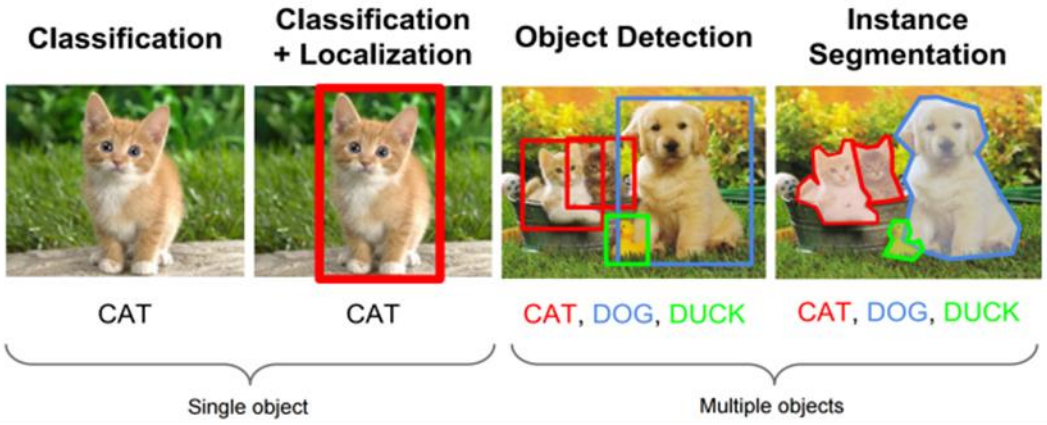
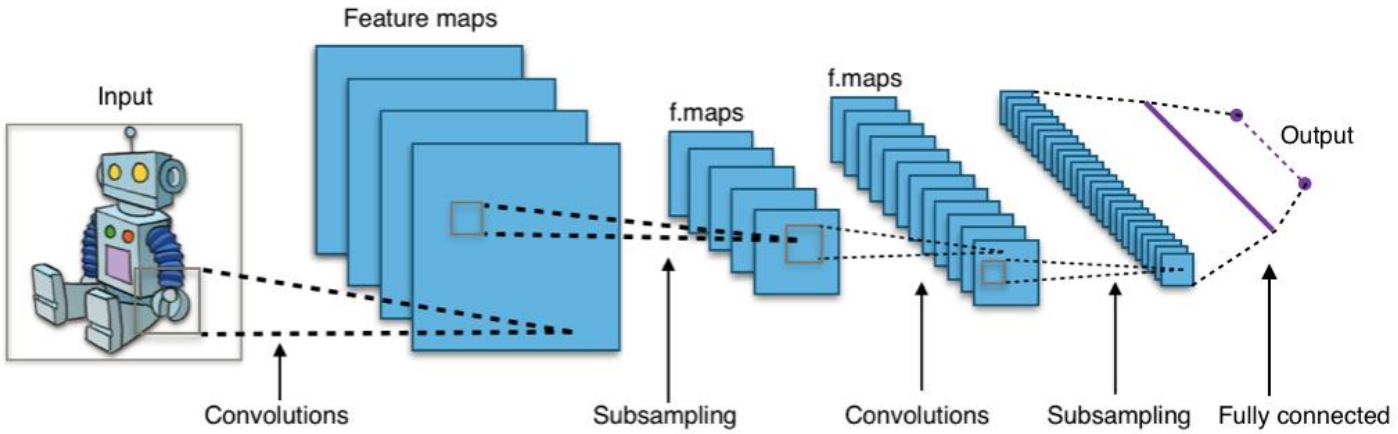
Convolutional Neural Network



https://www.kdnuggets.com/2018/09/object-detection-image-classification-yolo.html

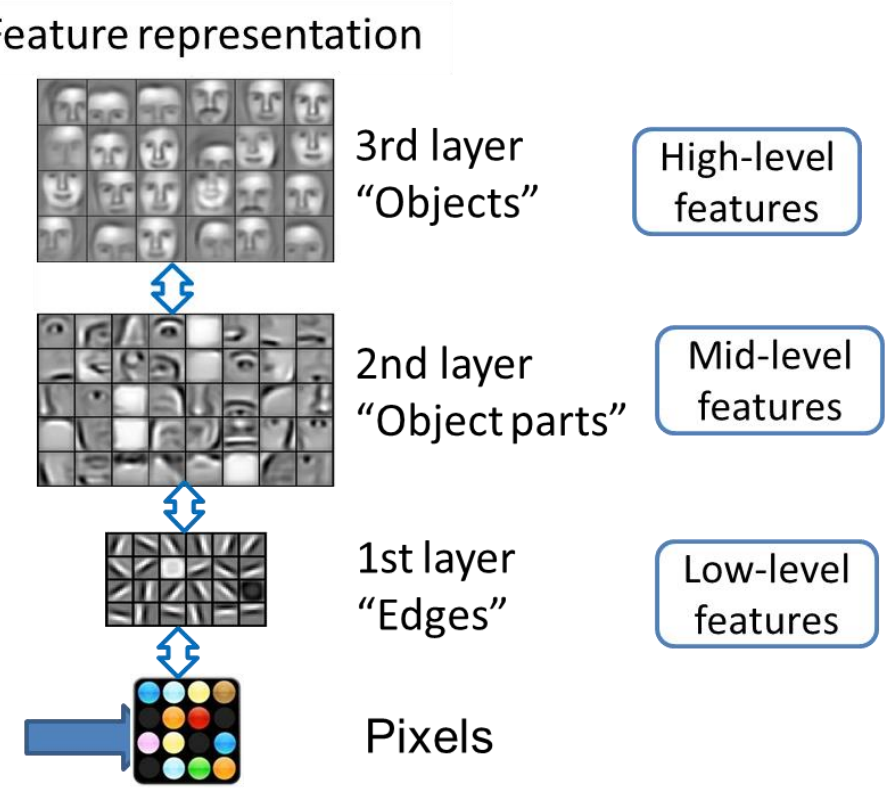


October 1, 1, 5, 2021

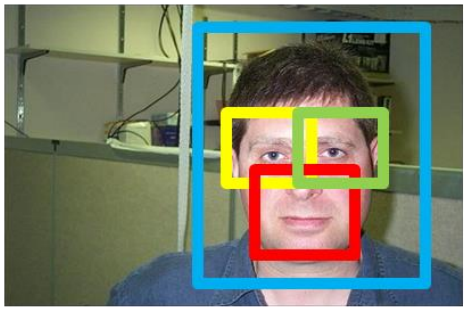
Learning a Hierarchy of Feature Extractors

Goal: Learn **useful higher-level features** from images

- Each layer of hierarchy extracts features from output of previous layer

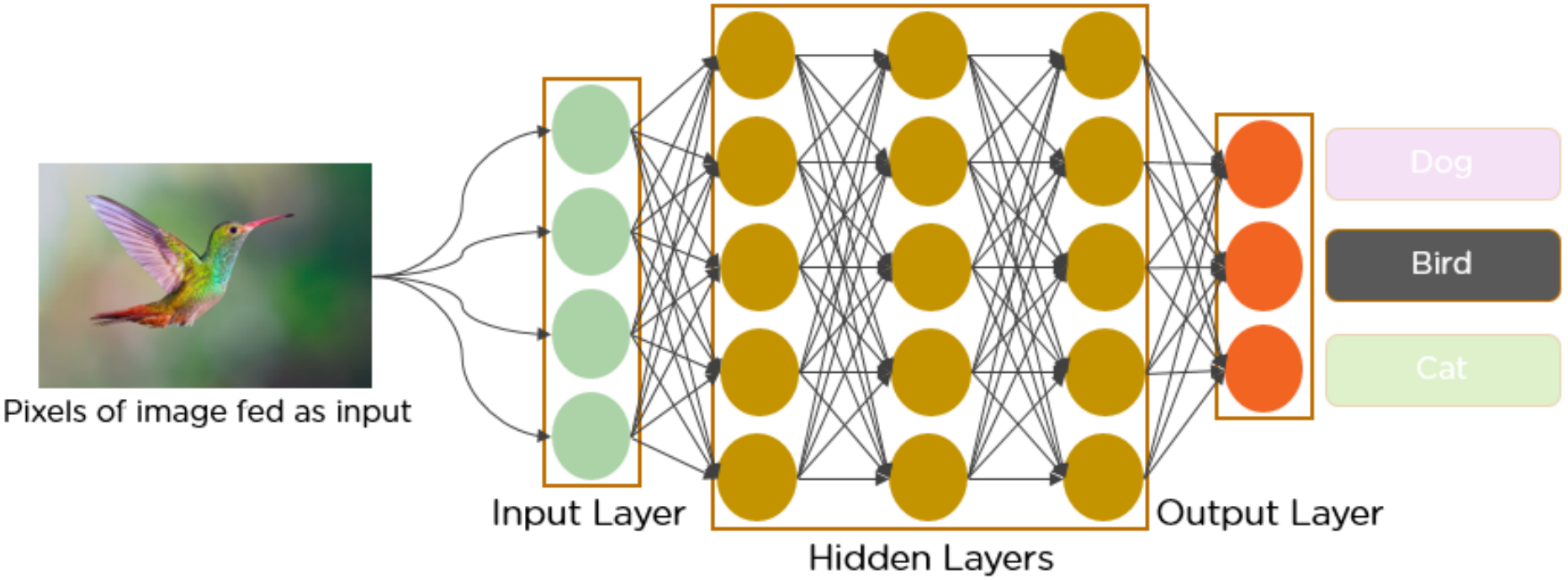


Input data



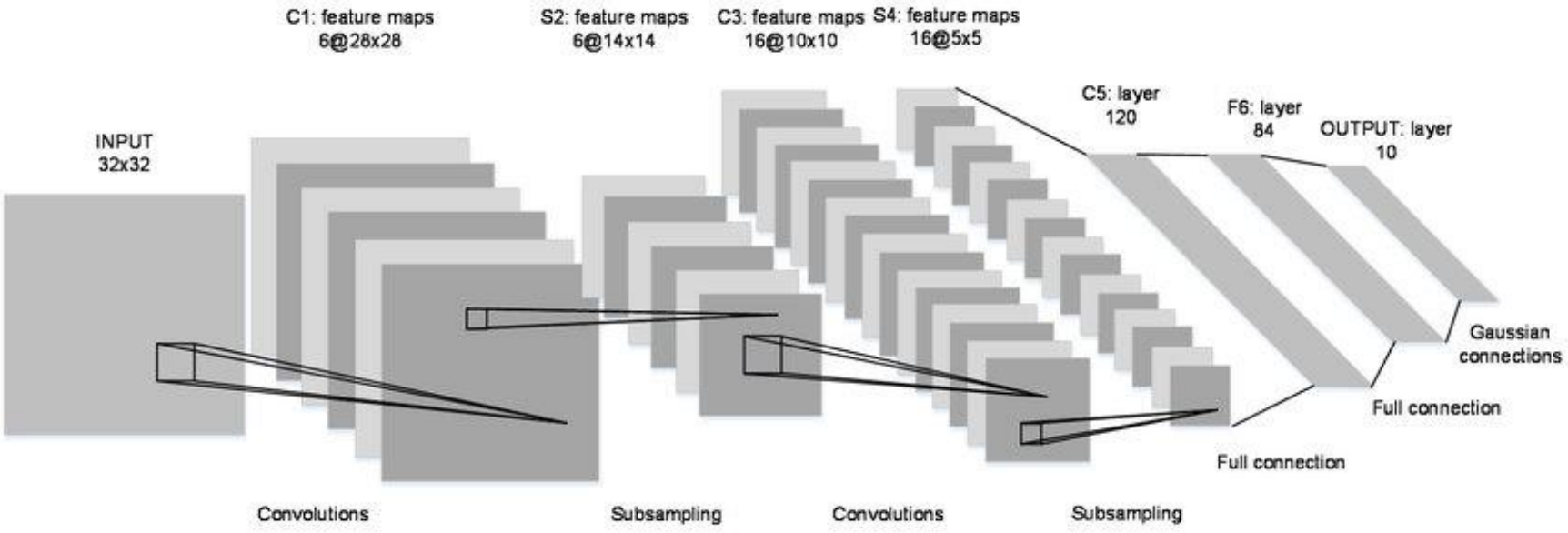
Lee et al., ICML 2009; CACM 2011

Convolutional Neural Network (CNN)



https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

Background of CNNs

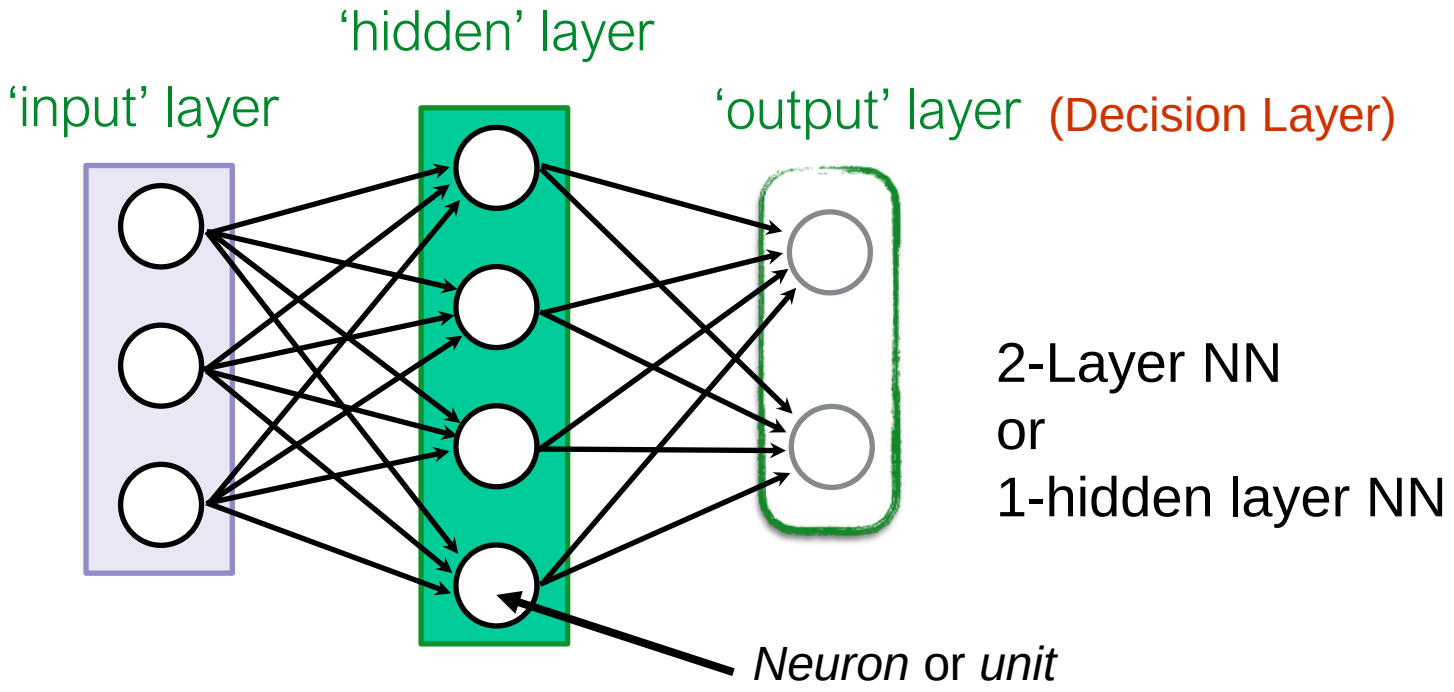


https://www.researchgate.net/figure/The-LeNet-5-Architecture-a-convolutional-neural-network_fig4_321586653

Connect a bunch of perceptrons together …

Neural Network

a collection of connected perceptrons

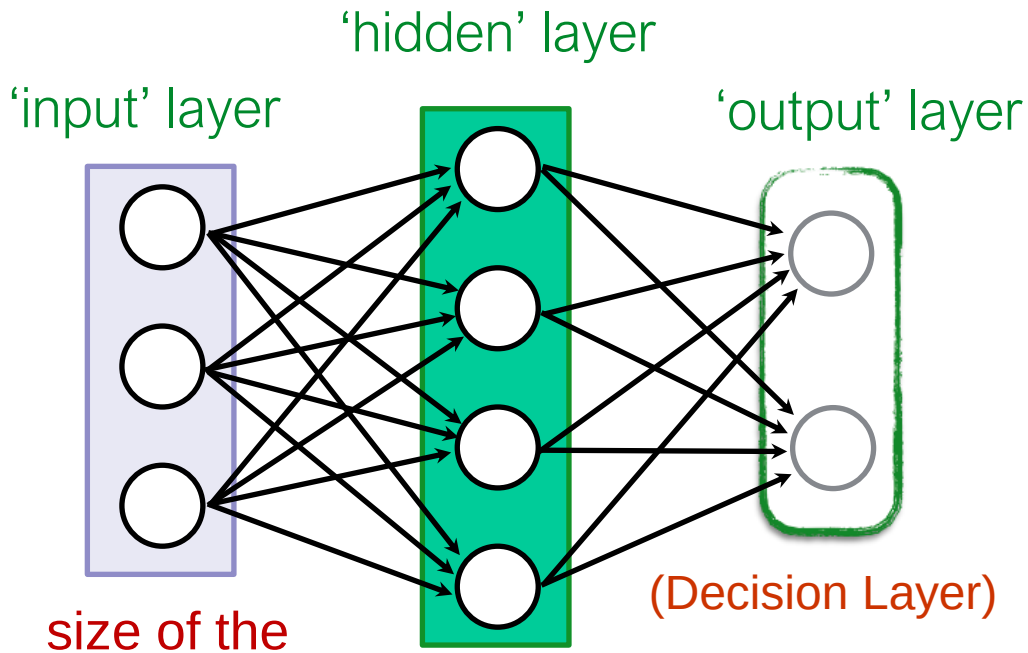


…also called a Multi-layer Perceptron (MLP)

Neural Network for Image Data

- Suppose the input vector is an image.
- You have to classify if the picture contains a bird or not.

256 × 256

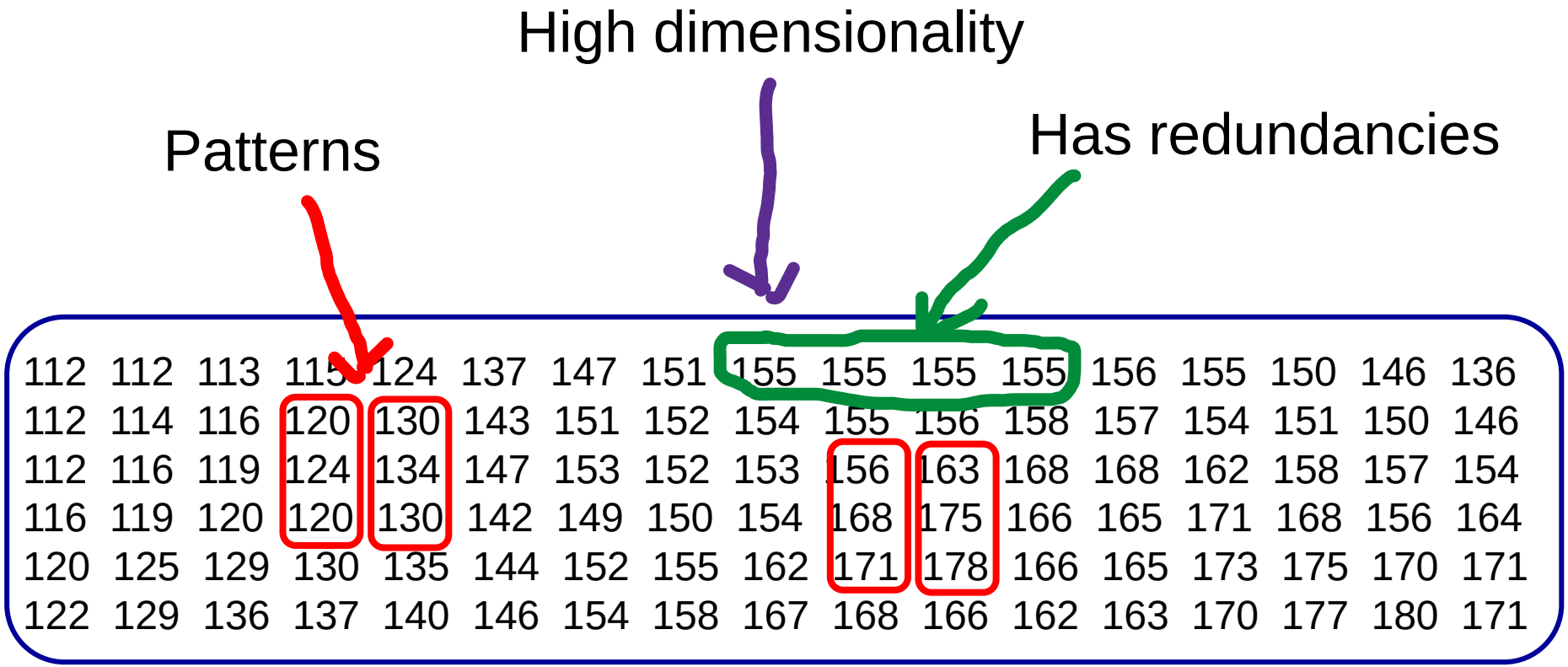


size of the input layer?

how many weight parameters, if hidden layer contains 1024 units?

Source: Prof. A. Ojha, IIITDM Jabalpur, Prof. Ioannis Gkioulekas, Carnegie Mellon University

Image Data

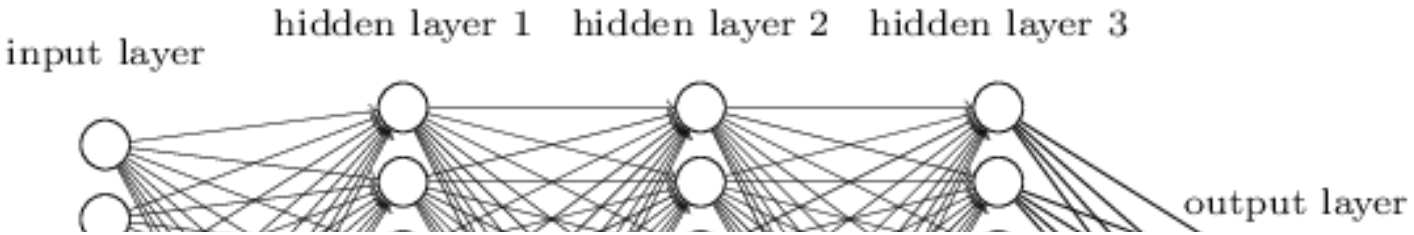


Pixel values of a block of an image

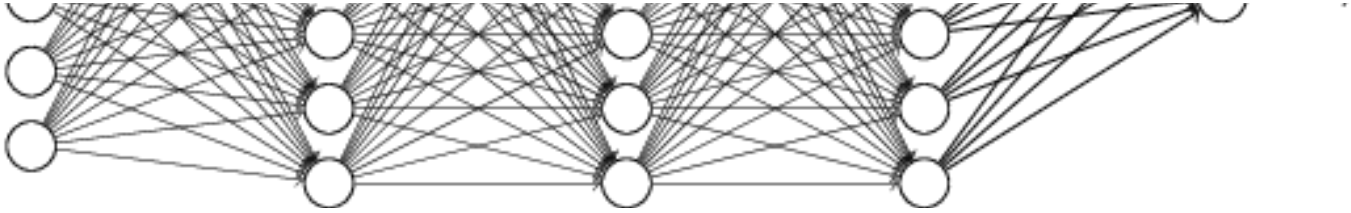
Source: Prof. A. Ojha, IIITDM Jabalpur

Smaller Network: CNN

- Main questions:
- Do we need all the edges in this fully connected model?
- Can some of the weights be shared?



CNN are just neural networks
with
3D activations and weight sharing

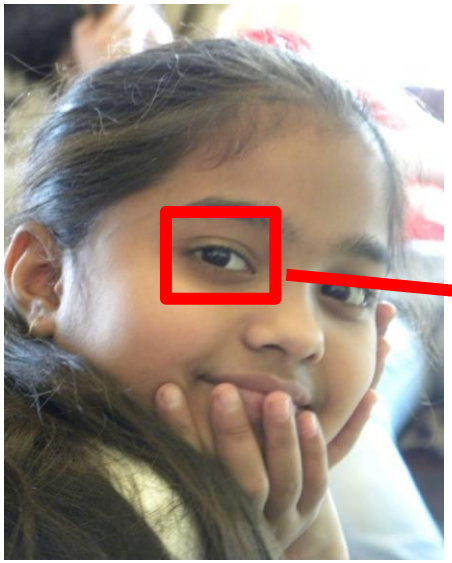


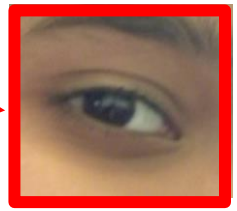
Source: Prof. A. Ojha, IIITDM Jabalpur

Image Understanding

- Patterns to learn can be very small compared to the image size.
- Example: Eye detector filter in images.

A small pattern detector can be used in place of the entire weight parameter matrix.



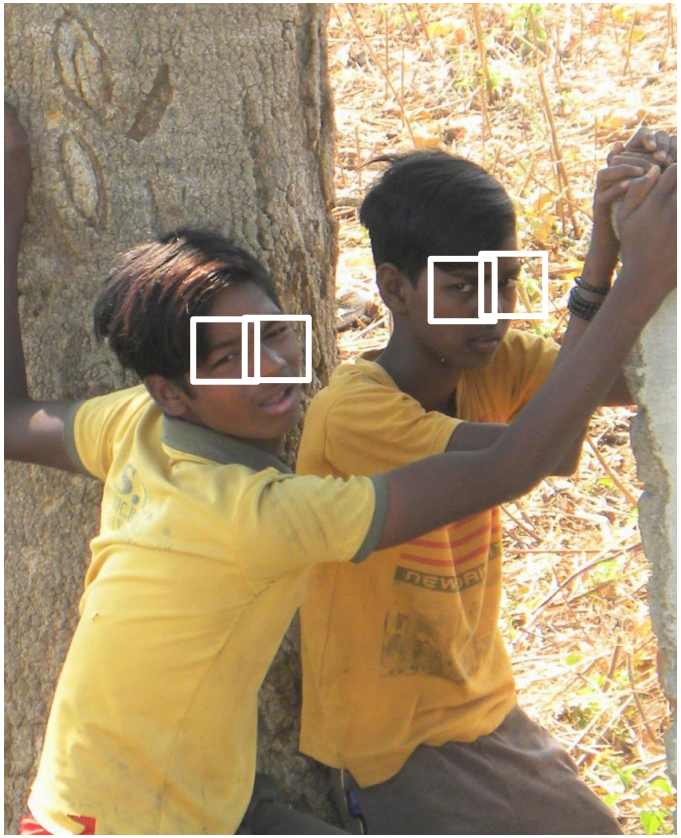


"eye" detector

Source: Prof. A. Ojha, IIITDM Jabalpur

Pattern Learning

- Same pattern can be at different positions in images.
- For examples – eyes.



Same pattern detector can move around and detect eyes at different positions !

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution Neural Networks (CNN)

- Use parameter sharing.
- Small pattern detectors called **filters** are used to convolve over the entire image.
- These filters are learned through NN training in the same way as in fully connected networks.
- Just like a hidden layer in a fully connected layer, convolution layers are used in CNNs.
- To handle large size of image data, pooling layers are introduced.

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution Operation

- Convolution with an image block outputs a real number.
- When we slide this filter over the entire image, shifting block by block, we get an array of real numbers.

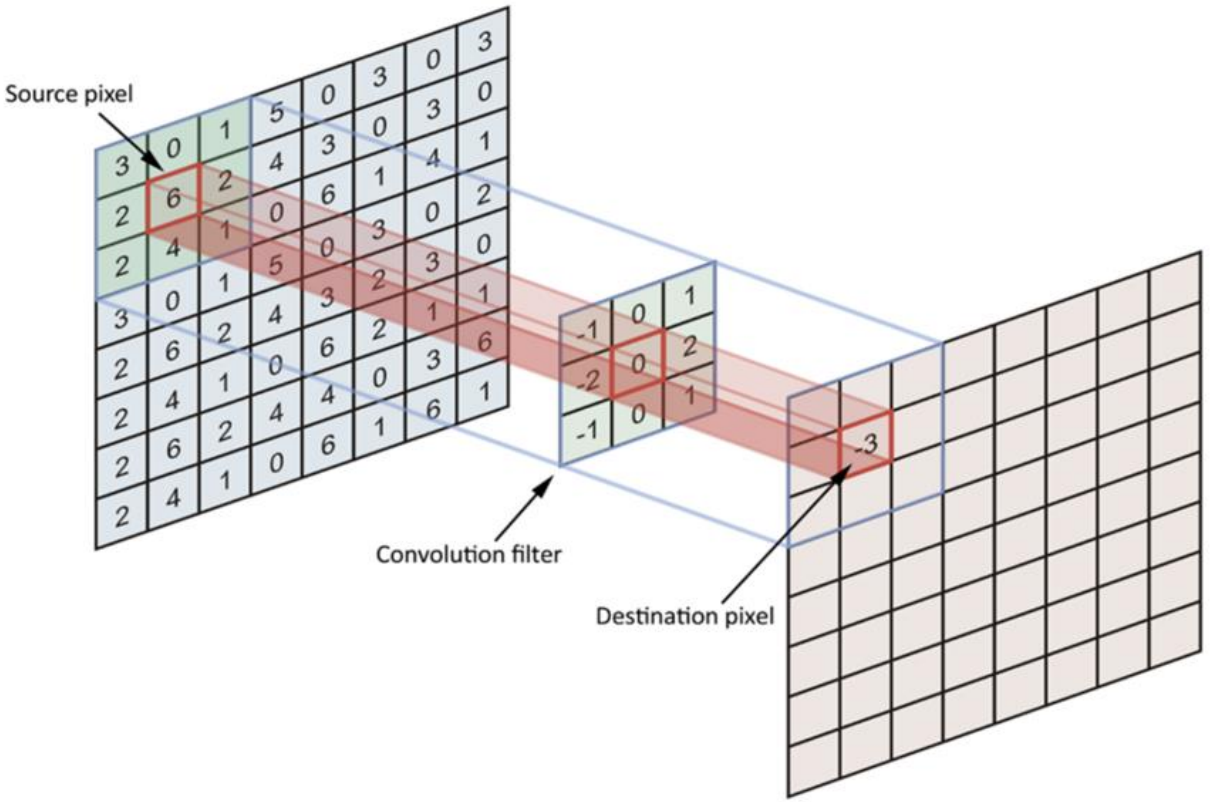


Figure source: https://miro.medium.com/max/1950/1*p-_47puSuVNmRJnOXYPQCg.png

Convolutional Layer

A convolutional layer has a number of filters that perform convolutional operation.

