

Demystifying Artificial Intelligence Sorcery

(Part 1: Fuzzy Logic & Neural Networks)^a

Abdelbacet Mhamdi
abdelbacet.mhamdi@bizerte.r-iset.tn

Dr.-Ing. in Electrical Engineering
Senior Lecturer at ISET Bizerte

^aAvailable @ <https://github.com/a-mhamdi/jlai/>



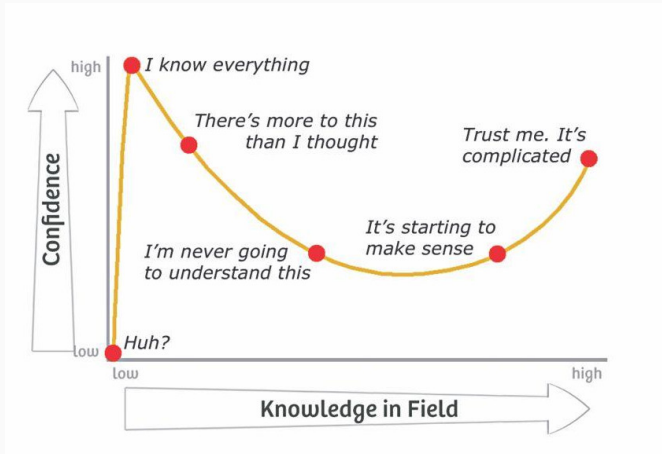
Disclaimer

This document features some materials gathered from multiple online sources.

Please note no copyright infringement is intended, and I do not own nor claim to own any of the original materials. They are used for educational purposes only.

I have included links solely as a convenience to the reader. Some links within these slides may lead to other websites, including those operated and maintained by third parties. The presence of such a link does not imply a responsibility for the linked site or an endorsement of the linked site, its operator, or its contents.

DUNNING-KRUGER EFFECT



Kruger, J. and Dunning, D. (1999) *Unskilled and unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments*. **J Pers Soc Psychol.** 77(6) pp. 1121–1134.

doi 10.1037/0022-3514.77.6.1121

“Knowledge isn’t free. You have to pay attention.”

Richard P. Feynman

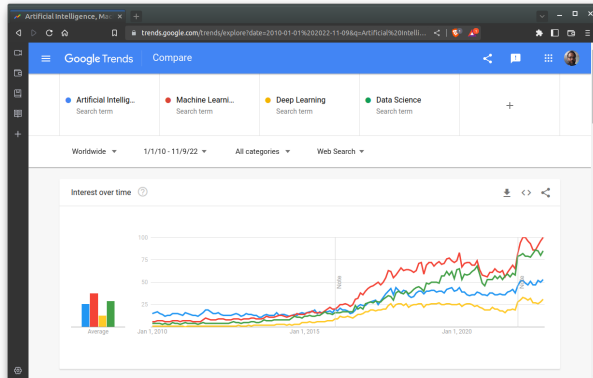


ROADMAP

1. An overview
2. Fuzzy Logic
3. Neural Networks
4. Quizzes

An overview

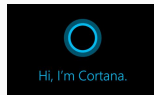
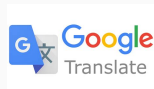
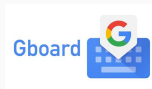
TRENDS



“Numbers represent search interest relative to the highest point on the chart for the given region and time.

- A value of 100 is the peak popularity for the term;
- A value of 50 means that the term is half as popular;
- A score of 0 means there was not enough data for this term.”

TOP USES



Artificial intelligence is a branch of computer science which focuses on automation of intelligent behavior.



SOME DEFINITIONS CAN BE CATEGORIZED INTO FOUR FRAMES.

Artificial intelligence is a branch of computer science which focuses on automation of intelligent behavior.



SOME DEFINITIONS CAN BE CATEGORIZED INTO FOUR FRAMES.

SYSTEMS THAT THINK LIKE HUMANS

[Bel78]

“[The automation of] activities that we associate with human thinking, activities such as decision-making, problem-solving, learning...”

Bellman, R. E. *An Introduction to Artificial Intelligence: Can Computers Think?* **Boyd & Fraser Publishing Company.**

[Hau89]

“The exciting new effort to make computers think[...] *machines with minds*, in the full and literal sense”

Haugeland, J. (1989). *Artificial Intelligence: The Very Idea.* **A Bradford book. MIT Press.**

SYSTEMS THAT THINK RATIONALLY

[CMM85]

“The study of mental faculties through the use of computational models.”

Charniak, E., McDermott, D., and McDermott, D. V. (1985). *Introduction to Artificial Intelligence*. Addison-Wesley series in computer science and information processing. Addison-Wesley.

[Win92]

“The study of the computations that make it possible to perceive, reason, and act.”

Winston, P. H. (1992). *Artificial Intelligence*. A-W Series in Computer Science. Addison-Wesley Publishing Company.

SYSTEMS THAT ACT LIKE HUMANS

[Kur92]

“The art of creating machines that perform functions that require intelligence when performed by people.”

Kurzweil, R. (1992). *The Age of Intelligent Machines*. **Viking**.

[RK91]

“The study of how to make computers do things at which, at the moment, people are better.”

Rich, E. and Knight, K. (1991). *Artificial Intelligence*. **Artificial Intelligence Series. McGraw-Hill**.

SYSTEMS THAT ACT RATIONALLY

[Sch90]

“A field of study that seeks to explain and emulate intelligent behavior in terms of computational processes.”

Schalkoff, R. J. (1990). *Artificial Intelligence: An Engineering Approach*. **McGraw-Hill Computer science series. McGraw-Hill.**

[LS93]

“The branch of computer science that is concerned with the automation of intelligent behavior.”

Luger, G. F. and Stubblefield, W. A. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. **Artificial intelligence. Benjamin/Cummings Publishing Company.**

THOUGHT-PROVOKING QUESTIONS



How to achieve intelligence on a computer system

What do we mean by “Intelligence”?

- Single faculty or gathering of abilities
- Learned or existing
- What happens when we learn
- Are creativity and intuition measurable
- Does observable behavior infer to intelligence
- How knowledge is routed in the human brain

THOUGHT-PROVOKING QUESTIONS



How to achieve intelligence on a computer system

What do we mean by “Intelligence”?

- ➡ Single faculty or gathering of abilities
- ➡ Learned or existing
- ➡ What happens when we learn
- ➡ Are creativity and intuition measurable
- ➡ Does observable behavior infer to intelligence
- ➡ How knowledge is routed in the human brain

TURING TEST

Alan Turing (1950)

The ability to achieve human level performance in all cognitive tasks, sufficient to fool an interrogator.

- ✓ Natural Language Processing (NLP) (*Communicate in human language*)
 - ✓ Knowledge Representation (*Store information*)
 - ✓ Automated Reasoning (*Answer questions & draw conclusions*)
 - ✓ Machine Learning (ML) (*Adapt to new circumstances, detect & extrapolate patterns*)
-

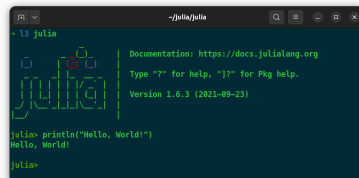
FORMS OF AI


- ☆ Expert Systems (*Based on knowledge or rule settings*)
- ☆ Fuzzy Systems (*Based on fuzzy set theory*)
- ☆ Artificial Neural Networks
- ☆ Genetic Algorithms
- ☆ Belief Networks
- ☆ Hybrid Systems (*Combine two or more approaches*)

PROGRAMMING LANGUAGE



julialang.org/

A screenshot of a Julia REPL window. The window title is "~julia/julia". The prompt is "julia". The output shows the Julia logo, documentation URL, help instructions, and version information. The user has entered "println(\"Hello, World!\")" and the output is "Hello, World!".

```
+ julia  
 Documentation: https://docs.julialang.org  
Type "?" for help, "}" for Pkg help.  
Version 1.6.3 (2021-09-23)  
  
julia> println("Hello, World!")  
Hello, World!  
  
julia>
```

DEVELOPMENT ENVIRONMENTS



Pluto.jl



▲ \$ docker compose up

▼ \$ docker compose down

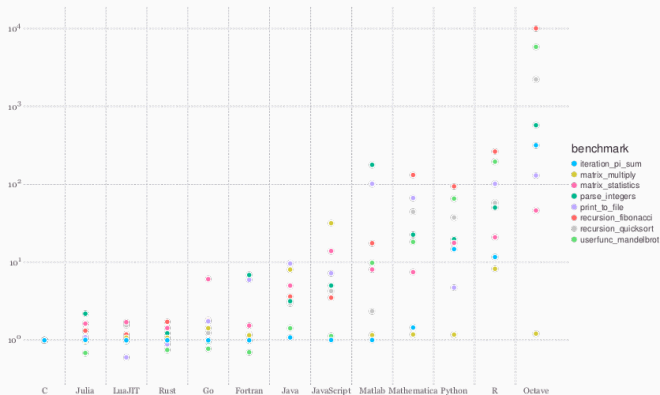


JULIA IN A NUTSHELL

- ▲ **Fast:** native code for multiple platforms via LLVM;
- ▲ **Dynamic:** good support for interactive use (*like a scripting language*);
- ▲ **Reproducible:** environment recreation across platforms, with pre-built binaries;
- ▲ **Composable:** multiple dispatch as a paradigm (*oop & functional programming*);
- ▲ **General:** asynchronous I/O, metaprogramming, debugging, logging; profiling, pkg, ...
- ▲ **Open Source:** GitHub repository at <https://github.com/JuliaLang/julia>.



JULIA MICRO-BENCHMARKS (1/2)



<https://julialang.org/benchmarks>



JULIA MICRO-BENCHMARKS (2/2)

Geometric Means¹ of Micro-Benchmarks by Language

1	C	1.0
2	Julia	1.17006
3	LuaJIT	1.02931
4	Rust	1.0999
5	Go	1.49917
6	Fortran	1.67022
7	Java	3.46773
8	JavaScript	4.79602
9	Matlab	9.57235
10	Mathematica	14.6387
11	Python	16.9262
12	R	48.5796
13	Octave	338.704



¹Measure of central tendency expressed as $(x_1 \times x_2 \times \dots \times x_n)^{1/n}$



SOURCE CONTROL MANAGEMENT (SCM)

The screenshot shows the GitHub repository page for 'a-mhamdi/jlai'. The repository is public and has 2 stars and 3 forks. The main branch is 'main'. The repository contains a README.md file, a LICENSE file, and a .gitignore file. The README.md file is titled 'Fuzzy Logic, Machine Learning and Deep Learning with Julia' and contains a table of contents with links to various sections. The repository also has a Dockerfile and a Docker image.

Repository Details:

- Repository: a-mhamdi / jlai (Public)
- Buttons: Unpin, Unwatch (2), Fork (3), Star (2)
- Navigation: Code (selected), Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, Settings
- Branches: main (selected), 1 branch, 0 tags
- Buttons: Go to file, Add file, Code (selected)

Files and Commits:

File	Commit Message	Commit Hash	Time
a-mhamdi	vgg and resnet transfer learning	fde8fca	yesterday
			87 commits
.github/workflows	Update docker-image.yml		2 weeks ago
Codes	vgg and resnet transfer learning		yesterday
Docker	rm Docker cheat sheet		3 days ago
Exams	exam w/ answers		4 days ago
Slides-Labs	change colors		yesterday
.gitignore	change colors		yesterday
LICENSE	Initial commit		4 months ago
README.md	update Docker README file		2 weeks ago

About:

An Introduction to Artificial Intelligence with Julia

Tags: flux, machine-learning, docker-image, fuzzy-logic, julialang, mij

Readme, MIT license, 2 stars, 2 watching, 3 forks

Languages:

Language	Percentage
Julia	94.3%
Dockerfile	3.4%
Batchfile	2.1%
TeX	0.2%

<https://github.com/a-mhamdi/jlai>



DOCKER IMAGE

The screenshot shows a web browser window displaying the Docker Hub page for the repository `abmhamdi/jlai-p1`. The page has a dark theme. At the top, there's a navigation bar with the Docker Hub logo and links for Explore, Repositories, Organizations, and Usage. A search bar is also present. The main content area shows the repository name `abmhamdi/jlai-p1` with a Docker logo icon. Below the name, it says 'By `abmhamdi` · Updated 12 minutes ago' and 'Artificial Intelligence Labs - Part 1 @ ISETBZ'. There are tags for 'DATA SCIENCE', 'LANGUAGES & FRAMEWORKS', and 'MACHINE LEARNING & AI'. The repository has 0 stars and 119 downloads. A 'Manage Repository' button is visible. Below this, there are tabs for 'Overview' and 'Tags'. The 'Overview' tab is selected, showing a description: 'This repository contains slides, labs and code examples for using Julia to implement some artificial intelligence related algorithms. Codes run on top of a Docker image, ensuring a consistent and reproducible environment.' There's also a 'Fuzzy Logic and Neural Networks with Julia' section. On the right, there's a 'Docker Pull Command' section with the command `docker pull abmhamdi/jlai-p1` and a 'Copy' button.

<https://hub.docker.com/r/abmhamdi/jlai-p1>

Fuzzy Logic

WHAT IS FUZZY LOGIC? (1/3)

“There are many misconceptions about fuzzy logic. To begin with, fuzzy logic is not fuzzy. Basically, fuzzy logic is a precise logic of imprecision. [...] fuzzy logic is designed to deal with imperfect information. Imperfect information is information which in one or more aspects is imprecise, uncertain, incomplete, unreliable, vague or partially true. In the real world, such information is the norm rather than exception.”

