Department of Computational and Data Sciences

INDIAN INSTITUTE OF SCIENCE

भारतीय विज्ञान संस्थान

Computer Vision

Deepak Subramani

Assistant Professor

Dept. of Computational and Data Science

Indian Institute of Science Bengaluru

Lecture and Assignment Guide

- This Slide Deck has Material for 6 hours of teaching divided into Parts 1-6
- We will go through
  - Week 01
    - Part 01 - Convolutional and Pooling Layers; AST 01
    - Part 02 - Transfer Learning and Modern CV Design Principle; AST 02
  - Week 02
    - Part 01 - Modern Convolutional Building Blocks for Image Classification; AST 03
    - Part 02 - Object Localization
    - Interpreting what convolutions learn (Advanced topic) – AST 03
  - Week 03
    - Part 01 - Object Detection (YOLO), Image Segmentation – Lec 05
    - Part 02 - Practical CVOps
    - AST04 – Object Detection with YOLO
  - Week 04
    - Revision
    - AST05 – Image Segmentation
- Additional Reading material to go in depth of math with references and code references are provided with the marking of “Additional Material” or “Additional Discussion” etc

Department of Computational and Data Sciences

भारतीय विज्ञान संस्थान

CV Week 01 Part 01

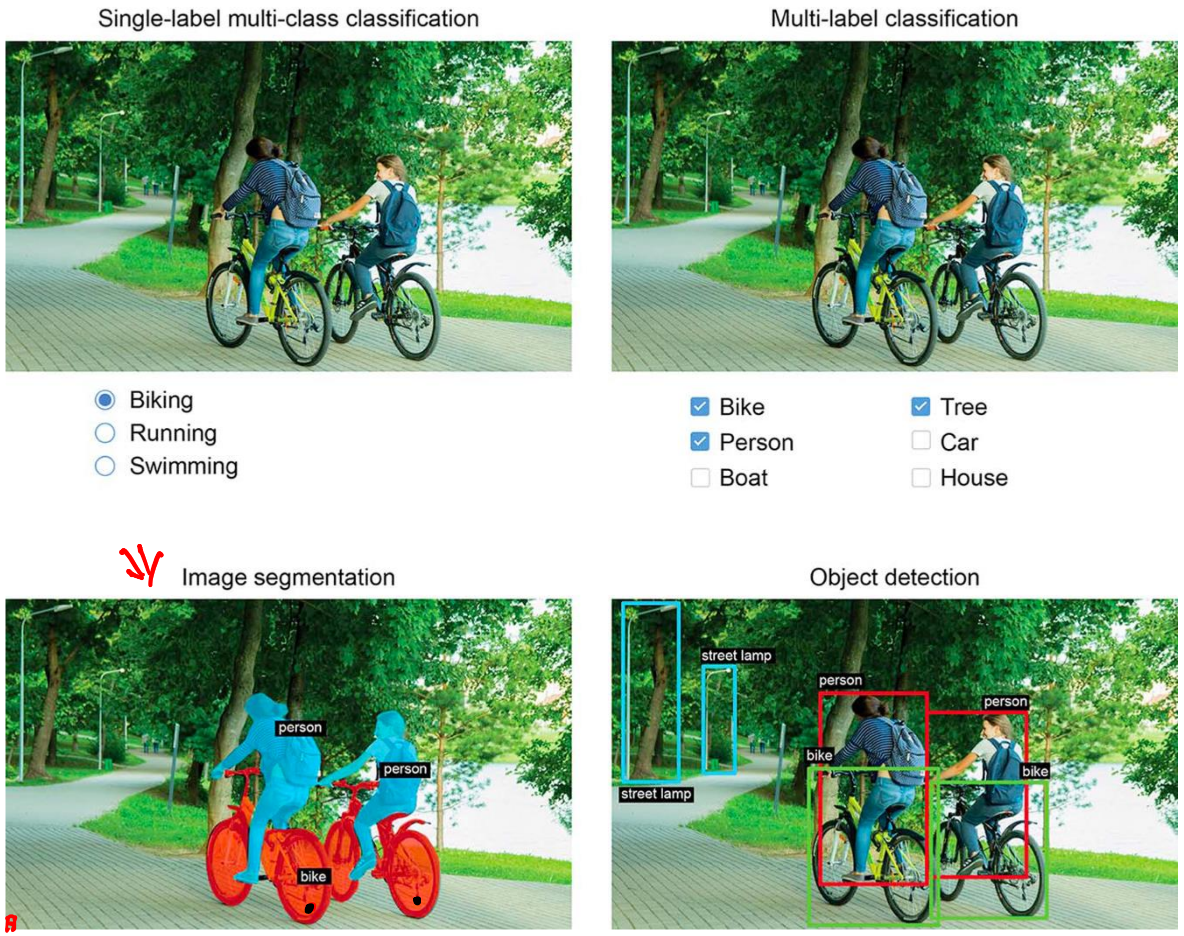
Deepak Subramani
Assistant Professor
Dept. of Computational and Data Science
Indian Institute of Science Bengaluru

Pre-Poll Survey ;)

1. What is semantic segmentation?
    a. Image level classification
    b. Pixel level classification
    c. Identifying a box around objects in an image and performing classification → Object Detection
    d. None of the above
2. What is the neural layer most needed for computer vision tasks?
    a. Dense layers
    b. Convolutional layers
    c. Recurrent layers
    d. None of the above

Three Essential Tasks in Computer Vision

- Image Classification
  - Single Label
    - Binary
    - Multiclass
  - Multi Label
- Image Segmentation
  - Pixel wise identify the class
  - Example: Zoom background replacement
- Object Detection
  - Bounding box around objects
  - Self-driving cars, face detection in cameras