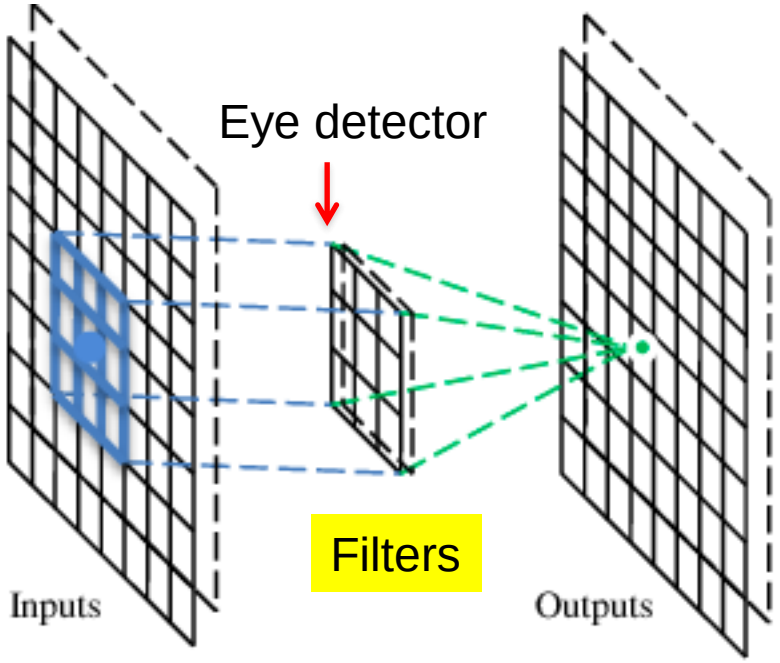


Figure source: Prof. Ming Li Lecture slides, University of Waterloo, Canada

Convolution Layer Example

5	1	7	7	8	1
7	2	5	7	8	1
5	3	5	4	4	2
4	4	3	3	2	1
3	3	4	7	8	1
5	7	8	2	3	1

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

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution Layer Example

5	1	7	7	8	1
7	2	5	7	8	1
5	3	5	4	4	2
4	4	3	3	2	1
3	3	4	7	8	1
5	7	8	2	3	1

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

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution Layer Example

5	1	7	7	8	1
7	2	5	7	8	1
5	3	5	4	4	2
4	4	3	3	2	1
3	3	4	7	8	1
5	7	8	2	3	1

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

$$5 \times 1 + 1 \times 0 + 7 \times (-1) + 7 \times 2 + 2 \times 0 + 5 \times (-2) + 5 \times 1 + 3 \times 0 + 5 \times (-1) = 2$$

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution Layer Example

5	1	7	7	8	1
7	2	5	7	8	1
5	3	5	4	4	2
4	4	3	3	2	1
3	3	4	7	8	1
5	7	8	2	3	1

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

$$x = \begin{bmatrix} 5 \\ 1 \\ 7 \\ 7 \\ 2 \\ 5 \\ 5 \\ 3 \\ 5 \end{bmatrix} \qquad W = \begin{bmatrix} 1 \\ 0 \\ -1 \\ 1 \\ 0 \\ -1 \\ 1 \\ 0 \\ -1 \end{bmatrix}$$

- Convert the image block and filter to single column vectors and calculate their dot product.

$$W^T x = 5 \times 1 + 1 \times 0 + 7 \times (-1) + 7 \times 2 + 2 \times 0 + 5 \times (-2) + 5 \times 1 + 3 \times 0 + 5 \times (-1) = 2$$

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution Layer Example

5	1	7	7	8	1
7	2	5	7	8	1
5	3	5	4	4	2
4	4	3	3	2	1
3	3	4	7	8	1
5	7	8	2	3	1

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

Slide the filter by one pixel and repeat the process.

$$1 \times 1 + 7 \times 0 + 7 \times (-1) + 2 \times 2 + 5 \times 0 + 7 \times (-2) + 3 \times 1 + 5 \times 0 + 4 \times (-1) = -17$$

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution Layer Example

5	1	7	7	8	1
7	2	5	7	8	1
5	3	5	4	4	2
4	4	3	3	2	1
3	3	4	7	8	1
5	7	8	2	3	1

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

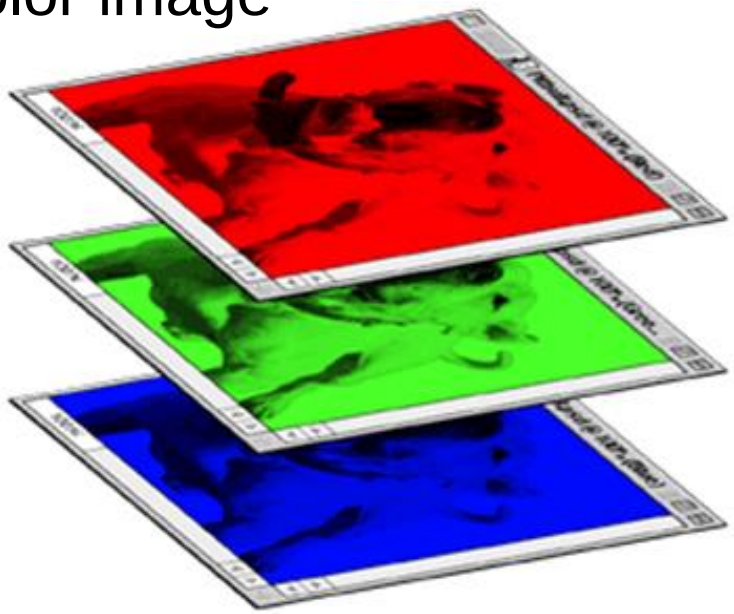
and the process continues in this way…

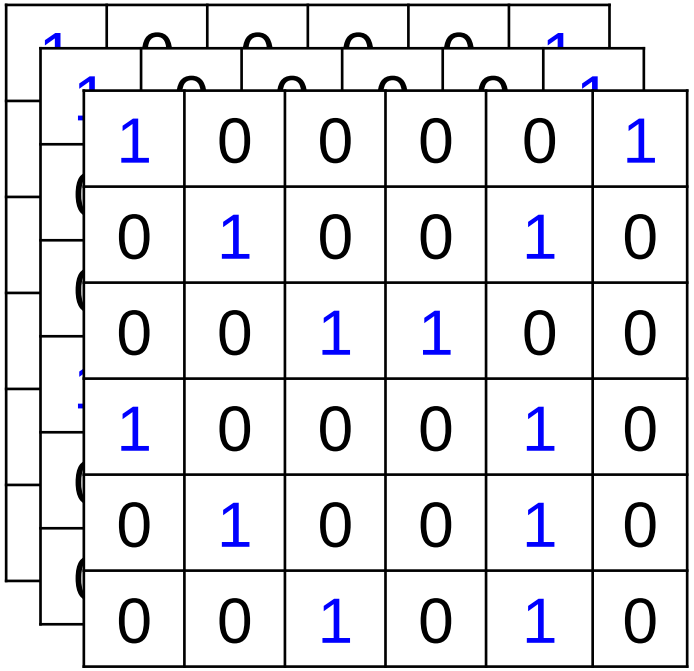
$7 \times 1 + 7 \times 0 + 8 \times (-1) + 5 \times 2 + 7 \times 0 + 8 \times (-2) + 5 \times 1 + 4 \times 0 + 4 \times (-1) = -6$

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution on RGB Images

Color image





Filter

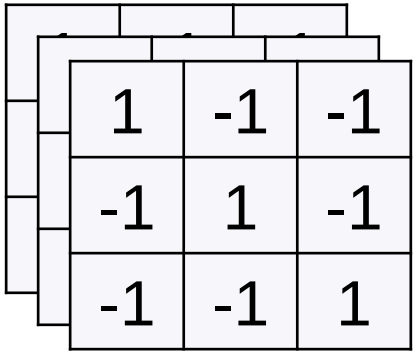
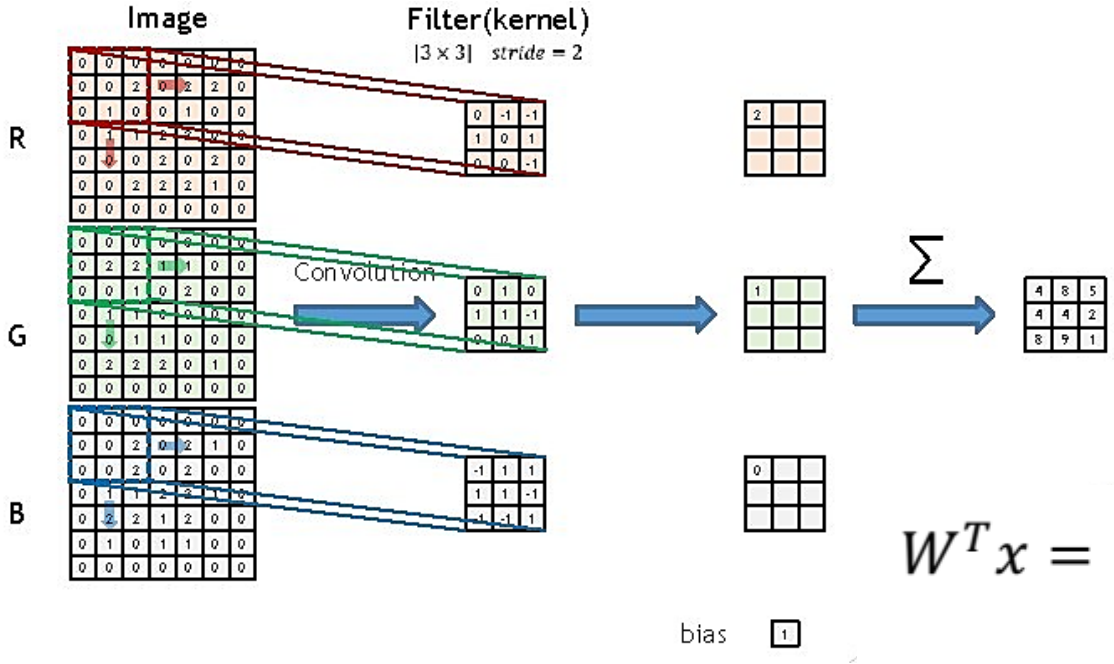


Figure source: Ming Li's Lecture slides, Waterloo University

Convolution on RGB Images

- Make three copies of the filter and apply convolution operation on each channel.
- You will get three values, one for each channel.
- Add all three values of filter mapping to get a single output.



$$W^T x = W^T r + W^T g + W^T b + bias$$

Figure source: Ming Li's Lecture slides, Waterloo University

A Convolution Layer

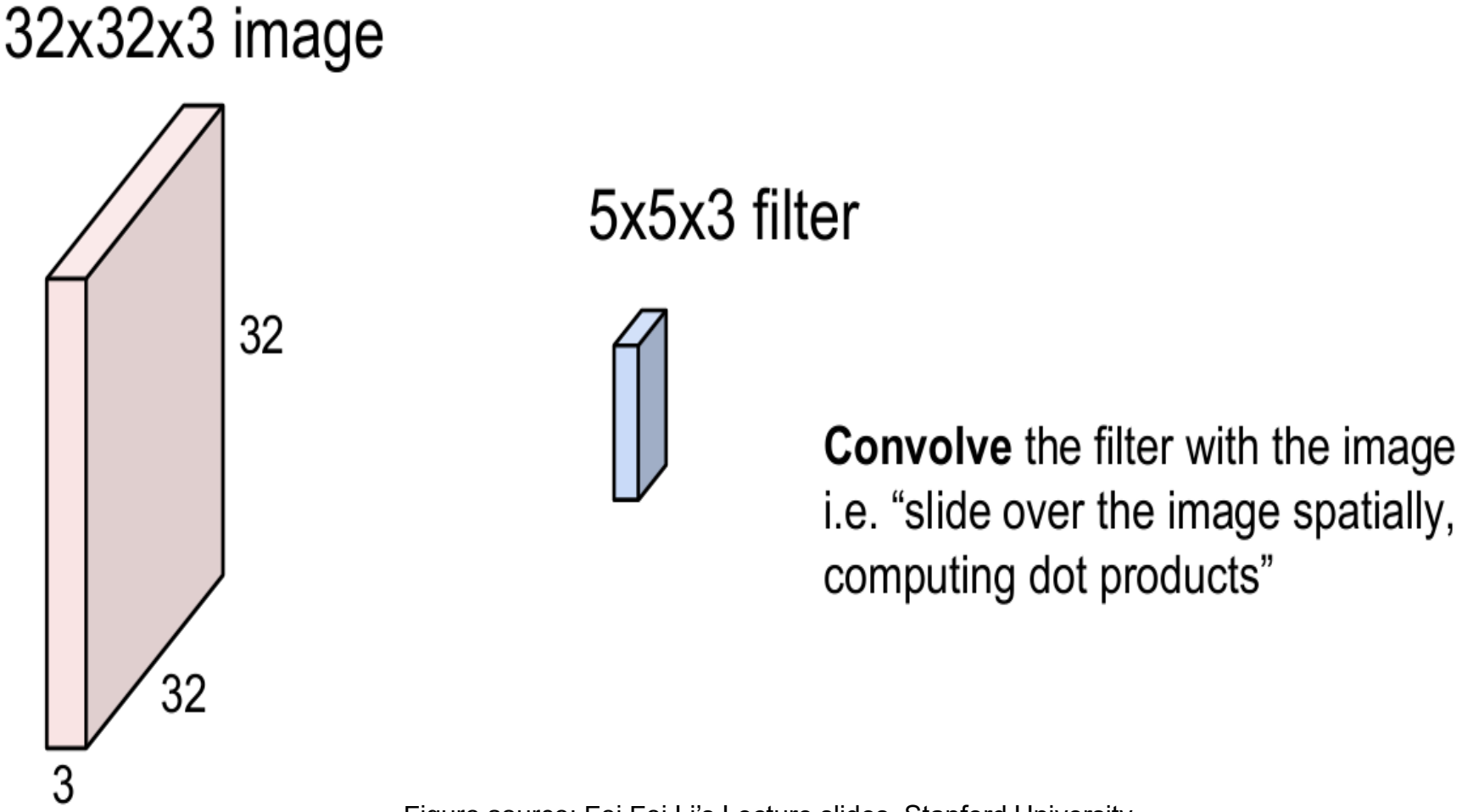


Figure source: Fei Fei Li’s Lecture slides, Stanford University

A Convolution Layer

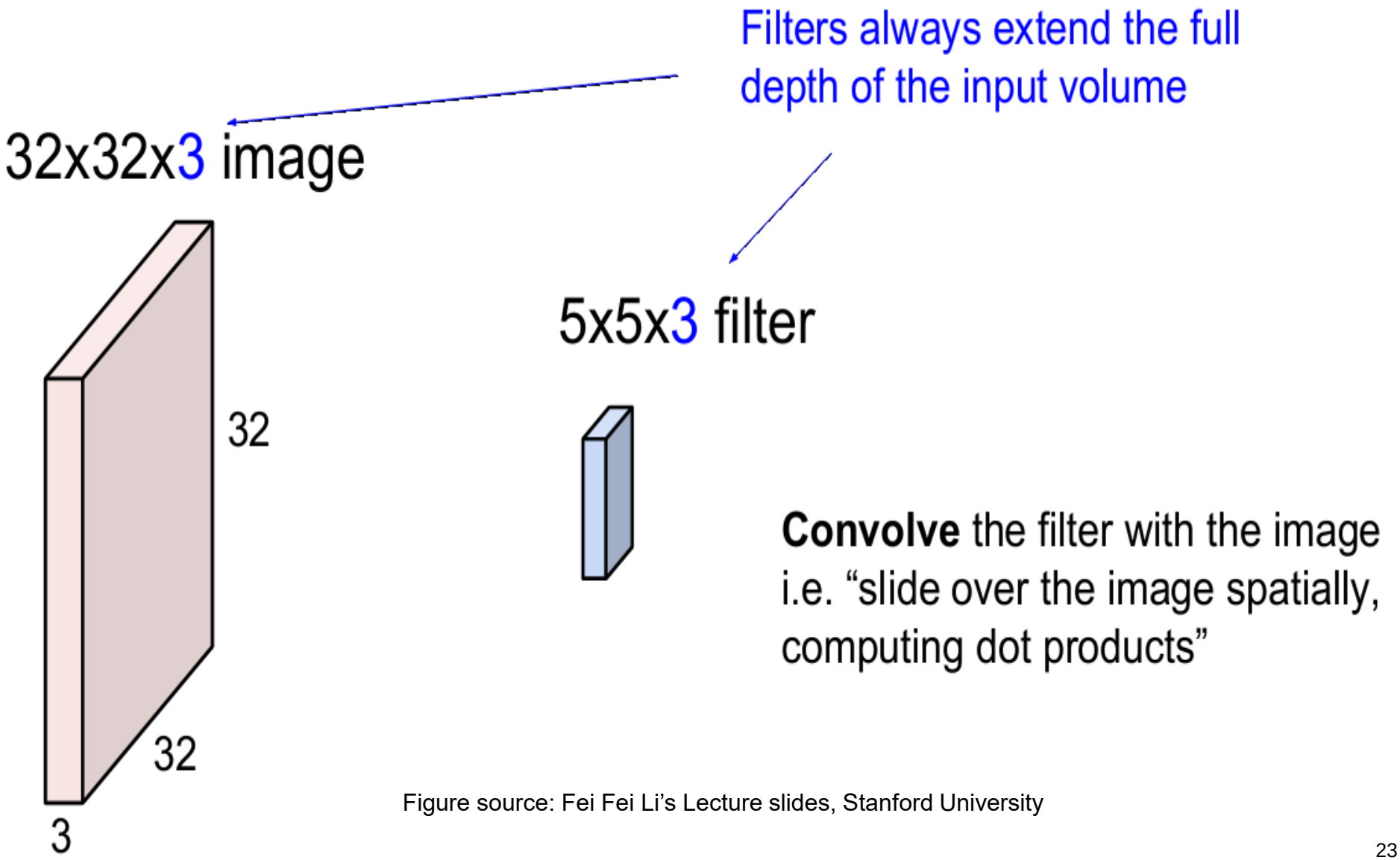
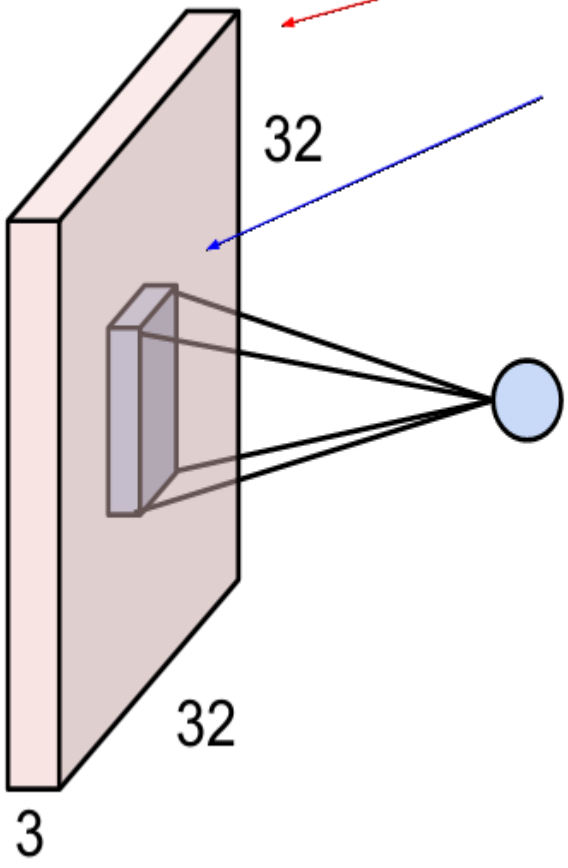


Figure source: Fei Fei Li’s Lecture slides, Stanford University

A Convolution Layer



32x32x3 image

5x5x3 filter w

A hidden neuron in next layer

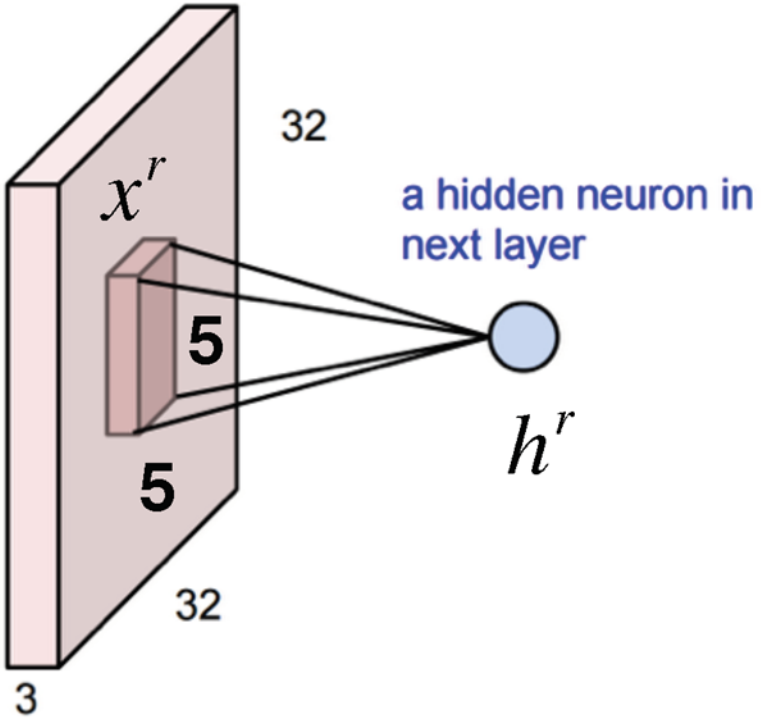
1 number:

the result of taking a dot product between the filter and a small 5x5x3 chunk of the image (i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

Figure source: Fei Fei Li's Lecture slides, Stanford University

A Convolution Layer
3D Activations



Example: consider the region of the input "x^r"

With output neuron h^r

Then the output is:

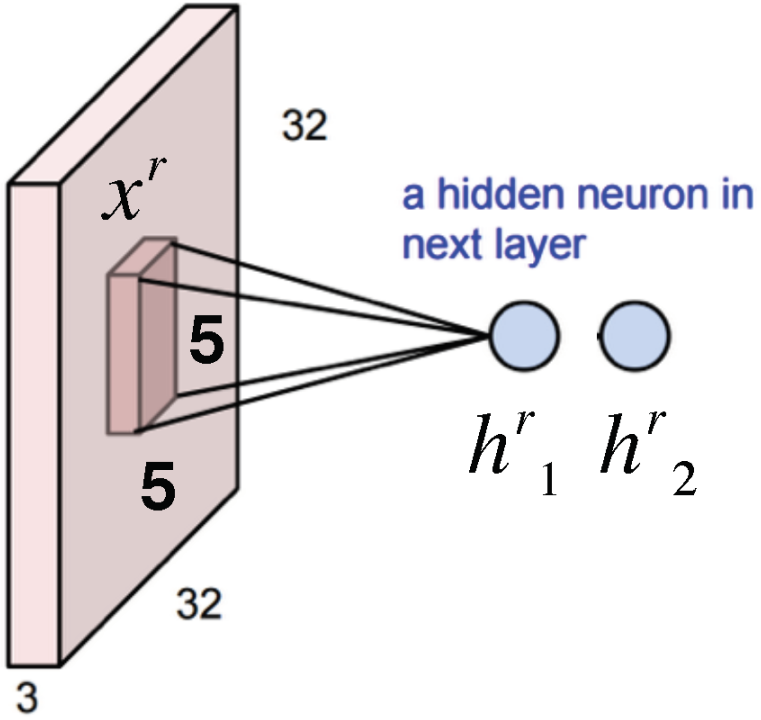
$$h^r = \sum_{ijk} x^r{}_{ijk} W_{ijk} + b$$

Sum over 3 axes

Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

A Convolution Layer
3D Activations



With **2** output neurons

$$h^r{}_1 = \sum_{ijk} x^r{}_{ijk} W_{1ijk} + b_1$$

$$h^r{}_2 = \sum_{ijk} x^r{}_{ijk} W_{2ijk} + b_2$$

Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

A Convolution Layer
3D Activations

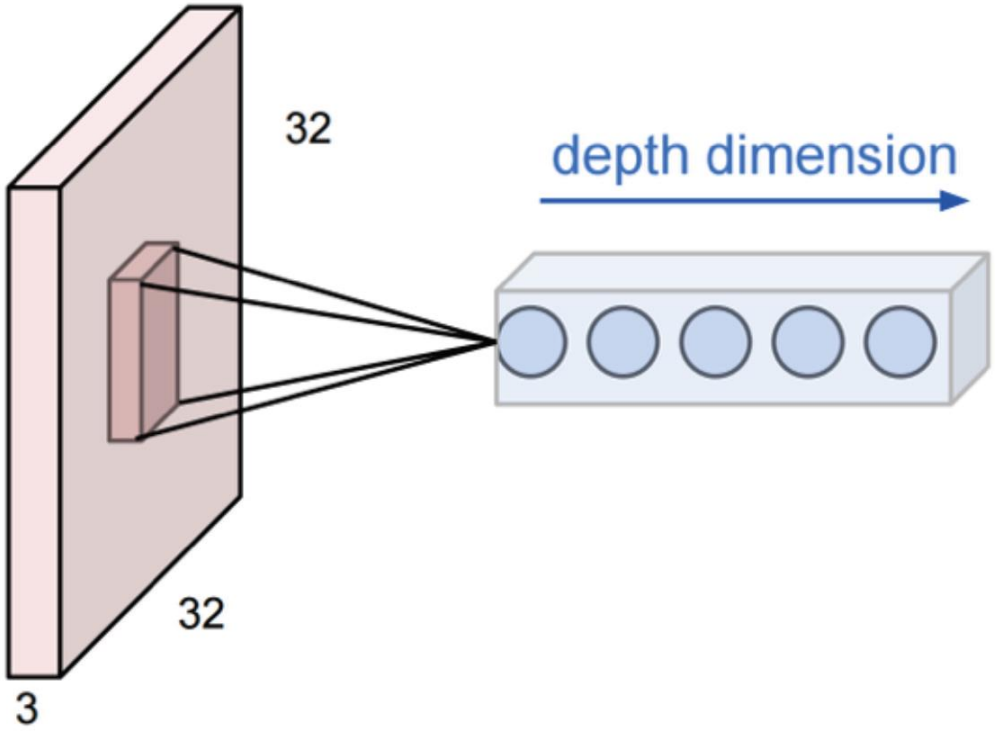
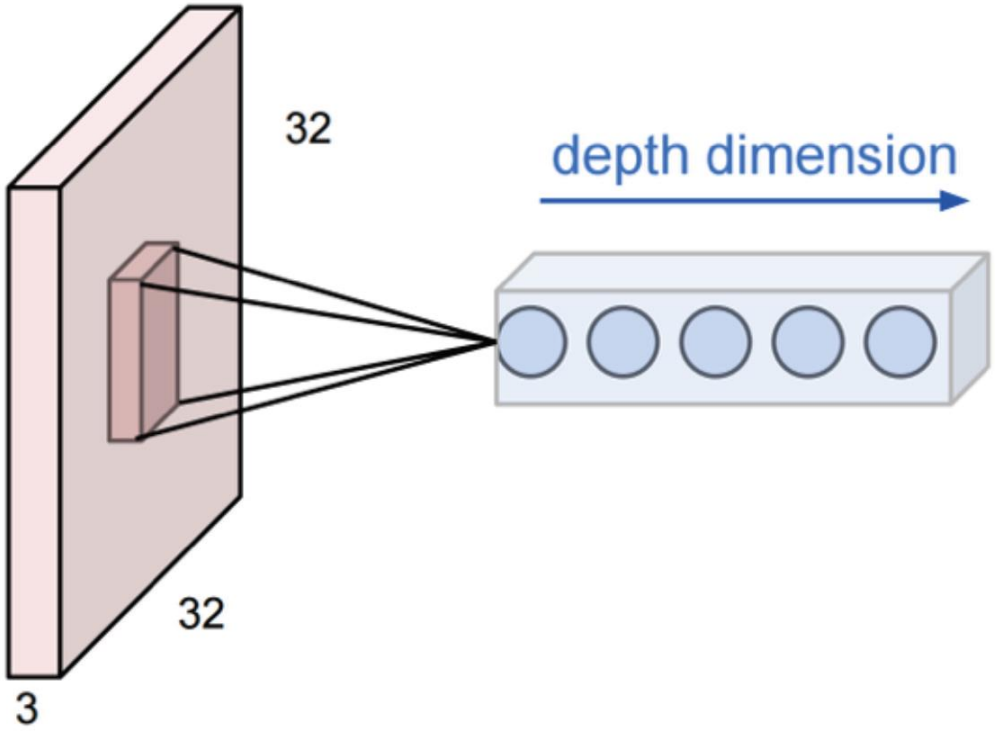


Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

A Convolution Layer
3D Activations



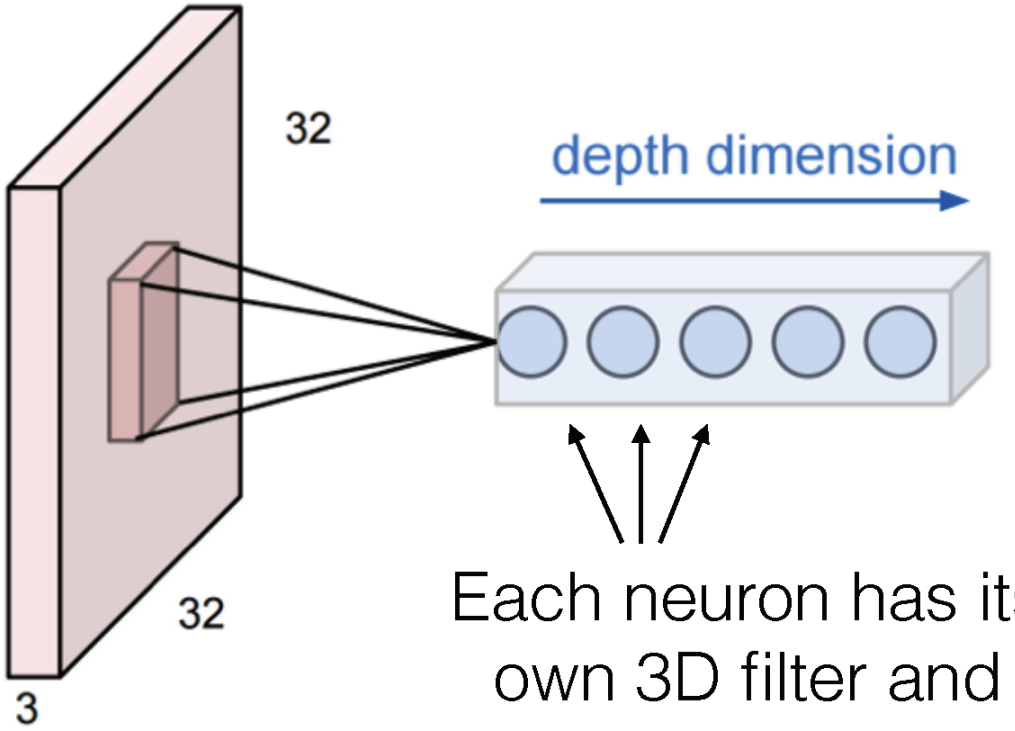
We can keep adding more outputs

These form a column in the output volume: [depth x 1 x 1]

Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

A Convolution Layer
3D Activations



Each neuron has its own 3D filter and own (scalar) bias

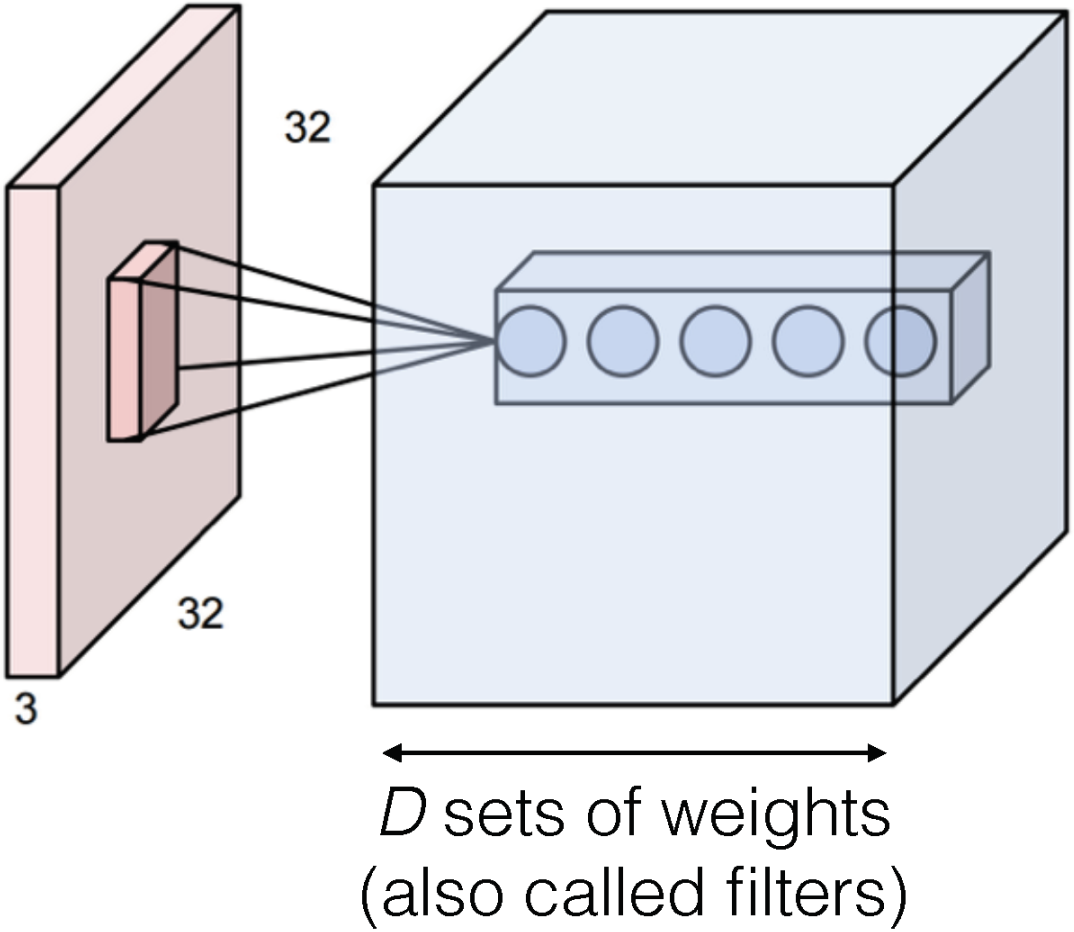
We can keep adding more outputs

These form a column in the output volume: [depth x 1 x 1]

Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

A Convolution Layer
3D Activations

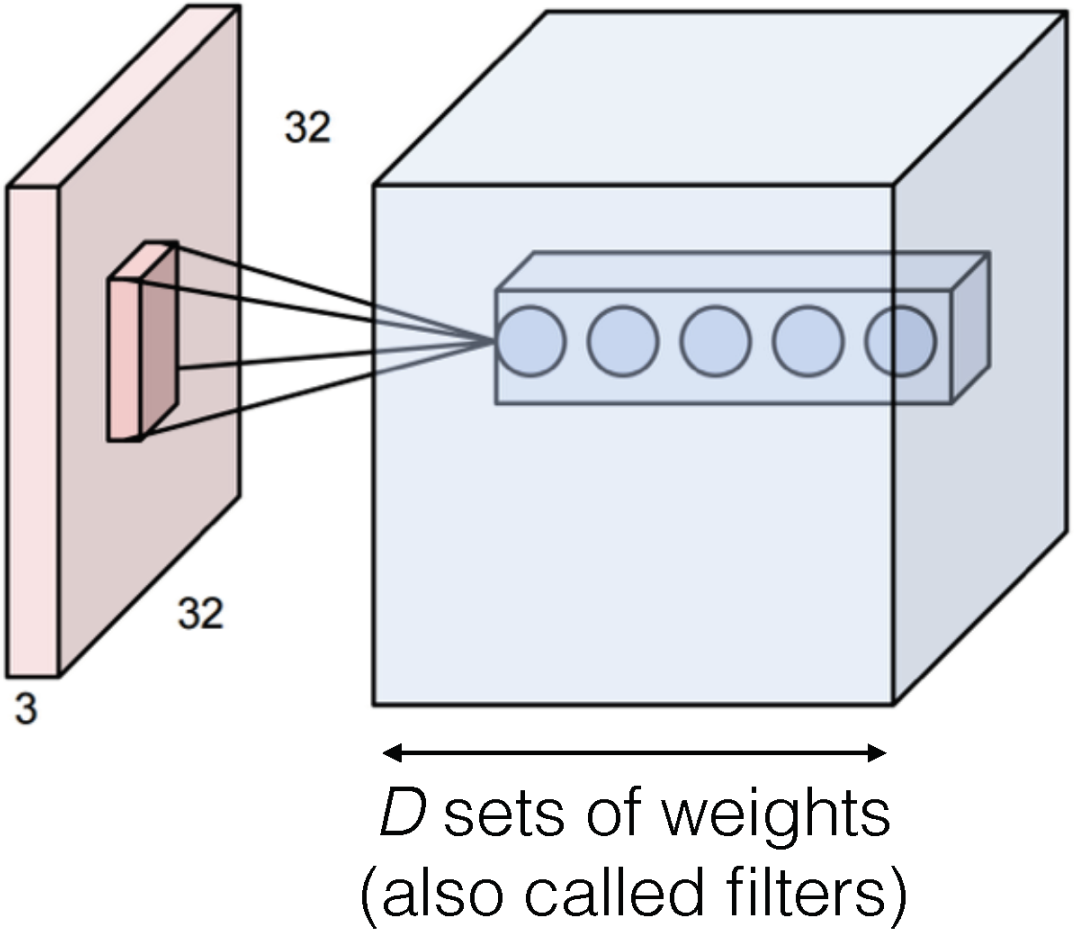


Now repeat this across the input

Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

A Convolution Layer
3D Activations



Now repeat this across the input

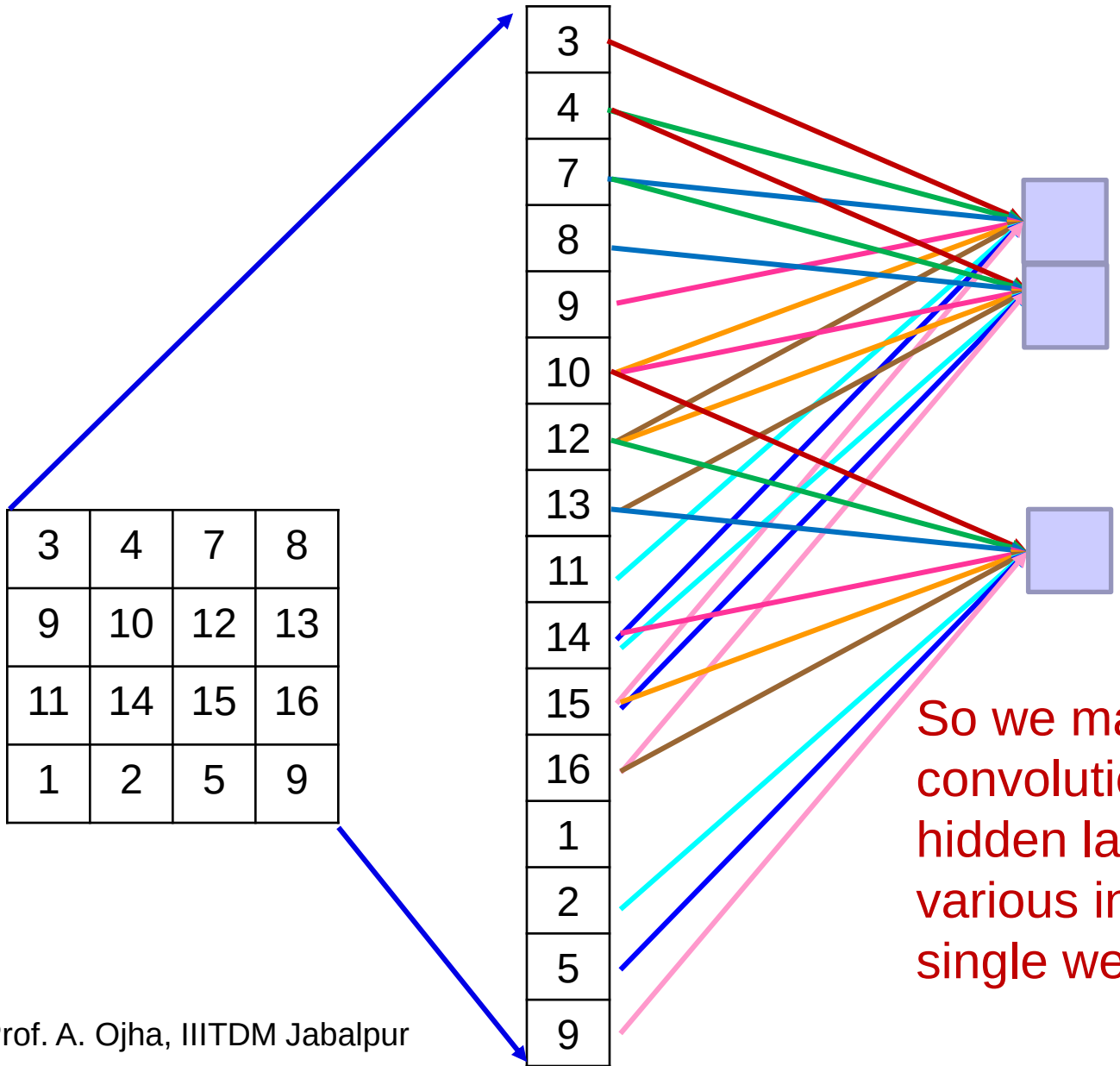
Weight sharing:

Each filter shares the same weights (but each depth index has its own set of weights)

Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

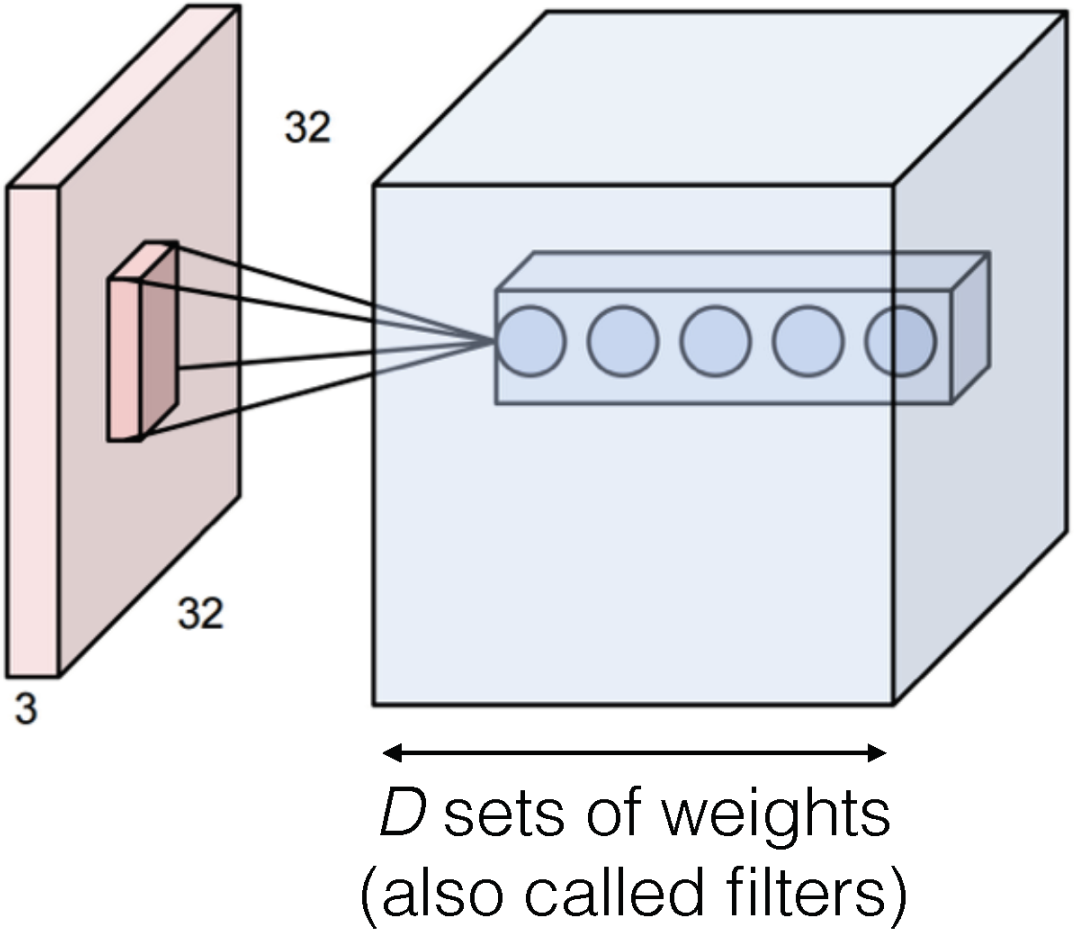
Weight Sharing



So we may think of convolution layer as a hidden layer in which various inputs share a single weight parameter.

Source: Prof. A. Ojha, IIITDM Jabalpur

A Convolution Layer
3D Activations

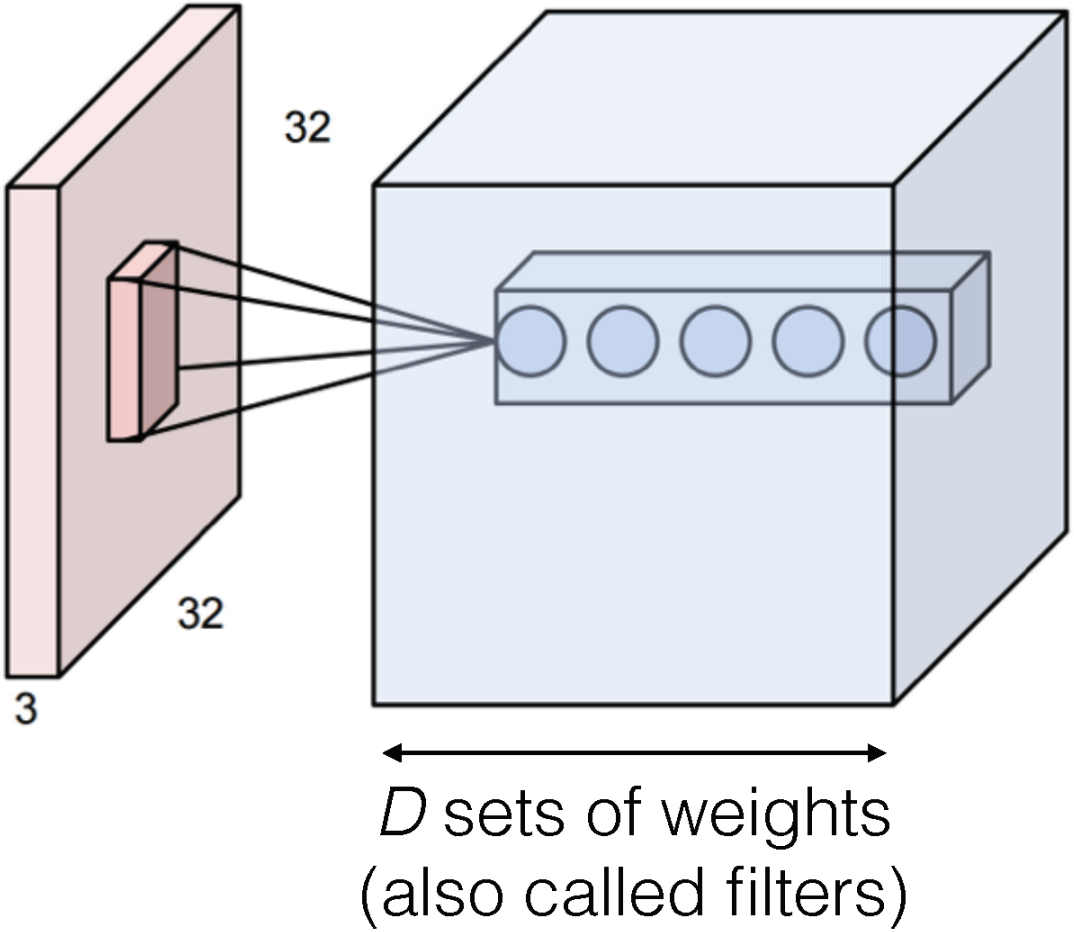


With weight sharing, this is called **convolution**

Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

A Convolution Layer
3D Activations



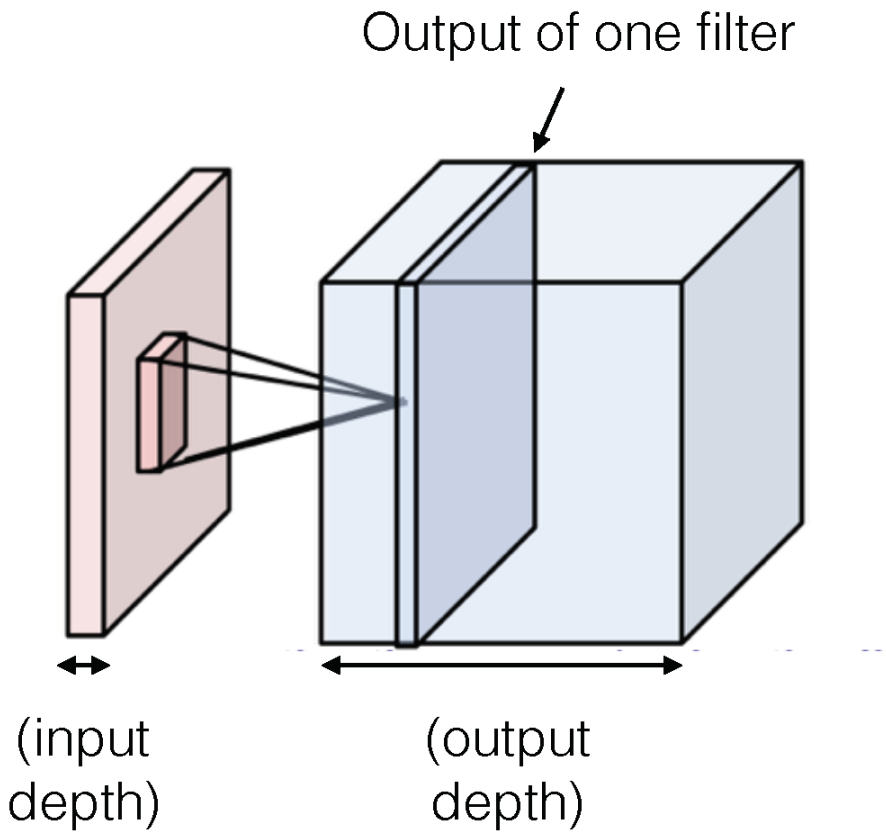
With weight sharing, this is called **convolution**

Without weight sharing, this is called a **locally connected layer**

Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

A Convolution Layer
3D Activations



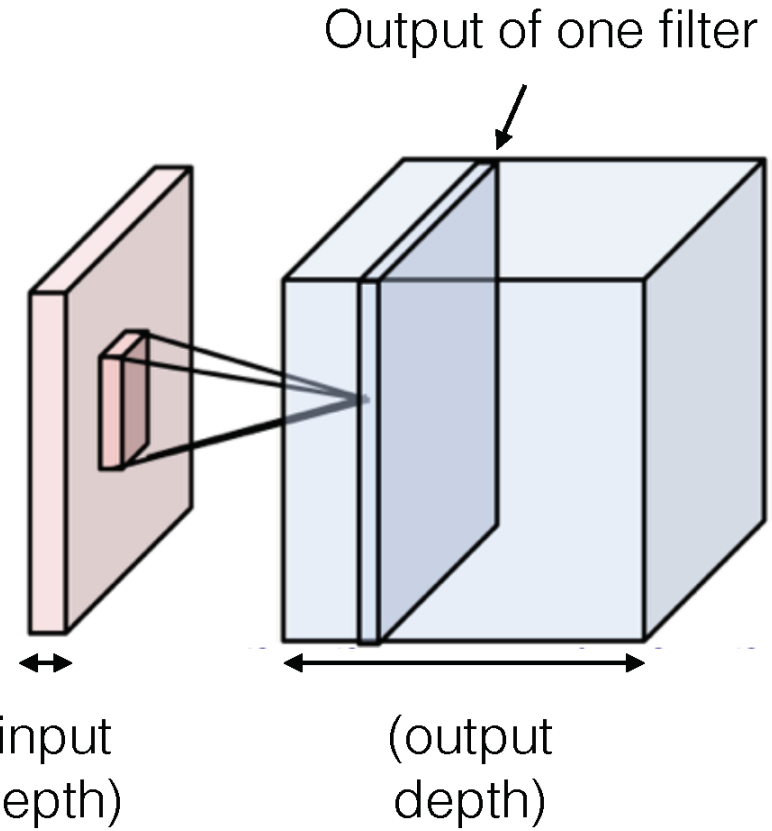
One set of weights gives one slice in the output

To get a 3D output of depth *D*, use *D* different filters

In practice, ConvNets use many filters (~64 to 1024)

Source: Prof. Ioannis Gkioulekas, CMU

A Convolution Layer
3D Activations



One set of weights gives one slice in the output

To get a 3D output of depth *D*, use *D* different filters

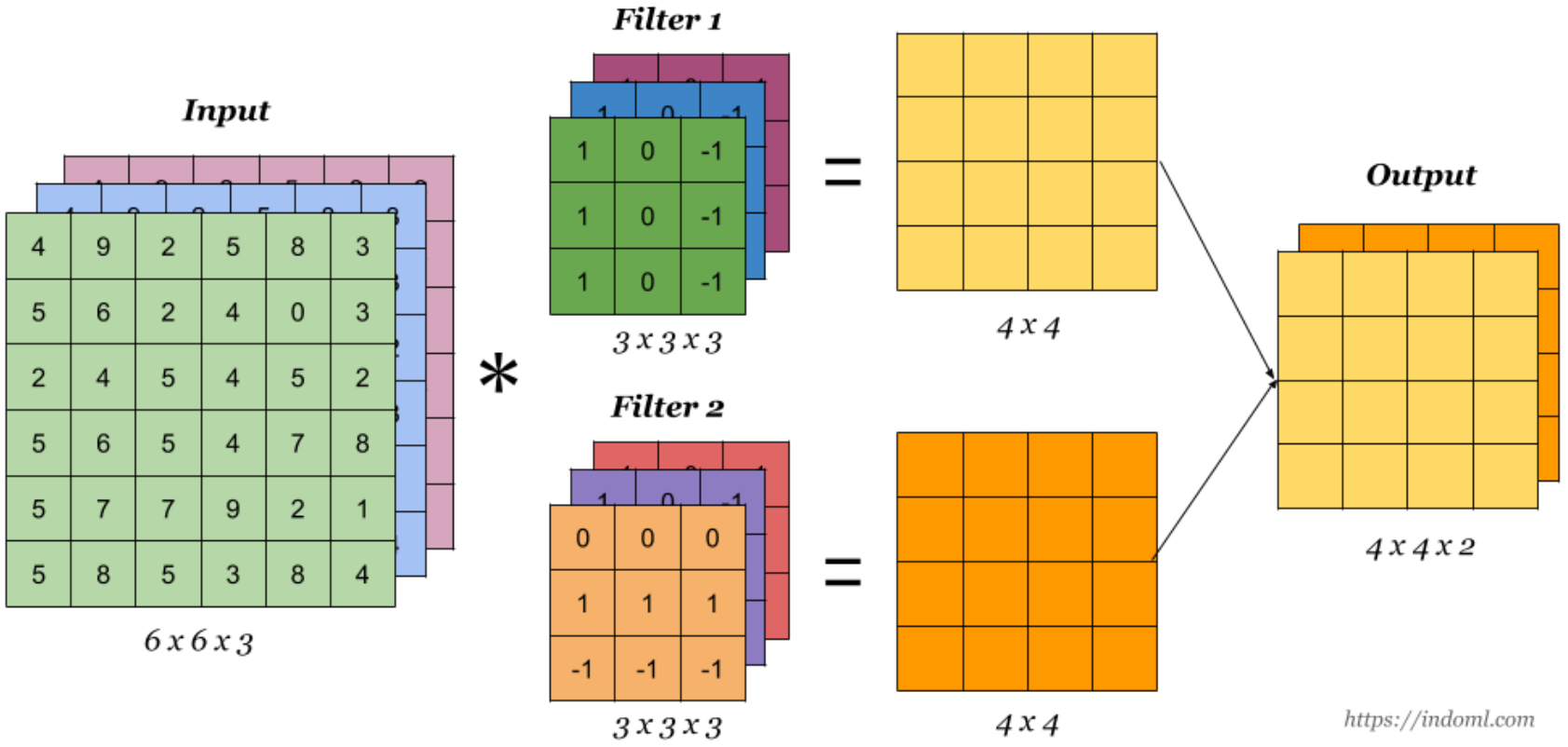
In practice, ConvNets use many filters (~64 to 1024)

All together, the weights are **4** dimensional:
(output depth, input depth, kernel height, kernel width)

Source: Prof. Ioannis Gkioulekas, CMU

Feature Maps

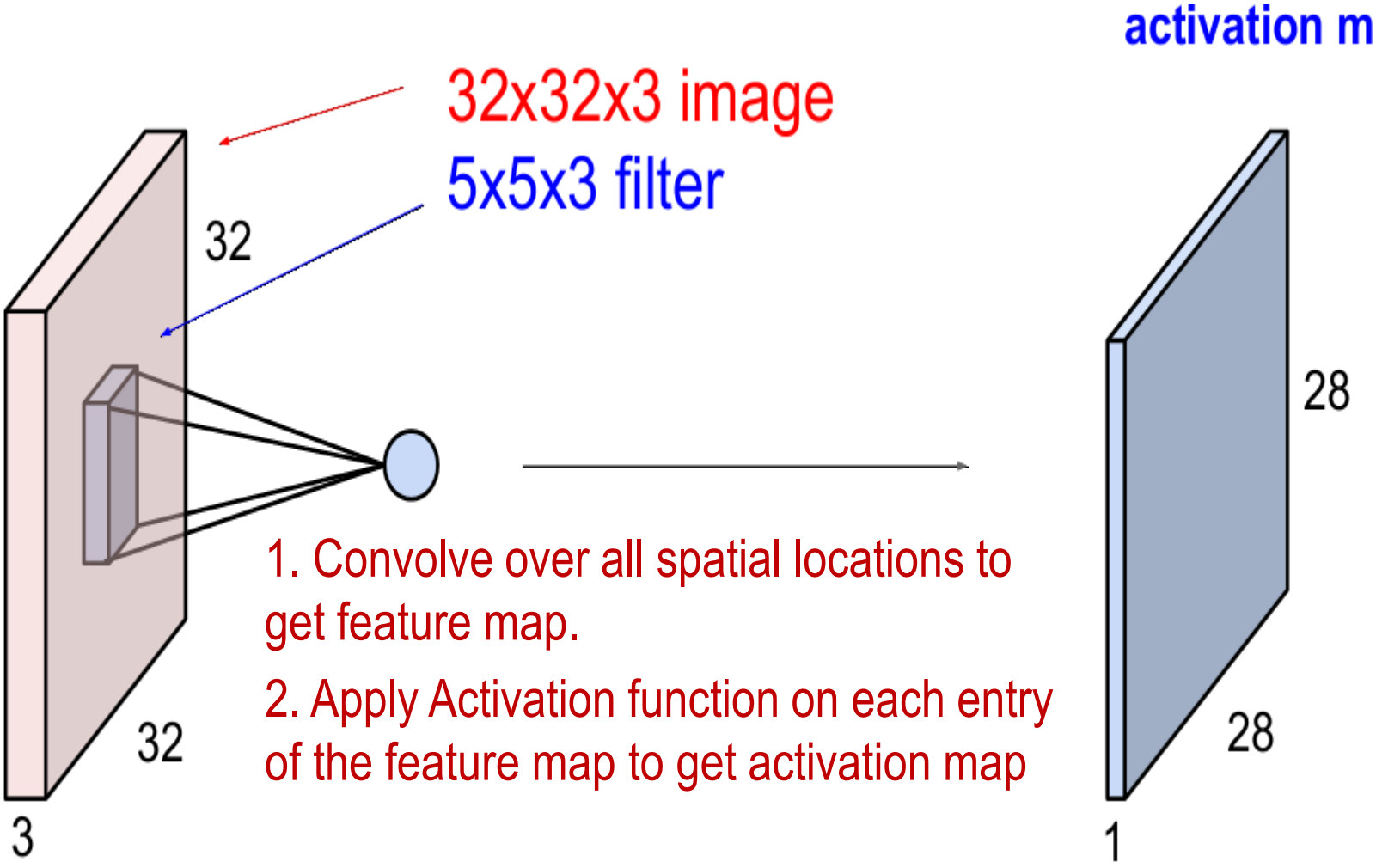
- With 2 filters, output will contain 2 channels, known as feature maps, one for each filter.
- Each feature map applies its own bias.



Activation Maps

- An activation function is applied on all the entries of each feature map to get the activation maps.

Filters as activation maps



One set of weights gives one slice in the output.

Figure source: Fei Fei Li's Lecture slides, Stanford University

Filters as activation maps

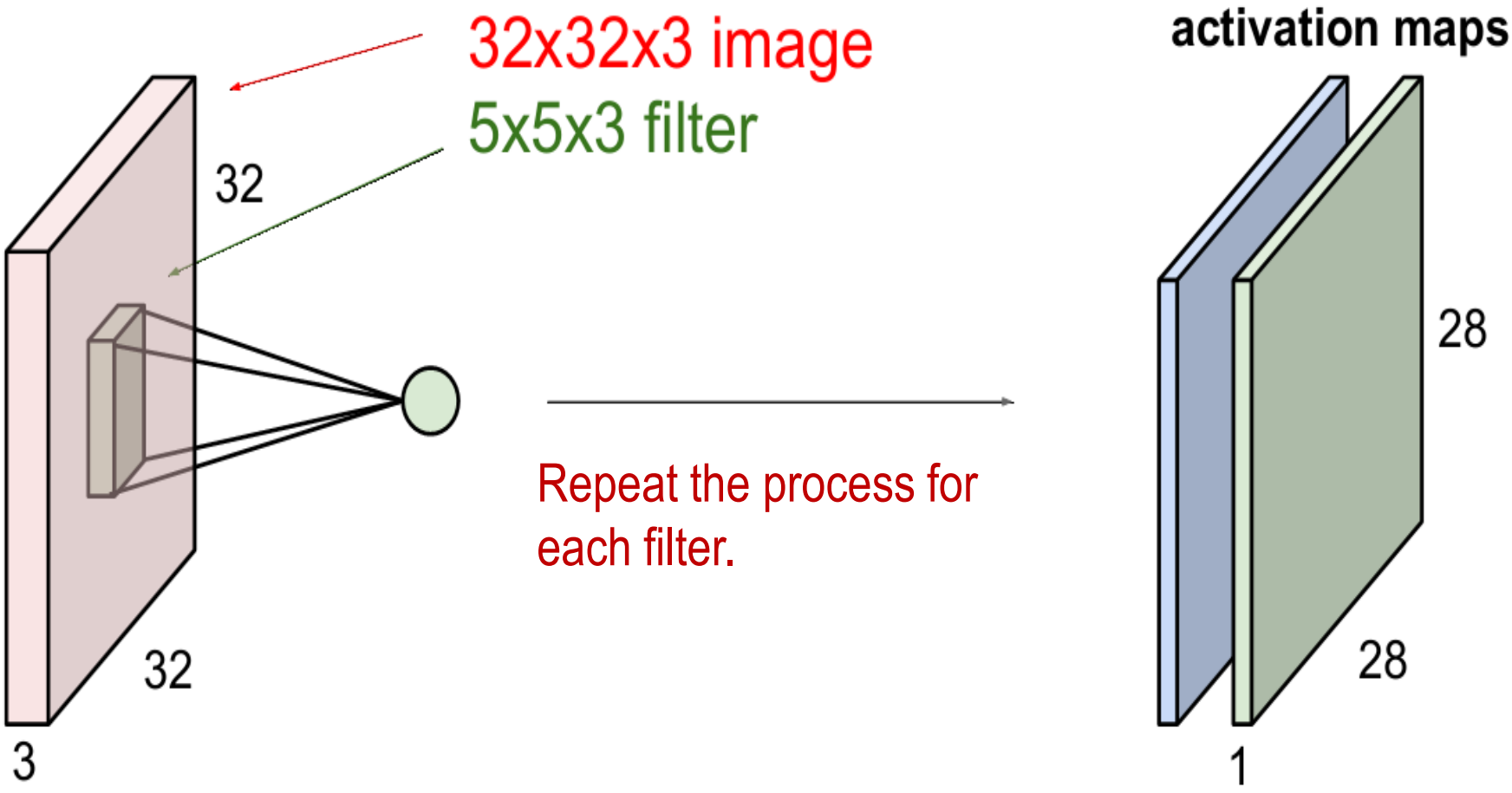
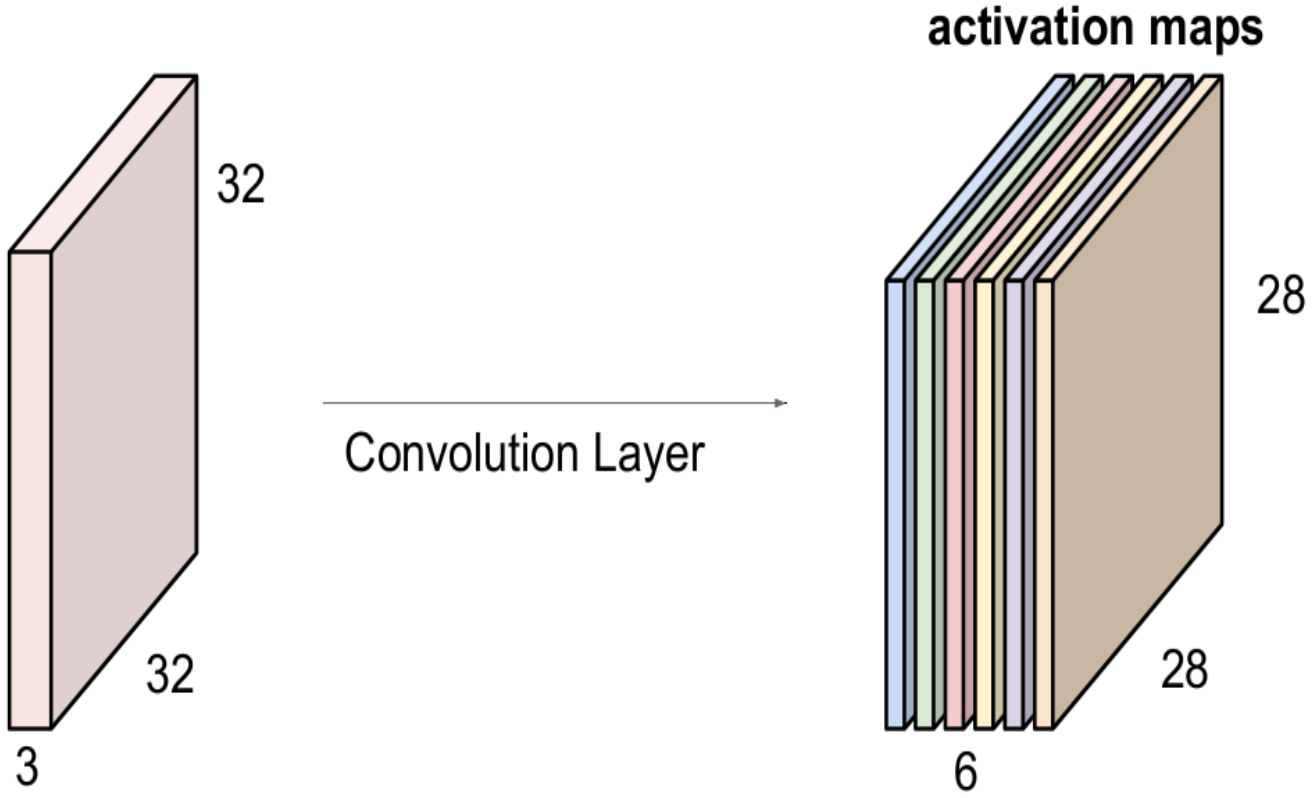


Figure source: Fei Fei Li's Lecture slides, Stanford University

Filters as activation maps

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

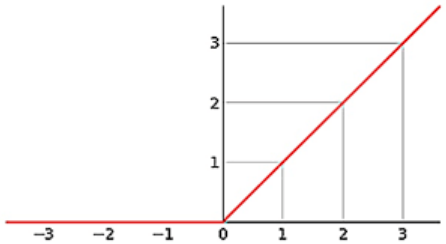


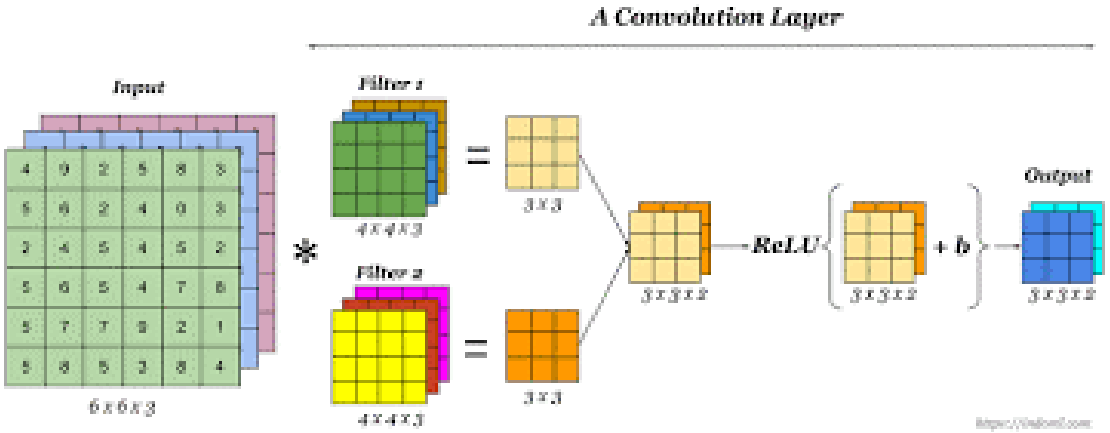
We stack these up to get a "new image" of size 28x28x6!

Figure source: Fei Fei Li's Lecture slides, Stanford University

Activation Maps

- An activation function is applied on all the entries of each feature map to get the activation maps.

