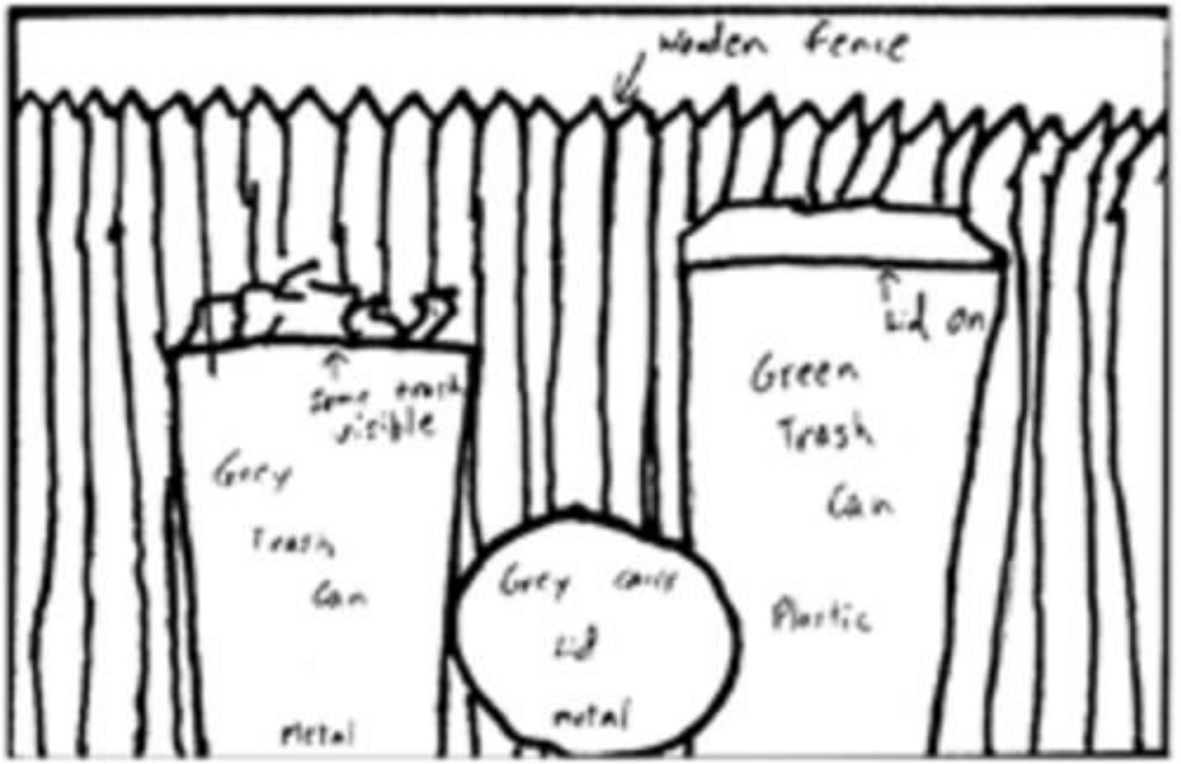
Representation Learning

Most slides from Phil Isola

Alexei Efros, CS280, Spring 2024

Observed image

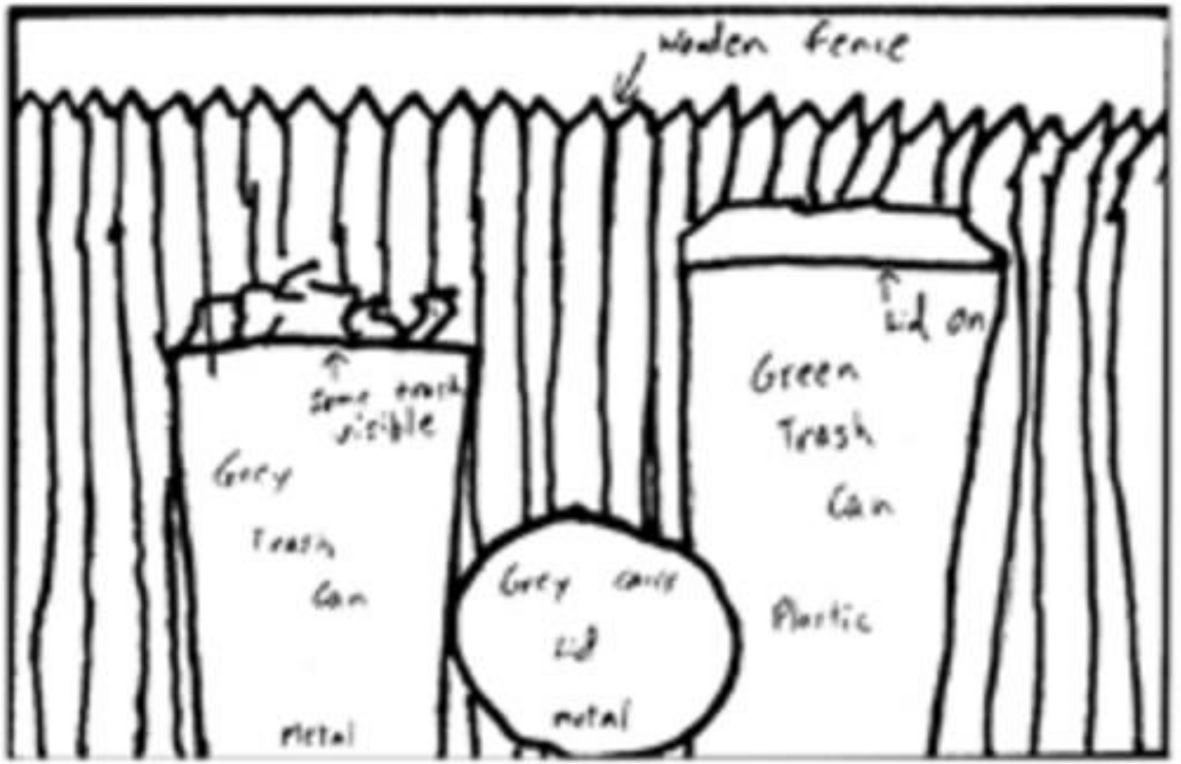
Drawn from memory



[Bartlett, 1932]
[Intraub & Richardson, 1989]

Observed image

Drawn from memory



[Bartlett, 1932]
[Intraub & Richardson, 1989]

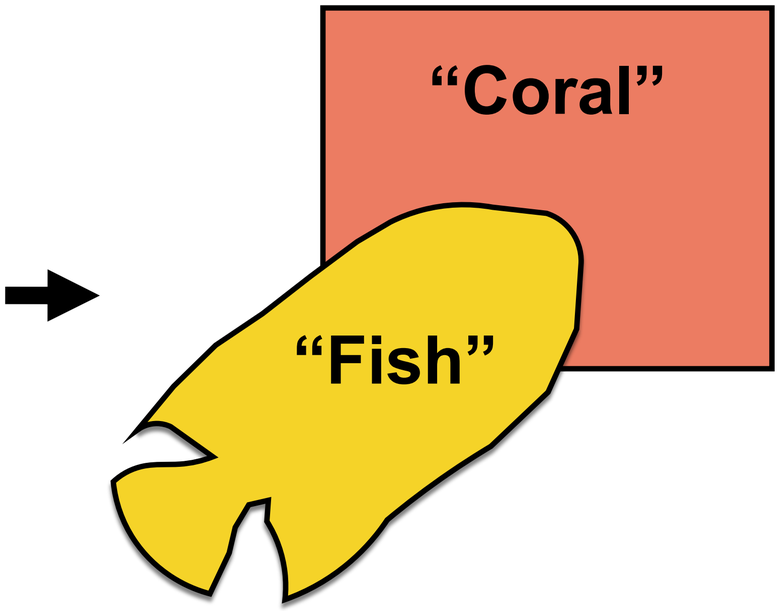
"I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees.”

— Max Wertheimer, 1923

Representation learning

$\mathbf{x}$

Image

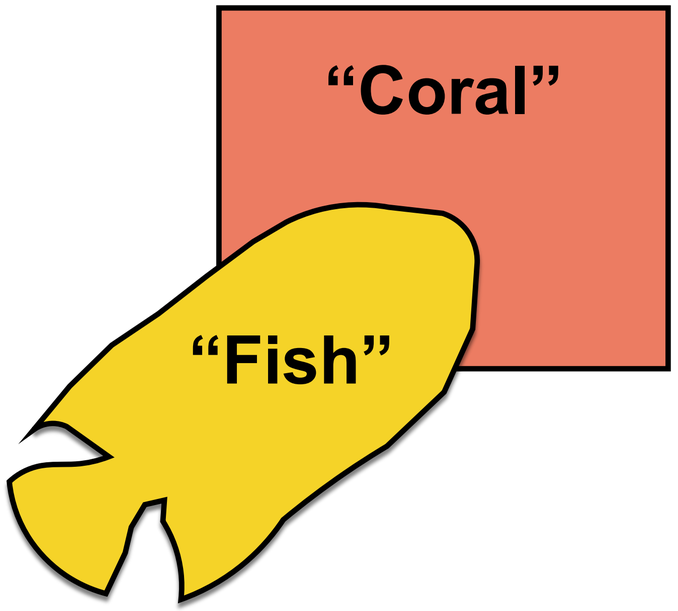


Compact mental representation

Representation learning

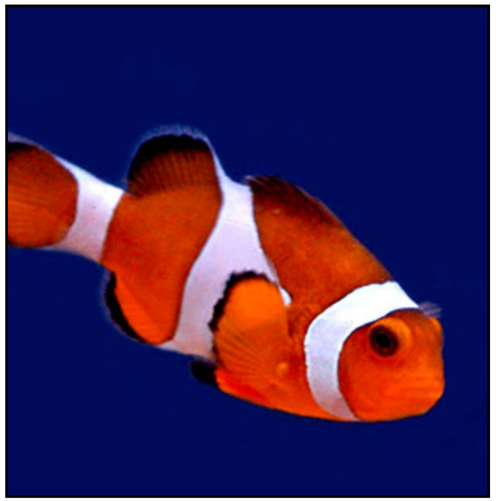
Good representations are:

1. Compact (*minimal*)
2. Explanatory (*sufficient*)
3. Disentangled (*independent factors*)
4. Interpretable
5. *Make subsequent problem solving easy*



[See “Representation Learning”, Bengio 2013, for more commentary]

Classical object recognition



Edges

Texture

Colors

Segments

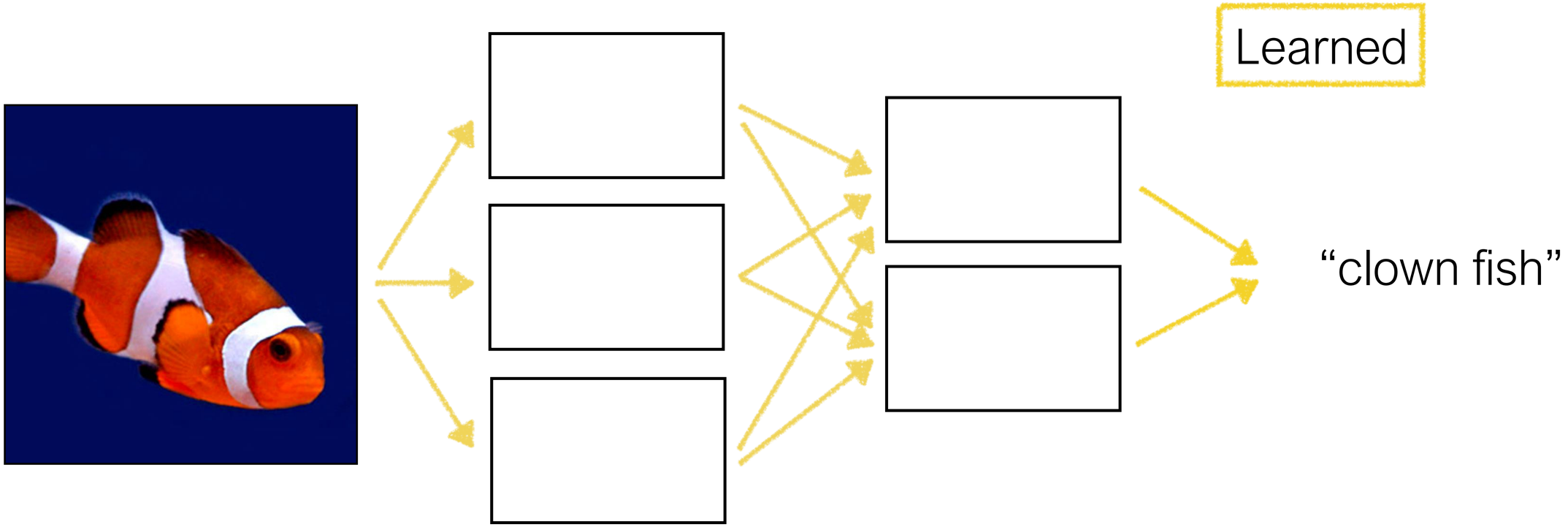
Parts

“clown fish”

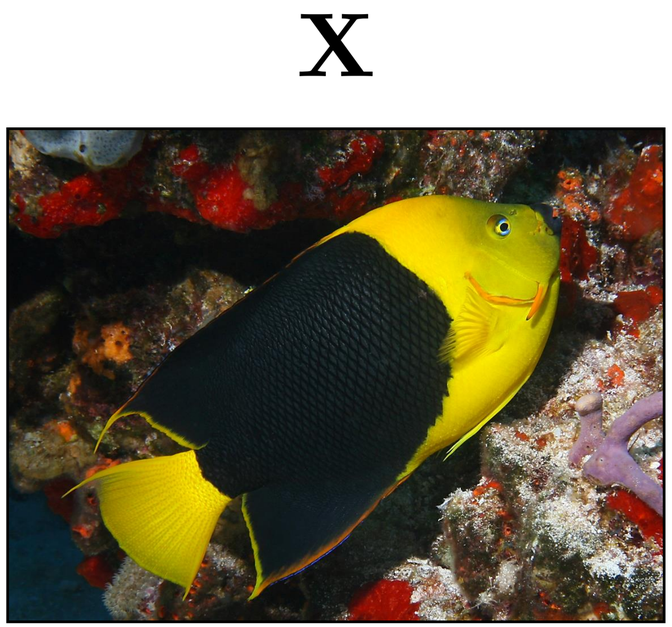
Feature extractors

Classifier

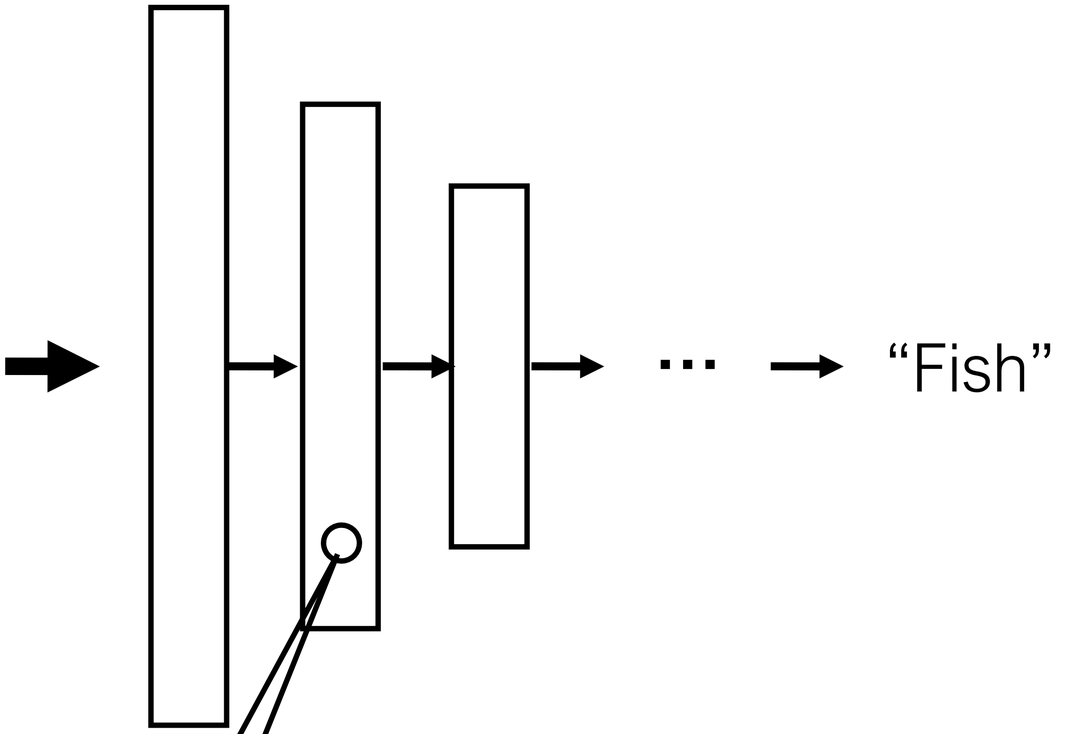
Deep learning

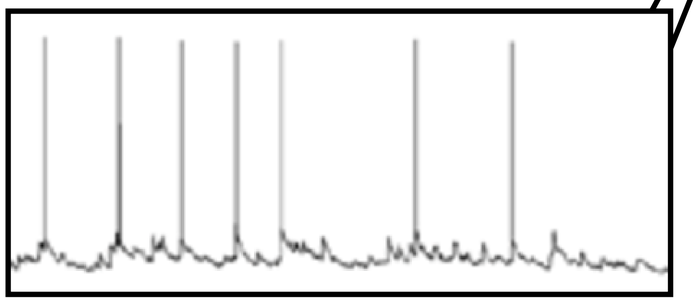


What do deep nets internally learn?



Deep Net “Electrophysiology”





[Zeiler & Fergus, ECCV 2014]

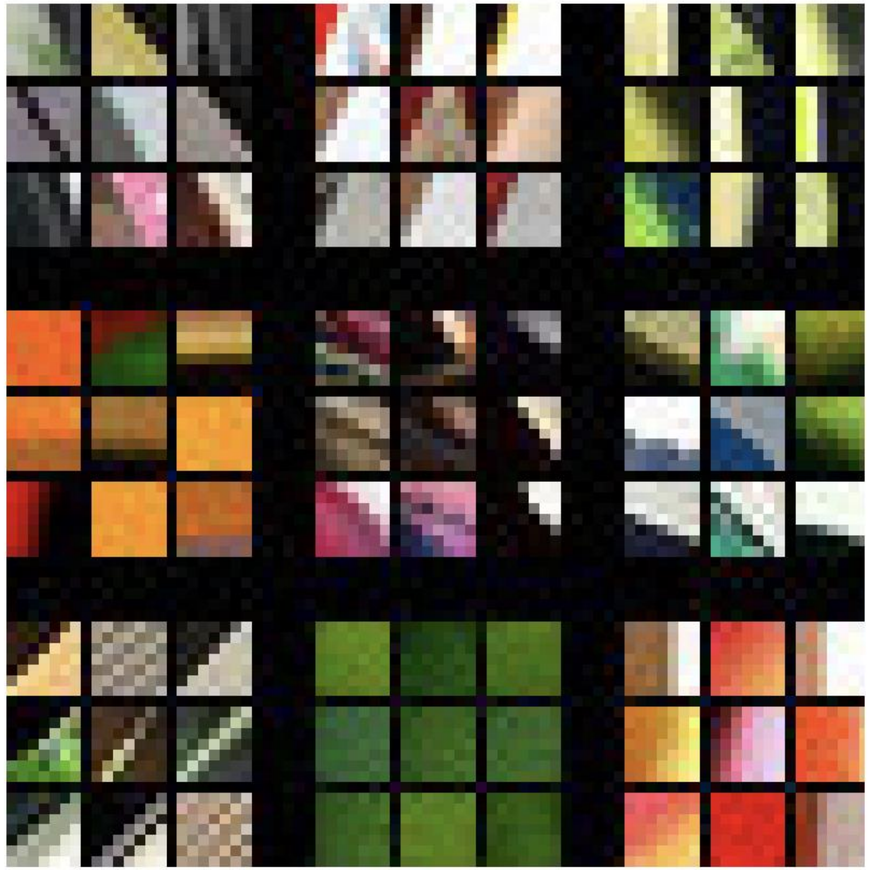
[Zhou et al., ICLR 2015]

Visualizing and Understanding CNNs

[Zeiler and Fergus, 2014]

