# Neural Networks and Deep Learning
## Introduction II

Nicholas Dronen

Department of Computer Science
dronen@colorado.edu

January 16, 2019

University of Colorado Boulder

Questions?

History

Biological Motivation

Artificial Neural Networks

# Hinton's Brief History of Deep Learning

What was hot in 1987?

- Back propagation
- Neural networks

What happened the past 30 years?

- Computers got faster
- Data sets got larger
- Software tools improved (automatic differentiation)

What is hot in 2017?

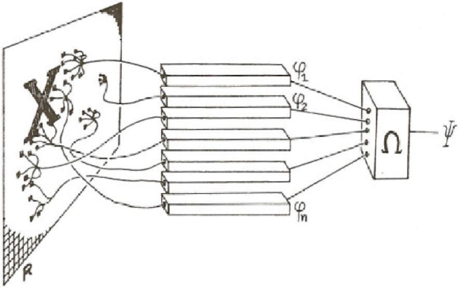
- Back propagation
- Neural networks

Neural Network History

# 1962

Frank Rosenblatt, Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms

*A Perceptron can learn anything you can program it to do.*

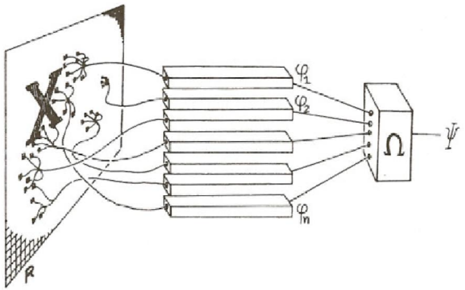


Neural Network History

## 1969

Minsky and Papert, Perceptrons: An introduction to computational geometry

*There are many things a perceptron can't learn to do.*



Neural Network History

## 1970-1985

Attempts to develop symbolic rule discovery algorithms.

## 1986

- Back propagation – Rumelhart, Hinton, and Williams (cf. R. Wengert, A Simple Automatic Derivative Evaluation Program, Article in Communications of the ACM, 1964)
- Overcame many of the Minsky & Papert objections
- Neural networks popular in cognitive science and AI


(circa 1990)

## Neural Network History

# 1990-2005

Bayesian approaches

- take the best ideas from neural networks – statistical computing, statistical learning

Support-Vector Machines

- unlike neural networks, SVMs can be proven to converge

A few old timers keep playing with neural nets

- Hinton, LeCun, Bengio, O'Reilly

Neural networks banished from NIPS.

Neural Network History

## 2005-2012

Attempts to resurrect neural networks with

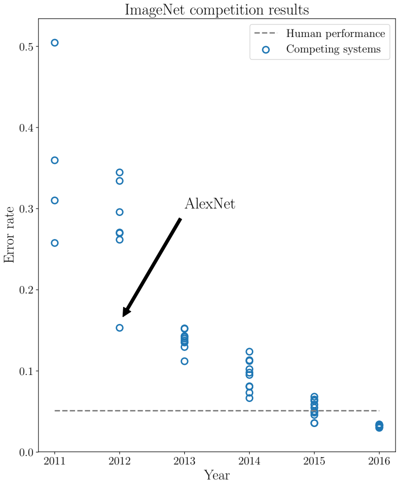
- Unsupervised pre-training
- Probabilistic neural nets
- Alternative learning rules

Neural Network History

# 2012 (AlexNet)-present

New techniques discarded in favor of 1980's-style supervised neural networks with

- Vastly larger supervised training sets
- Hardware accelerators for fast training and inference
- Refinements in optimization and generalization (mostly from Hinton)



## Brains Versus Computers

Tasks that are easy for brains are not easy for computers, and vice versa.

Brains

- Recognizing faces
- Retrieving information based on partial descriptions
- Organizing information – the more information, the better the brain operates

Computers

- Arithmetic
- Deductive logic: $((p \to q) \land \neg q)) \to \neg p$
- Retrieving information based on arbitrary features

$\to$ Brains must operate quite differently than ordinary computers.

# Caricature of How the Brain Operates

The brain is composed of neurons.

Neurons convey and transform information.

What is this information?

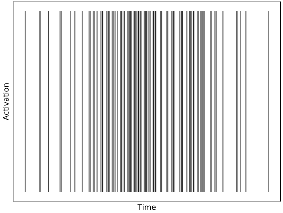



Figure 1: Neural spikes over time.