Lotfi Zadeh, WCECS 2014



WHAT IS FUZZY LOGIC? (2/3)

“Fuzzy Logic, in computer science, is a form of logic used in some expert systems and other artificial-intelligence applications in which variables can have degrees of truthfulness or falsehood represented by a range of values between 1 (true) and 0 (false). With fuzzy logic, the outcome of an operation can be expressed as a probability rather than as a certainty. For example, in addition to being either true or false, an outcome might have such meanings as probably true, possibly true, possibly false, and probably false.”

Fuzzy Logic, Microsoft® Encarta® Online Encyclopedia 2009

https://www.refseek.com/data/cache/en/1/Fuzzy_Logic.html

WHAT IS FUZZY LOGIC? (3/3)

In fuzzy logic, an element can *partially* belong to a set with a membership degree $\mu_{\mathcal{A}}(x) \in [0, 1]$

- ▶ Extension of classical Boolean logic
- ▶ Deals with **degrees of truth** rather than absolute true/false
- ▶ Variables can have values between 0 and 1
- ▶ Mimics human reasoning and decision-making

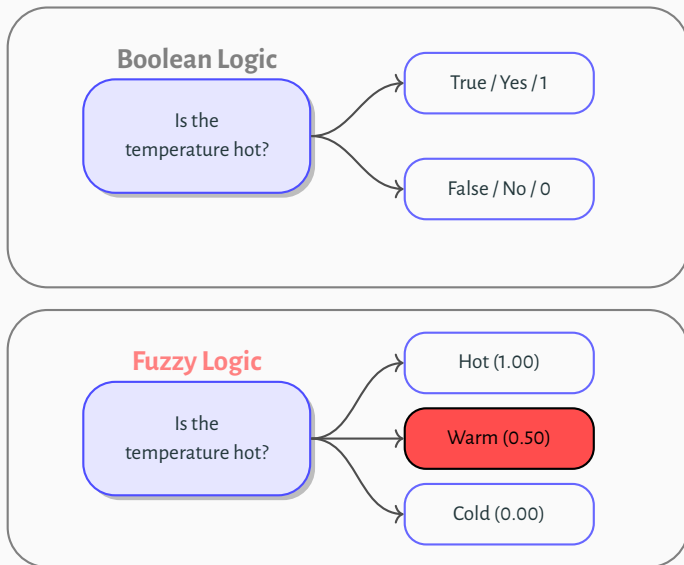
Crisp Set (Classical)

- ▶ Binary membership: 0 or 1
- ▶ Sharp boundaries
- ▶ Example: Age ≥ 65 = OLD

Fuzzy Set

- ▶ Gradual membership: $[0, 1]$
- ▶ Smooth transitions
- ▶ Example: Age 60 = 0.5 OLD

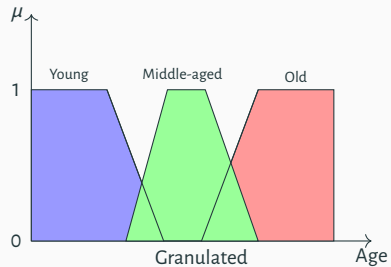
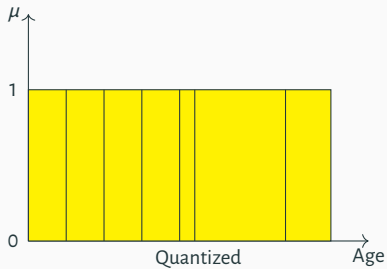
FUZZY LOGIC AS AN EXTENSION OF THE BOOLEAN LOGIC



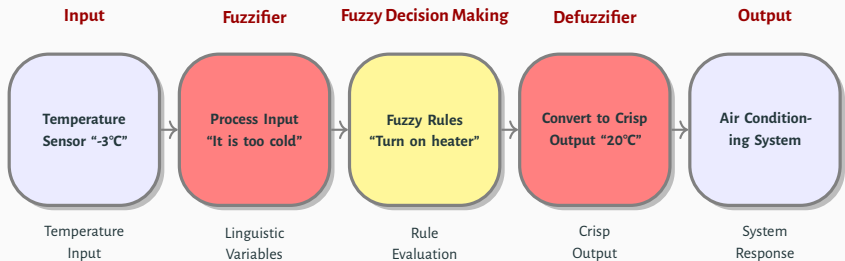
WHAT DOES FUZZY LOGIC HAVE TO OFFER?

Fuzzy Logic aims at formalizing/mechanizing two noticeable human capabilities:

1. communicating, reasoning and rational decision making
(in presence of imprecision, uncertainty & partiality of truth)
2. performing a wide variety of tasks
(w/o measurements or computations)

Continuous \rightarrow Quantized \rightarrow Granulated

EXAMPLE OF A FUZZY CONTROL SYSTEM



ARCHITECTURE

Rule Base is provided by experts. It contains the set of rules to govern the decision making.

Fuzzification converts crisp numbers to fuzzy sets.

Inference Engine decides which rules to be fired matching degree of the current fuzzy inputs.

Defuzzification converts the fuzzy sets delivered by the inference engine into some crisp value

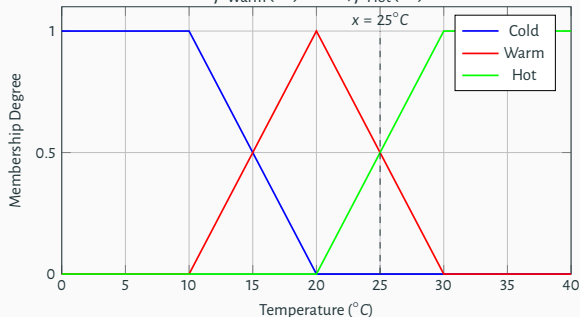
FUZZIFICATION PROCESS

Fuzzification transforms crisp input values into fuzzy membership degrees using membership functions.

Steps:

1. Define linguistic variables (e.g., temperature: Cold, Warm, Hot)
2. Choose membership functions for each linguistic term
3. Calculate membership degrees for input values

For $x = 25^{\circ}\text{C}$: $\mu_{\text{Warm}}(25) = 0.5$, $\mu_{\text{Hot}}(25) = 0.5$



Common types of membership functions:

- **Triangular**
- **Trapezoidal**
- **Gaussian**
- **Sigmoid**

FUZZY RULE BASE

IF-THEN Rules

Rules express expert knowledge in linguistic form:

General Form:

IF x is X AND y is Y THEN z is Z

Example Rules:

1. IF temperature is Cold AND humidity is High THEN fan speed is Low
2. IF temperature is Hot AND humidity is Low THEN fan speed is High
3. IF temperature is Warm THEN fan speed is Medium

INFERENCE METHODS

Mamdani Inference:

- ▶ Most common method
- ▶ Output is a fuzzy set
- ▶ Uses min-max composition

Sugeno Inference:

- ▶ Output is a linear function or constant
- ▶ Computationally efficient
- ▶ Common in adaptive systems

Tsukamoto Inference:

- ▶ Output membership functions must be monotonic
- ▶ Direct crisp output calculation

DEFUZZIFICATION METHODS

A fuzzy value can be defuzzified through multiple ways.

Common Methods:

1. Centroid (Center of Area - **COA**, aka Center of Gravity - **COG**)
2. Bisector of Area (**BOA**)
3. Mean of Maximum (**MOM**)
4. Smallest/Largest of Maximum (**SOM** and **LOM**)
5. Weighted Average

CENTROID METHOD (COA | COG)

Most widely used method

$$z^* = \frac{\int \mu_Z(z) \cdot z \, dz}{\int \mu_Z(z) \, dz}$$

For discrete case:

$$z^* = \frac{\sum_{i=1}^n \mu_Z(z_i) \cdot z_i}{\sum_{i=1}^n \mu_Z(z_i)}$$

- ▶ Finds the center of gravity of the fuzzy set
- ▶ Considers entire output distribution
- ▶ Smooth output variation

LOGICAL SYMBOLS

In formal logic \neg is NOT, \vee is OR and \wedge is AND

Consider the following propositions \mathcal{E} and \mathcal{S}

\mathcal{E} "The earth is round"

\mathcal{S} "The sun spins on its axis"

Thus

$\neg \mathcal{E}$ "The earth is not round"

$\neg \mathcal{S}$ "The sun does not spin on its axis"

$\mathcal{E} \vee \mathcal{S}$ "The earth is round **or** the sun spins on its axis"

$\mathcal{E} \wedge \mathcal{S}$ "The earth is round **and** the sun spins on its axis"

$\neg \mathcal{E} \vee \mathcal{S}$ "The earth is not round **or** the sun spins on its axis"

$\mathcal{E} \wedge \neg \mathcal{S}$ "The earth is round **and** the sun does not spin on its axis"

CRISP VALUE IN FUZZY NOTATION (1/2)

Boolean Values as Fuzzy Sets

- ▶ $0 \rightarrow \{0\}$ (completely non-membership)
- ▶ $1 \rightarrow \{1\}$ (completely membership)
- ▶ In between 0 and 1, *e.g.*, $0.5 \rightarrow \{0.5\}$ (partial membership)

Crisp Set Notation as Fuzzy Set Notation

- ▶ A crisp set $\mathbf{A} = \{x \mid P(x)\}$ can be represented as a fuzzy set \mathcal{A} with membership function:

$$\mu_{\mathcal{A}}(x) = \begin{cases} 1 & \text{if } x \in \mathbf{A} \\ 0 & \text{if } x \notin \mathbf{A} \end{cases}$$

- ▶ Alternatively, a fuzzy set \mathcal{A} can be defined with a membership function:

$$\mu_{\mathcal{A}}(x) = \begin{cases} 1 & \text{if } x \text{ is completely in } \mathcal{A} \\ 0.5 & \text{if } x \text{ is partially in } \mathcal{A} \\ 0 & \text{if } x \text{ is completely out of } \mathcal{A} \end{cases}$$

CRISP VALUE IN FUZZY NOTATION (2/2)

Fuzzy Notation for Crisp Set Elements

A crisp element $x \in \mathbf{U}$ can be represented as a fuzzy set $\{x\}$ with membership function:

$$\mu_{\{x\}}(x') = \delta(x, x')$$

where δ is the Dirac delta function.

Fuzzy Representation of Crisp Set Operations

For fuzzy sets \mathcal{A} and \mathcal{B} :

- Union ($\mathcal{A} \cup \mathcal{B}$):

$$\mu_{\mathcal{A} \cup \mathcal{B}}(x) = \max(\mu_{\mathcal{A}}(x), \mu_{\mathcal{B}}(x))$$

- Intersection ($\mathcal{A} \cap \mathcal{B}$):

$$\mu_{\mathcal{A} \cap \mathcal{B}}(x) = \min(\mu_{\mathcal{A}}(x), \mu_{\mathcal{B}}(x))$$

- Complement ($\neg \mathcal{A}$):

$$\mu_{\neg \mathcal{A}}(x) = 1 - \mu_{\mathcal{A}}(x)$$

FUZZY OPERATORS

AND Operation (min)

$$\mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x))$$

OR Operation (max)

$$\mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x))$$

NOT Operation

$$\mu_{\neg A}(x) = 1 - \mu_A(x)$$

Implication

$$\mu_{A \rightarrow B}(x, y) = \min(\mu_A(x), \mu_B(y))$$

Example

If temp is Hot (0.7) AND humidity is High (0.8) \Rightarrow Activation = $\min(0.7, 0.8) = 0.7$

FUZZY NUMBERS (1/6)

★ Represent imprecise numbers: number & linguistic modifier (*e.g., nearly, around, etc.*)

- ▶ approximately five kilos
- ▶ about 12 pm

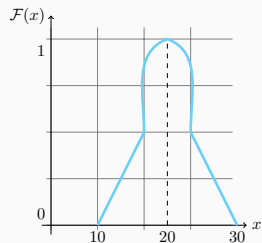
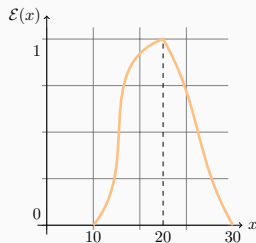
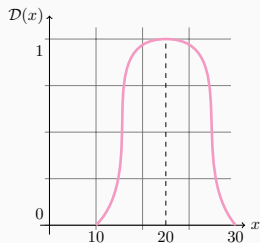
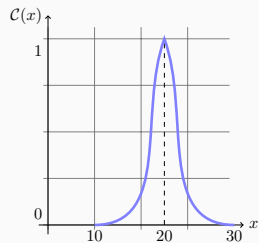
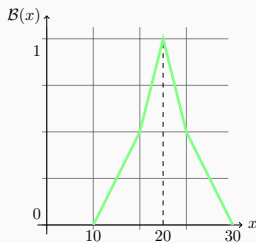
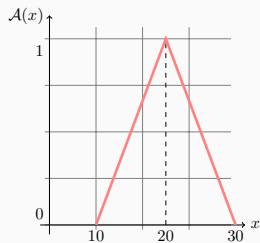
★ Play an important role in decision making, approximate reasoning, statistics with imprecise probabilities and fuzzy control.

We need to perform arithmetic operations on fuzzy numbers (*e.g., calculate a ratio of some fuzzy output over some fuzzy input*)

“around 20”

- ▶ includes some number values on either side of the central value of 20
- ▶ Central value is fully compatible with concept
- ▶ Number around central value are compatible with it to lesser degrees
- ▶ Degree of compatibility represented by fuzzy set; Membership value decreases from 1.0 to 0.0 on both sides of central value = fuzzy number.