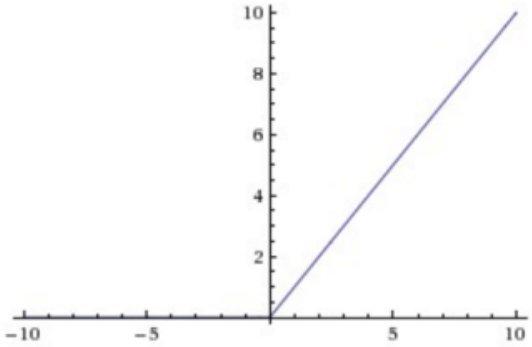


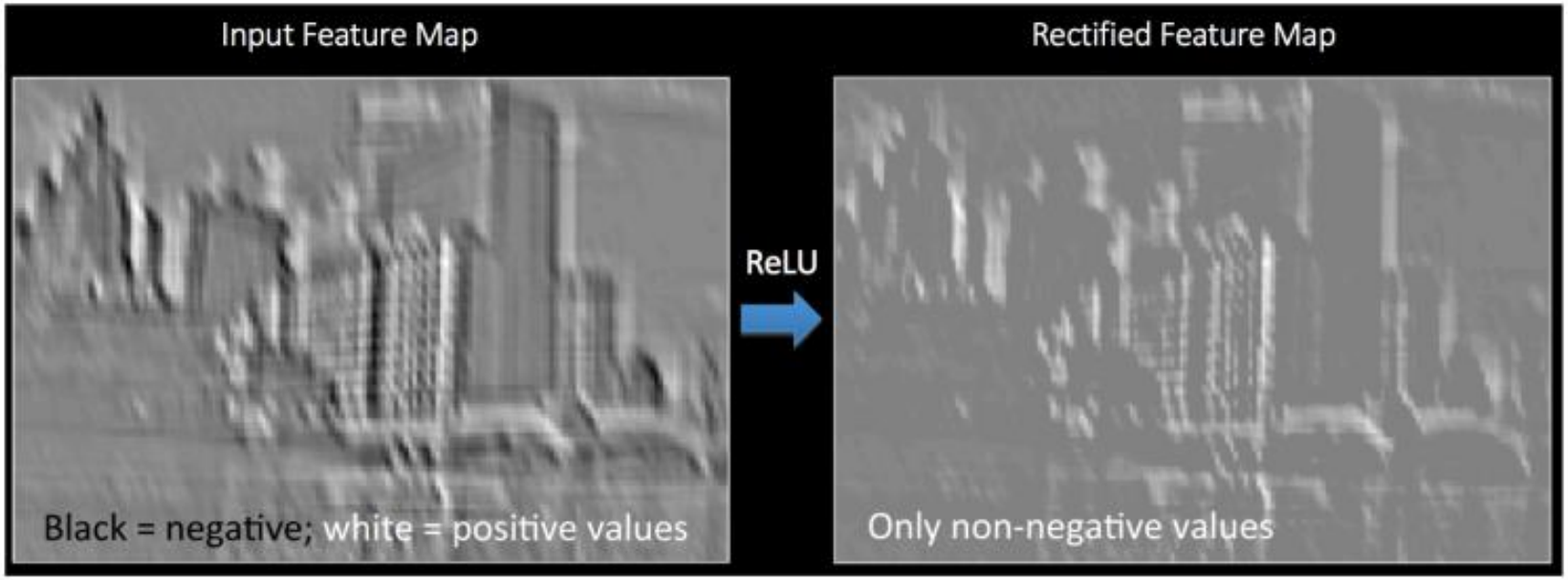


$$a = f(Z) = f(W^T X + b)$$

Activation Maps - Example

Output = Max(zero, Input)

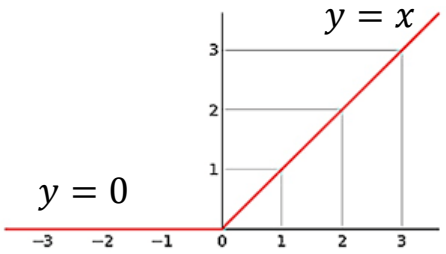




https://ujwlkarn.files.wordpress.com/2016/08/screen-shot-2016-08-07-at-6-18-19-pm.png?w=1024

Filters and Activation Maps

- A CNN is a sequence of convolutional layers, interspersed with activation functions (and possibly other layer types)



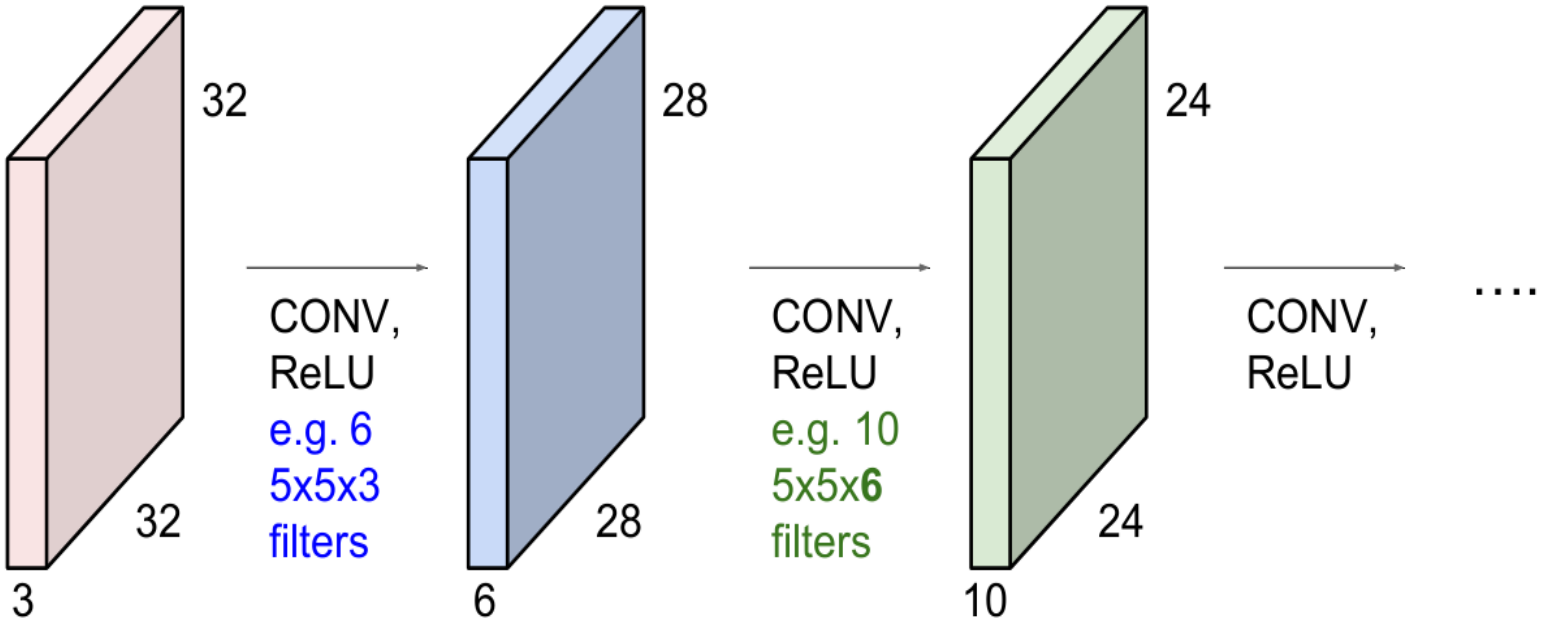


Figure source: Fei Fei Li's Lecture slides, Stanford University

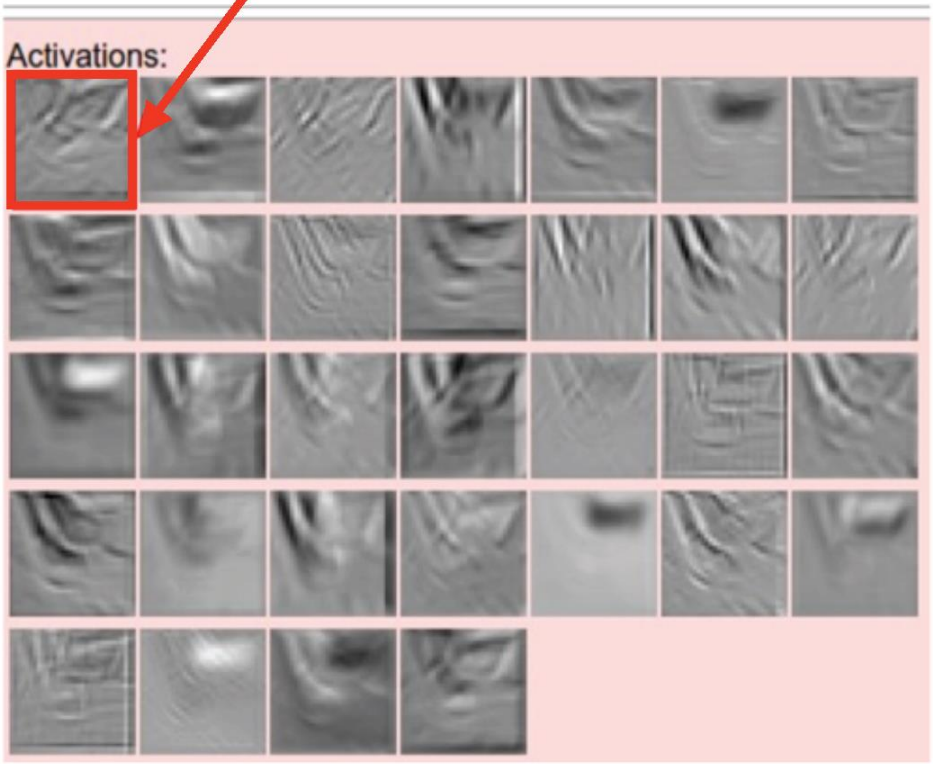
Filters and Activation Maps

We can unravel the 3D cube and show each layer separately:

(Input)

one filter = one depth slice (or activation map)

(32 filters, each 3x5x5)



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

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

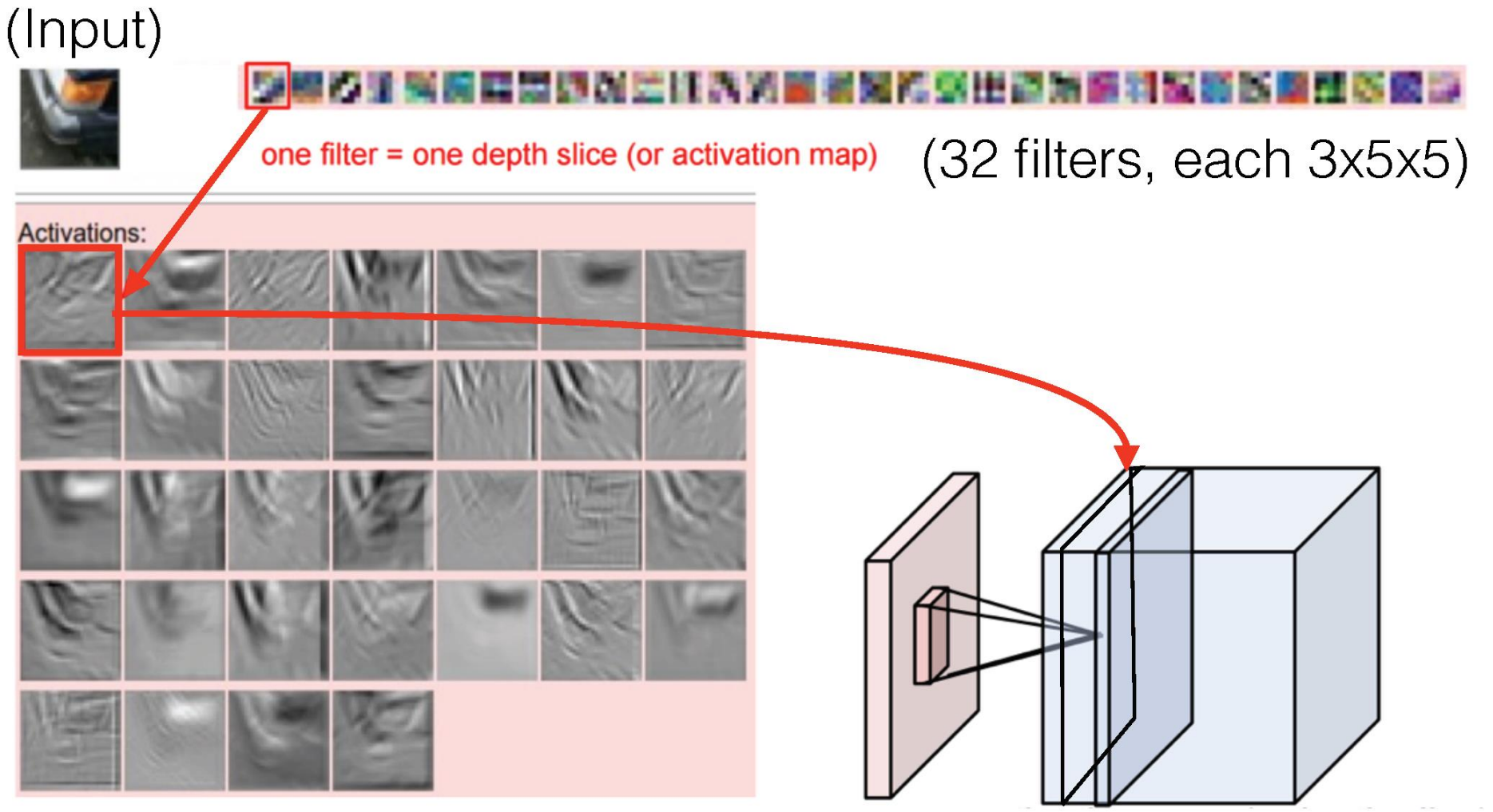
elementwise multiplication and sum of a filter and the signal (image)

source: Fei Fei Li's Lecture slides, Stanford University

Figure: Andrej Karpathy

Filters and Activation Maps

We can unravel the 3D cube and show each layer separately:

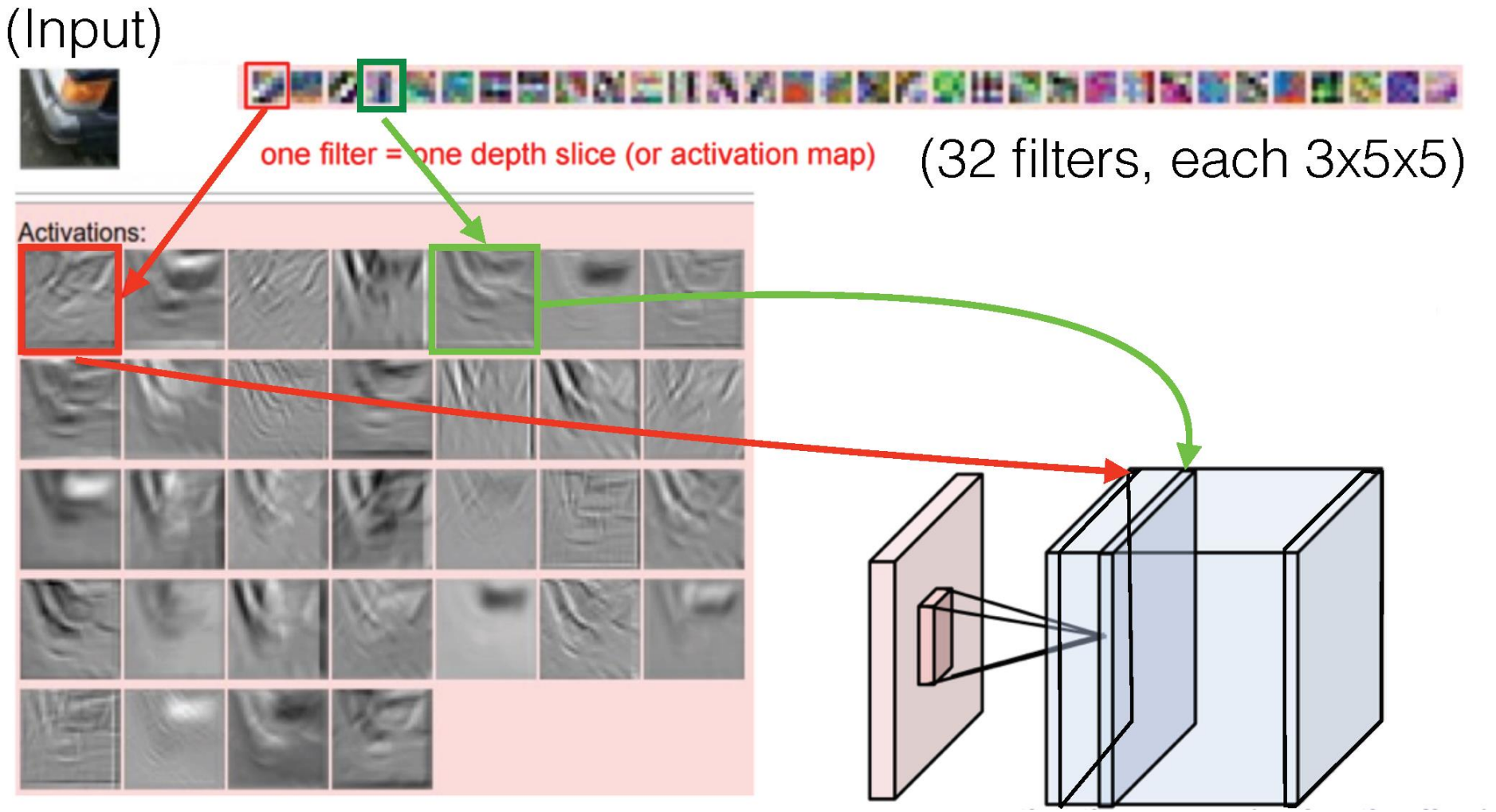


Source: Prof. Ioannis Gkioulekas, CMU

Figure: Andrej Karpathy

Filters and Activation Maps

We can unravel the 3D cube and show each layer separately:



Source: Prof. Ioannis Gkioulekas, CMU

Figure: Andrej Karpathy

Filters and Activation Maps

We can unravel the 3D cube and show each layer separately:

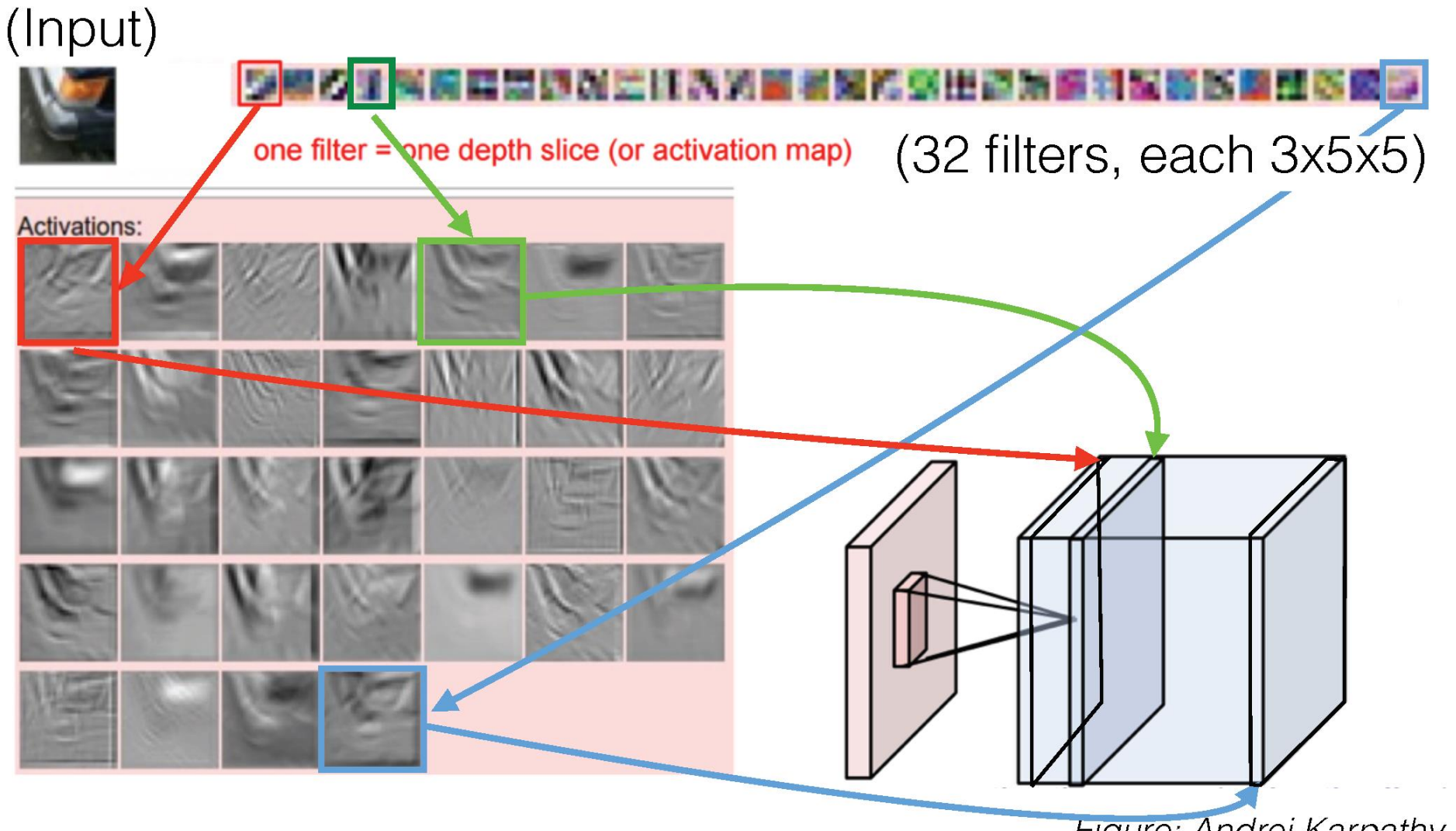
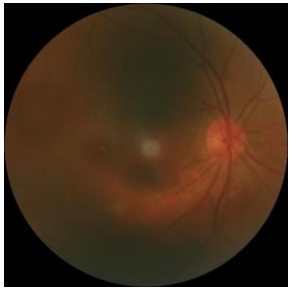


Figure: Andrej Karpathy

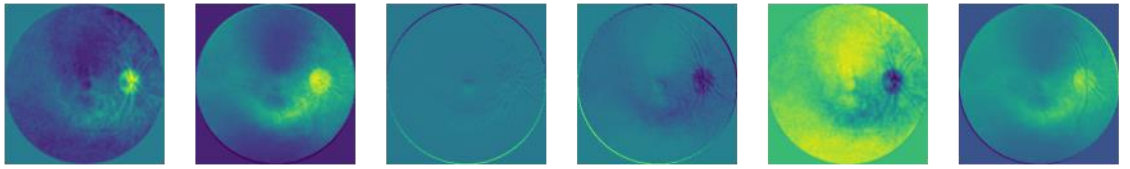
Source: Prof. Ioannis Gkioulekas, CMU

Activation Maps: Example

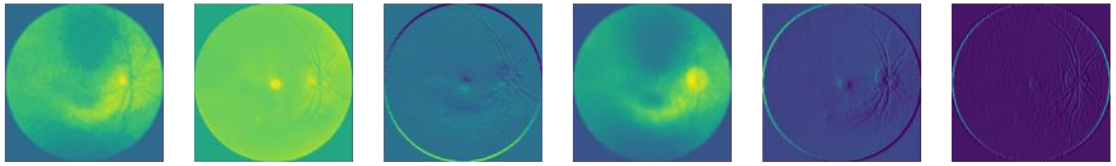
Peking university international competition on ocular disease intelligent recognition (ODIR-2019)



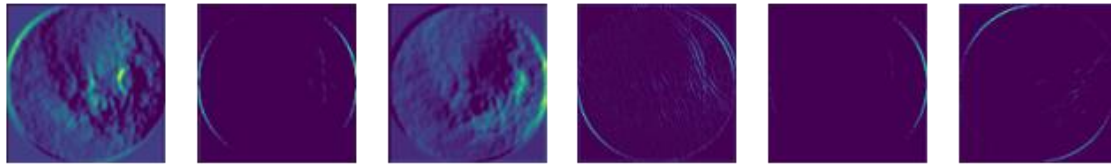
Actual fundus image with cataract in left eye



1st Convolutional Layer Activation Maps

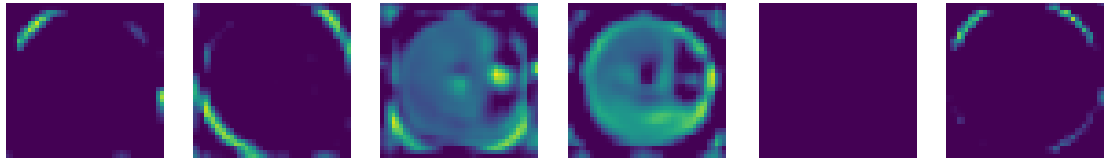


2nd Convolutional Layer Activation Maps



4th Convolutional Layer Activation Maps

7th Convolutional Layer Activation Maps

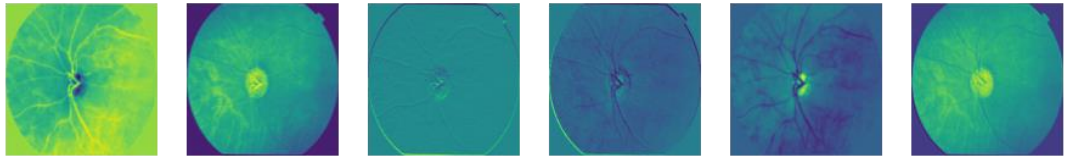


10th Convolutional Layer Activation Maps

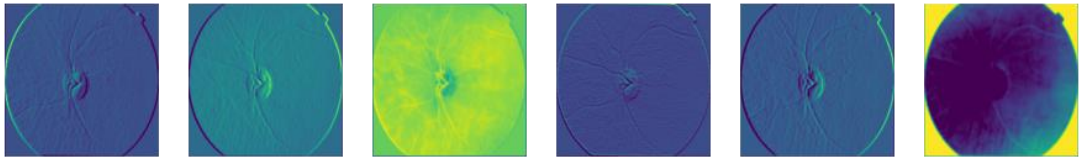
Level of abstraction increases at the high level layers

Activation Maps: Example

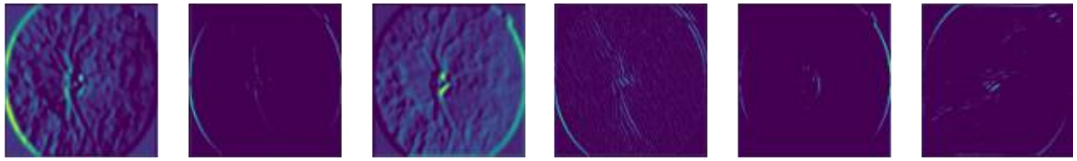
Actual image of glaucoma eye disease from Drishti-GS dataset



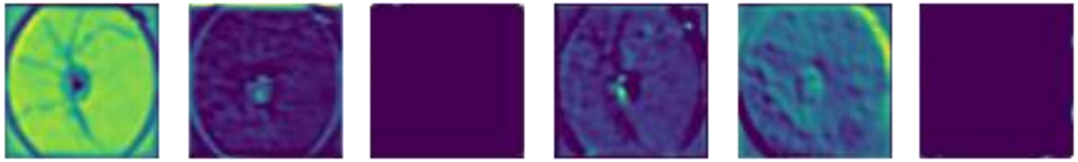
1st Convolutional Layer Activation Maps



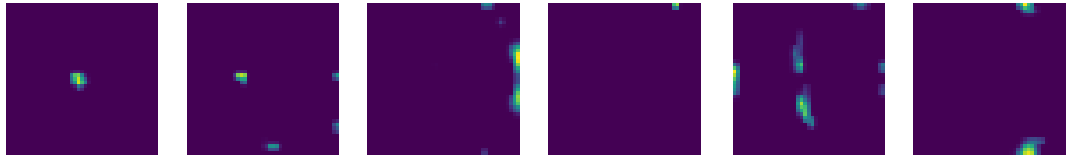
2nd Convolutional Layer Activation Maps



4th Convolutional Layer Activation Maps



7th Convolutional Layer Activation Maps

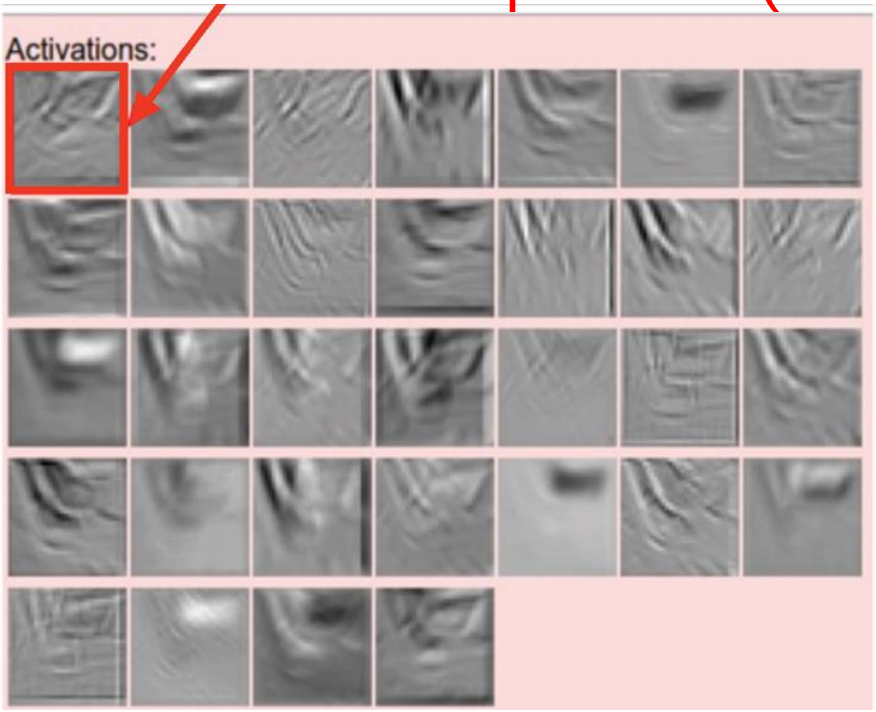


10th Convolutional Layer Activation Maps

Drishti-GS: Retinal Image Dataset for Optic Nerve Head(ONH) Segmentation. Sivaswamy J, Krishnadas K. R, Joshi G. D, Jain Madhulika, Ujjwal and Syed Abbas T., IEEE ISBI, Beijing

Filters and Redundancies

One filter = one depth slice (or activation map)



- Observe that some filters detect almost same patterns.

Source: Prof. Ioannis Gkioulekas, CMU

Convolution Layer

- Number of filters in a single Convolution layer are chosen by the user of the CNN (Depends on application and data input type).
- Typically, a sizable number of filters are chosen to extract the maximum number of features.
- So, what do you learn here ?

