CONVOLUTIONAL NEURAL NETWORKS

- ThS. Đoàn Chánh Thống
- ThS. Nguyễn Cường Phát
- ThS. Nguyễn Hữu Lợi
- ThS. Trương Quốc Dũng
- ThS. Nguyễn Thành Hiệp
- ThS. Võ Duy Nguyên
- ThS. Nguyễn Văn Toàn
- ThS. Lê Ngô Thục Vi
- TS. Nguyễn Duy Khánh
- TS. Nguyễn Tấn Trần Minh Khang

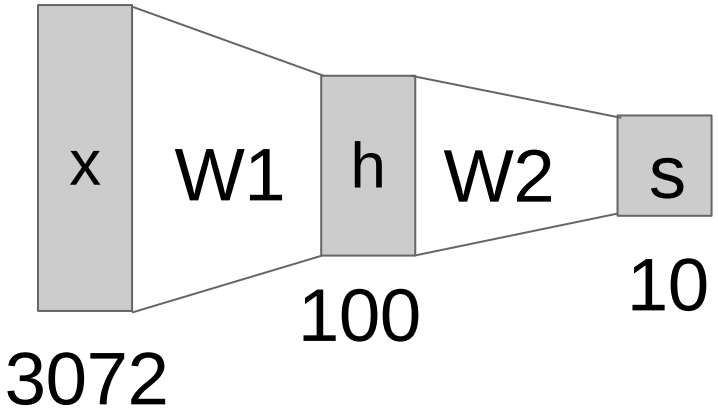
Administrative

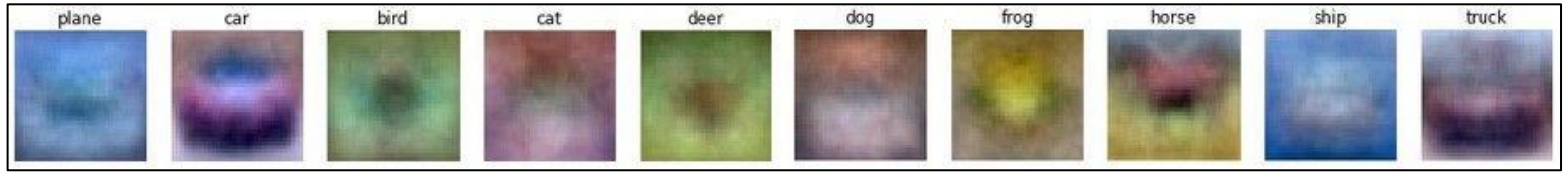
- Assignment 1 due Thursday April 20, 11:59 pm on Canvas.
- Bài tập 1 sẽ nộp vào Thứ Năm ngày 20 tháng 4 lúc 23:59 trên Canvas.
- Assignment 2 will be released Thursday.
- Bài tập 2 sẽ được công bố vào thứ năm.

Last time: Neural Networks

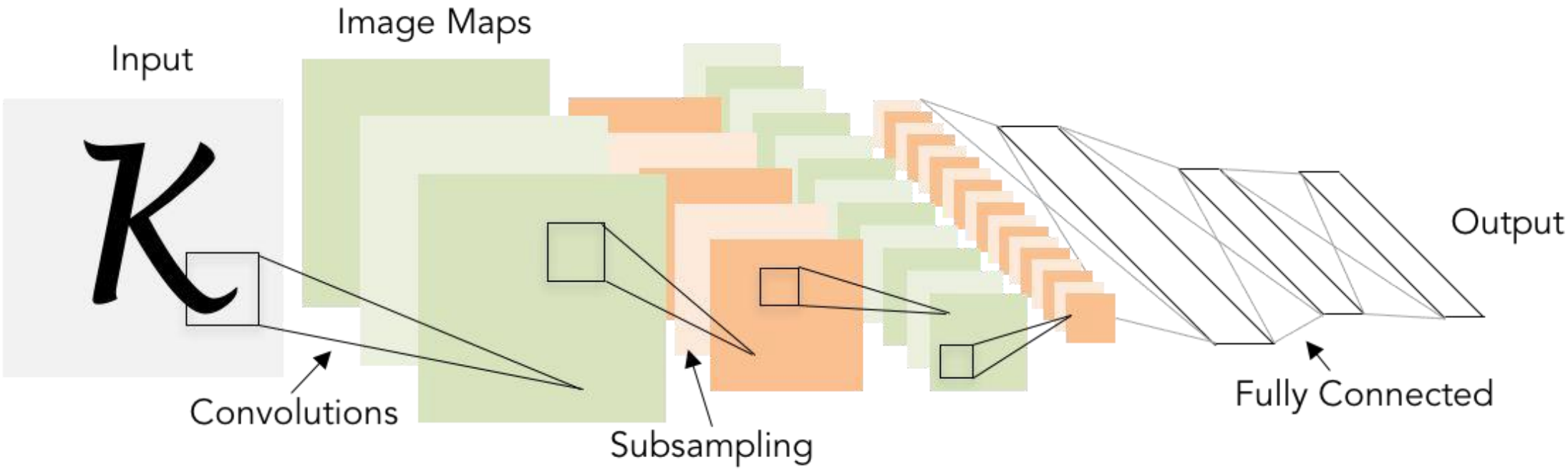
– Linear score function: $f = Wx$

– 2 – layer Neural Network: $f = W_2 max(0, W_1 x)$





Next: Convolutional Neural Networks



A BIT OF HISTORY

1957: Perceptron

- The Mark I Perceptron machine was the first implementation of the perceptron algorithm.
- The machine was connected to a camera that used 20×20 cadmium sulfide photocells to produce a 400-pixel image.

1957: Perceptron

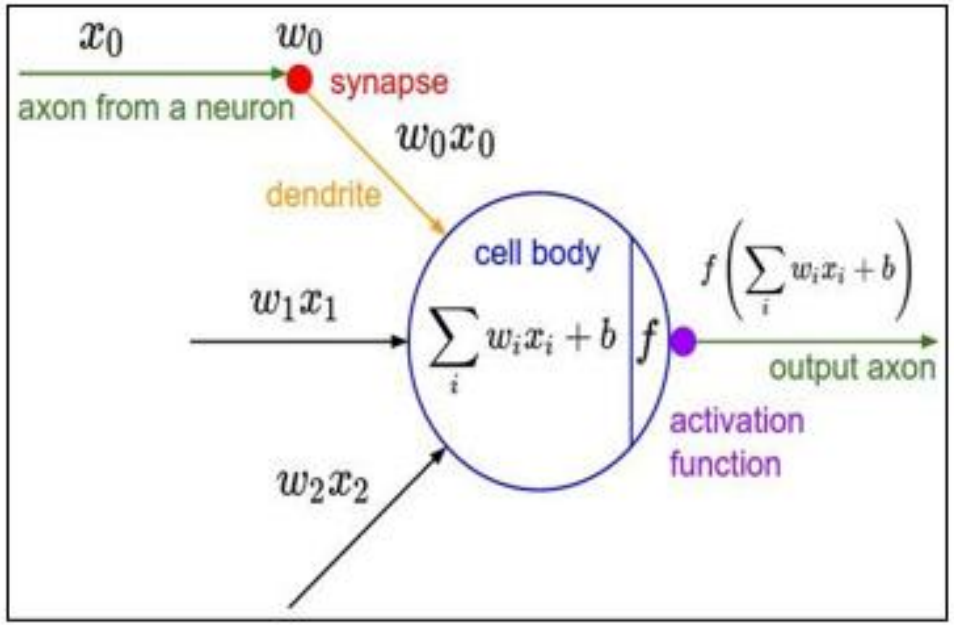
- Recognized letters of the alphabet

$$f(x) = \begin{cases} 1 & if\ w \cdot x + b > 0 \\ 0 & otherwiswe \end{cases}$$

- Update rules

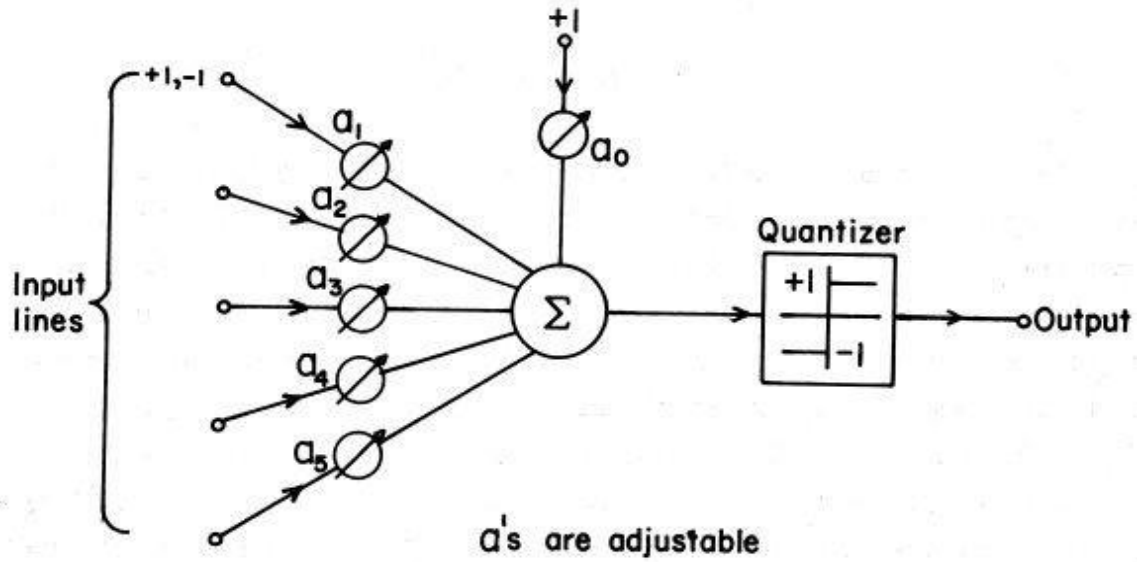
$$w_i(t+1) = w_i(t) + \alpha\left(d_j - y_j(t)\right)x_{j,i}$$

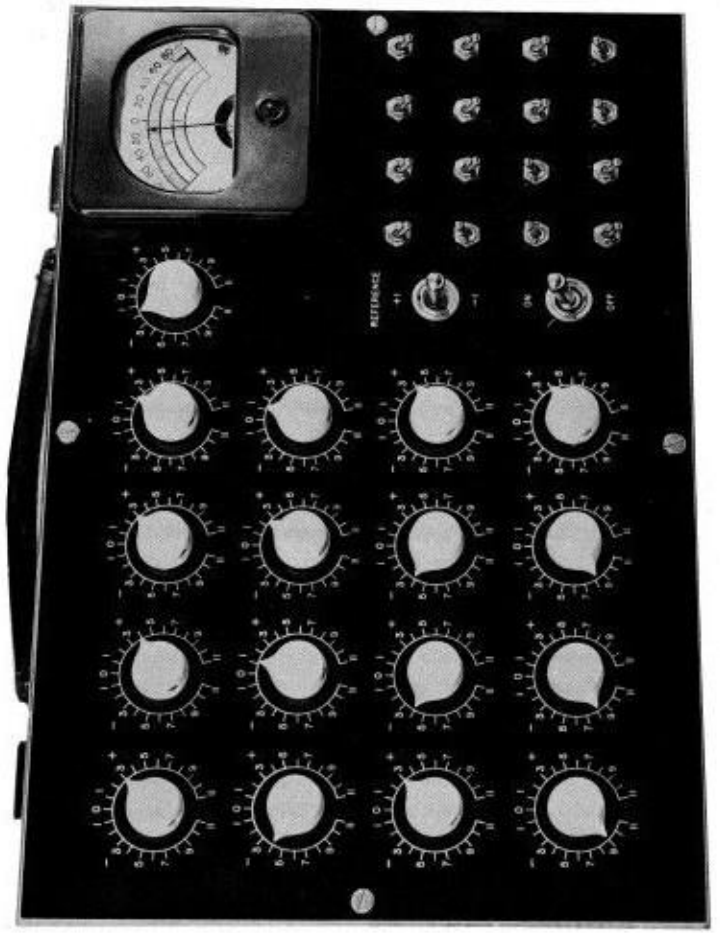
- Frank Rosenblatt, ~1957: Perceptron.



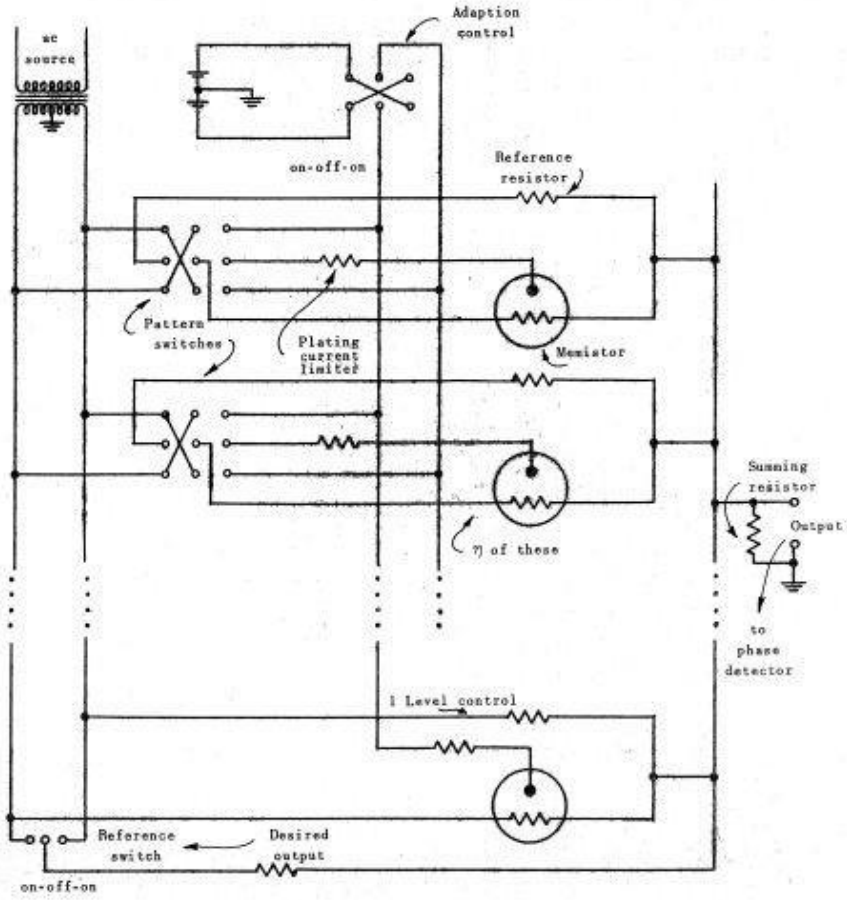
1960: Adaline/Madaline

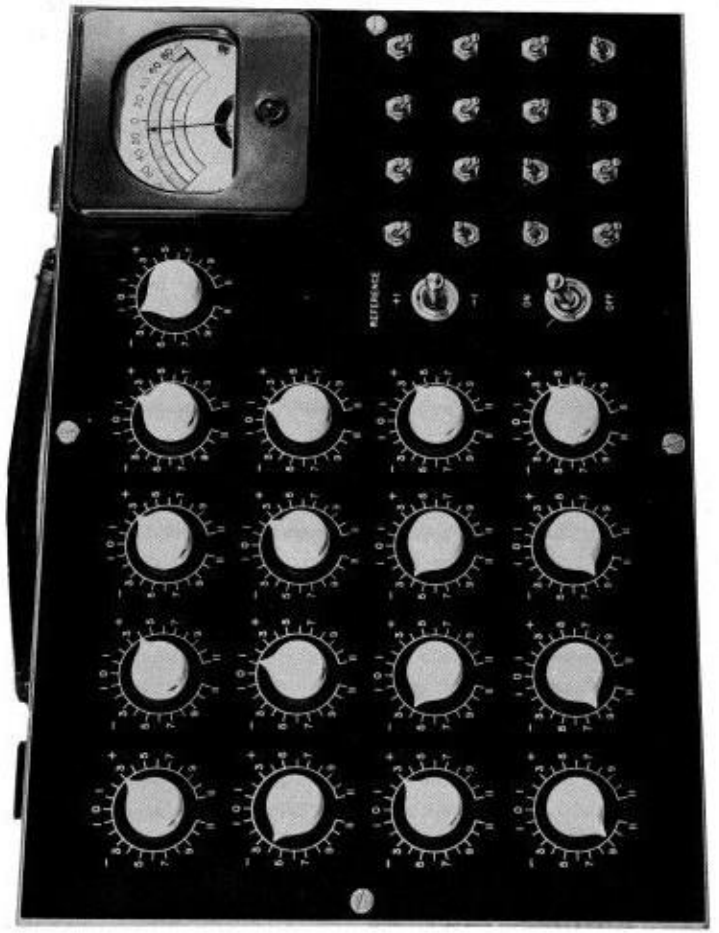
– Widrow and Hoff, ~1960: Adaline/Madaline.





1960: Adaline/Madaline



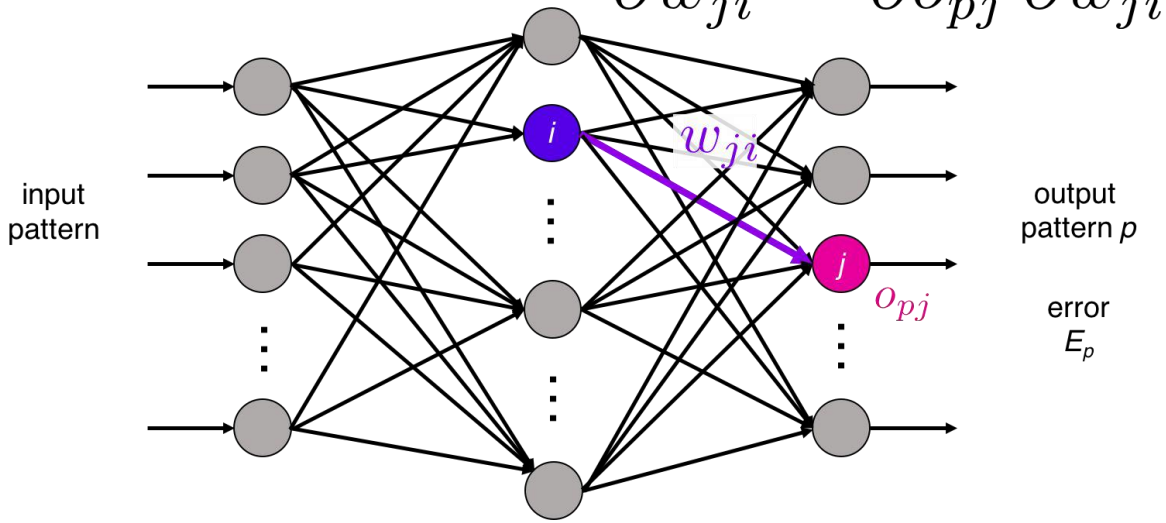


1986: Back – propagation

- Rumelhart, 1986: First time back – propagation became popular.
- Công trình của Rumelhart đã làm cho thuật toán lan truyền ngược trở nên phổ biến và được công nhận rộng rãi trong cộng đồng nghiên cứu.

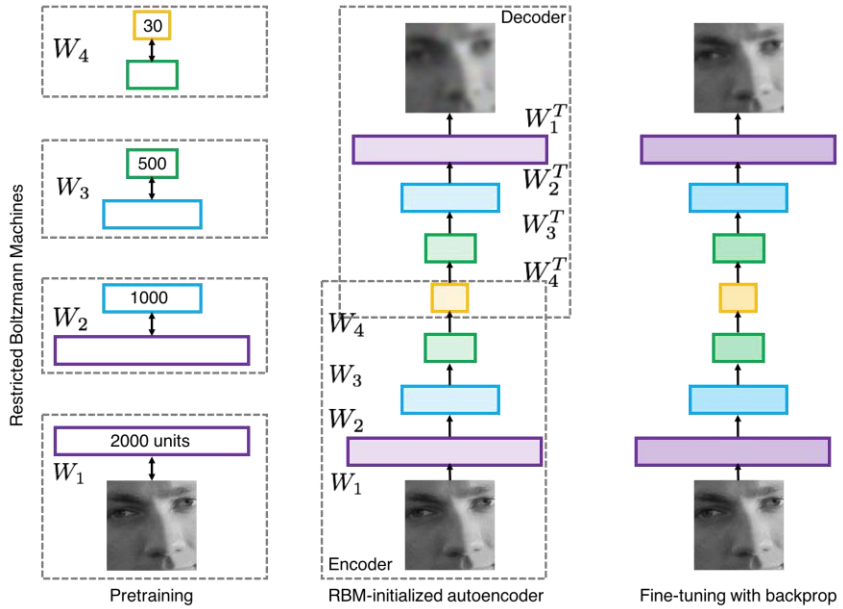
$$\frac{\partial E_p}{\partial w_{ji}} = \frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial w_{ji}}$$

recognizable math



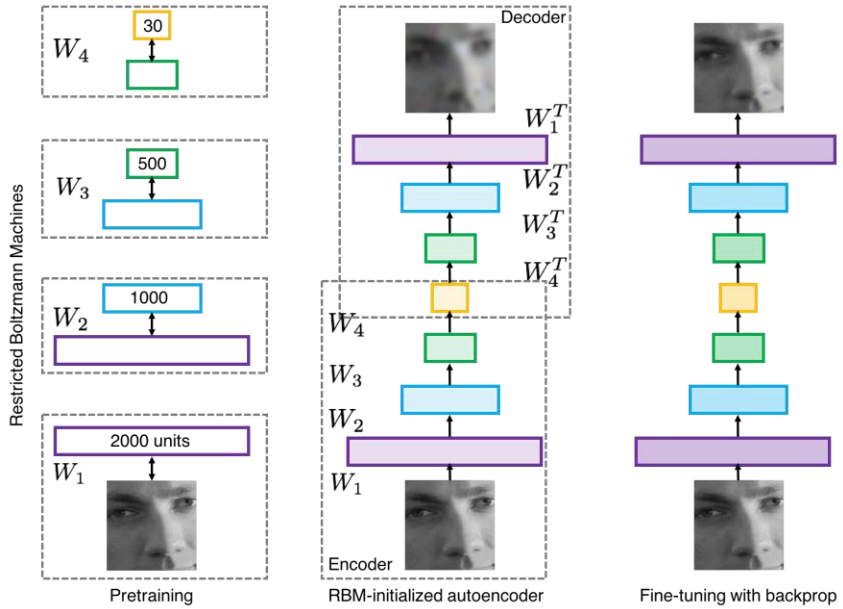
2006, Research in Deep Learning

- Hinton and Salakhutdinov 2006: Reinvigorated research in Deep Learning.
- Công bố "Reducing the Dimensionality of Data with Neural Networks" trên tạp chí Science năm 2006. Công trình đã chứng minh sức mạnh của các mạng nơ-ron sâu và góp phần quan trọng vào sự tái quan tâm đến các kỹ thuật học sâu.



2006, Research in Deep Learning

- Hinton và Salakhutdinov đã giới thiệu cách sử dụng autoencoders sâu (deep autoencoders) để giảm số chiều dữ liệu.
- Công trình của Hinton và Salakhutdinov đã chứng minh được sức mạnh của các mạng nơ-ron sâu và đóng góp quan trọng vào sự tái quan tâm đến các kỹ thuật học sâu.



2010, First strong results

- Acoustic Modeling using Deep Belief Networks
- Abdel-rahman Mohamed, George Dahl, Geoffrey Hinton, 2010.
- Công trình giới thiệu Deep Belief Networks (DBN) để xây dựng các mô hình âm học, đây là một phần quan trọng trong nhận dạng giọng nói tự động (Automatic Speech Recognition - ASR).

SUBMITTED TO IEEE TRANS. ON AUDIO, SPEECH, AND LANGUAGE PROCESSING 1

Acoustic Modeling using Deep Belief Networks

Abdel-rahman Mohamed, George E. Dahl, and Geoffrey Hinton

Abstract—**Gaussian mixture models are currently the dominant technique for modeling the emission distribution of hidden Markov models for speech recognition. We show that better phone recognition on the TIMIT dataset can be achieved by replacing Gaussian mixture models by deep neural networks that contain many layers of features and a very large number of parameters. These networks are first pre-trained as a multi-layer generative model of a window of spectral feature vectors without making use of any discriminative information. Once the generative pre-training has designed the features, we perform discriminative fine-tuning using backpropagation to adjust the features slightly to make them better at predicting a probability distribution over the states of monophone hidden Markov models.**

Index Terms—**Acoustic Modeling, deep belief networks, neural networks, phone recognition**

I. Introduction

Automatic speech Recognition (ASR) has evolved significantly over the past few decades. Early systems typically discriminated isolated digits or yes/no, whereas current systems can do quite well at recognizing telephone-quality, spontaneous speech. A huge amount of progress has been made in improving word recognition rates, but the core acoustic modeling has remained fairly stable, despite many attempts to develop better alternatives.

A typical ASR system uses Hidden Markov Models (HMMs) to model the sequential structure of speech signals, with each HMM state using a mixture of Gaussians to model a spectral representation of the sound wave. The most common spectral representation is a set of Mel Frequency Cepstral coefficients (MFCCs) derived from a window of about 25 ms of speech. The window is typically advanced by about 10 ms per frame, and each frame of coefficients is augmented with differences and differences of differences with nearby frames.

One research direction involves using deeper acoustic models that contain many layers of features. The work in [1] proposes a hierarchical framework where each layer is designed to capture a set of distinctive feature landmarks. For each feature, a specialized acoustic representation is constructed in which that feature is easy to detect. In [2], a probabilistic generative model is introduced where the dynamic structure in the hidden vocal tract resonance space is used to characterize long-span contextual influence across phonetic units.

Feedforward neural networks have been used in many using a feature vector that describes segments of the temporal evolution of critical-band spectral densities within a single critical band. Sub-word posterior probabilities are estimated using feedforward neural networks for each critical band and these probabilities are merged to produce the final estimate of the posterior probabilities using another feedforward neural network. In [8], the split temporal context system is introduced which modifies the TRAP system by including, in the middle layer of the system, splits over time as well as over frequency bands before the final merger neural network.

Feedforward neural networks offer several potential advantages over GMMs:

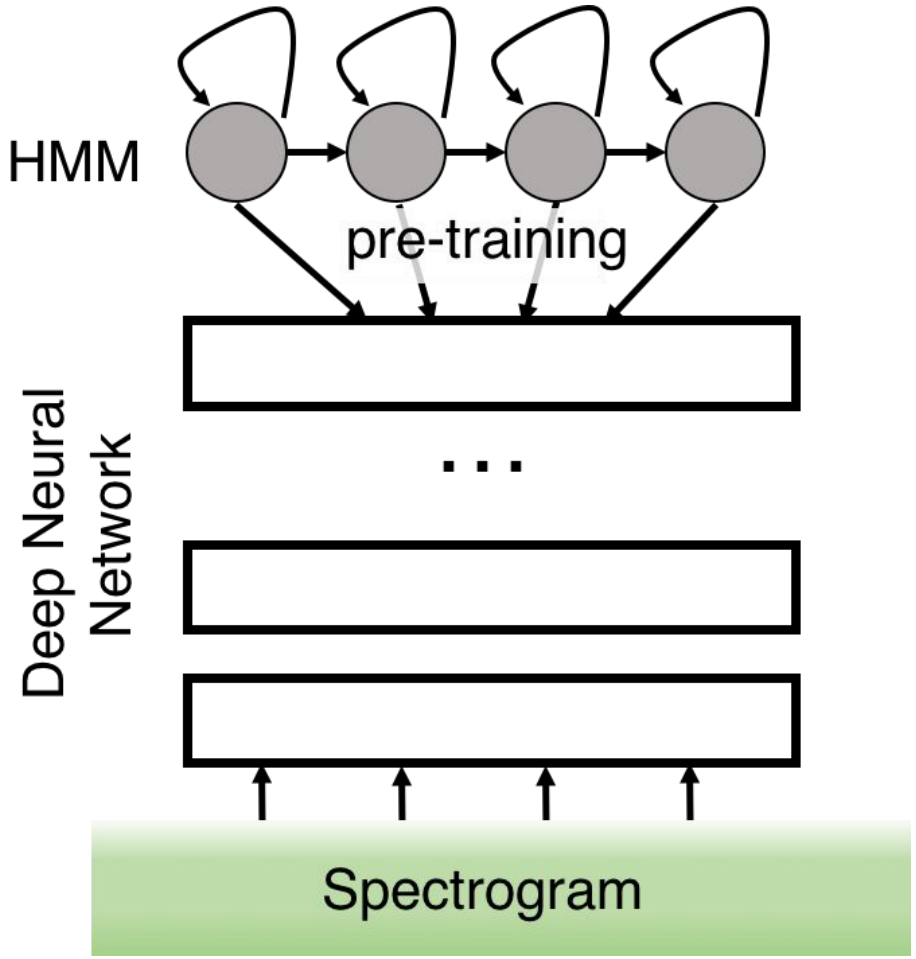
- Their estimation of the posterior probabilities of HMM states does not require detailed assumptions about the data distribution.
- They allow an easy way of combining diverse features, including both discrete and continuous features.
- They use far more of the data to constrain each parameter because the output on each training case is sensitive to a large fraction of the weights.

The benefit of each weight in a neural network being constrained by a larger fraction of training cases than each parameter in a GMM has been masked by other differences in training. Neural networks have traditionally been trained purely discriminatively, whereas GMMs are typically trained as generative models (even if discriminative training is performed later in the training procedure). Generative training allows the data to impose many more bits of constraint on the parameters (see below), thus partially compensating for the fact that each component of a large GMM must be trained on a very small fraction of the data.

MFCCs, GMMs, and HMMs co-evolved as a way of doing speech recognition when computers were too slow to explore more computationally intensive approaches. MFCCs throw away a lot of the information in the sound wave, but preserve most of the information required for discrimination. By including temporal differences, MFCCs partially overcome the very strong conditional independence assumption of HMMs, namely that successive frames are independent given the hidden state of the HMM. The temporal differences also allow diagonal covariance Gaussians to model the strong temporal covariances by reducing these particular pairwise covariances to individual coefficients. As we shall see, a Fourier transform

2012, First strong results

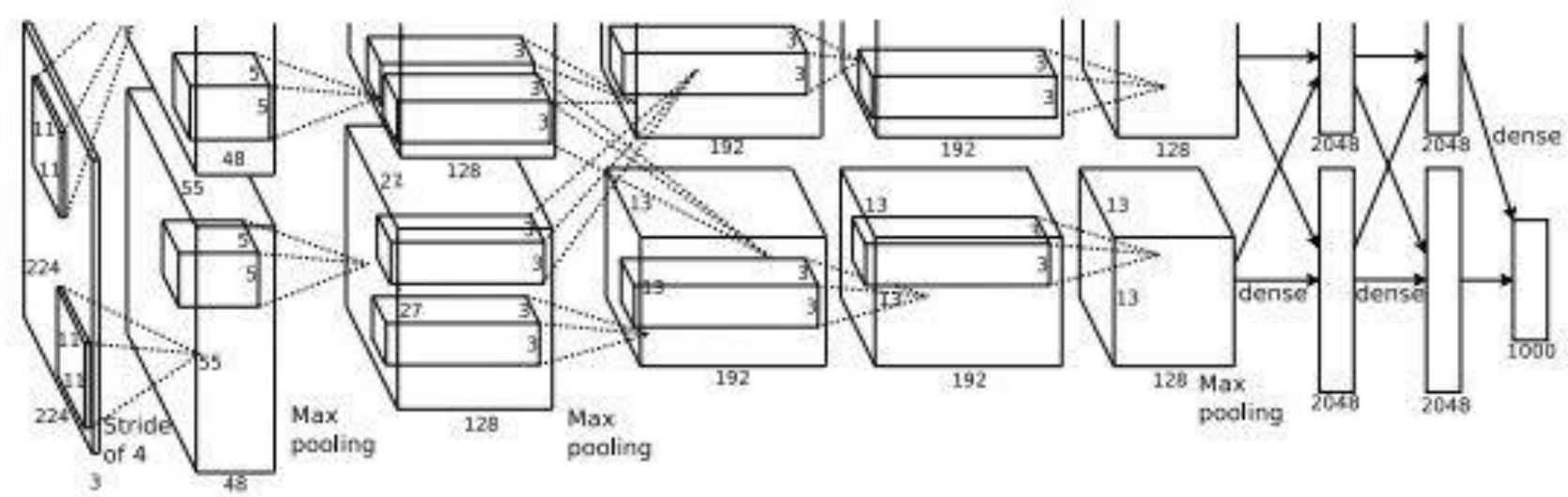
- Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition
- George Dahl, Dong Yu, Li Deng, Alex Acero, 2012.
- Sử dụng các mạng nơ-ron sâu để nhận dạng giọng nói với số lượng từ vựng lớn.



