Informatics 1 Cognitive Science

Lecture 5: The Perceptron

Frank Keller

23 January 2025

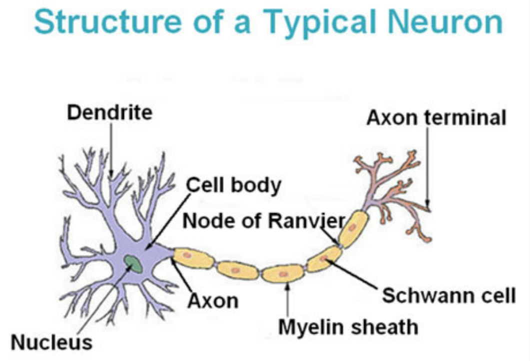
School of Informatics
University of Edinburgh
keller@inf.ed.ac.uk

Slide credits: Frank Mollica, Chris Lucas, Mirella Lapata

Overview

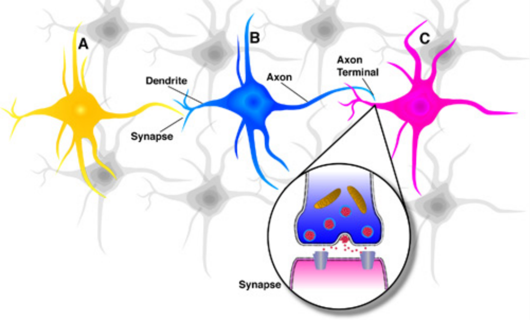
Neural Networks

A Single Neuron



- A neuron receives inputs and combines these in the cell body.
- If the input reaches a threshold, then the neuron may fire (produce an output).
- Some inputs are excitatory, while others are inhibitory.

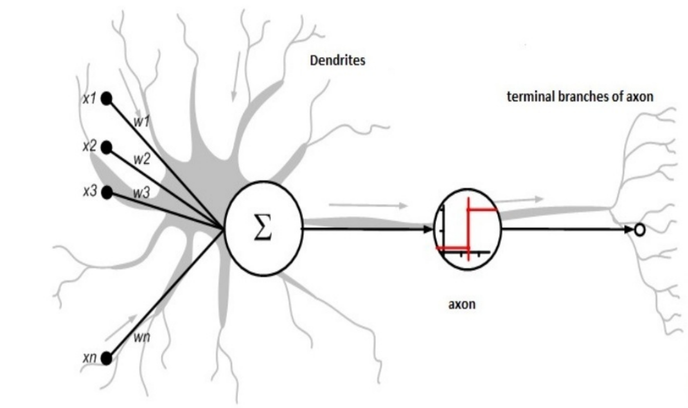
Biological Neural Networks



- In biological neural networks, neurons are connected by synapses.
- An input connection is a conduit through which a neuron receives information.
- An output connection is a conduit through which a neural sends information.

Neural Networks

Neural networks (aka deep learning) is a computer modeling approach inspired by information processing in networks of biological neurons.



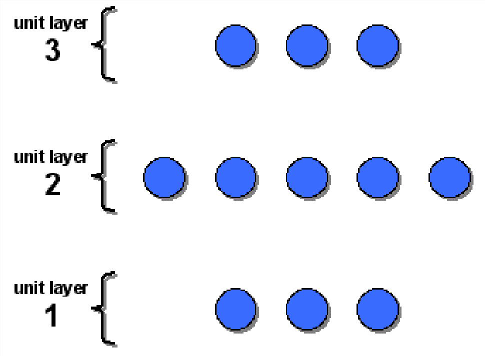
Anatomy of a Neural Network

Units are to a neural net model what neurons are to a biological neural network — basic information processing structures.

Anatomy of a Neural Network

Units are to a neural net model what neurons are to a biological neural network — basic information processing structures.

