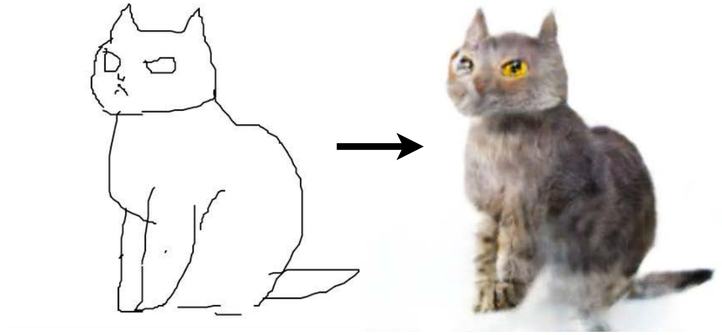
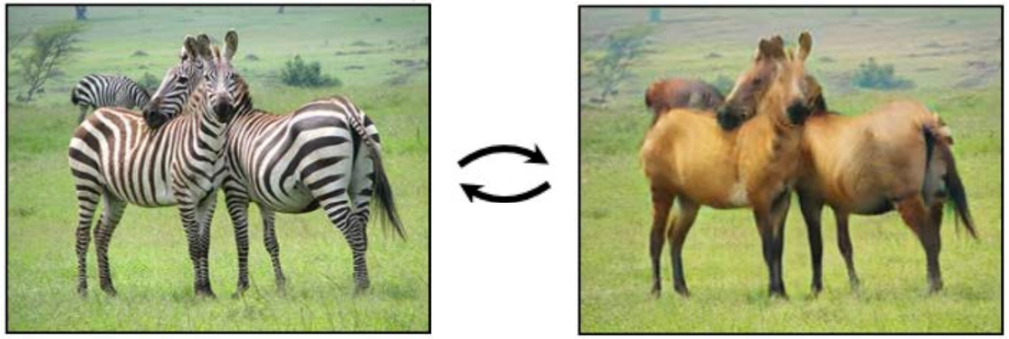
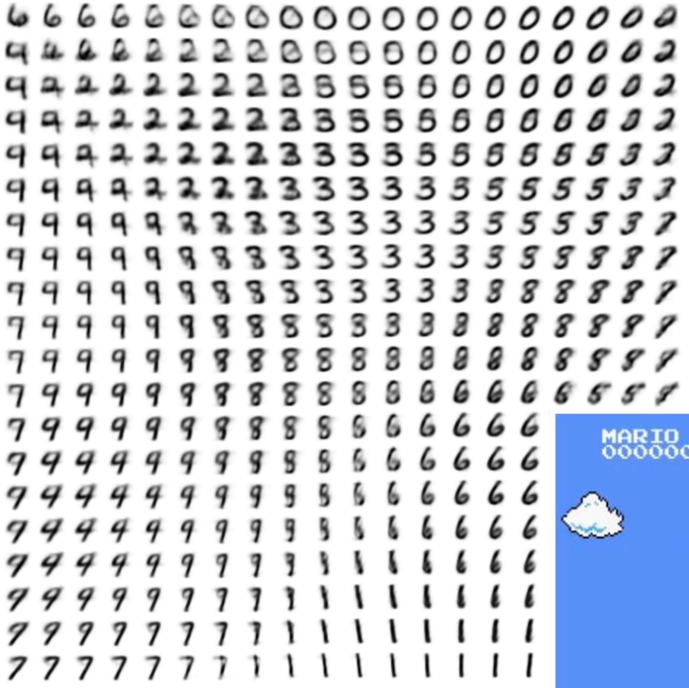






Generative Models II

Phillip Isola, MIT, OpenAI

DLSS

7/27/18

What's a generative model?

For this talk: models that output high-dimensional data

(Or, anything involving a GAN, VAE, PixelCNN, etc)

Useful for lots of problems beyond density estimation and sampling random images!

What can you do with generative models?

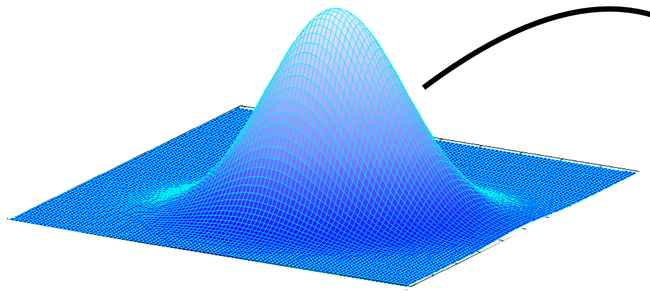
1. Data prediction
2. Domain mapping
3. Representation learning
4. Model-based intelligence

What can you do with generative models?

1. **Data prediction**
2. Domain mapping
3. Representation learning
4. Model-based intelligence

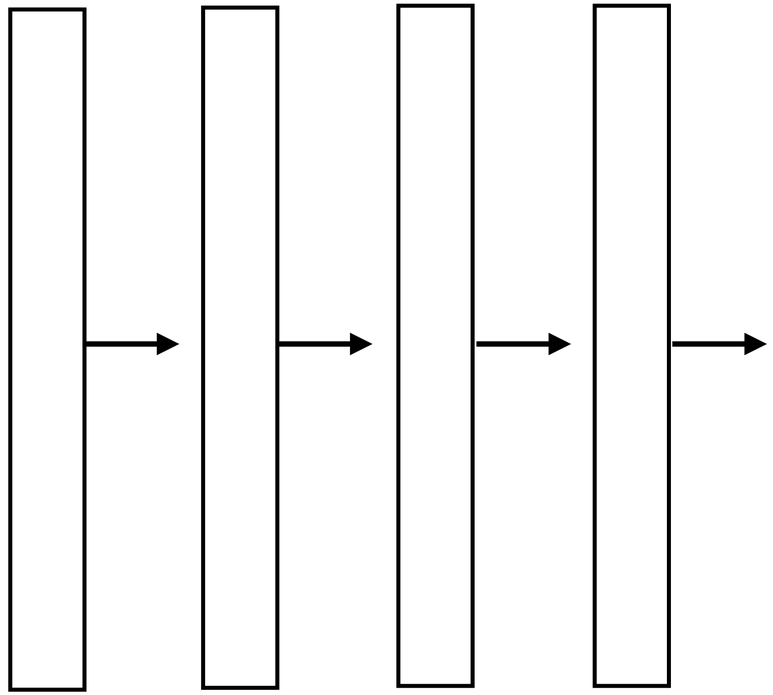
Generative Model

$\mathbf{z}$



Gaussian noise

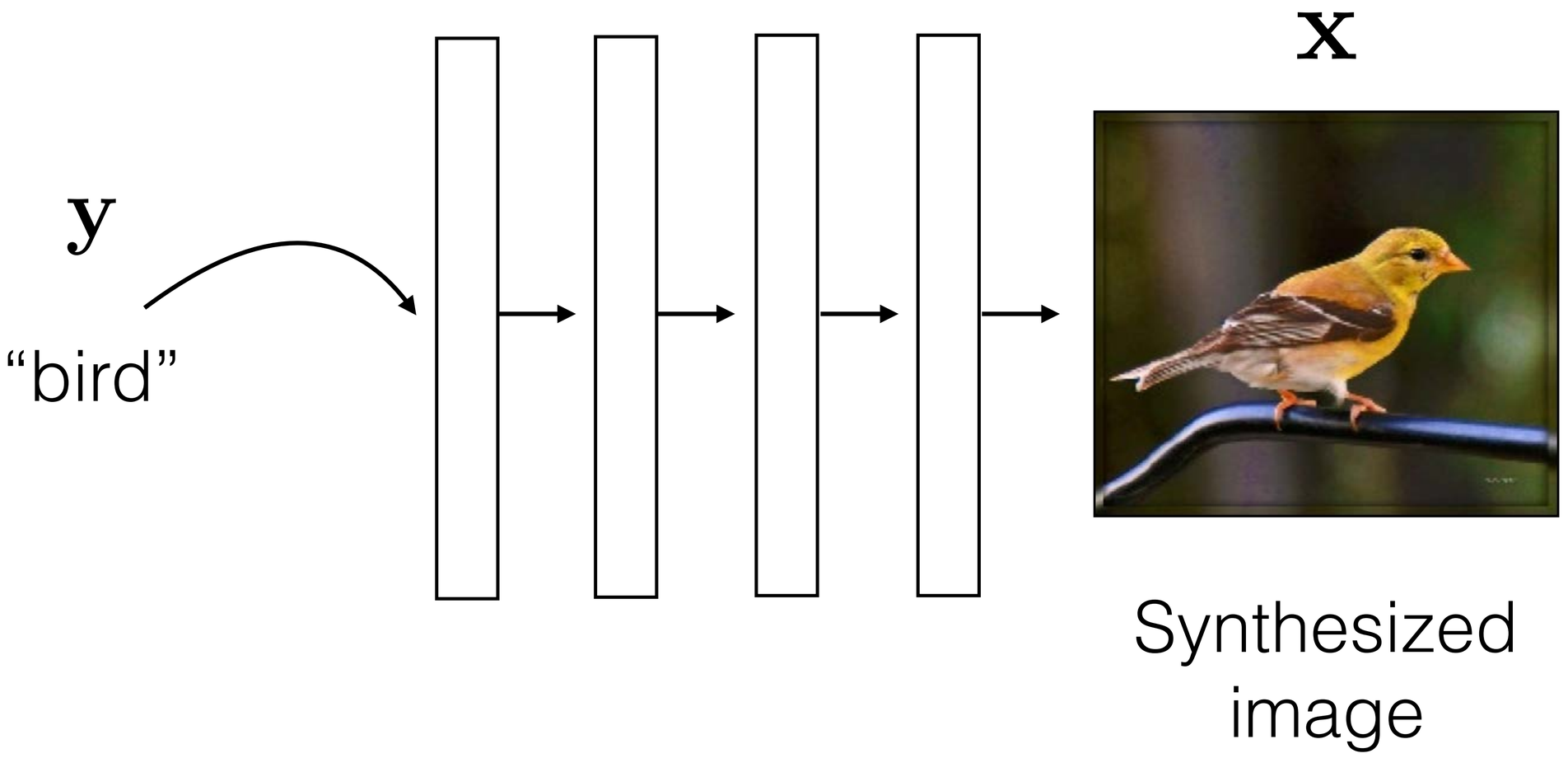
$z \sim \mathcal{N}(\vec{0}, 1)$



$\mathbf{x}$

Synthesized image

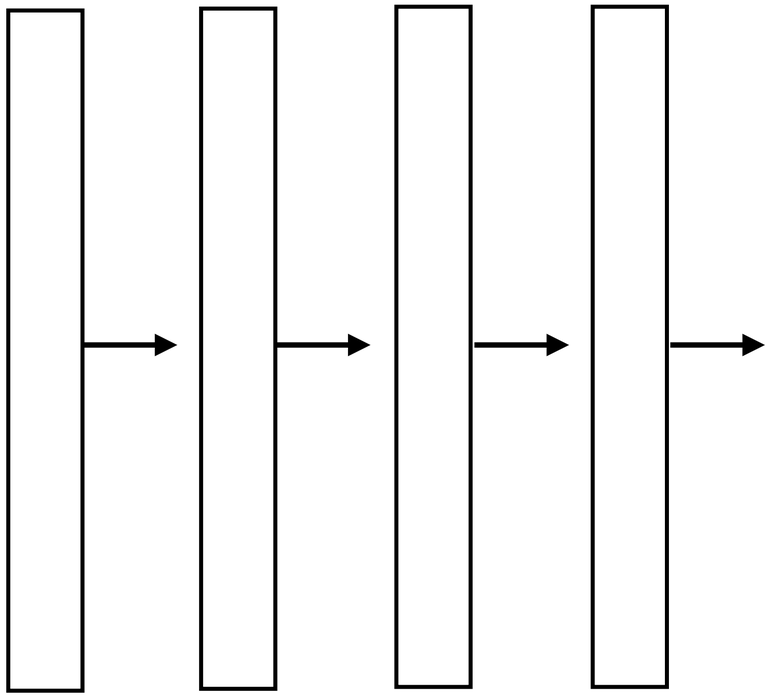
Conditional Generative Model



Conditional Generative Model

$\mathbf{y}$

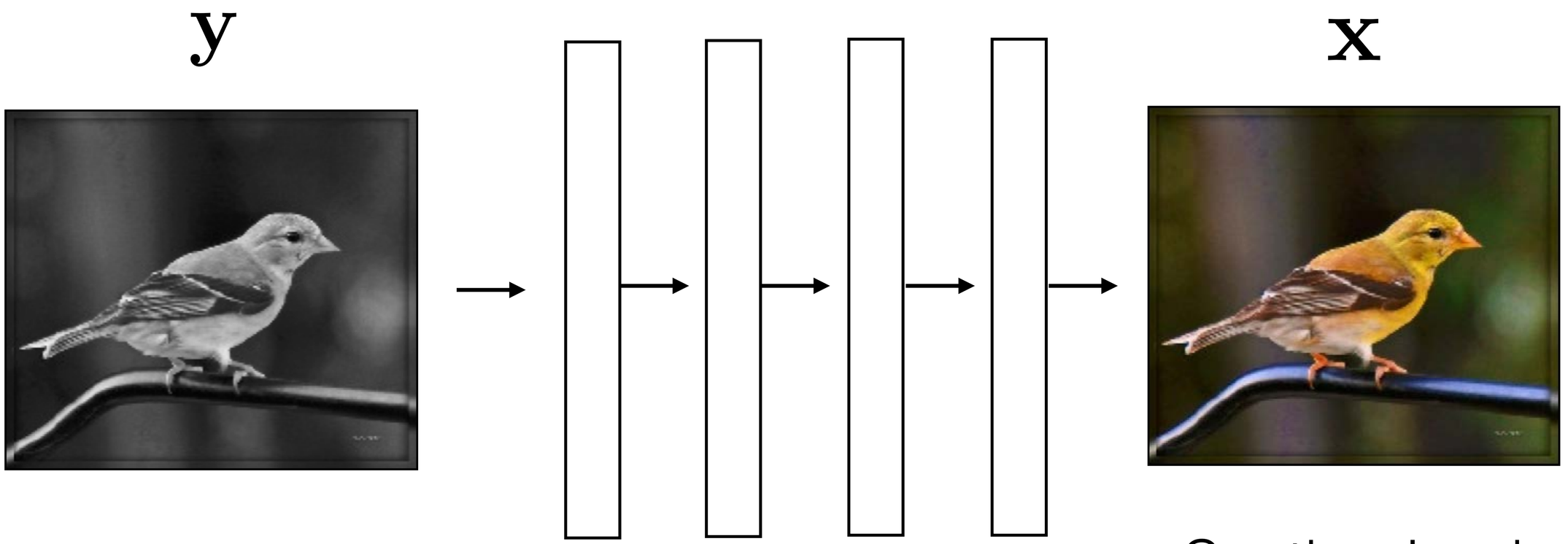
"A yellow bird on a branch"



$\mathbf{x}$

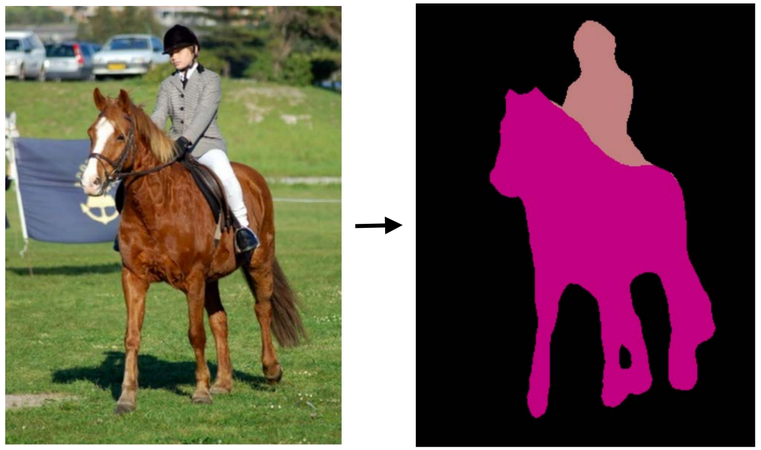
Synthesized image

Conditional Generative Model



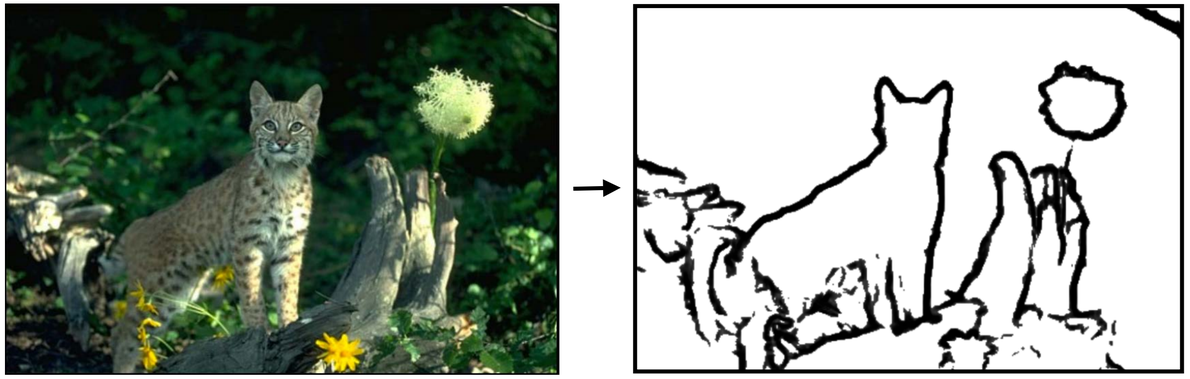
Data prediction problems ("structured prediction")

Object labeling



[Long et al. 2015, …]

Edge Detection



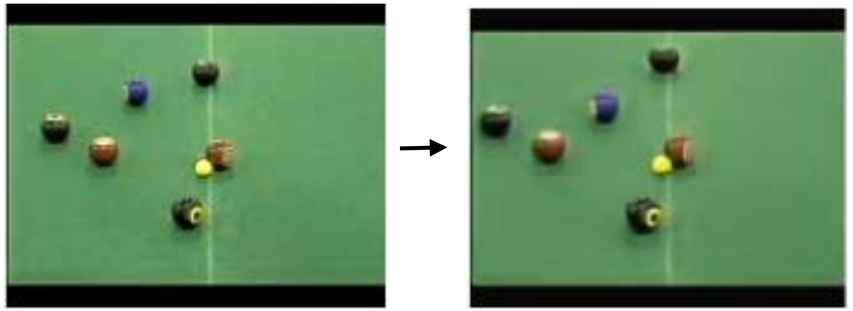
[Xie et al. 2015, …]

Text-to-photo

"this small bird has a pink breast and crown…"

[Reed et al. 2014, …]

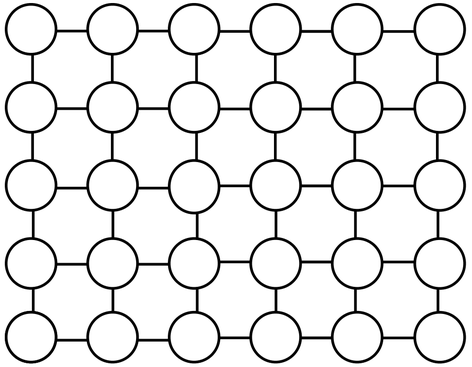
Future frame prediction



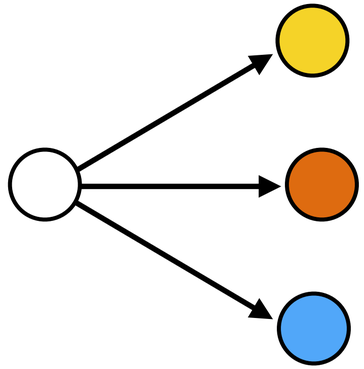
[Mathieu et al. 2016, …]

Challenges in data prediction

1. Output is a high-dimensional, structured objects

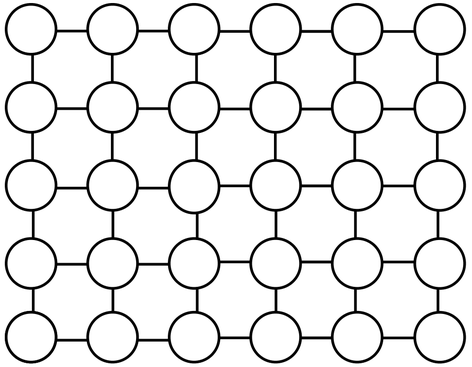


2. Uncertainty in the mapping, many plausible outputs

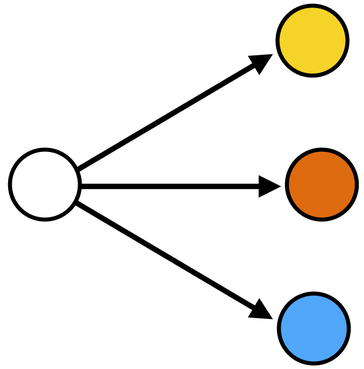


Properties of generative models

1. Model high-dimensional, structured objects



2. Model uncertainty; a whole distribution of possible outputs



Conditional Generative Models

Two ways to do it:

1. Model p(x,y), then do inference to get conditional p(y|x)

$$\arg\max_y p(x, y) = \arg\max_y p(y|x)$$

2. Directly model p(y|x)

Image-to-Image Translation

Input $\mathbf{x}$

Output $\mathbf{y}$

$\mathcal{F}$

Training data

$\mathbf{x}$ $\mathbf{y}$

$$\arg\min_{\mathcal{F}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[L(\mathcal{F}(\mathbf{x}), \mathbf{y})]$$

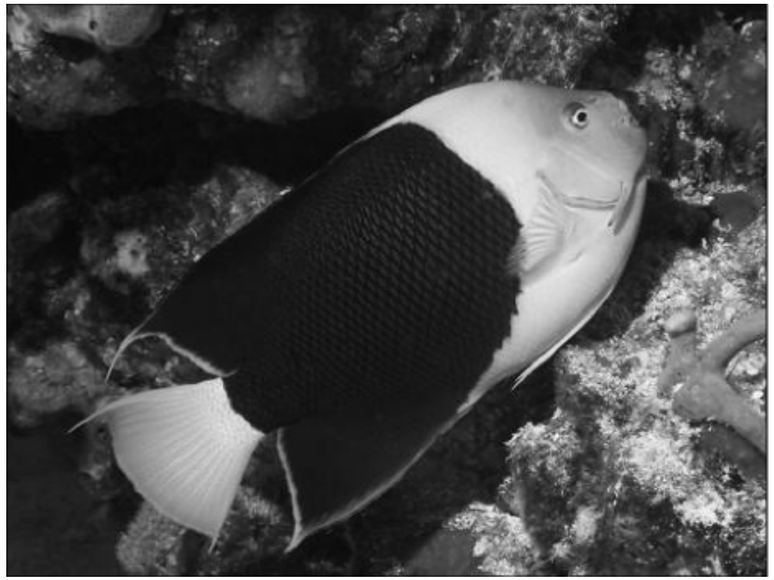
Objective function (loss)

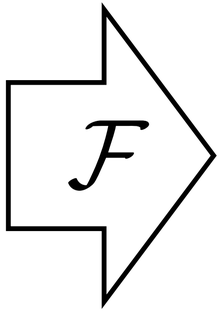
Neural Network

[Zhang et al., ECCV 2016]

Image-to-Image Translation

Input $\mathbf{x}$





Output $\mathbf{y}$

$$\arg\min_{\mathcal{F}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[L(\mathcal{F}(\mathbf{x}), \mathbf{y})]$$

"**What** should I do"

"**How** should I do it?"

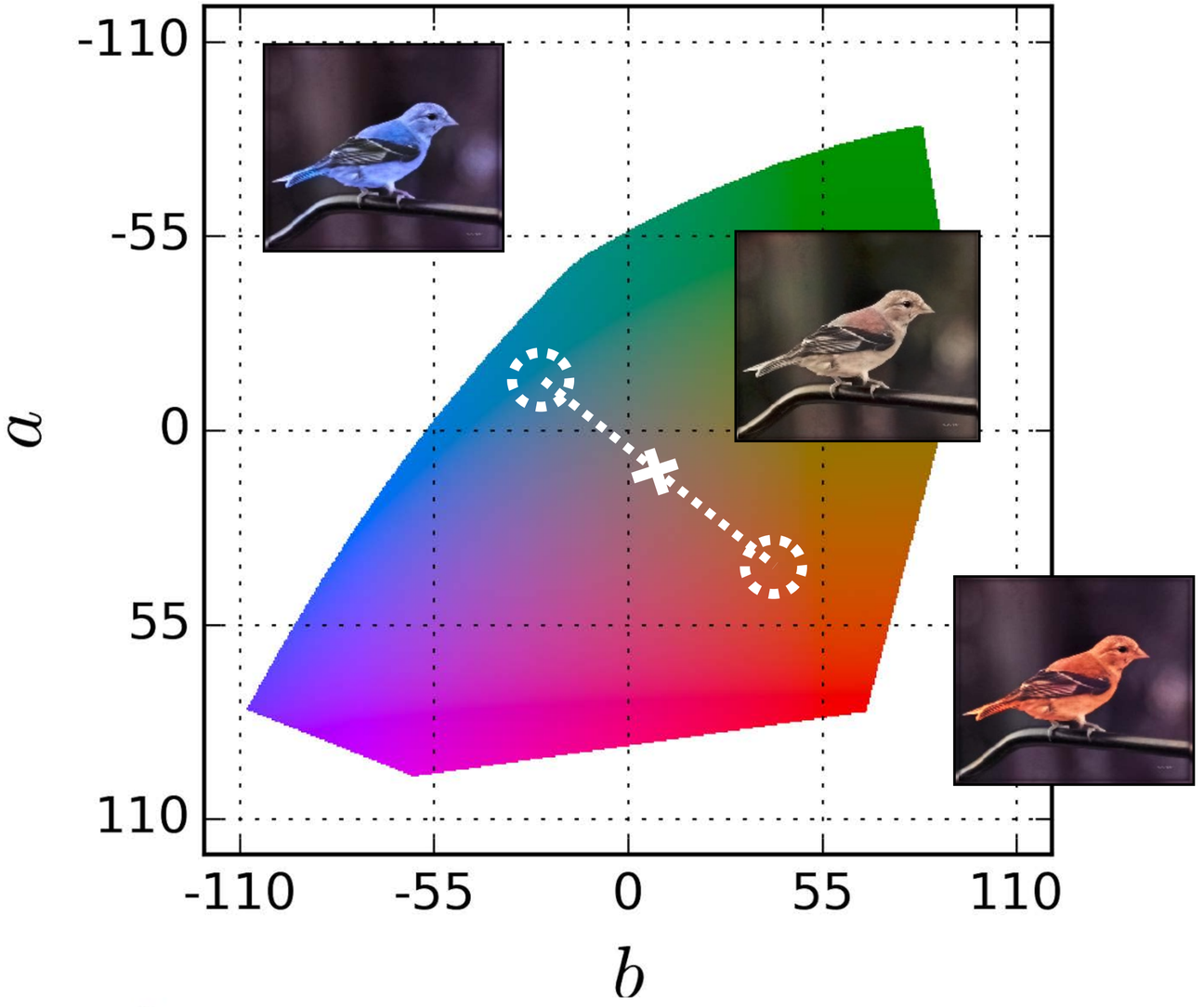
Designing loss functions

Input

Output

Ground truth

$$\mathrm{L}_2(\widehat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2}\sum_{h,w} \|\mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w}\|_2^2$$



$$\mathrm{L}_2(\widehat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w}\|_2^2$$

Designing loss functions

Input

Zhang et al. 2016

Ground truth

Color distribution cross-entropy loss with colorfulness enhancing term.

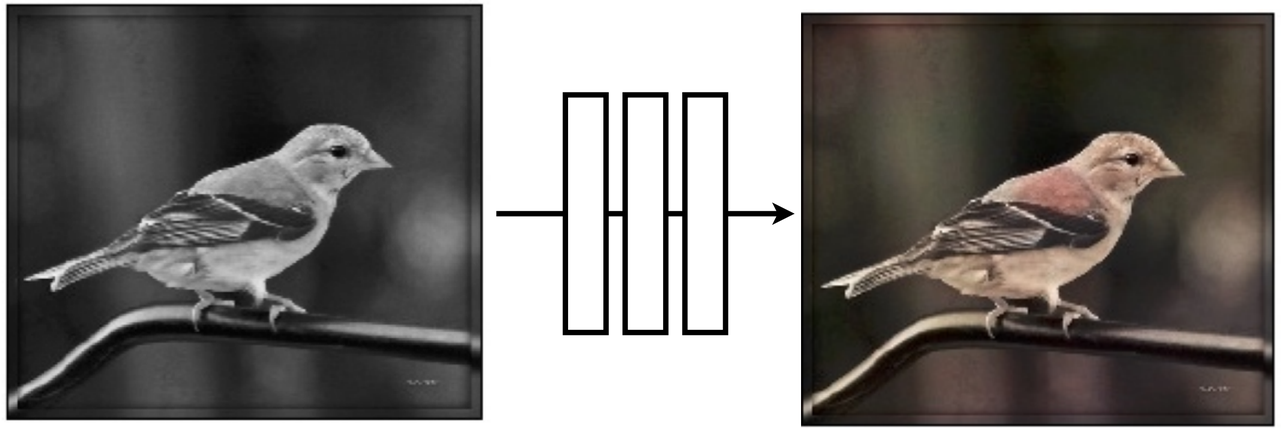
DADS

Designing loss functions

Be careful what you wish for!

Designing loss functions

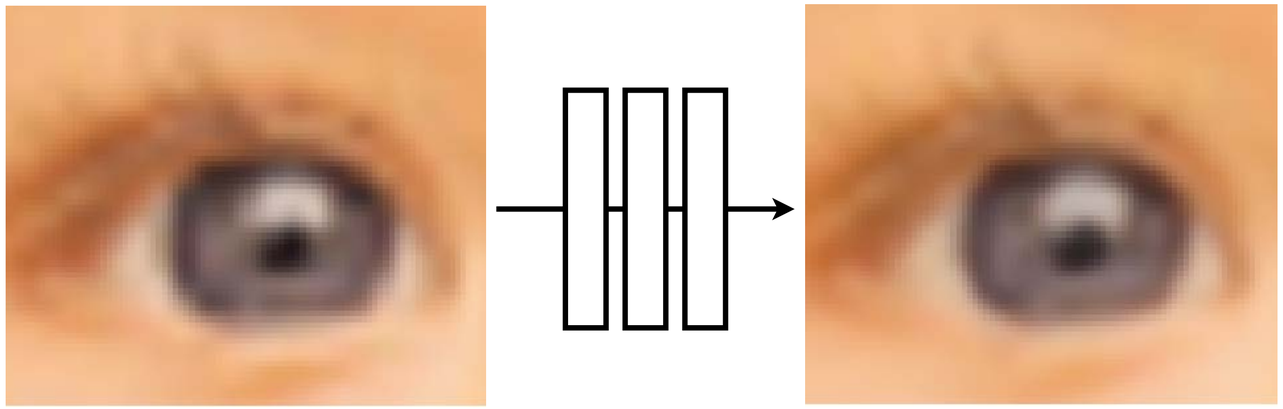
Image colorization



[Zhang, Isola, Efros, ECCV 2016]

L2 regression

Super-resolution

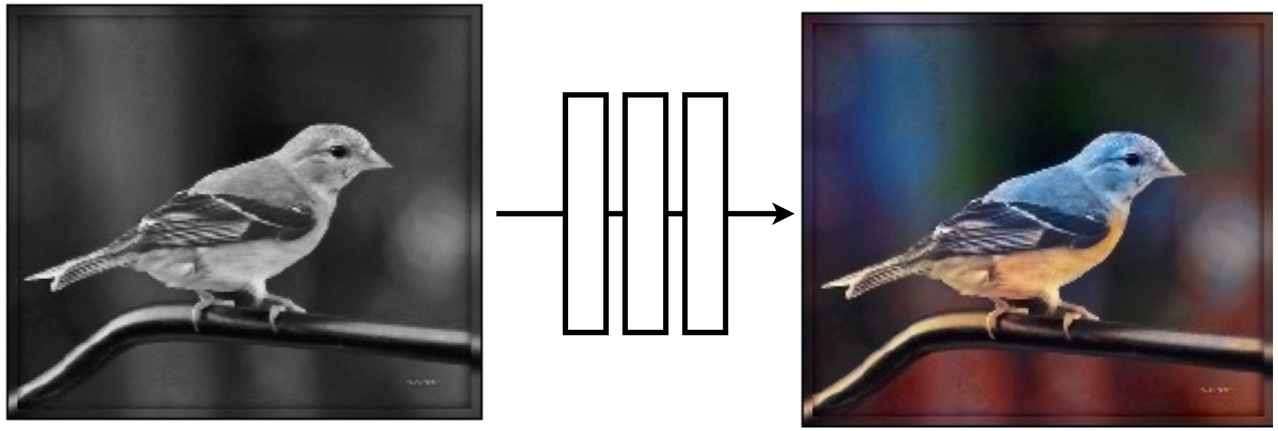


[Johnson, Alahi, Li, ECCV 2016]

L2 regression

Designing loss functions

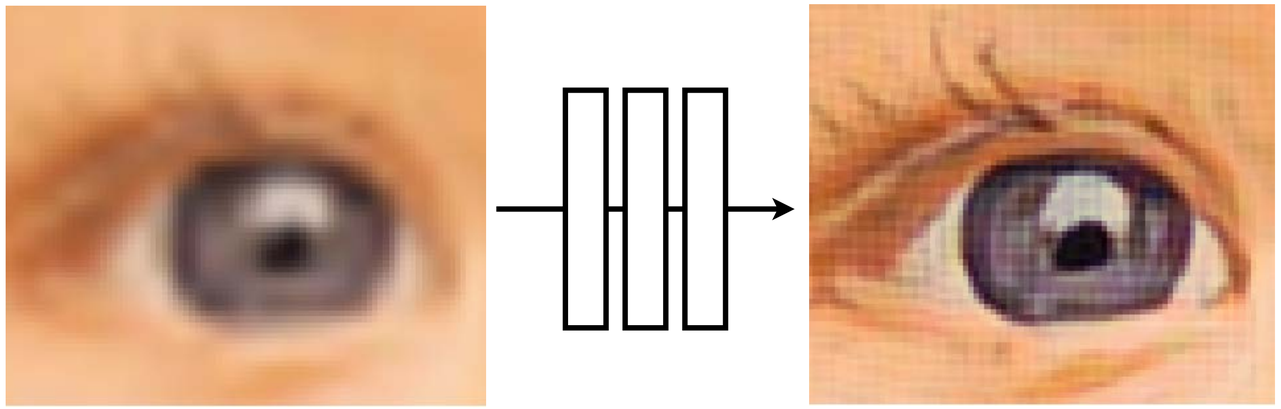
Image colorization



[Zhang, Isola, Efros, ECCV 2016]

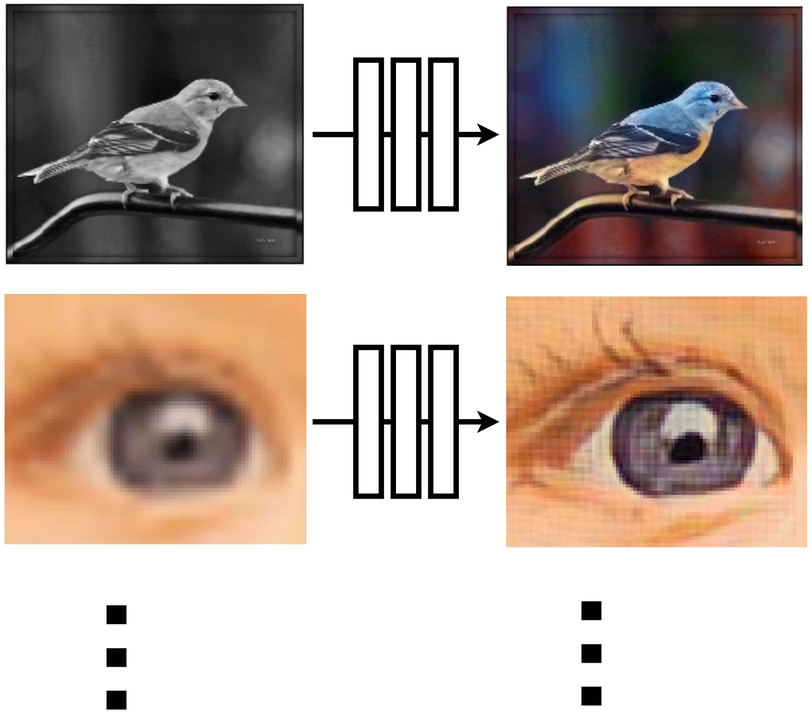
Cross entropy objective,
with colorfulness term

Super-resolution



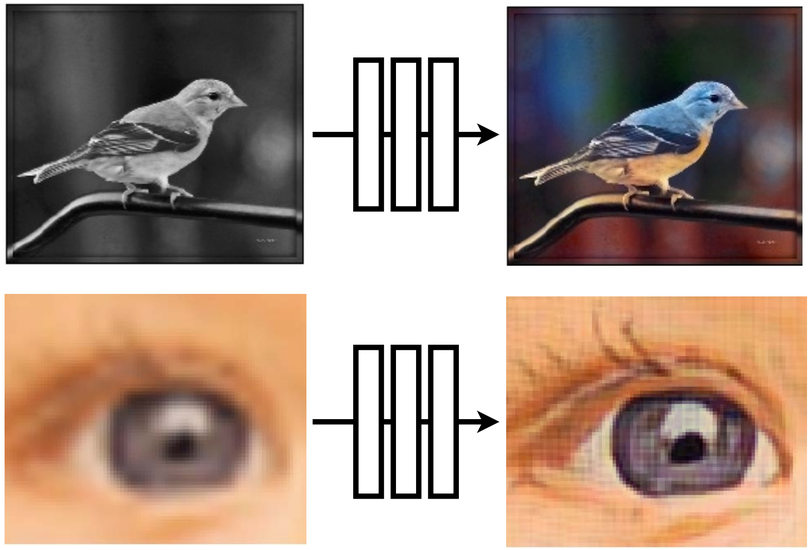
[Johnson, Alahi, Li, ECCV 2016]

Deep feature covariance
matching objective



Universal loss?

