DSE 3151 DEEP LEARNING

Convolutional Neural Networks

Dr. Rohini Rao & Dr. Abhilash K Pai

Dept. of Data Science and Computer Applications

MIT Manipal

The Convolution Operation - 1D

- Convolution is a linear operation on two functions of a real-valued argument, where one function is applied over the other to yield element-wise dot products.

- Example: Consider a discrete signal 'x_t' which represents the position of a spaceship at time 't' recorded by a laser sensor.

x_0

- Now, suppose that this sensor is noisy.

- To obtain a less noisy measurement we would like to average several measurements.

x_1

- Considering that, the most recent measurements are more important, we would like to take a weighted average over 'x_t'. The new estimate at time 't' is computed as follows:

$$s_t = \sum_{a=0}^{\infty} x_{t-a} w_{-a} = (\underbrace{x}_{\text{input}} \overset{\text{convolution}}{*} \underbrace{w}_{\text{Filter/Mask/Kernel}})_t$$

x_2

The Convolution Operation - 1D

- In practice, we would sum only over a small window.

For example: $$s_t = \sum_{a=0}^{6} x_{t-a} w_{-a}$$

- We just slide the filter over the input and compute the value of s_t based on a window around x_t

	w_{-6}	w_{-5}	w_{-4}	w_{-3}	w_{-2}	w_{-1}	w_0		
w	0.01	0.01	0.02	0.02	0.04	0.4	0.5		
	*	*	*	*	*	*	*		
x	1.0	1.10	1.20	1.40	1.70	1.80	1.90	2.10	2.20
s							1.80		

Content adapted from : CS7015 Deep Learning, Dept. of CSE, IIT Madras

The Convolution Operation - 1D

- In practice, we would sum only over a small window.

For example: $$s_t = \sum_{a=0}^{6} x_{t-a} w_{-a}$$

- We just slide the filter over the input and compute the value of s_t based on a window around x_t

	w_{-6}	w_{-5}	w_{-4}	w_{-3}	w_{-2}	w_{-1}	w_0
w	0.01	0.01	0.02	0.02	0.04	0.4	0.5
	*	*	*	*	*	*	*

x	1.0	1.10	1.20	1.40	1.70	1.80	1.90	2.10	2.20
s							1.80	1.96	

Content adapted from : CS7015 Deep Learning, Dept. of CSE, IIT Madras

The Convolution Operation - 1D

- In practice, we would sum only over a small window.

For example: $$s_t = \sum_{a=0}^{6} x_{t-a} w_{-a}$$

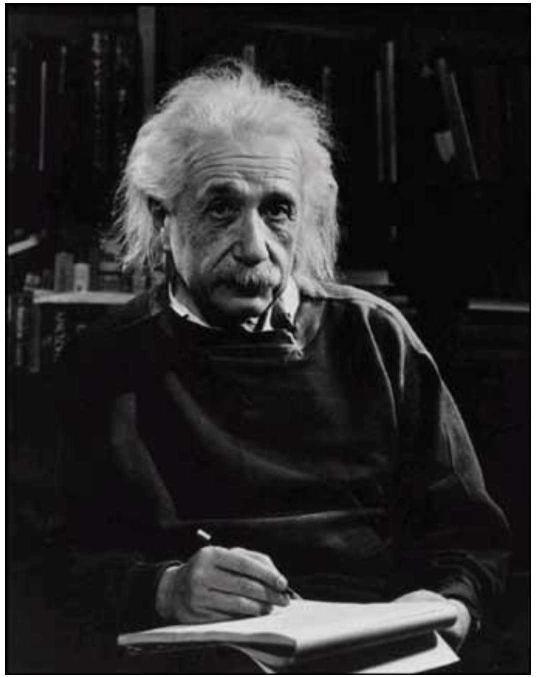
- We just slide the filter over the input and compute the value of s_t based on a window around x_t

			w_{-6}	w_{-5}	w_{-4}	w_{-3}	w_{-2}	w_{-1}	w_0
w			0.01	0.01	0.02	0.02	0.04	0.4	0.5
			*	*	*	*	*	*	*
x	1.0	1.10	1.20	1.40	1.70	1.80	1.90	2.10	2.20
s							1.80	1.96	2.11

- Use cases of 1-D convolution : Audio signal processing, stock market analysis, time series analysis etc.

Content adapted from : CS7015 Deep Learning, Dept. of CSE, IIT Madras

Convolution in 2-D using Images : What is an Image?



What we see

0	3	2	5	4	7	6	9	8
3	0	1	2	3	4	5	6	7
2	1	0	3	2	5	4	7	6
5	2	3	0	1	2	3	4	5
4	3	2	1	0	3	2	5	4
7	4	5	2	3	0	1	2	3
6	5	4	3	2	1	0	3	2
9	6	7	4	5	2	3	0	1
8	7	6	5	4	3	2	1	0

What a computer sees

Convolution in 2-D using Images : What is an Image?

- An image can be represented mathematically as a function $f(x,y)$ which gives the intensity value at position (x,y), where, $f(x,y)$ ε $\{0,1,....,I_{max-1}\}$ and x,y ε $\{0,1,.....,N-1\}$.

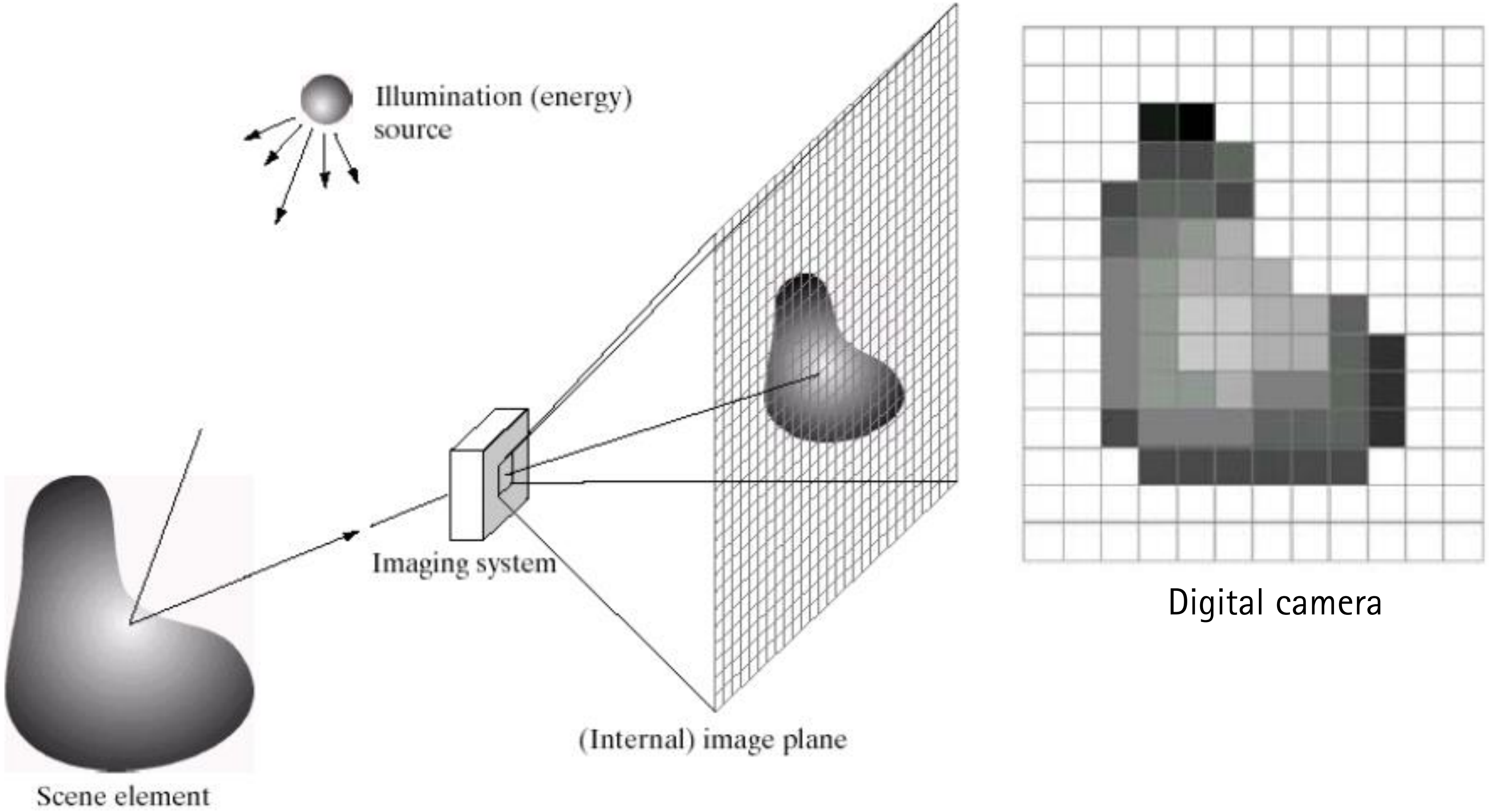
- Larger the value of N, more is the clarity of the picture (larger resolution), but more data to be analyzed in the image.

- If the image is a Gray-scale (8-bit per pixel) image, then it requires N^2 Bytes for storage.

- If the image is color - RGB, each pixel requires 3 Bytes of storage space.

N is the resolution of the image and I_{max} is the level of discretized brightness value.

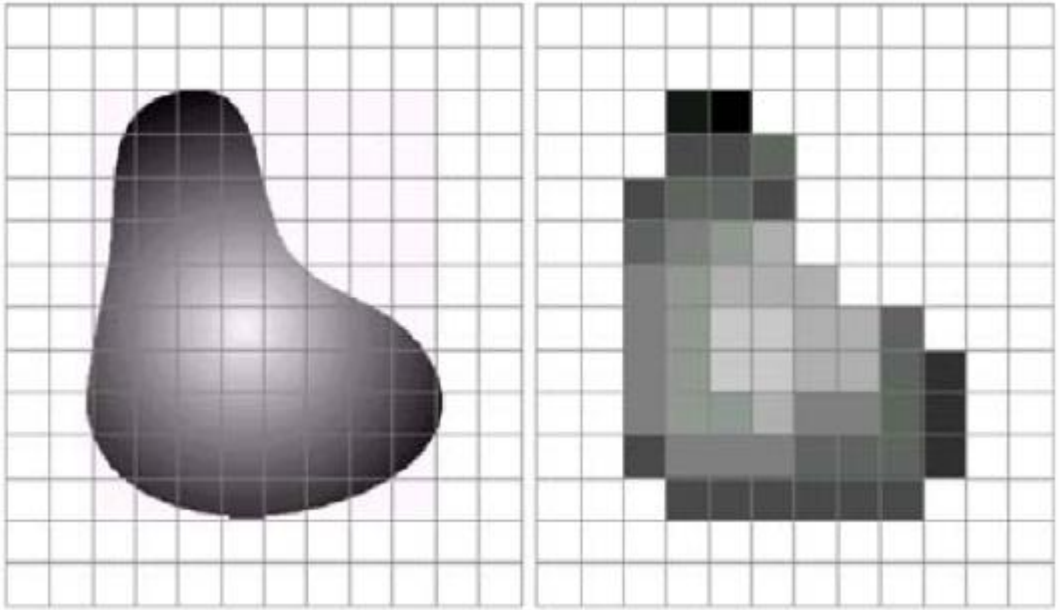
Convolution in 2-D using Images : What is an Image?



[Source: D. Hoiem]

Convolution in 2-D using Images : What is an Image?

- **Sample** the 2-D space on a regular grid.
- **Quantize** each sample, i.e., the photons arriving at each active cell are integrated and then digitized.



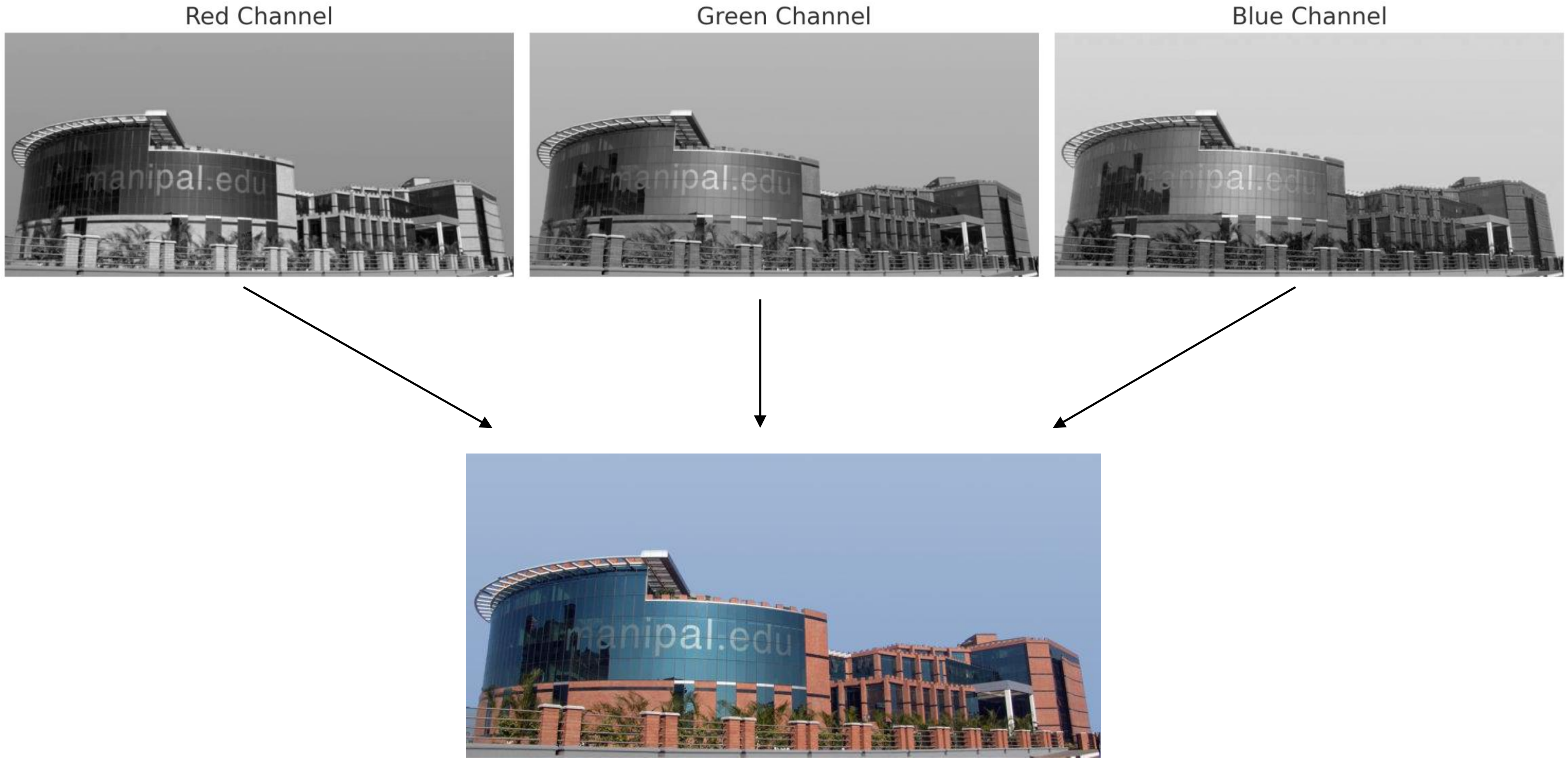
[Source: D. Hoiem]

Convolution in 2-D using Images : What is an Image?

