The Perceptron

LIN 313 Language and Computers
UT Austin Fall 2025
Instructor: Gabriella Chronis

Admin

- HW 2 grades posted this afternoon
- HW 3 due 10/15

Overview 10/8

- Review homework
- Artificial Neurons
  - components
    - weights
    - bias
    - activation
- The Perceptron Algorithm
  - vector notation
  - dot product
  - use the algorithm to predict outputs from inputs

Problem 2: Bayesian Spelling Correction

Part 2: the bigram model

Error model probabilities

- P(korekt | correct) = .034
- P(korekt | kraken) = .007
- P(korekt | carrot) = .015

Language model probabilities

- P(correct | great) = 0/1013 = 0
- P(kraken | great) = 5/1013 = 0.004935
- P(carrot | great) = 1/1013 = 0.000987

Problem 2: Bayesian Spelling Correction

Part 2: the bigram model

Error model probabilities

- P(korekt | correct) = .034
- P(korekt | kraken) = .007
- P(korekt | carrot) = .015

Language model probabilities

- P(correct | great) = 0/1013 = 0
- P(kraken | great) = 5/1013 = 0.004935
- P(carrot | great) = 1/1013 = 0.000987

Derive the formula we are going to use:

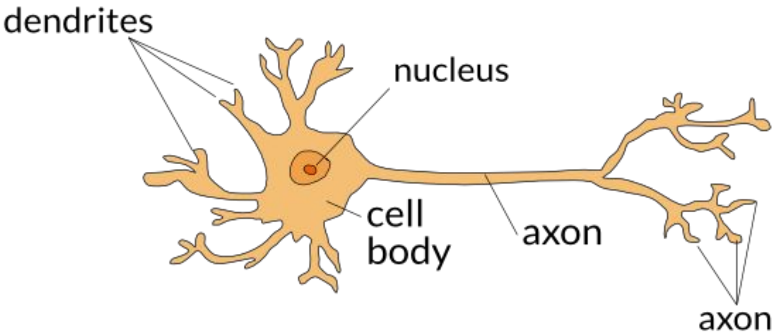
1. conditional independence assumption
   a. P(a | b, c) = P (a | b) x P(a | c)
2. Substitute our events
   a. P(candidate | korekt , great) = P(candidate | great) x P(candidate | korekt)
3. We already have P(candidate | great).
4. We need P(candidate | korekt). Use Bayes Law
   a. P(candidate | korekt, great) = P(candidate | great) x P(korekt | candidate) x P(candidate) / P(korekt)
5. We will be comparing these probabilities, so we can ignore the denominator because it's the same for all
   a. P(candidate | korekt, great) = P(candidate | great) x P(korekt | candidate) x P(candidate)

Apply the formula to each candidate:

1. correct
   a. P(correct | korekt, great) ∝ P(correct | great) x P(korekt | correct) x P(correct)
   b. P(correct | korect, great) ∝ 0 x 0.034 x 0.000845
   c. P(correct | korect, great) ∝ 0
2. kraken
   a. P(kraken | korekt, great) ∝ P(kraken | great) x P(korekt | kraken) x P(kraken)
   b. P(kraken | korekt, great) ∝ 0.004935 x 0.007 x 0.000264
   c. P(kraken | korekt, great) ∝ 9.11988e-9
3. carrot
   a. P(carrot | korekt, great) ∝ P(carrot | great) x P(korekt | carrot) x P(carrot)
   b. P(carrot | korekt, great) ∝ 0.000987 x 0.015 x 0.000211
   c. P(carrot | korekt, great) ∝ 3.123855e-9

