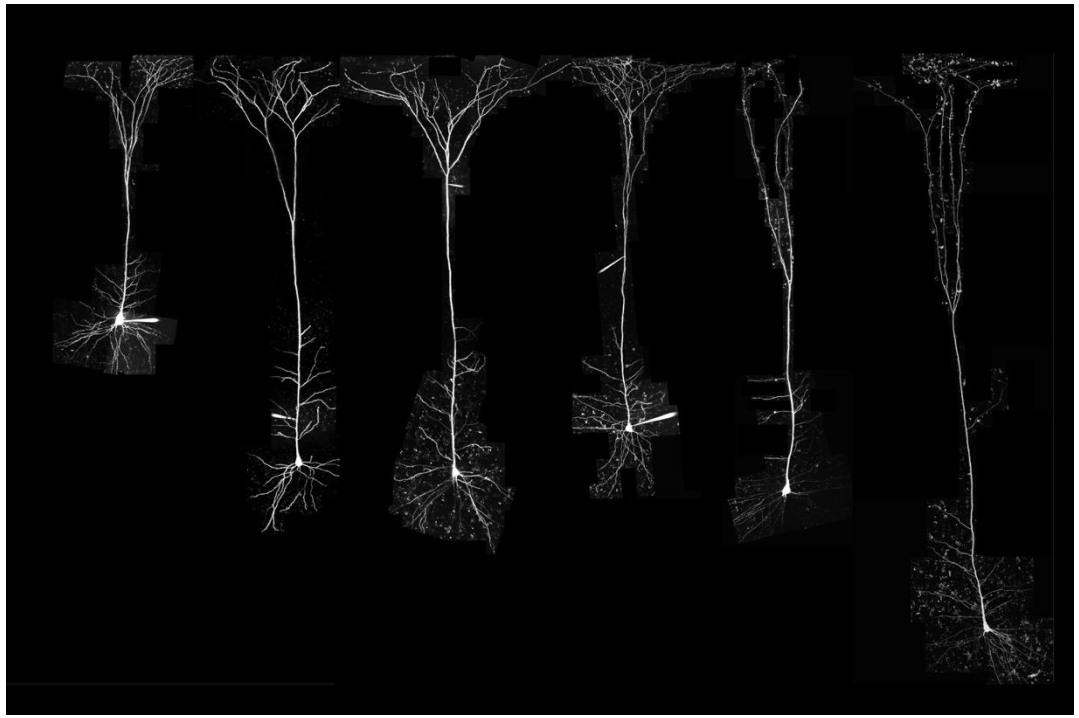
Universitat de Girona

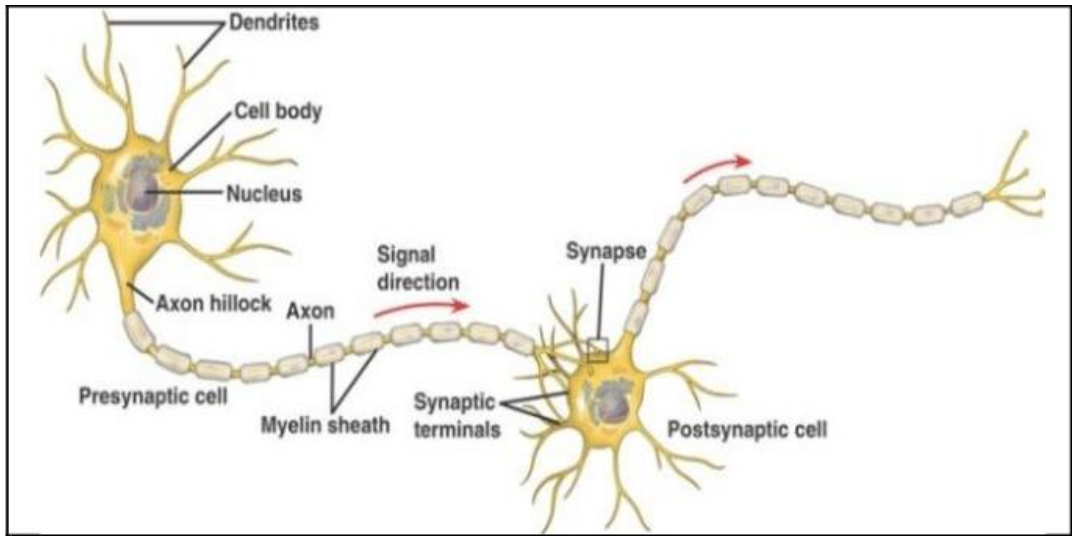
Neural Networks

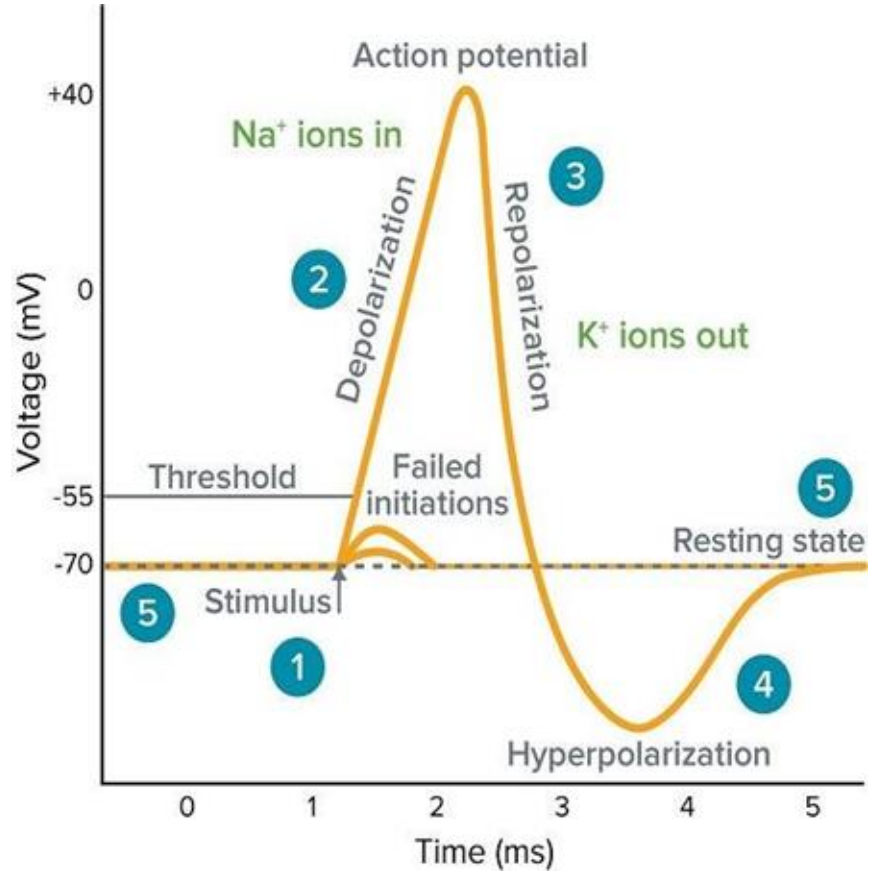
Guillem Guigó i Corominas

guillem.guigo@uvic.cat

UNIVERSITAT DE VIC
UNIVERSITAT CENTRAL
DE CATALUNYA

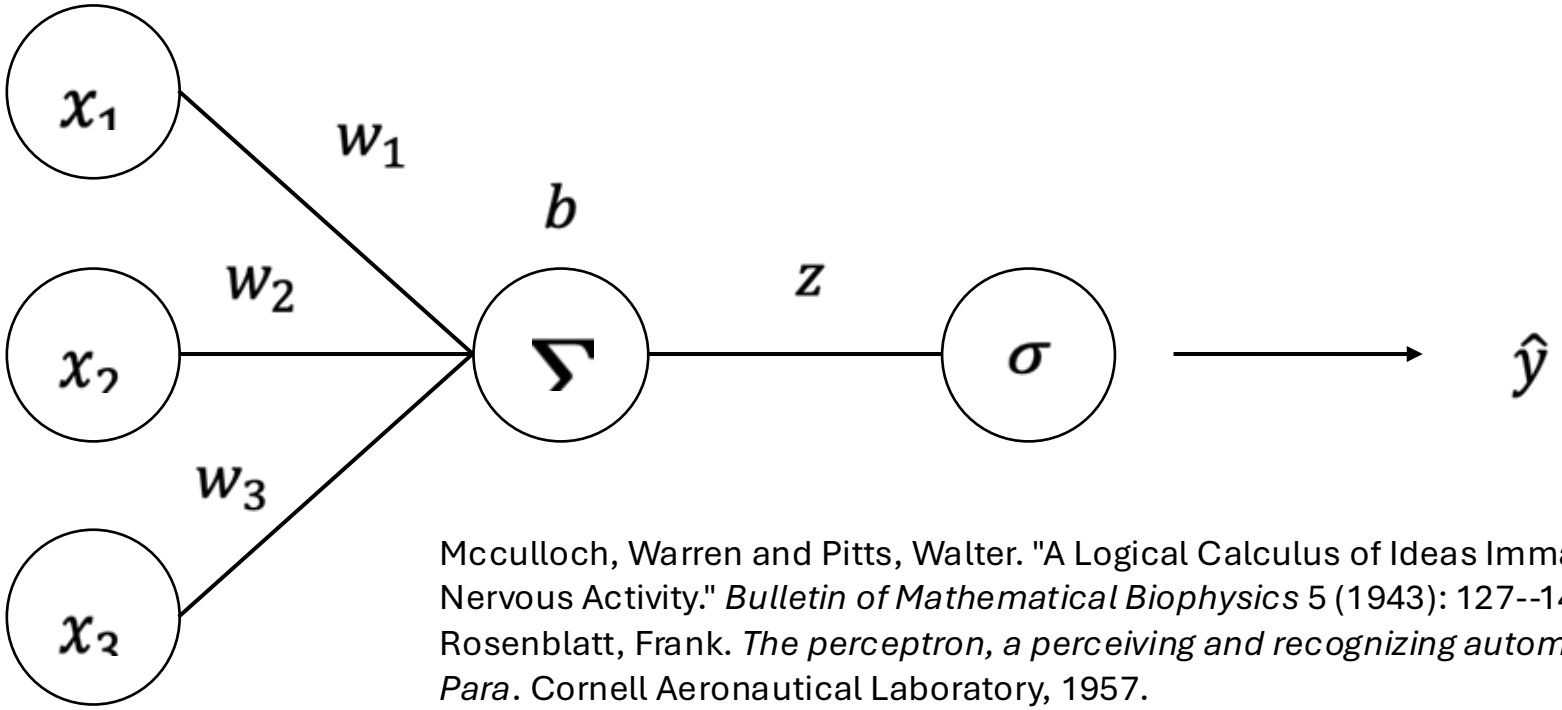



Dendrites
Cell body
Nucleus
Axon hillock
Axon
Signal direction
Synapse
Presynaptic cell
Myelin sheath
Synaptic terminals
Postsynaptic cell


Action potential
+40
Na⁺ ions in
Depolarization
Repolarization
3
2
0
K⁺ ions out
Voltage (mV)
Threshold
Failed initiations
5
-55
Resting state
-70
5
Stimulus
1
4
Hyperpolarization
0
1
2
3
4
5
Time (ms)

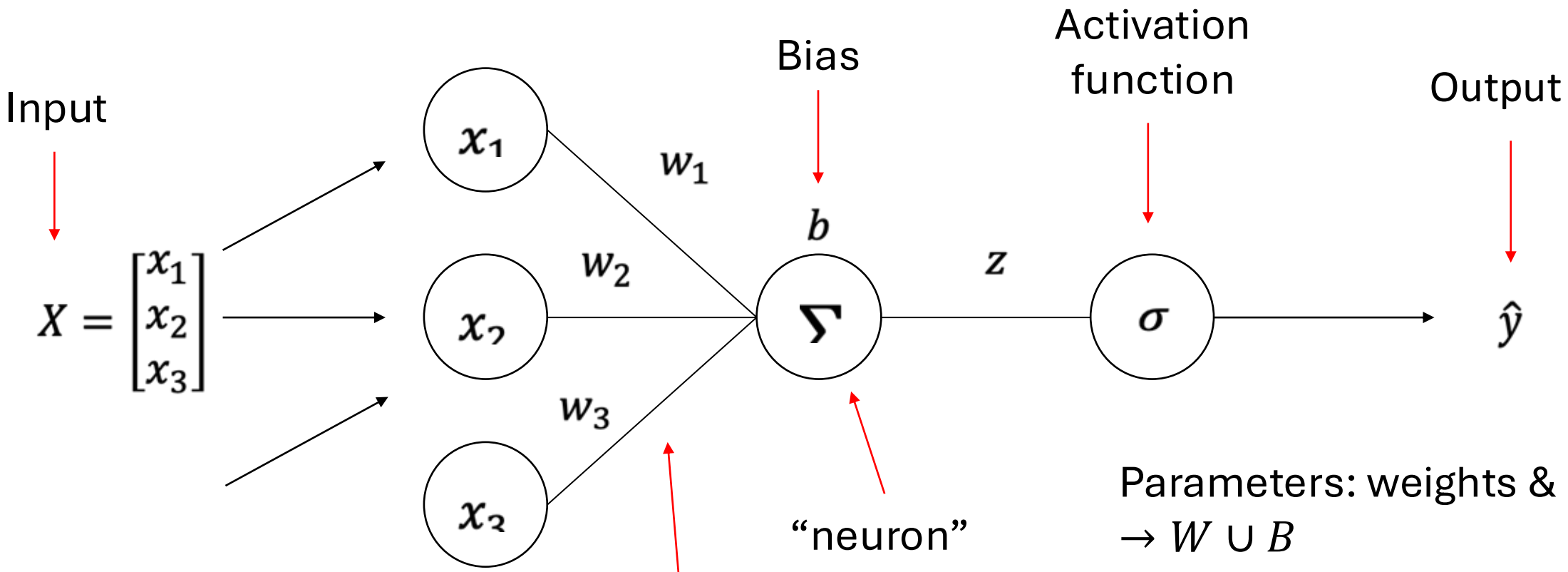
Single neuron model: The Perceptron

- McCulloch and Pitts (1943) – Model description
- Frank Rosenblatt (1957) – Machine implementation



Mcculloch, Warren and Pitts, Walter. "A Logical Calculus of Ideas Immanent in Nervous Activity." *Bulletin of Mathematical Biophysics* 5 (1943): 127--147.
Rosenblatt, Frank. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957.

Single neuron model: The Perceptron

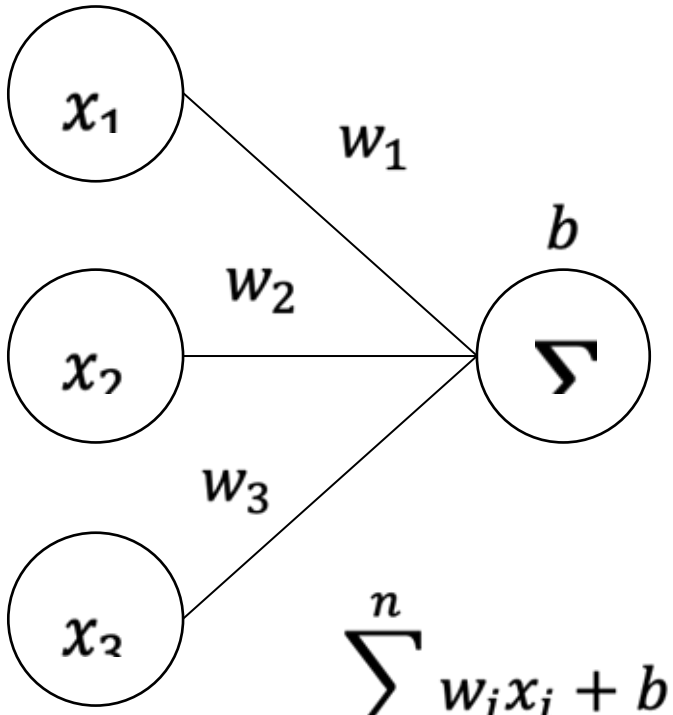


Parameters: weights & bias
$\rightarrow W \cup B$

$W = \{w_1, w_2, w_3\}$

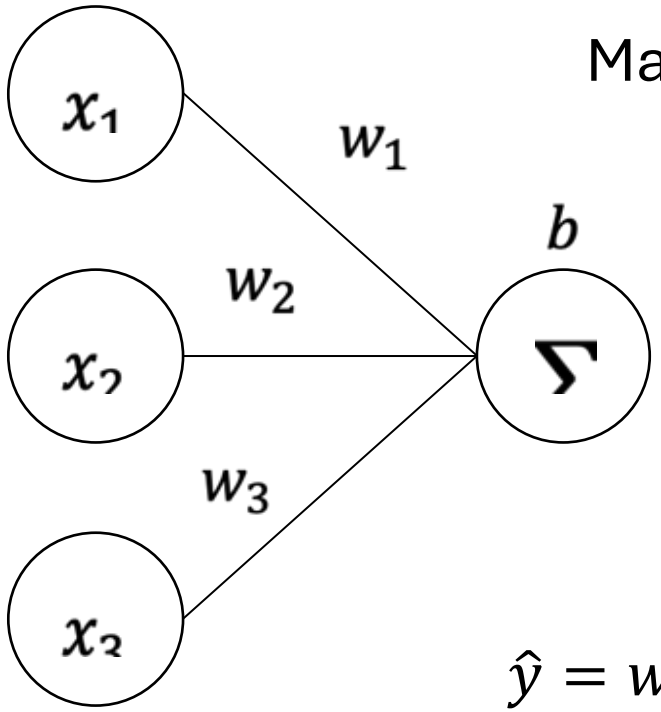
$B = \{b\}$

Single neuron model: The Perceptron



$$\sum_{i=1}^{n} w_i x_i + b = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

Single neuron model: The Perceptron



Linear combination of inputs scaled by the weights

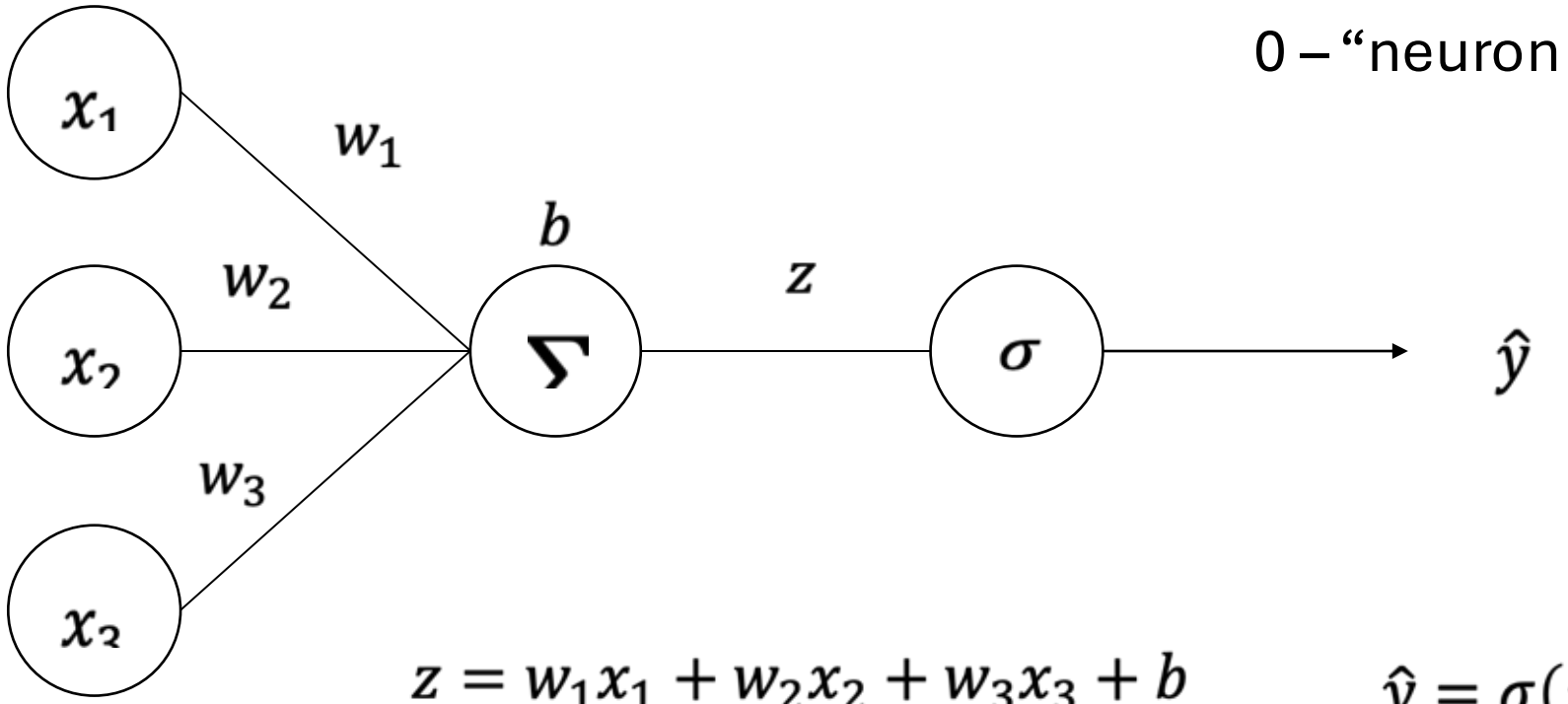
Mathematically equivalent to linear regression

How many trainable parameters does this perceptron have?

$$\hat{y} = w^T x + b \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \qquad b = b$$

Single neuron model: The Perceptron

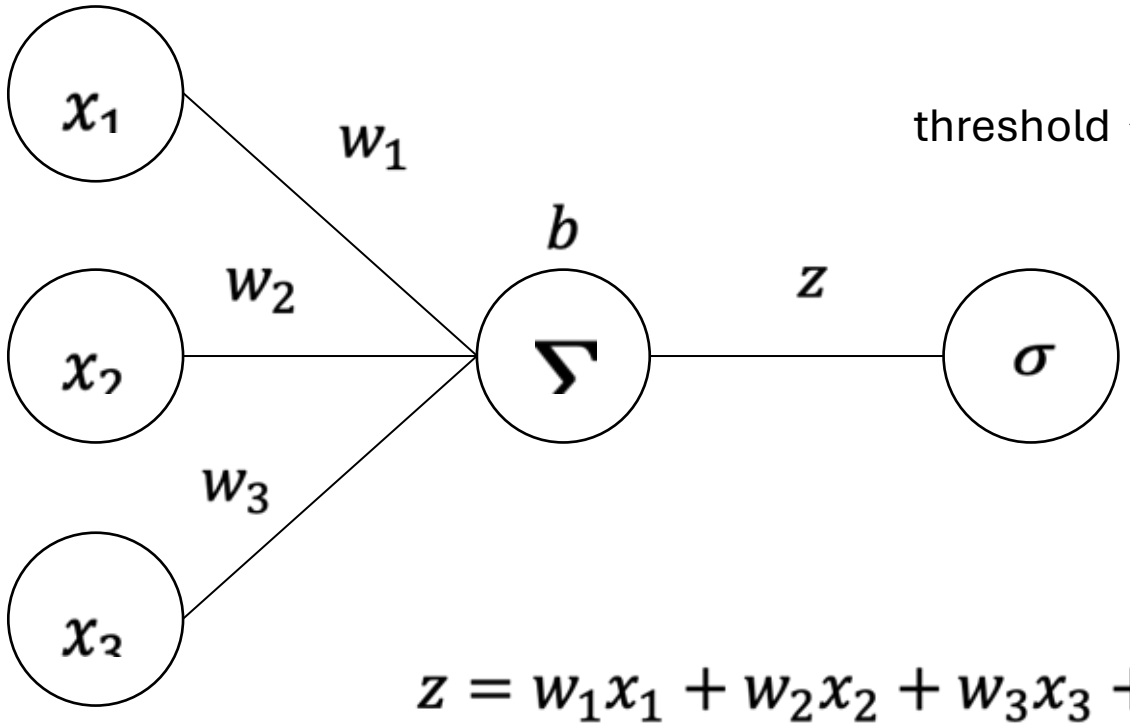
1 – “neuron activates”
0 – “neuron doesn’t activate”

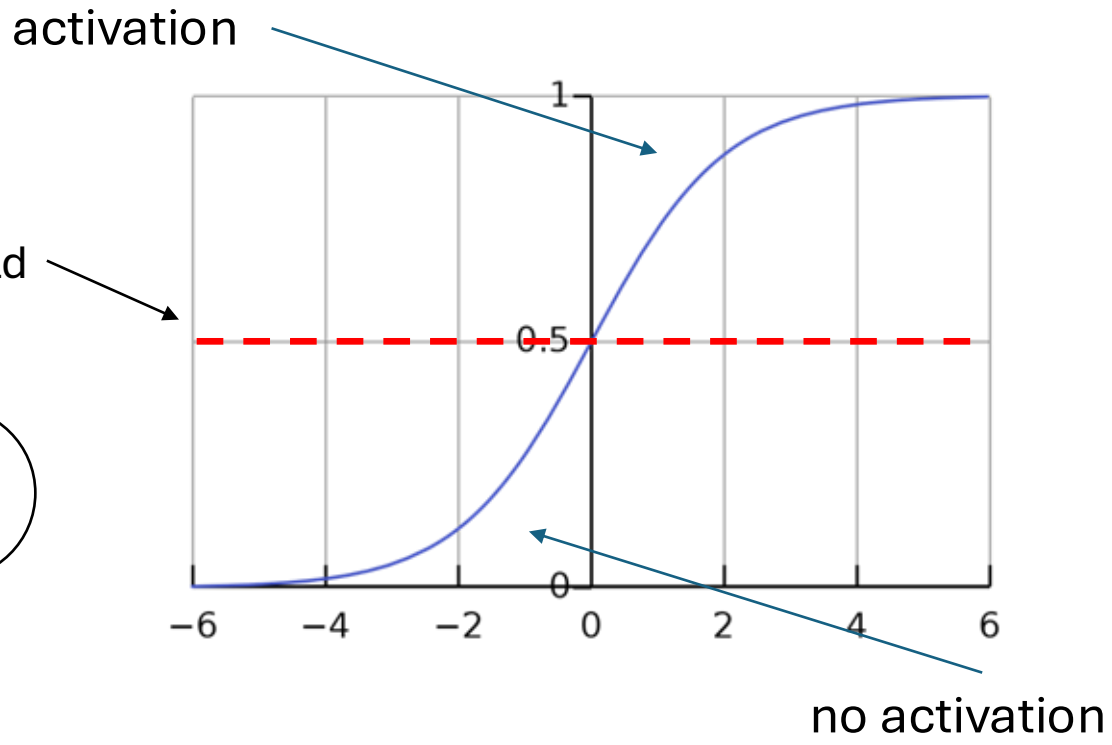


$$z = w_1x_1 + w_2x_2 + w_3x_3 + b$$

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$$

Single neuron model: The Perceptron





$$z = w_1x_1 + w_2x_2 + w_3x_3 + b$$

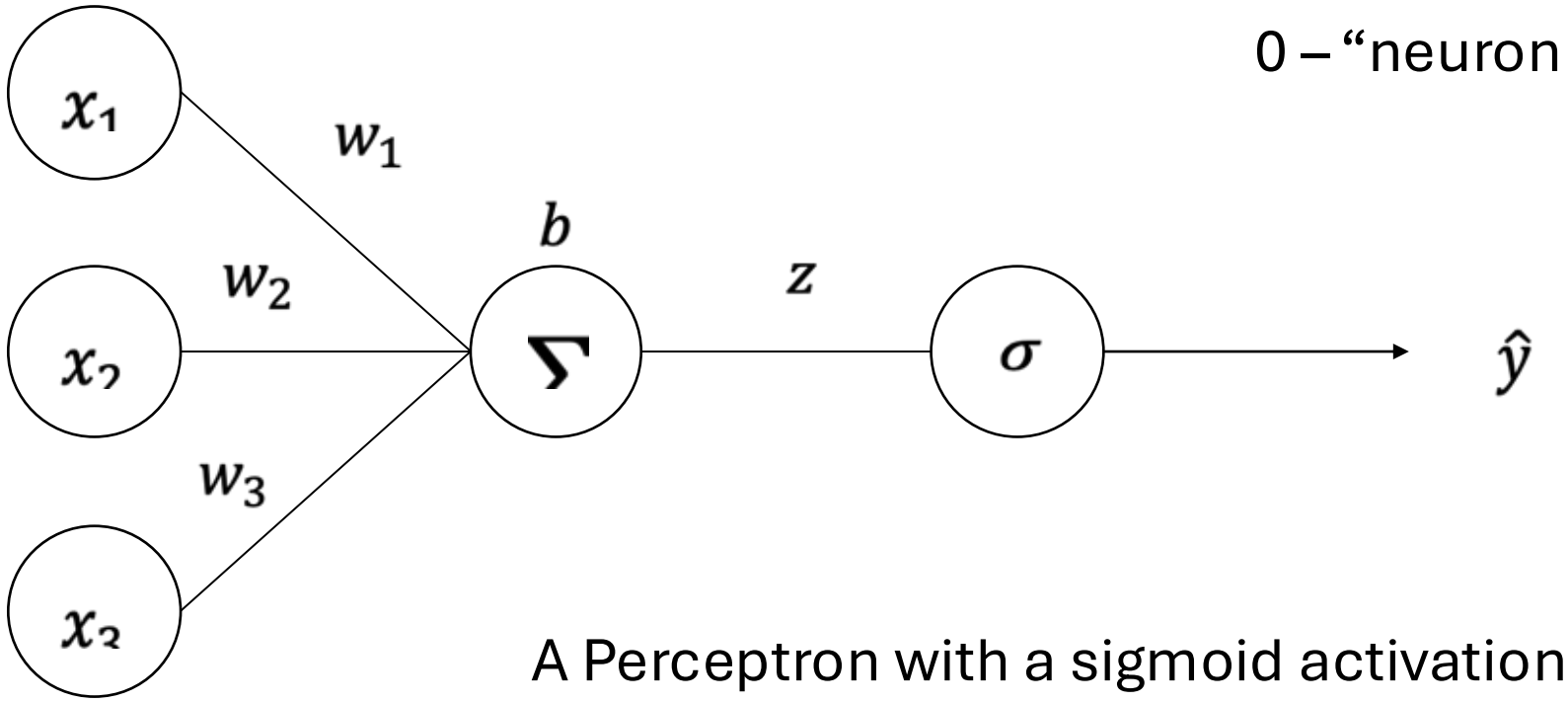
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

$$\hat{y} = z = \sigma(w_1x_1 + w_2x_2 + w_3x_3)$$

Sigmoid activation

Single neuron model: The Perceptron

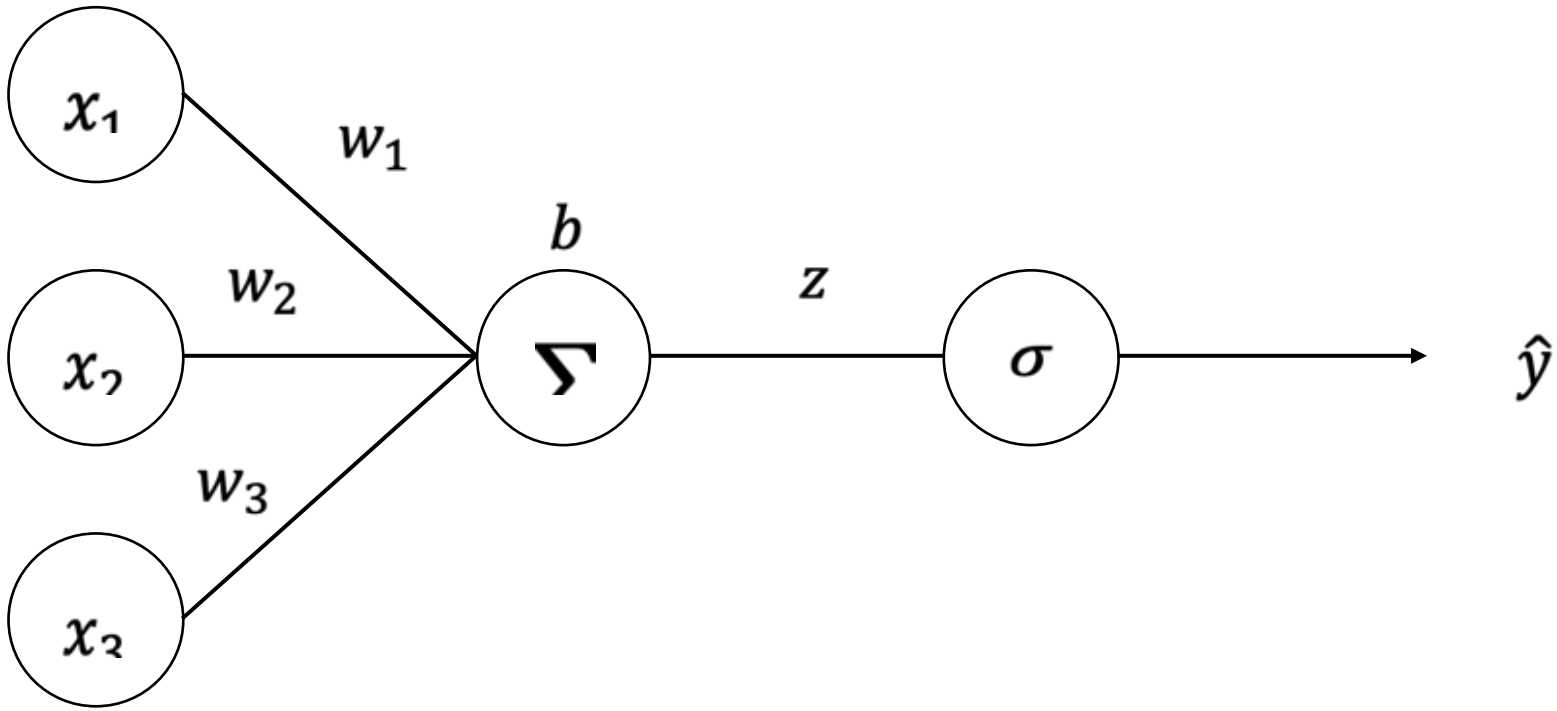
1 – “neuron activates”
0 – “neuron doesn’t activate”



