NEURAL NETWORKS

Juan Jesús Roldán Gómez

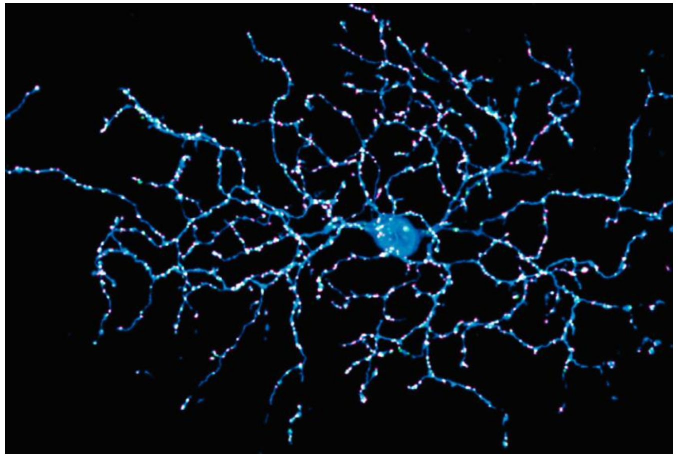
- Web: www.jjrg.org
- Email: juan.roldan@uam.es
- Office: EPS-UAM, B-321

NEURAL NETWORKS

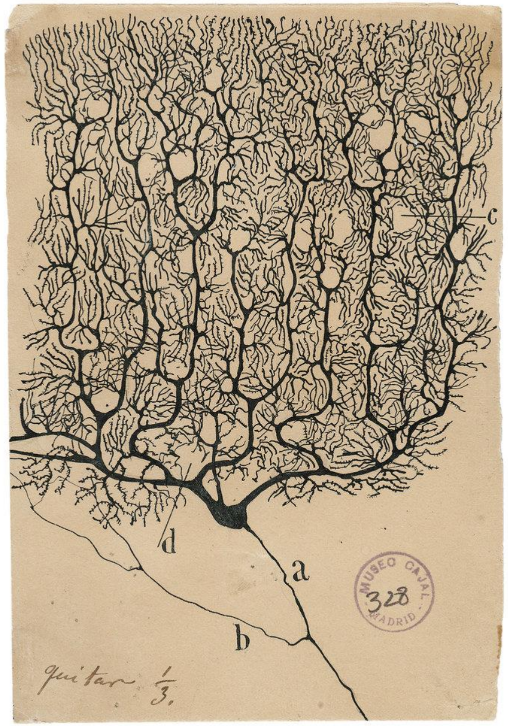
Bioinspiration

Biological neuron: An electrically excitable cell that forms the nervous tissue in most animals.

- **Sensory neurons:** Respond to stimuli such as touch, sound, or light.
- **Motor neurons:** Control muscle contractions and glandular outputs.
- **Interneurons:** Connect neurons to other neurons in **neural circuits**.



Super-Resolution Imaging of neuron, Harvard University

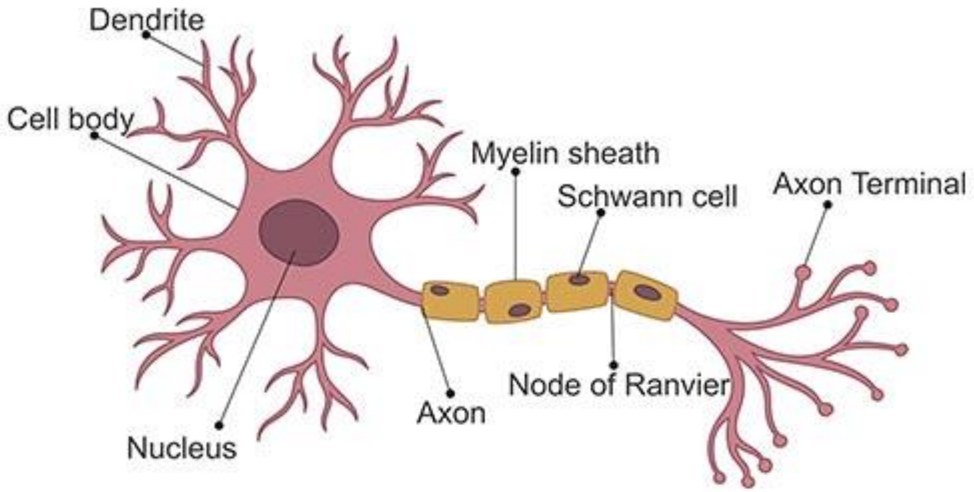


Purkinje neuron, Santiago Ramón y Cajal

Bioinspiration

Biological neuron: An electrically excitable cell that forms the nervous tissue in most animals.

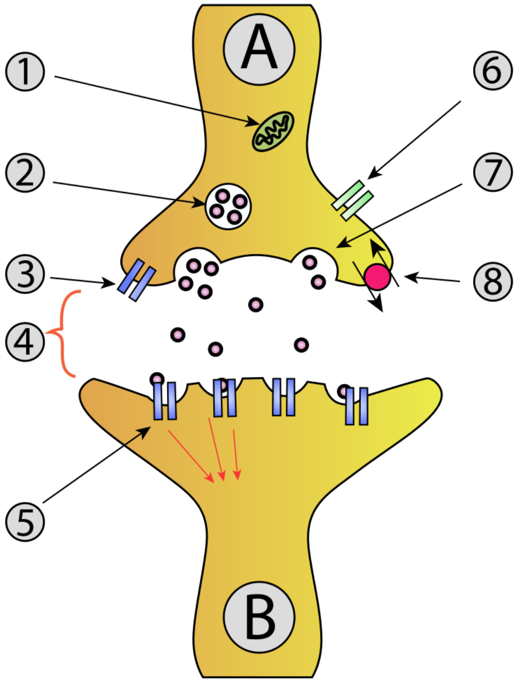
- **Soma:** Body of the neuron that contains the nucleus, where most cellular processes take place.
- **Dendrites:** Cellular extensions with many branches, which receive most of the inputs to the neuron.
- **Axon:** Cable-like projection that carries the outputs from the soma and some information back to it.



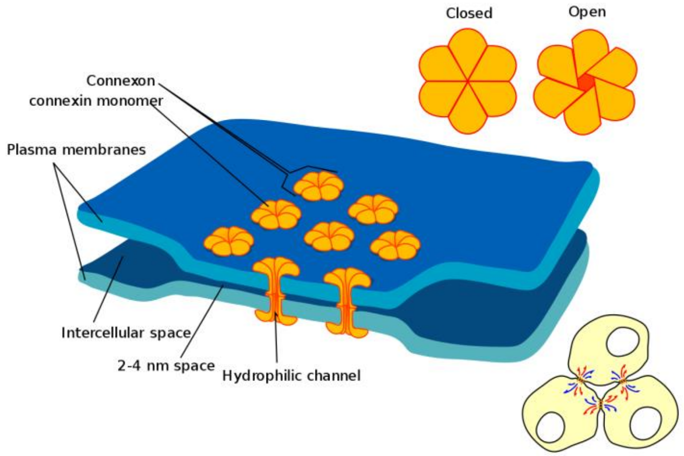
NEURAL NETWORKS

Bioinspiration

Synapse: Neural junction used for communication between neurons.



- **Chemical synapse:** The first neuron releases neurotransmitter molecules into the small space adjacent to the second one, these molecules attach to the receptors of that neuron, and then are cleared through various mechanisms.
  - Long distance: 20-40 nm.
  - Gain: Signal can be amplified.
  - Perception and though.
- **Electrical synapse:** Mechanical and electrically conductive link between two neurons in a narrow gap called gap junction.
  - Short distance: 3.8 nm.
  - High speed.
  - Defensive reflexes.



Bioinspiration

Neural circuit: Population of neurons interconnected by synapses to carry out a specific function.

- **Diverging circuit:** One neuron synapses with a number of postsynaptic cells (e.g., motor neurons with muscles).
- **Converging circuit:** Inputs from many sources are converged into one output (e.g., respiratory center of the brainstem).
- **Reverberating circuit:** It produces a repetitive output by sending the signal back to the initial neuron (e.g., respiratory muscles in inhalation and exhalation).
- **Parallel after-discharge circuit:** One neuron inputs to several chains of neurons that converge to one neuron, producing different delays in the original signal (e.g., some reflex arcs).

NEURAL NETWORKS

Bioinspiration

Large scale brain networks: Collections of neural circuits in widespread brain regions that show functional connectivity.