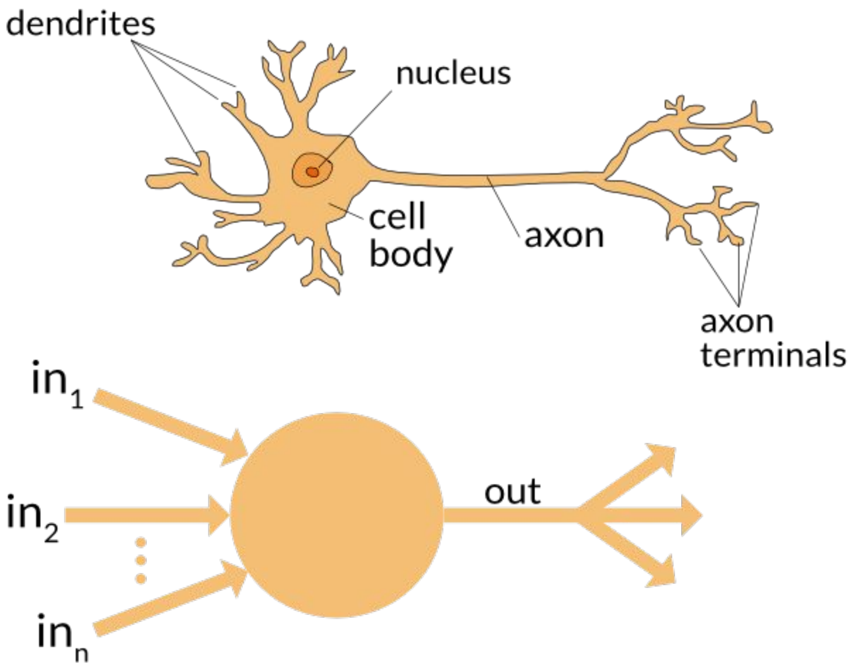
What is a neural network?

What is a neuron?



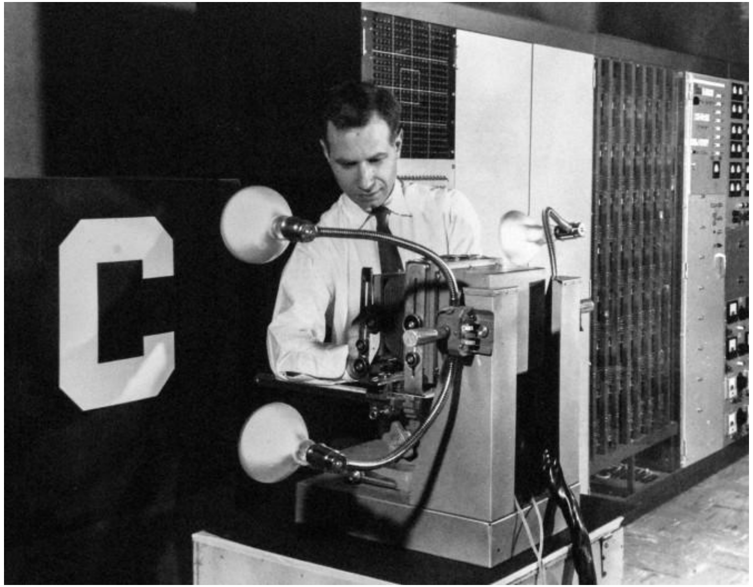
- a very long cell
- **dendrites** collect stimuli from other cells, chemical signals
- if the **action potential** is reached, the neuron fires

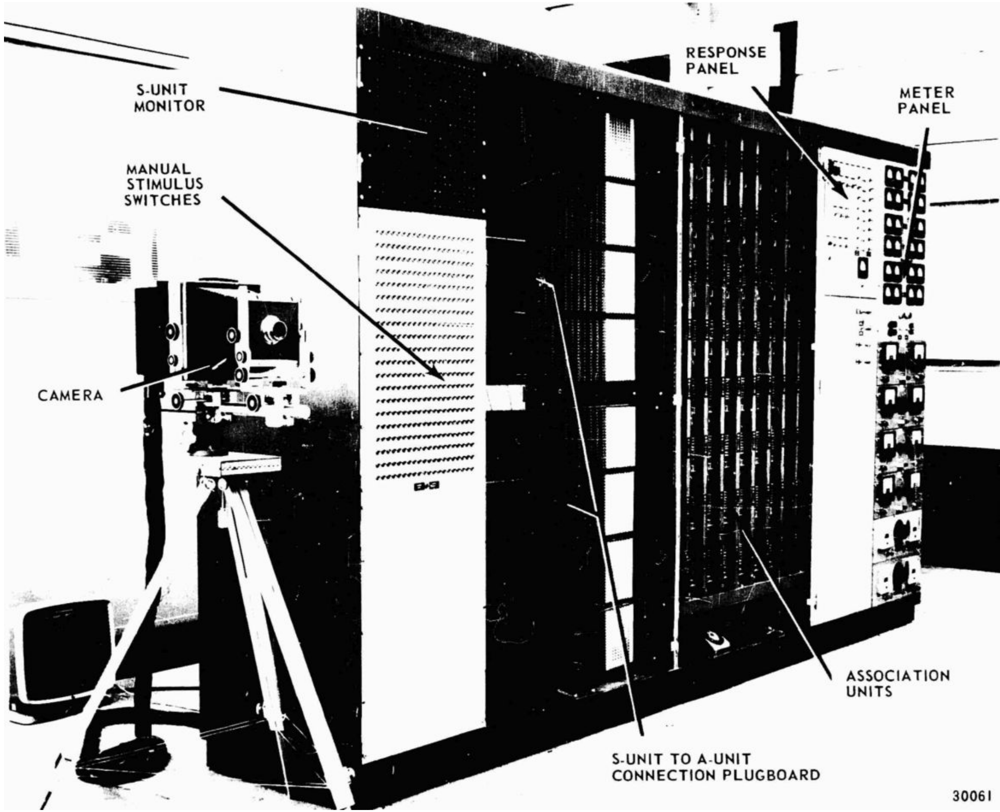
The Artificial Neuron (McCulloch-Pitts, 1943)



