CS 1674: Intro to Computer Vision

Unsupervised Learning

Prof. Adriana Kovashka
University of Pittsburgh
April 21, 2022

Motivation

- So far we assumed access to plentiful labeled data
- **What if we have limited or no labeled data?**
- Learn from unlabeled data (unsupervised learning)
  - Use structure in data as “labels” (self-supervised learning)
  - Use structure in data to generate similar data (generation)
  - Mine for interesting patterns (discovery)
- Another approach (not discussed): carefully choose which data to label (active learning, human-in-the-loop)

Supervised vs Unsupervised Learning

Supervised Learning

Data: (x, y)
x is data, y is label

Goal: Learn a *function* to map x -> y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

Unsupervised Learning

Data: x
Just data, **no labels!**

Goal: Learn some underlying hidden *structure* of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

Plan for this last lecture

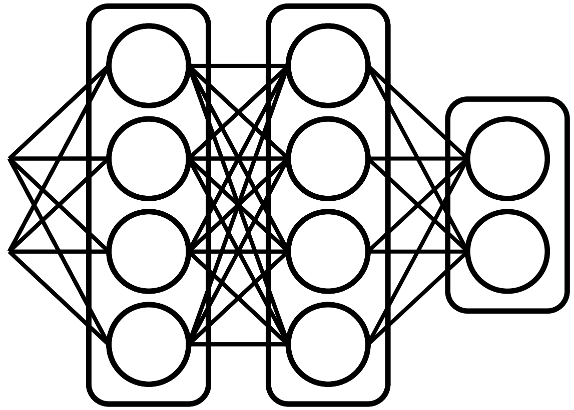
- Self-supervised learning
  - For images
  - For video
- Visual discovery
  - Discovering style-specific elements
- Generation not recognition
  - Theory/technique
  - Applications

Unsupervised Visual Representation Learning by Context Prediction

Carl Doersch, Alexei Efros and Abhinav Gupta

ICCV 2015

ImageNet + Deep Learning

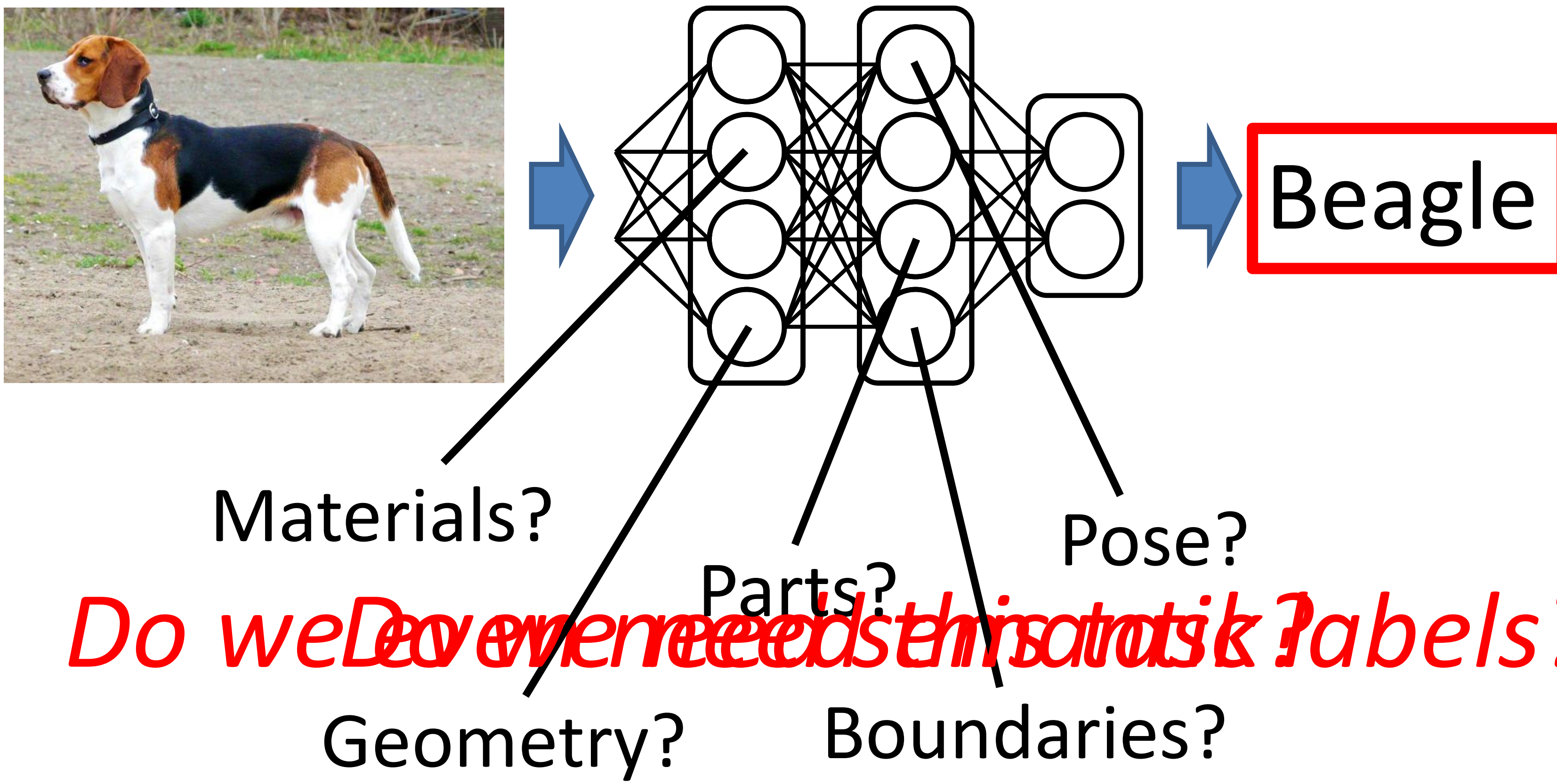


Beagle

- Image Retrieval
- Detection (RCNN)
- Segmentation (FCN)
- Depth Estimation
- ...

Doersch et al., “Unsupervised Visual Representation Learning by Context Prediction”, ICCV 2015

ImageNet + Deep Learning



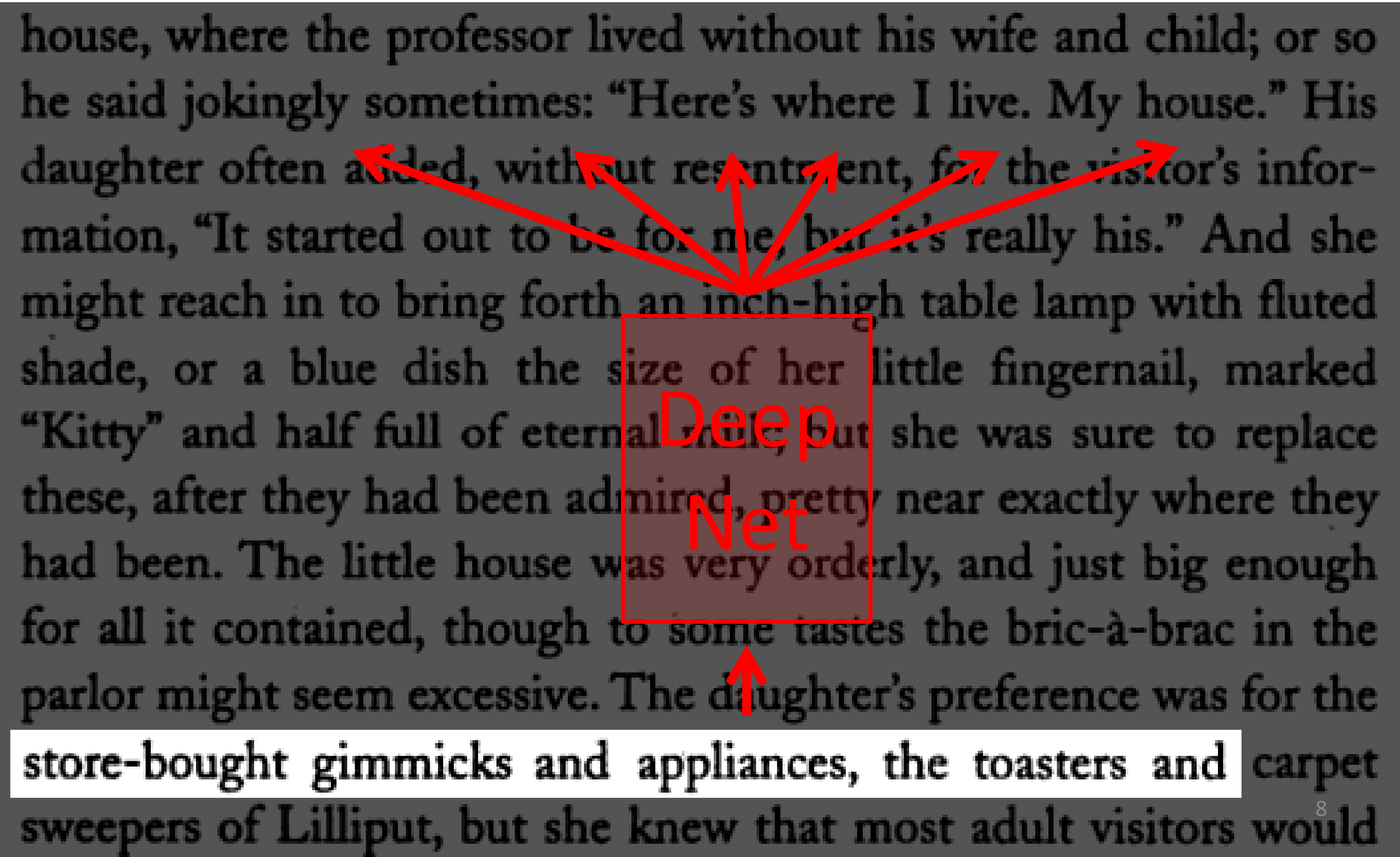
Do we need semantic labels?

Do we even need this task?

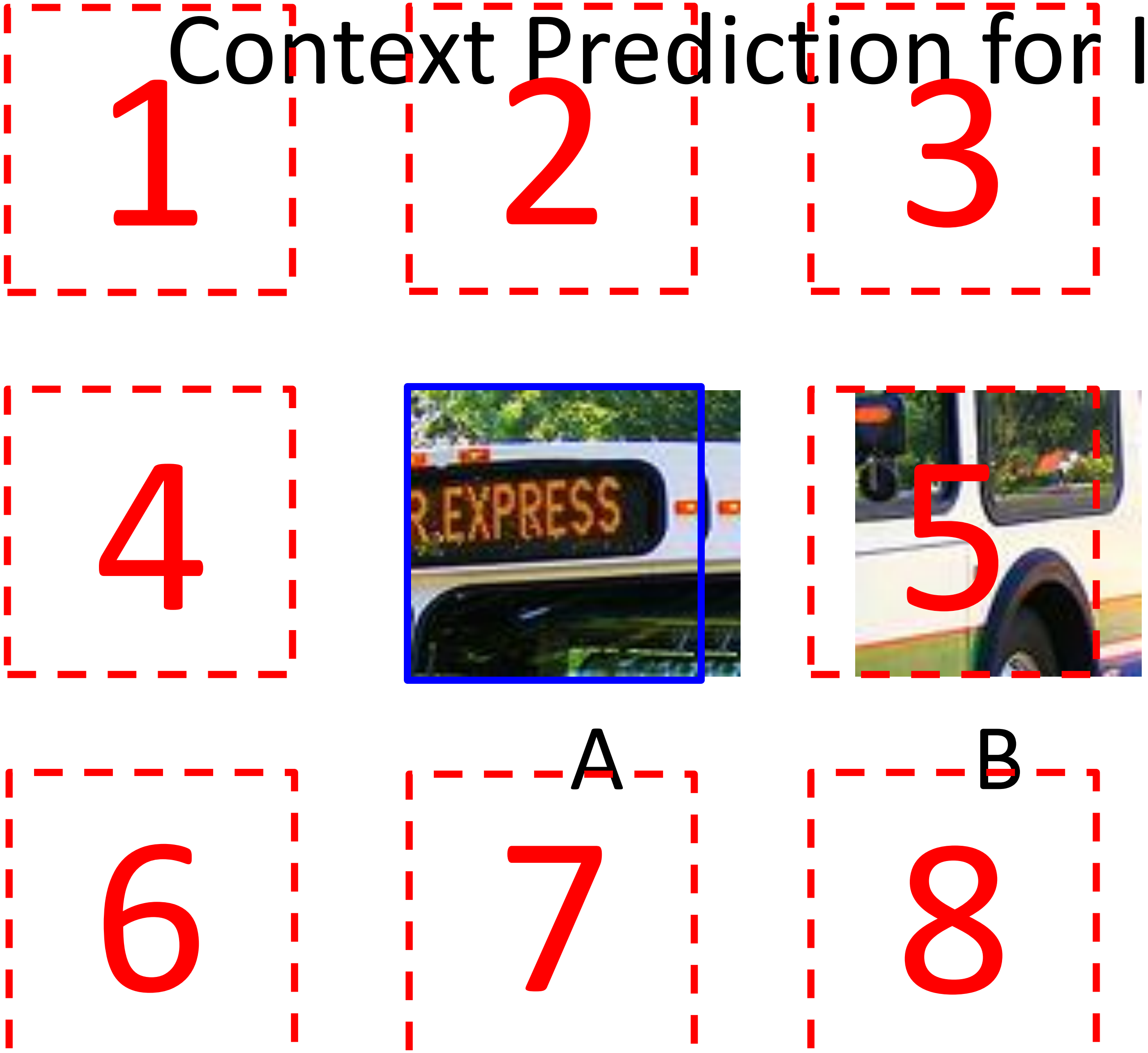
Doersch et al., “Unsupervised Visual Representation Learning by Context Prediction”, ICCV 2015

