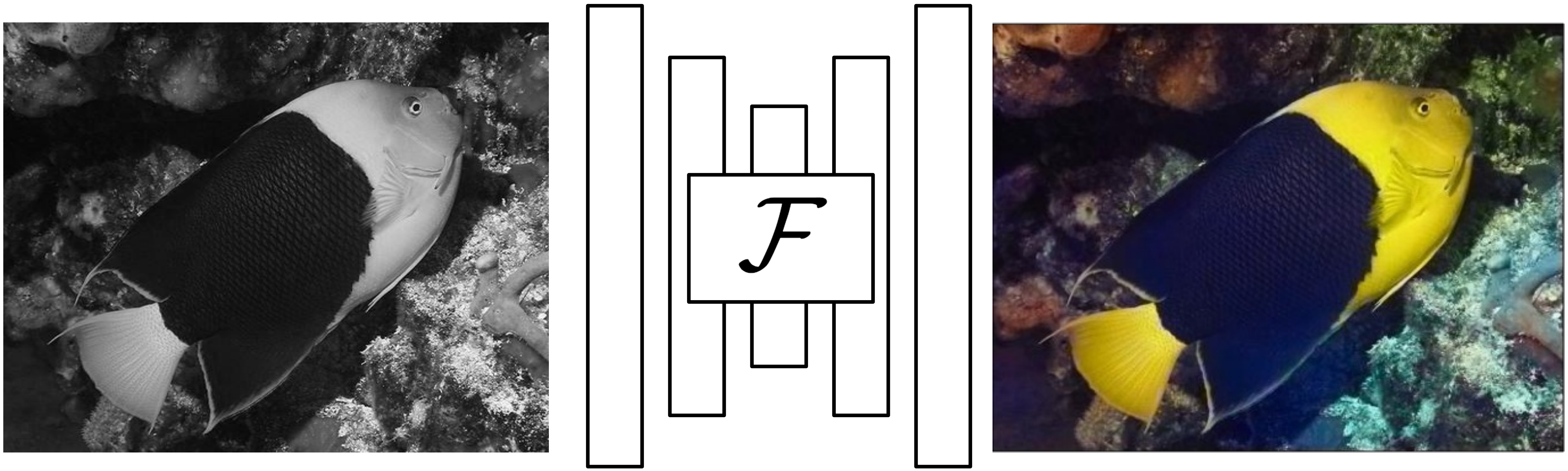


Convolutional Network for Image Synthesis

Jun-Yan Zhu

16-726 Learning-based Image Synthesis, Spring 2023

many slides from Alyosha Efros, Phillip Isola, Richard Zhang, James Hays, and Andrea Vedaldi, Jitendra Malik.

Review (data-driven graphics)

Review (data-driven graphics)

Nearest neighbor methods:

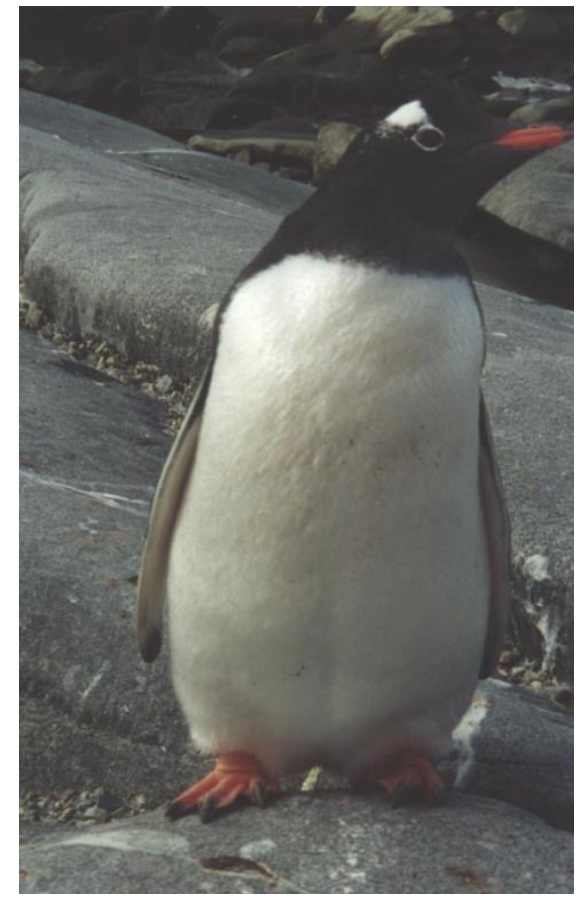
1. Stored examples
2. Calculate distance between two examples
3. Voting (label transfer): image blending/averaging

Visual similarity via labels

“Penguin”

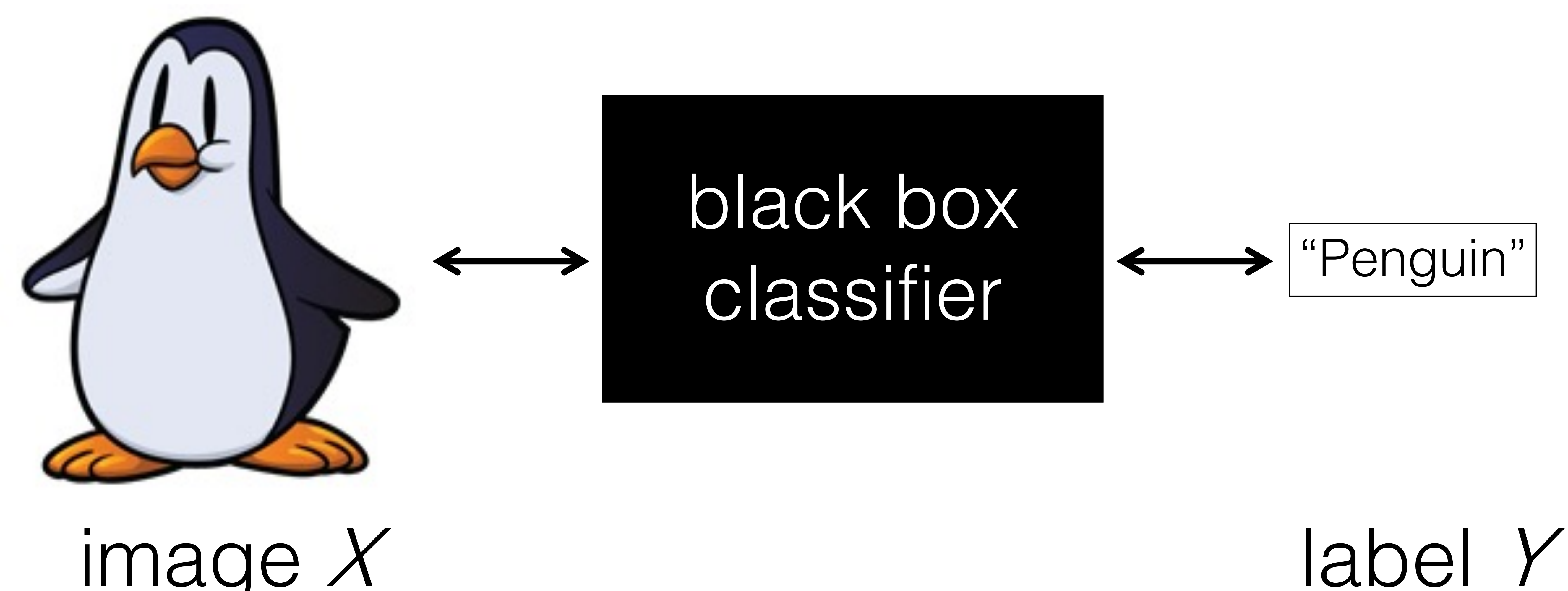
?

==



“Penguin”

Machine Learning as data association



At test time...

image X

black box classifier

?

Examples from MNIST dataset [LeCun. 1998]

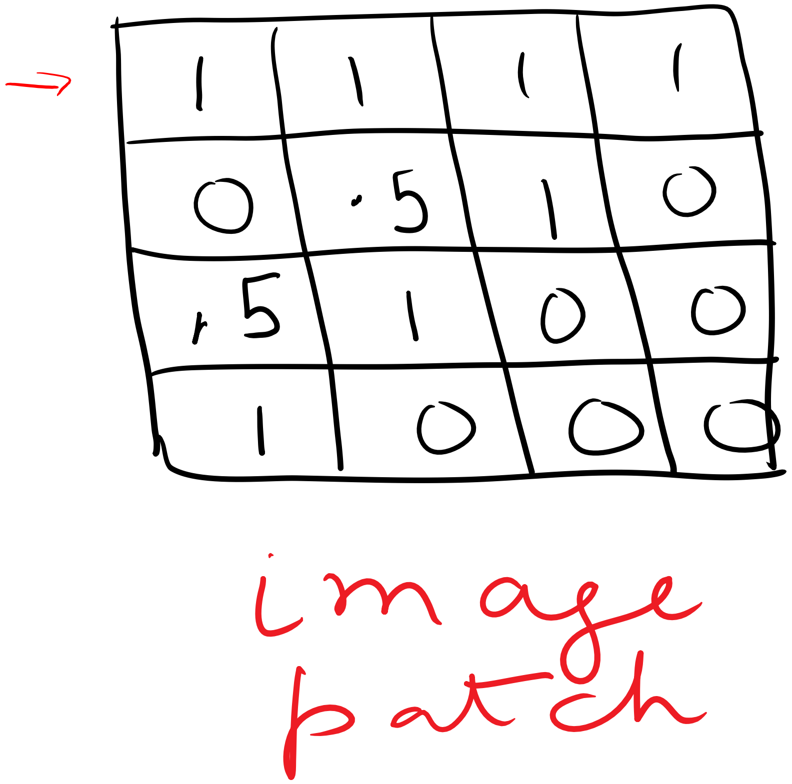
Warm-up Example:
Binary Digit Classification

vs.

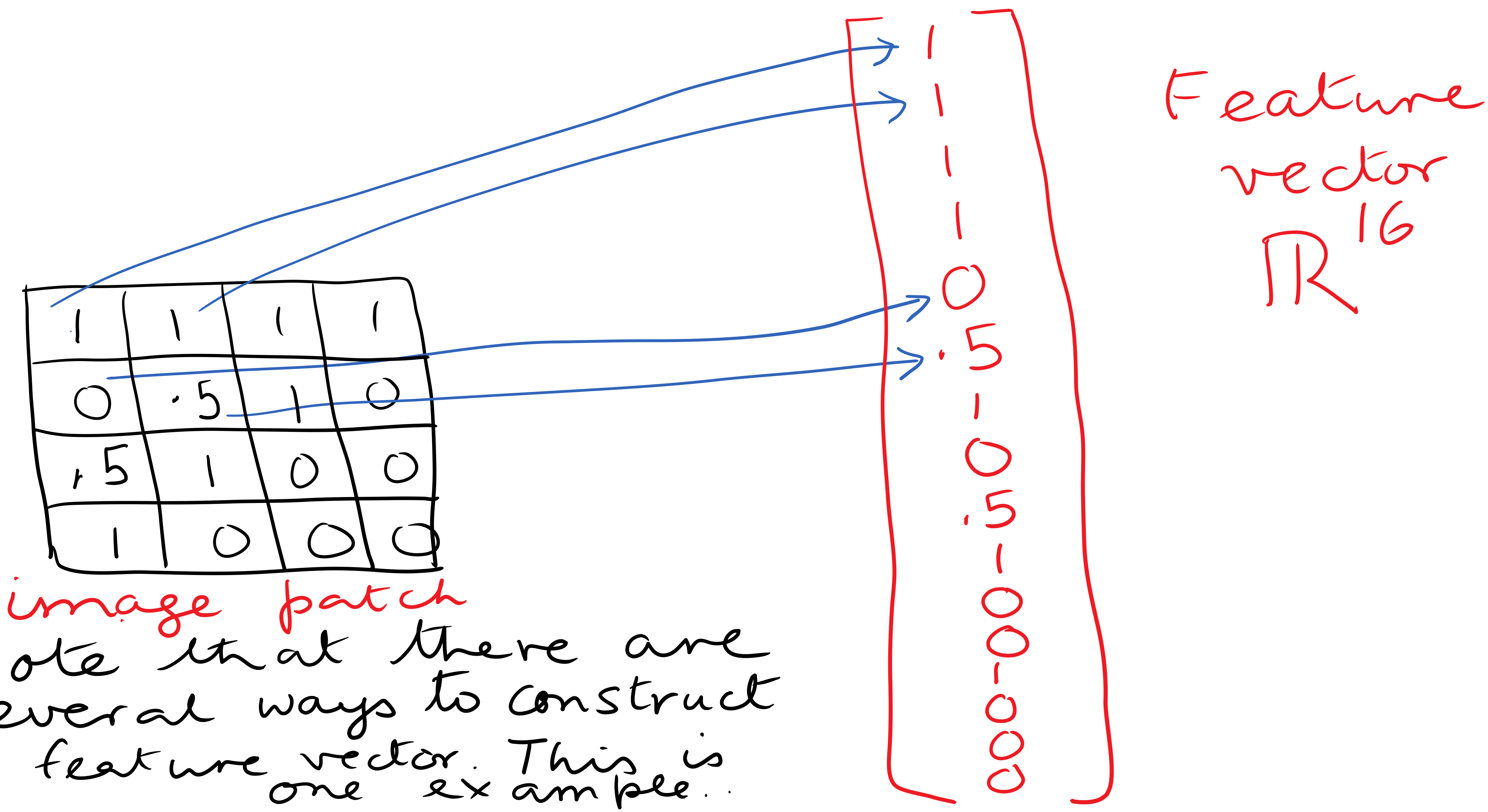
Learning Approach to Digit Recognition

- **Collect Training Images**
  - Positive: 
  - Negative: 
- **Training Time**
  - Compute feature vectors for positive and negative example images
  - Train a classifier
- **Test Time**
  - Compute feature vector on new test image:
  - Evaluate classifier

Let us take an example…

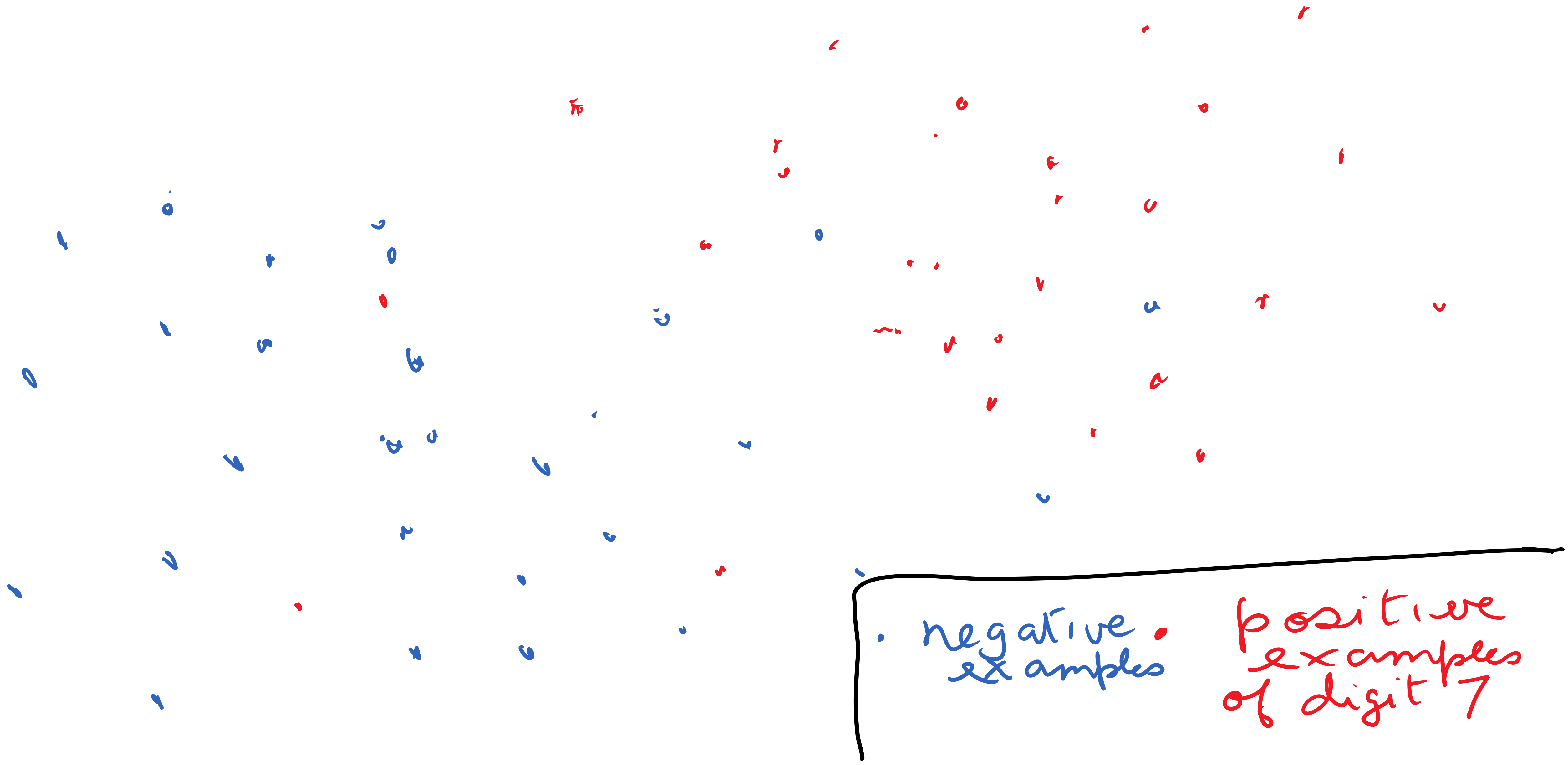


Let us take an example…

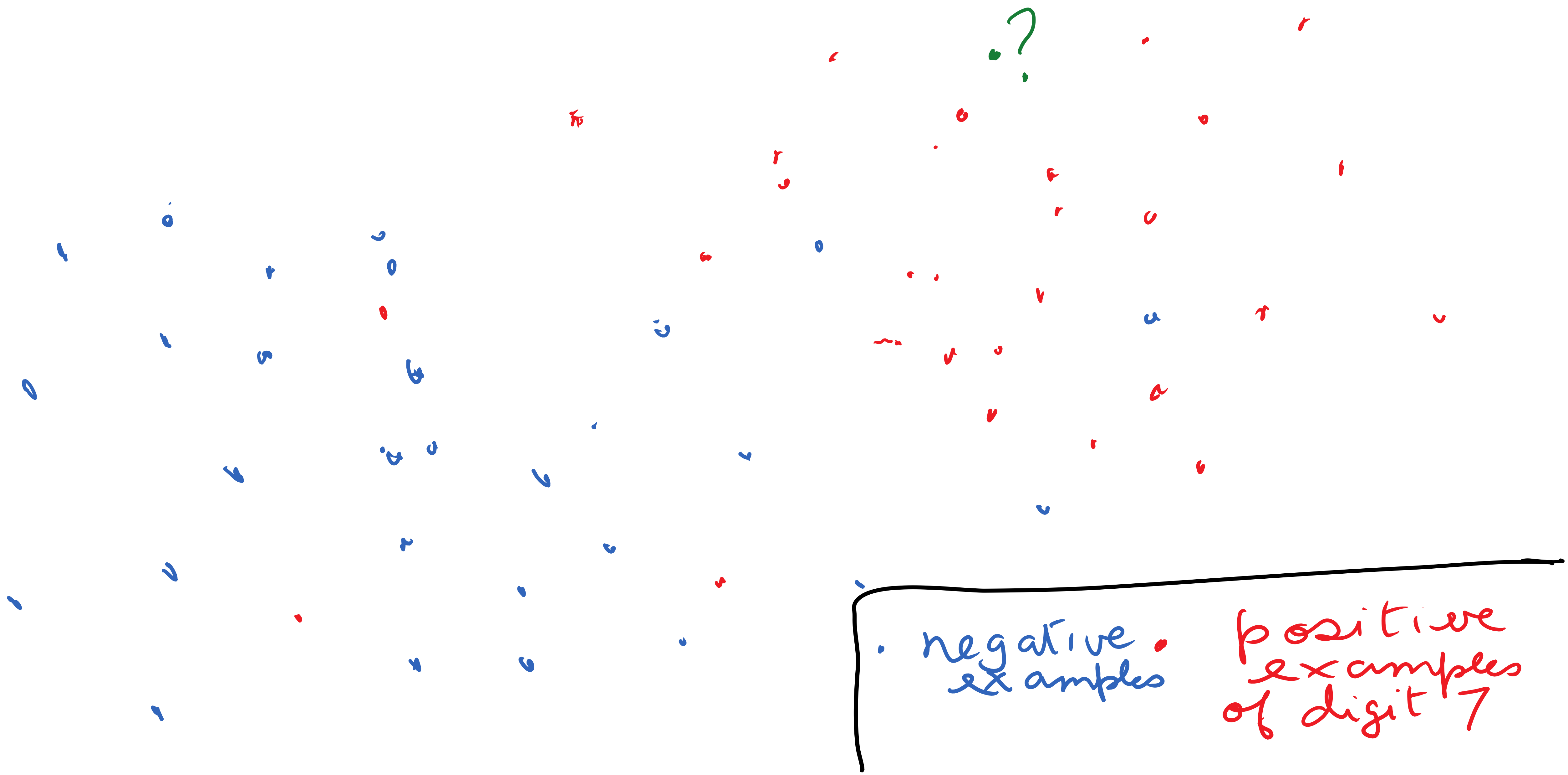


Note that there are several ways to construct a feature vector. This is one example..

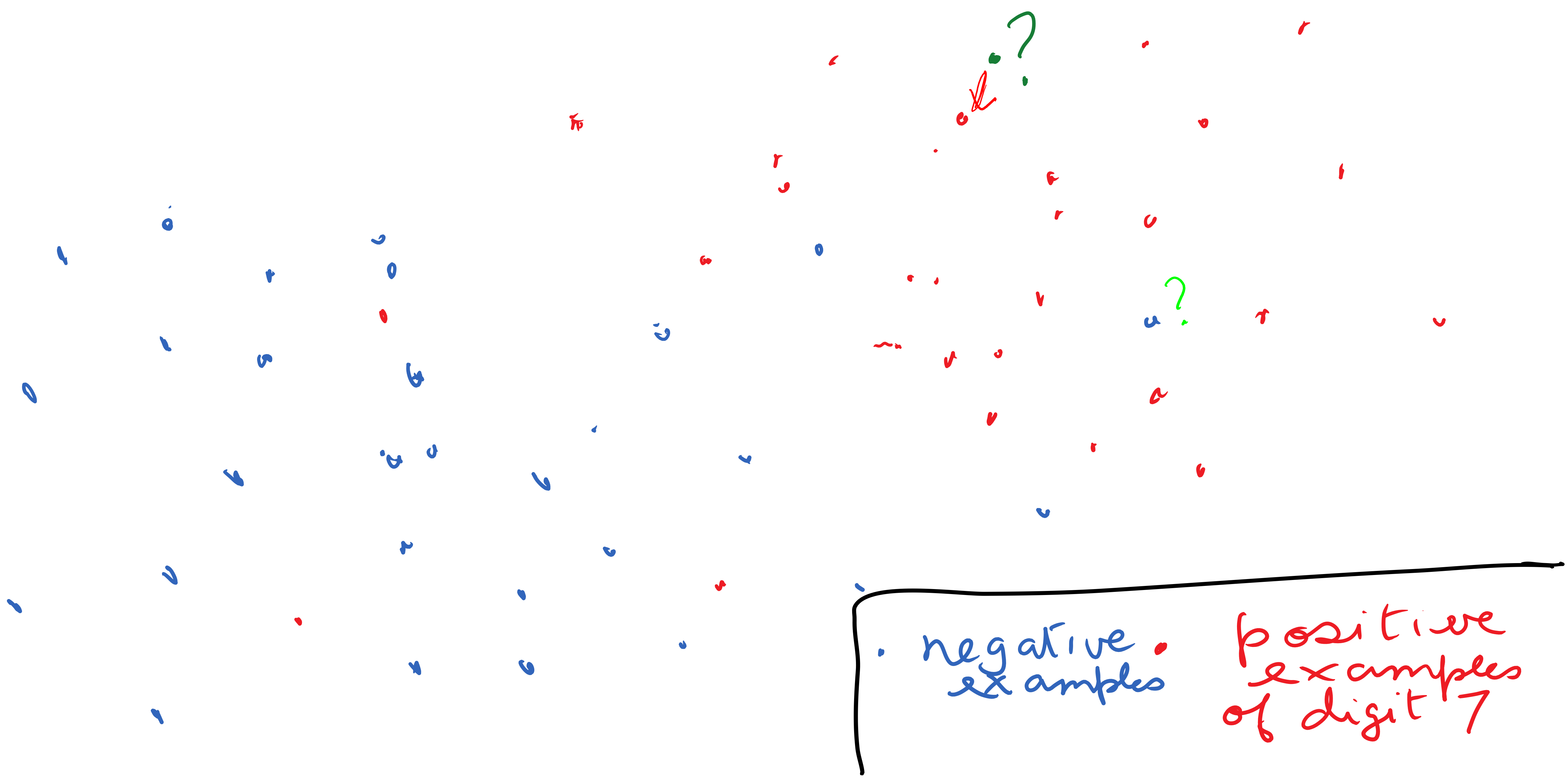
In feature space, positive and negative examples are just points…



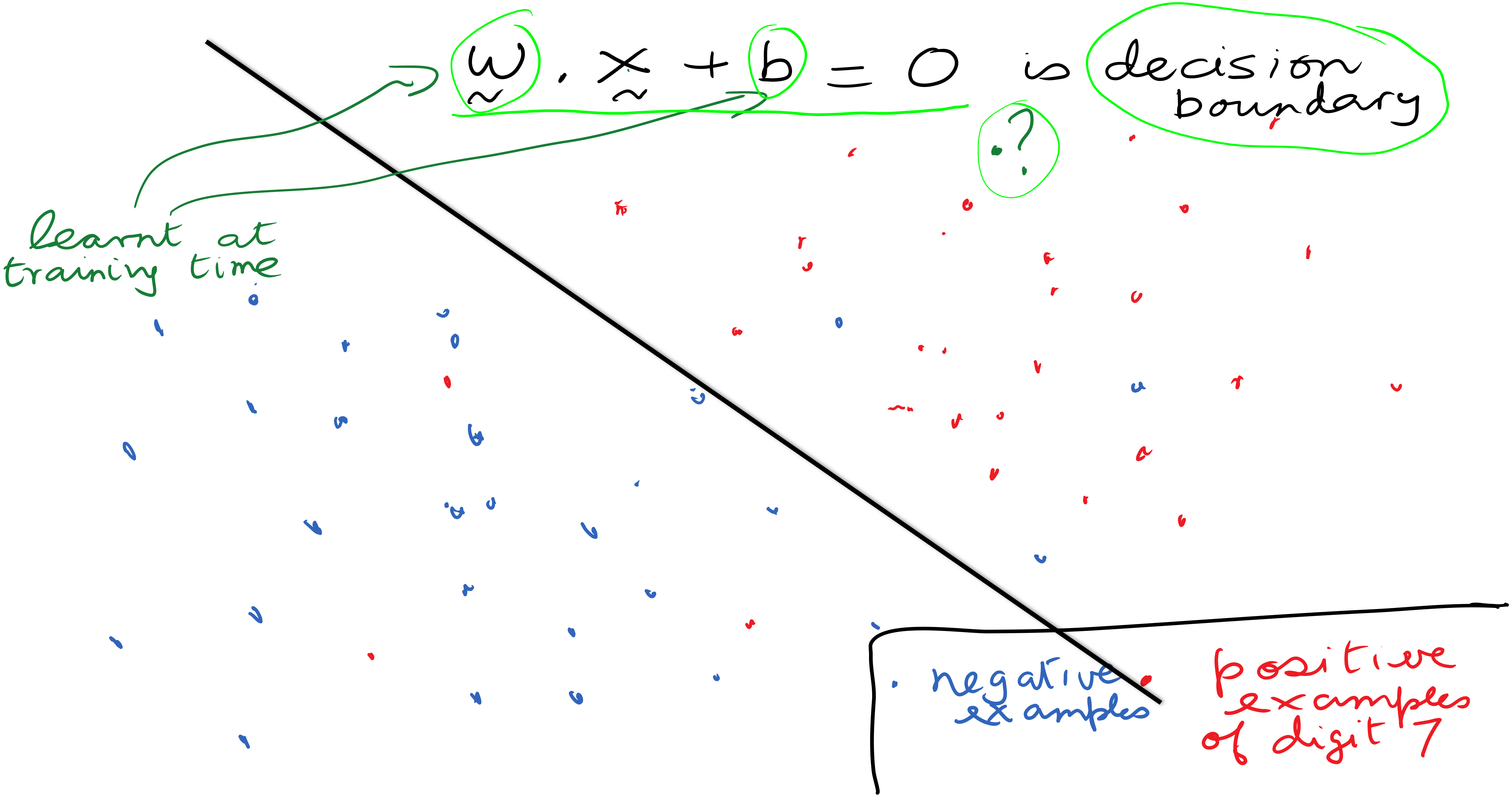
How do we classify a new point?



