CO542
Neural Networks and Fuzzy Systems

NEURON GLOW

Presented to you by:
FUZZBUSTERS

OUR TEAM

FuzzBusters

Ravindu Suraweeraarachchi

E/19/393

Pinindu Thiwanka

E/19/407

Layani Senevirathne

E/19/365

Ruwindya Jayasekara

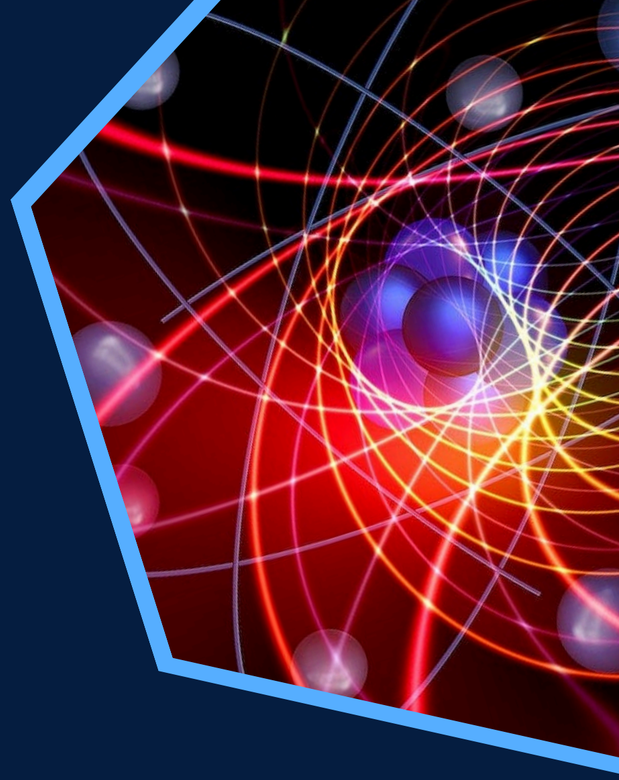
E/19/159

Janith Wanasinghe

E/20/420

OVERVIEW

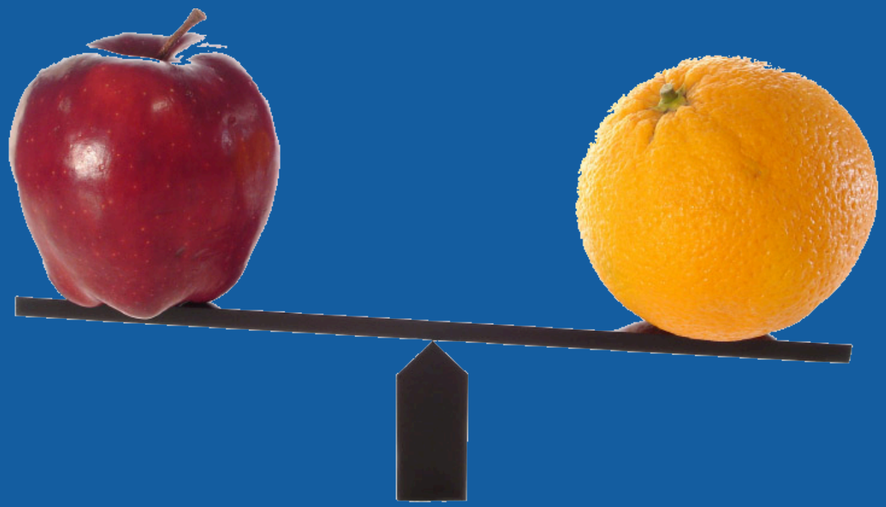
Our Key Talking Points



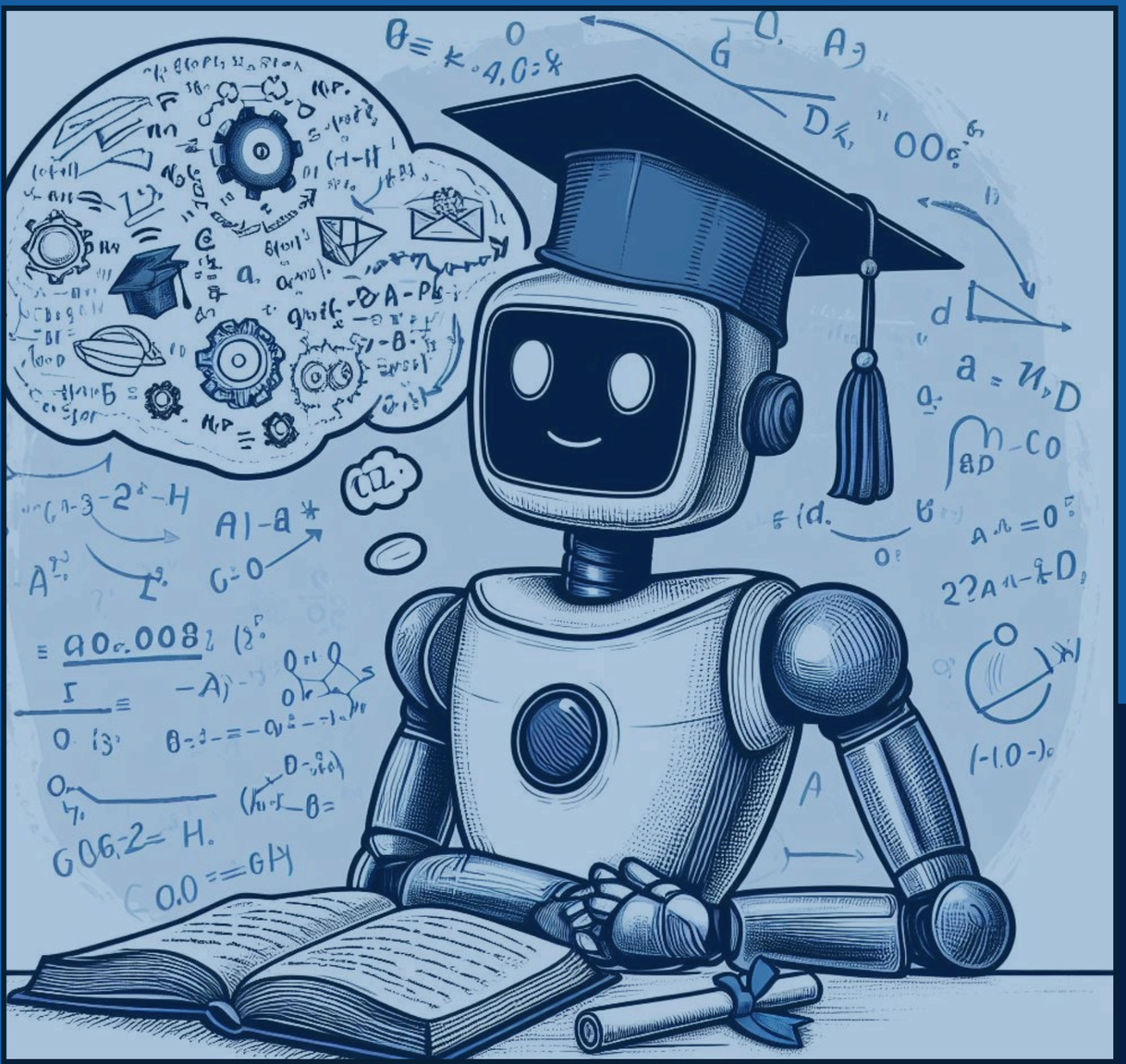
- Introduction
- Problem Definition
- Our Solution
- How it works
- Challenges
- Future Improvements
- Individual Contributions
- Conclusion

INTRODUCTION

How do we teach a toddler to recognize apples and oranges?



