

CNN









!!! Warning !!!

Learning jargon is always painful...

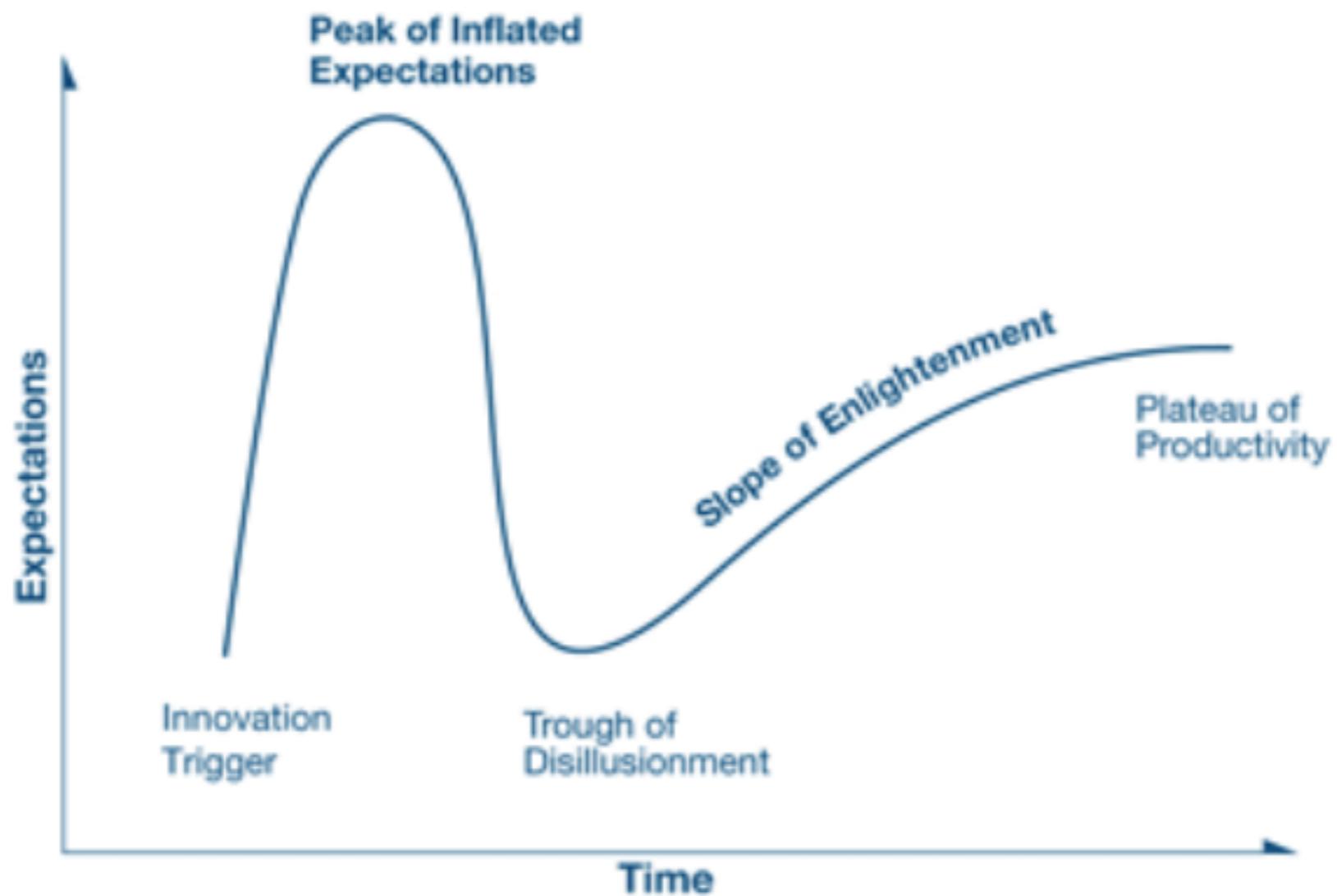
...even if the concepts behind the jargon are not hard.

So, let's get used to it.

“In mathematics you don't understand things.
You just get used to them.”

von Neumann (a joke)

Gartner Hype Cycle



The limits of learning?

So far...

- PASCAL VOC = ~75%
- ImageNet = ~75%; human performance = ~95%

Smart human brains used intuition and understanding of how we think vision works, and it's pretty good.

Image formation (+database+labels)

Captured+manual.

Filtering (gradients/transforms)

Hand designed.

Feature points (saliency+description)

Hand designed.

Dictionary building (compression)

Hand designed.

Classifier (decision making)

Learned.

Recognition:

Classification

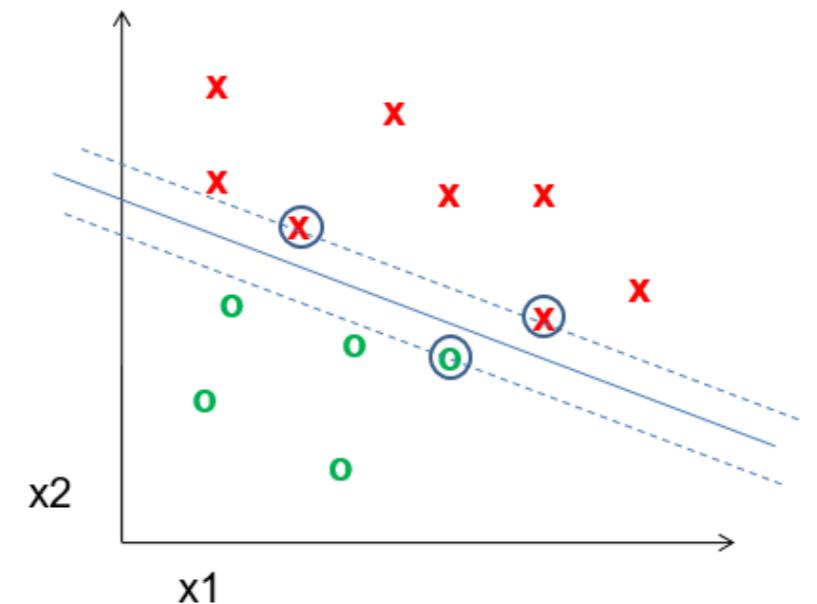
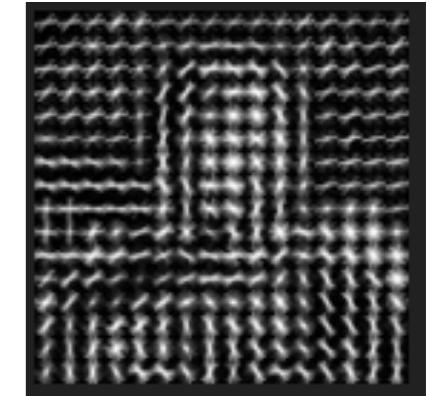
Object Detection

Segmentation

Well, what do we have?

Best performing vision systems have commonality:

- Hand designed features
 - Gradients + non-linear operations (exponentiation, clamping, binning)
 - Features in combination (parts-based models)
 - Multi-scale representations
- Machine learning from databases
- Some classifiers (e.g., SVM)



But it's still not that good...

- PASCAL VOC = ~75%
- ImageNet = ~75%; human performance = ~95%

Problems:

- Lossy features
- Lossy quantization
- Imperfect classifier

But it's still not that good...

- PASCAL VOC = ~75%
- ImageNet = ~75%; human performance = ~95%

How to solve?

- Features: More principled modeling?
- Quantization: More data and more compute?
It's just an interpolation problem; let's represent the space with less approximation.
- Classifier: ...

Wouldn't it be great if we could...

Image formation (+database+labels)

Captured+manual.

Filtering (gradients/transforms)

Learned.
(space specified a bit)

Feature points (saliency+description)

Learned.

Dictionary building (compression)

Learned.

Classifier (decision making)

Learned.

Recognition:

Classification

Object Detection

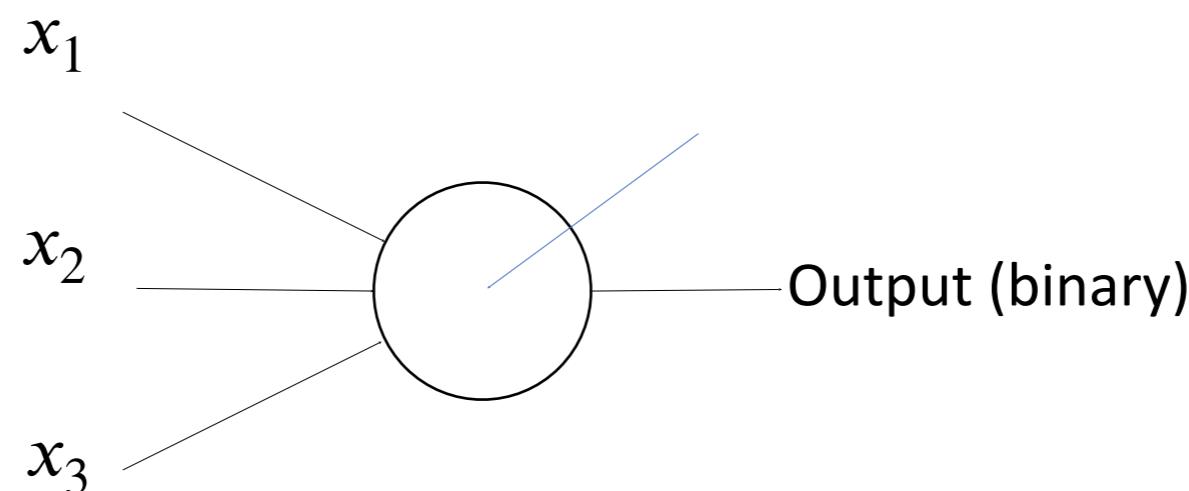
Segmentation

End to end
learning!

Neural Networks

Neural Networks

- Basic building block for composition is a perceptron (Rosenblatt c.1960)
- Linear classifier – vector of weights w and a ‘bias’ b



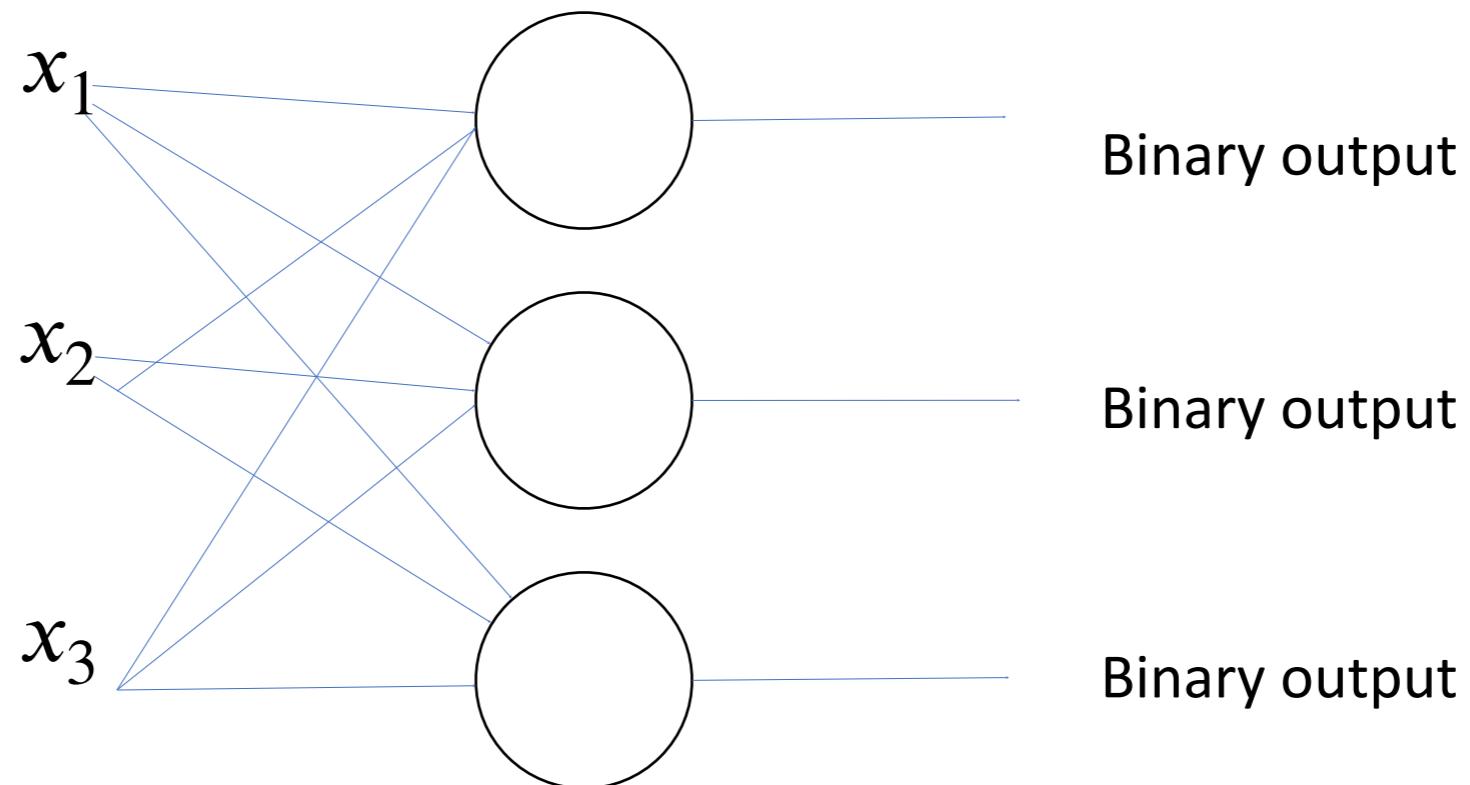
$$\text{output} = \begin{cases} 0 & \text{if } w \cdot x + b \leq 0 \\ 1 & \text{if } w \cdot x + b > 0 \end{cases} \quad w \cdot x \equiv \sum_j w_j x_j;$$

Binary classifying an image

- Each pixel of the image would be an input.
- So, for a 28×28 image, we vectorize.
- $x = 1 \times 784$
- w is a vector of weights for each pixel, 784×1
- b is a scalar bias per perceptron
- $\text{result} = xw + b \rightarrow (1 \times 784) \times (784 \times 1) + b = (1 \times 1) + b$

Neural Networks - multiclass

- Add more perceptrons

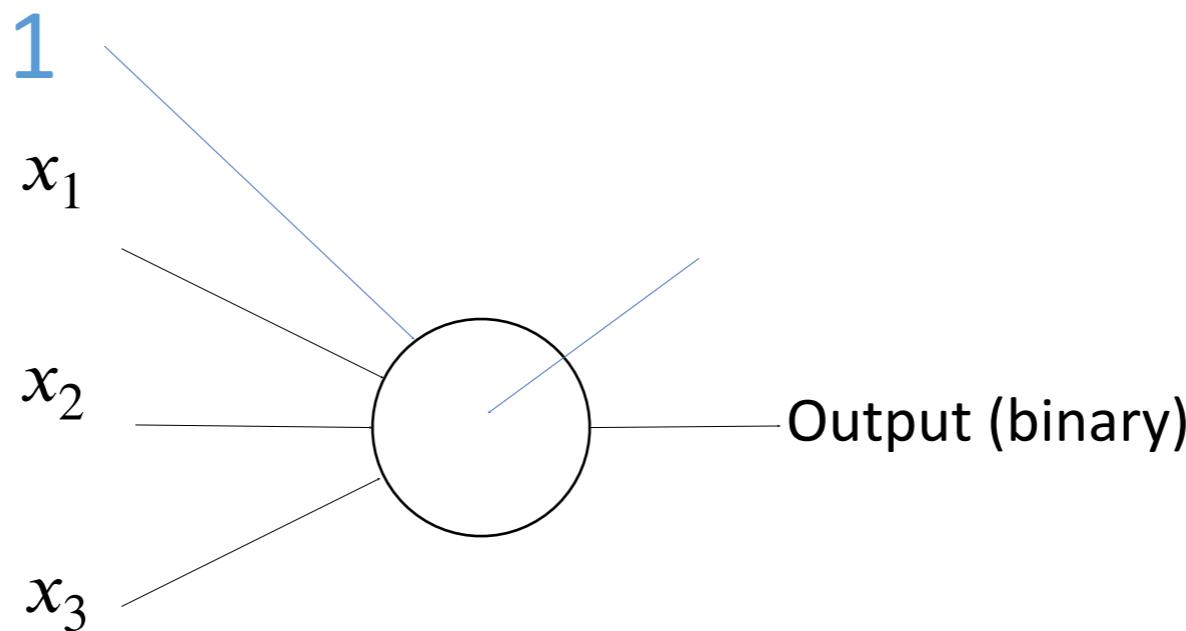


Multi-class classifying an image

- Each pixel of the image would be an input.
- So, for a 28×28 image, we vectorize.
- $x = 1 \times 784$
- W is a matrix of weights for each pixel/each perceptron
 - $W = 10 \times 784$ (10-class classification)
- b is a bias per perceptron (vector of biases); (1×10)
- $\text{result} = xW + b \rightarrow (1 \times 784) \times (784 \times 10) + b$
 $\rightarrow (1 \times 10) + (1 \times 10) = \text{output vector}$

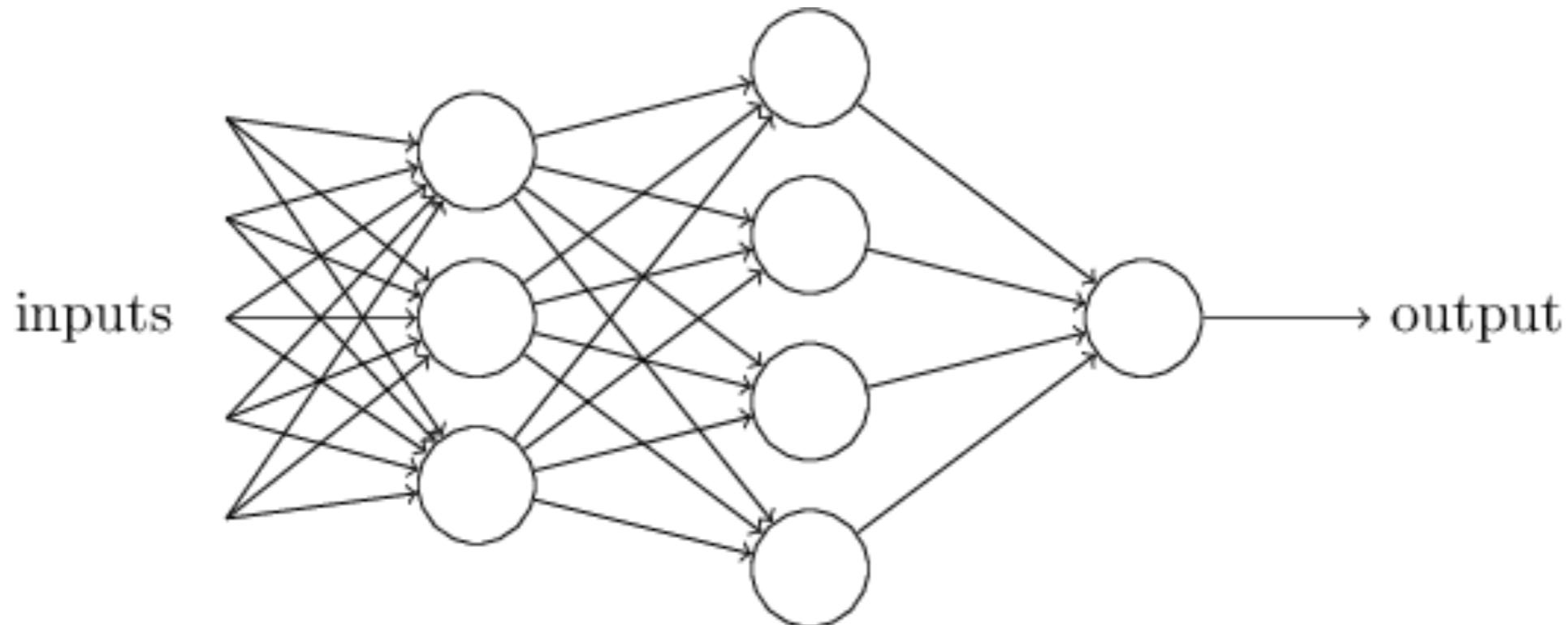
Bias convenience

- To turn this classification operation into a multiplication only:
 - Create a ‘fake’ feature with value 1 to represent the bias
 - Add an extra weight that can vary



$$\text{output} = \begin{cases} 0 & \text{if } w \cdot x \leq 0 \\ 1 & \text{if } w \cdot x > 0 \end{cases} \quad w \cdot x \equiv \sum_j w_j x_j$$

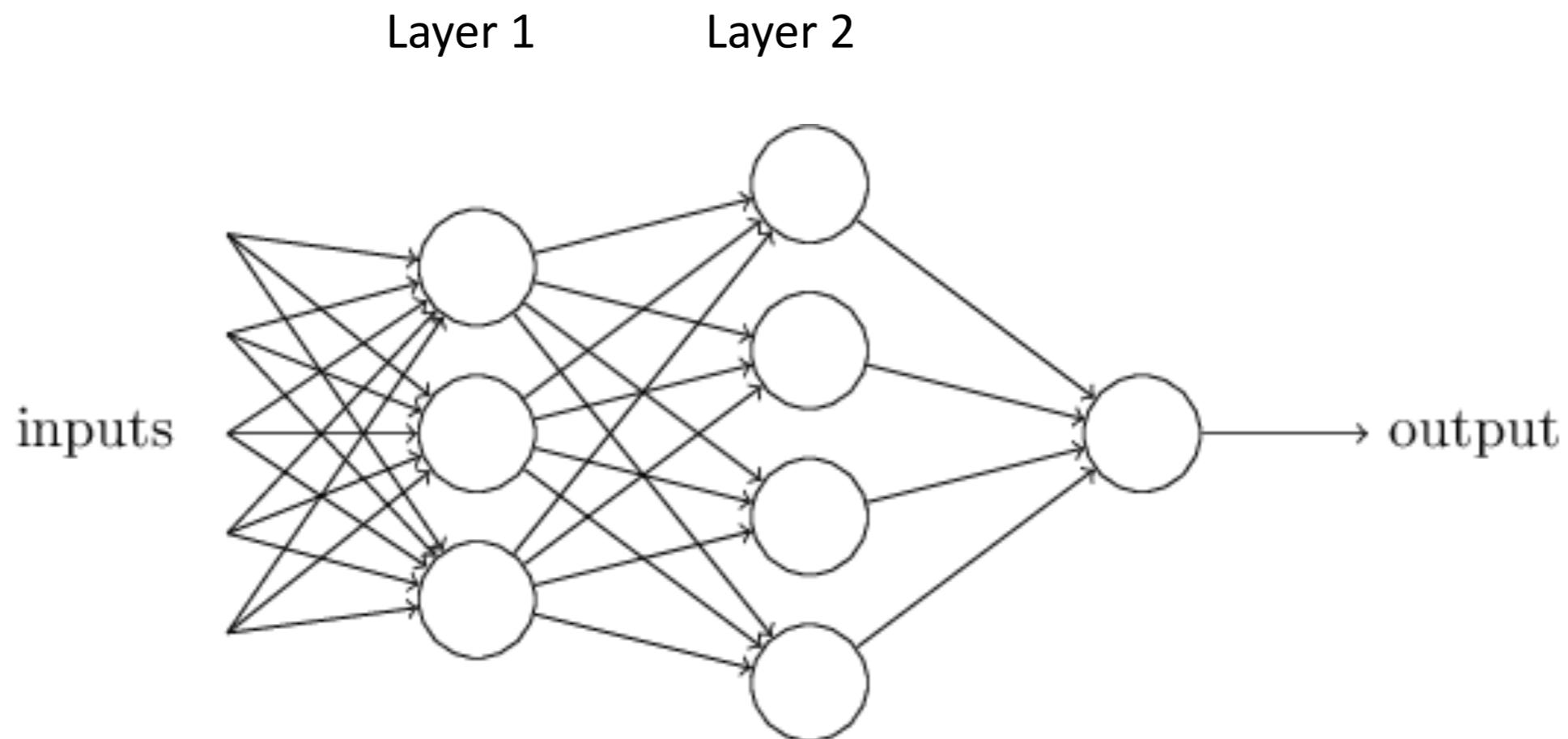
Composition



Attempt to represent complex functions as compositions of smaller functions.

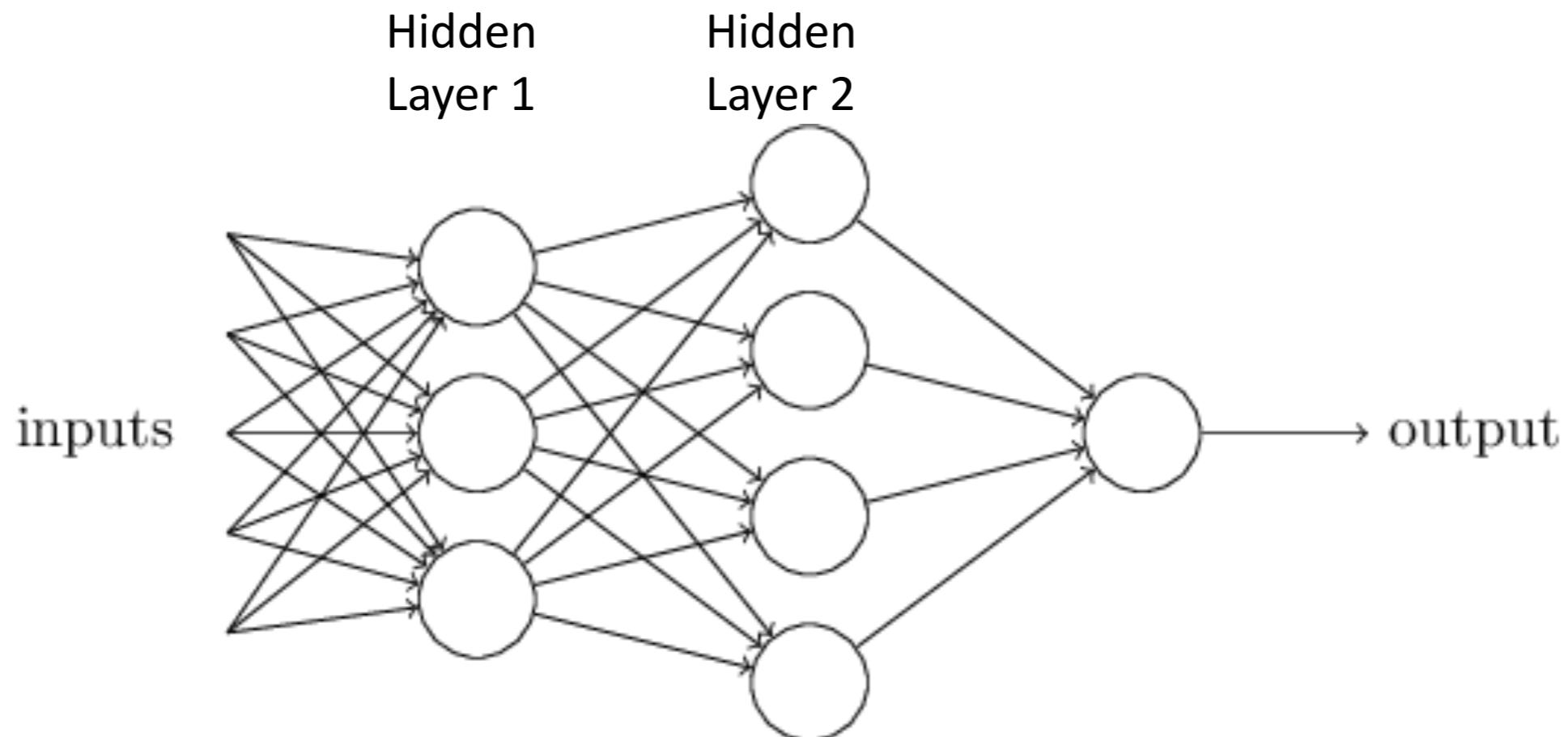
Outputs from one perception are fed into inputs of another perceptron.

