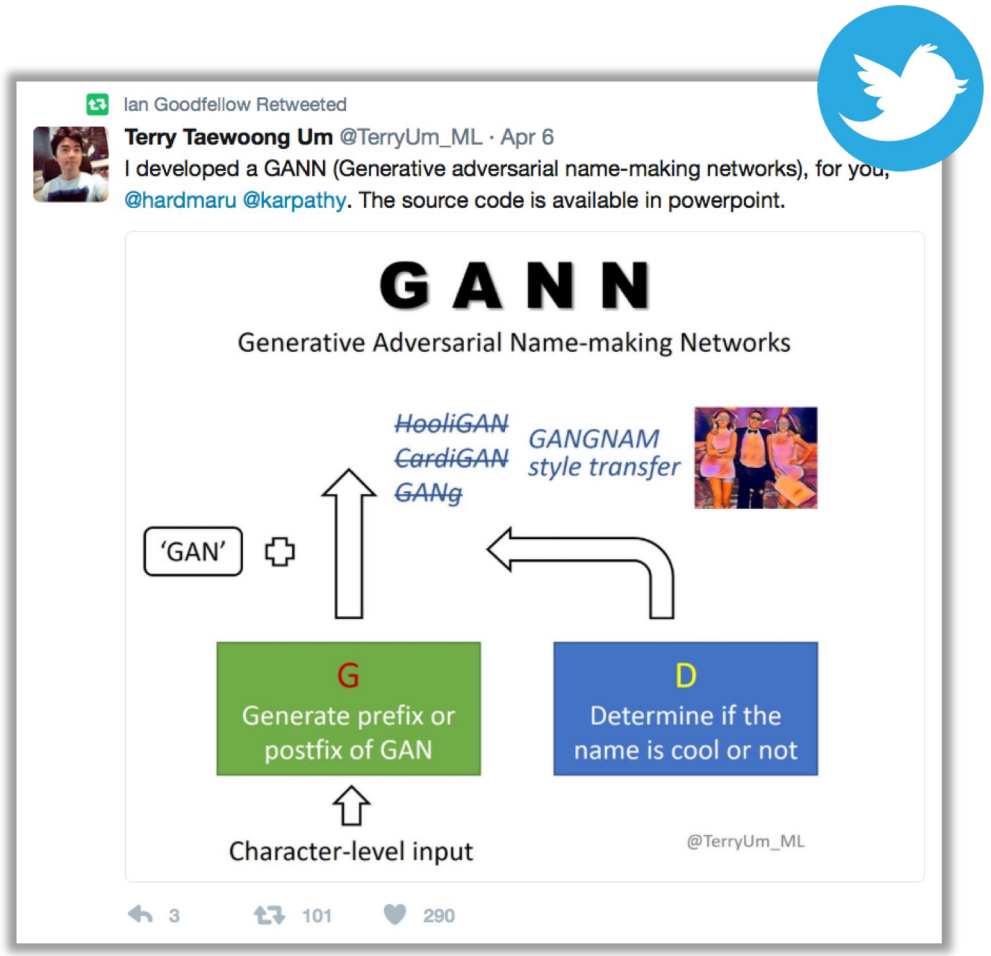
Q: What are some recent and potentially upcoming breakthroughs in deep learning?

A: The most important one, in my opinion, is adversarial training (also called GAN for Generative Adversarial Networks) … This, and the variations that are now being proposed is **the most interesting idea in the last 10 years in ML**, in my opinion.

Quora Session with Yann LeCun
July 29, 2016

Progress in GANs





Source: https://deephunt.in/the-gan-zoo-79597dc8c347

Generative Adversarial Networks (GANs)

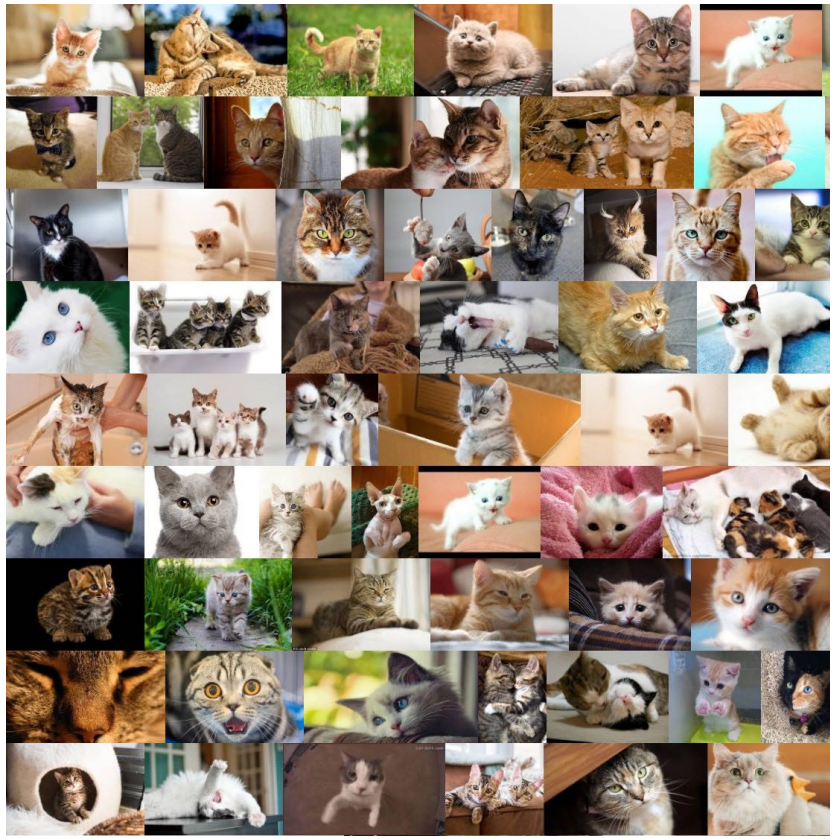
(Goodfellow et al., 2014)

Noise
(random input)

Generative
Model

$z \sim \text{Uniform}_{100}$

think of this as a transformation



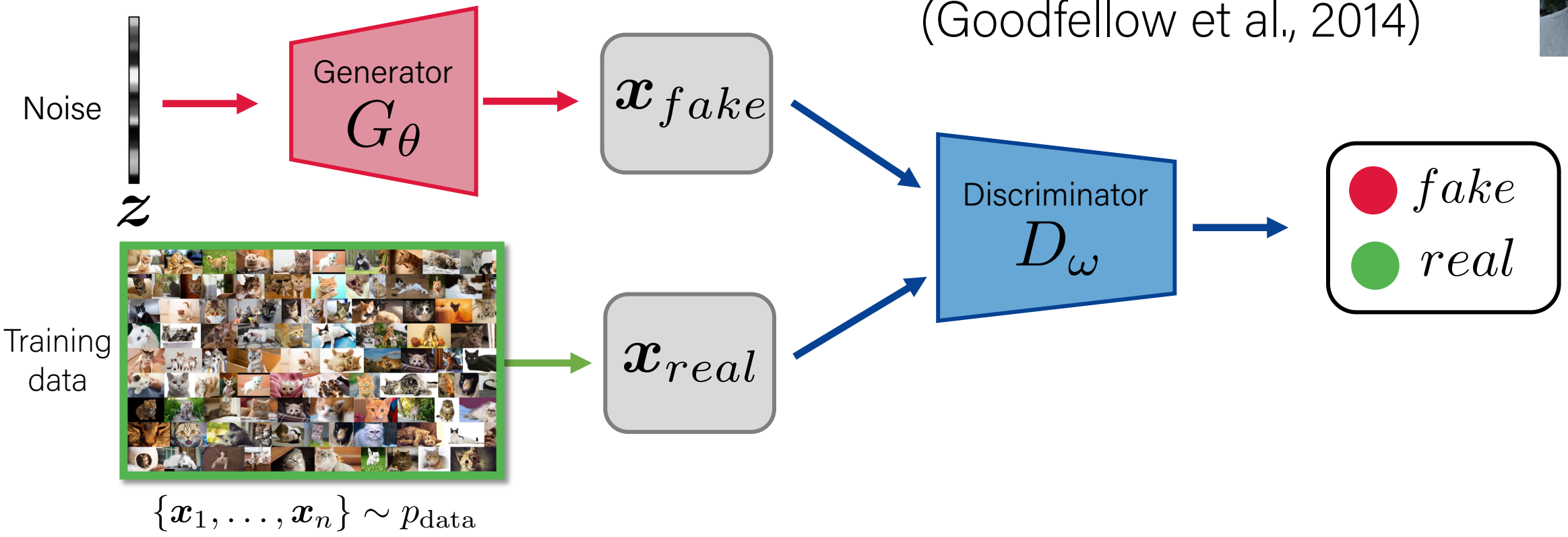
- A game-theoretic likelihood free model

Advantages:

- Uses a latent code
- No Markov chains needed
- Produces the best samples

Generative Adversarial Networks (GANs)

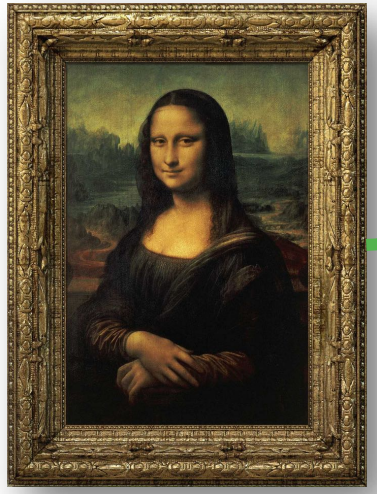
(Goodfellow et al., 2014)



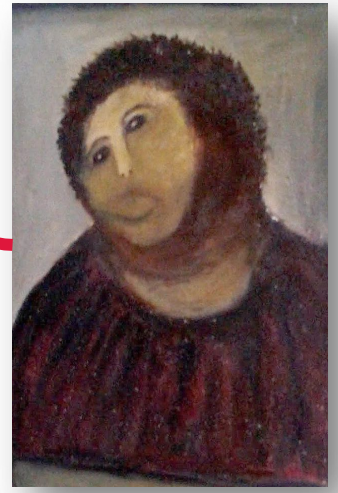
- A game between a generator $G_\theta(z)$ and a discriminator $D_\omega(x)$
  - Generator tries to fool discriminator (i.e. generate realistic samples)
  - Discriminator tries to distinguish fake from real samples

Intuition behind GANs

D_ω: Discriminator (*Art Critic*)



$\boldsymbol{x}_{real}$



$\boldsymbol{x}_{fake}$

G_θ: Generator (*Forger*)

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)

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

Real data

Noise vector used to

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

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

ross-entropy
ss for binary
assification

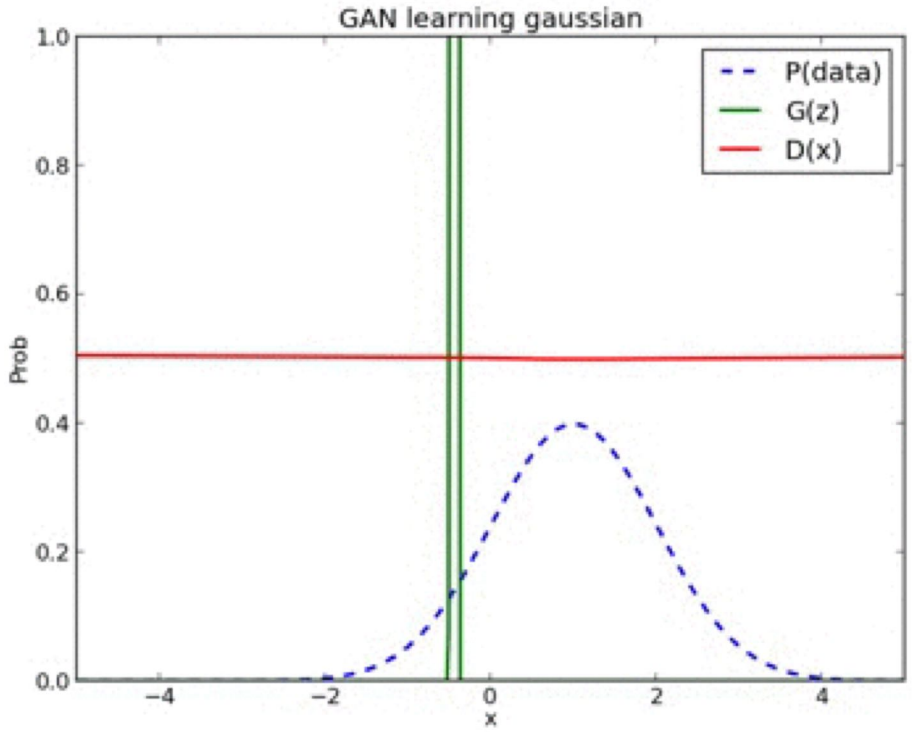
probability
of the discriminator being mistaken

Important question is "Does this converge??"

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

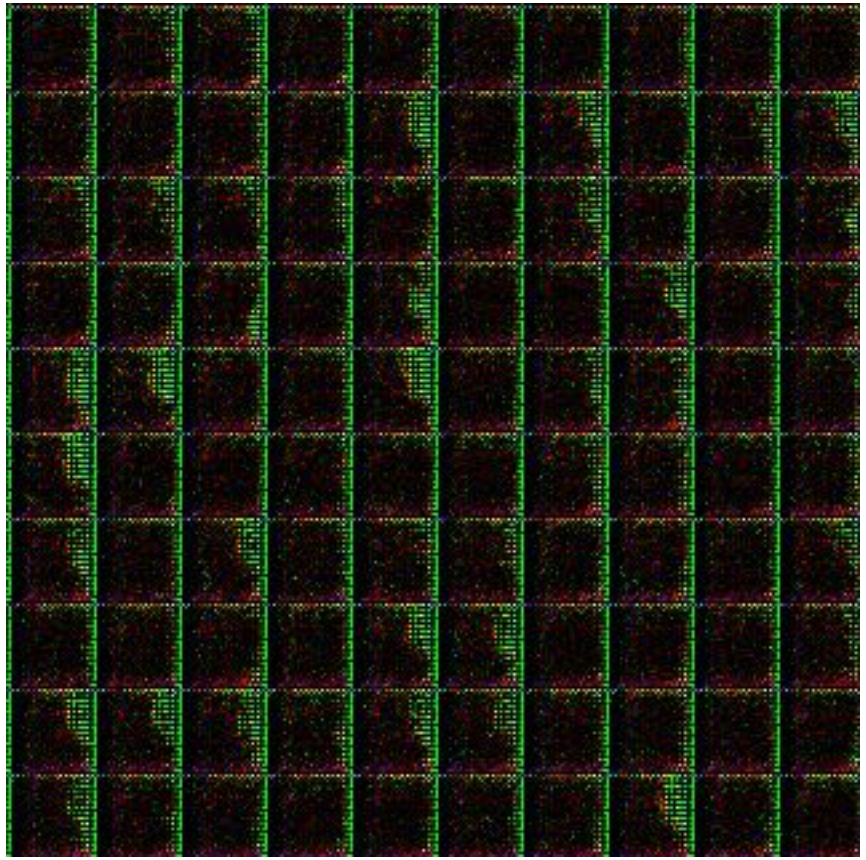
Training Procedure

(Goodfellow et al., 2014)



Source: Alec Radford

Generating 1D points



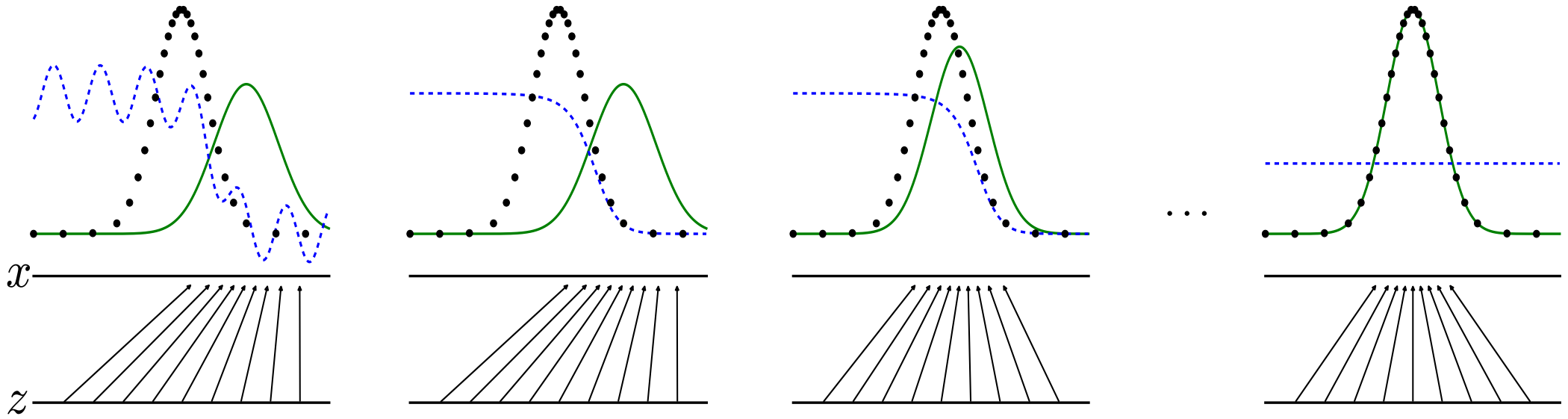
Source: OpenAI blog

Generating images

Training Procedure

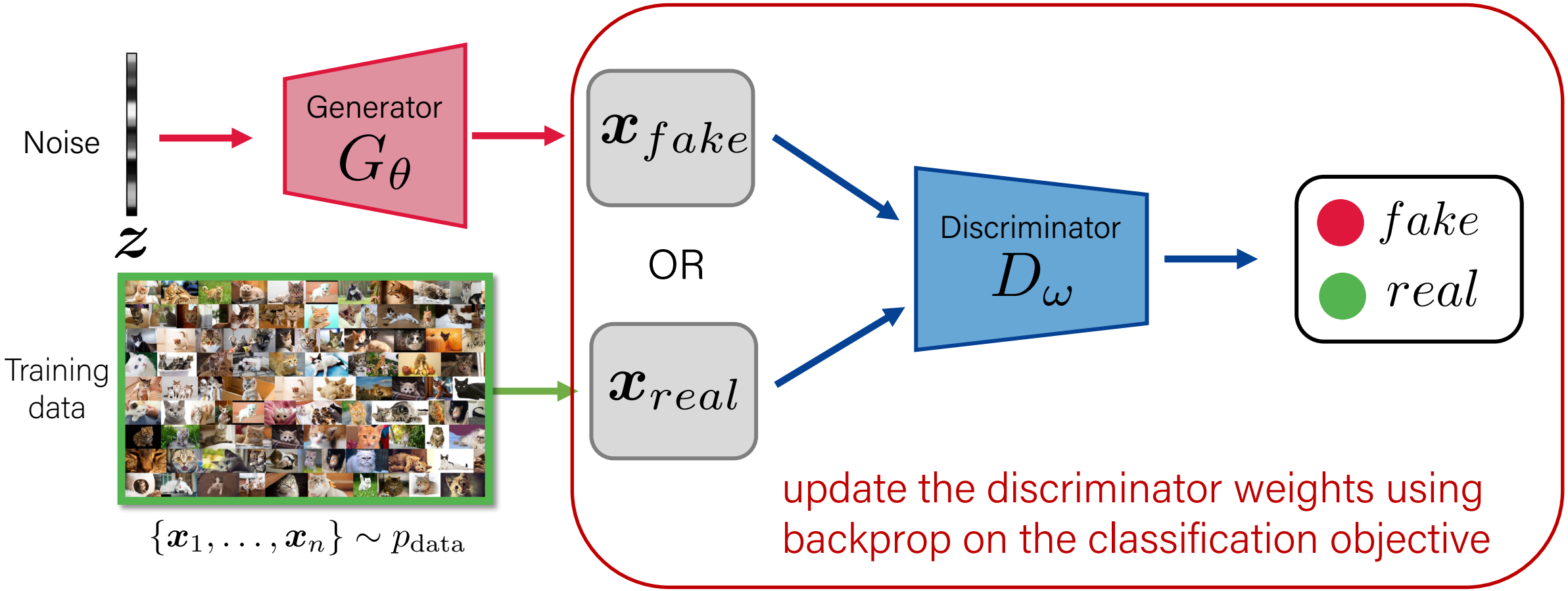
(Goodfellow et al., 2014)