- A grid (matrix) of intensity values.

255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	0	0	255	255	255	255	255	255	255
255	255	255	75	75	75	255	255	255	255	255	255
255	255	75	95	95	75	255	255	255	255	255	255
255	255	96	127	145	175	255	255	255	255	255	255
255	255	127	145	175	175	175	255	95	255	255	255
255	255	127	145	200	200	175	175	95	255	255	255
255	255	127	145	145	175	127	127	95	47	255	255
255	255	127	145	145	175	127	127	95	47	255	255
255	255	74	127	127	127	95	95	95	47	255	255
255	255	255	74	74	74	74	74	74	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255

Convolution in 2-D using Images : What is an Image?



The Convolution Operation - 2D

- Images are good examples of 2-D inputs.

- A 2-D convolution of an Image 'I' using a filter 'K' of size 'm x n' is now defined as (looking at previous pixels):

$$S_{ij} = (I * K)_{ij} = \sum_{a=0}^{m-1} \sum_{b=0}^{n-1} I_{i-a,j-b} \; K_{a,b}$$

- In practice, one of the way is to look at the succeeding pixels:

$$S_{ij} = (I * K)_{ij} = \sum_{a=0}^{m-1} \sum_{b=0}^{n-1} I_{i+a,j+b} \; K_{a,b}$$

- Another way is to consider center pixel as reference pixel, and then look at its surrounding pixels:

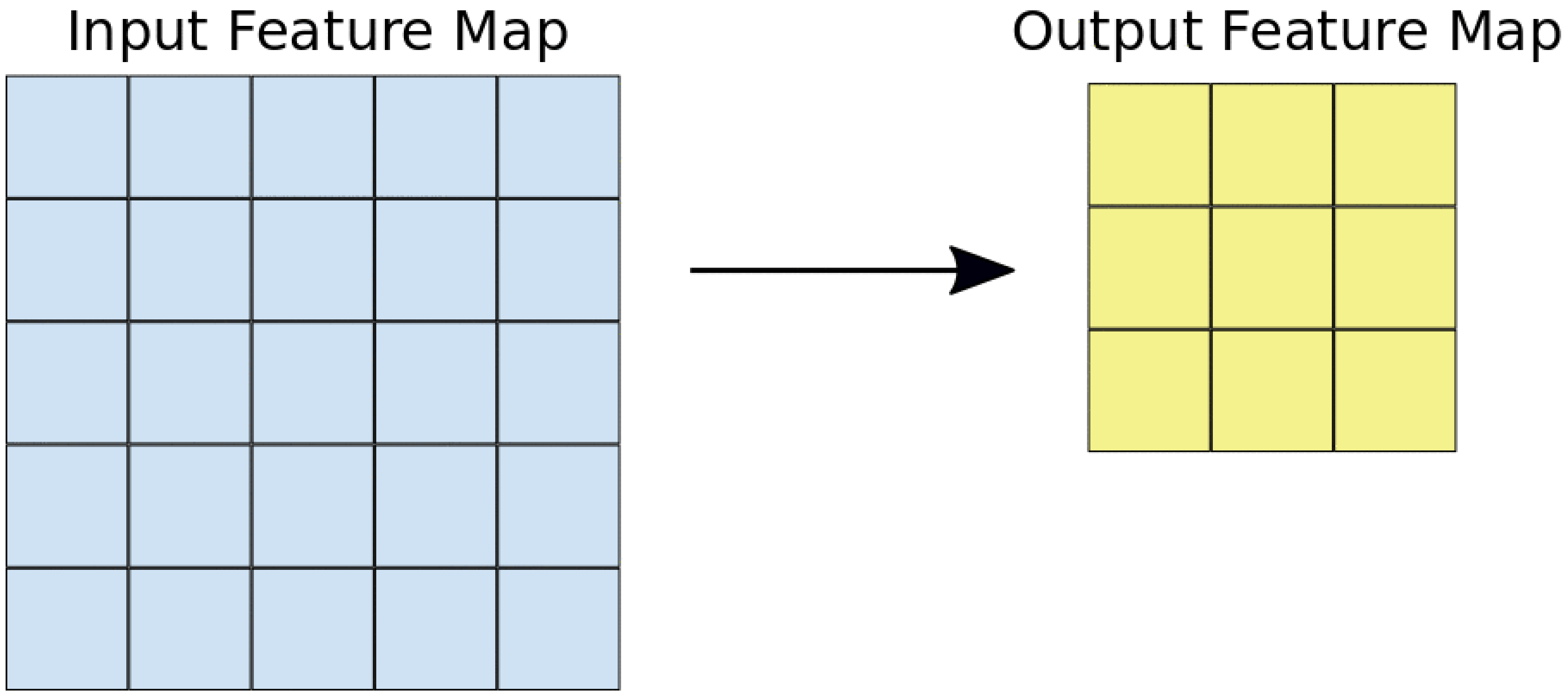
$$S_{ij} = (I * K)_{ij} = \sum_{a=\lfloor -m/2 \rfloor}^{\lfloor m/2 \rfloor} \sum_{b=\lfloor -n/2 \rfloor}^{\lfloor n/2 \rfloor} I_{i-a,j-b} * K_{(m/2)+a,\ (n/2)+b}$$

Pixel of interest

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

Content adapted from : CS7015 Deep Learning, Dept. of CSE, IIT Madras

The Convolution Operation - 2D



Source: https://developers.google.com/

The Convolution Operation - 2D

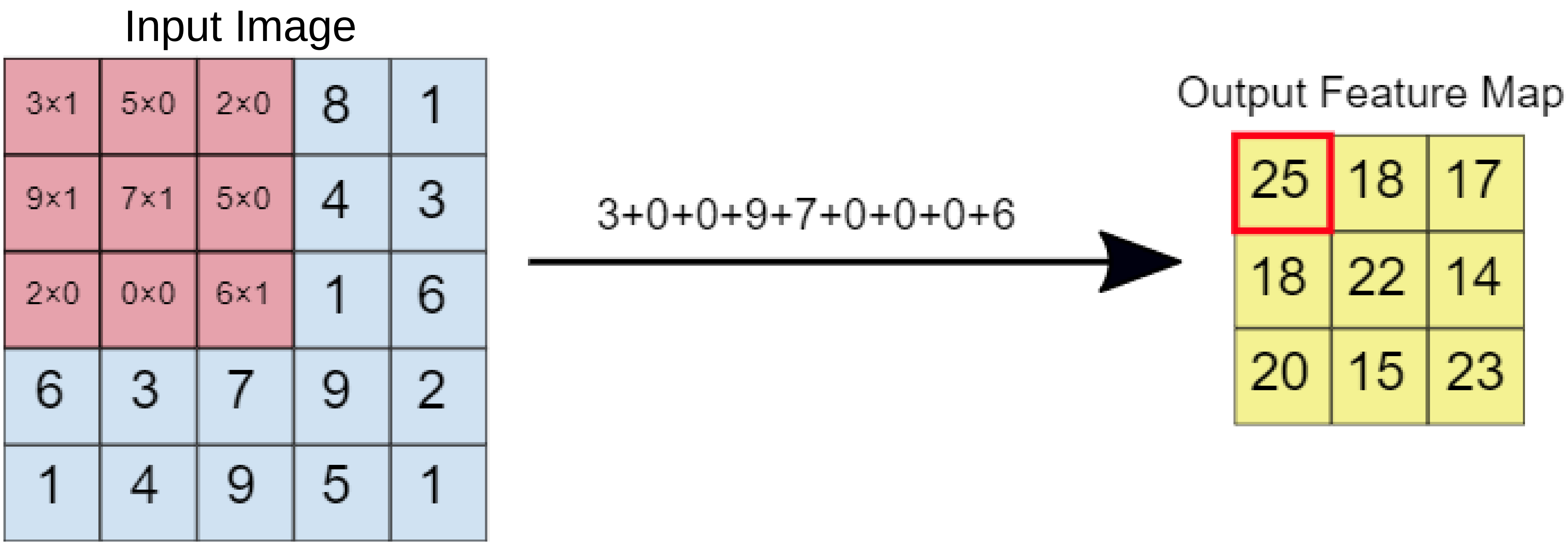
Input Image

3	5	2	8	1
9	7	5	4	3
2	0	6	1	6
6	3	7	9	2
1	4	9	5	1

Convolutional Filter

1	0	0
1	1	0
0	0	1

Source: https://developers.google.com/



Source: https://developers.google.com/

The Convolution Operation - 2D

$$* \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix} =$$

Smoothening Filter

The Convolution Operation - 2D

$$* \begin{array}{ccc} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{array} =$$

Sharpening Filter

The Convolution Operation - 2D

$$* \quad \begin{matrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{matrix} \quad =$$

Filter for edge detection

The Convolution Operation – 2D : Various filters (edge detection)

Prewitt

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

S_x

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

S_y

Sobel

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

S_x

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

S_y

Laplacian

0	1	0
1	-4	1
0	1	0

Roberts

0	1
-1	0

S_x

1	0
0	-1

S_y

Input image

After applying Horizontal edge detection filter

After applying Vertical edge detection filter

The Convolution Operation - 2D

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

stride=1

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

Input image: 6 x 6

Dot product → 3 -1

Note: Stride is the number of "unit" the kernel is shifted per slide over rows/ columns

The Convolution Operation - 2D

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

If stride=2

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

Input image: 6 x 6

3 -3

Note: Stride is the number of "unit" the kernel is shifted per slide over rows/ columns

The Convolution Operation - 2D

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

stride=1

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

Input image: 6 x 6

3	-1	-3	-1
-3	1	0	-3
-3	-3	0	1
3	-2	-2	-1

4 x 4 Feature Map

The Convolution Operation - 2D

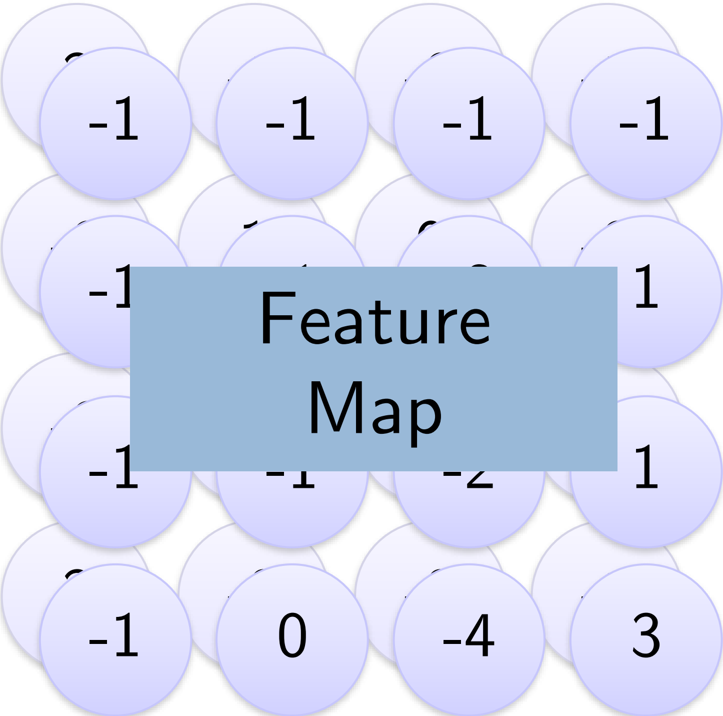
-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

stride=1

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

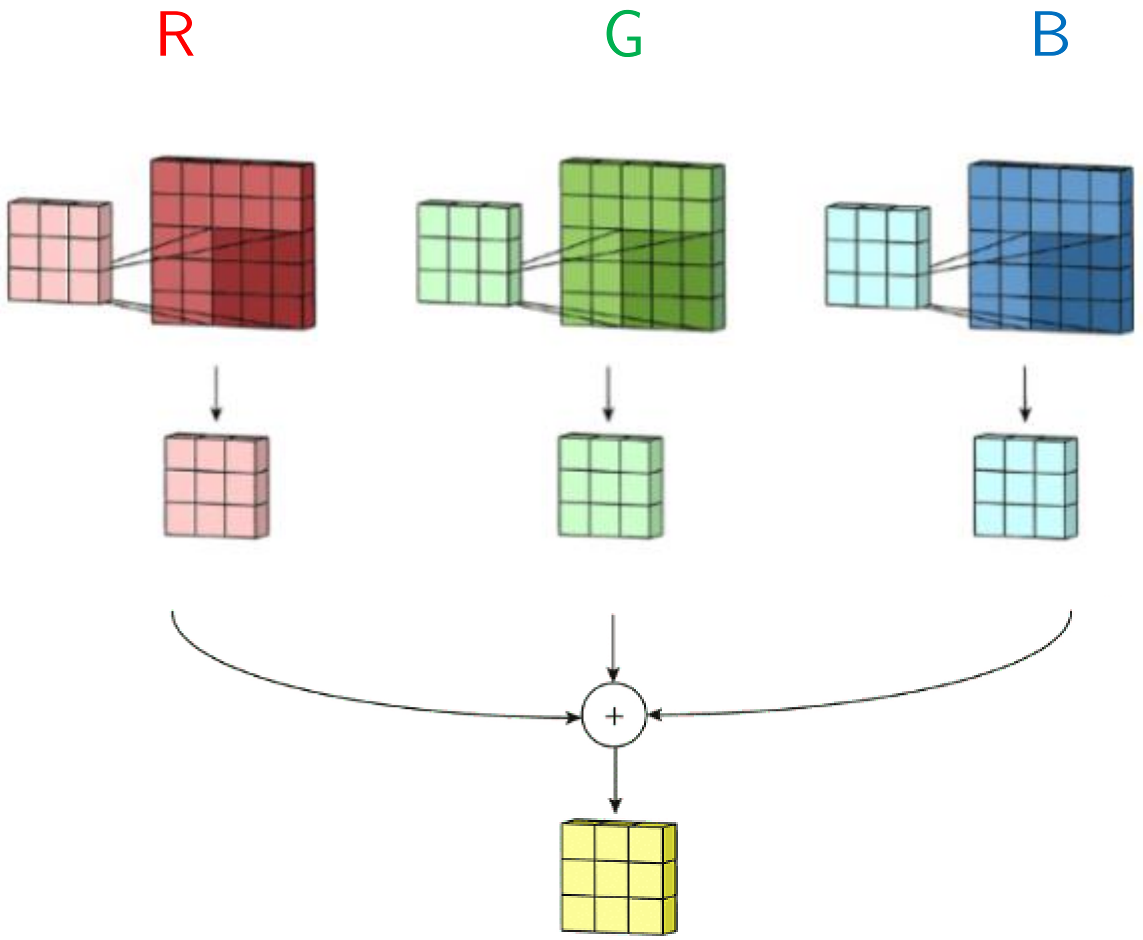
Input image: 6 x 6



Repeat for each filter!