Generated images



“Generative Adversarial Network” (GANs)

Generated vs Real (classifier)

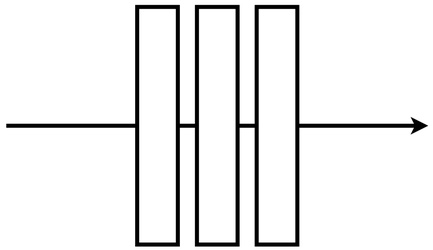
Real photos

...

[Goodfellow, Pouget-Abadie, Mirza, Xu, Warde-Farley, Ozair, Courville, Bengio 2014]

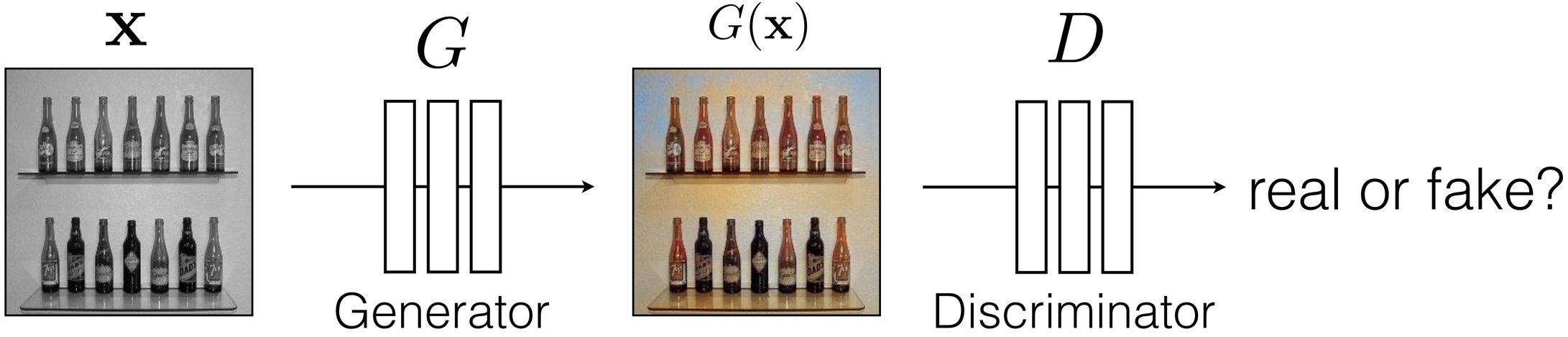
$\mathbf{x}$

G



Generator

$G(\mathbf{x})$



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

D tries to identify the fakes

$\mathbf{x}$

G

$G(\mathbf{x})$

D

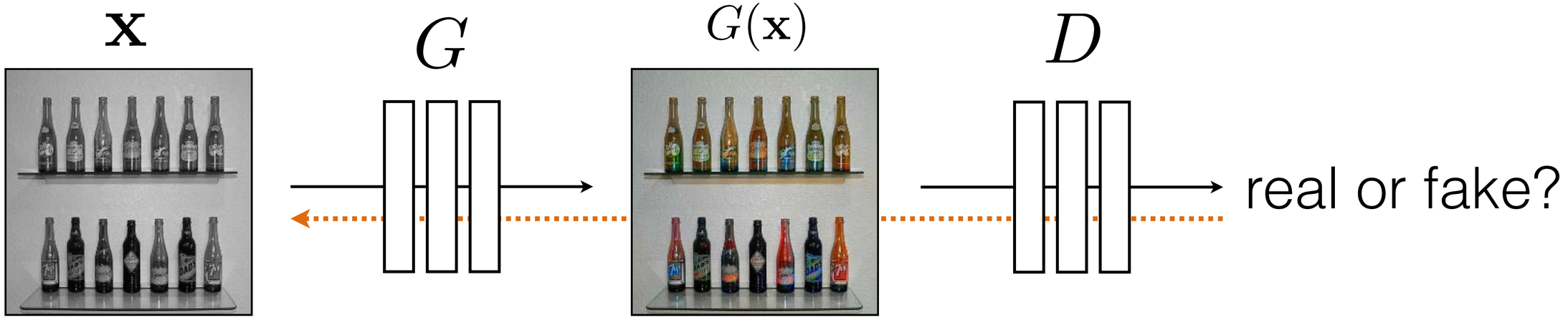
fake (0.9)

$\mathbf{y}$

D

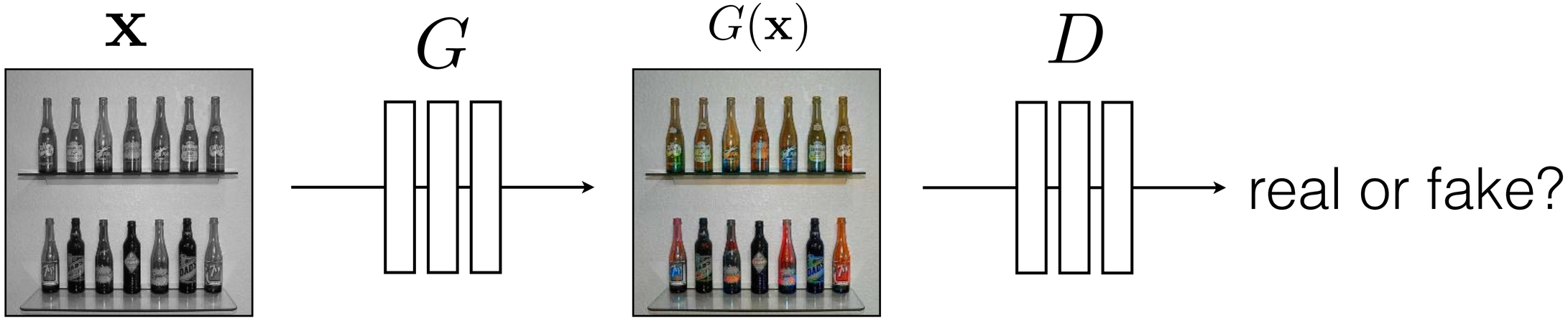
real (0.1)

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



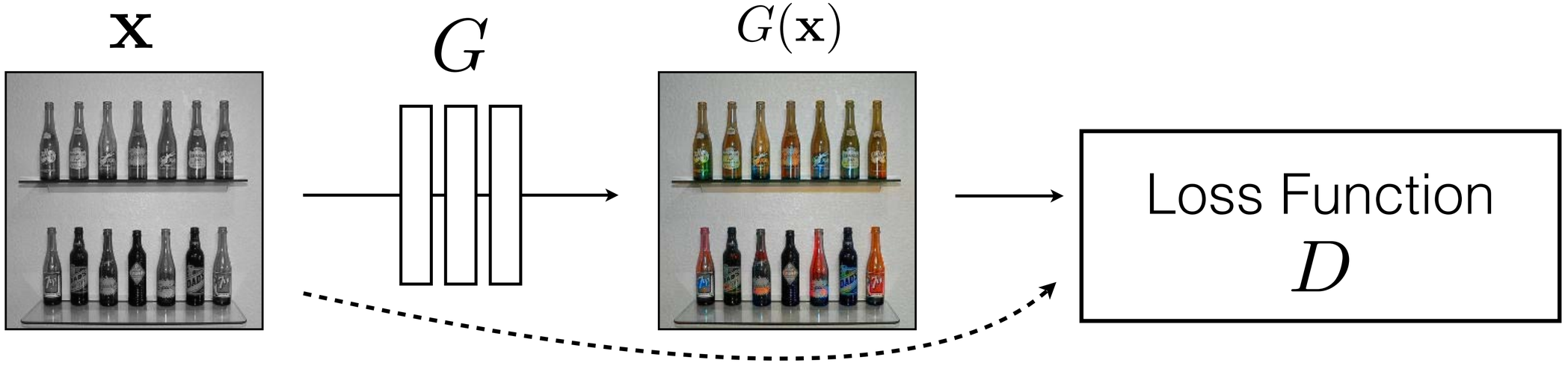
G tries to synthesize fake images that ***fool*** **D**:

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



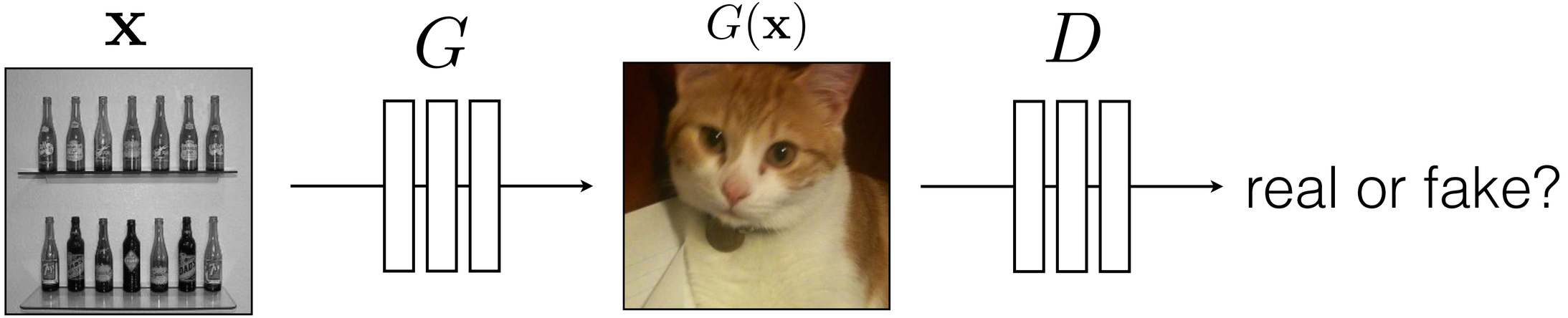
G tries to synthesize fake images that ***fool*** the ***best*** **D**:

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

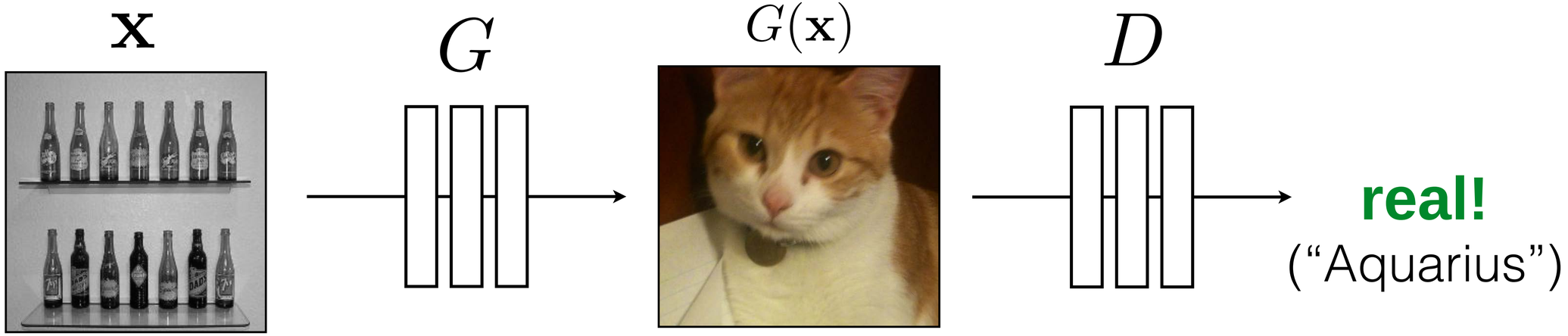


G's perspective: **D** is a loss function.

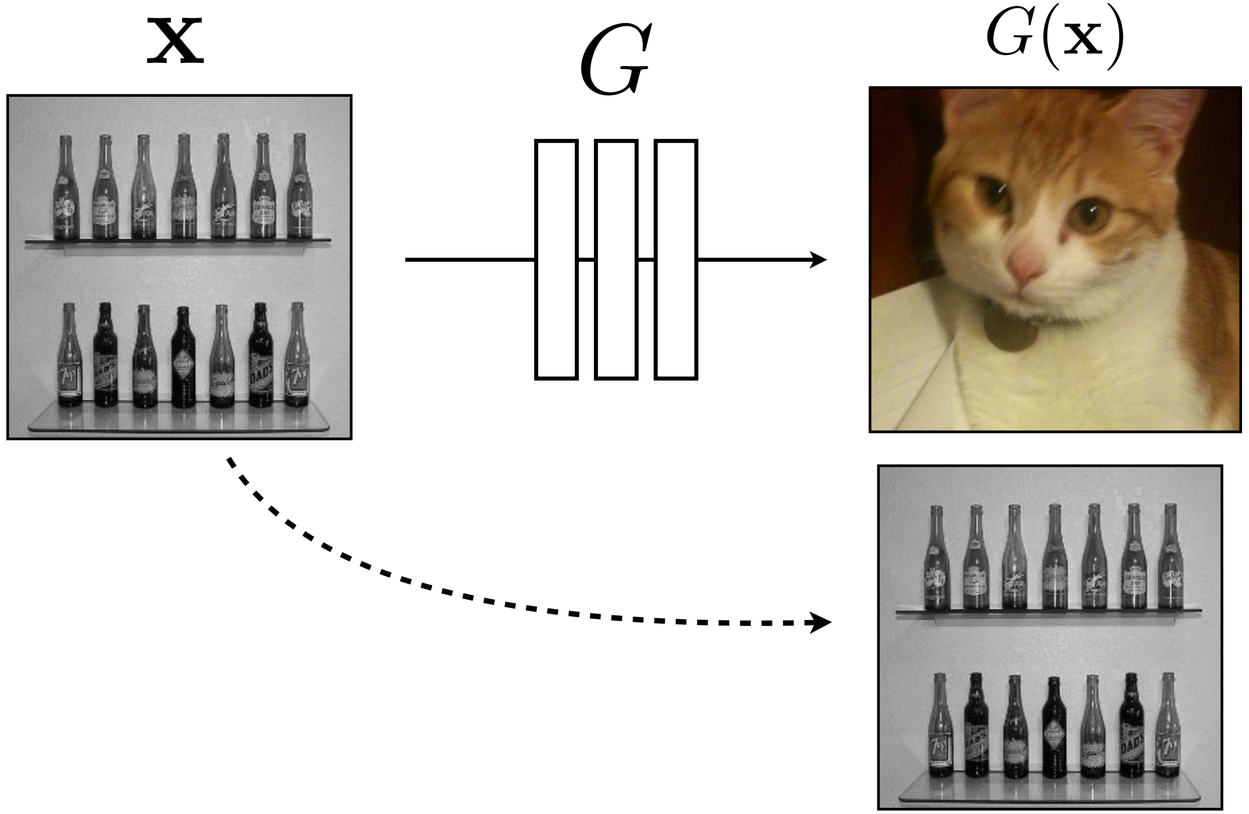
Rather than being hand-designed, it is *learned*.



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



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



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

$\mathbf{x}$

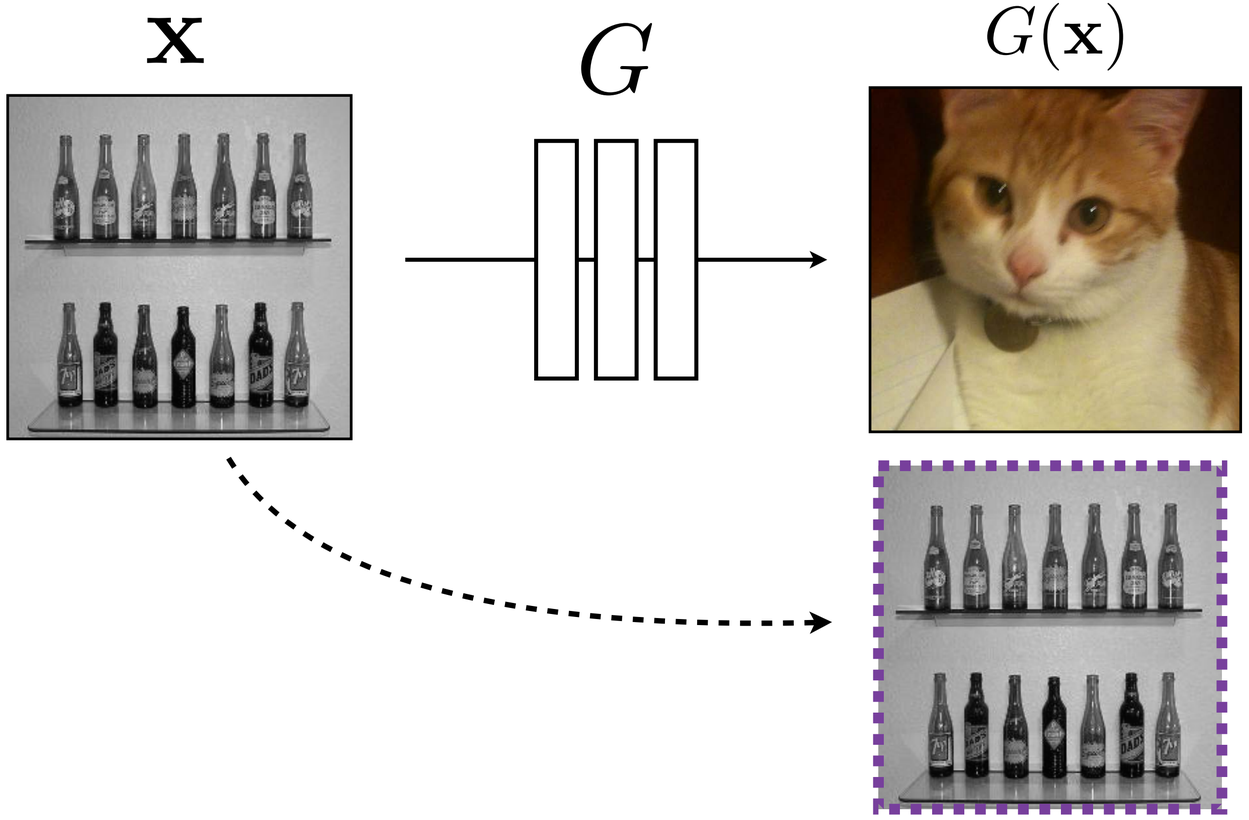
G

$G(\mathbf{x})$

D

real or fake *pair* ?

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



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

$\mathbf{x}$

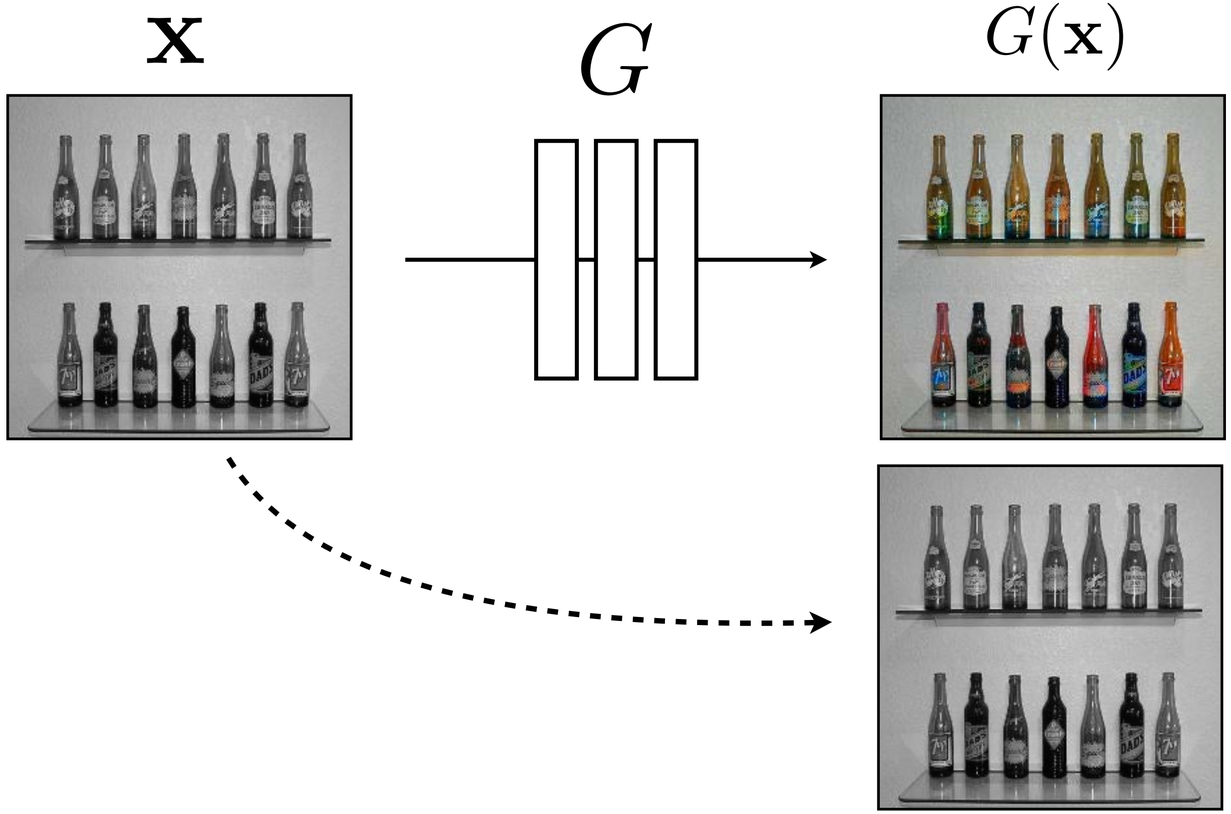
G

$G(\mathbf{x})$

D

real *pair*

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



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

Training Details: Loss function

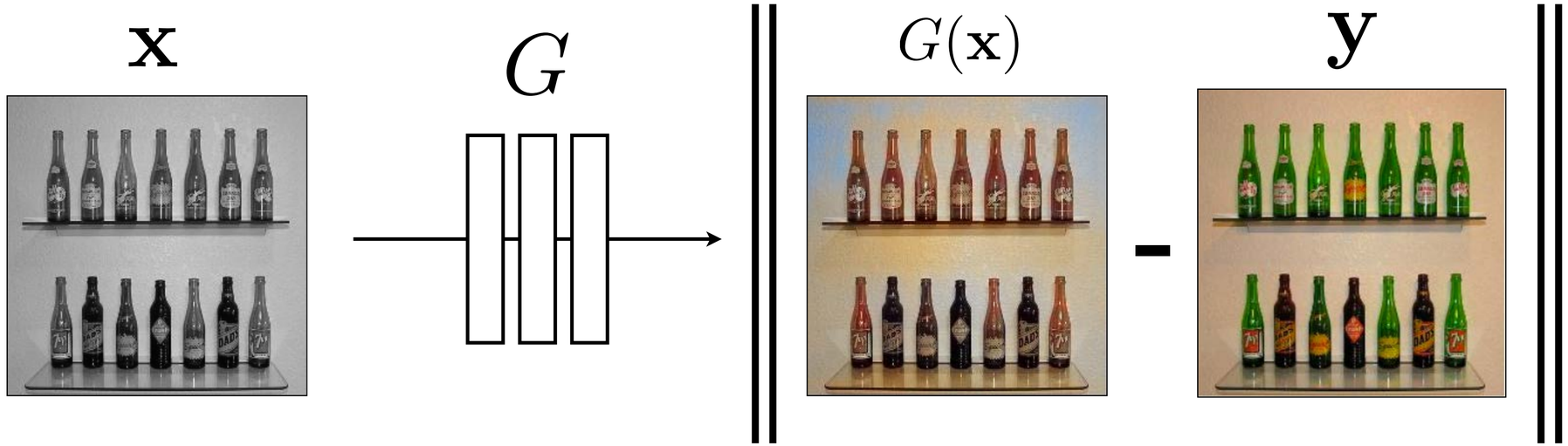
Conditional GAN

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

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

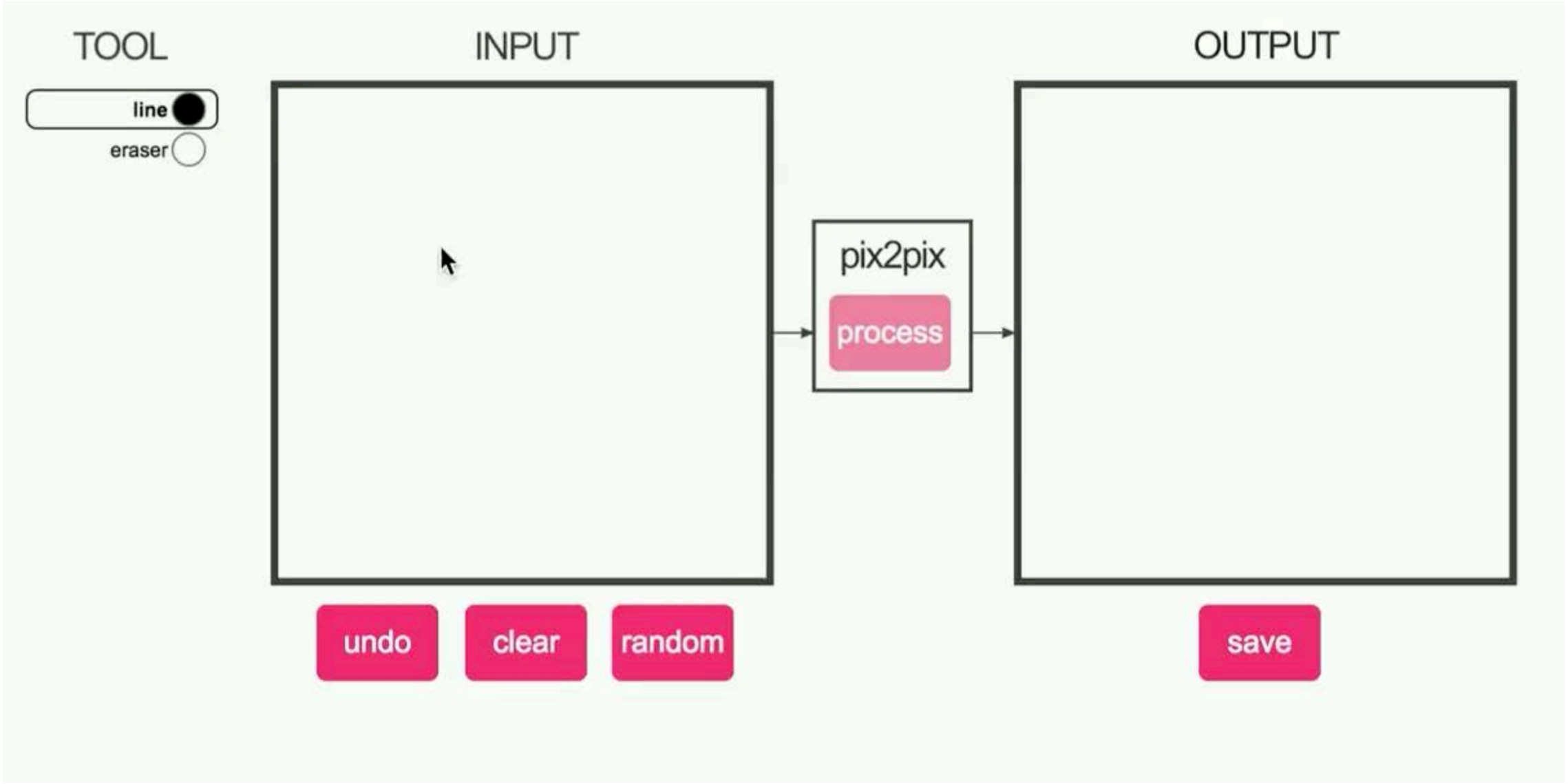
[c.f. Pathak et al. CVPR 2016]

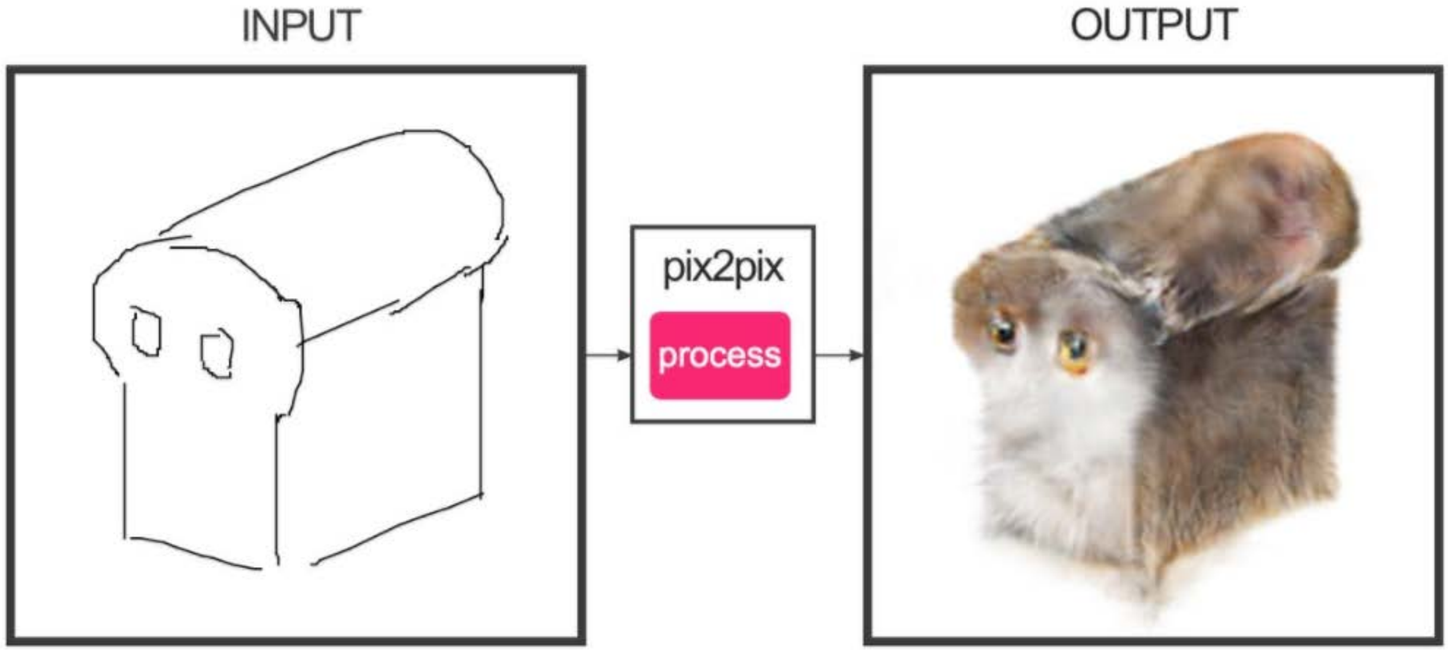
BW → Color

Input Output Input Output Input Output

Data from [Russakovsky et al. 2015]