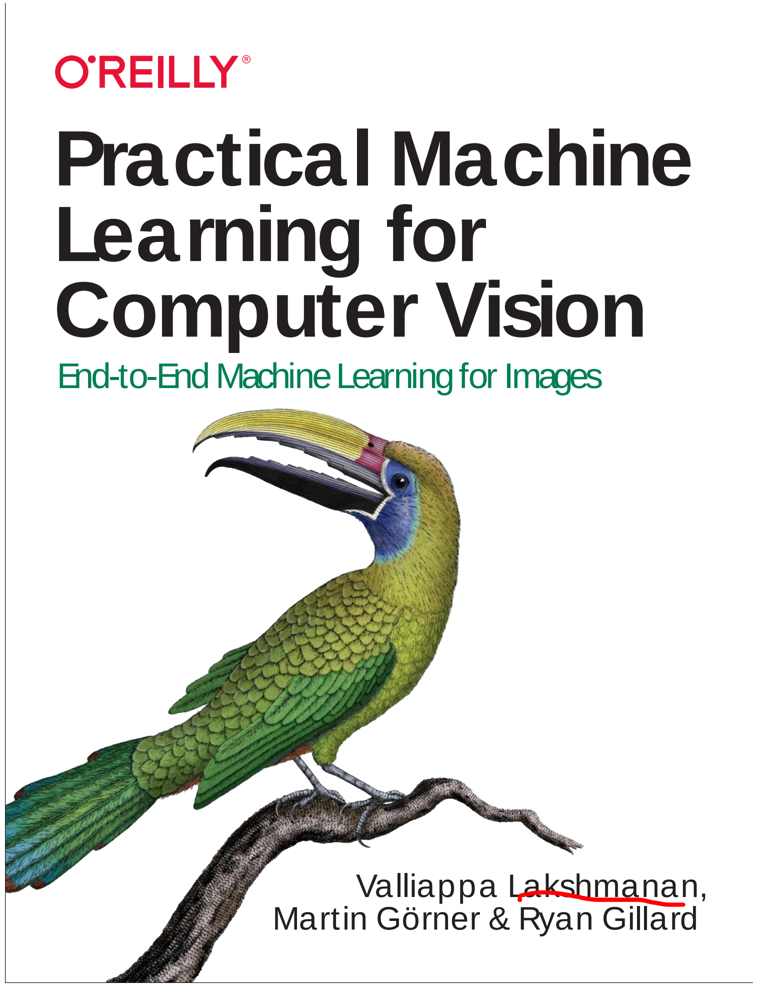


person
bike
background

Department of Computational and Data Sciences

भारतीय विज्ञान संस्थान

Another Textbook

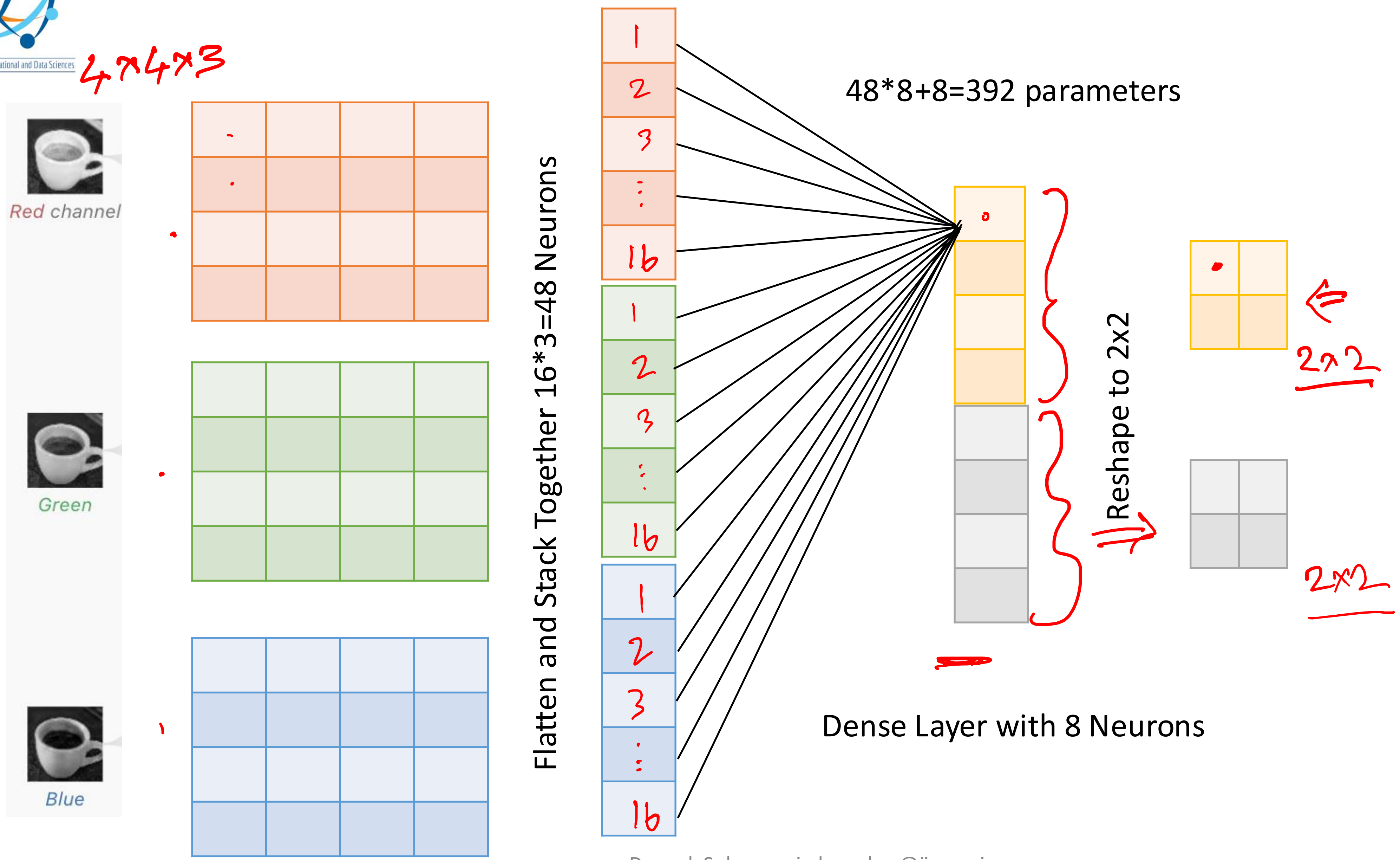


Geron

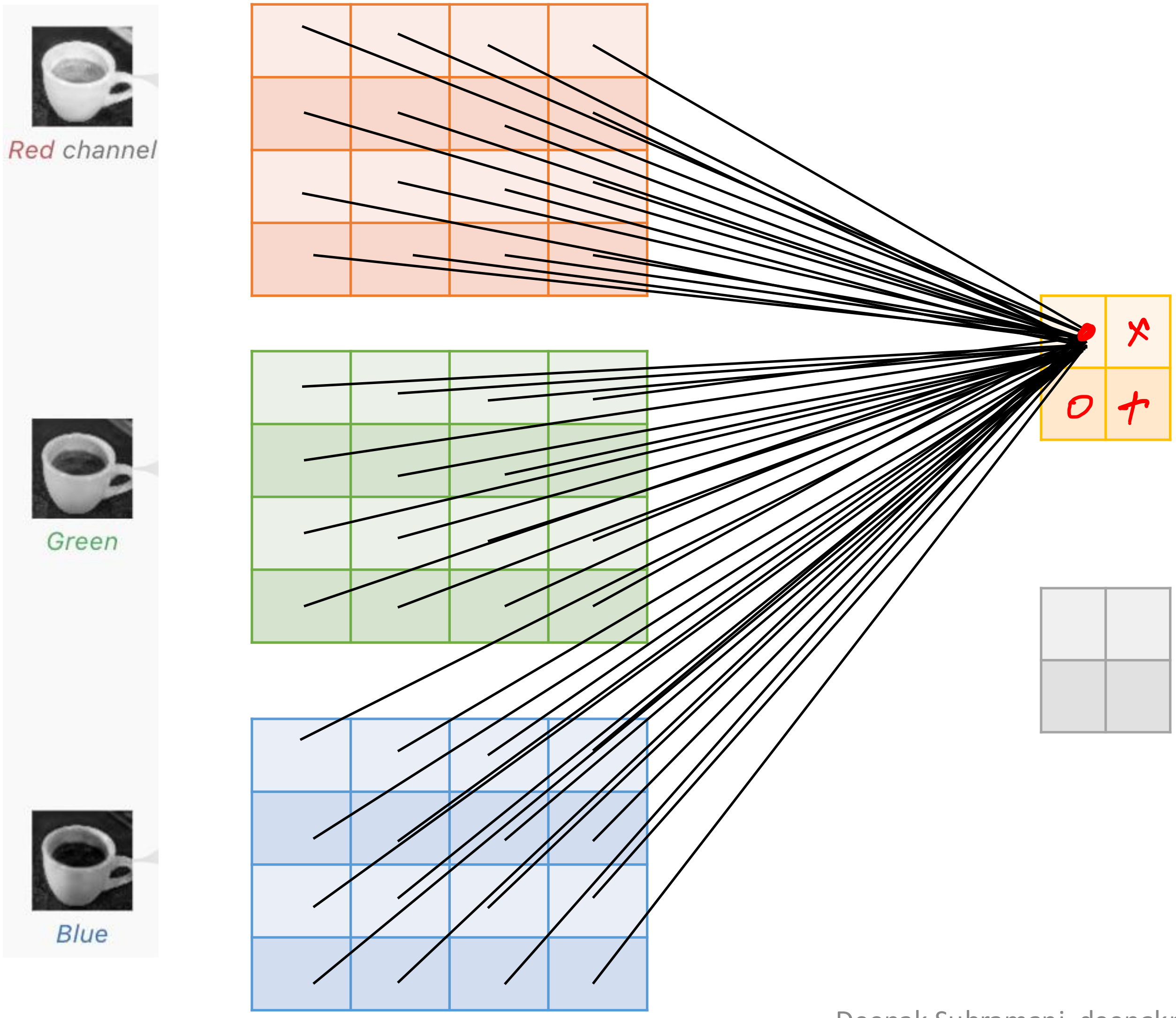
Chollet

→ CVops

Neuron Arrangement in Dense Layer



Ugly Figure for Dense



Visual Cortex: Biological Inspiration to Modern Architectures

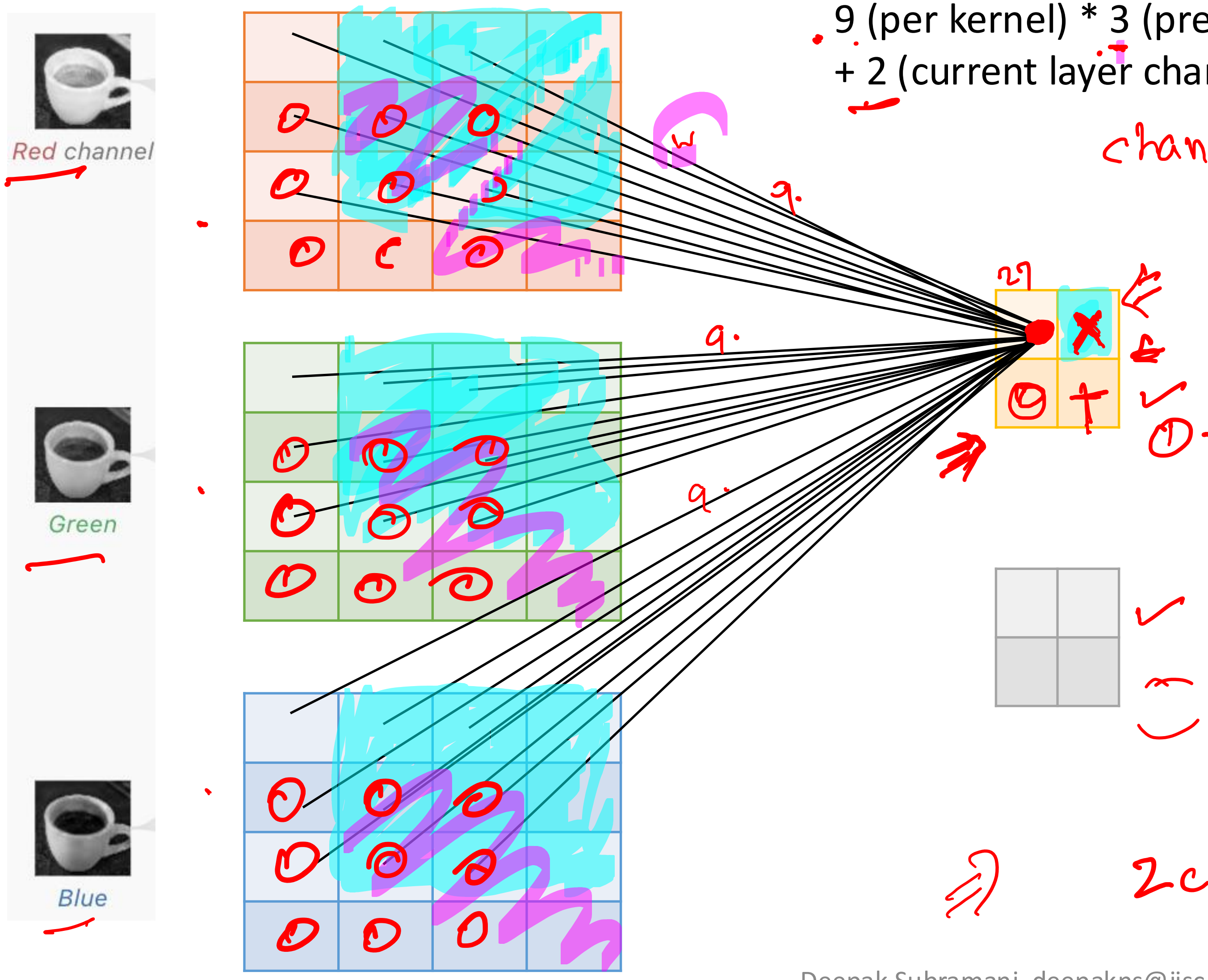
- Hubel and Wiesel (Nobel Prize in Physiology/Medicine in 1981) for their 1958/59 work on understanding the visual cortex through experiments on cats
- Key insight – local receptive field
  - Neurons in the visual cortex react to stimuli only in a limited region of the receptive field
  - Some neurons have larger receptive fields that react to complex patterns formed by a combination of lower-level patterns