Two 4 x 4 images
Forming 4 x 4 x 2 matrix

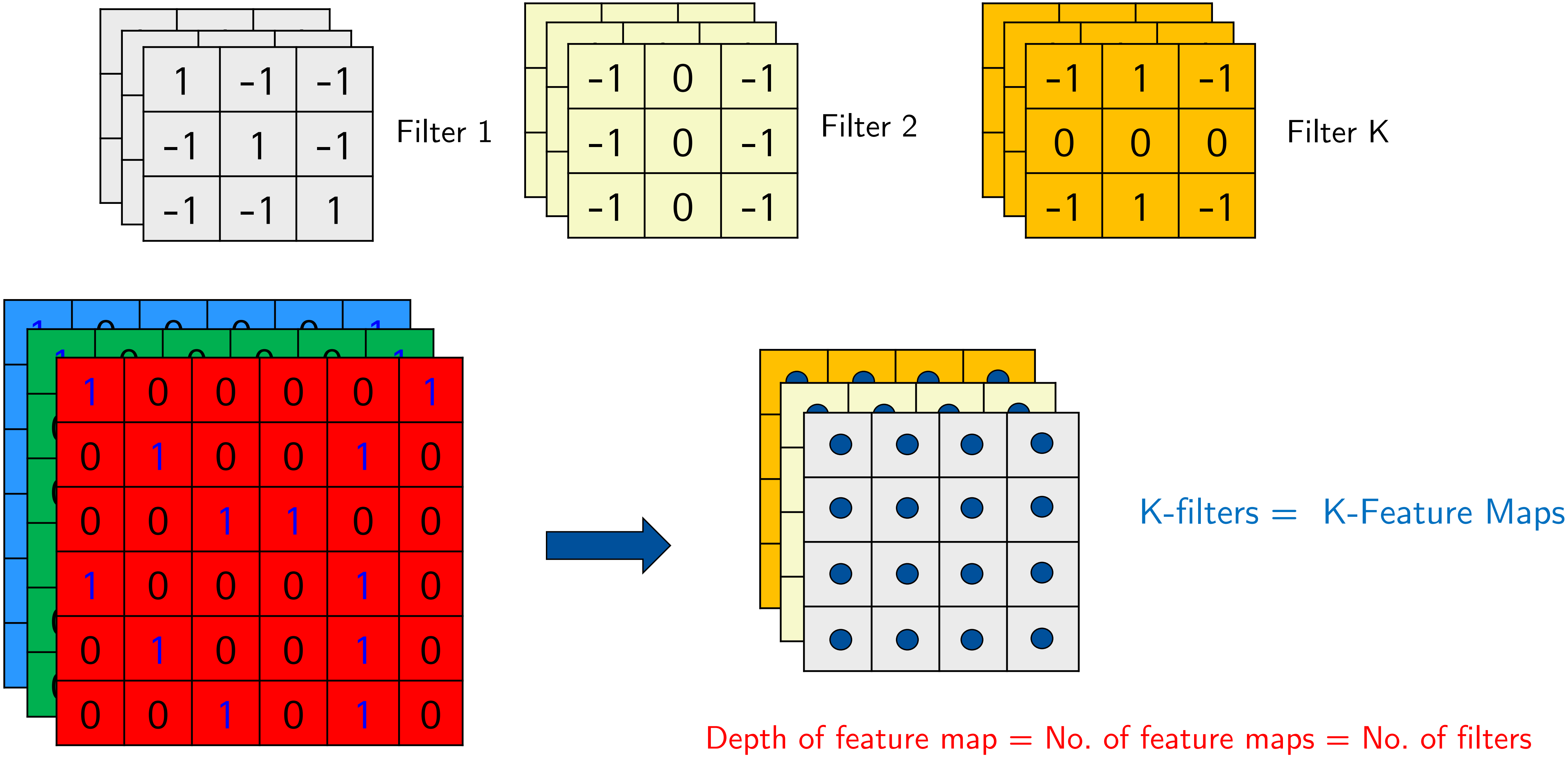
The Convolution Operation –RGB Images



Apply the filter to R, G, and B channels of the image and combine the resultant feature maps to obtain a 2-D feature map.

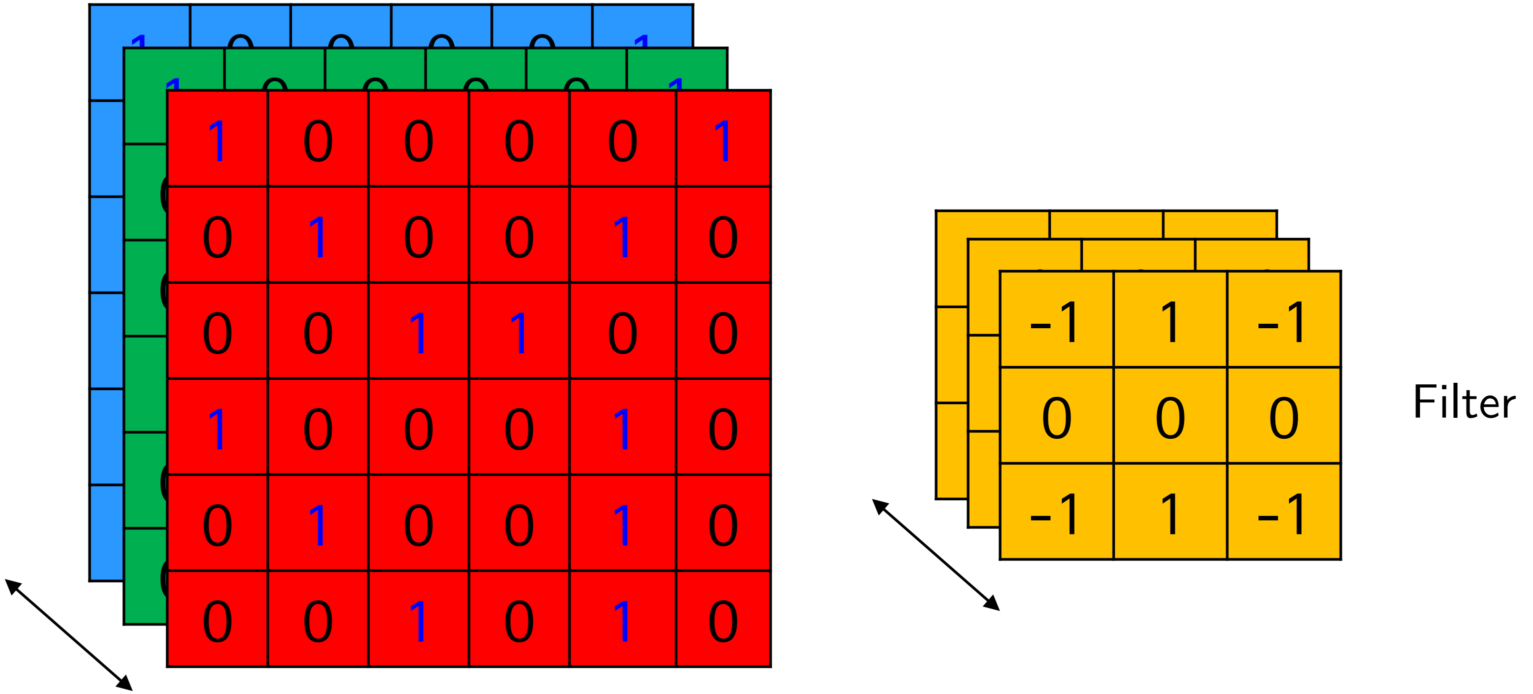
Source: Intuitively Understanding Convolutions for Deep Learning | by Irhum Shafkat | Towards Data Science

The Convolution Operation –RGB Images multiple filters

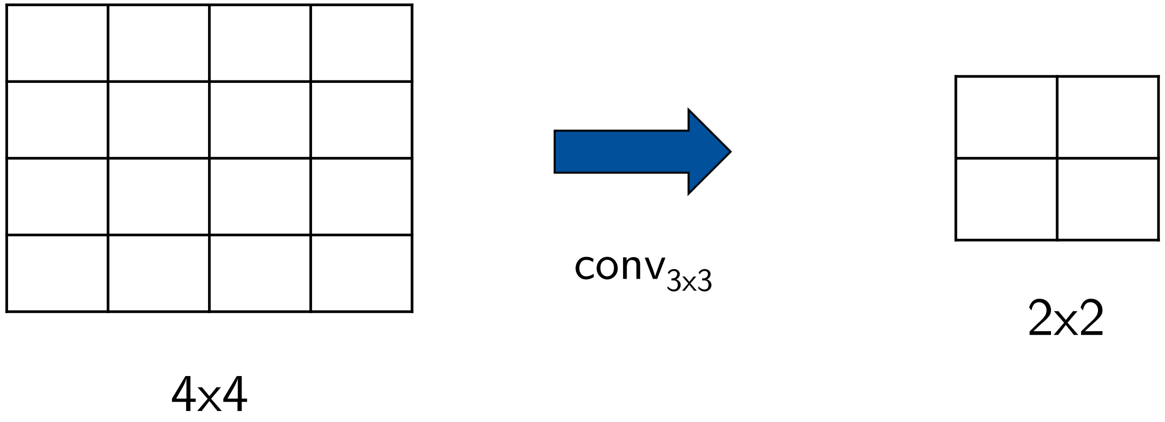


Depth of feature map = No. of feature maps = No. of filters

The Convolution Operation : Terminologies



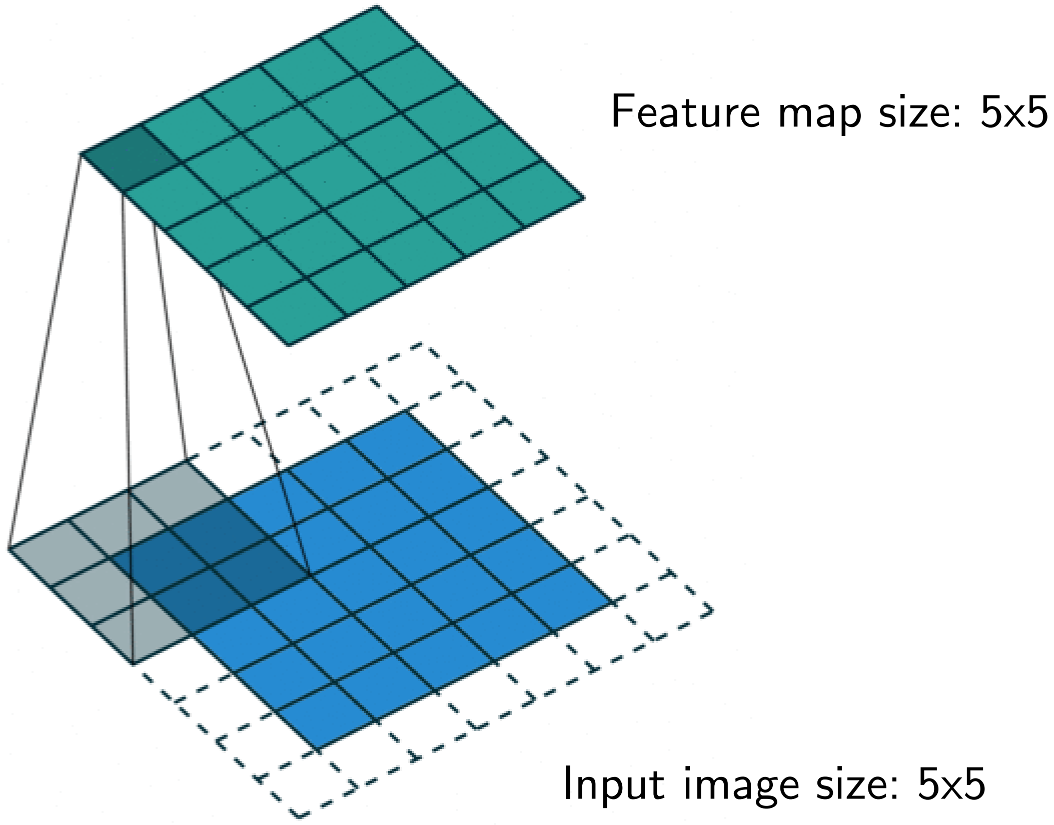
1. Depth of an Input Image = No. of channels in the Input Image = Depth of a filter
2. Assuming square filters, Spatial Extent (F) of a filter is the size of the filter



0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

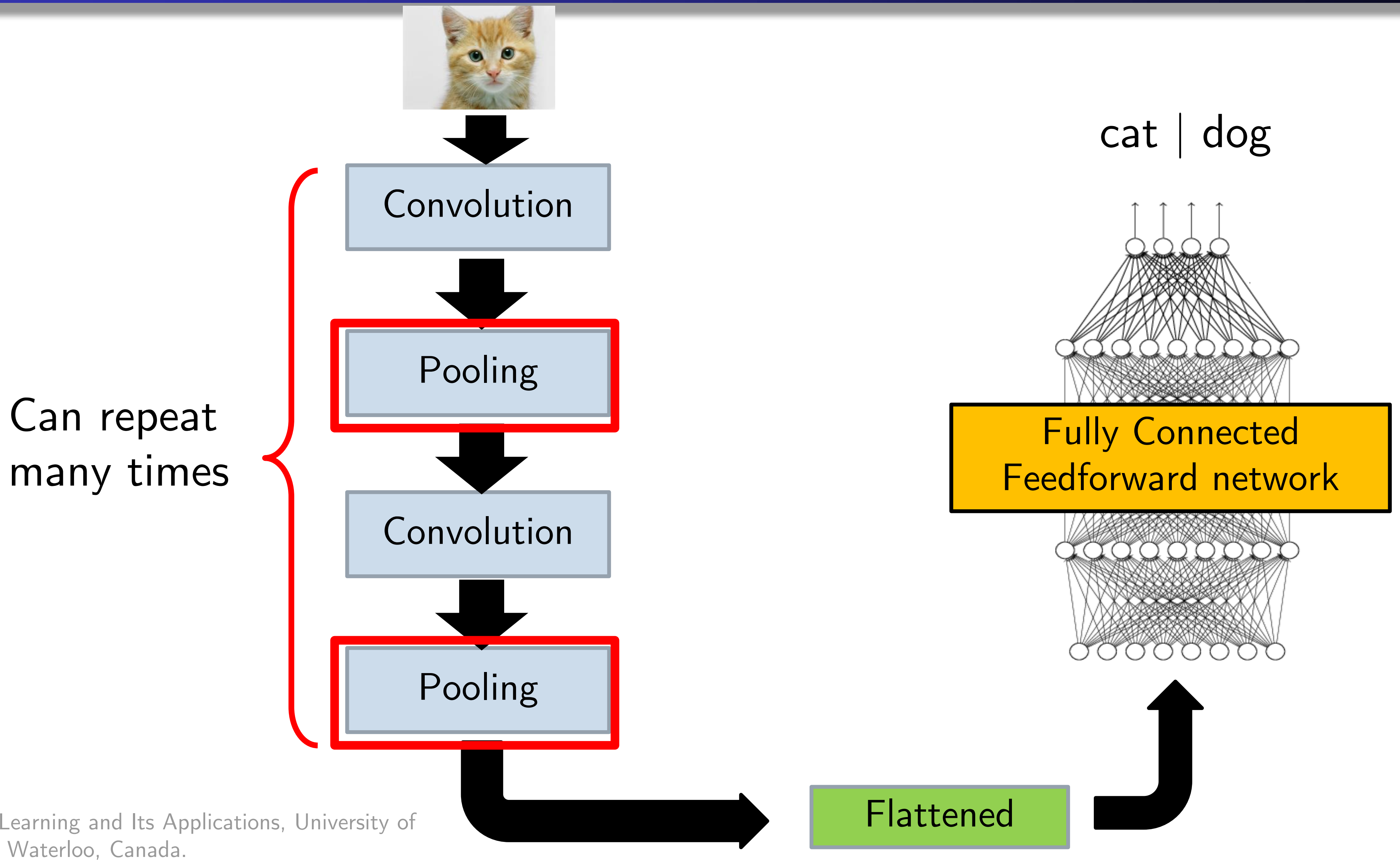
Pad Zeros and then convolve to obtain a feature map with dimension = input image dimension

The Convolution Operation : Zero Padding



Source: Intuitively Understanding Convolutions for Deep Learning | by Irhum Shafkat | Towards Data Science

Convolutional Neural Network (CNN) : At a glance



Source: CS 898: Deep Learning and Its Applications, University of Waterloo, Canada.