A Perceptron with a sigmoid activation function is mathematically equivalent to logistic regression

Single neuron model: The Perceptron

Example: predicting whether someone has a sleep disorder given the quality of the sleep, the daily amount of exercise and the stress level

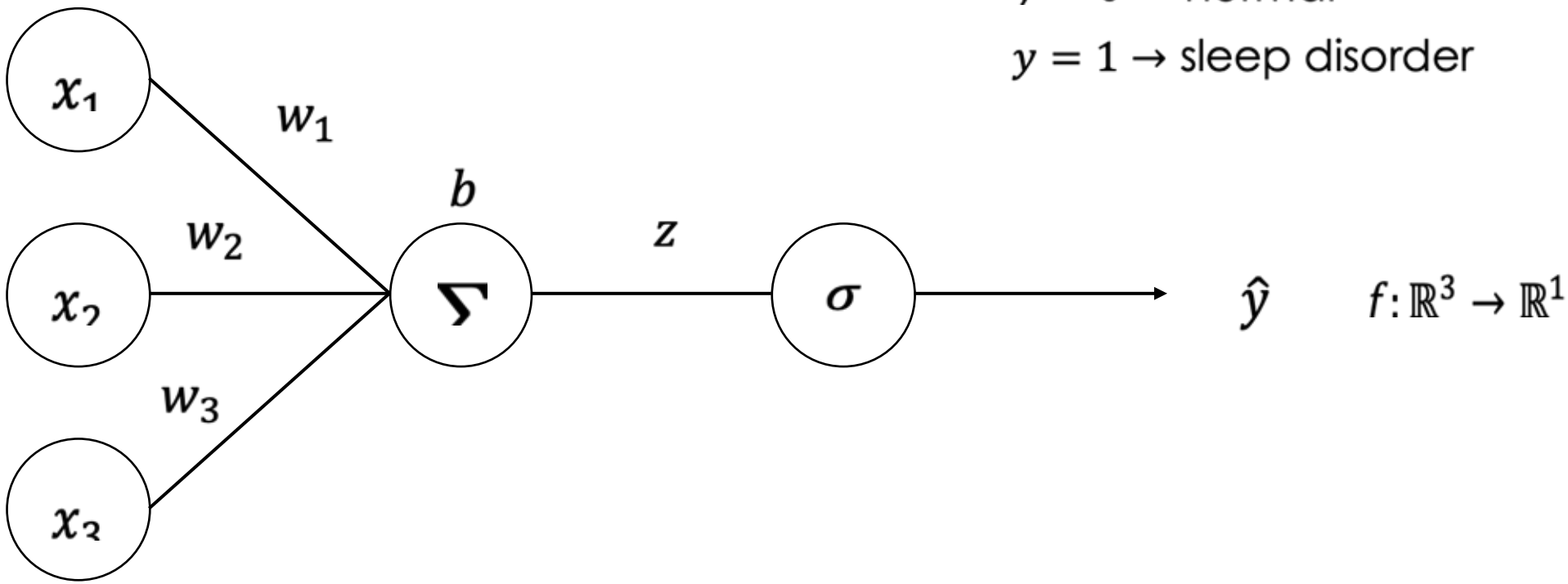


Single neuron model: The Perceptron

x_1 = sleep quality (1 - 10) x_2 = exercise (min/daily) x_3 = stress level (1 - 10)

$y = 0 \rightarrow$ normal

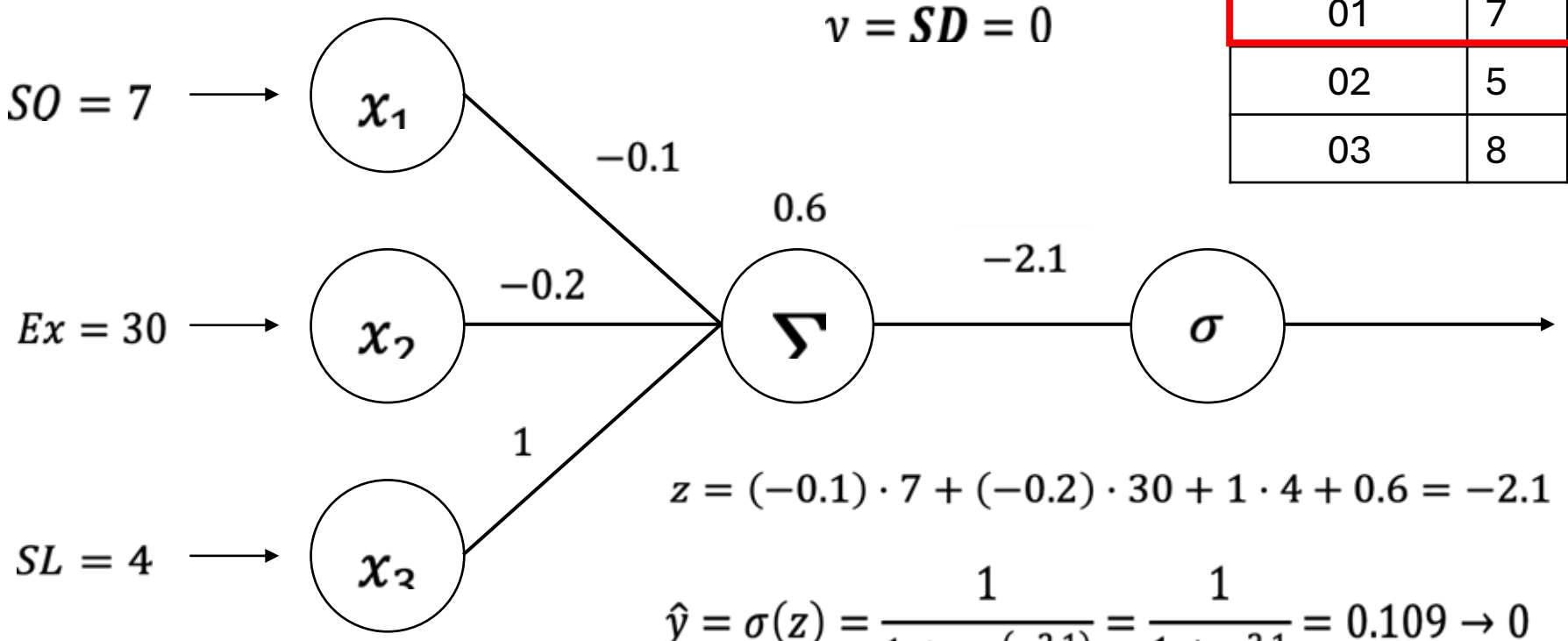
$y = 1 \rightarrow$ sleep disorder



Single neuron model: The Perceptron

x_1 = sleep quality (1 - 10) x_2 = exercise (min/daily) x_3 = stress level (1 - 10)

Patient ID	SQ	Ex	SL	SD
01	7	30	4	0
02	5	20	8	1
03	8	60	8	0



$$z = (-0.1) \cdot 7 + (-0.2) \cdot 30 + 1 \cdot 4 + 0.6 = -2.1$$

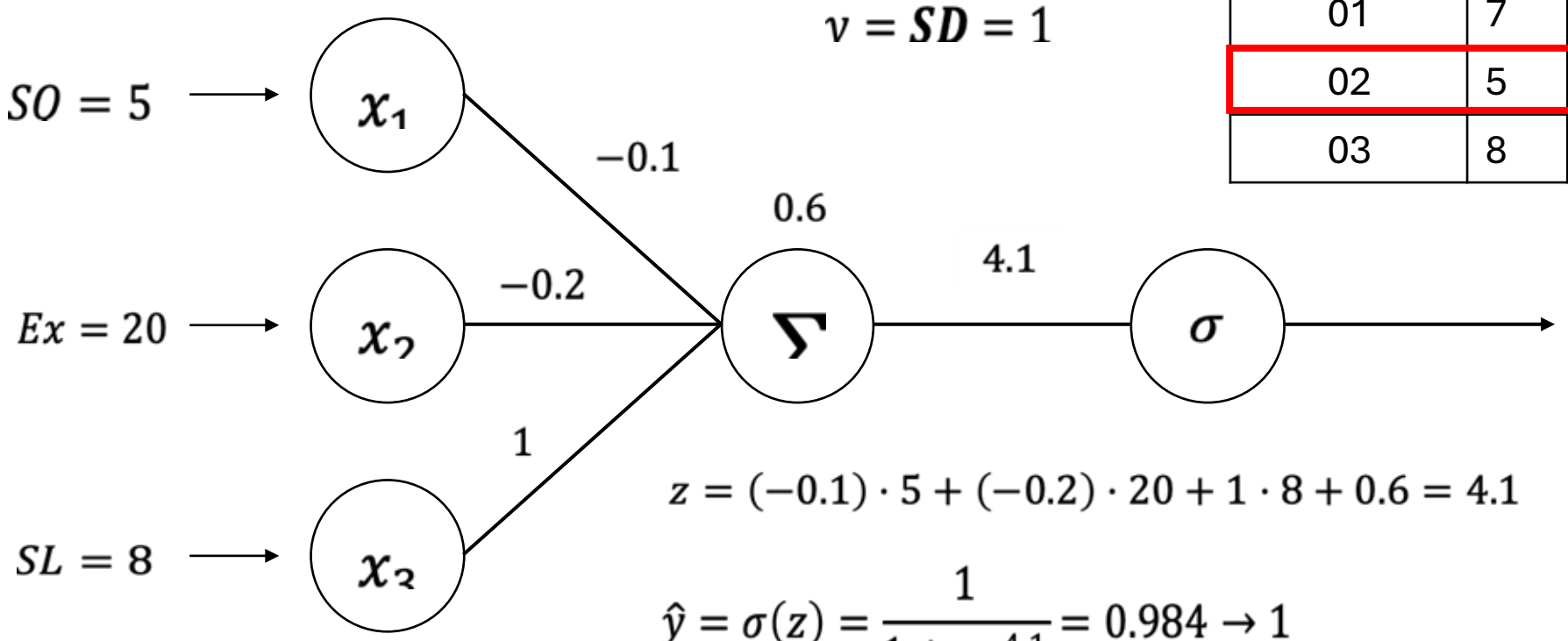
$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-(-2.1)}} = \frac{1}{1 + e^{2.1}} = 0.109 \rightarrow 0$$

$w_1 = -0.1$
$w_2 = -0.2$
$w_3 = 1$
$b = 0.6$

Single neuron model: The Perceptron

x_1 = sleep quality (1 - 10) x_2 = exercise (min/daily) x_3 = stress level (1 - 10)

Patient ID	SQ	Ex	SL	SD
01	7	30	4	0
02	5	20	8	1
03	8	60	8	0



$$z = (-0.1) \cdot 5 + (-0.2) \cdot 20 + 1 \cdot 8 + 0.6 = 4.1$$

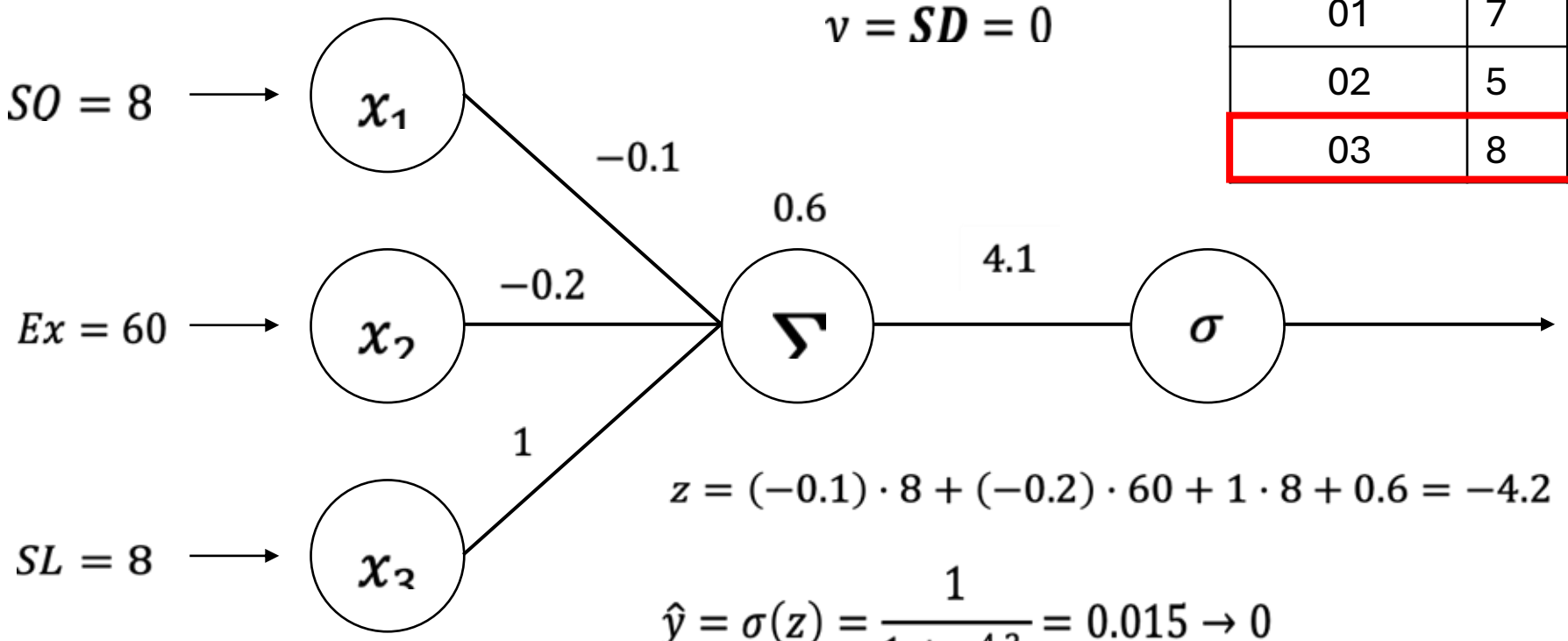
$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-4.1}} = 0.984 \rightarrow 1$$

$w_1 = -0.1$
$w_2 = -0.2$
$w_3 = 1$
$b = 0.6$

Single neuron model: The Perceptron

x_1 = sleep quality (1 - 10) x_2 = exercise (min/daily) x_3 = stress level (1 - 10)

Patient ID	SQ	Ex	SL	SD
01	7	30	4	0
02	5	20	8	1
03	8	60	8	0



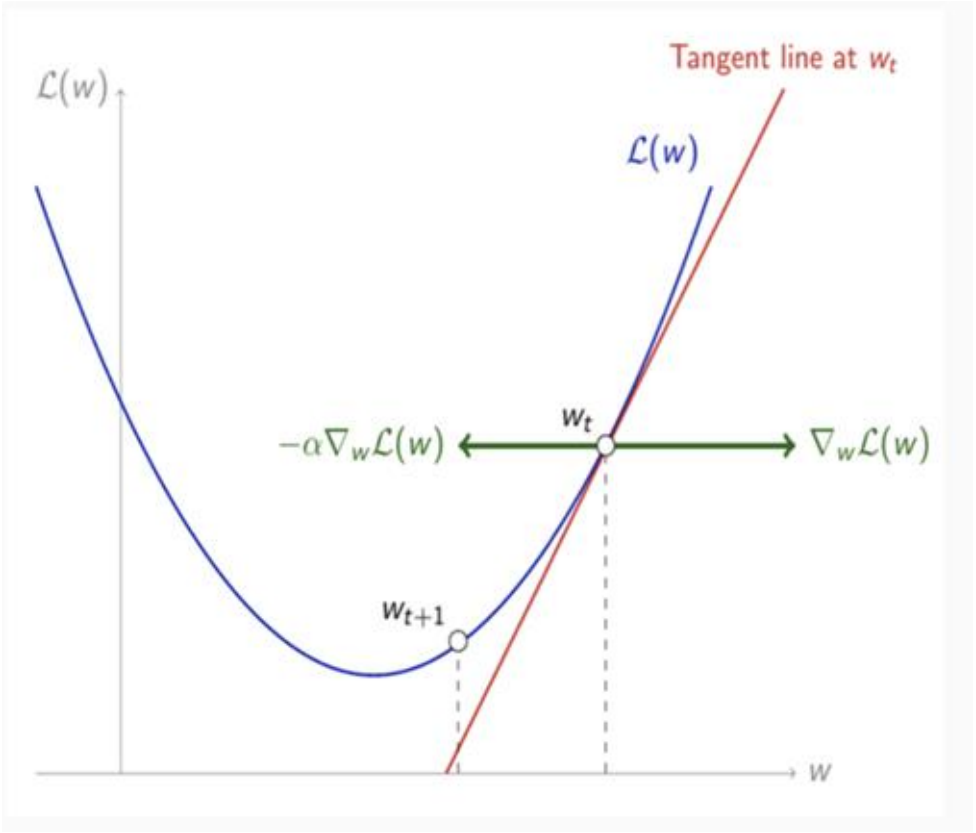
$$z = (-0.1) \cdot 8 + (-0.2) \cdot 60 + 1 \cdot 8 + 0.6 = -4.2$$

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{4.2.}} = 0.015 \rightarrow 0$$

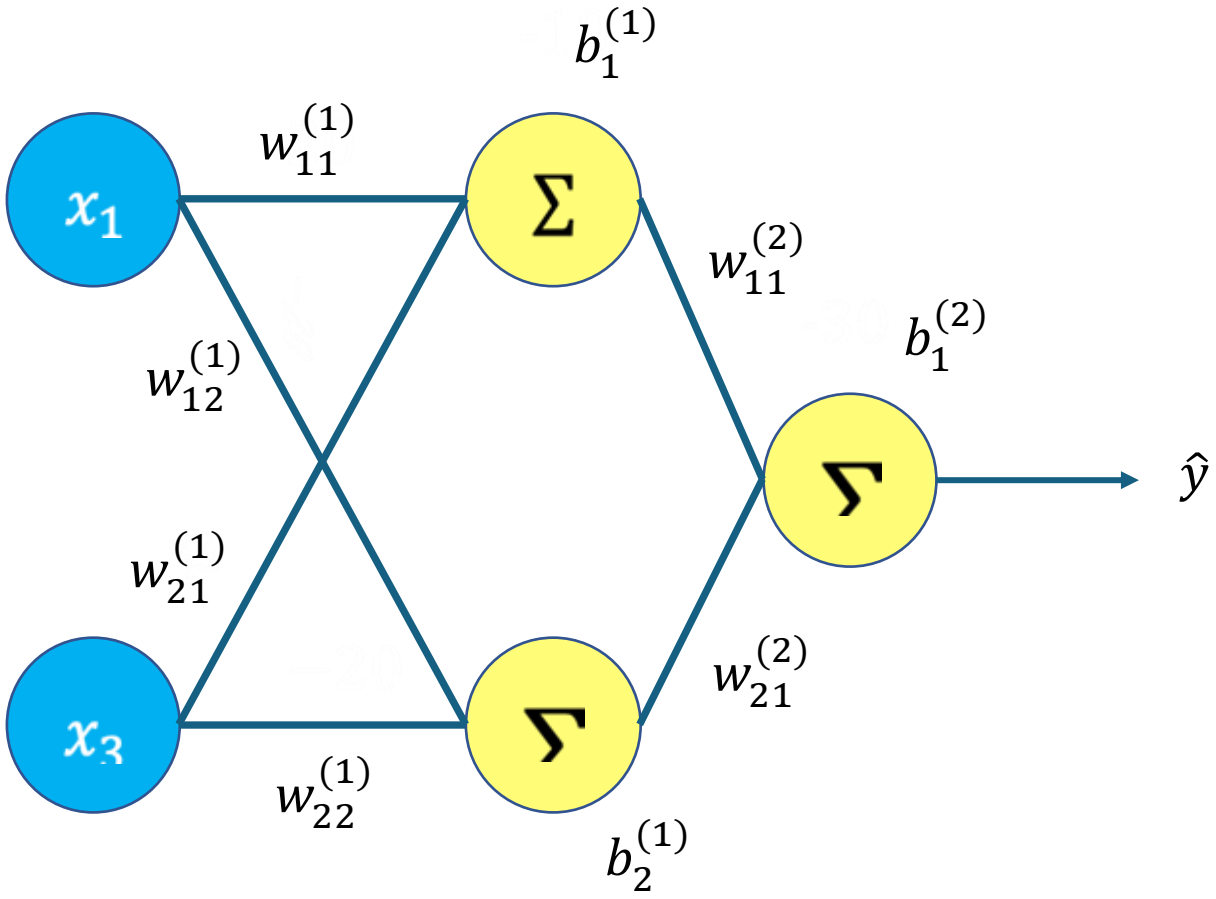
$w_1 = -0.1$
$w_2 = -0.2$
$w_3 = 1$
$b = 0.6$

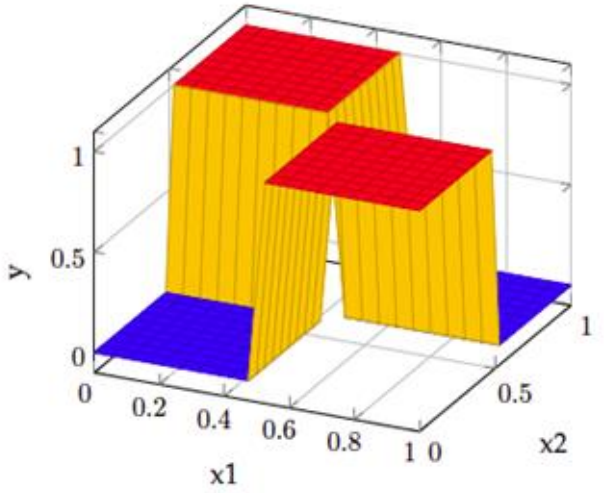
Training a Perceptron

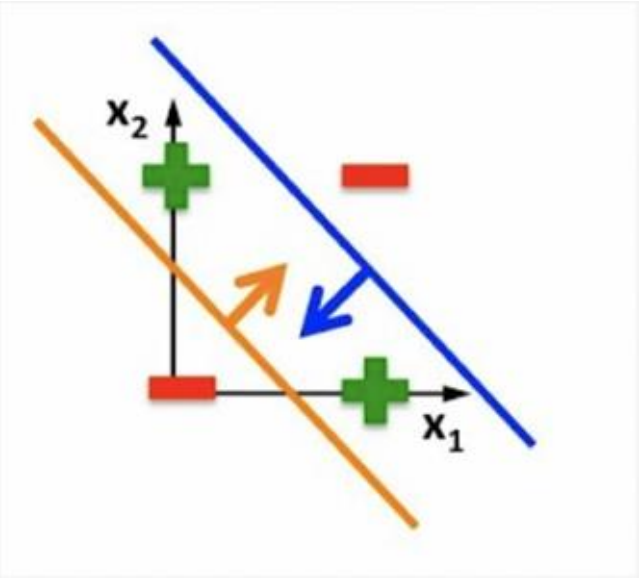
- How do we find w_1, w_2, w_3 and b?
- Using the training data, we will gradually adjust the values of the parameters such that we minimize the errors in the predictions
- Optimization algorithm: **gradient descent** $\theta_i := \theta_i - \alpha \frac{\partial \mathcal{L}}{\partial \theta_i}$



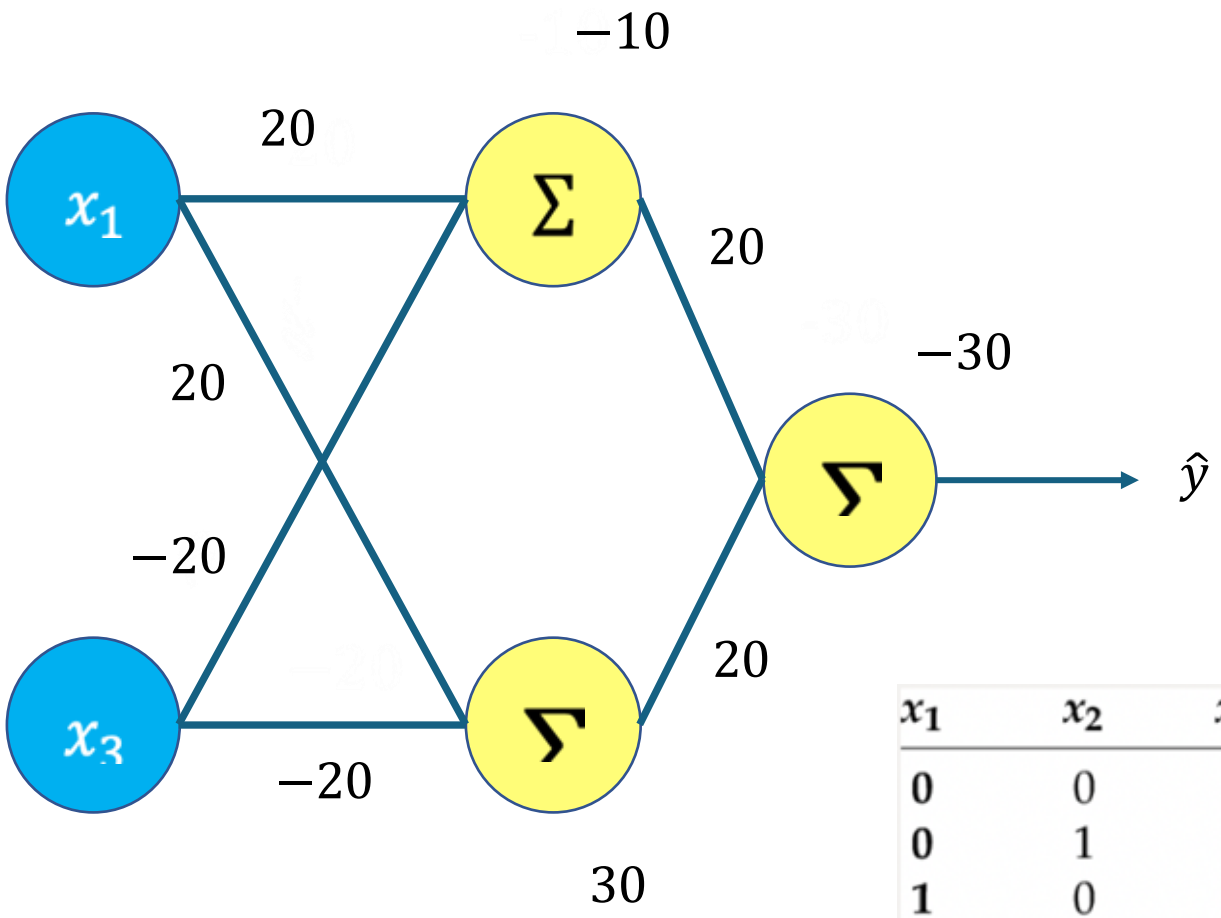
The perceptron and the XOR problem

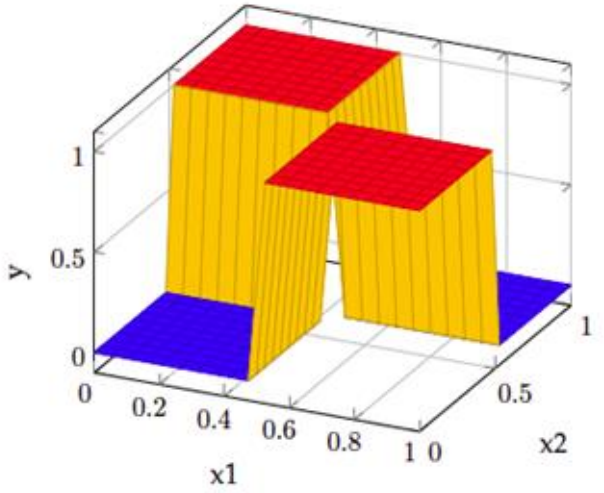




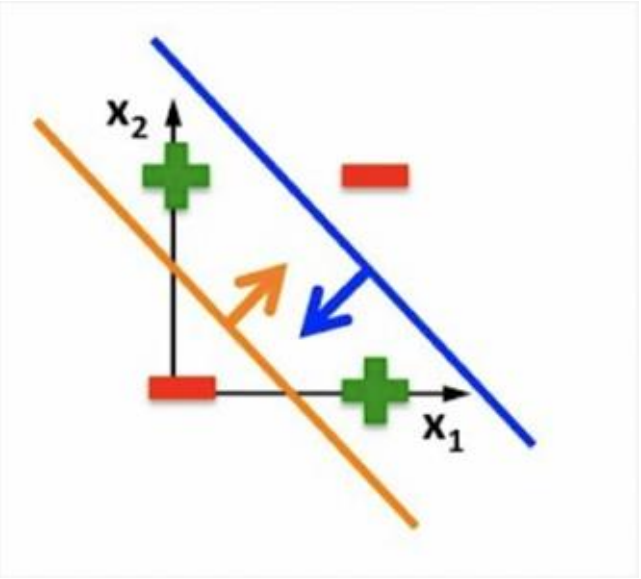


The perceptron and the XOR problem

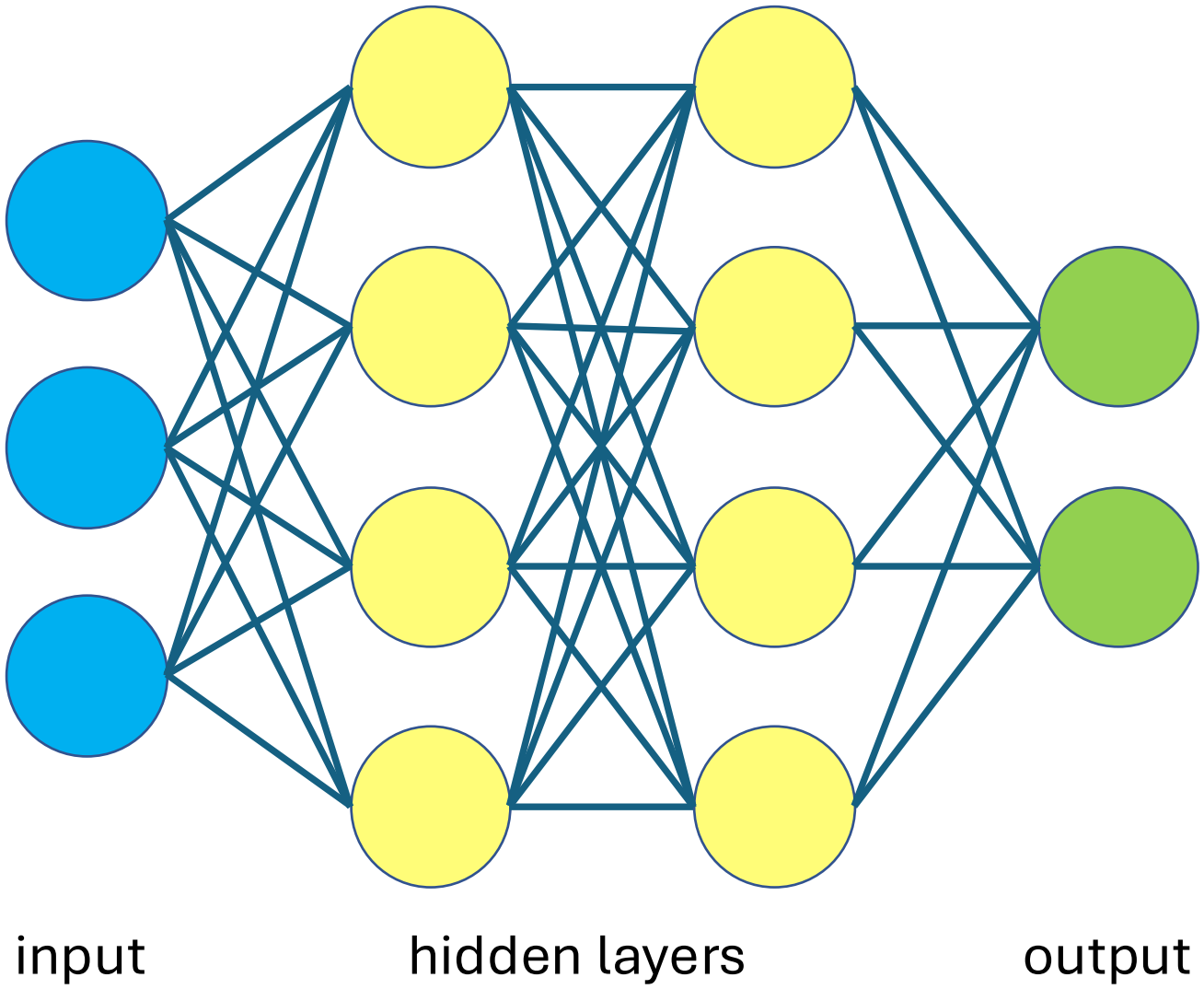




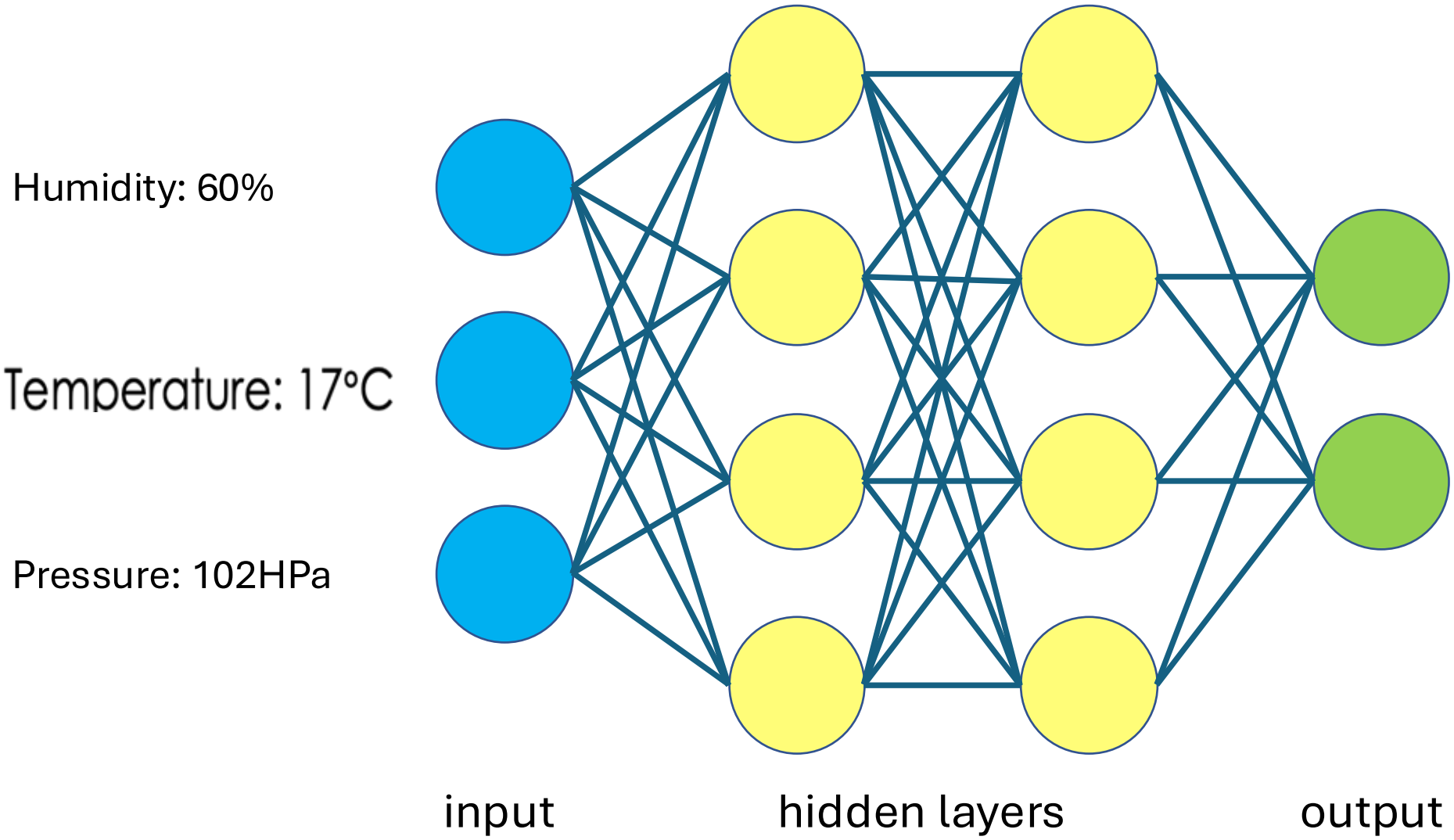
x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0



