Image Synthesis
(Plus some bonuses)

EECS 443 – David Fouhey

Winter 2022, University of Michigan

http://web.eecs.umich.edu/~fouhey/teaching/EECS442_W23/

https://github.com/NVlabs/stylegan3 [Karras et al., “Alias-Free Generative Adversarial Networks”, 2021]

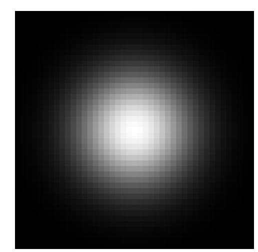
How Many Images Are There?

- Set height and width to 1024
- Assume 256^3 (aka 2^{24}) values per pixel
- **How many images can I create?**
- $(256^3)^{\sim 1M}$
- **Why might it be quite a bit less?**

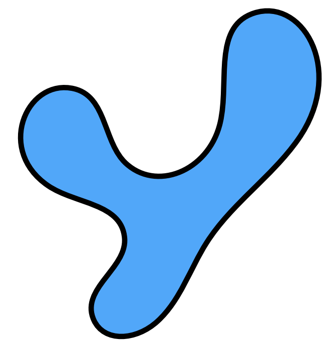
Learning a Mapping

- Want to learn a mapping
- **From:** a "latent" space z, often assume to be the result of sampling N-D Gaussian noise
- **To:** the space of valid images

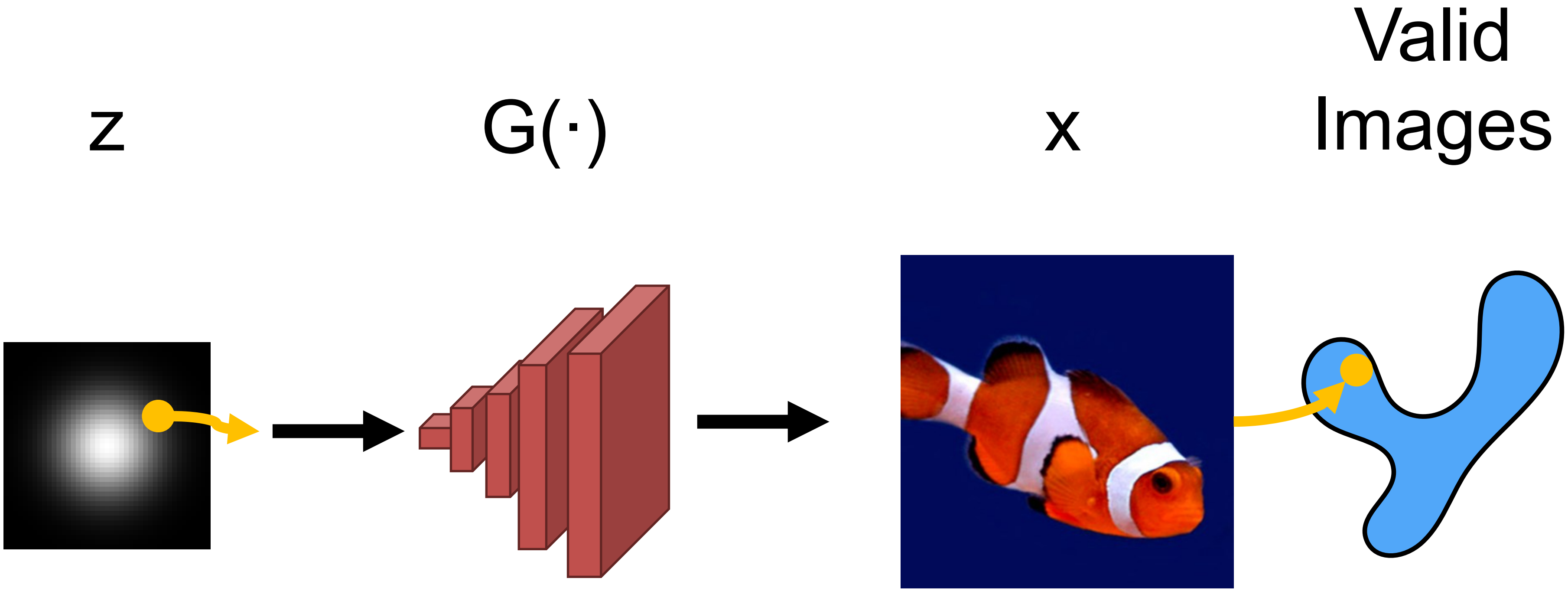
z



Valid Images

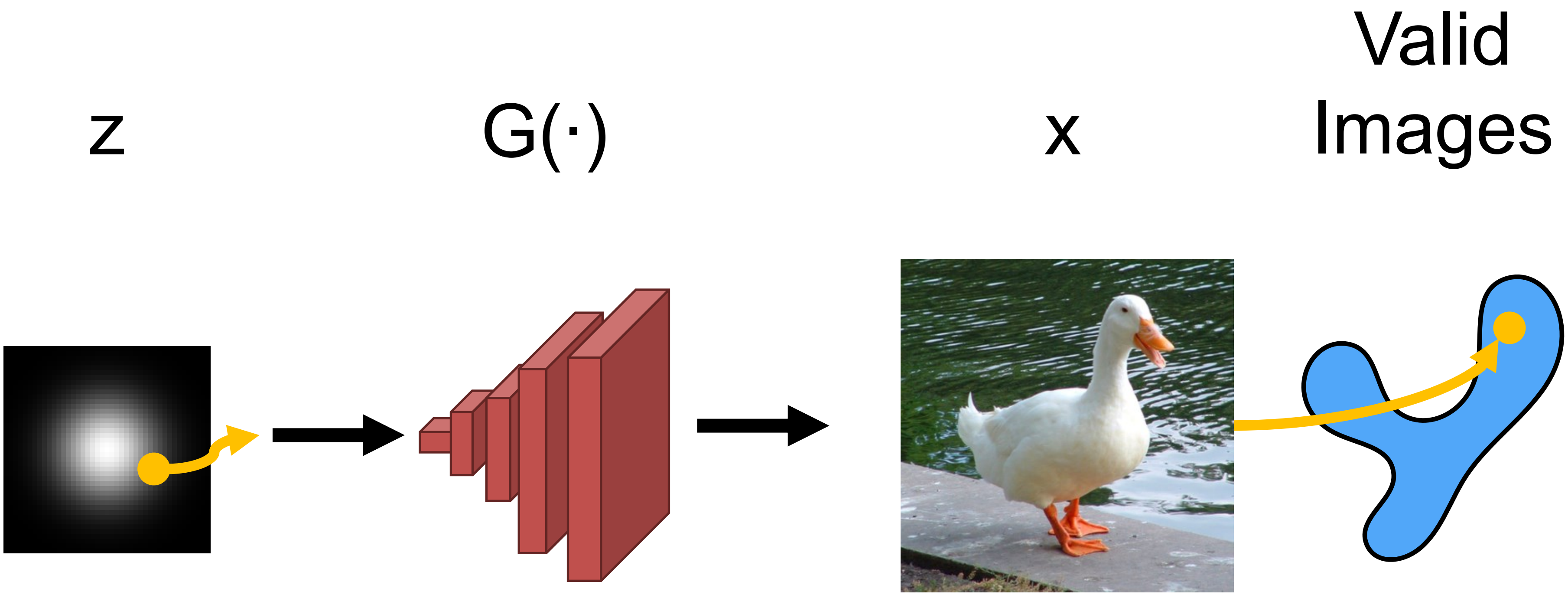


Generating Data



Remake of Slides by Isola, Freeman, Torralba, taken from Owens; diagram, duck + clownfish pictures from IFT.

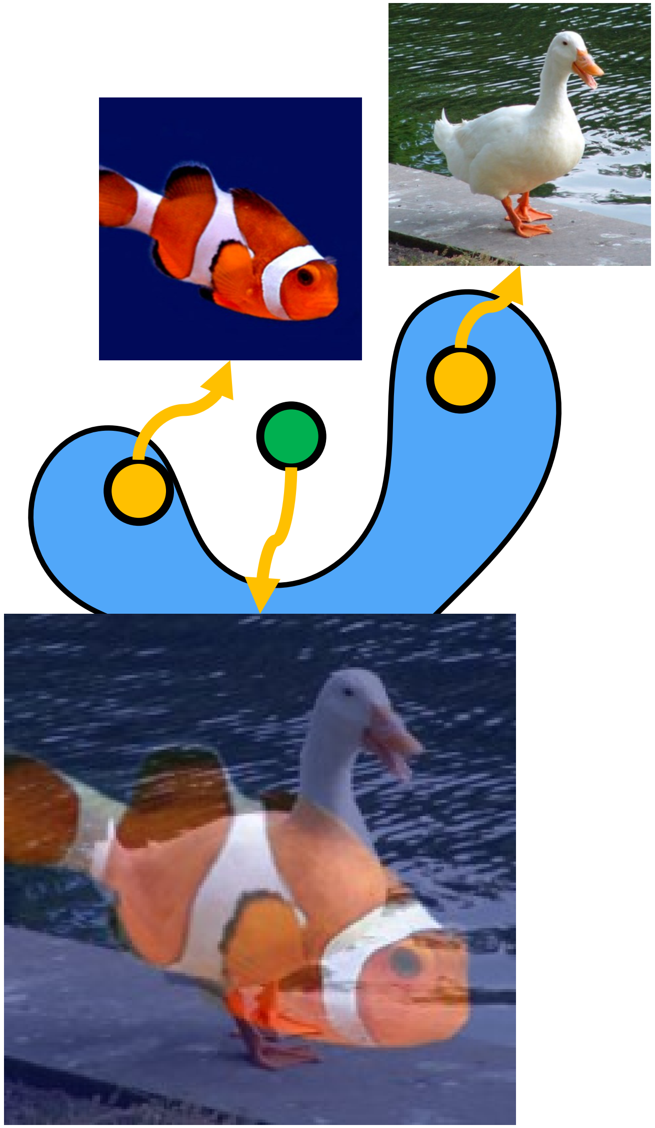
Generating Data



Remake of Slides by Isola, Freeman, Torralba, taken from Owens; diagram, duck + clownfish pictures from IFT.

Why The Funny Shape?

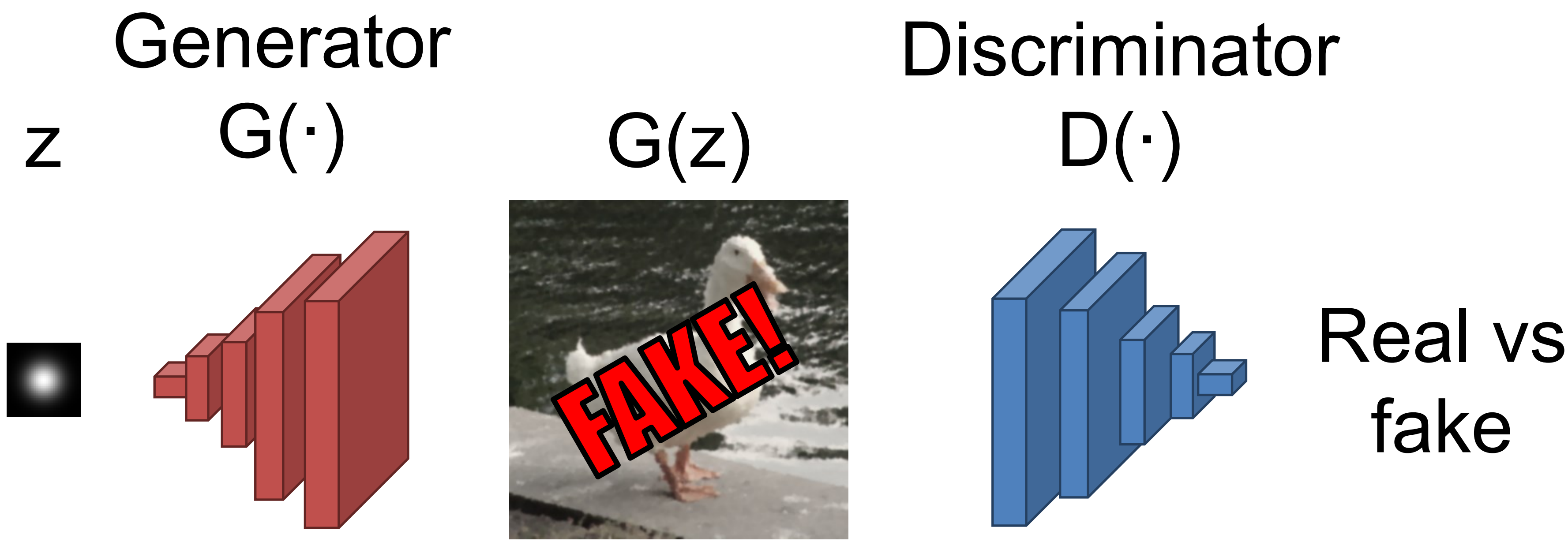
Given two valid images, what about their average?



Key things to remember
- Linear combinations of images aren't images.
- Explains funny shape ("manifold") and need for a deep network

Generative Adversarial Networks

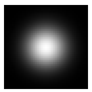
- Generator tries to make fake images – accepts noise and makes an image
- Discriminator tries to identify fakes – outputs p(fake)

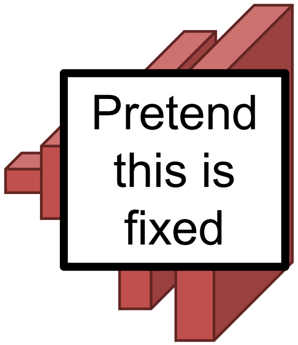


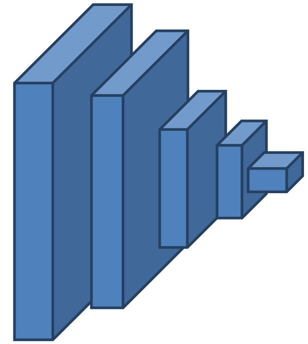
Remake of Slides by Isola, Freeman, Torralba, taken from Owens; diagram, duck + clownfish pictures from IFT.

Generative Adversarial Networks

z G(·) G(z) D(·)



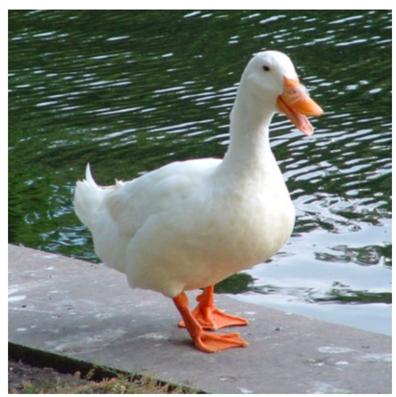


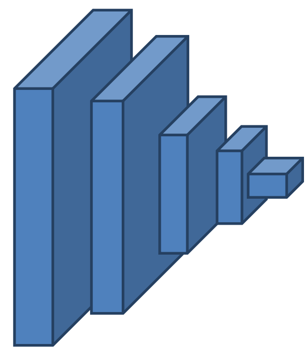


Want: fake (high)

x D(·)

1-D(x) = p(x is real)
Want to maximize log p(real) for real data





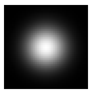
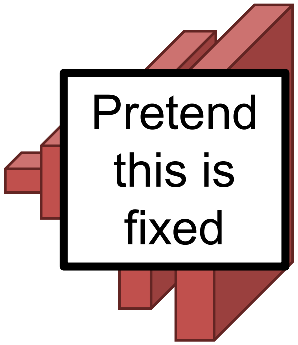
Want: real (low)

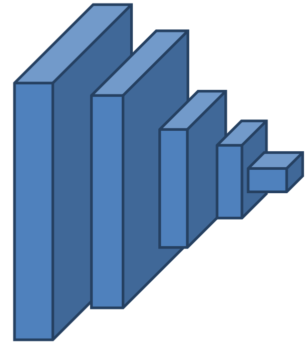
$$\arg\max_{D} E_{z,x}[\log(D(G(z))) + \boxed{\log(1 - D(x))}]$$

Remake of Slides by Isola, Freeman, Torralba, taken from Owens; diagram, duck + clownfish pictures from IFT.

