Artificial Intelligence

Xiaoqing Zheng
zhengxq@fudan.edu.cn

Deep Learning

- ***Deep learning*** algorithms have been proposed in recent years to move ***machine learning*** systems towards the discovery of ***multiple levels of representation***.
- Companies like ***Google***, ***Microsoft***, ***Apple***, ***IBM*** and ***Baidu*** are investing in deep learning, with the first widely distributed products being used by consumers aimed at ***speech recognition***.
- The ***New York Times*** covered the subject twice in 2012, with front-page articles. Another series of articles (including a third New York Times article) covered a more recent event showing off the application of deep learning in amajor Kaggle competition for drug discovery (for example see "Deep Learning - The Biggest Data Science Breakthrough of the Decade"
- ***Google*** bought out ("acqui-hired") a company (***DNNresearch***) created by University of Toronto professor ***Geoffrey Hinton*** (the founder and leading researcher of deep learning) and two of his PhD students, ***Ilya Sutskever*** and ***Alex Krizhevsky***, with the press writing titles such as "***Google Hires Brains that Helped Supercharge Machine Learning***"

Deep Learning Guys

Yann LeCun
New York University

Andrew Ng
Stanford

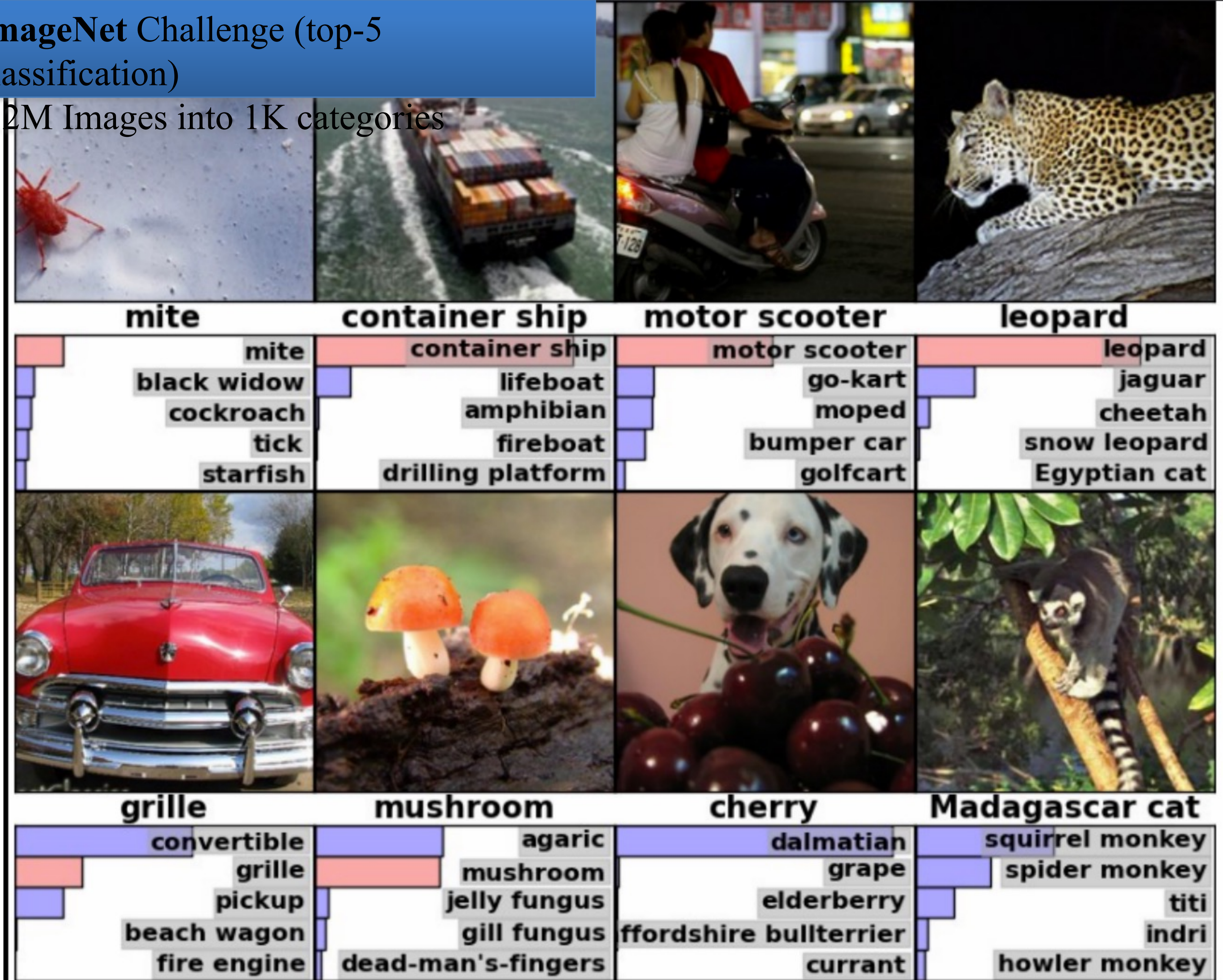
Yoshua Bengio
University of Montreal

Geoffrey Hinton
University of Toronto
Google

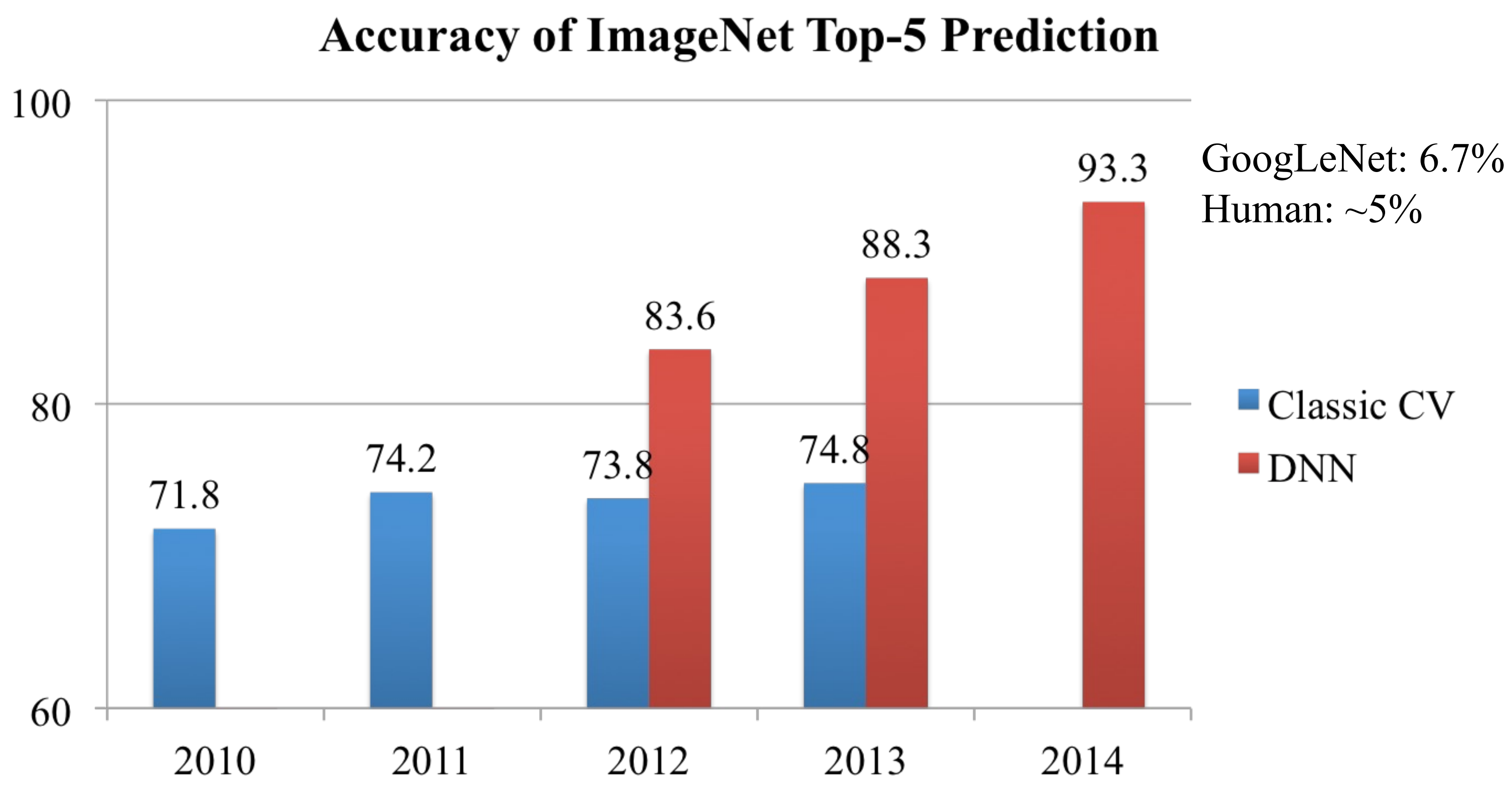
Hinton with his two PhD Students
DNNresearch

ImageNet Challenge (top-5 classification)

