

Charu C. Aggarwal  
IBM T J Watson Research Center  
Yorktown Heights, NY

# Convolutional Neural Networks

Neural Networks and Deep Learning, Springer, 2018  
Chapter 8.1–8.2

## Convolutional Neural Networks

- Like recurrent neural networks, convolutional neural networks are *domain-aware* neural networks.
  - The structure of the neural network encodes domain-specific information.
  - Specifically designed for images.
- Images have length, width, and a *depth* corresponding to the number of color channels (typically 3: RGB).
- All layers are spatially structured with length, width, and depth.

## History

- Motivated by Hubel and Wiesel's understanding of the cat's visual cortex.
  - Particular shapes in the visual field excite neurons  $\Rightarrow$  Sparse connectivity with shared weights.
  - Hierarchical arrangement of neurons into simple and complex cells.
  - Neocognitron was first variant  $\Rightarrow$  Motivated *LeNet-5*
- Success in *ImageNet (ILSVRC)* competitions brought attention to deep learning.

## Basic Structure of a Convolutional Neural Network

- Most layers have length, width, and depth.
  - The length and width are almost always the same.
  - The depth is 3 for color images, 1 for grayscale, and an arbitrary value for hidden layers.
- Three operations are convolution, max-pooling, and ReLU.
  - Maxpooling substituted with strided convolution in recent years.
  - The convolution operation is analogous to the matrix multiplication in a conventional network.

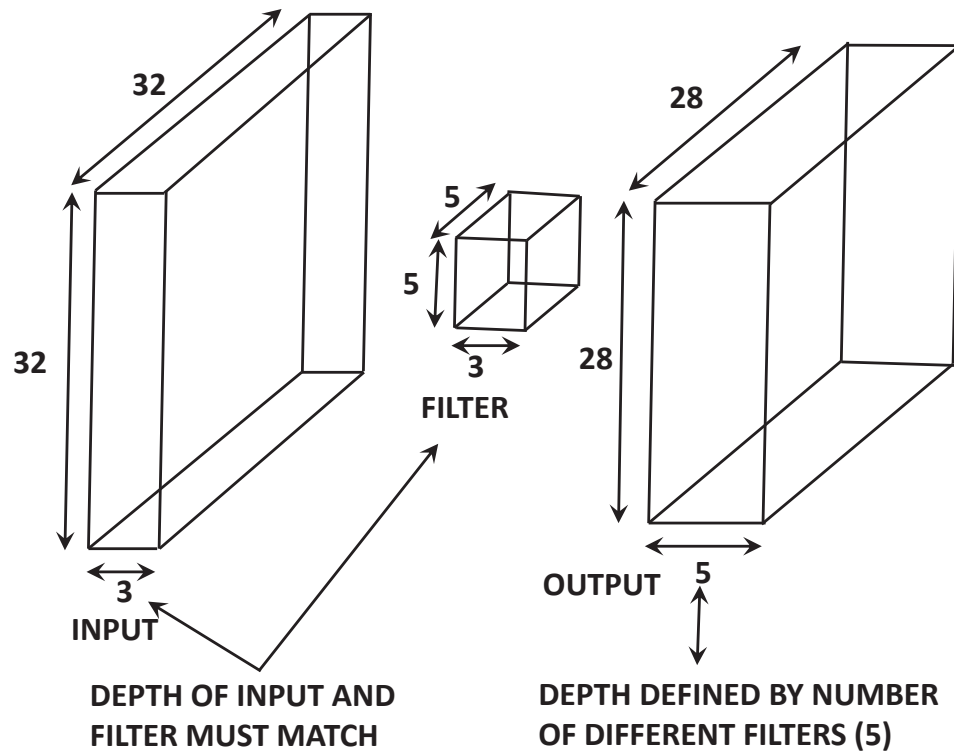
## Filter for Convolution Operation

- Let the input volume of layer  $q$  have dimensions  $L_q \times B_q \times d_q$ .
- The operation uses a *filter* of size  $F_q \times F_q \times d_q$ .
  - The filter's spatial dimensions must be no larger than layer's spatial dimensions.
  - The filter depth *must* match input volume.
  - Typically, the filter spatial size  $F_q$  is a small odd number like 3 or 5.

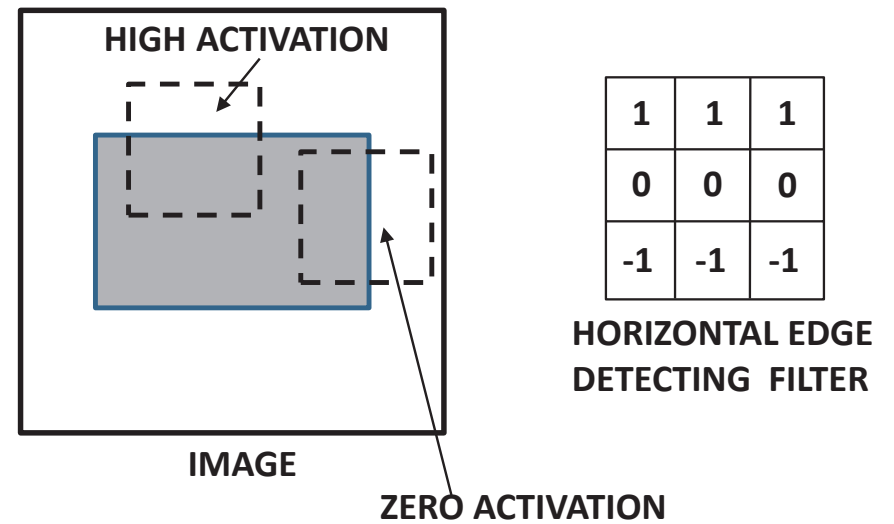
## Convolution Operation

- Spatially align the top-left corner of filter with each of  $(L_q - F_q + 1) \times (B_q - F_q + 1)$  spatial positions.
  - Corresponds to number of positions for top left corner in input volume, so that filter fully fits inside layer volume.
- Perform elementwise multiplication between input/filter over all  $F_q \times F_q \times d_q$  aligned elements and add.
- Creates a single spatial map in the output of size  $(L_q - F_q + 1) \times (B_q - F_q + 1)$ .
- Multiple filters create depth in output volume.