Nearest neighbor rule
“transfer label of nearest example”



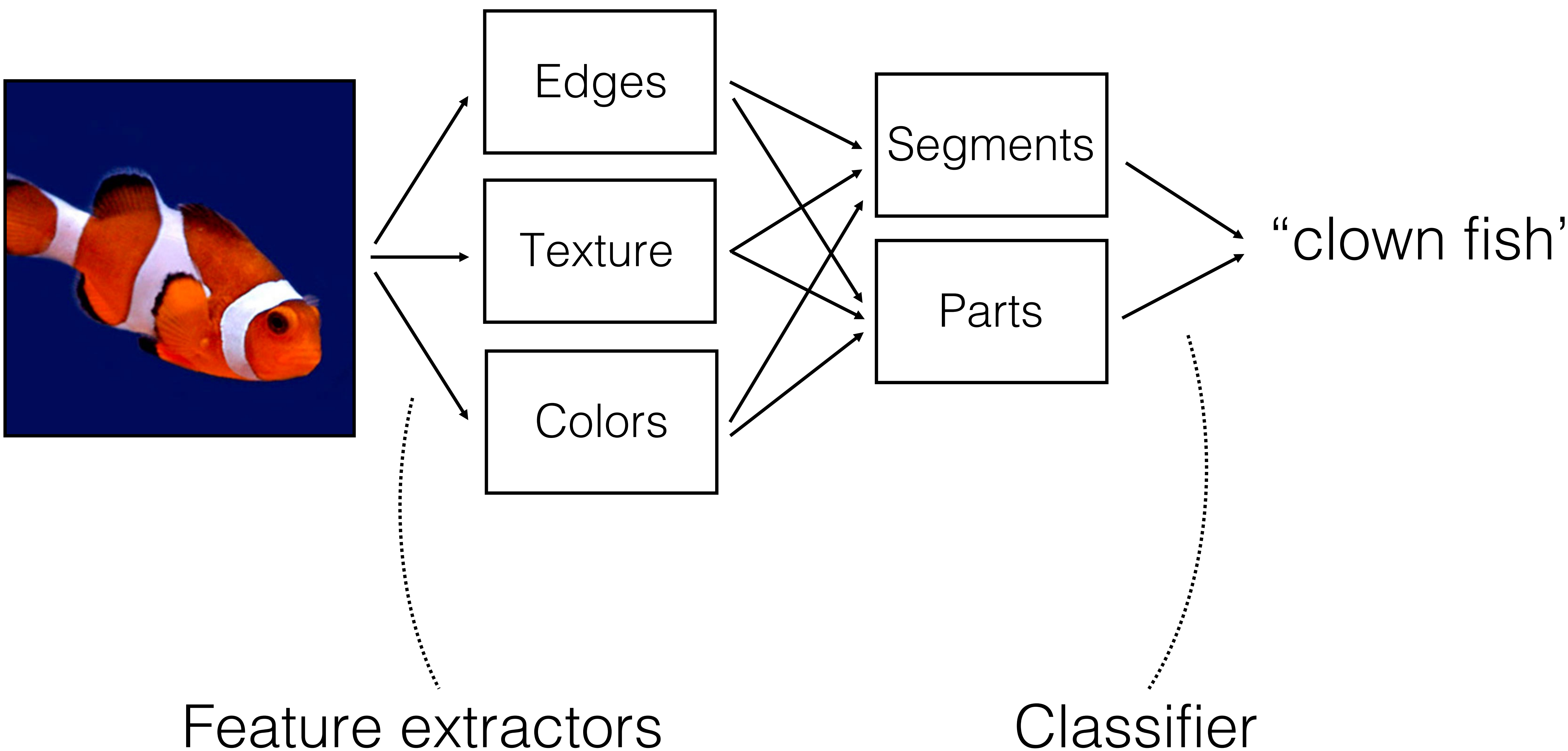
Linear classifier rule



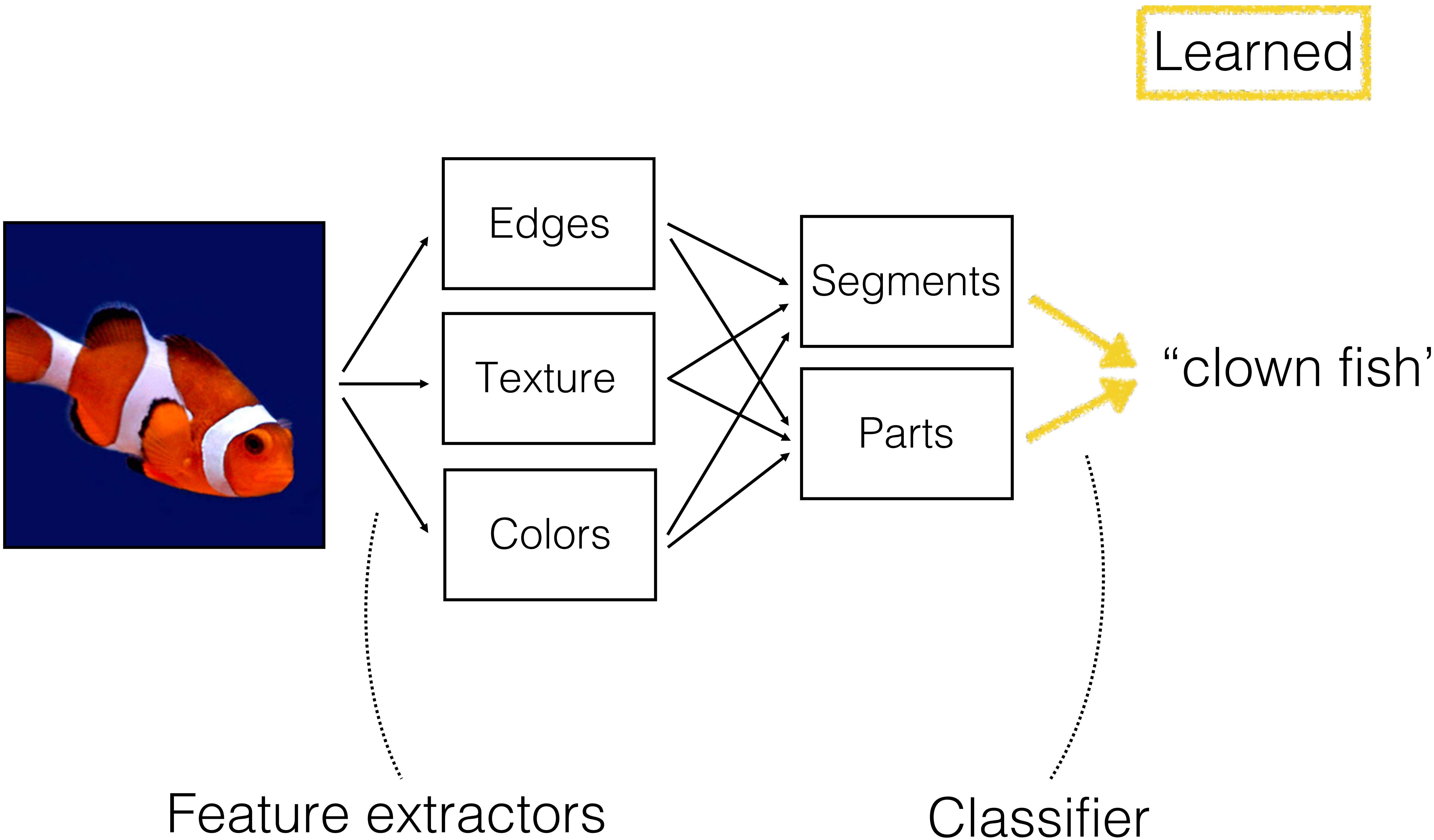
Basic idea

→ Brain/Machine → “clown fish”

