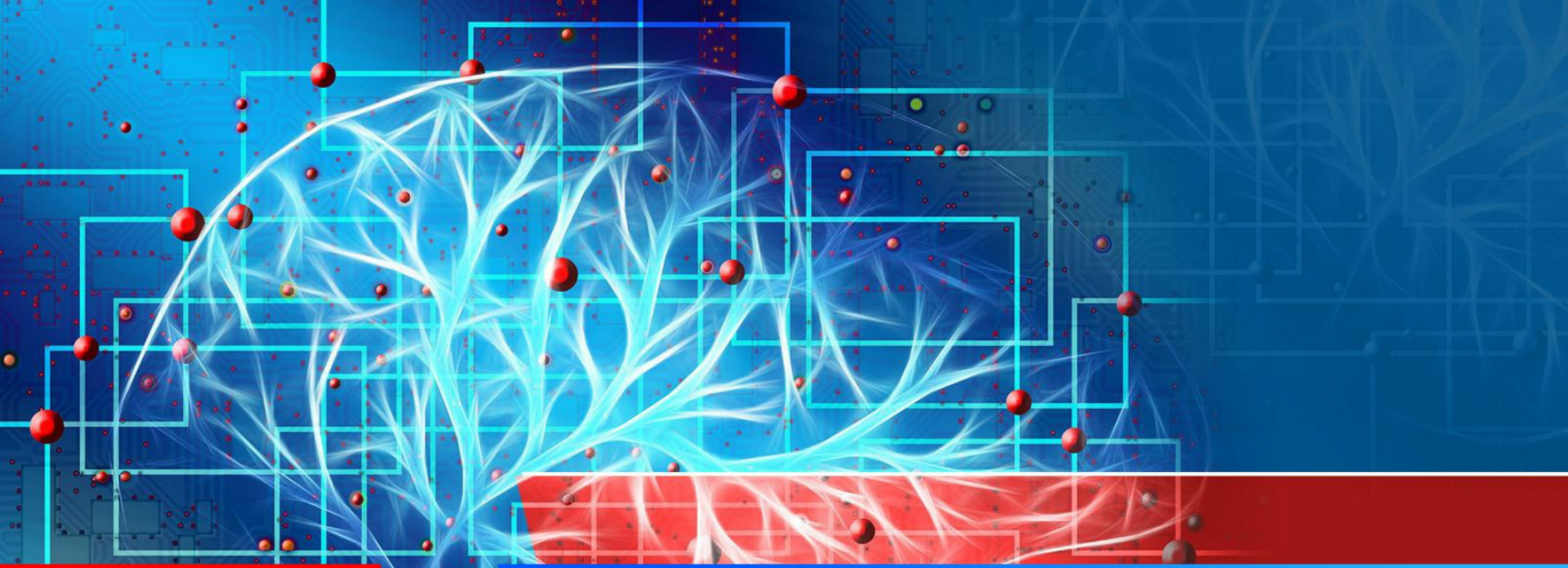


Neural Network Basics

Neural Network Definition

- An ***artificial neural network*** is an information-processing system that has certain performance characteristics in common with biological neural networks.

- Artificial neural networks have been developed as generalizations of mathematical models that mimic the way the human brain operates, based on the assumptions :
  - Information processing occurs at many simple elements called **neurons**.
  - Signals are passed between neurons over **connection** links.
  - Each connection link has an associated **weight**, which, in a typical neural net, multiplies the signal transmitted.
  - Each neuron applies an **activation function** (usually nonlinear) to its net input (sum of weighted input signals) to determine its output signal.

Artificial Neuron

- An **artificial neuron** is the basic building block that construct complicated neural networks which will make a particular computation based on other units it's connected to.

$$a(\mathbf{x}) = b + \sum_i w_i x_i = b + \mathbf{w}^\top \mathbf{x}$$

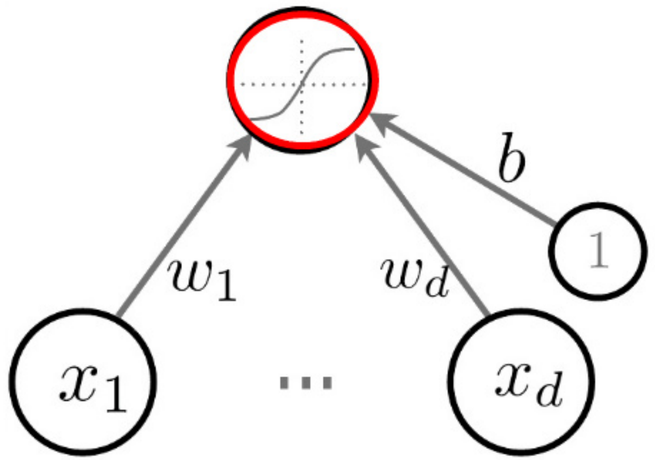
- Neuron (output) activation

$$h(\mathbf{x}) = g(a(\mathbf{x})) = g(b + \sum_i w_i x_i)$$

$\mathbf{w}$ are the connection weights

b is the neuron bias

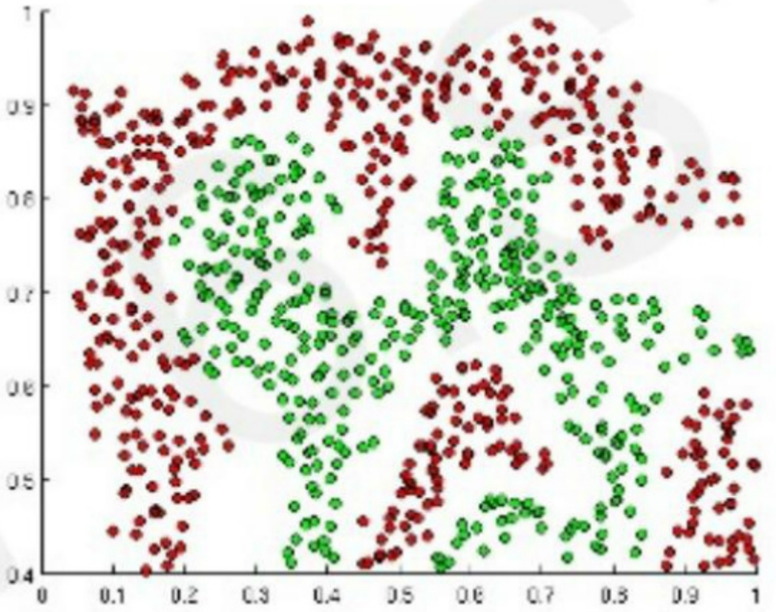
$g(\cdot)$ is called the activation function



Slide Credit: Hugo Larochelle

Importance of Activation Functions

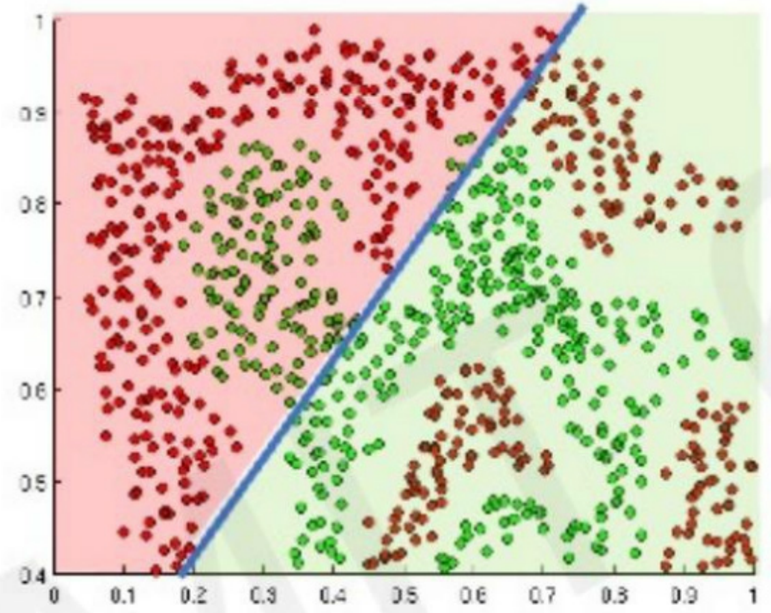
- The purpose of activation functions is to introduce **non-linearities** into the network.



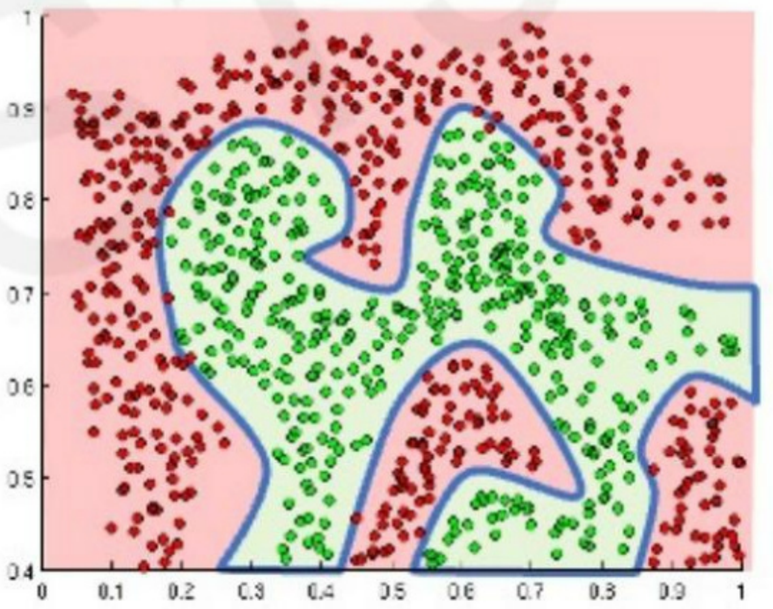
What if we wanted to build a neural network to distinguish green vs red points?

Importance of Activation Functions

- The purpose of activation functions is to introduce **non-linearities** into the network.



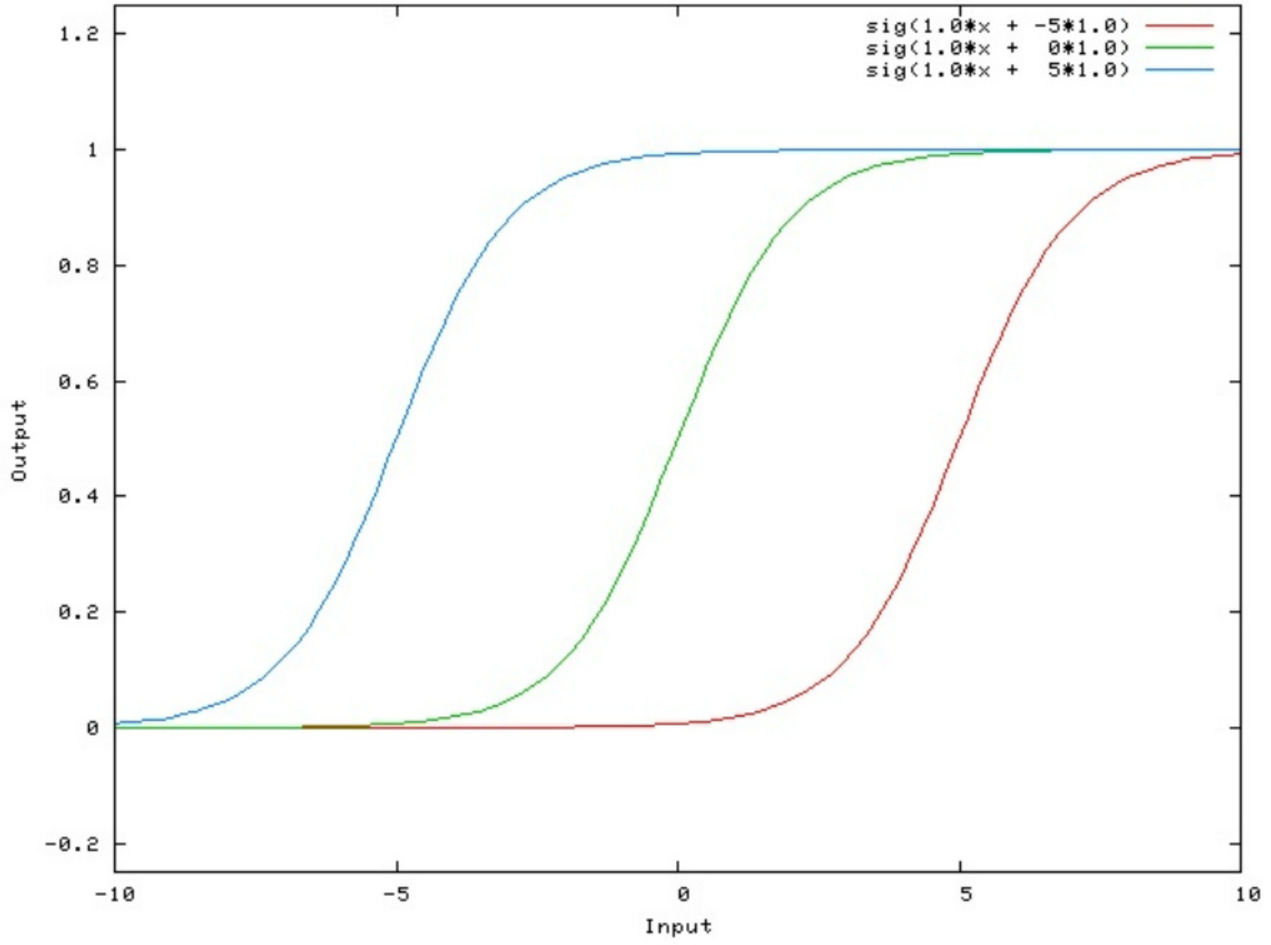
Linear activation functions produce linear decisions no matter the network size



Non-linearities allow us to approximate arbitrarily complex functions

Artificial Neuron

- Bias unit
  - allows the activation function to be shifted to the left or right, to better fit the data.
  - the bias only influences the output values, it doesn't interact with the actual input data.
- While changes to the weights alter the steepness of the activation function.



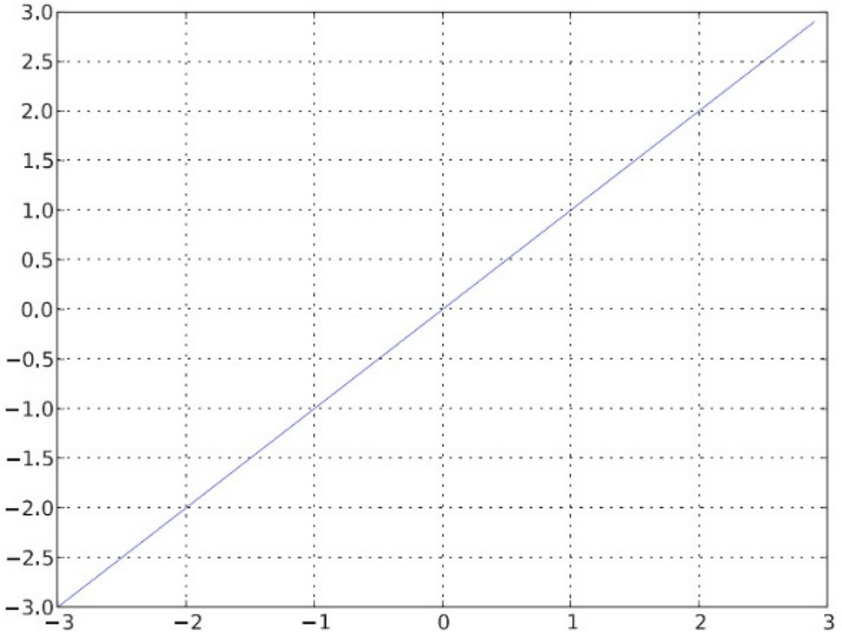
link.medium.com/d2zM1iep71

Neural Network Characterization

- A neural network is characterized by
  - its pattern of connections between the neurons (called its ***architecture***),
  - its ***activation function*** , and
  - its method of determining the weights on the connections (called its ***training***, or *learning,* ***algorithm***)
    - This Includes determining many ***hyperparameters*** such as learning rate, epochs, and batch size.

