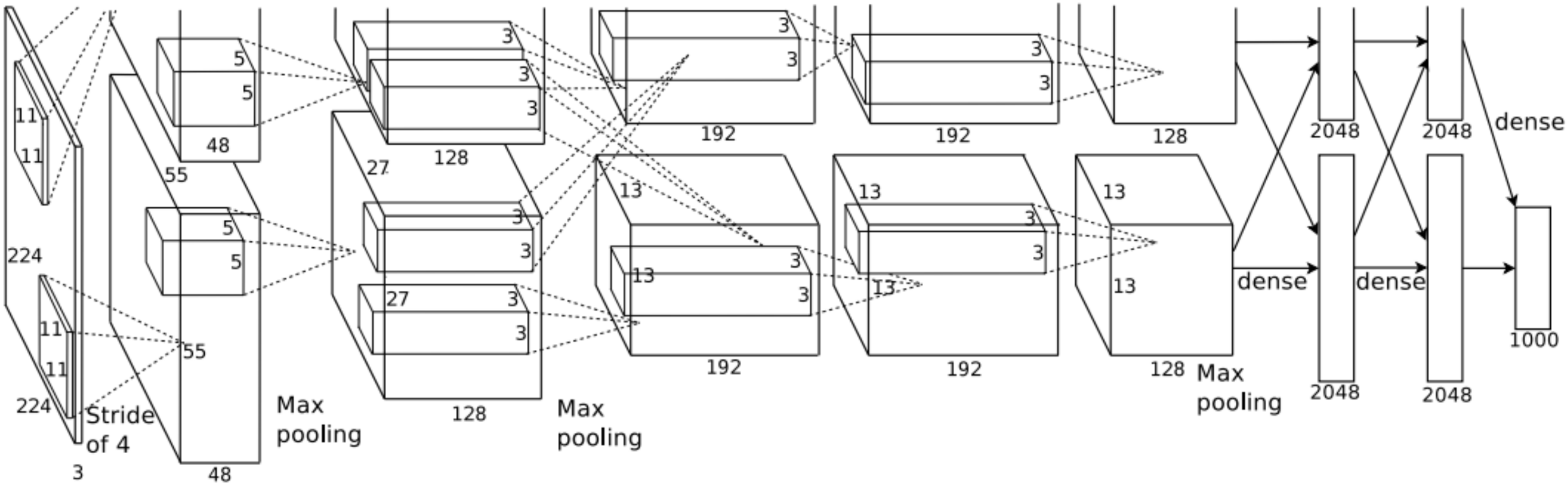
Deep Learning in Comp. Photo

With slides from Phil Isola, James Hays, and Andrea Vedaldi



CS194: Image Manipulation & Computational Photography

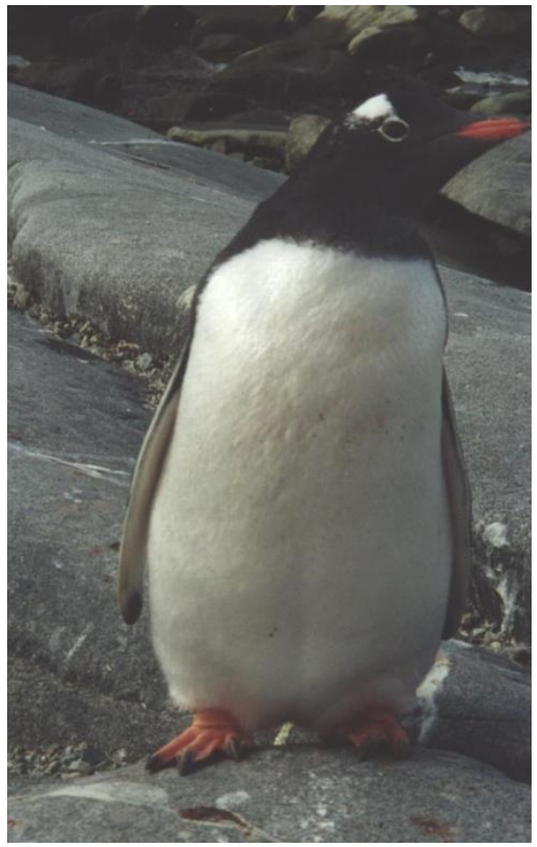
Alexei Efros, UC Berkeley, Fall 2018

Visual similarity via labels

“Penguin”

?

==



“Penguin”

Machine Learning as data association

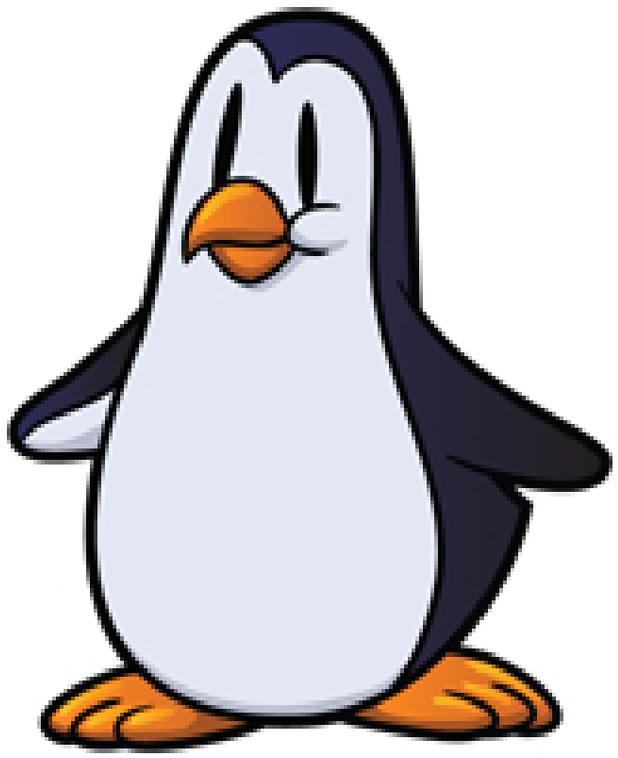
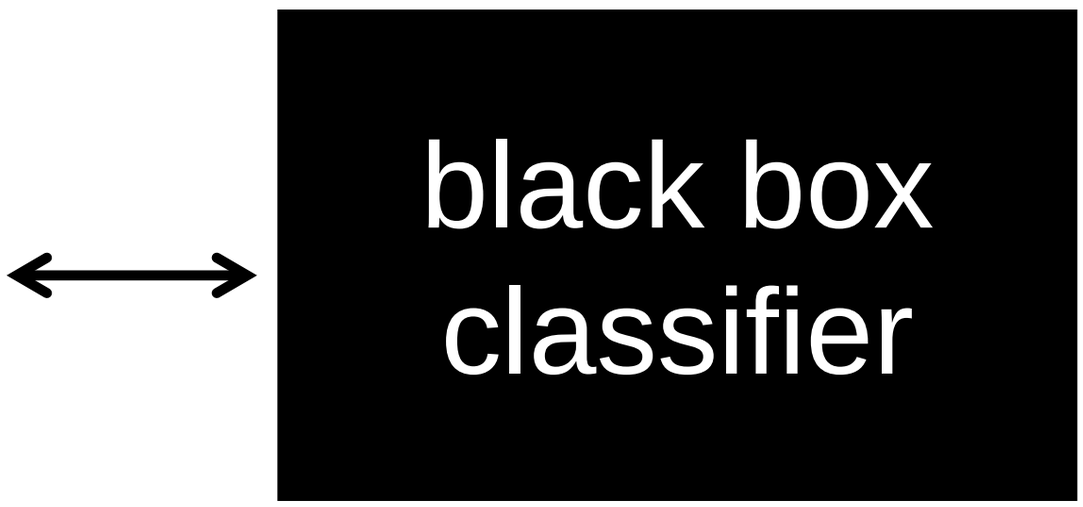


image *X*



label *Y*

At test time…

black box classifier

?

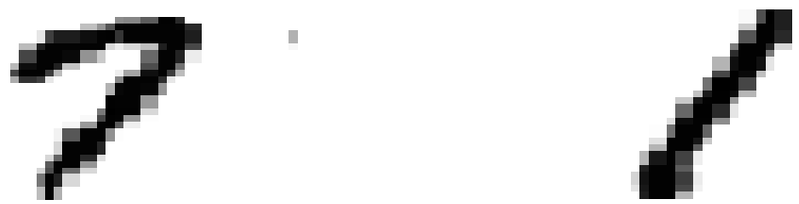
image *X*

Quick Background on Supervised Learning

Jitendra Malik

Fig. 4. Size-normalized examples from the MNIST database.

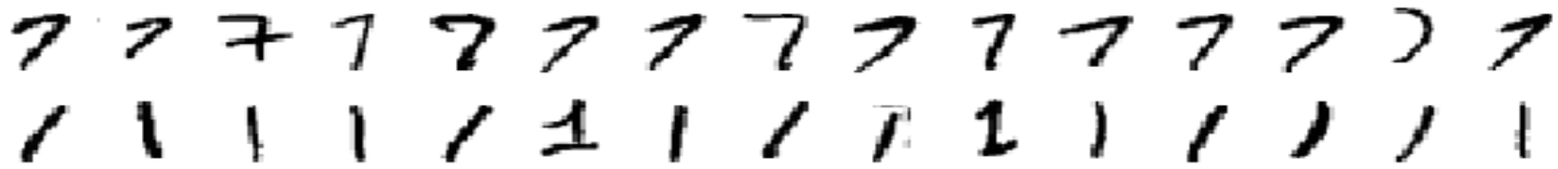
Warm-up Example: Binary Digit Classification



vs.

Learning Approach to Object 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…

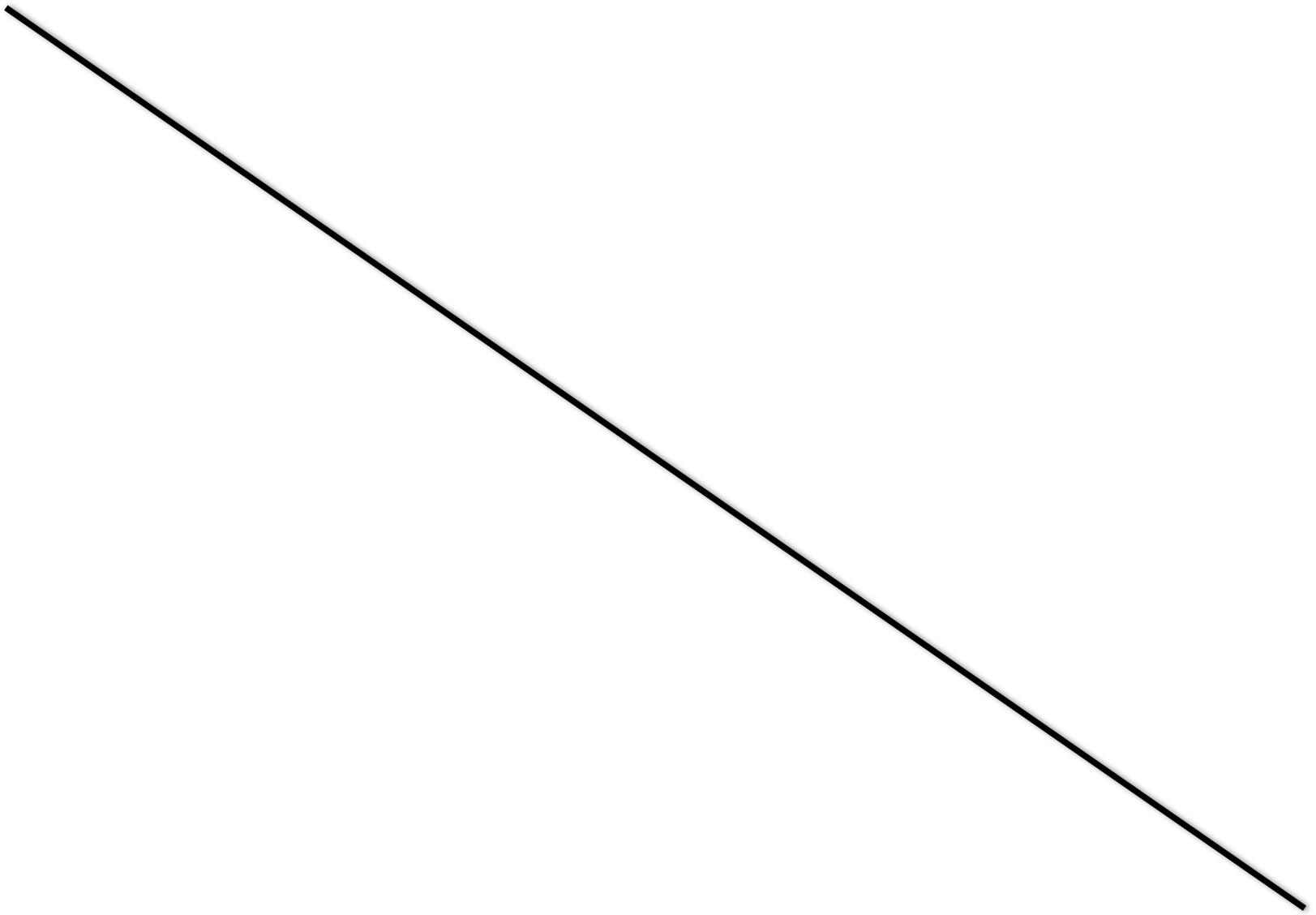
Let us take an example…

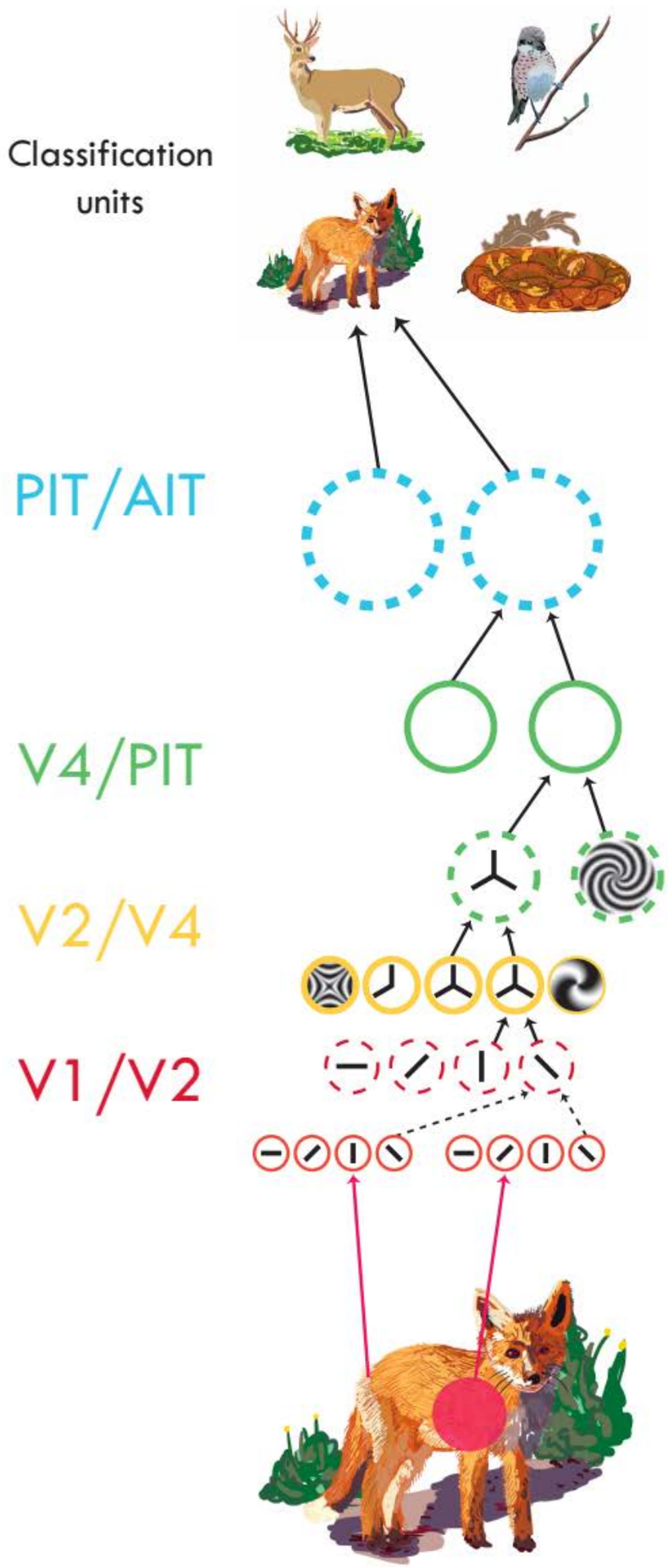
In feature space, positive and negative examples are just points…

How do we classify a new point?