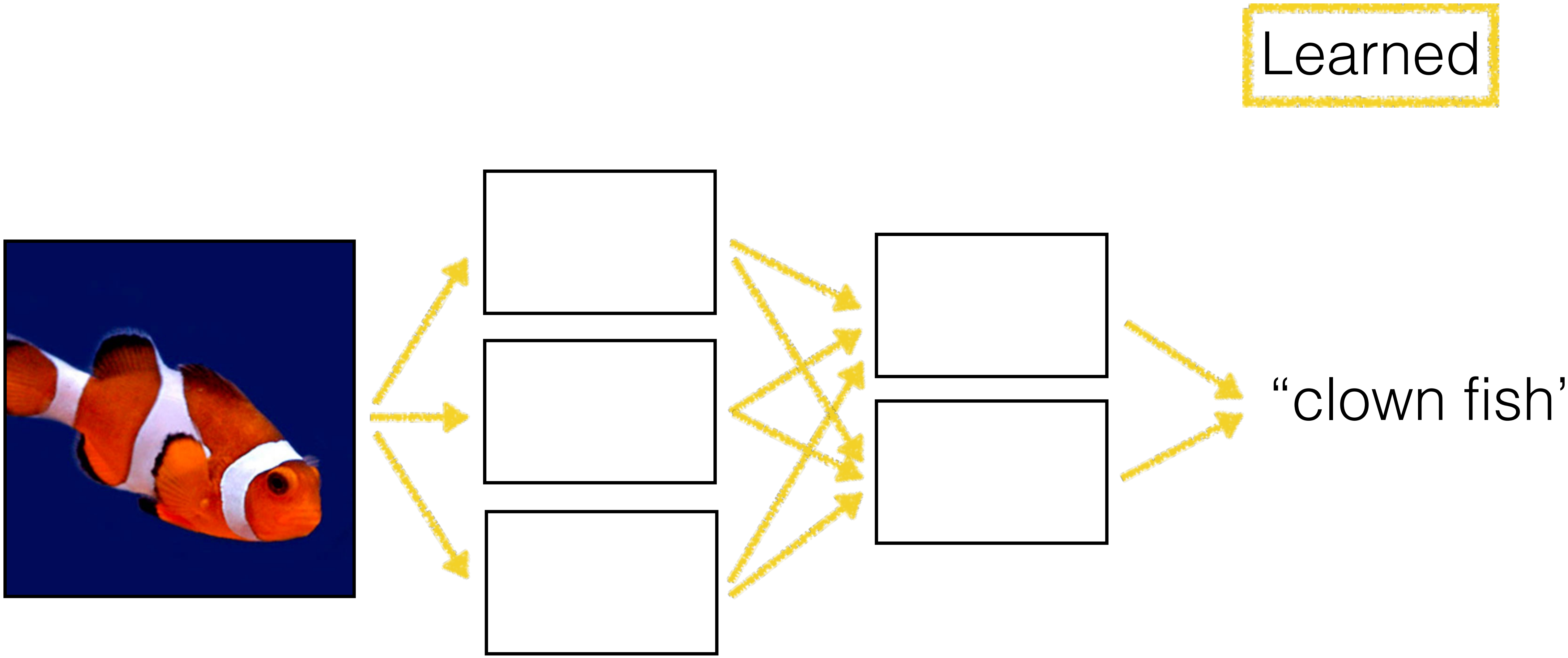
Object recognition



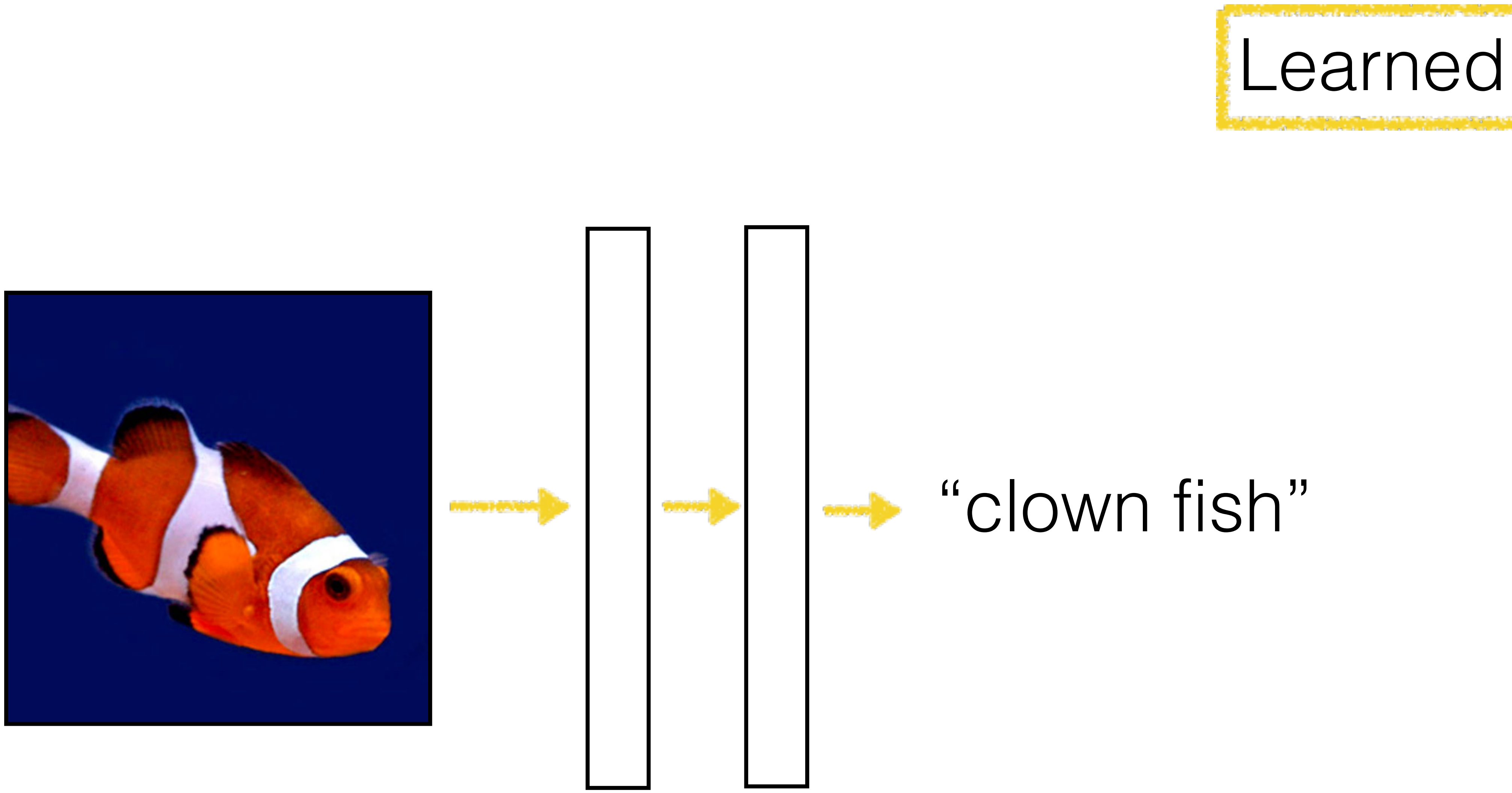
Object recognition



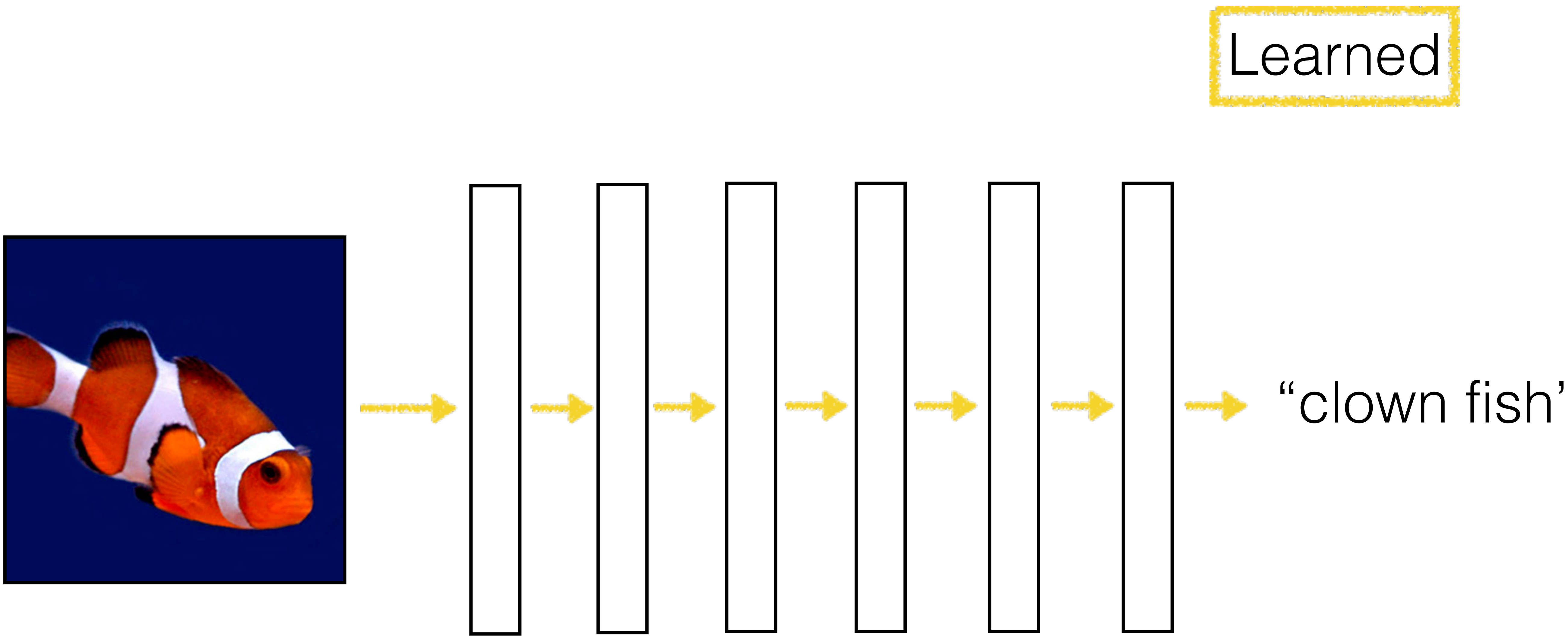
Neural network



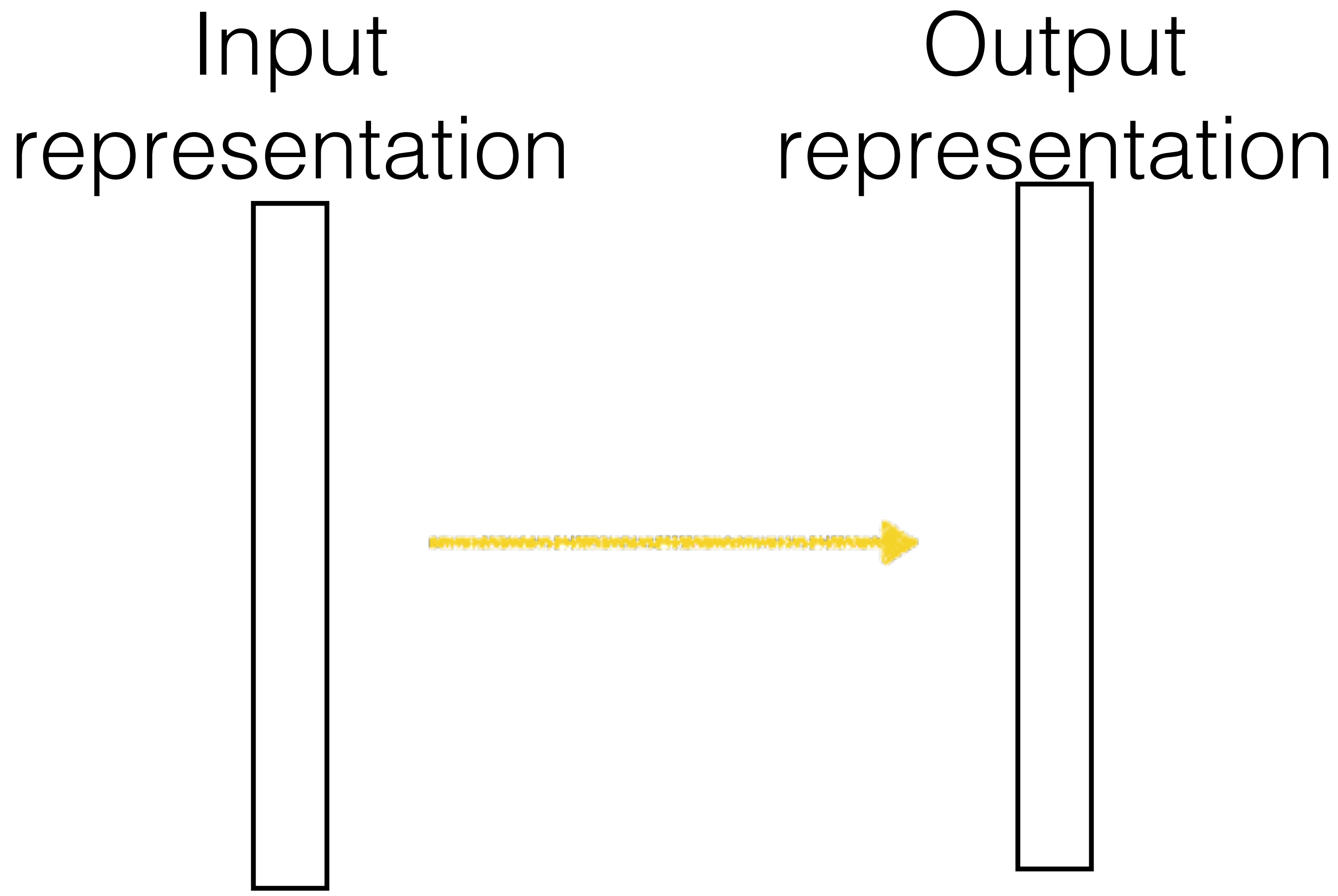
Neural network



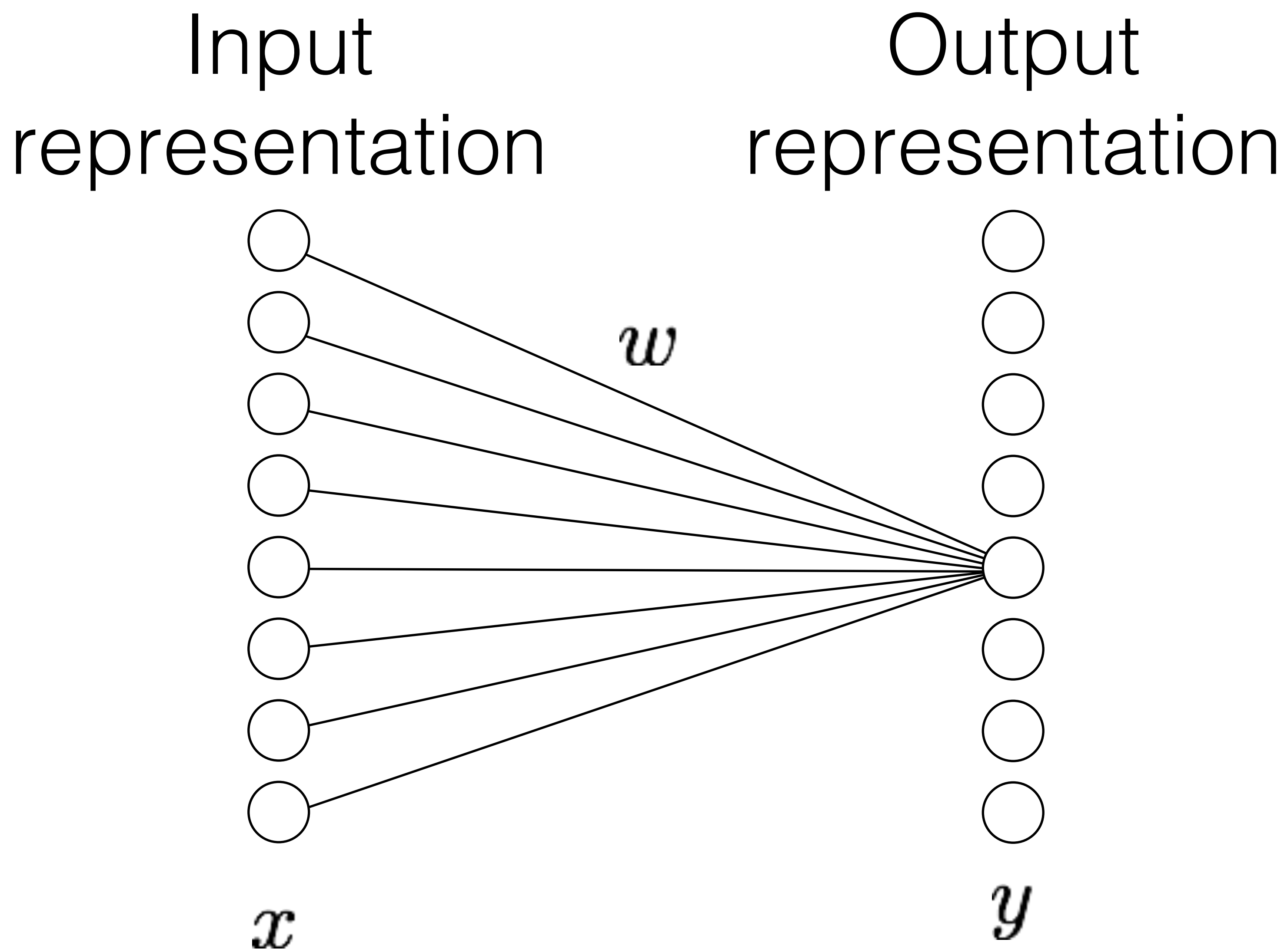
Deep neural network



Computation in a neural net



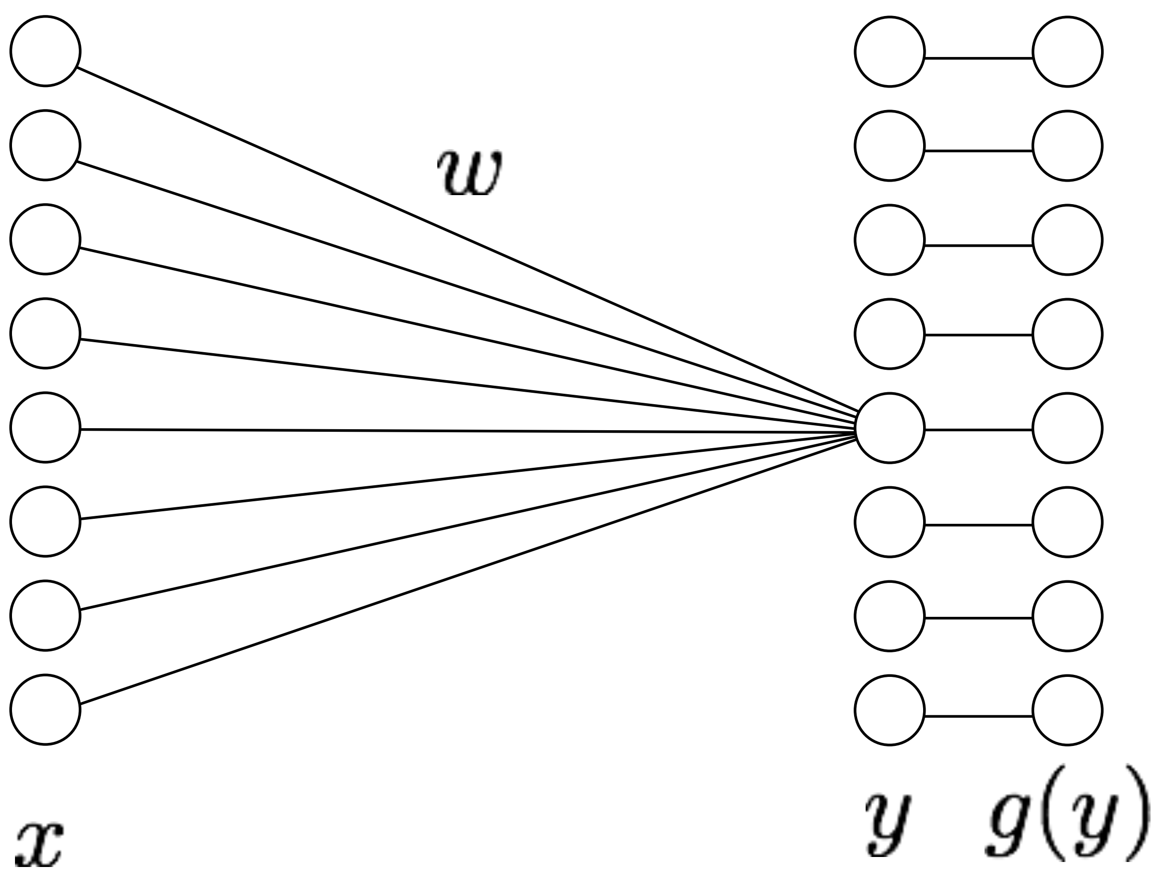
Computation in a neural net



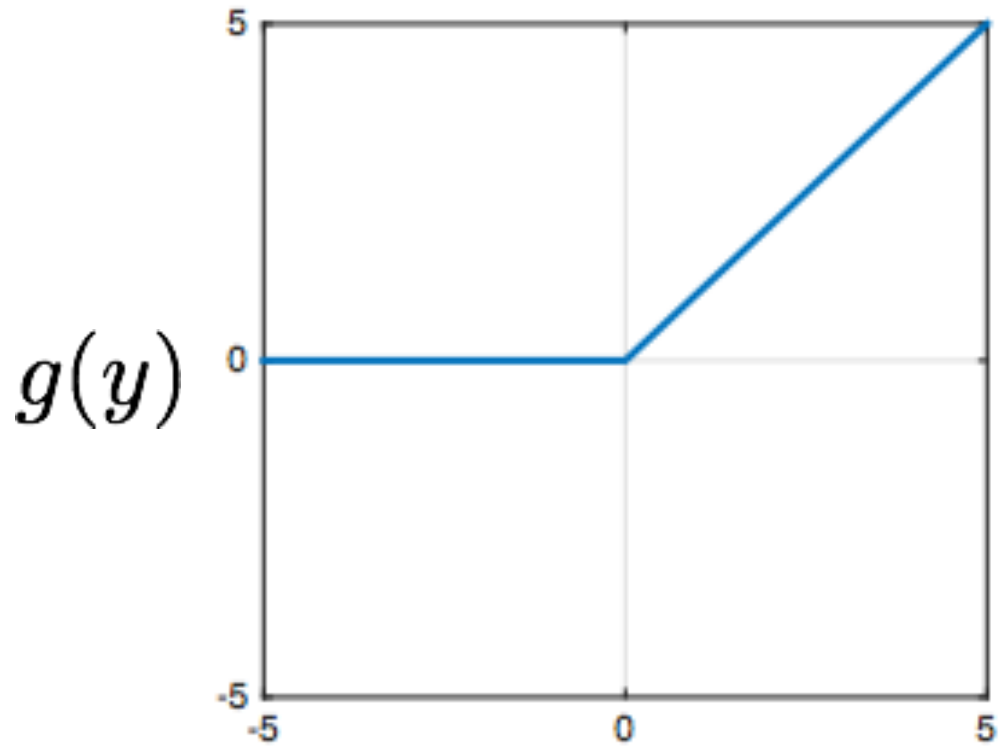
$$y_j = \sum_i w_{ij} x_i$$

i: the i^{th} dimension of x, j; the j^{th} dimension of y

Computation in a neural net



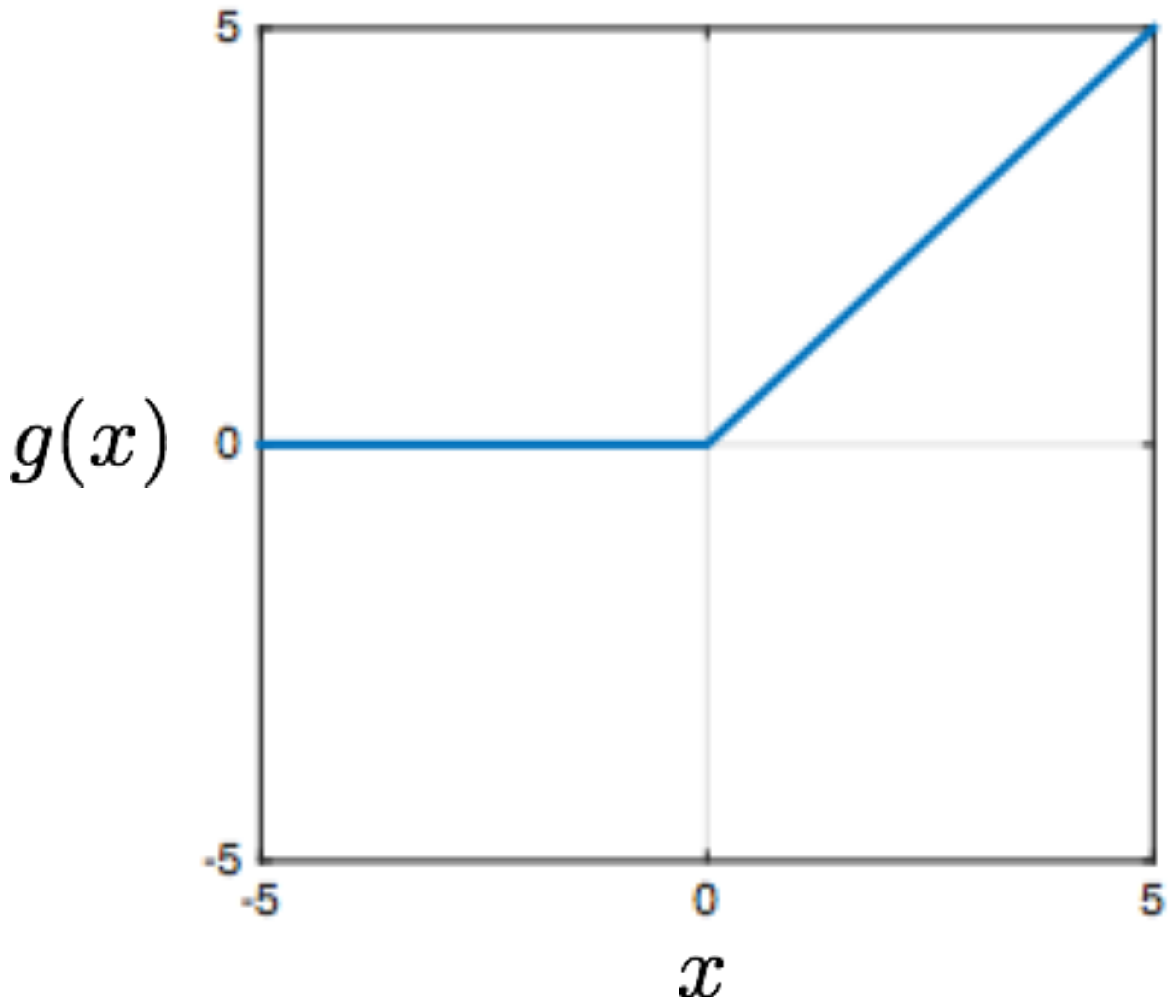
Rectified linear unit (ReLU)



$$g(y) = \max(0, y)$$

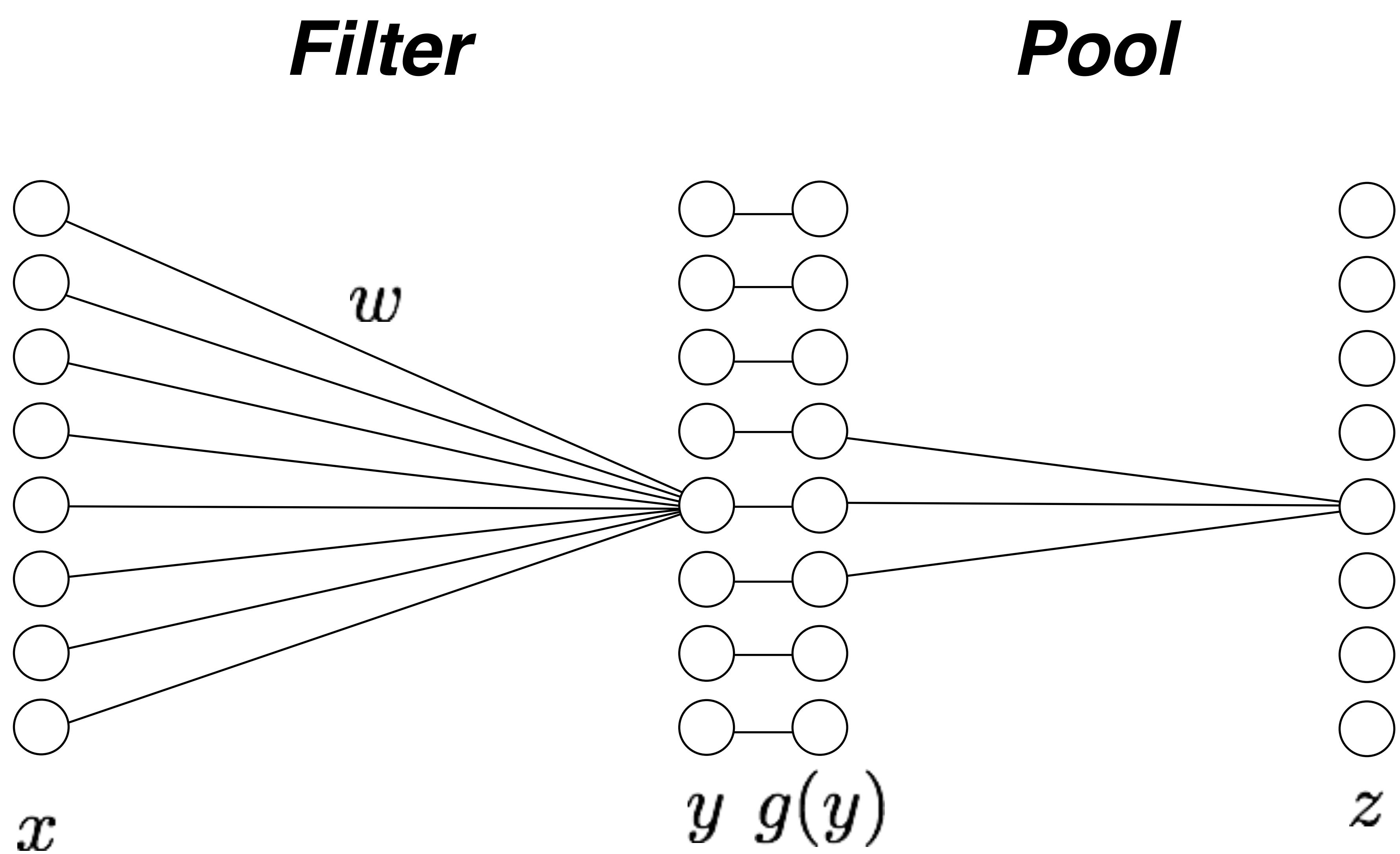
Computation in a neural net

Rectified linear unit (ReLU)