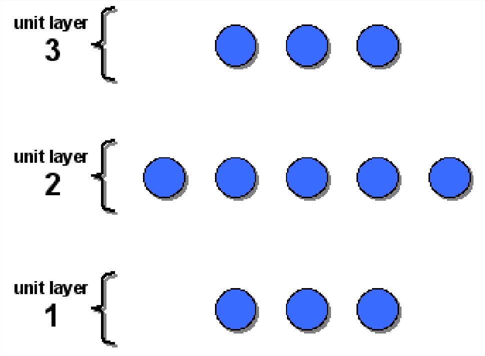
Biological neural networks are organized in layers of neurons. Neural net models are organized in layers of units, not random clusters.



Anatomy of a Neural Network

Units are to a neural net model what neurons are to a biological neural network — basic information processing structures.

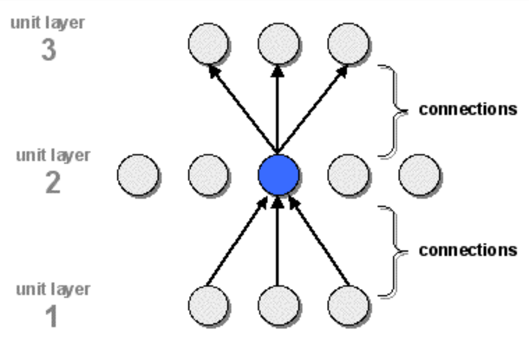
Biological neural networks are organized in layers of neurons. Neural net models are organized in layers of units, not random clusters.



But what you see here still isn't a network. Something is missing.

Anatomy of a Neural Network

Network connections are conduits through which information flows between the units in a network:

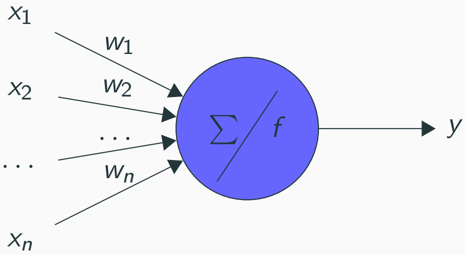


- Connections are represented with lines
- Arrows in a neural net indicate the flow of information from one unit to the next.

The Perceptron

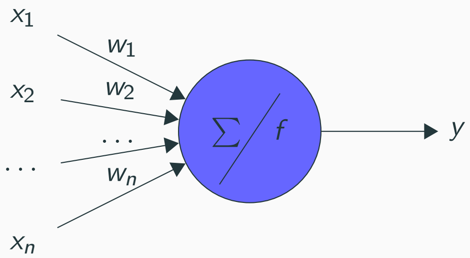
Perceptron: An Artificial Neuron

The perceptron was developed by Frank Rosenblatt in 1957. It's the simplest kind of artificial neural network.



Perceptron: An Artificial Neuron

The perceptron was developed by Frank Rosenblatt in 1957. It's the simplest kind of artificial neural network.

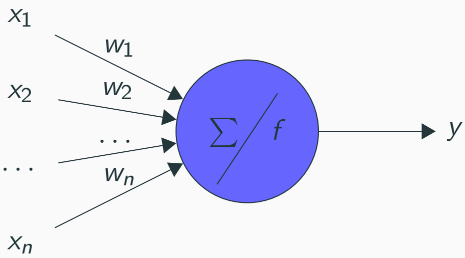


Input function:

$$u(x) = \sum_{i=1}^{n} w_i x_i$$

Perceptron: An Artificial Neuron

The perceptron was developed by Frank Rosenblatt in 1957. It's the simplest kind of artificial neural network.



Input function:

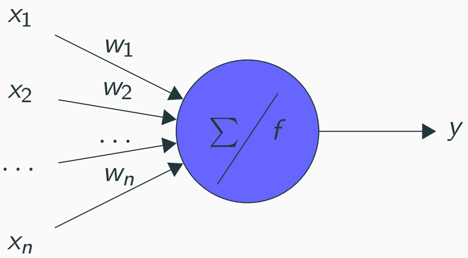
$$u(\mathrm{x}) = \sum_{i=1}^{n} w_i x_i$$

Activation function: threshold

$$y = f(u(\mathrm{x})) = \begin{cases} 1, & \text{if } u(\mathrm{x}) > \theta \\ 0, & \text{otherwise} \end{cases}$$

Perceptron: An Artificial Neuron

The perceptron was developed by Frank Rosenblatt in 1957. It's the simplest kind of artificial neural network.



