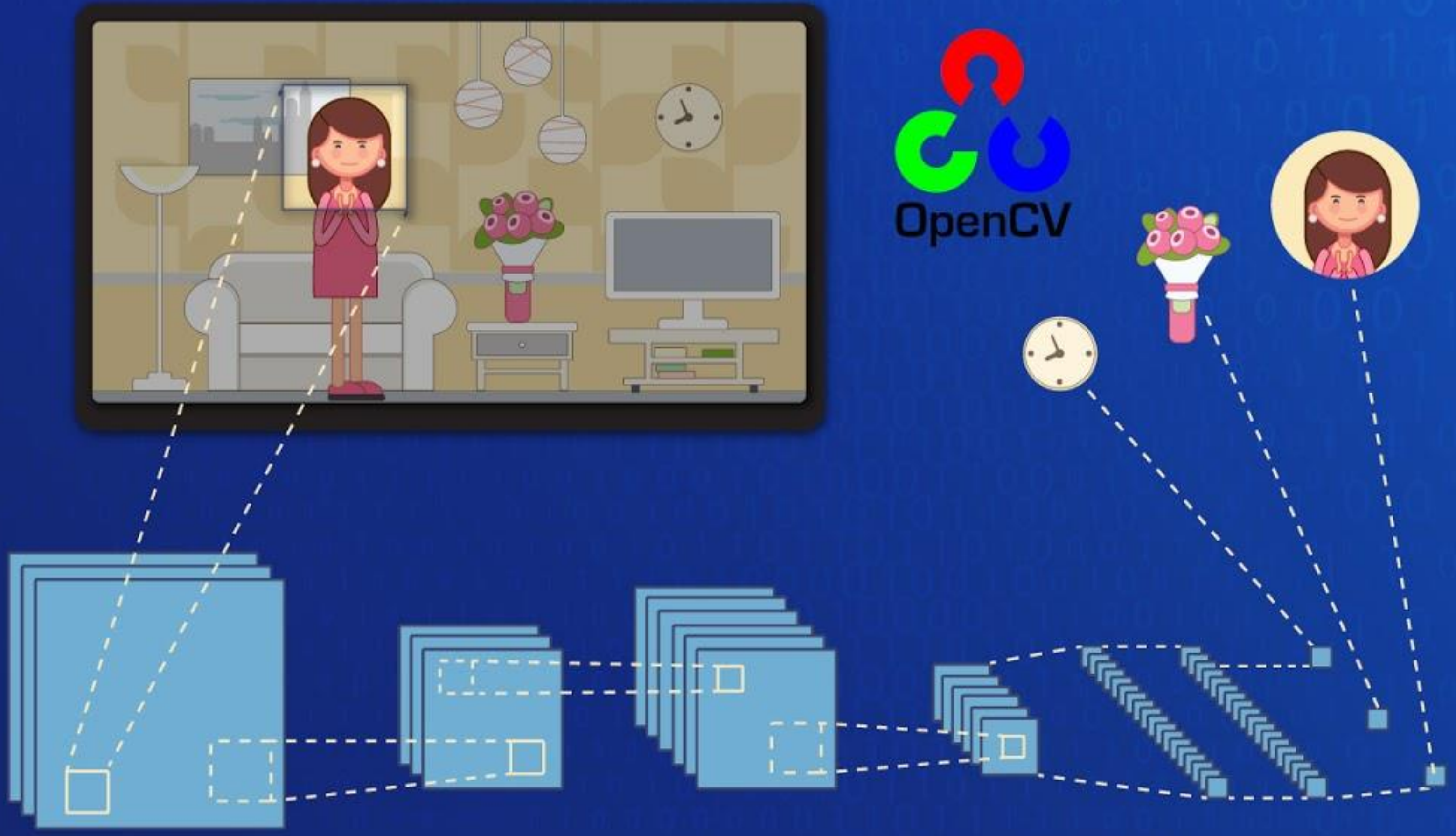
تركيب

THE**ASSEMBLY**

DATA SCIENCE



SPEAKER

ANUDEEP SEKHAR

AI/COMPUTER VISION SPECIALIST

NOVEMBER 30

DEEP LEARNING FOR COMPUTER VISION

About The Assembly

COMMUNITY
INNOVATION
WORKSPACE

- A smart lab based out of **in5 Tech** since Dec 2014
- Almost 300 free workshops done
- ASSEMBLY: HACK - Embedded systems, IoT and hardware
- ASSEMBLY: CODE - Software projects - APIs, frameworks, apps
- Age range: 16-60 - students, professionals, entrepreneurs
- Focus on smart technology and practical applications
- Forum: **members.theassembly.ae**

Tag Us On Social Media

FACEBOOK	The Assembly (@MakeSmartThings)
TWITTER	@MakeSmartThings
INSTAGRAM	@MakeSmartThings
YOUTUBE	The Assembly

Introducing The Assembly: Data Science

- Covers
  - **Data Science Platforms & Techniques**
  - **Machine Learning & Artificial Intelligence Paradigms**
- More sophisticated content
- Deep dive into the topics – lots of technical knowledge
- Hands on guidance from experts

GitHub Link

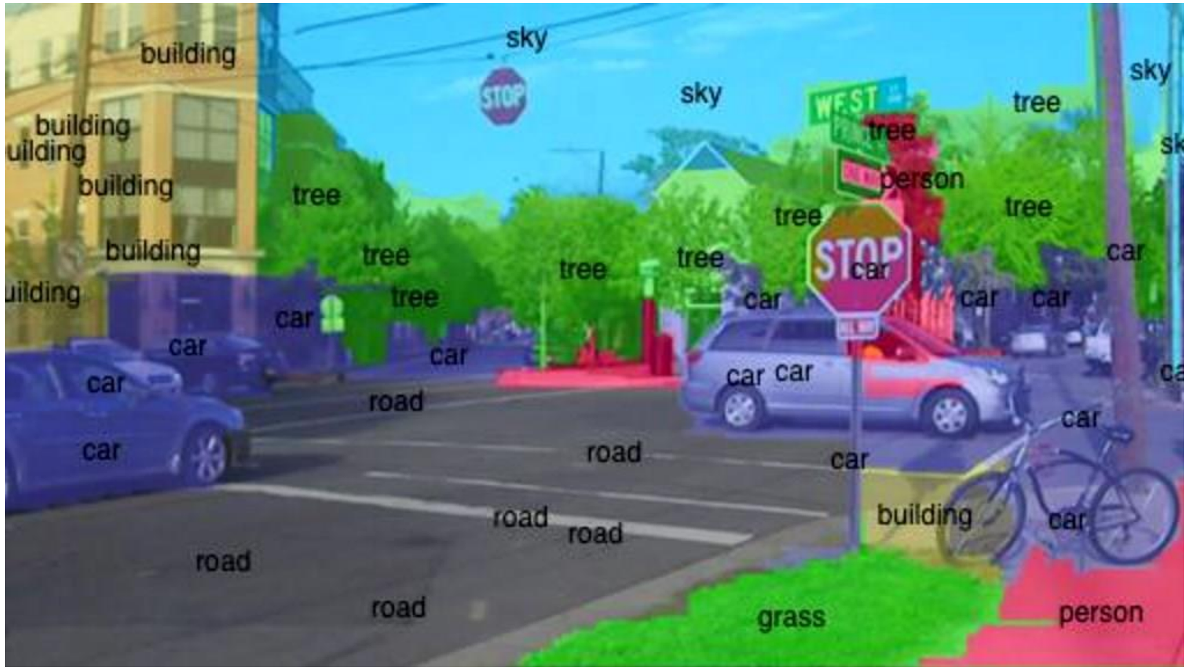
https://github.com/anudeepsekhar/The-Assembly-Computer-Vision-Worshop

If We Want Machines to Think, We Need to Teach Them to See

By Anudeepsekhar Bolimera

What is Computer Vision?

An interdisciplinary scientific field that deals with how computers can be made to gain high-level understanding from digital images or videos. From the perspective of engineering, it seeks to automate tasks that the human visual system can do.



How does it work?

The process involves

- Acquiring digital images
- Processing images to enhance features
- Analyzing & understanding the images
- Extracting high-dimensional data from the real world in order to produce numerical or symbolic information

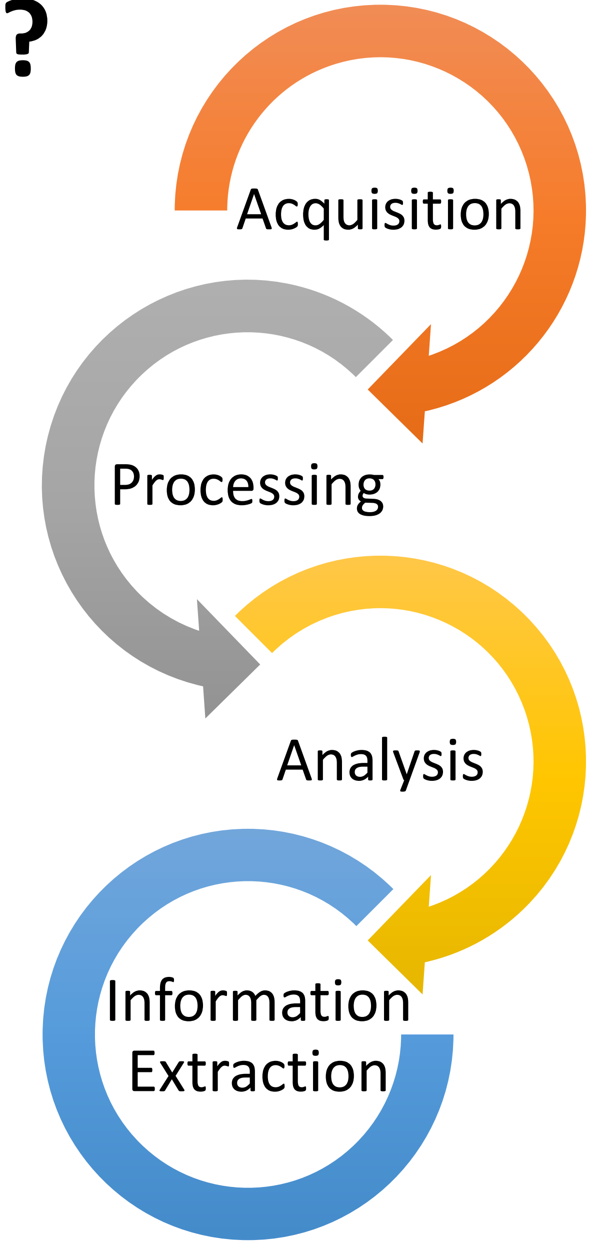
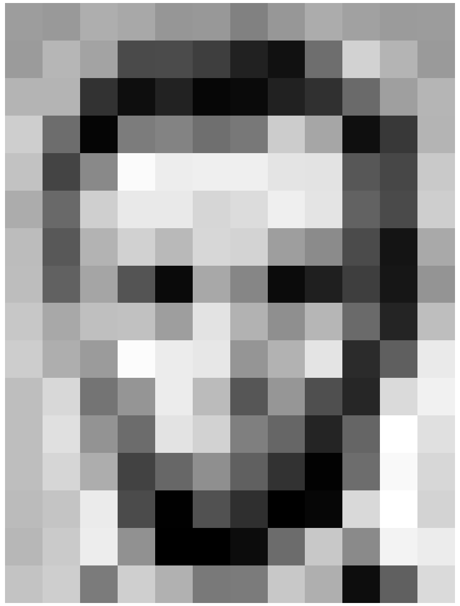
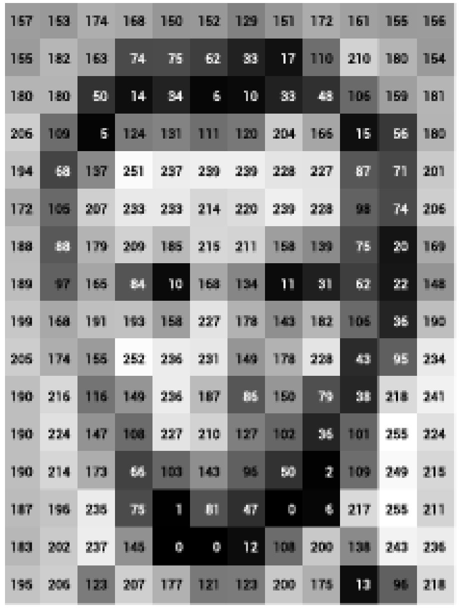
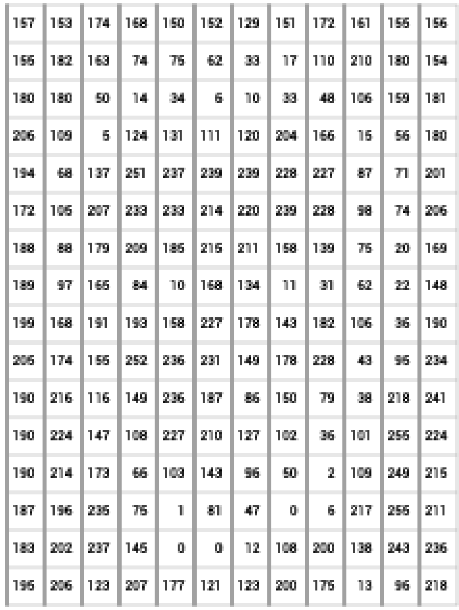


Image Acquisition

*The primary sensor machines use to see the world around them as images is **the camera**.* Machines interpret images (acquired from camera) very simply: **as a series of pixels**, each with their own set of color values.







157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Image Processing

Operations are performed on images to get intended manipulations – which will enhance useful information available in the images.