#edges2cats [Chris Hesse]





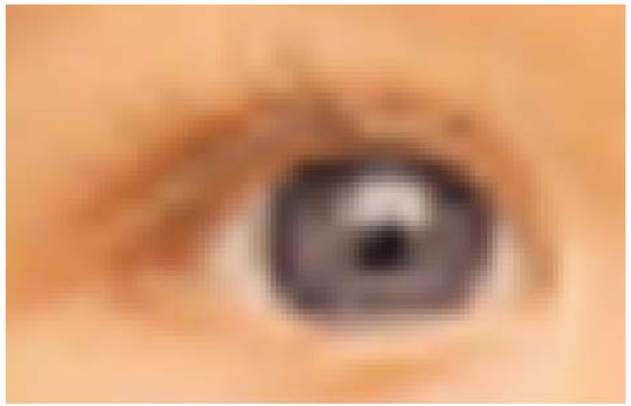
Ivy Tasi @ivymyt

Vitaly Vidmirov @vvid

Structured Prediction

Input

$\mathbf{x}$

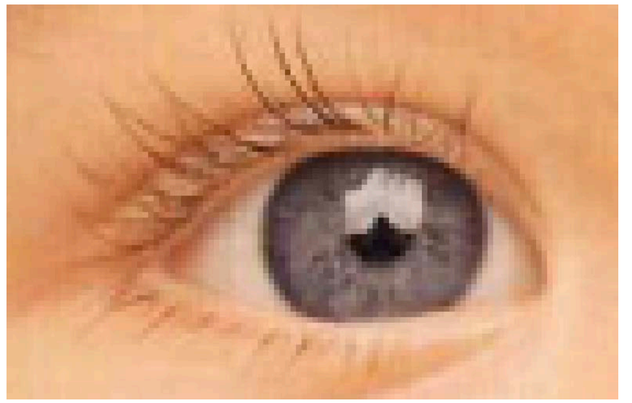


Output

$\hat{\mathbf{y}}$

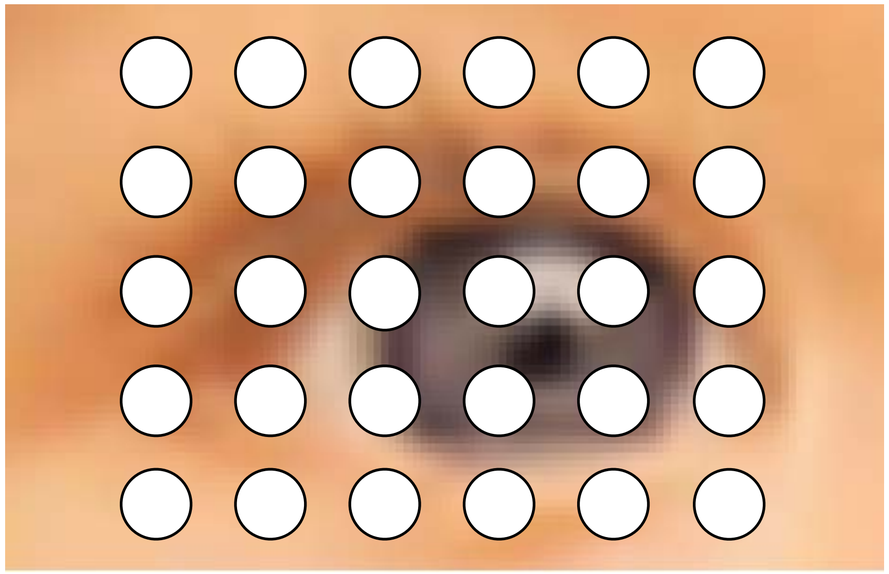
Target

$\mathbf{y}$



$$L(\hat{\mathbf{y}}, \mathbf{y}) = \|\hat{\mathbf{y}} - \mathbf{y}\|_2$$

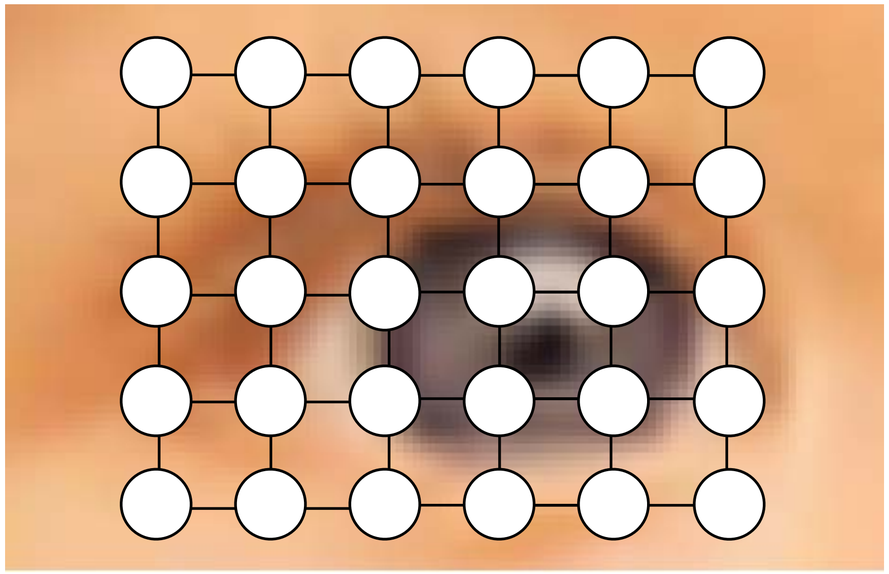
Structured Prediction



Each pixel treated as independent

$$\prod_i p(y_i|\mathbf{x})$$

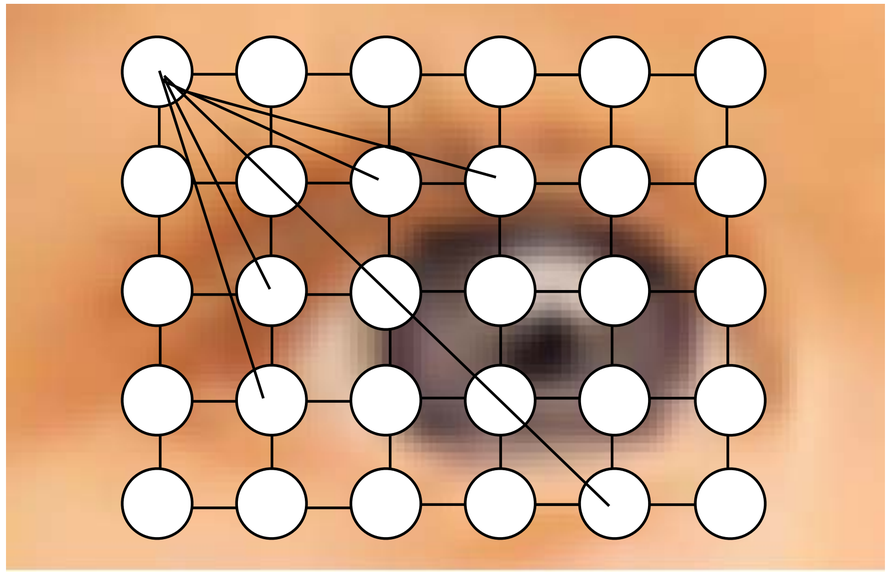
CRF



Models at pairwise configuration of pixels

$$\frac{1}{Z}\prod_{i,j} p(y_i, y_j|\mathbf{x})$$

Structured Prediction



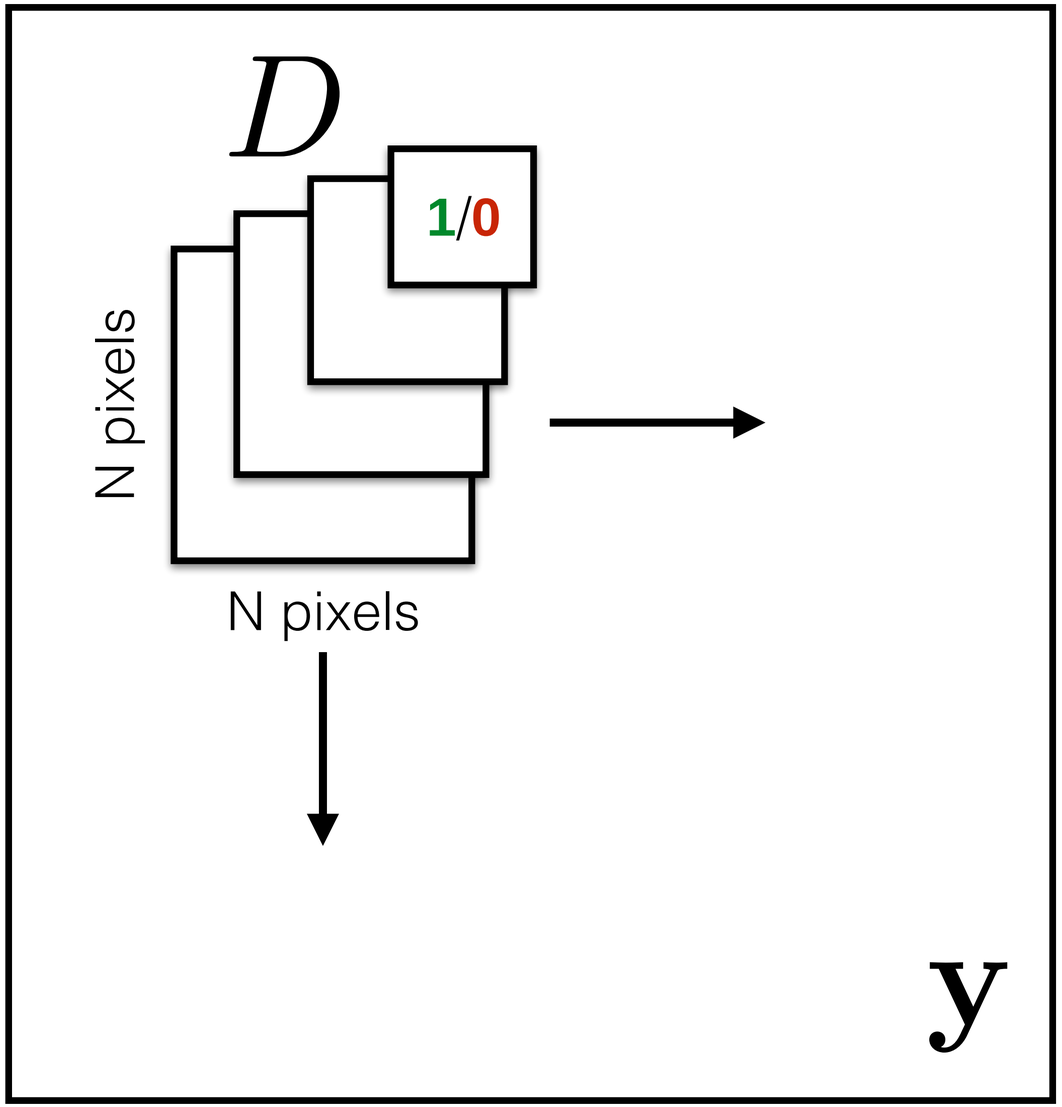
Model *joint* configuration of all pixels

$$p(\mathbf{y}|\mathbf{x})$$

A GAN, with sufficient capacity, samples from the full joint distribution (at equilibrium)

Most generative models have this property! Give them **sufficient capacity** and **infinite data**, and they are the complete solution to prediction problems.

Shrinking the capacity: Patch Discriminator

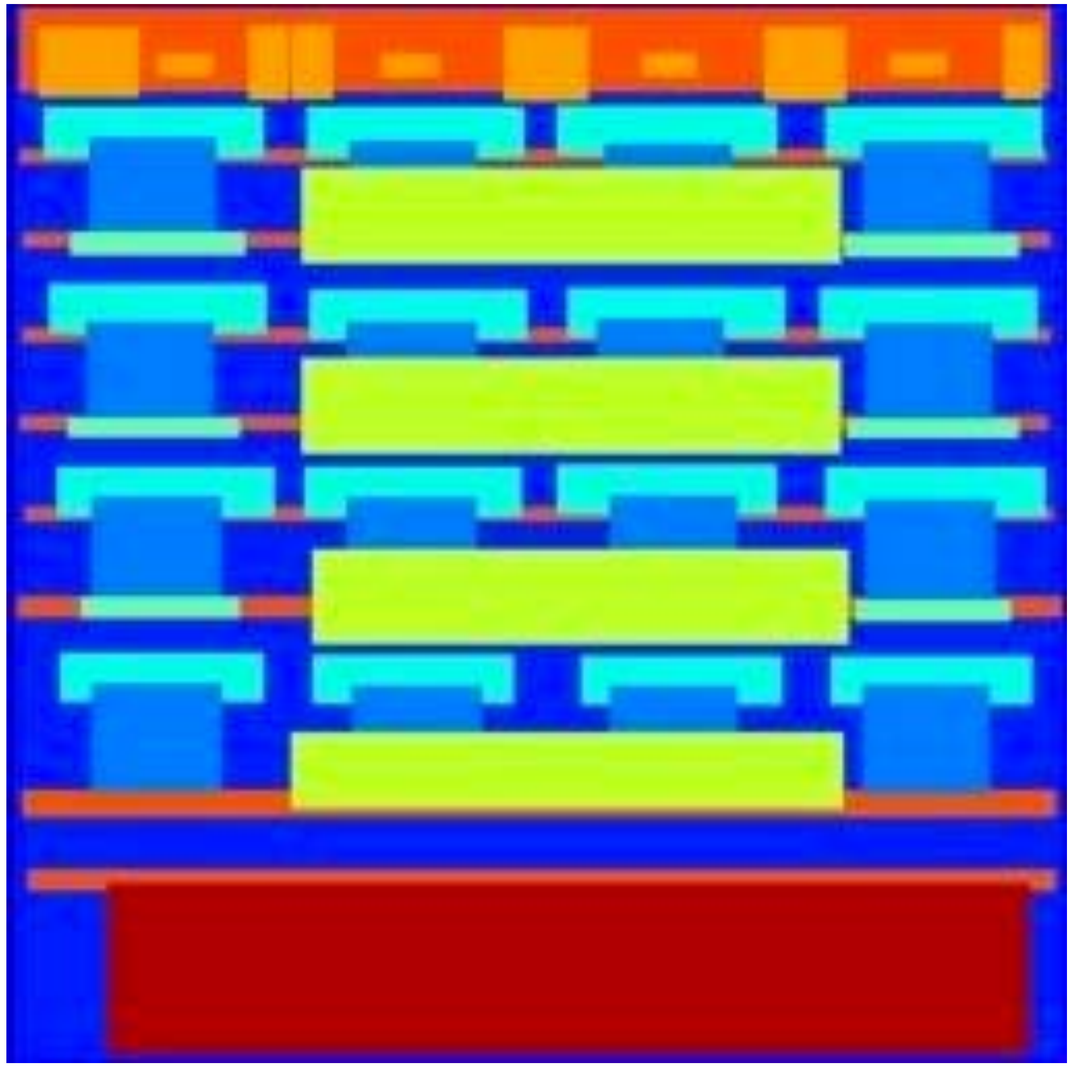


Rather than penalizing if output *image* looks fake, penalize if each overlapping *patch* in output looks fake

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

Labels → Facades

Input

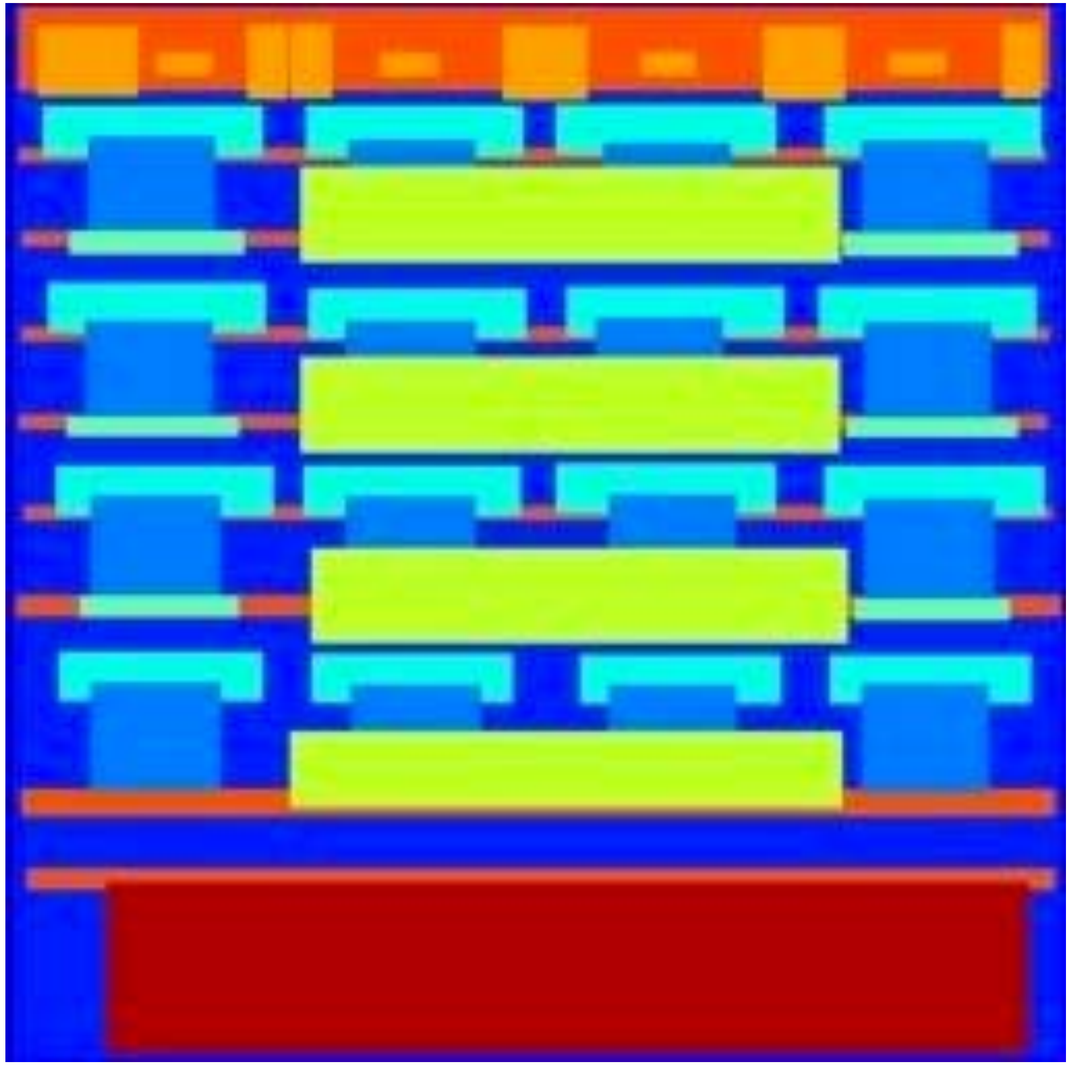


1x1 Discriminator

Data from [Tylecek, 2013]

Labels → Facades

Input

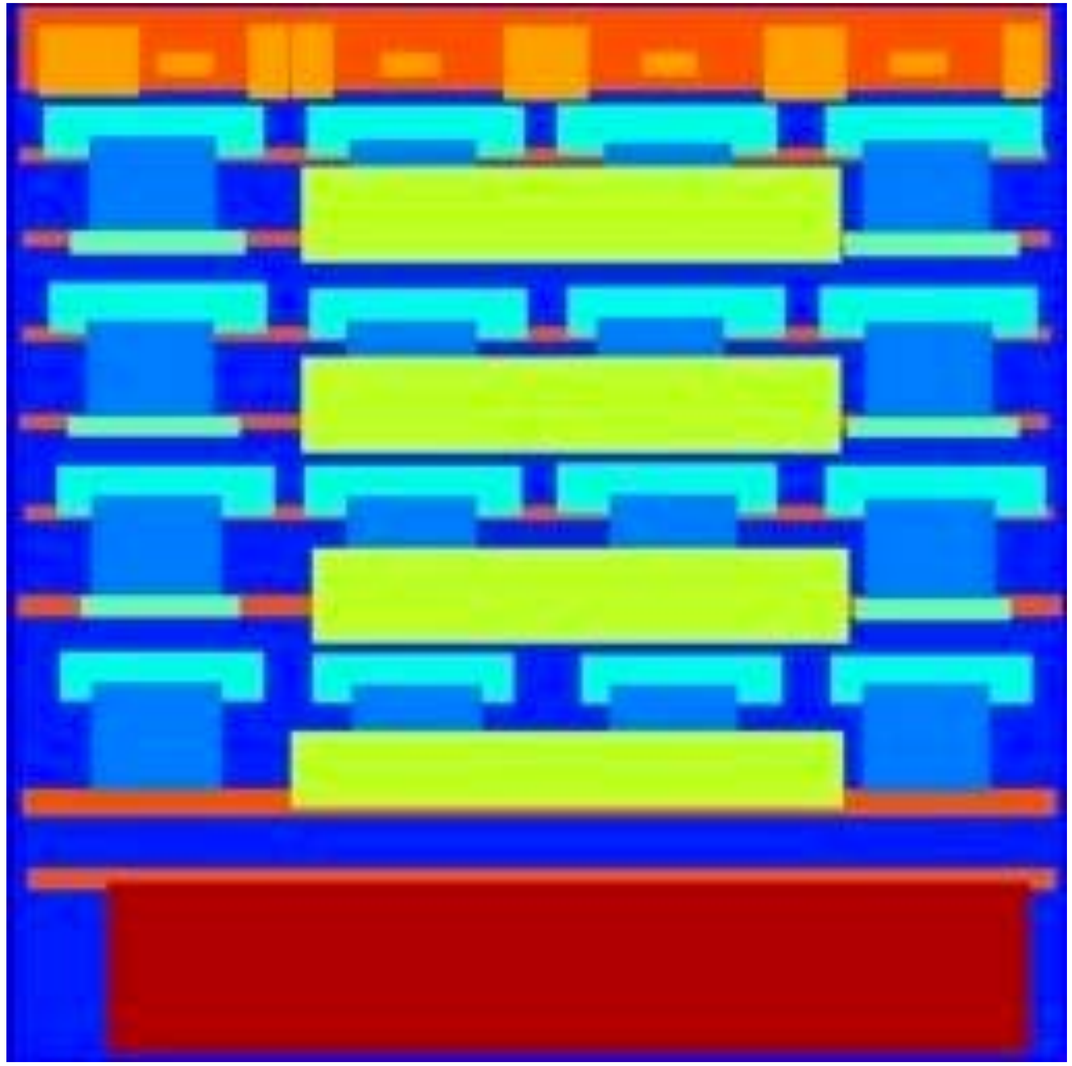


16x16 Discriminator

Data from [Tylecek, 2013]

Labels → Facades

Input

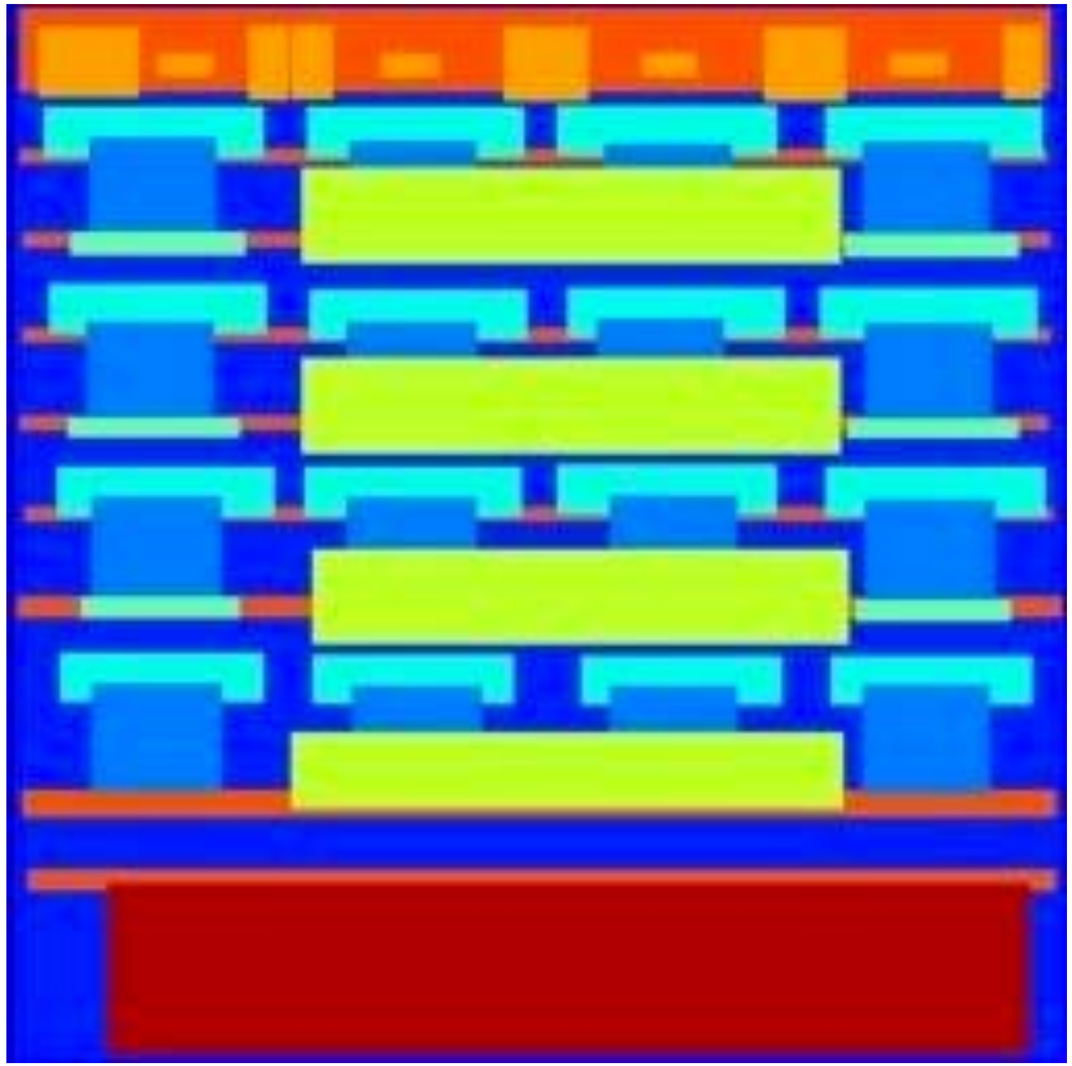


70x70 Discriminator

Data from [Tylecek, 2013]

Labels → Facades

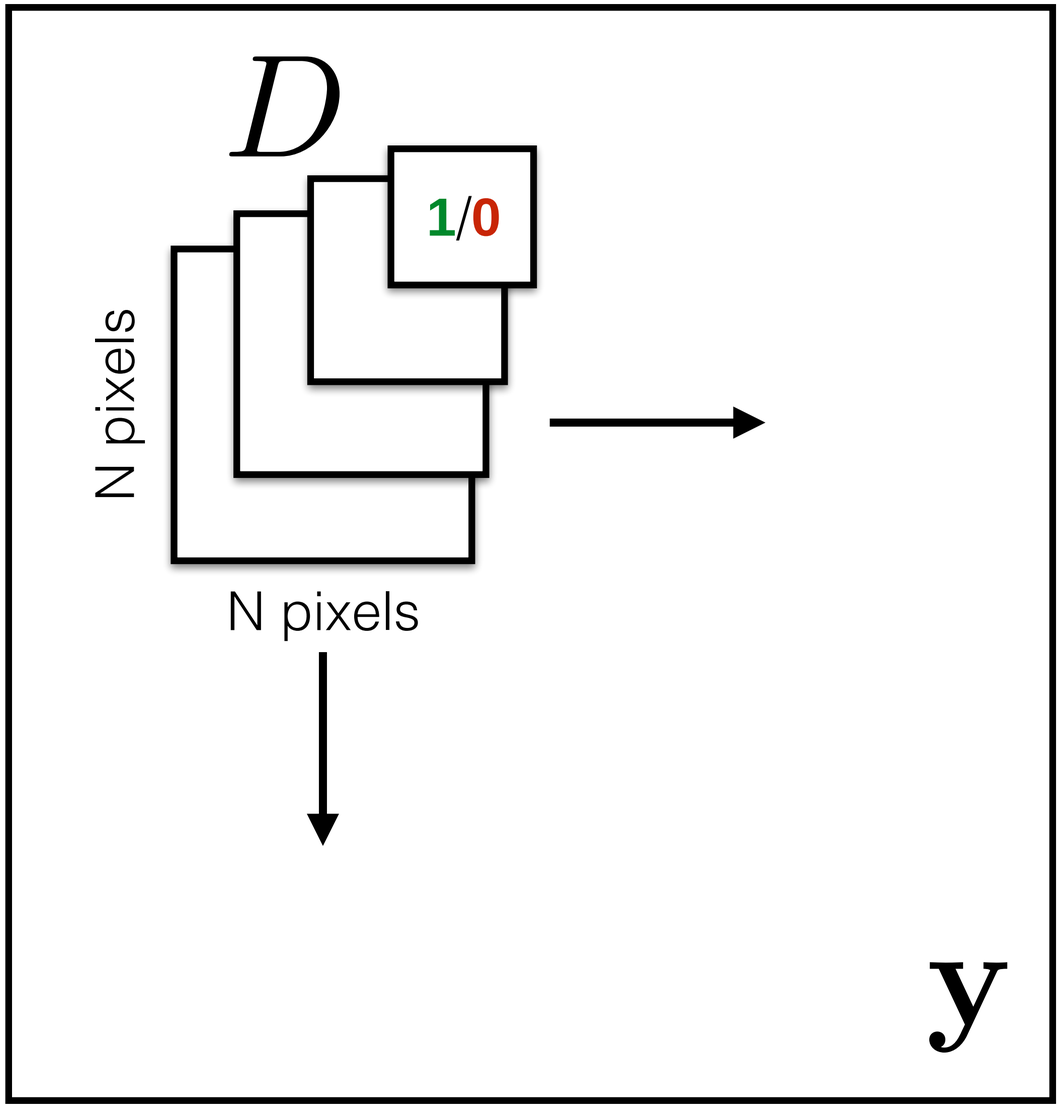
Input



Full image Discriminator

Data from [Tylecek, 2013]

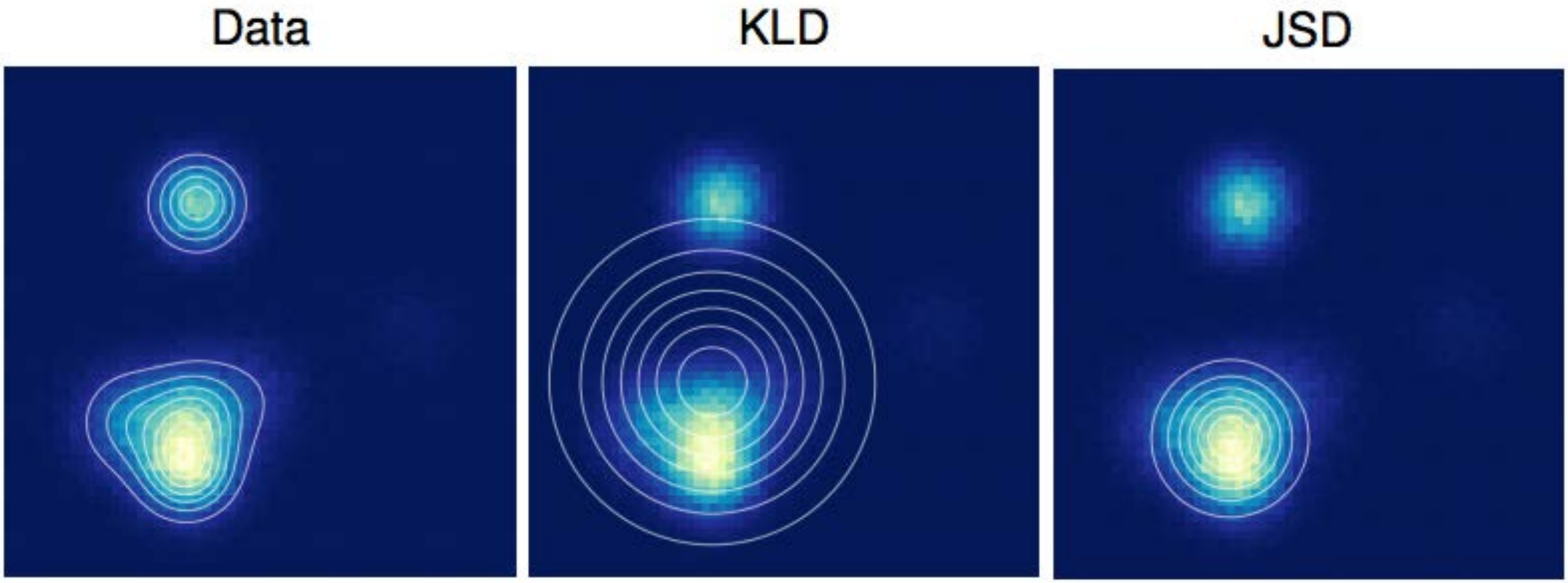
Patch Discriminator



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

Shrinking the capacity: model misspecification



[Theis et al. 2016]

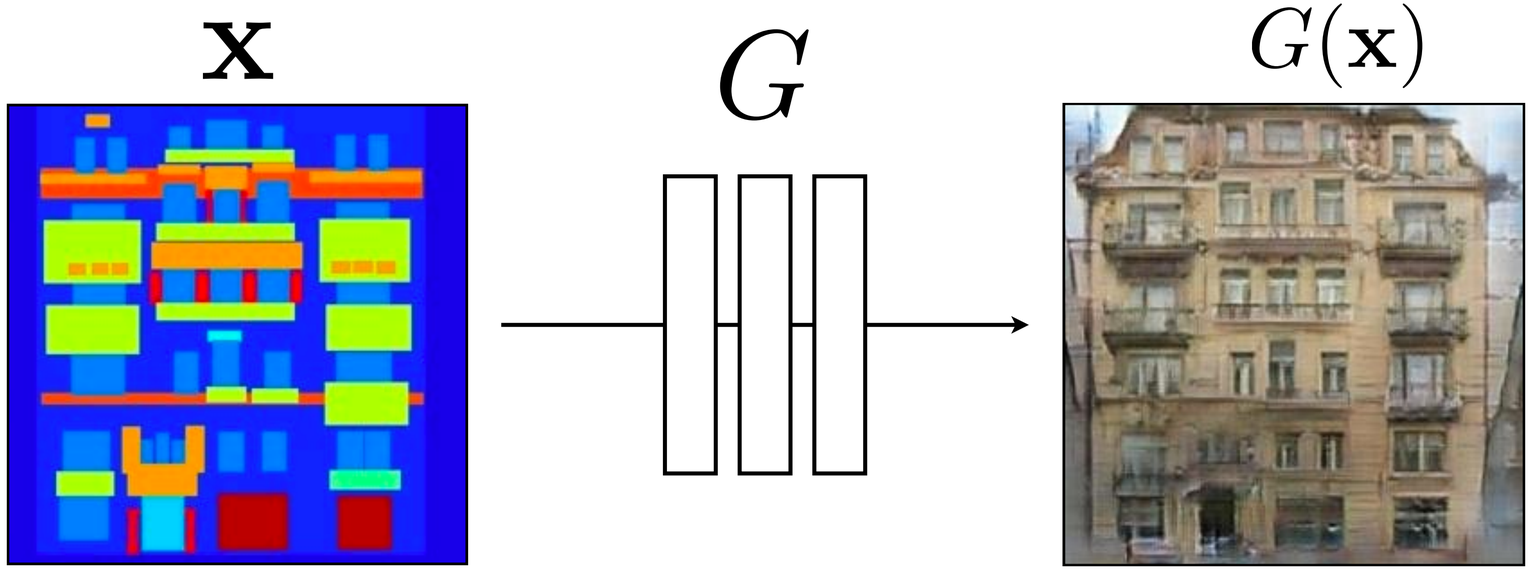
Mode covering versus mode seeking

VAE

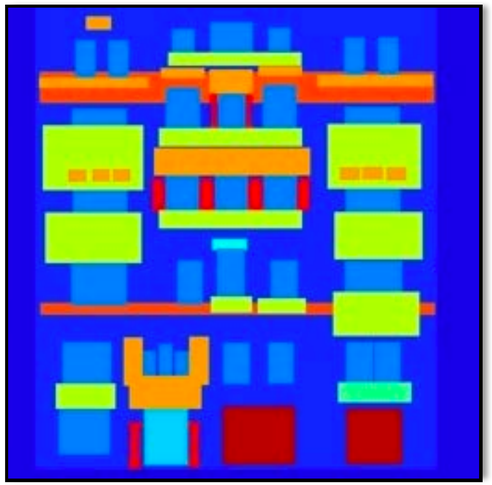
GAN

[Larsen et al. 2016]

Modeling multiple possible outputs



Modeling multiple possible outputs



Input

?