Nearest neighbor rule “transfer label of nearest example”

Linear classifier rule





Serre, 2014

Basic idea

Brain/Machine

“clown fish”

Object recognition

Edges

Texture

Colors

Segments

Parts

“clown fish”

Feature extractors

Classifier

Object recognition

Learned

Edges

Texture

Colors

Segments

Parts

“clown fish”

Feature extractors

Classifier

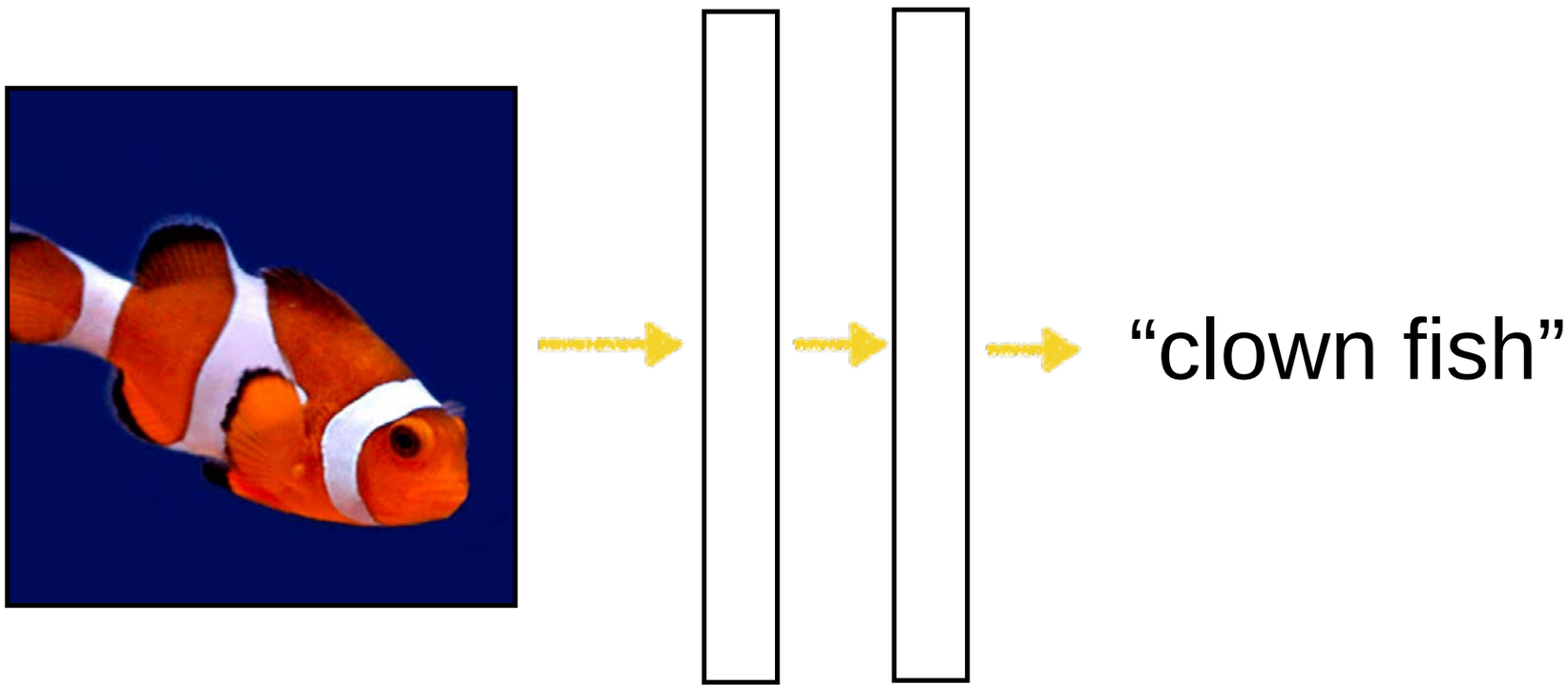
Neural network

Learned

“clown fish”

Neural network

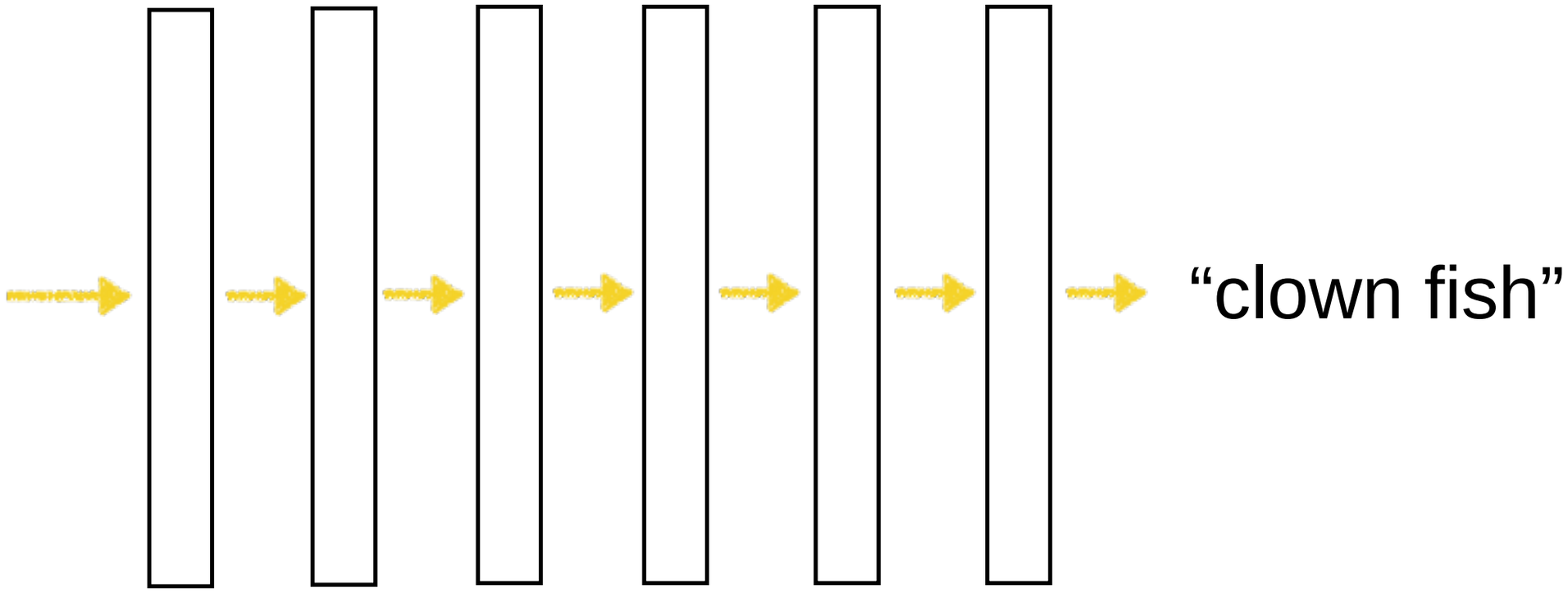
Learned



“clown fish”

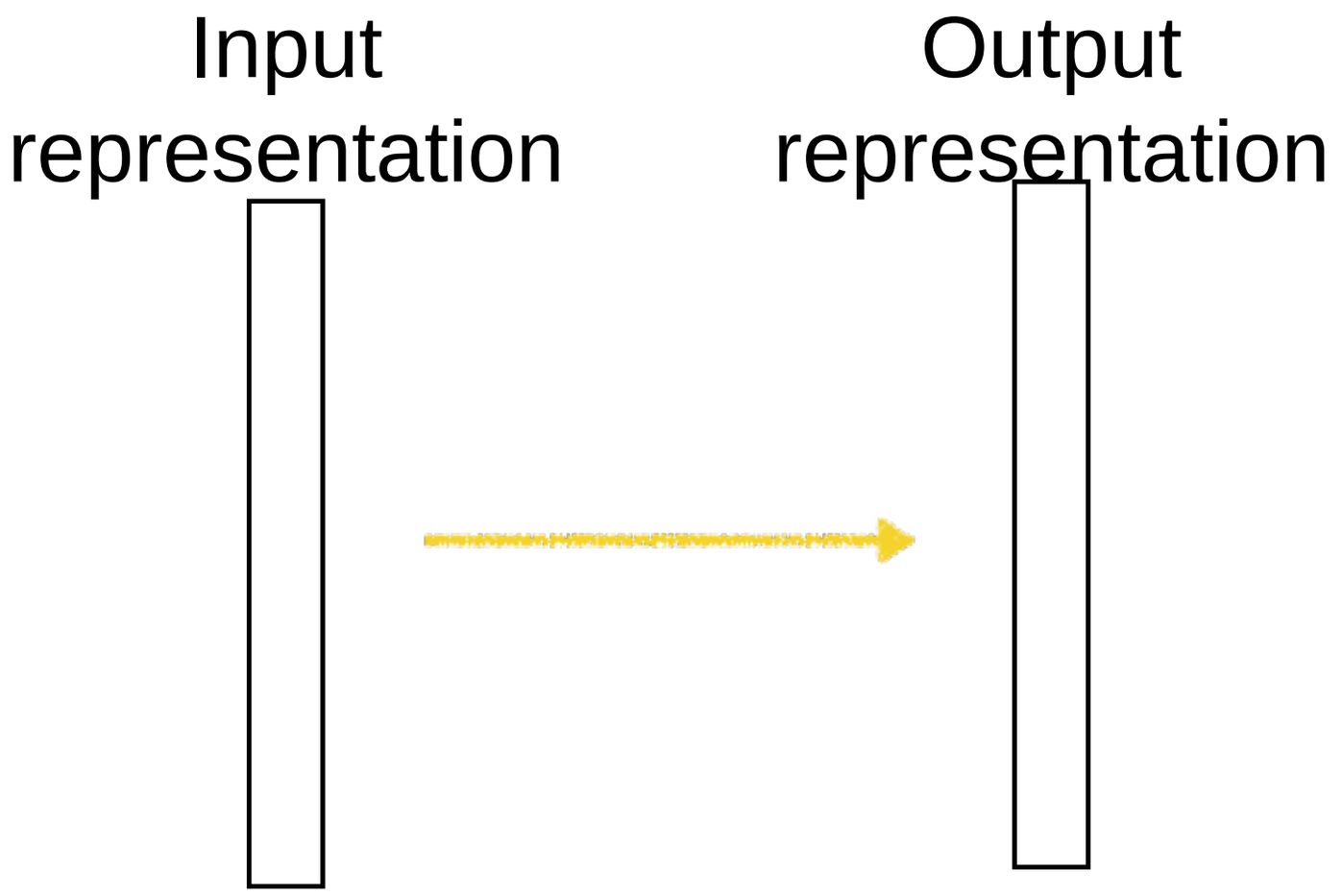
Deep neural network

Learned

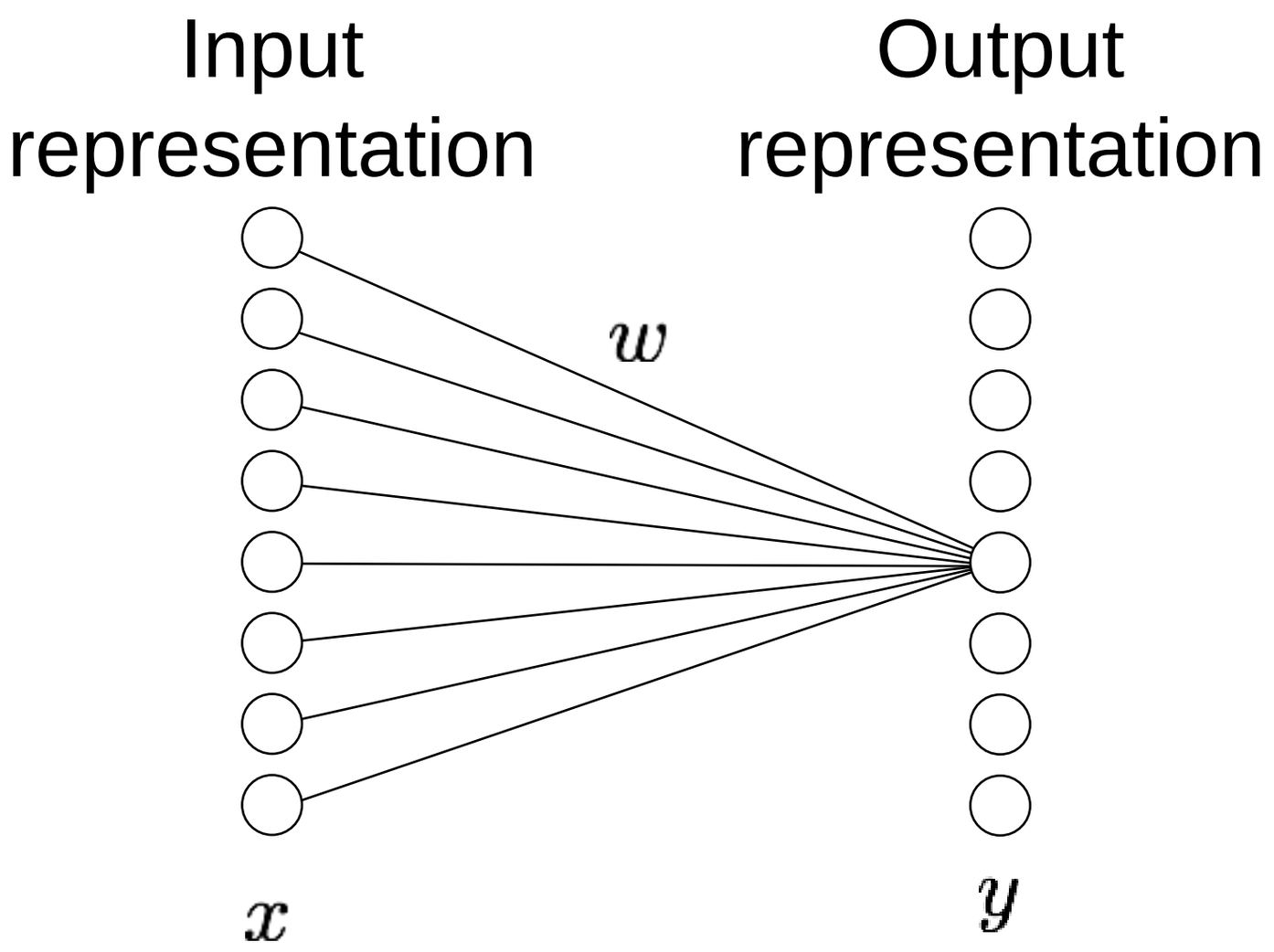


“clown fish”