Activation Functions

- **Linear activation function**
  - Performs no input squashing
  - Not very interesting...
  - There is no benefit to depth in a multilayer NN with a linear activation function as the whole model becomes linear.

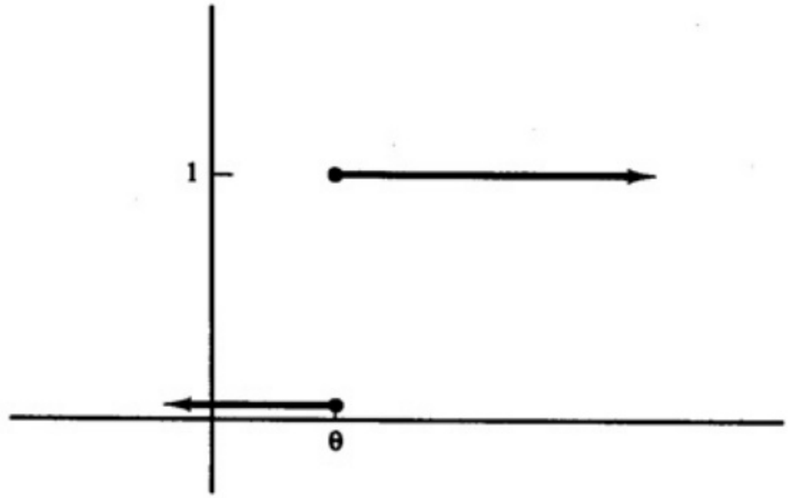


$$g(a) = a$$

Slide Credit: Hugo Larochelle

Activation Functions

- **Binary step function**:
  - A binary step function is a threshold-based activation function.
  - In early model, single-layer nets use a step function to convert the net input, which is a continuously valued variable, to an output unit that is a binary (l or 0) or bipolar (l or -1)

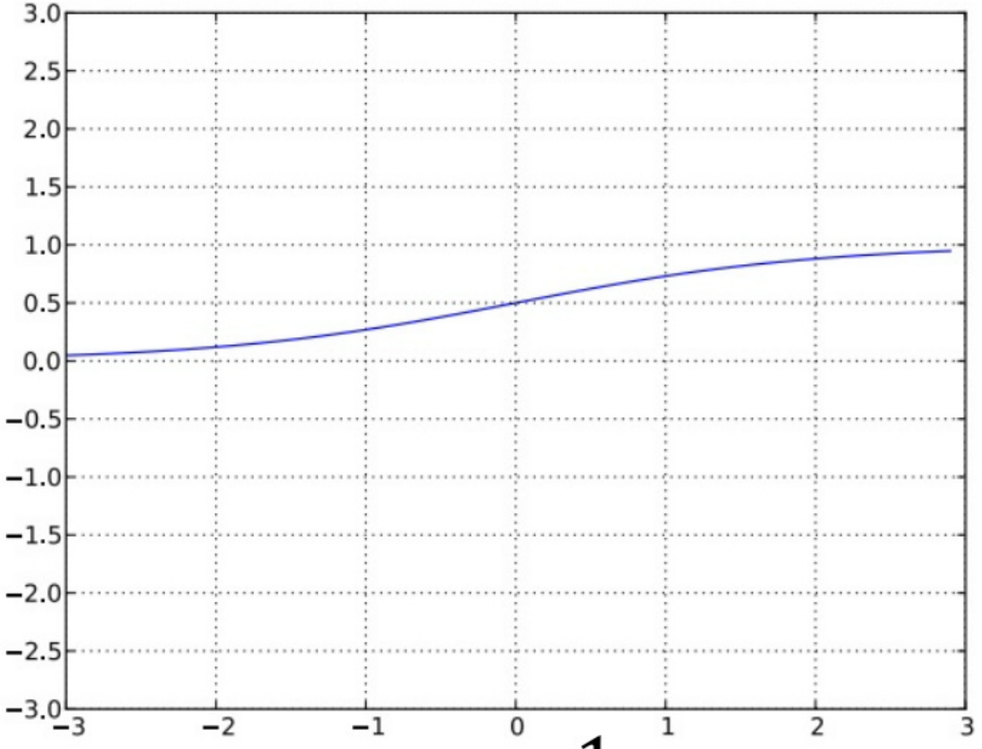


$$g(a) = \begin{cases} 1 & \text{if } a \geq \theta \\ 0 & \text{if } a < \theta \end{cases}$$

Activation Functions

- **Sigmoid activation function**
  - Squashes the neuron's pre-activation between 0 and 1
  - Good for probabilistic interpretations
  - Bounded
  - Strictly increasing

- Disadvantages
  - **Vanishing gradient**—for very high or very low values of a, there is almost no change to the prediction, causing a vanishing gradient problem.
  - Computationally expensive



$$g(a) = \frac{1}{1 + \exp(-a)}$$

$$g'(a) = g(a)(1 - g(a))$$

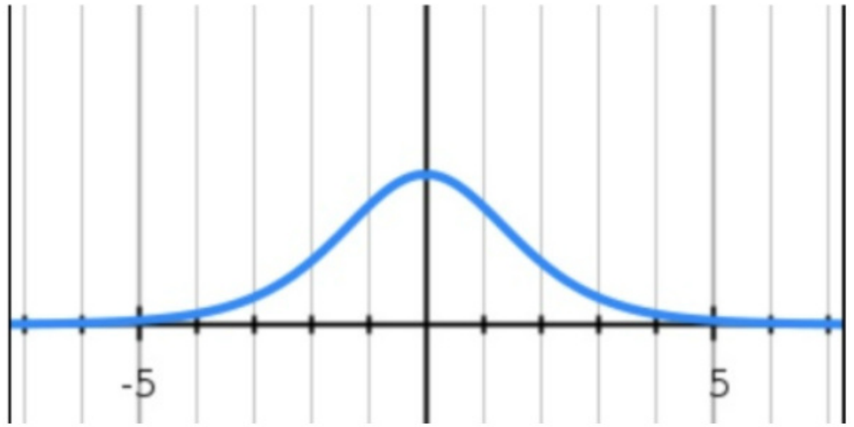
Slide Credit: Hugo Larochelle

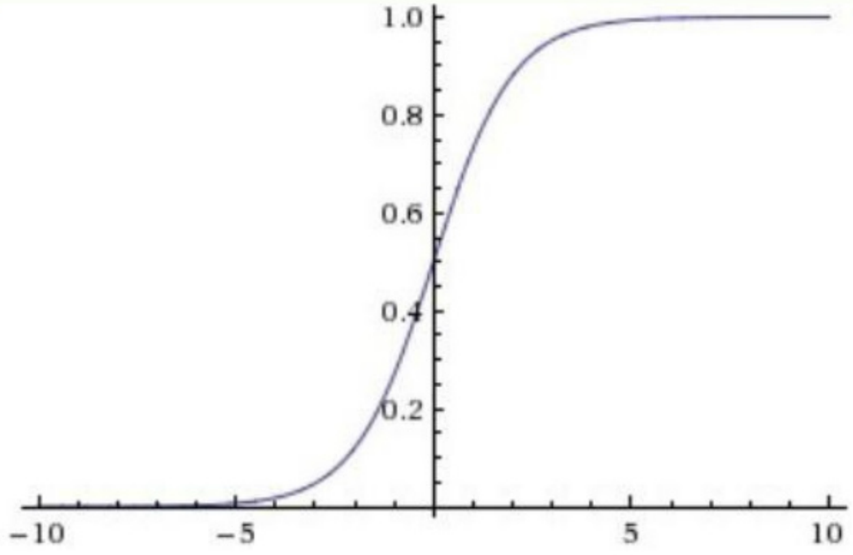
Activation Functions

- **Sigmoid activation function**
  - What happens when x = 10?
  - What happens when x = 0?
  - What happens when x = -10?

$$\sigma(x) = \sim 1 \qquad \frac{\partial \sigma(x)}{\partial x} = \sigma(x)\,(1 - \sigma(x)\,) = 1(1-1) = 0$$

$$\sigma(x) = \sim 0 \qquad \frac{\partial \sigma(x)}{\partial x} = \sigma(x)\,(1 - \sigma(x)\,) = 0(1-0) = 0$$





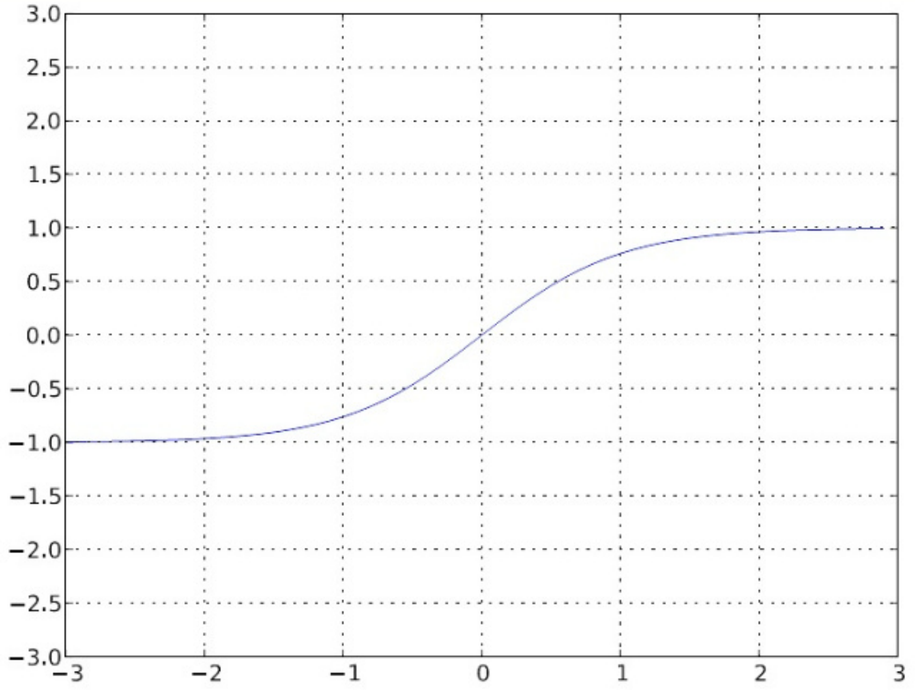
$$g'(a) = g(a)(1 - g(a))$$

Why is this a problem?

If all the gradients flowing back will be zero and weights will never change

Activation Functions

- **Hyperbolic tangent** (“tanh”) activation function
  - Squashes the neuron’s pre-activation between -1 and 1
  - Zero-centered output, often leads to faster convergence than Sigmoid.
  - Bounded, Strictly increasing
  - Also saturates, causing vanishing gradients.



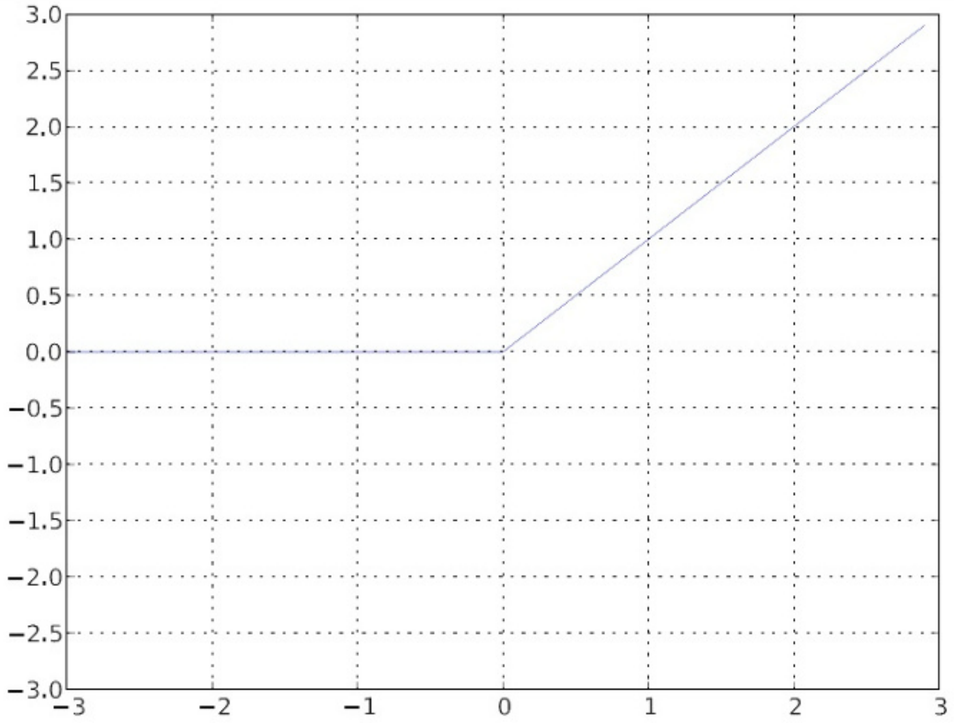
$$g(a) = \tanh(a) = \frac{\exp(a)-\exp(-a)}{\exp(a)+\exp(-a)} = \frac{\exp(2a)-1}{\exp(2a)+1}$$

$$g'(a) = (1 - g(a)^2)$$

Slide Credit: Hugo Larochelle

Activation Functions

- **Rectified linear (ReLU) activation function**
  - Bounded below by 0 (always non-negative), Not upper bounded
  - Tends to give neurons with sparse activities
  - Non-saturating for $x>0$, reducing vanishing gradient problems and often speeding up training.
  - Computationally efficient—allows the network to converge very quickly
- **The Dying ReLU problem**—when inputs approach zero, or are negative, the gradient of the function becomes zero, the network cannot perform backpropagation and cannot learn.
  - Leaky ReLU, Parametric ReLU