?

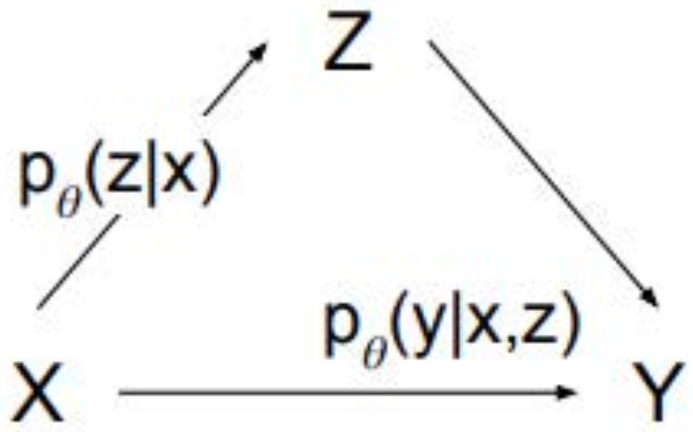
?

?

?

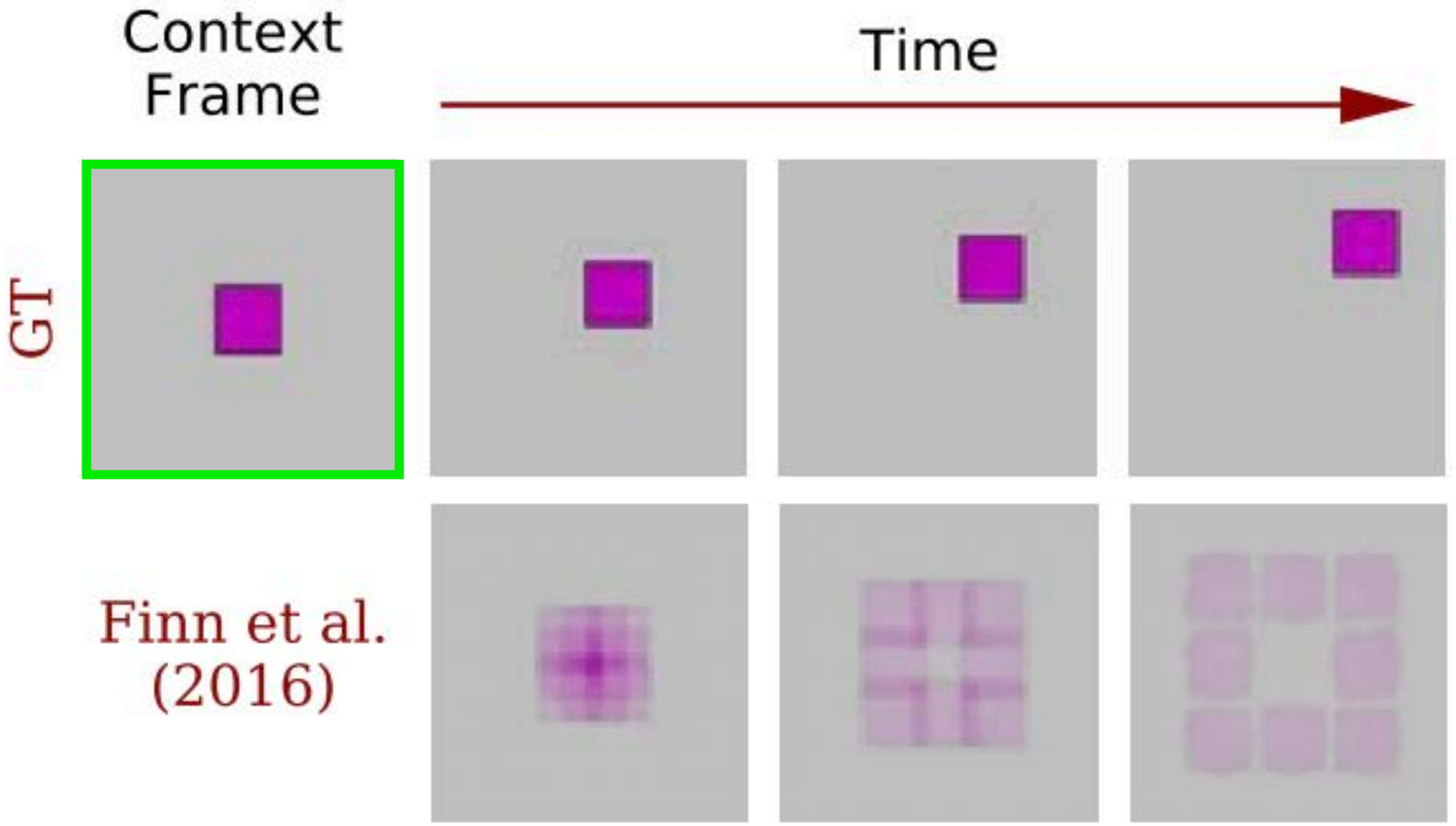
Possible outputs

Conditional Variational Autoencoders



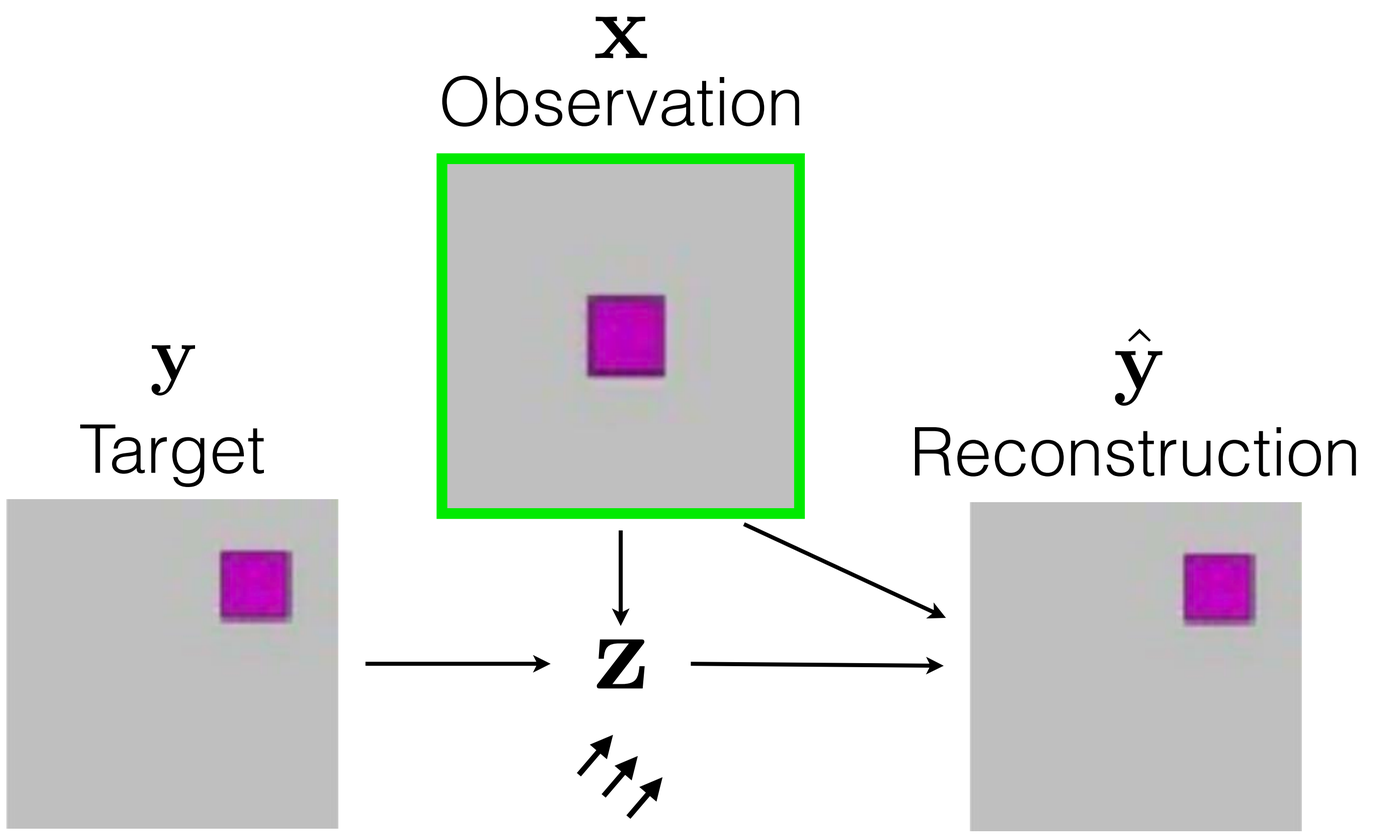
$$\log p_\theta(\mathbf{y}|\mathbf{x}) \geq -KL\left(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})\|p_\theta(\mathbf{z}|\mathbf{x})\right) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})}\left[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{z})\right]$$

[Sohn, Yan, Lee, NIPS 2015]



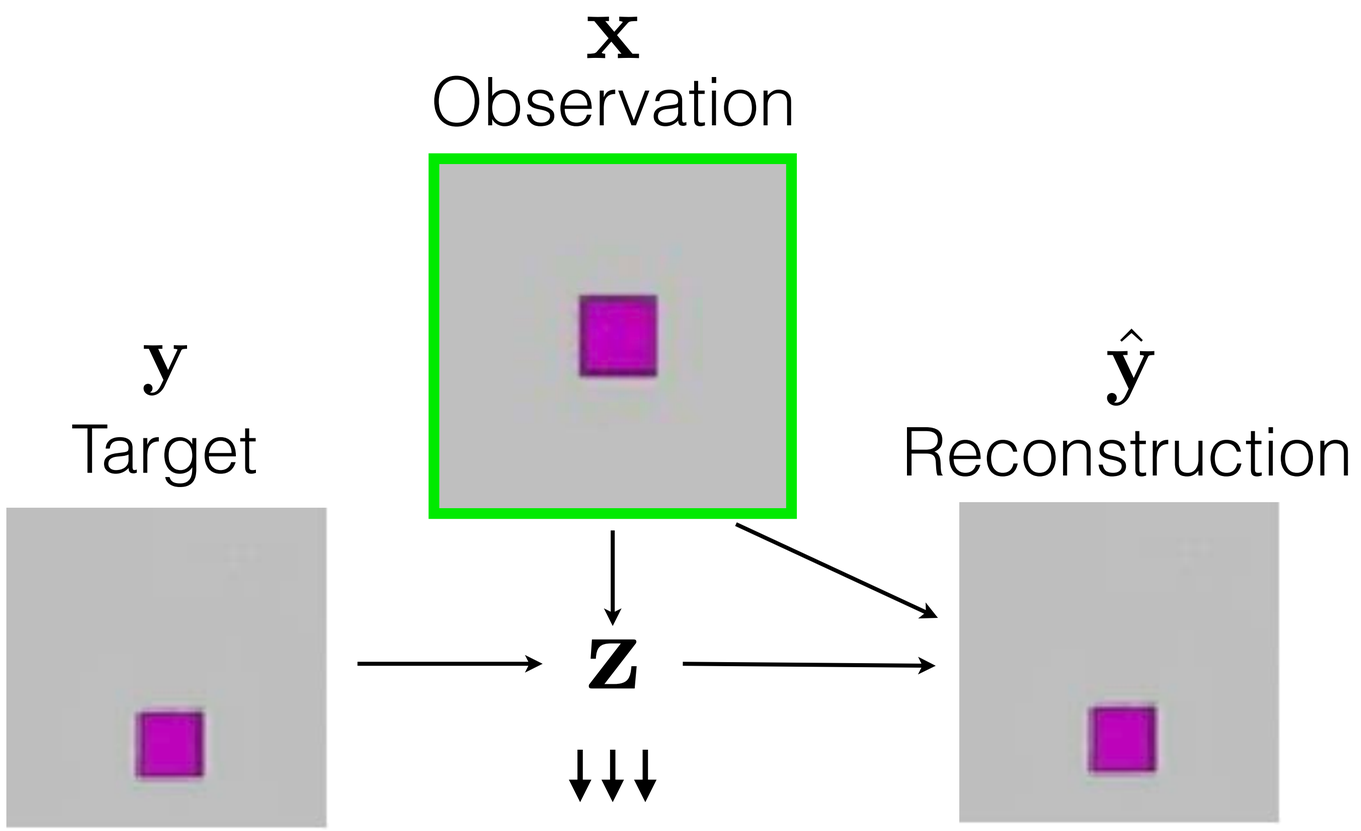
Example from [Babaeizadeh et al., ICLR 2018]

see also [Walker et al., ECCV 2016], [Xue*, Wu*, et al., NIPS 2016]



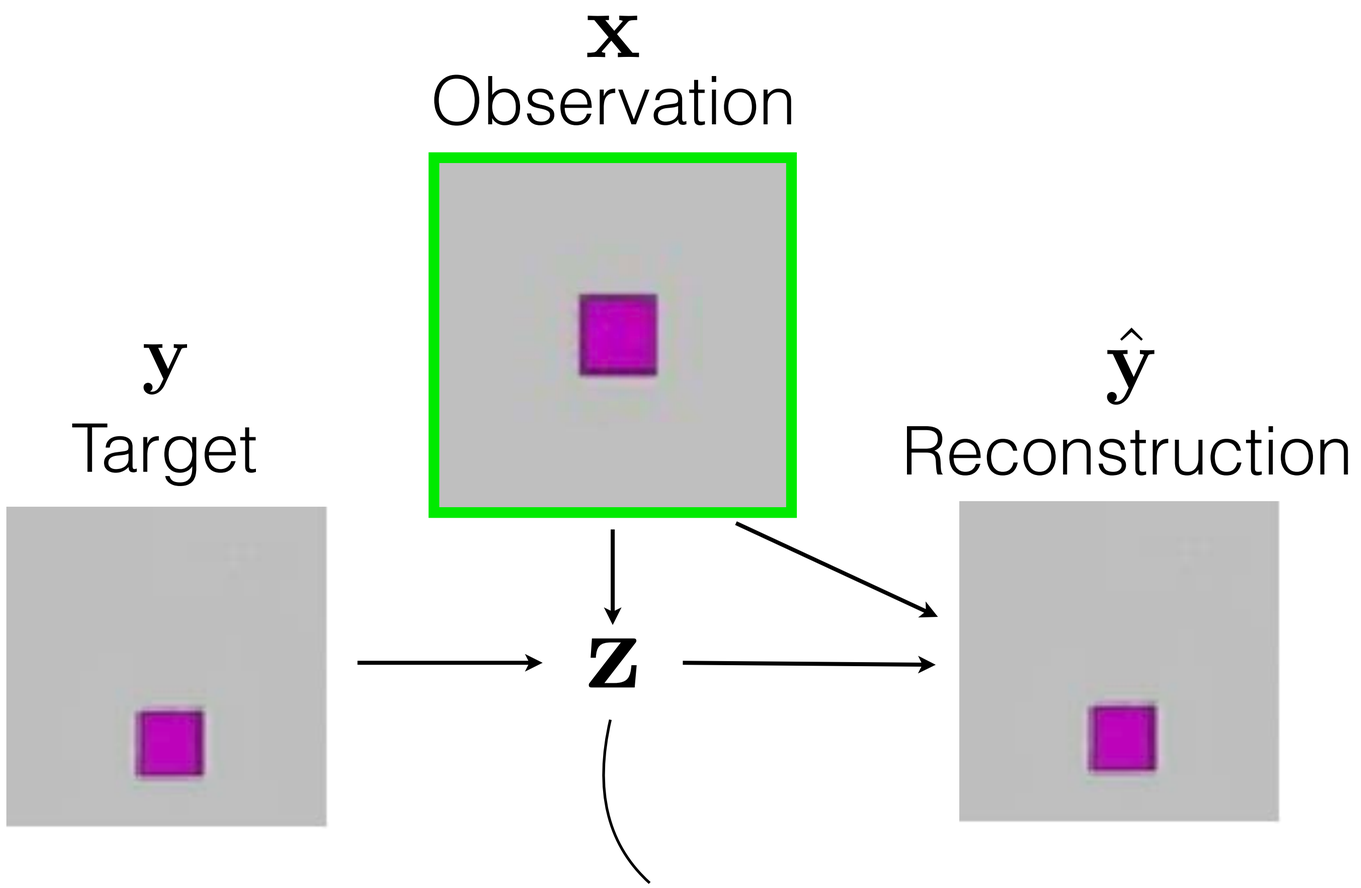
Example from [Babaeizadeh et al., ICLR 2018]

see also [Walker et al., ECCV 2016], [Xue*, Wu*, et al., NIPS 2016]



Example from [Babaeizadeh et al., ICLR 2018]

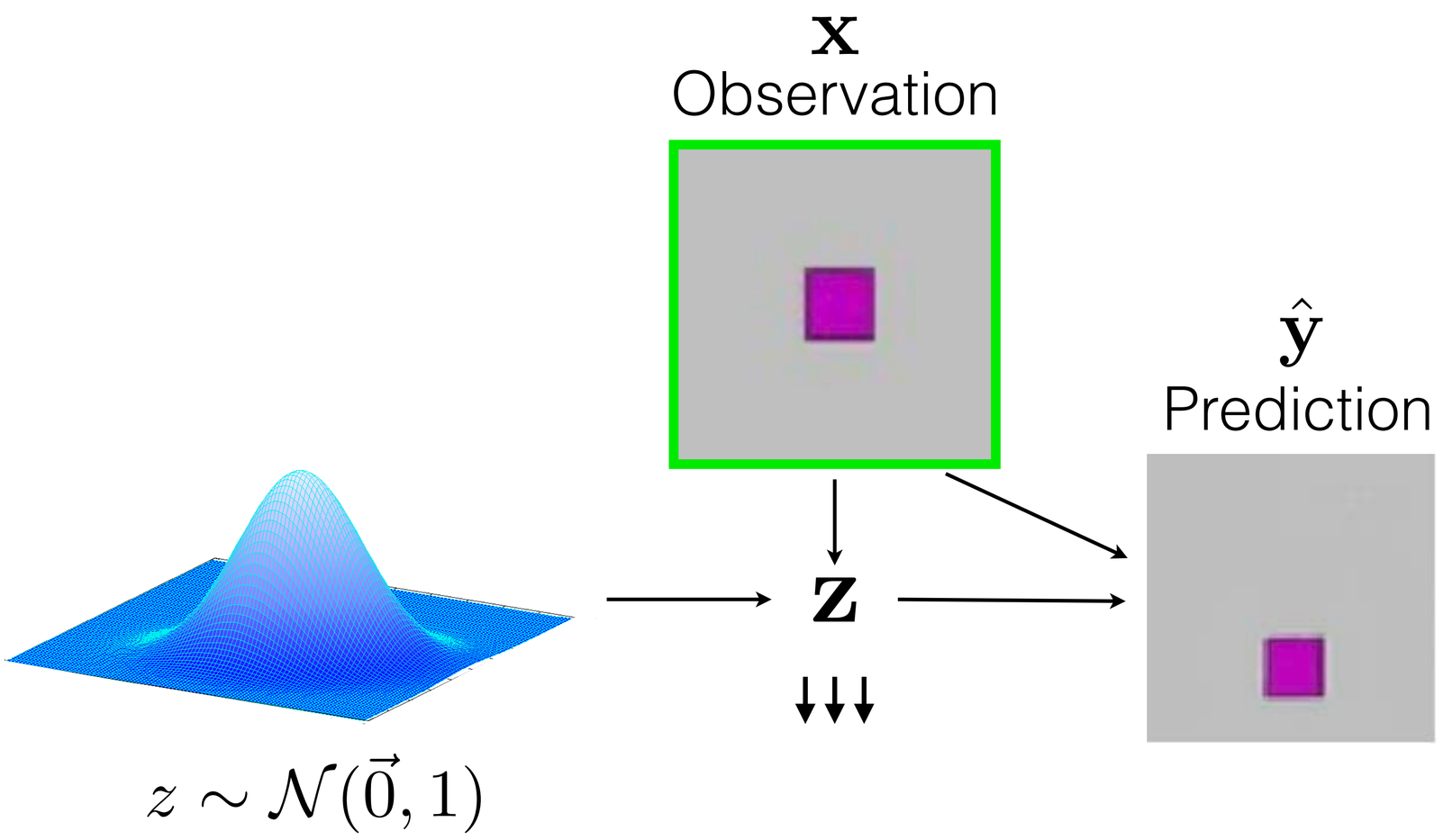
see also [Walker et al., ECCV 2016], [Xue*, Wu*, et al., NIPS 2016]



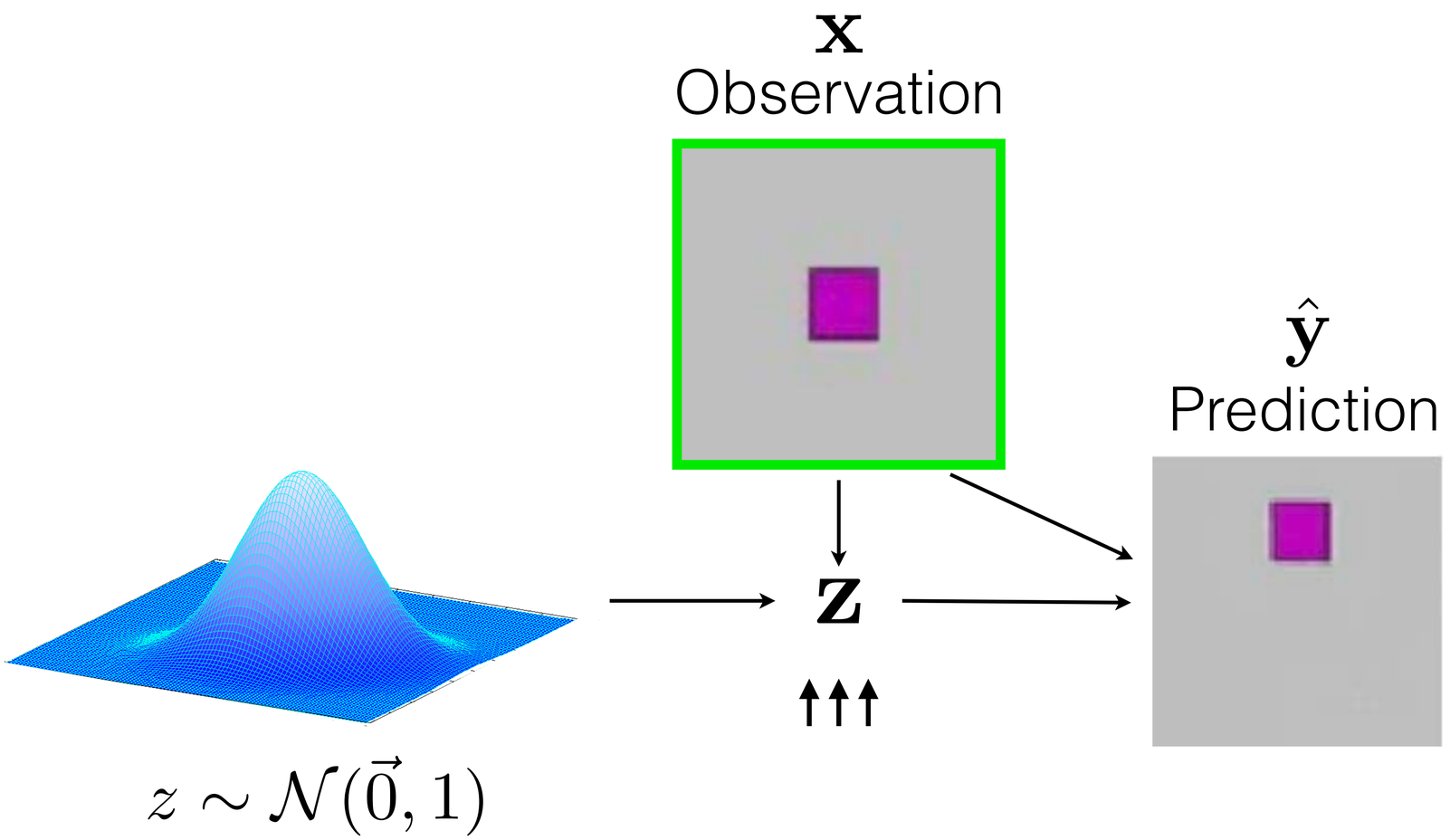
z learns to encode the missing information necessary to predict y from x, i.e. the direction in which the purple box moves

Example from [Babaeizadeh et al., ICLR 2018]

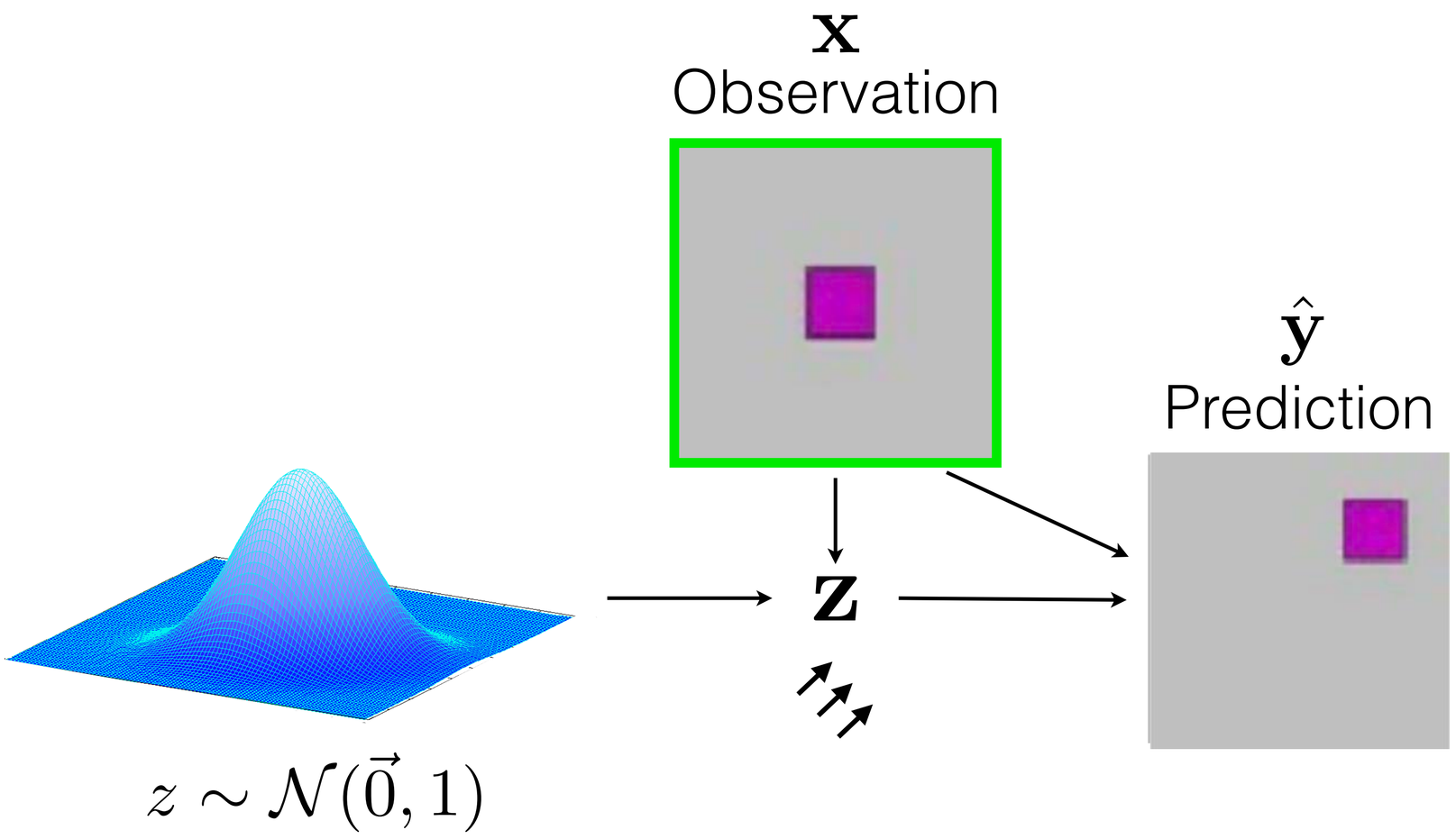
see also [Walker et al., ECCV 2016], [Xue*, Wu*, et al., NIPS 2016]



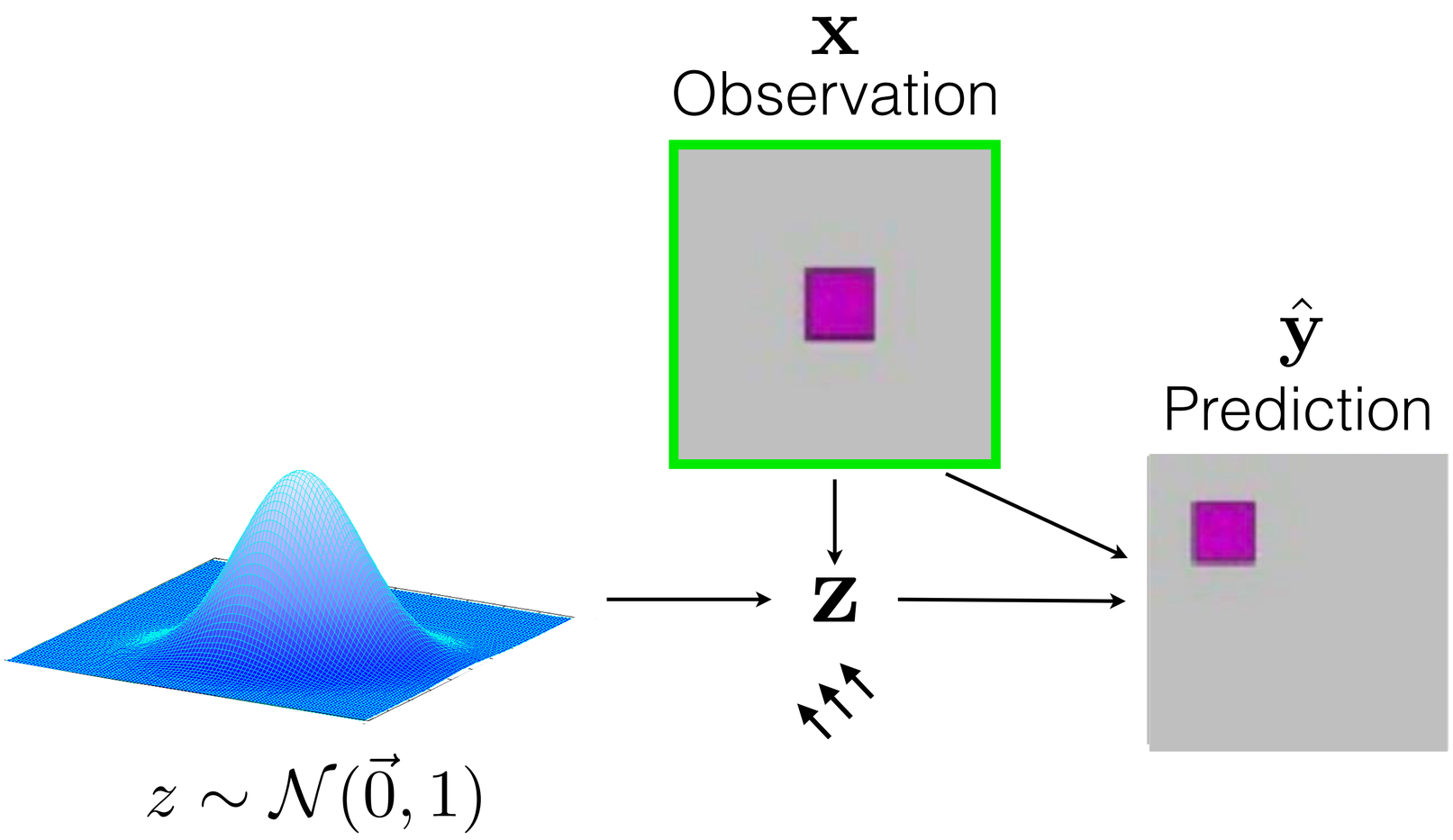
see also [Walker et al., ECCV 2016], [Xue*, Wu*, et al., NIPS 2016]



see also [Walker et al., ECCV 2016], [Xue*, Wu*, et al., NIPS 2016]



