CMSC 373 Artificial Intelligence
Fall 2023
11-SubsymbolicAI

Deepak Kumar
Bryn Mawr College

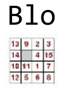
1

Symbolic *versus* Subsymbolic AI

- **Symbolic AI**

  Everything is represented using symbols.

  A is a block `Block(A)`

  Representation of a state 

  Expert Systems, Frames, Scripts, Semantic Nets, Knowledge Graphs etc.

- **Subsymbolic AI**

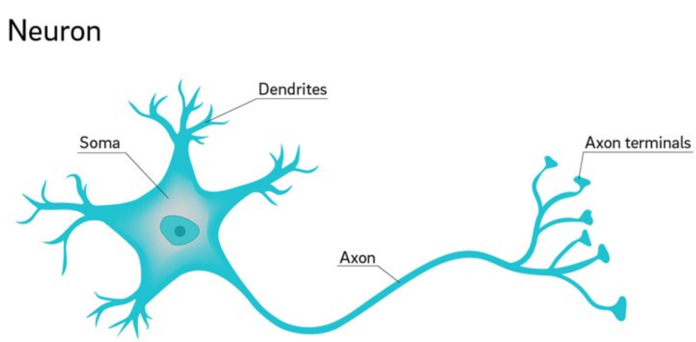
  There are NO SYMBOLS.

  Approaches that employ Neural Networks and other statistical mechanisms

2

2

Neural Networks – “Inspired by the Brain”



From: https://today.ucsd.edu/story/why_are_neuron_axons_long_and_spindly

3

3

A Neural Network

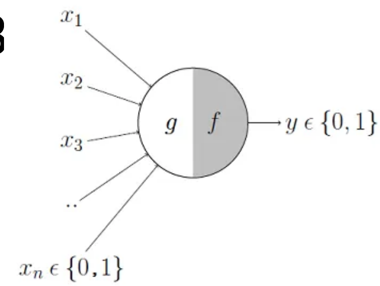
From: https://sites.dartmouth.edu/dujs/2015/01/20/neural-networks-rival-the-primate-brain-at-object-recognition/

4

4

McCulloch-Pitts Neuron, 1943

- Binary Threshold Units
- Captures the inhibitory and excitatory connections between biological neurons.
- Limited in what such a model can actually do.
- Missing the learning capability: how to model changes in inhibitory and excitatory connections. This was later included in Hebb's model (1949). Repeated firings can modify the nature of the connections.
- Frank Rosenblatt, 1958 combined these ideas into the model of a **Perceptron**.



$$g(x_1, x_2, x_3, ..., x_n) = g(\mathbf{x}) = \sum_{i=1}^{n} x_i$$

$$y = f(g(\mathbf{x})) = 1 \quad if \quad g(\mathbf{x}) \geq \theta$$
$$= 0 \quad if \quad g(\mathbf{x}) < \theta$$

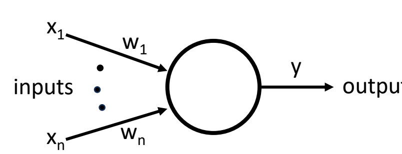
From: https://towardsdatascience.com/mcculloch-pitts-model-5fdf65ac5dd1

5

5

The Perceptron – A Gross Approximation (Rosenblatt, 1958)

- A single "neuron" (**<u>unit</u>**)
  aka Threshold Logic Unit (TLU)



- **Transfer Function**
  T is the Threshold value (assume $T = 0$)

$$I = \sum_{i=1}^{i=n} w_i x_i$$

$$y = \begin{cases} +1, \text{if } I \geq T \\ -1, \text{if } I < T \end{cases}$$

6

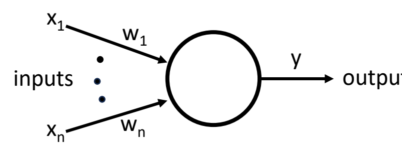
6

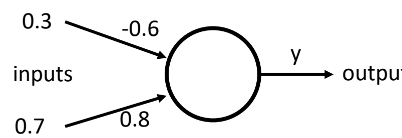
The Perceptron

- A single unit
- Transfer Function (assume $T = 0$)

$$I = \sum_{i=1}^{i=n} w_i x_i$$

$$y = \begin{cases} +1, \text{if } I \geq T \\ -1, \text{if } I < T \end{cases}$$