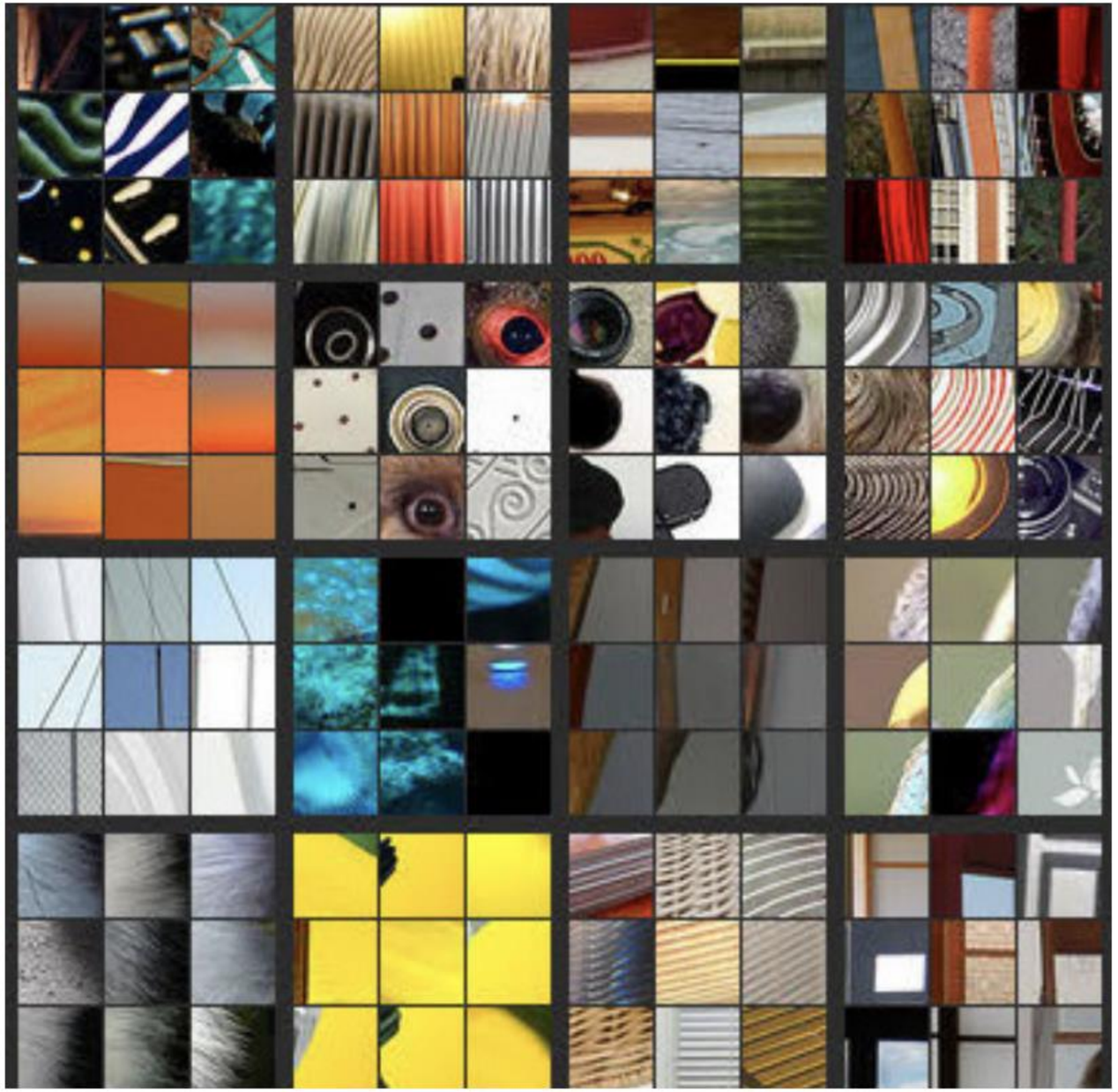
Gabor-like filters learned by **layer 1**

Image patches that activate each of the **layer 1** filters most strongly



[Zeiler and Fergus, 2014]

Image patches that activate several of the **layer 2** neurons most strongly



[Zeiler and Fergus, 2014]

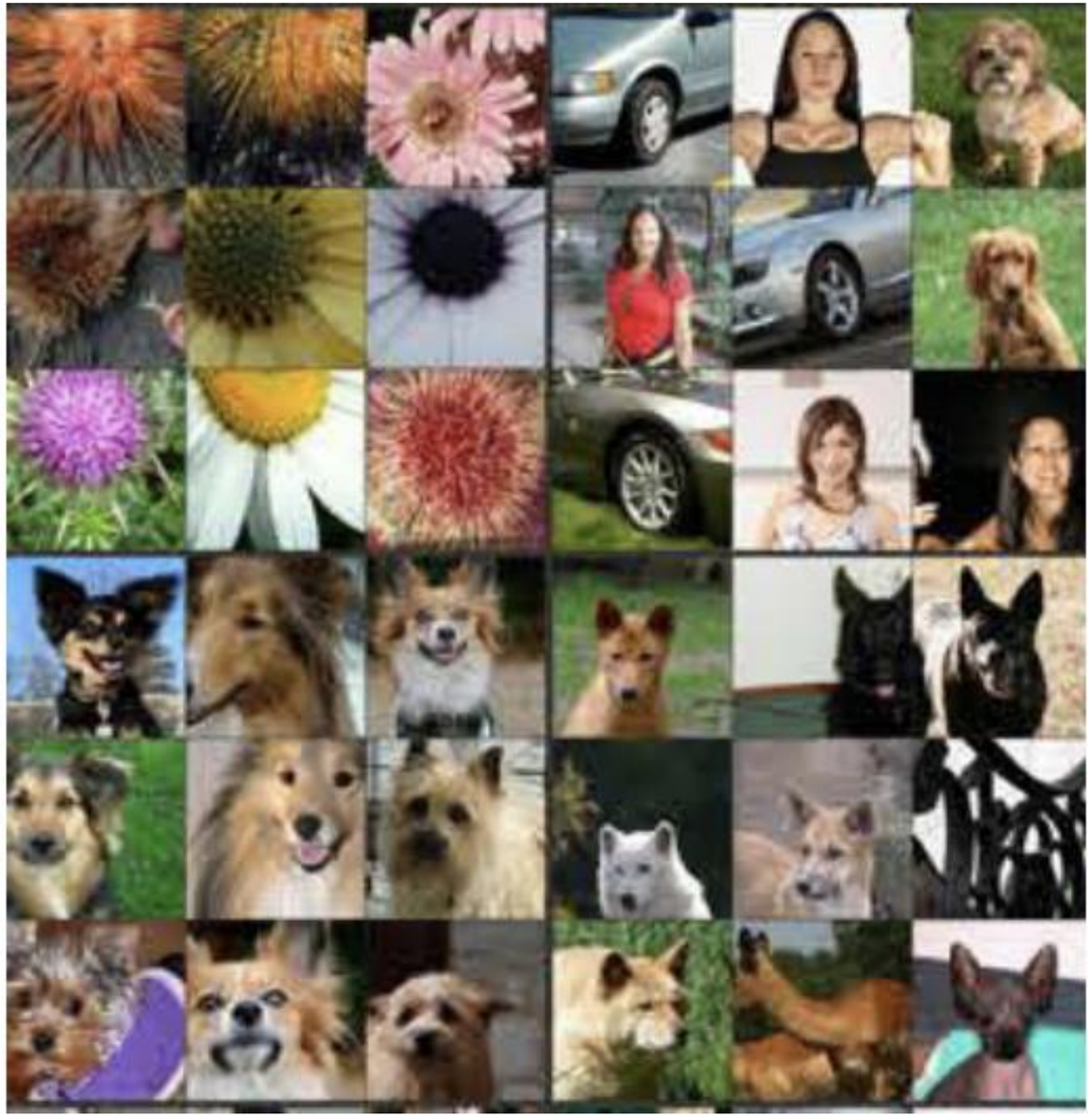
Image patches that activate several of the **layer 3** neurons most strongly

[Zeiler and Fergus, 2014]

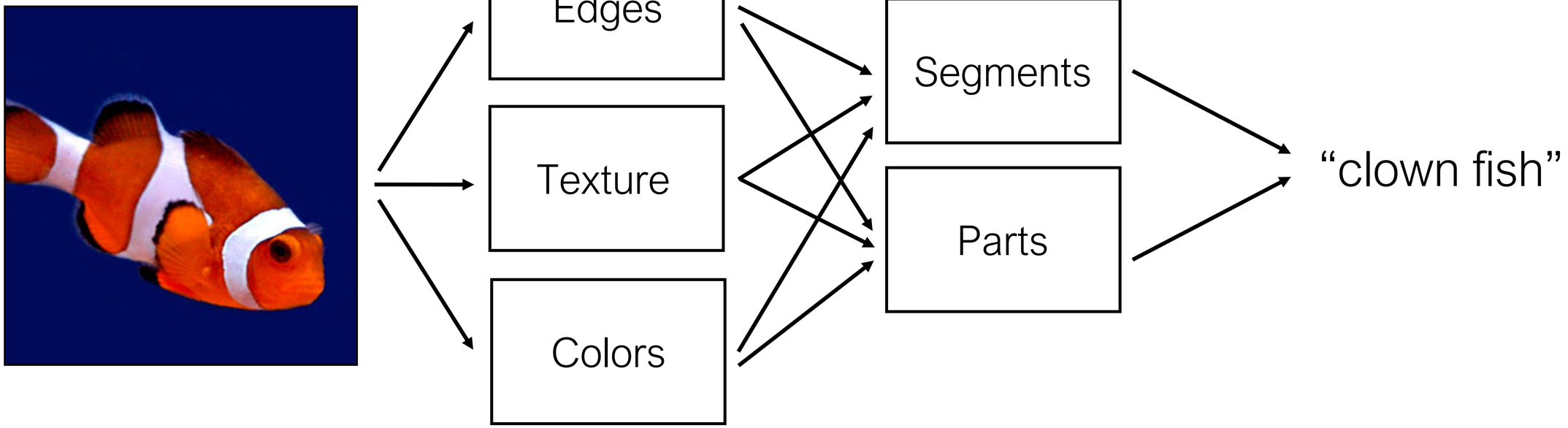
Image patches that activate several of the **layer 4** neurons most strongly

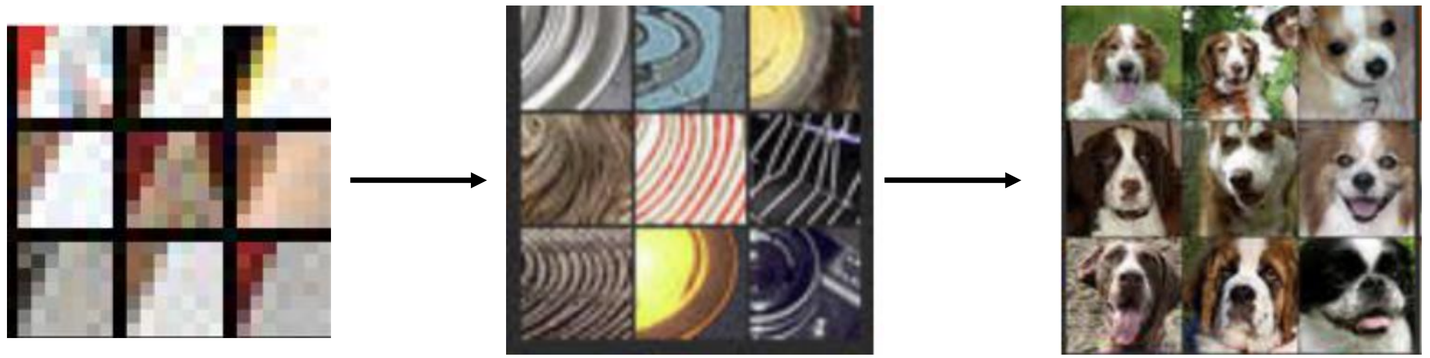
[Zeiler and Fergus, 2014]

Image patches that activate several of the **layer 5** neurons most strongly



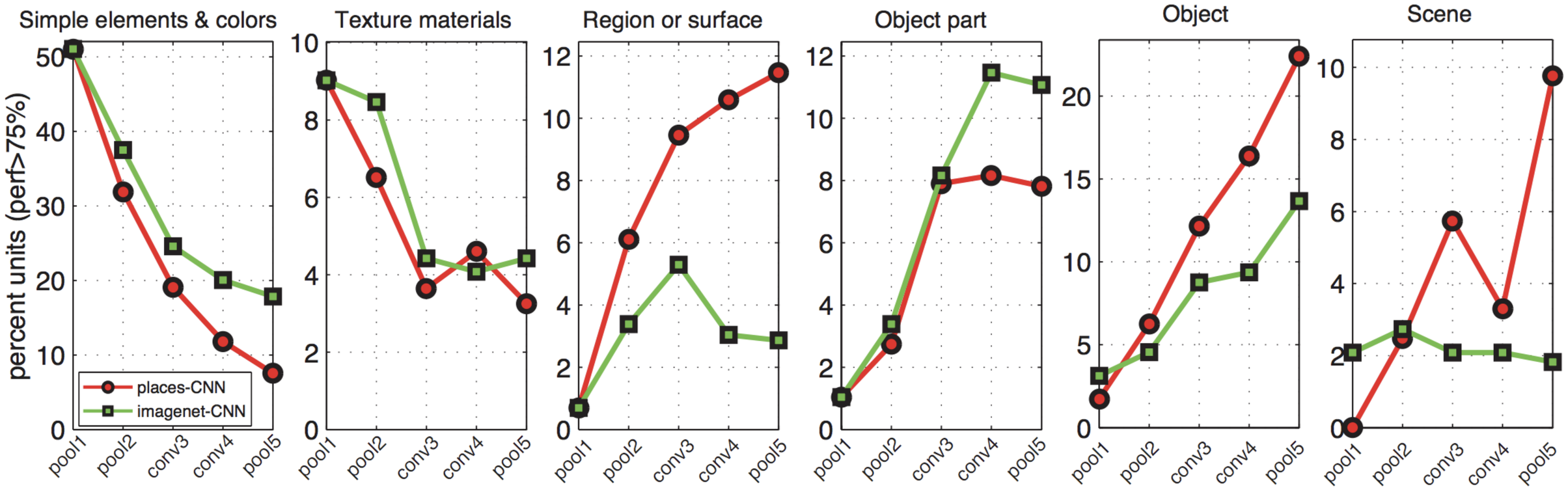
CNNs *learned* the classical visual recognition pipeline!





Object Detectors Emergence in Deep Scene CNNs

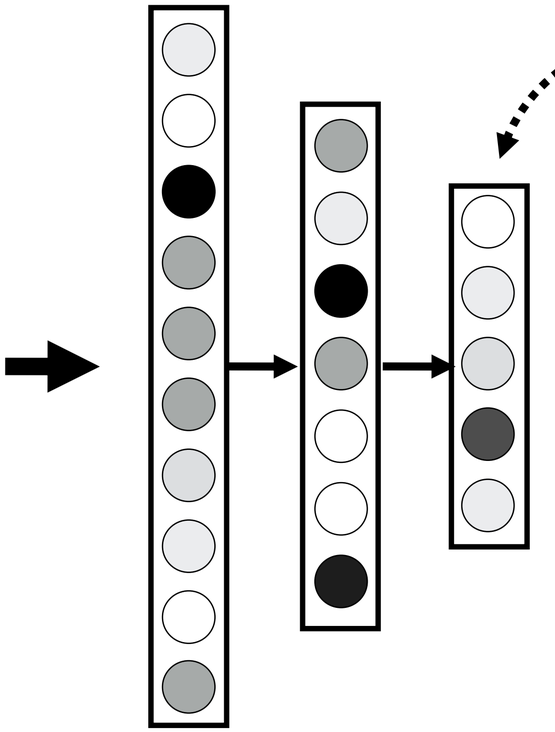
[Zhou, Khosla, Lapedriza, Oliva, Torralba, ICLR 2015]



im2vec

$\mathbf{x}$

Image



layer 3 representation of image

layer 1 representation of image

Represent image as a neural **embedding** — a vector/tensor of neural activations

(perhaps representing a vector of detected texture patterns or object parts)

Investigating a representation via similarity analysis

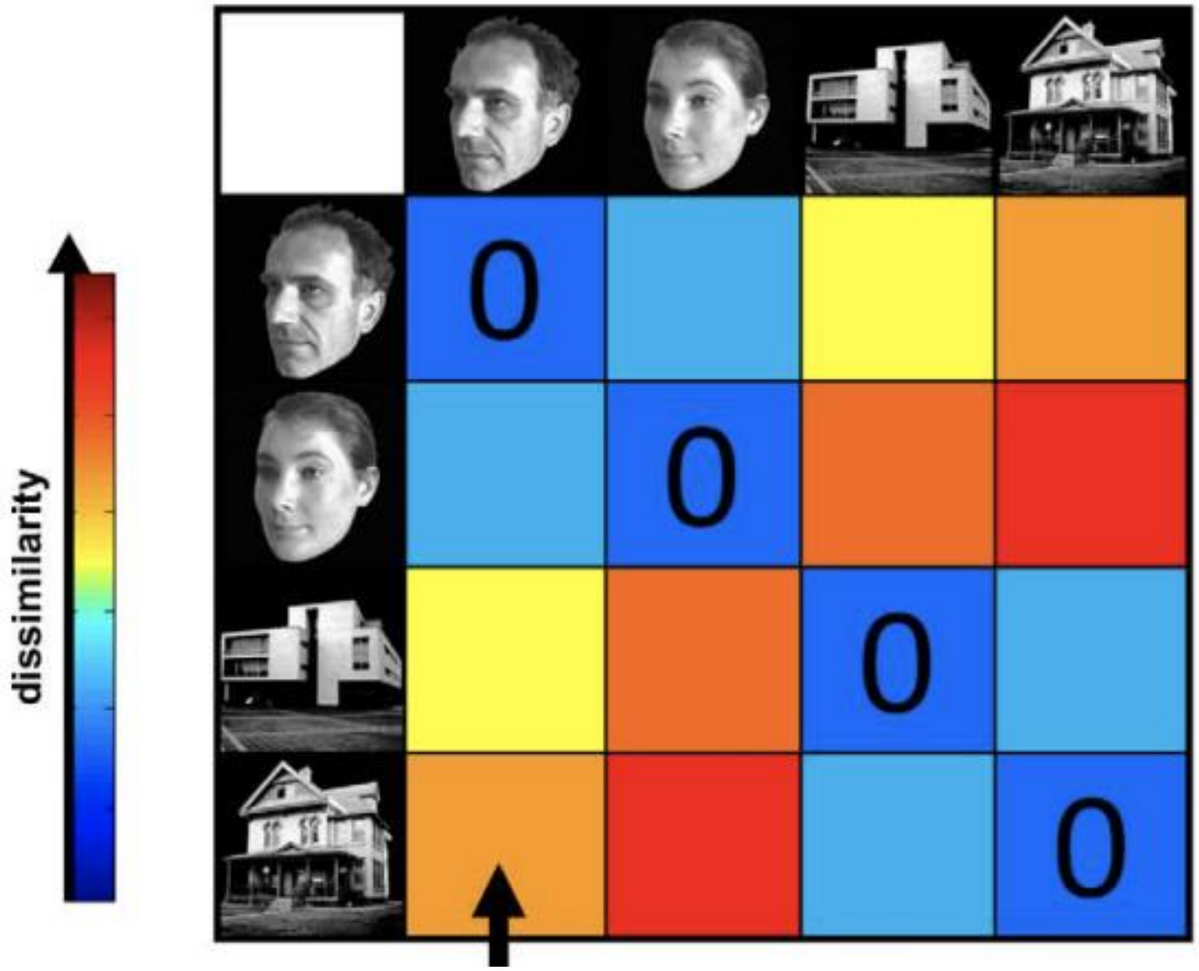
How similar are these two images?

How about these two?

[Kriegeskorte et al. 2008]

Investigating a representation via similarity analysis

Representational Dissimilarity Matrix



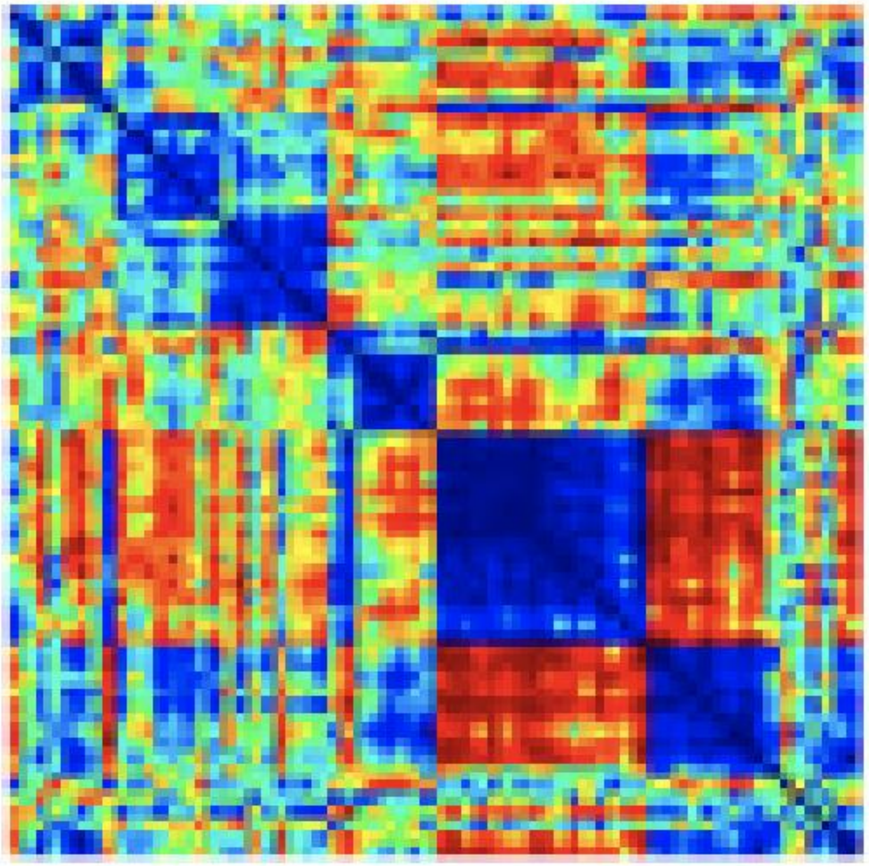
$$\|\mathbf{h}_i - \mathbf{h}_j\|$$

Neural activation vector

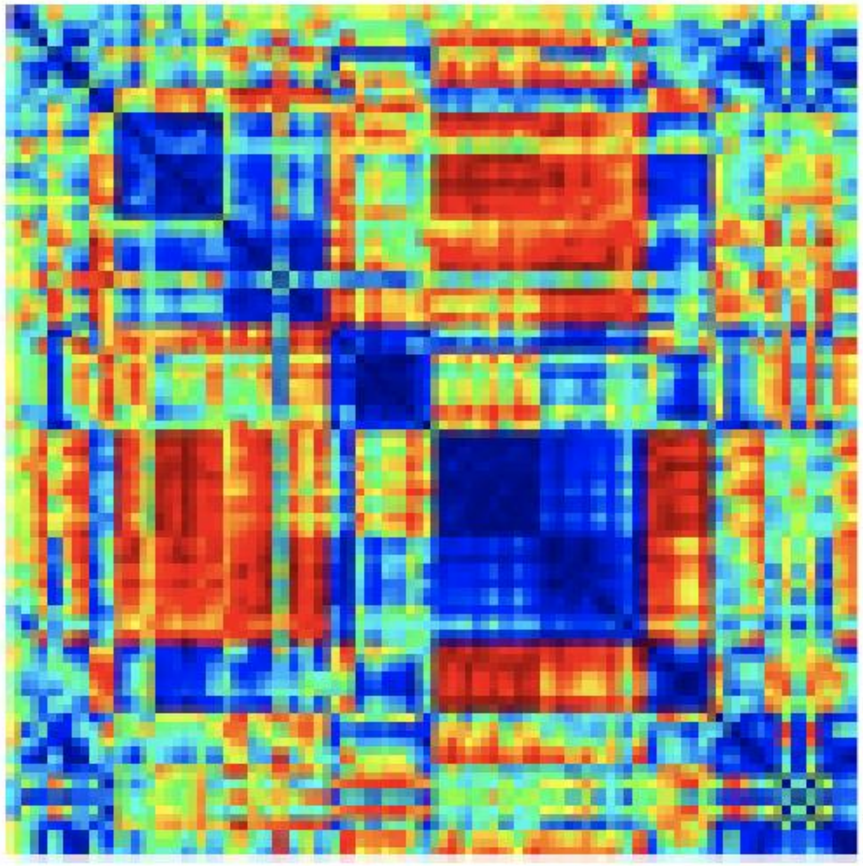
[Kriegeskorte, Mur, Ruff, et al. 2008]

Investigating a representation via similarity analysis

IT Neuronal Units



Deep net (in paricular, HMO)



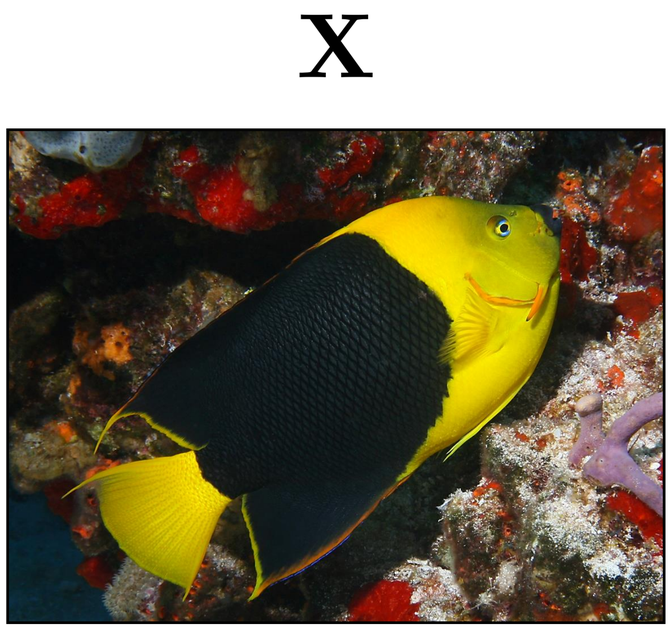
[Yamins, Hong, Cadieu, Solomon, Seibert, DiCarlo, PNAS 2014]

Investigating a representation via similarity analysis

Deep nets and the primate brain both learn similar metric spaces.