Composition



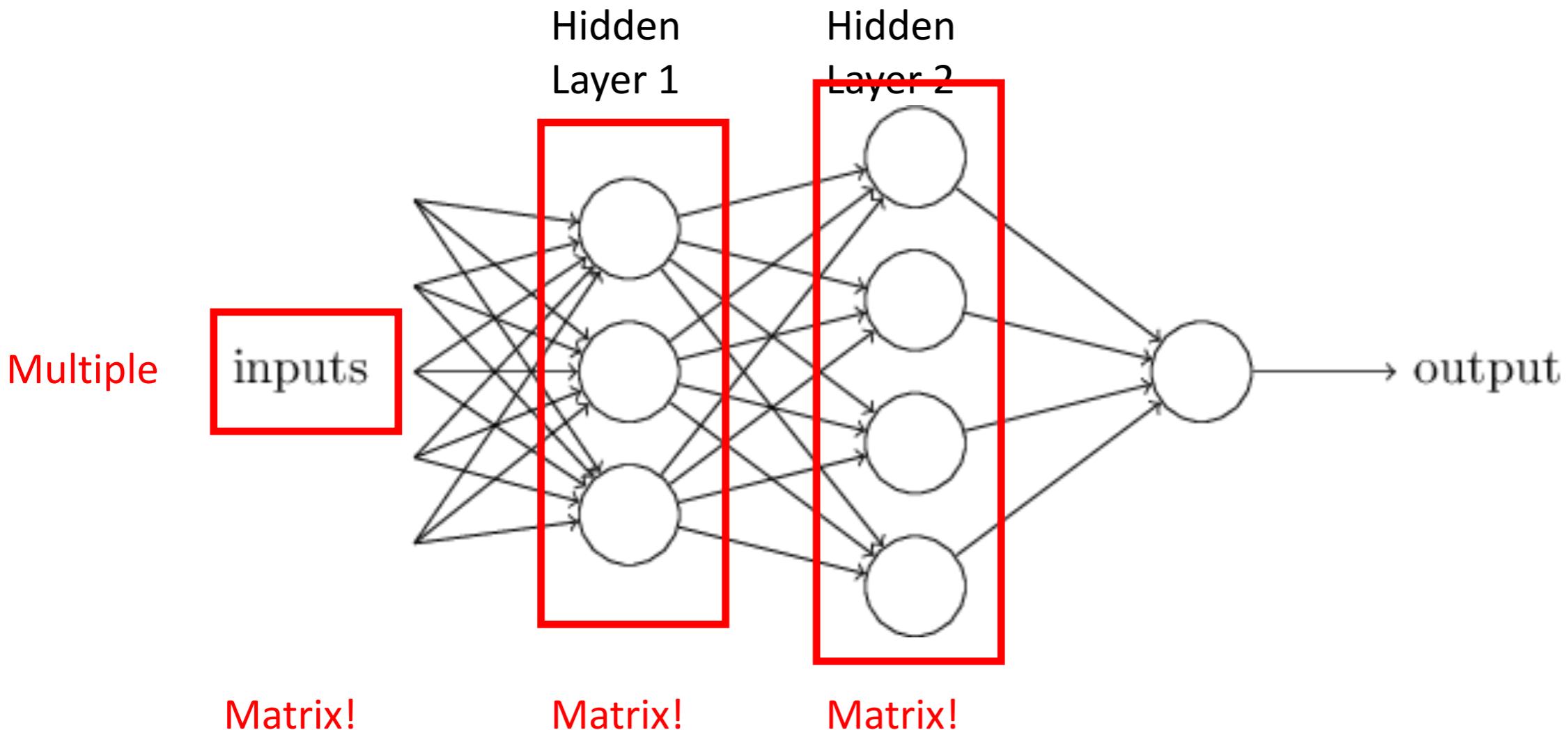
Sets of layers and the connections (weights) between them define the network architecture.

Composition



Layers that are in between the input and the output are called hidden layers, because we are going to learn their weights via an optimization process.

Composition



It's all just matrix multiplication!

GPUs -> special hardware for fast/large matrix multiplication.

Problem 1 with all linear functions

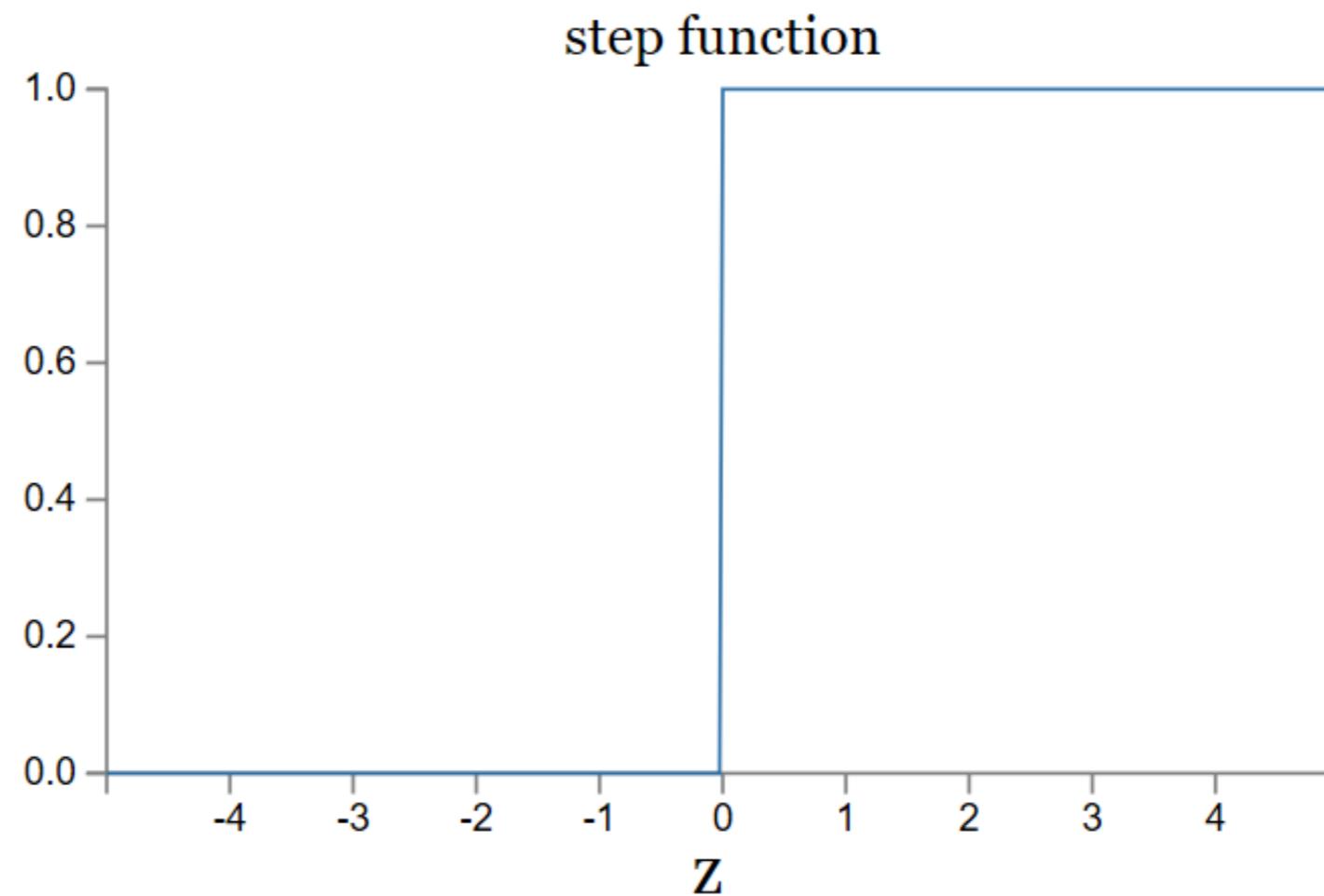
- We have formed chains of linear functions.
- We know that linear functions can be reduced
 - $g = f(h(x))$

Our composition of functions is really just a single function : (

Problem 2 with all linear functions

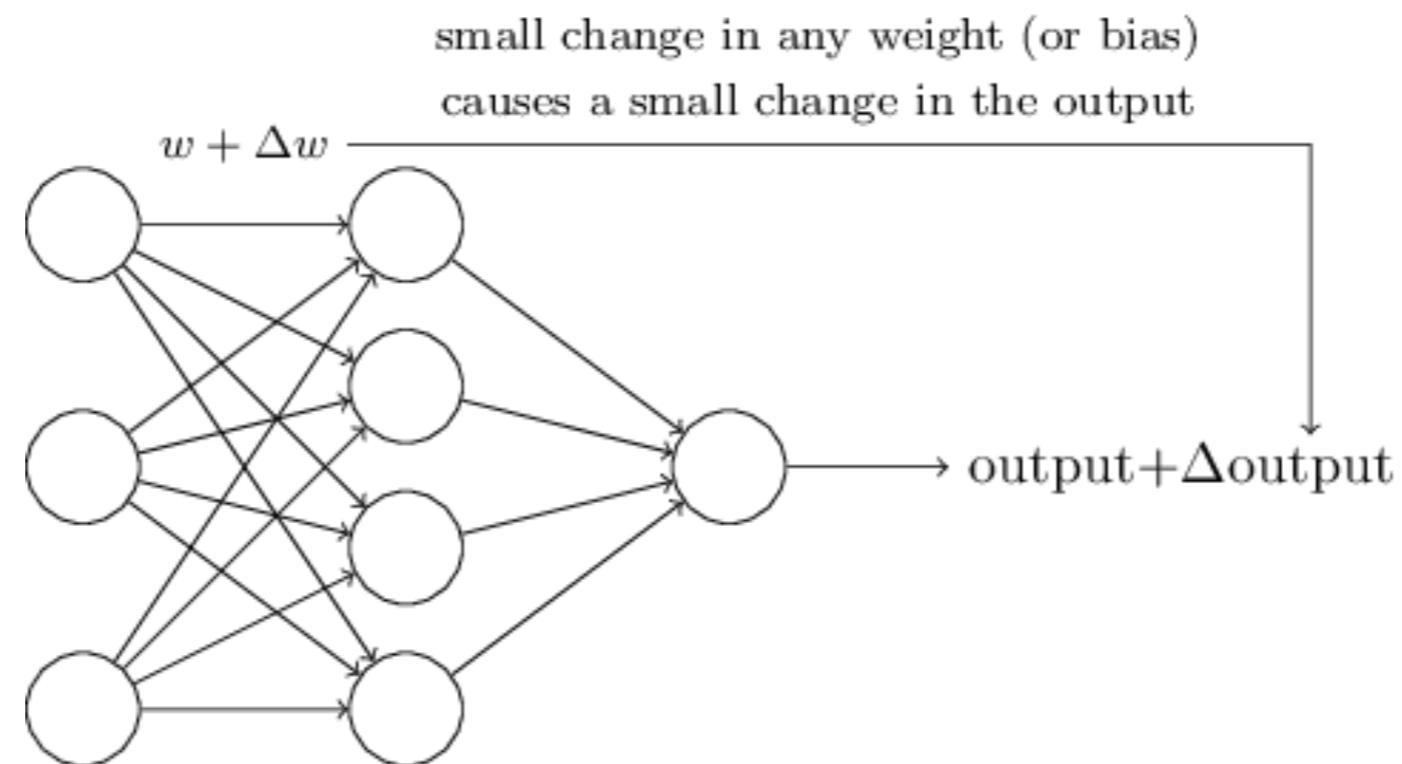
- Linear classifiers: small change in input can cause large change in binary output
= problem for composition of functions

Activation
function



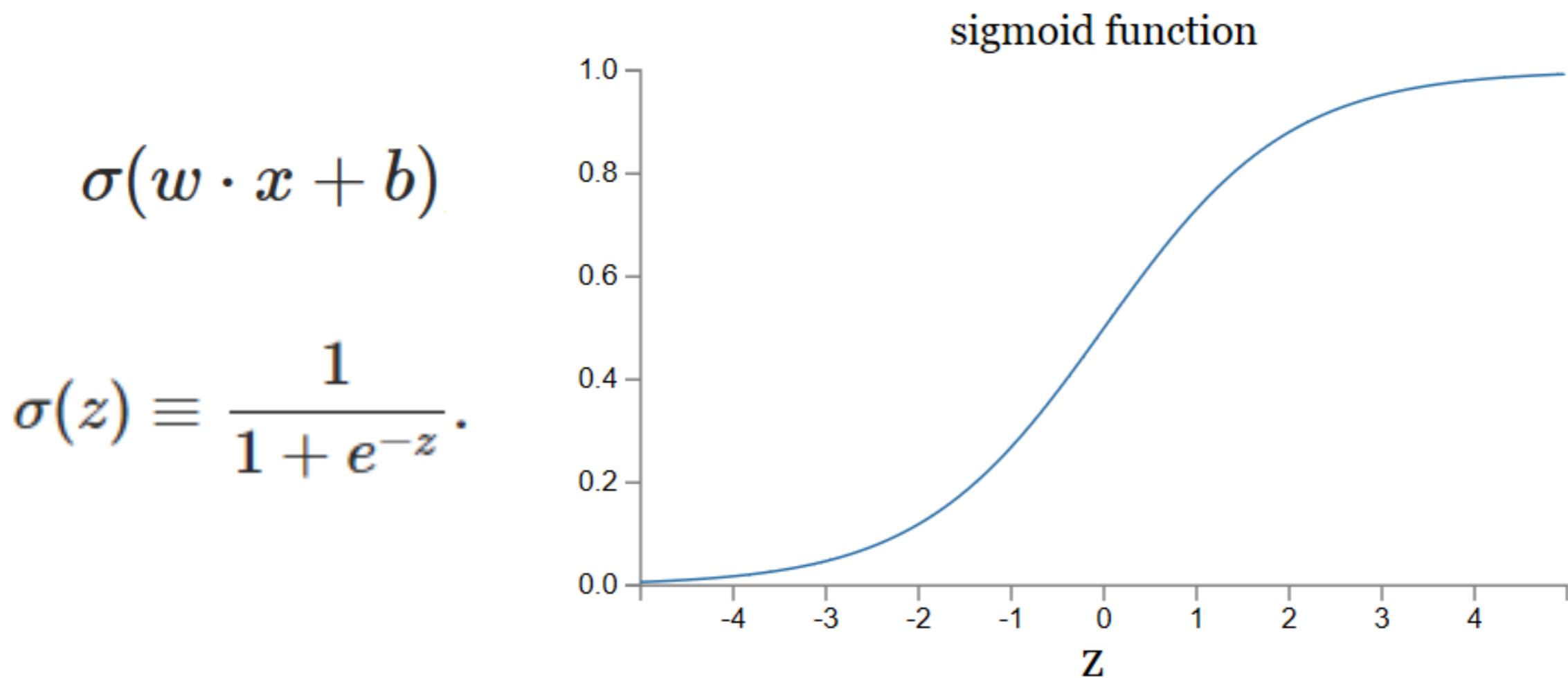
Problem 2 with all linear functions

- Linear classifiers: small change in input can cause large change in binary output.
- We want:



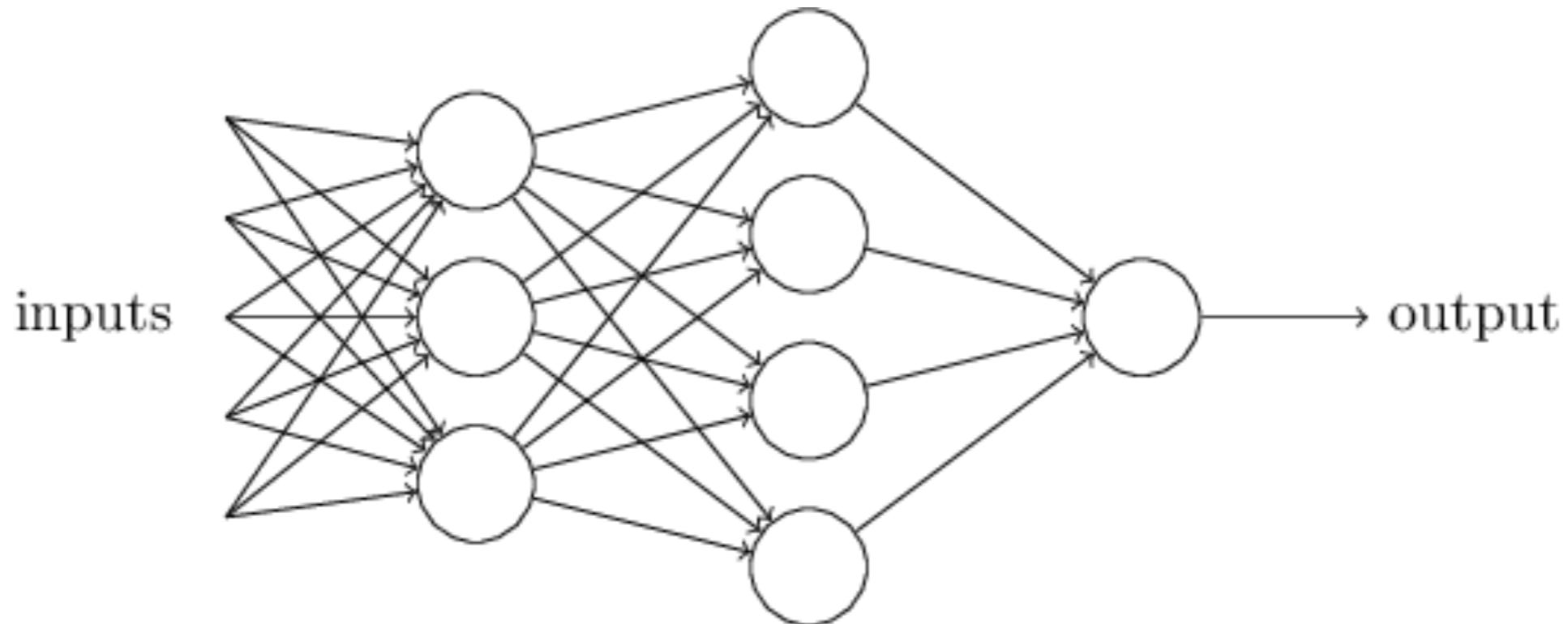
Let's introduce non-linearities

- We're going to introduce non-linear functions to transform the features.



Multi-layer perceptron (MLP)

- ...is a ‘fully connected’ neural network with non-linear activation functions.



- ‘Feed-forward’ neural network

MLP

- Use is grounded in theory
 - Universal approximation theorem (Goodfellow 6.4.1)
- Can represent a NAND circuit, from which any binary function can be built by compositions of NANDs
- With enough parameters, it can approximate any function (next lecture).

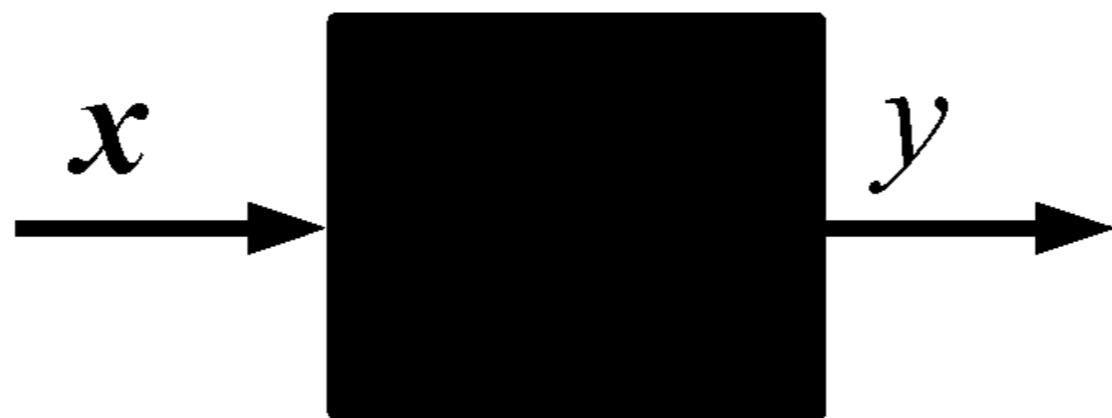
Supervised Learning

$\{(\mathbf{x}^i, y^i), i=1 \dots P\}$ training dataset

\mathbf{x}^i i-th input training example

y^i i-th target label

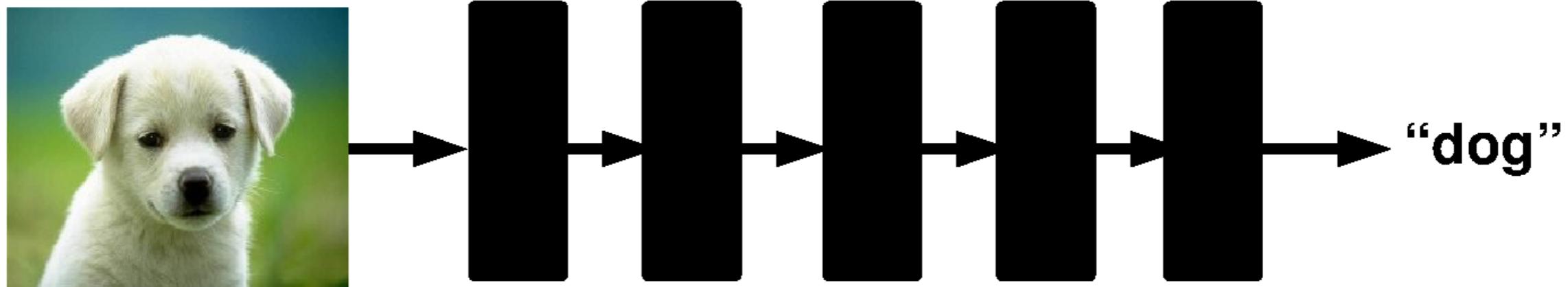
P number of training examples



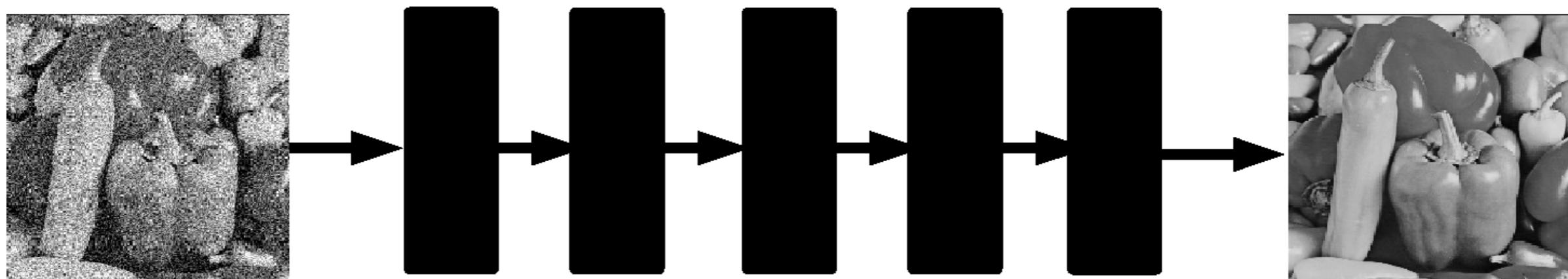
Goal: predict the target label of unseen inputs.