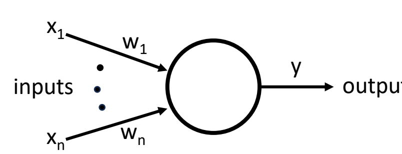


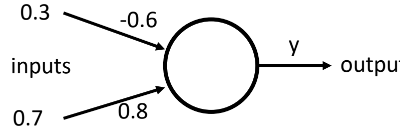
The Perceptron

- A single unit.
- Transfer Function (assume $T = 0$)

$$I = \sum_{i=1}^{i=n} w_i x_i$$

$$y = \begin{cases} +1, \text{if } I \geq T \\ -1, \text{if } I < T \end{cases}$$





$I = w_1 x_1 + w_2 x_2$

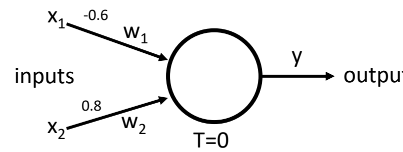
$I = -0.6 * 0.3 + 0.8 * 0.7$

$I = -0.18 + 0.56 = 0.38$

Since I = 0.38 > (T=0)

$y = +1$

Training Data



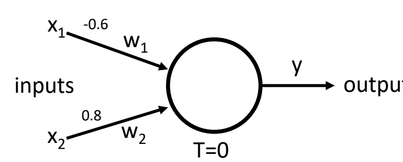
	x_1	x_2	Desired Output
A1	0.3	0.7	+1
B1	-0.6	0.3	-1
A2	0.7	0.3	+1
B2	-0.2	-0.8	-1

	$w_1 x_1$	$w_2 x_2$	Actual Output
A1	-0.18	0.56	+1
B1	0.36	0.24	+1
A2	-0.42	0.24	-1
B2	0.12	-0.56	-1

9

9

Training Data



	x_1	x_2	Desired Output
A1	0.3	0.7	+1
B1	-0.6	0.3	-1
A2	0.7	0.3	+1
B2	-0.2	-0.8	-1

	$w_1 x_1$	$w_2 x_2$	Actual Output
A1	-0.18	0.56	+1
B1	0.36	0.24	+1
A2	-0.42	0.24	-1
B2	0.12	-0.56	-1

Incorrect Output!

10

10

Perceptron Learning Rule

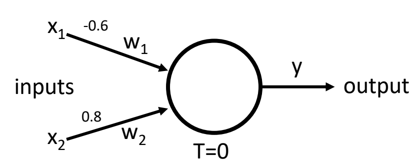
- Changes the weights

$\overline{\boldsymbol{w}} = [w_1, w_2]$ weight vector

$\overline{\boldsymbol{x}} = [x_1, x_2]$ input vector

$\overline{\boldsymbol{w_{new}}} = \overline{\boldsymbol{w_{old}}} - y^* \, \overline{\boldsymbol{x}}$ Training Rule

Training Data



$\overline{\boldsymbol{w_{new}}} = \overline{\boldsymbol{w_{old}}} - y^* \, \overline{\boldsymbol{x}}$ Training Rule

	x_1	x_2	Desired Output
A1	0.3	0.7	+1
B1	-0.6	0.3	-1
A2	0.7	0.3	+1
B2	-0.2	-0.8	-1

	w_1x_1	w_2x_2	Actual Output
A1	-0.18	0.56	+1
B1	0.36	0.24	+1
A2	-0.42	0.24	-1
B2	0.12	-0.56	-1

	w_1	w_2	$\sum_{i=1}^{i=n} w_i x_i$	y
A1	-0.6	0.8	0.56	+1
B1	-0.6	0.8	0.6	+1
A2	0.0	0.5	0.15	+1
B2	0.0	0.5	-0.4	-1

Vocabulary

- **Labelled Training Dataset**
  N samples/patterns/input vector with desired outputs (targets/labels)
- **Output Error (Loss)**
  Error = Desired Output – Actual Output
- **Learning Rule**
  Specifies change in the weights using the Error
- **Prediction/Forward Pass**
  Application of a pattern to produce output
- **Epoch**
  1 pass through the training dataset