- Default mode network.
- Salience network.
- Dorsal attention network.
- Frontoparietal network.
- Sensorimotor network.
- Visual cortex.
- …

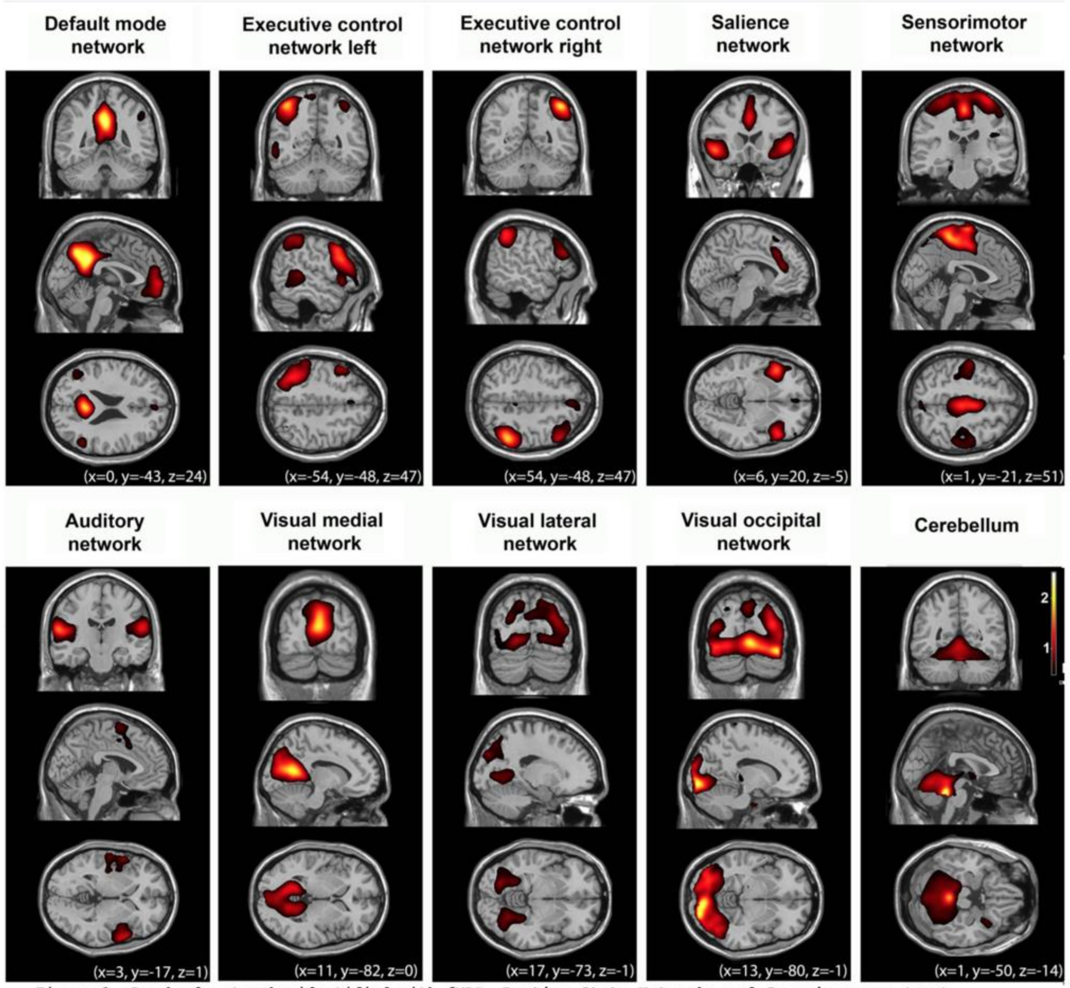


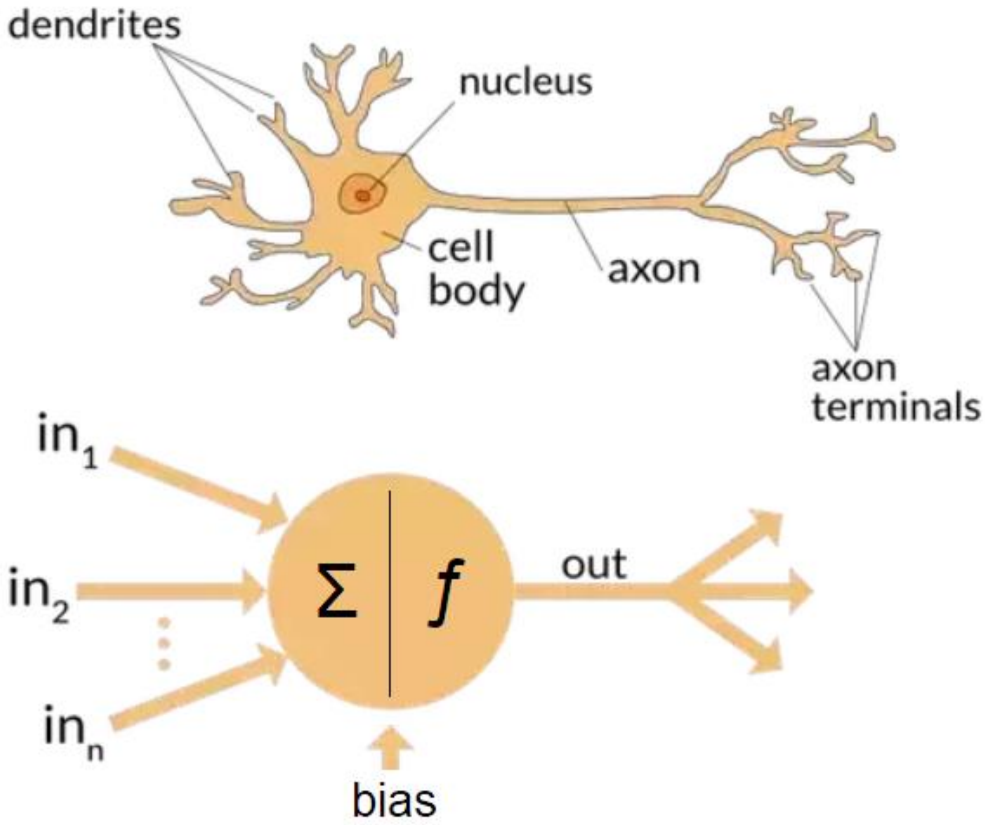
Figure 1. Cerebral networks identified with fMRI. **Resting State Networks and Consciousness** (2012) Lizette Heine, Andrea Soddu, Francisco Gómez, Audrey Vanhaudenhuyse, Luaba Tshibanda, Marie Thonnard, Vanessa Charland-Verville, Murielle Kirsch, Steven Laureys, and Athena Demertzi doi:10.3389/fpsyg.2012.00295

NEURAL NETWORKS

Bioinspiration

Biology	Computer Science
Neuron	Artificial Neuron
Neural Circuit	Artificial Neural Network

Artificial neural networks are inspired by **biological neural circuits**, but they are usually not strict copies of them.

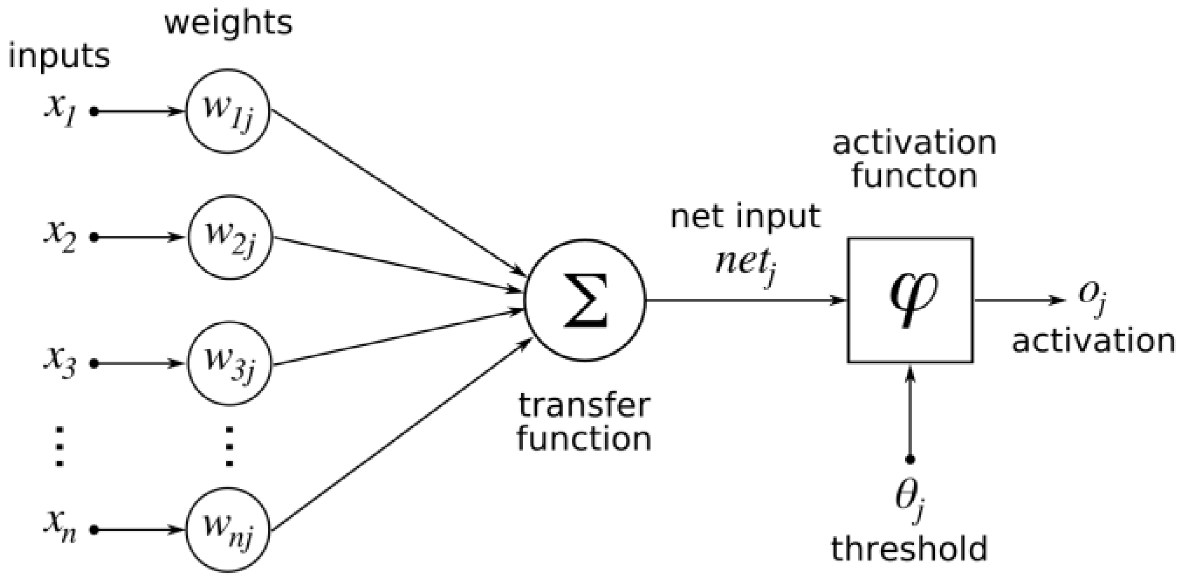


Artificial Neural Networks

Artificial neuron:

Biological neuron	Artificial neuron
Dendrites	Input vector
Soma	Transfer function
Axon	Activation function
Synapses	Output vector

Difference: Biological neurons fire in discrete pulses when the potential reaches a certain threshold, whereas artificial neurons can output continuous values.