Computation in a neural net

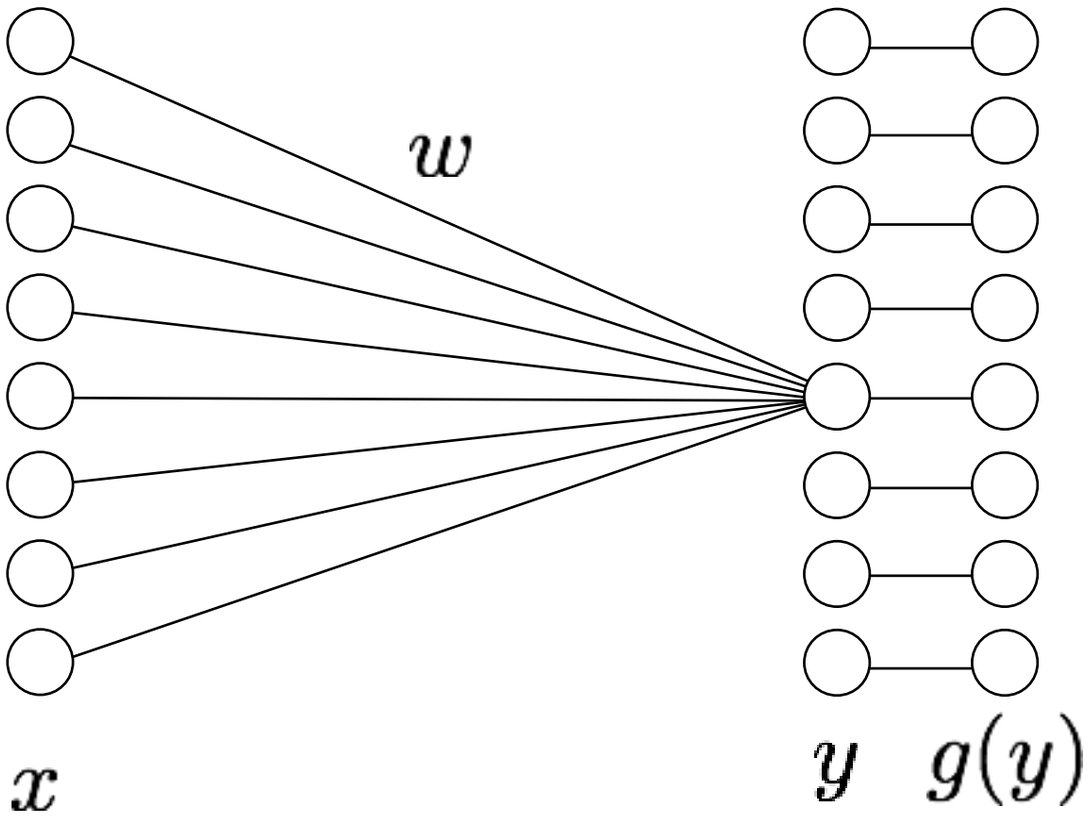


Computation in a neural net

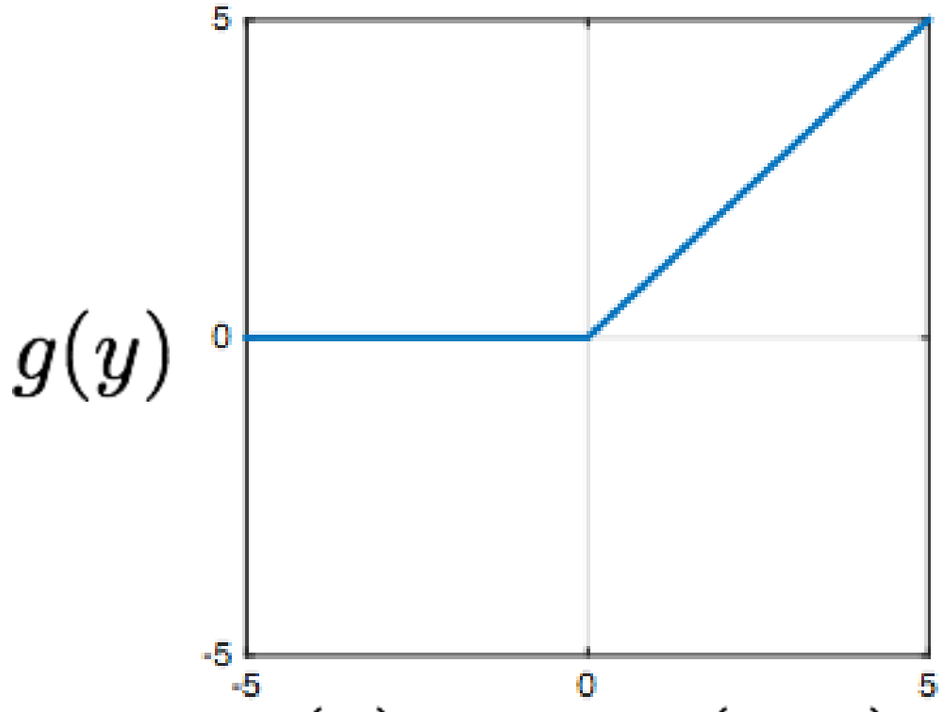


$$y_j = \sum_i w_{ij} x_i$$

Computation in a neural net

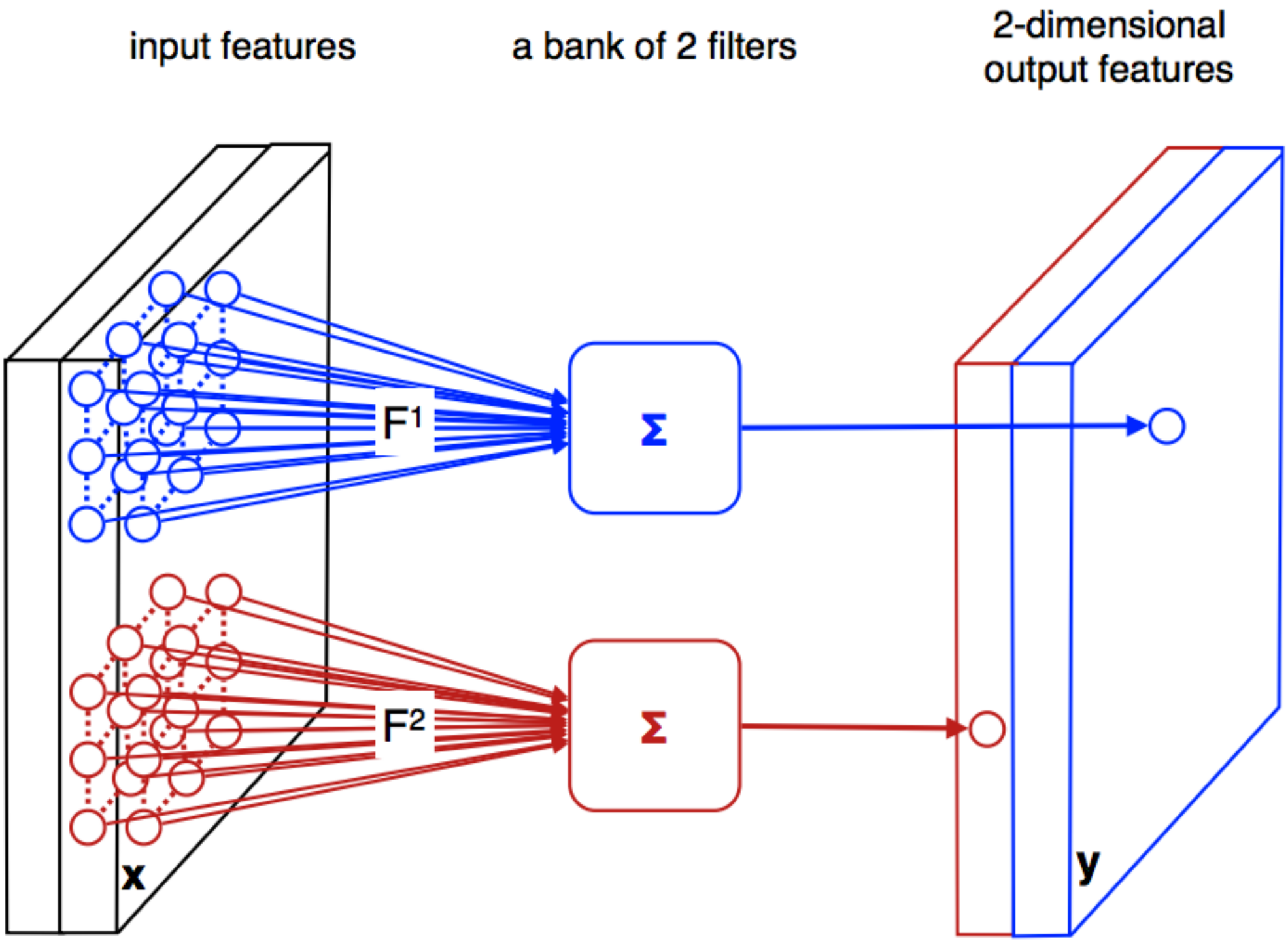


Rectified linear unit (ReLU)



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

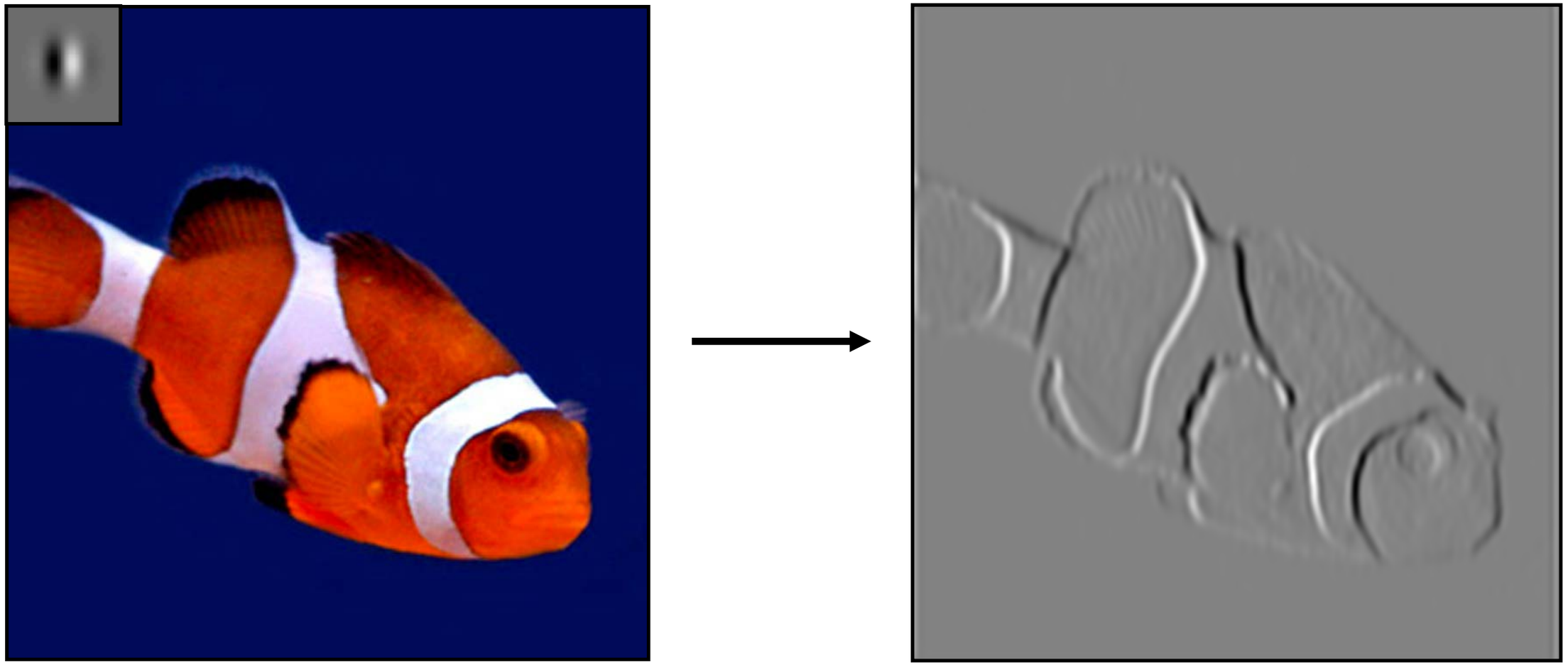
Convolutional Neural Nets



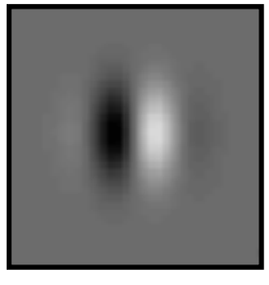
Slide from Andrea Vedaldi

Convolutional Neural Nets

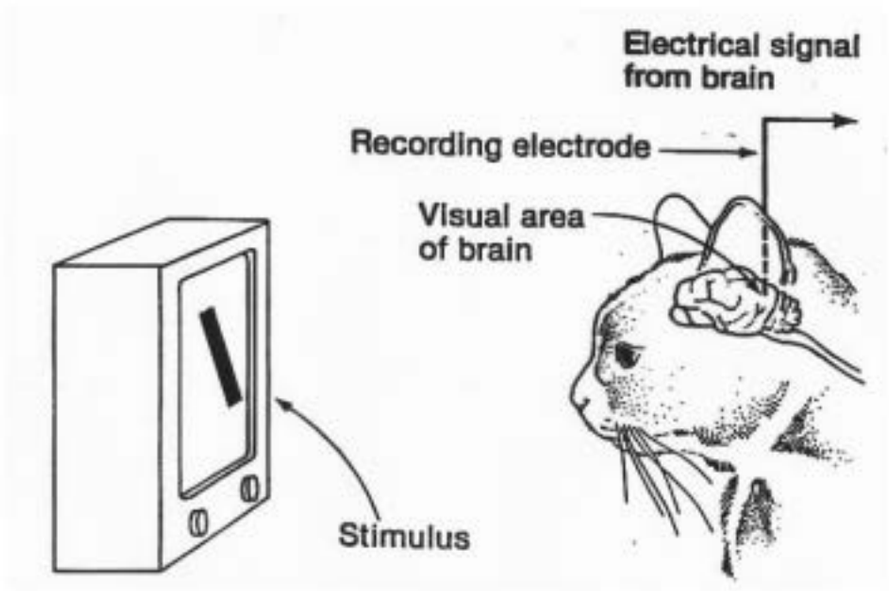
Convolution

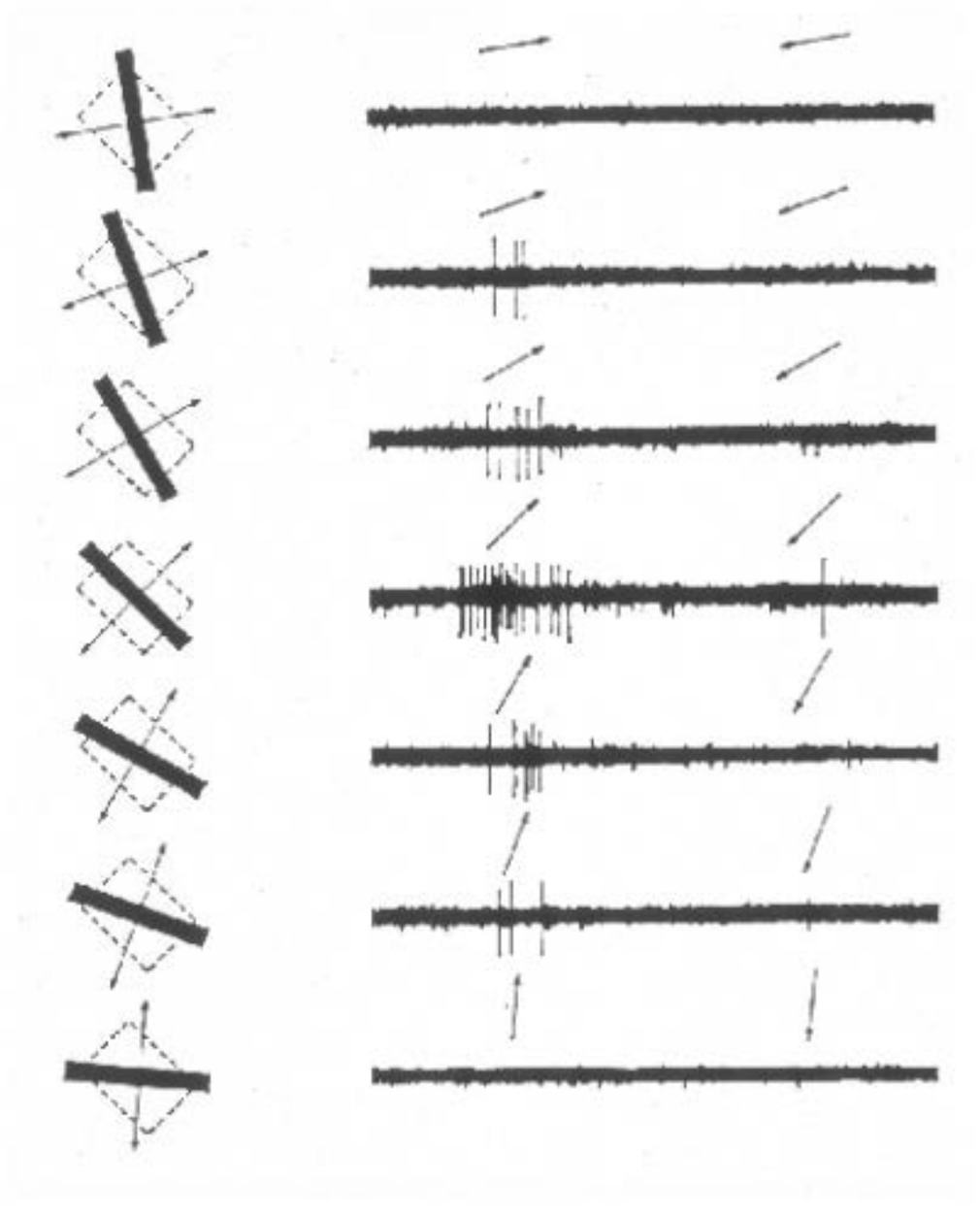


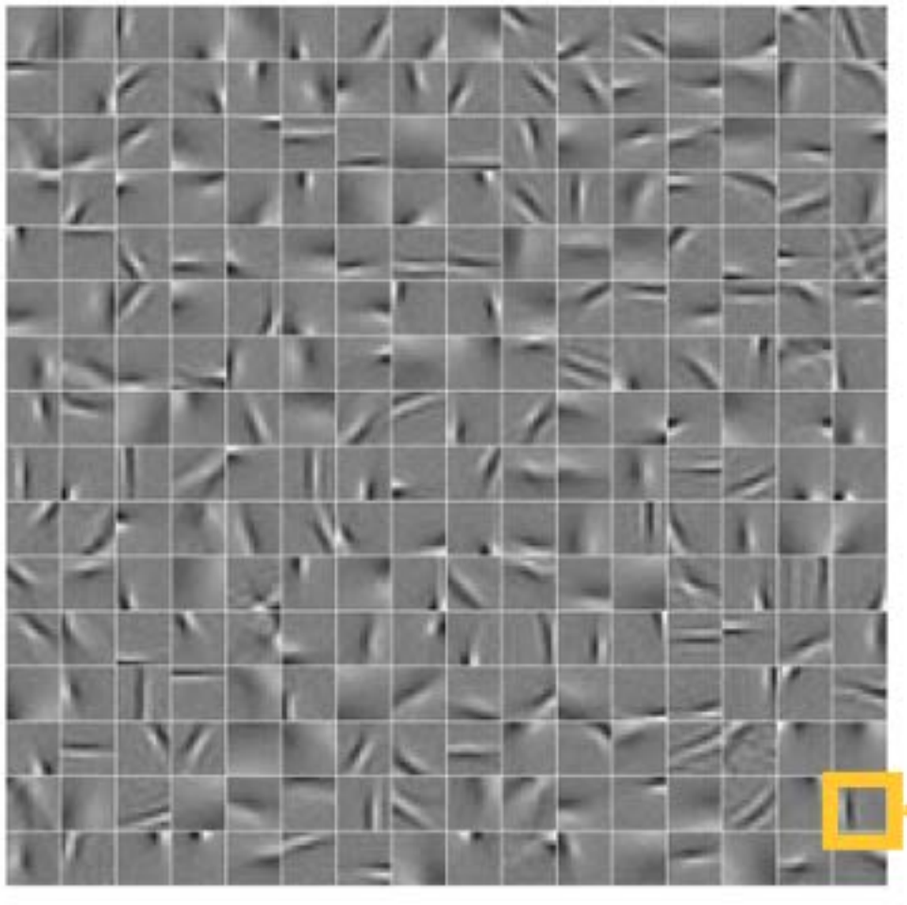
filter

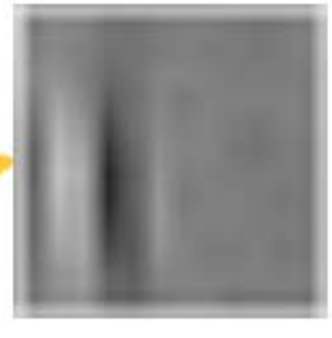


[Hubel and Wiesel 59]