Source: Prof. A. Ojha, IIITDM Jabalpur

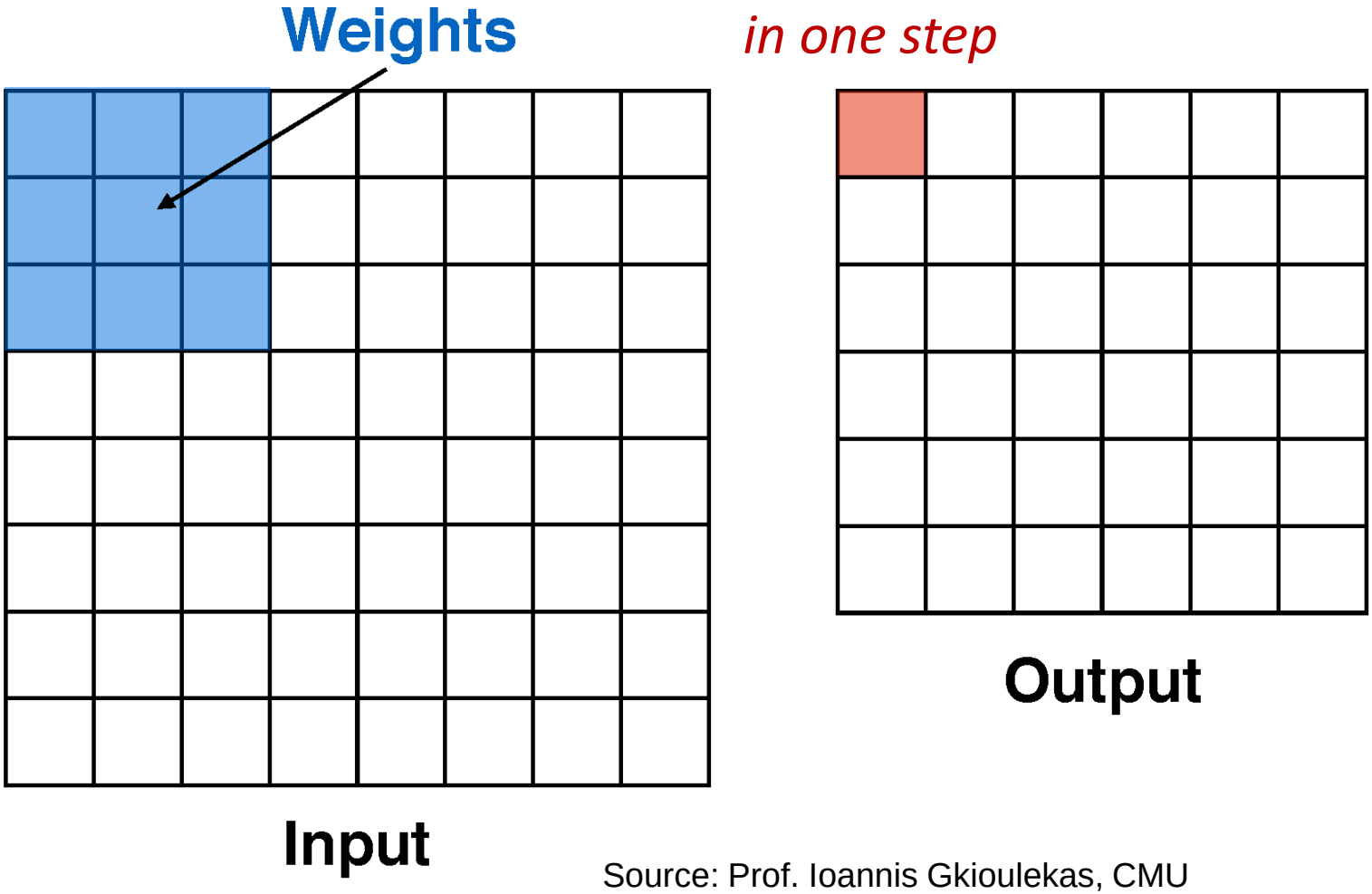
Demos

- http://cs231n.stanford.edu/
- http://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html

Convolution layer and Stride

During convolution, the weights "slide" along the input to generate each output

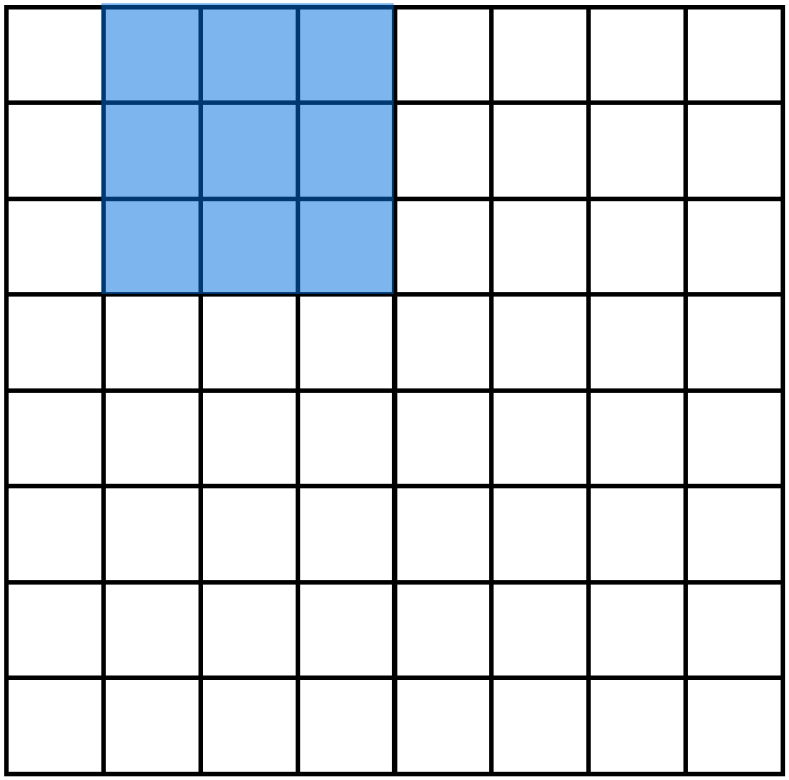
how many cells filter moves next in one step



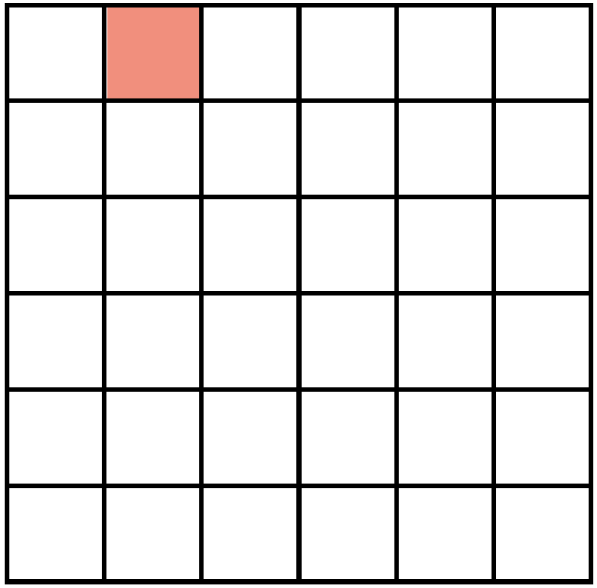
Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

During convolution, the weights “slide” along the input to generate each output



Input

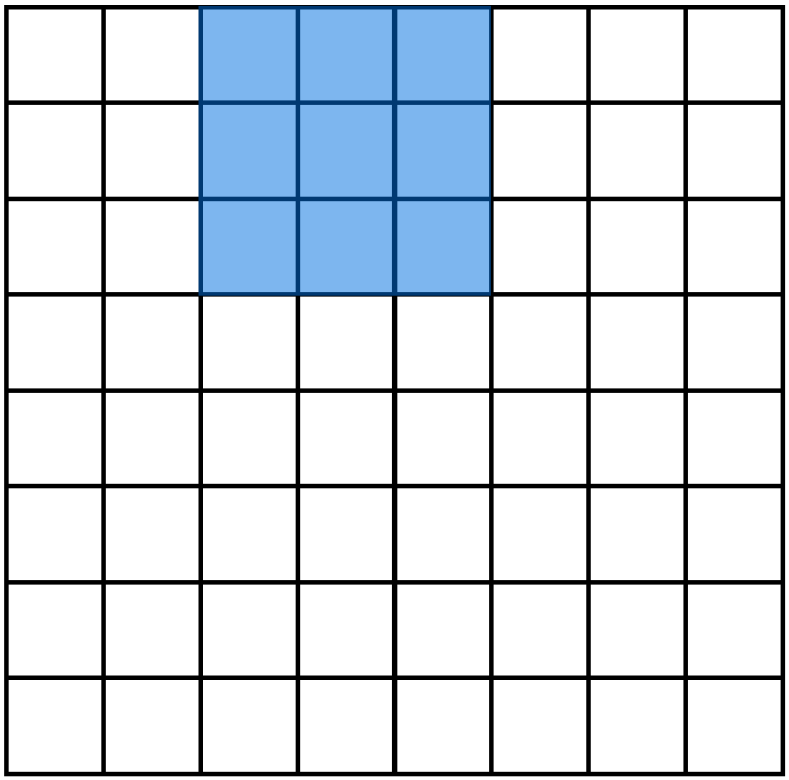


Output

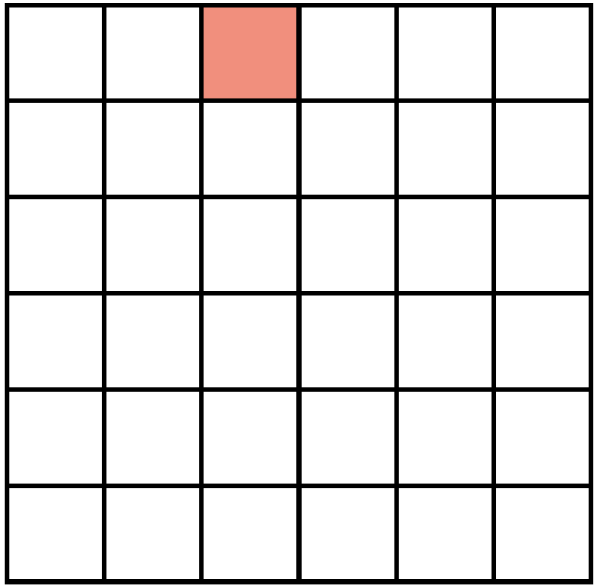
Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

During convolution, the weights “slide” along the input to generate each output



Input

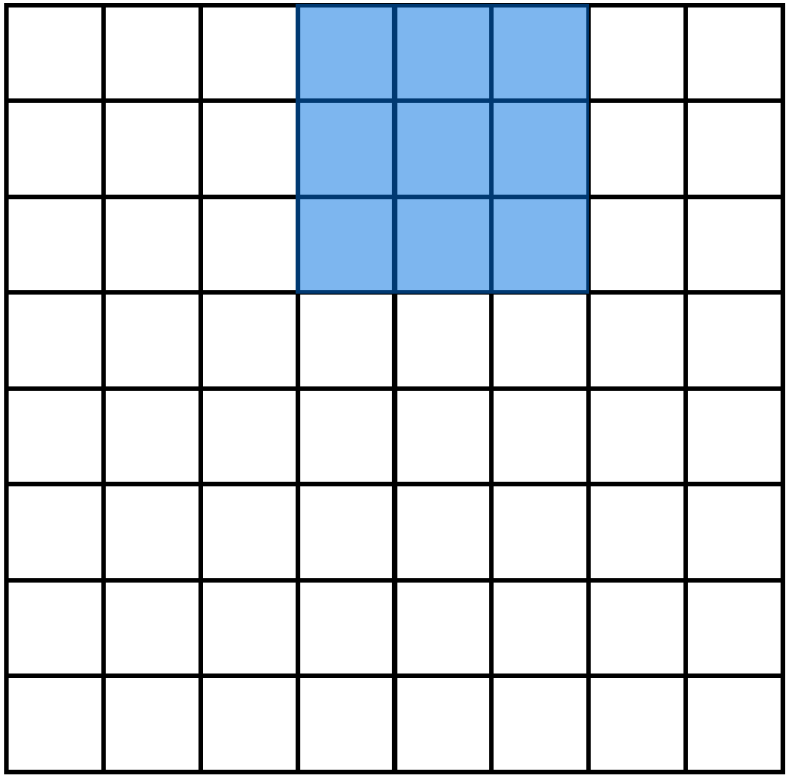


Output

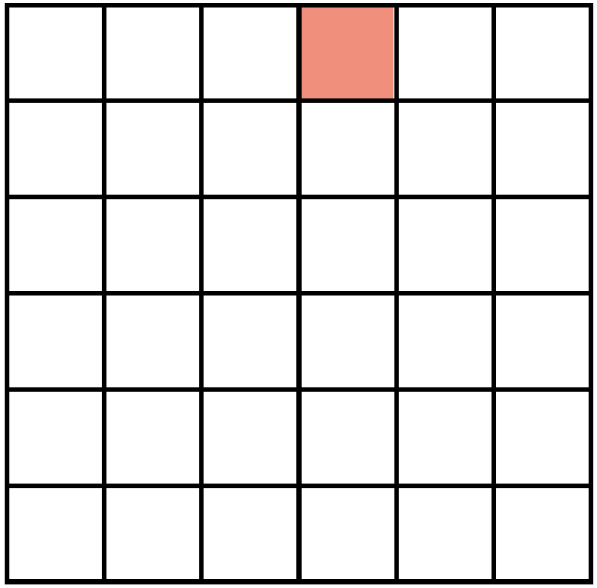
Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

During convolution, the weights “slide” along the input to generate each output



Input

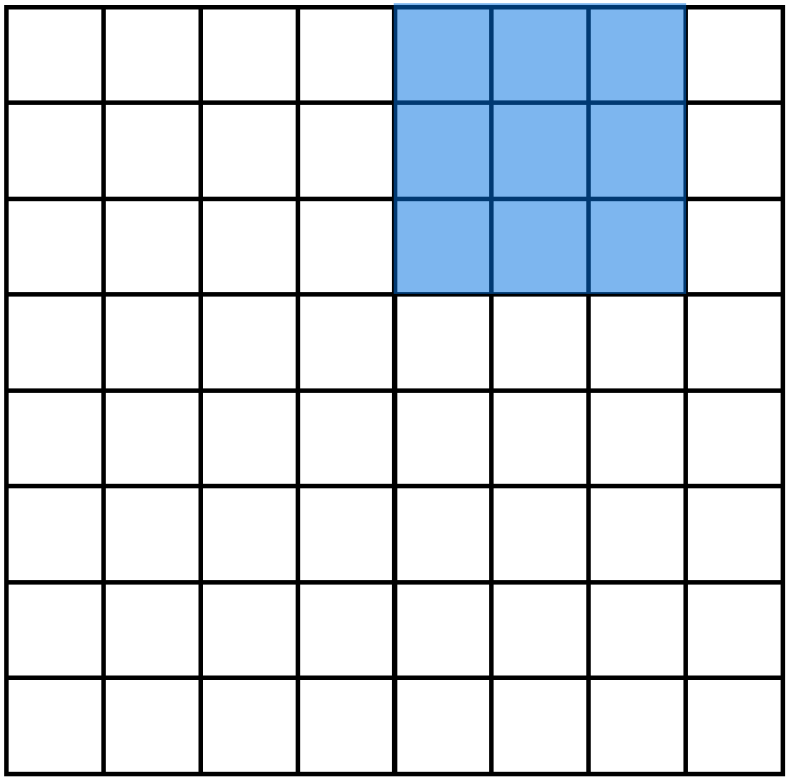


Output

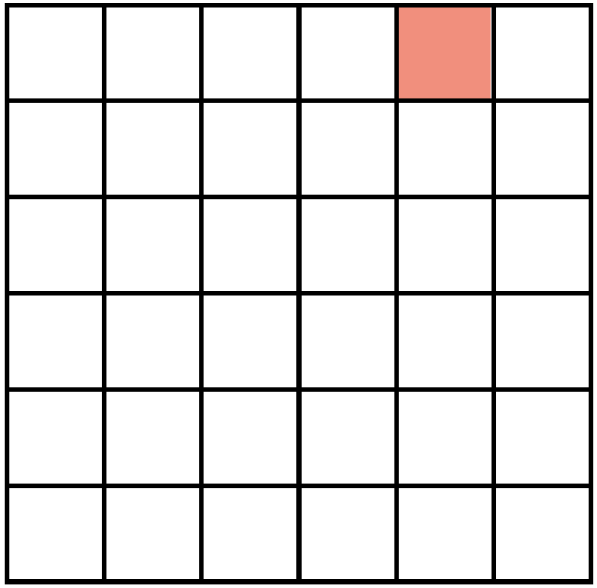
Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

During convolution, the weights “slide” along the input to generate each output



Input

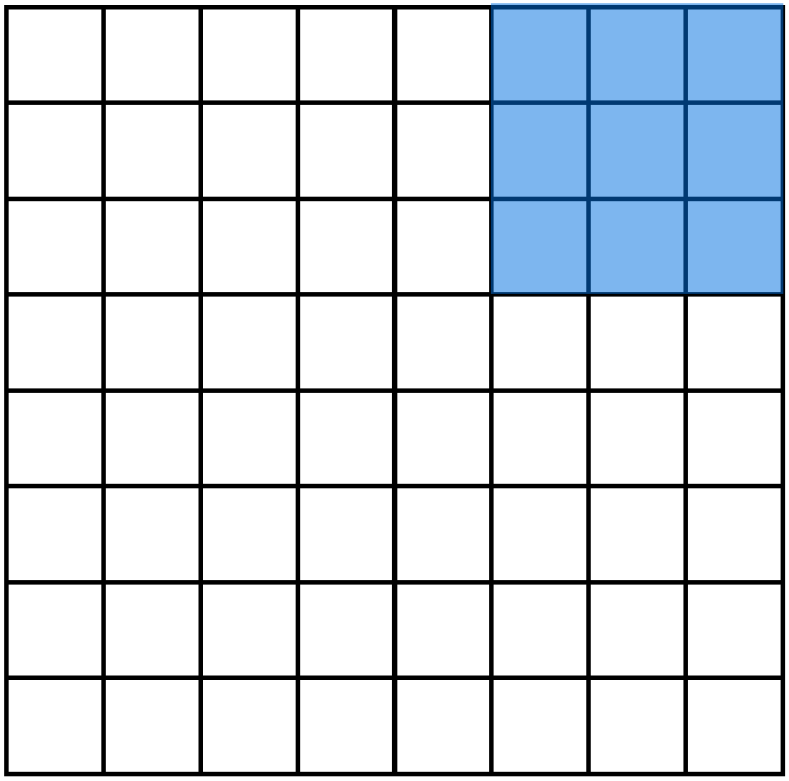


Output

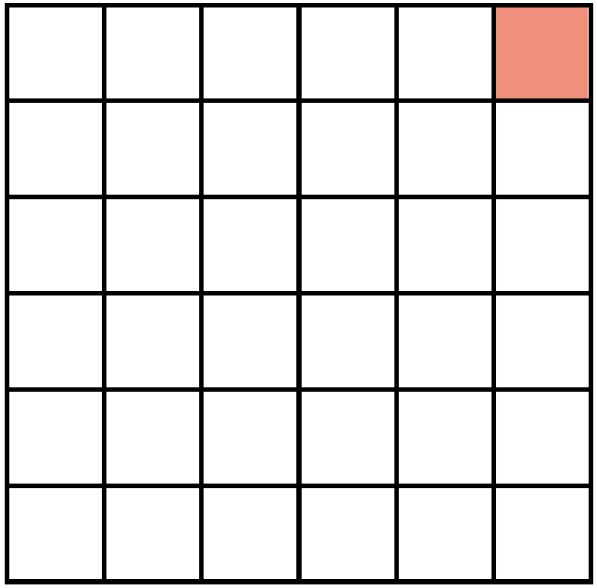
Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

During convolution, the weights “slide” along the input to generate each output



Input

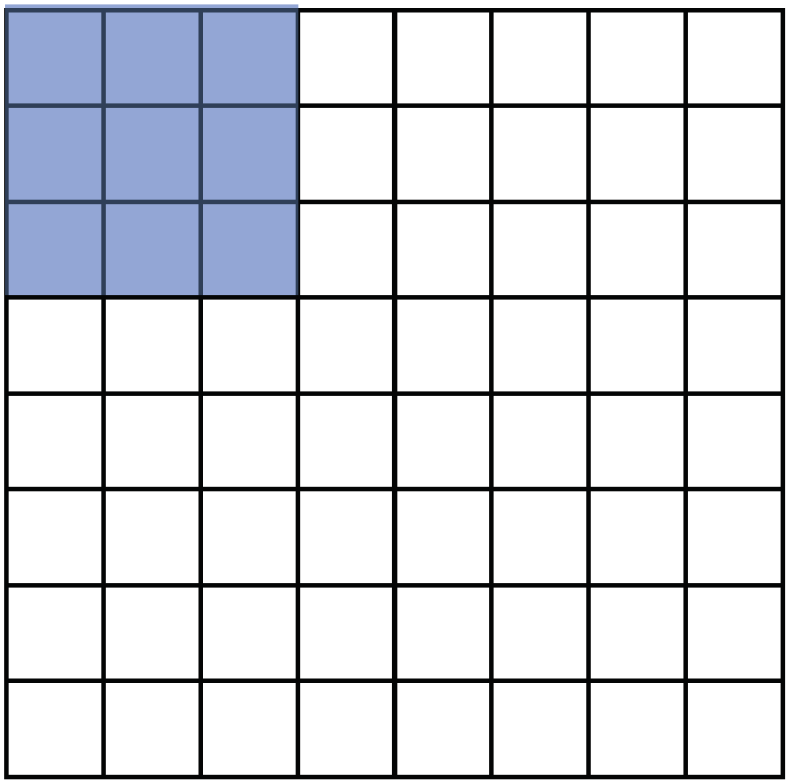


Output

Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

During convolution, the weights "slide" along the input to generate each output



Input

Recall that at each position, we are doing a **3D** sum:

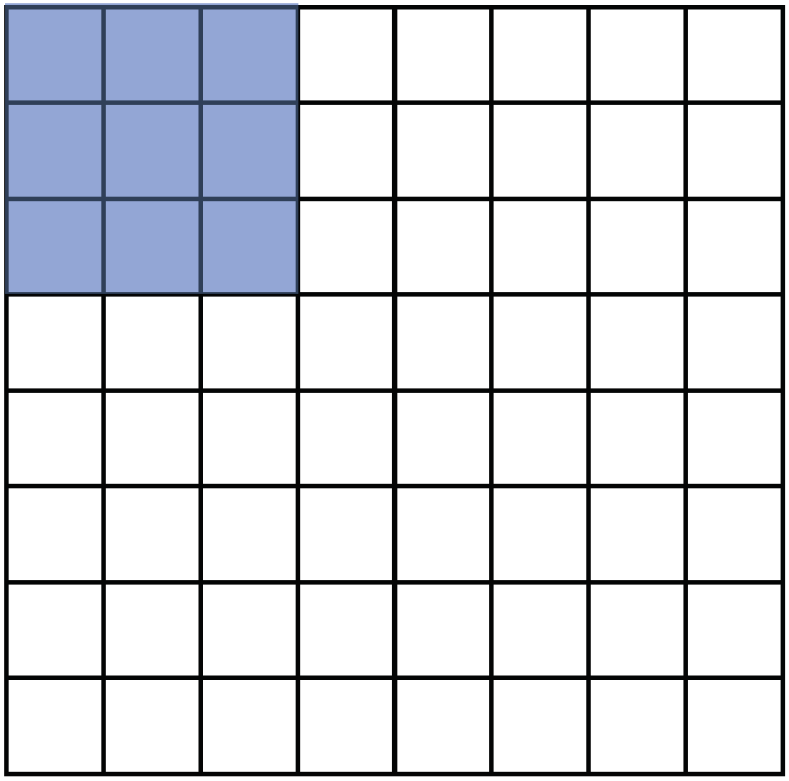
$$h^r = \sum_{ijk} x^r{}_{ijk} W_{ijk} + b$$

(channel, row, column)

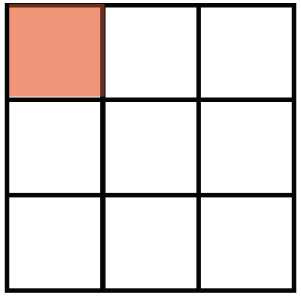
Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

But we can also convolve with a **stride**, e.g. stride = 2



Input

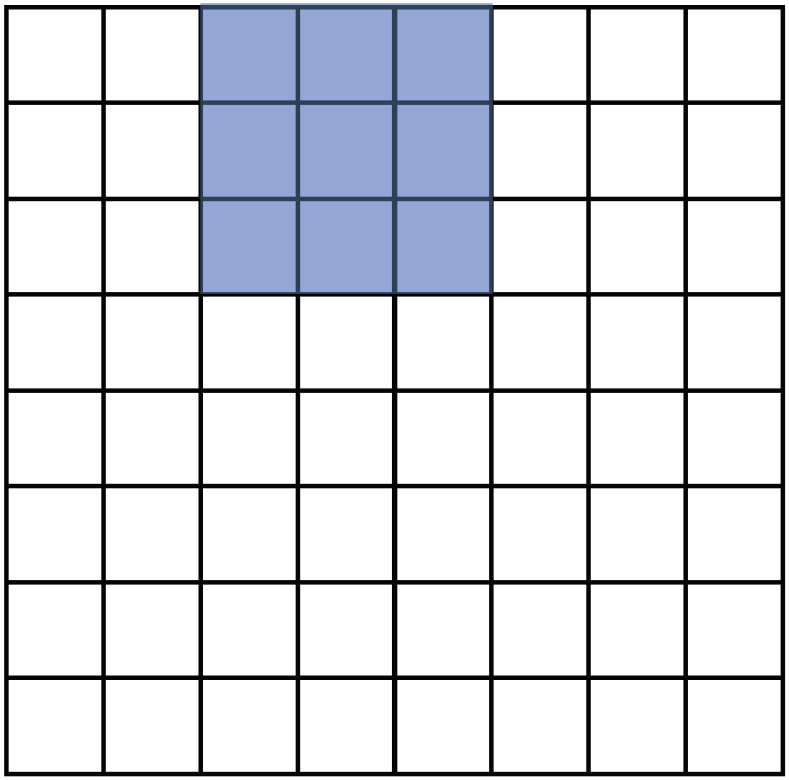


Output

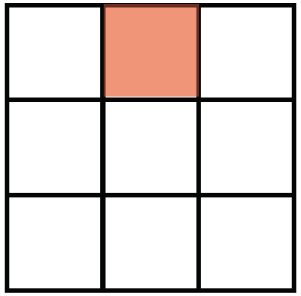
Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

But we can also convolve with a **stride**, e.g. stride = 2



Input

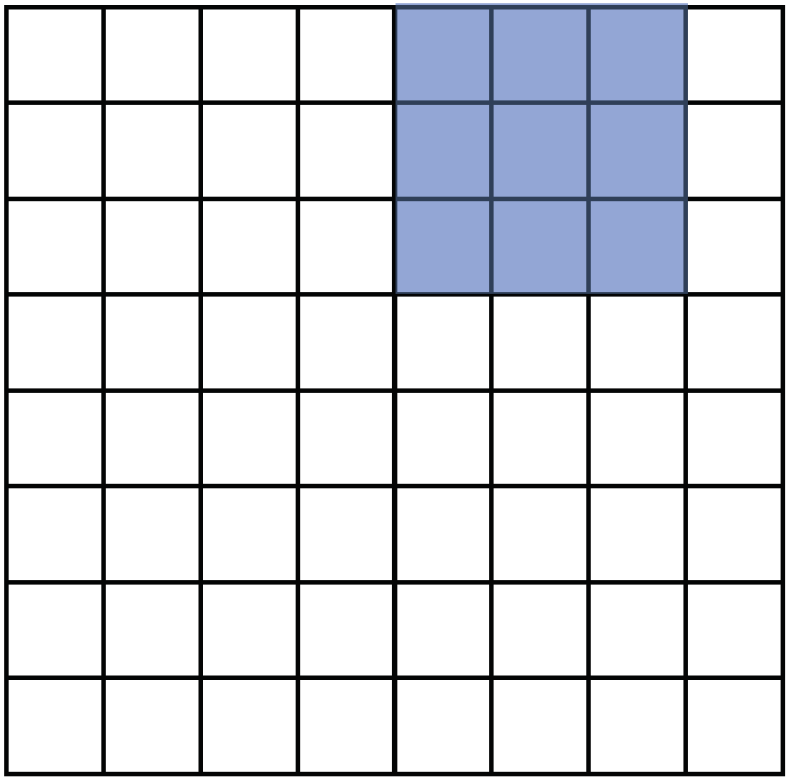


Output

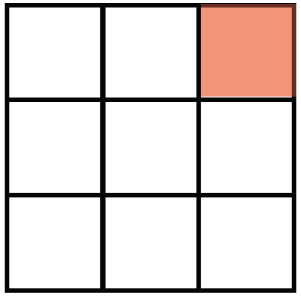
Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

But we can also convolve with a **stride**, e.g. stride = 2



Input

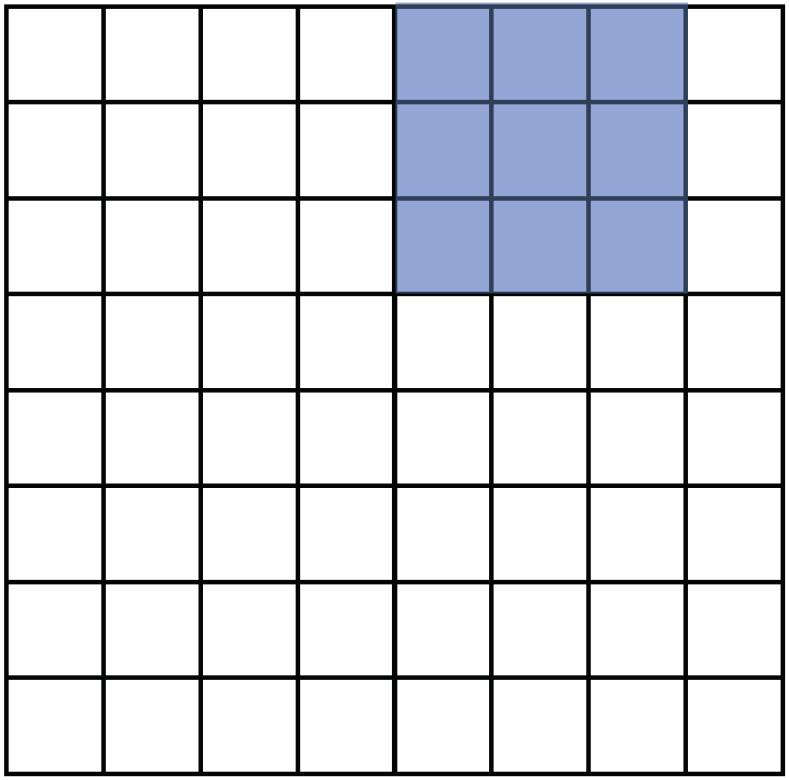


Output

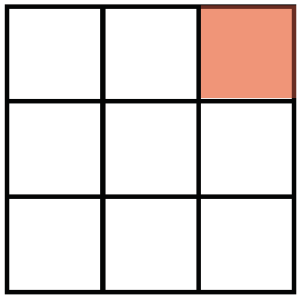
Source: Prof. Ioannis Gkioulekas, CMU

Convolution layer and Stride

But we can also convolve with a **stride**, e.g. stride = 2



Input



Output

- Notice that with certain strides, we may not be able to cover all of the input

- The output is also half the size of the input

Source: Prof. Ioannis Gkioulekas, CMU

Convolution layers and image dimension

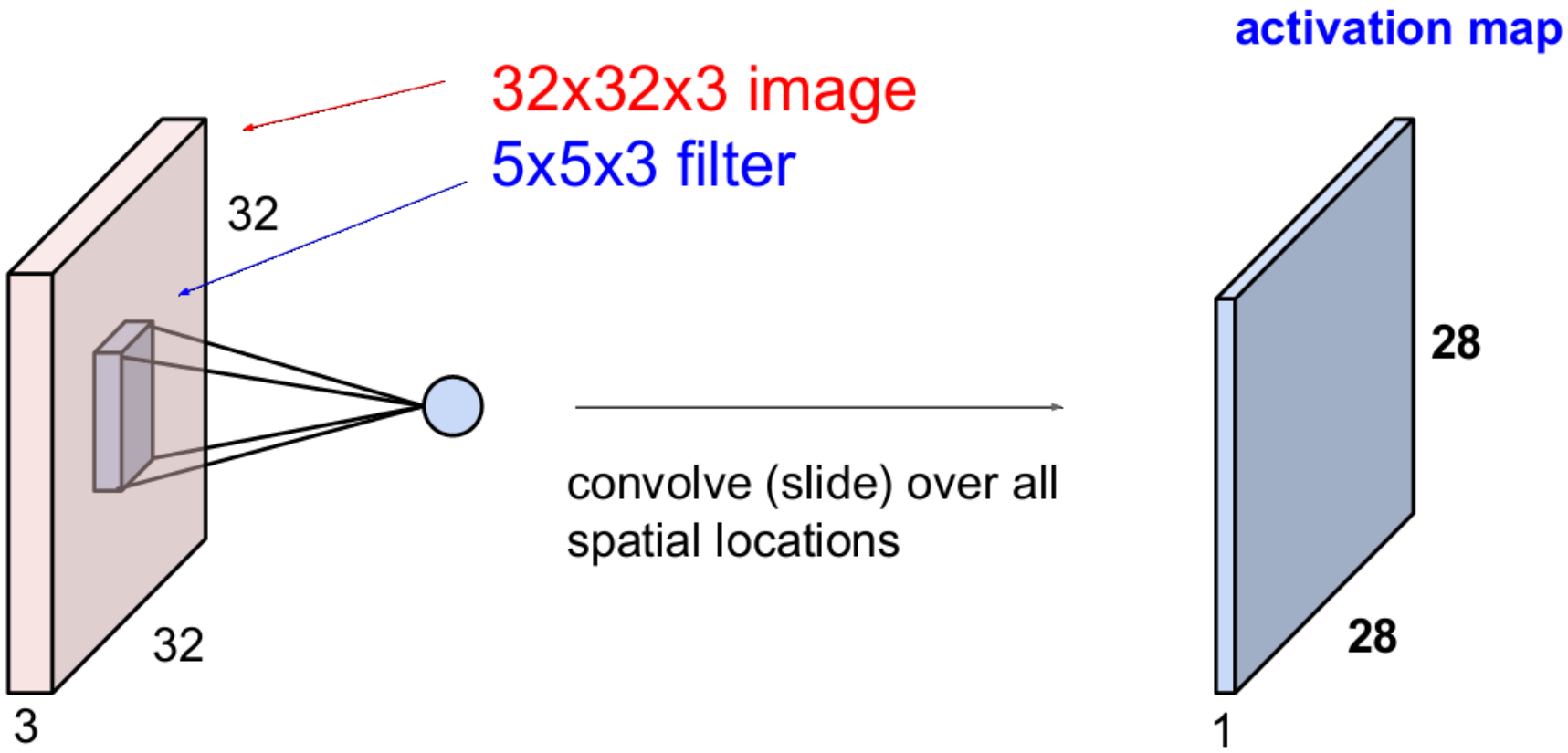


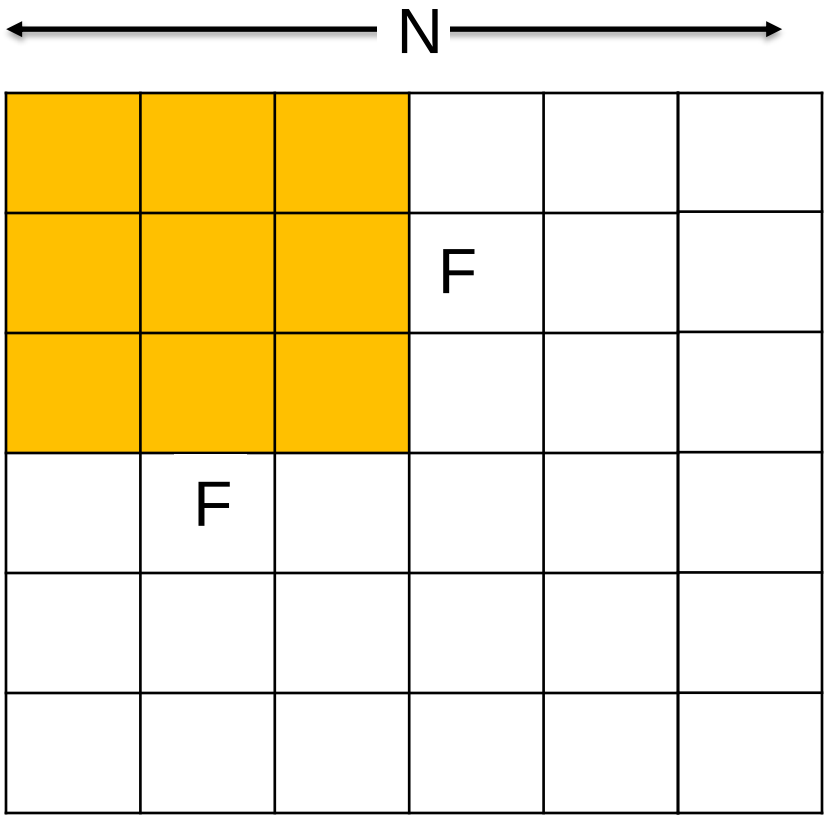
Figure source: Fei Fei Li's Lecture slides, Stanford University

Output Volume

- Operation with filters produces an output volume.
- Filter values are initially chosen randomly.
- Gradually, values of filters are updated in each iteration. Each filter learns to detect a small pattern.
- Suppose the image is 256X256X3 and there are 8 filters, each of size 5X5, what will be the output volume ?
- Each filter 5x5 will be converted to a 5x5x3 filter volume.
- Then it will convolve with each block of 5x5x3 in the image to give just one output value.
- Whole convolution process with one filter will give a layer of size 252X252.
- For 8 filters, it will be a 252X252X8 volume.