1.2M Images into 1K categories

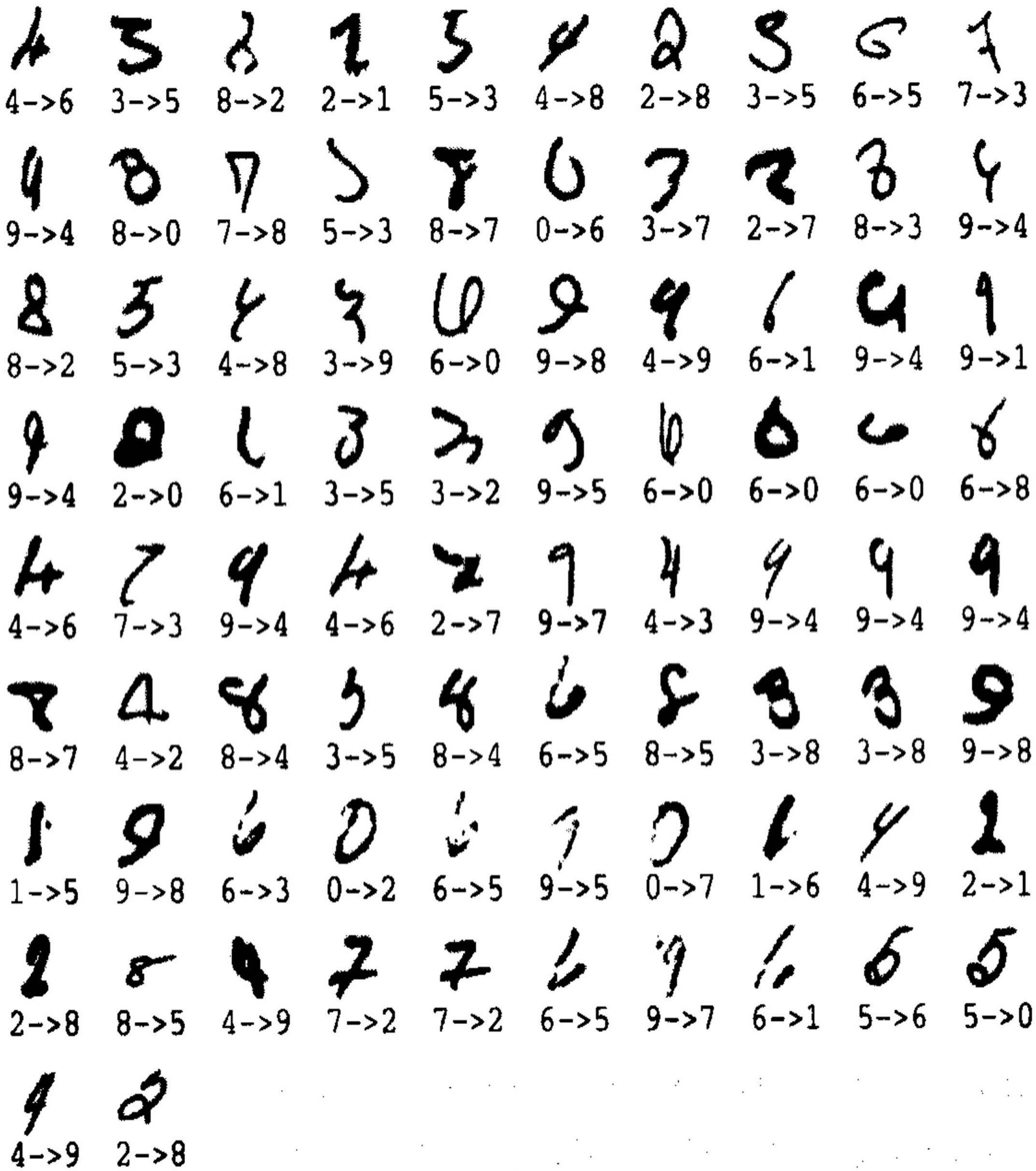


The Rise of Deep Neural Net



Show and Tell

Figure 5. A selection of evaluation results, grouped by human rating.

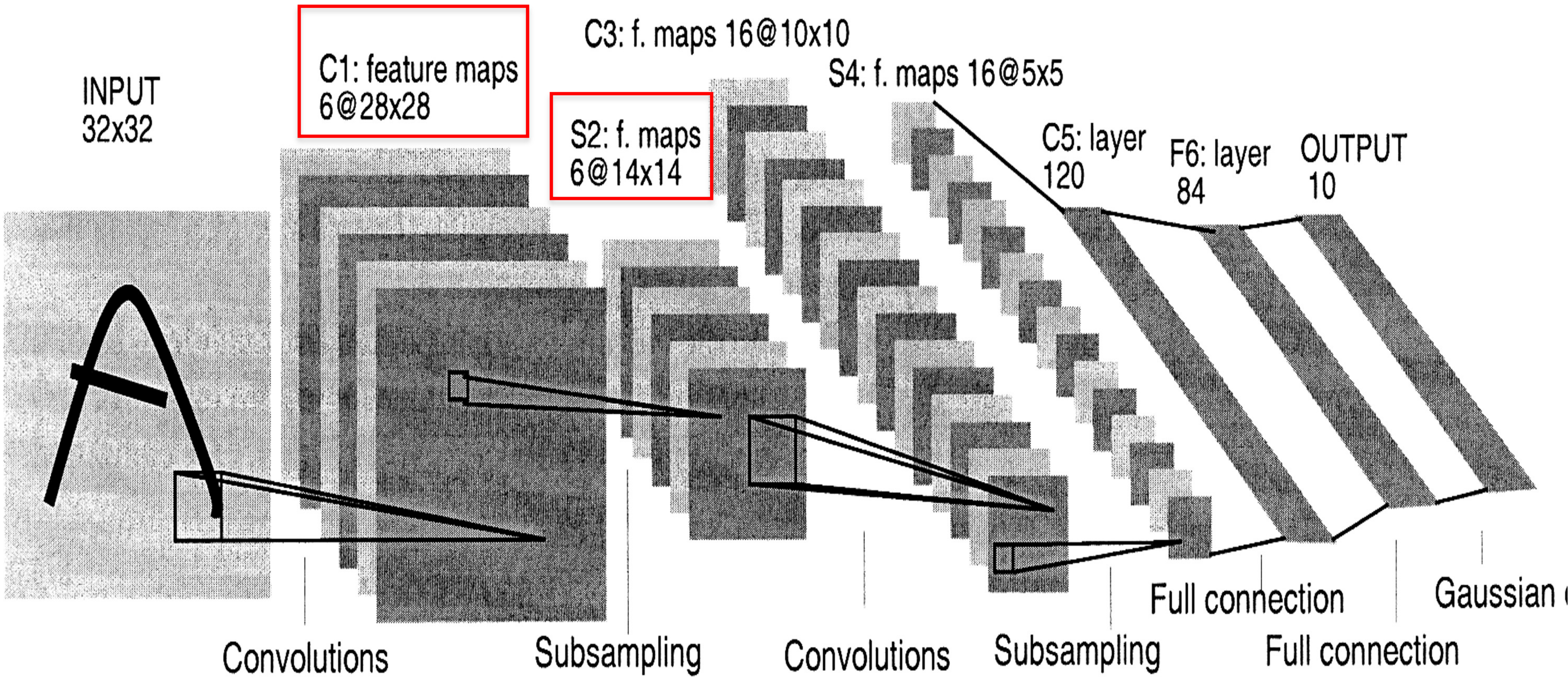


The 82 errors made by LeNet5

Notice that most of the errors are cases that people find quite easy.

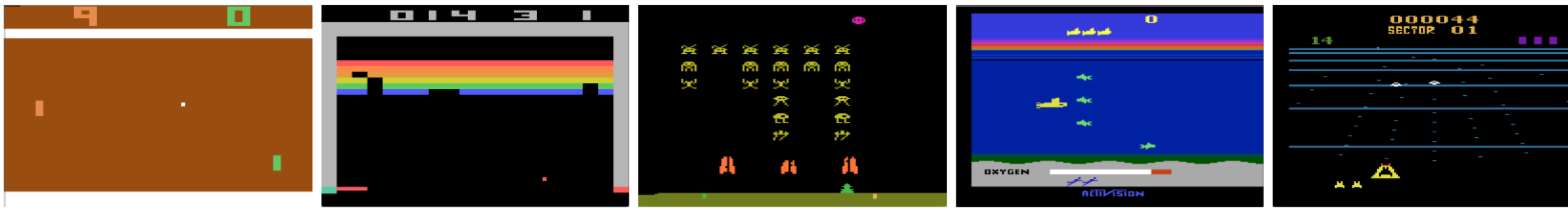
The human error rate is probably 20 to 30 errors but nobody has had the patience to measure it.

Read of the US checks!



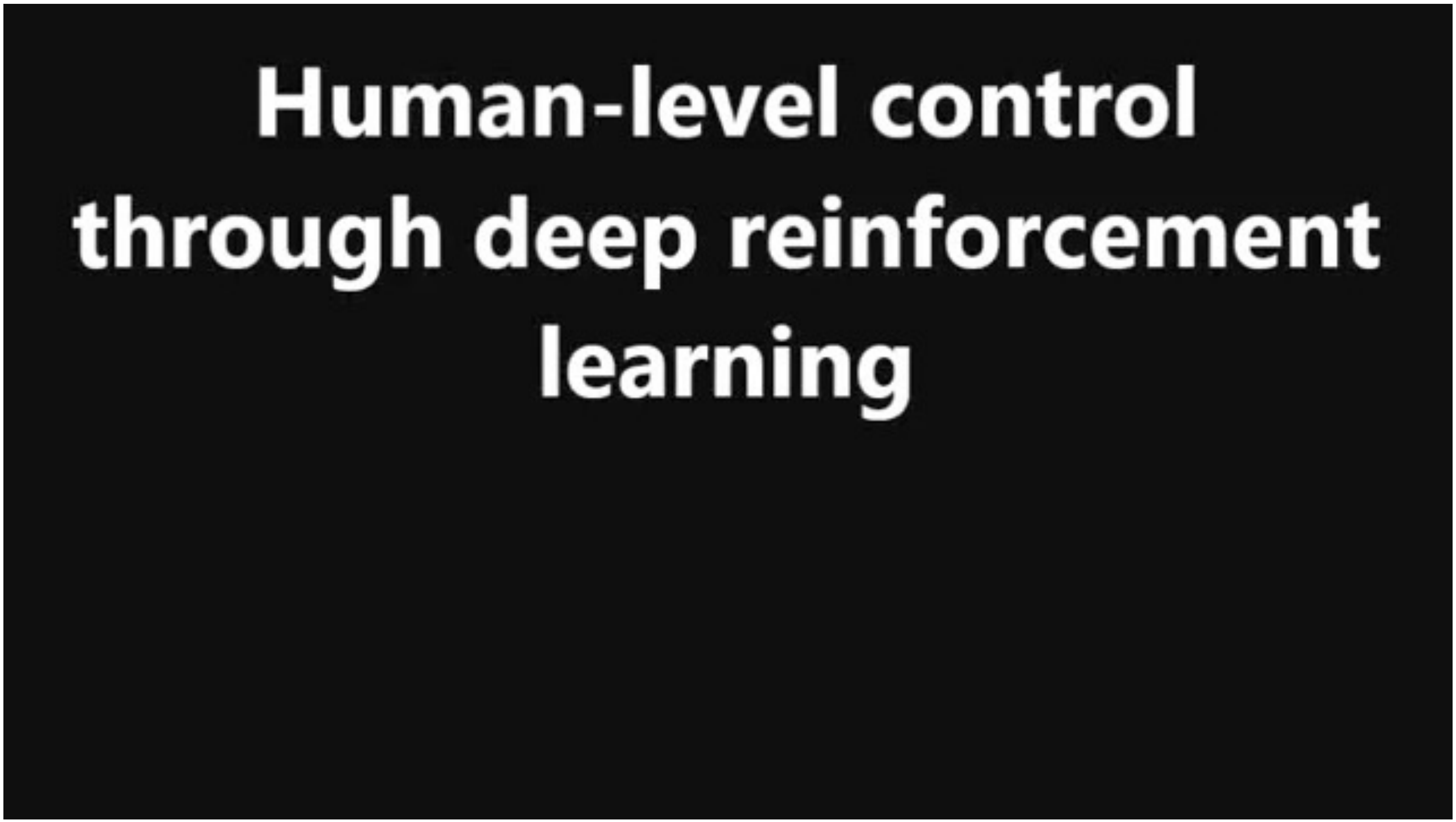
- LeNet5
- Invented by Prof Yann LeCun from NYU-Courant