Context as Supervision

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

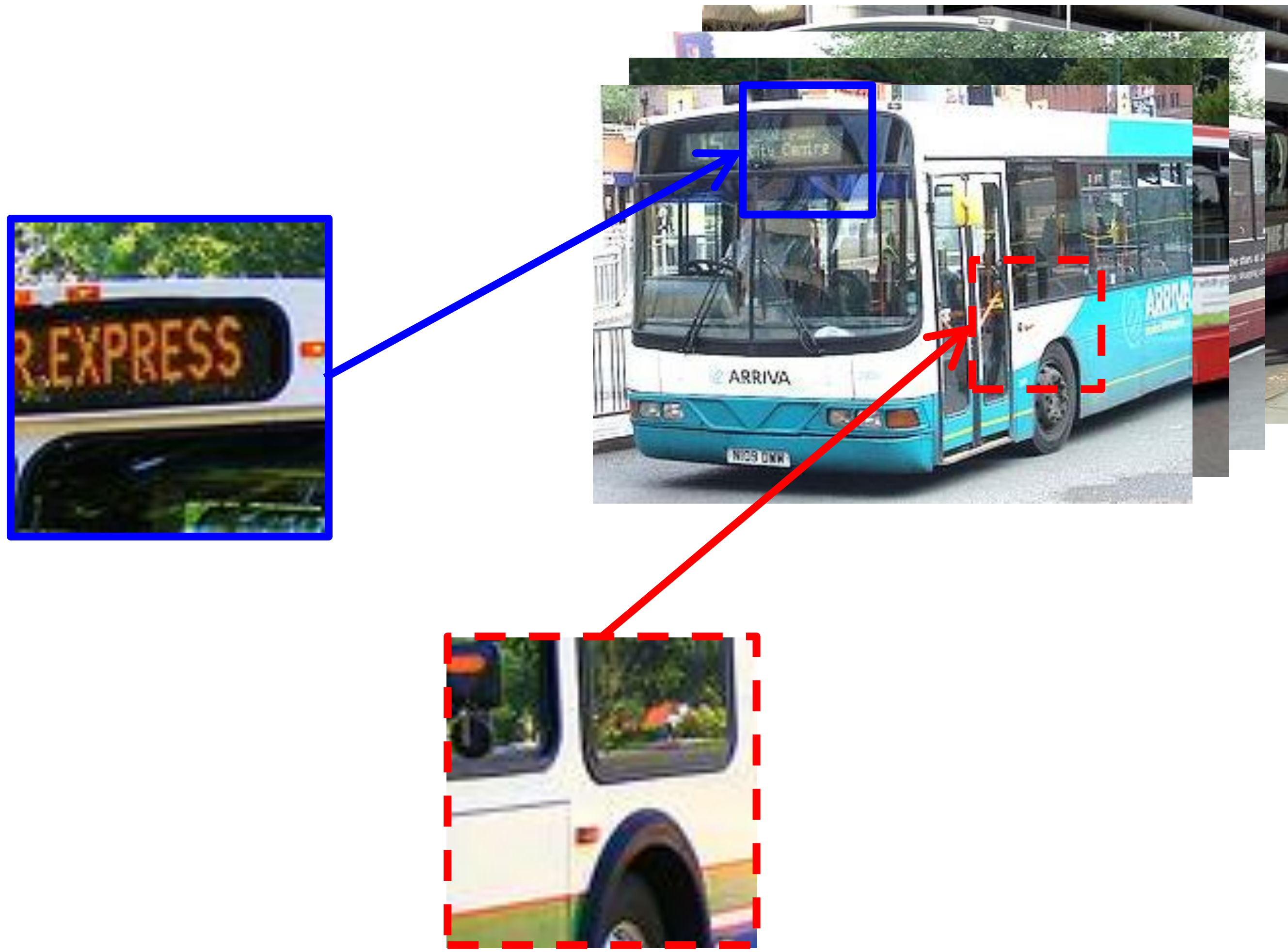


Context Prediction for Images



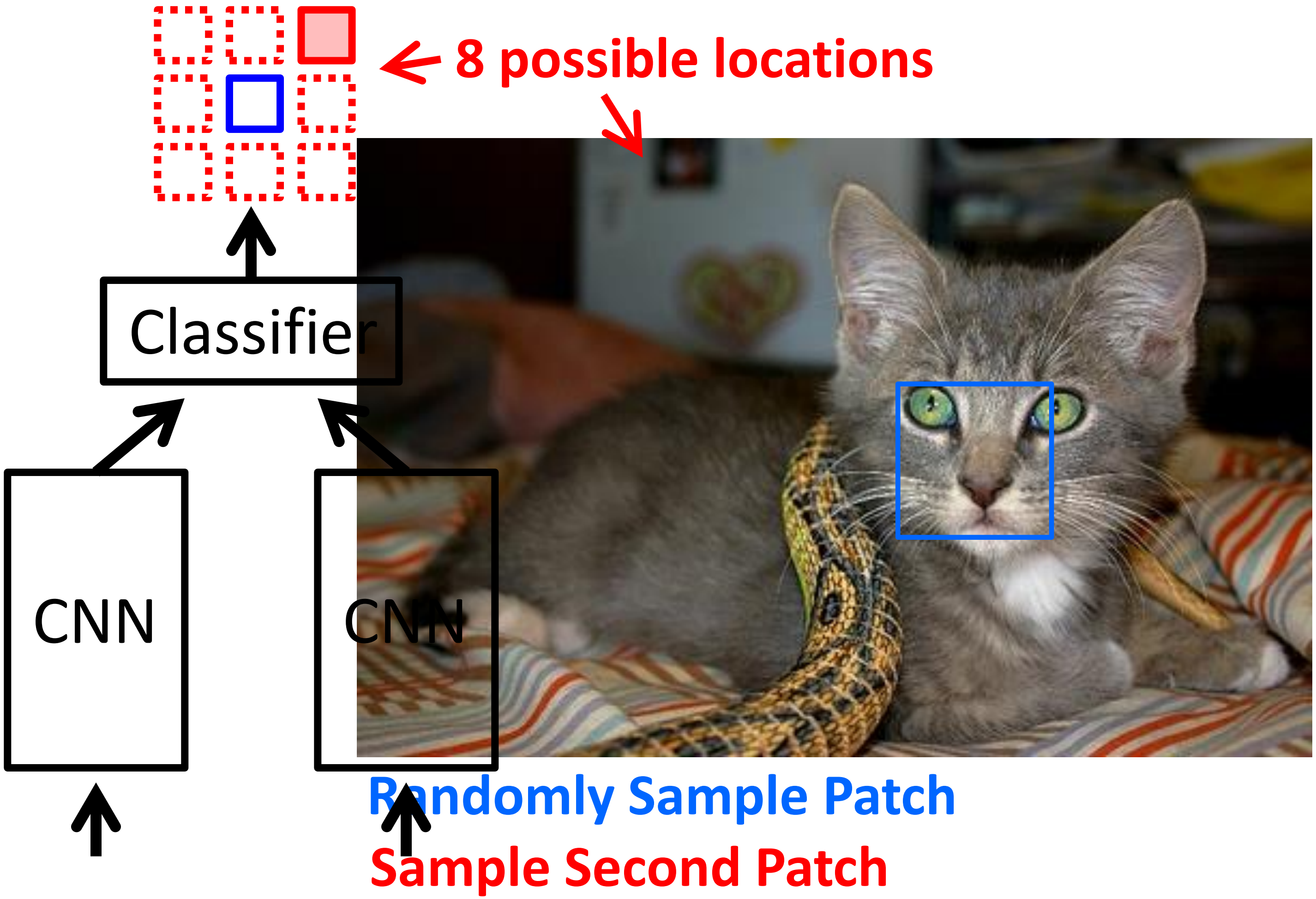
Doersch et al., “Unsupervised Visual Representation Learning by Context Prediction”, ICCV 2015

Semantics from a non-semantic task

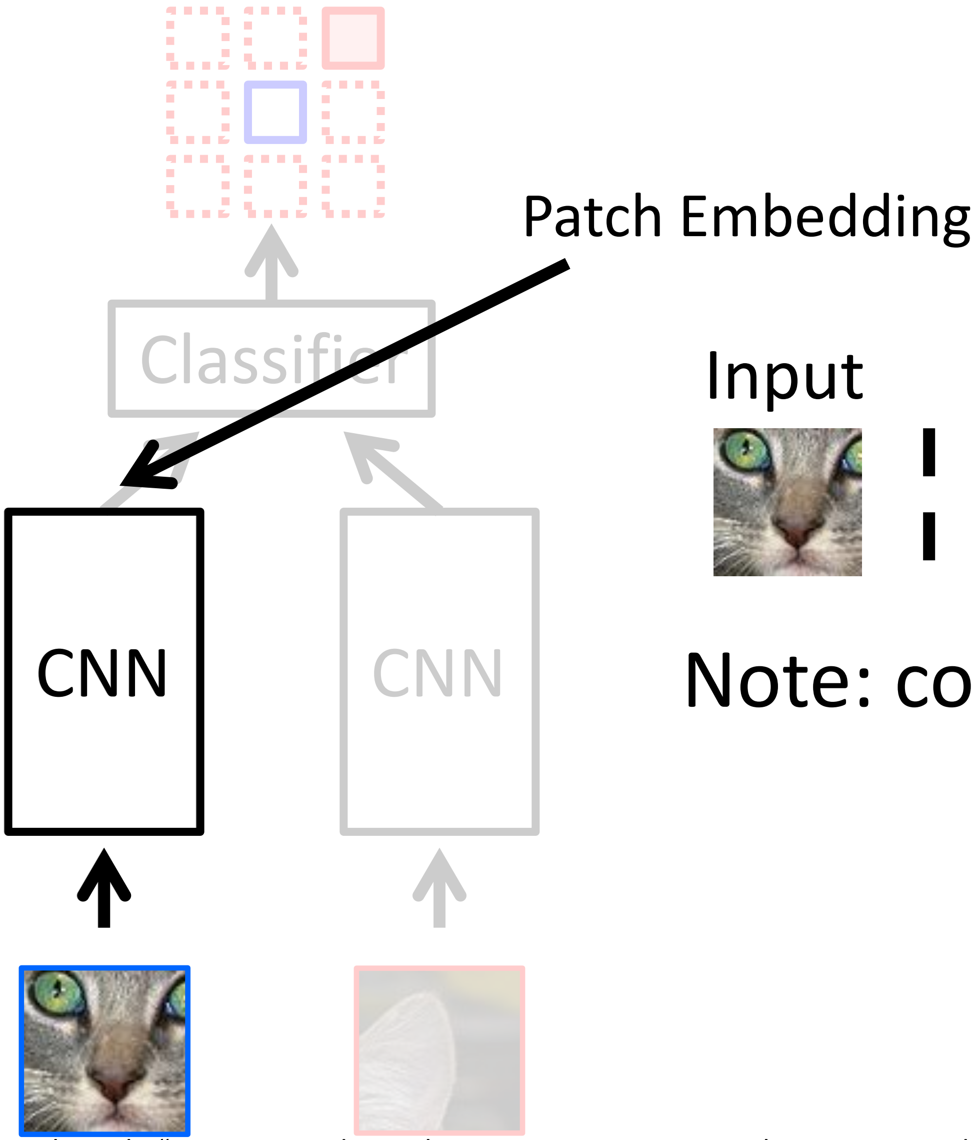


Doersch et al., “Unsupervised Visual Representation Learning by Context Prediction”, ICCV 2015

Relative Position Task



Doersch et al., “Unsupervised Visual Representation Learning by Context Prediction”, ICCV 2015

