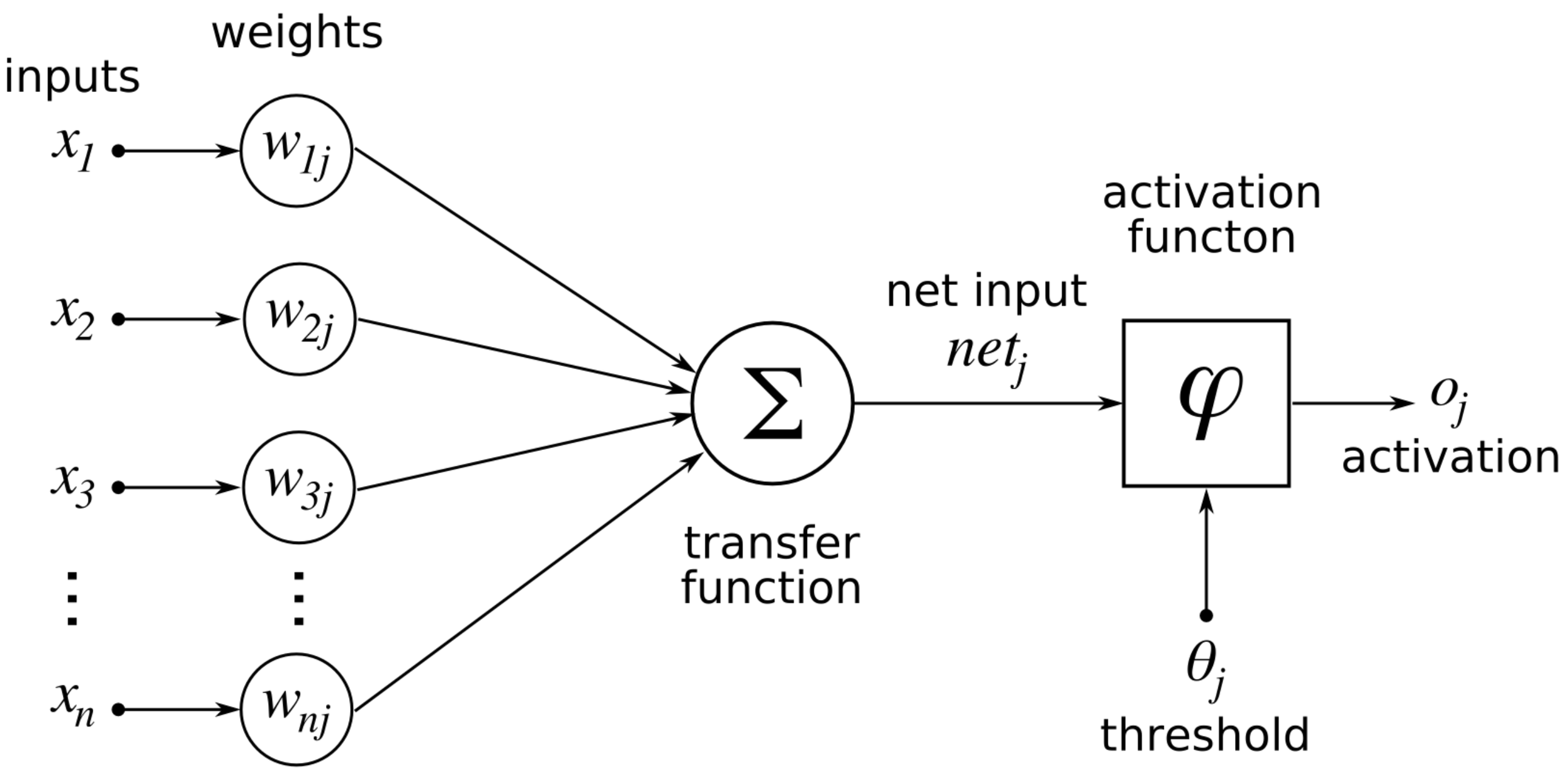


Artificial Neural Networks

Optimizing Functions with Gradient Descent

Popularized: Deep Learning

Artificial Neural Networks (ANNs) are statistical learning algorithms inspired by biological neural networks.

ANNs are often used to approximate unknown functions that can depend on a large number of inputs and are hard to solve with handmade rules.