FUZZY NUMBERS (2/6)



FUZZY NUMBERS (3/6)

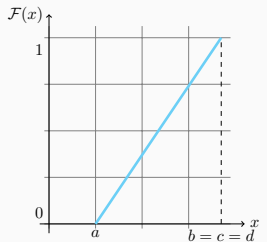
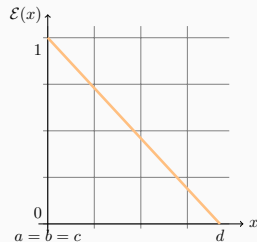
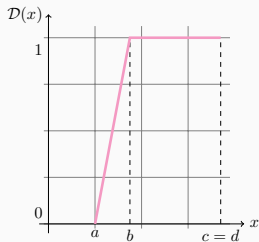
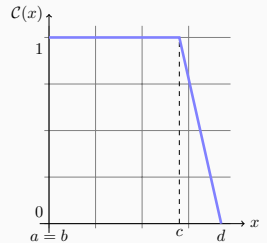
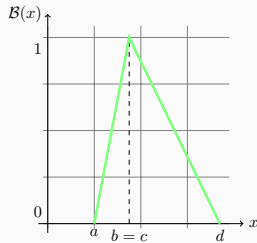
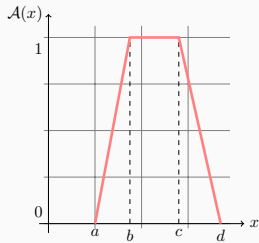
For a fuzzy membership function to qualify as a fuzzy number, it must capture our intuitive concept of a set of numbers around a given real number or interval of real numbers

$$\mathcal{A}(x) = \begin{cases} f(x) & \text{for } x \in [a, b] \\ 1 & \text{for } x \in [b, c] \\ g(x) & \text{for } x \in [c, d] \\ 0 & \text{for } x < a \text{ or } x > d \end{cases} \quad (1)$$

Common shapes of Fuzzy Numbers

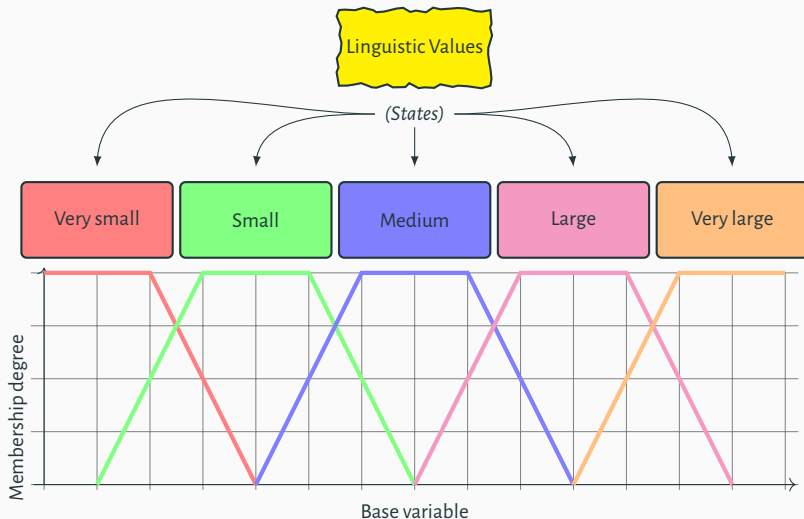
- ▶ Most common membership functions are trapezoidal and triangular (*easy to construct and manipulate*)
- ▶ Choice of a, b, c and d is important and is highly context-dependant
- ▶ Most applications not significantly affected by shapes of functions (*i.e., use linear shapes*)
- ▶ When some of real numbers (a, b, c, d) are equal, get degenerated forms of fuzzy numbers

FUZZY NUMBERS (4/6)



FUZZY NUMBERS (5/6)

States are fuzzy numbers which represent linguistic concepts



FUZZY NUMBERS (6/6)

1. Fuzzy numbers are normal fuzzy sets (height=1)
2. Fuzzy numbers are convex fuzzy sets
3. Support of every fuzzy number is open interval (a, d) of real numbers (support must be bounded)
4. Interval analysis can be used to define arithmetic operations on fuzzy numbers

Basic arithmetic operations:

- ▶ addition $[a, b] + [c, d] = [a + c, b + d]$
- ▶ Subtraction $[a, b] - [c, d] = [a - d, b - c]$
- ▶ Multiplication $[a, b] \times [c, d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)]$
- ▶ Division² $[a, b] \div [c, d] = [a, b] \times [1/d, 1/c]$

²Interval division assumes that the number 0 is not one of the elements in the divisor interval $[c, d]$.

CONSTRUCTING FUZZY SETS (1/5)



HOW WOULD YOU ASSESS TODAY'S TEMPERATURE?

We can describe a parameter describing a phenomena (*e.g.*, *Temperature for environment or Error for distance measurement*) using a finite, small number of descriptors, referred to as linguistic variables of parameter.

Temperature (T) {Cold, Average, Warm}

Error (E) {Small, Medium, Large}



The number of linguistic variables should be kept small (7 ± 2) due to our limited capacity to distinguish more. Commonly 3 to 5 linguistics variables are used in describing parameters.

CONSTRUCTING FUZZY SETS (2/5)

FUZZY SETS → MEMBERSHIP FUNCTIONS

- ▶ Fuzzy sets offer an important and unique approach to describe linguistic variables
- ▶ Membership functions

$$\mathcal{A}(x) = \mathcal{X} \rightarrow [0, 1]$$

are mathematical functions that are used to describe fuzzy sets

- ▶ Choosing membership functions require understanding of:
 - nature of the problem and parameter at hand
 - Level of details to be captured
 - Context of application

Prerequisites

- ▶ Concepts and linguistic values (*e.g., cold temperature*)
- ▶ Numerical measurements and/or linguistic assessments (*e.g., degrees Celsius*)
- ▶ Given context
- ▶ Data or Expert

CONSTRUCTING FUZZY SETS (3/5)

To construct fuzzy sets:

Expert-Driven Using developer, user, decision-maker, etc.

1. Direct methods

- Answers to questions that explicitly pertain to the constructed membership function
- Single or multiple experts

2. indirect methods

- Simpler questions, easier to answer, less sensitive to subjective biases, pertain to membership function only implicitly
- Single or multiple experts

Data-Driven Form data to fuzzy sets

CONSTRUCTING FUZZY SETS (4/5)

Direct Methods with Multiple Experts

Example

n experts were asked to validate the proposition “ x belongs to A ” as either true or false

True $a_i(x) = 1$

False $a_i(x) = 0$

where $i \in \{1 \cdots n\}$ denotes the i^{th} expert.

$$A(x) = \frac{1}{n} \sum_{i=1}^n a_i(x)$$



Can also distinguish degrees of competence c_i of individual experts:

$$A(x) = \sum_{i=1}^n c_i a_i(x), \quad \text{where} \quad \sum_{i=1}^n c_i = 1$$

CONSTRUCTING FUZZY SETS (5/5)

- Given 5 labourers {Q1, Q2, Q3, Q4, Q5}
- Need to determine membership function "A" that captures linguistic term "Excellent Labourer"
- Ask 10 superintendents if particular person is excellent labourer (*answer either yes (1) or no (0)*)
- For each labourer, calculate membership grade of belonging to fuzzy set "A" by taking ratio of total number of yes (1) to total number of responses.

	Q1	Q2	Q3	Q4	Q5
E#1	1	1	1	1	1
E#2	0	0	1	1	1
E#3	0	1	0	1	0
E#4	1	0	1	1	1
E#5	0	0	1	1	1
E#6	0	1	1	1	1
E#7	0	0	0	0	0
E#8	1	1	1	1	1
E#9	0	0	0	1	0
E#10	0	0	0	1	0

⇒ Opinions of individual experts must be aggregated

The resulting set would be: $A = 0.3/Q1 + 0.4/Q2 + 0.6/Q3 + 0.9/Q4 + 0.6/Q5$

Tipping Problem

What should be the TIP at a restaurant, given the quality of FOOD and of SERVICE. These latter are represented by some scores ranging from 0 (*poor*) to 10 (*excellent*).

Rules Base

1. FOOD is rancid \vee SERVICE is poor \implies TIP is cheap;
2. SERVICE is good \implies TIP is average;
3. FOOD is delicious \vee SERVICE is excellent \implies TIP is generous.

Tipping Problem

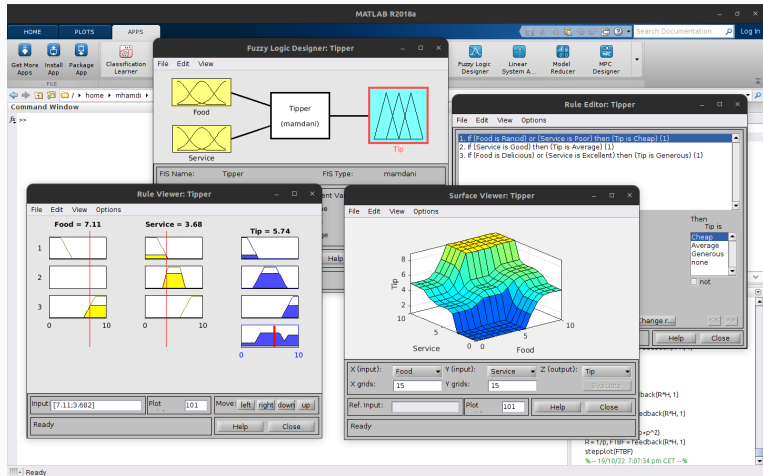
What should be the TIP at a restaurant, given the quality of FOOD and of SERVICE. These latter are represented by some scores ranging from 0 (*poor*) to 10 (*excellent*).

Rules Base

1. FOOD is rancid \vee SERVICE is poor \implies TIP is cheap;
2. SERVICE is good \implies TIP is average;
3. FOOD is delicious \vee SERVICE is excellent \implies TIP is generous.



USING FUZZY LOGIC TOOLBOX



Code is available at <https://github.com/a-mhamdi/cosnip/>
 → Matlab → Fuzzy → Tipper.fis

USING FUZZY.JL PACKAGE



The code is available @ github.com/a-mhamdi/jlai → *Codes* → *Julia* → *Part-1*

→ *Pluto* → *tipper.jl*

Pluto.jl 

→ *Jupyter* → *tipper.ipynb*



MAMDANI FIS

A fuzzy logic controller that uses linguistic rules to map input values to output values.

- ▶ was introduced by Ebrahim (Abe) H. Mamdani in 1975
- ▶ works using rules of linguistics, style like human concepts
(*more intuitive and easier to understand*)
- ▶ well suited to applications where rules are inspired from human expert knowledge.

EXAMPLE

We want to control the temperature of a living room using a thermostat.

- Inputs**
1. ε difference between the desired and measured temperatures
 2. $\delta\varepsilon$ rate of change of the temperature error

Output heating/cooling command(H/C) is the recommended action to take (*e.g., heat, cool or no action*)

FUZZY RULE BASE:

\mathfrak{R}_1 : IF ε is “cold” AND $\delta\varepsilon$ is “rapidly decreasing” THEN H/C is “heat”

\mathfrak{R}_2 : IF ε is “hot” AND $\delta\varepsilon$ is “rapidly increasing” THEN H/C is “cool”

\mathfrak{R}_3 : IF ε is “near target” AND $\delta\varepsilon$ is “slow” THEN H/C is “no action”

Task #1

Consider a fuzzy logic system with two inputs u , v and an output w . We suppose that each variable ranges from $0 \rightarrow 10$. w changes by a unit step. The membership functions of the fuzzy variables are described below.

► u can be:

Negative (N) $\mathcal{L}(2, 4)$

Zero (Z) $\Delta(3, 6, 9)$

Positive (P) $\Gamma(6, 8)$.

► v can be:

Negative (N) $\mathcal{L}(2, 5)$

Zero (Z) $\Pi(2, 4, 6, 8)$

Positive (P) $\Gamma(6, 8)$.

► w can be:

Small (S) $\mathcal{L}(2, 4)$

Medium (M) $\Delta(3, 5, 7)$

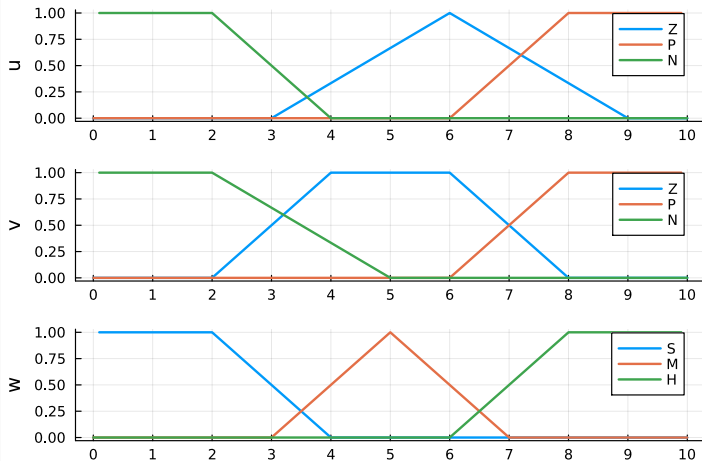
High (H) $\Gamma(6, 8)$.

Rule Base - case of \wedge

		u		
		N	Z	P
v	N	S	S	M
	Z	S	M	H
	P	M	H	H



Evaluate
 w if $u = 4$ & $v = 6$.



$$\left\{ \begin{array}{l} 4 = \{(\mathbf{N}, 0), (\mathbf{Z}, \frac{1}{3}), (\mathbf{P}, 0)\} \\ 6 = \{(\mathbf{N}, 0), (\mathbf{Z}, 1), (\mathbf{P}, 0)\} \end{array} \right. \Rightarrow w^* = \frac{3 \times 0 + 4 \times \frac{1}{3} + 5 \times \frac{1}{3} + 6 \times \frac{1}{3} + 7 \times 0}{0 + \frac{1}{3} + \frac{1}{3} + \frac{1}{3} + 0} = 5$$

Task #2³

Design a fuzzy lighting controller system, in which the control system dims the bulb light automatically according to the environmental light. Assume that the inputs to the system are the environmental light x_1 and the changing rate of the environmental light x_2 . The output y represents the control value of the dimmer.