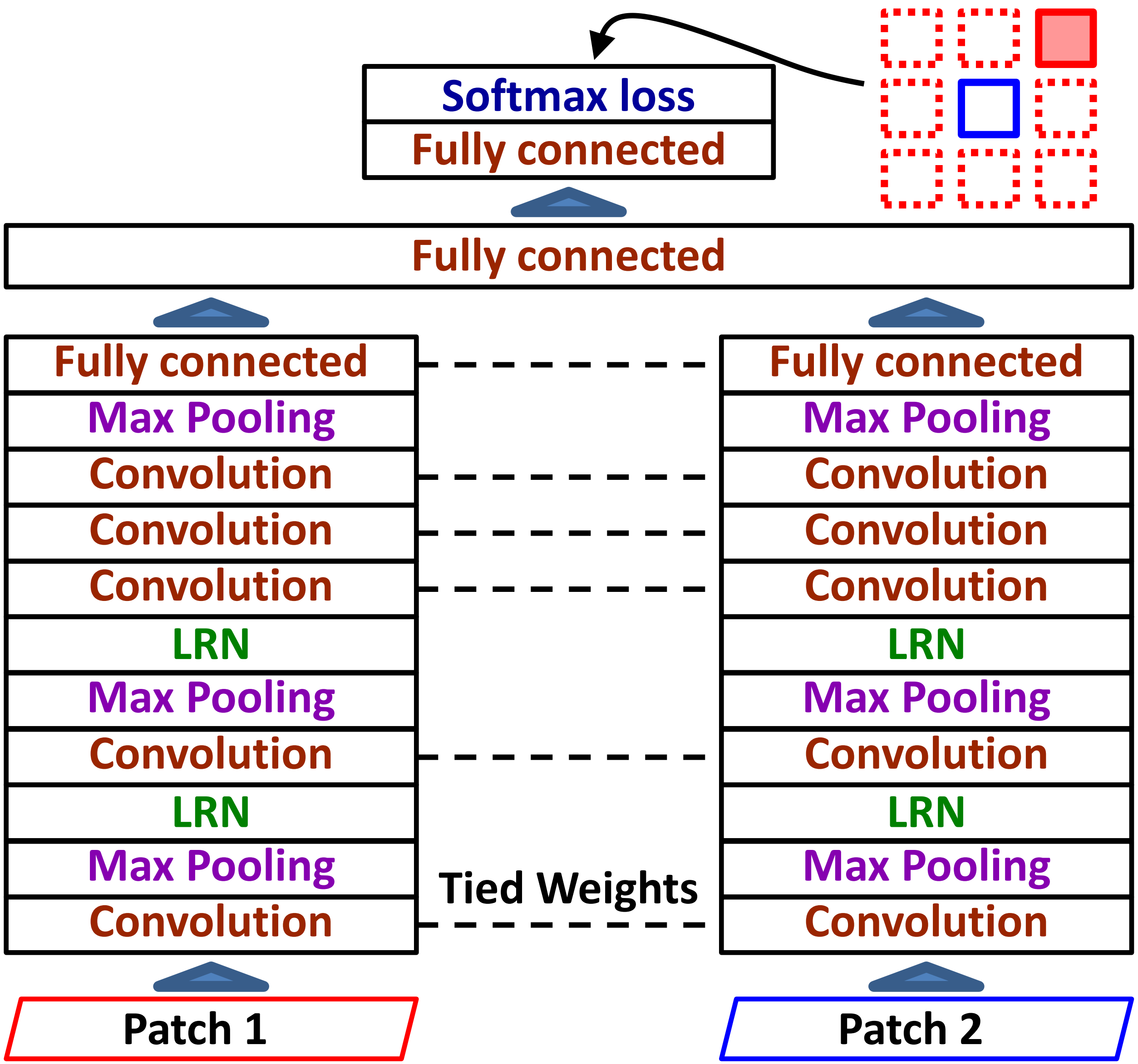


Input Nearest Neighbors

Note: connects ***across*** instances!

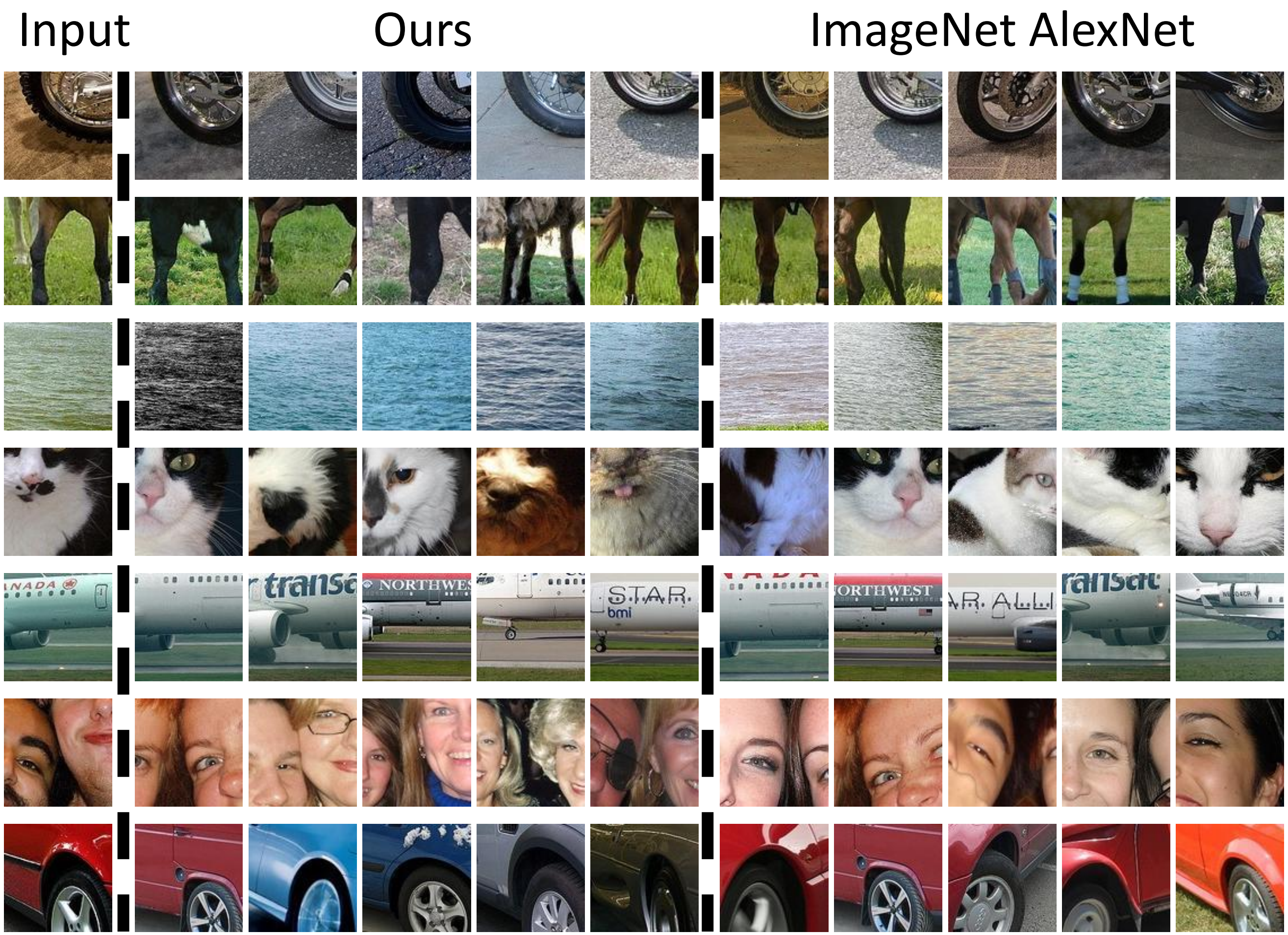
Doersch et al., "Unsupervised Visual Representation Learning by Context Prediction", ICCV 2015

Architecture



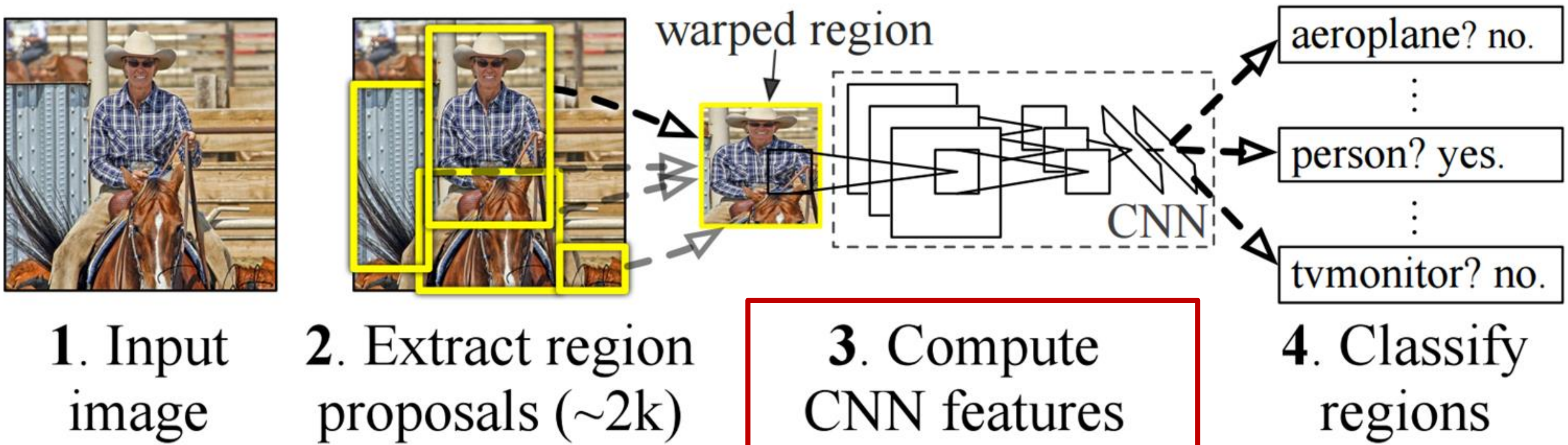
Doersch et al., “Unsupervised Visual Representation Learning by Context Prediction”, ICCV 2015

What is learned?



Doersch et al., “Unsupervised Visual Representation Learning by Context Prediction”, ICCV 2015

Pre-Training for R-CNN



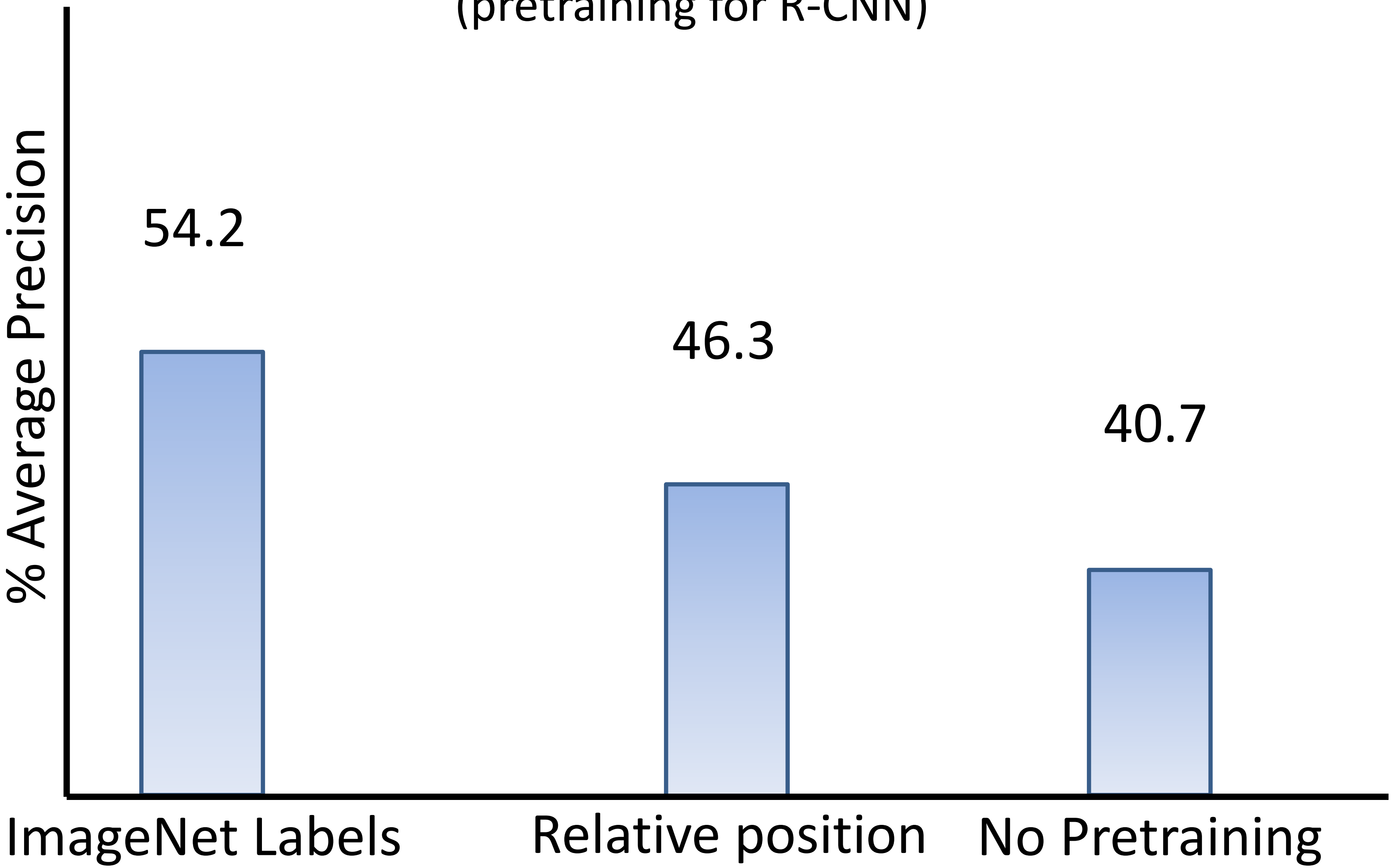
Pre-train on relative-position task, w/o labels

