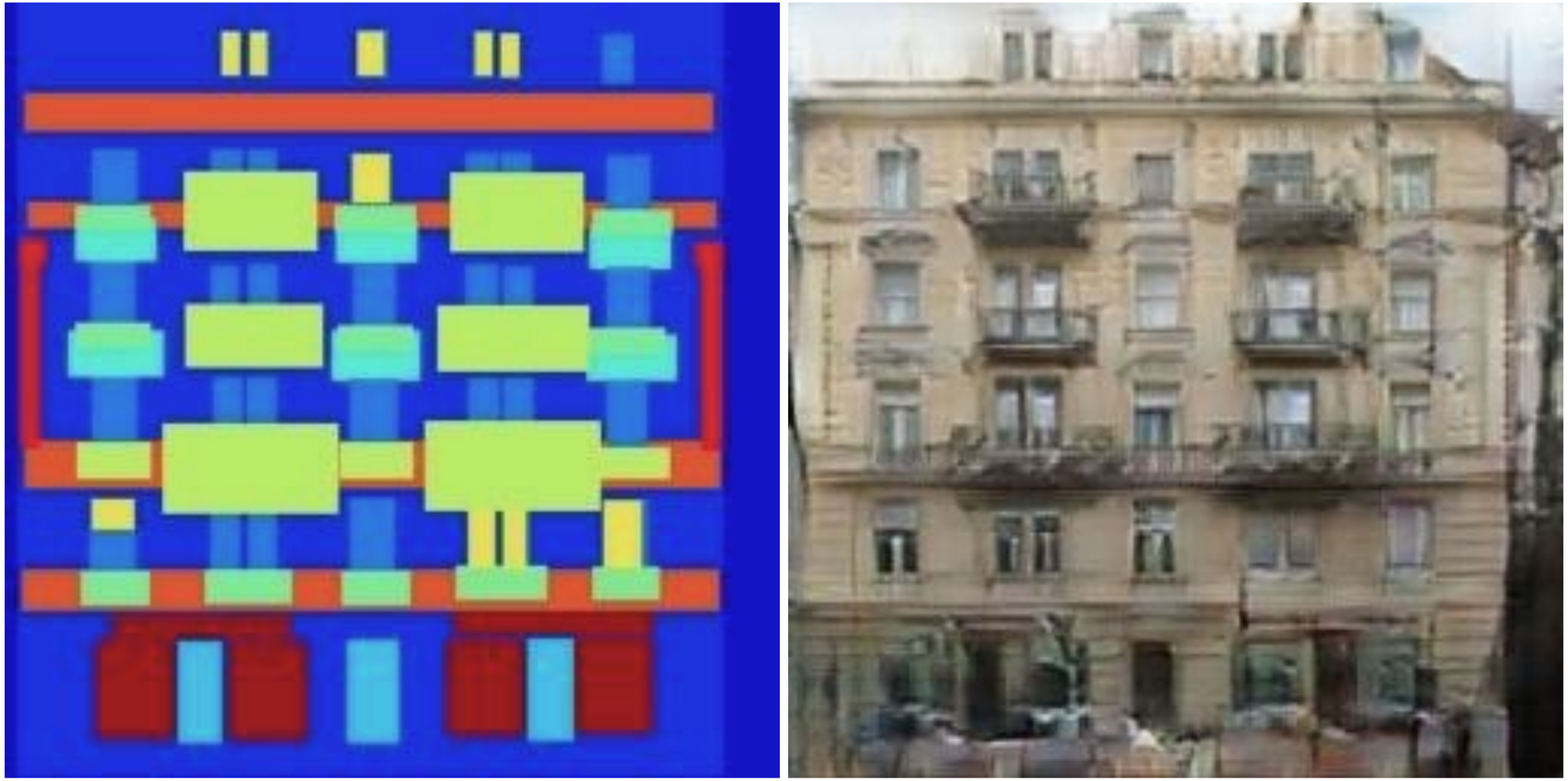
Image-to-Image Translation



CS180: Intro to Comp. Vision and Comp. Photo

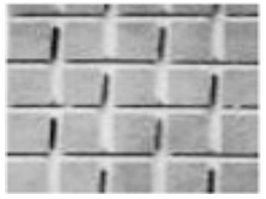
Alexei Efros, UC Berkeley, Fall 2025

Visual Similarity is hard

42 - 24 = 18

SLUMP - TRUMP = edit distance of 2 letters

 - = distance of 50 grey values

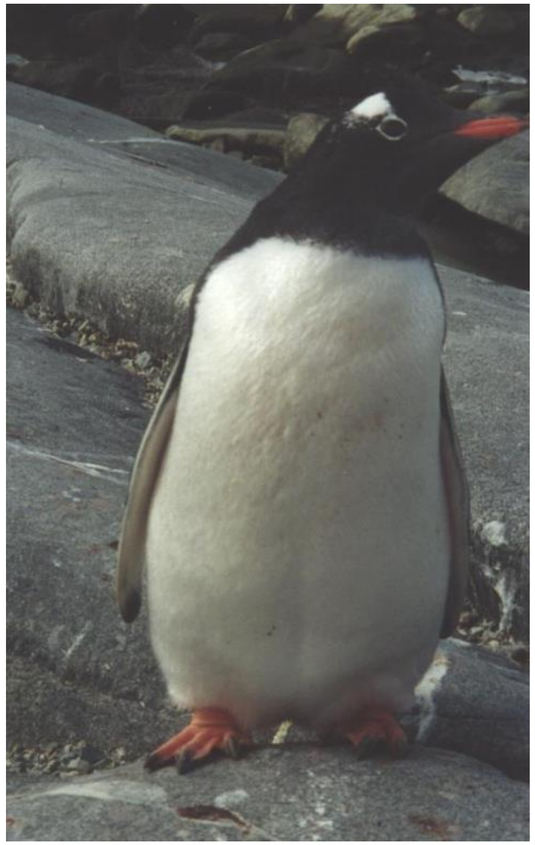
 -  = ?

When are two textures similar?





Visual similarity via labels



Machine Learning as data association

black box classifier

“Penguin”

image *X*

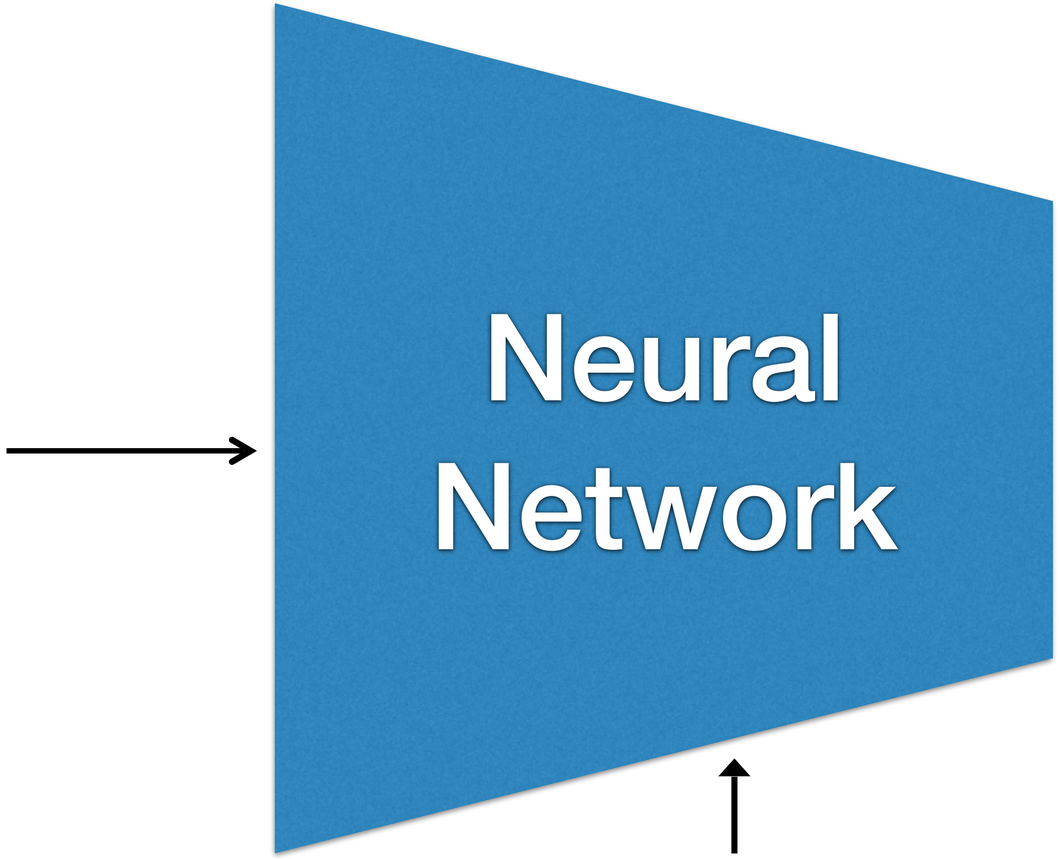
label *Y*

At test time…

image *X*

Deep Learning: a high-capacity classifier

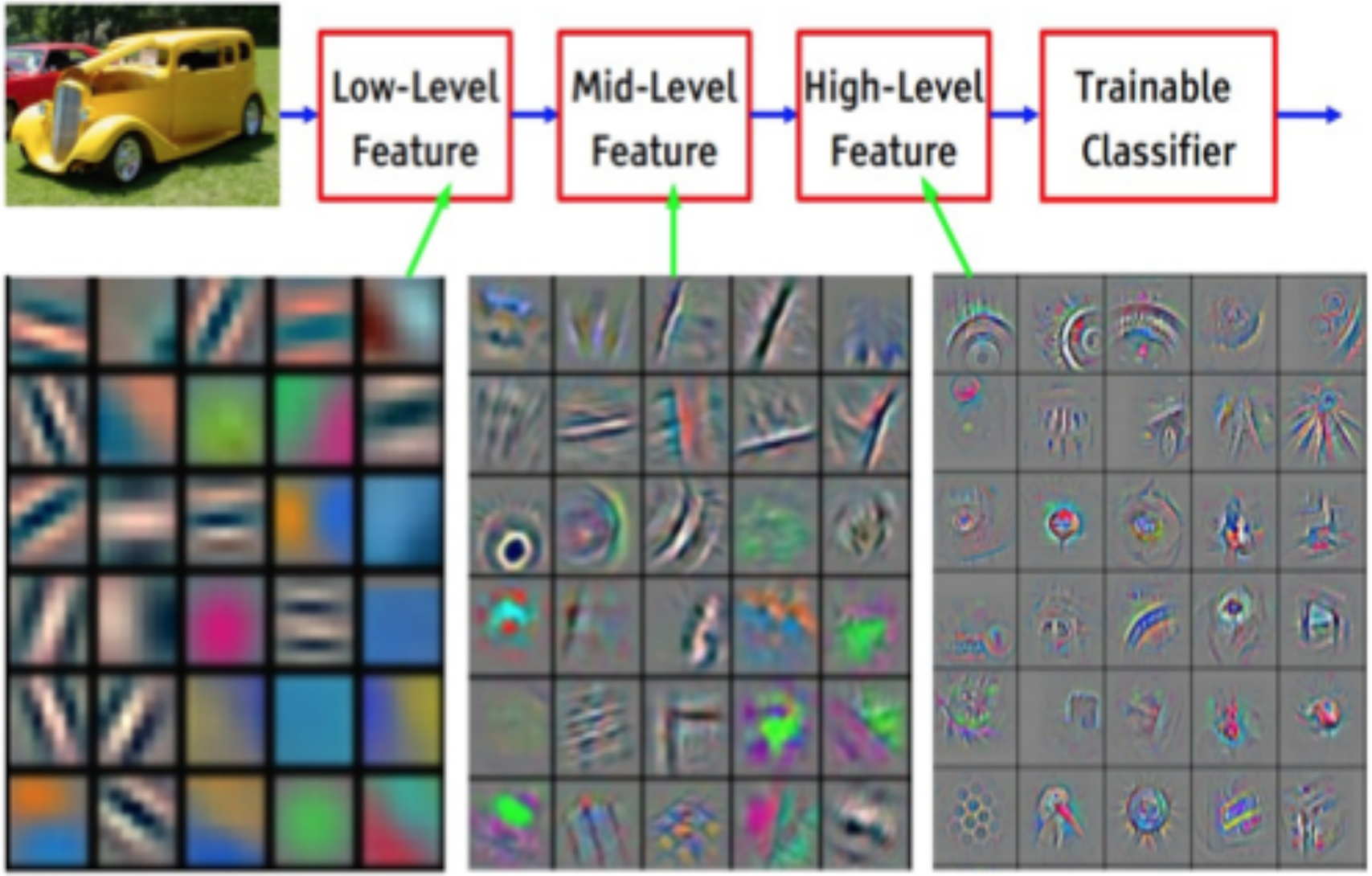
image *X*



Billions of Parameters!

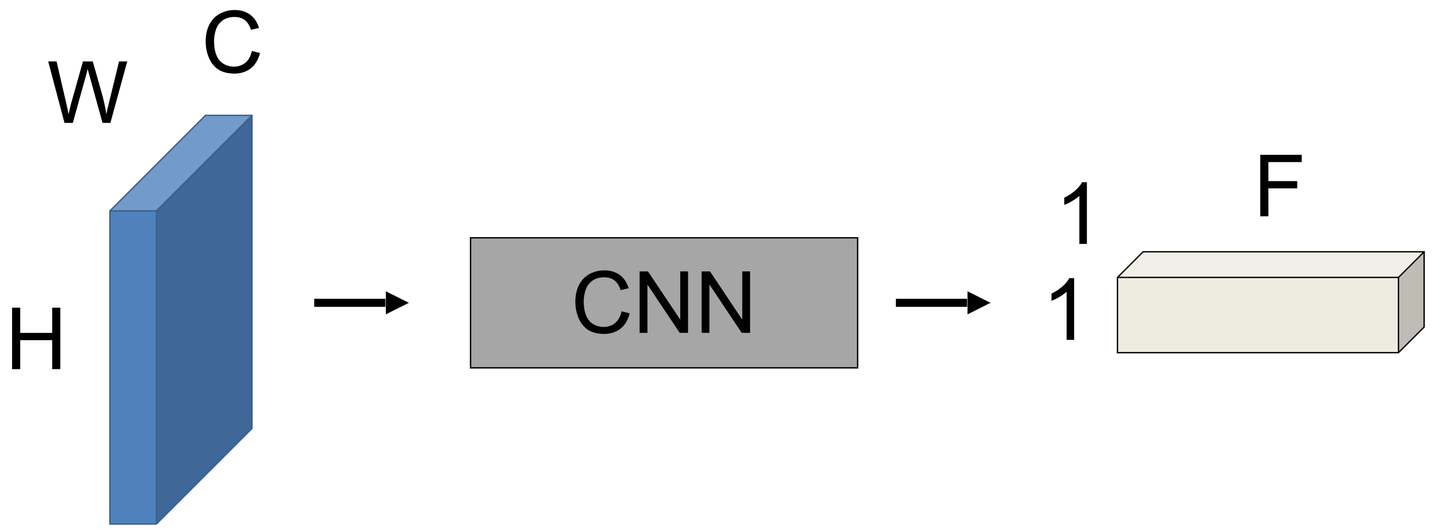
label *Y*

Convolutional Neural Networks



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Recall CS189/CS182

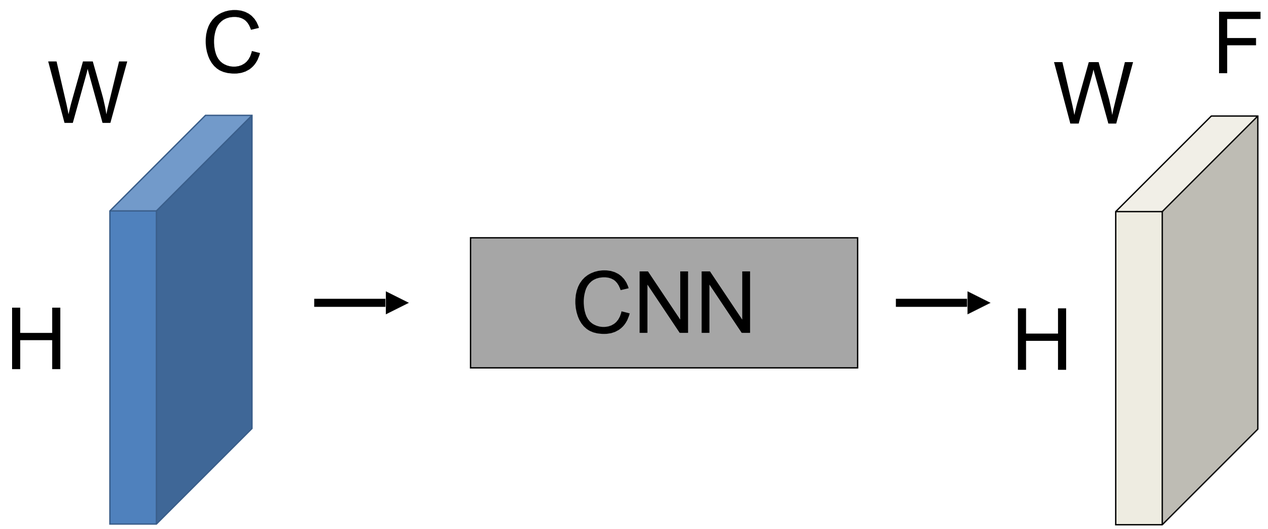


Convert HxW image into a F-dimensional vector

Is this image a cat?
At what distance was this photo taken?
Is this image fake?