- binary inputs (either on or off)
- summation function
  - adds up the input values
- some *activation function*
  - decides whether to fire

The Perceptron (Rosenblatt, 1958)

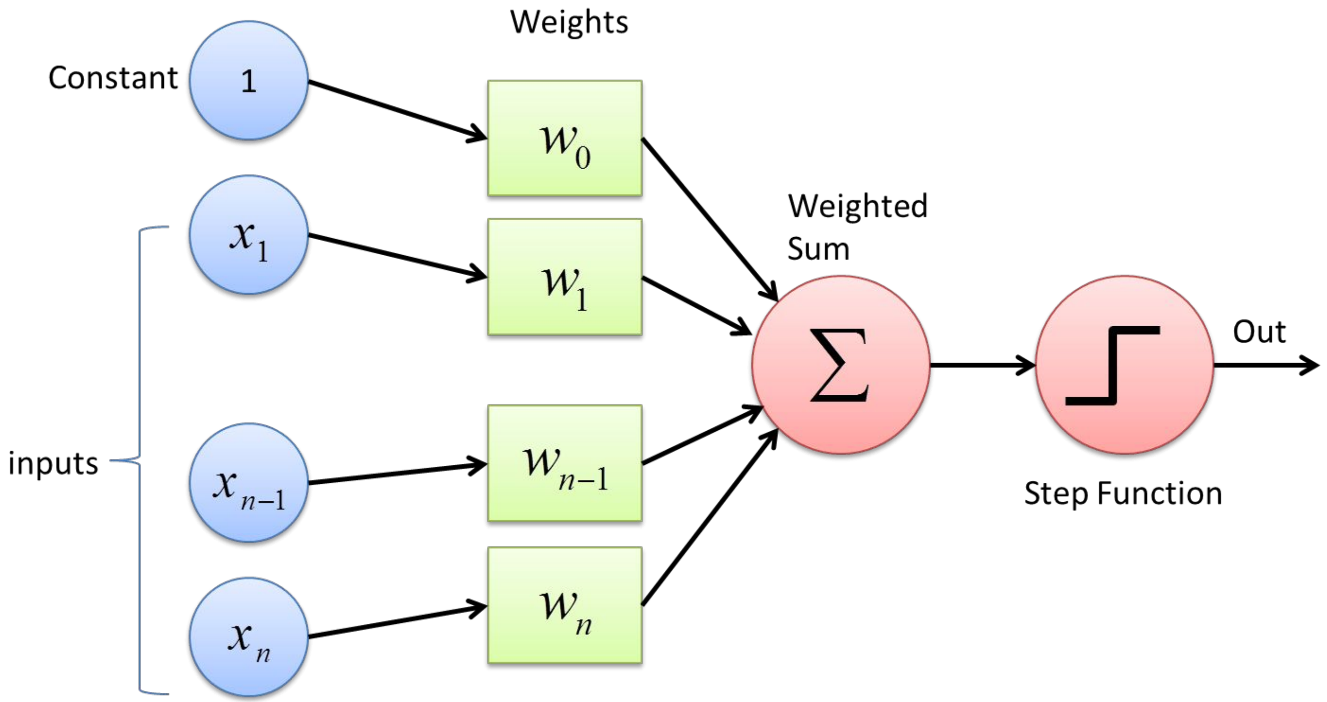




Perceptron Intuition

Should you buy a ticket to the next UT Game?