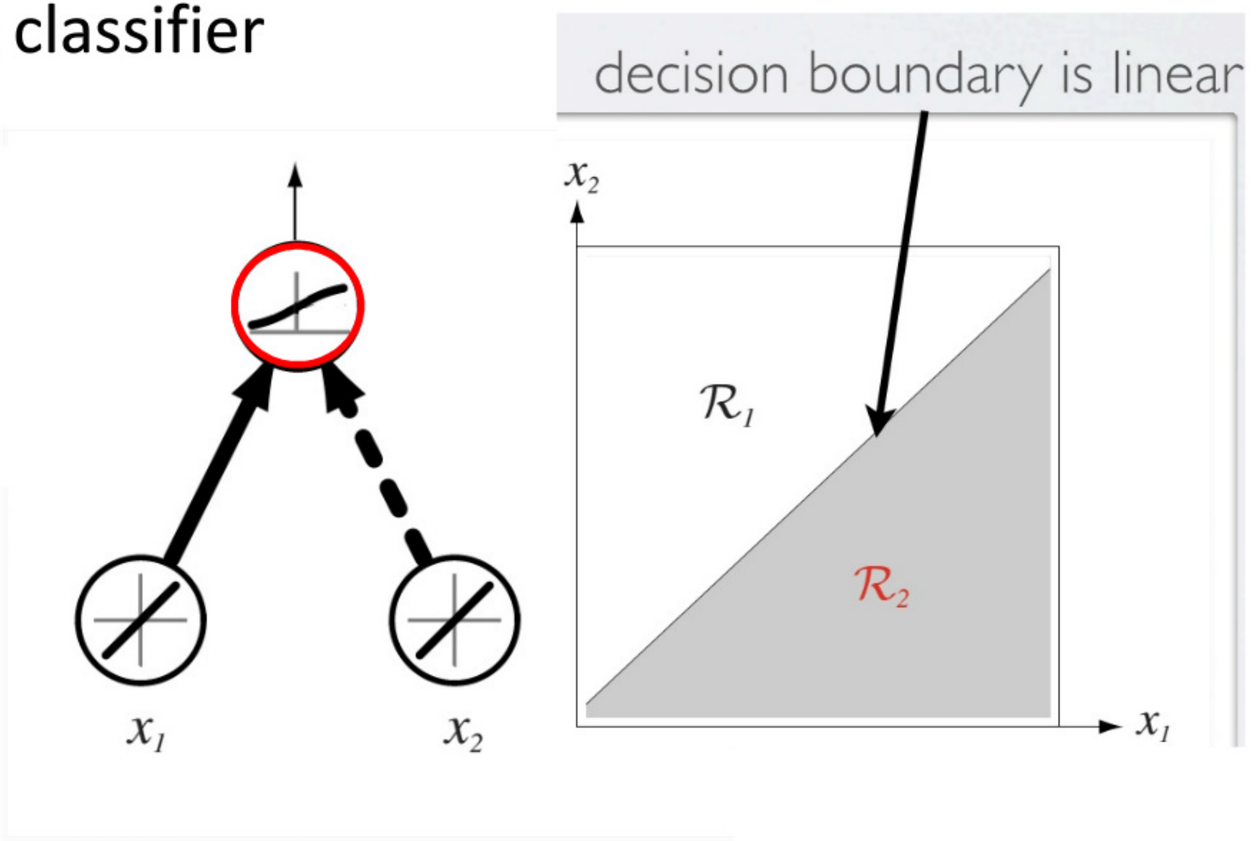
$$g(a) = \text{reclin}(a) = \max(0, a)$$

$$g'(a) = 1, \text{if a} > 0$$

Slide Credit: Hugo Larochelle

Capacity of Single Neuron

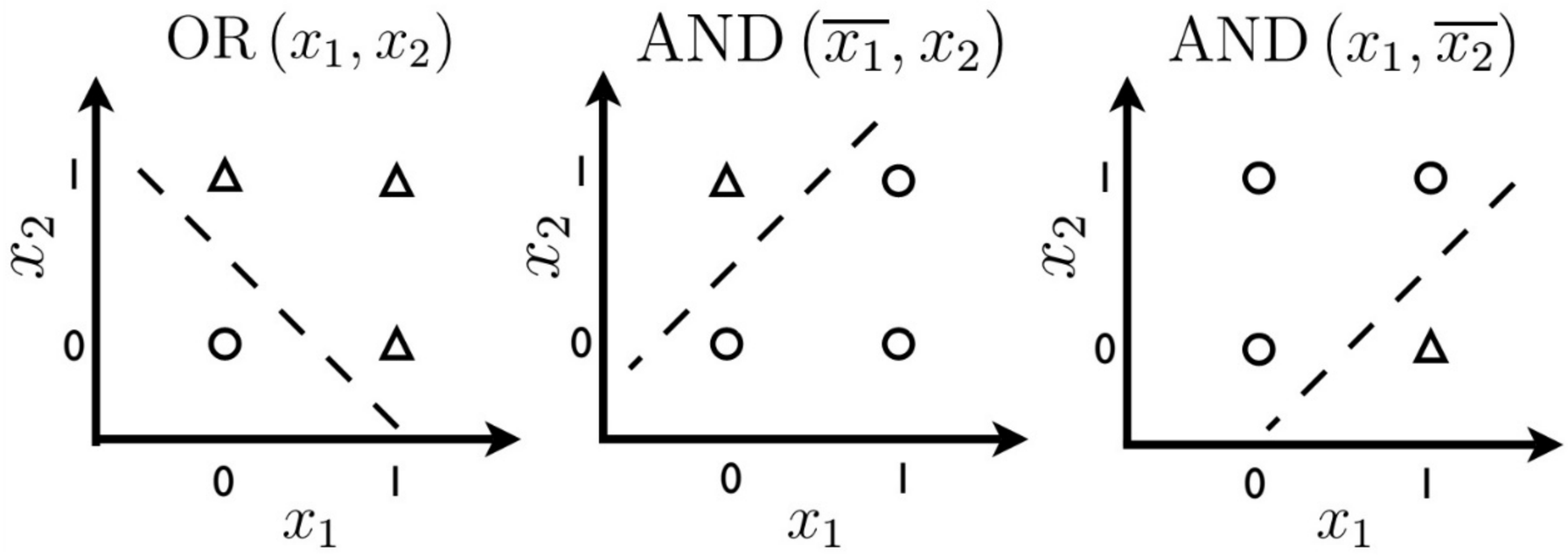
- Could do binary classification:
  - with sigmoid, can interpret neuron as estimating $p(y = 1 / \mathbf{x})$
  - also known as logistic regression classifier
    - if greater than 0.5, predict class 1
    - otherwise, predict class 0



Slide Credit: Hugo Larochelle

Capacity of Single Neuron

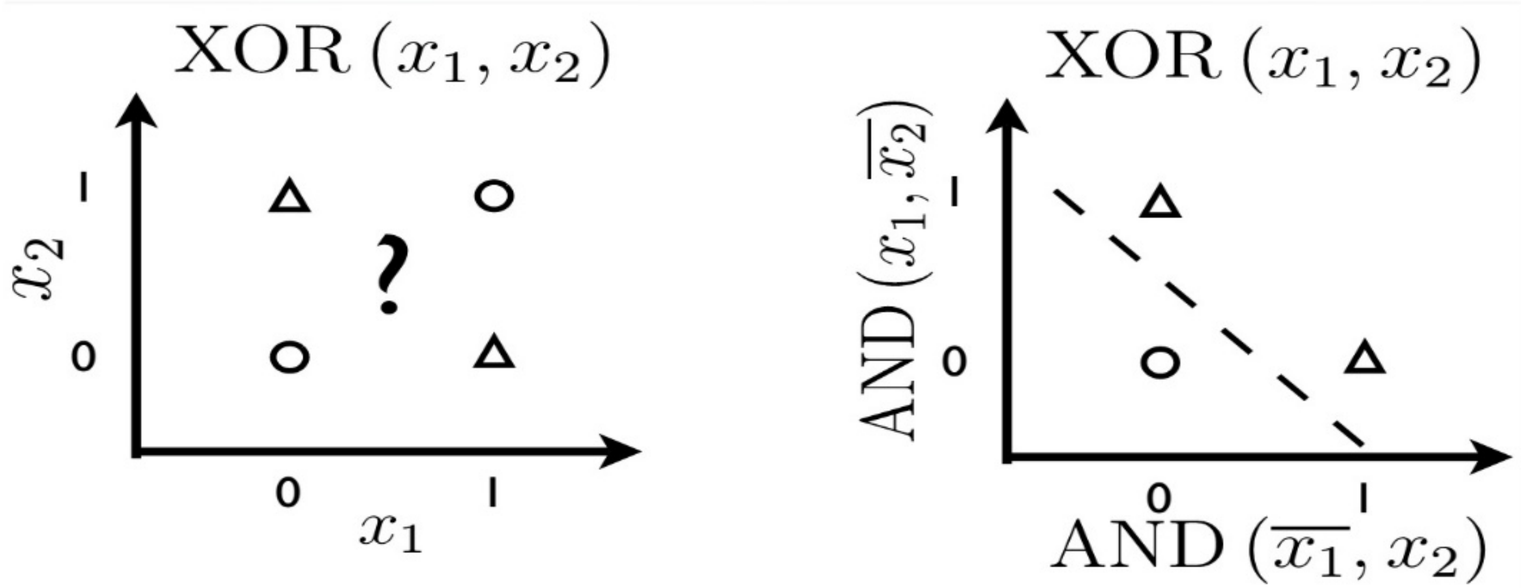
- Can solve linearly separable problems



Slide Credit: Hugo Larochelle

Capacity of Single Neuron

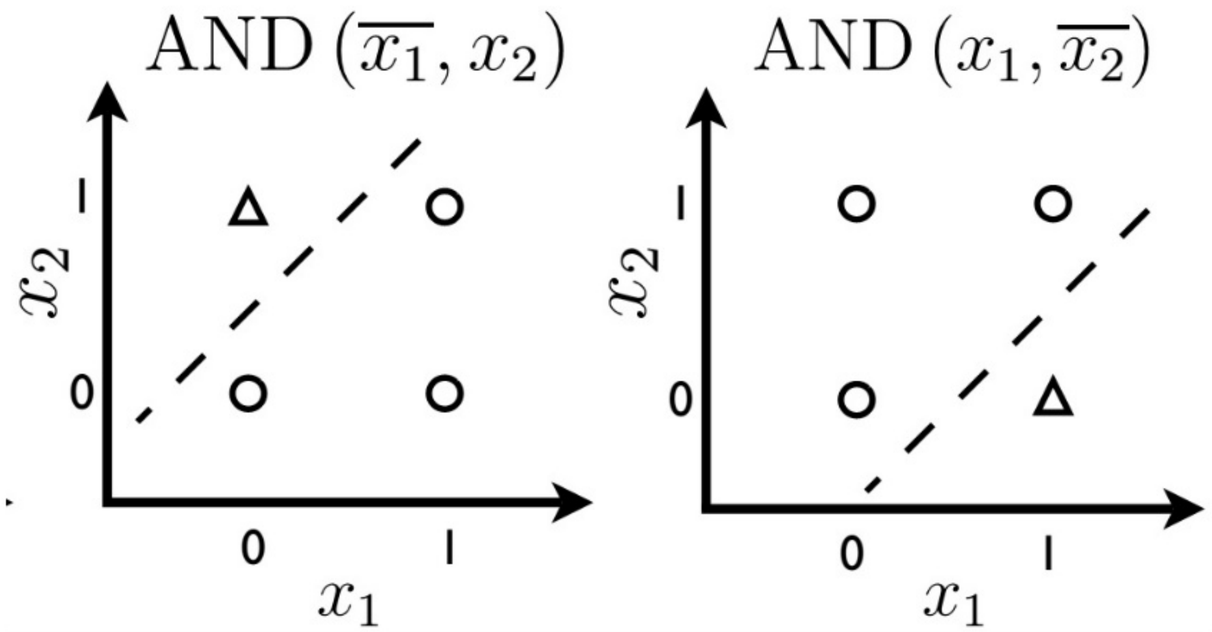
- Can’t solve non linearly separable problems...