Slide by David Fouhey

Pixel Labeling



Convert HxW image into a F-dimensional vector

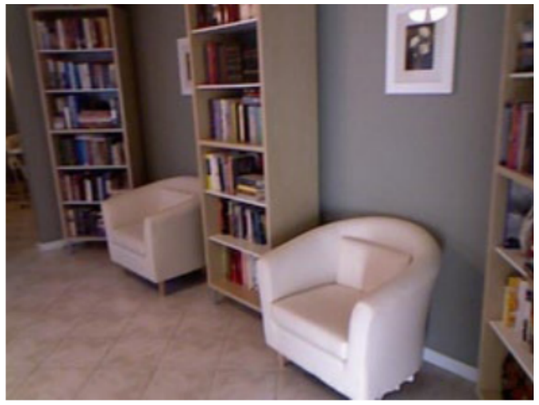
Which pixels in this image are a cat?
How far is each pixel away from the camera?
Which pixels of this image are fake?

Slide by David Fouhey

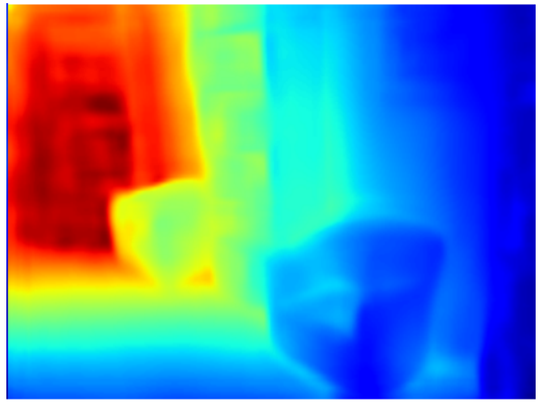
e.g. Depth Prediction

Instead: give label of depthmap, train network to do regression (e.g., $\left\| z_i - \hat{z}_i \right\|$ where z_i is the ground-truth and $\hat{z}_i$ the prediction of the network at pixel i).

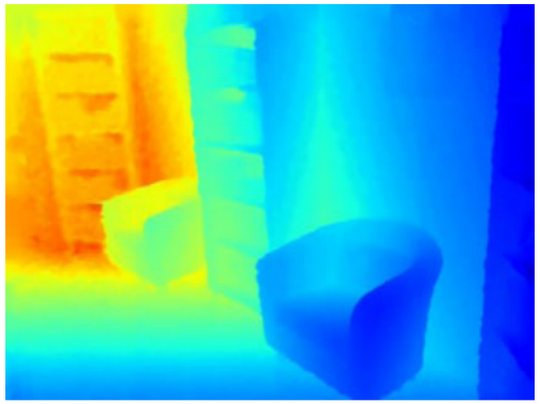
Input HxWx3
RGB Image



Output HxWx1
Depth Image



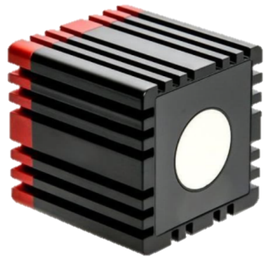
True HxWx1
Depth Image

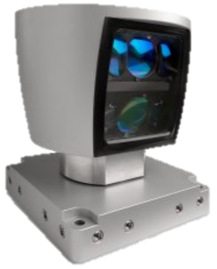


Result credit: Eigen and Fergus, ICCV 2015

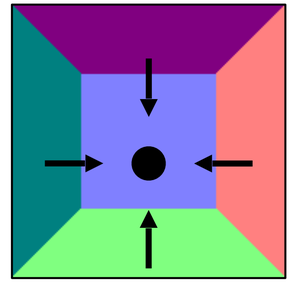
Slide by David Fouhey

Surface Normals

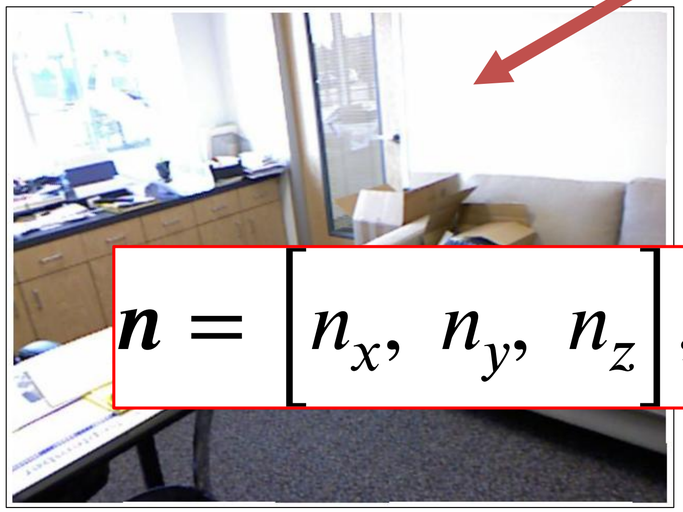




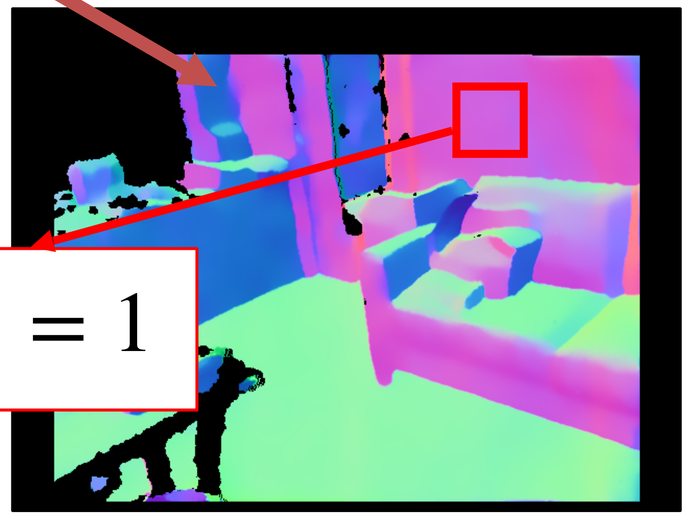
Room



Legend



$$\boldsymbol{n} = \left[n_x,\ n_y,\ n_z \right], \|\boldsymbol{n}\| = 1$$



Color Image

Normals

Image credit: NYU Dataset, Silberman et al. ECCV 2012

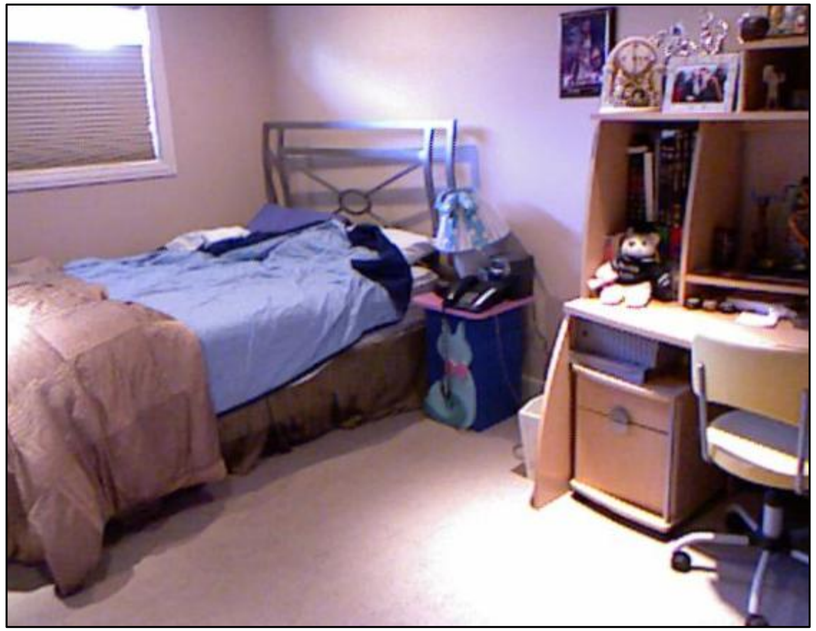
Slide by David Fouhey

Surface Normals

Instead: train normal network to minimize $\left\| \boldsymbol{n}_i - \hat{\boldsymbol{n}}_i \right\|$

where $\boldsymbol{n}_i$ is ground-truth and $\hat{\boldsymbol{n}}_i$ prediction at pixel i.

Input: HxWx3
RGB Image



Output: HxWx3
Normals

Result credit: X. Wang, D. Fouhey, A. Gupta, *Designing Deep Networks for Surface Normal Estimation.* CVPR 2014

Slide by David Fouhey

Image Denoising

Denoising neural network

Slide source: Steve Seitz

“Semantic Segmentation”

Each pixel has label, inc. **background**, and unknown
Usually visualized by colors.
Note: don’t distinguish between object *instances*

Input Label Input Label

Image Credit: Everingham et al. Pascal VOC 2012.

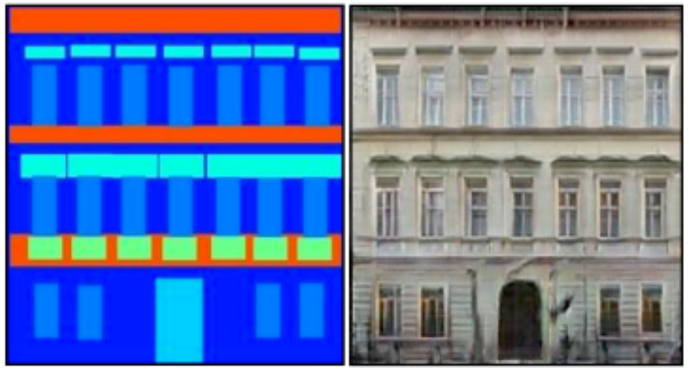
Slide by David Fouhey

Generic: Image-to-Image Translation

Labels to Street Scene

input output

Labels to Facade



input output

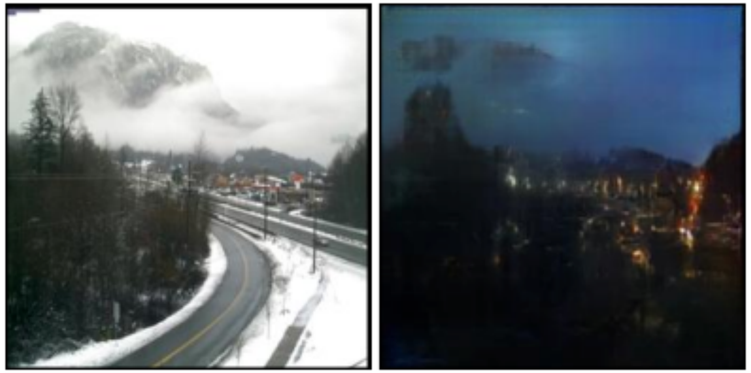
BW to Color

input output

Aerial to Map

input output

Day to Night



input output

Edges to Photo

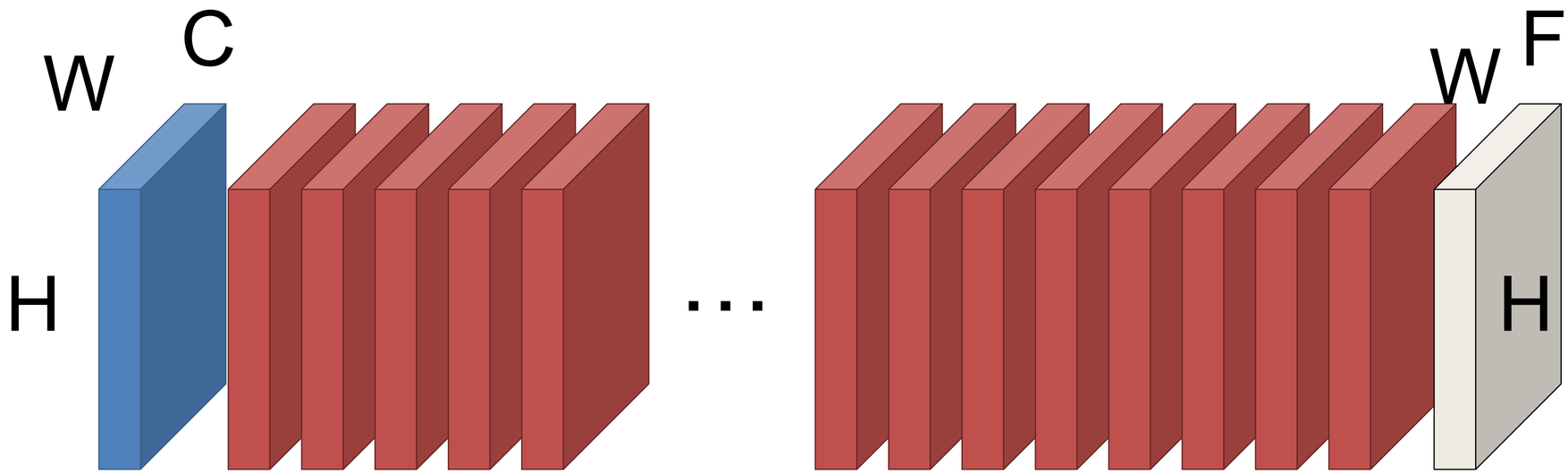
input output

First – Two “Wrong” Ways

- It’s helpful to see two “wrong” ways to do this.

Slide by David Fouhey

Why Not Stack Convolutions?



n 3x3 convs have a receptive field of 2n+1 pixels

How many convolutions until >=200 pixels?

100

Slide by David Fouhey

Idea #2

Crop out every sub-window and predict the label in the middle.

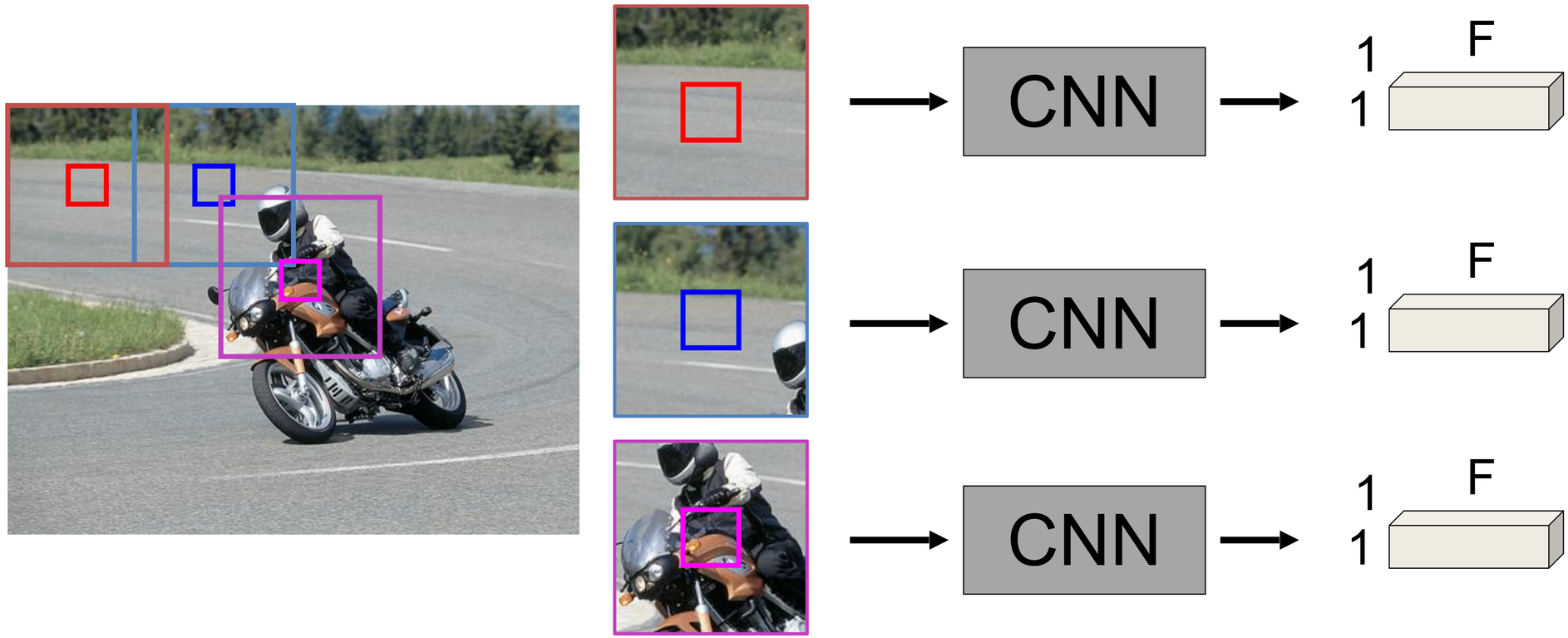


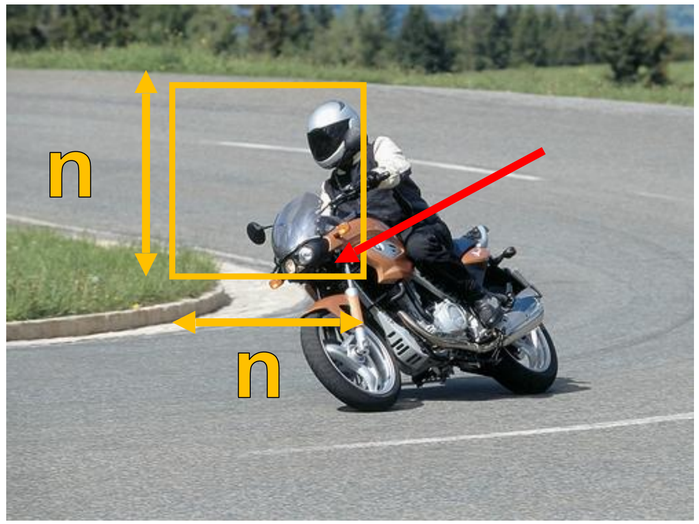
Image credit: PASCAL VOC, Everingham et al.