Doersch et al., “Unsupervised Visual Representation Learning by Context Prediction”, ICCV 2015

[Girshick et al. 2014]

VOC 2007 Performance

(pretraining for R-CNN)



Doersch et al., “Unsupervised Visual Representation Learning by Context Prediction”, ICCV 2015

Shuffle and Learn: Unsupervised Learning using Temporal Order Verification

Ishan Misra, C. Lawrence Zitnick, and Martial Hebert

ECCV 2016

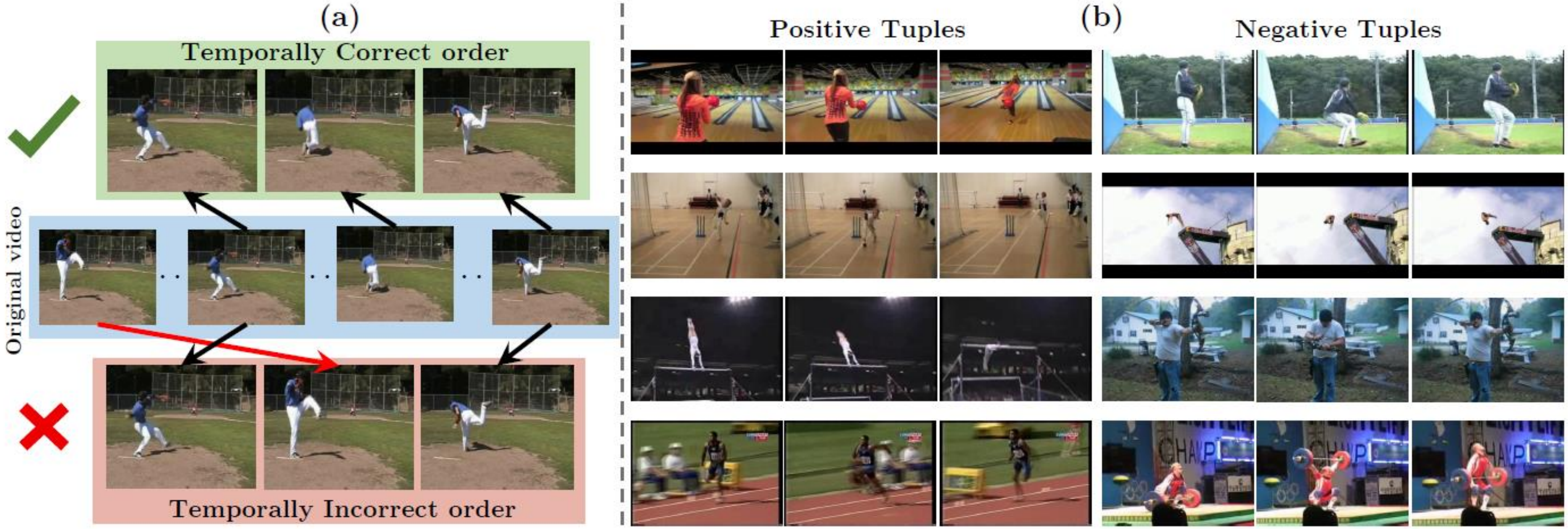


Fig. 1: **(a)** A video imposes a natural temporal structure for visual data. In many cases, one can easily verify whether frames are in the correct temporal order (shuffled or not). Such a simple sequential verification task captures important spatiotemporal signals in videos. We use this task for unsupervised pre-training of a Convolutional Neural Network (CNN). **(b)** Some examples of the automatically extracted positive and negative tuples used to formulate a classification task for a CNN.

Misra et al., "Shuffle and Learn: Unsupervised Learning using Temporal Order Verification", ECCV 2016

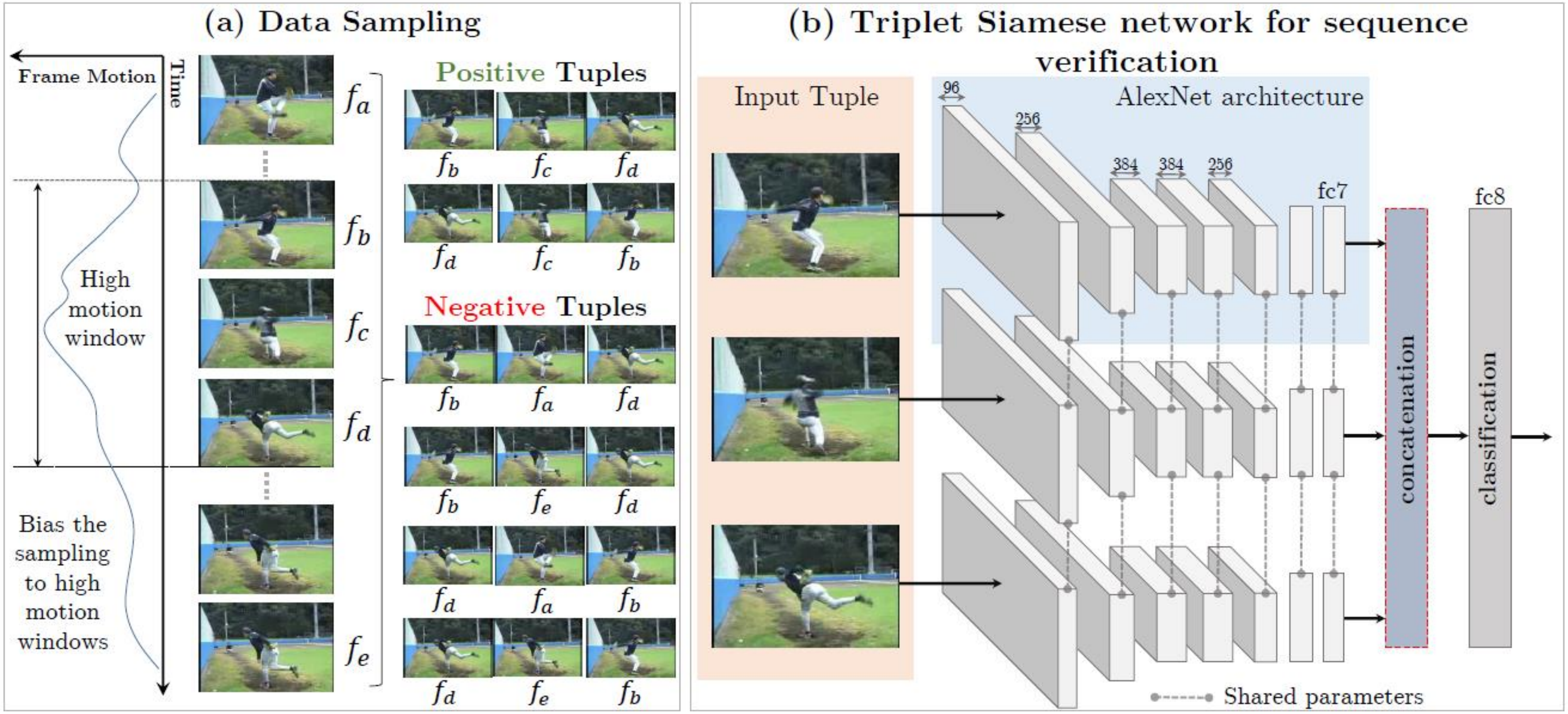


Fig. 2: **(a)** We sample tuples of frames from high motion windows in a video. We form positive and negative tuples based on whether the three input frames are in the correct temporal order. **(b)** Our triplet Siamese network architecture has three parallel network stacks with shared weights upto the `fc7` layer. Each stack takes a frame as input, and produces a representation at the `fc7` layer. The concatenated `fc7` representations are used to predict whether the input tuple is in the correct temporal order.

Misra et al., "Shuffle and Learn: Unsupervised Learning using Temporal Order Verification", ECCV 2016

Benefit of unsupervised but in-domain training

Table 2: Mean classification accuracies over the 3 splits of UCF101 and HMDB51 datasets. We compare different initializations and finetune them for action recognition.

Dataset	Initialization	Mean Accuracy
UCF101	Random	38.6
	(Ours) Tuple verification	**50.2**
HMDB51	Random	13.3
	UCF Supervised	15.2
	(Ours) Tuple verification	**18.1**

Misra et al., “Shuffle and Learn: Unsupervised Learning using Temporal Order Verification”, ECCV 2016

A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen, Simon Kornblith, Mohammad Norouzi, Geoffrey Hinton

ICML 2020

https://github.com/google-research/simclr

Plan for this last lecture

- Self-supervised learning
  – For images
  – For video
- Visual discovery
  – Discovering style-specific elements
- Generation not recognition
  – Theory/technique
  – Applications

What Makes Paris Look like Paris?