Generative Adversarial Networks

z G(·) G(z) D(·)

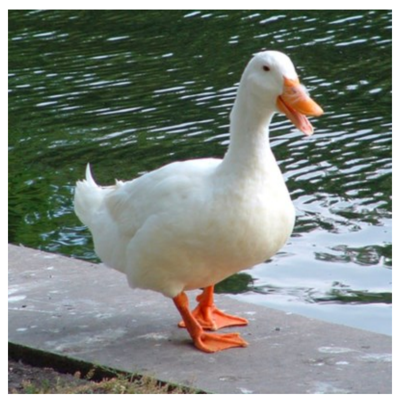
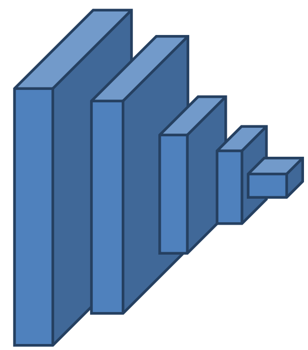





Want: fake (high)

x D(·)

D(G(z)) = p(G(z) is fake)
Want to maximize log p(fake) for fake data

Want: real (low)

$$\arg\max_{D} E_{z,x}\left[\boxed{\log\left(D\left(G(z)\right)\right)} + \log\left(1 - D(x)\right)\right]$$

Remake of Slides by Isola, Freeman, Torralba, taken from Owens; diagram, duck + clownfish pictures from IFT.

Generative Adversarial Networks

Goal of generator G: fool the discriminator D while getting to use gradients from D
Analogy: art forger and art detective

How good are you at spotting forgeries?

$$\arg\min_{G} E_{z,x}[\log(D(G(z)) \;\; + \log(1 - D(x))]$$

Remake of Slides by Isola, Freeman, Torralba, taken from Owens; diagram, duck + clownfish pictures from IFT.

Generative Adversarial Networks

Final goal: find the generator that fools the best D that you could find.

In practice, important not to let the discriminator get too good. **Why?**

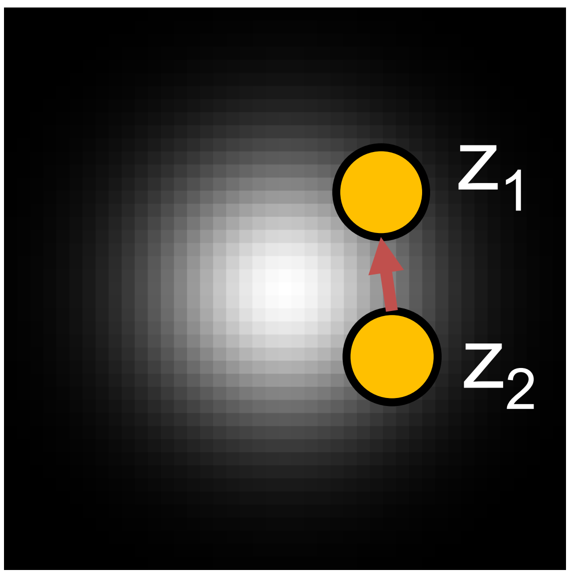
Theory: optimum when G produces distribution

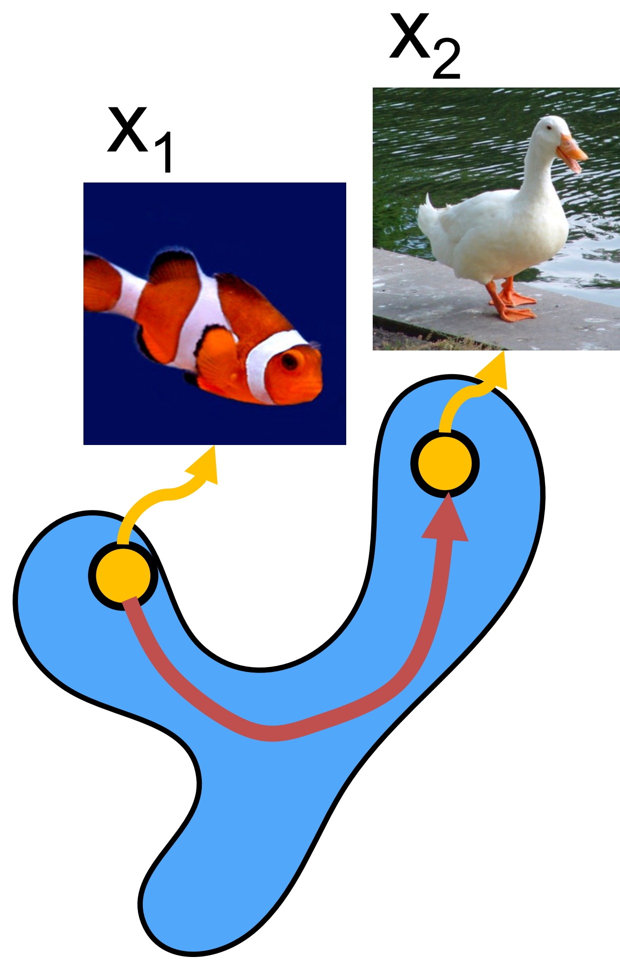
$$\arg\min_{G}\max_{\mathrm{D}} E_{z,x}[\log(D(G(z)) \quad + \log(1 - D(x))]$$

Remake of Slides by Isola, Freeman, Torralba, taken from Owens; diagram, duck + clownfish pictures from IFT.

Revisiting Averages

Can use z to walk latent space
$G(\alpha z_1+(1-\alpha)z_2)$ for α in [0,1]





Remake of Slides by Isola, Freeman, Torralba, taken from Owens; diagram, duck + clownfish pictures from IFT.

[Karras et al., “Alias-Free Generative Adversarial Networks”, 2021]

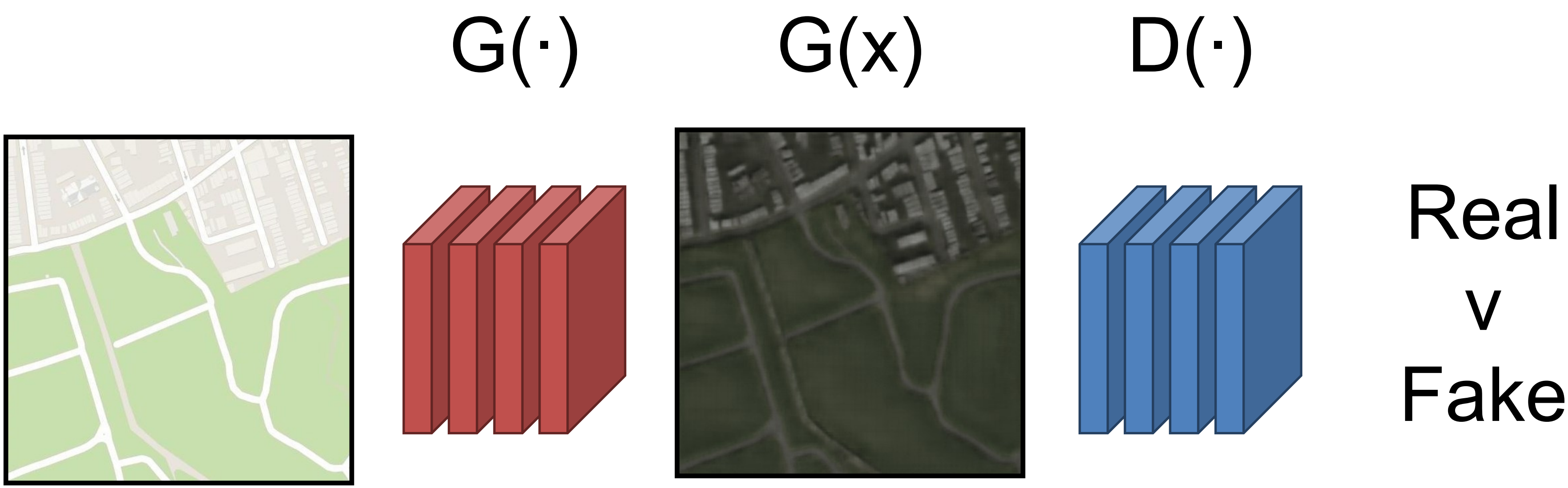
[Karras et al., “Alias-Free Generative Adversarial Networks”, 2021]

Conditioning on Things

- Turning noise into pictures of things is all fun and good, but what if we want control over our synthetic images?

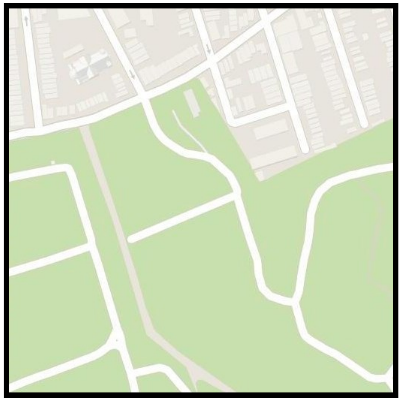
Conditional GANs (Pix2Pix)

- Generator tries to make fake images – accepts **image** and makes an image
- Discriminator tries to identify fakes – outputs p(fake), potentially at each pixel

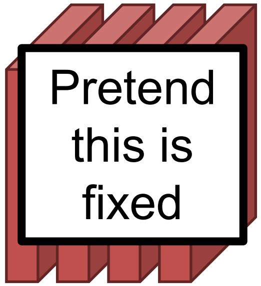


Remake of Slides by Isola, Freeman, Torralba, taken from Owens.

Conditional GANs – Discriminator

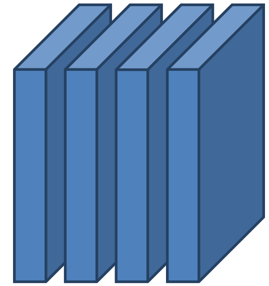


G(·)



G(x)

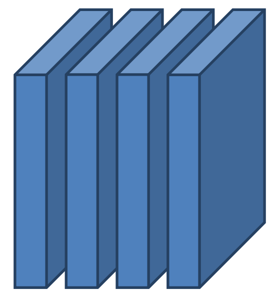
D(·)



Want:
Fake

y

D(·)



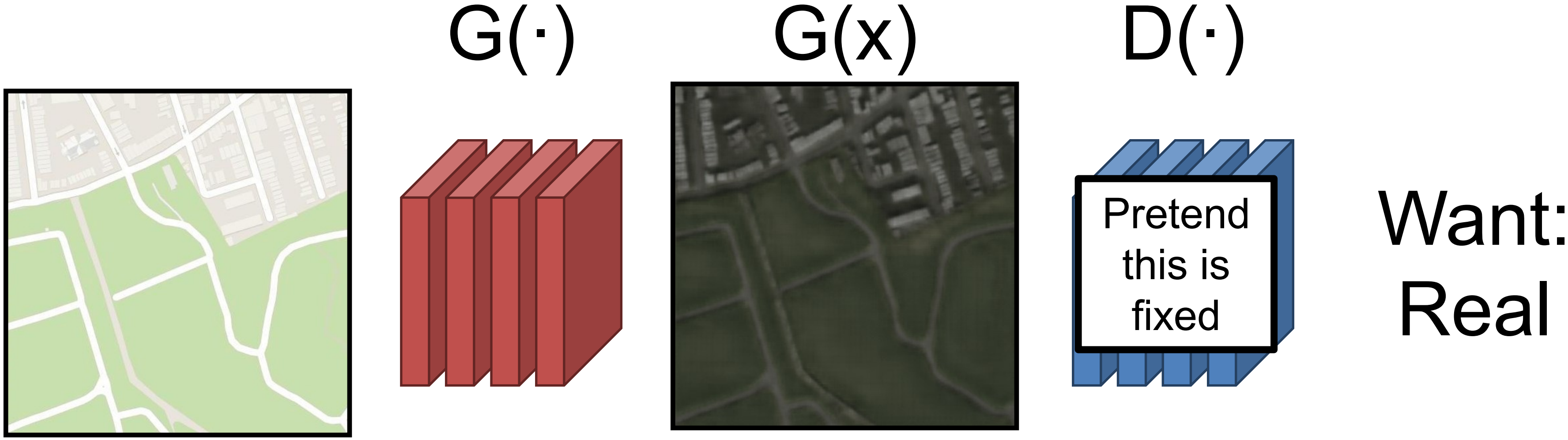
Want:
Real

Want real outputs y to be low, fake output G(x) high

$$\arg\max_{\mathrm{D}} E_{z,x}[\,\log(D\big(G(x)\big) \;+ \log\big(1 - D(y)\big)]$$

Remake of Slides by Isola, Freeman, Torralba, taken from Owens.

Conditional GANs – Generator



If you're the generator, want to make fakes that fool the discriminator into think they're real

$$\arg\min_{G} E_{z,x}[\log(D(G(x)) + \log(1 - D(y))]$$

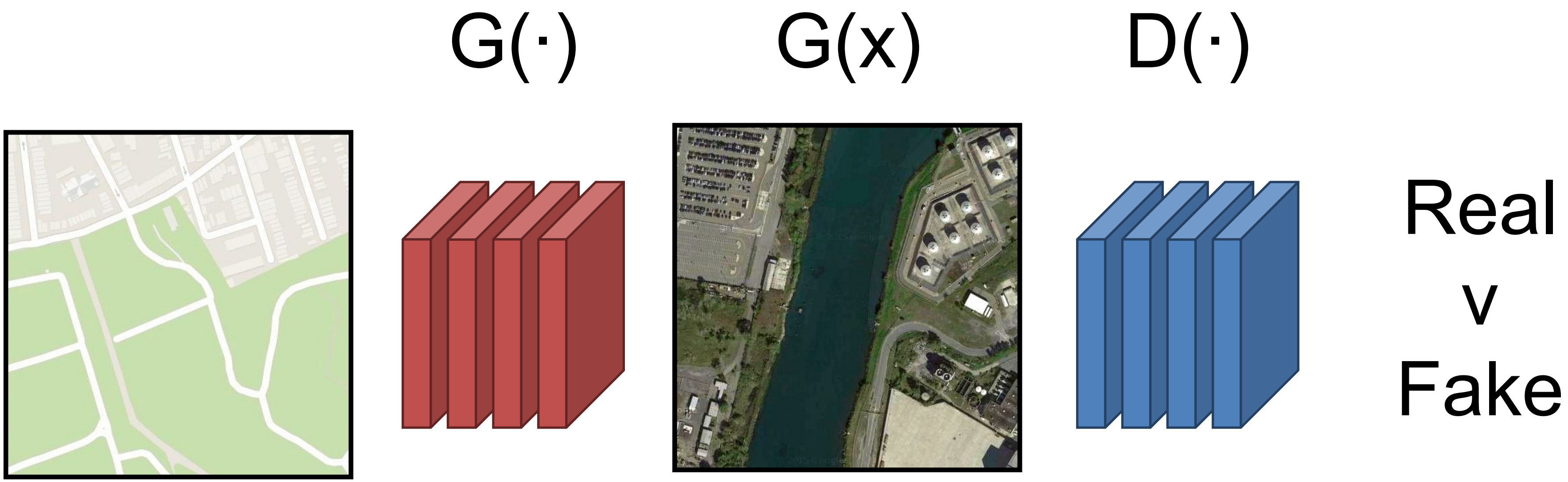
Same min/max game as before

$$\arg\min_{G}\max_{D} E_{z,x}[\log(D(G(x)) + \log(1 - D(y))]$$

Remake of Slides by Isola, Freeman, Torralba, taken from Owens.