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution Layer and Output Dimension



Output size:

$$\frac{N - F}{stride} + 1$$

$$Stride = 1$$

N= 6 and F = 3, Stride S = 1.
What will be the output volume?

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution Layers and Output Dimensions

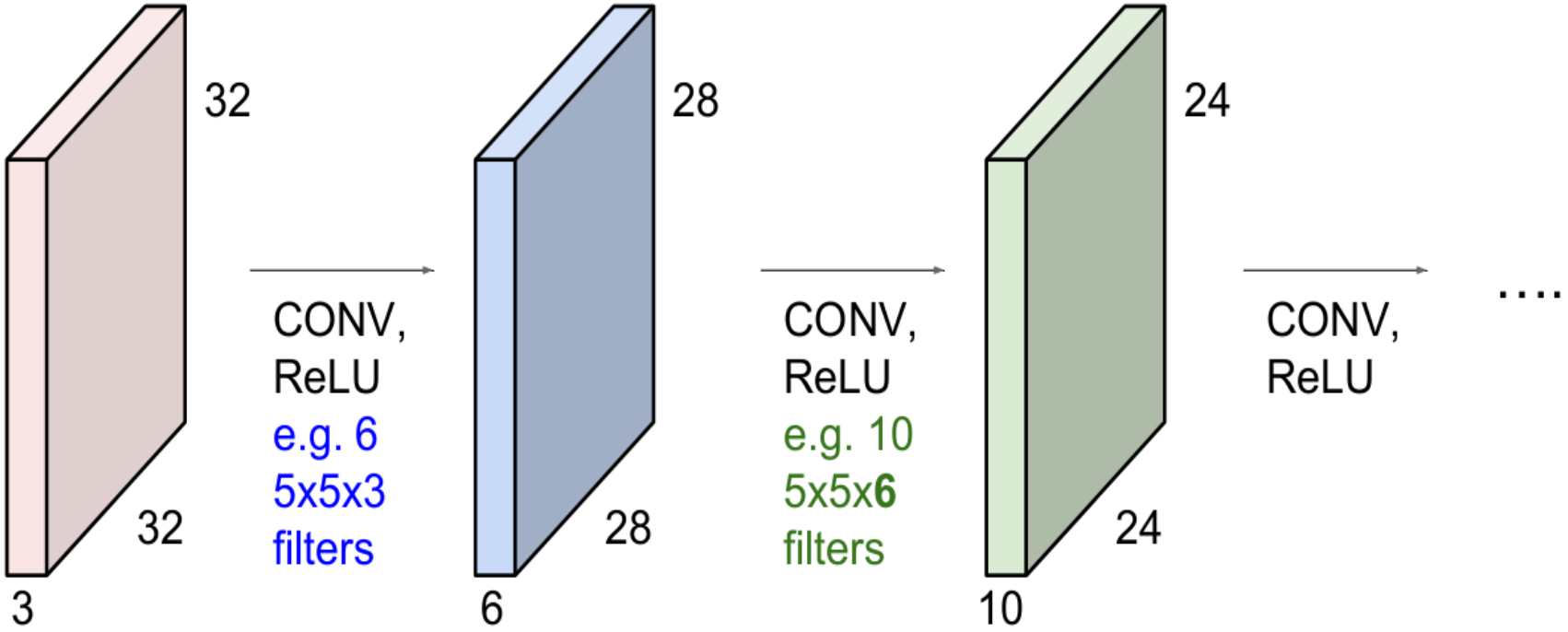
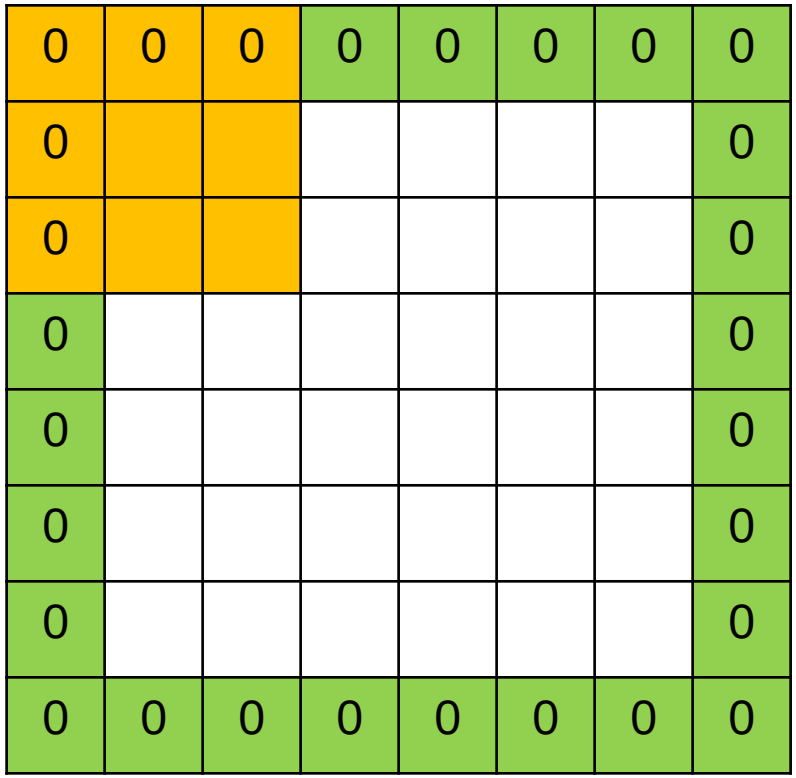


Figure source: Fei Fei Li's Lecture slides, Stanford University

Zero Padding

- In practice: Common to zero pad the border

- Two advantages
  - Dimension reduction is controlled.
  - Corner pixels have better contribution.



Source: Prof. A. Ojha, IIITDM Jabalpur

Zero Padding

- Example. Input image of size 7x7
- 3x3 filter, applied with stride 1
- Pad with 1 pixel border. what will be the output?
  - 7x7
- CONV layers mostly use stride S = 1.
- So, what should be the size of zero padding to maintain the image's height and width ?
- (F-1)/2
- Example
  - F = 3 => zero pad with 1
  - F = 5 => zero pad with 2

Source: Prof. A. Ojha, IIITDM Jabalpur

Input and Output Volumes

- Suppose the input volume is 64 x 64 x 3
- Apply 10 filters each of size 5 x 5
- Padding P=2, stride S=1
- What will be the output volume?
- Number of parameters in a layer?
- How many parameters to learn?

(Hint: After convolution apply bias)

Source: Prof. A. Ojha, IIITDM Jabalpur

Input and Output Volumes

- Now device a formula if the input volume is of the size : $WXHX3$
- Width : $\left\lfloor \frac{W-F+2P}{S} \right\rfloor + 1$
- Height : $\left\lfloor \frac{H-F+2P}{S} \right\rfloor + 1$
- Depth = number of filters.

Source: Prof. A. Ojha, IIITDM Jabalpur

Convolution vs Fully Connected Layer

- Fully connected layer
  - The input image size is $32 \times 32 \times 3$
  - Flatten it first (convert to 1-d vector)
  - Dimension of input vector: $32 \times 32 \times 3 = 3072 \times 1$
  - If there are 10 units (neuron) in the first hidden layer of a FC network, then each unit looks at the full input vector

$$a = f(Z) = f(W^T X + b).$$

Trainable parameters

Weights: 3072×10

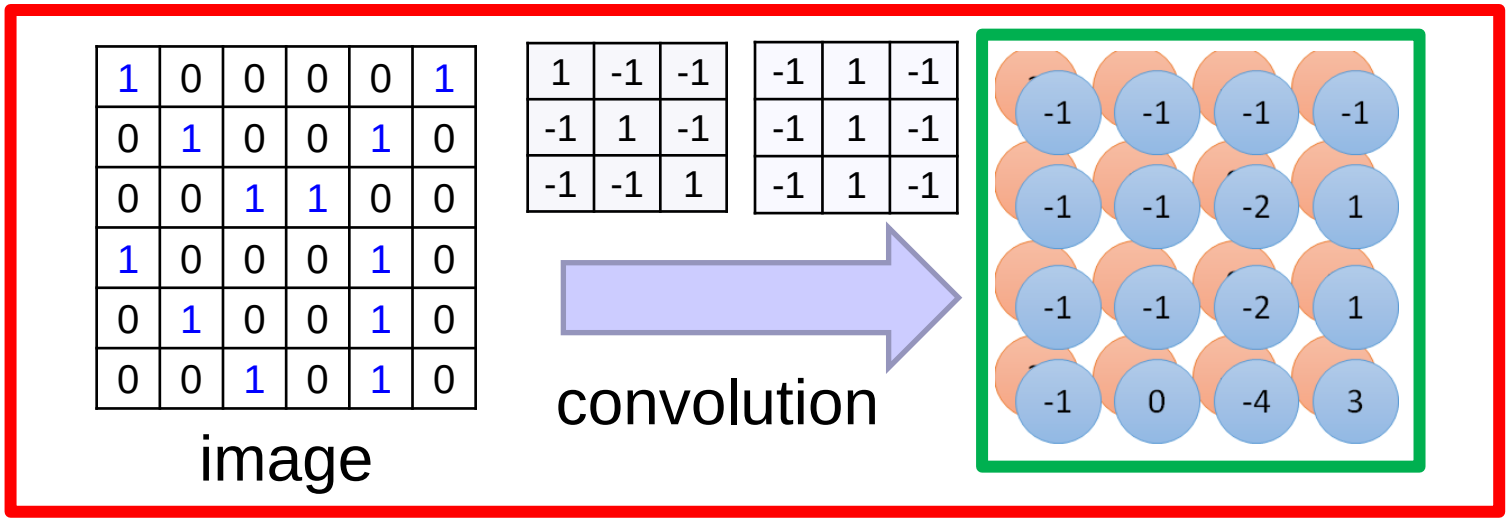
Bias: 10

Total trainable parameters : 30730

Convolution vs Fully Connected Layer

- Convolution layer
  - The input image size is 32 × 32 × 3
  - If there are 10 filters, each of size 3 × 3 × 3 in the first Conv layer of the network.
  - Total trainable parameters ?
  - 280

Convolution vs Fully Connected layer



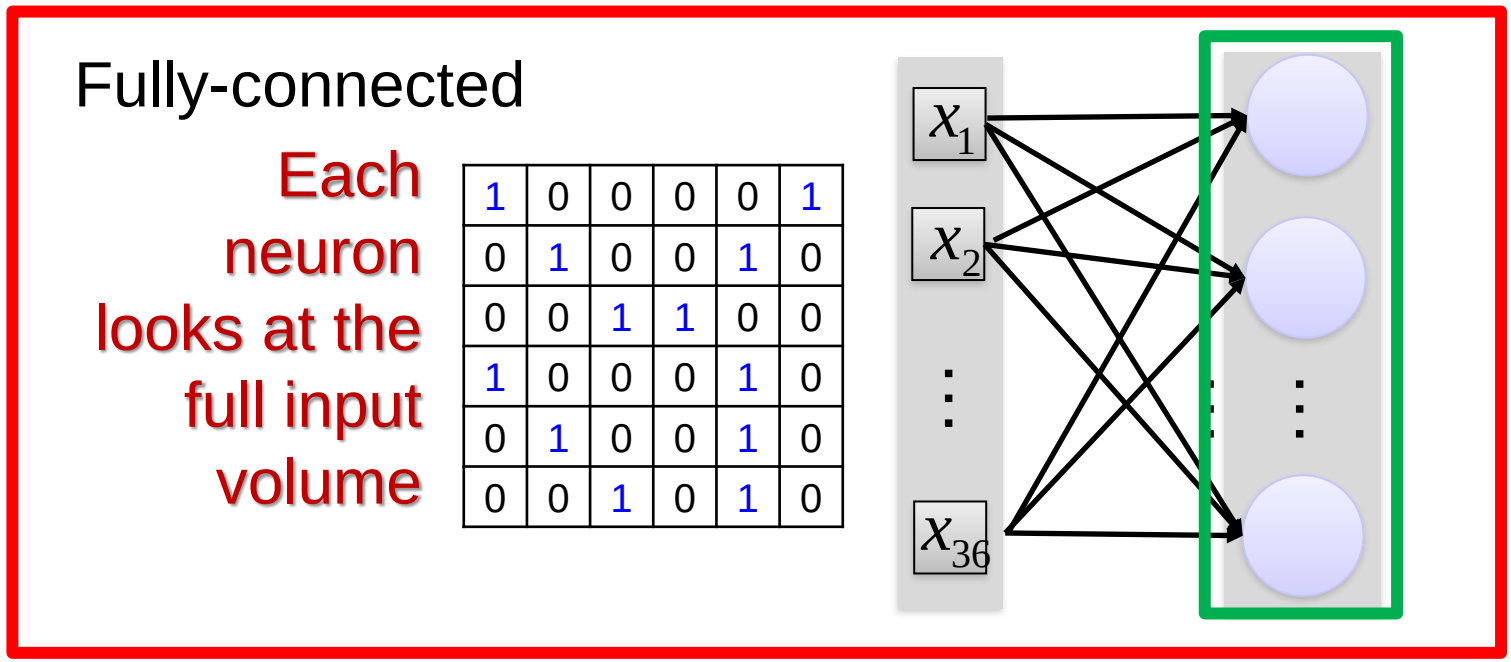
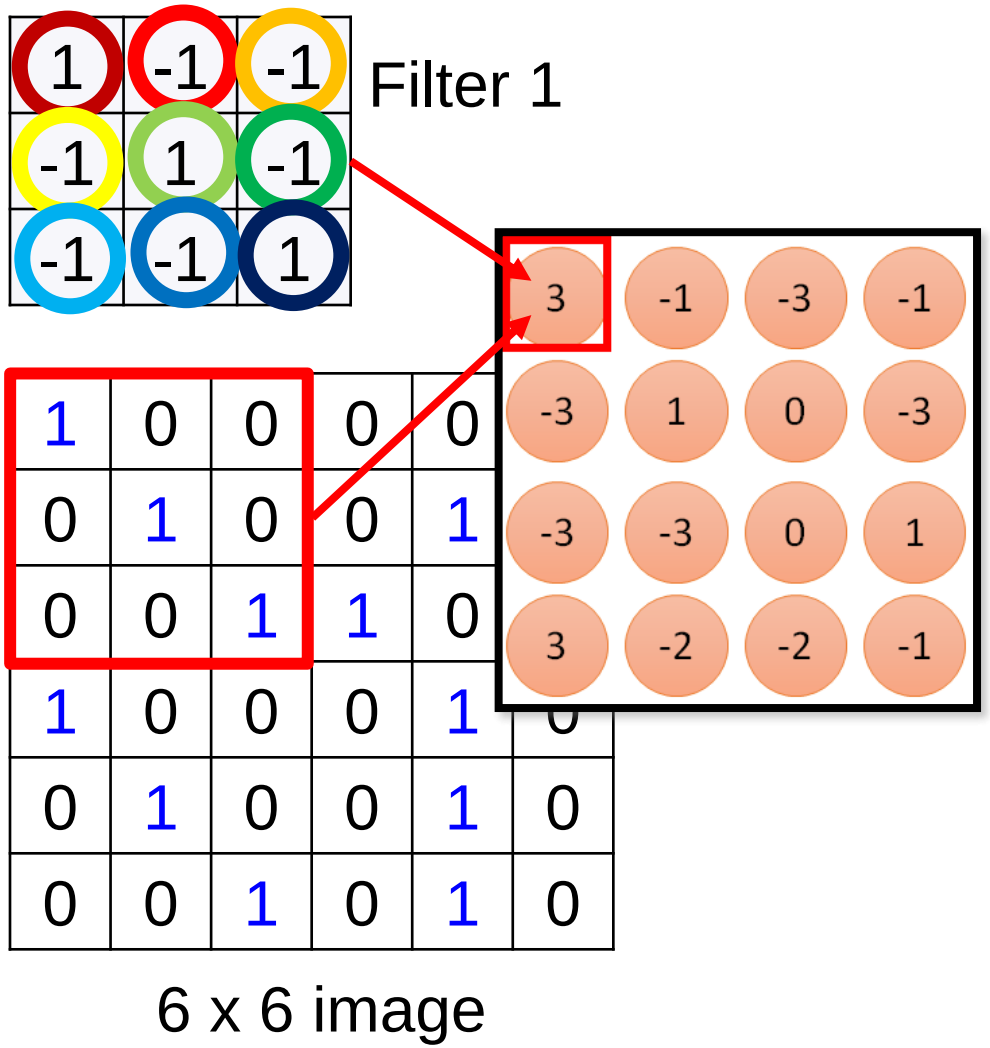
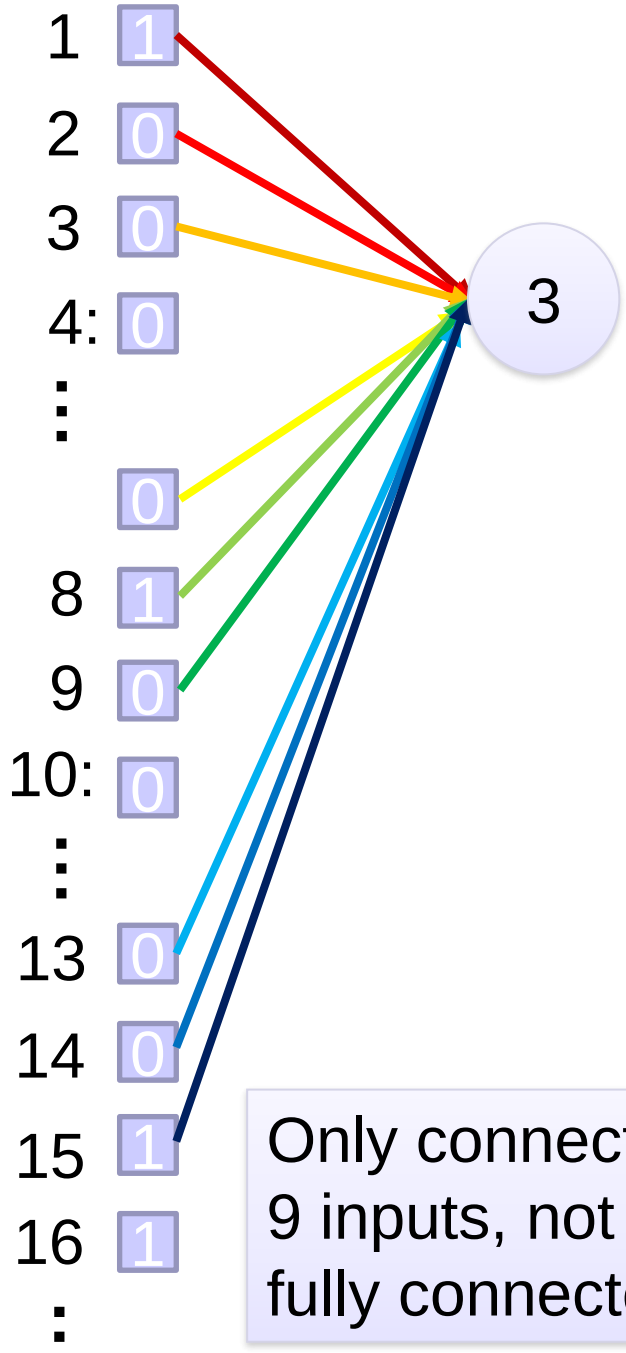


Figure source: Ming Li's Lecture slides, Waterloo University

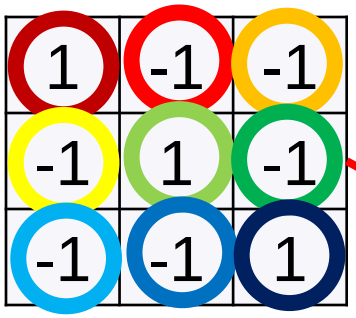


fewer parameters!

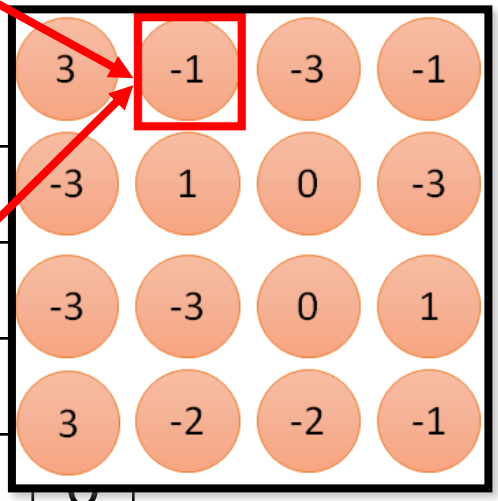


Only connect to 9 inputs, not fully connected

Figure source: Ming Li's Lecture slides, Waterloo University



Filter 1

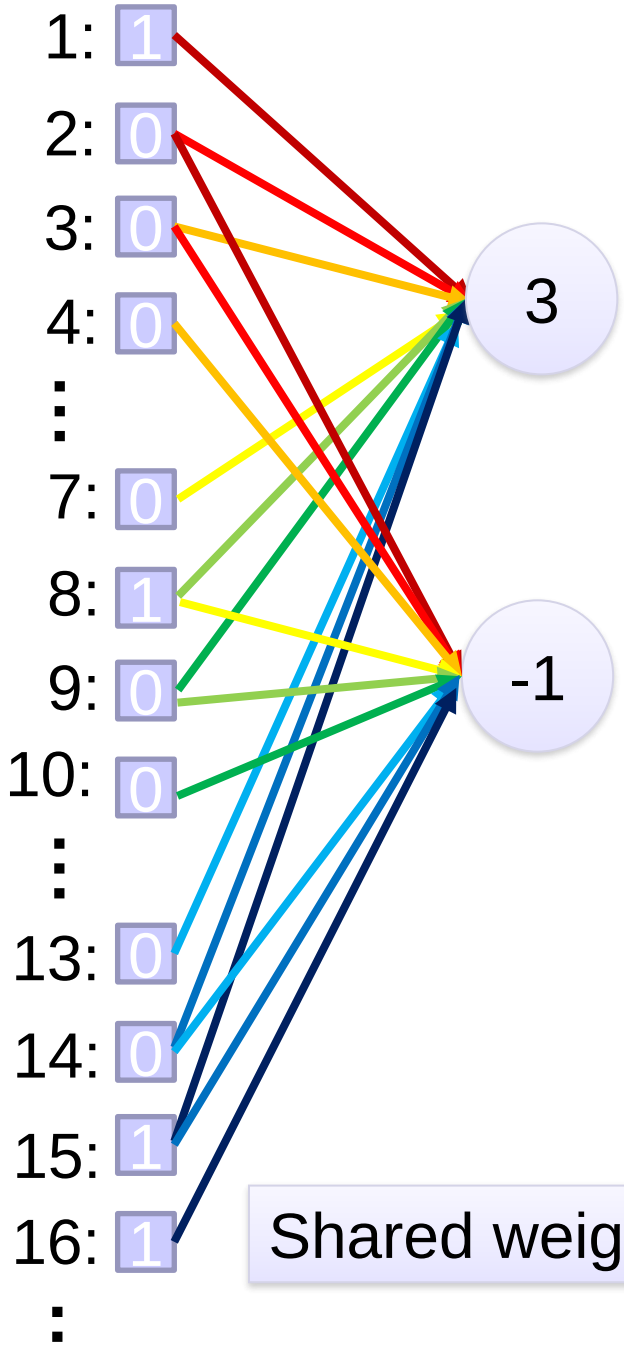


1	0	0	0	0	
0	1	0	0	1	
0	0	1	1	0	
1	0	0	0	1	
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

Fewer parameters

Even fewer parameters



Shared weights

Figure source: Ming Li's Lecture slides, Waterloo University

The whole CNN Architecture

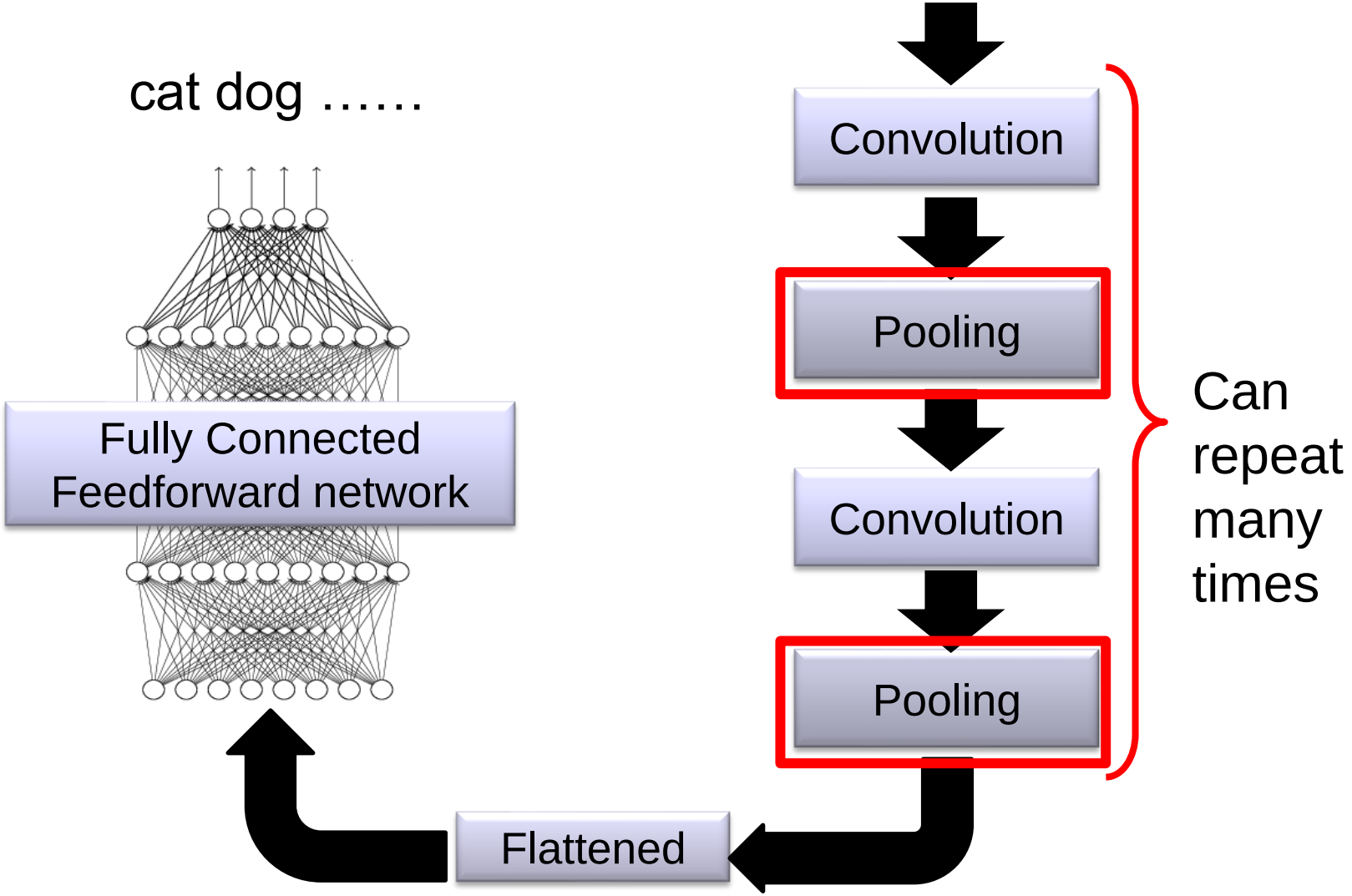


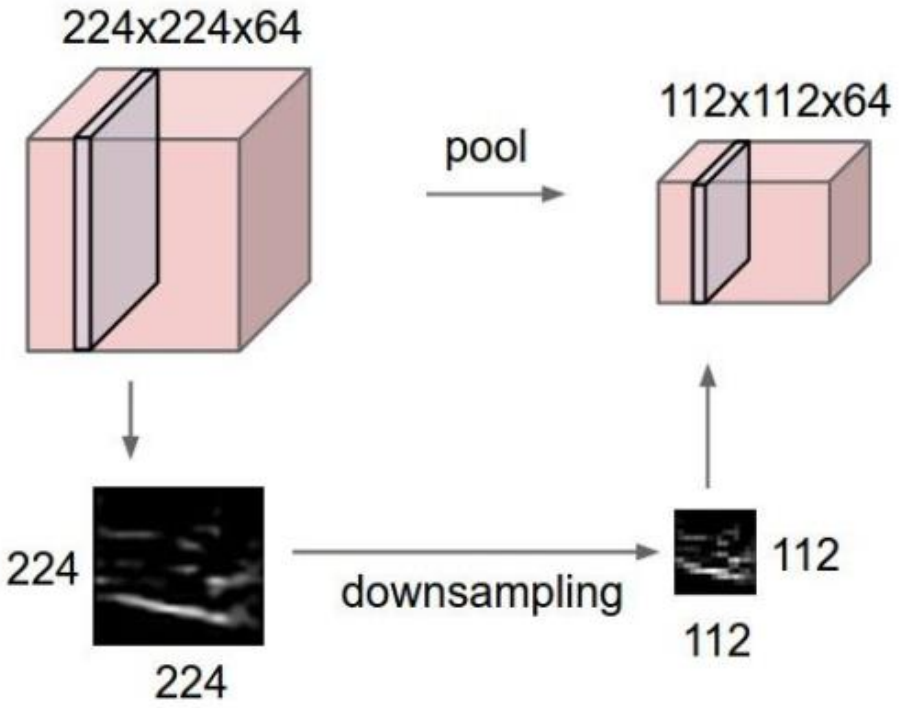
Figure source: Ming Li's Lecture slides, Waterloo University

Pooling layer

- When the image is large, one needs to compress the same for managing the input and output volumes between different convolution layers.
- Pooling layers make the representations smaller.
- Output volumes are more manageable.
- Operates over each activation map independently.