Carl Doersch, Saurabh Singh, Abhinav Gupta, Josef Sivic,
Alexei Efros

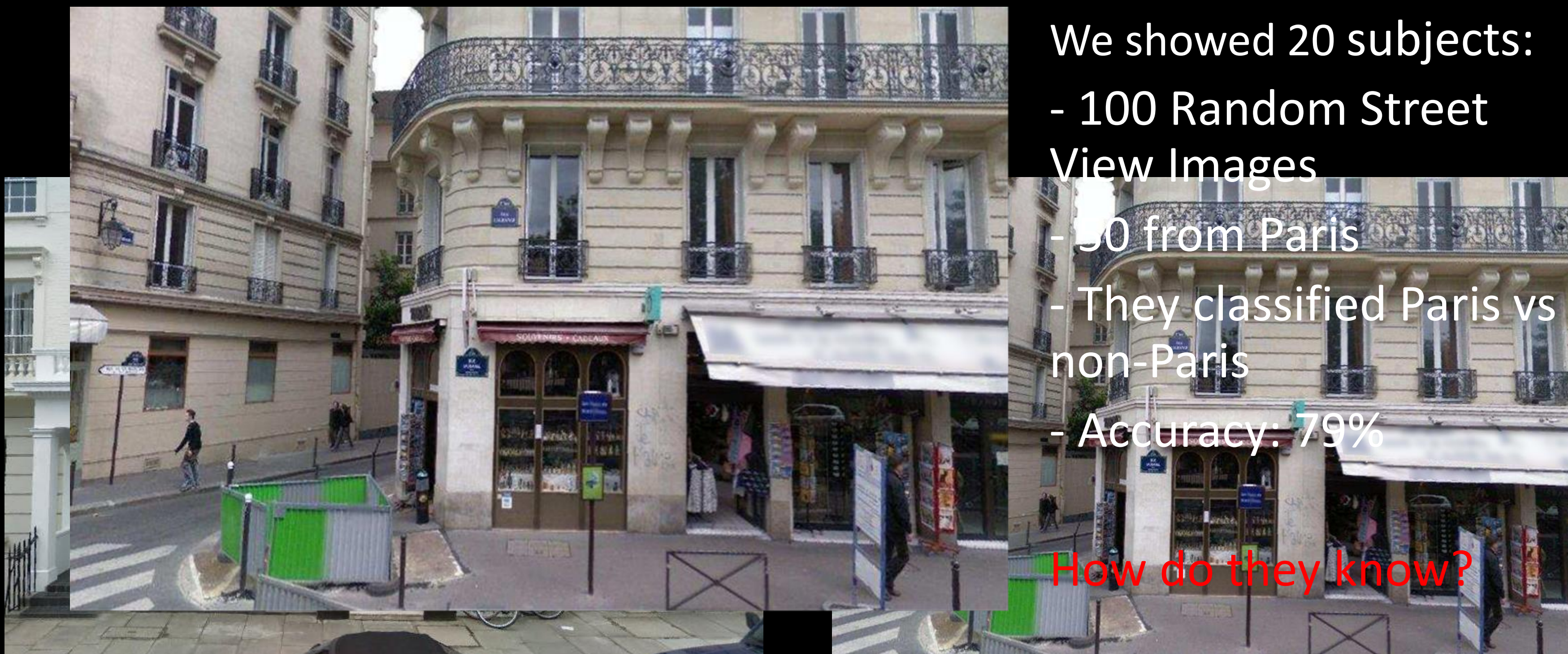
SIGGRAPH 2012

One of these is from Paris
Raise your hand if…

…this is Paris

Doersch et al., "What Makes Paris Look Like Paris?", SIGGRAPH 2012

Raise your hand if…

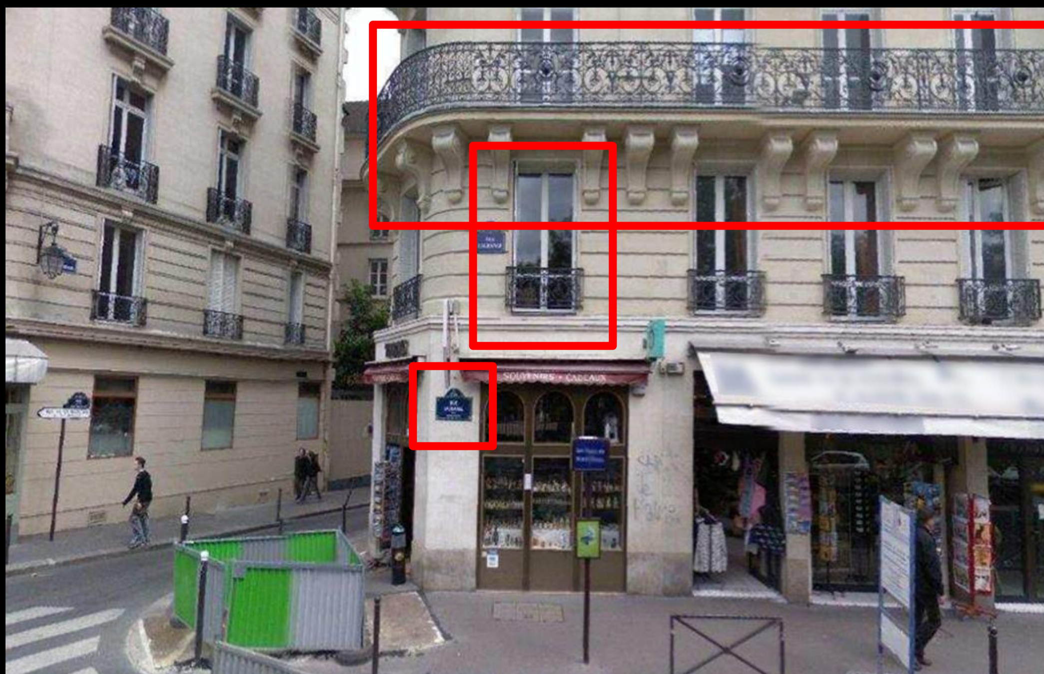


We showed 20 subjects:

- 100 Random Street View Images
- 50 from Paris
- They classified Paris vs non-Paris
- Accuracy: 79%

How do they know?

Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012



We showed 20 subjects:

- 100 Random Street View Images
- 50 from Paris
- They classified Paris non-Paris
- Accuracy: 79%

How do they know?

Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

Our Goal:

Given a large geo-tagged image dataset,
we automatically discover ***visual elements***
that characterize a geographic location

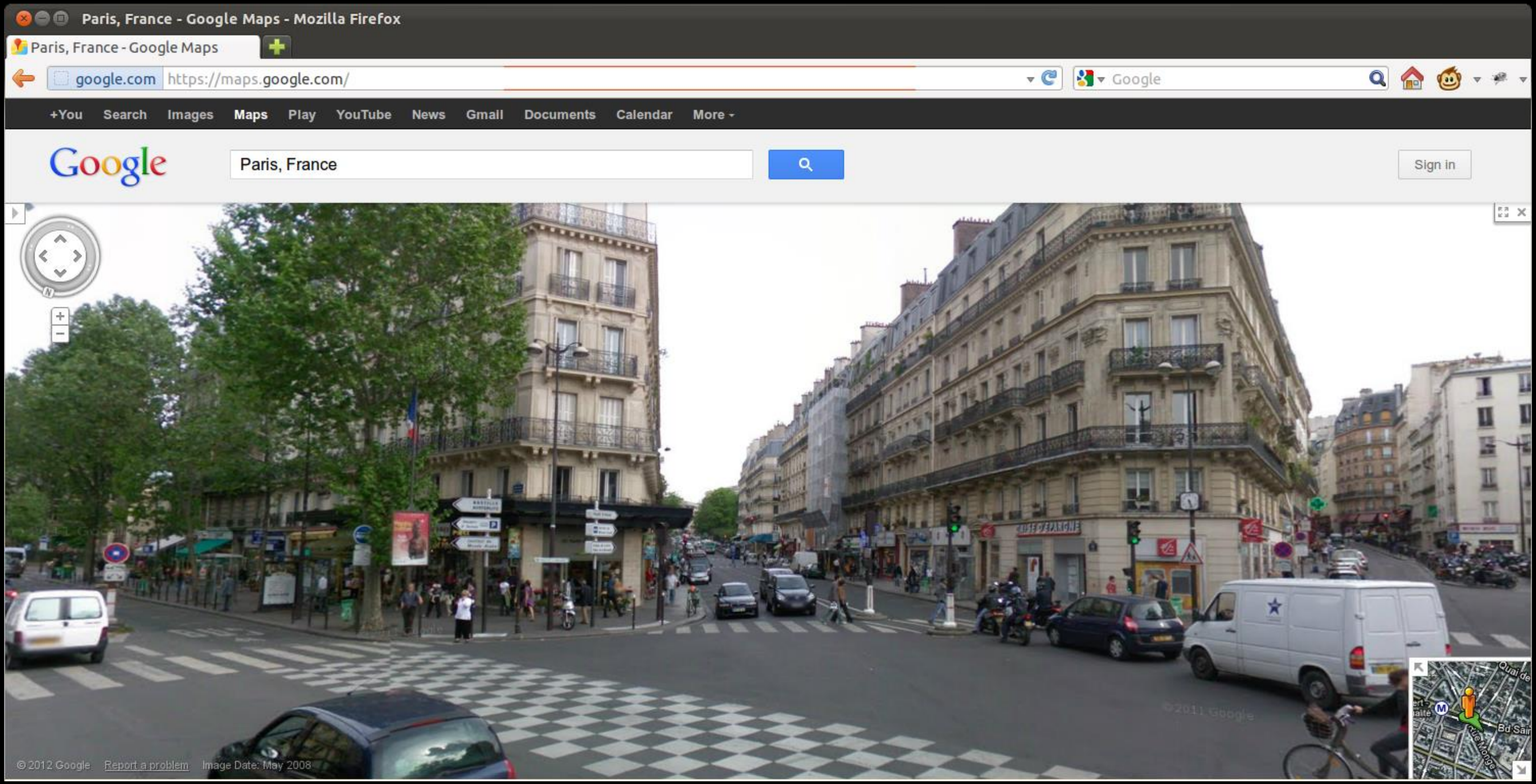
Why might this be a useful task?

Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

Our Hypothesis

- The visual elements that capture Paris:
  - Frequent: Occur often in Paris
  - Discriminative: Are not found outside Paris

Doersch et al., "What Makes Paris Look Like Paris?", SIGGRAPH 2012



Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

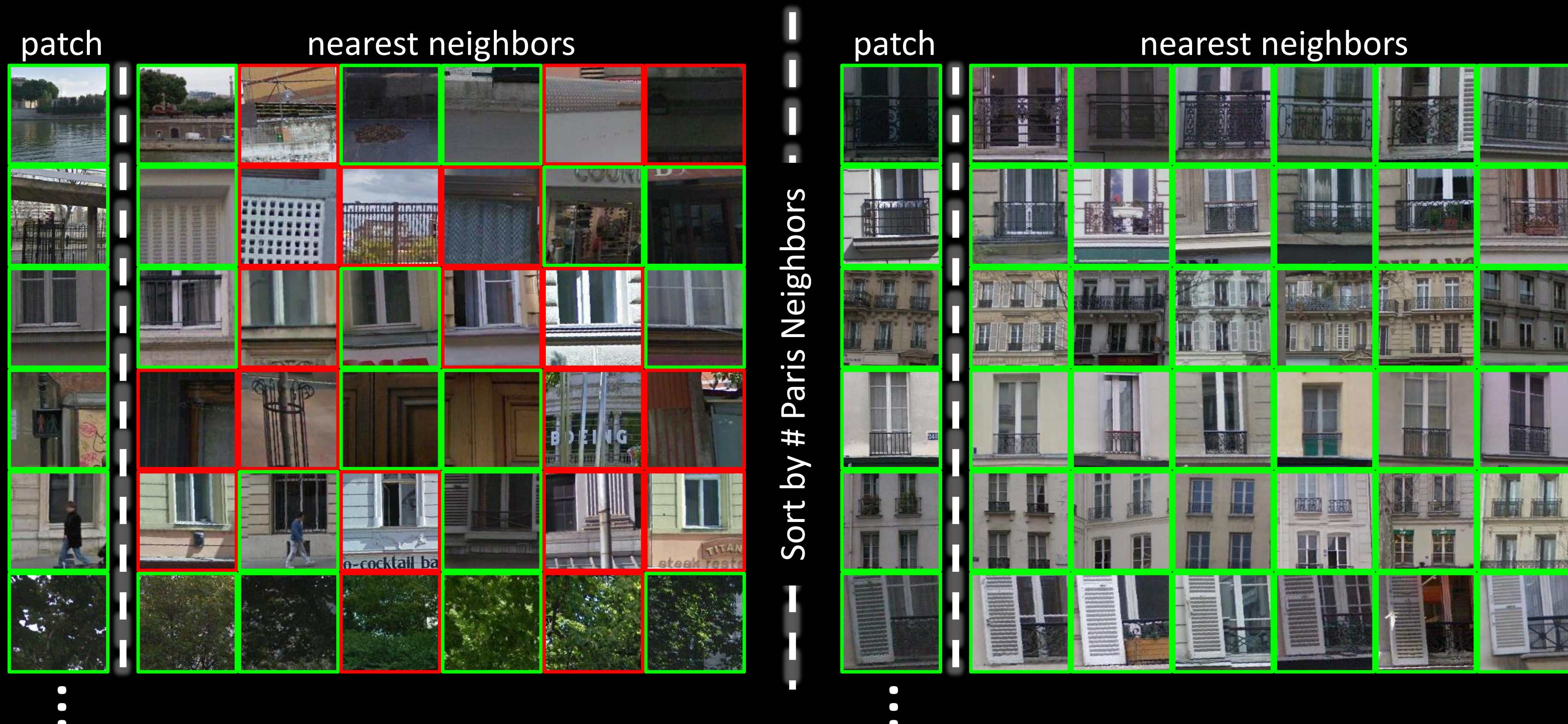
Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

Step 1: Nearest Neighbors for Every Patch

Using normalized correlation of HOG features as a distance metric

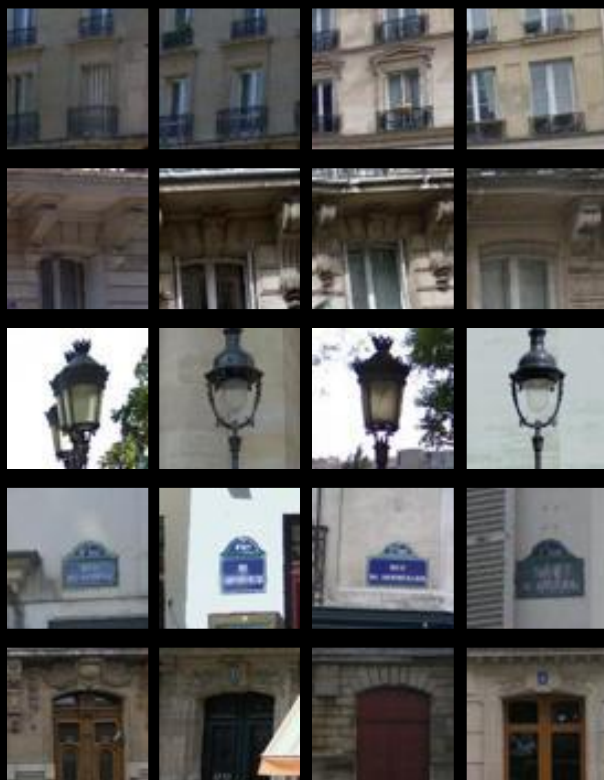
Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