INTRODUCTION CONTD.



Computers learn the same way when it comes to ML!

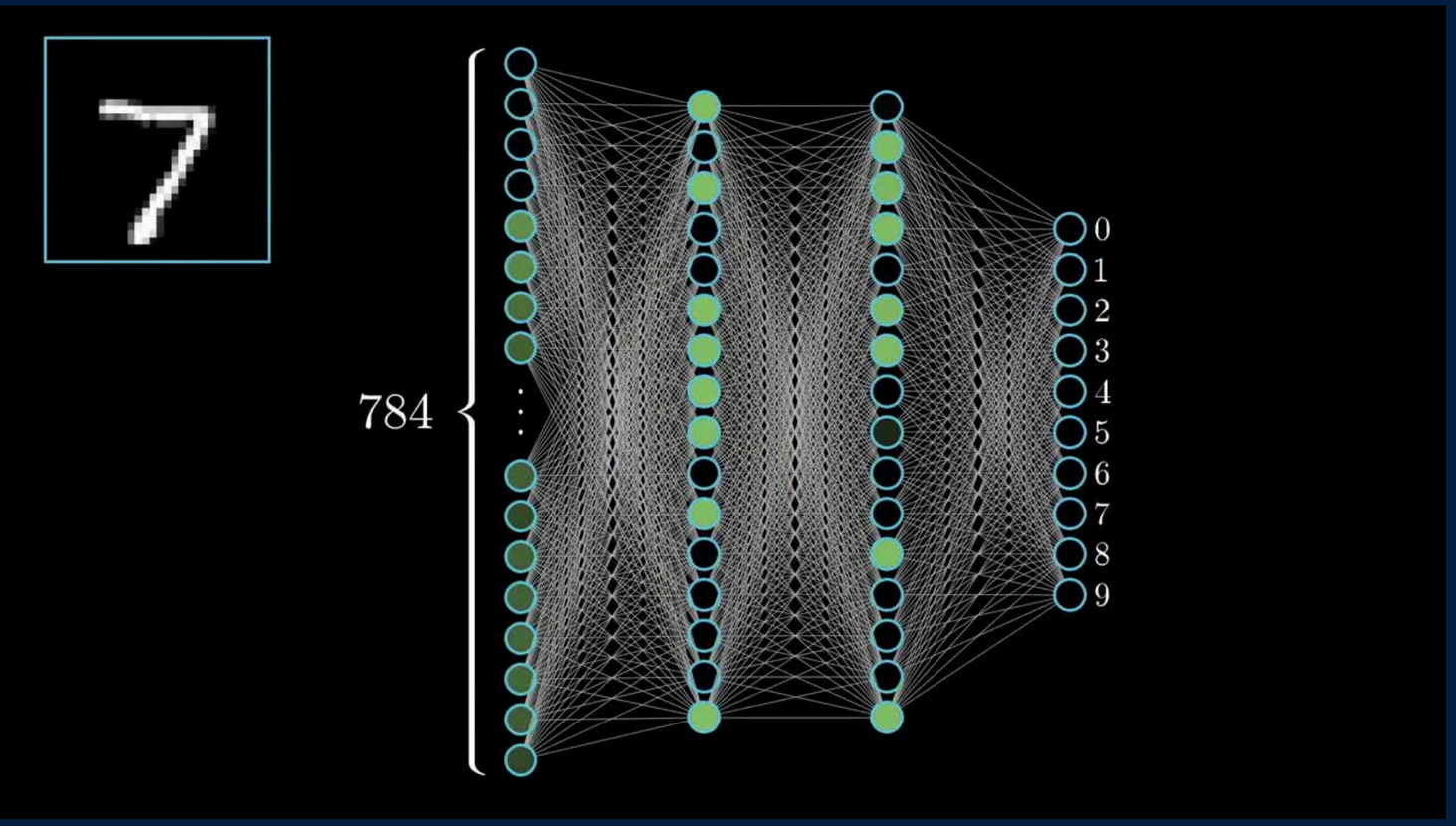
“Computers start with no knowledge, make mistakes, and improve over time”

WE AIM TO HELP PEOPLE UNDERSTAND HOW A MACHINE LEARNS

PROBLEM DEFINITION

How do machines learn?

- A computer improves at a task by learning from data
- One of the core techniques: **Neural Network**



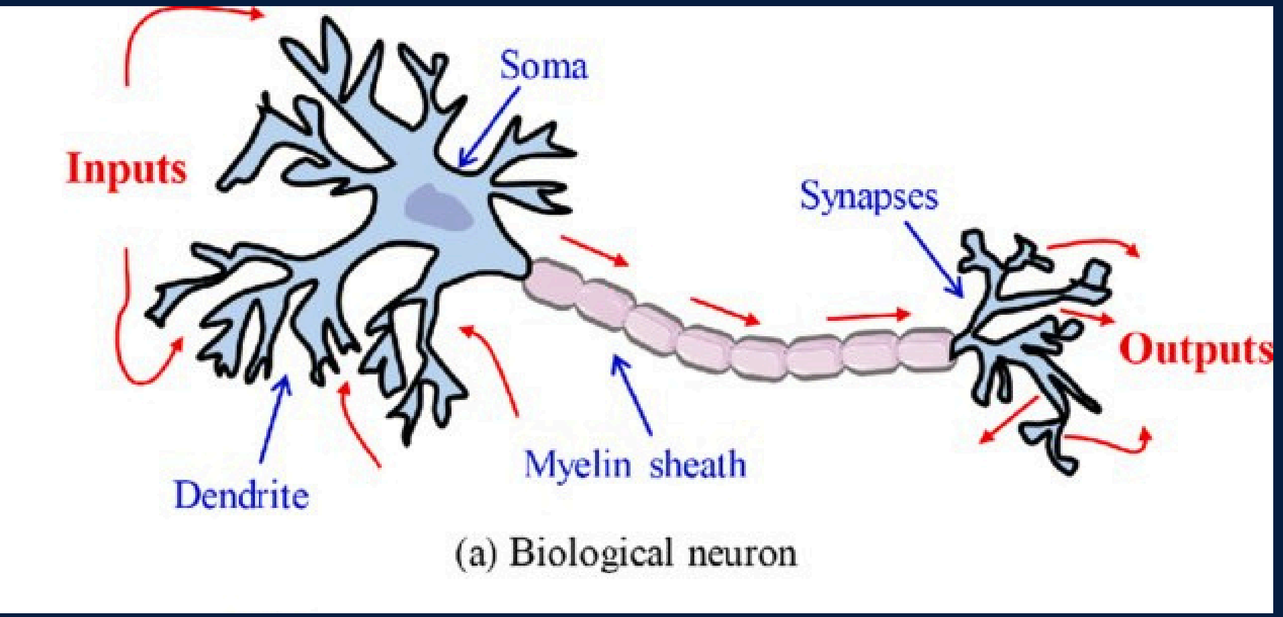
Source : https://youtu.be/aircAruvnKk?feature=shared (3Blue1Brown)

What is a Neural Network?