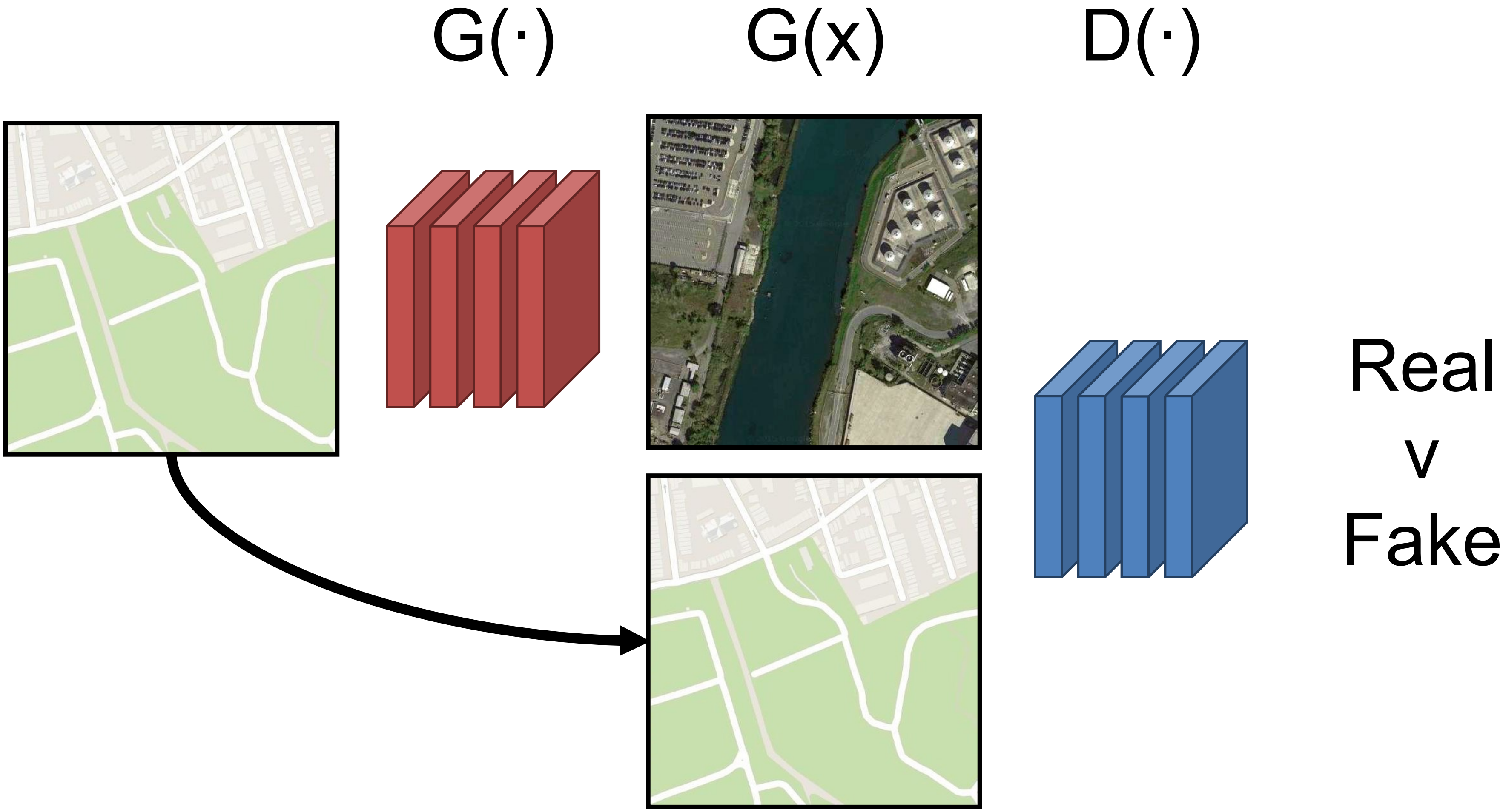
One Catch

- G can just output random good images.
- Solution – make the D look at the input too



Remake of Slides by Isola, Freeman, Torralba, taken from Owens.

One Catch



Remake of Slides by Isola, Freeman, Torralba, taken from Owens.

More Broadly

- Neural networks are lazy and will do precisely what you ask and no more
- You *have to* be careful what you ask them to do

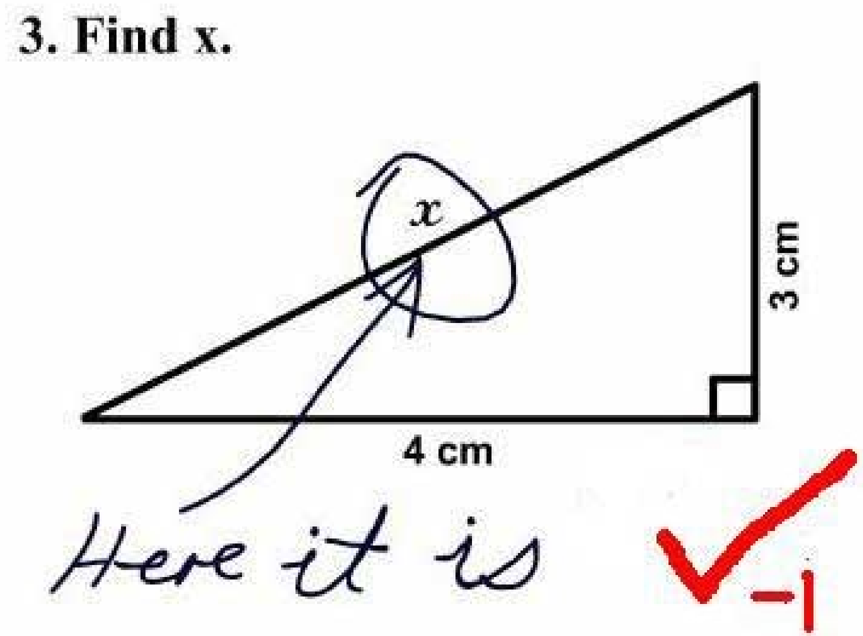


Photo credit: I found it at uregina.ca, but this is an old classic

Input Output Groundtruth

Data from
[maps.google.com]

Slide by Isola, Freeman, Torralba, taken from Owens.

Input

Output

Groundtruth

Data from [maps.google.com]

Slide by Isola, Freeman, Torralba, taken from Owens.

Input

L1 loss

Why is it blurry junk? Hold that thought!

Slide by Isola, Freeman, Torralba, taken from Owens.

Input

L1 loss + discriminator

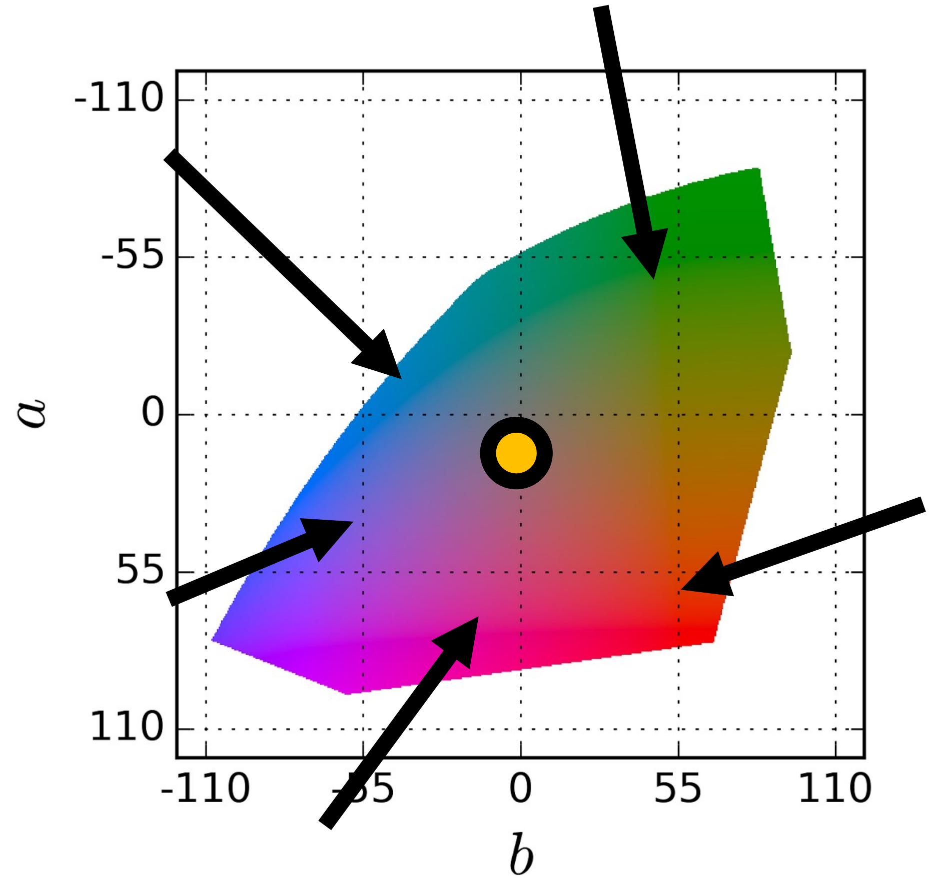
Let’s Talk About Blurry Pictures

What Color Is This Bird?

To make things more concrete: what color is this the pixel under this gold circle?

Example credit and diagrams: Richard Zhang

What Color Is This Bird?



Many options. **What minimizes mean-squared error?**
What minimizes the L1 distance?

Example credit and diagrams: Richard Zhang

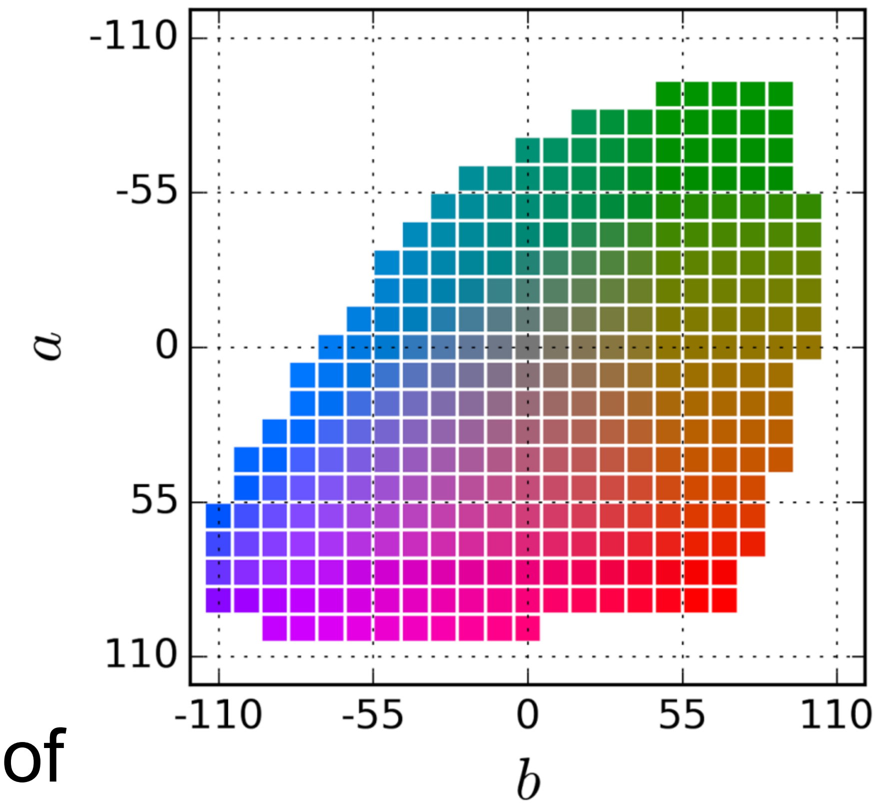
What Color Is This Bird?

Option 1: Discretize / quantize

Before learning: assign pixel nearest color index

After: convert index to value

Works because network can express its uncertainty. Loads of details if you want a good version.

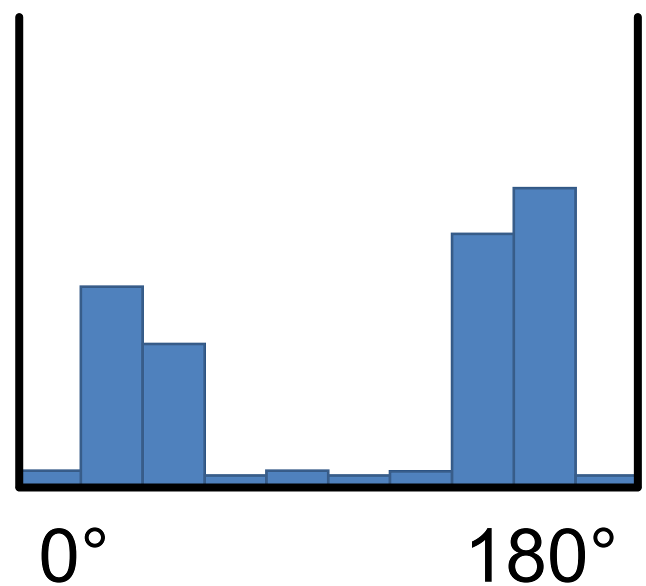


Example credit and diagrams: Richard Zhang

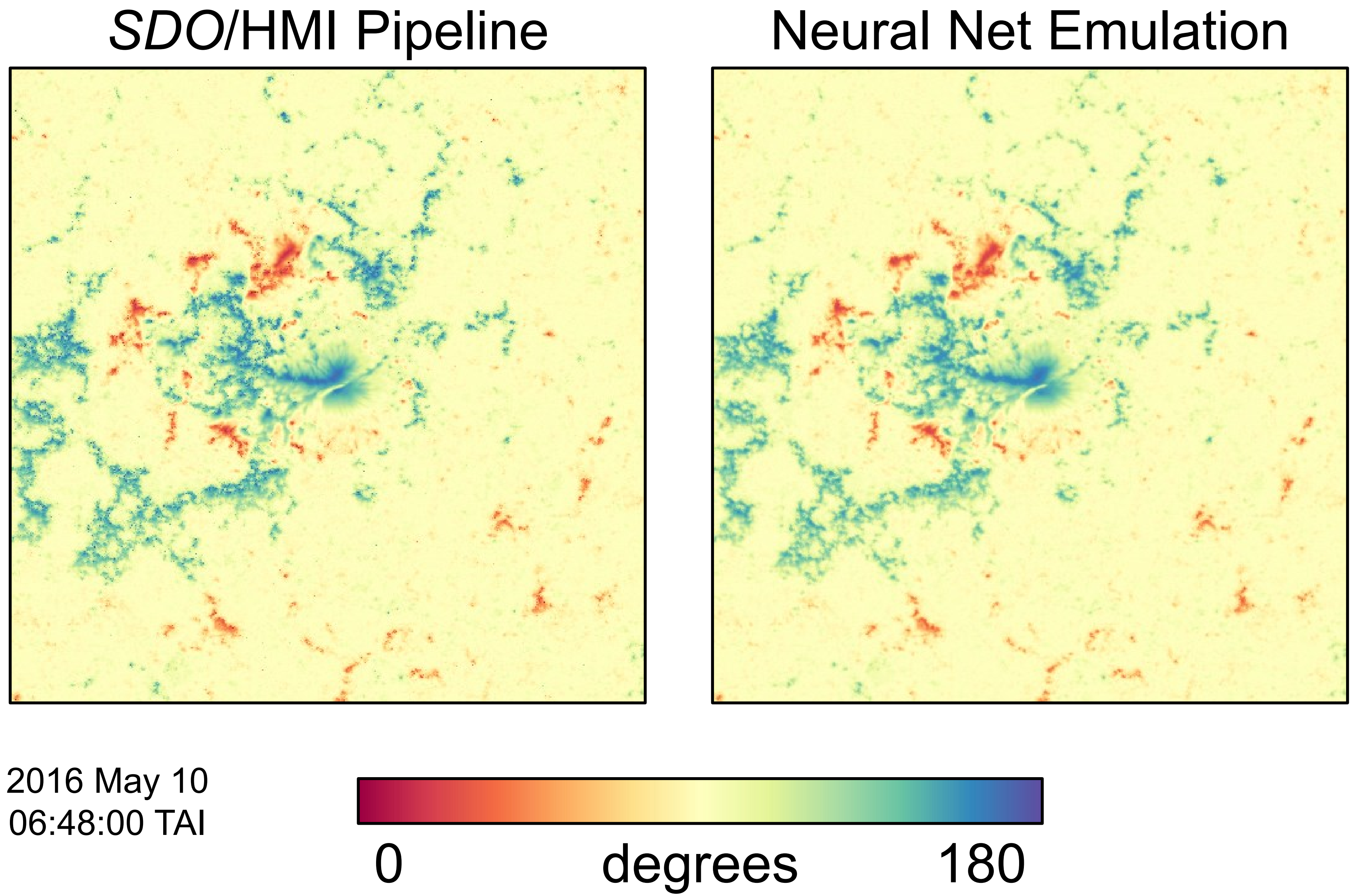
Discretized Values – Angles

Imagine predicting an angle from 0° to 180°. Having bins enables:

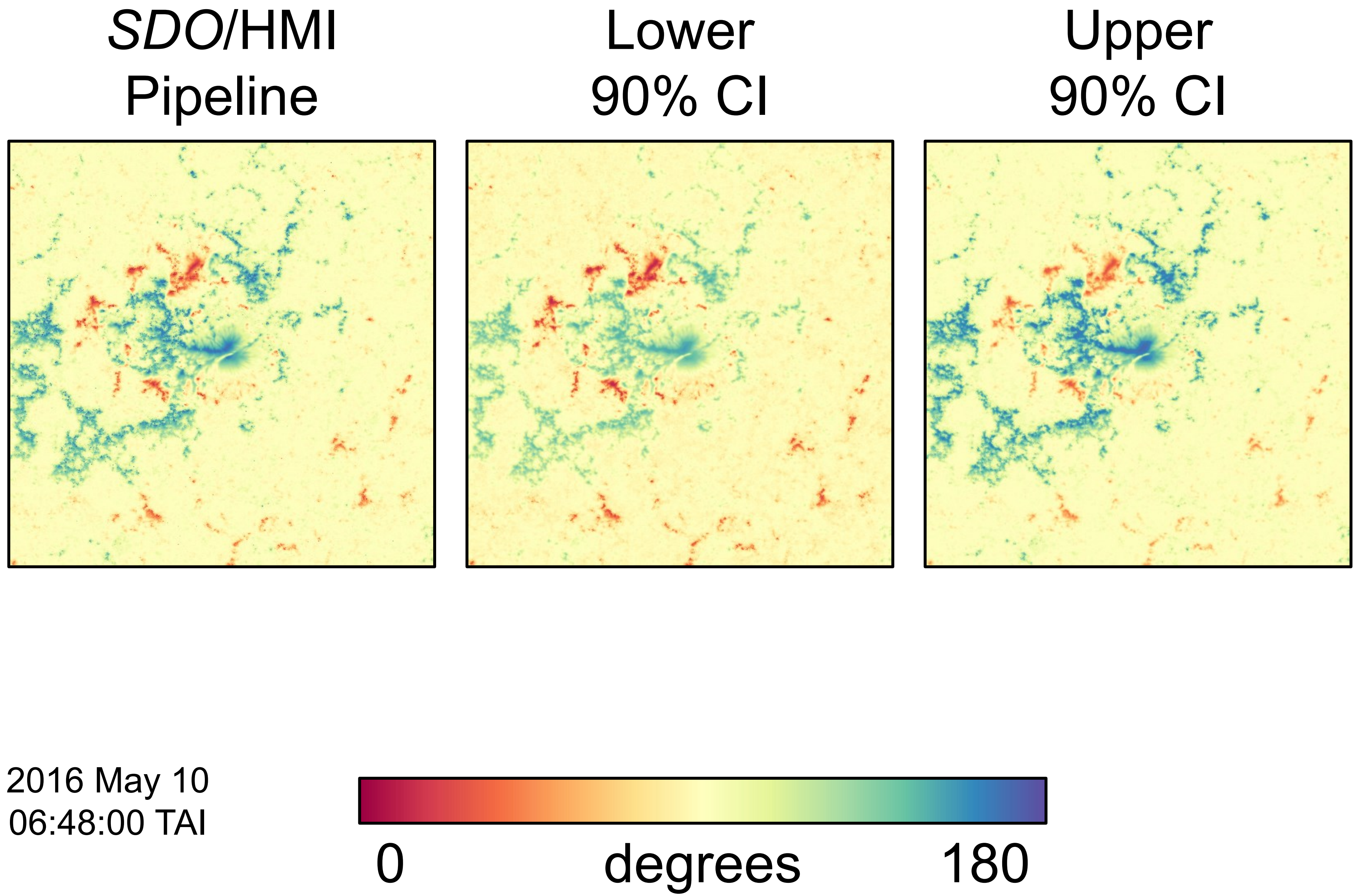
- Expressing bimodal distributions (e.g., either 30° or 150°)
- Getting a confidence from the prediction



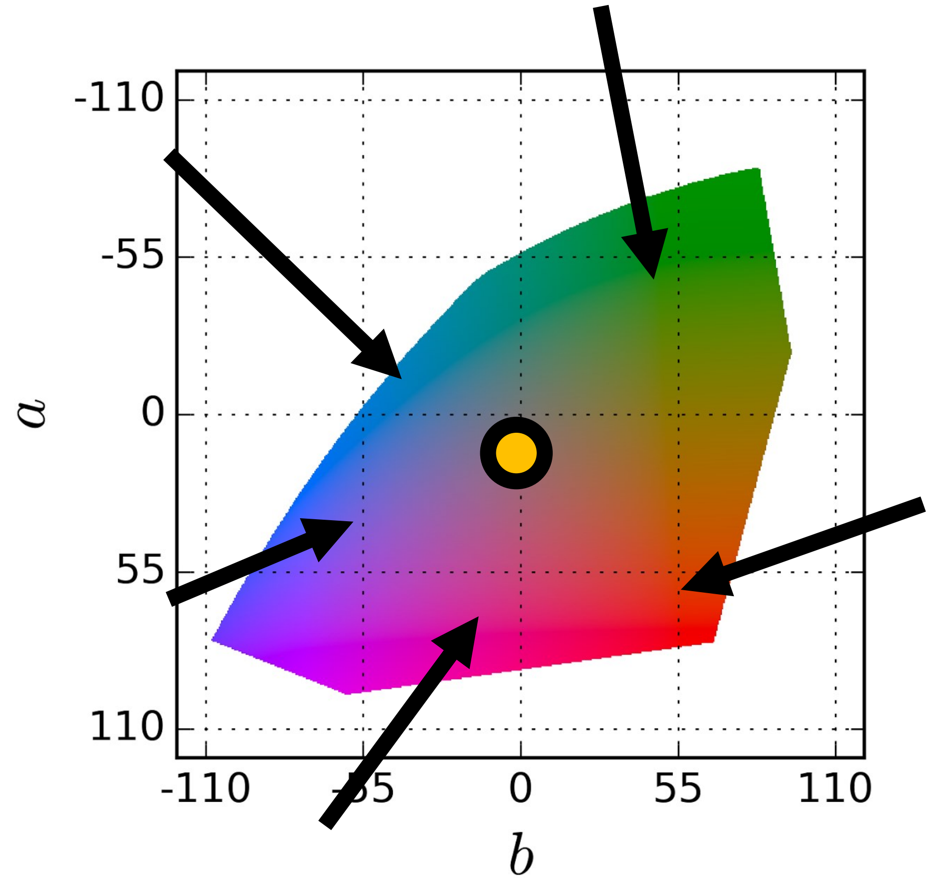
Discretized Values – Angles



Discretized Values – Angles



Option 2 – GAN



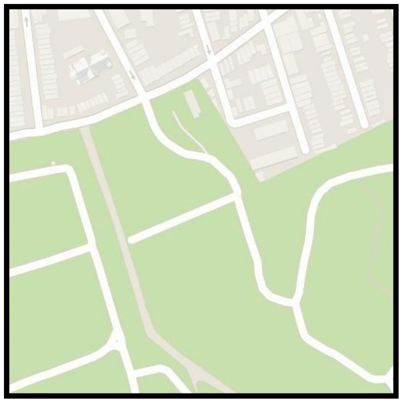
GAN – the discriminator will prevent us from making birds grey or brown. **Why?**

Example credit and diagrams: Richard Zhang

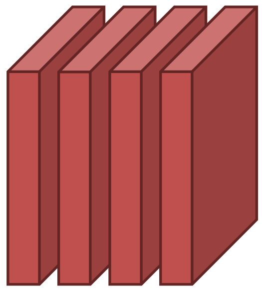
What to Take Away

- Be careful what you ask a deep net to solve.
- The objective you’re asking it to solve bakes in assumptions
- Most solutions broken in one way or another
- Deep learning is not magic

Aside: Perceptual Losses



G(·)



G(x)

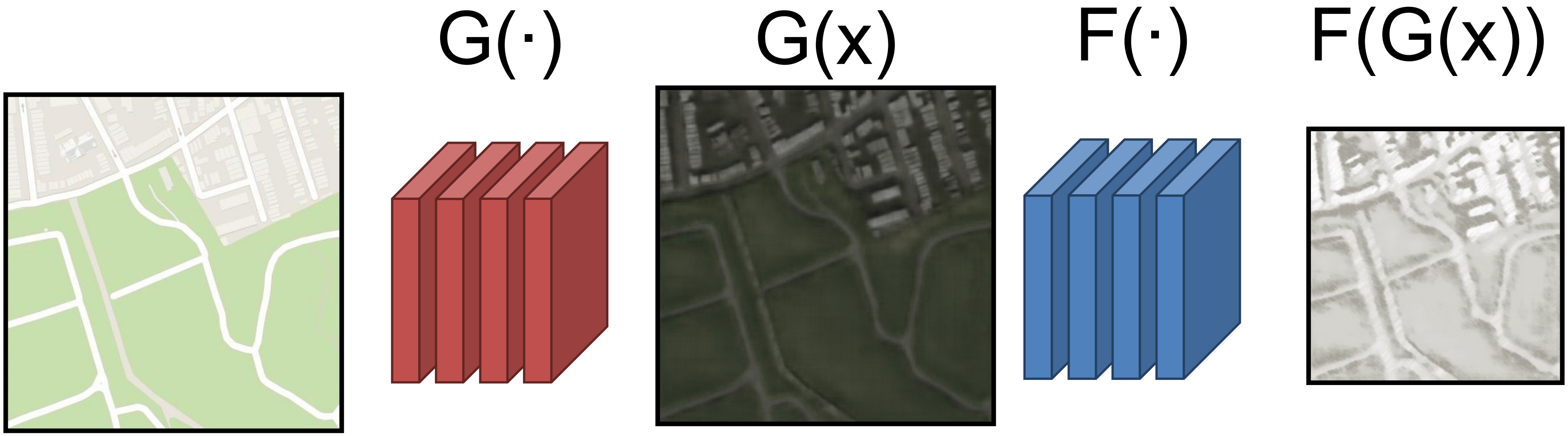
y

Conventionally, minimize distance in pixel space:

$$||G(\boldsymbol{x}) - \boldsymbol{y}||$$

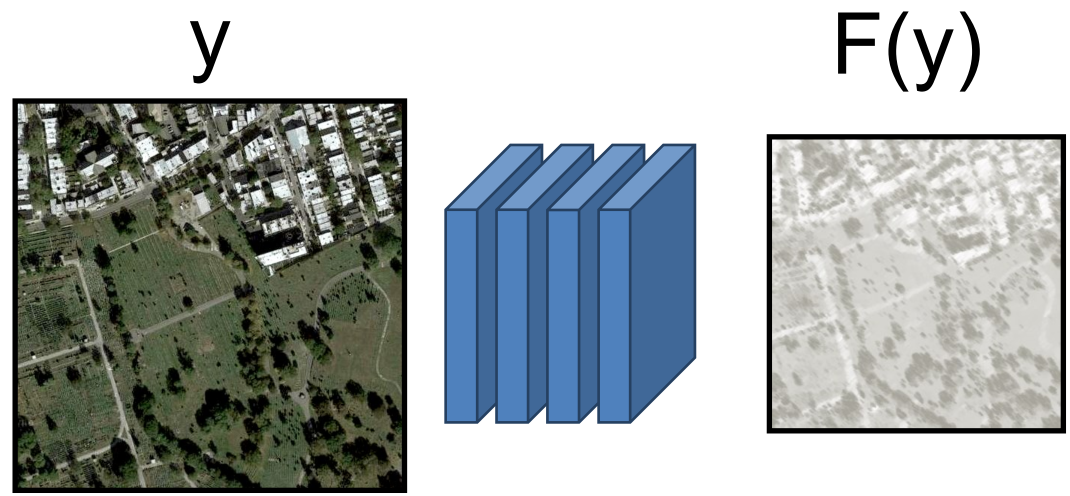
See Johnson et al. Perceptual Losses for Real-Time Style Transfer and Super-Resolution, ECCV 2016.

Aside: Perceptual Losses



Instead measure distance after passing through network

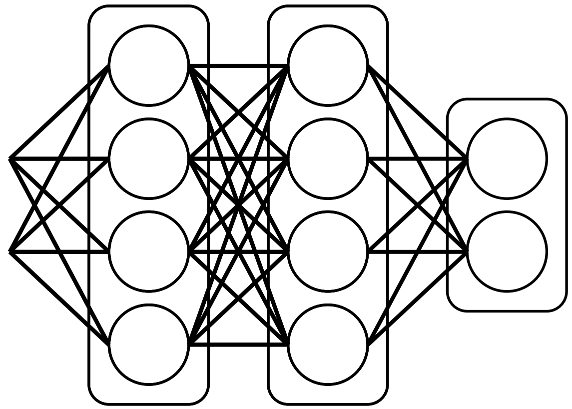
$$||F(G(\boldsymbol{x})) - F(\boldsymbol{y})||$$



See Johnson et al. Perceptual Losses for Real-Time Style Transfer and Super-Resolution, ECCV 2016.