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



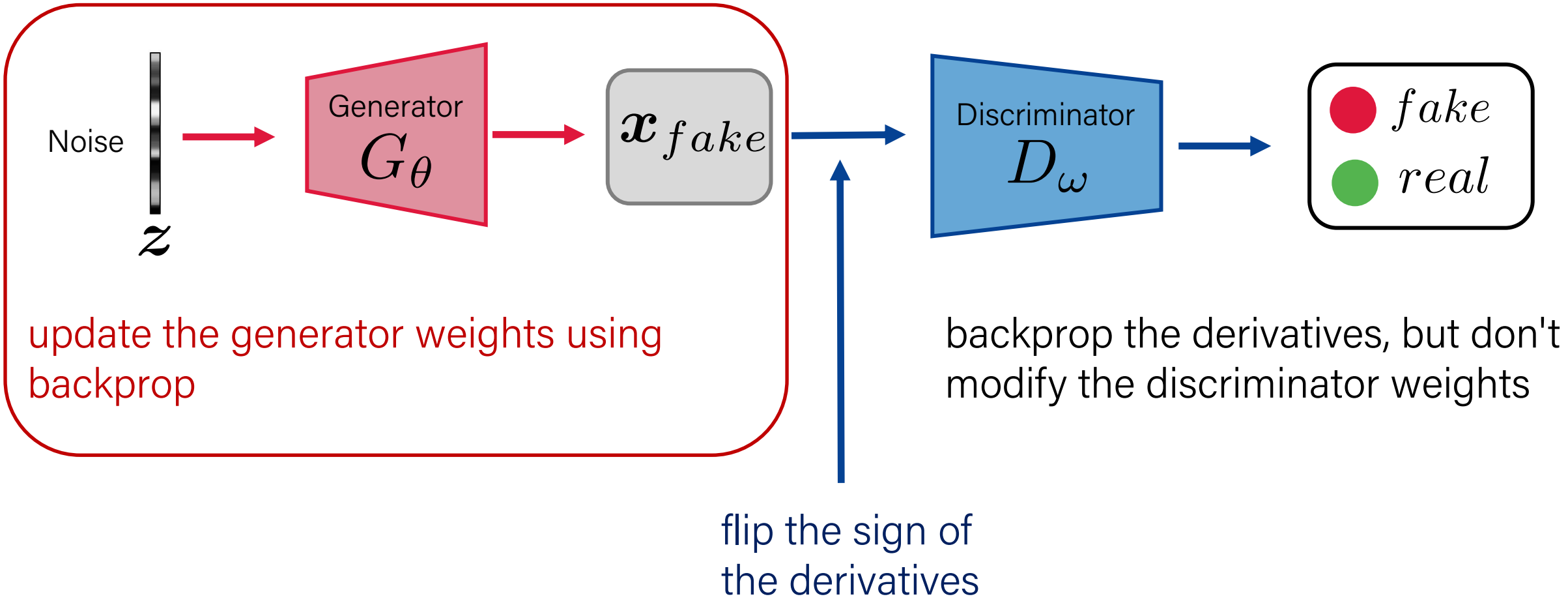
Training Procedure

- Updating the discriminator:



Training Procedure

- Updating the generator:



Intuition behind GAN Training

https://www.youtube.com/watch?v=No26JKQKZNE

Results

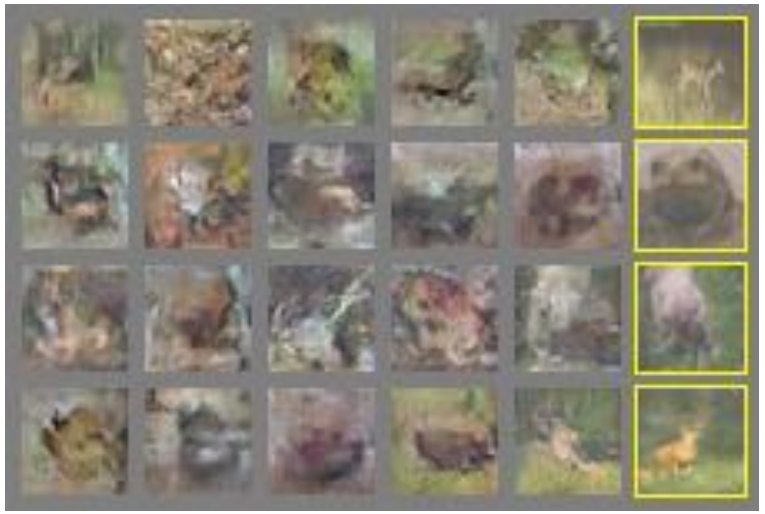
(Goodfellow et al., 2014)

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

MNIST samples

TFD samples

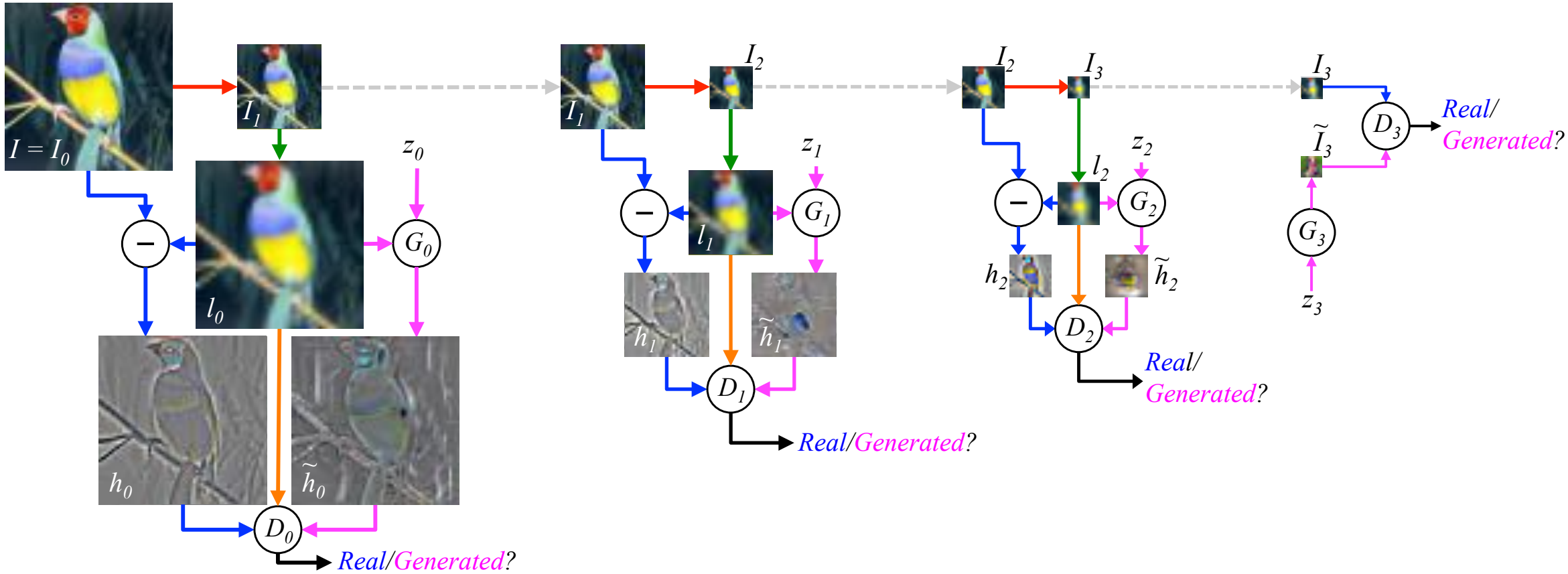
CIFAR10 samples
(*fully-connected model*)



CIFAR10 samples
(*convolutional discriminator, deconvolutional generator*)

Laplacian GANs (LAPGAN) (Denton et al., 2015)

- **Idea:** Combine GAN with a multi-scale image representation (Laplacian pyramid)



LAPGAN for LSUN Towers

64×64 pixels
~700K images

(Denton et al., 2015)

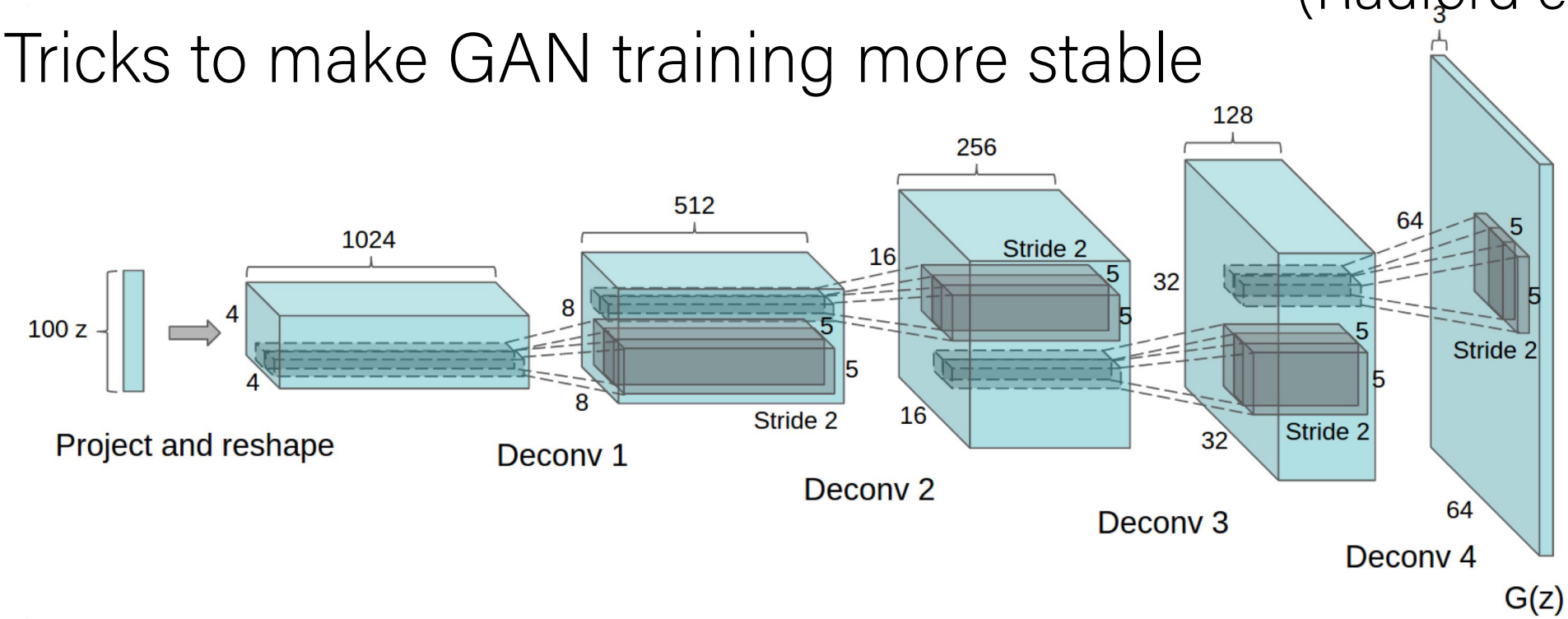
LAPGAN for LSUN Bedrooms

64×64 pixels
~3M images (Denton et al., 2015)

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

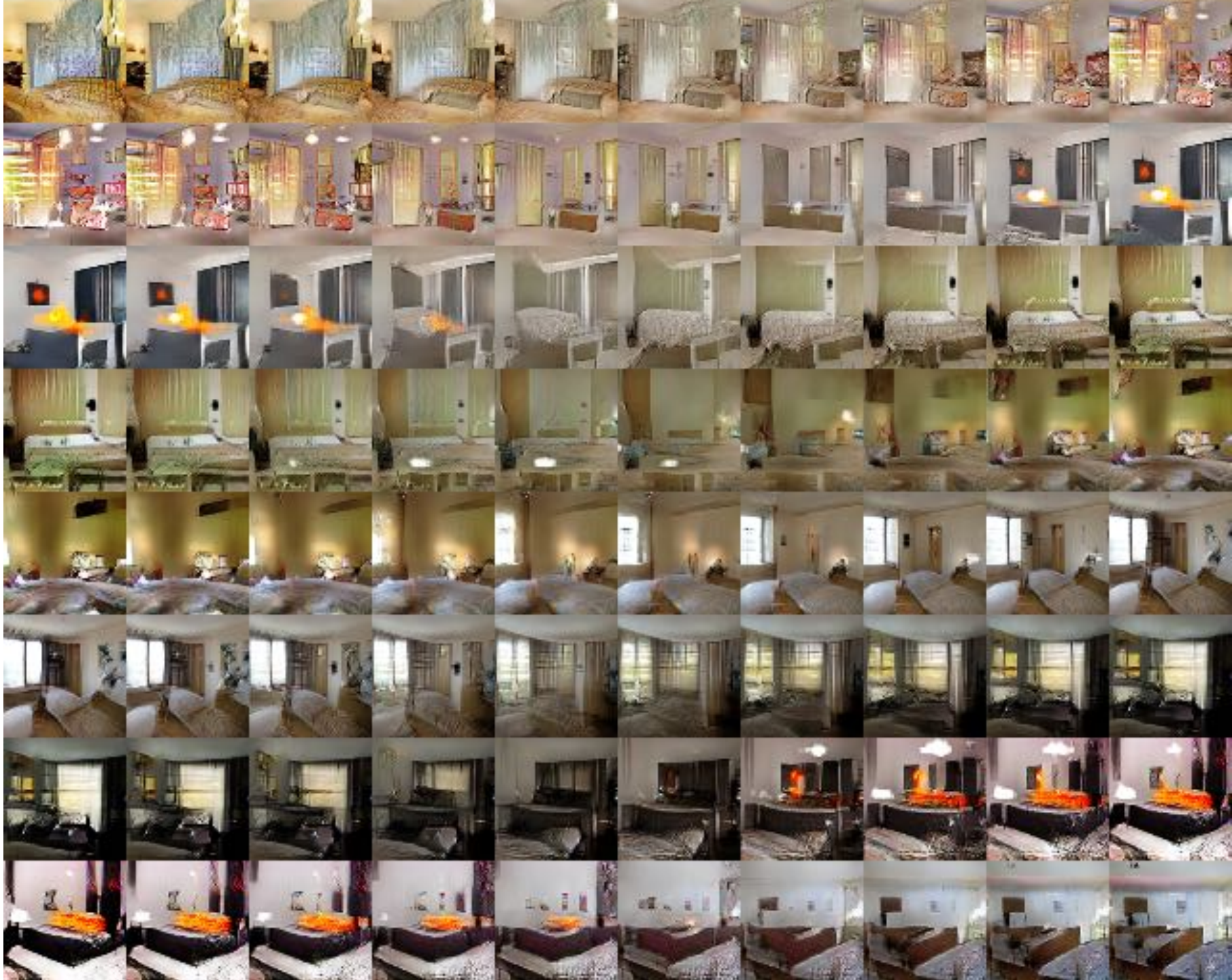
64×64 pixels
~3M images

(Radford et al., 2015)

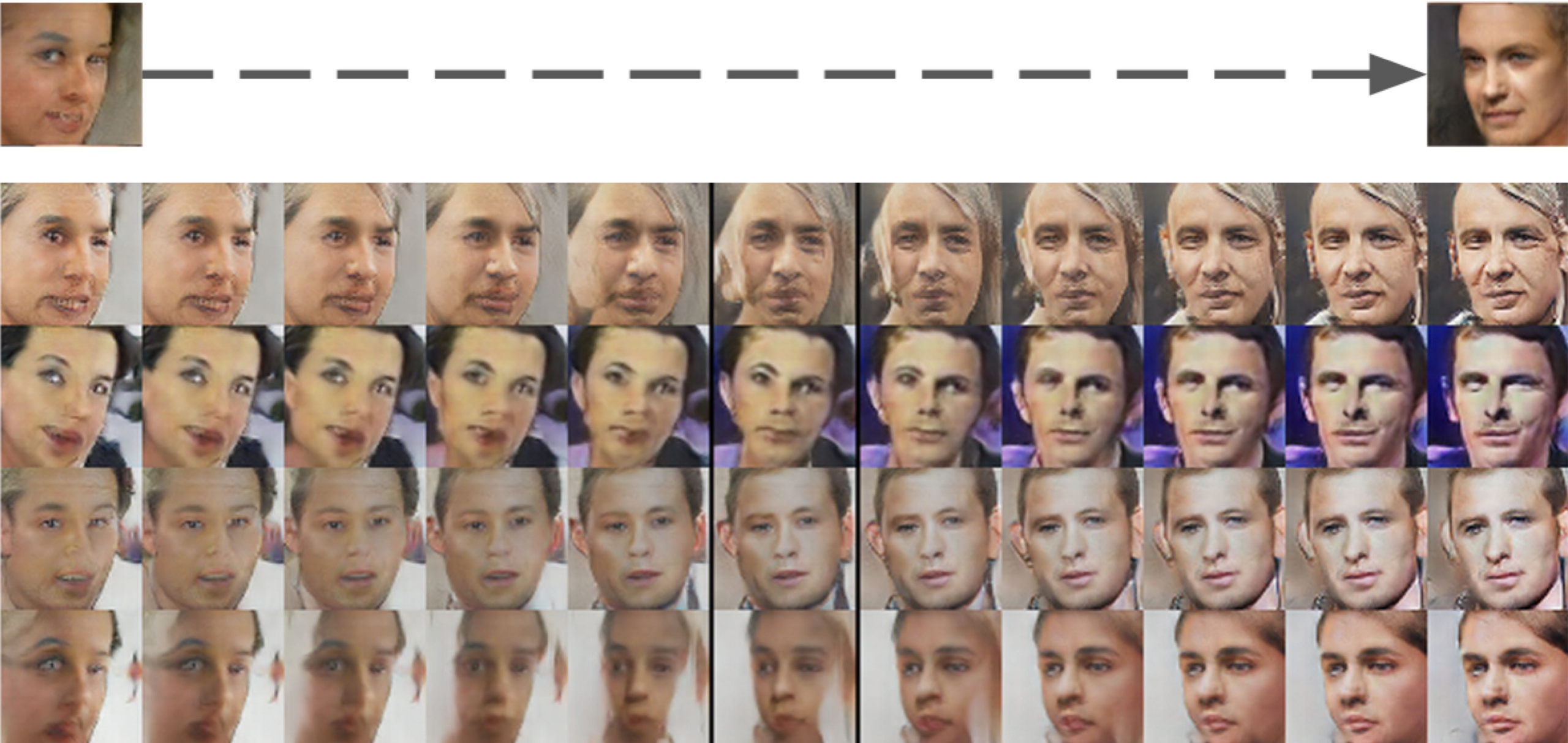
Walking over the latent space

(Radford et al., 2015)

- Interpolation suggests non-overfitting behavior

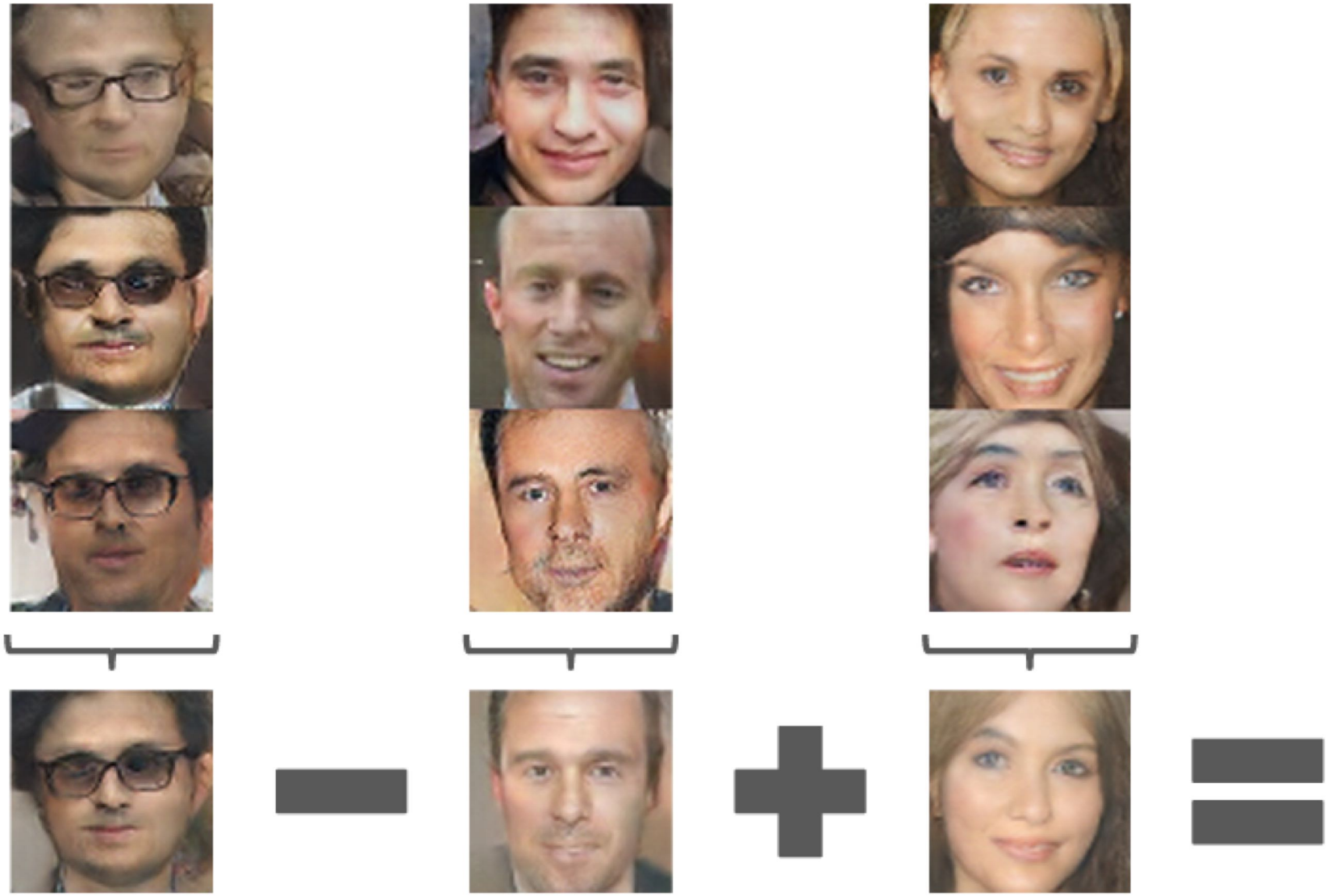


Walking over the latent space (Radford et al., 2015)