Artificial Neural Network



Artificial Neural Network

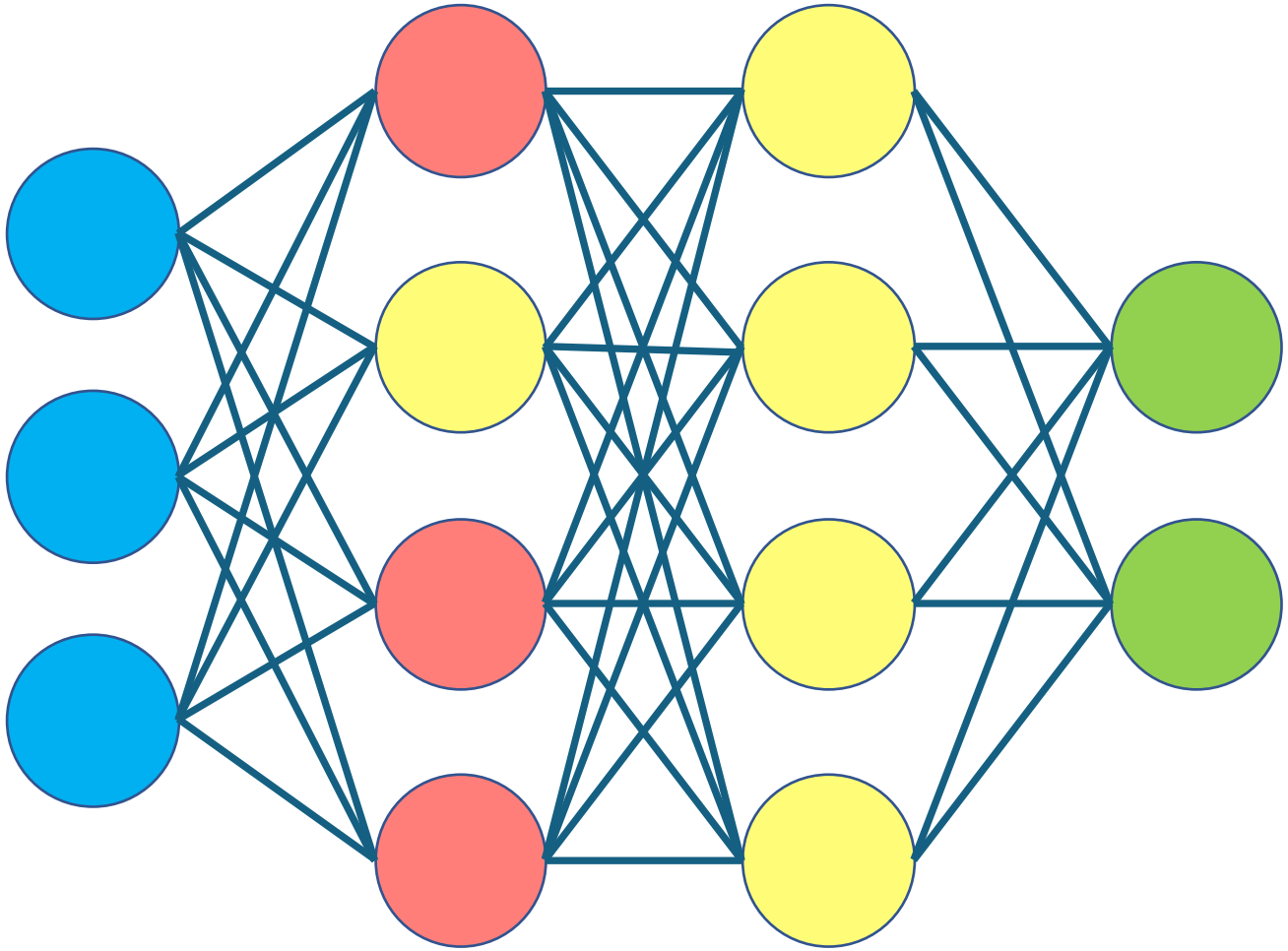


Artificial Neural Network

Humidity: 60%

Temperature: 17°C

Pressure: 102HPa



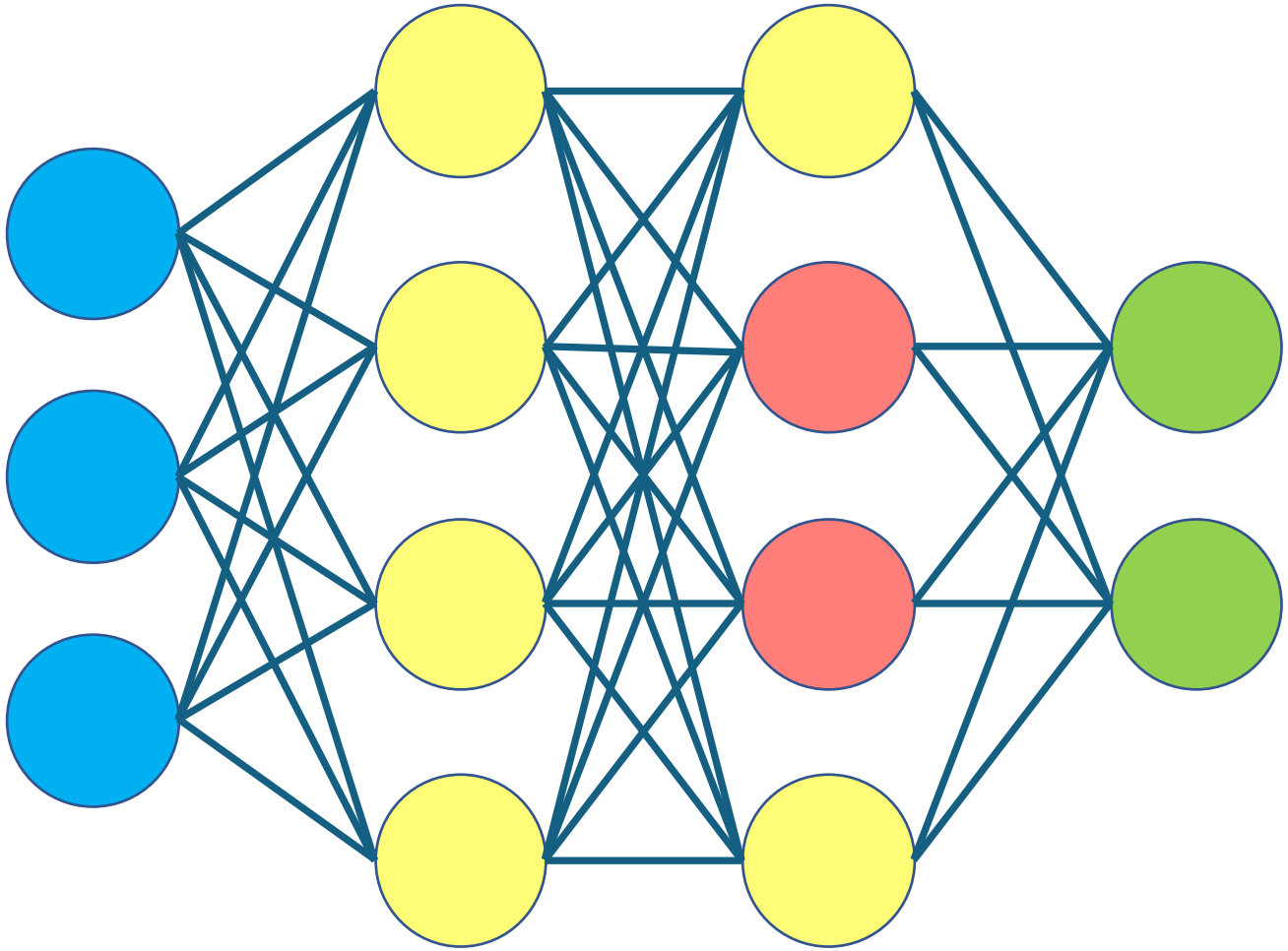
input hidden layers output

Artificial Neural Network

Humidity: 60%

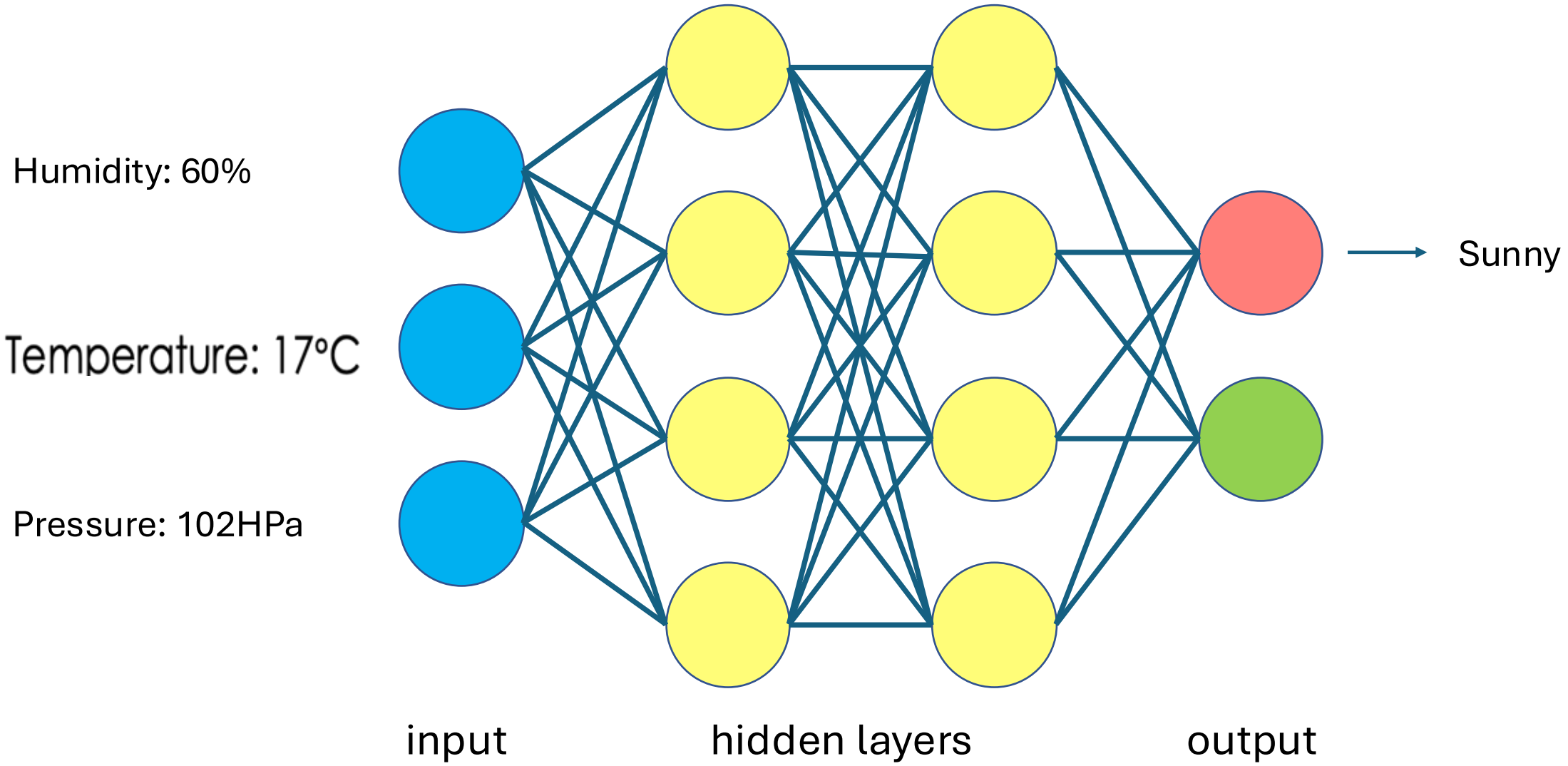
Temperature: 17°C

Pressure: 102HPa



input hidden layers output

Artificial Neural Network

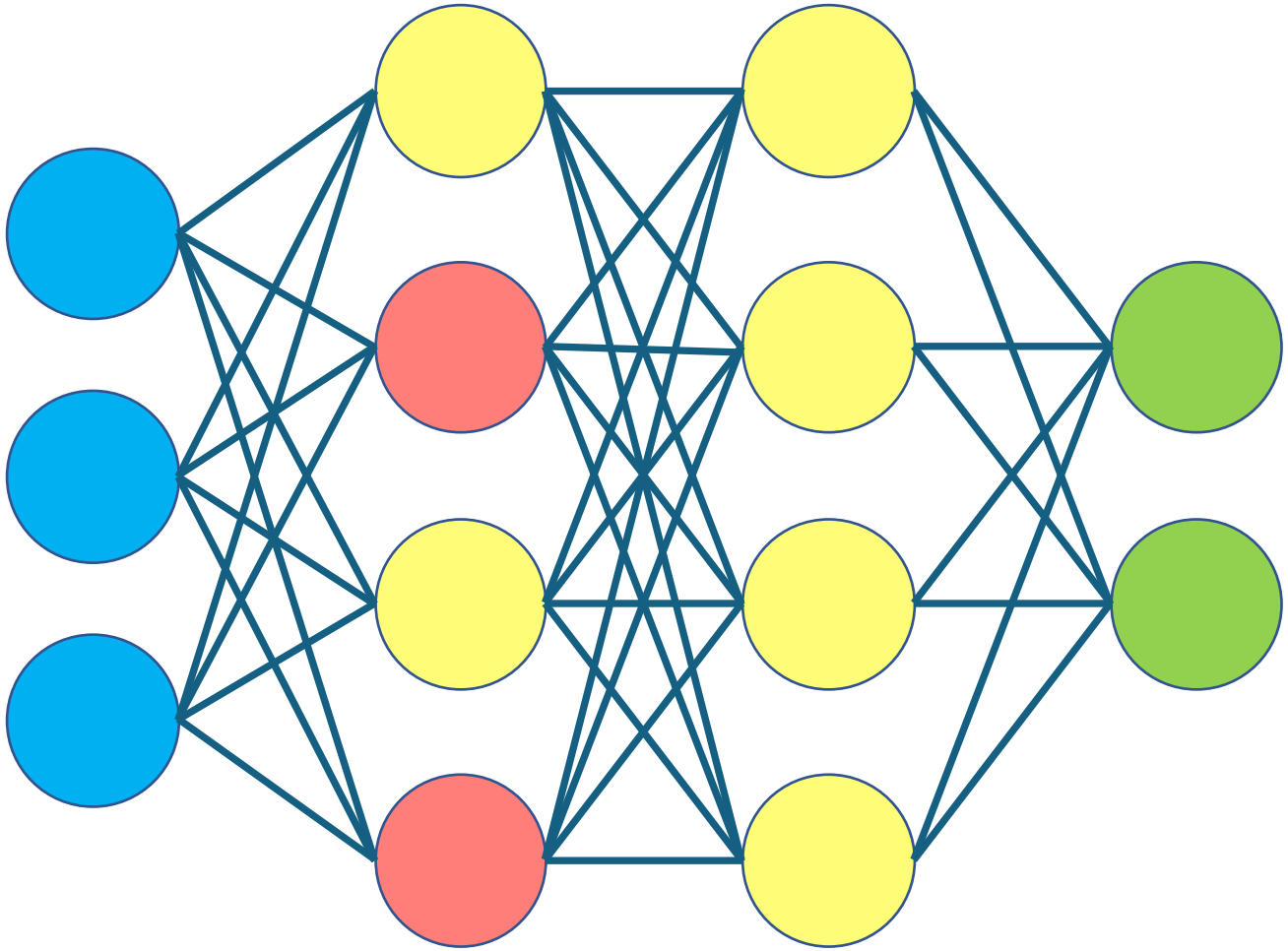


Artificial Neural Network

Humidity: 90%

Temperature: 16°C

Pressure: 100HPa



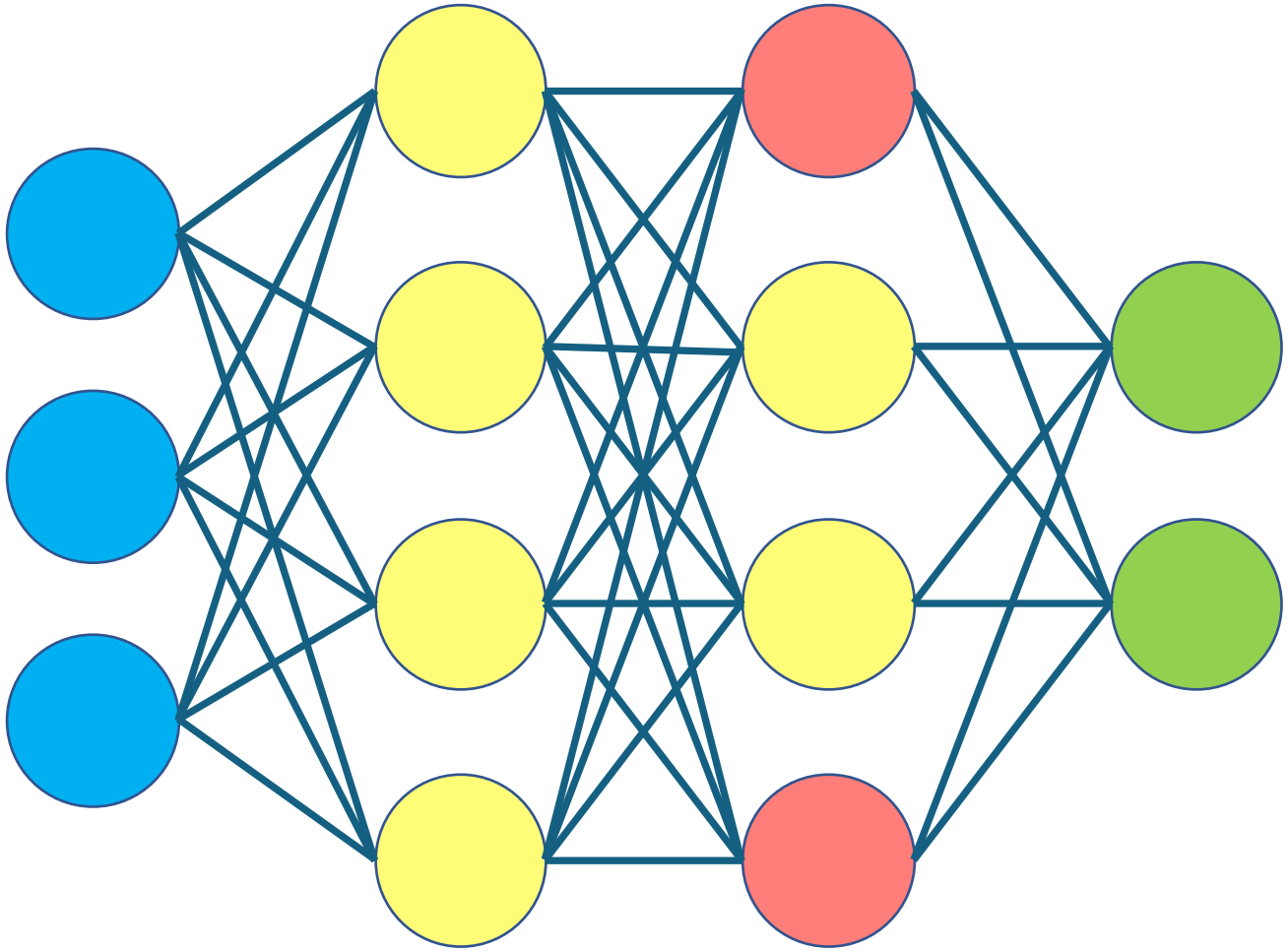
input hidden layers output

Artificial Neural Network

Humidity: 90%

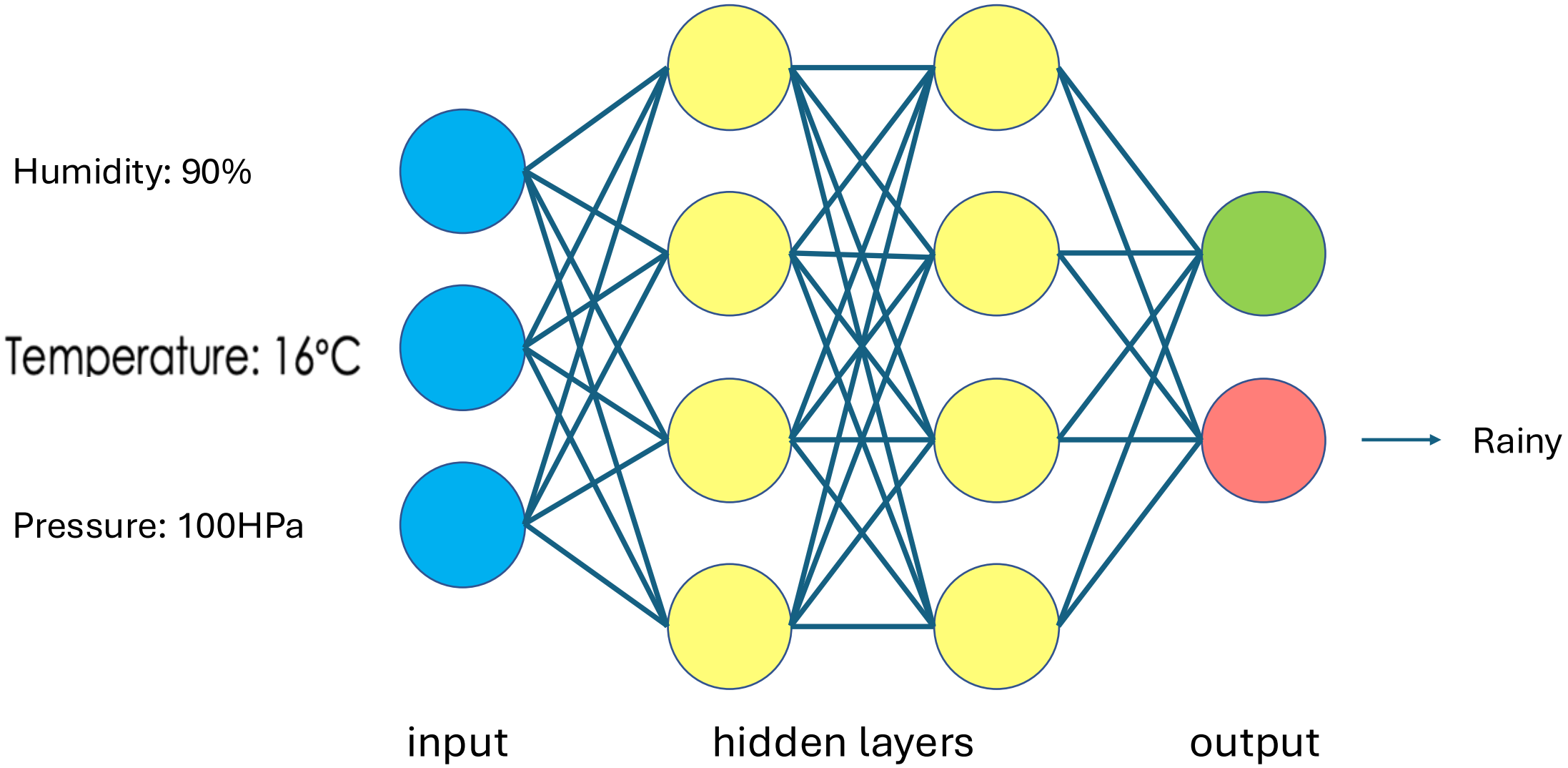
Temperature: 16°C

Pressure: 100HPa



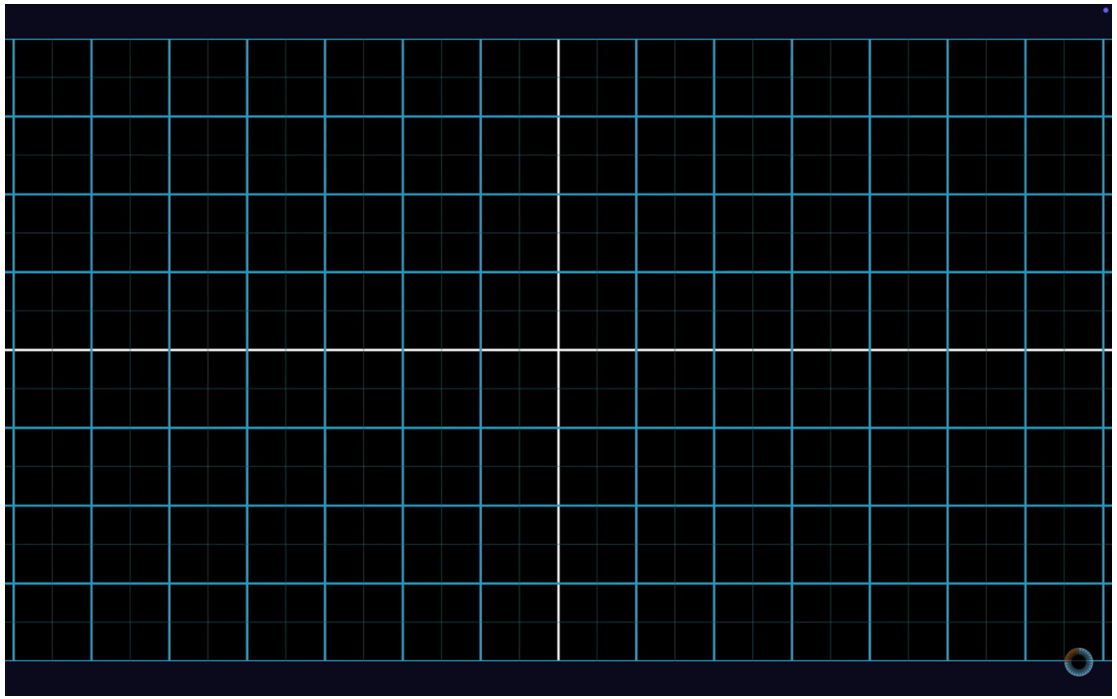
input hidden layers output

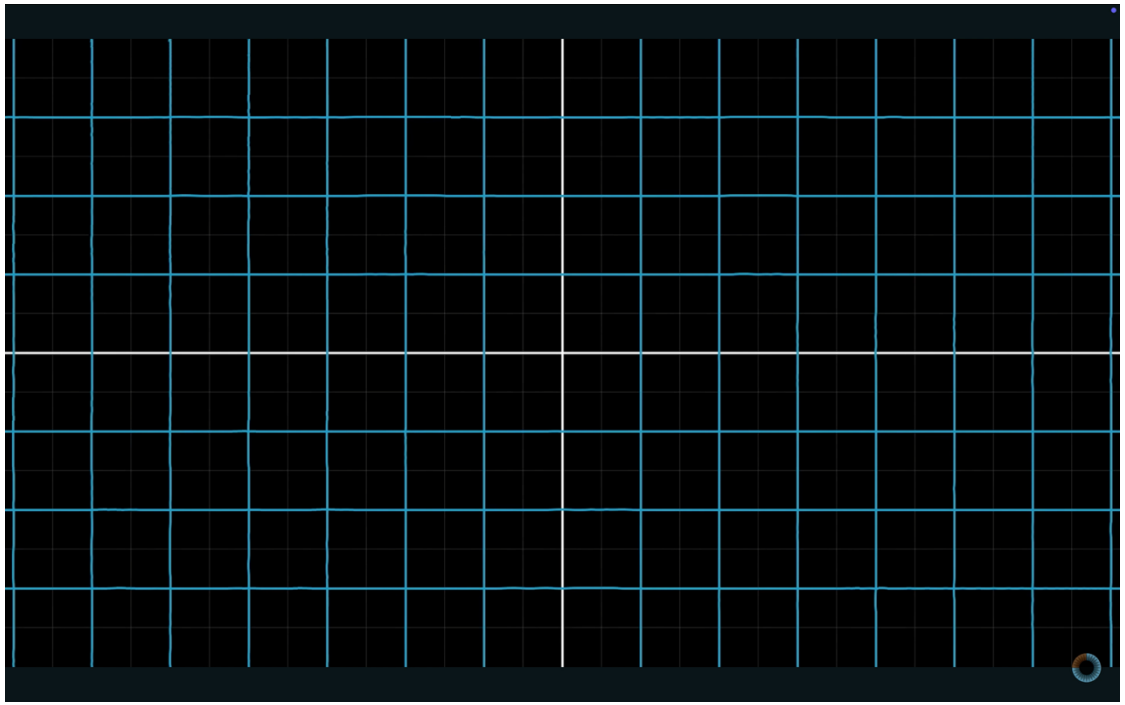
Artificial Neural Network



Matrices as linear transformations

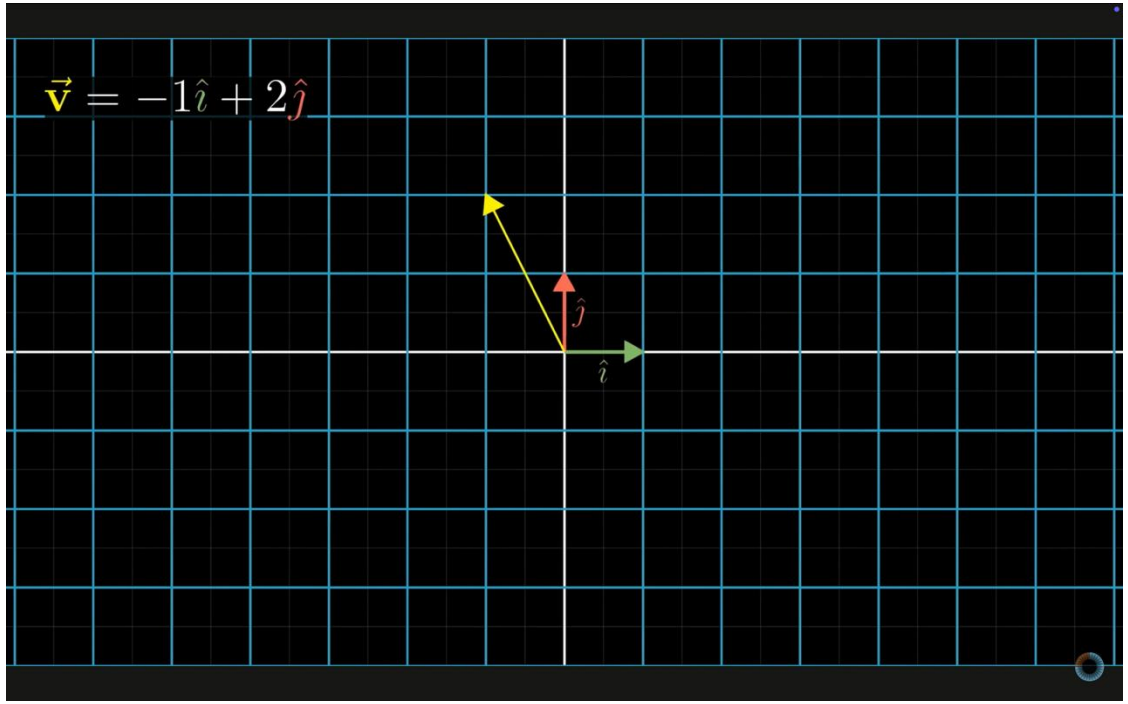
- Center remains at origin
- Lines remain parallel