Input function:

$$u(\mathrm{x}) = \sum_{i=1}^{n} w_i x_i$$

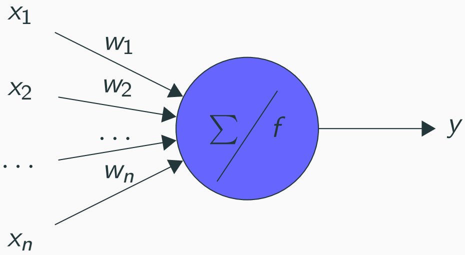
Activation function: threshold

$$y = f(u(\mathrm{x})) = \begin{cases} 1, & \text{if } u(\mathrm{x}) > \theta \\ 0, & \text{otherwise} \end{cases}$$

Activation state:
0 or 1 (-1 or 1)

Perceptron: An Artificial Neuron

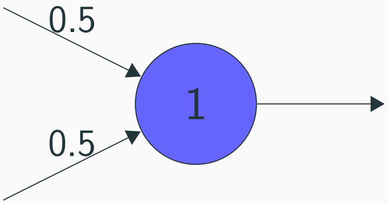
The perceptron was developed by Frank Rosenblatt in 1957. It's the simplest kind of artificial neural network.



- Inputs are in the range [0, 1], where 0 is "off" and 1 is "on".
- Weights can be any real number (positive or negative).