DeepMind (now part of Google)

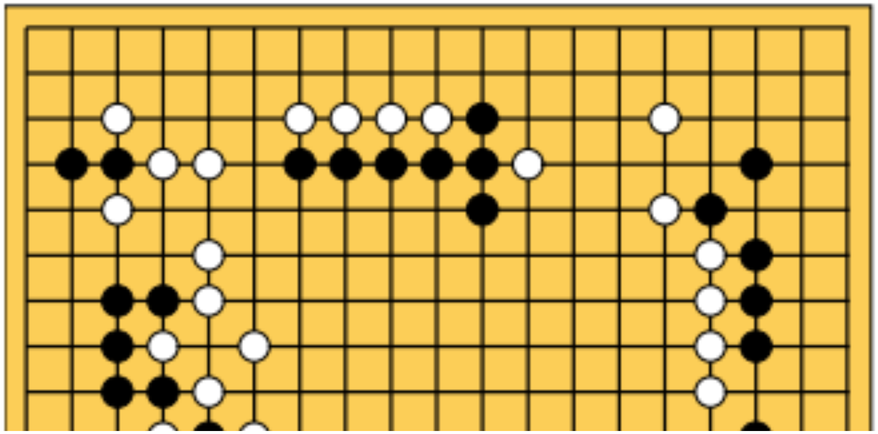


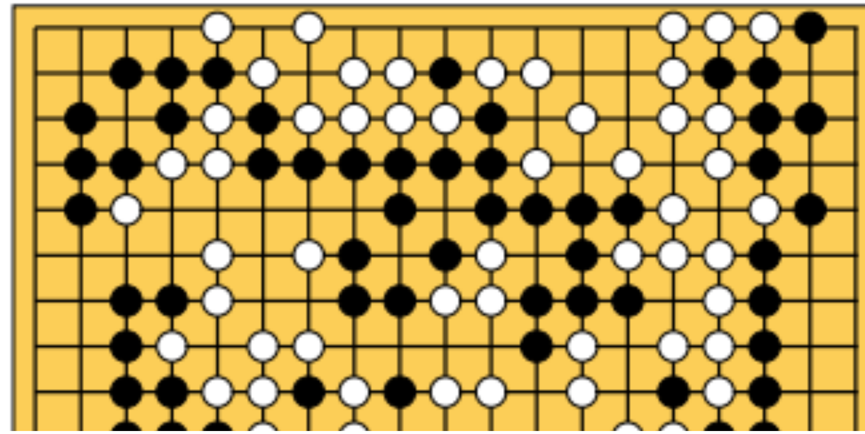
- The latest AI work before the much publicized 650M$ acquisition
  - Learns to play Atari 2600 games by playing it
  - GPU hardware are making it possible
- Deep reinforcement learning
  - Deep net maps input image to action-value functions
  - Reinforcement learning proposes the action as teaching labels

Human-level Control by Deep Learning



Deep nets now plays Go



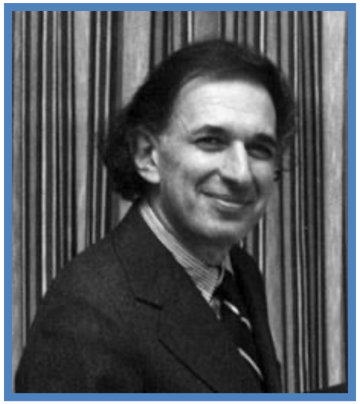


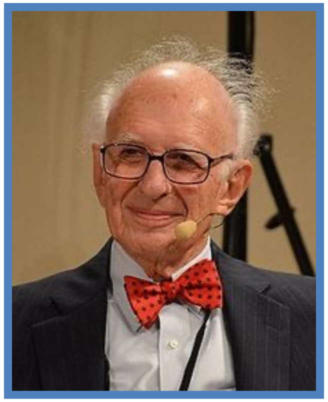
Network	Chinese Rules	Japanese Rules
KGS	0.86	0.85
GoGoD	0.87	0.91
GoGoD Small	0.71	0.67

Now wins over previous alternatives using searches

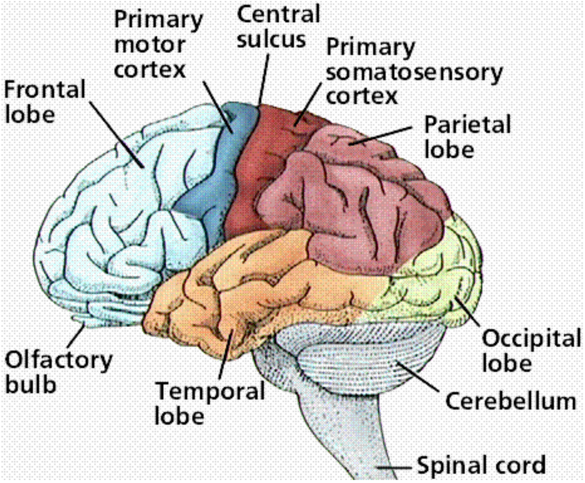
Eric Richard Kandel

- Eric Richard Kandel (German: [ˈkandəl]; born November 7, 1929) is an American neuropsychiatrist. He was a recipient of the ***2000 Nobel Prize in Physiology or Medicine*** for his research on the physiological basis of memory storage in neurons.
- Kandel, who had studied psychoanalysis, wanted to understand how memory works. His mentor, Harry Grundfest, said, “If you want to understand the brain you’re going to have to take a reductionist approach, one cell at a time.” So Kandel studied the ***neural system of the sea slug*** Aplysia californica, which has large nerve cells amenable to experimental manipulation and is a member of the simplest group of animals known to be capable of learning.
- Kandel is a professor of biochemistry and biophysics at the ***College of Physicians and Surgeons at Columbia University***. He is a Senior Investigator in the Howard Hughes Medical Institute.