- x_1 ranges between 120 and 220 lumens. x_1 can be:

Dark (D) $\mathcal{L}(130, 150)$

Ambient (A) $\Pi(130, 150, 190, 210)$

Light (L) $\Gamma(190, 210)$.

- x_2 ranges between -10 and $+10$. x_2 can be:

Negative-Small (NS) $\mathcal{L}(-10, 0)$

Zero (Z) $\Delta(-10, 0, 10)$

Positive-Small (PS) $\Gamma(0, 10)$.

- y ranges between 0 and $+10$. y can be:

Very-Small (VS) $\mathcal{L}(2, 4)$

Small (S) $\Delta(2, 4, 6)$

Big (B) $\Delta(4, 6, 8)$

Very-Big (VB) $\Gamma(6, 8)$.

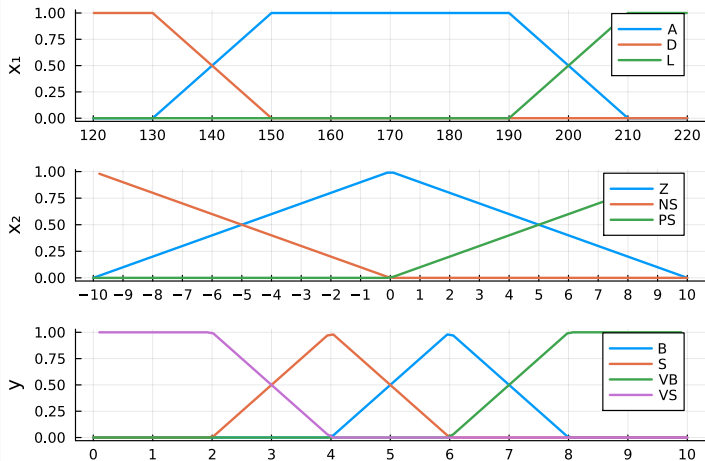
Rule Base - case of \wedge

$x_2 \backslash x_1$	D	A	L
	D	A	L
NS	VB	B	B
Z	B	B	S
PS	B	S	VS



Evaluate
 y if $x_1 = 125$ & $x_2 = -6$.

³Credit: Dr. Mohammed A. T.



$$125 = \{(D, 1), (A, 0), (L, 0)\} \quad \text{and} \quad -6 = \{(NS, \frac{3}{5}), (Z, \frac{2}{5}), (PS, 0)\}$$

$$\Rightarrow y^* = \frac{5 \times \frac{2}{5} + 6 \times \frac{2}{5} + 7 \times \frac{1}{2} + 8 \times \frac{3}{5} + 9 \times \frac{3}{5} + 10 \times \frac{3}{5}}{\frac{2}{5} + \frac{2}{5} + \frac{3}{5} + \frac{1}{2} + \frac{3}{5} + \frac{3}{5} + \frac{3}{5}} \approx 7.775$$

TAKAGI-SUGENO-KANG (TSK) FIS

Each fuzzy rule has the following structure: “IF (antecedent or premise) THEN (consequent)”

Antecedent composed by fuzzy sets defined by MF

Consequent represented by a polynomial function of the fuzzy inputs

EXAMPLE

We want to design a TSK FIS to control the temperature of a room. Our system accepts two inputs: room temperature (T) and outdoor temperature (T_o). The output is the heating or cooling action (H/C).

$$\mathcal{R}_1: \text{IF } T \text{ is “cold” AND } T_o \text{ is “cold” THEN } H/C = 1/2T + 1/3T_o$$

$$\mathcal{R}_2: \text{IF } T \text{ is “cold” AND } T_o \text{ is “hot” THEN } H/C = 1/5T + 3/5T_o$$

$$\mathcal{R}_3: \text{IF } T \text{ is “hot” AND } T_o \text{ is “cold” THEN } H/C = -4/5T + 1/4T_o$$

-
- ▲ Simple and efficient defuzzification
 - ▲ Interpolation (*incomplete or sparse rule bases*)
 - ▲ Smooth and continuous output
 - ▼ Limited rule complexity
 - ▼ Tuning (*careful tuning of the fuzzy rule base*)

Task #3

Suppose we have three fuzzy predicates: \mathcal{A} , \mathcal{B} and \mathcal{C} described by these trapezoidal fuzzy sets:

$$\mathcal{A} \sqcap (0, 2, 5, 9)$$

$$\mathcal{B} \sqcap (2, 8, 13, 16)$$

$$\mathcal{C} \sqcap (11, 16, 19, 19)$$

x and y are fuzzy variables, each one ranges between 0 and 19. Given the following three rules:

$$\mathfrak{R}_1 \quad (x \text{ is } \mathcal{A}) \wedge (y \text{ is } \mathcal{C}) \rightarrow u = 10$$

$$\mathfrak{R}_2 \quad \neg(x \text{ is } \mathcal{A}) \vee (y \text{ is } \mathcal{B}) \rightarrow u = 2$$

$$\mathfrak{R}_3 \quad (x \text{ is } \mathcal{B}) \wedge \neg(y \text{ is } \mathcal{C}) \rightarrow u = 5$$

Compute the degree of satisfaction for each case:

$$\textcircled{1} \quad x_1 = 5 \ \& \ y_1 = 12 \qquad \textcircled{2} \quad x_2 = 0 \ \& \ y_2 = 15 \qquad \textcircled{3} \quad x_3 = 7 \ \& \ y_3 = 13$$

$$x_1 = 5 \text{ \& } y_1 = 12$$

 \mathcal{R}_1

$$\mu_{\mathcal{A}}(x_1) \min \mu_{\mathcal{C}}(y_1)$$

$$1 \min 1/5 = 1/5$$

 \mathcal{R}_2

$$[1 - \mu_{\mathcal{A}}(x_1)] \max \mu_{\mathcal{B}}(y_1)$$

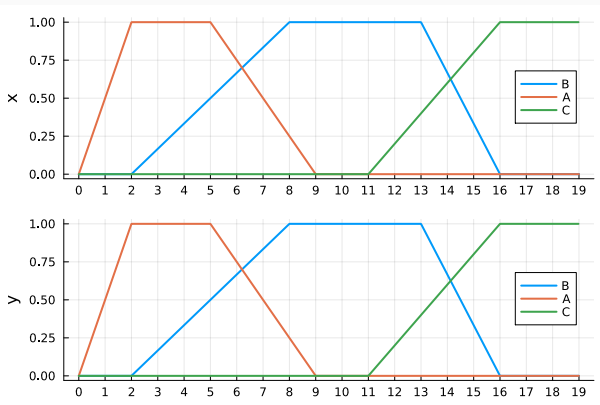
$$0 \max 1 = 1$$

 \mathcal{R}_3

$$\mu_{\mathcal{B}}(x_1) \min [1 - \mu_{\mathcal{C}}(y_1)]$$

$$1/2 \min 4/5 = 1/2$$

$$u_1 = 3.82$$



$$x_2 = 0 \text{ \& } y_2 = 15$$

 \mathcal{R}_1

$$\mu_{\mathcal{A}}(x_2) \min \mu_{\mathcal{C}}(y_2)$$

$$0 \min _ = 0$$

 \mathcal{R}_2

$$[1 - \mu_{\mathcal{A}}(x_2)] \max \mu_{\mathcal{B}}(y_2)$$

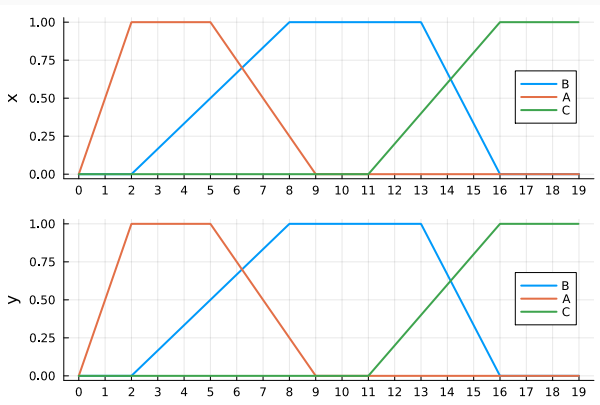
$$1 \max _ = 1$$

 \mathcal{R}_3

$$\mu_{\mathcal{B}}(x_2) \min [1 - \mu_{\mathcal{C}}(y_2)]$$

$$0 \min _ = 0$$

$$u_2 = 2$$



$$x_3 = 7 \ \& \ y_3 = 13$$

 \mathcal{R}_1

$$\mu_{\mathcal{A}}(x_3) \min \mu_{\mathcal{C}}(y_3)$$

$$1/2 \min 2/5 = 2/5$$

 \mathcal{R}_2

$$[1 - \mu_{\mathcal{A}}(x_3)] \max \mu_{\mathcal{B}}(y_3)$$

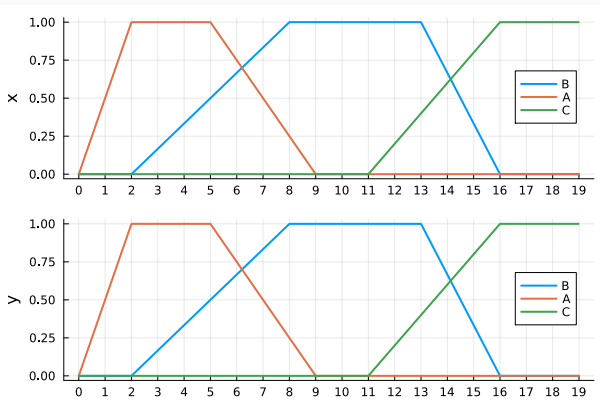
$$1/2 \max 1 = 1$$

 \mathcal{R}_3

$$\mu_{\mathcal{B}}(x_3) \min [1 - \mu_{\mathcal{C}}(y_3)]$$

$$5/6 \min 3/5 = 3/5$$

$$u_3 = 4.5$$



Task #4⁴

Let us consider a TSK fuzzy model with the input (x_1, x_2) and a single output y . Evaluate y^* for $(x_1, x_2) = (1.5, 0.5)$. The rules are:

$$\mathcal{R}_1 \text{ IF } x_1 \text{ is } \mathcal{S} \text{ AND } x_2 \text{ is } \mathcal{S} \text{ THEN } y = -x_1 + x_2 + 1$$

$$\mathcal{R}_2 \text{ IF } x_1 \text{ is } \mathcal{S} \text{ AND } x_2 \text{ is } \mathcal{L} \text{ THEN } y = -x_2 + 3$$

$$\mathcal{R}_3 \text{ IF } x_1 \text{ is } \mathcal{L} \text{ AND } x_2 \text{ is } \mathcal{S} \text{ THEN } y = -x_1 + 3$$

$$\mathcal{R}_4 \text{ IF } x_1 \text{ is } \mathcal{L} \text{ AND } x_2 \text{ is } \mathcal{L} \text{ THEN } y = -x_1 + x_2 + 2$$

The fuzzy predicates \mathcal{S} and \mathcal{L} are described by these membership functions:

$$\mathcal{S} \quad \mathcal{L}(-2, 2)$$

$$\mathcal{L} \quad \Gamma(-2, 2)$$

⁴Credit: Dr. Hashim A. H.

$$x_1 = 1.5 \text{ \& } x_2 = 0.5$$

 \mathfrak{R}_1

$$\begin{aligned} &\mu_S(x_1) \min \mu_S(x_2) \\ &0.125 \min 0.375 = 0.125 \\ &(y = 0) \end{aligned}$$

 \mathfrak{R}_2

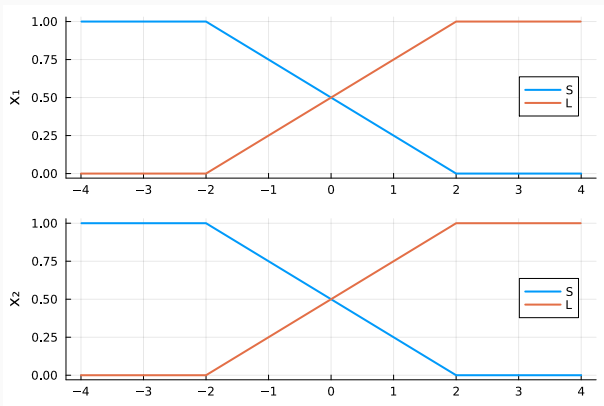
$$\begin{aligned} &\mu_S(x_1) \min \mu_L(x_2) \\ &0.125 \min 0.625 = 0.125 \\ &(y = 2.5) \end{aligned}$$

 \mathfrak{R}_3

$$\begin{aligned} &\mu_L(x_1) \min \mu_S(x_2) \\ &0.875 \min 0.375 = 0.375 \\ &(y = 1.5) \end{aligned}$$

 \mathfrak{R}_4

$$\begin{aligned} &\mu_L(x_1) \min \mu_L(x_2) \\ &0.875 \min 0.625 = 0.625 \\ &(y = 1) \end{aligned}$$



$$y^* = \frac{0 \times 0.125 + 2.5 \times 0.125 + 1.5 \times 0.375 + 1 \times 0.625}{0.125 + 0.125 + 0.375 + 0.625} = 1.2$$

TSUKAMOTO FIS

A type of fuzzy logic-based system developed by TSUKAMOTO in the 1970s. The system would compute the crisp output for each rule based on the firing strength and then take the weighted average to produce the final performance evaluation.

In this method, for each fired rule i with firing strength α_i , the representative output value $y_i(\alpha_i)$ is determined as the value where the consequent membership function $\mu_{y_i}(y) = \alpha_i$.

Due to the **monotonicity** of μ_{y_i} , there exists a unique $y_i(\alpha_i)$.