Perceptrons for Logic

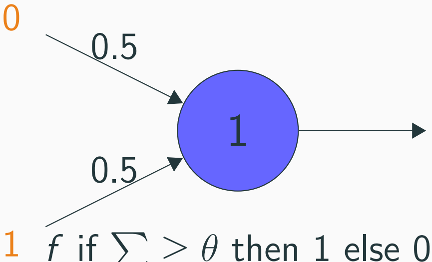
Perceptron for AND



f if $\sum \geq \theta$ then 1 else 0

input		output $y =$
x_1	x_2	x_1 AND x_2
0	0	0
0	1	0
1	0	0
1	1	1

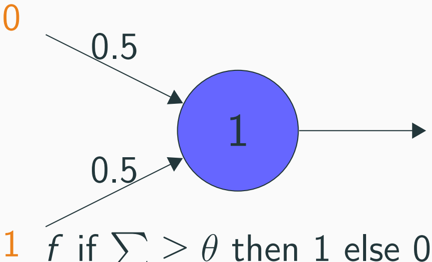
Perceptron for AND



f if $\sum \geq \theta$ then 1 else 0

input		output $y =$
x_1	x_2	x_1 AND x_2
0	0	0
0	1	0
1	0	0
1	1	1

Perceptron for AND

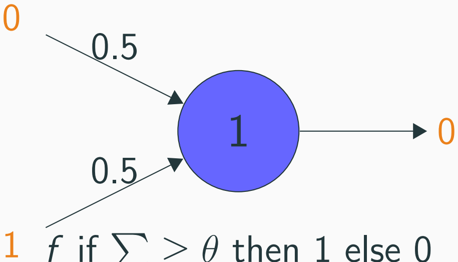


f if $\sum \geq \theta$ then 1 else 0

$0 \cdot 0.5 + 1 \cdot 0.5 = 0.5$

input x_1	input x_2	output $y = x_1$ AND x_2
0	0	0
0	1	0
1	0	0
1	1	1

Perceptron for AND



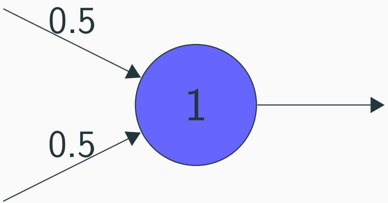
f if $\sum \geq \theta$ then 1 else 0

$$0 \cdot 0.5 + 1 \cdot 0.5 = 0.5$$

input		output $y =$
x_1	x_2	x_1 AND x_2
0	0	0
0	1	0
1	0	0
1	1	1

Perceptrons for Logic

Perceptron for AND

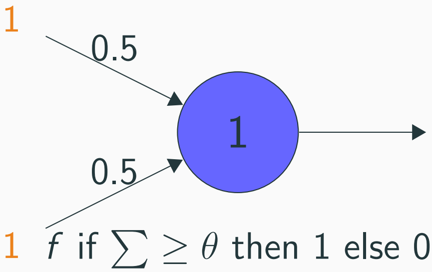


f if $\sum \geq \theta$ then 1 else 0

input		output $y =$
x_1	x_2	x_1 AND x_2
0	0	0
0	1	0
1	0	0
1	1	1

Perceptrons for Logic

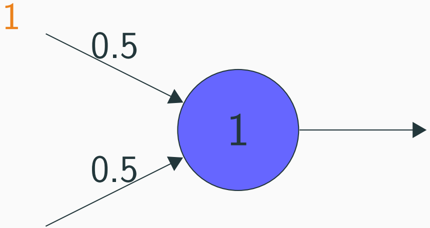
Perceptron for AND



input		output $y =$
x_1	x_2	x_1 AND x_2
0	0	0
0	1	0
1	0	0
1	1	1

Perceptrons for Logic

Perceptron for AND



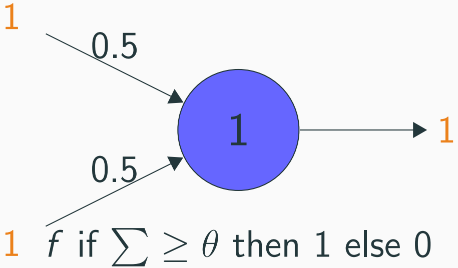
f if $\sum \geq \theta$ then 1 else 0

$$1 \cdot 0.5 + 1 \cdot 0.5 = 1$$

input		output $y =$
x_1	x_2	x_1 AND x_2
0	0	0
0	1	0
1	0	0
1	1	1

Perceptrons for Logic

Perceptron for AND

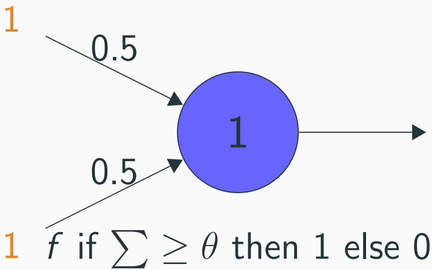


f if $\sum \geq \theta$ then 1 else 0

$$1 \cdot 0.5 + 1 \cdot 0.5 = 1$$

input x_1	input x_2	output $y = x_1$ AND x_2
0	0	0
0	1	0
1	0	0
1	1	1

Perceptron for AND



f if $\sum \geq \theta$ then 1 else 0

$$1 \cdot 0.5 + 1 \cdot 0.5 = 1$$

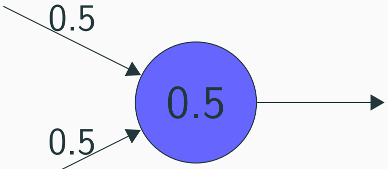
input		output $y =$
x_1	x_2	x_1 AND x_2
0	0	0
0	1	0
1	0	0
1	1	1

We can easily devise perceptrons that compute the logical functions AND and OR.

Can we compute all logical functions? What about XOR?

Perceptrons for Logic

Perceptron for XOR

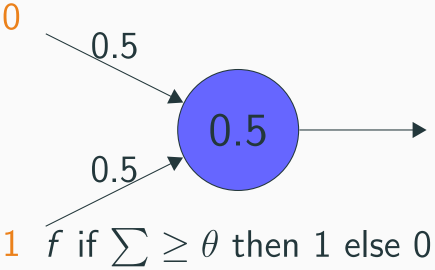


f if $\sum \geq \theta$ then 1 else 0

input x_1	x_2	output $y =$ x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

XOR is an exclusive OR because it only returns a true value of 1 if the two values are exclusive, i.e., they are both different.