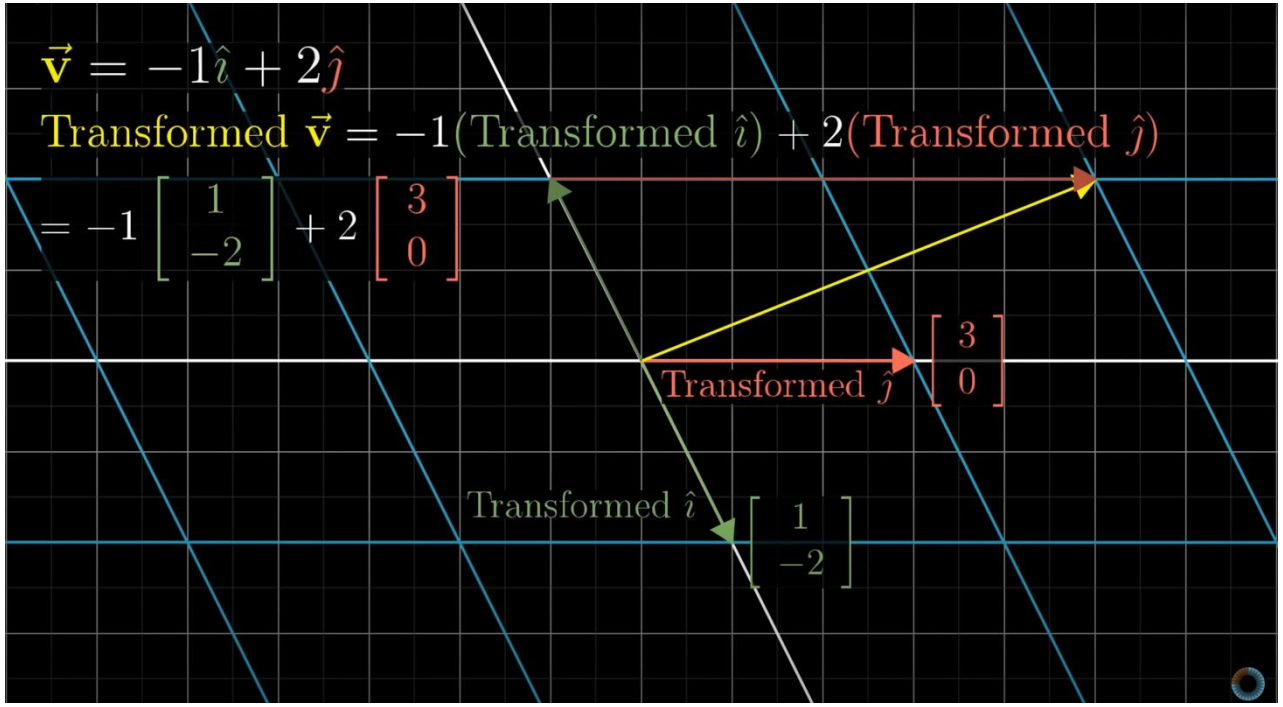




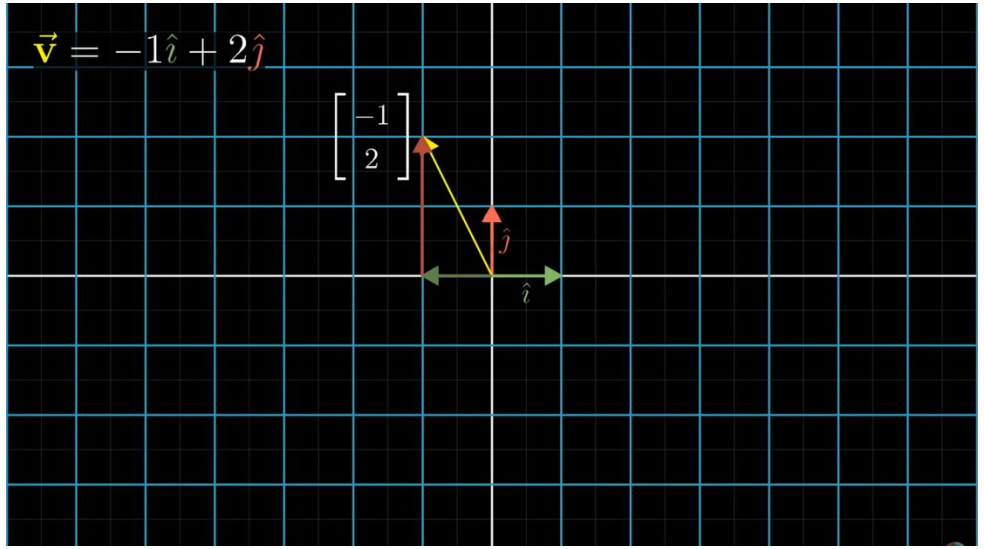
https://www.youtube.com/@3blue1brown

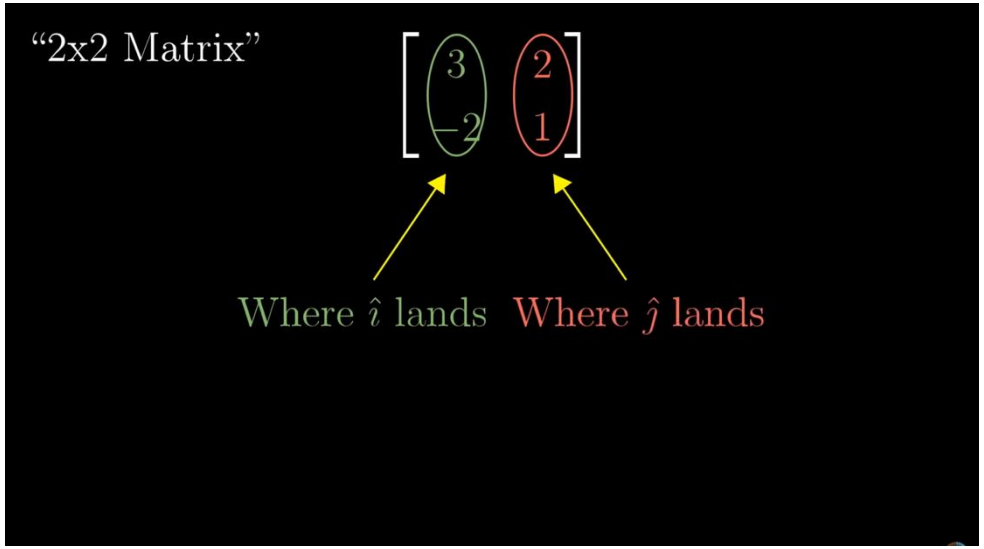
Matrices as linear transformations

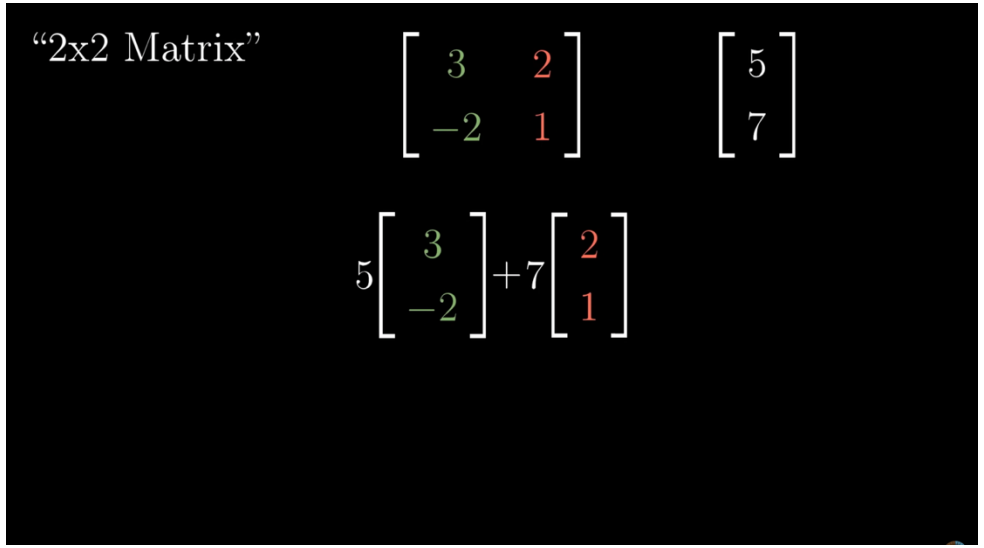


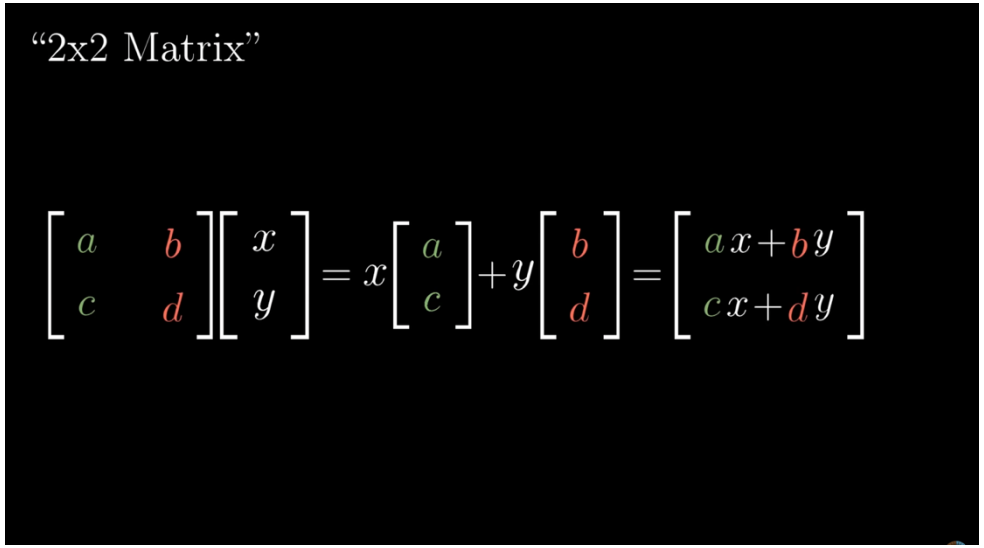


Matrices as linear transformations



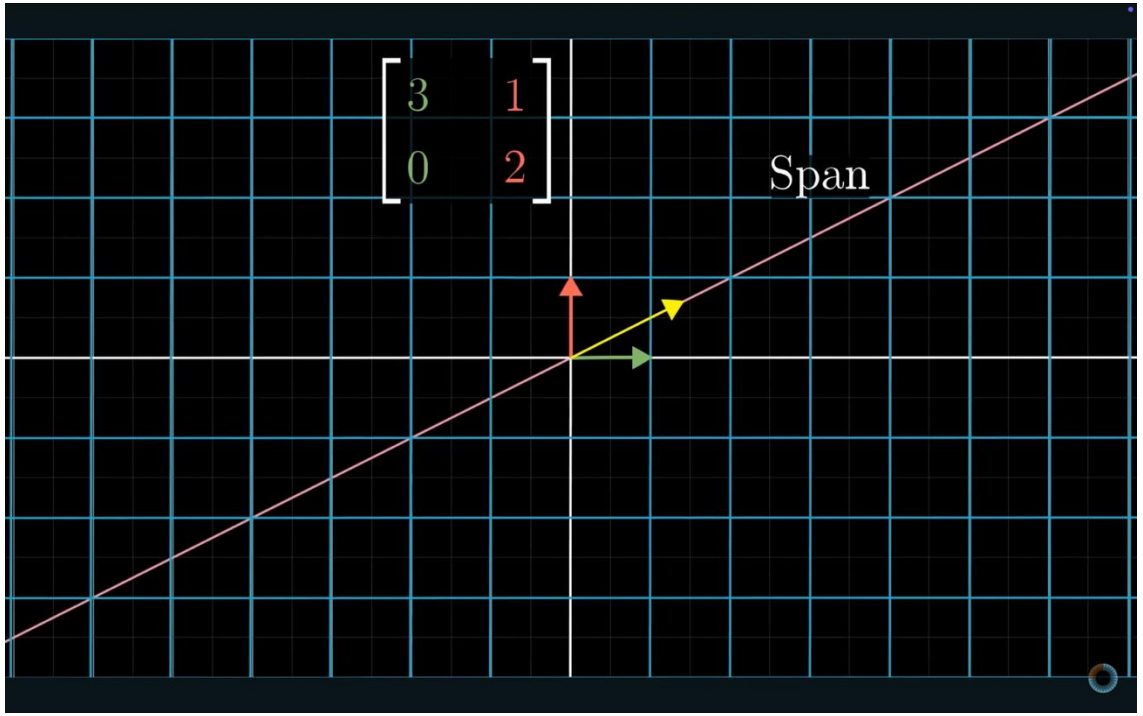




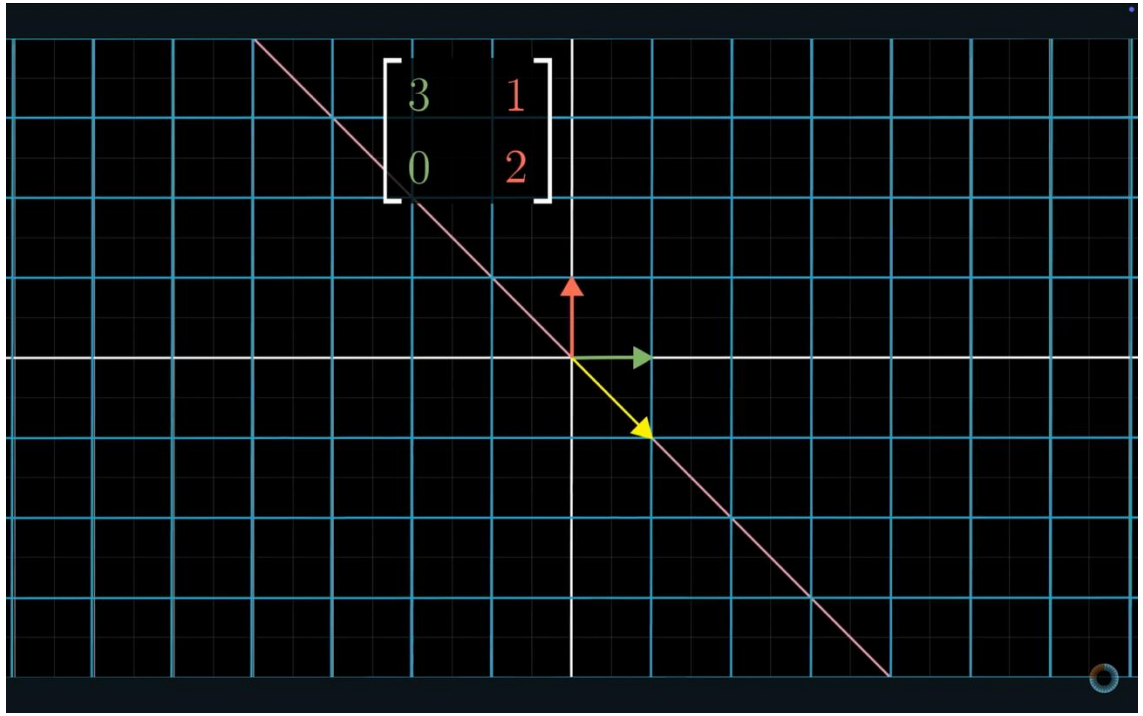


Matrices as linear transformations

- Eigenvectors are only scaled by the eigenvalues

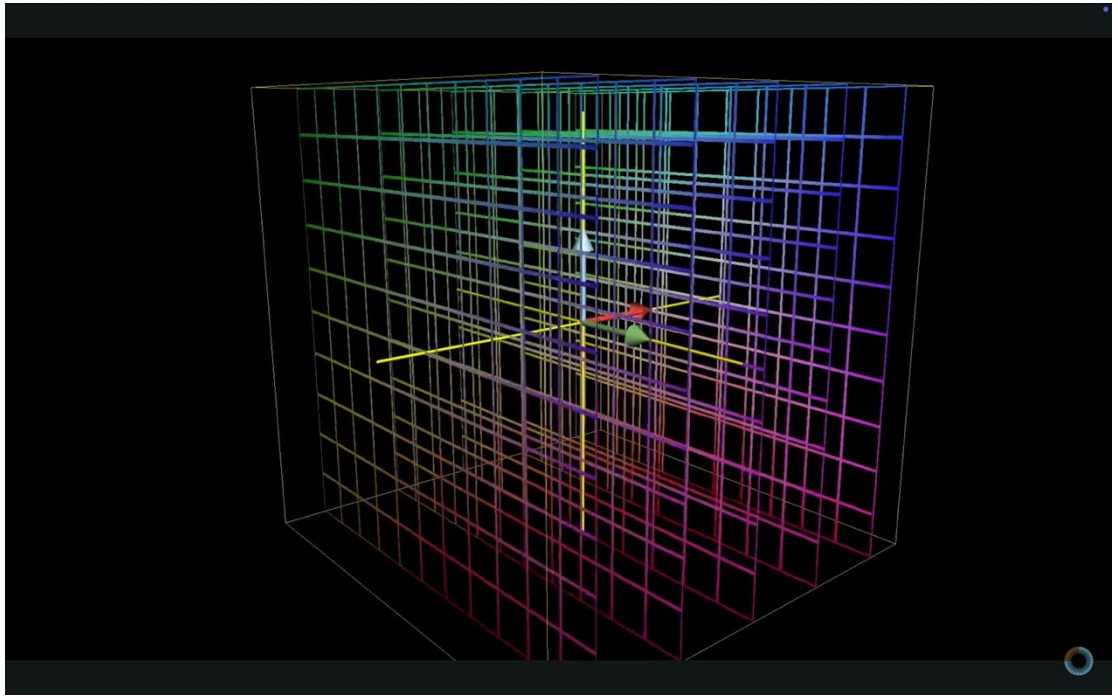


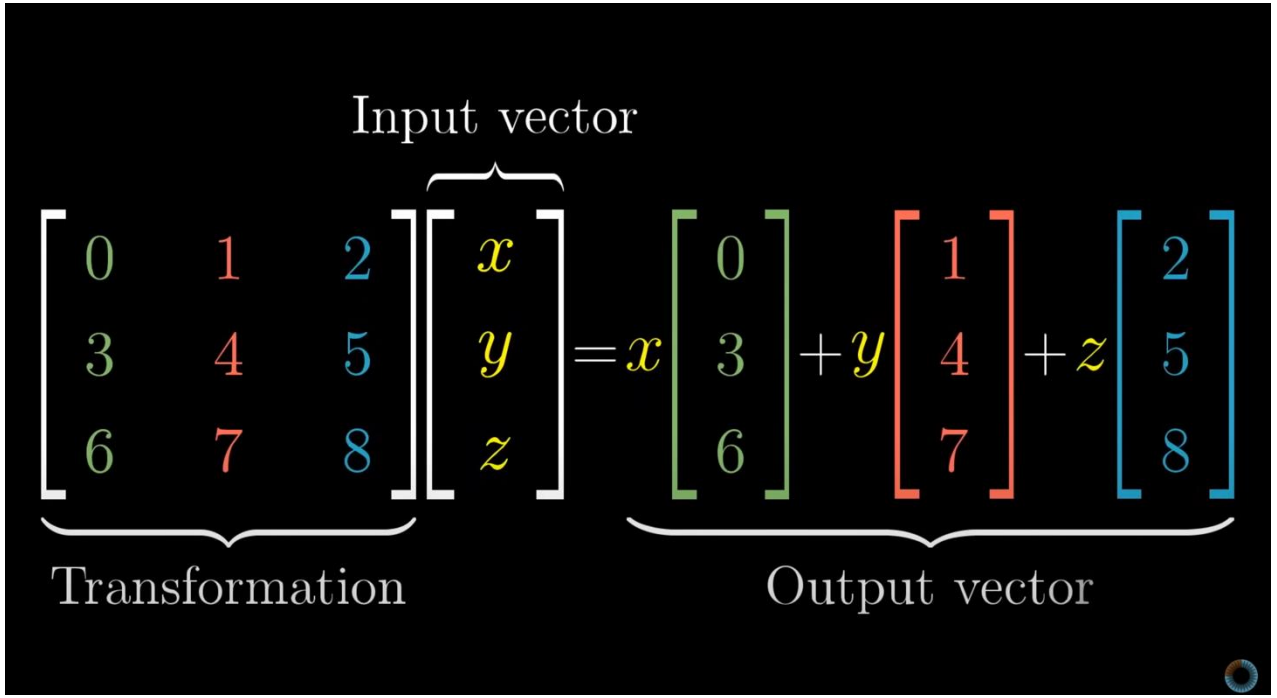
No eigenvector



Eigenvector

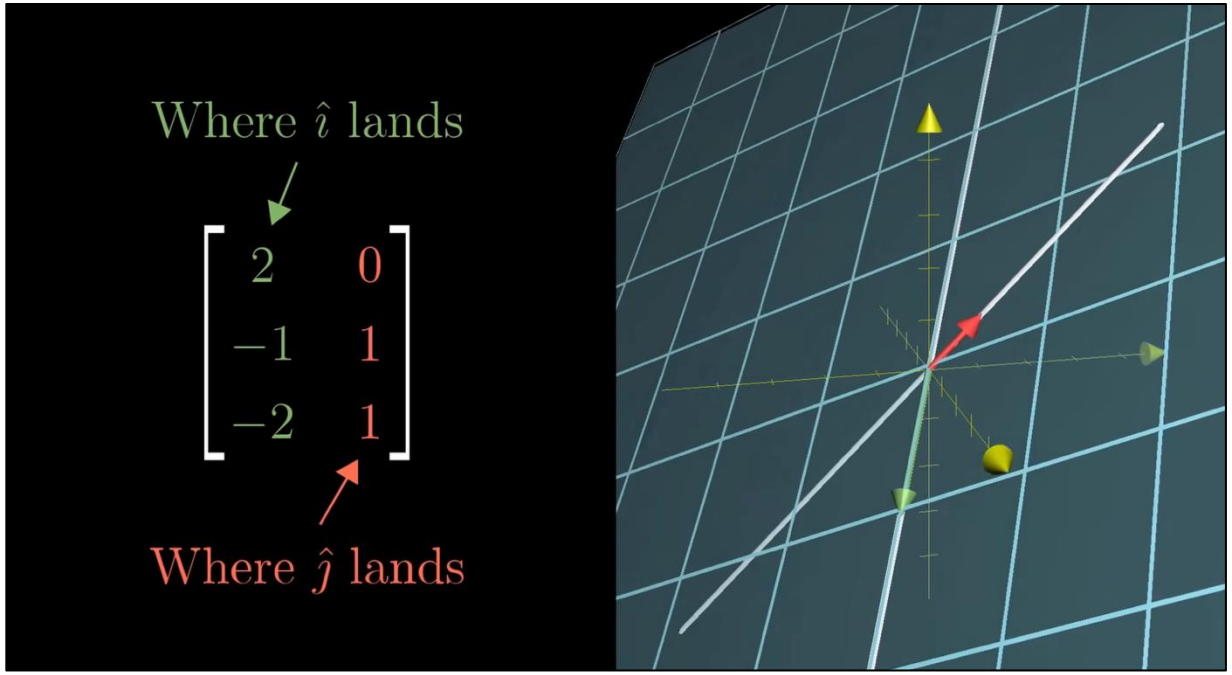
Linear transformations in 3D

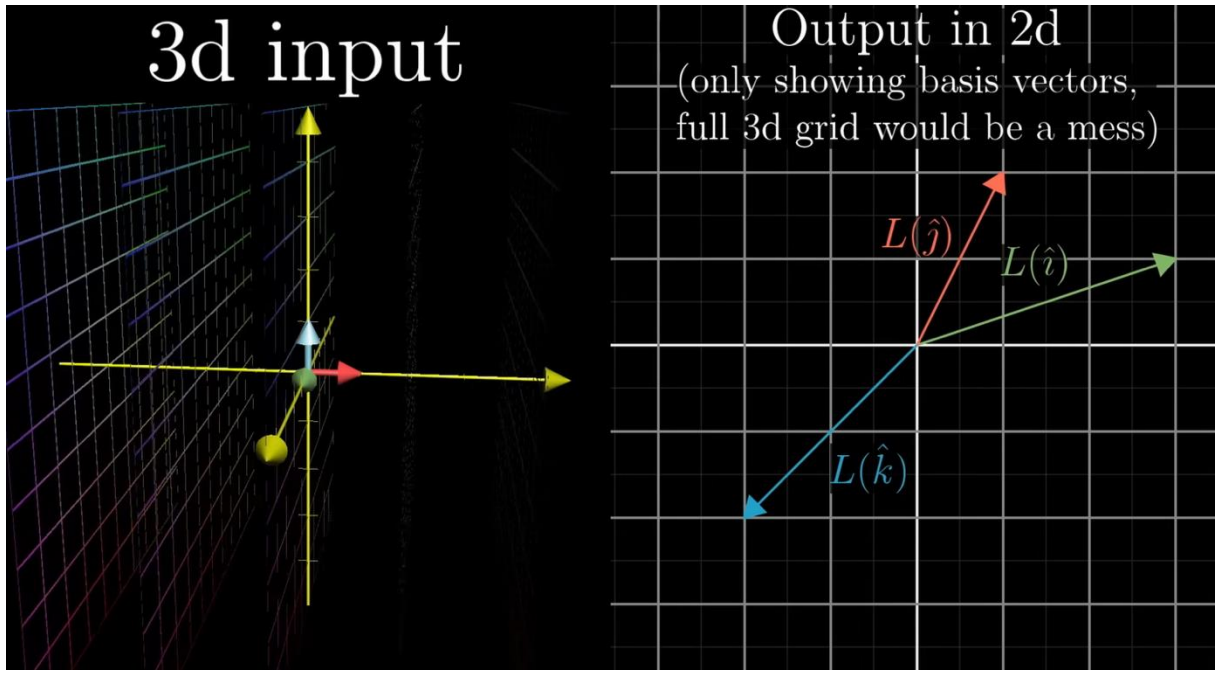




Non-square matrices as transformations between dimensions

- What transformation does a 3x2 matrix represent? And a 3x2?

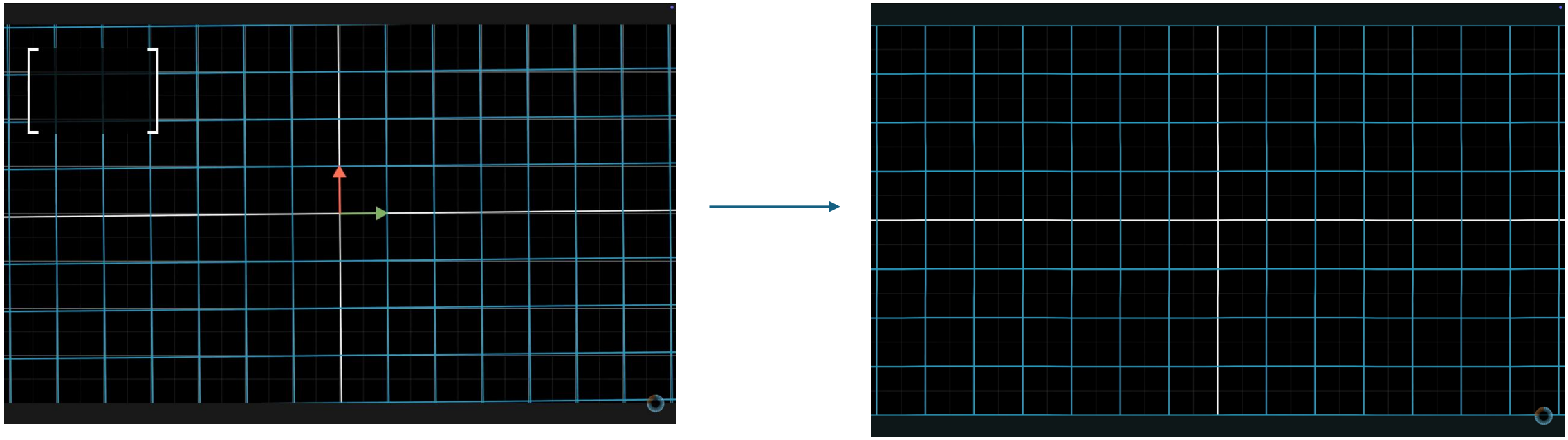




Matrix multiplication is linear → we can repeatedly apply transformations to a vector between any number of dimensions

Neural Networks

- Each layer of (trainable) weights applies a linear transformation to the input vector, followed by a non-linear activation. The output of a given layer becomes the input to the following layer

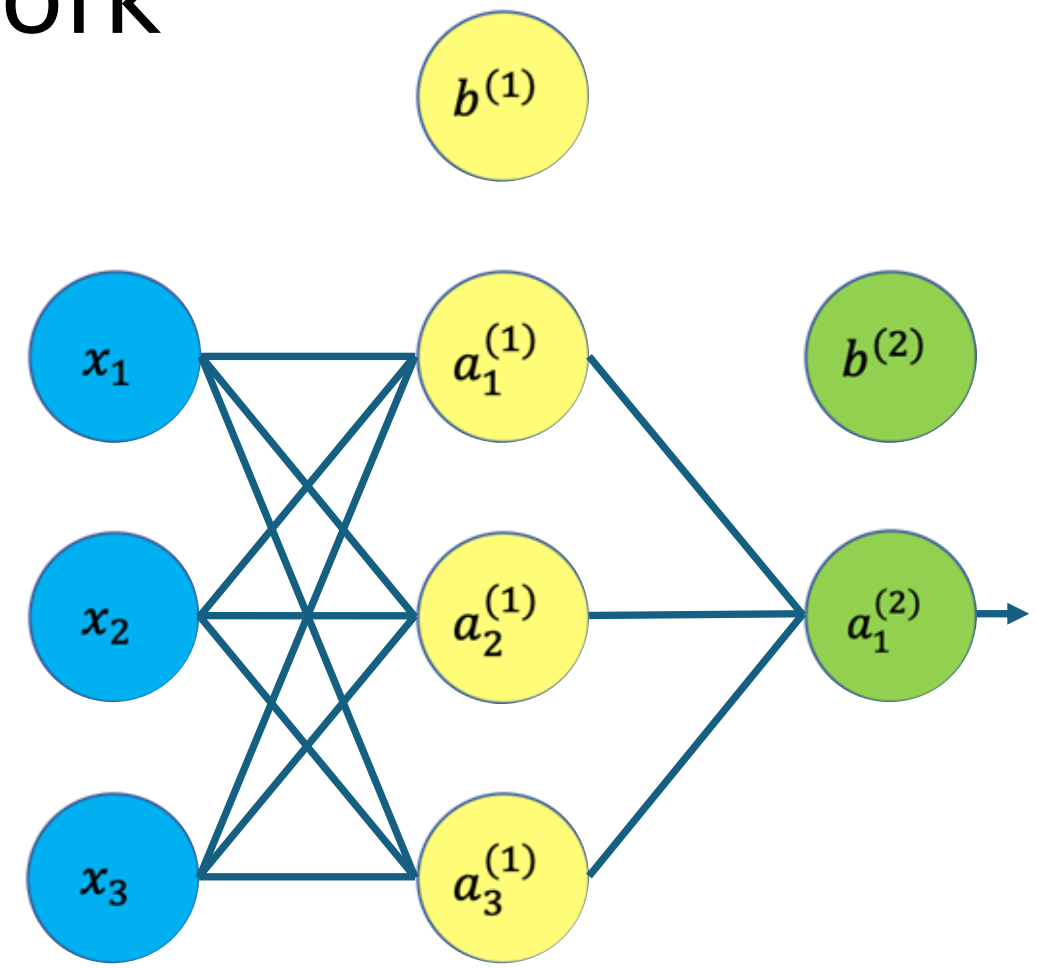


Building a neural network

- **Input layer (layer 0):** this layer consists of the input features x_1, x_2 and x_3
- **Hidden layer (layer 1):** contains values that we don't observe in the training set
- **Output layer (layer 2):** it has a neuron/units (can have more units) that outputs the final value computed by the hypothesis

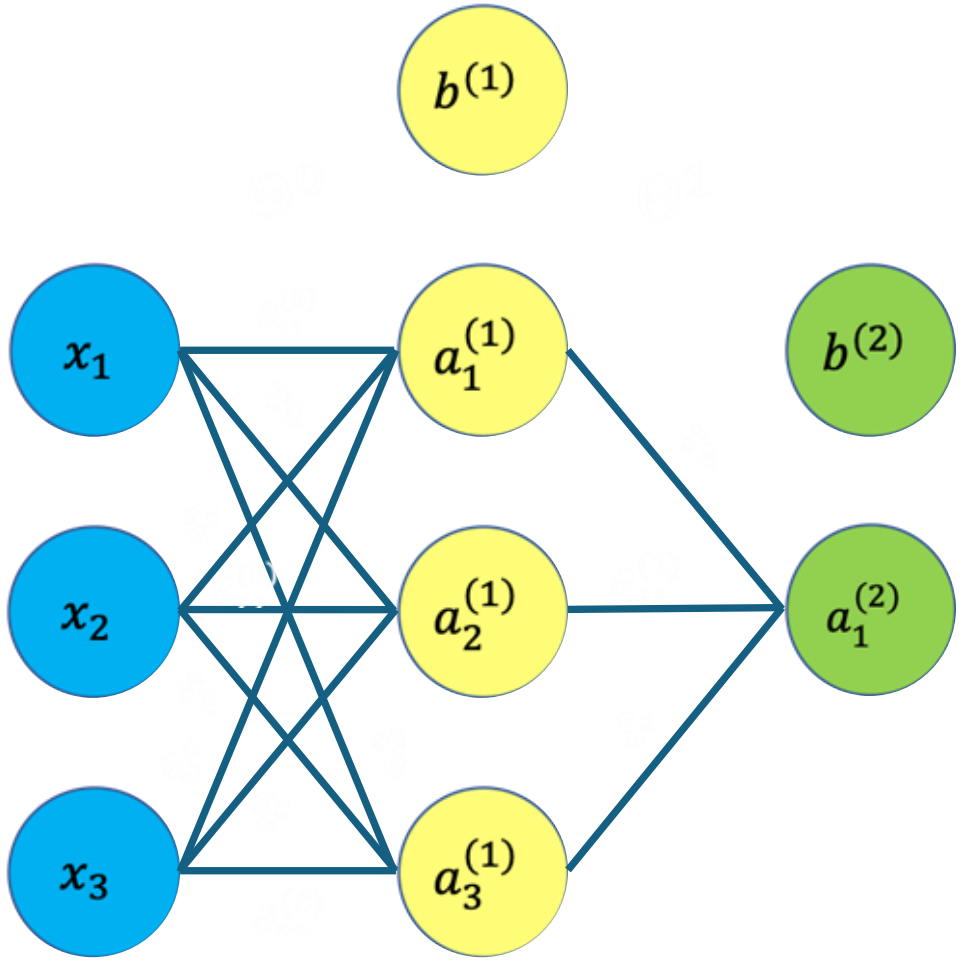
$f: \mathbb{R}^M \rightarrow \mathbb{R}^N$