2012, First strong results

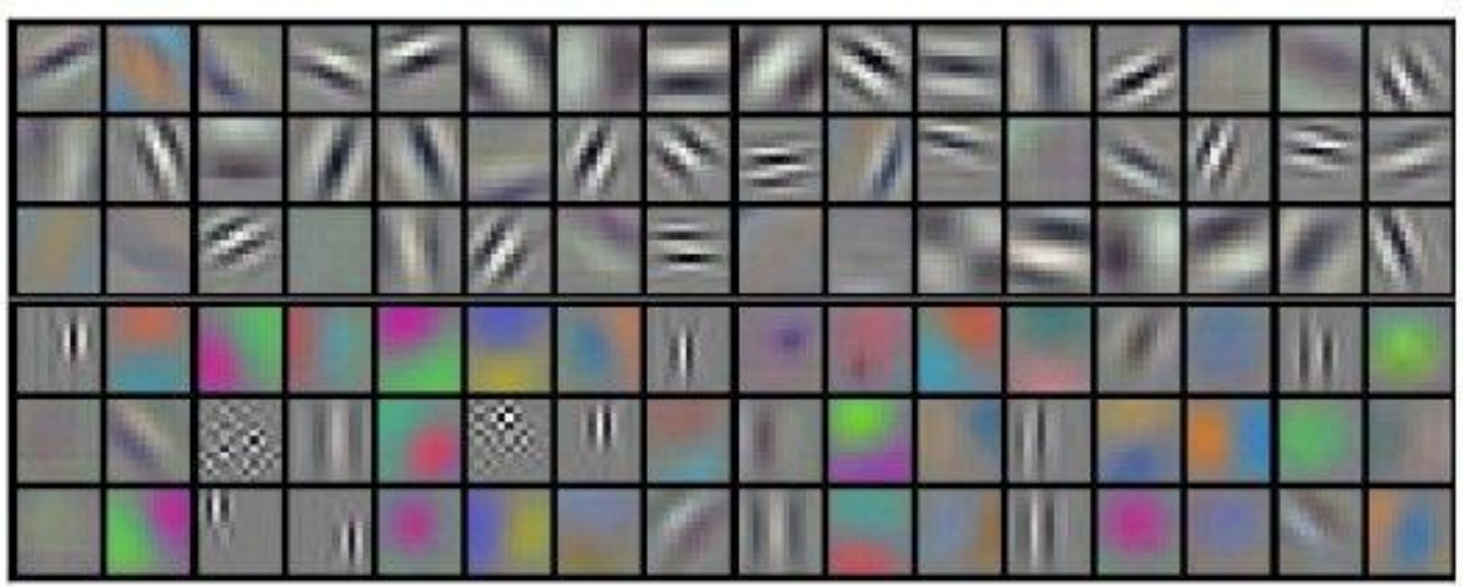
– Imagenet classification with deep convolutional neural networks

– Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton, 2012



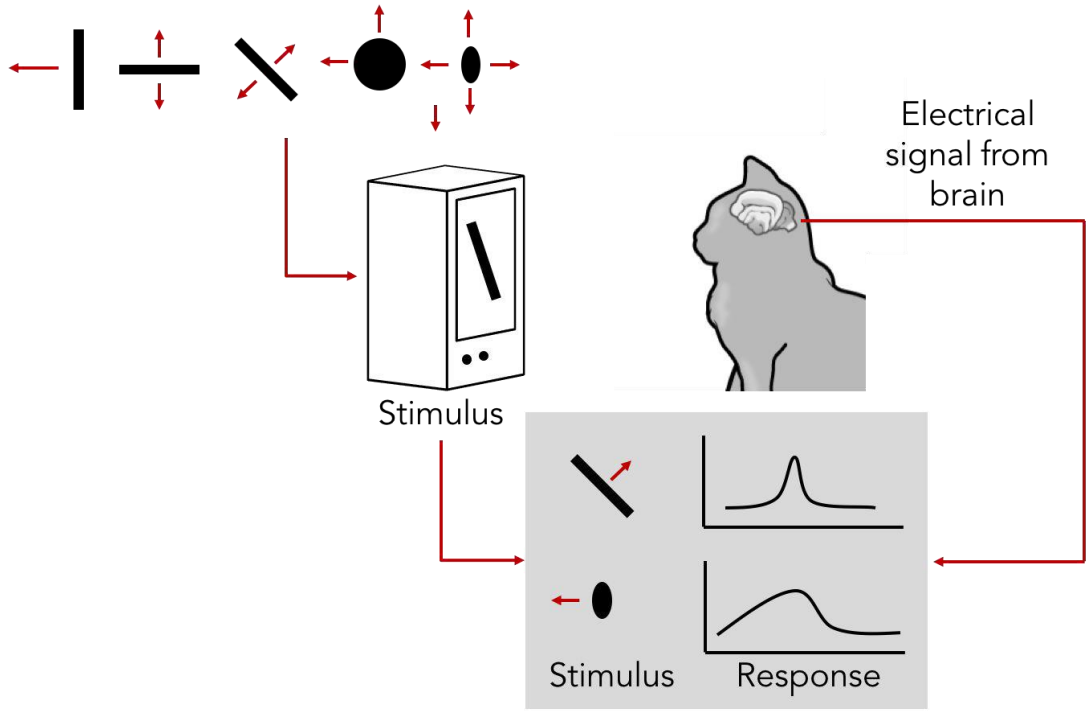
2012, First strong results

- Imagenet classification with deep convolutional neural networks
- Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton, 2012



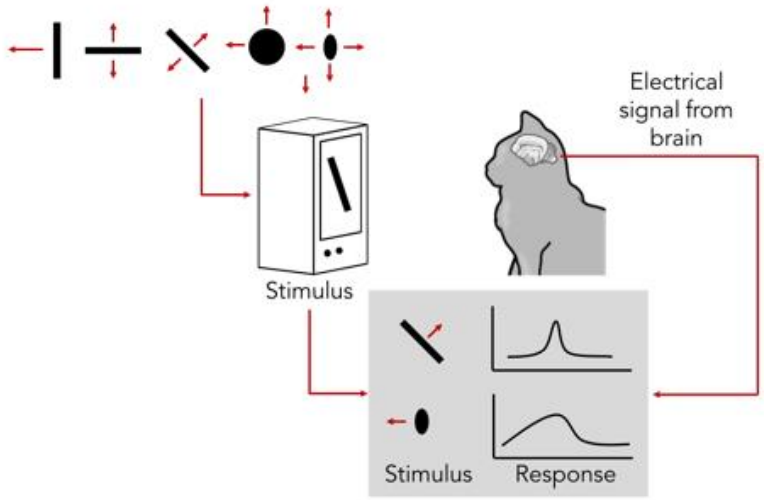
1959, Hubel & Wiesel

- 1959: receptive fields of single neurones in the cat's striate cortex
- 1962: receptive fields, binocular interaction and functional architecture in the cat's visual cortex
- 1968...



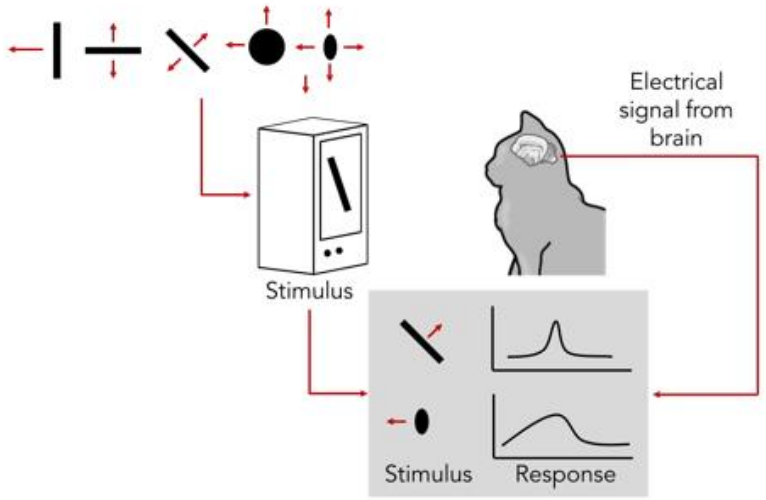
1959, Hubel & Wiesel

- Hubel và Wiesel đã tiến hành các thí nghiệm trên mèo và khỉ để hiểu cách mà tế bào thần kinh trong vỏ não thị giác phản ứng với các kích thích thị giác.



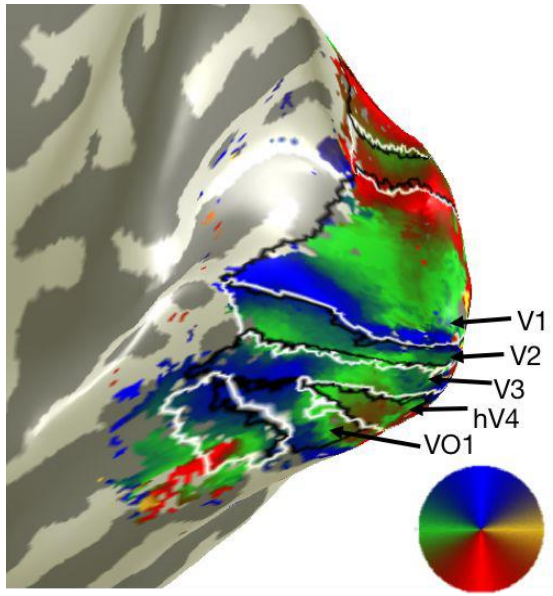
1959, Hubel & Wiesel

- Hubel và Wiesel đã phát hiện ra rằng các tế bào thần kinh trong vỏ não thị giác có những phản ứng đặc trưng đối với các đặc điểm cụ thể của kích thích thị giác như cạnh, góc và chuyển động. Điều này đã dẫn đến việc hiểu rõ hơn về cách hệ thống thị giác của động vật (và con người) xử lý thông tin thị giác.

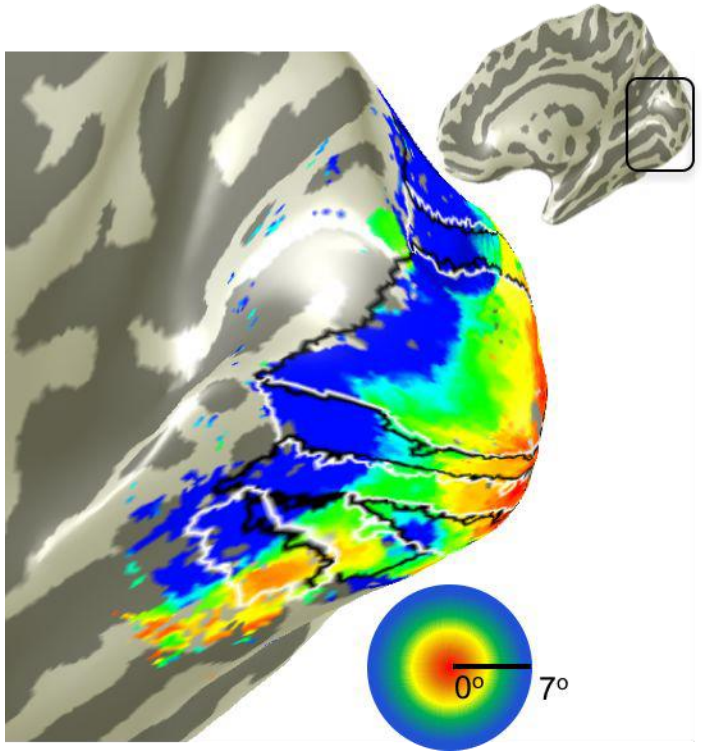


Topographical mapping in the cortex

Topographical mapping in the cortex: nearby cells in cortex represent nearby regions in the visual field

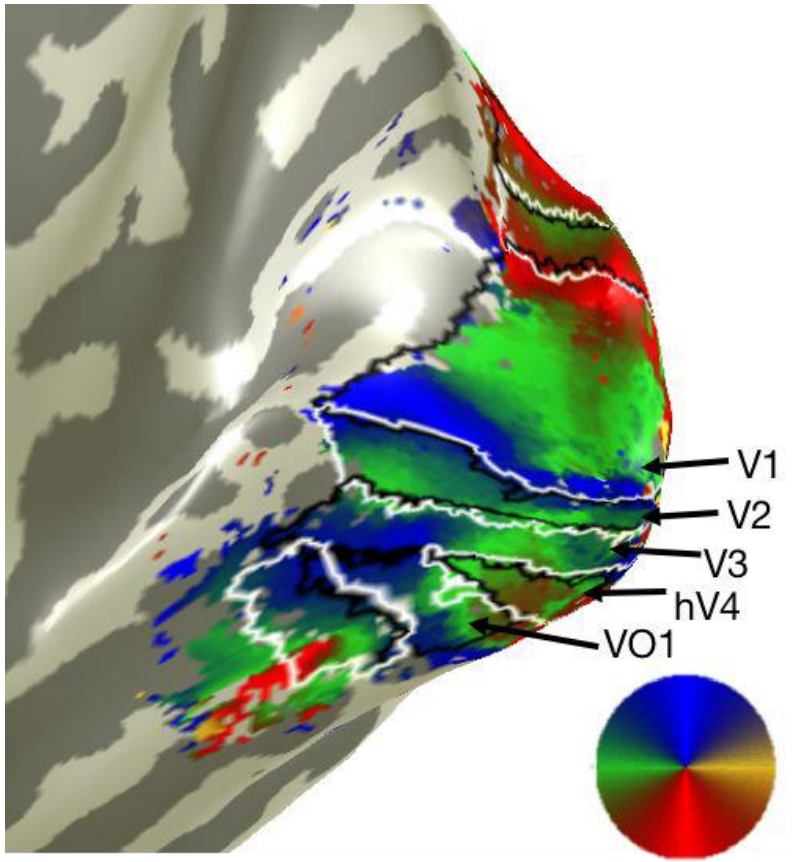


Human brain



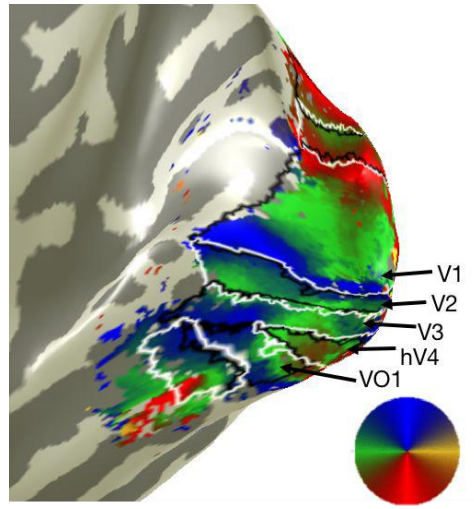
Topographical mapping in the cortex

- Bản đồ thị giác (topographical mapping) trong vỏ não thị giác của con người.
- Đây là cách não bộ tổ chức và xử lý thông tin thị giác từ môi trường xung quanh.
- Trong hình, các vùng khác nhau của vỏ não thị giác được đánh dấu với các ký hiệu $V1, V2, V3, hV4, VO1$.



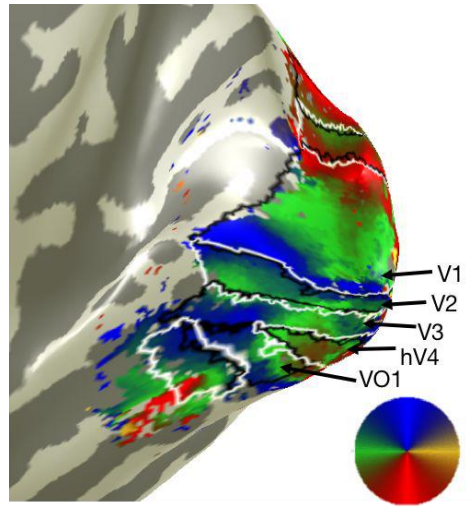
Topographical mapping in the cortex

- V1 (Primary Visual Cortex): Đây là vùng thị giác chính của vỏ não, nơi thông tin thị giác từ mắt được xử lý đầu tiên. V1 chịu trách nhiệm xử lý các đặc trưng cơ bản của hình ảnh như cạnh, đường viền và độ tương phản.
- V2 (Secondary Visual Cortex): Sau khi thông tin được xử lý ở V1, nó được gửi đến V2. V2 tiếp tục phân tích các đặc trưng phức tạp hơn như hình dạng và cấu trúc của đối tượng.



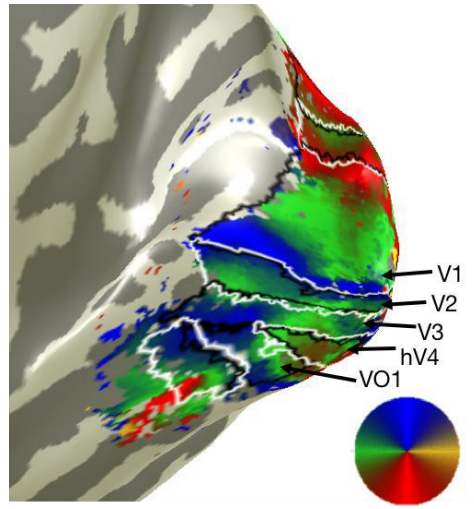
Topographical mapping in the cortex

- V3: Vùng này xử lý thông tin liên quan đến hình dạng và chuyển động của đối tượng. V3 kết nối với các vùng khác để hoàn thiện việc nhận diện đối tượng trong không gian 3 chiều.
- hV4 (Human Visual Area 4): Vùng này chịu trách nhiệm xử lý thông tin màu sắc và hình dạng phức tạp hơn.

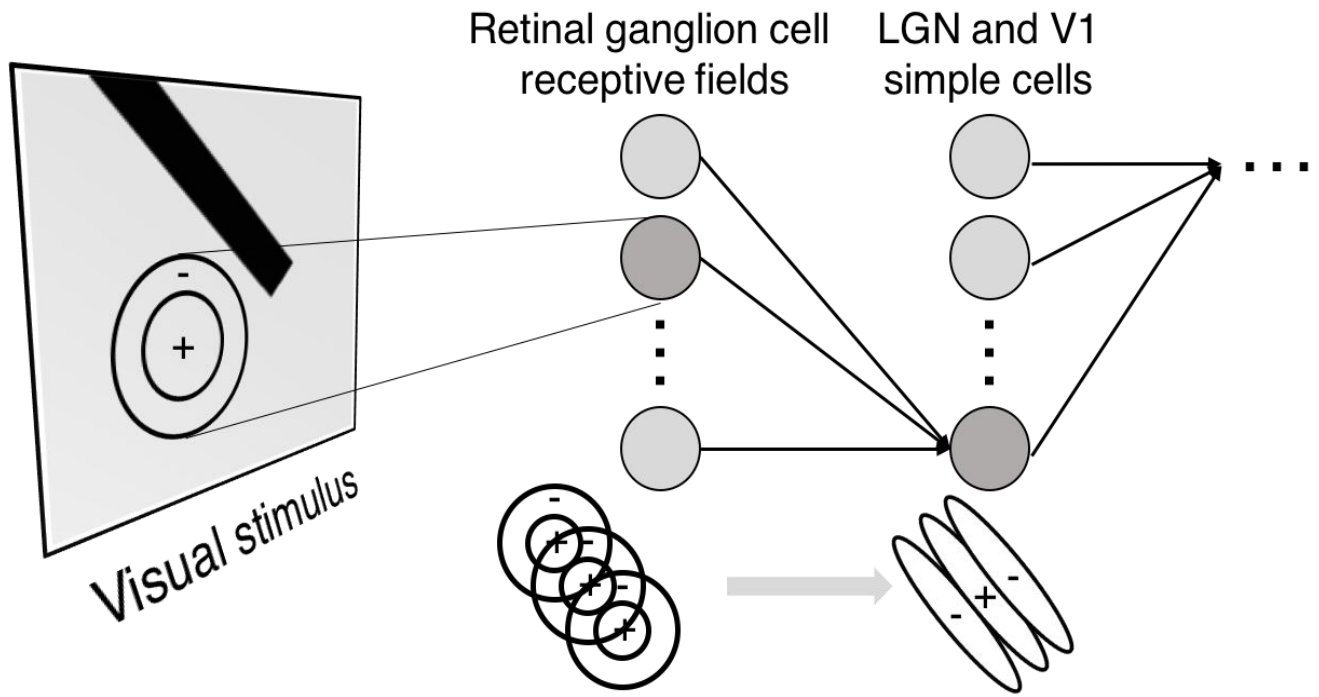


Topographical mapping in the cortex

- VO1 (Ventral Occipital 1): Vùng này nằm ở phần dưới của vỏ não thị giác và có liên quan đến việc nhận diện các đối tượng và cảnh vật trong trường thị giác.



Hierarchical organization



Simple cells:
Response to light orientation

Complex cells:
Response to light orientation and movement

Hypercomplex cells:
response to movement with an end point

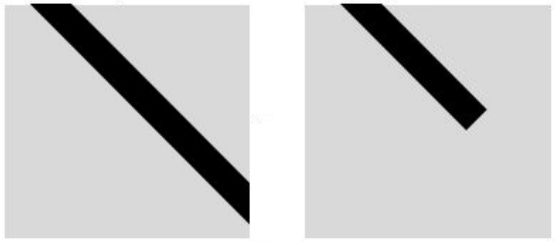
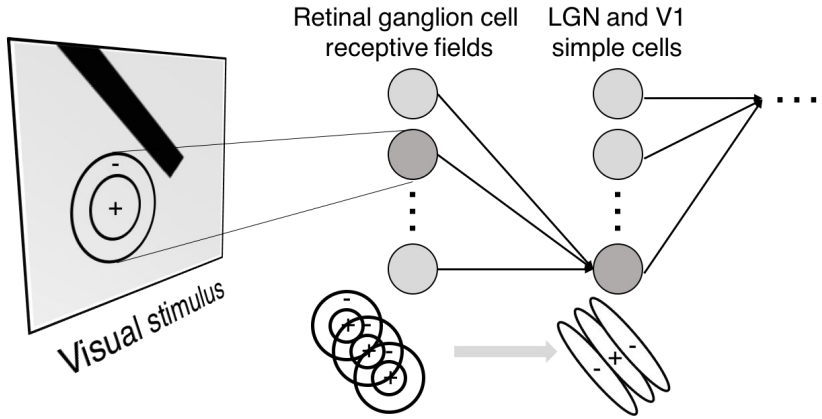


Illustration of hierarchical organization in early visual pathways by Lane McIntosh, copyright CS231n 2017.

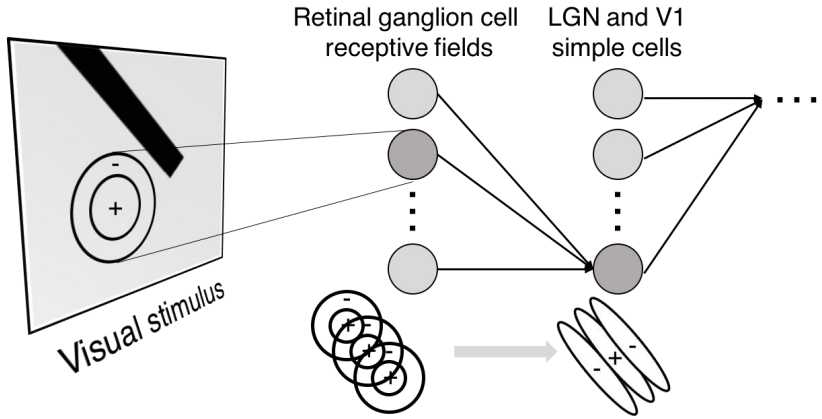
Hierarchical organization

- Hình ảnh mô tả cách các tế bào thần kinh trong vỏ não thị giác phản ứng với các kích thích thị giác, đặc biệt là cách mà các tế bào thần kinh ở V1 (Primary Visual Cortex) phát hiện các cạnh và đường viền trong trường thị giác.



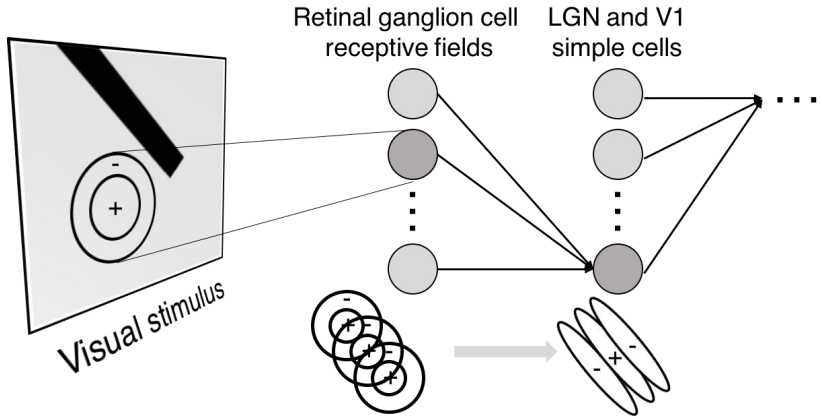
Hierarchical organization

- Hình bên trái cho thấy một kích thích thị giác đơn giản, là một thanh đen nghiêng trên nền trắng.
- Kích thích này được chiếu lên võng mạc.



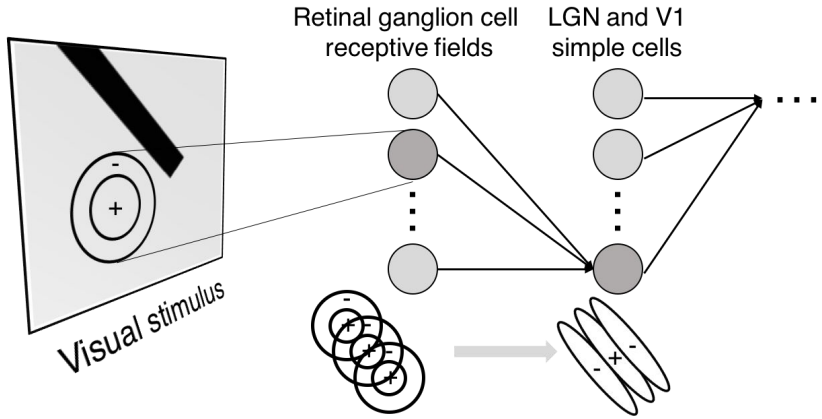
Hierarchical organization

- Các vòng tròn ở giữa hình biểu thị các tế bào thần kinh trong vùng V1 của vỏ não thị giác.
- Mỗi tế bào thần kinh trong V1 có một trường thụ cảm (receptive field) - vùng trong trường thị giác mà một tế bào thần kinh phản ứng mạnh nhất.



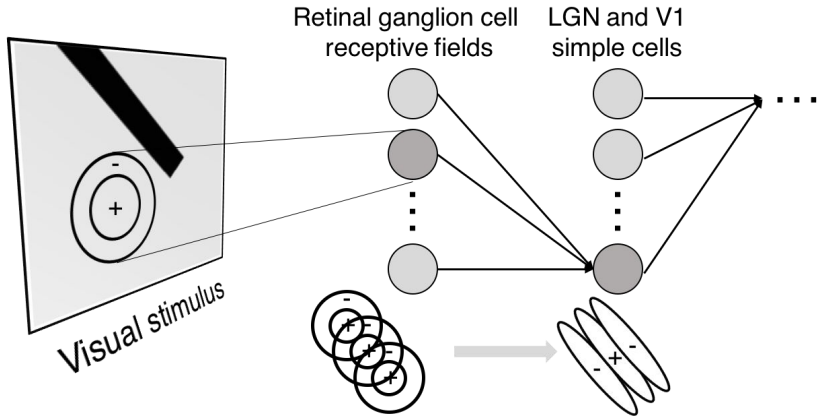
Hierarchical organization

– Trong hình, các vòng tròn sáng và tối thể hiện các tế bào thần kinh có trường thụ cảm khác nhau, với các vùng nhạy cảm (+) và không nhạy cảm (-).



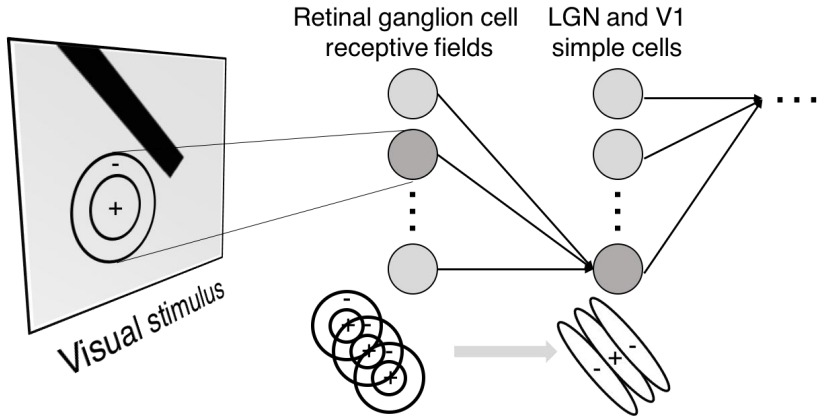
Hierarchical organization

- Khi thanh đen nghiêng, kích thích các tế bào thần kinh trong V1, một số tế bào thần kinh sẽ phản ứng mạnh hơn vì kích thích này phù hợp với trường thụ cảm của chúng.
- Các tế bào thần kinh có trường thụ cảm phù hợp với hướng và vị trí của thanh đen sẽ bị kích hoạt (+), trong khi các tế bào không phù hợp sẽ không phản ứng hoặc có phản ứng yếu hơn (-).



Hierarchical organization

- Ở phía dưới bên phải của hình là một nhóm các tế bào thần kinh với các trường thụ cảm kết hợp (+ và -) tạo thành một mô hình phức tạp hơn.
- Các tế bào thần kinh trong V1 không chỉ phản ứng với các điểm đơn lẻ mà còn với các đường viền và cạnh, tạo ra các mô hình phức tạp hơn khi nhiều tế bào thần kinh được kích hoạt đồng thời.



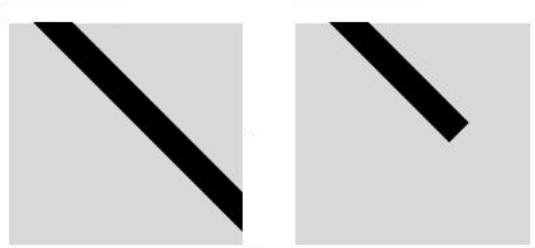
Simple Cells

- Tế bào đơn giản có các trường thụ cảm hình chữ nhật với các vùng kích thích (excitatory) và ức chế (inhibitory) rõ ràng.
- Các tế bào đơn giản có thể phát hiện các cạnh và đường thẳng ở những vị trí và hướng cụ thể trong trường thụ cảm của mình.

Simple cells:
Response to light orientation

Complex cells:
Response to light orientation and movement

Hypercomplex cells:
response to movement with an end point



No response

Response (end point)

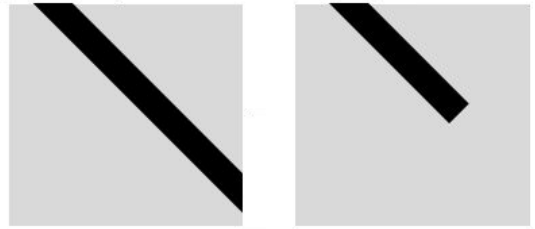
Simple Cells

- Các tế bào này đóng vai trò quan trọng trong việc phát hiện các đặc điểm cơ bản của hình ảnh như cạnh, đường biên và hướng của các đường thẳng.
- Ví dụ, một tế bào đơn giản có thể phản ứng mạnh mẽ khi xuất hiện một cạnh sáng theo chiều ngang trong trường thụ cảm của nó.

Simple cells:
Response to light orientation

Complex cells:
Response to light orientation and movement

Hypercomplex cells:
response to movement with an end point



No response

Response (end point)

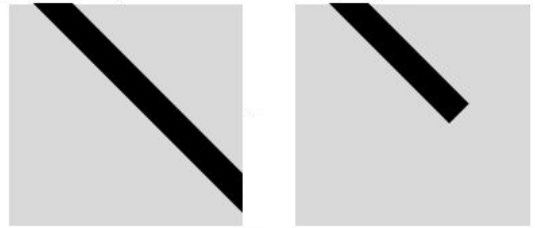
Complex Cells

- Tế bào phức tạp có các trường thụ cảm lớn hơn và không có các vùng kích thích và ức chế rõ ràng như tế bào đơn giản.
- Tế bào phức tạp có khả năng phát hiện các đặc điểm phức tạp hơn, chẳng hạn như các cạnh và đường thẳng di chuyển trong một vùng rộng.

Simple cells:
Response to light orientation

Complex cells:
Response to light orientation and movement

Hypercomplex cells:
response to movement with an end point



No response

Response (end point)

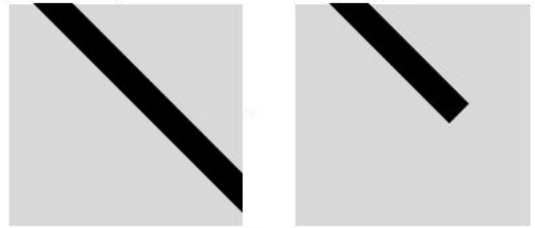
Hypercomplex Cells

- Tế bào siêu phức tạp có các trường thụ cảm lớn hơn và có khả năng phản ứng với các đặc điểm như chiều dài, góc cạnh hoặc các đường thẳng kết thúc tại một điểm cụ thể.
- Tế bào siêu phức tạp nhạy cảm với các đặc điểm cao cấp hơn như các đoạn thẳng có chiều dài giới hạn hoặc các góc cạnh cụ thể.

Simple cells:
Response to light orientation

Complex cells:
Response to light orientation and movement

Hypercomplex cells:
response to movement with an end point

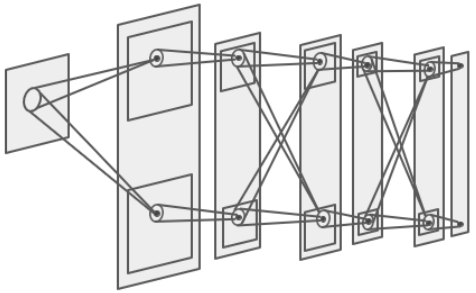


No response

Response (end point)

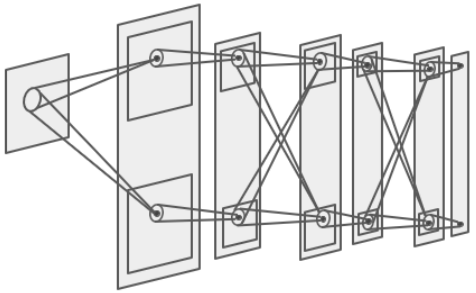
1980, Neocognitron

- Neocognitron là một mạng nơ-ron nhân tạo được phát triển bởi Kunihiko Fukushima vào năm 1980.
- Neocognitron là một trong những mô hình tiên phong trong lĩnh vực học sâu (deep learning) và nhận diện mẫu (pattern recognition).
- Neocognitron được thiết kế để mô phỏng quá trình xử lý hình ảnh của hệ thống thị giác sinh học, đặc biệt là cách mà não người nhận diện các đối tượng.



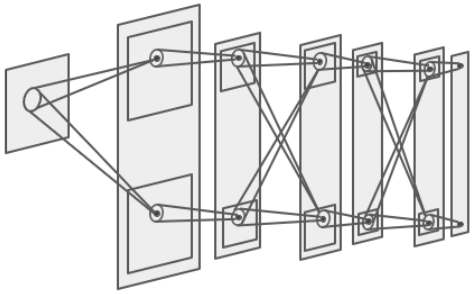
1980, Neocognitron

- Neocognitron [Fukushima 1980].
- Neocognitron có một cấu trúc phân cấp gồm nhiều lớp, mỗi lớp bao gồm các tế bào nơ-ron.
- Simple cells: modifiable parameters
- Tế bào S chịu trách nhiệm trích xuất các đặc trưng cơ bản từ đầu vào.
- Complex cells: perform pooling.
- Tế bào C tổng hợp các đặc trưng từ tế bào S để nhận diện các đối tượng phức tạp hơn.



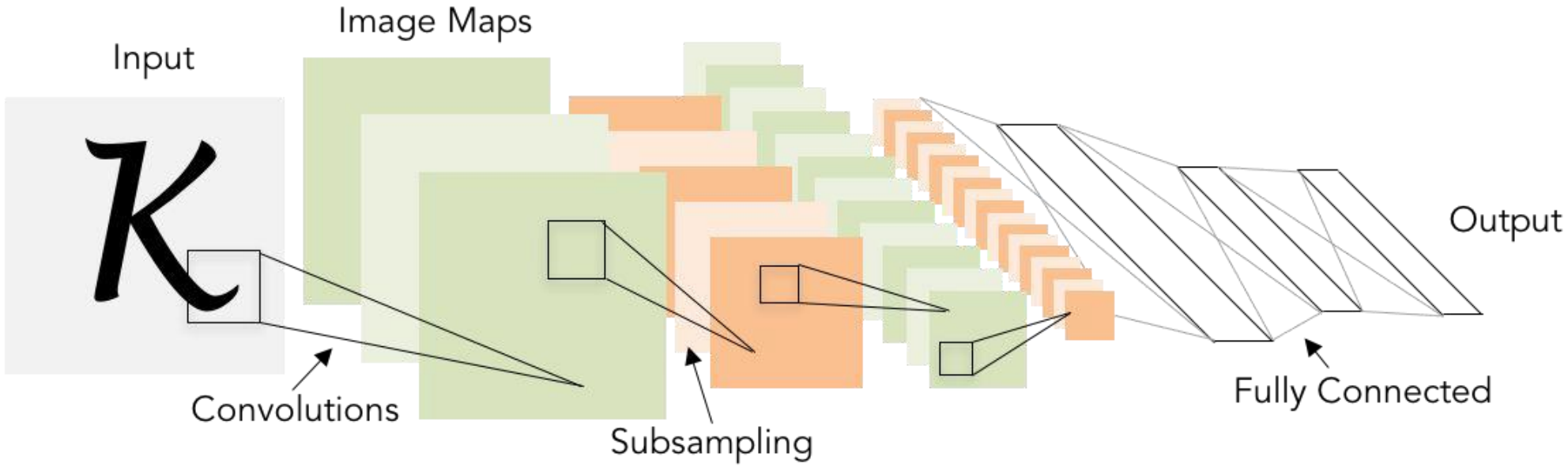
1980, Neocognitron

- Neocognitron đã được áp dụng thành công trong các bài toán nhận diện hình ảnh, đặc biệt là nhận diện ký tự viết tay.
- Mô hình Neocognitron đã đặt nền móng cho sự phát triển của các mạng nơ-ron tích chập (Convolutional Neural Networks - CNNs) sau này, như mạng LeNet của Yann LeCun.