$$g(x) = \max(0, x)$$

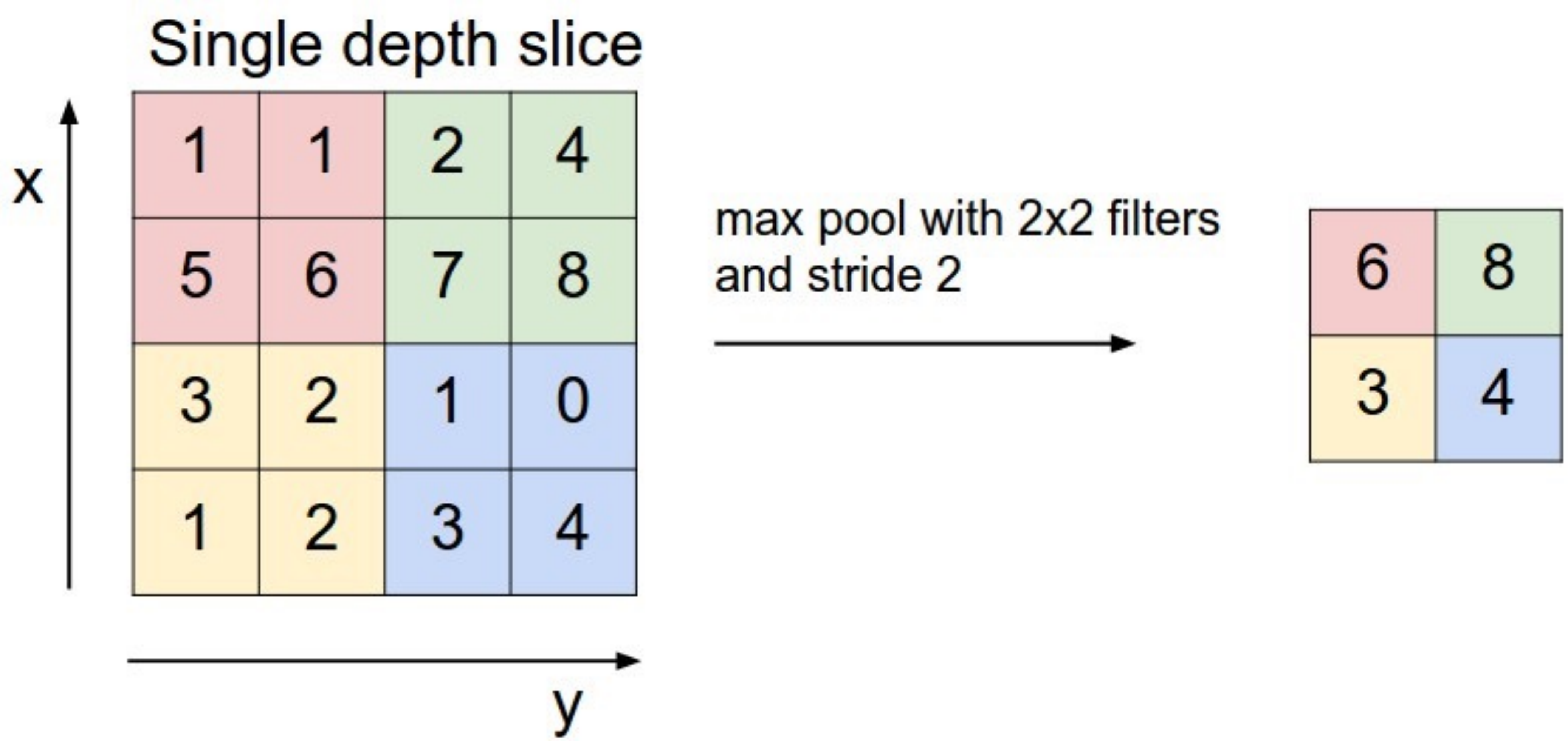
Computation in a neural net

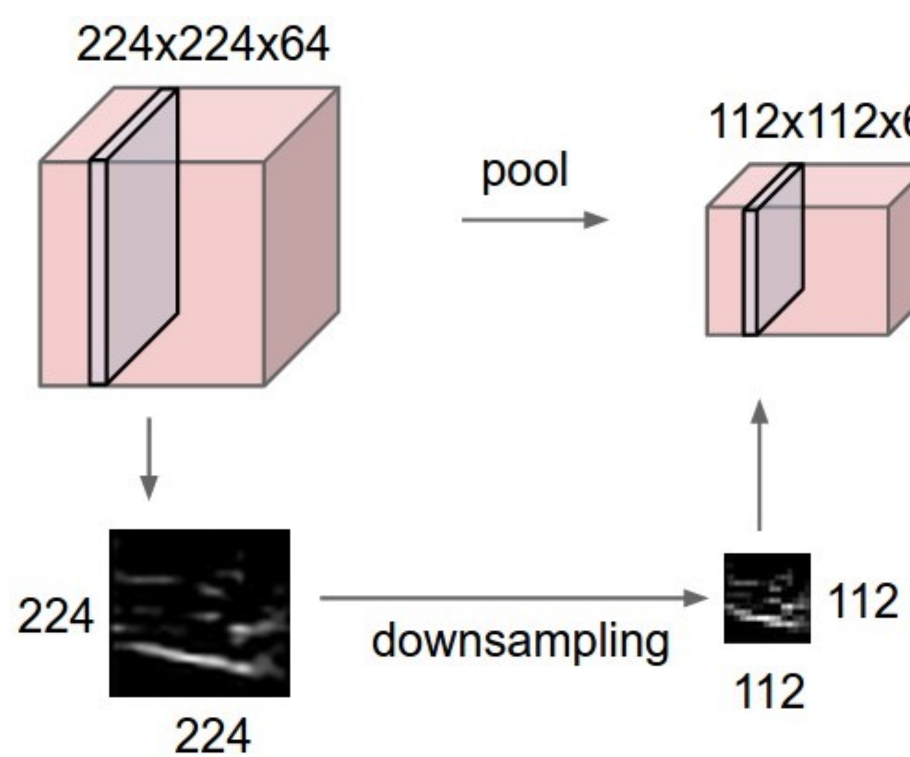


$$y_j = \sum_i w_{ij} x_i \qquad z_k = \max_{j \in \mathcal{N}(j)} g(y_j)$$

i: the i^{th} dimension of x, j; the j^{th} dimension of y

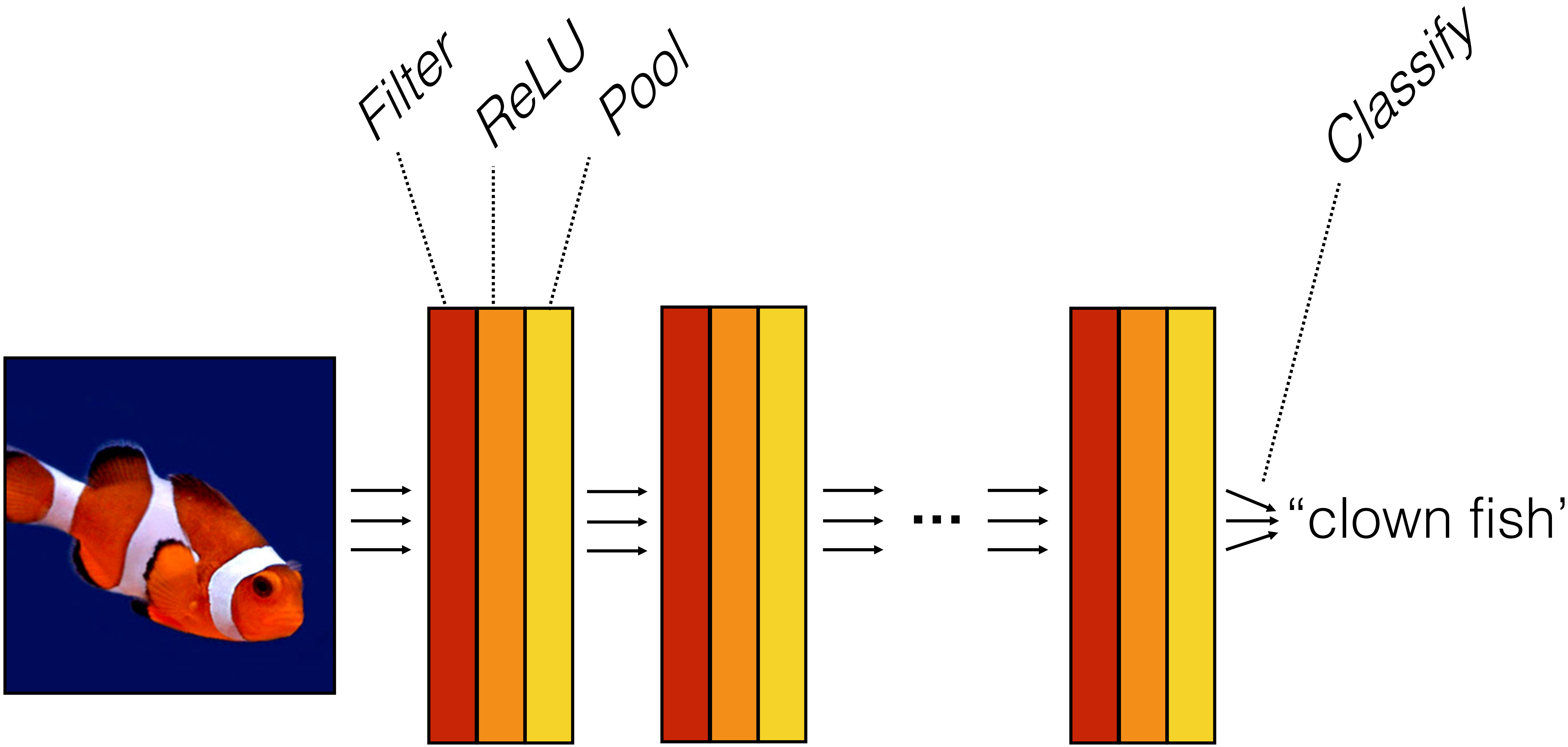
Computation in a neural net





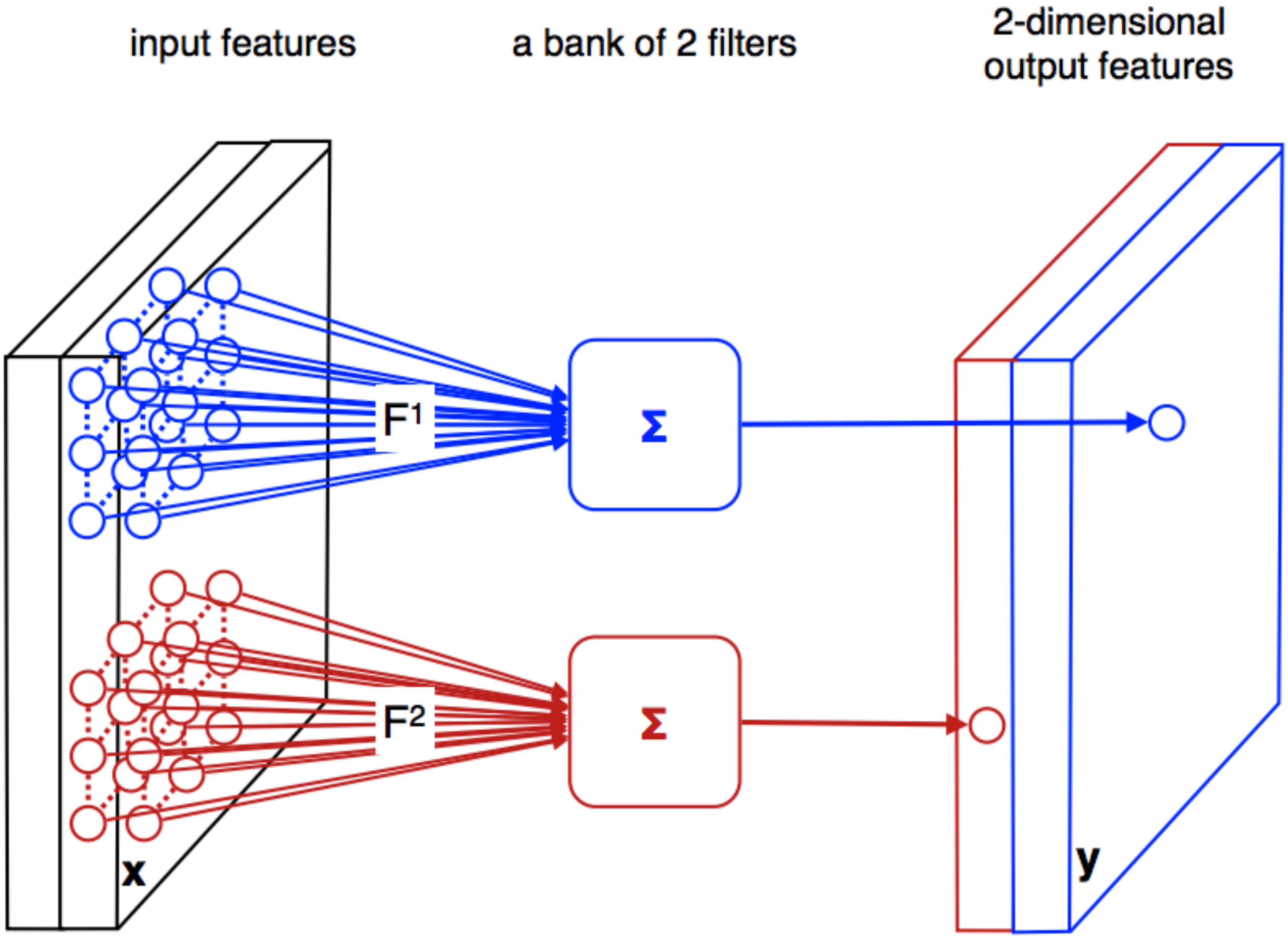
CS231n at Stanford

Computation in a neural net



$$f(\mathbf{x}) = f_L(\dots f_2(f_1(\mathbf{x})))$$

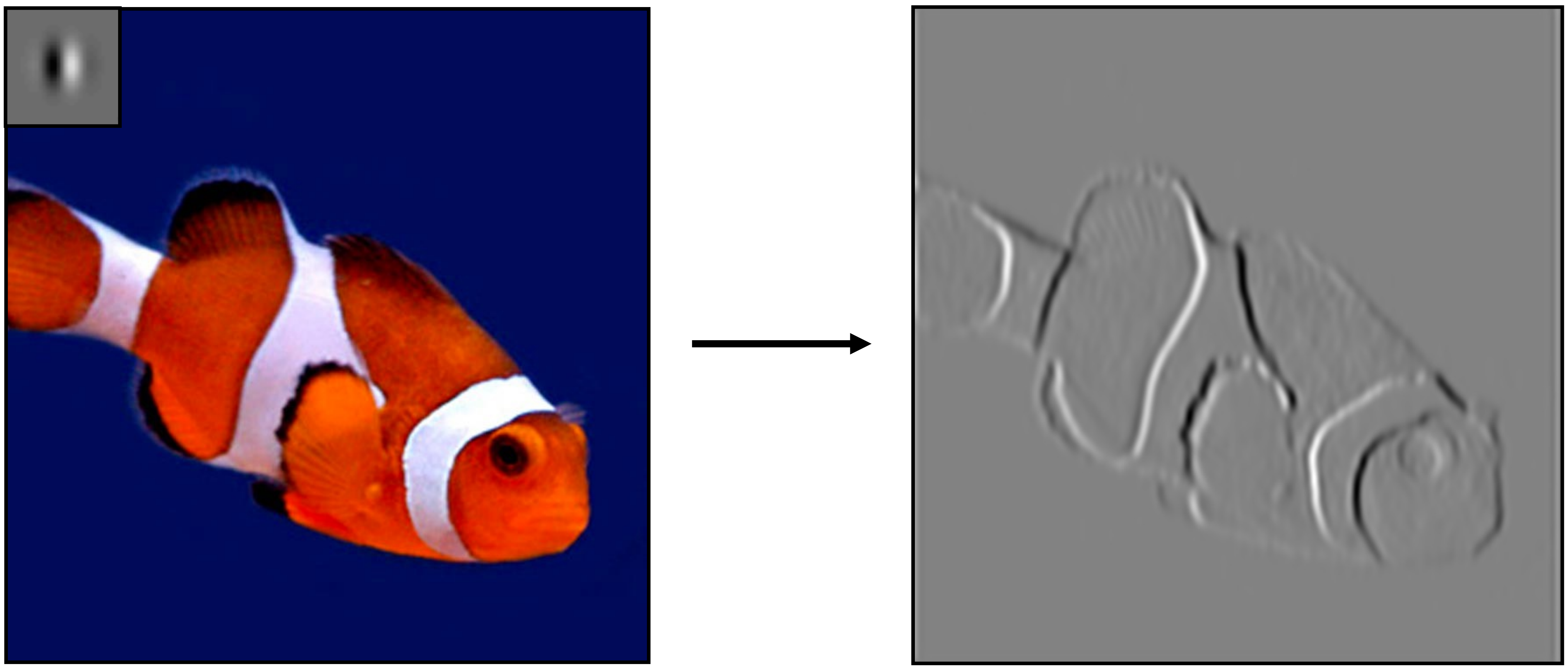
Convolutional Neural Nets



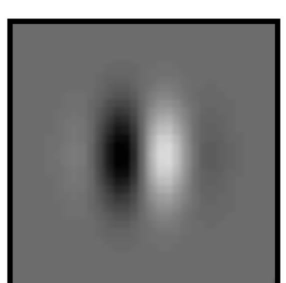
Slide from Andrea Vedaldi

Convolutional Neural Nets

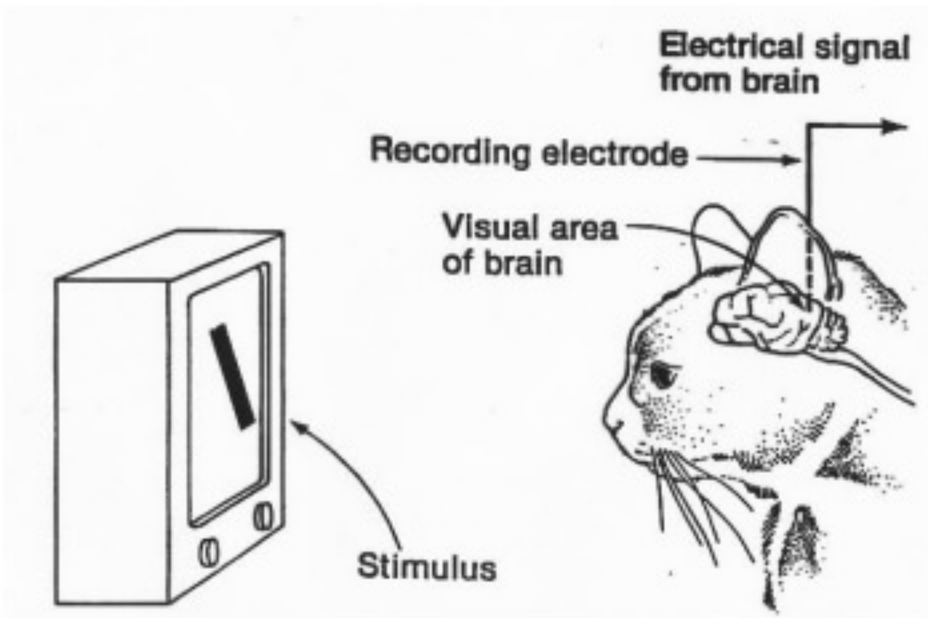
Convolution

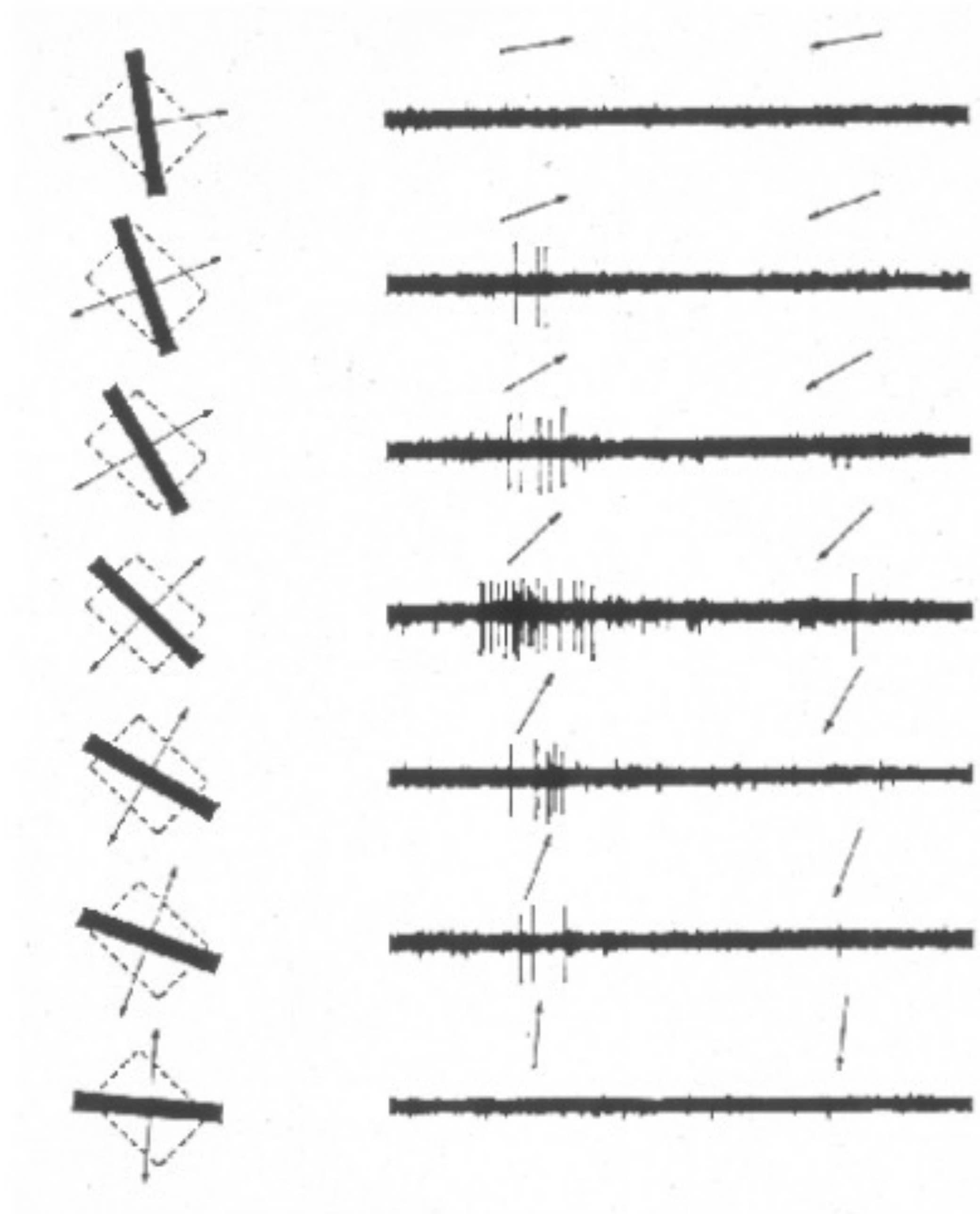


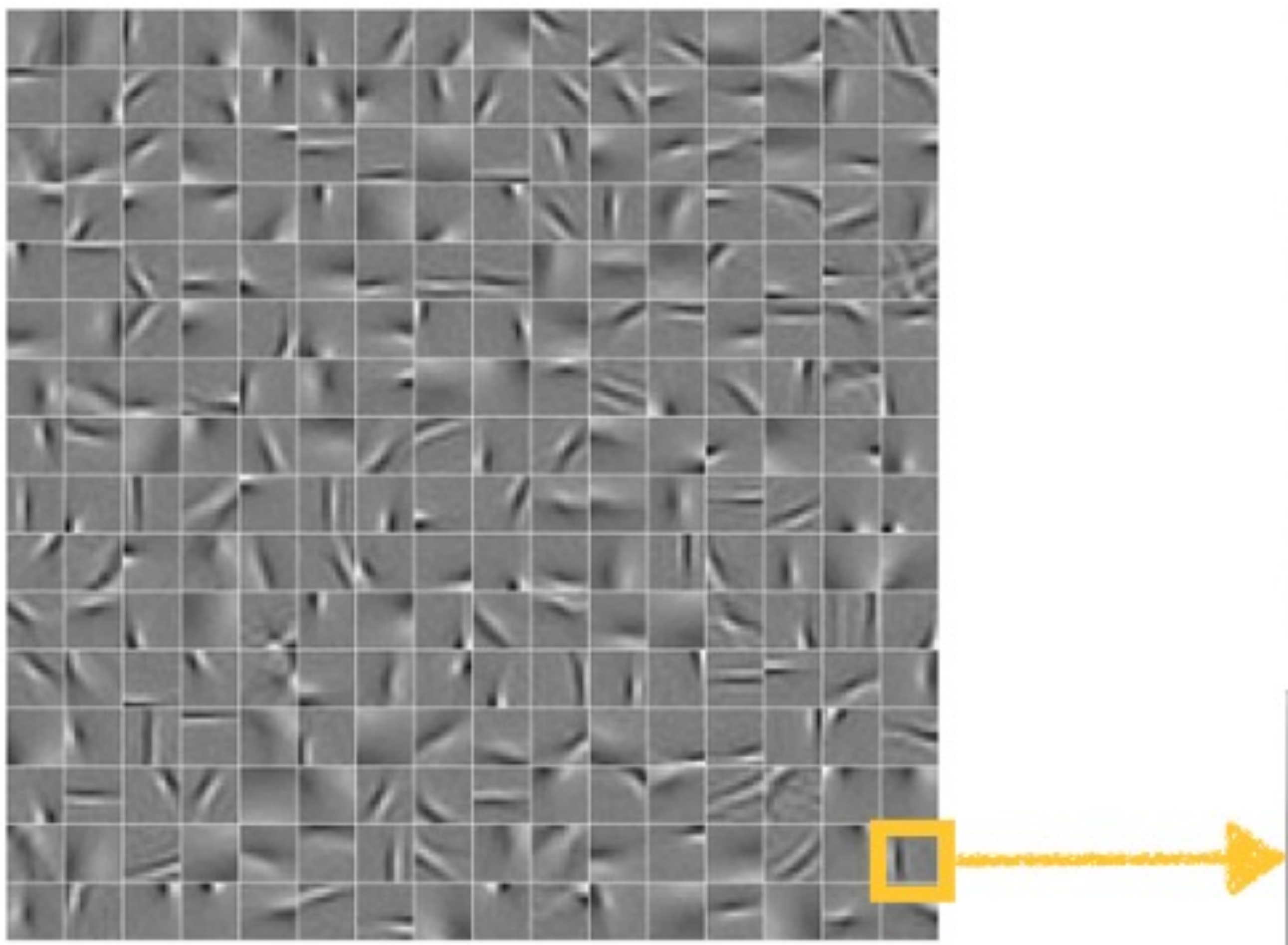
filter



[Hubel and Wiesel 59]





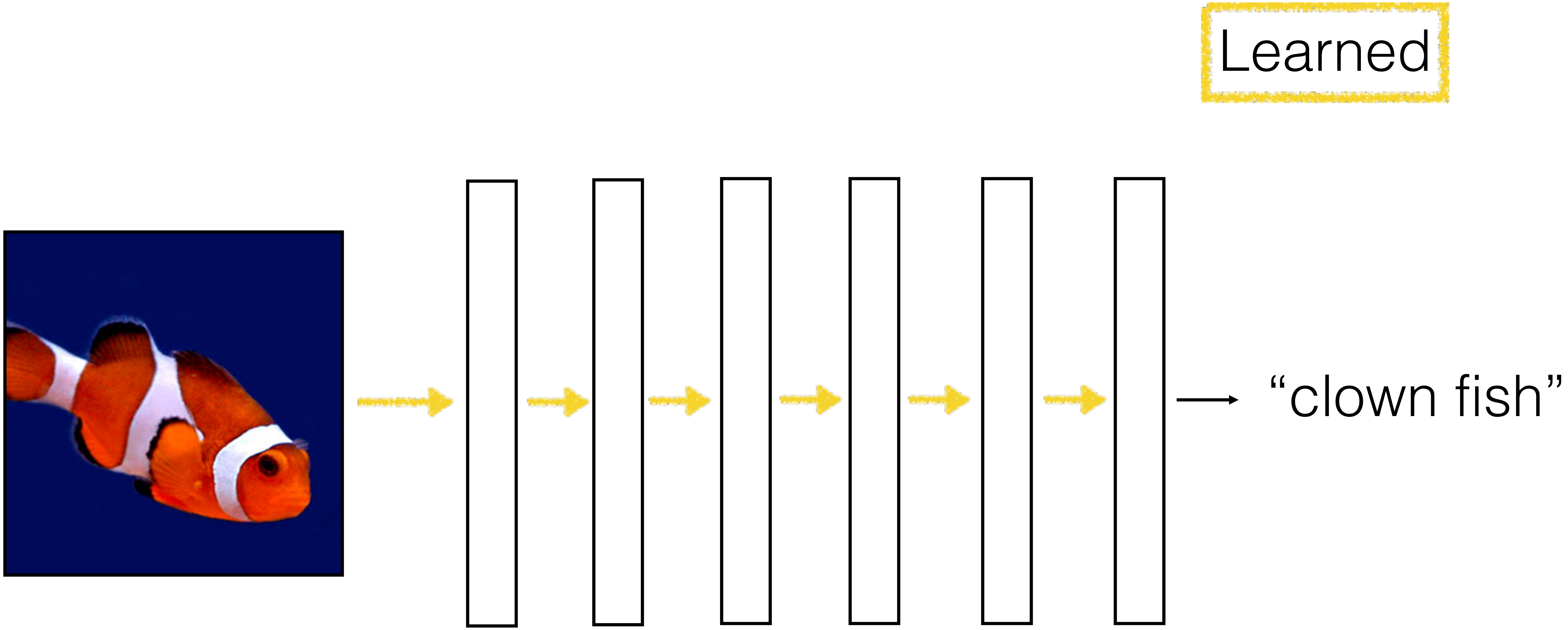


oriented filter

Slide from Andrea Vedaldi

Computation in a neural net

Learning with deep nets



Learning with deep nets

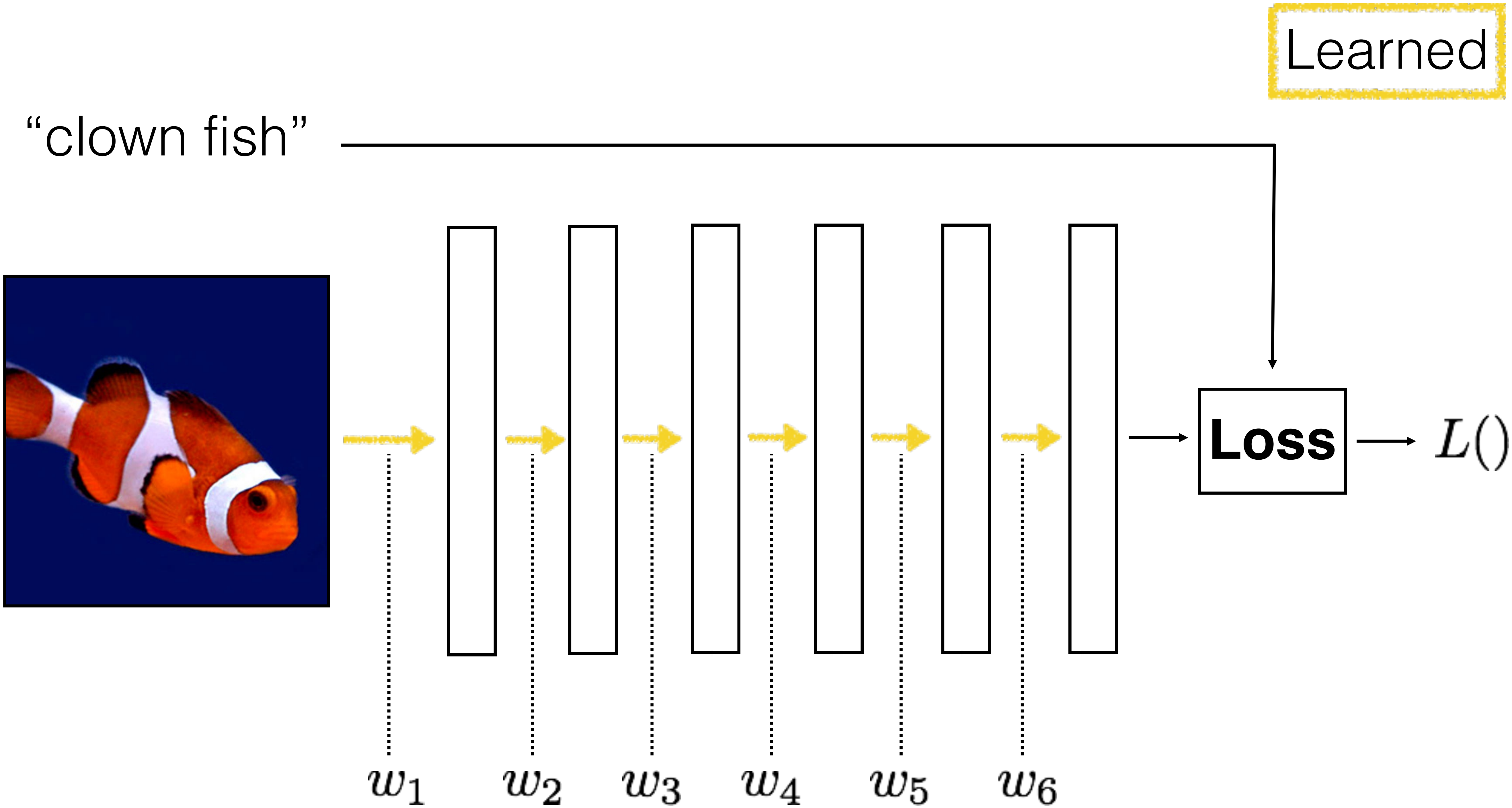
→ "clown fish"

→ "grizzly bear"

→ "chameleon"

