MDEC™

t- academy | Future Skills Academy for Emerging Technologies

CERTIFIED DEEP LEARNING PROFESSIONAL (CDLP)

ICTC International Council for Technology Certifications

Digital Tech Faculty Expert (DTeX)

International Council for Technology Certifications

Introduction to CNNs

Deep Learning

Deep Learning

- Welcome to the section on Convolutional Neural Networks!’
- CNNs are a specific architecture of Neural Networks that are extremely effective at dealing with image data.
- Let’s review what we will learn in this section!

Deep Learning

CNN Section

- Understanding CNNs
  - Image Kernels and Filters
  - Convolutions
  - Pooling Layers
- MNIST Dataset (Grayscale Images)
- CIFAR-10 Dataset (Color Images)

Deep Learning

CNN Section

- Working with image files (jpg,png,etc…) with CNNs.
- CNNs for malaria blood cell classification.
- CNN exercise on Fashion Image Dataset.

ICTC International Council for Technology Certifications

MNIST Data

Deep Learning

Deep Learning

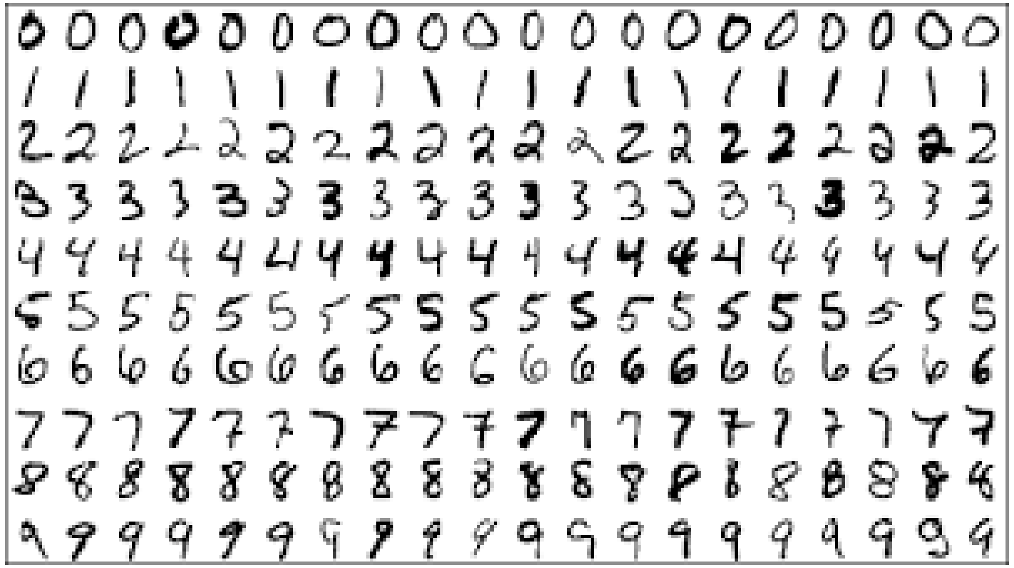
- A classic data set in Deep Learning is the MNIST data set.
- Let’s quickly cover some basics about it since we’ll be using it quite frequently during this section of the course!

Deep Learning

- Fortunately this data is easy to access with PyTorch Vision. We will download:
  - 60,000 training images
  - 10,000 test images

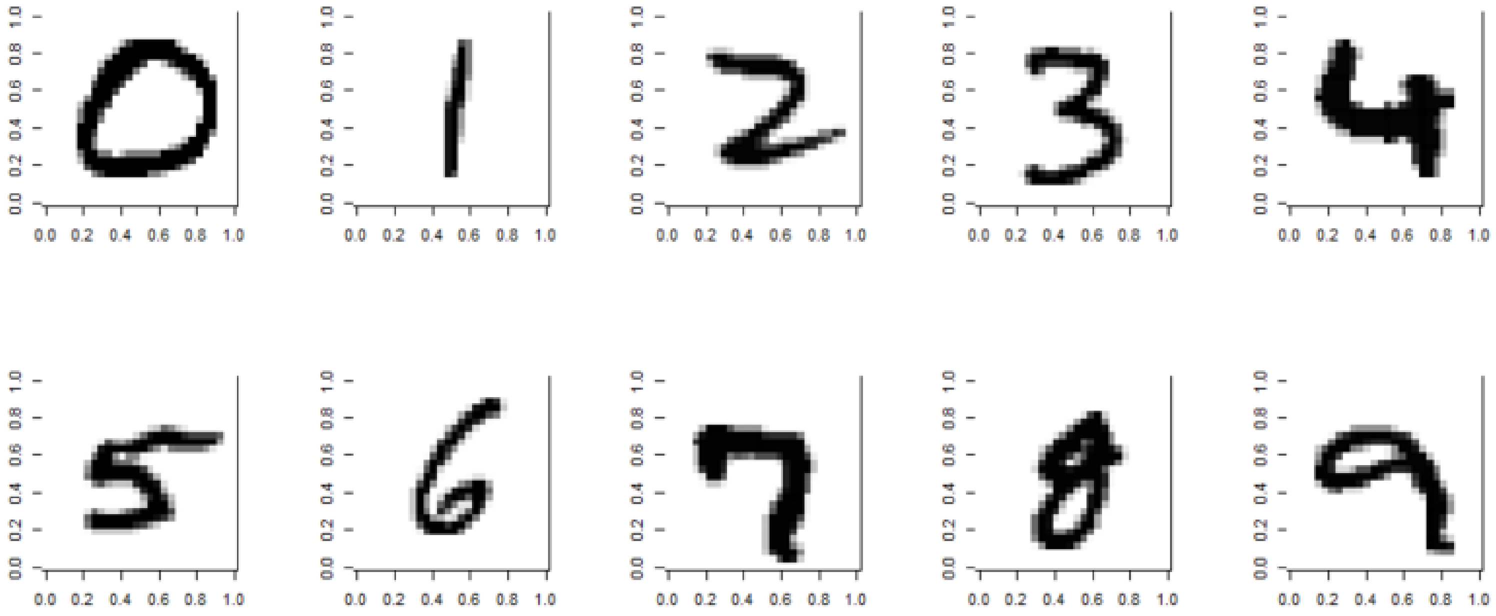
Deep Learning

- The MNIST data set contains handwritten single digits from 0 to 9



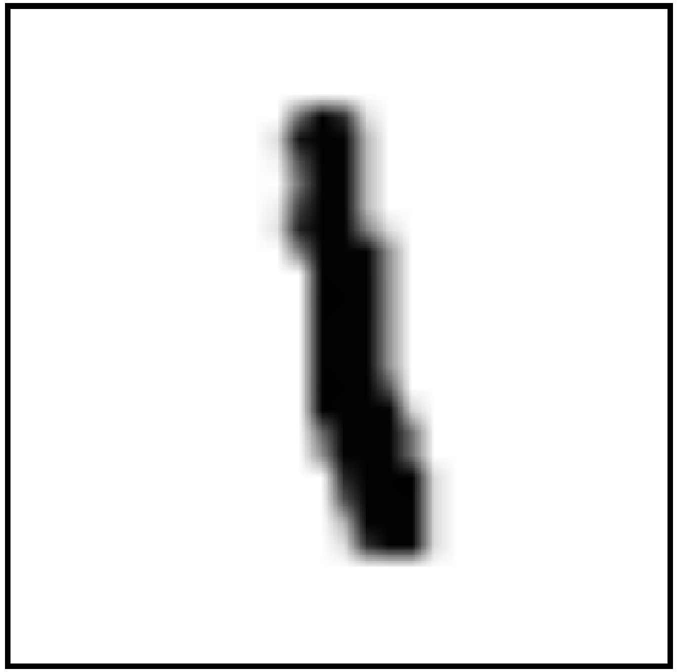
Deep Learning

- A single digit image can be represented as an array



Deep Learning

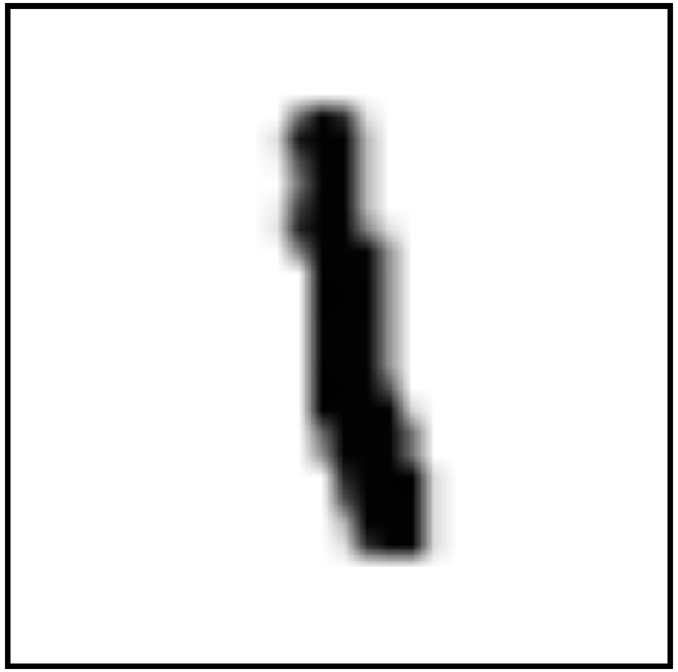
- Specifically, 28 by 28 pixels



$$\simeq \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .6 & .8 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .7 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .7 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .5 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .7 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .9 & 1 & .1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .3 & 1 & .1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

Deep Learning

- The values represent the grayscale image



$$\simeq \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .6 & .8 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .7 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .7 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .5 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .7 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .9 & 1 & .1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .3 & 1 & .1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

Deep Learning

- We will first explore how we could approach this data set with a standard ANN.
- In this case, to feed it into our network we will need to flatten the 28 by 28 array to 784

Deep Learning

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

Deep Learning

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

0 0 0 0 0 .6 .8 … 1 .1 0 0 0 0

Deep Learning

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

$$\Rightarrow \underbrace{0\quad 0\quad 0\quad 0\quad 0\quad .6\quad .8\quad \ldots\quad 1\quad .1\quad 0\quad 0\quad 0\quad 0}_{\text{784 inputs}}$$

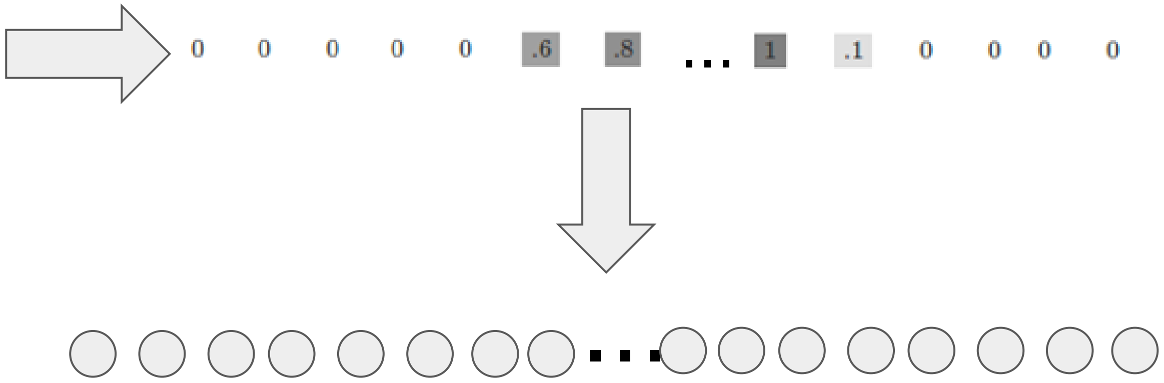
Deep Learning

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

0 0 0 0 0 .6 .8 ... 1 .1 0 0 0 0

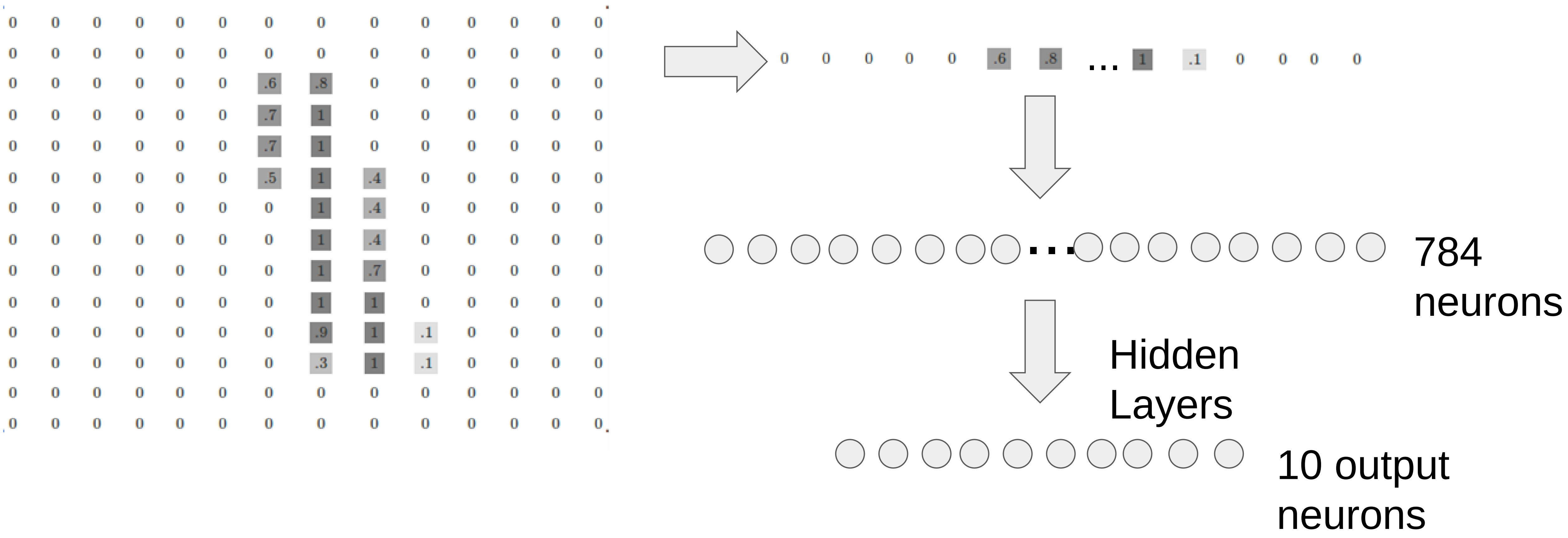
Deep Learning

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



784 neurons

Deep Learning



Deep Learning

- Flattening out the image ends up removing some of the 2-D information, such as the relationship of a pixel to its neighboring pixels.
- For now, we’ll ignore this, but come back to it later when we discuss CNN in depth!

MNIST ANN

PART ONE: DATA

MNIST ANN

PART TWO: NEURAL NETWORK

MNIST ANN

PART THREE: TRAINING AND EVALUATION

MNIST ANN

PART FOUR: EVALUATION

ICTC International Council for Technology Certifications

Image Filters

Deep Learning

Deep Learning

- Before we dive into CNNs, let’s first discuss a few key ideas in computer vision.
- Computer vision is a general term of using computer programs to process image data.

Deep Learning

- If you've ever used photo editing software, you have probably seen filters, such as a blur filter. But how do these work?

Deep Learning

- Filters are essentially an **image kernel**, which is a small matrix applied to an entire image.
- Certain popular filters are well known, for example a blur filter:

$$\begin{pmatrix} 0.0625 & 0.125 & 0.0625 \\ 0.125 & 0.25 & 0.125 \\ 0.0625 & 0.125 & 0.0625 \end{pmatrix}$$

Deep Learning

- Let’s explore how these image kernel/filters actually get applied to an image.

Deep Learning

- Filters allow us to transform images

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

Deep Learning

- Here we have an grayscale image

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

Deep Learning

- Values scaled between -1 and 1

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

Deep Learning

- Filters allow us to transform images

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

0	0	0
0	-1	1
0	1	-1

3 by 3 Filter

Deep Learning

- Filters allow us to transform images

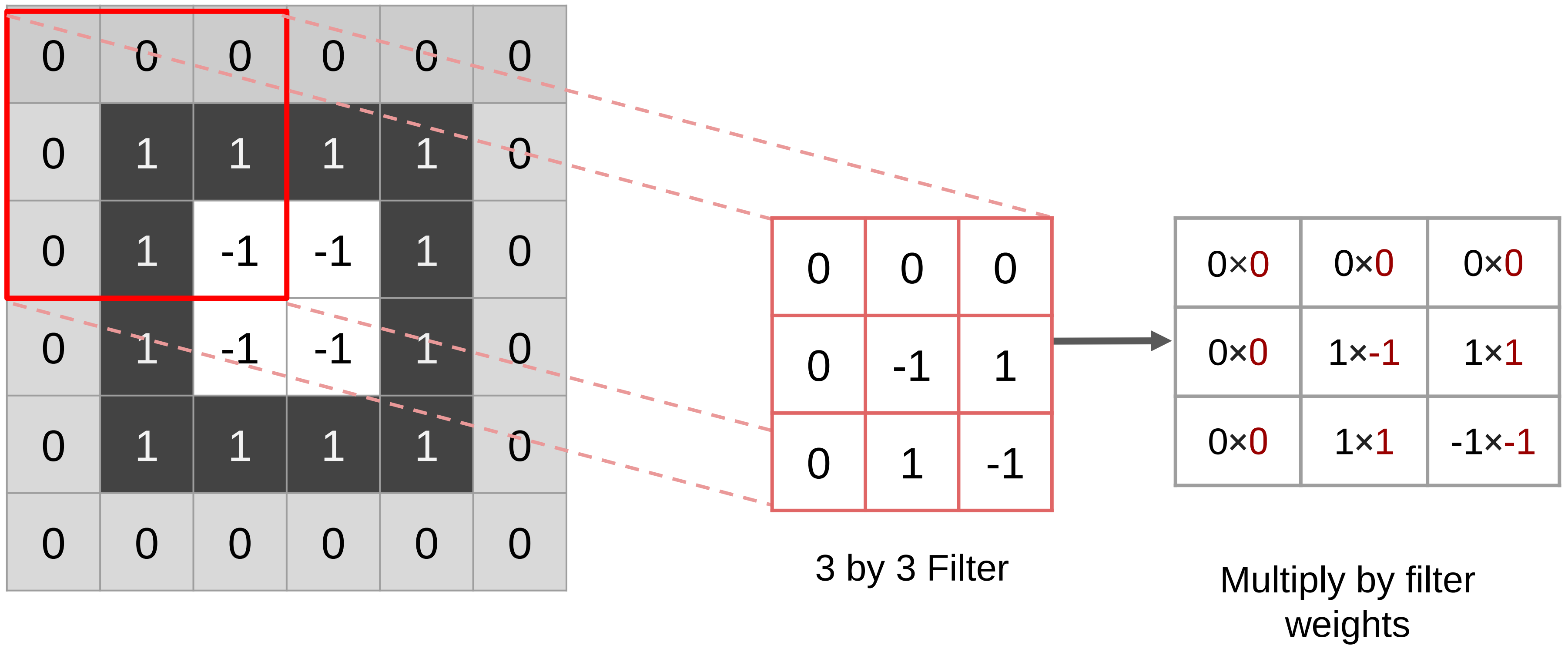
0	0	0	0	0	0
0	1	1	1	1	0
0	1	-1	-1	1	0
0	1	-1	-1	1	0
0	1	1	1	1	0
0	0	0	0	0	0