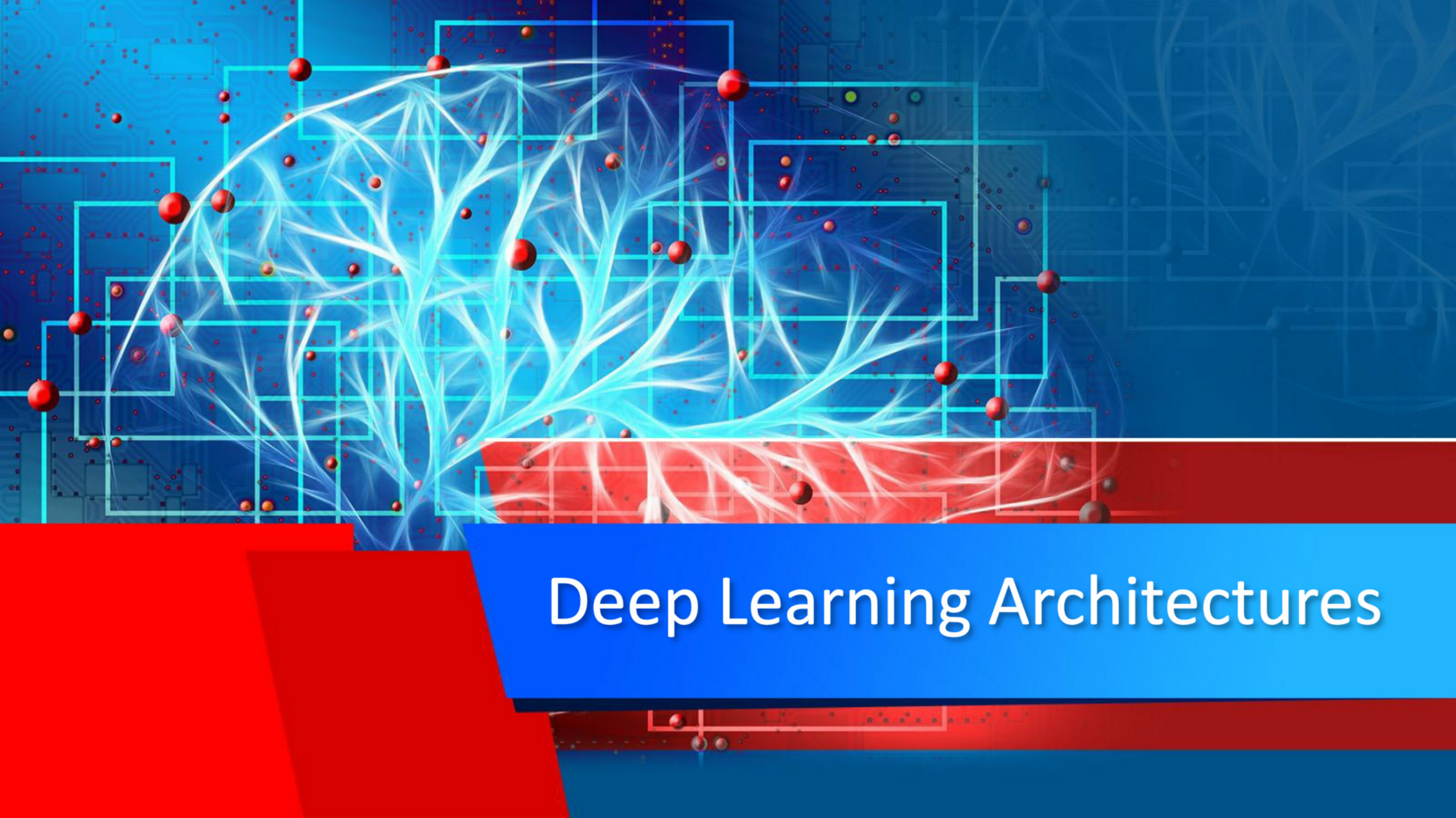


- ... unless the input is transformed in a better representation

Slide Credit: Hugo Larochelle

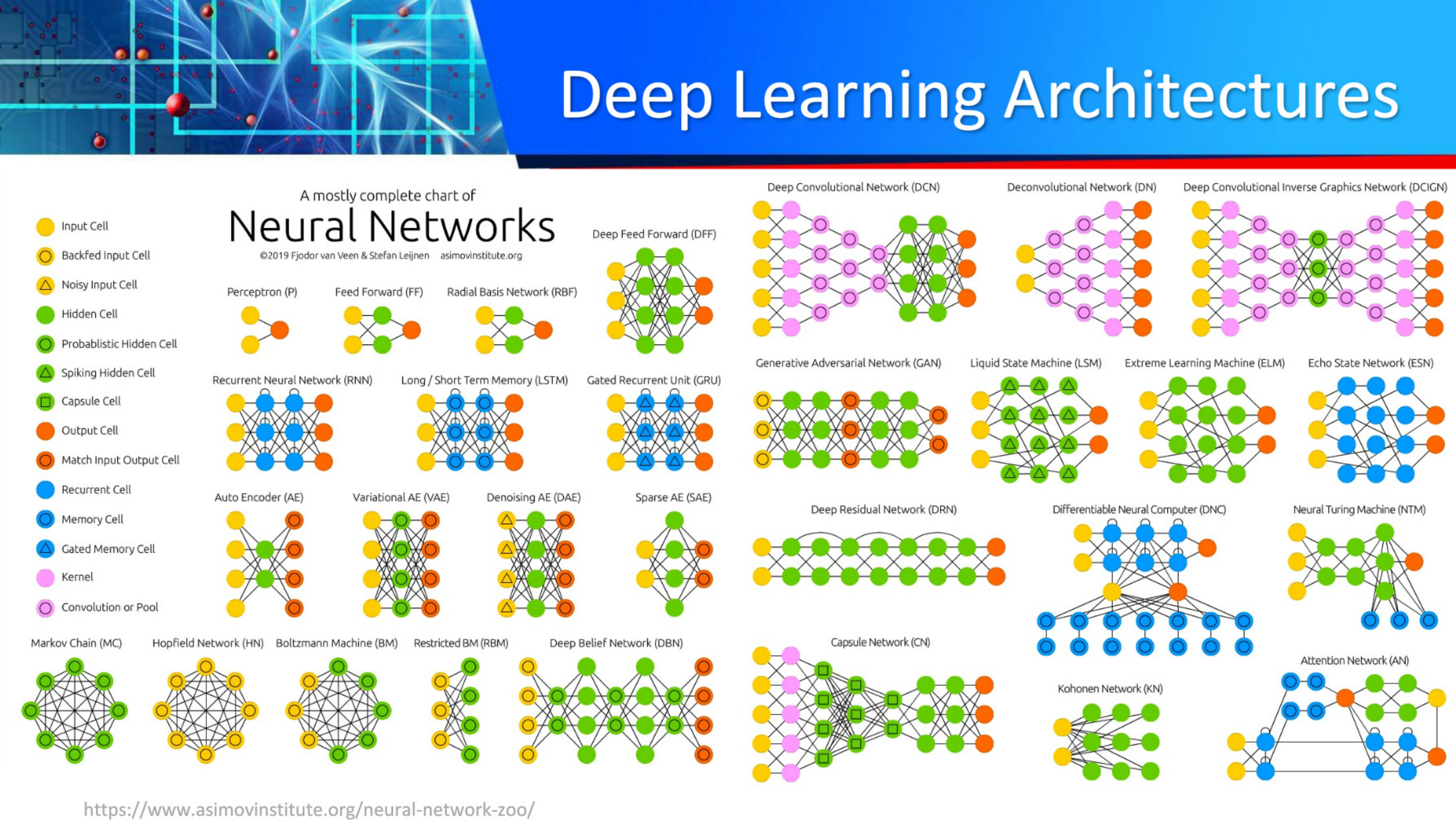
x1	x2	$x1^-$	$x2^-$	AND ($x1^-$,x2)	AND ($x2^-$,x1)
0	0	1	1	0	0
1	0	0	1	0	1
0	1	1	0	1	0
1	1	0	0	0	0





Deep Learning Architectures

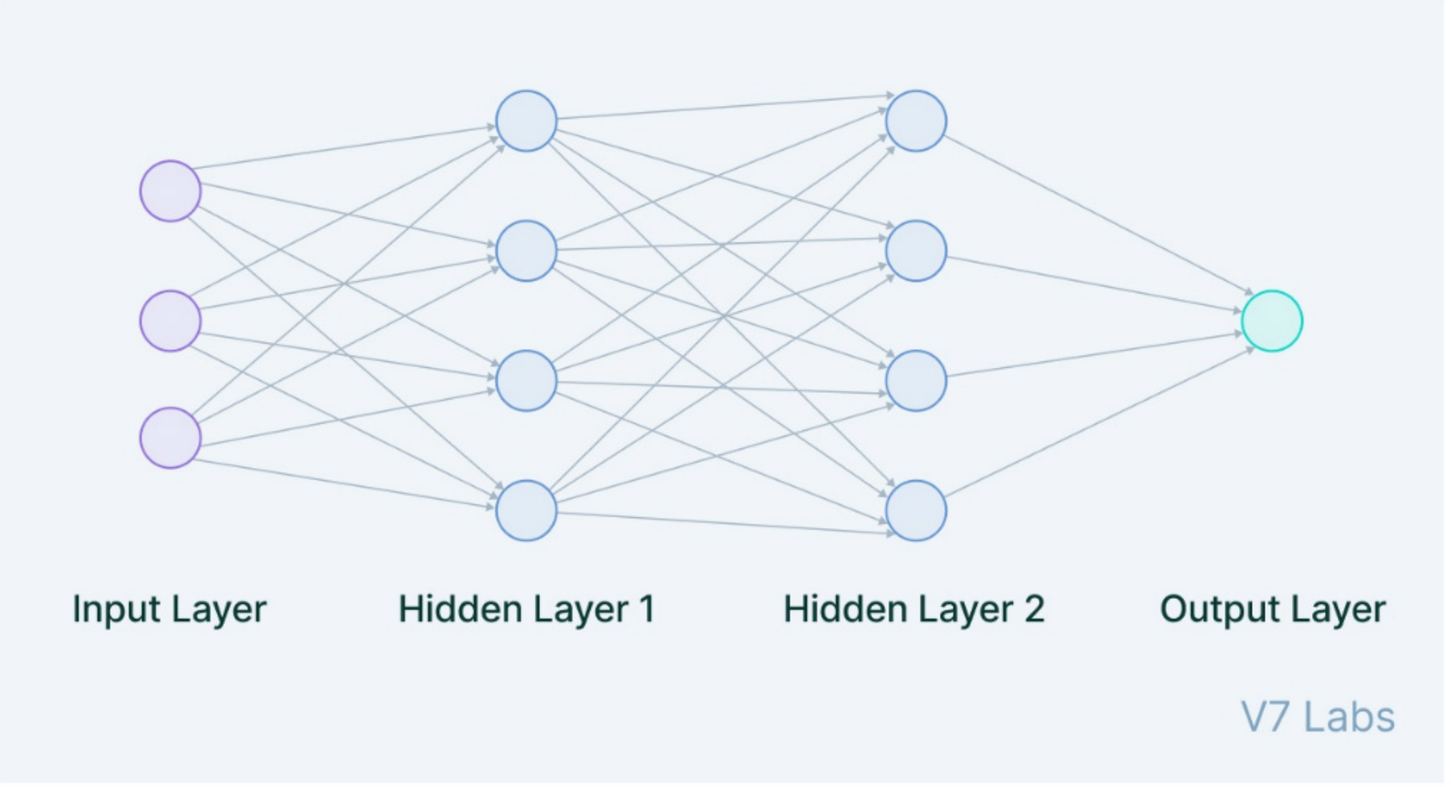
Deep Learning Architectures



https://www.asimovinstitute.org/neural-network-zoo/

Multi-layer Neural Network

- This is a feedforward neural Network.
- The input units are fully connected to output units.
- Has one or more hidden layers of nodes.
- Can solve more complicated problems than single-layer nets, but training may be more difficult.



Single-hidden Layer Networks

- Hidden layer pre-activation:

$$\mathbf{a}(\mathbf{x}) = \mathbf{b}^{(1)} + \mathbf{W}^{(1)}\mathbf{x}$$

$$\left(a(\mathbf{x})_i = b_i^{(1)} + \sum_j W_{i,j}^{(1)} x_j\right)$$

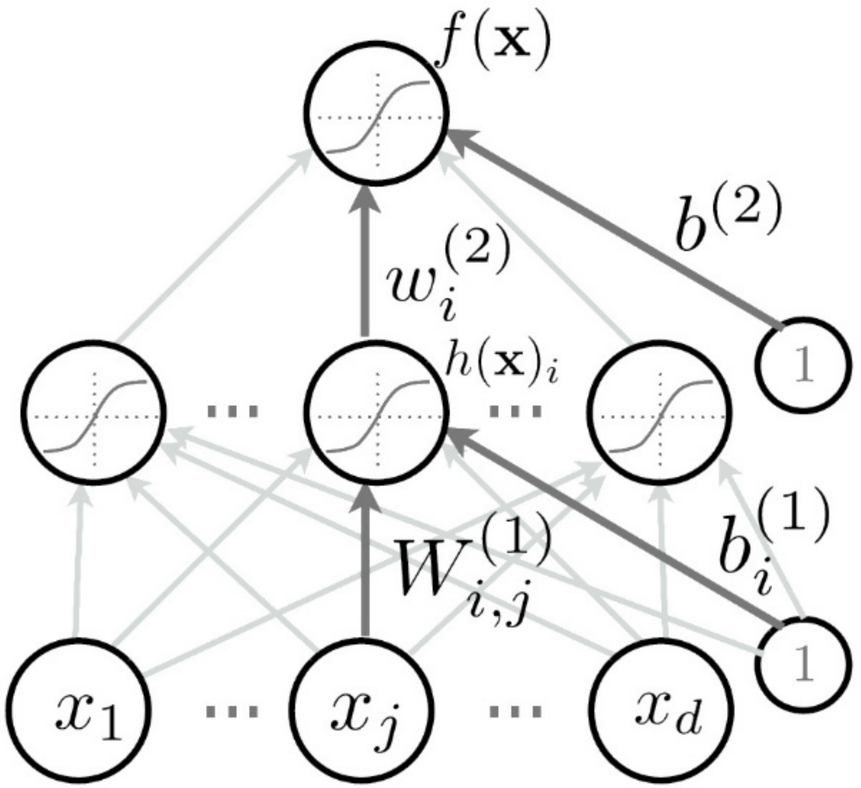
- Hidden layer activation:

$$\mathbf{h}(\mathbf{x}) = \mathbf{g}(\mathbf{a}(\mathbf{x}))$$

- Output layer activation:

$$f(\mathbf{x}) = o\left(b^{(2)} + \mathbf{w}^{(2)\top}\mathbf{h}^{(1)}\mathbf{x}\right)$$

output activation function



Slide Credit: Hugo Larochelle

Multilayer Neural Network

- Could have L hidden layers:
  - layer pre-activation for $k>0$ $(\mathbf{h}^{(0)}(\mathbf{x}) = \mathbf{x})$

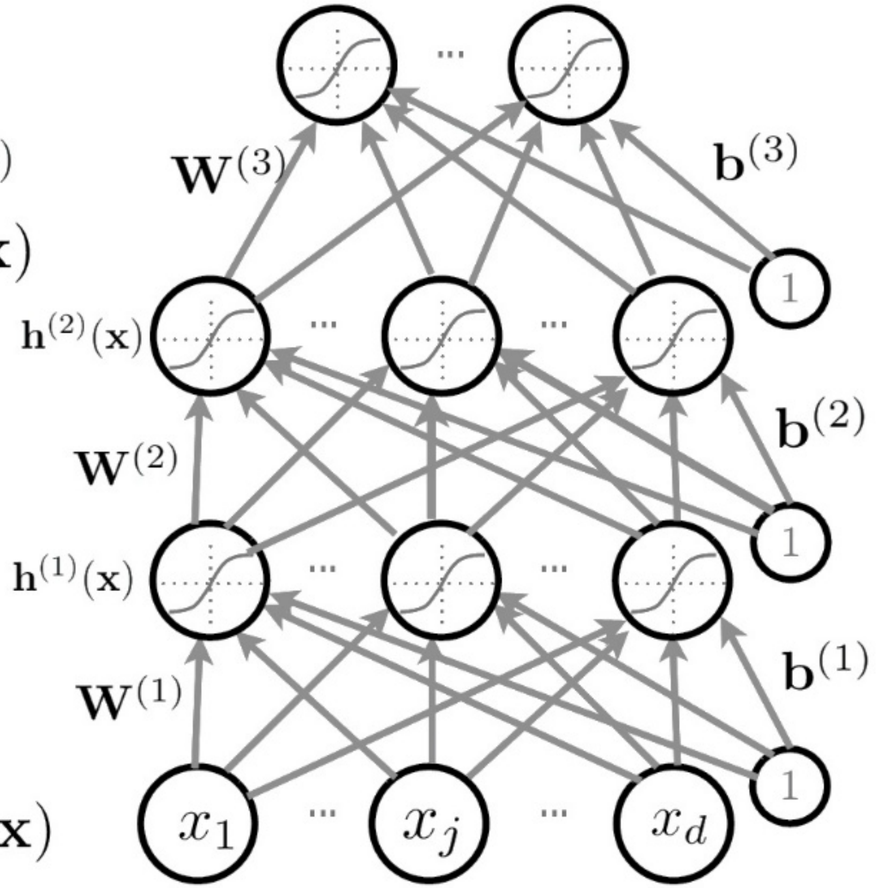
$$\mathbf{a}^{(k)}(\mathbf{x}) = \mathbf{b}^{(k)} + \mathbf{W}^{(k)}\mathbf{h}^{(k-1)}(\mathbf{x})$$

  - hidden layer activation (k from 1 to L):

$$\mathbf{h}^{(k)}(\mathbf{x}) = \mathbf{g}(\mathbf{a}^{(k)}(\mathbf{x}))$$

  - output layer activation ($k=L+1$):

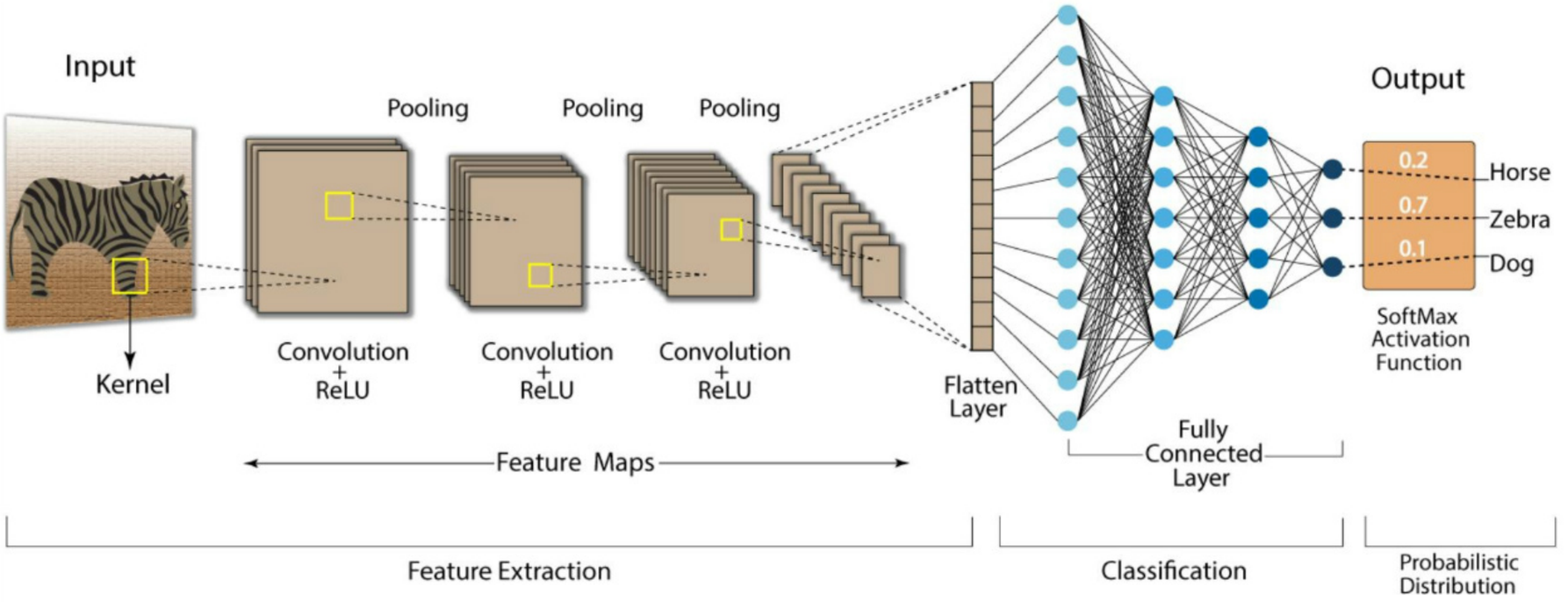
$$\mathbf{h}^{(L+1)}(\mathbf{x}) = \mathbf{o}(\mathbf{a}^{(L+1)}(\mathbf{x})) = \mathbf{f}(\mathbf{x})$$