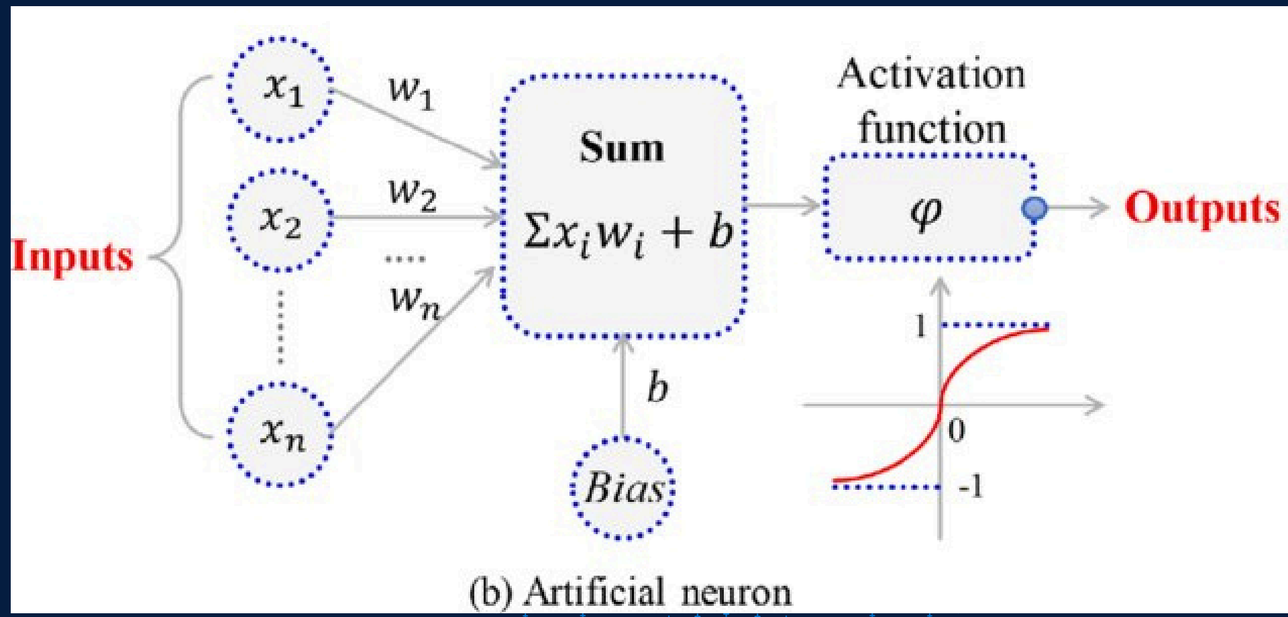
- Mimics how the human brain works
- Layers of Artificial **Neurons**
- Adjusts “weights” over time to make better decisions

PROBLEM DEFINITION

How Neural Networks Mimic the Brain



(a) Biological neuron



(b) Artificial neuron

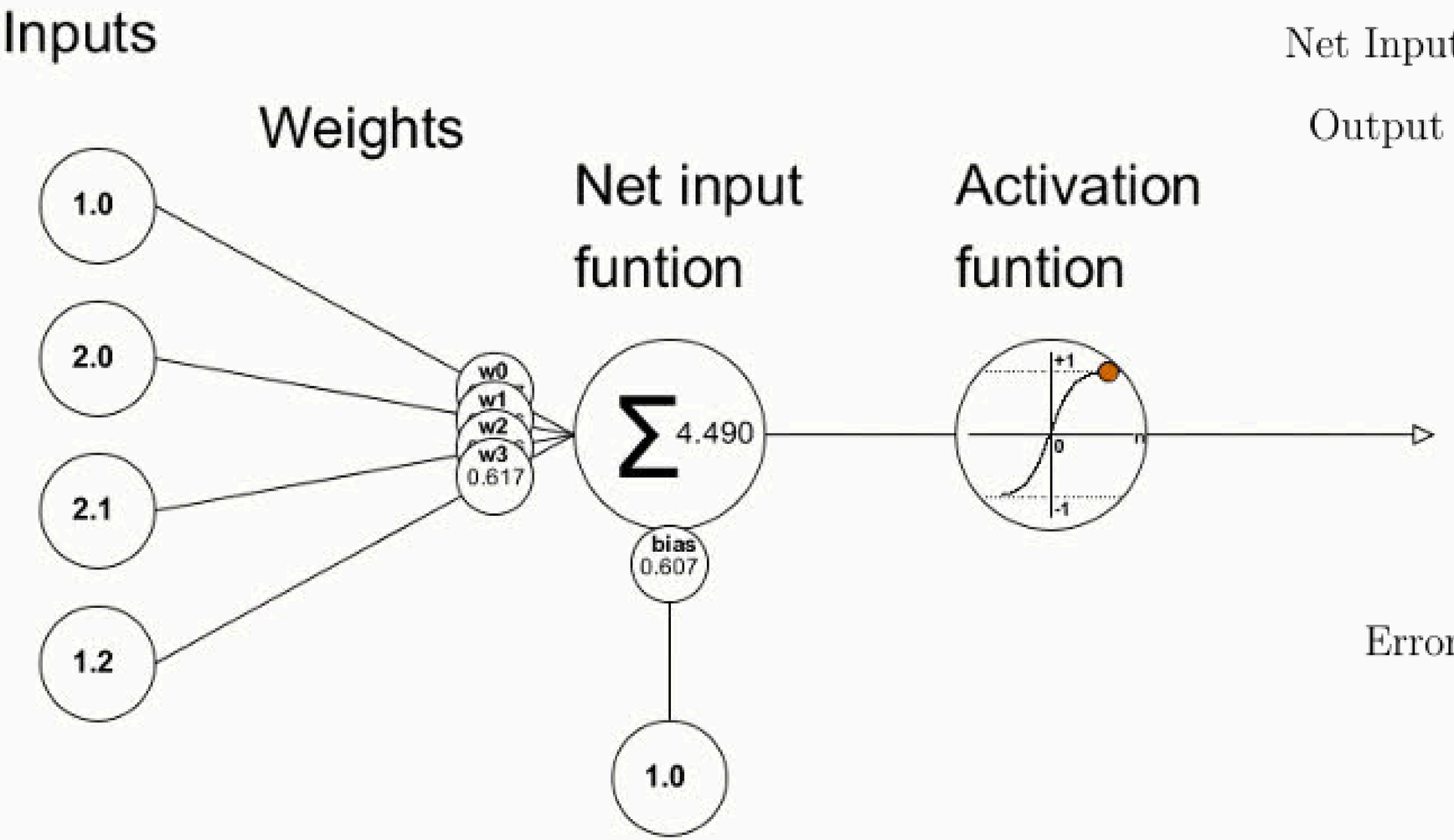
PROBLEM DEFINITION

- Learning Neural Networks is TRICKY!