0	0	0
0	-1	1
0	1	-1

3 by 3 Filter

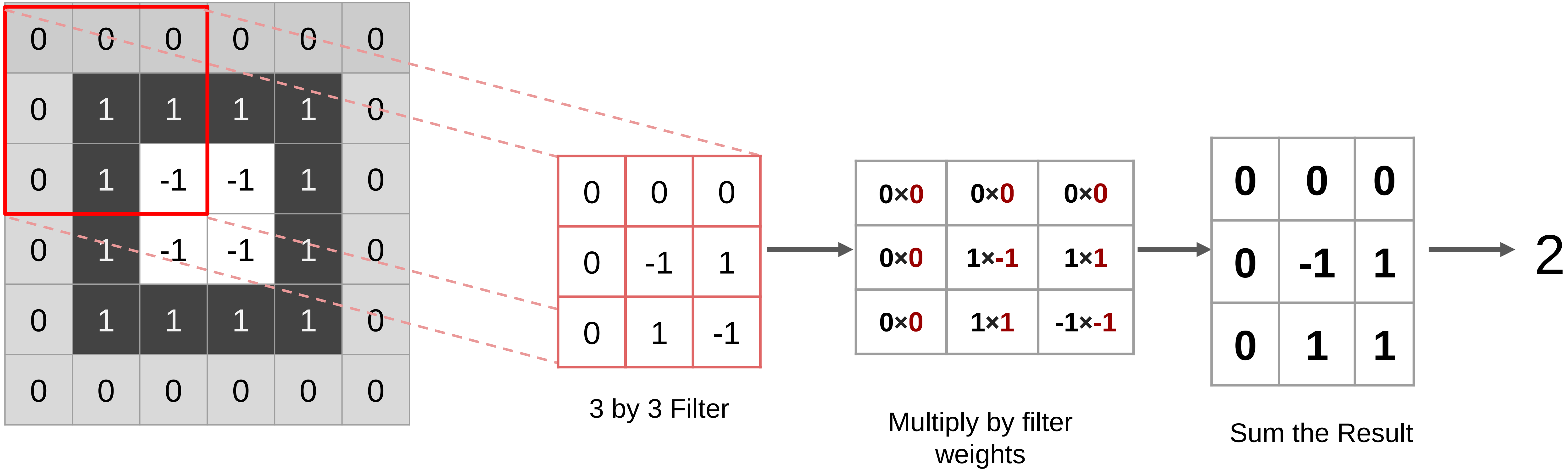
Deep Learning

- Filters are essentially a matrix



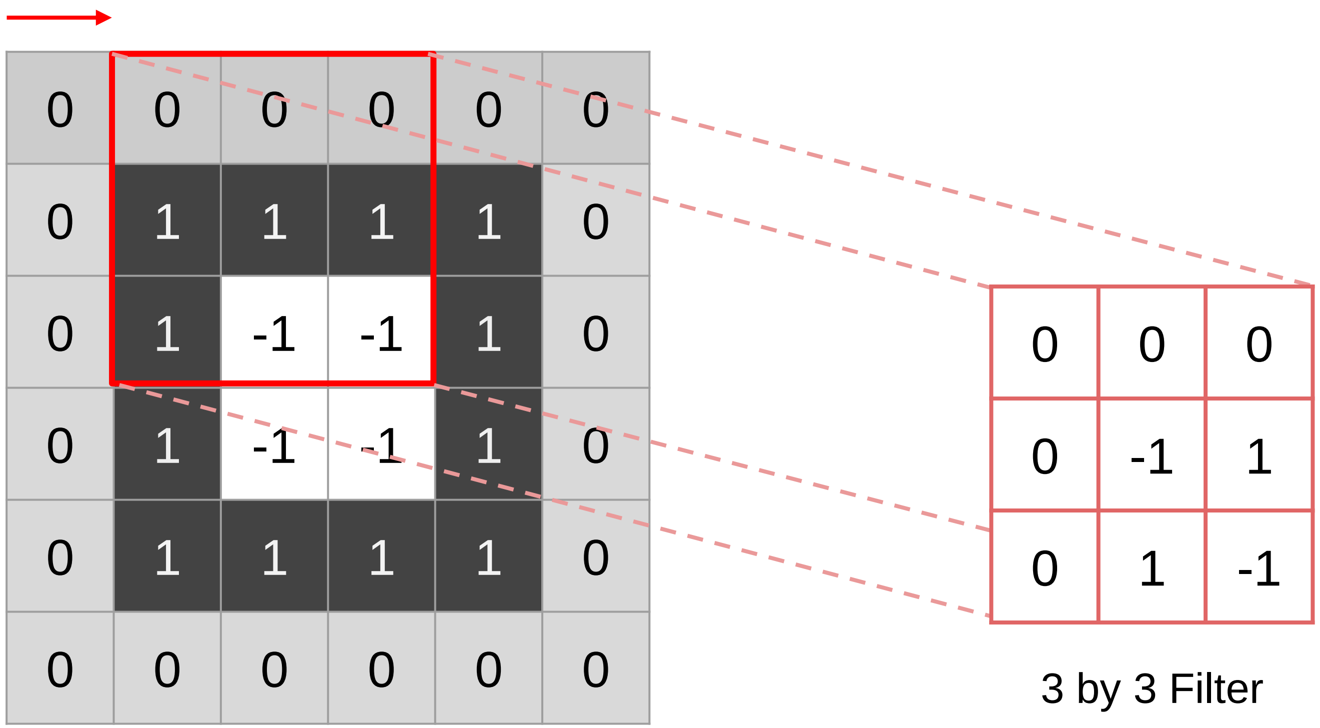
Deep Learning

- Notice how the resolution will decrease



Deep Learning

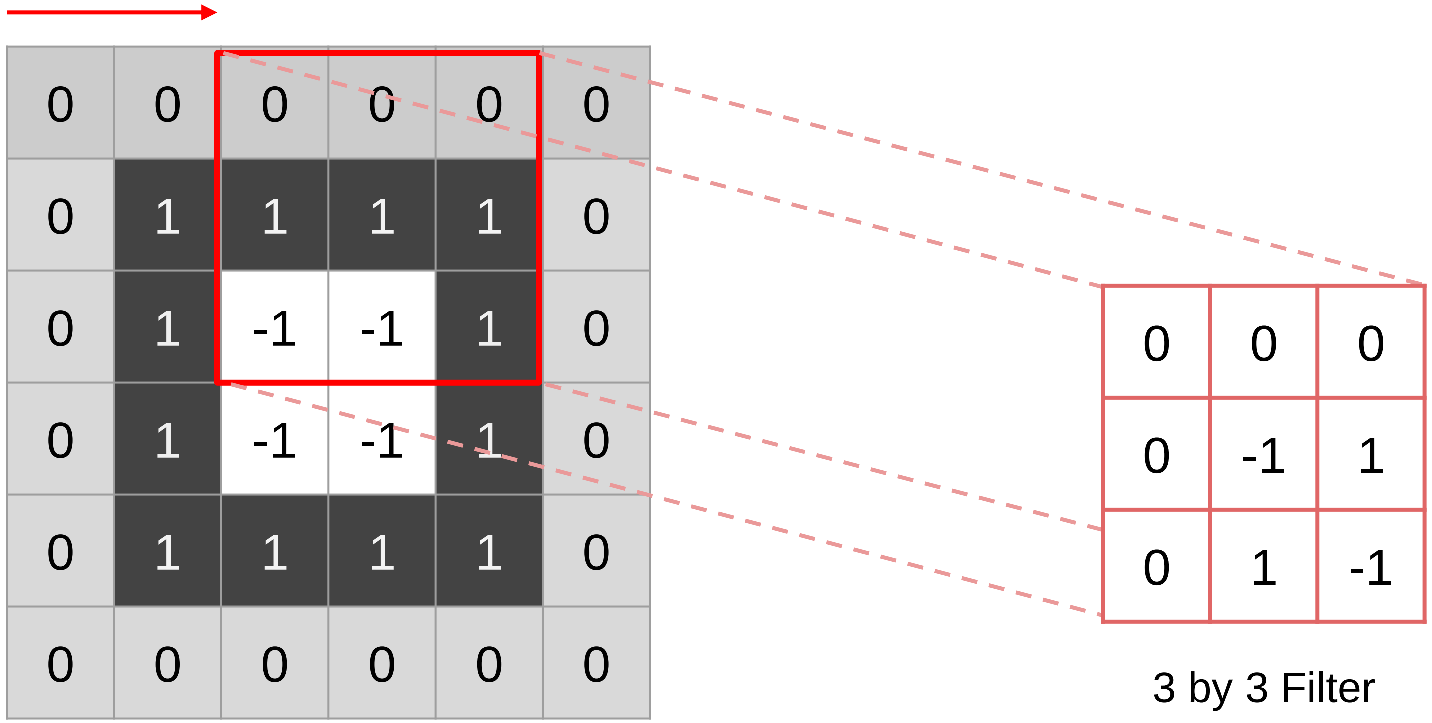
- We can also edit our stride distance



3 by 3 Filter

Deep Learning

- Stride Distance of 2 Example



3 by 3 Filter

Deep Learning

Let’s explore an interactive example!

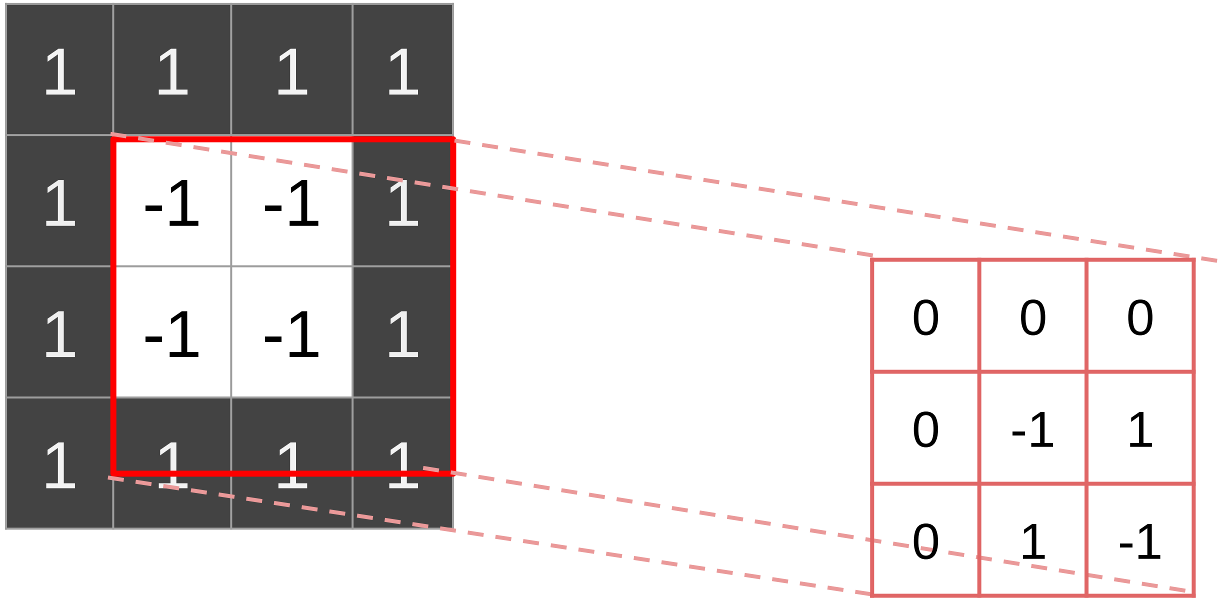
- **setosa.io/ev/image-kernels/**

Deep Learning

- In the context of CNNs, these “filters” are referred to as **convolution kernels**.
- The process of passing them over an image is known as **convolution**.
- Let’s go over a few more important factors!

Deep Learning

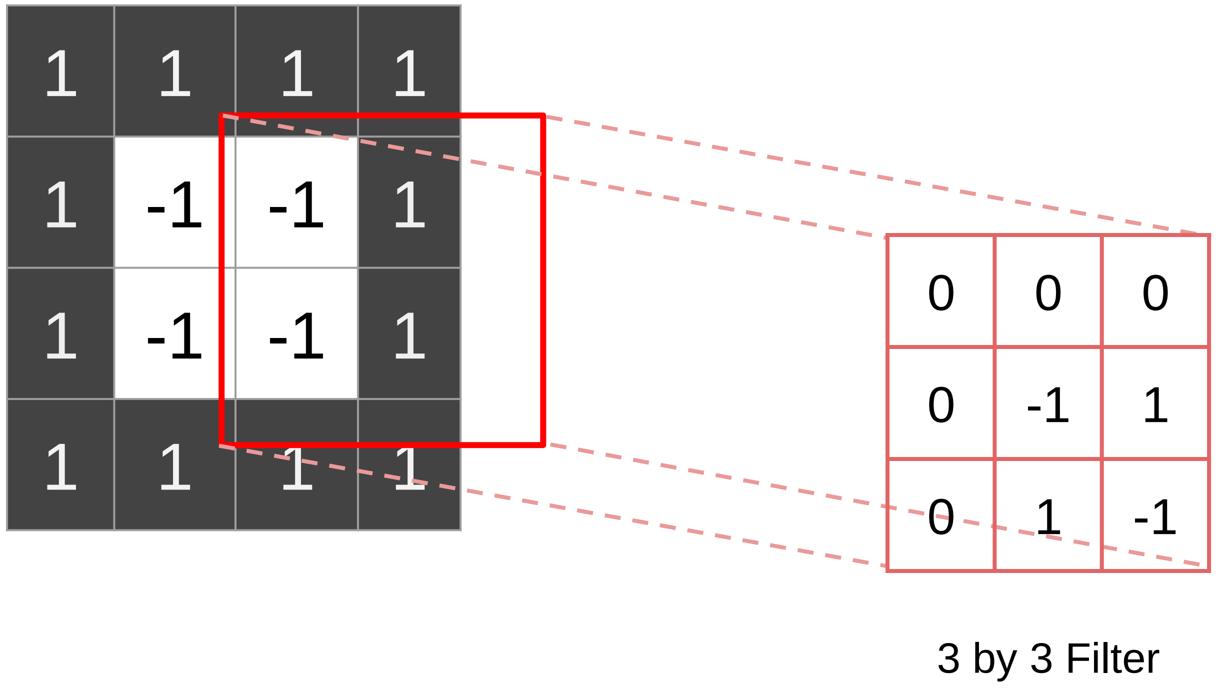
During convolution, we would lose borders



3 by 3 Filter

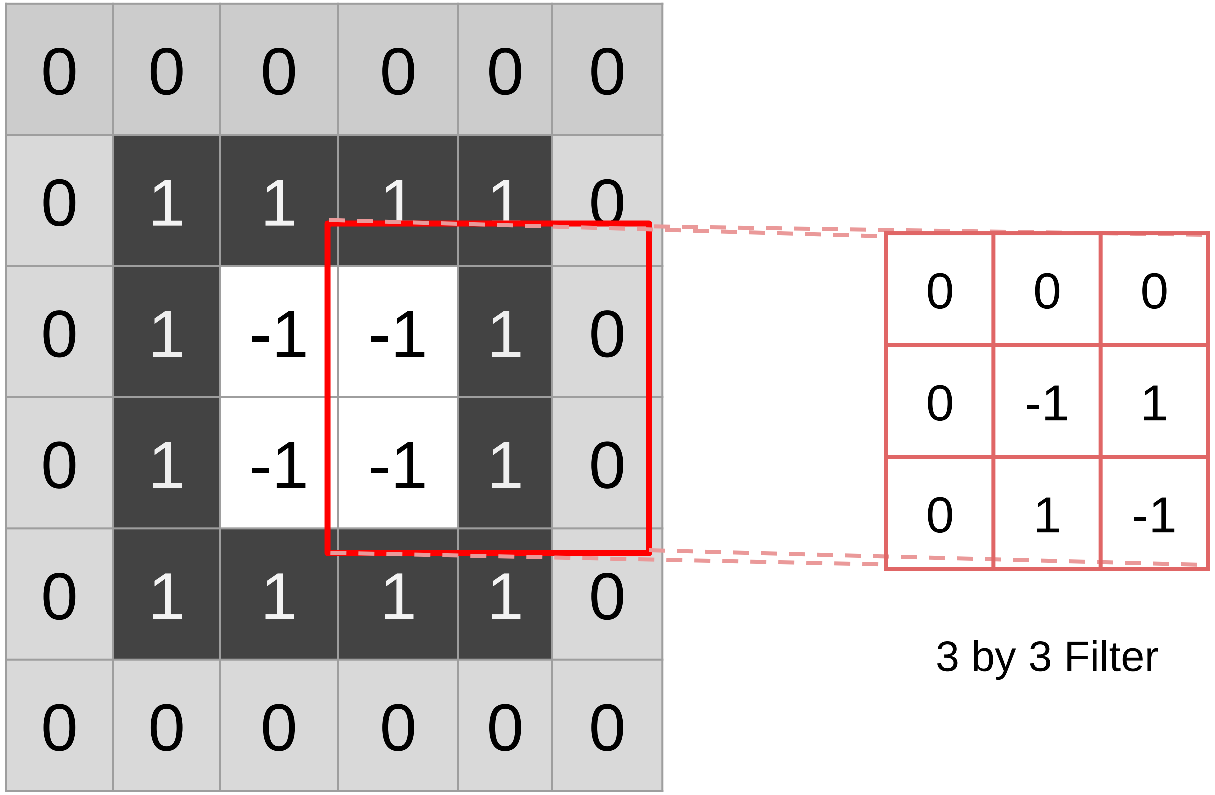
Deep Learning

During convolution, we would lose borders



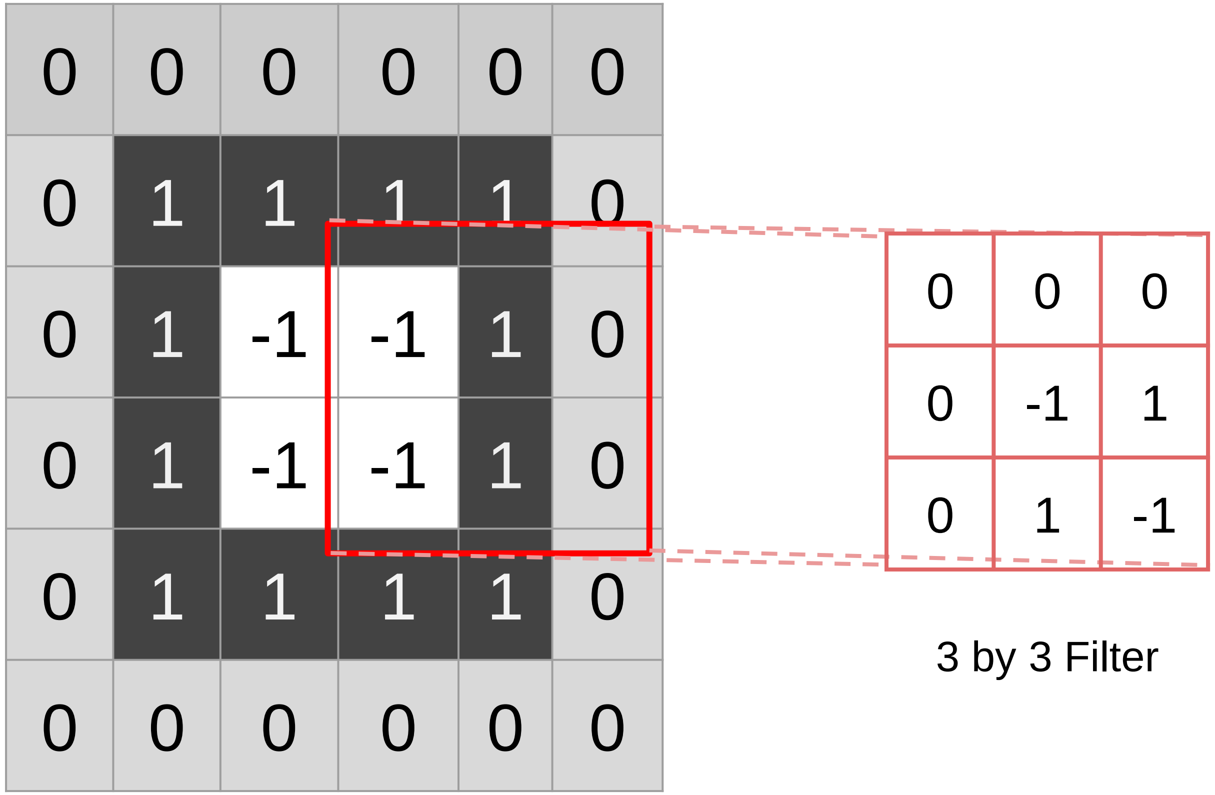
Deep Learning

We can **pad** the image with more values



Deep Learning

This allows us to preserve the image size



3 by 3 Filter

Deep Learning

- Now that we understand image filters, let’s explore the architecture of a CNN that allows the network to come up with the best weights for a filter in the **convolutional layer**.

ICTC International Council for Technology Certifications

Convolutional Layers

Deep Learning

Deep Learning

- Recall that running an ANN for the MNIST data set resulted in a network with relatively good accuracy.
- However, there are some issues with always using ANN models for image data.

Deep Learning

ANNs

- Large amount of parameters (over 100,000 for tiny 28 by 28 images)
- We lose all 2D information by flattening out the image
- Will only work on very similar, well centered images.

Deep Learning

- A CNN can use convolutional layers to help alleviate these issues.
- A convolutional layer is created when we apply multiple image filters to the input images.
- The layer will then be trained to figure out the best filter weight values.