Perceptron for XOR

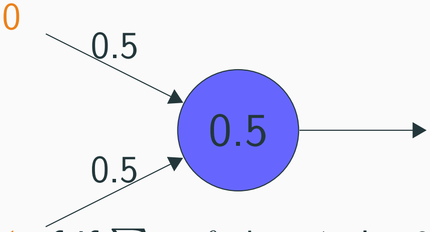


input		output $y =$
x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

XOR is an exclusive OR because it only returns a true value of 1 if the two values are exclusive, i.e., they are both different.

Perceptrons for Logic

Perceptron for XOR



f if $\sum \geq \theta$ then 1 else 0

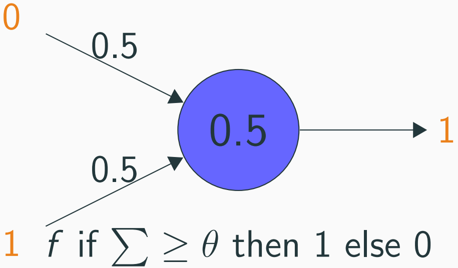
$$0 \cdot 0.5 + 1 \cdot 0.5 = 0.5$$

input x_1	input x_2	output $y = x_1$ XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

XOR is an exclusive OR because it only returns a true value of 1 if the two values are exclusive, i.e., they are both different.

Perceptrons for Logic

Perceptron for XOR



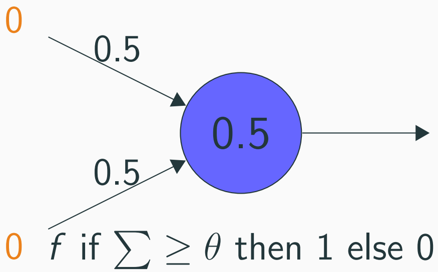
f if $\sum \geq \theta$ then 1 else 0

$$0 \cdot 0.5 + 1 \cdot 0.5 = 0.5$$

input		output $y =$
x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

XOR is an exclusive OR because it only returns a true value of 1 if the two values are exclusive, i.e., they are both different.

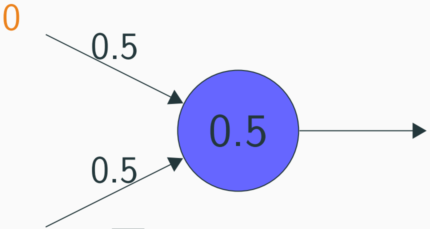
Perceptron for XOR



input		output $y =$
x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

Perceptrons for Logic

Perceptron for XOR



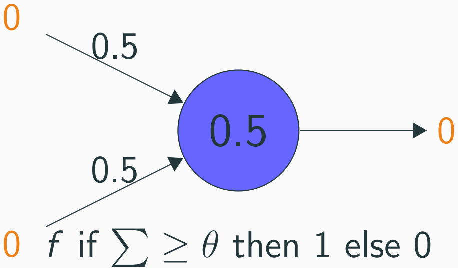
0 f if $\sum \geq \theta$ then 1 else 0

$$0 \cdot 0.5 + 0 \cdot 0.5 = 0$$

input		output $y =$
x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

Perceptrons for Logic

Perceptron for XOR

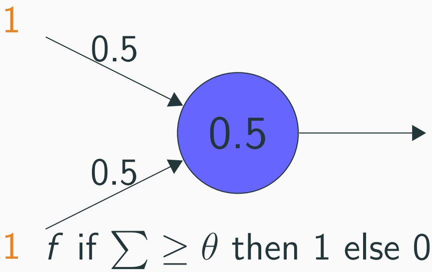


f if $\sum \geq \theta$ then 1 else 0

$0 \cdot 0.5 + 0 \cdot 0.5 = 0$

input		output $y =$
x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

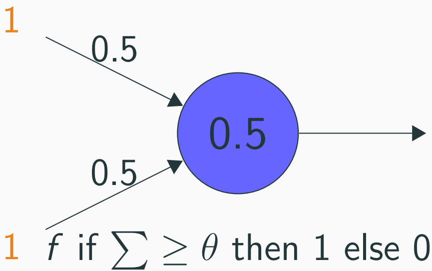
Perceptron for XOR



input x_1	input x_2	output $y = x_1$ XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