[Yamins, Hong, Cadieu, Solomon, Seibert, DiCarlo, PNAS 2014]

What do deep nets internally learn?



A CNN is a multiscale, hierarchical representation of data

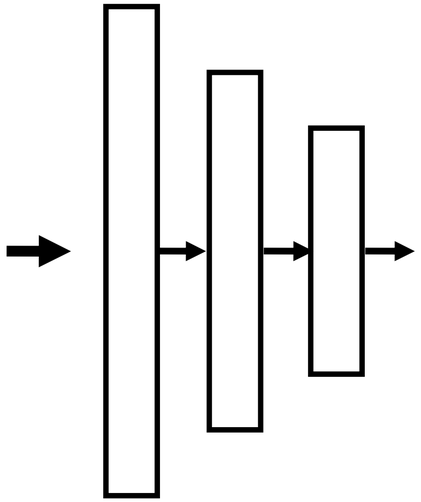
Transfer learning

“Generally speaking, a good representation is one that makes a subsequent learning task easier.” — *Deep Learning*, Goodfellow et al. 2016

?

Training

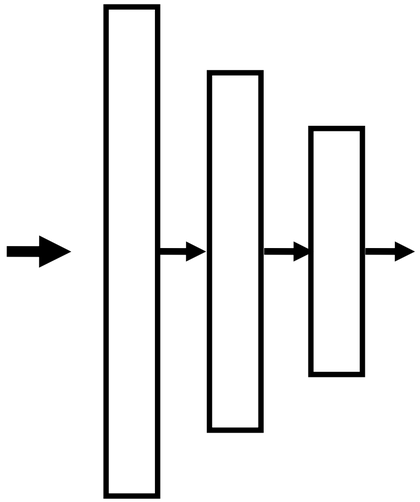
Object recognition



"Fish"

Testing

Place recognition

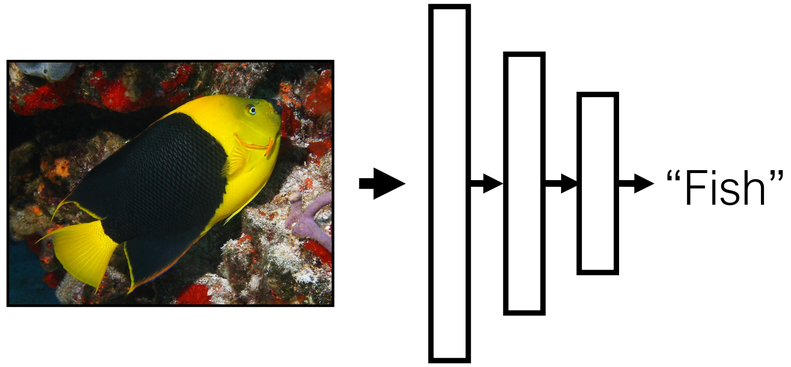


?

Often, what we will be "tested" on is to learn to do a new thing.

Pretraining

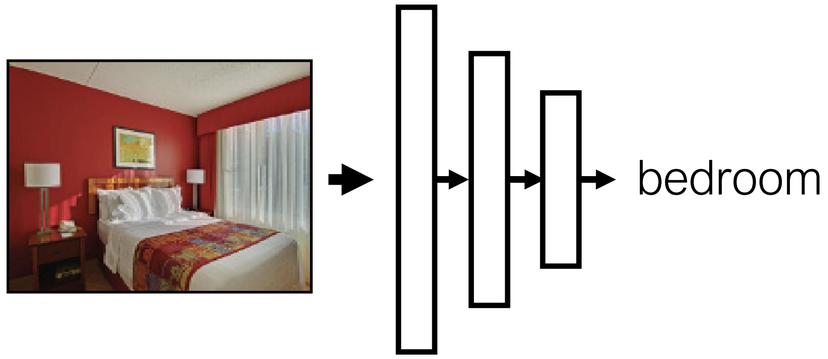
Object recognition



A lot of data

Finetuning

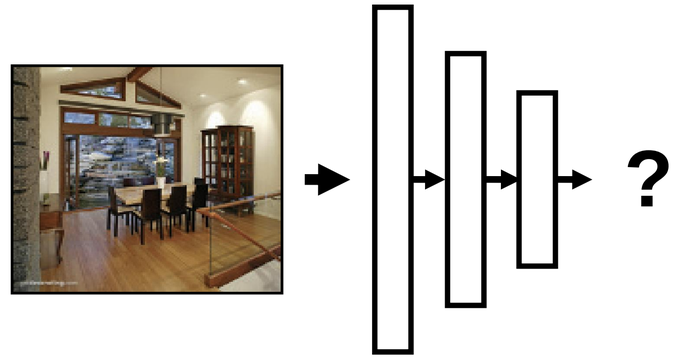
Place recognition



A little data

Testing

Place recognition



Finetuning starts with the representation learned on a previous task, and adapts it to perform well on a new task.

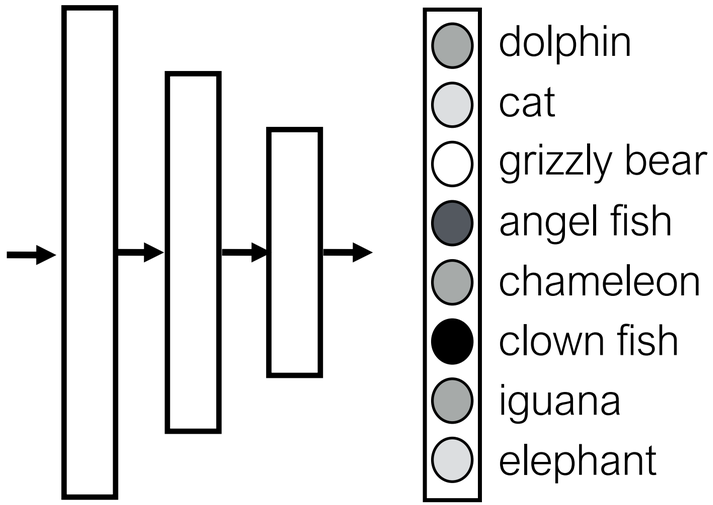
Finetuning in practice

- Pretrain a network on task A (often object recognition), resulting in parameters **W** and **b**
- Initialize a second network with some or all of **W** and **b**
- Train the second network on task B, resulting in parameters **W'** and **b'**
- **Linear Probing:** only fine-tune last layer

Finetuning in practice

Pretraining

Object recognition



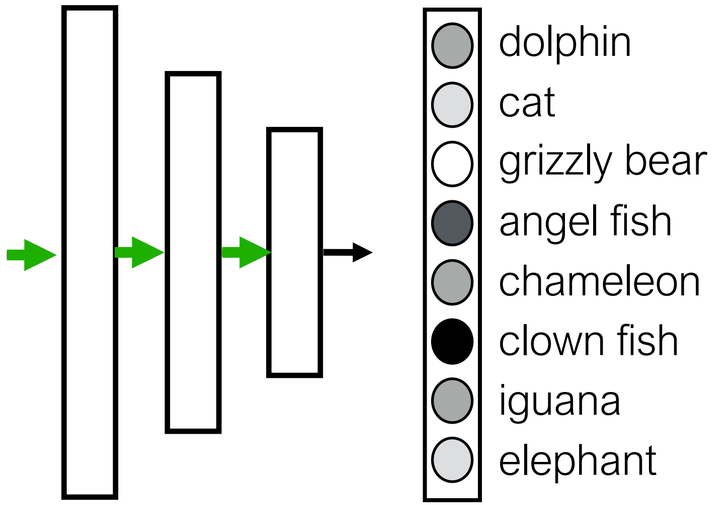
Finetuning

Place recognition

Finetuning in practice

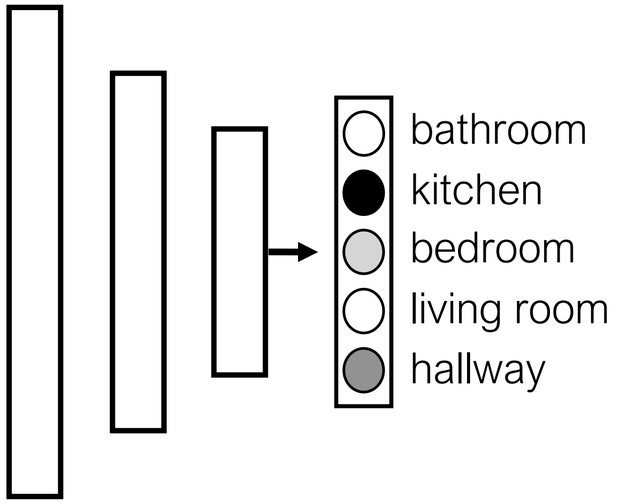
Pretraining

Object recognition



Finetuning

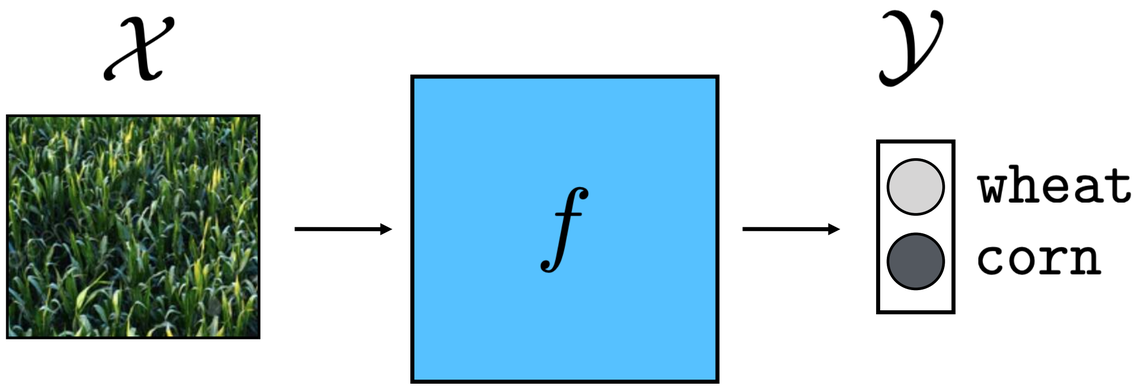
Place recognition



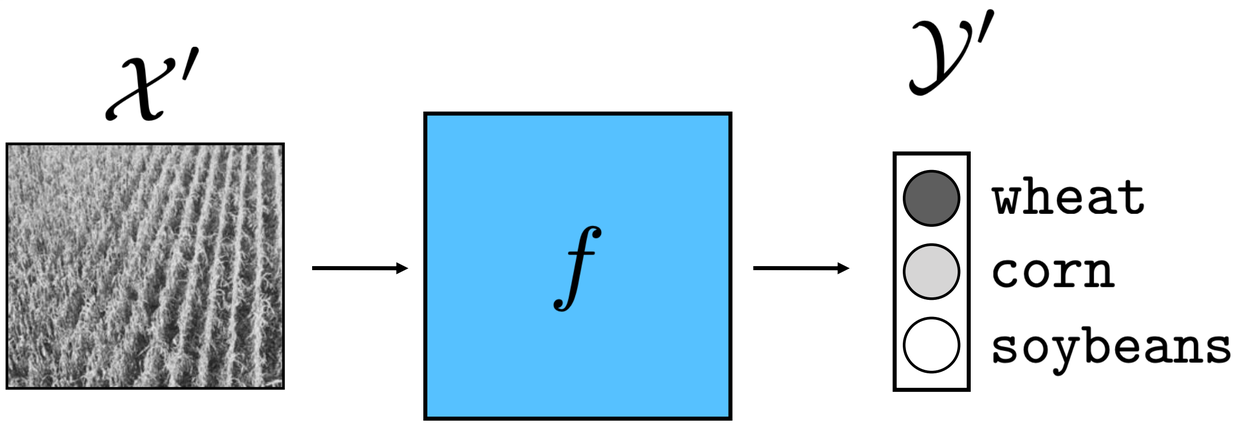
The “learned representation” is just the weights and biases, so that’s what we transfer

What if the input/output dimensions don't match?

Pretraining

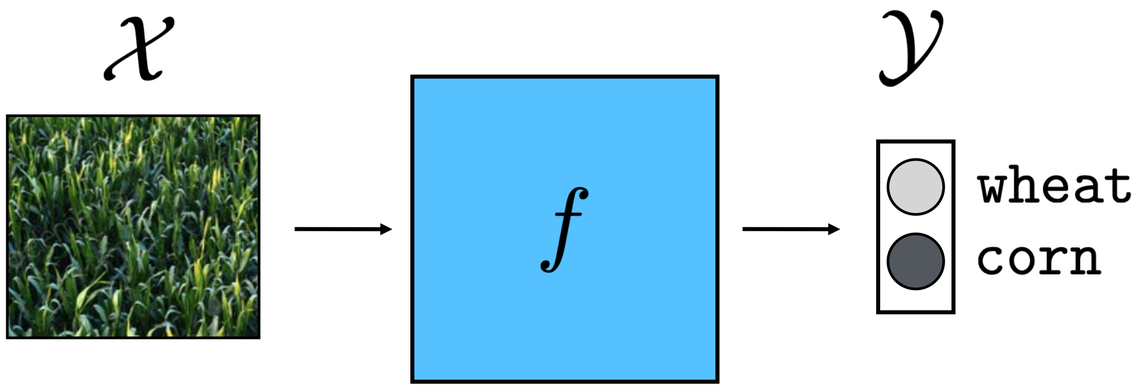


Finetuning

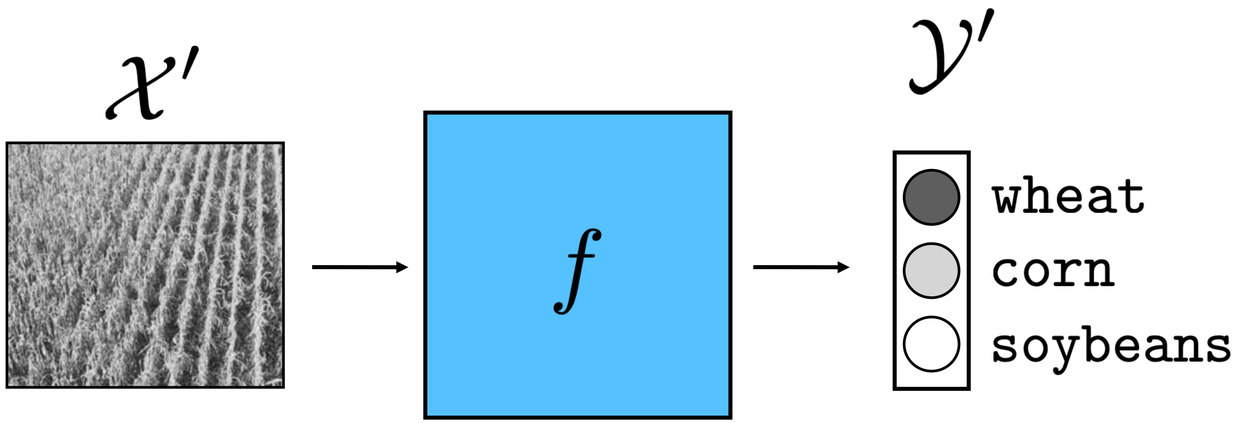


What if the input/output dimensions don't match?

Pretraining



Finetuning



$$\mathcal{X}' \neq \mathcal{X}$$

$$\mathcal{Y}' \neq \mathcal{Y}$$

What if the input/output dimensions don't match?

Pretraining

x f_1 z_1 f_2 z_2 f_3 y

wheat
corn

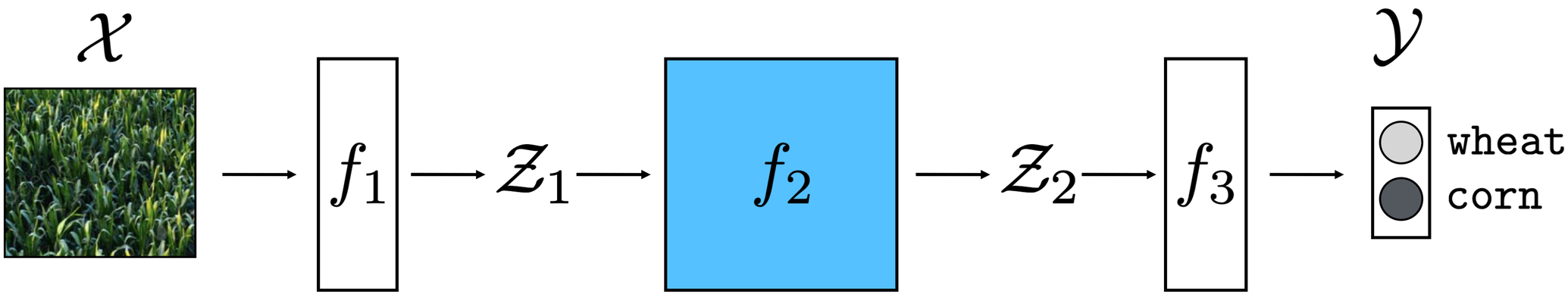
Finetuning

x' f_1' z_1 f_2 z_2 f_3' y'

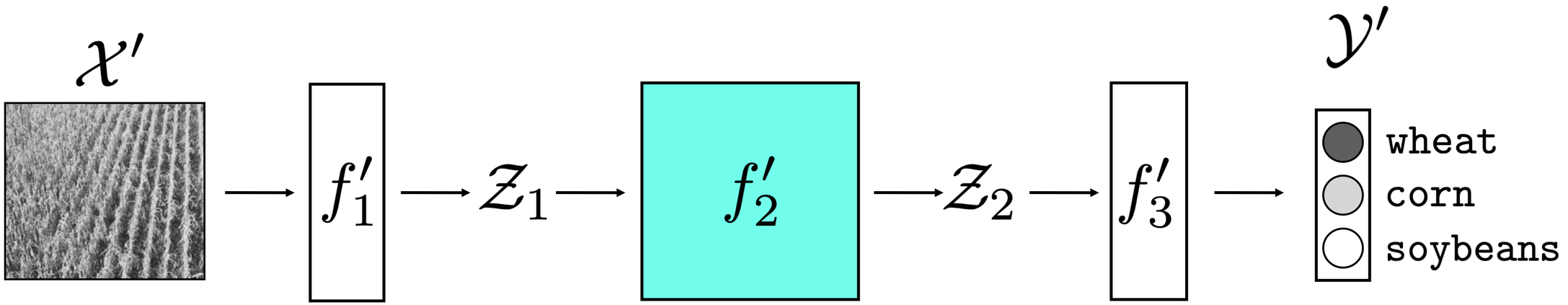
wheat
corn
soybeans

What if the input/output dimensions don't match?