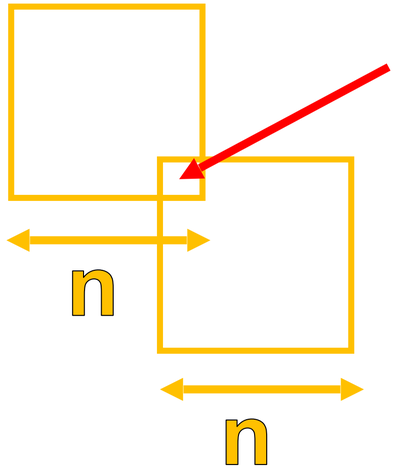
Slide by David Fouhey

Idea #2

Meet “Gabor”. We extract NxN patches and do independent CNNs. **How many times does Gabor filter the red pixel?**

Gabor





Answer:
(2n-1)*(2n-1)

Image credit: PASCAL VOC, Everingham et al.

Slide by David Fouhey

The Big Issue

We need to:

1. Have large receptive fields to figure out what we're looking at
2. Not waste a ton of time or memory while doing so

These two objectives are in total conflict

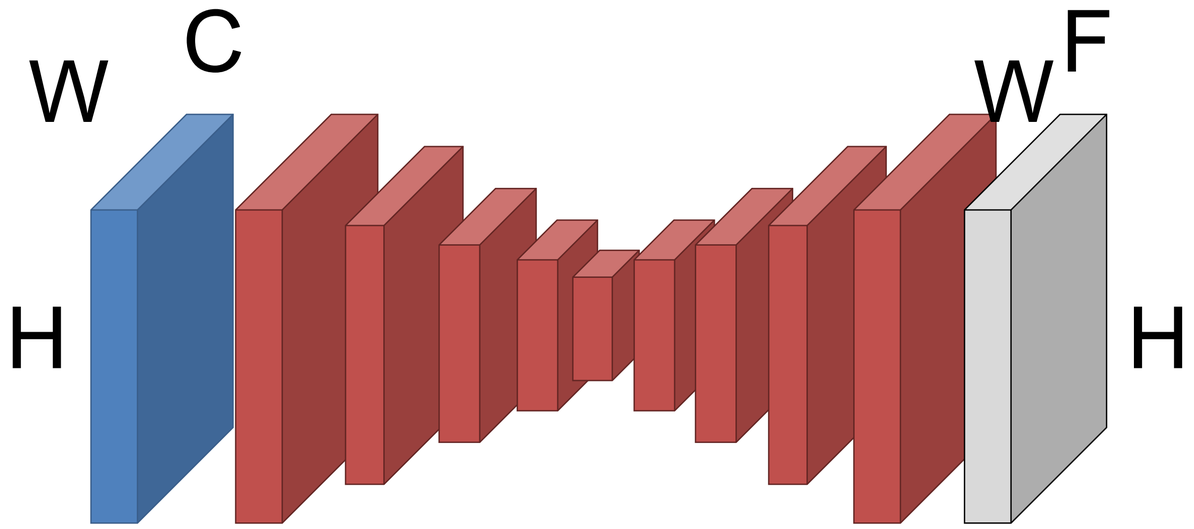
Slide by David Fouhey

Encoder-Decoder

Key idea: First **downsample** towards middle of network. Then **upsample** from middle.

How do we downsample?

Convolutions, pooling



Slide by David Fouhey

Putting it Together

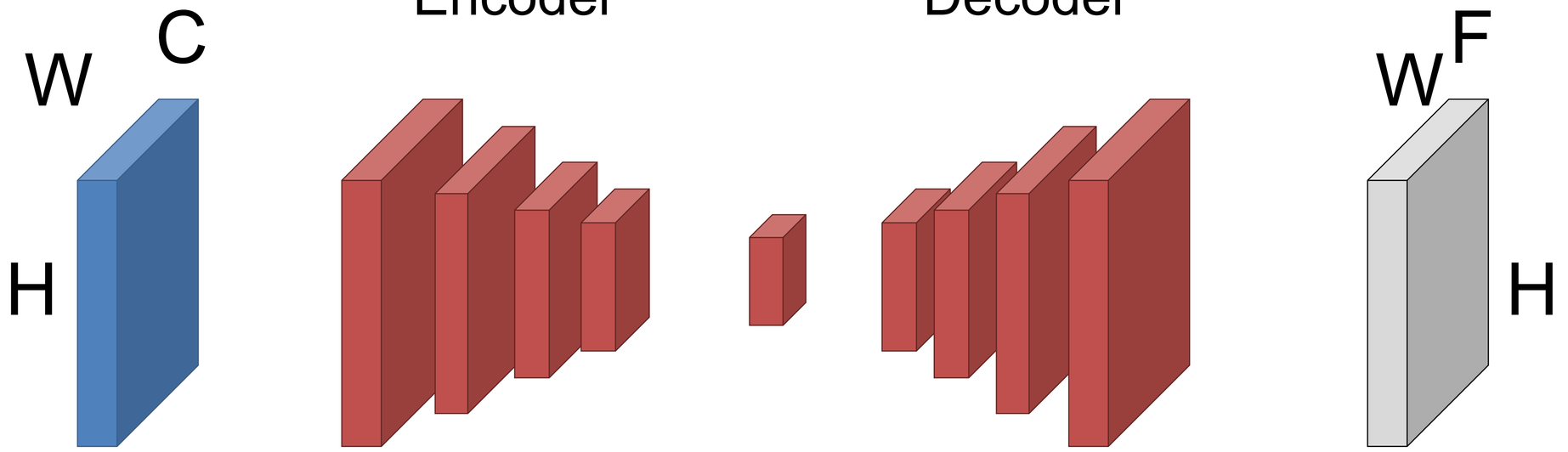
Convolutions + pooling downsample/compress/encode
Transpose convs./unpoolings upsample/uncompress/decode

Input

Downsample
Conv, pool
“Encoder”

Upsample
Tr. Conv./Unpool
“Decoder”

Output

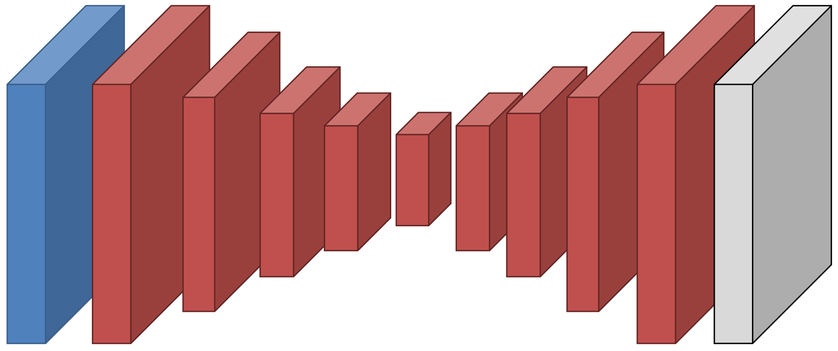


Slide by David Fouhey

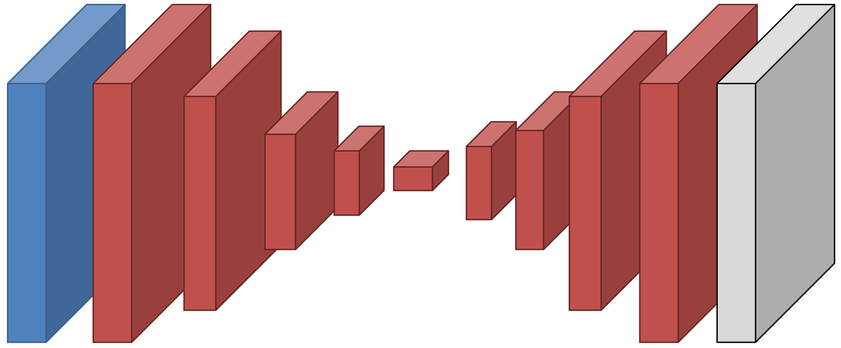
Putting It Together – Block Sizes

- Networks come in lots of forms
- **Don’t take any block sizes literally.**
- Often (not always) keep some spatial resolution

Encode to spatially smaller tensor, then decode.



Encode to 1D vector then decode

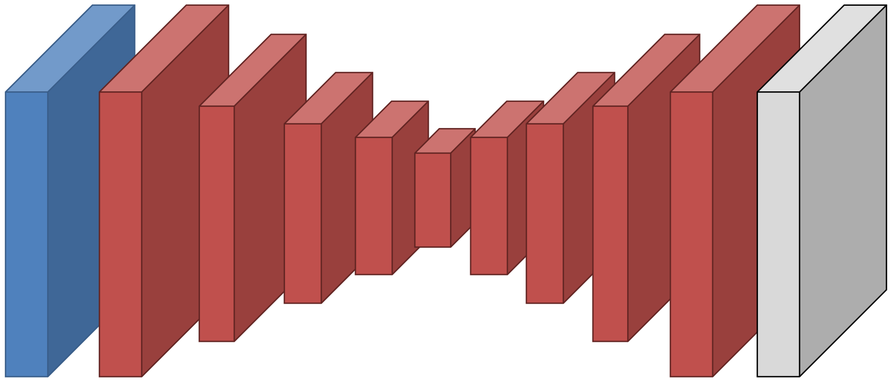


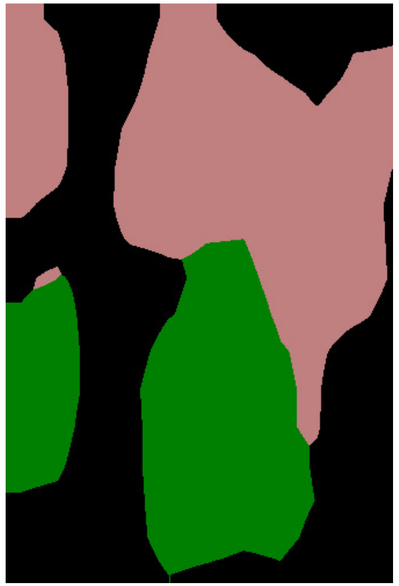
Slide by David Fouhey

Missing Details

While the output *is* HxW, just upsampling often produces results without details/not aligned with the image.
Why?

Information about details lost when downsampling!



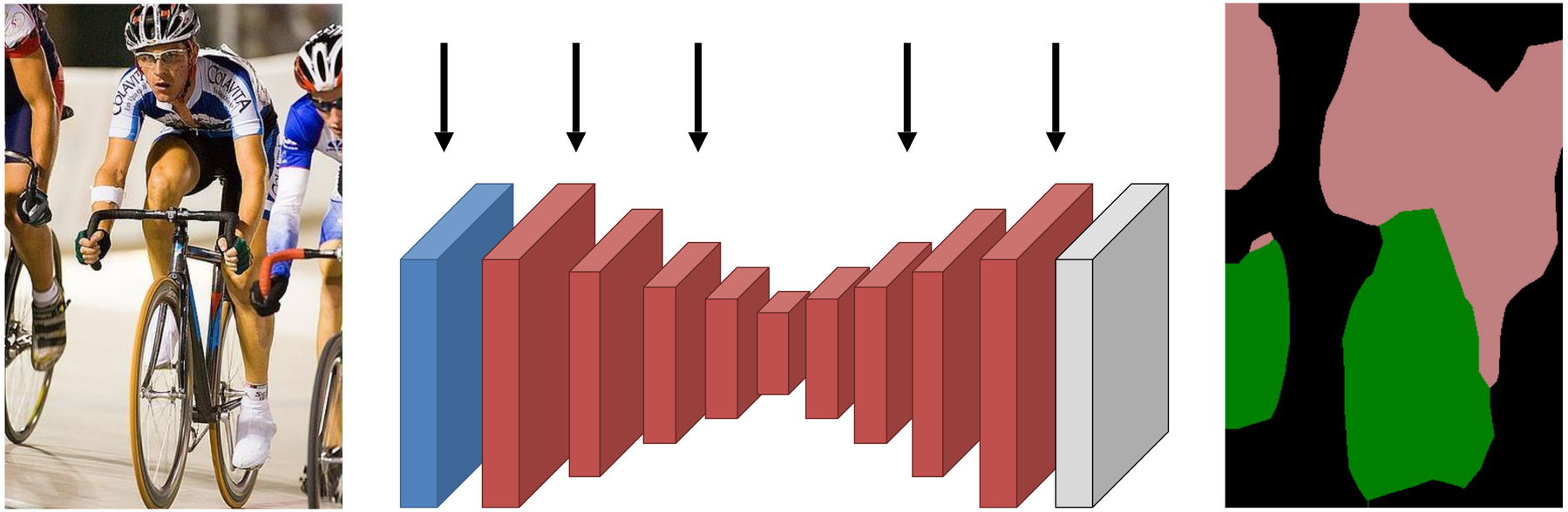


Result from Long et al. *Fully Convolutional Networks For Semantic Segmentation*. CVPR 2014

Slide by David Fouhey

Missing Details

Where is the useful information about the high-frequency details of the image?



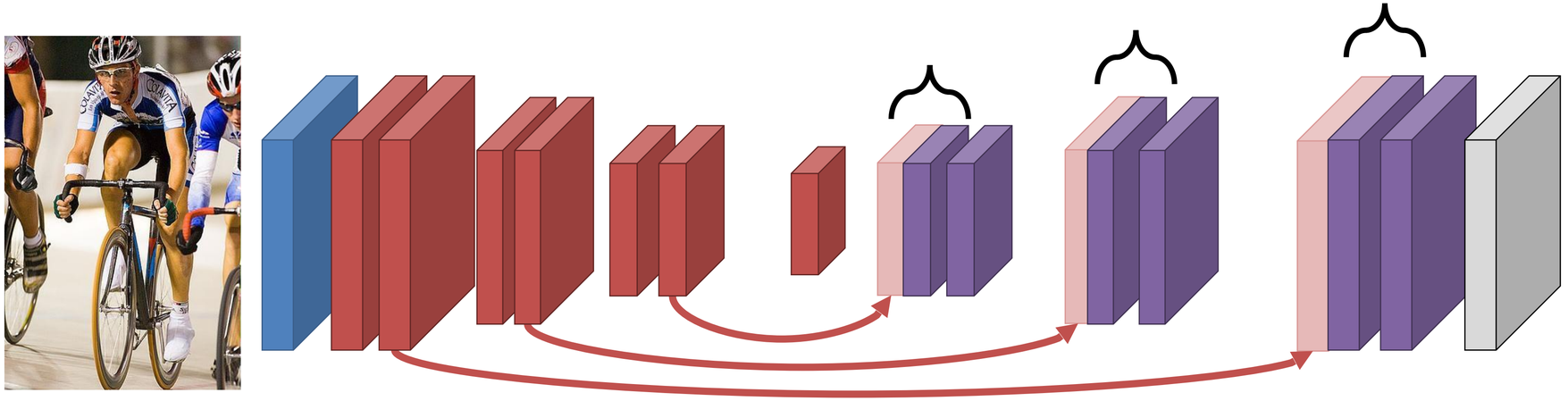
Result from Long et al. *Fully Convolutional Networks For Semantic Segmentation*. CVPR 2014

Slide by David Fouhey

Missing Details

How do you send details forward in the network?