Perceptrons for Logic

Perceptron for XOR

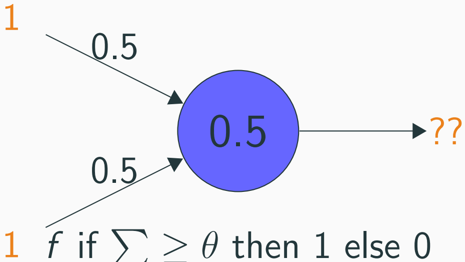


f if $\sum \geq \theta$ then 1 else 0

$$1 \cdot 0.5 + 1 \cdot 0.5 = 1$$

input		output $y =$
x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

Perceptron for XOR



f if $\sum \geq \theta$ then 1 else 0

$$1 \cdot 0.5 + 1 \cdot 0.5 = 1$$

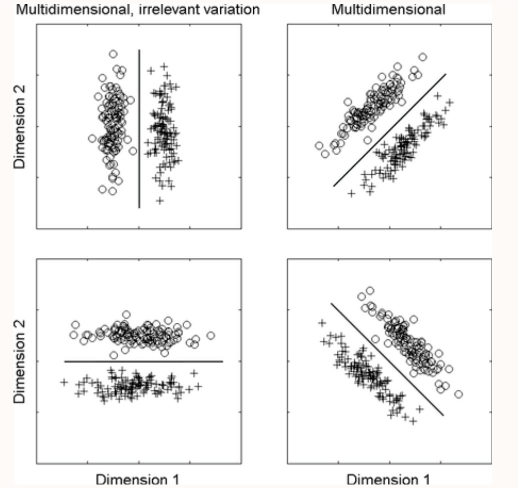
input		output $y =$
x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

Time for a short quiz on Wooclap!

https://app.wooclap.com/GEKKBD

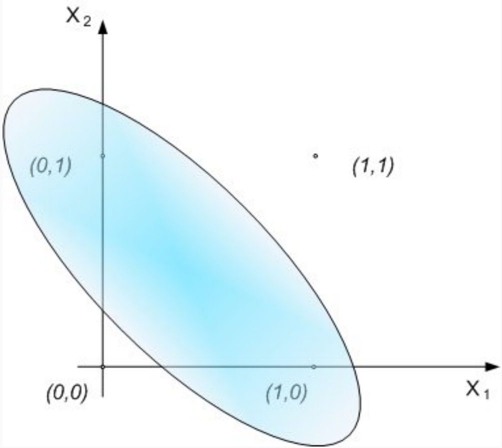
Perceptrons as Classifiers

Perceptrons are linear classifiers, i.e., they can only separate points with a hyperplane (a straight line in two dimensions).



The XOR problem again

The XOR function is not linearly separable, as more than one line is required to separate the two classes {(0,0), (1,1)} and {(0,1), (1,0)}. A single-layer perceptron cannot compute XOR.



Learning in Perceptrons

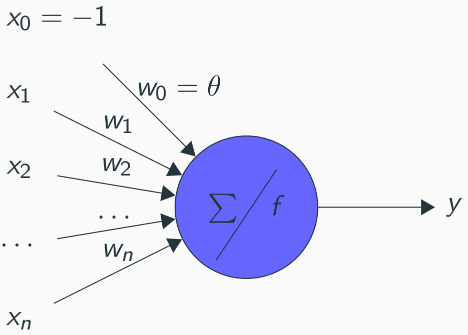
Learning

Q_1: But choosing weights and threshold θ for the perceptron is not easy! How to learn the weights and threshold from examples?