The crisp output y^* is then computed as the weighted average of each rule's output:

$$y^* = \frac{\sum_{i=1}^n \alpha_i y_i(\alpha_i)}{\sum_{i=1}^n \alpha_i}$$

- ▲ Simplified Rule Base
- ▲ Faster Computation

- ▼ Limited Transparency
- ▼ Limited Flexibility

EXAMPLE

We want to evaluate employee performance based on two criteria:

① job satisfaction and ② productivity

We can define three fuzzy sets for each criterion: “low” (**L**), “medium” (**M**), and “high” (**H**)

FUZZY RULE BASE:

\mathfrak{R}_1 : IF job satisfaction is **L**, THEN performance is **L**.

\mathfrak{R}_2 : IF job satisfaction is **M** AND productivity is **L**, THEN performance is **L**.

\mathfrak{R}_3 : IF job satisfaction is **M** AND productivity is **M**, THEN performance is **H**.

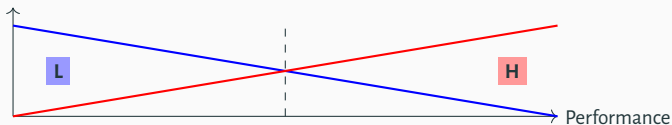
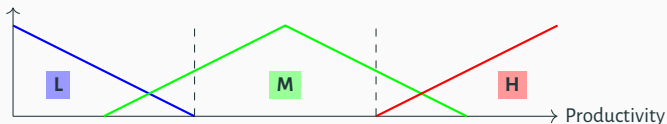
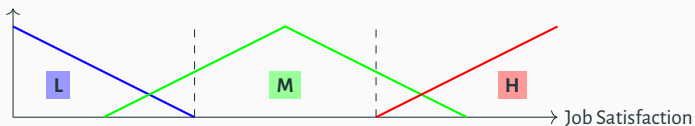
\mathfrak{R}_4 : IF job satisfaction is **M** AND productivity is **H**, THEN performance is **H**.

\mathfrak{R}_5 : IF job satisfaction is **H** AND productivity is **L**, THEN performance is **L**.

\mathfrak{R}_6 : IF job satisfaction is **H** AND productivity is **M**, THEN performance is **H**.

\mathfrak{R}_7 : IF job satisfaction is **H** AND productivity is **H**, THEN performance is **H**.

1. Each rule will compute a firing strength (minimum of input memberships), and then use the inverse of the output membership function to get a crisp value.
2. The final output is a weighted average of all rule outputs.



The method assumes that each consequent fuzzy set \mathcal{Y}_i has a monotonic membership function (either decreasing or increasing), ensuring a one-to-one correspondence between the firing level α_i and the output value $y_i(\alpha_i)$.

Task #5

Consider a simple fuzzy logic controller for a heater system. The input is the **error** (difference between desired and actual temperature, in $^{\circ}\text{C}$), ranging from -10 to 10 . The output is the **heating power** (in %), ranging from 0 to 100 .

Input Fuzzy Sets (for Error):

- ▶ **Negative (\mathcal{N}):** Triangular membership function with peaks at -10 ($\mu = 1$), -5 ($\mu = 1$), 0 ($\mu = 0$).
- ▶ **Zero (\mathcal{Z}):** Triangular membership function with peaks at -5 ($\mu = 0$), 0 ($\mu = 1$), 5 ($\mu = 0$).
- ▶ **Positive (\mathcal{P}):** Triangular membership function with peaks at 0 ($\mu = 0$), 5 ($\mu = 1$), 10 ($\mu = 1$).

Output Fuzzy Sets (for Heating Power, y):

Each is a singleton (crisp value) for TSUKAMOTO's method, assuming monotonicity:

\mathfrak{R}_1 IF Error is \mathcal{N} , THEN $y = 10\%$ (low heat).

\mathfrak{R}_2 IF Error is \mathcal{Z} , THEN $y = 50\%$ (medium heat).

\mathfrak{R}_3 IF Error is \mathcal{P} , THEN $y = 90\%$ (high heat).

Given Crisp Input: Error = -3°C .

- Compute the membership degrees for the input fuzzy sets: $\mu_{\mathcal{N}}(-3)$, $\mu_{\mathcal{Z}}(-3)$, $\mu_{\mathcal{P}}(-3)$.
- Determine the fired rules and their output membership strengths (α_i).
- Apply TSUKAMOTO's defuzzification to compute the crisp output heating power y^* .

TSUKAMOTO's formula:

$$y^* = \frac{\sum \alpha_i \cdot y_i}{\sum \alpha_i},$$

where y_i is the crisp output for rule i , and α_i is the strength of rule i (min of antecedents).

Part (a): Membership Degrees For Error = -3°C :

- ▶ $\mu_N(-3) = 0.6$
- ▶ $\mu_Z(-3) = 0.4$
- ▶ $\mu_P(-3) = 0$

Part (b): Fired Rules and Strengths (α_i)

$$\mathcal{R}_1 \text{ IF } \mathcal{N}, \alpha_1 = \mu_N(-3) = 0.6 \rightarrow y_1 = 10.$$

$$\mathcal{R}_2 \text{ IF } \mathcal{Z}, \alpha_2 = \mu_Z(-3) = 0.4 \rightarrow y_2 = 50.$$

$$\mathcal{R}_3 \text{ IF } \mathcal{P}, \alpha_3 = \mu_P(-3) = 0 \rightarrow \text{Not fired.}$$

Part (c): TSUKAMOTO's Defuzzification

$$y^* = \frac{\alpha_1 \times y_1 + \alpha_2 \times y_2}{\alpha_1 + \alpha_2} = \frac{0.6 \times 10 + 0.4 \times 50}{0.6 + 0.4} = \frac{6 + 20}{1} = 26\%.$$

The crisp heating power output is 26%.

A slightly negative error suggests low heat, weighted toward 10% but pulled up by the \mathcal{Z} membership.

APPLICATIONS OF FUZZY LOGIC

Control Systems:

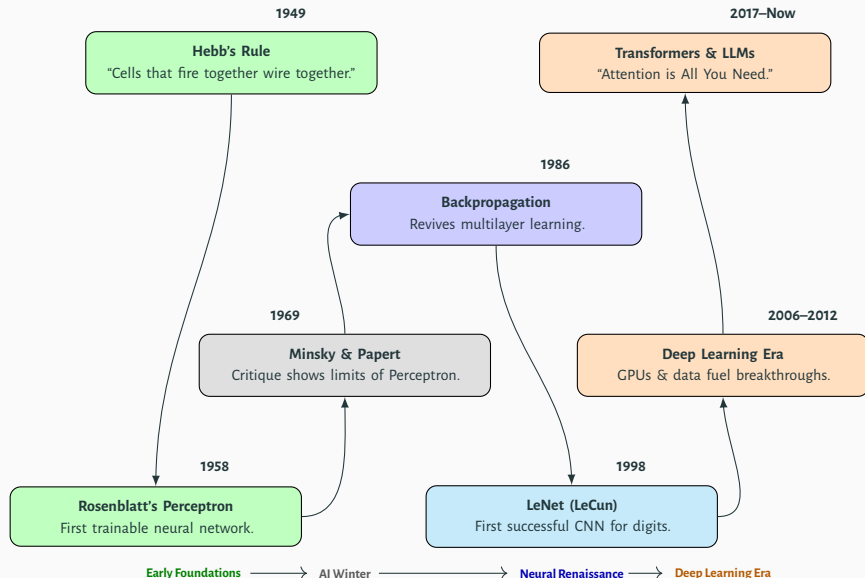
- ▶ Air conditioning
- ▶ Washing machines
- ▶ Automotive systems
- ▶ Robotics

Decision Support:

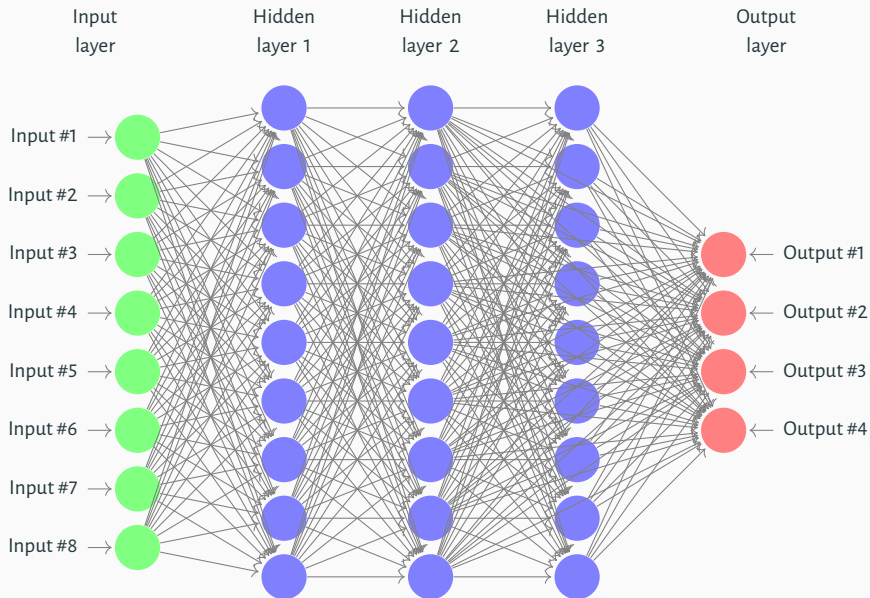
- ▶ Medical diagnosis
- ▶ Financial analysis
- ▶ Pattern recognition
- ▶ Expert systems

Neural Networks

TIMELINE OF NEURAL NETWORK MILESTONES

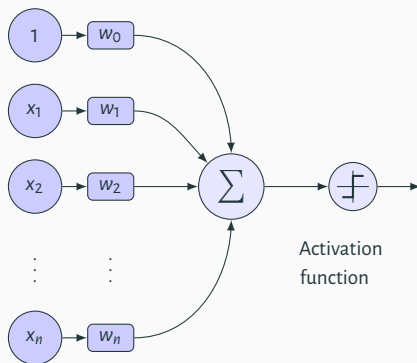


GENERAL ARCHITECTURE



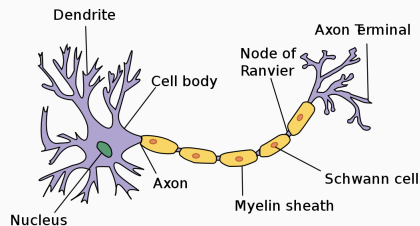
FUNDAMENTAL UNIT OF A NEURAL NETWORK (1/4)

Artificial neuron



Inputs Weights

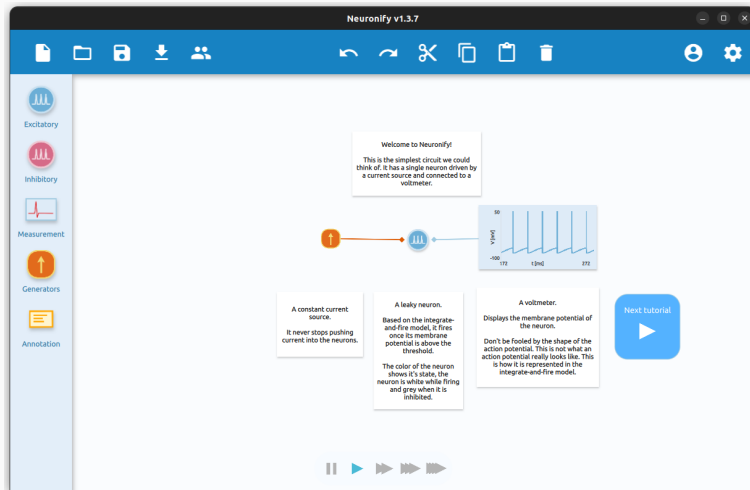
Biological neuron



https://id.wikipedia.org/wiki/Sel_saraf

FUNDAMENTAL UNIT OF A NEURAL NETWORK (2/4)

Neural simulation



<http://ovilab.net/neuronify/>

FUNDAMENTAL UNIT OF A NEURAL NETWORK (3/4)

Origin

- Proposed by **Warren McCulloch** (neuroscientist) and **Walter Pitts** (logician) in 1943;
- Published in the paper "*A Logical Calculus of the Ideas Immanent in Nervous Activity*";
- Marked the beginning of computational neuroscience.

Mechanism

$$y = \begin{cases} 1 & \text{if } \sum_i w_i x_i \geq \theta \\ 0 & \text{otherwise} \end{cases}$$

Properties

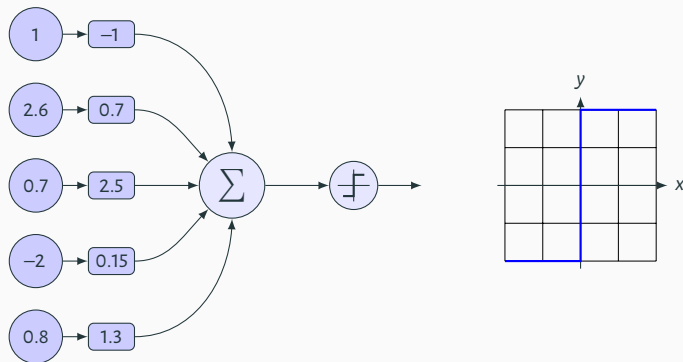
- ★ A simple mathematical model of a neuron;
- ★ Processes binary inputs (0 or 1) to produce a binary output;
- ★ Uses weighted summation of inputs;
- ★ Activates (outputs 1) if the sum meets or exceeds a threshold (θ).

McCulloch, W. S. and Pitts, W. (1943) *A logical calculus of the ideas immanent in nervous activity*. **The bulletin of mathematical biophysics**. 5 pp. 115–133.