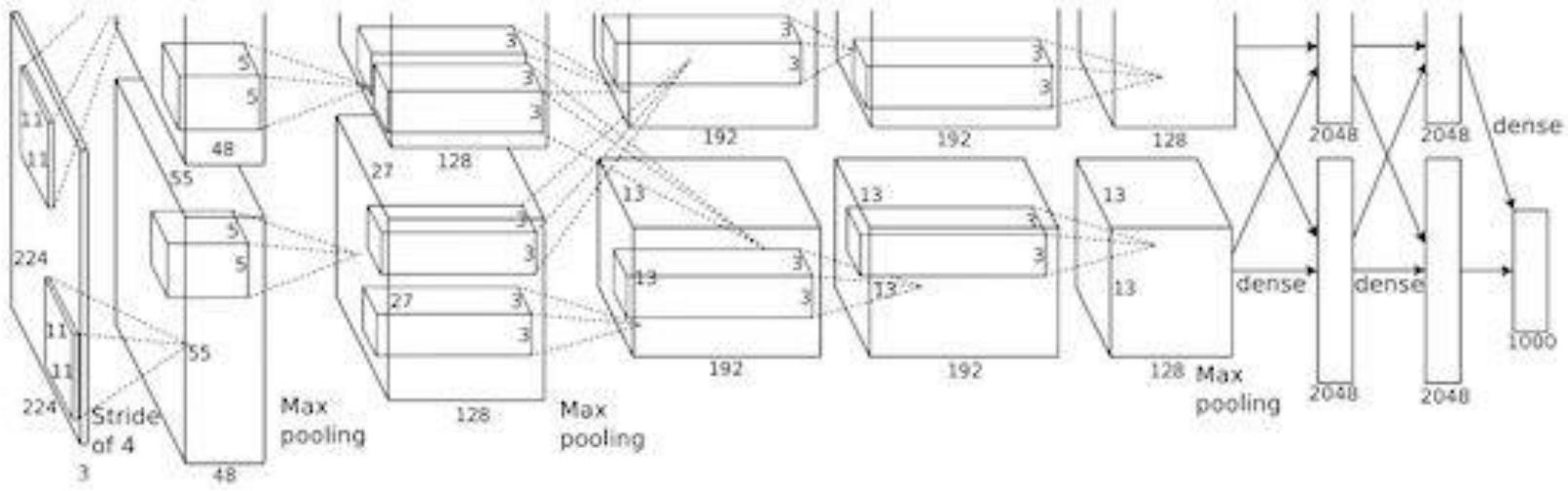
1998, LeCun – LeNet-5 Model

- Gradient-based learning applied to document recognition [LeCun, Bottou, Bengio, Haffner 1998].
- Hình ảnh mô tả kiến trúc của một mạng nơ-ron tích chập.



2012, AlexNet

- ImageNet Classification with Deep Convolutional Neural Networks [Krizhevsky, Sutskever, Hinton, 2012].
- Hình ảnh mô tả kiến trúc của một mạng nơ-ron tích chập (Convolutional Neural Network - CNN) phức tạp – Alexnet.



ConvNets are everywhere

Whaler ecognition

Mnih and Hinton, 2010

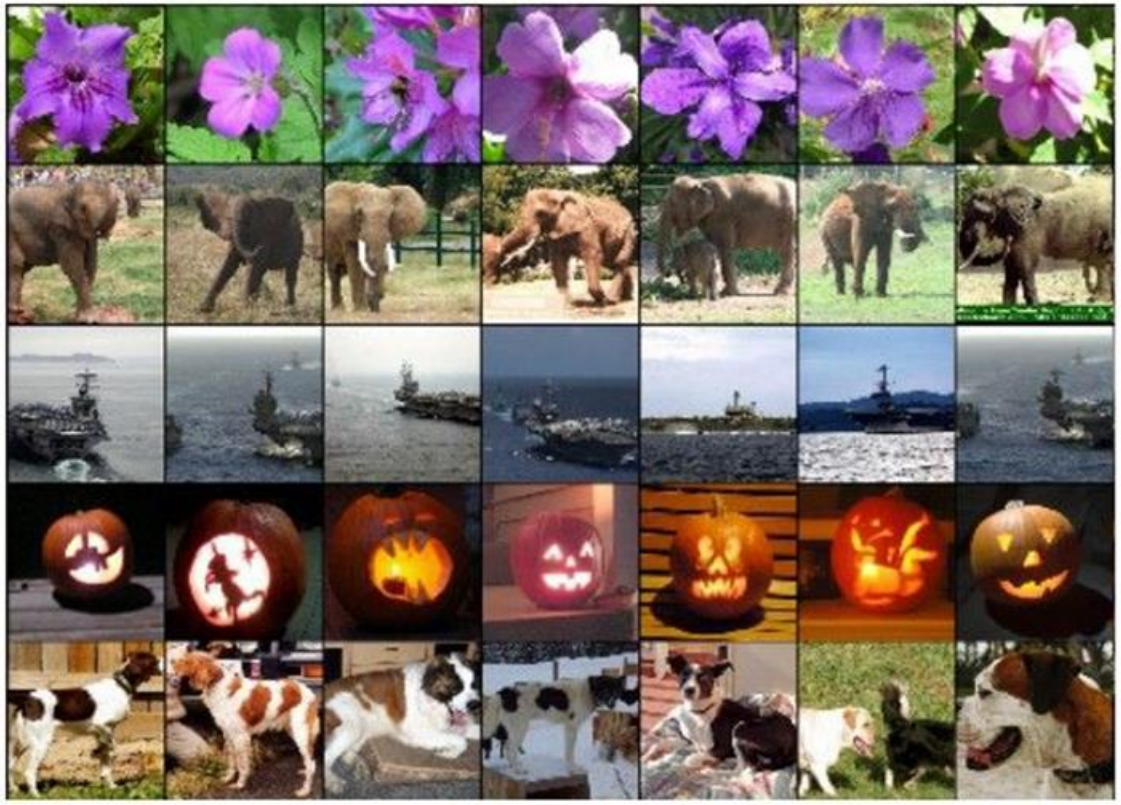
ConvNets are everywhere

- *[Sermanet et al. 2011] [Ciresan et al.]*
- Một trong những nghiên cứu nổi bật của *Sermanet* vào năm 2011 là sử dụng mạng nơ-ron tích chập (Convolutional Neural Networks - CNNs) để nhận diện hình ảnh.

ConvNets are everywhere

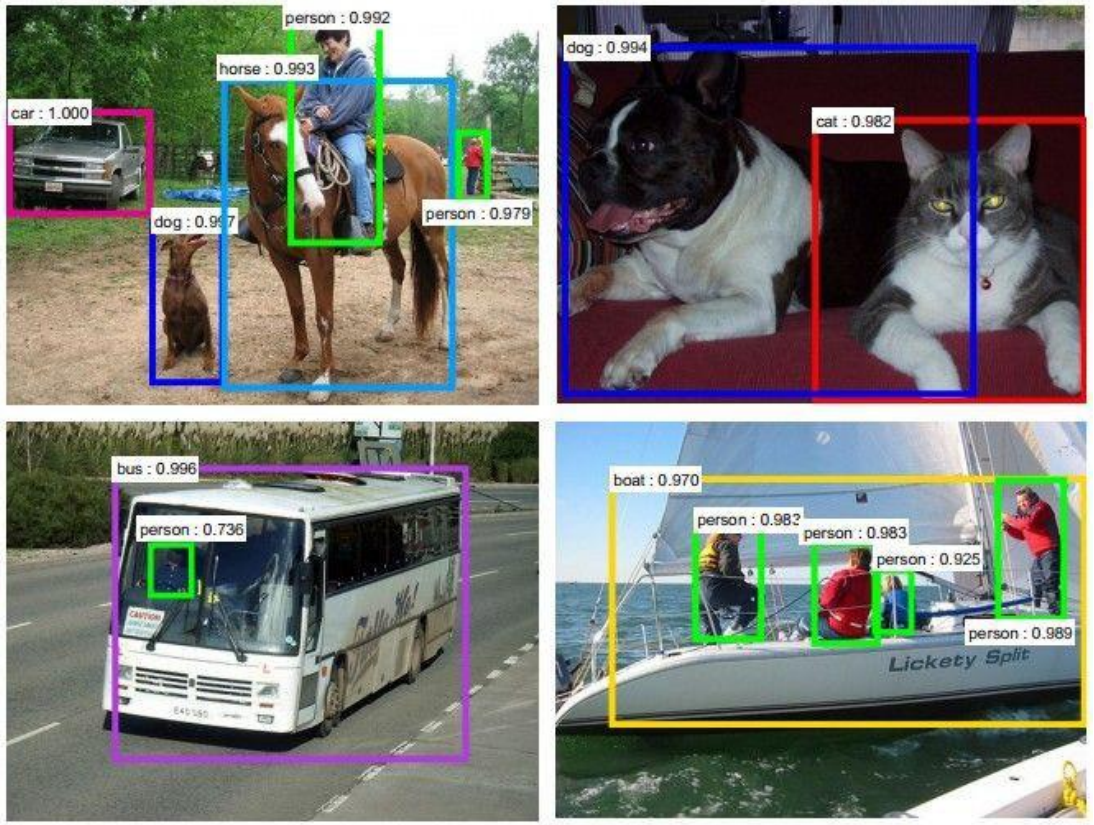
2012, Classification

2012, Retrieval

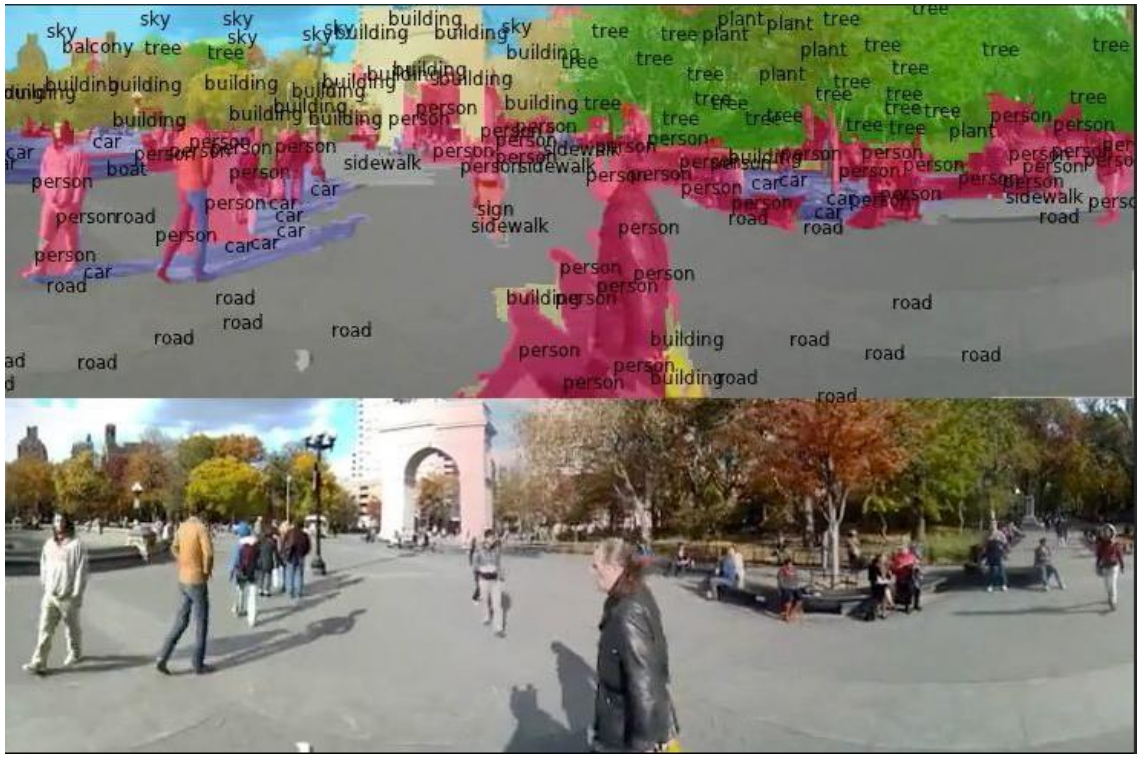


ConvNets are everywhere

2015, Detection

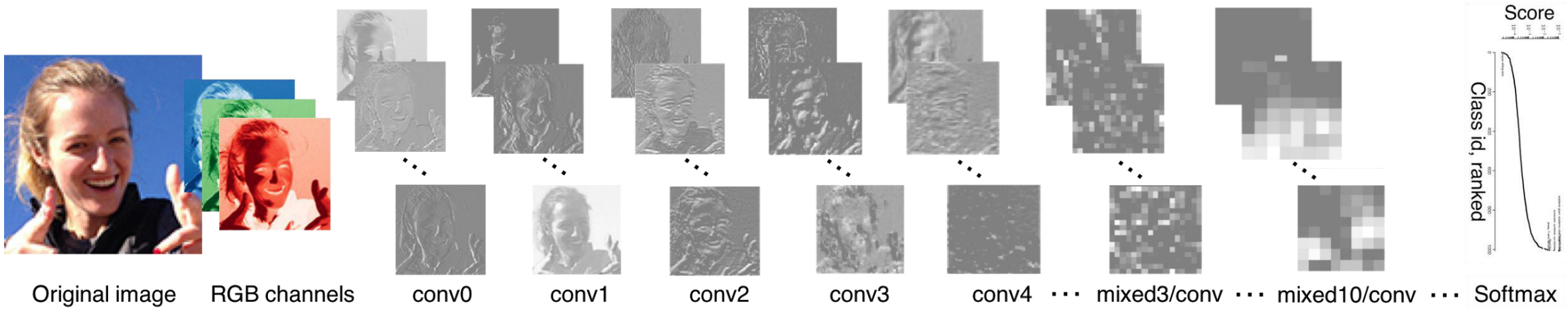


2012, Segmentation



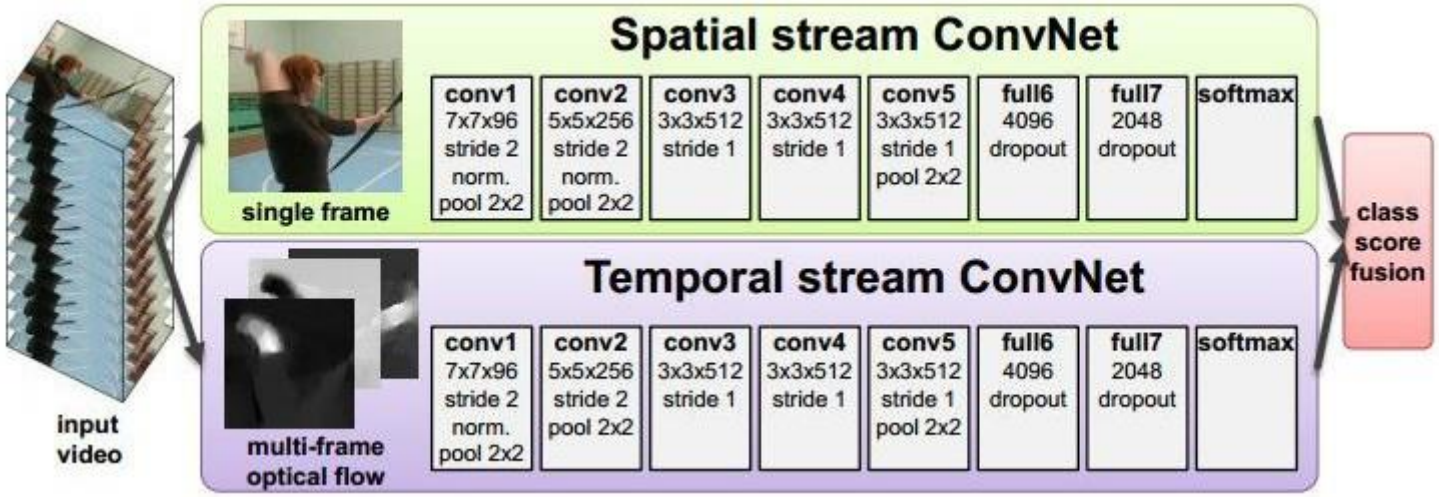
ConvNets are everywhere

– Năm 2014, Yaniv Taigman và các đồng nghiệp tại Facebook AI Research đã công bố một bài báo quan trọng giới thiệu hệ thống DeepFace.



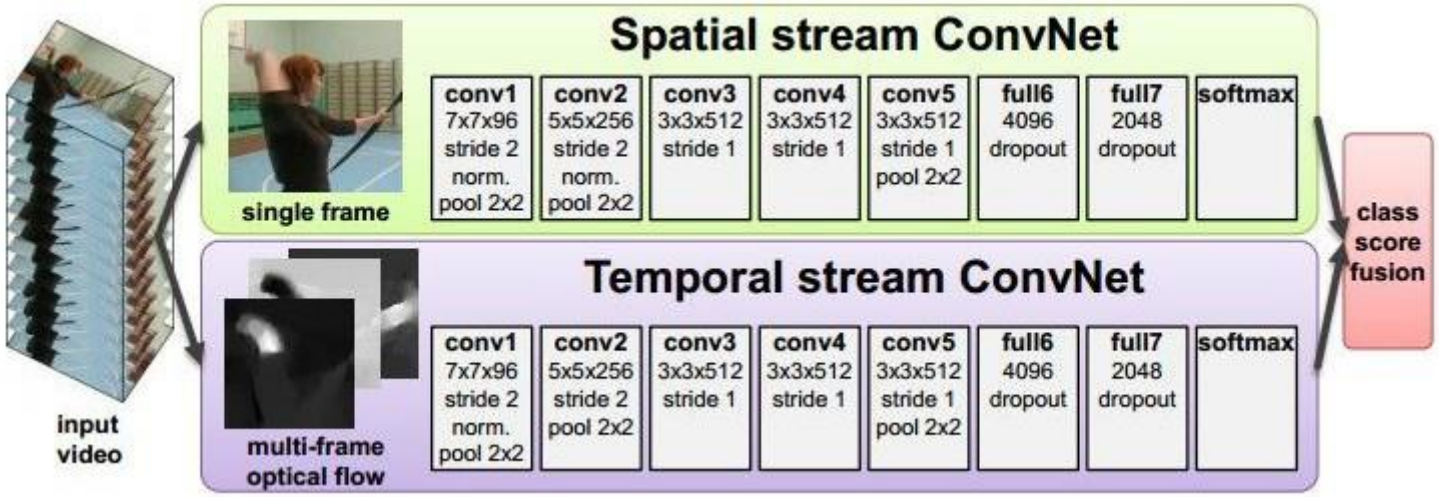
ConvNets are everywhere

– Simonyan et al, 2014. Video analysis using spatiotemporal convolutional networks.



ConvNets are everywhere

– Hình ảnh minh họa kiến trúc của một mô hình phân tích video sử dụng mạng nơ-ron tích chập không gian-thời gian (spatiotemporal convolutional networks). Bao gồm hai luồng ConvNet: luồng không gian (spatial stream) và luồng thời gian (temporal stream).

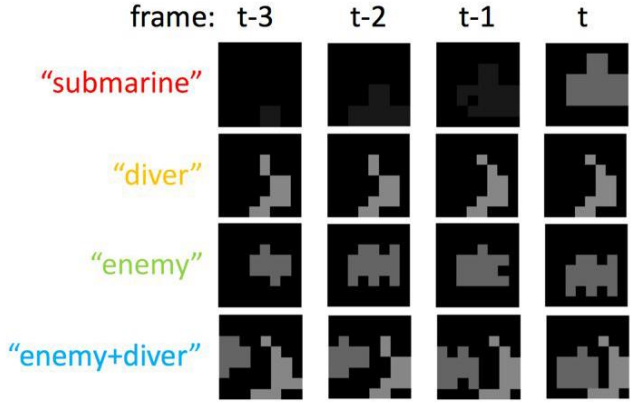


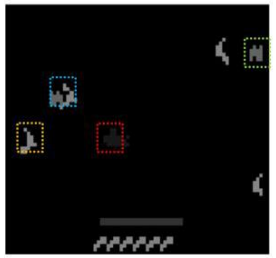
ConvNets are everywhere

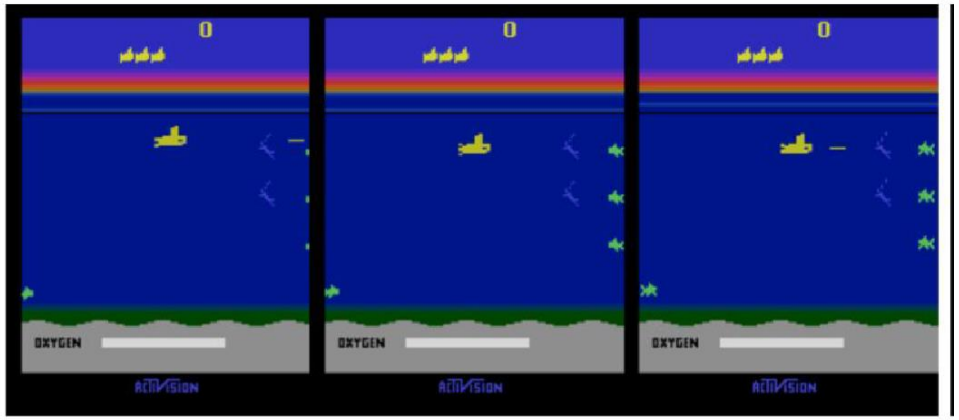
- Toshev, Szegedy, 2014.
- Hình ảnh minh họa kết quả của phương pháp nhận diện tư thế cơ thể người.

ConvNets are everywhere

- Guo, et al. 2014
- Hình ảnh minh họa việc sử dụng mạng nơ-ron tích chập để nhận diện và phân loại các đối tượng trong trò chơi video.







ConvNets are everywhere

- [Dieleman et al. 2014]
- Công trình của Dieleman năm 2014 liên quan đến việc sử dụng học sâu (deep learning) để phân loại các thiên hà trong các hình ảnh thiên văn học.

2015, Image Captioning

- [Vinyals et al., 2015] [Karpathy and Fei-Fei, 2015]

- *No errors*
  - + *A white teddy bear sitting in the grass.*
  - + Một con gấu bông màu trắng ngồi trên bãi cỏ.

2015, Image Captioning

- [Vinyals et al., 2015] [Karpathy and Fei-Fei, 2015]

- *No errors*
  - + A man riding a wave on top of a surfboard.
  - + Một người đàn ông cưỡi sóng trên ván lướt song.

2015, Image Captioning

- [Vinyals et al., 2015] [Karpathy and Fei-Fei, 2015]

- *Minor errors*
  - + A man in a baseball uniform throwing a ball.
  - + Một người đàn ông mặc đồng phục bóng chày đang ném bóng.

2015, Image Captioning

- [Vinyals et al., 2015] [Karpathy and Fei-Fei, 2015]

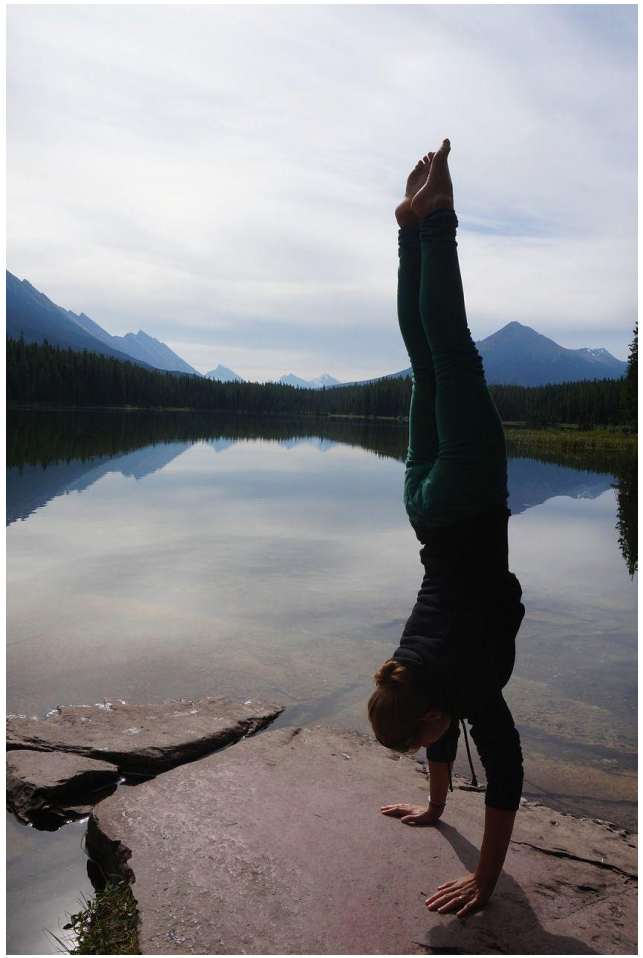
- *Minor errors*
  - + A cat sitting on a suitcase on the floor.
  - + Một con mèo ngồi trên chiếc vali trên sàn nhà.

2015, Image Captioning

- [Vinyals et al., 2015] [Karpathy and Fei-Fei, 2015]

- *Somewhat related*
  - + A woman is holding a cat in her hand.
  - + Một người phụ nữ đang ôm một con mèo trên tay.

2015, Image Captioning

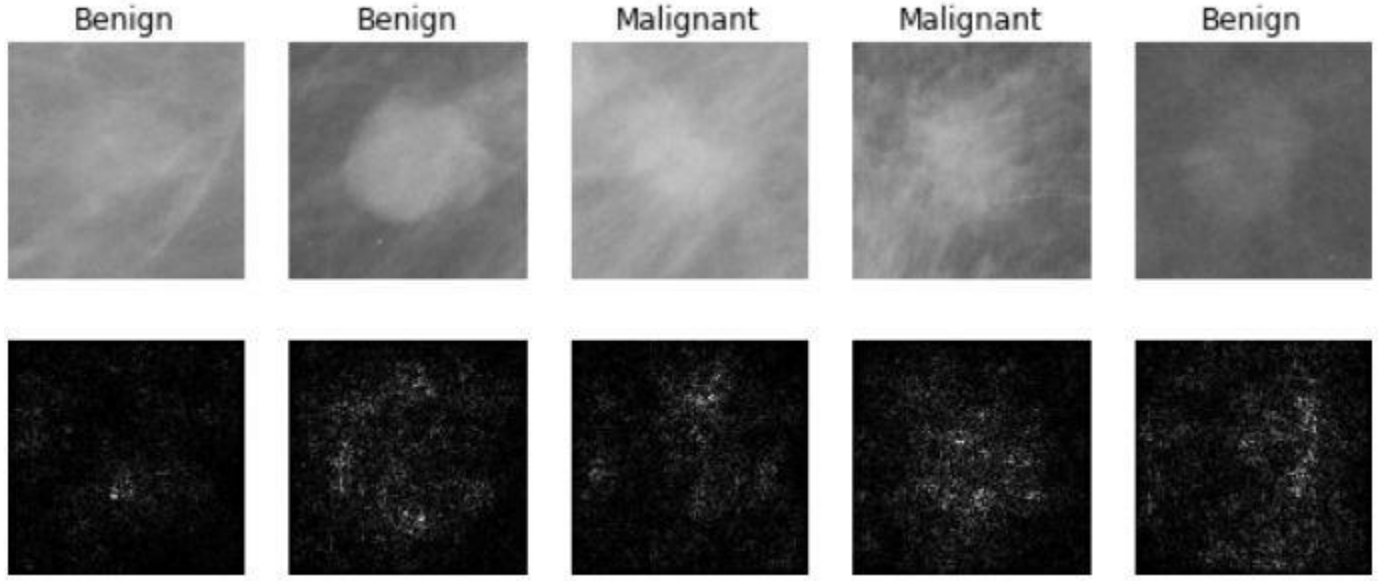


- [Vinyals et al., 2015] [Karpathy and Fei-Fei, 2015]

- *Somewhat related*
  - + A woman standing on a beach holding a surfboard.
  - + Một người phụ nữ đứng trên bãi biển cầm ván lướt sóng.

ConvNets are everywhere

– Nghiên cứu của Levy về việc sử dụng mạng nơ-ron tích chập để phân loại hình ảnh y tế. Đặc biệt trong việc phân biệt giữa các khối u lành tính (benign) và ác tính (malignant) trong ảnh chụp X-quang vú.



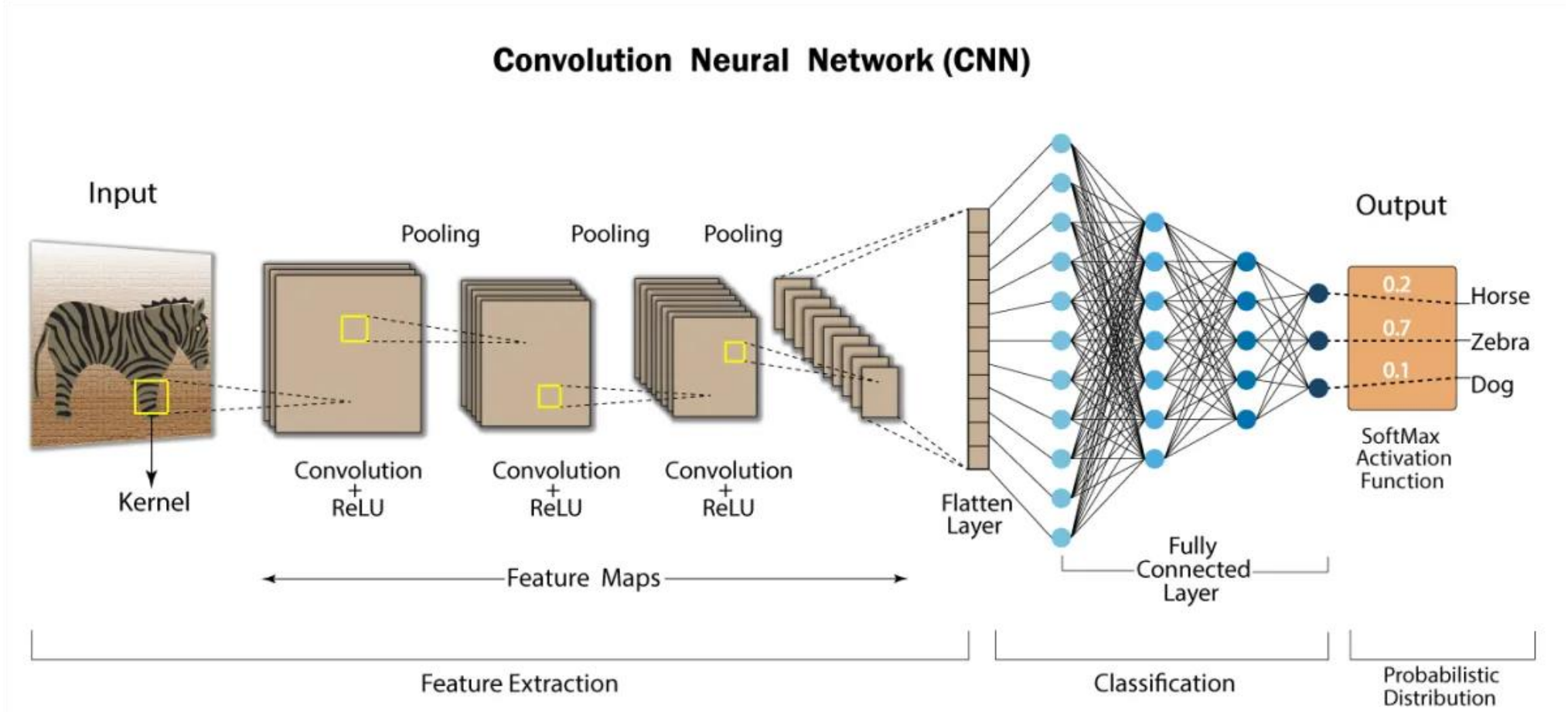
ConvNets are everywhere

2017, Self-driving cars

NVIDIA Tesla line

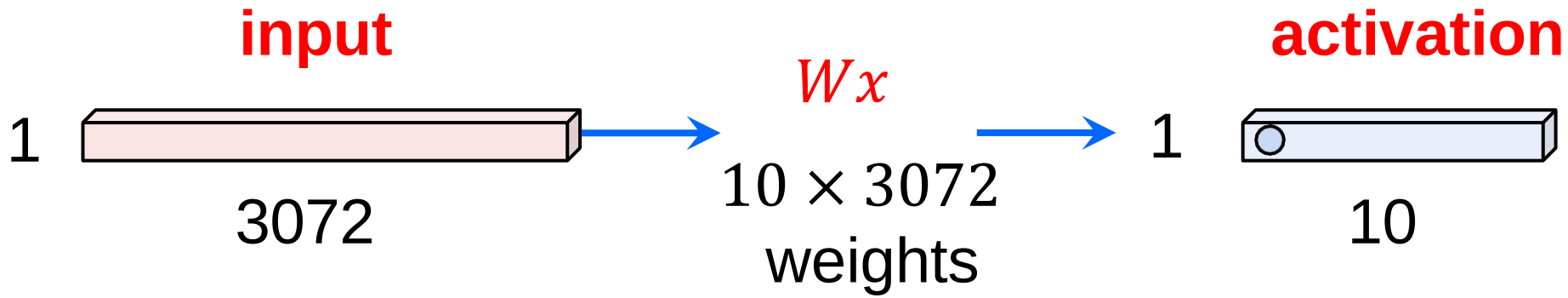
CONVOLUTIONAL NEURAL NETWORKS

Convolutional neural networks



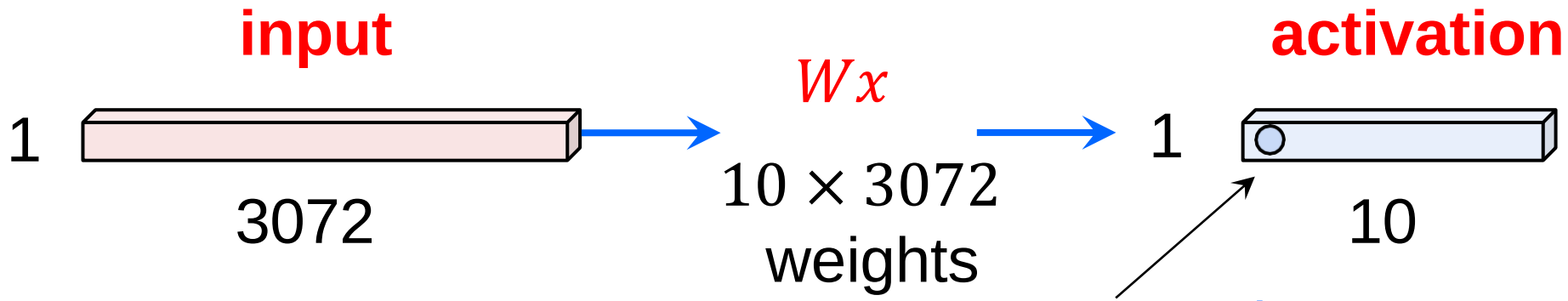
Fully Connected Layer

– Image: $32 \times 32 \times 3 \Longrightarrow$ stretch to 3072×1.



Fully Connected Layer

– Image: $32 \times 32 \times 3 \Longrightarrow$ stretch to 3072×1.