Step 2: Find the Parisian Clusters by Sorting

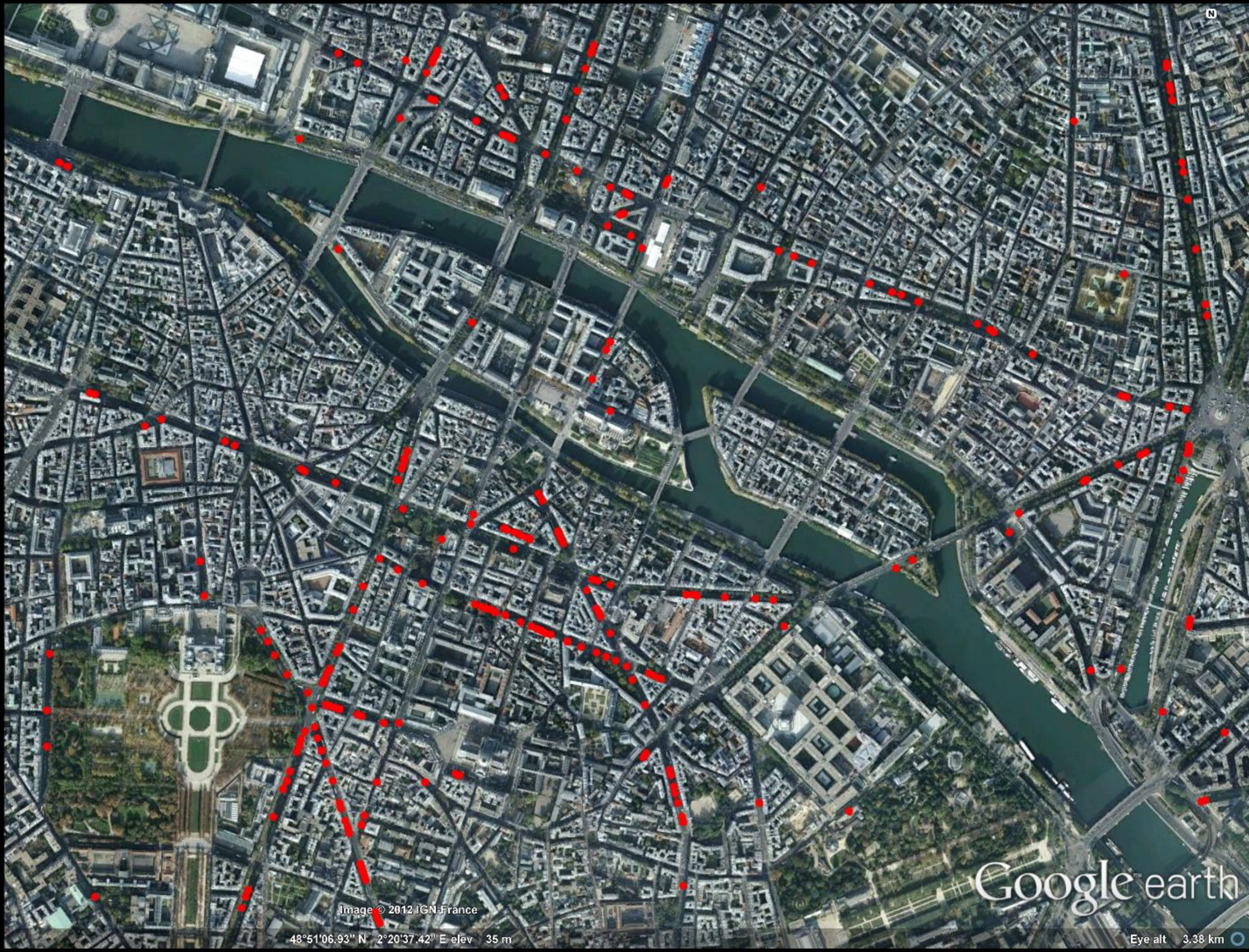


Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

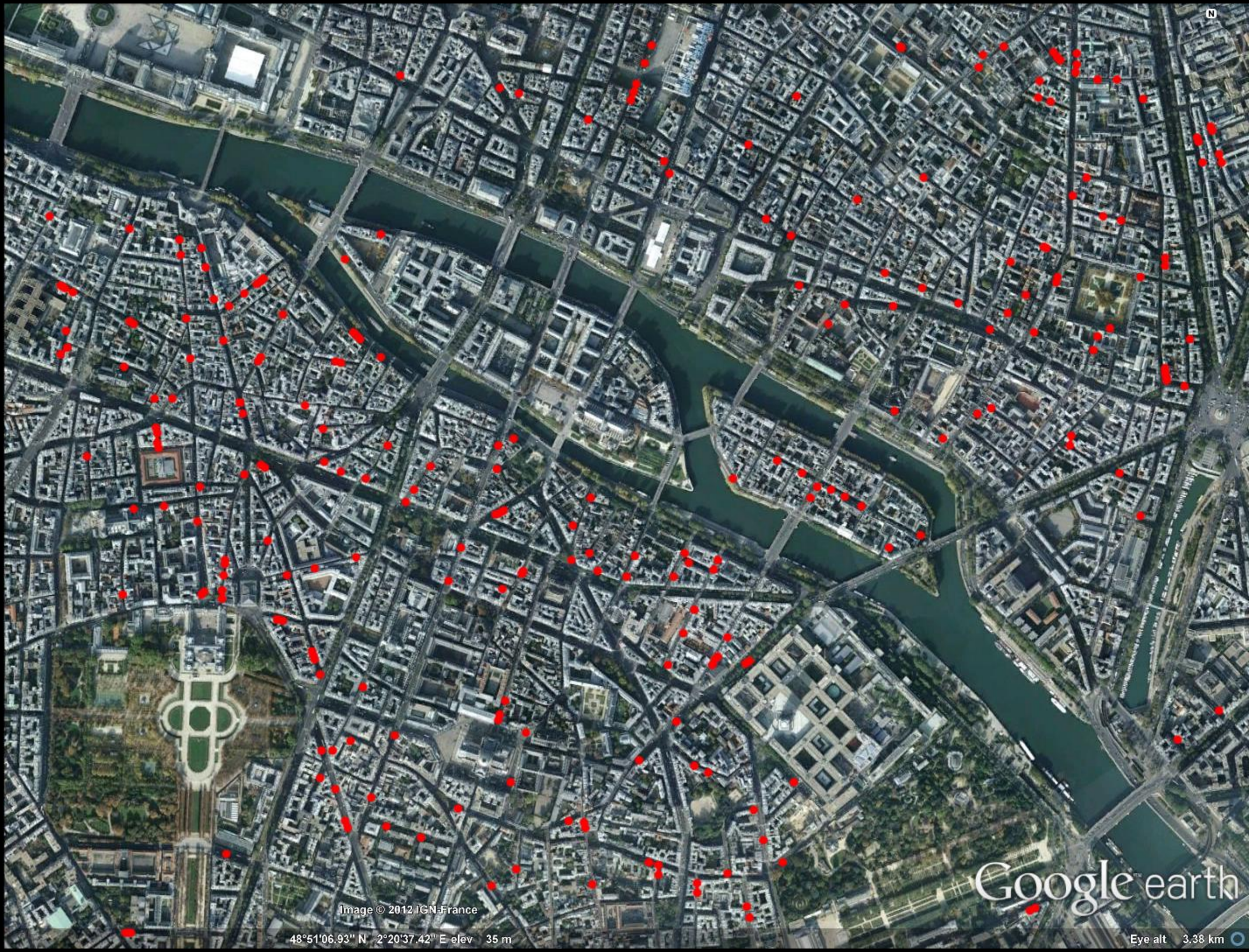
Paris: A Few Top Elements



Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012



Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012



Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

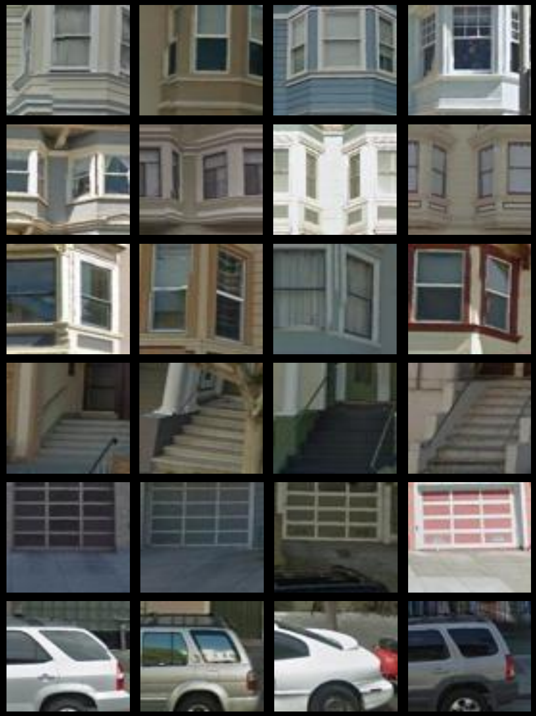
Elements from Prague

Elements from London

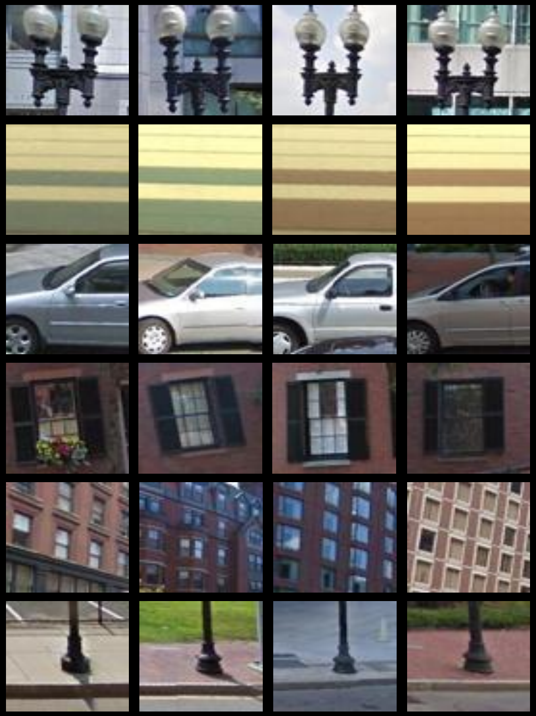
Elements from Barcelona

Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

In the U.S.



Elements from San Francisco



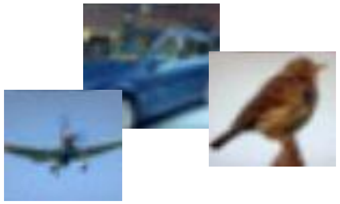
Elements from Boston

Doersch et al., “What Makes Paris Look Like Paris?”, SIGGRAPH 2012

Plan for this last lecture

- Self-supervised learning
  - For images
  - For video
- Visual discovery
  - Discovering style-specific elements
- Generation not recognition
  - Theory/technique
  - Applications

Generative Models

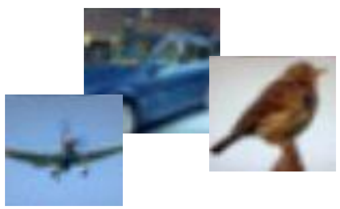


Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

Generative Models



Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

Addresses density estimation, a core problem in unsupervised learning

Several flavors:

- Explicit density estimation: explicitly define and solve for $p_{model}(x)$
- Implicit density estimation: learn model that can sample from $p_{model}(x)$ w/o explicitly defining it

Generative Adversarial Networks

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Problem: Want to sample from complex, high-dimensional training distribution. No direct way to do this!

Solution: Sample from a simple distribution, e.g. random noise. Learn transformation to training distribution.

Q: What can we use to represent this complex transformation?

Generative Adversarial Networks

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014

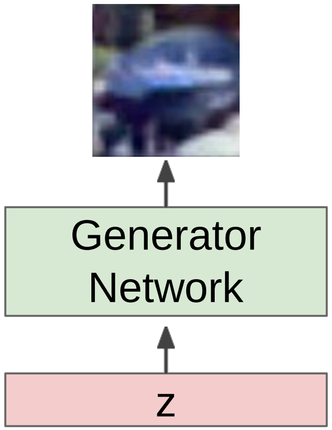
Problem: Want to sample from complex, high-dimensional training distribution. No direct way to do this!

Solution: Sample from a simple distribution, e.g. random noise. Learn transformation to training distribution.

Q: What can we use to represent this complex transformation?

A: A neural network!