Deep Learning

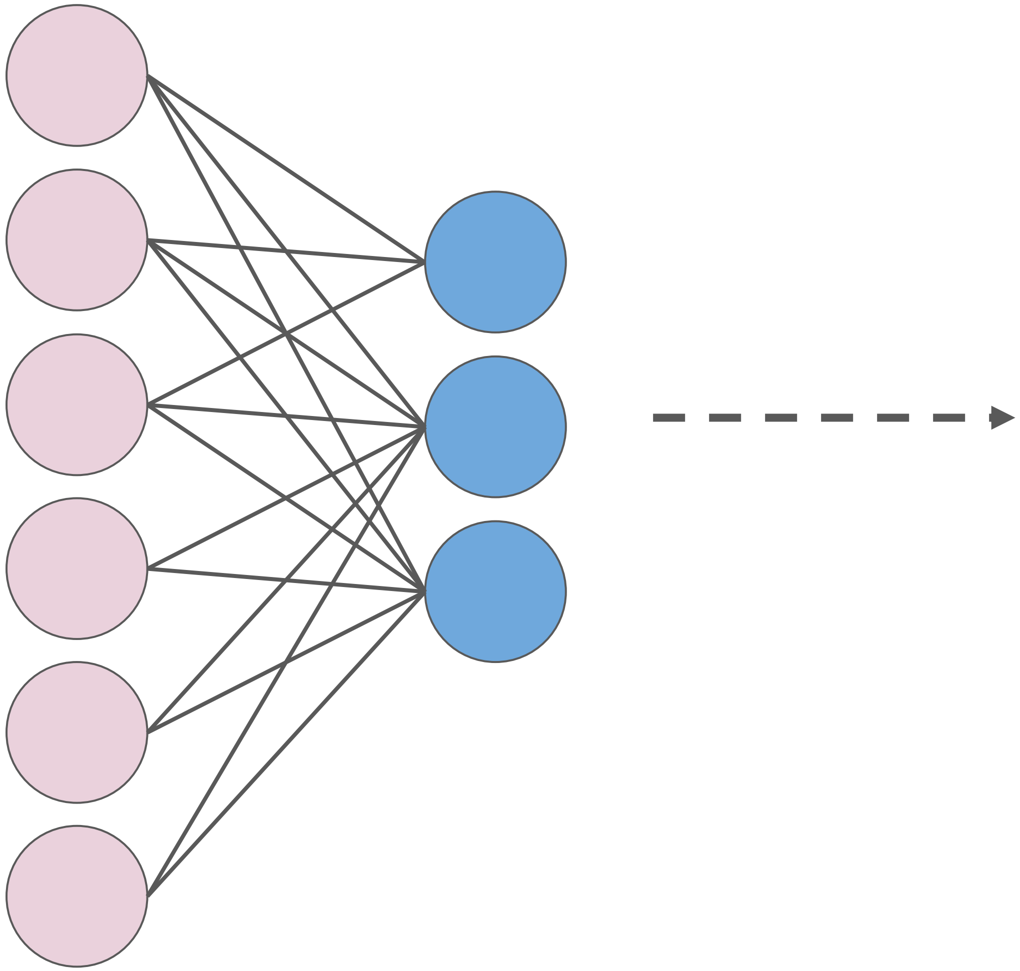
- A CNN also helps reduce parameters by focusing on **local connectivity.**
- Not all neurons will be fully connected.
- Instead, neurons are only connected to a subset of local neurons in the next layer (these end up being the filters!)

Deep Learning

- Let’s understand this local connectivity and its connection to filters by beginning with a simplified 1-D example.
- We will later expand this to 2-D inputs for a grayscale image and then later on to 3-D tensor inputs for color images.

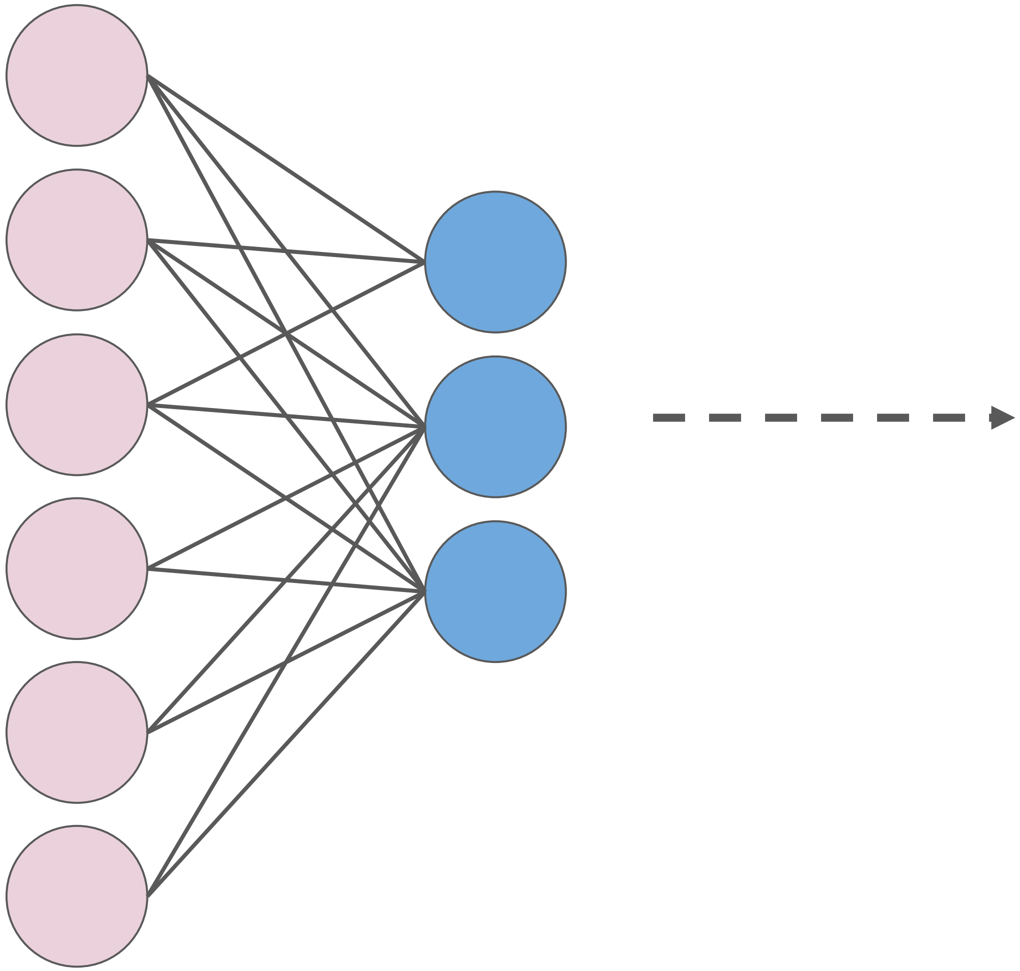
Deep Learning

- ANN



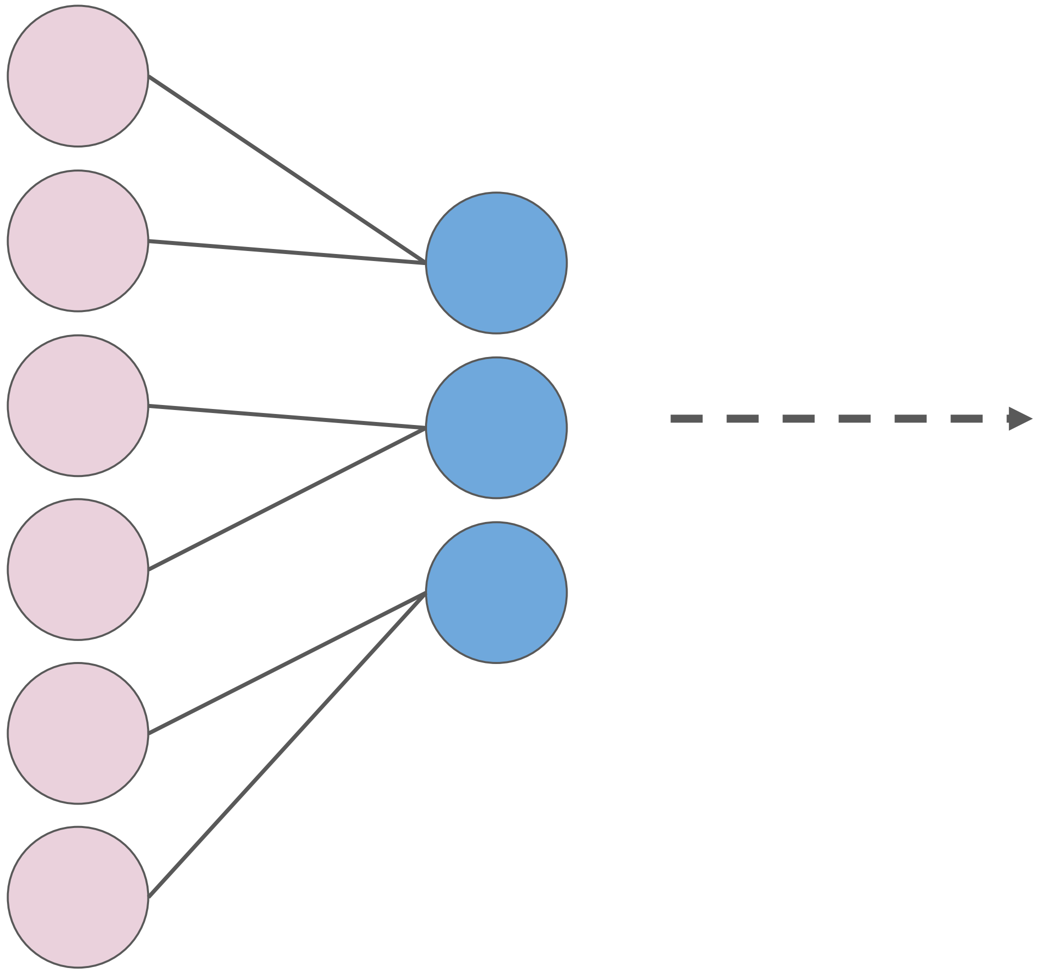
Deep Learning

- Fully Connected, lots of parameters!



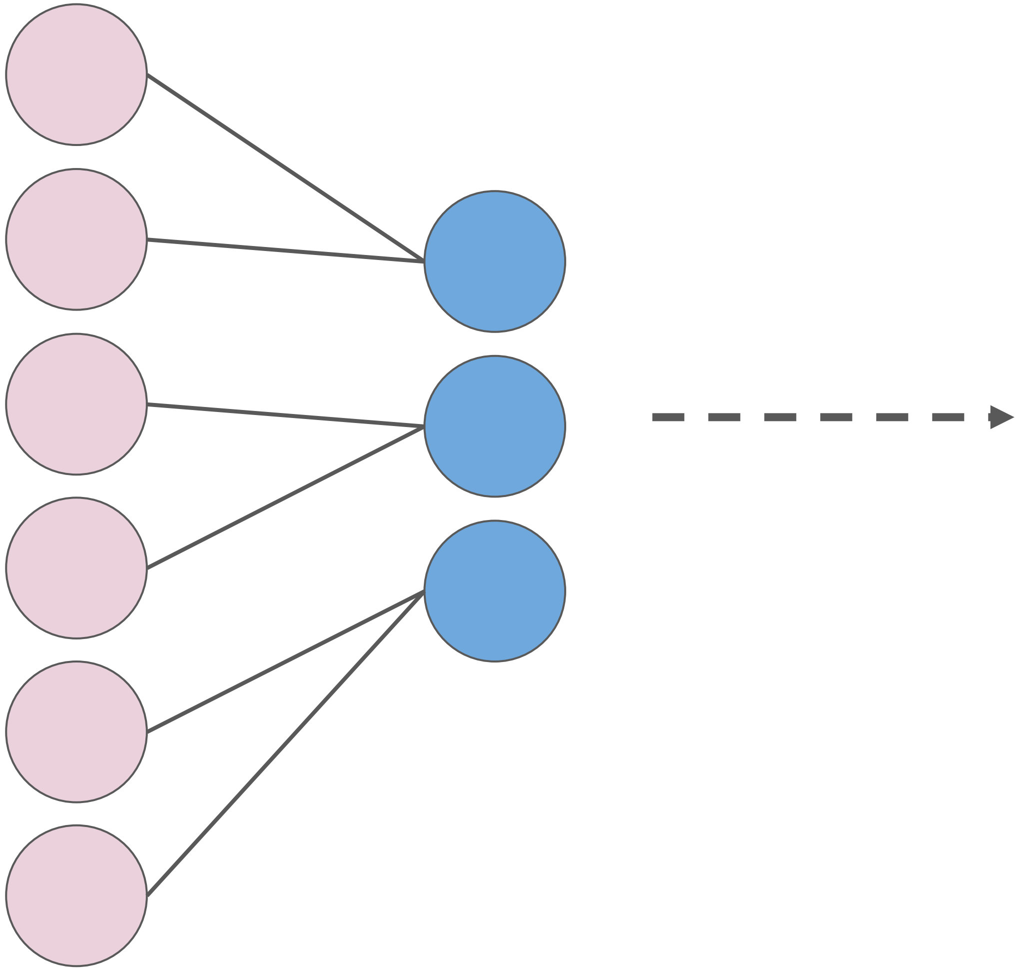
Deep Learning

- CNN - Convolutional Layer



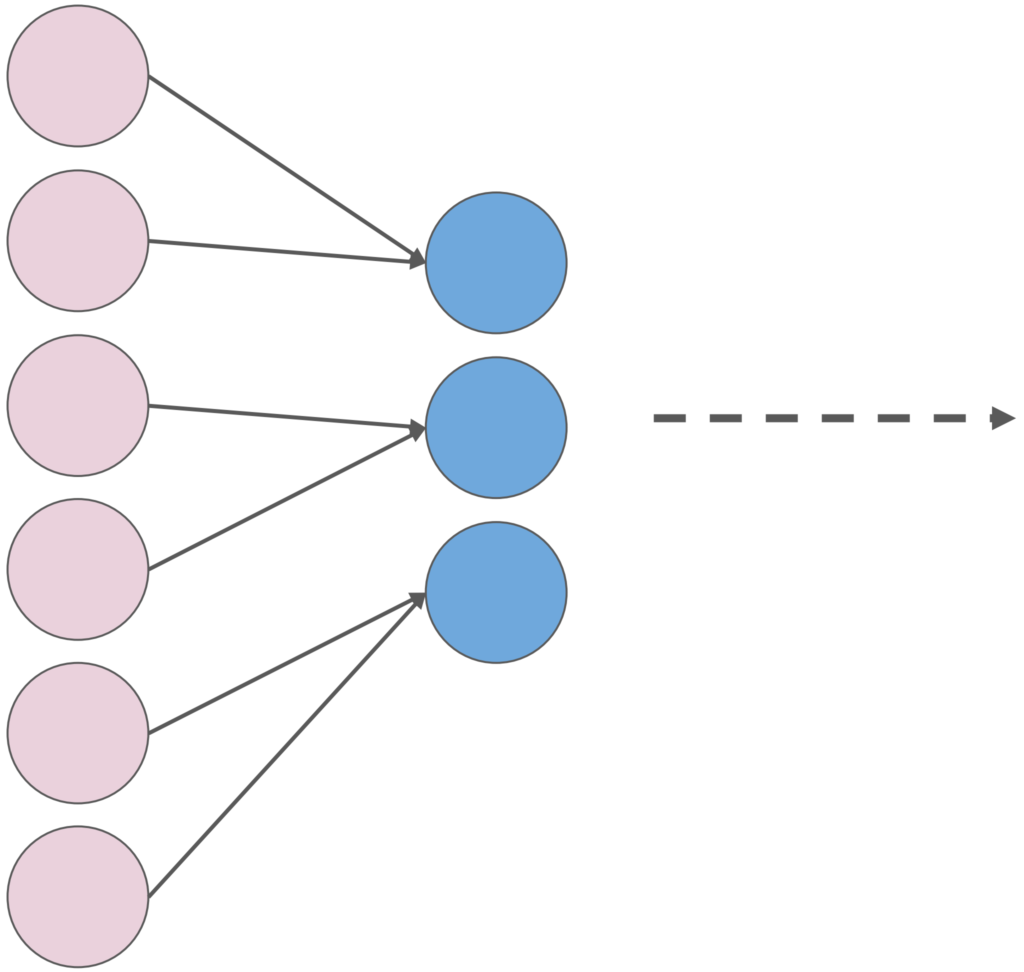
Deep Learning

- CNN - Localized Connections



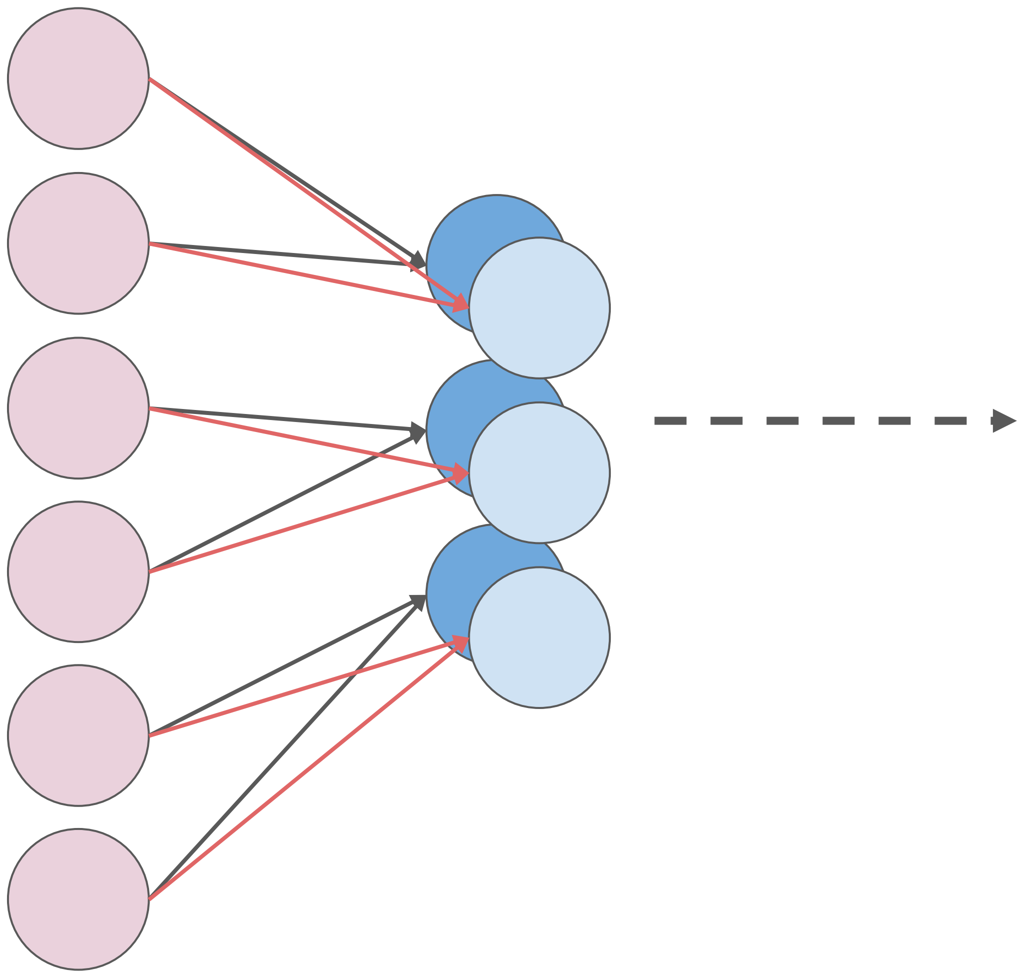
Deep Learning

- CNN - Here we have 1 filter



Deep Learning

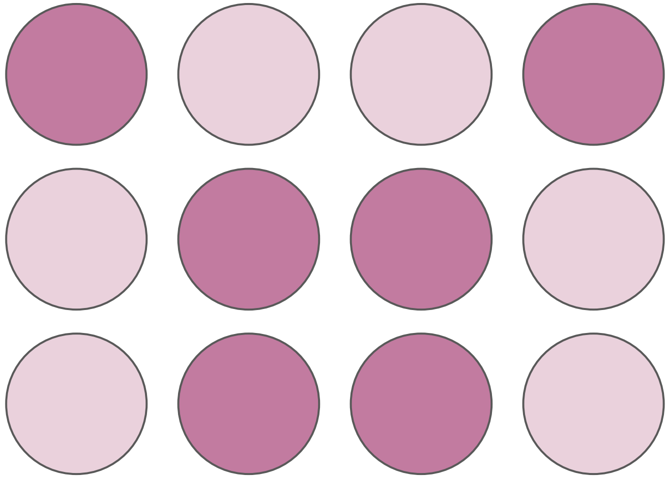
- CNN - Here we have 2 filters



Deep Learning

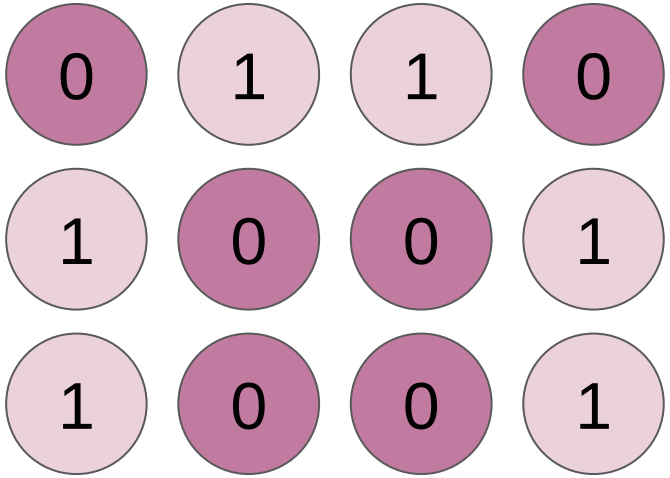
- We just saw a 1-D example of convolution, but recall, grayscale images are 2-D!
- Plus we want to preserve that 2-D relational information in the convolutional layer.