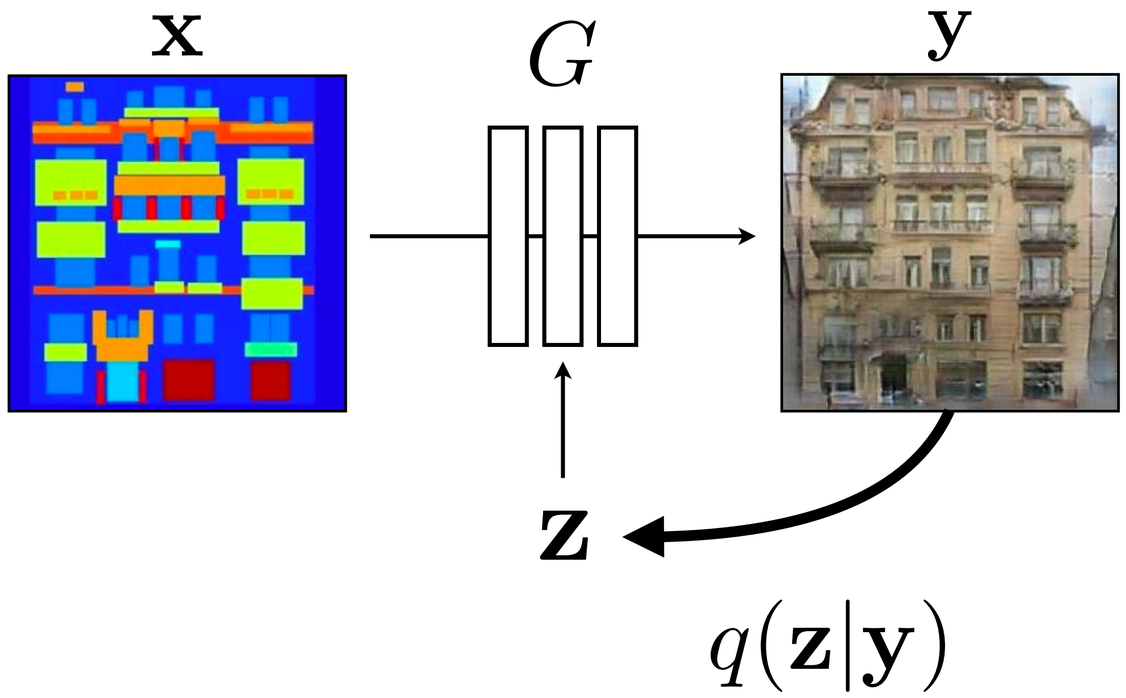
see also [Walker et al., ECCV 2016], [Xue*, Wu*, et al., NIPS 2016]



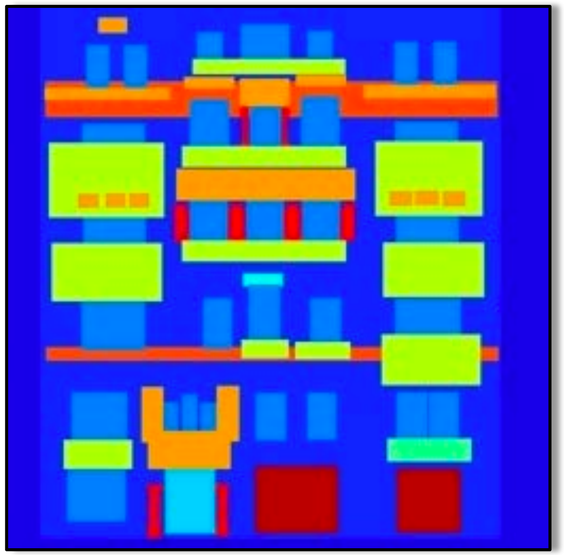
see also [Walker et al., ECCV 2016], [Xue*, Wu*, et al., NIPS 2016]

InfoGAN [Chen et al. 2016]

BiCycleGAN [Zhu et al., NIPS 2017]



Encourages z to relay information about the target.



Labels

Randomly generated facades

[BiCycleGAN, Zhu et al., NIPS 2017]

What can you do with generative models?

1. **Data prediction**
2. Domain mapping
3. Representation learning
4. Model-based intelligence

What can you do with generative models?

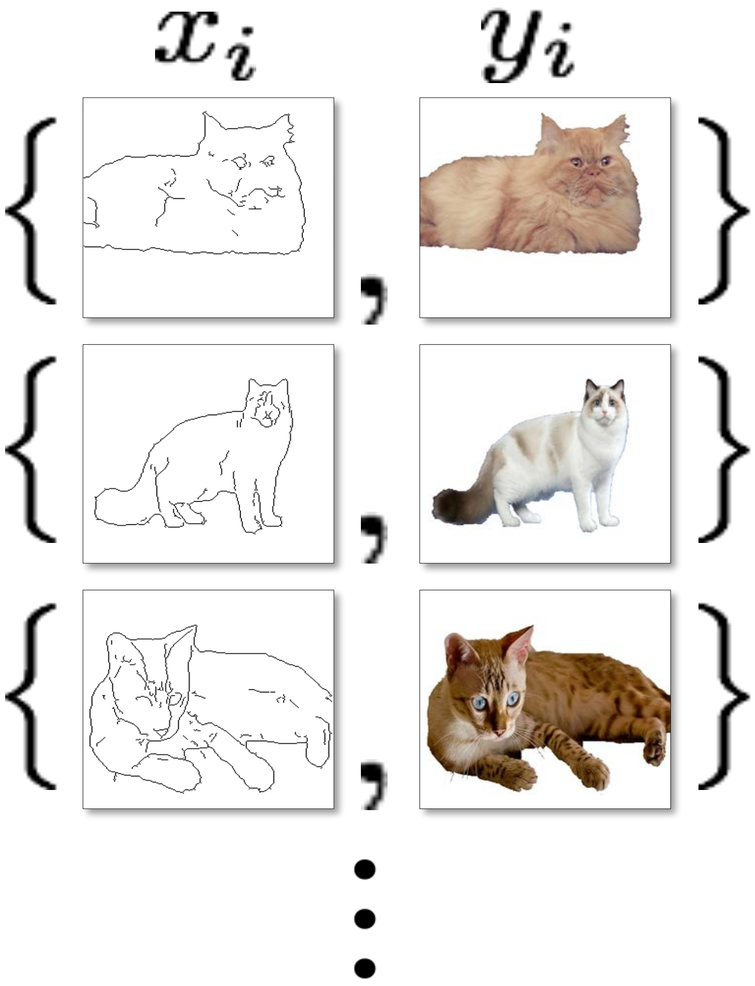
1. Data prediction

2. **Domain mapping**

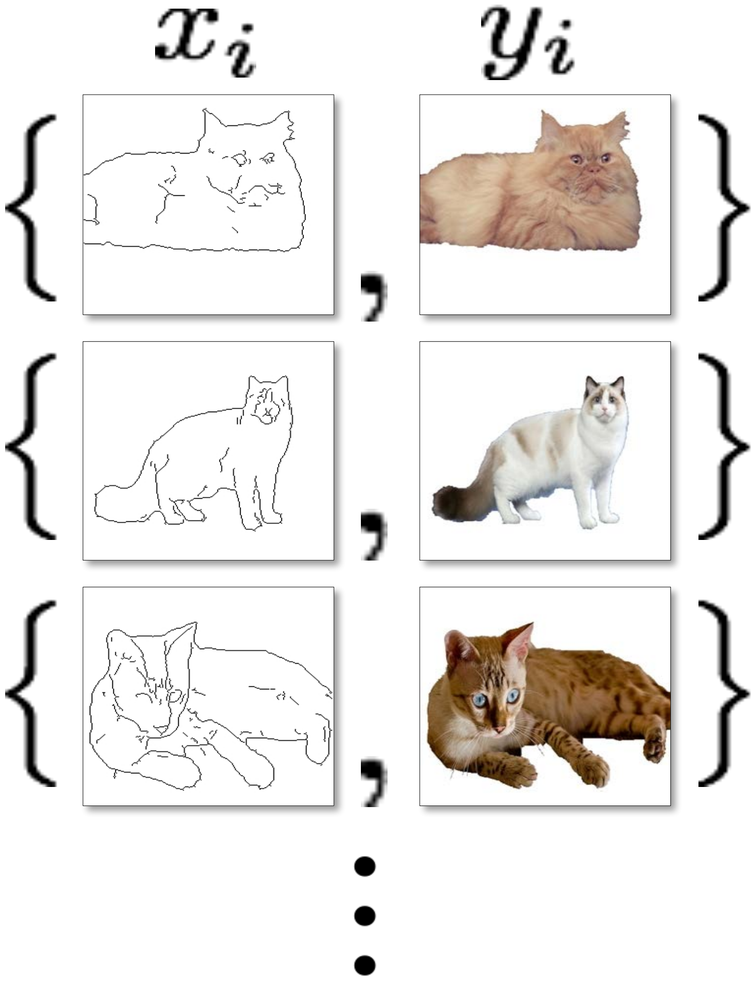
3. Representation learning

4. Model-based intelligence

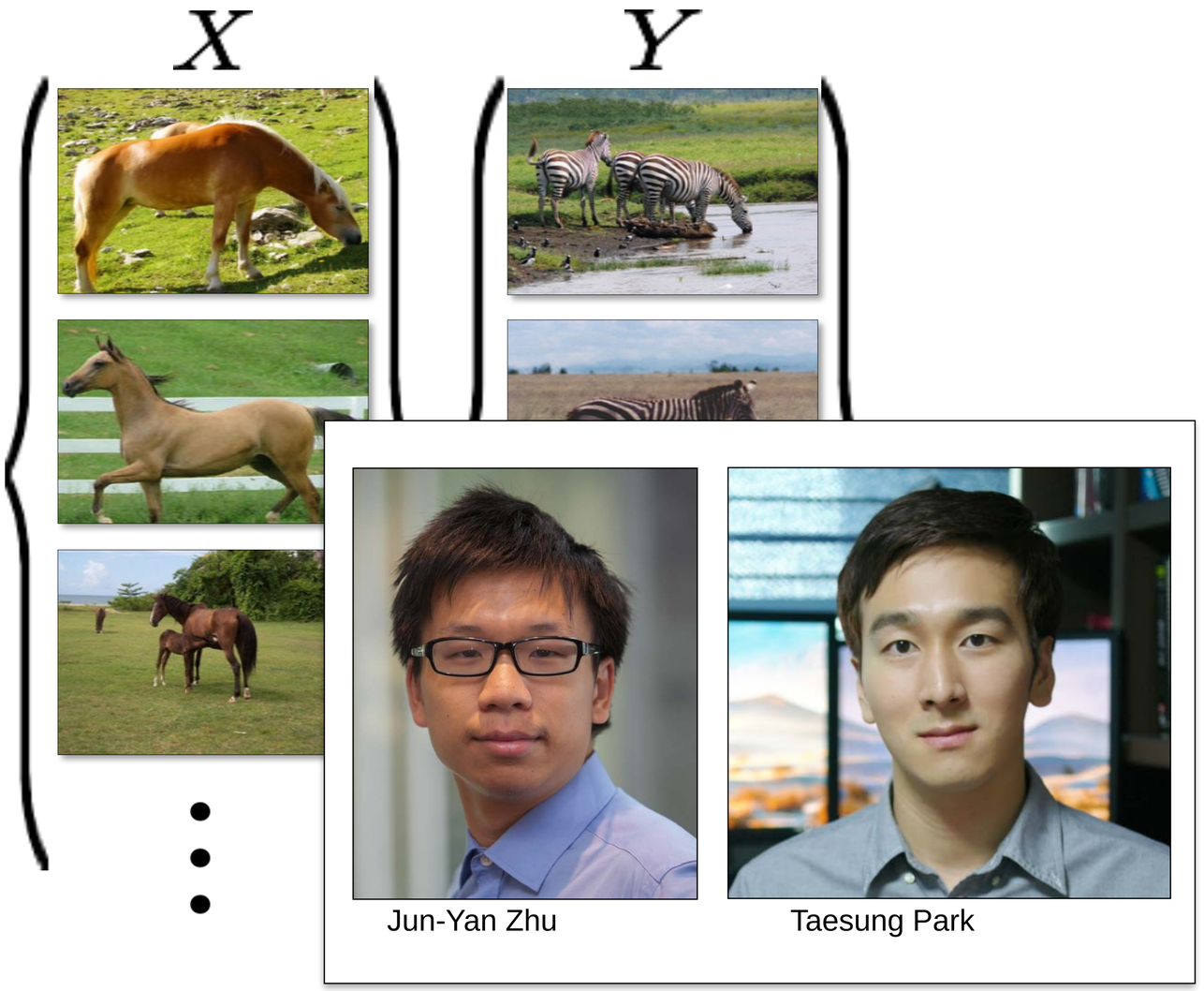
Paired data

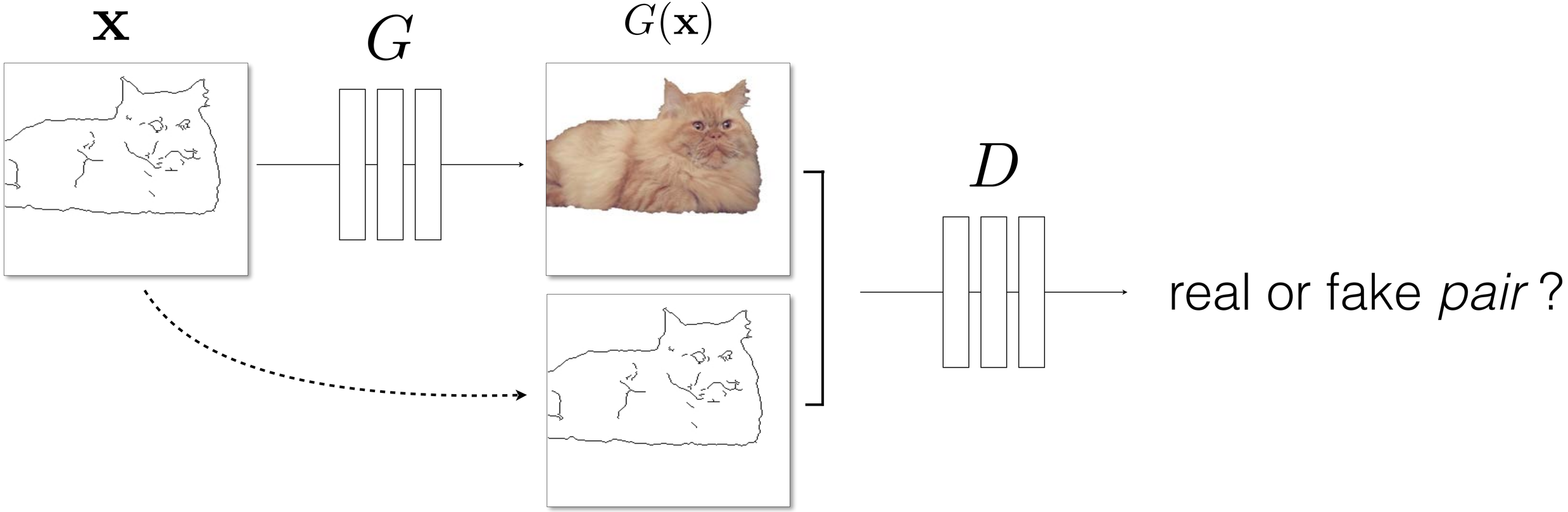


Paired data

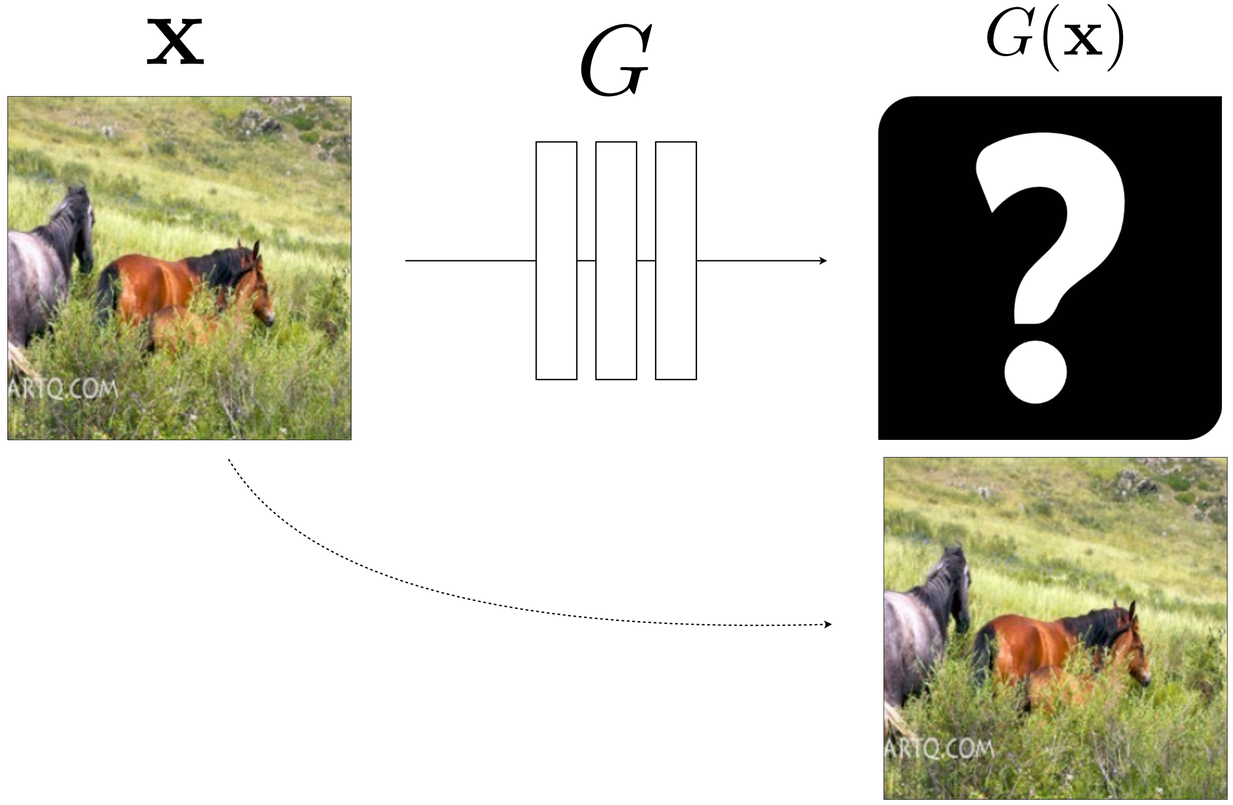


Unpaired data





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

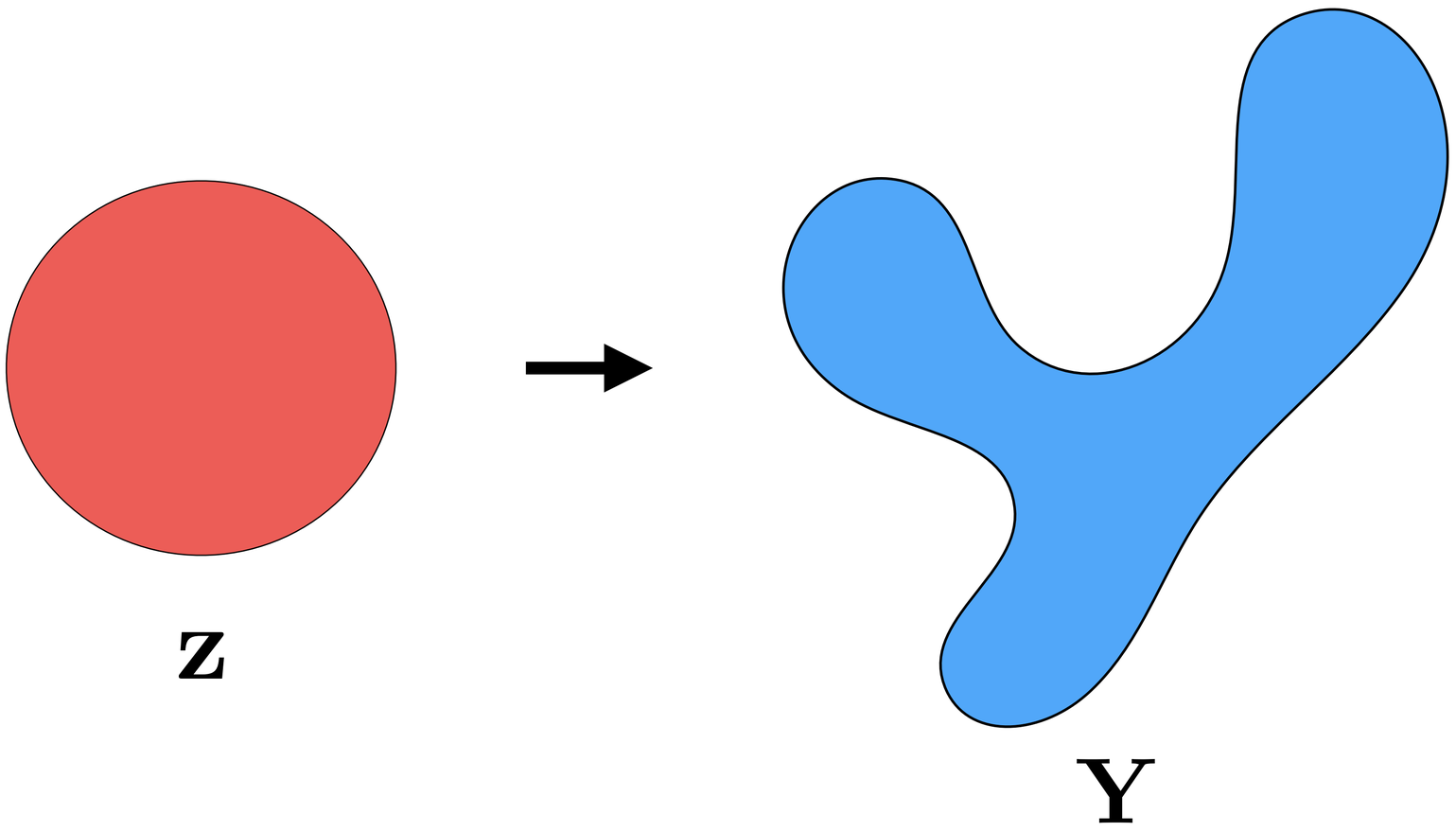


$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\;\; \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y}))\;\;]$$

No input-output pairs!

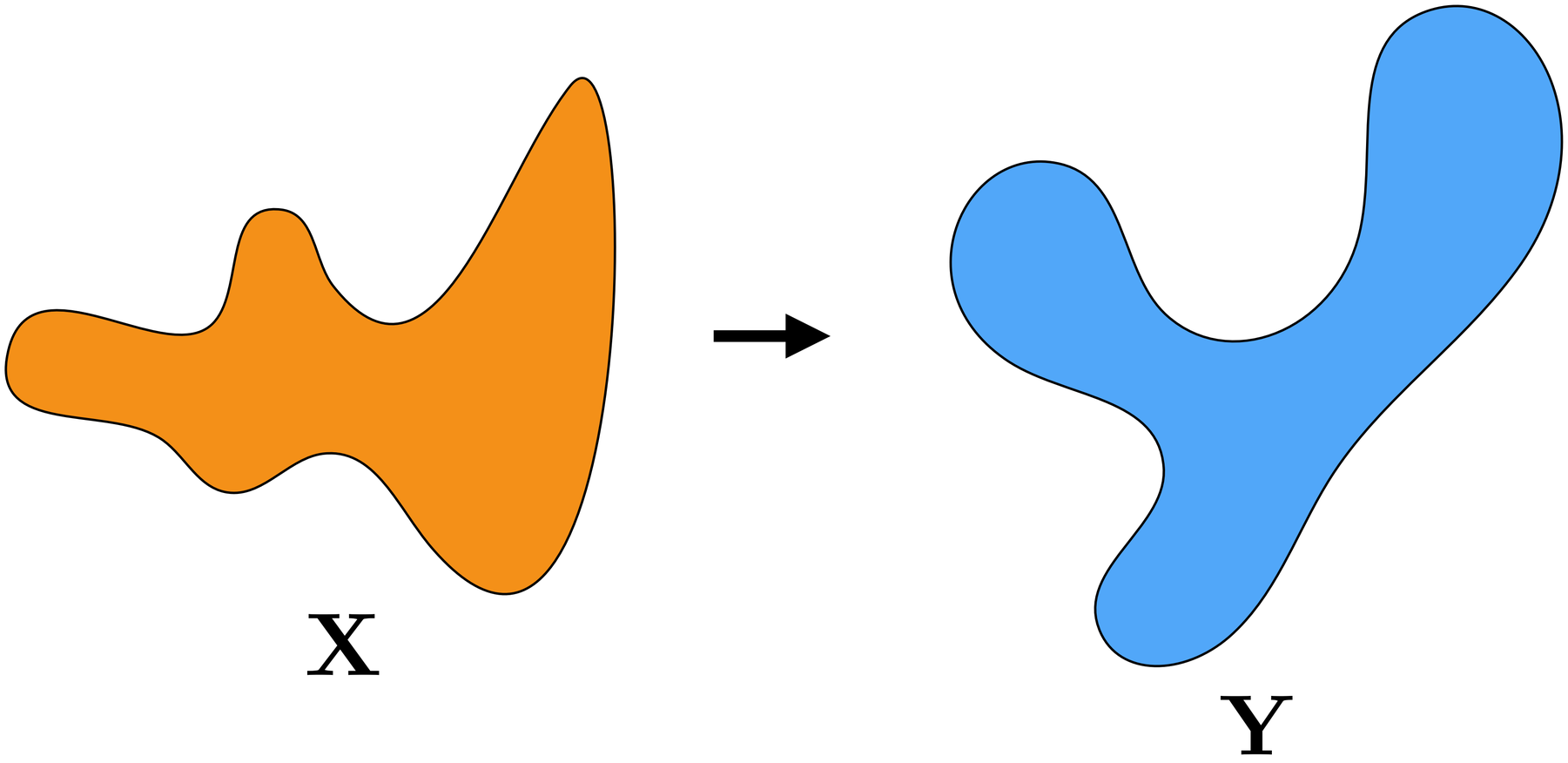
Gaussian

Target distribution



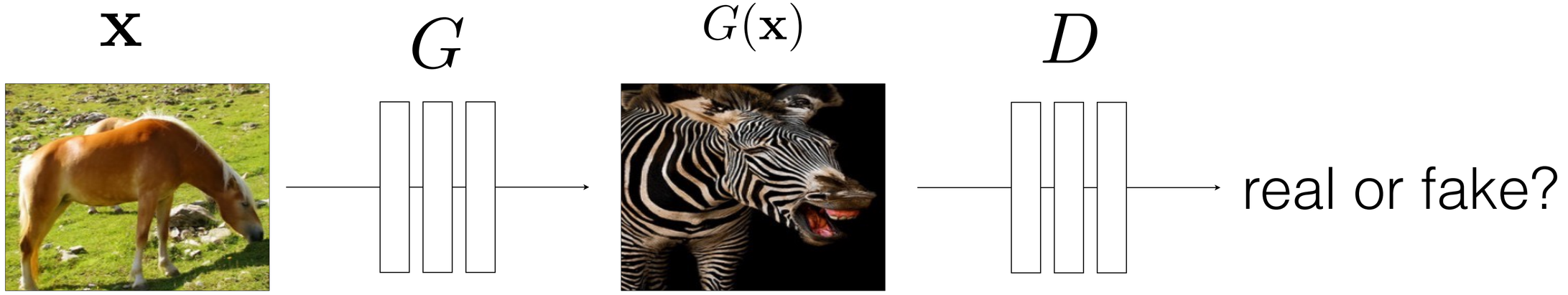
Horses

Zebras



X

Y



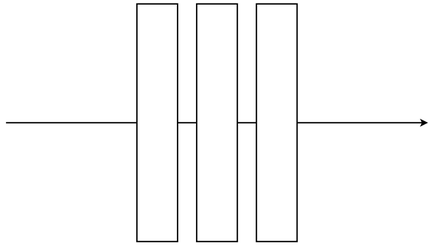
$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(G(\mathbf{x})) \quad + \quad \log(1 - D(\mathbf{y}))\;]$$

Usually loss functions check if output matches a target *instance*

GAN loss checks if output is part of an admissible *set*

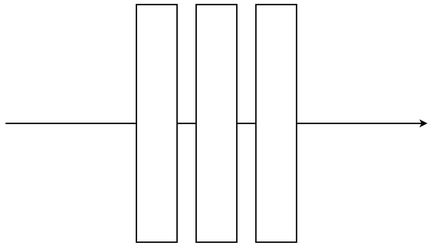
$\mathbf{x}$

G

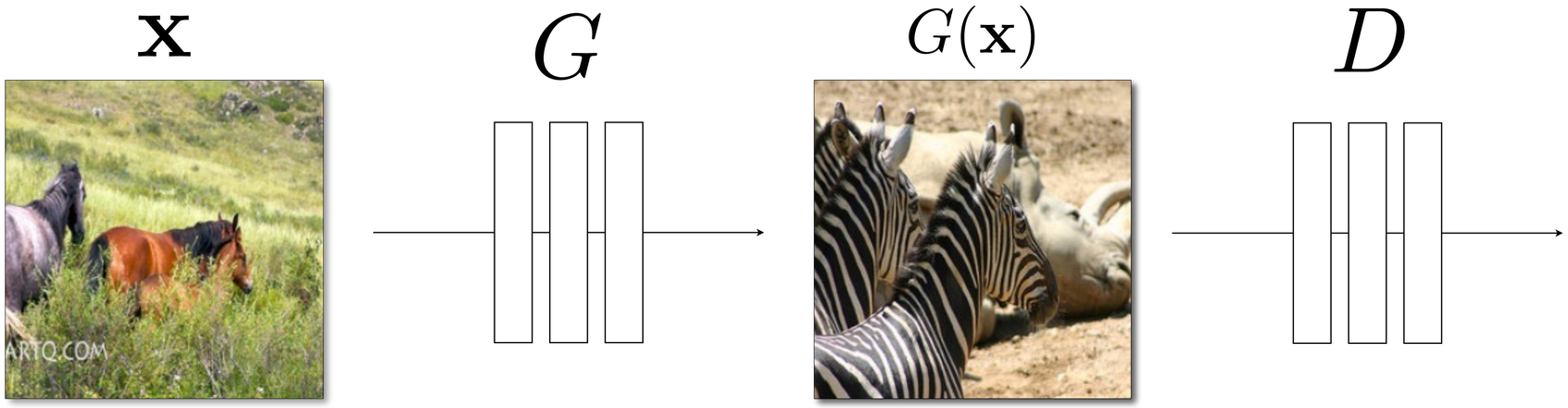


$G(\mathbf{x})$

D

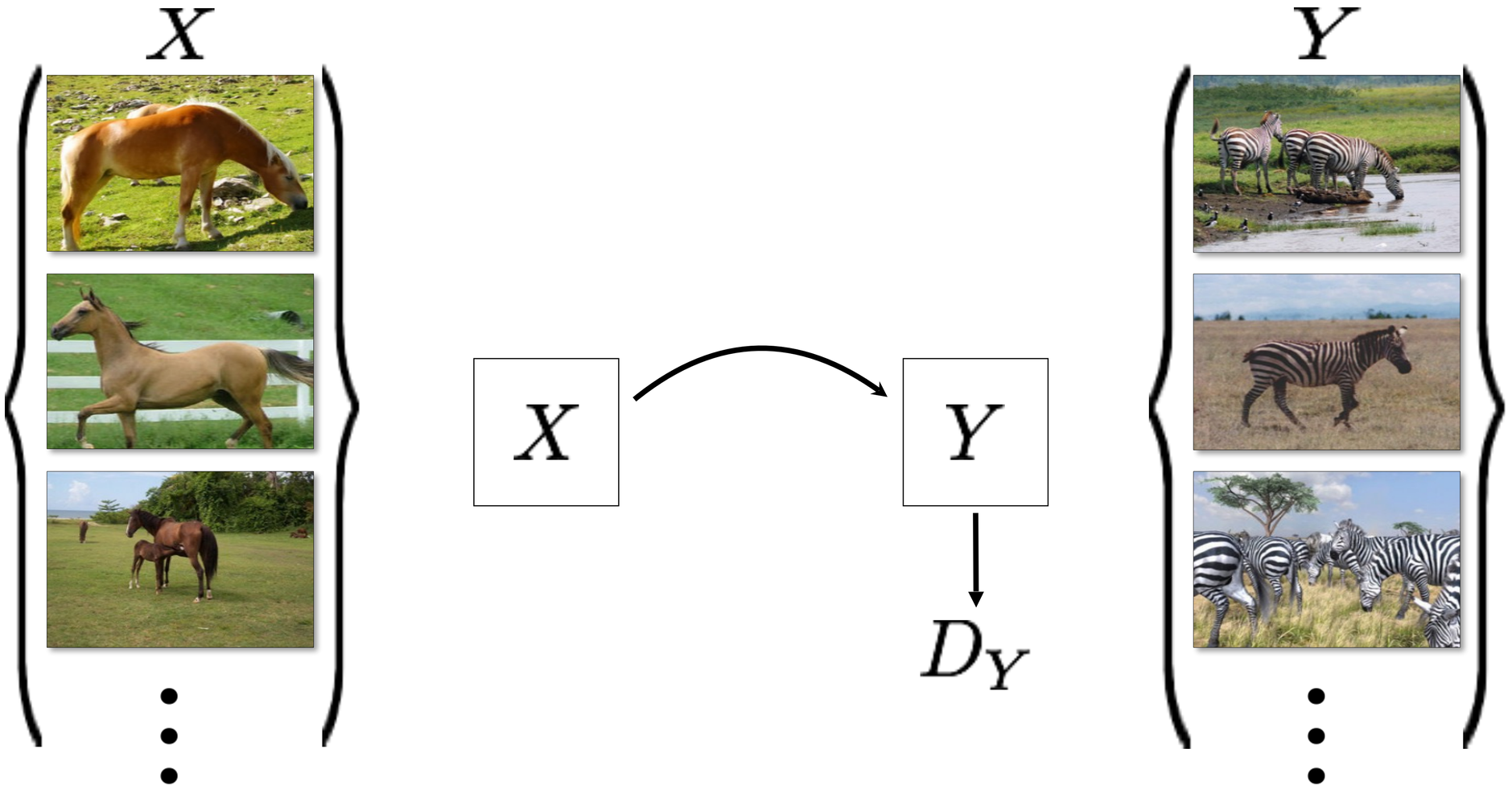


Real!