Slide Credit: Hugo Larochelle

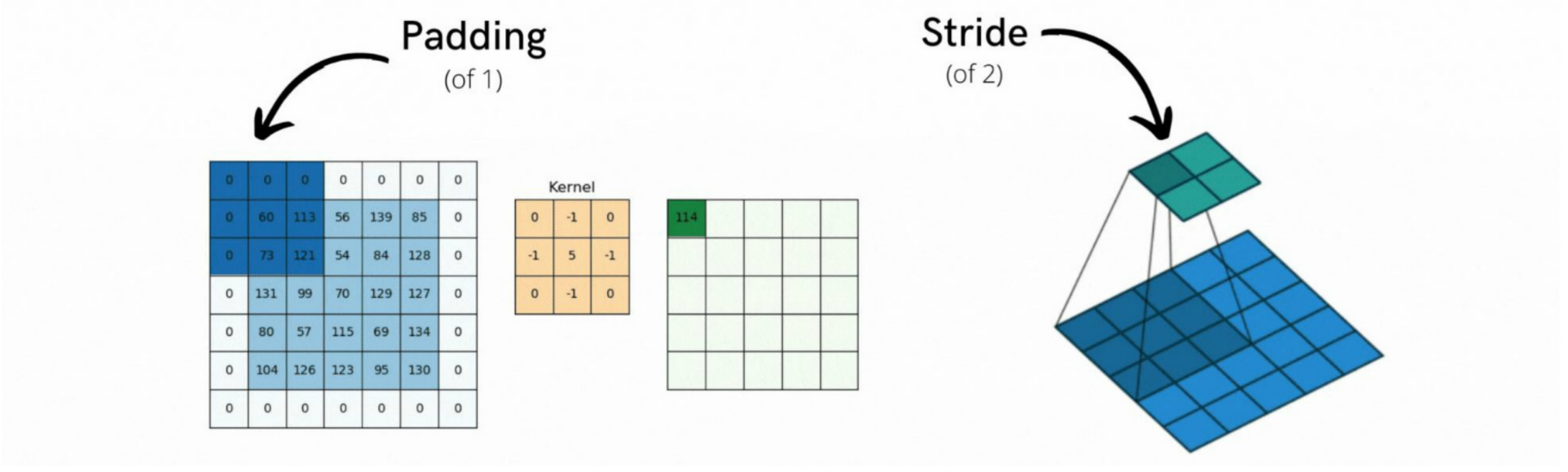
Convolutional Neural Networks (CNNs)

- CNN is a multilayer feedforward that use minimal preprocessing. This means that the network learns features that in traditional algorithms were hand-engineered.
- Their wide applications is in image and video recognition, and natural language processing.
- It has the ability to automatically and adaptively learn spatial hierarchies of features from input images.



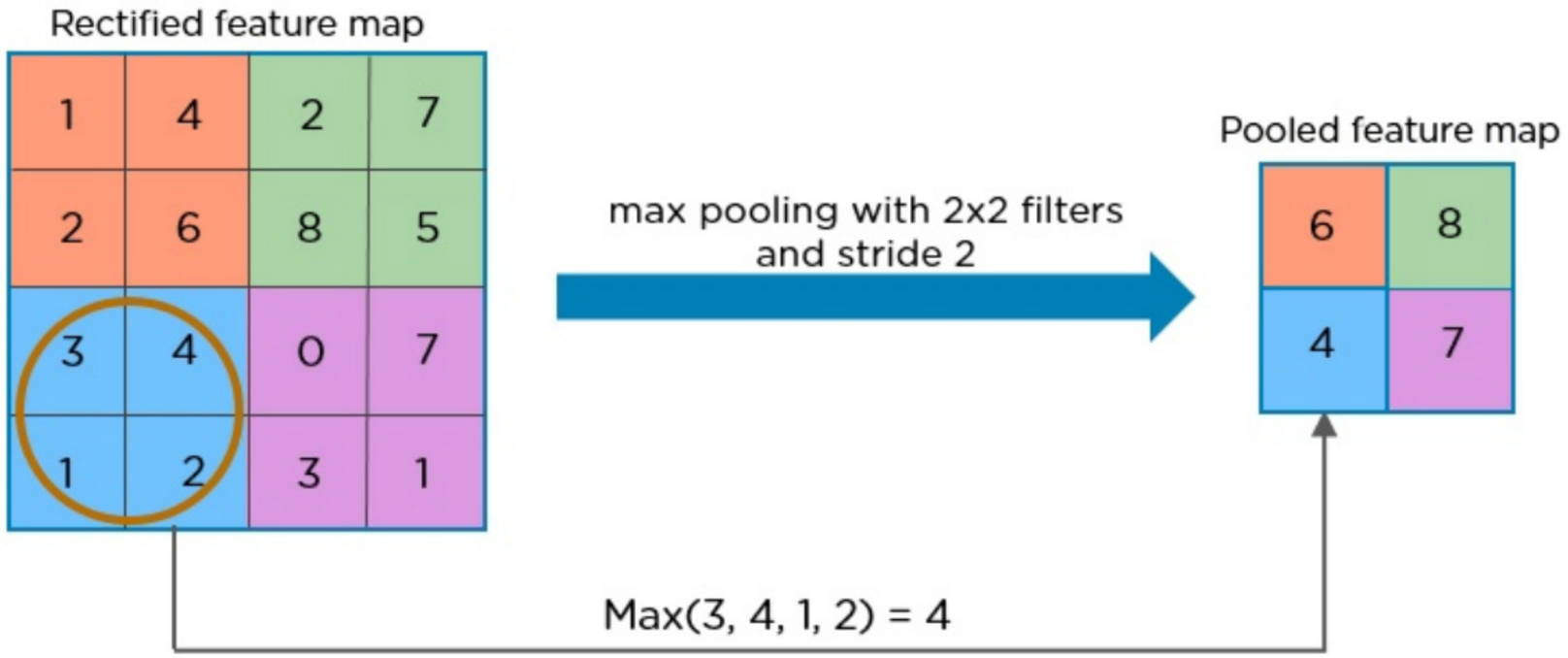
Convolution layer

- A convolution layer has **several filters** that perform the convolution operation.
- It performs convolution operations as it is scanning the input II with respect to its dimensions.
- Its hyperparameters include the **filter size F, stride S, and Padding P.**

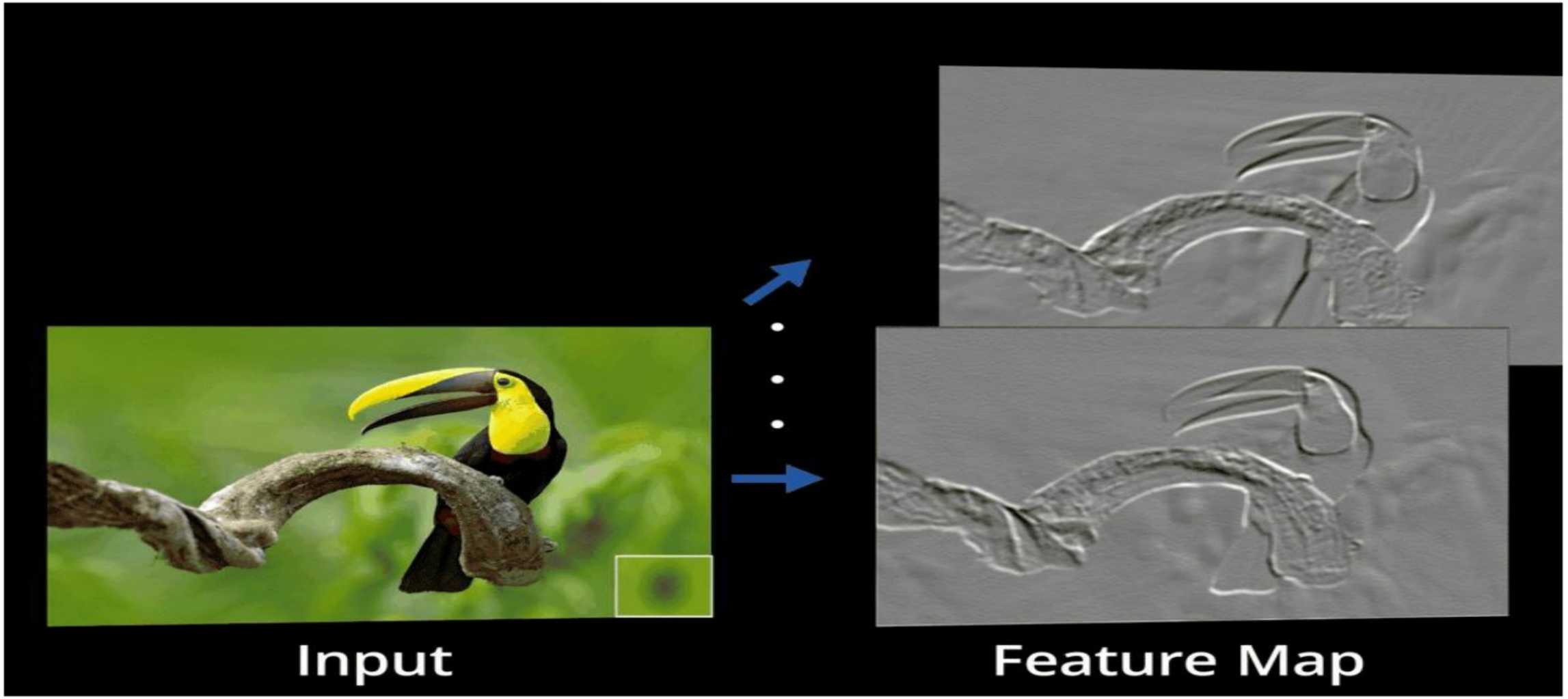


Pooling layer

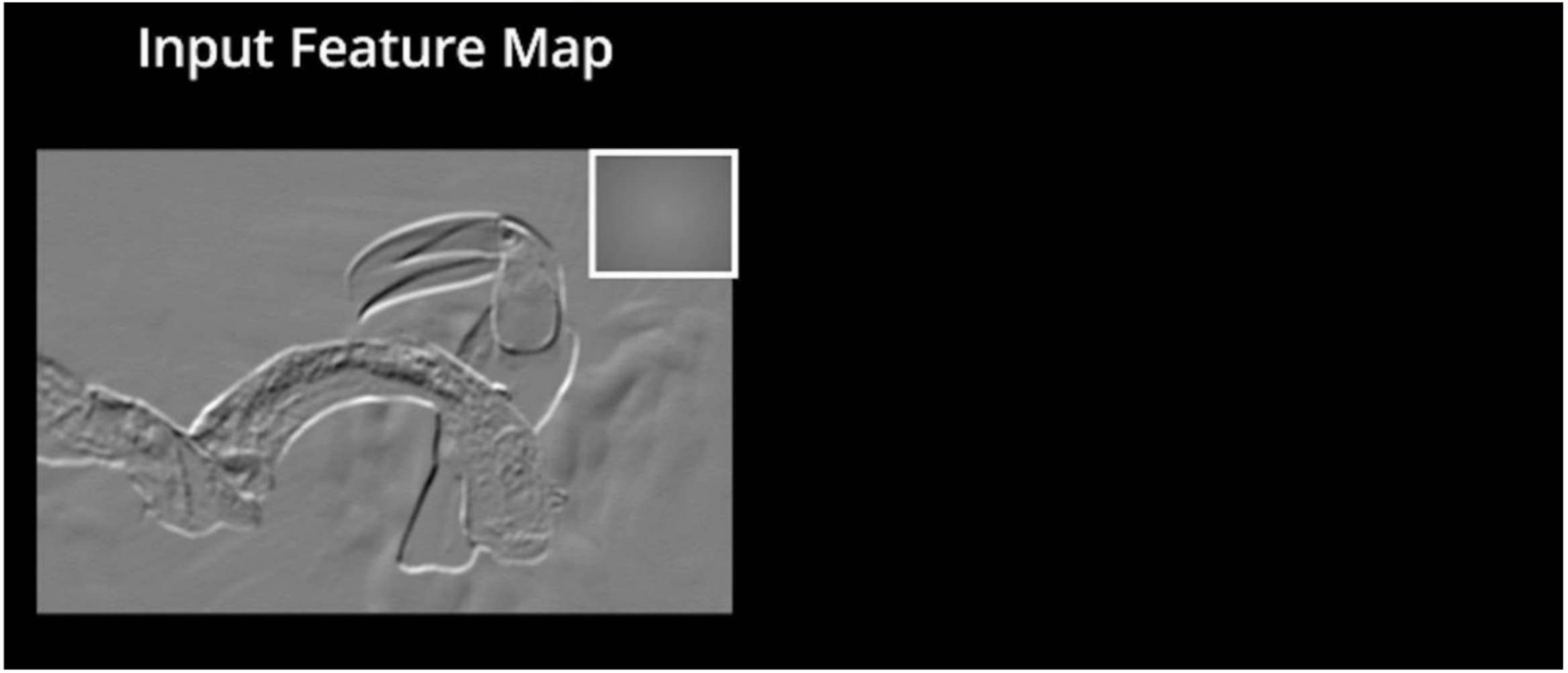
- This layer is a down-sampling operation that reduces the dimensionality of the feature maps while retaining the most essential information.
- Common types include max pooling and average pooling.