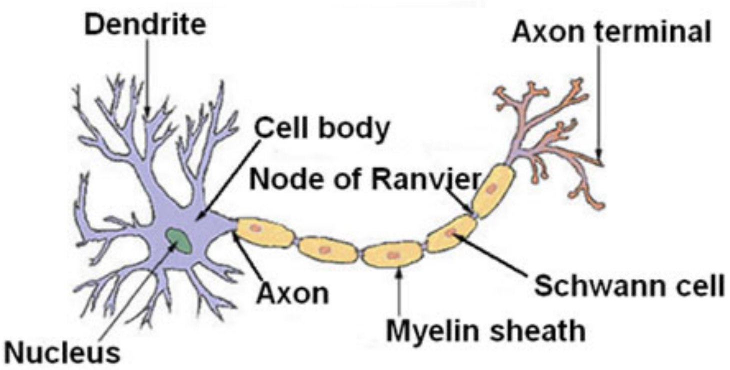


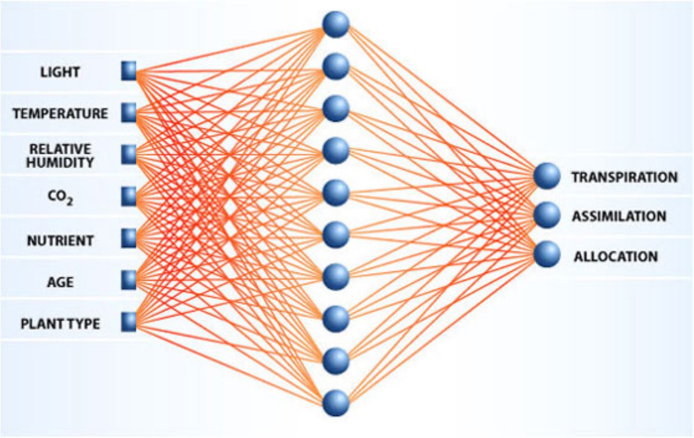


Artificial neural systems

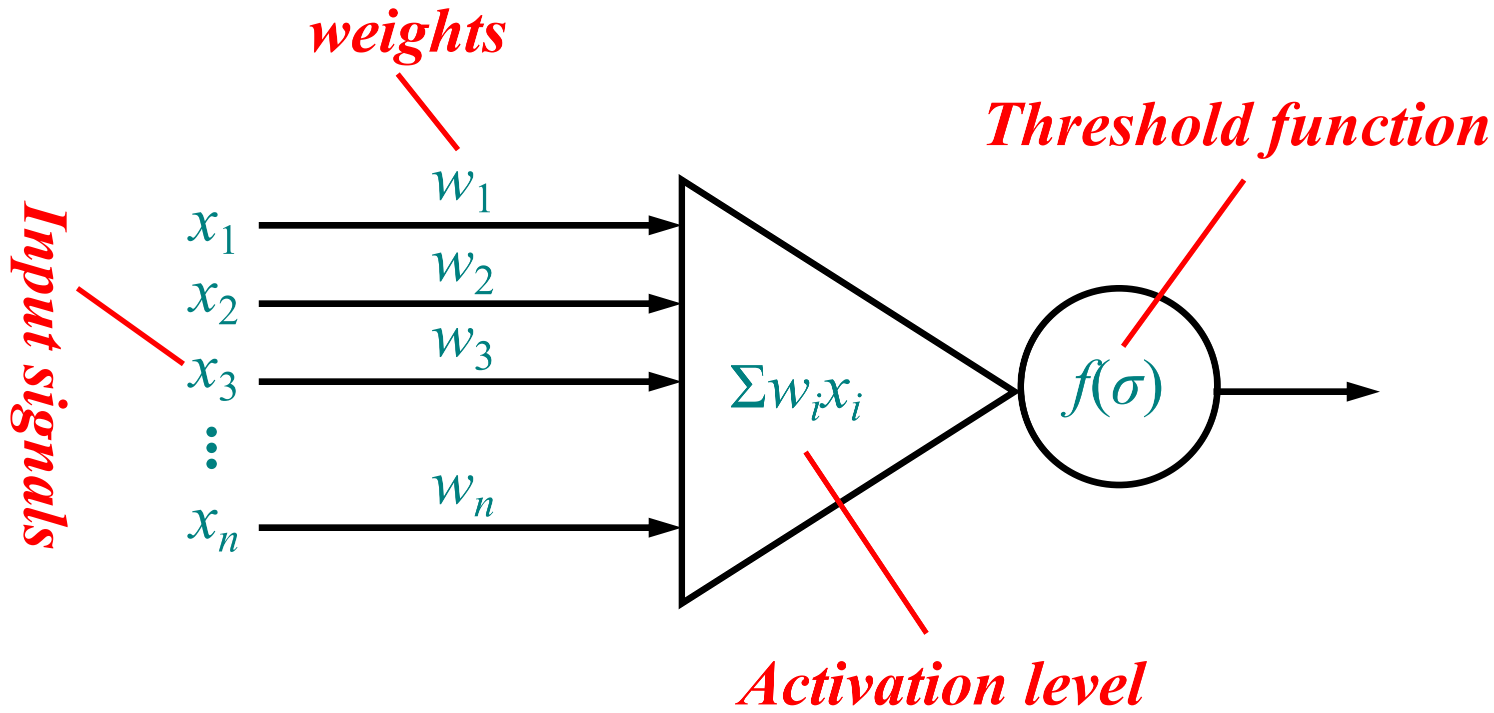


Structure of a Typical Neuron

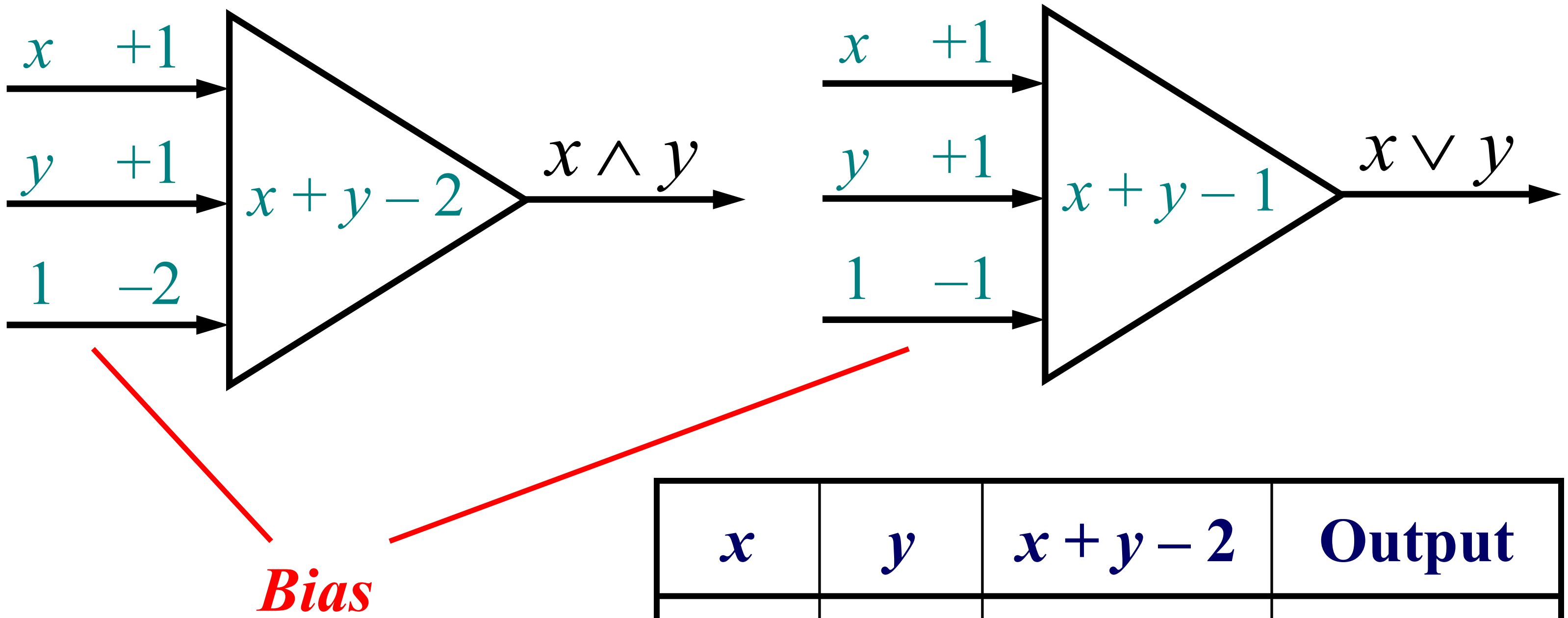




Artificial neuron



McCulloch-Pitts neuron (1949)



x	y	$x+y-2$	**Output**
1	1	0	1
1	0	–1	–1
0	1	–1	–1
0	0	–2	–1

Frank Rosenblatt Perceptron (1958)

- Given input values x_i, weights w_i, and a threshold t, the perceptron computers its ***output value*** as:

  $1 \quad \text{if } \Sigma w_i x_i \geq t$

  $-1 \quad \text{if } \Sigma w_i x_i < t$

- The ***adjustment*** for the weight on the ith component of the input vector.

  $\Delta w_i = r(d - \text{sign}(\Sigma w_i x_i))x_i$

  The $\text{sign}(\Sigma w_i x_i)$ is the perceptron output value. It is $+1$ or -1.

Frank Rosenblatt Perceptron (1958)

- Therefore for each component of the input vector.
  - If the desired output and actual output values are equal, ***do nothing***.
  - If the actual output value is -1 and should be 1, ***increment*** the weights on the ith line by $2rx_i$.
  - If the actual output value is 1 and should be -1, ***decrement*** the weights on the ith line by $2rx_i$.

Perceptron network to classify

x_1	x_2	**Output**
1.0	1.0	1
9.4	6.4	–1
2.5	2.1	1
8.0	7.7	–1
0.5	2.2	1
7.9	8.4	–1
7.0	7.0	–1
2.8	0.8	1
1.2	3.0	1
7.8	6.1	–1

Learning constant be 0.2, and
Random initialization: [0.75, 0.5, –0.6]

$$f(\sigma)^1 = f(0.75 \times 1 + 0.5 \times 1 - 0.6 \times 1) = f(0.65) = 1$$

Do nothing.

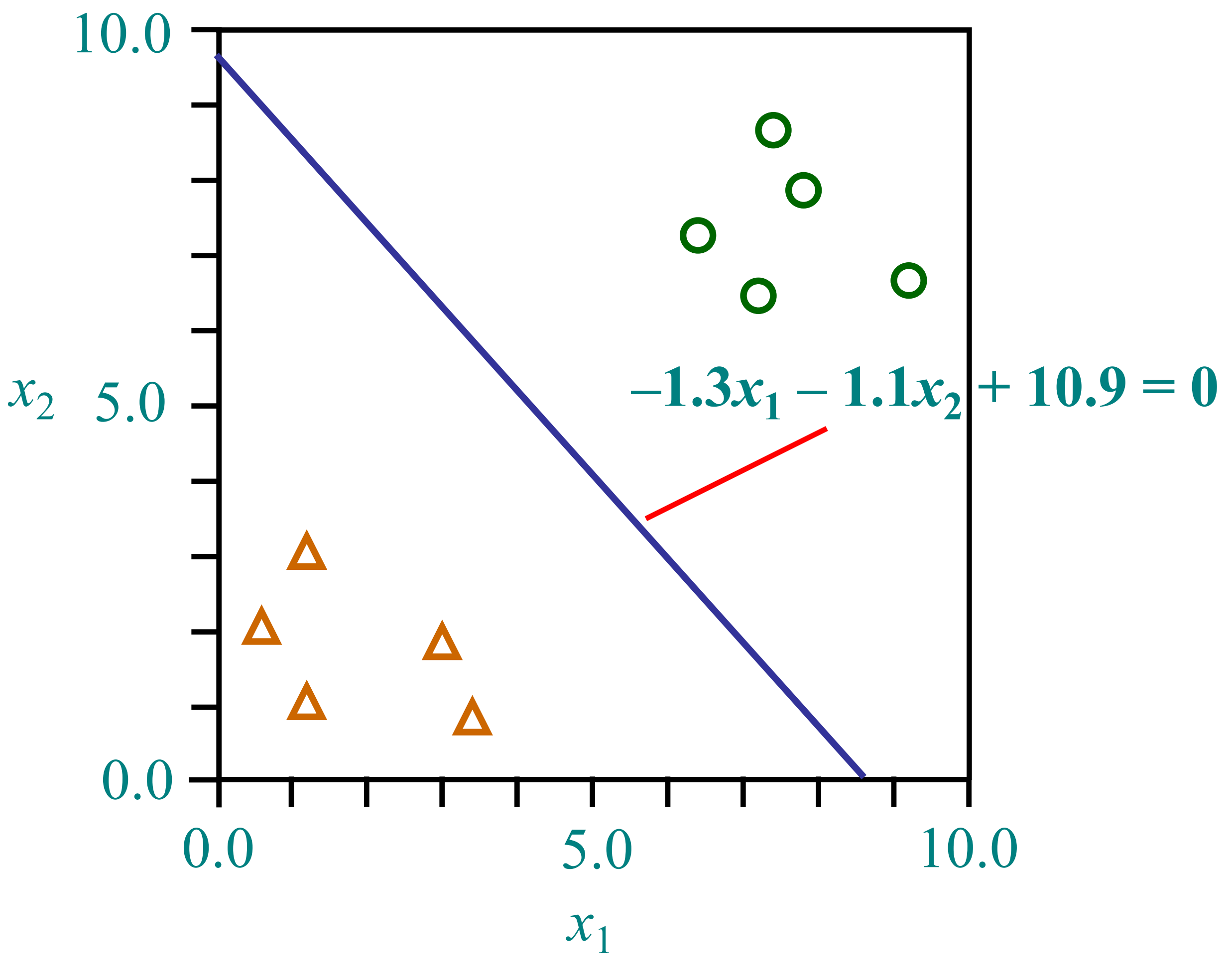
$$f(\sigma)^2 = f(0.75 \times 9.4 + 0.5 \times 6.4 - 0.6 \times 1) = f(9.65) = 1$$

Adjustment.

$$W^3 = W^2 + 0.2(-1 - 1)x^2 = [0.75, 0.50, -0.6] - 0.4[9.4, 6.4, 1.0] = [-3.01, -2.06, -1.00]$$

After about 500 iterations, the weight vector converges to [–1.3, –1.1, 10.9]