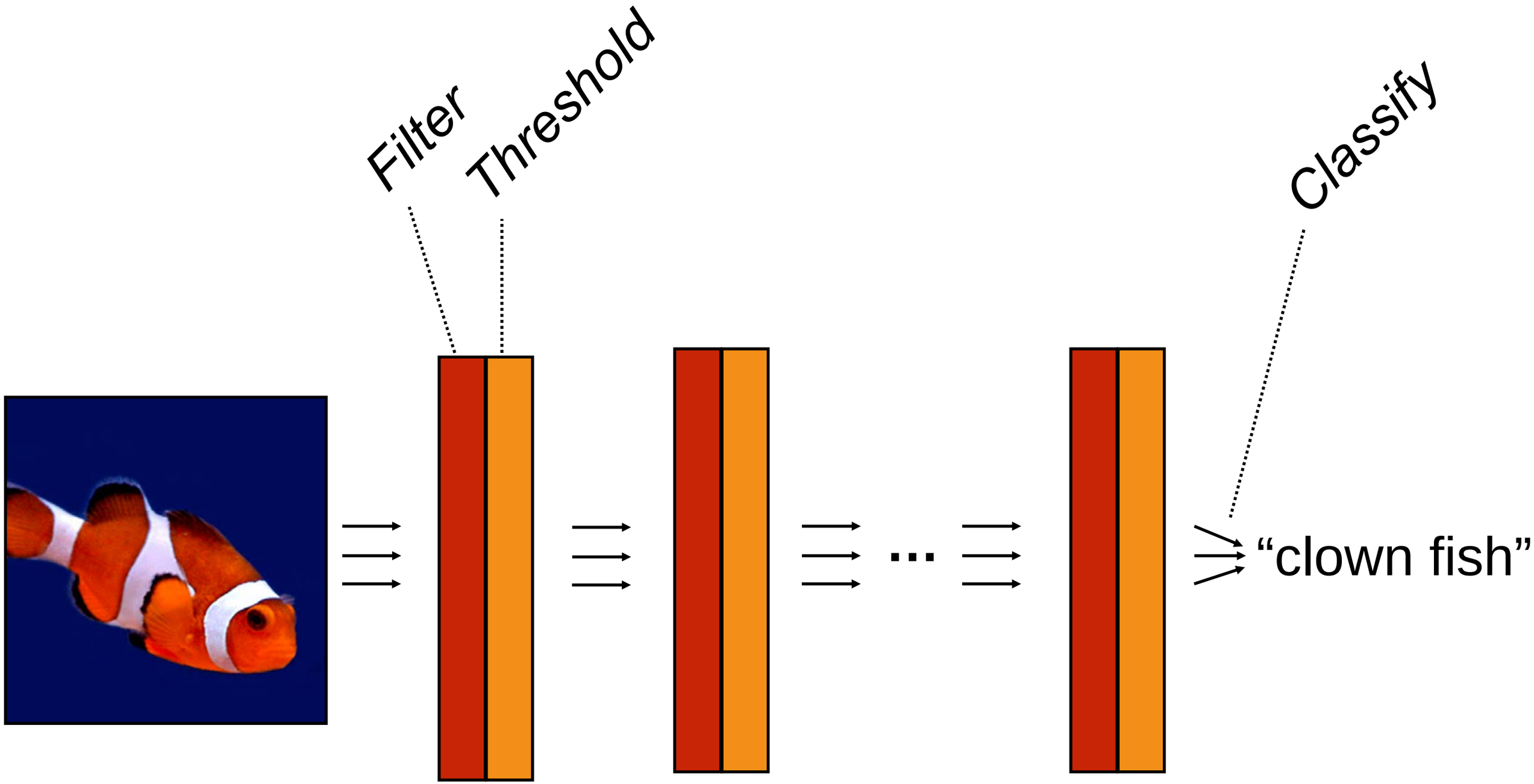




oriented filter

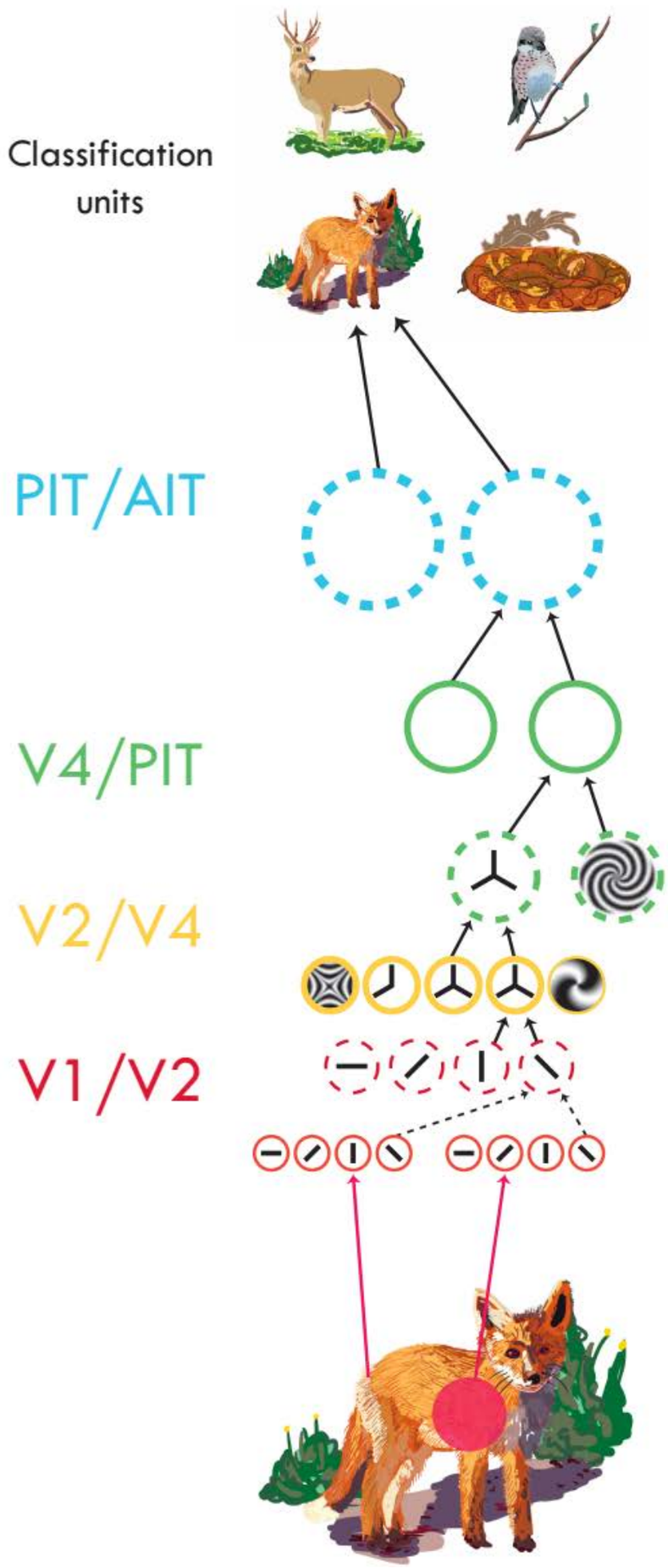
Slide from Andrea Vedaldi

Computation in a neural net



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

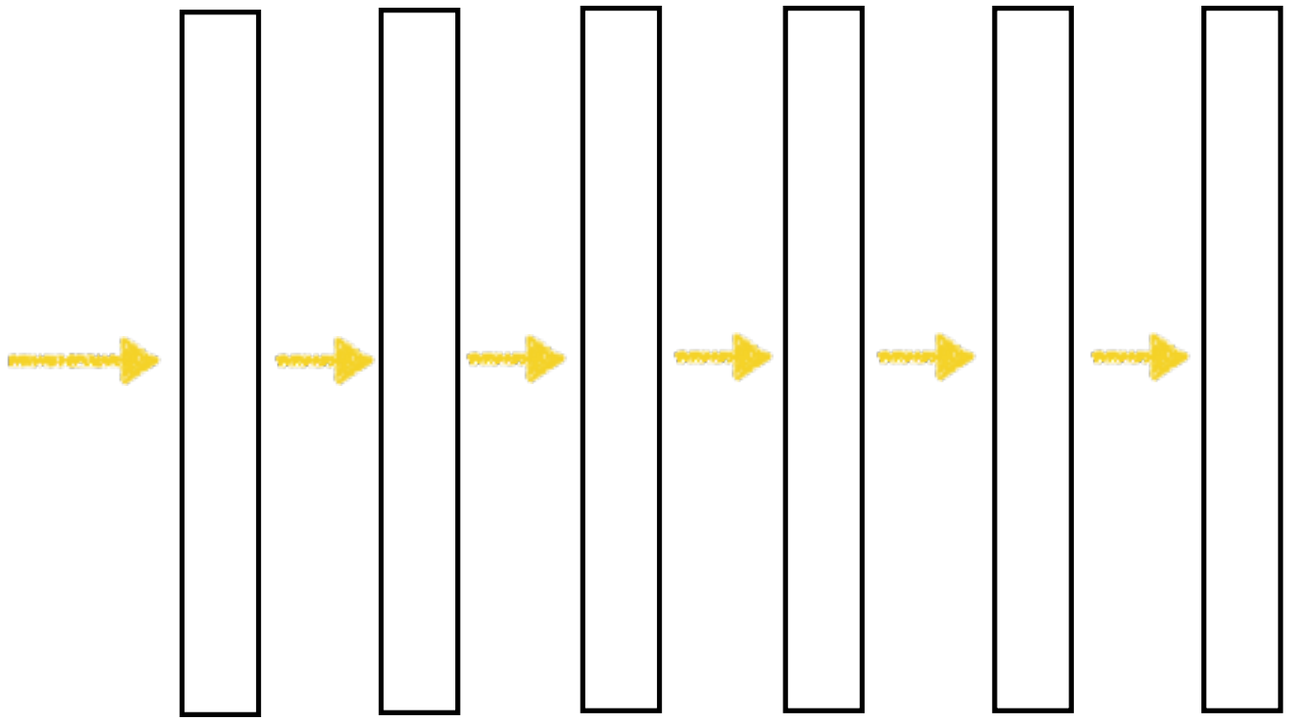
Computation in a neural net



Serre, 2014

Learning with deep nets

Learned



→ “clown fish”

Learning with deep nets

→ “clown fish”

→ “grizzly bear”

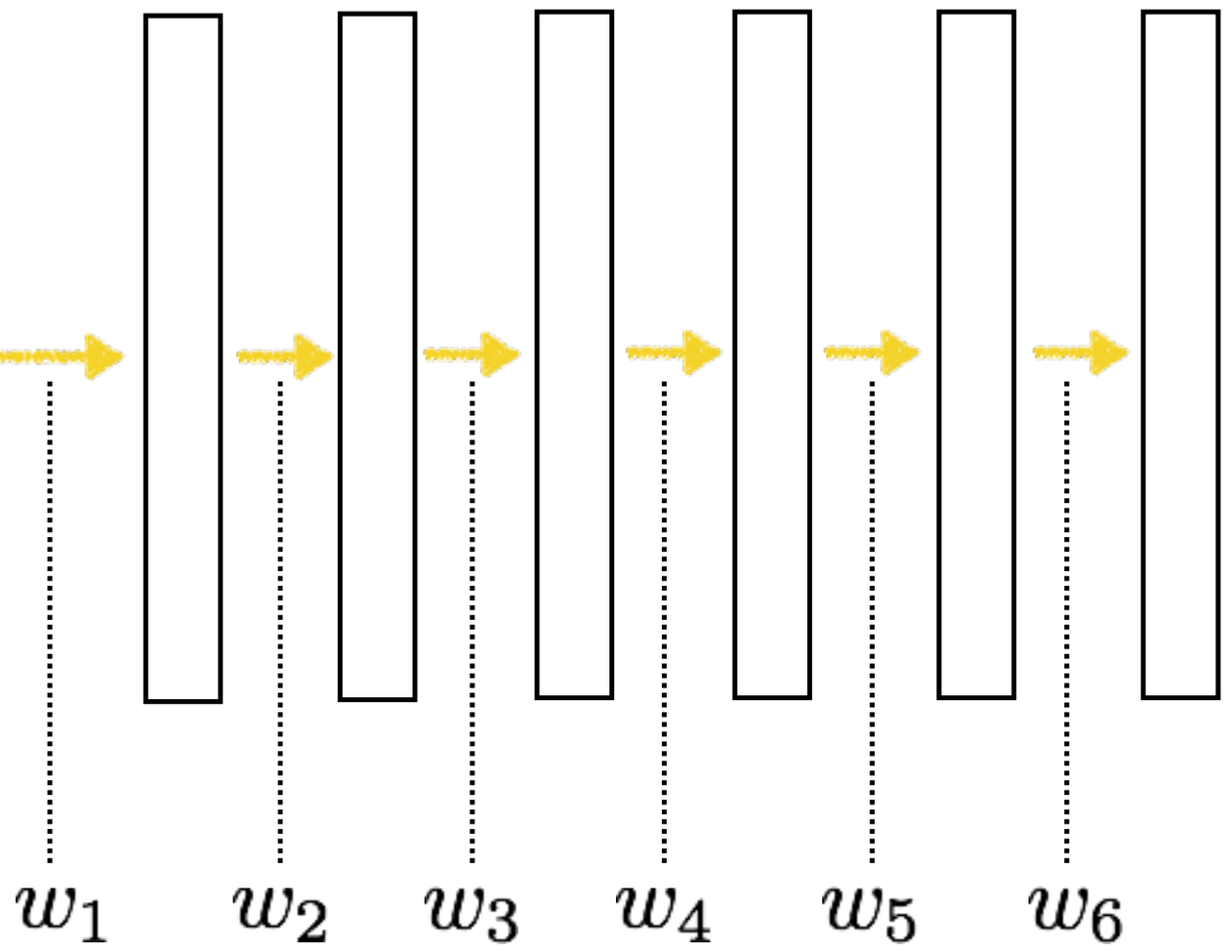
→ “chameleon”

Train network to associate the right label with each image

Learning with deep nets

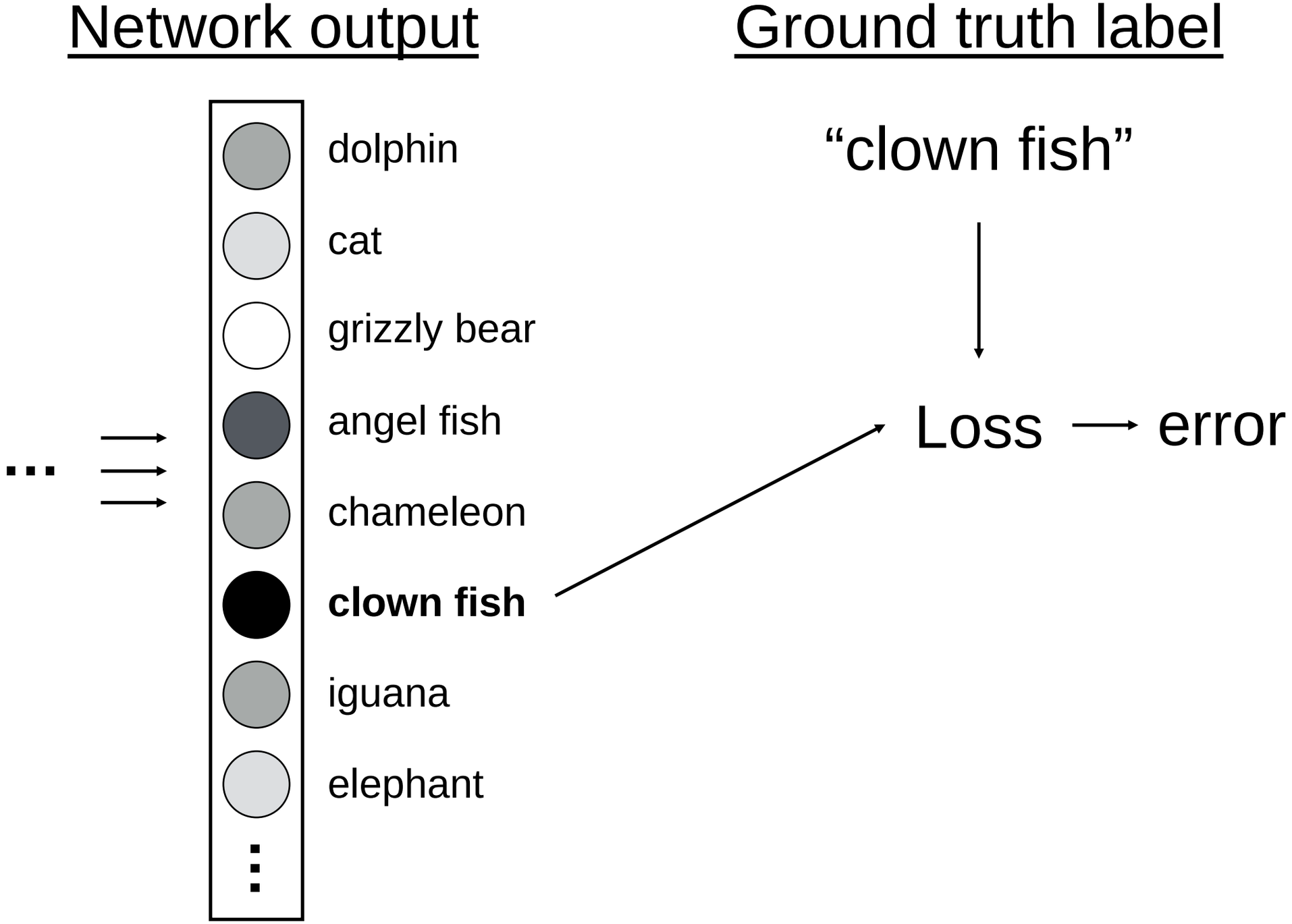
Learned

"clown fish"