Example

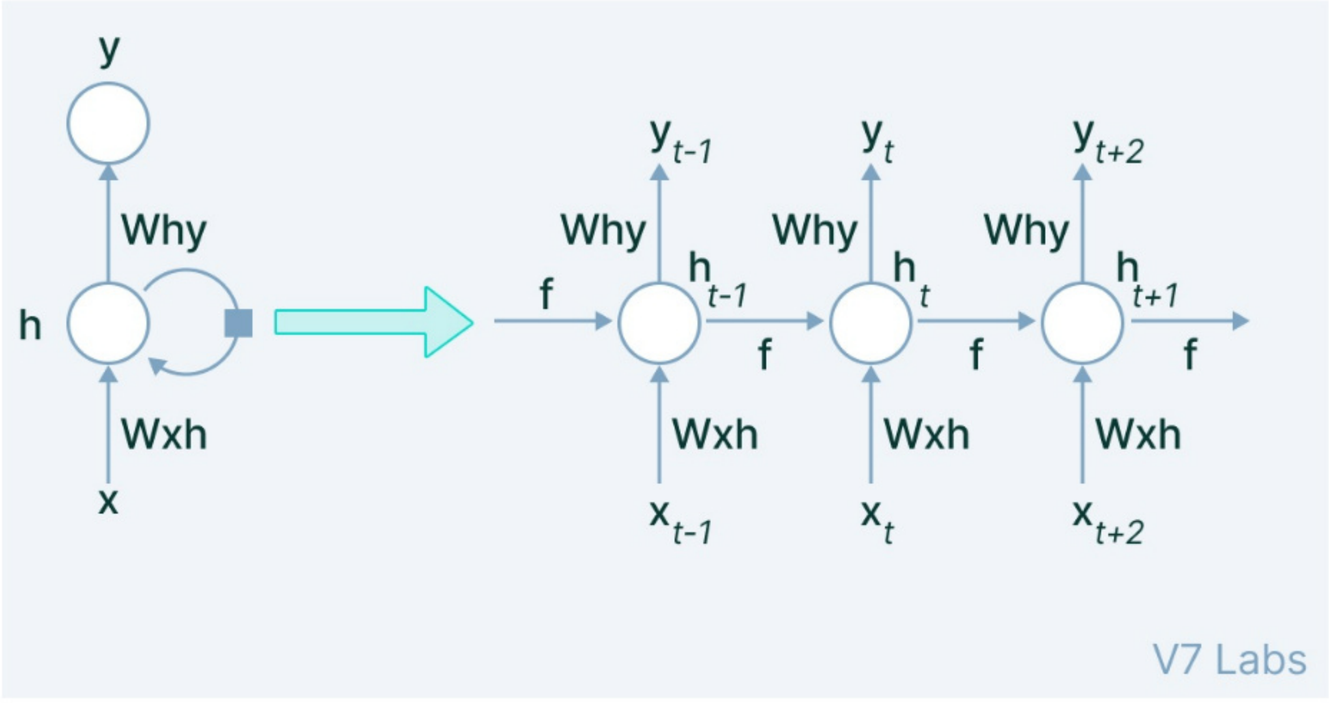


Example



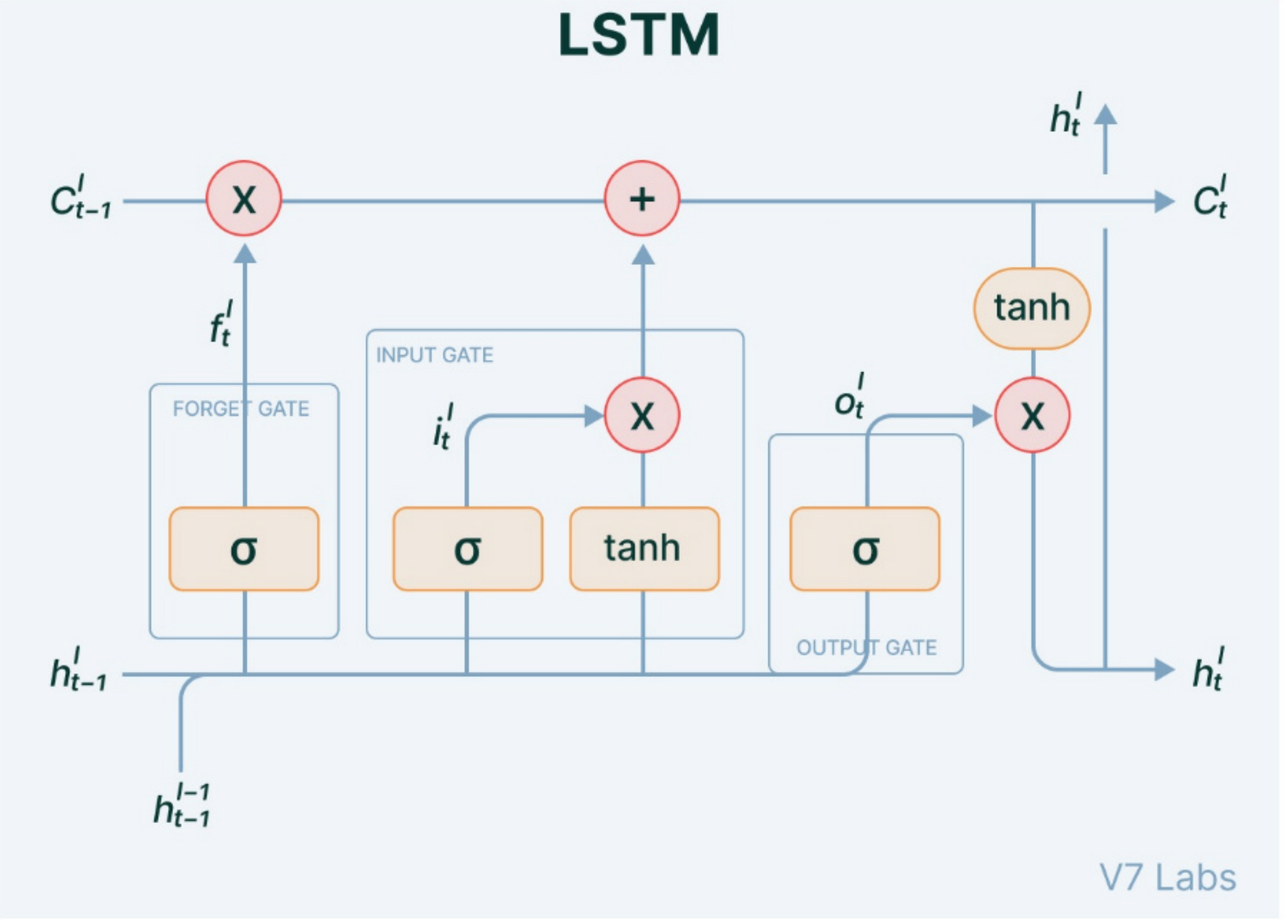
Recurrent Neural Networks (RNNs)

- It has at least one feedback loop.
- It maintain a hidden state that serves as a memory, capturing information about previous inputs in a sequence allowing them to capture temporal dependencies.
- Well-suited for speech recognition, language modeling, and time-series prediction problems.
- However, traditional RNNs have limitations in capturing long-term dependencies.



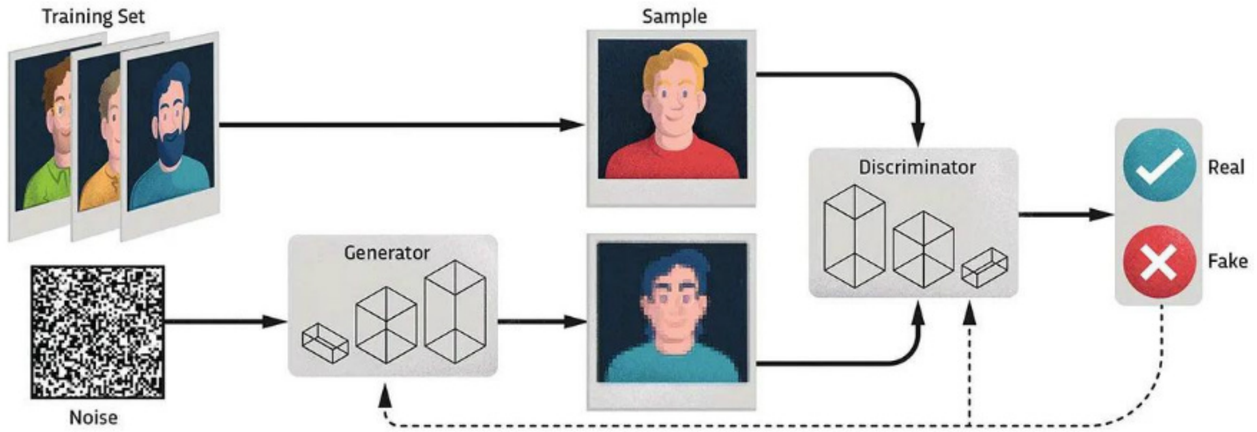
Long Short-Term Memory (LSTM)

- LSTMs are a special kind of RNN capable of learning long-term dependencies.
- **Cell State**: LSTMs have a cell state that runs through the entire sequence and can carry information across many steps.
- **Gates**: control the flow of information:
  - Input Gate: Determines which information from the current input should be updated in the cell state.
  - Forget Gate: Decides what information should be discarded from the cell state.
  - Output Gate: Controls the information that should be outputted based on the cell state.



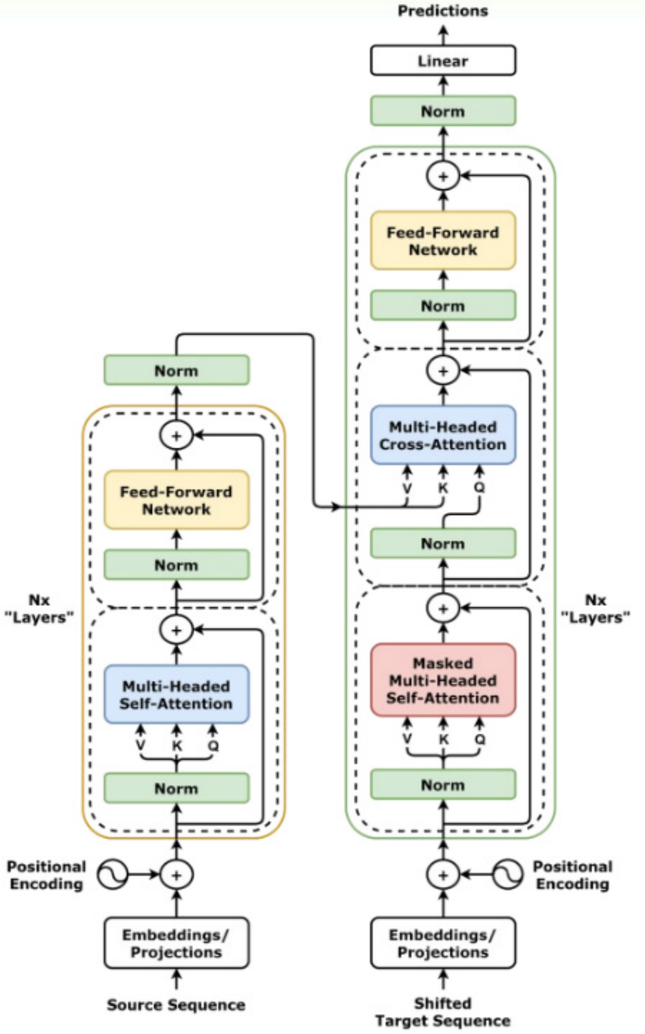
Generative Adversarial Networks (GANs)

- GANs generate new synthetic data by training two neural networks in a competitive setting.
- They have been used to create realistic images, videos, and audio.
  - **Generator Network**: Creates fake data from random noise.
  - **Discriminator Network**: Evaluates the authenticity of the data, distinguishing between real and fake data.
  - Training Process: The generator tries to fool the discriminator by producing better fake data, while the discriminator tries to get better at detecting counterfeit data.

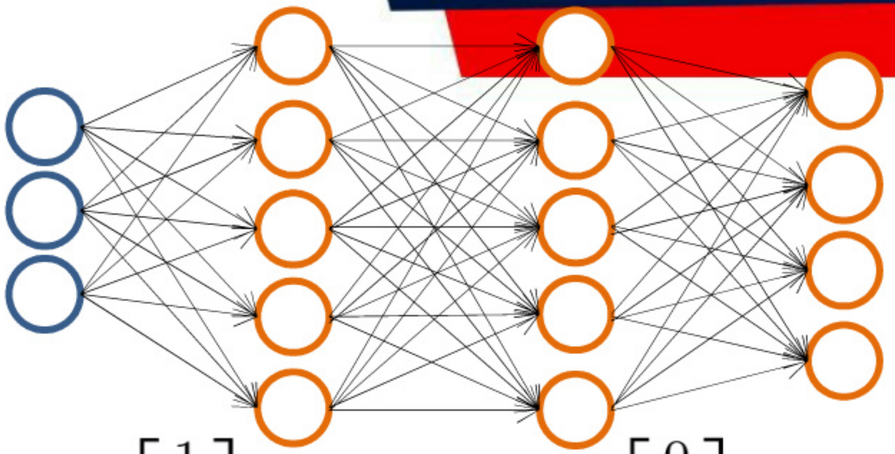


Transformer Networks

- Transformers are the backbone of many modern NLP models.
- They process input data using self-attention, allowing for parallelization and improved handling of long-range dependencies.
  - **Self-Attention Mechanism**: This mechanism computes the importance of each part of the input relative to every other part, enabling the model to weigh the significance of different words in a sentence differently.
  - **Encoder-Decoder Architecture**: Consists of an encoder that processes the input sequence and a decoder that generates the output sequence. Each consists of multiple layers of self-attention and feed-forward networks.



Multiple output units: One-vs-all.



$$h_\Theta(x) \in \mathbb{R}^4$$

Want $h_\Theta(x) \approx \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, etc.

when pedestrian when car when motorcycle

Training set: $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$

$y^{(i)}$ one of $\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$

pedestrian car motorcycle truck

Slide Credit: Andrew Ng

Softmax Activation Function

- For multi-class classification:
  - we need multiple outputs (1 output per class)
  - we would like to estimate the conditional probability $p(y = c|\mathbf{x})$

$$\mathbf{o}(\mathbf{a}) = \text{softmax}(\mathbf{a}) = \left[\frac{\exp(a_1)}{\sum_c \exp(a_c)} \cdots \frac{\exp(a_C)}{\sum_c \exp(a_c)} \right]^\top$$

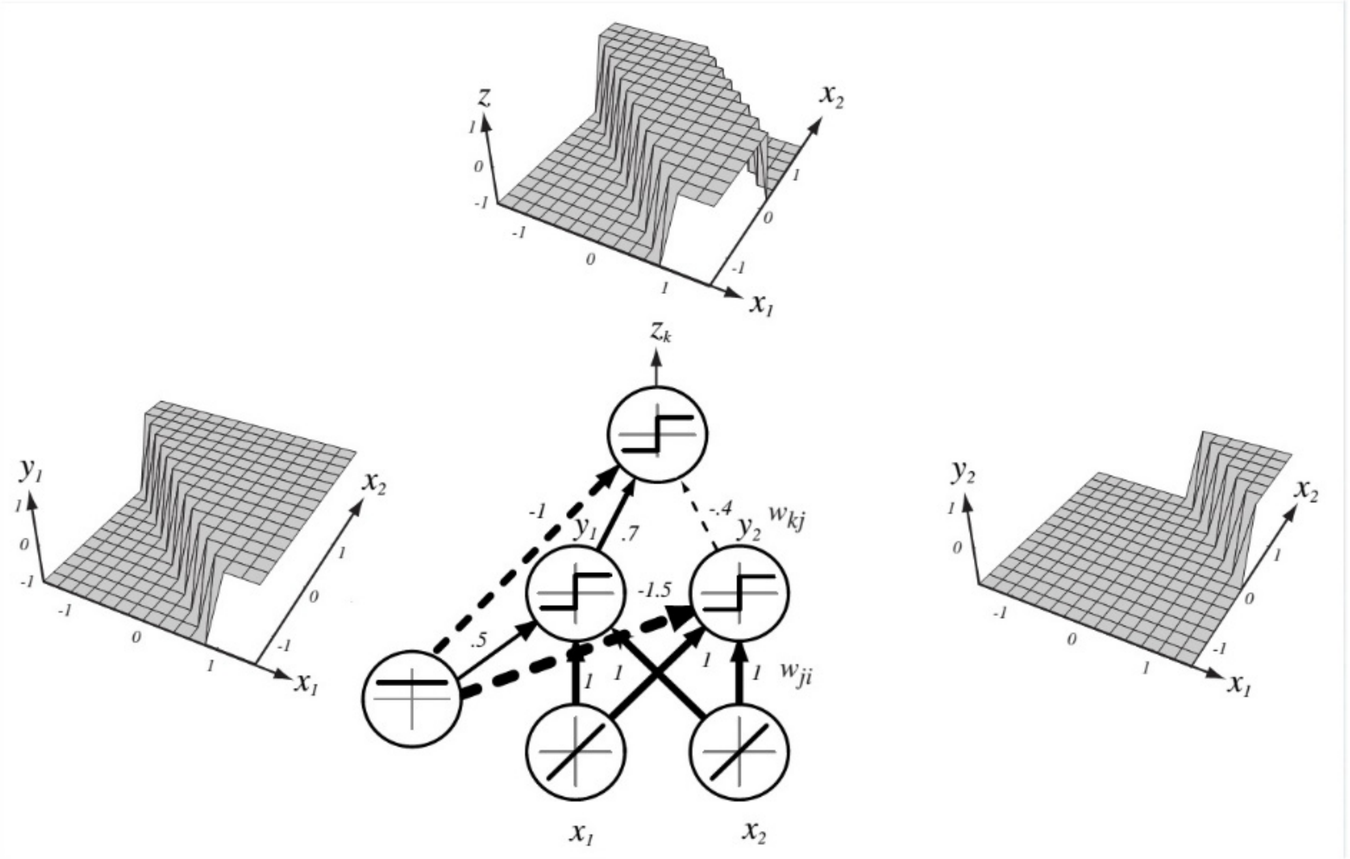
Vector of output neurons, where a_c is pre-activation of neuron c.