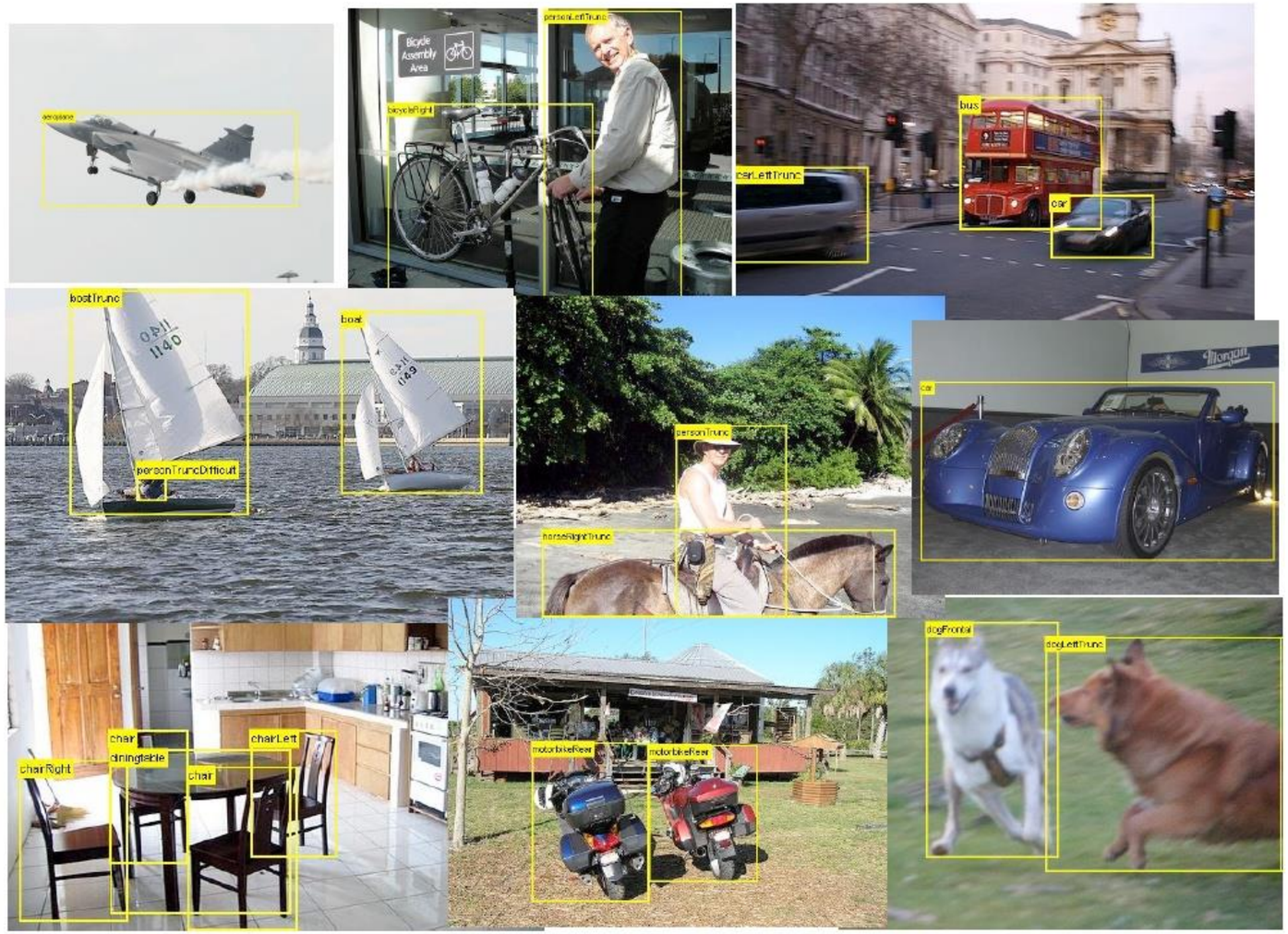
Pretraining



Finetuning



Generic object detection



Slide Credit: S. Lazebnik

PASCAL VOC Challenge (2005-2012)

- 20 challenge classes:
- *Person*
- *Animals:* bird, cat, cow, dog, horse, sheep
- *Vehicles:* aeroplane, bicycle, boat, bus, car, motorbike, train
- *Indoor:* bottle, chair, dining table, potted plant, sofa, tv/monitor
- Dataset size (by 2012): 11.5K training/validation images, 27K bounding boxes, 7K segmentations

Slide Credit: S. Lazebnik

http://host.robots.ox.ac.uk/pascal/VOC/

R-CNN, Fast R-CNN, Faster R-CNN

Rich feature hierarchies for accurate object detection and semantic segmentation

Tech report (v5)

Ross Girshick Jeff Donahue Trevor Darrell Jitendra Malik
UC Berkeley
{rbg,jdonahue,trevor,malik}@eecs.berkeley.edu

Abstract

Object detection performance, as measured on the canonical PASCAL VOC dataset, has plateaued in the last few years. The best-performing methods are complex ensemble systems that typically combine multiple low-level image features with high-level context. In this paper, we propose a simple and scalable detection algorithm that improves mean average precision (mAP) by more than 30% relative to the previous best result on VOC 2012—achieving a mAP of 53.3%. Our approach combines two key insights: (1) one can apply high-capacity convolutional neural networks (CNNs) to bottom-up region proposals in order to localize and segment objects and (2) when labeled training data is scarce, supervised pre-training for an auxiliary task, followed by domain-specific fine-tuning, yields a significant performance boost. Since we combine region proposals with CNNs, we call our method R-CNN: *Regions with CNN features. We also compare R-CNN to OverFeat, a recently proposed sliding-window detector based on a similar CNN architecture. We find that R-CNN outperforms OverFeat by a large margin on the 200-class ILSVRC2013 detection dataset. Source code for the complete system is available at* http://www.cs.berkeley.edu/~rbg/rcnn.

1. Introduction

Features matter. The last decade of progress on various visual recognition tasks has been based considerably on the use of SIFT [29] and HOG [7]. But if we look at performance on the canonical visual recognition task, PASCAL VOC object detection [15], it is generally acknowledged that progress has been slow during 2010-2012, with small gains obtained by building ensemble systems and employing minor variants of successful methods.

SIFT and HOG are blockwise orientation histograms, a representation we could associate roughly with complex cells in V1, the first cortical area in the primate visual pathway. But we also know that recognition occurs several stages downstream, which suggests that there might be hierarchical, multi-stage processes for computing features that are even more informative for visual recognition.

Fukushima's "neocognitron" [19], a biologically-inspired hierarchical and shift-invariant model for pattern recognition, was an early attempt at just such a process. The neocognitron, however, lacked a supervised training algorithm. Building on Rumelhart et al. [33], LeCun et al. [26] showed that stochastic gradient descent via back-propagation was effective for training convolutional neural networks (CNNs), a class of models that extend the neocognitron.

CNNs saw heavy use in the 1990s (e.g., [27]), but then fell out of fashion with the rise of support vector machines. In 2012, Krizhevsky et al. [25] rekindled interest in CNNs by showing substantially higher image classification accuracy on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [9, 10]. Their success resulted from training a large CNN on 1.2 million labeled images, together with a few twists on LeCun's CNN (e.g., $\max(x, 0)$ rectifying non-linearities and "dropout" regularization).

The significance of the ImageNet result was vigorously

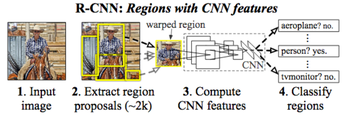


Figure 1: Object detection system overview. Our system (1) takes an input image, (2) extracts around 2000 bottom-up region proposals, (3) computes features for each proposal using a large convolutional neural network (CNN), and then (4) classifies each region using class-specific linear SVMs. R-CNN achieves a mean average precision (mAP) of **53.7% on PASCAL VOC 2010**. For comparison, [39] reports 35.1% mAP using the same region proposals, but with a spatial pyramid and bag-of-visual-words approach. The popular deformable part models perform at 33.4%. On the 200-class **ILSVRC2013 detection dataset, R-CNN's mAP is 31.4%**, a large improvement over OverFeat [34], which had the previous best result at 24.3%.

https://arxiv.org/pdf/1311.2524.pdf

Fast R-CNN

Ross Girshick
Microsoft Research
rbg@microsoft.com

Abstract

This paper proposes a Fast Region-based Convolutional Network method (Fast R-CNN) *for object detection. Fast R-CNN builds on previous work to efficiently classify object proposals using deep convolutional networks. Compared to previous work, Fast R-CNN employs several innovations to improve training and testing speed while also increasing detection accuracy. Fast R-CNN trains the very deep VGG16 network 9× faster than R-CNN, is 213× faster at test-time, and achieves a higher mAP on PASCAL VOC 2012. Compared to SPPnet, Fast R-CNN trains VGG16 3× faster, tests 10× faster, and is more accurate. Fast R-CNN is implemented in Python and C++ (using Caffe) and is available under the open-source MIT License at* https://github.com/rbgirshick/fast-rcnn.

1. Introduction

Recently, deep ConvNets [14, 16] have significantly improved image classification [14] and object detection [9, 19] accuracy. Compared to image classification, object detection is a more challenging task that requires more complex methods to solve. Due to this complexity, current approaches (*e.g.*, [9, 11, 19, 25]) train models in multi-stage pipelines that are slow and inelegant.

Complexity arises because detection requires the accurate localization of objects, creating two primary challenges. First, numerous candidate object locations (often called "proposals") must be processed. Second, these candidates provide only rough localization that must be refined to achieve precise localization. Solutions to these problems often compromise speed, accuracy, or simplicity.

In this paper, we streamline the training process for state-of-the-art ConvNet-based object detectors [9, 11]. We propose a single-stage training algorithm that jointly learns to classify object proposals and refine their spatial locations.

The resulting method can train a very deep detection network (VGG16 [20]) 9× faster than R-CNN [9] and 3× faster than SPPnet [11]. At runtime, the detection network processes images in 0.3s (excluding object proposal time) while achieving top accuracy on PASCAL VOC 2012 [7] with a mAP of 66% (vs. 62% for R-CNN).[1]

1.1. R-CNN and SPPnet

The Region-based Convolutional Network method (R-CNN) [9] achieves excellent object detection accuracy by using a deep ConvNet to classify object proposals. R-CNN, however, has notable drawbacks:

1. **Training is a multi-stage pipeline.** R-CNN first fine-tunes a ConvNet on object proposals using log loss. Then, it fits SVMs to ConvNet features. These SVMs act as object detectors, replacing the softmax classifier learnt by fine-tuning. In the third training stage, bounding-box regressors are learned.
2. **Training is expensive in space and time.** For SVM and bounding-box regressor training, features are extracted from each object proposal in each image and written to disk. With very deep networks, such as VGG16, this process takes 2.5 GPU-days for the 5k images of the VOC07 trainval set. These features require hundreds of gigabytes of storage.
3. **Object detection is slow.** At test-time, features are extracted from each object proposal in each test image. Detection with VGG16 takes 47s / image (on a GPU).

R-CNN is slow because it performs a ConvNet forward pass for each object proposal, without sharing computation. Spatial pyramid pooling networks (SPPnets) [11] were proposed to speed up R-CNN by sharing computation. The SPPnet method computes a convolutional feature map for the entire input image and then classifies each object proposal using a feature vector extracted from the shared feature map. Features are extracted for a proposal by max-pooling the portion of the feature map inside the proposal into a fixed-size output (*e.g.*, 6×6). Multiple output sizes are pooled and then concatenated as in spatial pyramid pooling [15]. SPPnet accelerates R-CNN by 10 to 100× at test time. Training time is also reduced by 3× due to faster proposal feature extraction.

[1] All timings use one Nvidia K40 GPU overclocked to 875 MHz.

https://arxiv.org/pdf/1504.08083.pdf

Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun

Abstract—State-of-the-art object detection networks depend on region proposal algorithms to hypothesize object locations. Advances like SPPnet [1] and Fast R-CNN [2] have reduced the running time of these detection networks, exposing region proposal computation as a bottleneck. In this work, we introduce a *Region Proposal Network* (RPN) that shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals. An RPN is a fully convolutional network that simultaneously predicts object bounds and objectness scores at each position. The RPN is trained end-to-end to generate high-quality region proposals, which are used by Fast R-CNN for detection. We further merge RPN and Fast R-CNN into a single network by sharing their convolutional features—using the recently popular terminology of neural networks with "attention" mechanisms, the RPN component tells the unified network where to look. For the very deep VGG-16 model [3], our detection system has a frame rate of 5fps (*including all steps*) on a GPU, while achieving state-of-the-art object detection accuracy on PASCAL VOC 2007, 2012, and MS COCO datasets with only 300 proposals per image. In ILSVRC and COCO 2015 competitions, Faster R-CNN and RPN are the foundations of the 1st-place winning entries in several tracks. Code has been made publicly available.

Index Terms—Object Detection, Region Proposal, Convolutional Neural Network.

1 Introduction

Recent advances in object detection are driven by the success of region proposal methods (*e.g.*, [4]) and region-based convolutional neural networks (R-CNNs) [5]. Although region-based CNNs were computationally expensive as originally developed in [5], their cost has been drastically reduced thanks to sharing convolutions across proposals [1], [2]. The latest incarnation, Fast R-CNN [2], achieves near real-time rates using very deep networks [3], *when ignoring the time spent on region proposals*. Now, proposals are the test-time computational bottleneck in state-of-the-art detection systems.

Region proposal methods typically rely on inexpensive features and economical inference schemes. Selective Search [4], one of the most popular methods, greedily merges superpixels based on engineered low-level features. Yet when compared to efficient detection networks [2], Selective Search is an order of magnitude slower, at 2 seconds per image in a CPU implementation. EdgeBoxes [6] currently provides the best tradeoff between proposal quality and speed, at 0.2 seconds per image. Nevertheless, the region proposal step still consumes as much running time as the detection network.

- *S. Ren is with University of Science and Technology of China, Hefei, China. This work was done when S. Ren was an intern at Microsoft Research. Email: sqren@mail.ustc.edu.cn*
- *K. He and J. Sun are with Visual Computing Group, Microsoft Research. E-mail: {kahe,jiansun}@microsoft.com*
- *R. Girshick is with Facebook AI Research. The majority of this work was done when R. Girshick was with Microsoft Research. E-mail: rbg@fb.com*

One may note that fast region-based CNNs take advantage of GPUs, while the region proposal methods used in research are implemented on the CPU, making such runtime comparisons inequitable. An obvious way to accelerate proposal computation is to re-implement it for the GPU. This may be an effective engineering solution, but re-implementation ignores the down-stream detection network and therefore misses important opportunities for sharing computation.

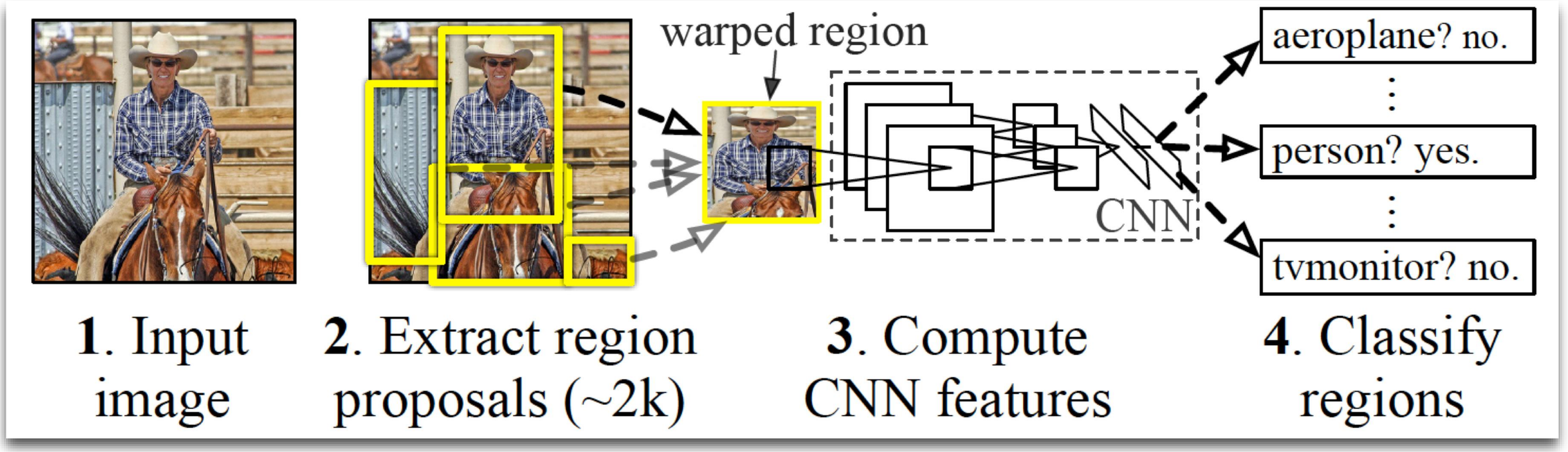
In this paper, we show that an algorithmic change—computing proposals with a deep convolutional neural network—leads to an elegant and effective solution where proposal computation is nearly cost-free given the detection network's computation. To this end, we introduce novel *Region Proposal Networks* (RPNs) that share convolutional layers with state-of-the-art object detection networks [1], [2]. By sharing convolutions at test-time, the marginal cost for computing proposals is small (*e.g.*, 10ms per image).

Our observation is that the convolutional feature maps used by region-based detectors, like Fast R-CNN, can also be used for generating region proposals. On top of these convolutional features, we construct an RPN by adding a few additional convolutional layers that simultaneously regress region bounds and objectness scores at each location on a regular grid. The RPN is thus a kind of fully convolutional network (FCN) [7] and can be trained end-to-end specifically for the task for generating detection proposals.

RPNs are designed to efficiently predict region proposals with a wide range of scales and aspect ratios. In contrast to prevalent methods [8], [9], [1], [2] that use

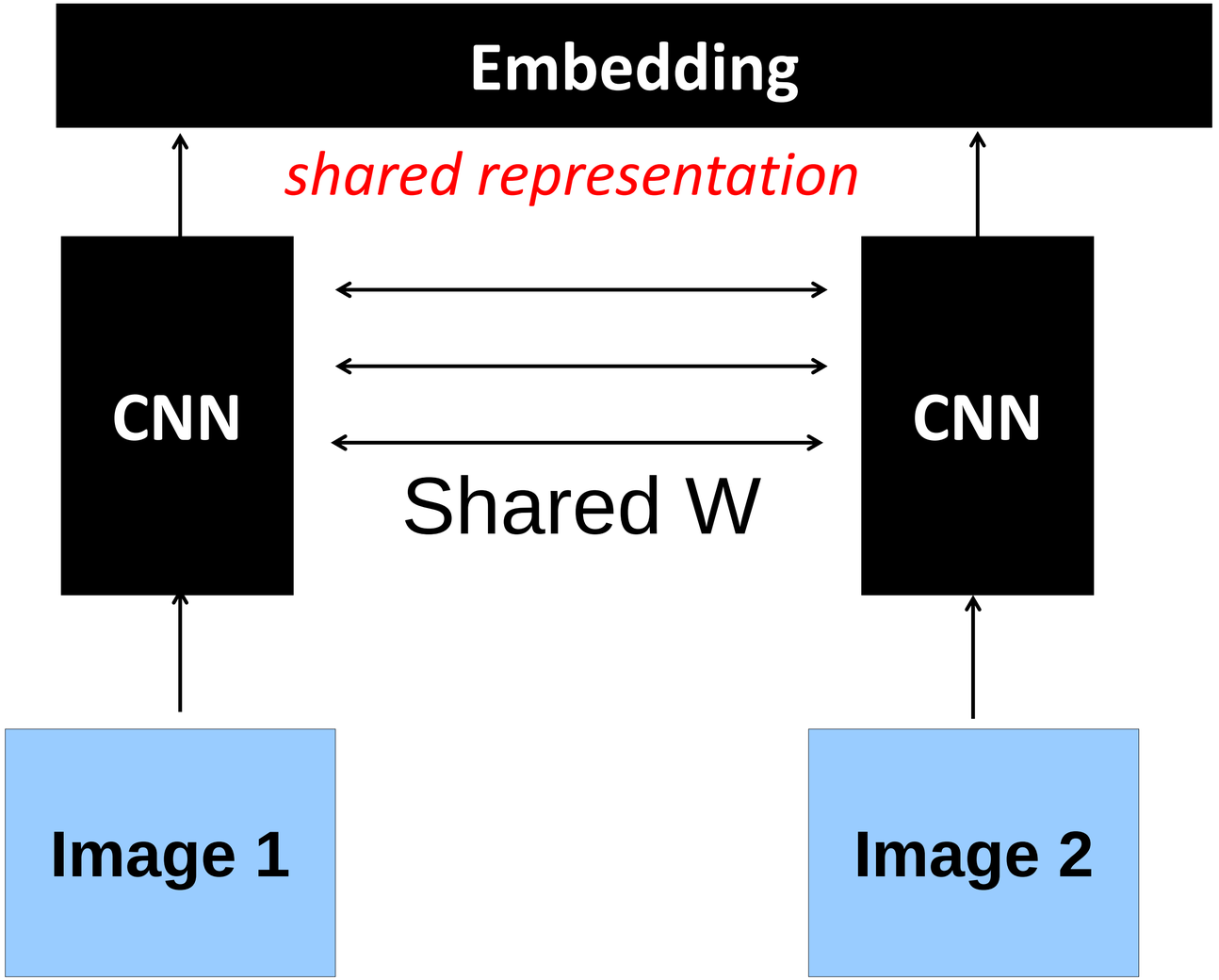
https://arxiv.org/pdf/1506.01497.pdf

R-CNN

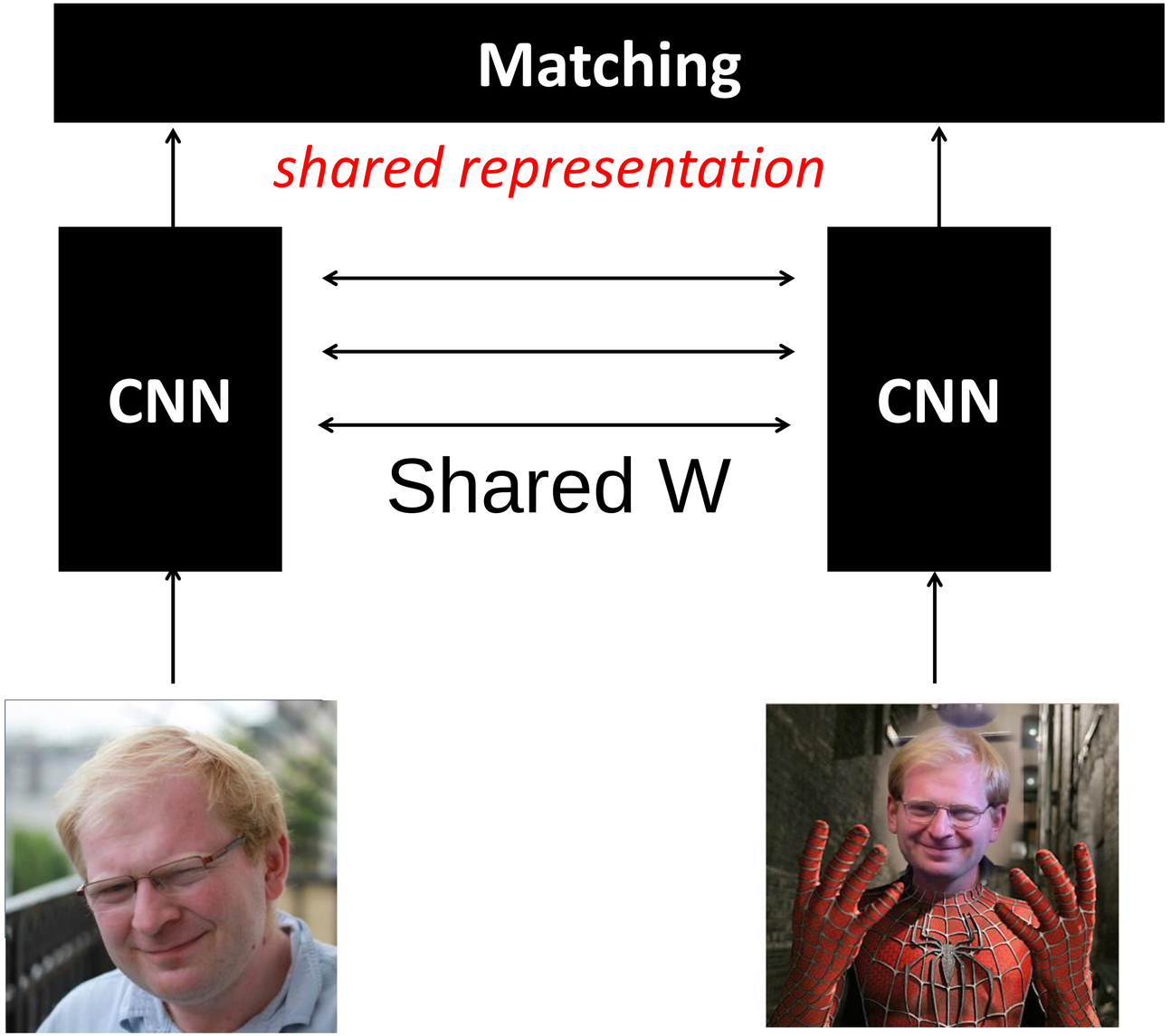


Girshick, R., J. Donahue, T. Darrell, and J. Malik. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation." *CVPR '14 Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*. Pages 580-587. 2014