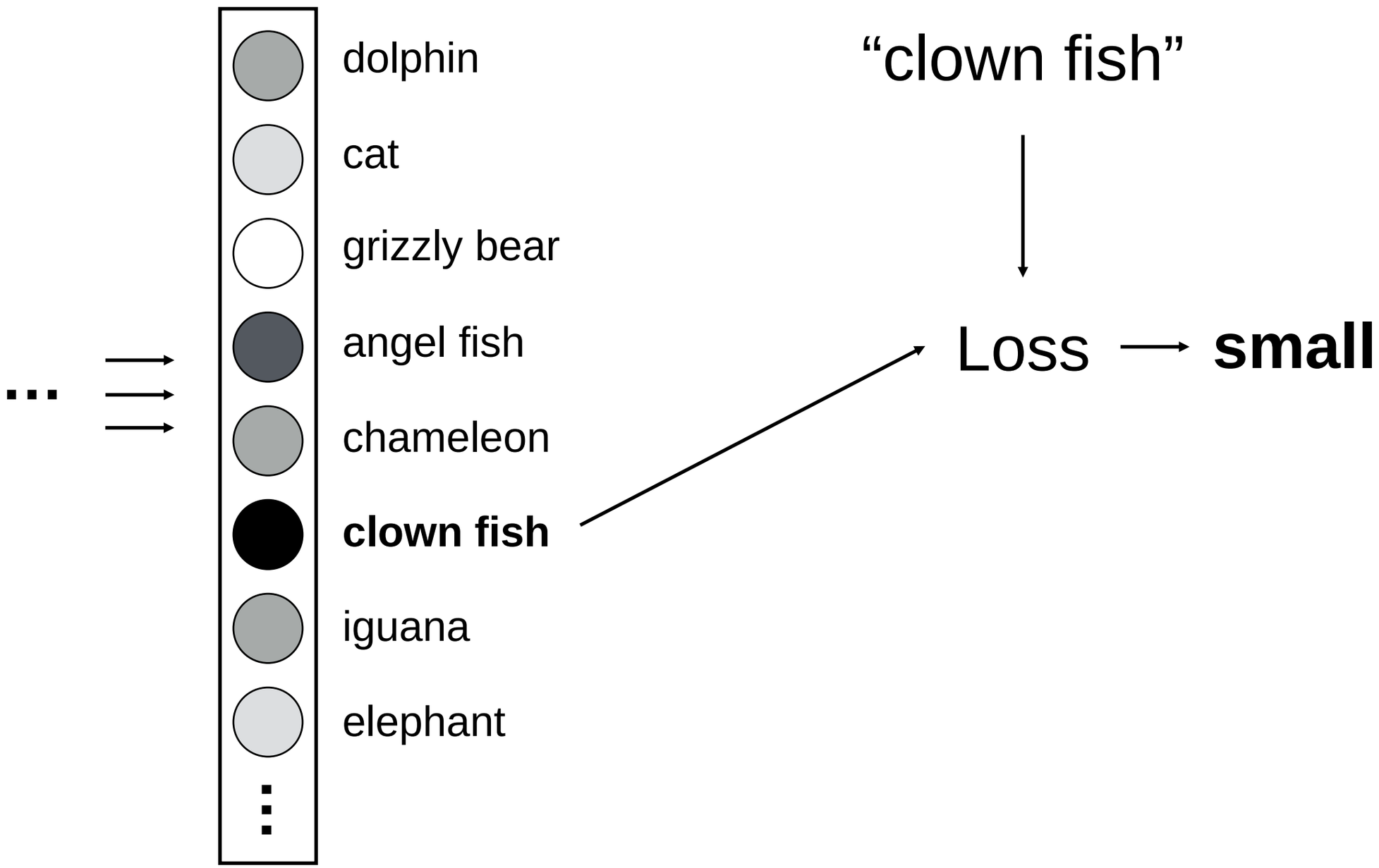


$$\underset{\mathbf{w}}{\text{argmin}} \quad L(w_1, \ldots, w_6)$$

Loss function



Loss function



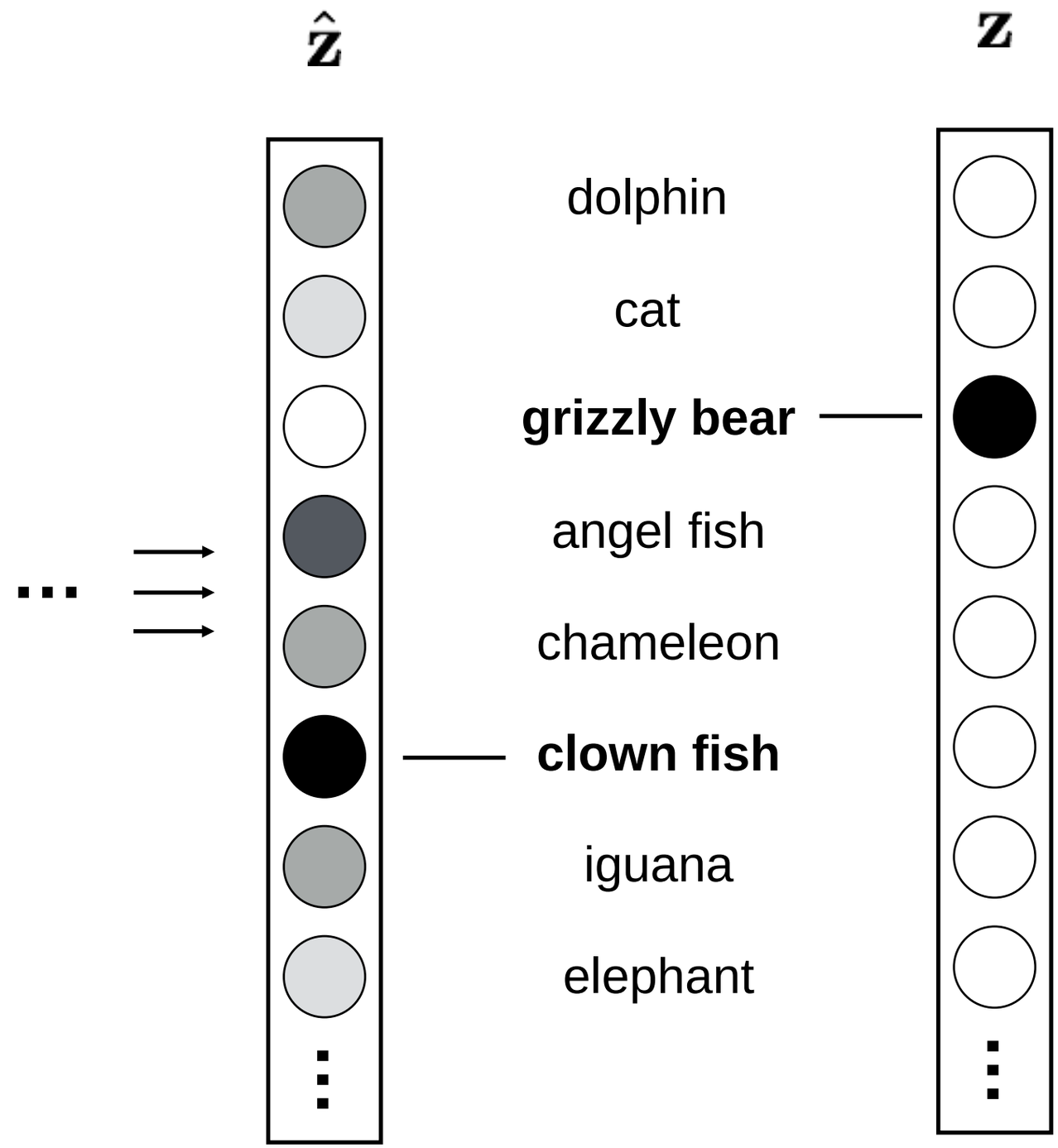
Loss function

Network output

Ground truth label

Loss function for classification

Network output Ground truth label



Probability of the observed data under the model

$$H(\hat{\mathbf{z}}, \mathbf{z}) = -\sum_c \hat{\mathbf{z}}_c \log \mathbf{z}_c$$

Results in learning a probability model $p(c|\mathbf{x})$*!*

Learning with deep nets

Learned

$\mathbf{z}_1$

"clown fish"

$\mathbf{x}_1$

Loss

$\ell(\mathbf{z}_1, f(\mathbf{x}_1; \mathbf{w}))$

w_1 w_2 w_3 w_4 w_5 w_6

Learning with deep nets

Learned

$\mathbf{z}_2$

"grizzly bear"

$\mathbf{x}_2$

Loss

$\ell(\mathbf{z}_2, f(\mathbf{x}_2; \mathbf{w}))$

w_1 w_2 w_3 w_4 w_5 w_6

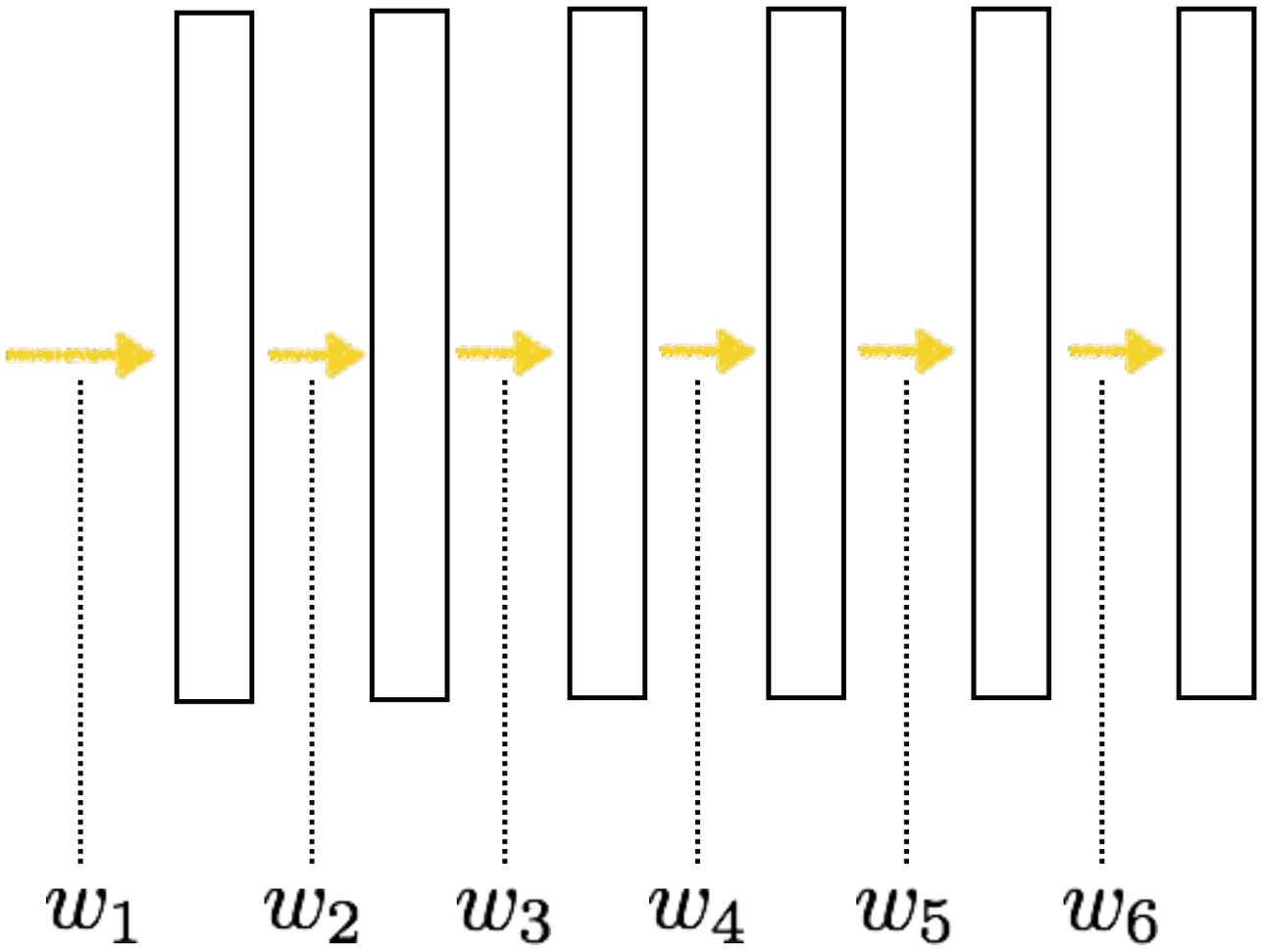
Learning with deep nets

Learned

$\mathbf{z}_i$

"chameleon"

$\mathbf{x}_i$



Loss

$\ell(\mathbf{z}_i, f(\mathbf{x}_i; \mathbf{w}))$

$$\underset{\mathbf{w}}{\operatorname{argmin}} \sum_i \ell(\mathbf{z}_i, f(\mathbf{x}_i; \mathbf{w}))$$

Gradient descent

$$\underset{\mathbf{w}}{\operatorname{argmin}} \quad \sum_i \ell(\mathbf{z}_i, f(\mathbf{x}_i; \mathbf{w})) = L(\mathbf{w})$$

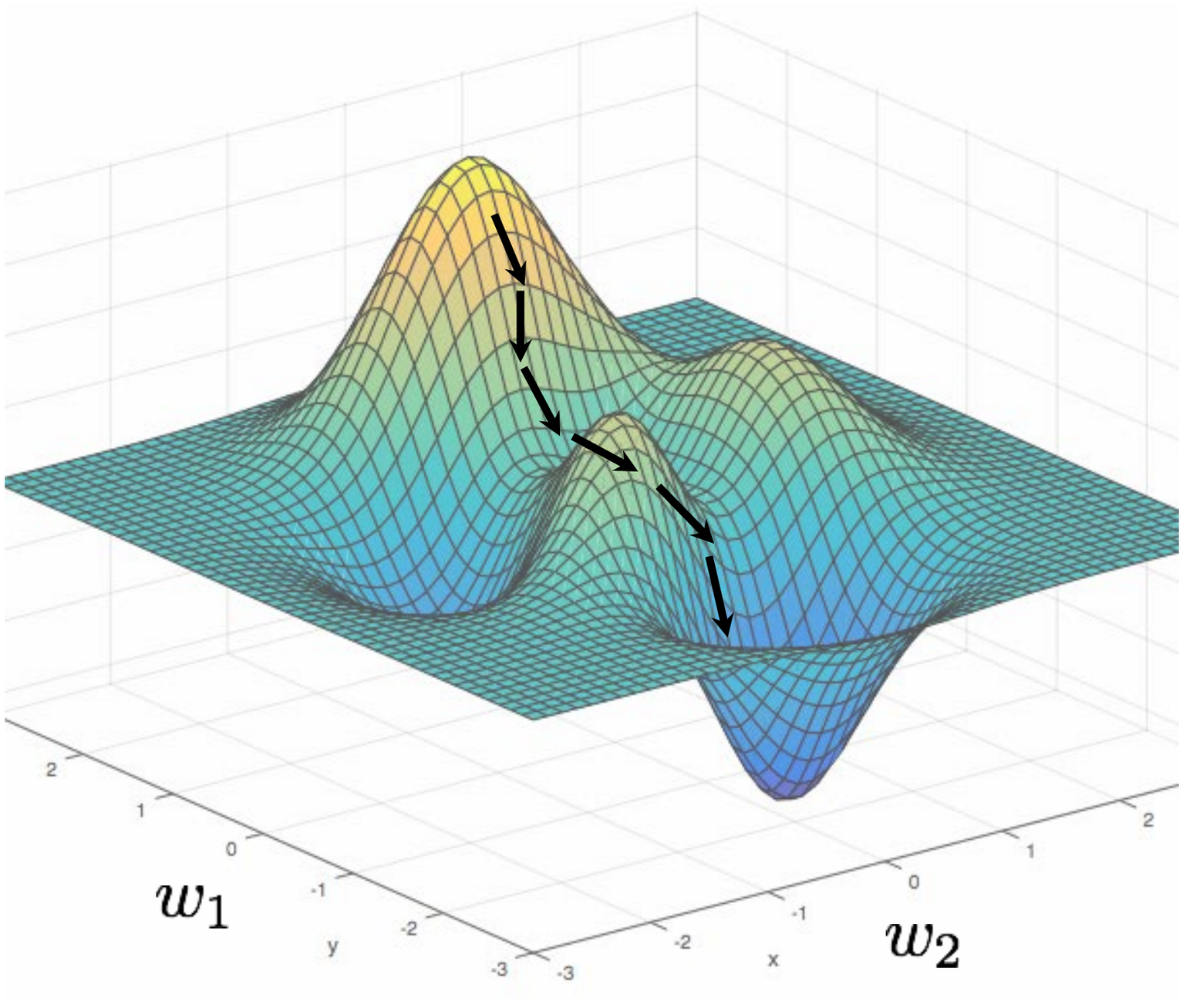
One iteration of gradient descent:

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

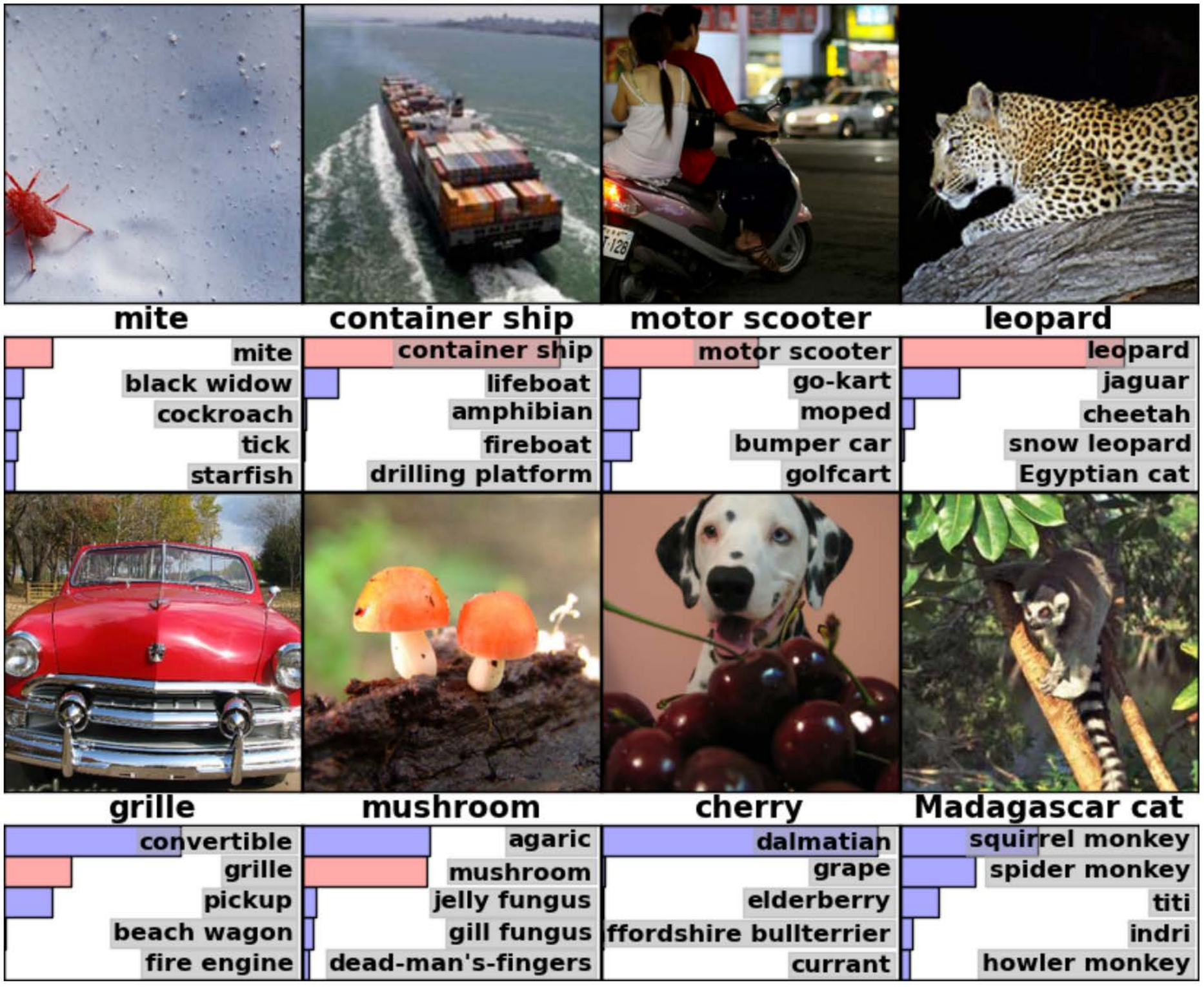
learning rate

Gradient descent

$L(\mathbf{w})$

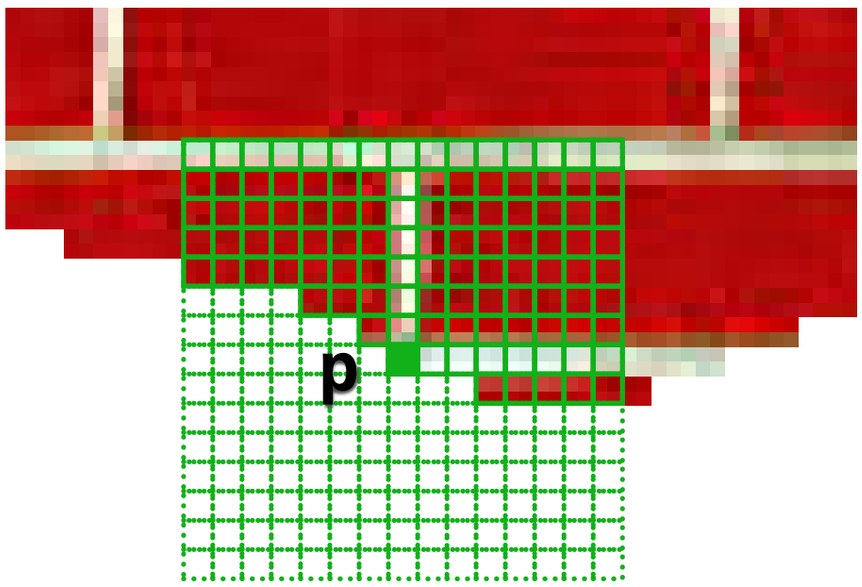


$$p(c|\mathbf{x})$$



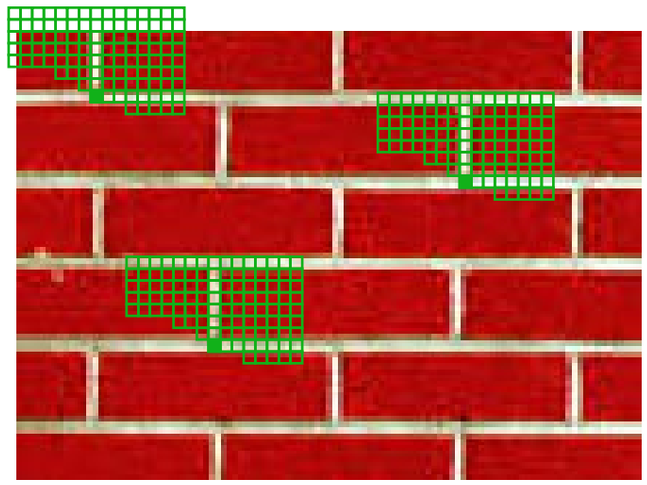
[Krizhevsky et al. NIPS 2012]