OUR SOLUTION - A HANDS-ON LEARNING TOOL

- Hardware powered Single-Layer Perceptron
  - A simple AI model that learns and makes decisions.
- Perceptron Living in ARDUINO
  - A way to visualize the Learning Process

OUR SOLUTION - A HANDS-ON LEARNING TOOL



$$\text{Net Input} = w_0x_0 + w_1x_1 + w_2x_2 + w_3x_3 + b$$

$$\text{Output} = \text{Activation Function}(\text{Net Input})$$

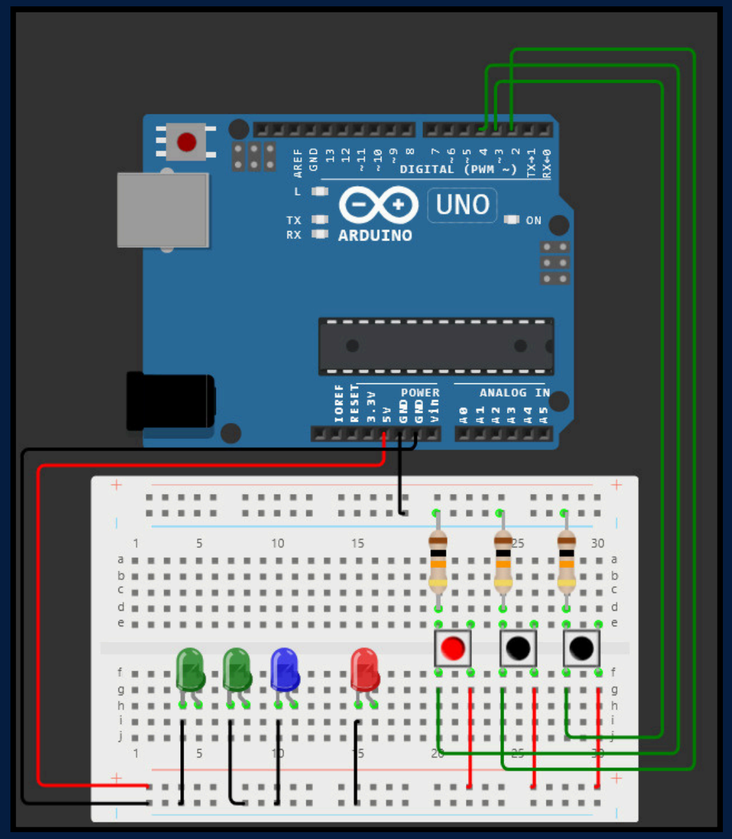
$$\text{Error} = \text{Output} - \text{Expected Output}$$