Pooling

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

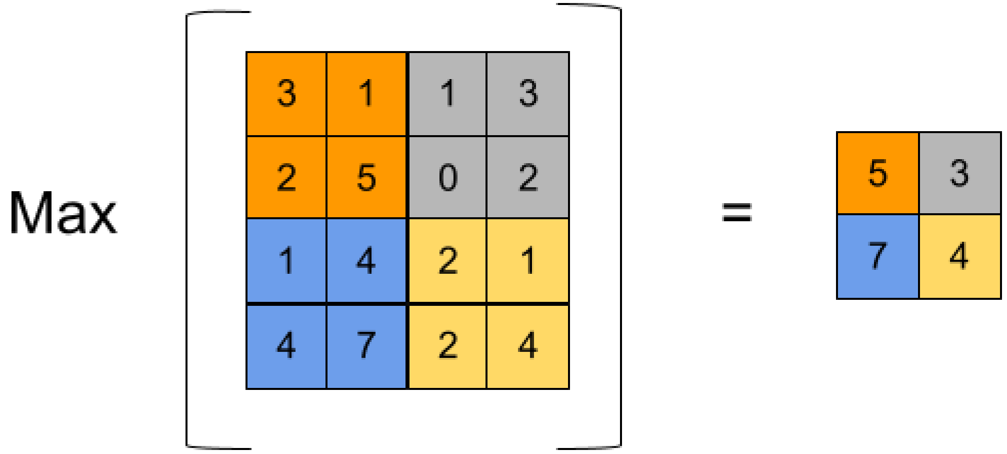
3	-1	-3	-1
-3	1	0	-3
-3	-3	0	1
3	-2	-2	-1

-1	-1	-1	-1
-1	-1	-2	1
-1	-1	-2	1
-1	0	-4	3

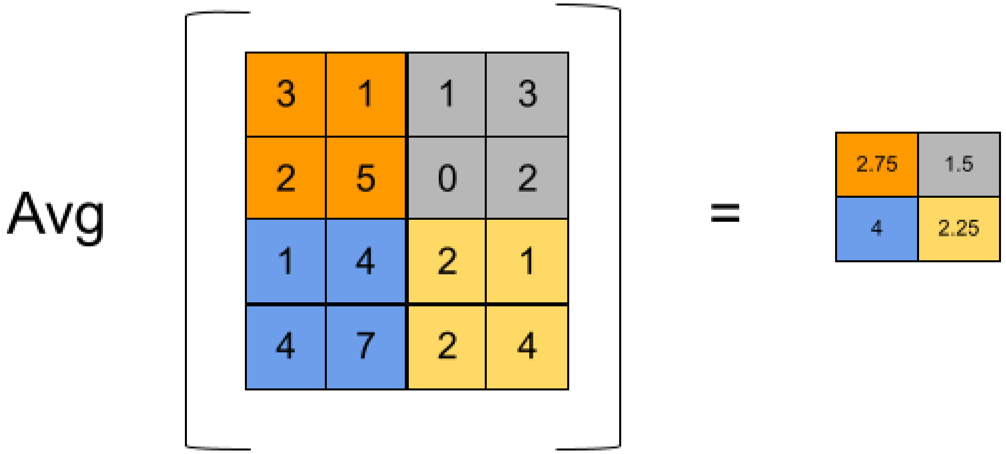
- Max Pooling
- Average Pooling

Source: CS 898: Deep Learning and Its Applications, University of Waterloo, Canada.

Pooling



Max. Pooling



Average Pooling

Stride ?

Why Pooling ?

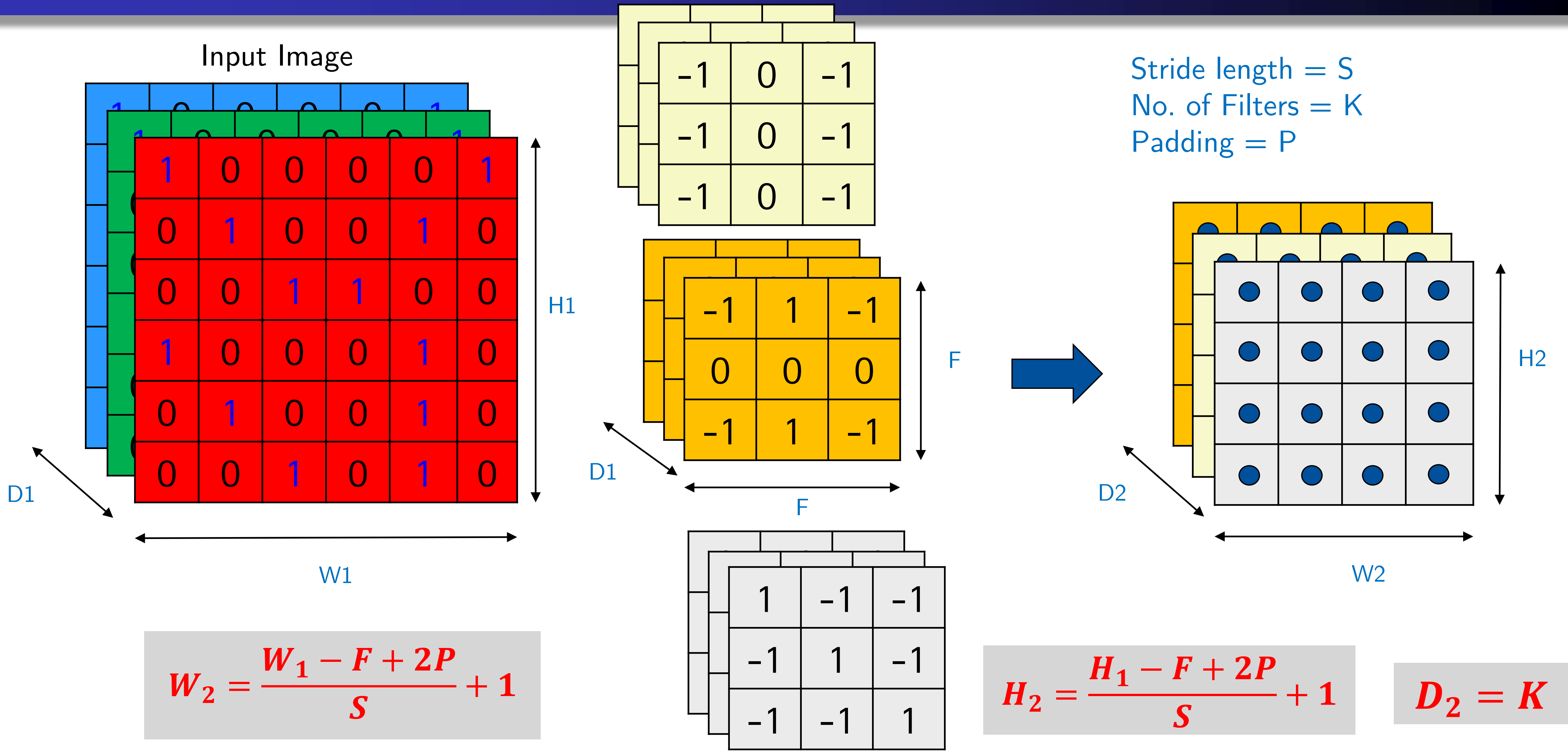
- Subsampling pixels will not change the object

- We can subsample the pixels to make image smaller

- Therefore, fewer parameters to characterize the image

Source: CS 898: Deep Learning and Its Applications, University of Waterloo, Canada.

Relation between i/p size, feature map size, filter size



Stride length = S
No. of Filters = K
Padding = P

$$W_2 = \frac{W_1 - F + 2P}{S} + 1$$

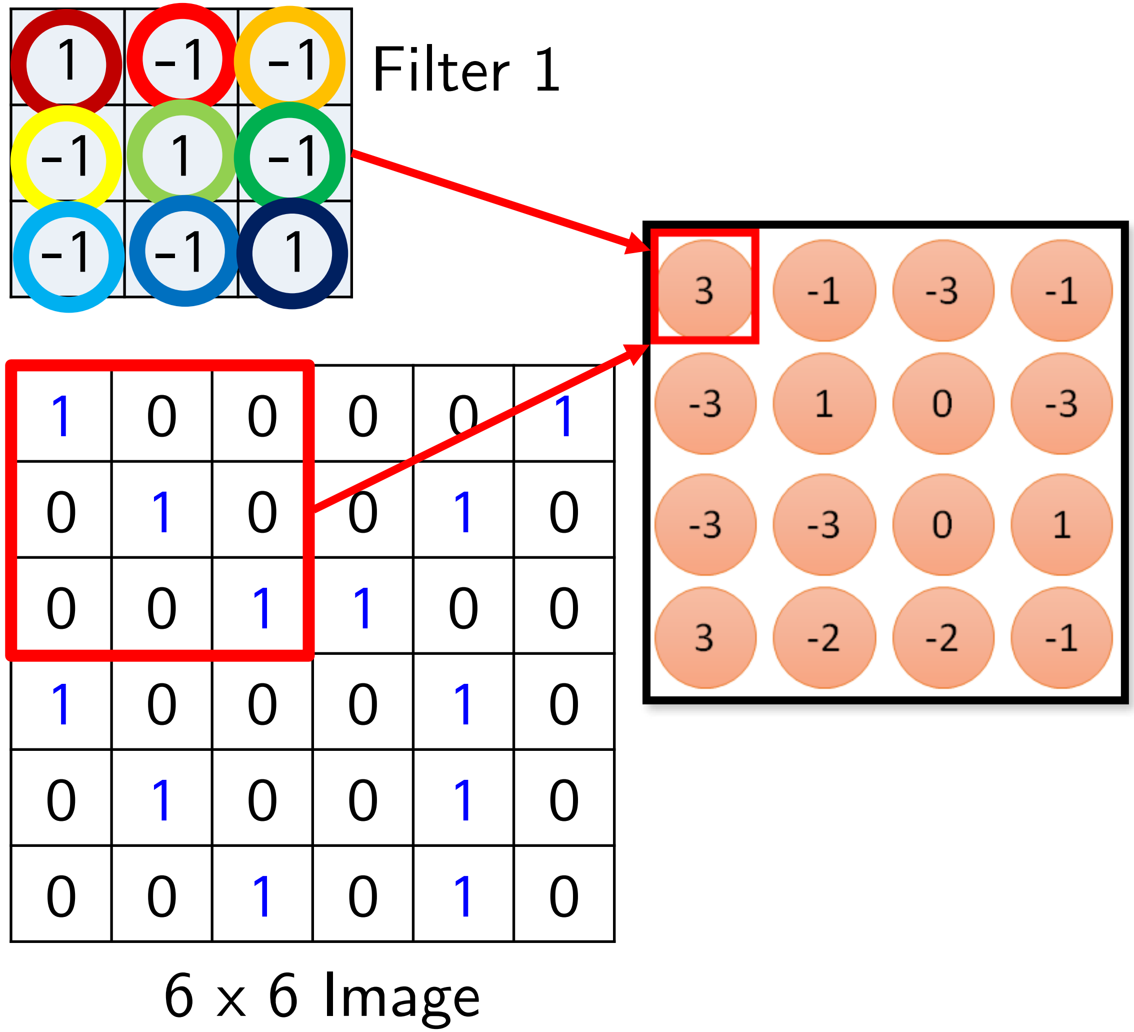
$$H_2 = \frac{H_1 - F + 2P}{S} + 1$$

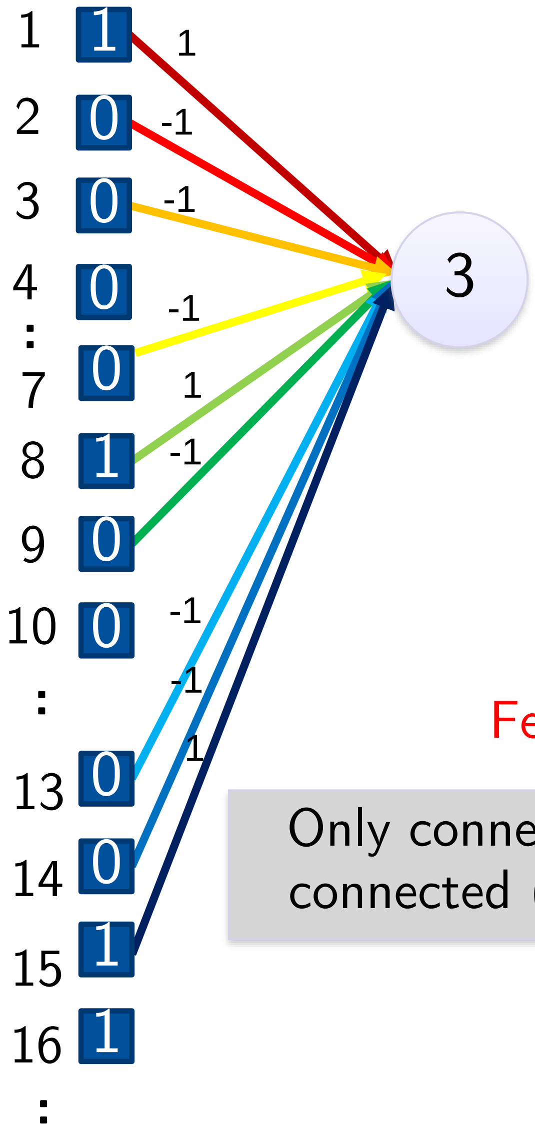
$$D_2 = K$$

Important properties of CNN

- Sparse Connectivity
- Shared weights
- Equivariant representation

Properties of CNN





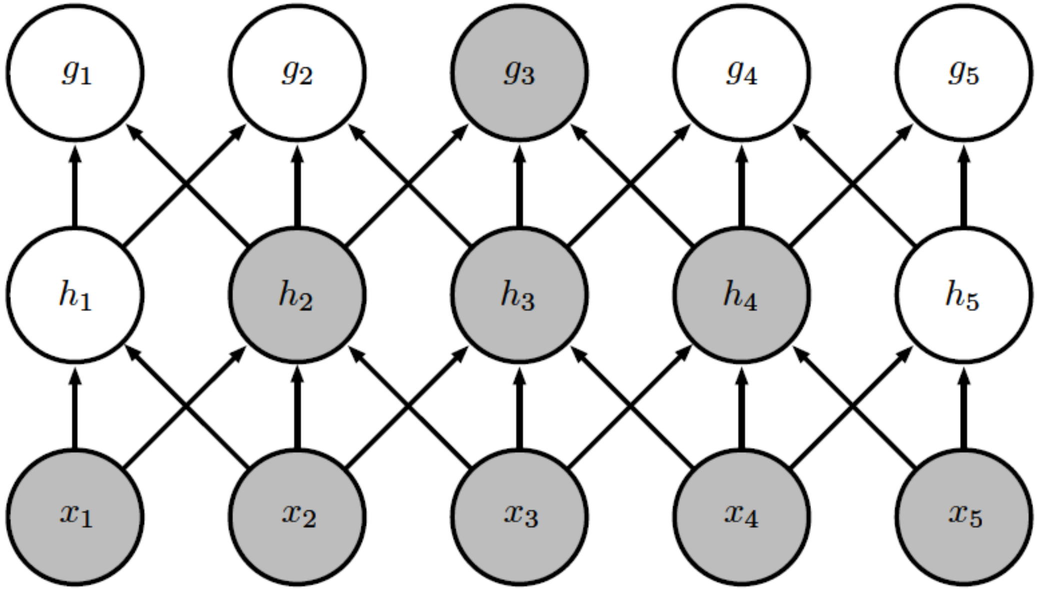
Fewer parameters!