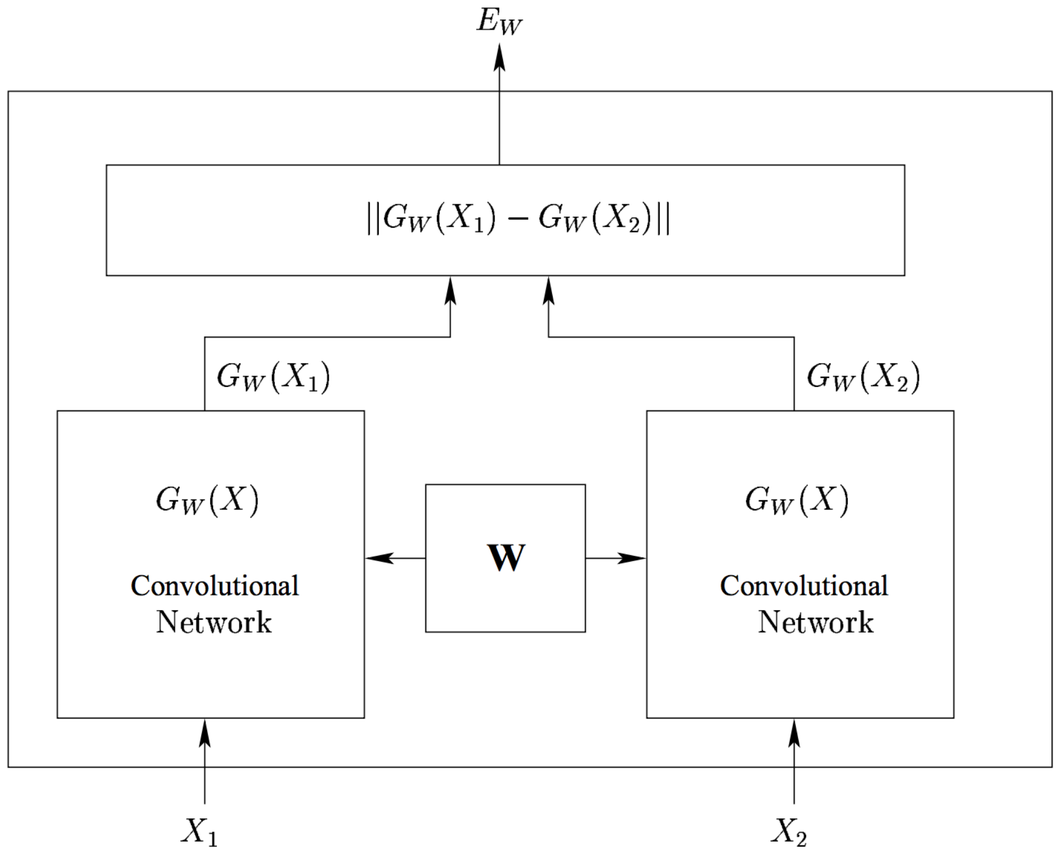
Directly training the embedding

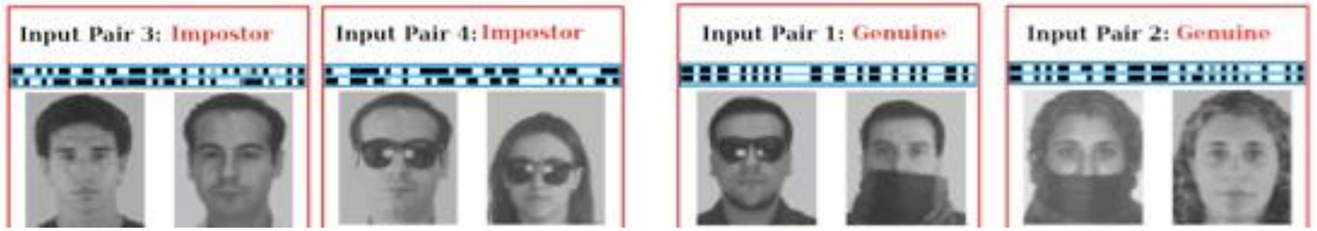


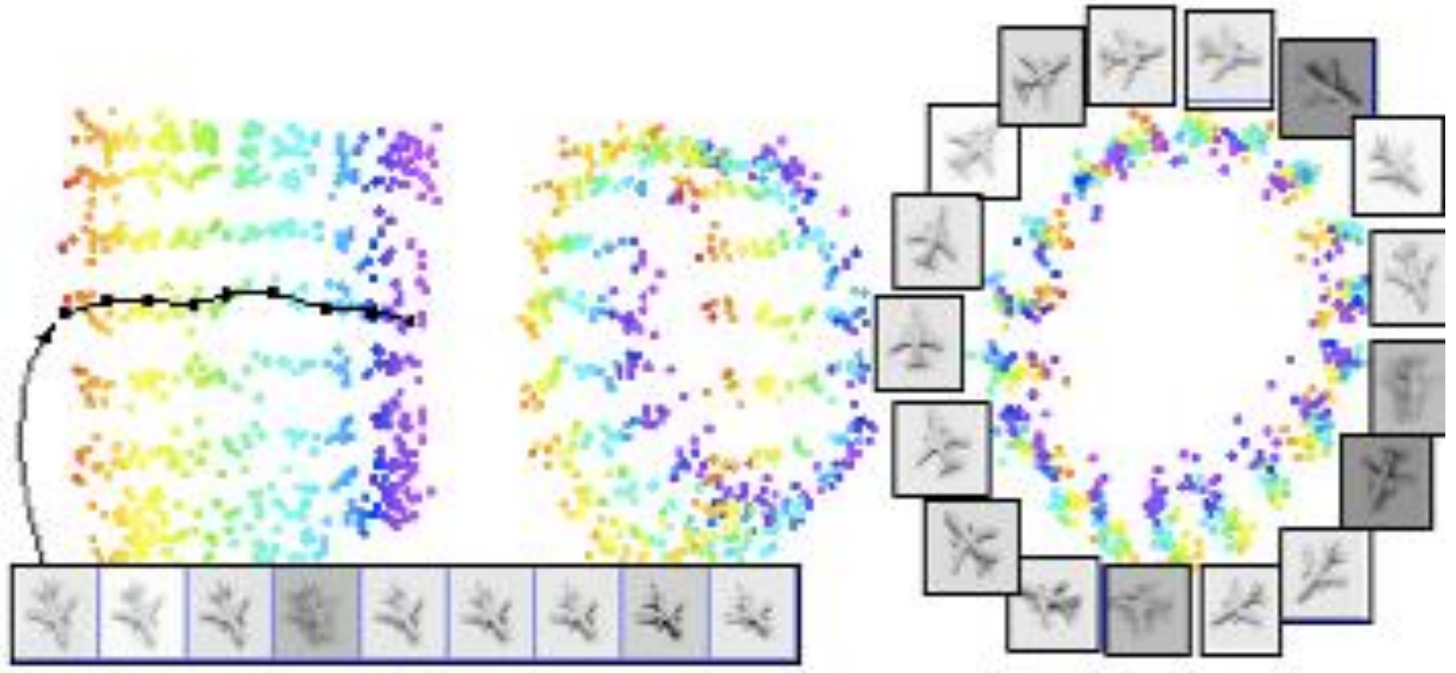
e.g. pairwise training signal



Siamese Networks with Contrastive Loss







Siamese Architecture
[Chopra 2005, Hadsell 2006]

LEARNING VISUAL SIMILARITY FOR PRODUCT DESIGN WITH CONVOLUTIONAL NEURAL NETWORKS

SEAN BELL AND KAVITA BALA
CORNELL UNIVERSITY

THE PROBLEM

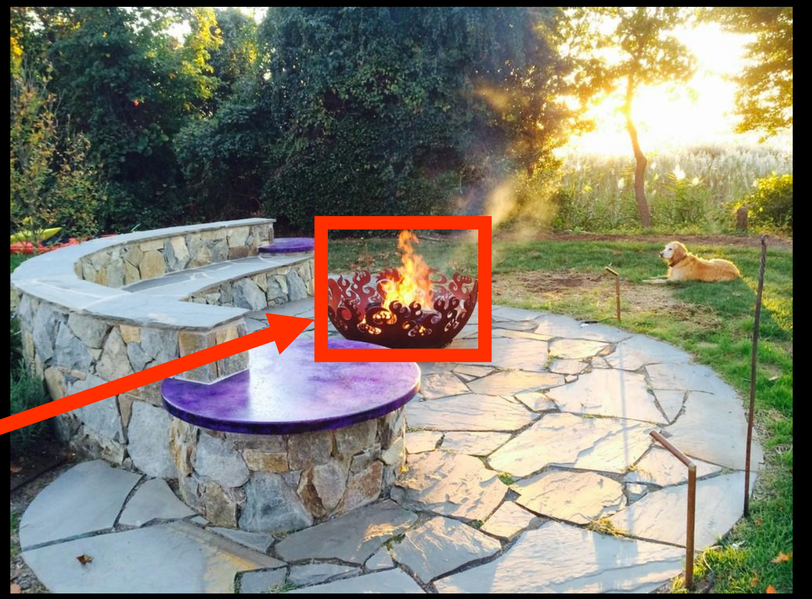
(1) “What is this?”

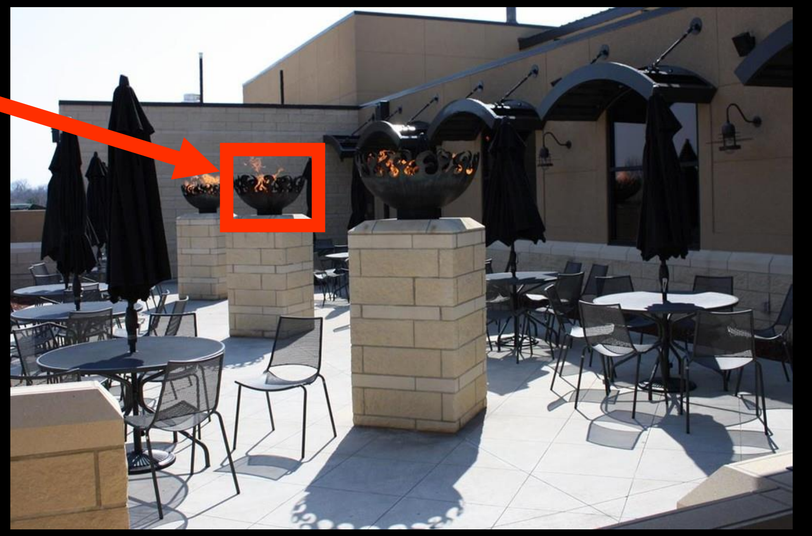
(2) “Where is it used?”

Name: ”Great Bowl O’Fire Sculptural Fire Bowl”

Category: Fire pit

Sold by: John T. Unger, LLC





THE PROBLEM

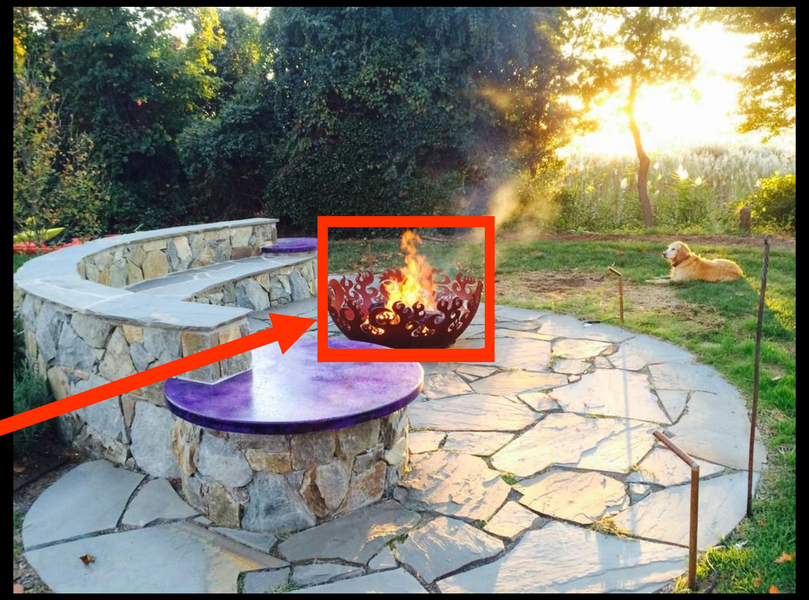
(1) “What is this?”

(2) “Where is it used?”

Name: ”Great Bowl O’Fire Sculptural Fire Bowl”

Category: Fire pit

Sold by: John T. Unger, LLC



Challenge: determine whether these are the same product (different resolution, viewpoint, color, lighting, occlusions)

TWO KINDS OF IMAGES

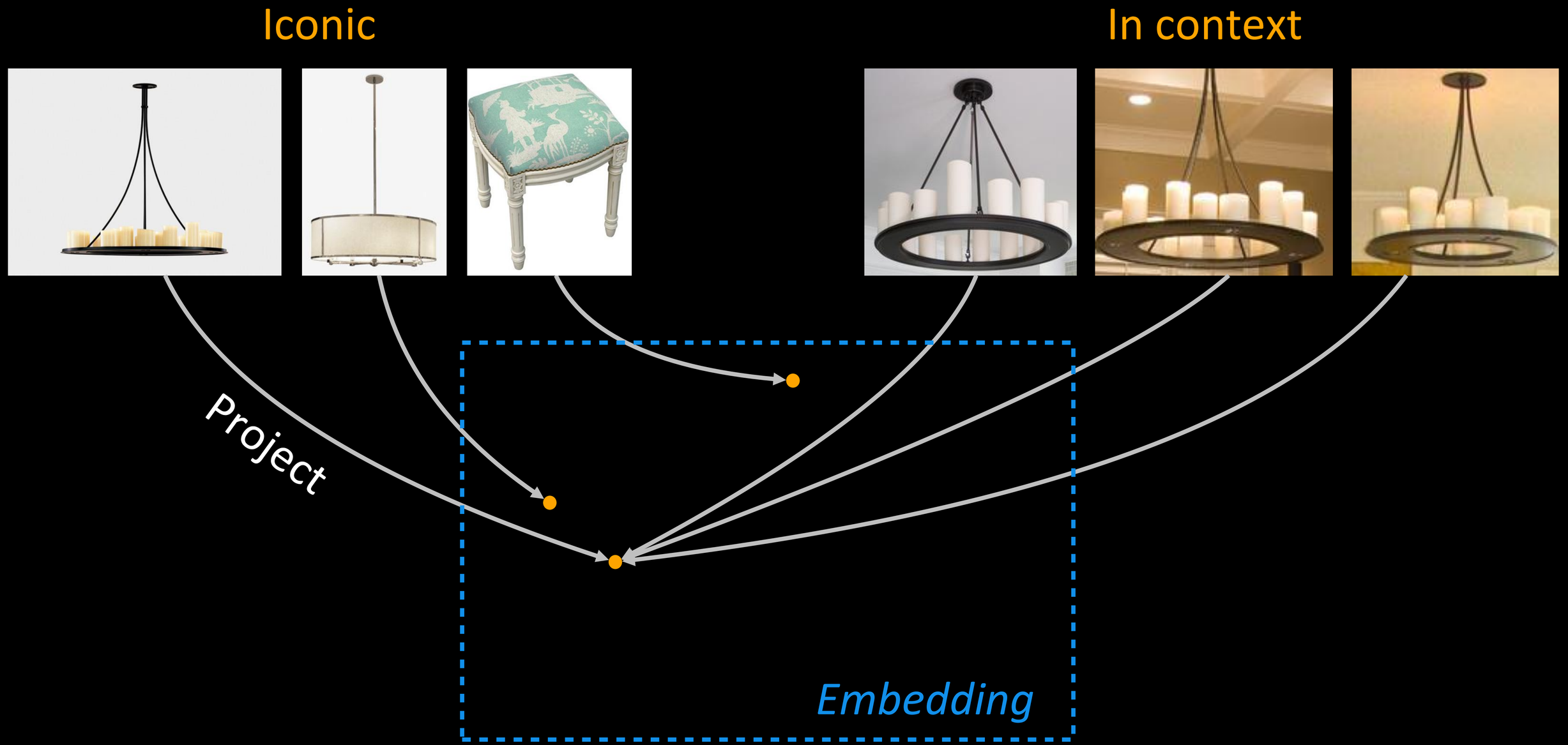
Iconic

(From a product website)

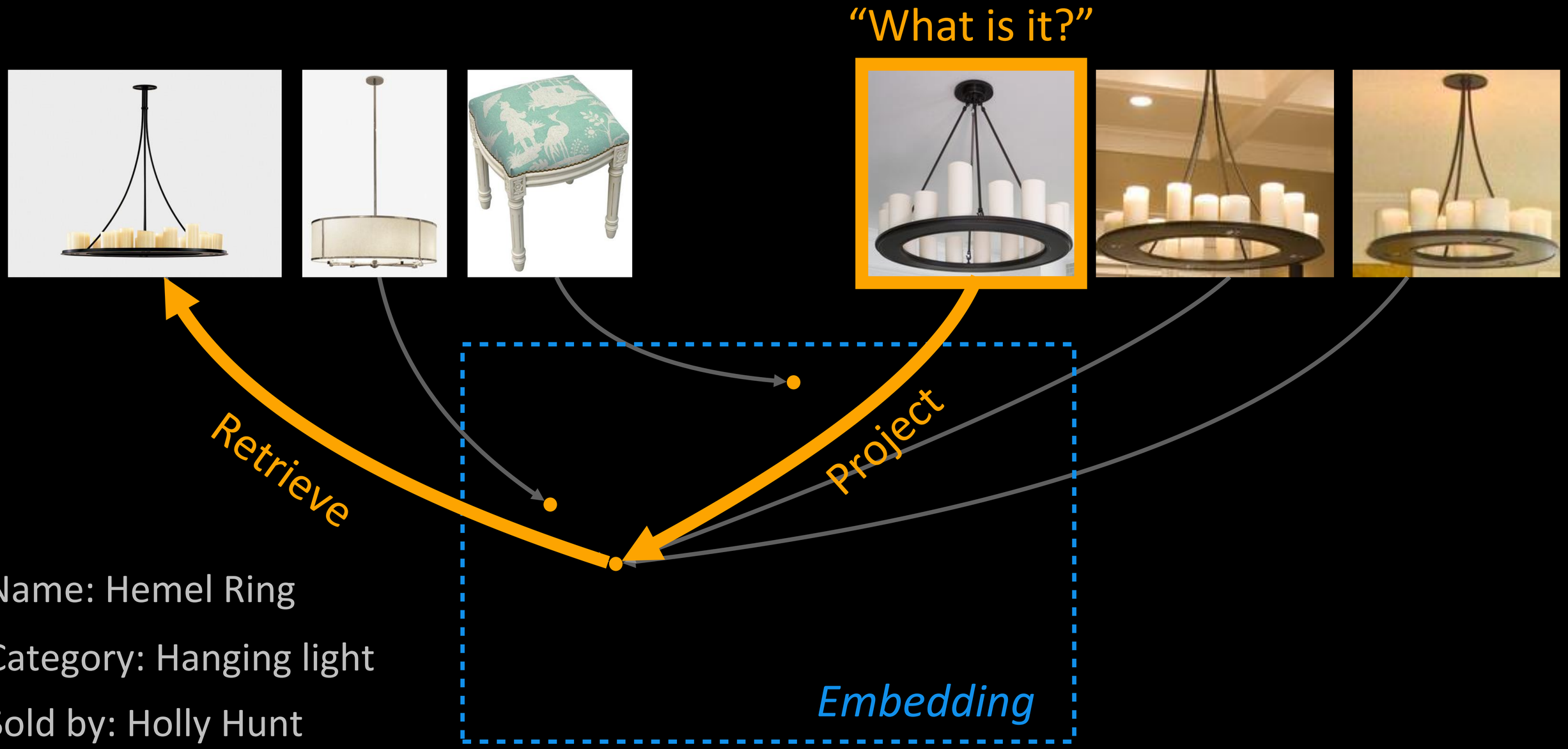
In context

(Cropped from a scene photo)