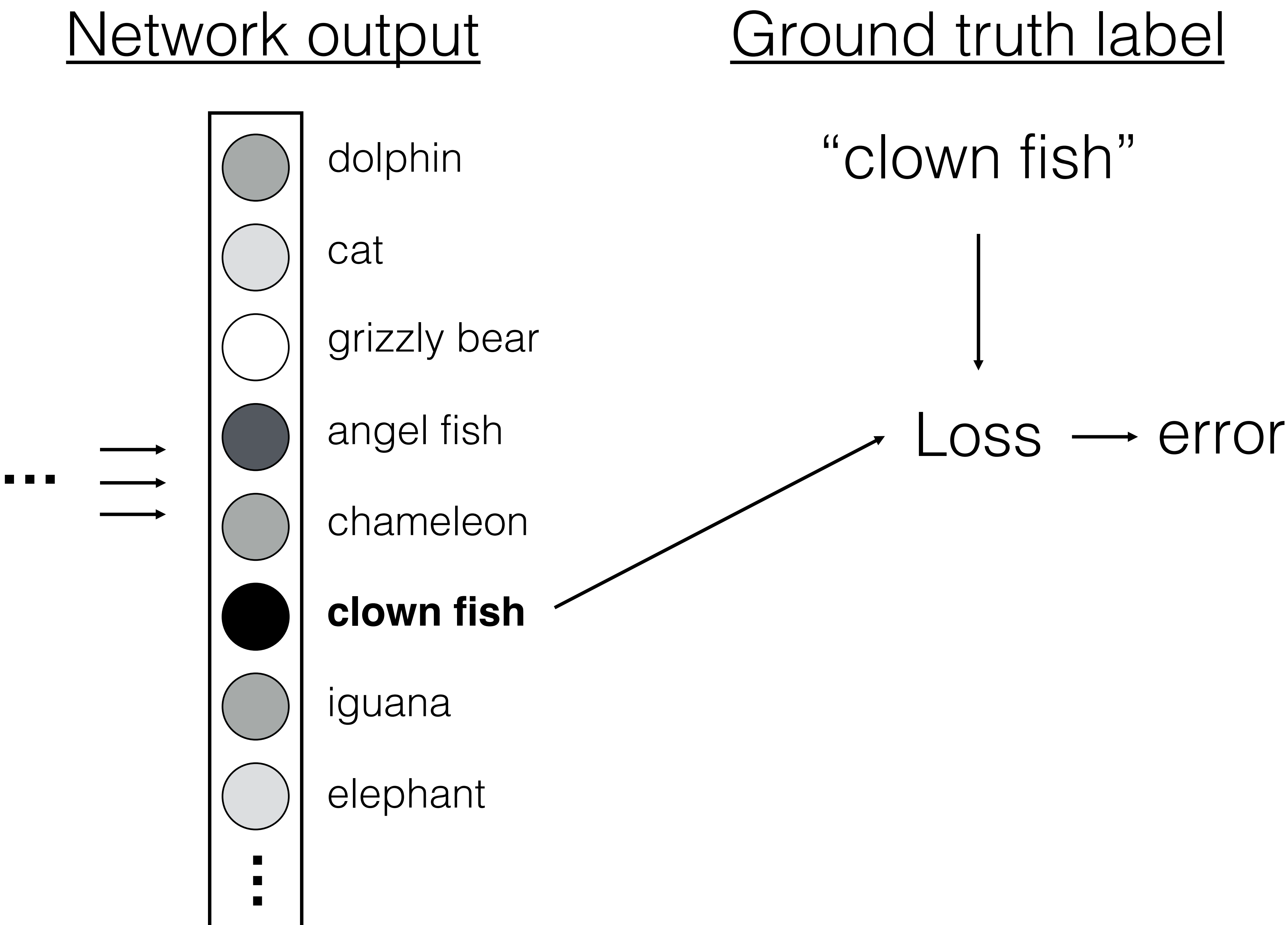
Train network to associate the right label with each image

Learning with deep nets

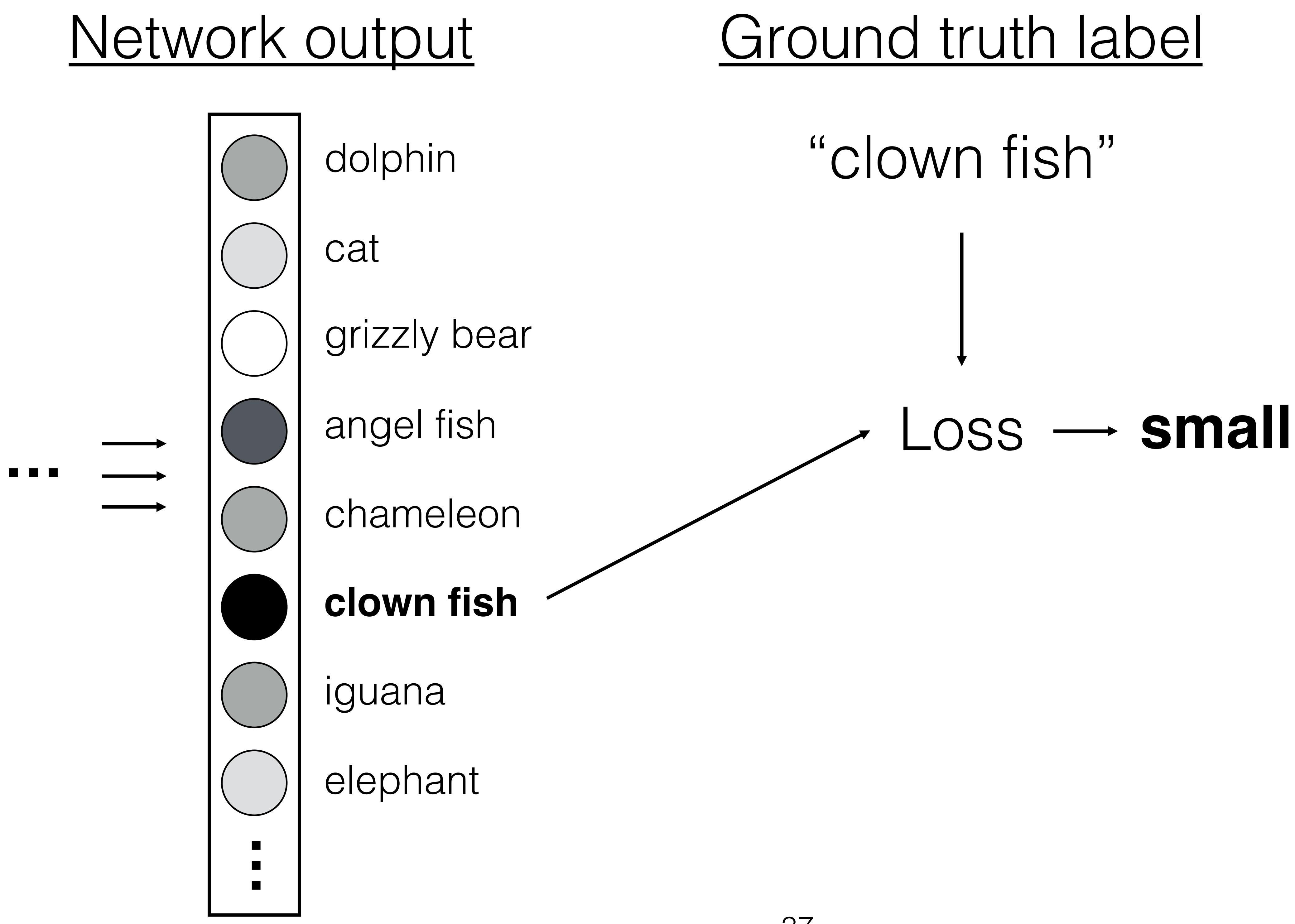


$$\underset{\mathbf{w}}{\operatorname{argmin}} \quad L(w_1, \ldots, w_6)$$

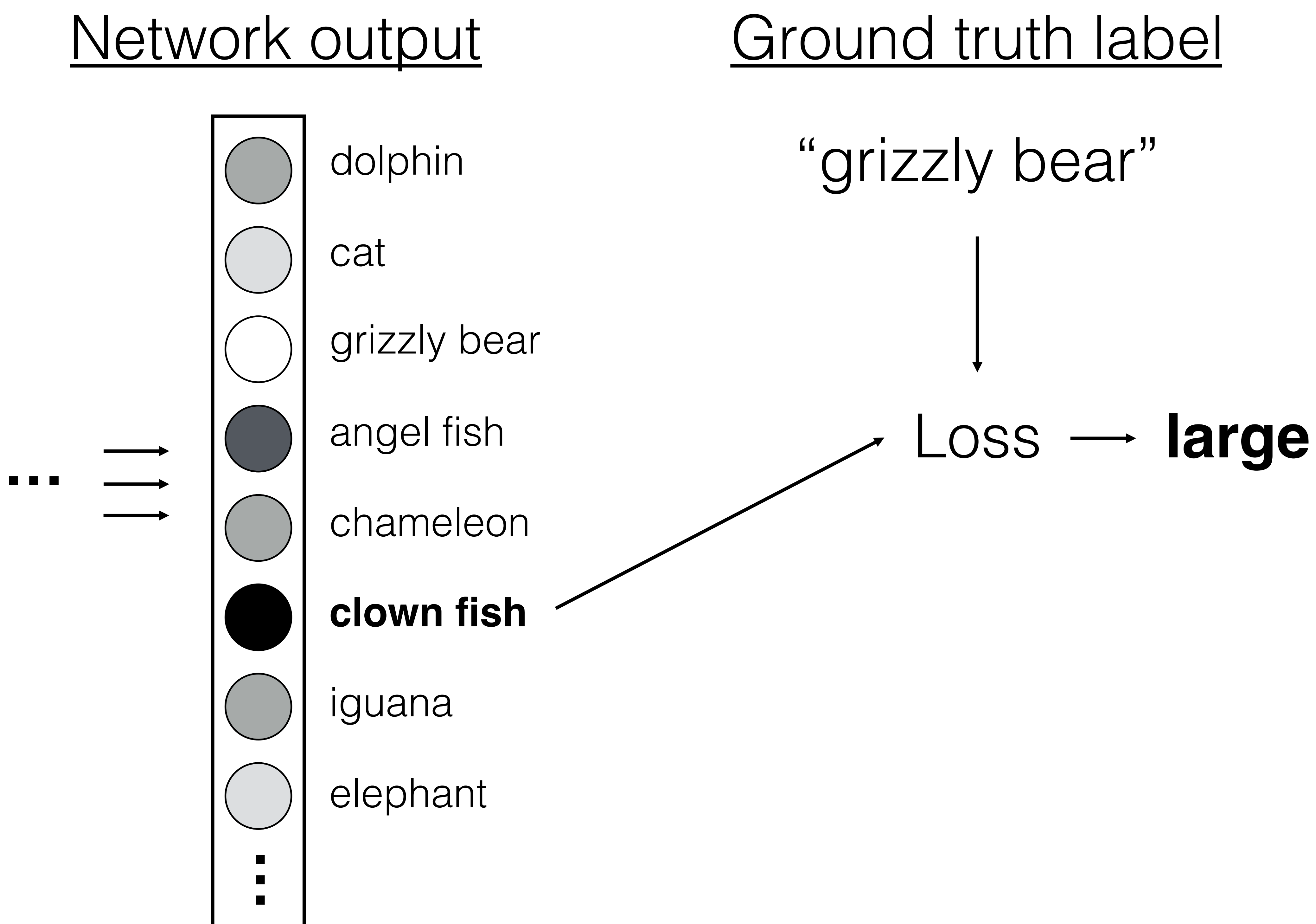
Loss function



Loss function

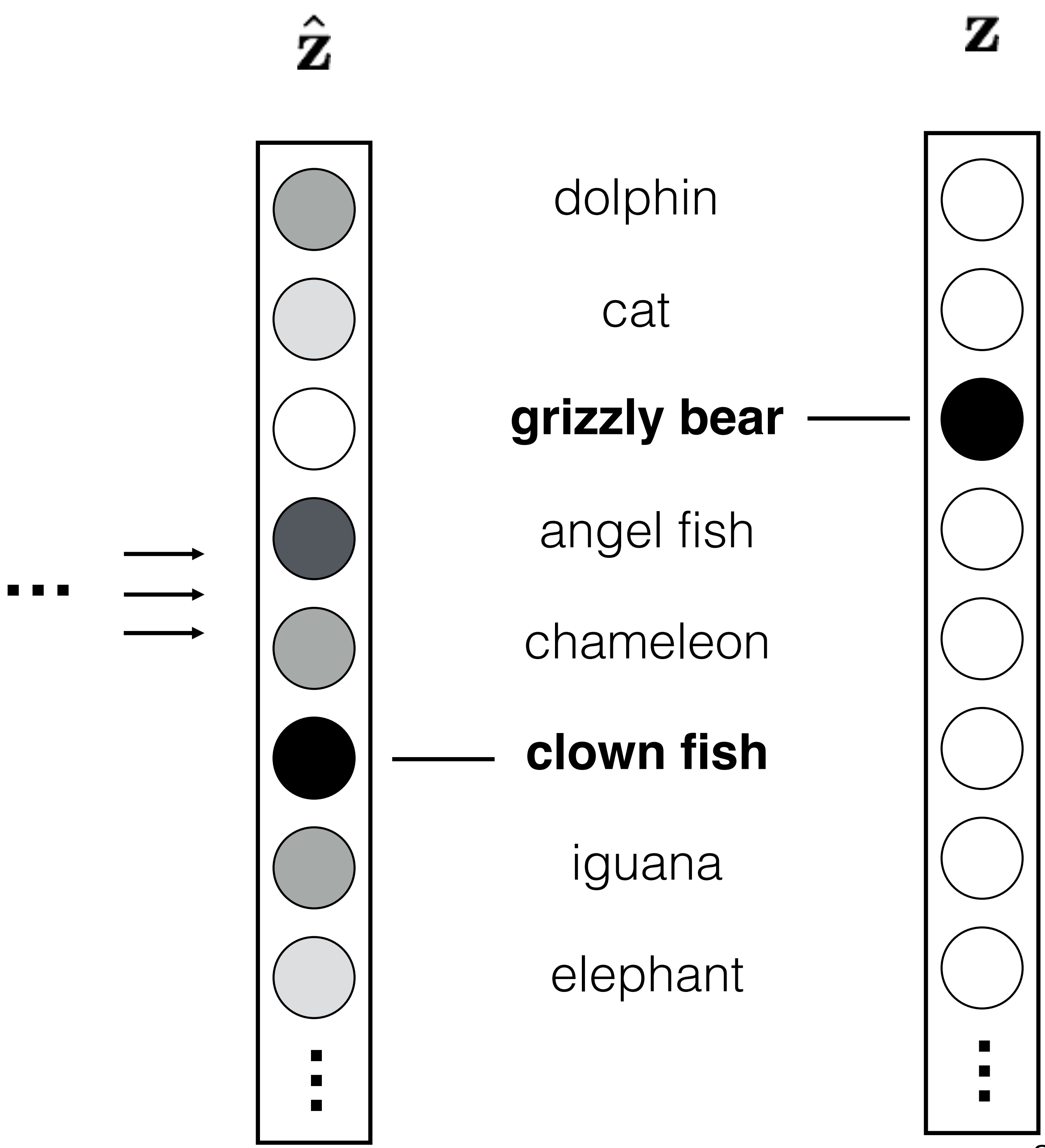


Loss function



Loss function for classification

Network output   Ground truth label



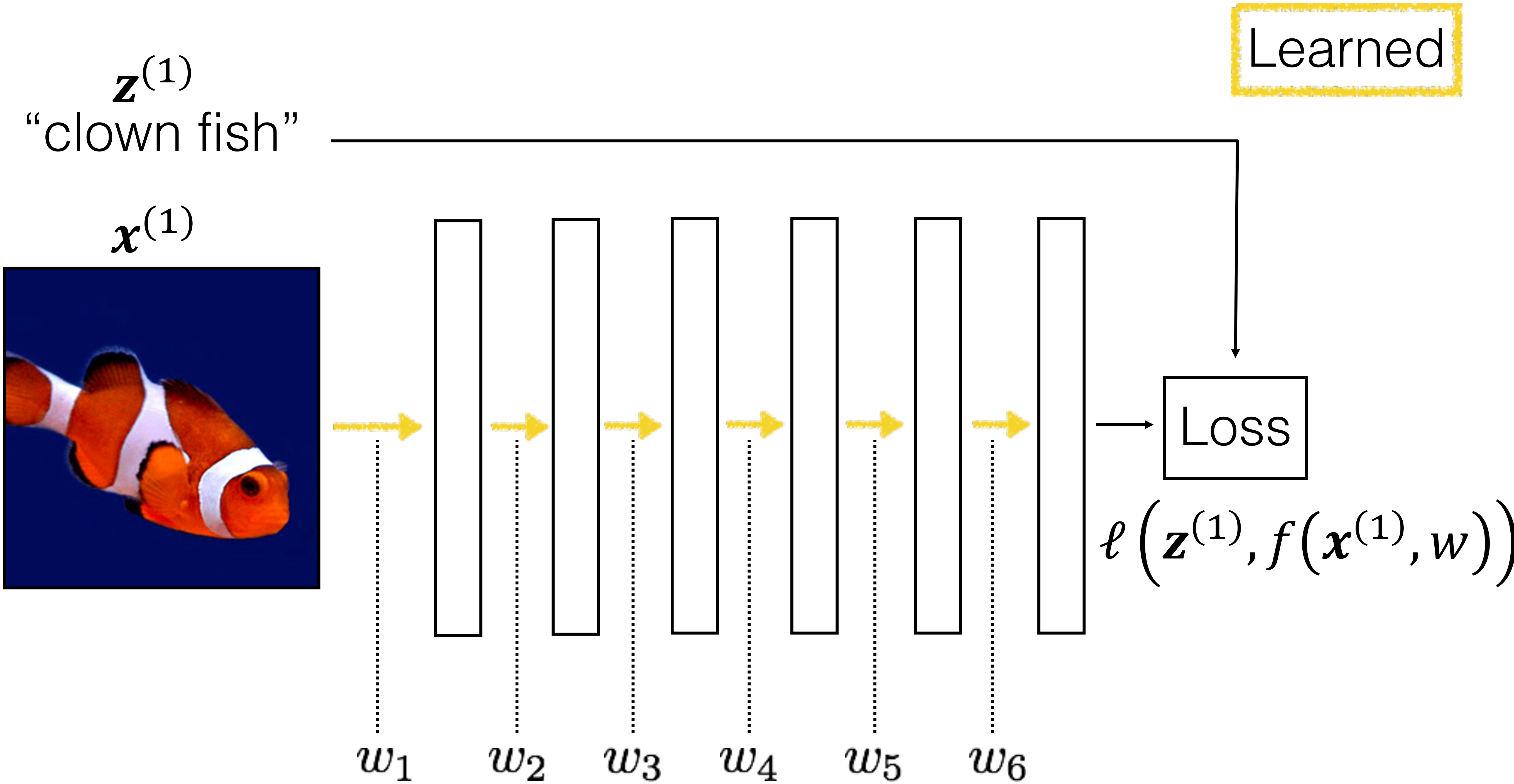
Probability of the observed data under the model

$$H(\hat{z}, z) = -\sum_{c} z_c \log \hat{z}_c$$

Cross-entropy loss

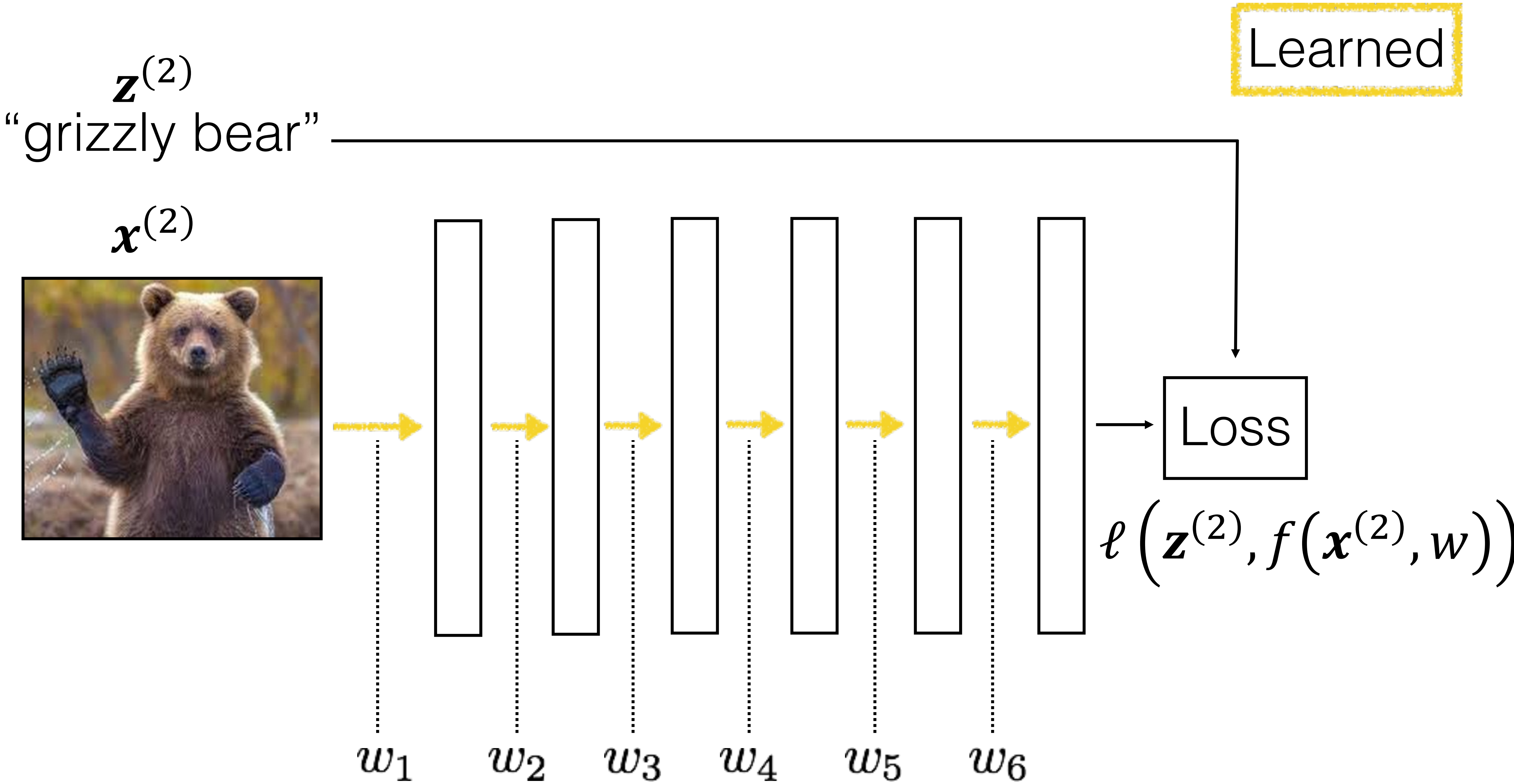
c is the c^{th} class in the output

Learning with deep nets



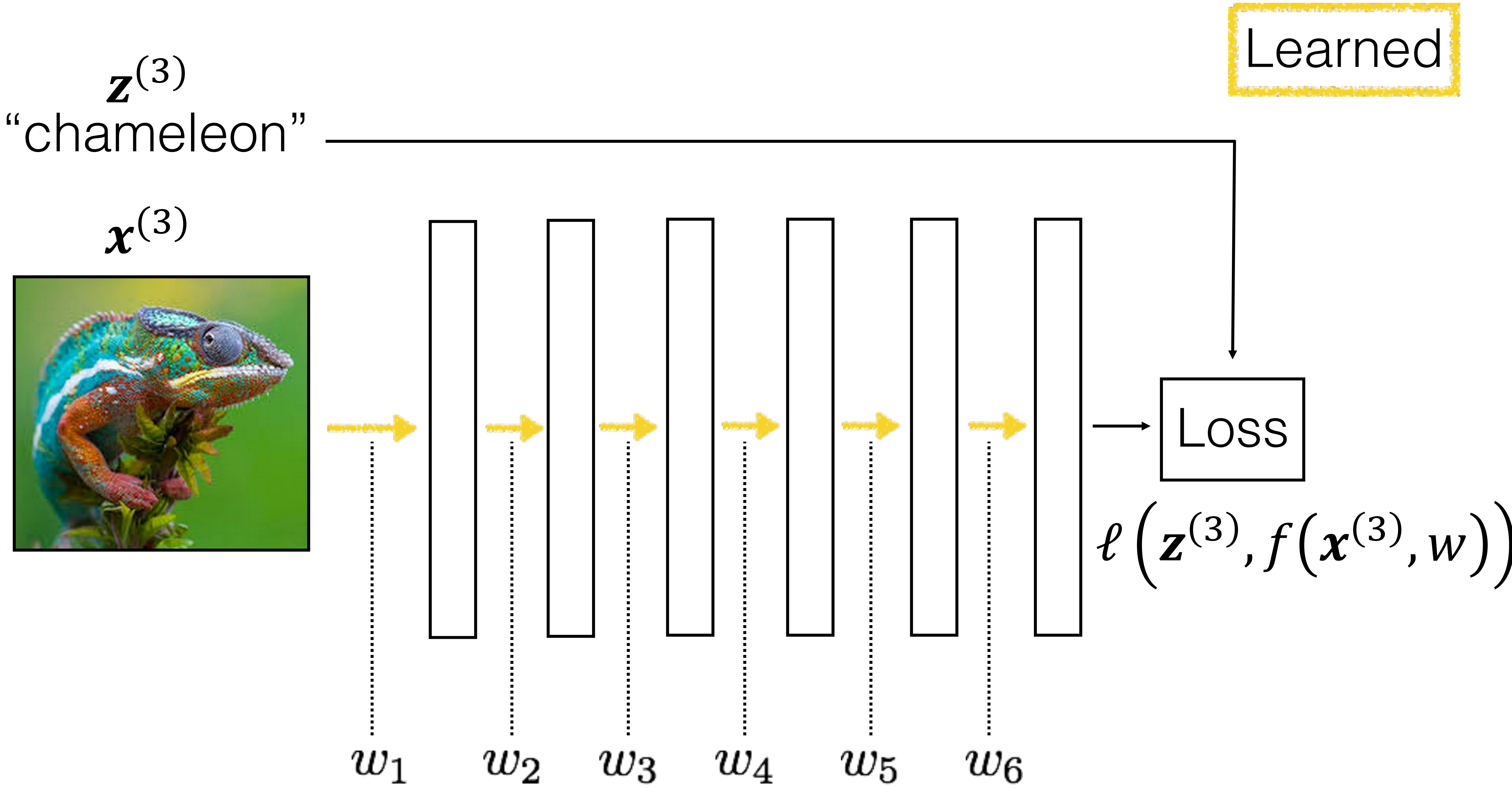
$\boldsymbol{x}^{(1)}, \boldsymbol{z}^{(1)}$ *is the input and label of the 1st training image*