$f: \mathbb{R}^3 \rightarrow \mathbb{R}^1$



Building a neural network

$$f: \mathbb{R}^3 \rightarrow \mathbb{R}^1$$



$$a_1^{(1)} = g(\theta_{11}^{(0)} x_1 + \theta_{21}^{(0)} x_2 + \theta_{31}^{(0)} x_3 + b_1^{(1)})$$
$$a_2^{(1)} = g(\theta_{12}^{(0)} x_1 + \theta_{22}^{(0)} x_2 + \theta_{32}^{(0)} x_3 + b_2^{(1)})$$
$$a_3^{(1)} = g(\theta_{13}^{(0)} x_1 + \theta_{23}^{(0)} x_2 + \theta_{33}^{(0)} x_3 + b_3^{(1)})$$

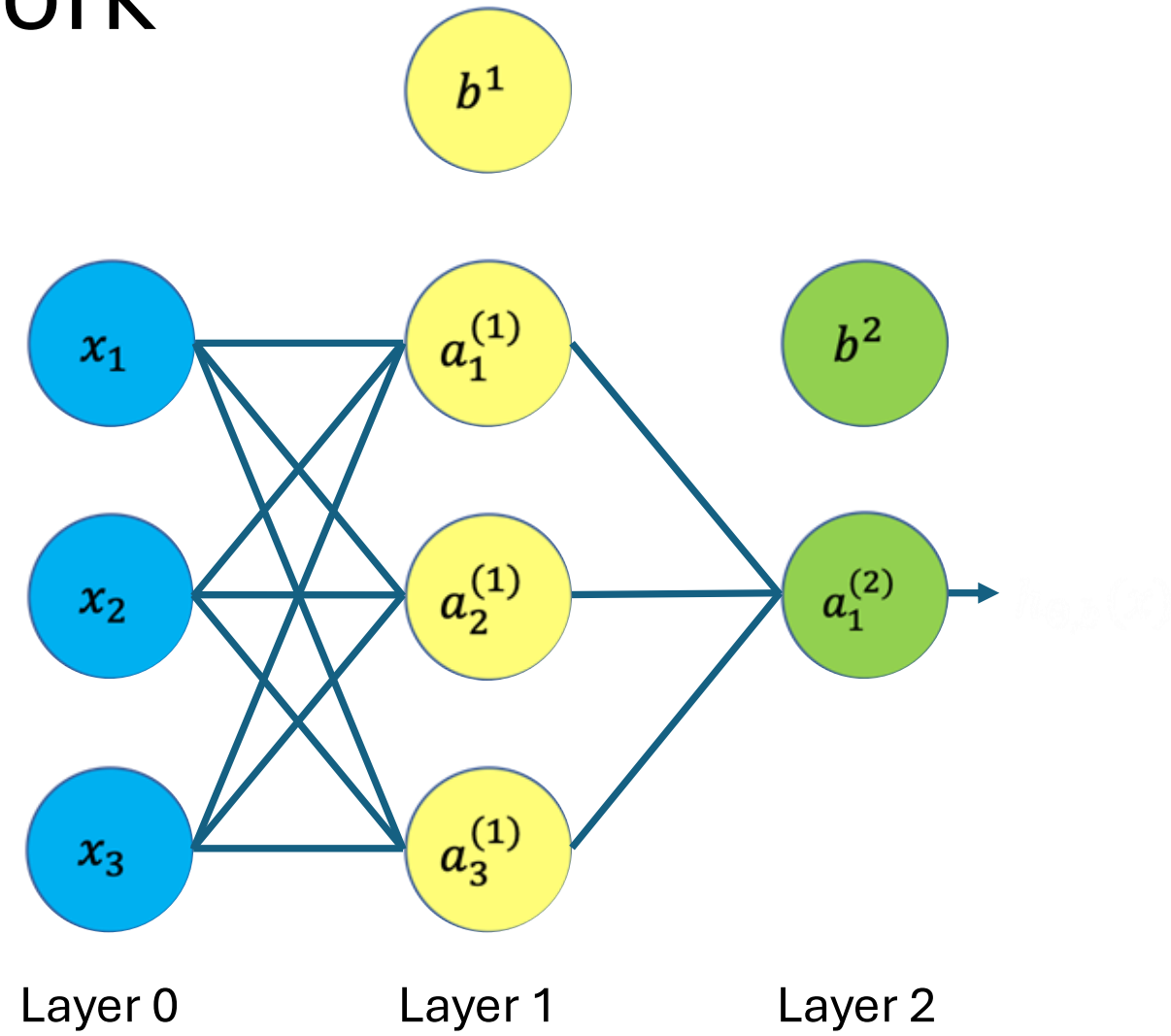
$$h_{\Theta,b}(x) = a_1^{(2)} = \theta_{11}^{(1)} a_1^{(1)} + \theta_{21}^{(1)} a_2^{(1)} + \theta_{31}^{(1)} a_3^{(1)} + b_1^{(2)}$$

$$\Theta^{(0)} = \begin{pmatrix} \theta_{11}^{(0)} & \theta_{12}^{(0)} & \theta_{13}^{(0)} \\ \theta_{21}^{(0)} & \theta_{22}^{(0)} & \theta_{23}^{(0)} \\ \theta_{31}^{(0)} & \theta_{32}^{(0)} & \theta_{33}^{(0)} \end{pmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad b^{(1)} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(1)} \\ b_3^{(1)} \end{bmatrix}$$

$$a^{(1)} = \begin{bmatrix} a_1^{(1)} \\ a_2^{(1)} \\ a_3^{(1)} \end{bmatrix} = (\Theta^{(0)})^T \cdot x + b^{(1)}$$

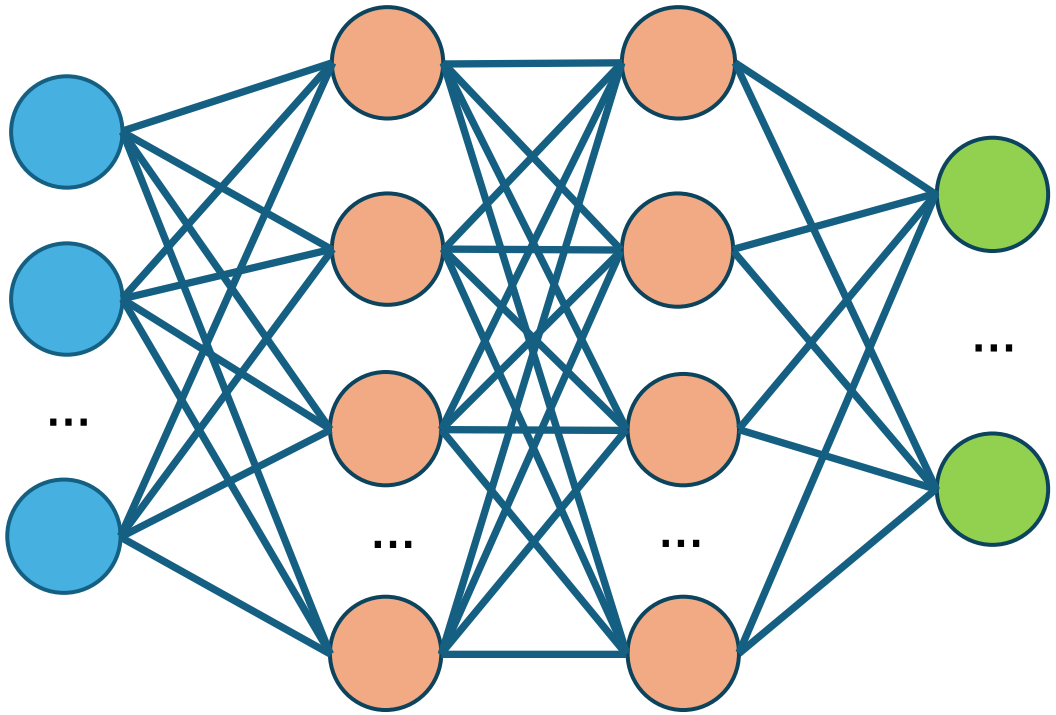
Building a neural network

- In this example the computation of $h_{\Theta,b}(x)$ is known as **forward propagation**
- It starts with the multiplication of the input units with the weights of the first layer.
- It propagates to the hidden layer computing the activation function of the second layer.
- The propagation follows with the activation of the output layer



Number of parameters

- How many parameters does the following neural network have?



- 3 layer neural network with 2 hidden layers
- Sigmoid activation funcions
- Softmax on the last layer

Universal approximation theorem

- A feedforward neural network with a single hidden layer containing a finite number of neurons can approximate any continuous function on compact subsets of $\mathbb{R}^n$, given appropriate activation functions
- Arbitrary width
- Arbitrary depth

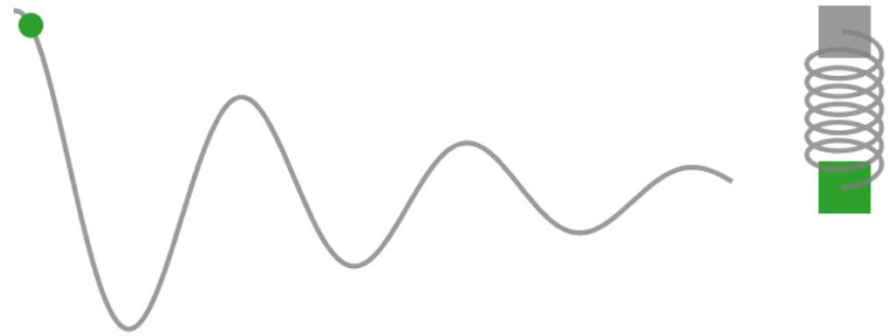
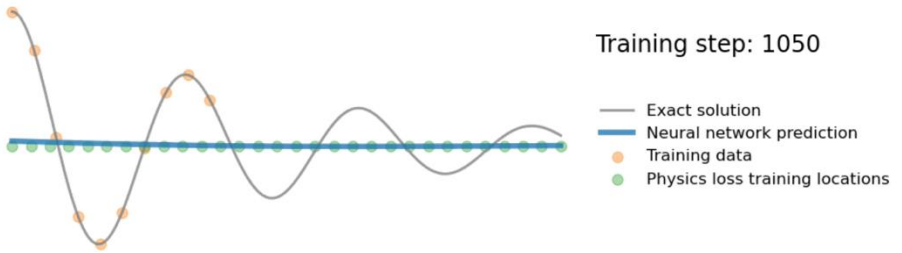


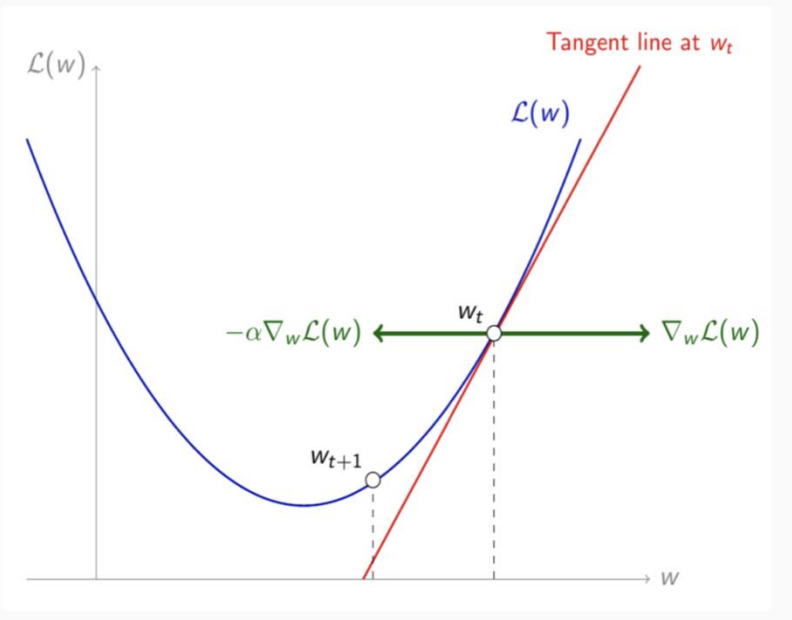
Fig 3: a 1D damped harmonic oscillator

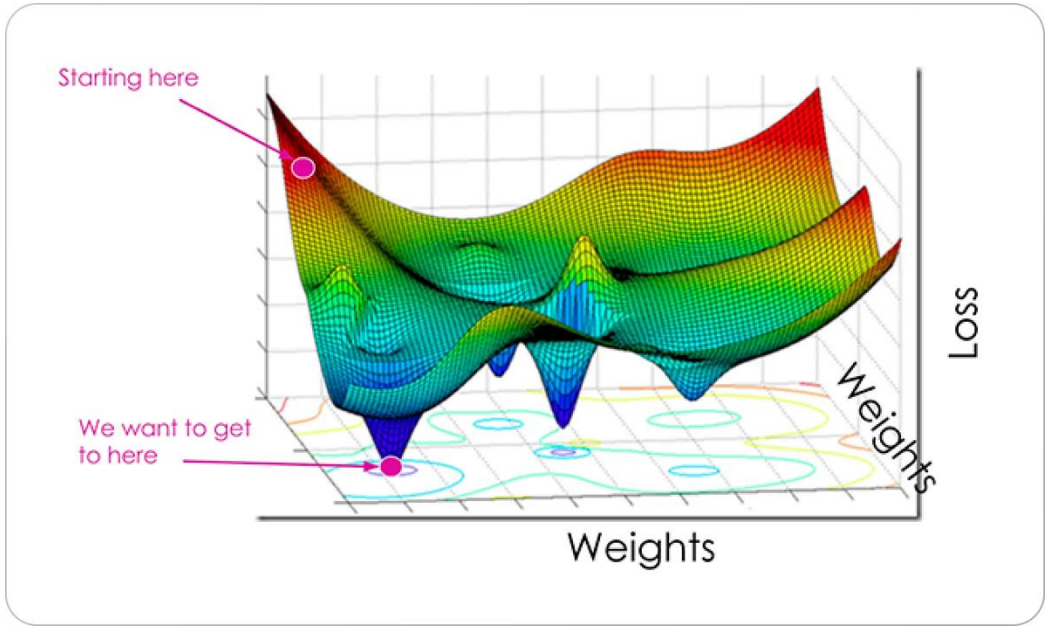


https://benmoseley.blog/my-research/so-what-is-a-physics-informed-neural-network/

Learning in neural networks

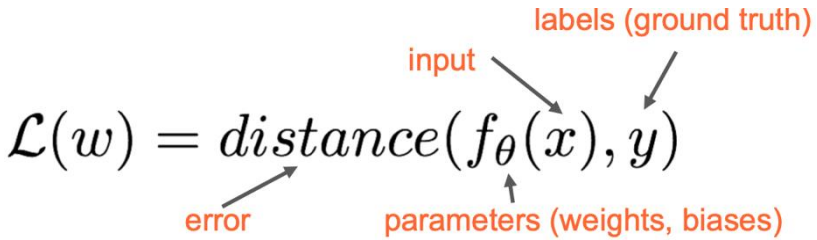
- The process of finding the combination of weights and biases that minimize the error on the training set – optimization problema
- Generalization: a well-trained model should perform roughly equal on seen vs unseed data





Loss function

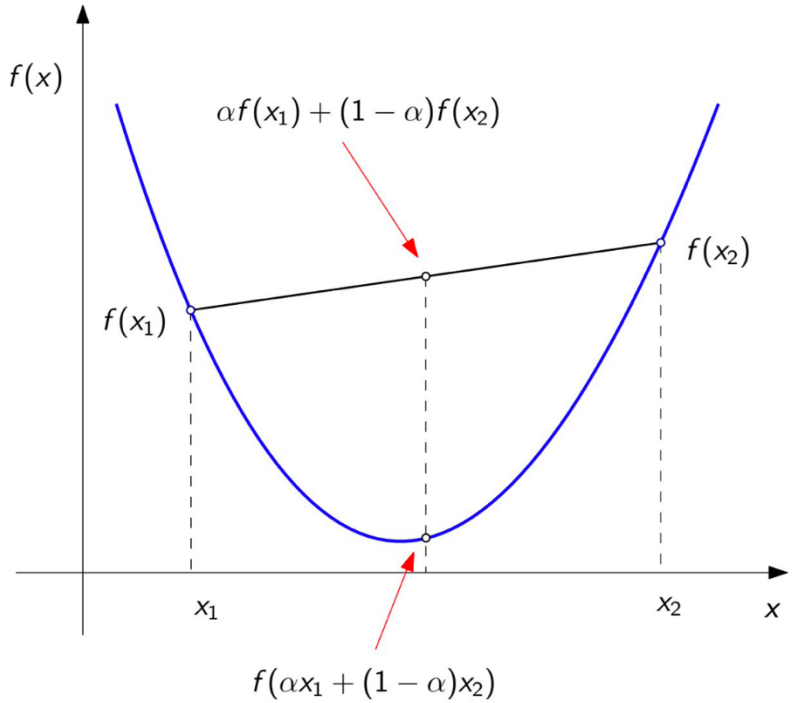
- The loss function gives us a distance (error) between our predictions and the target values



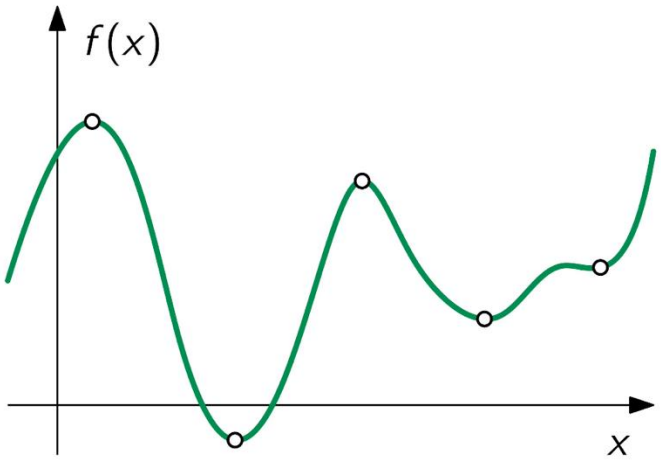
Task	Error type	Loss function	Note
Regression	Mean-squared error	$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$	Easy to learn but sensitive to outliers (MSE, L2 loss)
	Mean absolute error	$\frac{1}{n}\sum_{i=1}^{n}\lvert y_i - \hat{y}_i\rvert$	Robust to outliers but not differentiable (MAE, L1 loss)
Classification	Cross entropy = Log loss	$-\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] =$	Quantify the difference between two probability

Loss function

- Convex functions have a single minimum
- Most loss functions of the problems that we tackle with deep learning are high-dimensional non-convex functions with many local minima



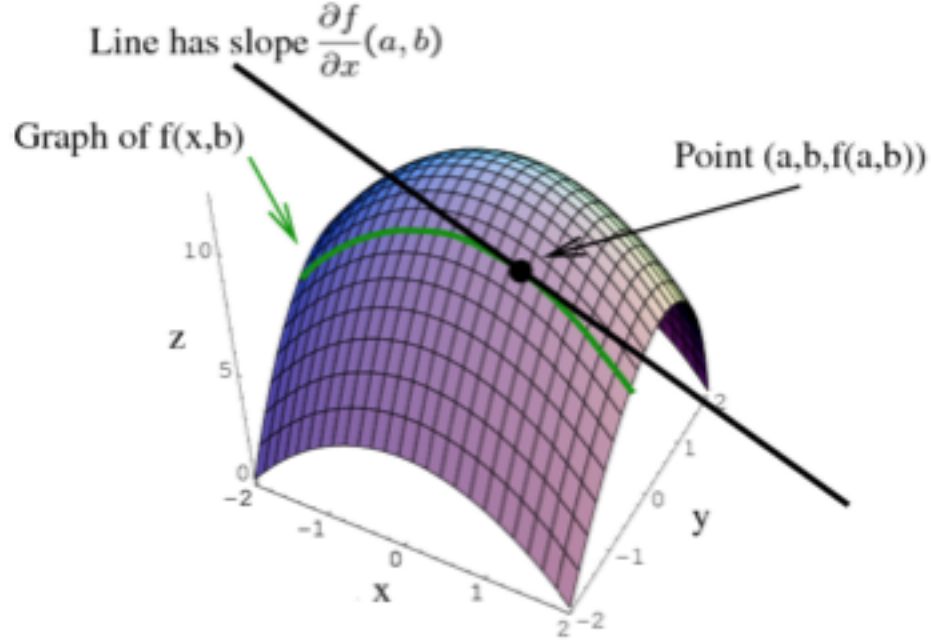
convex



non-convex

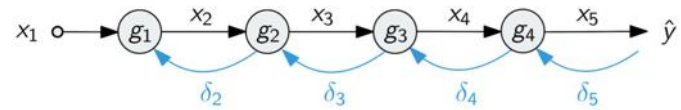
Partial derivatives

- In a multivariate function, the partial derivatives tell us how the output of our function changes with respect to each one of the variables in the input set
- When we derivate with respect to one of the variables, we treat the rest of the variables as constants



Training a neural network

- **Goal**: Find the set $\{w, b\}$ that minimizes an objective function (loss function) over all the samples in the training set
- Optimization algorithm: **gradient descent** $\rightarrow \theta_i := \theta_i - \alpha \frac{\partial \mathcal{L}}{\partial \theta_i}$
- The gradient of the loss with respect to the parameters of the model is computed with **backpropagation**



Decompose into steps again. Let $\delta_k = \frac{\partial \hat{y}}{\partial x_k}$. **Backpropagation**:

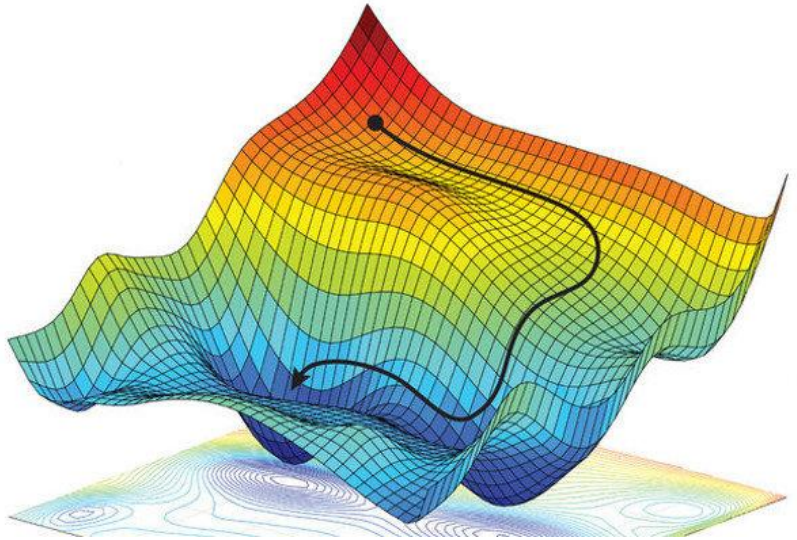
$$\delta_5 = \frac{\partial \hat{y}}{\partial x_5} = 1$$

$$\delta_4 = \frac{\partial \hat{y}}{\partial x_4} = \frac{\partial \hat{y}}{\partial x_5}\frac{\partial x_5}{\partial x_4} = \delta_5 g_4'(x_4)$$

$$\delta_3 = \frac{\partial \hat{y}}{\partial x_3} = \frac{\partial \hat{y}}{\partial x_4}\frac{\partial x_4}{\partial x_3} = \delta_4 g_3'(x_3)$$

$$\delta_2 = \frac{\partial \hat{y}}{\partial x_2} = \frac{\partial \hat{y}}{\partial x_3}\frac{\partial x_3}{\partial x_2} = \delta_3 g_2'(x_2)$$

$$\delta_1 = \frac{\partial \hat{y}}{\partial x_1} = \frac{\partial \hat{y}}{\partial x_2}\frac{\partial x_2}{\partial x_1} = \delta_2 g_1'(x_1)$$



The chain rule

Easily differentiate compositions of functions.

How does a variation ("difference") on x affect y ?

$$y = f(z)$$
$$z = g(x)$$

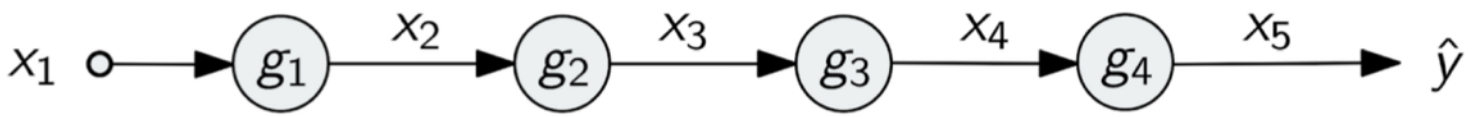
It will depend on the how y is affected by a variation in z...

$$\frac{dy}{dx} = \frac{dy}{dz}\frac{dz}{dx}$$

...scaled (multiplied) on how z is affected by a variation in x.

The chain rule

$$\hat{y} = g_4(g_3(g_2(g_1(x_1))))$$

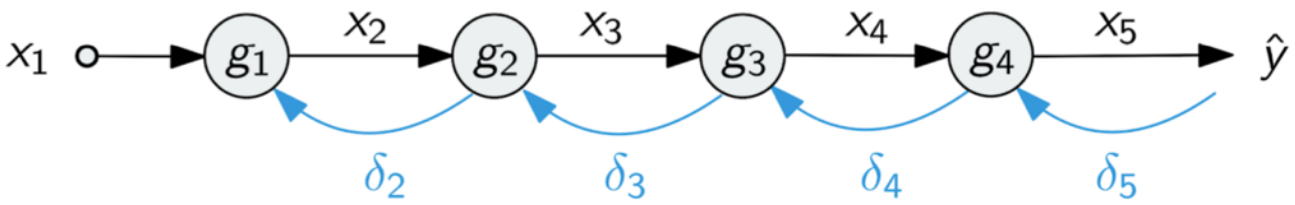


Decompose into steps (**forward propagation**):

$$\begin{aligned} x_2 &= g_1(x_1) \\ x_3 &= g_2(x_2) \\ x_4 &= g_3(x_3) \\ \hat{y} = x_5 &= g_4(x_4) \end{aligned}$$

The chain rule

$$\hat{y} = g_4(g_3(g_2(g_1(x_1))))$$

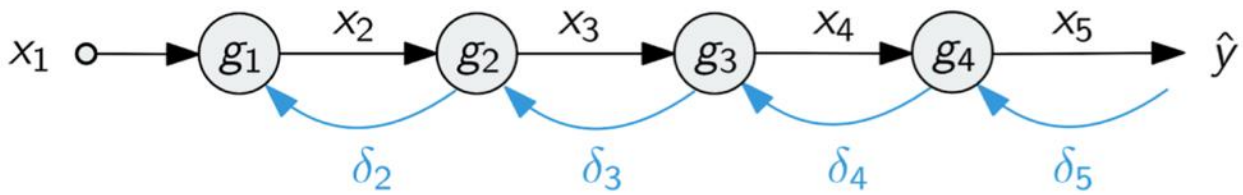


Want to find $\frac{\partial \hat{y}}{\partial x_1}$. Chain rule:

How does a variation ("difference") on the input affect the prediction ?

$$\frac{\partial \hat{y}}{\partial x_1} = \frac{\partial \hat{y}}{\partial x_4}\frac{\partial x_4}{\partial x_3}\frac{\partial x_3}{\partial x_2}\frac{\partial x_2}{\partial x_1}$$

The chain rule



Decompose into steps again. Let $\delta_k = \frac{\partial \hat{y}}{\partial x_k}$. **Backpropagation**:

$$\delta_5 = \frac{\partial \hat{y}}{\partial x_5} = 1$$

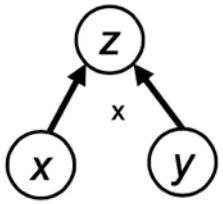
$$\delta_4 = \frac{\partial \hat{y}}{\partial x_4} = \frac{\partial \hat{y}}{\partial x_5}\frac{\partial x_5}{\partial x_4} = \delta_5 g_4'(x_4)$$

$$\delta_3 = \frac{\partial \hat{y}}{\partial x_3} = \frac{\partial \hat{y}}{\partial x_4}\frac{\partial x_4}{\partial x_3} = \delta_4 g_3'(x_3)$$

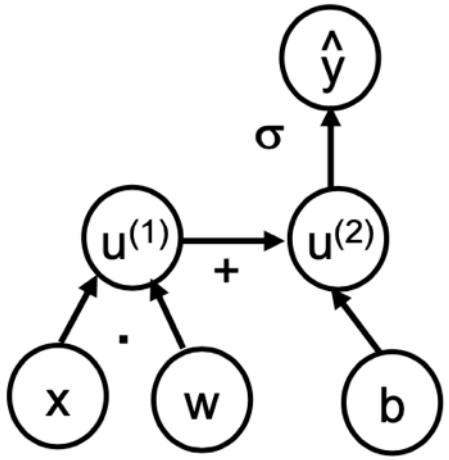
$$\delta_2 = \frac{\partial \hat{y}}{\partial x_2} = \frac{\partial \hat{y}}{\partial x_3}\frac{\partial x_3}{\partial x_2} = \delta_3 g_2'(x_2)$$

$$\delta_1 = \frac{\partial \hat{y}}{\partial x_1} = \frac{\partial \hat{y}}{\partial x_2}\frac{\partial x_2}{\partial x_1} = \delta_2 g_1'(x_1)$$