# Convolution Operation: Pictorial Illustration of Dimensions

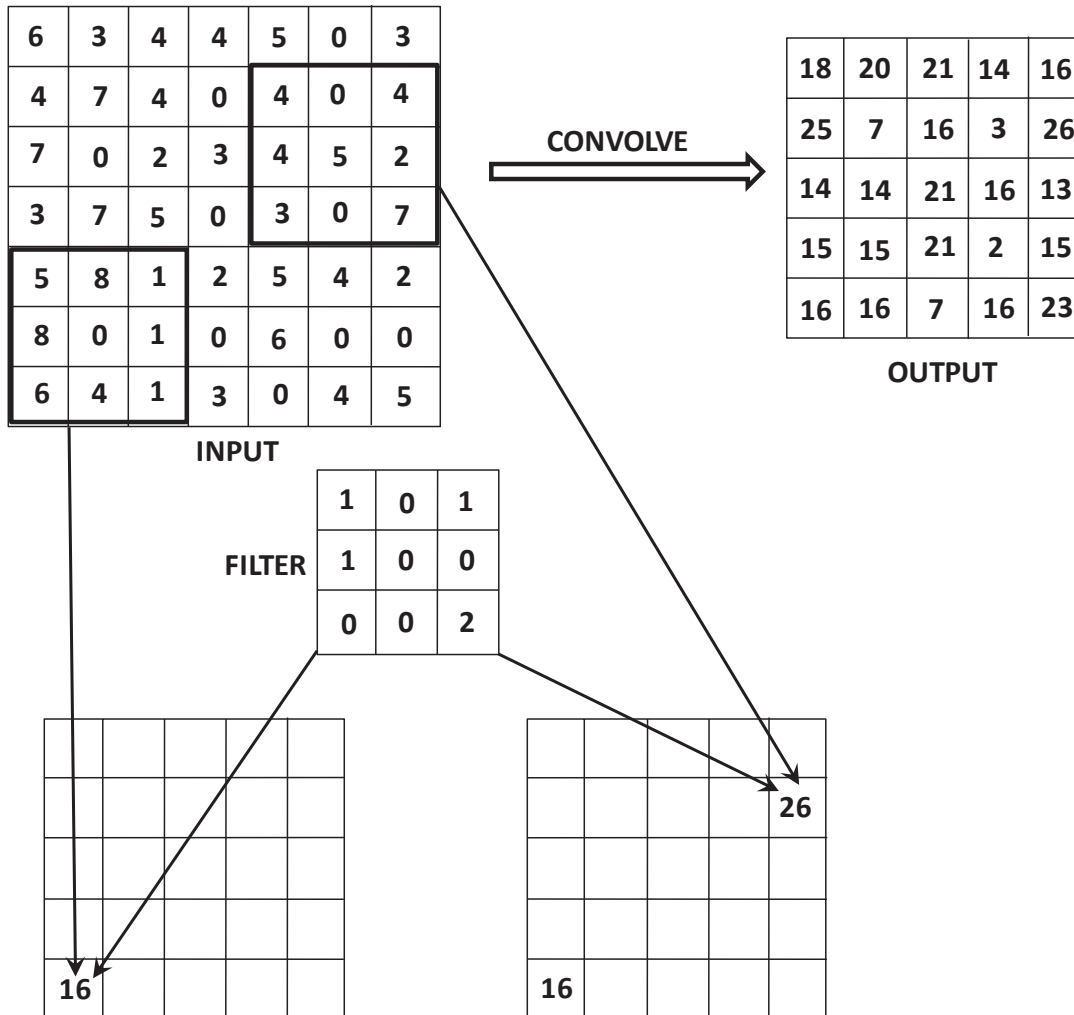


(a) Input and output dimensions



(b) Sliding the filter

# Convolution Operation: Numerical Example with Depth 1



- Add up over multiple activations maps from the depth

## Understanding Convolution

- Sparse connectivity because we are creating a feature from a region in the input volume of the size of the filter.
  - Trying to explore smaller regions of the image to find shapes.
- Shared weights because we use the same filter across entire spatial volume.
  - Interpret a shape in various parts of the image in the same way.

## Effects of Convolution

- Each feature in a hidden layer captures some properties of a region of input image.
- A convolution in the  $q$ th layer increases the *receptive field* of a feature from the  $q$ th layer to the  $(q + 1)$ th layer.
- Consider using a  $3 \times 3$  filter successively in three layers:
  - The activations in the first, second, and third hidden layers capture pixel regions of size  $3 \times 3$ ,  $5 \times 5$ , and  $7 \times 7$ , respectively, in the *original input image*.

## Need for Padding

- The convolution operation reduces the size of the  $(q + 1)$ th layer in comparison with the size of the  $q$ th layer.
  - This type of reduction in size is not desirable in general, because it tends to lose some information along the borders of the image (or of the feature map, in the case of hidden layers).
- This problem can be resolved by using *padding*.
- By adding  $(F_q - 1)/2$  “pixels” all around the borders of the feature map, one can maintain the size of the spatial image.



## Types of Padding

- No padding: When no padding is used around the borders of the image  $\Rightarrow$  Reduces the size of spatial footprint by  $(F_q - 1)$ .
- Half padding: Pad with  $(F_q - 1)/2$  “pixels”  $\Rightarrow$  Maintain size of spatial footprint.
- Full padding: Pad with  $(F_q - 1)$  “pixels”  $\Rightarrow$  Increase size of spatial footprint with  $(F_q - 1)$ .

$$2 * \text{Amount Padded} + \text{Size Reduction} = (F_q - 1) \quad (1)$$

- Note that amount padded is on both sides  $\Rightarrow$  Explains factor of 2

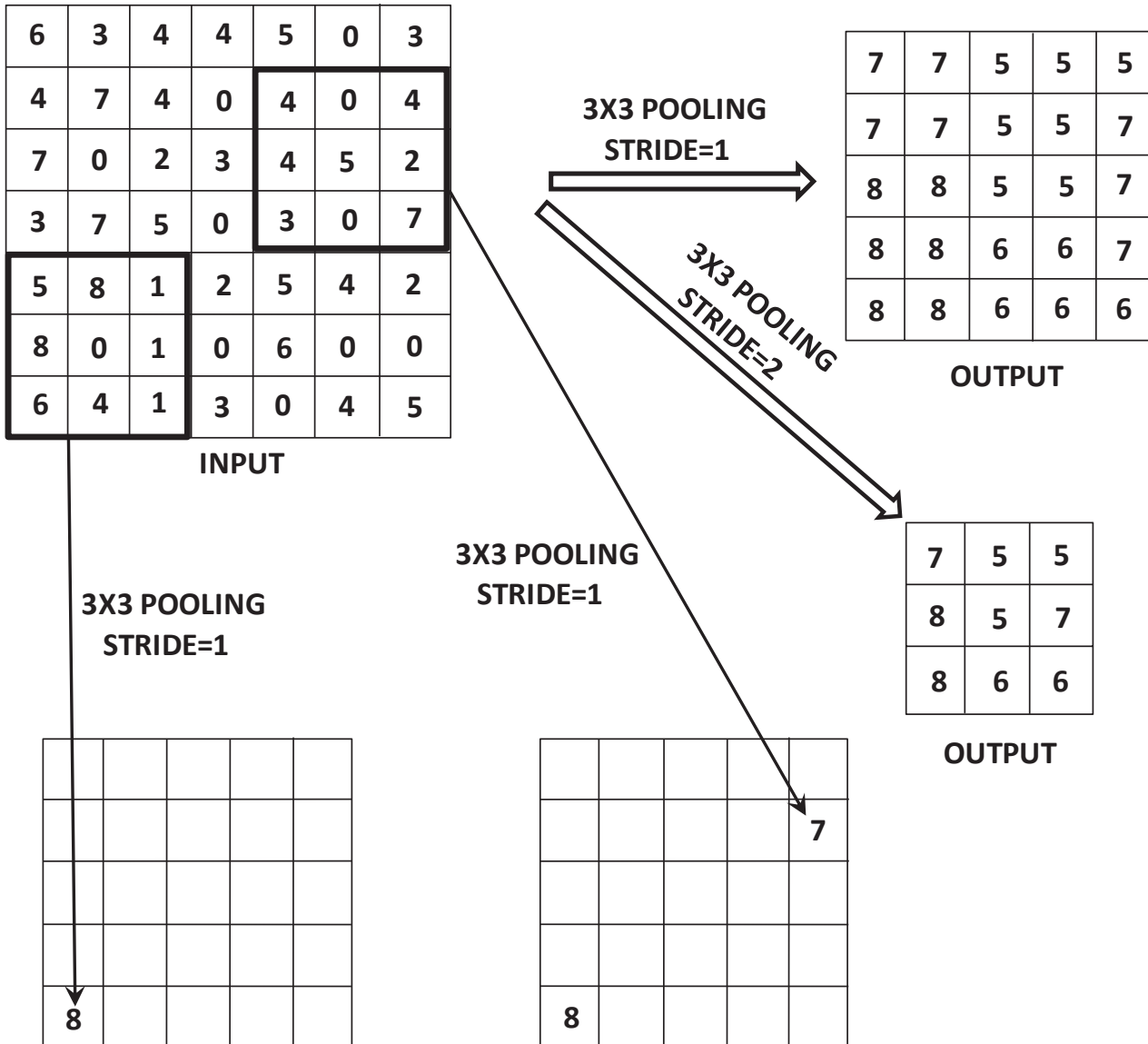
## Strided Convolution

- When a stride of  $S_q$  is used in the  $q$ th layer, the convolution is performed at the locations  $1, S_q + 1, 2S_q + 1$ , and so on along both spatial dimensions of the layer.
- The spatial size of the output on performing this convolution has height of  $(L_q - F_q)/S_q + 1$  and a width of  $(B_q - F_q)/S_q + 1$ .
  - Exact divisibility is required.
- Strided convolutions are sometimes used in lieu of max-pooling.

## Max Pooling

- The pooling operation works on small grid regions of size  $P_q \times P_q$  in each layer, and produces another layer *with the same depth*.
- For each square region of size  $P_q \times P_q$  in each of the  $d_q$  activation maps, the *maximum* of these values is returned.
- It is common to use a stride  $S_q > 1$  in pooling (often we have  $P_q = S_q$ ).
- Length of the new layer will be  $(L_q - P_q)/S_q + 1$  and the breadth will be  $(B_q - P_q)/S_q + 1$ .
- Pooling drastically reduces the spatial dimensions of each activation map.

# Pooling Example



## ReLU

- Use of ReLU is a straightforward one-to-one operation.
- The number of feature maps and spatial footprint size is retained.
- Often stuck at the end of a convolution operation and not shown in architectural diagrams.

## Fully Connected Layers: Stuck at the End

- Each feature in the final spatial layer is connected to each hidden state in the first fully connected layer.
- This layer functions in exactly the same way as a traditional feed-forward network.
- In most cases, one might use more than one fully connected layer to increase the power of the computations towards the end.
- The connections among these layers are exactly structured like a traditional feed-forward network.
- The vast majority of parameters lie in the fully connected layers.