Source: Prof. A. Ojha, IIITDM Jabalpur

Pooling layer

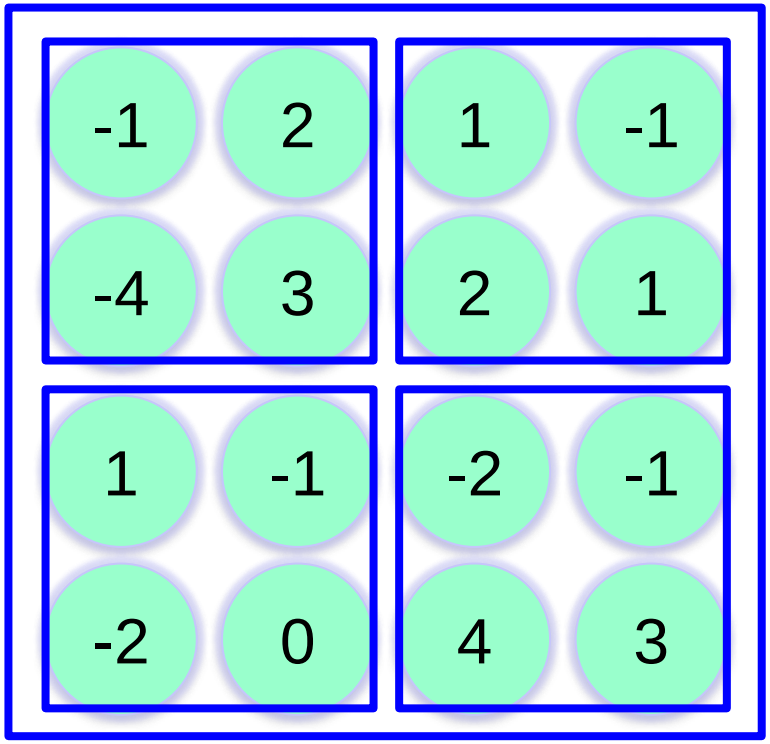


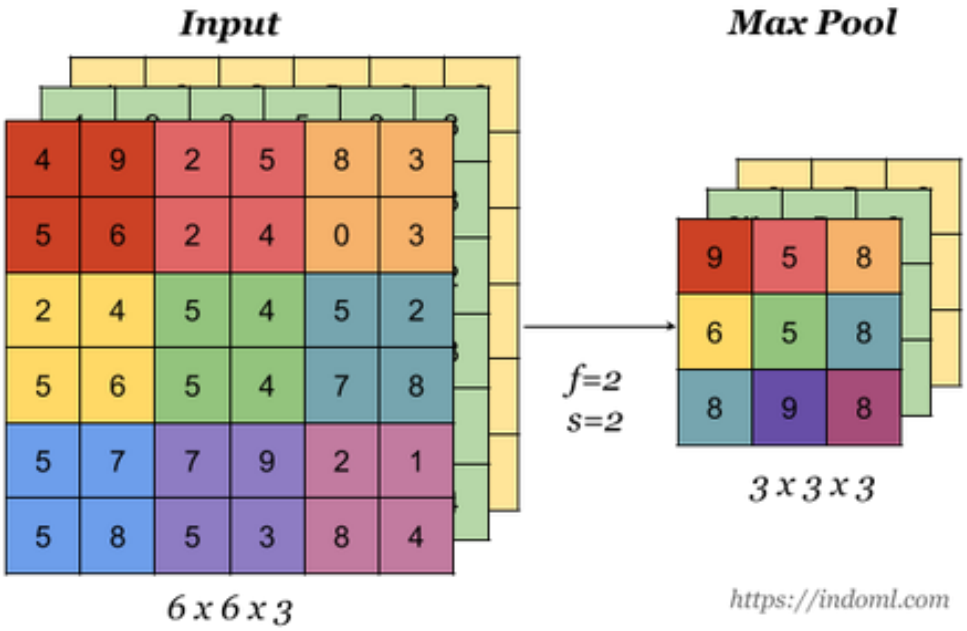
- Max pooling
- Average pooling
- Others: Min pooling.

source: Fei Fei Li's Lecture slides, Stanford University

Max Pooling

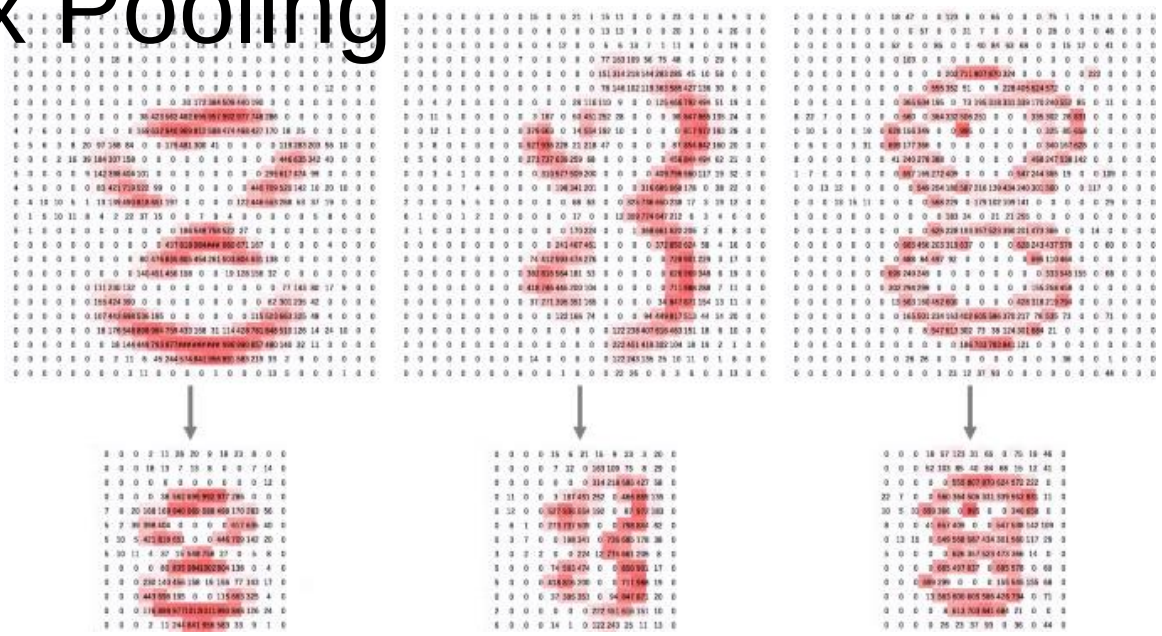
- With 2 × 2 filters and stride 2





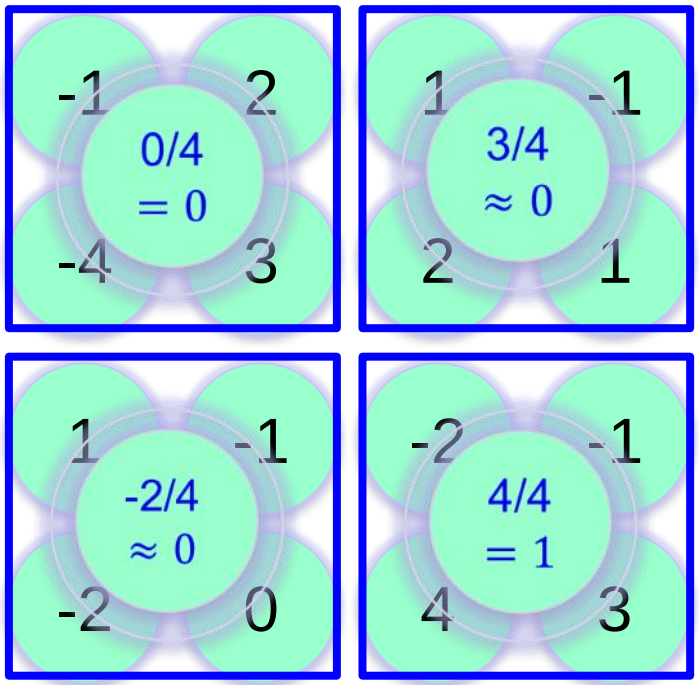
Source: Prof. A. Ojha, IIITDM Jabalpur

Max Pooling



Average Pooling

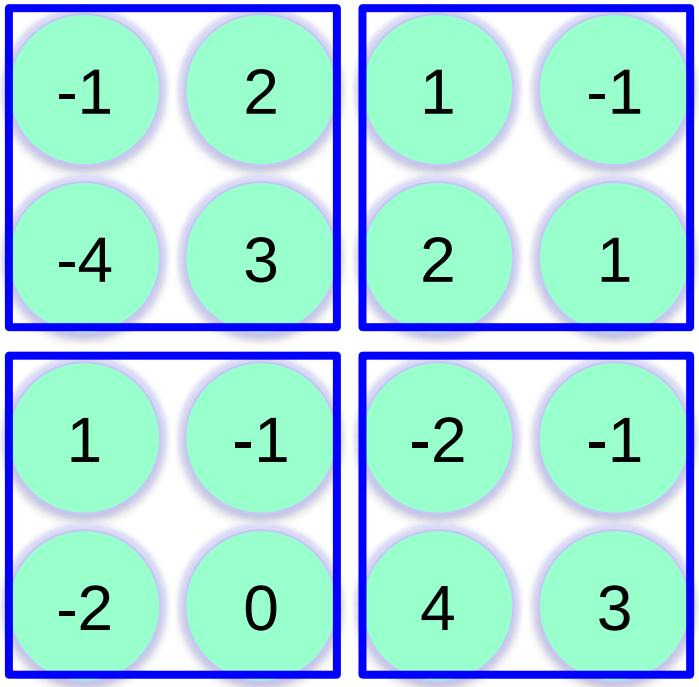
- With 2×2 filters and stride 2



Source: Prof. A. Ojha, IIITDM Jabalpur

Min Pooling

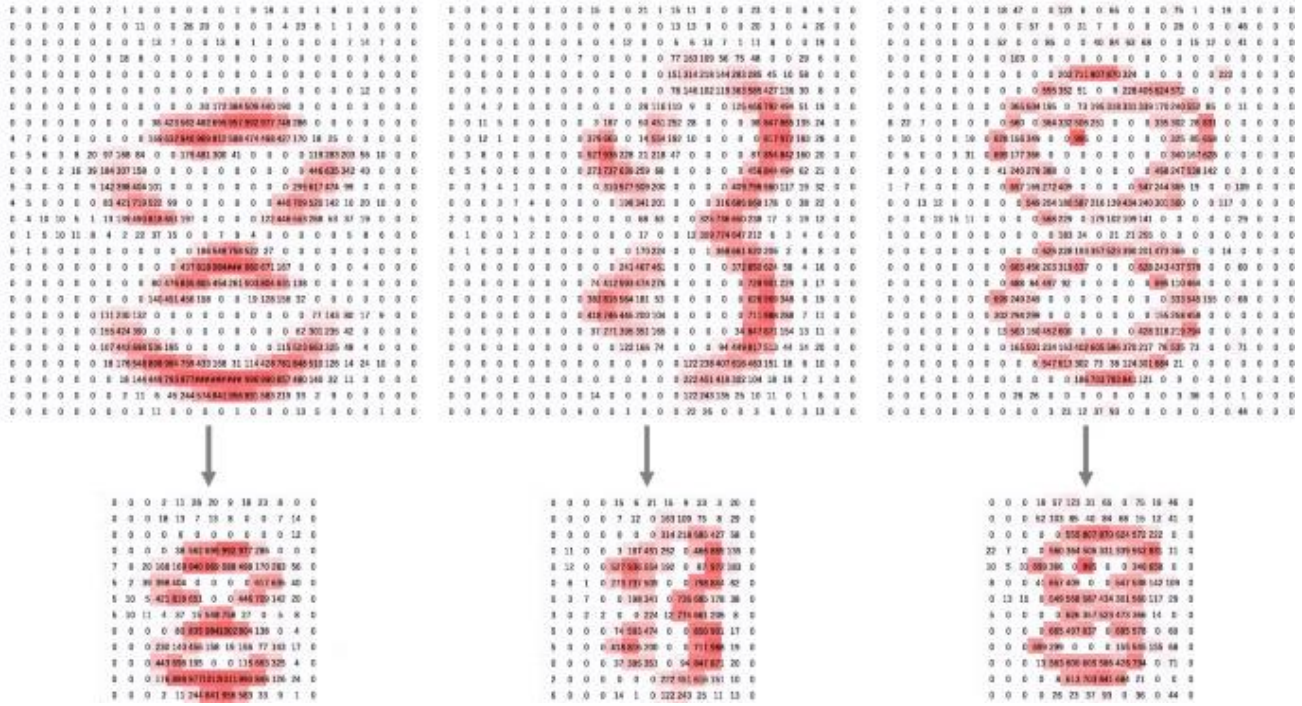
- With 2 × 2 filters and stride 2



Source: Prof. A. Ojha, IIITDM Jabalpur

Max Pooling – Another Example

- With 2 × 2 filters and stride 2
- Note that images in the upper row are downsampled by a factor of 2, from 26 × 26 to 13 × 13



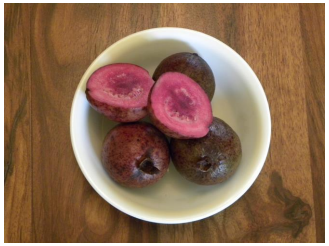
https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9/figures/6

Pooling Advantage

- Pooling is a kind of subsampling.
- Subsampling the image does not loose image information.

A bowl of food on a plate

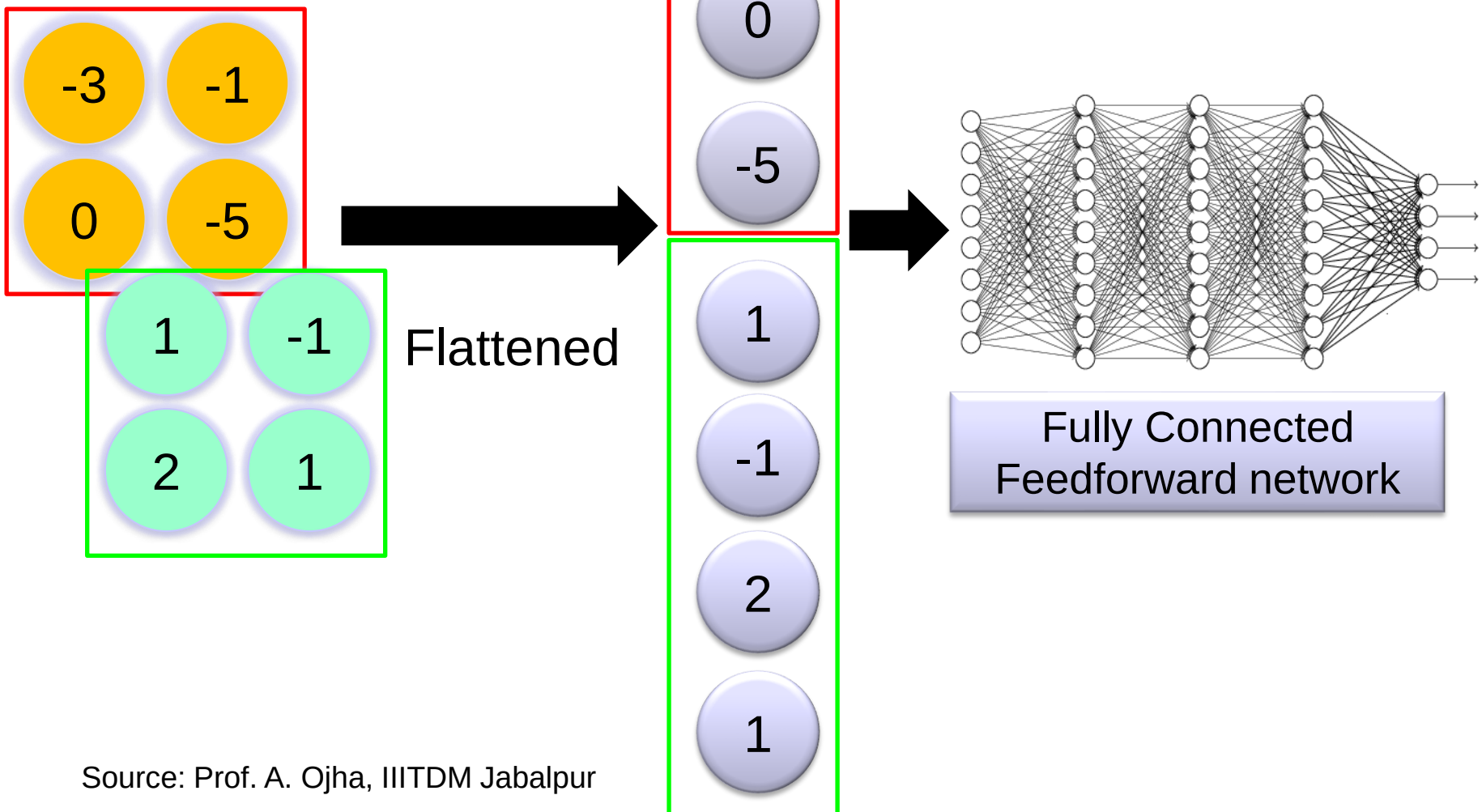
A bowl of food on a plate



Fewer parameters needed to extract features for pattern recognition and characterizing the image.

Source: Prof. A. Ojha, IIITDM Jabalpur

Flattening



what's left: Fully Connected Layers

Source: Prof. A. Ojha, IIITDM Jabalpur

A Generic CNN Architecture

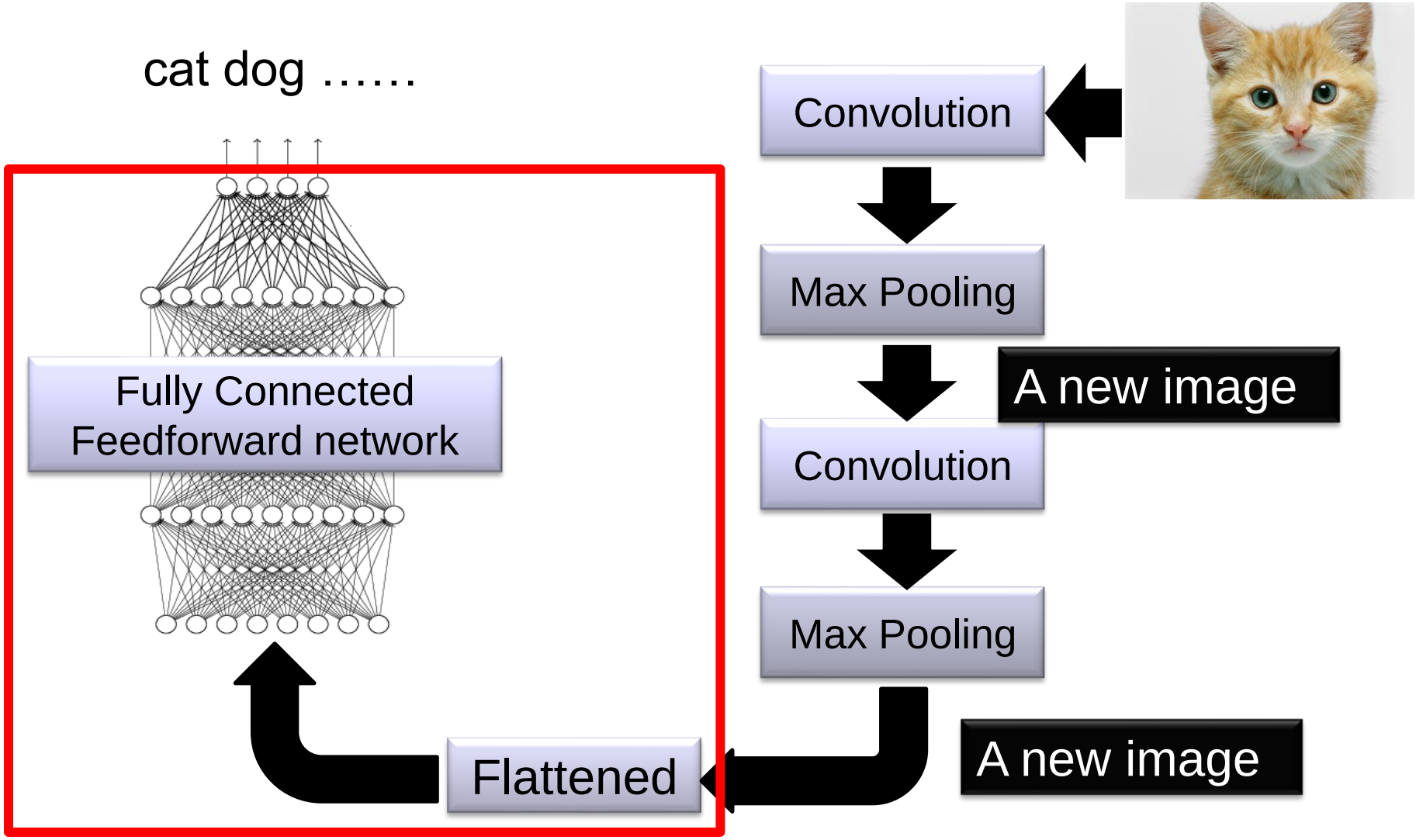
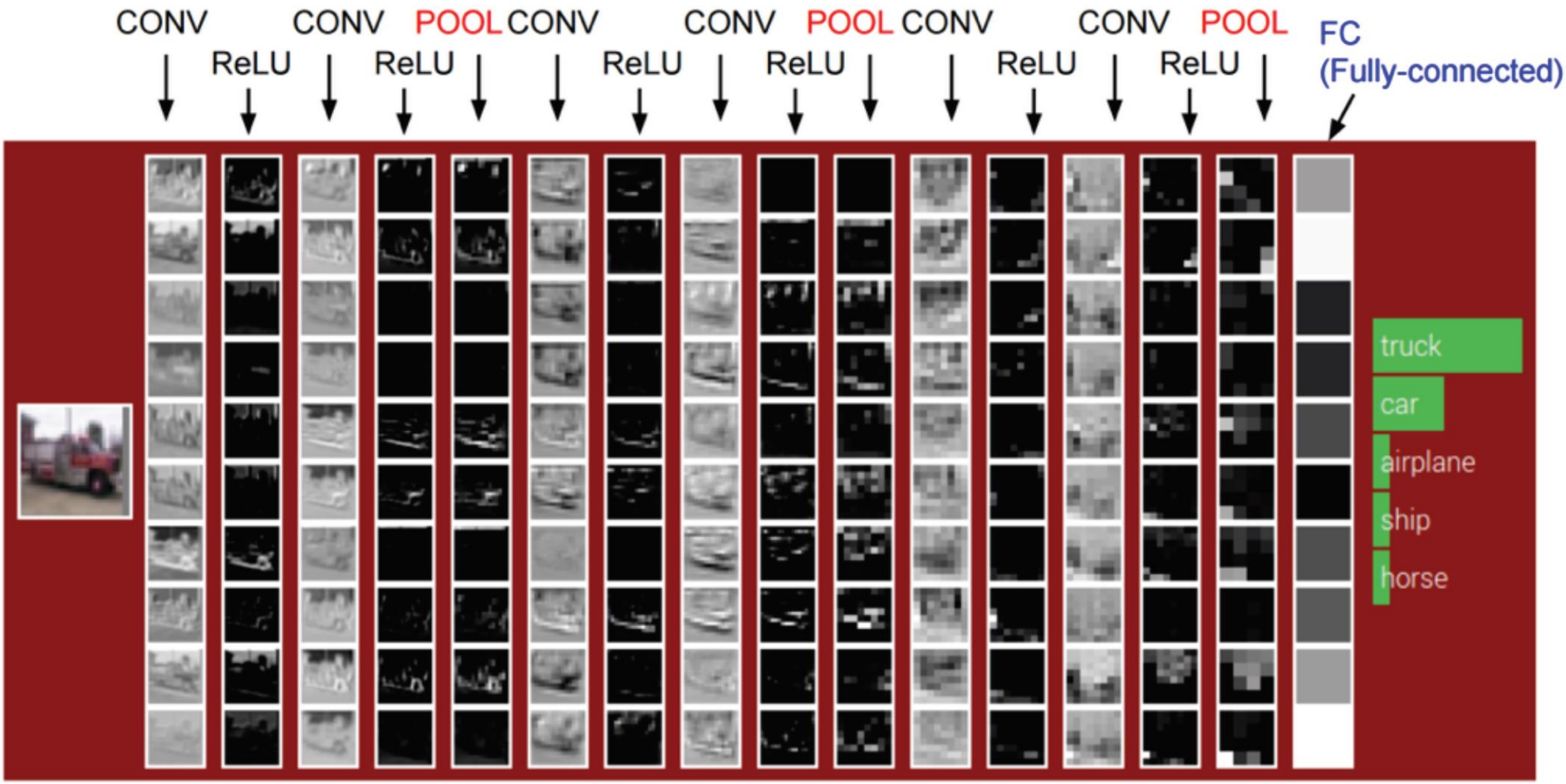


Figure source: Ming Li's Lecture slides, Waterloo University

Another Example ConvNet



10x3x3 conv filters, stride 1, pad 1

2x2 pool filters, stride 2

Figure: Andrej Karpathy

Source: Prof. Ioannis Gkioulekas, CMU

CNN Advantage in Dealing High Dimension Data

- Number of connections are reduced (Sparse interactions).
- Parameter sharing.
- Equivariant representations.
- Ability to work with inputs of variable size.
- Pooling layers compress the output volumes.
- As a result, number of parameters to learn are much less.
- The number of training parameters depends on how many filters you choose for training.

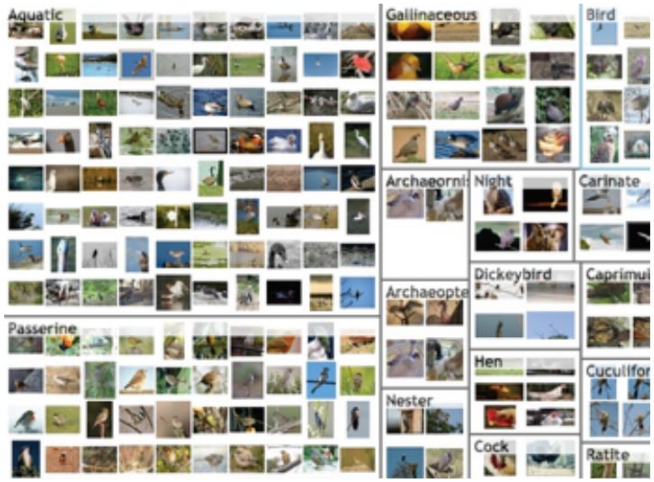
Source: Prof. A. Ojha, IIITDM Jabalpur

Training a CNN

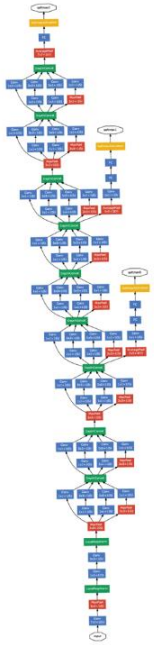
How do you actually train these things?

Roughly Speaking:

Gather labeled data



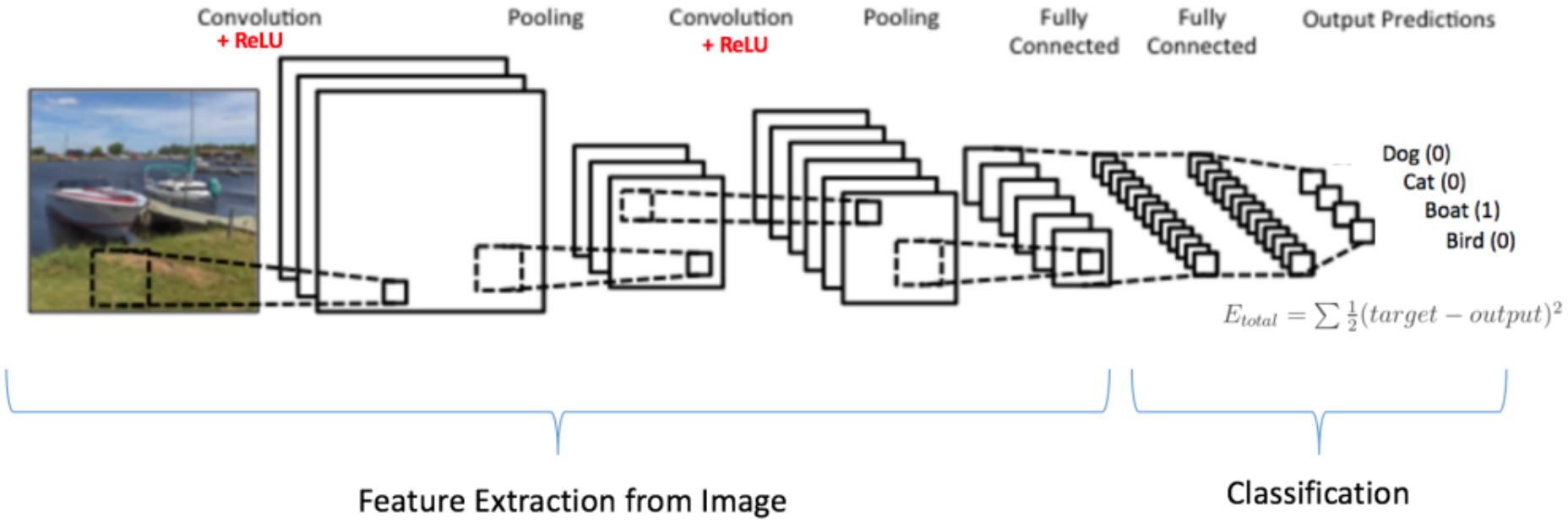
Find a ConvNet architecture



Minimize the loss

Source: Prof. Ioannis Gkioulekas, CMU

Training a CNN - Example



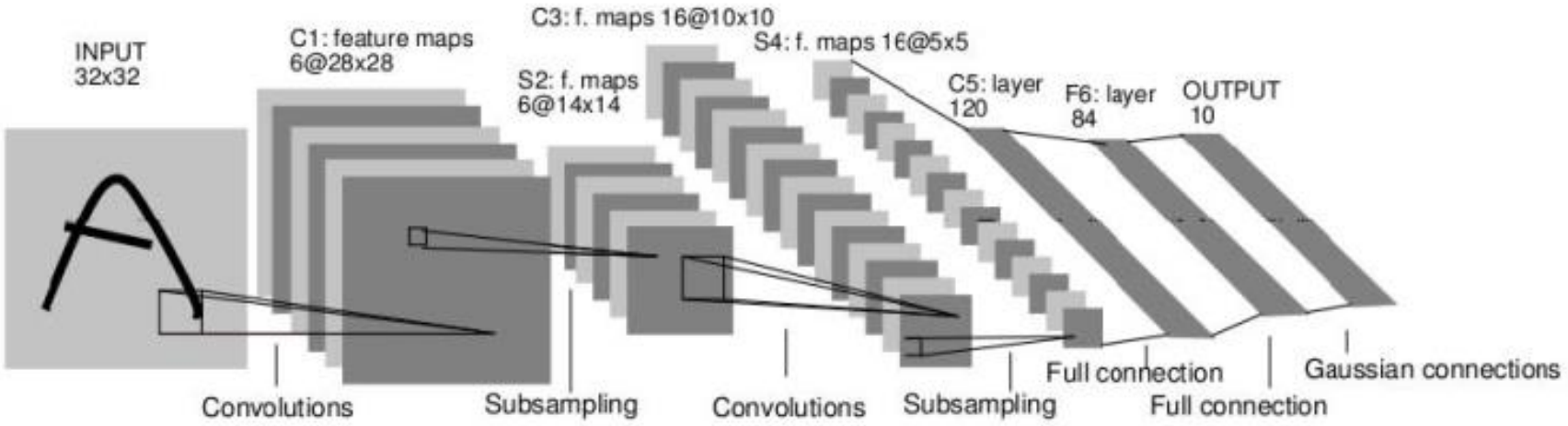
https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

CNN Architectures

- Case Studies
  - AlexNet - Krizhevsky, Sutskever, Hinton, 2012
    - 8 layers
  - VGG - Simonyan and Zisserman, 2014
    - 16-19 layers
  - GoogLeNet - Szegedy et al., 2014
    - 22 layers
  - ResNet – He et al., 2015
    - 152-layers
- Also....
  - NiN (Network in Network)
  - Wide ResNet
  - ResNeXT
  - Stochastic Depth
  - -dense network
  - -fractal network
  - -sqeezeNet

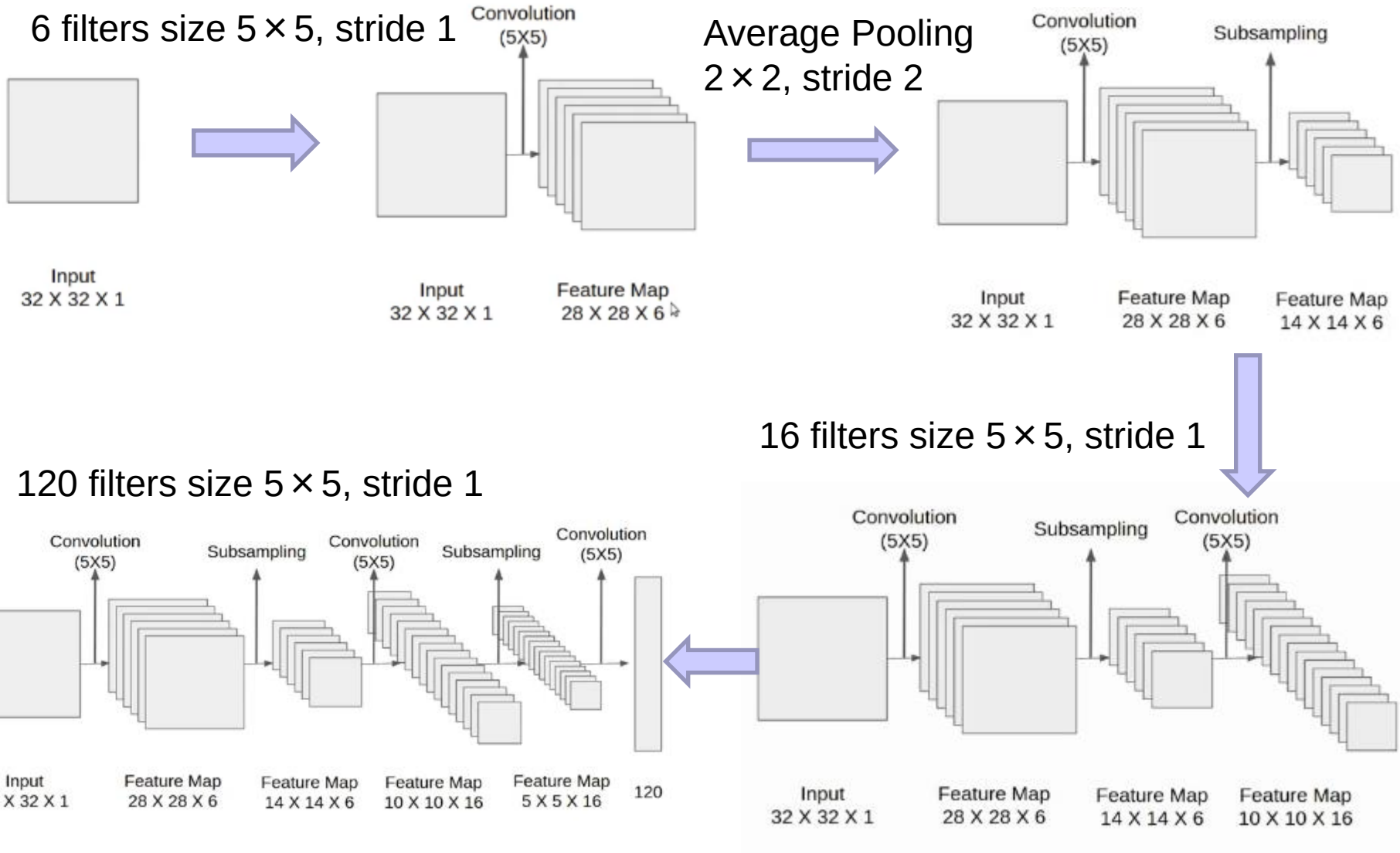
Source: Prof. A. Ojha, IIITDM Jabalpur

LeNet-5 (LeCun 1998)



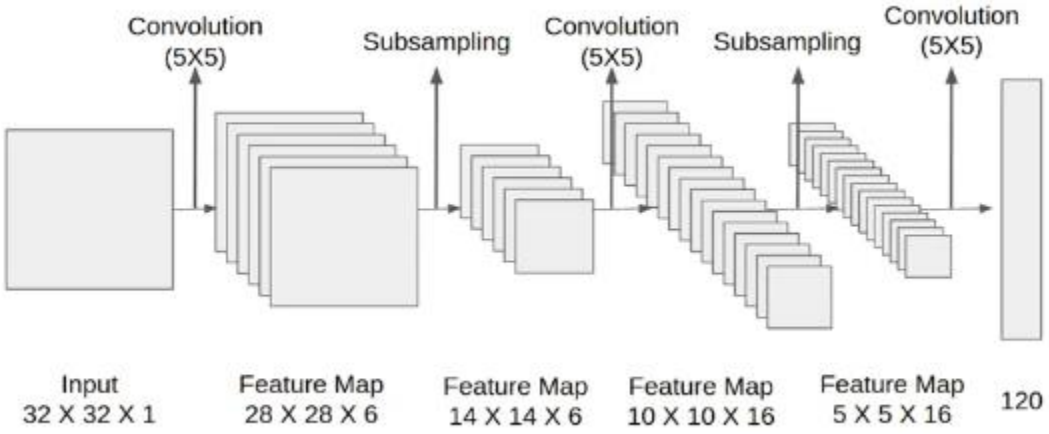
- 5 layers with learnable parameters 5 layers with learnable parameters
- How many parameters?