Texture synthesis by non-parametric sampling



Synthesizing a pixel

non-parametric sampling



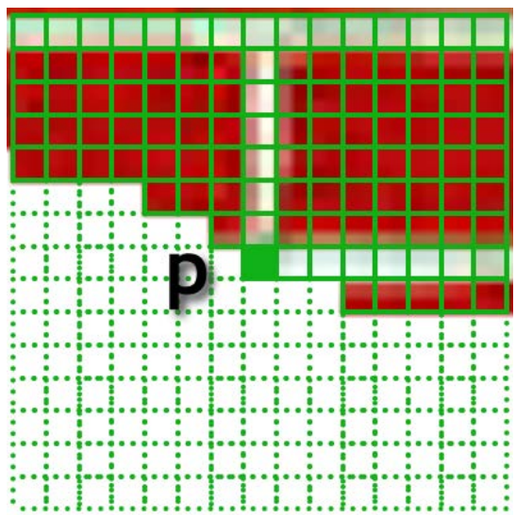
Input image

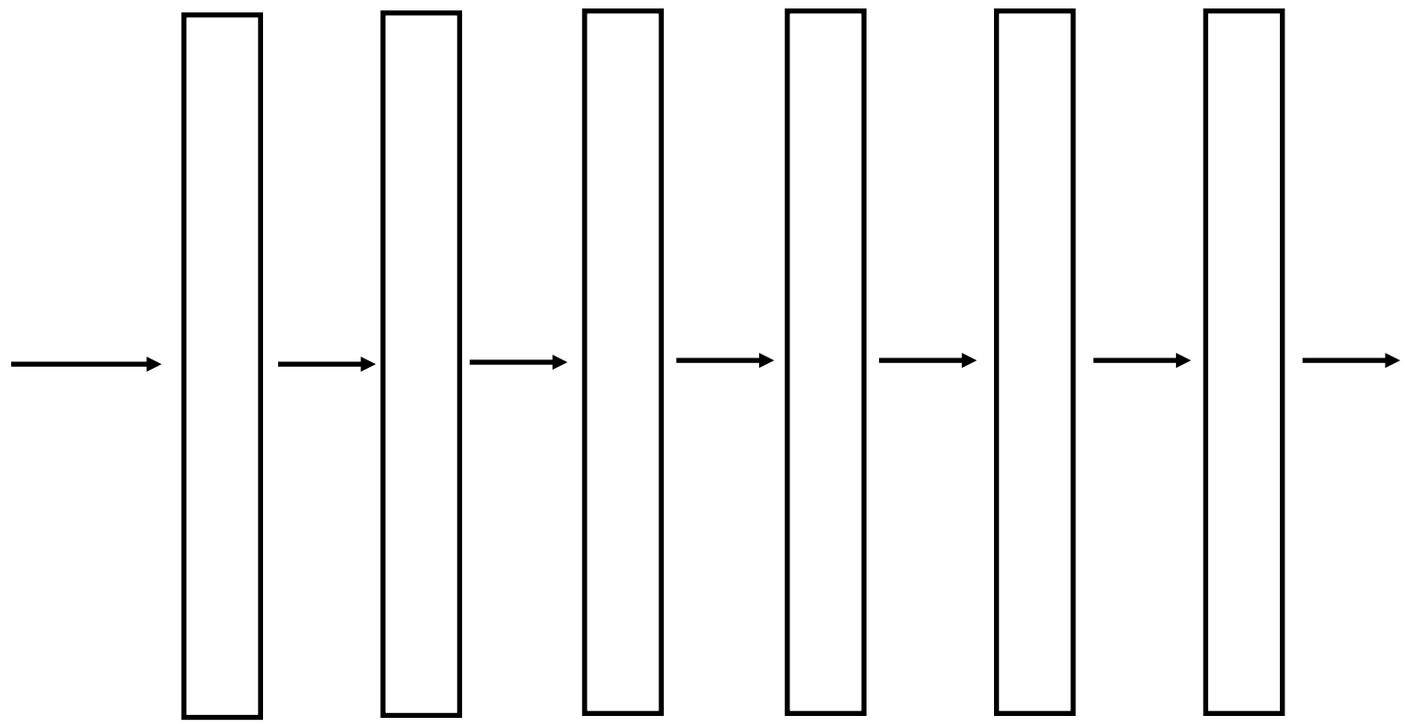
Models $P(p|N(p))$

[Efros & Leung 1999]

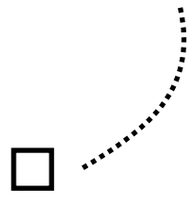
Texture synthesis with a deep net

Input partial image





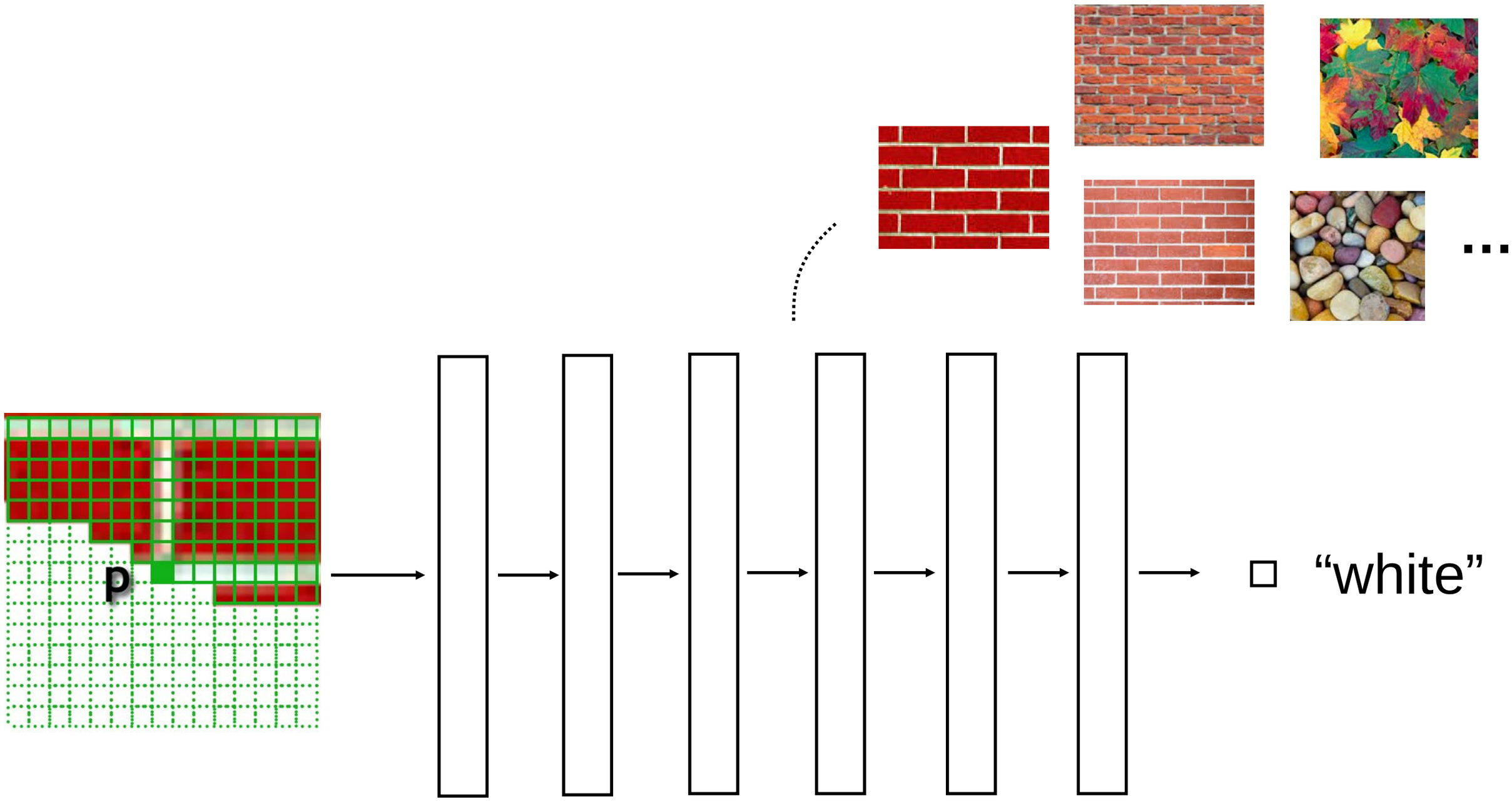
Predicted color of next pixel



“white”

[van der Oord et al. 2016]

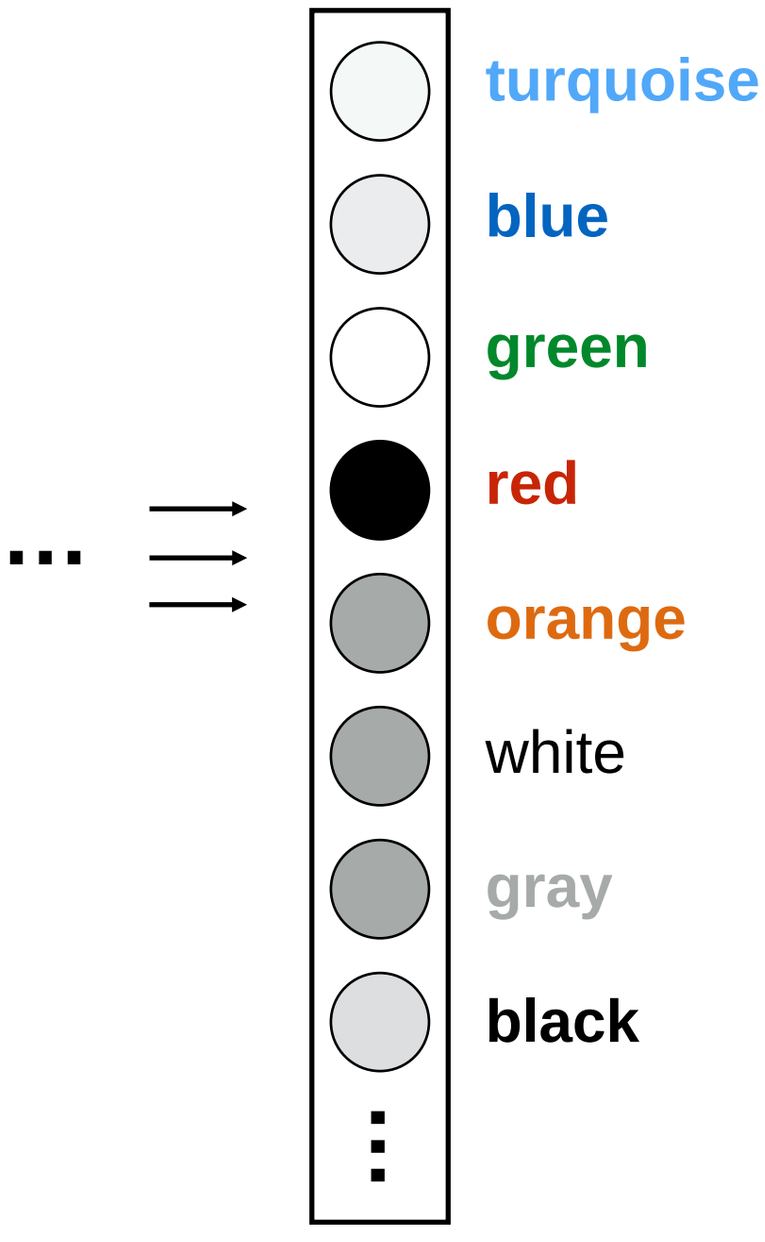
Texture synthesis with a deep net

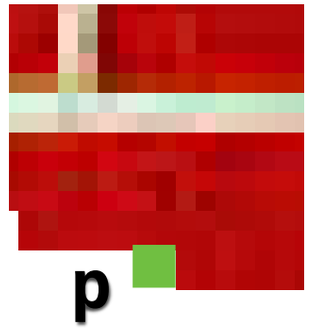


[van der Oord et al. 2016]