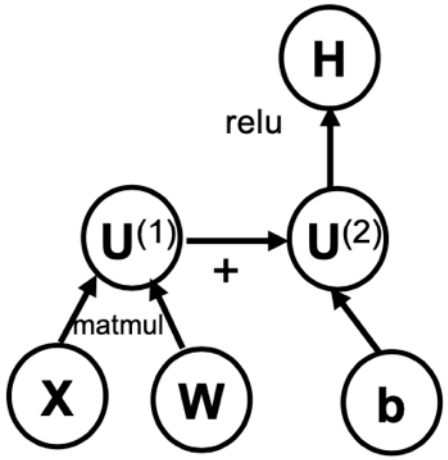
Computational graphs



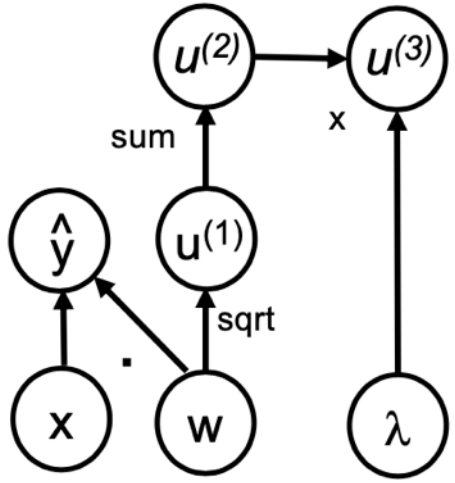
$z = xy$



$\hat{y}=\sigma(\mathrm{x}^T\mathrm{w}+\mathrm{b})$



$\boldsymbol{H}=\max\{0, \boldsymbol{XW}+\boldsymbol{b}\}$



$\hat{y}=\mathrm{x}^T\mathrm{w}$

$\lambda \sum_i w_i^2$

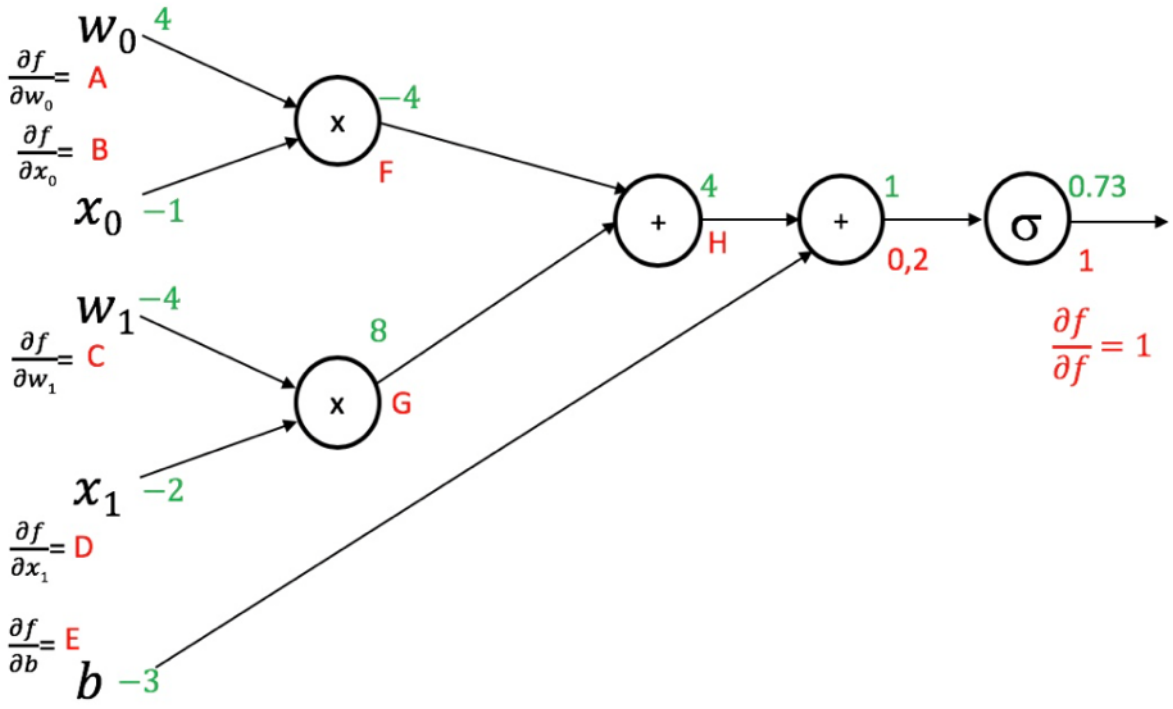
Solving a computational graph

Find the values (from A to H) of the following computational graph:

$$f(x, y, z) = \sigma(w_0 x_0 + w_1 x_1 + b)$$

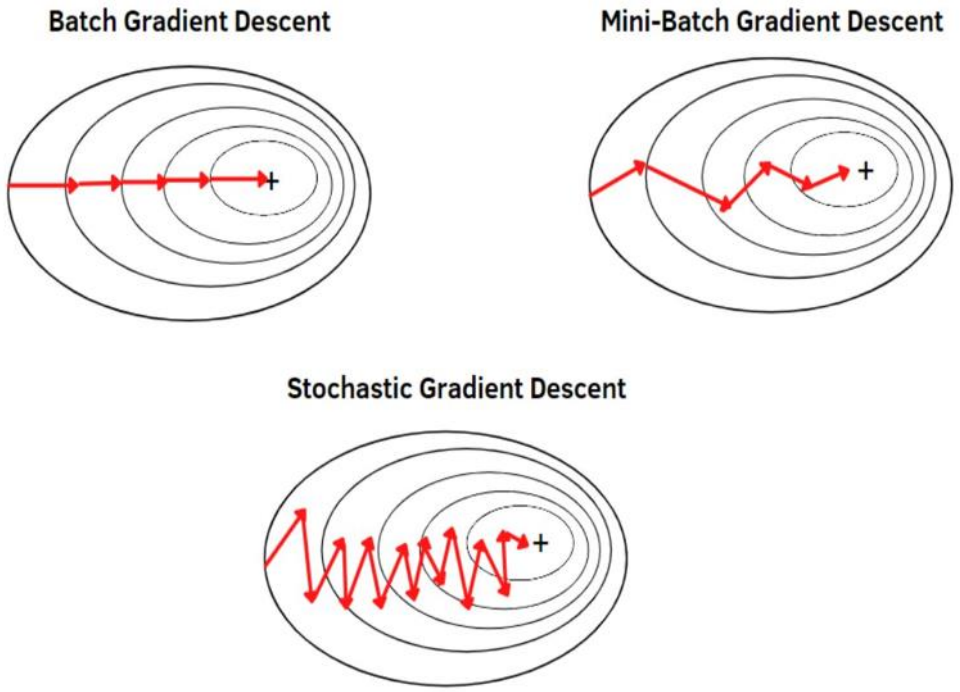
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

$$\frac{d\sigma(x)}{dx} = (1 - \sigma(x))(\sigma(x))$$



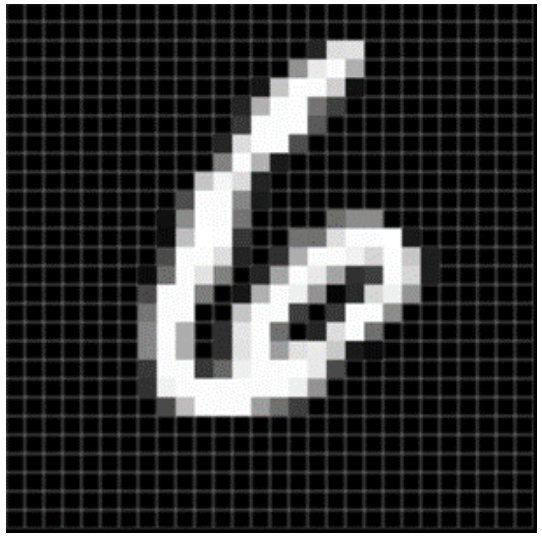
Stochastic Gradient Descent

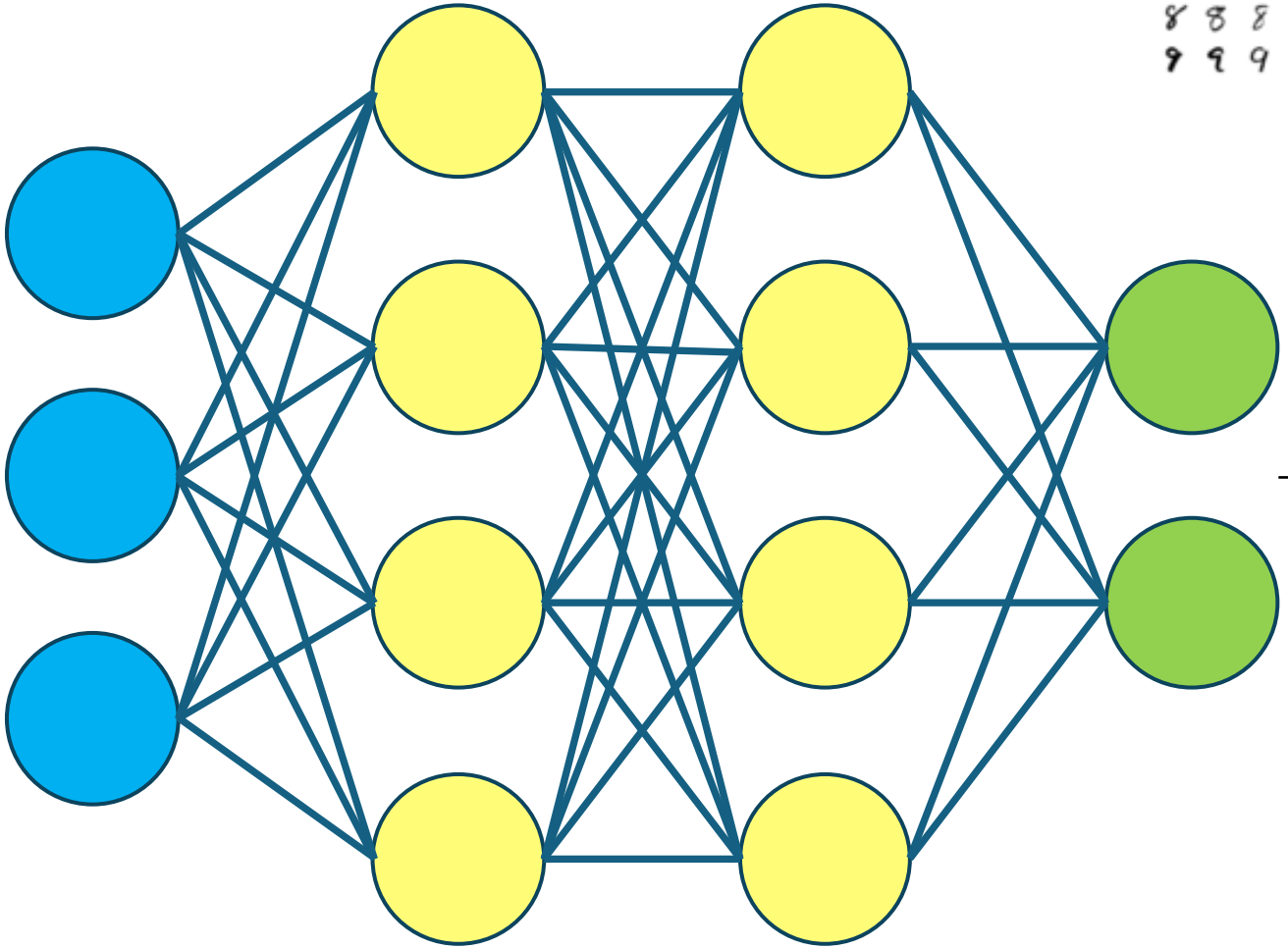
- Stochastic gradient descent: We compute the gradient with one sample at a time
- Batch gradient descent: we compute the gradient for the entire training set and then take a step towards the minumum
- **Mini-batch gradient descent**: we select a subset of samples (batch) and compute the gradient for that given subset



Training a Neural Network

Input image and perform feed-forward operation



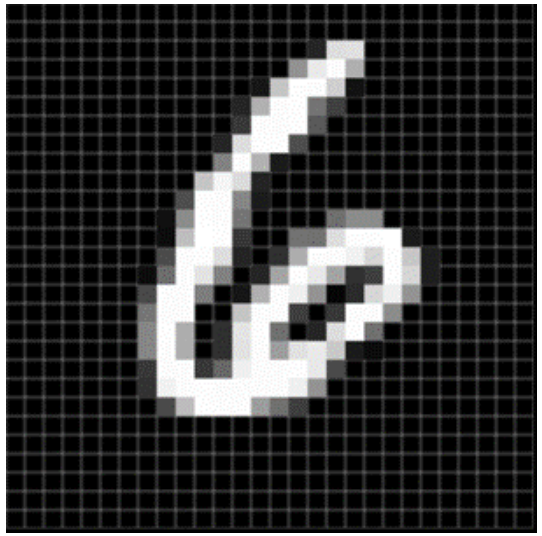


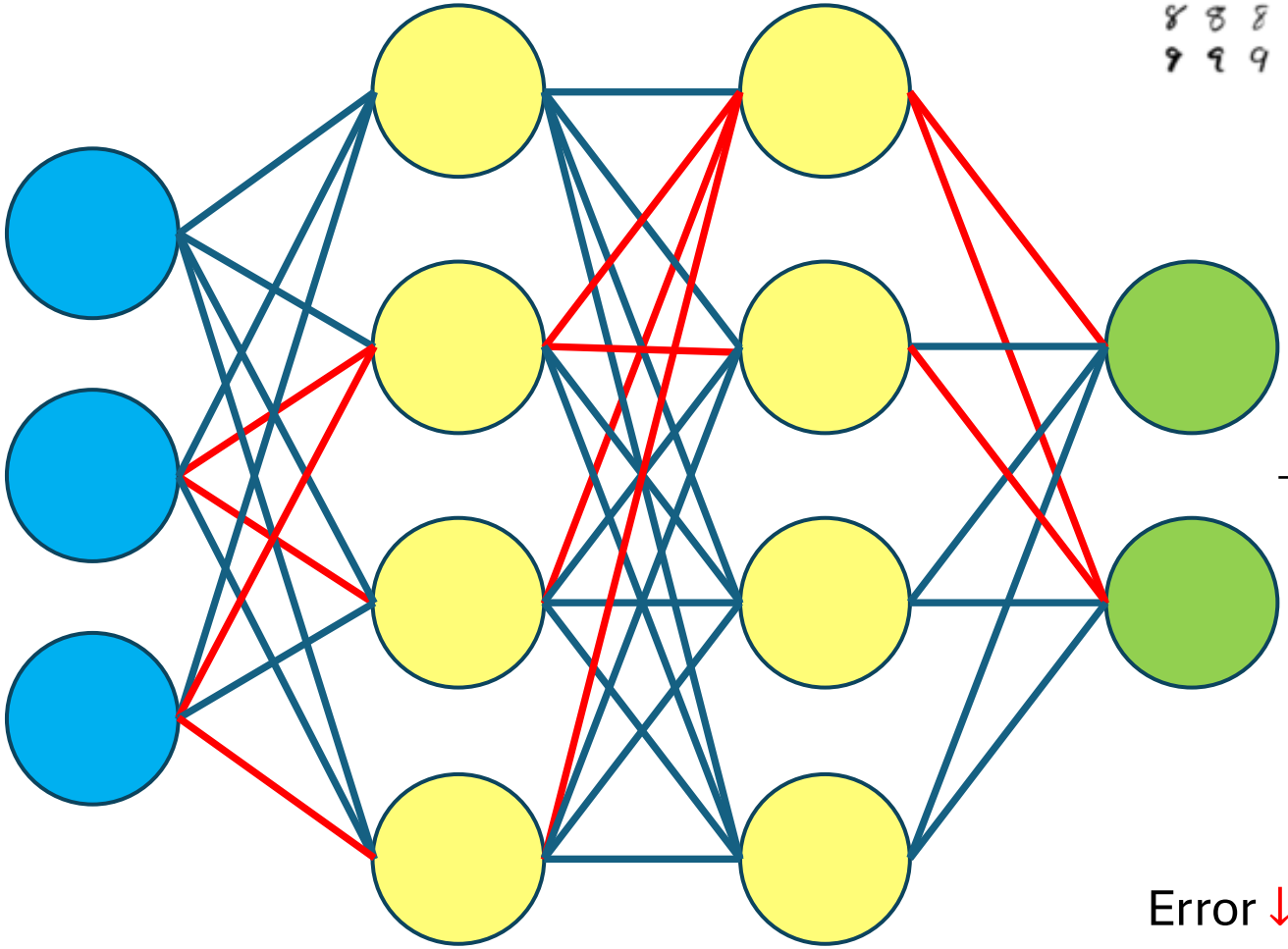
$$\begin{bmatrix} 0.1 \\ 0.02 \\ 0.05 \\ 0.03 \\ 0.03 \\ 0.01 \\ 0.7 \\ 0.01 \\ 0.03 \\ 0.02 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Output Label

Training a Neural Network

Update weights





$$\begin{bmatrix} 0.1 \\ 0.02 \\ 0.04 \\ 0.02 \\ 0.02 \\ 0.01 \\ 0.74 \\ 0.01 \\ 0.02 \\ 0.02 \end{bmatrix} \quad \text{error} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

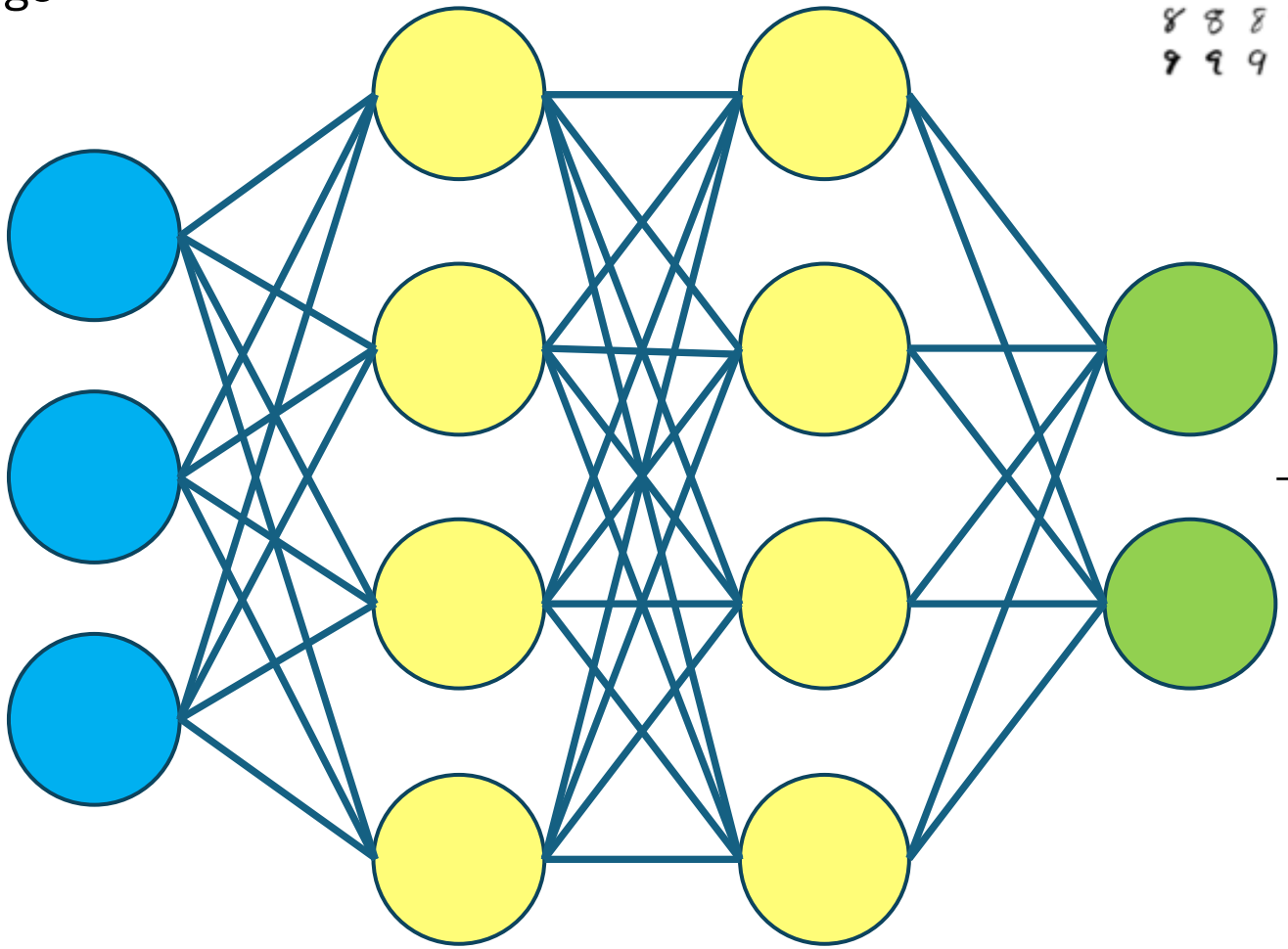
Error ↓

Output

Label

Training a Neural Network

Repeat with new image

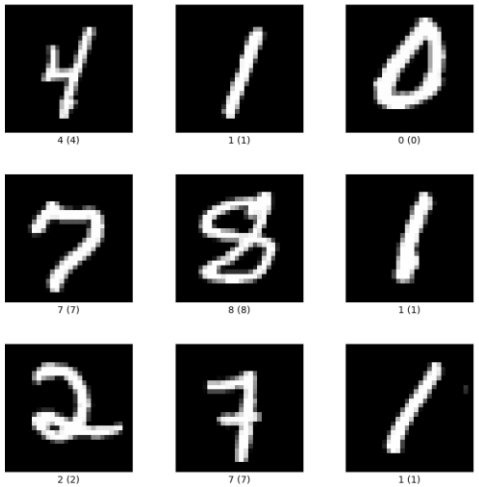


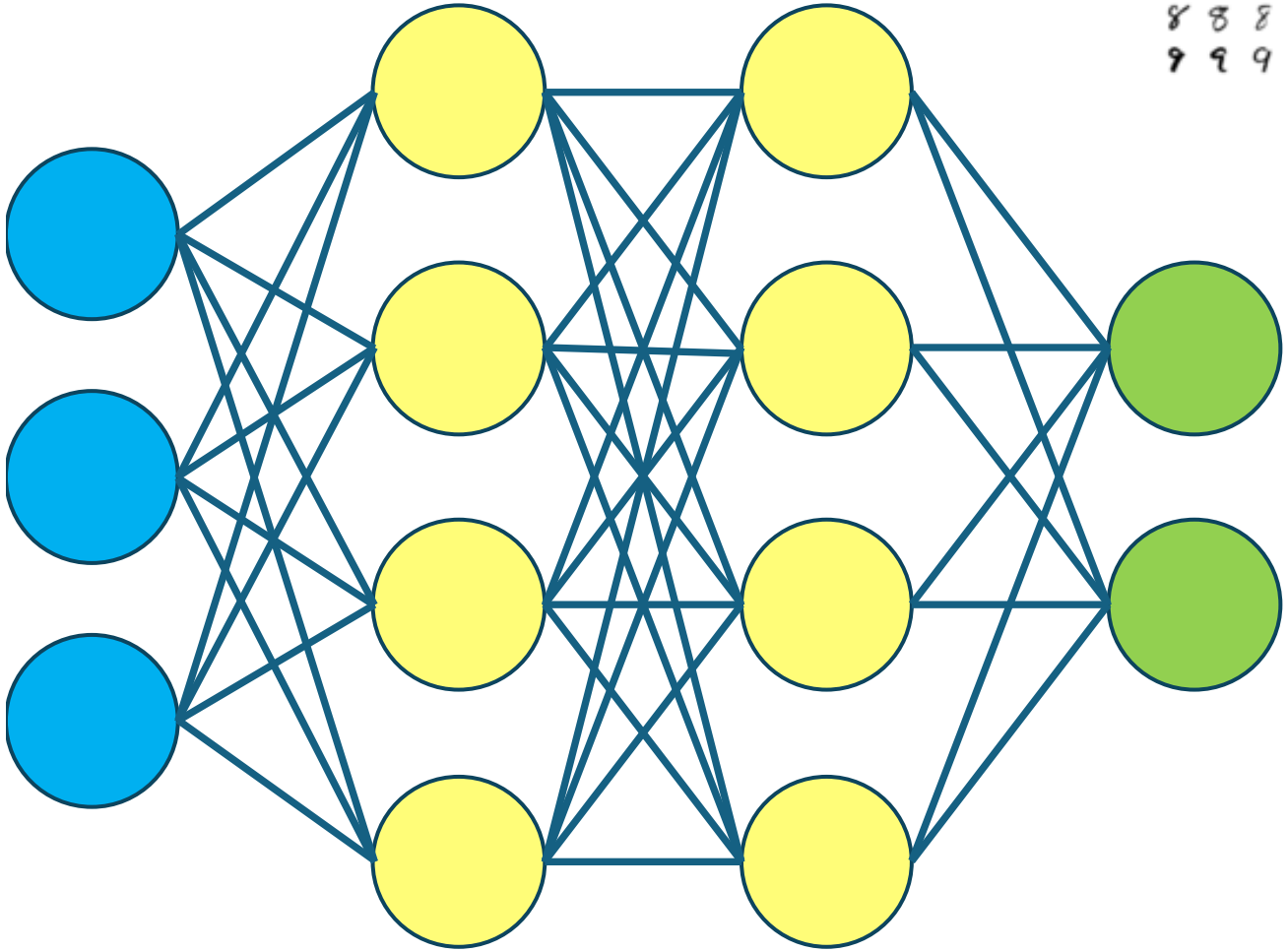
$$\begin{bmatrix} 0.1 \\ 0.03 \\ 0.03 \\ 0.5 \\ 0.02 \\ 0.07 \\ 0.18 \\ 0.02 \\ 0.01 \\ 0.02 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Output Label

Training a Neural Network

Batch





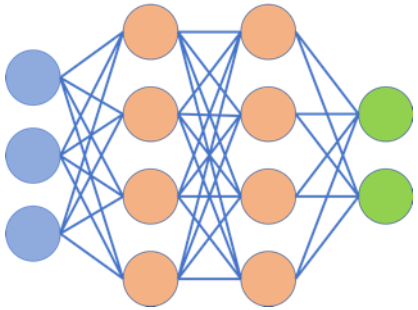
$$\begin{pmatrix} 0.1 & 0.2 & \dots \\ 0.3 & 0.05 & \dots \\ \dots & \dots & \dots \end{pmatrix}$$

Outputs

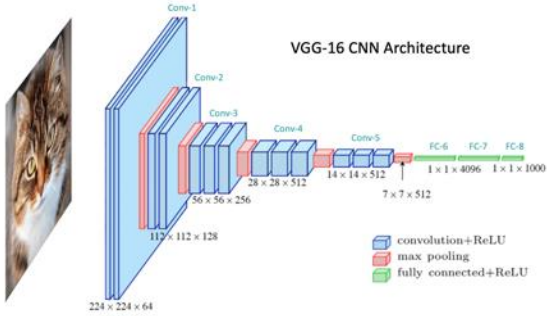
$$\begin{pmatrix} 0 & 0 & \dots \\ 0 & 1 & \dots \\ \dots & \dots & \dots \end{pmatrix}$$

Labels

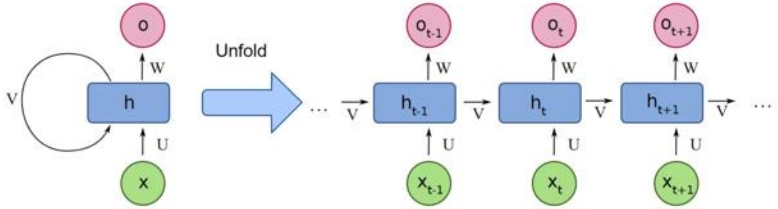
Neural Network architectures



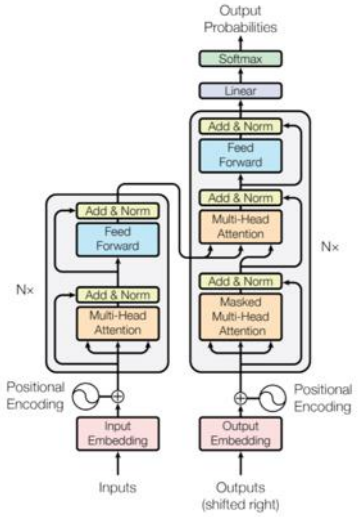
fully connected/linear/dense layers or MLP
→
linear map + non-linear activation functions



convolutional layers → CNNs
computer vision, signal processing, etc.

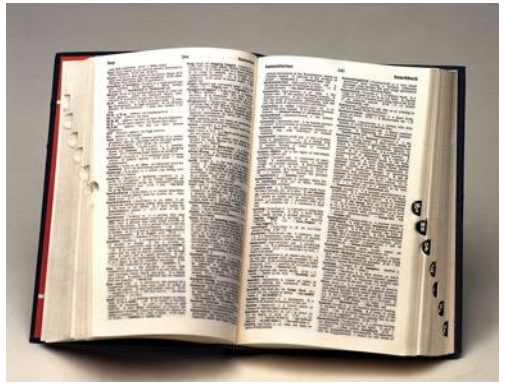


recurrent layers → RNNs, LSTMs, GRUs
time series, natural language processing



Transformers
Seq to seq models
Multimodal models
Large language models
Generative models
Etc.

Task: next word prediction



A
Aardvark
Abeam
Abacus
Abandon
...

Vocab size

Embedding dimension

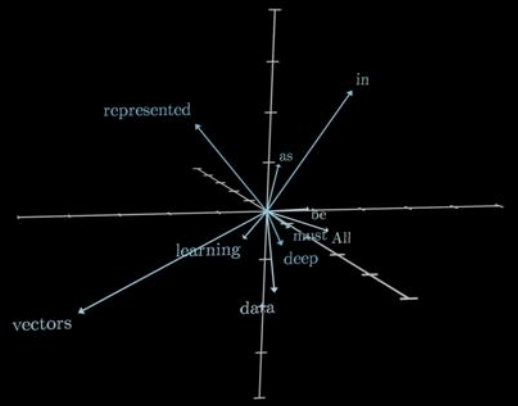
All words, ~ 50k

aah	aardvark	aardwolf	aargh	ab	aback	abacterial	abacus	abalone	abandon	…	zygoid	zygomatic	zygomorphic	zygosis	zygote	zygotic	zyme	zymogen	zymosis	zzz
+1.0	+4.3	+2.0	+0.9	−1.5	+2.9	−1.2	+7.8	+9.2	−2.3	⋯	+0.6	+1.3	+8.4	−8.5	−8.2	−9.5	+6.6	+5.5	+7.3	+9.5
+5.9	−0.8	+5.6	−7.6	+2.8	−7.1	+8.8	+0.4	−1.7	−4.7	⋯	−0.9	+1.4	−9.5	+2.3	+2.2	+2.3	+8.8	+3.6	−2.8	−1.2
+3.9	−8.7	+3.3	+3.4	−5.7	−7.3	−3.7	−2.7	+1.4	−1.2	⋯	−7.9	−5.8	−6.7	+3.0	−4.9	−0.7	−5.1	−6.8	−7.7	+3.1
−7.2	−6.0	−2.6	+6.4	−8.0	+6.7	−8.0	+9.4	−0.6	+9.4	⋯	+4.7	−9.1	−4.3	−7.5	−4.0	−7.5	−3.6	−1.7	−8.6	+3.8
+1.3	−4.6	+0.5	−8.0	+1.5	+8.5	−3.6	+3.3	−7.3	+4.3	⋯	−6.3	+1.7	−9.5	+6.5	−9.8	+3.5	−4.6	+4.7	+9.2	−5.0
+1.5	+1.8	+1.4	−5.5	+9.0	−1.0	+6.9	+3.9	−4.0	+6.2	⋯	+7.5	+1.6	+7.6	+3.8	+4.5	+0.0	+9.0	+2.9	−1.5	+2.1
−9.5	−3.9	+3.2	−4.2	+2.3	−1.4	−7.2	−4.0	+1.4	+1.8	⋯	+3.0	+3.0	−1.4	+7.9	−2.6	−1.3	+7.8	+6.1	+4.0	−7.9
+8.3	+4.2	+9.9	−6.9	+7.3	−6.7	+2.3	−7.4	+6.9	+6.1	⋯	−1.8	−8.5	+3.9	−0.9	+4.4	+7.3	+9.4	+7.0	−9.7	−2.8
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
−3.7	−2.0	−5.7	−6.2	+8.8	+4.7	−0.2	−5.4	−4.9	−8.8	⋯	−3.7	+3.9	−2.4	−6.3	−9.4	−8.6	+3.6	−0.9	+0.7	+7.9

Embedding matrix

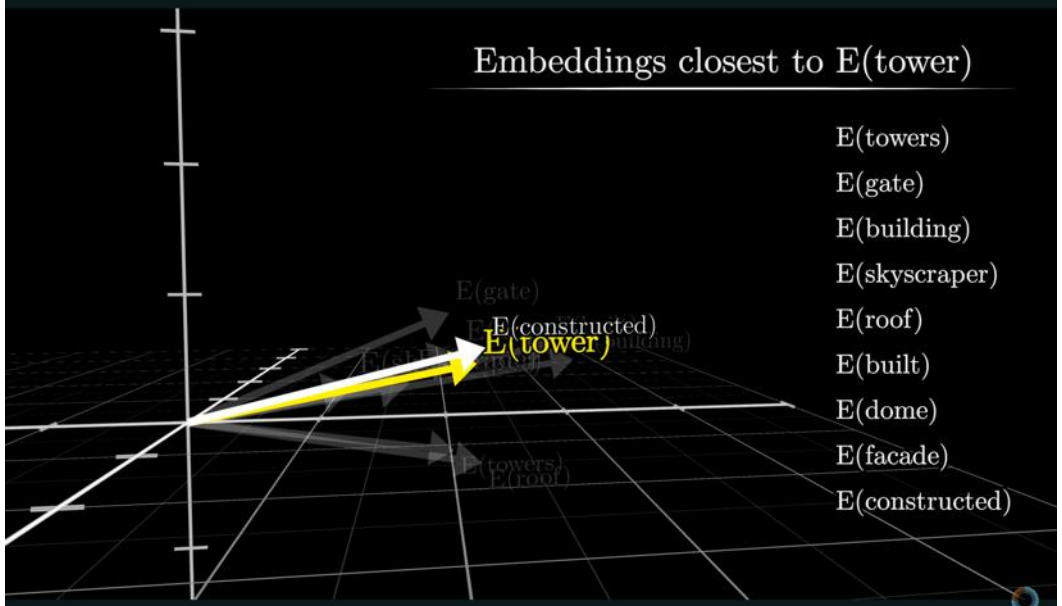
Words "Embedding" Vectors

All
data
in
deep
learning
must
be
represented
as
vectors

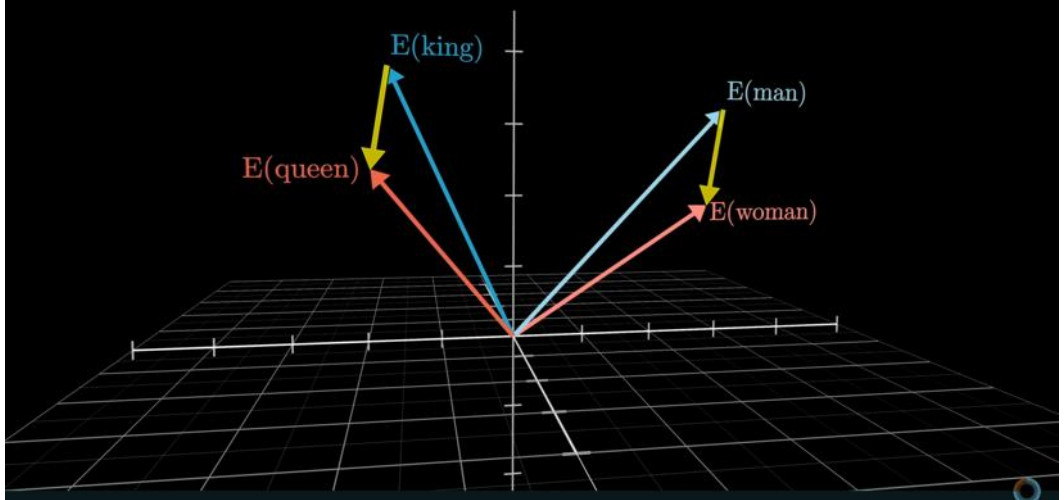


Embeddings closest to E(tower)

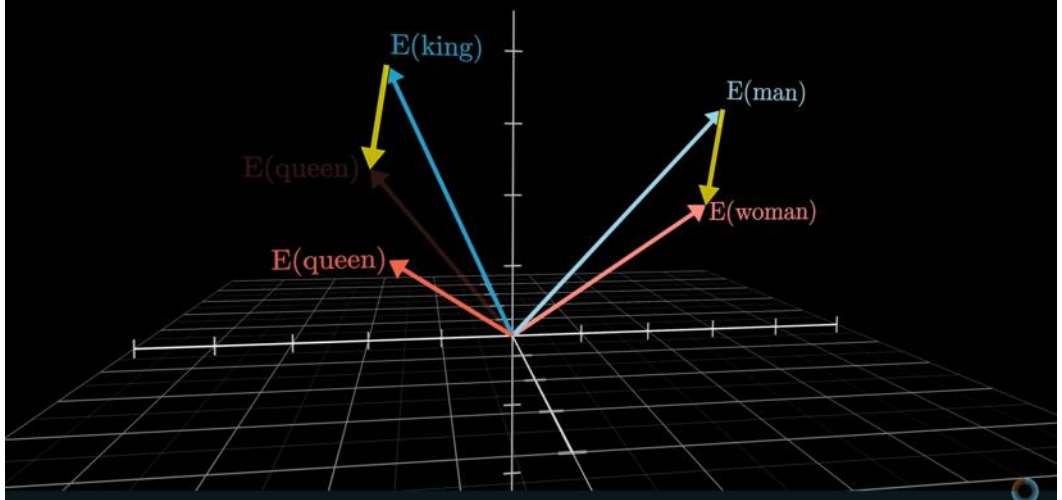
E(towers)
E(gate)
E(building)
E(skyscraper)
E(roof)
E(built)
E(dome)
E(facade)
E(constructed)