Deep Learning



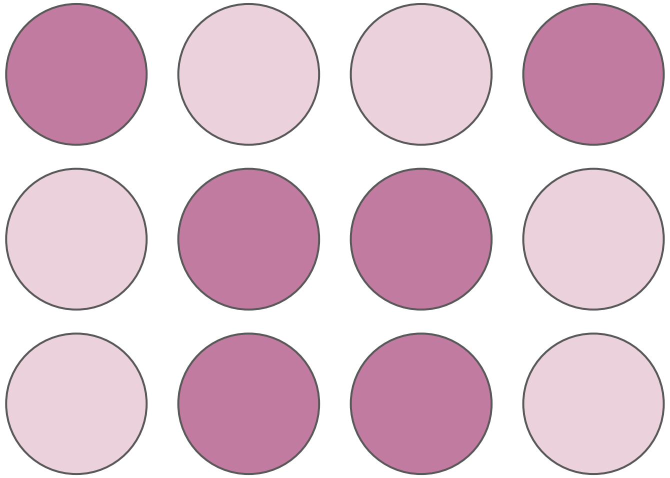
Input Image

Deep Learning



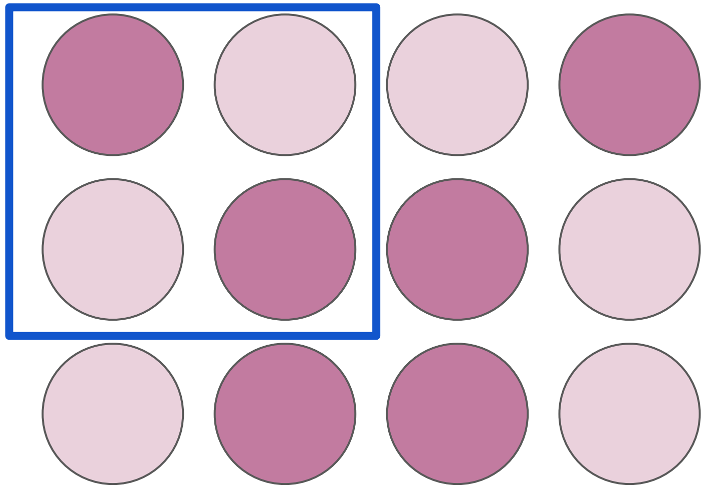
Input Image

Deep Learning



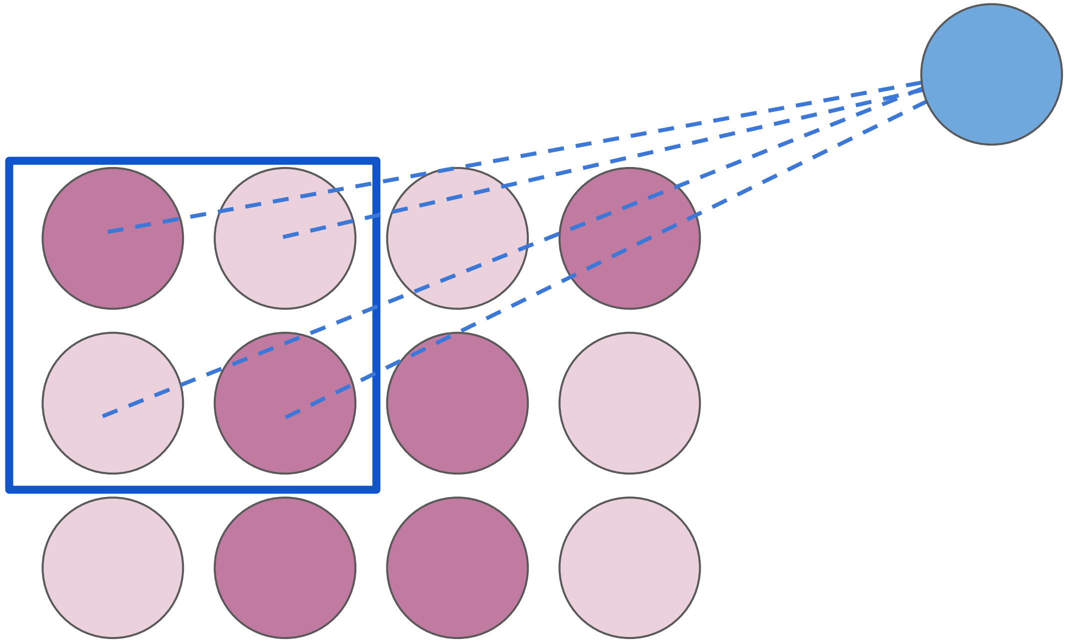
Input Image

Deep Learning



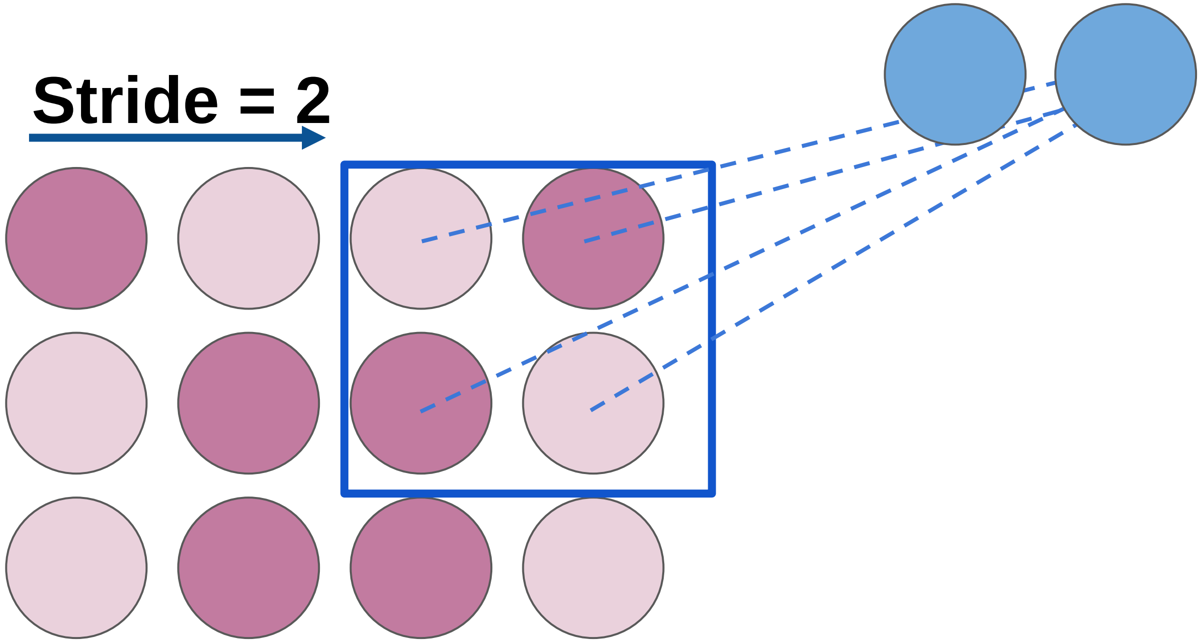
Input Image

Deep Learning



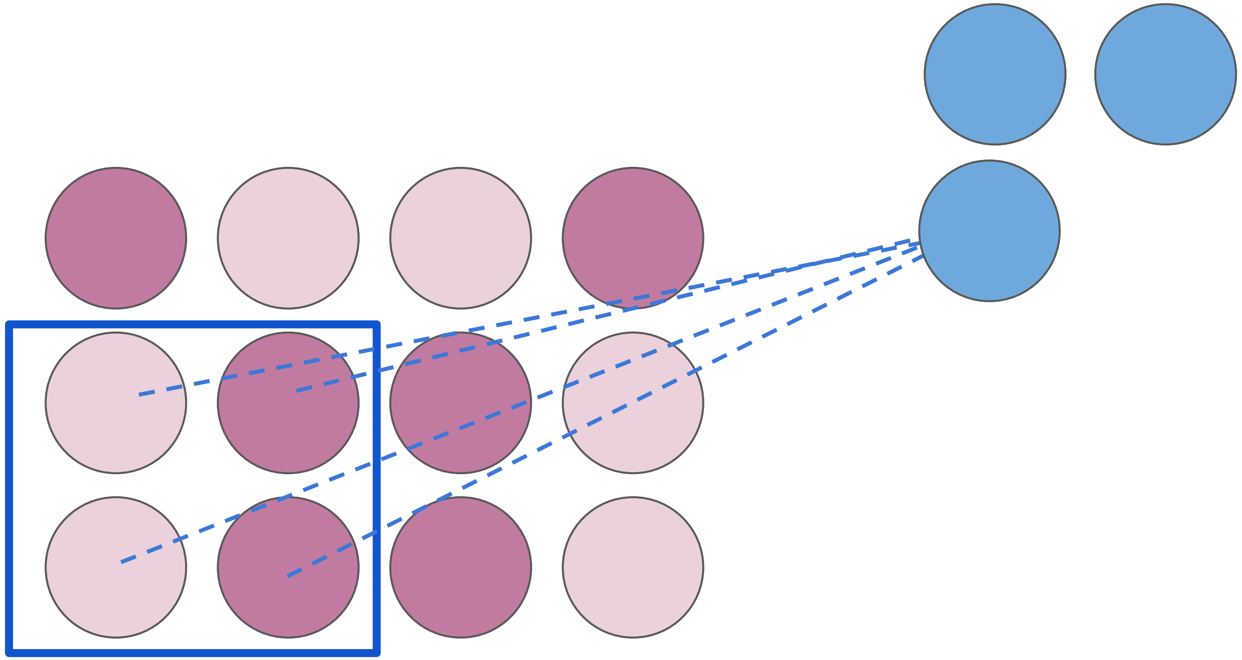
Input Image

Deep Learning



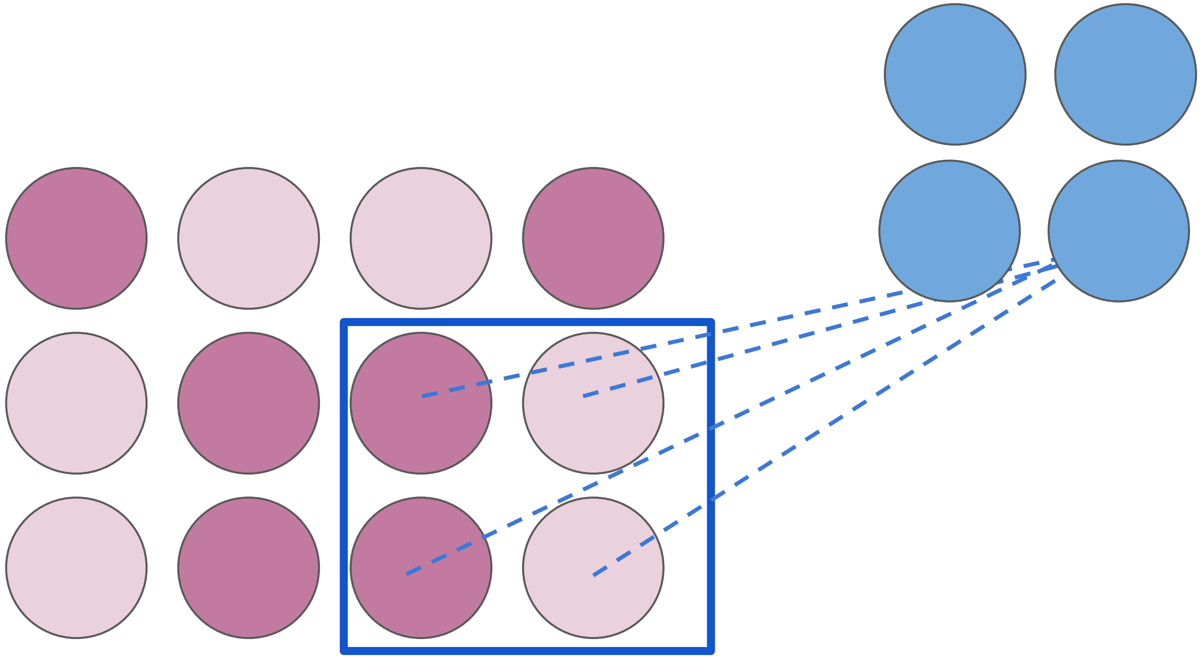
Input
Image

Deep Learning



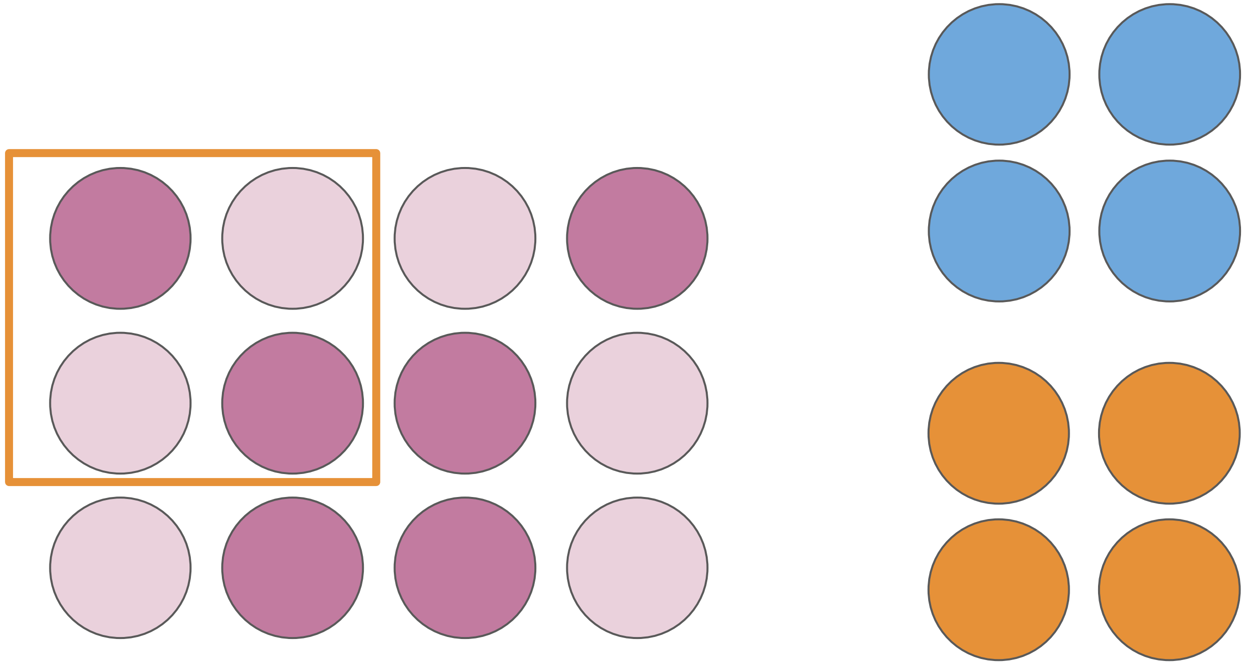
Input Image

Deep Learning



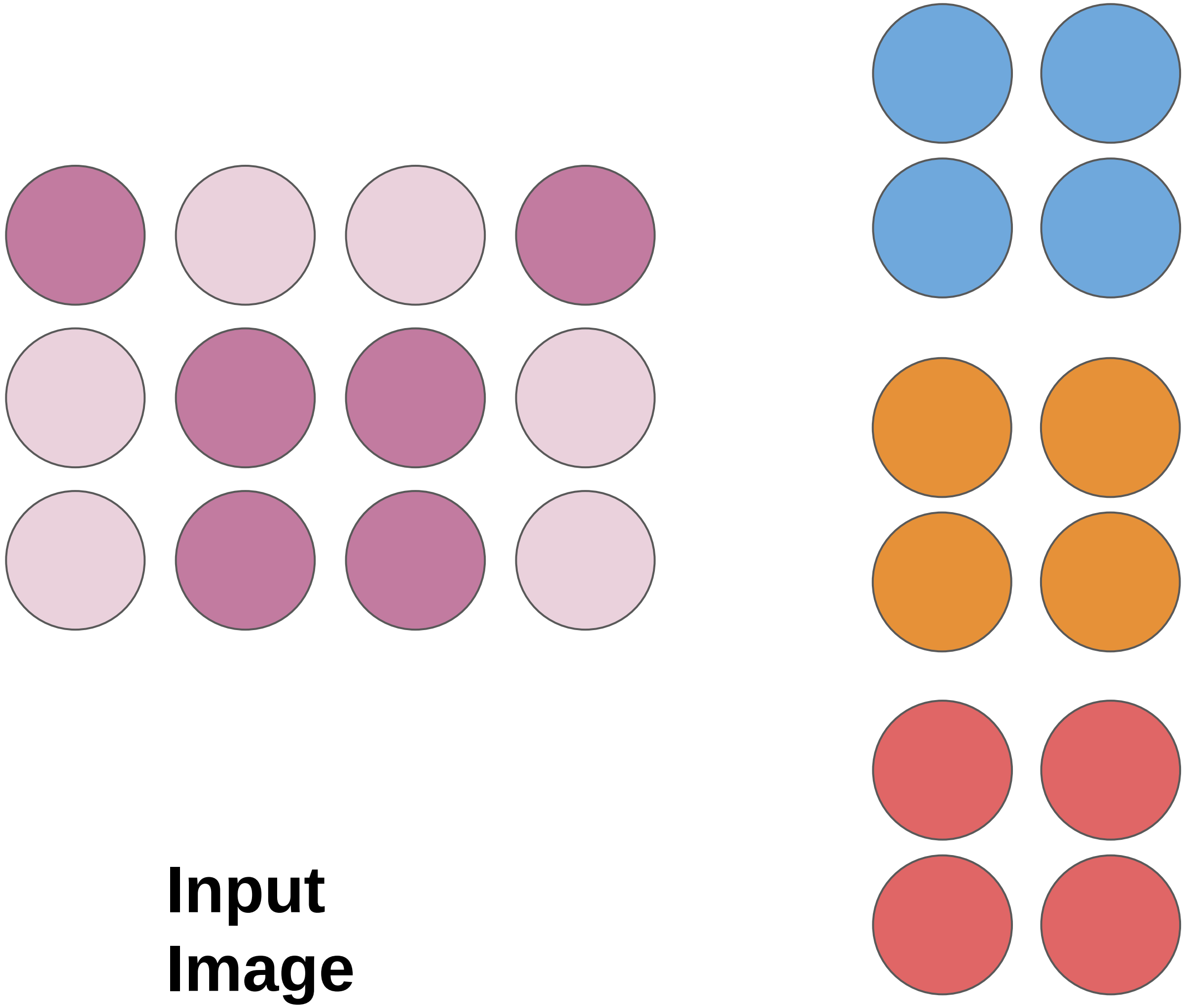
**Input
Image**

Deep Learning



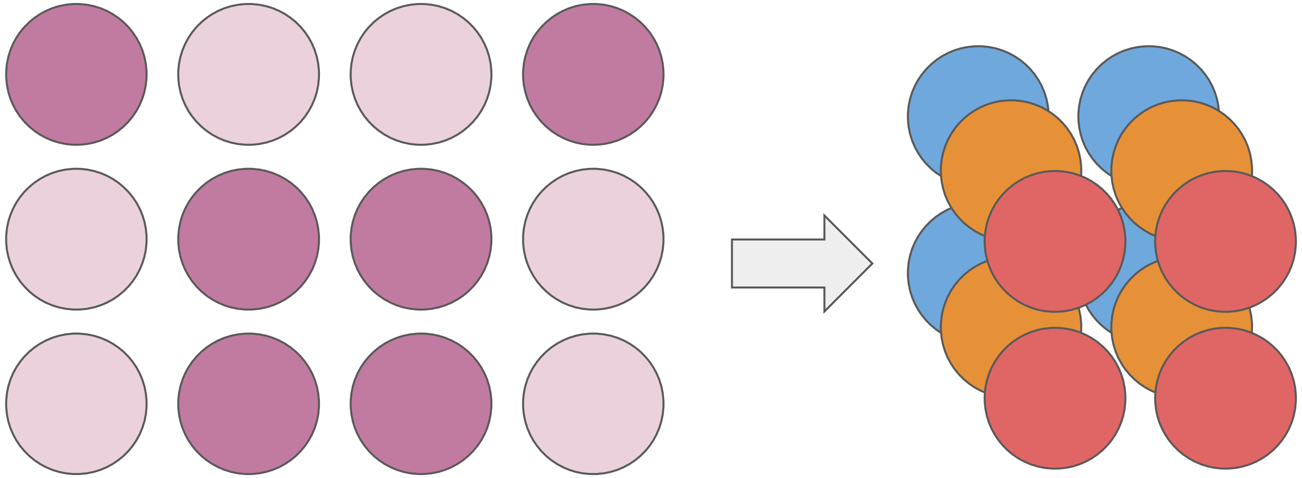
Input Image

Deep Learning



Input
Image

Deep Learning



Input
Image

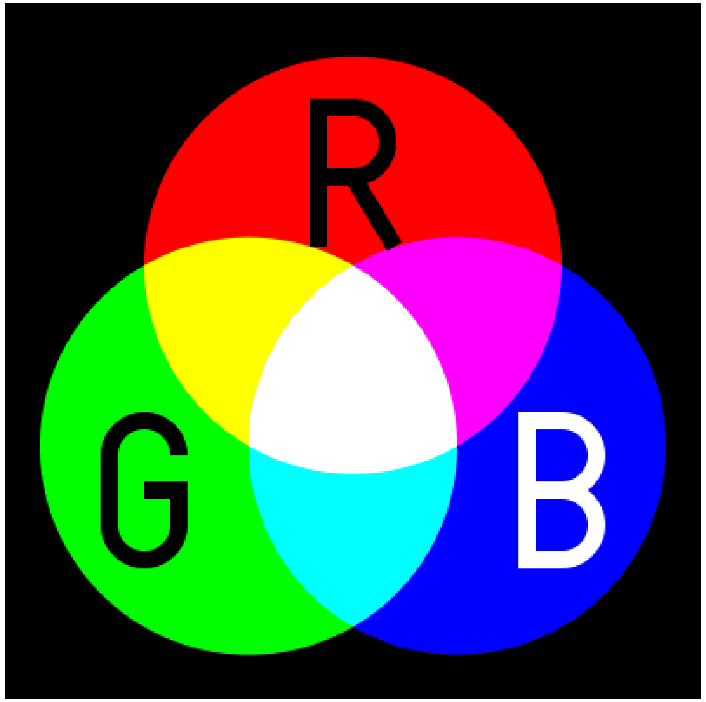
Convolutio
nal
Layer

Deep Learning

- But this was just in 2D for grayscale images.
- What about color images?
- Color Images can be thought of as 3D Tensors consisting of Red, Green, and Blue color channels.