Neuron Arrangement in Conv Layer with 3x3 Kernel

9

9 (per kernel) * 3 (previous layer channel) * 2 (current layer channels) + 2 (current layer channel) = 56 parameters

channel / Filter / Feature map



One filter is

3 x 3 x num_channels_prev_layer

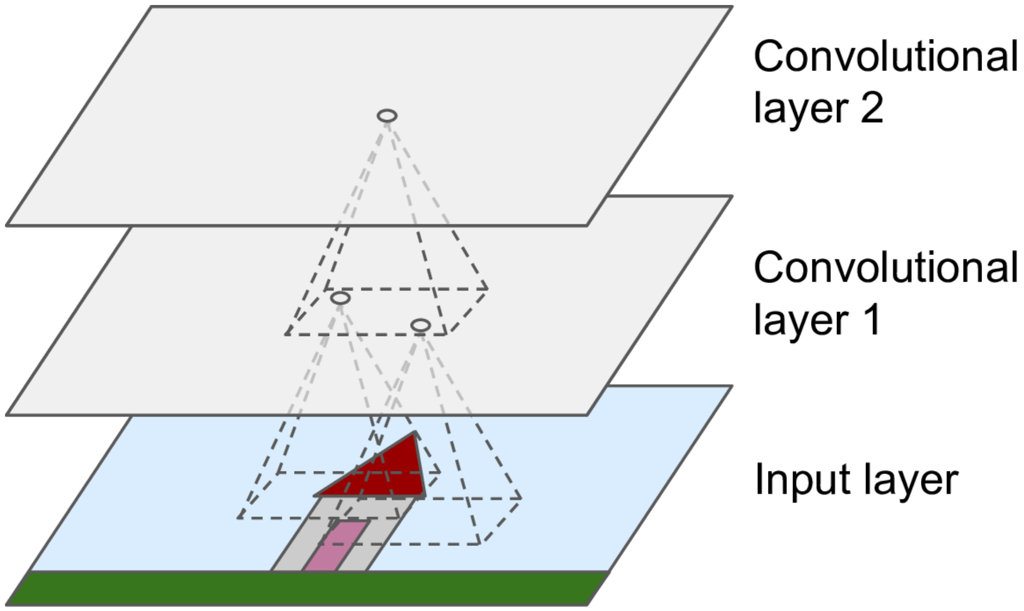
A channel in the current layer has only one filter

All neurons in a channel of the current layer shares weights and bias

2 channels

Convolutional Layer

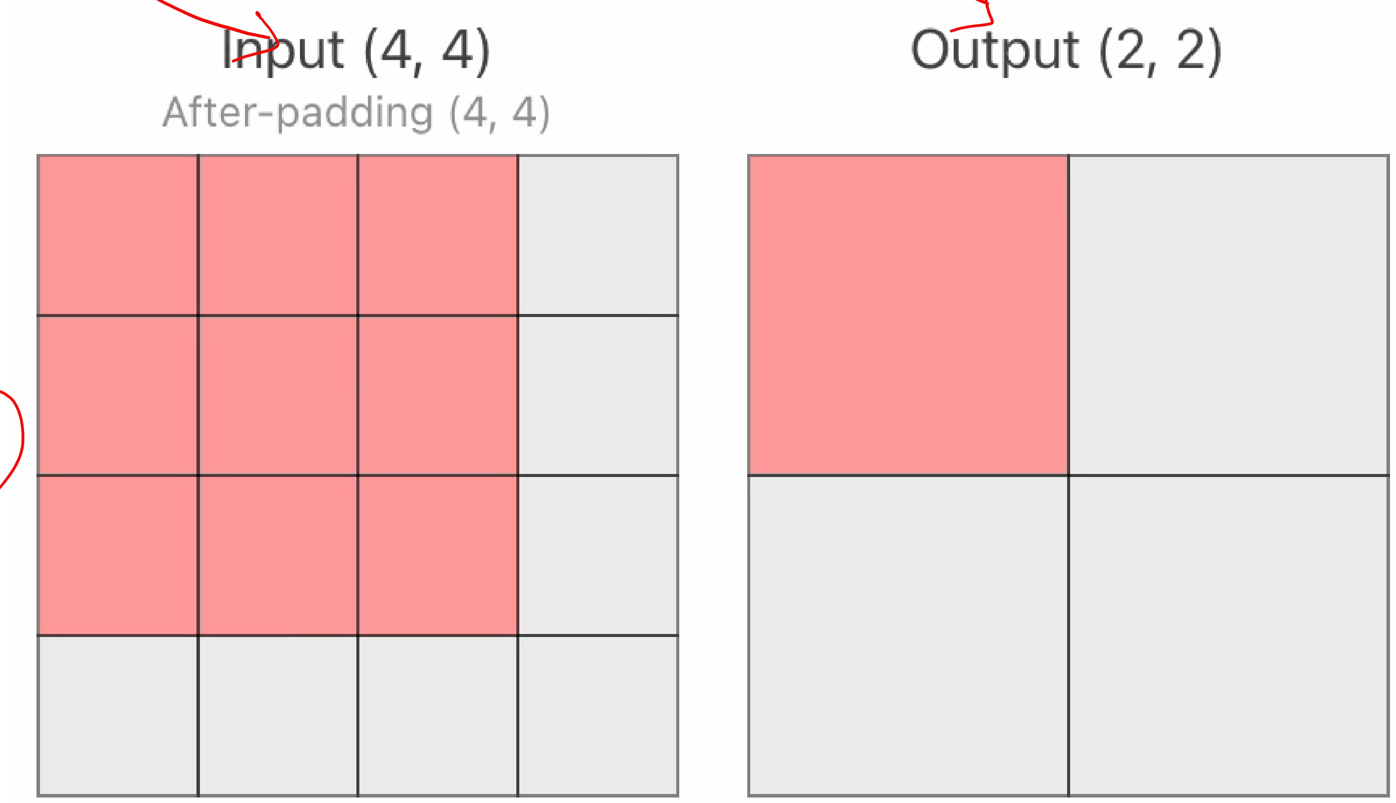
- Convolutional Layer
    - Neurons have connections to only a limited receptive field in the previous layer
    - Neurons in each layer are represented in 2D making visualization of connections easy



[Geron Fig 14-2]

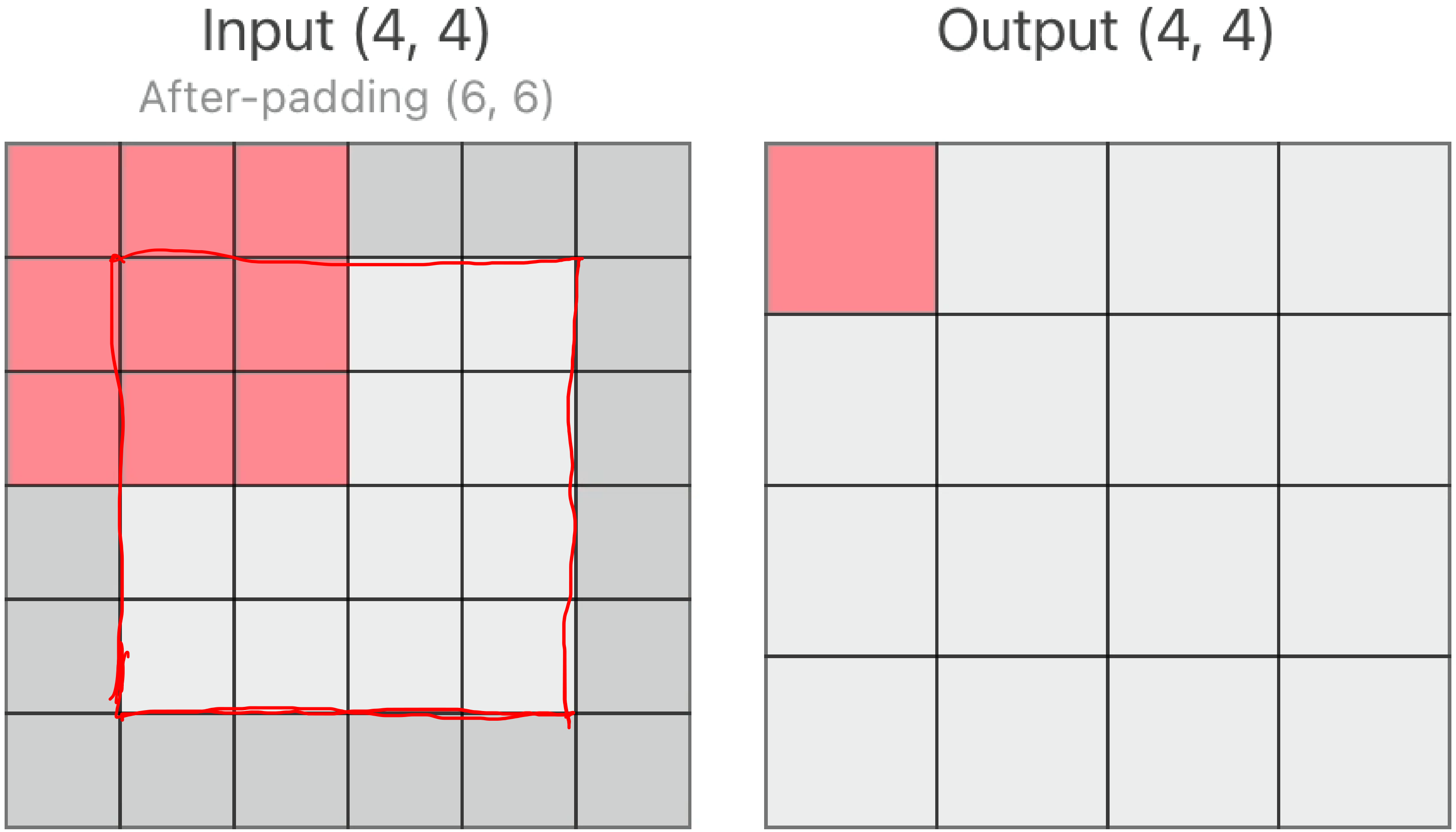
Convolutional Kernels or Filters

- Each neuron is connected to only a limited rectangular area in the previous layer
- The weights of these connections are in the form of a convolutional kernel
- The output is obtained by multiplying inputs in that receptive field with the kernel and adding them together