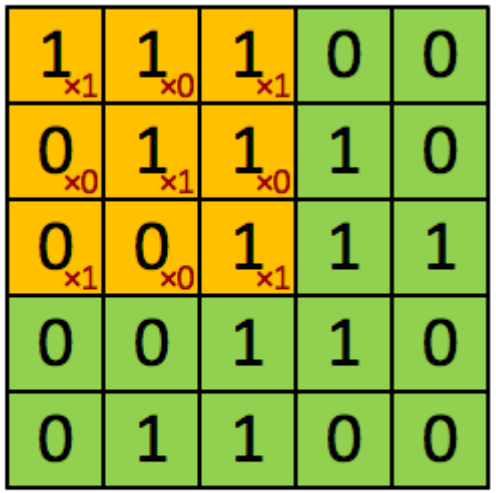
1 number:
the result of taking a dot product between a row of W and the input (a 3072 - dimensional dot product)

Convolution

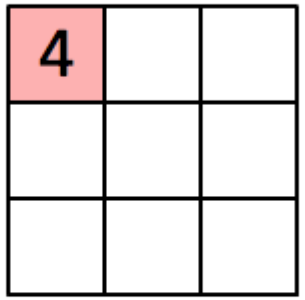
- Convolution là một phép toán toán học kết hợp hai hàm số để tạo ra một hàm số thứ ba.
- Convolution biểu thị sự chồng lấp của hai hàm số khi một hàm được dịch chuyển qua hàm kia.
- Công thức tổng quát của convolution $f * g$ của hai hàm số f và g được định nghĩa như sau:

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau) g(t - \tau)\, d\tau$$

Convolution



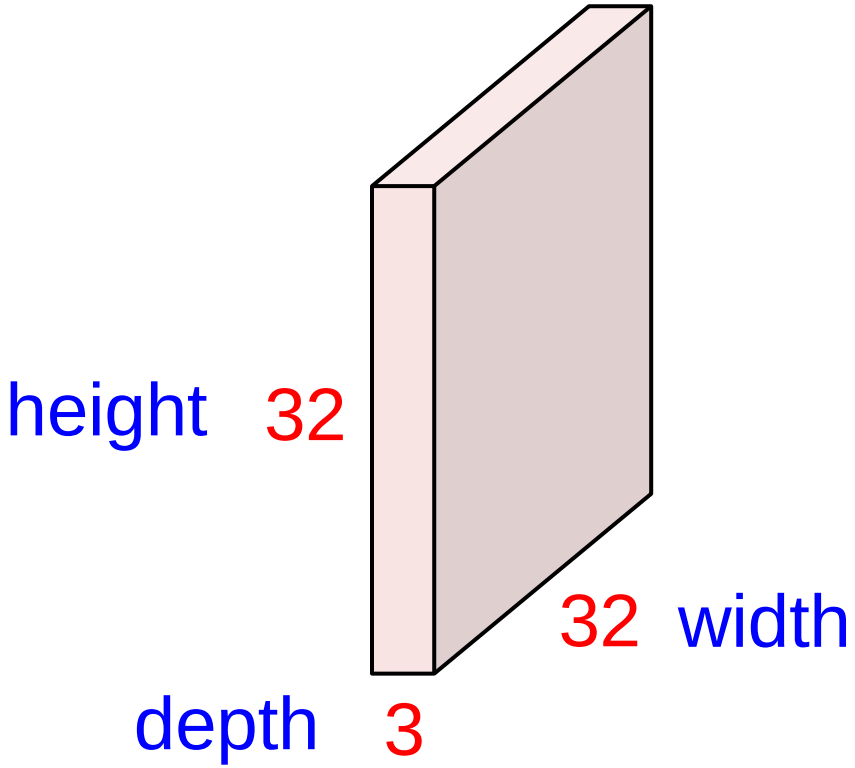
Image



Convolved Feature

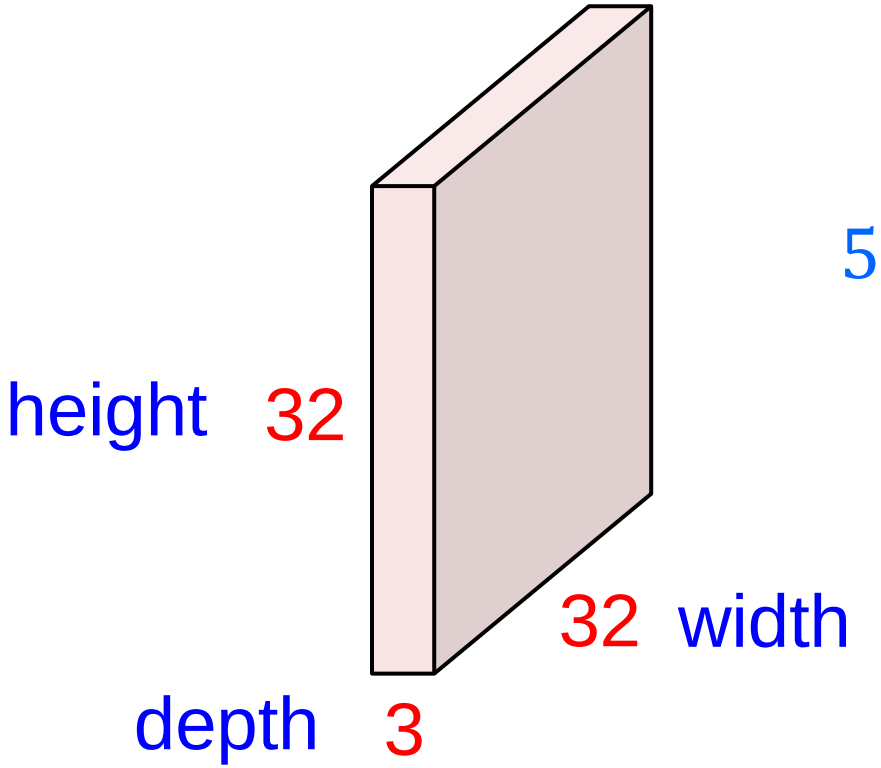
Convolution Layer

– Image: $32 \times 32 \times 3 \Longrightarrow$ preserve spatial structure.

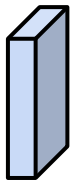


Convolution Layer

– Image: $32 \times 32 \times \mathbf{3}$



$5 \times 5 \times \mathbf{3}$ filter



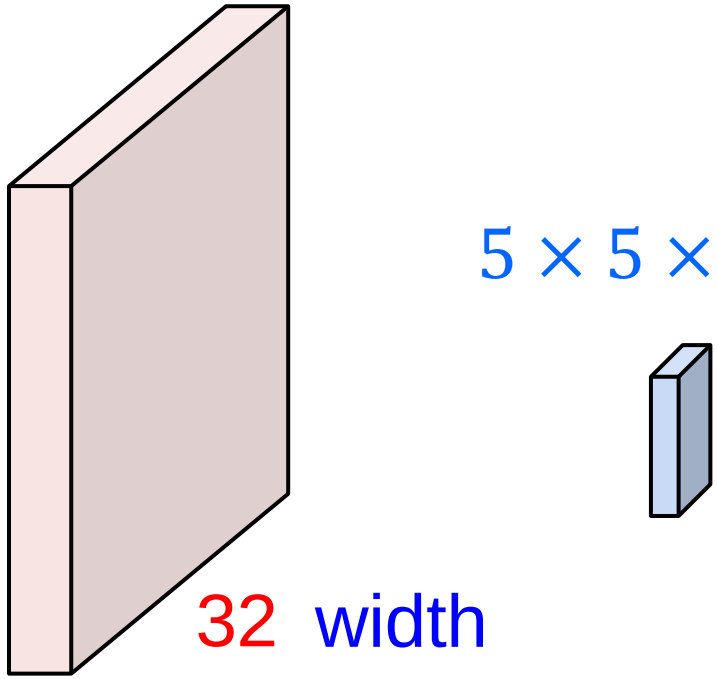
Convolve the filter with the image i.e. ”slide over the image spatially, computing dot products”

Convolution Layer

– Image: $32 \times 32 \times \mathbf{3}$

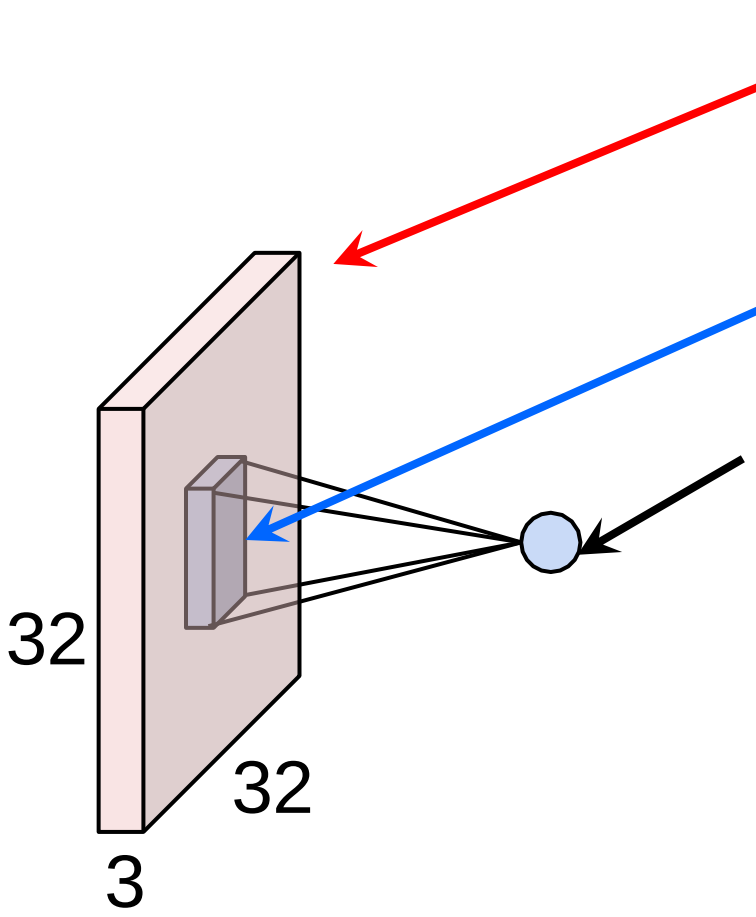
Filters always extend the full depth of the input volume

$5 \times 5 \times \mathbf{3}$ filter



Convolve the filter with the image i.e. ”slide over the image spatially, computing dot products”

Convolution Layer

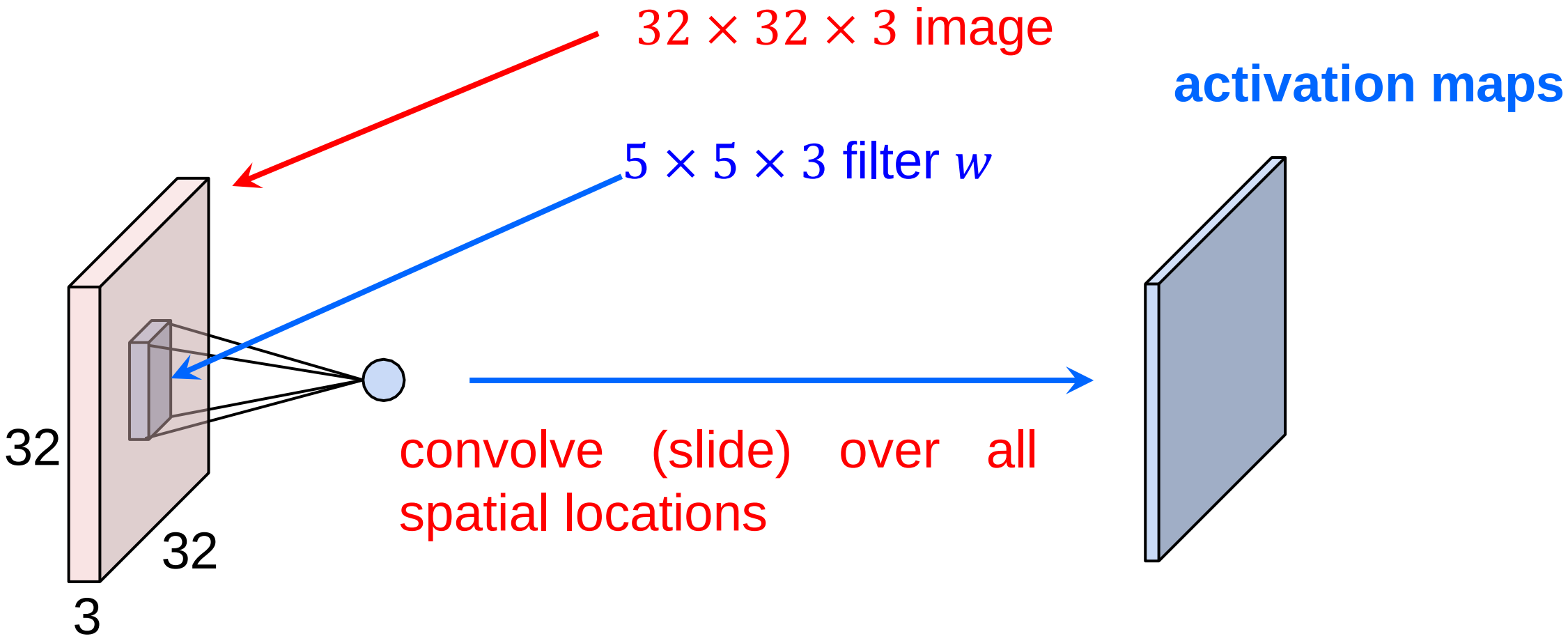


1 number:

the result of taking a dot product between the filter and a small $5 \times 5 \times 3$ chunk of the image (i.e. $5 \times 5 \times 3 = 75 - dimensional$ dot product + bias)

$$w^T x + b$$

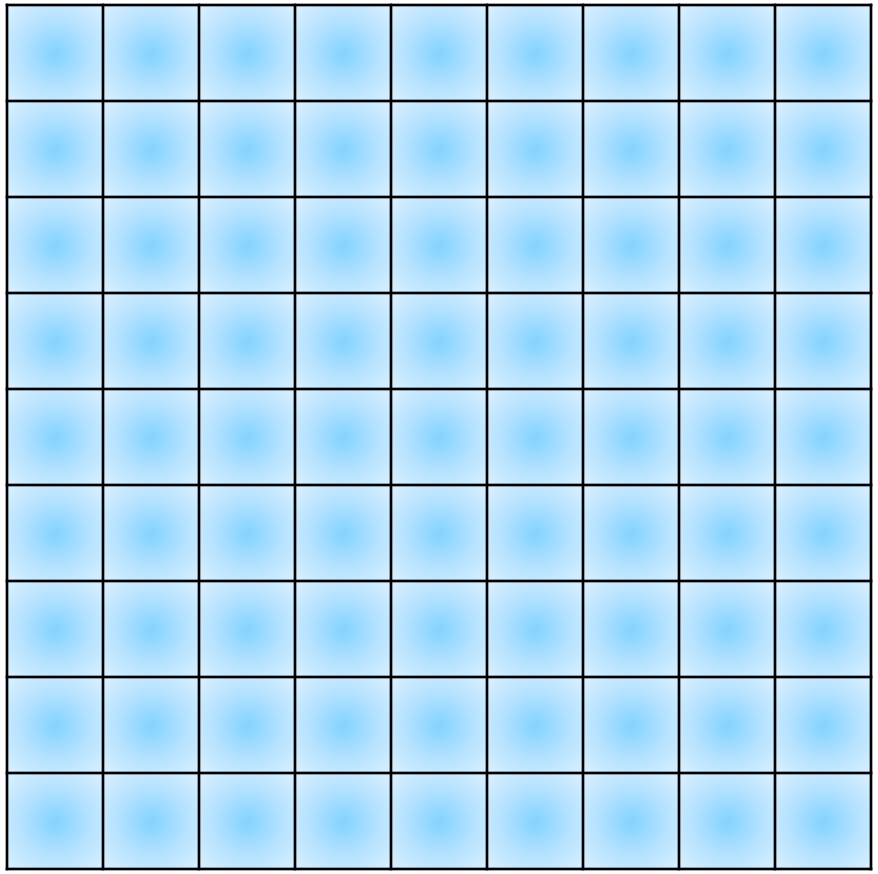
Convolution Layer



Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1



Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

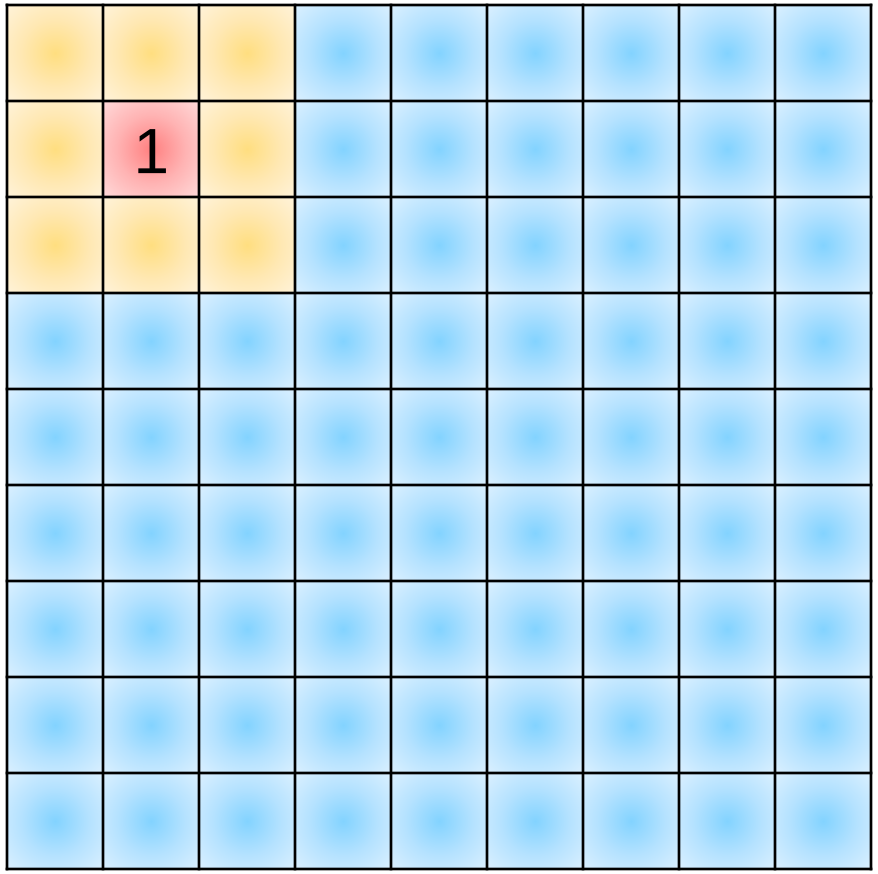
1×1	1×0	0×-1
0×1	0×0	1×-1
1×1	1×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1



Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	0×-1
0×1	1×0	1×-1
1×1	0×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1						

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	0×0	0×-1
1×1	1×0	1×-1
0×1	0×0	0×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0					

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	0×0	1×-1
1×1	1×0	0×-1
0×1	0×0	1×-1

- Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1				

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

0×1	1×0	1×-1
1×1	0×0	0×-1
0×1	1×0	1×-1

– Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1			

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	1×0	0×-1
0×1	0×0	1×-1
1×1	1×0	1×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0		

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

1×1	0×0	1×-1
0×1	1×0	1×-1
1×1	1×0	0×-1

– Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	0×0	1×-1
1×1	1×0	0×-1
0×1	1×0	1×-1

- Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1							

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	1×-1
1×1	0×0	0×-1
1×1	1×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1						

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	1×0	1×-1
0×1	0×0	0×-1
1×1	0×0	1×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0					

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	1×0	0×-1
0×1	0×0	1×-1
0×1	1×0	0×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0				

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	0×-1
0×1	1×0	1×-1
1×1	0×0	1×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0			

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

0×1	0×0	1×-1
1×1	1×0	1×-1
0×1	1×0	0×-1

– Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1		

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	1×-1
1×1	1×0	0×-1
1×1	0×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	1×0	0×-1
0×1	1×0	1×-1
1×1	0×0	1×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0							

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	0×-1
1×1	1×0	0×-1
0×1	1×0	1×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1						

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	0×0	0×-1
1×1	0×0	1×-1
1×1	1×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1					

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	0×0	1×-1
0×1	1×0	0×-1
1×1	0×0	0×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0				

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	1×-1
1×1	0×0	1×-1
0×1	0×0	1×-1

- Result: -2.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2			

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	1×0	1×-1
0×1	1×0	0×-1
0×1	1×0	0×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0		

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	1×0	0×-1
1×1	0×0	0×-1
1×1	0×0	1×-1

- Result: 2.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	1×-1
1×1	0×0	1×-1
0×1	1×0	0×-1

- Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1							

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	1×0	0×-1
0×1	1×0	1×-1
1×1	0×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1						

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	1×-1
1×1	1×0	0×-1
0×1	0×0	1×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0					

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	0×-1
1×1	0×0	0×-1
0×1	1×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1				

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	1×-1
0×1	0×0	1×-1
1×1	0×0	0×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0			

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

0×1	1×0	0×-1
0×1	1×0	0×-1
0×1	0×0	1×-1

– Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1		

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	0×-1
1×1	0×0	1×-1
0×1	1×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	1×-1
0×1	1×0	0×-1
1×1	0×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1							

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	1×-1
1×1	0×0	0×-1
0×1	0×0	1×-1

- Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1						

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	1×0	0×-1
0×1	0×0	1×-1
0×1	1×0	0×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0					

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	0×-1
0×1	1×0	0×-1
1×1	0×0	1×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1				

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

0×1	0×0	1×-1
1×1	0×0	0×-1
0×1	1×0	0×-1

– Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0			

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

0×1	1×0	0×-1
0×1	0×0	1×-1
1×1	0×0	1×-1

– Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1		

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

1×1	0×0	1×-1
0×1	1×0	0×-1
0×1	1×0	1×-1

– Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	0×-1
1×1	0×0	0×-1
0×1	1×0	1×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0							

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	0×-1
0×1	0×0	1×-1
1×1	1×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1						

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	0×0	1×-1
0×1	1×0	0×-1
1×1	0×0	0×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0					

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	0×-1
1×1	0×0	1×-1
0×1	0×0	0×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0				

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	0×-1
0×1	1×0	0×-1
0×1	0×0	1×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0			

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	0×0	1×-1
1×1	0×0	1×-1
0×1	1×0	0×-1

- Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1		

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	0×-1
0×1	1×0	1×-1
1×1	0×0	1×-1

- Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1	-1	

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

1×1	0×0	0×-1
0×1	1×0	1×-1
0×1	1×0	0×-1

– Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1	-1	
	0							

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	0×0	1×-1
1×1	1×0	0×-1
1×1	0×0	0×-1

- Result: 1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1	-1	
	0	1						

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

0×1	1×0	0×-1
1×1	0×0	0×-1
0×1	0×0	1×-1

- Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1	-1	
	0	1	0					

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

- Convolution Operation

1×1	0×0	1×-1
0×1	0×0	0×-1
0×1	1×0	1×-1

- Result: -1.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1	-1	
	0	1	0	-1				

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

0×1	1×0	0×-1
0×1	0×0	1×-1
1×1	1×0	0×-1

– Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1	-1	
	0	1	0	-1	0			

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

1×1	0×0	1×-1
0×1	1×0	0×-1
1×1	0×0	1×-1

– Result: 0.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1	-1	
	0	1	0	-1	0	0		

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

– Convolution Operation

0×1	1×0	1×-1
1×1	0×0	1×-1
0×1	1×0	1×-1

– Result: -2.

Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1	-1	
	0	1	0	-1	0	0	-2	

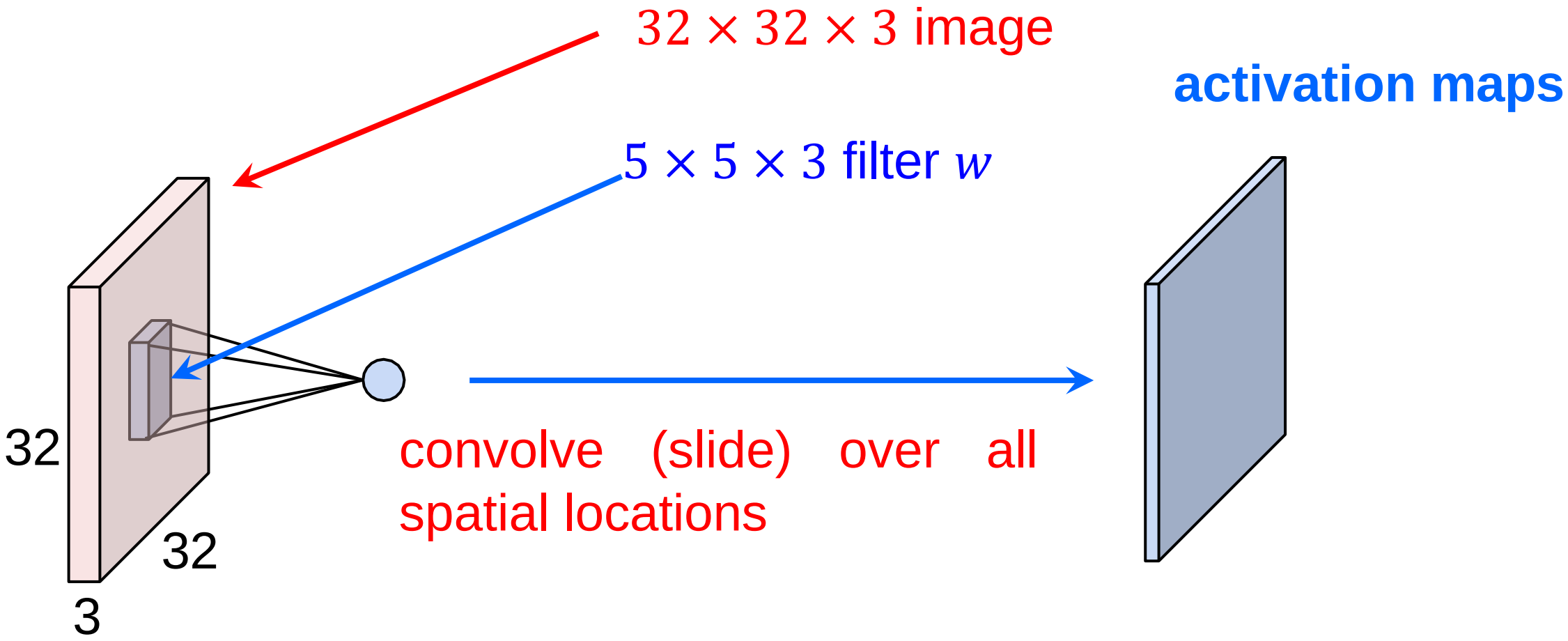
Convolution Layer

1	1	0	0	0	1	1	0	1
0	0	1	1	1	0	0	1	1
1	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	0	1	1
0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1

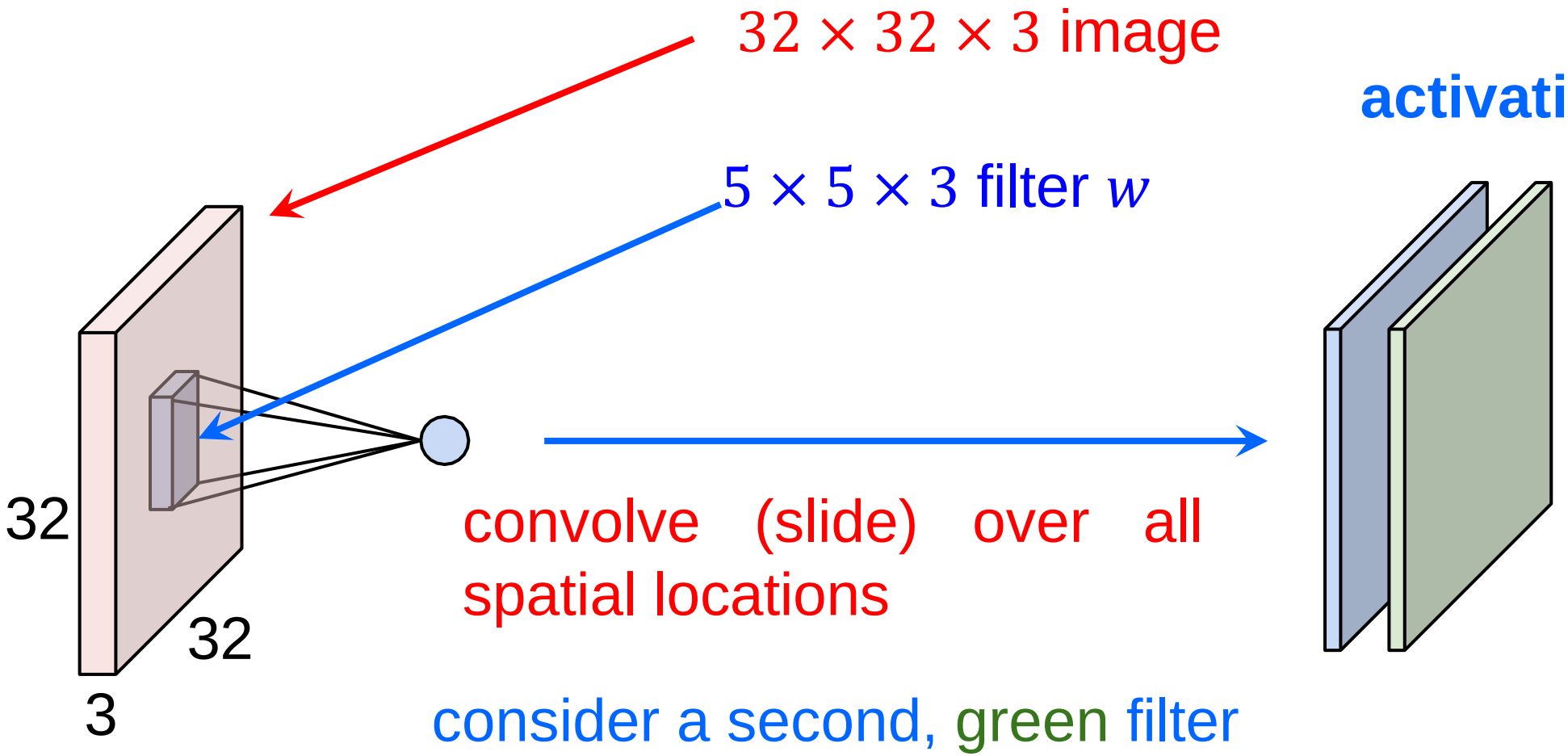
1	0	-1
1	0	-1
1	0	-1

	1	1	0	-1	-1	0	0	
	-1	1	0	0	0	-1	1	
	0	1	1	0	-2	0	2	
	-1	1	0	1	0	-1	1	
	1	-1	0	1	0	-1	-1	
	0	1	0	0	0	-1	-1	
	0	1	0	-1	0	0	-2	

Convolution Layer



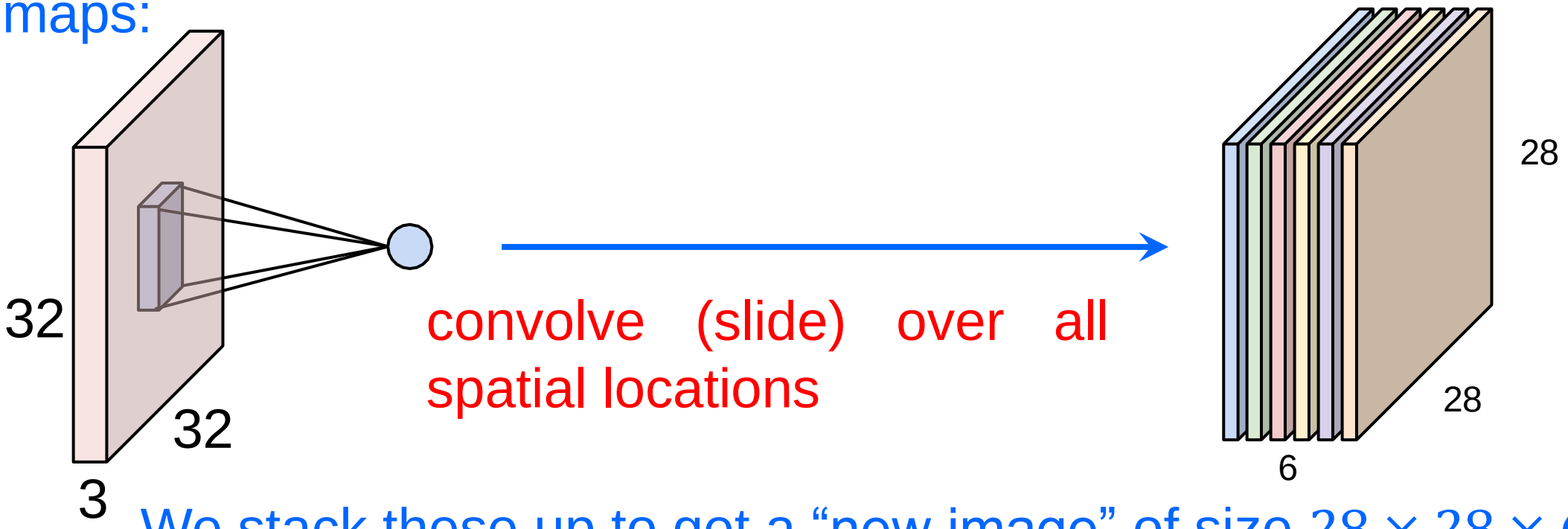
Convolution Layer



Convolution Layer

For example, if we had 6 $5 \times 5 \times 3$ filters, we'll get 6 separate activation maps:

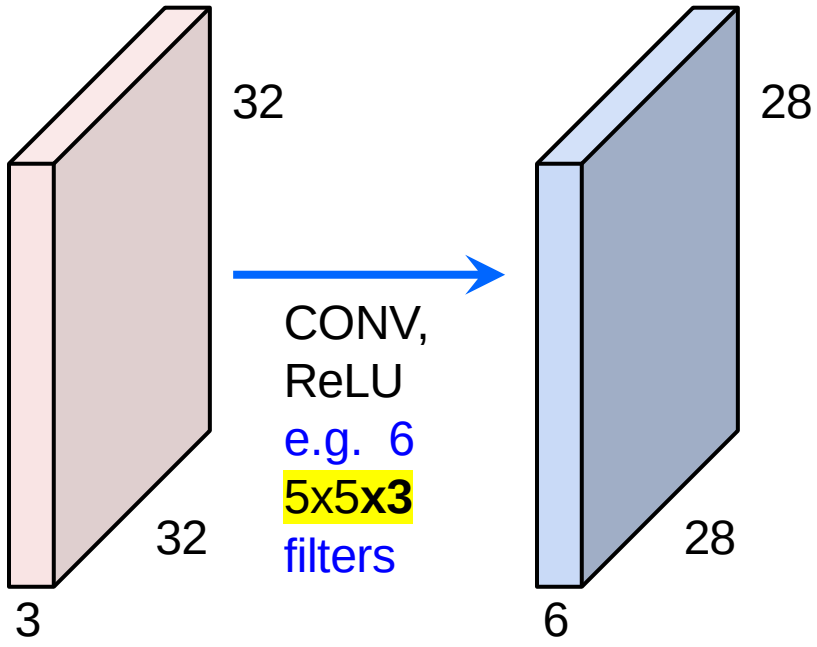
activation maps



We stack these up to get a "new image" of size $28 \times 28 \times 6$.