Vector Space Arithmetic

(Radford et al., 2015)



man
with glasses

man
without glasses

woman
without glasses

woman with glasses

Vector Space Arithmetic

(Radford et al., 2015)

smiling woman

neutral woman

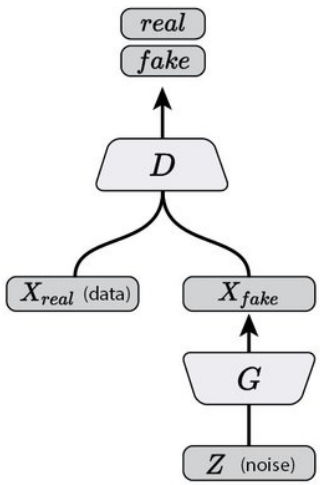
neutral man

smiling man

Subclasses of GANs

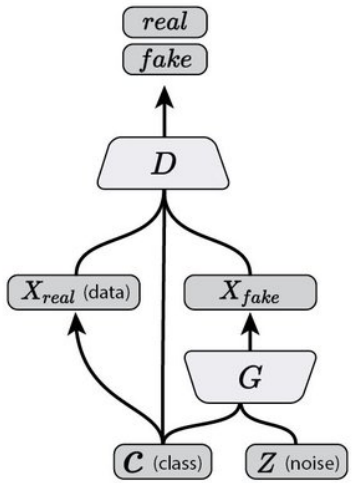
Vanilla GAN

Vanilla GAN
(Goodfellow, et al., 2014)

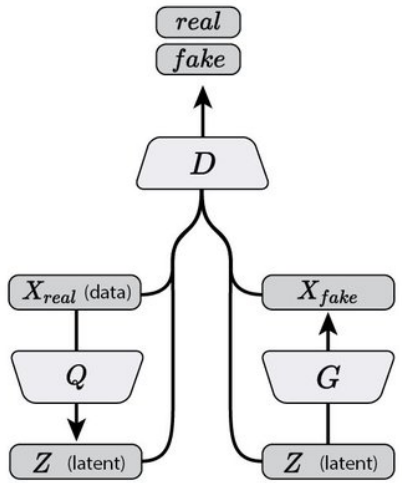


Discriminator Looks at Latent Variables

Conditional GAN
(Mirza & Osindero, 2014)

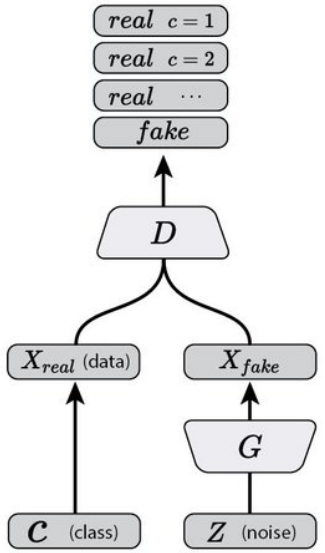


Bidirectional GAN
(Donahue, et al., 2016; Dumoulin, et al., 2016)

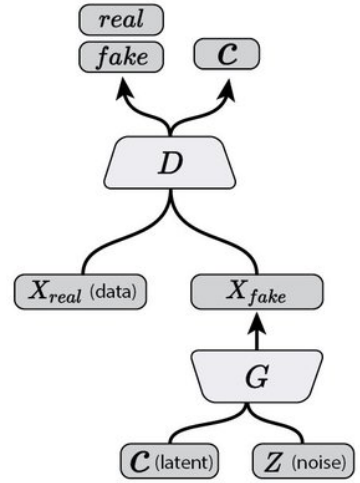


Discriminator Predicts Latent Variables

Semi-Supervised GAN
(Odena, 2016; Salimans, et al., 2016)



InfoGAN
(Chen, et al., 2016)



Auxiliary Classifier GAN
(Odena, et al., 2016)

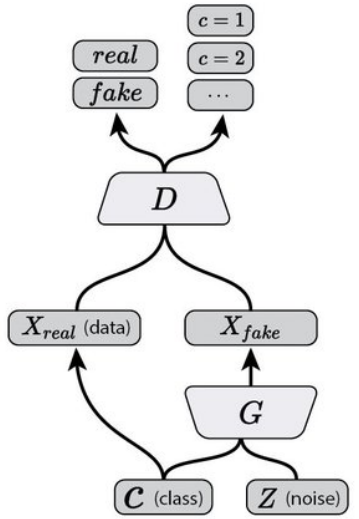
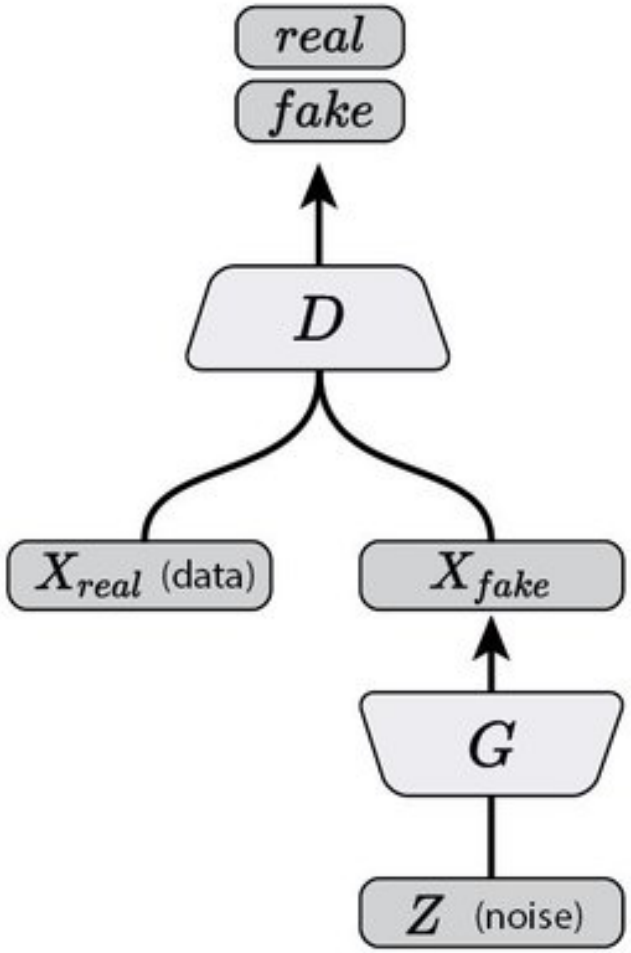
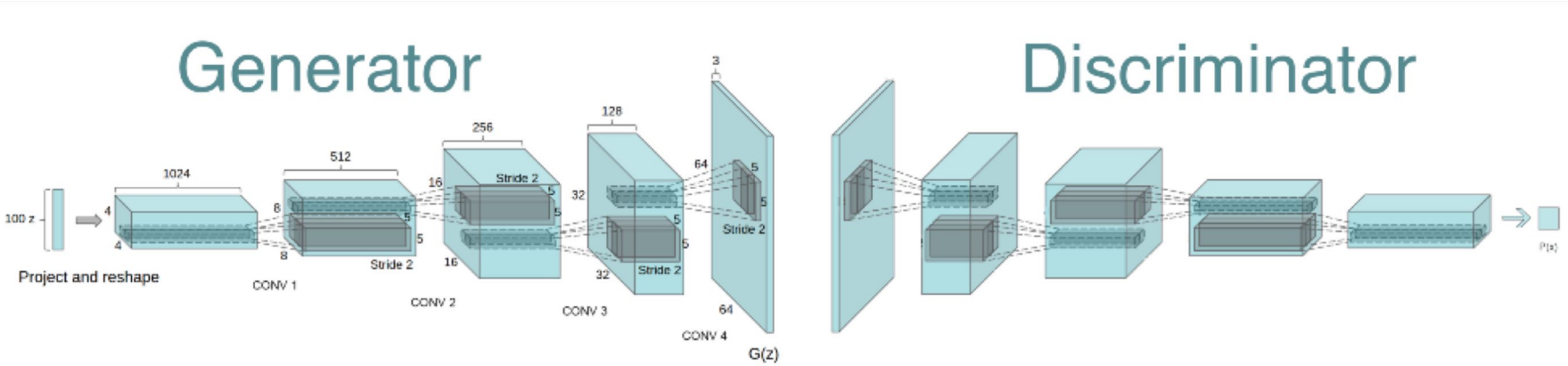


Image: Christopher Olah

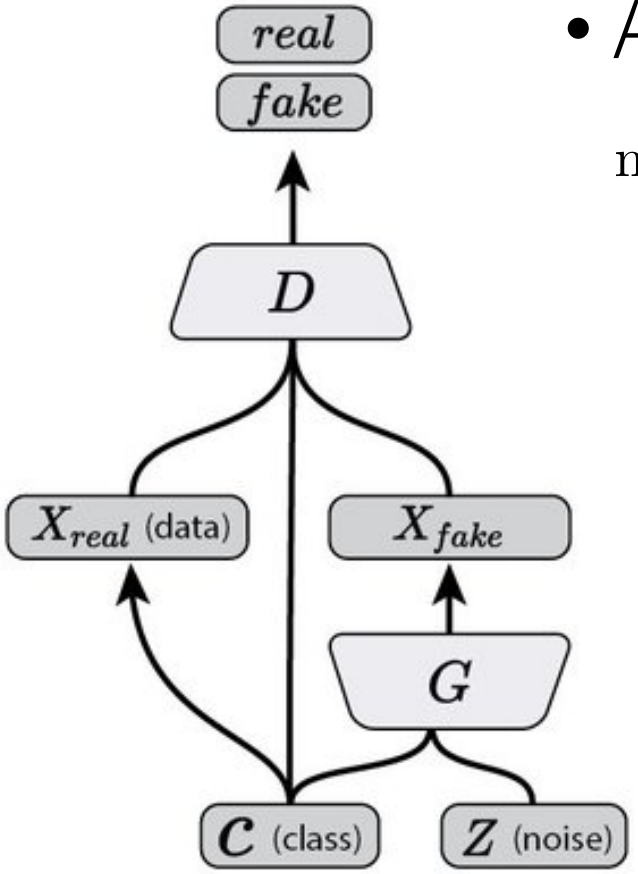
Vanilla GAN (Goodfellow et al., 2014)





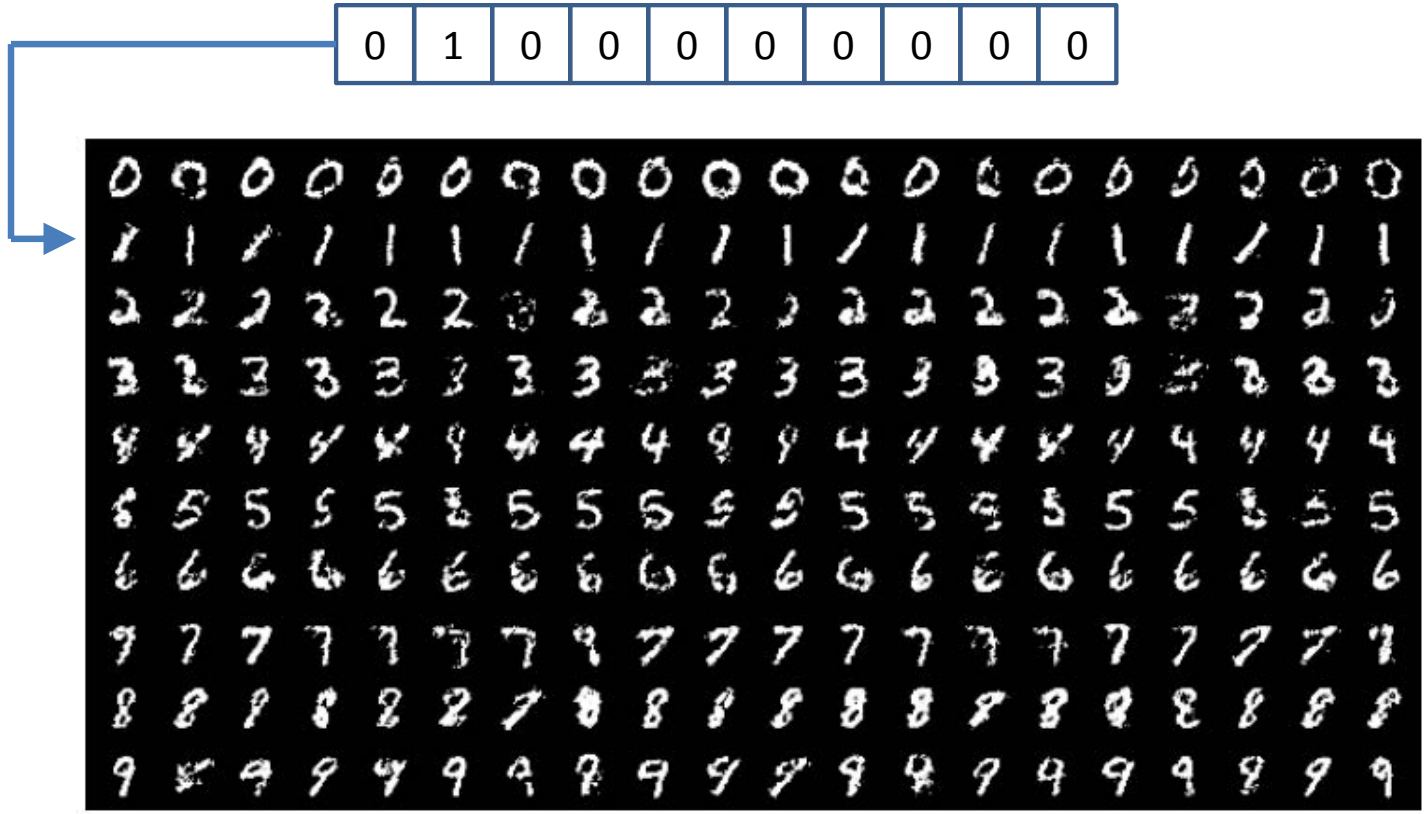
DCGAN (Radford et al., 2015)

Conditional GAN (Mirza and Osindero, 2014)

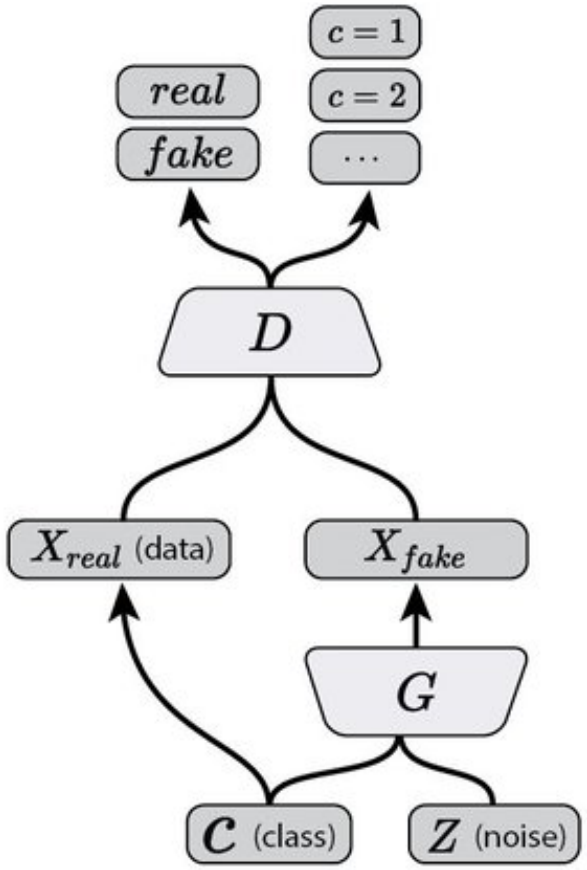


- Add conditional variables $\boldsymbol{y}$ into G and D

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))]$$



Auxiliary Classifier GAN (Odena et al., 2016)



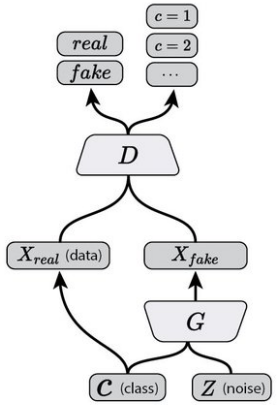
- Every generated sample has a corresponding class label

$$L_S = E[\log P(S = real \mid X_{real})] + E[\log P(S = fake \mid X_{fake})]$$
$$L_C = E[\log P(C = c \mid X_{real})] + E[\log P(C = c \mid X_{fake})]$$

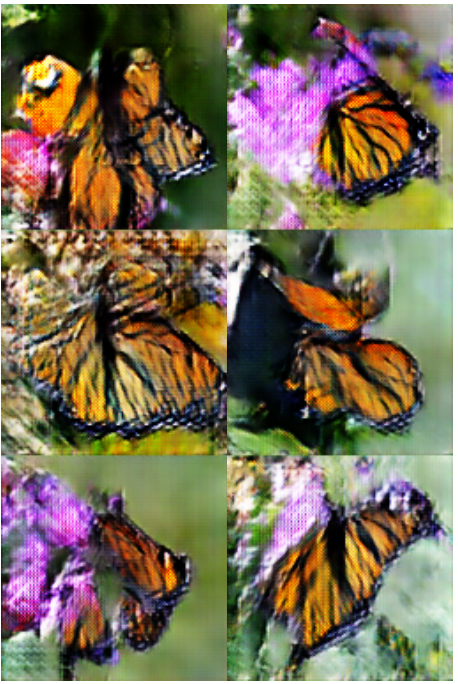
- D is trained to maximize $L_S + L_C$
- G is trained to maximize $L_C - L_S$

- Learns a representation for z that is independent of class label

Auxiliary Classifier GAN (Odena et al., 2016)



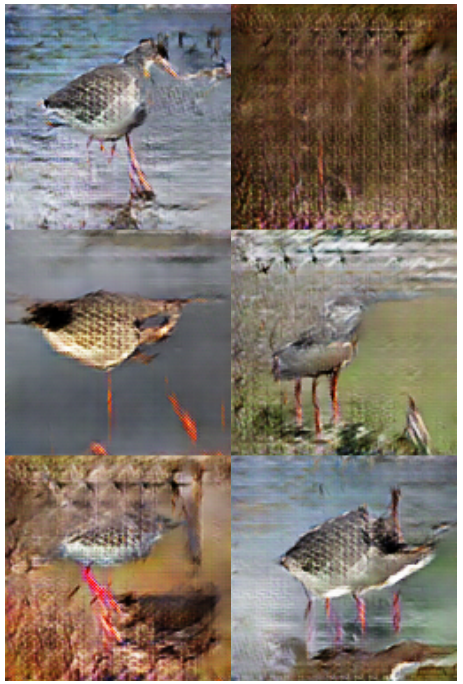
128×128 resolution samples from 5 classes taken from an AC-GAN trained on the ImageNet



monarch butterfly

goldfinch

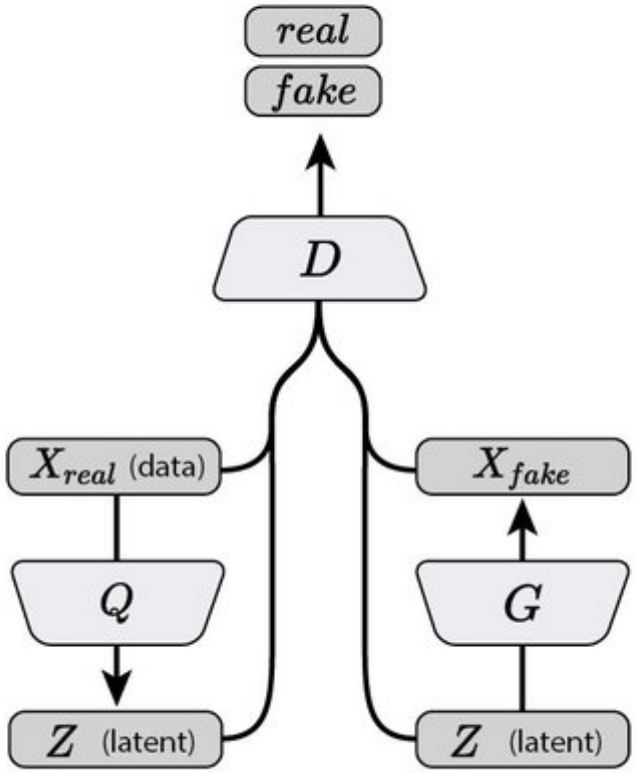
daisy



redshank

grey whale

Bidirectional GAN (Donahue et al., 2016; Dumoulin et al., 2016)