Learning with deep nets



$\boldsymbol{x}^{(2)}, \boldsymbol{z}^{(2)}$ *is the input and label of the 2nd training image*

Learning with deep nets



$$argmin_w \sum_i \ell\left(z^{(i)}, f\left(x^{(i)}, w\right)\right)$$

Gradient descent

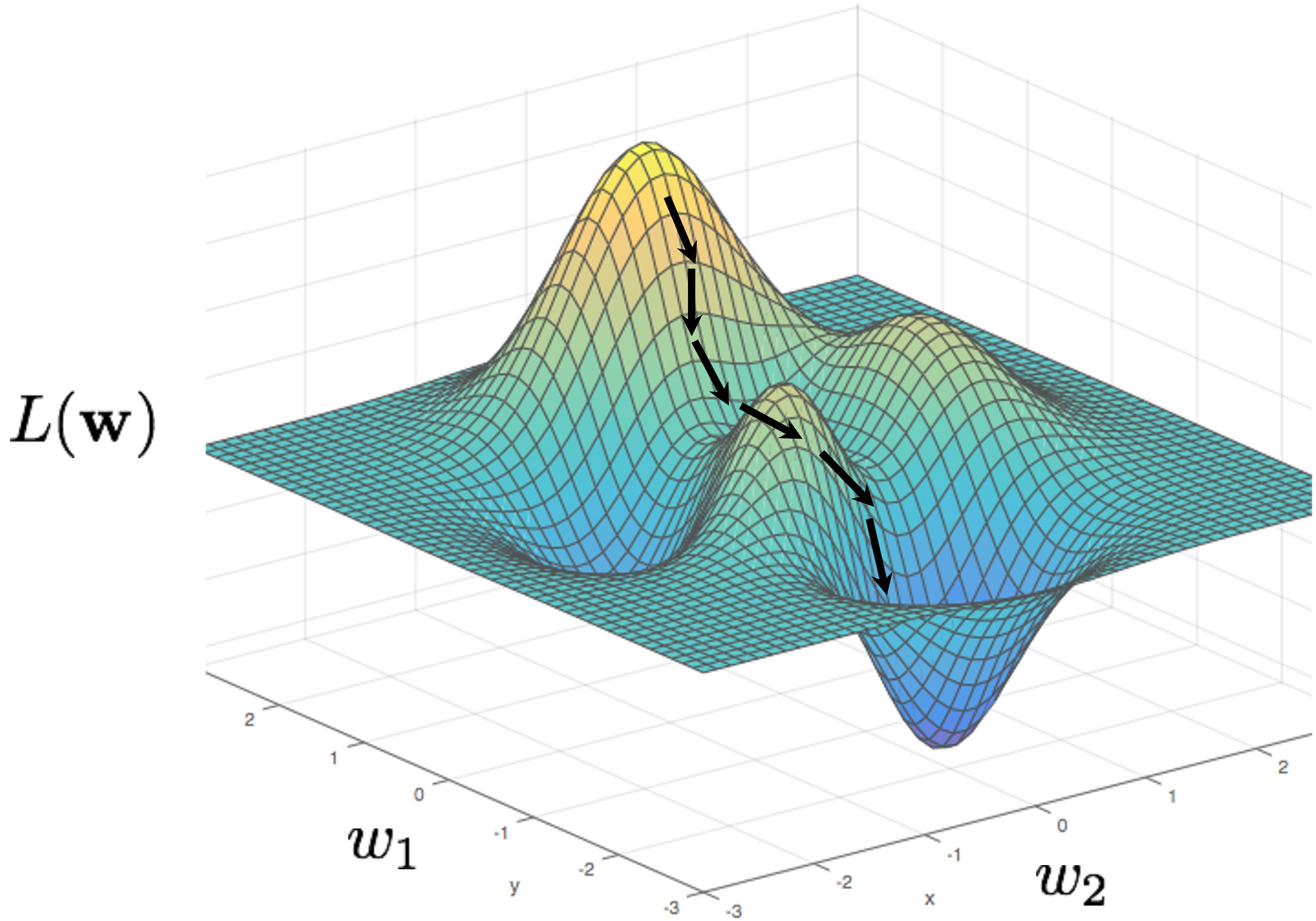
$$argmin_w \sum_i \ell\left(z^{(i)}, f(x^{(i)}, w)\right) = argmin_w L(w)$$

One iteration of gradient descent:

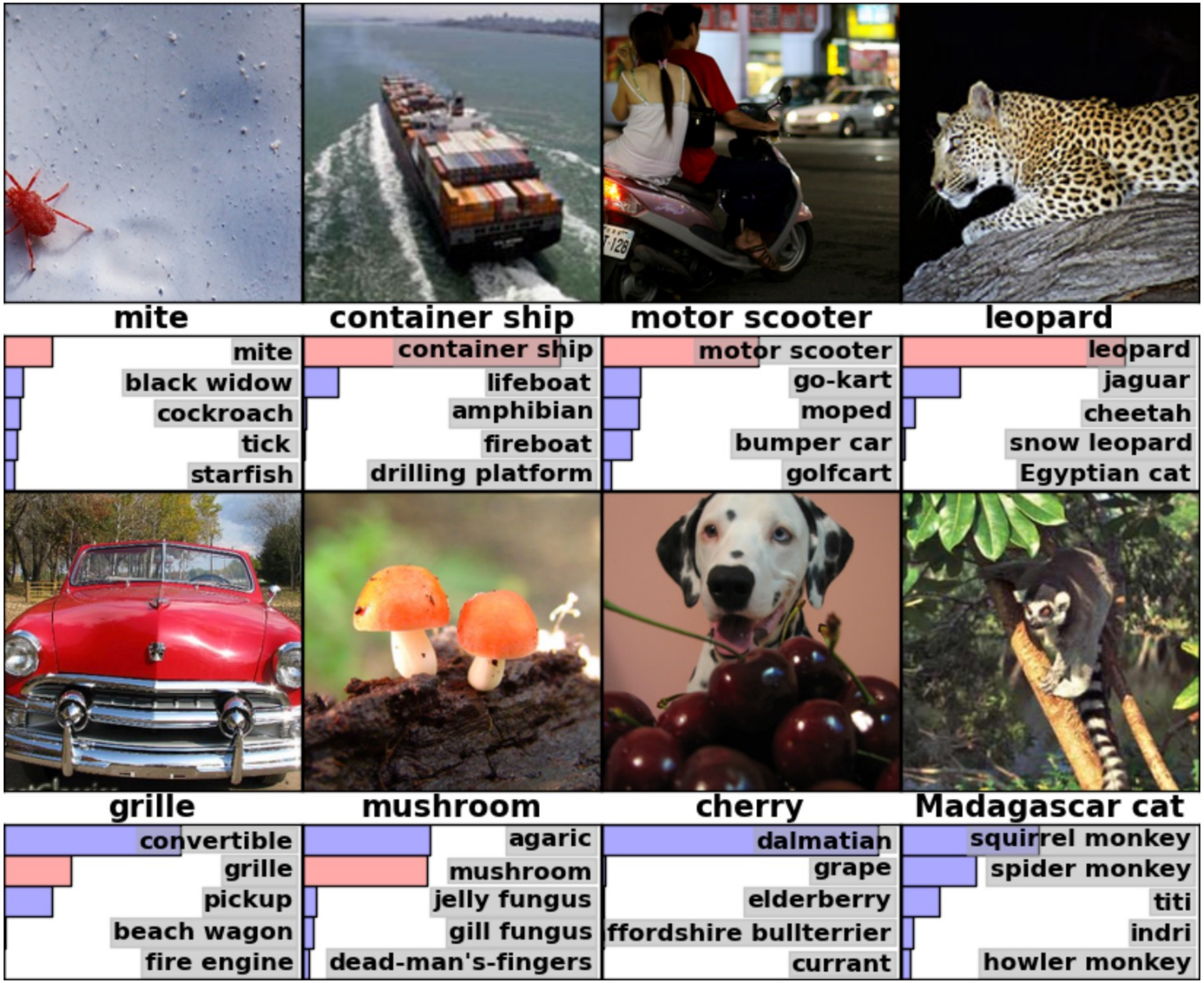
$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta_t \frac{\partial L(\mathbf{w^t})}{\partial \mathbf{w}}$$

learning rate

Gradient descent

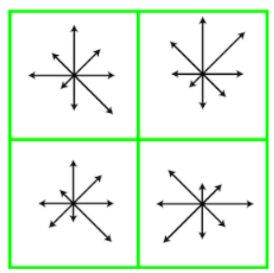


$$p(c|\mathbf{x})$$

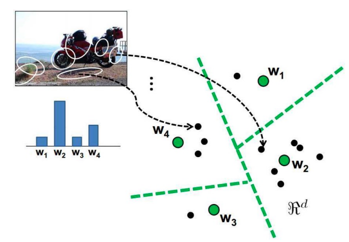


[Krizhevsky et al. NIPS 2012]

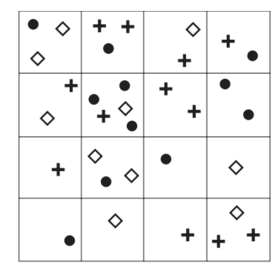
Computer Vision before 2012



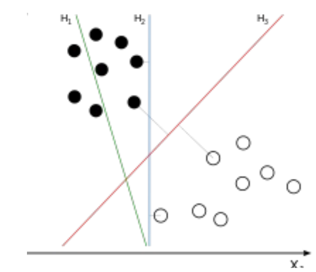
Features



Clustering



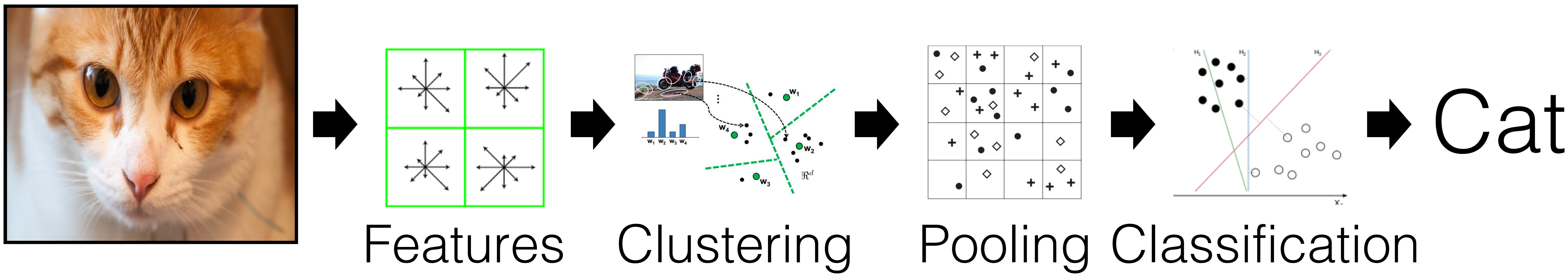
Pooling

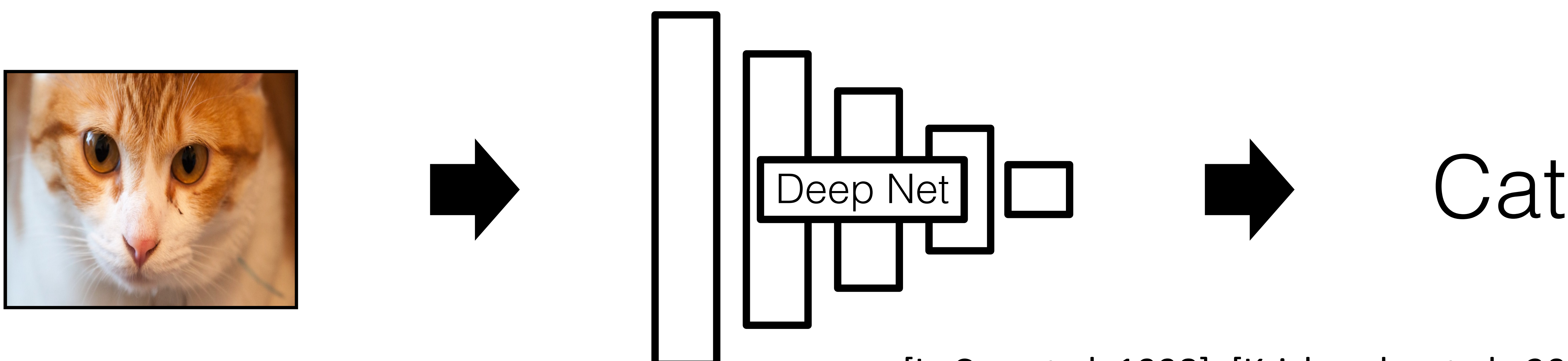


Classification

Cat

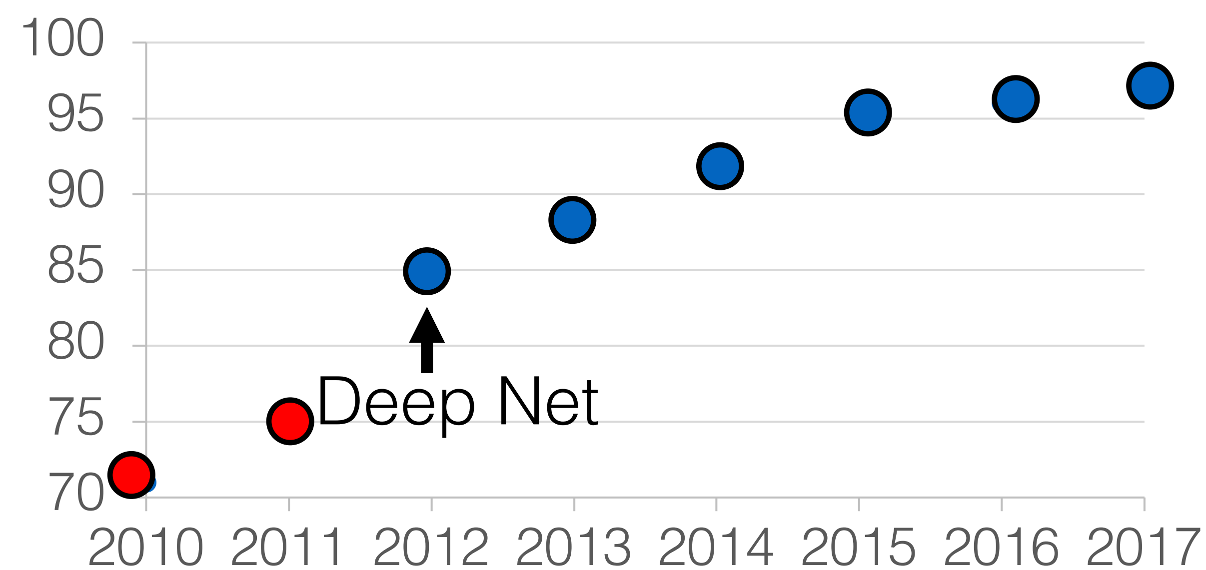
Computer Vision Now



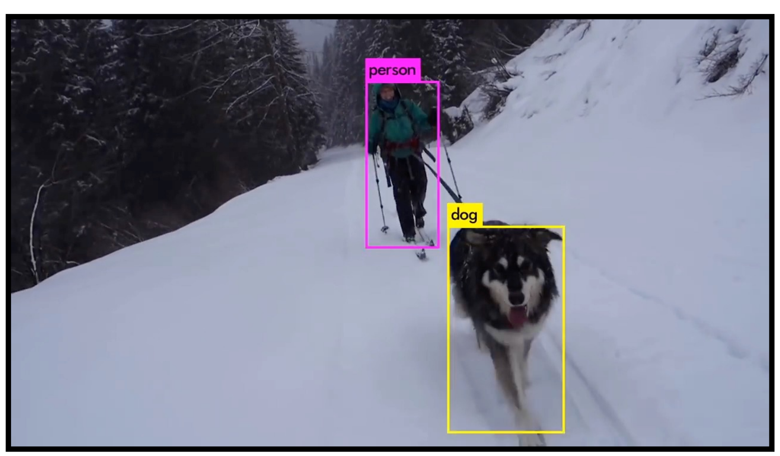


[LeCun et al, 1998], [Krizhevsky et al, 2012]

Deep Learning for Computer Vision

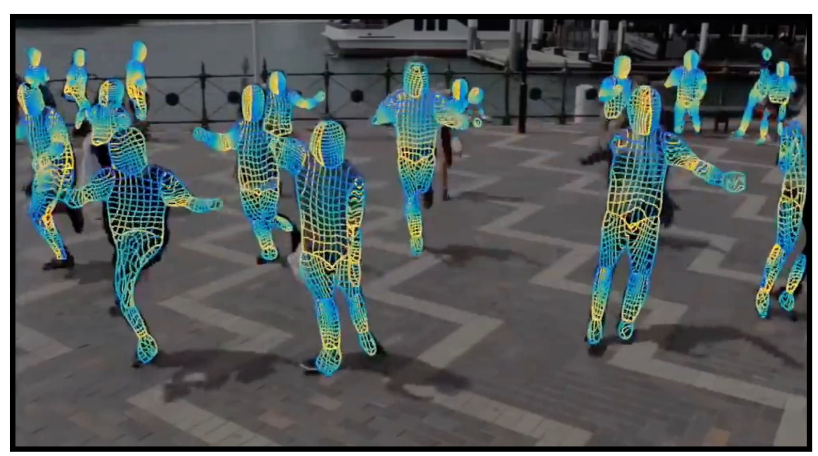


Top 5 **accuracy** on ImageNet benchmark



[Redmon et al., 2018]

Object detection



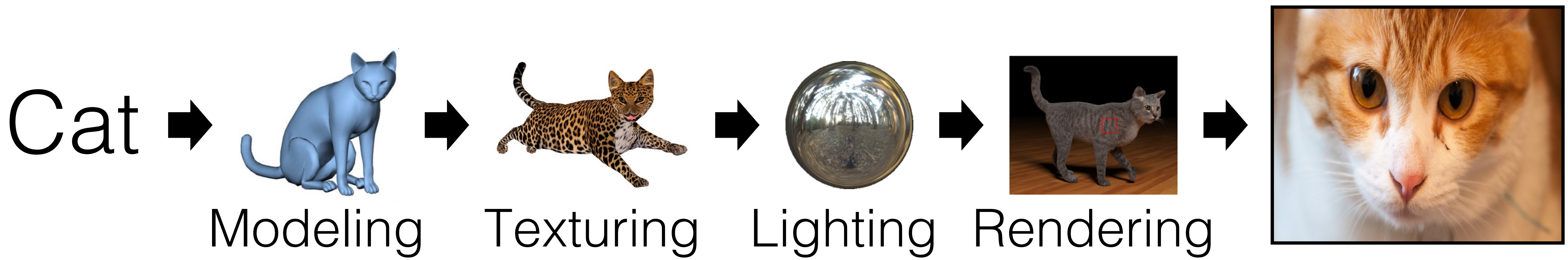
[Güler et al., 2018]

Human understanding

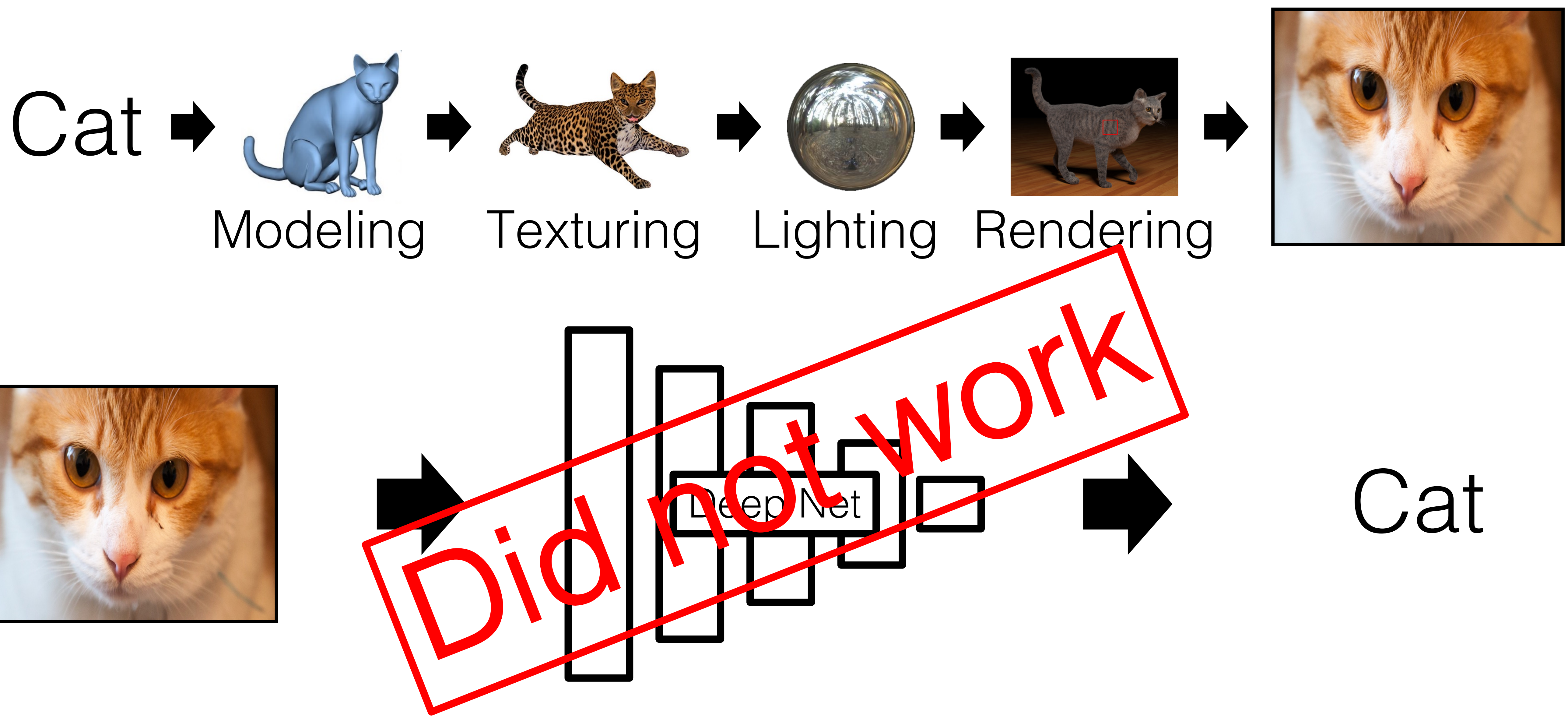
[Zhao et al., 2017]

Autonomous driving

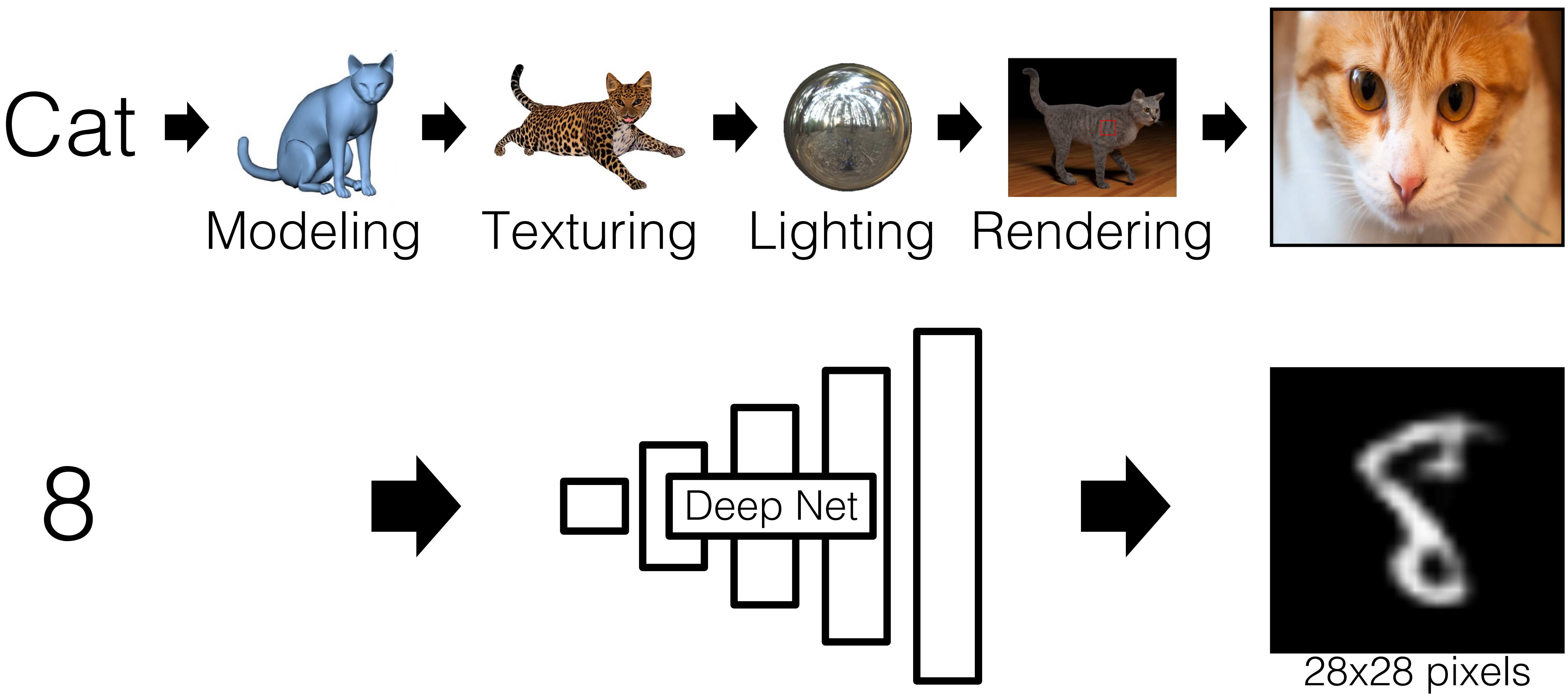
Can Deep Learning **Help** Graphics?