Sampling

Network output

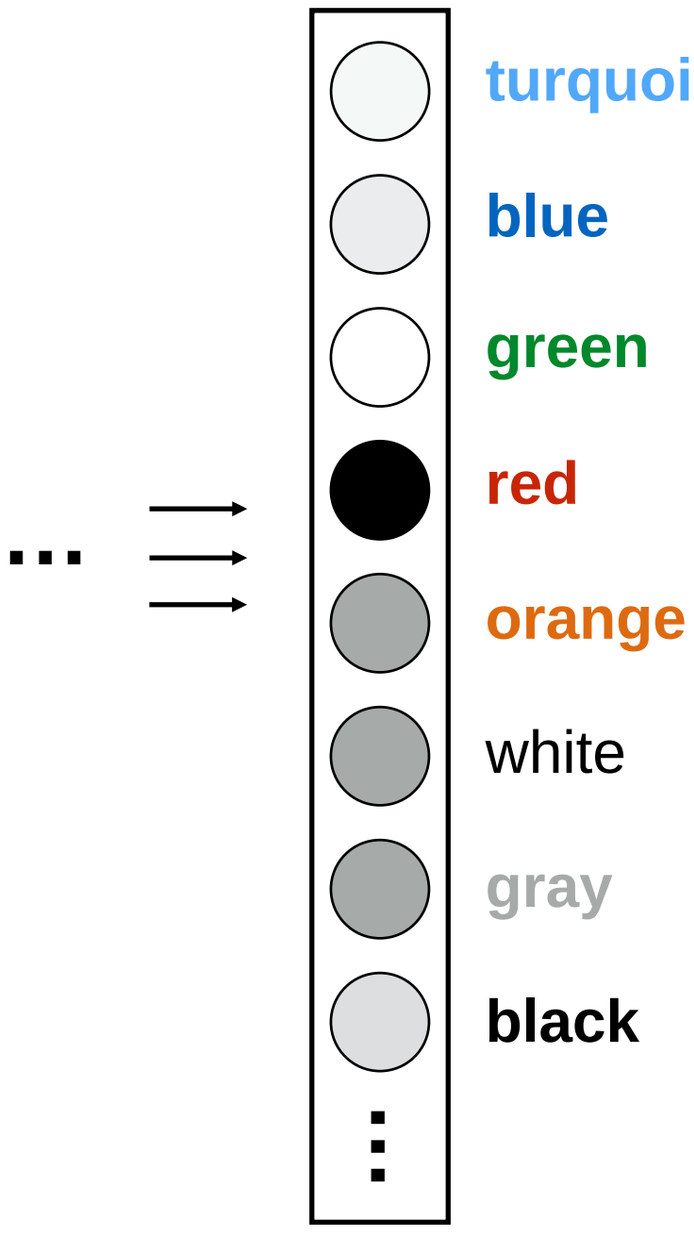


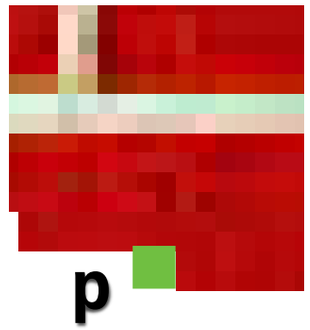


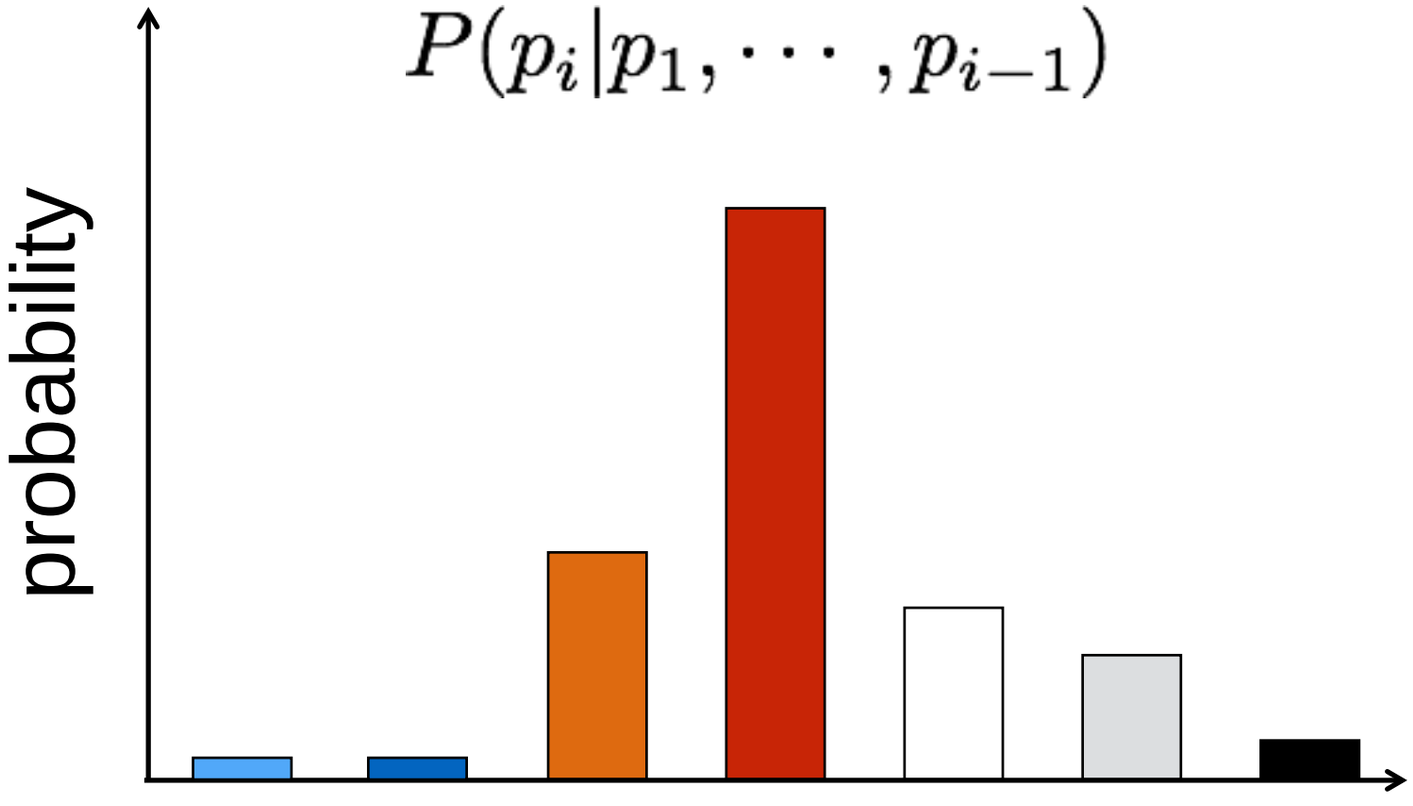
$$P(p_i|p_1, \cdots, p_{i-1})$$

Sampling

Network output

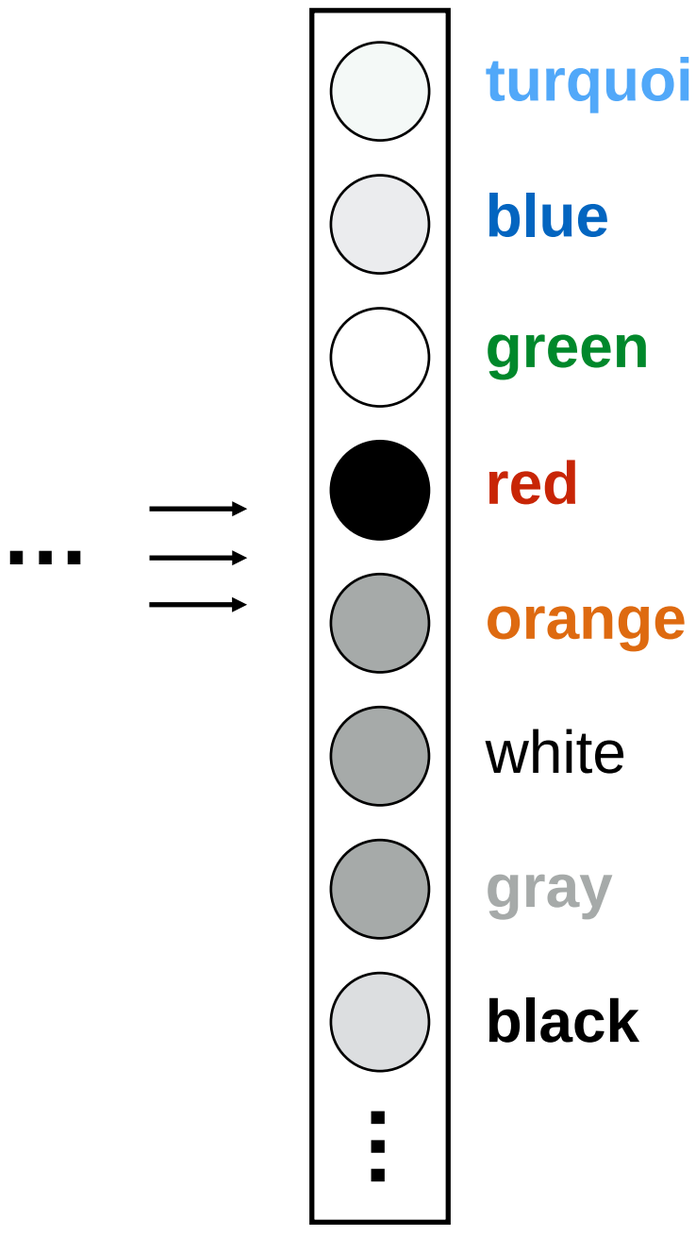


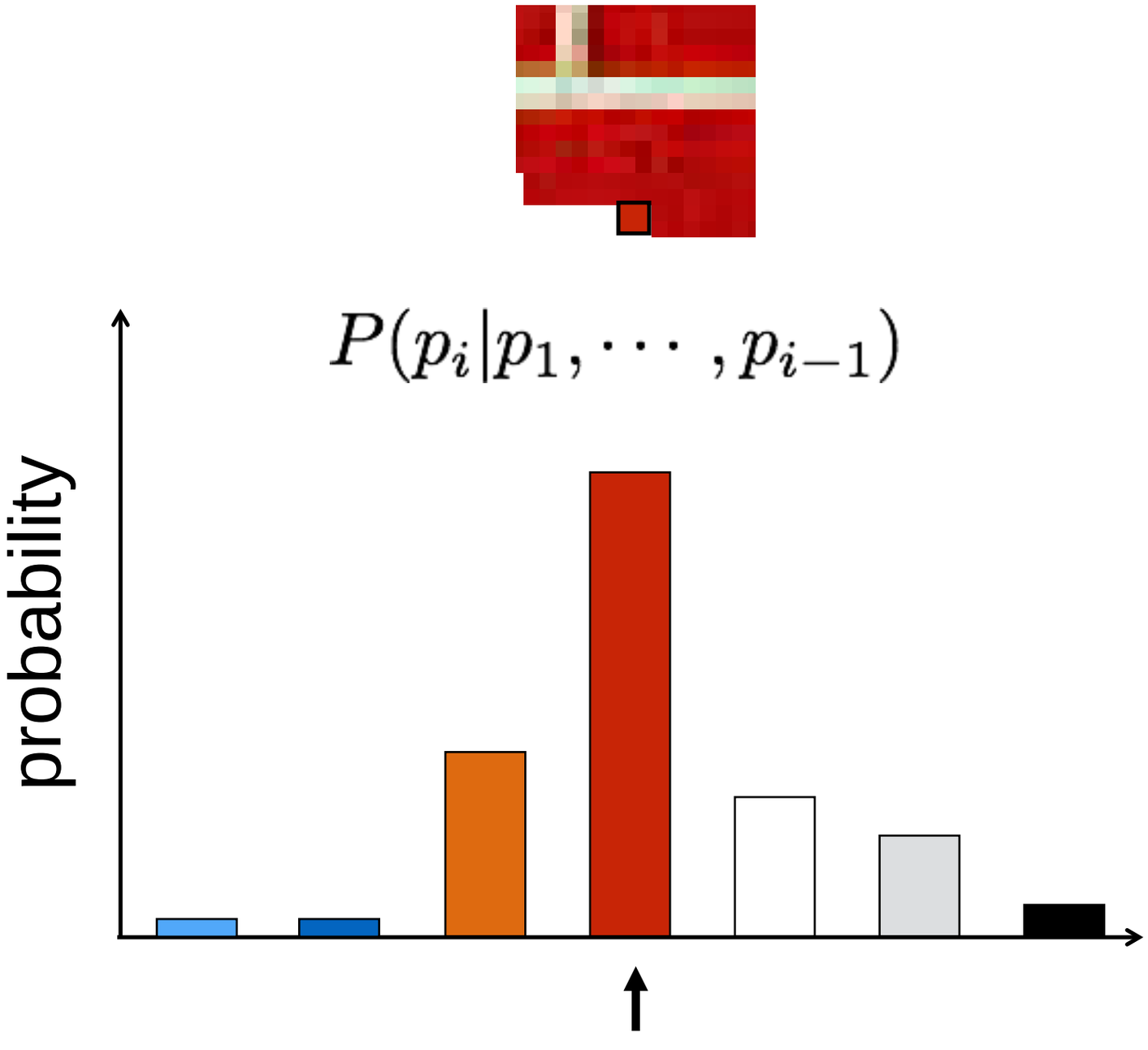




Sampling

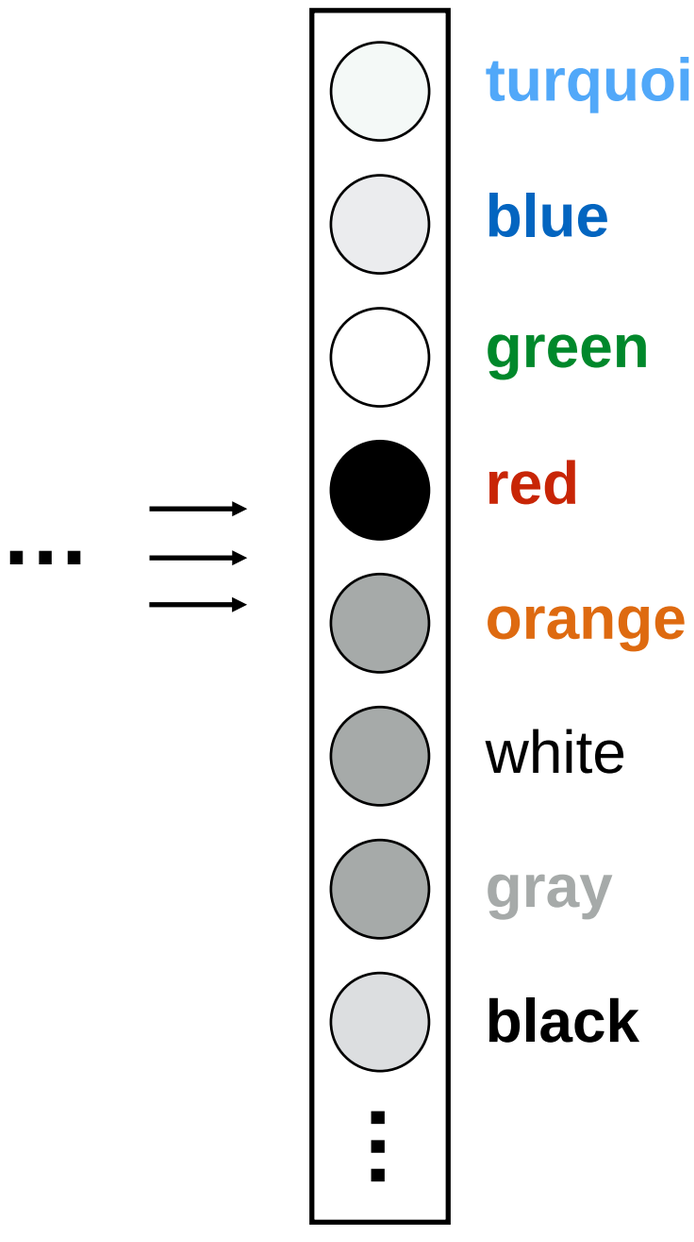
Network output

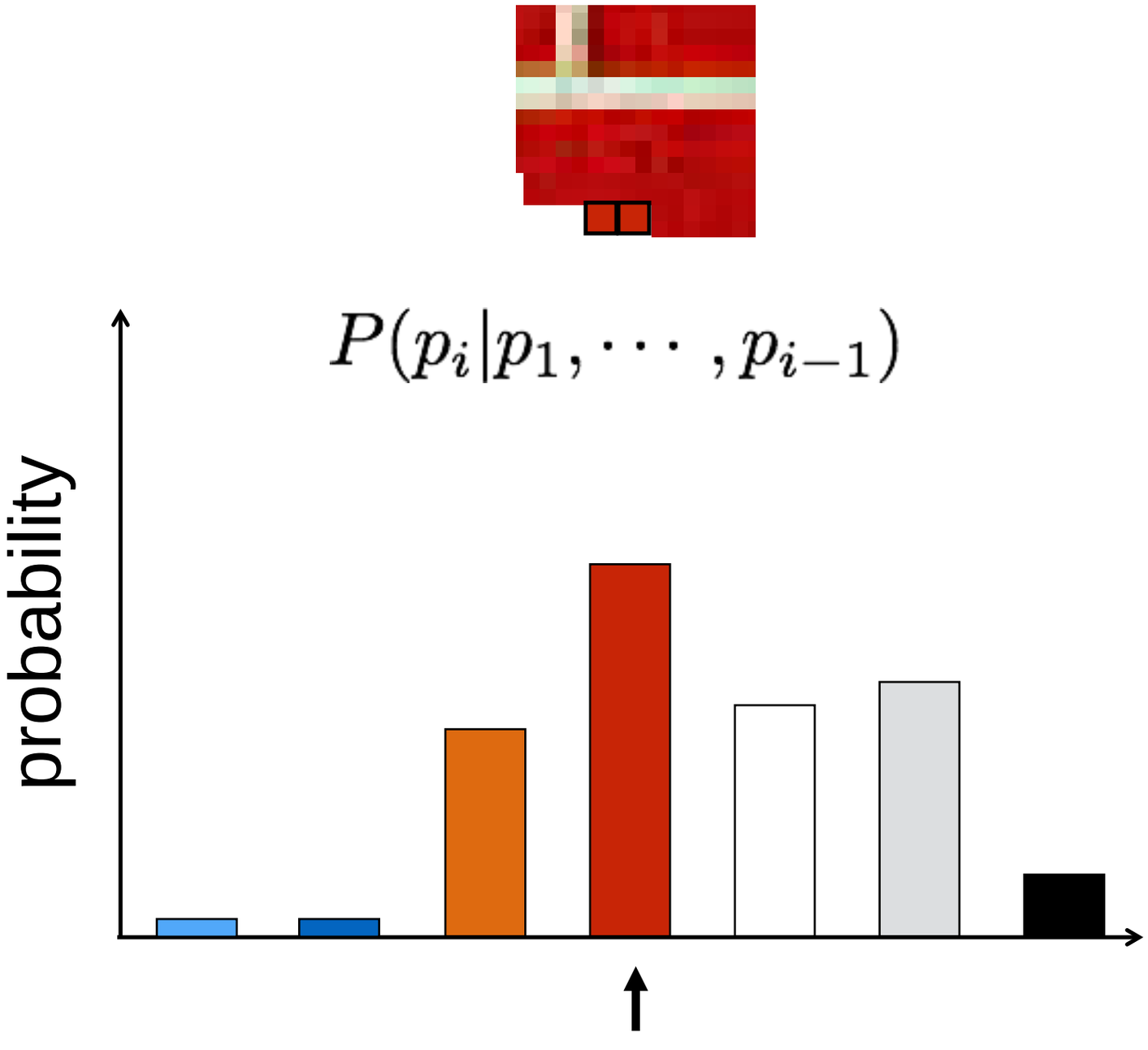