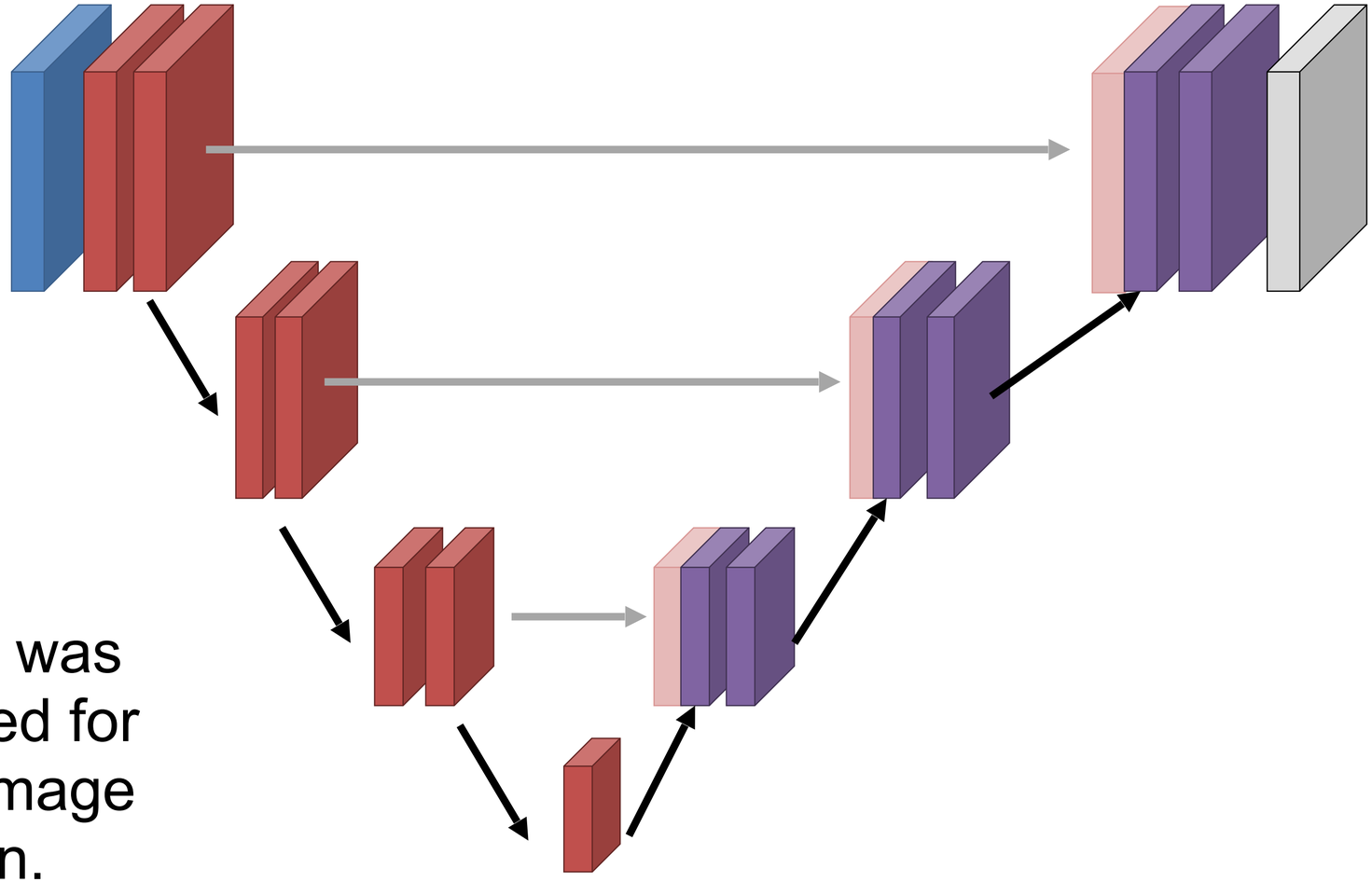
You copy the activations forward.
Subsequent layers at the same resolution figure out how to fuse things.



Copy

Result from Long et al. *Fully Convolutional Networks For Semantic Segmentation*. CVPR 2014

U-Net

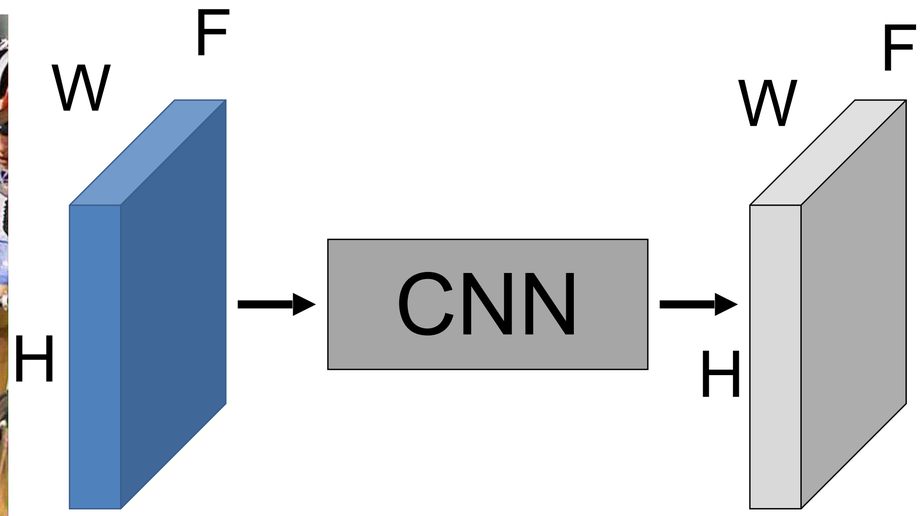


Extremely popular architecture, was originally used for biomedical image segmentation.

Ronneberger et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation. MICCAI 2015

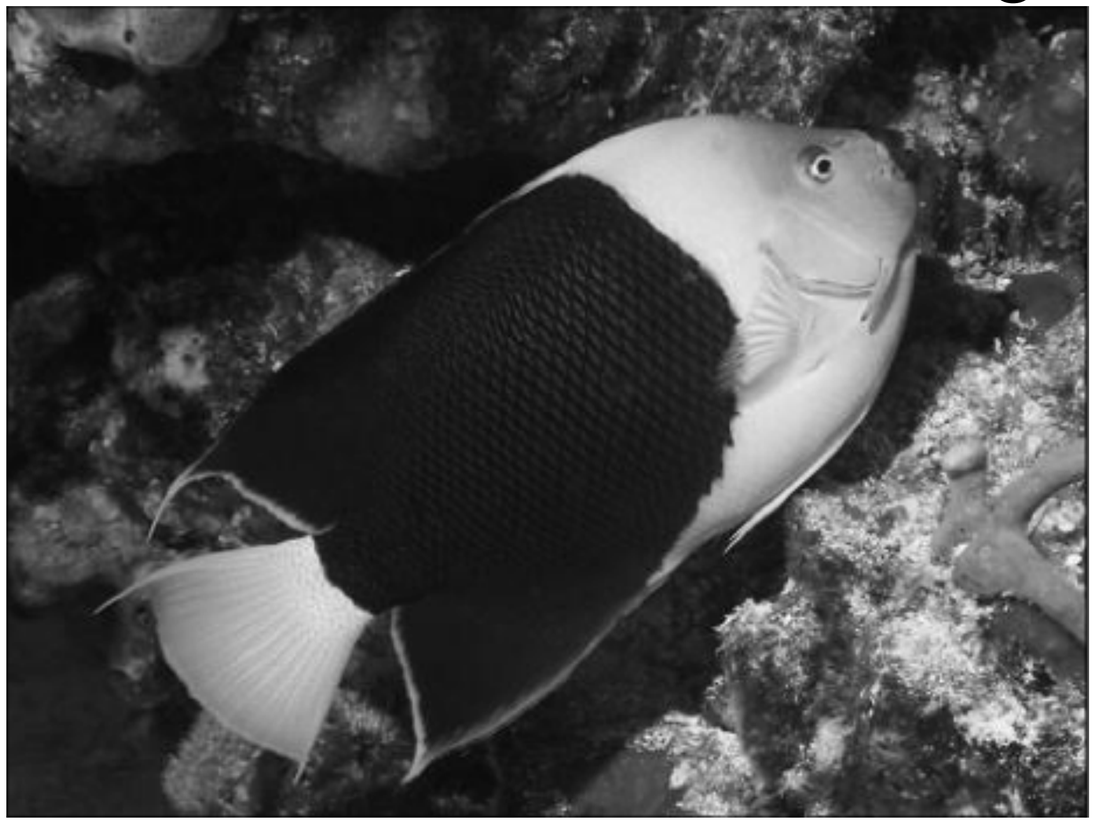
U-Net improves performance

Input Image



Predicted Classes

Image Colorization



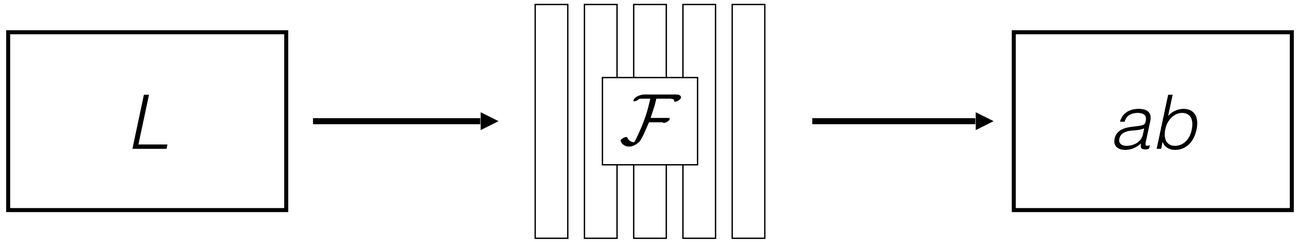
$\mathcal{F}$

Grayscale image: *L* channel

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

Color information: *ab* channels

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



Zhang, Isola, Efros. *Colorful Image Colorization.* In *ECCV*, 2016.

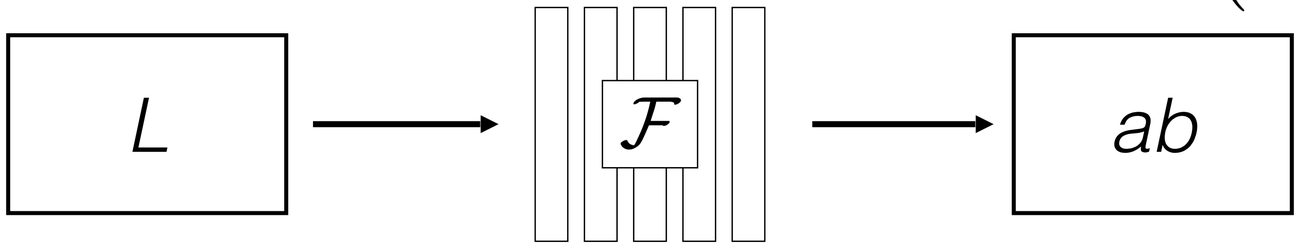
$\mathcal{F}$

Grayscale image: *L* channel

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

Concatenate (*L*,*ab*) channels

$$(\mathbf{X}, \hat{\mathbf{Y}})$$



Zhang, Isola, Efros. *Colorful Image Colorization.* In *ECCV*, 2016.

Regressing to pixel values doesn't work ☹

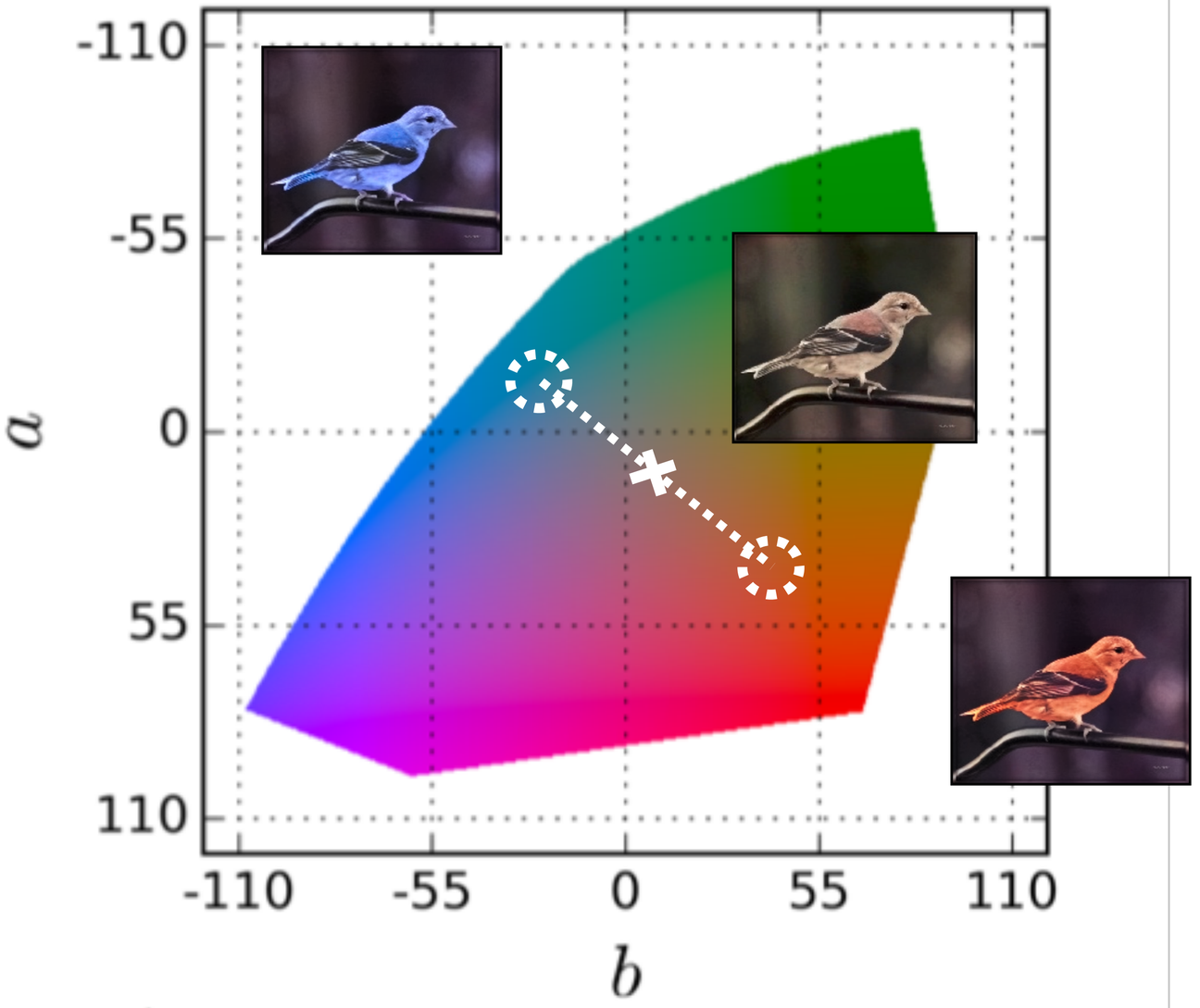
Input

Output

Ground truth

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

Slide by Richard Zhang



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

Slide by Richard Zhang

Better Loss Function

$$\theta^* = \arg\min_\theta \ell(\mathcal{F}_\theta(\mathbf{X}), \mathbf{Y})$$

- Regression with L2 loss inadequate

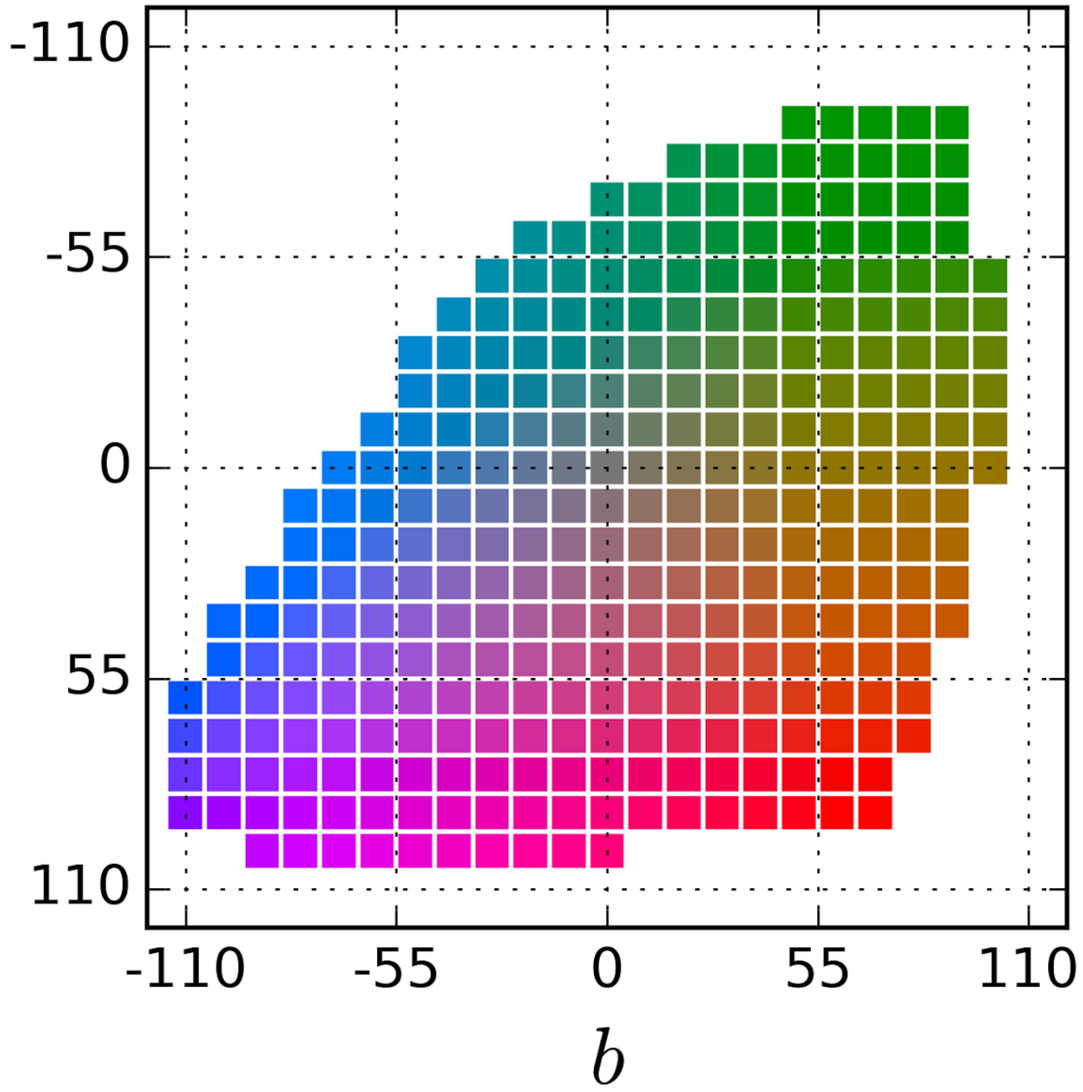
$$\mathrm{L}_2(\widehat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2}\sum_{h,w} \|\mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w}\|_2^2$$