LeNet-5 Architecture



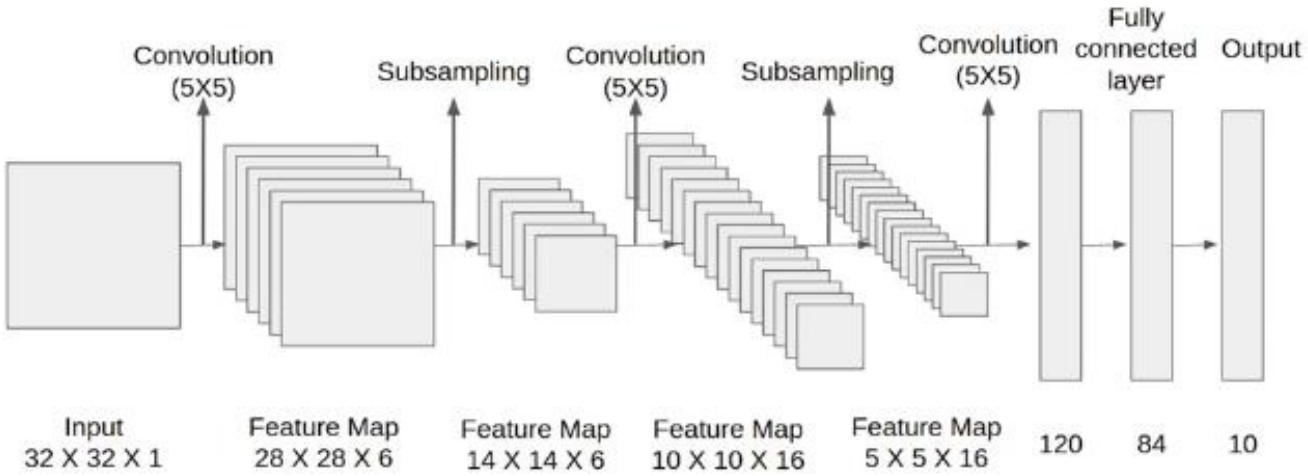
https://cdn.analyticsvidhya.com/wp-content/uploads/2021/03/Screenshot-from-2021-03-18-12-52-17.png

LeNet-5 Architecture



Fully connected layer with 84 neurons

Last layer is the output layer with 10 neurons and Softmax function



https://cdn.analyticsvidhya.com/wp-content/uploads/2021/03/Screenshot-from-2021-03-18-12-52-17.png

LeNet-5 Architecture

Layer	# filters / neurons	Filter size	Stride	Size of feature map	Activation function
Input	-	-	-	32 X 32 X 1	
Conv 1	6	5 * 5	1	28 X 28 X 6	tanh
Avg. pooling 1		2 * 2	2	14 X 14 X 6	
Conv 2	16	5 * 5	1	10 X 10 X 16	tanh
Avg. pooling 2		2 * 2	2	5 X 5 X 16	
Conv 3	120	5 * 5	1	120	tanh
Fully Connected 1	-	-	-	84	tanh
Fully Connected 2	-	-	-	10	Softmax

- How many parameters?
- Around 60,000

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

- The idea conceived by Prof. Fei-Fei Li
- Launched in 2009.
- Visual Dataset contains more than 15 million of labeled high-resolution images covering almost 22,000 categories.
- High quality dataset, highly popular among researchers to test their image classification model on this dataset.

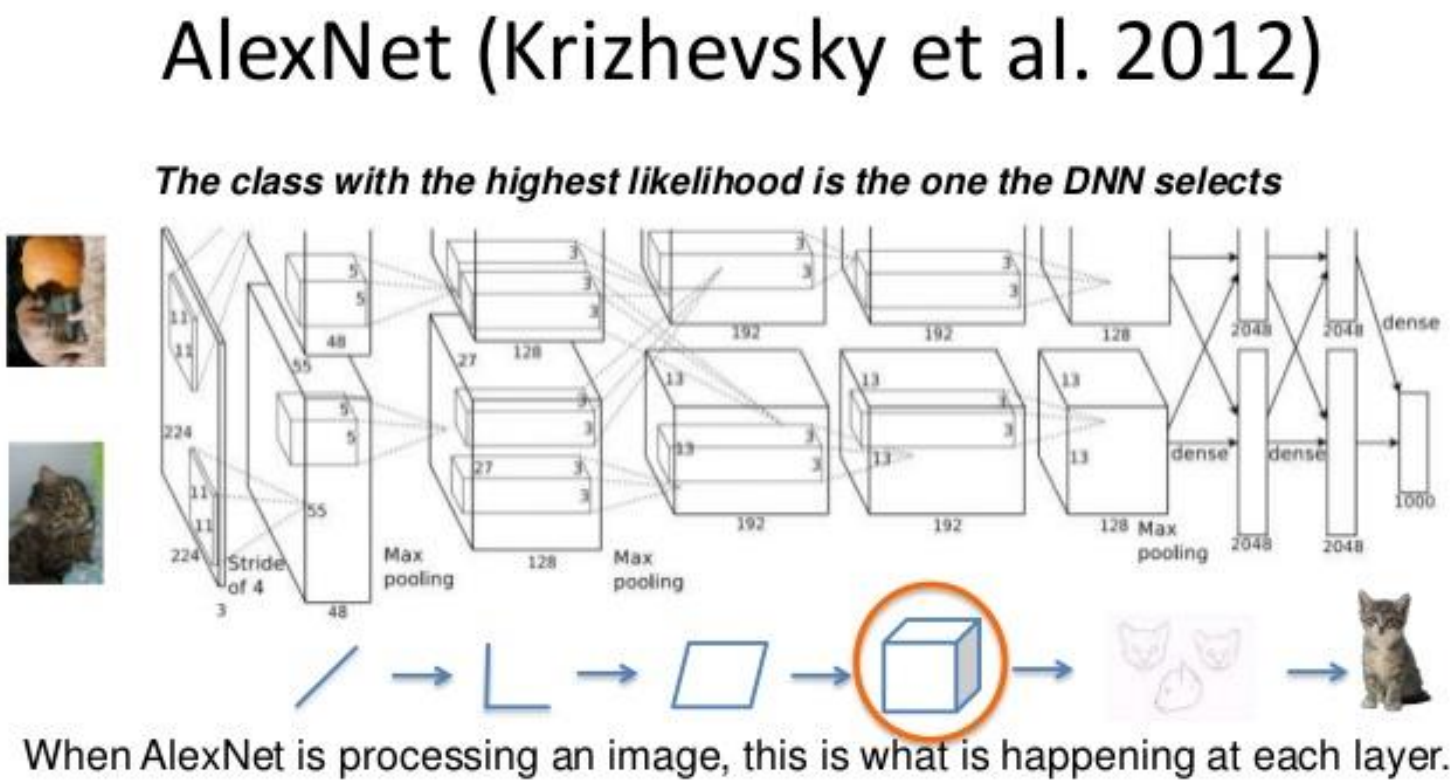
https://machinelearningknowledge.ai/popular-image-classification-models-in-imagenet-challenge-ilsvrc-competition-history/

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

- Launched in 2010; held yearly till 2017.
- Uses the smaller portion of the ImageNet consisting of only 1000 categories.
- The total count of training images is 1.3 million, accompanied by 50,000 validation images, and 1,00,000 testing images.
- Tasks: image classification, object detection, object localization
- Minimal top-1 and top-5 error rates achievers - winner.
- Top-5 error rate - the percent of images where the correct label is not one of the model's five most likely labels

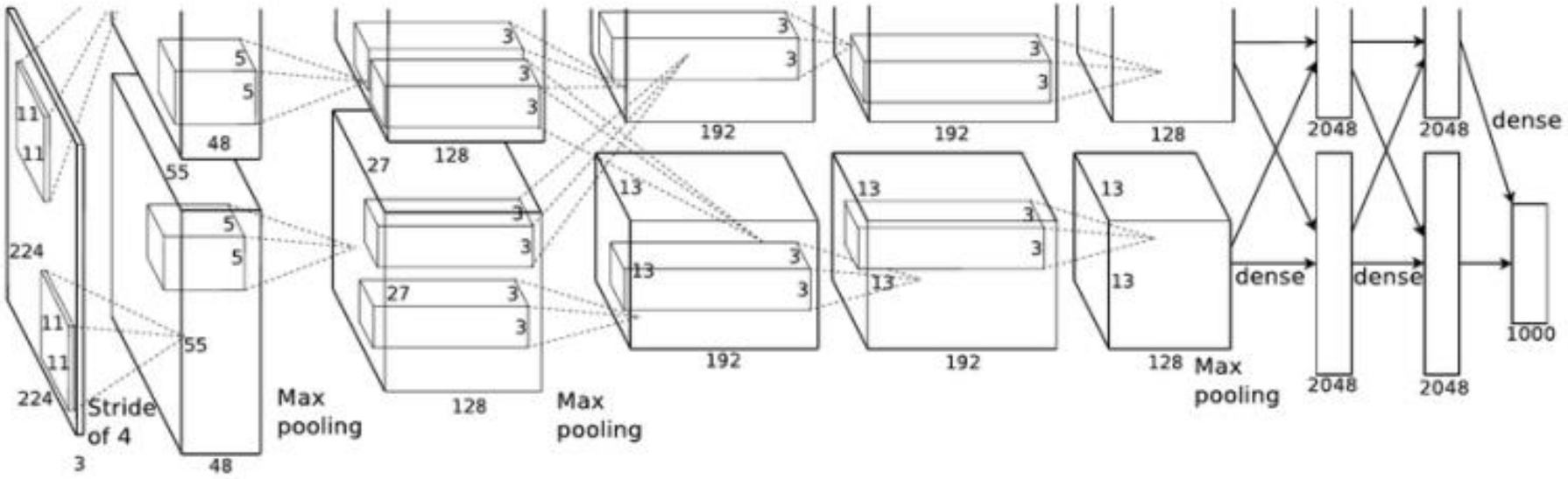
AlexNet Architecture

- ImageNet Classification with Deep Convolutional Neural Networks



https://crowdai-shared.s3.eu-central-1.amazonaws.com/markdown_editor/594cad579bcc68ef57e00ac7850a65e5_deep-learning-with-python-pydata-seattle-2015-35-638.jpg

AlexNet Architecture



- 8 layers: 5 convolutional layers + 3 fully-connected layers
- ReLU activation function to add nonlinearity & improve convergence rate
- Multiple GPUs for faster training
- 60 million parameters - susceptible to overfitting
- Data augmentation and dropouts to avoid overfitting
- Overlapping pooling to reduce the error.

AlexNet Architecture

Convolution and Maxpooling Layers

Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
Input	-	-	-	-	227 x 227 x 3	-
Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-

AlexNet Architecture

Convolution and Maxpooling Layers

Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
Input	-	-	-	-	227 x 227 x 3	-
Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-

- Width and height of output = (227-11)/4 +1 = 55
- Number of parameters in the first layer
  - $11 \times 11 \times 3 \times 96 = 34848$

AlexNet Architecture

Convolution and Maxpooling Layers

Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
Input	-	-	-	-	227 x 227 x 3	-
Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-

- Next layer: Overlapping Max Pooling

AlexNet Architecture

Convolution and Maxpooling Layers

Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
Input	-	-	-	-	227 x 227 x 3	-
Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-

- 2nd convolution operation
- Max-pooling layer

AlexNet Architecture

Convolution and Maxpooling Layers

Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
Input	-	-	-	-	227 x 227 x 3	-
Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-

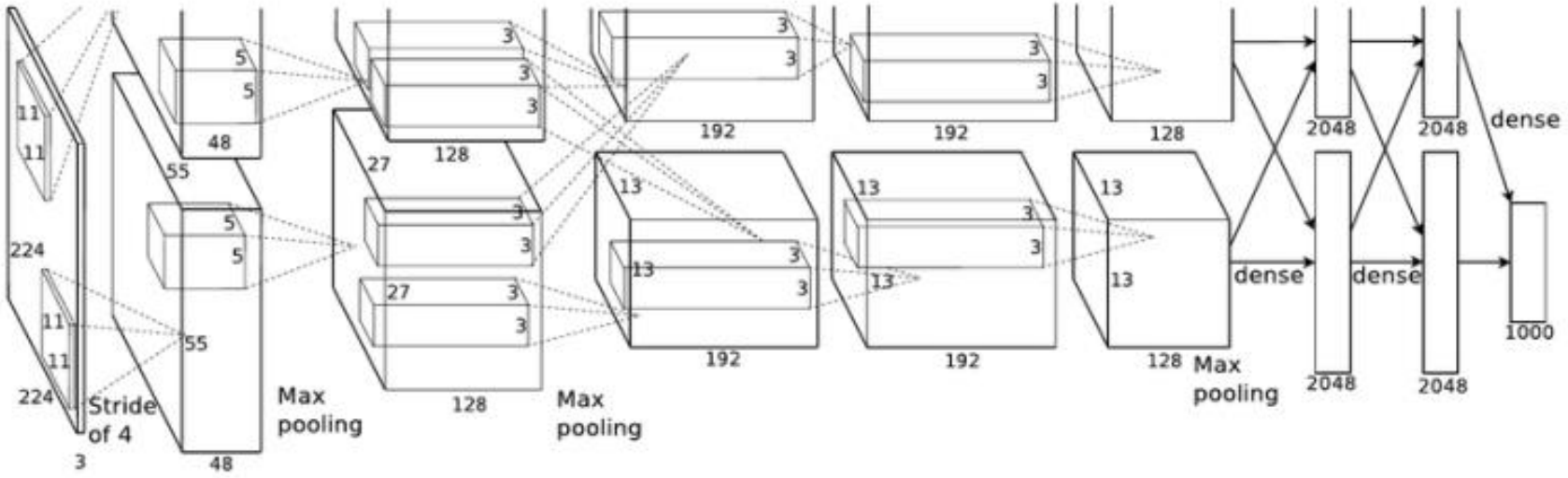
AlexNet Architecture

Fully Connected and Dropout Layers

Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
-	-	-	-	-	-	-
-	-	-	-	-	-	-
-	-	-	-	-	-	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-
Fully Connected 1	-	-	-	-	4096	ReLU
Dropout 2	rate = 0.5	-	-	-	4096	-
Fully Connected 2	-	-	-	-	4096	ReLU
Fully Connected 3	-	-	-	-	1000	Softmax

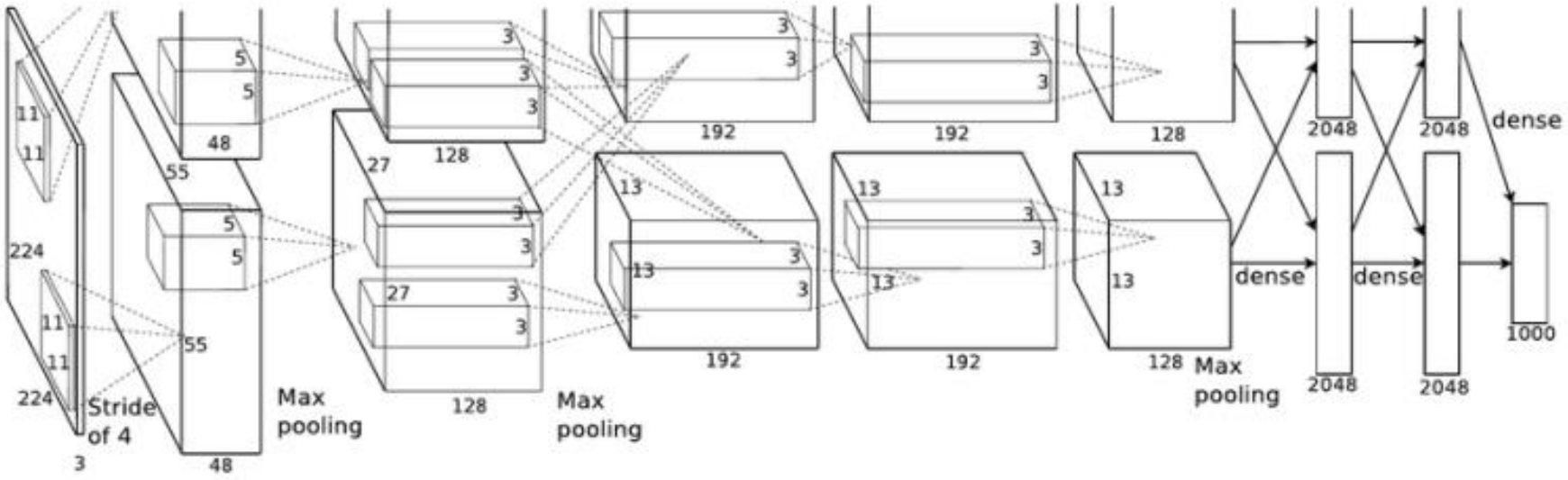
- Last fully connected layer or output layer with 1000 neurons as we have 1000 classes in the data set.
- Parameters: Dropout rate 0.5, Batch size = 128, Weight decay term: 0.0005, Momentum term $\alpha = 0.9$, learning rate $\eta = 0.01$, manually reduced by factor of ten on monitoring validation loss.

AlexNet Architecture



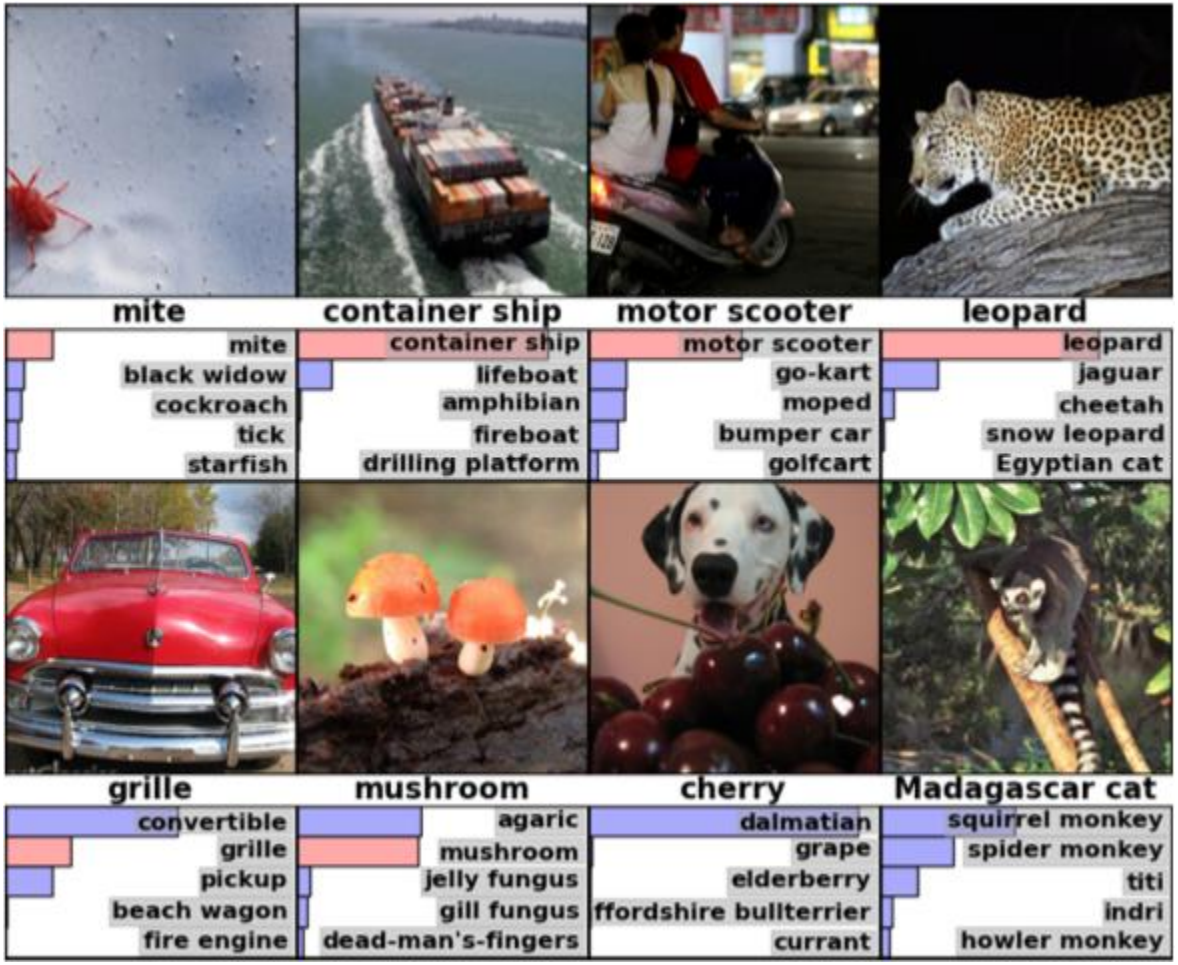
- Popularized the use of ReLUs
- Used heavy data augmentation (flipped images, random crops of size 227 × 227, color normalization, etc.)
- 62.3 million learnable parameters

AlexNet Architecture



- AlexNet was trained on a GTX 580 GPU with only 3 GB of memory which couldn't fit the entire network.
- The network was split across 2 GPUs, with half of the neurons (feature maps) on each GPU.
- See a split in the architecture diagram.

AlexNet Results



https://miro.medium.com/max/733/1*cQwk1chsEgnSlyILwenqsw.png

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

Classification: Top-5 error

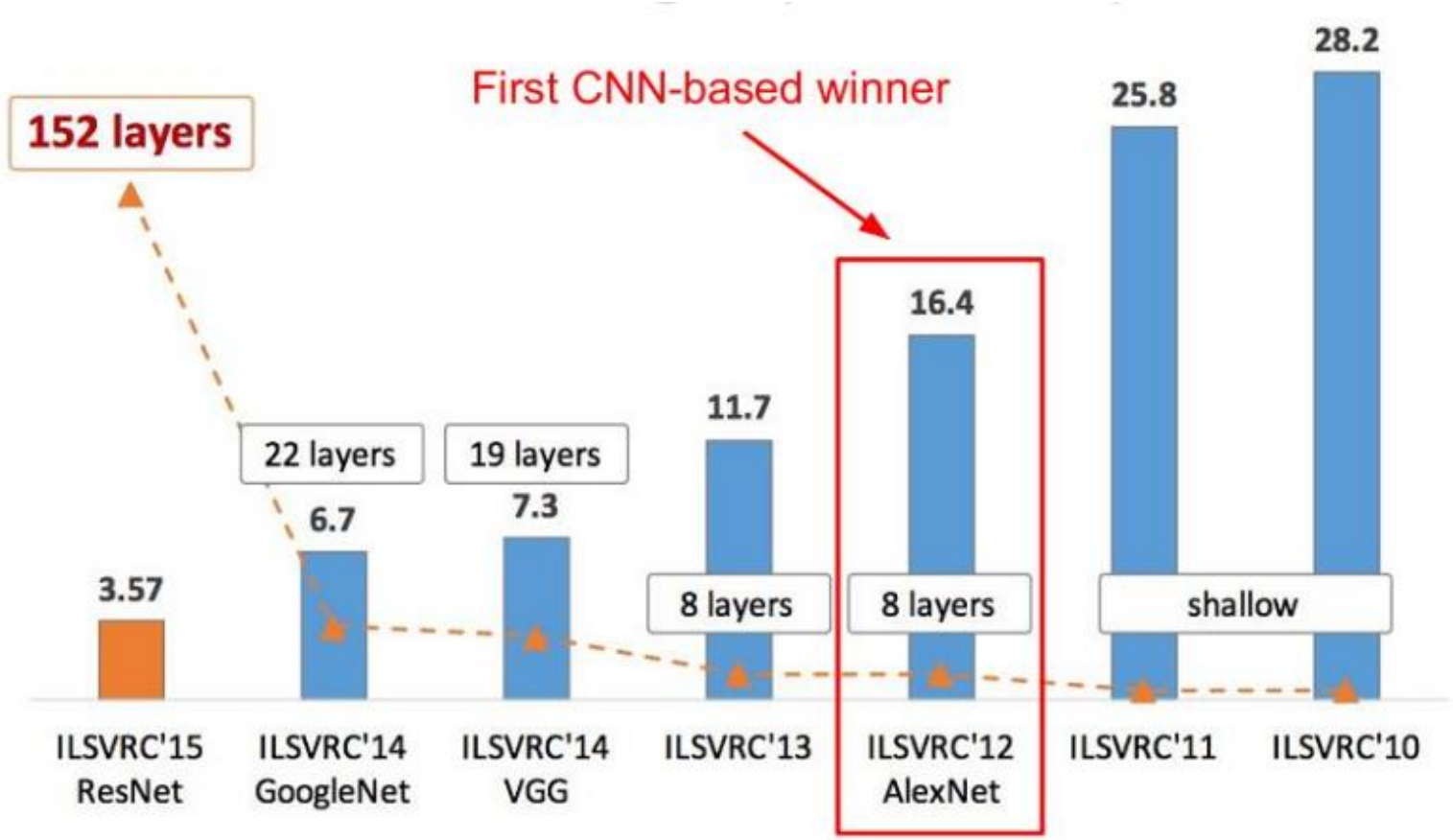
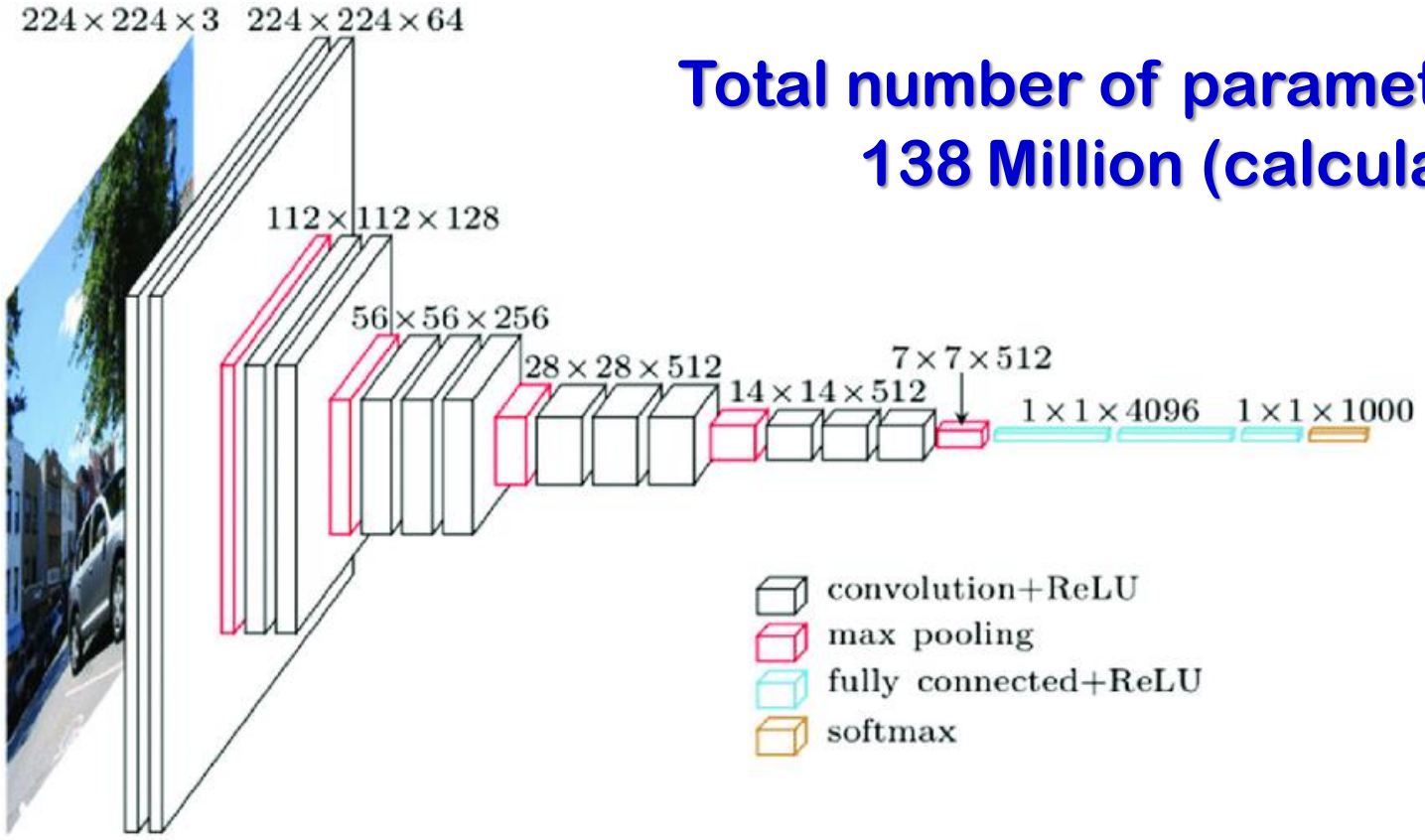


Figure: Kaiming He, MSR

VGG

- K. Simonyan and A. Zisserman (2014) “Very Deep Convolutional Networks for Large-Scale Image Recognition”

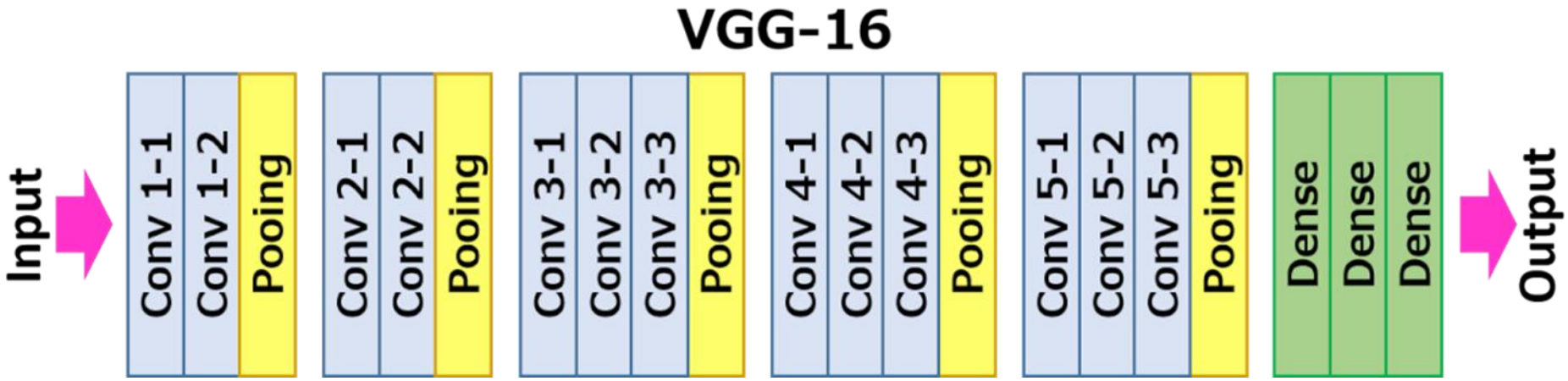


Total number of parameters:
138 Million (calculate!)

https://www.researchgate.net/figure/An-overview-of-the-VGG-16-model-architecture-this-model-uses-simple-convolutional-blocks_fig2_328966158

VGG

- K. Simonyan and A. Zisserman (2014) “Very Deep Convolutional Networks for Large-Scale Image Recognition”



Reduce the size of the convolution kernel and increase the number of convolution layers

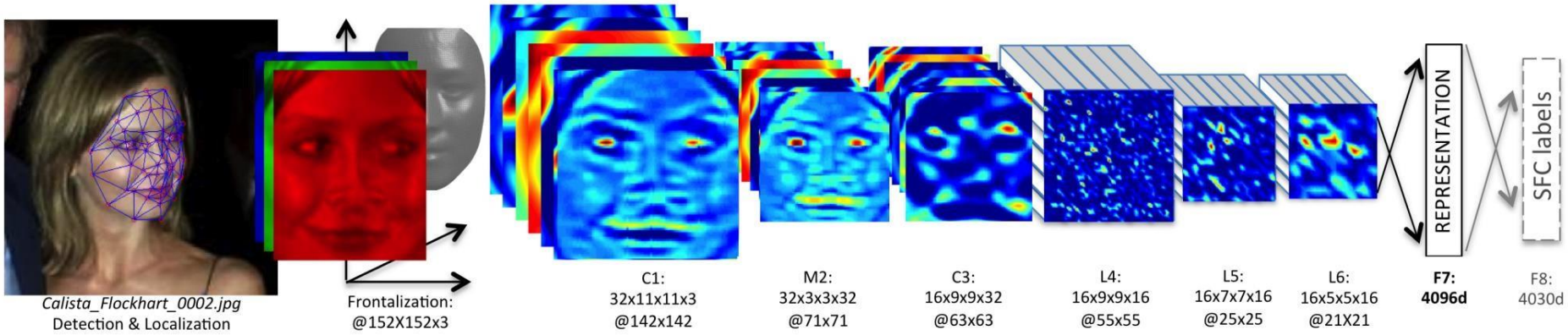
https://www.researchgate.net/figure/An-overview-of-the-VGG-16-model-architecture-this-model-uses-simple-convolutional-blocks_fig2_328966158

VGG Config-urations

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Industry Deployment

- Used in Facebook, Google, Microsoft
- Image Recognition, Speech Recognition, ….
- Fast at test time



Taigman et al. DeepFace: Closing the Gap to Human-Level Performance in Face Verification, CVPR'14

Slide: R. Fergus

References

- Ian Goodfellow, Deep Learning, MIT Press, 2016
- Andrew Ng, Coursera Course on CNN
- Lecture slides of Prof. Aparajita Ojha, IIITDM Jabalpur (CS617 Basics of Deep Learning)
- Lecture slides of Prof. Ming Li Lecture slides, University of Waterloo, Canada (CS 898: Deep Learning and Its Applications, Spring 2017).
- Lecture slides of Prof. Fei-Fei Li, Justin Johnson, Serena Yeung,Stanford University, (CS231n Convolution Neural Networks for Visual recognition).
- Lecture slides of Prof. Ioannis Gkioulekas, Carnegie Mellon University (16-385 Computer Vision, Spring 2019).