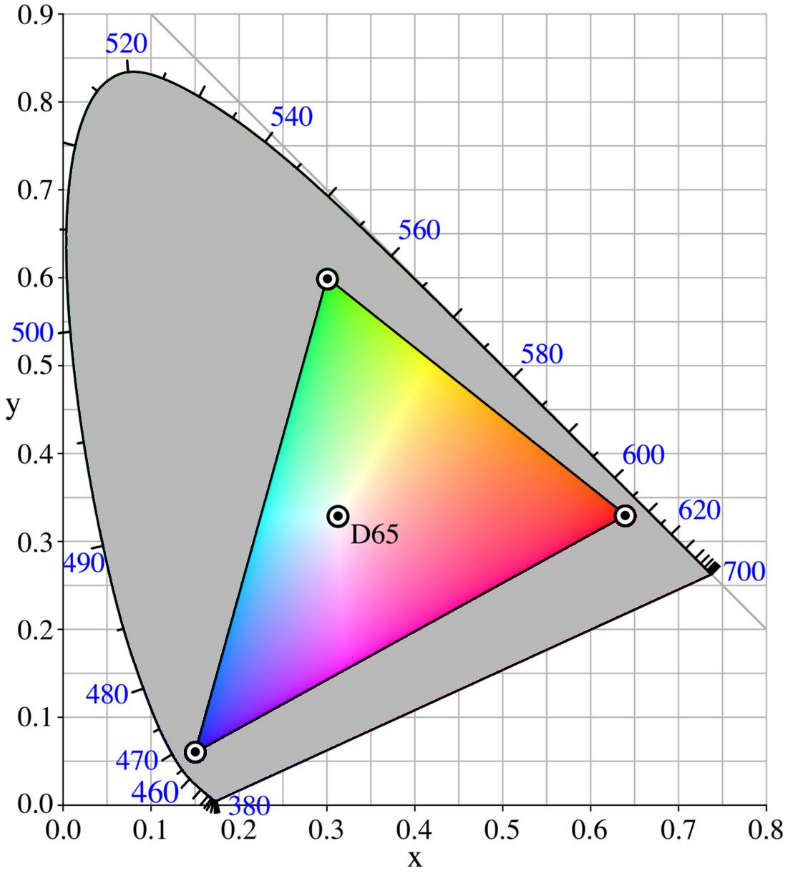
Deep Learning

- Additive color mixing allows us to represent a wide variety of colors by simply combining different amounts or R, G, B.



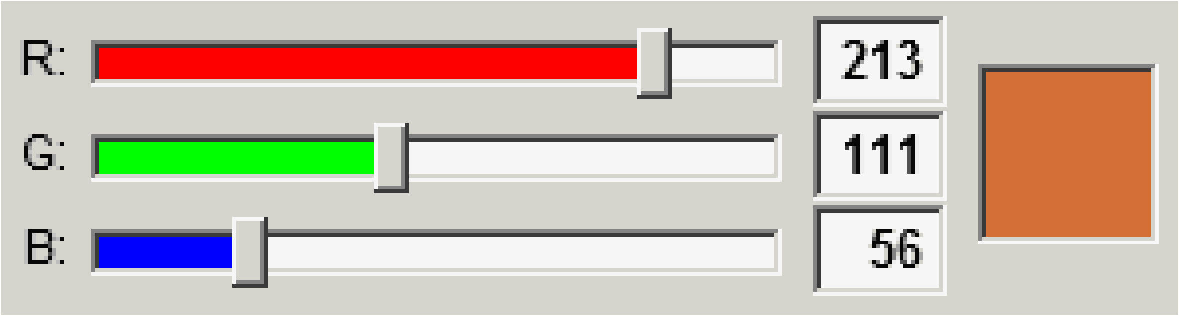
Deep Learning

- RGB allows to produce a range of colors

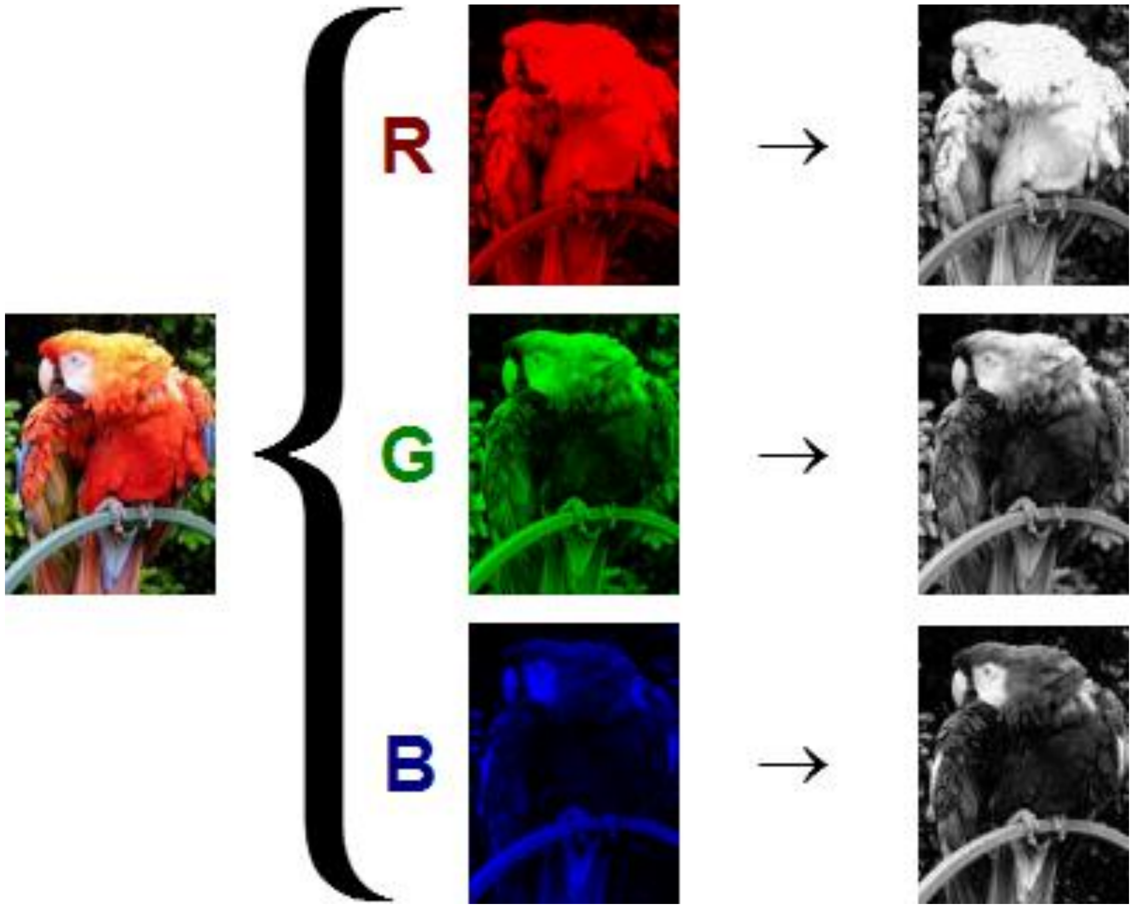


Deep Learning

- Each color channel will have intensity values.
- You may have already seen this sort of representation in other software with RGB sliders.



Deep Learning



- The shape of the color array then has 3 dimensions.
- Height
- Width
- Color Channels

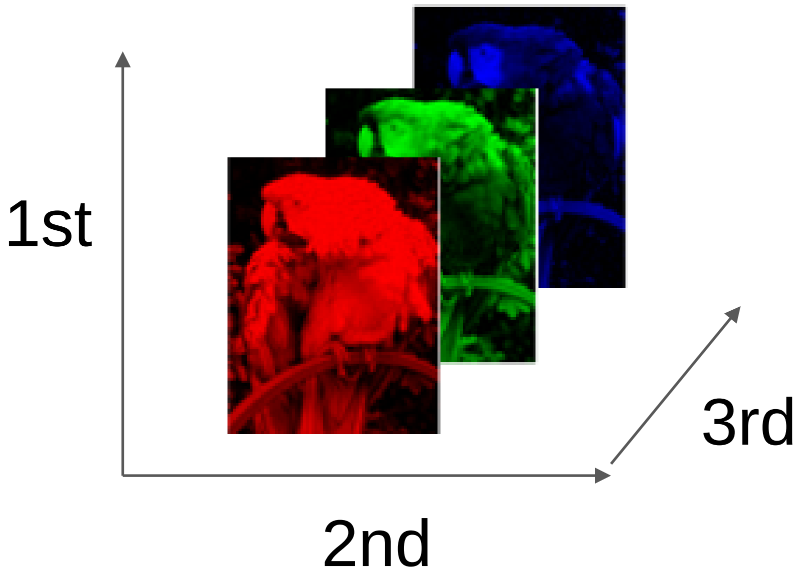
Deep Learning

This means when you read in an image and check its shape, it will look something like:

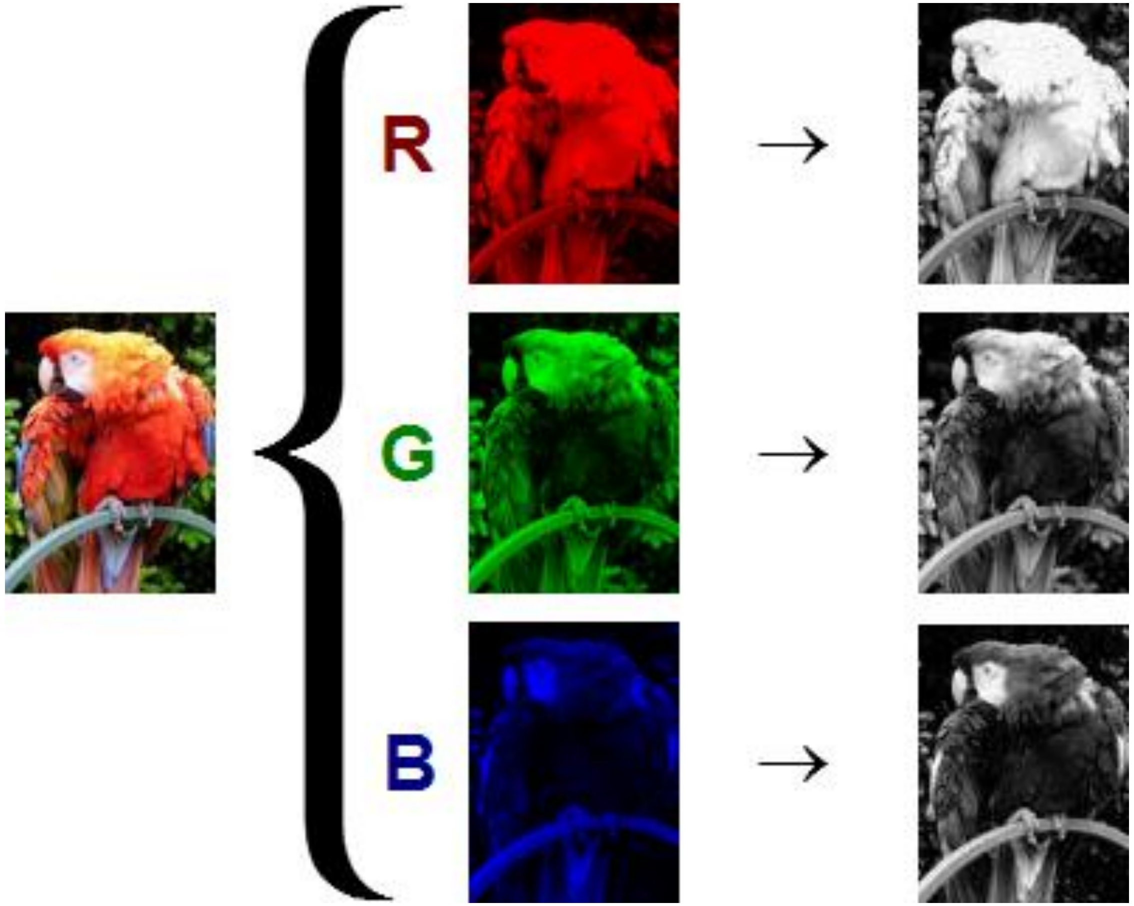
- **(1280,720,3)**
- **1280** pixel width
- **720** pixel height
- **3** color channels

Deep Learning

- This means when you read in an image and check its shape, it will look something like:
  - **(720,1280,3)**
    - **720** pixel height
    - **1280** pixel width
    - **3** color channels



Deep Learning

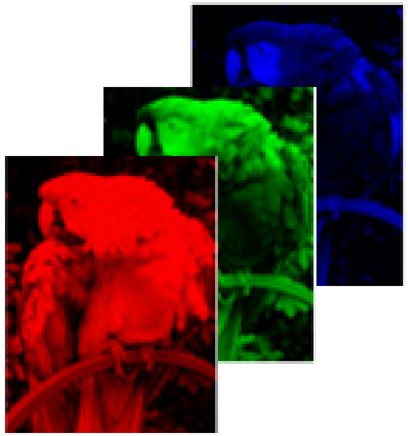


Keep in mind the computer won’t “know” a channel is Red, it just knows that there are now 3 intensity channels.

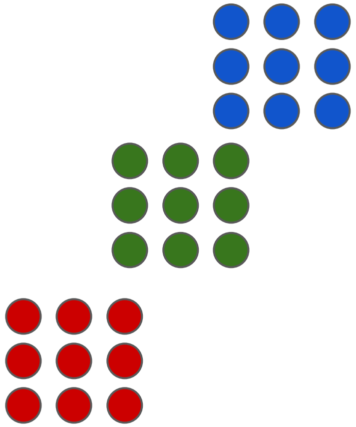
Deep Learning

- How do we then perform a convolution on a color image?
- We end up with a 3D filter, with values for each color channel.

Deep Learning

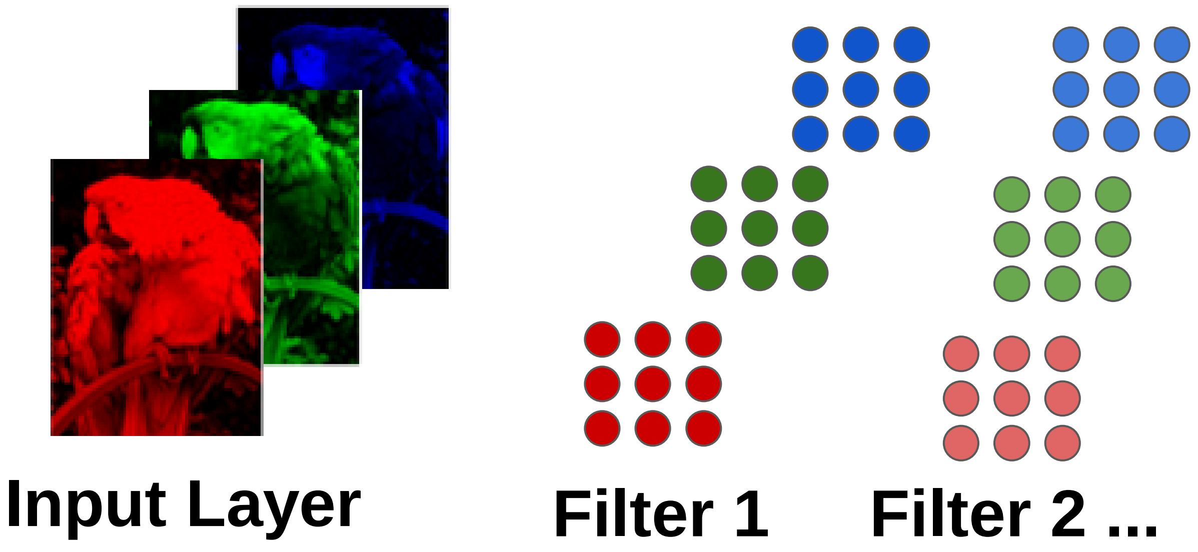


Input Layer



Filter

Deep Learning



Deep Learning

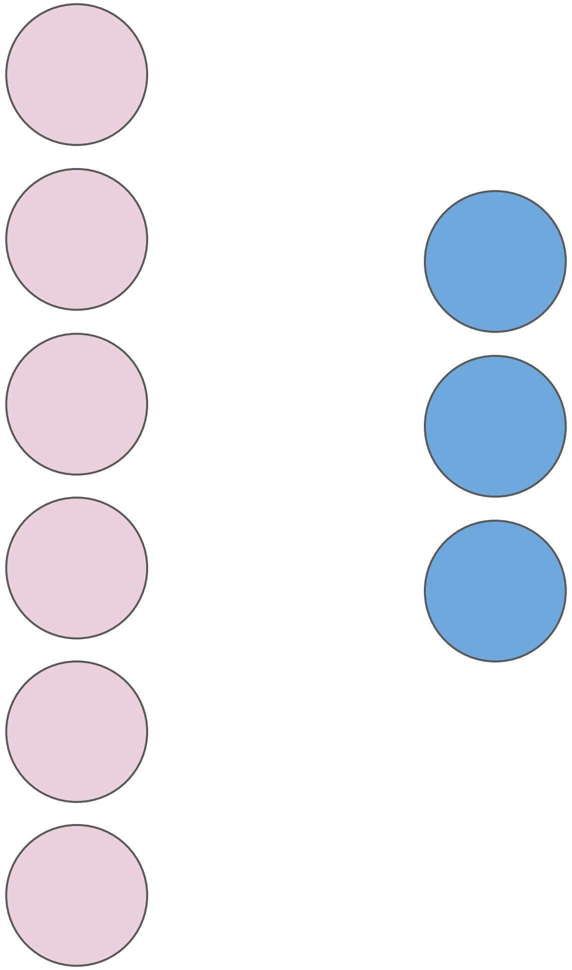
- Often convolutional layers are fed into another convolutional layer.
- This allows the networks to discover patterns within patterns, usually with more complexity for later convolutional layers.