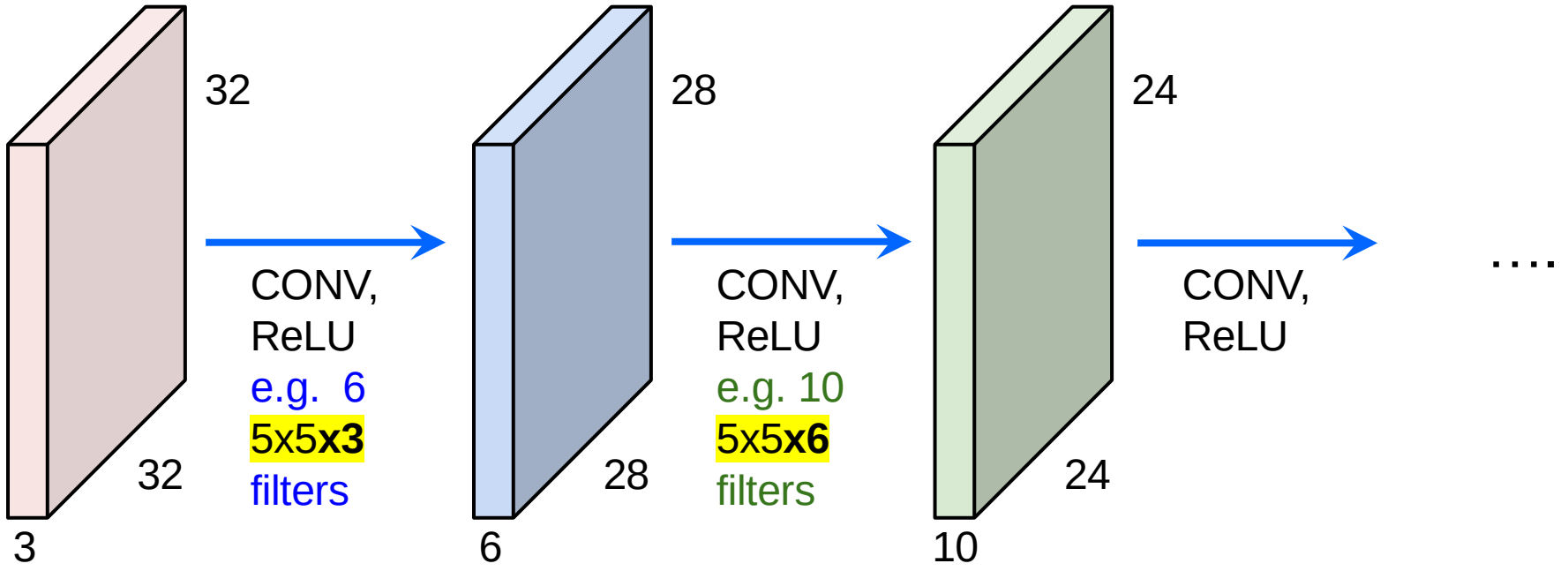
ConvNet

– Preview: ConvNet is a sequence of Convolution Layers, interspersed with activation functions.

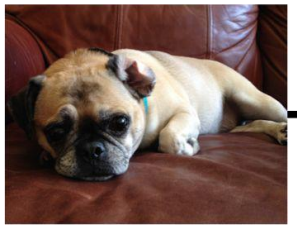


ConvNet

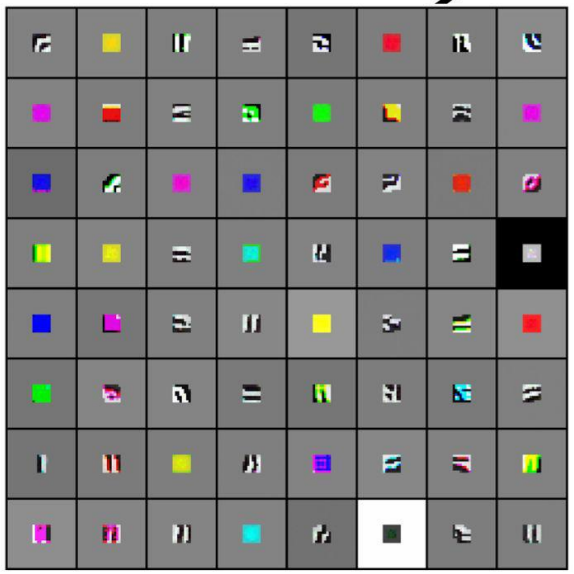
– Preview: ConvNet is a sequence of Convolution Layers, interspersed with activation functions.



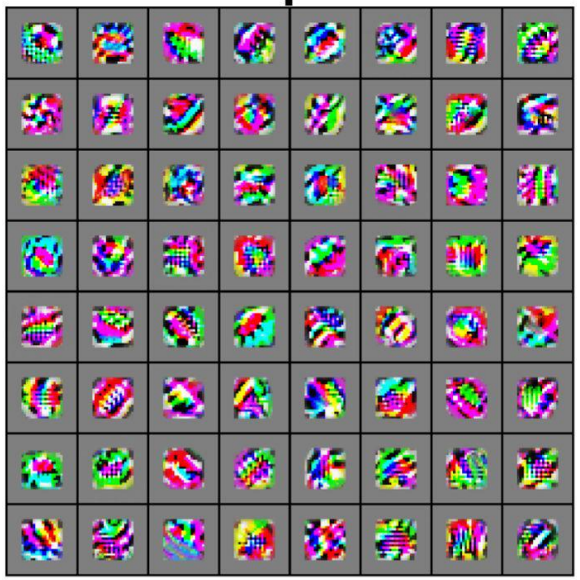
ConvNet



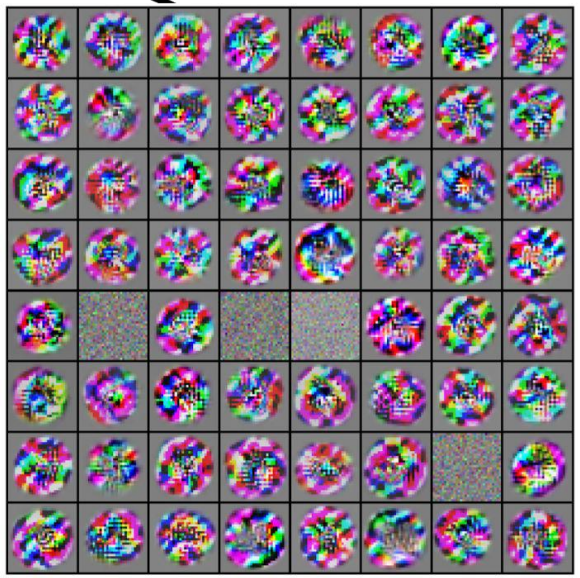
Low-level features → Mid-level features → High-level features → Linearly separable classifier



VGG-16 Conv1_1



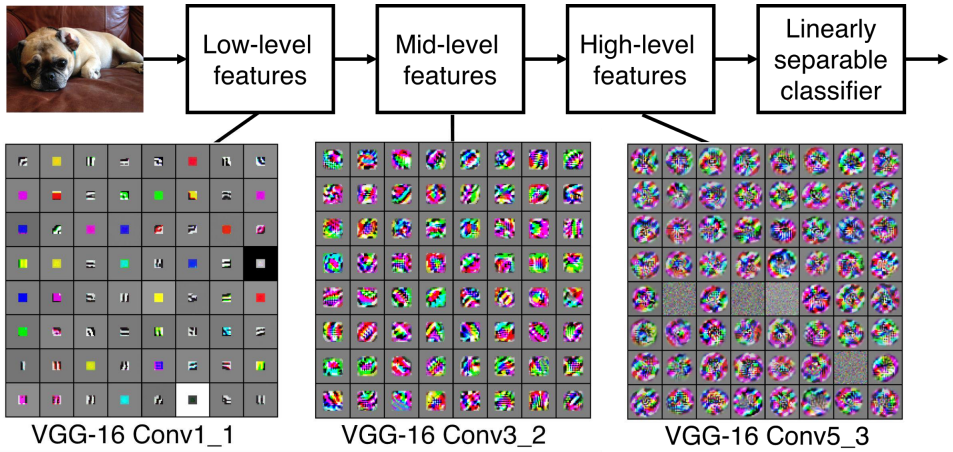
VGG-16 Conv3_2

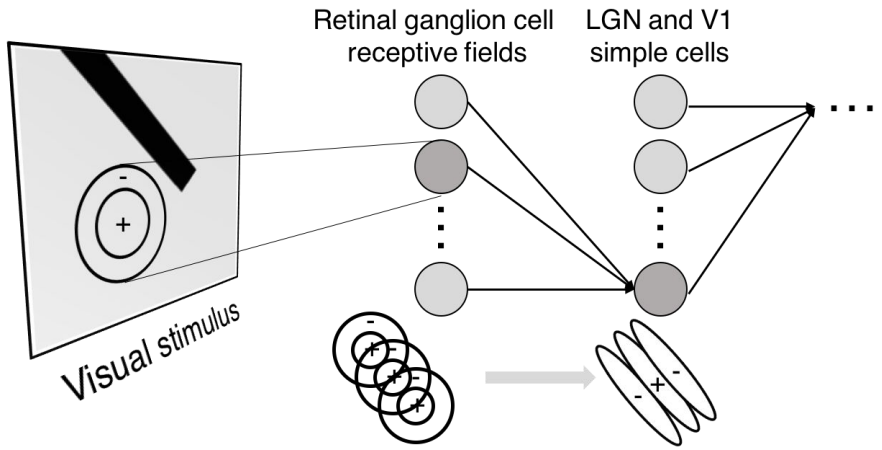


VGG-16 Conv5_3

[Zeiler and Fergus 2013]

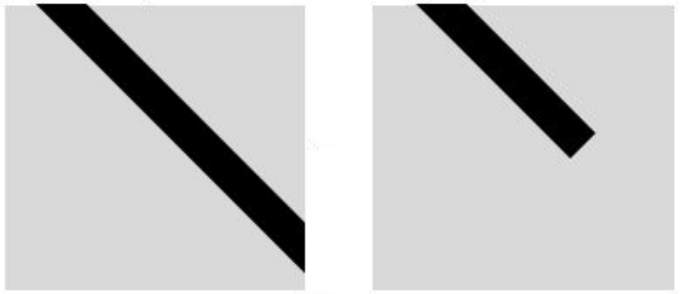
ConvNet





Complex cells:
Response to light orientation and movement

Hypercomplex cells:
response to movement with an end point



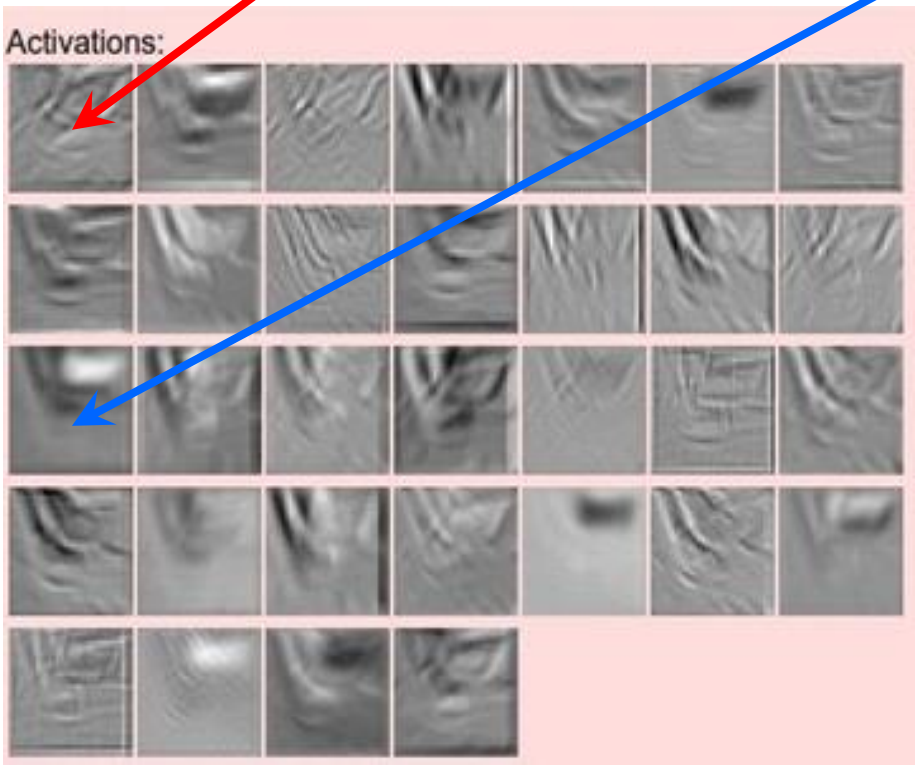
No response

Response (end point)

ConvNet

one filter ⟹ one activation map

example 5×5 filters (32 total)

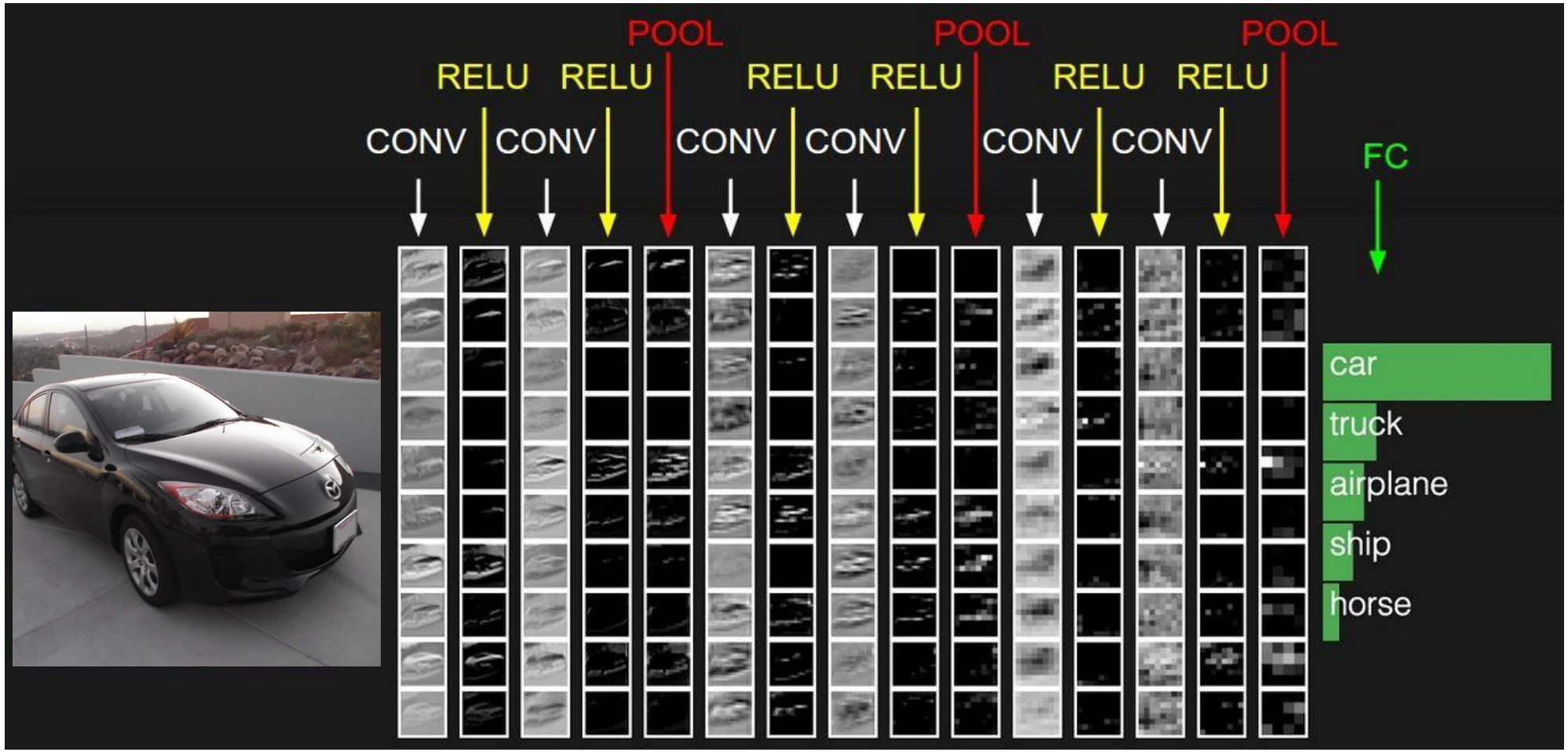


We call the layer convolutional because it is related to convolution of two signals:

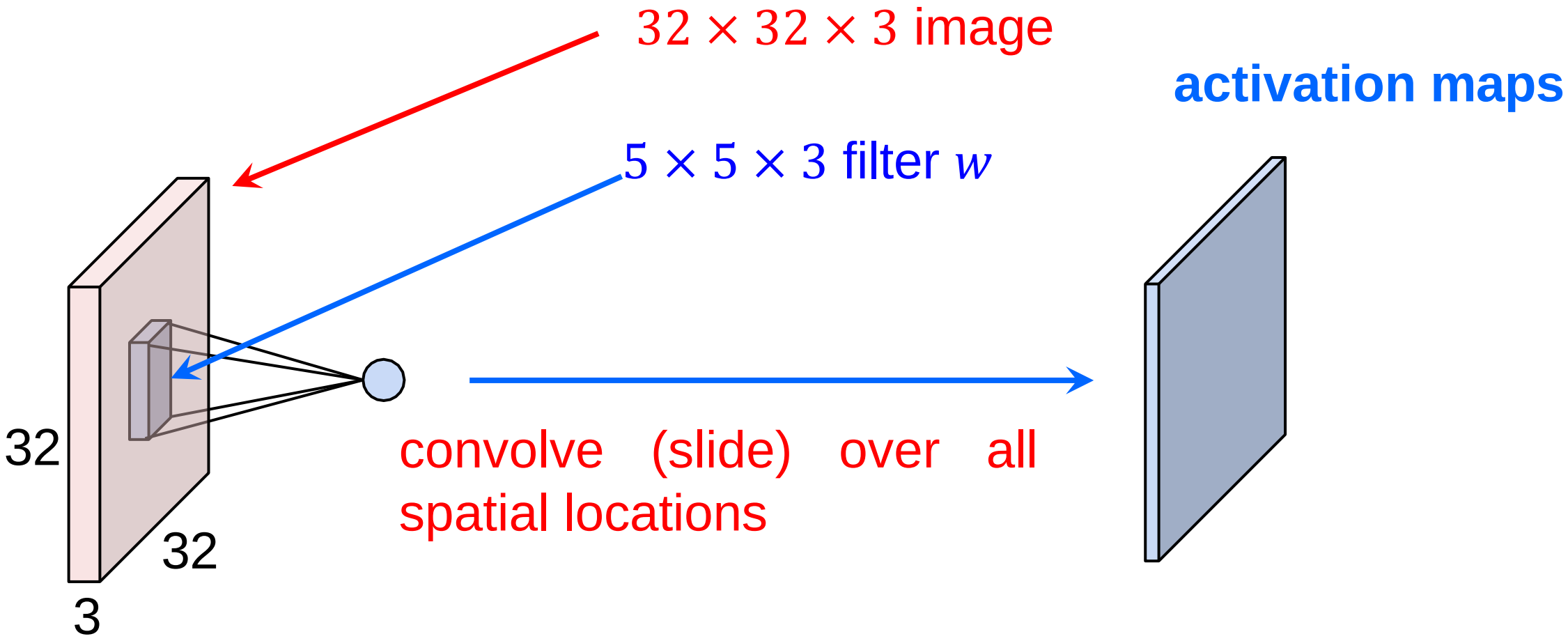
$$f[x,y] * g[x,y] = \sum_{n_1=-\infty}^{+\infty} \sum_{n_1=-\infty}^{+\infty} f[n_1,n_2] \cdot f[x-n_1, y-n_2]$$

elementwise multiplication and sum of a filter and the signal (image)

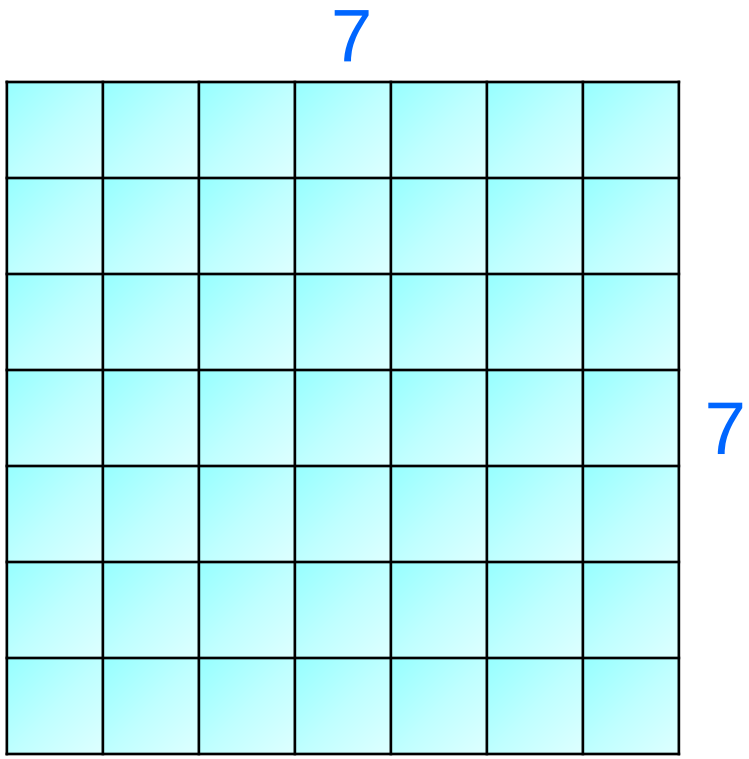
ConvNet


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

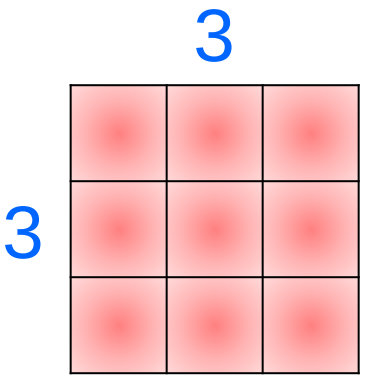
A closer look at spatial dimensions



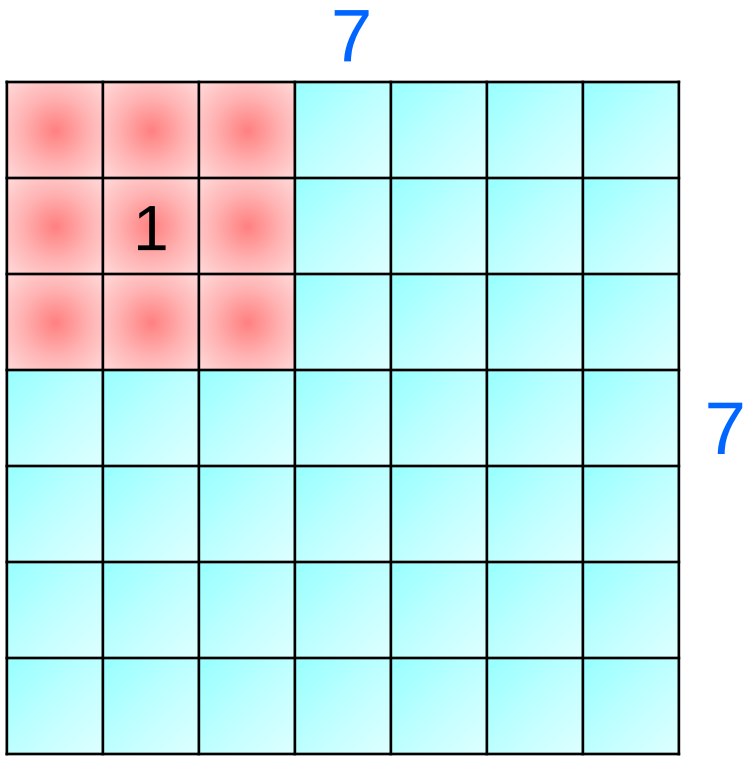
A closer look at spatial dimensions



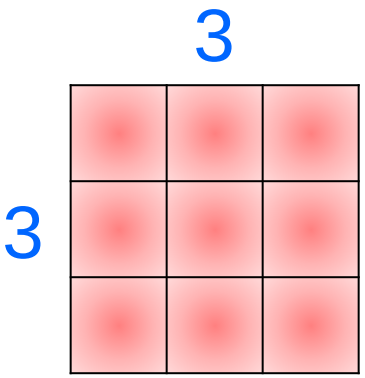
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



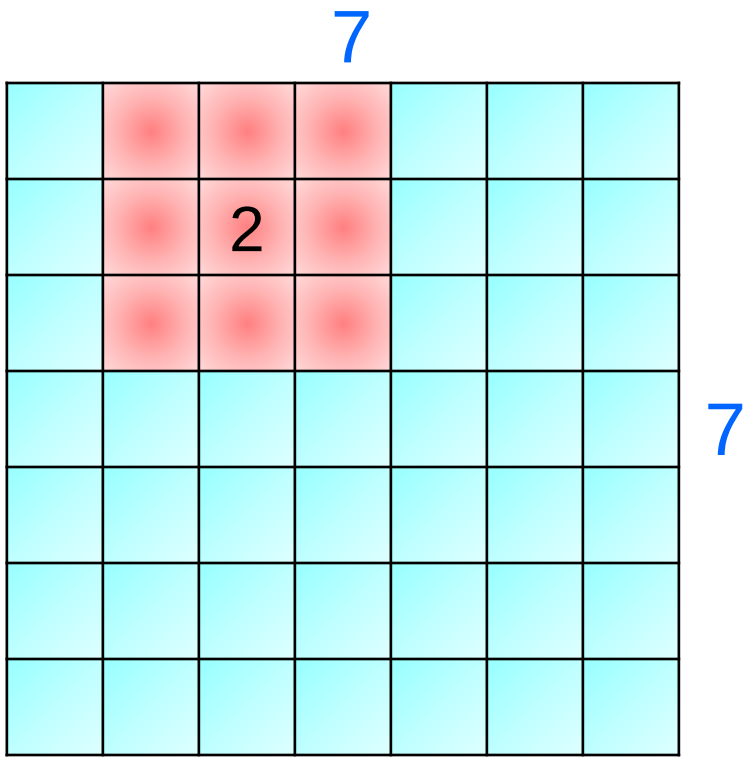
A closer look at spatial dimensions



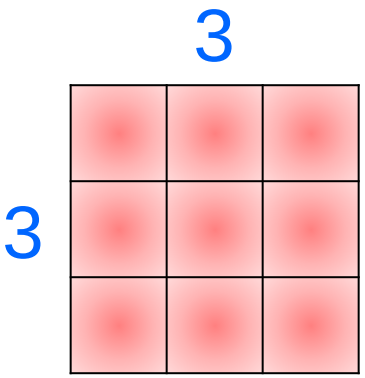
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



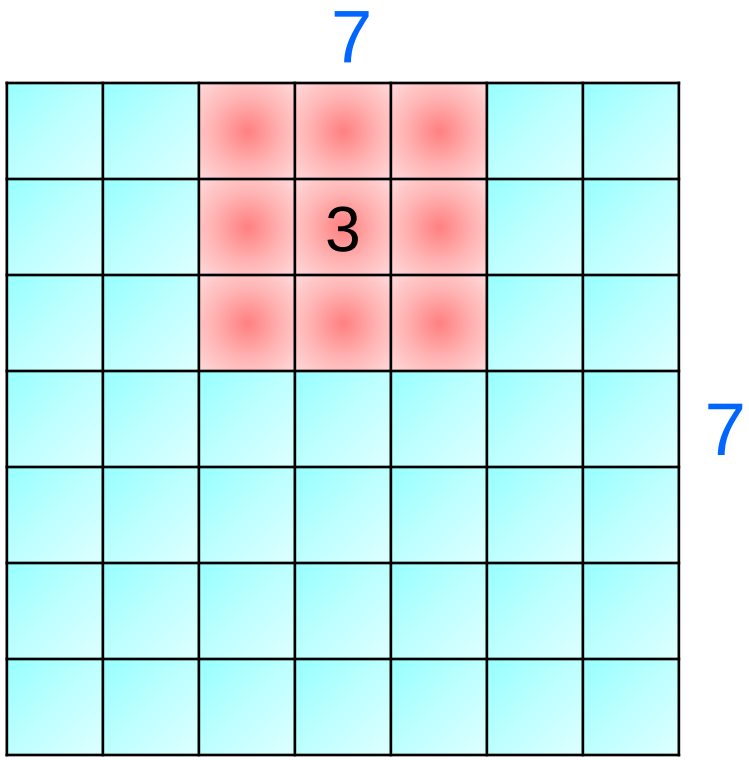
A closer look at spatial dimensions



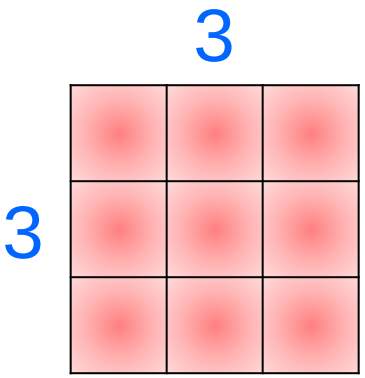
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



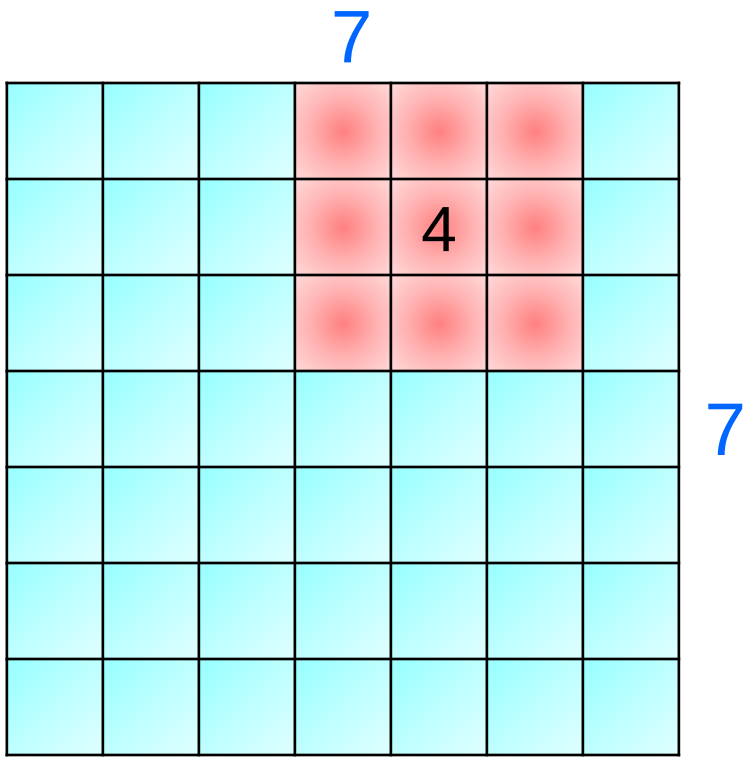
A closer look at spatial dimensions



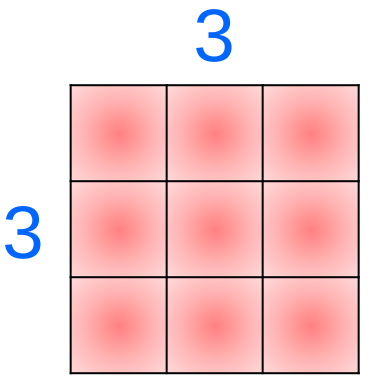
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



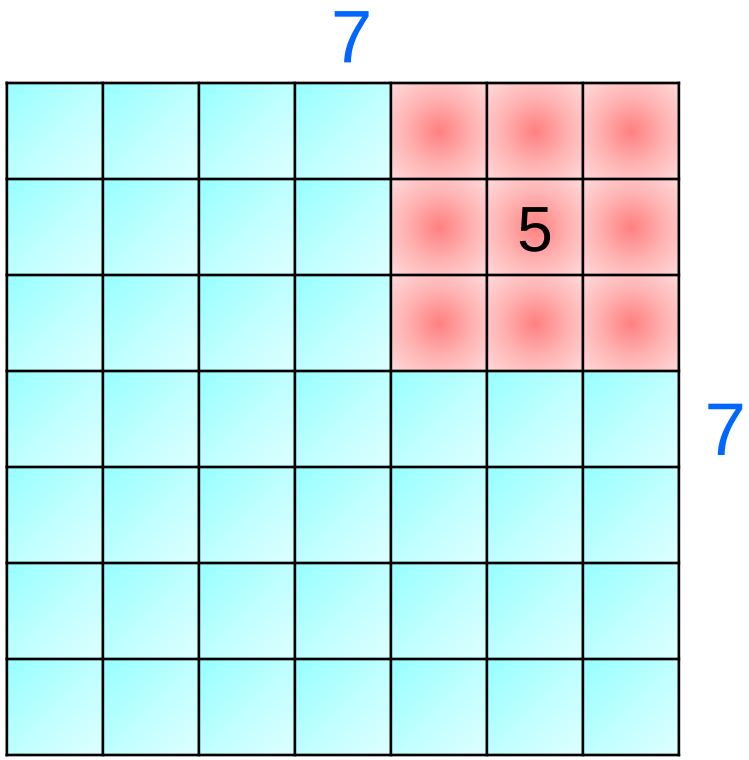
A closer look at spatial dimensions



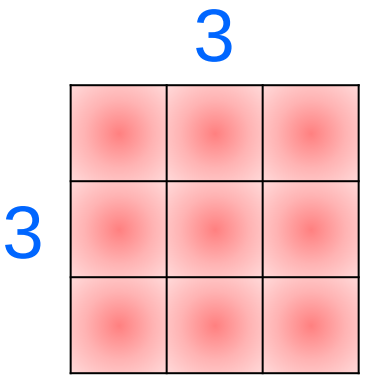
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



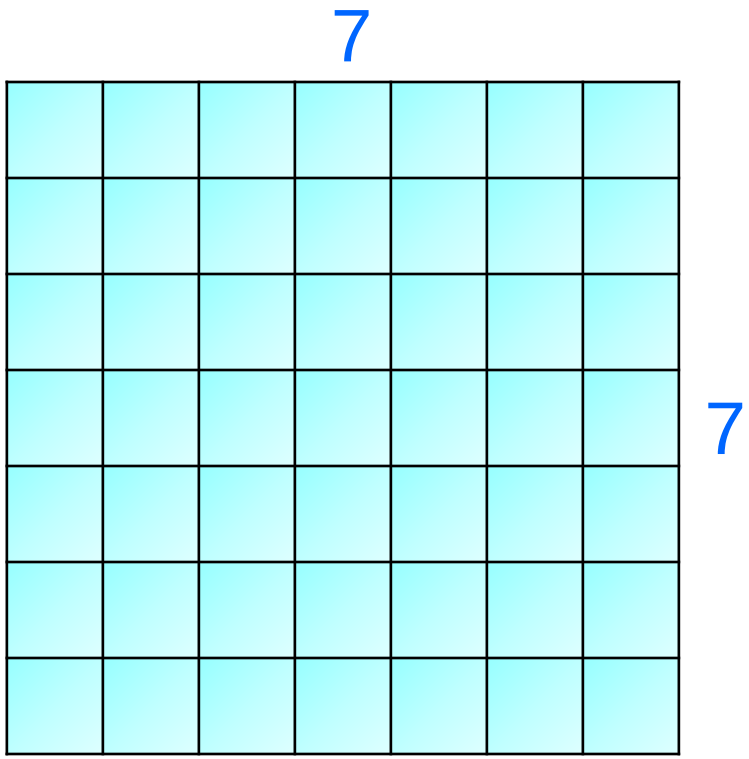
A closer look at spatial dimensions



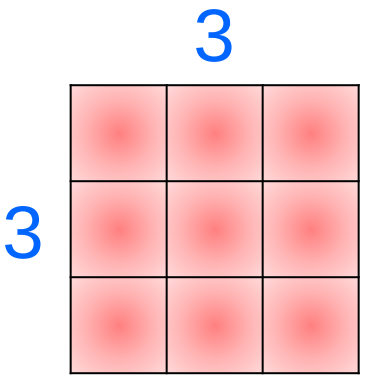
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