Perceptron network to classify



Limitations of the perceptron

x_1	x_2	**Output**
1	1	0
1	0	1
0	1	1
0	0	0

Truth table for exclusive-or

$w_1 \times 1 + w_2 \times 1 < t$, from line **1**

$w_1 \times 1 + w_2 \times 0 > t$, from line **2**

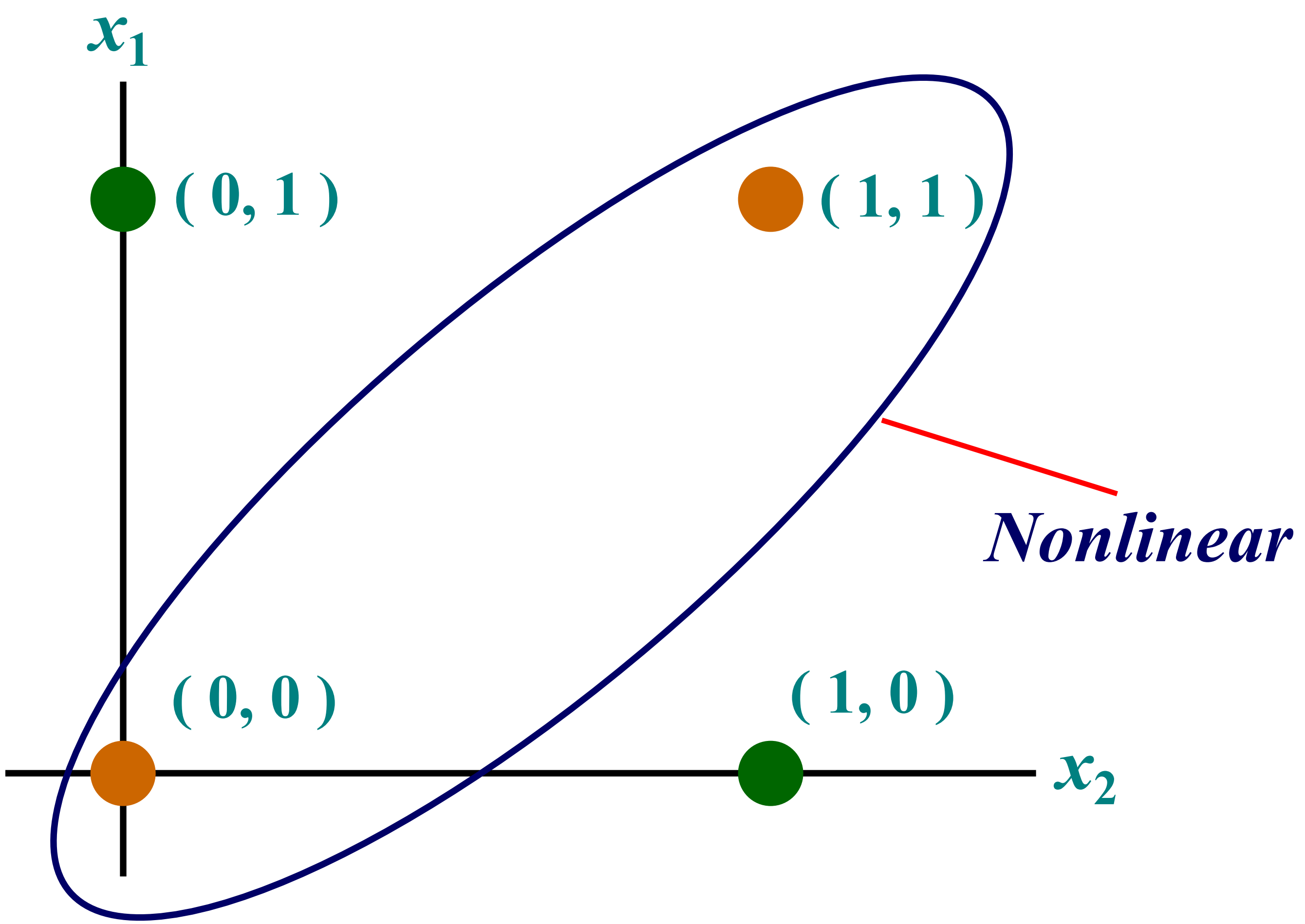
$w_1 \times 0 + w_2 \times 1 > t$, from line **3**

$w_1 \times 0 + w_2 \times 0 < t$, from line **4**

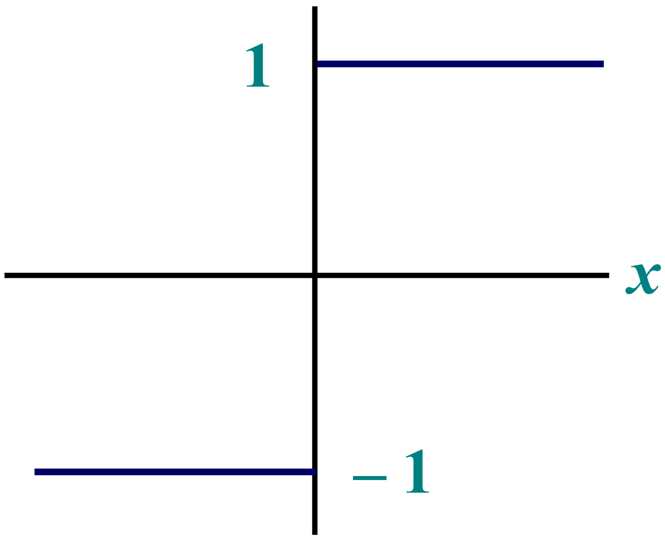
where t is threshold

This series of equations on w_1, w_2, and t has no solution, proving that a ***perceptron*** that solves exclusive-or is ***impossible***.