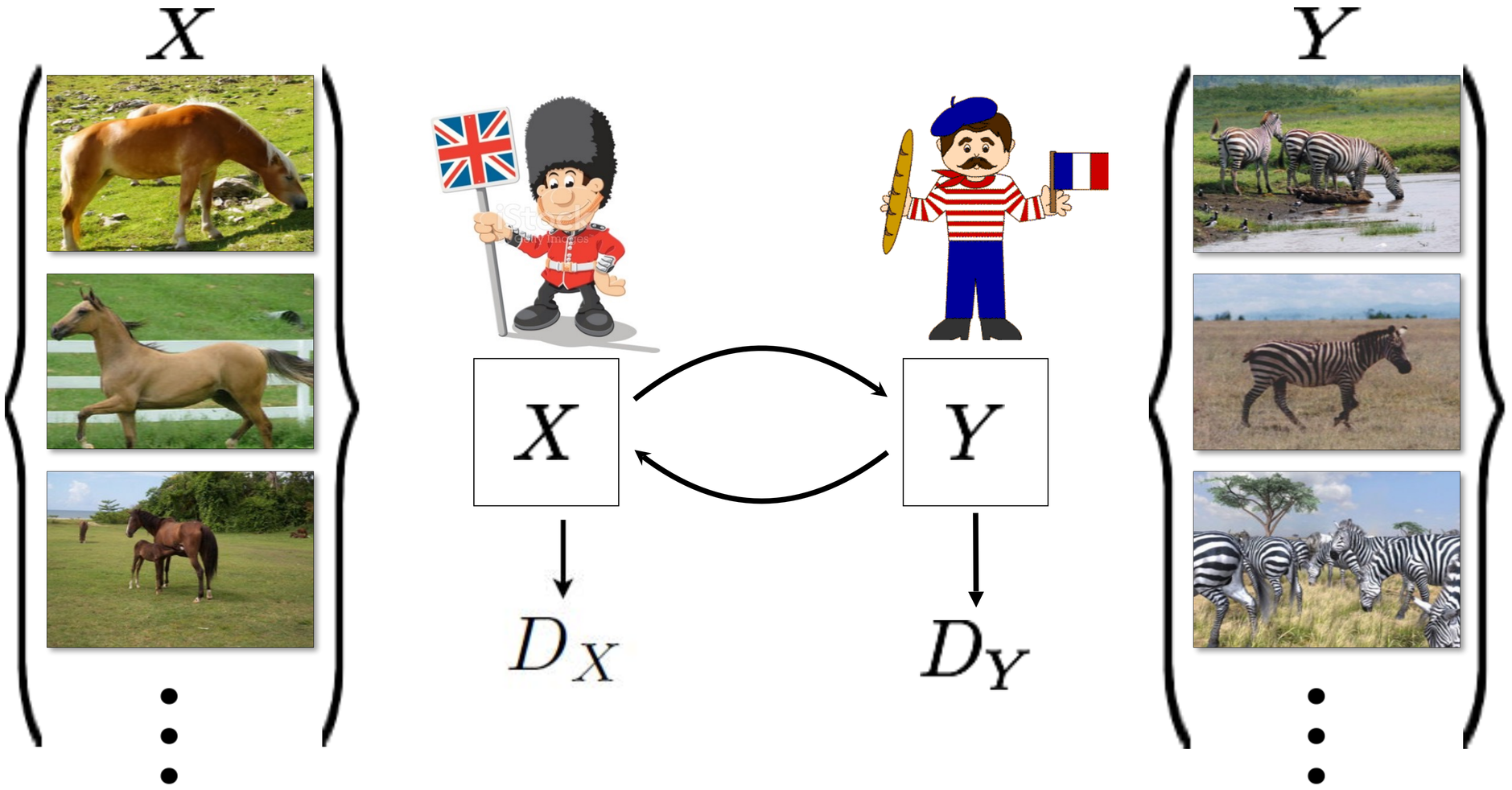
Nothing to force output to correspond to input

Cycle-Consistent Adversarial Networks

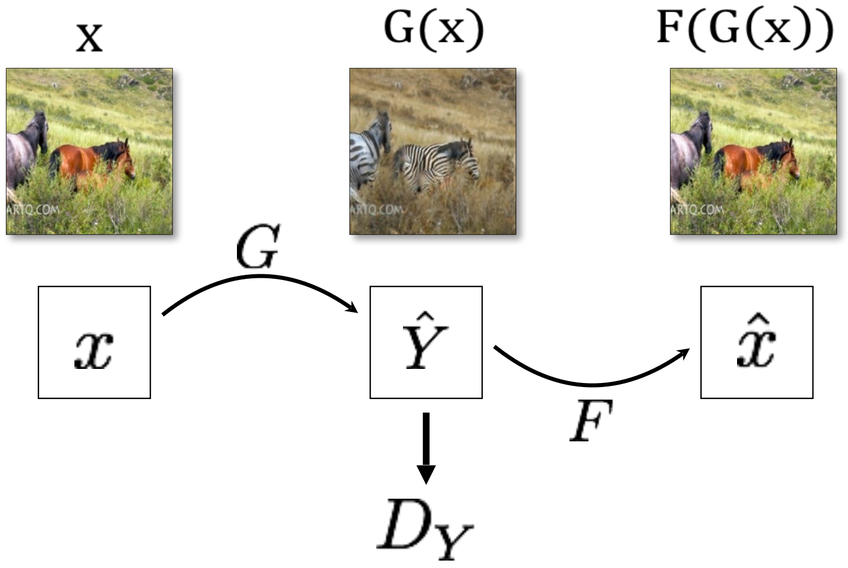


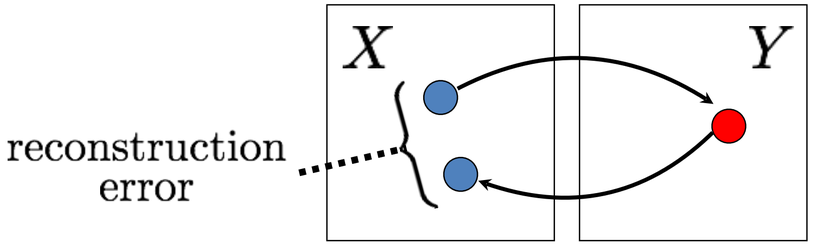
[Zhu et al. 2017], [Yi et al. 2017], [Kim et al. 2017]

Cycle-Consistent Adversarial Networks



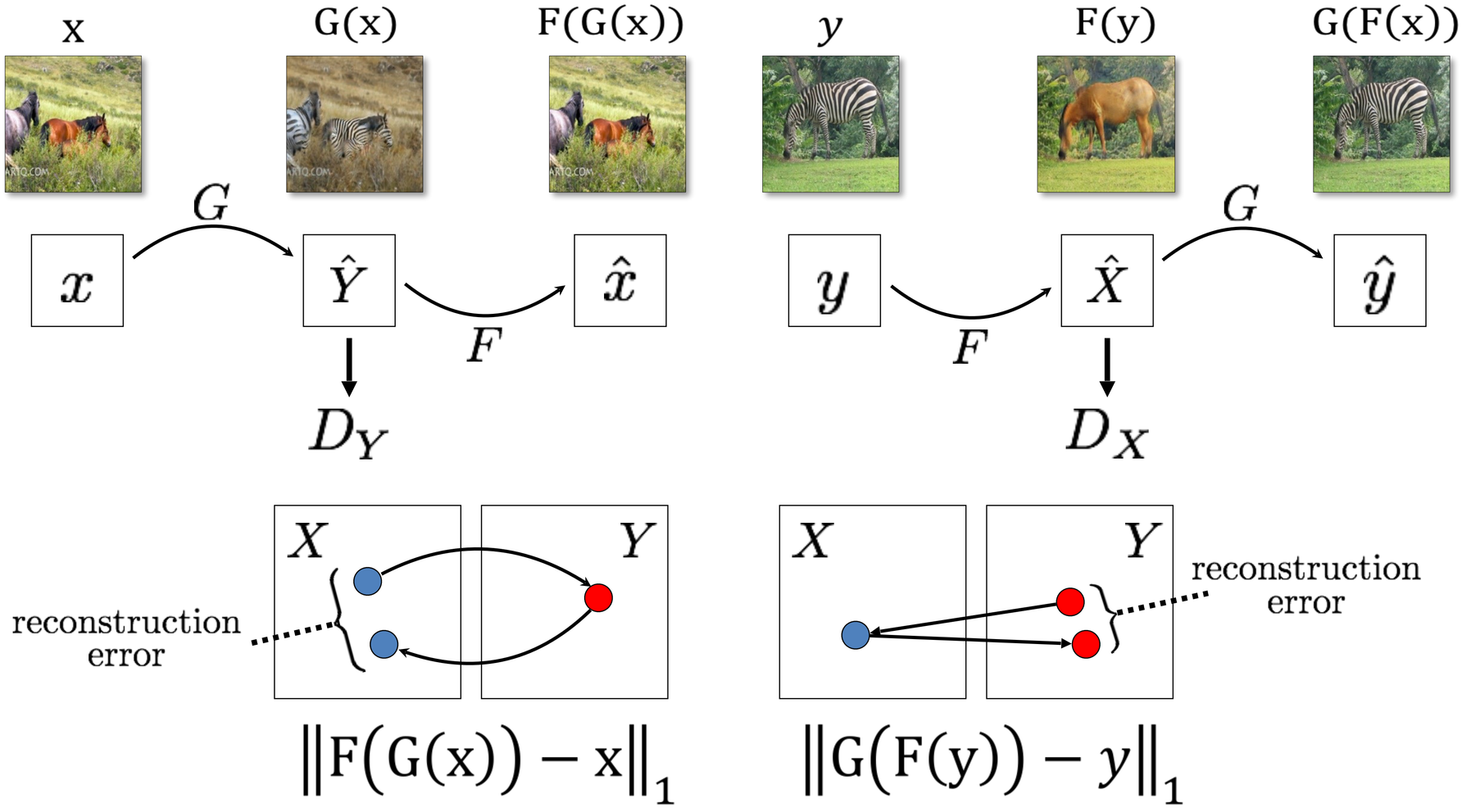
Cycle Consistency Loss



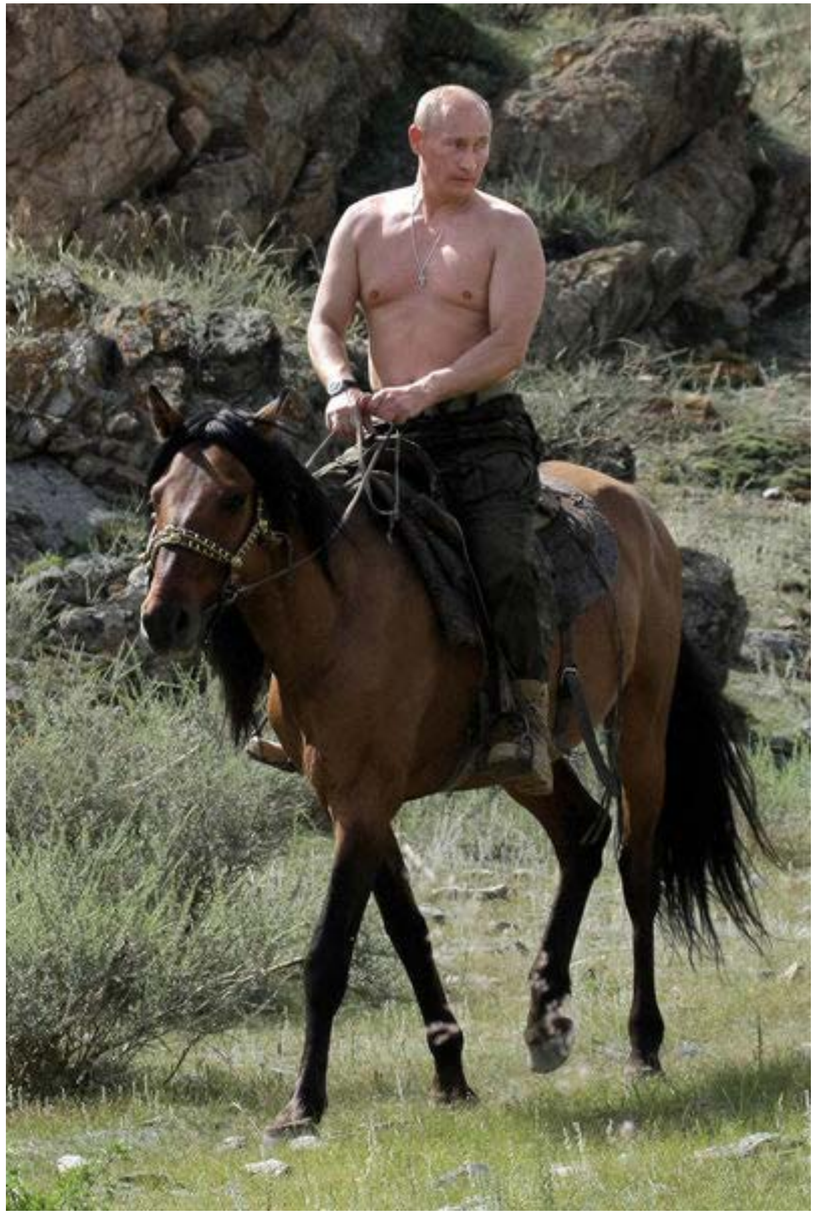


$$\|F(G(x)) - x\|_1$$

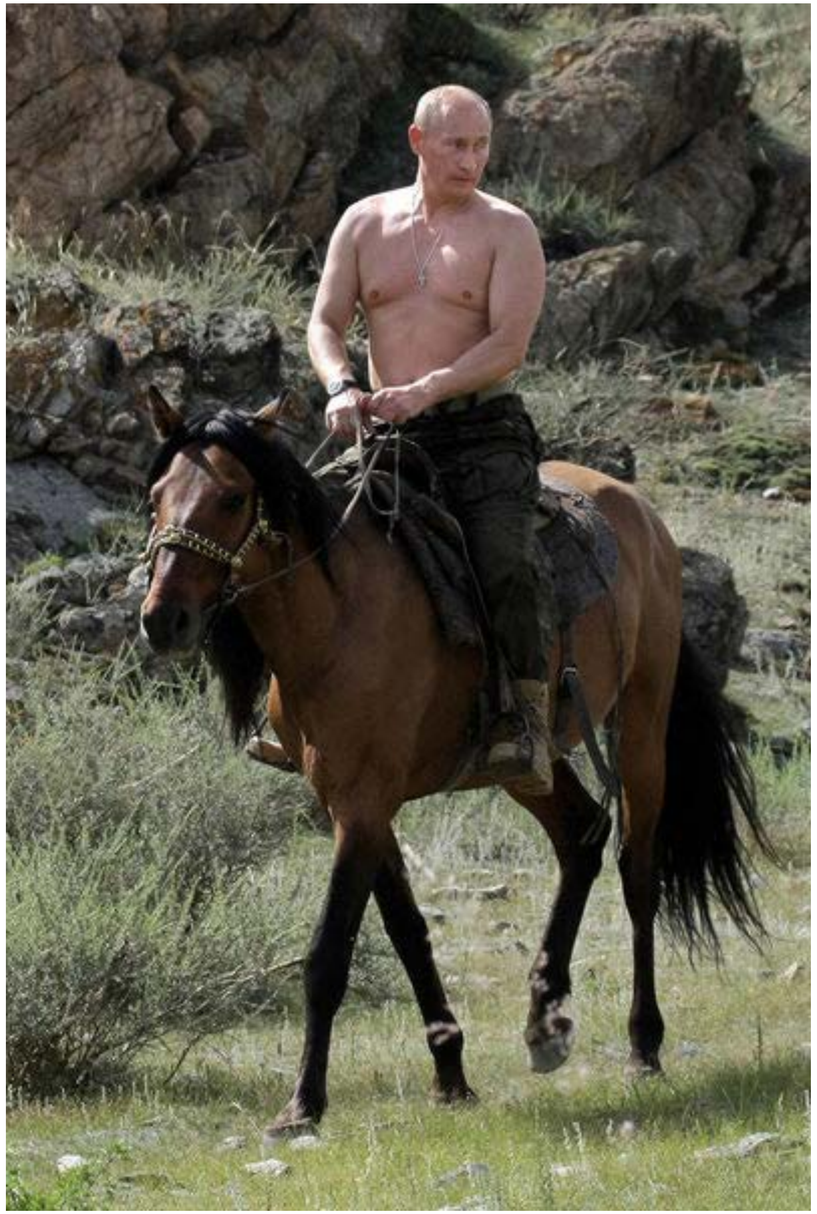
Cycle Consistency Loss



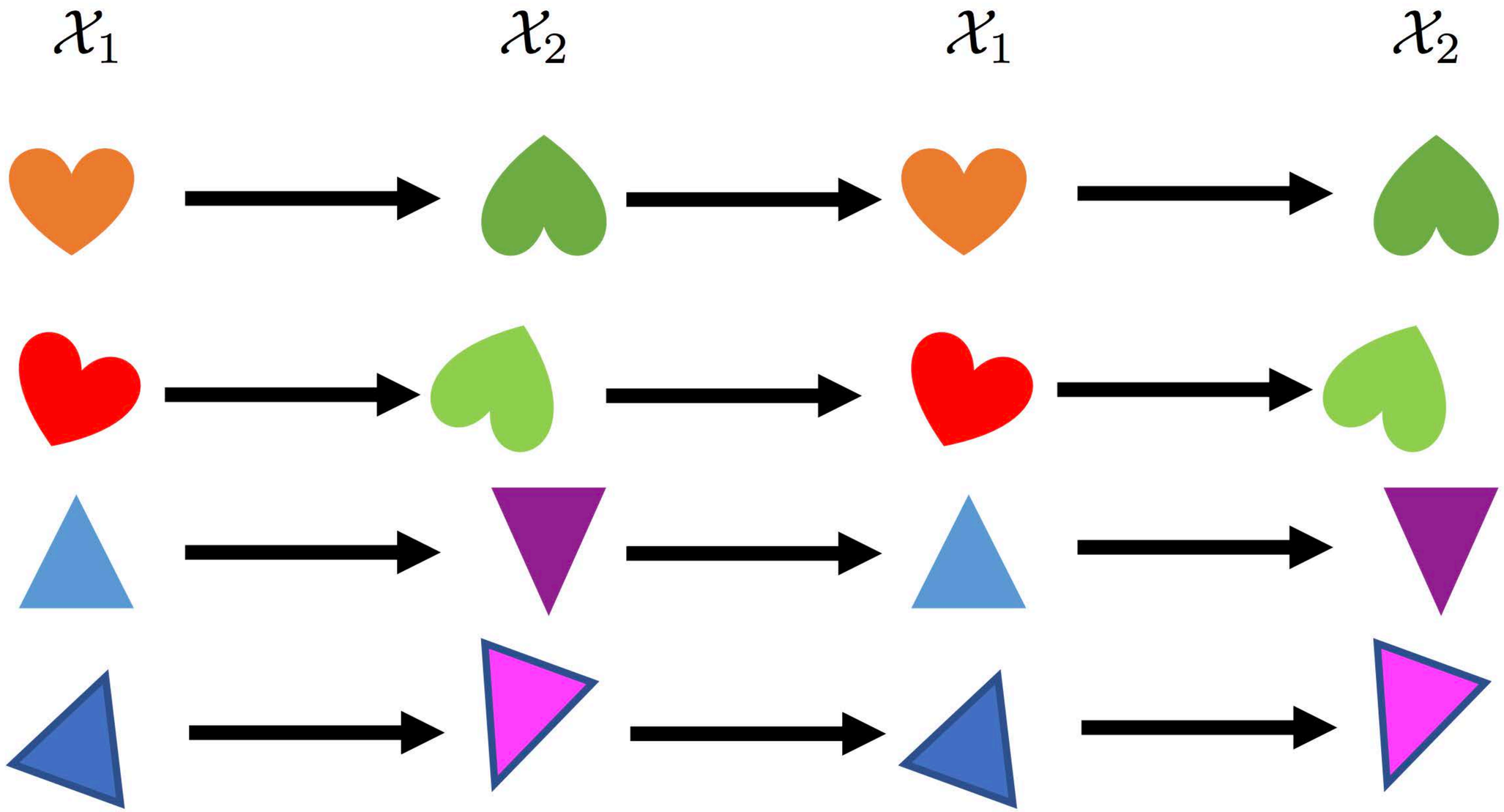
Failure case



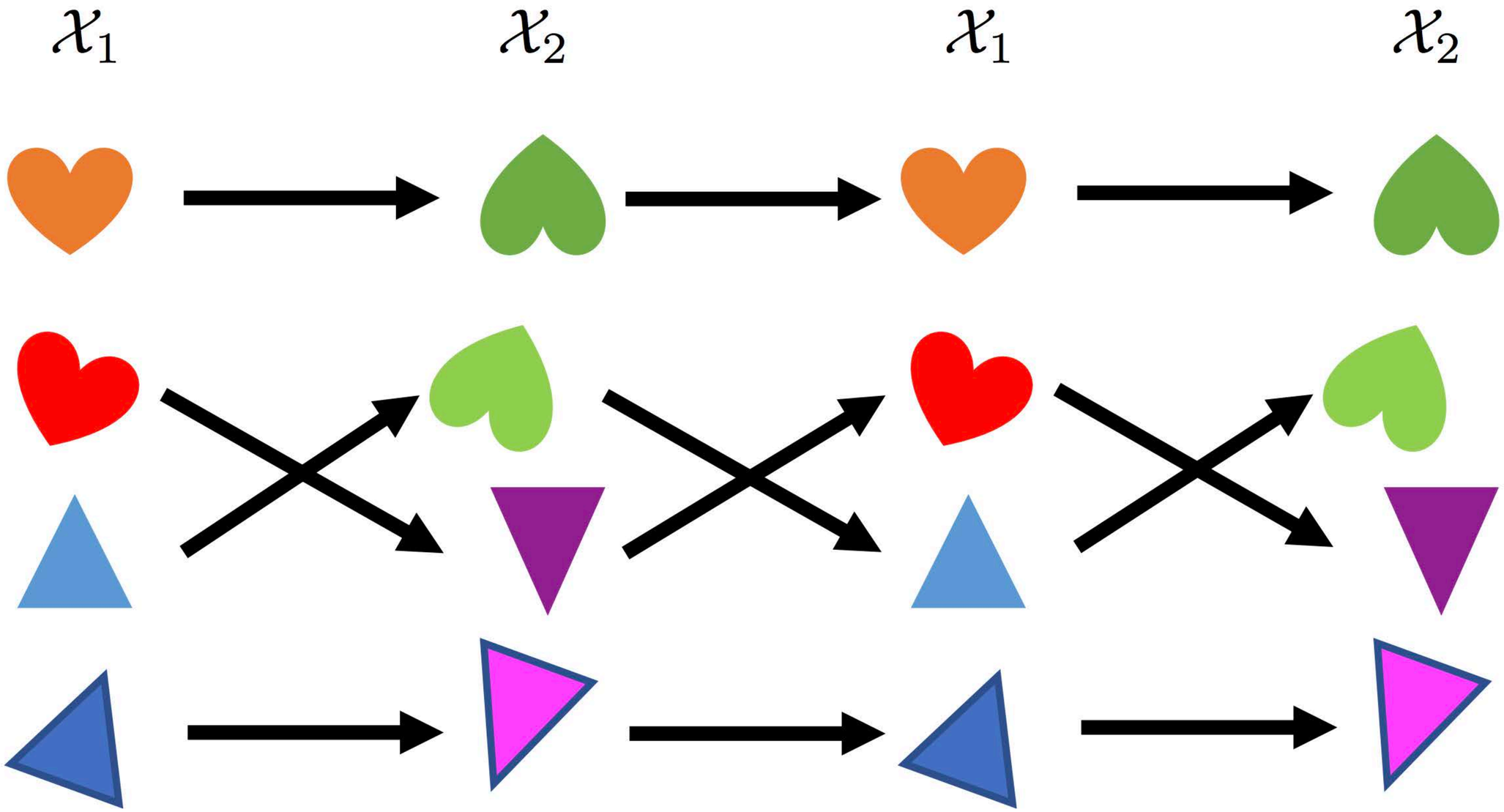
Failure case



Why does CycleGAN work?



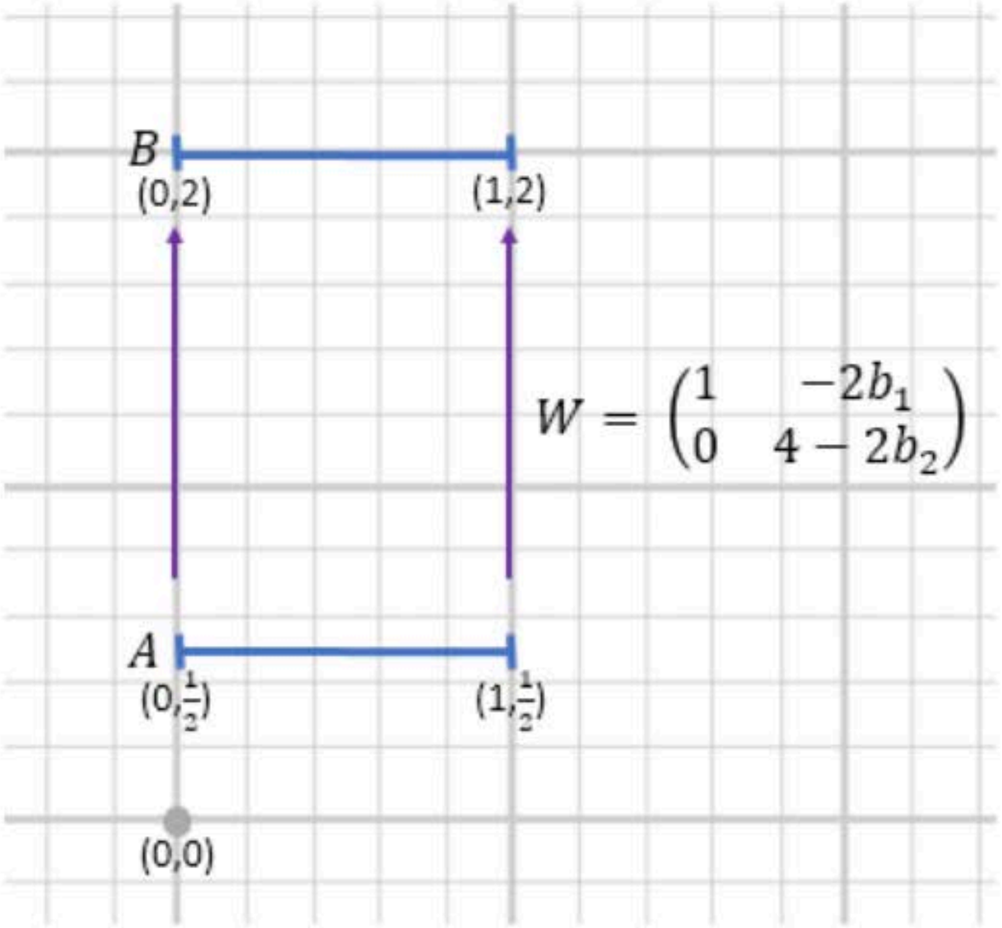
Slide credit: Ming-Yu Liu

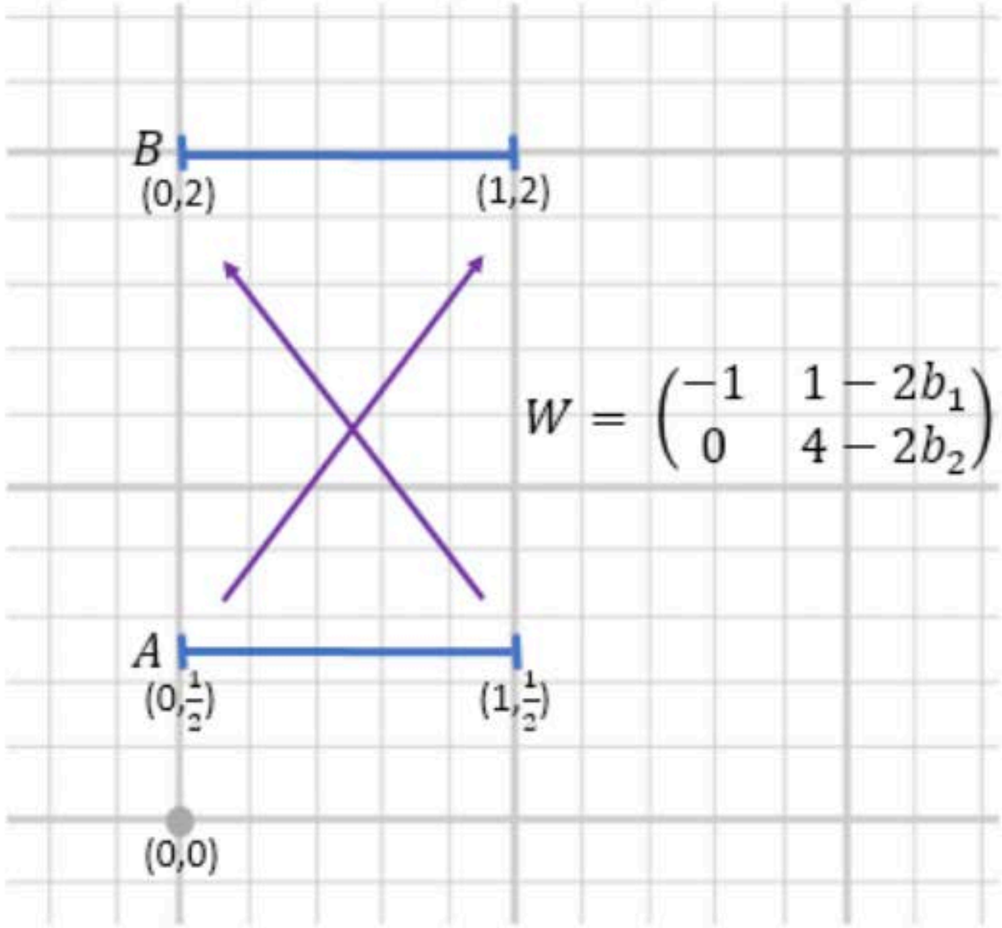


Slide credit: Ming-Yu Liu

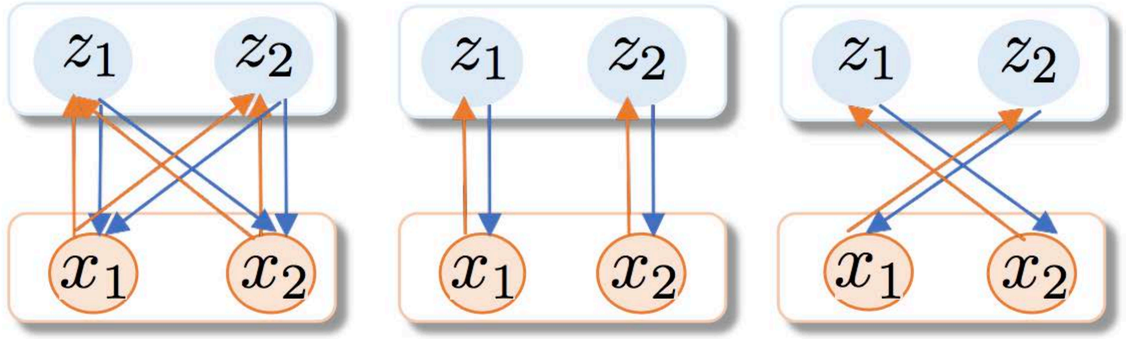
Simplicity hypothesis

[Galanti, Wolf, Benaim, 2018]