- Use per-pixel multinomial classification

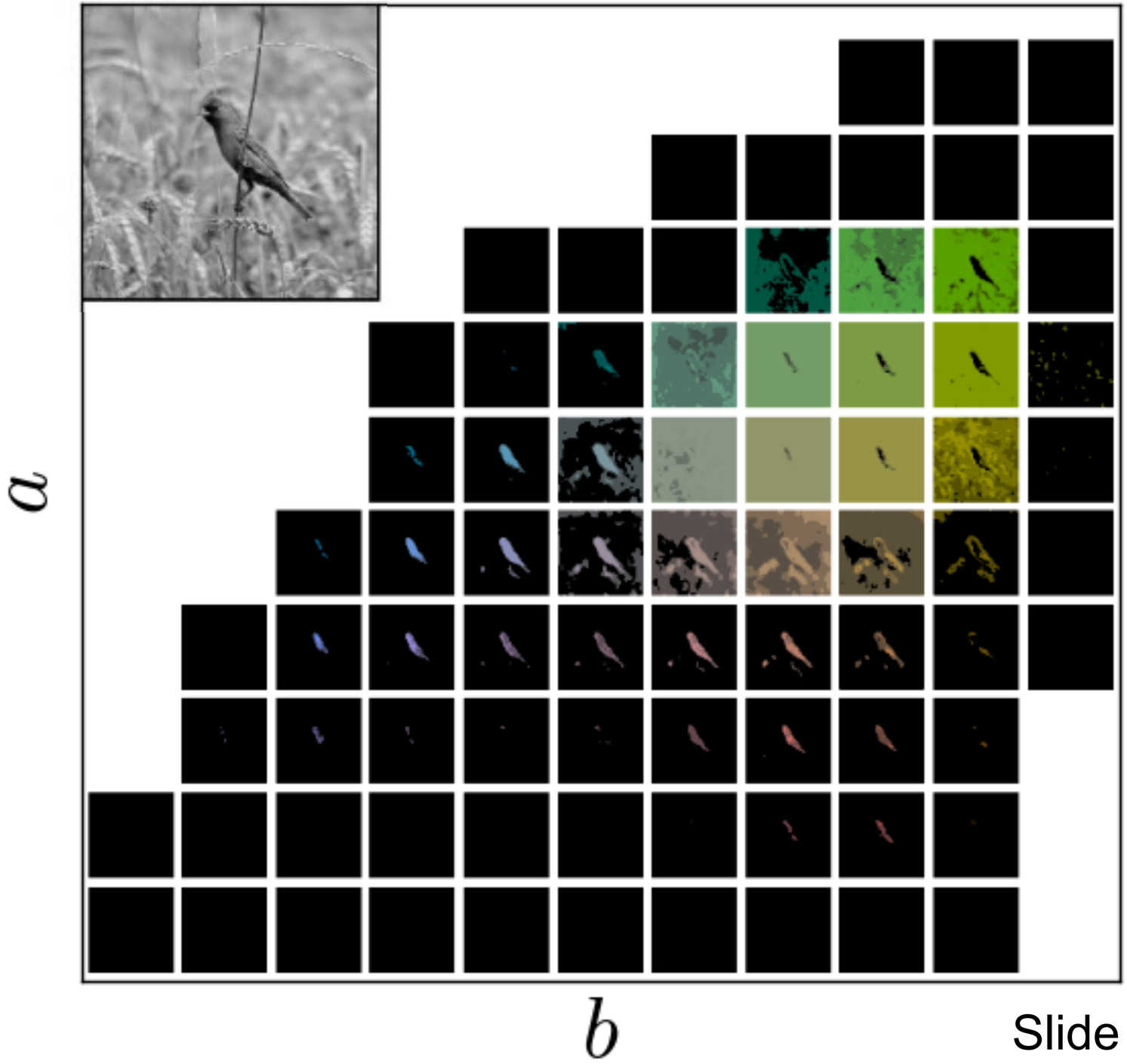
$$\mathrm{L}(\widehat{\mathbf{Z}}, \mathbf{Z}) = -\frac{1}{HW}\sum_{h,w}\sum_{q} \mathbf{Z}_{h,w,q} \log(\widehat{\mathbf{Z}}_{h,w,q})$$

Colors in *ab* space

(discrete)



Slide by Richard Zhang



Slide by Richard Zhang

Designing pixel loss functions

Input

Zhang et al. 2016

Ground truth

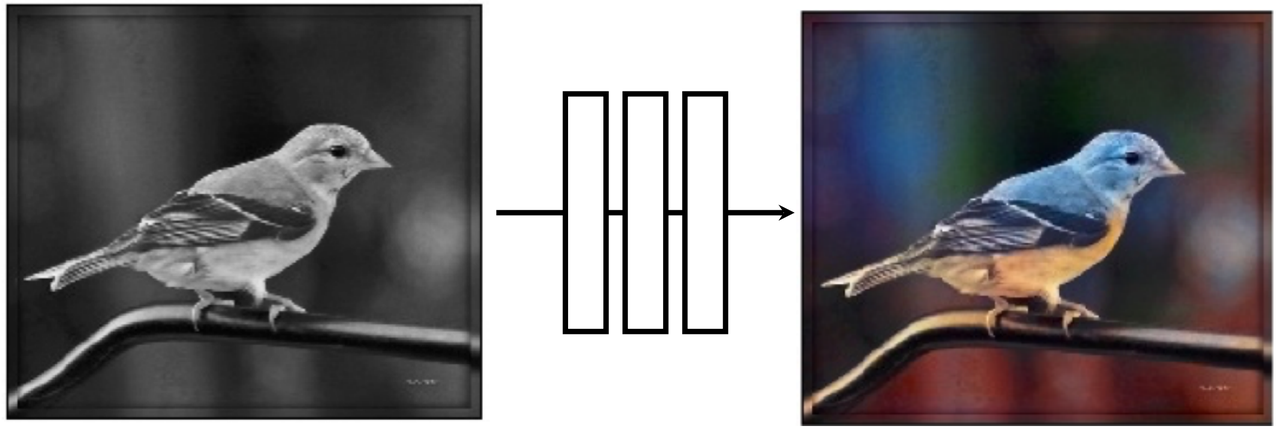
Color distribution cross-entropy loss with colorfulness enhancing term.

[Zhang, Isola, Efros, ECCV 2016]


DADS
DADS

Designing pixel loss functions

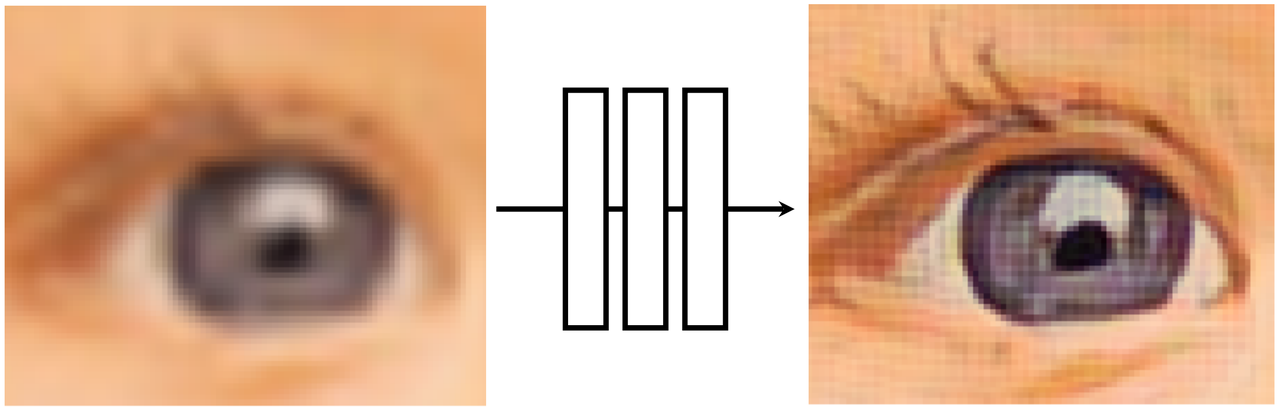
Image colorization



[Zhang et al. 2016]

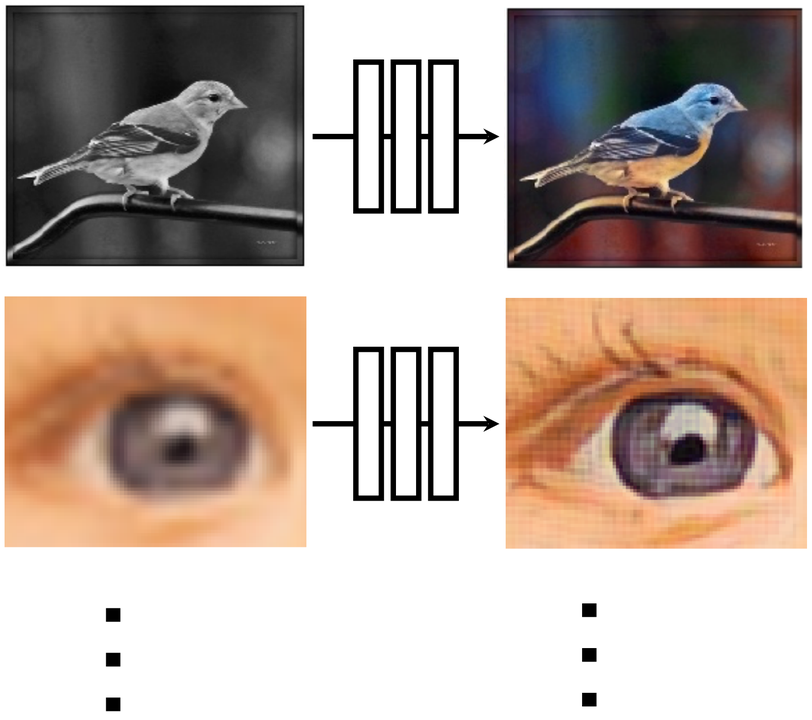
Cross entropy loss, with colorfulness term

Super-resolution



[Johnson et al. 2016]

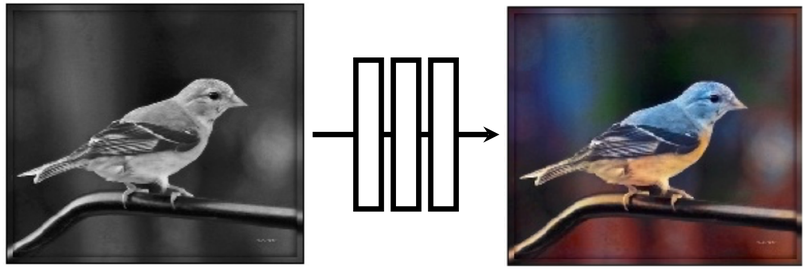
“semantic feature loss” (VGG feature covariance matching objective)

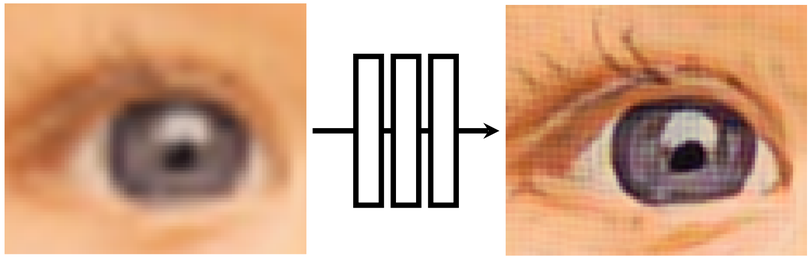


Universal loss?

Generative Adversarial Network (GANs)

Generated images



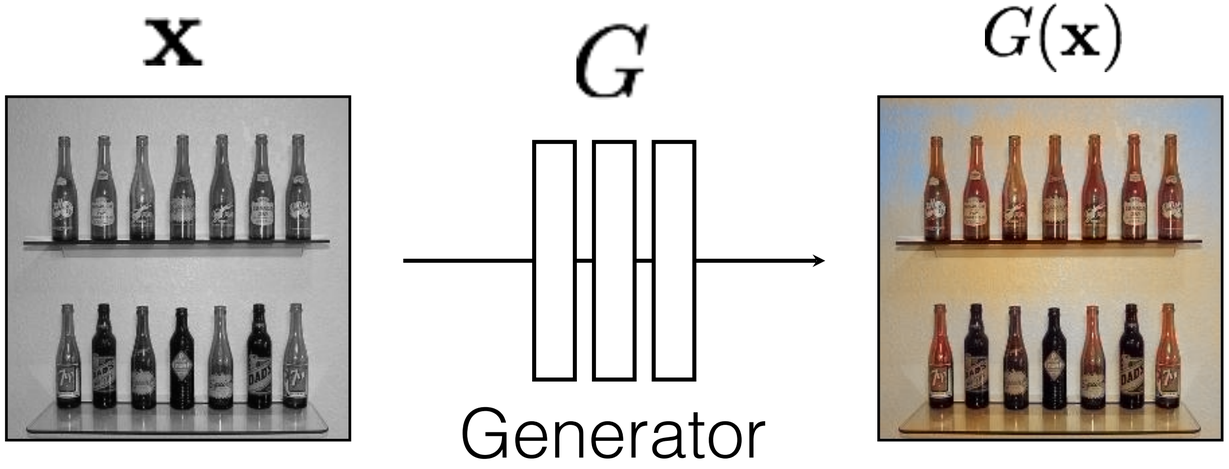


Generated vs Real (classifier)

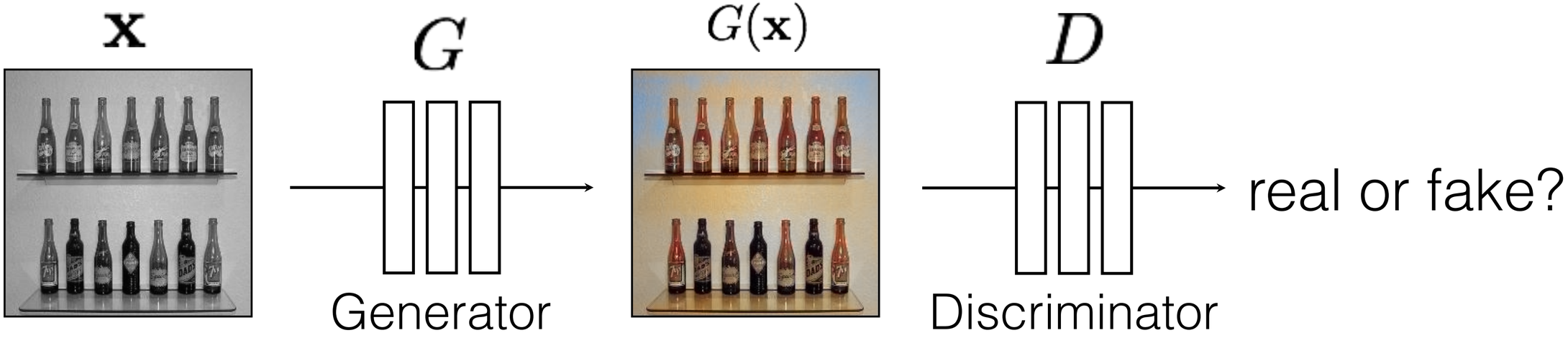
Real photos

...