$$w_i = w_i + \eta \times x_i \times \text{Error}$$

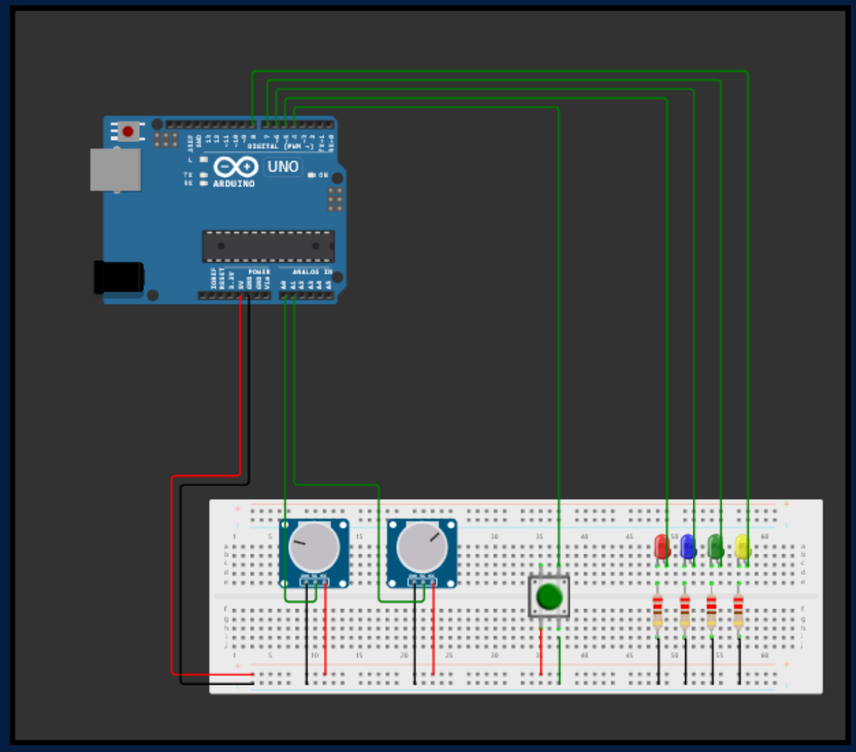
$$b = b + \eta \times \text{Error}$$

WHAT WE DID - IN A NUTSHELL

TRAINING “AND” GATE USING PERCEPTRON

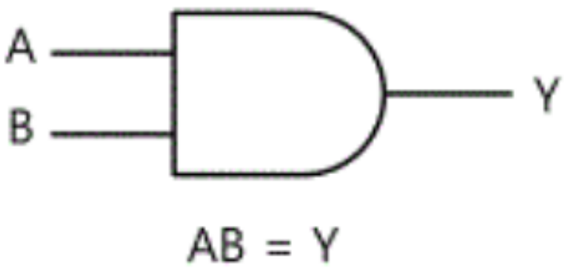


FUZZY-INSPIRED TRAINING: LEARNING OUTPUT BASED ON POTENTIOMETER RANGES

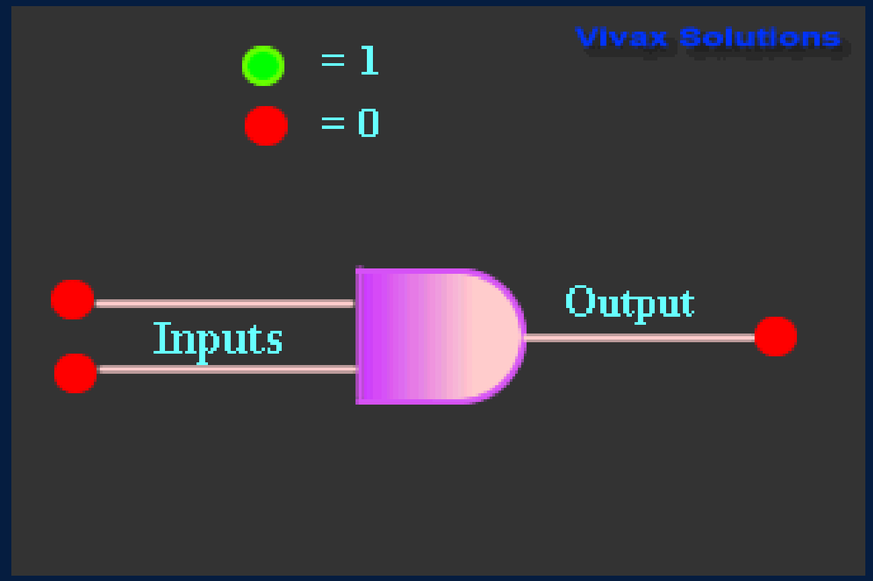


TRAINING “AND” GATE USING PERCEPTRON

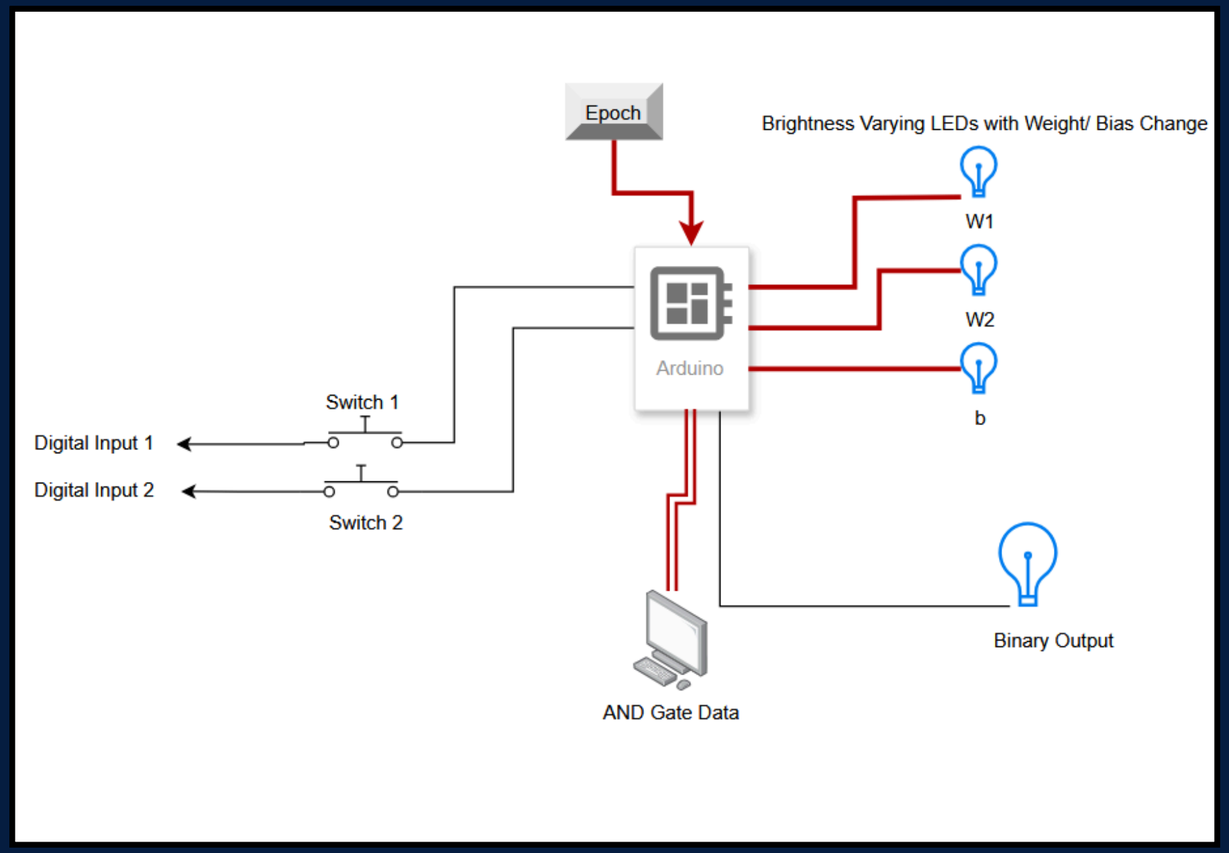
AND GATE



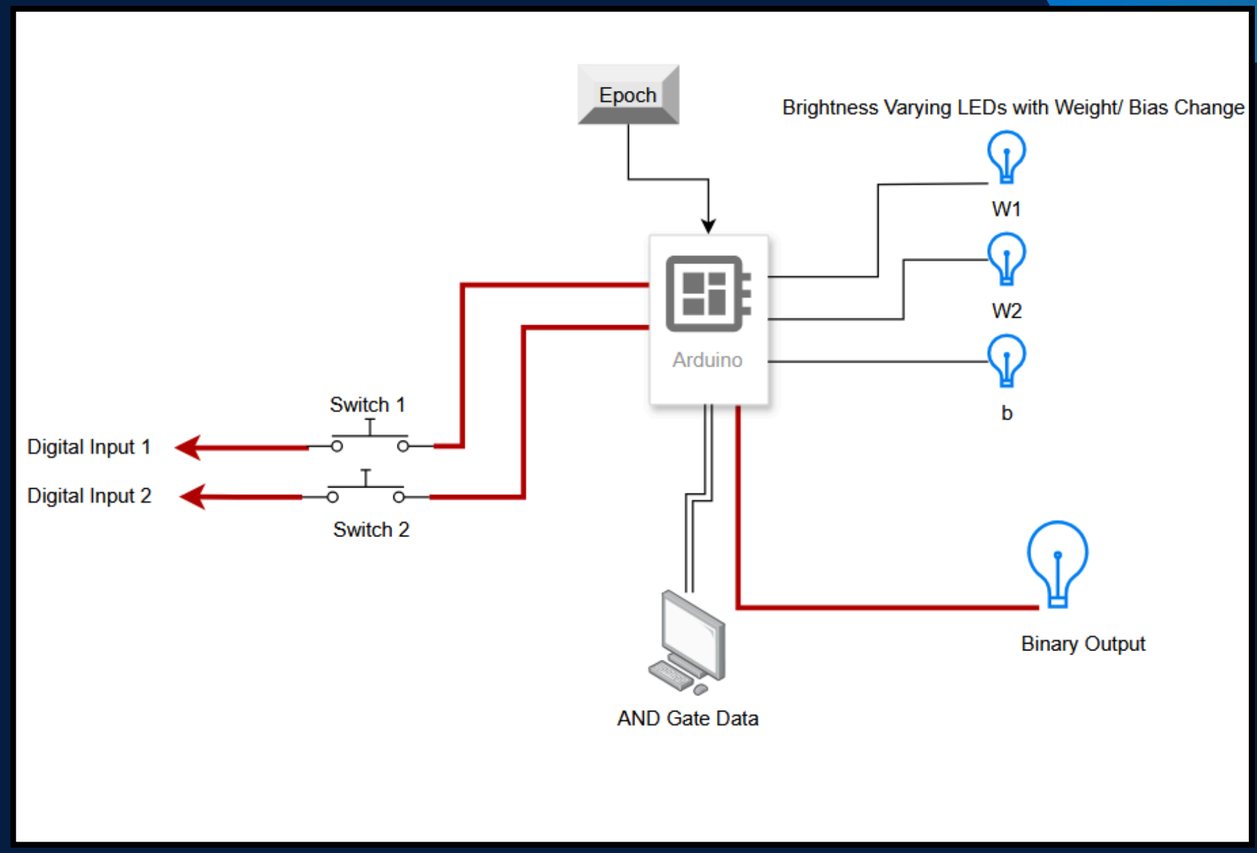
A	B	Y
Low	Low	Low
Low	High	Low
High	Low	Low
High	High	High