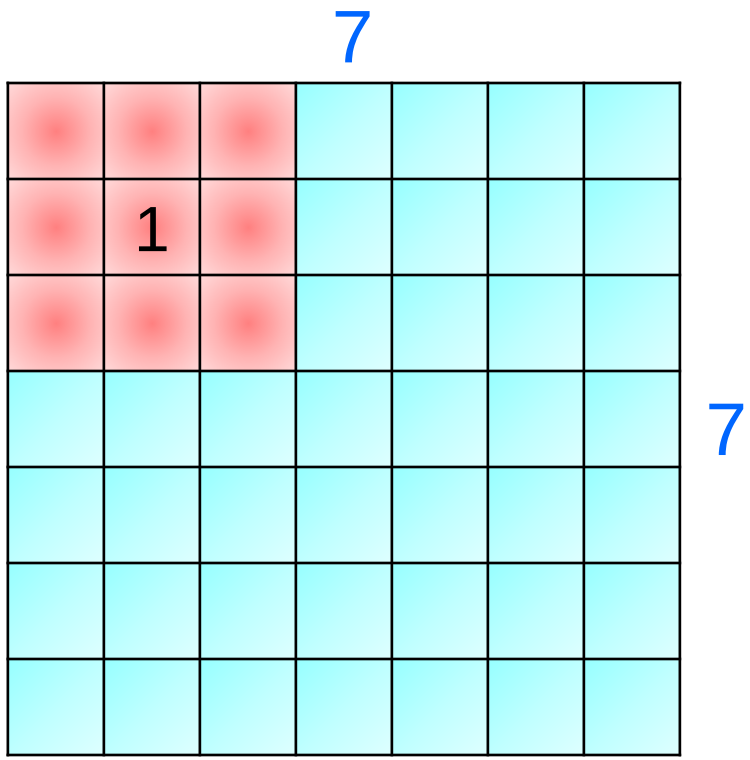
A closer look at spatial dimensions



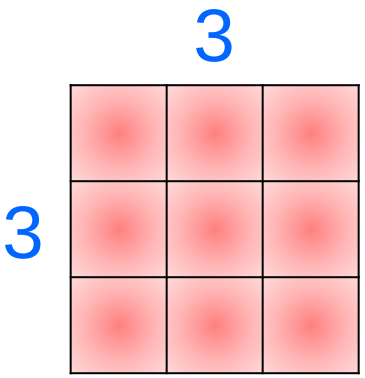
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



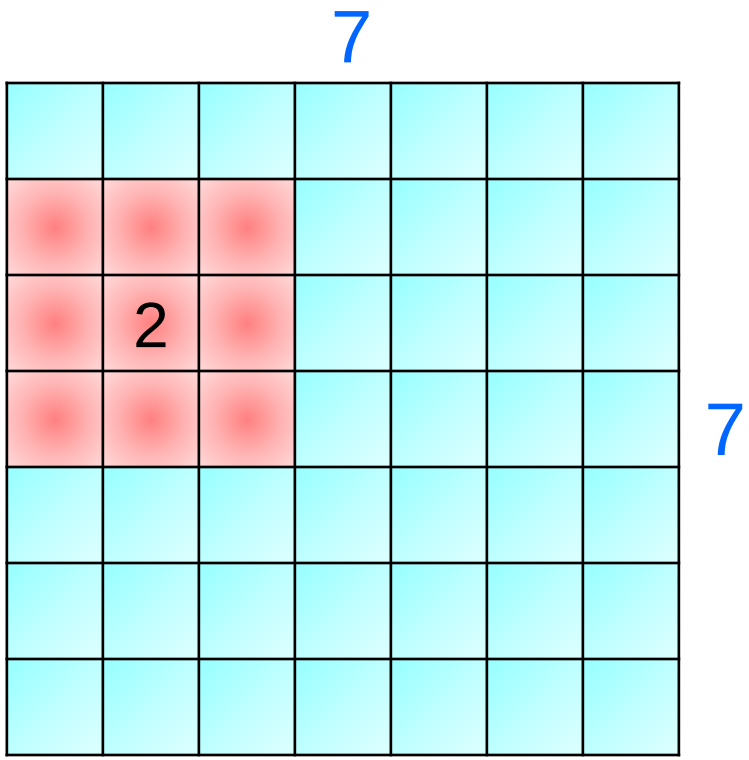
A closer look at spatial dimensions



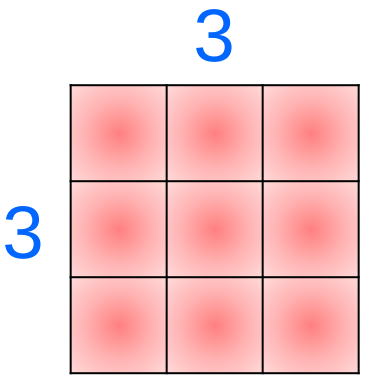
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



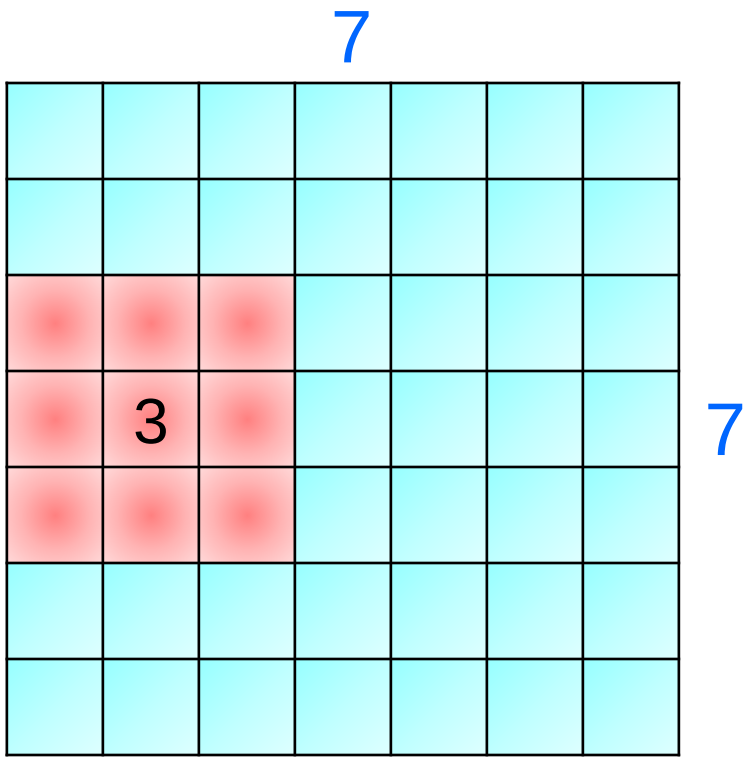
A closer look at spatial dimensions



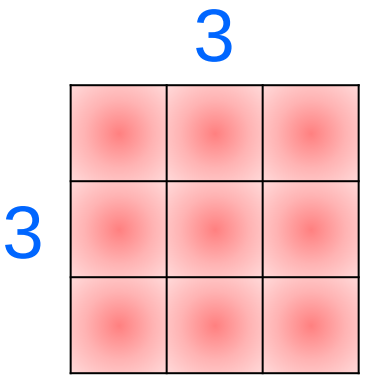
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



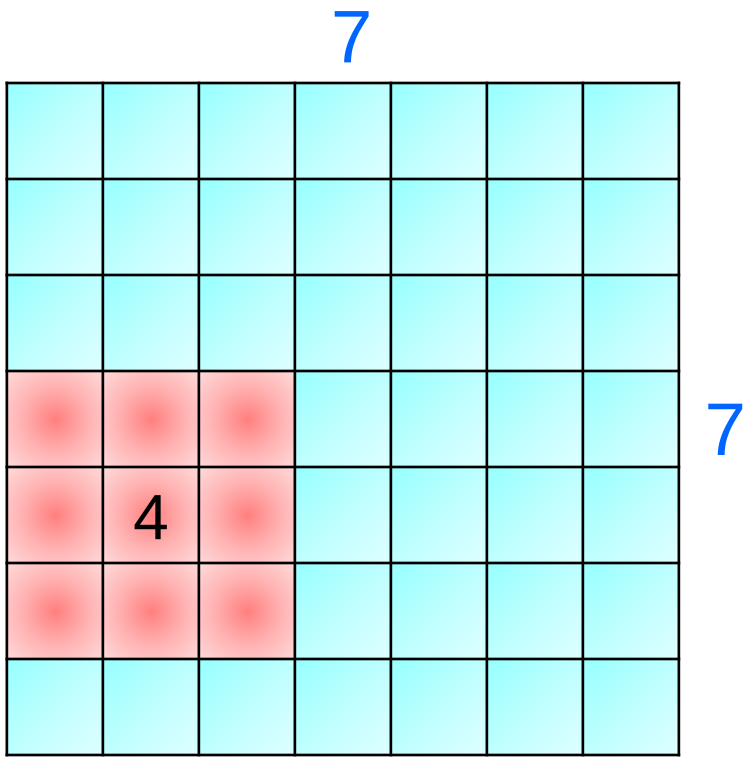
A closer look at spatial dimensions



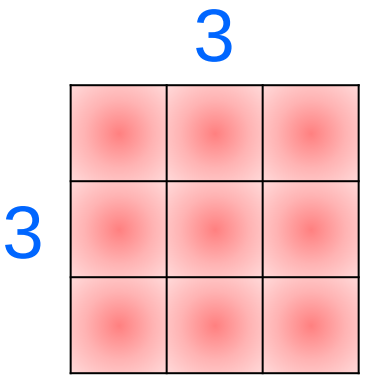
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



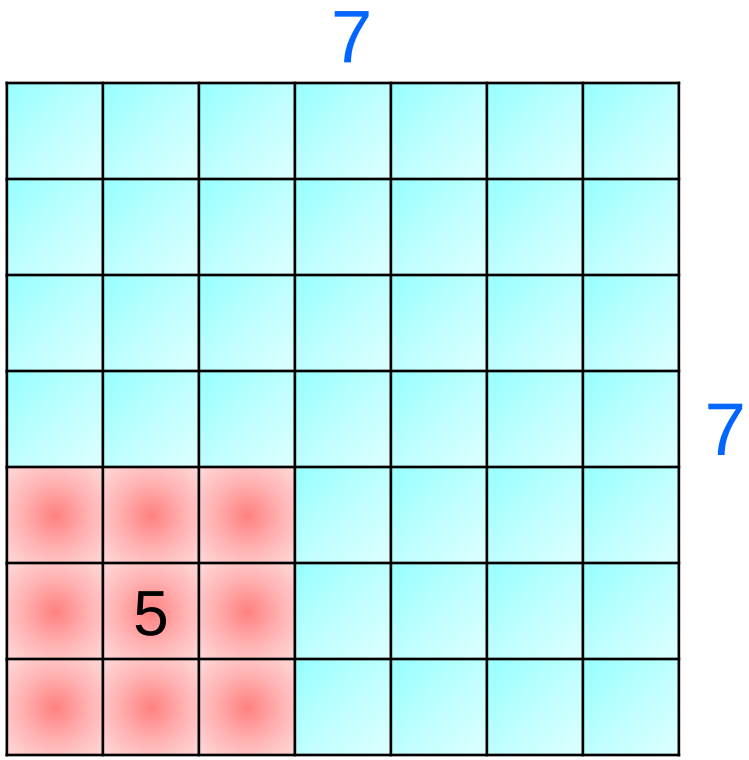
A closer look at spatial dimensions



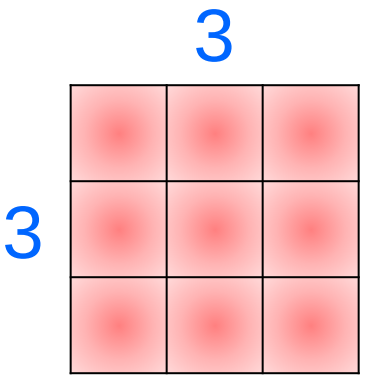
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



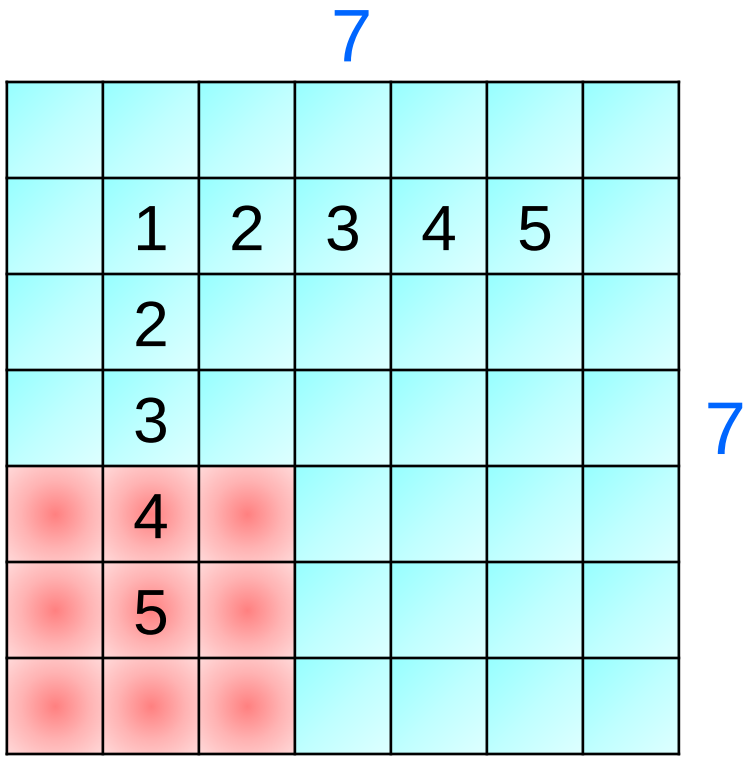
A closer look at spatial dimensions



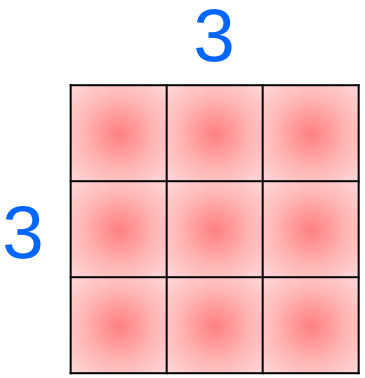
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?



A closer look at spatial dimensions

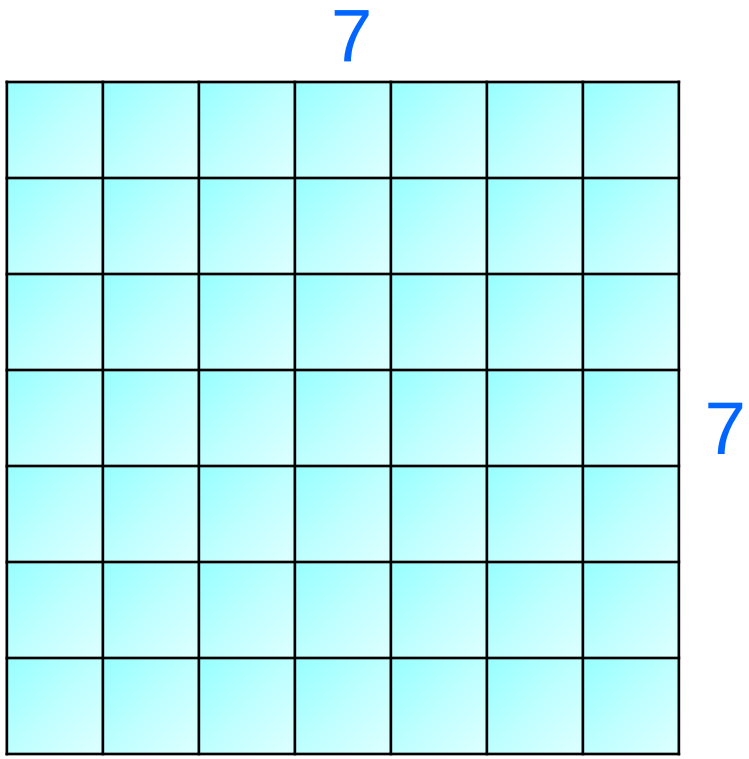


$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 1$?

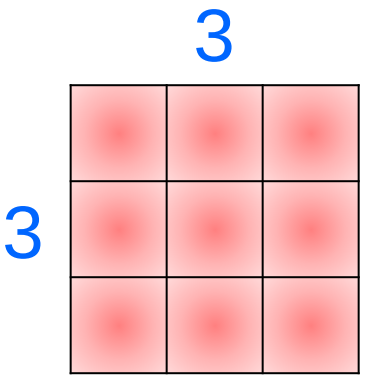


fit!
can apply 3×3 filter
on 7×7 input with stride 1
$\Longrightarrow 5 \times 5$ output

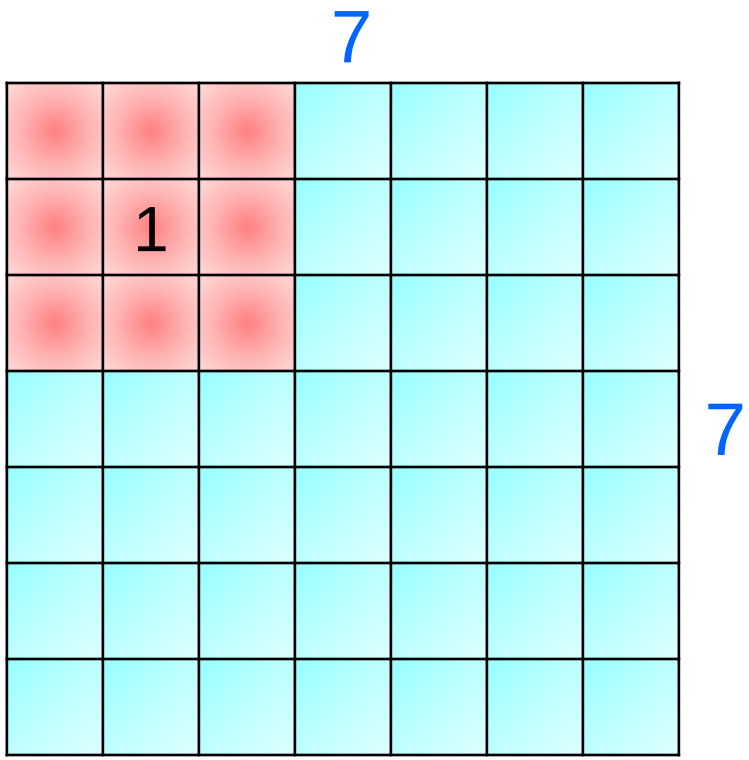
A closer look at spatial dimensions



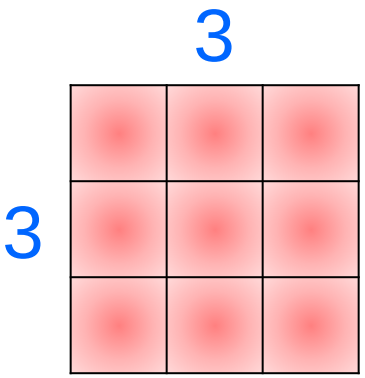
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 2$?



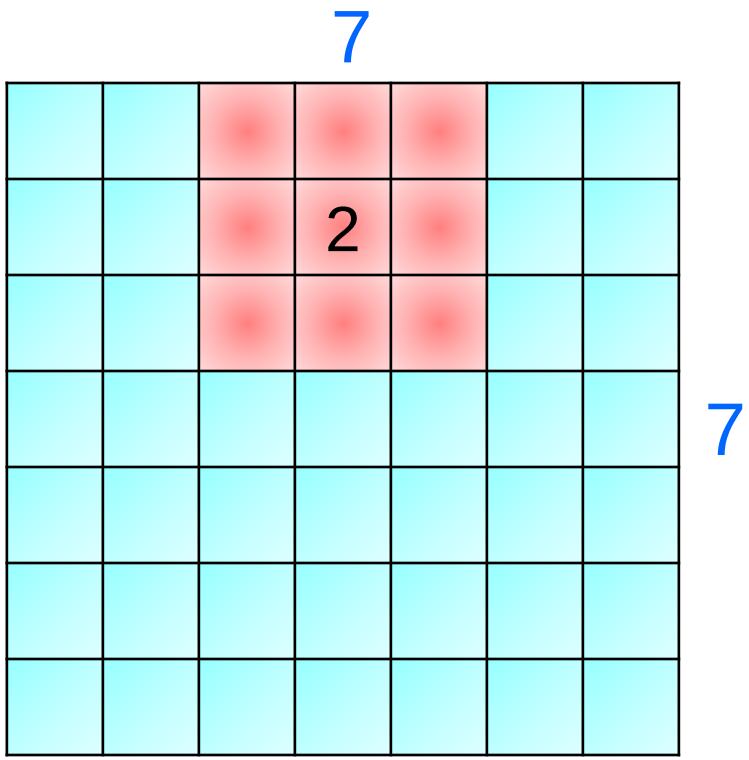
A closer look at spatial dimensions



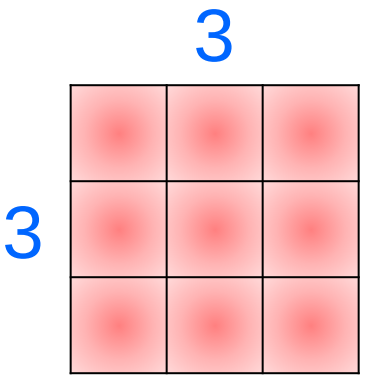
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 2$?



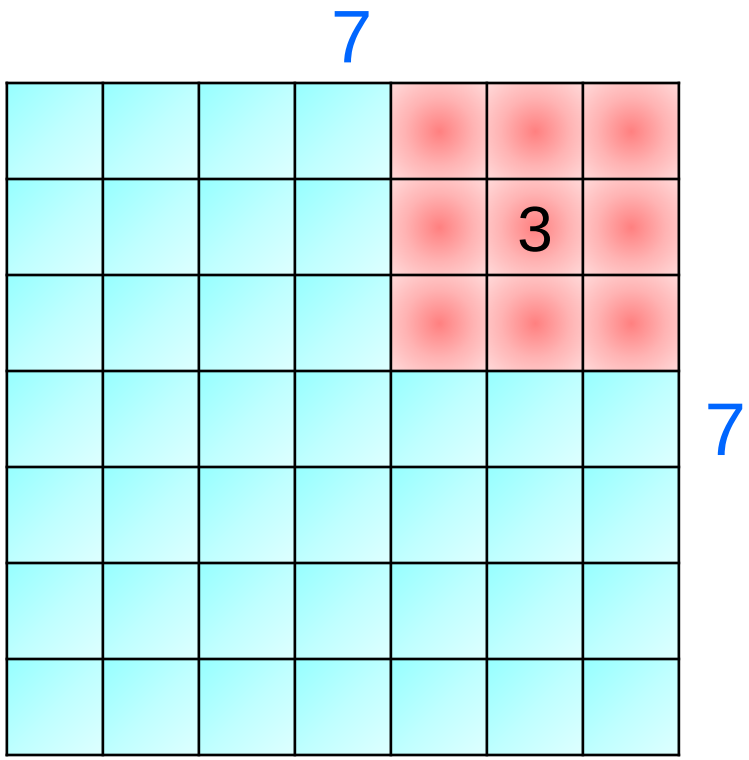
A closer look at spatial dimensions



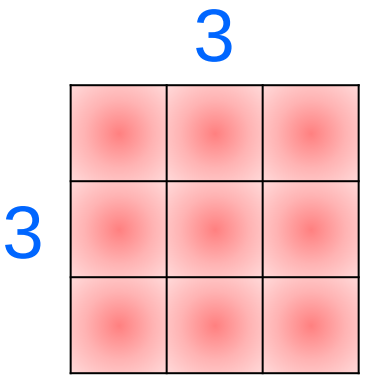
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 2$?



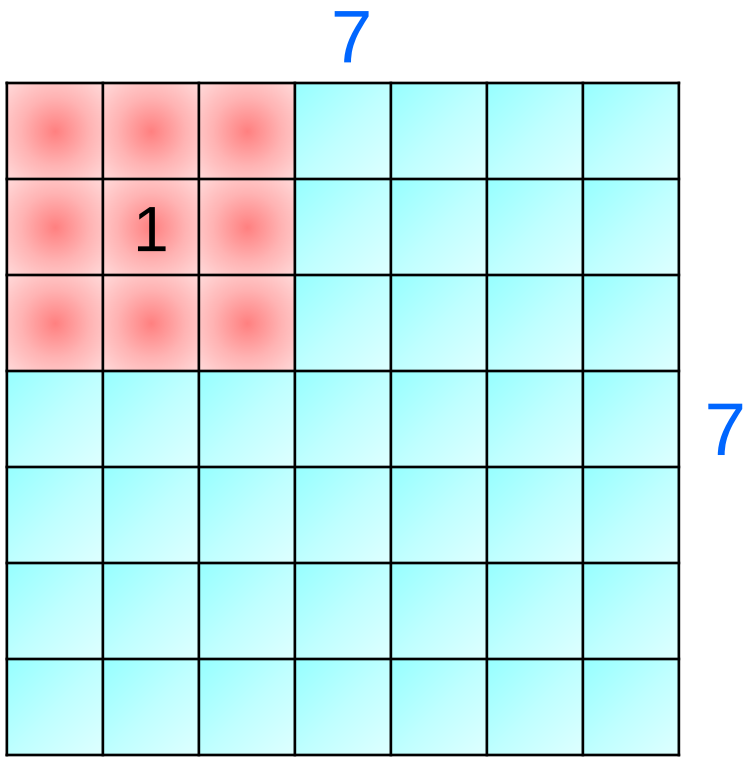
A closer look at spatial dimensions



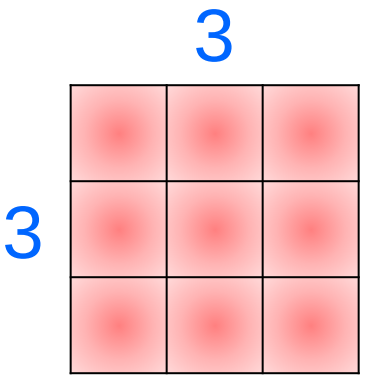
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 2$?



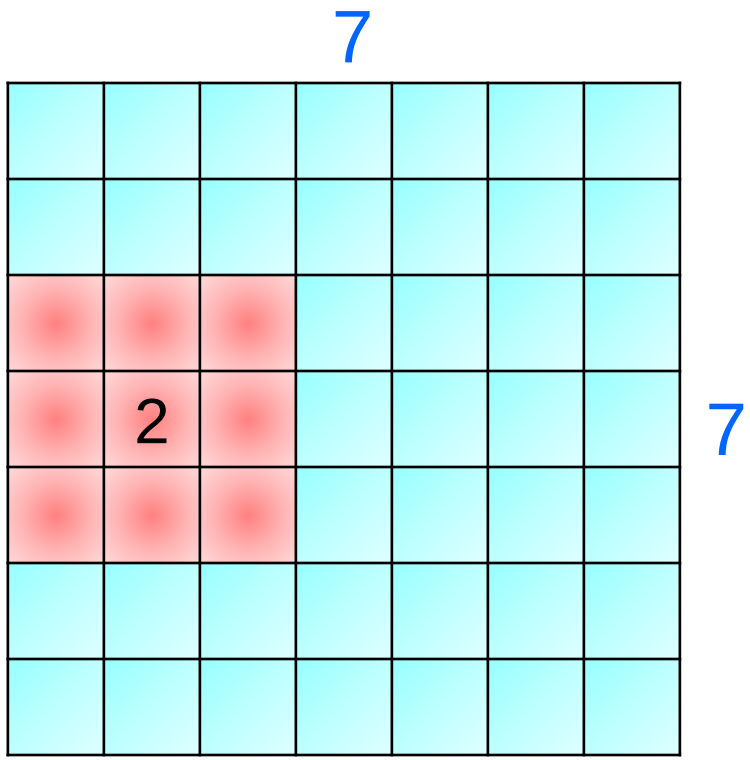
A closer look at spatial dimensions



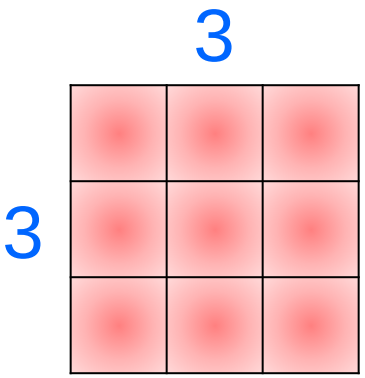
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 2$?



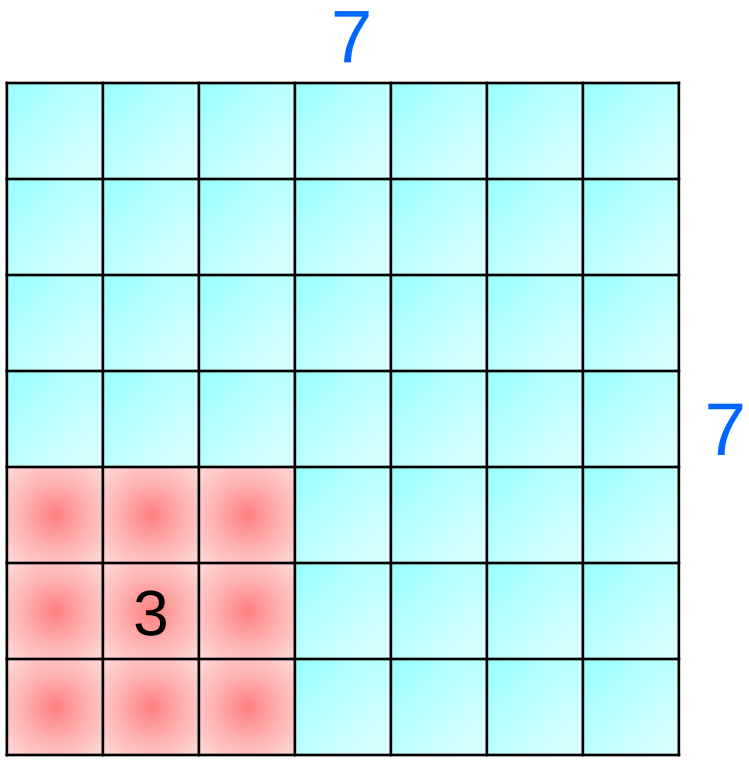
A closer look at spatial dimensions



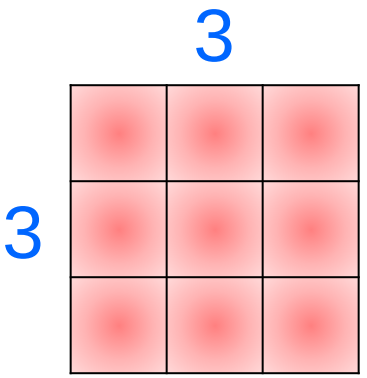
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 2$?



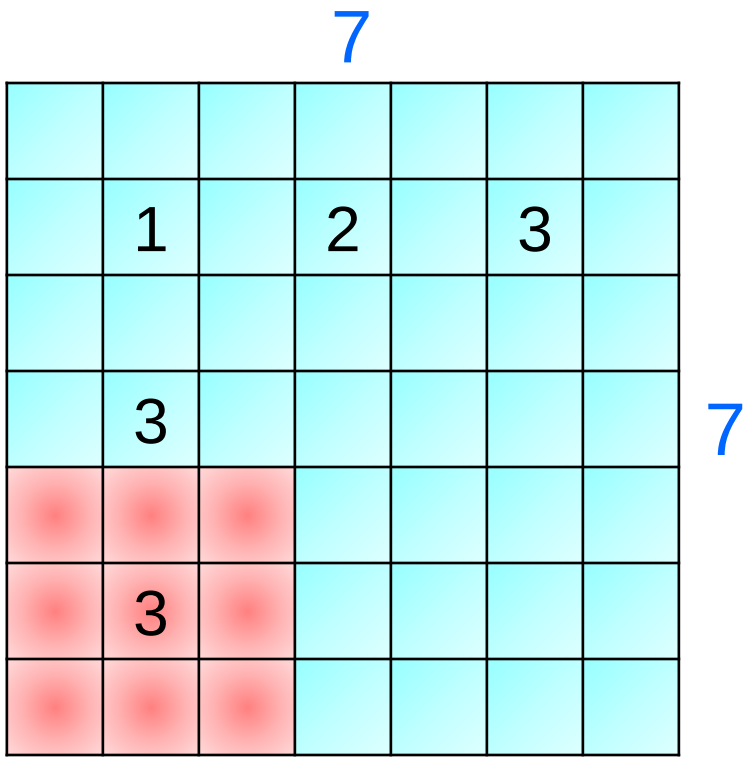
A closer look at spatial dimensions



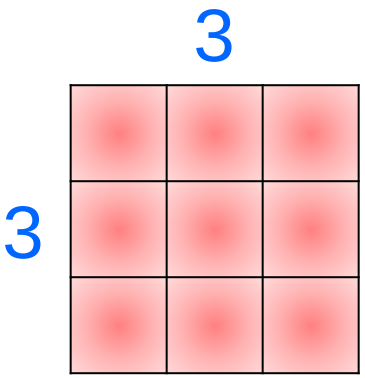
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 2$?



A closer look at spatial dimensions

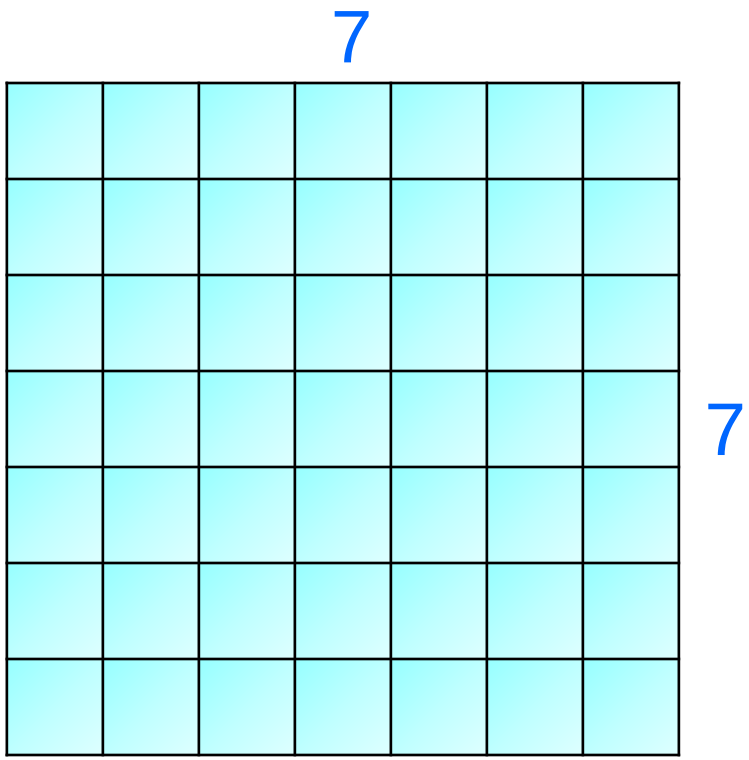


$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 2$?