Supervised Deep Learning

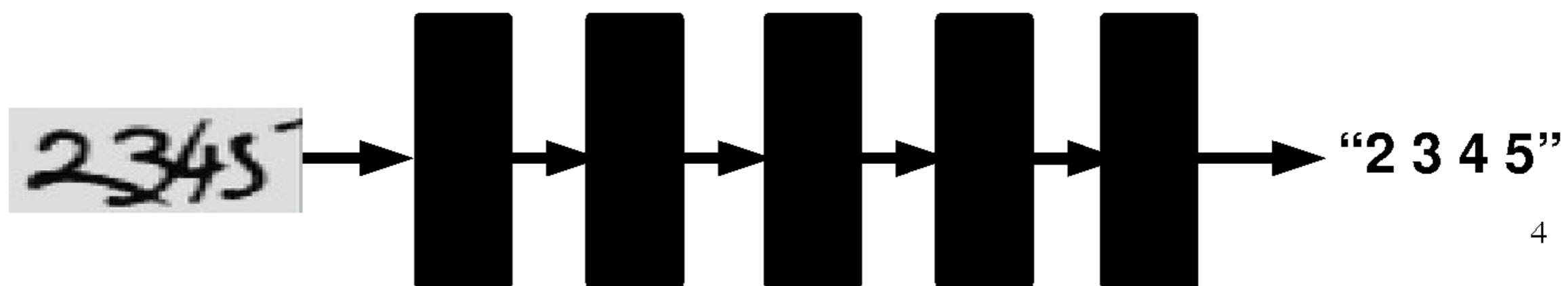
Classification



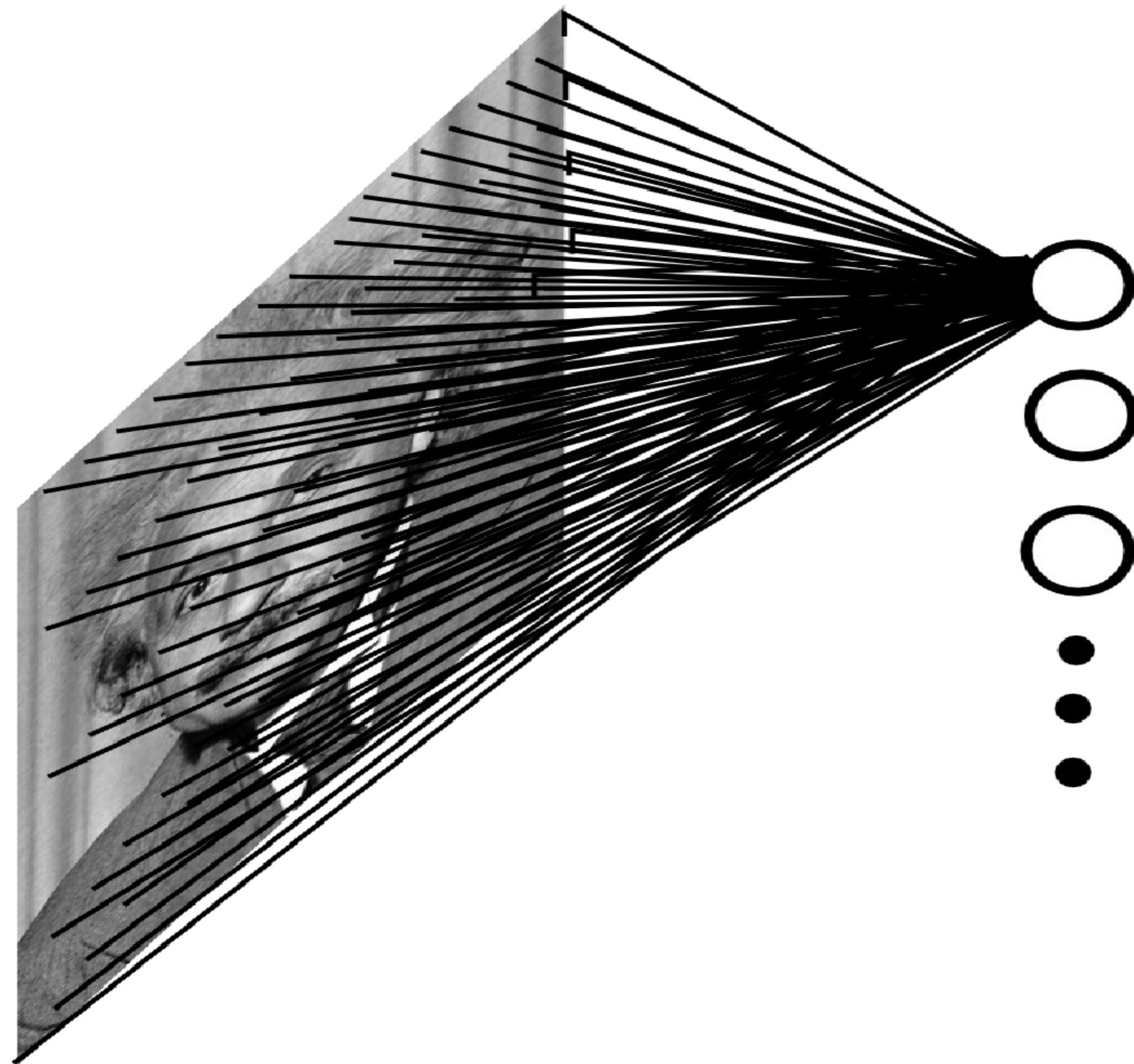
Denoising



OCR

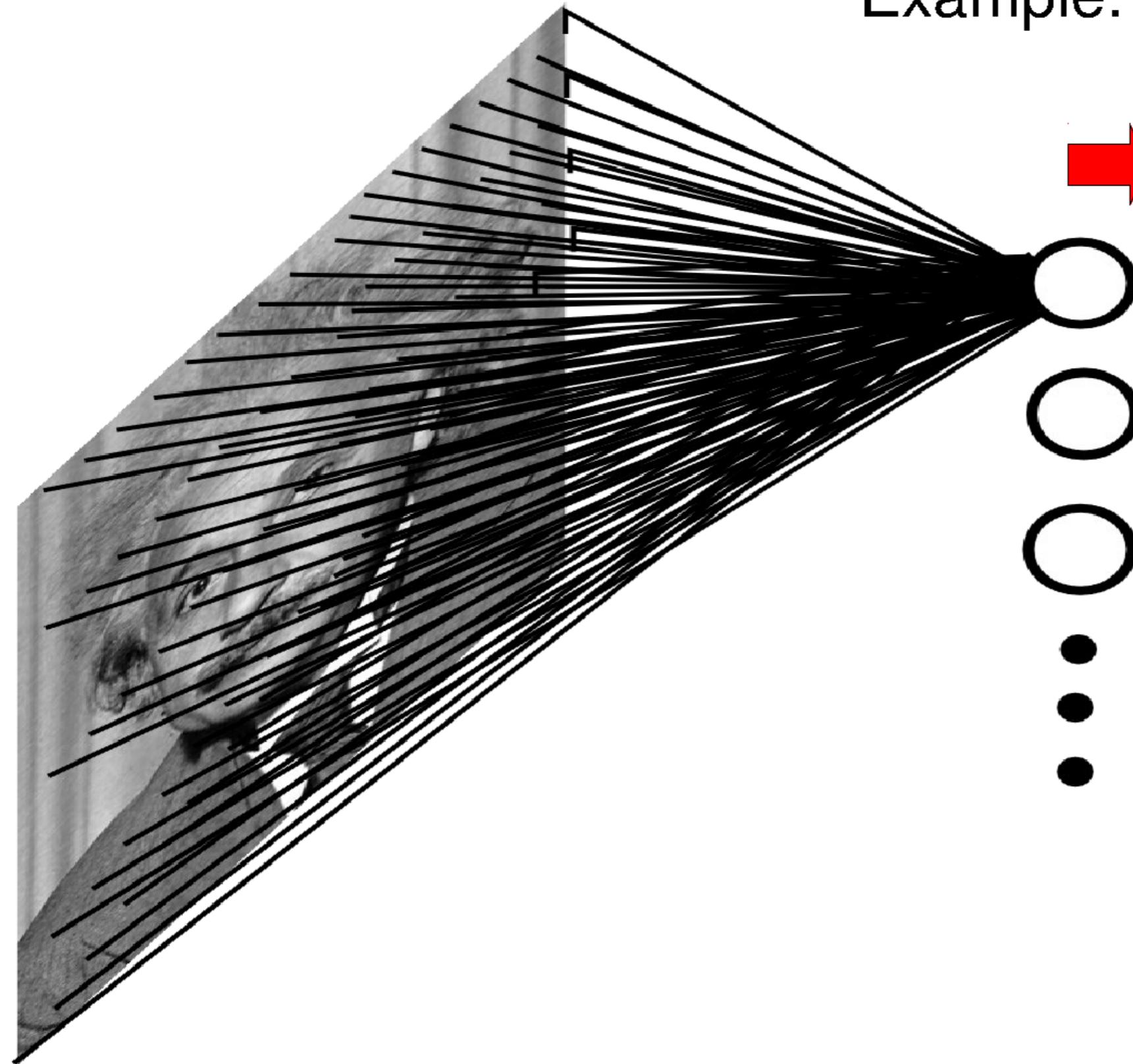


Images as input to neural networks



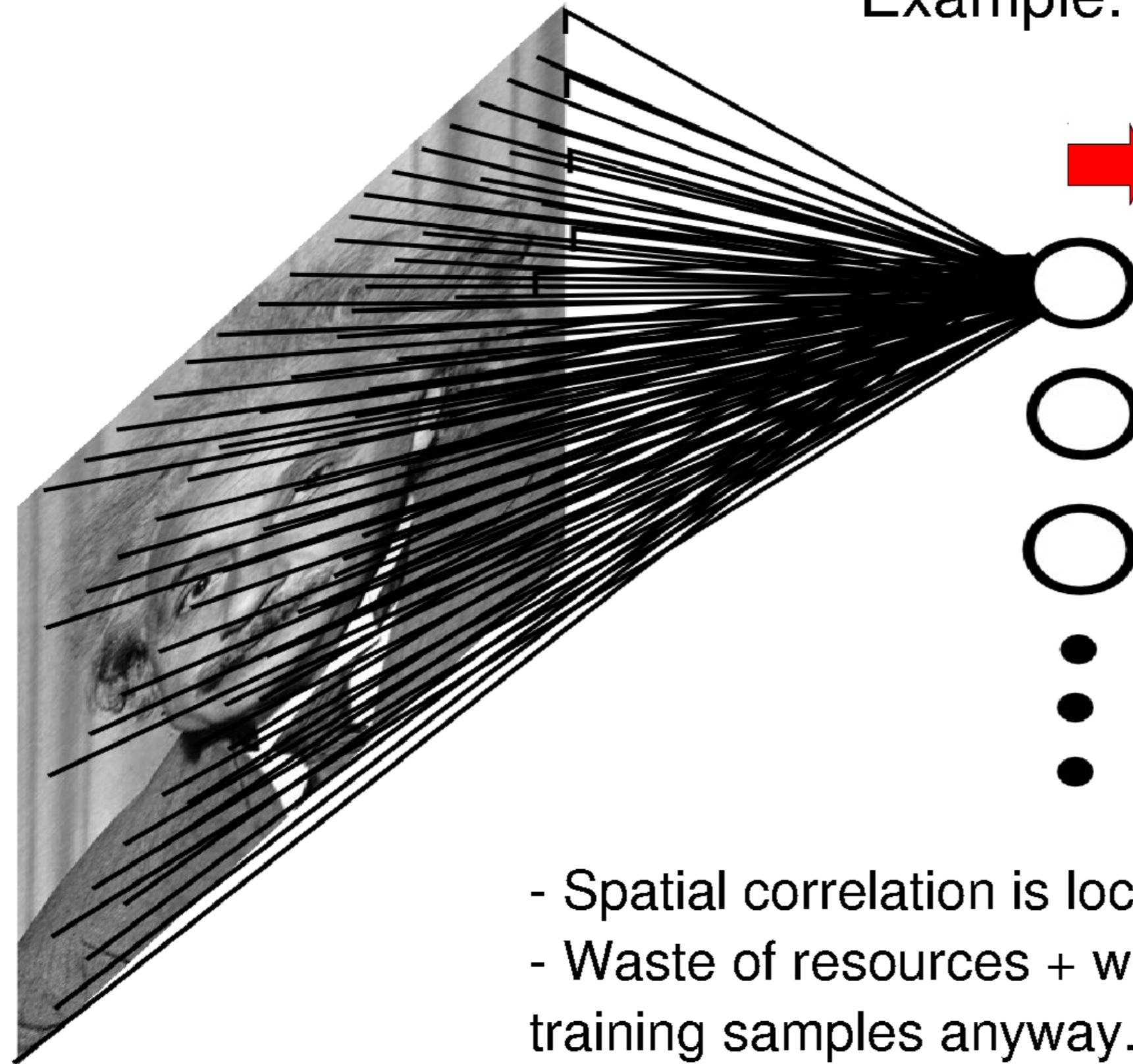
Images as input to neural networks

Example: 200x200 image
40K hidden units
~2B parameters!!!



Images as input to neural networks

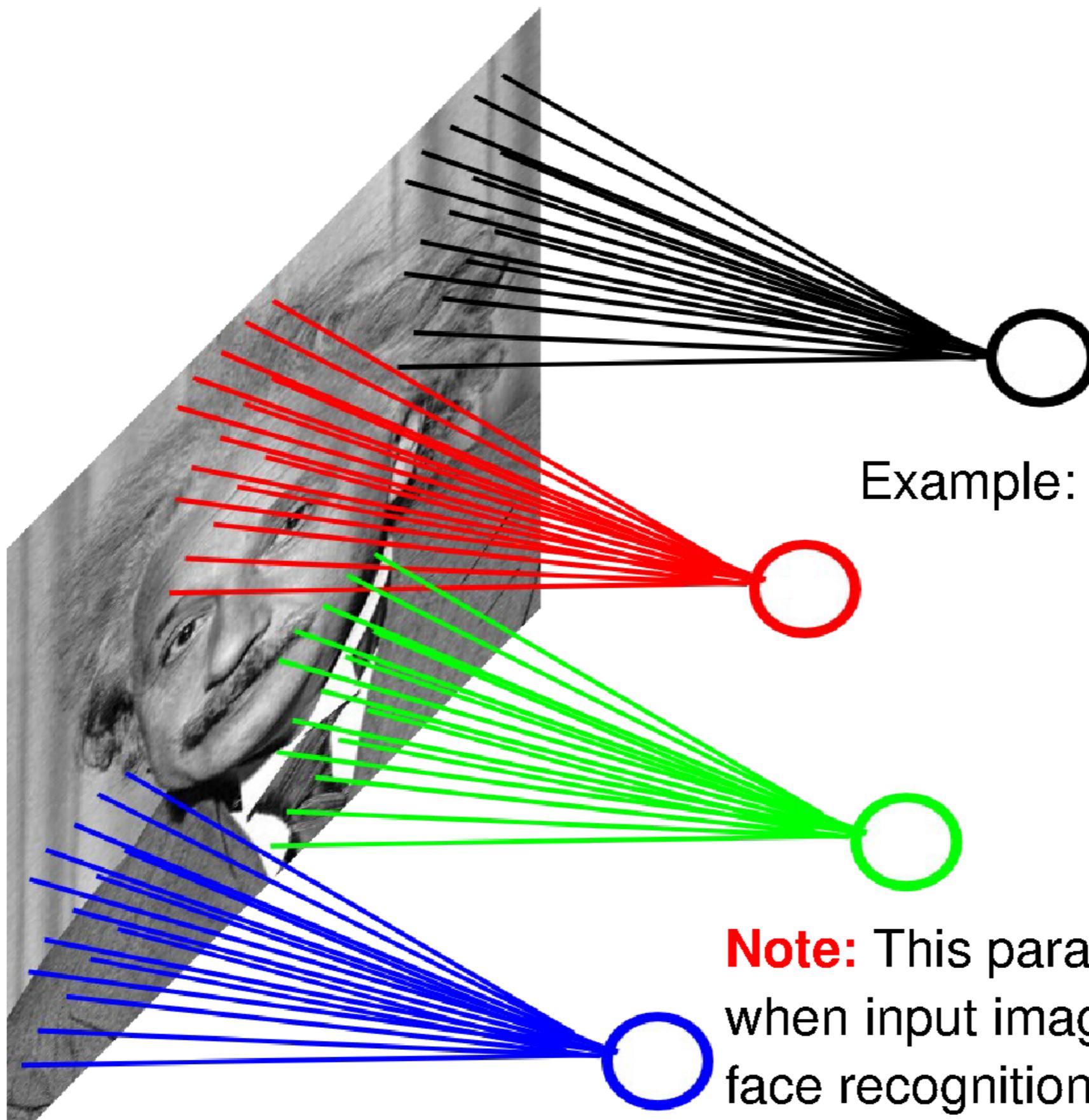
Example: 200x200 image
40K hidden units
~2B parameters!!!



- Spatial correlation is local
- Waste of resources + we have not enough training samples anyway..

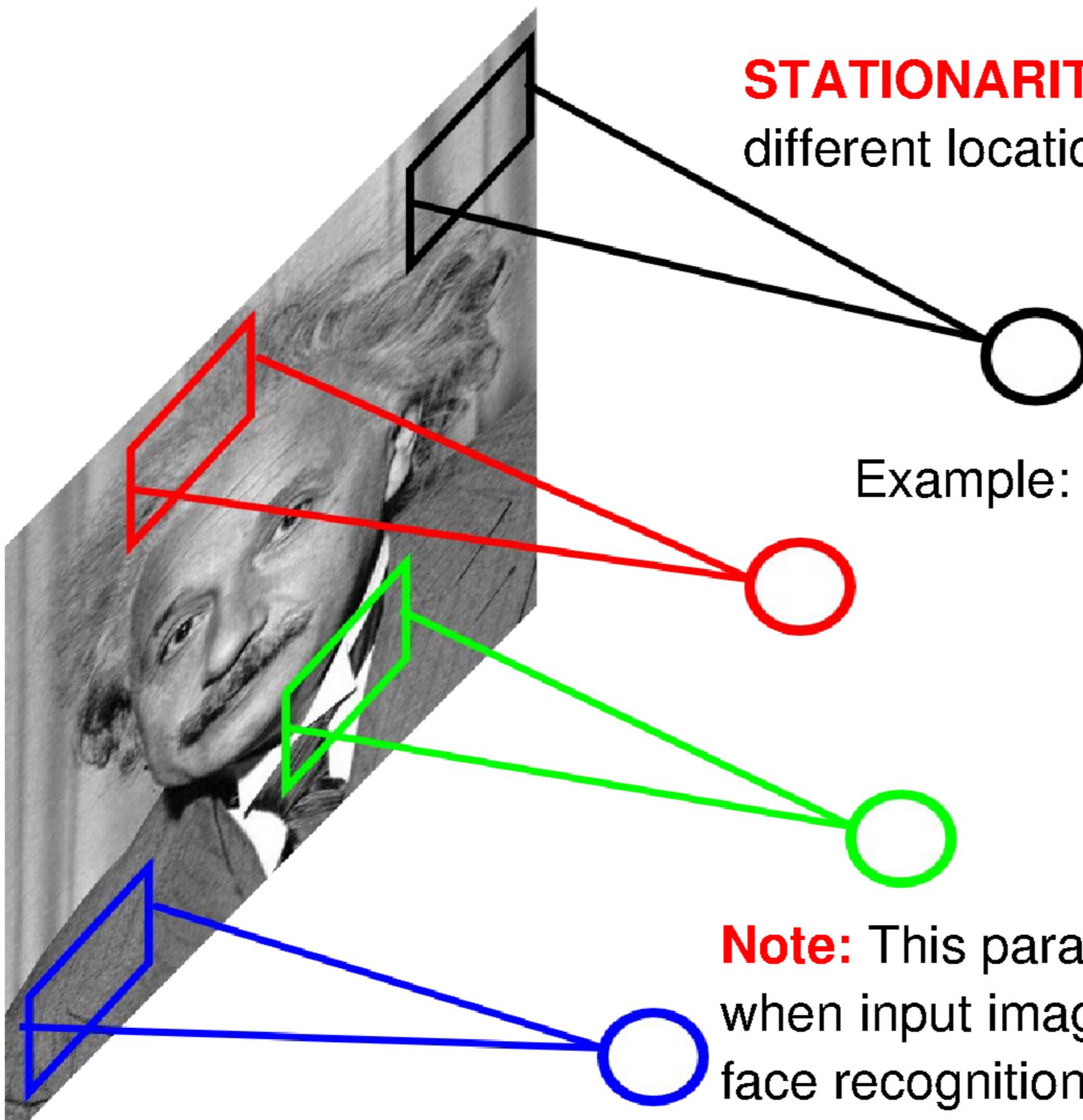
Motivation

- Sparse interactions – receptive fields
 - Assume that in an image, we care about ‘local neighborhoods’ only for a given neural network layer.
 - Composition of layers will expand local -> global.



Example: 200x200 image
40K hidden units
Filter size: 10x10
4M parameters

Note: This parameterization is good
when input image is registered (e.g.,
face recognition).



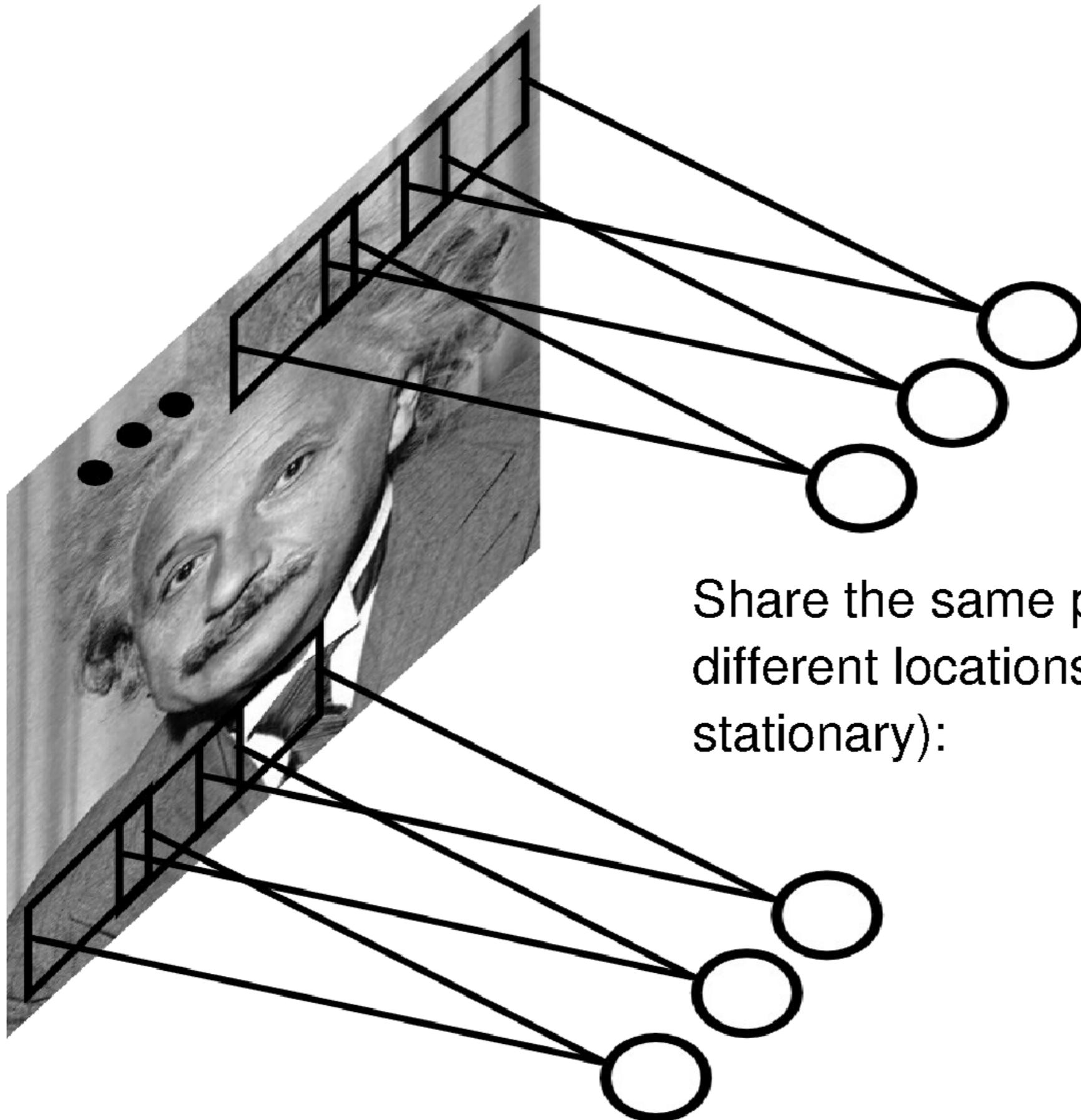
STATIONARITY? Statistics is similar at different locations

Example: 200x200 image
40K hidden units
Filter size: 10x10
4M parameters

Note: This parameterization is good when input image is registered (e.g., face recognition).

Motivation

- Sparse interactions – receptive fields
 - Assume that in an image, we care about ‘local neighborhoods’ only for a given neural network layer.
 - Composition of layers will expand local -> global.
- Parameter sharing
 - ‘Tied weights’ – use same weights for more than one perceptron in the neural network.
 - Leads to equivariant representation
 - If input changes (e.g., translates), then output changes similarly



Share the same parameters across
different locations (assuming input is
stationary):

Filtering remainder: Correlation (rotated convolution)

$$f[\cdot, \cdot] = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$I[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

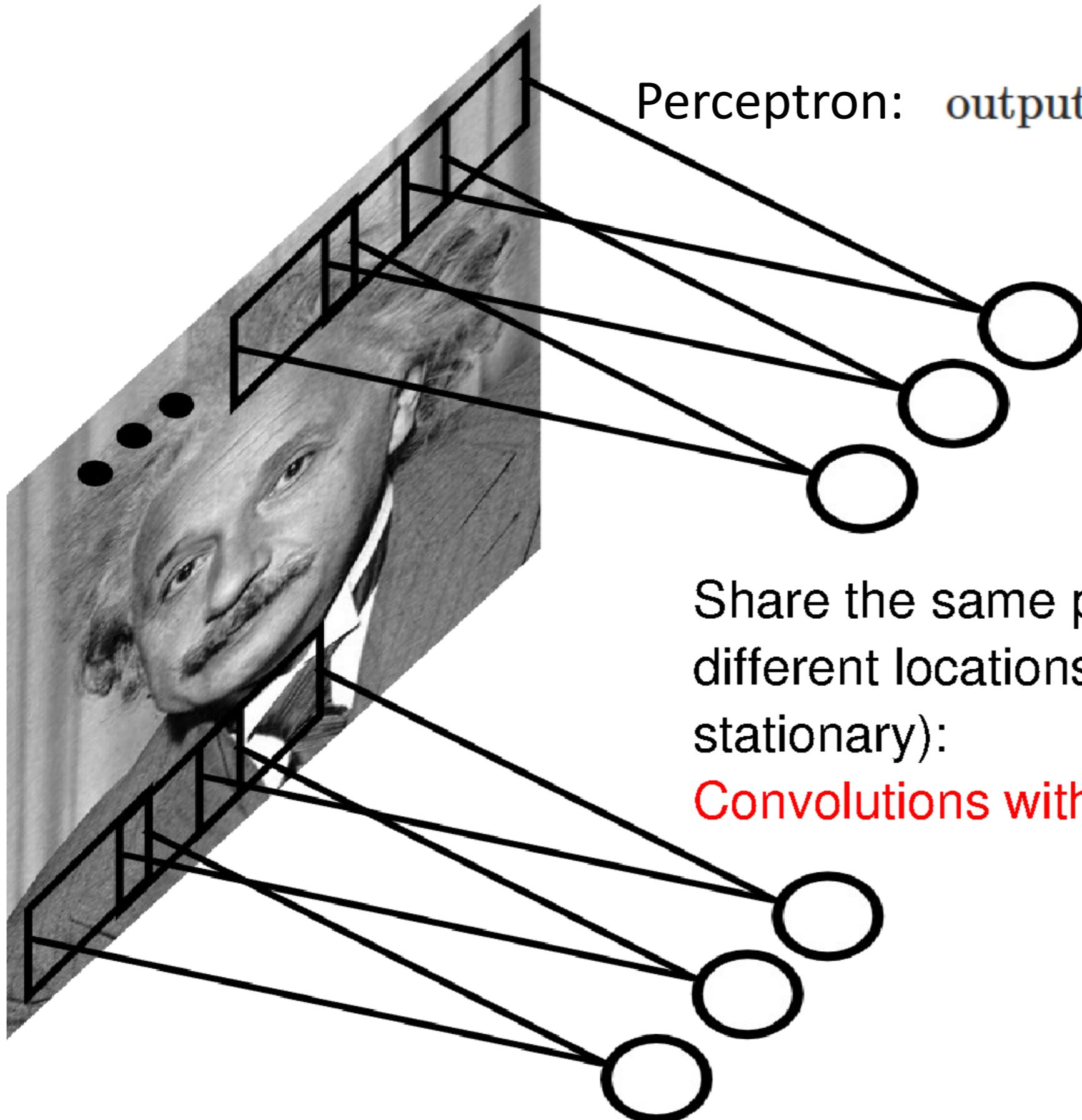
$h[.,.]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$h[m, n] = \sum_{k,l} f[k, l] I[m + k, n + l]$$

Credit: S. Seitz

Convolutional Layer



Perceptron:

$$\text{output} = \begin{cases} 0 & \text{if } w \cdot x + b \leq 0 \\ 1 & \text{if } w \cdot x + b > 0 \end{cases}$$

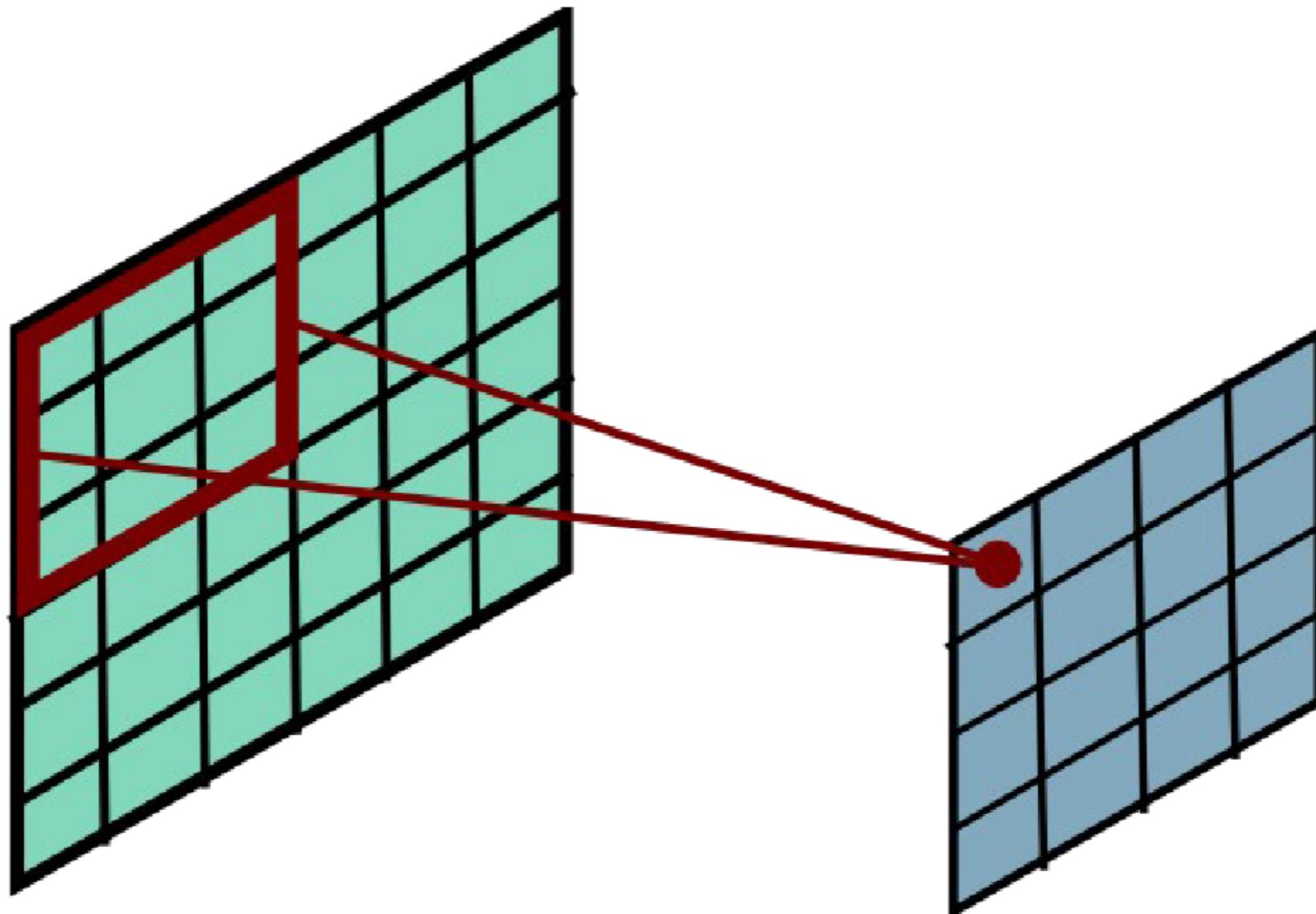
$$w \cdot x \equiv \sum_j w_j x_j;$$

This is convolution!