PROJECTING INTO A JOINT EMBEDDING

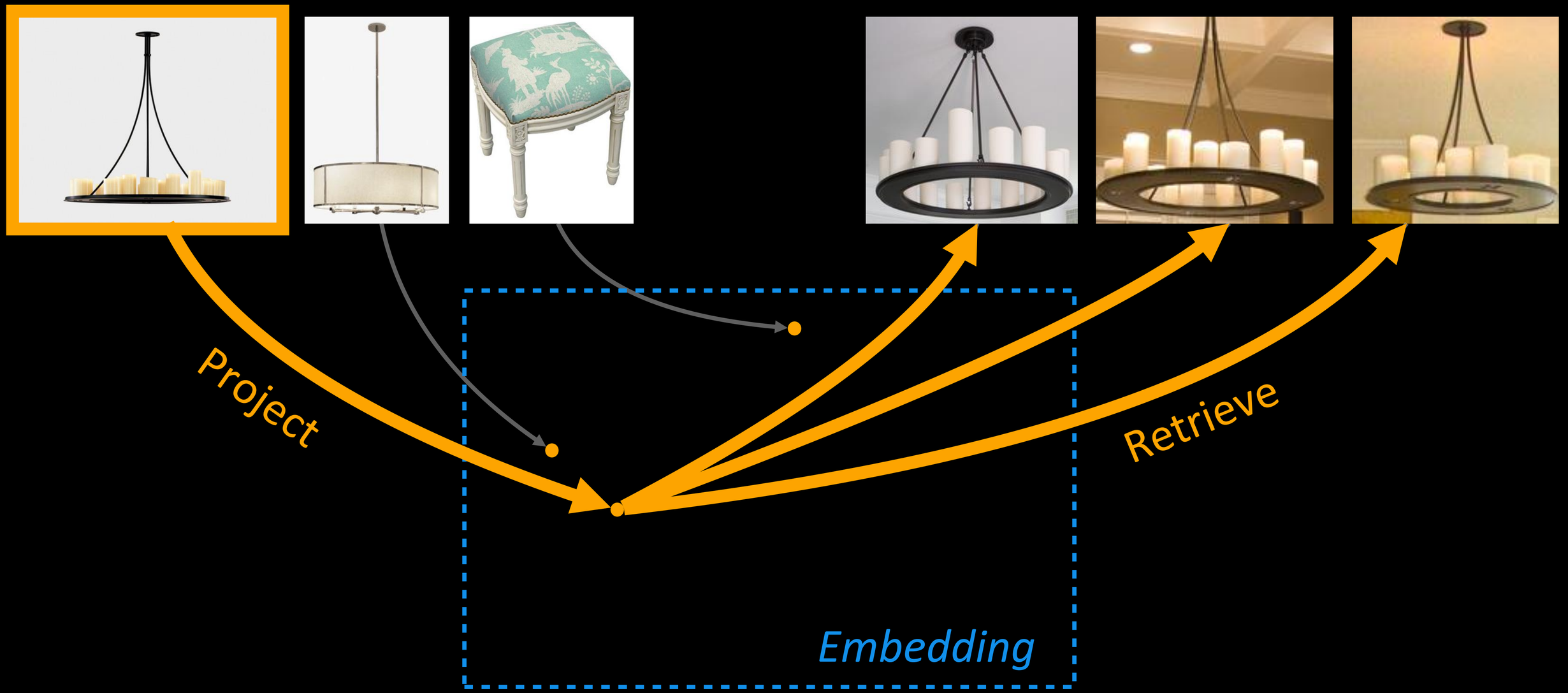


SEARCH USING THE EMBEDDING

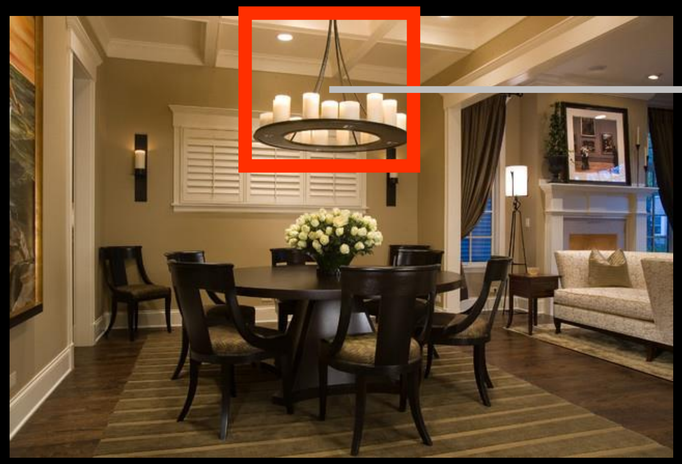


SEARCH USING THE EMBEDDING

"Where is it used?"



CONTRASTIVE LOSS: POSITIVE EXAMPLE



In context

Iconic (same)

CNN

Parameters θ

CNN

x_q

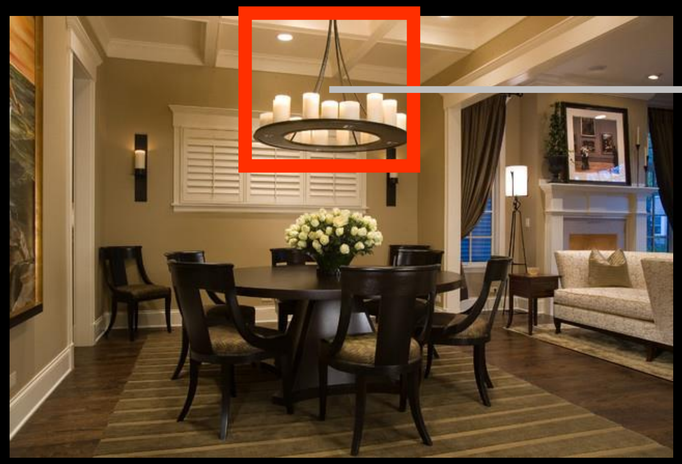
Loss L_p

x_p

Embedding

$$L_p(x_q, x_p) = ||x_q - x_p||_2^2$$

CONTRASTIVE LOSS: NEGATIVE EXAMPLE



In context

Iconic (different)

CNN

Parameters θ

CNN

Loss L_n

x_q

x_n

Margin *m*

Embedding

$$L_n(x_q, x_n) = \max\left(0, m^2 - ||x_q - x_n||_2^2\right)$$

CONTRASTIVE (TRIPLET) LOSS: ALL TOGETHER

$$L(\theta) = \underbrace{\sum_{(x_q, x_p)} L_p(x_q, x_p)}_{\text{Penalty for similar images that are far away}} + \underbrace{\sum_{(x_q, x_n)} L_n(x_q, x_n)}_{\text{Penalty for dissimilar images that are nearby}}$$

$$L_p(x_q, x_p) = ||x_q - x_p||_2^2$$

$$L_n(x_q, x_n) = \max\left(0, m^2 - ||x_q - x_n||_2^2\right)$$

Margin

Minimize L(θ) with stochastic gradient descent and momentum

[Chopra 2005, Hadsell 2006]

RESULTS: “WHAT IS IT?”

In context

RESULTS: “WHAT IS IT?”

In context

Iconic

Louis Ghost Chair with Arms

Living Room | Contemporary | 1 | View jpg | Outdoor Lounge Chairs | Eclectic | 44 | View jpg

2 boxes

Top 4 results:

Victoria Ghost Chair by Philippe Starck for Kartell | Design Within Reach

Dining Chairs | Modern | 15 | View jpg | View detail

0.019

Kartell | Victoria Ghost Chair, Set of 4

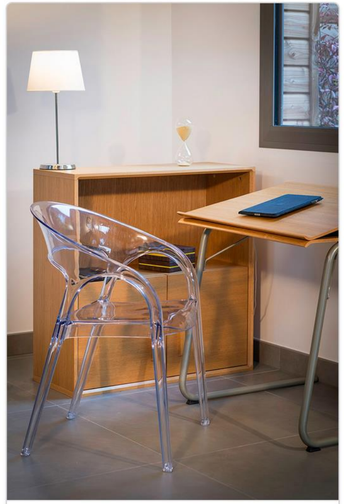
Dining Chairs | Modern | View jpg | View detail

0.023

Kartell | Louis Ghost Chair

Dining Chairs | Contemporary | View jpg | View detail

0.024



Chaises

Office Chairs | Unknown style | View jpg | View detail

0.025

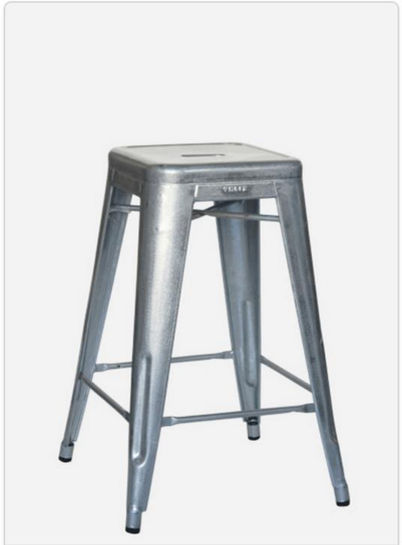
RESULTS: “WHAT IS IT?”

In context

RESULTS: “WHAT IS IT?”

In context

Iconic



Tolix Stool “Tabouret”

Kitchen | Contemporary | 1 | View jpg | Bar Stools And Counter Stools | Industrial | 305 | View jpg

2 boxes

Top 4 results:

East Berlin District Metal Barstool

Serving Dishes And Platters | Modern | View jpg | View detail

0.046

Amelia Metal Cafe Barstool in Orange - Set of

Bar Stools And Counter Stools | Contemporary | 3 | View jpg | View detail

0.049

Amelia Metal Cafe Barstool (Set of 2), Red

Bar Stools And Counter Stools | Traditional | 2 | View jpg | View detail

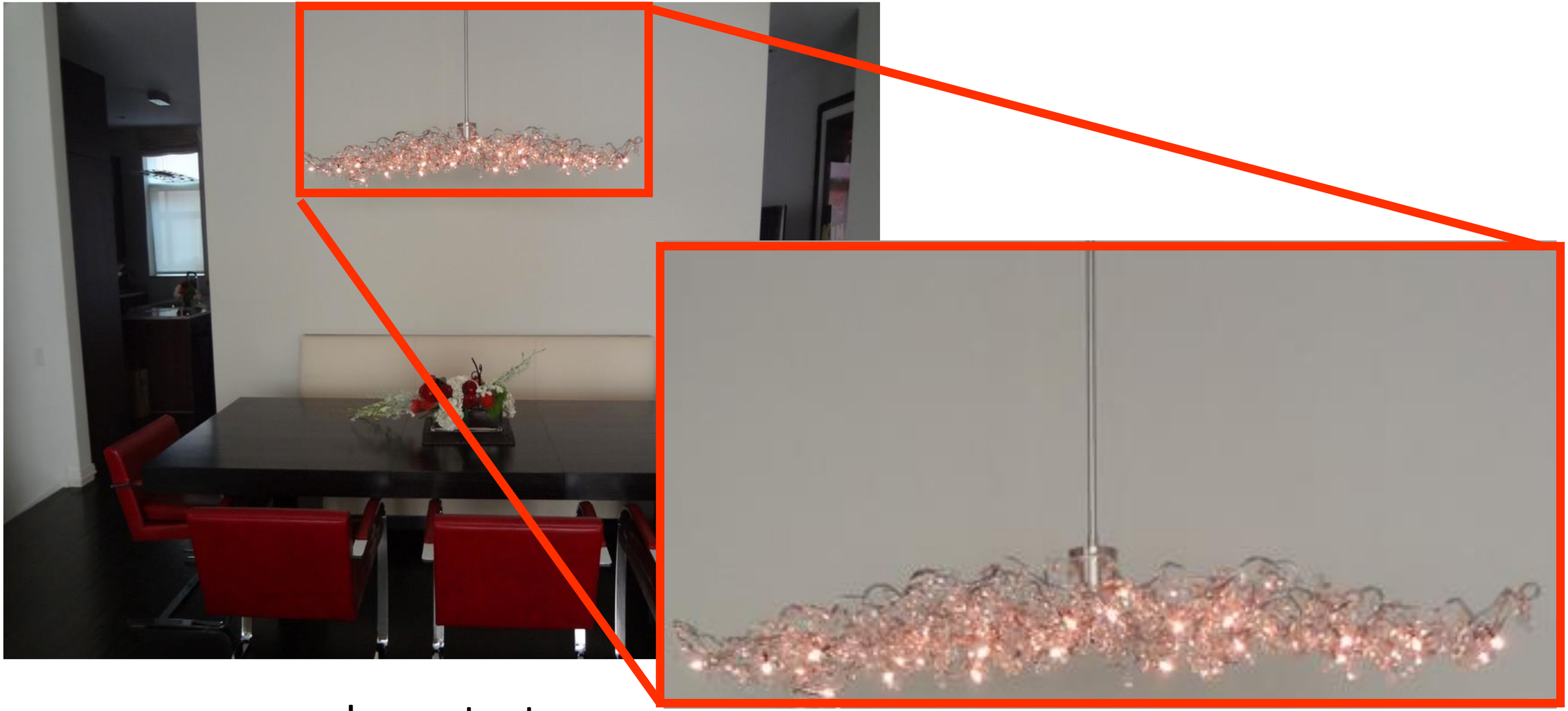
0.050

Winsome Dubliner 30 in. Bar Stool - Set of 2

Bar Stools And Counter Stools | Modern | View jpg | View detail

0.050

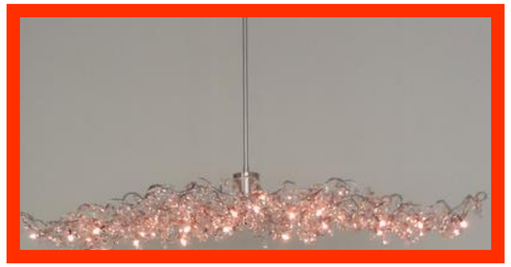
RESULTS: “WHAT IS IT?”



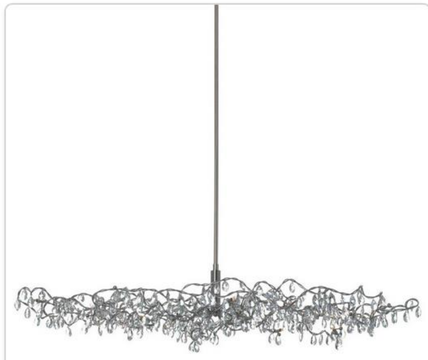
In context

RESULTS: “WHAT IS IT?”

In context



Iconic



Tiara Oval Suspension by Harco Loor

Dining Room | Contemporary | 1 | View jpg | Wall Sconces | Unknown style | 1 | View jpg

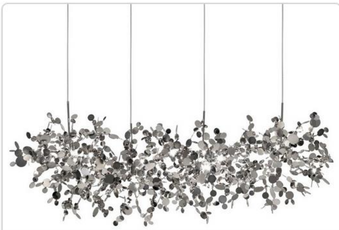
2 boxes

Top 4 results:

HARCO LOOR Tiara Chandelier

Chandeliers | Contemporary | 1 | View jpg | View detail

0.035



Argent N92S Suspension Light

Bathroom Vanity Lighting | Modern | View jpg | View detail

0.038

Eurofase 25620-016 Divo 9 Light Pendant in Nickel 25620-016

Pendant Lighting | Modern | View jpg | View detail

0.041

Eurofase 25618-013 Divo 6 Light Pendant in Nickel 25618-013

Pendant Lighting | Modern | View jpg | View detail

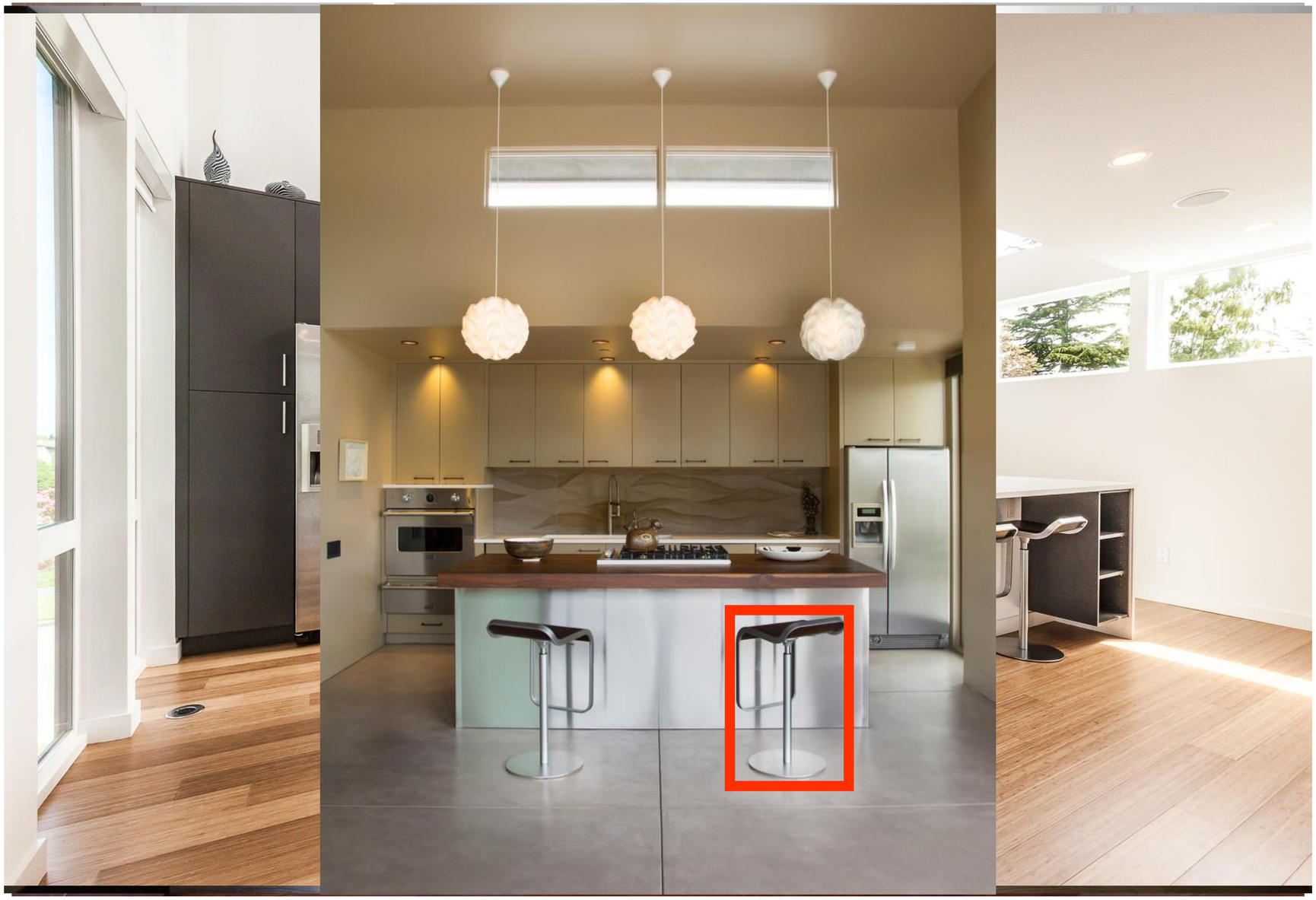
0.047

RESULTS: “WHERE IS IT USED?”

“Maskros
Pendant Lamp”

RESULTS: “WHERE IS IT USED?”