A_1: We can use a learning algorithm that adjusts the weights and threshold based on examples.

http://www.youtube.com/watch?v=vGwemZhPlsA&feature=youtu.be

Learning: A trick to learn θ



- We can consider θ as a weight to be learned!
- The input is fixed as -1. The activation function is then:

$$y = f(u(\mathrm{x})) = \begin{cases} 1, & \text{if } u(\mathrm{x}) > 0 \\ 0, & \text{otherwise} \end{cases}$$

What is the Perceptron Really Seeing?

Sequence of exemplars presented to the perceptron during training:

N	**input** x	**target** t
1	(0,1,0,0)	1
2	(1,0,0,0)	0
3	(0,1,1,1)	0
4	(1,0,1,0)	0
5	(1,1,1,1)	1
6	(0,1,0,0)	1

- This perceptron has 4 inputs (binary) $\approx$ feature vector representing exemplars
- The perceptron sees 6 exemplars or training items

What is the Perceptron Really Seeing?

Sequence of exemplars presented to the perceptron during training:

N	**input** x	**target** t
1	(0,1,0,0)	1
2	(1,0,0,0)	0
3	(0,1,1,1)	0
4	(1,0,1,0)	0
5	(1,1,1,1)	1
6	(0,1,0,0)	1

- This perceptron has 4 inputs (binary) $\approx$ feature vector representing exemplars
- The perceptron sees 6 exemplars or training items
- We don't know the weights/threshold!

What is the Perceptron Really Seeing?

Sequence of exemplars presented to the perceptron during training:

N	**input** x	**target** t	**output** o
1	(0,1,0,0)	1	0
2	(1,0,0,0)	0	0
3	(0,1,1,1)	0	1
4	(1,0,1,0)	0	1
5	(1,1,1,1)	1	0
6	(0,1,0,0)	1	1

- This perceptron has 4 inputs (binary) $\approx$ feature vector representing exemplars
- The perceptron sees 6 exemplars or training items
- We don't know the weights/threshold!
- But we know the perceptron's output o and can compare it to the correct answer, the target t

The Perceptron Learning Rule

Learning Rule

Key idea: Adjust the weights so that o (the output of the perceptron) moves closer to t (the target, i.e., the desired correct output):

Perceptron Learning Rule

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i = \eta(t - o)x_i$$

- η, $0 < \eta \leq 1$ is a constant called learning rate.
- t is the target for the current example.
- o is the perceptron output for the current example.

Learning Rule

Perceptron Learning Rule

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i = \eta(t - o)x_i$$

$$o = 1 \text{ and } t = 1$$
$$o = 0 \text{ and } t = 1$$

- Learning rate η is positive; controls how big changes Δw_i are.
- If $x_i > 0$, $\Delta w_i > 0$ then w_i increases in an attempt to make $w_i x_i$ become larger than θ.
- If $x_i < 0$, $\Delta w_i < 0$ then w_i reduces.

Perceptron Learning Rule

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i \ = \eta(t - o)x_i$$

$$\begin{aligned} o = 1 \text{ and } t = 1 \quad \Delta w_i \ &= \eta(t - o)x_i = \eta(1 - 1)x_i = 0 \\ o = 0 \text{ and } t = 1 \quad \Delta w_i \ &= \eta(t - o)x_i = \eta(1 - 0)x_i = \eta x_i \end{aligned}$$

- Learning rate η is positive; controls how big changes Δw_i are.
- If $x_i > 0$, $\Delta w_i > 0$ then w_i increases in an attempt to make $w_i x_i$ become larger than θ.
- If $x_i < 0$, $\Delta w_i < 0$ then w_i reduces.

Perceptrons for Logic

Time for a short quiz on Wooclap!

https://app.wooclap.com/GEKKBD

Learning Rule: Exercise

Perceptron Learning Rule

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i \; = \eta(t - o)x_i$$

Consider a perceptron with only one input x_1, weight $w_1 = 0.5$, threshold $\theta = 0$ and learning rate $\eta = 0.6$. Consider also the training example $\{x_1 = -1, t = 1\}$. For now, let's temporarily ignore the learning of the threshold and consider it fixed.