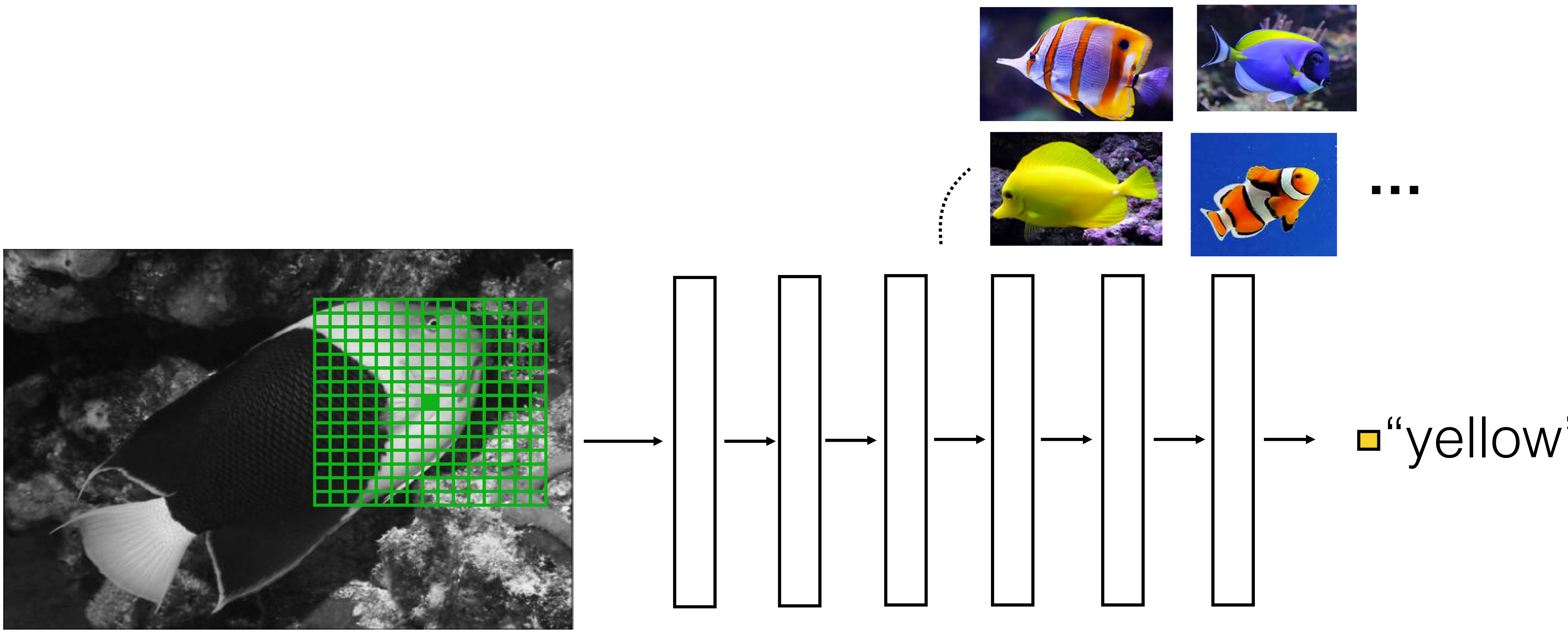
Can Deep Learning **Help** Graphics?



Generating images is hard!

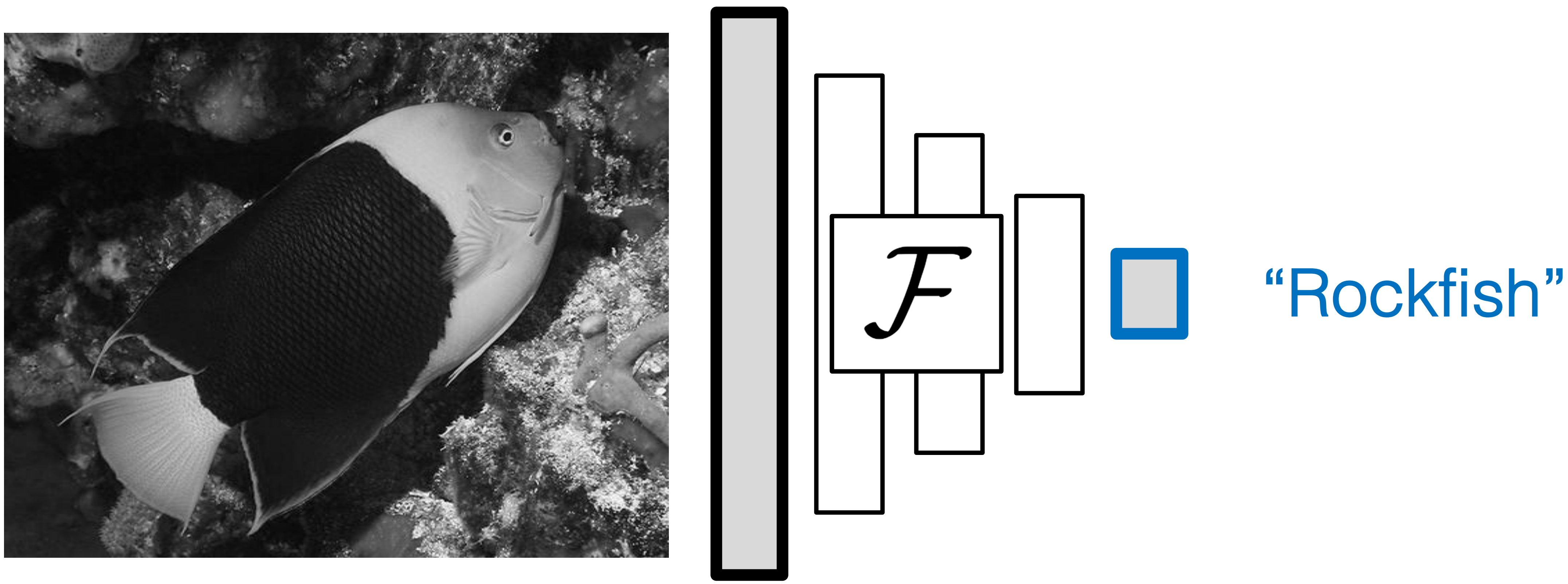


from Classification to Generation

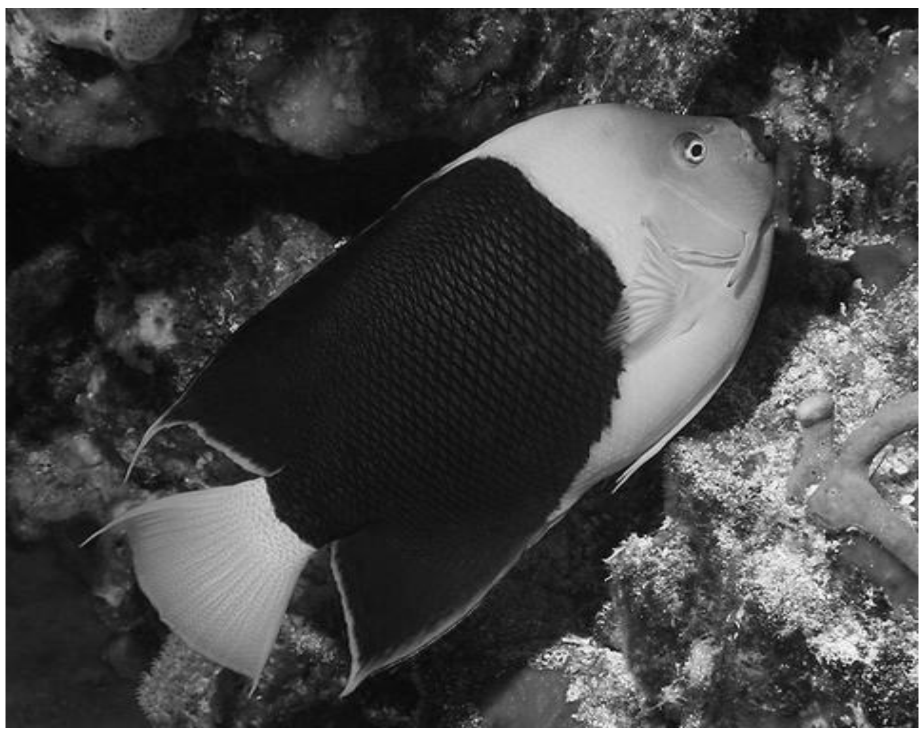


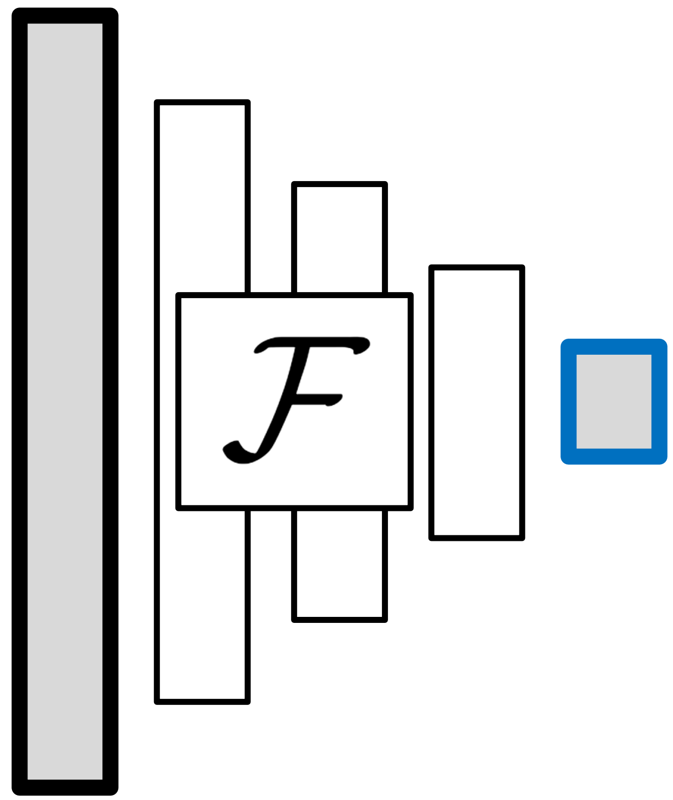
Predicting the color value of an output pixel given a patch

Discriminative Deep Networks



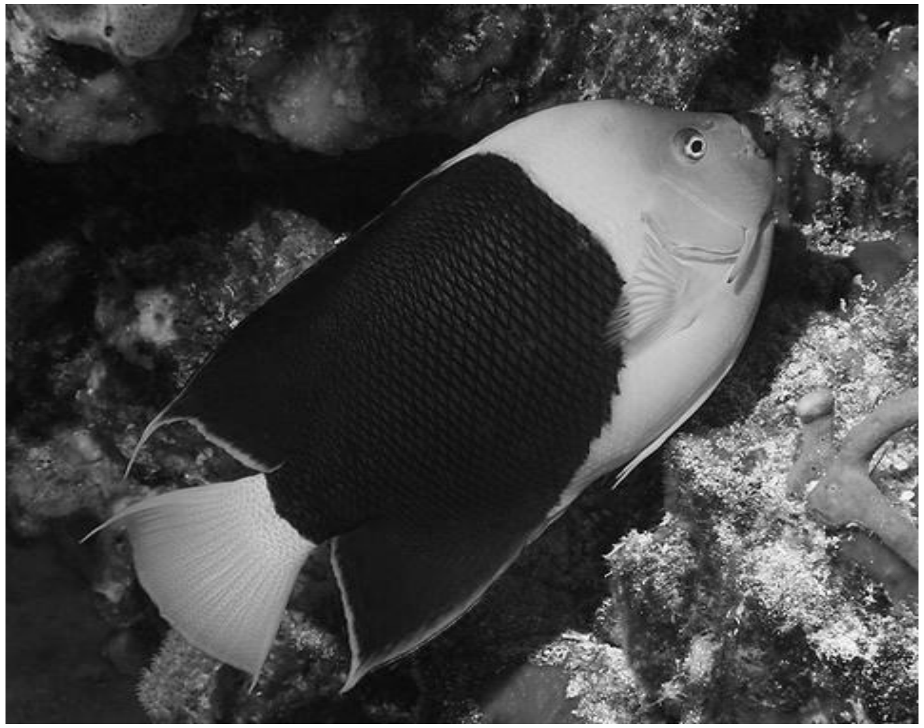
Discriminative Deep Networks

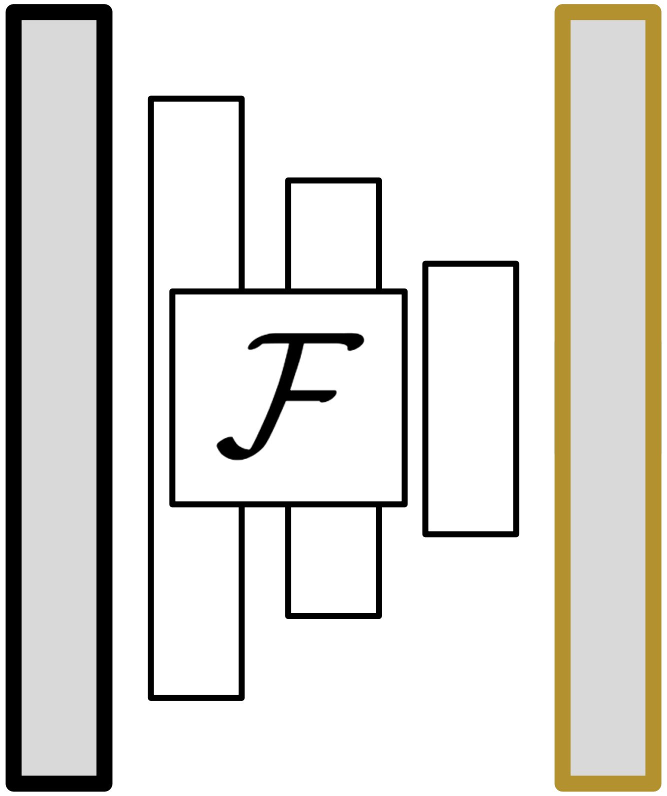




Raw, Unlabeled Pixels

Generative Deep Networks

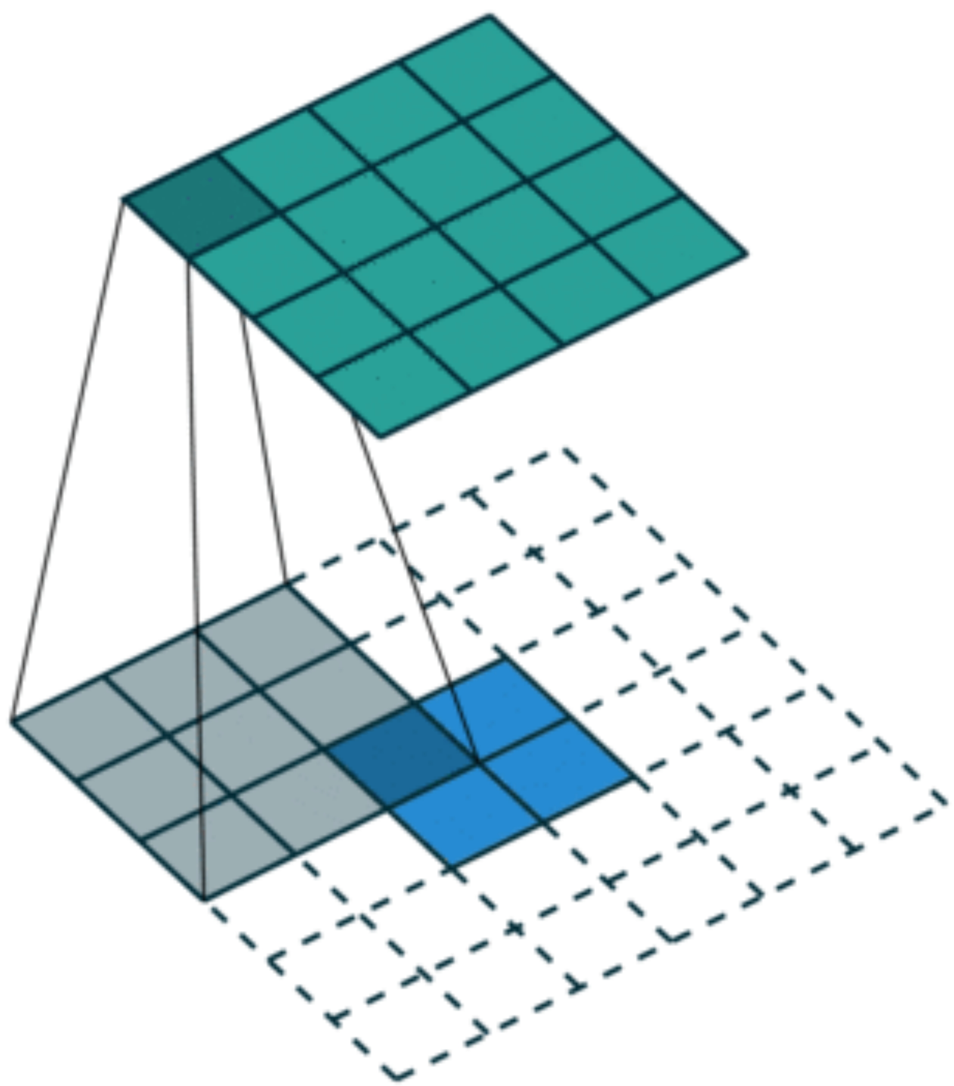




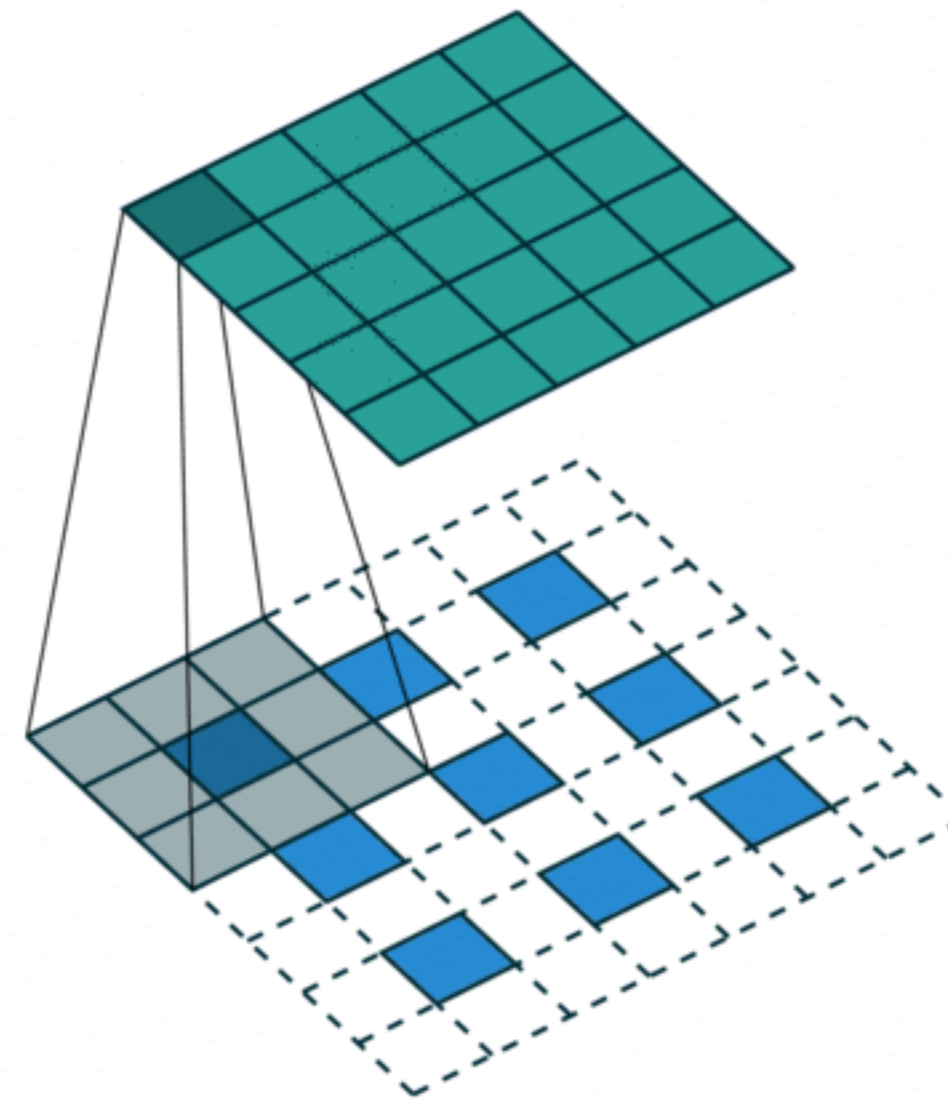
Raw, Unlabeled Pixels

Better Architectures

Fractionally-strided Convolution

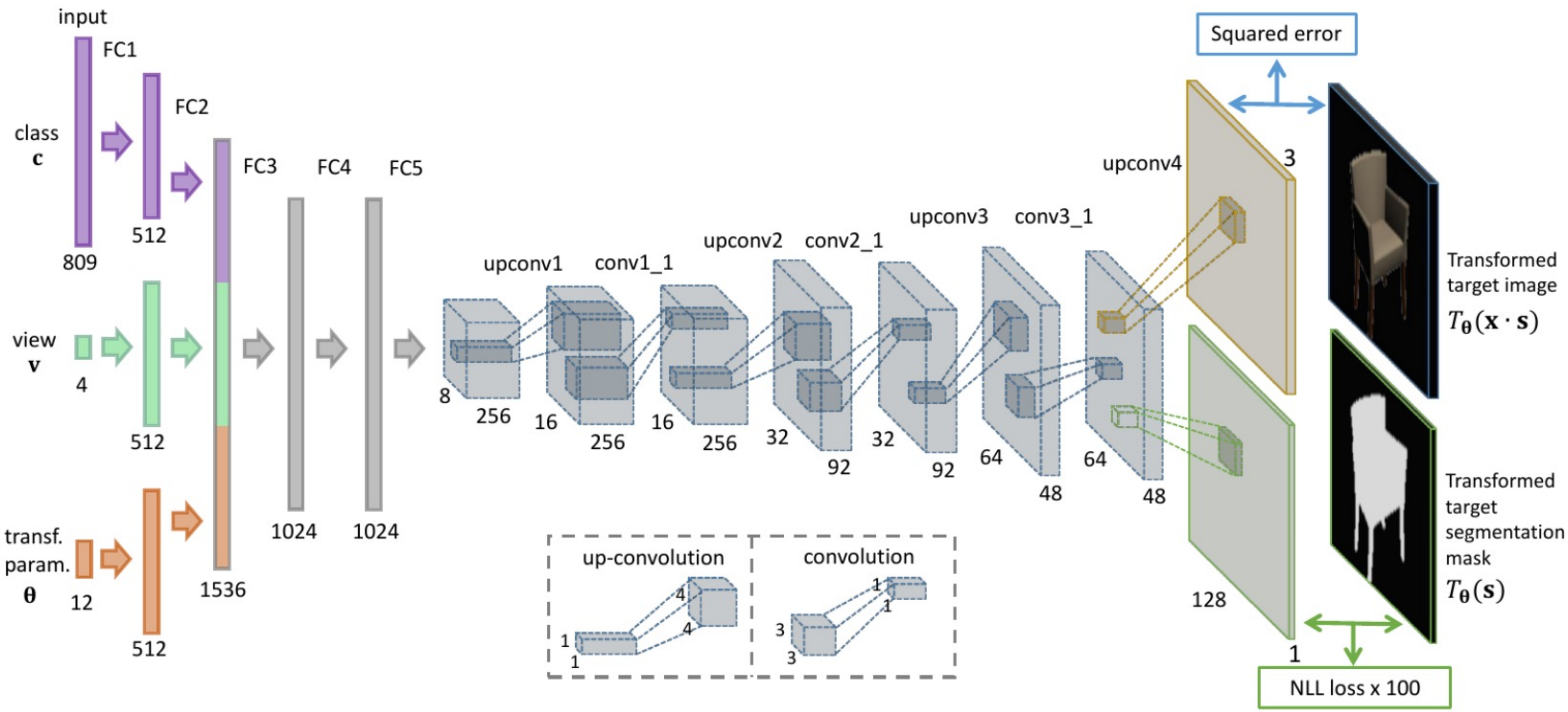


Regular conv



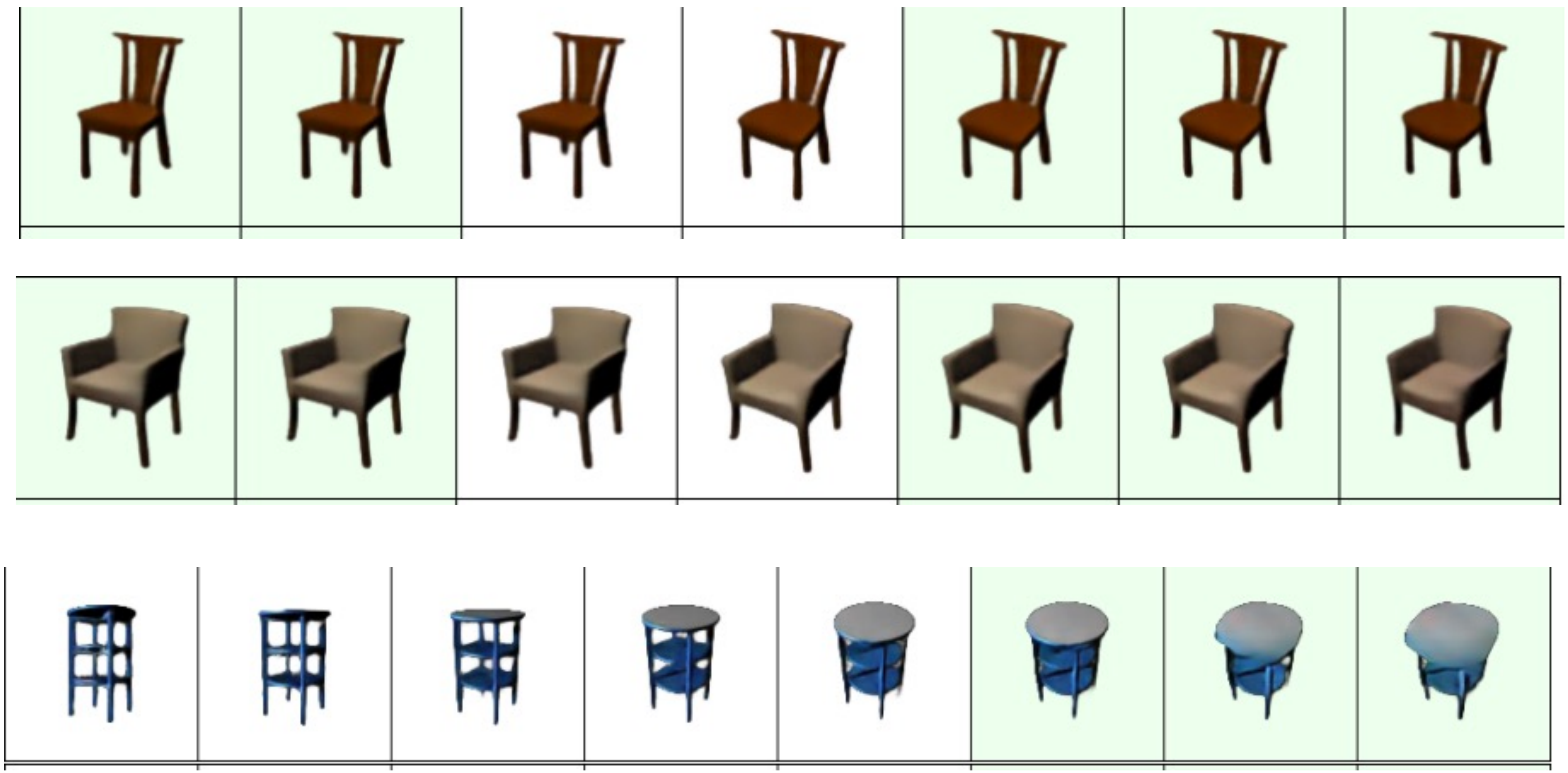
Fractiaionally-strided conv

Generating chairs conditional on chair ID, viewpoint, and transformation parameters