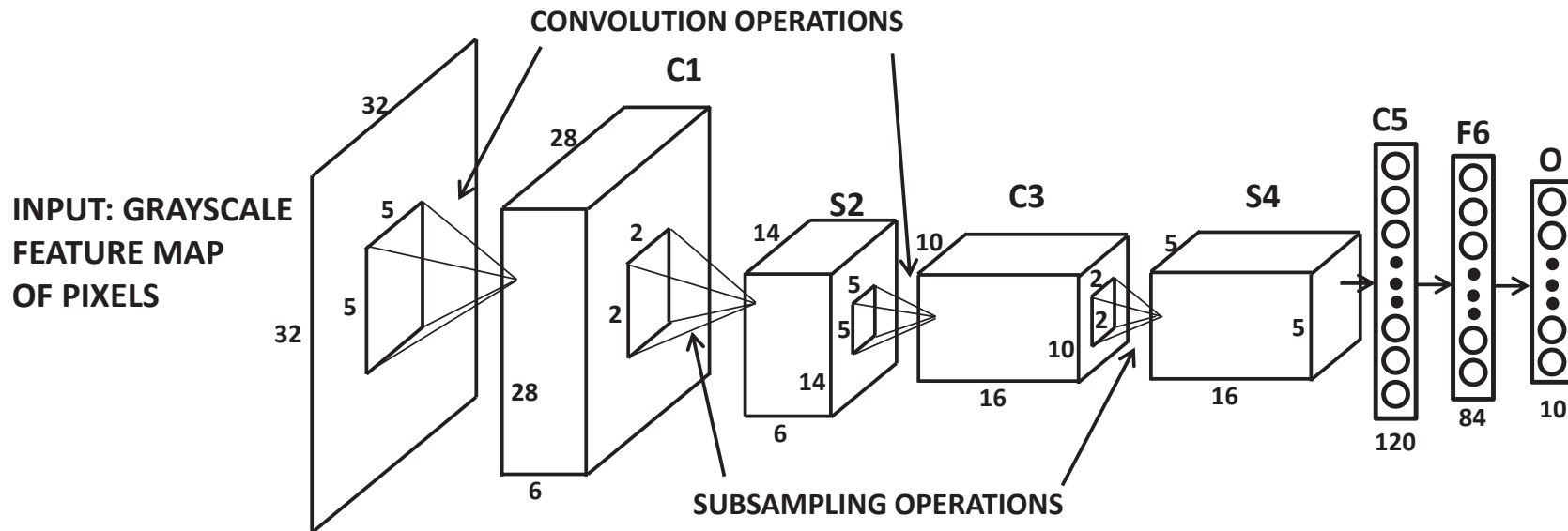
## Interleaving Between Layers

- The convolution, pooling, and ReLU layers are typically interleaved in order to increase expressive power.
- The ReLU layers often follow the convolutional layers, just as a nonlinear activation function typically follows the linear dot product in traditional neural networks.
- After two or three sets of convolutional-ReLU combinations, one might have a max-pooling layer.
- Examples:

CRCRP

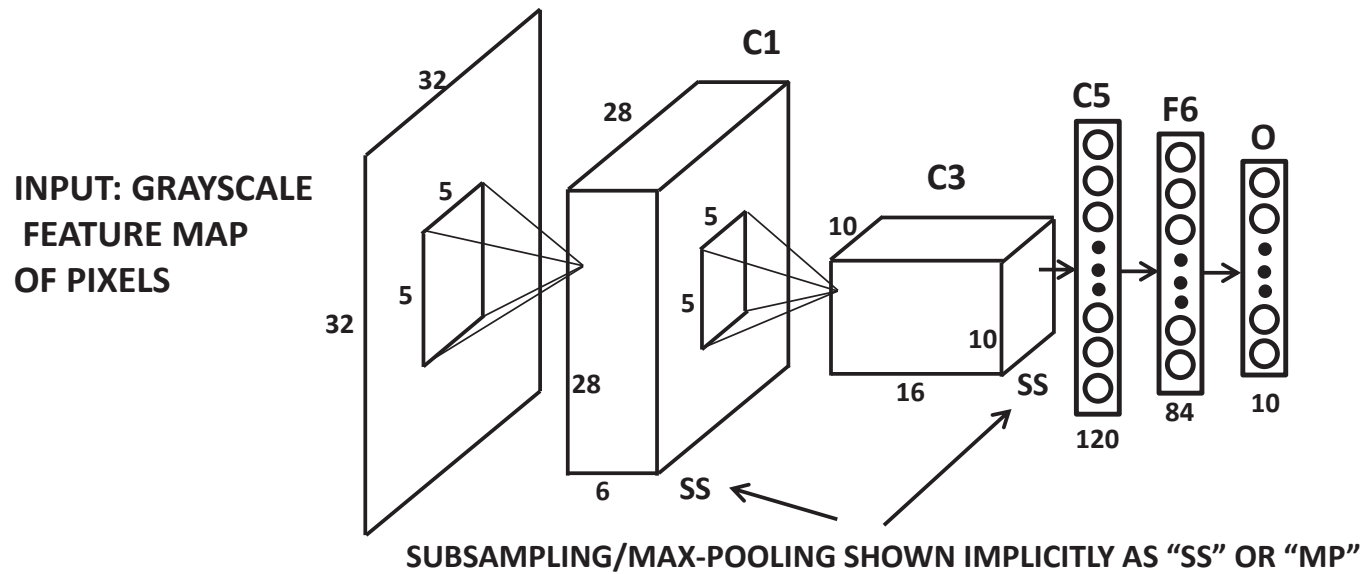
CRCRCRP

## Example: LeNet-5: Full Notation



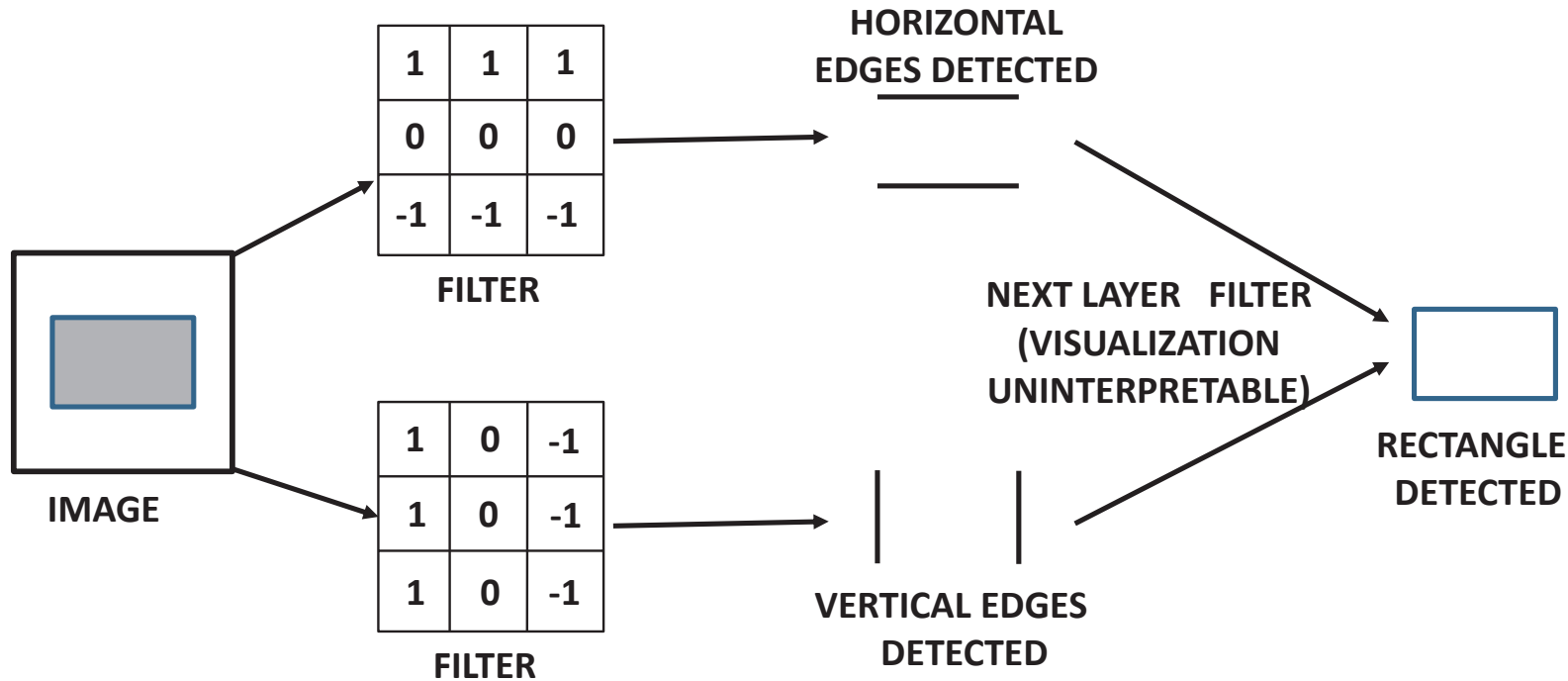
- The earliest convolutional network

## Example: LeNet-5: Shorthand Notation



- Subsampling layers are not explicitly shown

# Feature Engineering



- The early layers detect primitive features and later layers complex ones.
- During training the filters will be learned to identify relevant shapes.