Deep Learning
en.wikipedia.org/wiki/Deep_learning

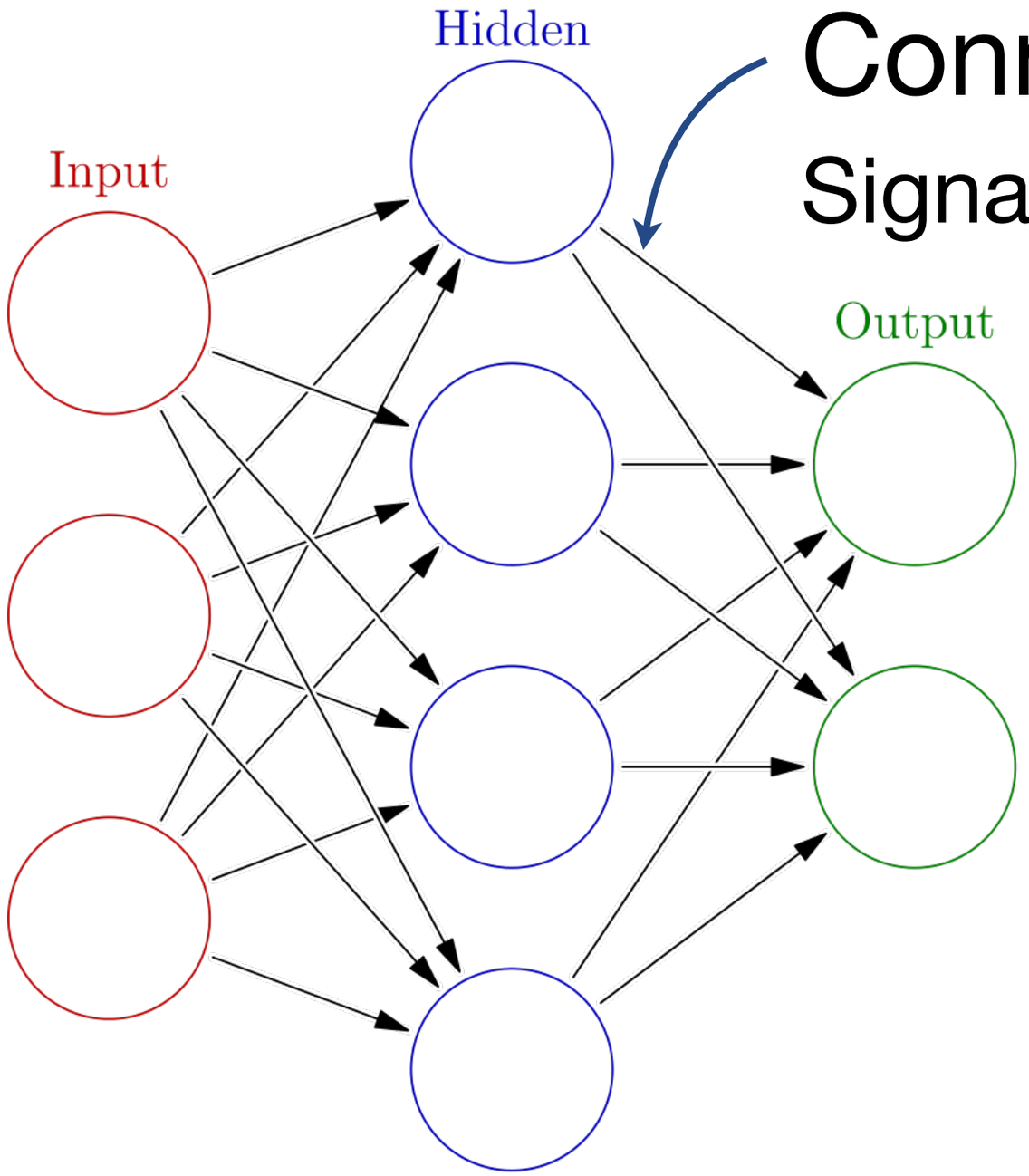
Types of Artificial Neural Nets

- **Feedforward Neural Networks:**
  Connections do not form any cycles
- **Convolutional Neural Networks:**
  Also acyclic but with shared weights
- **Recurrent Neural Networks:**
  With cycles to learn temporal pattern

Artificial Neural Network
en.wikipedia.org/wiki/Artificial_neural_network

Basic Setup of Neurons

Neuron:

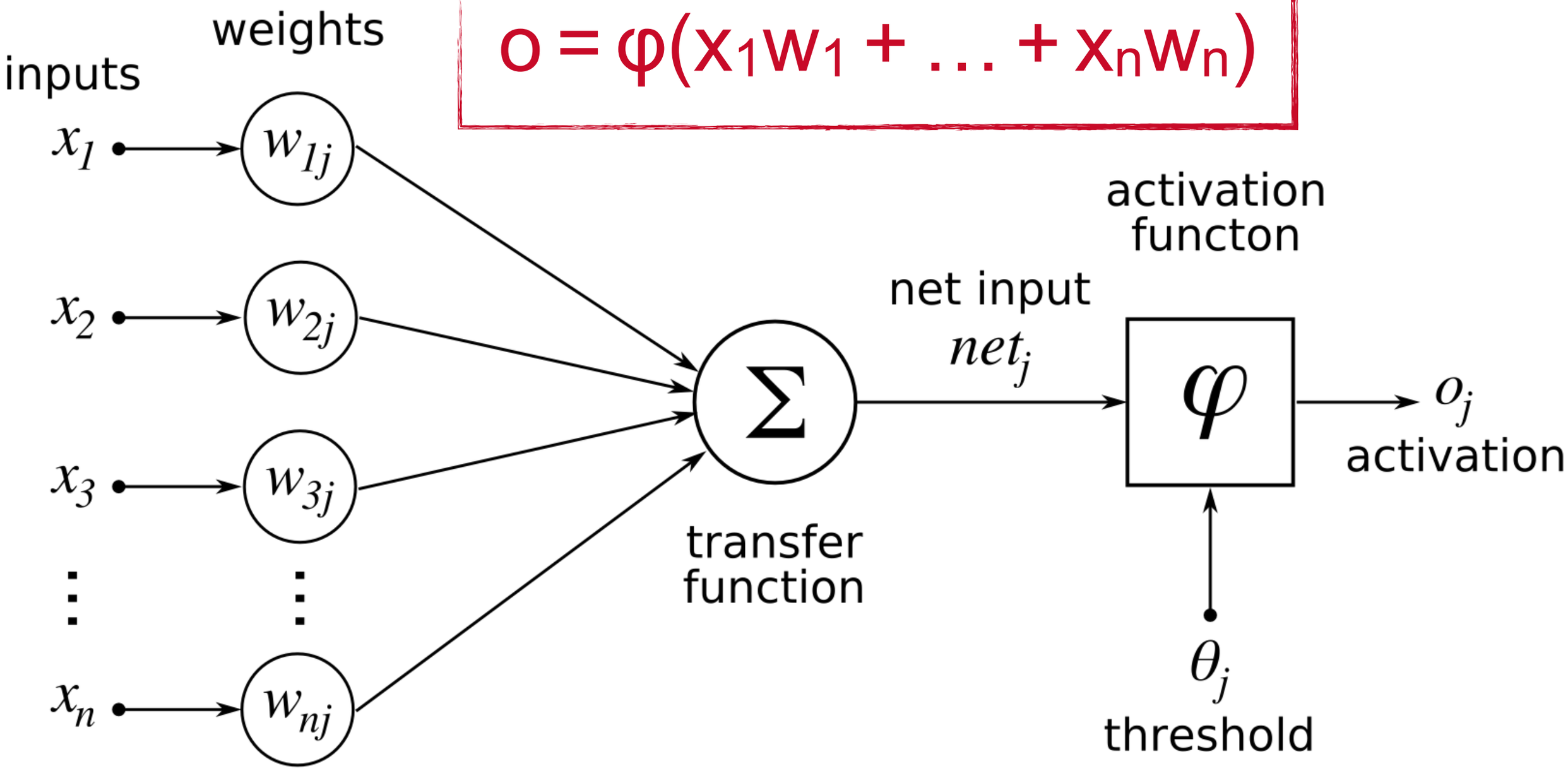


Connection for Signal Forwarding

Feedforward Neural Network
en.wikipedia.org/wiki/Feedforward_neural_network

Activation Function

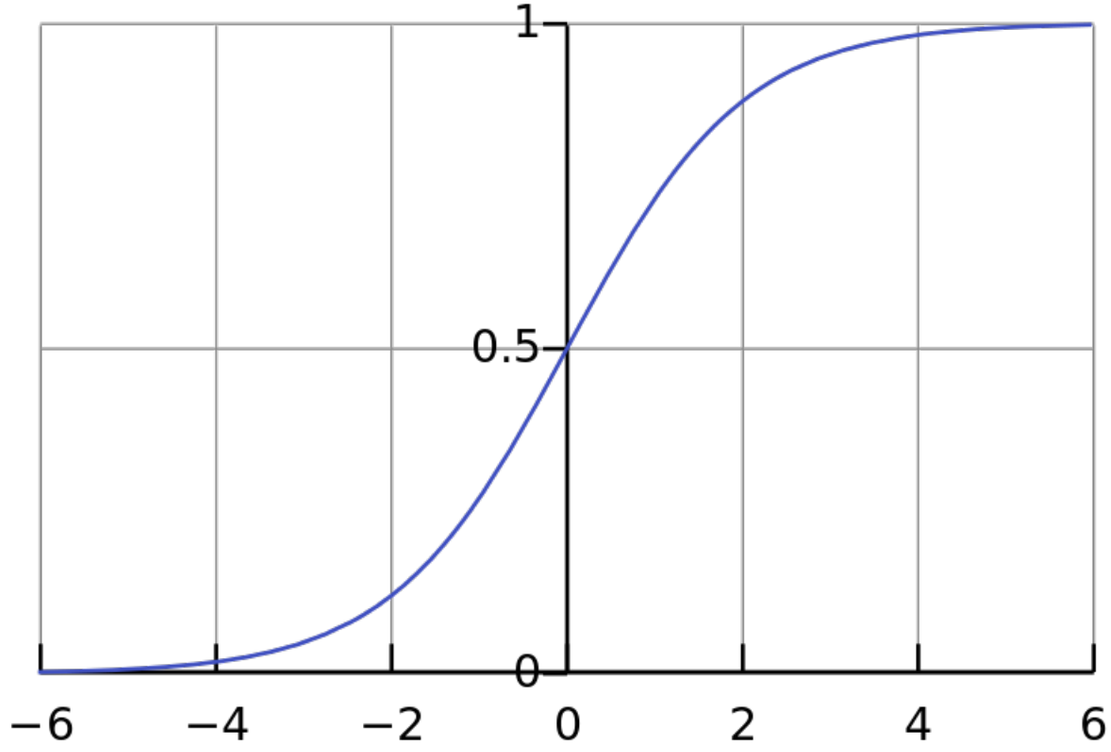
$$o = \varphi(x_1 w_1 + \ldots + x_n w_n)$$