Output: Sample from training distribution



Input: Random noise

Training GANs: Two-player game

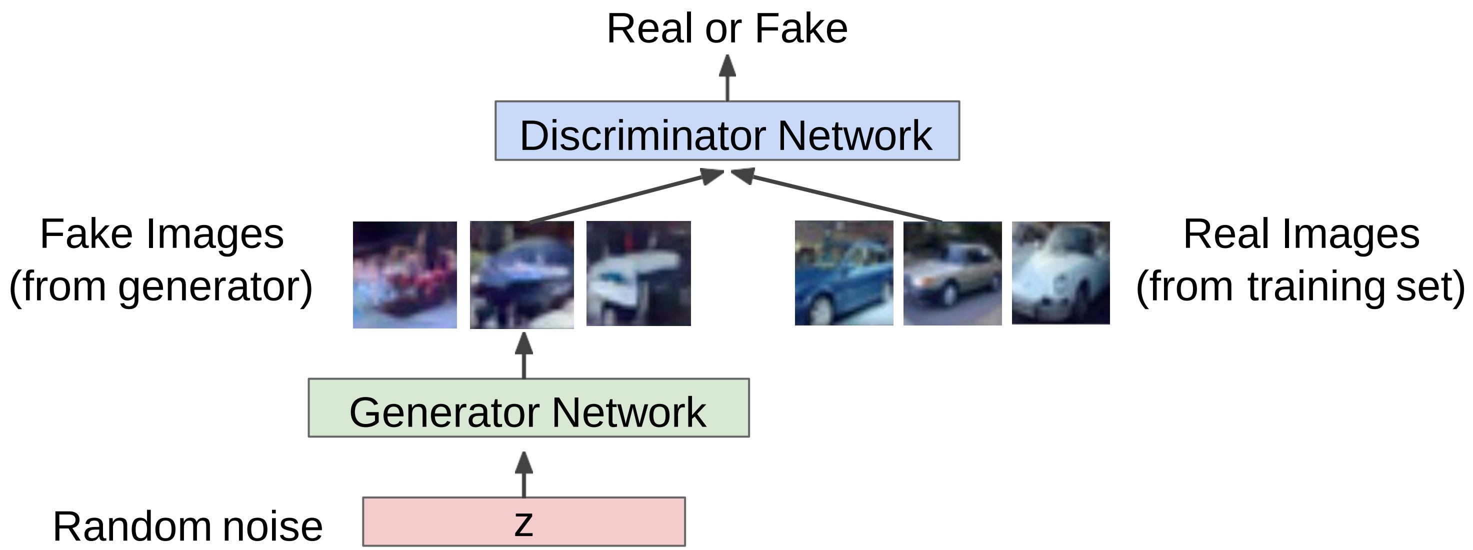
Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

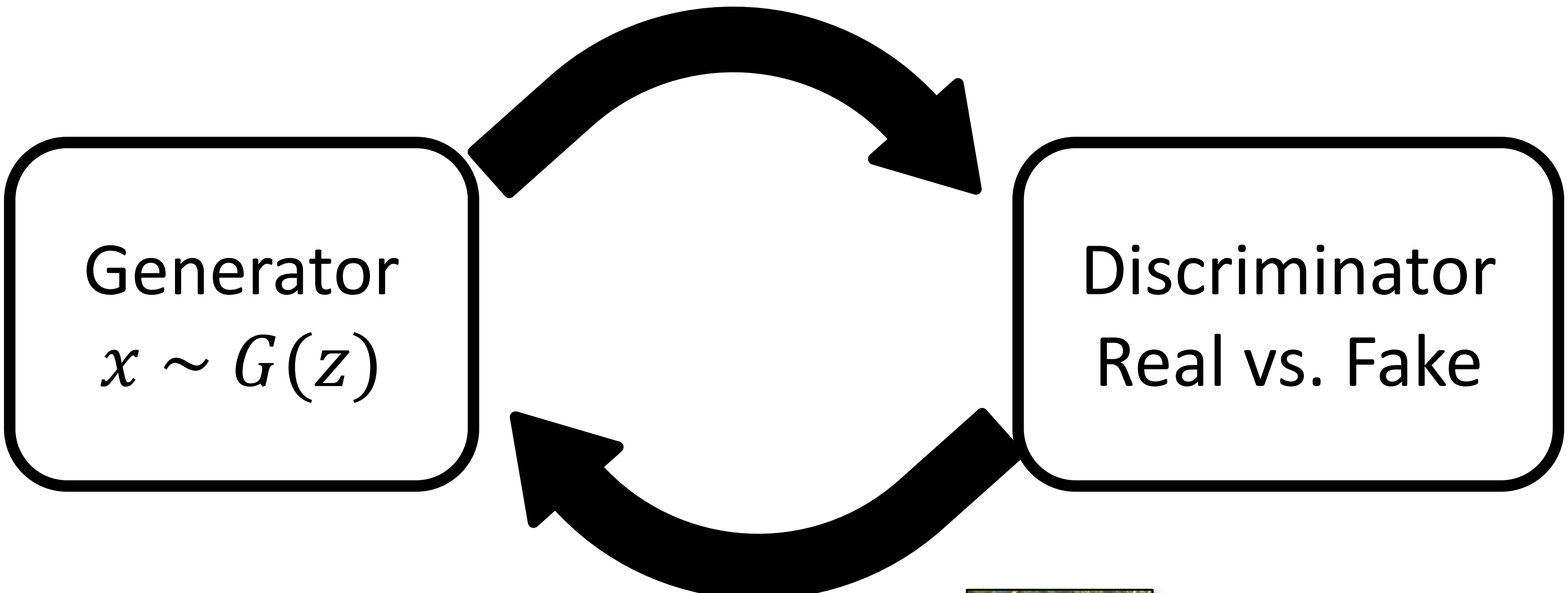
Training GANs: Two-player game

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images



Adversarial Networks Framework



[Goodfellow et al. 2014]

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

Train jointly in **minimax game**

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

Train jointly in **minimax game**

Discriminator outputs likelihood in (0,1) of real image

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log \underbrace{D_{\theta_d}(x)}_{\text{Discriminator output for real data x}} + \mathbb{E}_{z \sim p(z)} \log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\text{Discriminator output for generated fake data G(z)}}) \right]$$

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

Train jointly in **minimax game**

Discriminator outputs likelihood in (0,1) of real image

Minimax objective function:

$$\boxed{\min_{\theta_g}} \boxed{\max_{\theta_d}} \left[\mathbb{E}_{x \sim p_{data}} \boxed{\log \underbrace{D_{\theta_d}(x)}_{\text{Discriminator output for real data x}}} + \mathbb{E}_{z \sim p(z)} \boxed{\log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\text{Discriminator output for generated fake data G(z)}}))} \right]$$

- Discriminator (θ_d) wants to **maximize objective** such that D(x) is close to 1 (real) and D(G(z)) is close to 0 (fake)
- Generator (θ_g) wants to **minimize objective** such that D(G(z)) is close to 1 (discriminator is fooled into thinking generated G(z) is real)

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1. **Gradient ascent** on discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

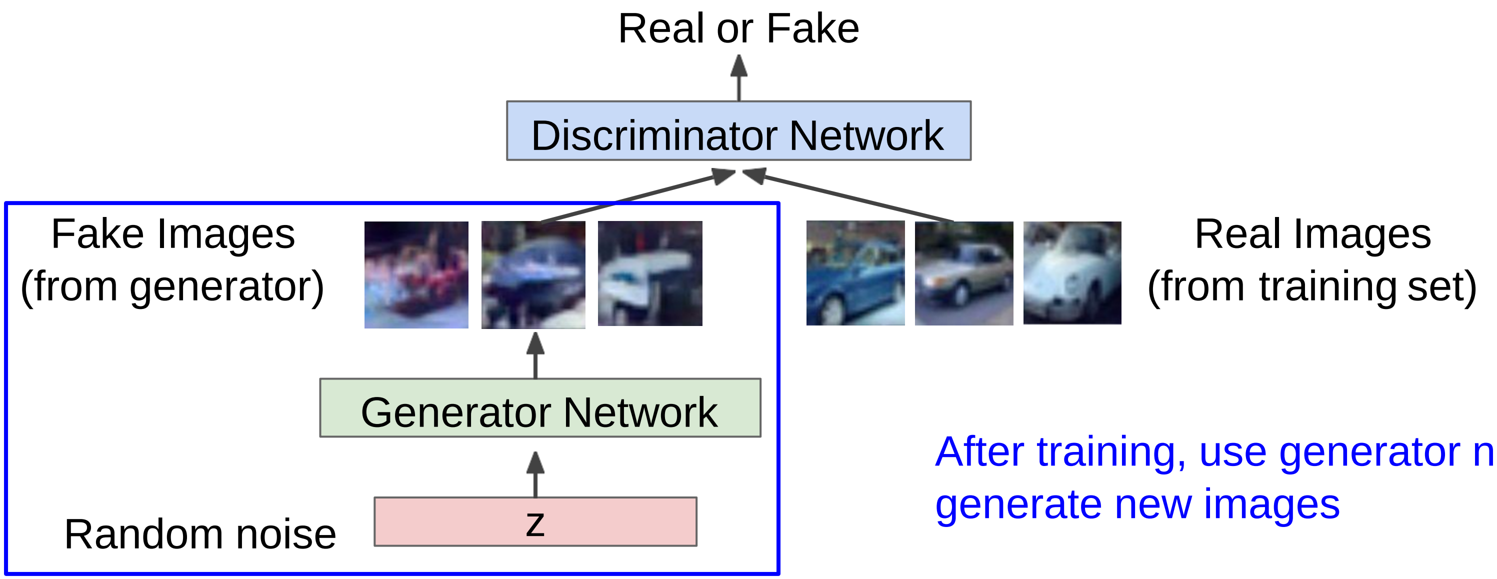
2. **Gradient descent** on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images



After training, use generator network to generate new images

Generative Adversarial Nets

Samples from the model look amazing!