## Hierarchical Feature Engineering

- Successive layers put together primitive features to create more complex features.
- Complex features represent regularities in the data, that are valuable for features.
- Mid-level features might be honey-combs.
- Higher-level features might be a part of a face.
- The network is a master of extracting repeating shapes in data-driven manner.

Charu C. Aggarwal  
IBM T J Watson Research Center  
Yorktown Heights, NY

# Backpropagation in Convolutional Neural Networks and Its Visualization Applications

Neural Networks and Deep Learning, Springer, 2018  
Chapter 8.3, 8.5

## Backpropagation in Convolutional Neural Networks

- Three operations of convolutions, max-pooling, and ReLU.
- The ReLU backpropagation is the same as any other network.
  - Passes gradient to a previous layer only if the original input value was positive.
- The max-pooling passes the gradient flow through the largest cell in the input volume.
- Main complexity is in backpropagation through convolutions.

## Backpropagating through Convolutions

- Traditional backpropagation is transposed matrix multiplication.
- Backpropagation through convolutions is transposed convolution (i.e., with an *inverted filter*).
- Derivative of loss with respect to each cell is backpropagated.
  - Elementwise approach of computing which cell in input contributes to which cell in output.
  - Multiplying with an inverted filter.
- Convert layer-wise derivative to weight-wise derivative and add over shared weights.

## Backpropagation with an Inverted Filter [Single Channel]

a	b	c
d	e	f
g	h	i

**FILTER DURING  
CONVOLUTION**

i	h	g
f	e	d
c	b	a

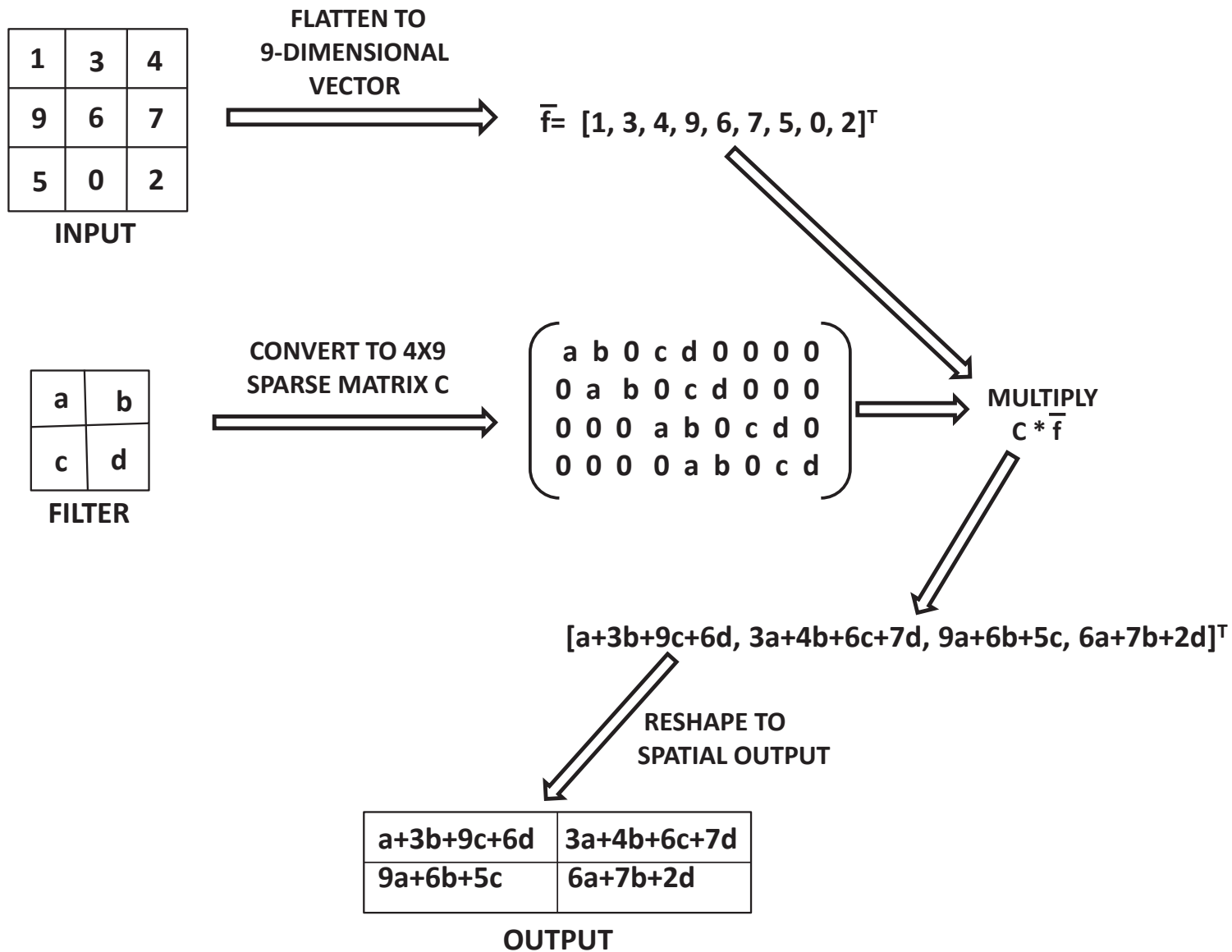
**FILTER DURING  
BACKPROPAGATION**

- Multichannel case: We have 20 filters for 3 input channels (RGB)  $\Rightarrow$  We have  $20 \times 3 = 60$  spatial slices.
- Each of these 60 spatial slices will be inverted and grouped into 3 sets of filters with depth 20 (one each for RGB).
- Backpropagate with newly grouped filters.

## Convolution as a Matrix Multiplication

- Convolution can be presented as a matrix multiplication.
  - Useful during forward and backward propagation.
  - Backward propagation can be presented as transposed matrix multiplication

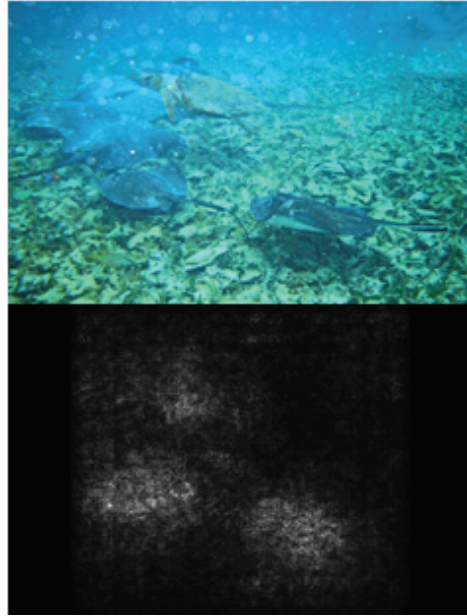
# Convolution as a Matrix Multiplication



## Gradient-Based Visualization

- Can backpropagate all the way back to the input layer.
- Imagine the output  $o$  is the probability of a class label like “dog”
- Compute  $\frac{\partial o}{\partial x_i}$  for each pixel  $x_i$  of each color.
  - Compute maximum absolute magnitude of gradient over RGB colors and create grayscale image.

## Gradient-Based Visualization



- Examples of portions of specific images activated by particular class labels. (©2014 Simonyan, Vedaldi, and Zisserman)

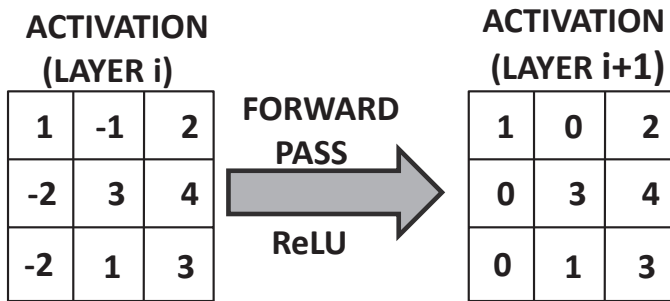
## Getting Cleaner Visualizations

- The idea of “deconvnet” is sometimes used for cleaner visualizations.
- Main difference is in terms of how ReLU’s are treated.
  - Normal backpropagation passes on gradient through ReLU when *input* is positive.
  - “Deconvnet” passes on gradient through ReLU when *backpropagated gradient* is positive.