- Jointly learns a generator network and an inference network using an adversarial process.

$$\min_G \max_D V(D,G) = \mathbb{E}_{q(\boldsymbol{x})}[\log(D(\boldsymbol{x}, G_z(\boldsymbol{x})))] + \mathbb{E}_{p(\boldsymbol{z})}[\log(1 - D(G_x(\boldsymbol{z}), \boldsymbol{z}))]$$

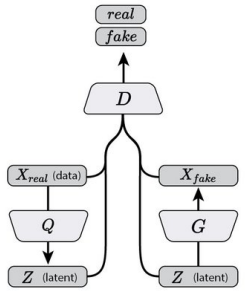
$$= \iint q(\boldsymbol{x})q(\boldsymbol{z} \mid \boldsymbol{x}) \log(D(\boldsymbol{x}, \boldsymbol{z}))d\boldsymbol{x}d\boldsymbol{z}$$

$$+ \iint p(\boldsymbol{z})p(\boldsymbol{x} \mid \boldsymbol{z}) \log(1 - D(\boldsymbol{x}, \boldsymbol{z}))d\boldsymbol{x}d\boldsymbol{z}.$$

CelebA reconstructions

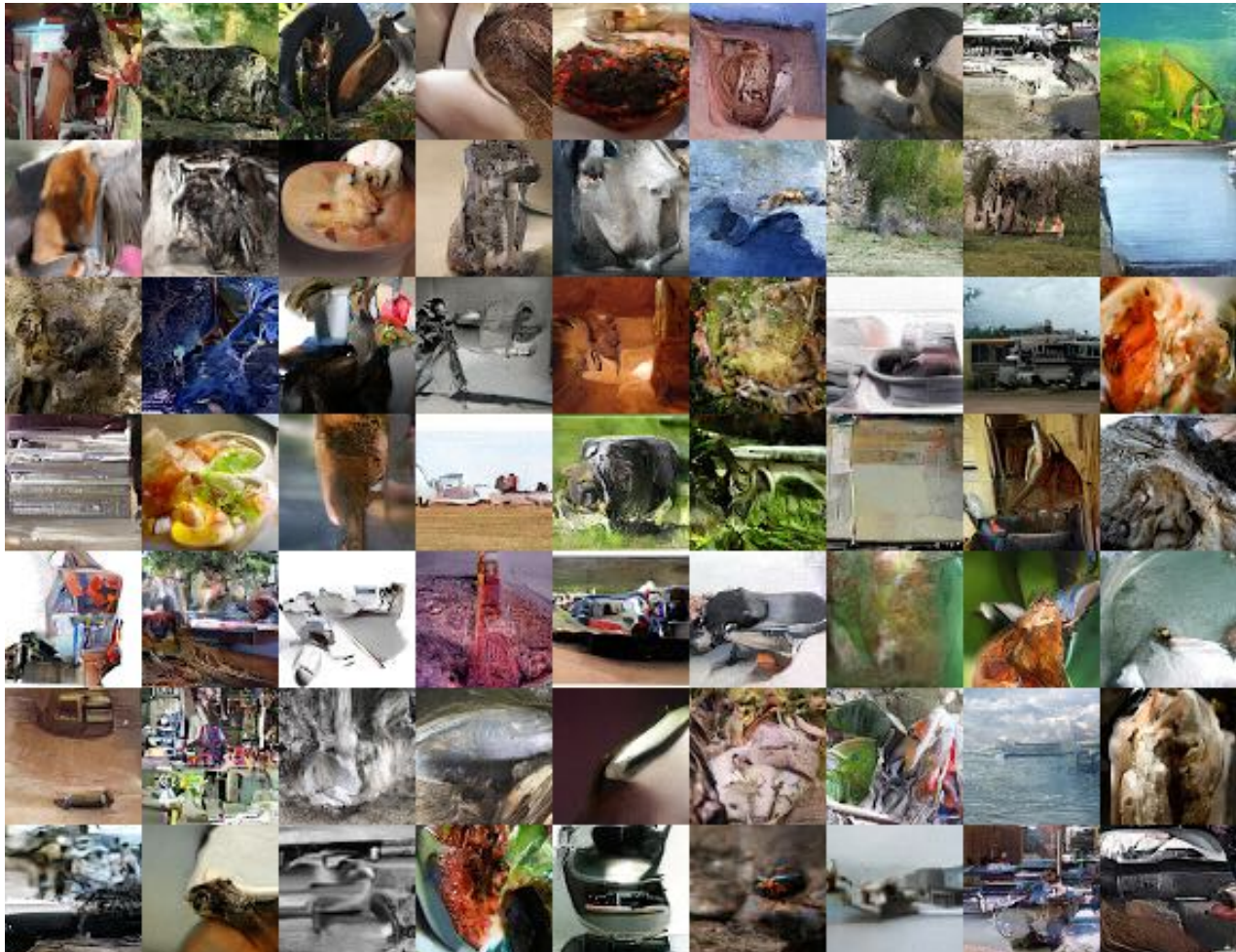
SVNH reconstructions

Bidirectional GAN (Donahue et al., 2016; Dumoulin et al., 2016)



LSUN bedrooms

Tiny ImageNet



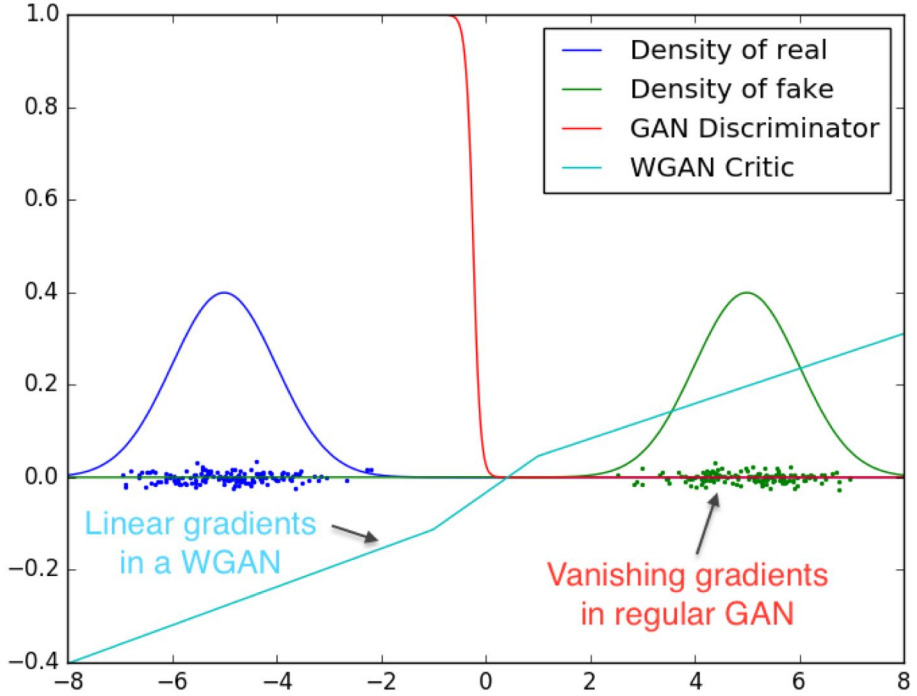
Recent Advances

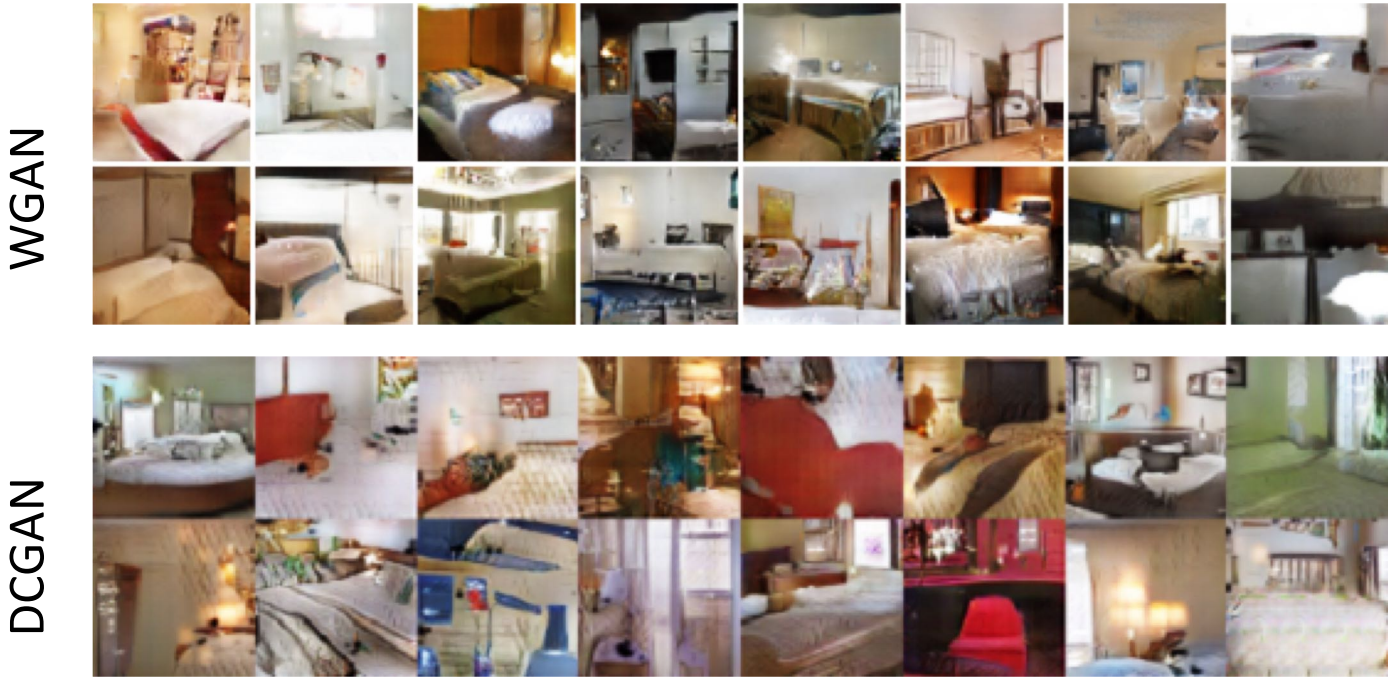
Wasserstein GAN (Arjovsky et al., 2016)

- Objective based on Earth-Mover or Wassertein distance:

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

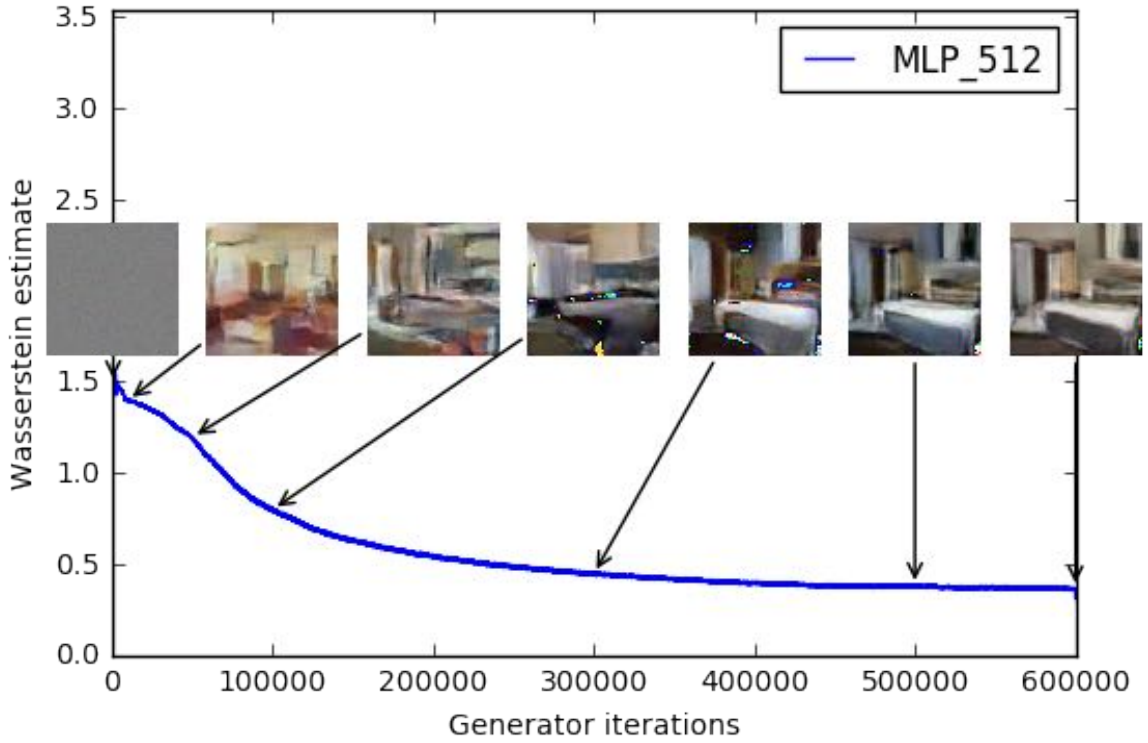
- Provides nice gradients over real and fake samples

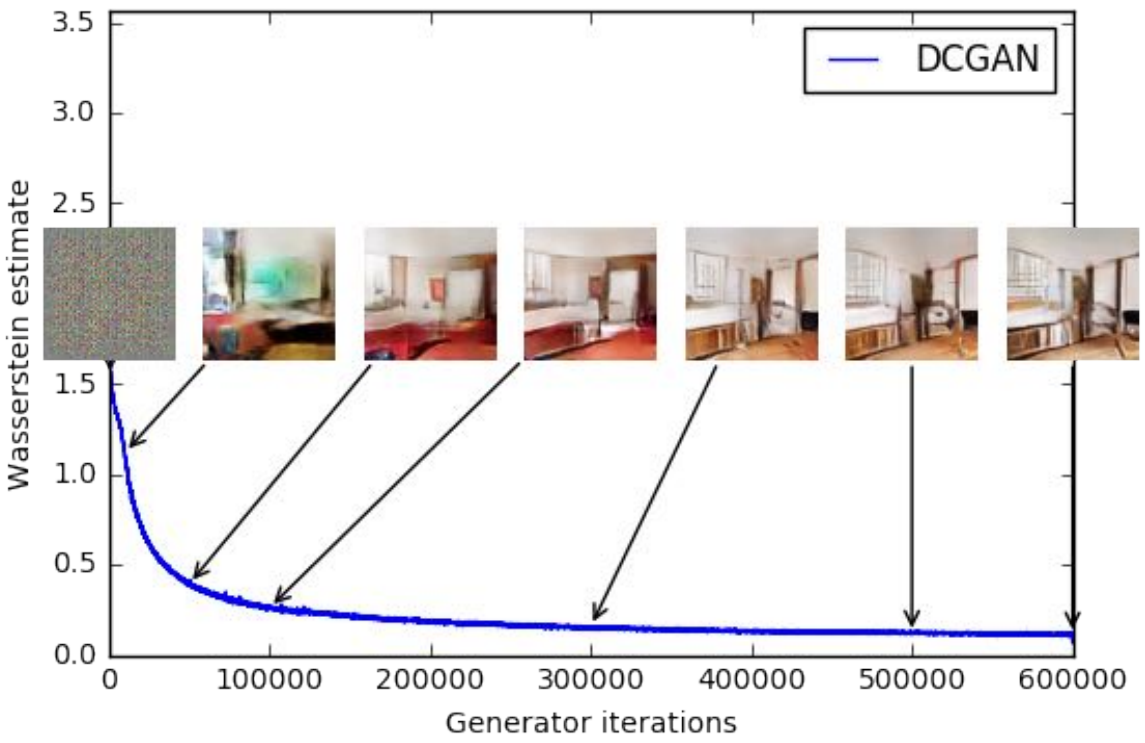




Wasserstein GAN (Arjovsky et al., 2016)

- Wasserstein loss seems to correlate well with image quality.





WGAN with gradient penalty ( )

$$L = \underbrace{\mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g}[D(\tilde{\boldsymbol{x}})] - \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathbb{P}_r}[D(\boldsymbol{x})]}_{\text{Original critic loss}} + \underbrace{\lambda \mathop{\mathbb{E}}_{\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{\boldsymbol{x}}}}\left[(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2\right]}_{\text{Our gradient penalty}}$$

- Faster convergence and higher-quality samples than WGAN with weight clipping
- Train a wide variety of GAN architectures with almost no hyperparameter tuning, including discrete models

Samples from a character-level GAN language model on Google Billion Word

WGAN with gradient penalty	
Busino game camperate spent odea	Solice Norkedin pring in since
In the bankaway of smarling the	ThiS record (31.) UBS) and Ch
SingersMay , who kill that imvic	It was not the annuas were plogr
Keray Pents of the same Reagun D	This will be us , the ect of DAN
Manging include a tudancs shat "	These leaded as most-worsd p2 a0
His Zuith Dudget , the Denmbern	The time I paidOa South Cubry i
In during the Uitational questio	Dour Fraps higs it was these del
Divos from The ’ noth ronkies of	This year out howneed allowed lo
She like Monday , of macunsuer S	Kaulna Seto consficutes to repor
The investor used ty the present	A can teal , he was schoon news
A papees are cointry congress oo	In th 200. Pesish picriers rega
A few year inom the group that s	Konney Panice rimimber the teami
He said this syenn said they wan	The new centuct cut Denester of
As a world 1 88 ,for Autouries	The near , had been one injostie
Foand , th Word people car , Il	The incestion to week to shorted
High of the upseader homing pull	The company the high product of
The guipe is worly move dogsfor	20 - The time of accomplete , wh
The 1874 incidested he could be	John WVuderenson seqiivic spends
The allo tooks to security and c	A ceetens in indestredly the Wat
Standard GAN objective	
ddddddddddddddddddddddddddddddd	ddddddddddddddddddddddddddddddd
ddddddddddddddddddddddddddddddd	ddddddddddddddddddddddddddddddd

Boundary Equilibrium GAN (BEGAN)

(Berthelot et al., 2017)

- A loss derived from the Wasserstein distance for training auto-encoder based GANs

$$\mathcal{L}(v) = |v - D(v)|^{\eta} \text{ where } \begin{cases} D : \mathbb{R}^{N_x} \mapsto \mathbb{R}^{N_x} & \text{is the autoencoder function.} \\ \eta \in \{1, 2\} & \text{is the target norm.} \\ v \in \mathbb{R}^{N_x} & \text{is a sample of dimension } N_x. \end{cases}$$

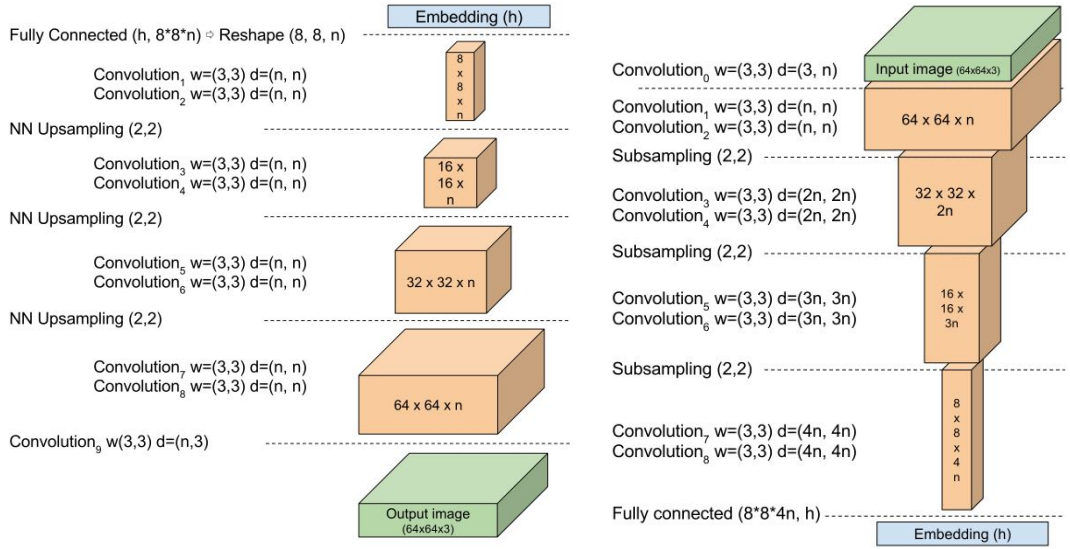
  - Wasserstein distance btw. the reconstruction losses of real and generated data

- Convergence measure:

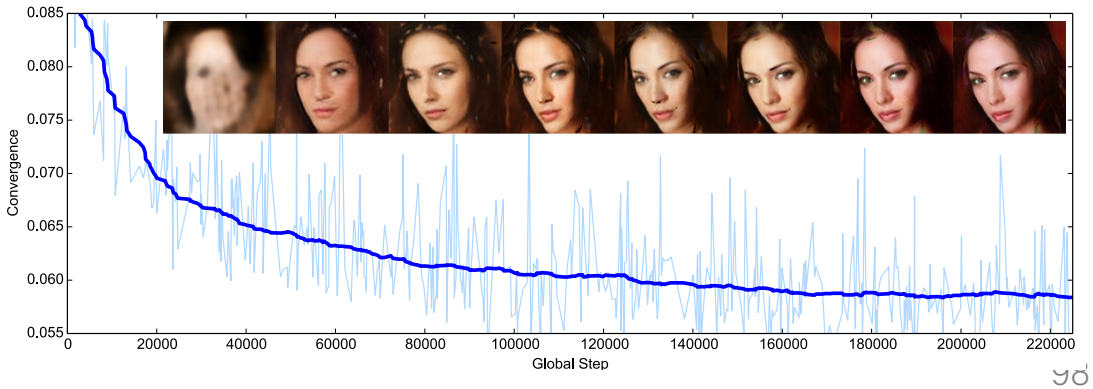
$$\mathcal{M}_{global} = \mathcal{L}(x) + |\gamma\mathcal{L}(x) - \mathcal{L}(G(z_G))|$$

- Objective:

$$\begin{cases} \mathcal{L}_D = \mathcal{L}(x) - k_t.\mathcal{L}(G(z_D)) & \text{for } \theta_D \\ \mathcal{L}_G = \mathcal{L}(G(z_G)) & \text{for } \theta_G \\ k_{t+1} = k_t + \lambda_k(\gamma\mathcal{L}(x) - \mathcal{L}(G(z_G))) & \text{for each training step } t \end{cases}$$



(a) Generator/Decoder (b) Encoder



BEGANs for CelebA

360K celebrity face images
128x128 with 128 filters

(Berthelot et al., 2017)

Interpolations in the latent space

Mirror interpolation example

Applications of GANs

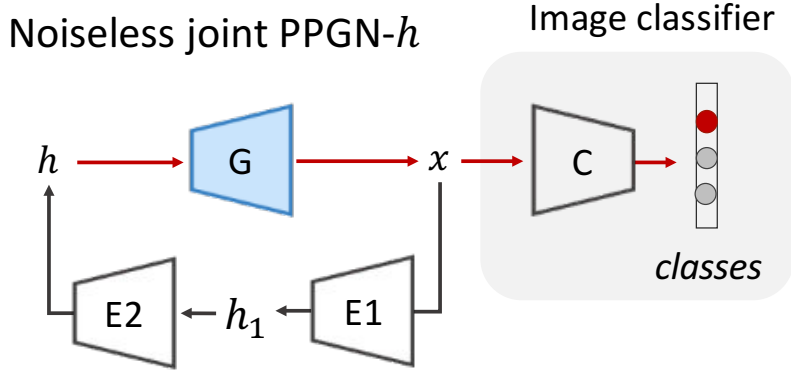
Semi-supervised Classification (Salimans et al., 2016; Dumoulin et al., 2016)

SVNH

Model	Misclassification rate
VAE (M1 + M2) (Kingma et al., 2014)	36.02
SWWAE with dropout (Zhao et al., 2015)	23.56
DCGAN + L2-SVM (Radford et al., 2015)	22.18
SDGM (Maaløe et al., 2016)	16.61
GAN (feature matching) (Salimans et al., 2016)	$\mathbf{8.11 \pm 1.3}$
ALI (ours, L2-SVM)	19.14 ± 0.50
ALI (ours, no feature matching)	$\mathbf{7.42 \pm 0.65}$

Class-specific Image Generation (Nguyen et al., 2016)

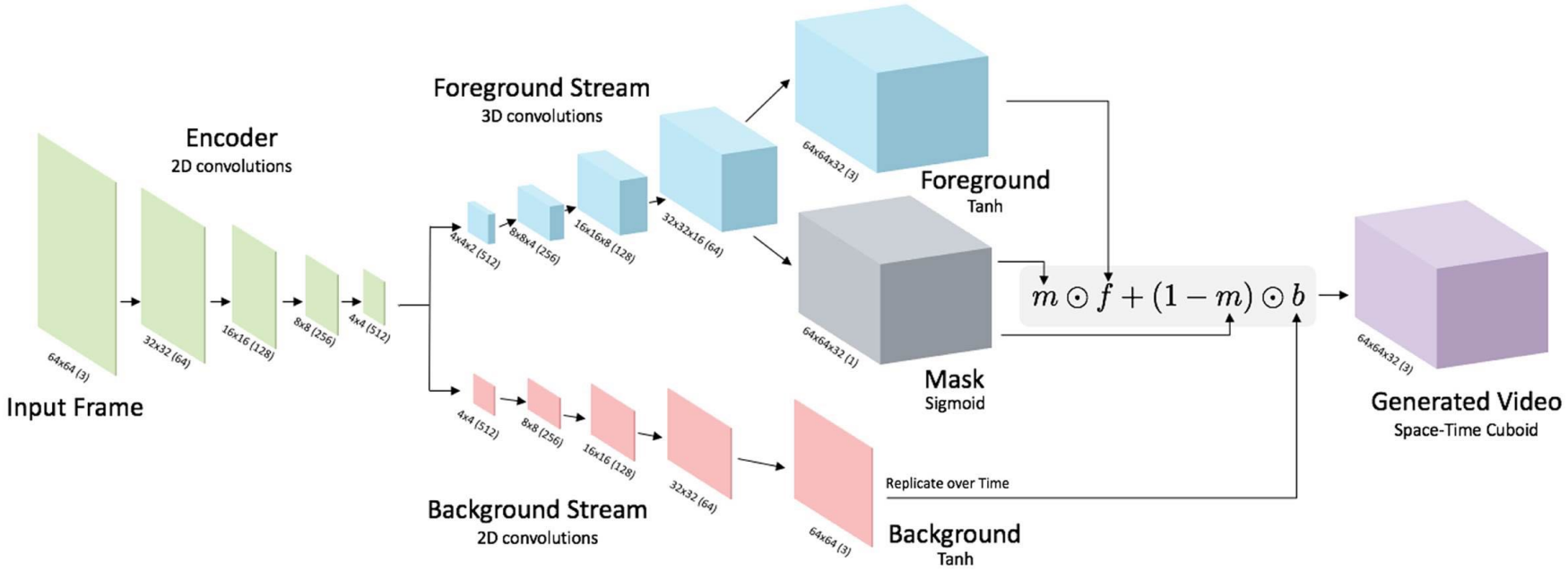
- Generates 227x227 realistic images from all ImageNet classes
- Combines adversarial training, moment matching, denoising autoencoders, and Langevin sampling



redshank ant monastery

volcano

Video Generation (Vondrick et al., 2016)

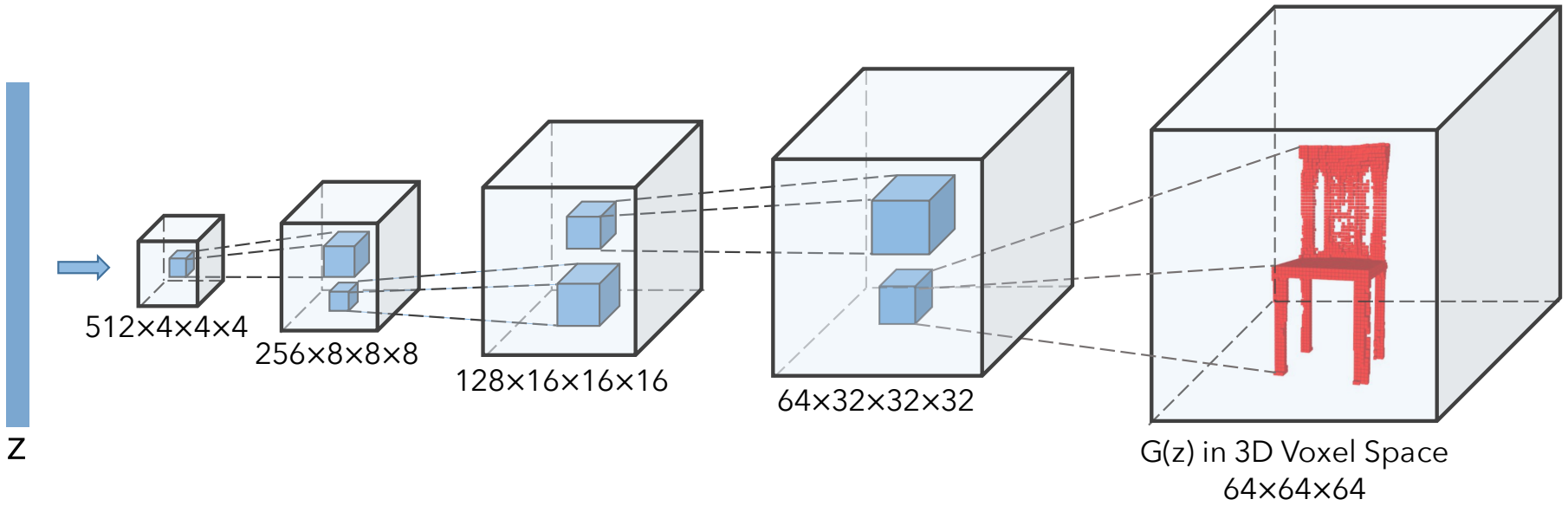


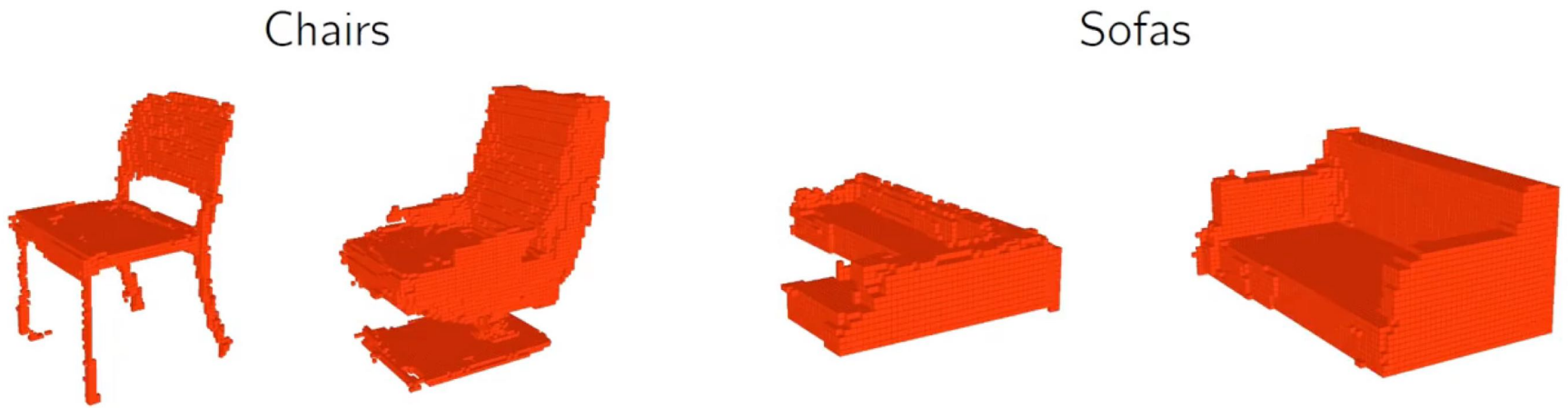
Beach

Golf

Train Station

Generative Shape Modeling (Wu et al., 2016)





Text-to-Image Synthesis (Zhang et al., 2016)

The small bird has a red head with feathers that fade from red to gray from head to tail

The petals of this flower are white with a large stigma	A unique yellow flower with no visible pistils protruding from the center	This flower is pink and yellow in color, with petals that are oddly shaped	This is a light colored flower with many different petals on a green stem	This flower is yellow and green in color, with petals that are ruffled	The flower have large petals that are pink with yellow on some of the petals	A flower that has white petals with some tones of yellow and green filaments

Single Image Super-Resolution (Ledig et al., 2016)

- Combine content loss with adversarial loss

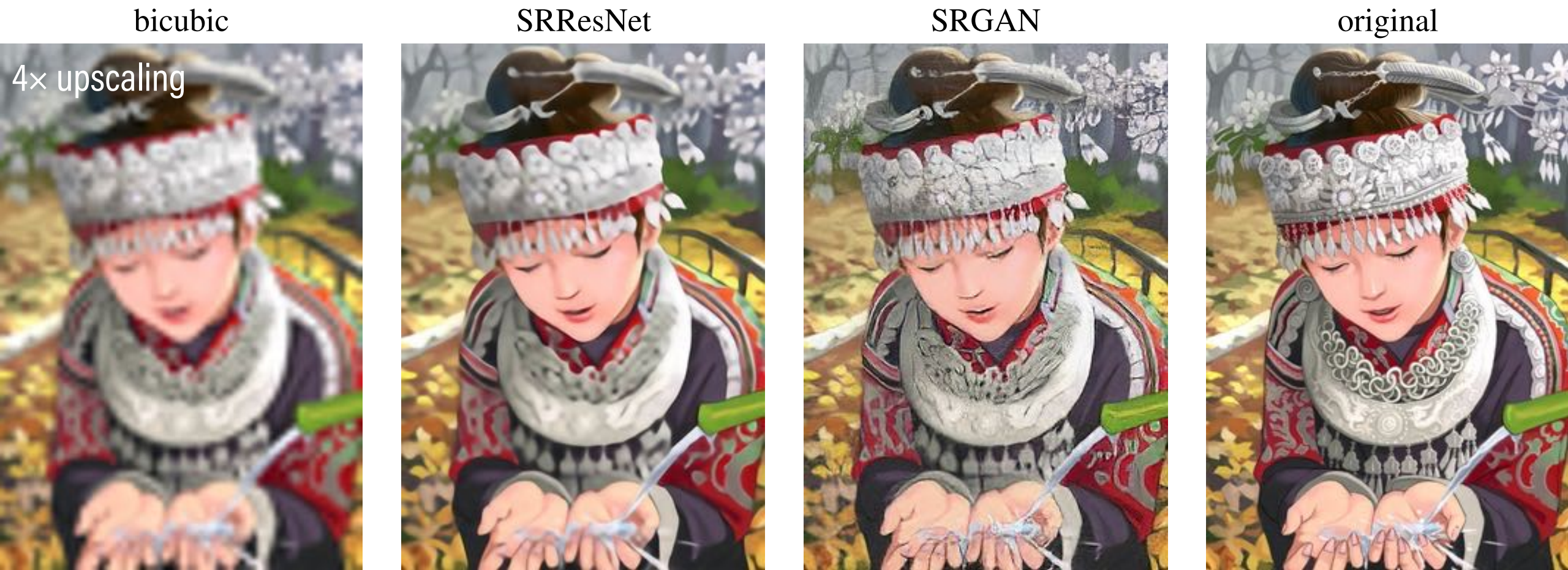


Image Inpainting (Pathak et al., 2016)

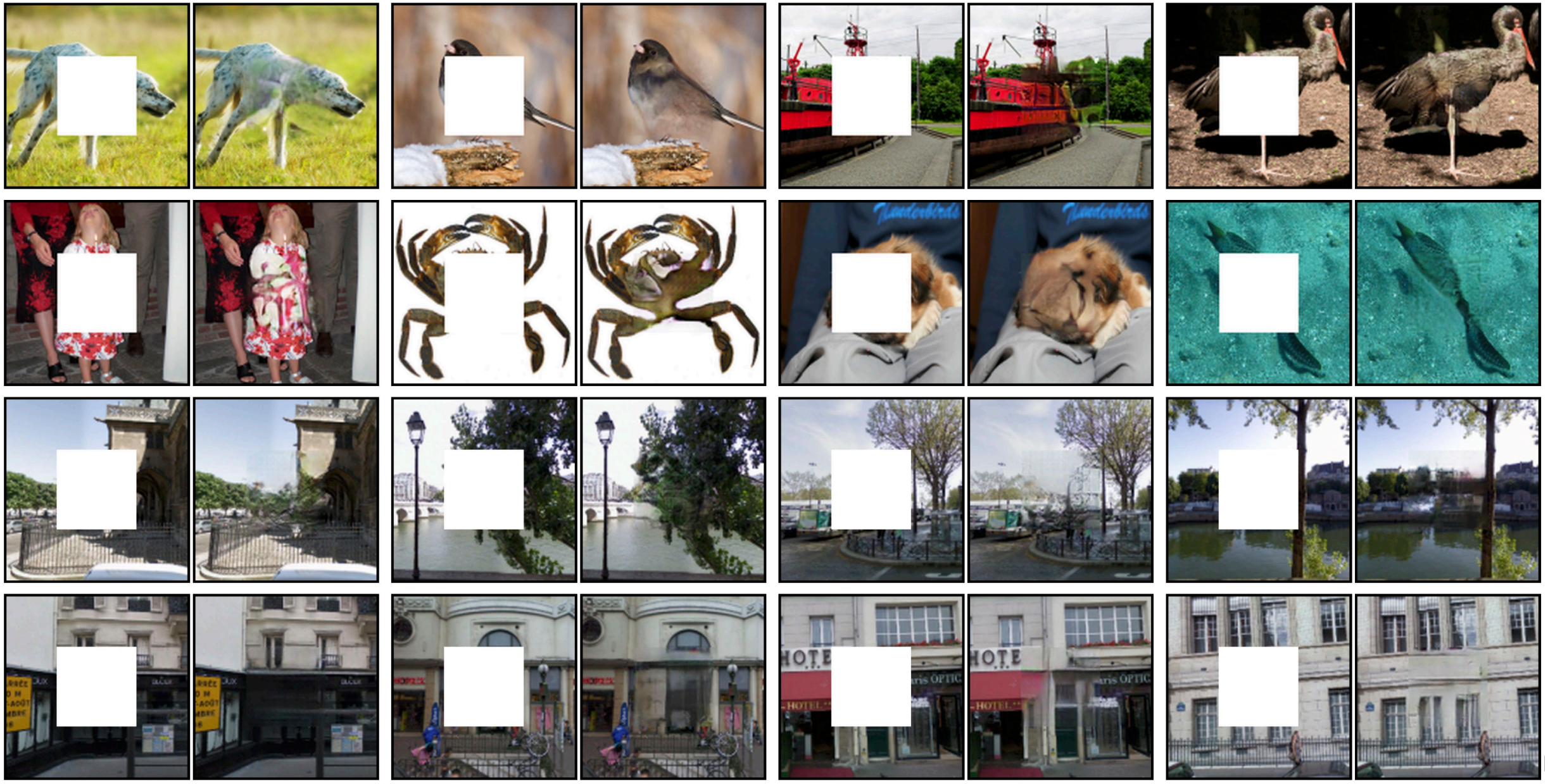
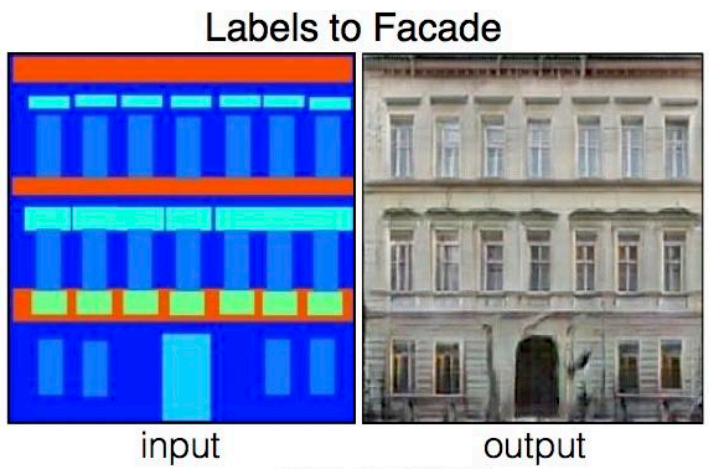
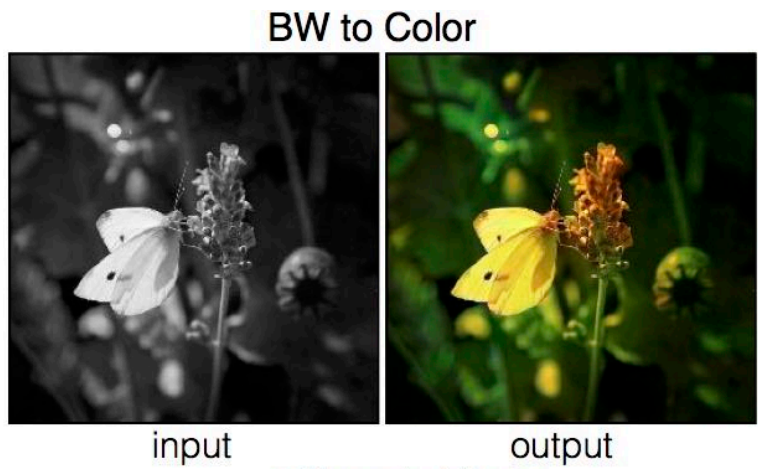
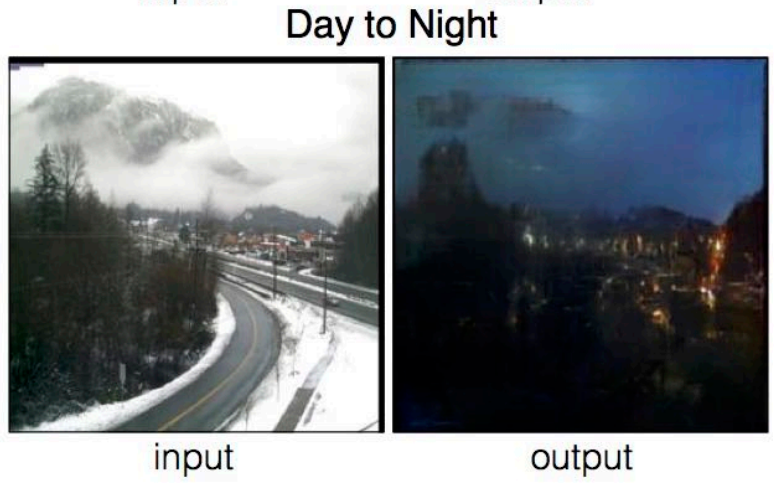