Only connect to 9 inputs, not fully connected **(Sparse Connectivity)**

Source: CS 898: Deep Learning and Its Applications, University of Waterloo, Canada.

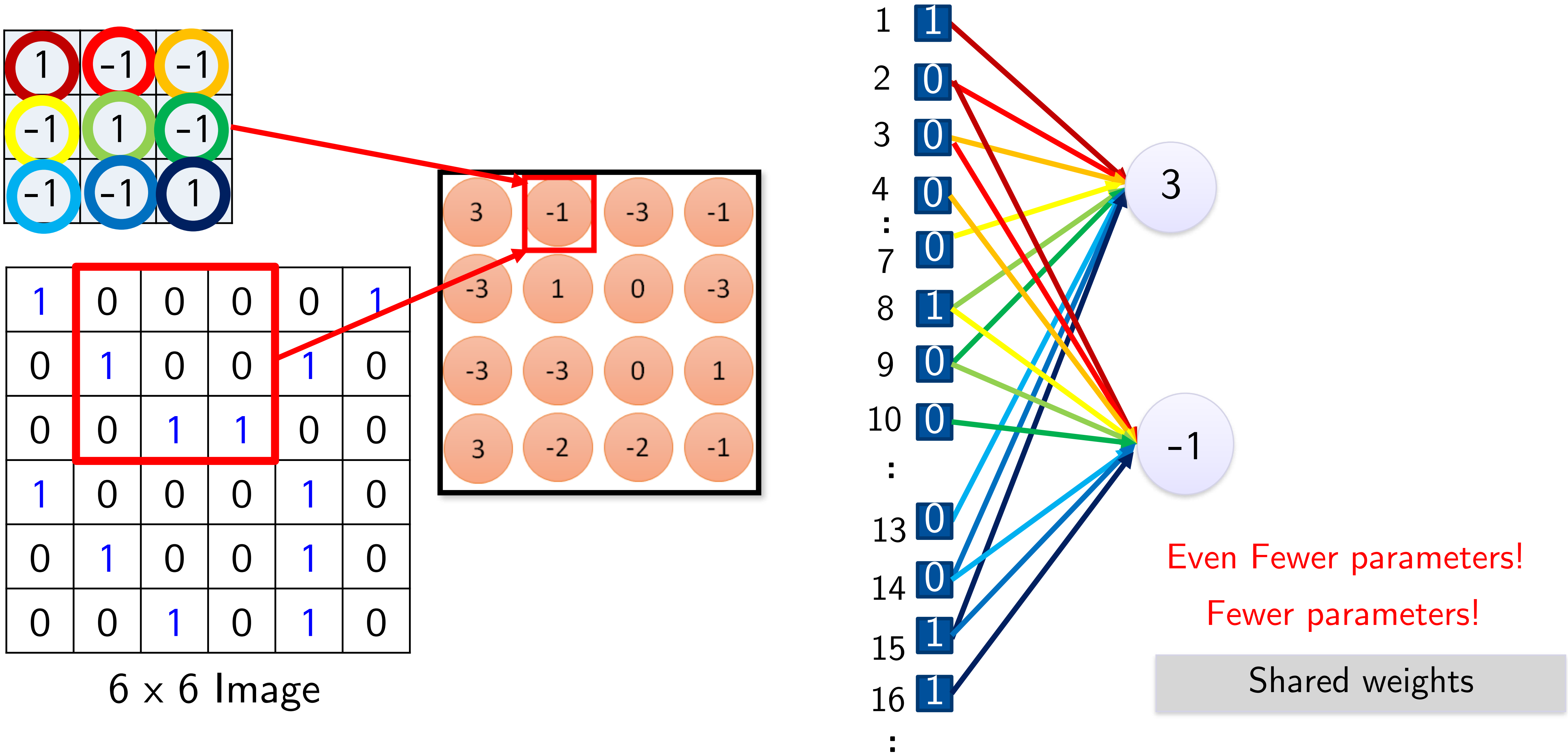
Properties of CNN

Is sparse connectivity good?



Ian Goodellow et al. 2016

Properties of CNN



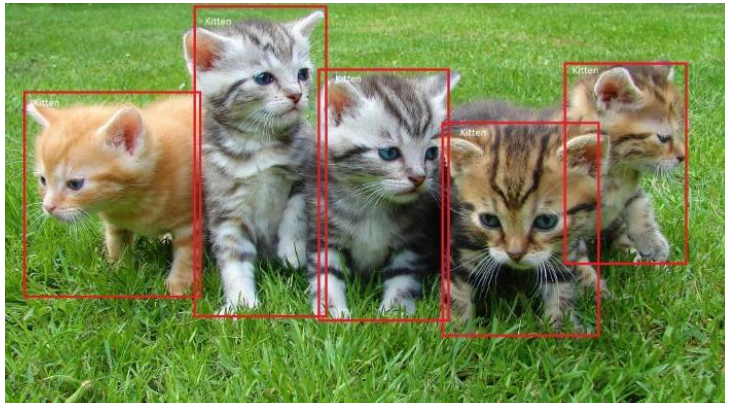
Source: CS 898: Deep Learning and Its Applications, University of Waterloo, Canada.

Equivariance to translation

- A function **f** is equivariant to a function **g** if **f(g(x)) = g(f(x))** or if the output changes in the same way as the input.

- This is achieved by the concept of weight sharing.

- As the same weights are shared across the images, hence if an object occurs in any image, it will be detected irrespective of its position in the image.



Source: [Translational Invariance Vs Translational Equivariance | by Divyanshu Mishra | Towards Data Science]

CNN vs Fully Connected NN

- A CNN compresses the fully connected NN in two ways:
  - Reducing the number of connections
  - Shared weights
- Max pooling further reduces the parameters to characterize an image.

Convolutional Neural Network (CNN) : Non-linearity with activation

Convolution + **ReLU**

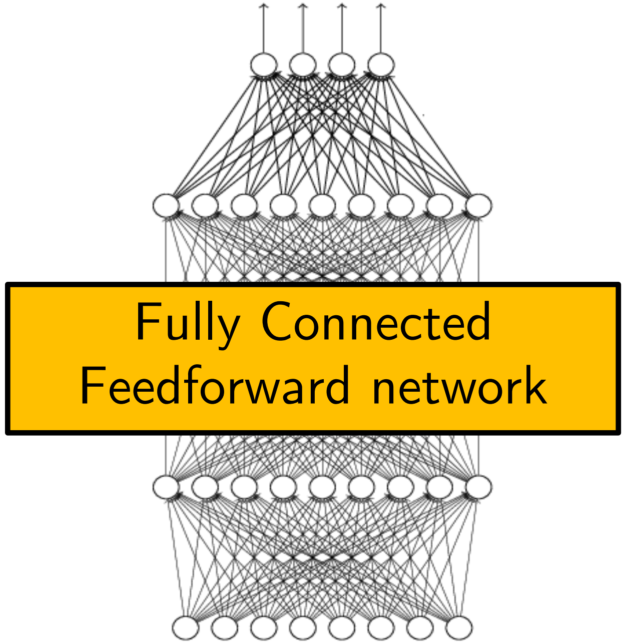
Pooling

Convolution+ **ReLu**

Pooling

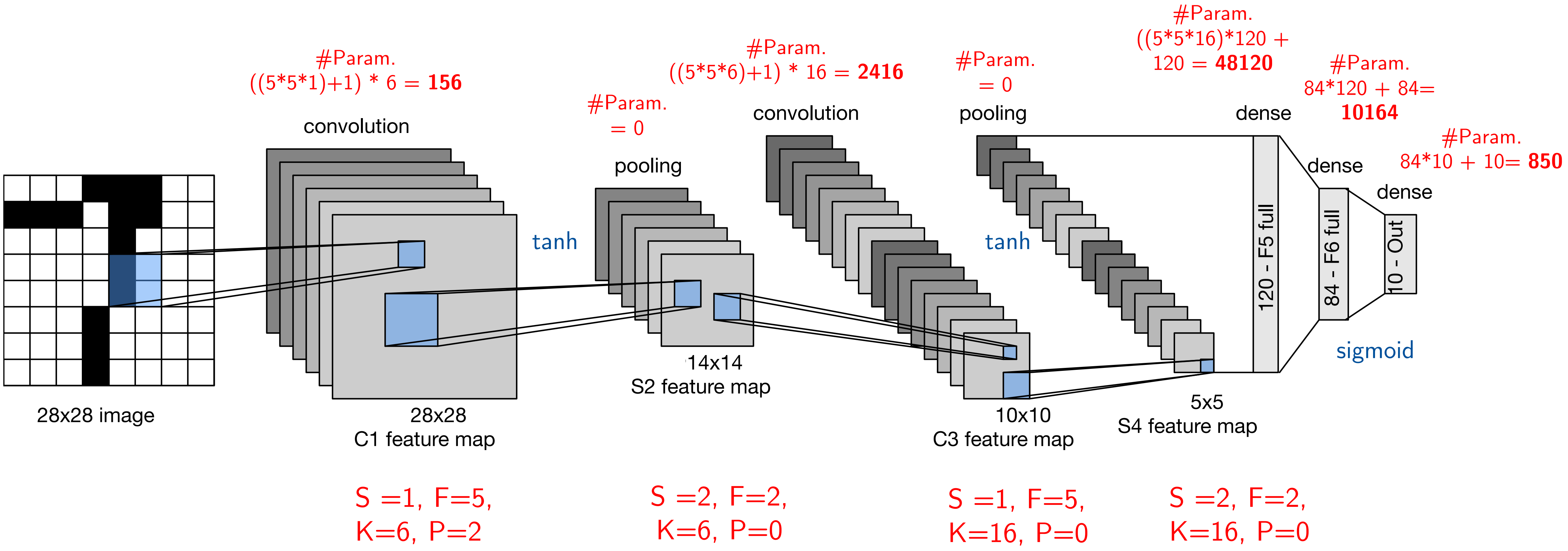
Flattened

cat | dog



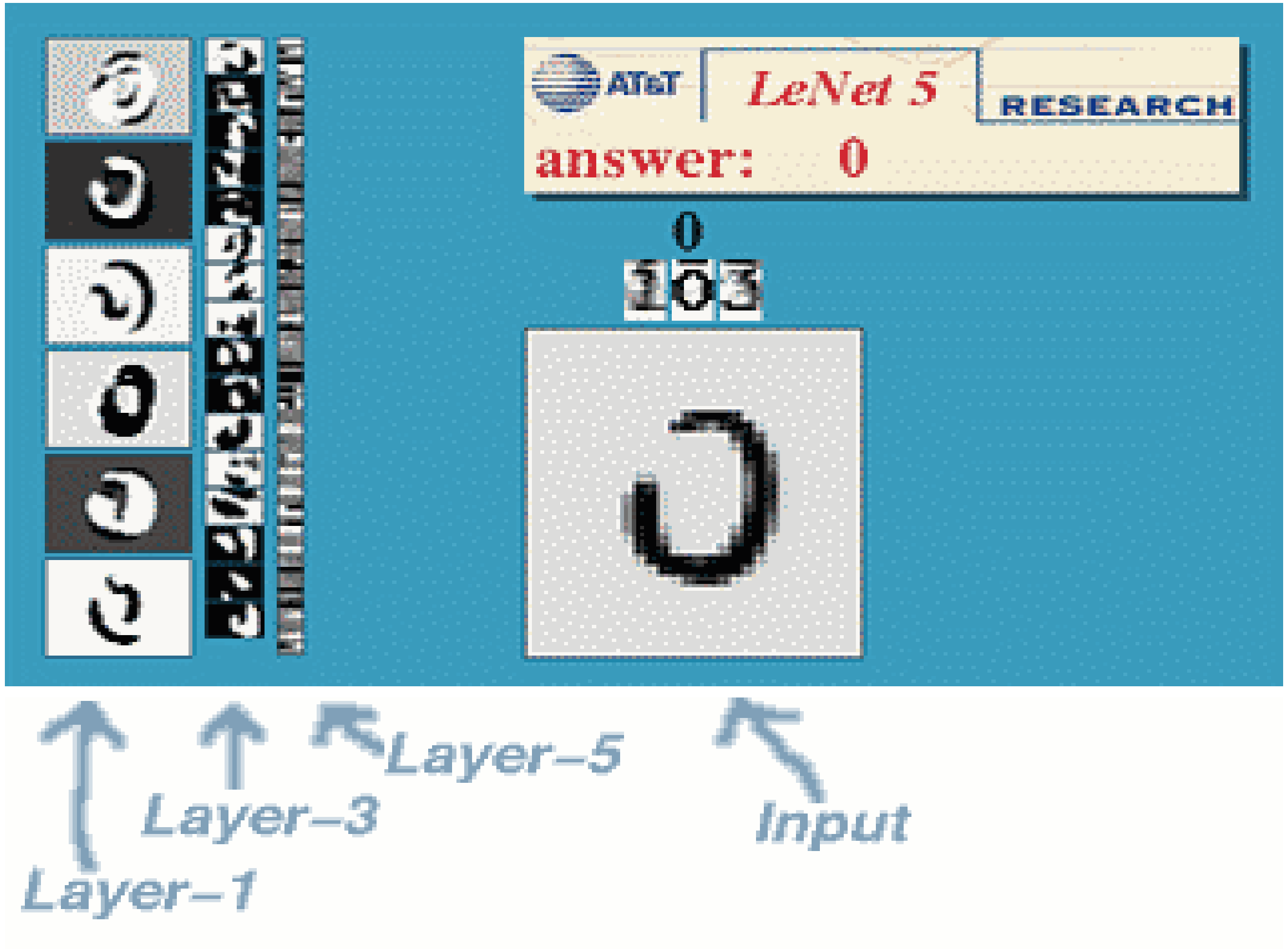
Source: CS 898: Deep Learning and Its Applications, University of Waterloo, Canada.