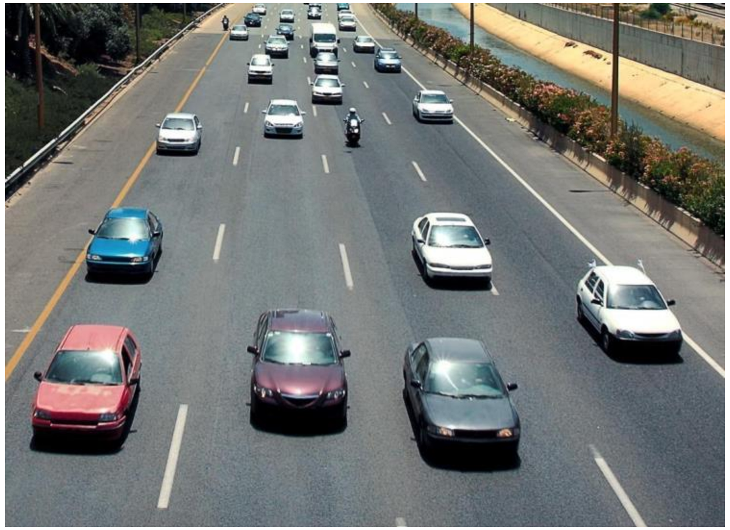


Image Processing

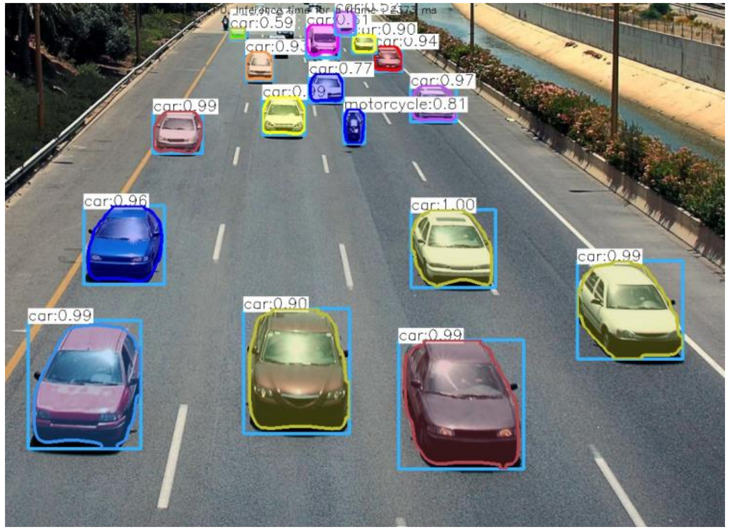
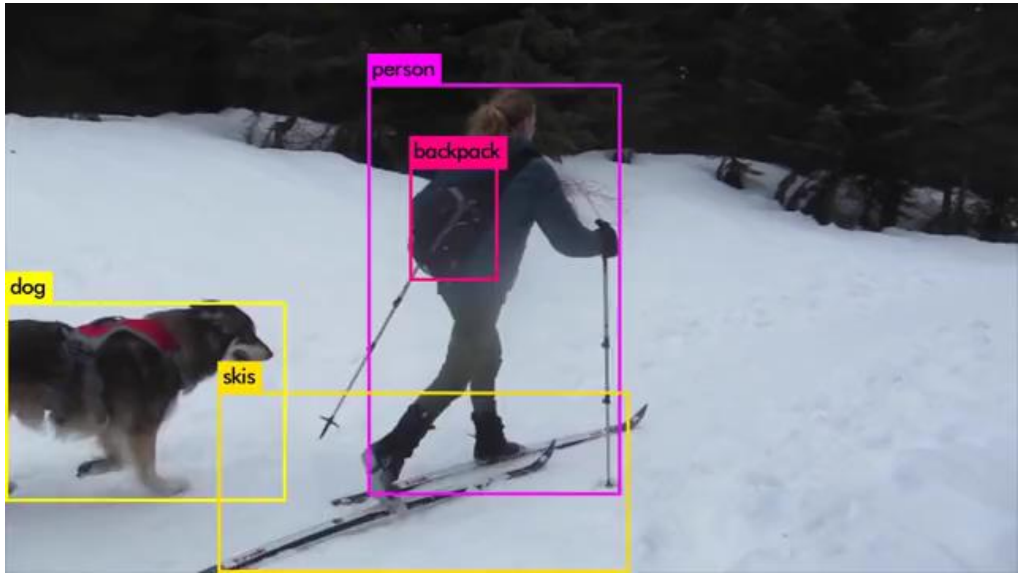
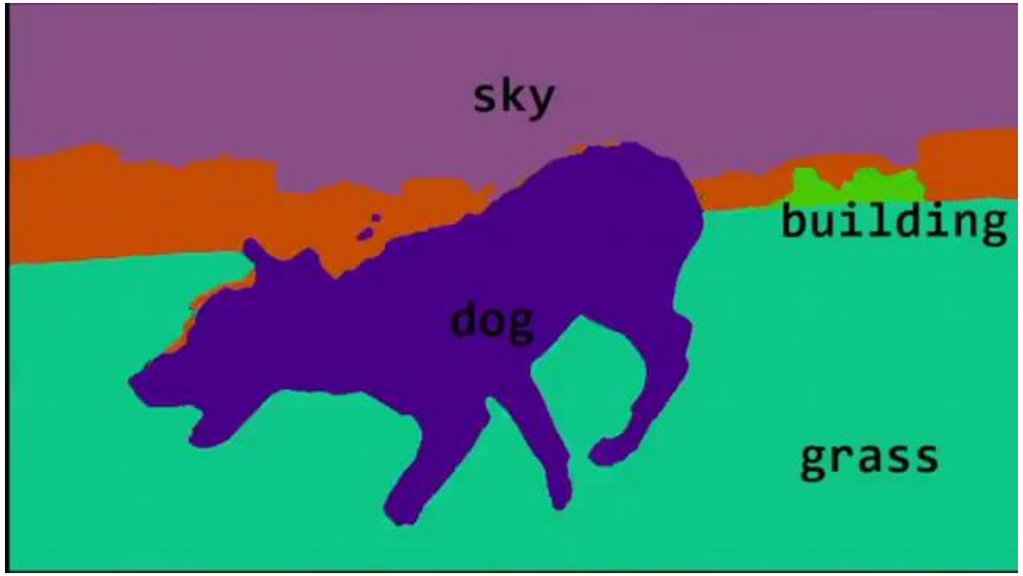


Image Analysis & Data Extraction

Once the image has been primed, we can analyze the contents and extract meaningful data from it





Core Tasks In Computer Vision

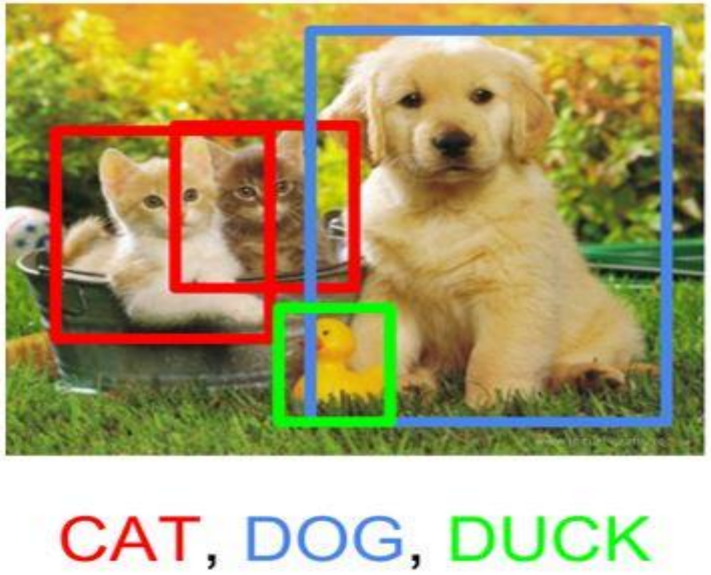
- **Object classification**
- **Object detection**
- **Image segmentation**
- **Scene reconstruction**
- **Image restoration**
- **Motion analysis**
- **Visual QnA**
- **Image generation**

CAT

Any other application that involves understanding pixels through software can safely be labeled as computer vision.

Core Tasks In Computer Vision

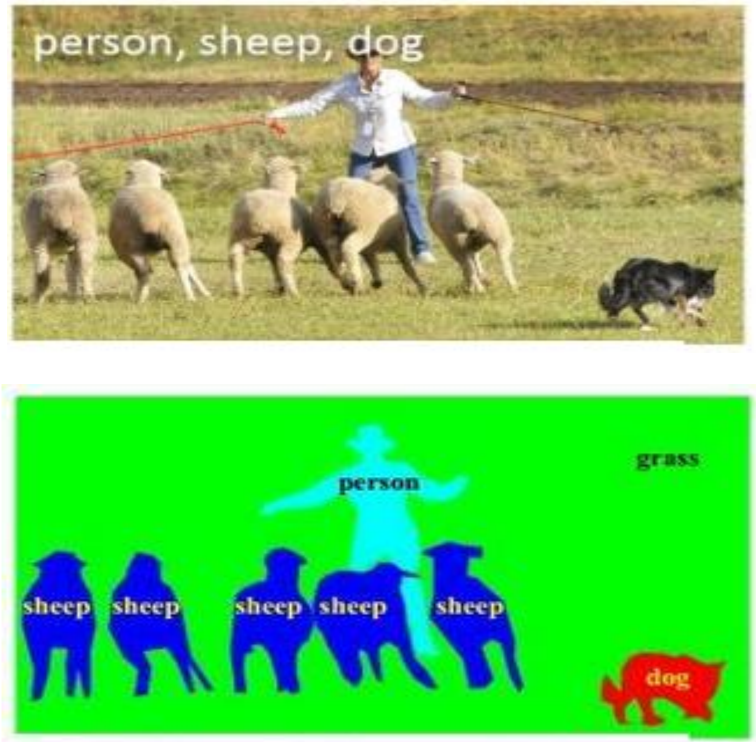
- **Object classification**
- **Object detection**
- **Image segmentation**
- **Scene reconstruction**
- **Image restoration**
- **Motion analysis**
- **Visual QnA**
- **Image generation**



Any other application that involves understanding pixels through software can safely be labeled as computer vision.

Core Tasks In Computer Vision

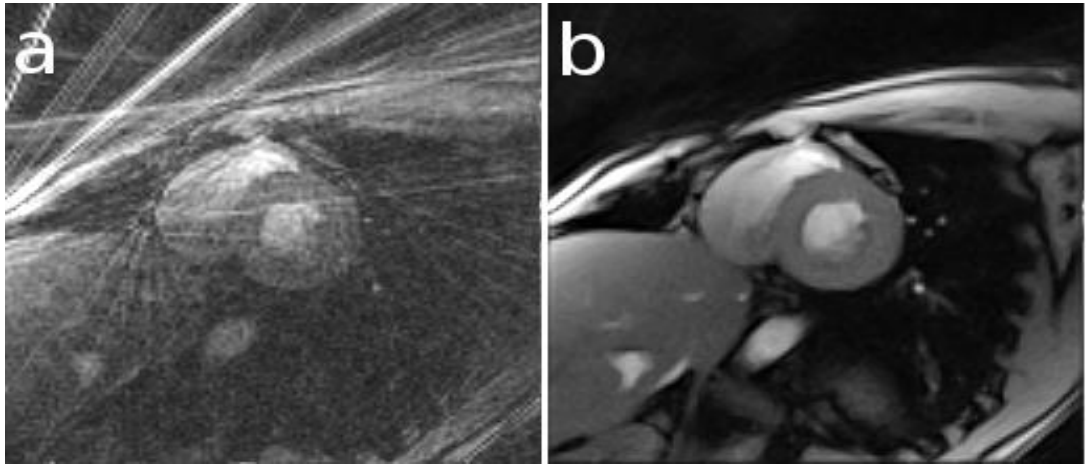
- Object classification
- Object detection
- Image segmentation
- Scene reconstruction
- Image restoration
- Motion analysis
- Visual QnA
- Image generation



Any other application that involves e
can safely be labeled as computer vision.

Core Tasks In Computer Vision

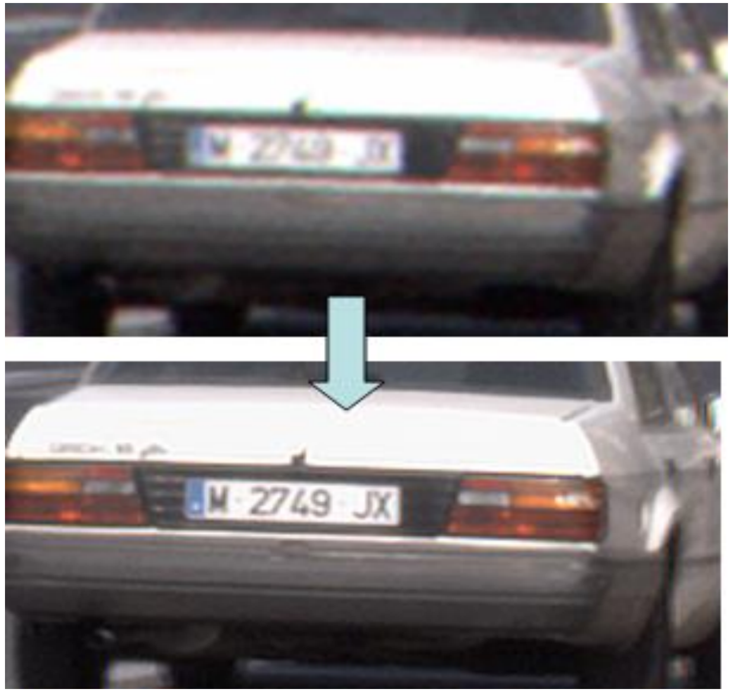
- **Object classification**
- **Object detection**
- **Image segmentation**
- **Scene reconstruction**
- **Image restoration**
- **Motion analysis**
- **Visual QnA**
- **Image generation**



Any other application that involve
can safely be labeled as computer vision.

Core Tasks In Computer Vision

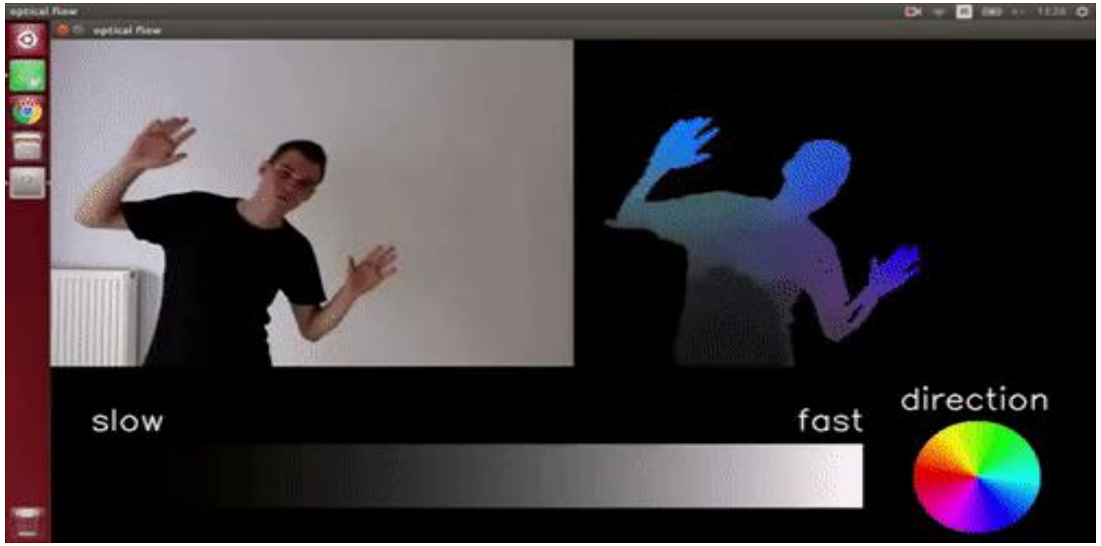
- Object classification
- Object detection
- Image segmentation
- Scene reconstruction
- Image restoration
- Motion analysis
- Visual QnA
- Image generation



Any other application that involv
can safely be labeled as computer

Core Tasks In Computer Vision

- **Object classification**
- **Object detection**
- **Image segmentation**
- **Scene reconstruction**
- **Image restoration**
- **Motion analysis**
- **Visual QnA**
- **Image generation**

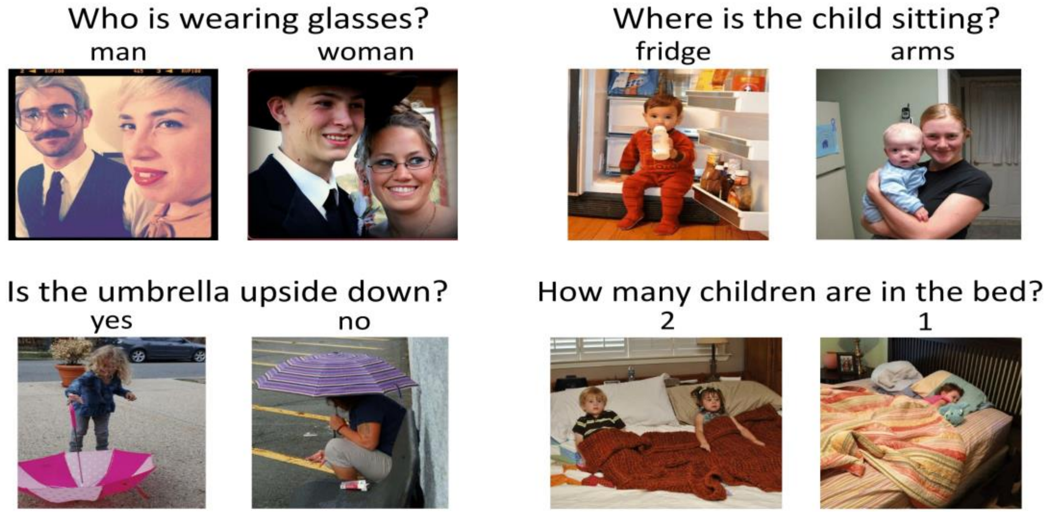


Any other application that involv
can safely be labeled as compute

Core Tasks In Computer Vision

- **Object classification**
- **Object detection**
- **Image segmentation**
- **Scene reconstruction**
- **Image restoration**
- **Motion analysis**
- **Visual QnA**
- **Image generation**