## Guided Backpropagation

- A variation of backpropagation, referred to as *guided backpropagation* is more useful for visualization.
  - Guided backpropagation is a combination of gradient-based visualization and “deconvnet.”
  - Set an entry to zero, if either of these rules sets an entry to zero.

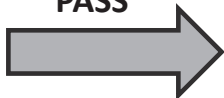
# Illustration of “Deconvnet” and Guided Backpropagation



“GRADIENTS”  
(LAYER i+1)

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

BACKWARD PASS



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

TRADITIONAL  
BACKPROPAGATION

“GRADIENTS”  
(LAYER i)

0	2	0
0	2	2
1	2	0

“DECONVNET”  
(APPLY ReLU)

0	0	0
0	2	2
0	2	0

GUIDED  
BACKPROPAGATION

# Derivatives of Features with Respect to Input Pixels

deconv



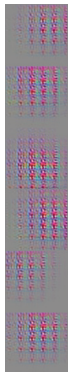
guided backpropagation



corresponding image crops



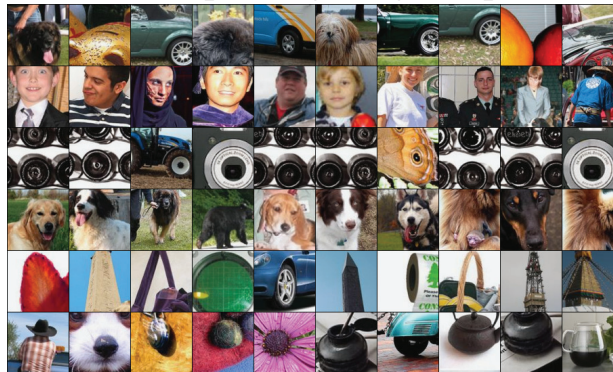
deconv



guided backpropagation



corresponding image crops



- ©2015 Springenberg, Dosovitskiy, Brox, Riedmiller

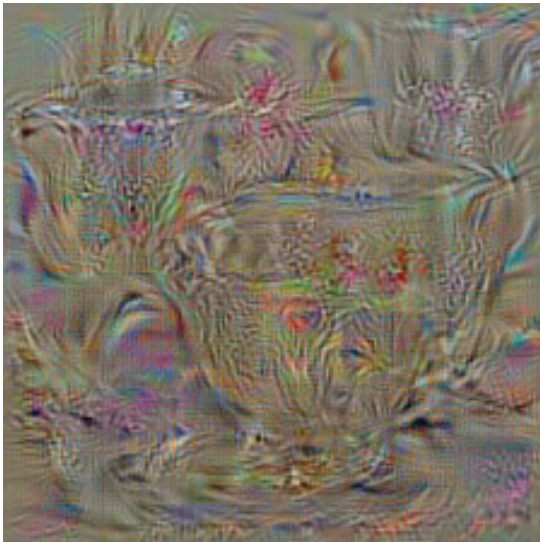
## Creating a fantasy image that matches a label

- The value of  $o$  might be the unnormalized score for “*banana*.”
- We would like to learn the input image  $\bar{x}$  that maximizes the output  $o$ , while applying regularization to  $\bar{x}$ :

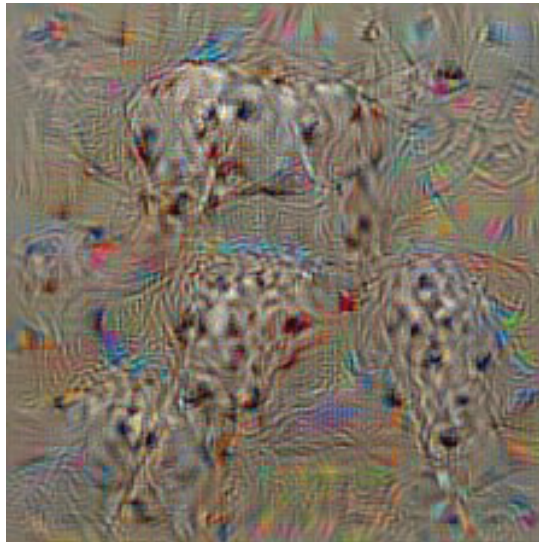
$$\text{Maximize}_{\bar{x}} J(\bar{x}) = (o - \lambda \|\bar{x}\|^2)$$

- Here,  $\lambda$  is the regularization parameter.
- Update  $\bar{x}$  while keeping weights fixed!

## Examples



**cup**



**dalmatian**



**goose**

## Generalization to Autoencoders

- Ideas have been generalized to convolutional autoencoders.
- Deconvolution operation similar to backpropagation.
- One can combine pretraining with backpropagation.

Charu C. Aggarwal  
IBM T J Watson Research Center  
Yorktown Heights, NY

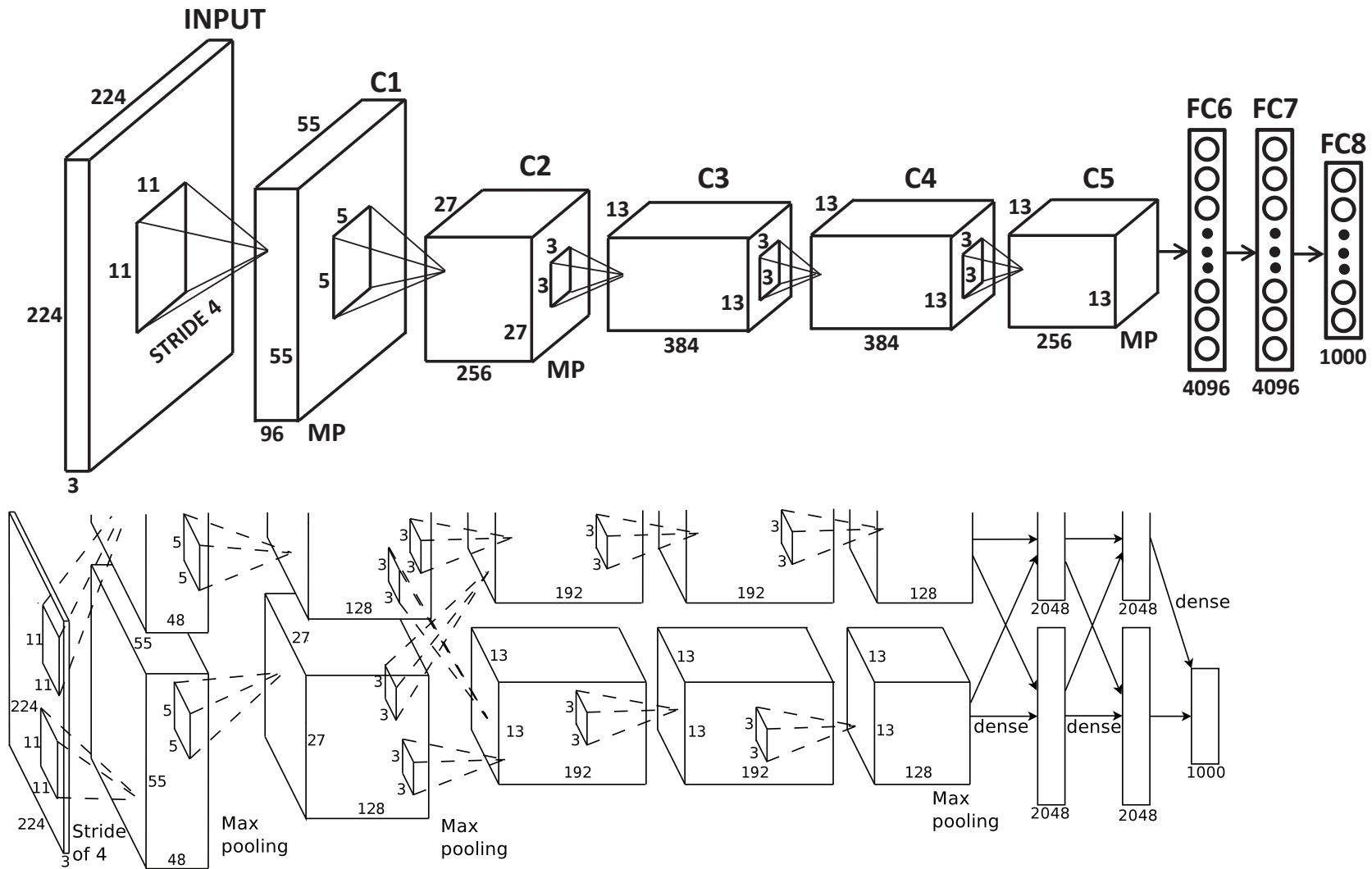
# Case Studies of Convolutional Neural Networks