https://poloclub.github.io/cnn-explainer/

Convolutional Kernels with Padding



https://poloclub.github.io/cnn-explainer/

Department of Computational and Data Sciences

Convolutional Calculation By Hand

INDIAN INSTITUTE OF SCIENCE
भारतीय विज्ञान संस्थान

4×4

Input

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

Kernel

0	-1	0
-1	0	1
0	1	0

b=0

Feature Map

(0) 0	(-1) 0	(0) 0	0	0	0
(-1) 0	(0) 1	(1) 1	1	0	0
(0) 0	(1) 0	(0) 0	0	1	0
0	0	1	0	1	0
0	1	0	0	1	0
0	0	0	0	0	0

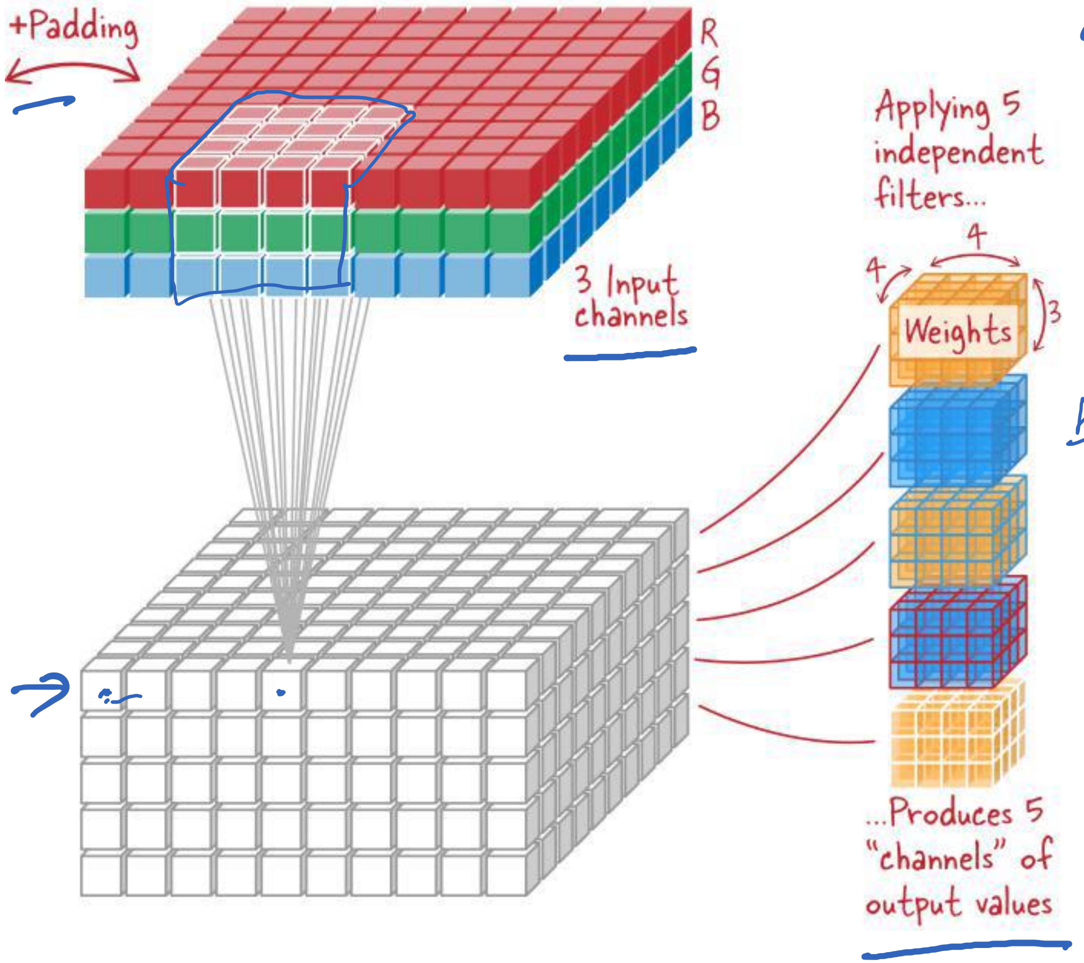
Out = 0*0 + -1*0 + 0*0 +
-1*0 + 0*1 + 1*1 +
0*0 + 1*0 + 0*0 + 0
= 1