TRAINING “AND” GATE USING PERCEPTRON

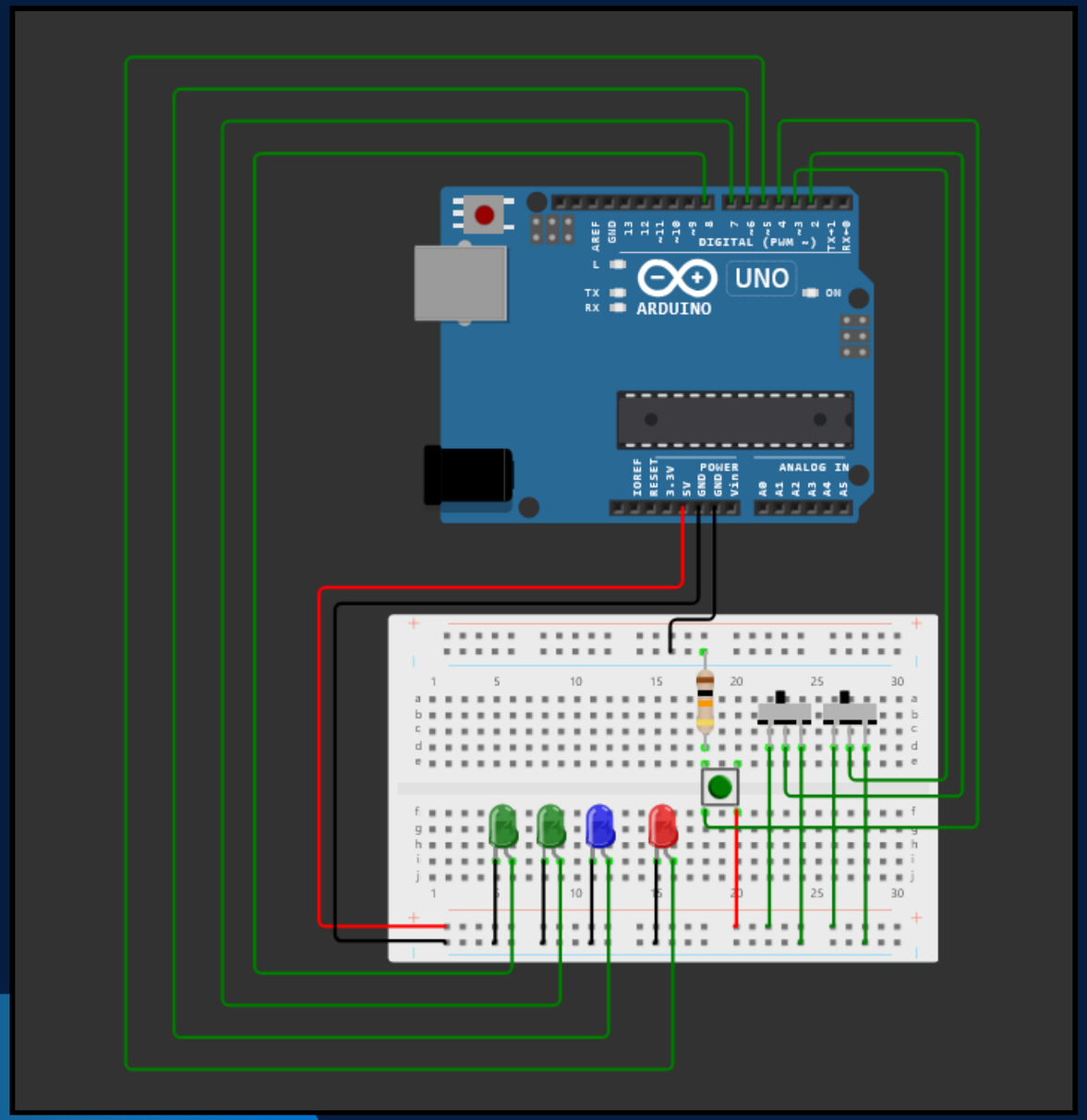


Training State



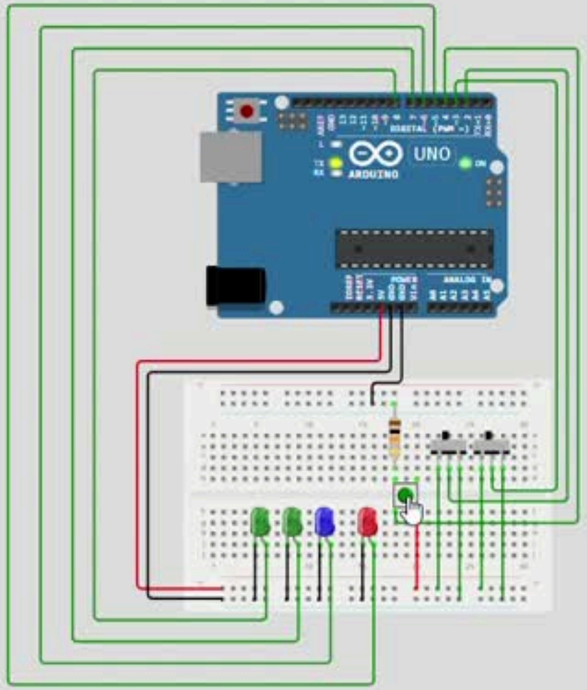
Predicting State

TRAINING “AND” GATE USING PERCEPTRON



- Button-Controlled Training Epochs
- Maps weights and bias to LED Brightness
- Initial weights/bias can be assigned
- Adjusts weights until it correctly mimics the AND gate behaviour

TRAINING “AND” GATE USING PERCEPTRON




```
Perceptron AND gate training ready. Press epoch button to train.

Epoch Started...
Epoch
1
Input: (0, 0) → Output: 1 | Expected: 0
Input: (0, 1) → Output: 1 | Expected: 0
```

FUZZY-INSPIRED TRAINING: LEARNING OUTPUT BASED ON POTENTIOMETER RANGES

- If the potentiometer values fall within a certain range
  - Eg: *pot1* between 0.1-0.4 AND *pot2* between 0.4-0.7,

    the output is trained to be 1;

    otherwise, it's trained to be 0.

- Adjusts weights using **Error Correction**
- LEDs visualize changes in Weights and Bias

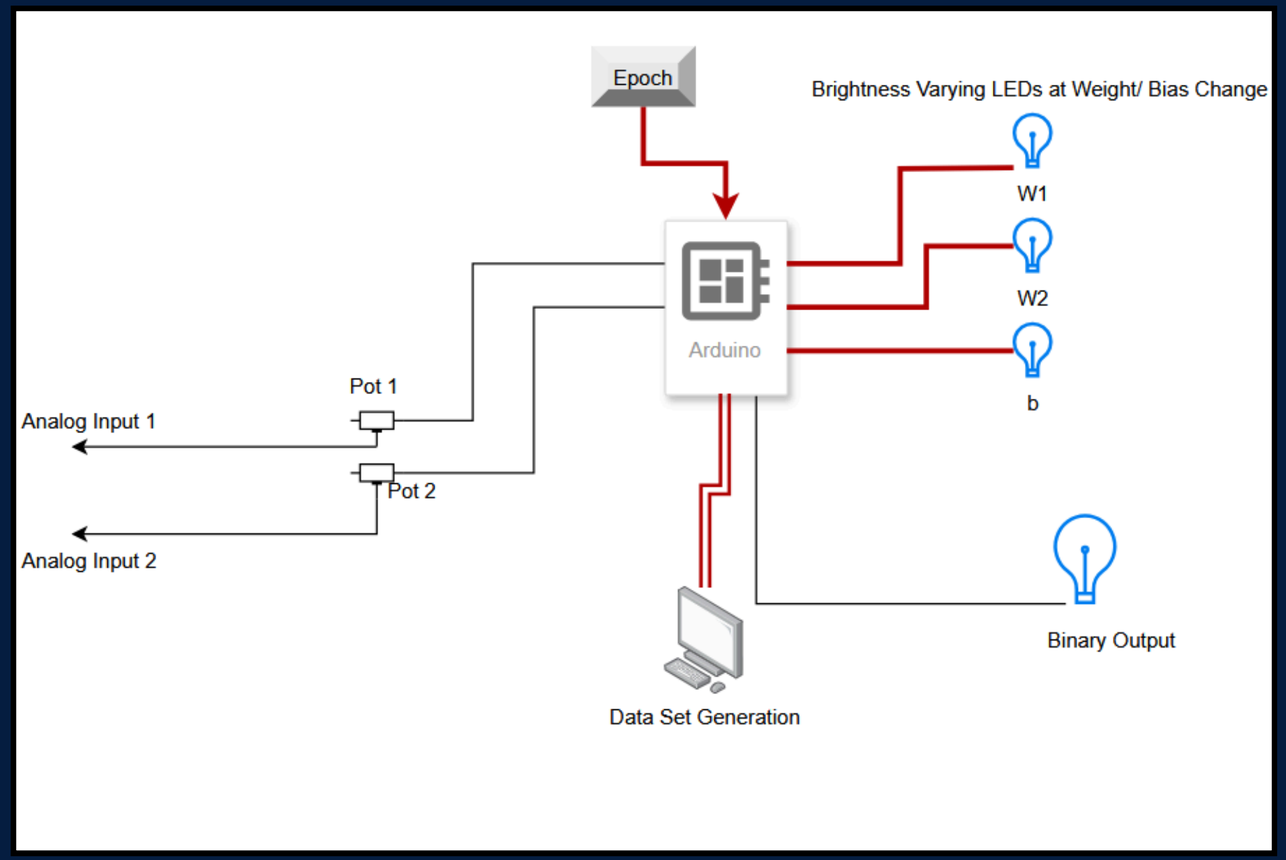
WHY FUZZY-INSPIRED?