The Perceptron Algorithm



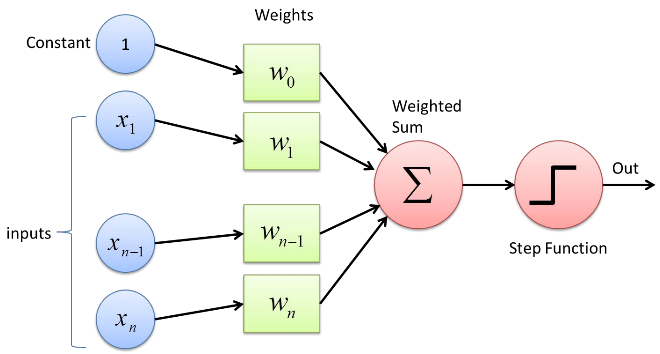
The Perceptron Algorithm

formalized as an equation, the Perceptron looks like this

$$\hat{y} = sign(z) = \begin{cases} 1, & \text{if } z > 0 \\ 0, & \text{if } z \leq 0 \end{cases}$$

$$z = \left(\sum_{i=1}^{n} w_i x_i \right) + b$$

$$z = (x_1 * w_1) + (x_2 * w_2) + (x_3 * w_3) + \ldots + (x_n * w_n) + b$$



Vectors

The equation gets simpler if we think of the input as vectors.

A vector is an ordered list of numbers.

An ordered pair is a vector.

An ordered pair has a **geometric interpretation**.

A vector is an ordered pair with more numbers

A vector also has a geometric interpretation

The Perceptron Algorithm (again)

$$\boxed{\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)}$$

w is the weight vector ($w_1 w_2 \ldots w_i$)

x is the input vector ($x_1 x_2 \ldots x_i$)

b is the bias, still a regular number

sign( ) is the activation function

w · x is the dot product of **w** and **x**

the dot product is just shorthand for the weighted sum!

The Dot Product

Definition

$$\mathbf{a} = \langle a_1, a_2, a_3 \rangle \quad \mathbf{b} = \langle b_1, b_2, b_3 \rangle$$

$$\boxed{\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + a_3 b_3}$$

Vector Dot Products

(

Putting it all together

What does a **forward pass** through the Perceptron look like for our UT example?

$$\boxed{\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)}$$

It's an away game. two of my friends are actually going, but I'm broke and I have two write an essay and finish a lab write up this weekend.

w =

x =

b =

Putting it all together

What does a **forward pass** through the Perceptron look like for our UT example?

$$\boxed{\hat{y} = \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} + b)}$$

It's a home game, the weather is balmy. I have a ton of homework but all of my friends are going.