Activation Functions
en.wikibooks.org/wiki/[…]/Activation_Functions

Activation Functions

- May **not** be **linear** or layers collapse
- **Sigmoid functions** like $f(x) = (1 + e^{-x})^{-1}$
- Also popular: **rectified linear** $f(x) = \max(0, x)$ with threshold



Activation Function
en.wikipedia.org/wiki/Activation_function

Optimization Problem

Goal: Find weights that minimize the mean squared error $E = \frac{1}{2} (f(\mathbf{x}) - \mathbf{y})^2$

Given search space and **cost function**, apply ordinary optimization algorithms:

- Evolutionary Algorithm
- Gradient Descent

Mathematical Optimization
en.wikipedia.org/wiki/Mathematical_optimization

Evolutionary Algorithm

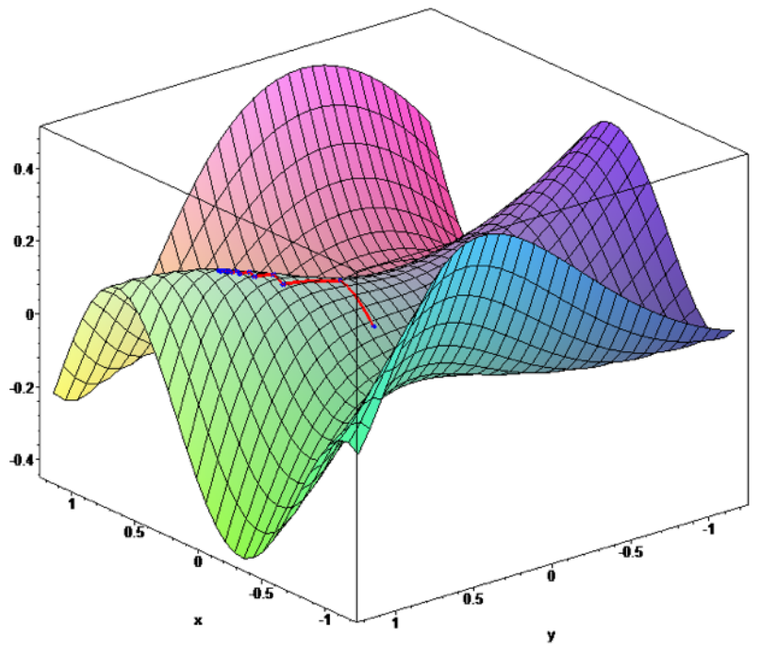
Inspired by biological evolution, apply **reproduction**, **mutation**, **recombination** and **selection** with the mean squared error as **fitness function** on parameters.

- No assumptions made about space
- Difficulty in finding a good crossover

Evolutionary Algorithm
en.wikipedia.org/wiki/Evolutionary_algorithm

Gradient Descent

- **Goal**: Find minimum of function F(**x**)
- Take steps proportional to negative of the **gradient**: $\mathbf{x}_2 = \mathbf{x}_1 - \gamma \nabla F(\mathbf{x}_1)$
- Only **local minimum**
- Function needs to be **differentiable**