Cycle Loss **upper bounds** Conditional Entropy



	z_1	z_2
x_1	$\delta/2$	$(1-\delta)/2$
x_2	$(1-\delta)/2$	$\delta/2$

	z_1	z_2
x_1	1/2	0
x_2	0	1/2

	z_1	z_2
x_1	0	1/2
x_2	1/2	0

High Conditional Entropy

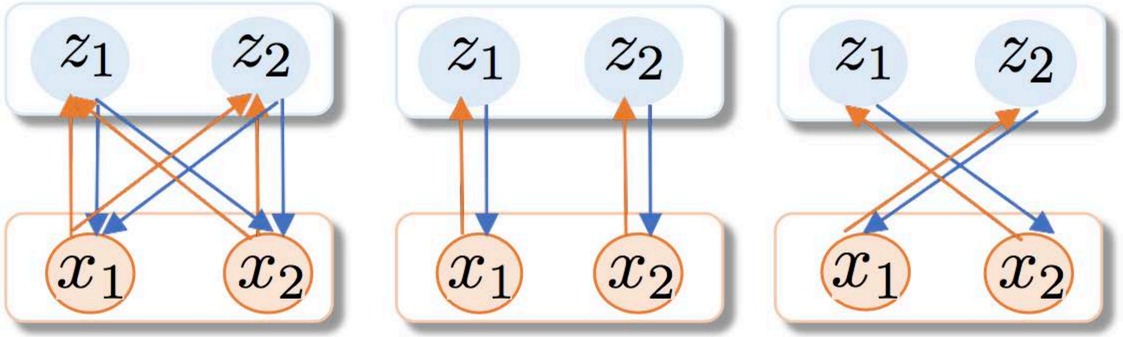
Low Conditional Entropy

Conditional Entropy

$$H^{\pi}(\boldsymbol{x}|\boldsymbol{z}) \triangleq -\mathbb{E}_{\pi(\boldsymbol{x},\boldsymbol{z})}[\log \pi(\boldsymbol{x}|\boldsymbol{z})]$$

"ALICE: Towards Understanding Adversarial Learning for Joint Distribution Matching" [Li et al. NIPS 2017]. Also see [Tiao et al. 2018] "CycleGAN as Approximate Bayesian Inference"

Cycle Loss **upper bounds** Conditional Entropy



	z_1	z_2
x_1	$\delta/2$	$(1-\delta)/2$
x_2	$(1-\delta)/2$	$\delta/2$

	z_1	z_2
x_1	1/2	0
x_2	0	1/2

	z_1	z_2
x_1	0	1/2
x_2	1/2	0

Conditional Entropy

$$H^{\pi}(\boldsymbol{x}|\boldsymbol{z}) \triangleq -\mathbb{E}_{\pi(\boldsymbol{x},\boldsymbol{z})}[\log \pi(\boldsymbol{x}|\boldsymbol{z})]$$

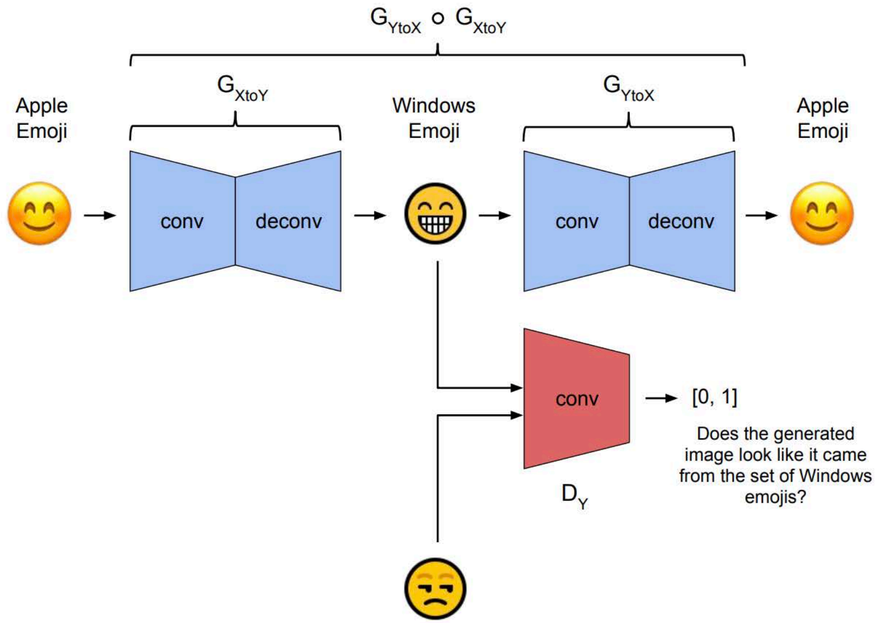
Lemma 3 *For joint distributions $p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})$ or $q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})$, we have*

$$\begin{aligned} H^{q_{\boldsymbol{\phi}}}(\boldsymbol{x}|\boldsymbol{z}) &\triangleq -\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})}[\log q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})] = -\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})}[\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}))] \\ &\leq -\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] \triangleq \mathcal{L}_{\mathrm{Cycle}}(\boldsymbol{\theta},\boldsymbol{\phi}). \end{aligned} \tag{6}$$

"ALICE: Towards Understanding Adversarial Learning for Joint Distribution Matching" [Li et al. NIPS 2017]. Also see [Tiao et al. 2018] "CycleGAN as Approximate Bayesian Inference"

CycleGAN at School

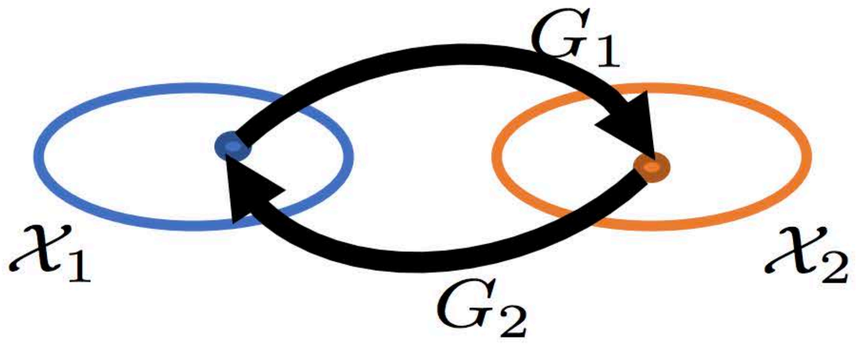
- Course assignment code and handout designed by Prof. Roger Grosse for CSC321 “Intro to Neural Networks and Machine Learning” at University of Toronto.



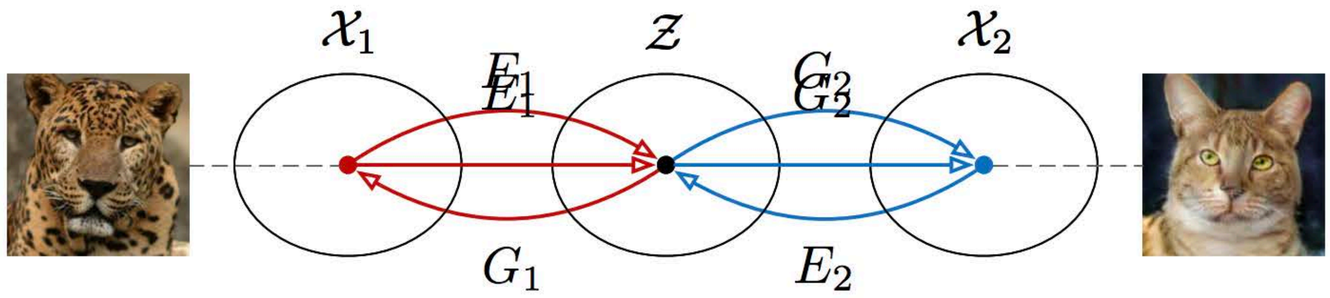

```
##   FILL THIS IN: CREATE ARCHITECTURE   ##
#########################################

# 1. Define the encoder part of the generator
# self.conv1 = ...
# self.conv2 = ...

# 2. Define the transformation part of the generator
# self.resnet_block = ...

# 3. Define the decoder part of the generator
# self.deconv1 = ...
# self.deconv2 = ...
```

Can other generative models do this?

CycleGAN

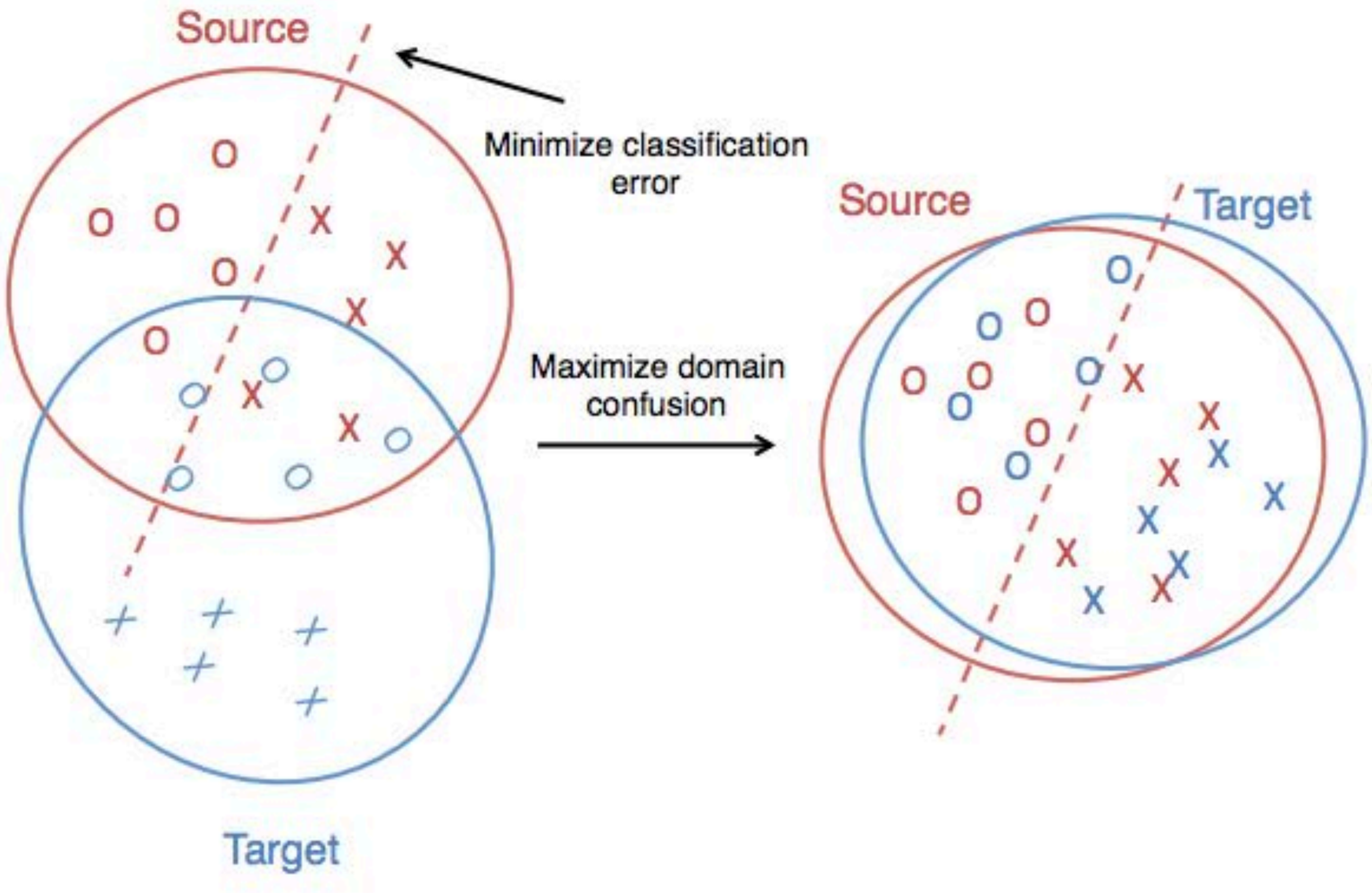


UNIT



Unsupervised image-to-image translation networks
[Liu, Breuel, Kautz, 2017]

Domain Adaptation



[Tzeng et al. 2014]

Sim2real

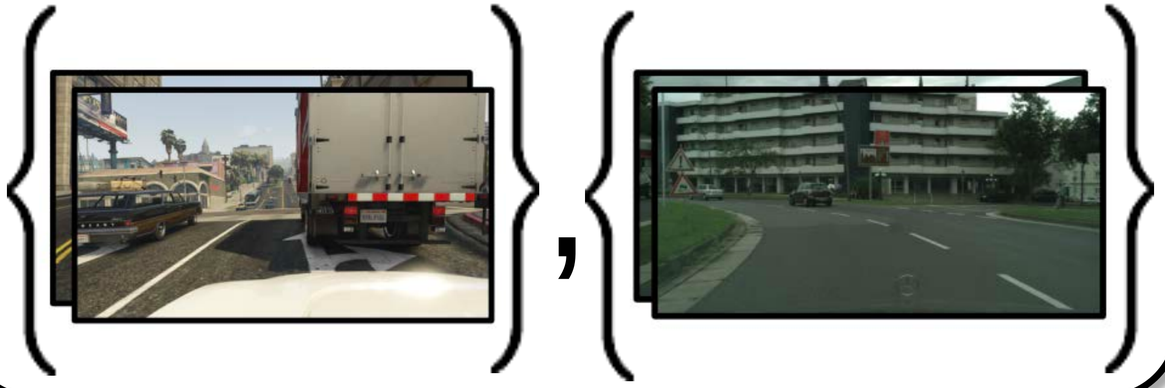
Simulated data

Real data

[Richter*, Vineet* et al. 2016] [Krähenbühl et al. 2018]

CycleGAN

Training data

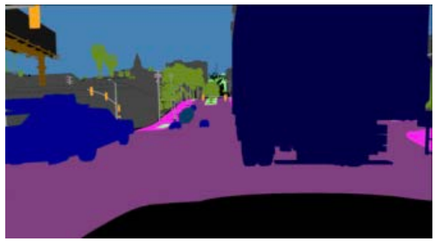


[Hoffman, Tzeng, Park, Zhu, Isola, Saenko, Darrell, Efros, 2018]

CycleGAN

Training data

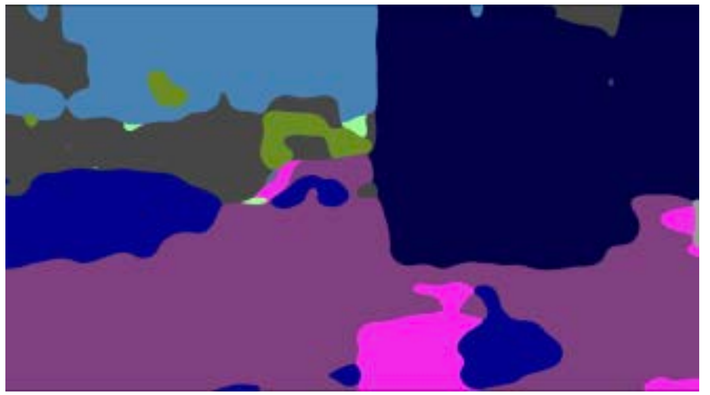
,



[Hoffman, Tzeng, Park, Zhu, Isola, Saenko, Darrell, Efros, 2018]

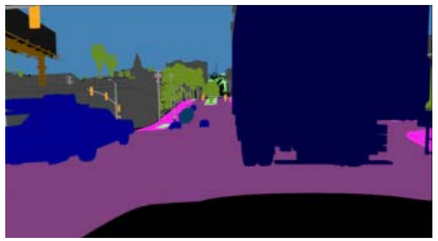
CycleGAN

FCN



Training data

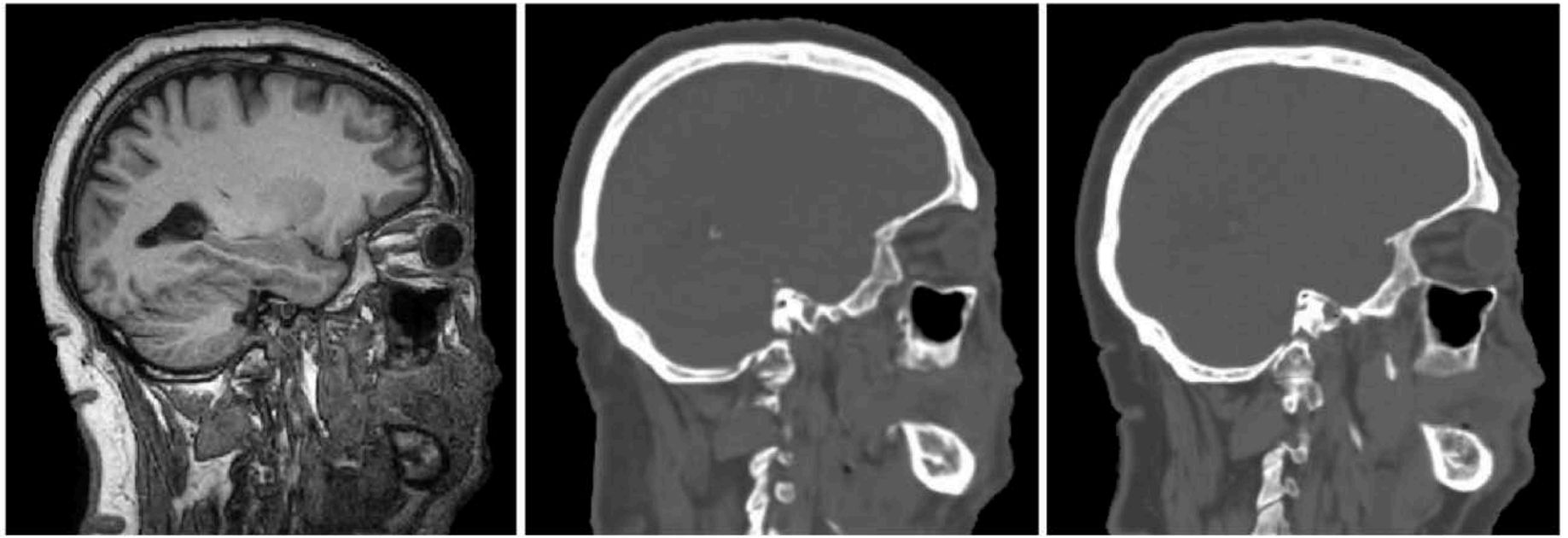
,



[Hoffman, Tzeng, Park, Zhu, Isola, Saenko, Darrell, Efros, 2018]

Medical domain adaptation

MR → CT [Wolterink et al] arxiv: 1708.01155



Input MR | Generated CT | Ground truth CT

- MRI reconstruction [Quan et al.] arxiv:1709.00753
- Cardiac MR images from CT [Chartsias et al. 2017]

What can you do with generative models?

1. Data prediction

2. **Domain mapping**

3. Representation learning

4. Model-based intelligence

What can you do with generative models?

1. Data prediction

2. Domain mapping

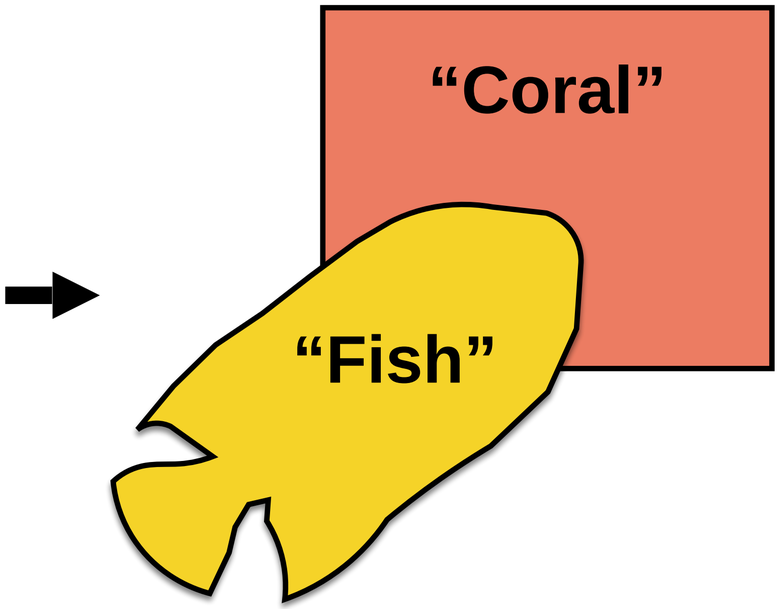
3. **Representation learning**

4. Model-based intelligence

Representation Learning

$\mathbf{x}$

Image



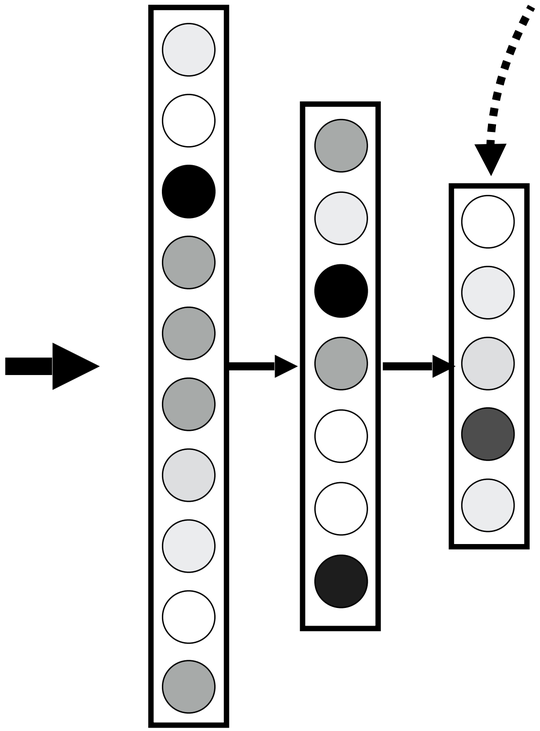
Compact mental representation

Representation Learning

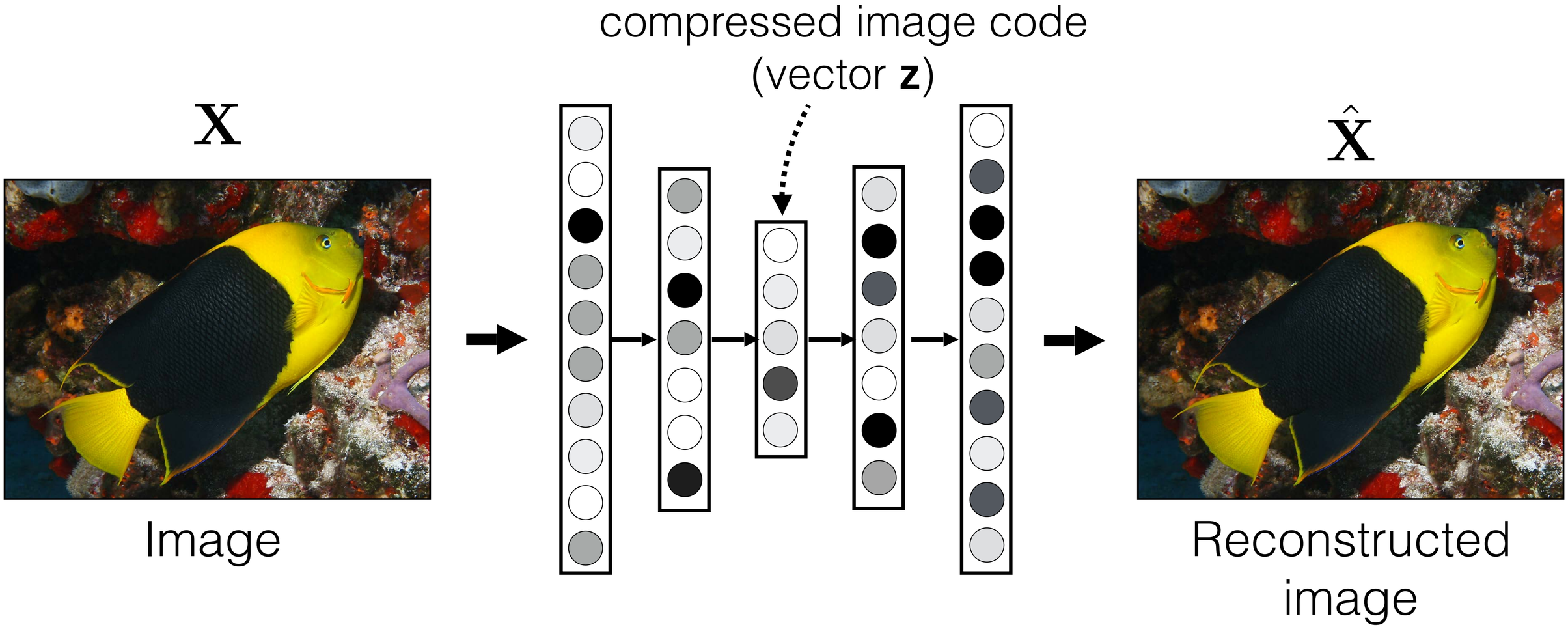
compressed image code
(vector **z**)

x

Image



Representation Learning



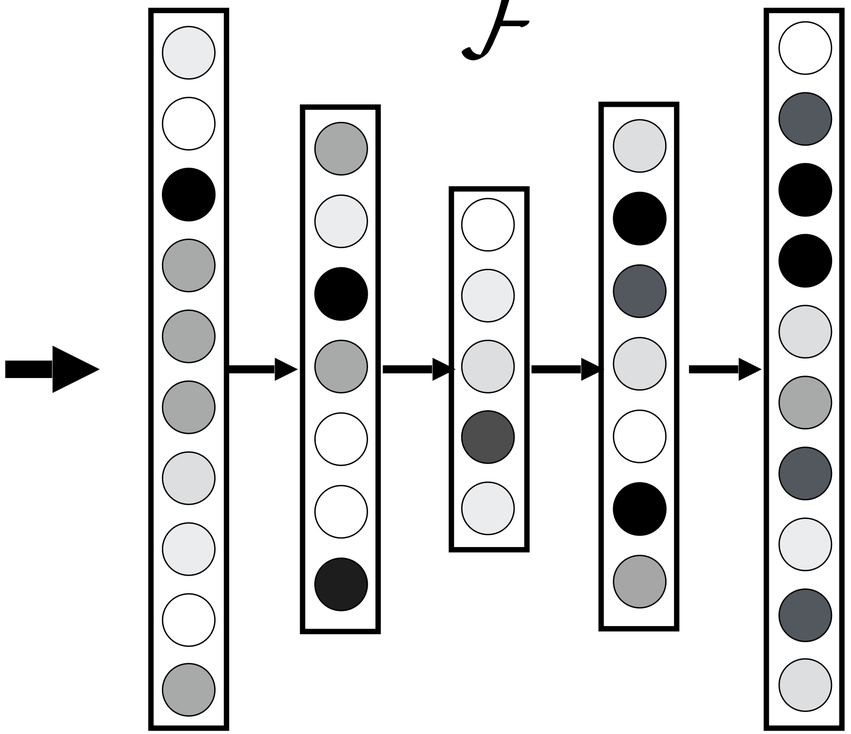
Autoencoder

Autoencoder

$\mathbf{X}$

Image

$\mathcal{F}$



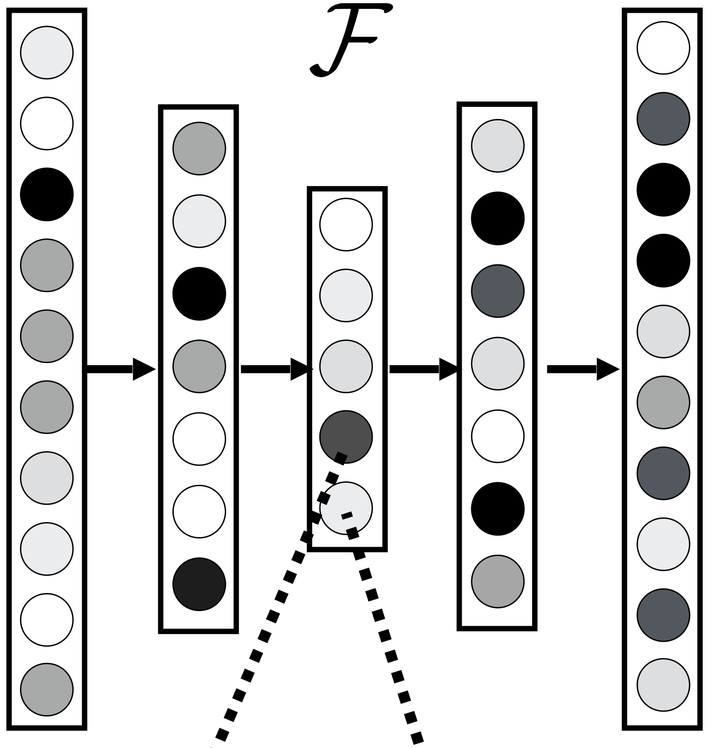
$\hat{\mathbf{X}} = \mathcal{F}(\mathbf{X})$

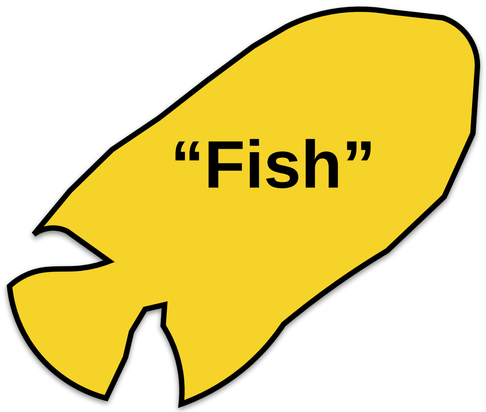
Reconstructed image

$\mathbf{X}$

Image

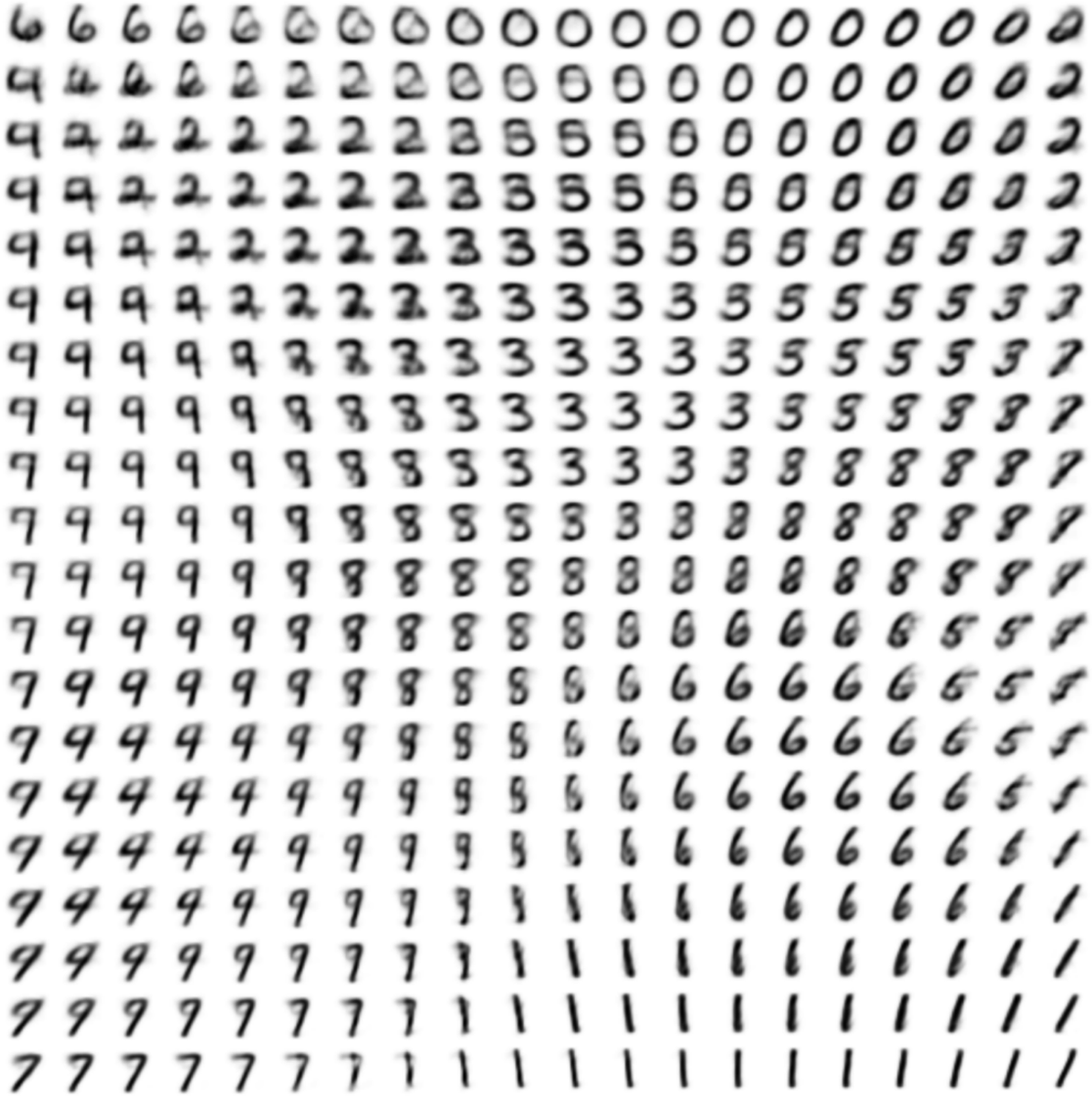
$\mathcal{F}$





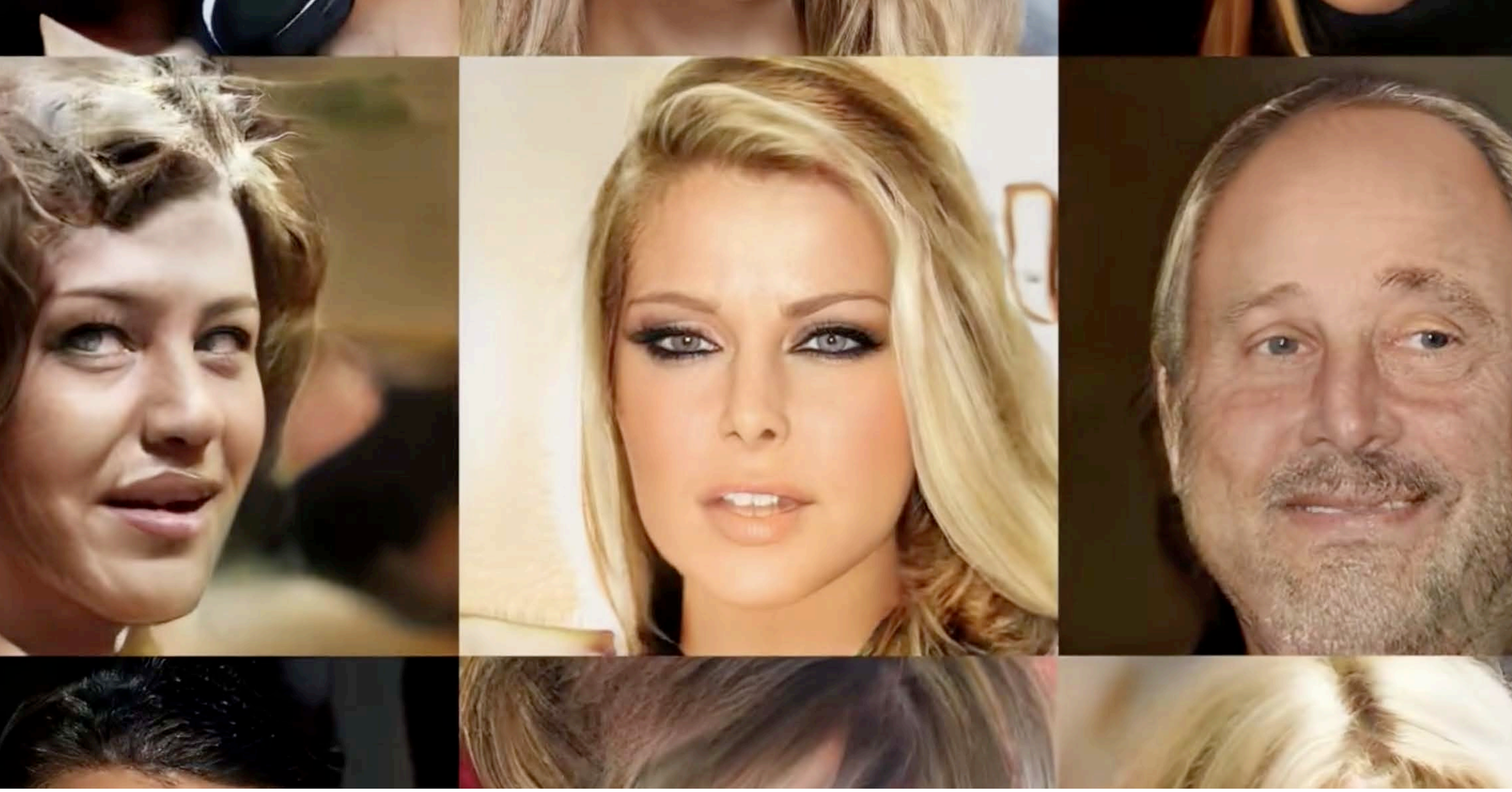
$\hat{\mathbf{X}} = \mathcal{F}(\mathbf{X})$