Artificial Neural Networks

Artificial neuron:

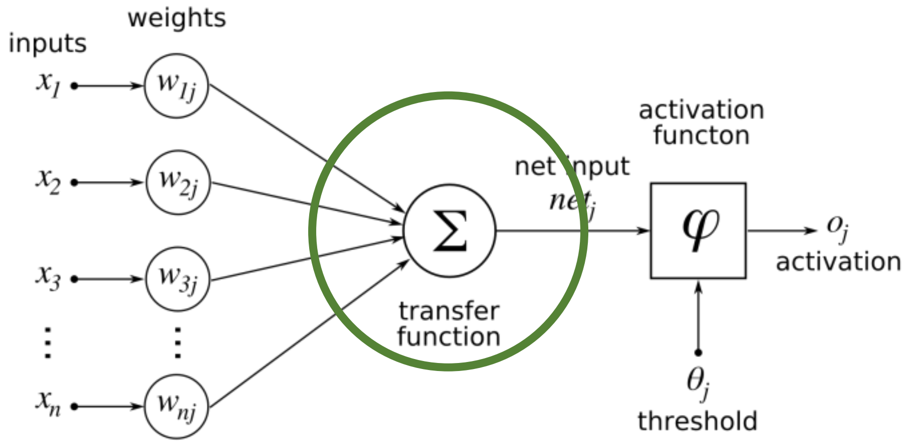
Transfer function:

Usually, a weighted sum of the inputs plus a bias term.

$$x = \sum_{i=1}^{n} w_i * x_i + b$$

Parameters:

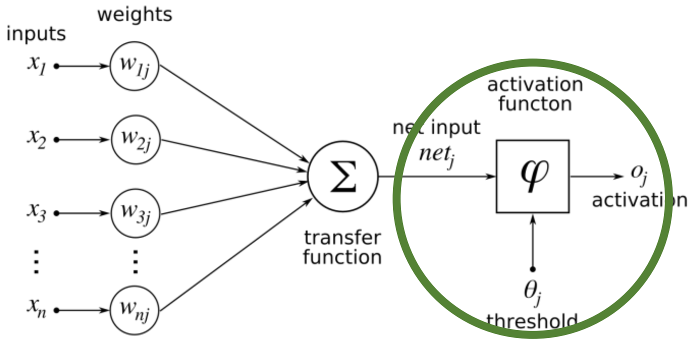
- **Weights** (w_i): Assign different importance to different inputs.
- **Bias** (b): Adjusts the output of the neuron.



NEURAL NETWORKS

Artificial Neural Networks

Artificial neuron:



Activation functions:

- Step function:

$$y = \begin{cases} 0 & if \quad u \leq \theta \\ 1 & if \quad u > \theta \end{cases}$$

- Rectifier:

$$y = \max(0, x)$$

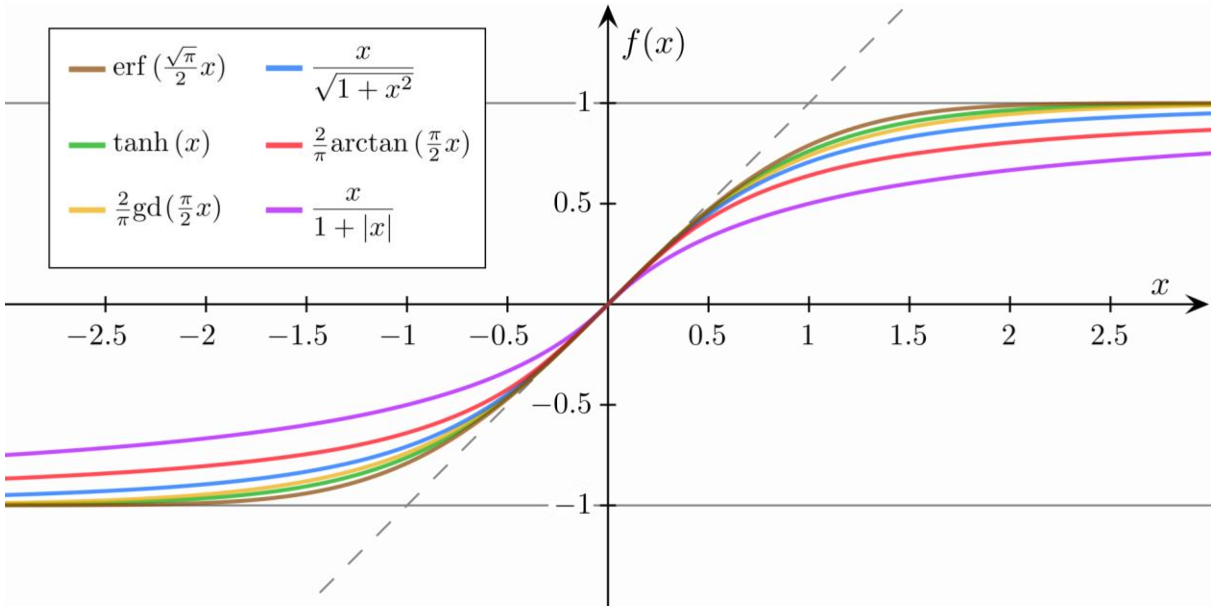
- Sigmoid functions:

Logistic function: $y = \frac{1}{1+e^{-x}}$ *(Tensorflow and Keras)*

Hyperbolic tangent: $y = \tanh(x)$

Arctangent function: $y = \arctan(x)$

…



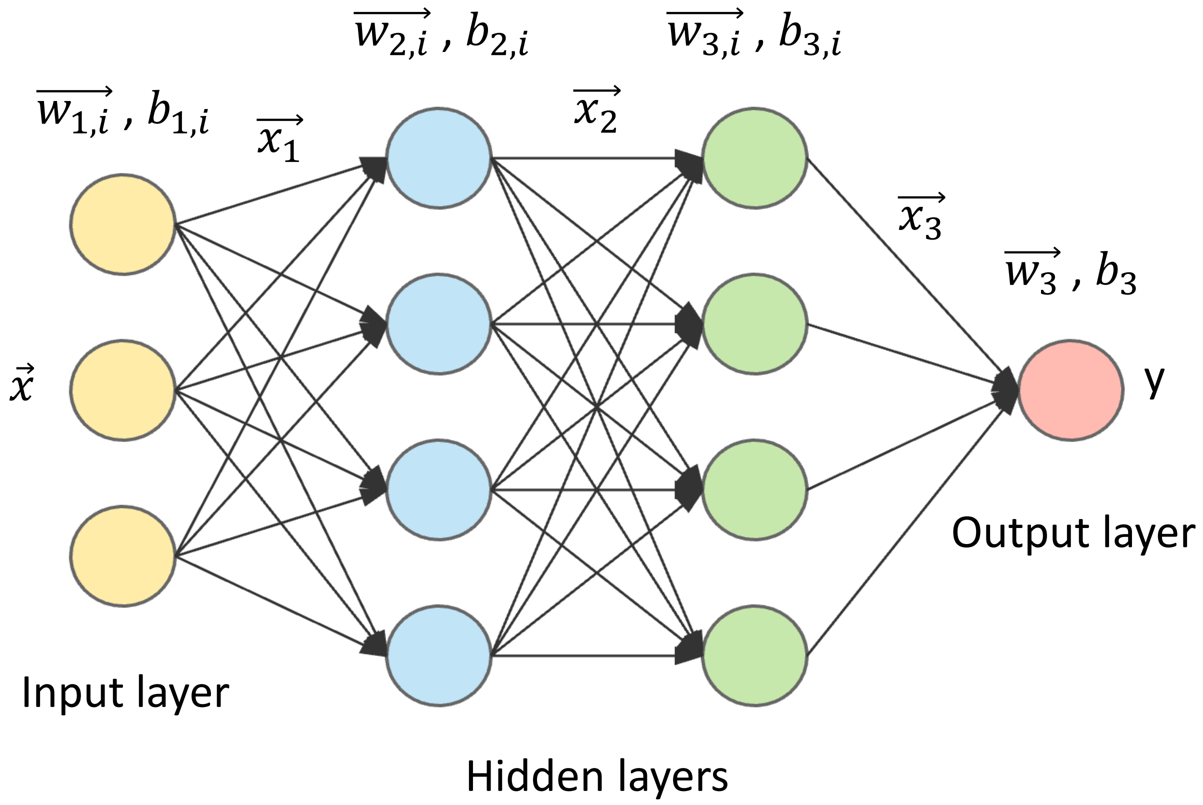
NEURAL NETWORKS

Artificial Neural Networks

Prediction:

Compute neural network output (y) from inputs ($\vec{x}$) using parameters (w and b).

Algorithm:
For each neuron i of each layer l…
$x_{l,i} = f_l(\overrightarrow{w_{l,i}} * \overrightarrow{x_{l-1}} + b_{l,i})$