Gradient Descent
en.wikipedia.org/wiki/Gradient_descent

Partial Derivative

Derivative with respect to one variable

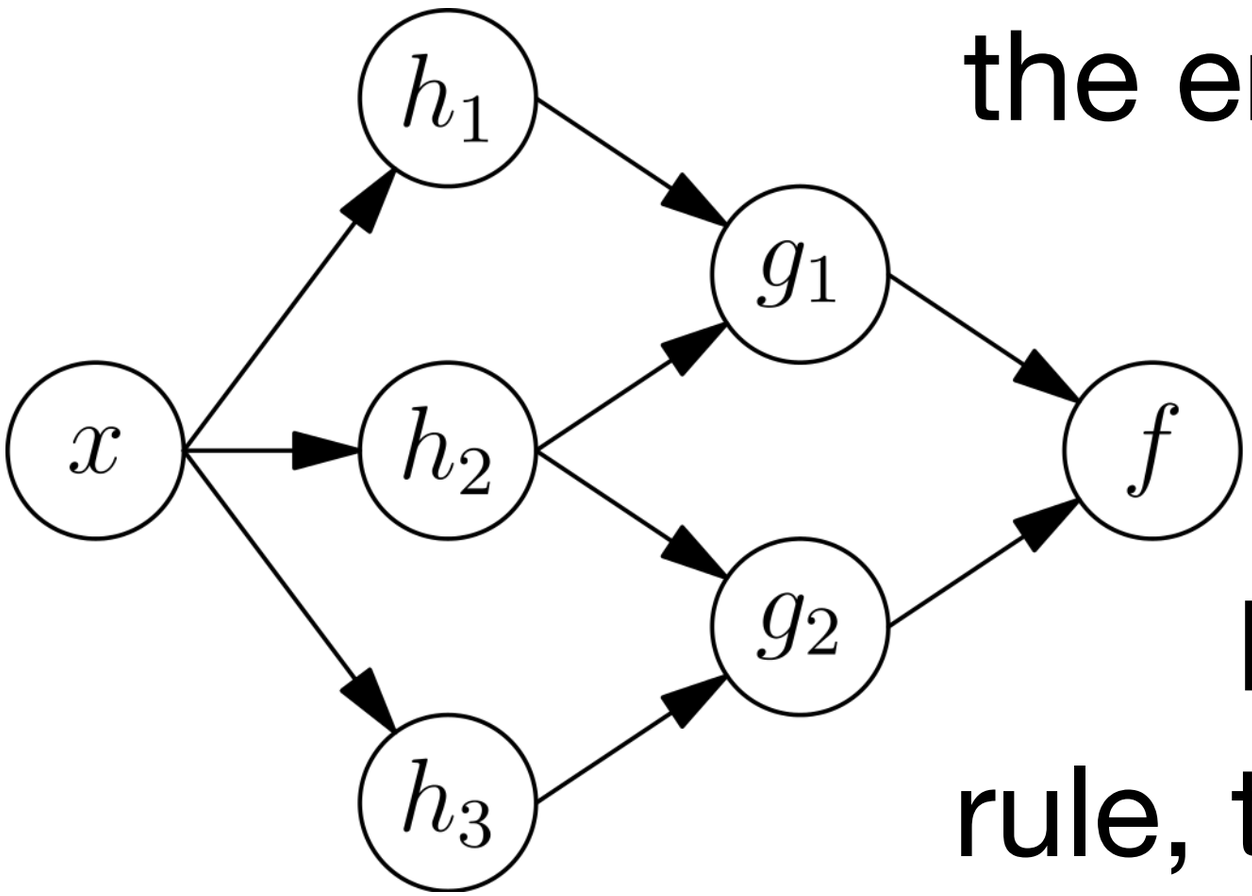
Example: $f(x, y) = x^2 + xy + y^2$, then the partial derivative wrt. x is $\partial f/\partial x = 2x + y$

Nabla symbol ∇ denotes the **gradient:**

$\nabla f(x_1, \ldots, x_n) = \text{vector } (\partial f/\partial x_1, \ldots, \partial f/\partial x_n)$

Chain rule: $(f(g(x)))' = f'(g(x)) \cdot g'(x)$

Backpropagation



Calculate the gradient of the error with respect to each weight.

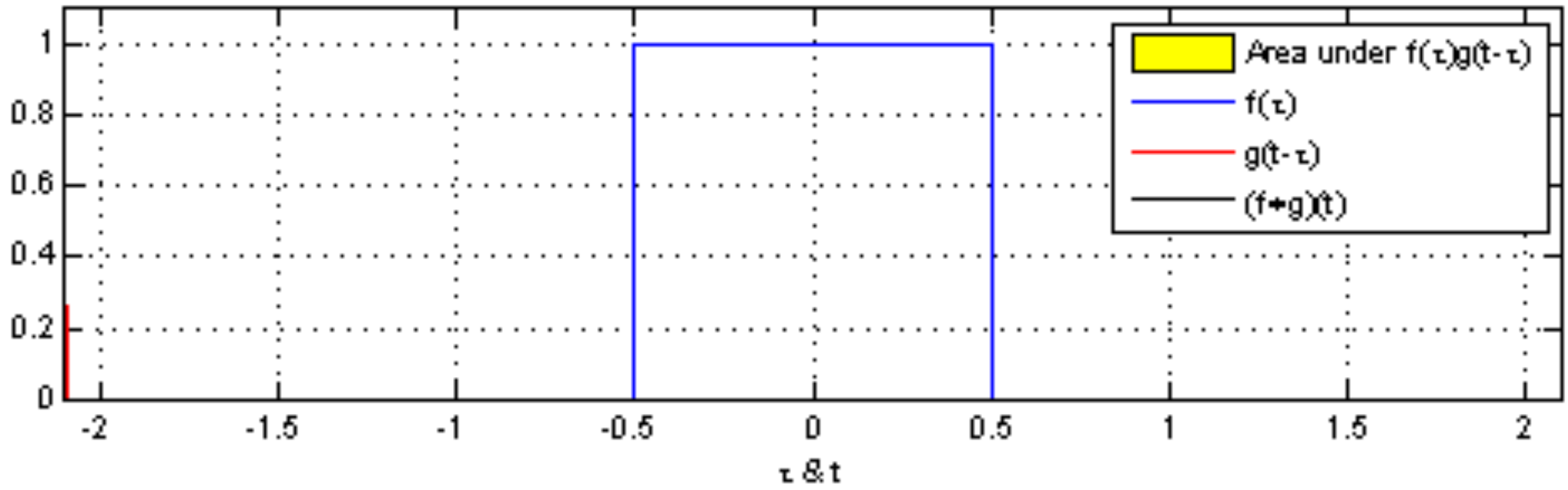
Due to the chain rule, the values of the right layers can be reused.