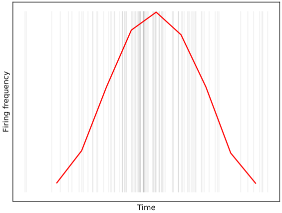


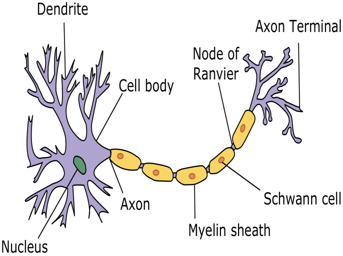

Figure 2: Averaging instantaneous binary-valued activity.

## Neurons

Dendrite receives, integrates signals from other neurons. Neuron cell body "decides". Axons communicate decision to other neurons.

Gross oversimplification

- Many types of neurons (sensory, motor, inter)
- Electrical and chemical interactions
- Many types of connections (e.g. dendrodendritic synapses)



## Key Features of Cortical Computation

- Neurons fire slowly - typically 100Hz, sometimes 1000Hz
- Large number of neurons (10-100 billion)
- Distributed, no central controller (unlike a CPU)
- Neurons receive input from a large number of other neurons (e.g. $10^4$ fan-in and fan-out of cortical pyramidal cells)
- Communication via excitation and inhibition

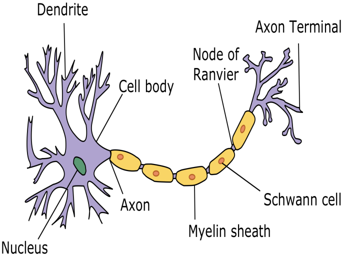
## Key Features of Cortical Computation (cont.)

- Statistical decision making (neurons that unilaterally turn on/off other neurons are rare)
- Learning involves modifying coupling strengths (the tendency of one cell to excite/inhibit another)
- Neural hardware is dedicated to particular tasks (vs. conventional computer memory)
- Information is conveyed by mean firing rate of neuron (i.e. by activation)

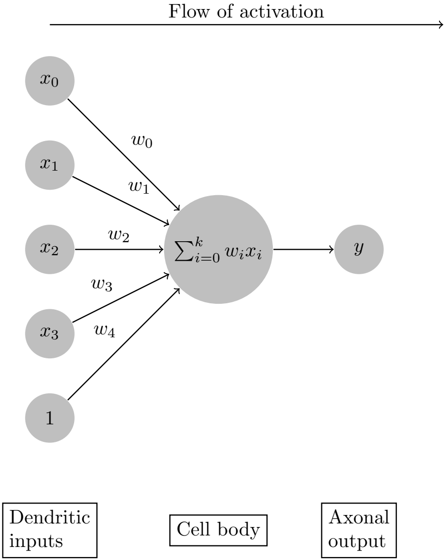
## Conventional Computer Versus the Brain

- Conventional computers
  - One very smart CPU
  - Many dumb memory cells
- Brains, connectionist computers
  - No CPU
  - Many slightly smart memory cells

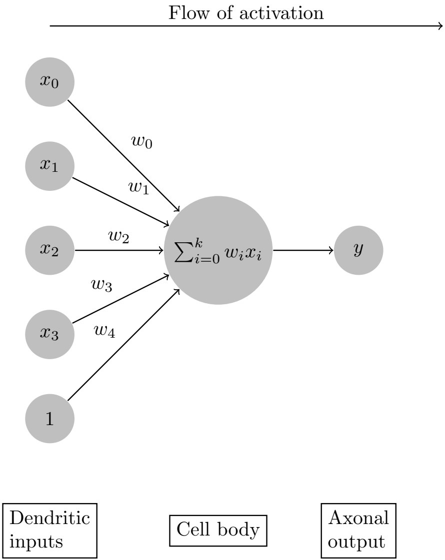
# Modeling Individual Neurons



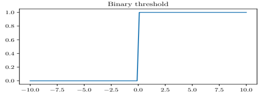

“Energy Landscape” is licensed under CC BY-SA 3.0.