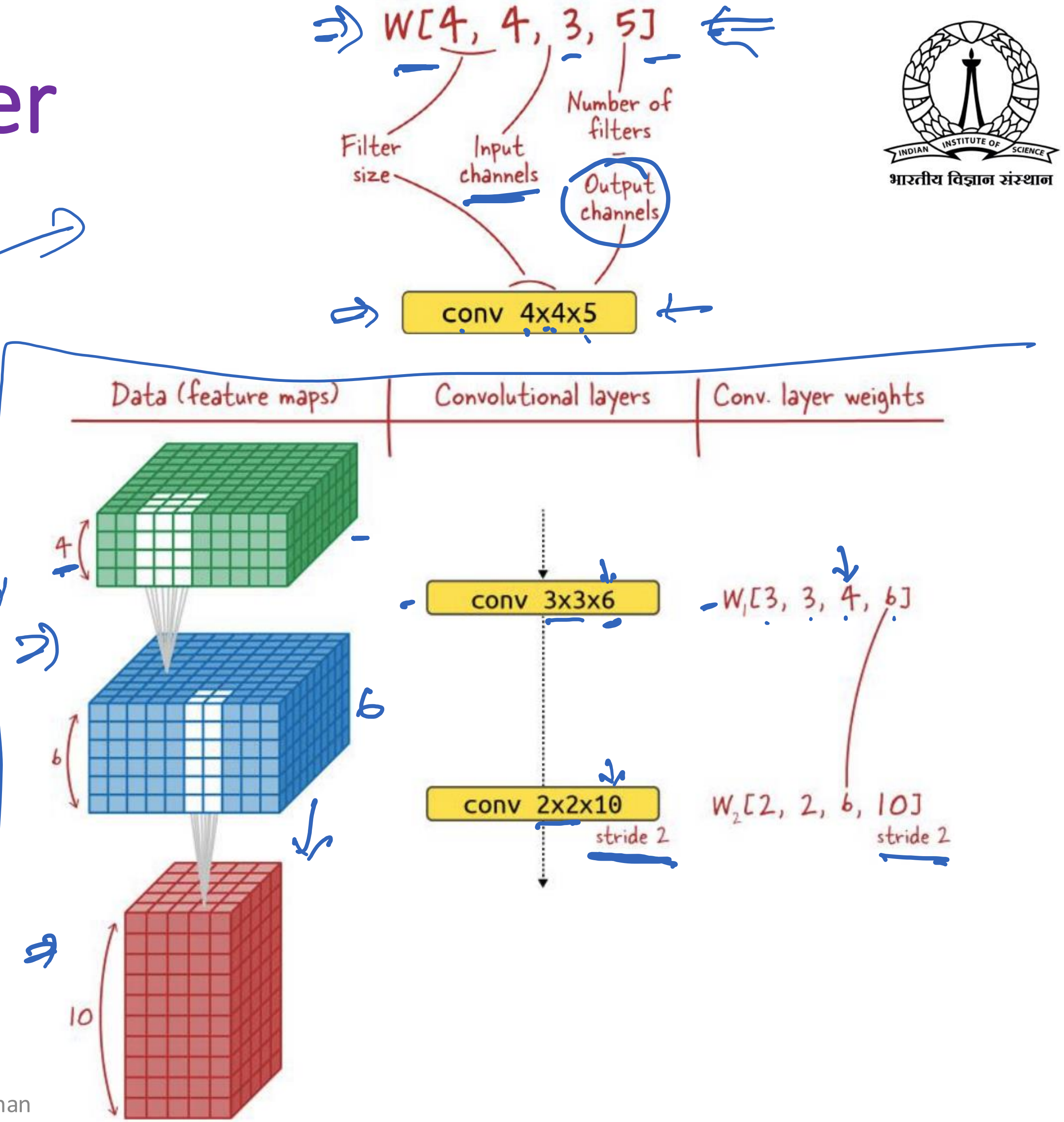
1	0	-1	-

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

0×0 + -1×0 + 0×0
+ -1×1 + 0×1 + 1×1
+ 0×0 + 1×0 + 0×0
+b
= 0

Convolutional Layer



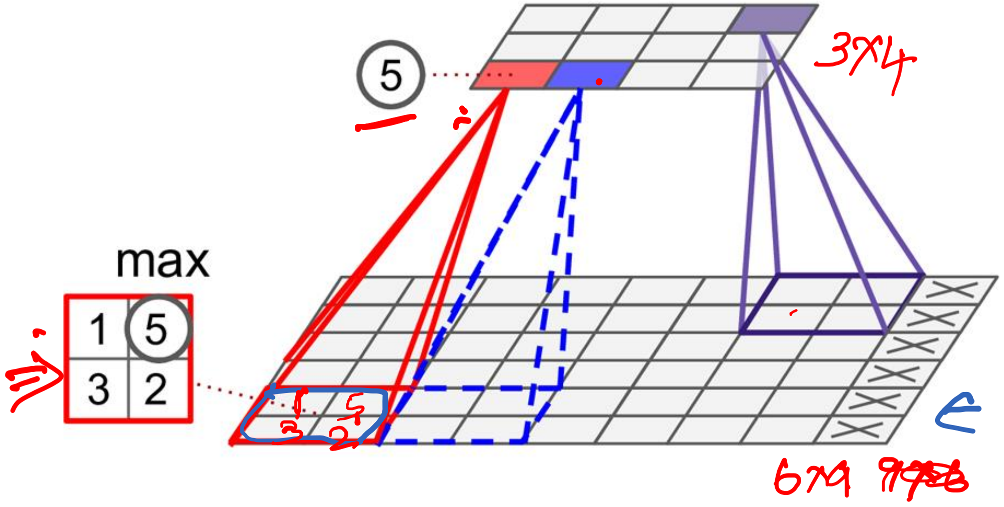


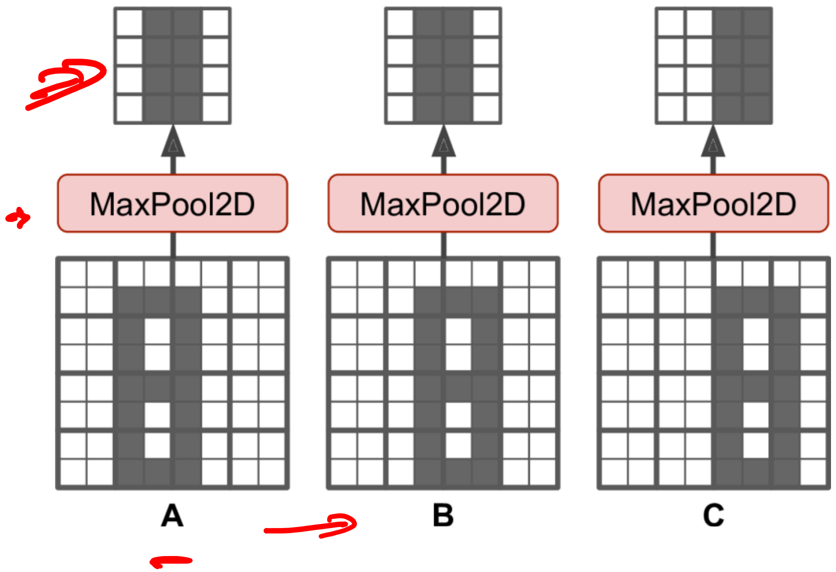
TensorFlow Implementation

- Image – 3D Tensor: [height, width, channels]
- Mini-Batch – 4D Tensor: [samples, height, width, channels]
- Weight of a convolutional layer – 4D Tensor
  - [Height of kernel, width of kernel, number of feature maps in previous layer, number of filters (feature maps) in current layer]
- Bias of a convolutional layer – 1D Tensor
  - [number of filters (feature maps) in current layer]
- tf.nn.conv2d(images, filters, strides, padding) – base implementation in TF
- Keras: keras.layers.Conv2D(filters, kernel_size, strides, padding, activation)
  - Use these layers in Sequential, Functional or Subclass API just as we used a dense layer – Simple!

Pooling Layers

- A pooling kernel is used (say 2x2)
- The maximum pixel value among the image within the pooling kernel is chosen as the output – Top figure
- Advantages:
  - Introduces invariance to small translation (Bottom Figure)
- Disadvantages:
  - Very destructive – a 2x2 pooling kernel drops 75% of the information
  - Invariance is not desirable in some application like semantic segmentation
- Keras: keras.layers.MaxPool2D(pool_size=2)
- Keras: keras.layers.AvgPool2D(pool_size=2)
  - Instead of max, choose the average





Worked Out Example

- How many parameters have to be learnt for a convolutional layer with 128 filters, acting on a previous layer with dimensions 14x14x64. The kernel size is 3.
  - 73856
  - 73728
  - 24576
  - 1152

Poll

1. Which of the following is TRUE?
    a. Weights are shared between neurons in one filter of a convolutional layer
    b. Feature maps and filters in a convolutional layer are different
    c. Layer and filters are synonyms (same meaning)
    d. One feature map in a convolutional layer "looks at" only a small receptive field in <u>ONE</u> feature map of the previous layer

Concept List

- Neuron arrangement in convolutional layer
  - Kernel size, num of filters, stride , padding
  - Operations
- Pooling Layer
  - Operations

- Transfer Learning
- Modern pipeline → Demo

Poll

1. What is the loss function used for multi class classification?
   - a. Cross entropy ✓
   - b. MSE
   - c. Huber Loss ✓
   - d. Log Loss → B.C.E

2. What is the activation fn on the output layer for MCC?

   a) ReLU

   b) Sigmoid

   ✓ c) Softmax

   d) Linear

Department of Computational and Data Sciences

62x62x10 60x60x10 30x30x10 28x28x10 26x26x10

Conv1 Conv2 Max

10 10 10

64 x

64 x

3

13 x

13 x

10

1690

Flatten

10

Classification Head.

13x13x10 1x10

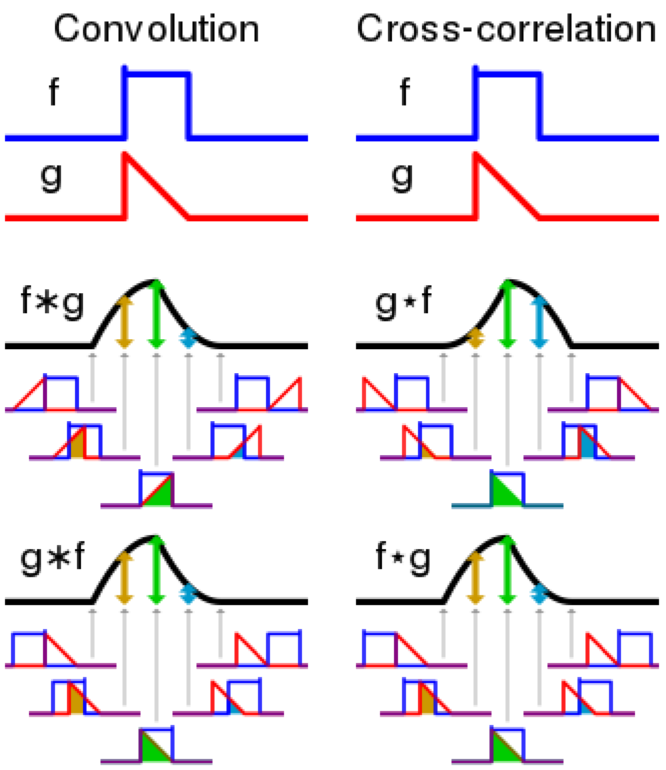
→ Global Max Pooling

Additional Discussion

- Convolution vs Correlation
- Kernel Math
- Visualization
- Depth Wise Max Pool
- Memory Issues

Convolution vs Cross-Correlation Function

- The filter operation that we saw is theoretically a correlation calculation, and not a convolution
- A true convolution needs the filter to be flipped
  - This flipping makes convolution commutative
  - Cross-Correlation (without filter flipping) is not commutative
  - But this does not have any effect on the training and the true convolution is needed only to write proofs
- Almost all libraries implement cross-correlation, but call it convolution



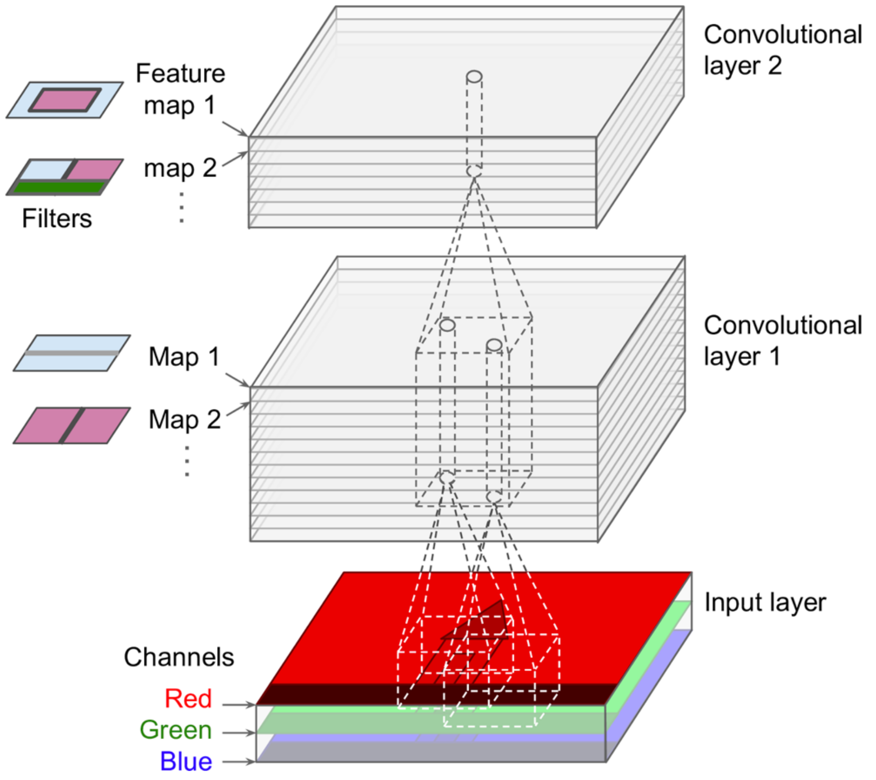
https://en.wikipedia.org/wiki/File:Comparison_convolution_correlation.svg