Backpropagation
en.wikipedia.org/wiki/Backpropagation

Convolution

The integral of the product of two functions after one is reversed and shifted:

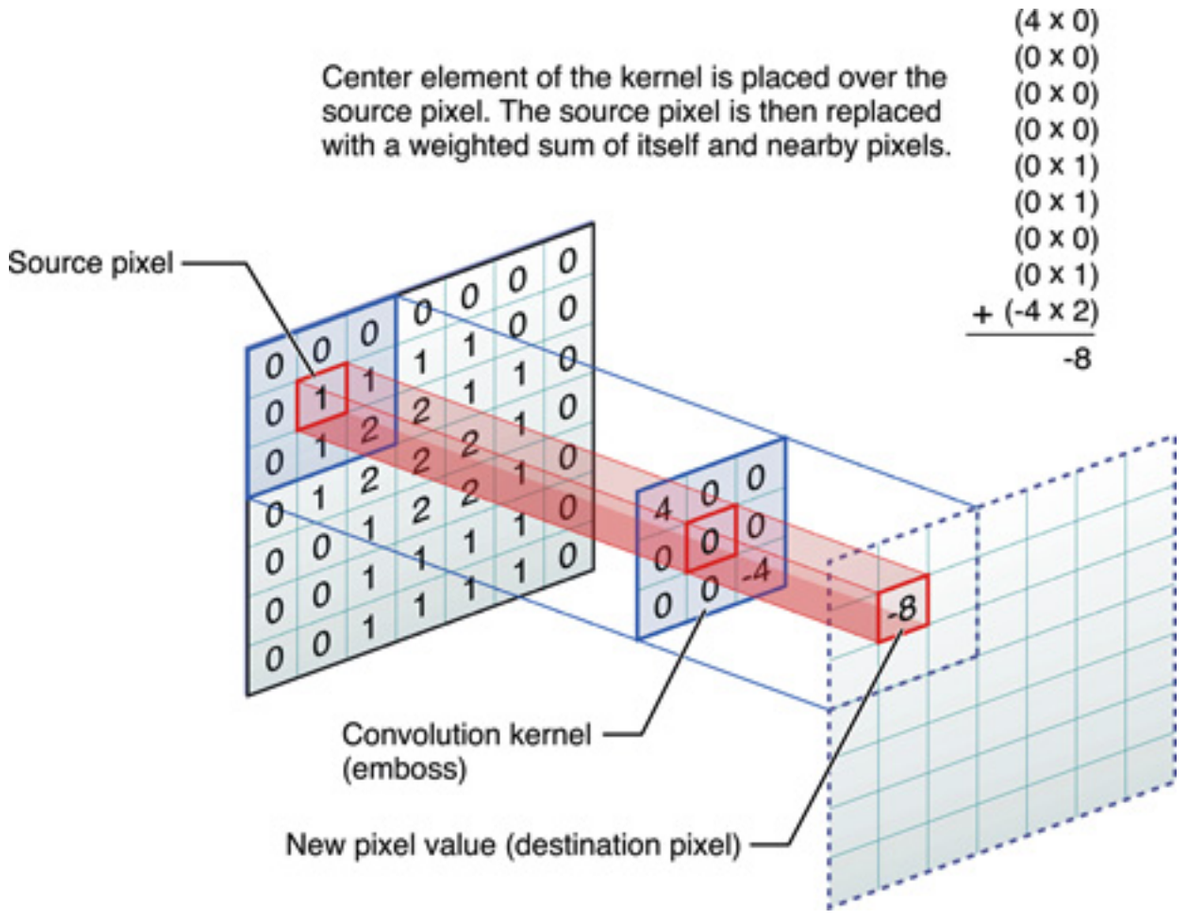
$(f * g)(t) = \int f(\tau)\, g(t-\tau)\, d\tau$ (as a definition)

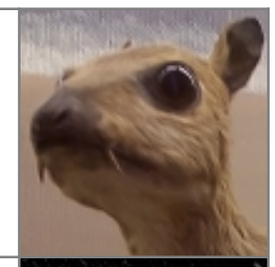
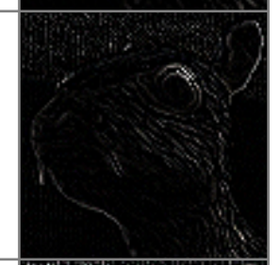
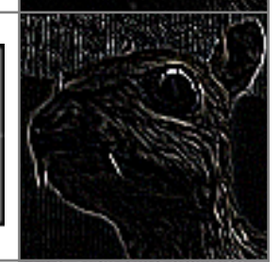
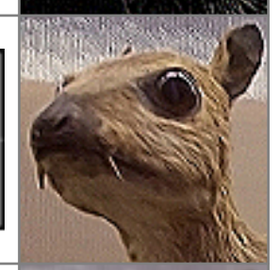
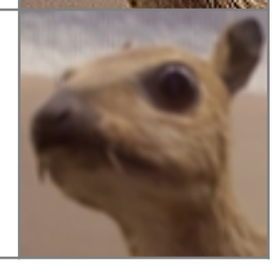