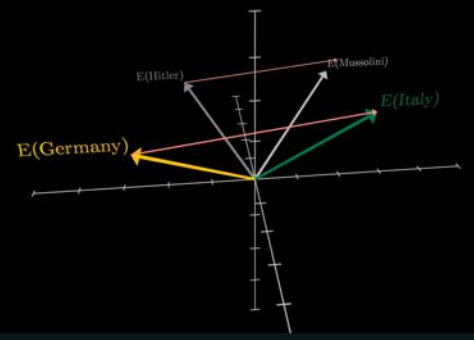
E(queen) ≈ E(king) + E(woman) - E(man)



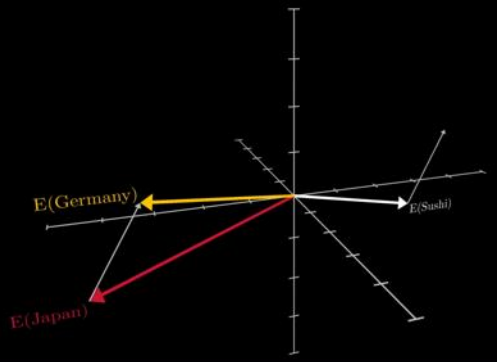
E(queen) ≈ E(king) + E(woman) - E(man)



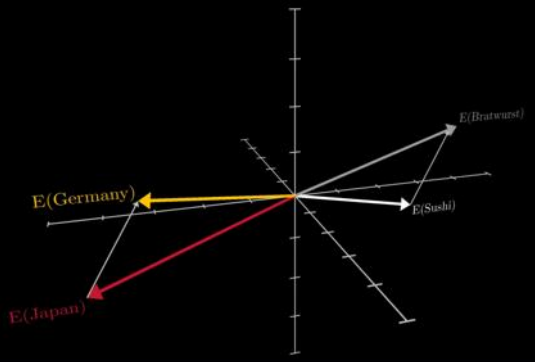
E(Hitler) + E(Italy) - E(Germany) ≈ E(Mussolini)



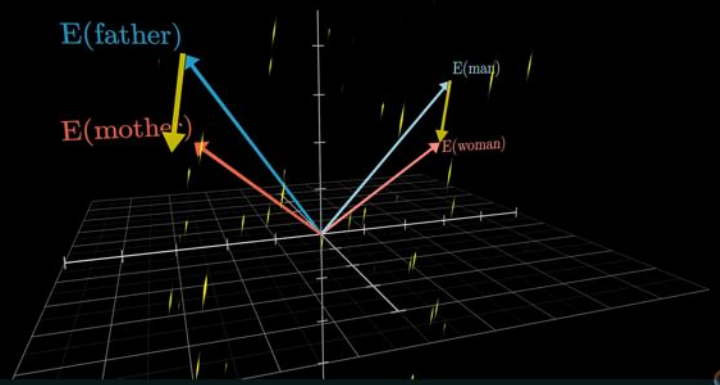
E(Sushi) + E(Germany) - E(Japan) ≈



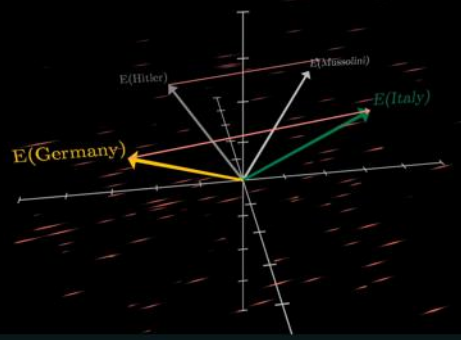
E(Sushi) + E(Germany) - E(Japan) ≈ E(Bratwurst)



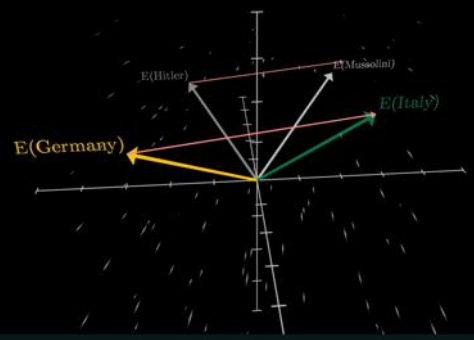
E(mother) - E(father) ≈ E(woman) - E(man)



E(Hitler) + E(Italy) - E(Germany) ≈ E(Mussolini)



E(Hitler) + E(Italy) - E(Germany) ≈ E(Mussolini)

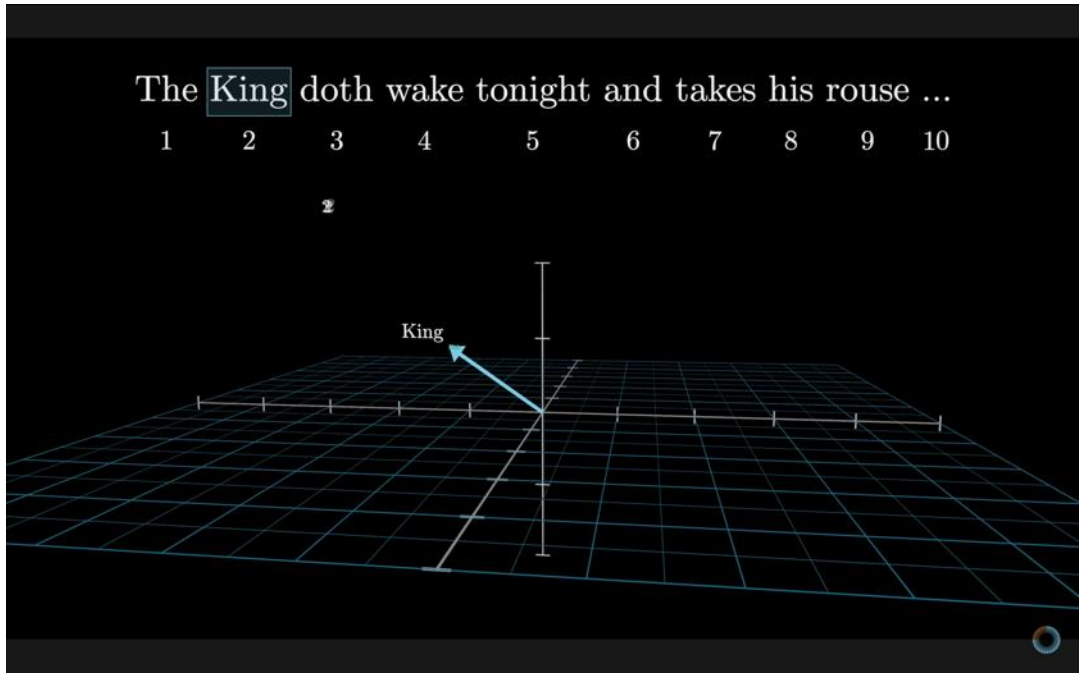


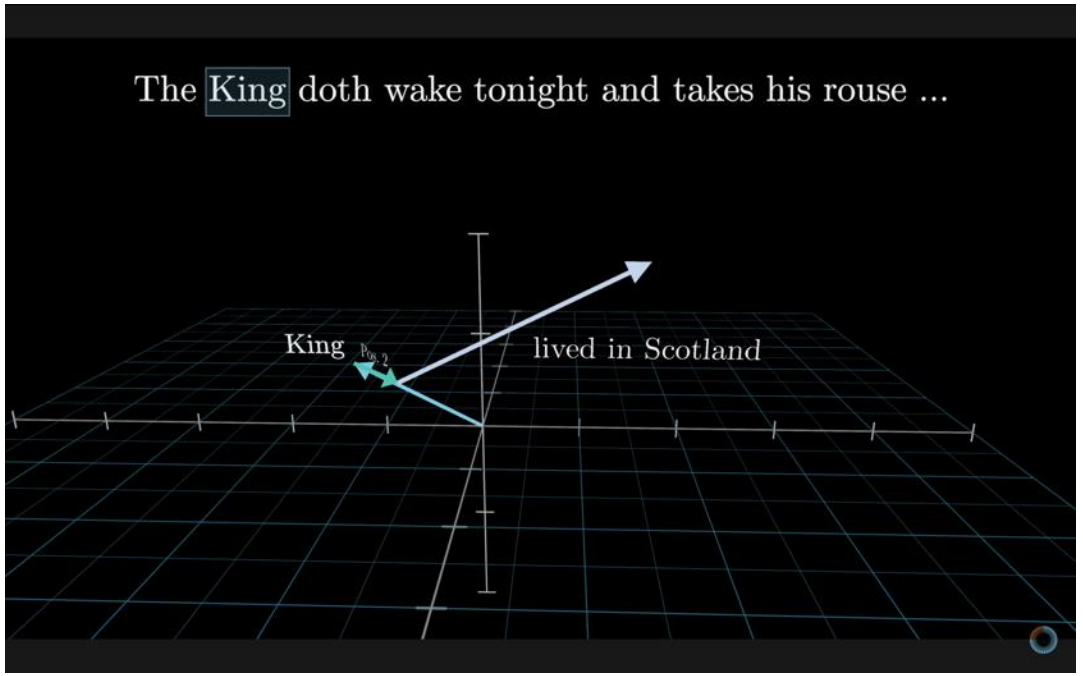
Task: next word prediction

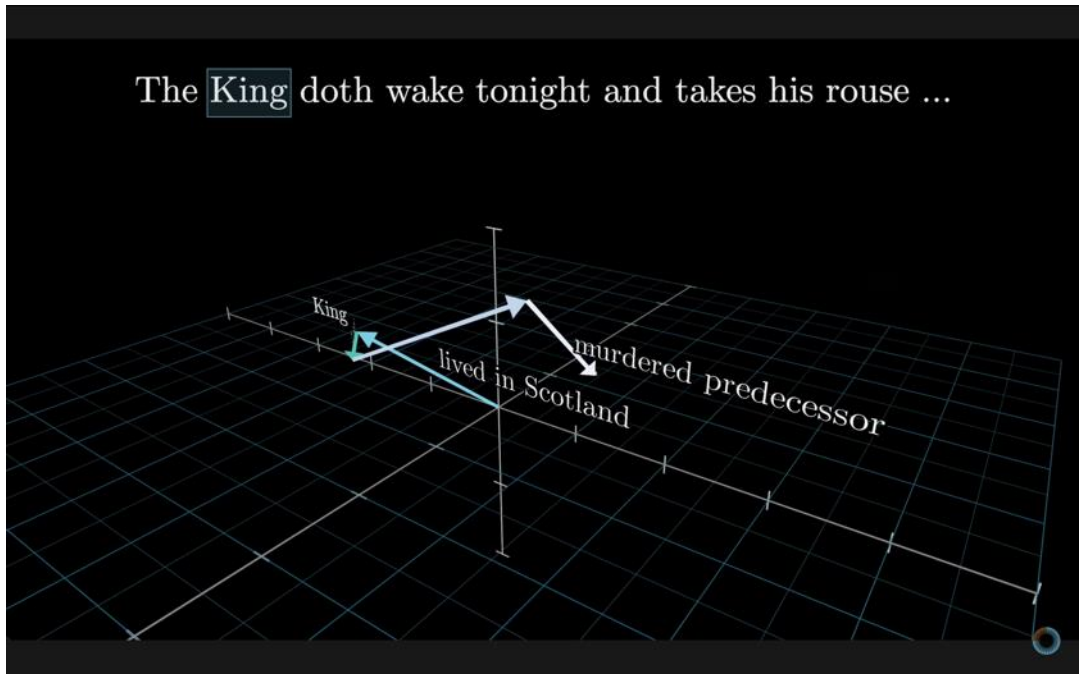
- We can use each word in a sentence to predict the following word; however, we would be missing the context that gives meaning to it
- Context: how the meaning of a word is affected by the meaning of the other words in a given text

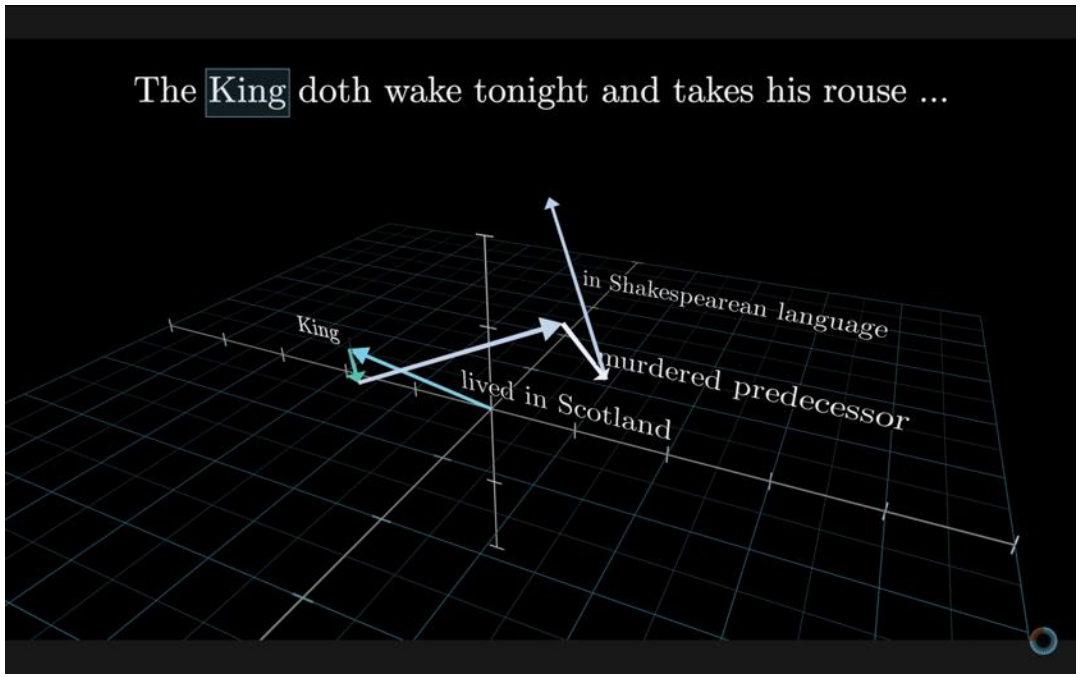
1. Machine learning model
2. Fashion model

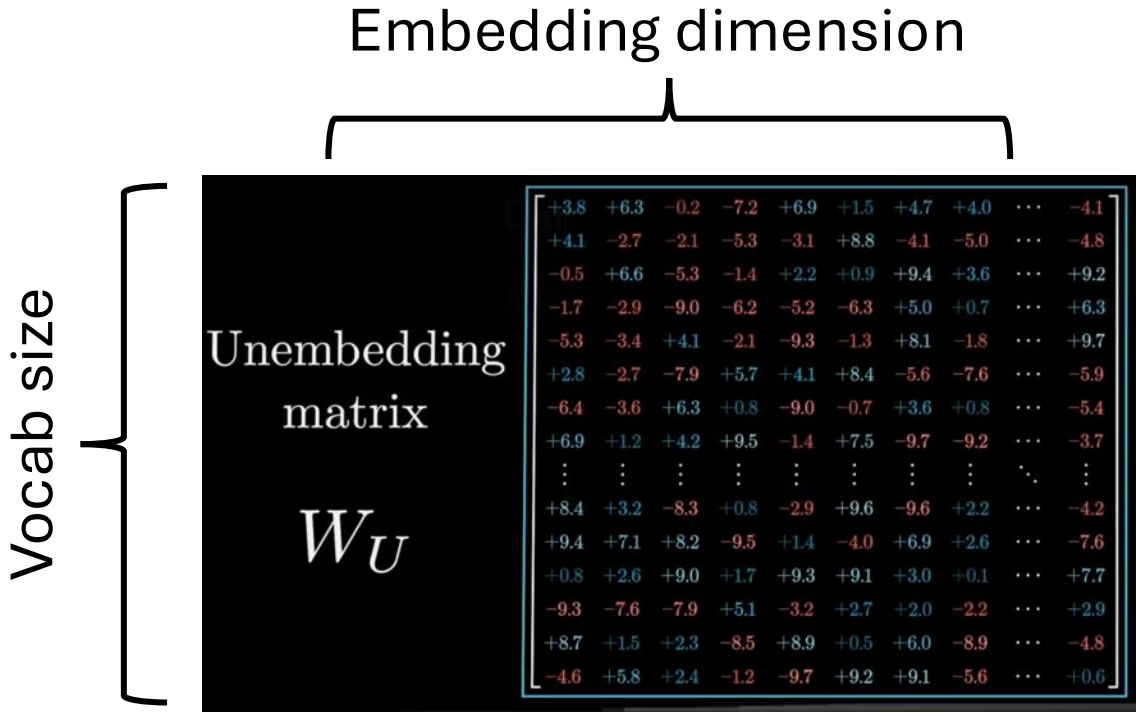
- Language model: from an initial embedding or snippet we want to obtain a single final embedding, which we can then run through an MLP and predict the probability of each word in the vocabulary appearing next

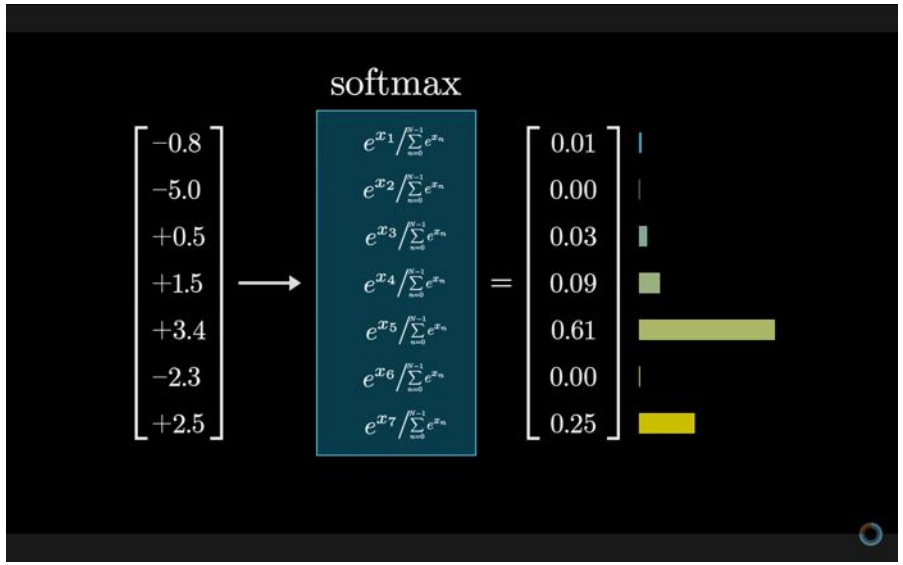

The King doth wake tonight and takes his rouse ...
1 2 3 4 5 6 7 8 9 10
King

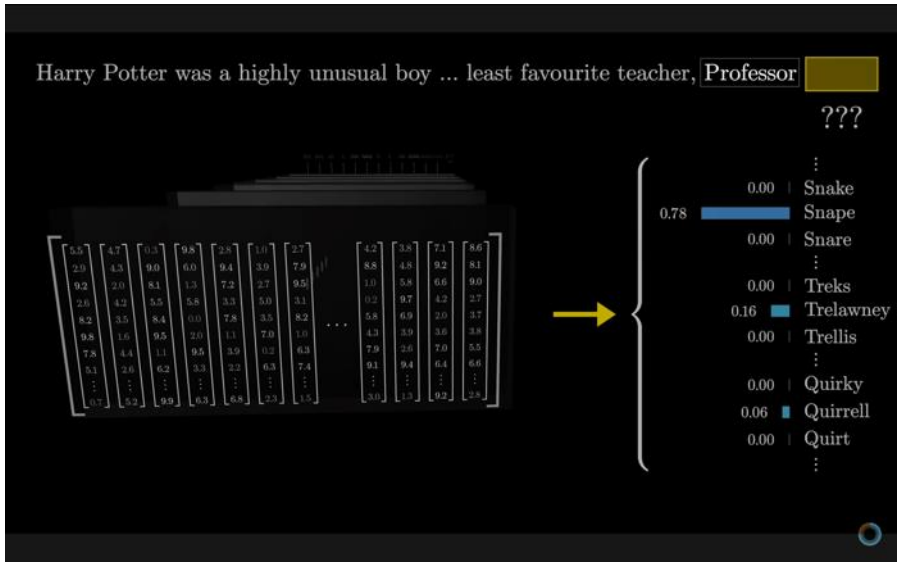

The King doth wake tonight and takes his rouse ...
King
lived in Scotland


The King doth wake tonight and takes his rouse ...
King
lived in Scotland
murdered predecessor


The King doth wake tonight and takes his rouse ...
in Shakespearean language
King
murdered predecessor
lived in Scotland

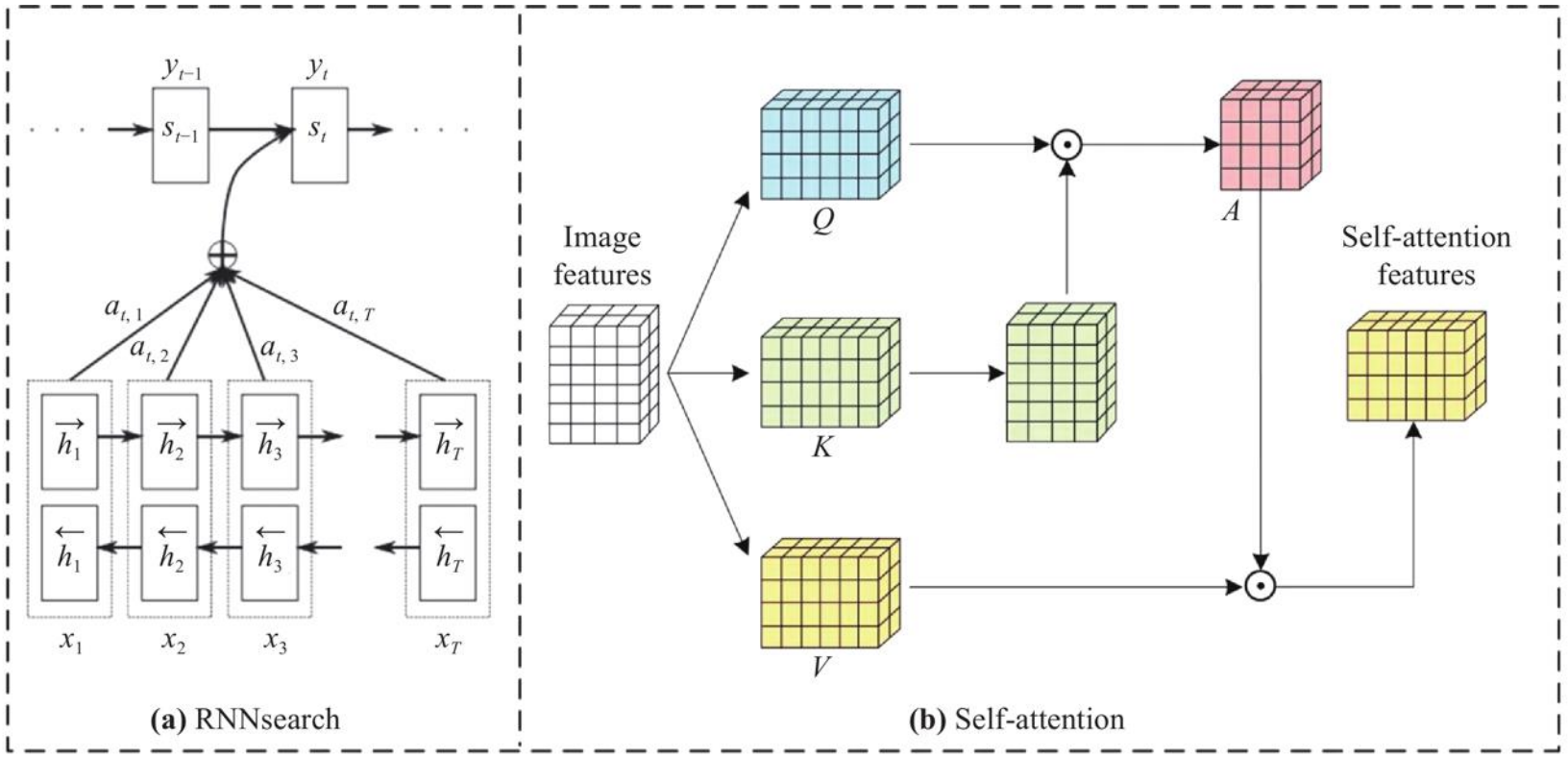

Embedding dimension
Vocab size
Unembedding matrix
W_U


softmax
−0.8
−5.0
+0.5
+1.5
+3.4
−2.3
+2.5
0.01
0.00
0.03
0.09
0.61
0.00
0.25

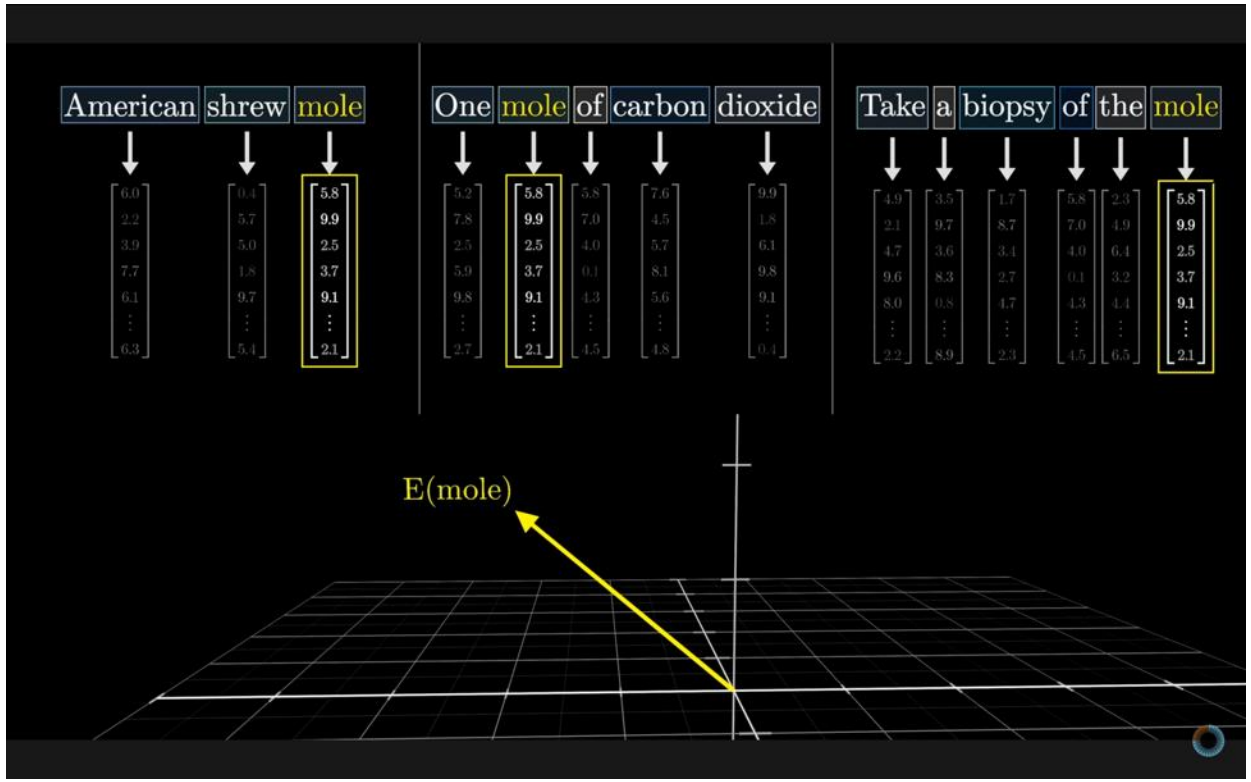

Harry Potter was a highly unusual boy ... least favourite teacher, Professor
???
0.00 Snake
0.78 Snape
0.00 Snare
0.00 Treks
0.16 Trelawney
0.00 Trellis
0.00 Quirky
0.06 Quirrell
0.00 Quirt

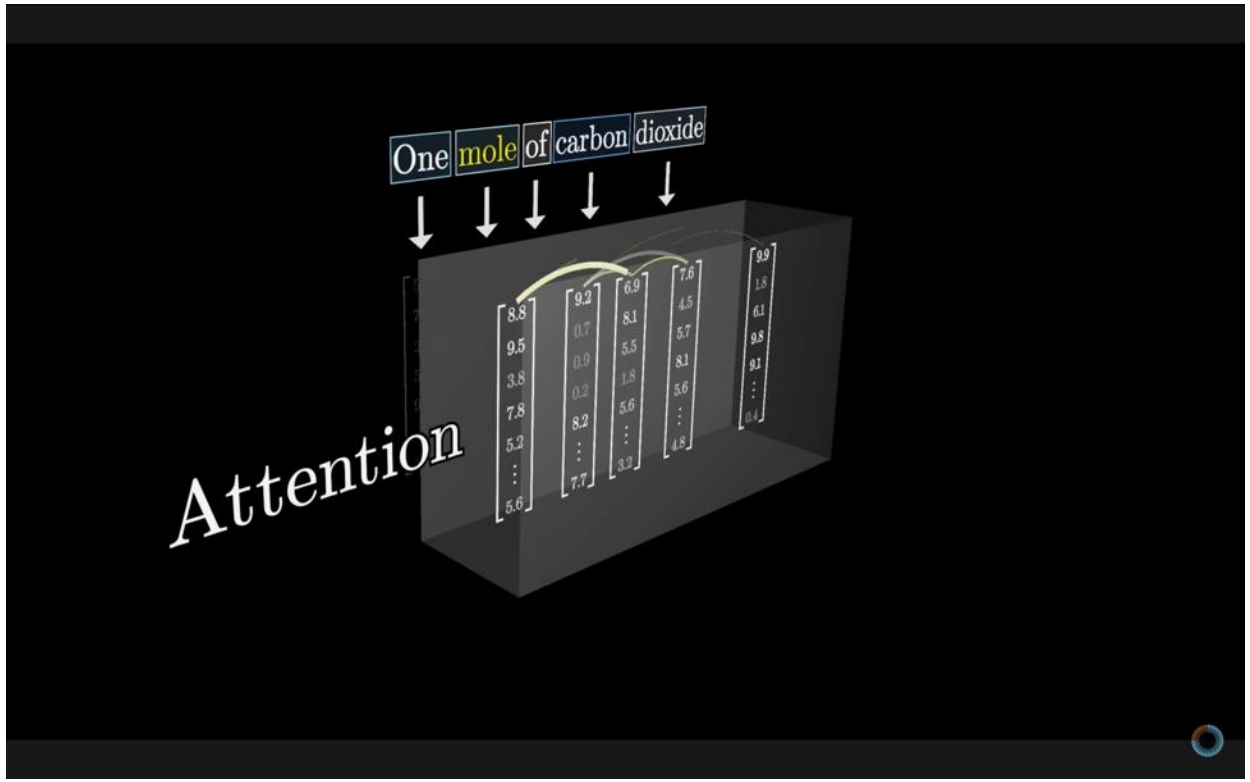
Attention mechanism

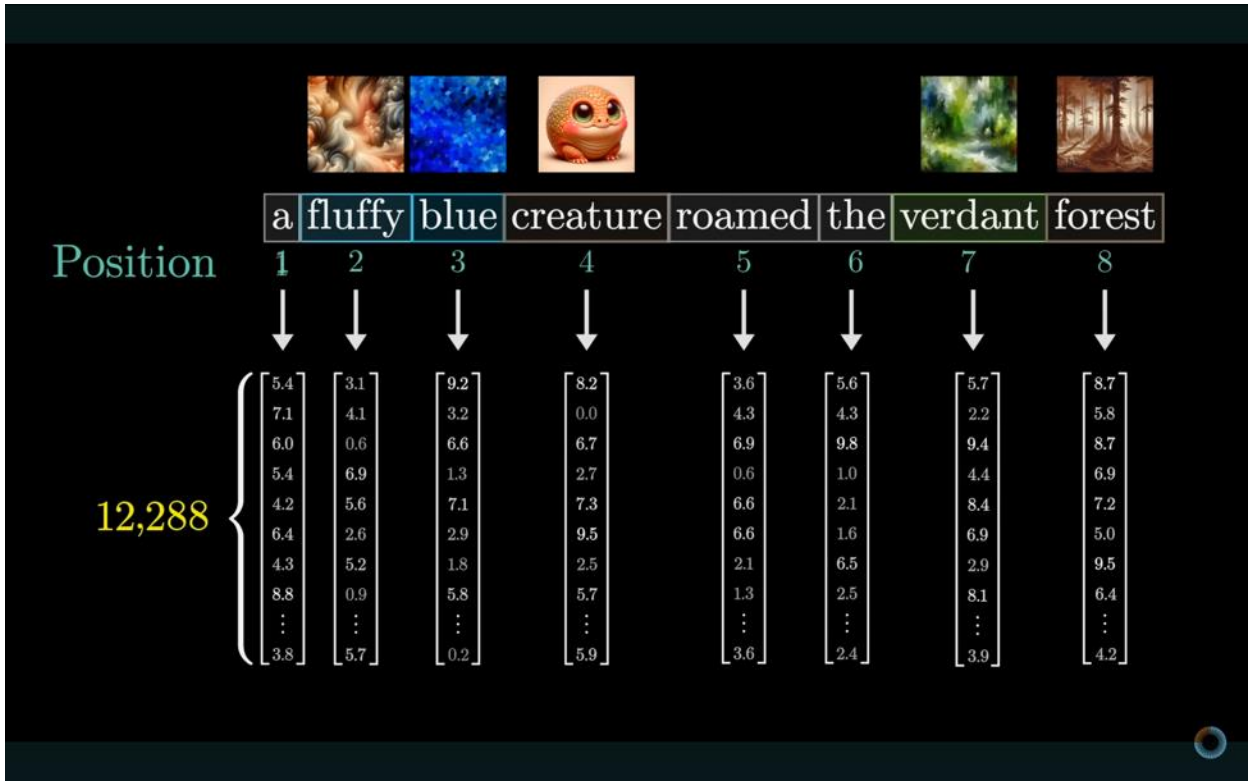
The teacher asked the class a question, and everyone turned to look at **her**