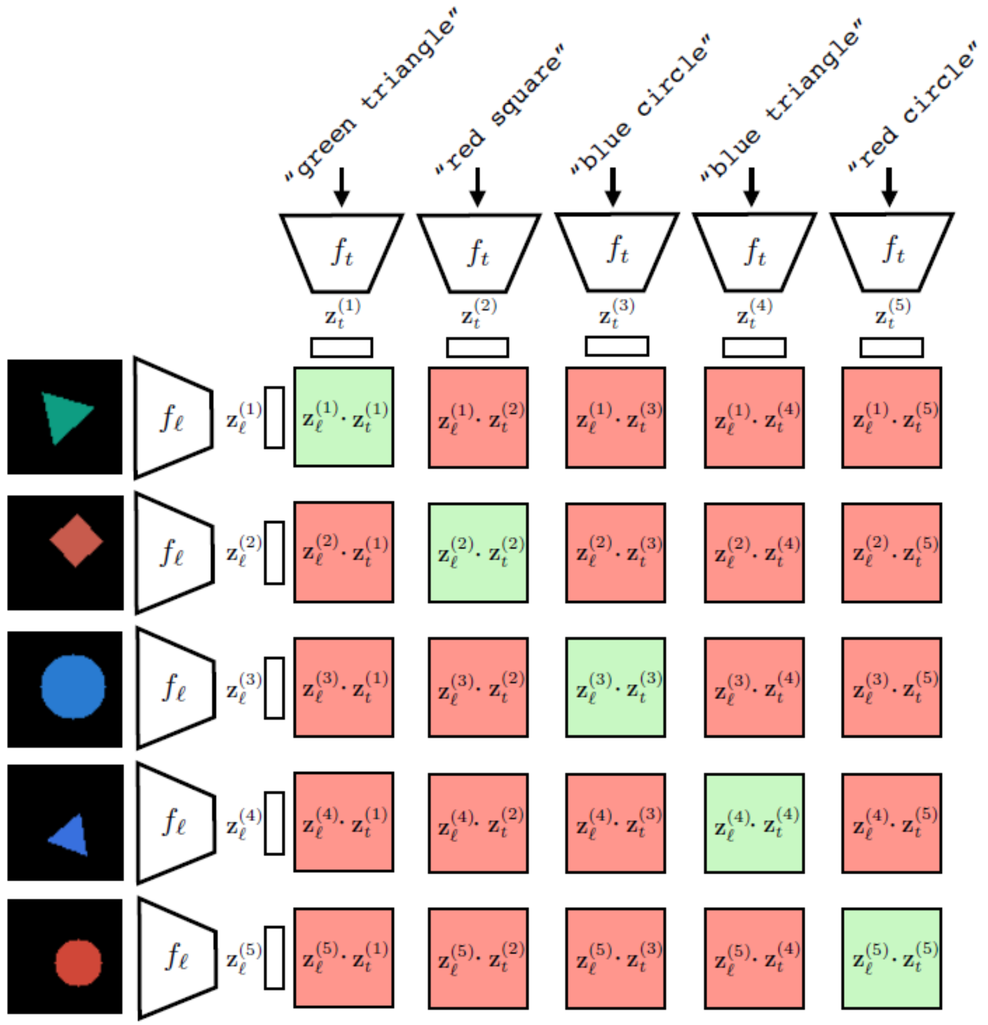
"LEM Piston Stool | Design Within Reach”



SEARCHING ACROSS CATEGORIES

Contrastive Language Image Pretraining (CLIP)

InfoNCE loss, a classification version of Triplet loss



A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. “Learning Transferable Visual Models from Natural Language Supervision.” In: International Conference on Machine Learning. 2021

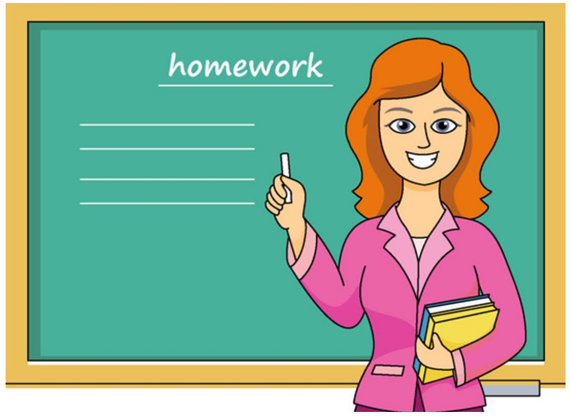
Supervised computer vision

Hand-curated training data

+ Informative

- Expensive

- Limited to teacher's knowledge



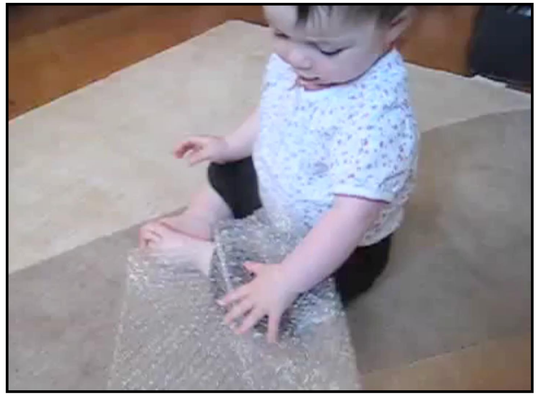
Vision in nature

Raw unlabeled training data

+ Cheap

- Noisy

- Harder to interpret



Learning from examples

(aka **supervised learning**)

Training data

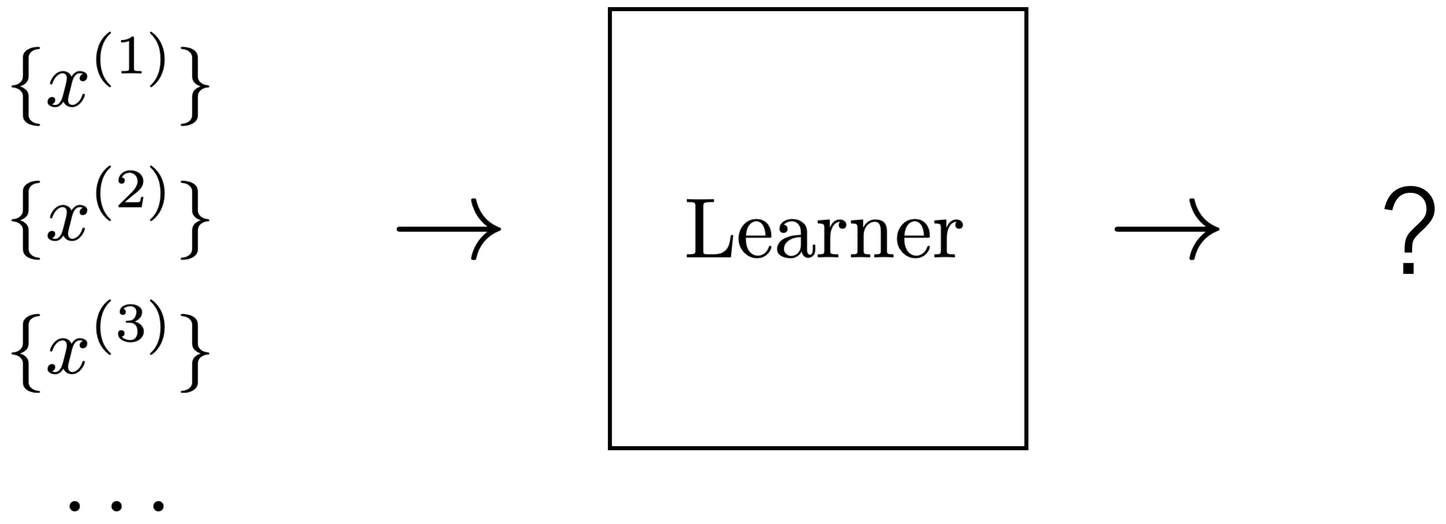
$$\{x^{(1)}, y^{(1)}\}$$
$$\{x^{(2)}, y^{(2)}\} \quad \rightarrow \quad \boxed{\text{Learner}} \quad \rightarrow \quad f : X \rightarrow Y$$
$$\{x^{(3)}, y^{(3)}\}$$
$$\ldots$$

$$f^* = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} \mathcal{L}(f(\mathbf{x}^{(i)}), \mathbf{y}^{(i)})$$

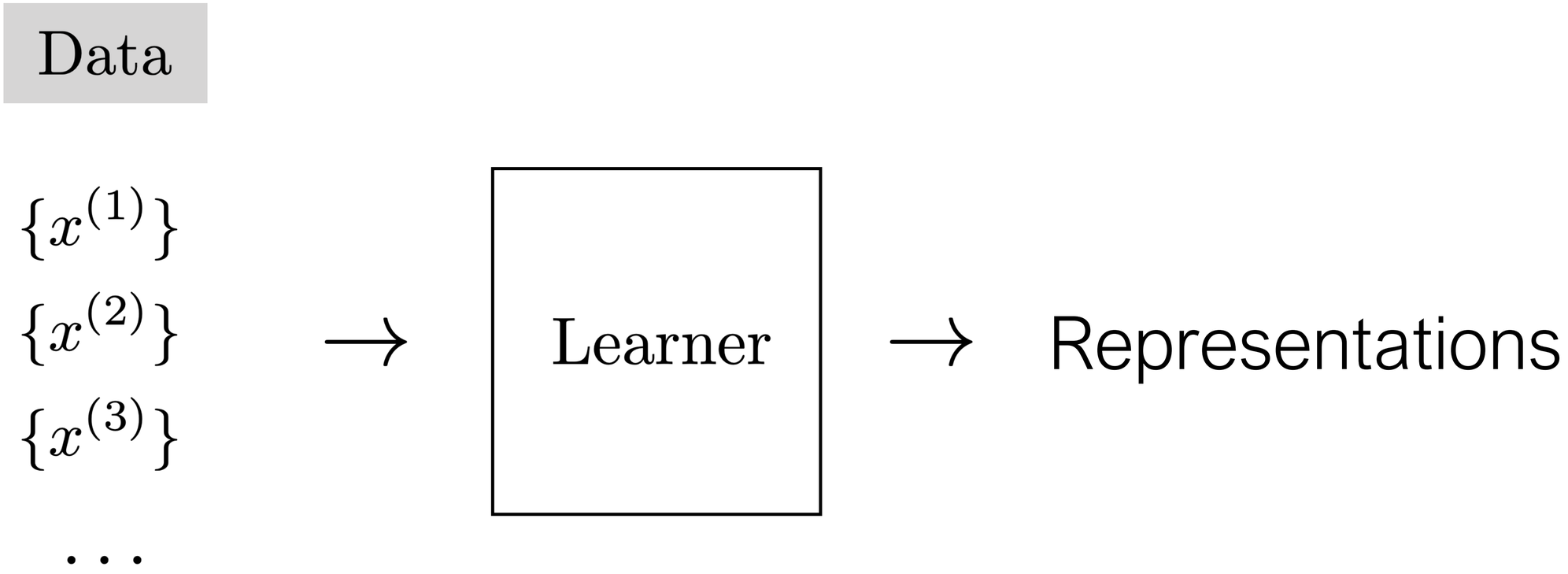
Learning without examples

(includes **unsupervised learning** and **reinforcement learning**)

Data



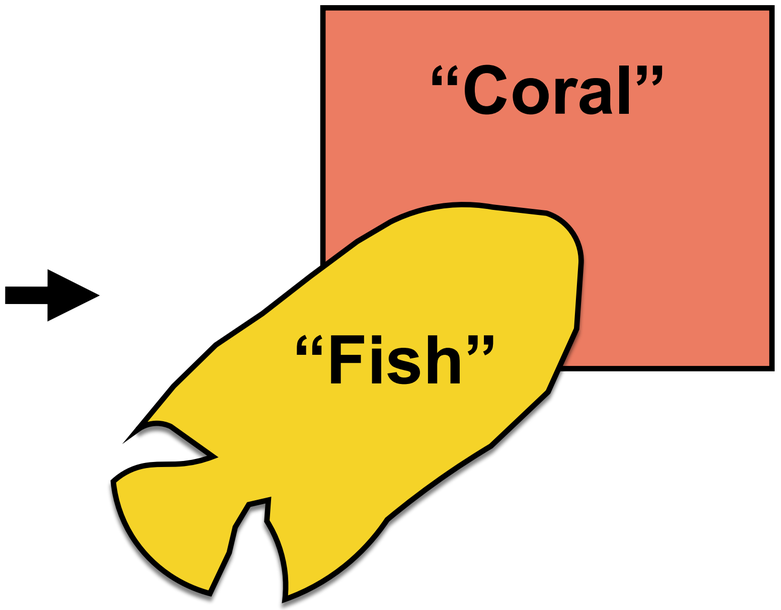
Representation Learning



Unsupervised Representation Learning

$\mathbf{x}$

Image



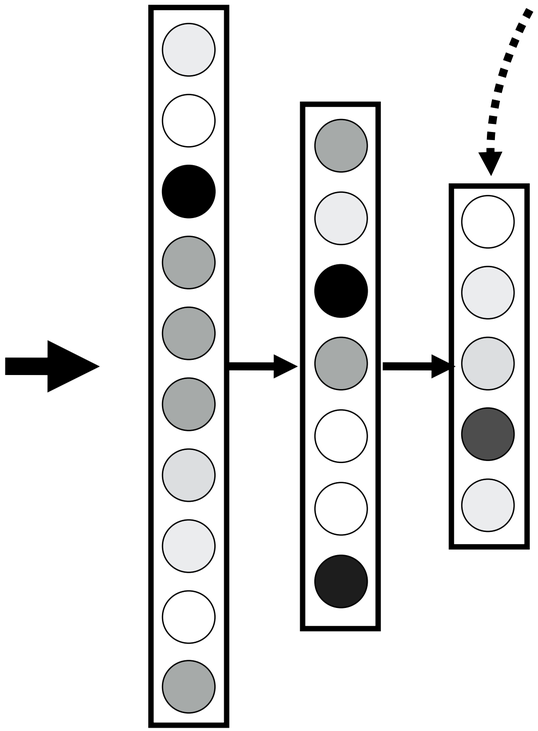
Compact mental representation

Unsupervised Representation Learning

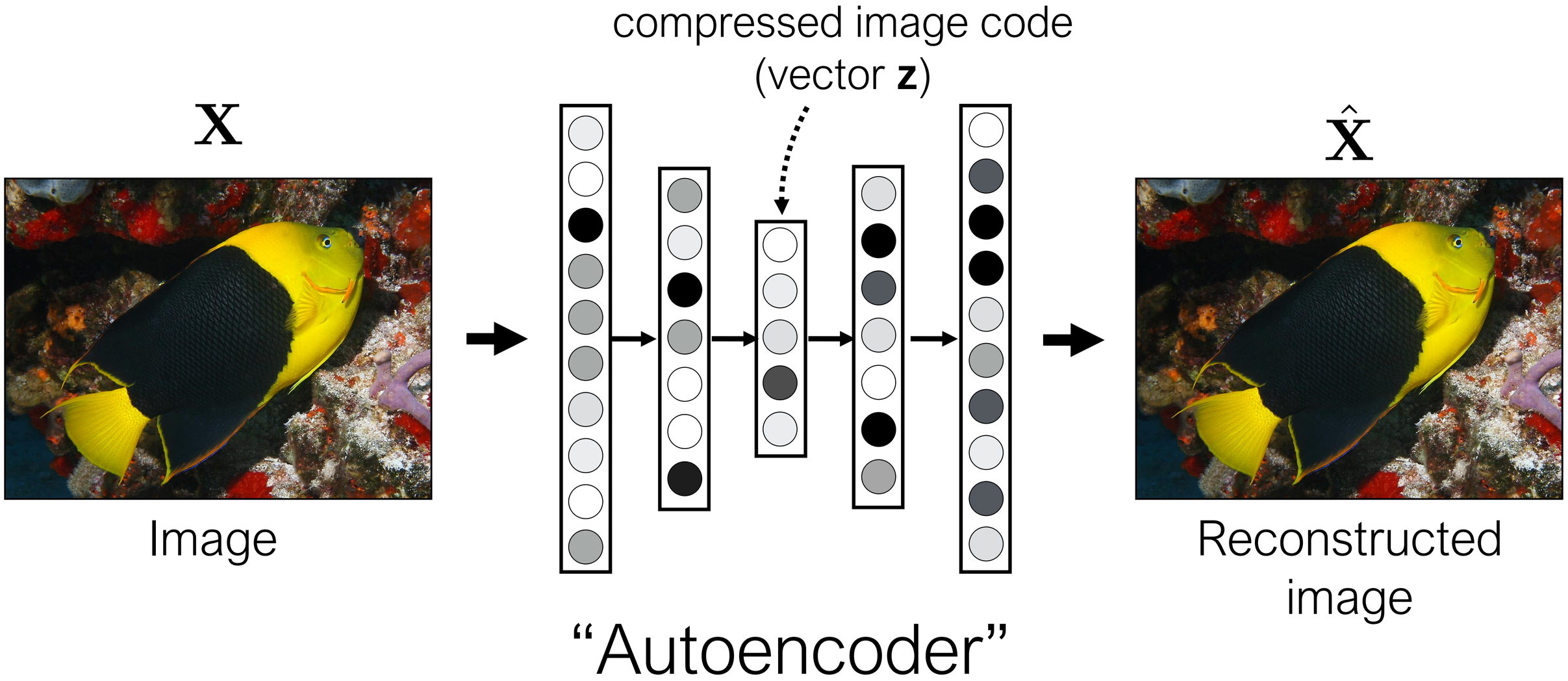
compressed image code
(vector **z**)

x

Image



Unsupervised Representation Learning



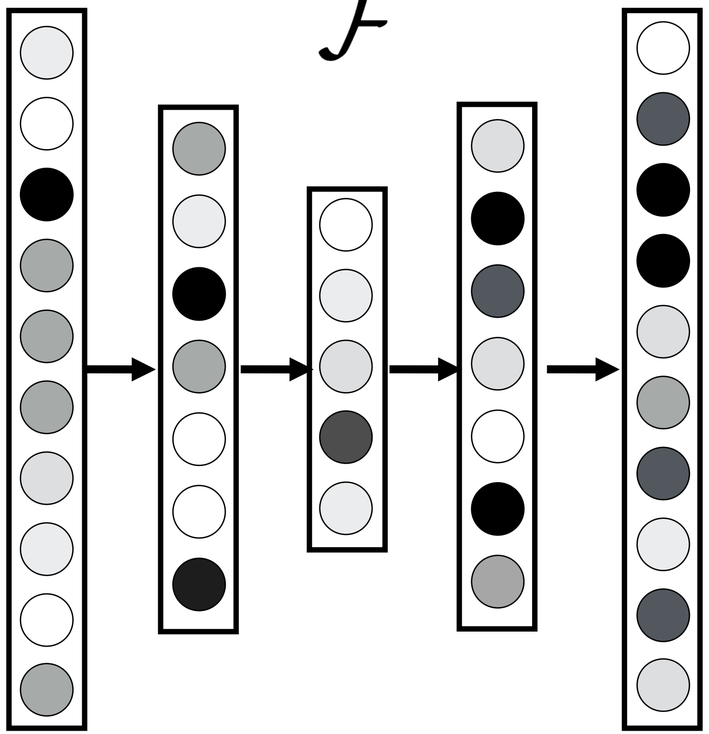
[e.g., Hinton & Salakhutdinov, Science 2006]

Autoencoder

$\mathbf{X}$

Image

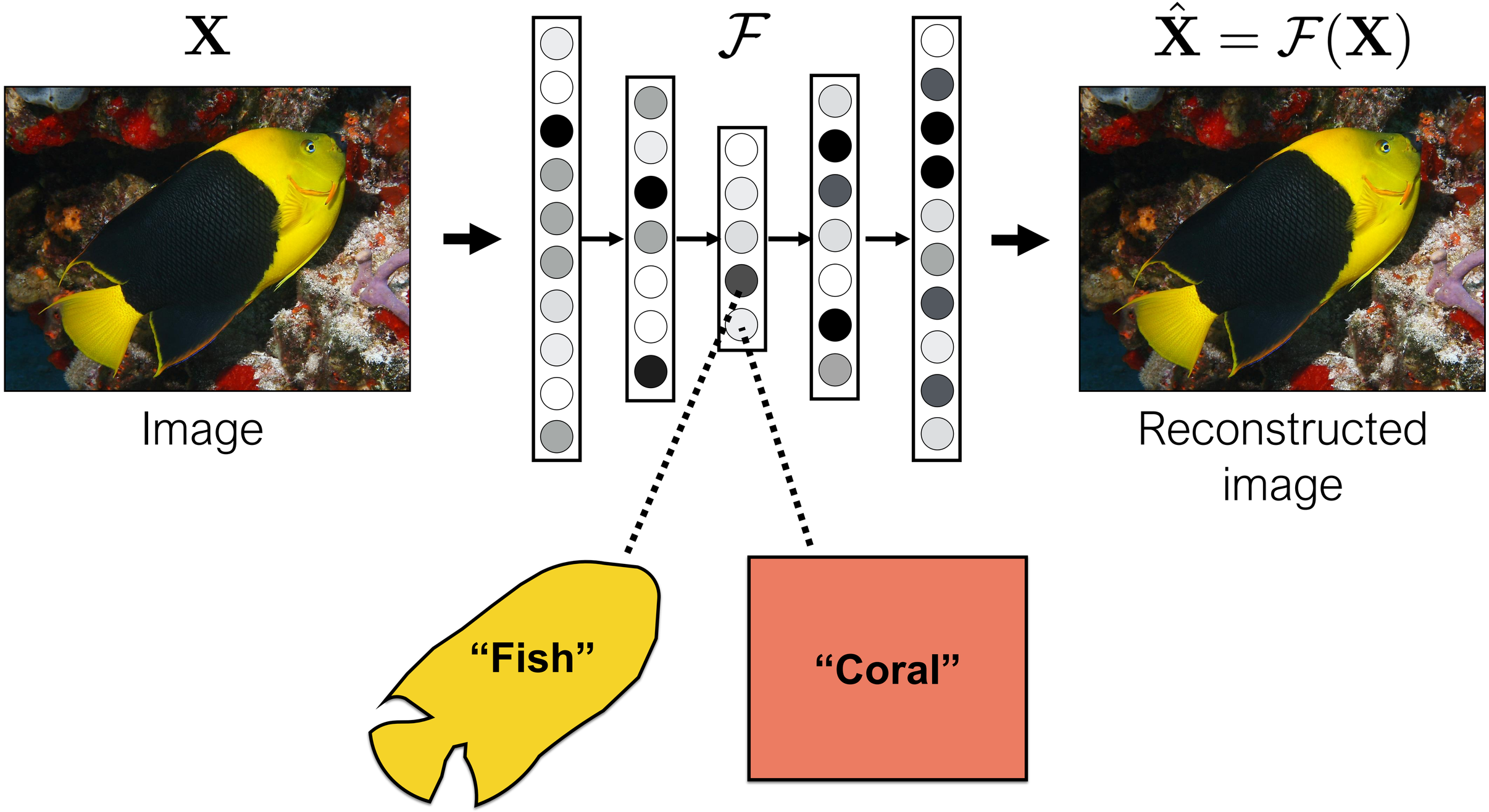
$\mathcal{F}$



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

Reconstructed image

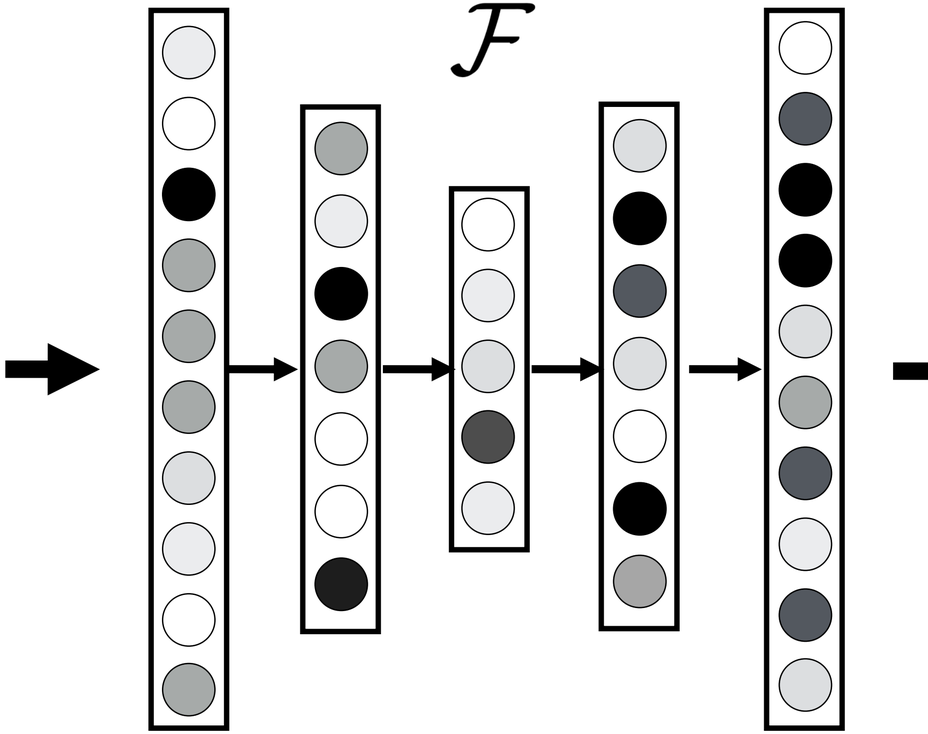
$$\arg\min_{\mathcal{F}} \mathbb{E}_{\mathbf{X}}[||\mathcal{F}(\mathbf{X}) - \mathbf{X}||]$$