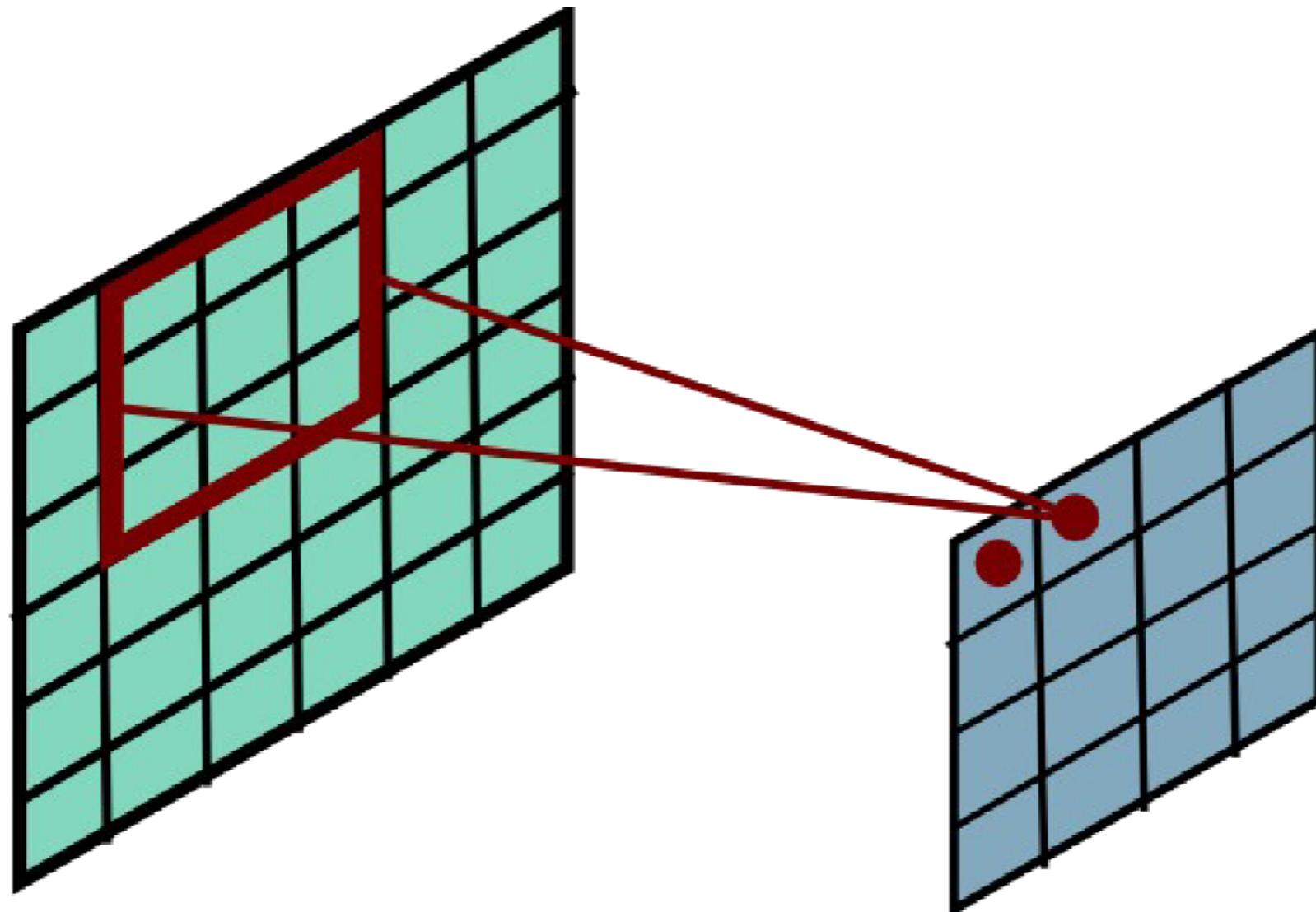
Share the same parameters across different locations (assuming input is stationary):