And Now For Something Completely Different

ImageNet + Deep Learning

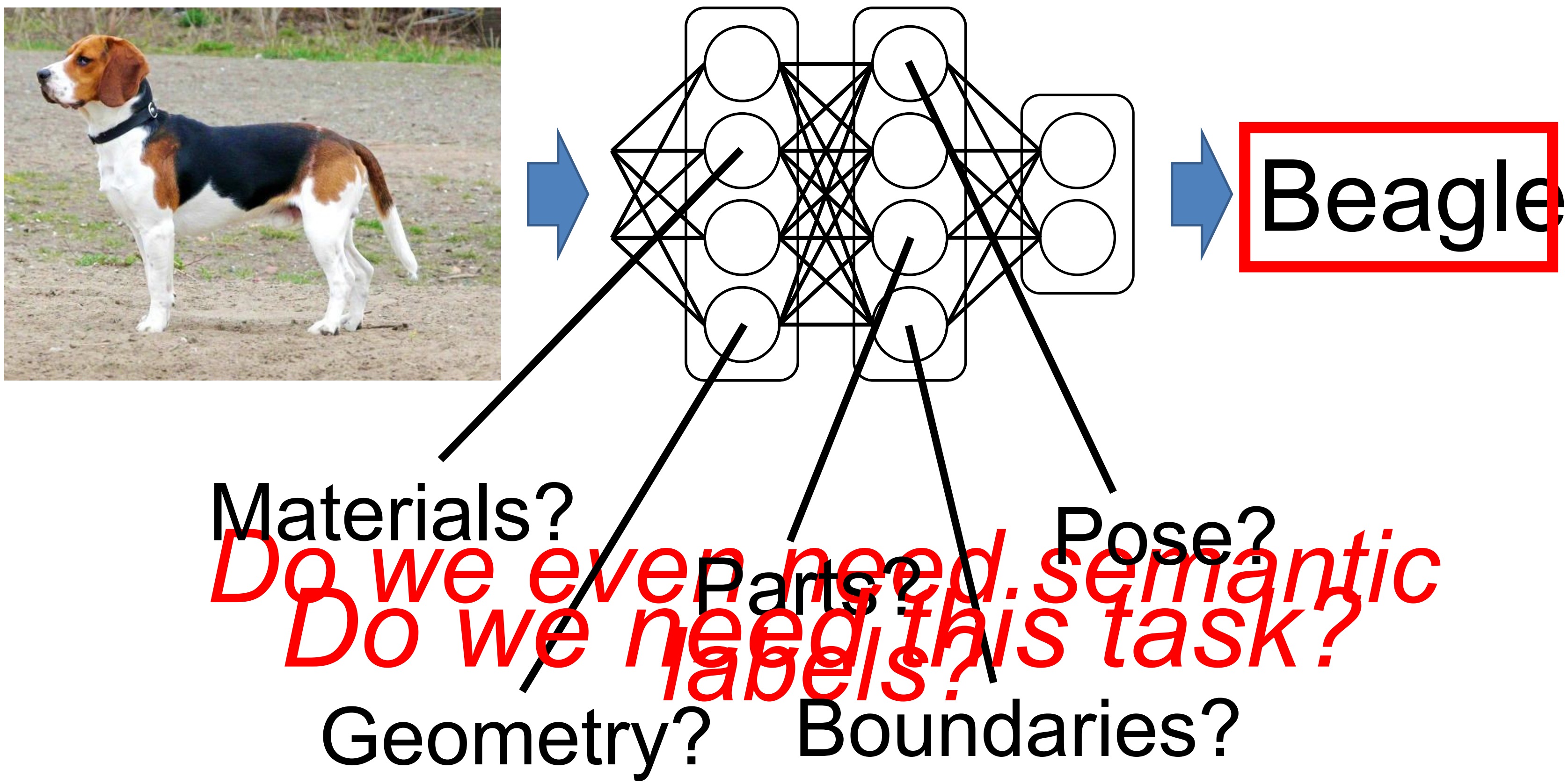


Beagle

- Image Retrieval
- Detection
- Segmentation
- Depth Estimation
- …

Slide Credit: C. Doersch

ImageNet + Deep Learning



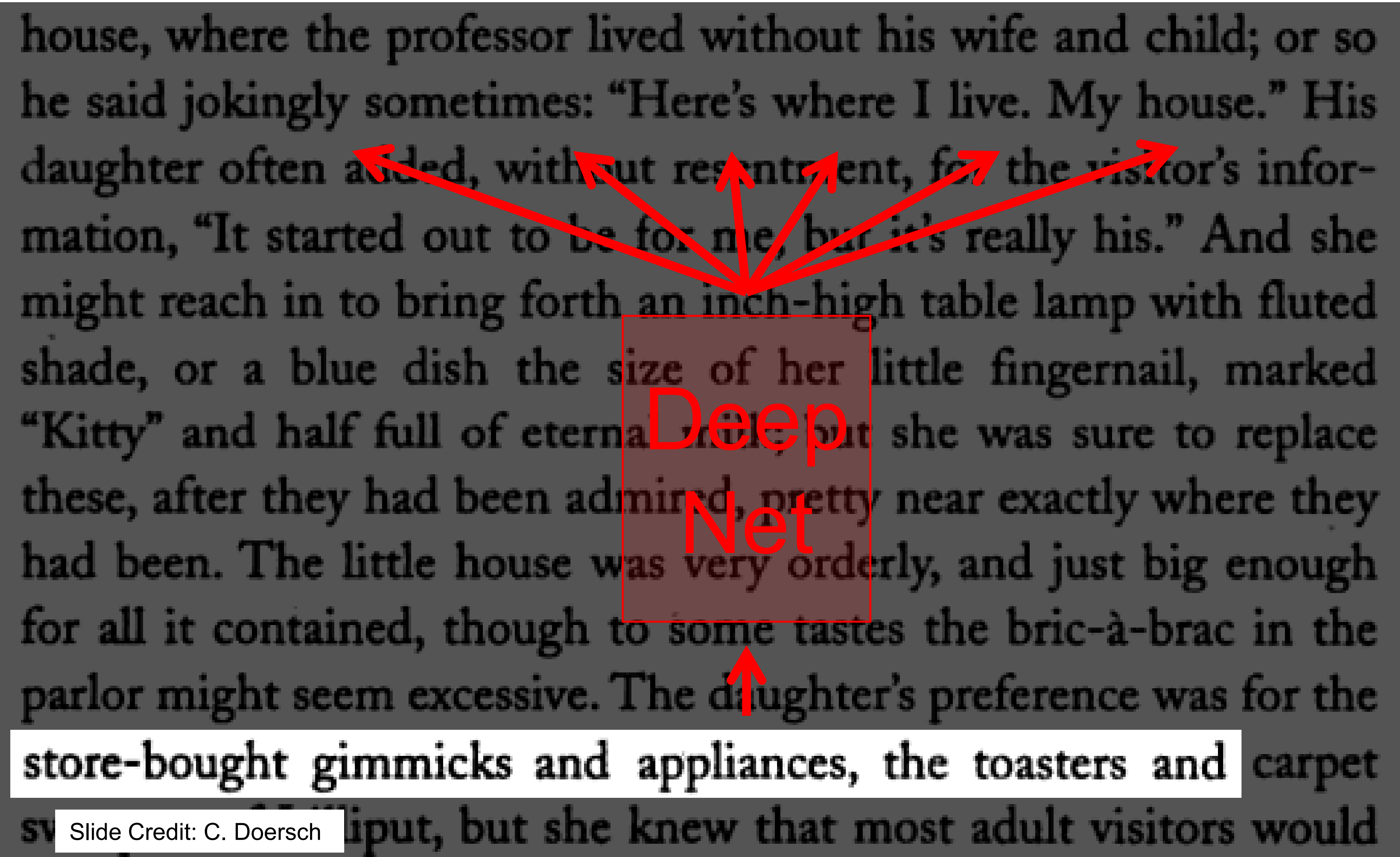
Slide Credit: C. Doersch

To Make It Super Clear

- w = **weights_from_somewhere_else**
- for batch in batches:
  - inputs, labels = batch
  - calculate gradient of loss function with respect to w applied to samples in inputs
  - w += gradient

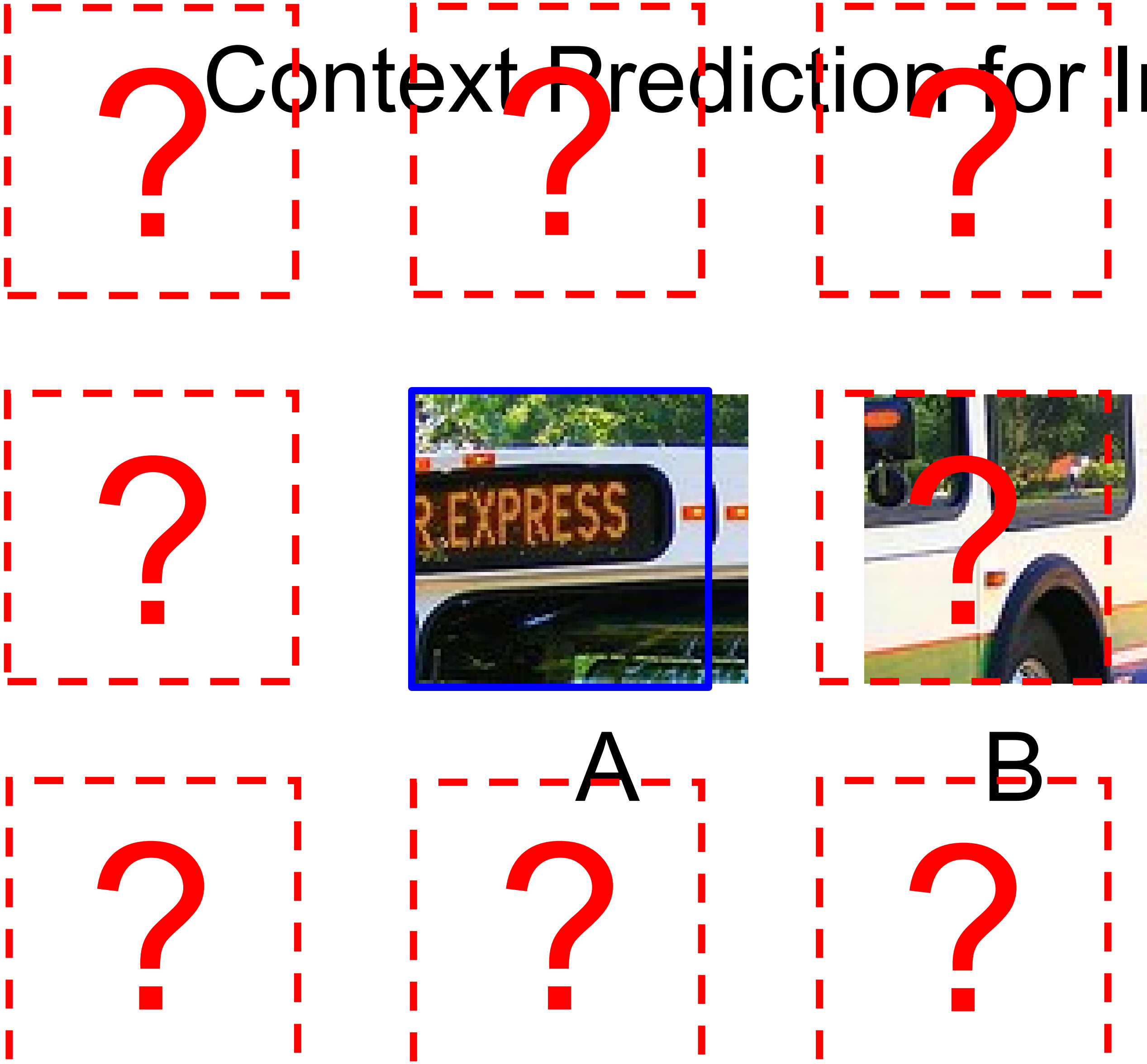
Context as Supervision

[Collobert & Weston 2008; Mikolov et al. 2013]