Neural Networks and Deep Learning, Springer, 2018  
Chapter 8.4

## AlexNet

- Winner of the 2012 ILSVRC contest
  - Brought attention to the area of deep learning
- First use of ReLU
- Use Dropout with probability 0.5
- Use of  $3 \times 3$  pools at stride 2
- Heavy data augmentation

# AlexNet Architecture



## Other Comments on AlexNet

- Popularized the notion of FC7 features
- The features in the penultimate layer are often extracted and used for various applications
- Pretrained versions of AlexNet are available in most deep learning frameworks.
- Single network had top-5 error rate of 18.2%
- Ensemble of seven CNN had top-5 error rate of 15.4%

## ZfNet

	<i>AlexNet</i>	<i>ZFNet</i>
Volume:	$224 \times 224 \times 3$	$224 \times 224 \times 3$
Operations:	Conv $11 \times 11$ (stride 4)	Conv $7 \times 7$ (stride 2), MP
Volume:	$55 \times 55 \times 96$	$55 \times 55 \times 96$
Operations:	Conv $5 \times 5$ , MP	Conv $5 \times 5$ (stride 2), MP
Volume:	$27 \times 27 \times 256$	$13 \times 13 \times 256$
Operations:	Conv $3 \times 3$ , MP	Conv $3 \times 3$
Volume:	$13 \times 13 \times 384$	$13 \times 13 \times 512$
Operations:	Conv $3 \times 3$	Conv $3 \times 3$
Volume:	$13 \times 13 \times 384$	$13 \times 13 \times 1024$
Operations:	Conv $3 \times 3$	Conv $3 \times 3$
Volume:	$13 \times 13 \times 256$	$13 \times 13 \times 512$
Operations:	MP, Fully connect	MP, Fully connect
FC6:	4096	4096
Operations:	Fully connect	Fully connect
FC7:	4096	4096
Operations:	Fully connect	Fully connect
FC8:	1000	1000
Operations:	Softmax	Softmax

- AlexNet Variation  $\Rightarrow$  (Clarifai) won 2013 (14.8%/11.1%)

## VGG

- One of the top entries in 2014 (but not winner).
- Notable for its design principle of reduced filter size and increased depth
- All filters had spatial footprint of  $3 \times 3$  and padding of 1
- Maxpooling was done using  $2 \times 2$  regions at stride 2
- Experimented with a variety of configurations between 11 and 19 layers

## Principle of Reduced Filter Size

- A single  $7 \times 7$  filter will have 49 parameters over one channel.
- Three  $3 \times 3$  filters will have a receptive field of size  $7 \times 7$ , but will have only 27 parameters.
- **Regularization advantage:** A single filter will capture only primitive features but three successive filters will capture more complex features.

# VGG Configurations

Name:	A	A-LRN	B	C	D	E
# Layers	11	11	13	16	16	19
	C3D64	C3D64	C3D64	C3D64	C3D64	C3D64
		LRN	C3D64	C3D64	C3D64	C3D64
	M	M	M	M	M	M
	C3D128	C3D128	C3D128	C3D128	C3D128	C3D128
			C3D128	C3D128	C3D128	C3D128
	M	M	M	M	M	M
	C3D256	C3D256	C3D256	C3D256	C3D256	C3D256
	C3D256	C3D256	C3D256	C3D256	C3D256	C3D256
				C1D256	C3D256	C3D256
						C3D256
	M	M	M	M	M	M
	C3D512	C3D512	C3D512	C3D512	C3D512	C3D512
	C3D512	C3D512	C3D512	C3D512	C3D512	C3D512
				C1D512	C3D512	C3D512
						C3D512
	M	M	M	M	M	M
	C3D512	C3D512	C3D512	C3D512	C3D512	C3D512
	C3D512	C3D512	C3D512	C3D512	C3D512	C3D512
				C1D512	C3D512	C3D512
						C3D512
	M	M	M	M	M	M
	FC4096	FC4096	FC4096	FC4096	FC4096	FC4096
	FC4096	FC4096	FC4096	FC4096	FC4096	FC4096
	FC1000	FC1000	FC1000	FC1000	FC1000	FC1000
	S	S	S	S	S	S

## VGG Design Choices and Performance

- Max-pooling had the responsibility of reducing spatial footprint.
- Number of filters often increased by 2 after each max-pooling
  - Volume remained roughly constant.
- VGG had top-5 error rate of 7.3%
- Column D was the best architecture [previous slide]

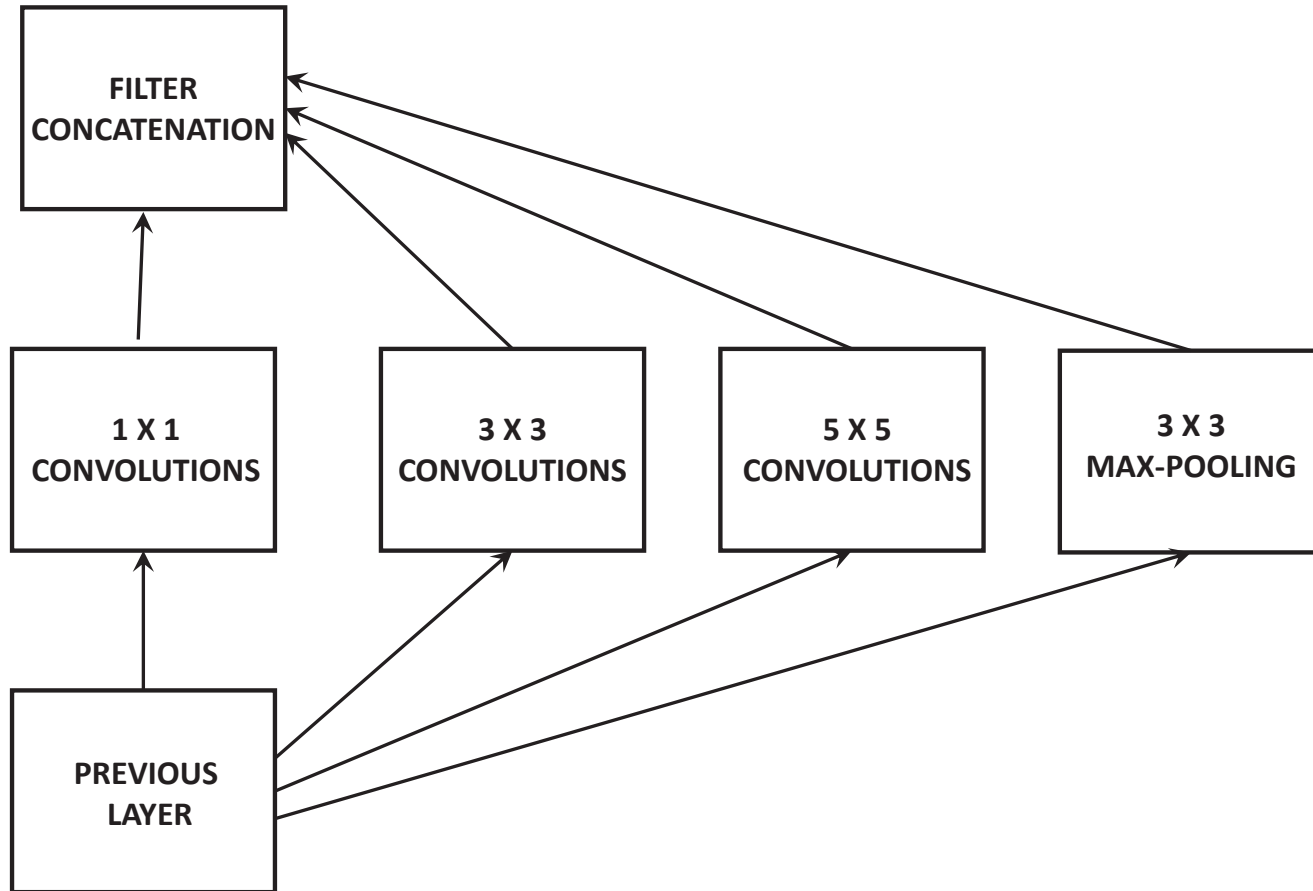
## GoogLeNet

- Introduced the principle of *inception architecture*.
- The initial part of the architecture is much like a traditional convolutional network, and is referred to as the *stem*.
- The key part of the network is an intermediate layer, referred to as an *inception module*.
- Allows us to capture images at varying levels of detail in different portions of the network.