Kernel: Important Points

- Application of a kernel reduces the dimensions by one pixel on all image boundaries
- If you want the output to be the same size as the input, then padding is required
- Common padding is the zero padding (used in the prev example)
  - In Keras: padding="same" is the zero padding
  - In Keras: padding="valid" means no padding is applied
- The kernels may be non-square
- The kernels may be moved by a distance *stride, not necessarily equal to 1,* each time
  - Using a stride more than 1 reduces the dimensionality of the image
- Kernels are learned during training
- Feature Maps: The same kernel is used in one Convolutional Layer
  - This produces a feature map, a 2D layer
  - It reduces the number of parameters to learn
  - It also makes identifying the same object in different parts of the image easy

Stacking Multiple Feature Maps

- We can stack multiple filters together to produce a stack of feature maps
- Usually, an image has three channels RGB
  - Satellite images have more channels corresponding to the spectrum of the instruments
- A Convolutional Layer contains a stack of feature maps
- Each neuron in Map 1 of Conv Layer 1 is connected to the receptive field of all feature maps in the previous layer
- Each Neuron of Map *k* has the same weights in its connection to Map *k′* of the previous layer



Geron Fig 14-6

$$z_{i,j,k} = b_k + \sum_{u=0}^{f_h-1} \sum_{v=0}^{f_w-1} \sum_{k'=0}^{f_{n'}-1} x_{i',j',k'} \cdot w_{u,v,k',k} \quad \text{with} \begin{cases} i' = i \times s_h + u \\ j' = j \times s_w + v \end{cases}$$

Convolution and Pooling as a Strong Prior

- Consider learning weights as a Bayesian parameter estimation problem
  - A prior distribution of weights
  - An observation – error of a mini-batch
  - A posterior distribution of weights
  - Iterate and stop when posterior distribution stops shifting
  - Pick MAP estimate
- Initializing weights is providing a prior to it
  - Example HeNormal
- A weak prior gives more weightage to the observations
- A strong prior plays a more active role in final parameter determination
- Imagine a Convolutional Layer as being similar to a Dense Layer, but with an infinitely strong prior over its weights
  - Some weights (outside the kernel) are set to zero
  - And weights for a filter (feature map) are shared
- A pooling layer is also a strong prior on invariance

Convolution Computation Viz

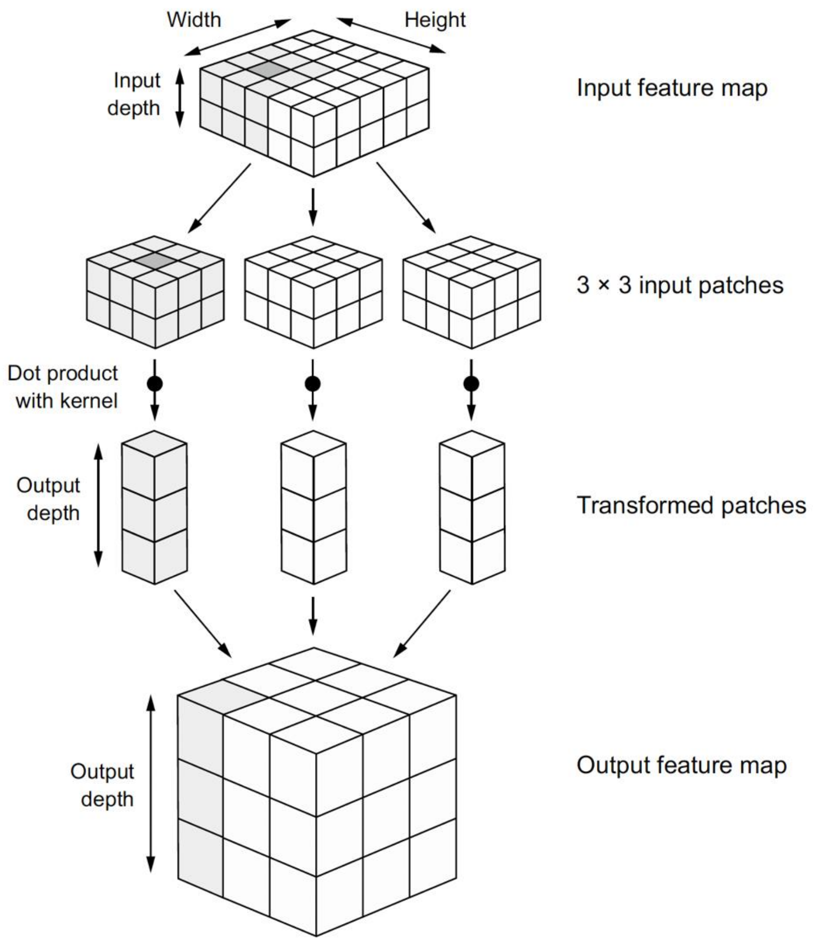


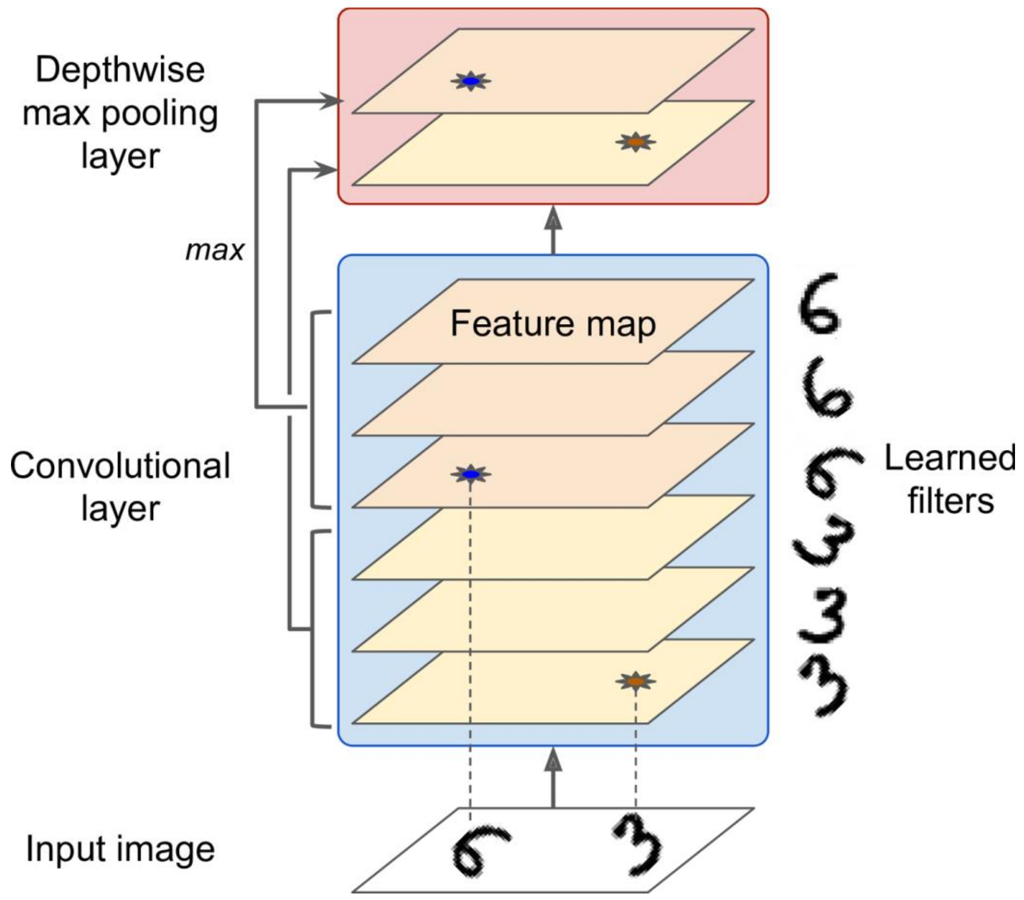
Figure 8.4 How convolution works

CNN Visualization

- https://poloclub.github.io/cnn-explainer/
- https://paperswithcode.com/sota/image-classification-on-imagenet

Depth wise maxpool

- A less common usage is to do Pooling in the depth (channel/filter) dimension instead of spatial dimension
- This makes the CNN be invariant to brightness, color, thickness, skew…
- Keras doesn't have a depth wise maxpool readymade layer
- Need to define your own layer (either a Lambda Layer or subclass the layer class)
  - Output = tf.nn.max_pool(images, ksize=(1,1,1,3),strides=(1,1,1,3), padding="valid")
- Global Average Pooling creates just one number per feature map
  - Used in ResNet architecture



Memory Issues

- CNNs need a lot of RAM during training
- Consider a small sized problem
  - 150x100 input with RGB channels
  - 5x5 filter outputting 200 feature maps
  - Total number of parameters = (5x5x3+1)x200 = 15,200 (+1 bias)
    - Compare to Fully Connected = 150^2 x 100^2 x 3 = 675 million
  - Each 200 feature map contains 150 x 100 (stride=1, padding=same)
  - Computations needed = 150x100x5x5x3x200 = 225 million floating point operations
  - If feature maps are 32-bit floats, then we need 200x150x100x32/8 = 12 MB RAM
  - With just 100 minibatch, we are looking at 1.2 GB of RAM
  - PER LAYER!
- During inference, ram can be released layer by layer, but during training the entire information has to be stored