Deep Learning

- We’ve learned a lot so far and we have one final theory topic to cover before coding our own CNNs, and that is the Pooling Layer! (also known as a downsampling layer)

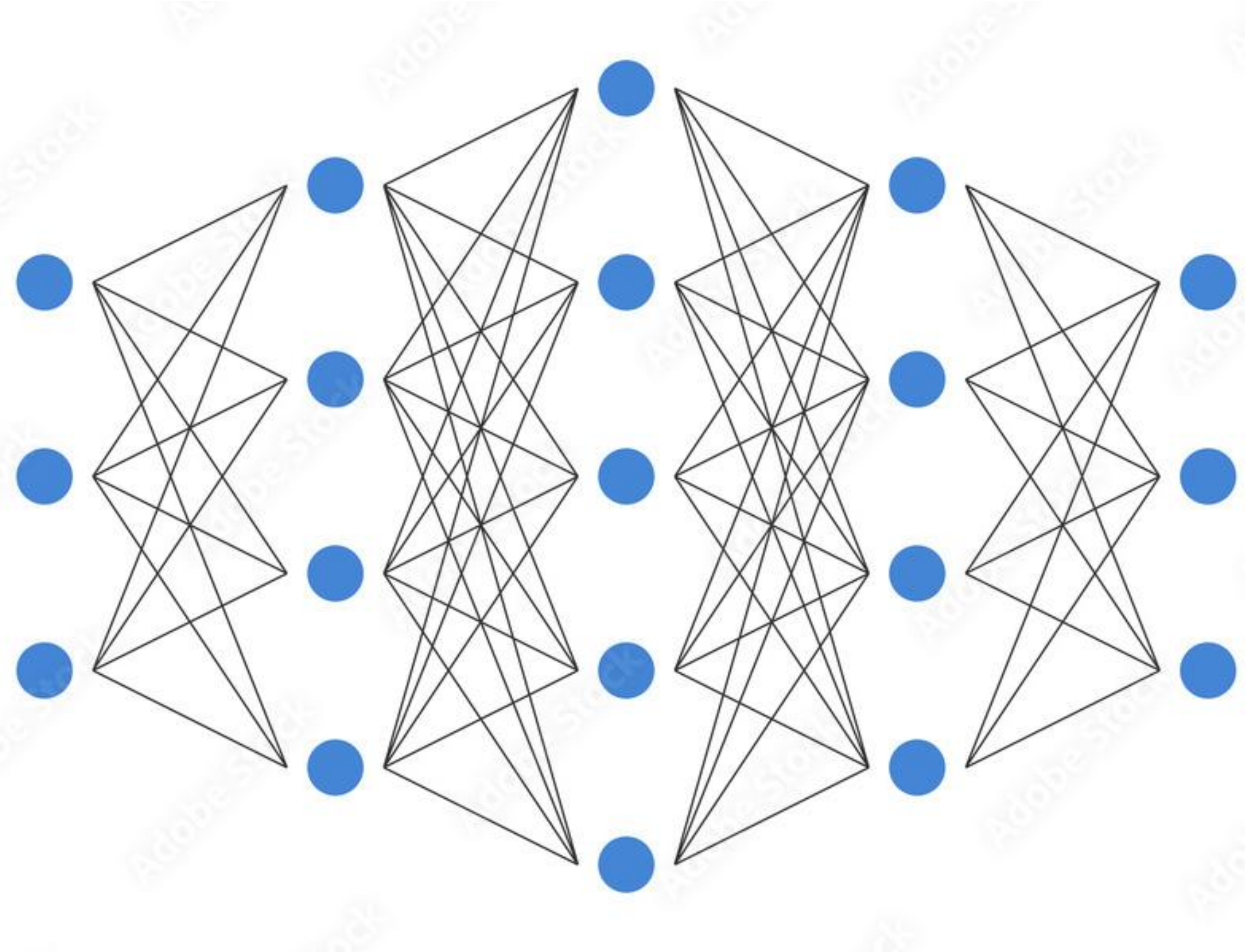
Deep Learning

- ANN



ICTC International Council for Technology Certifications

Pooling Layers



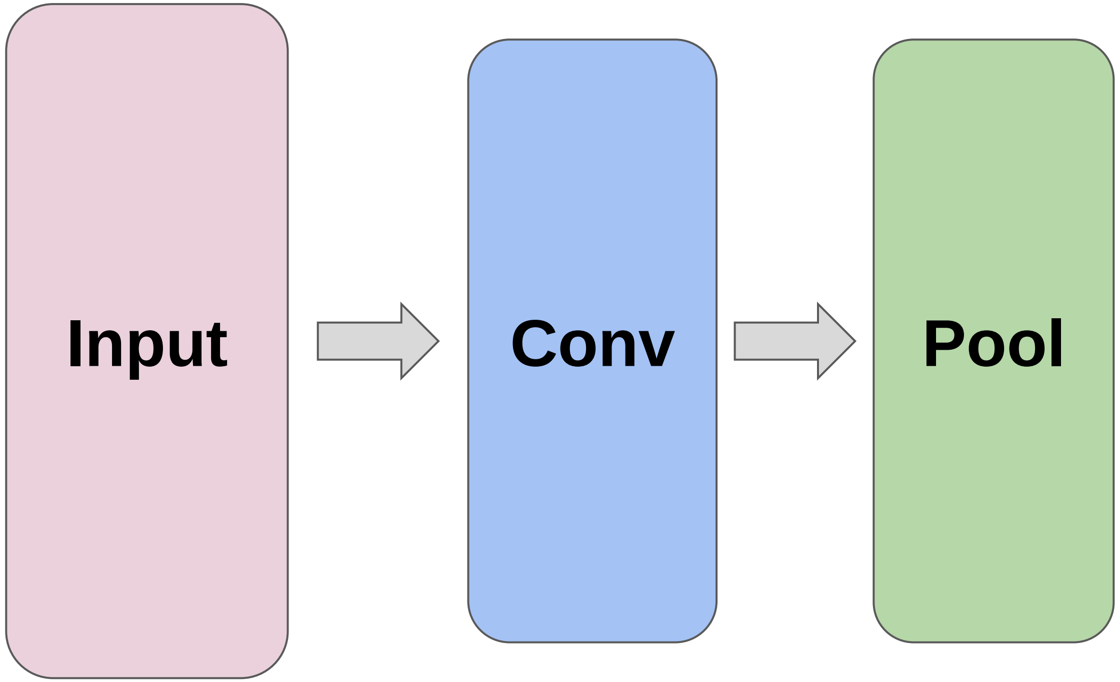
Deep Learning

Deep Learning

- Even with local connectivity, when dealing with color images and possibly 10s or 100s of filters we will have a large amount of parameters.
- We can use pooling layers to reduce this.

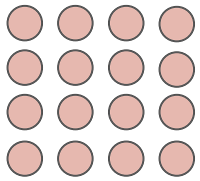
Deep Learning

- Pooling layers accept convolutional layers as input:



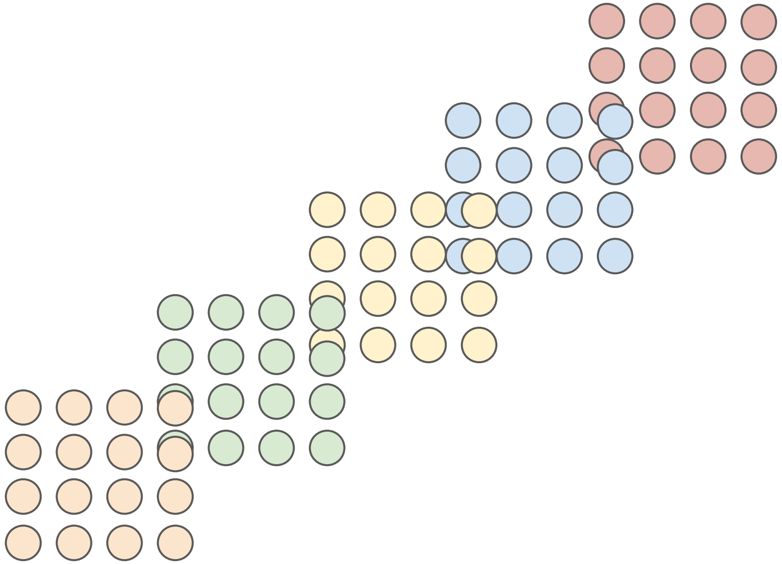
Deep Learning

- Our convolutional layers will often have many filters



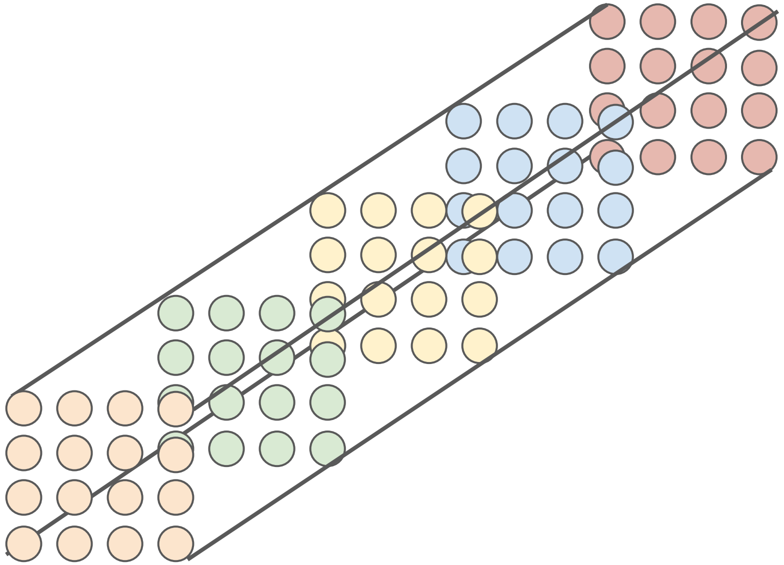
Deep Learning

- Our convolutional layers will often have many filters



Deep Learning

- Our convolutional layers will often have many filters



Deep Learning

- Let’s imagine a filter

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

Deep Learning

- We can reduce the size with subsampling

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

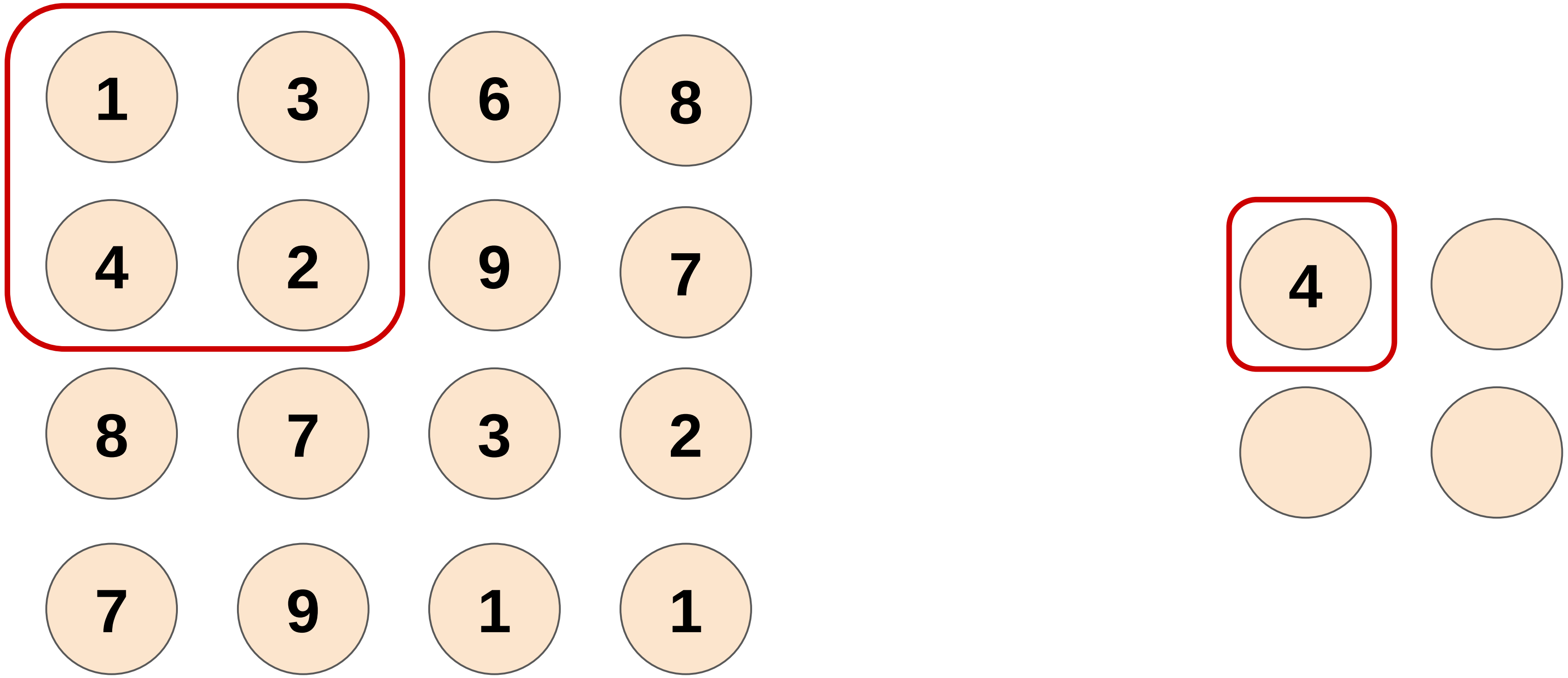
Deep Learning

- We can reduce the size with subsampling

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

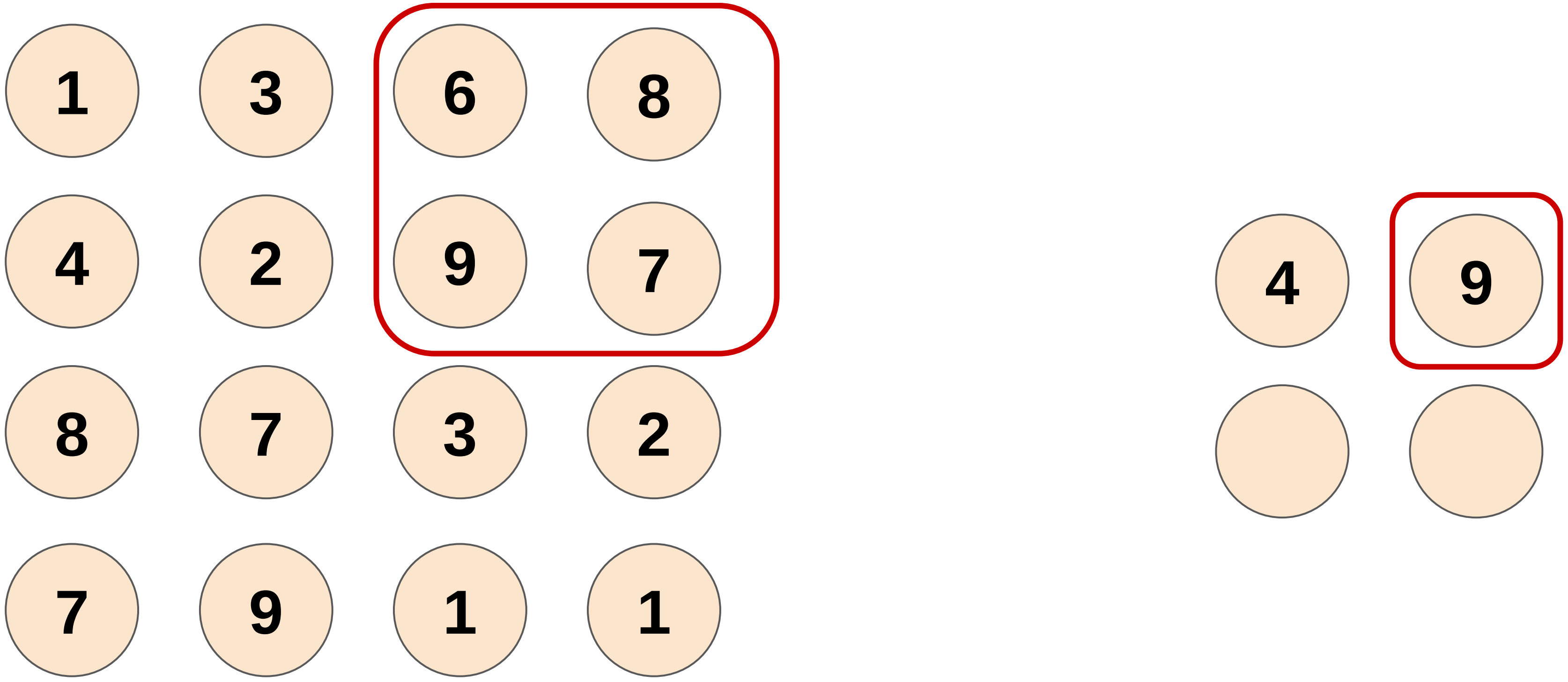
Deep Learning

- Max Pooling: Window 2x2 , Stride: 2



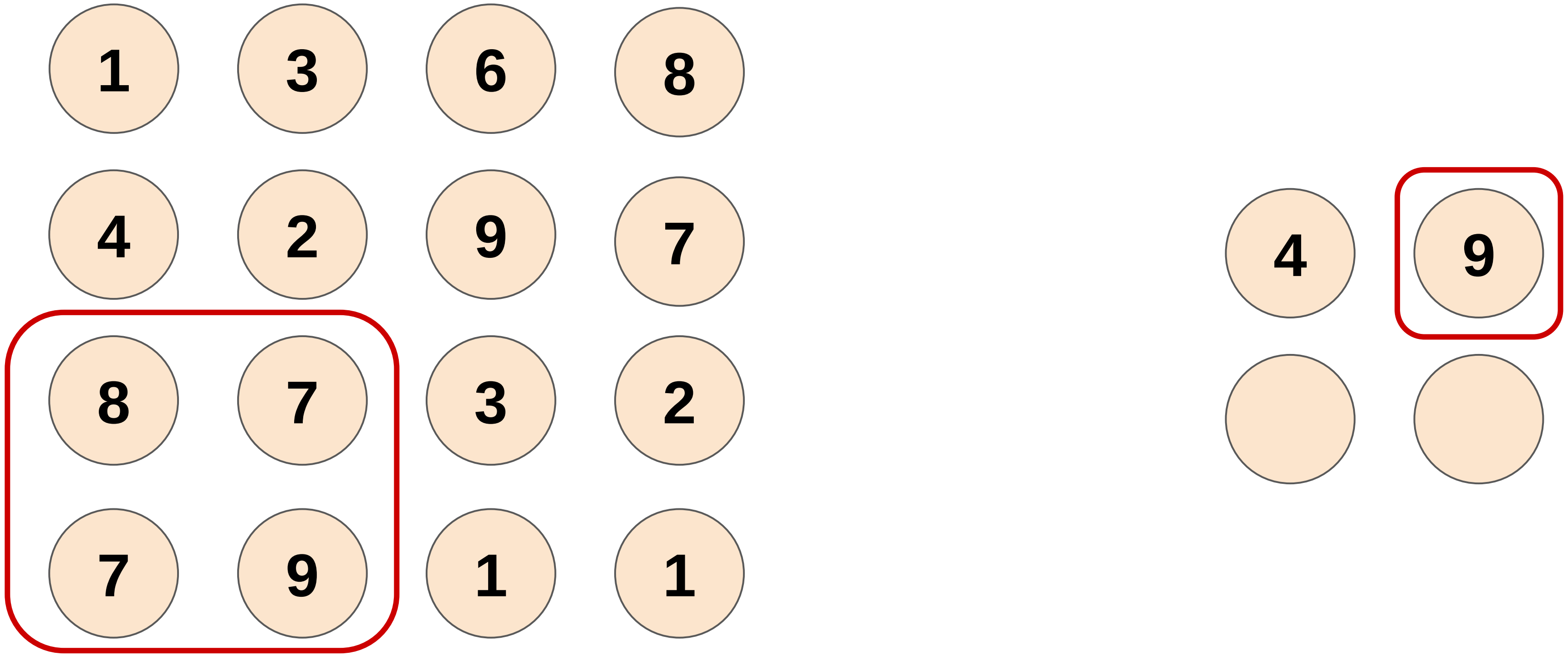
Deep Learning

- Max Pooling: Window 2x2 , Stride: 2



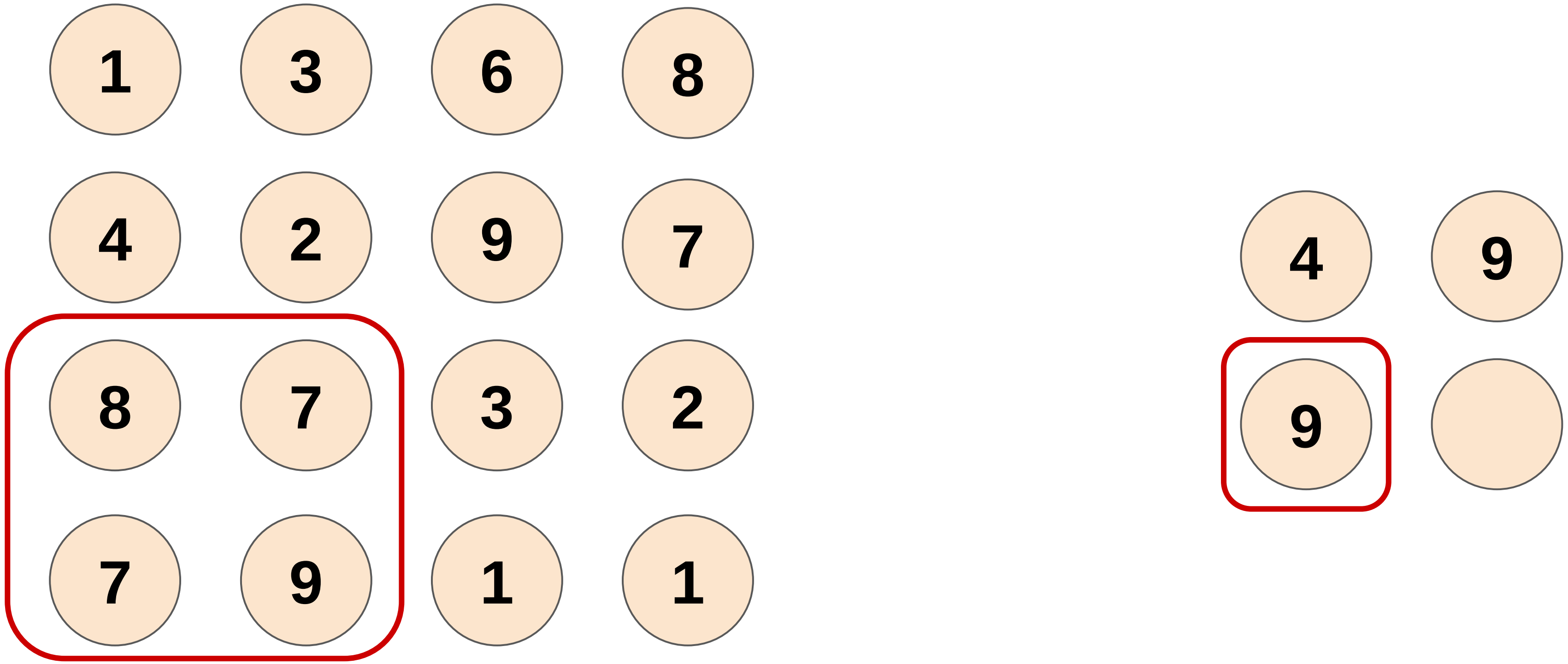
Deep Learning

- Max Pooling: Window 2x2 , Stride: 2



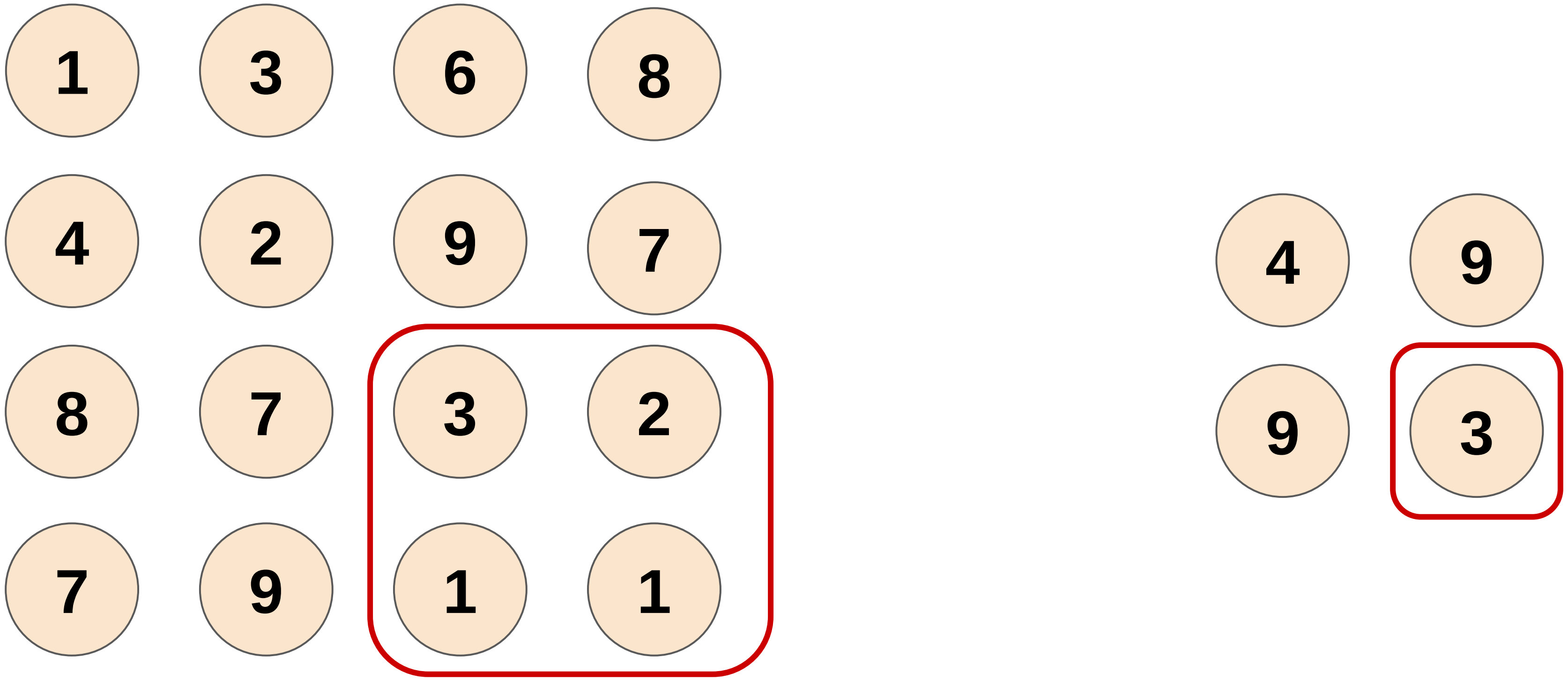
Deep Learning

- Max Pooling: Window 2x2 , Stride: 2



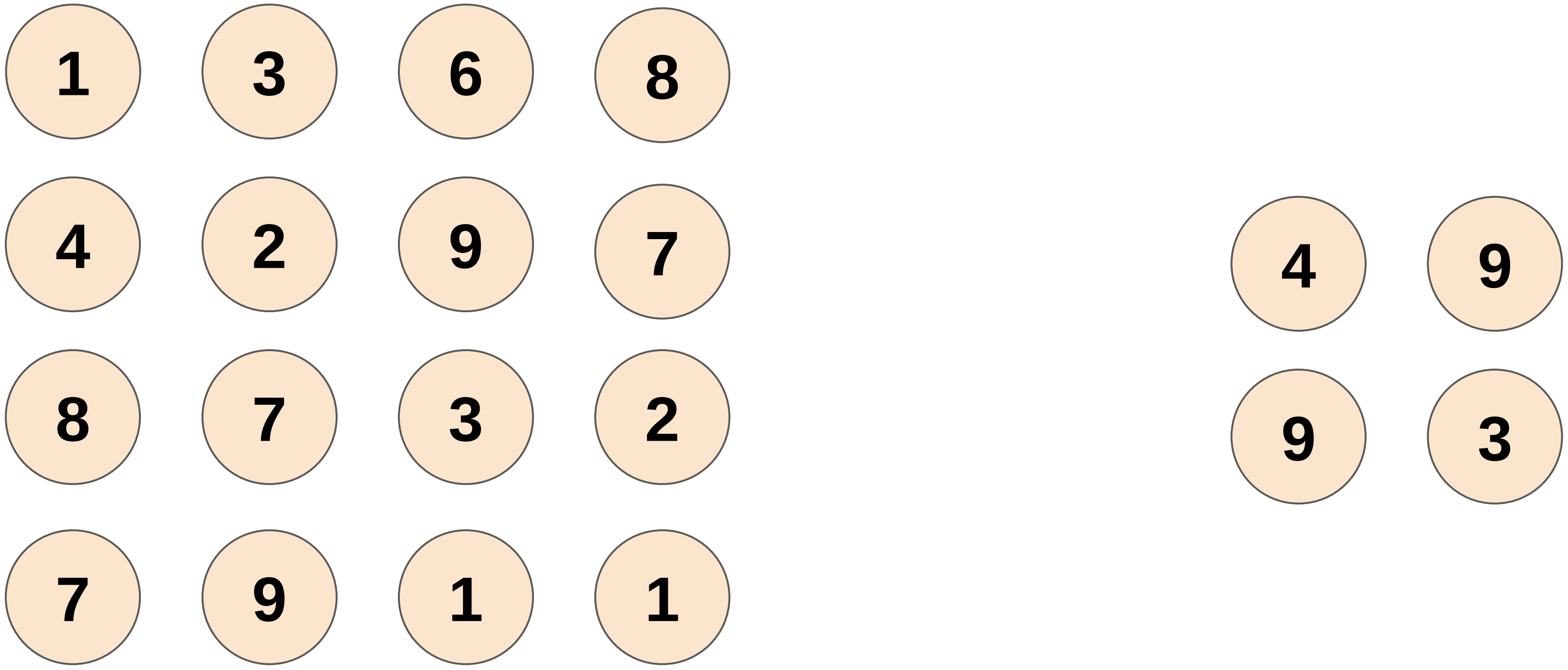
Deep Learning

- Max Pooling: Window 2x2 , Stride: 2



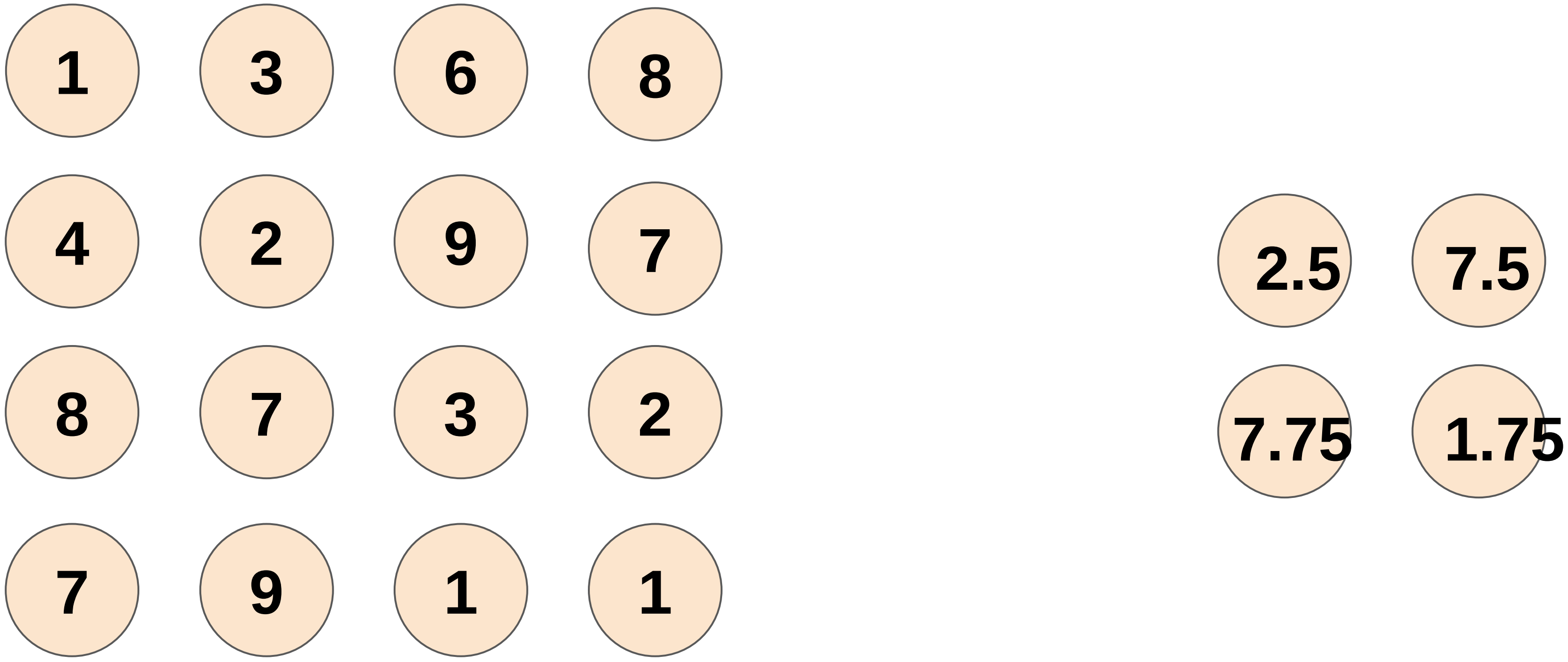
Deep Learning

- Max Pooling: Window 2x2 , Stride: 2



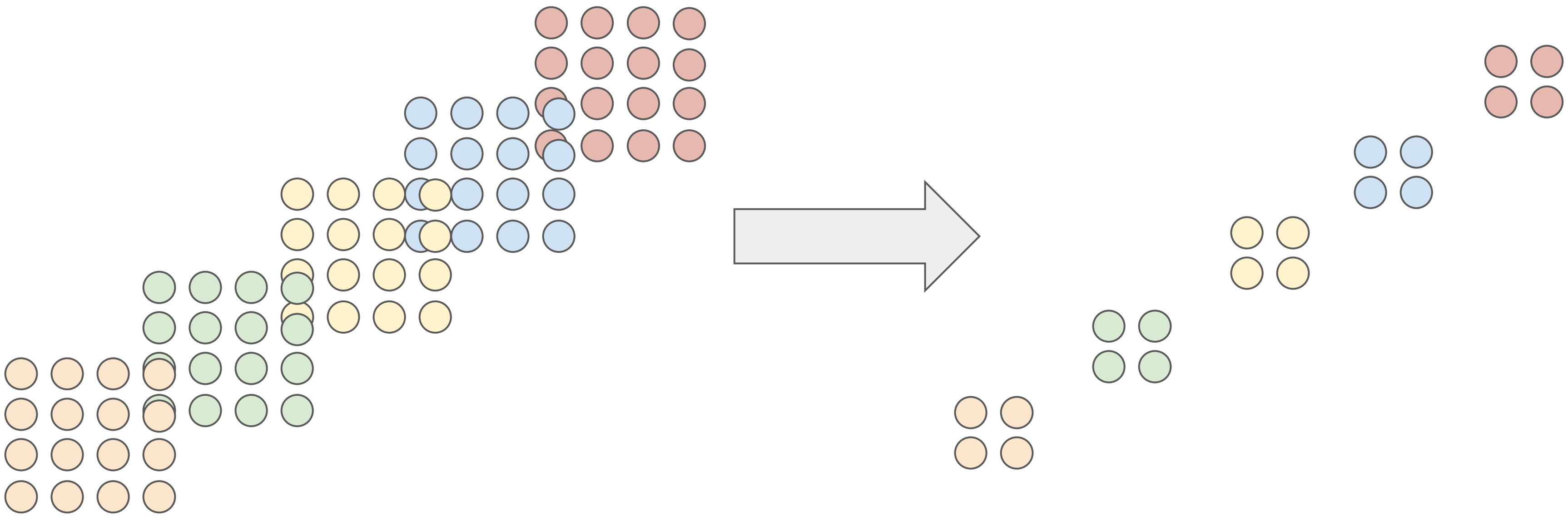
Deep Learning

- Average Pooling: Window 2x2 , Stride: 2



Deep Learning

- This greatly reduces our number of parameters!



Deep Learning

- This pooling layer will end up removing a lot of information, even a small pooling “kernel” of 2 by 2 with a stride of 2 will remove 75% of the input data.

Deep Learning

- Another common technique deployed with CNN is called "Dropout"
- Dropout can be thought of as a form of regularization to help prevent overfitting.
- During training, units are randomly dropped, along with their connections.

Deep Learning

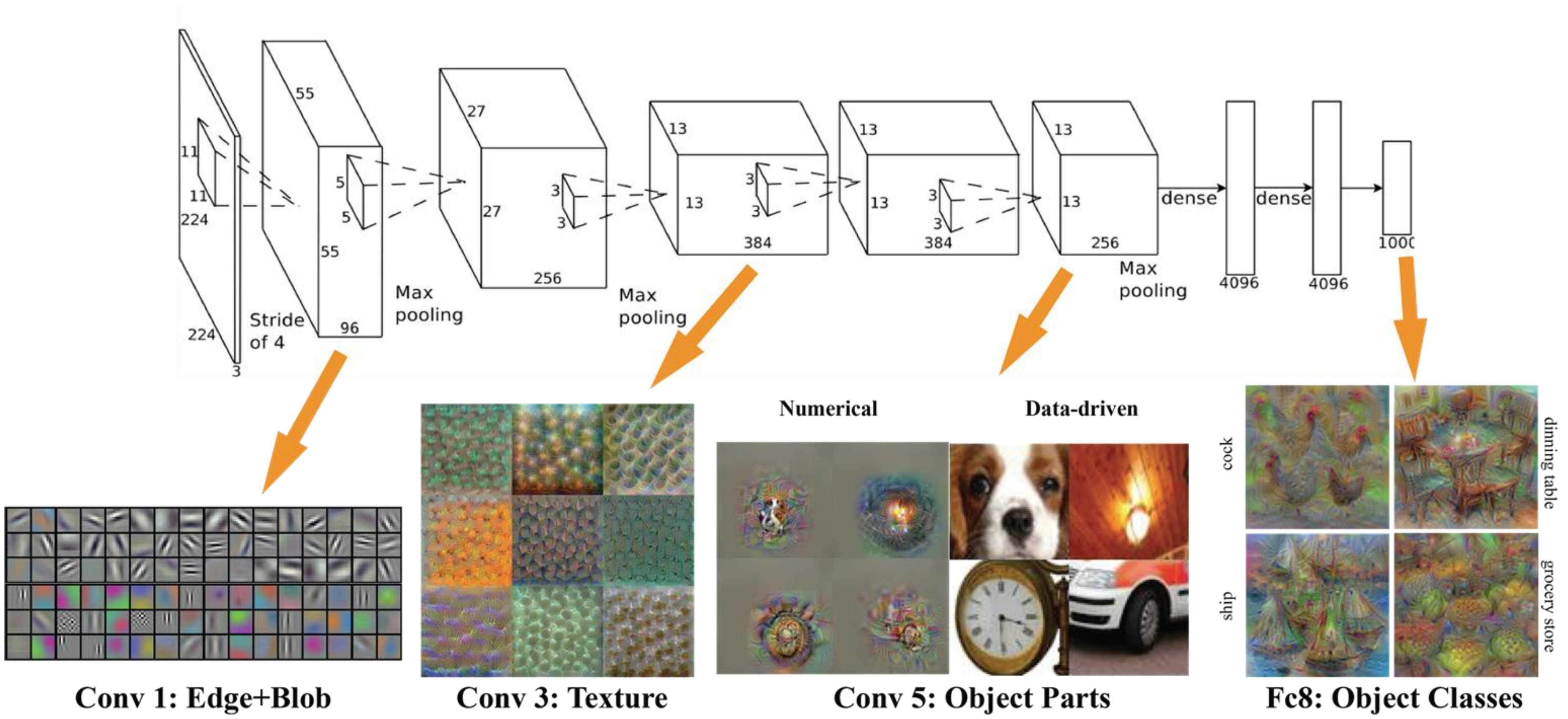
- This helps prevent units from “co-adapting” too much.
- Let’s also quickly point out some famous CNN architectures!

Deep Learning

- LeNet-5 by Yann LeCun
- AlexNet by Alex Krizhevsky et al.
- GoogLeNet by Szegedy at Google Research
- ResNet by Kaiming He et al.
- Check out the resource links to the papers discussing these architectures!