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



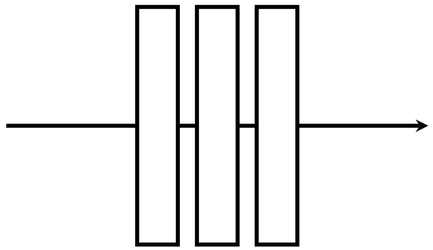
[Goodfellow et al., 2014]

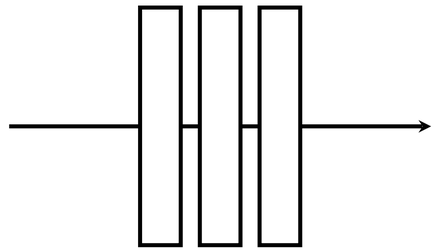


G tries to synthesize fake images that fool **D**

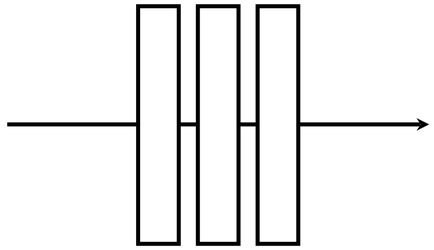
D tries to identify the fakes

[Goodfellow et al., 2014]





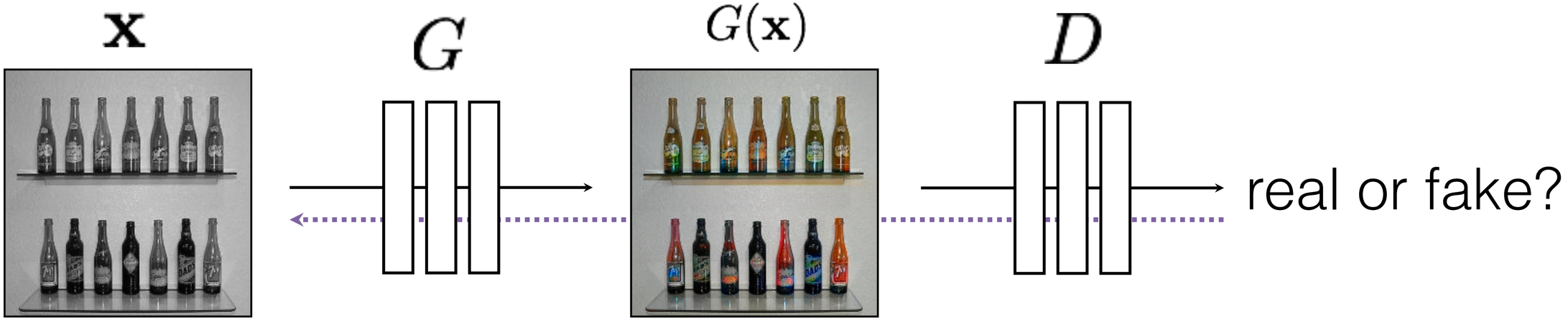
fake (0.9)



real (0.1)

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

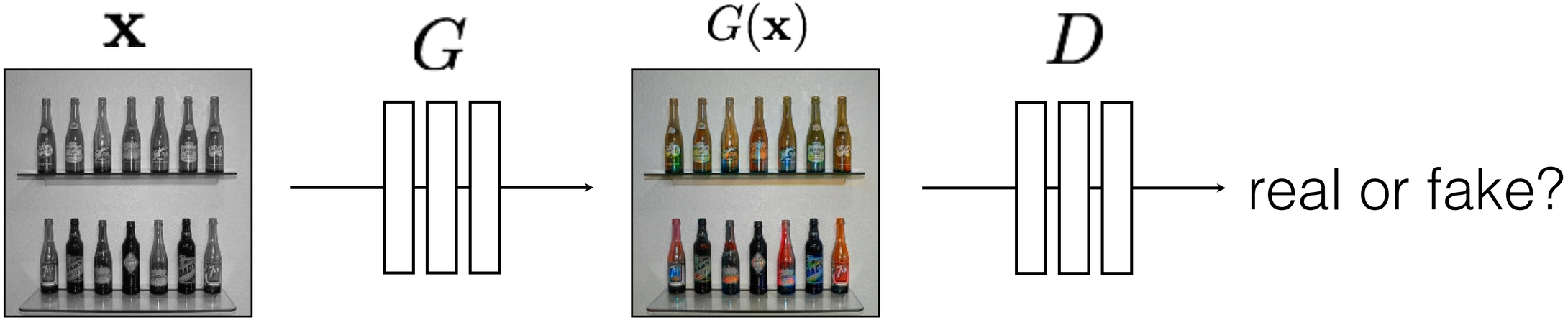
[Goodfellow et al., 2014]



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

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

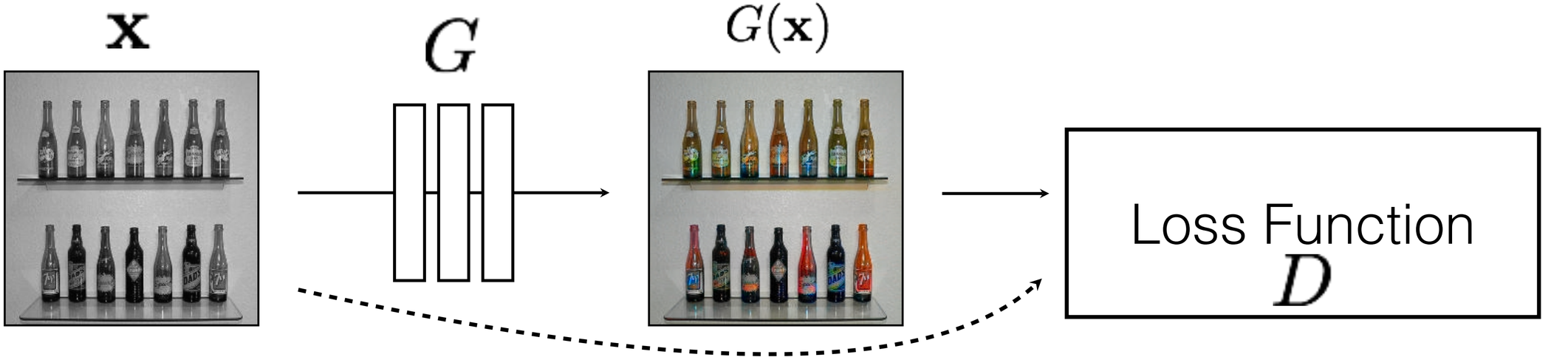
[Goodfellow et al., 2014]



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

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

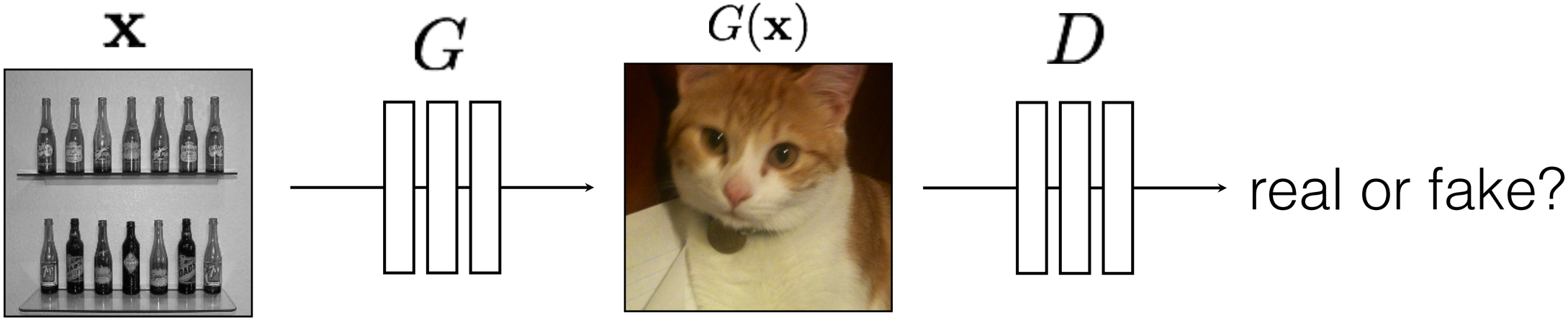
[Goodfellow et al., 2014]



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

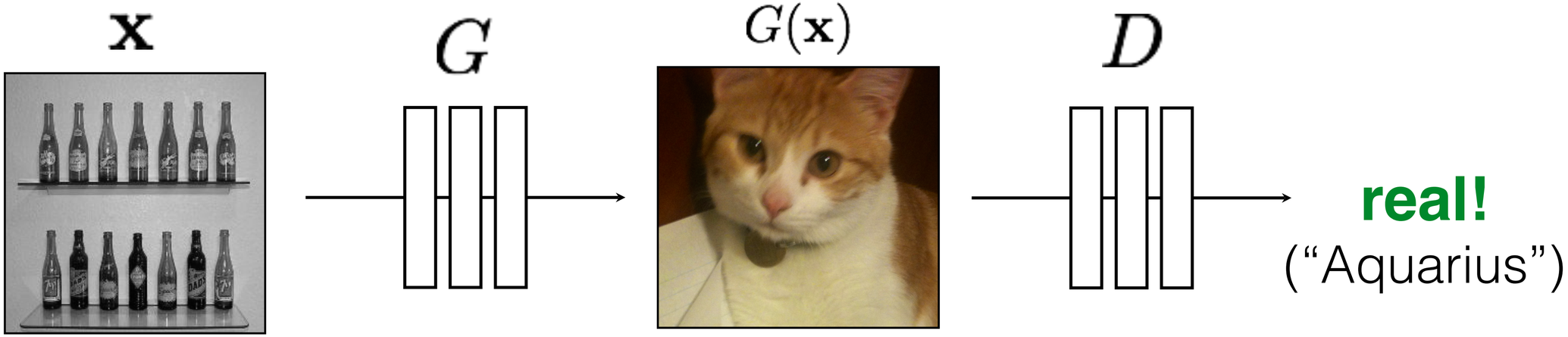
Rather than being hand-designed, it is *learned*.

[Goodfellow et al., 2014]
[Isola et al., 2017]



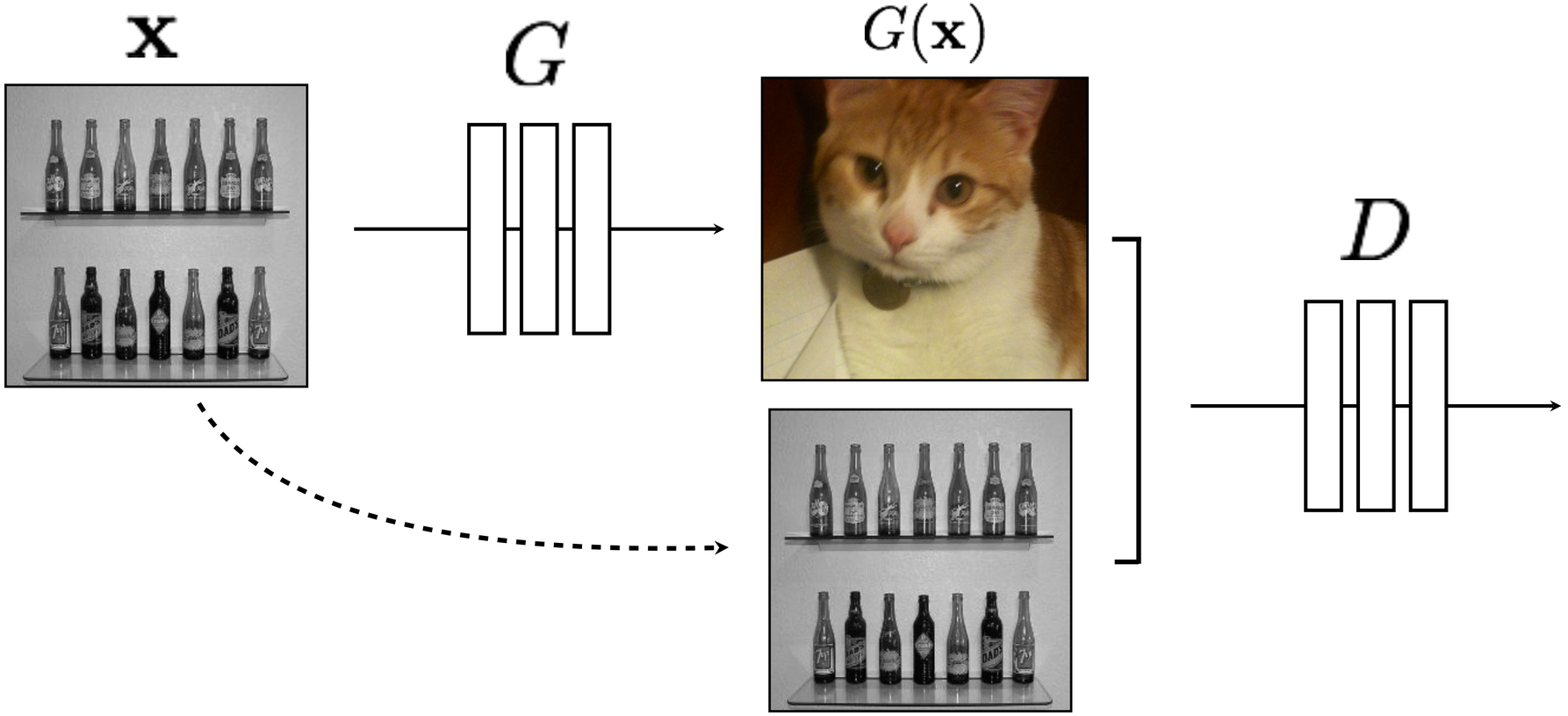
$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(G(\mathbf{x})) \quad + \quad \log(1 - D(\mathbf{y})) \;\;]$$

[Goodfellow et al., 2014]



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

[Goodfellow et al., 2014]



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

[Goodfellow et al., 2014]

[Isola et al., 2017]

$\mathbf{x}$

G

$G(\mathbf{x})$

D

real or fake *pair* ?

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

[Goodfellow et al., 2014]

[Isola et al., 2017]

$\mathbf{x}$

G

$G(\mathbf{x})$

D

fake *pair*

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

[Goodfellow et al., 2014]

[Isola et al., 2017]

$\mathbf{x}$

G

$G(\mathbf{x})$

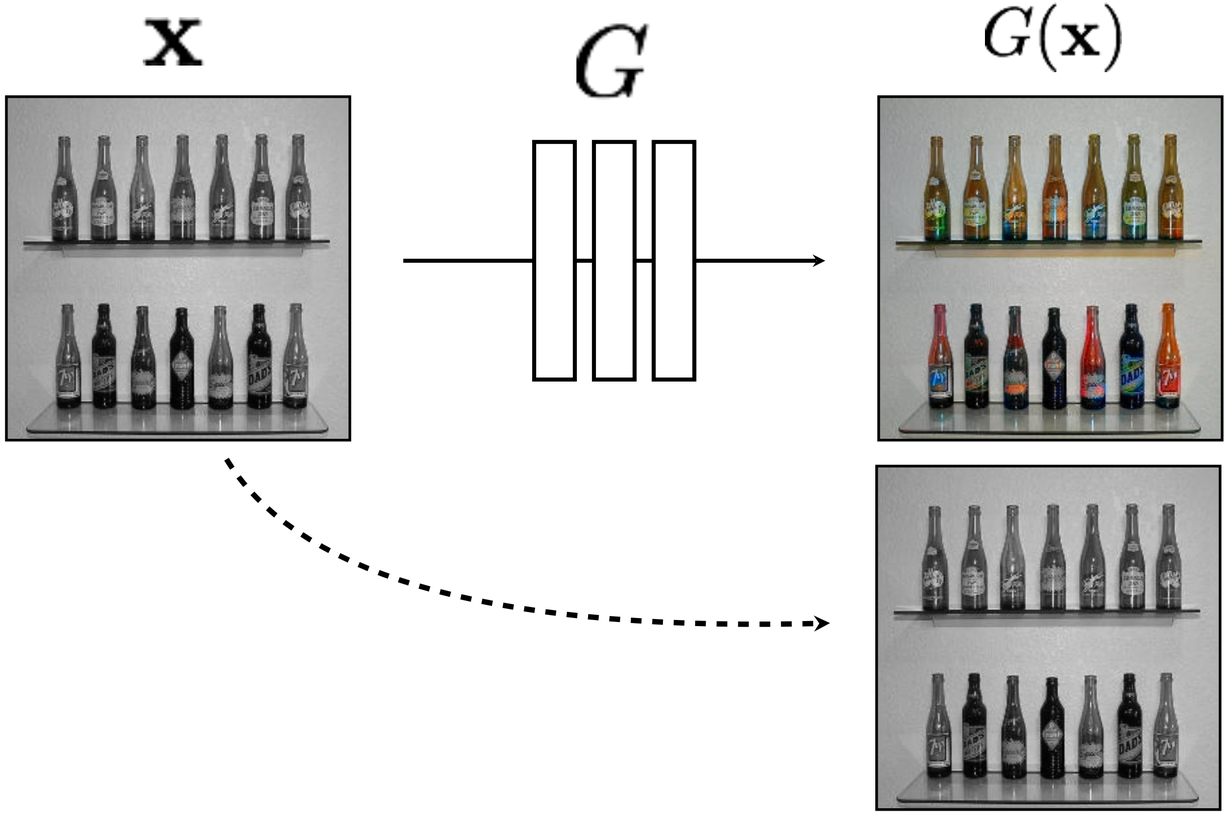
D

real *pair*

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

[Goodfellow et al., 2014]

[Isola et al., 2017]



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

[Goodfellow et al., 2014]

[Isola et al., 2017]