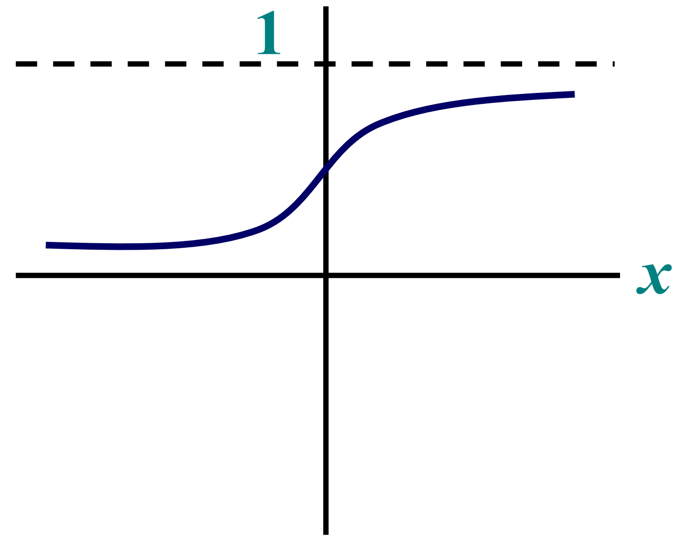
Limitations of the perceptron



Sigmoidal function



A hard limiting and bipolar linear threshold

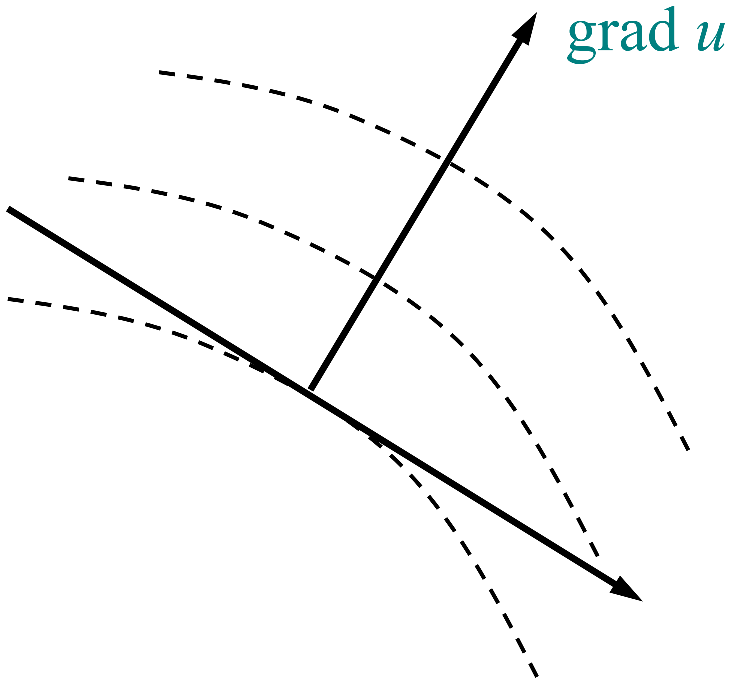


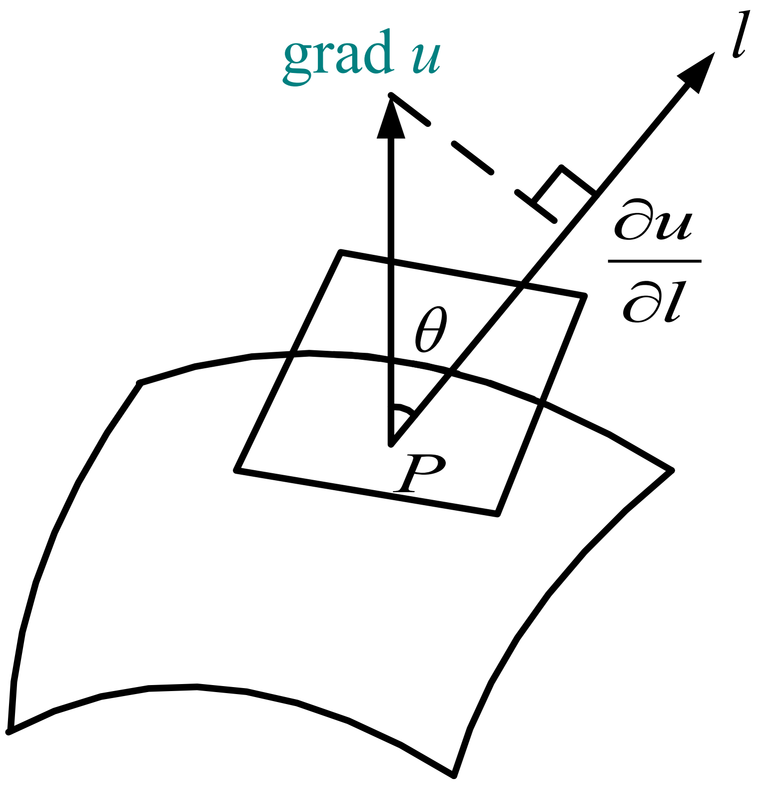
A sigmiodal and unipolar threshold

$f(\sigma) = 1 \,/\, (1 + e^{-\sigma})$, where $\sigma = \Sigma w_i x_i$

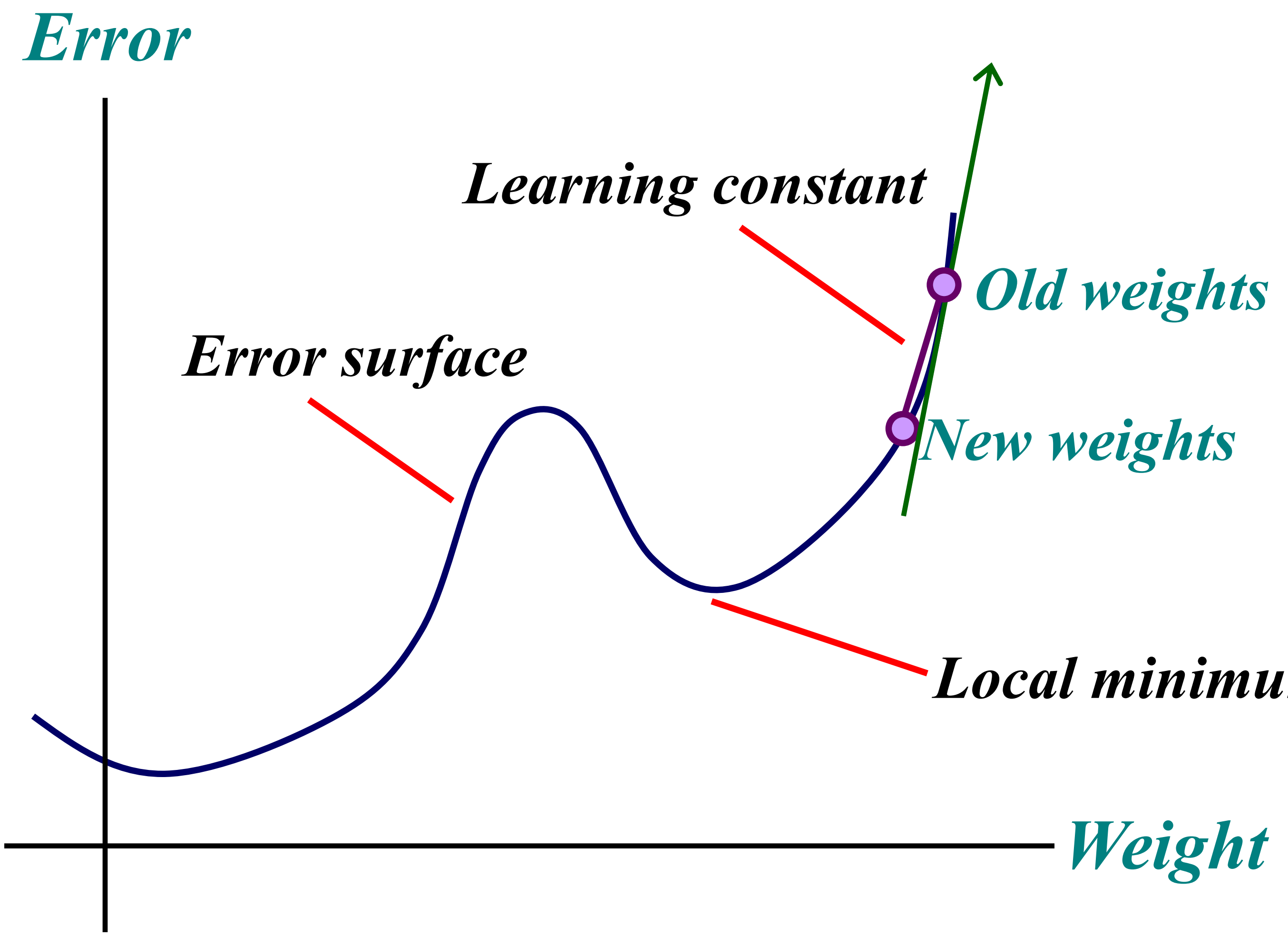
$f'(\sigma) = f(\sigma)(1 - f(\sigma))$

Gradient





Gradient descent learning



The mean squared network error

$$Error = \frac{1}{2}\sum_i (d_i - O_i)^2$$

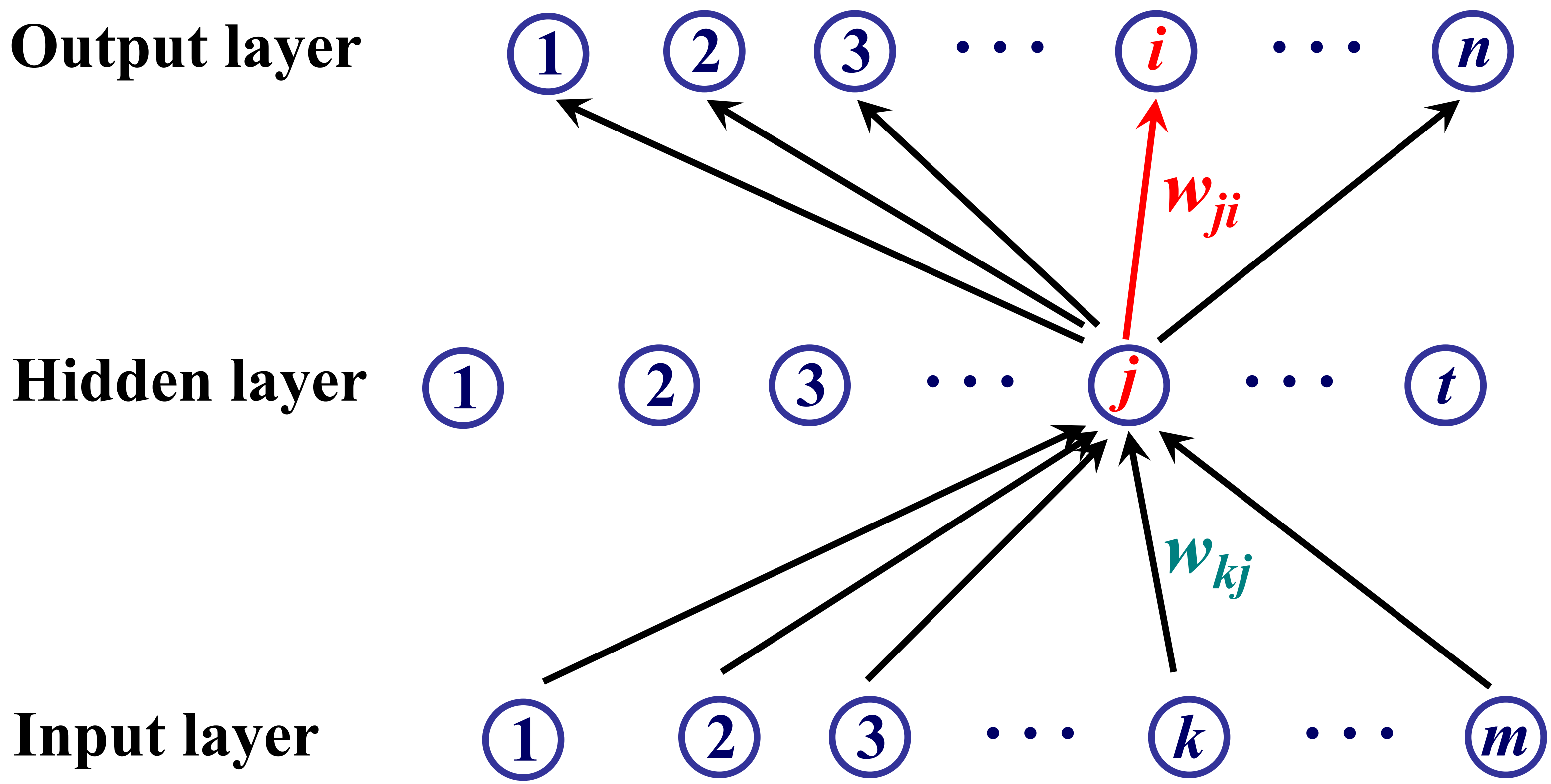
d_i is desired value for each output node

O_i is the actual output of the node.

$$\frac{\partial Error}{\partial O_i} = \frac{\partial (1/2)\sum_i (d_i - O_i)^2}{\partial O_i}$$

$$= \frac{\partial (1/2)(d_i - O_i)^2}{\partial O_i} = -(d_i - O_i)$$

Backpropagation learning



$$\Delta w_{ji} = r(d_i - O_i)O_i(1 - O_i)O_j$$

Chain rule for partial derivatives

$$\frac{\partial Error}{\partial w_j} = \frac{\partial Error}{\partial O_i} \cdot \frac{\partial O_i}{\partial w_j}$$

$$= -(d_i - O_i) \cdot \frac{\partial O_i}{\partial w_j}$$

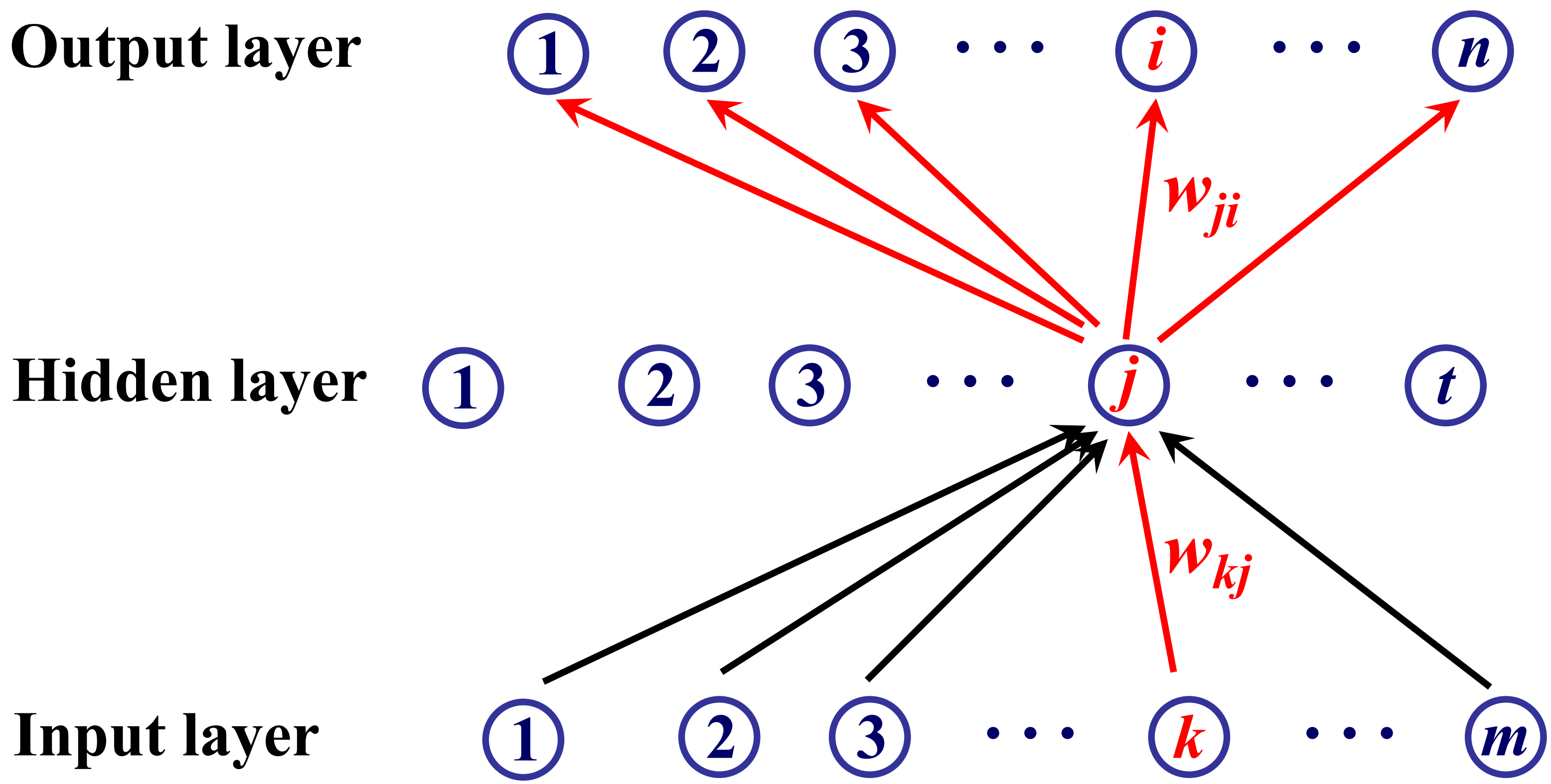
$O_i = f(\sigma)$, where $\sigma = \sum_k w_k x_k$