Image to Image Translation (pix2pix) (Isola et al., 2016)

- Requires paired training data







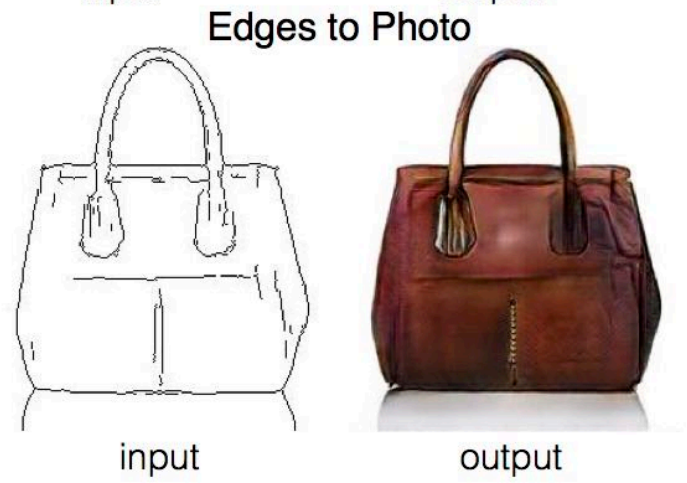


Image to Image Translation (CycleGAN) (Zhu et al., 2017)

- Unsupervised, does not require paired training data

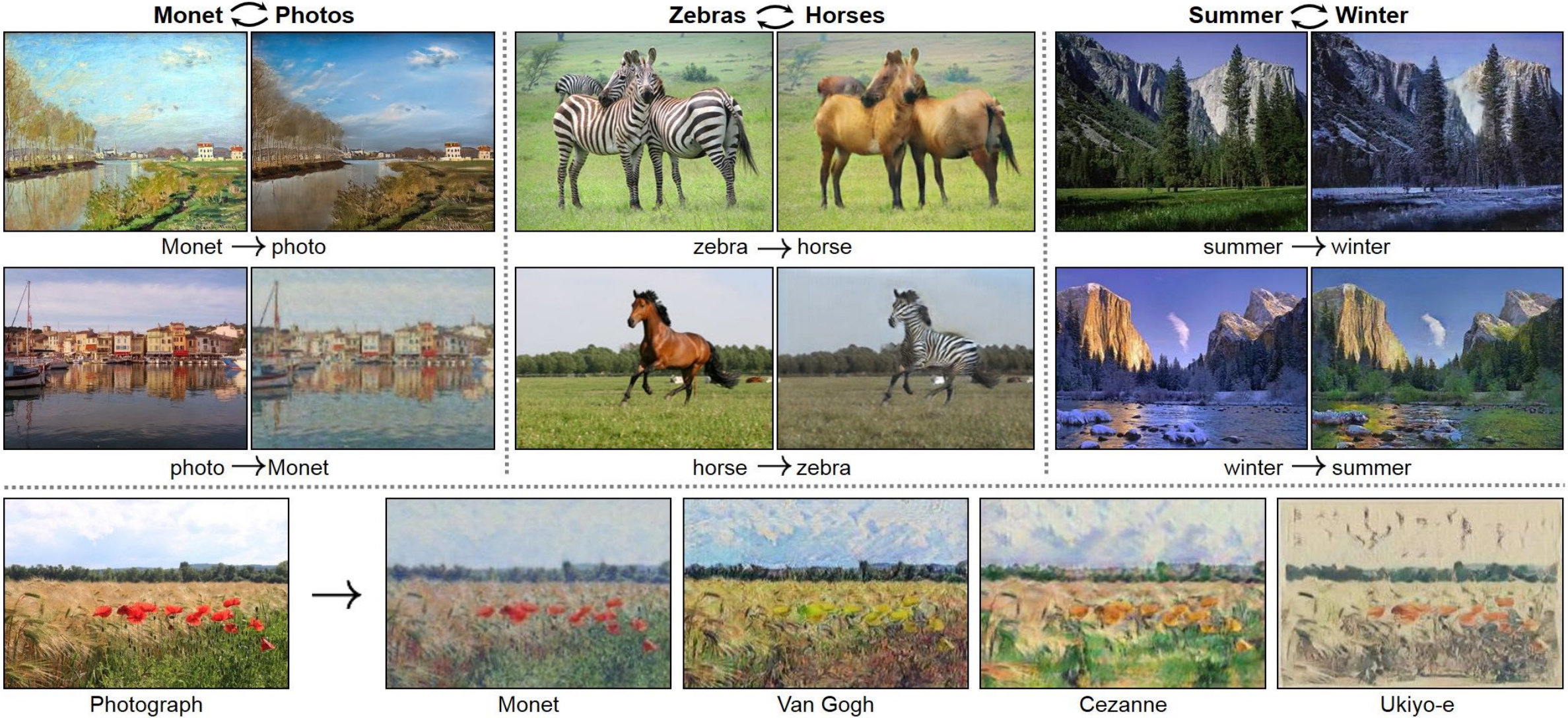
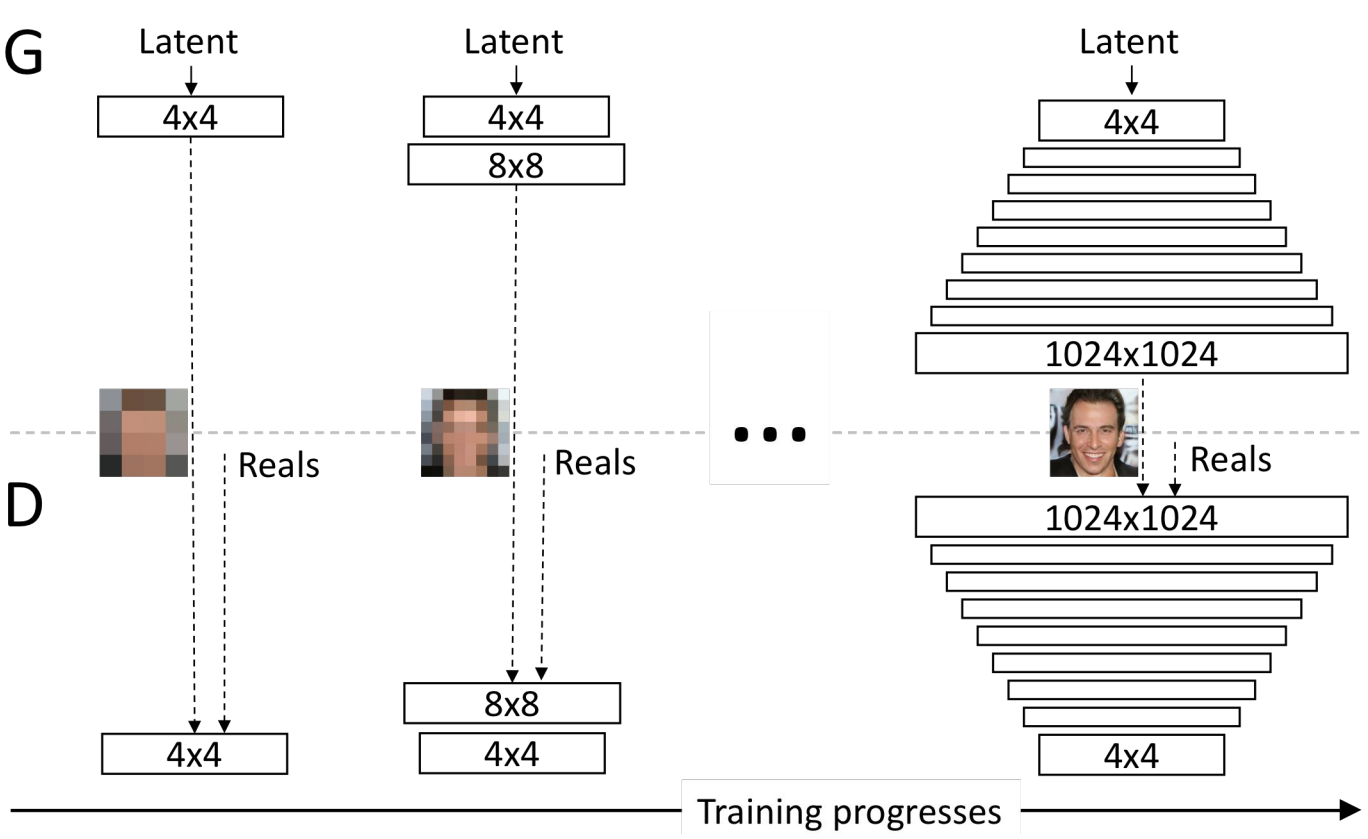


Image to Image Translation (CycleGAN) (Zhu et al., 2017)

- Unsupervised, does not require paired training data

Progressive GANs (Karras et al., 2017)

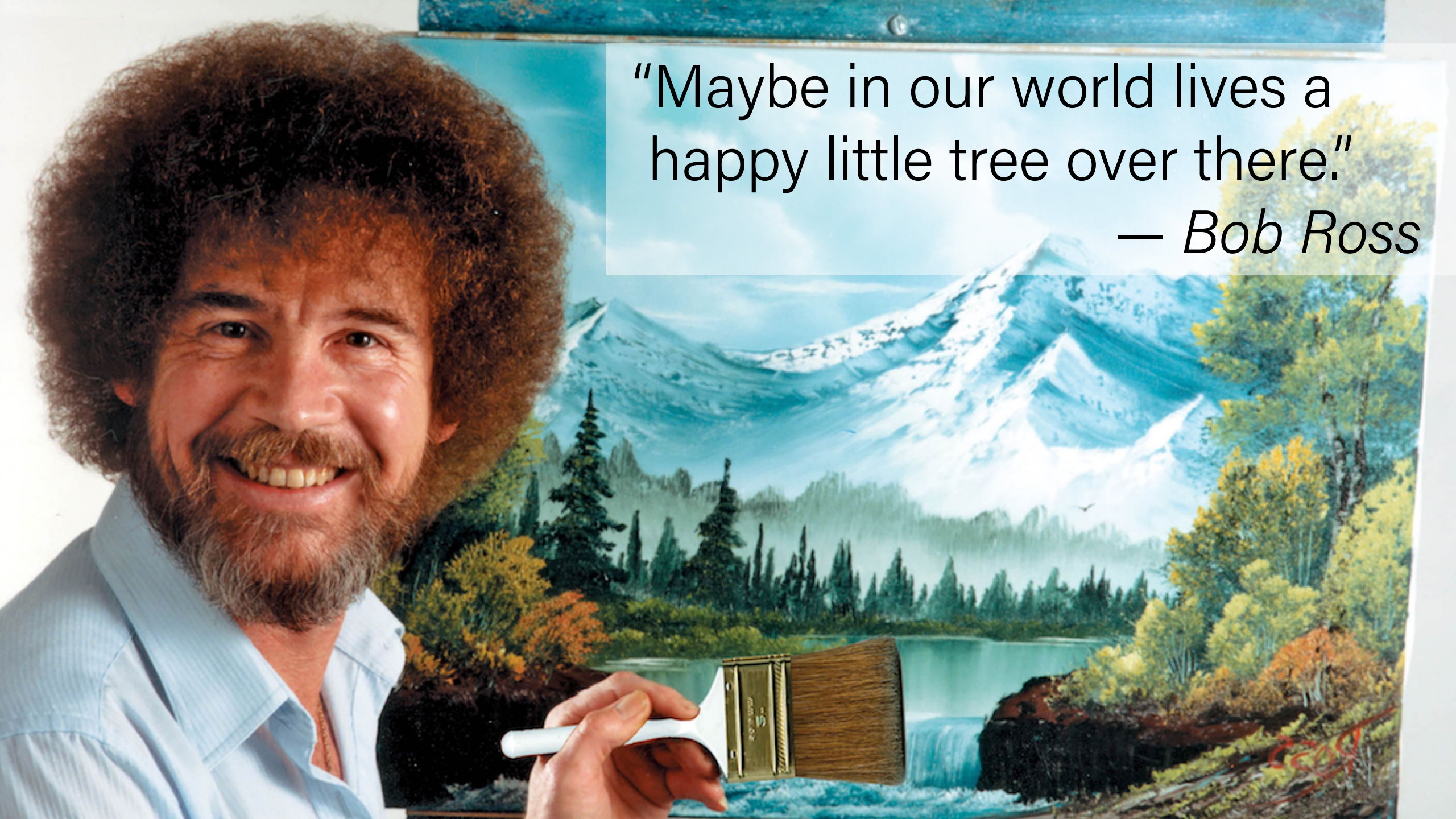
- Allows to obtain high resolution synthetic images

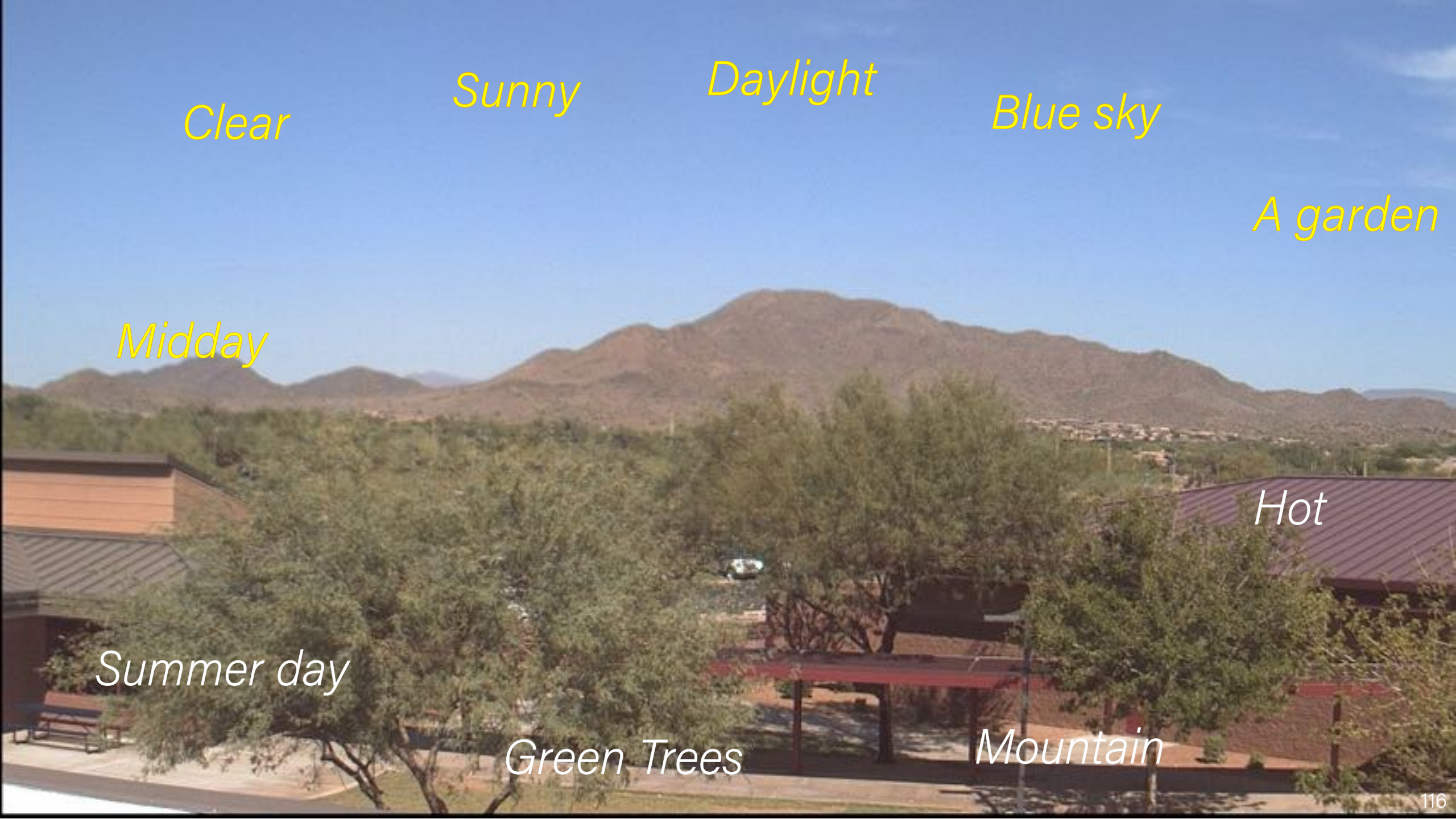


Progressive GANs (Karras et al., 2017)

- Allows to obtain high resolution synthetic images

Image Editing with GANs

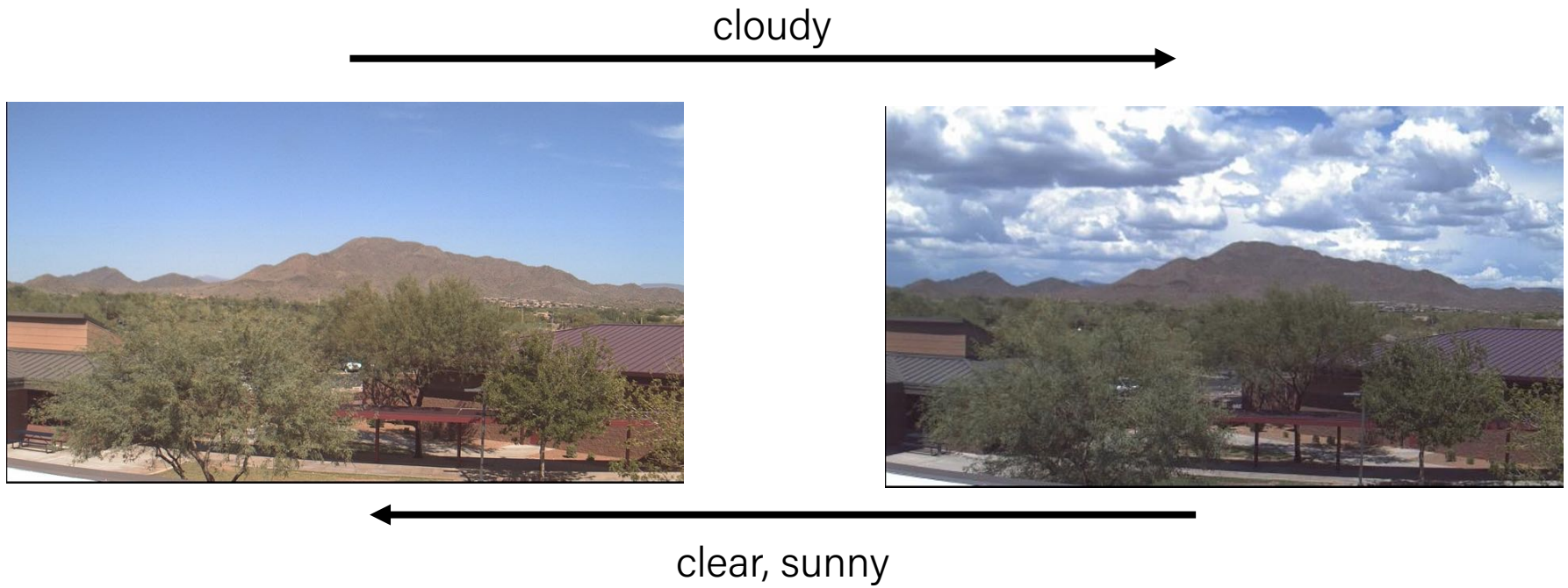

“Maybe in our world lives a
happy little tree over there.”
— Bob Ross


Clear
Sunny
Daylight
Blue sky
A garden
Midday
Hot
Summer day
Green Trees
Mountain