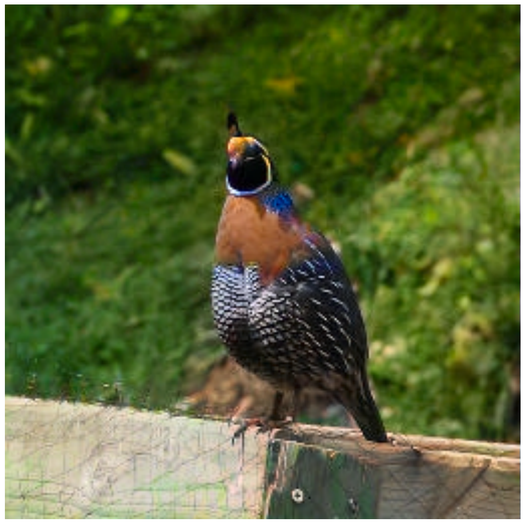
BW → Color



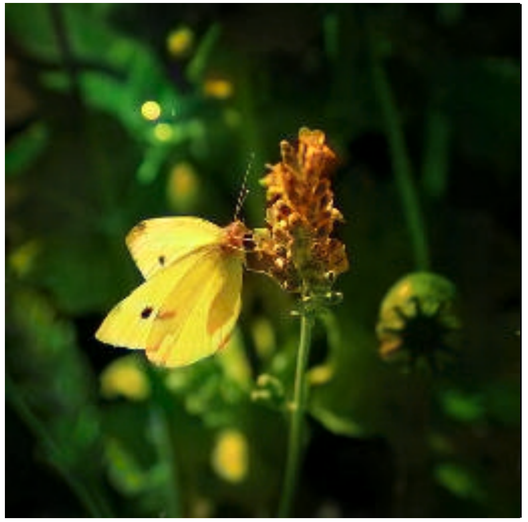



Data from [Russakovsky et al. 2015]

BW → Color

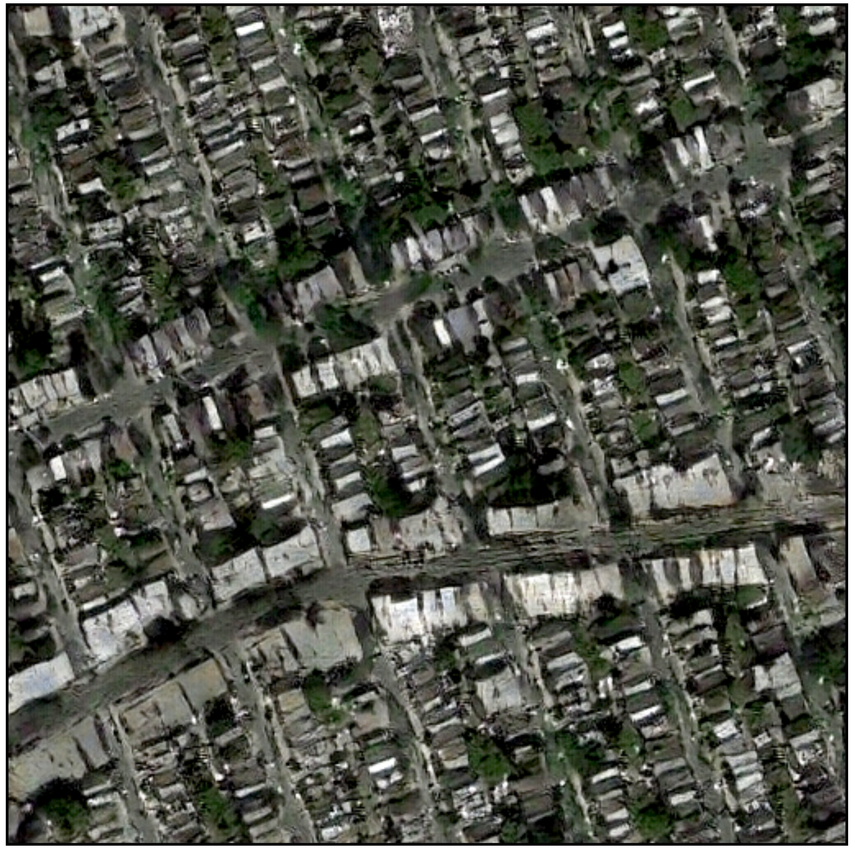




Data from [Russakovsky et al. 2015]

Input

Output



Groundtruth

Data from
[maps.google.com]

Input

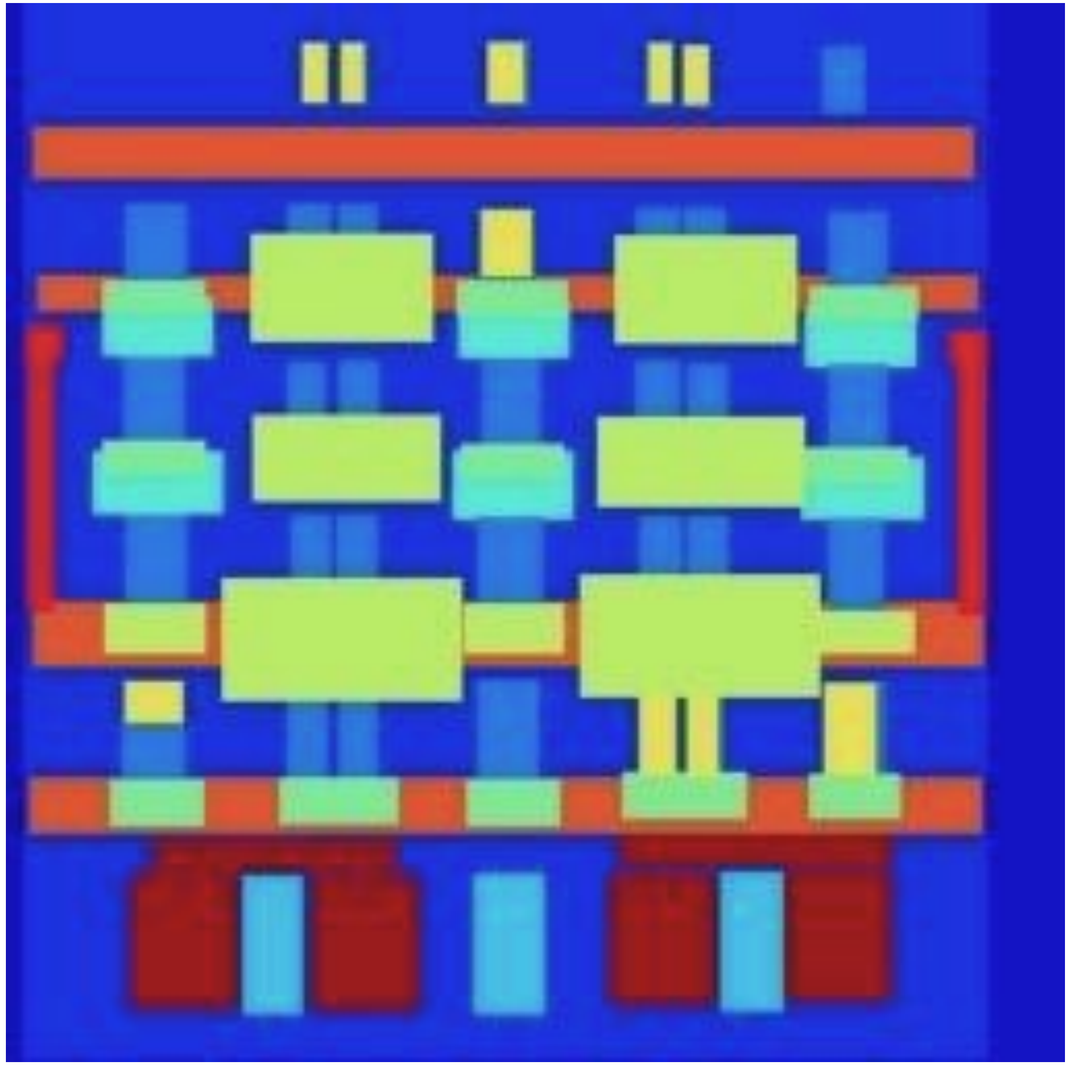
Output

Groundtruth

Data from [maps.google

Labels → Facades

Input

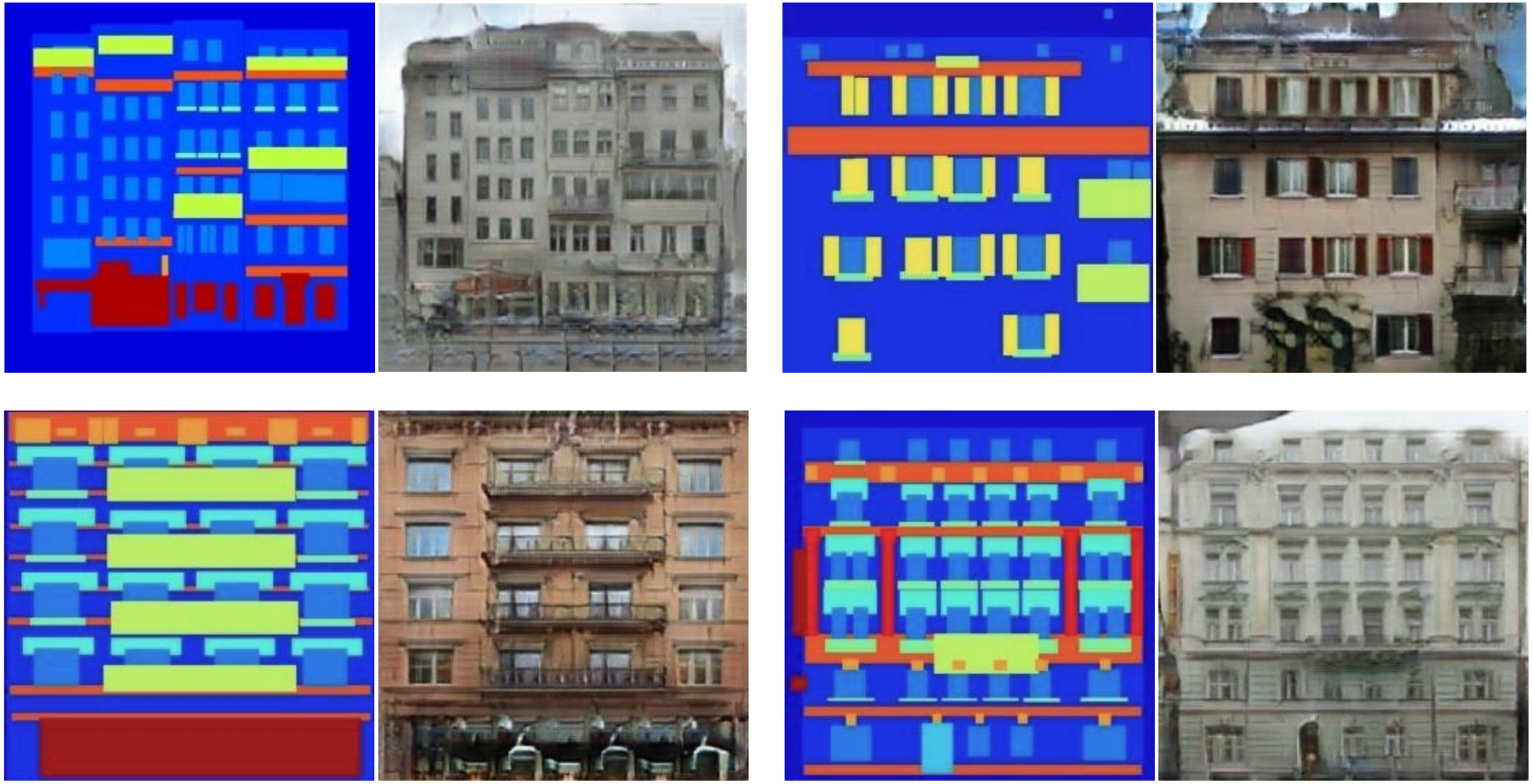


Output

Data from [Tylecek, 2013]

Labels → Facades

Input | Output | Input | Output



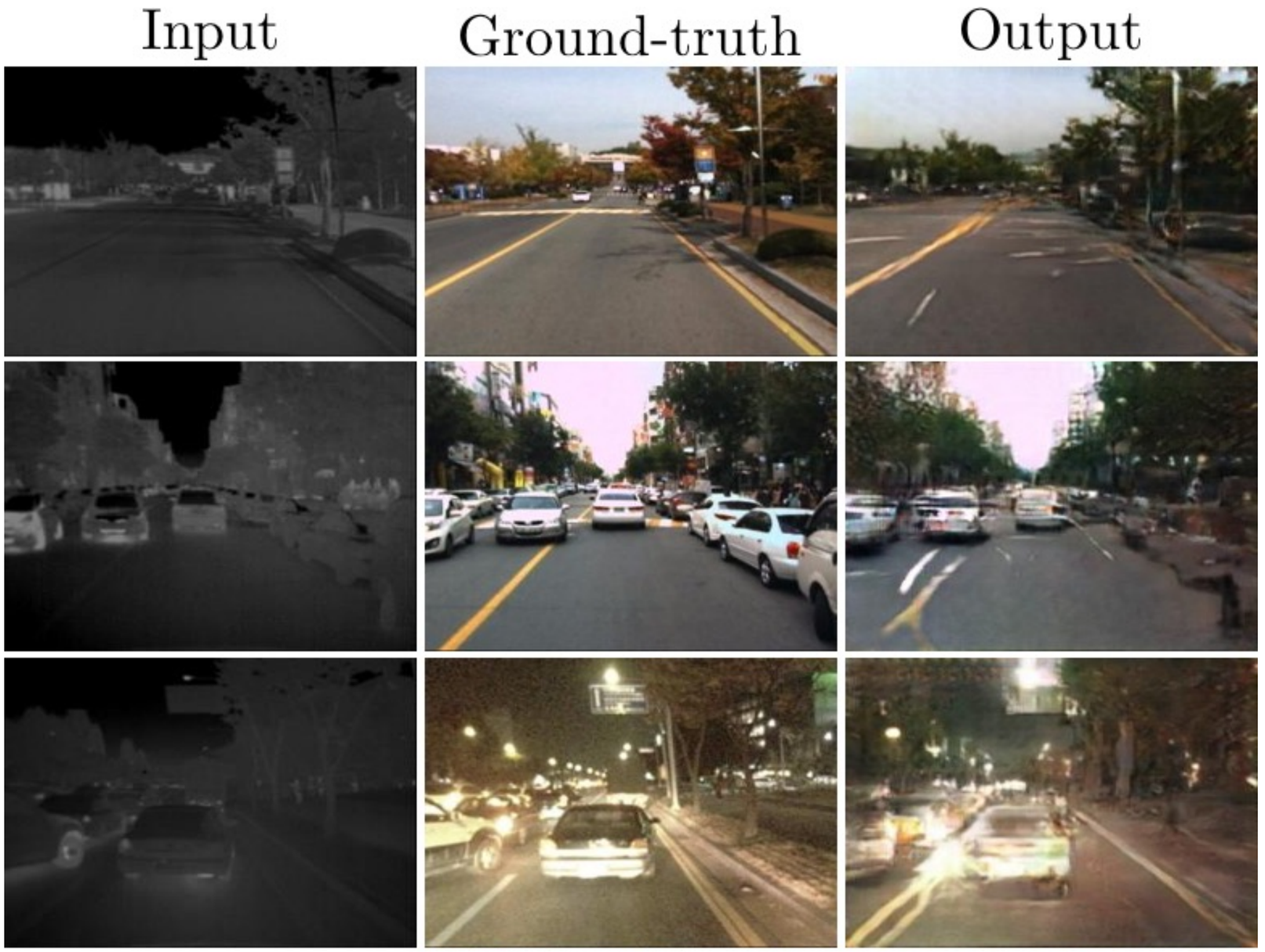
Data from [Tylecek, 2013]

Day → Night

Input Output Input Output Input Output

Data from [Laffont et al., 2014]

Thermal → RGB



Edges → Images

Input | Output | Input | Output | Input | Output

Edges from [Xie & Tu, 2015]