Convolutions with learned kernels

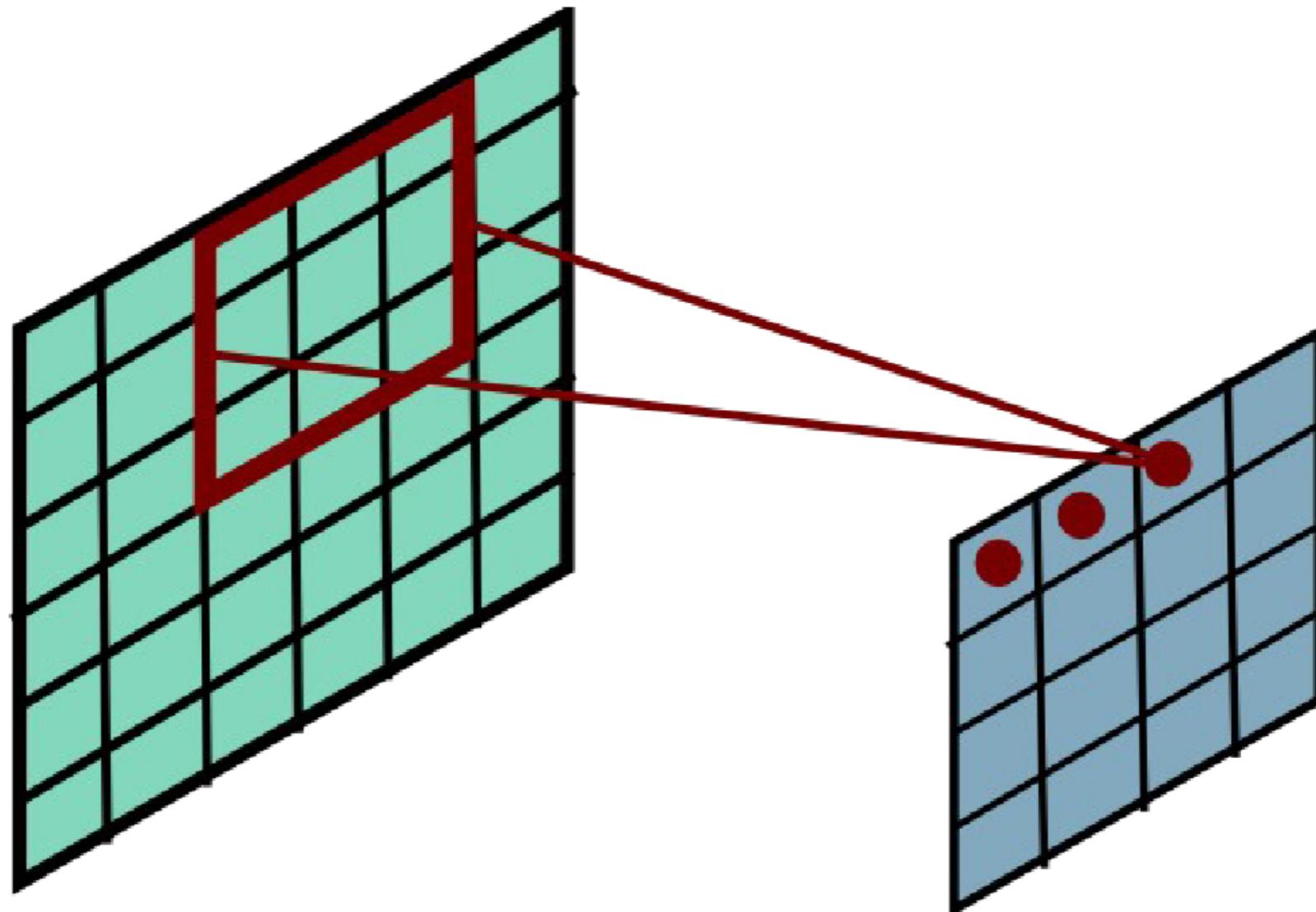
Convolutional Layer



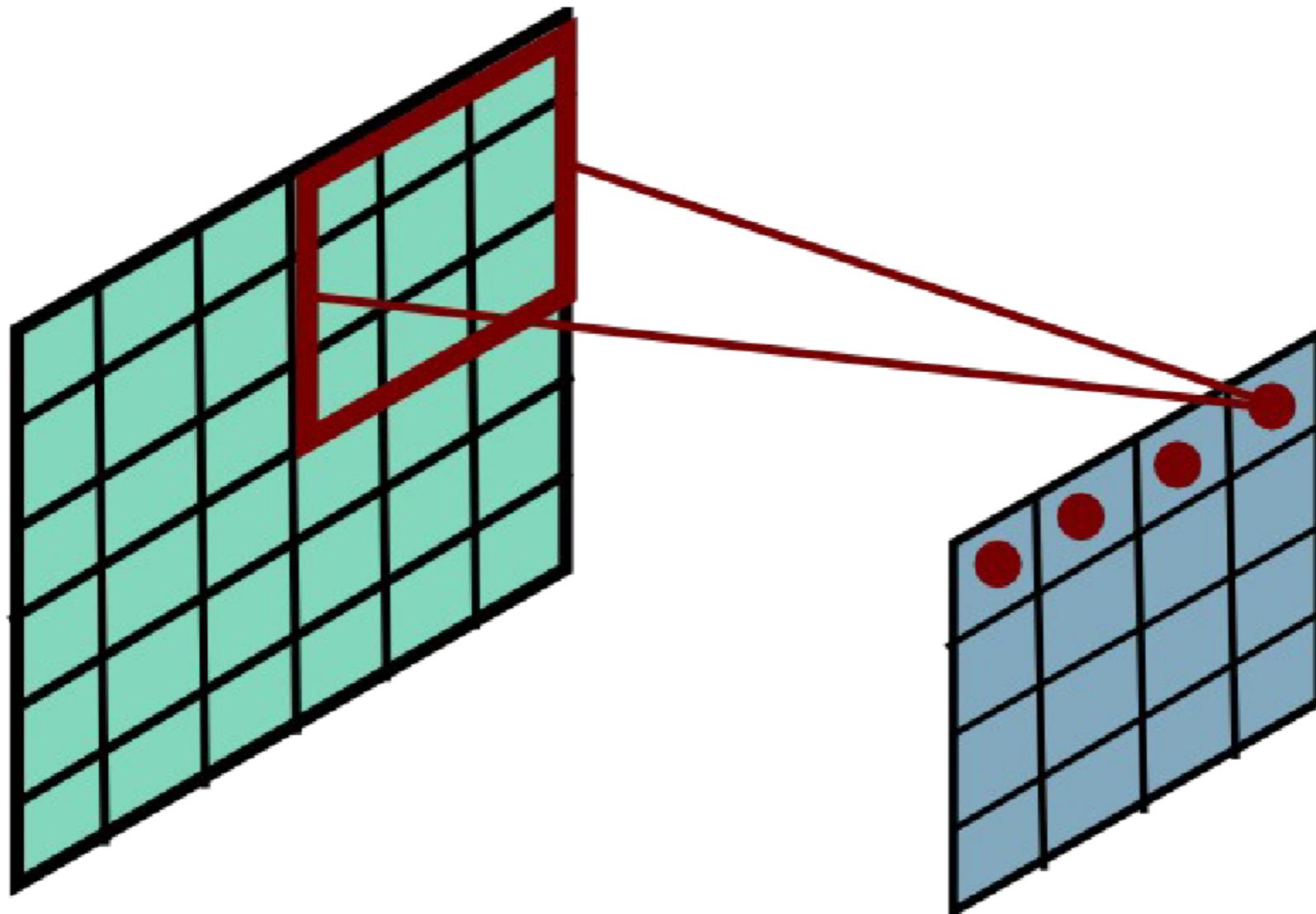
Convolutional Layer



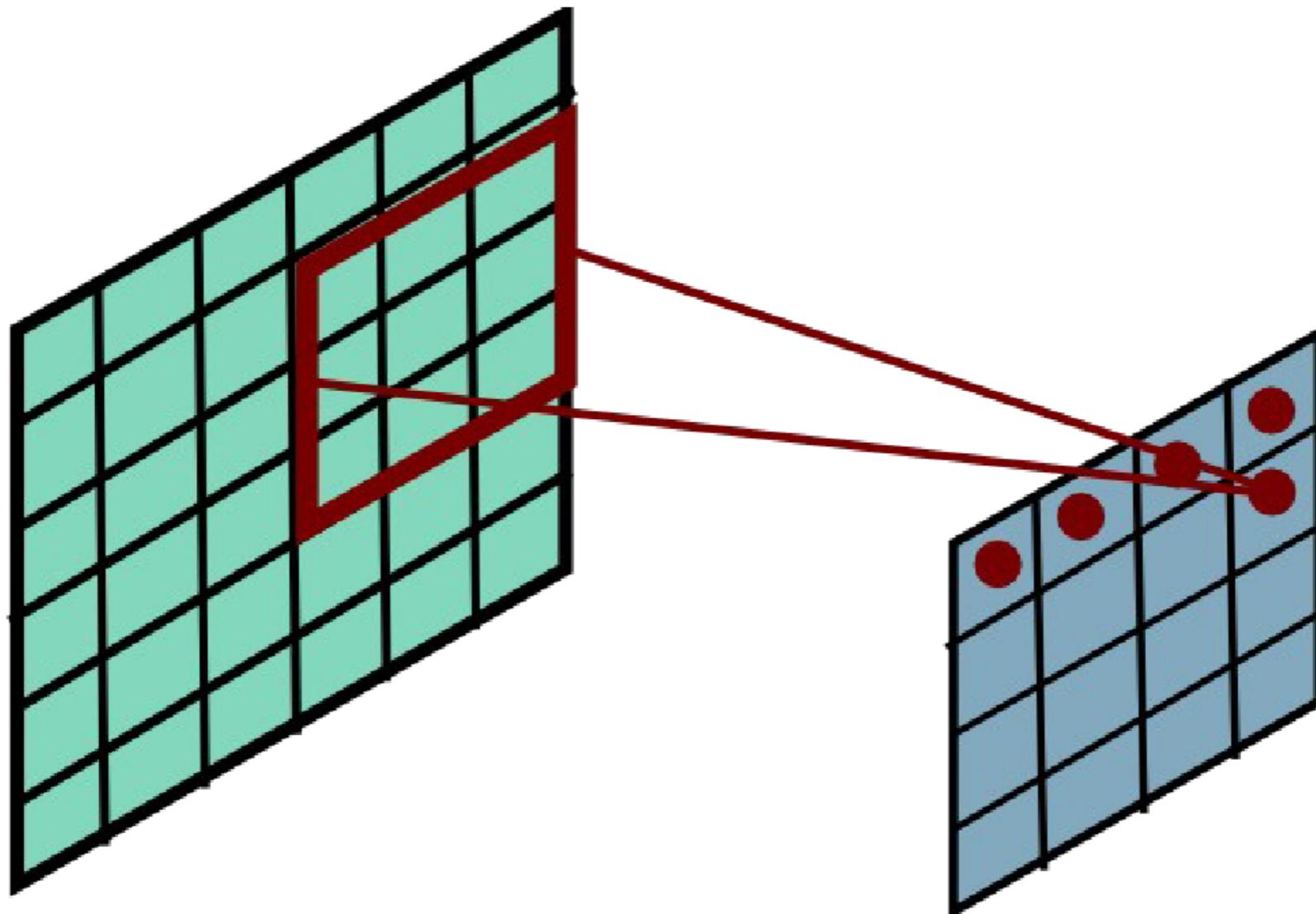
Convolutional Layer



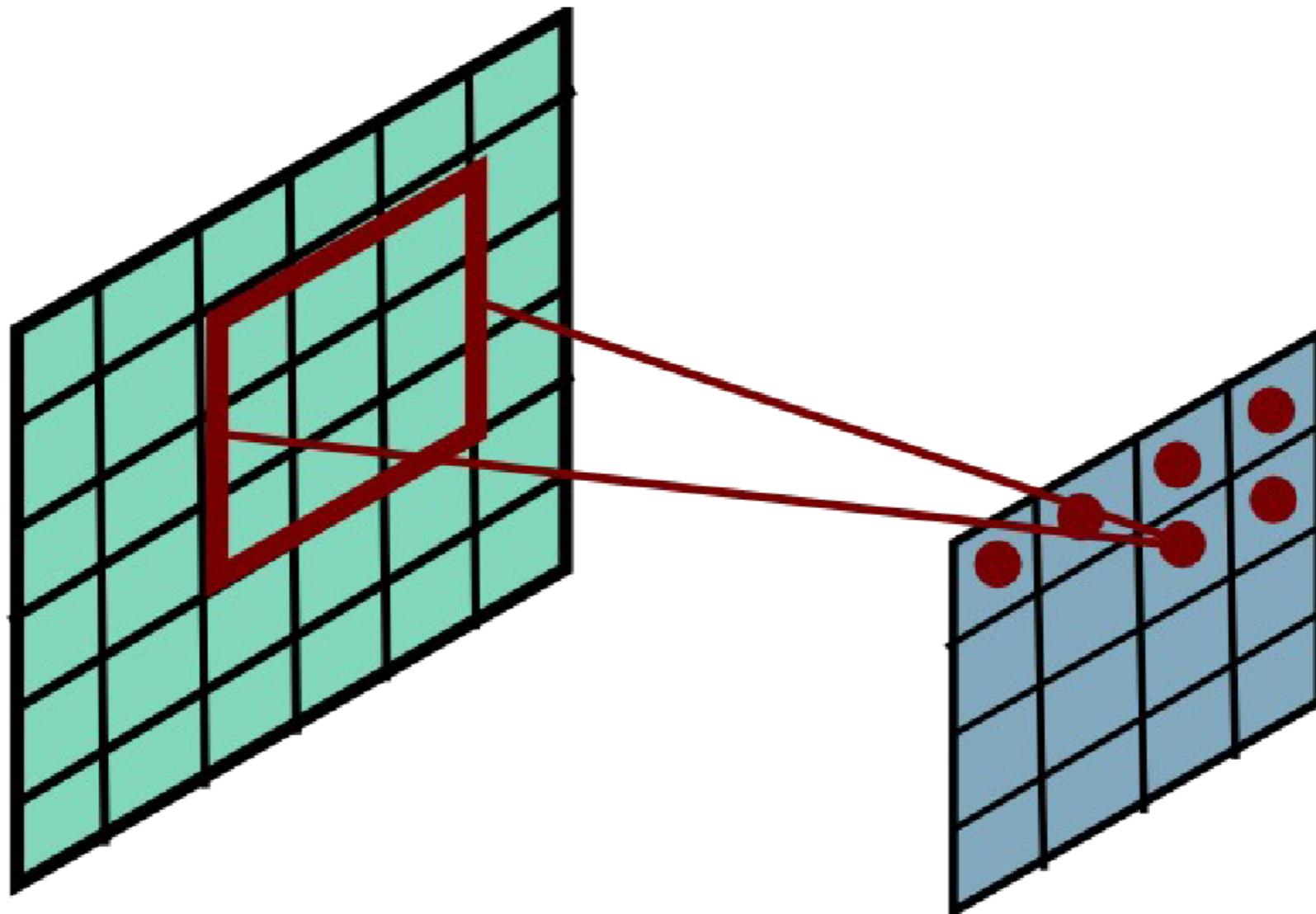
Convolutional Layer



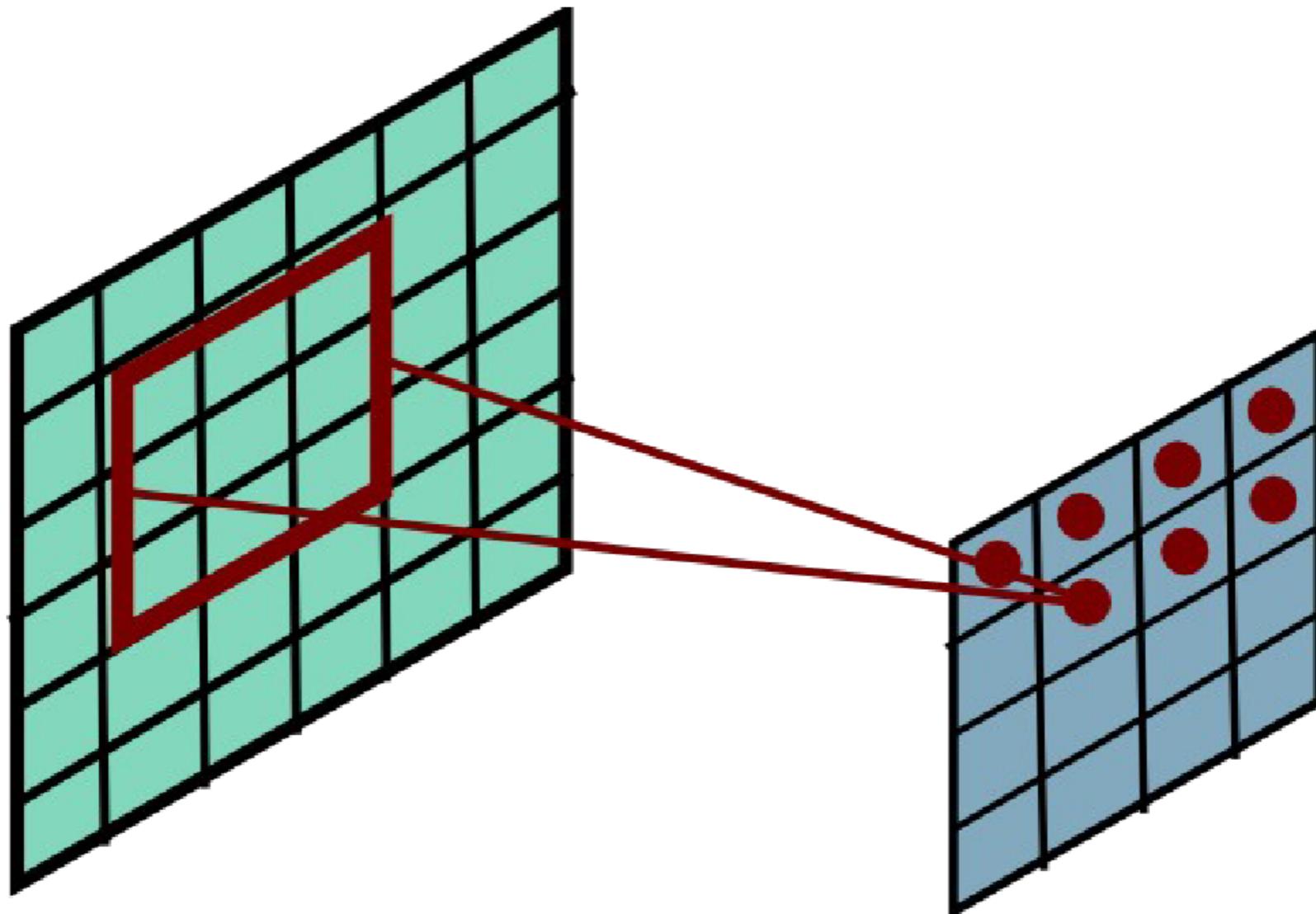
Convolutional Layer



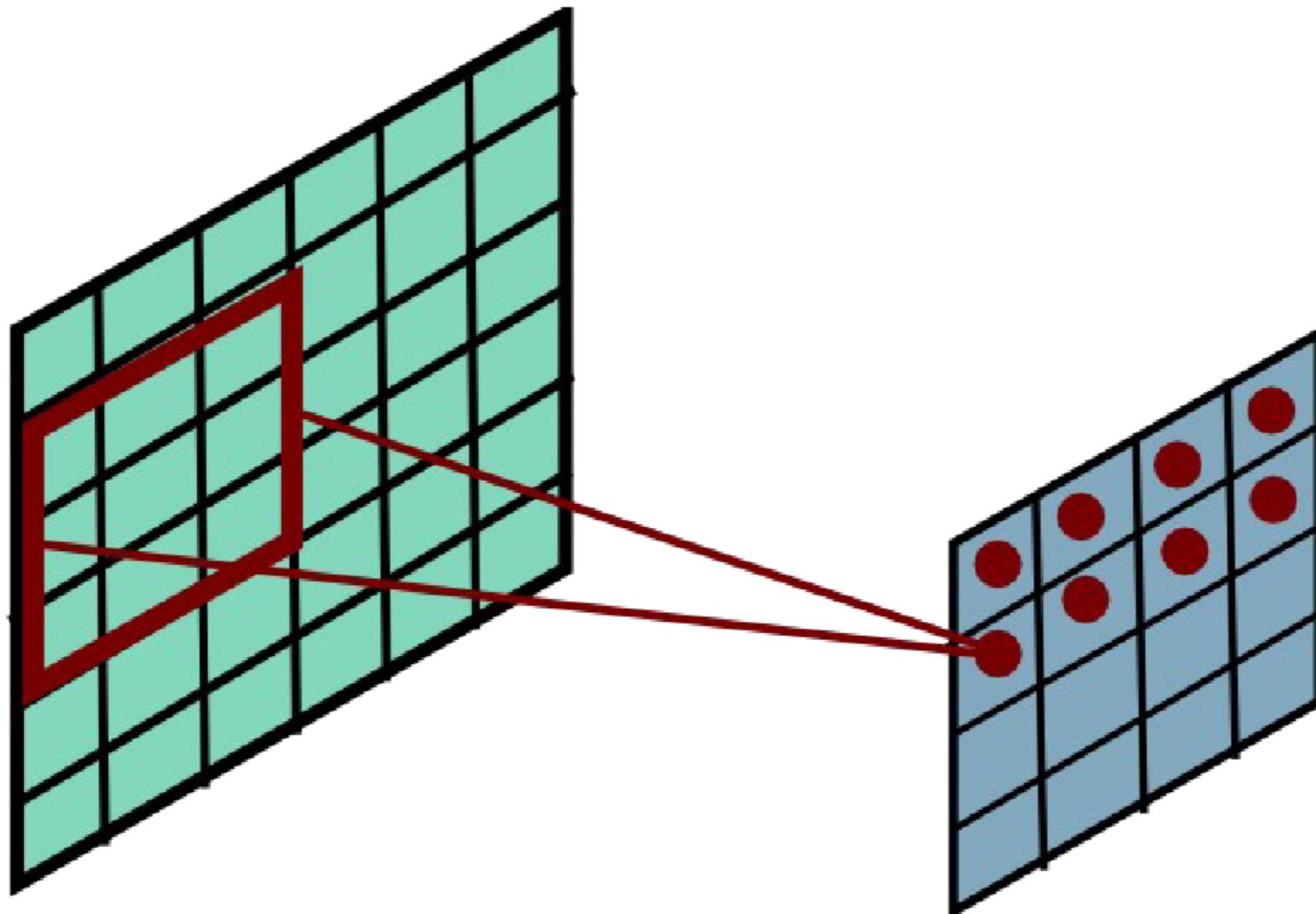
Convolutional Layer



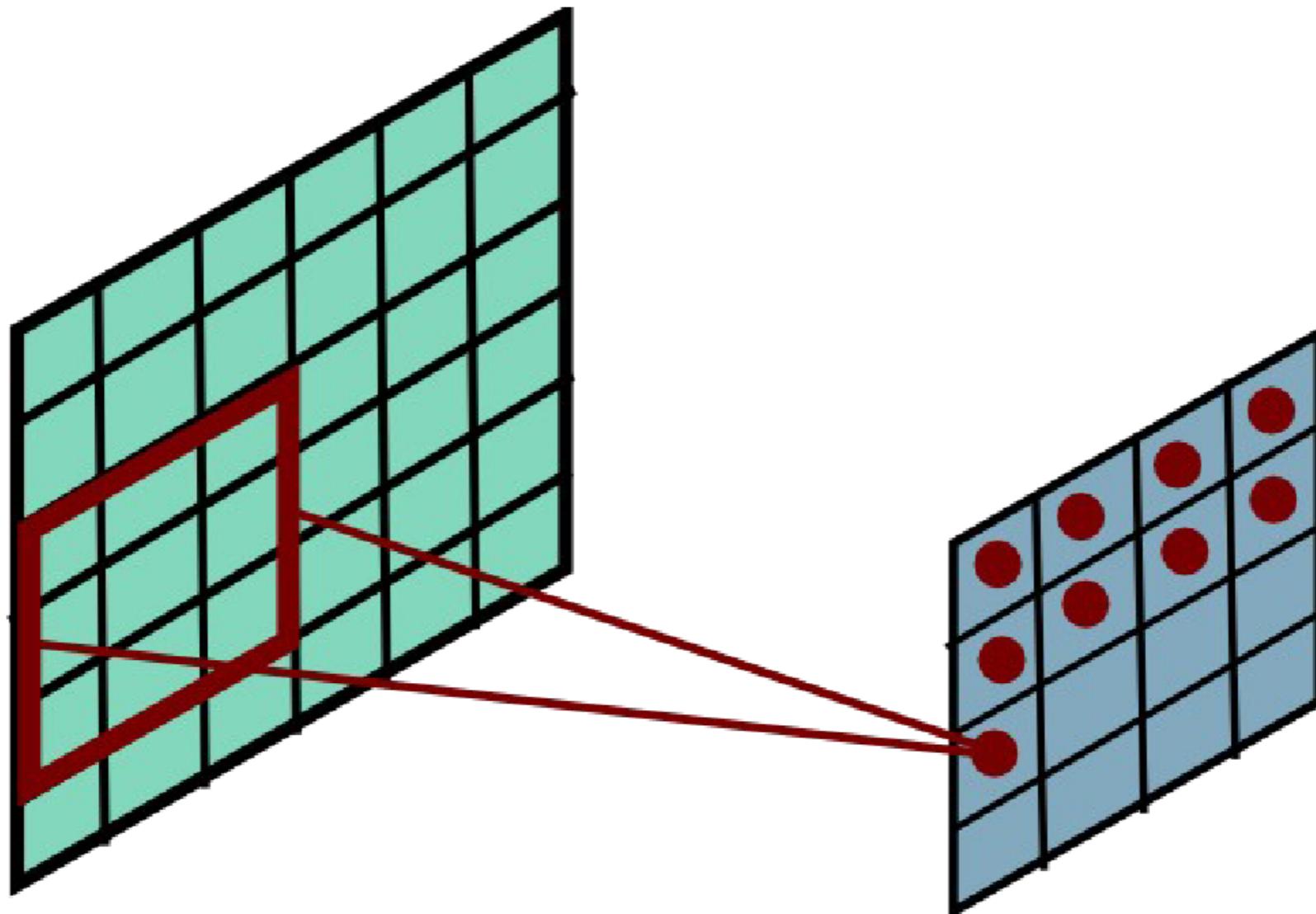
Convolutional Layer



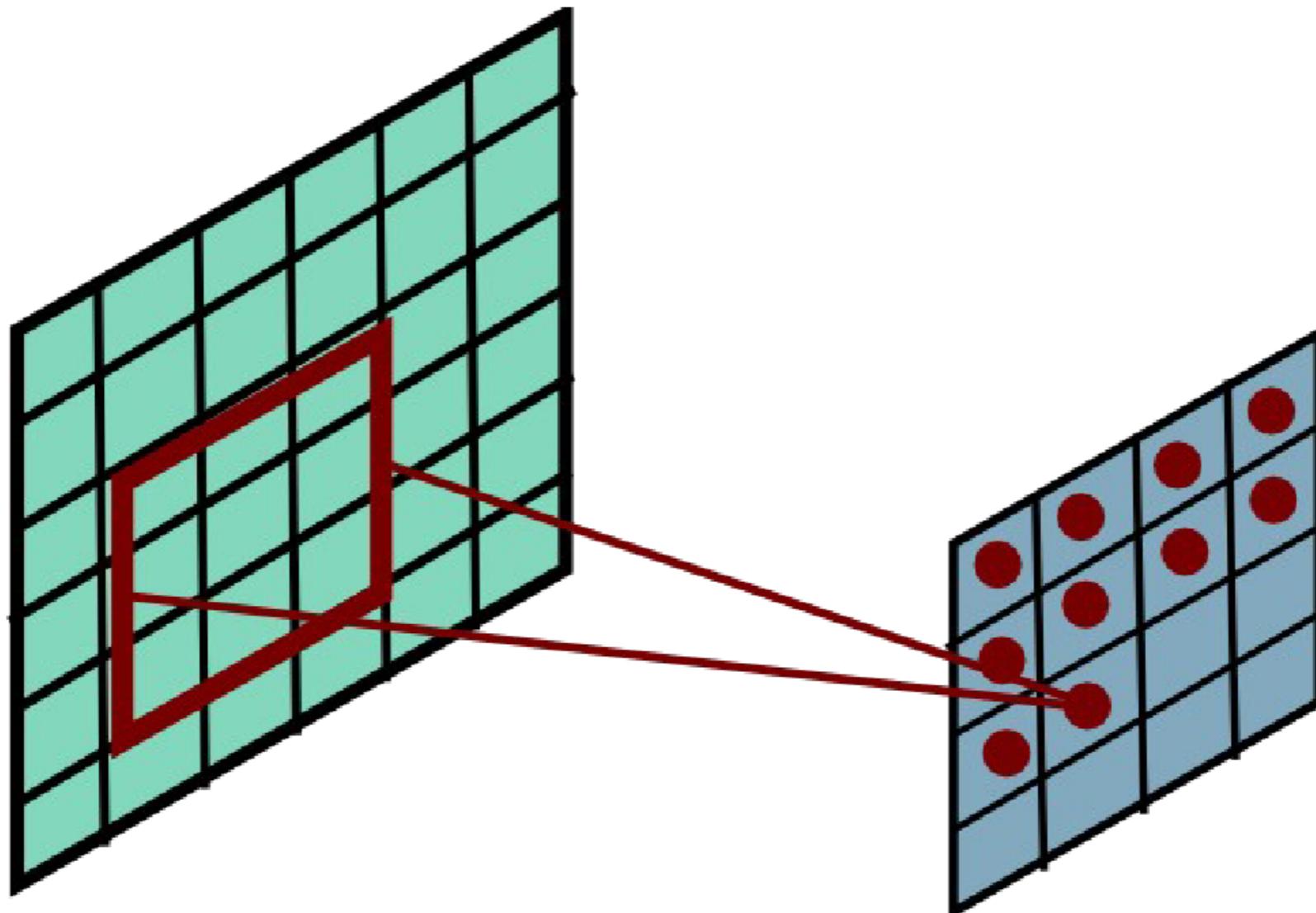
Convolutional Layer



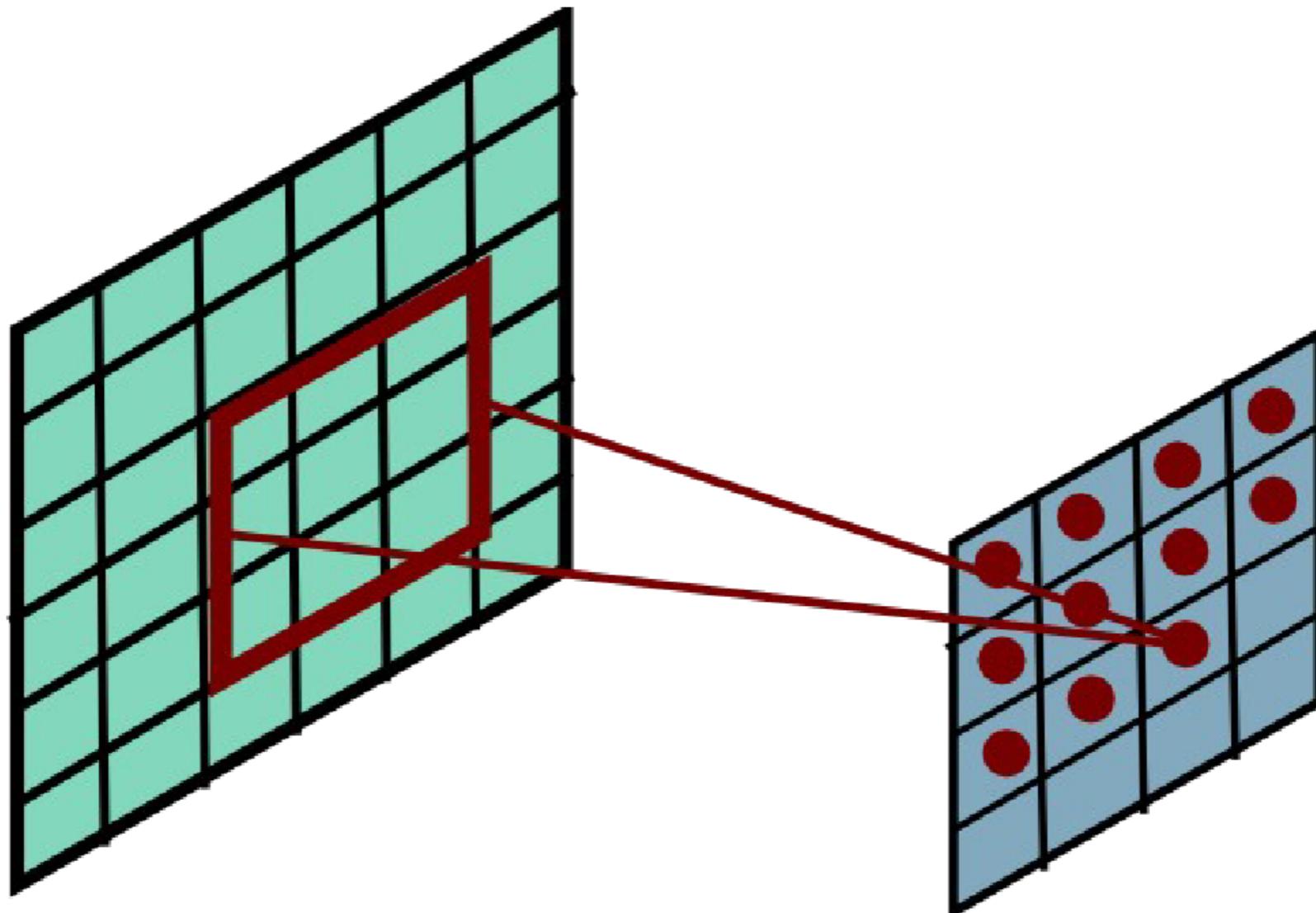
Convolutional Layer



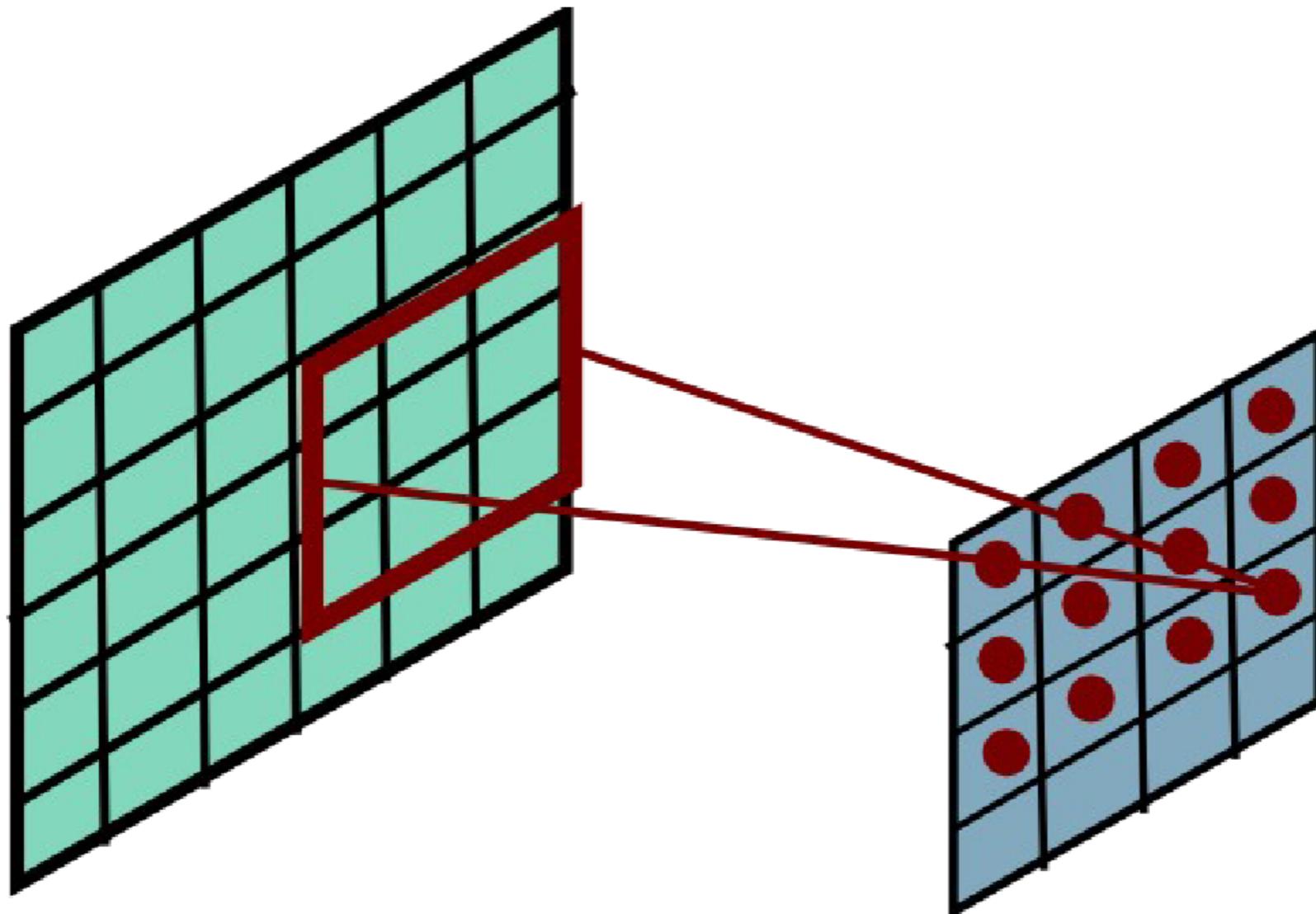
Convolutional Layer



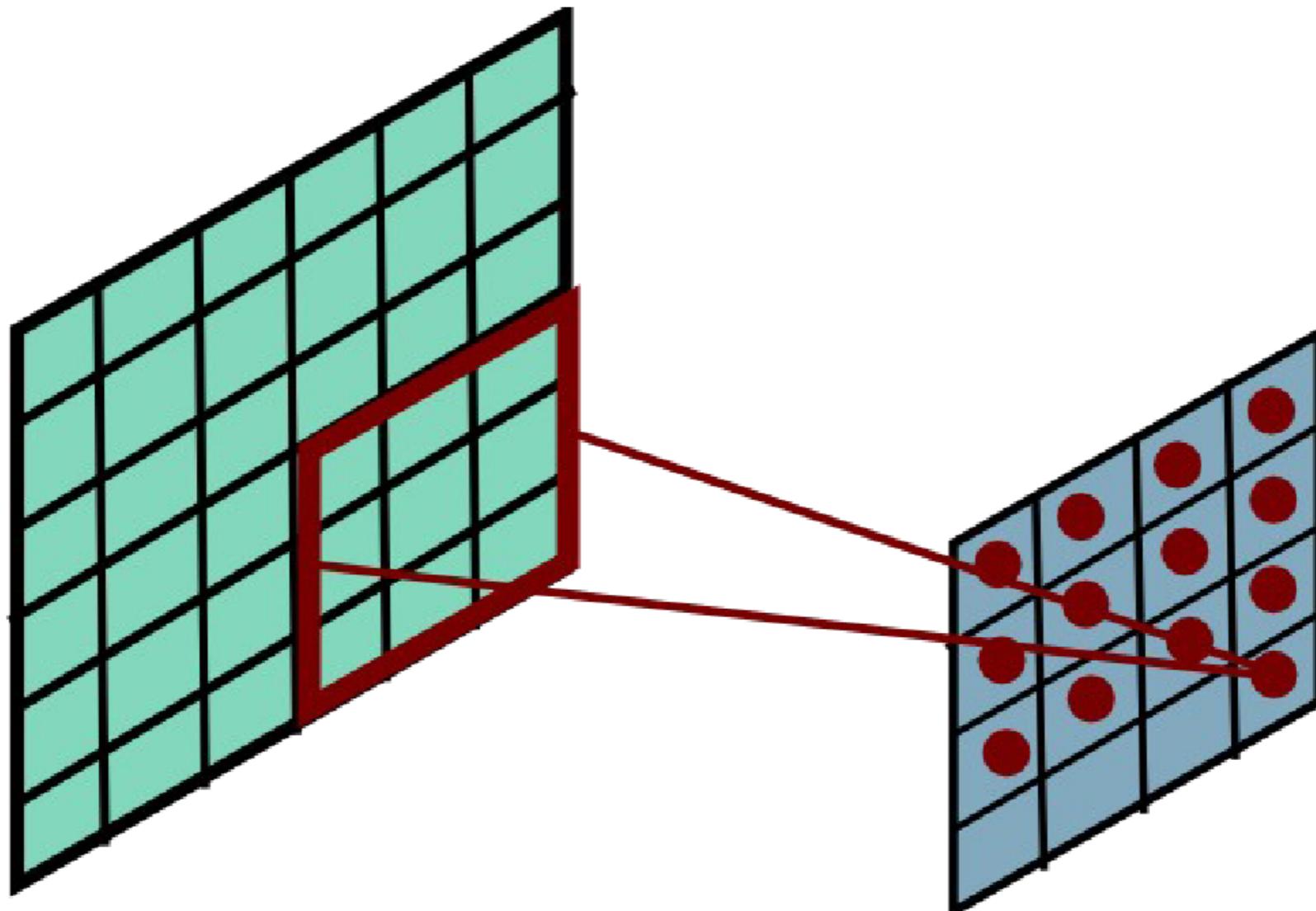
Convolutional Layer



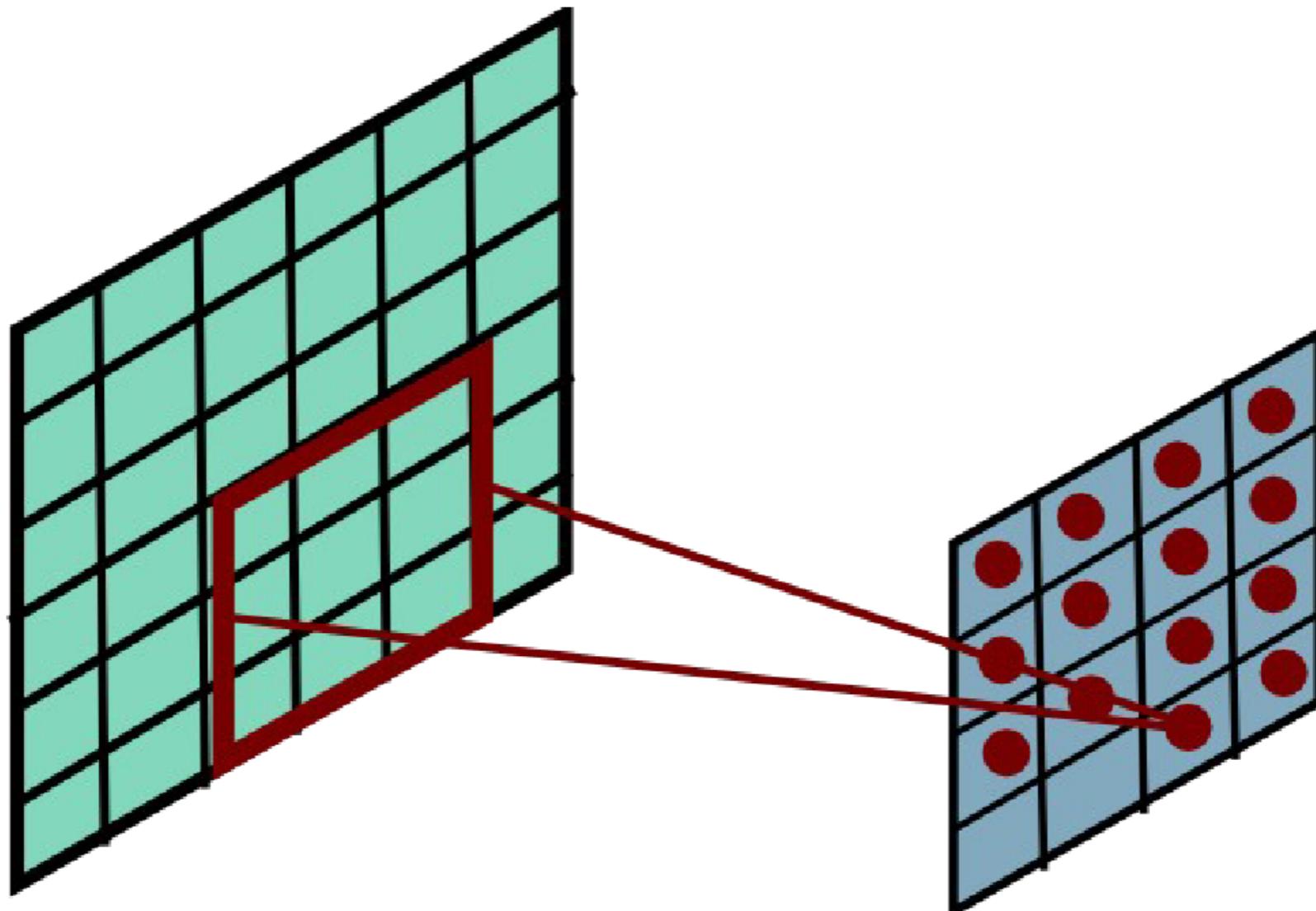
Convolutional Layer



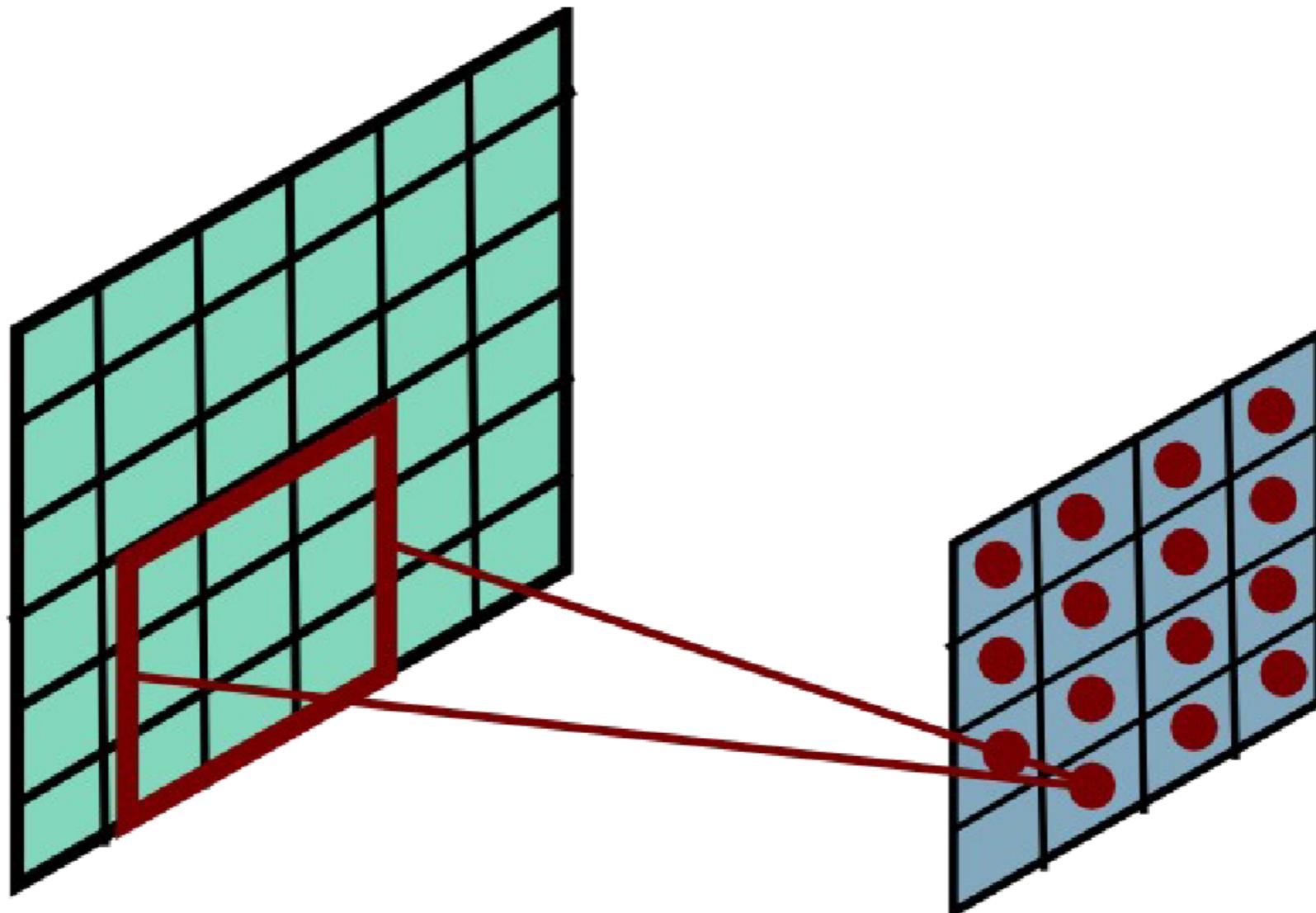
Convolutional Layer



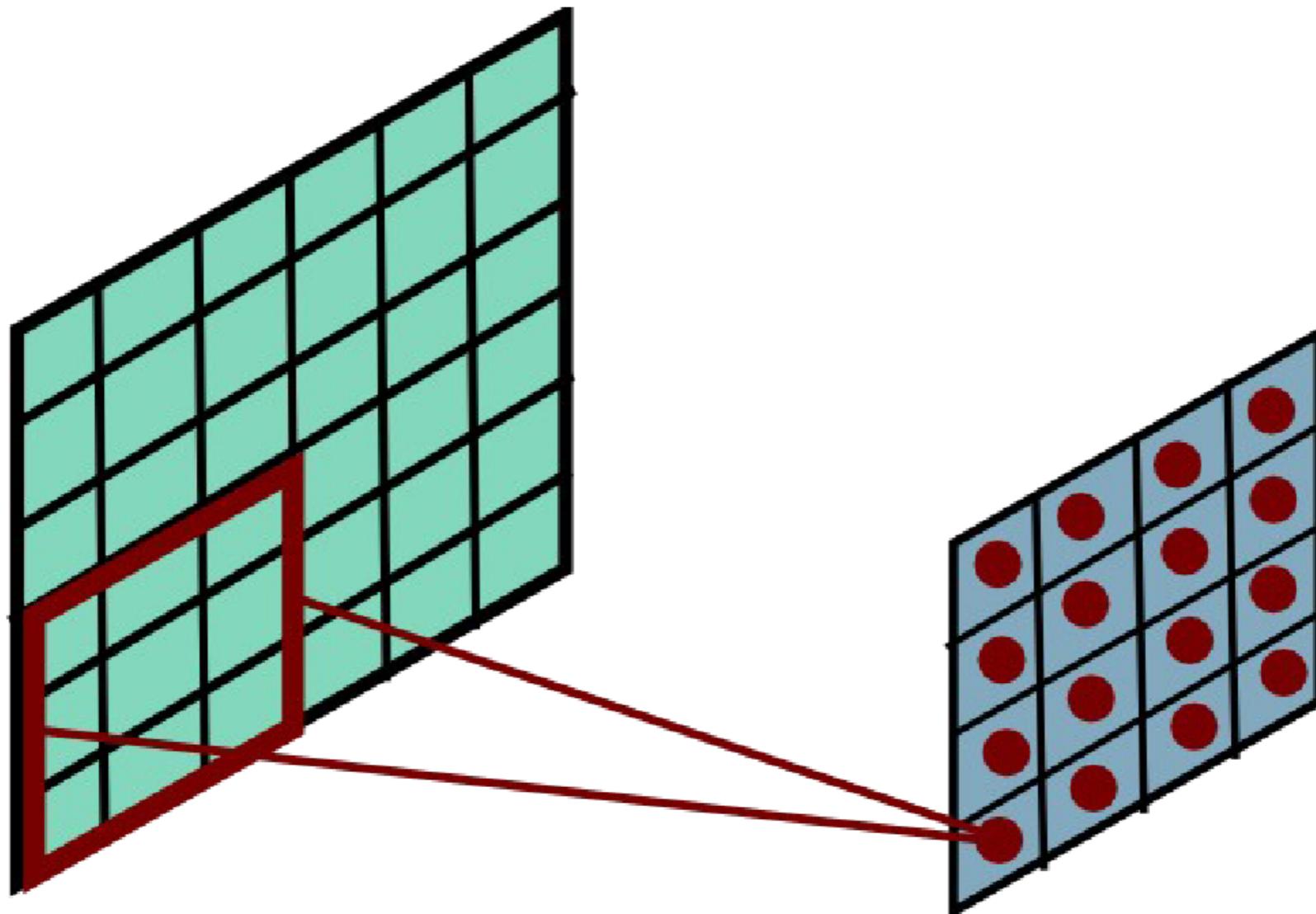
Convolutional Layer



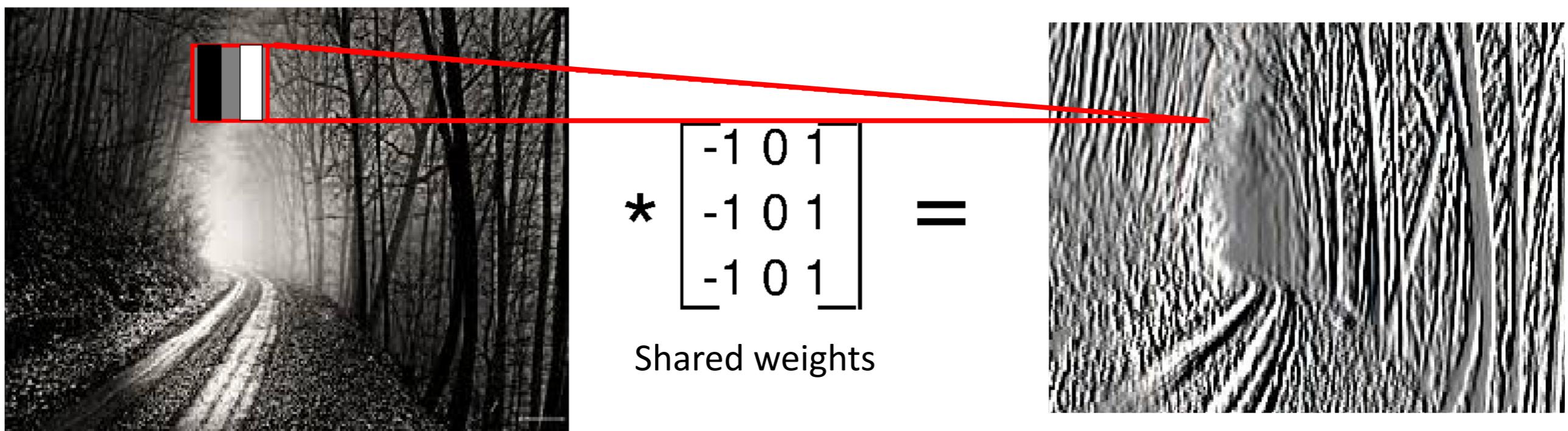
Convolutional Layer



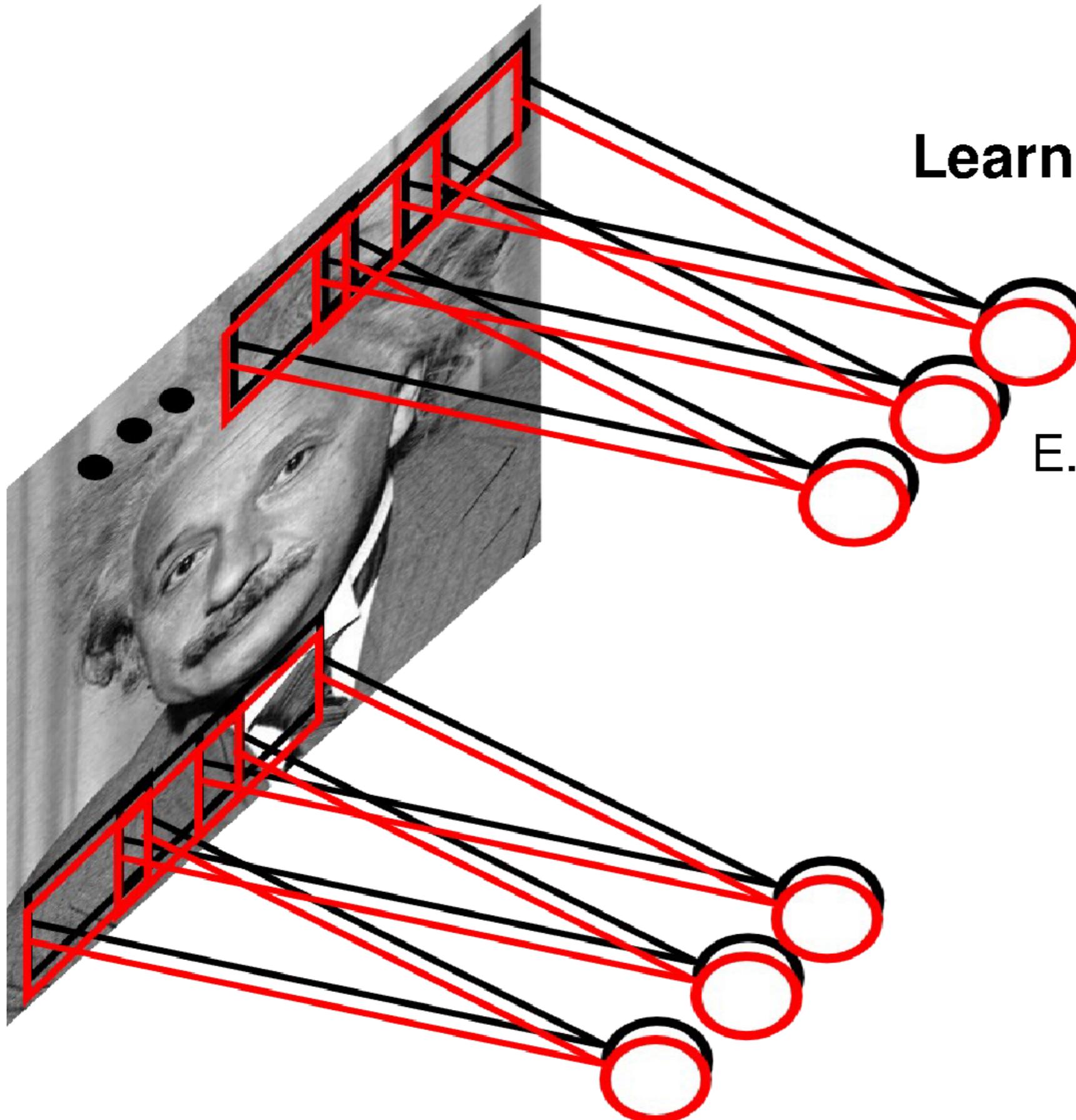
Convolutional Layer



Convolutional Layer



Convolutional Layer

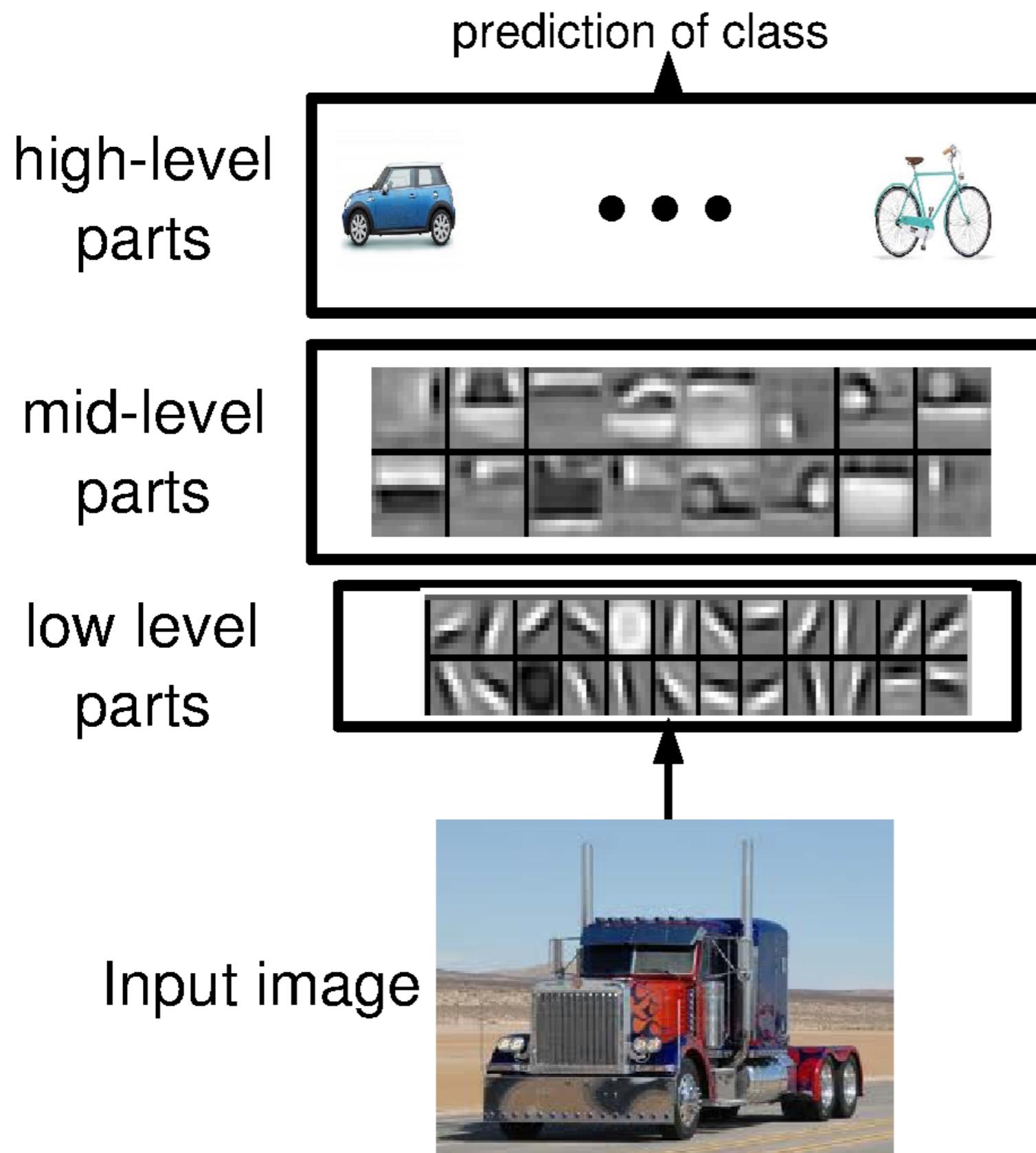


Learn multiple filters.

Filter = 'local' perceptron.
Also called kernel.

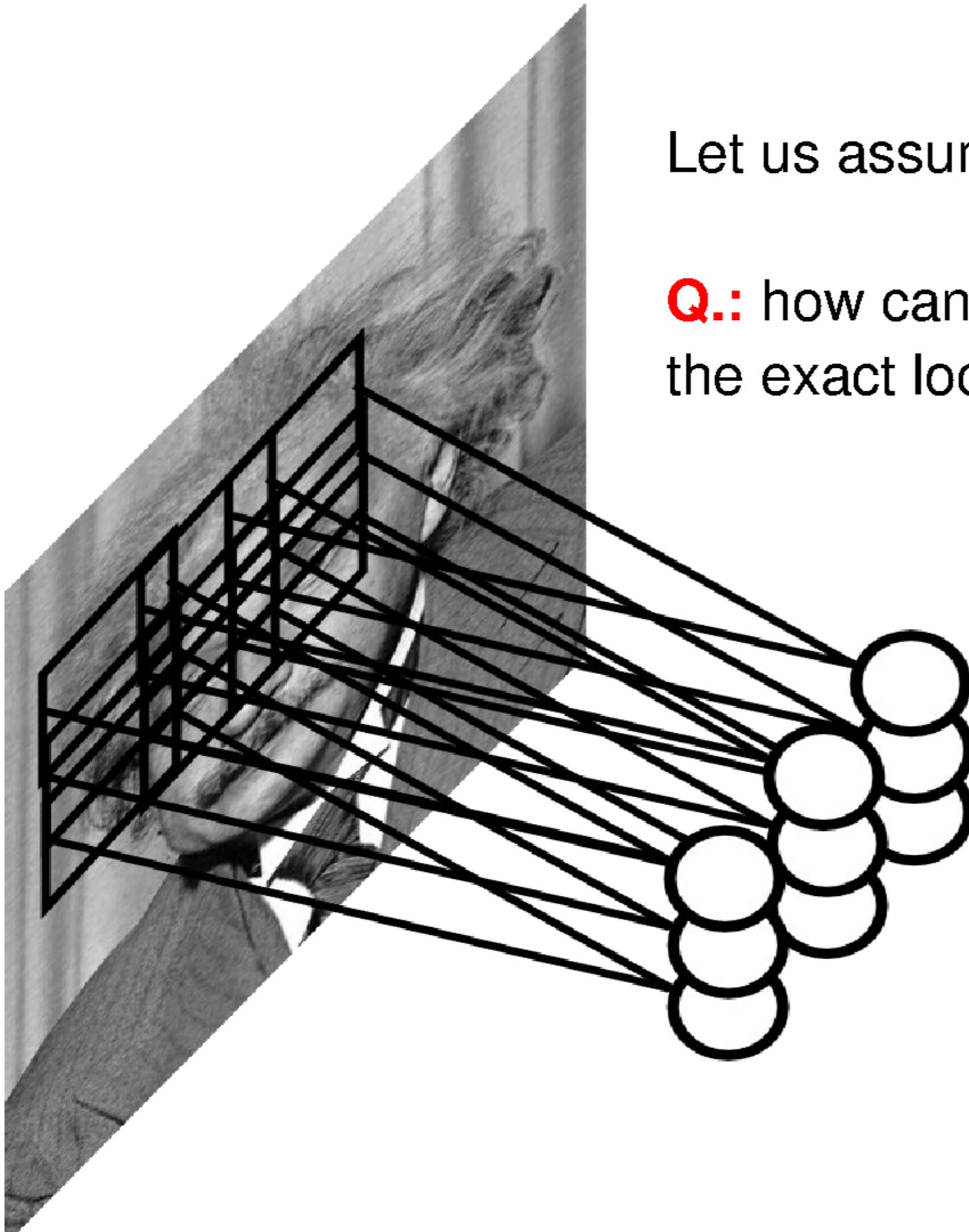