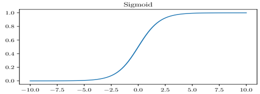


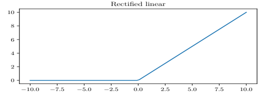
## Modeling Individual Neurons




$$y = \sigma\left(\left(\sum_{i=0}^{k} w_i x_i\right) + \beta\right)$$



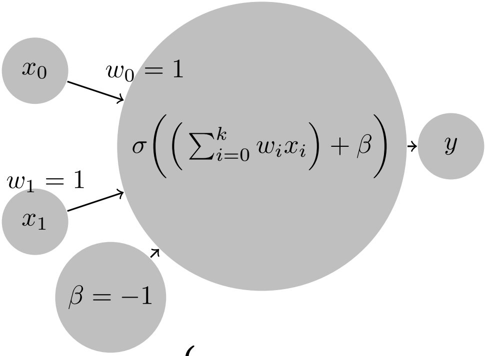



## Computation with a Binary Threshold Unit

# AND gate

| $x_0$ | $x_1$ | $y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

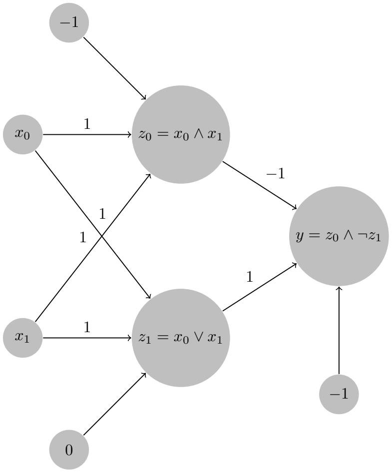



$$\sigma = \begin{cases} 1, & \left(\left(\sum_{i=0}^{k} w_i x_i\right) + \beta\right) > 0 \\ 0, & \left(\left(\sum_{i=0}^{k} w_i x_i\right) + \beta\right) \leq 0 \end{cases}$$

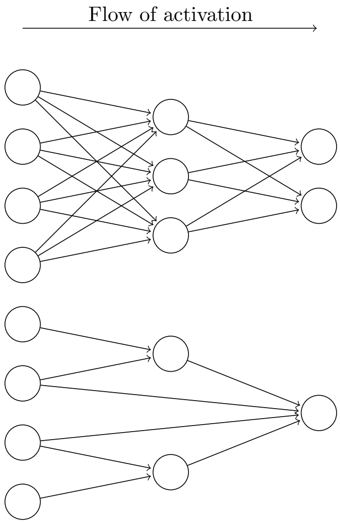
Computation with a Binary Threshold Unit

# XOR gate

| $x_0$ | $x_1$ | $y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |



## Feed-Forward Architectures



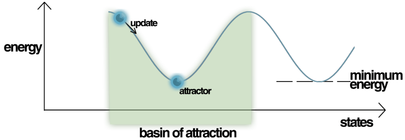

- Activation flows in one direction.
- Associates input and output patterns.

| | | |
|---|---|---|
| big, growling | $\rightarrow$ | run away |
| small, round, orange | $\rightarrow$ | eat |
| small, round, red | $\rightarrow$ | eat |
| small, growling | $\rightarrow$ | run away |

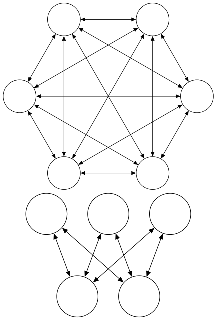
- Learning: adjust connections to achieve input-output mapping

## Recurrent Architectures

- Example: Hopfield Network (demo, video demo)
- Achieves best interpretation of partial or noisy patterns – e.g. MAR--M-LLOW.
- Learning: establishes new attractors and shifts attractor boundaries.



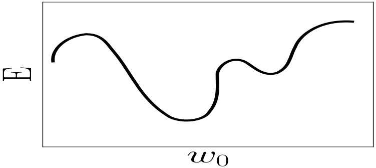


Supervised Learning in Neural Networks

1. Assume a set of training examples $(x^i, y^i)$.
   - big, growling $\rightarrow$ run away
   - MAR--M-LLOW $\rightarrow$ MARSHMALLOW
2. Define a measure, $E$, of network error (cost, loss)

$$E = \sum_i ||y^i - \hat{y}^i||^2$$




3. Make small, incremental changes to the weights to decrease the error (cost, loss) (i.e. gradient descent)