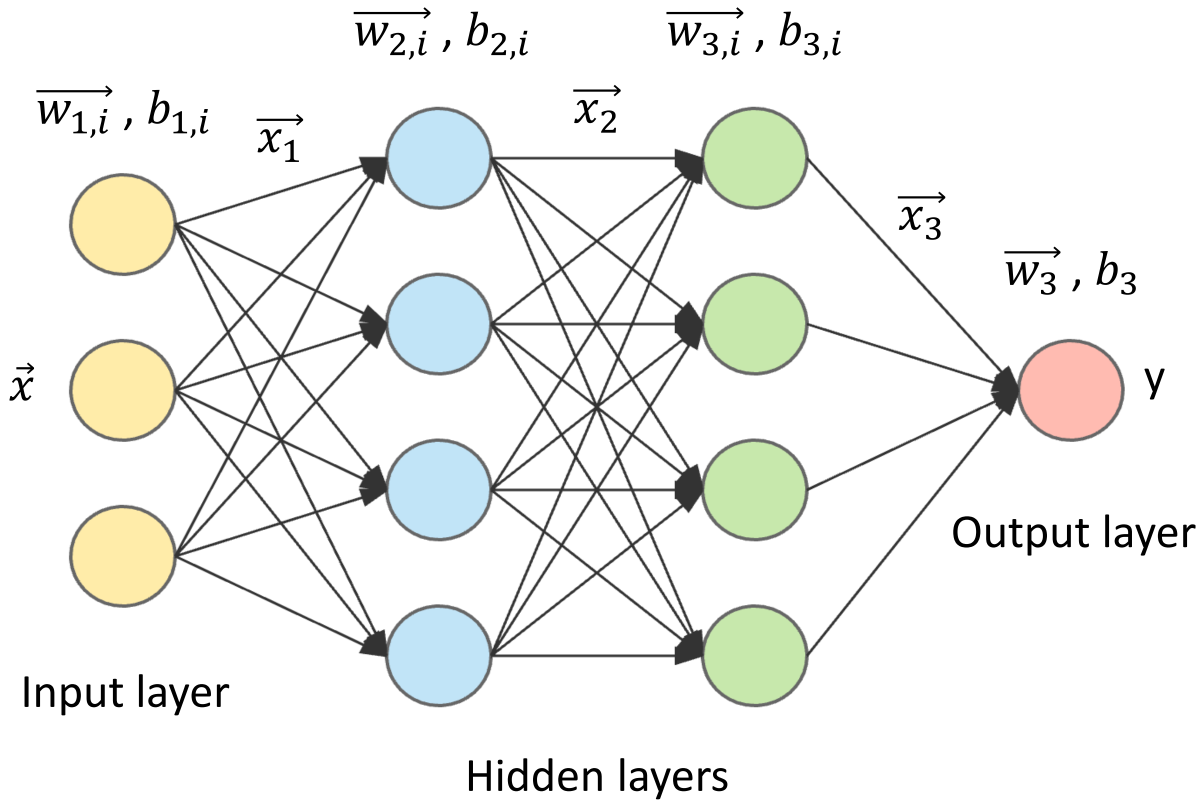
Artificial Neural Networks

Learning:

Adjust neural network parameters (w and b) from the input ($\vec{x}$) and desired output (y) through the backpropagation of errors.

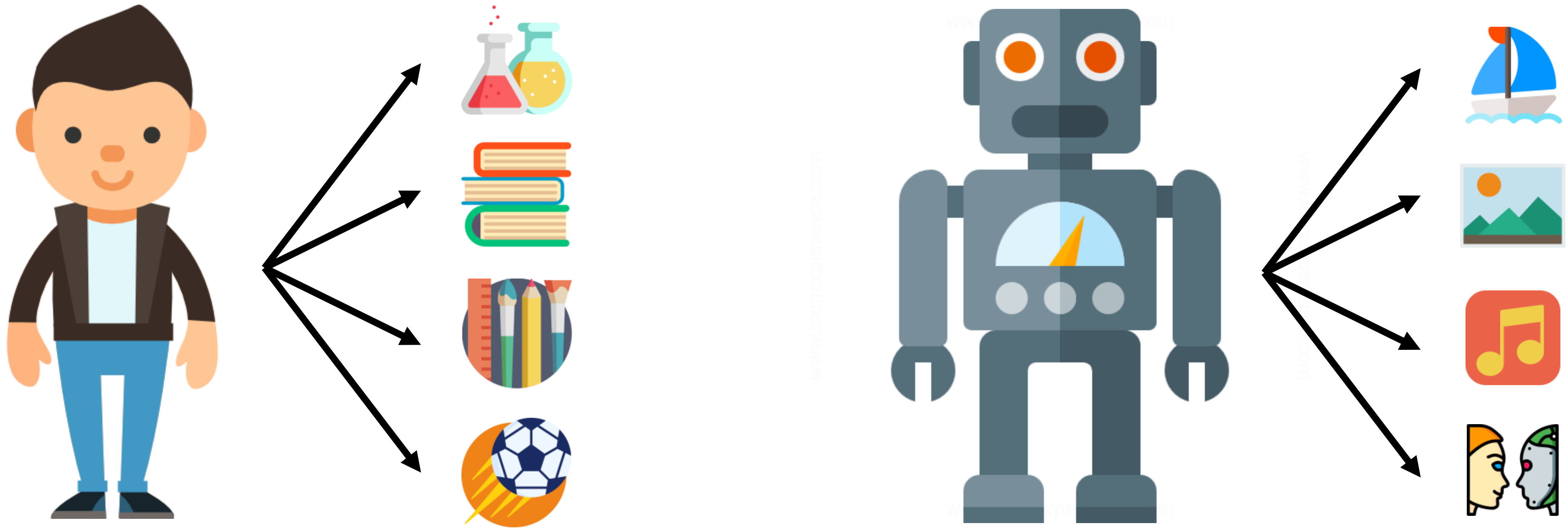
Algorithm:

1. Given a dataset with $\vec{x}$ and y values
2. Initialize w and b values randomly
3. Forward propagation: $\vec{x}$ → … → $\hat{y}$
4. Compute error: $e_y = (y - \hat{y})^2$
5. Backpropagation: e_y → … → $\overrightarrow{e_x}$
6. Update w and b values with optimization method

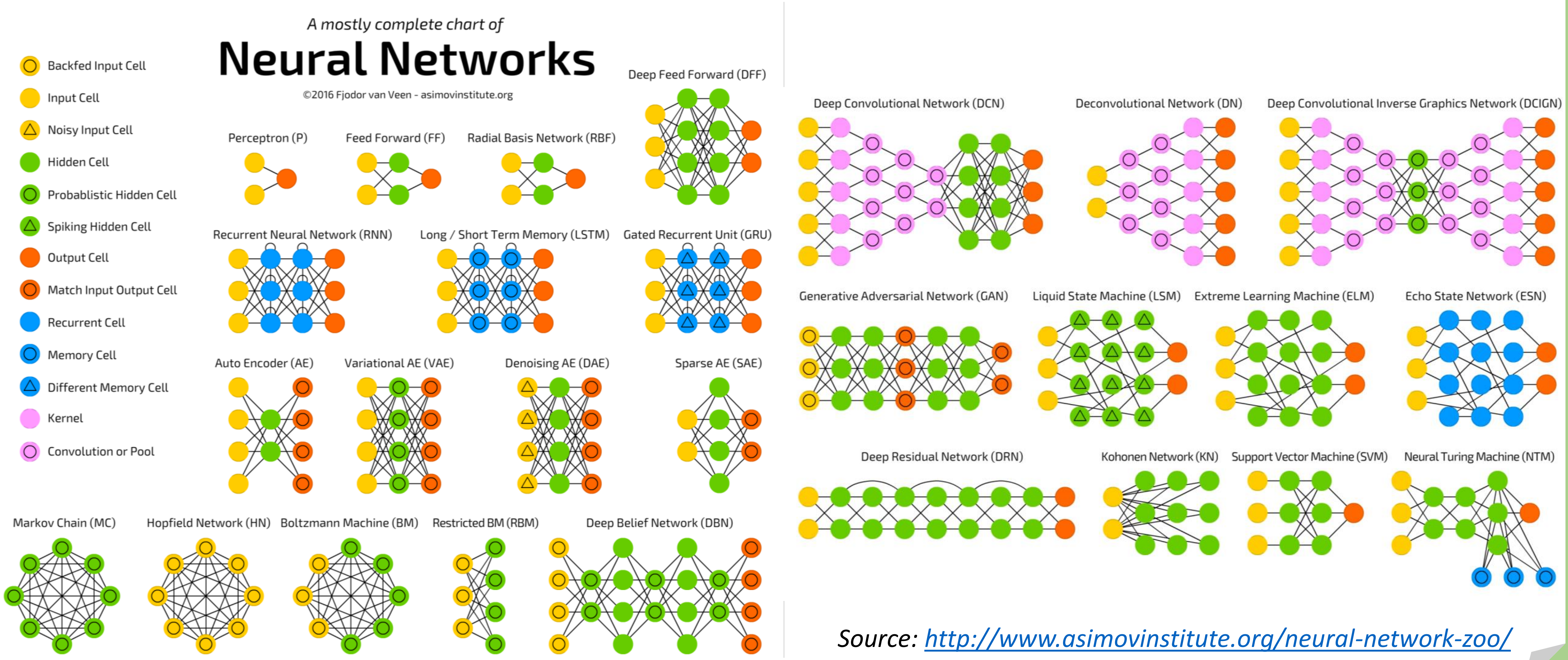


Artificial Neural Networks

Specialization:



NEURAL NETWORKS



Source: http://www.asimovinstitute.org/neural-network-zoo/

Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a class of Artificial Neural Networks that use **convolutional layers** instead of **fully connected layers** and are commonly applied to analyze **visual imagery**.

They are applied to image and video recognition, recommender systems, image classification, image segmentation, medical image analysis, natural language processing, brain-computer interfaces, and financial time series.

Convolutional layers reduce the **complexity** and tendency to **overfitting** of **fully connected layers**, where each neuron of one layer is connected to all the neurons of the next one.

CNNs are **inspired** by the **animal visual cortex**, where each cortical neuron responds to the stimuli from a restricted visual region, and the detections of different neurons partially overlap to cover the whole visual field.

NEURAL NETWORKS

Convolutional Neural Networks

Why CNNs?