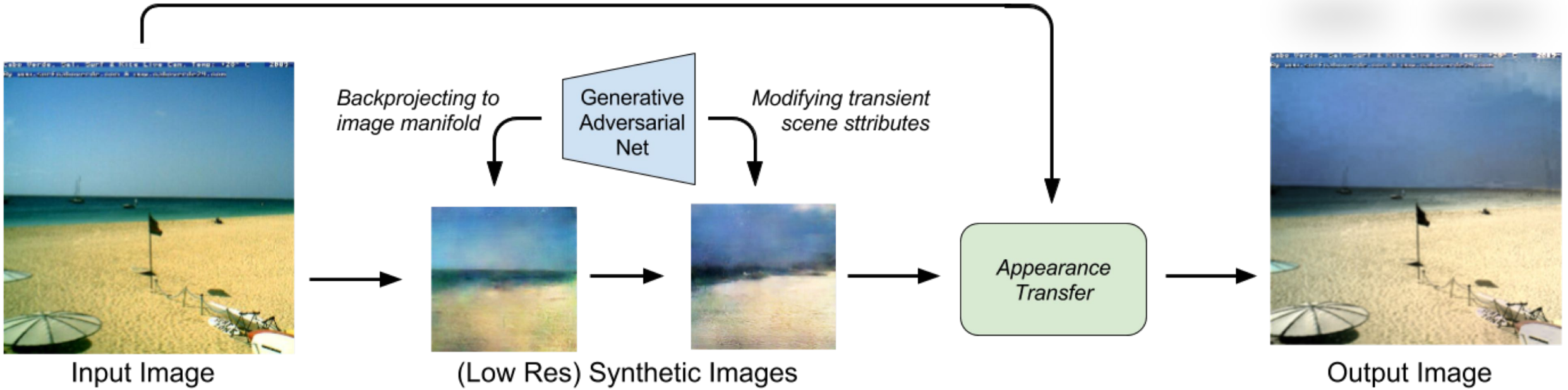

Cloudy
Sunny
Daylight
Blue sky
A garden
Midday
Cool
Summer day
Green Trees
Mountain

Natural Scenes

- Can we change the appearance of an outdoor scene using visual attributes?
  - Leverage the big visual data
  - Learn visual attributes
  - Change the attributes



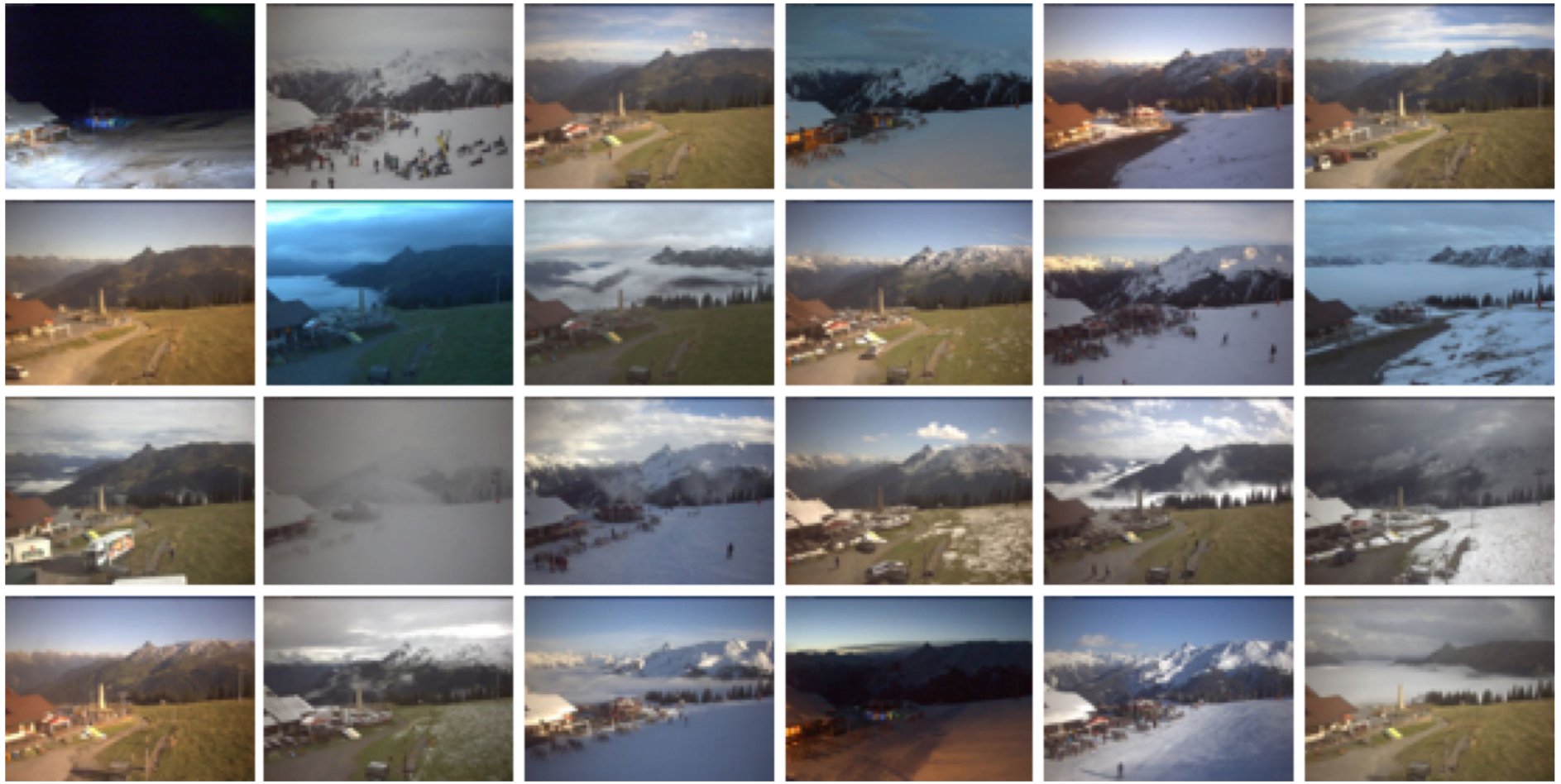
Our Approach



- A conditional GAN-based model (*conditions: transient scene attributes*)
  - Backprojecting input image on noise space (Zhu et al., 2006)
  - Modify scene attributes
  - Appearance transfer

Levent Karacan, Aykut Erdem, Erkut Erdem, "Adjusting Transient Attributes of Outdoor Images using Generative Adversarial Networks", SIU 2017

Transient Attributes Dataset



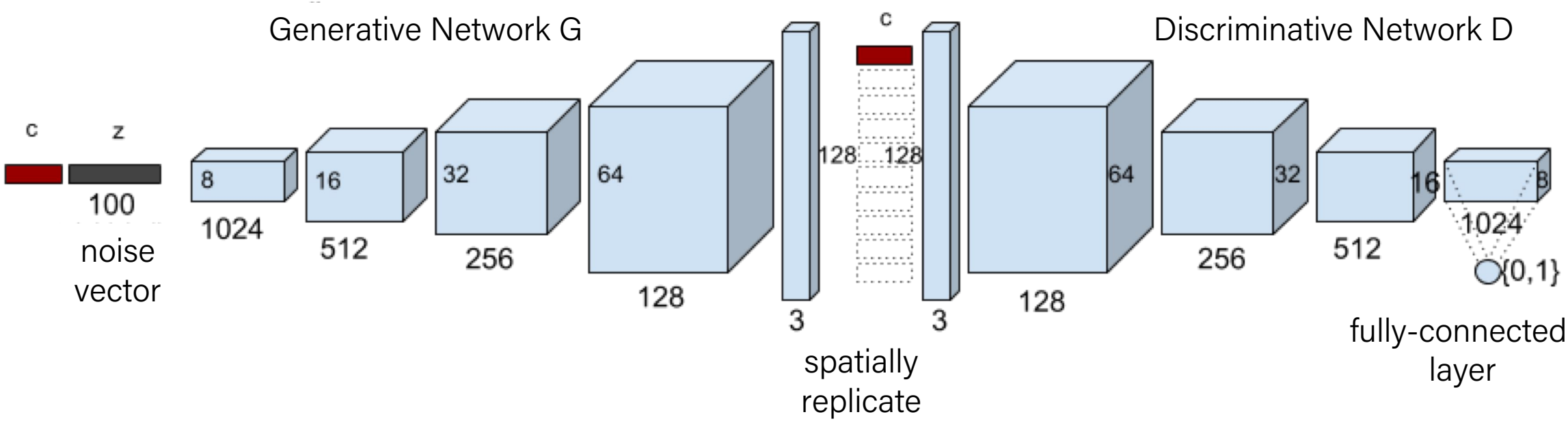
- 101 webcams 8571 outdoor scenes
- 40 transient attribute for each image

P.-Y. Laffont, Z. Ren, X. Tao, C. Qian, and J. Hays, "Transient attributes for high-level understanding and editing of outdoor scenes," ACM Transactions on Graphics, vol. 33, no. 4, 2014

Proposed conditioned GAN model

$$\mathcal{L}_{GAN}(G,D) = \mathbb{E}_{x,c\sim p_{data}(x,c)}[\log D(x,c)] + \mathbb{E}_{x,c\sim p_{data}(x,c),z\sim p_z(z)}[\log(1 - D(G(z,c),c))]$$

$$G^* = \min_G \max_D \mathcal{L}_{cGAN}(G,D)$$



Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **night** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Increasing **sunset** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

Decreasing **snow** attribute

Generated samples from learned image manifold

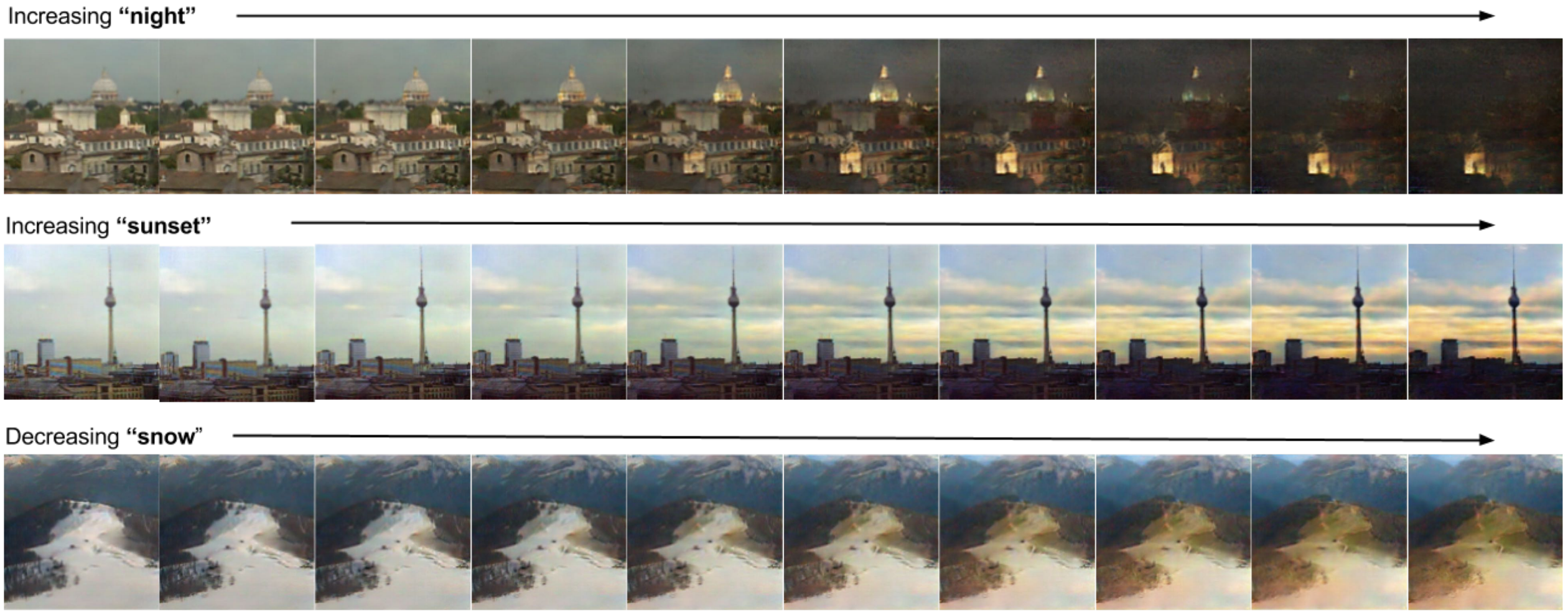
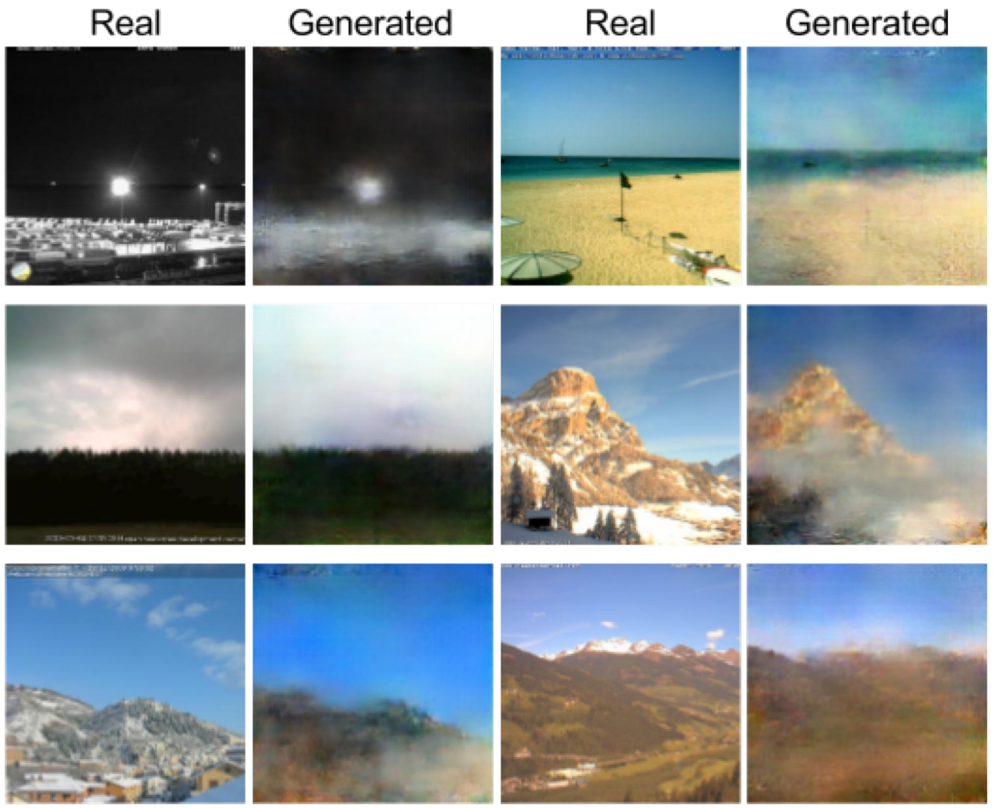


Image editing depending on transient attributes

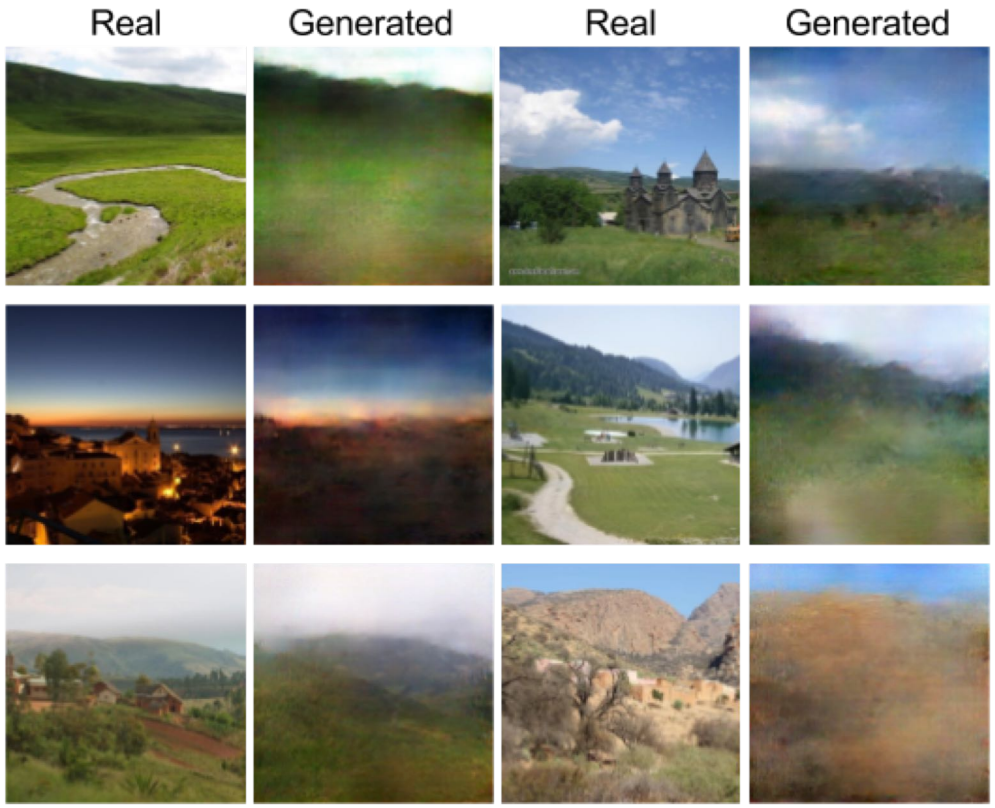
- Projection with L-BFGS-B

$$\mathcal{L}(x_1, x_2) = \|C(x_1) - C(x_2)\|^2$$

$$(z^*, c^*) = arg \min_{c \in \mathbb{C}, z \in \tilde{\mathbb{Z}}} \mathcal{L}(G(z, c), x^R)$$