w =

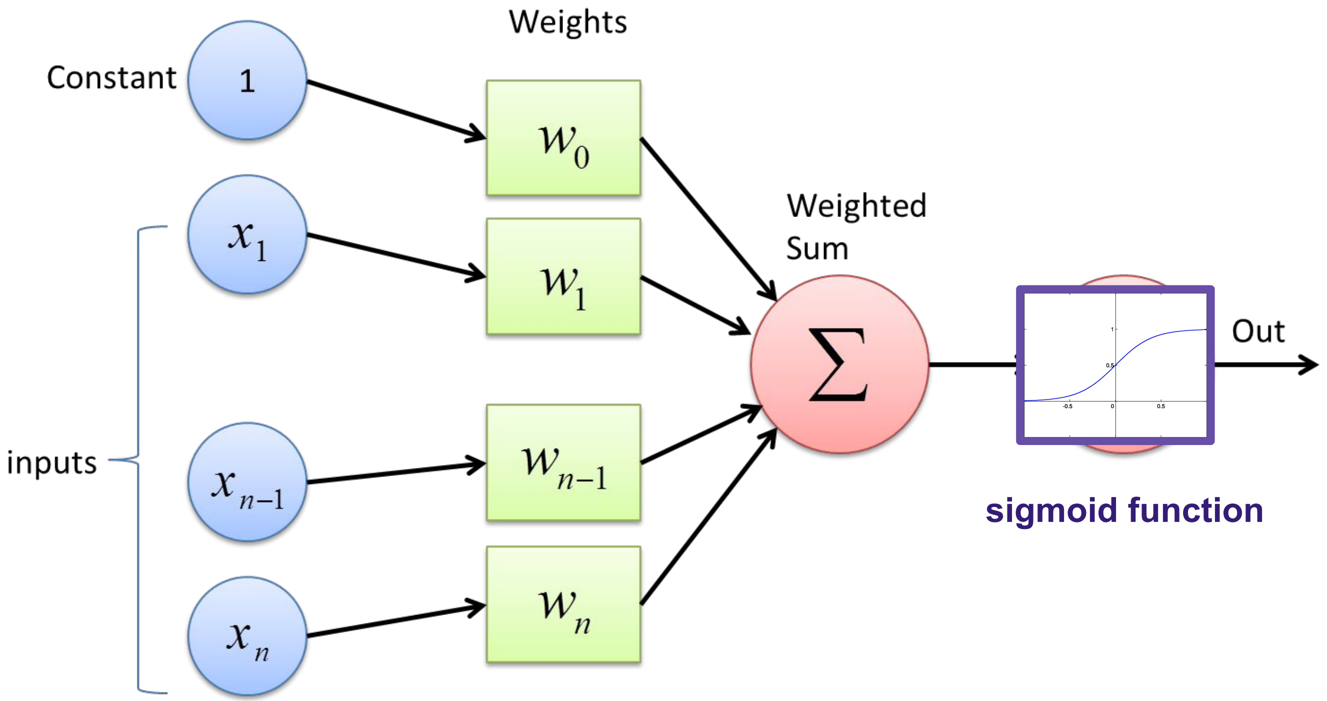
x =

b =

Linear Regression

The only difference is the activation function that we choose!

Linear regression uses the **sigmoid** activation function



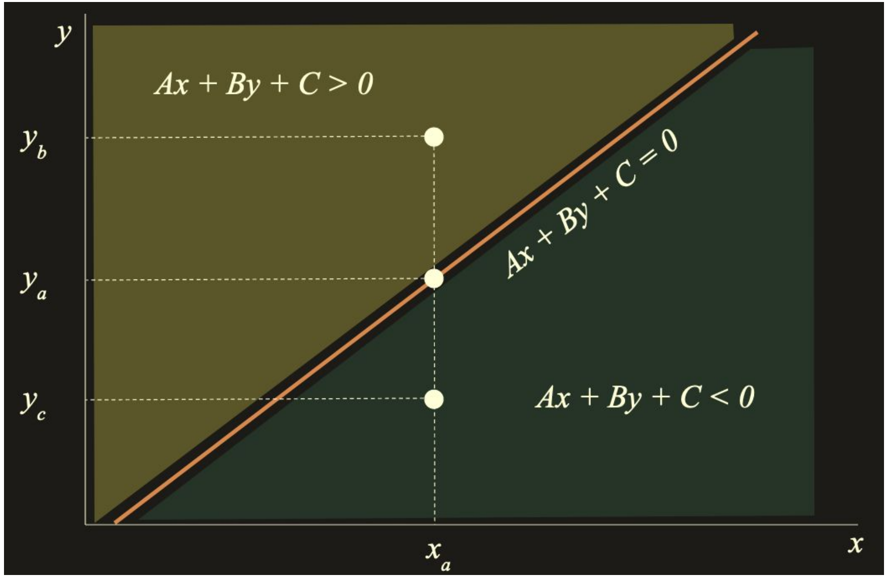
Geometric Interpretation

we can make decision function (the weighted sum) into an equation for a line:

$$w_1x_1 + w_2x_2 + b = 0$$

Rewrite that line in standard y=mx+b form

$$x_2 = -(w_1/w_2)x_1 - (b/w_2)$$



Note: This is assuming that the coefficients are **positive**. If not, the top region could perhaps be where $Ax + By + C < 0$ instead and the bottom be $Ax + By + C > 0$. You will see this as you interact with a model later in the post.

https://karthikvedula.com/2024/01/05/visualizing-the-perceptron-learning-algorithm/