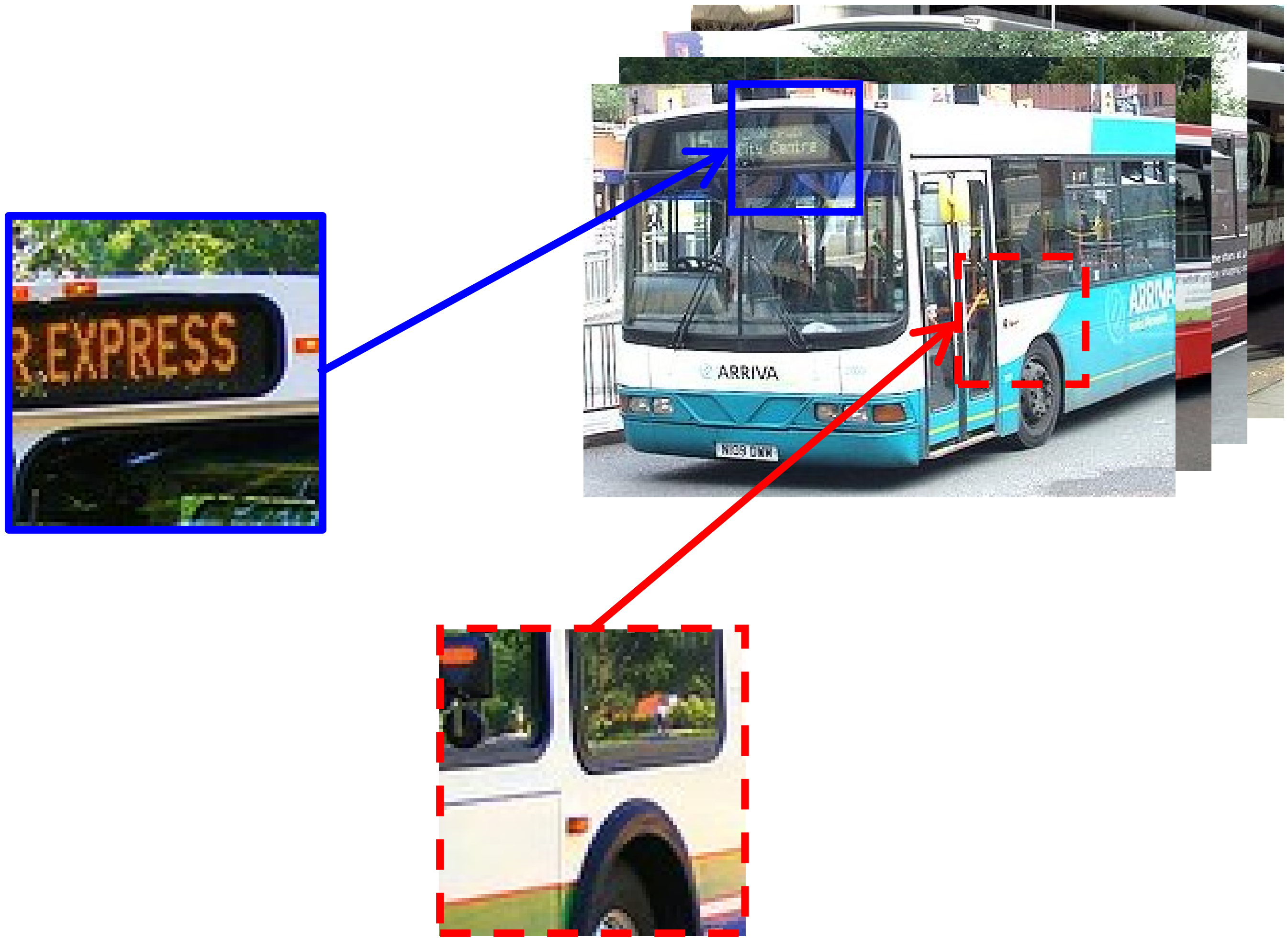
Slide Credit: C. Doersch

Context Prediction for Images



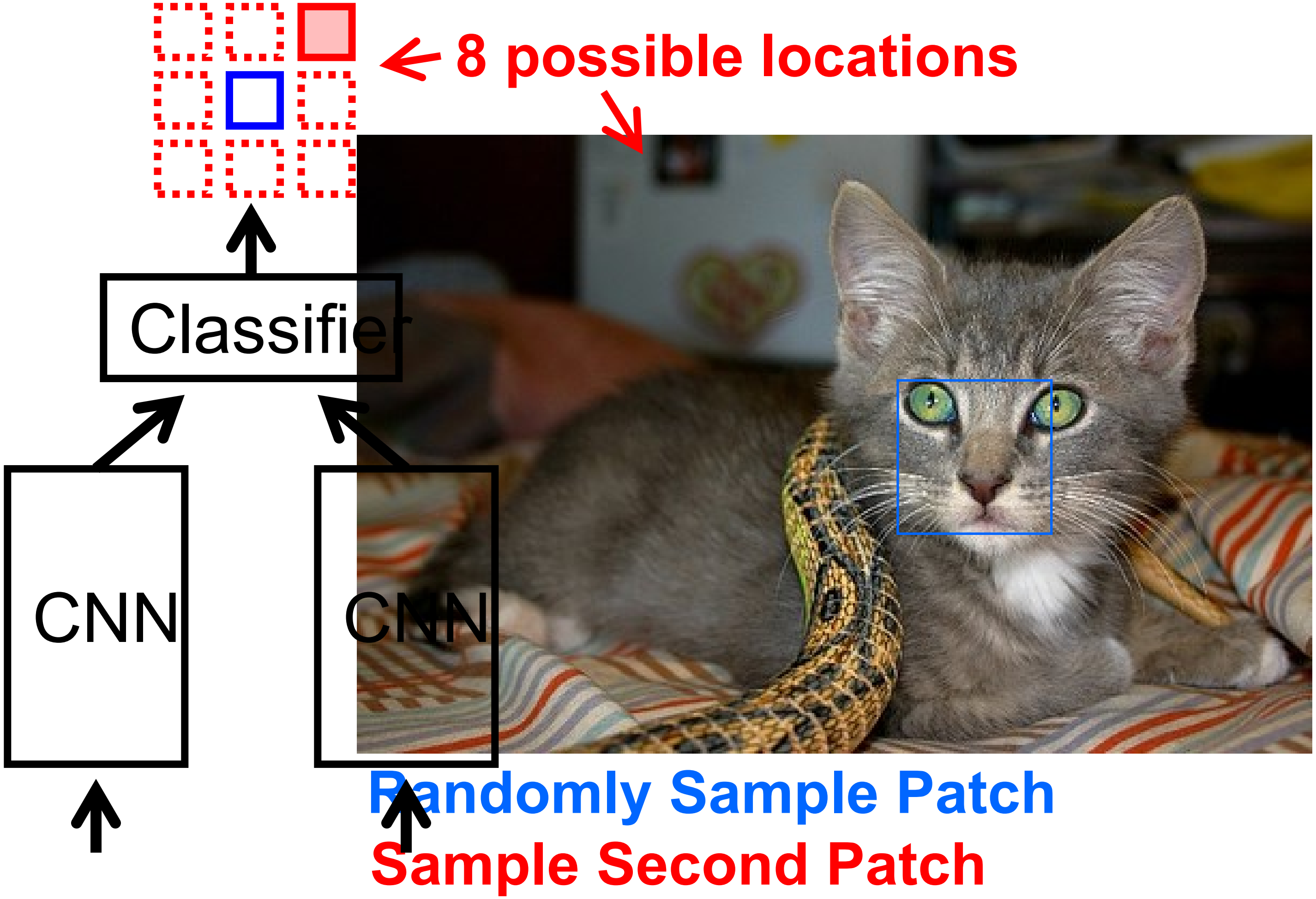
Slide Credit: C. Doersch

Semantics from a non-semantic task

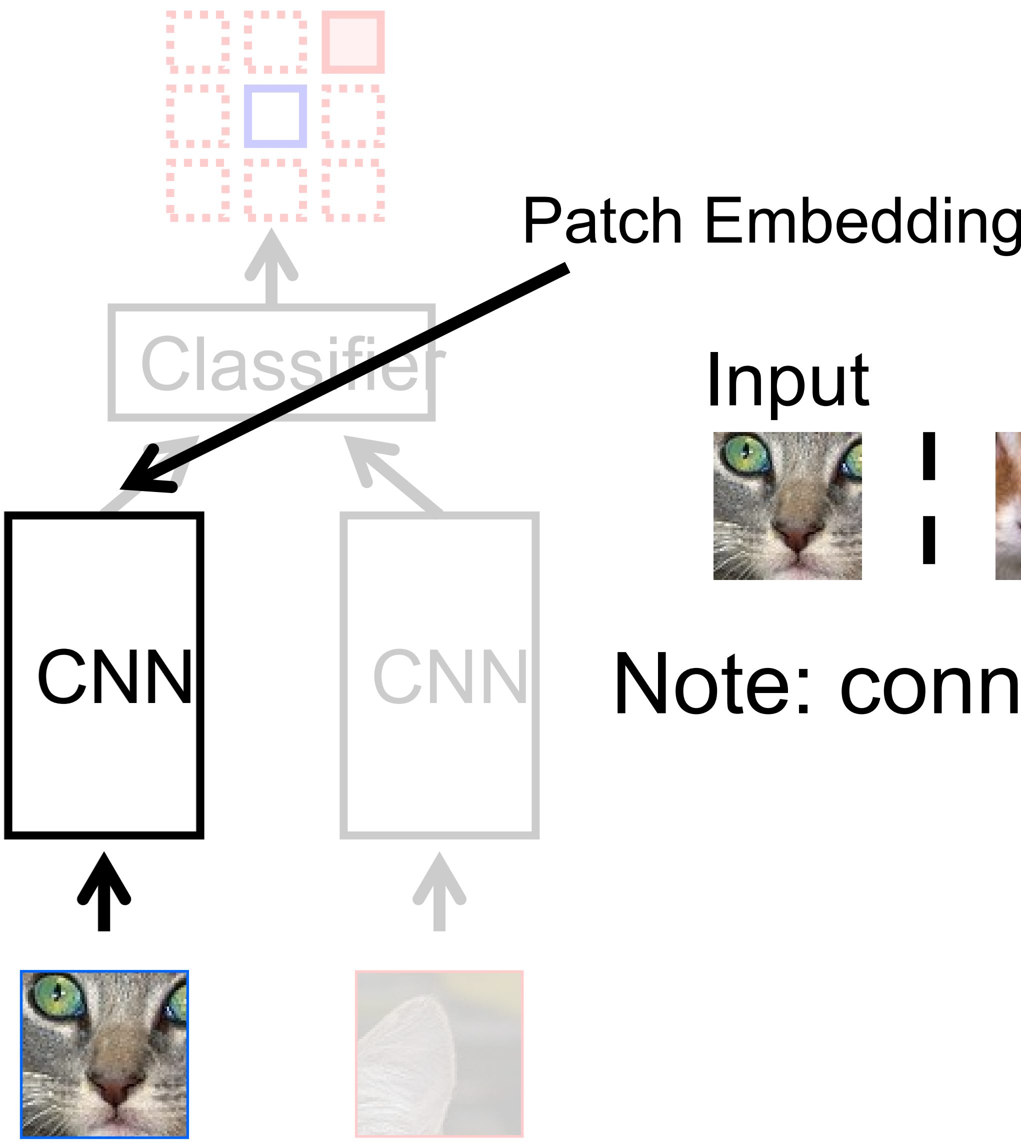


Slide Credit: C. Doersch

Relative Position Task



Slide Credit: C. Doersch

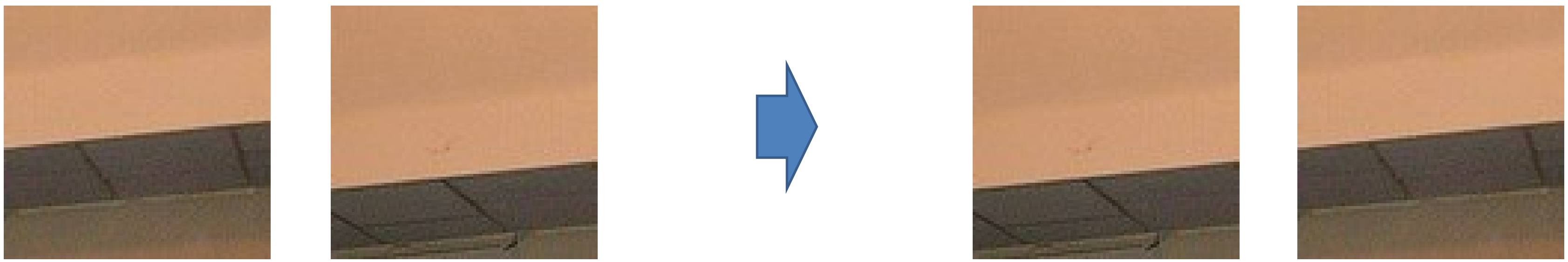


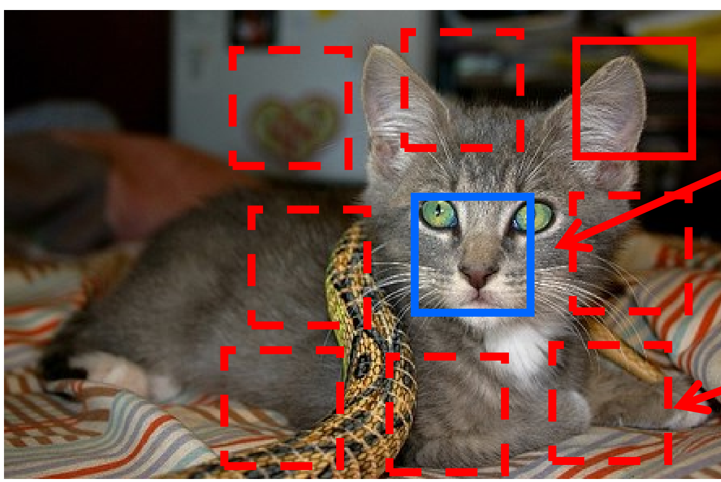
Input Nearest Neighbors

Note: connects ***across*** instances

Slide Credit: C. Doersch

Avoiding Trivial Shortcuts