FUNDAMENTAL UNIT OF A NEURAL NETWORK (4/4)

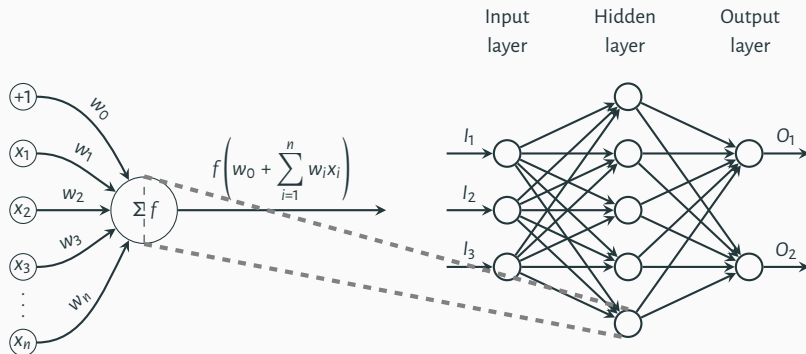
Task #6

Compute the output of the following neuron.



$$y = \text{sign}(1 \times -1 + 2.6 \times 0.7 + 0.7 \times 2.5 - 2 \times 0.15 + 0.8 \times 1.3) = 1$$

MULTILAYER PERCEPTRON (MLP) (1/2)

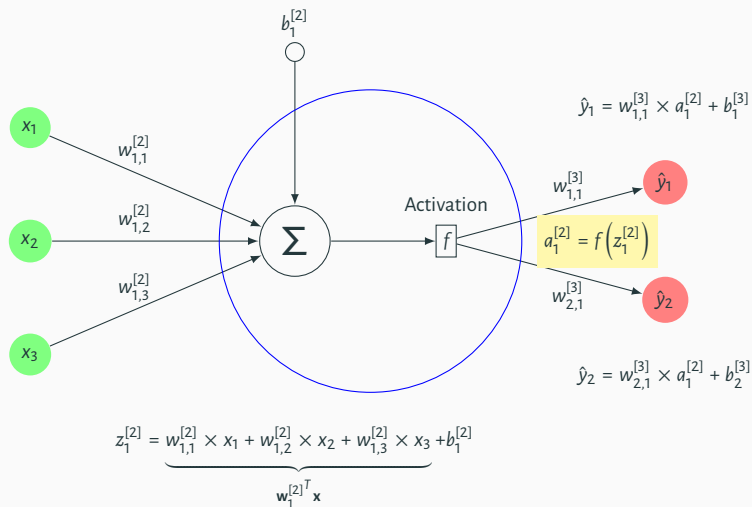


Task #7

For the above structure, determine how many parameters are to be adjusted.

$$\# \text{ params} = 5 \times 3 + 5 + 2 \times 5 + 2 = 32$$

MULTILAYER PERCEPTRON (MLP) (2/2)

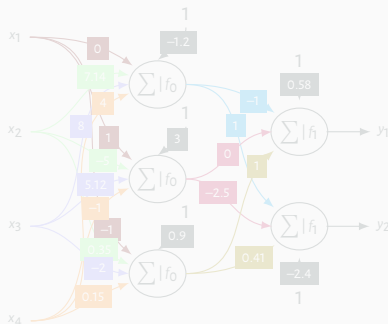


Task #8

Given the following weight matrices and biases vectors. Draw the corresponding neural network architecture. (Place the values of the synaptic weights and biases on the arrows.)

$$\mathcal{W}^{[1]} = \begin{bmatrix} 0 & 7.14 & 8 & 4 \\ 1 & -5 & 5.12 & -1 \\ -1 & 0.35 & -2 & 0.15 \end{bmatrix} \quad \text{and} \quad \mathbf{b}^{[1]} = \begin{bmatrix} -1.2 \\ 3 \\ 0.9 \end{bmatrix}$$

$$\mathcal{W}^{[2]} = \begin{bmatrix} -1 & 0 & 1 \\ 1 & -2.5 & 0.41 \end{bmatrix} \quad \text{and} \quad \mathbf{b}^{[2]} = \begin{bmatrix} 0.58 \\ -2.4 \end{bmatrix}$$

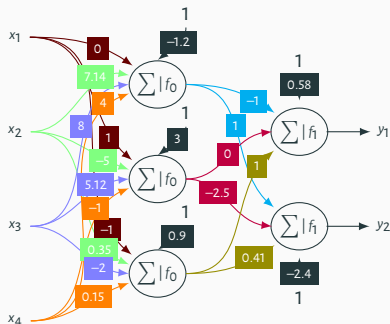


Task #8

Given the following weight matrices and biases vectors. Draw the corresponding neural network architecture. (Place the values of the synaptic weights and biases on the arrows.)

$$\mathcal{W}^{[1]} = \begin{bmatrix} 0 & 7.14 & 8 & 4 \\ 1 & -5 & 5.12 & -1 \\ -1 & 0.35 & -2 & 0.15 \end{bmatrix} \quad \text{and} \quad \mathbf{b}^{[1]} = \begin{bmatrix} -1.2 \\ 3 \\ 0.9 \end{bmatrix}$$

$$\mathcal{W}^{[2]} = \begin{bmatrix} -1 & 0 & 1 \\ 1 & -2.5 & 0.41 \end{bmatrix} \quad \text{and} \quad \mathbf{b}^{[2]} = \begin{bmatrix} 0.58 \\ -2.4 \end{bmatrix}$$



HEBBIAN LEARNING RULE

Origin Proposed by **Donald Hebb** (1949), summarized as:

“CELLS THAT FIRE TOGETHER WIRE TOGETHER.”

Mechanism

$$\Delta w_{ij} = \eta \cdot x_i \cdot y_j$$

where:

- Δw_{ij} : Change in the weight between neurons i and j
- η : Learning rate
- x_i : Activation of neurons i
- y_j : True label

Properties

- ★ Biologically inspired and unsupervised;
- ★ Encourages associative memory and pattern recognition;
- ★ No normalization mechanism; weights can grow unbounded.

Hebb D. (1949). *The organisation of behaviour: A Neuropsychological Theory* **Psychology Press**.

ROSENBLATT LEARNING RULE (PERCEPTRON LEARNING RULE)

Origin Introduced by **Frank Rosenblatt** (1958) with the Perceptron model.

Mechanism


$$\Delta w_{ij} = \eta \cdot x_i \cdot (y_j - \hat{y}_j)$$

where:

- Δw_{ij} : Change in the weight for feature i and output j
- η : Learning rate
- x_i : Input feature value
- y_j : True label (target output)
- \hat{y}_j : Predicted label

Properties

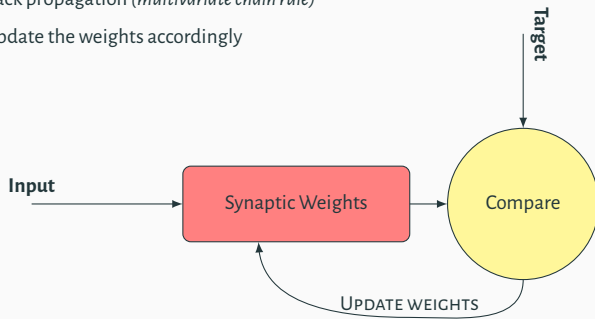
- ★ Supervised learning rule for binary classification;
- ★ Guarantees convergence if data is linearly separable;
- ★ Limited to linear decision boundaries.

Rosenblatt, F. (1958). *The perceptron: A probabilistic model for information storage and organization in the brain*. **Psychological Review**. 65(6), 386–408.  10.1037/h0042519

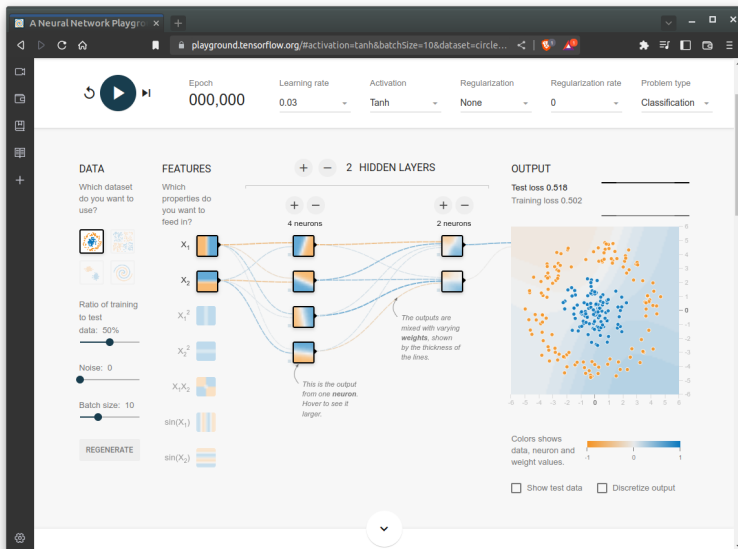
KEY DIFFERENCES

Aspect	Hebbian Learning Rule	Rosenblatt Learning Rule
Type of Learning	Unsupervised	Supervised
Primary Goal	Strengthen correlated activations	Minimize classification errors
Weight Update	Based on correlation of inputs	Based on prediction errors
Applications	Associative memory and pattern recognition	Training perceptrons and related linear classifiers.
Biological Basis	Inspired by neuroscience	Abstract, focused on AI

- ✓ Design a structure
- ✓ Specify a loss function to minimize
- ✓ Optimize using gradient descent
 - ① Feedforward propagation (*matrix multiplication and point-wise activation*)
 - ② Back propagation (*multivariate chain rule*)
 - ③ Update the weights accordingly



TINKER WITH A NEURAL NETWORK



<https://playground.tensorflow.org/>

REGRESSION VS CLASSIFICATION

Regression

- ▶ Predicts **continuous** values
- ▶ Output: Real numbers
- ▶ Examples:
 - House prices
 - Temperature
 - Stock prices
 - Age prediction

$$y \in \mathbb{R} \text{ (e.g., \$250,000)}$$

Classification

- ▶ Predicts **discrete** categories
- ▶ Output: Class labels
- ▶ Examples:
 - Spam/Not spam
 - Disease diagnosis
 - Image recognition
 - Sentiment analysis

$$y \in \{C_1, C_2, \dots, C_n\} \text{ (e.g., \{Yes, No\})}$$

LOSS VS COST FUNCTION

y denotes true labels, \hat{y} denotes predictions and $\varepsilon = y - \hat{y}$ is the prediction error.

Loss Function

- Error on a single training example

$$\mathcal{L}_i^{(j)} = \frac{1}{2} \left(y_i^{(j)} - \hat{y}_i^{(j)} \right)^2 = \frac{1}{2} \left(\varepsilon_i^{(j)} \right)^2,$$

where $y_i^{(j)}$ is the i -th label of the j -th sample.

Cost Function

- Aggregate error across the entire dataset

$$\mathcal{J} = \frac{1}{n} \sum_j \sum_i \mathcal{L}_i^{(j)}$$

Loss → one example | Cost → all examples

LOSS FUNCTIONS AND CORRESPONDING DERIVATIVES (1/2)

Mean Squared Error (MSE/L2)

commonly used for regression

$$\mathcal{L} = \frac{1}{2} \sum_i (y_i - \hat{y}_i)^2$$
$$\frac{\partial \mathcal{L}}{\partial \hat{y}_i} = -(y_i - \hat{y}_i)$$

Mean Absolute Error (MAE/L1)

robust regression alternative

$$\mathcal{L} = \sum_i |y_i - \hat{y}_i|$$
$$\frac{\partial \mathcal{L}}{\partial \hat{y}_i} = -\text{sign}(y_i - \hat{y}_i)$$

LOSS FUNCTIONS AND CORRESPONDING DERIVATIVES (2/2)

Binary Cross-Entropy (Log Loss)

for binary classification

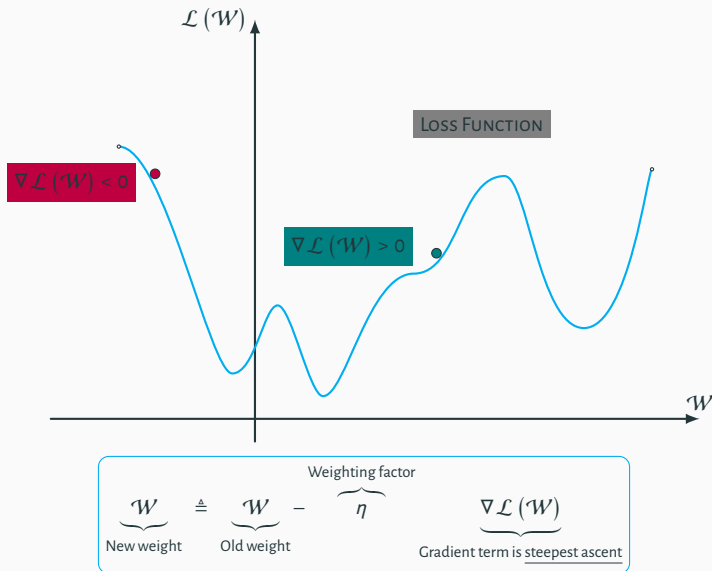
$$\mathcal{L} = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$$
$$\frac{\partial \mathcal{L}}{\partial \hat{y}} = -\left(\frac{y}{\hat{y}} - \frac{1-y}{1-\hat{y}}\right)$$