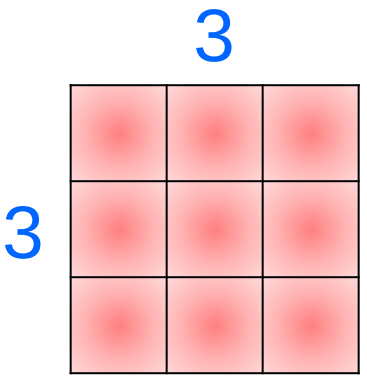


fit!
can apply 3×3 filter
on 7×7 input with stride 2
$\Longrightarrow 3 \times 3$ output

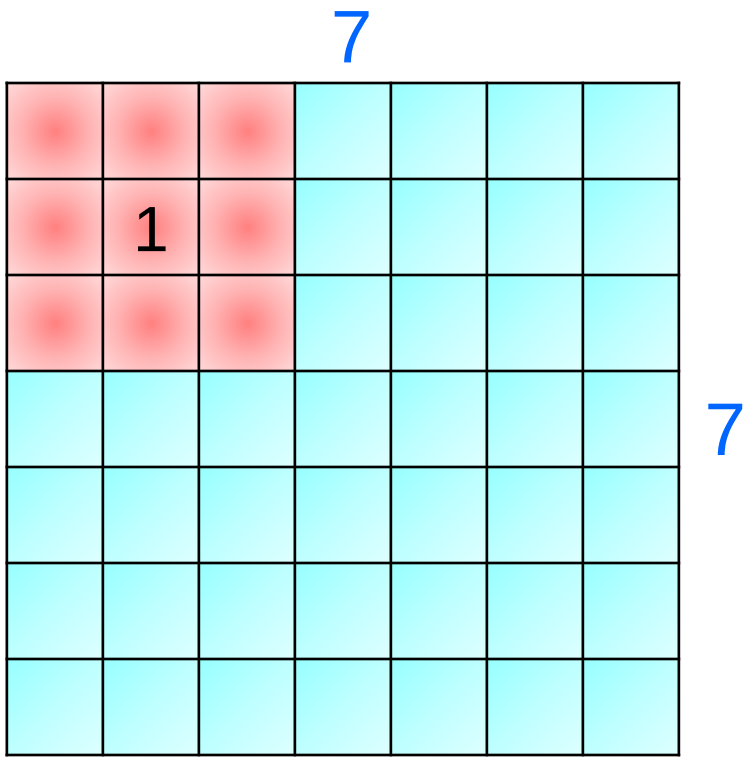
A closer look at spatial dimensions



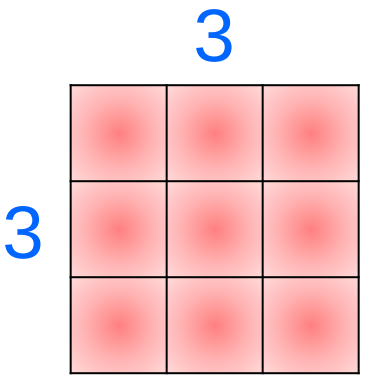
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 3$?



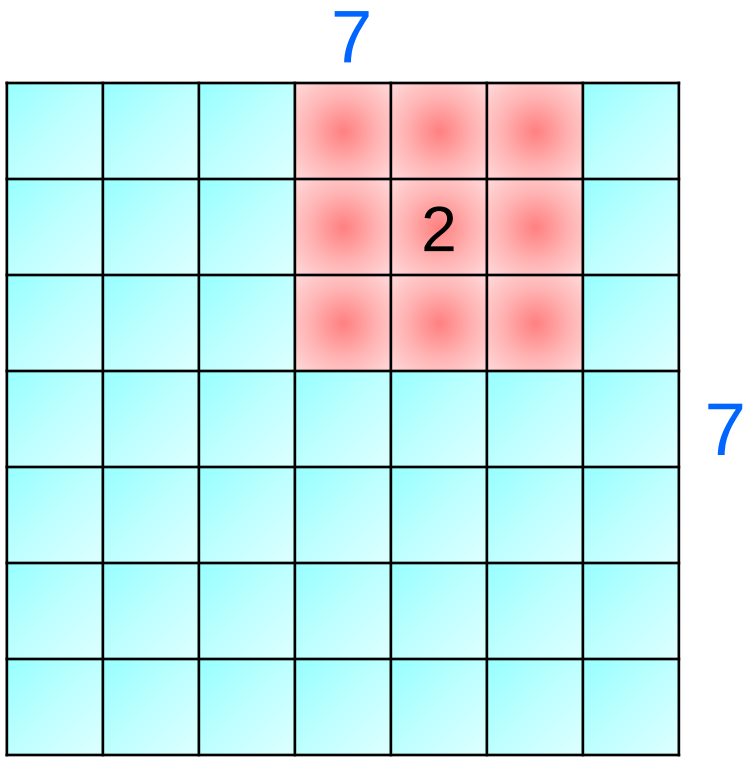
A closer look at spatial dimensions



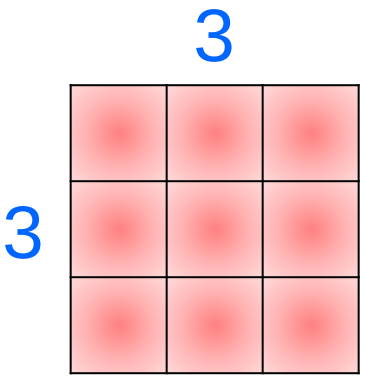
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 3$?



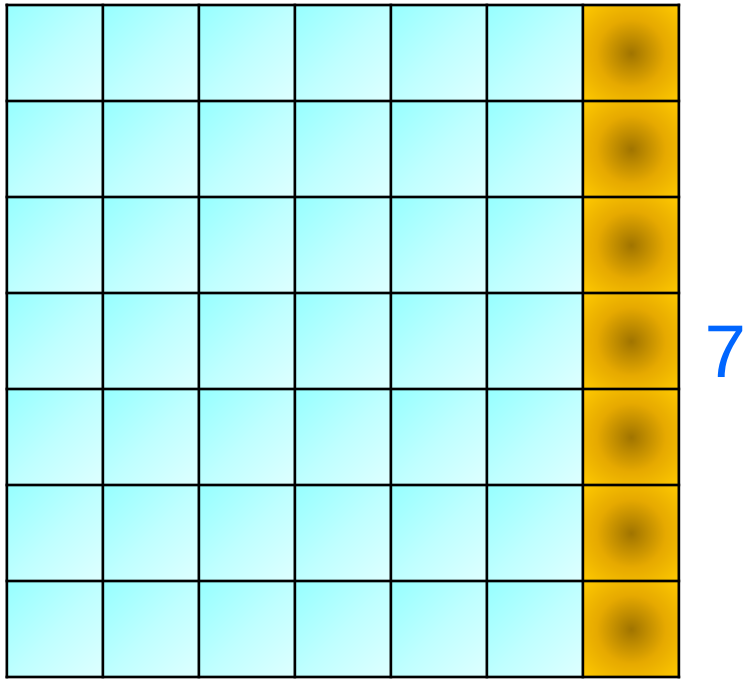
A closer look at spatial dimensions



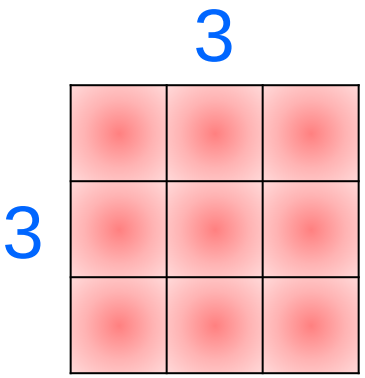
$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 3$?



A closer look at spatial dimensions

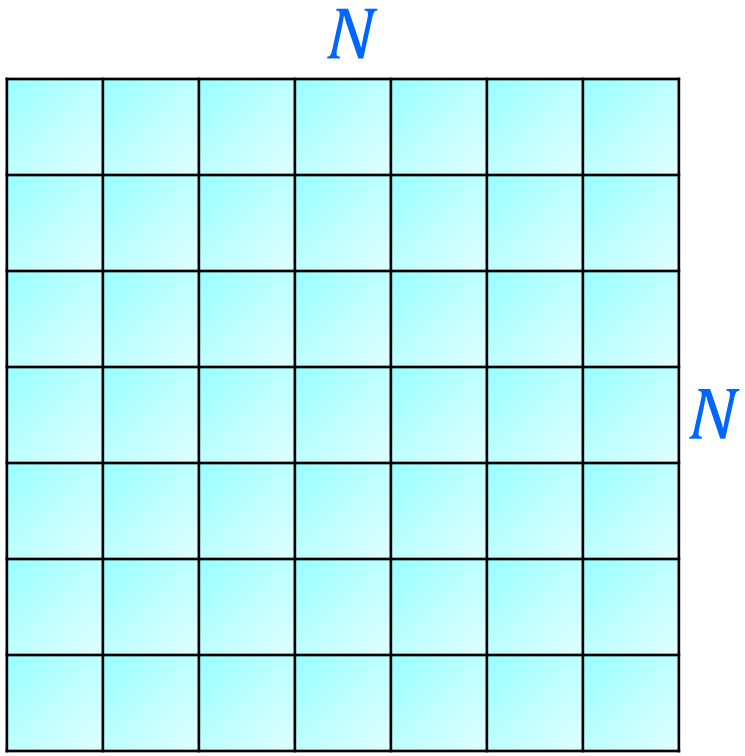


$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride = 3$?

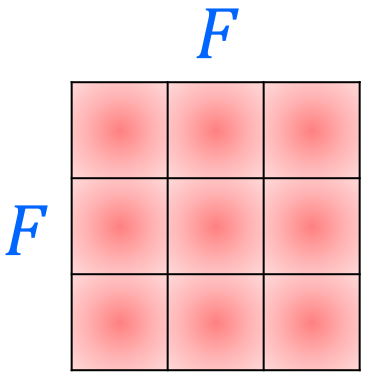


doesn't fit!
cannot apply 3×3 filter
on 7×7 input with stride 3.

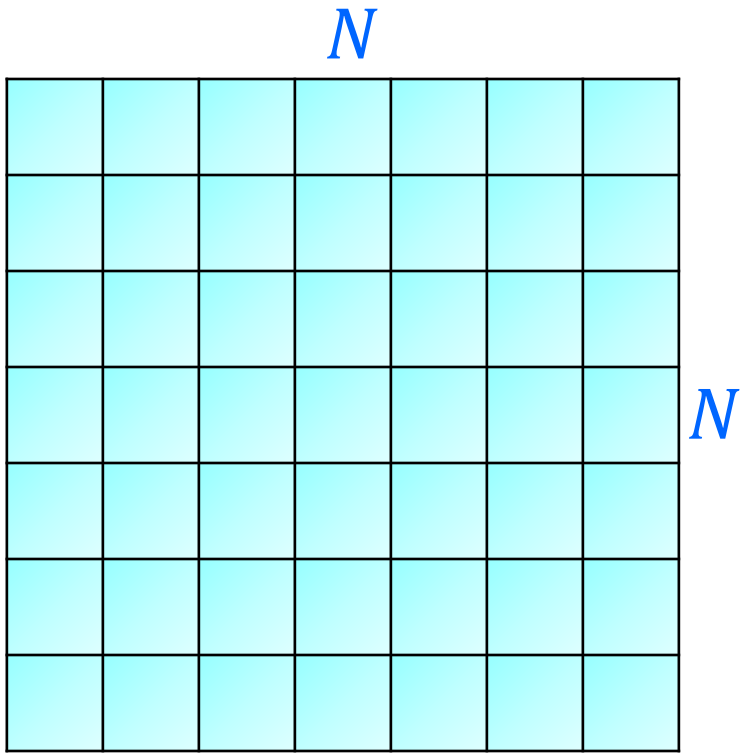
A closer look at spatial dimensions



$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride$?

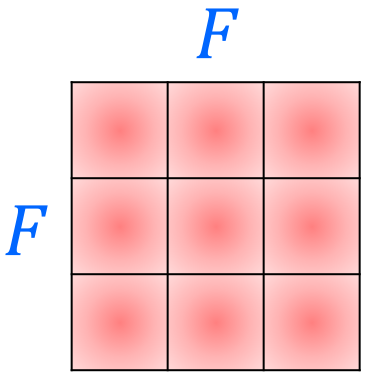


A closer look at spatial dimensions

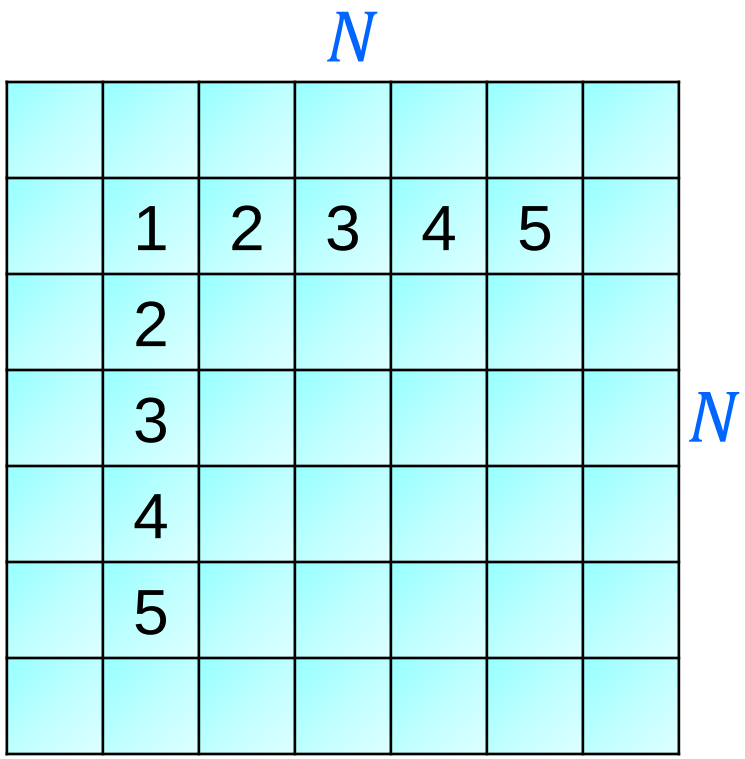


$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride$?

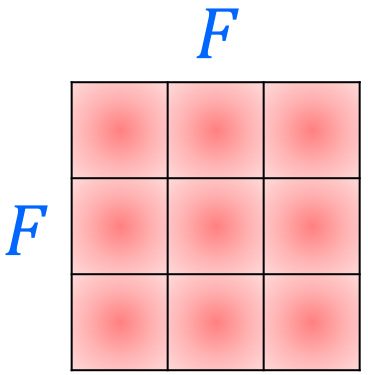
Output size $\frac{(N-F)}{stride} + 1$



A closer look at spatial dimensions



$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride$?

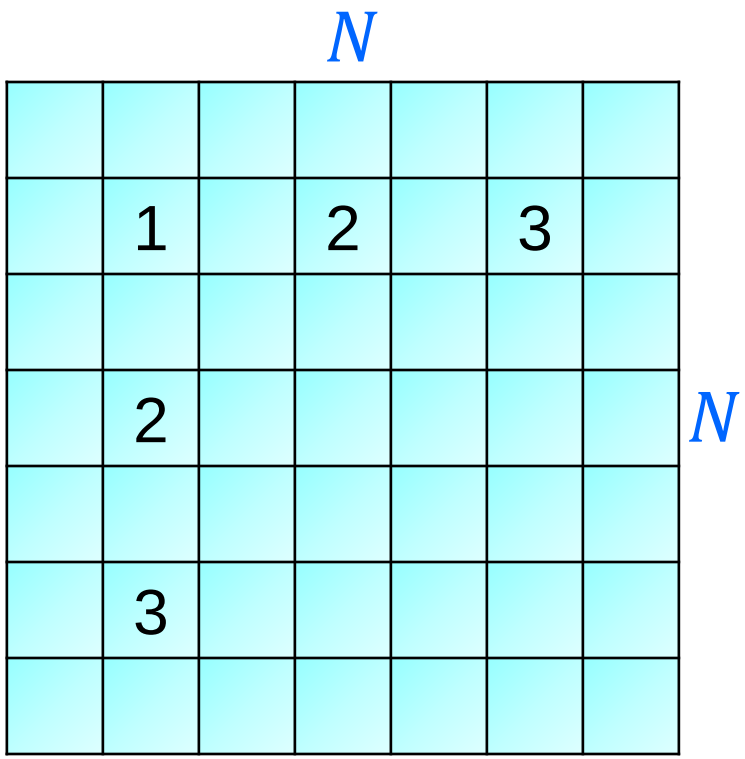


Output size $\frac{(N-F)}{stride} + 1$

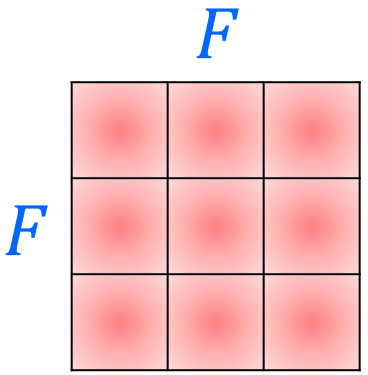
$N = 7, F = 3, stride = 1 \Rightarrow \frac{(7-3)}{1} + 1 = 5$

$\Rightarrow 5 \times 5$ output

A closer look at spatial dimensions



$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride$?

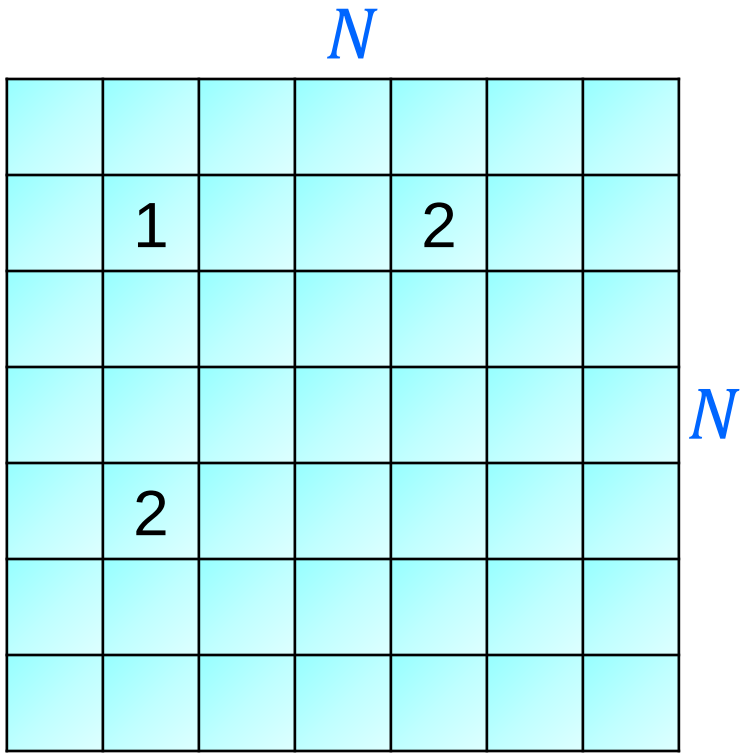


Output size $\frac{(N-F)}{stride} + 1$

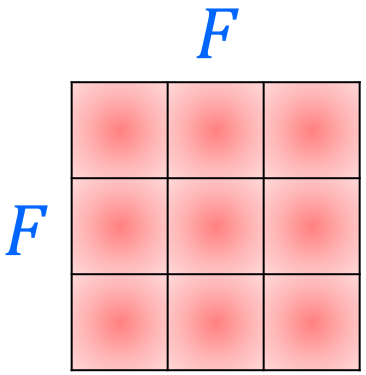
$N = 7, F = 3, stride = 2 \Rightarrow \frac{(7-3)}{2} + 1 = 3$

$\Rightarrow 3 \times 3$ output

A closer look at spatial dimensions



$N \times N$ input (spatially)
assume $F \times F$ filter
applied with $stride$?



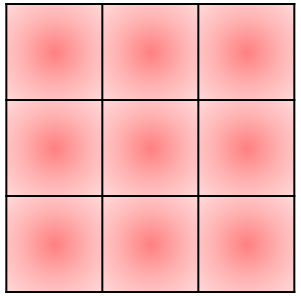
Output size $\frac{(N-F)}{stride} + 1$

$N = 7, F = 3, stride = 3 \Rightarrow \frac{(7-3)}{3} + 1 = 2{,}3$

$\Rightarrow$ doesn't fit!

Zero pad the border

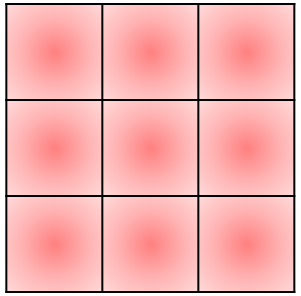
0	1	1	1	0	0	1
1	0	0	0	1	1	1
1	1	0	1	0	1	0
0	1	1	0	0	1	0
1	0	0	1	0	0	1
0	0	1	0	1	0	1
1	1	0	0	0	1	0



- Example:
  - + input 7×7,
  - + 3×3 filter,
  - + applied with stride 1,
  - + pad with 1 pixel border
- what is the output?

Zero pad the border

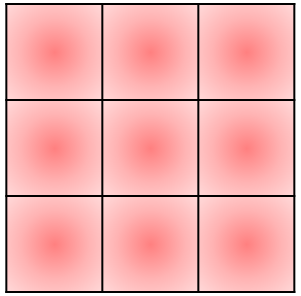
0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	1	0
0	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
0	0	1	1	0	0	1	0	0
0	1	0	0	1	0	0	1	0
0	0	0	1	0	1	0	1	0
0	1	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0



- Example:
  - + input 7×7,
  - + 3×3 filter,
  - + applied with stride 1,
  - + pad with 1 pixel border
- what is the output?

Zero pad the border

0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	1	0
0	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
0	0	1	1	0	0	1	0	0
0	1	0	0	1	0	0	1	0
0	0	0	1	0	1	0	1	0
0	1	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0

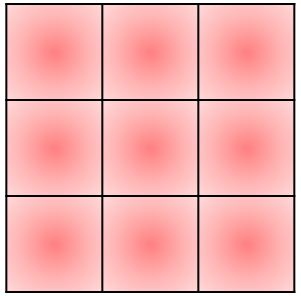


- Example:
  - + input 7×7,
  - + 3×3 filter,
  - + applied with stride 1,
  - + pad with 1 pixel border
- what is the output?

- Output size $\frac{(N-F)}{stride} + 1$

Zero pad the border

0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	1	0
0	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
0	0	1	1	0	0	1	0	0
0	1	0	0	1	0	0	1	0
0	0	0	1	0	1	0	1	0
0	1	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0



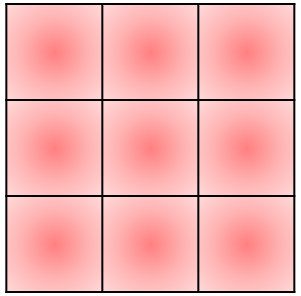
- Example:
  - + input 7×7,
  - + 3×3 filter,
  - + applied with stride 1,
  - + pad with 1 pixel border
- what is the output?

$\Longrightarrow 7 \times 7$ output

- Output size $\frac{(N-F)}{stride} + 1$

Zero pad the border

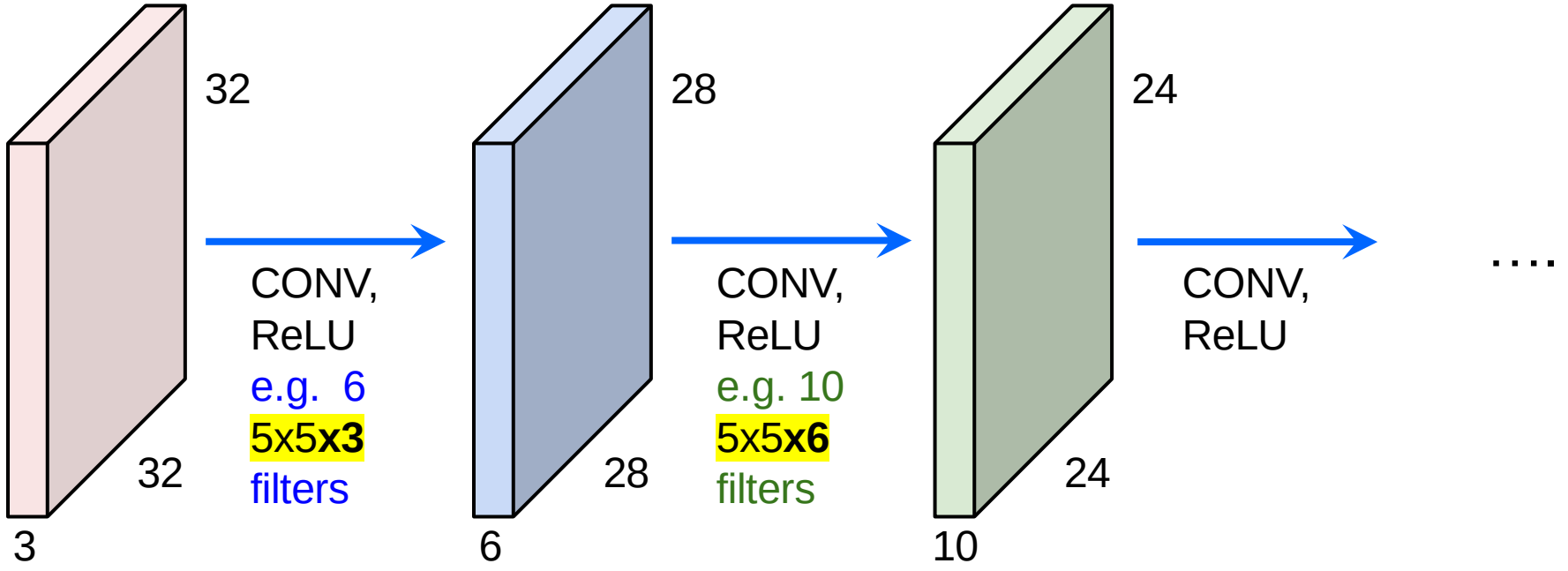
0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	1	0
0	1	0	0	0	1	1	1	0
0	1	1	0	1	0	1	0	0
0	0	1	1	0	0	1	0	0
0	1	0	0	1	0	0	1	0
0	0	0	1	0	1	0	1	0
0	1	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0



- in general, common to see CONV layers with stride 1, filters of size $F \times F$, and zero-padding with $\frac{(F-1)}{2}$ (will preserve size spatially).
- e.g.
  - $F = 3 \Rightarrow$ zero pad with 1.
  - $F = 5 \Rightarrow$ zero pad with 2.
  - $F = 7 \Rightarrow$ zero pad with 3.

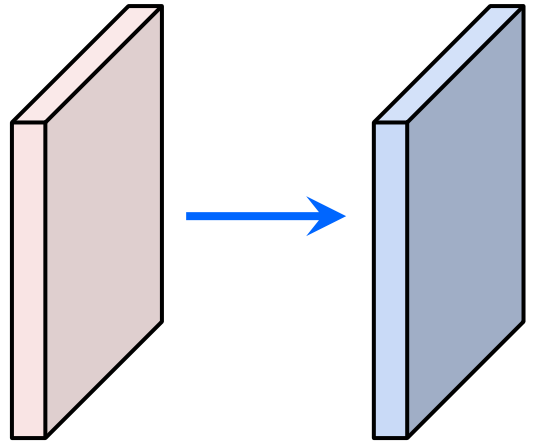
ConvNet

– E.g. 32×32 input convolved repeatedly with 5×5 filters shrinks volumes spatially! $(32 \rightarrow 28 \rightarrow 24 \rightarrow \cdots)$. Shrinking too fast is not good, doesn't work well.



Examples time

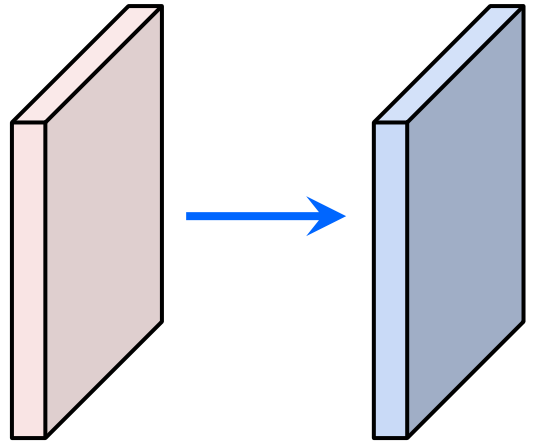
- Input volume: $32 \times 32 \times 3$.
- 10 5×5 filters with stride 1, pad 2.



- Output volume size: ?

Examples time

- Input volume: $32 \times 32 \times 3$.
- 10 5×5 filters with stride 1, pad 2.

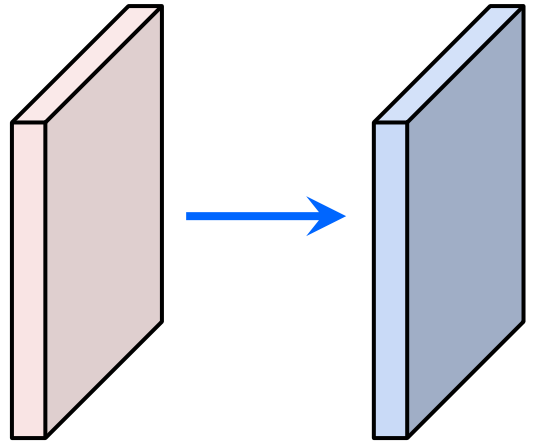


- Output volume size:

$\frac{(32+2\times2-5)}{1} + 1 = 32$ spatially, so $32 \times 32 \times 10$.

Examples time

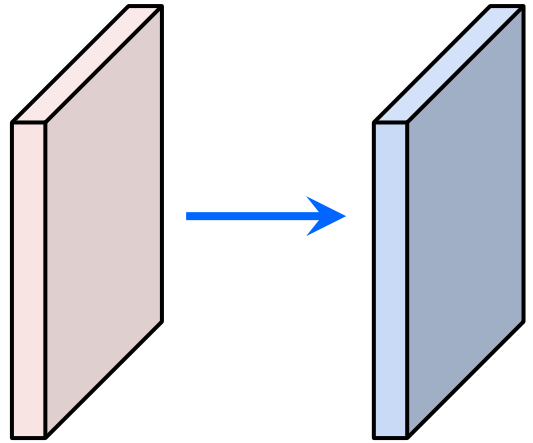
- Input volume: $32 \times 32 \times 3$.
- 10 5×5 filters with stride 1, pad 2.



- Number of parameters in this layer?

Examples time

- Input volume: $32 \times 32 \times 3$.
- 10 5×5 filters with stride 1, pad 2.



- Number of parameters in this layer?
- Each filter has $5 \times 5 \times 3 + 1 = 76$ (+1 for bias)

params $\Rightarrow 76 \times 10 = \mathbf{760}$.

Summary: the Conv Layer

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = \frac{(W_2 - F + 2P)}{S} + 1$
  - $H_2 = \frac{(H_2 - F + 2P)}{S} + 1$
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_2$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the $d - th$ depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the $d - th$ filter over the input volume with a stride of S, and then offset by $d - th$ bias.

Common settings: K = (powers of 2, e.g. 32, 64, 128, 512)

$F = 3, S = 1, P = 1$

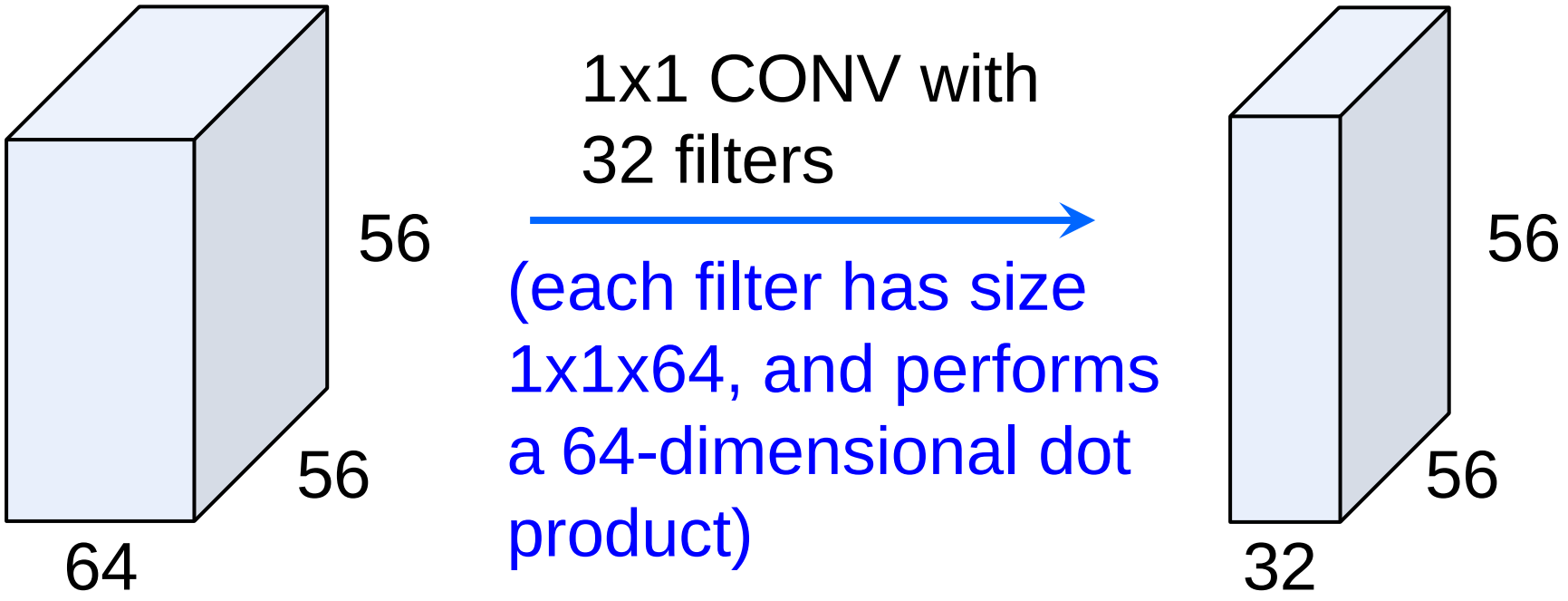
$F = 5, S = 1, P = 2$

$F = 5, S = 2, P =$? (whatever fits)

$F = 1, S = 1, P = 0$

ConvNet

– 1x1 convolution layers make perfect sense



Example: CONV layer in Torch

Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.

SpatialConvolution

```
module = nn.SpatialConvolution(nInputPlane, nOutputPlane, kW, kH, [dW], [dH], [padW], [padH])
```

Applies a 2D convolution over an input image composed of several input planes. The `input` tensor in `forward(input)` is expected to be a 3D tensor (`nInputPlane x height x width`).

The parameters are the following:

- `nInputPlane` : The number of expected input planes in the image given into `forward()` .
- `nOutputPlane` : The number of output planes the convolution layer will produce.
- `kW` : The kernel width of the convolution
- `kH` : The kernel height of the convolution
- `dW` : The step of the convolution in the width dimension. Default is `1` .
- `dH` : The step of the convolution in the height dimension. Default is `1` .
- `padW` : The additional zeros added per width to the input planes. Default is `0` , a good number is `(kW-1)/2` .
- `padH` : The additional zeros added per height to the input planes. Default is `padW` , a good number is `(kH-1)/2` .

Note that depending of the size of your kernel, several (of the last) columns or rows of the input image might be lost. It is up to the user to add proper padding in images.

If the input image is a 3D tensor `nInputPlane x height x width` , the output image size will be `nOutputPlane x oheight x owidth` where

```
owidth  = floor((width  + 2*padW - kW) / dW + 1)
oheight = floor((height + 2*padH - kH) / dH + 1)
```

ConvNet

Summary: To summarize, the Conv Layer:

– Accepts a volume of size $W_1 \times H_1 \times D_1$

– Requires four hyperparameters:

+ Number of filters K,

+ their spatial extent F,

+ the stride S,

+ the amount of zero padding P.