Any other application that invo
can safely be labeled as compute


Who is wearing glasses?
man
woman
Where is the child sitting?
fridge
arms
Is the umbrella upside down?
yes
no
How many children are in the bed?
2
1

Core Tasks In Computer Vision

- Object classification
- Object detection
- Image segmentation
- Scene reconstruction
- Image restoration
- Motion analysis
- Visual QnA
- Image generation

Any other application that invo
can safely be labeled as compute

Introduction to OpenCV

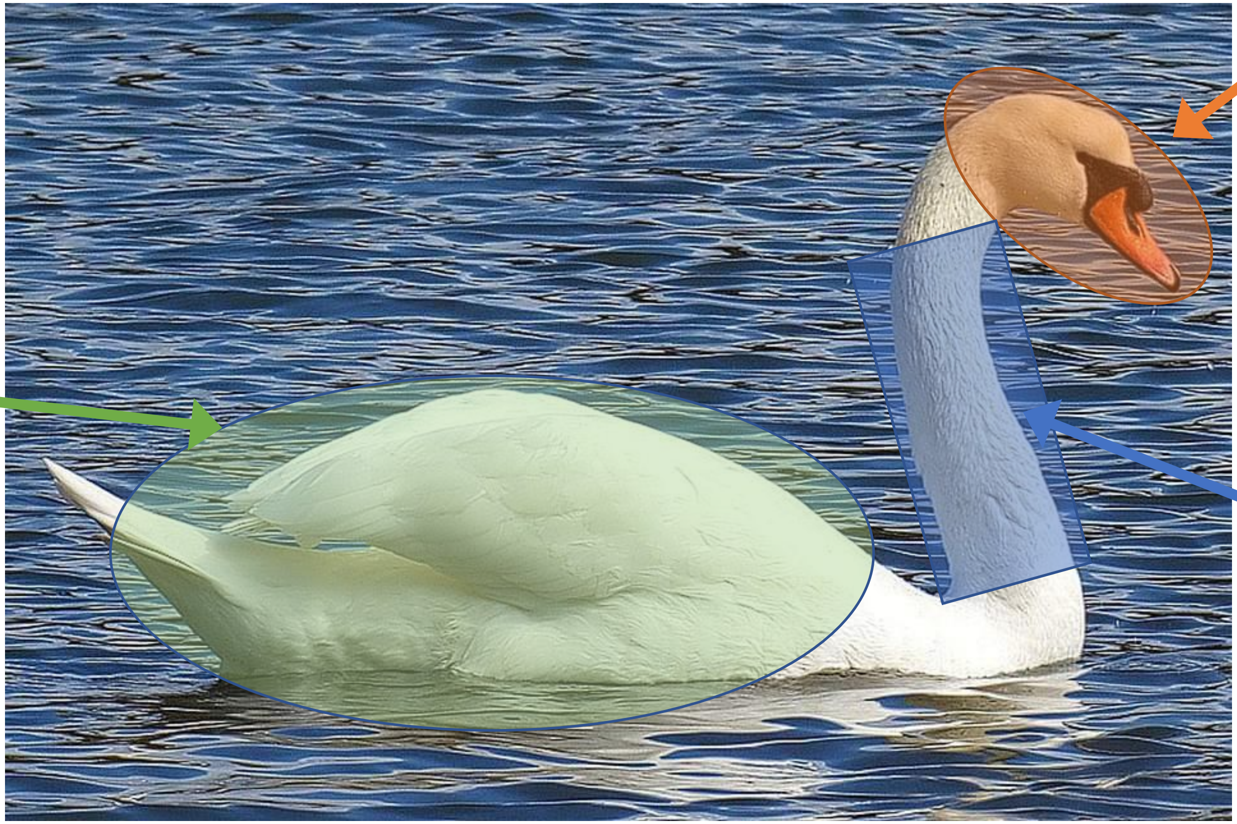
- OpenCV stands for ***Open Source Computer Vision*** library and it was invented by Intel in 1999.
- Has over **2500 modules** to help you with image-processing and other computer vision tasks
- Free and open-source.
- **Multiplatform** - Windows, Linux, MacOS, Android, IOS.
- **Multilingual** - Python, Java, JavaScript, etc.

Basics Of OpenCV

- **Getting started with images and OpenCV**: First step – import an image into OpenCV
- **Getting started with video**: Then we modify our code to work with video (streams of images)
- **Changing Color spaces**: We will explore different color spaces and learn how we can change between them using OpenCV
- **Working with Drawing functions**: We will use OpenCV to add text and draw on our frames
- **Geometric Transformations**: We learn how to apply perform basic geometric transformations to our images

Image Analysis

How does the human brain recognize an object in an image?

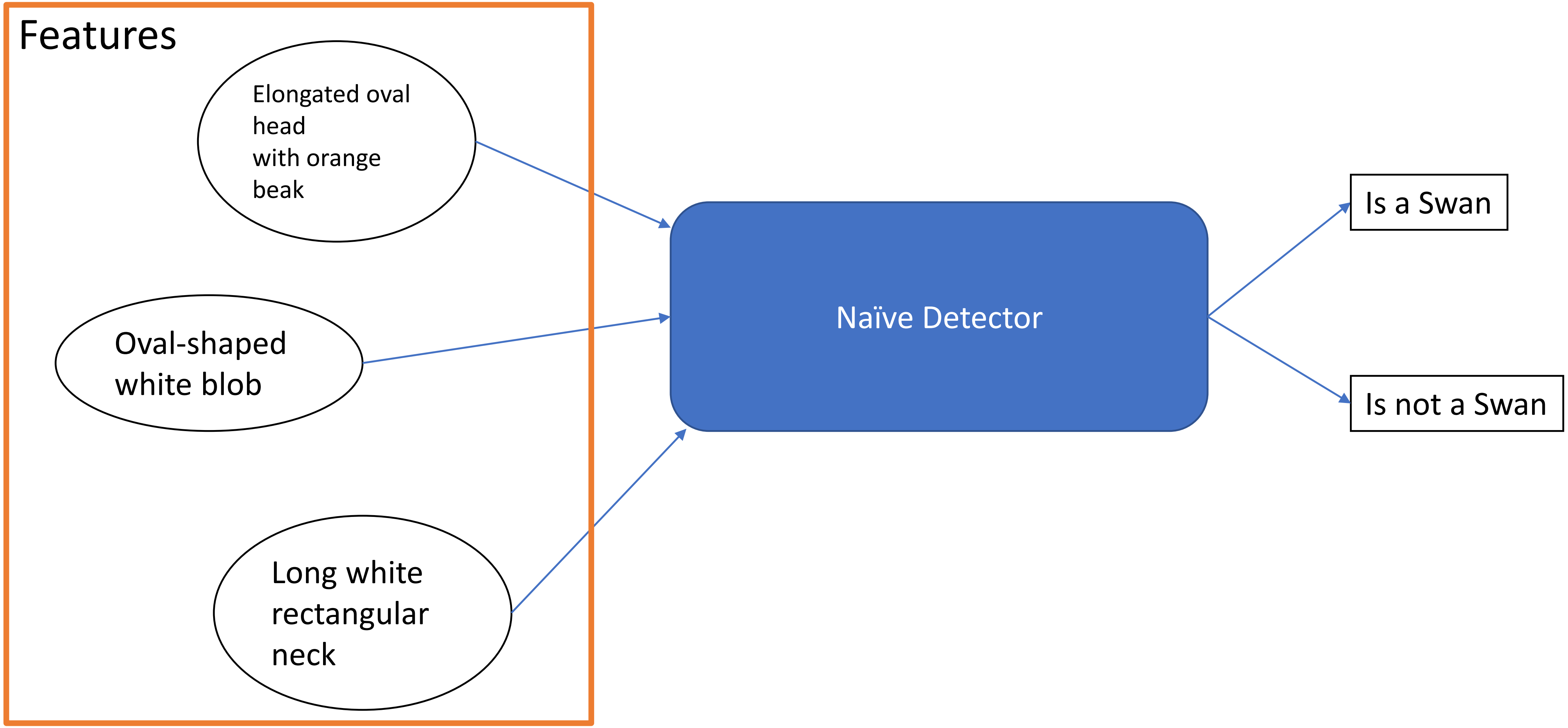


Elongated oval head with orange beak

Oval-shaped white blob

Long white rectangular neck

Image Analysis – Naïve Detector



Features

Elongated oval head with orange beak

Oval-shaped white blob

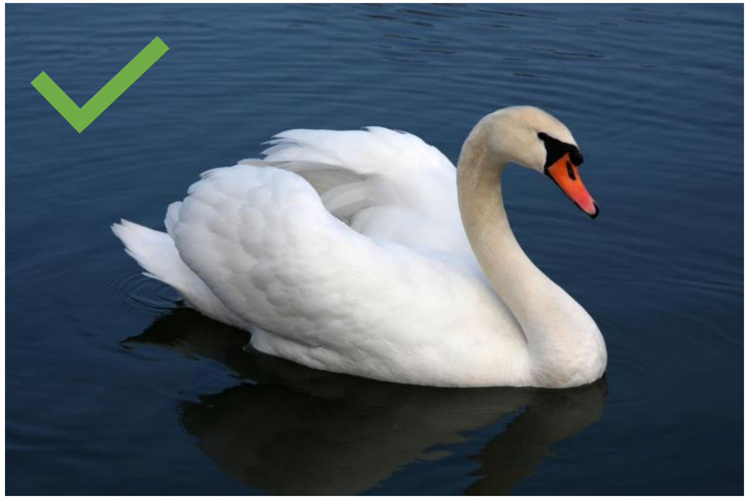
Long white rectangular neck

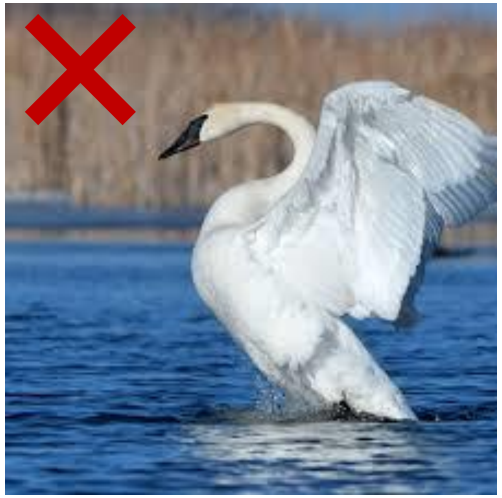
Naïve Detector

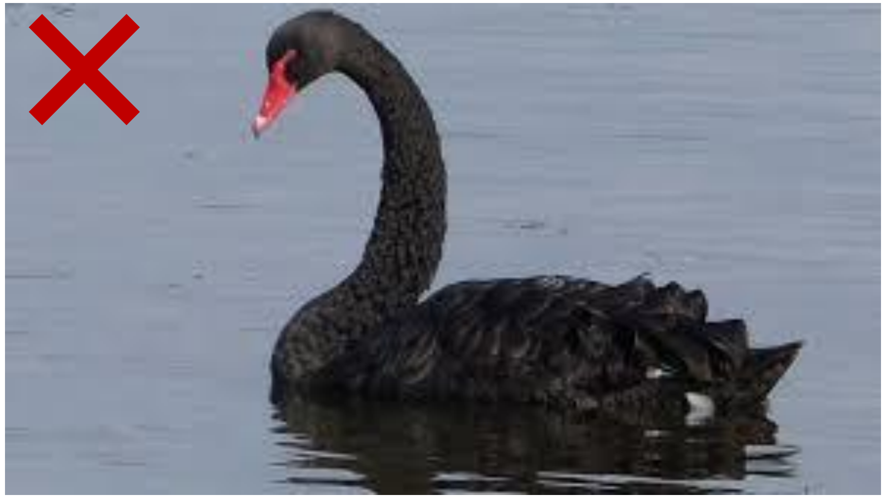
Is a Swan

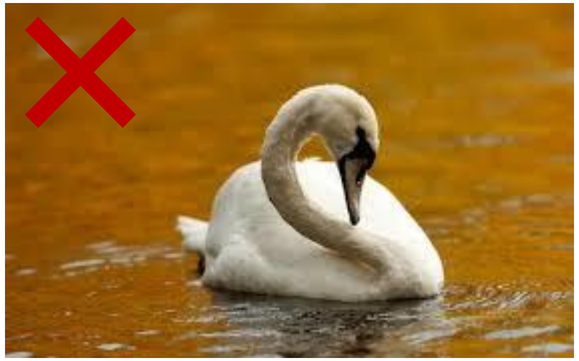
Is not a Swan

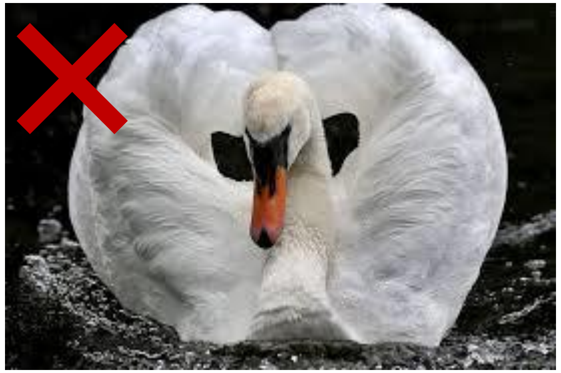
Let’s Test Our Classifier











Better Feature Definitions?

Researchers built multiple computer vision techniques to deal with issues

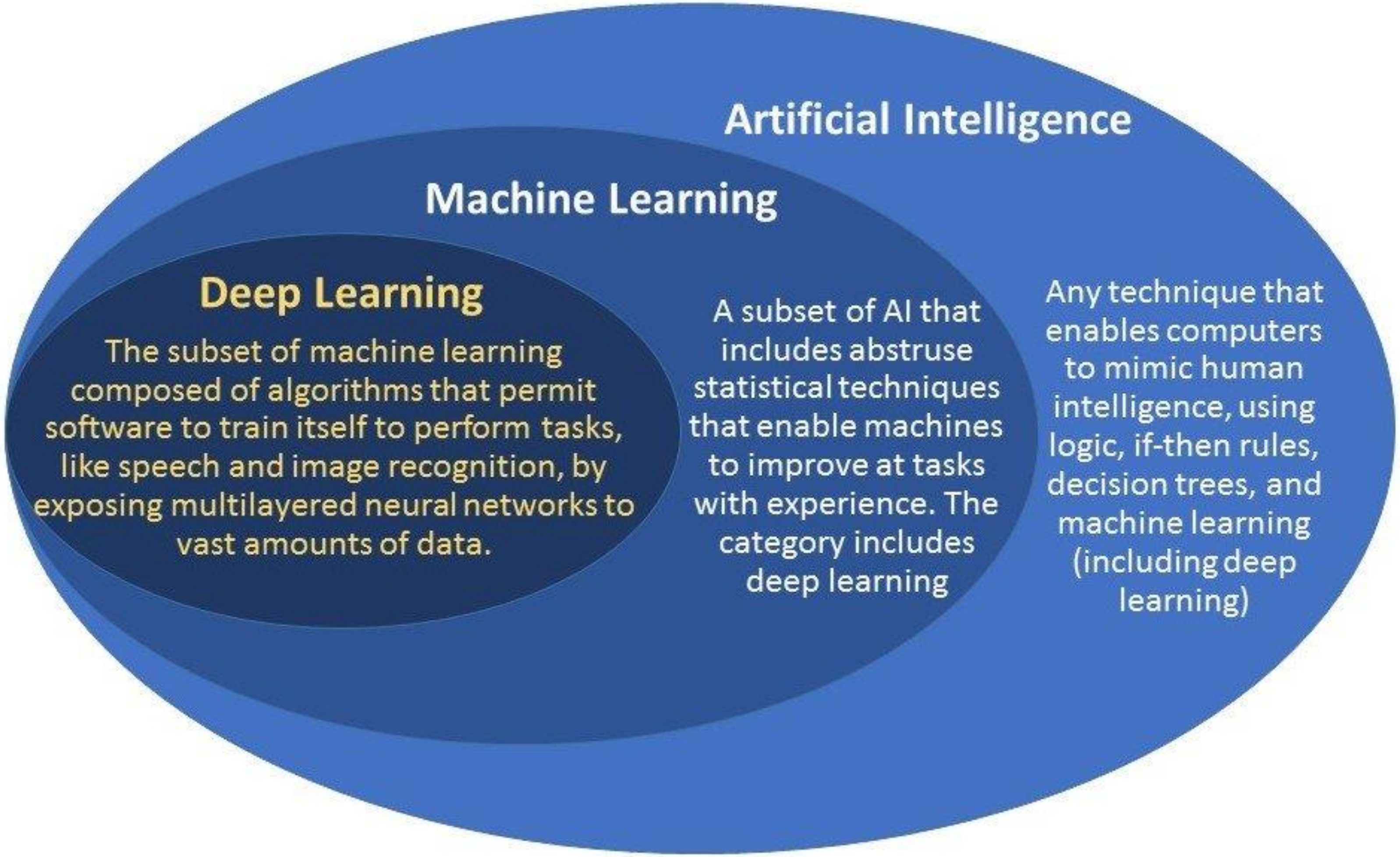
- SIFT
- FAST
- SURF
- BRIEF - etc

The problem?