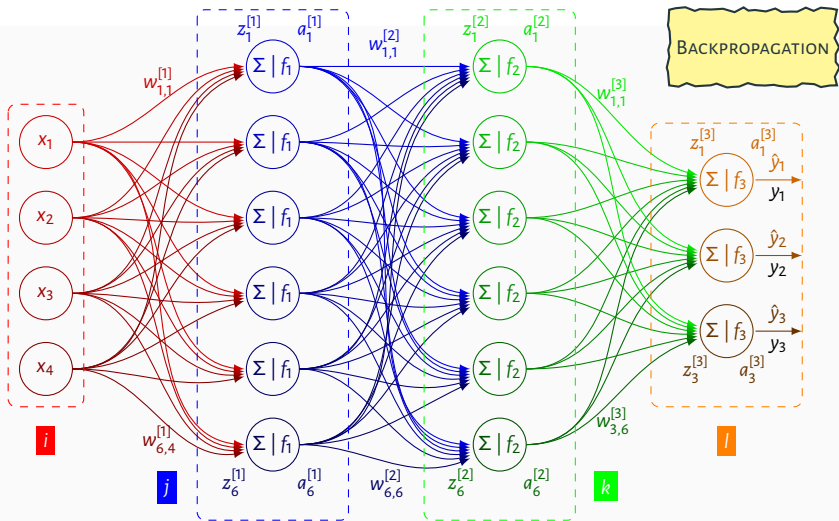
Categorical Cross-Entropy

for multi-class classification

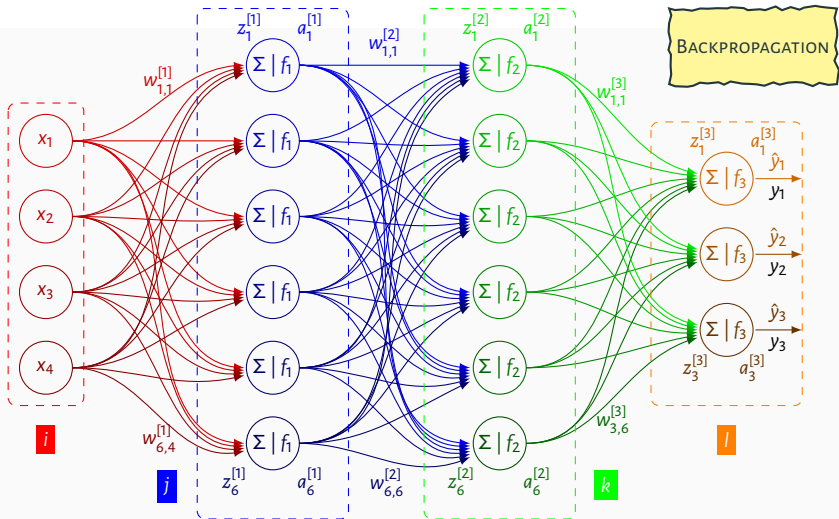
$$\mathcal{L} = -\sum_i y_i \log(\hat{y}_i) = -\log\left(\prod_i \hat{y}_i^{y_i}\right)$$
$$\frac{\partial \mathcal{L}}{\partial \hat{y}_i} = -\frac{y_i}{\hat{y}_i}$$

GRADIENT DESCENT

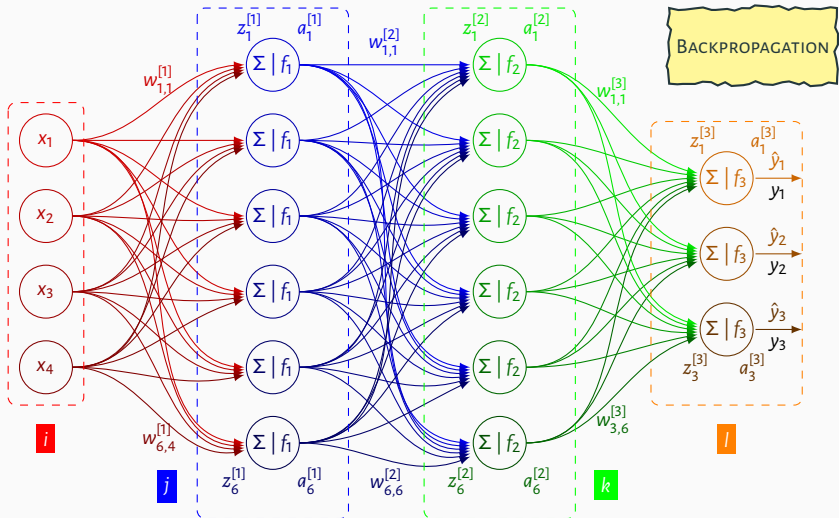




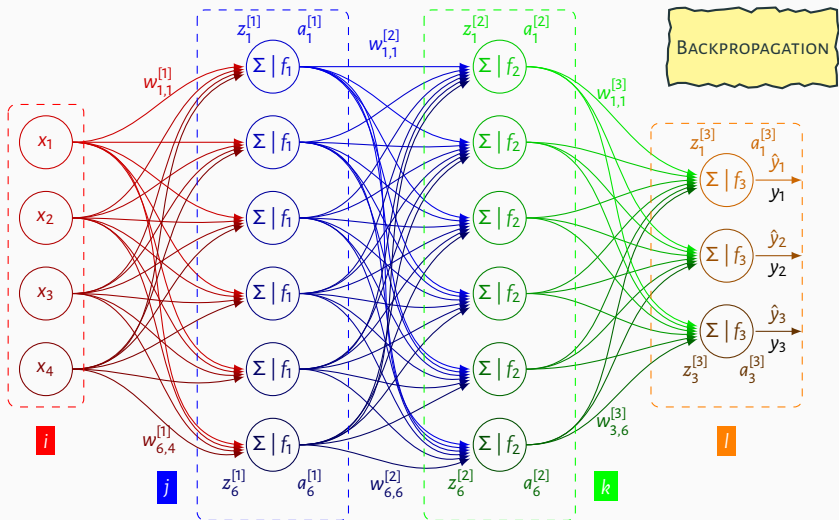
$$\mathcal{L}(\mathcal{W}) = \frac{1}{2} \sum_l (y_l - \hat{y}_l)^2 \Rightarrow \frac{\partial \mathcal{L}}{\partial w_{l,k}^{[3]}} = - \underbrace{(y_l - \hat{y}_l) \dot{f}_3(z_l^{[3]})}_{\delta_l^{[3]}} a_k^{[2]}$$



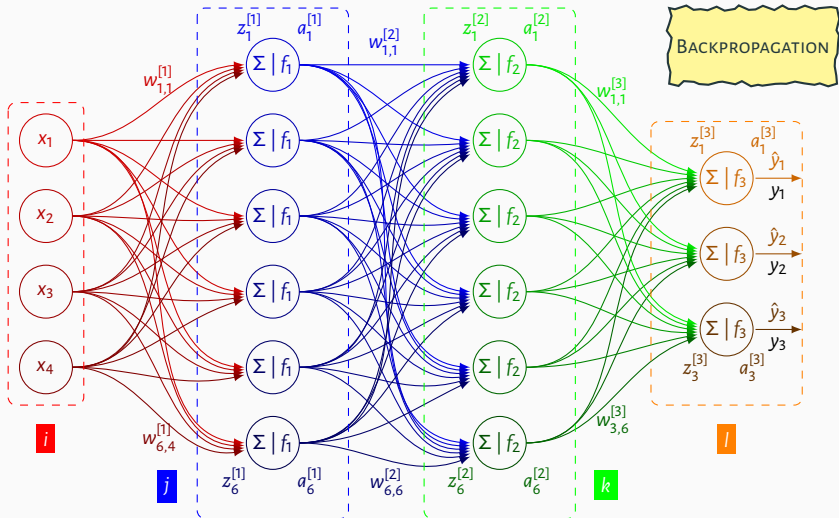
$$\frac{\partial \mathcal{L}}{\partial w_{k,j}^{[2]}} = - \sum_l (y_l - \hat{y}_l) \tilde{f}_3(z_l^{[3]}) w_{l,k}^{[3]} \tilde{f}_2(z_k^{[2]}) a_j^{[1]} = - \underbrace{\sum_l \delta_l^{[3]} w_{l,k}^{[3]} \tilde{f}_2(z_k^{[2]}) a_j^{[1]}}_{\delta_k^{[2]}}$$



$$\delta_l^{[3]} = (y_l - \hat{y}_l) \times \dot{f}_3(z_l^{[3]}) \implies \Delta w_{l,k}^{[3]} = \eta \delta_l^{[3]} \times a_k^{[2]}$$



$$\delta_k^{[2]} = \left(\delta_1^{[3]} w_{1,k}^{[2]} + \delta_2^{[3]} w_{2,k}^{[2]} + \delta_3^{[3]} w_{3,k}^{[2]} \right) \times \dot{f}_2 \left(z_k^{[2]} \right) \implies \Delta w_{k,j}^{[2]} = \eta \delta_k^{[2]} \times a_j^{[1]}$$



$$\delta_j^{[1]} = \left(\delta_1^{[2]} w_{1,j}^{[1]} + \dots + \delta_6^{[2]} w_{6,j}^{[1]} \right) \times \dot{f}_2 \left(z_j^{[2]} \right) \implies \Delta w_{j,i}^{[1]} = \eta \delta_j^{[1]} \times x_i$$

MULTIVARIATE CHAIN RULE

Output layer → hidden layer #2

$$\frac{\partial \hat{y}_l}{\partial w_{l,k}^{[3]}} = \underbrace{\frac{\partial \hat{y}_l}{\partial z_l^{[3]}}}_{f_3'(z_l^{[3]})} \underbrace{\frac{\partial z_l^{[3]}}{\partial w_{l,k}^{[3]}}}_{a_k^{[2]}}$$

Output layer → hidden layer #1

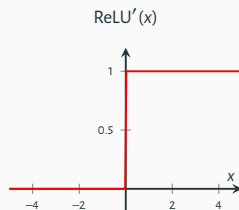
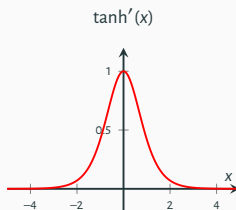
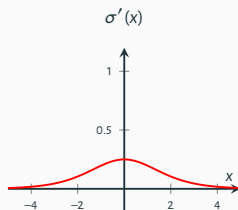
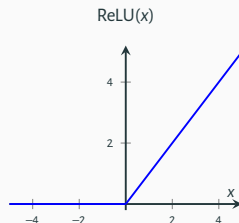
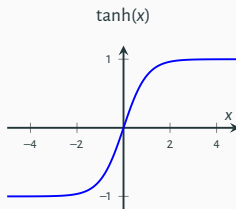
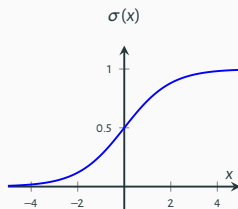
$$\frac{\partial \hat{y}_l}{\partial w_{k,j}^{[2]}} = \underbrace{\frac{\partial \hat{y}_l}{\partial z_l^{[3]}}}_{f_3'(z_l^{[3]})} \underbrace{\frac{\partial z_l^{[3]}}{\partial a_k^{[2]}}}_{w_{l,k}^{[3]}} \underbrace{\frac{\partial a_k^{[2]}}{\partial z_k^{[2]}}}_{f_2'(z_k^{[2]})} \underbrace{\frac{\partial z_k^{[2]}}{\partial w_{k,j}^{[2]}}}_{a_j^{[1]}}$$

Output layer → input layer

$$\frac{\partial \hat{y}_l}{\partial w_{j,i}^{[1]}} = \underbrace{\frac{\partial \hat{y}_l}{\partial z_l^{[3]}}}_{f_3'(z_l^{[3]})} \underbrace{\frac{\partial z_l^{[3]}}{\partial a_k^{[2]}}}_{w_{l,k}^{[3]}} \underbrace{\frac{\partial a_k^{[2]}}{\partial z_k^{[2]}}}_{f_2'(z_k^{[2]})} \underbrace{\frac{\partial z_k^{[2]}}{\partial a_j^{[1]}}}_{w_{k,j}^{[2]}} \underbrace{\frac{\partial a_j^{[1]}}{\partial z_j^{[1]}}}_{f_1'(z_j^{[1]})} \underbrace{\frac{\partial z_j^{[1]}}{\partial w_{j,i}^{[1]}}}_{x_i}$$

ACTIVATION FUNCTIONS AND THEIR DERIVATIVES (1/2)

Each function (blue) and its derivative (red) are shown.



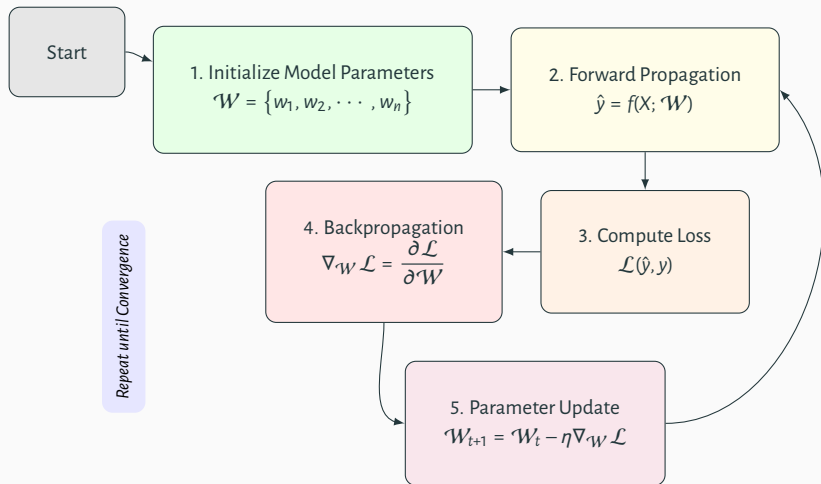
σ and \tanh saturate for large $|x|$, while ReLU avoids this but is non-smooth at $x = 0$.