E.g.: 200x200 image
100 Filters
Filter size: 10x10
10K parameters

Interpretation



- distributed representations
- feature sharing
- compositionality

Pooling Layer



Let us assume filter is an “eye” detector.

Q.: how can we make the detection robust to the exact location of the eye?

Pooling Layer: Examples

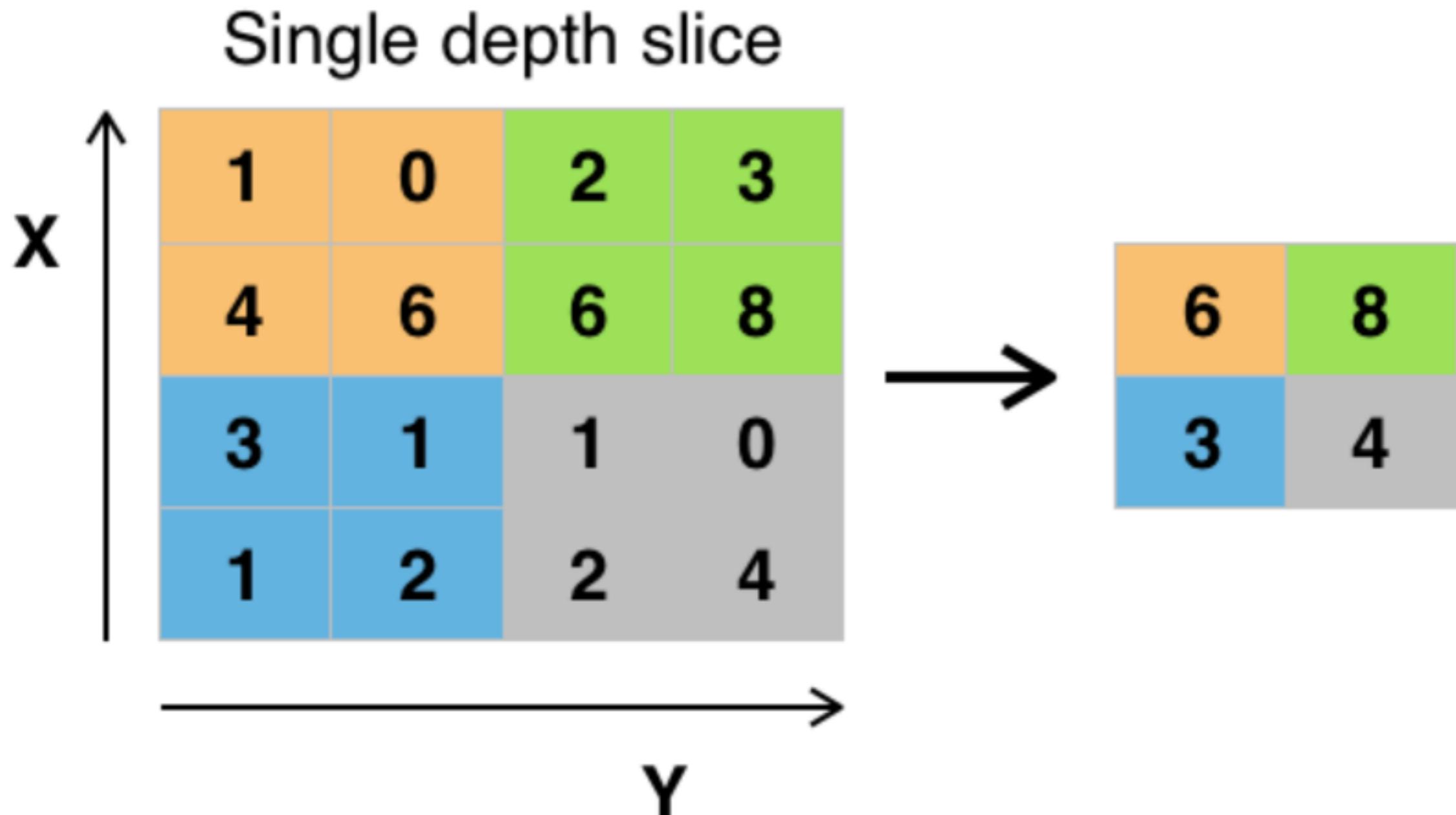
Max-pooling:

$$h_j^n(x, y) = \max_{\bar{x} \in N(x), \bar{y} \in N(y)} h_j^{n-1}(\bar{x}, \bar{y})$$

Average-pooling:

$$h_j^n(x, y) = 1/K \sum_{\bar{x} \in N(x), \bar{y} \in N(y)} h_j^{n-1}(\bar{x}, \bar{y})$$

Max pooling



Pooling Layer: Examples

Max-pooling:

$$h_j^n(x, y) = \max_{\bar{x} \in N(x), \bar{y} \in N(y)} h_j^{n-1}(\bar{x}, \bar{y})$$

Average-pooling:

$$h_j^n(x, y) = 1/K \sum_{\bar{x} \in N(x), \bar{y} \in N(y)} h_j^{n-1}(\bar{x}, \bar{y})$$

L2-pooling:

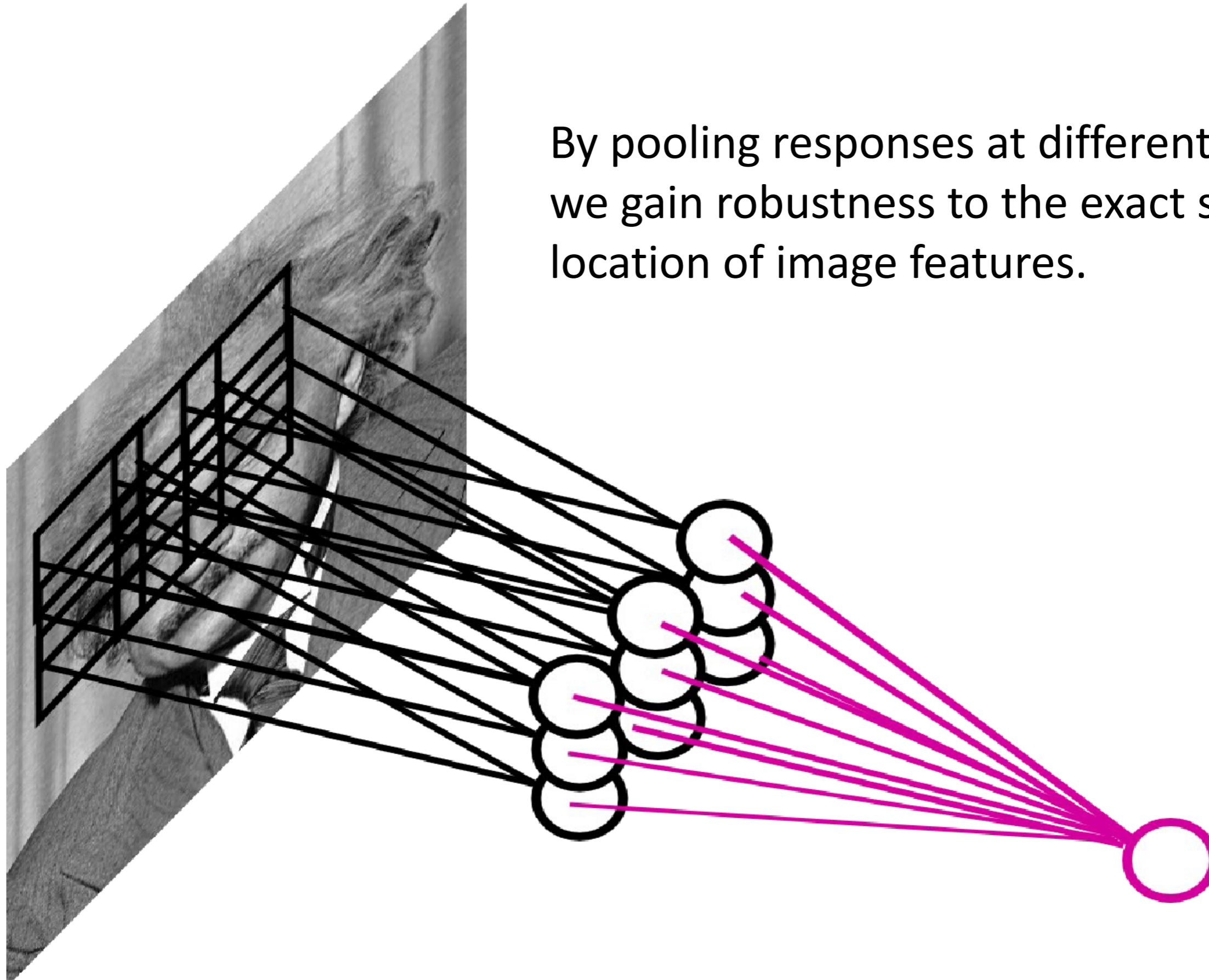
$$h_j^n(x, y) = \sqrt{\sum_{\bar{x} \in N(x), \bar{y} \in N(y)} h_j^{n-1}(\bar{x}, \bar{y})^2}$$

L2-pooling over features:

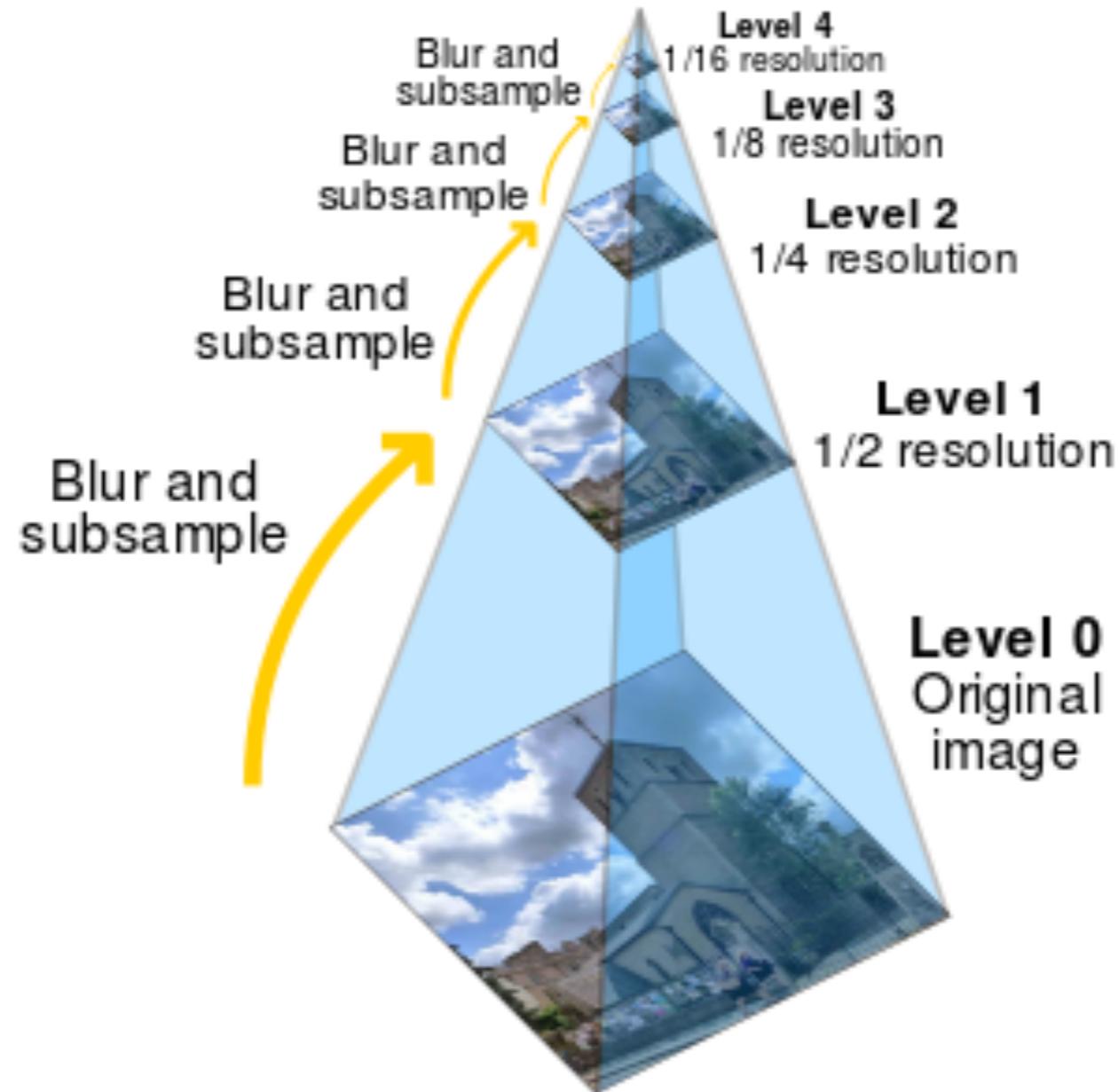
$$h_j^n(x, y) = \sqrt{\sum_{k \in N(j)} h_k^{n-1}(x, y)^2}$$

Pooling Layer

By pooling responses at different locations, we gain robustness to the exact spatial location of image features.