Example: Animal detector

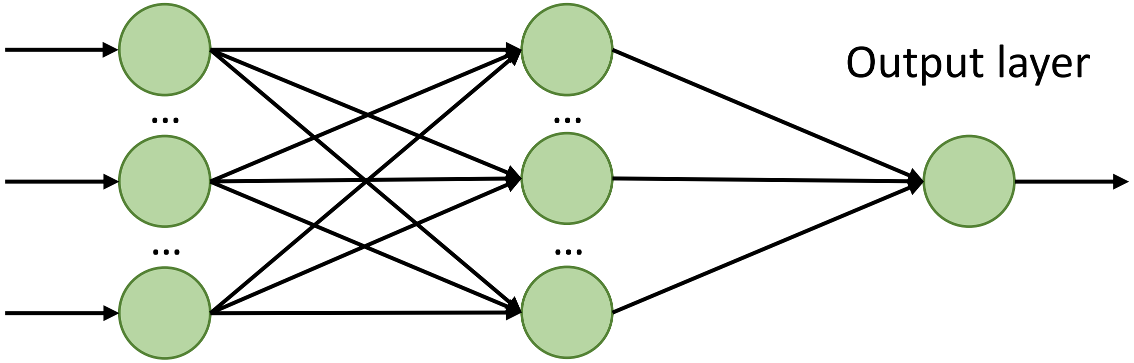
Input: Image

300 x 300 x 3 = 270k pixels

Input layer

Hidden layer

Output layer



Output: Detection

Input layer	Hidden layer	Output layer
270k neurons	1k neurons	1 neuron
270k w's	270M w's	1k w's
270k b's	1k b's	1 b
540,000 params	270,001,000 params	1,001 params

Total:
270,542,001 params

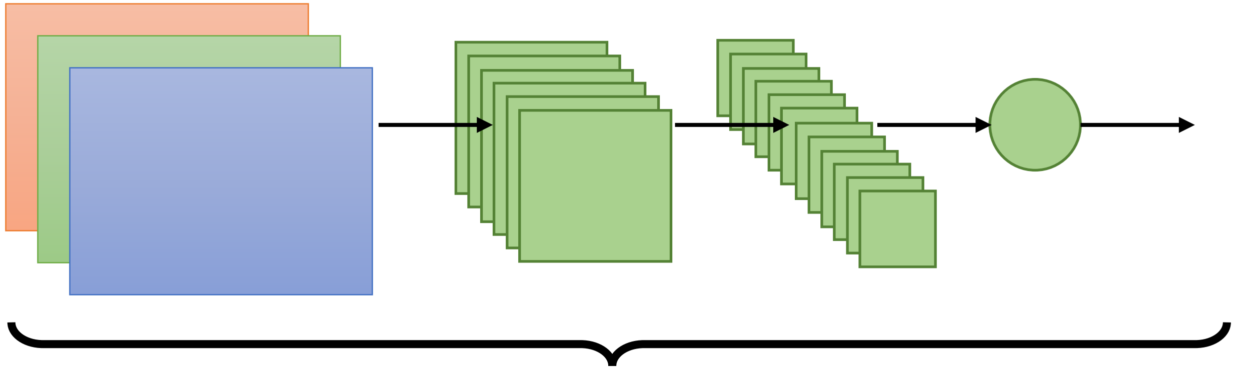
Convolutional Neural Networks

Why CNNs?

Example: Animal detector

Input: Image

300 x 300 x 3 = 270k pixels

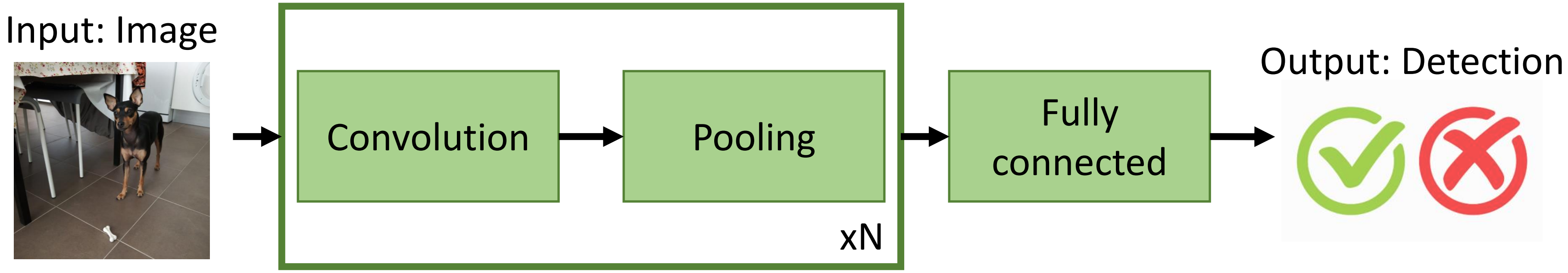


Output: Detection

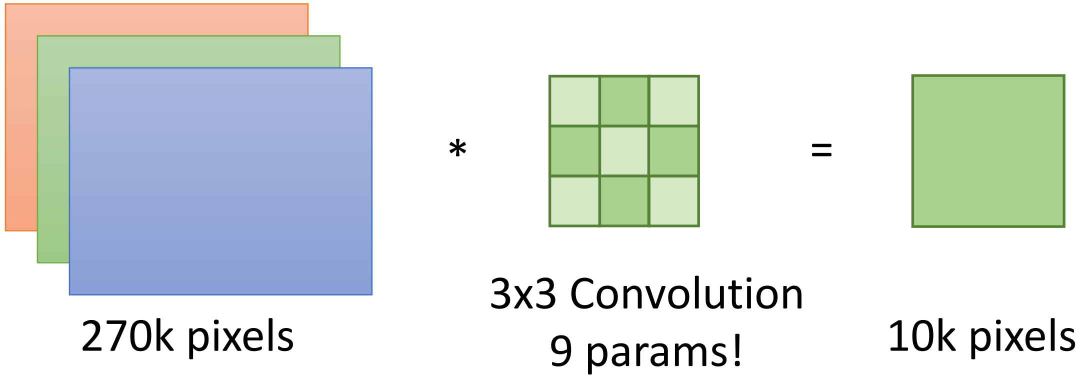
Parameters ~ 100k << 270M

Convolutional Neural Networks

Convolutional NNs:



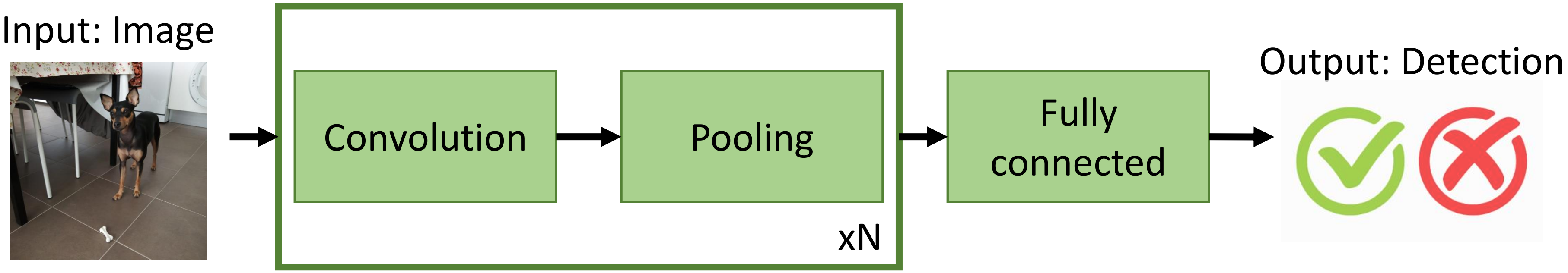
Convolution



NEURAL NETWORKS

Convolutional Neural Networks

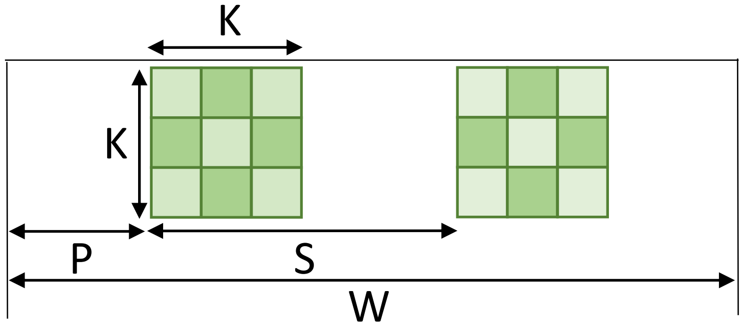
Convolutional NNs:



Convolution

Parameters:

- Input size: W
- Kernel size: K
- Stride: S
- Zero padding: P

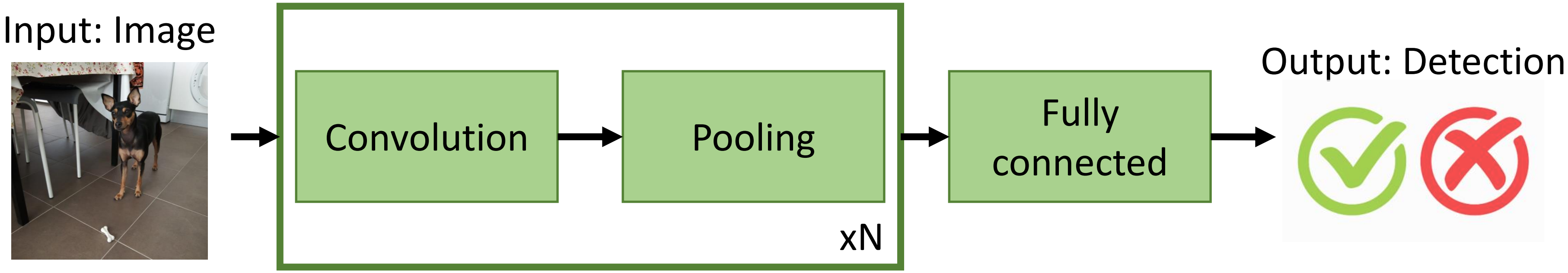


$$R = \sum_{i=1}^{K} \sum_{j=1}^{K} Image(i,j) * Kernel(i,j)$$

$$N = \frac{W - K + 2P}{S} + 1$$

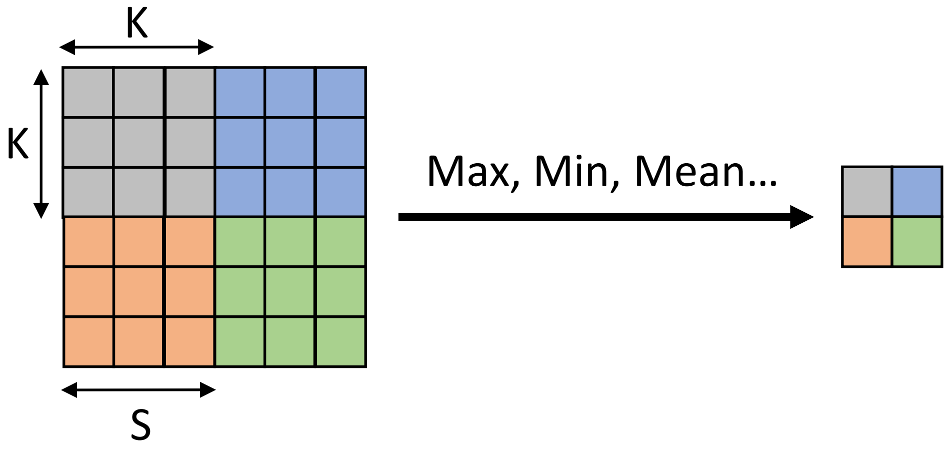
Convolutional Neural Networks

Convolutional NNs:



Pooling:

Parameters:
- Input size: W
- Kernel size: K
- Stride: S



NEURAL NETWORKS

Recurrent Neural Networks

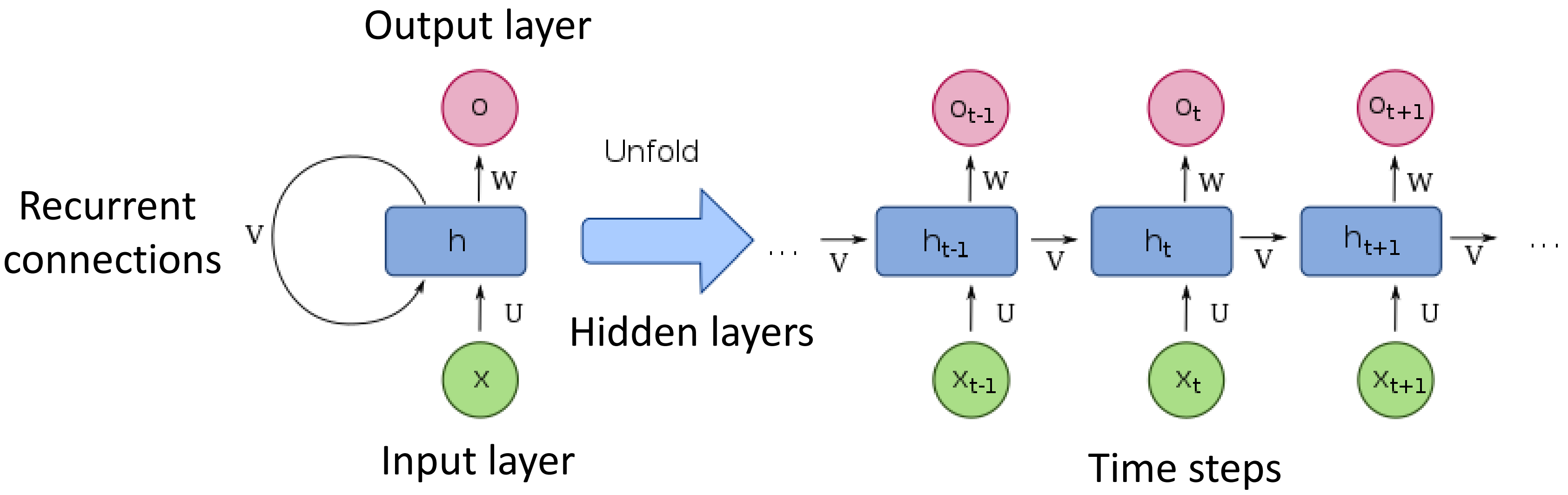
Recurrent Neural Networks (RNNs) are a class of Artificial Neural Networks that form **directed graphs along temporal sequences**, which allows them to have a **temporal dynamic behavior**, using their **internal state** to process variable length sequences of inputs.

They are applied to process temporal sequences in tasks like handwriting recognition, speech recognition, language translation, natural language processing...

While **traditional NNs** assume that **outputs** at a certain moment only depend on **inputs at that time**, **RNNs** can process **outputs** that depend on **past and even future inputs**.

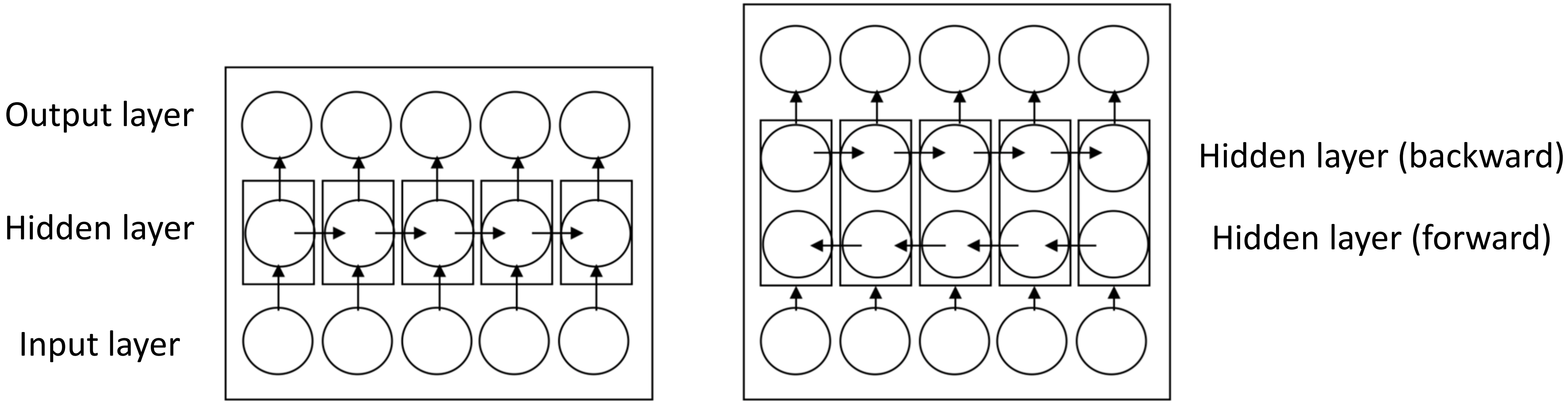
Recurrent Neural Networks

Unidirectional RNNs: Each **neuron** in a **hidden layer** is connected to its **inputs**, **outputs**, and **previous state**.



Recurrent Neural Networks

Bidirectional RNNs: They have **two hidden layers of opposite directions** connected to the same inputs and outputs, so the output layer can get **information from past** (backward) **and future** (forward) **states** simultaneously.



Unidirectional RNN

Bidirectional RNN

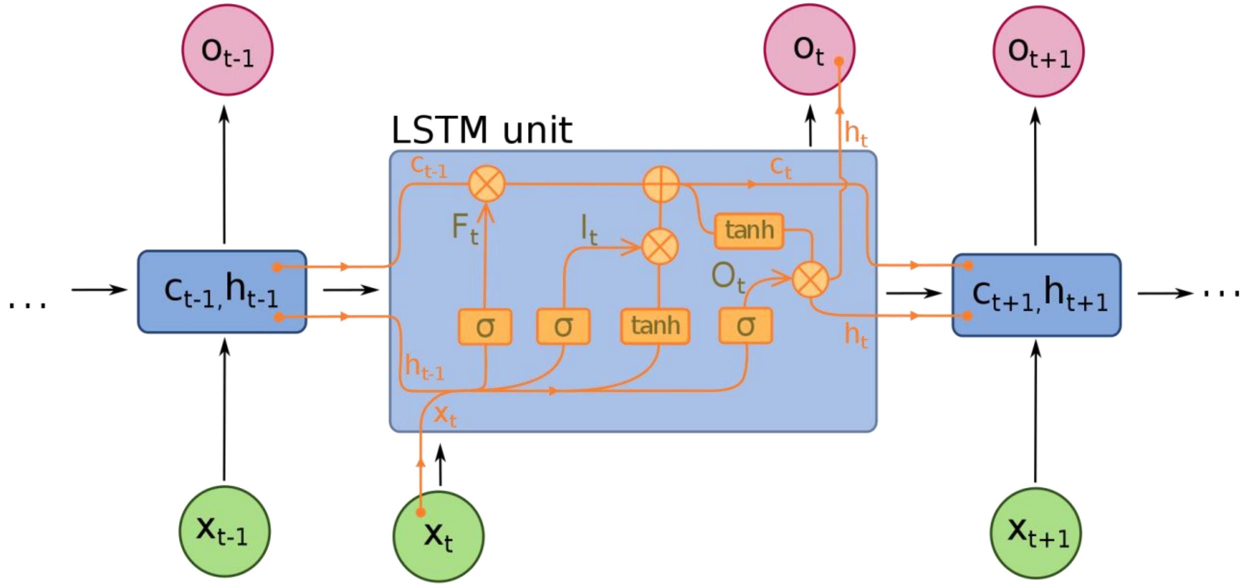
NEURAL NETWORKS

Recurrent Neural Networks

Long Short-Term Memory (LSTM): Architecture of RNN that can learn order dependence in sequence problems even when there are lags of unknow duration between the relevant events.