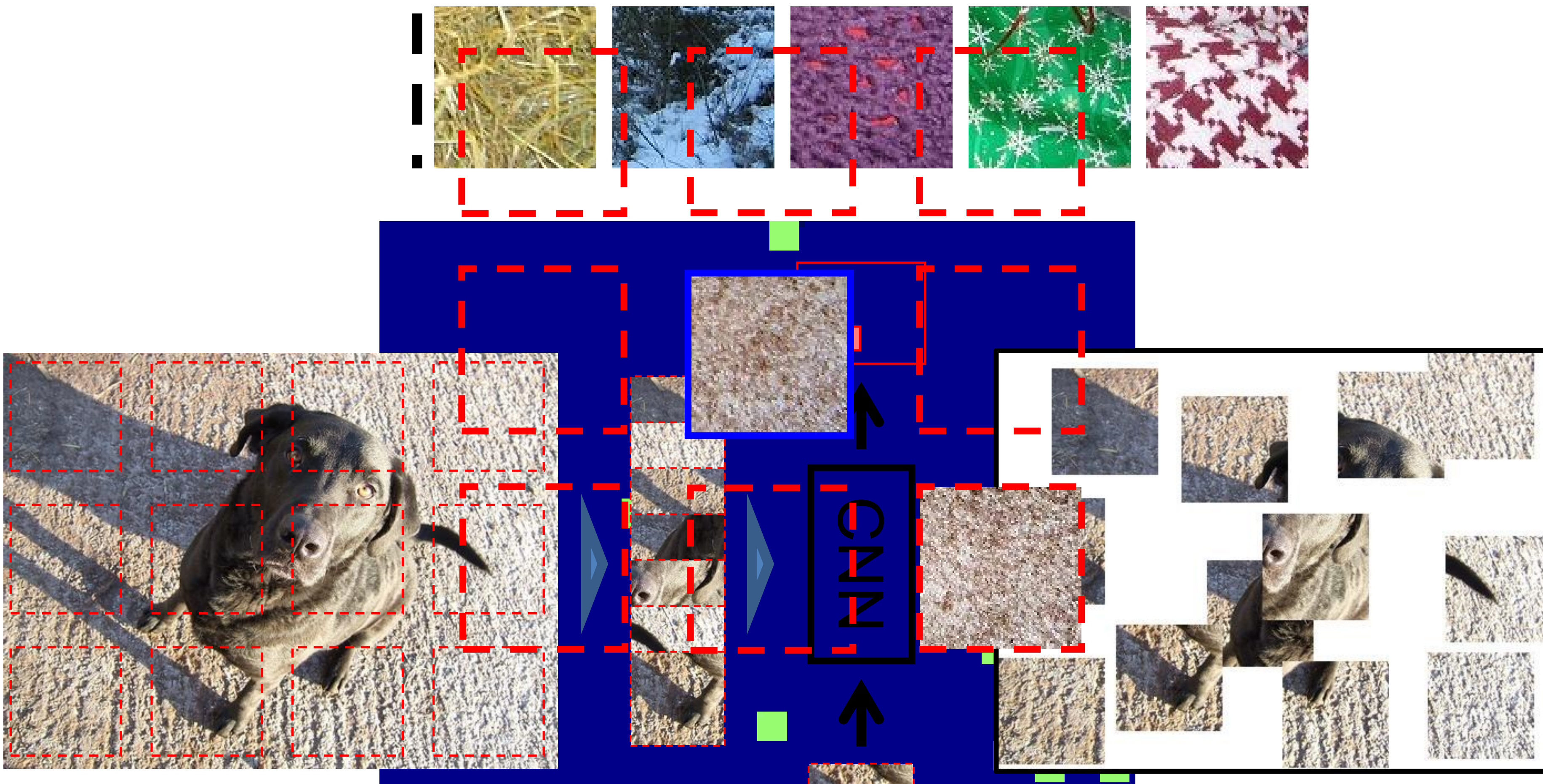


Include a gap

Jitter the patch locations

Slide Credit: C. Doersch

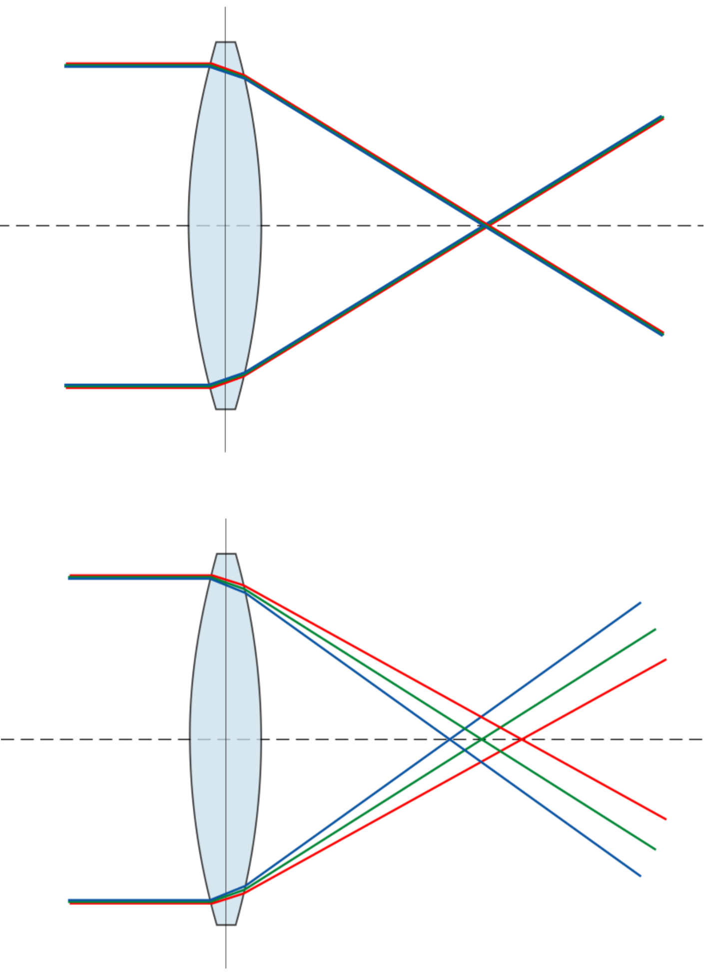
A Not-So “Trivial” Shortcut

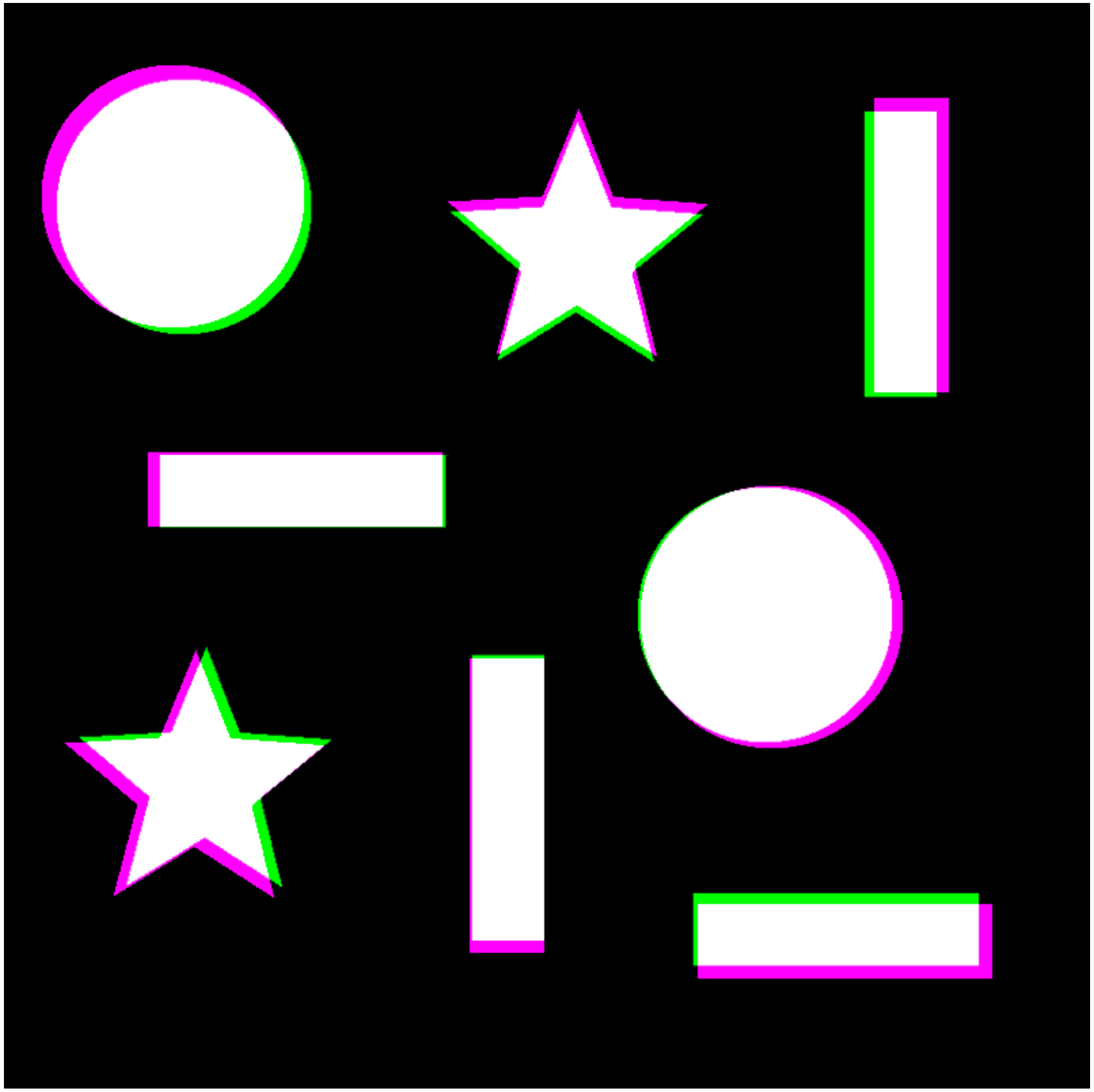


Position i mage

Slide Credit: C. Doersch

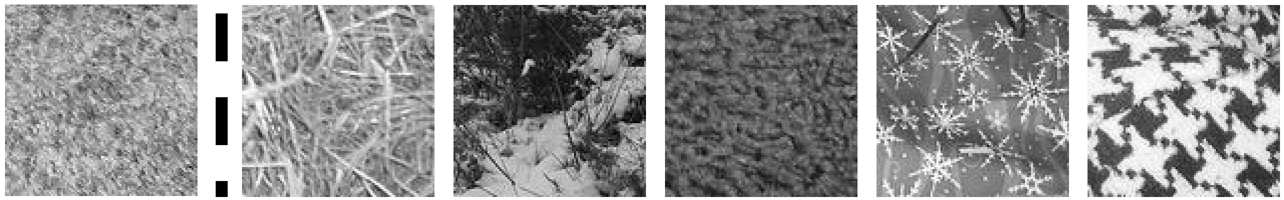
Chromatic Aberration

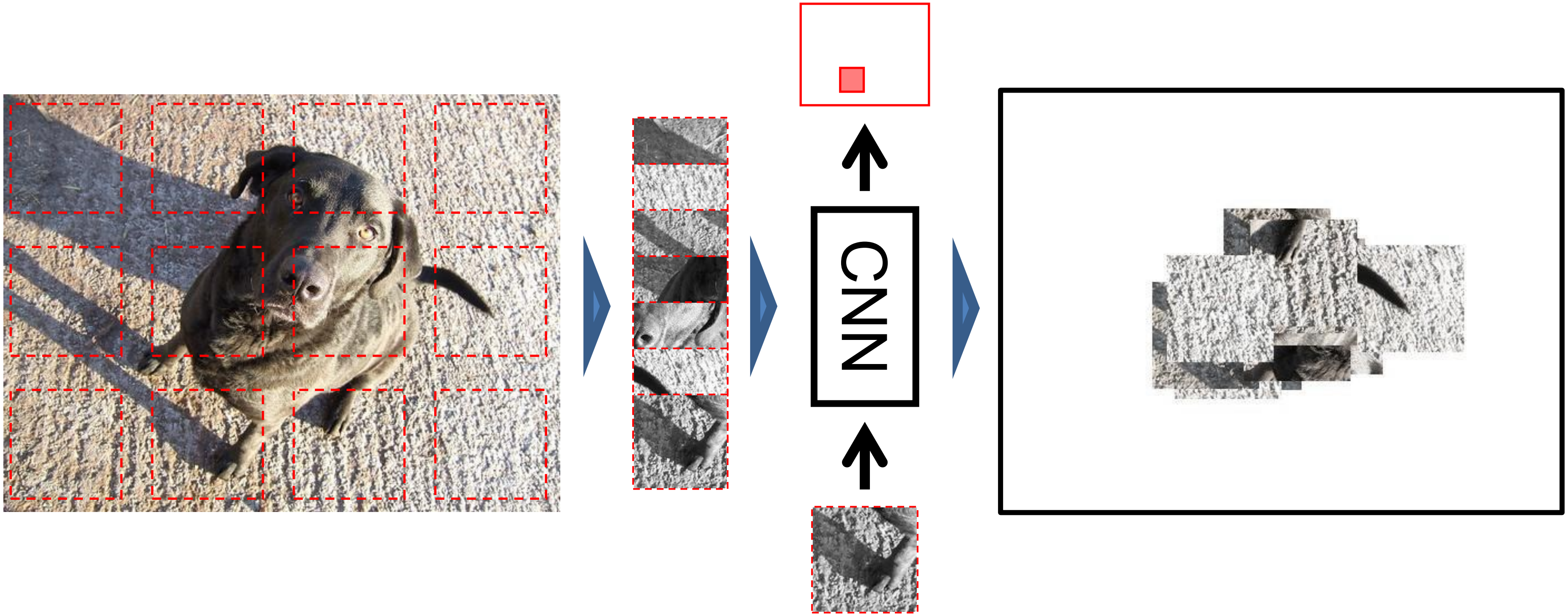




Slide Credit: C. Doersch

Chromatic Aberration

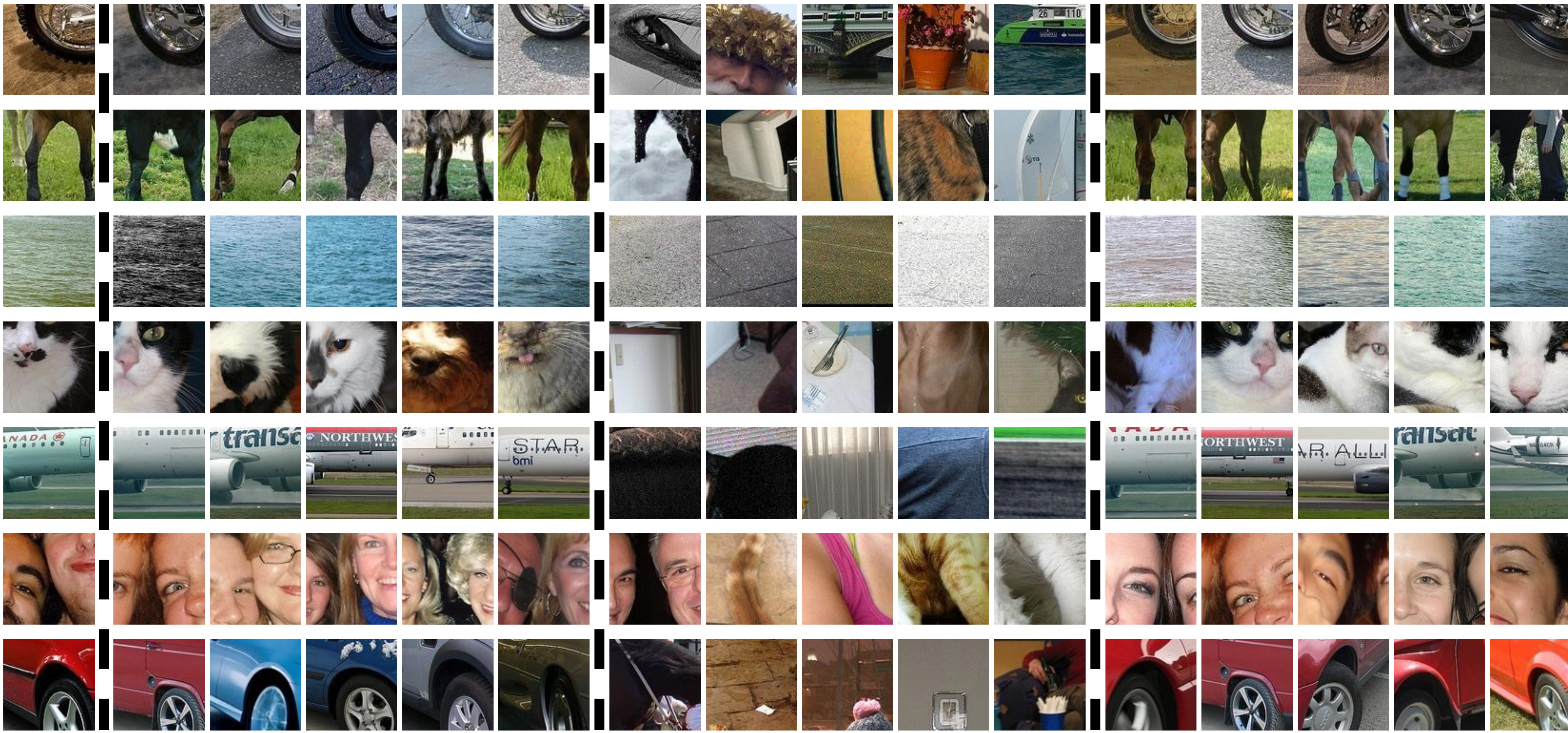




Slide Credit: C. Doersch

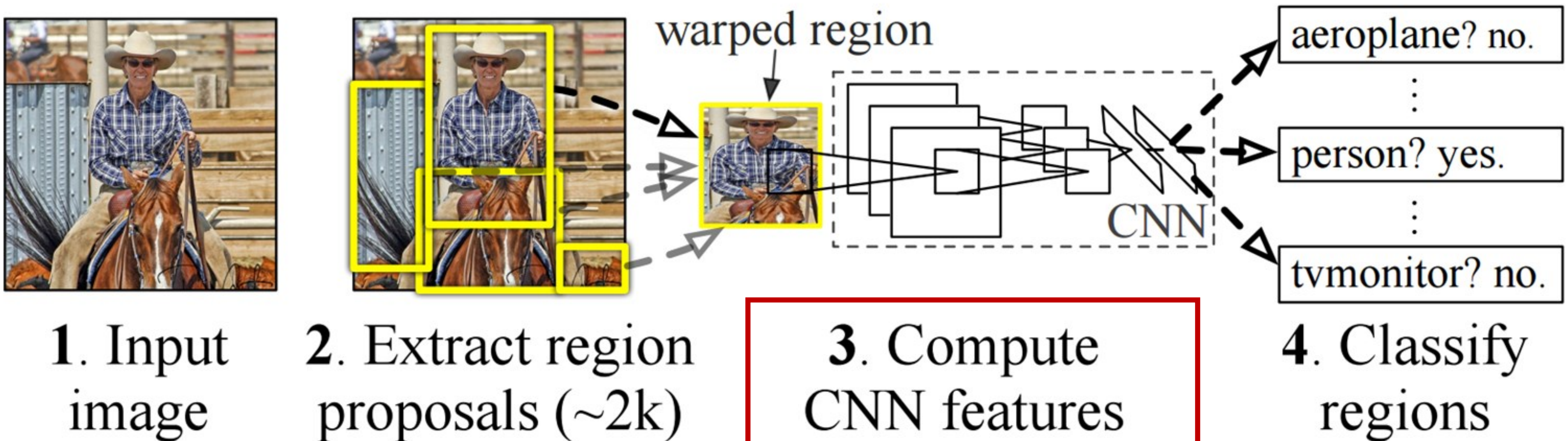
What is learned?