Learning Rule: Exercise

Perceptron Learning Rule

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i \ = \eta(t - o)x_i$$

Consider a perceptron with only one input x_1, weight $w_1 = 0.5$, threshold $\theta = 0$ and learning rate $\eta = 0.6$. Consider also the training example $\{x_1 = -1, t = 1\}$. For now, let's temporarily ignore the learning of the threshold and consider it fixed.

- Determine the output of the perceptron for the input -1:

Learning Rule: Exercise

Perceptron Learning Rule

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i = \eta(t - o)x_i$$

Consider a perceptron with only one input x_1, weight $w_1 = 0.5$, threshold $\theta = 0$ and learning rate $\eta = 0.6$. Consider also the training example $\{x_1 = -1, t = 1\}$. For now, let's temporarily ignore the learning of the threshold and consider it fixed.

- Determine the output of the perceptron for the input -1:
  $w_1 x_1 = 0.5(-1) = -0.5 \leq \theta \rightarrow o = 0$
- The new weight w_1 after applying the learning rule:

Learning Rule: Exercise

Perceptron Learning Rule

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i \ = \eta(t - o)x_i$$

Consider a perceptron with only one input x_1, weight $w_1 = 0.5$, threshold $\theta = 0$ and learning rate $\eta = 0.6$. Consider also the training example $\{x_1 = -1, t = 1\}$. For now, let's temporarily ignore the learning of the threshold and consider it fixed.

- Determine the output of the perceptron for the input -1:
  $w_1 x_1 = 0.5(-1) = -0.5 \leq \theta \rightarrow o = 0$
- The new weight w_1 after applying the learning rule:
  $\Delta w_1 = 0.6(1 - 0)(-1) = -0.6 \rightarrow w_1 = 0.5 - 0.6 = -0.1$

Learning Rule: Exercise

Perceptron Learning Rule

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i \; = \eta(t - o)x_i$$

Consider a perceptron with only one input x_1, weight $w_1 = 0.5$, threshold $\theta = 0$ and learning rate $\eta = 0.6$. Consider also the training example $\{x_1 = -1, t = 1\}$. For now, let's temporarily ignore the learning of the threshold and consider it fixed.

- Determine the output of the perceptron for the input -1:
  $w_1 x_1 = 0.5(-1) = -0.5 \leq \theta \rightarrow o = 0$
- The new weight w_1 after applying the learning rule:
  $\Delta w_1 = 0.6(1 - 0)(-1) = -0.6 \rightarrow w_1 = 0.5 - 0.6 = -0.1$
- The new output of the perceptron for the input -1:

Learning Rule: Exercise

Perceptron Learning Rule

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i \; = \eta(t - o)x_i$$

Consider a perceptron with only one input x_1, weight $w_1 = 0.5$, threshold $\theta = 0$ and learning rate $\eta = 0.6$. Consider also the training example $\{x_1 = -1, t = 1\}$. For now, let's temporarily ignore the learning of the threshold and consider it fixed.

- Determine the output of the perceptron for the input -1:
  $w_1 x_1 = 0.5(-1) = -0.5 \leq \theta \rightarrow o = 0$
- The new weight w_1 after applying the learning rule:
  $\Delta w_1 = 0.6(1 - 0)(-1) = -0.6 \rightarrow w_1 = 0.5 - 0.6 = -0.1$
- The new output of the perceptron for the input -1:
  $w_1 x_1 = -0.1(-1) = 0.1 \geq \theta \rightarrow o = 1$

Learning Algorithm

```
Initialize all weights randomly.
repeat
    for each training example do
        Apply the learning rule.
    end for
until the error is acceptable or a certain number
of iterations is reached
```

This algorithm is guaranteed to find a solution with zero error in a limited number of iterations as long as the examples are linearly separable.

Summary

- Neural networks (aka deep learning) is a computer modeling approach inspired by networks of biological neurons.
- A neural net consists of units and connections.
- The perceptron is the simplest neural network model; it is a linear classifier.
- A learning algorithm for perceptrons exists.
- **Key limitation:** only works for linearly separable data.