$$\frac{\partial O_i}{\partial w_j} = f'(\sigma) x_j = f(\sigma)(1 - f(\sigma)) x_j = O_i(1 - O_i) O_j$$

$$\Delta w_j = r(d_i - O_i) O_i (1 - O_i) O_j$$

The ***minimization*** of the error requires that that weight changes be in the direction of the ***negative gradient*** component.

Backpropagation learning



$$\Delta w_{kj} = rO_j(1-O_j)O_k\sum_i w_{ji}(O_i(1-O_i))\cdot(d_i-O_i)$$

Backpropagation learning

$$\frac{\partial Error}{\partial w_{kj}} = \sum_i \frac{\partial Error}{\partial O_i} \cdot \frac{\partial O_i}{\partial O_j} \cdot \frac{\partial O_j}{\partial w_{kj}}$$

$$= \sum_i \frac{\partial Error}{\partial O_i} \cdot \frac{\partial O_i}{\partial O_j} \cdot O_j(1 - O_j)O_k$$

$$= O_j(1 - O_j)O_k \sum_i w_{ji}(O_i(1 - O_i)) \cdot \frac{\partial Error}{\partial O_i}$$

$$= O_j(1 - O_j)O_k \sum_i w_{ji}(O_i(1 - O_i)) \cdot -(d_i - O_i)$$

$$\Delta w_{kj} = -rO_j(1 - O_j)O_k \sum_i w_{ji}(O_i(1 - O_i)) \cdot -(d_i - O_i)$$

$$= rO_j(1 - O_j)O_k \sum_i w_{ji}(O_i(1 - O_i)) \cdot (d_i - O_i)$$

NETtalk

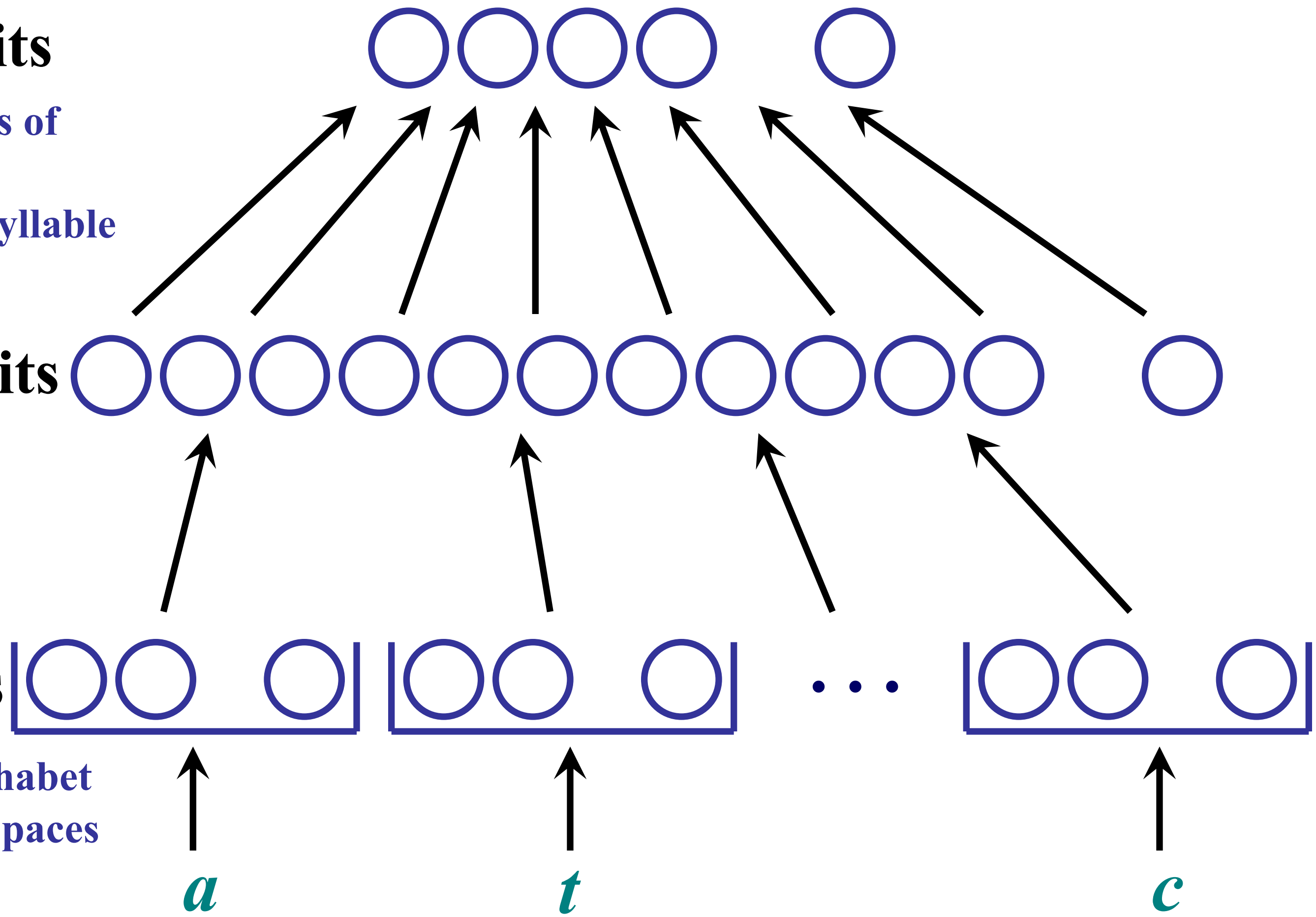
26 output units

21 different features of human articulation
5 encode stress an syllable boundaries