13

13

Vocabulary - Perceptron

- **Labelled training Dataset**
  N samples/patterns/input vector with desired outputs (targets/labels)
- **Output Error (Loss)**
  y is the perceptron's answer/output

$$\boldsymbol{\beta} = \begin{cases} +\mathbf{1}, \text{ if perceptron's answer is correct} \\ -\mathbf{1}, \text{ if perceptron's answer is wrong} \end{cases}$$

- **Learning Rule**
  Specifies change in the weights using the Error
  Perceptron Learning Rule: $\overline{\boldsymbol{w_{new}}} = \overline{\boldsymbol{w_{old}}} + \beta y^* \,\overline{\boldsymbol{x}}$
- **Prediction/Forward Pass**
  Application of a pattern to produce output
- **Epoch**
  1 pass through the training dataset

14

14

Perceptron Training Algorithm

Initialize all weights to random values

    #In what range? Typically [-1.0..1.0]

Set #Epochs to some N

    // How to decide what N should be?

Do N times or until all outputs are correct

    Do for each pattern in the training set

        apply the pattern to the perceptron

        change the weight vector as defined

15

15

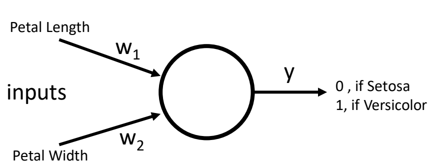
Example – Iris Dataset

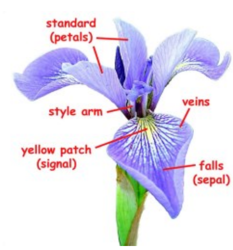
- 150 Samples, 50 of each variety

Sepal Length & Width

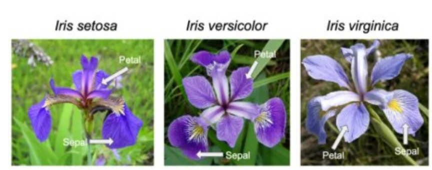
```
[5.1, 3.5, 1.4, 0.2],
[4.9, 3.0, 1.4, 0.2],
[4.7, 3.2, 1.3, 0.2],
[4.6, 3.1, 1.5, 0.2],
...
```

Petal Length & Width





From: https://www.fs.usda.gov/wildflowers/beauty/iris/flower.shtml



https://peaceadegbite1.medium.com/iris-flower-classification-60790e9718a1

16

16

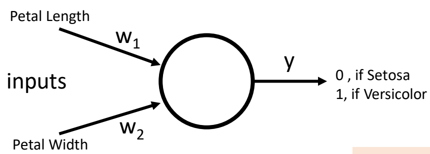
Example – The Iris Dataset

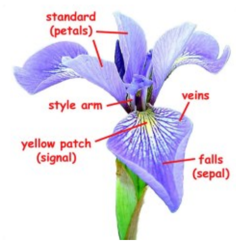
- 150 Samples, 50 of each variety
  Labelled: 0 (Setosa), 1(Versicolor), 2(Virginica)

Sepal Length & Width

```
[5.1, 3.5, 1.4, 0.2],
[4.9, 3. , 1.4, 0.2],
[4.7, 3.2, 1.3, 0.2],
[4.6, 3.1, 1.5, 0.2],
...
```

Petal Length & Width





From: https://www.fs.usda.gov/wildflowers/beauty/iris/flower.shtml

https://peaceadegbite1.medium.com/iris-flower-classification-60790e9718a1

Parameters = 2
Hyperparameters: # of epochs

17

17

Introducing Google Colab

- Live demo…
- Writing a Perceptron from scratch…

18

18

References

- M. Caudill and C. Butler: Understanding Neural Networks, Volume 1, MIT Press, 1993.
- F. Chollet: *Deep Learning with Python, Second Edition*, Manning2021.
- A Geron: *Hands-on Machine Learning with SciKit-Learn, Keras and TensorFlow*, Oreilly, 2019.
- M. Mitchell: *Artificial Intelligence: A Guide For Thinking Humans*, Farrar, Strouss, Giroux, 2019.
- M. Wooldridge: *A Brief History of Artificial Intelligence*. Flatiron Books, 2020.