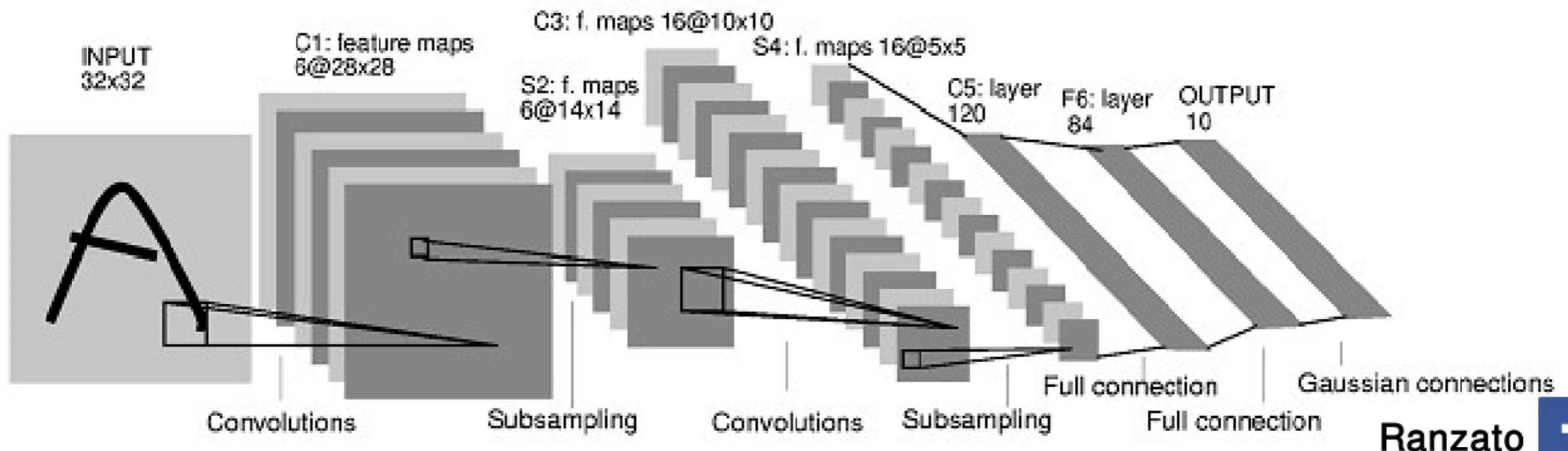


Pooling is similar to downsampling



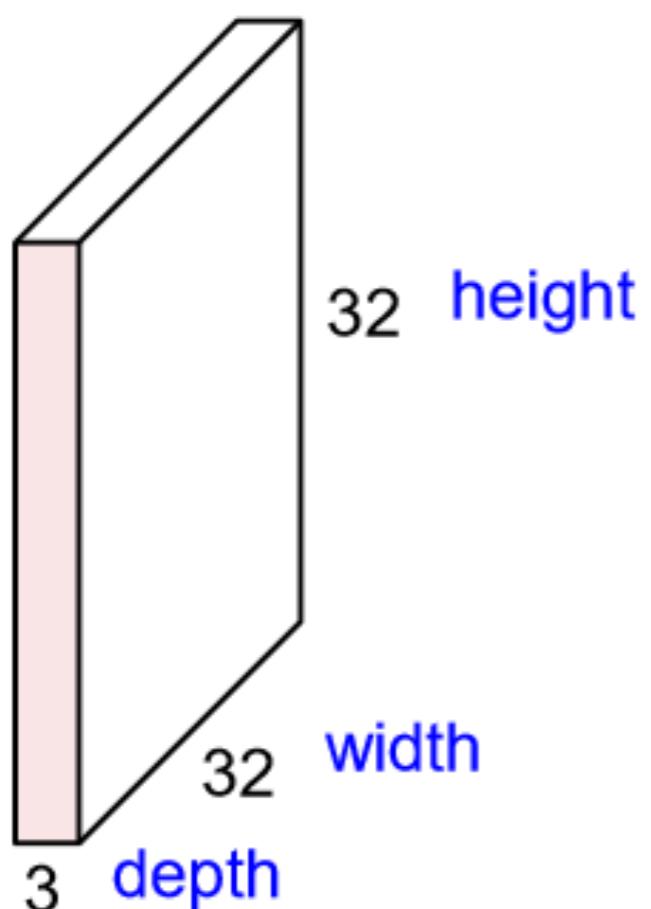
...except sometimes we don't want to blur,
as other functions might be better for classification.

Yann LeCun's MNIST CNN architecture



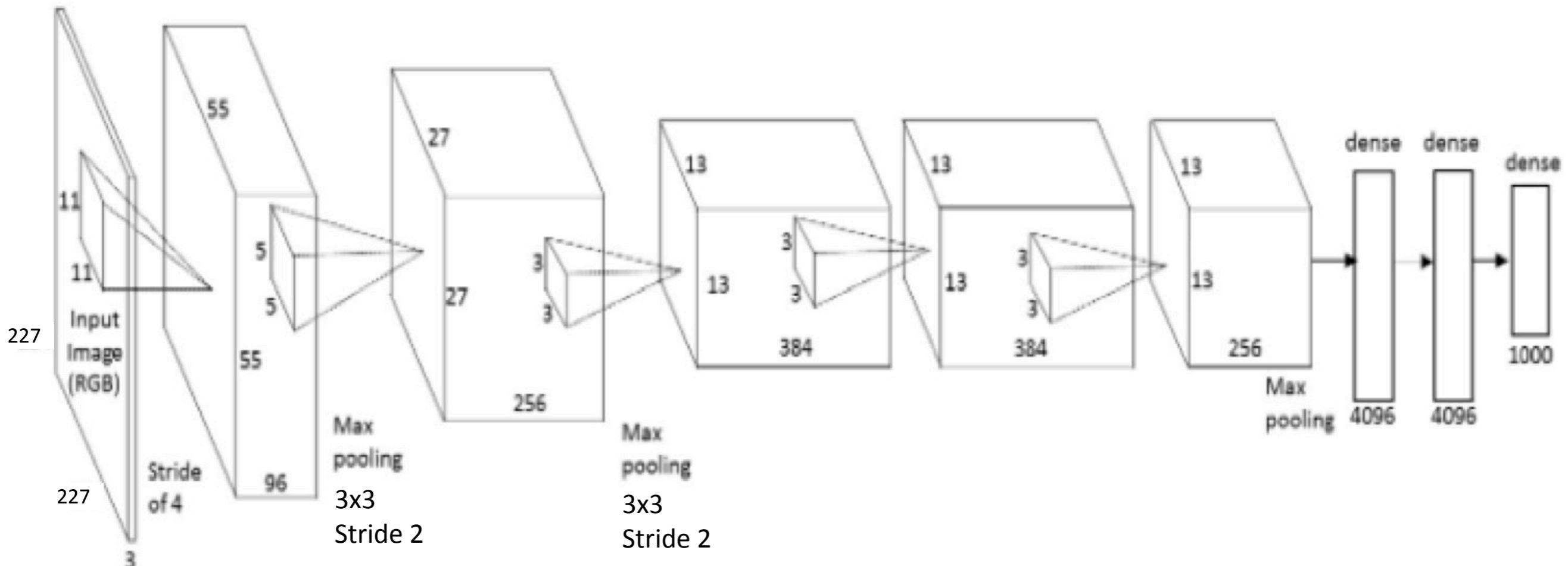
Convolutions: More detail

32x32x3 image



AlexNet diagram (simplified)

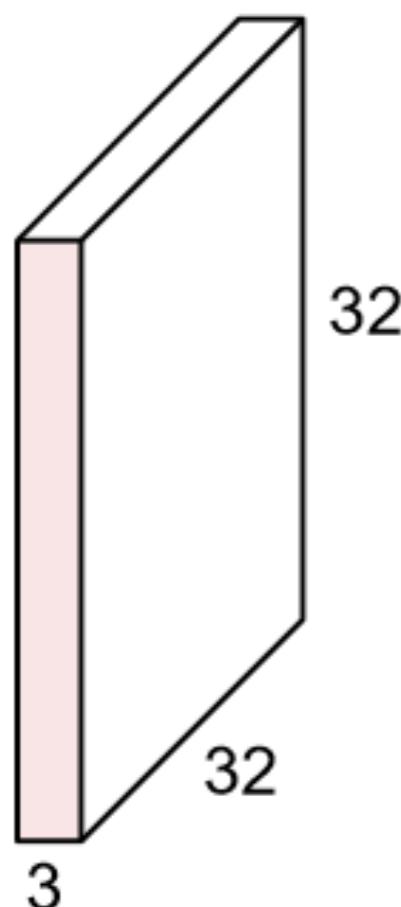
Input size
227 x 227 x 3



Conv 1	Conv 2	Conv 3	Conv 4	Conv 4
$11 \times 11 \times 3$	$5 \times 5 \times 48$	$3 \times 3 \times 256$	$3 \times 3 \times 192$	$3 \times 3 \times 192$
Stride 4	Stride 1	Stride 1	Stride 1	Stride 1
96 filters	256 filters	384 filters	384 filters	256 filters

Convolutions: More detail

32x32x3 image

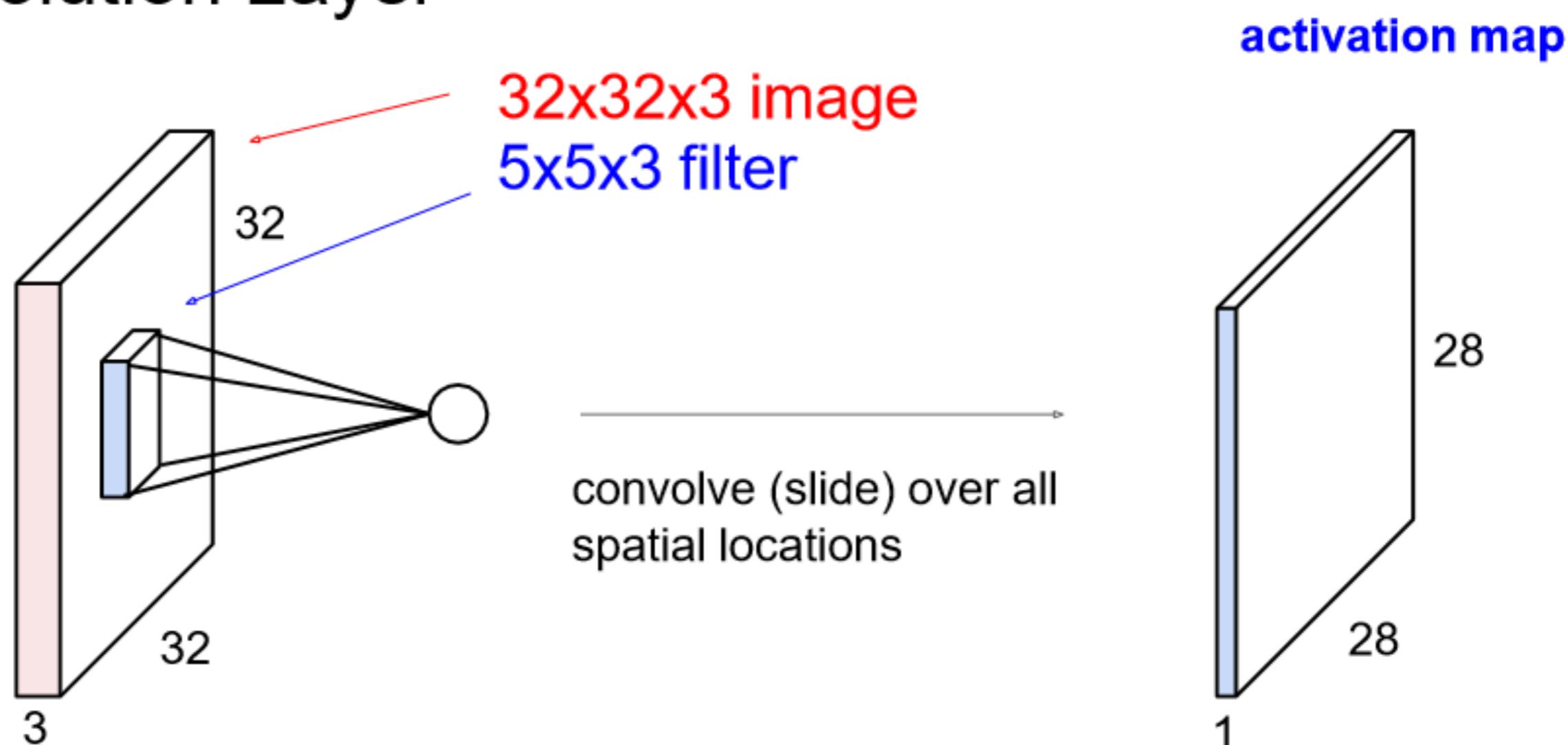


5x5x3 filter



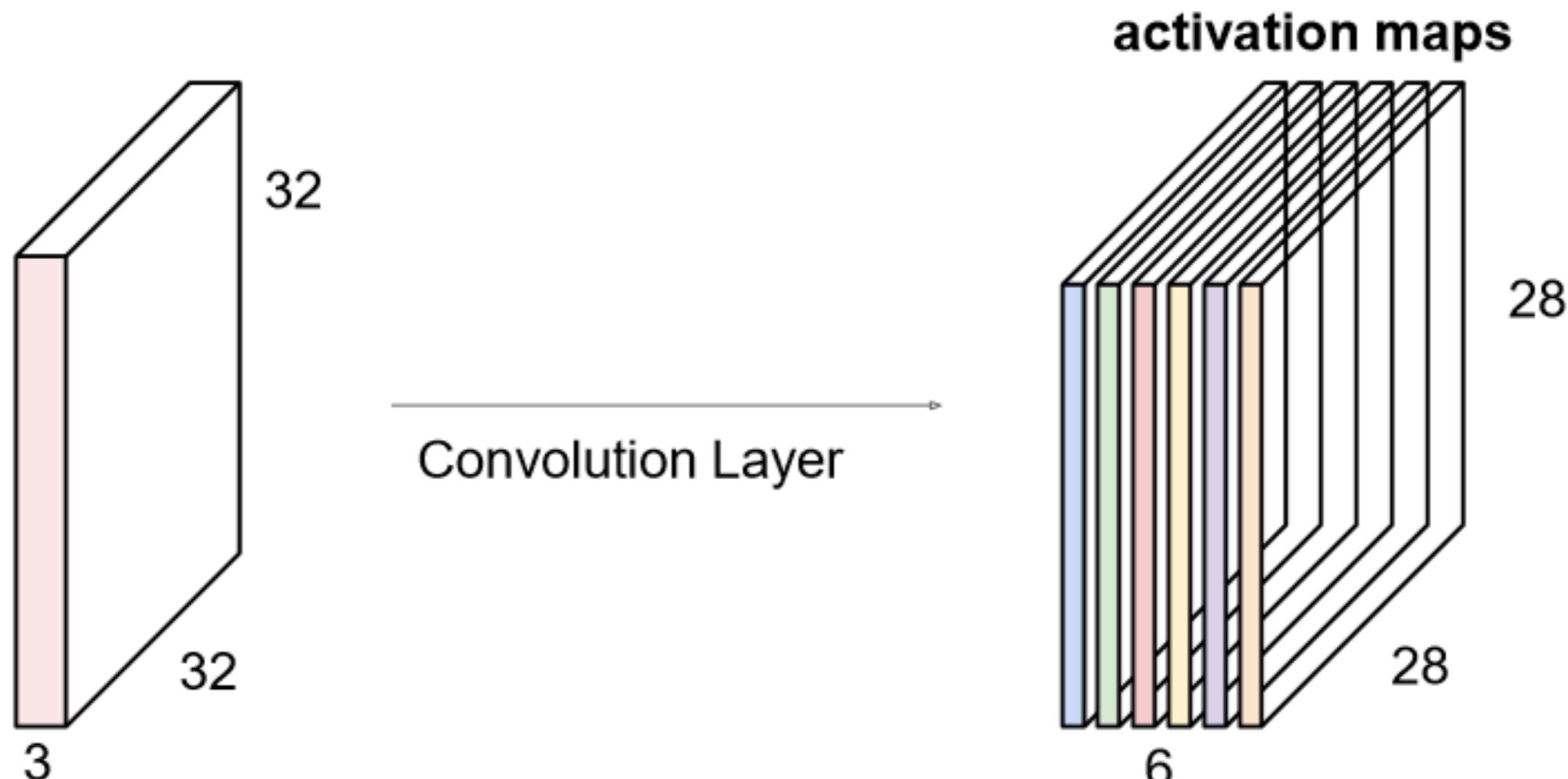
Convolutions: More detail

Convolution Layer



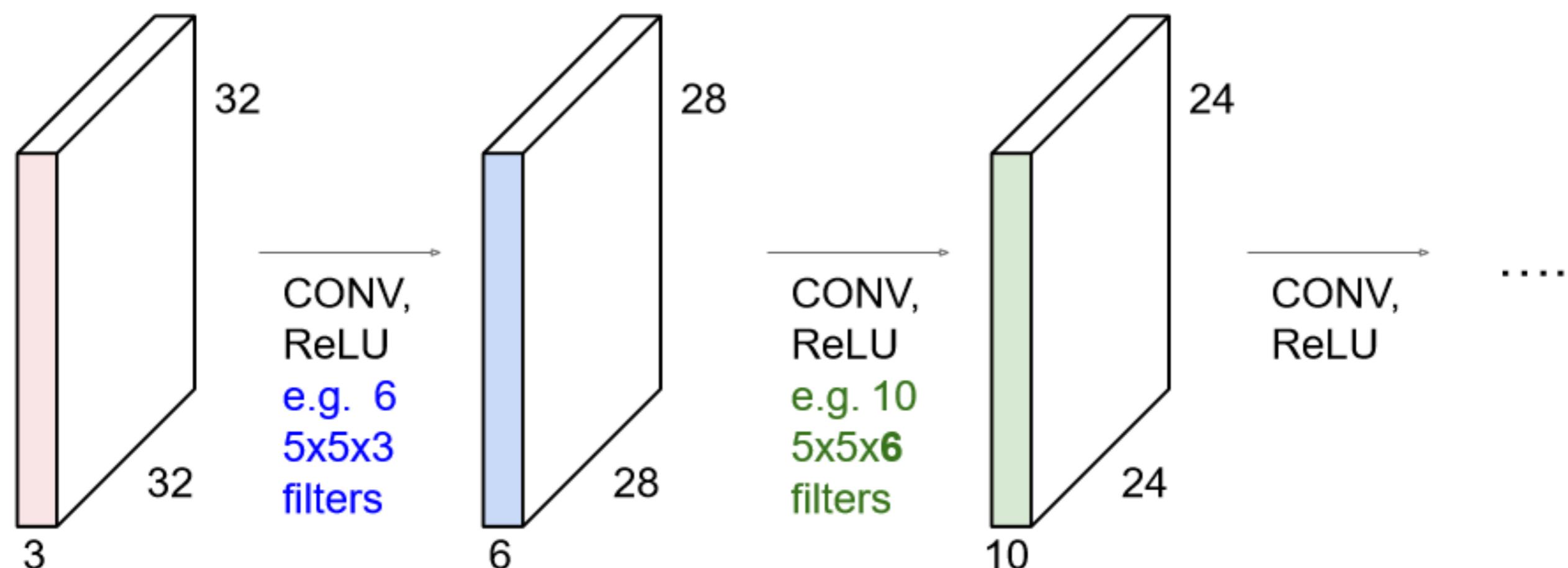
Convolutions: More detail

For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



We stack these up to get a “new image” of size $28 \times 28 \times 6$!