Elements:

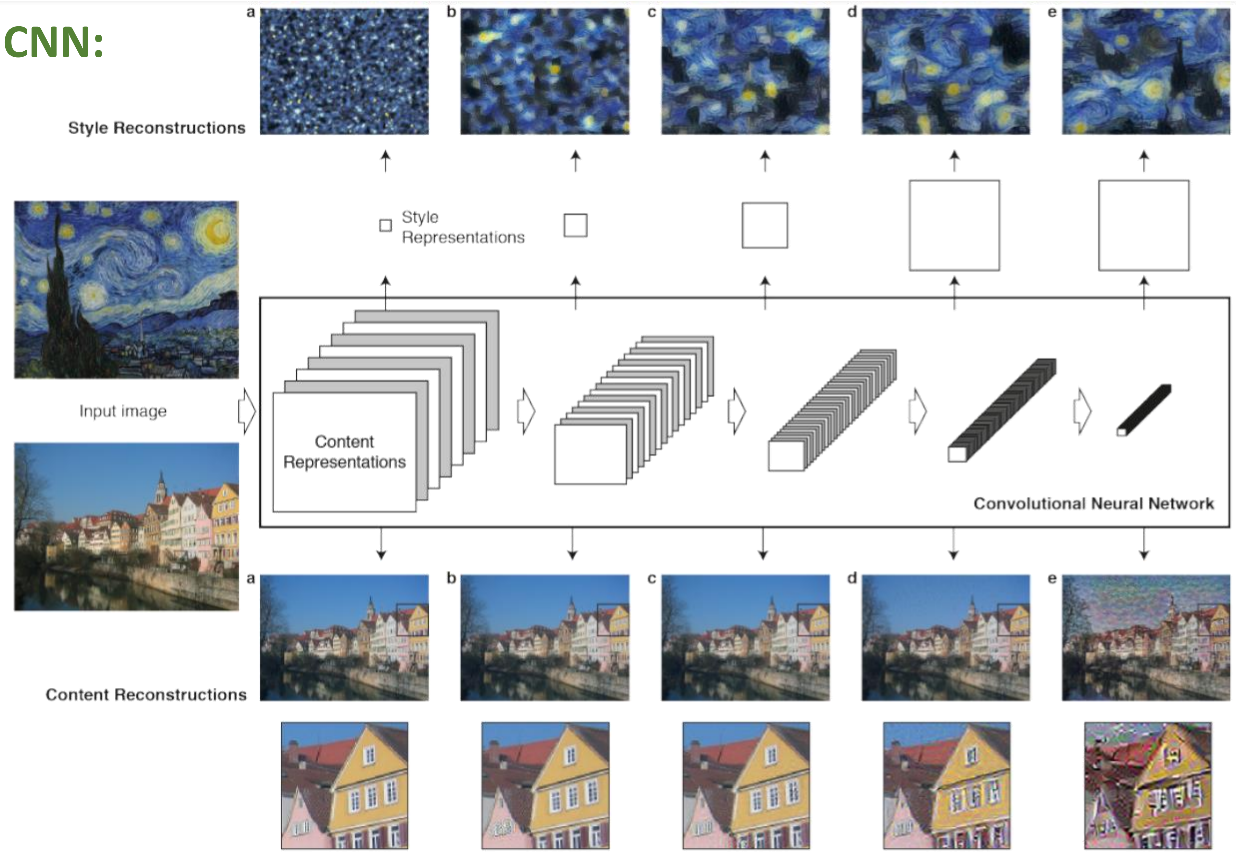
- **Memory cell:** Remembers values over arbitrary time intervals.
- **Input gate** (I_t): Controls the information received by the unit filtering the irrelevant inputs.
- **Output gate** (O_t): Controls the information sent by the unit filtering the outputs affected by irrelevant memory contents.
- **Forget gate** (F_t): Resets the values stored at the memory cell when it is required.



NEURAL NETWORKS

Applications

CNN:

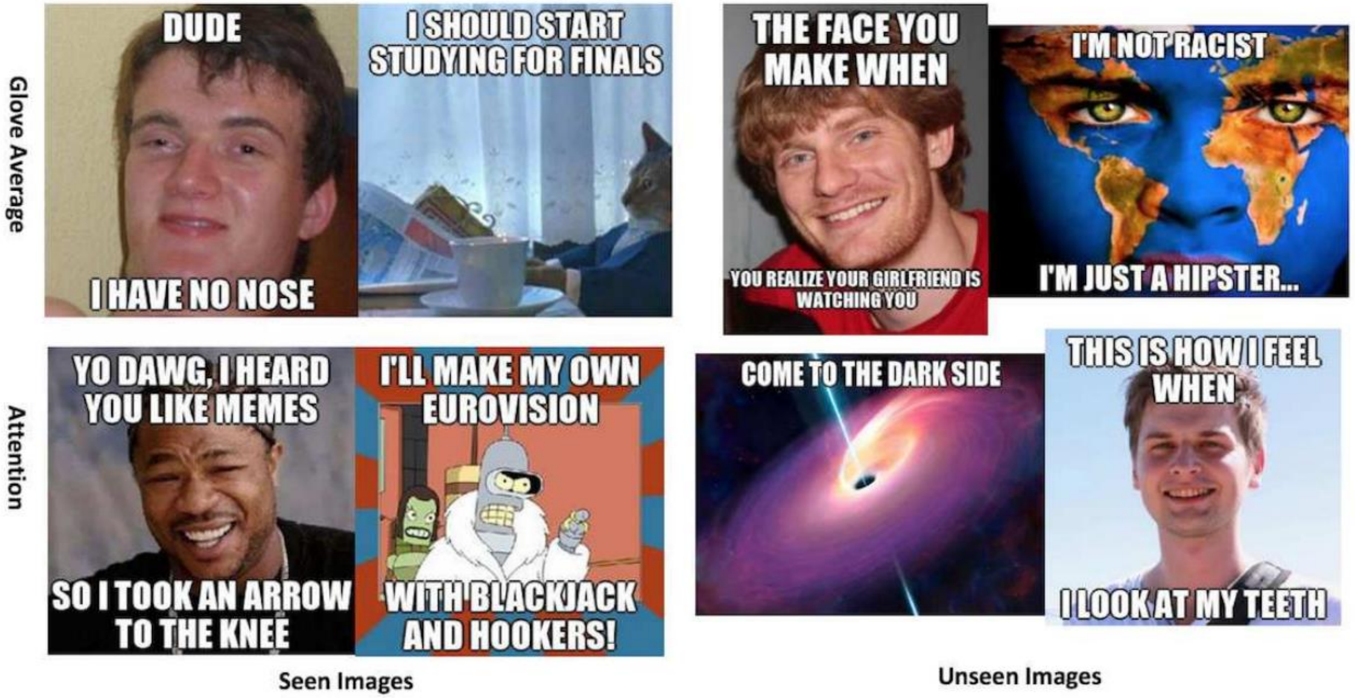


Source: Gatys et al. 2015 - https://arxiv.org/pdf/1508.06576.pdf

NEURAL NETWORKS

Applications

CNN+RNN:



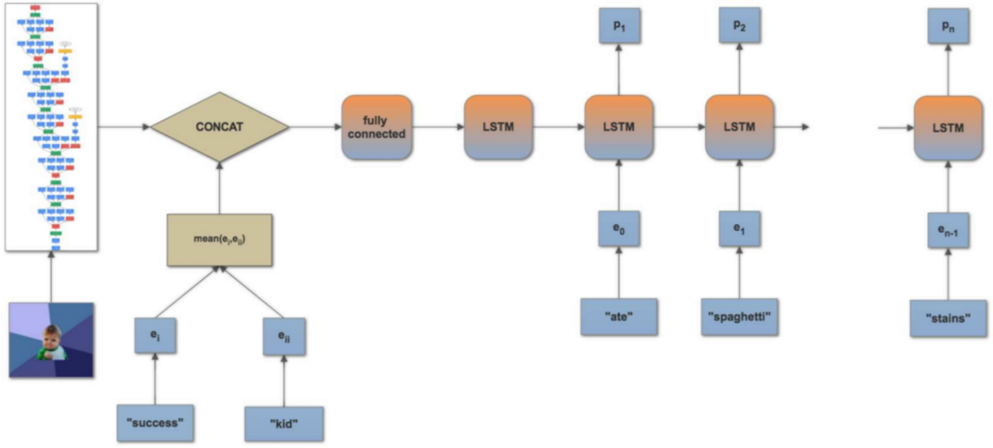
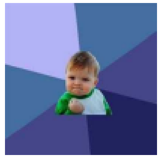


Image	Label	Caption
	• success kid	• Didnt study for a test still get a higher grade than someone who did • Ate spaghetti with a white shirt on no stains • New neighbors Free Wifi ...
	• awkward seal	• You laugh when your friend says something He was being serious • took a photo camera the wrong way • Goes to friends house Friend isn't there yet ...