Convolution
en.wikipedia.org/wiki/Convolution

Feature Detection

Shift kernel across function or image



Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

Kernel (Image Processing)
en.wikipedia.org/wiki/Kernel_(image_processing)

Convolutional Neural Networks

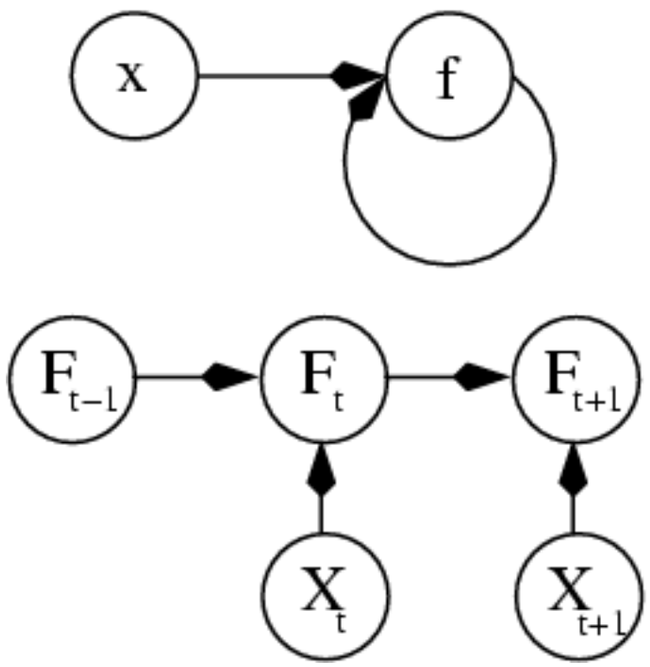
- Feedforward network with input tiled such that there are overlapping areas
- Convolution where **kernels also learnt**
- **Weight sharing** reduces free variables
- Often with **rectified linear** units (ReLU)
- **Prevent overfitting** w. cross-validation

Recurrent Neural Networks

Connections between neurons form a directed **cycle**, creating internal states

Useful for processing **sequences of inputs** (like speech)

Most RNNs can compute anything a conventional computer can compute!



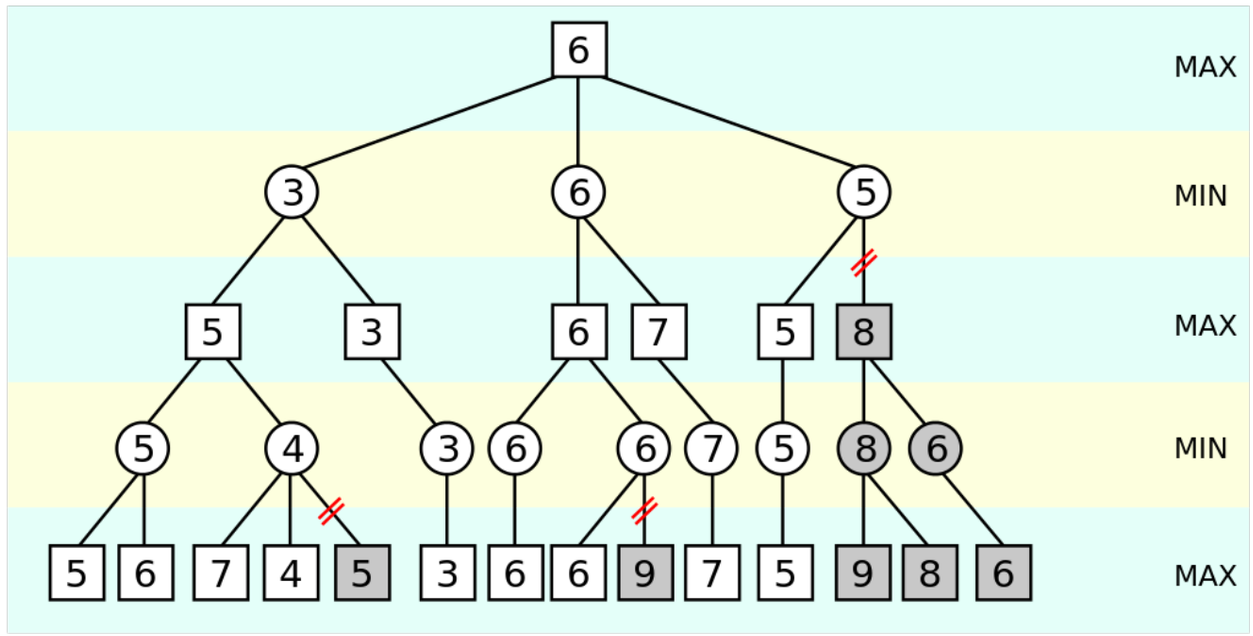
Recurrent Neural Network
en.wikipedia.org/wiki/Recurrent_neural_network

Combinatorial Explosion

Games consist of states & transitions.