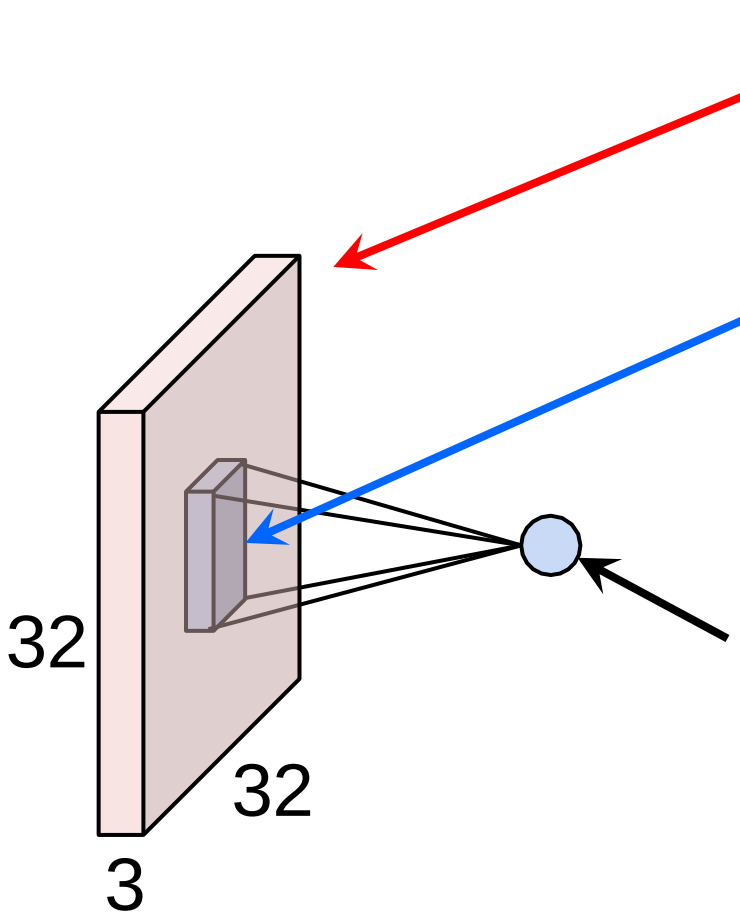
Example: CONV layer in Caffe

Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.

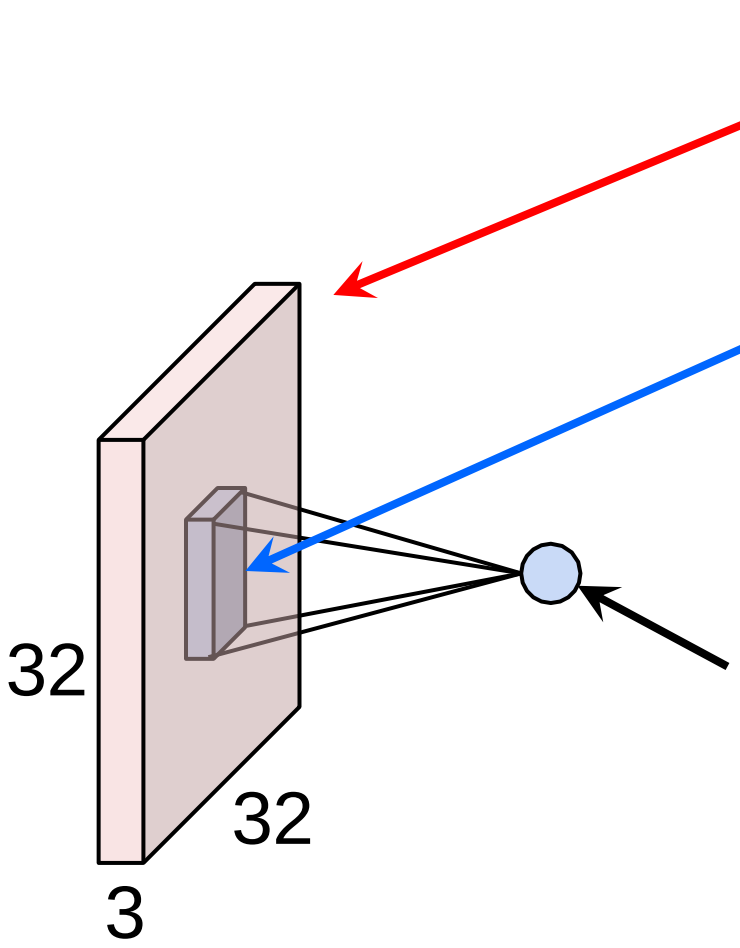
```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  # learning rate and decay multipliers for the filters
  param { lr_mult: 1 decay_mult: 1 }
  # learning rate and decay multipliers for the biases
  param { lr_mult: 2 decay_mult: 0 }
  convolution_param {
    num_output: 96     # learn 96 filters
    kernel_size: 11    # each filter is 11x11
    stride: 4          # step 4 pixels between each filter application
    weight_filler {
      type: "gaussian" # initialize the filters from a Gaussian
      std: 0.01        # distribution with stdev 0.01 (default mean: 0)
    }
    bias_filler {
      type: "constant" # initialize the biases to zero (0)
      value: 0
    }
  }
}
```

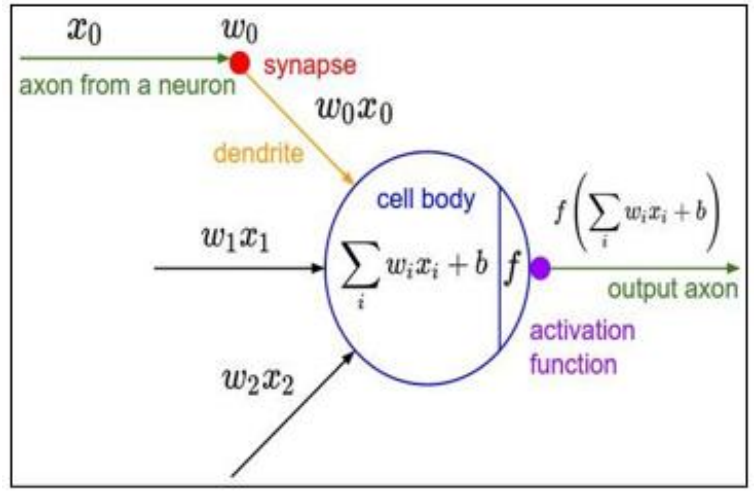
The brain/neuron view of CONV Layer



1 number:
the result of taking a dot product between the filter and a small $5 \times 5 \times 3$ chunk of the image (i.e. $5 \times 5 \times 3 = 75 - dimensional$ dot product)

The brain/neuron view of CONV Layer



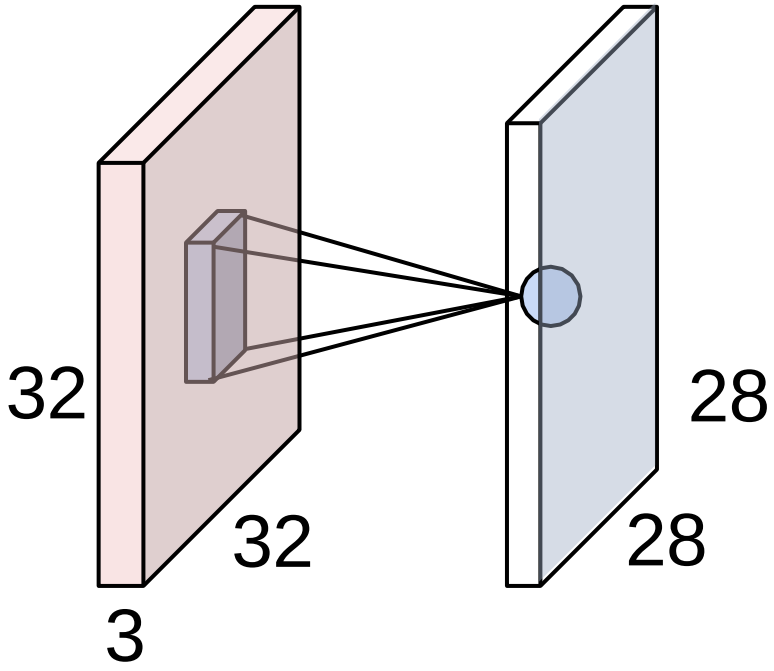


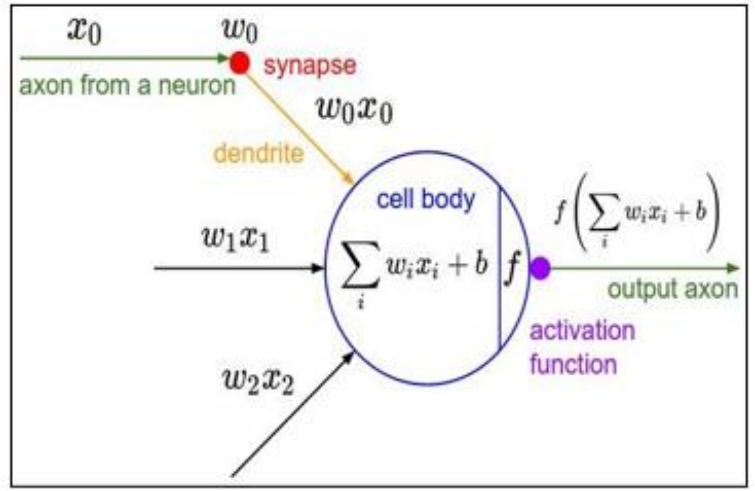
It's just a neuron with local connectivity...

1 number:

the result of taking a dot product between the filter and a small $5 \times 5 \times 3$ chunk of the image (i.e. $5 \times 5 \times 3 = 75 - dimensional$ dot product)

The brain/neuron view of CONV Layer



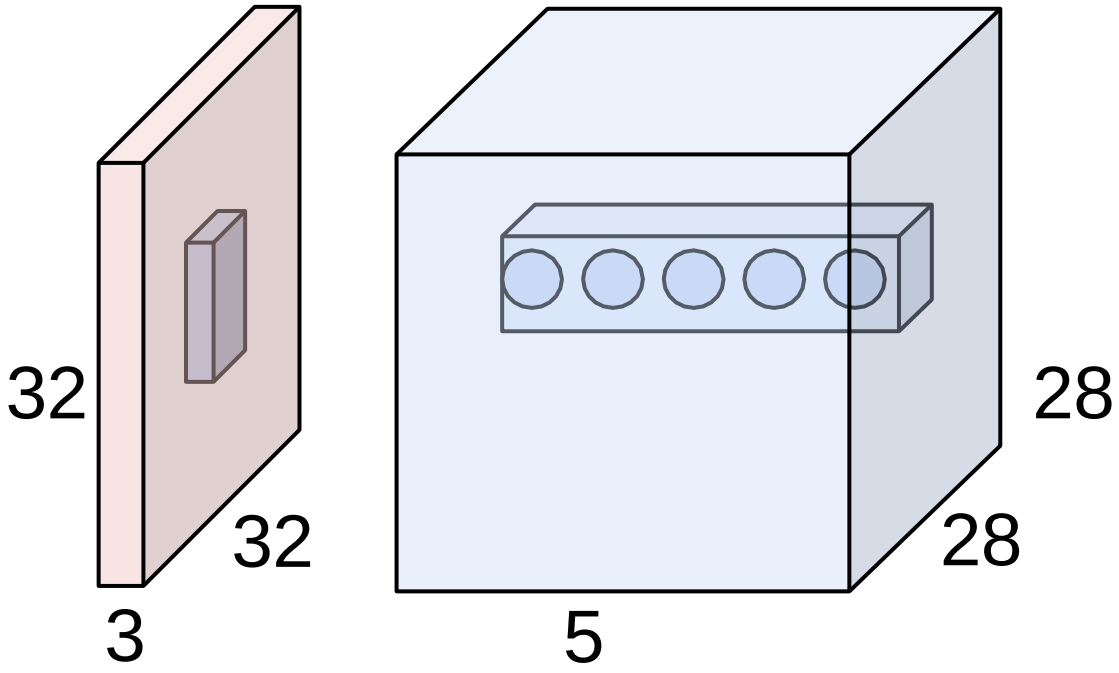


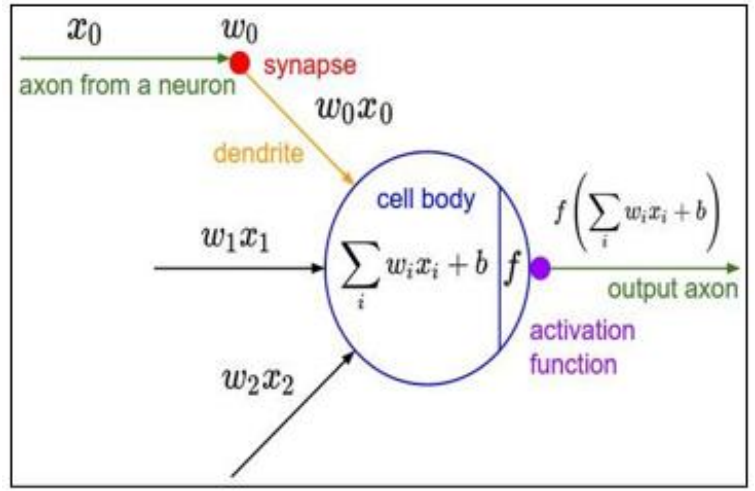
An activation map is a 28x28 sheet of neuron outputs:

1. Each is connected to a small region in the input
2. All of them share parameters

"5x5 filter" -> "5x5 receptive field for each neuron"

The brain/neuron view of CONV Layer





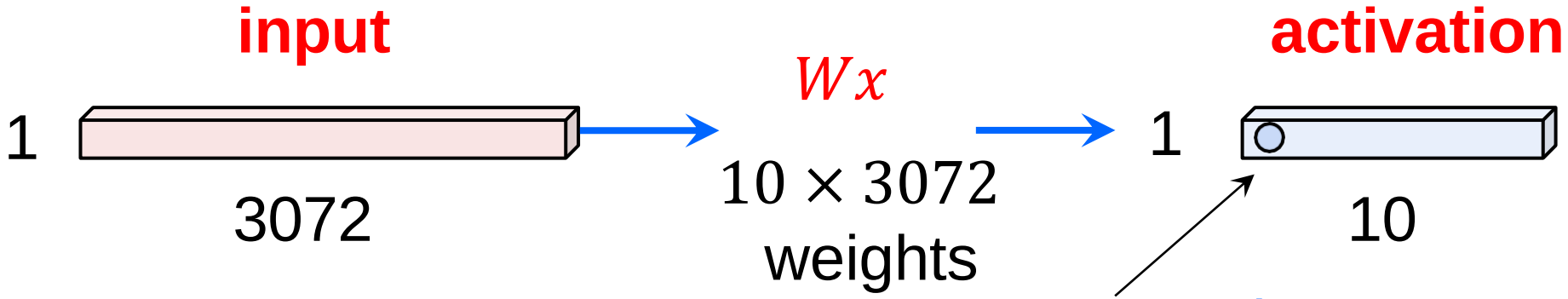
E.g. with 5 filters,
CONV layer consists of neurons arranged in a 3D grid (28x28x5)

There will be 5 different neurons all looking at the same region in the input volume

Reminder: Fully Connected Layer

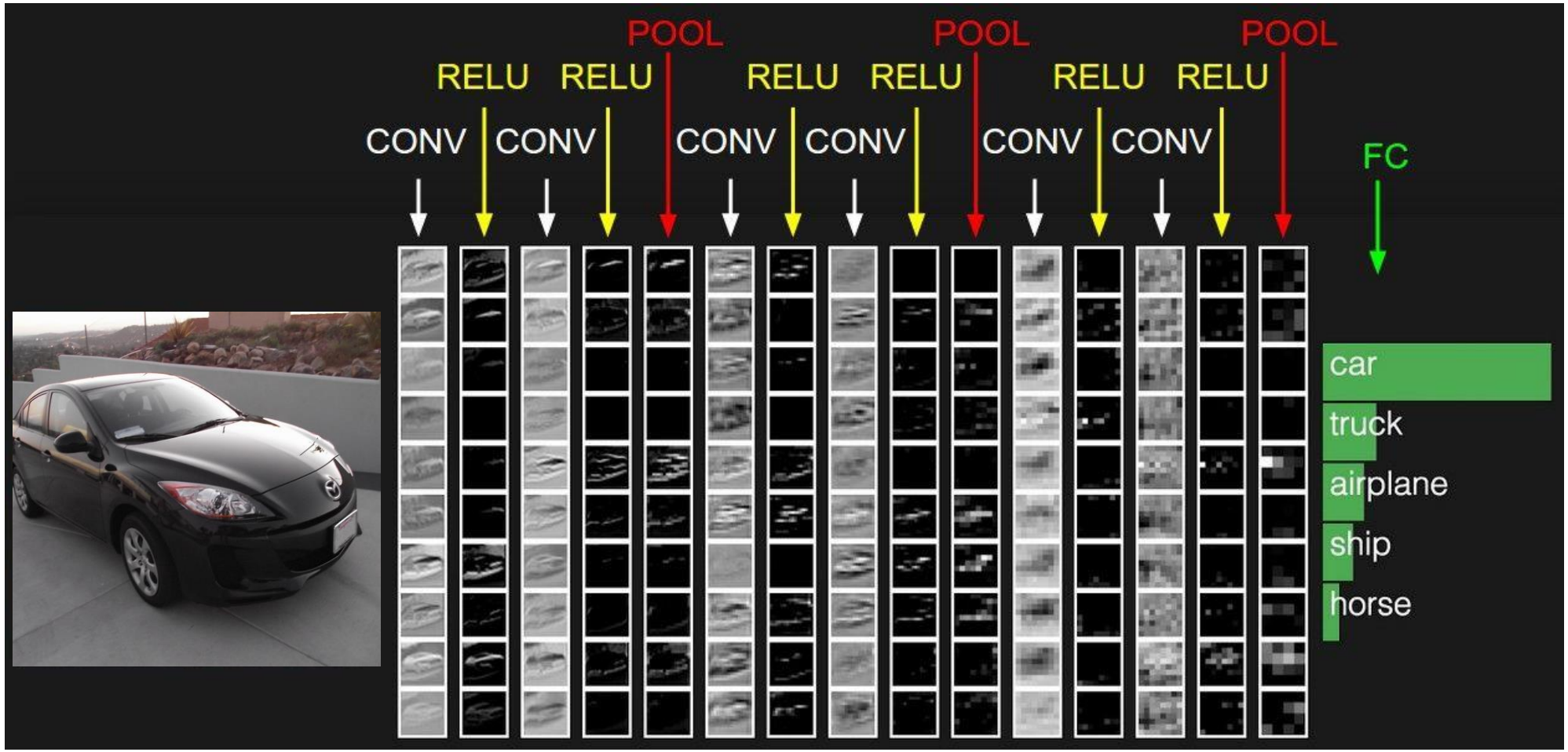
– Image: $32 \times 32 \times 3 \Longrightarrow$ stretch to 3072×1.

Each neuron looks at the full input volume



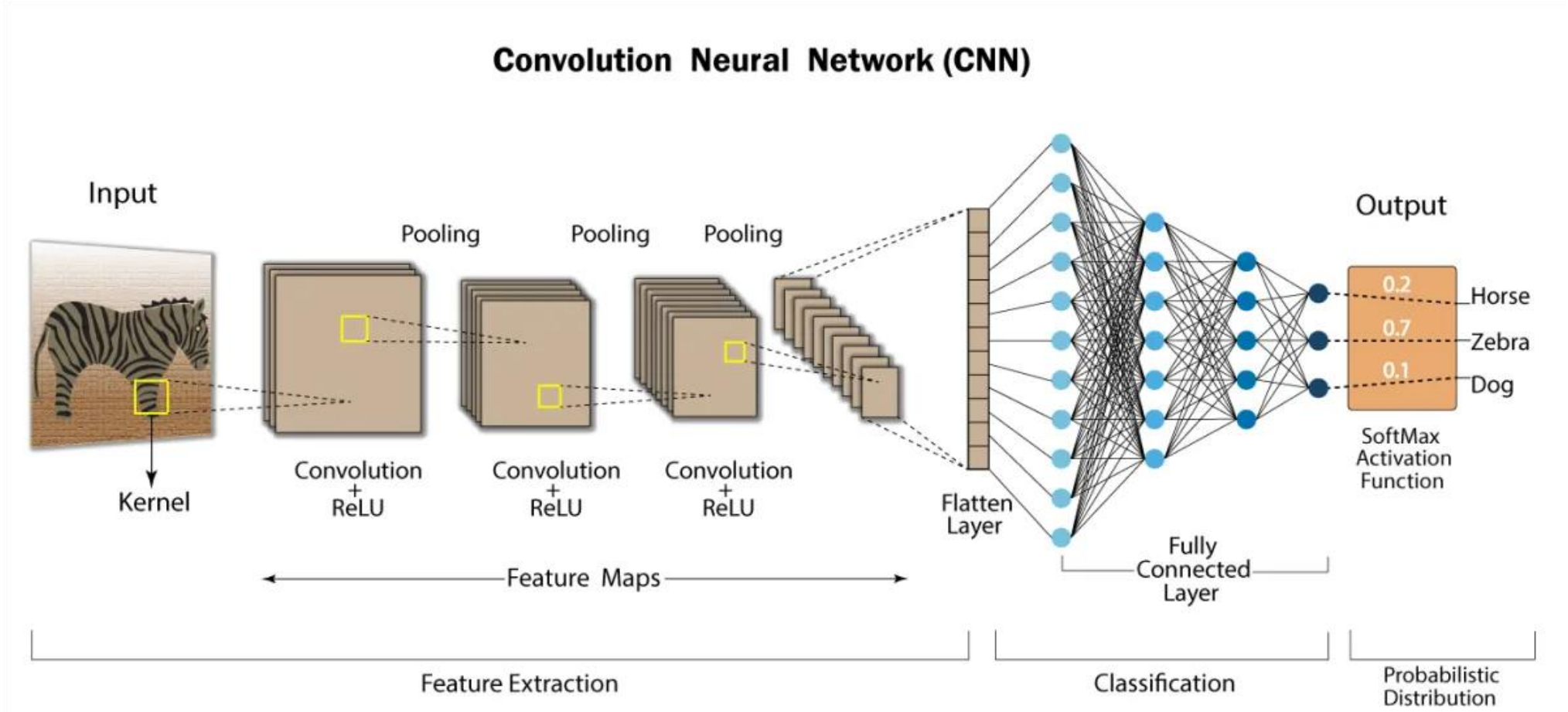
1 number:
the result of taking a dot product between a row of W and the input (a 3072 - dimensional dot product)

ConvNet


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

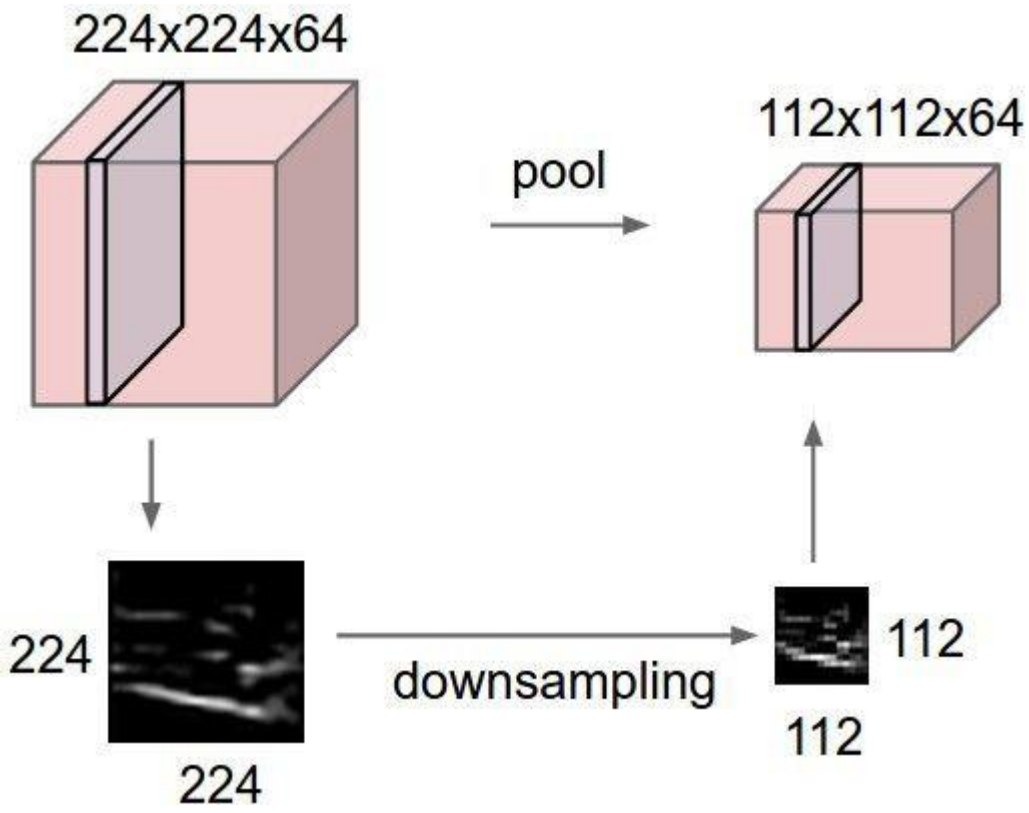
POOLING LAYER

Convolutional neural networks



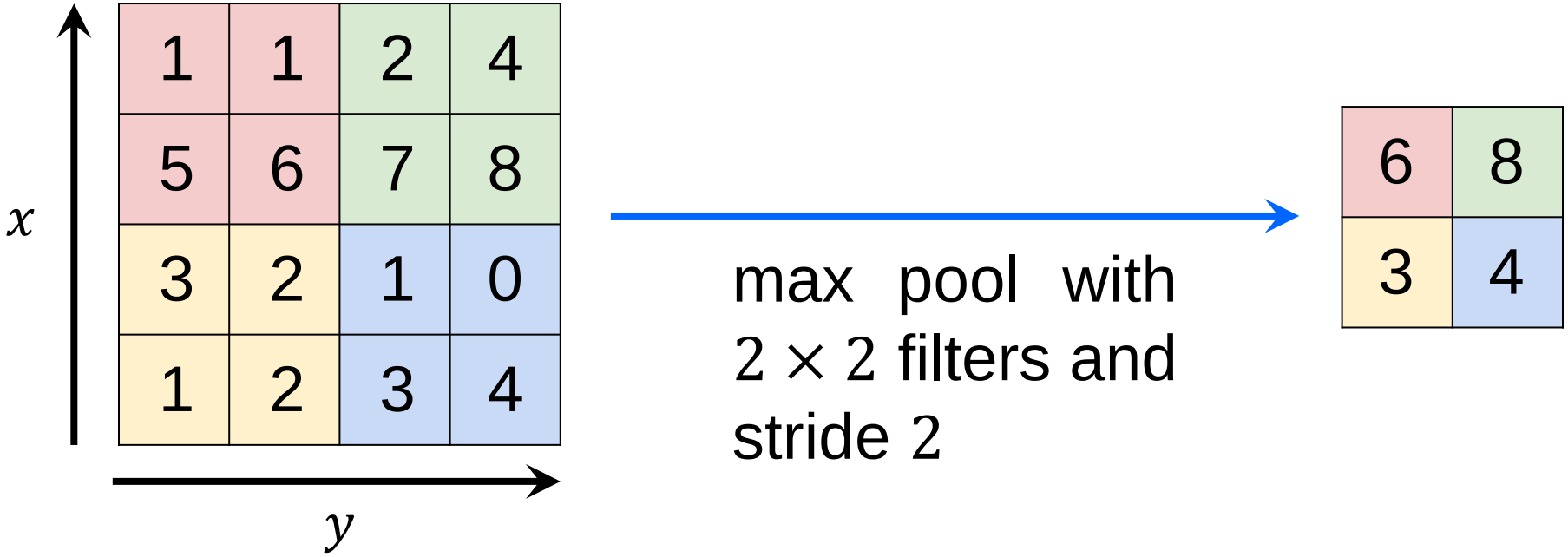
Pooling layer

- Makes the representations smaller and more manageable
- Làm cho các biểu diễn nhỏ hơn và dễ quản lý hơn
- Operates over each activation map independently:
- Hoạt động độc lập trên từng activation map:



Max pooling

Single depth slice



Max pooling

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires three hyperparameters:
  + Their spatial extent F,
  + The stride S,
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  + $W_2 = (W_1 - F)/S + 1$
  + $H_2 = (H_1 - F)/S + 1$
  + $D_2 = D_1$

- Introduces zero parameter since it computes a fixed function of the input.
- Note that it is not common to use zero-padding for Pooling layers.

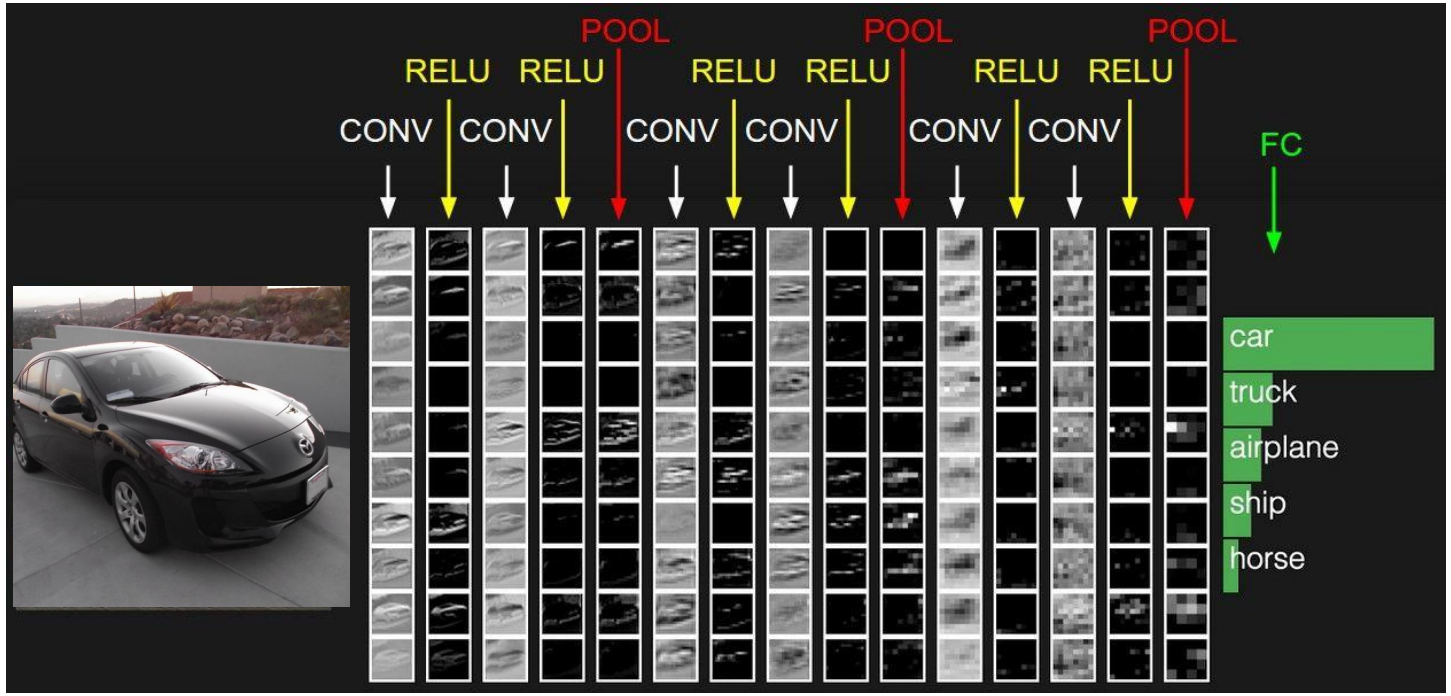
Common settings:

$F = 2, \quad S = 2$

$F = 3, \quad S = 2$

Max pooling

Fully Connected Layer



– Contains neurons that connect to the entire input volume, as in ordinary Neural Networks.

ConvNetJS demo: training on CIFAR-10

ConvNetJS CIFAR-10 demo

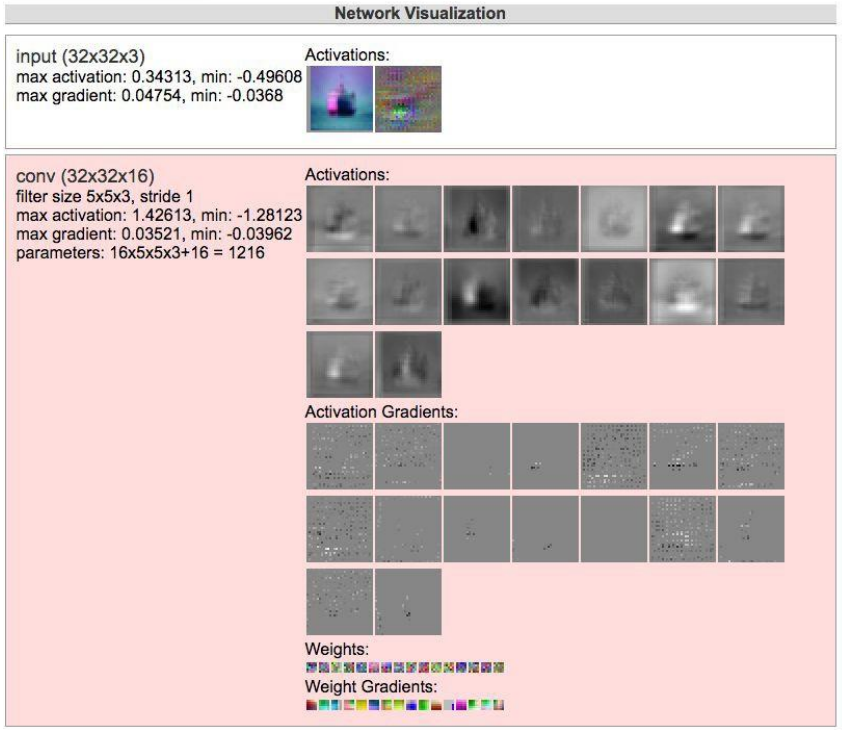
Description

This demo trains a Convolutional Neural Network on the CIFAR-10 dataset in your browser, with nothing but Javascript. The state of the art on this dataset is about 90% accuracy and human performance is at about 94% (not perfect as the dataset can be a bit ambiguous). I used this python script to parse the original files (python version) into batches of images that can be easily loaded into page DOM with img tags.

This dataset is more difficult and it takes longer to train a network. Data augmentation includes random flipping and random image shifts by up to 2px horizontally and verically.

By default, in this demo we're using Adadelta which is one of per-parameter adaptive step size methods, so we don't have to worry about changing learning rates or momentum over time. However, I still included the text fields for changing these if you'd like to play around with SGD+Momentum trainer.

Report questions/bugs/suggestions to @karpathy.



http://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html

Summary

- ConvNets stack CONV, POOL, FC layers.
- ConvNets là dãy các lớp CONV, POOL, FC.
- Trend towards smaller filters and deeper architectures.
- Xu hướng hướng tới các bộ lọc nhỏ hơn và kiến trúc mạng sâu hơn.
- Trend towards getting rid of POOL/FC layers (just CONV).
- Xu hướng loại bỏ các lớp POOL/FC (chỉ CONV).

Summary

- Typical architectures look like – Kiến trúc CNN điển hình:
  $[(\mathrm{CONV} - \mathrm{RELU}) * \mathrm{N} - \mathrm{POOL?}] * \mathrm{M} - (\mathrm{FC} - \mathrm{RELU}) * \mathrm{K}, \mathrm{SOFTMAX}$
- Where – Trong đó:
  + N is usually up to ~5 (small $1 - 2$, average $2 - 3$, complex $3 - 5$).
  + M is large (small $2 - 3$, average $3 - 5$, complex $5 - 10$).
  + $0 \leq K \leq 3$ (small $1 - 2$, average $2 - 3$, complex $2 - 3$).
- Recent advances ResNet/GoogLeNet challenge this paradigm.
- ResNet/GoogLeNet thách thức mô hình này.

Chúc các bạn học tốt

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN TP.HCM

Nhóm UIT-Together

TS. Nguyễn Tấn Trần Minh Khang