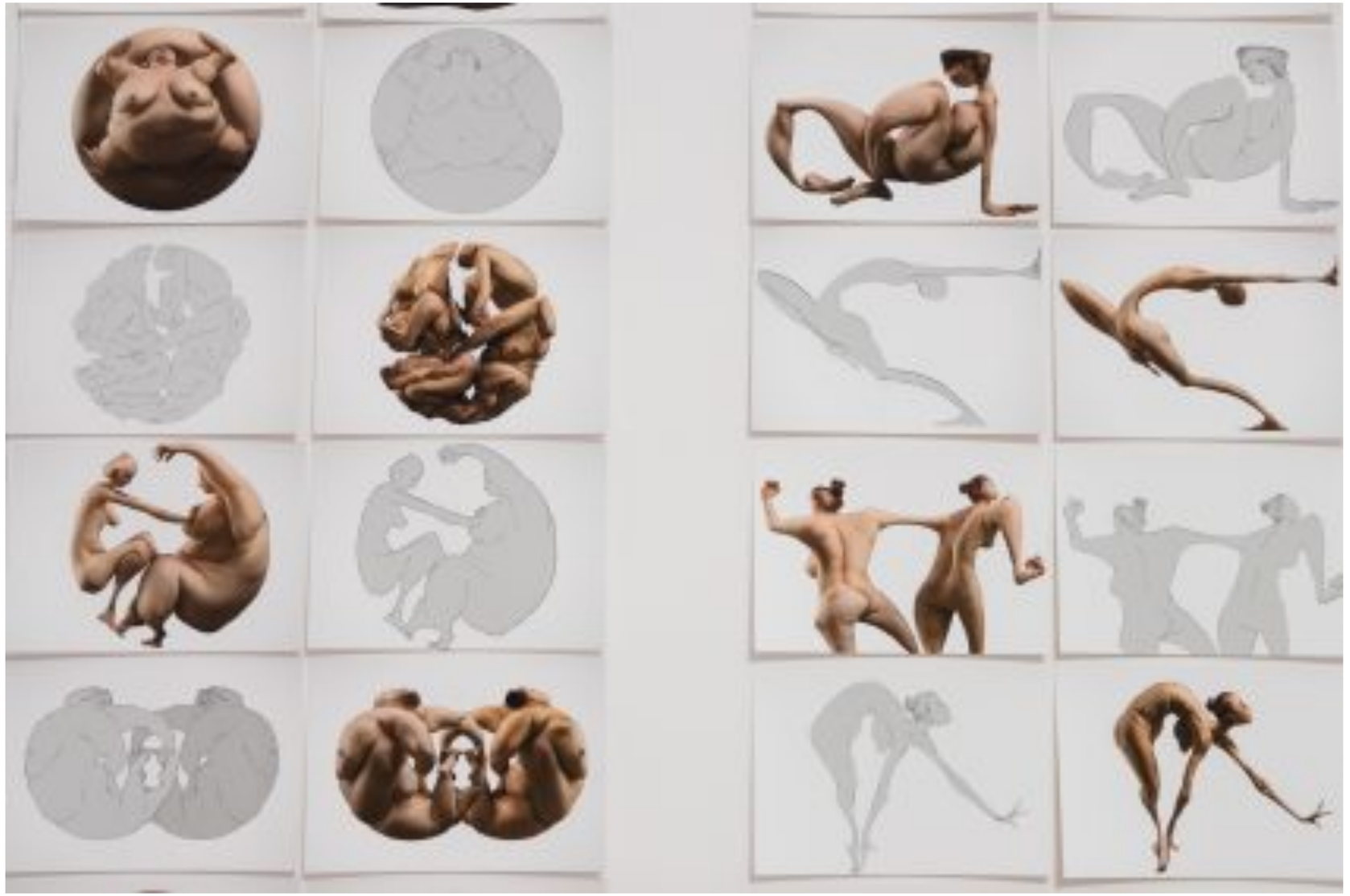
Sketches → Images

Input | Output | Input | Output | Input | Output

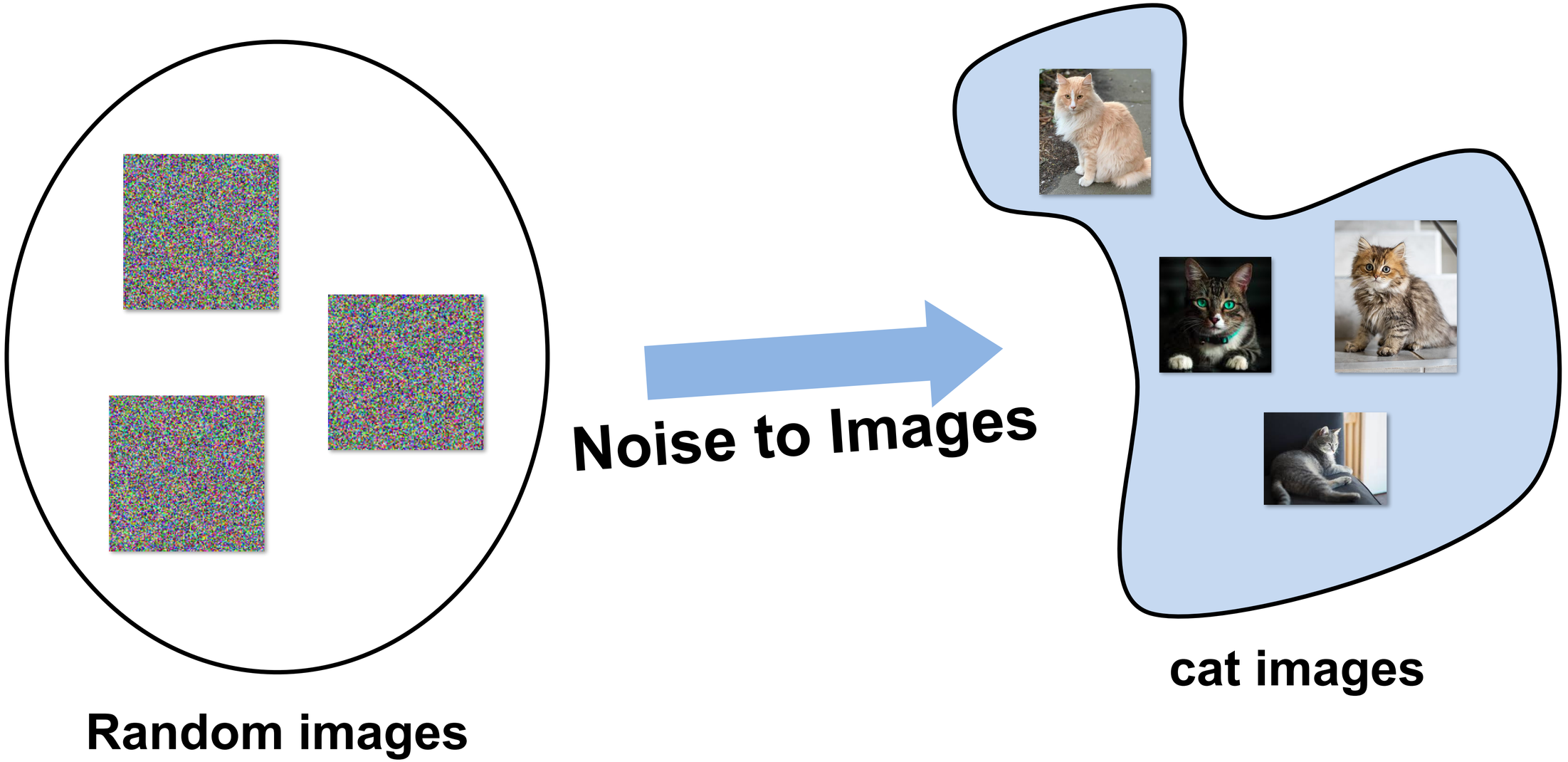
Trained on Edges → Images

Data from [Eitz, Hays, Alexa, 2012]

Scott Eaton (http://www.scott-eaton.com/)

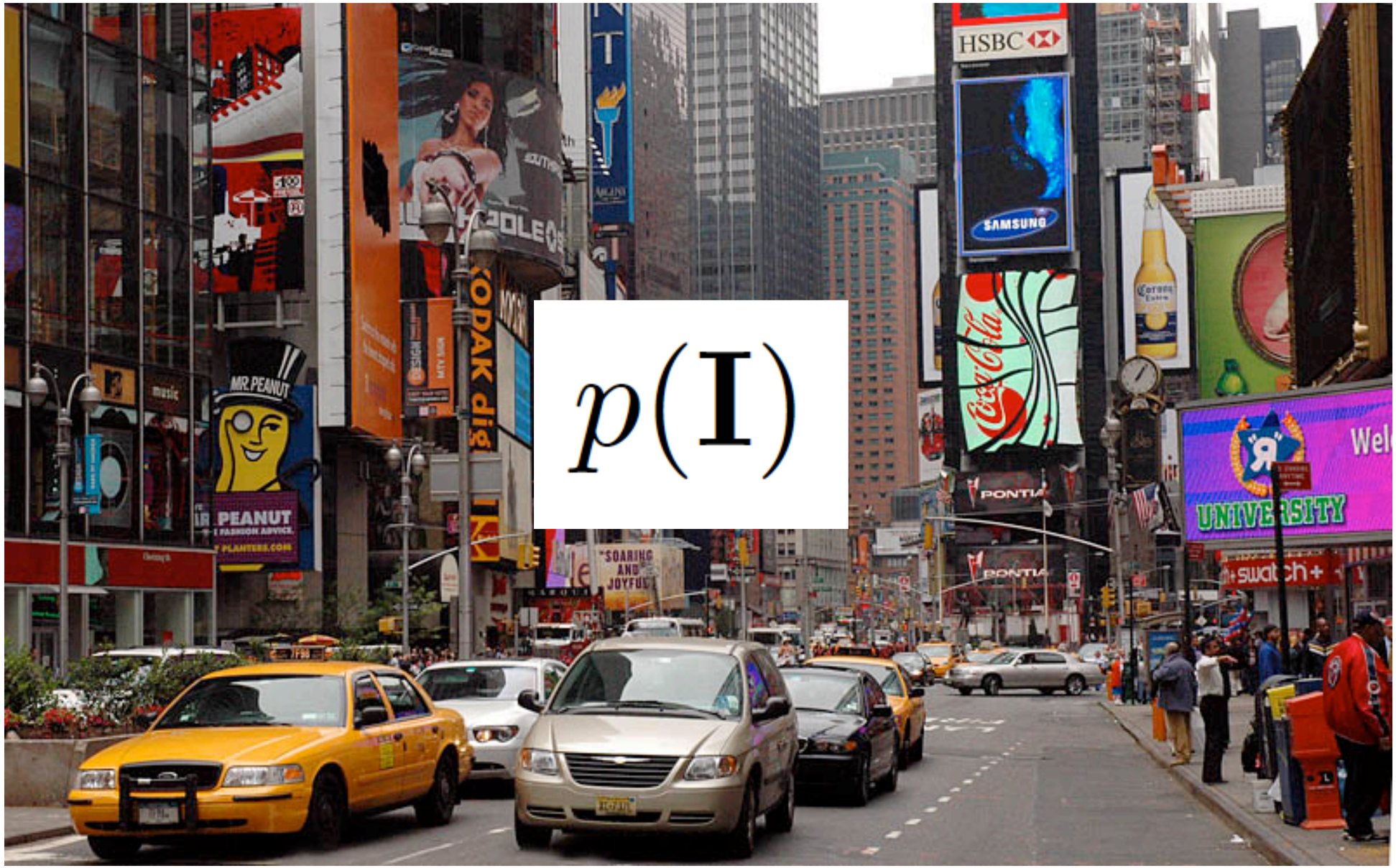


Generating Images from Scratch

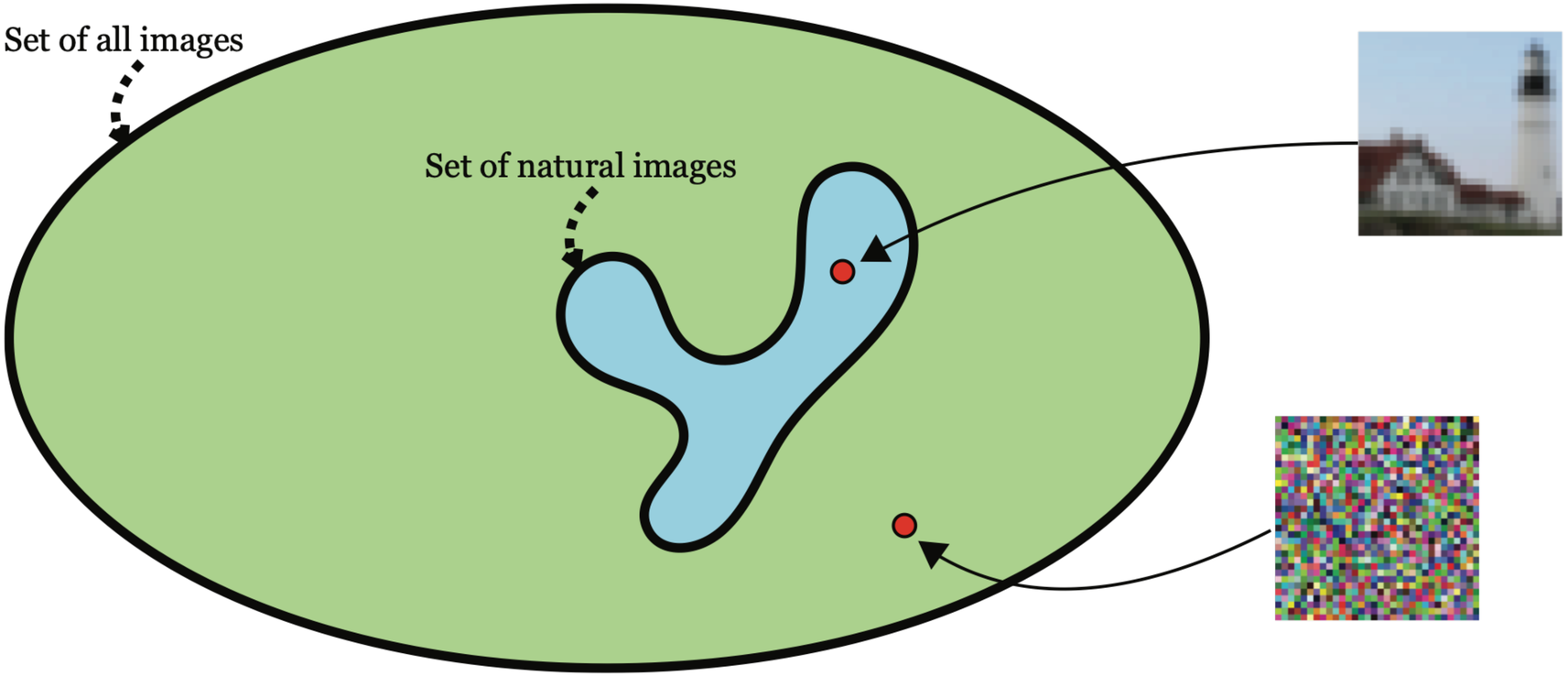


Slide source: Steve Seitz

Statistical modeling of images

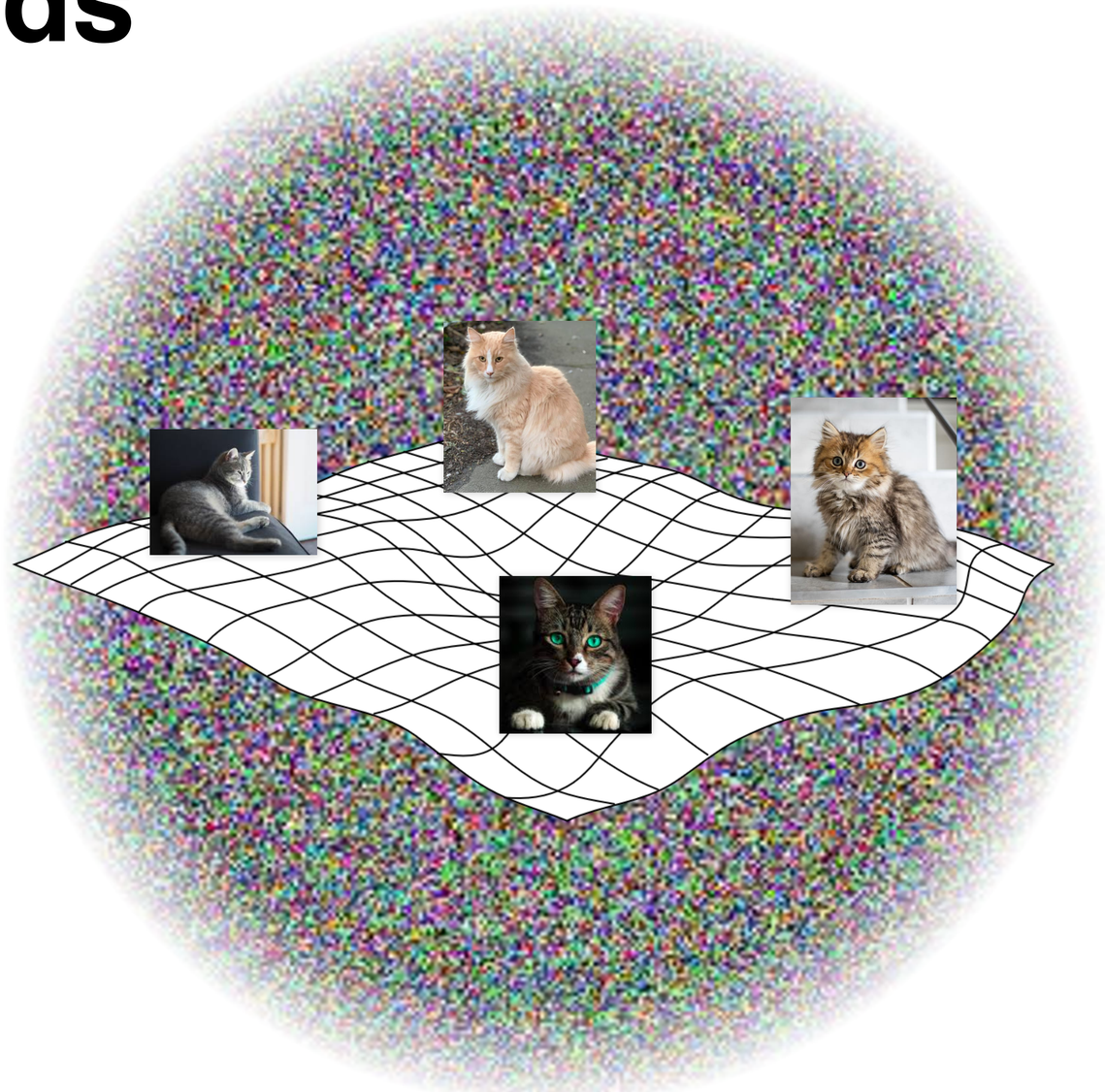


Statistical modeling of images



Natural Image Manifolds

- Most images are “noise”
- “Meaningful” images tend to form some manifold within the space of all images



The Space of All Images

Slide source: Steve Seitz