## Inception Module: Motivation

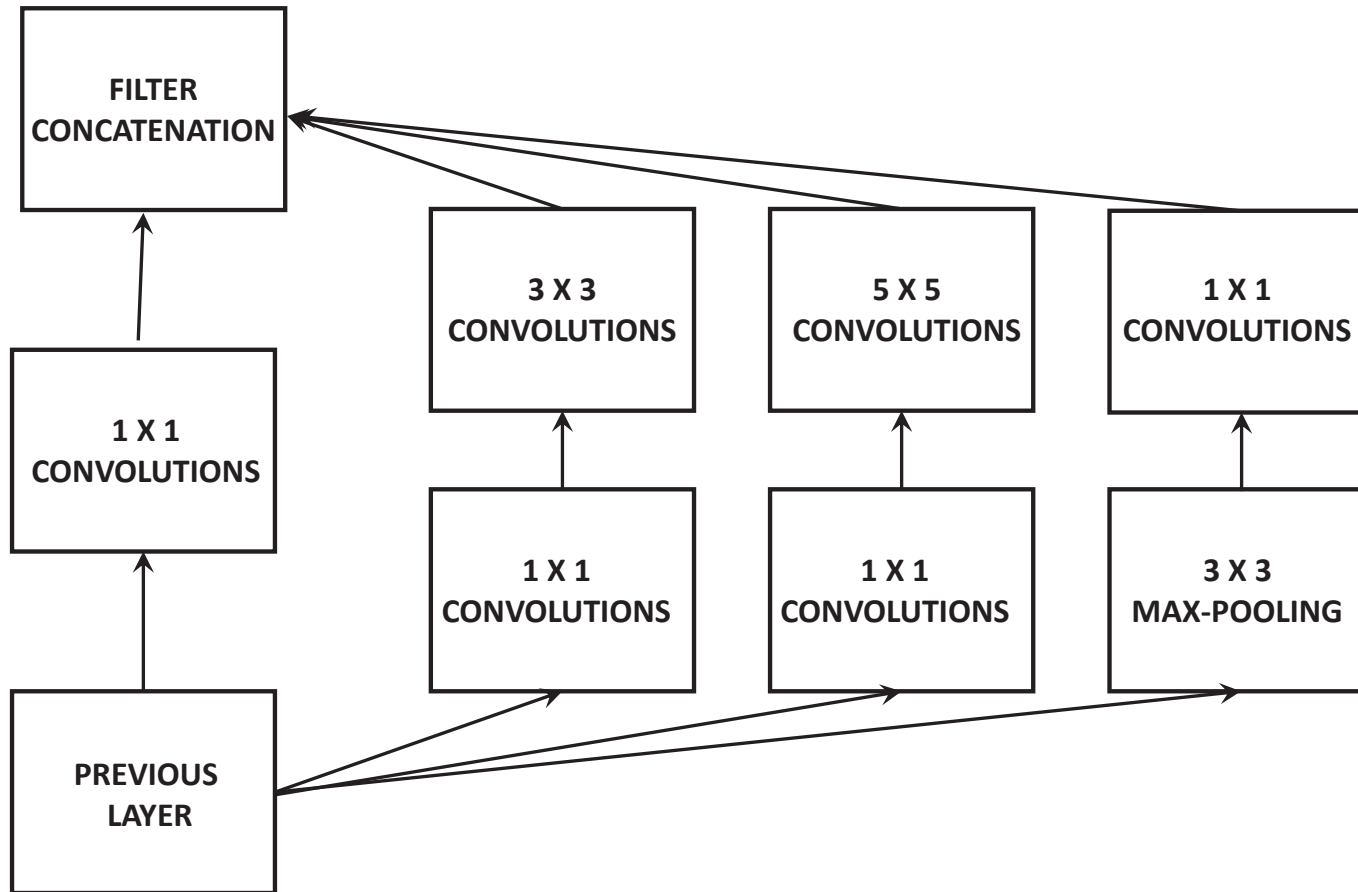
- Key information in the images is available at different levels of detail.
  - Large filter can capture information in a bigger area containing limited variation.
  - Small filter can capture detailed information in a smaller area.
- Piping together many small filters is wasteful  $\Rightarrow$  Why not let the neural network decide?

## Basic Inception Module



- Main problem is computational efficiency  $\Rightarrow$  First reduce depth

## Computationally Efficient Inception Module

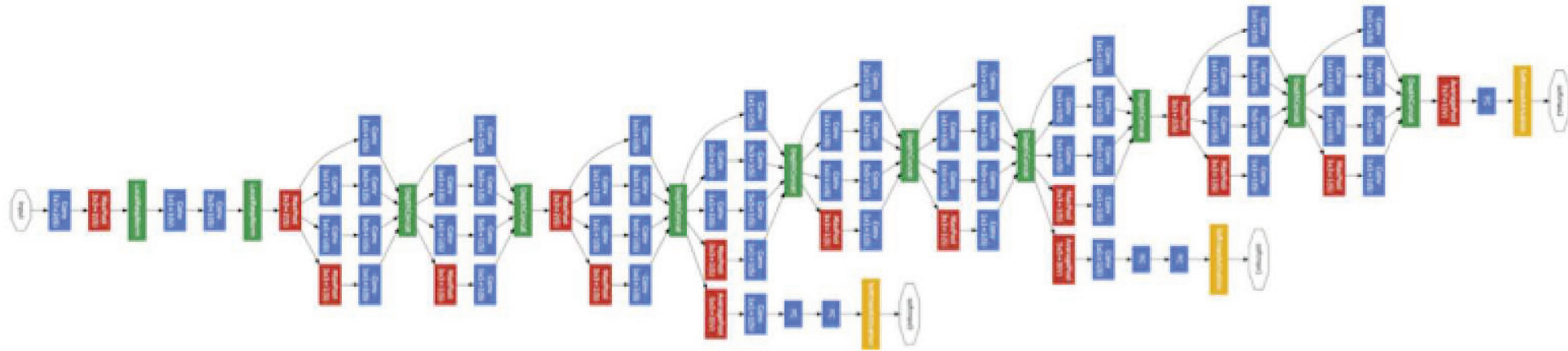


- Note the  $1 \times 1$  filters

## Design Principles of Output Layer

- It is common to use fully connected layers near the output.
- *GoogLeNet* uses average pooling across the whole spatial area of the final set of activation maps to create a single value.
- The number of features created in the final layer will be exactly equal to the number of filters.
  - Helps in reducing parameter footprint
- Detailed connectivity is not required for applications in which only a class label needs to be predicted.

# GoogLeNet Architecture



**Convolution**  
**Pooling**  
**Softmax**  
**Other**

## Other Details on GoogLeNet

- Winner of ILSVRC 2014
- Reached top-5 error rate of 6.7%
- Contained 22 layers
- Several advanced variants with improved accuracy.
- Some have been combined with ResNet