Sampling

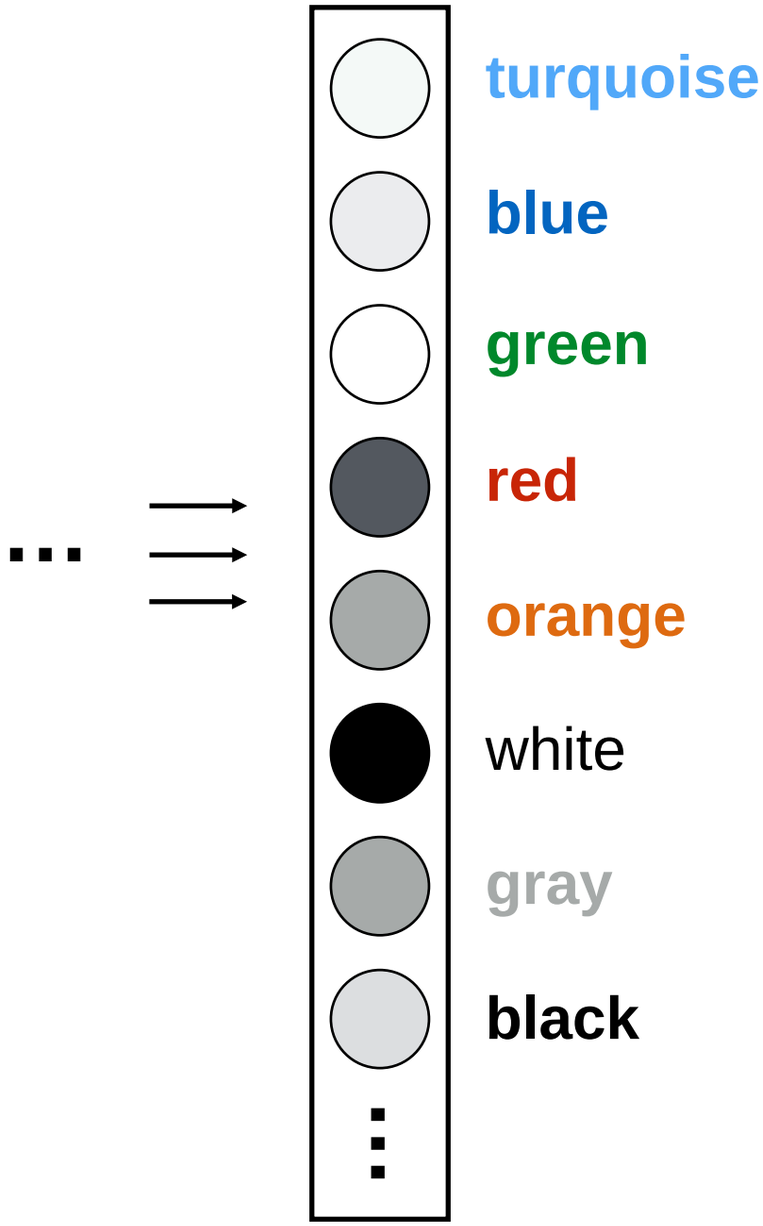
Network output

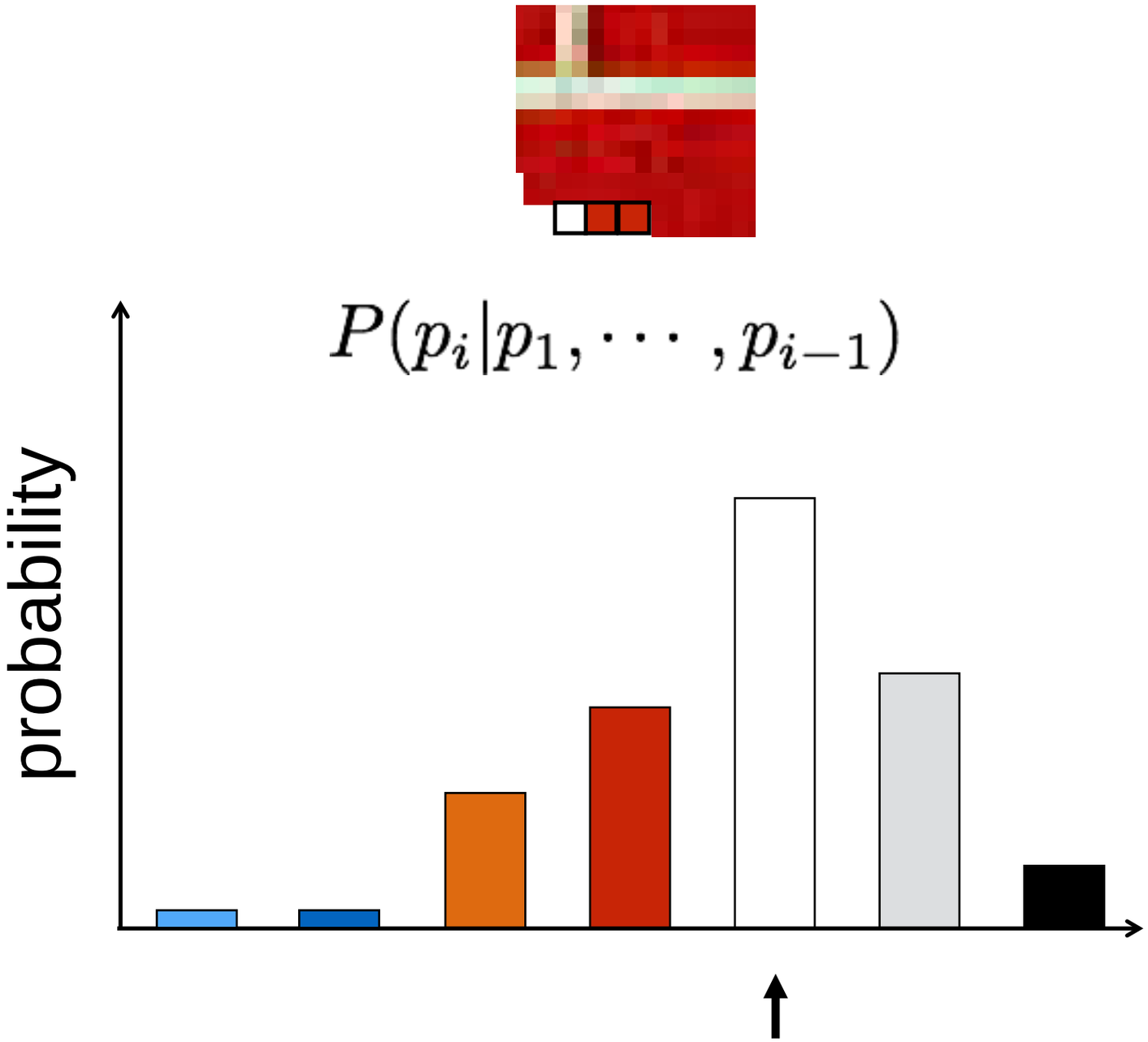




Sampling

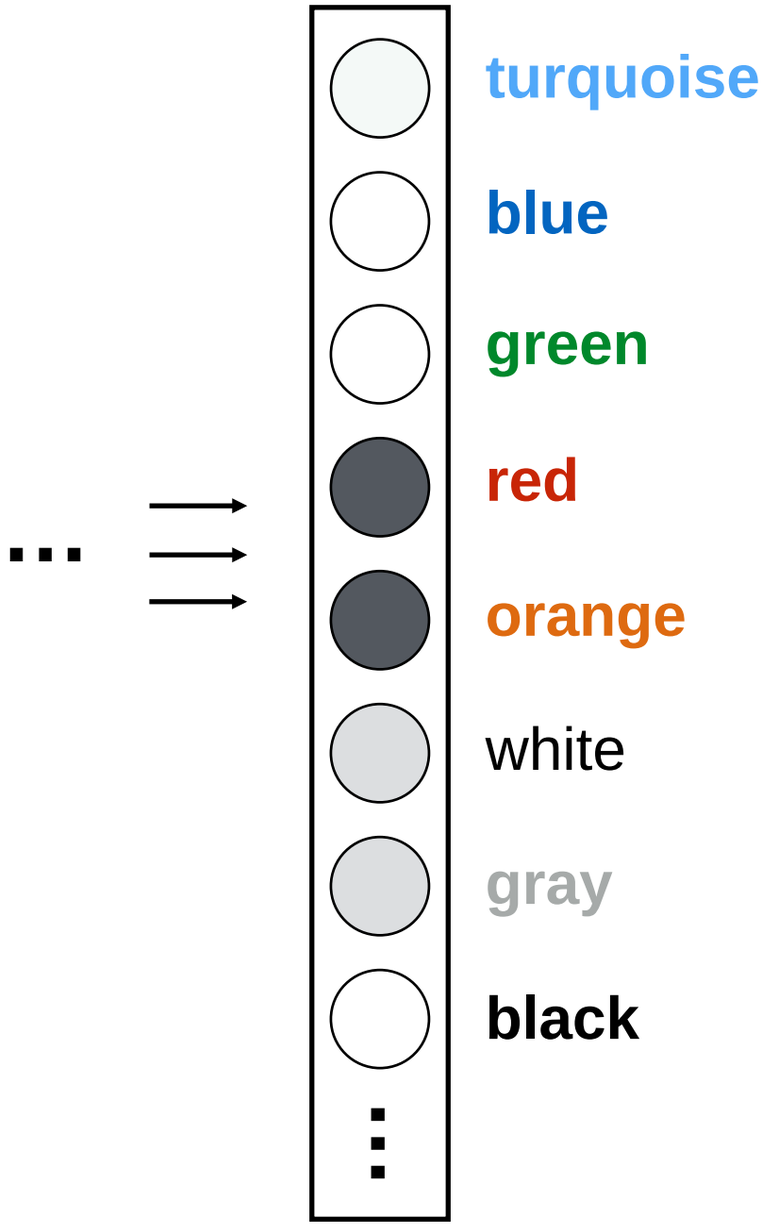
Network output

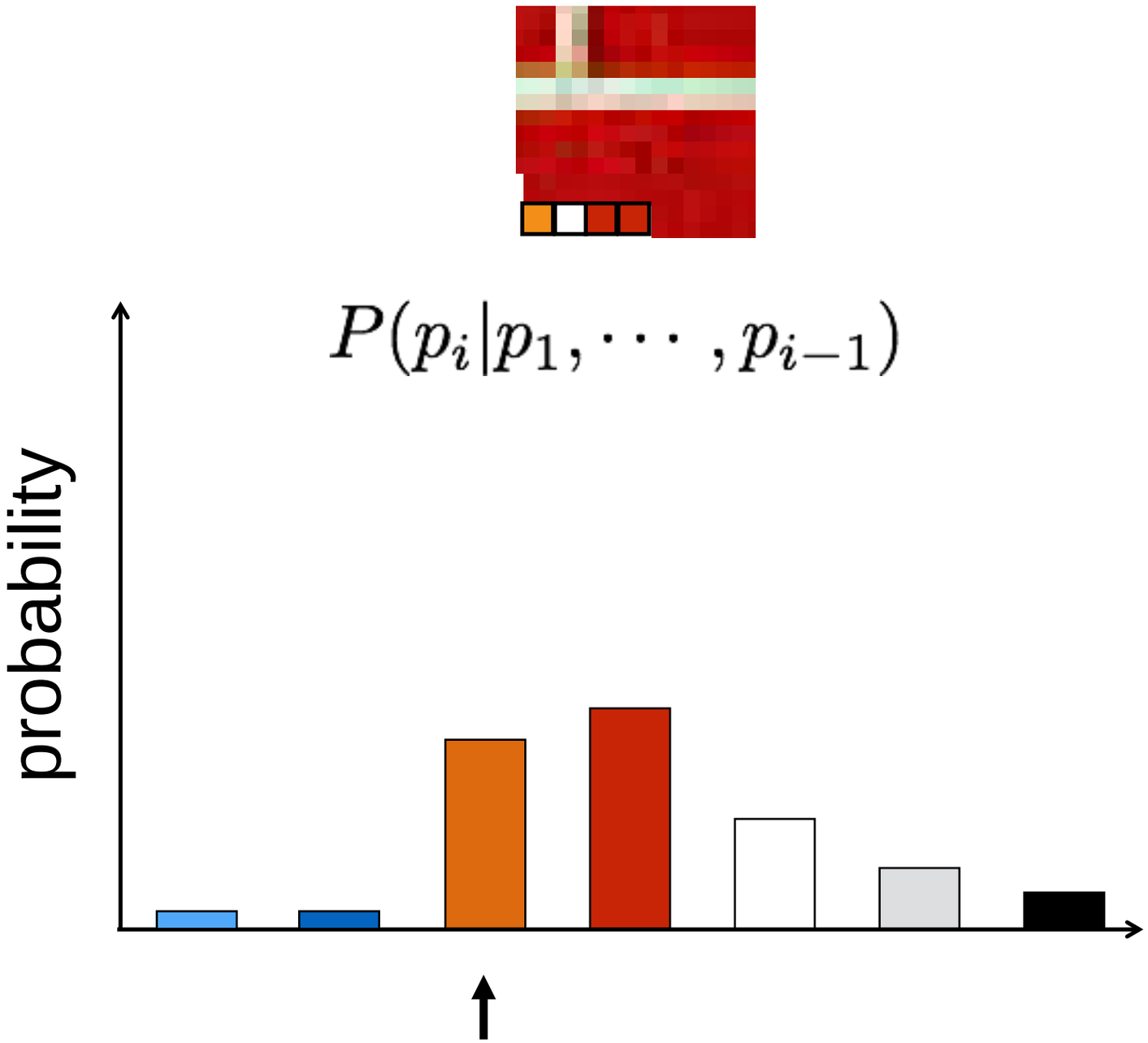




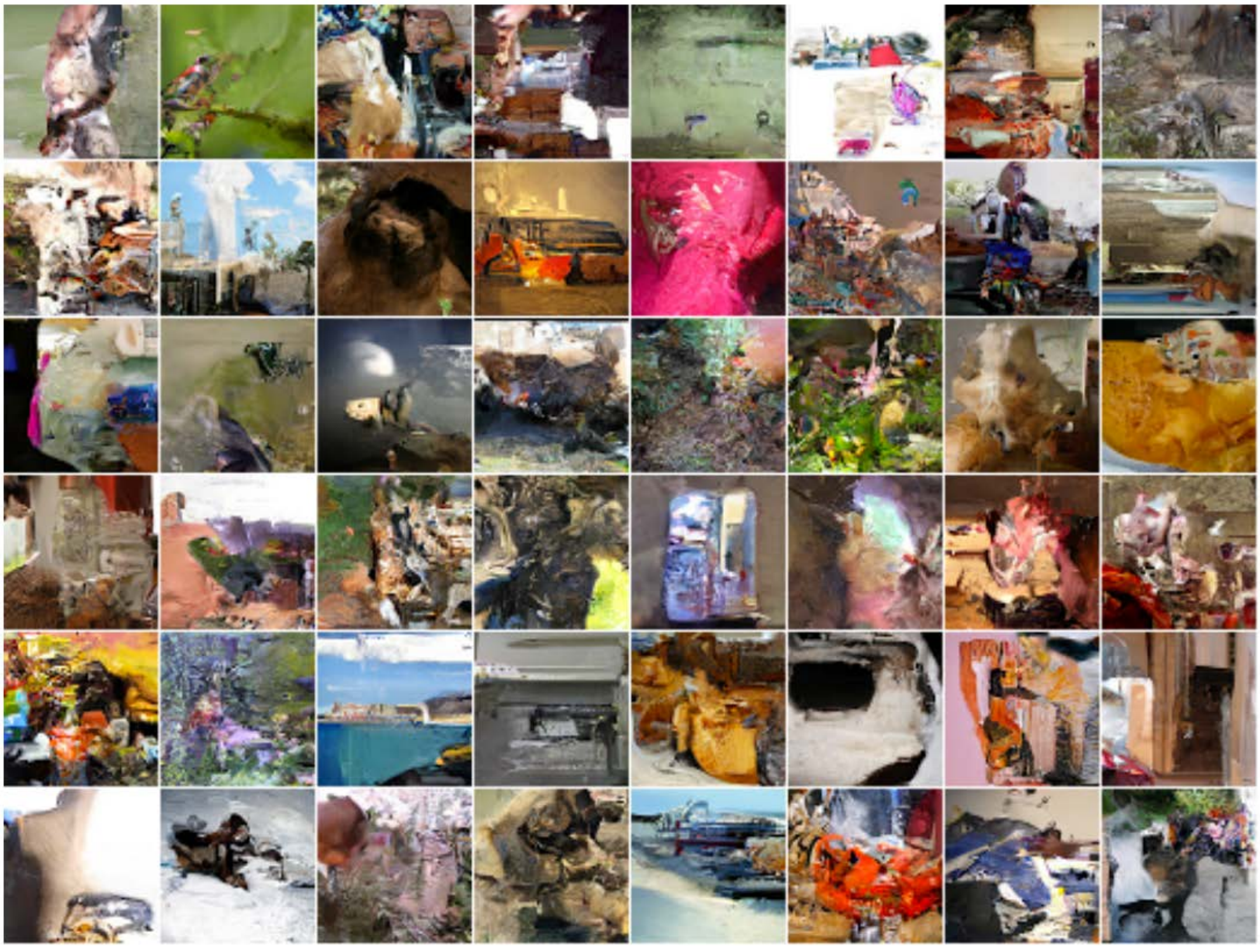
Sampling

Network output



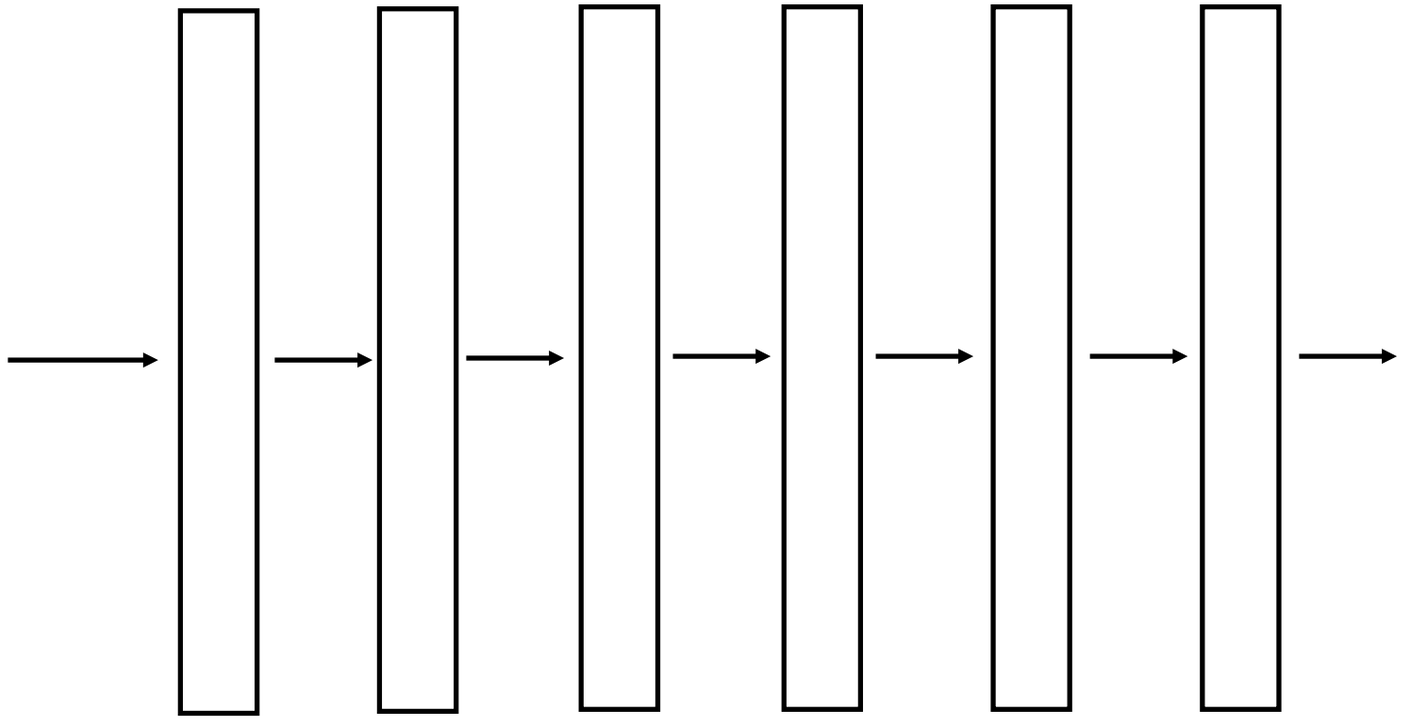


Sampling full images



Inpainting

...



- “yellow”