$\mathbf{X}$
$\mathcal{F}$
$\hat{\mathbf{X}} = \mathcal{F}(\mathbf{X})$
Image
Reconstructed image
“Fish”
“Coral”

$\mathbf{X}$

Image

$\mathcal{F}$

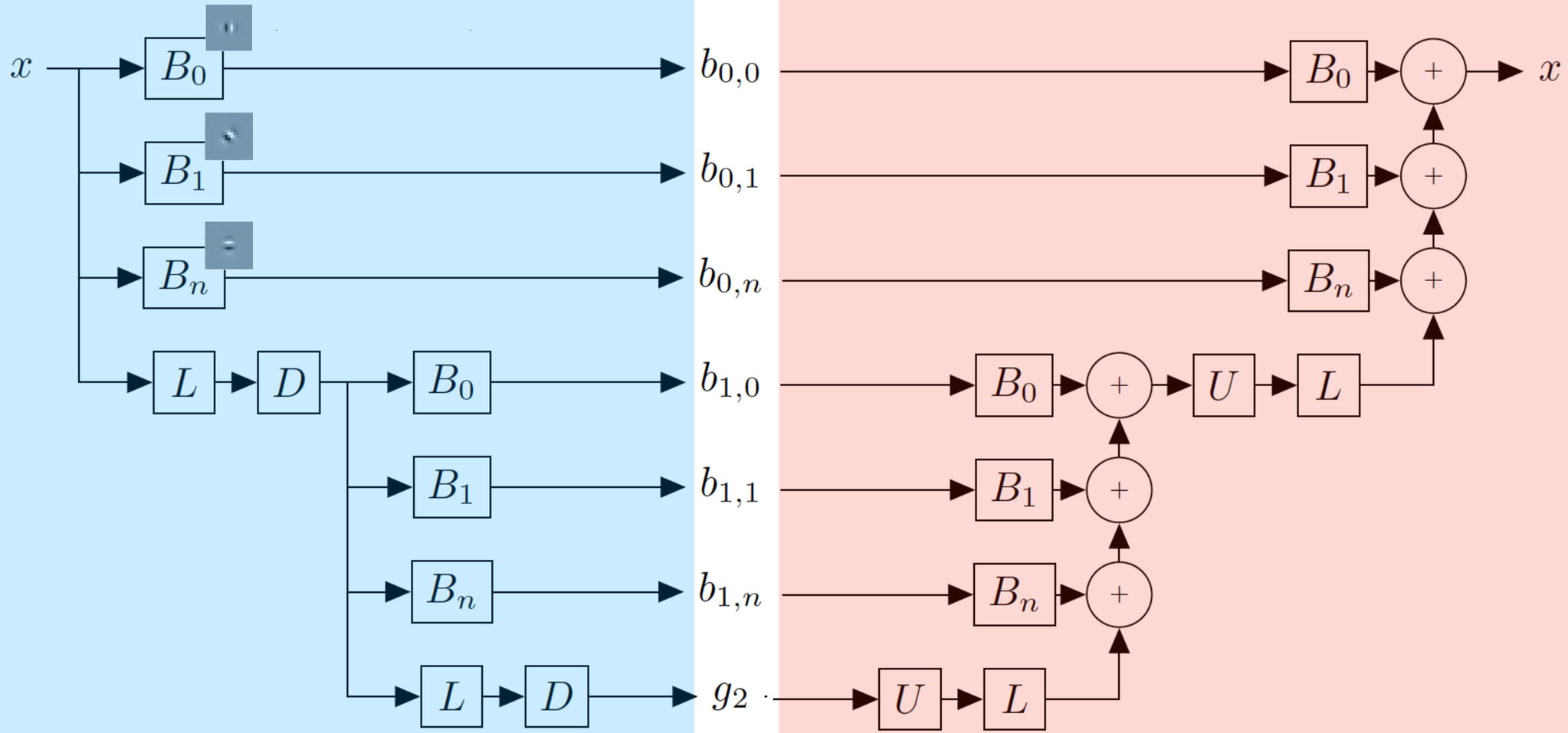


Encoder **Decoder**

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

Reconstructed image

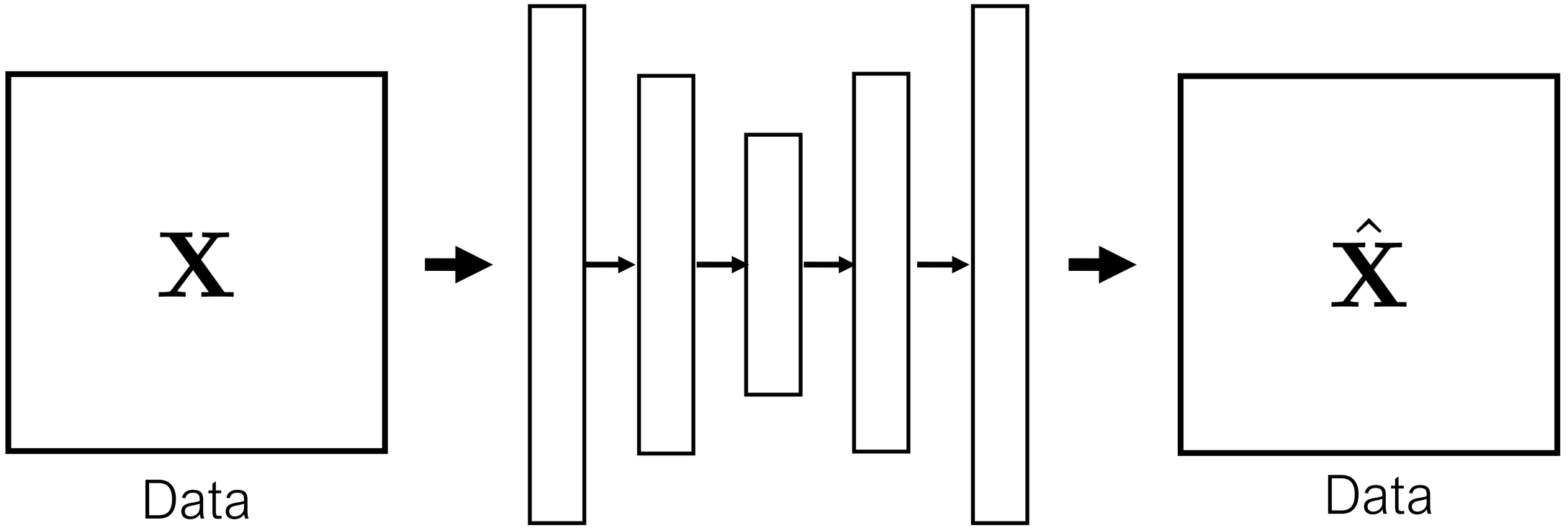
Steerable Pyramid — *A hard-coded autoencoder*



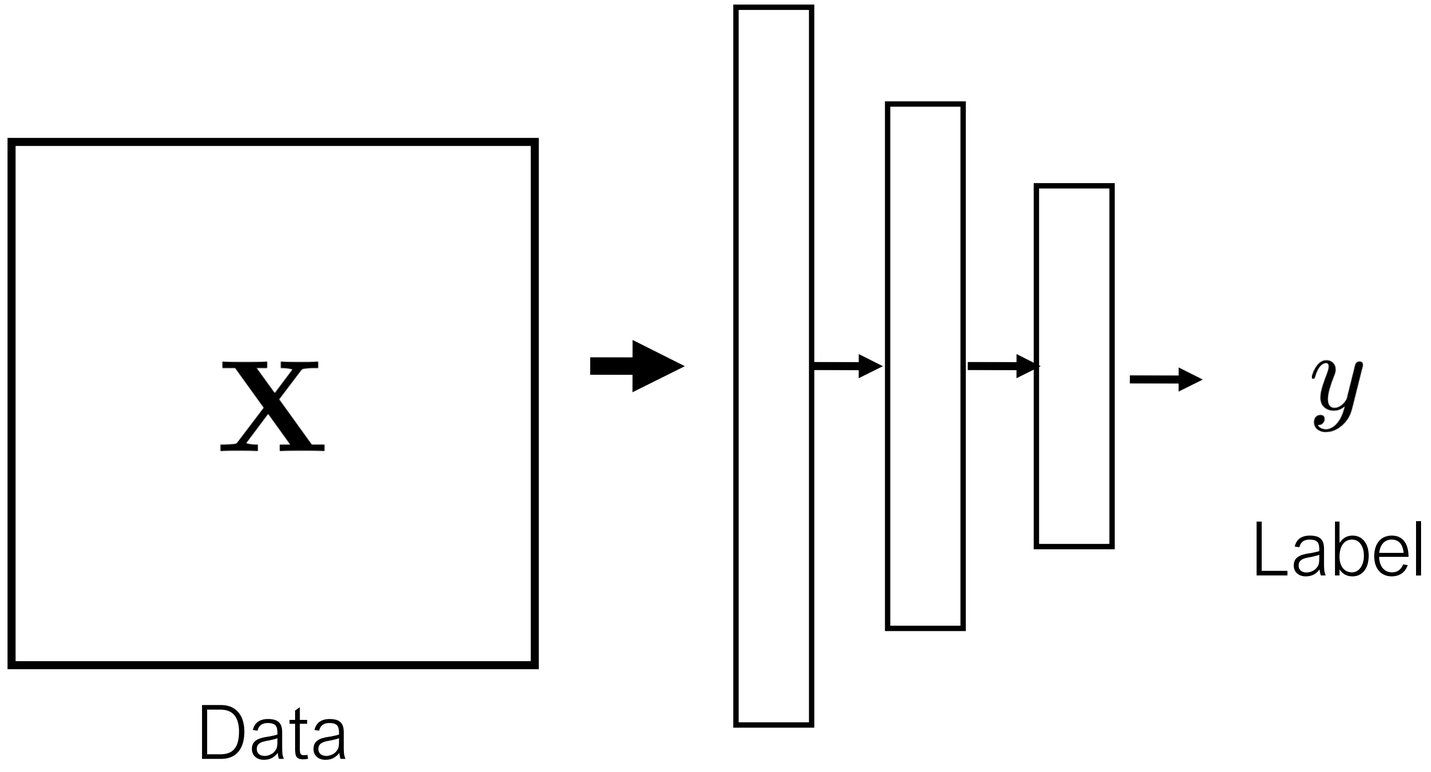
Analysis/Encoder

Synthesis/Decoder

Data compression

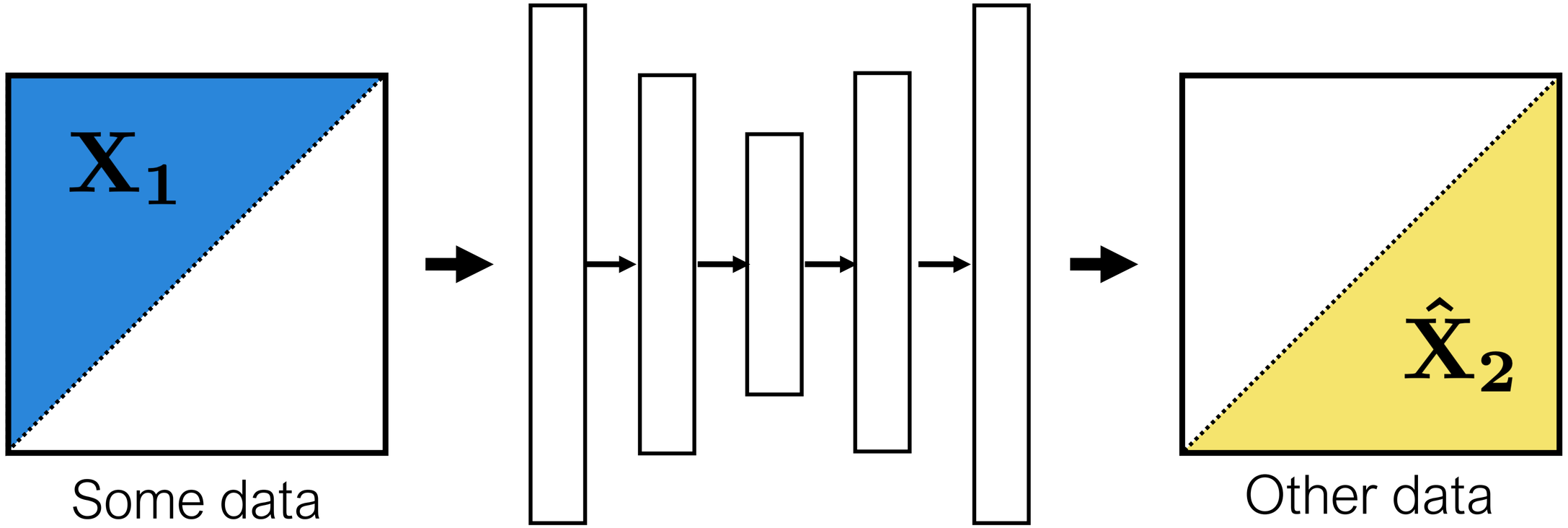


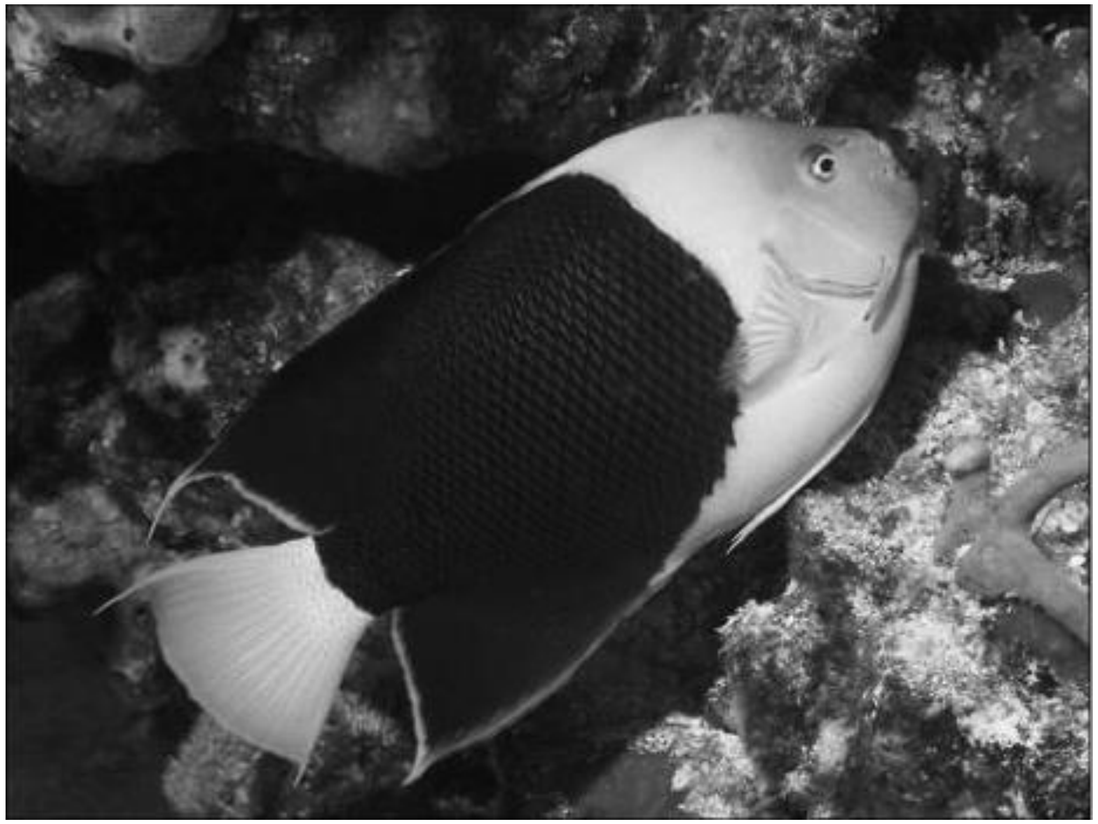
Label prediction



e.g., image classification

Data prediction
aka “self-supervised learning”





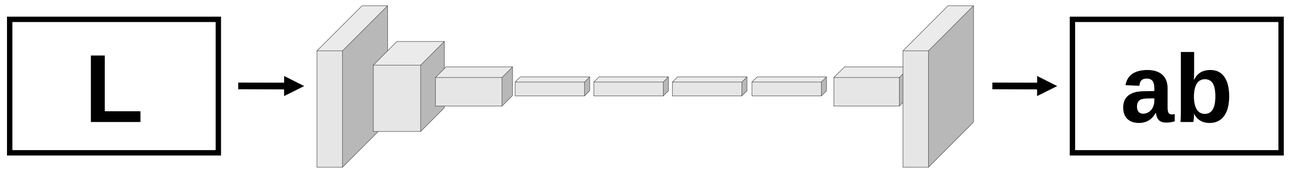
$\mathcal{F}$

Grayscale image: L channel

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

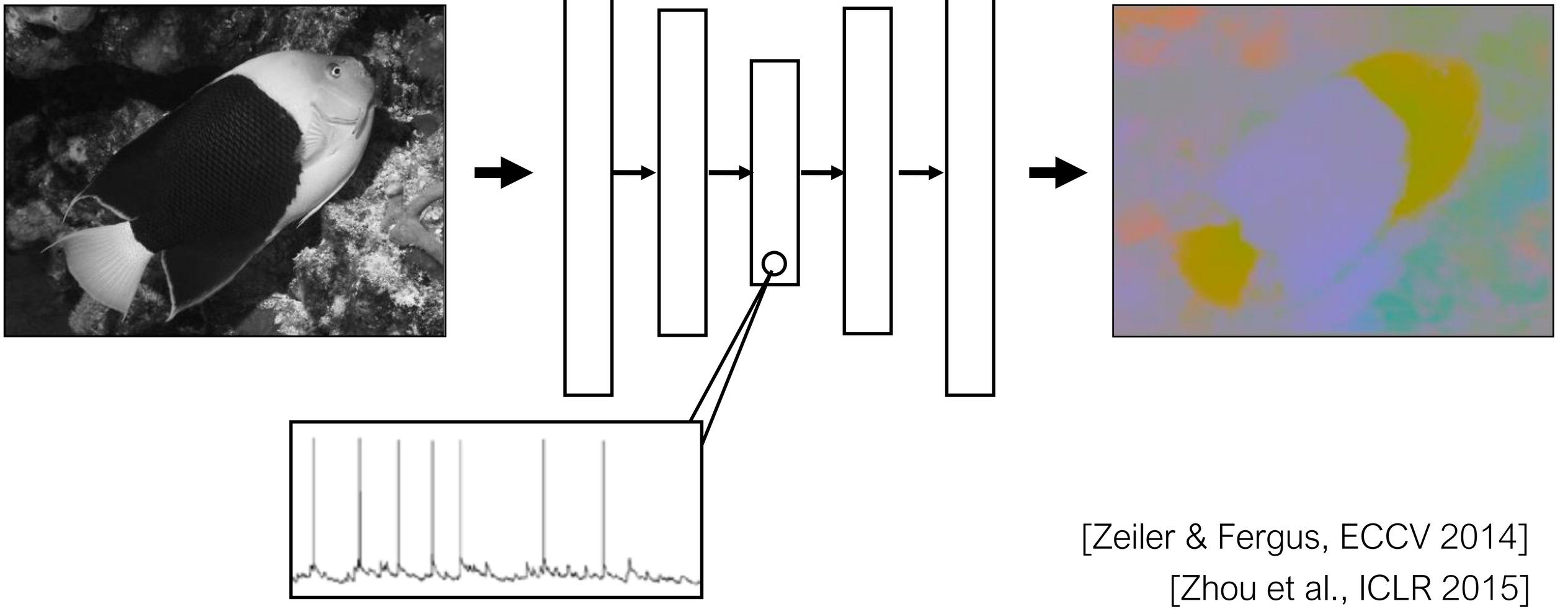
Color information: ab channels

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



[Zhang, Isola, Efros, ECCV 2016]

Deep Net “Electrophysiology”



[Zeiler & Fergus, ECCV 2014]

[Zhou et al., ICLR 2015]

Stimuli that drive selected neurons (conv5 layer)

faces

dog faces

flowers