- The detectors were too general OR
- Too over engineered to generalize

Can These Features Be Learned?

Introducing Deep Learning



Artificial Intelligence

Machine Learning

Deep Learning

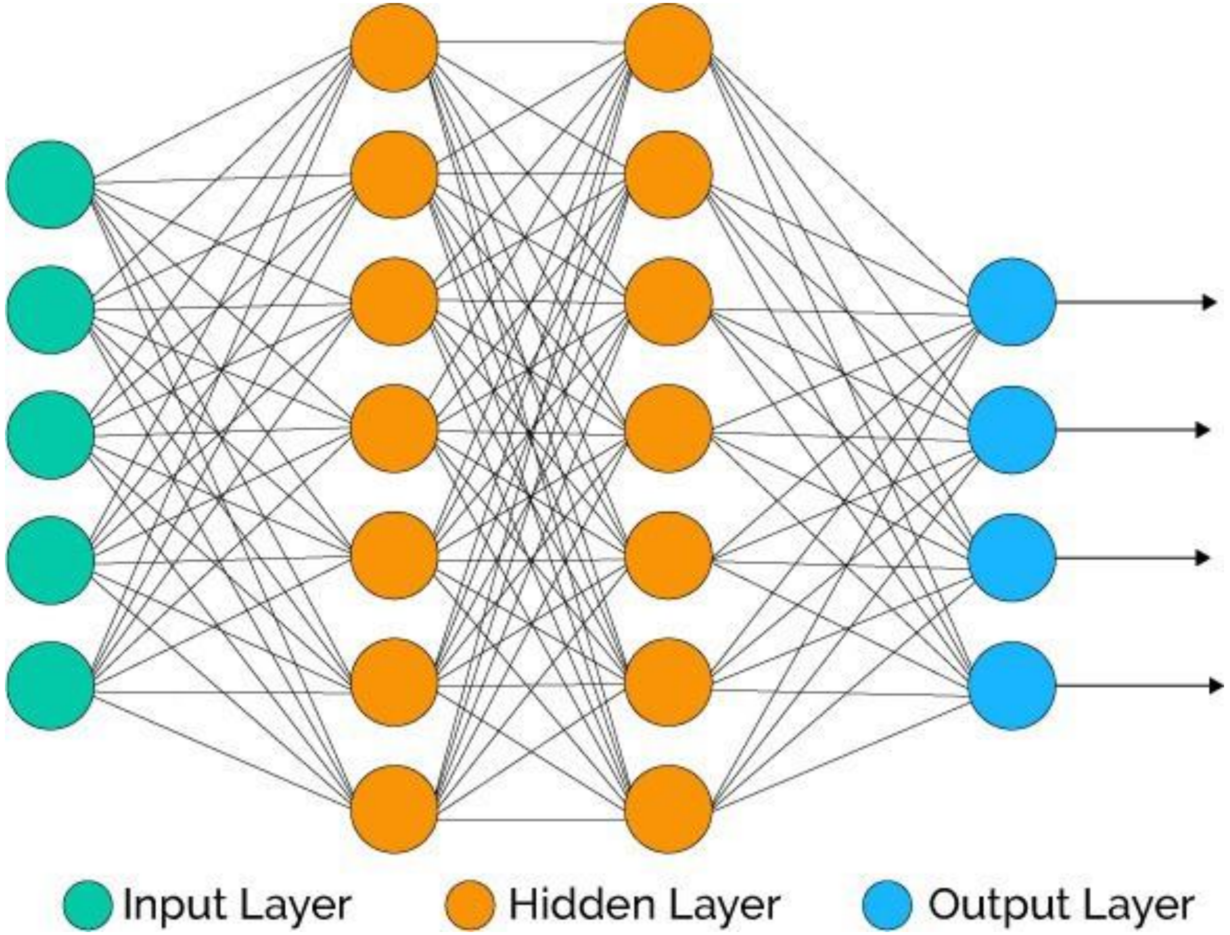
The subset of machine learning composed of algorithms that permit software to train itself to perform tasks, like speech and image recognition, by exposing multilayered neural networks to vast amounts of data.

A subset of AI that includes abstruse statistical techniques that enable machines to improve at tasks with experience. The category includes deep learning

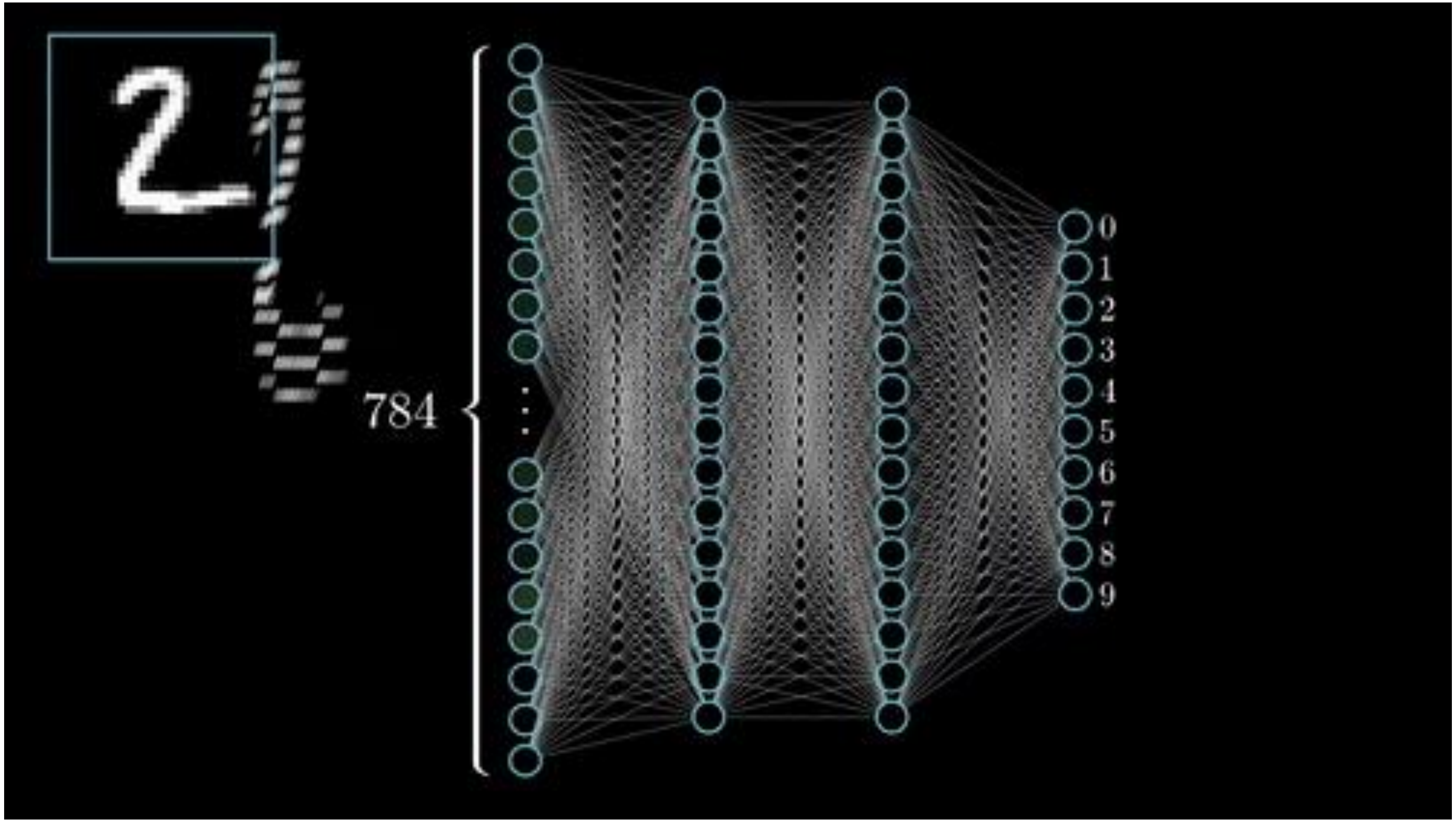
Any technique that enables computers to mimic human intelligence, using logic, if-then rules, decision trees, and machine learning (including deep learning)

What Are Neural Networks?

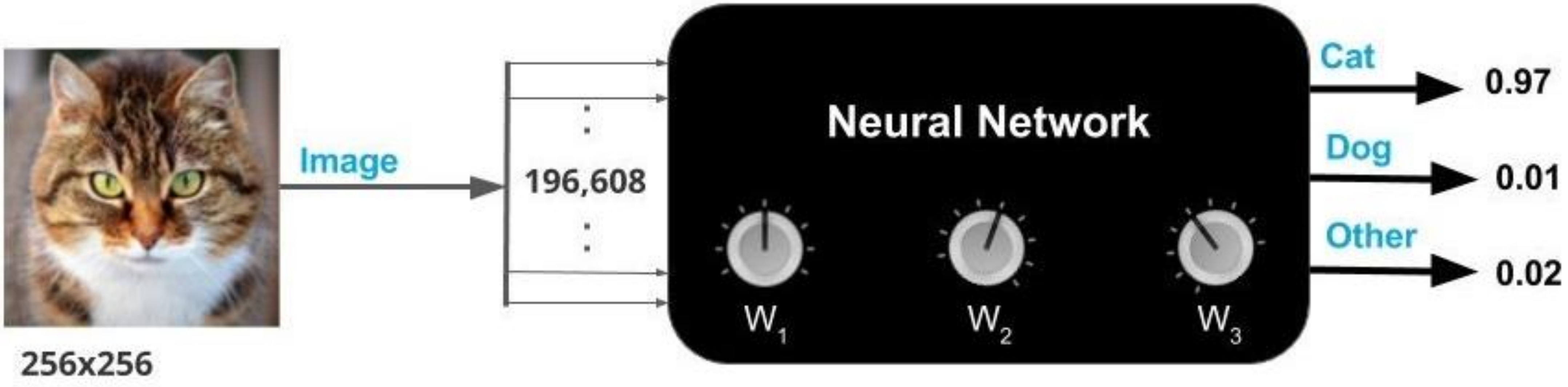
- Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns



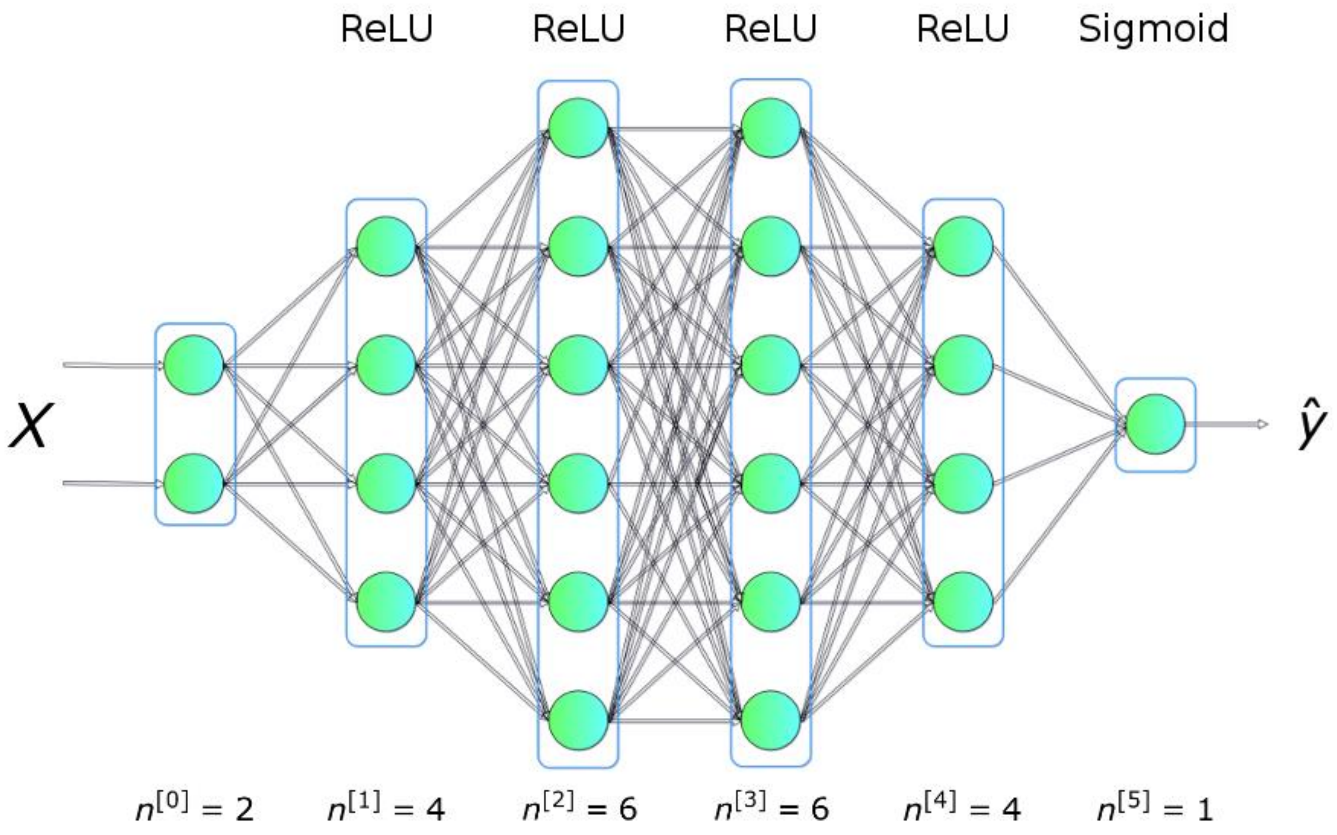
Introducing Artificial Neural Networks



The Knobs Analogy



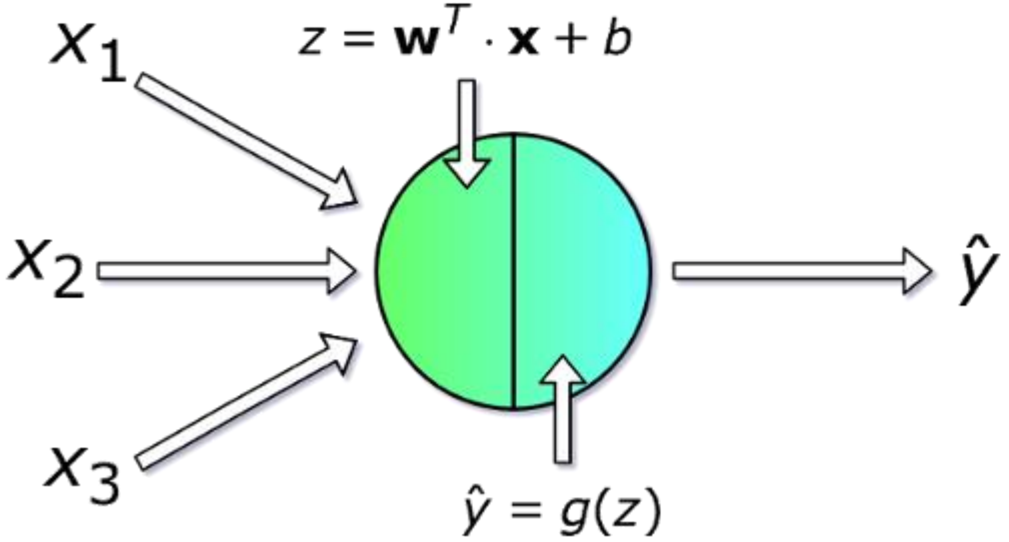
The Neural Network Architecture



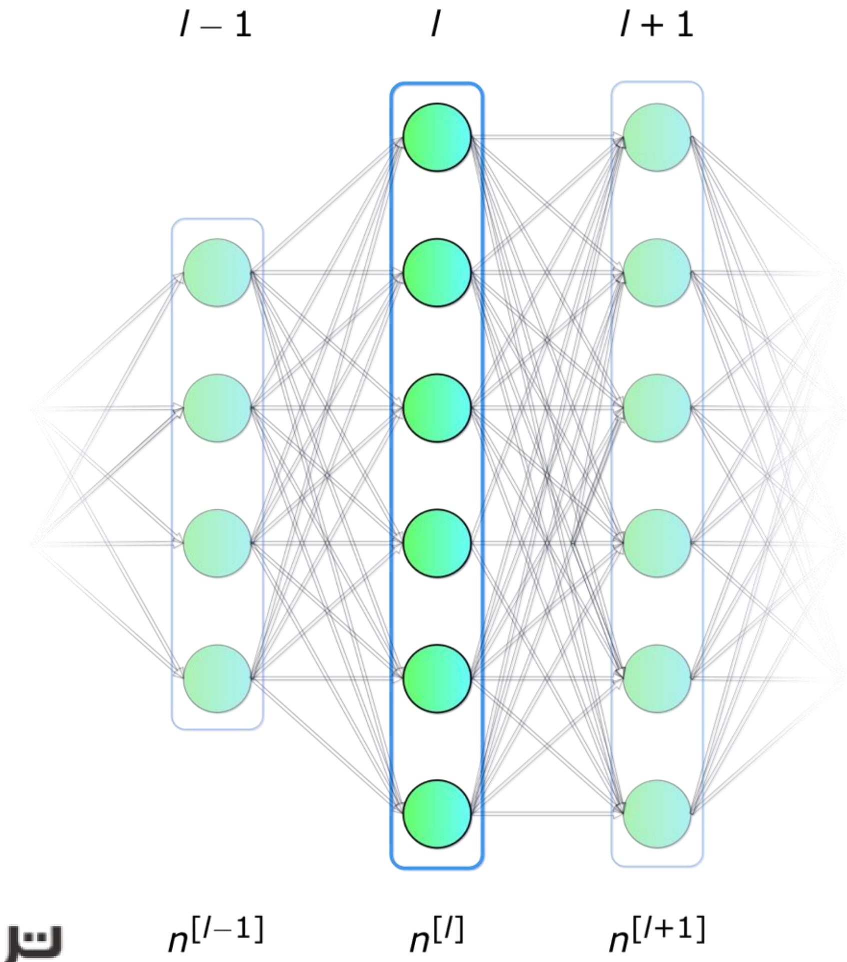
COMMUNITY
INNOVATION
WORKSPACE

The Artificial Neuron

$$z = w_1x_1 + w_2x_2 + w_3x_3 + \ldots + w_nx_n = \mathbf{w}^T \cdot \mathbf{x}$$



Single layer



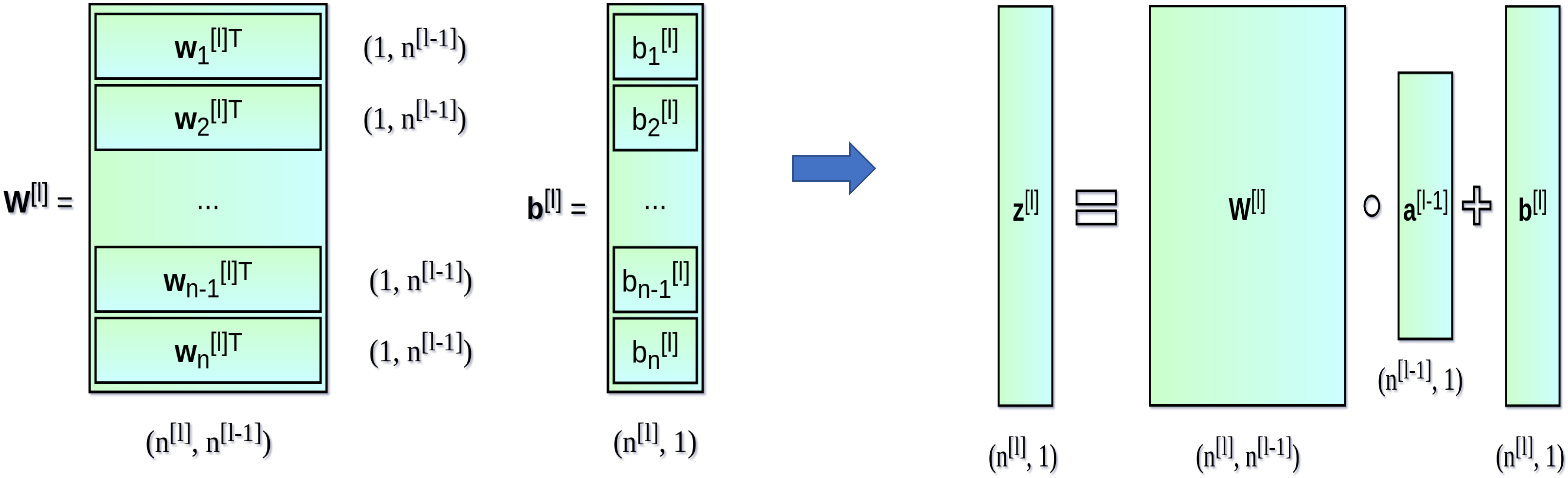
$$z_i^{[l]} = \mathbf{w}_i^T \cdot \mathbf{a}^{[\mathbf{l-1}]} + b_i \qquad a_i^{[l]} = g^{[l]}(z_i^{[l]})$$

$$\downarrow$$

$$\begin{aligned}
z_1^{[2]} &= \mathbf{w}_1^T \cdot \mathbf{a}^{[1]} + b_1 & a_1^{[2]} &= g^{[2]}(z_1^{[2]}) \\
z_2^{[2]} &= \mathbf{w}_2^T \cdot \mathbf{a}^{[1]} + b_2 & a_2^{[2]} &= g^{[2]}(z_2^{[2]}) \\
z_3^{[2]} &= \mathbf{w}_3^T \cdot \mathbf{a}^{[1]} + b_3 & a_3^{[2]} &= g^{[2]}(z_3^{[2]}) \\
z_4^{[2]} &= \mathbf{w}_4^T \cdot \mathbf{a}^{[1]} + b_4 & a_4^{[2]} &= g^{[2]}(z_4^{[2]}) \\
z_5^{[2]} &= \mathbf{w}_5^T \cdot \mathbf{a}^{[1]} + b_5 & a_5^{[2]} &= g^{[2]}(z_5^{[2]}) \\
z_6^{[2]} &= \mathbf{w}_6^T \cdot \mathbf{a}^{[1]} + b_6 & a_6^{[2]} &= g^{[2]}(z_6^{[2]})
\end{aligned}$$

Vectorizing the equations

$\mathbf{z}^{[l]} = \mathbf{W}^{[l]} \cdot \mathbf{a}^{[l-1]} + \mathbf{b}^{[l]}$ $\mathbf{a}^{[l]} = g^{[l]}(\mathbf{z}^{[l]})$



$\mathbf{W}^{[l]} =$ $w_1^{[l]T}$ $(1, n^{[l-1]})$, $w_2^{[l]T}$ $(1, n^{[l-1]})$, …, $w_{n-1}^{[l]T}$ $(1, n^{[l-1]})$, $w_n^{[l]T}$ $(1, n^{[l-1]})$

$(n^{[l]}, n^{[l-1]})$

$\mathbf{b}^{[l]} =$ $b_1^{[l]}$, $b_2^{[l]}$, …, $b_{n-1}^{[l]}$, $b_n^{[l]}$

$(n^{[l]}, 1)$

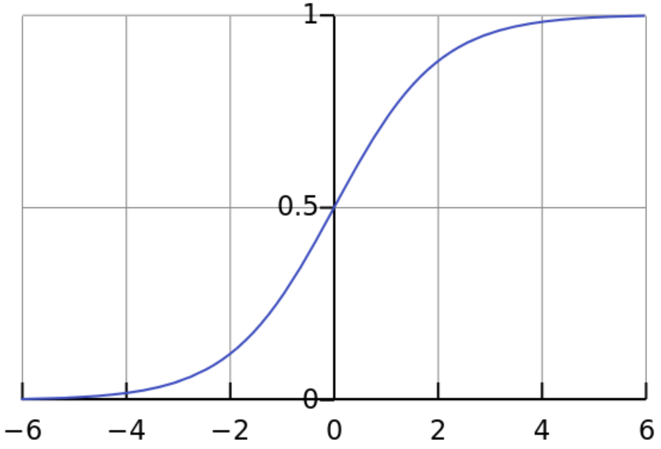
$$\mathbf{z}^{[l]} = \mathbf{W}^{[l]} \circ \mathbf{a}^{[l-1]} + \mathbf{b}^{[l]}$$

$(n^{[l]}, 1)$ $(n^{[l]}, n^{[l-1]})$ $(n^{[l-1]}, 1)$ $(n^{[l]}, 1)$

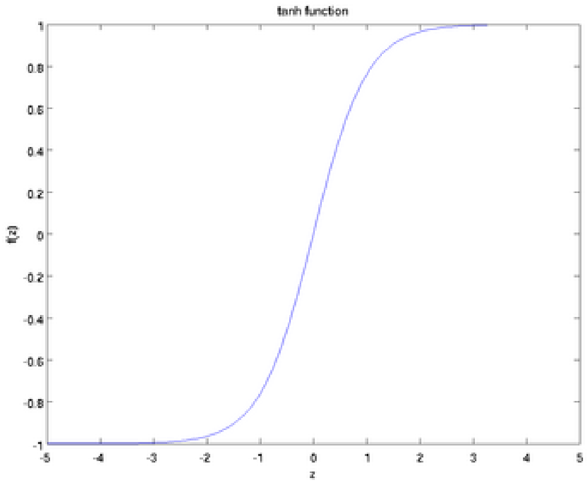
Activation Function And Why Do We Need It?

- **Without them, our neural network would become a combination of linear functions, so it would be just a linear function itself**
- The activation function also has a significant impact on the speed of learning, which is one of the main criteria for their selection

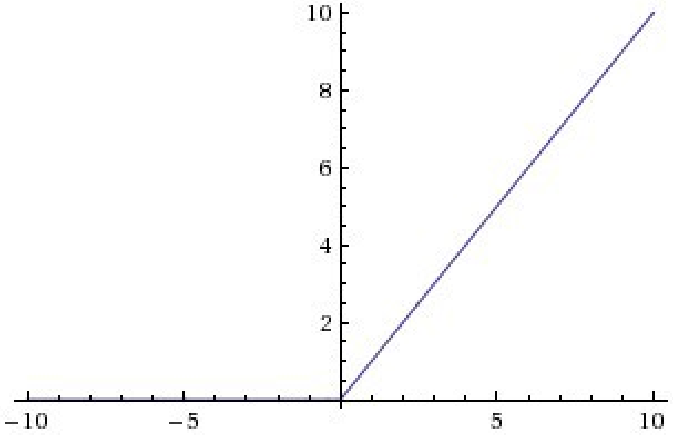
Activation Function And Why Do We Need It?



$$A = \frac{1}{1+e^{-x}}$$

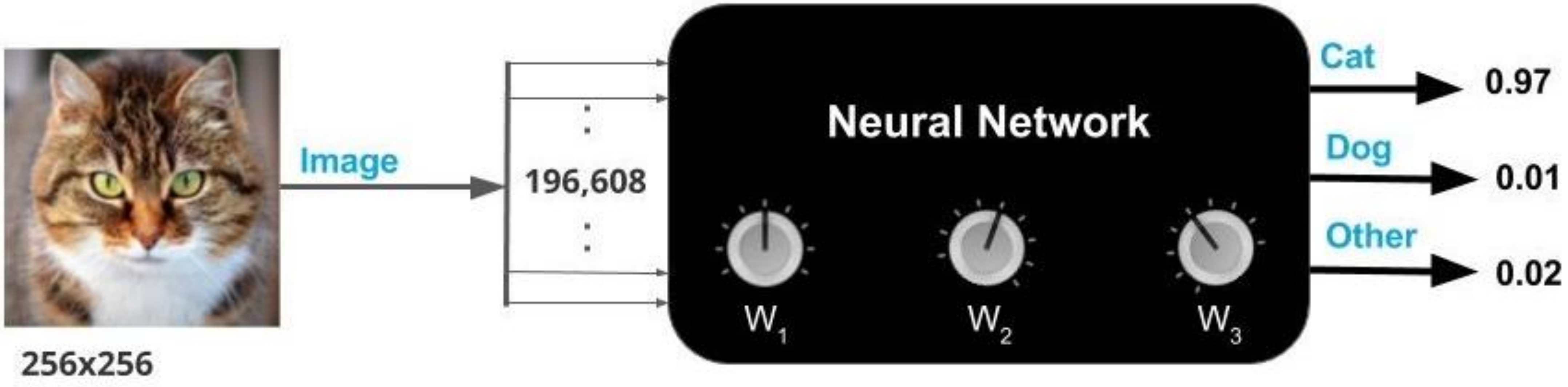


$$f(x) = tanh(x) = \frac{2}{1+e^{-2x}} - 1$$



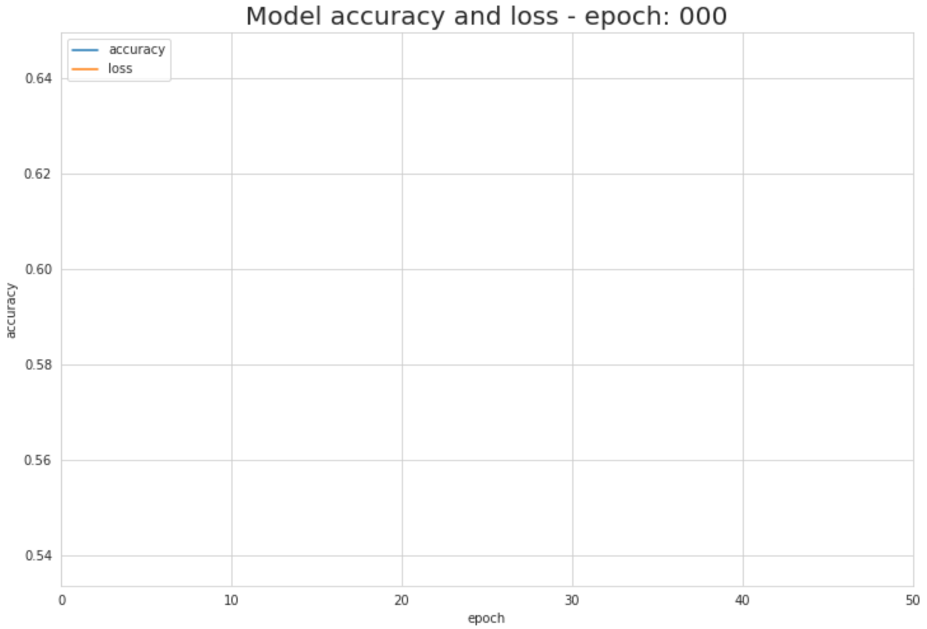
$$A(x) = max(0,x)$$

The Knobs Analogy



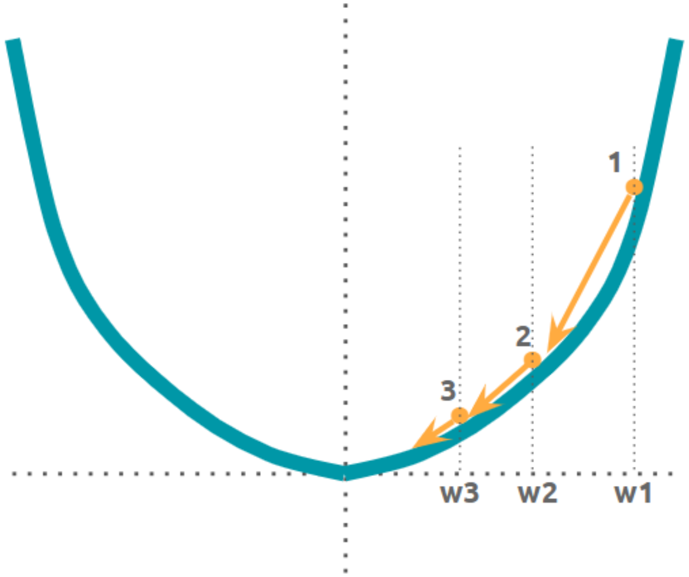
The Loss Function

- The basic source of information on the progress of the learning process is the value of the loss function.