ACTIVATION FUNCTIONS AND THEIR DERIVATIVES (2/2)


Function	Formula	Derivative	Notes
Sigmoid	$\sigma(x) = \frac{1}{1 + e^{-x}}$	$\sigma'(x) = \sigma(x)(1 - \sigma(x))$	Smooth, bounded
Tanh	$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$\tanh'(x) = 1 - \tanh^2(x)$	Zero-centered
ReLU	$\text{ReLU}(x) = \max(0, x)$	$\text{ReLU}'(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases}$	Sparse gradients

Activation functions introduce non-linearity, enabling neural networks to approximate complex mappings.

NEURAL NETWORK TRAINING



BACKPROPAGATION BY HAND



Matt Mazur

I'm an indie founder building and writing about **Preceden**, a SaaS timeline maker, and **Emergent Mind**, an AI research assistant for computer scientists.

- Home
- Projects
- About
- Contact
- Twitter

A Step by Step Backpropagation Example

March 17, 2015




Background

Backpropagation is a common method for training a neural network. There is **no shortage of papers** online that attempt to explain how backpropagation works, but few that include an example with actual numbers. This post is my attempt to explain how it works with a concrete example that folks can compare their own calculations to in order to ensure they understand backpropagation correctly.

Backpropagation in Python

You can play around with a Python script that I wrote that implements the backpropagation algorithm in **this Github repo**.

Continue learning with Emergent Mind

 Comment
  Reblog
  Subscribe
 ...

<https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/>

LIST OF AVAILABLE OPTIMIZERS (1/3)

These are a few typical optimizers for artificial neural networks:

$$\Delta \hat{\mathcal{W}} \triangleq \mathcal{F} \left(\underbrace{\nabla \mathcal{L}(\hat{\mathcal{W}})}_{\text{Loss Function}} \right) \equiv \hat{\mathcal{W}} \triangleq \hat{\mathcal{W}} + \mathcal{F}(\nabla \mathcal{L}(\hat{\mathcal{W}})) \quad \nabla \mathcal{L}(\hat{\mathcal{W}}) = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial \hat{w}_0} \\ \vdots \\ \frac{\partial \mathcal{L}}{\partial \hat{w}_n} \end{bmatrix}$$

Stochastic Gradient Descent (SGD)

$$\hat{\mathcal{W}} \triangleq \hat{\mathcal{W}} - \eta \nabla \mathcal{L}(\hat{\mathcal{W}})$$

Mini-batch Gradient Descent

$$\hat{\mathcal{W}} \triangleq \hat{\mathcal{W}} - \frac{\eta}{m} \nabla \sum_{i=1}^m \mathcal{L}(\hat{\mathcal{W}}) \quad \longleftarrow m \text{ denotes the size of the mini-batch}$$

LIST OF AVAILABLE OPTIMIZERS (2/3)

Momentum

$$\hat{\mathcal{W}} \triangleq \hat{\mathcal{W}} - \mathcal{V}, \quad \text{where} \quad \mathcal{V} \triangleq \alpha \mathcal{V} + \eta \nabla \mathcal{J}(\hat{\mathcal{W}})$$

Nesterov Accelerated Gradient (NAG)

$$\hat{\mathcal{W}} \triangleq \hat{\mathcal{W}} - \mathcal{V} \quad \text{where} \quad \mathcal{V} \triangleq \alpha \mathcal{V} + \eta \nabla \mathcal{J}(\hat{\mathcal{W}} - \alpha \mathcal{V})$$

AdaGrad

$$\hat{\mathcal{W}} \triangleq \hat{\mathcal{W}} - \frac{\eta}{\sqrt{\mathcal{G} + \epsilon}} \nabla \mathcal{J}(\hat{\mathcal{W}}) \quad \text{where} \quad \mathcal{G} \triangleq \mathcal{G} + \left(\nabla \mathcal{J}(\hat{\mathcal{W}}) \right)^2$$

RMSProp

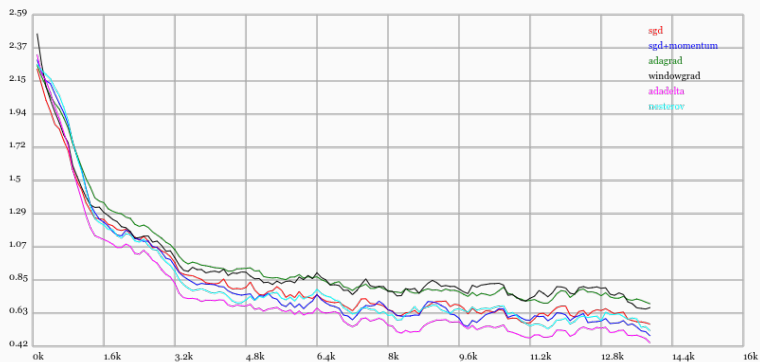
$$\hat{\mathcal{W}} \triangleq \hat{\mathcal{W}} - \frac{\eta}{\sqrt{\mathcal{G} + \epsilon}} \nabla \mathcal{J}(\hat{\mathcal{W}}) \quad \text{where} \quad \mathcal{G} \triangleq \mathcal{G} + (1 - \beta) \left(\nabla \mathcal{J}(\hat{\mathcal{W}}) \right)^2$$

LIST OF AVAILABLE OPTIMIZERS (3/3)

Adam

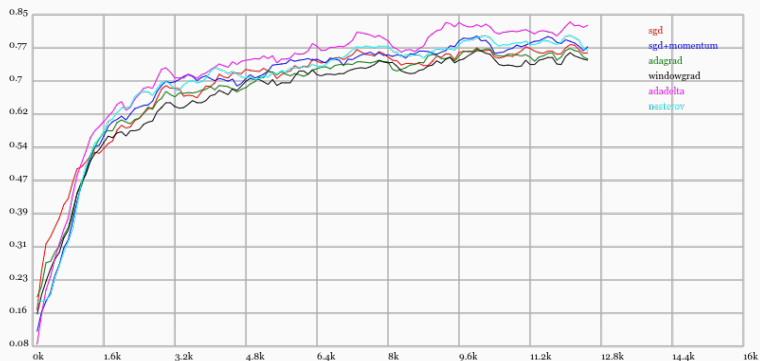
$$\begin{aligned}
 \mathcal{M} &\triangleq \beta_1 \mathcal{M} + (1 - \beta_1) \nabla \mathcal{J}(\hat{\mathcal{W}}) && \longleftarrow \text{Estimate of first moment} \\
 \mathcal{V} &\triangleq \beta_2 \mathcal{V} + (1 - \beta_2) \left(\nabla \mathcal{J}(\hat{\mathcal{W}}) \right)^2 && \longleftarrow \text{Estimate of second moment} \\
 \hat{\mathcal{M}} &= \frac{\mathcal{M}}{1 - \beta_1^k} && \longleftarrow \text{@ every } k^{\text{th}} \text{ iteration} \\
 \hat{\mathcal{V}} &= \frac{\mathcal{V}}{1 - \beta_2^k} && \longleftarrow \text{@ every } k^{\text{th}} \text{ iteration} \\
 \hat{\mathcal{W}} &\triangleq \hat{\mathcal{W}} - \frac{\eta}{\sqrt{\hat{\mathcal{V}} + \epsilon}} \hat{\mathcal{M}}
 \end{aligned}$$

EFFECT OF OPTIMIZER ON LOSS VALUES



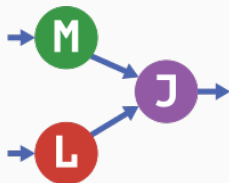
<https://cs.stanford.edu/people/karpathy/convnetjs/demo/trainers.html>

EFFECT OF OPTIMIZER ON TESTING ACCURACY VALUES



<https://cs.stanford.edu/people/karpathy/convnetjs/demo/trainers.html>

FRAMEWORKS TO BE USED



<https://juliapackages.com/p/mlj>



<https://juliapackages.com/p/flux>



The code is available @ github.com/a-mhamdi/jlai → *Codes* → *Julia* → *Part-1*

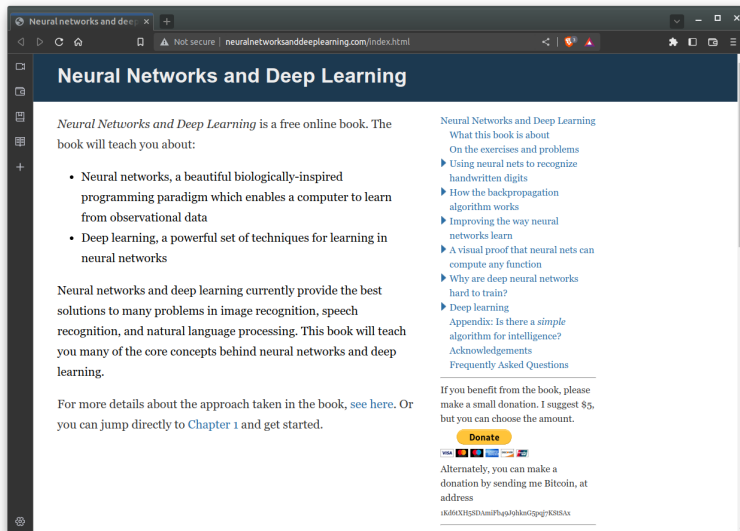
→ *Pluto* → *xor-gate.jl*

Pluto.jl 

→ *Jupyter* → *xor-gate.ipynb*



- This will be continued in the following e-book.



<http://neuralnetworksanddeeplearning.com/>



NEURAL NETWORK FROM SCRATCH

The screenshot shows the GitHub repository page for 'a-mhamdi/neural-network-from-scratch-in-Julia'. The repository is public and has 1 star and 0 forks. The description states: 'Without using any deep learning frameworks, we construct and train a neural network architecture in Julia from the ground up.' The repository includes tags for 'neural-networks', 'backpropagation', and 'julia-lang'. The file list shows the following files and their commit dates:

File	Commit Message	Commit Date
Images	add versioninfo and pkgs st to README file	3 months ago
src	fix Random.seed value and update struct	3 months ago
.gitignore	ignore folder 'SITE-CONF'	4 months ago
LICENSE	Initial commit	last year
Project.toml	RDatasets	4 months ago
README.md	fix typos and minor updates	3 months ago
main.jl	fix Random.seed value and update struct	3 months ago

<https://github.com/a-mhamdi/neural-network-from-scratch-in-Julia>

Quizzes

KNOWLEDGE CHECK



1

Go to wooclap.com

2

Enter the event code in the top banner

Event code

JLAI1

<https://app.wooclap.com/JLAI1>

FURTHER READING (1/3)

References

- [Bel78] R. E. Bellman. ***An Introduction to Artificial Intelligence: Can Computers Think?*** Boyd & Fraser Publishing Company, Jan. 1, 1978 (cit. on p. 11).
- [CMM85] E. Charniak, D. McDermott, and D. V. McDermott. ***Introduction to Artificial Intelligence***. Addison-Wesley series in computer science and information processing. Addison-Wesley, 1985 (cit. on p. 12).
- [Dad12] E. Dadios, ed. ***Fuzzy Logic - Controls, Concepts, Theories and Applications***. IntechOpen, Mar. 28, 2012. 430 pp.
- [ENM15] I. El Naqa and M. J. Murphy. “**What Is Machine Learning?**” In: *Machine Learning in Radiation Oncology: Theory and Applications*. Ed. by I. El Naqa, R. Li, and M. J. Murphy. Cham: Springer International Publishing, 2015, pp. 3–11. DOI: 10.1007/978-3-319-18305-3_1.
- [Gac15] L. Gacogne. ***Intelligence artificielle. Cours, exercices corrigés et projets***. Ed. by Ellipses. Nov. 2015. 240 pp.
- [GBC16] I. Goodfellow, J. Bengio, and A. Courville. ***Deep Learning***. MIT Press Ltd, Nov. 18, 2016. 800 pp.

FURTHER READING (2/3)

- [Hau89] J. Haugeland. **Artificial Intelligence: The Very Idea**. A Bradford book. MIT Press, 1989 (cit. on p. 11).
- [JPM21] L. M. John Paul Mueller. **Machine Learning For Dummies**. Wiley John + Sons, Apr. 8, 2021. 464 pp.
- [Kur92] R. Kurzweil. **The Age of Intelligent Machines**. Viking, 1992 (cit. on p. 13).
- [LS93] G. F. Luger and W. A. Stubblefield. **Artificial Intelligence: Structures and Strategies for Complex Problem Solving**. Artificial intelligence. Benjamin/Cummings Publishing Company, 1993 (cit. on p. 14).
- [Mit97] T. M. Mitchell. **Machine Learning**. McGraw-Hill International Editions. McGraw-Hill, 1997.
- [RK91] E. Rich and K. Knight. **Artificial Intelligence**. Artificial Intelligence Series. McGraw-Hill, 1991 (cit. on p. 13).
- [Rob14] A. Robinson. **Construction Informatics**. Ed. by H. S. of Construction Engineering. 2014.
- [Sch90] R. J. Schalkoff. **Artificial Intelligence: An Engineering Approach**. McGraw-Hill Computer science series. McGraw-Hill, 1990 (cit. on p. 14).
- [SNK12] T. Sai, D. Nakhaeina, and B. Karasfi. **“Application of Fuzzy Logic in Mobile Robot Navigation”**. In: *Fuzzy Logic - Controls, Concepts, Theories and Applications*. InTech, Mar. 2012. DOI: 10.5772/36358.

FURTHER READING (3/3)

- [Win92] P. H. Winston. ***Artificial Intelligence***. A-W Series in Computer Science. Addison-Wesley Publishing Company, 1992 (cit. on p. 12).
- [Woj12] J. Wojtusiak. “**Machine Learning**”. In: *Encyclopedia of the Sciences of Learning*. Springer US, 2012, pp. 2082–2083. DOI: 10.1007/978-1-4419-1428-6_1927.