Convolutions: More detail

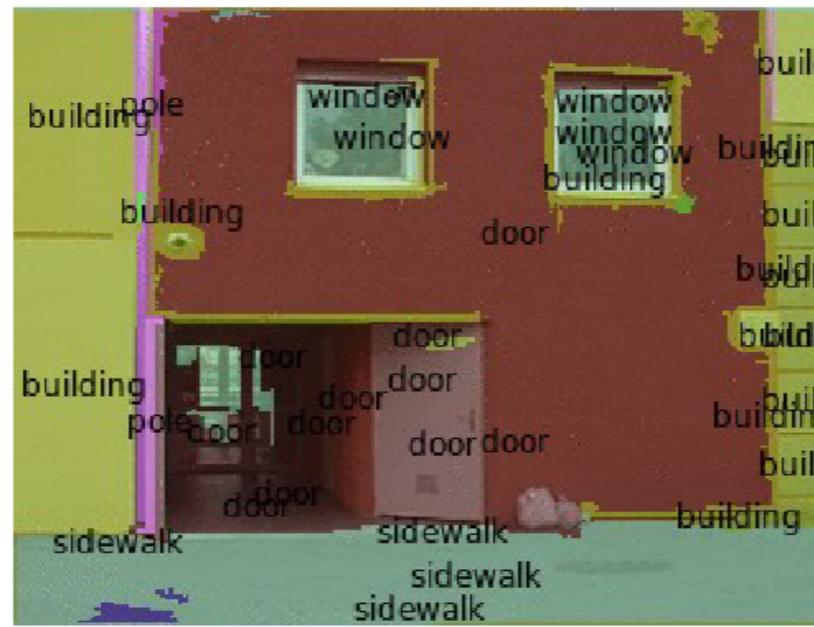
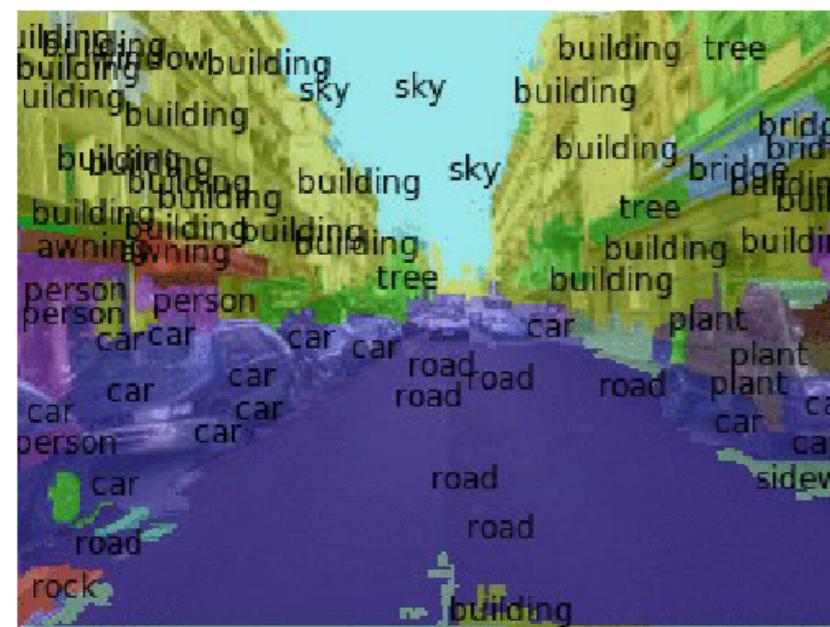


Outline

- Supervised Neural Networks
- Convolutional Neural Networks
- Examples
- Tips

CONV NETS: EXAMPLES

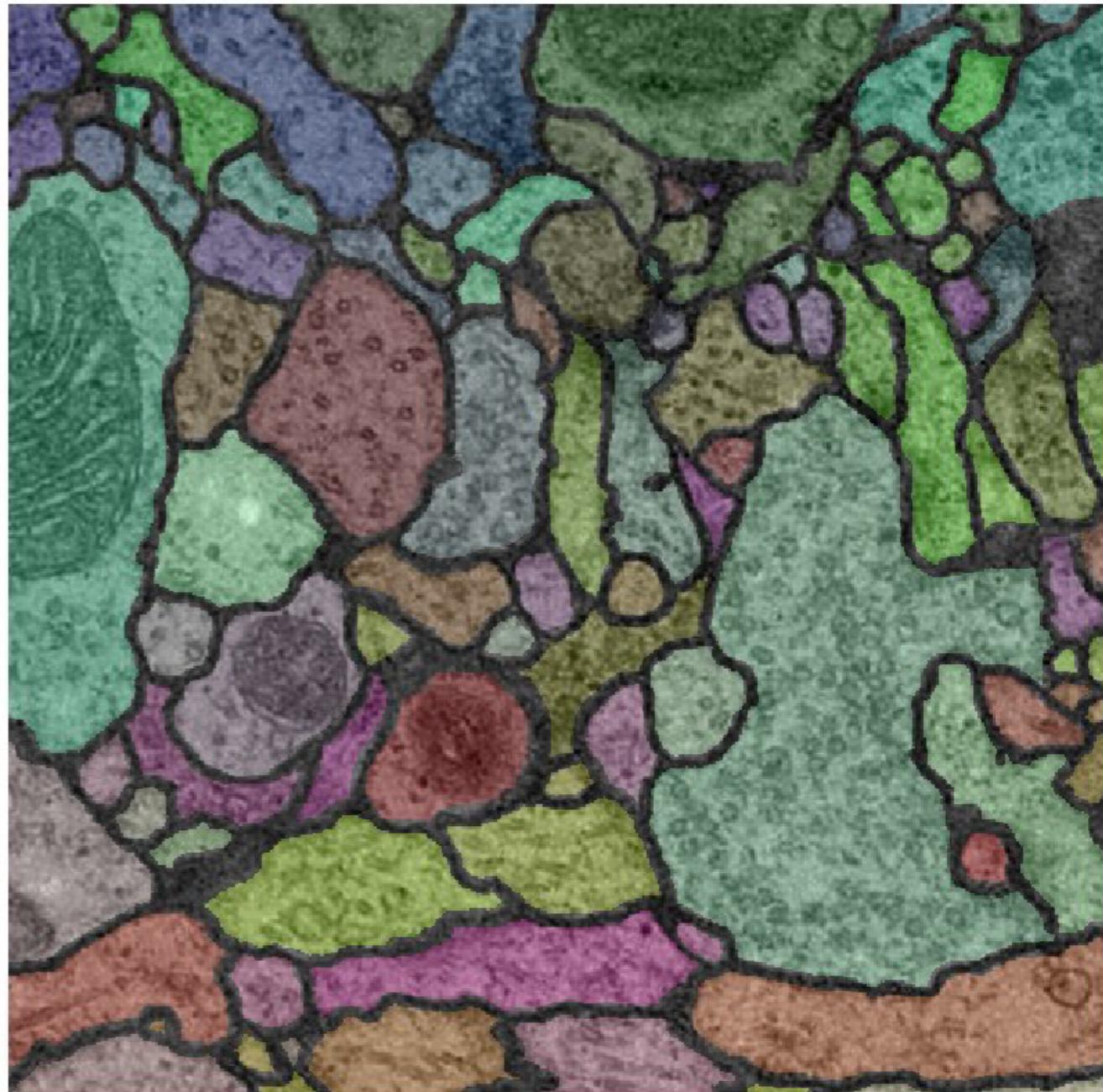
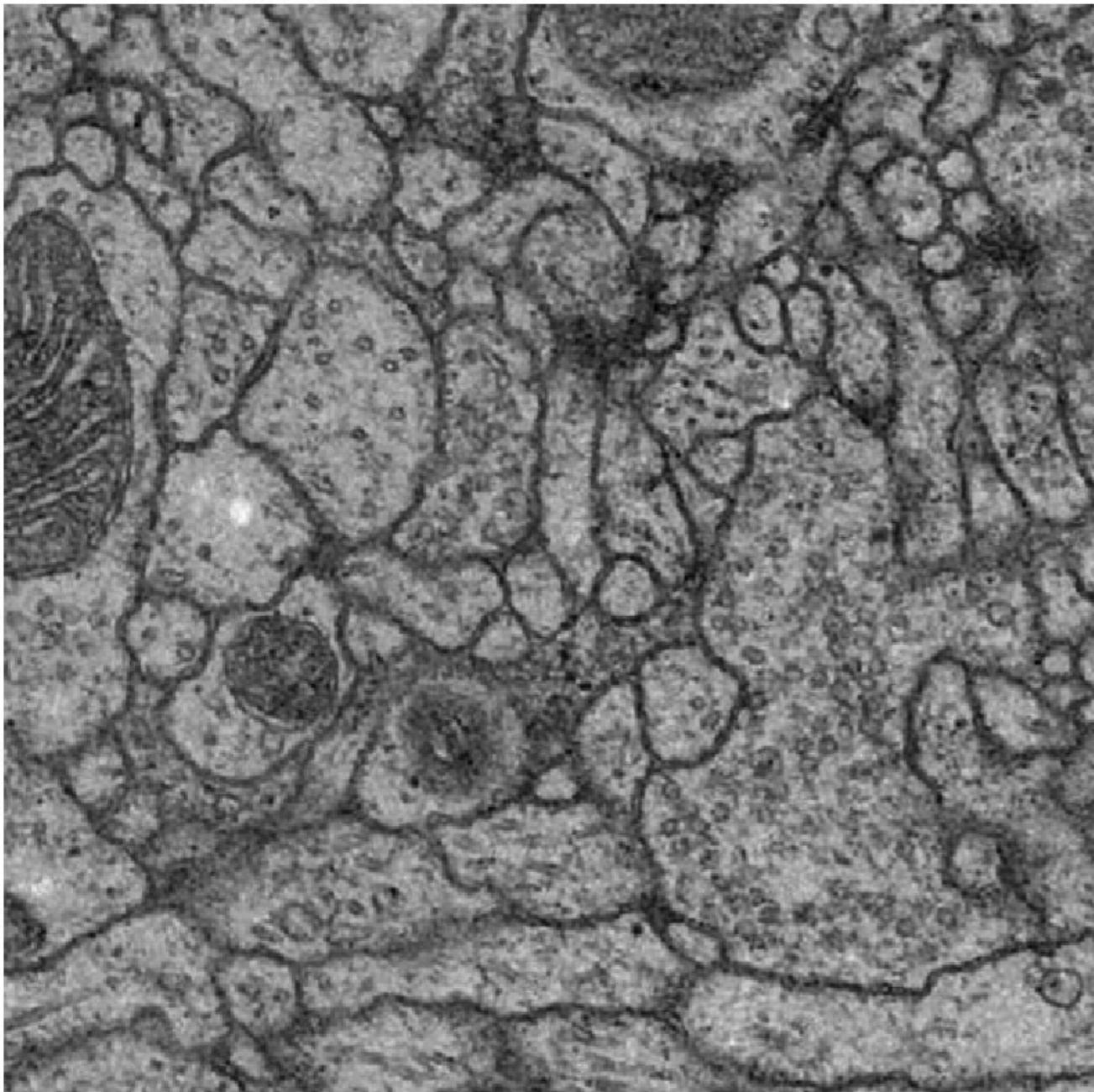
- Scene Parsing



Farabet et al. "Learning hierarchical features for scene labeling" PAMI 2013
Pinheiro et al. "Recurrent CNN for scene parsing" arxiv 2013

CONV NETS: EXAMPLES

- Segmentation 3D volumetric images

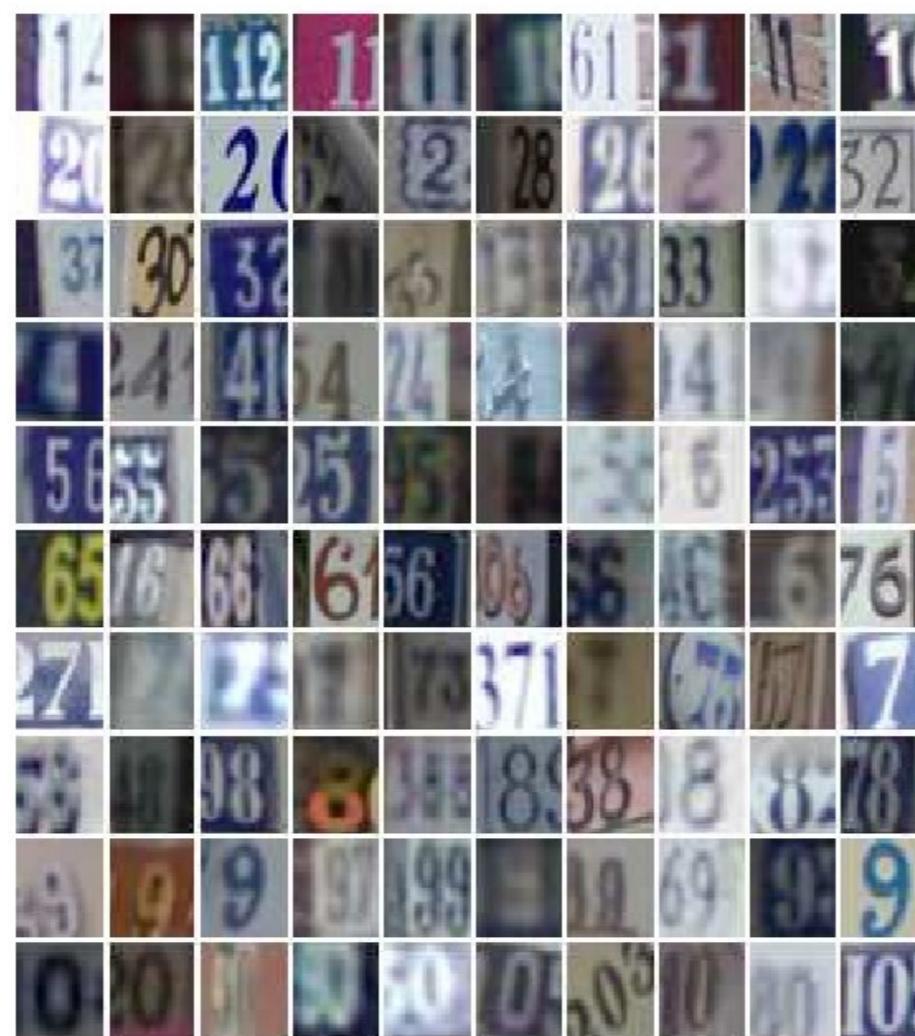
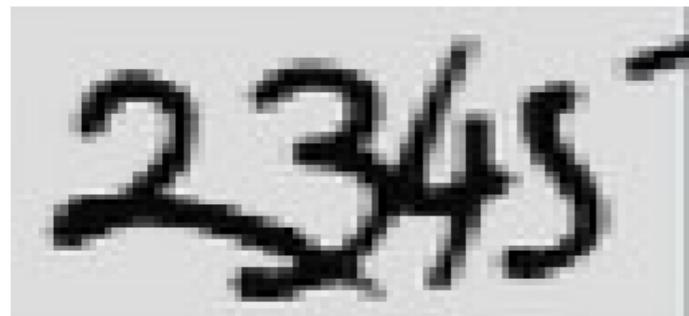


Ciresan et al. "DNN segment neuronal membranes..." NIPS 2012

Turaga et al. "Maximin learning of image segmentation" NIPS 2009

CONV NETS: EXAMPLES

- OCR / House number & Traffic sign classification



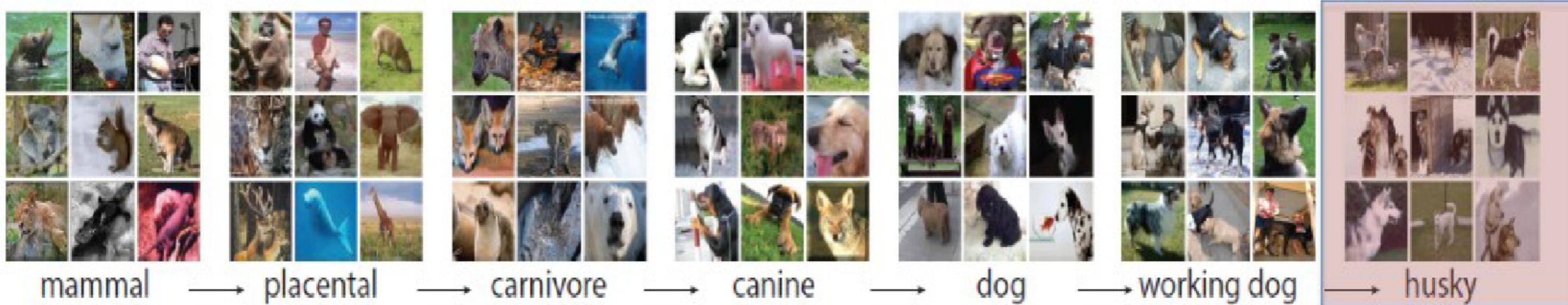
Ciresan et al. "MCDNN for image classification" CVPR 2012

Wan et al. "Regularization of neural networks using dropconnect" ICML 2013

82

Jaderberg et al. "Synthetic data and ANN for natural scene text recognition" arXiv 2014

Dataset: ImageNet 2012



- S: (n) [Eskimo dog](#), [husky](#) (breed of heavy-coated Arctic sled dog)
 - *direct hypernym / inherited hypernym / sister term*
- S: (n) [working dog](#) (any of several breeds of usually large powerful dogs bred to work as draft animals and guard and guide dogs)
- S: (n) [dog](#), [domestic dog](#), [Canis familiaris](#) (a member of the genus *Canis* (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds) "the dog barked all night"
 - S: (n) [canine](#), [canid](#) (any of various fissiped mammals with nonretractile claws and typically long muzzles)
 - S: (n) [carnivore](#) (a terrestrial or aquatic flesh-eating mammal) "terrestrial carnivores have four or five clawed digits on each limb"
 - S: (n) [placental](#), [placental mammal](#), [eutherian](#), [eutherian mammal](#) (mammals having a placenta; all mammals except monotremes and marsupials)
 - S: (n) [mammal](#), [mammalian](#) (any warm-blooded vertebrate having the skin more or less covered with hair; young are born alive except for the small subclass of monotremes and nourished with milk)
 - S: (n) [vertebrate](#), [craniate](#) (animals having a bony or cartilaginous skeleton with a segmented spinal column and a large brain enclosed in a skull or cranium)
 - S: (n) [chordate](#) (any animal of the phylum Chordata having a notochord or spinal column)
 - S: (n) [animal](#), [animate being](#), [beast](#), [brute](#), [creature](#), [fauna](#) (a living organism characterized by voluntary movement)
 - S: (n) [organism](#), [being](#) (a living thing that has (or can develop) the ability to act or function independently)
 - S: (n) [living thing](#), [animate thing](#) (a living (or once living) entity)
 - S: (n) [whole](#), [unit](#) (an assemblage of parts that is regarded as a single entity) "how big is that part compared to the whole?"; "the team is a unit"
 - S: (n) [object](#), [physical object](#) (a tangible and visible entity; an entity that can cast a shadow) "it was full of rackets, balls and other objects"
 - S: (n) [physical entity](#) (an entity that has physical existence)
 - S: (n) [entity](#) (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))



mite

mite
black widow
cockroach
tick
starfish

container ship

container ship
lifeboat
amphibian
fireboat
drilling platform

motor scooter

motor scooter
go-kart
moped
bumper car
golfcart

leopard

leopard
jaguar
cheetah
snow leopard
Egyptian cat



grille

convertible
grille
pickup
beach wagon
fire engine

mushroom

agaric
mushroom
jelly fungus
gill fungus
dead-man's-fingers

cherry

dalmatian
grape
elderberry
ffordshire bullterrier
currant

Madagascar cat

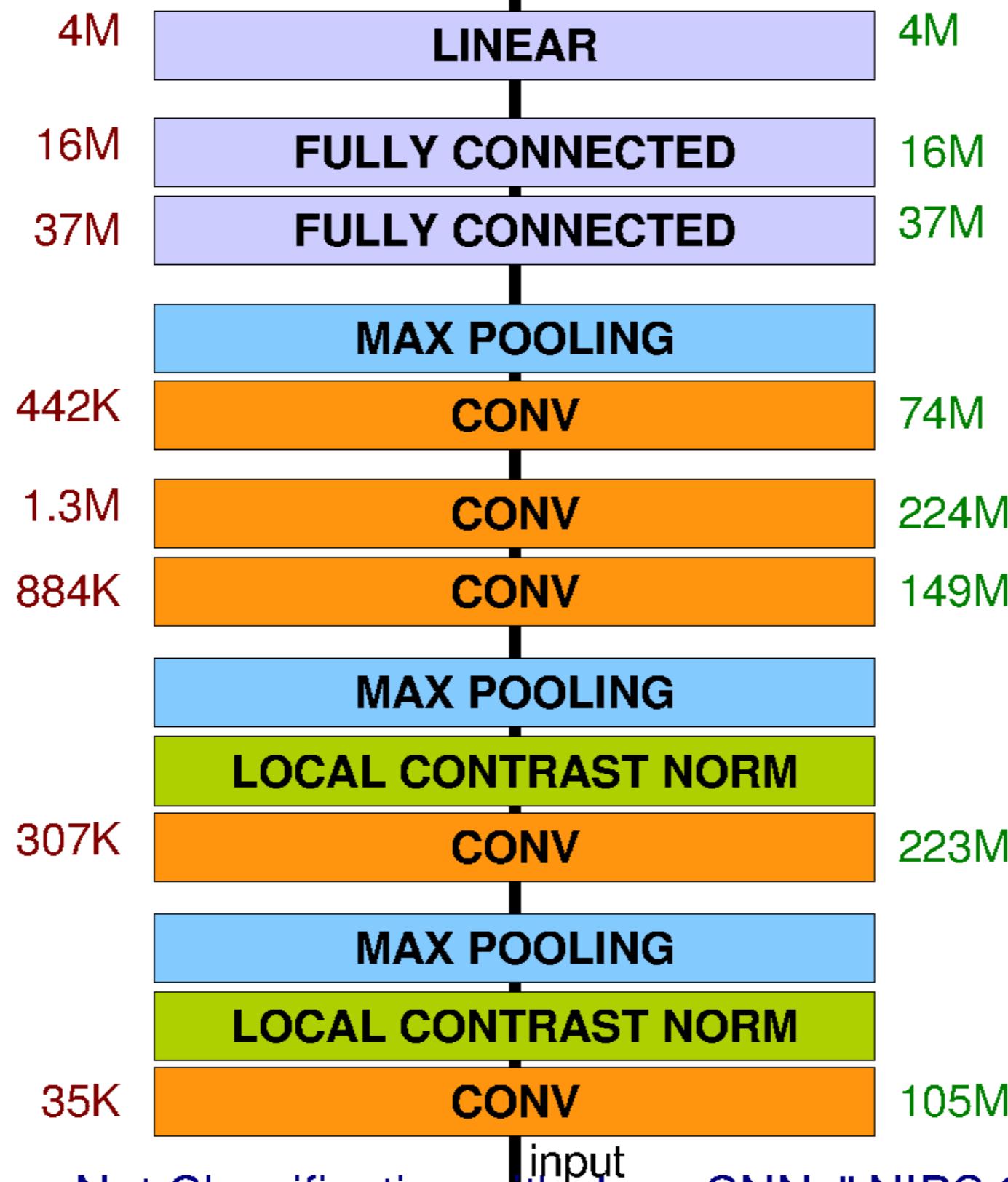
squirrel monkey
spider monkey
titi
indri
howler monkey

Architecture for Classification

Total nr. params: 60M

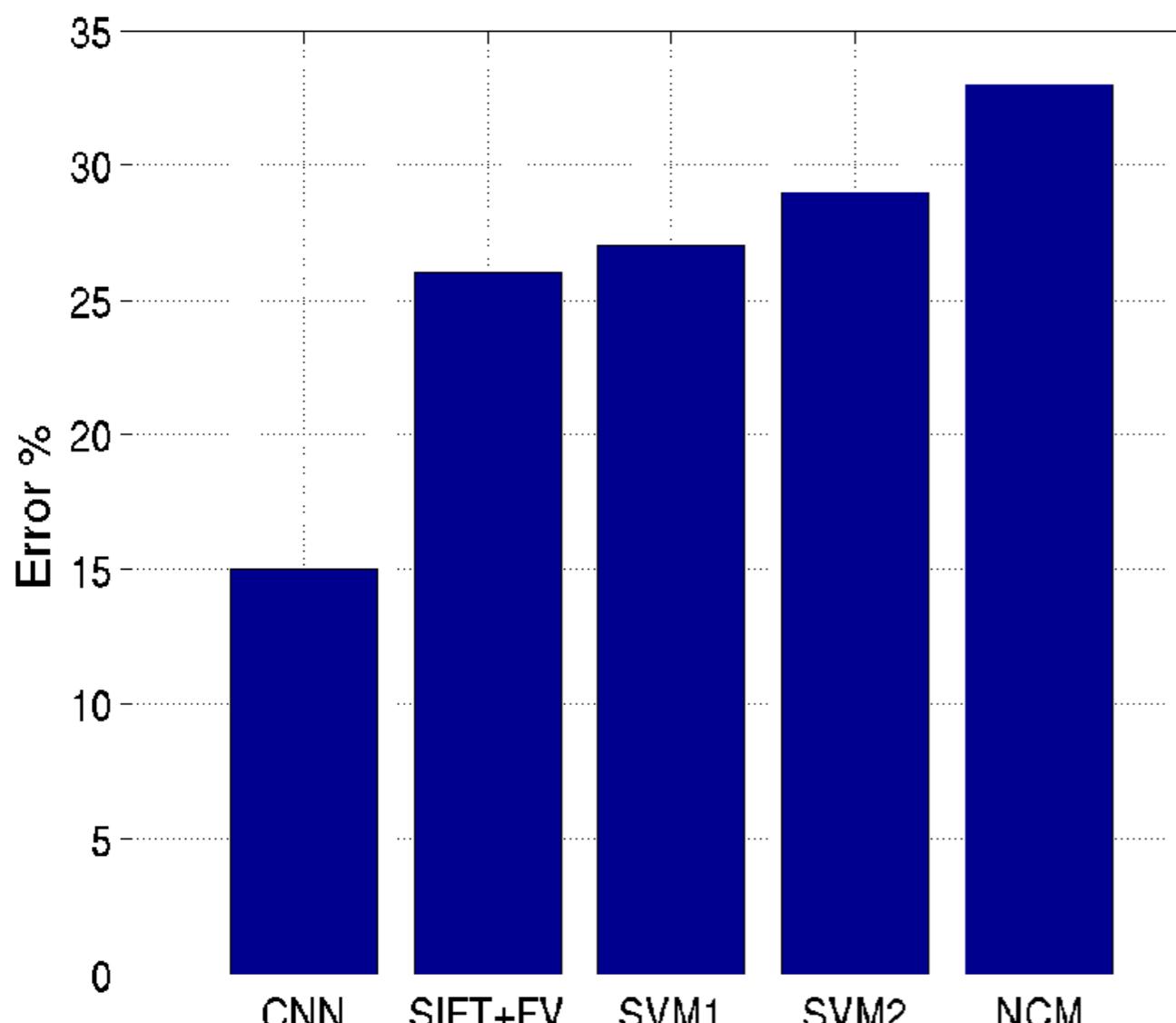
category
prediction

Total nr. flops: 832M

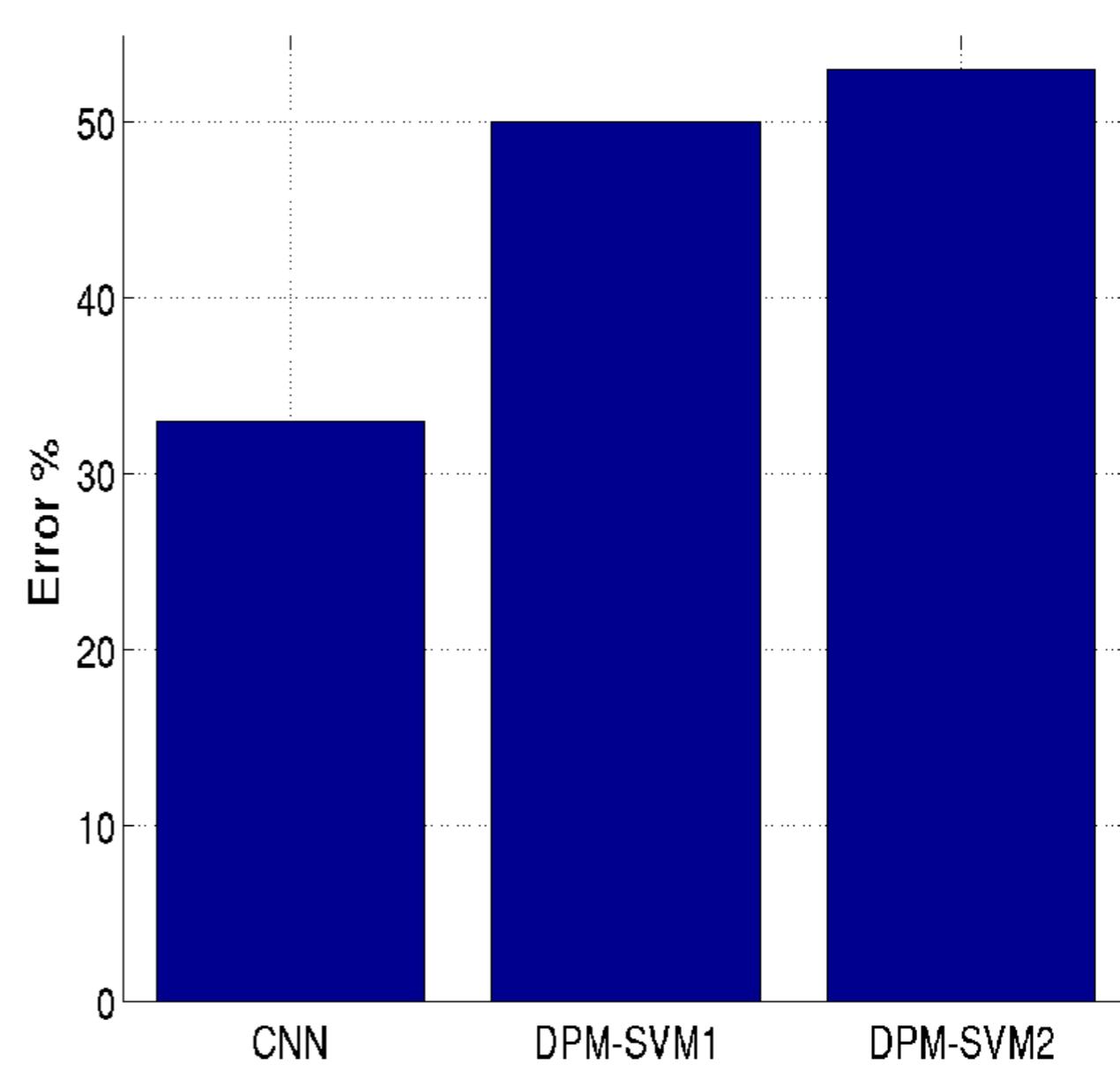


Results: ILSVRC 2012

TASK 1 - CLASSIFICATION



TASK 2 - DETECTION



More ConvNet explanations

- <https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/>