- We use the softmax activation function at the output:
  - strictly positive
  - sums to one
- Predicted class is the one with highest estimated probability

Slide Credit: Hugo Larochelle

Capacity of Neural Network

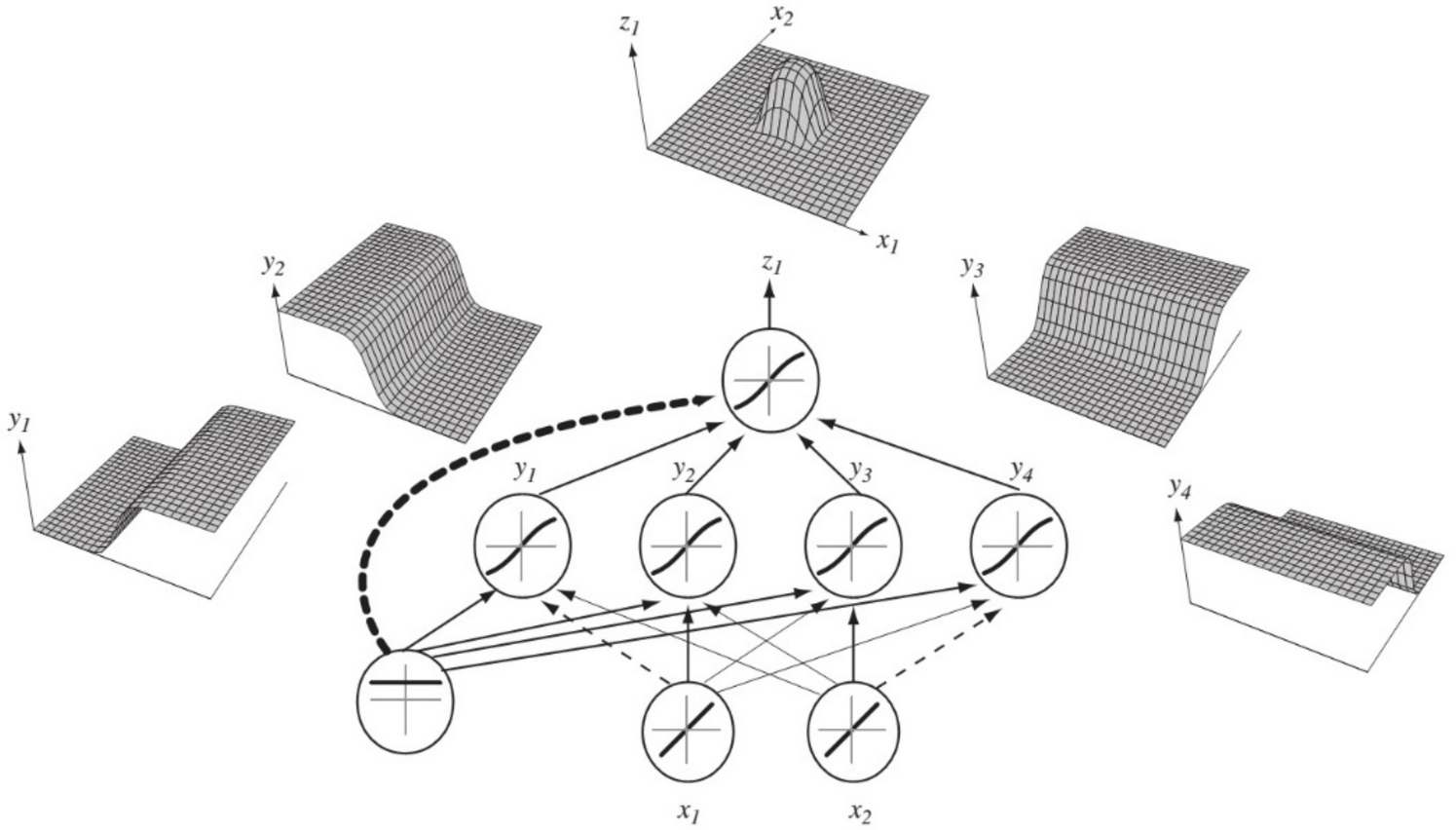
- single hidden layer neural network



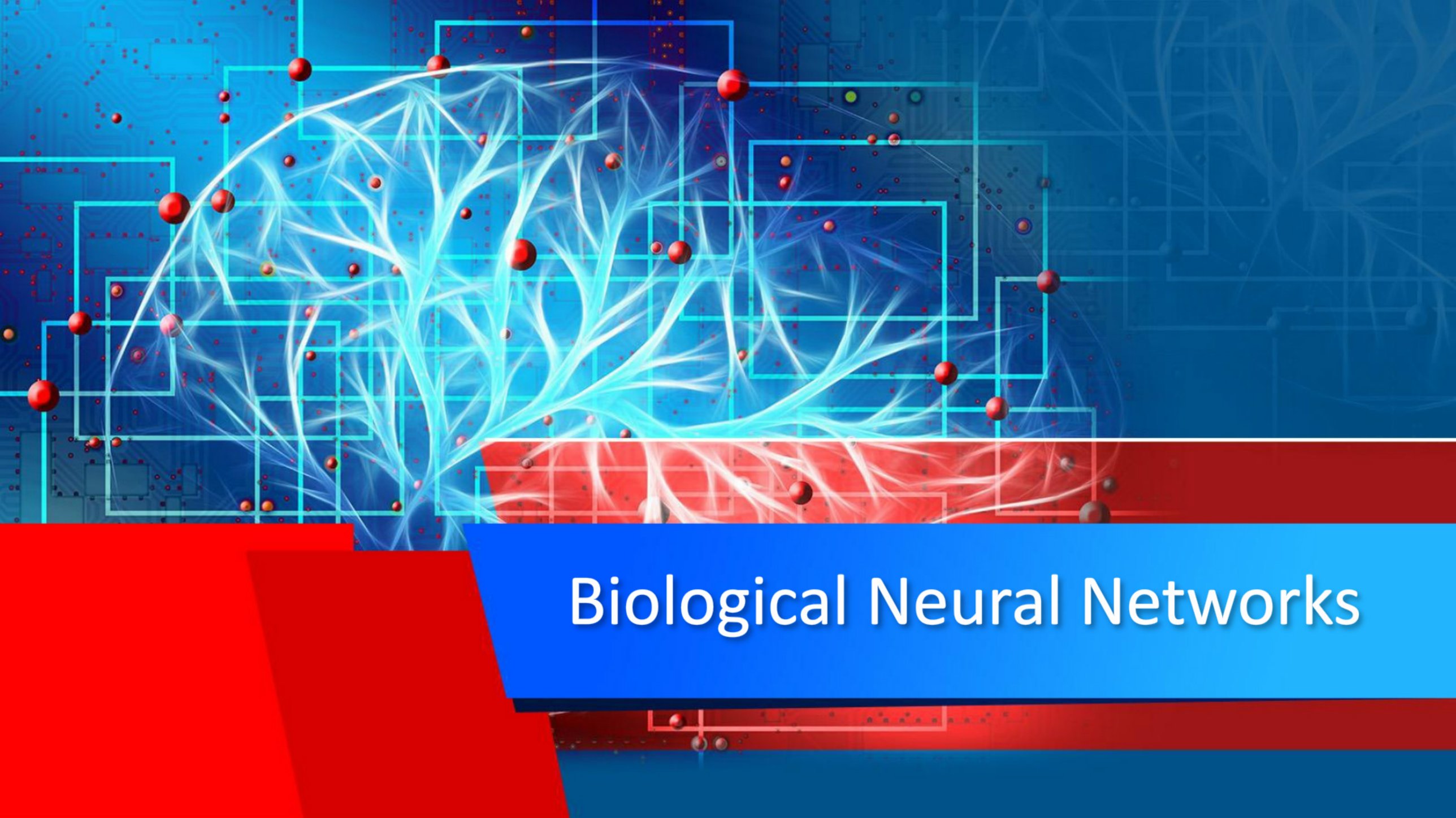
Slide Credit: Hugo Larochelle

Capacity of Neural Network

- Single hidden layer neural network



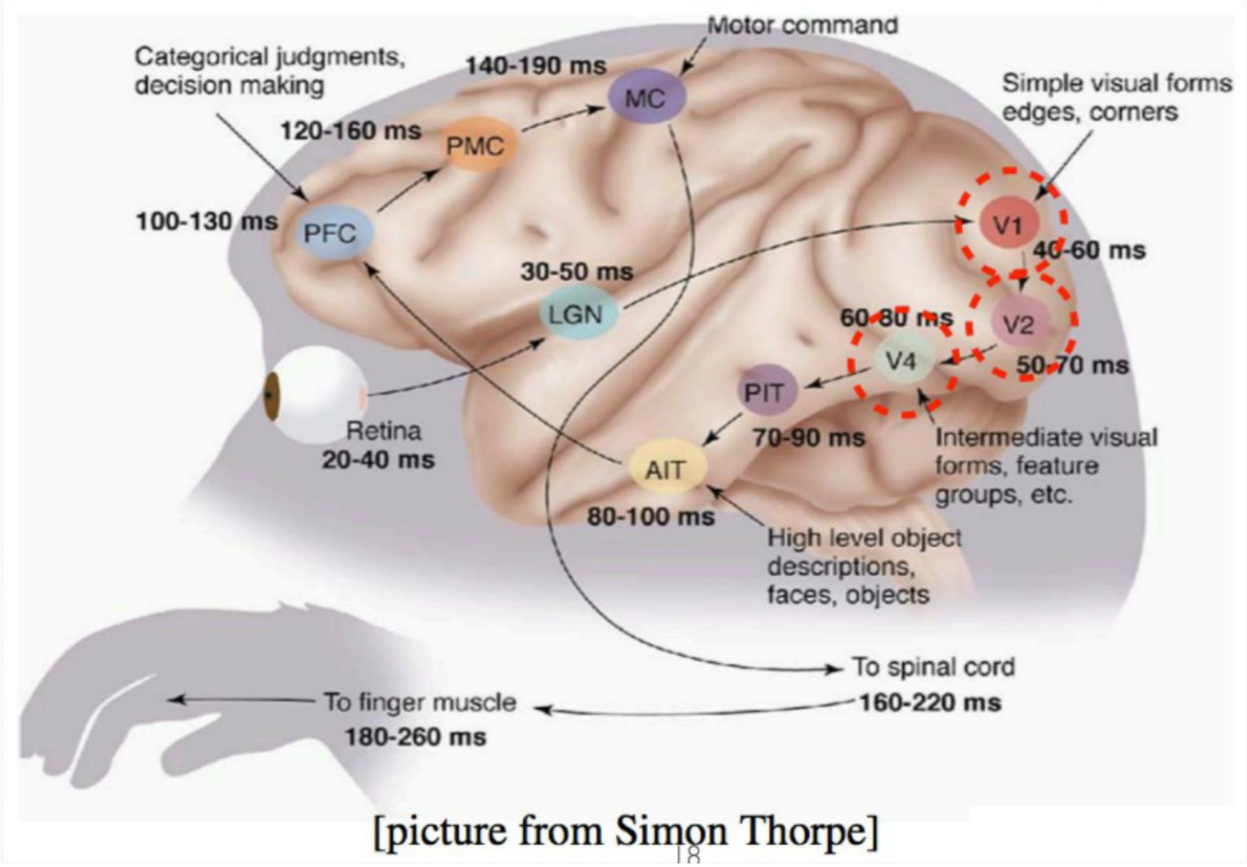
Slide Credit: Hugo Larochelle



Biological Neural Networks

Visual Cortex

- The **visual cortex** is the part of the brain responsible for processing visual information that we get from our retina.