End of Additional Discussion

- Convolution vs Correlation
- Kernel Math
- Visualization
- Depth Wise Max Pool
- Memory Issues

Department of Computational and Data Sciences

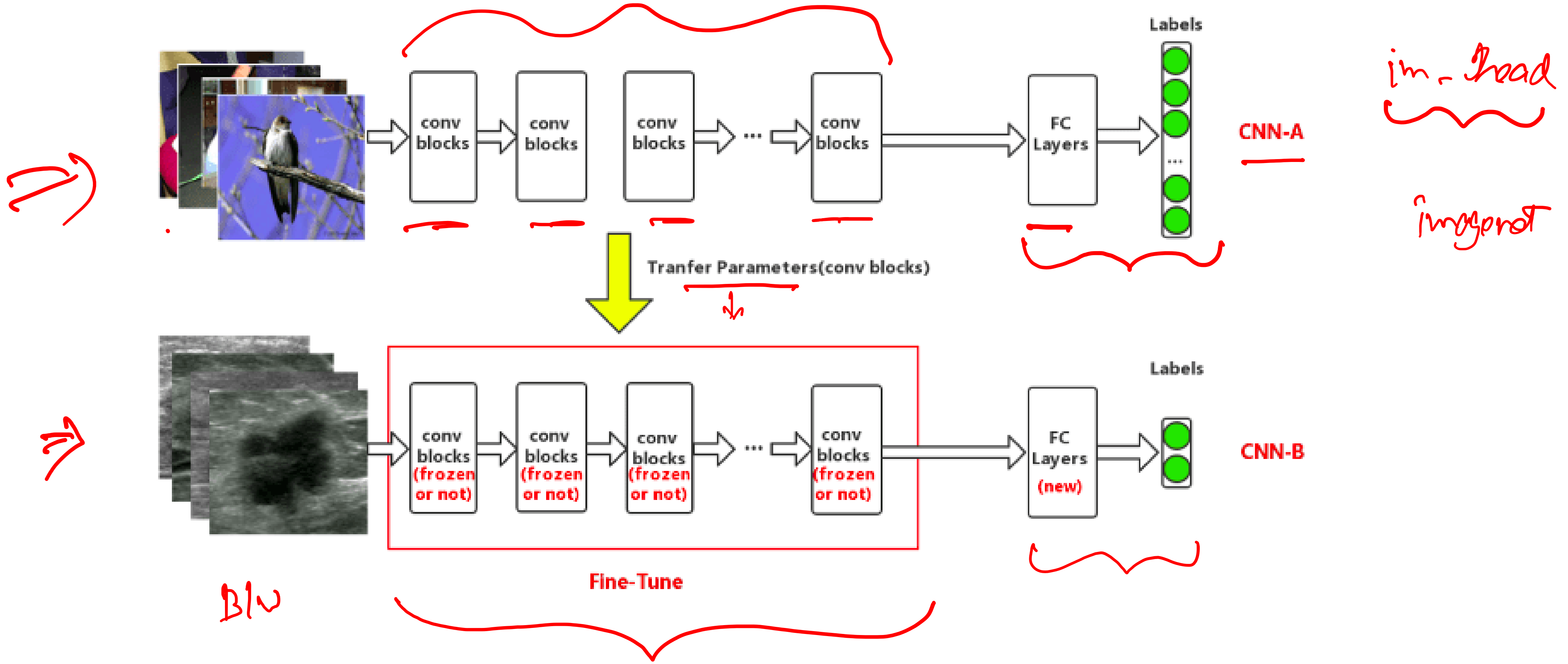
INDIAN INSTITUTE OF SCIENCE

भारतीय विज्ञान संस्थान

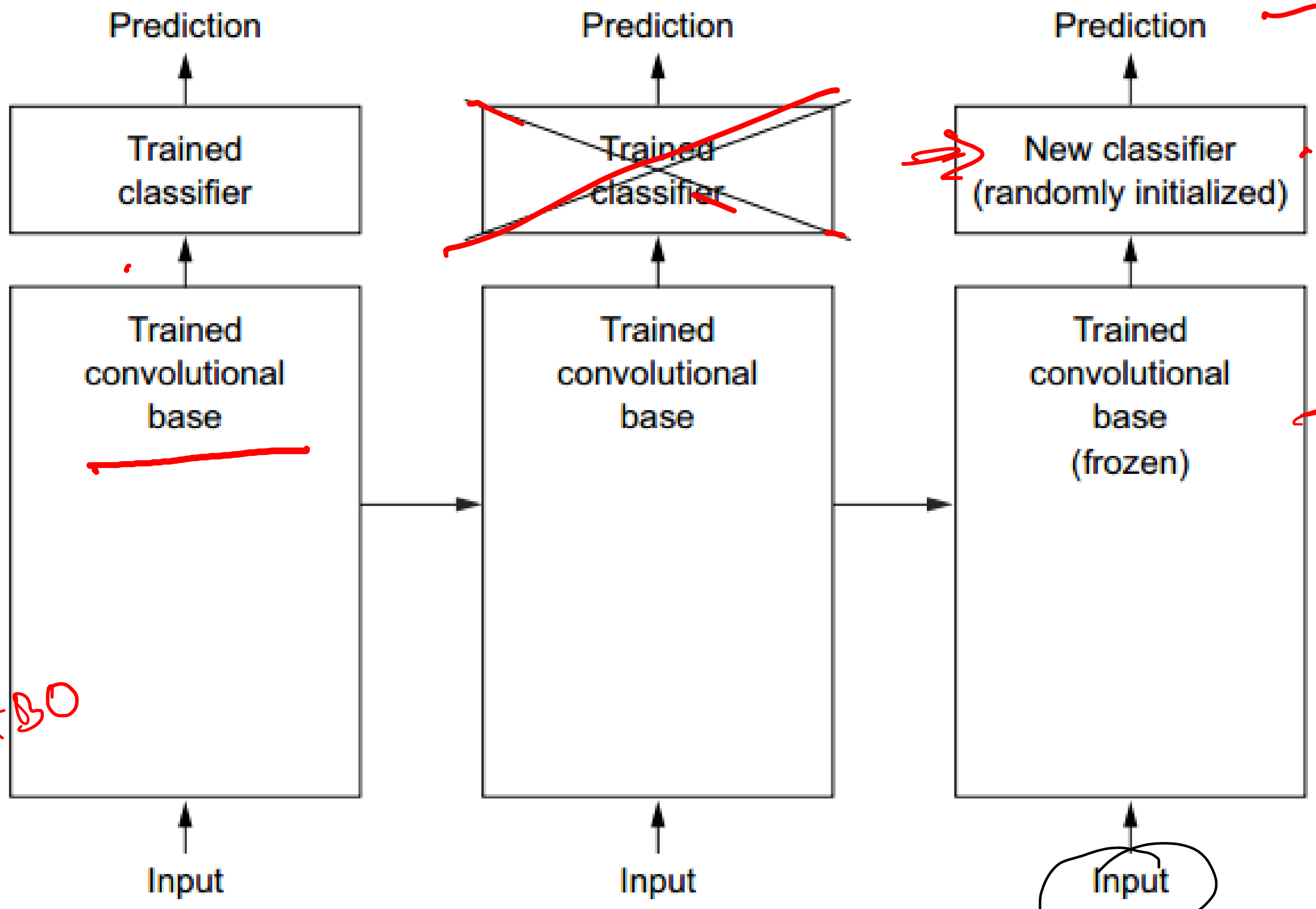
Week 01 Part 02

Deepak Subramani
Assistant Professor
Dept. of Computational and Data Science
Indian Institute of Science Bengaluru

Convolutional Blocks as Representation Learning Systems



From Tutorial



post-process

Process

Pre-process

ResNet50
EfficientNetB0
BW ← Xception
Separable

Recommended Strategy

- Small Dataset (<1000 labelled images) – Use Transfer Learning
- Medium Dataset (Upto 5000-10000) – Use Fine Tuning
- Large Dataset (Beyond 10k) – Train from scratch
  - Rules of thumb!
- Edge Devices use MobileNetv2
- SoTA needed? – Use Efficient Net (or even ViT)
- Traditional firms who like time-tested methods
  - ResNet50, VGG19
- If training cost and inference time are not a concern, use all three and do an ensemble!

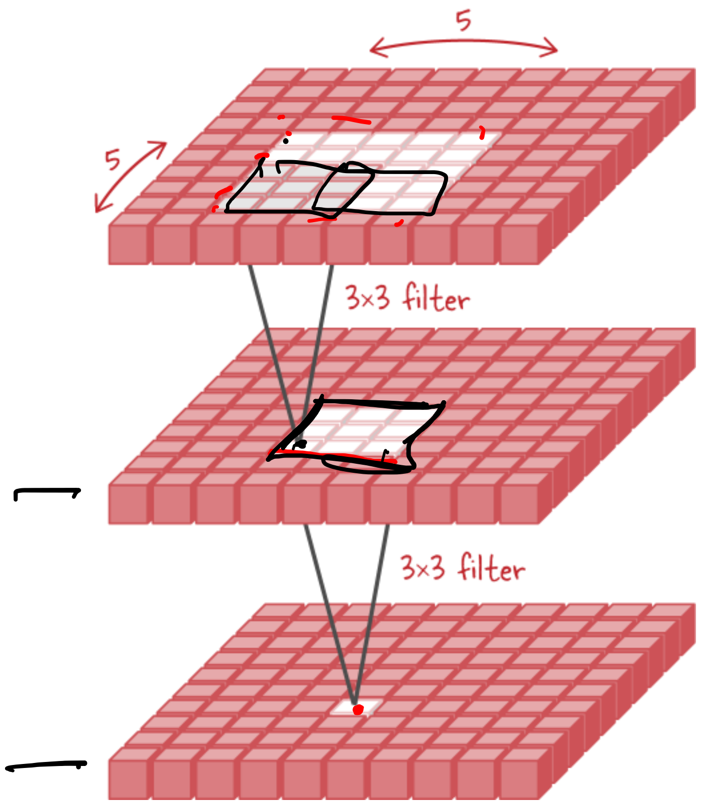
Quest for Depth

Deeper networks are favored for the following arguments

- Expressivity Argument
  - Single layers is only a linear classifier
  - Multiple layers with ReLU activation can capture complex nonlinear relations
- Generalization Argument
  - Adding neurons to a layer increases its memory (it by-hearts data!)
  - Adding layers makes the network learn hierarchical features
- Perceptive field argument
  - If a significant portion is 128x128 pixels, and we used only one layer, the filter needs to be 128x128
  - Deeper networks allows us to use 3x3 5x5 7x7 etc and progressively build a 128x128 receptive field