Radford et al, ICLR 2016

Serena Young

Interpretable Vector Math

Radford et al, ICLR 2016

Smiling woman Neutral woman Neutral man

Samples from the model

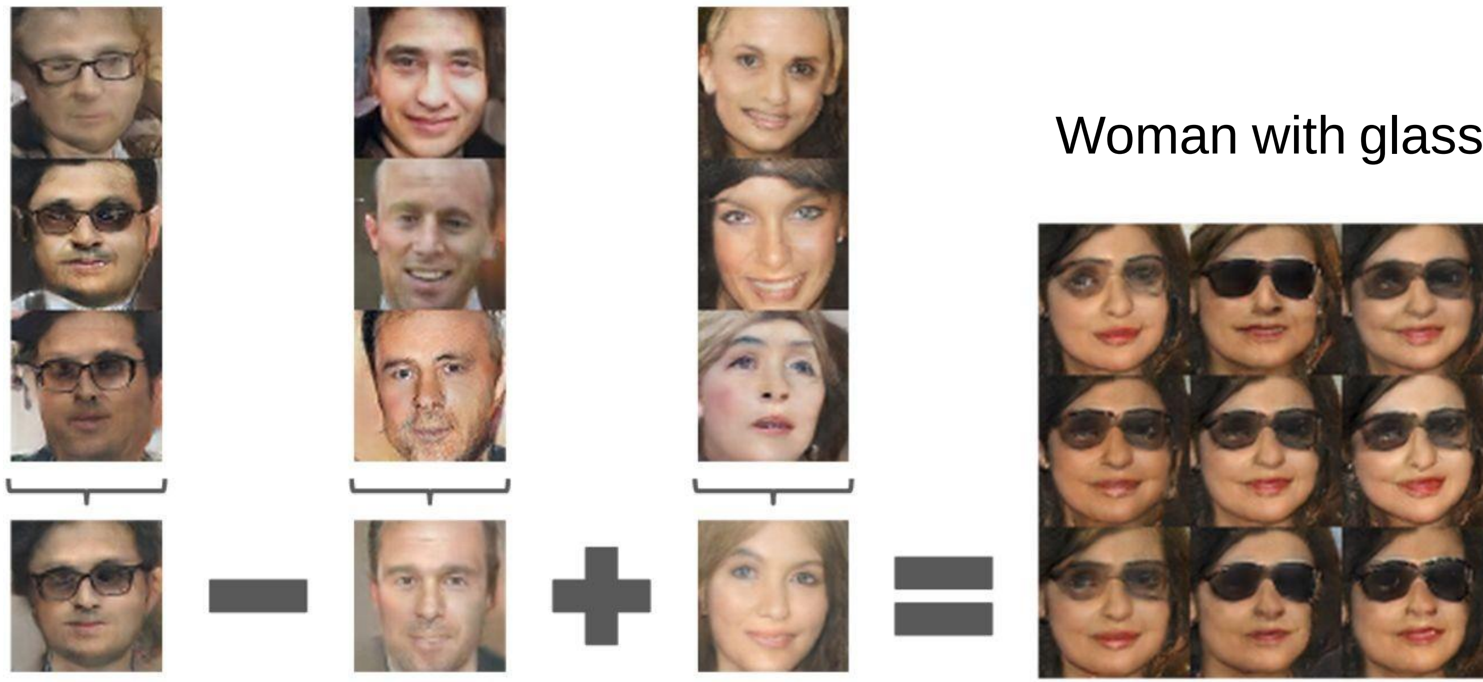
Average features, do arithmetic

Interpretable Vector Math

Glasses man No glasses man No glasses woman

Radford et al,
ICLR 2016

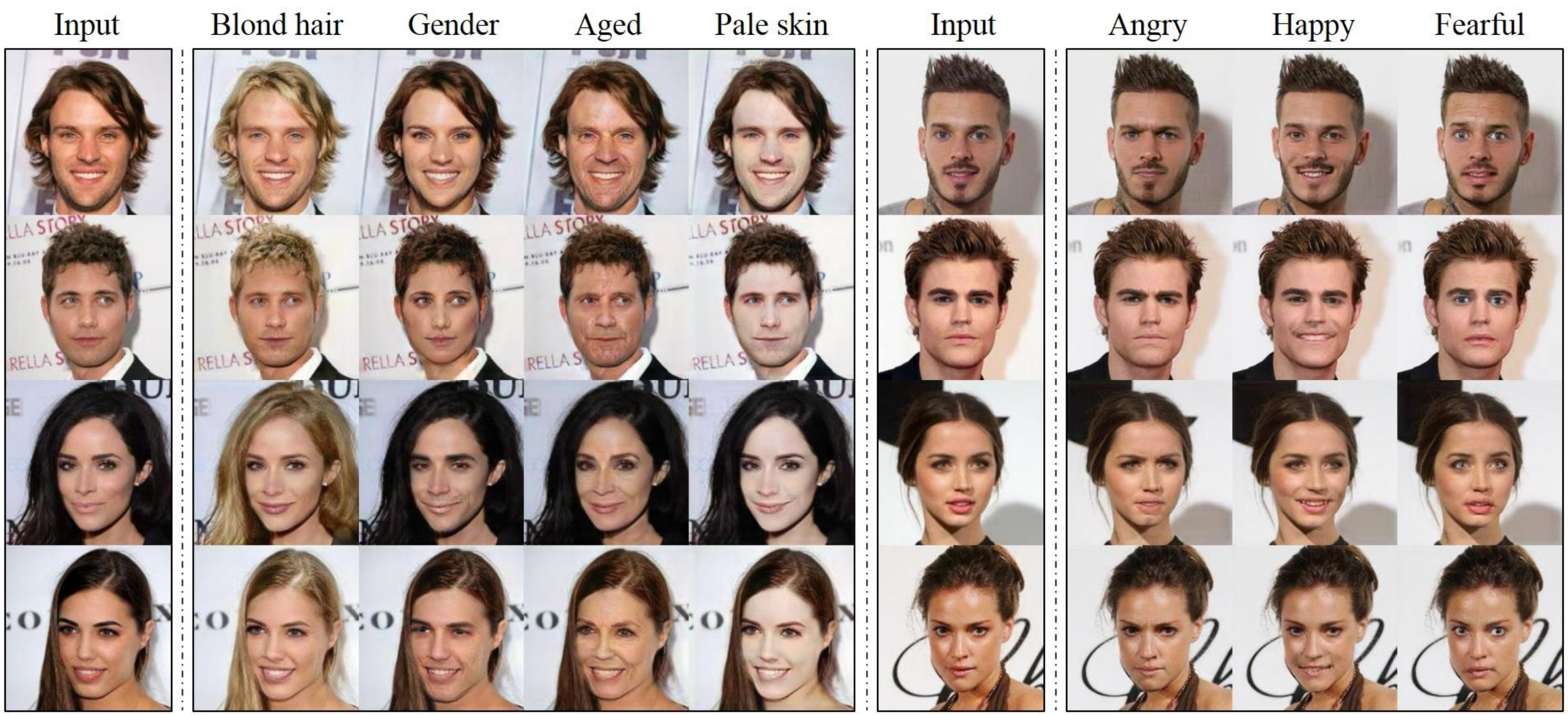
Woman with glasses



Celebrities Who Never Existed

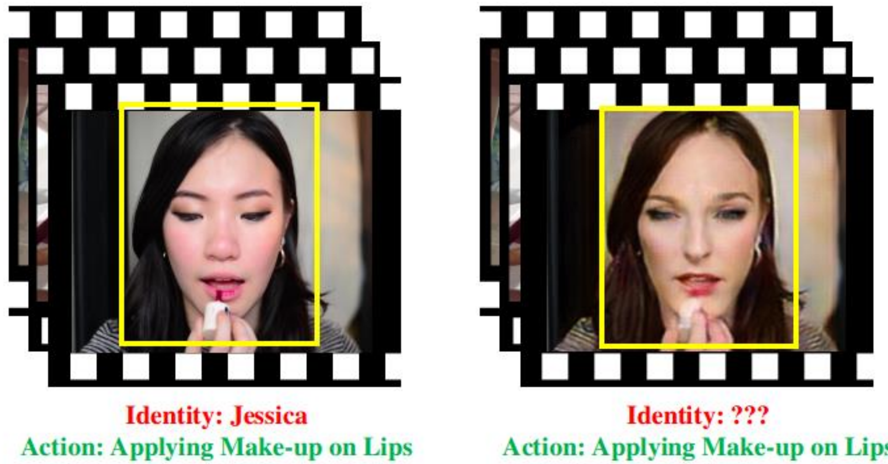
Karras et al., "Progressive Growing of GANs for Improved Quality, Stability, and Variation", ICLR 2018

StarGAN



Choi et al., "[StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation]", CVPR 2018

GANs for Privacy (Action Detection)



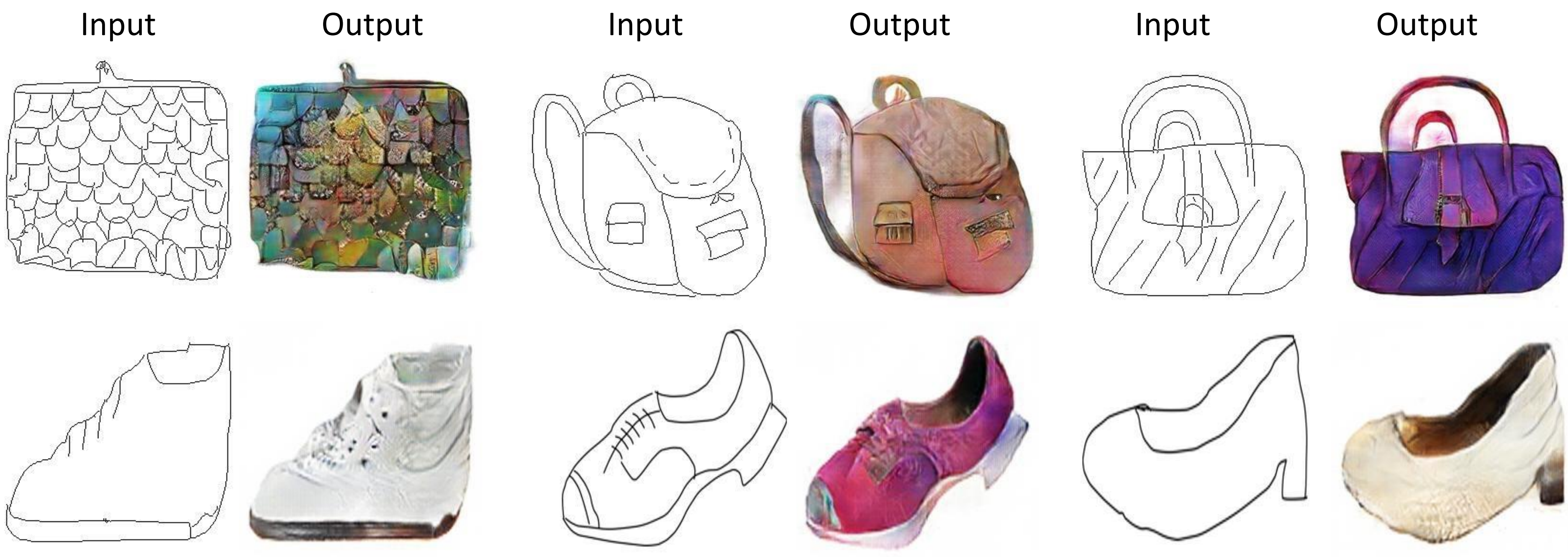
Ren et al., "Learning to Anonymize Faces for Privacy Preserving Action Detection", arxiv 2018

Artificial Fashion: vue.ai

Edges → Images

Edges from [Xie & Tu, 2015]

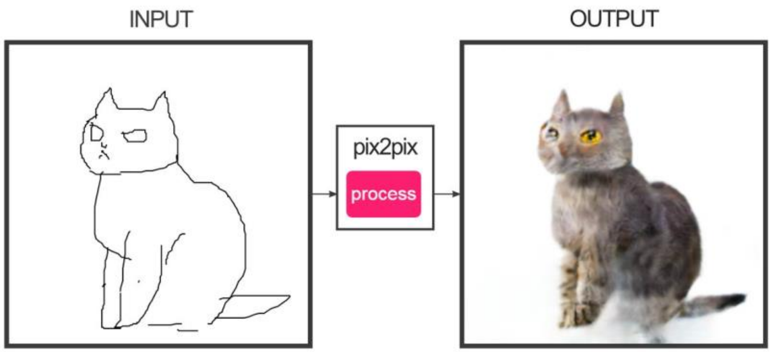
Sketches → Images



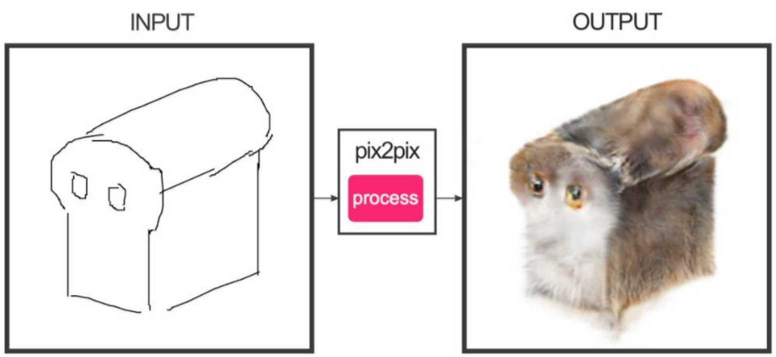
Trained on Edges → Images

Data from [Eitz, Hays, Alexa, 2012]

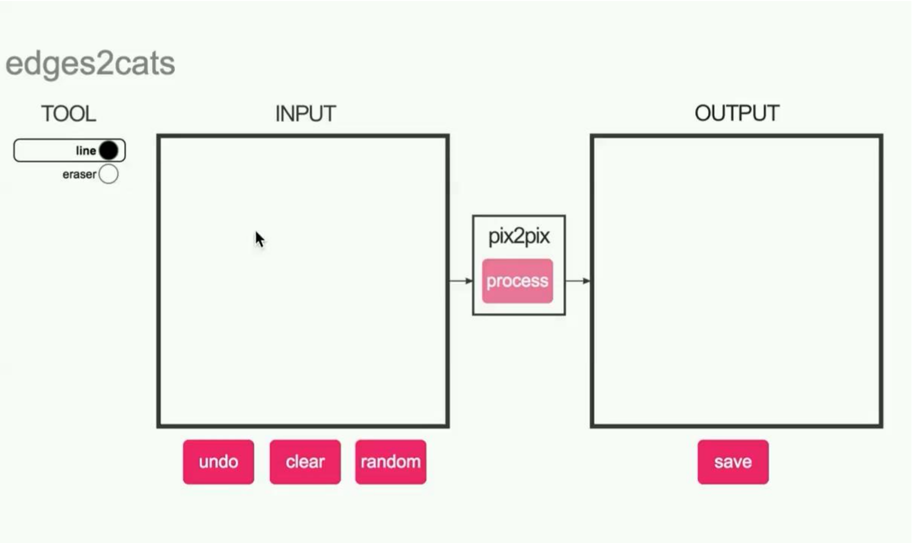
#edges2cats [Christopher Hesse]



@gods_tail



Ivy Tasi @ivymyt



@matthematician

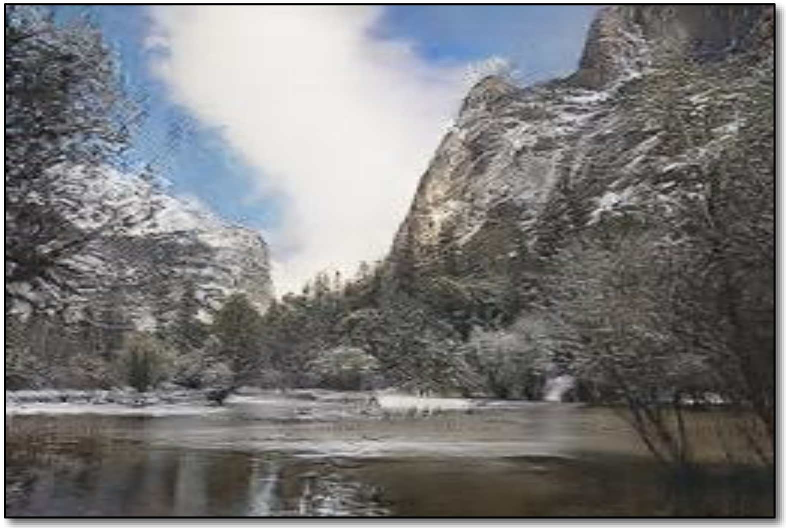
Vitaly Vidmirov @vvid

https://affinelayer.com/pixsrv/

Changing artistic style

Pix2pix / CycleGAN

Changing seasons

Pix2pix / CycleGAN