Input Ours Random Initialization ImageNet AlexNet



Slide Credit: C. Doersch

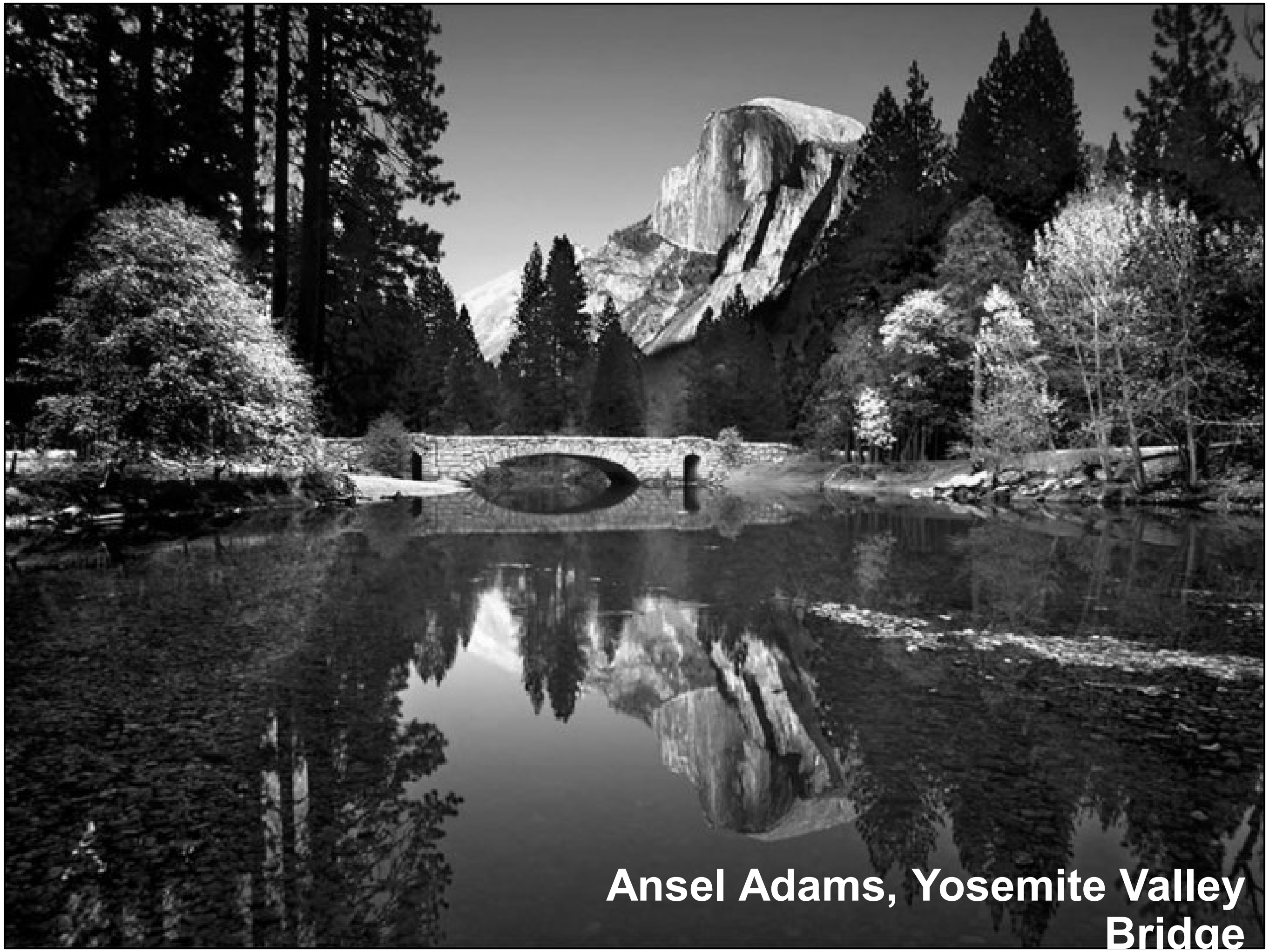
Pre-Training for R-CNN



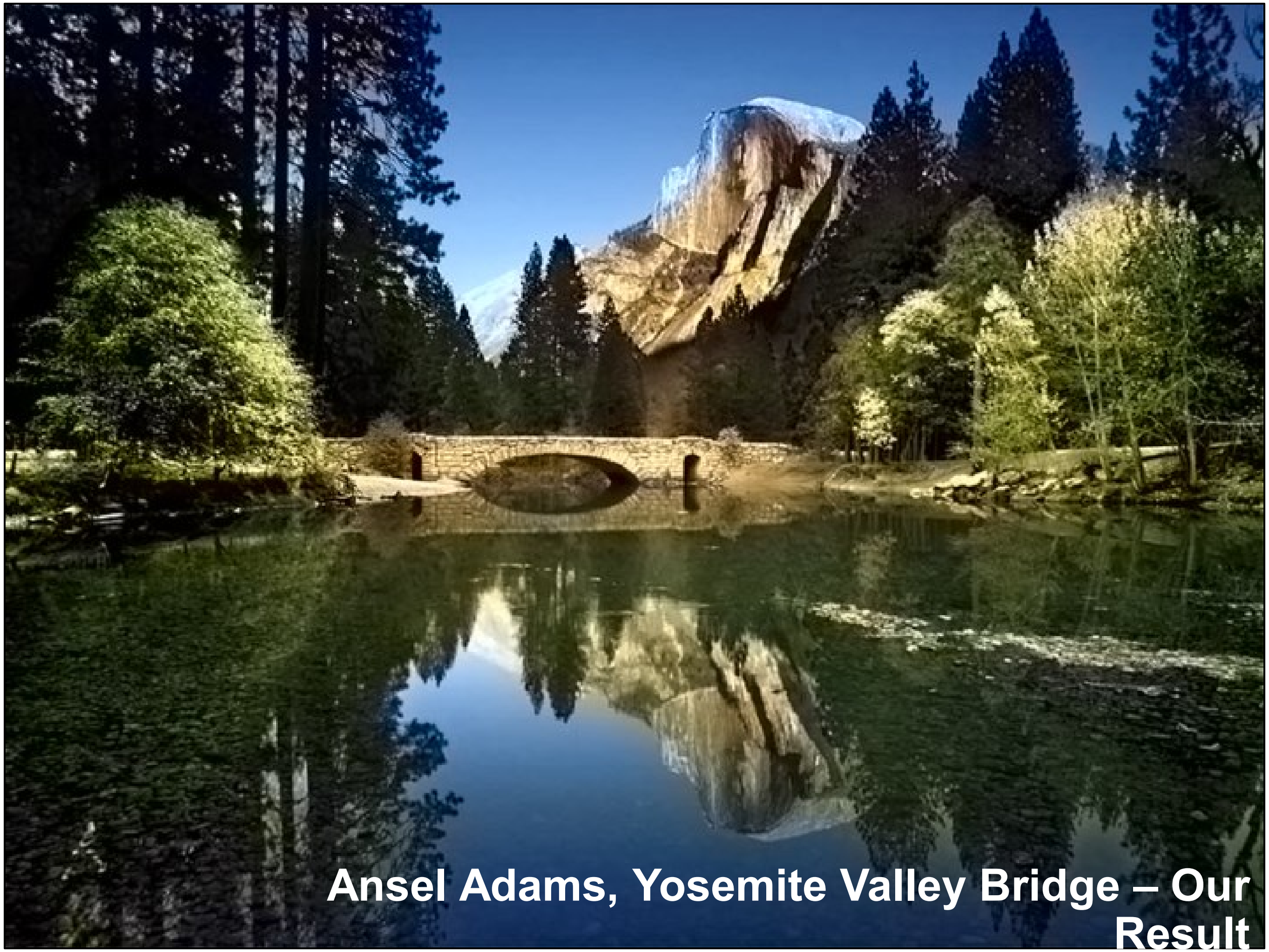
Pre-train on relative-position task, w/o labels

[Girshick et al. 2014]

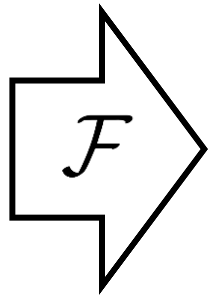
Other Sources Of Signal



Slide Credit: R. Zhang



Slide Credit: R. Zhang

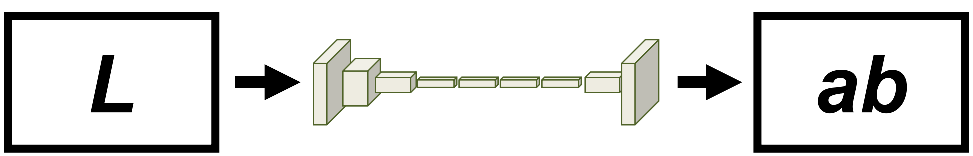


Grayscale image: *L* channel

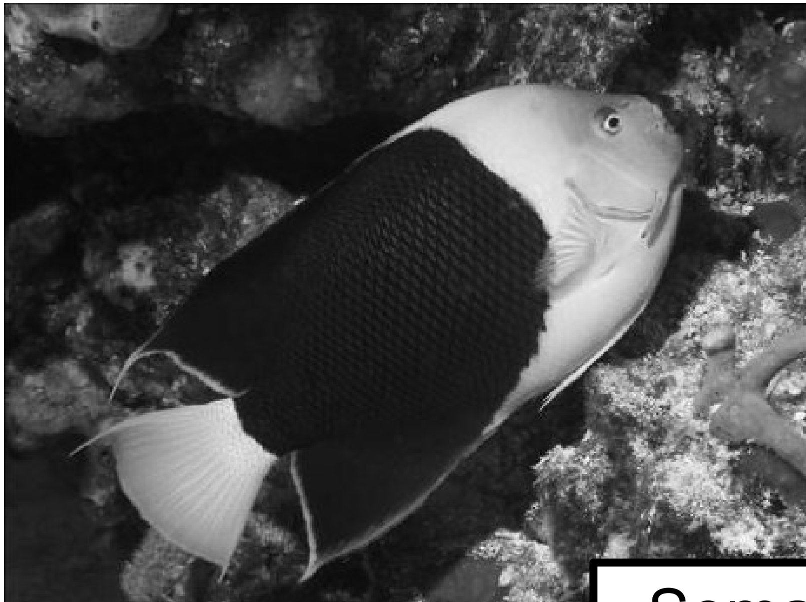
$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$

Color information: *ab* channel

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



Slide Credit: R. Zhang



$\mathcal{F}$

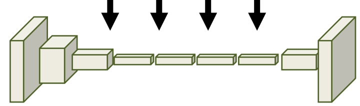
Semantics? Higher-level abstraction?

Grayscale image: *L*

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

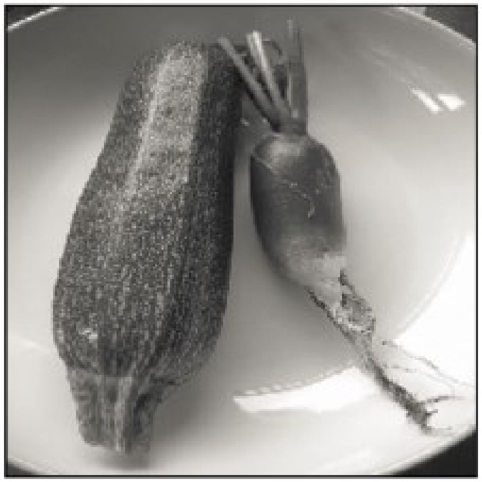
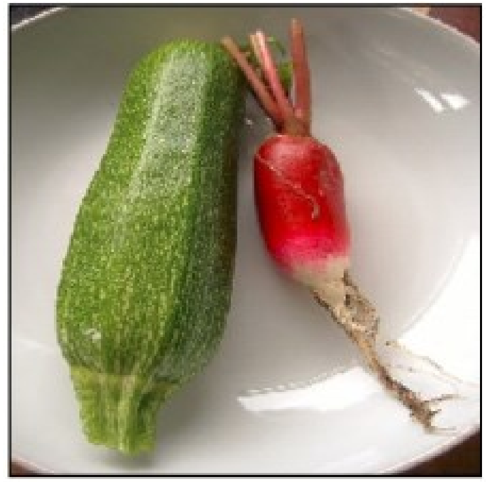
oncatenate (L,ab)

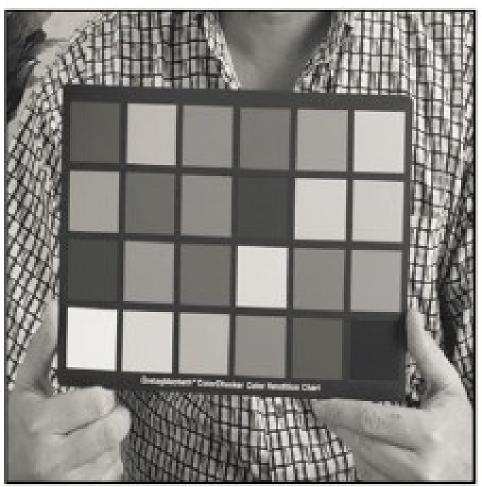
$(\mathbf{X}, \widehat{\mathbf{Y}})$

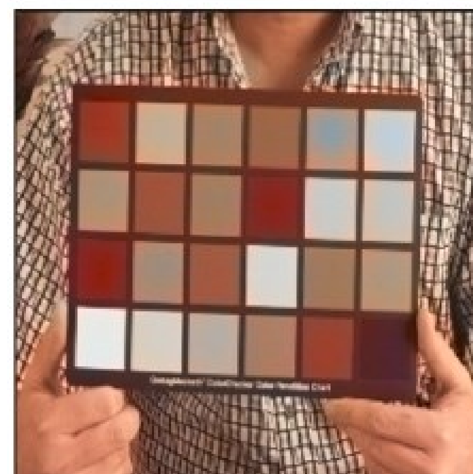
L →  → ***ab*** ← "Free" supervisory signal

Slide Credit: R. Zhang

Input **Ground Truth** **Output**

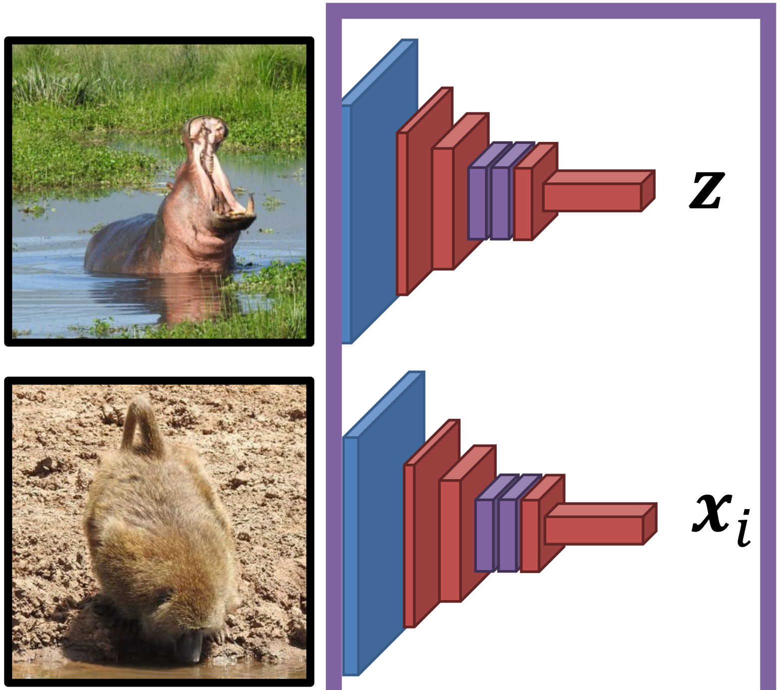




Slide Credit: R. Zhang


Visually Indicated Sounds
Andrew Owens
Phillip Isola
Josh McDermott
Antonio Torralba
Edward Adelson
William Freeman

Contrastive Learning



Weights shared between networks, but details important

Given sample, construct feature z; take a bunch of other images, minimize:

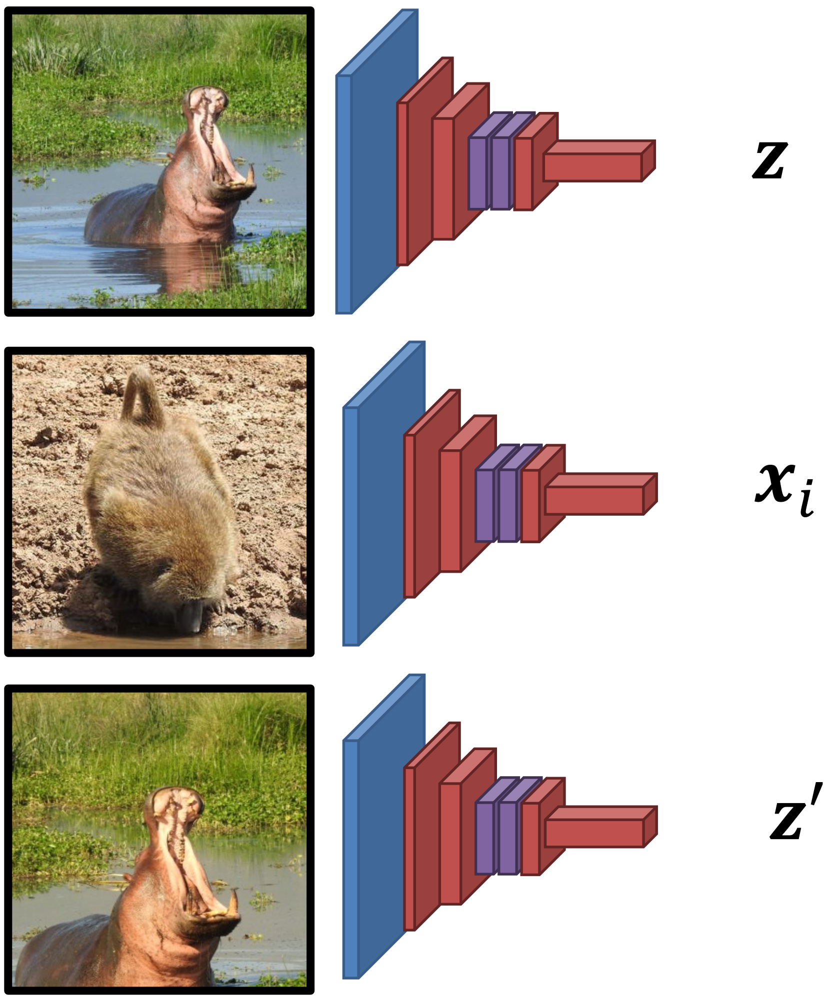
$$\frac{\exp(\boldsymbol{z}^T\boldsymbol{z})}{\exp(\boldsymbol{z}^T\boldsymbol{z} + \sum_i \boldsymbol{z}^T\boldsymbol{x}_i)}$$

Basically, a scoring function with w = z:

$$\frac{\exp(\boldsymbol{w}^T\boldsymbol{z})}{\exp(\boldsymbol{w}^T\boldsymbol{z} + \sum_i \boldsymbol{w}^T\boldsymbol{x}_i)}$$

He et al. Momentum Contrastive Learning, 2019. Wu et al. Instance discrimination 2018

Contrastive Learning



Best performing methods measure distance to augmented sample z':

$$\frac{\exp(z^T z')}{\exp(z^T z' + \sum_i z^T x_i)}$$

Goal: score augmented sample higher than everything else.

He et al. Momentum Contrastive Learning, 2019. Wu et al. Instance discrimination 2018