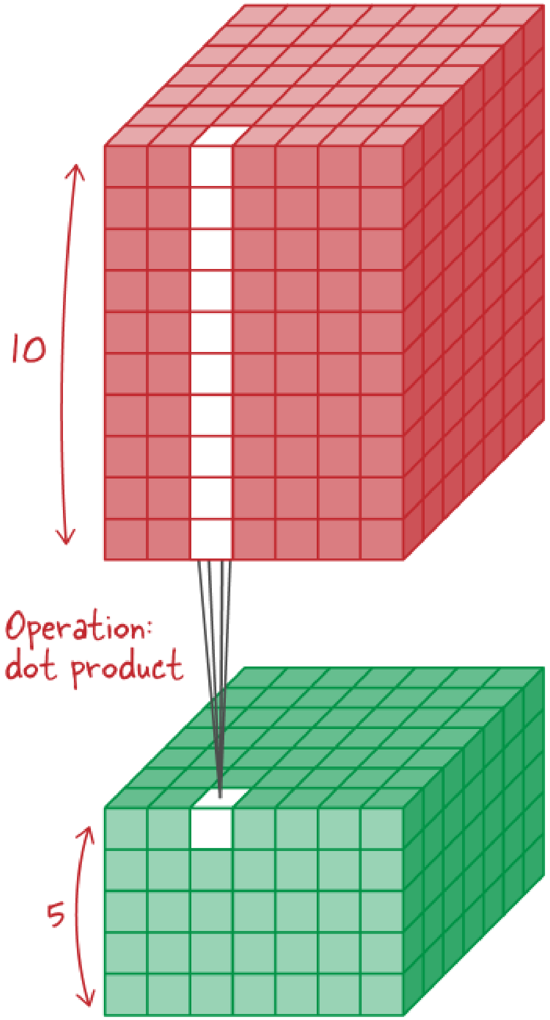
Filter Factorization

- Which is better? 5x5 or two 3x3 filters one after the other
- Both have the same receptive field
- But 5x5 has 25 learnable weights but two 3x3 has only 2*3*3=18 learnable weights
- In practice we use blocks which has two convolutional layers of 3x3



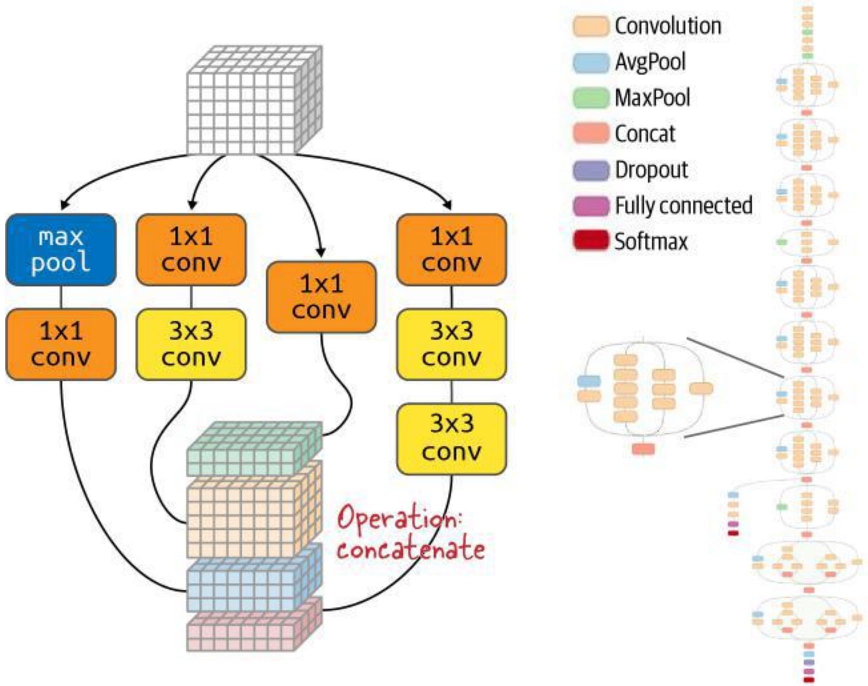
1x1 Convolution

- Produce a feature map that combines together all the feature maps (or channels) of the previous layer by a weighted linear combination
- 1x1 is often used to change the number of channels in the data
- 1x1 kernels can't learn spatial patterns, they capture patterns along the depth dimension



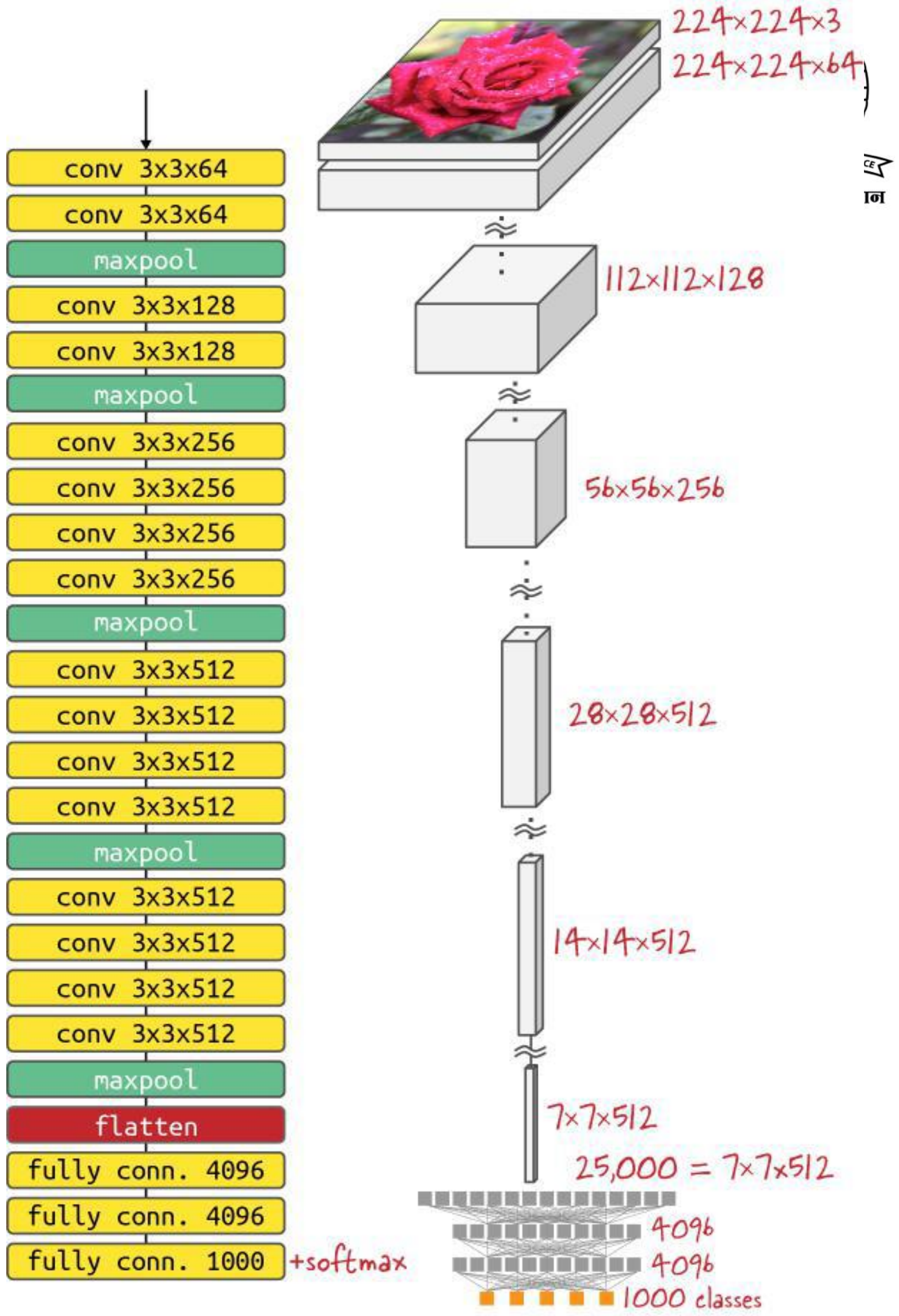
Modular Architecture

- A deep convolutional network is composed of stacks of convolutional blocks
- Each block has multiple convolutional layers arranged in a particular fashion
- Some Modules are important
  - Inception Module – 2014 Winner
  - ResNet Module – 2015 Winner
  - Xception Module – Depthwise Separable Convolutions
  - Inverted Residual Bottleneck – MobileNet, EfficientNet



VGG16, VGG19

- VGG - Visual Geometry Group
- Larger and deeper ConvNet
- Uses the same flatten layer structure
- Uses only 3x3 conv



Convolution to Dense

- In both ~~VGG Net~~ and Alex Net, a flatten operation is done to get a vector representation
- Global Average pooling is another way to bring to a vector representation