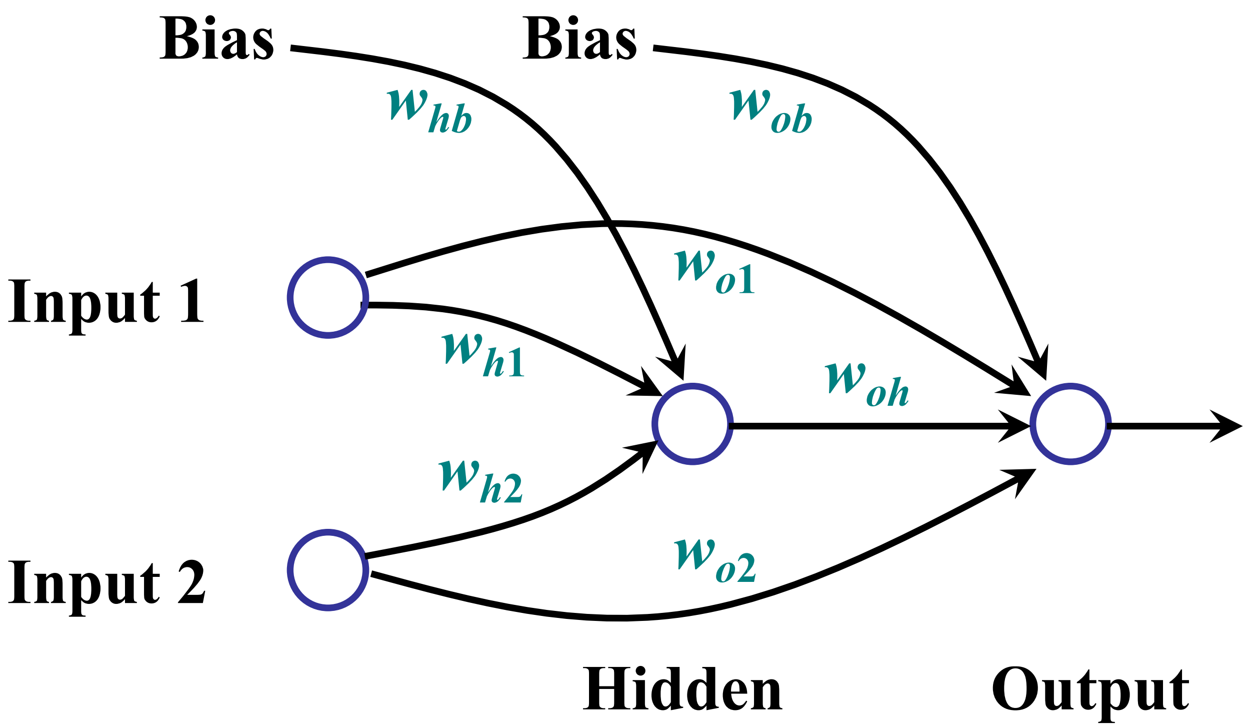
80 hidden units

7 × 29 inputs

26 letters of the alphabet
3 punctuation and spaces



Exclusive-or problem

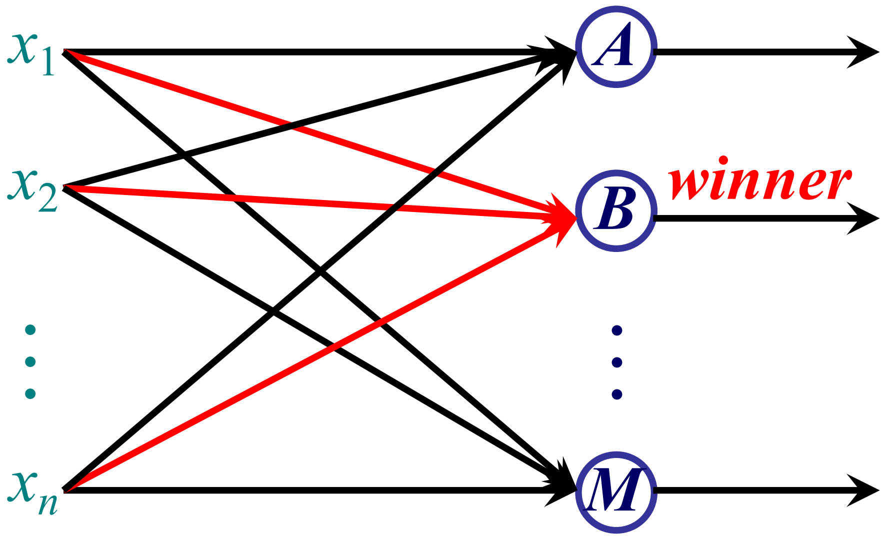


$w_{h1} = -7.0$ $w_{hb} = 2.6$ $w_{o1} = -5.0$ $w_{oh} = -11.0$

$w_{h2} = -7.0$ $w_{ob} = 7.0$ $w_{o2} = -4.0$

Winner-take-all algorithm

A vector of input values $X = (x_1, x_2, ..., x_n)$



Winner-take-all algorithm

Euclidean distance

$$\| X - W \| = \sqrt{(x_i - w_i)^2}$$

Learning

$$\Delta W^t = c(X^{t-1} - W^{t-1})$$

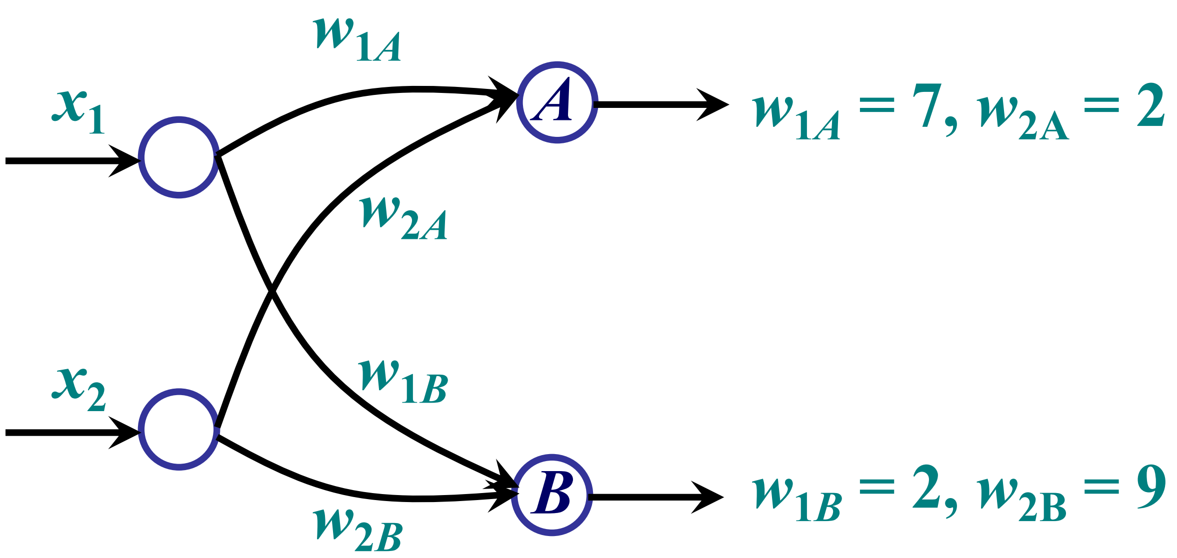
Learning for winner-take-all is unsupervised in that the winner is determined by a ***maximum activation*** *test. The weight vector of the winner is rewarded by bringing its components* ***closer*** *to those of the input vector.*

Set of data

x_1	x_2	Output
1.0	1.0	
9.4	6.4	
2.5	2.1	
8.0	7.7	
0.5	2.2	
7.9	8.4	
7.0	7.0	
2.8	0.8	
1.2	3.0	
7.8	6.1	

Unsupervised learning

Kohonen based learning network



Winner-take-all algorithm to classify

For point $(1, 1)$

$\|(1, 1) - (7, 2)\| = (1 - 7)^2 + (1 - 2)^2 = 37$, and
$\|(1, 1) - (2, 9)\| = (1 - 2)^2 + (1 - 9)^2 = 65$

Node A **is the winner.**

$$W^2 = W^1 + c(X^1 - W^1)$$
$$= (7, 2) + 0.5((1, 1) - (7, 2))$$
$$= (4, 1.5)$$

Winner-take-all algorithm to classify

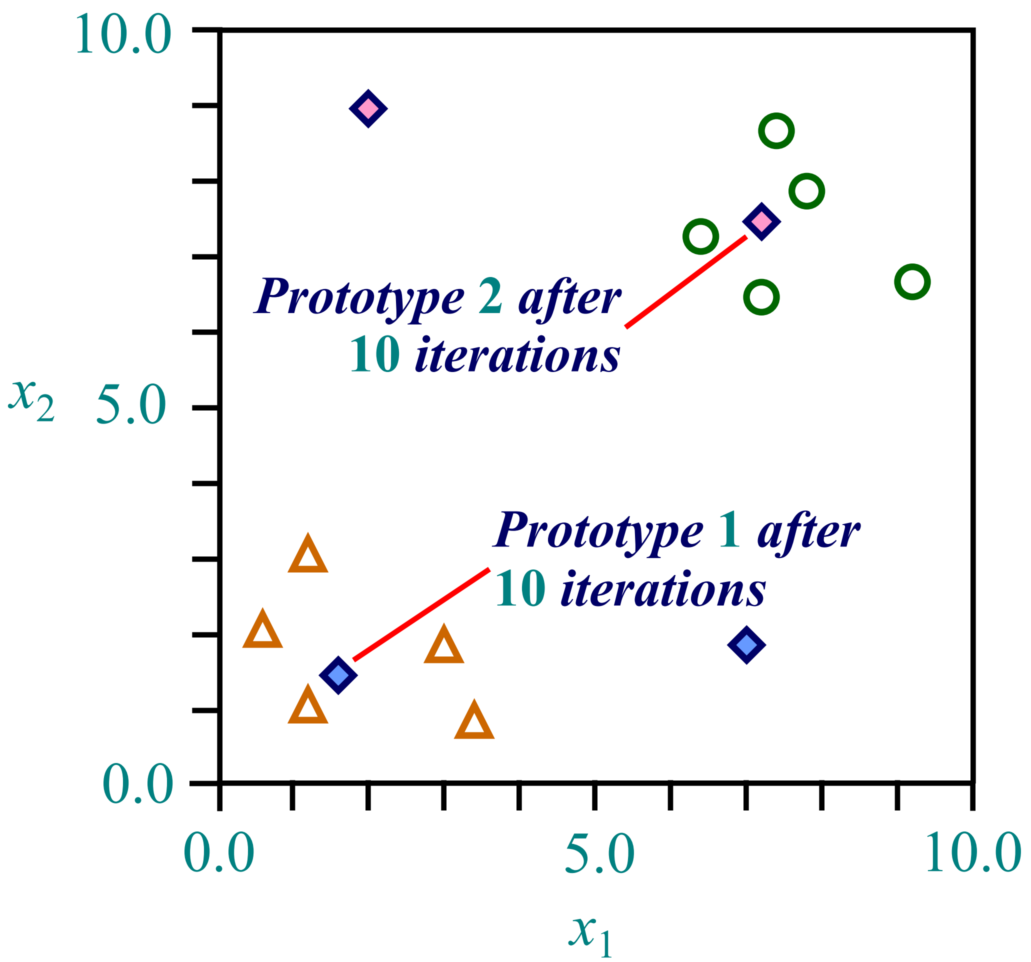
For point $(9.4, 6.4)$

$\|(9.4, 6.4) - (4, 1.5)\| = (9.4 - 4)^2 + (6.4 - 1.5)^2 = 53.17$, and
$\|(9.4, 6.4) - (2, 9)\| = (9.4 - 2)^2 + (6.4 - 9)^2 = 60.15$

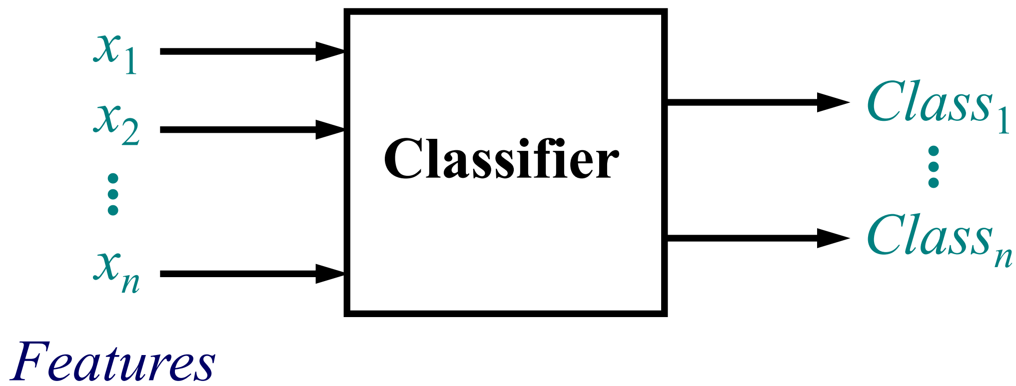
Again, node A **is the winner.**

$$W^3 = W^2 + c(X^2 - W^2)$$
$$= (4, 1.5) + 0.5((9.4, 6.4) - (4, 1.5))$$
$$= (6.7, 4)$$

Winner-take-all algorithm to classify



Full classification system



Tasks

- ***Classification***: deciding the category or grouping to which an input value belongs;
- ***Pattern recognition***: identifying structure or pattern in data;
- ***Memory recall***: including the problem of content addressable memory;
- ***Prediction***: such as identifying disease from symptoms, causes from effects;
- ***Optimization***: finding the "best" organization of constraints; and
- ***Noise filtering***: or separating signal from background, factoring out the irrelevant components of a signal.

Any question?

Xiaoqing Zheng
Fudan University