LeNet-5 Architecture for handwritten text recognition



LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., & others. (**1998**). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

LeNet-5 Architecture for handwritten number recognition



Source: http://yann.lecun.com/

ImageNet Dataset

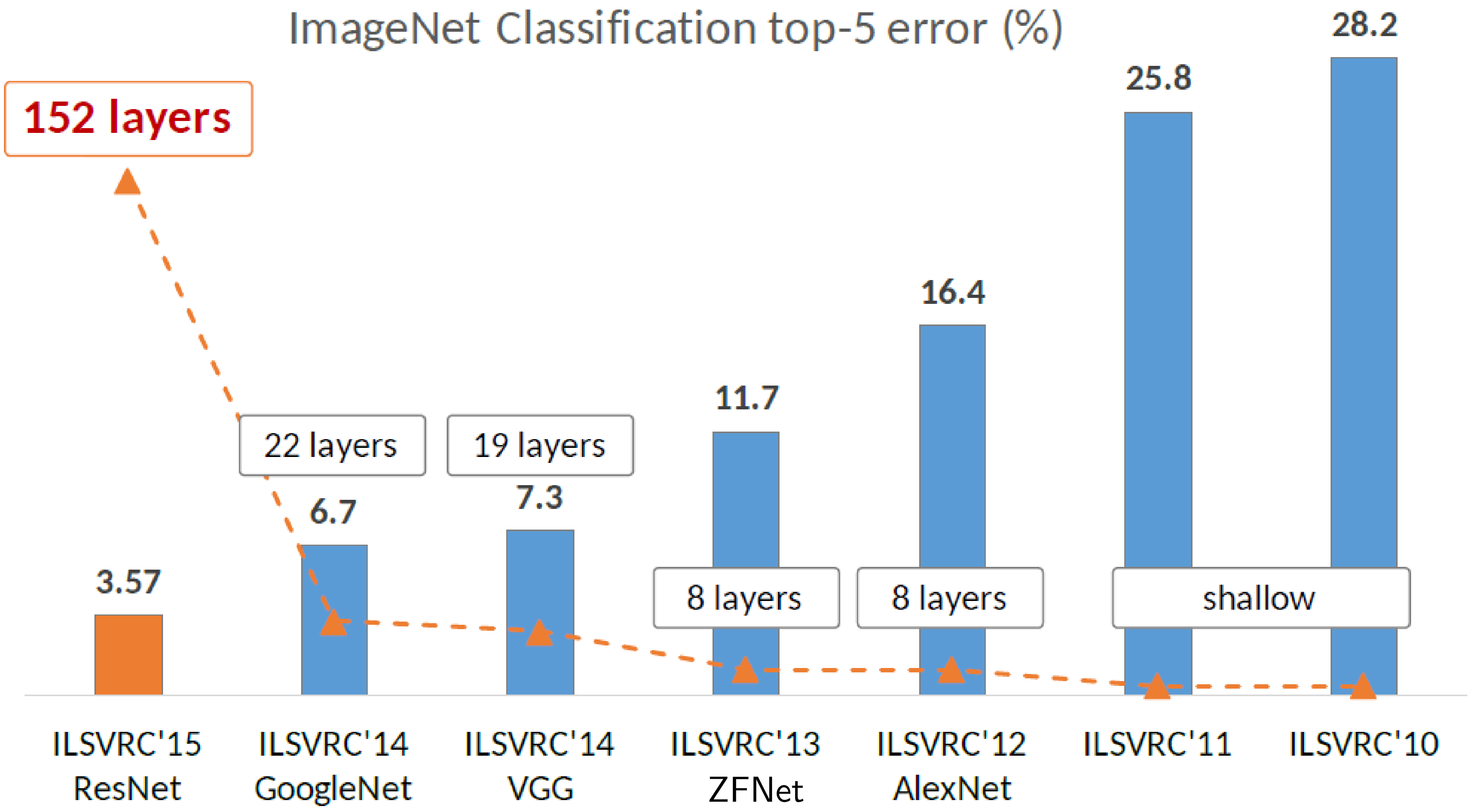
More than 14 million images.

22,000 Image categories

Deng, Jia, et al. "Imagenet: A large-scale hierarchical image database." *IEEE conference on computer vision and pattern recognition*. IEEE, 2009.

ImageNet Large Scale Visual Recognition Challenge

- 1000 ImageNet Categories



AlexNet (2012)

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

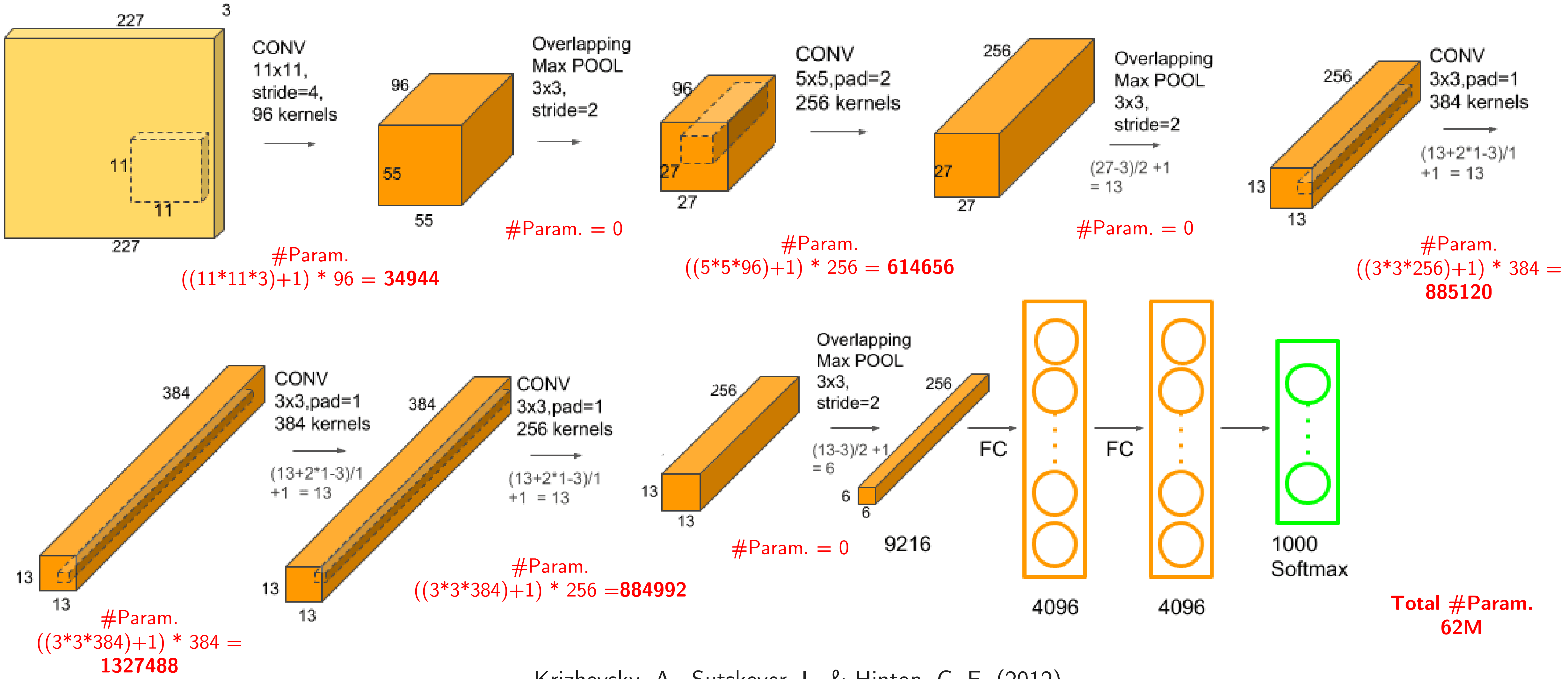
Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

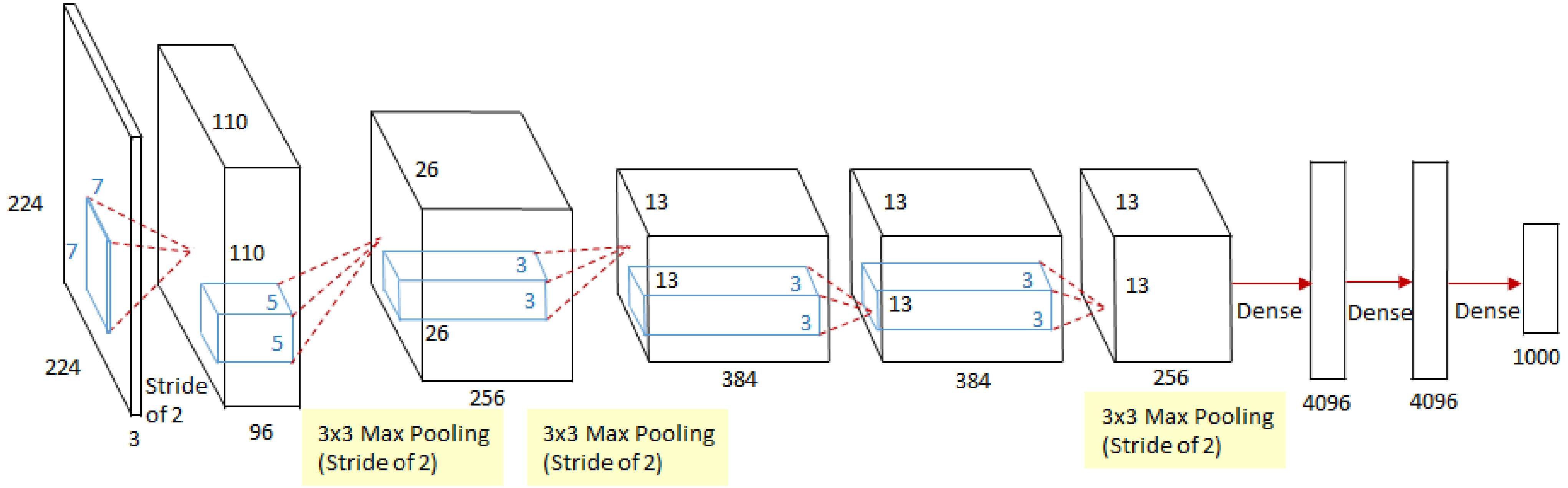
- Used **ReLU** activation function instead of sigmoid and tanh.
- Used **data augmentation** techniques that consisted of image translations, horizontal reflections, and patch extractions.
- Implemented **dropout** layers.

AlexNet Architecture



Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012).
Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems, 25.*

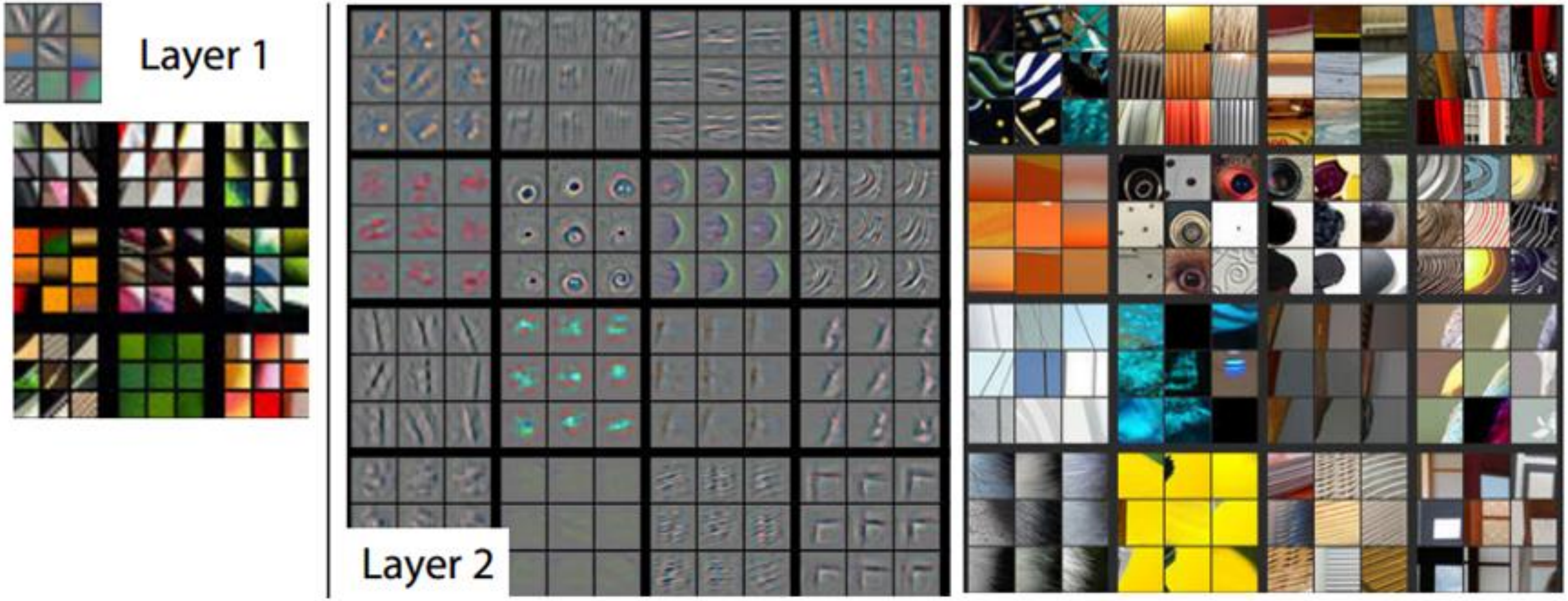
ZFNet Architecture (2013)



- Used filters of size 7x7 instead of 11x11 in AlexNet
- Used Deconvnet to visualize the intermediate results.

Zeiler, M. D., & Fergus, R. (2013). Visualizing and understanding convolutional networks. In *European conference on computer vision* (pp. 818-833). Springer, Cham.