Source: Peirson and Meltem, 2018 - https://arxiv.org/pdf/1806.04510.pdf

How to work with NNs

1. **Analyze the problem**
2. Choose an architecture
3. Train the model
4. Evaluate its performance

What do you want to do?

Which are the objectives?

Classification? Prediction? Control? Detection?

What data is available?

Logs? Text? Images? Video?

Is there enough data? *Is the data consistent?*

Should you use neural networks?

Can the problem be solved with traditional methods?

Which method works best? *Which is easier to apply?*

How to work with NNs

1. Analyze the problem
2. **Choose an architecture**
3. Train the model
4. Evaluate its performance

The process:

1. Choose an architecture: conventional, CNN, RNN…
2. Select the number of layers.
3. Give values to the hyperparameters.

Some advice:

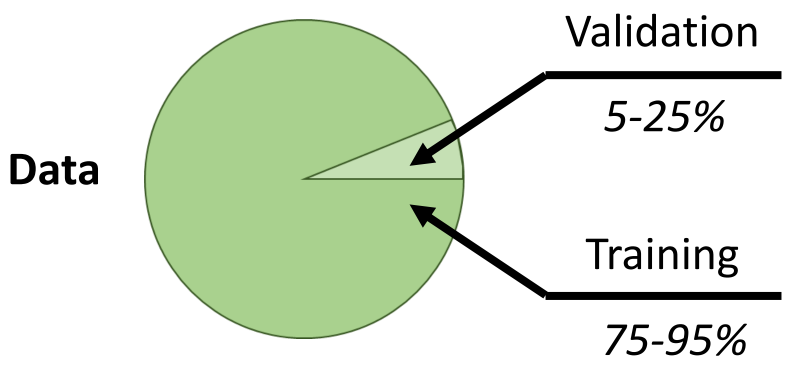
- Start from the easy and go to the complex.
- Look for inspiration in previous works.
- Similar application? Try similar architecture.
- Try different configurations until you get good results.

How to work with NNs

1. Analyze the problem
2. Choose an architecture
3. **Train the model**
4. Evaluate its performance

The process:

1. Split the data into training and validation sets.
2. Choose an optimization algorithm (e.g., gradient descent)
3. Train the model using the optimization algorithm and training set.



How to work with NNs

1. Analyze the problem
2. Choose an architecture
3. Train the model
4. **Evaluate its performance**

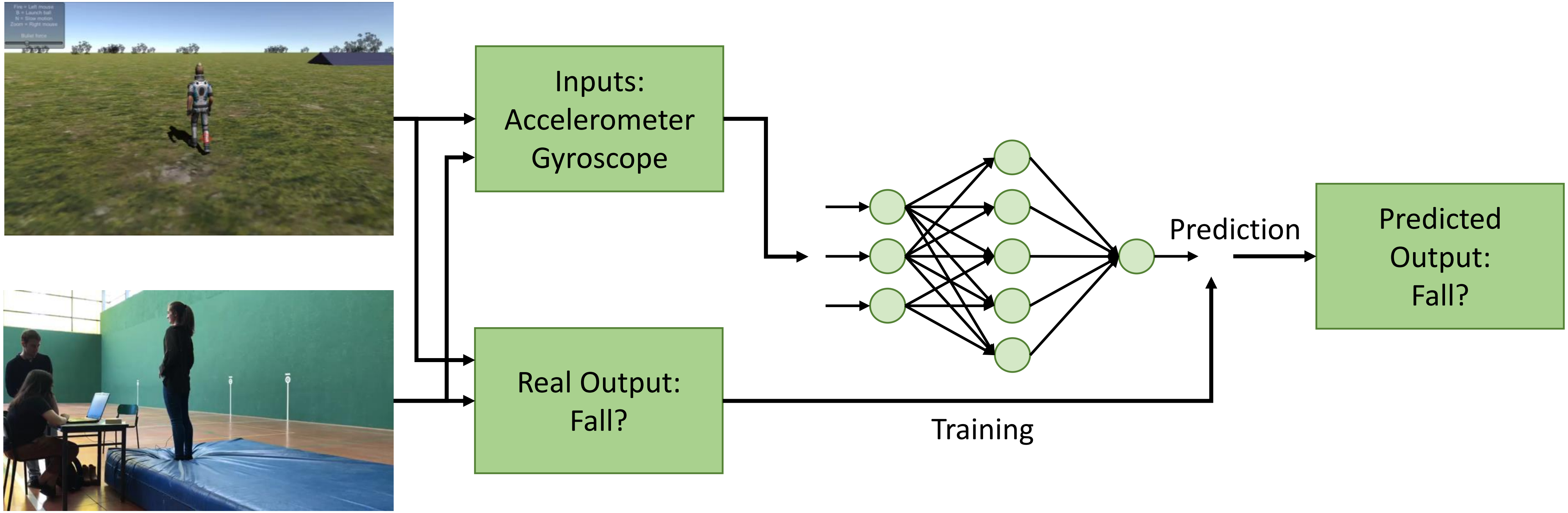
▲ Training error	▲ Validation error	→	Try again!
▼ Training error	▲ Validation error	→	Take care of overfitting
▼ Training error	▼ Validation error	→	Congratulations!

▲▼ Reference: Bayesian error (human error)

If results aren't good, keep calm and start again…
Sometimes it's a matter of magic!

NEURAL NETWORKS

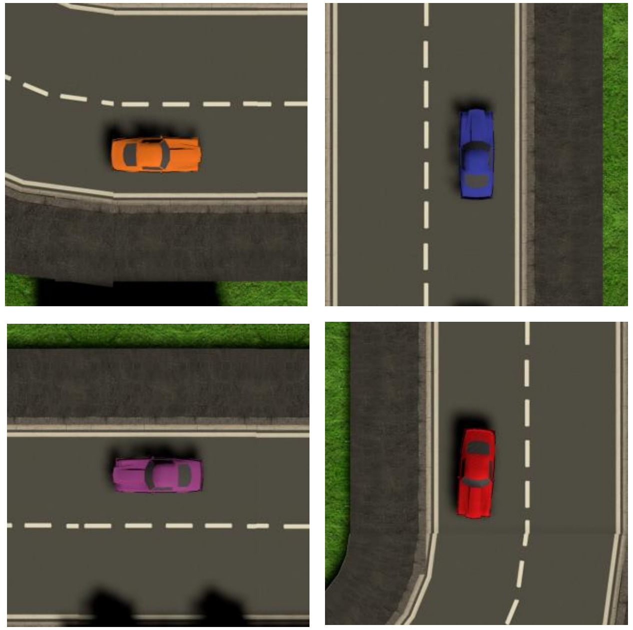
Fall detector

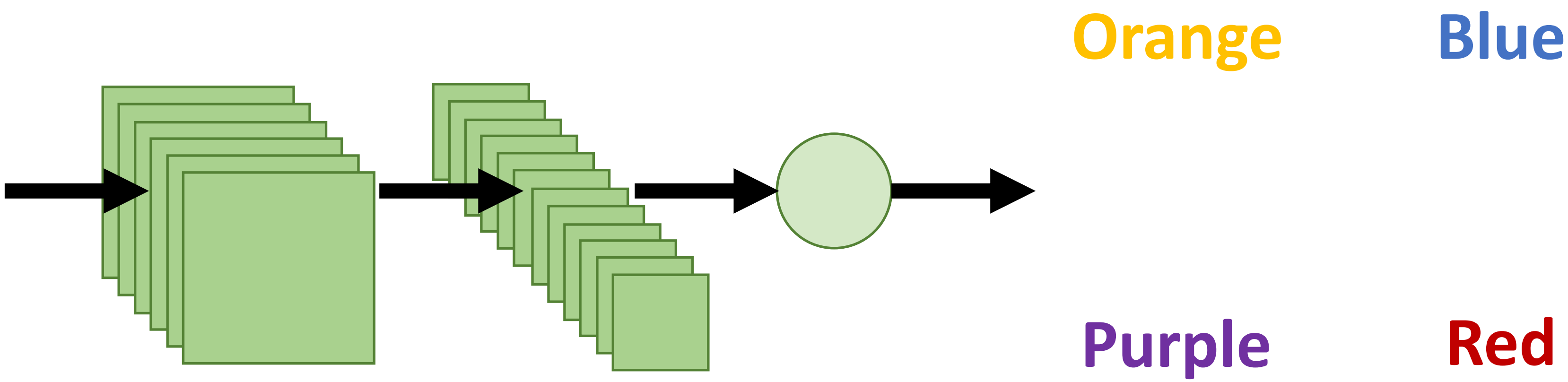


Link: *https://github.com/jjroldangomez/fall-detector*

NEURAL NETWORKS

Car detector





Link: https://github.com/jjroldangomez/car-detector