Similar to Fuzzy Membership Functions

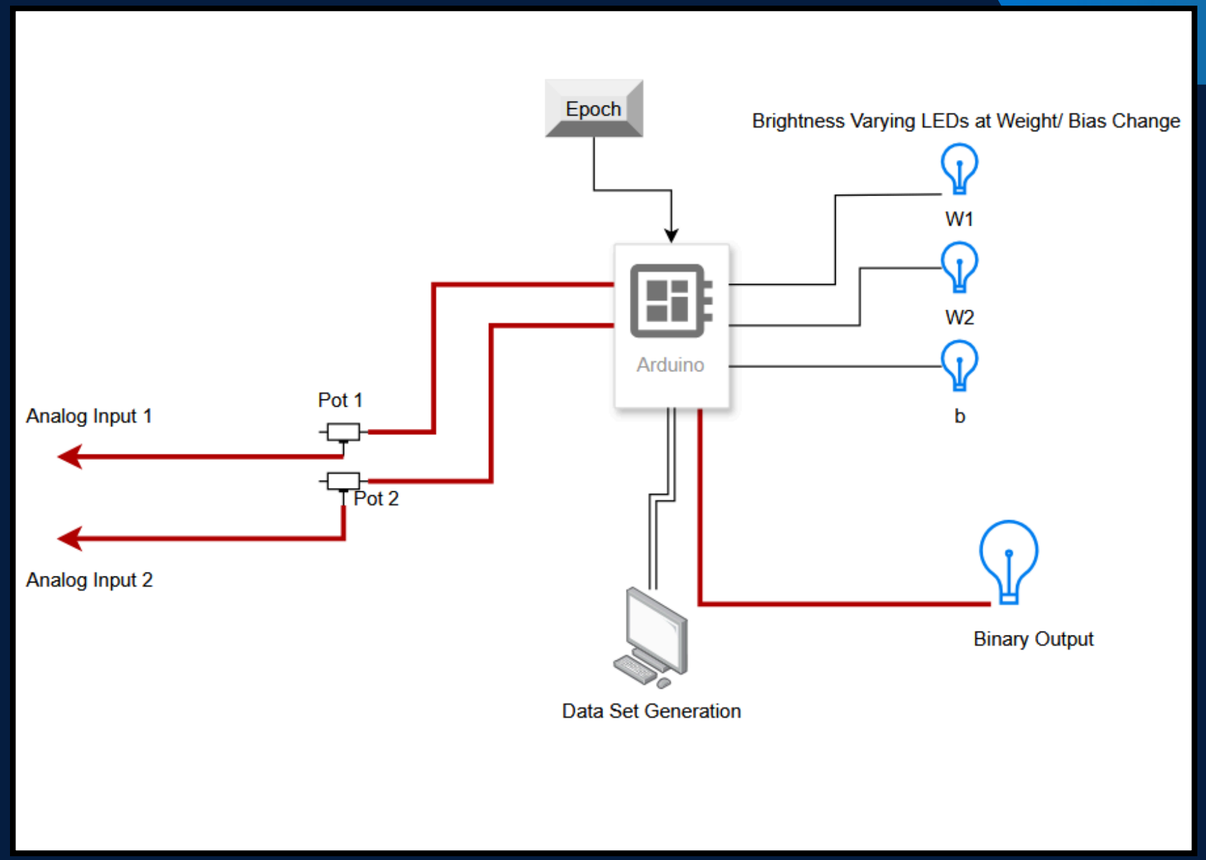
Continuous Input Range

Inputs mimic Fuzzy; output does not.