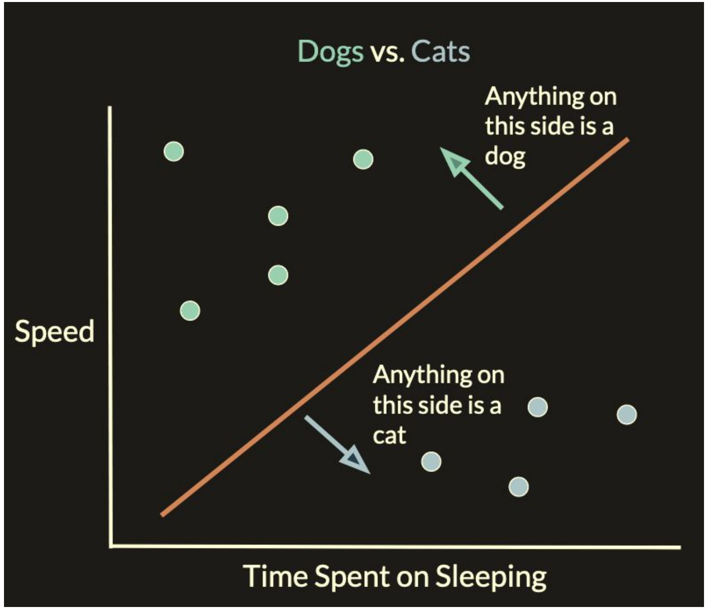
Geometric Interpretation

we can make decision function (the weighted sum) into an equation for a line:

$$w_1x_1 + w_2x_2 + b = 0$$

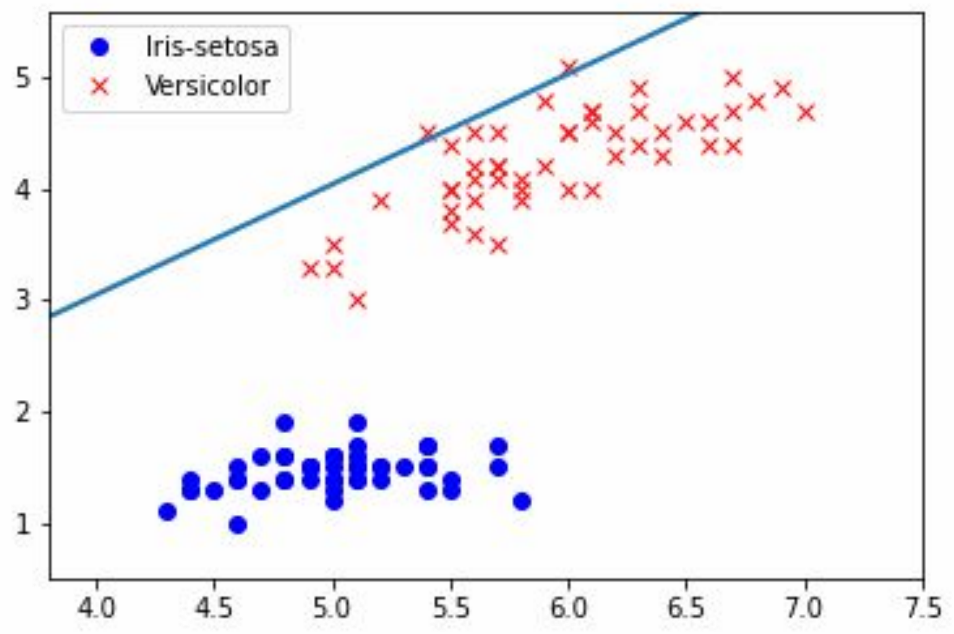
Rewrite that line in standard y=mx+b form

$$x_2 = -(w_1/w_2)x_1 - (b/w_2)$$



https://karthikvedula.com/2024/01/05/visualizing-the-perceptron-learning-algorithm/

Perceptron Learning (next class)



Interactive Demos

- perceptron: https://karthikvedula.com/2024/01/05/visualizing-the-perceptron-learning-algorithm/
- perceptron: https://perceptron.streamlit.app/
- dot product: https://maththebeautiful.com/dot-product/

If ever the twain shall meet…

Introducing the world's first biological neural network: https://corticallabs.com/cl1