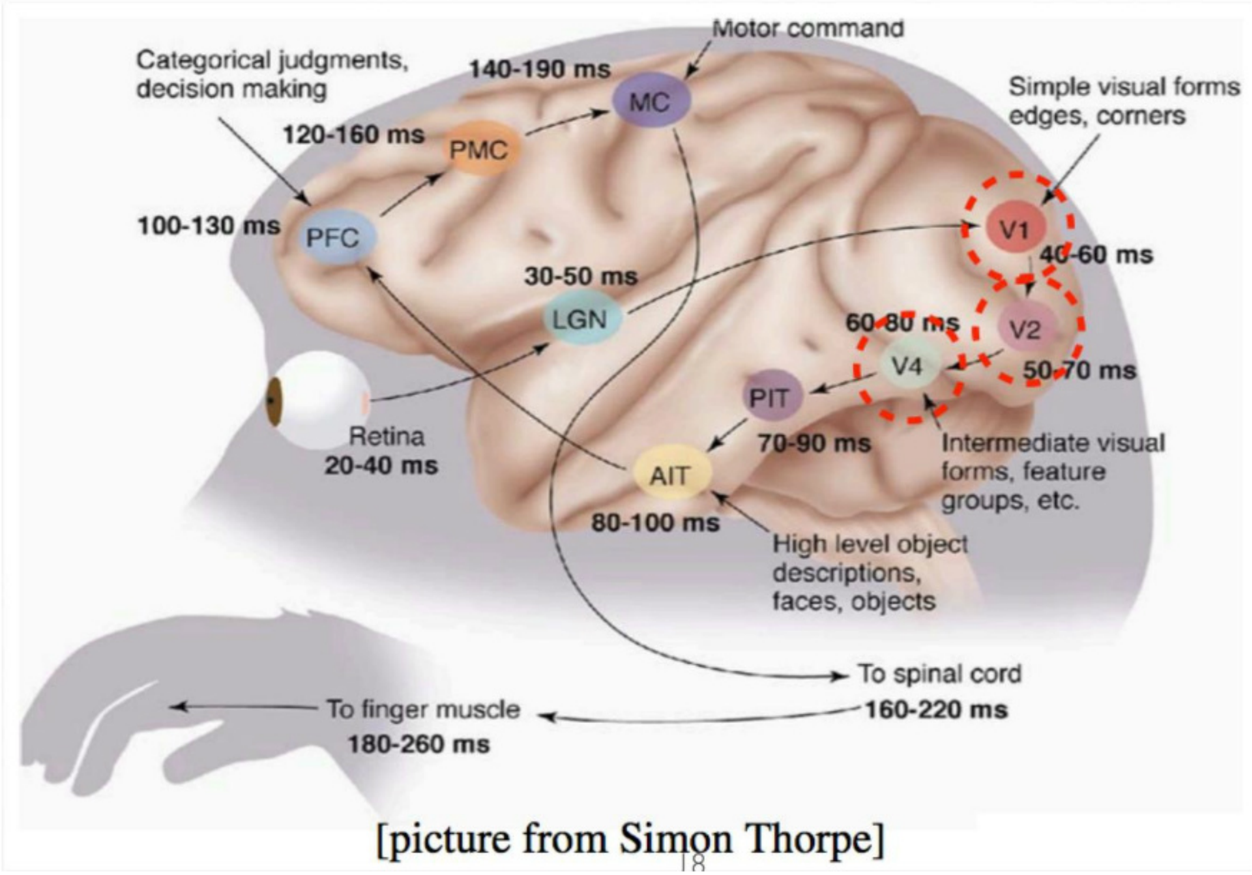


[picture from Simon Thorpe]

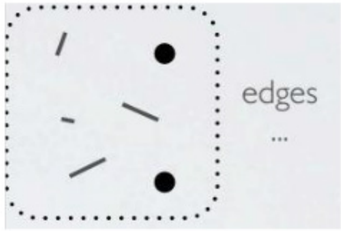
Slide Credit: Hugo Larochelle

Visual Cortex

- **V1** has a set of neurons (simple cells) that are sensitive to particular simple visual forms like edges and corners



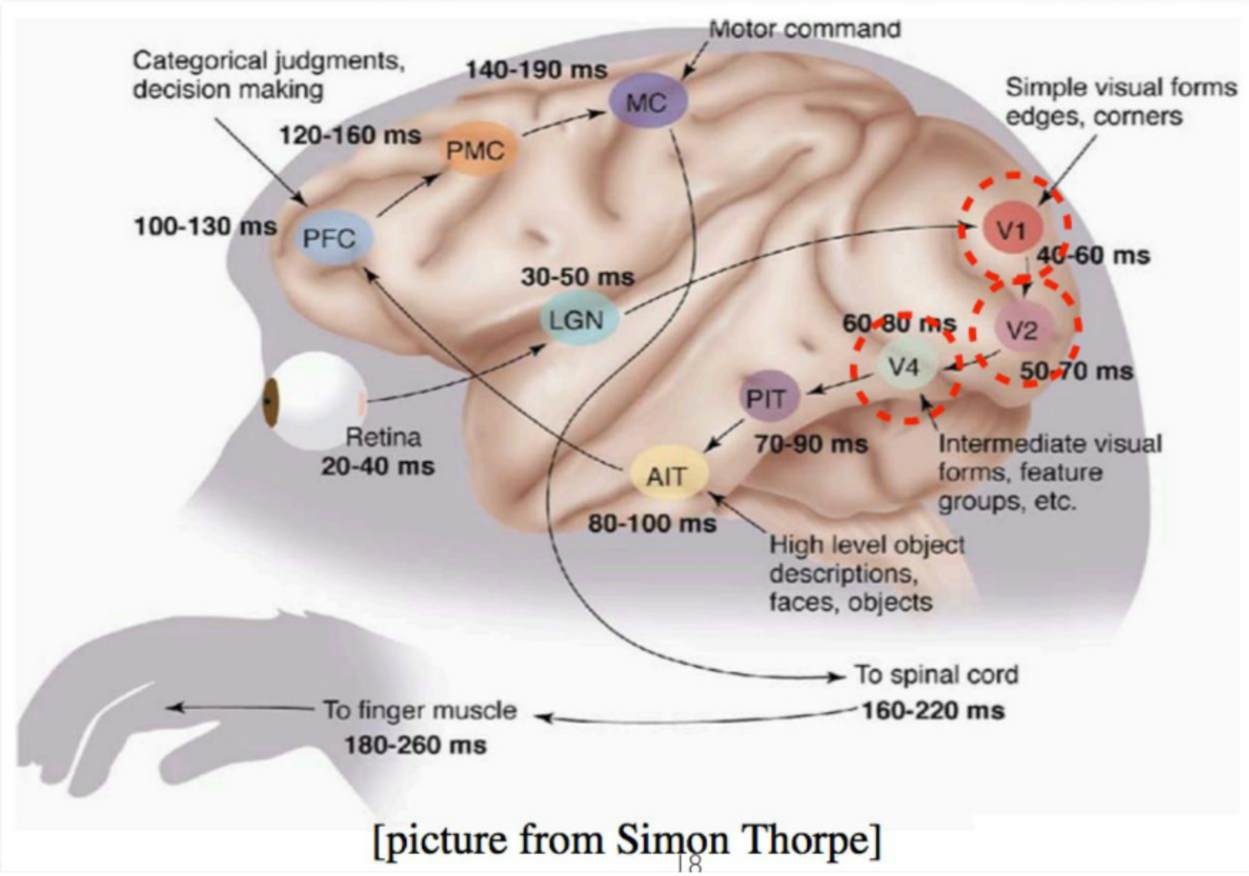
[picture from Simon Thorpe]



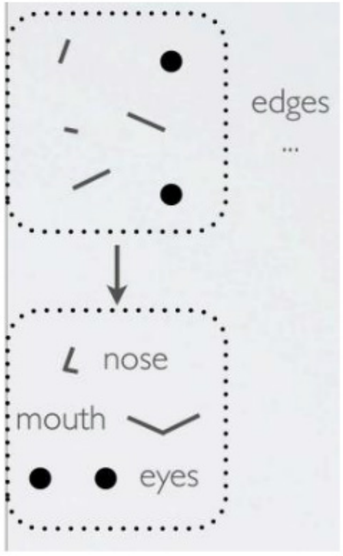
Slide Credit: Hugo Larochelle

Visual Cortex

- **V4** is tuned for object features of intermediate complexity, like simple geometric shapes.



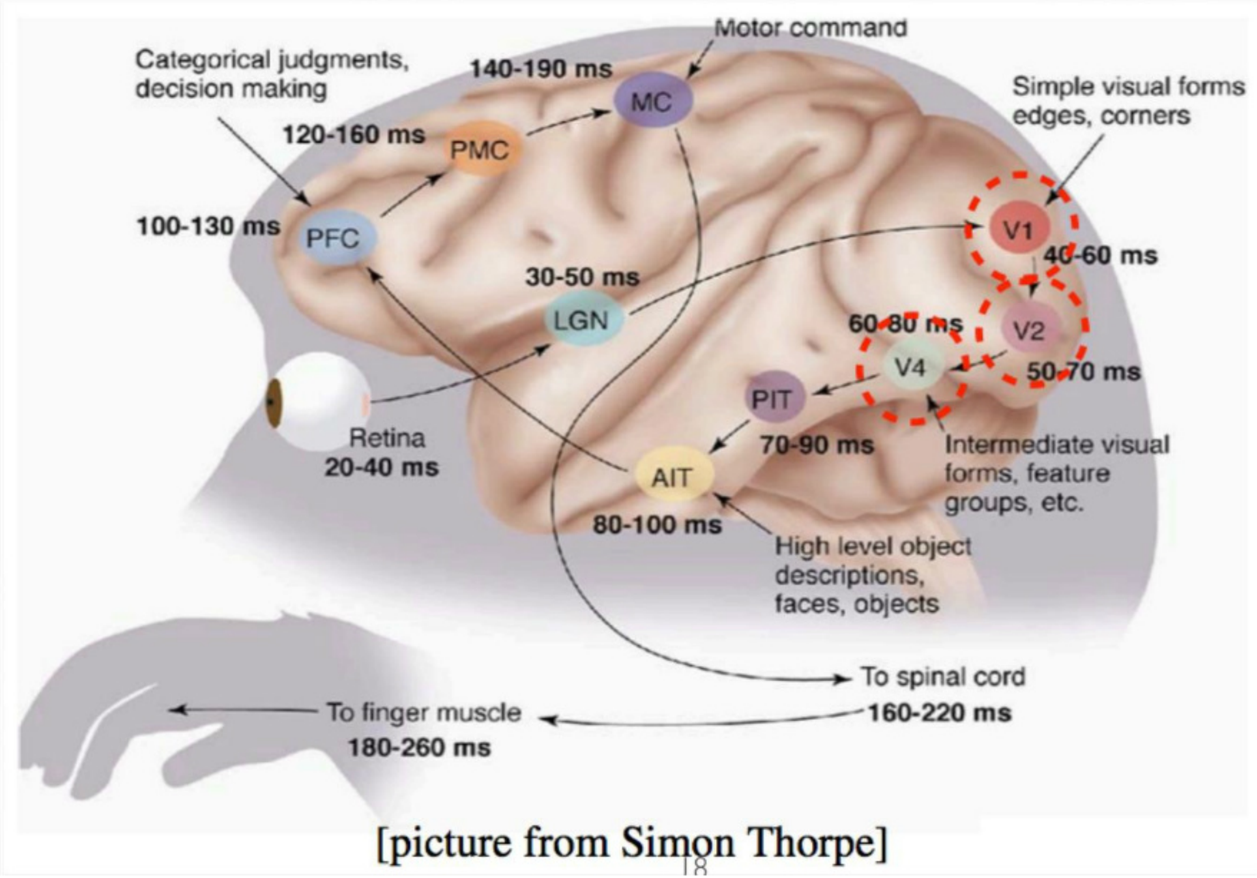
[picture from Simon Thorpe]



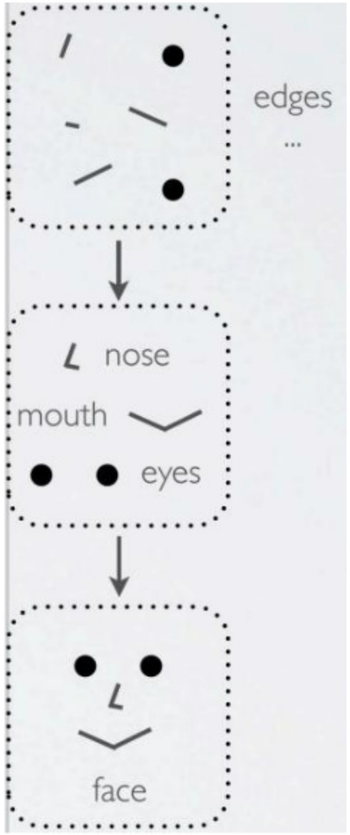
Slide Credit: Hugo Larochelle

Visual Cortex

- AIT



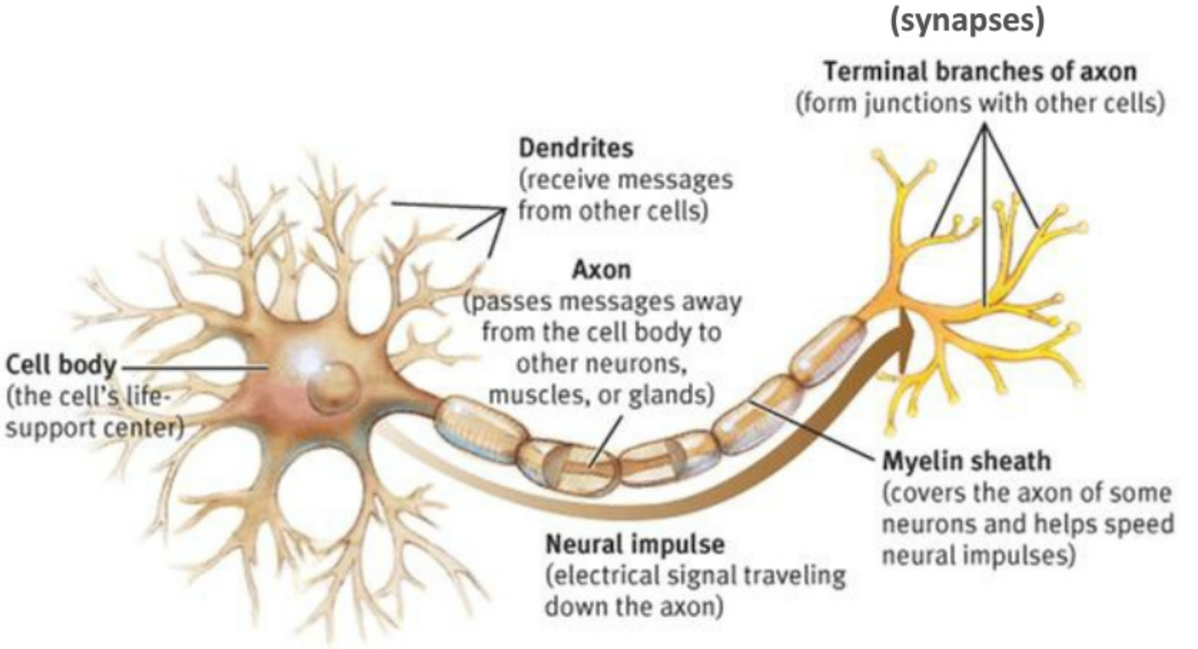
[picture from Simon Thorpe]



Slide Credit: Hugo Larochelle

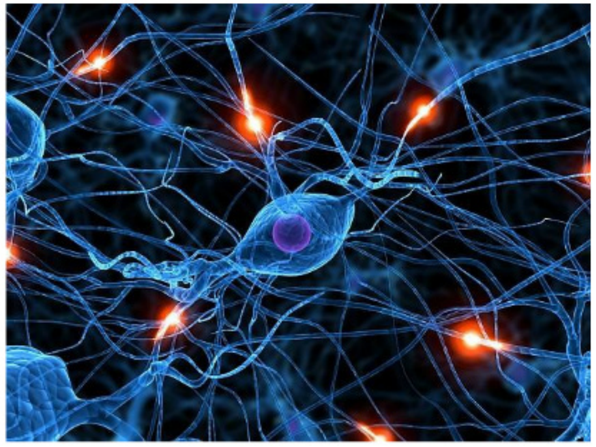
Biological Neurons

- We estimate around 10^{10} and 10^{11} the number of neurons in the human brain:
  - they receive information from other neurons through their **dendrites**
  - the ‘‘process’’ the information in their **cell body** (soma)
  - they send information through a ‘‘cable’’ called an **axon**
  - the point of connection between the axon branches and other neurons’ dendrites are called **synapses**, which regulate a chemical connection whose strength affects the input to the cell.



Biological Neurons

- An **action potential** is an electrical impulse that travels through the axon:
  - this is how neurons communicate
  - it generates a ‘‘spike’’ in the electric potential (voltage) of the axon
  - an action potential is generated at neuron only if it receives enough (over some threshold) of the ‘‘right’’ pattern of spikes from other neurons



- Neurons can generate several such spikes every seconds:
  - the frequency of the spikes, called **firing rate**, is what characterizes the activity of a neuron
  - neurons are always firing a little bit, (spontaneous firing rate), but they will fire more, given the right stimulus

Biological Neurons

- This is what artificial neurons approximate:
  - Weights (synapse) between neurons model whether neurons excite or inhibit each other
  - Inputs (dendrites) that carry the signal to neuron (cell body) where they all get summed
  - Summation of inputs and activation model processes the incoming activations and converts them into output activations (cell body)