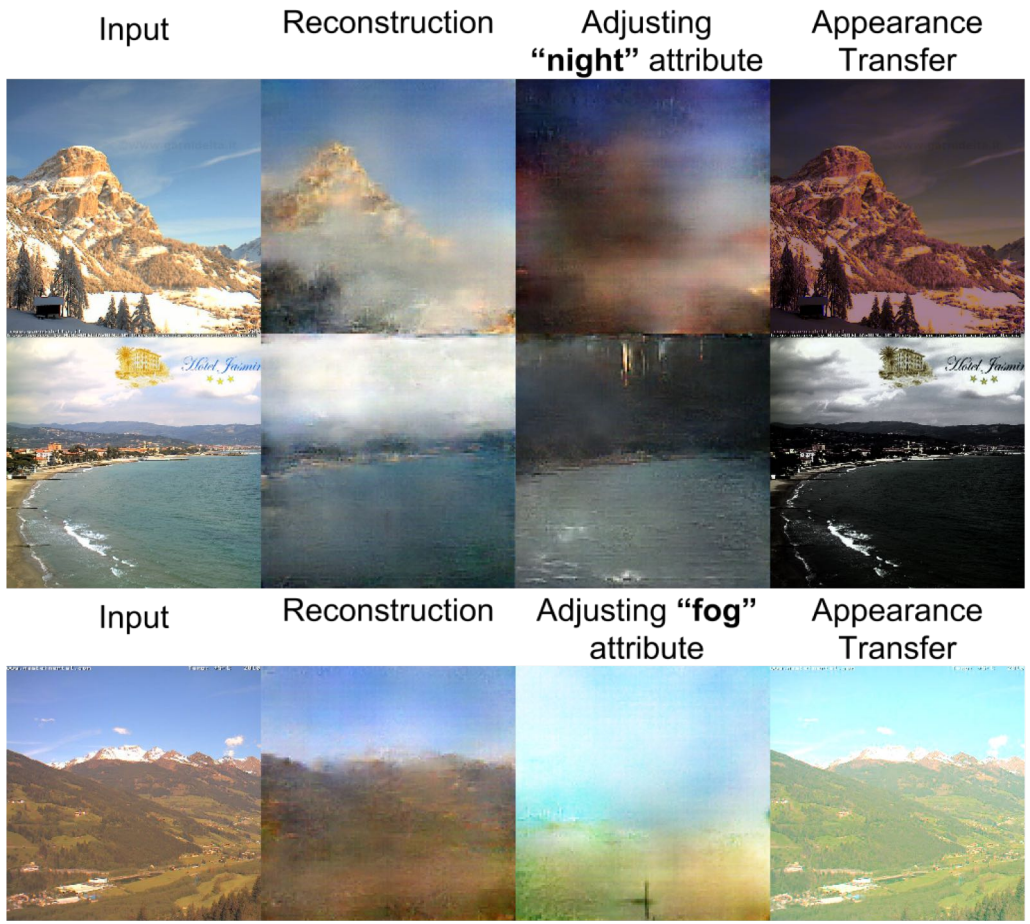


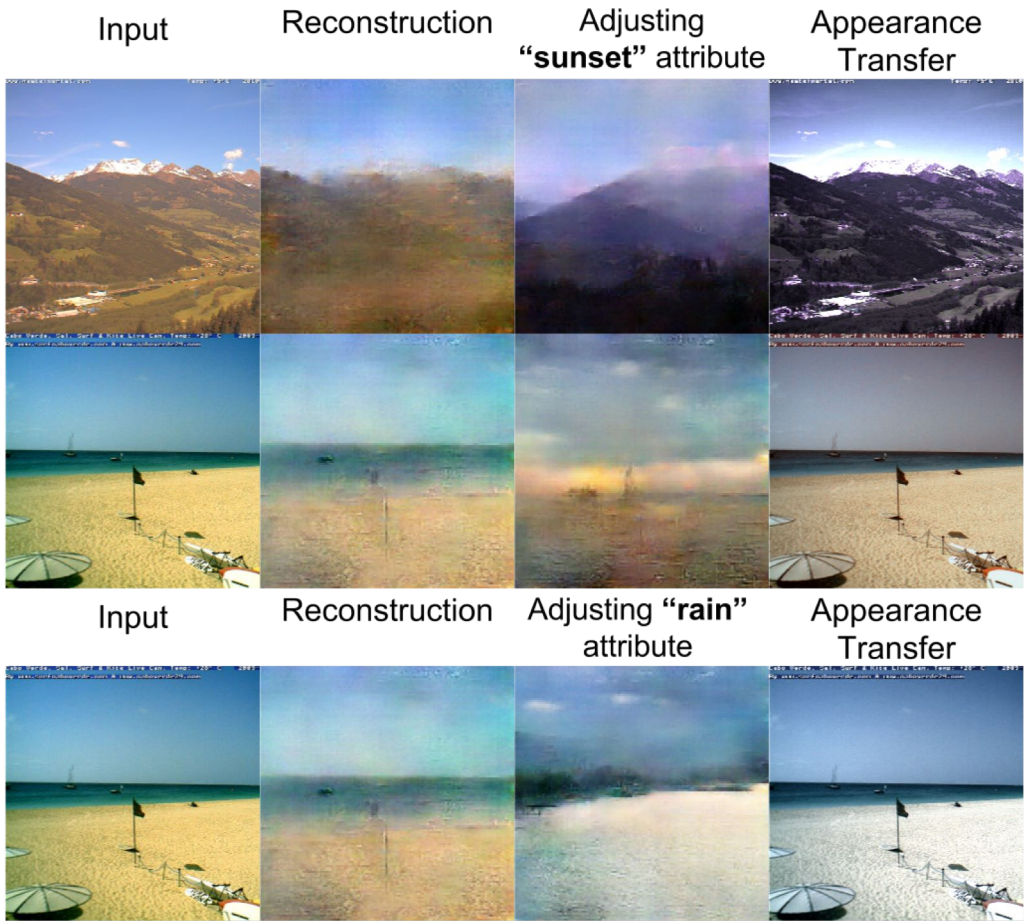
Training set samples



Test set samples

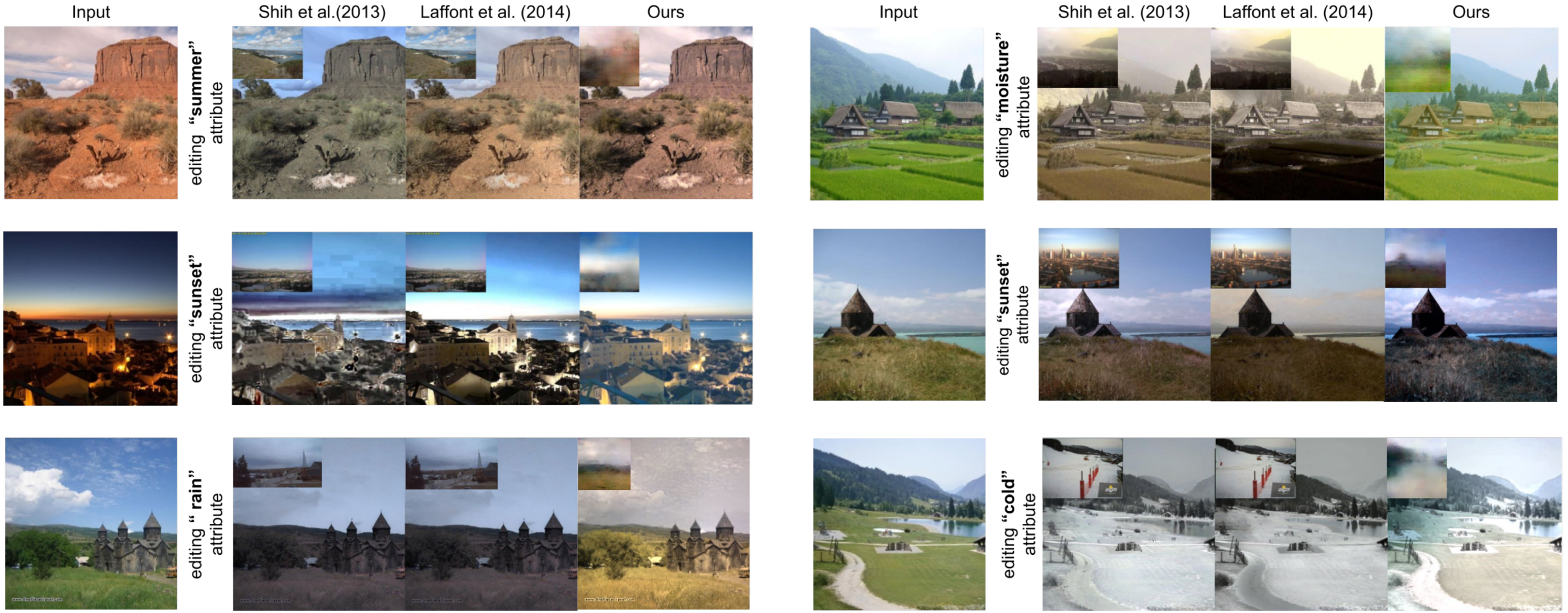
Appearance Transfer (Shih et al., 2013)



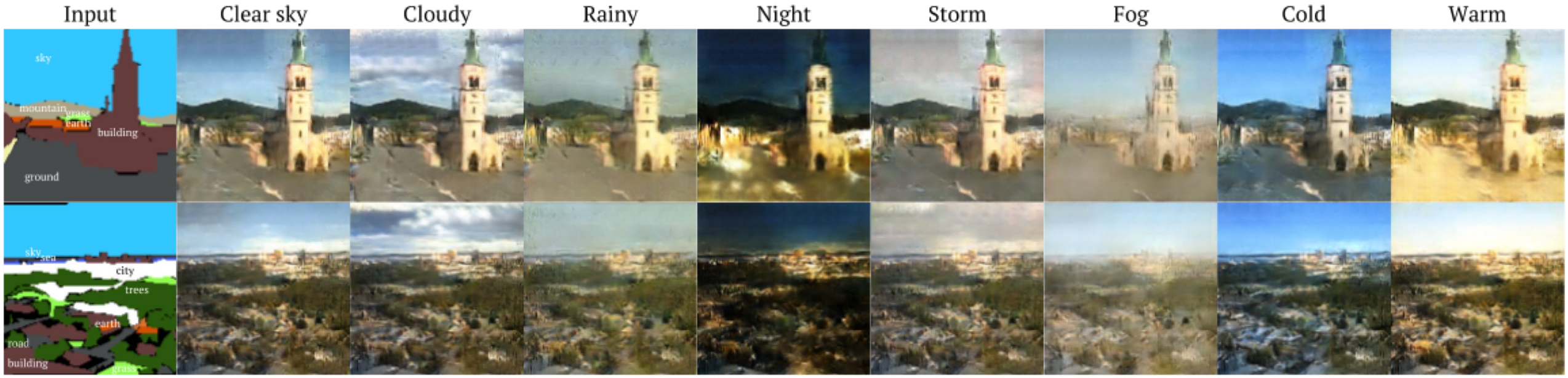


- Appearance transfer from generated image to original image.
- Sampling based local affine model in color space.
- Affine models by least square optimization are used to transfer appearance.

Comparison with Related Works

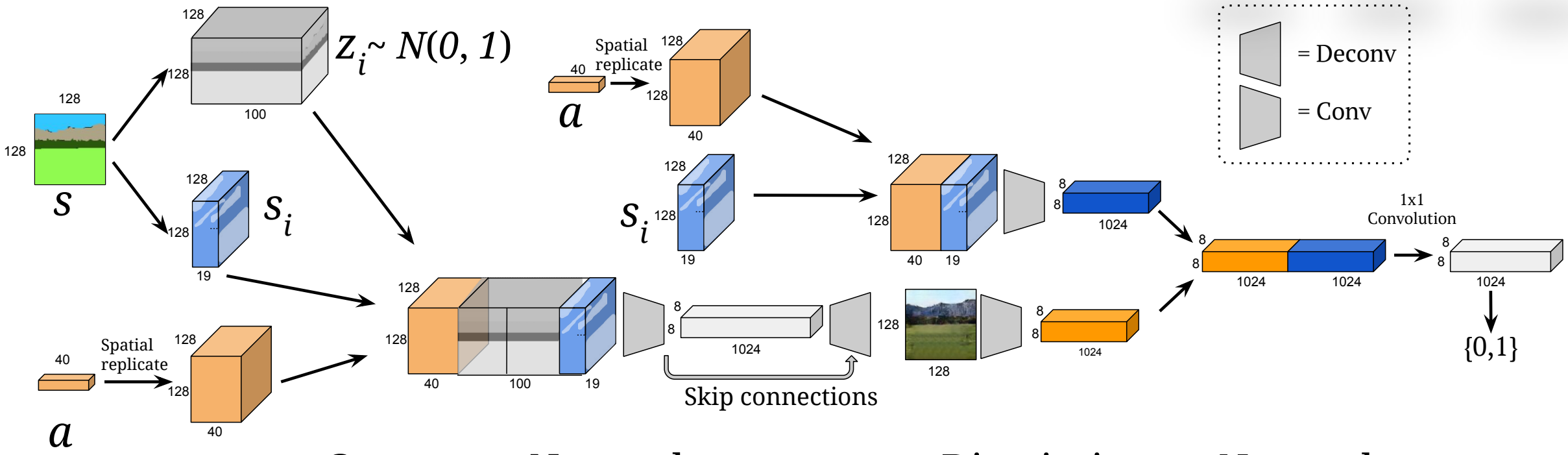


Generating Outdoor Scenes under Various Conditions



Levent Karacan, Zeynep Akata, Aykut Erdem, Erkut Erdem, “Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts." arXiv preprint arXiv:1612.00215 (2016)

Generating Outdoor Scenes under Various Conditions



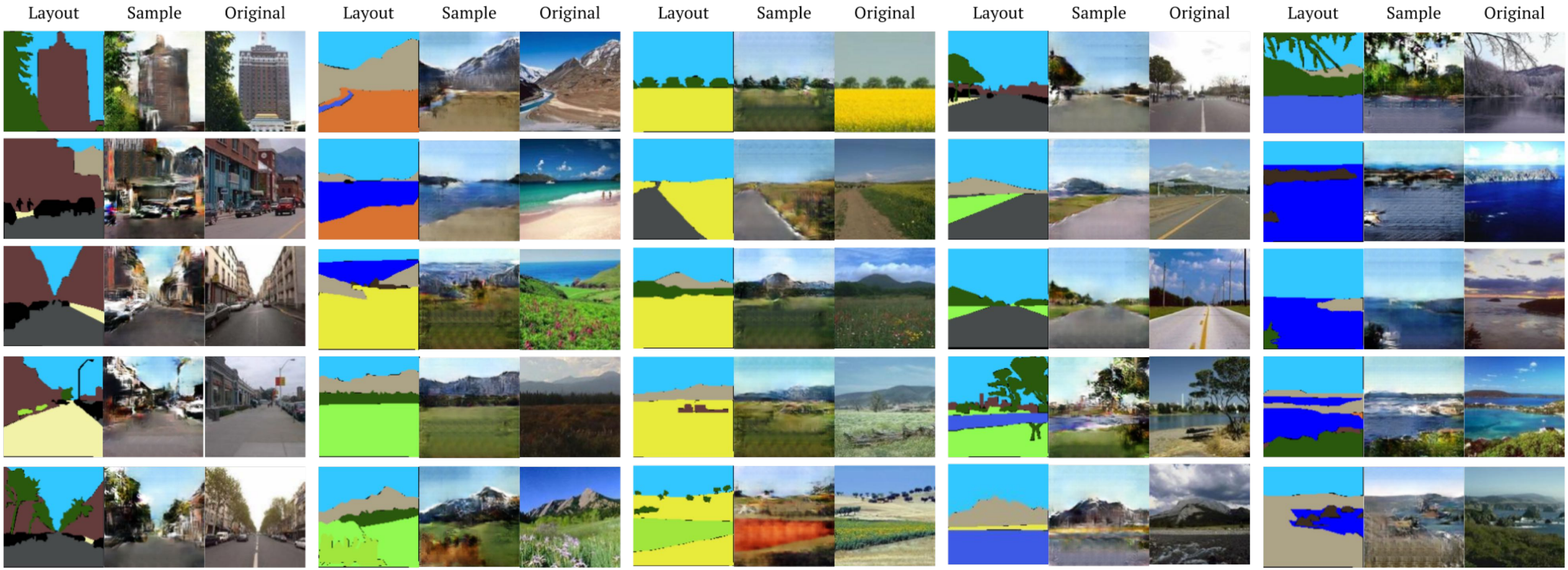
Generator Network Discriminator Network

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,s,a\sim p_{data}(x,s,a)}[\log D(x, s, a)] + \mathbb{E}_{s,a\sim p_{data}(s,a),z\sim p_z(z)}[\log(1 - D(x, G(z, s, a)))]$$

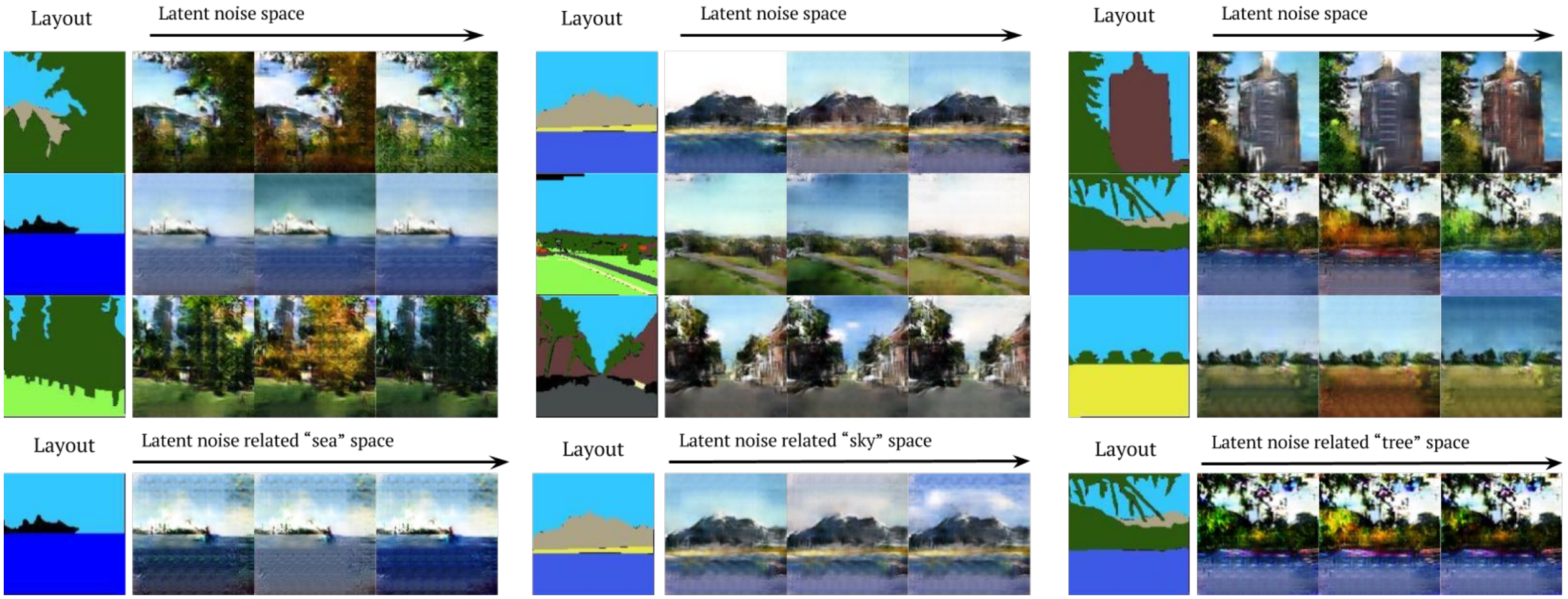
$$\min_G \max_D \mathcal{L}_{cGAN}(G, D)$$

- The noise vectors z are specific to the semantic layout.
- This provides the diversity in generated samples.

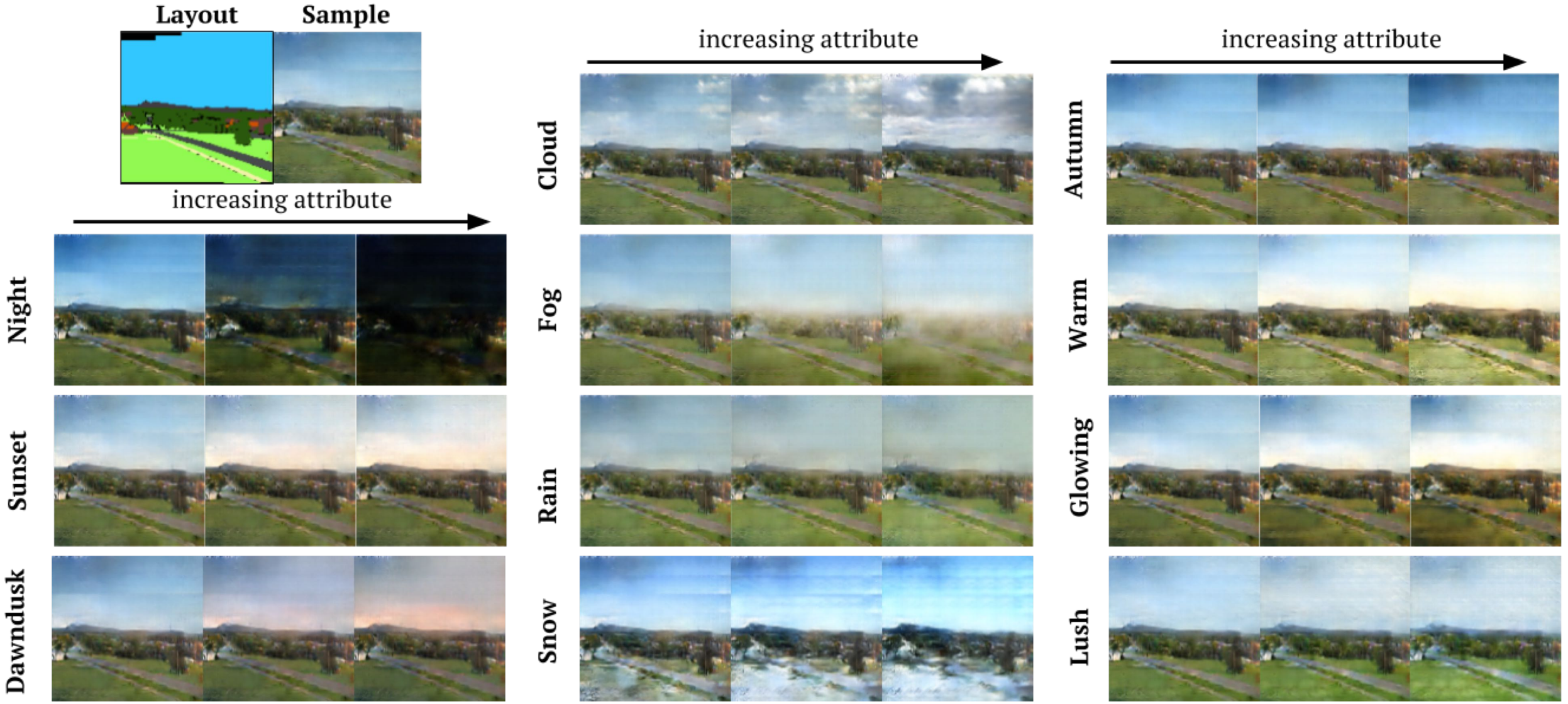
Generated images from given scene layouts



Diversity in samples – Playing with the noise



Diversity in samples – Playing with attributes



AL-CGAN vs pix2pix

Summary

- Why Generative Models
- Types of Generative Models
  - Autoregressive Generative Models
  - Latent Variable Models
  - Transformation Models
- Image Editing with GANs