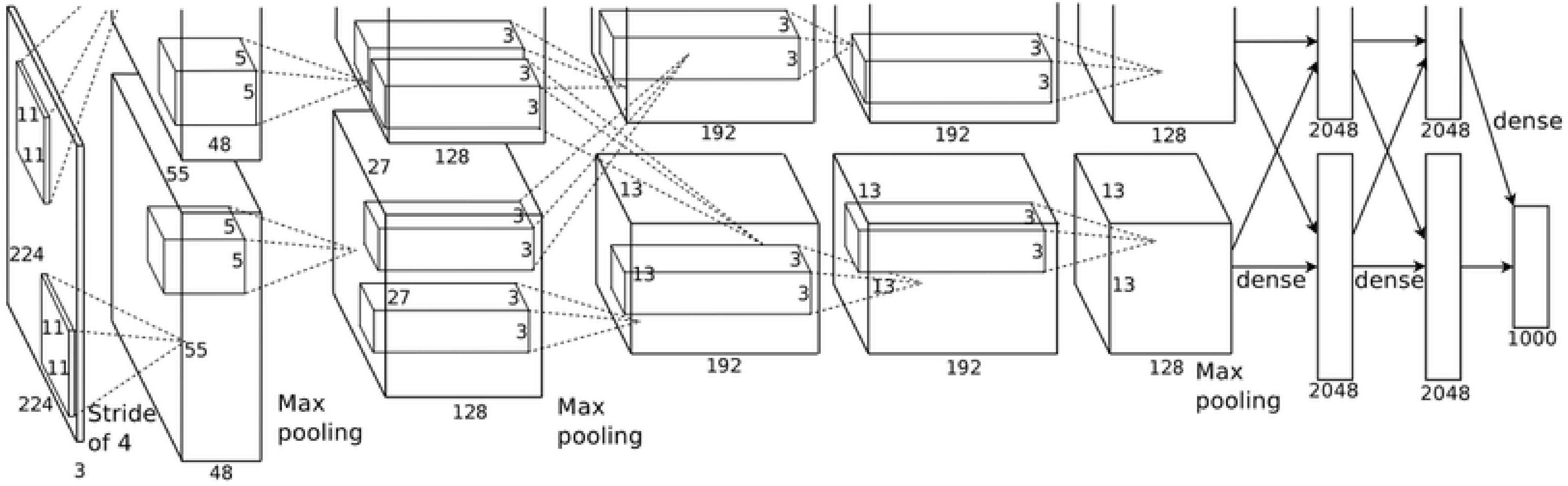
Deep Learning

- AlexNet



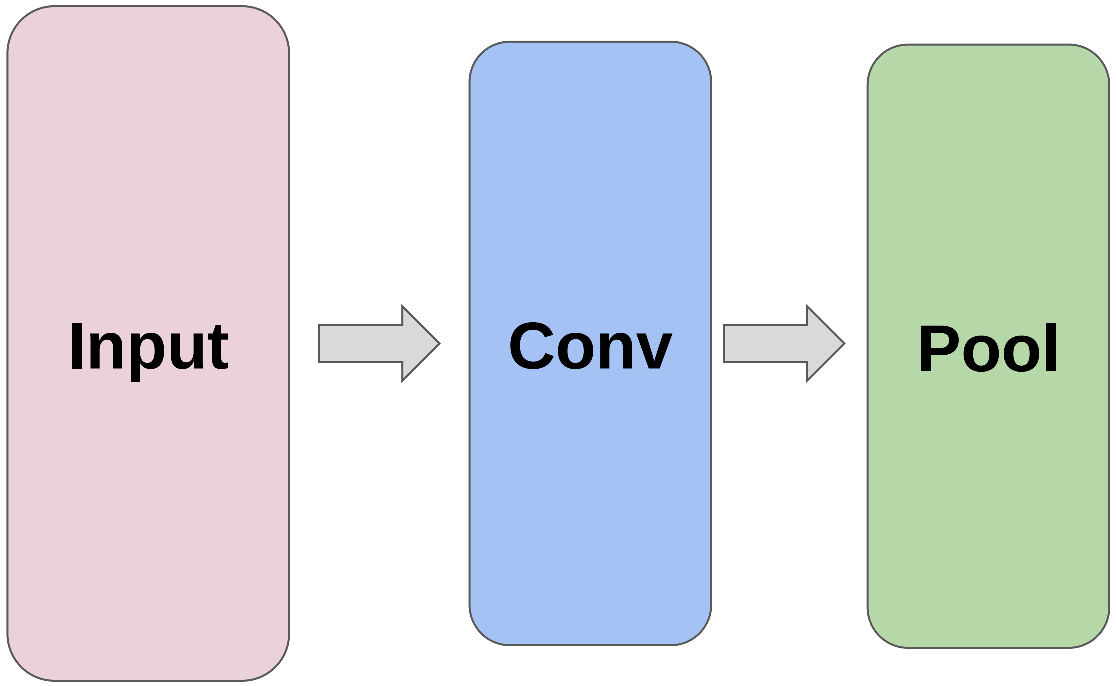
Deep Learning

- AlexNet



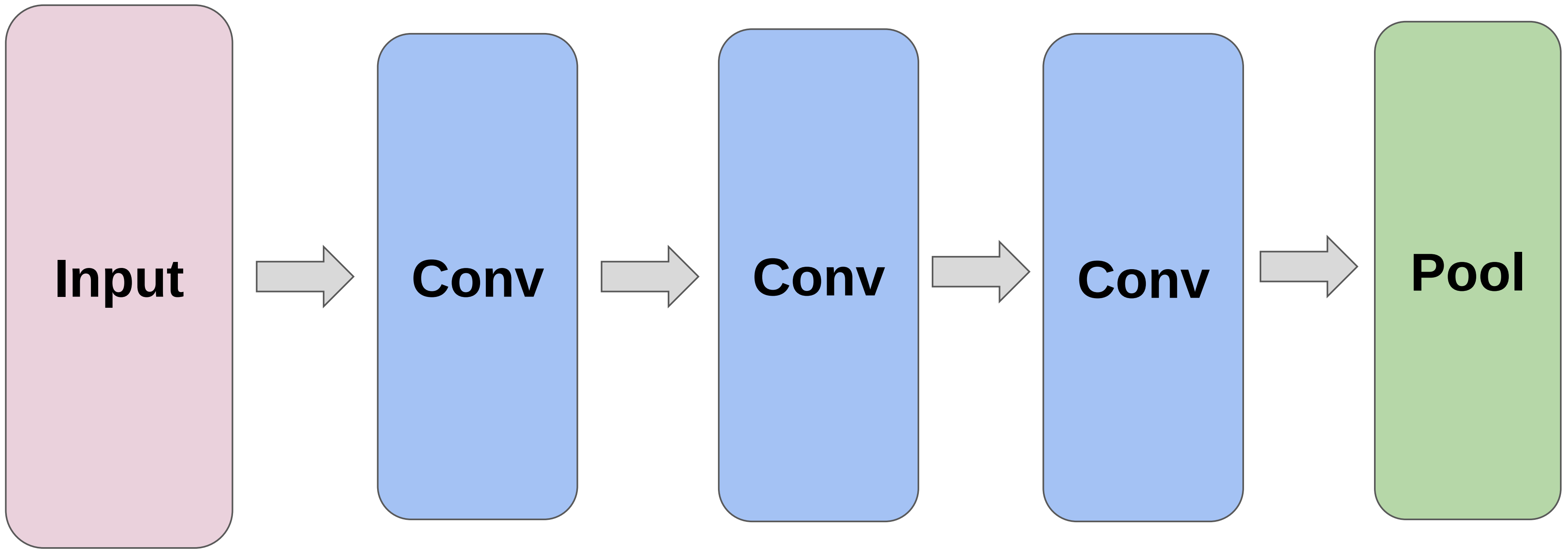
Deep Learning

- CNNs can have all types of architectures!



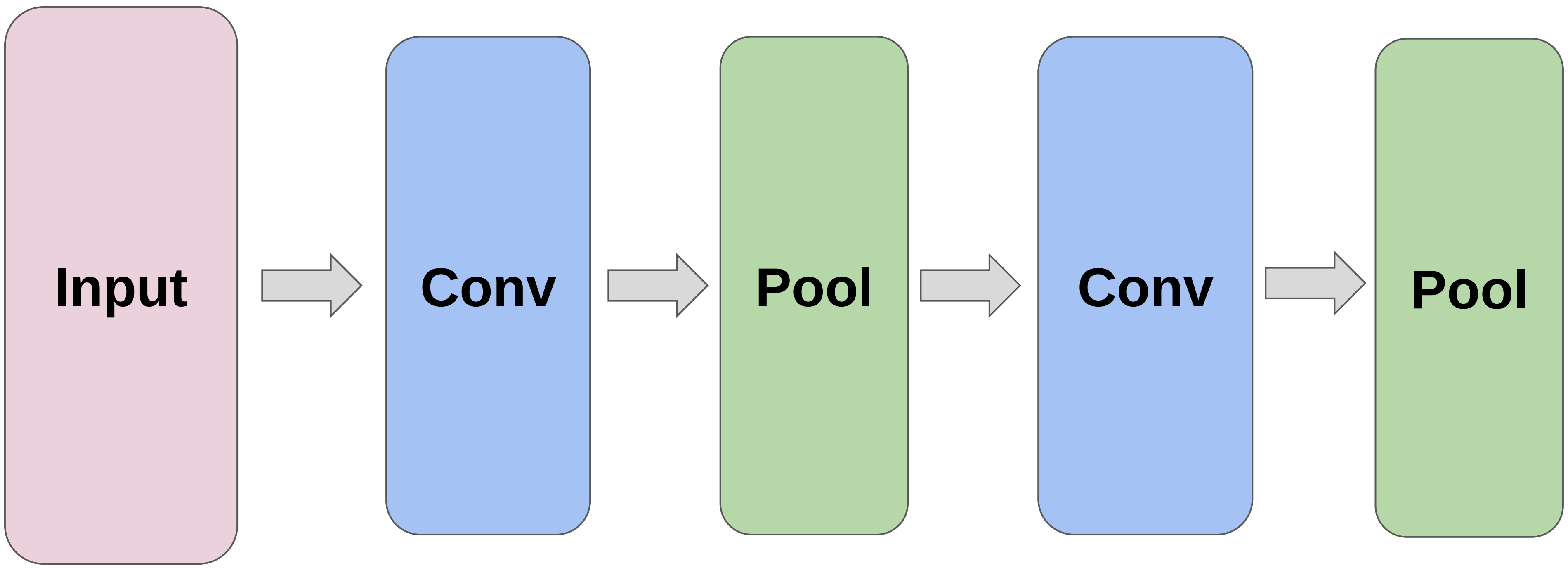
Deep Learning

- CNNs can have all types of architectures!



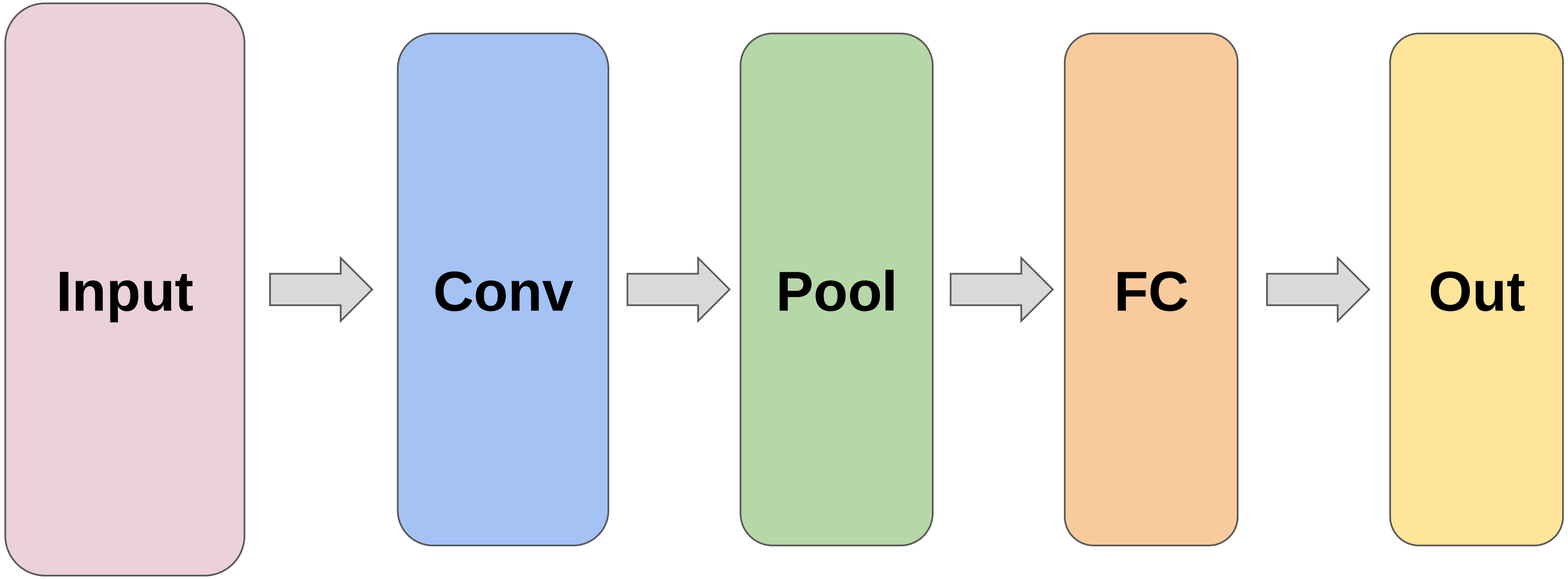
Deep Learning

- CNNs can have all types of architectures!



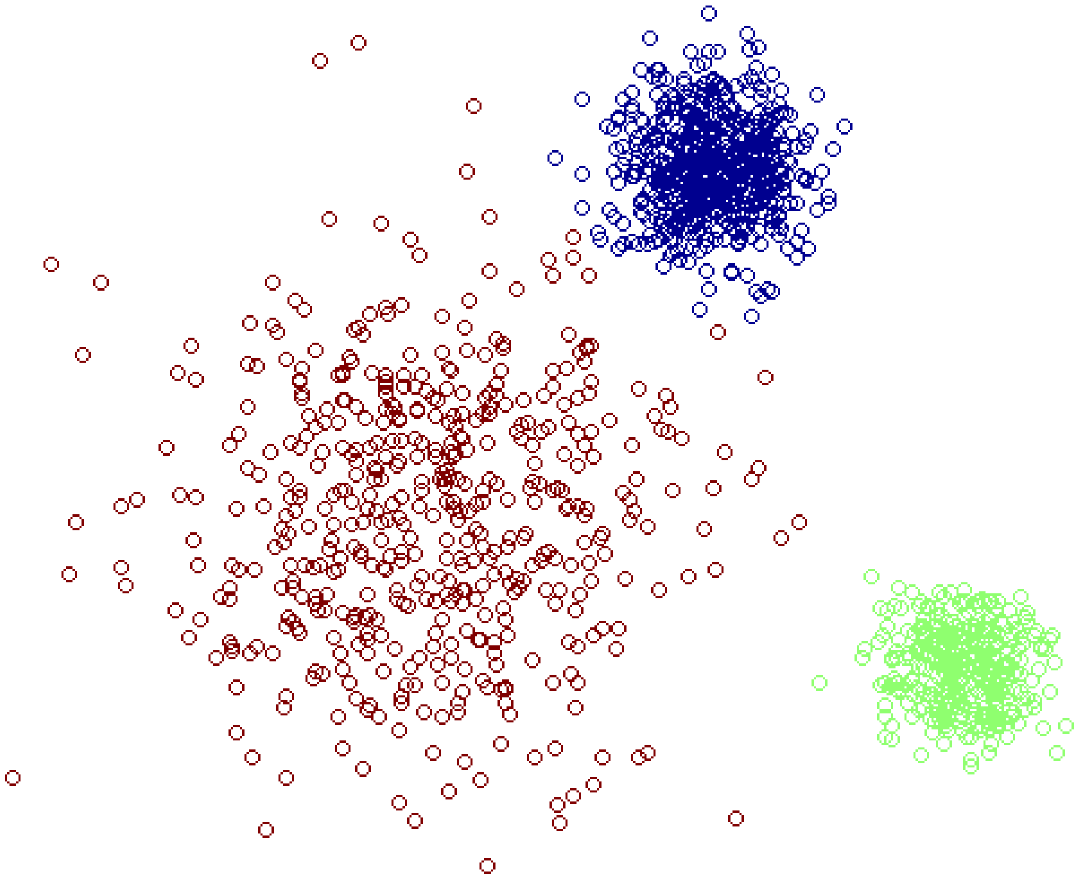
Deep Learning

- CNNs can have all types of architectures!



International Council for Technology Certifications

MNIST Data Revisited



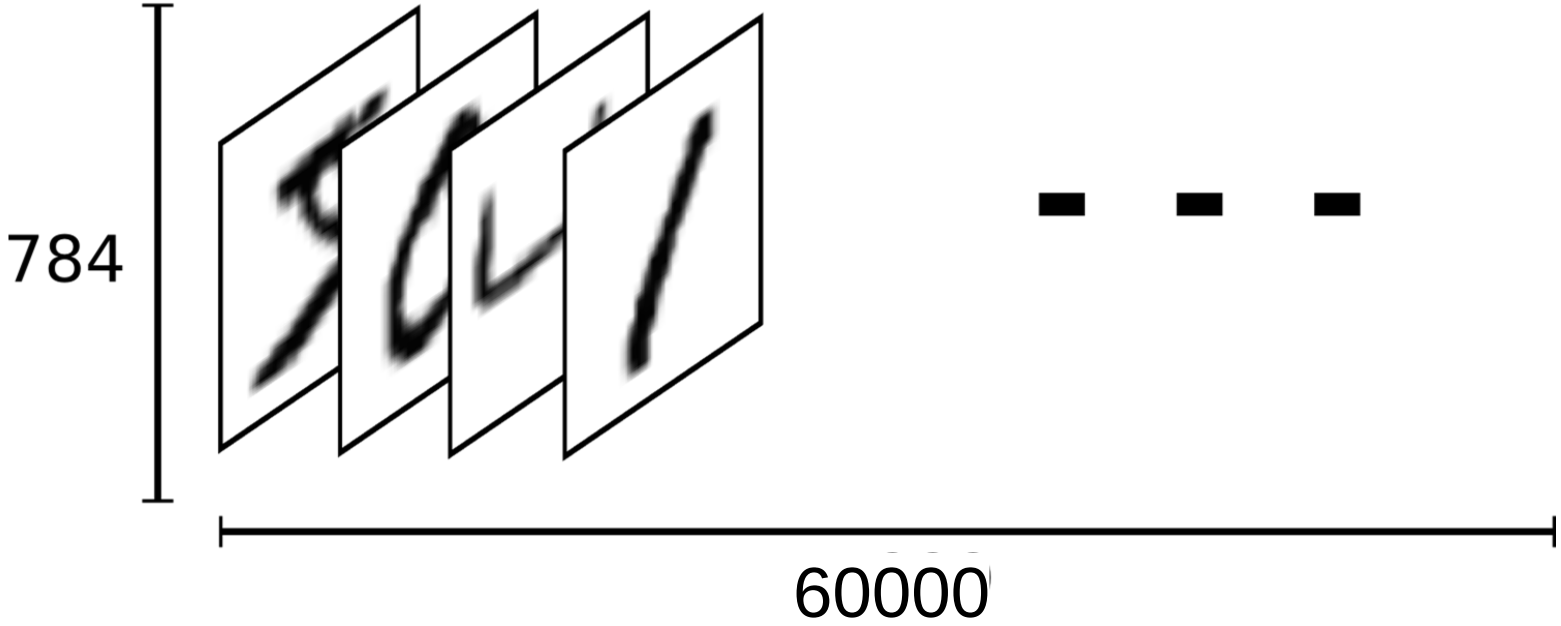
Deep Learning

Deep Learning

- Recall that flattening out the MNIST data caused us to lose 2D information.
- With CNNs, we can feed in the data as an array of 2D images.

Deep Learning

- We can think of the entire group of the 60,000 images as a tensor (an n-dimensional array)



Deep Learning

- For the labels we'll use One-Hot Encoding.
- This means that instead of having labels such as "One", "Two", etc… we'll have a single array for each image.

Deep Learning

- The label is represented based off the index position in the label array.
- The corresponding label will be a 1 at the index location and zero everywhere else.
- For example, 4 would have this label array:
    - [0,0,0,0,1,0,0,0,0,0]

Deep Learning

Keep in mind that when dealing with tensors of image data, we actually end up with 4 dimensions:

- Number of Images
- Height
- Width
- Color Channels

MNIST with CNN

PART ONE - The Data

MNIST with CNN

PART TWO - Model and Training

MNIST with CNN

PART THREE - Model Evaluation

CIFAR-10 with CNN

PART ONE - Data

Deep Learning

CIFAR-10 dataset are 32by32 images of 10 different objects:

- Airplane,Car,Bird,Cat,Deer,Dog,Frog, Horse,Ship,Truck
- These are color images!

Deep Learning

- In this program, we will be reusing a lot of the previous CNN code, so we will only focus on new additions due to the introduction of 3 color channels (RGB).

CIFAR-10 with CNN

PART TWO - Model Evaluation

International Council for Technology Certifications

Downloading Image Data

Deep Learning

Deep Learning

- So far we’ve only dealt with pre-packaged data sets such as MNIST and CIFAR-10.
- But what about working with real image files? Like .jpg or .png?

Deep Learning

- Let’s learn about TensorFlow’s built in tools for generating image data batches from directories with real files.
- First we need to download a large zip file of data.

Deep Learning

- This dataset is available as a zip file called **cell_images.zip**
- The link is a supplemental resource in this lecture, it is also within the same Google Drive resource as the slides.
- Let’s explore getting the data!

CNN on Custom Images

Part 1 - The Data

CNN on Custom Images

Part 2 Image Data Generator

CNN on Custom Images

Part 3 Creating the Model

CNN on Custom Images

Part 4 Model Evaluation

Let’s get started!

CNN Exercises

CNN Exercises Solutions