Dosovitskiy et al. Learning to Generate Chairs, Tables and Cars with Convolutional Networks
PAMI 2017 (CVPR 2015)

With Varying Viewpoints



Dosovitskiy et al. Learning to Generate Chairs, Tables and Cars with Convolutional Networks
PAMI 2017 (CVPR 2015)

With Varying Transformation Parameters

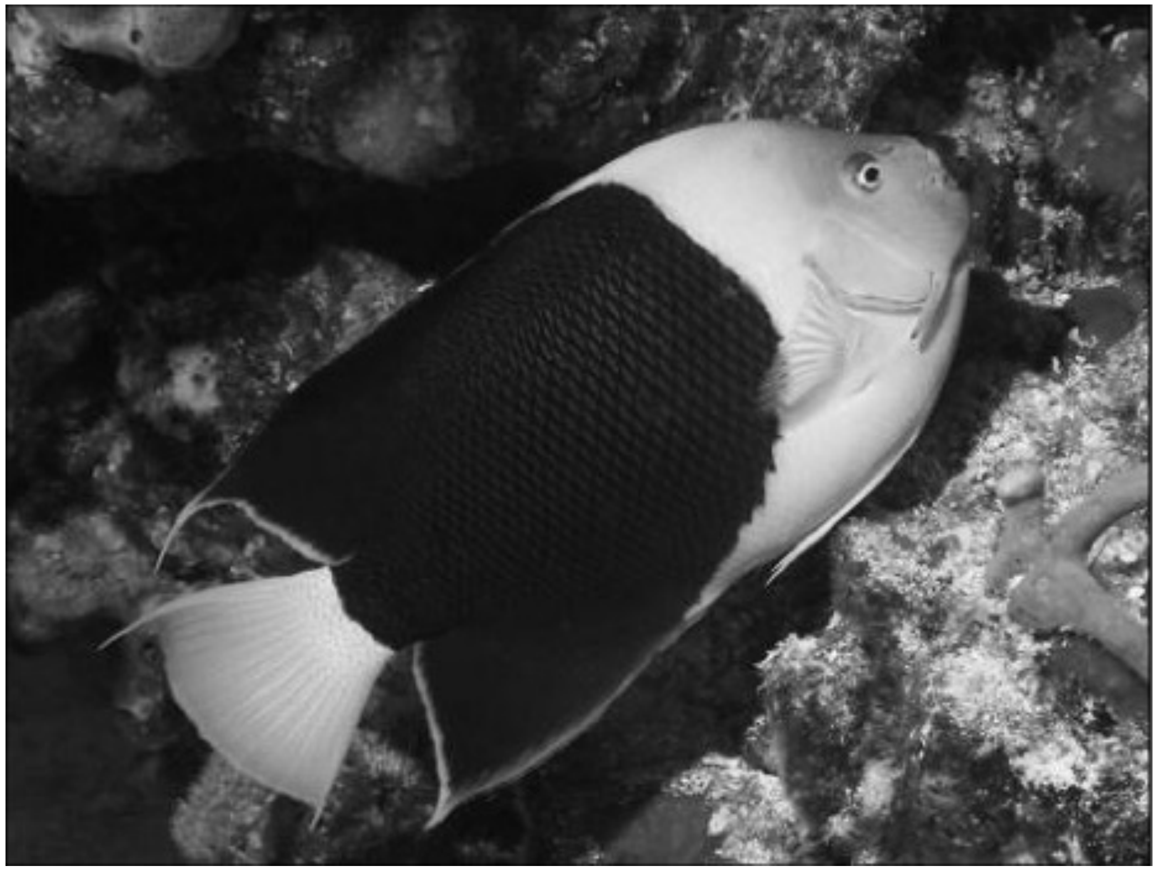
Dosovitskiy et al. Learning to Generate Chairs, Tables and Cars with Convolutional Networks
PAMI 2017 (CVPR 2015)

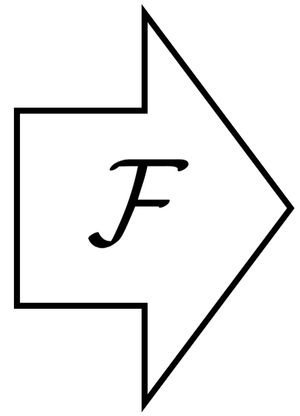
Interpolation between Two Chairs

Dosovitskiy et al. Learning to Generate Chairs, Tables and Cars with Convolutional Networks PAMI 2017 (CVPR 2015)

Better Loss Functions

Ansel Adams. ***Yosemite Valley Bridge.***



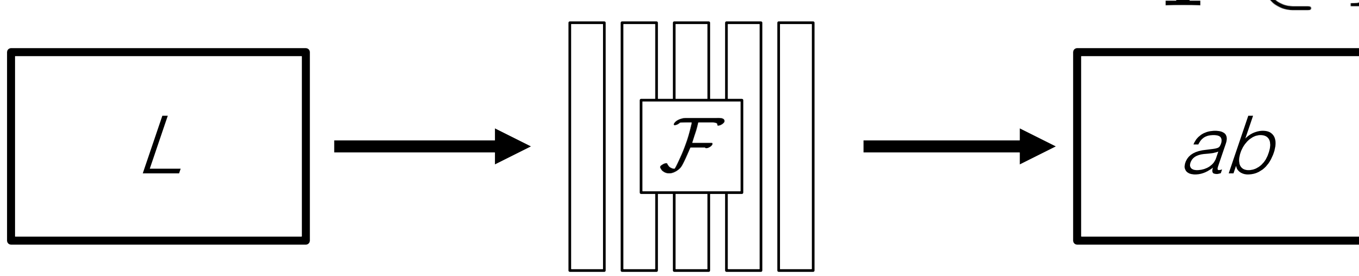


Grayscale image: *L* channel

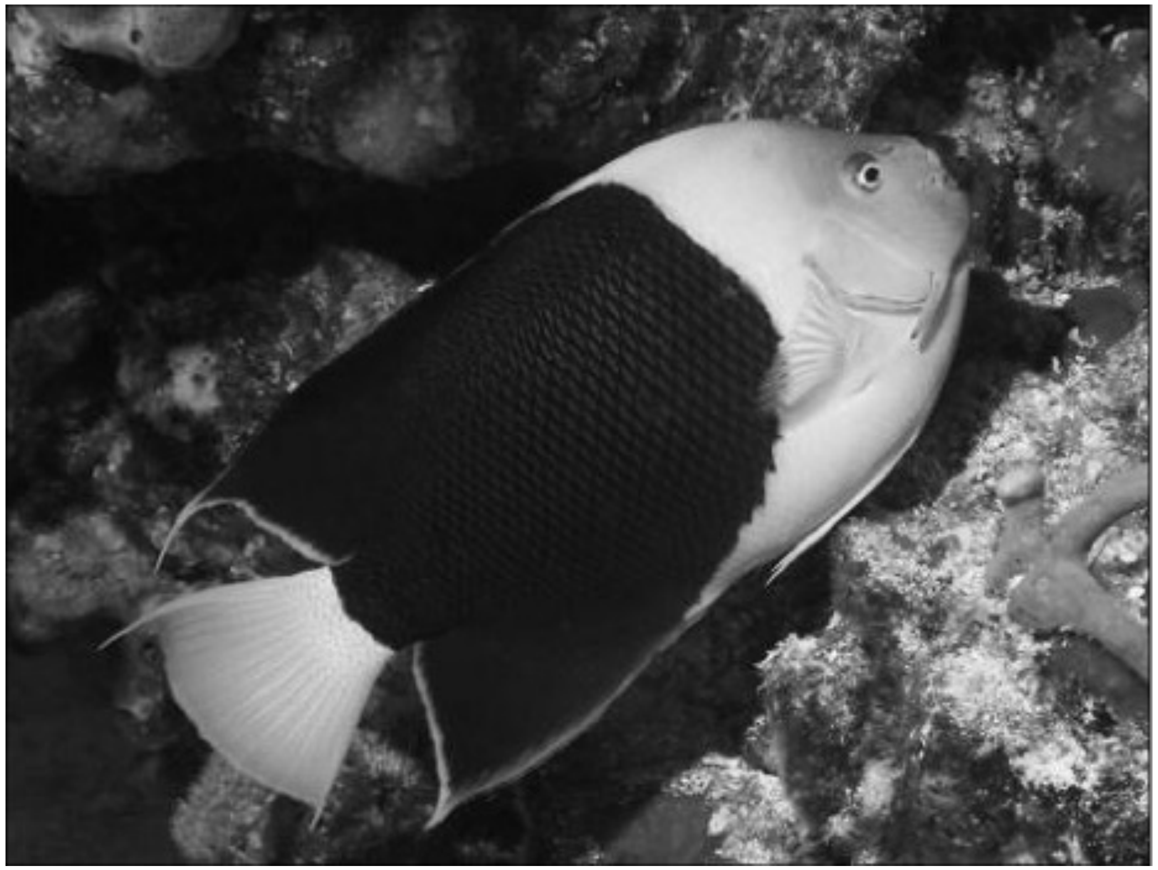
$$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$$

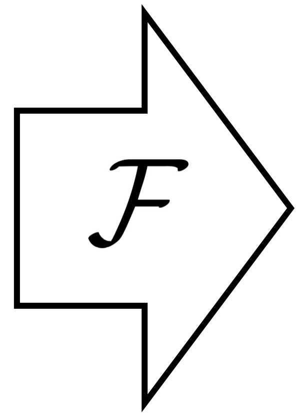
Color information: *ab* channels

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



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



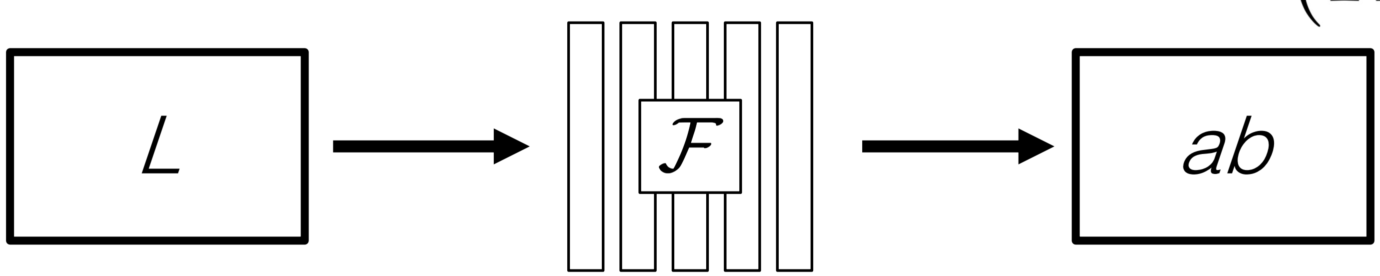


Grayscale image: *L* channel

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

Concatenate (*L*,*ab*) channels

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



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

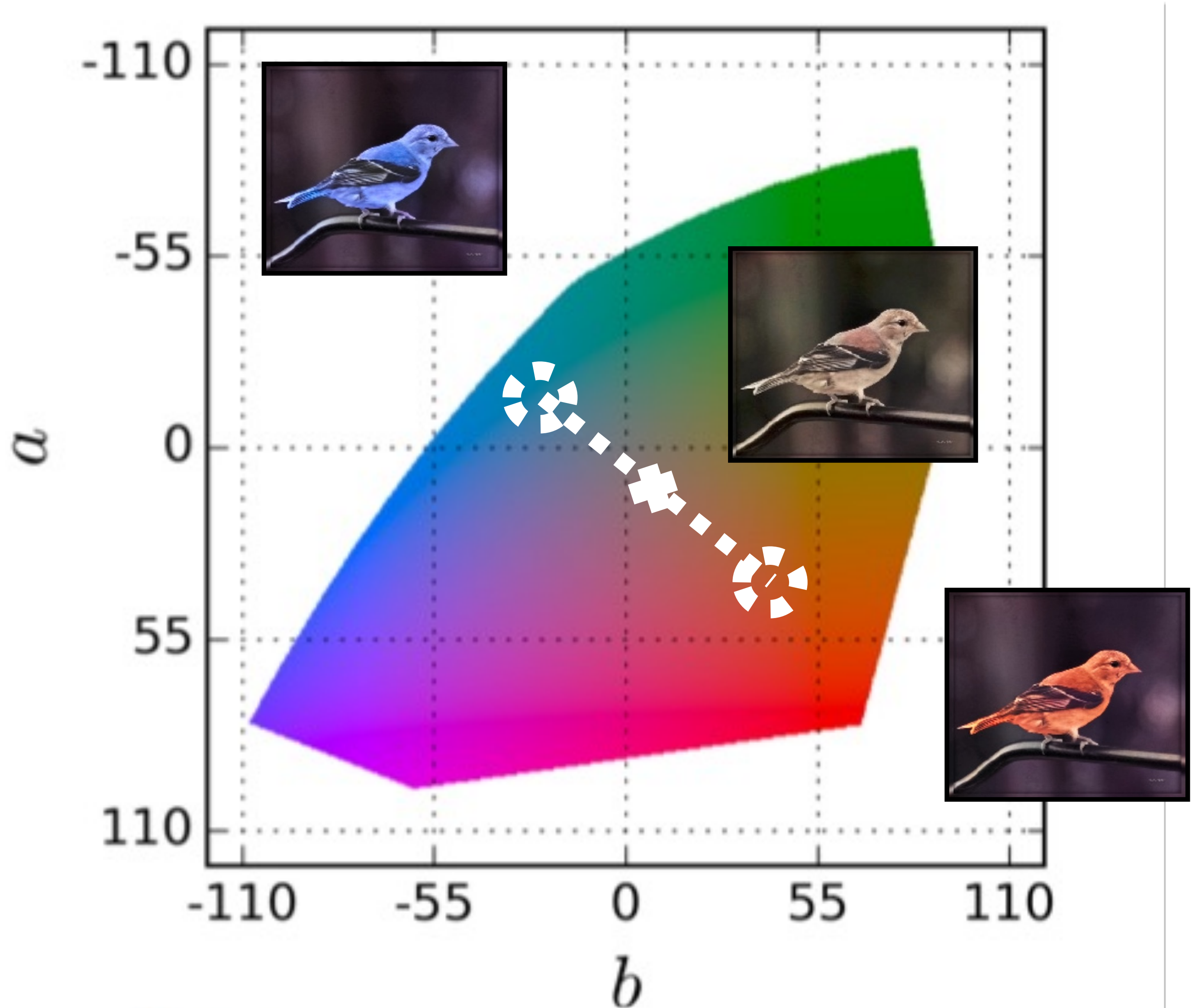
Simple L2 regression doesn't work ☹

Input

Output

Ground truth

$$L_2(\widehat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \| \mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w} \|_2^2$$



$$L_2(\widehat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w}\|_2^2$$

Better Loss Function

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

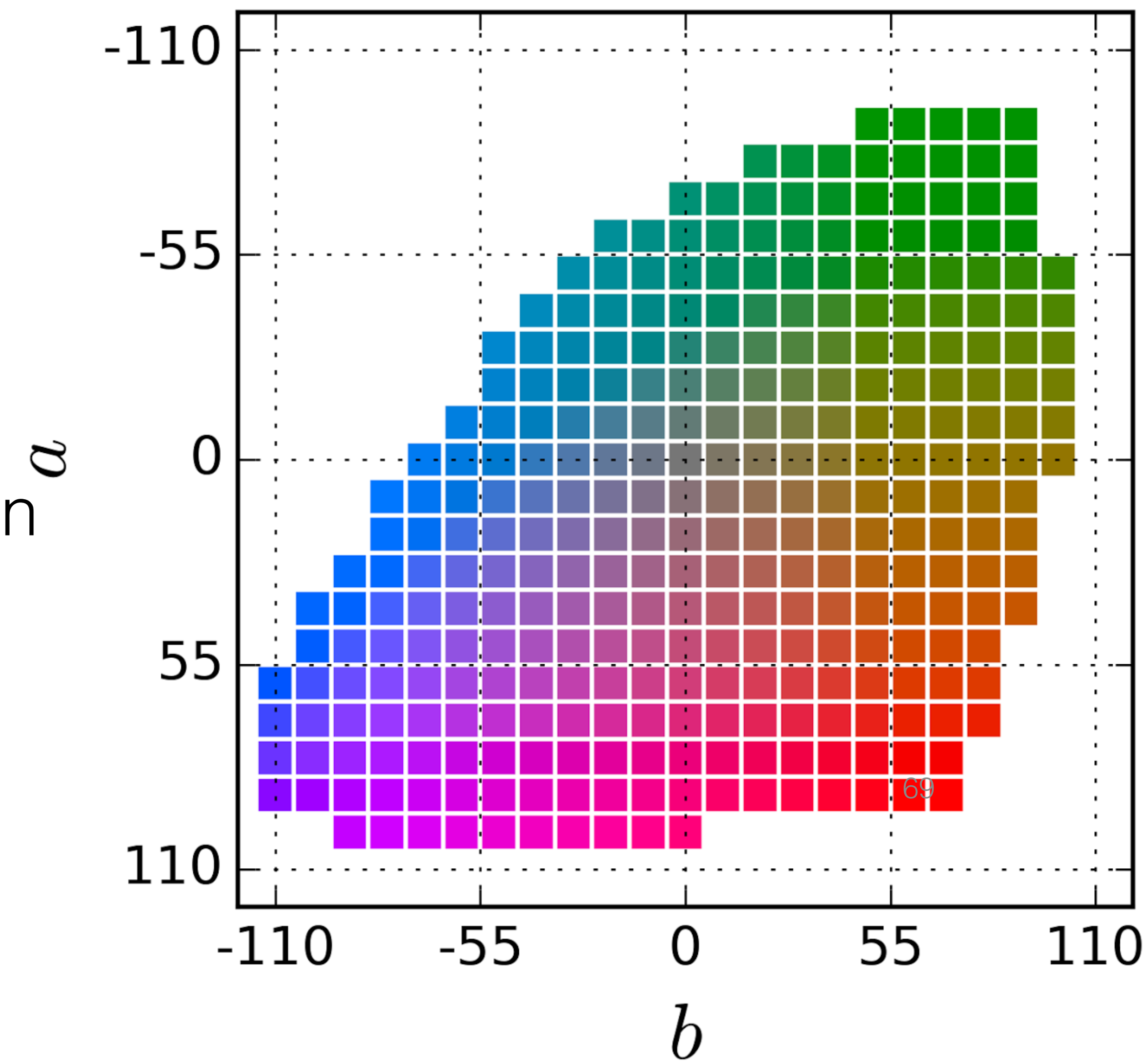
- Regression with L2 loss inadequate

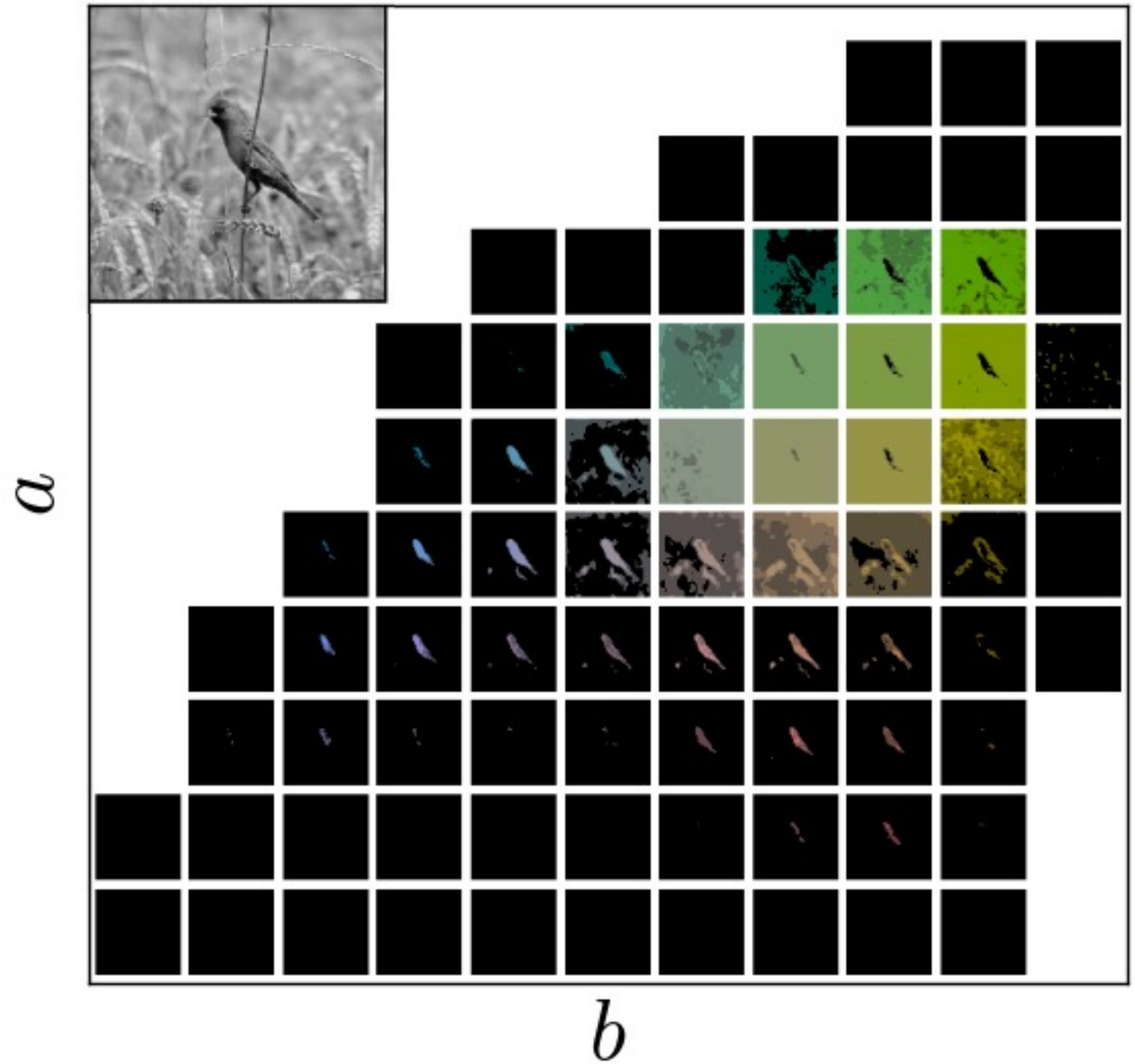
$$\mathrm{L}_2(\widehat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w}\|_2^2$$

- 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)





a
b

Designing loss functions

Color distribution cross-entropy loss with colorfulness enhancing term.

[Zhang, Isola, Efros, ECCV 2016]

Thank You!

16-726, Spring 2023

https://learning-image-synthesis.github.io/