How do neural networks learn?

- The learning process is about changing the values of the **W** and **b** parameters so that the loss function is minimized. In order to achieve this goal, we will turn for help to calculus and **use gradient descent method to find a function minimum**.



$$W = W - \alpha \times dW$$

W2 = W1 - α*dW1
W3 = W2 - α*dW2
.
.

Backpropagation

- Backpropagation is an algorithm that allows us to calculate a very complicated gradient, like the one we need.
- The parameters of the neural network are adjusted according to the following formulae.

$$\mathbf{W}^{[l]} = \mathbf{W}^{[l]} - \alpha \mathbf{dW}^{[l]}$$
$$\mathbf{b}^{[l]} = \mathbf{b}^{[l]} - \alpha \mathbf{db}^{[l]}$$

- In the equations above, α represents learning rate - a hyperparameter which allows you to control the value of performed adjustment

The Training Process

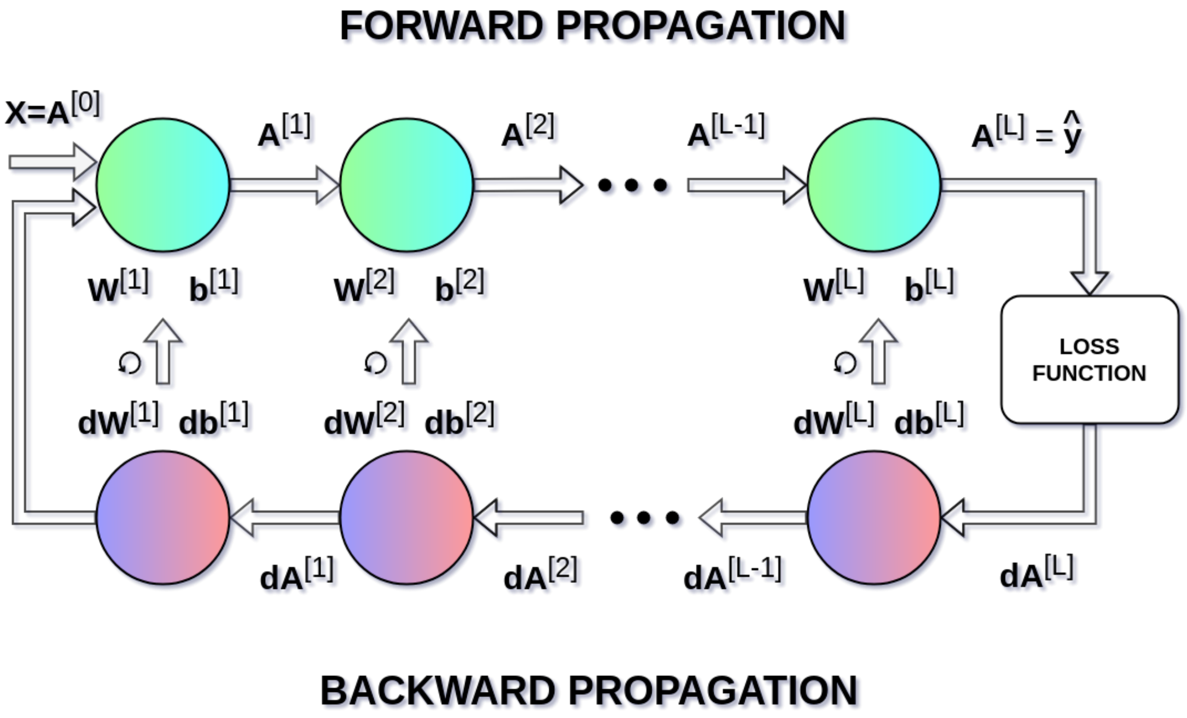
- **dW** and **db** are calculated using the chain rule, partial derivatives of loss function with respect to **W** and **b**.

$$\mathbf{dW}^{[l]} = \frac{\partial L}{\partial \mathbf{W}^{[l]}} = \frac{1}{m}\mathbf{dZ}^{[l]}\mathbf{A}^{[l-1]T}$$

$$\mathbf{db}^{[l]} = \frac{\partial L}{\partial \mathbf{b}^{[l]}} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{dZ}^{[l](i)}$$

$$\mathbf{dA}^{[l-1]} = \frac{\partial L}{\partial \mathbf{A}^{[l-1]}} = \mathbf{W}^{[l]T}\mathbf{dZ}^{[l]}$$

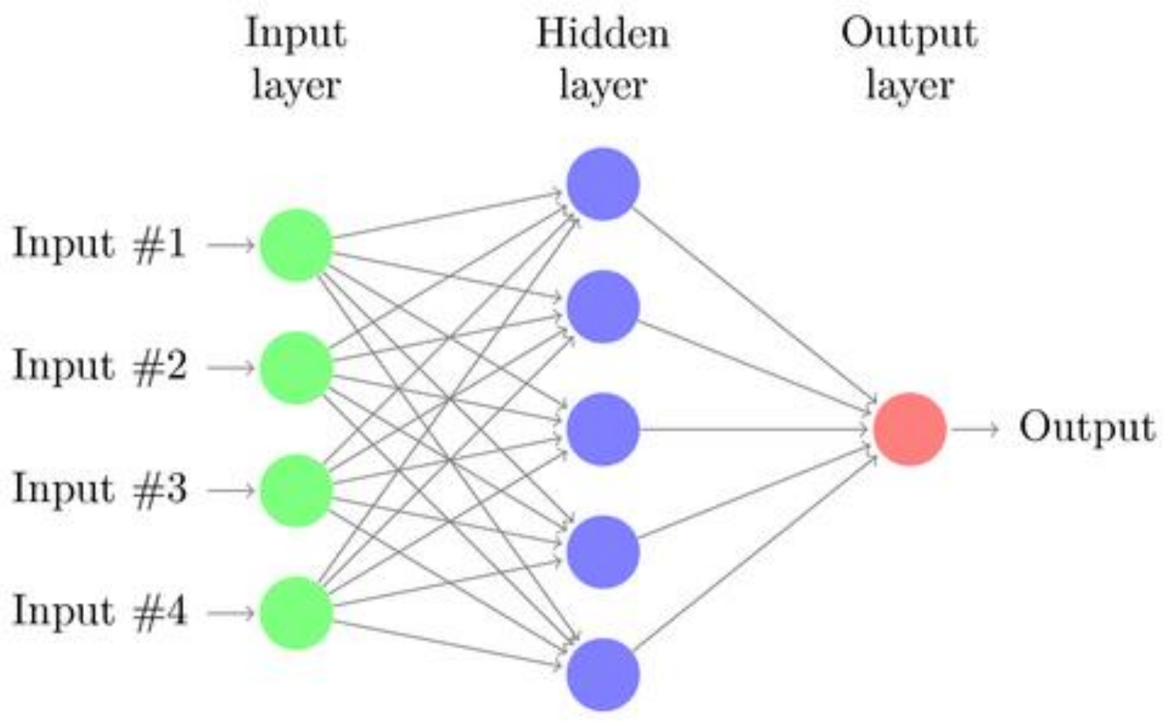
$$\mathbf{dZ}^{[l]} = \mathbf{dA}^{[l]} * g'(\mathbf{Z}^{[l]})$$



Let’s Build Our First Neural Network

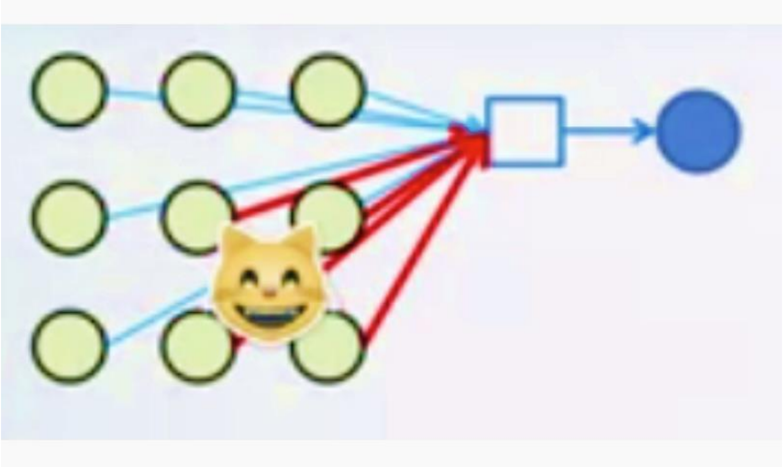
The Problem With Traditional Neural Networks

- MLPs use one perceptron for each input (e.g. pixel in an image, multiplied by 3 in RGB case)
- The amount of weights rapidly becomes unmanageable for large images
- For a 224 x 224 pixel image with 3 color channels there are around 150,000 weights that must be trained!

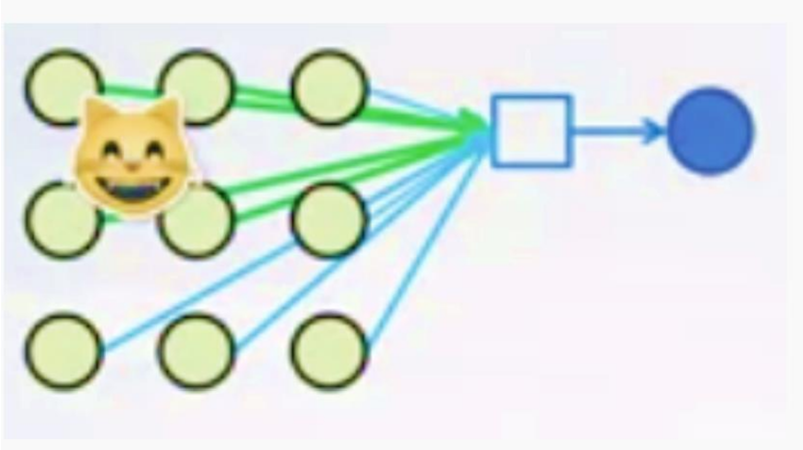


The Problem With Traditional Neural Networks

- MLPs react differently to an input (images) and its shifted version — **they are NOT translation invariant**.
- MLPs do not consider spatial information as each pixel is considered as an individual input

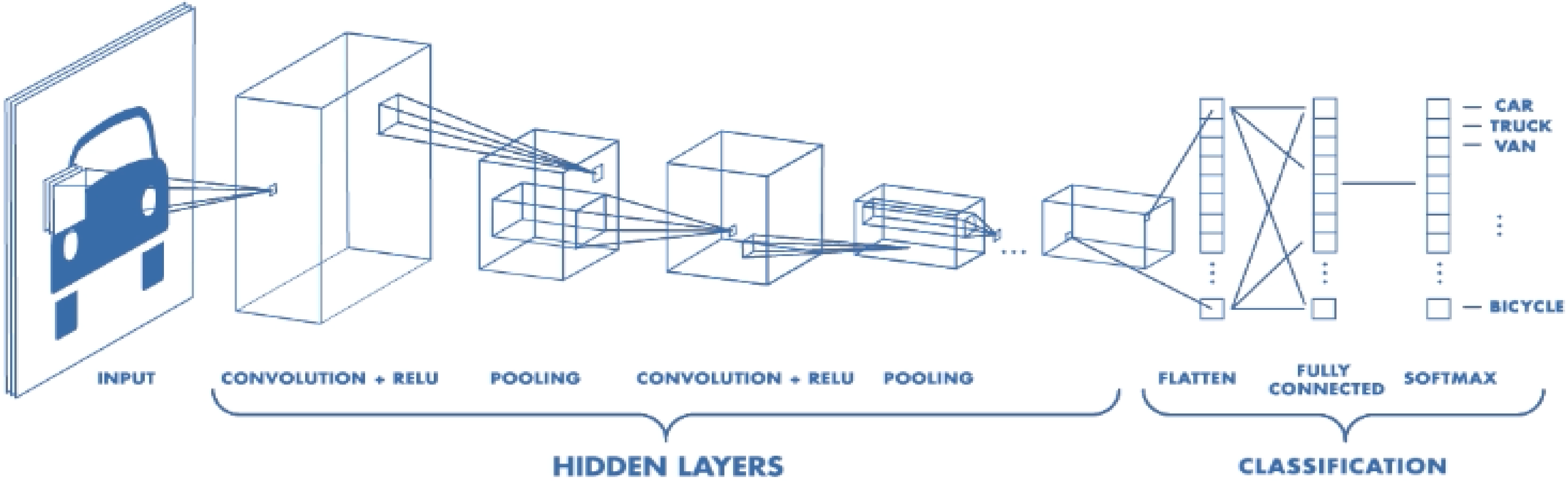


In this case red weights will get modified to better recognize cats



In this case green weights will get modified to better recognize cats

Enter the Convolutional Neural Network

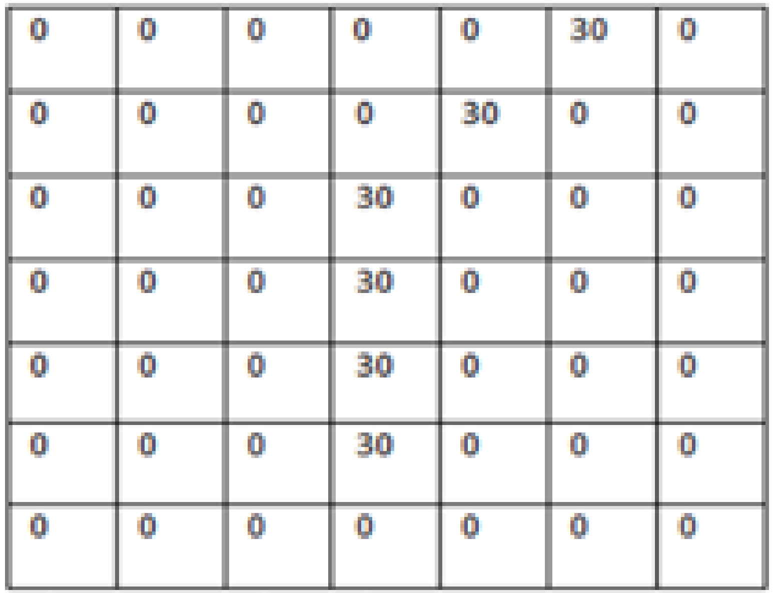


Core Idea Of Convolutional Neural Networks

- Our brain captures the patterns in figures to classify an object
- What if our machine can be trained to classify images by detecting some patterns like us?
- CNNs leverage the fact that nearby pixels are more strongly related than distant ones and analyze the influence of nearby pixels by using something called a **filter**
- In simpler terms, CNNs use filters to find patterns in an image just like our brain does

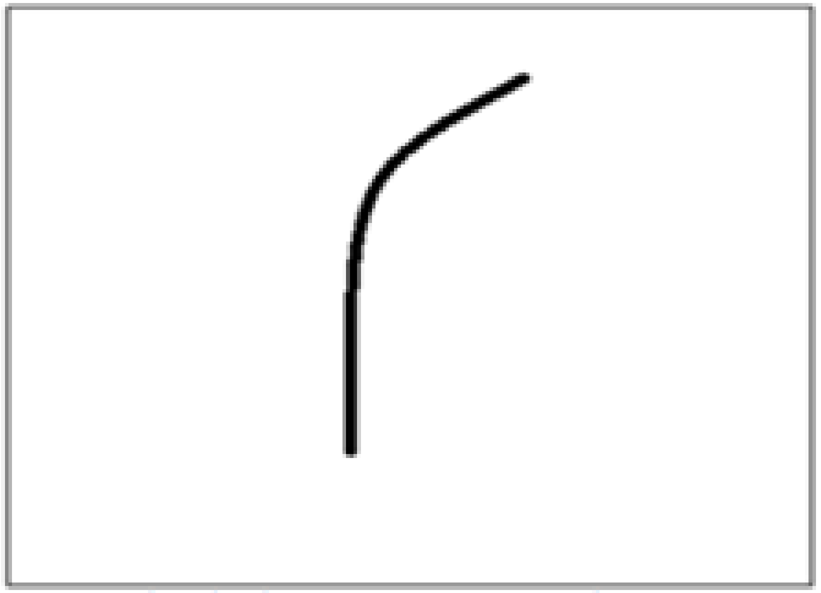
What Are Filters?