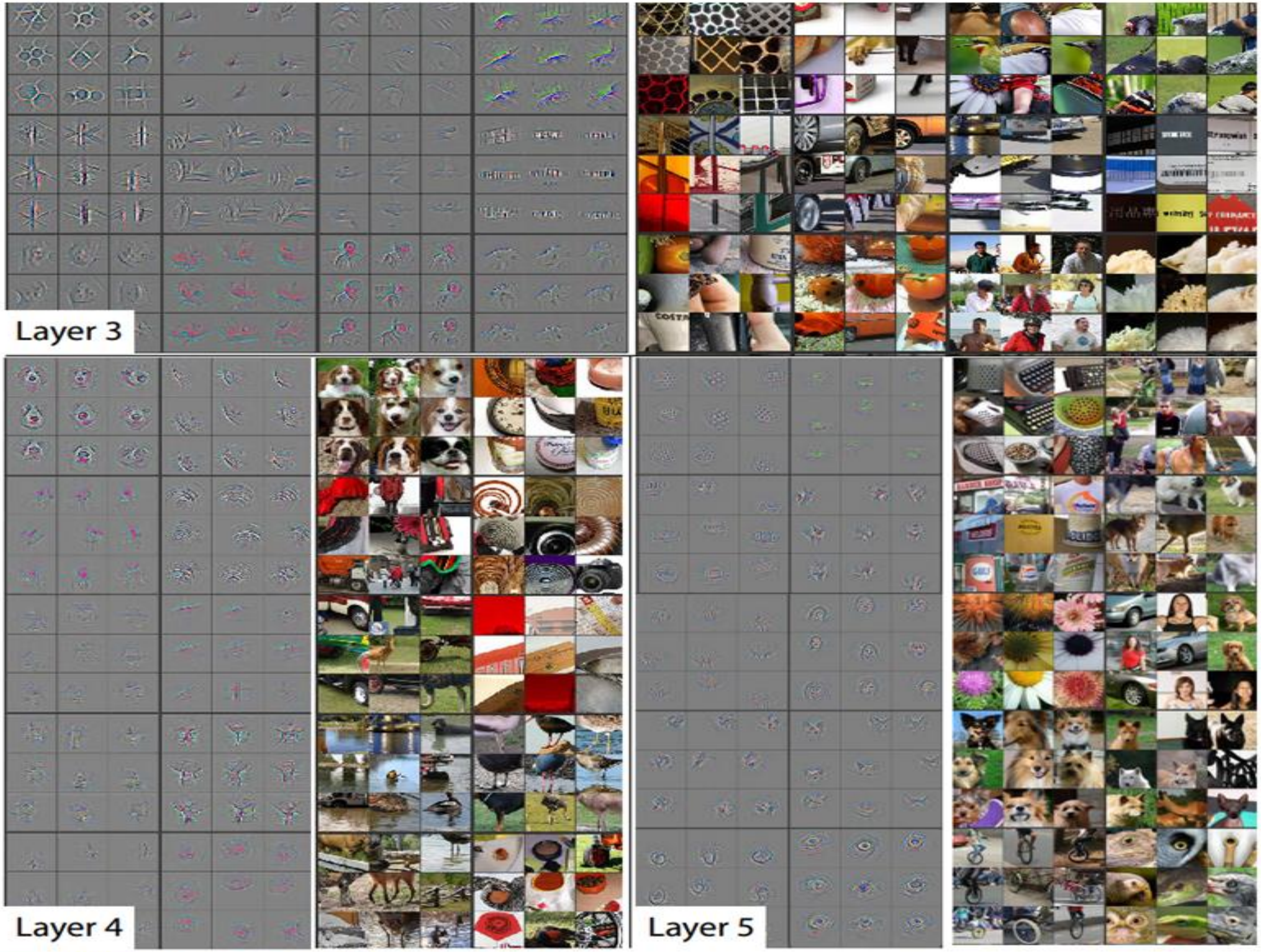
ZFNet



Visualizations of Layer 1 and 2. Each layer illustrates 2 pictures, one which shows the filters themselves and one that shows what part of the image are most strongly activated by the given filter. For example, in the space labled Layer 2, we have representations of the 16 different filters (on the left)

[Visualizing and Understanding Deep Neural Networks by Matt Zeiler - YouTube]

ZFNet



Visualizations of Layers 3, 4, and 5

VGGNet Architecture (2014)

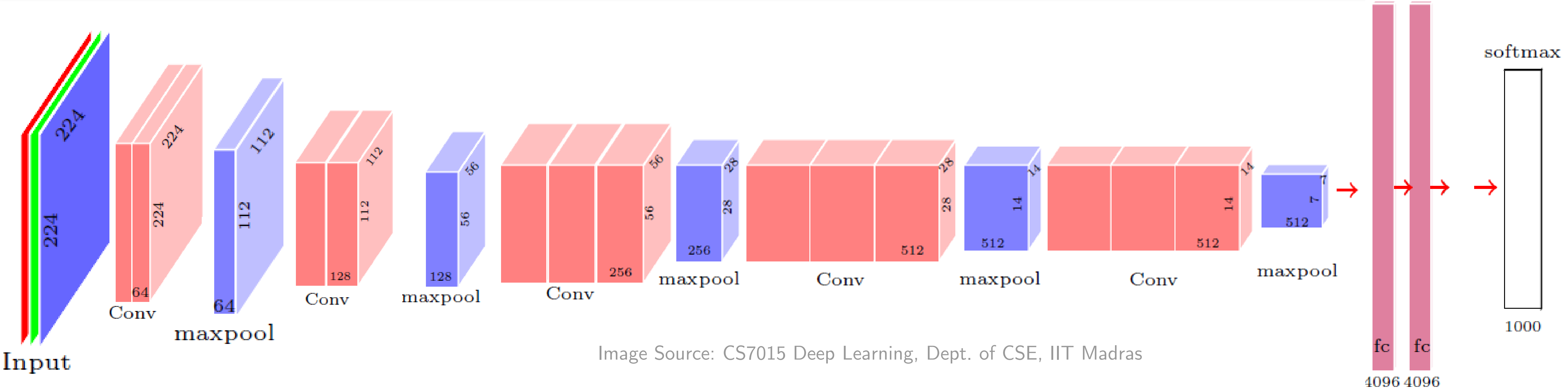
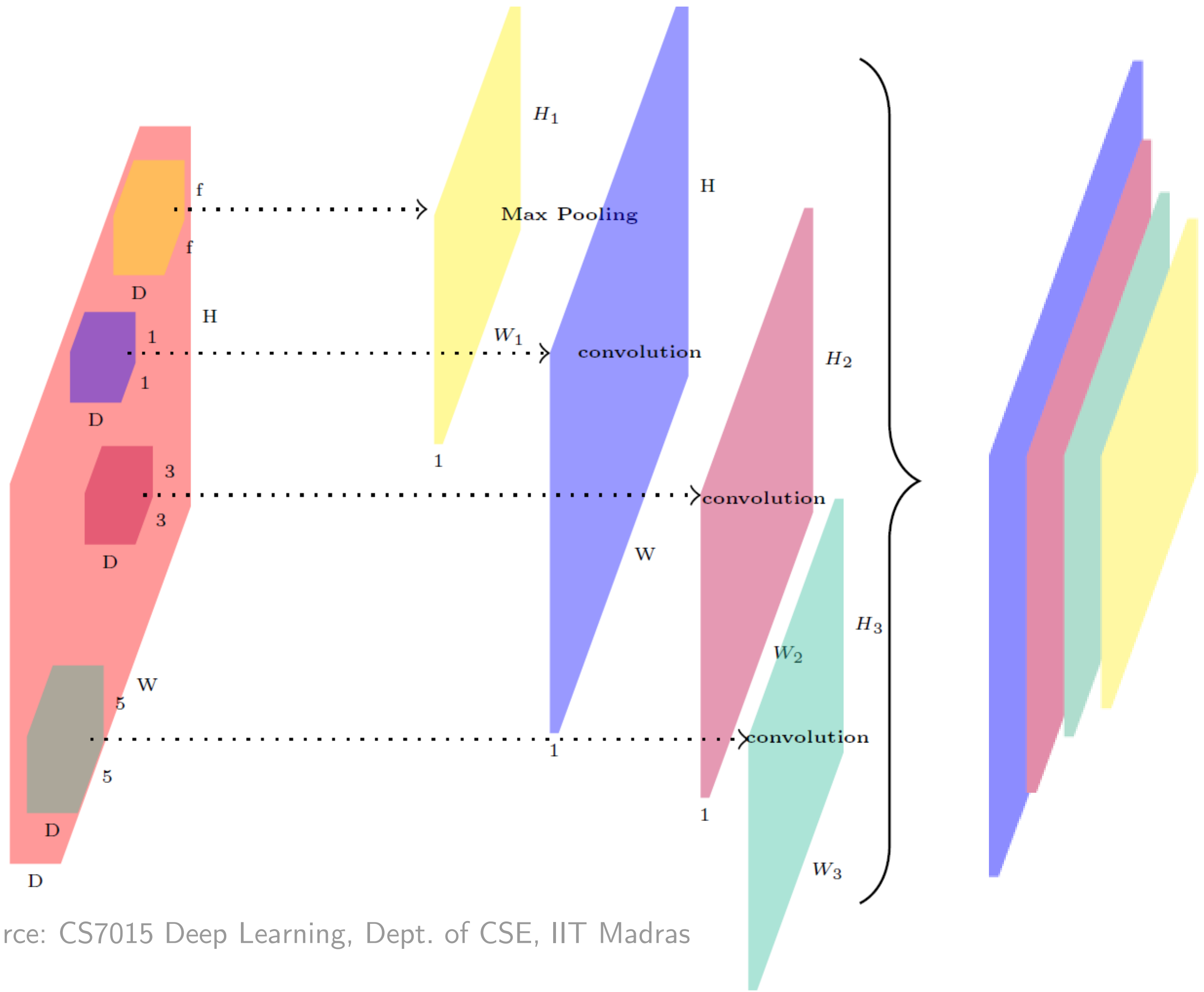


Image Source: CS7015 Deep Learning, Dept. of CSE, IIT Madras

- Used filters of size 3x3 in all the convolution layers.
- 3 conv layers back-to-back have an effective receptive field of 7x7.
- Also called VGG-16 as it has 16 layers.
- This work reinforced the notion that convolutional neural networks have to have a deep network of layers in order for this hierarchical representation of visual data to work

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition , International Conference on Learning Representations (ICLR14)

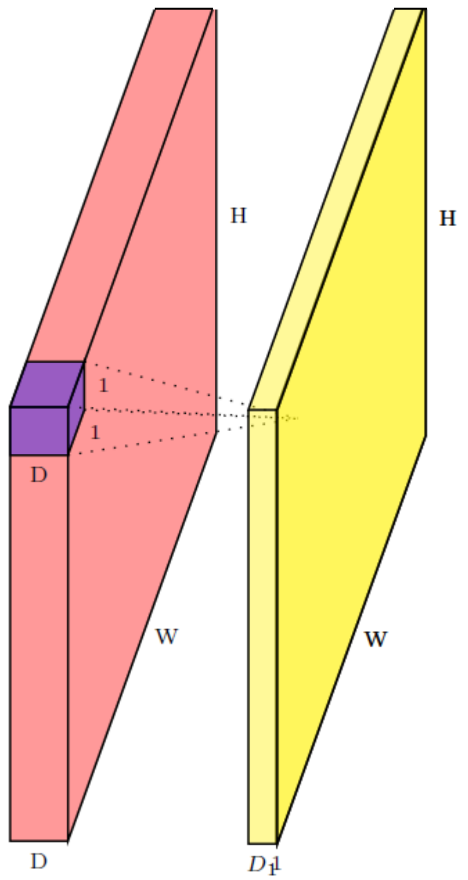
GoogleNet Architecture (2014)



Source: CS7015 Deep Learning, Dept. of CSE, IIT Madras

- Most of the architectures discussed till now apply either of the following after each convolution operation:
  - Max Pooling
  - 3x3 convolution
  - 5x5 convolution
- Idea: Why cant we apply them all together at the same time and concatenate the feature maps.
- Problem: This will result in large number of computations.
- Specifically, each element of the output required **O(FxFxD)** computations

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (CVPR'15)

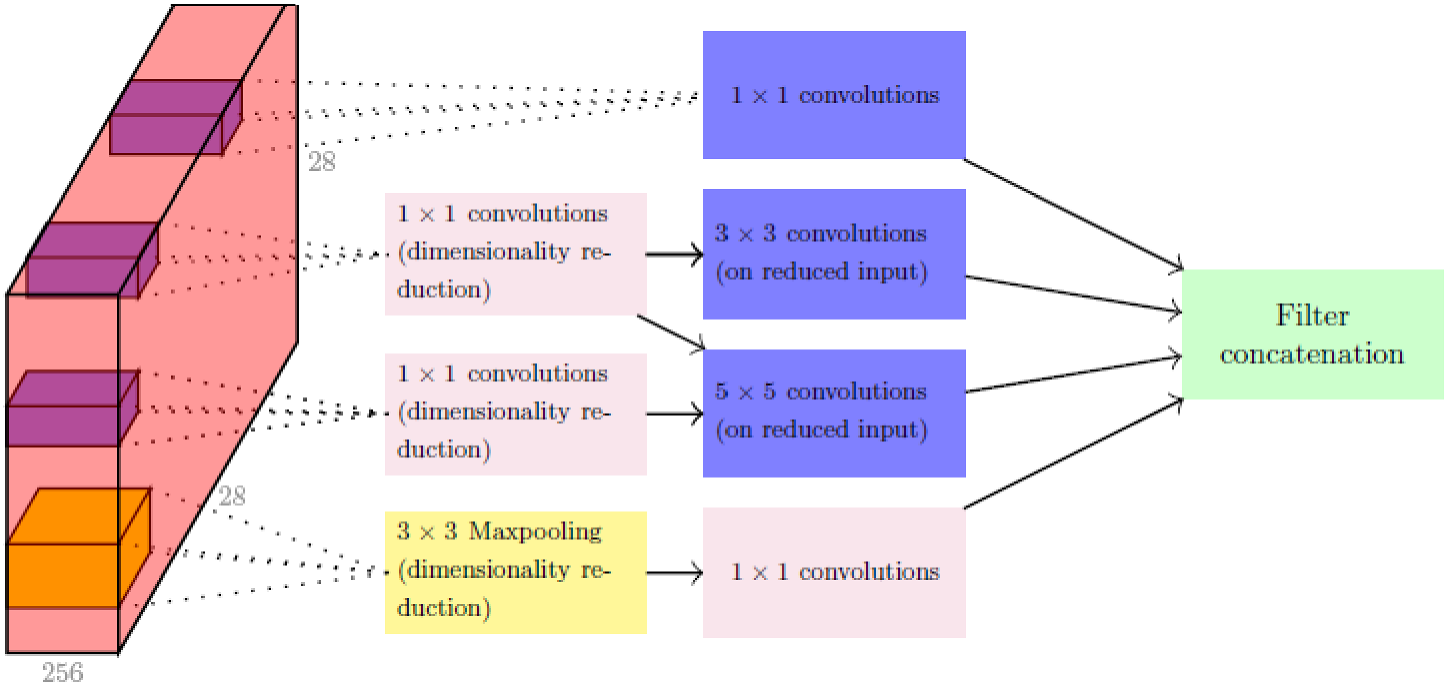


Source: CS7015 Deep Learning, Dept. of CSE, IIT Madras

- Solution: Apply 1x1 convolutions
- 1x1 convolution aggregates along the depth.
- So, if we apply D_1 1x1 convolutions (D_1<D), we will get an output of size W x H x D_1
- So, the total number of computations will reduce to **O(FxFxD_1)**
- We could then apply subsequent 3x3, 5 x5 filters on this reduced output

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (CVPR'15)

GoogleNet Architecture (2014)



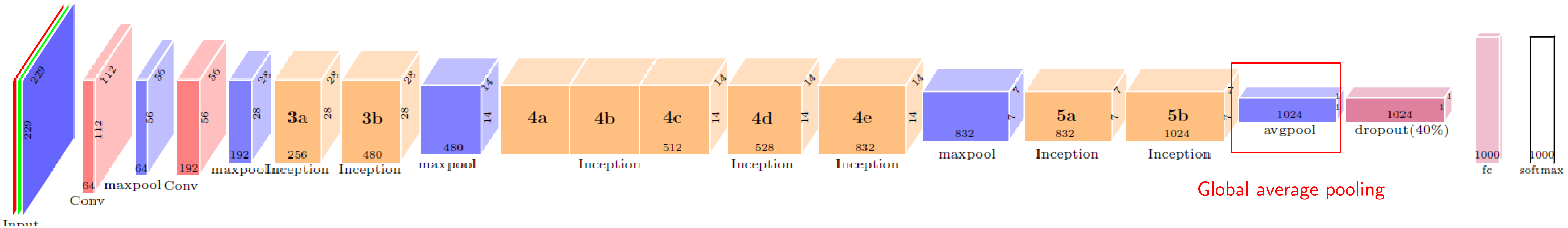
The Inception module

- Also, we might want to use different dimensionality reductions (applying 1x1 convolutions of different sizes) before the 3x3 and 5x5 filters.
- We can also add the maxpooling layer followed by 1x1 convolution.
- After this, we concatenate all these layers.
- This is called the **Inception module.**
- **GoogleNet** contains many such inception modules.