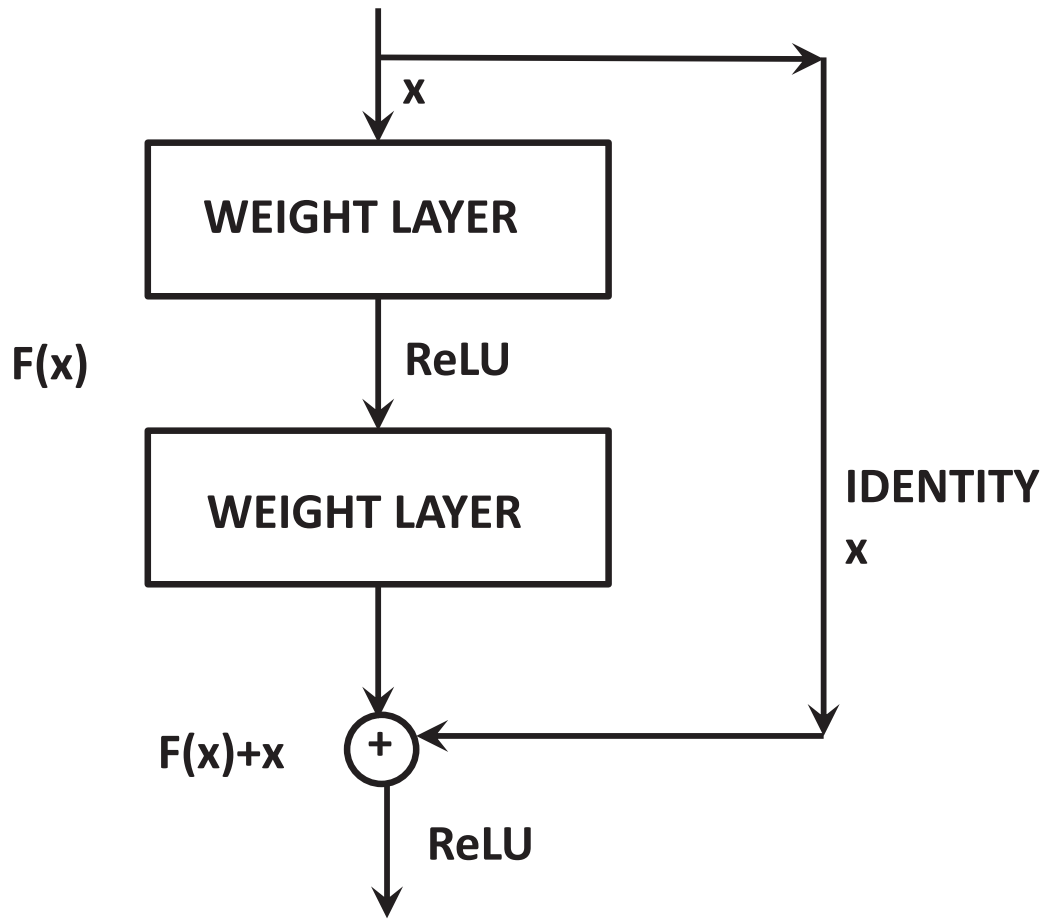
## ResNet Motivation

- Increasing depth has advantages but also makes the network harder to train.
- Even the error on the training data is high!
  - Poor convergence
- Selecting an architecture with unimpeded gradient flows is helpful!

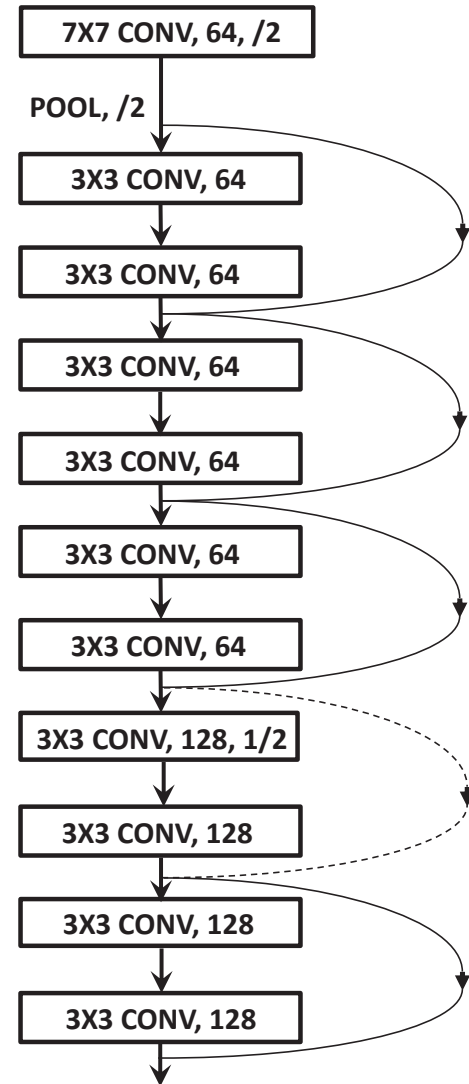
## ResNet

- ResNet brought the depth of neural networks into the hundreds.
- Based on the principle of iterative feature engineering rather than hierarchical feature engineering.
- Principle of partially copying features across layers.
  - Where have we heard this before?
- Human-level performance with top-5 error rate of 3.6%.

# Skip Connections in ResNet



(a) Residual module



(b) Partial architecture

## Details of ResNet

- A  $3 \times 3$  filter is used at a stride/padding of 1  $\Rightarrow$  Adopted from *VGG* and maintains dimensionality.
- Some layers use strided convolutions to reduce each spatial dimension by a factor of 2.
  - A linear projection matrix reduces the dimensionality.
  - The projection matrix defines a set of  $1 \times 1$  convolution operations with stride of 2.

## Importance of Depth

Name	Year	Number of Layers	Top-5 Error
-	Before 2012	$\leq 5$	$> 25\%$
<i>AlexNet</i>	2012	8	15.4%
<i>ZfNet/Clarifai</i>	2013	8/ $> 8$	14.8% / 11.1%
<i>VGG</i>	2014	19	7.3%
<i>GoogLeNet</i>	2014	22	6.7%
<i>ResNet</i>	2015	152	3.6%

## Pretrained Models

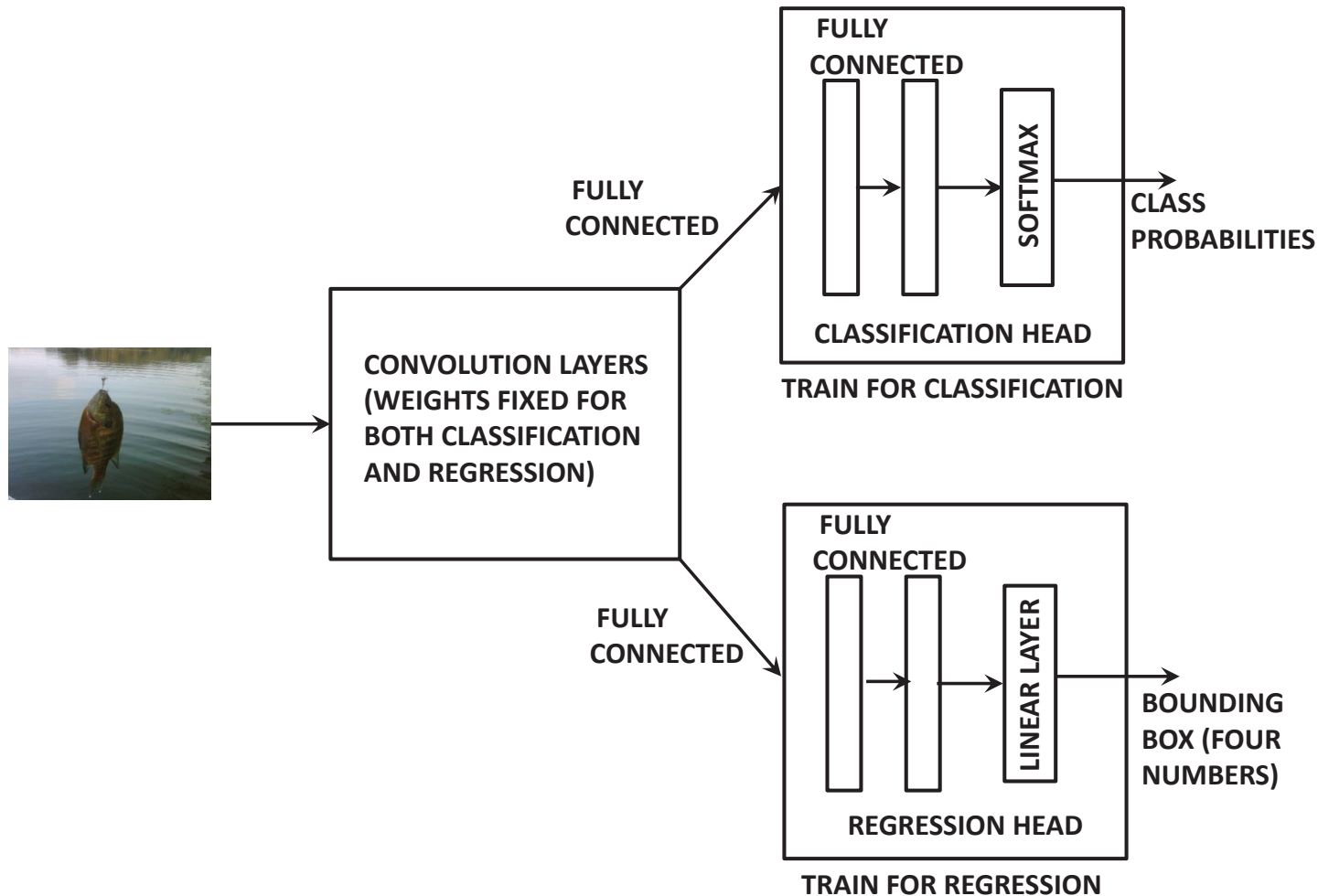
- Many of the models discussed in previous slides are available as pre-trained models with *ImageNet* data.
- One can use the features from fully connected layers for nearest neighbor search.
  - A nearest neighbor classifier on raw pixels vs features from fully connected layers will show huge differences.
- One can train output layer for other applications like regression while retaining weights in other layers.

## Object Localization



- Need to classify image together with bounding box (four numbers)

# Object Localization



- Fully connected layers for classification and regression heads trained separately.

## Other Applications

- Object detection: Multiple objects
- Text and sequence processing applications
  - 1-dimensional convolutions
- Video and spatiotemporal data
  - 3-dimensional convolutions