- A filter is nothing more than a small 2-dimensional array of values which represents a pattern that we are looking for`.



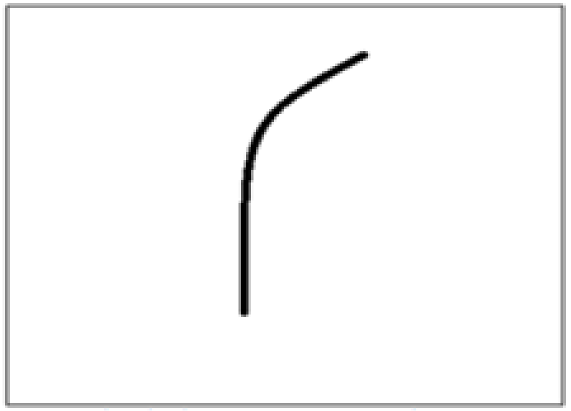
0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

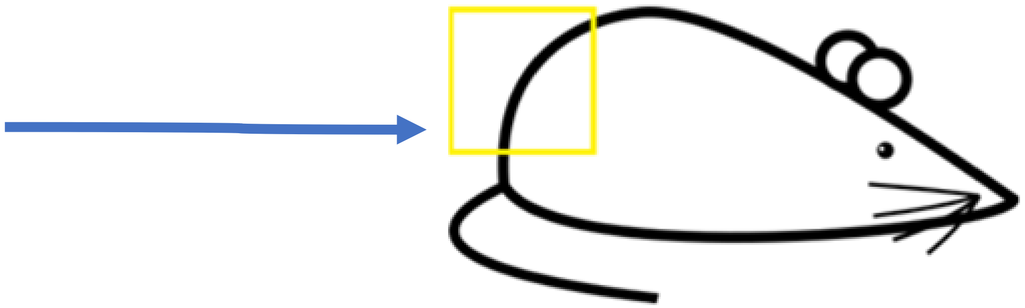
Pixel representation of a filter



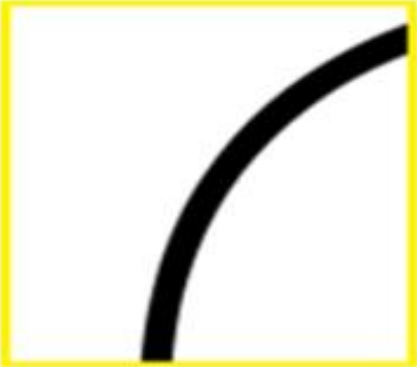
Visualization of a filter

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0





The sum of the multiplication value that is generated is = 4(50*30)+(20*30) = 6600 (large number)*



Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

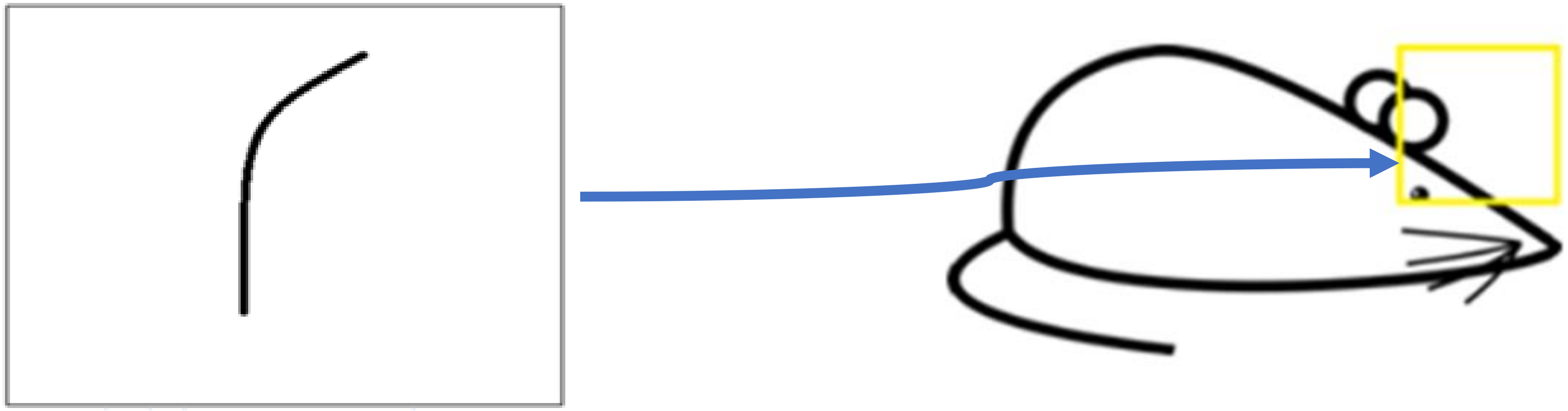
*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

➡ Large Value

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0



The sum of the multiplication value that is generated is = 0 (small number)

Visualization of the filter on the image

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

Pixel representation of receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

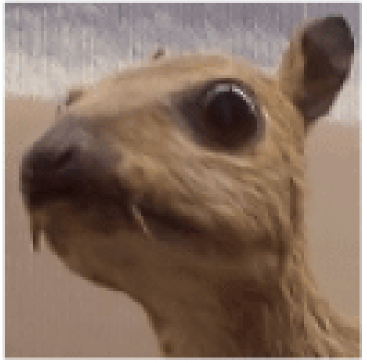
Pixel representation of filter

➡ Small Value

What is Convolution?

- By definition, **convolution** is a mathematical operation on two objects to produce an outcome that expresses how the shape of one is modified by the other.
- With this computation, we detect a particular feature from the input image and get the result having information about that feature
- This is called a '***feature map***.' Outcome with a real world example below.

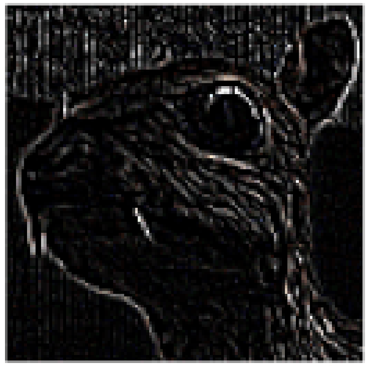
Input image



Convolution Kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

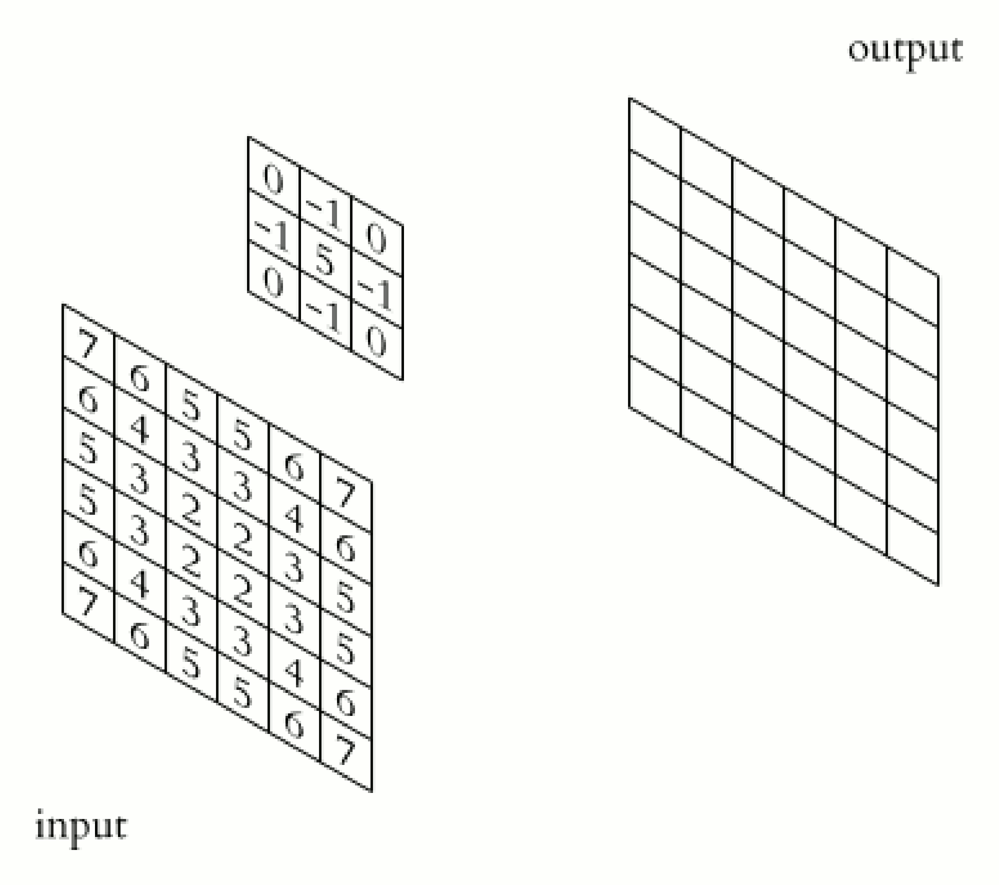
Feature map



How Does Convolution Work?

The output image, C, is computed from the input image, A, and the convolution kernel, H, according to the following expression.

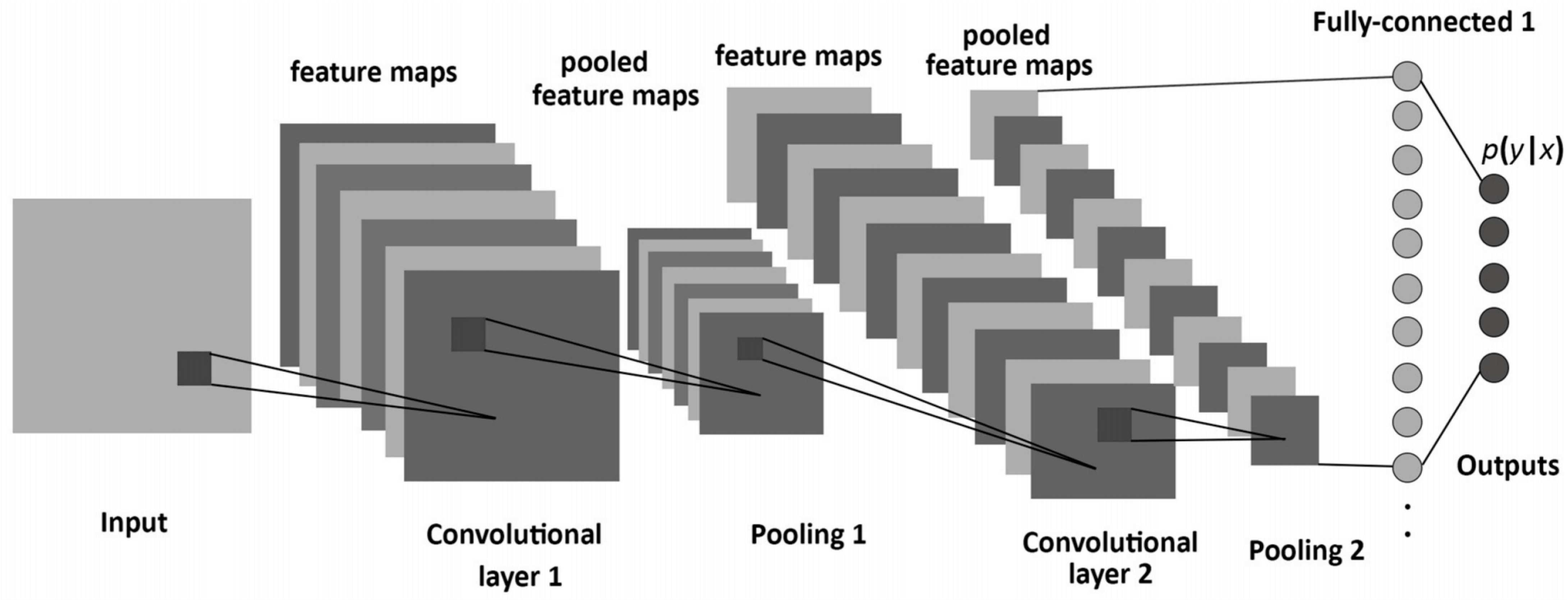
$$C[m][n] = \sum_{u} \sum_{v} (A[m+u][n+v] \cdot H[u][v])$$



0	-1	0
-1	5	-1
0	-1	0

7	6	5	5	6	7
6	4	3	3	4	6
5	3	2	2	3	5
5	3	2	2	3	5
6	4	3	3	4	6
7	6	5	5	6	7

The Vanilla CNN



+ ReLU

+ ReLU

Layers Of A CNN

- We use three main types of layers to build CNN architectures: **Convolutional Layer**, **Pooling Layer**, and **Fully-Connected Layer**
- We can stack these layers to form a full CNN **architecture**
- **INPUT** will hold the raw pixel values of the image
- **CONV** layer will compute the output of neurons that are connected to local regions in the input
- **RELU** layer will apply an elementwise activation function, such as the max(0,x) thresholding at zero
- **POOL** layer will perform a downsampling operation along the spatial dimensions (width, height)
- **FC** (i.e. fully-connected) layer will compute the class scores