Search trees grow very fast but ANNs can be used to learn useful heuristics!

Alpha-beta pruning for **minimax** algorithm:



Combinatorial Explosion
en.wikipedia.org/wiki/Combinatorial_explosion

Reinforcement Learning

- Idea: Train a CNN to learn the goodness of actions given recent frames
- If environment cannot be controlled, adjust current estimate after next estimate (temporal difference learning)
- Favor sooner rewards w. greediness factor because no progress otherw.

Reinforcement Learning (CNN)

DEEPMIND

Google develops self-learning computer program
www.theguardian.com/technology/2015/feb/25/[…]

Outlook: Huge Responsibility

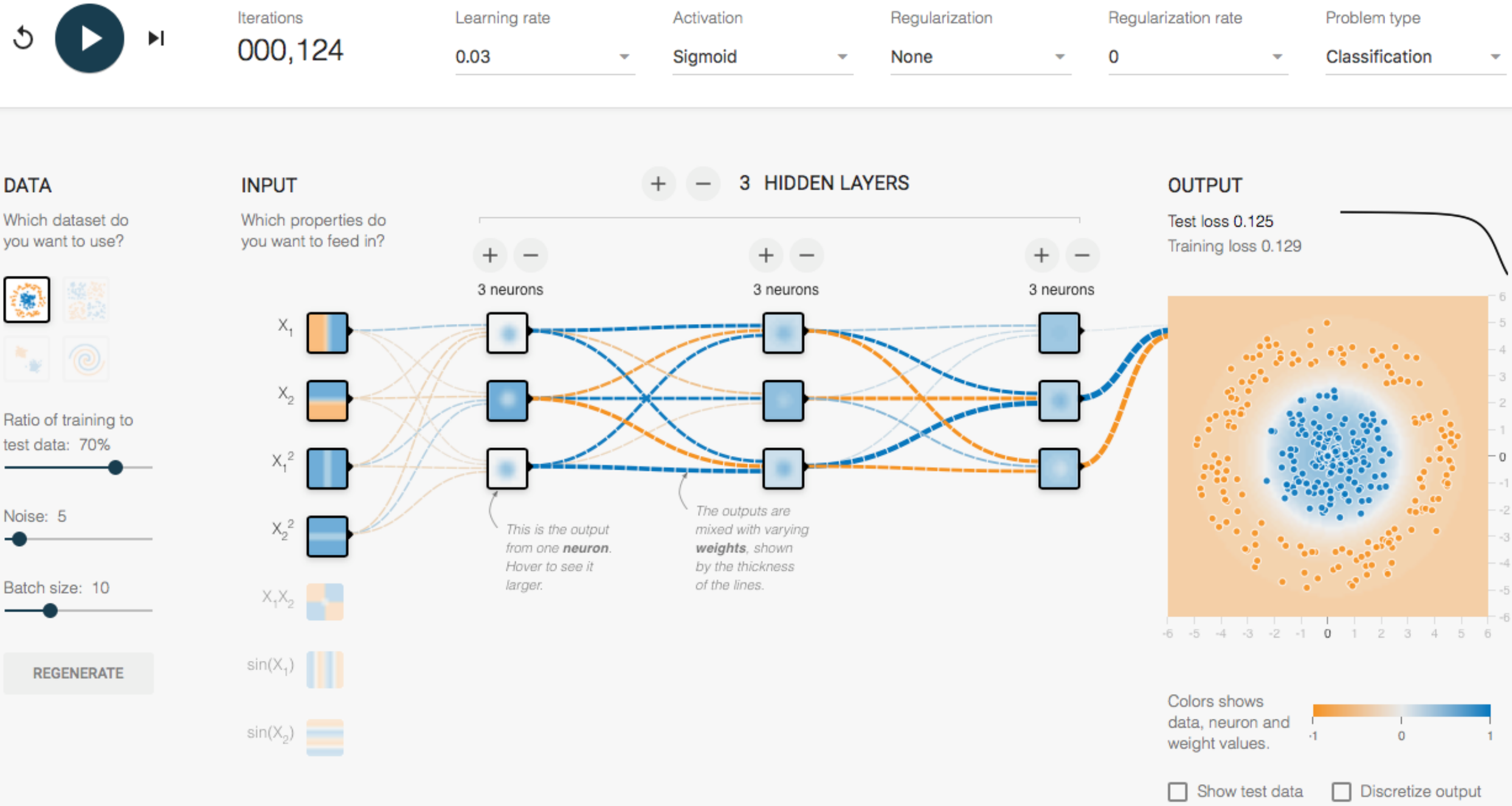
… in the near-term. For the long-term, see www.superintelligence.ch.

Train CNN

Deploy CNN

Campaign to Stop Killer Robots
www.stopkillerrobots.org

Train your own Neural Network:



Neural Network in Browser
playground.tensorflow.org