FUZZY-INSPIRED TRAINING: LEARNING OUTPUT BASED ON POTENTIOMETER RANGES

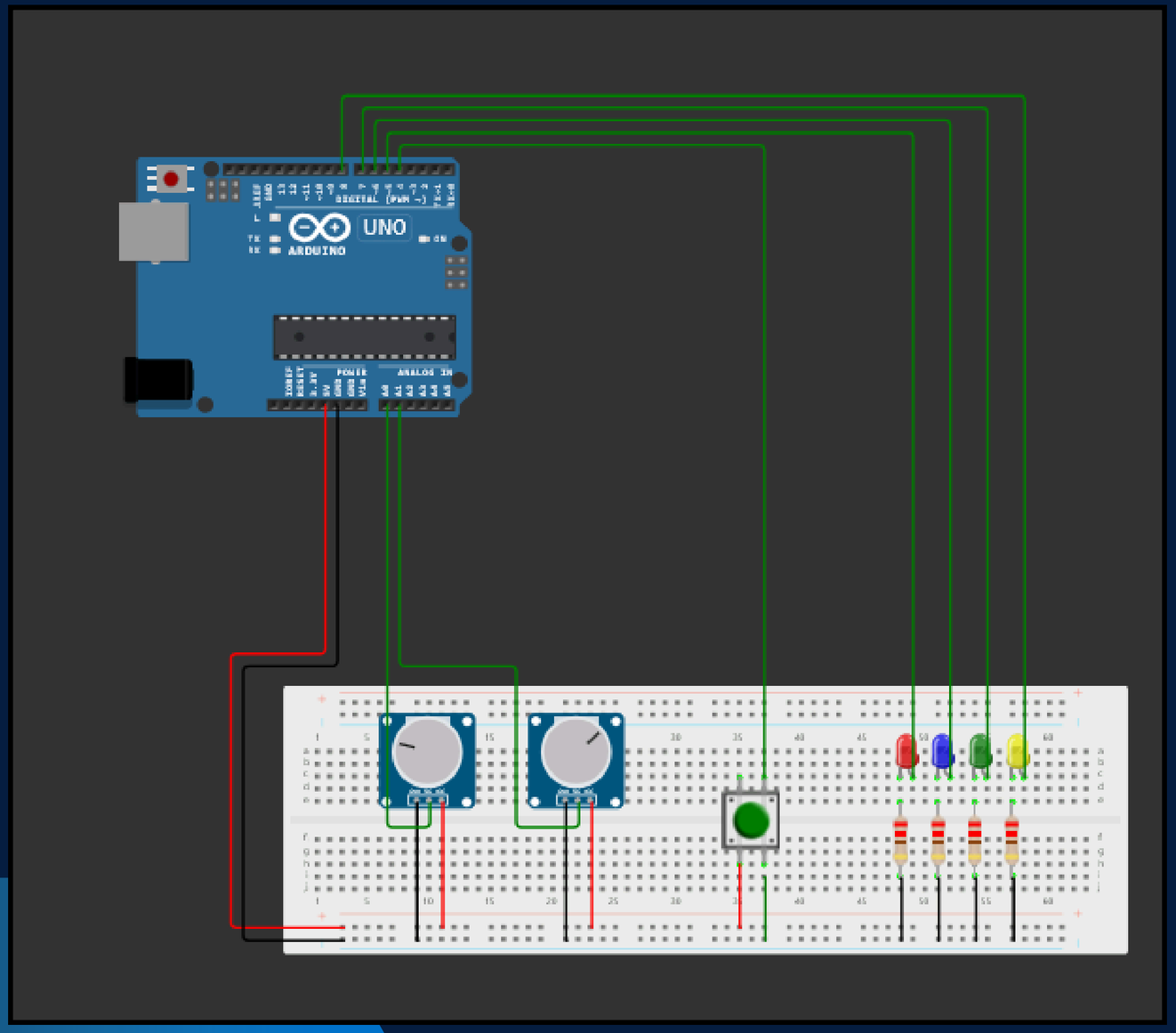


Training State



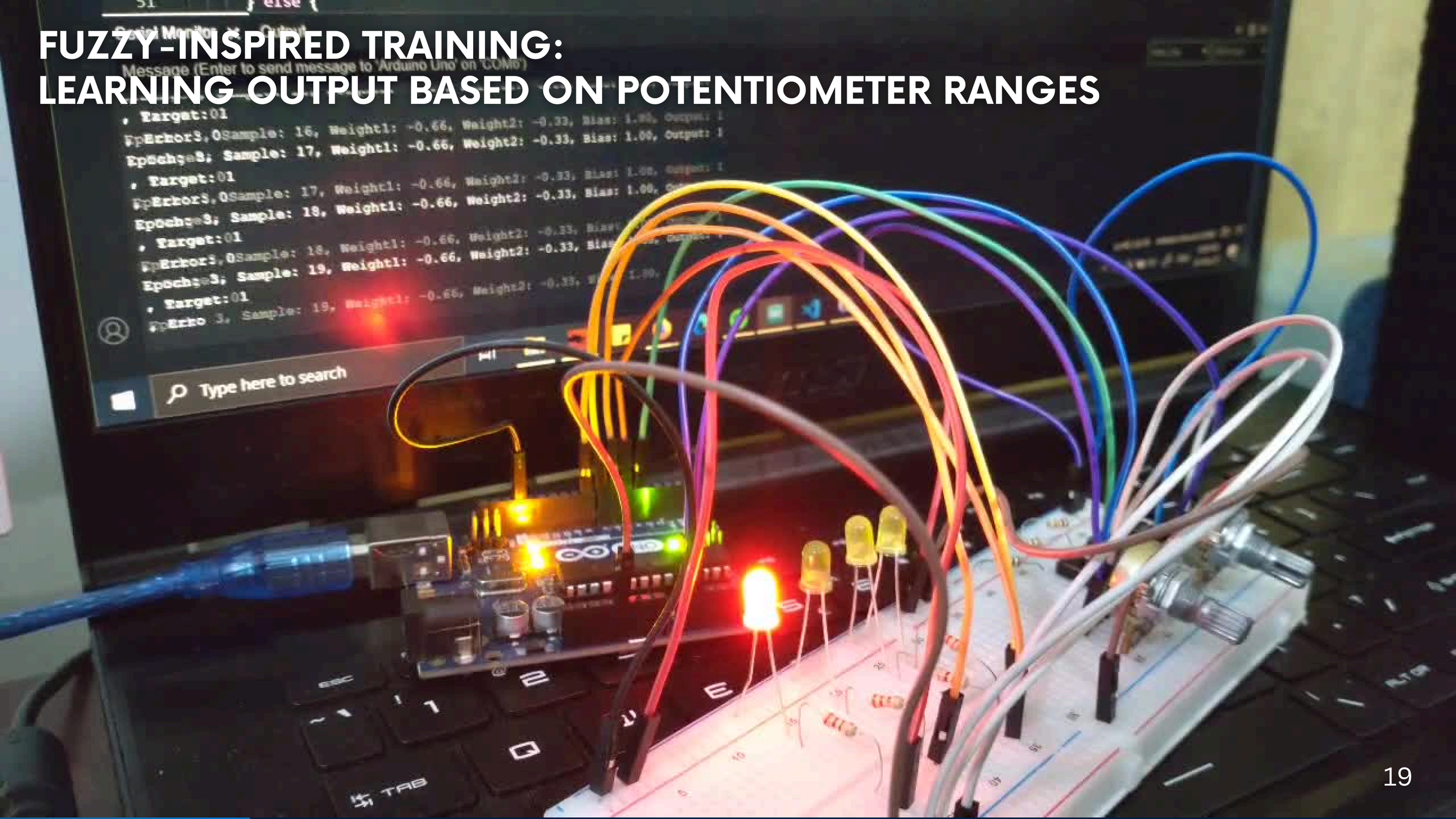
Predicting State

FUZZY-INSPIRED TRAINING: LEARNING OUTPUT BASED ON POTENTIOMETER RANGES



- Button-Controlled Training Epochs
- Learning Rate-based Weight / Bias Adjustments
- Maps weight changes to LED Brightness
- Randomly Generated Dataset for Learning

FUZZY-INSPIRED TRAINING:
LEARNING OUTPUT BASED ON POTENTIOMETER RANGES



CHALLENGES

- Arduino UNO : Hardware Limitations
- Negative Weight Representation
- Memory Limitations
- ARDUINO Usage as a State Machine
- Difficulties in mapping the brightness of LEDs to weights and bias
- Currently limited only for Binary Classification
- Integrating MLOps

FUTURE IMPROVEMENTS

- Using FPGA : To model perceptron with State Machine Approach
- Using LED Bar or RGB to represent negative weights and biases
- Using an LCD Display or 7 Segment Displays to show how the weights and biases are changing
- Using an LCD screen to show internal calculations
  - Error Calculation
  - Respective weights / bias calculations
- Use of MLOps in future iterations

INDIVIDUAL CONTRIBUTIONS

- Janith Wanasinghe (**E/20/420**)
  - Leading the project idea forward
  - Scrum Board Maintenance
  - High Level Architecture Design
- Ravindu Suraweeraarachchi (**E/19/393**)
  - Coding and Hardware Implementation of the two Perceptron Models
  - Simulation using Wokwi
- Ruwindya Jayasekara (**E/19/159**)
  - Simulation of Perceptron Models
- Layani Senevirathne (**E/19/365**)
  - Literature review on perceptron-based learning models & algorithms
- Pinindu Thiwanka **(E/19/407)**
  - Documentation of Goal, Problem Definition, Scope, and Justification sections

CONCLUSION

- NeuronGlow demonstrates perceptron learning in an interactive, hardware-based setup
- Bridges the gap between theoretical ML and real-world implementation
- Potential for expansion into deeper neural networks and automation
- Encourages hands-on learning for students and AI enthusiasts

REFERENCES

- F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," Psychological Review, vol. 65, no. 6, pp. 386–408, 1958.
- M. Minsky and S. Papert, Perceptrons: An Introduction to Computational Geometry. Cambridge, MA: MIT Press, 1969.
- M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in Proc. 12th USENIX Symp. Oper. Syst. Des. Implement.*, 2016, pp. 265–283.
- A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in Adv. Neural Inf. Process. Syst., vol. 32, 2019.
- A. Suresh et al., "Hardware-based neural network simulation using Arduino," Int. J. Artif. Intell. Appl., vol. 11, no. 4, pp. 32–45, 2020.
- W. Jiang et al., "An interactive perceptron learning tool with Arduino," Educ. Robot. J., vol. 9, no. 3, pp. 18–30, 2021.
- A. Gupta and S. Verma, "Analog implementations of perceptron-based classification," J. Electron. Learn. Technol., vol. 6, no. 1, pp. 55–68, 2019.
- S. Rothhaupt, W. Meyerle, and B. Kormann, "FPGA-based hardware acceleration of artificial neural network inference," in Proc. 6th AccML Workshop, 2024, pp. 1–6.

THANK YOU!

Presented to you by:
FUZZBUSTERS