Convolution Layer

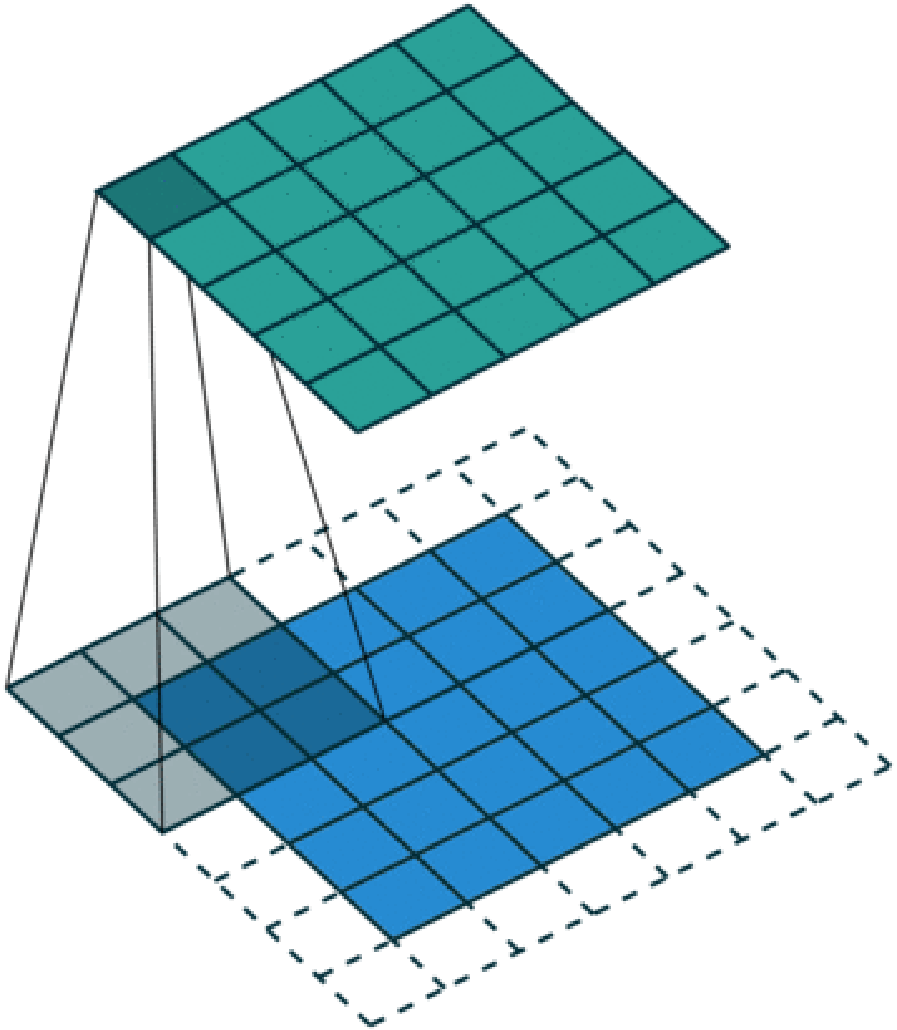
- Accepts a volume of size W1×H1×D1
- Requires four hyperparameters:
  - Number of filters K
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size W2×H2×D2

  where:
  - W2=(W1−F+2P)/S+1
  - H2=(H1−F+2P)/S+1

  (i.e. width and height are computed equally by symmetry)
  - D2=K

Padding And Convolution

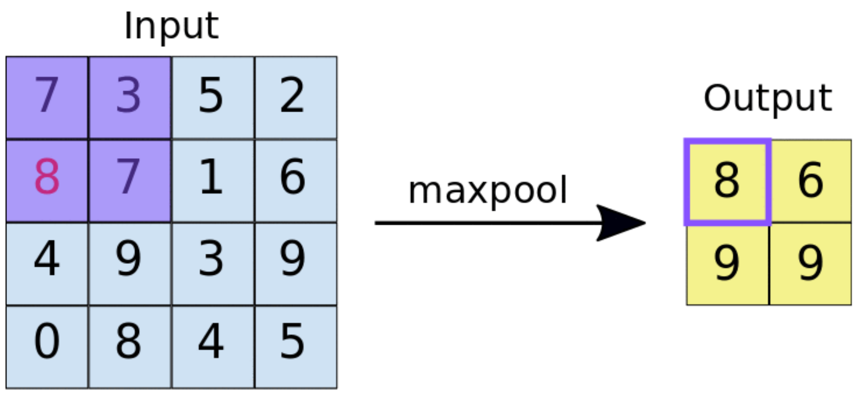
- If we apply convolutions on a normal image, the result will be down-sampled by an amount depending on the size of the filter
- Padding essentially makes the feature maps produced by the filter kernels the same size as the original image



Pooling Layer

- Accepts a volume of size W1×H1×D1
- Requires two hyperparameters:
  - their spatial extent F,
  - the stride S,
- Produces a volume of size W2×H2×D2

  where:
  - W2=(W1−F)/S+1
  - H2=(H1−F)/S+1
  - D2=D1
- Introduces zero parameters since it computes a fixed function of the input



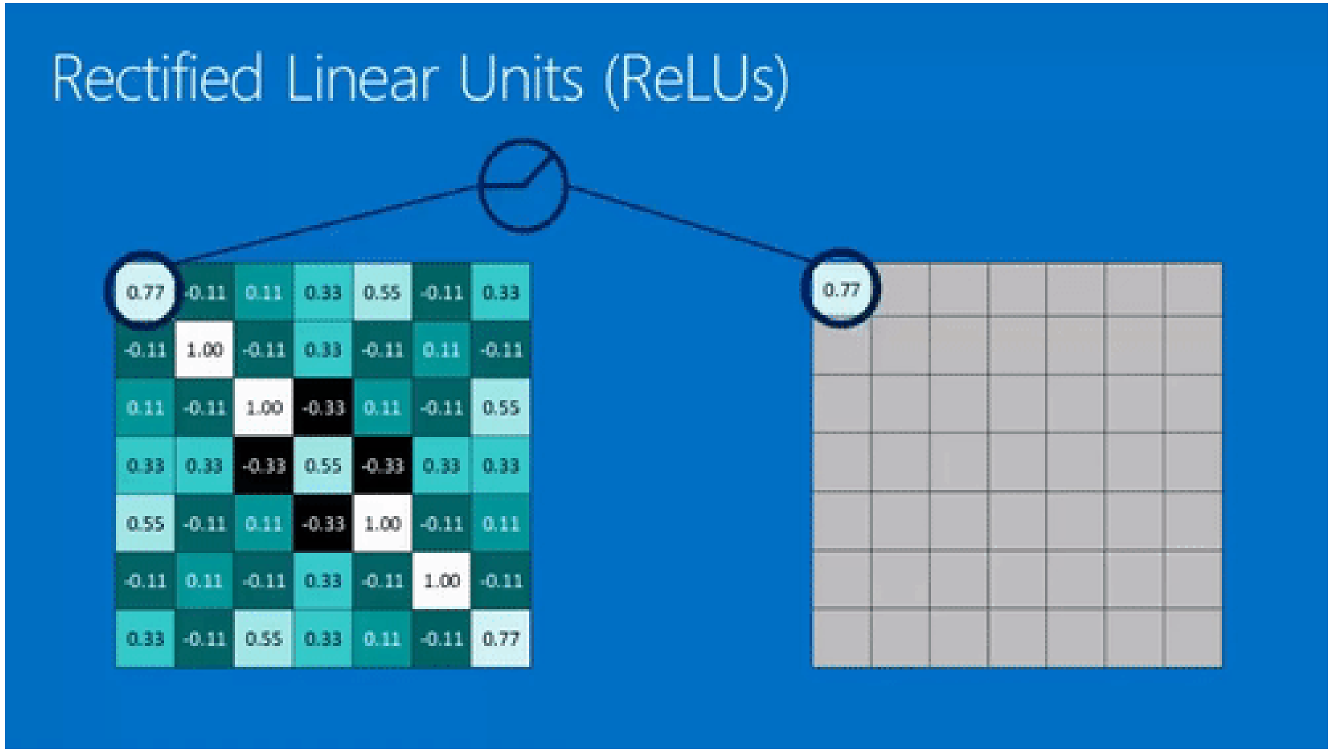
Input

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

Output

8	6
9	9

Activation Layer And Relu

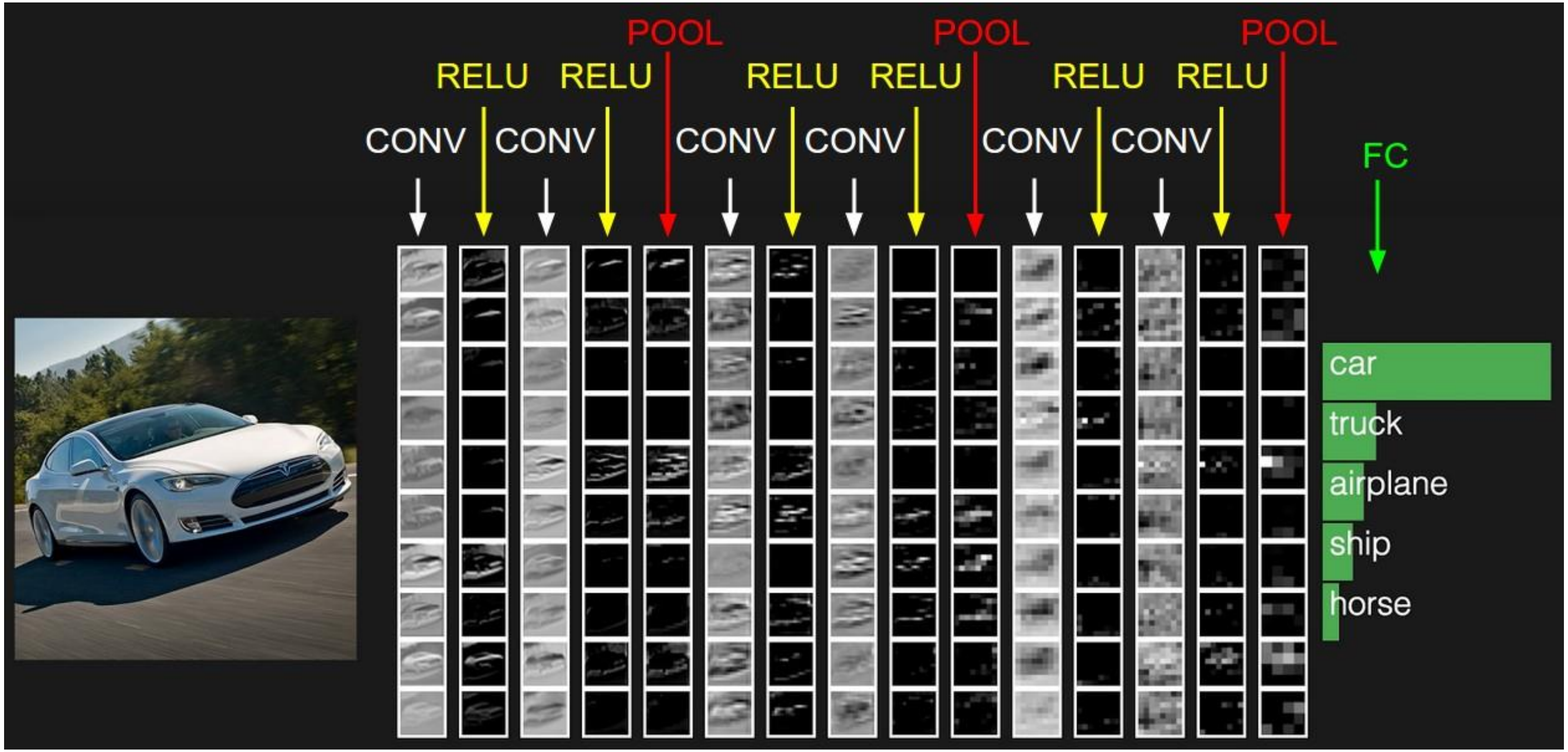


Rectified Linear Units (ReLUs)

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

0.77						

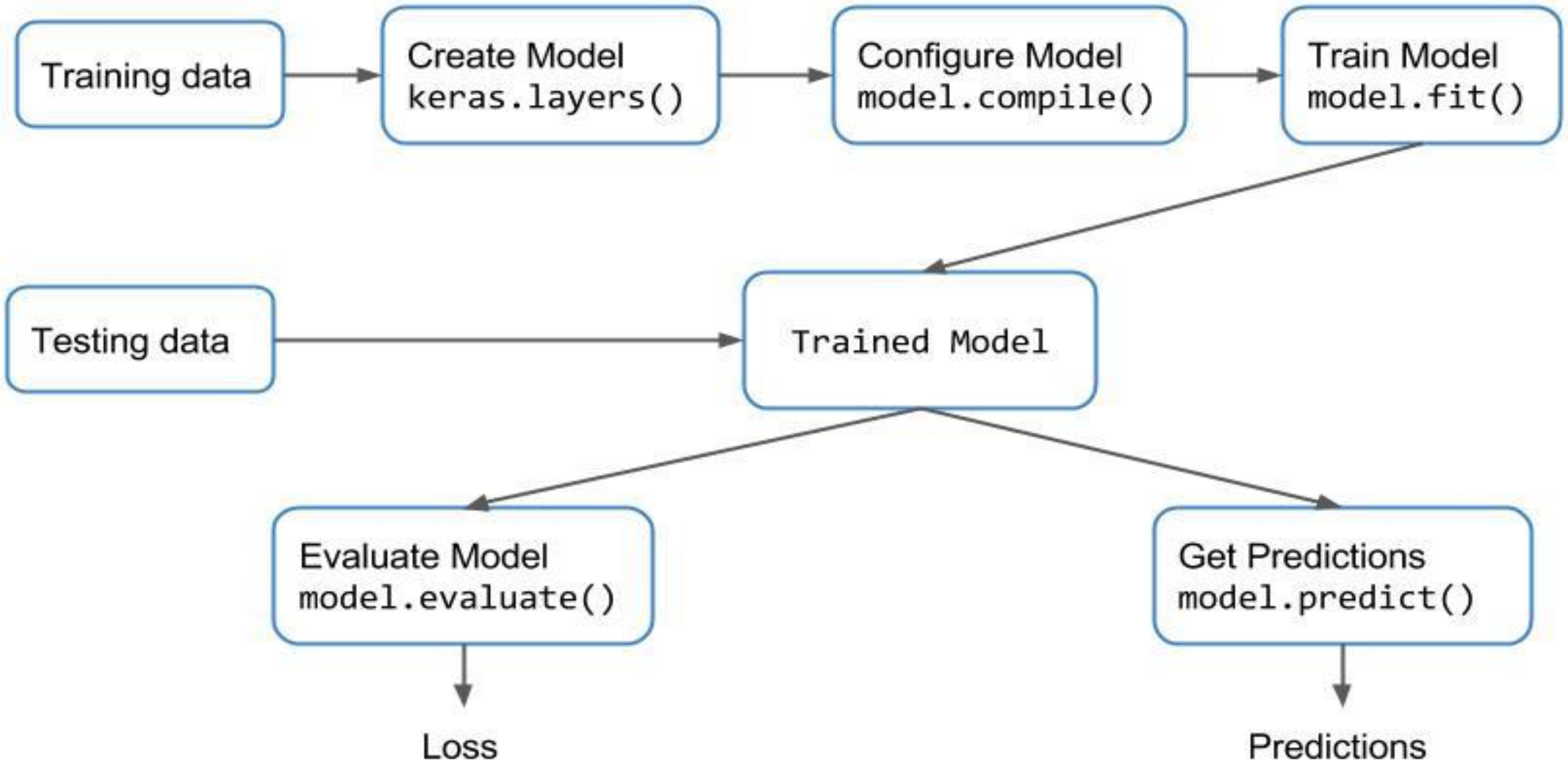
Inside The Layers Of A CNN



3D CNN Demo

http://www.scs.Ryerson.ca/~aharley/vis/conv/

Keras Workflow for training the network



Training data

Create Model
keras.layers()

Configure Model
model.compile()

Train Model
model.fit()

Testing data

Trained Model

Evaluate Model
model.evaluate()

Get Predictions
model.predict()

Loss

Predictions

COMMUNITY
INNOVATION
WORKSPACE

Let’s Build Your Own CNN with Keras

تركيب
THEASSEMBLY
MAKE | SMART | THINGS

By Anudeepsekhar Bolimera

Thank You