Reconstructed image

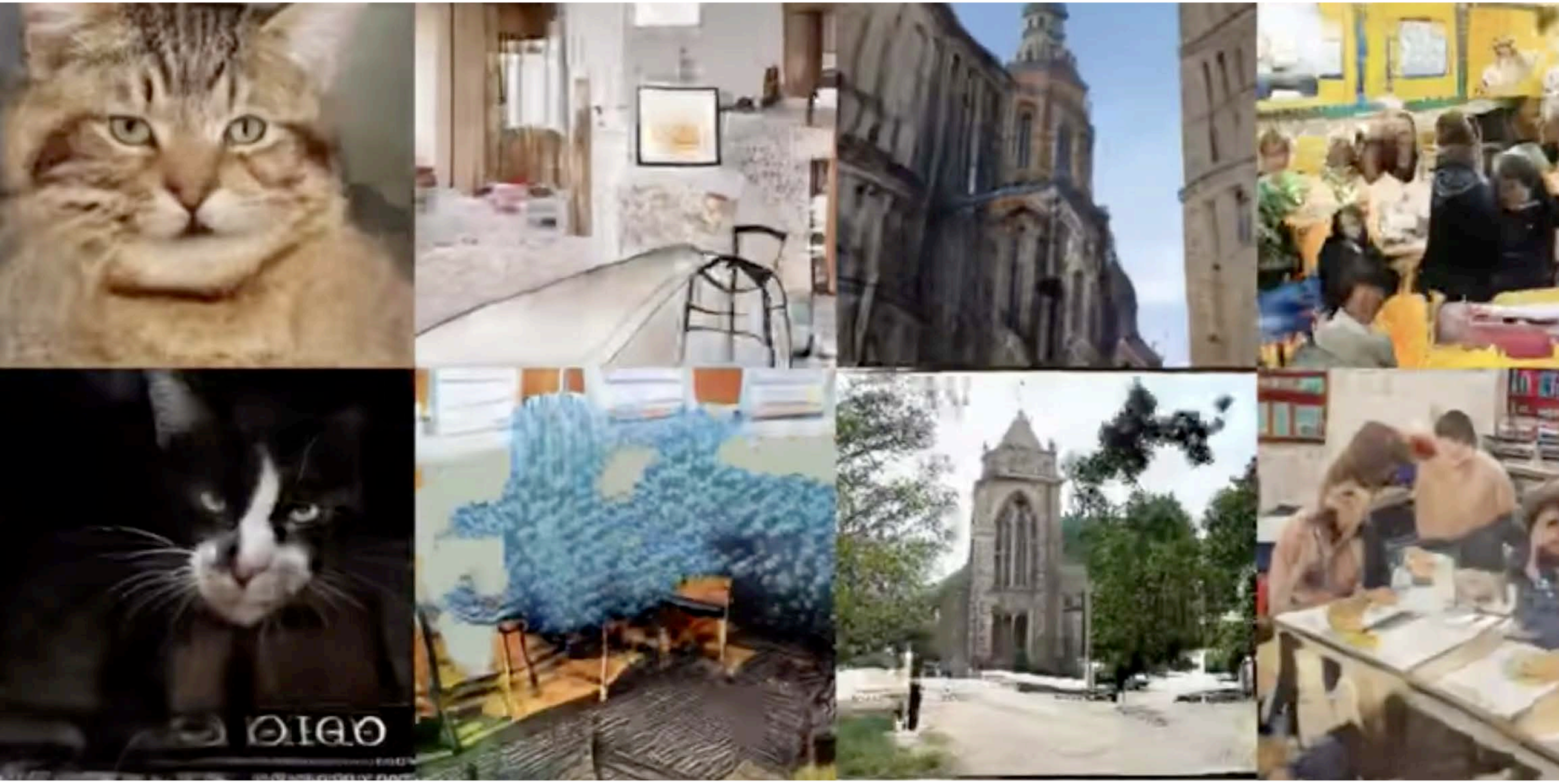


MNIST manifold learned by a VAE

[Kingma & Welling, 2014]

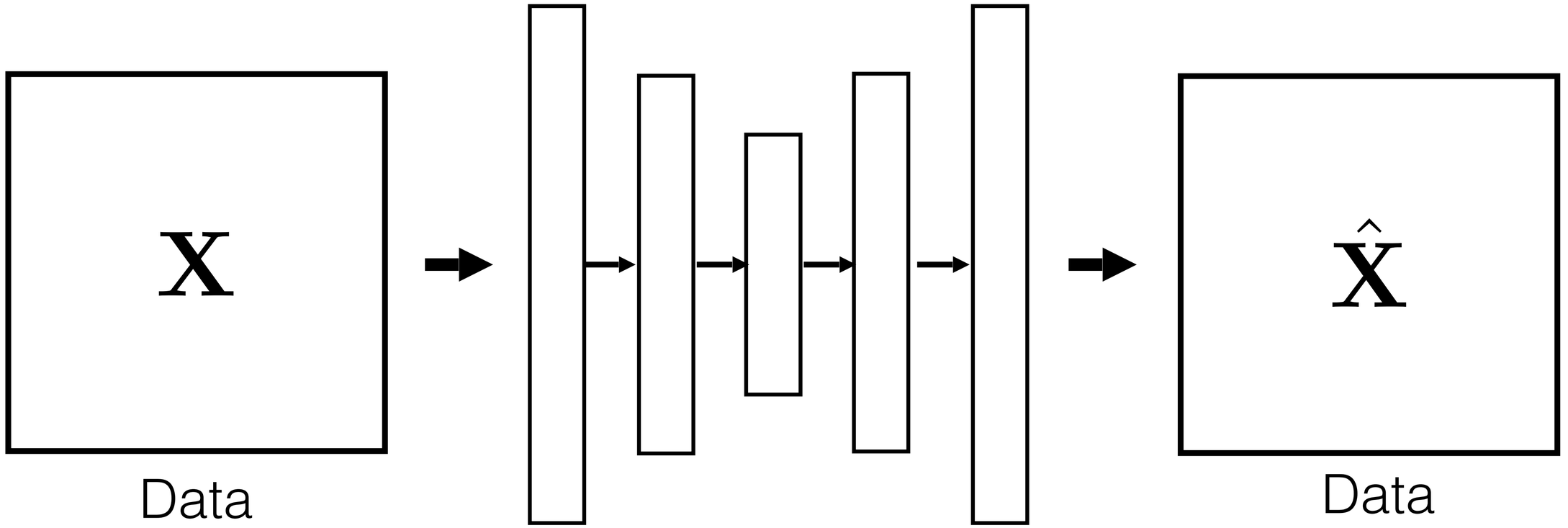


Progressive GAN [Karras et al., 2018]

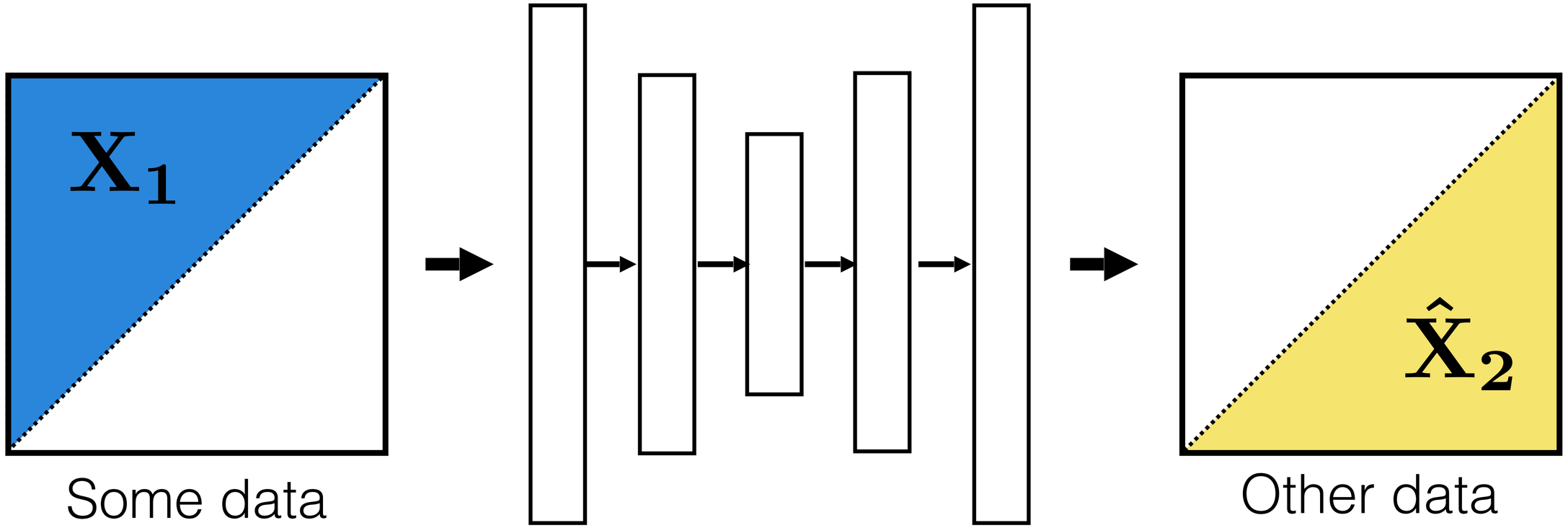


Progressive GAN [Karras et al., 2018]

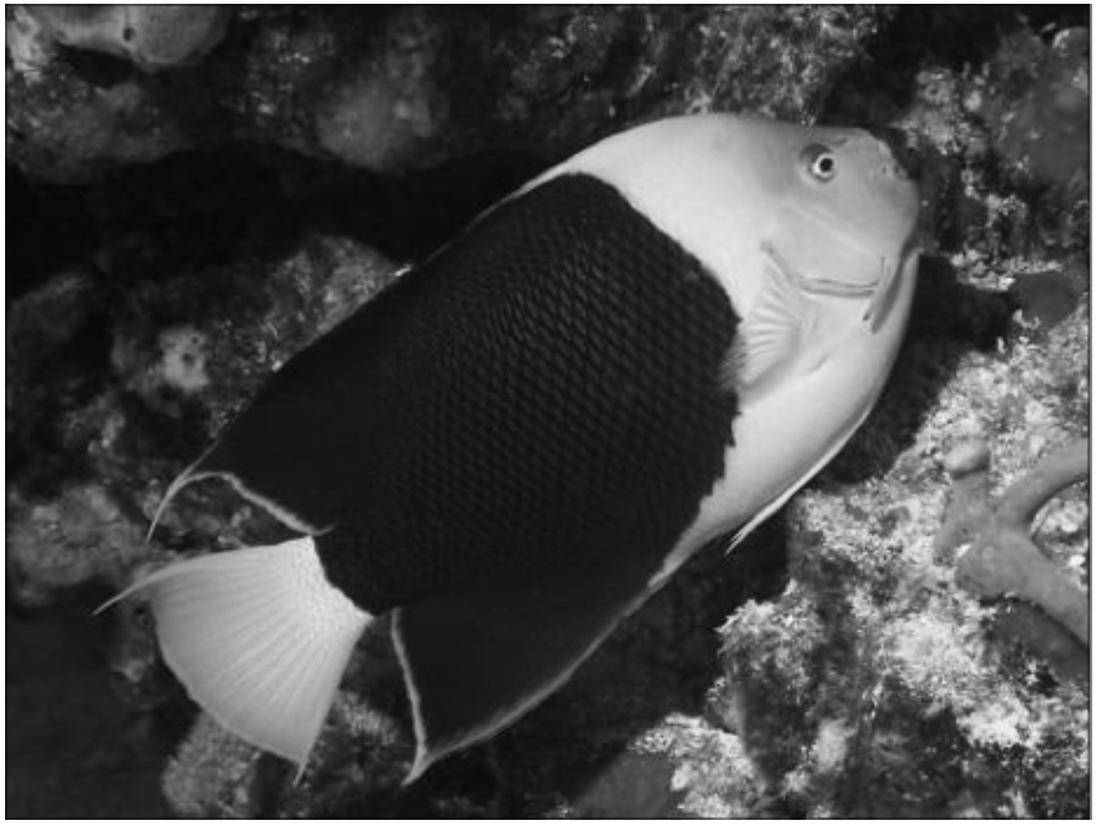
Data compression



Data prediction



see also “Predictive coding”, Denoising autoencoders [Vincent et al., 2008]



$\mathcal{F}$

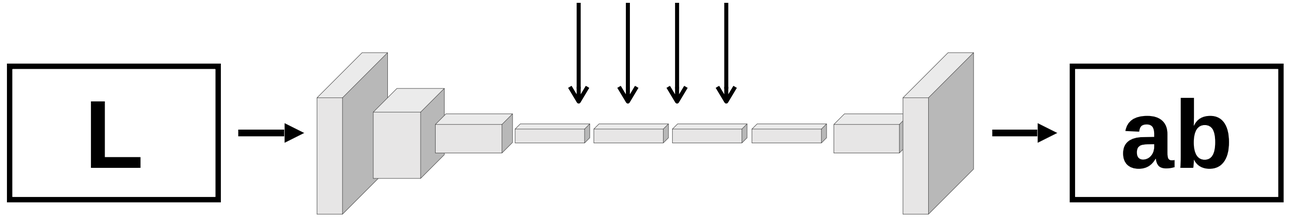
Grayscale image: L cha

$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$

Semantics? Higher-level abstraction?

information: ab channels

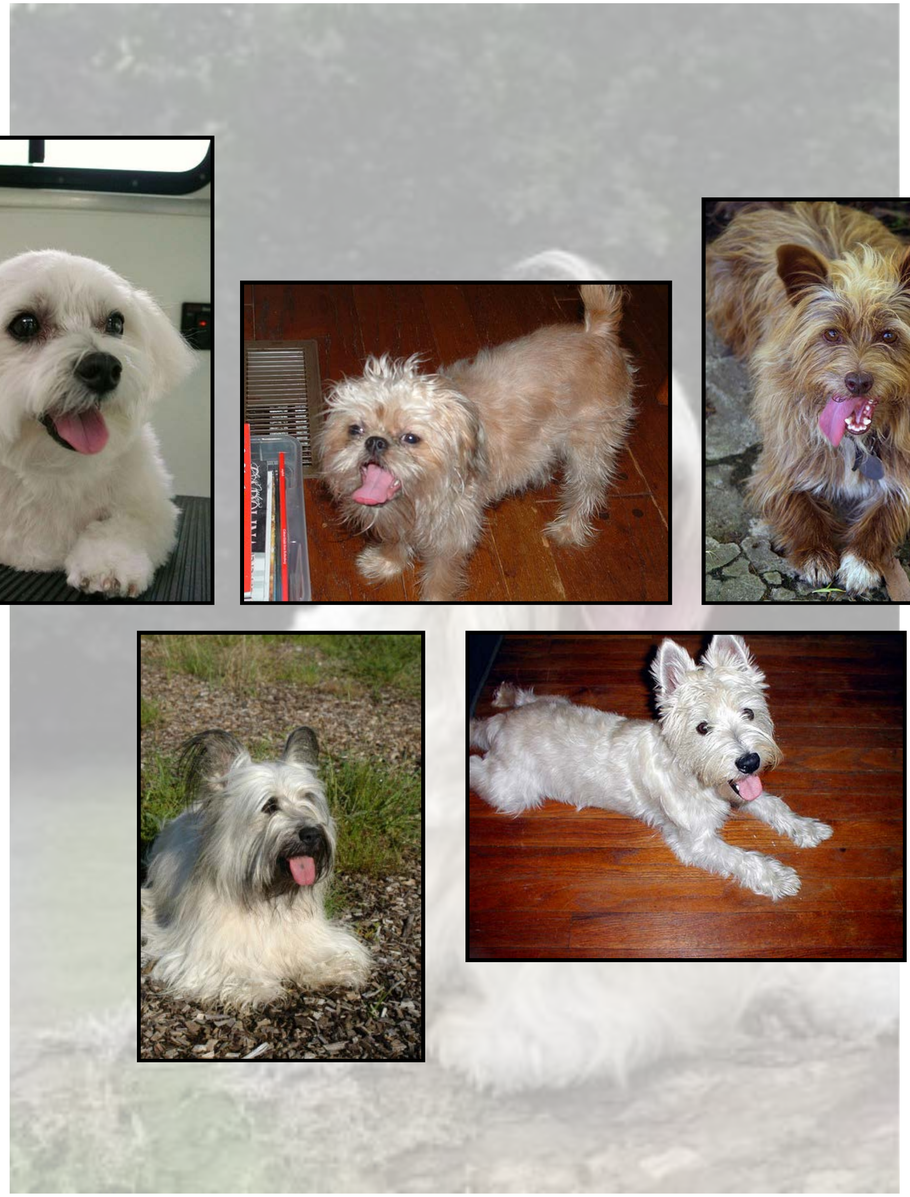
$\hat{\mathbf{Y}} \in \mathbb{R}^{H \times W \times 2}$



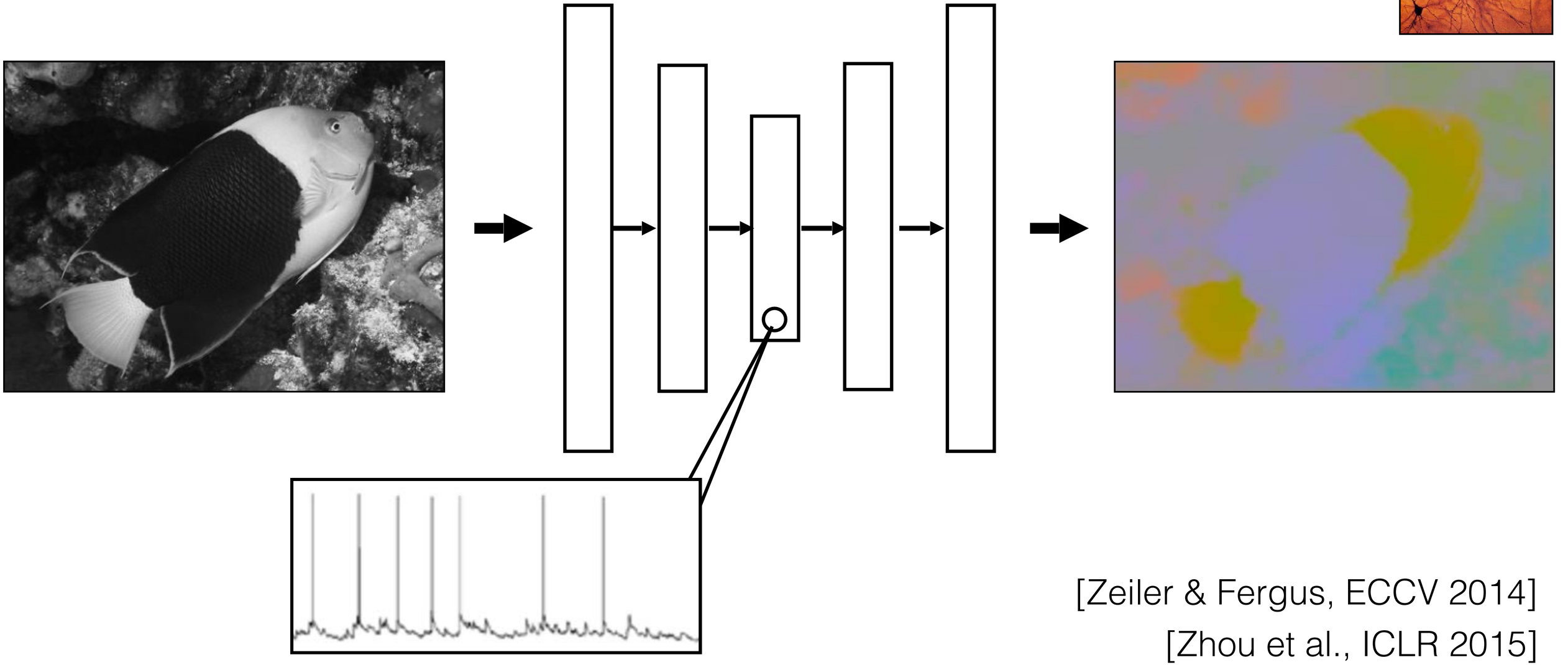
[Zhang, Isola, Efros, ECCV 2016]

Instructive failure

Instructive failure



Deep Net “Electrophysiology”



[Zeiler & Fergus, ECCV 2014]

[Zhou et al., ICLR 2015]

Stimuli that drive selected neurons (conv5 layer)

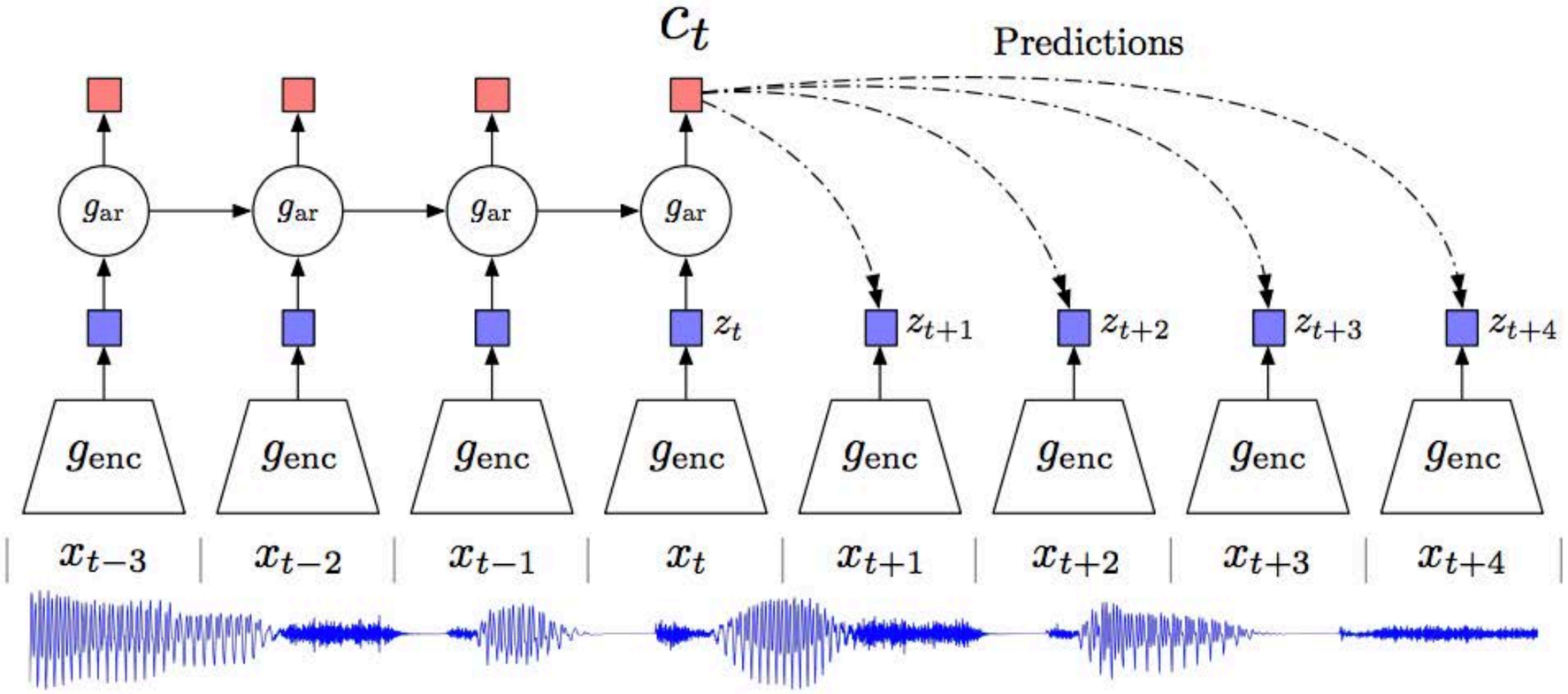
faces

dog faces

flowers

Representation Learning with Contrastive Predictive Coding

[van den Oord, Li, and Vinyals, 2018]

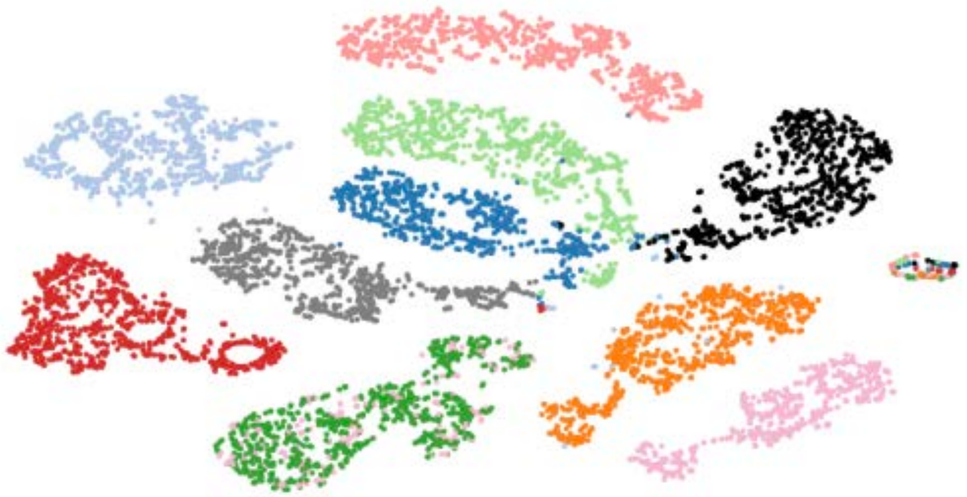


$$I(x;c) = \sum_{x,c} p(x,c) \log \frac{p(x|c)}{p(x)}$$

Representation Learning with Contrastive Predictive Coding

[van den Oord, Li, and Vinyals, 2018]

Sound representation



Each color represents a different speaker

Image representation

Method	Top-1 ACC
Using AlexNet conv5	
Video [27]	29.8
Relative Position [11]	30.4
BiGan [34]	34.8
Colorization [10]	35.2
Jigsaw [28] *	38.1
Using ResNet-V2	
Motion Segmentation [35]	27.6
Exemplar [35]	31.5
Relative Position [35]	36.2
Colorization [35]	39.6
CPC	**48.7**

What can you do with generative models?

1. Data prediction

2. Domain mapping

3. **Representation learning**

4. Model-based intelligence

What can you do with generative models?

1. Data prediction
2. Domain mapping
3. Representation learning
4. **Model-based intelligence**

Model-based intelligence

Predict consequences of my actions.

Then use that predictive model to plan good actions (or learn good policies).

Can be faster, safer, and more abstracted than directly trying actions in real world (or in simulator).

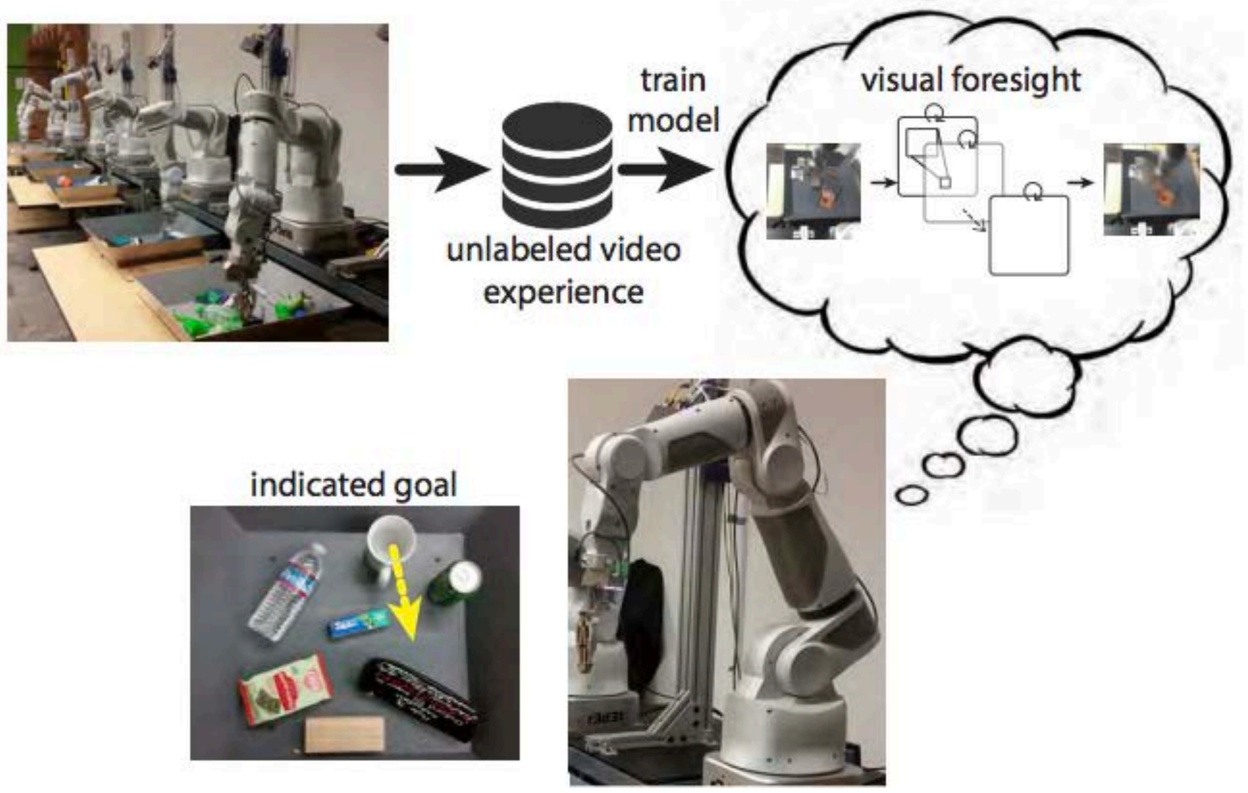
Yann LeCun's cake

Deep Visual Foresight For Robotic Planning
[Finn & Levine 2017]

1. Train model to predict future frame given action

2. Specify target future frame

3. Use prediction model to pick action that maximizes probability of reachin target frame

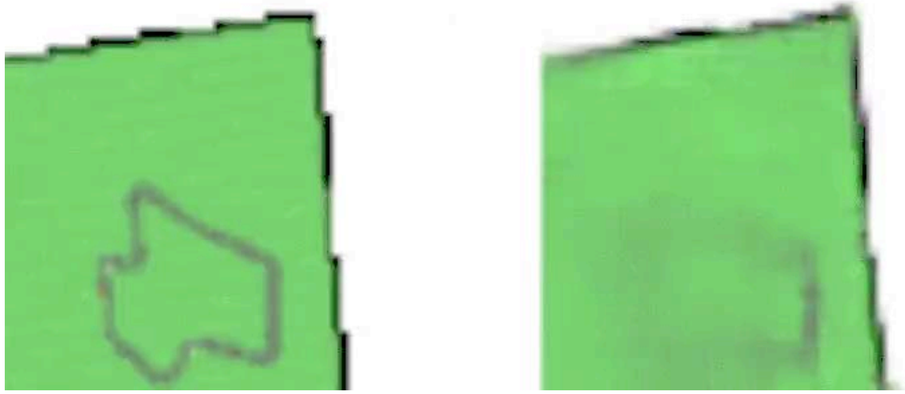


c.f. [Weber et al., 2017], […]

World Models
[Ha & Schmidhuber 2018]

1. Train an RNN to simulate a video game

2. Do policy optimization using the RNN simulate trajectories, rather than the actual game (policy reasons on top of latent space)



Model	Parameter Count
VAE	4,348,547
MDN-RNN	422,368
Controller	867

Incentivizing exploration in reinforcement learning with deep predictive models
[Stadie et al., 2015]

1. Train model to predict next state given action and previous state: $p(s_{t+1}|s_t, a_t)$

2. Reward agent for reaching states that surprise it: $r_t = -\log p(s_{t+1}|s_t, a_t)$

[Pathak et al., 2017]

What can you do with generative models?

1. Data prediction
2. Domain mapping
3. Representation learning
4. Model-based intelligence

Thank you!