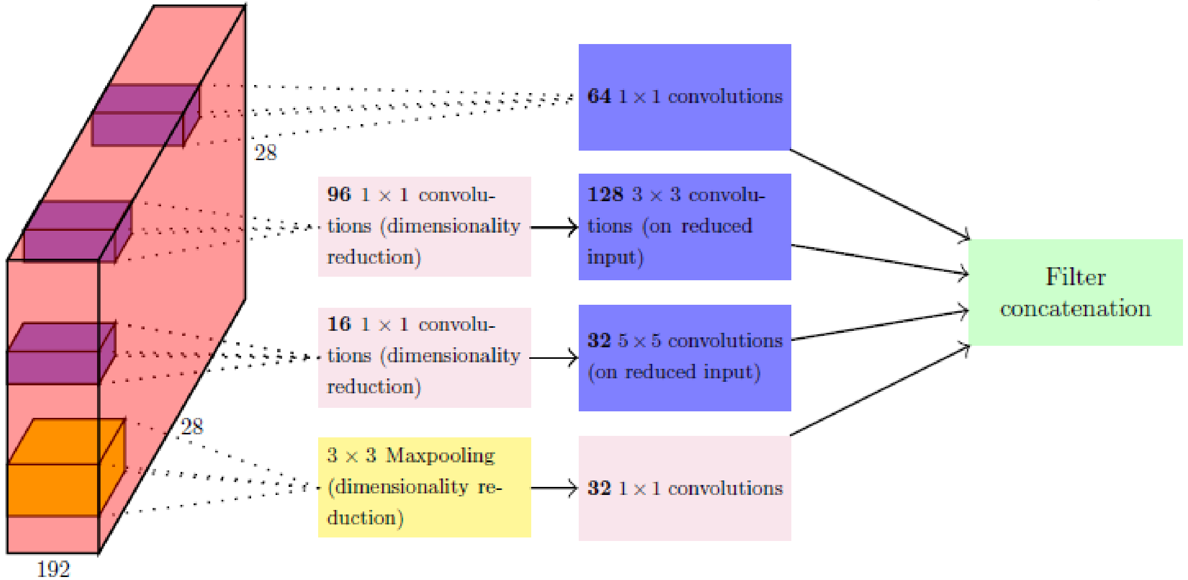
Source: CS7015 Deep Learning, Dept. of CSE, IIT Madras

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (CVPR'15)

GoogleNet Architecture (2014)



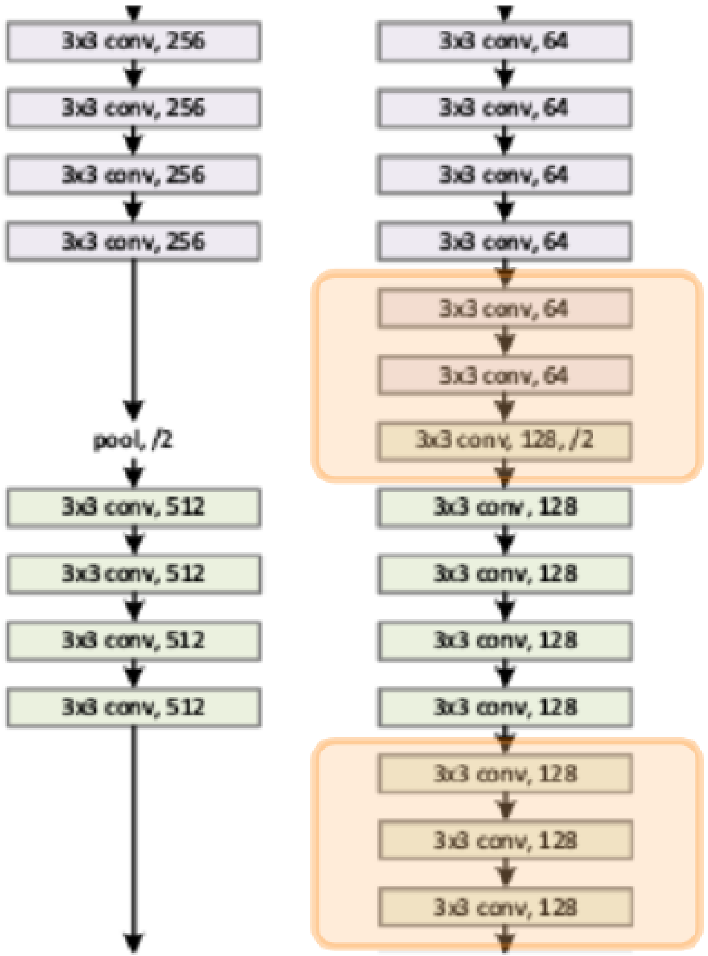
Source: CS7015 Deep Learning, Dept. of CSE, IIT Madras



- 12 times less parameters and 2 times more computations than AlexNet
- Used Global Average Pooling instead of Flattening.

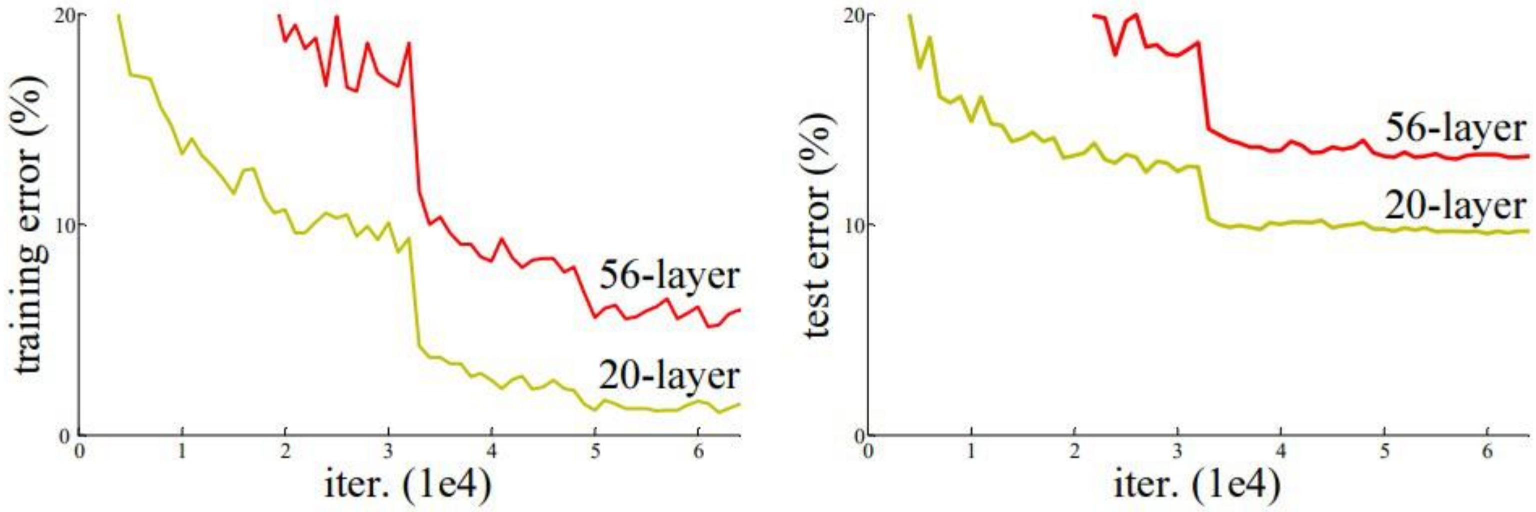
Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (CVPR'15)

ResNet Architecture (2015)



Shallow CNN

Shallow CNN +
Additional layers

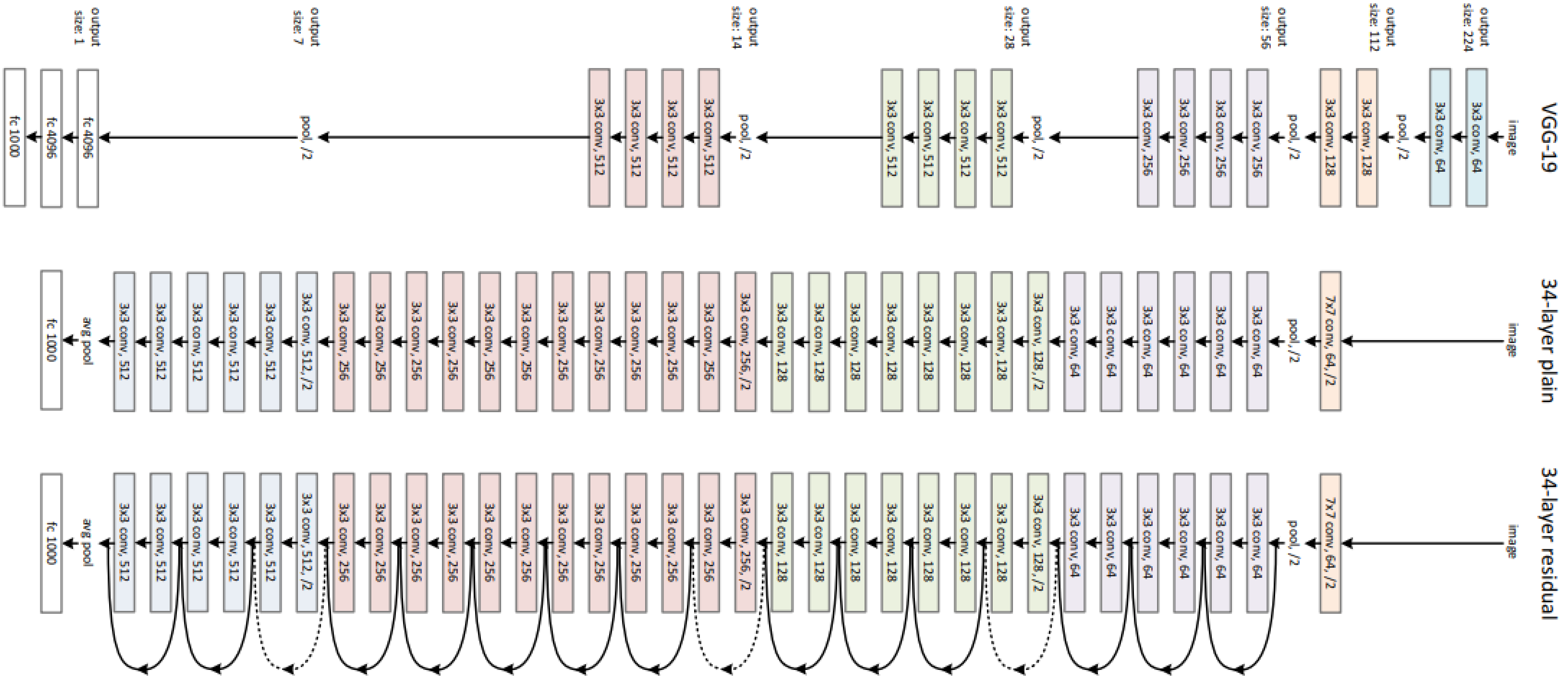


Effect of increasing layers of shallow CNN when experimented over the CIFAR dataset

Source: [Residual Networks (ResNet) - Deep Learning - GeeksforGeeks]

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition.
In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

ResNet Architecture (2015)

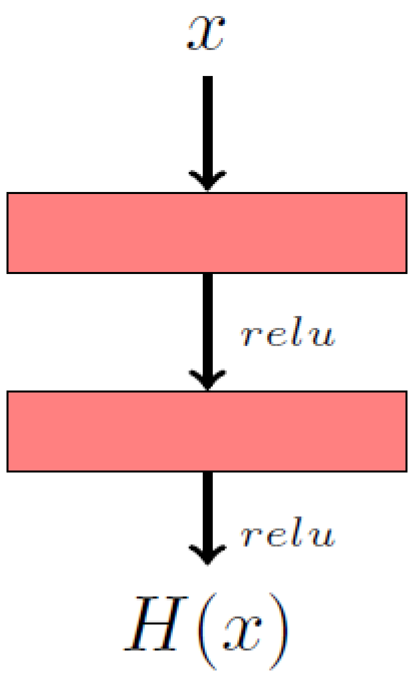


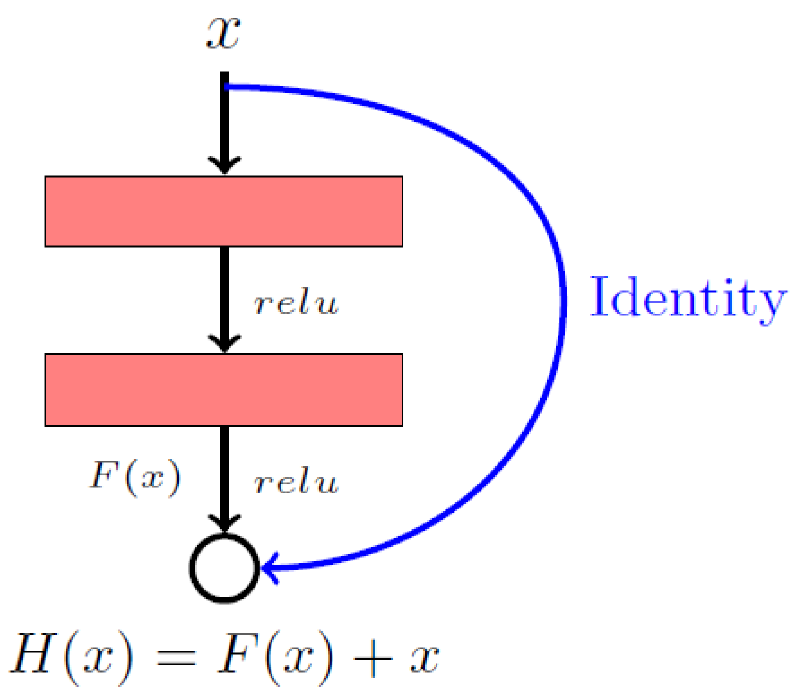
ResNet-34

Source: [Residual Networks (ResNet) - Deep Learning - GeeksforGeeks]()

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

ResNet Architecture (2015)





Source: CS7015 Deep Learning, Dept. of CSE, IIT Madras

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).