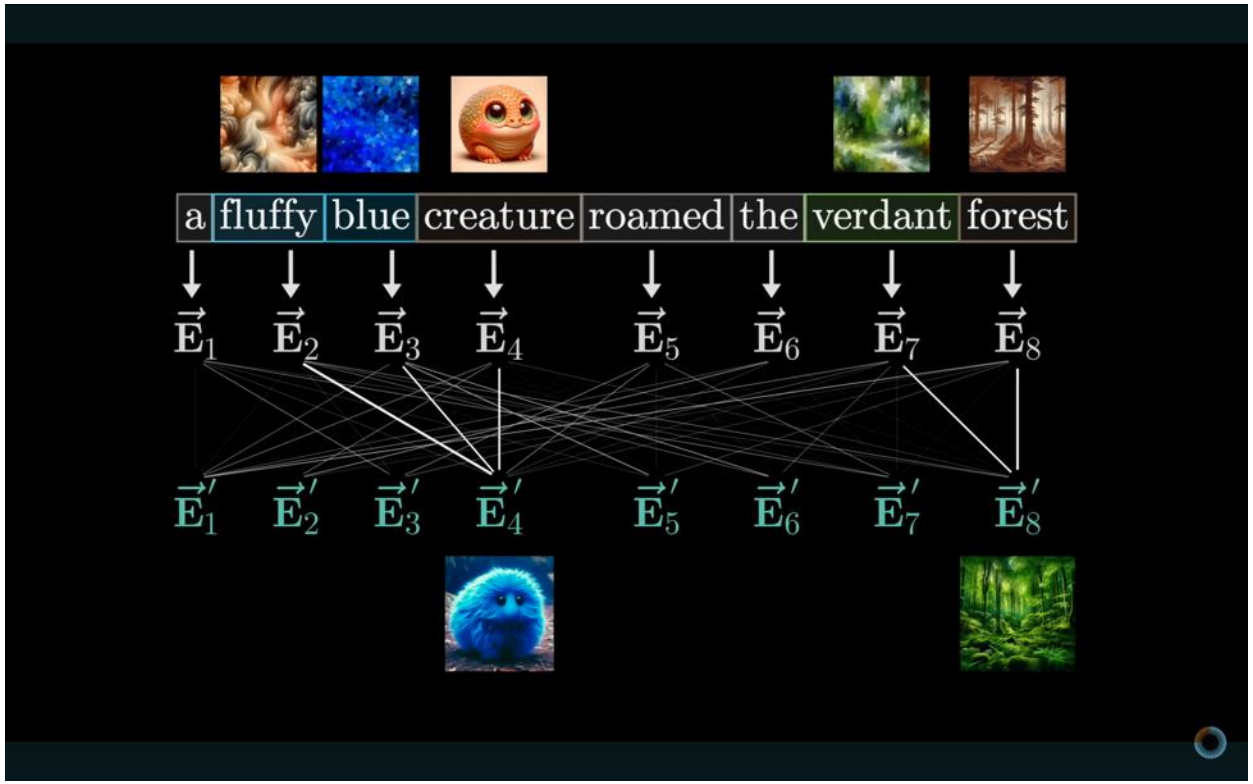


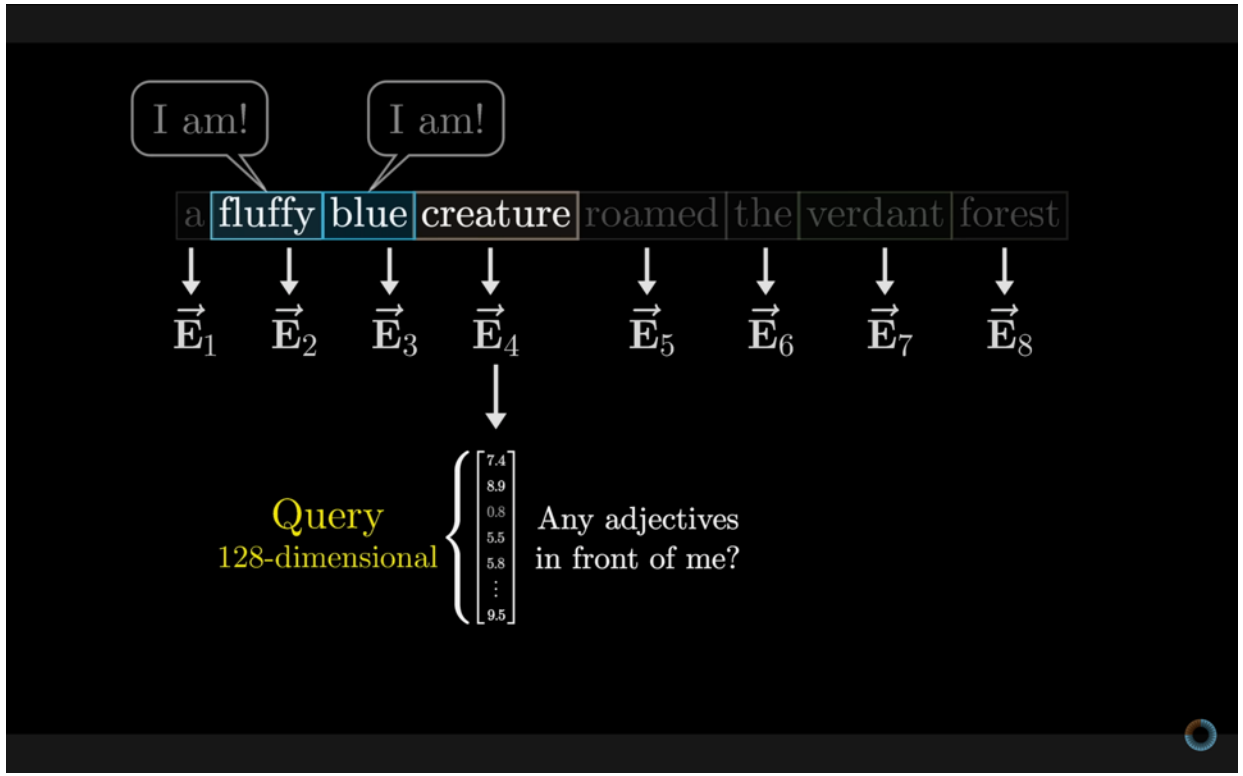
Li, X., Li, M., Yan, P., Li, G., Jiang, Y., Luo, H., & Yin, S. Deep Learning Attention Mechanism in Medical Image Analysis: Basics and Beyonds. *International Journal of Network Dynamics and Intelligence*. 2023, *2*(1), 93–116.

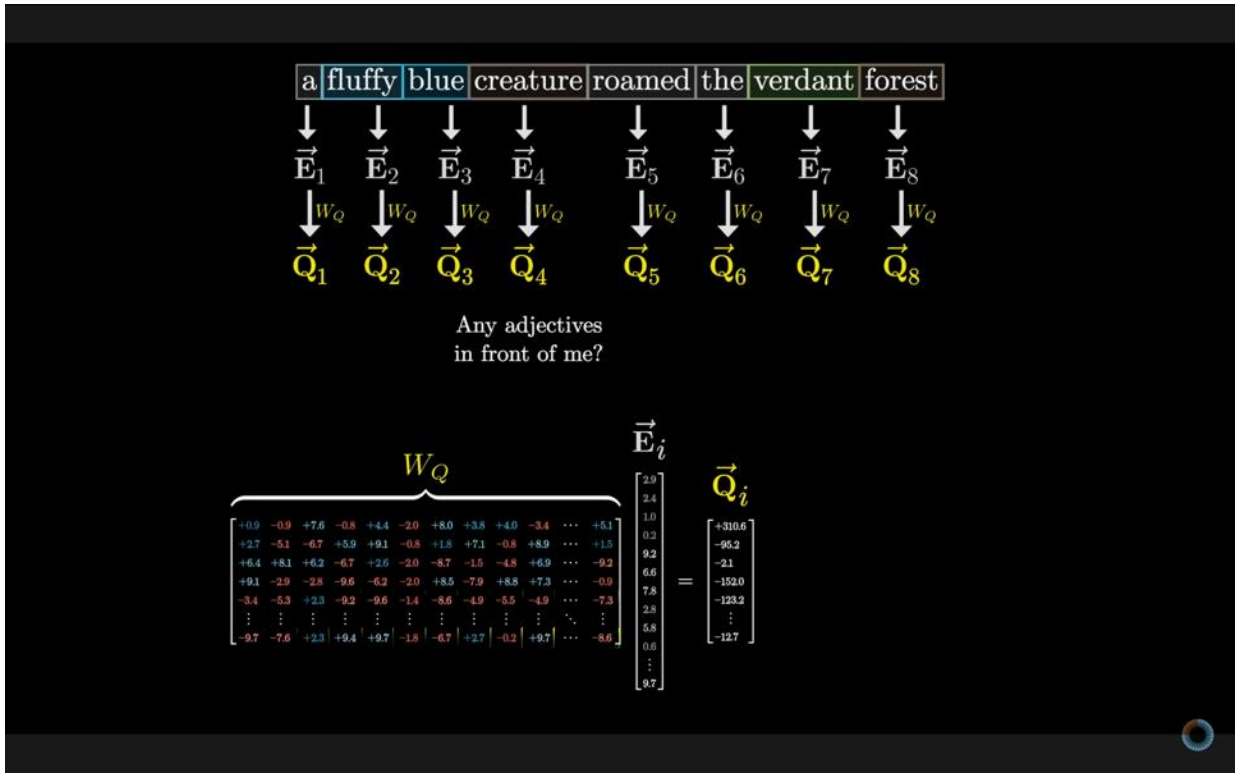

American shrew mole
One mole of carbon dioxide
Take a biopsy of the mole
E(mole)


One mole of carbon dioxide
Attention

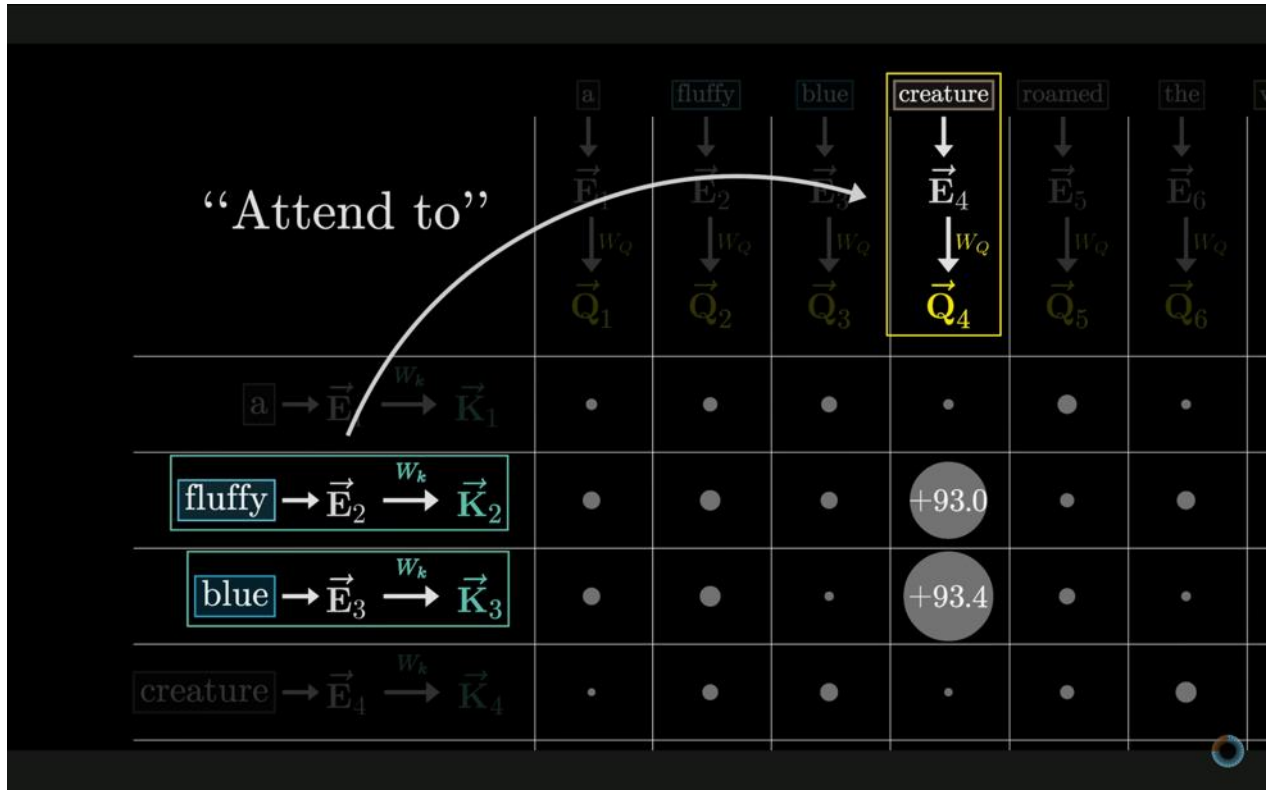

a fluffy blue creature roamed the verdant forest
Position
1 2 3 4 5 6 7 8
12,288
5.4 7.1 6.0 5.4 4.2 6.4 4.3 8.8 ⋮ 3.8
3.1 4.1 0.6 6.9 5.6 2.6 5.2 0.9 ⋮ 5.7
9.2 3.2 6.6 1.3 7.1 2.9 1.8 5.8 ⋮ 0.2
8.2 0.0 6.7 2.7 7.3 9.5 2.5 5.7 ⋮ 5.9
3.6 4.3 6.9 0.6 6.6 6.6 2.1 1.3 ⋮ 3.6
5.6 4.3 9.8 1.0 2.1 1.6 6.5 2.5 ⋮ 2.4
5.7 2.2 9.4 4.4 8.4 6.9 2.9 8.1 ⋮ 3.9
8.7 5.8 8.7 6.9 7.2 5.0 9.5 6.4 ⋮ 4.2


a fluffy blue creature roamed the verdant forest
$\vec{E}_1$ $\vec{E}_2$ $\vec{E}_3$ $\vec{E}_4$ $\vec{E}_5$ $\vec{E}_6$ $\vec{E}_7$ $\vec{E}_8$
$\vec{E}'_1$ $\vec{E}'_2$ $\vec{E}'_3$ $\vec{E}'_4$ $\vec{E}'_5$ $\vec{E}'_6$ $\vec{E}'_7$ $\vec{E}'_8$


I am!
I am!
a fluffy blue creature roamed the verdant forest
$\vec{E}_1$ $\vec{E}_2$ $\vec{E}_3$ $\vec{E}_4$ $\vec{E}_5$ $\vec{E}_6$ $\vec{E}_7$ $\vec{E}_8$
Query
128-dimensional
7.4
8.9
0.8
5.5
5.8
9.5
Any adjectives in front of me?


a fluffy blue creature roamed the verdant forest
$\vec{E}_1$ $\vec{E}_2$ $\vec{E}_3$ $\vec{E}_4$ $\vec{E}_5$ $\vec{E}_6$ $\vec{E}_7$ $\vec{E}_8$
W_Q
$\vec{Q}_1$ $\vec{Q}_2$ $\vec{Q}_3$ $\vec{Q}_4$ $\vec{Q}_5$ $\vec{Q}_6$ $\vec{Q}_7$ $\vec{Q}_8$
Any adjectives in front of me?
W_Q
$\vec{E}_i$
$\vec{Q}_i$
=


a fluffy blue creature roamed the verdant forest
$\vec{E}_1$ $\vec{E}_2$ $\vec{E}_3$ $\vec{E}_4$ $\vec{E}_5$ $\vec{E}_6$ $\vec{E}_7$ $\vec{E}_8$
W_Q
$\vec{Q}_1$ $\vec{Q}_2$ $\vec{Q}_3$ $\vec{Q}_4$ $\vec{Q}_5$ $\vec{Q}_6$ $\vec{Q}_7$ $\vec{Q}_8$
a → $\vec{E}_1$ W_k → $\vec{K}_1$
fluffy → $\vec{E}_2$ W_k → $\vec{K}_2$
blue → $\vec{E}_3$ W_k → $\vec{K}_3$
creature → $\vec{E}_4$ W_k → $\vec{K}_4$
roamed → $\vec{E}_5$ W_k → $\vec{K}_5$
the → $\vec{E}_6$ W_k → $\vec{K}_6$
verdant → $\vec{E}_7$ W_k → $\vec{K}_7$
forest → $\vec{E}_8$ W_k → $\vec{K}_8$
$\vec{K}_1 \cdot \vec{Q}_1$ $\vec{K}_1 \cdot \vec{Q}_2$ $\vec{K}_1 \cdot \vec{Q}_3$ $\vec{K}_1 \cdot \vec{Q}_4$ $\vec{K}_1 \cdot \vec{Q}_5$ $\vec{K}_1 \cdot \vec{Q}_6$ $\vec{K}_1 \cdot \vec{Q}_7$ $\vec{K}_1 \cdot \vec{Q}_8$
$\vec{K}_2 \cdot \vec{Q}_1$ $\vec{K}_2 \cdot \vec{Q}_2$ $\vec{K}_2 \cdot \vec{Q}_3$ $\vec{K}_2 \cdot \vec{Q}_4$ $\vec{K}_2 \cdot \vec{Q}_5$ $\vec{K}_2 \cdot \vec{Q}_6$ $\vec{K}_2 \cdot \vec{Q}_7$ $\vec{K}_2 \cdot \vec{Q}_8$
$\vec{K}_3 \cdot \vec{Q}_1$ $\vec{K}_3 \cdot \vec{Q}_2$ $\vec{K}_3 \cdot \vec{Q}_3$ $\vec{K}_3 \cdot \vec{Q}_4$ $\vec{K}_3 \cdot \vec{Q}_5$ $\vec{K}_3 \cdot \vec{Q}_6$ $\vec{K}_3 \cdot \vec{Q}_7$ $\vec{K}_3 \cdot \vec{Q}_8$
$\vec{K}_4 \cdot \vec{Q}_1$ $\vec{K}_4 \cdot \vec{Q}_2$ $\vec{K}_4 \cdot \vec{Q}_3$ $\vec{K}_4 \cdot \vec{Q}_4$ $\vec{K}_4 \cdot \vec{Q}_5$ $\vec{K}_4 \cdot \vec{Q}_6$ $\vec{K}_4 \cdot \vec{Q}_7$ $\vec{K}_4 \cdot \vec{Q}_8$
$\vec{K}_5 \cdot \vec{Q}_1$ $\vec{K}_5 \cdot \vec{Q}_2$ $\vec{K}_5 \cdot \vec{Q}_3$ $\vec{K}_5 \cdot \vec{Q}_4$ $\vec{K}_5 \cdot \vec{Q}_5$ $\vec{K}_5 \cdot \vec{Q}_6$ $\vec{K}_5 \cdot \vec{Q}_7$ $\vec{K}_5 \cdot \vec{Q}_8$
$\vec{K}_6 \cdot \vec{Q}_1$ $\vec{K}_6 \cdot \vec{Q}_2$ $\vec{K}_6 \cdot \vec{Q}_3$ $\vec{K}_6 \cdot \vec{Q}_4$ $\vec{K}_6 \cdot \vec{Q}_5$ $\vec{K}_6 \cdot \vec{Q}_6$ $\vec{K}_6 \cdot \vec{Q}_7$ $\vec{K}_6 \cdot \vec{Q}_8$
$\vec{K}_7 \cdot \vec{Q}_1$ $\vec{K}_7 \cdot \vec{Q}_2$ $\vec{K}_7 \cdot \vec{Q}_3$ $\vec{K}_7 \cdot \vec{Q}_4$ $\vec{K}_7 \cdot \vec{Q}_5$ $\vec{K}_7 \cdot \vec{Q}_6$ $\vec{K}_7 \cdot \vec{Q}_7$ $\vec{K}_7 \cdot \vec{Q}_8$
$\vec{K}_8 \cdot \vec{Q}_1$ $\vec{K}_8 \cdot \vec{Q}_2$ $\vec{K}_8 \cdot \vec{Q}_3$ $\vec{K}_8 \cdot \vec{Q}_4$ $\vec{K}_8 \cdot \vec{Q}_5$ $\vec{K}_8 \cdot \vec{Q}_6$ $\vec{K}_8 \cdot \vec{Q}_7$ $\vec{K}_8 \cdot \vec{Q}_8$


“Attend to”
a fluffy blue creature roamed the
$\vec{E}_4$
W_Q
$\vec{Q}_4$
fluffy → $\vec{E}_2$ W_k → $\vec{K}_2$
blue → $\vec{E}_3$ W_k → $\vec{K}_3$
creature → $\vec{E}_4$ W_k → $\vec{K}_4$
+93.0
+93.4

	a $\vec{E}_1 \xrightarrow{W_Q} \vec{Q}_1$	fluffy $\vec{E}_2 \xrightarrow{W_Q} \vec{Q}_2$	blue $\vec{E}_3 \xrightarrow{W_Q} \vec{Q}_3$	creature $\vec{E}_4 \xrightarrow{W_Q} \vec{Q}_4$	roamed $\vec{E}_5 \xrightarrow{W_Q} \vec{Q}_5$	the $\vec{E}_6 \xrightarrow{W_Q} \vec{Q}_6$	verdant $\vec{E}_7 \xrightarrow{W_Q} \vec{Q}_7$	forest $\vec{E}_8 \xrightarrow{W_Q} \vec{Q}_8$
a $\vec{E}_1 \xrightarrow{W_K} \vec{K}_1$	+0.7	−83.7	−24.7	−27.8	−5.2	−89.3	−45.2	−36.1
fluffy $\vec{E}_2 \xrightarrow{W_K} \vec{K}_2$	−73.4	+2.9	−5.4	+93.0	−48.2	−87.3	−49.7	+7.8
blue $\vec{E}_3 \xrightarrow{W_K} \vec{K}_3$	−53.4	−5.7	+1.8	+93.4	−55.6	−56.0	−26.1	−62.1
creature $\vec{E}_4 \xrightarrow{W_K} \vec{K}_4$	−21.5	−29.7	−56.1	+4.9	−32.4	−92.3	−9.5	−28.1
roamed $\vec{E}_5 \xrightarrow{W_K} \vec{K}_5$	−20.1	−40.9	−87.8	−55.4	+0.6	−64.7	−96.7	−18.9
the $\vec{E}_6 \xrightarrow{W_K} \vec{K}_6$	−87.9	−33.3	−22.6	−31.4	+5.5	+0.6	−4.6	−96.8
verdant $\vec{E}_7 \xrightarrow{W_K} \vec{K}_7$	−41.2	−55.5	−42.3	−59.8	−79.0	−97.9	+3.7	+93.8
forest $\vec{E}_8 \xrightarrow{W_K} \vec{K}_8$	−58.9	−75.5	−91.1	−90.6	−75.6	−89.0	−70.8	+4.7

	a $\vec{E}_1 \xrightarrow{W_Q} \vec{Q}_1$	fluffy $\vec{E}_2 \xrightarrow{W_Q} \vec{Q}_2$	blue $\vec{E}_3 \xrightarrow{W_Q} \vec{Q}_3$	creature $\vec{E}_4 \xrightarrow{W_Q} \vec{Q}_4$	roamed $\vec{E}_5 \xrightarrow{W_Q} \vec{Q}_5$	the $\vec{E}_6 \xrightarrow{W_Q} \vec{Q}_6$	verdant $\vec{E}_7 \xrightarrow{W_Q} \vec{Q}_7$	forest $\vec{E}_8 \xrightarrow{W_Q} \vec{Q}_8$
a $\vec{E}_1 \xrightarrow{W_K} \vec{K}_1$	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
fluffy $\vec{E}_2 \xrightarrow{W_K} \vec{K}_2$	0.00	1.00	0.00	0.42	0.00	0.00	0.00	0.00
blue $\vec{E}_3 \xrightarrow{W_K} \vec{K}_3$	0.00	0.00	1.00	0.58	0.00	0.00	0.00	0.00
creature $\vec{E}_4 \xrightarrow{W_K} \vec{K}_4$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
roamed $\vec{E}_5 \xrightarrow{W_K} \vec{K}_5$	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00
the $\vec{E}_6 \xrightarrow{W_K} \vec{K}_6$	0.00	0.00	0.00	0.00	0.99	1.00	0.00	0.00
verdant $\vec{E}_7 \xrightarrow{W_K} \vec{K}_7$	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00
forest $\vec{E}_8 \xrightarrow{W_K} \vec{K}_8$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

	Q_1	Q_2	Q_3	Q_4	Q_5	$\cdots$	Q_n
K_1	$\frac{Q_1 \cdot K_1}{\sqrt{d_k}}$	$\frac{Q_2 \cdot K_1}{\sqrt{d_k}}$	$\frac{Q_3 \cdot K_1}{\sqrt{d_k}}$	$\frac{Q_4 \cdot K_1}{\sqrt{d_k}}$	$\frac{Q_5 \cdot K_1}{\sqrt{d_k}}$	$\cdots$	$\frac{Q_n \cdot K_1}{\sqrt{d_k}}$
K_2	$\frac{Q_1 \cdot K_2}{\sqrt{d_k}}$	$\frac{Q_2 \cdot K_2}{\sqrt{d_k}}$	$\frac{Q_3 \cdot K_2}{\sqrt{d_k}}$	$\frac{Q_4 \cdot K_2}{\sqrt{d_k}}$	$\frac{Q_5 \cdot K_2}{\sqrt{d_k}}$	$\cdots$	$\frac{Q_n \cdot K_2}{\sqrt{d_k}}$
K_3	$\frac{Q_1 \cdot K_3}{\sqrt{d_k}}$	$\frac{Q_2 \cdot K_3}{\sqrt{d_k}}$	$\frac{Q_3 \cdot K_3}{\sqrt{d_k}}$	$\frac{Q_4 \cdot K_3}{\sqrt{d_k}}$	$\frac{Q_5 \cdot K_3}{\sqrt{d_k}}$	$\cdots$	$\frac{Q_n \cdot K_3}{\sqrt{d_k}}$
K_4	$\frac{Q_1 \cdot K_4}{\sqrt{d_k}}$	$\frac{Q_2 \cdot K_4}{\sqrt{d_k}}$	$\frac{Q_3 \cdot K_4}{\sqrt{d_k}}$	$\frac{Q_4 \cdot K_4}{\sqrt{d_k}}$	$\frac{Q_5 \cdot K_4}{\sqrt{d_k}}$	$\cdots$	$\frac{Q_n \cdot K_4}{\sqrt{d_k}}$
K_5	$\frac{Q_1 \cdot K_5}{\sqrt{d_k}}$	$\frac{Q_2 \cdot K_5}{\sqrt{d_k}}$	$\frac{Q_3 \cdot K_5}{\sqrt{d_k}}$	$\frac{Q_4 \cdot K_5}{\sqrt{d_k}}$	$\frac{Q_5 \cdot K_5}{\sqrt{d_k}}$	$\cdots$	$\frac{Q_n \cdot K_5}{\sqrt{d_k}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$

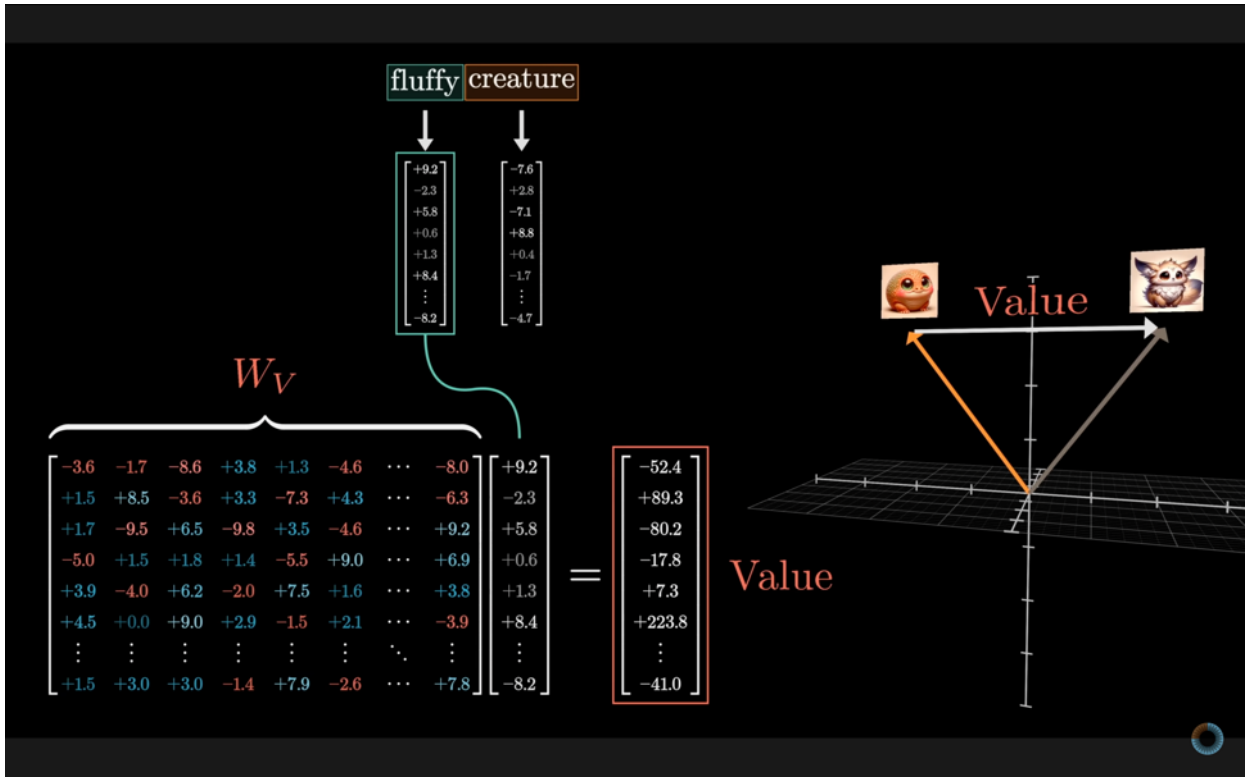
Unnormalized Attention Pattern

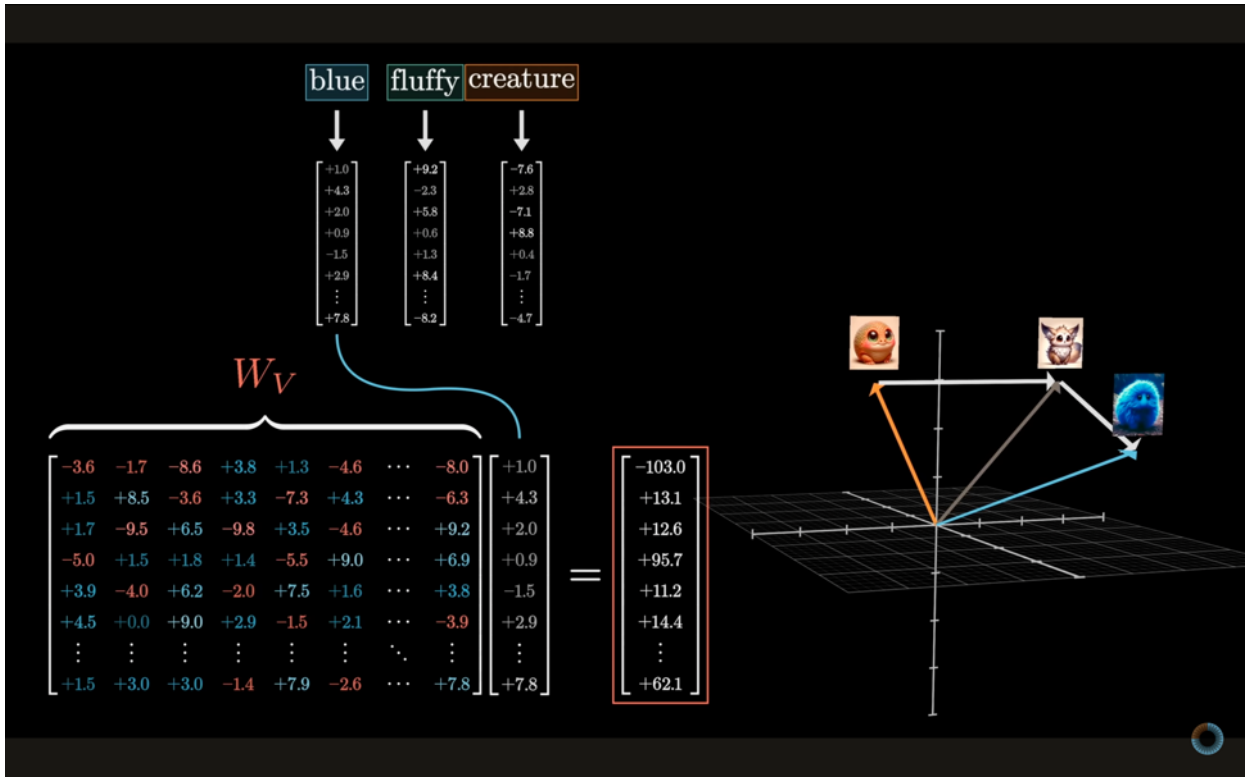
+3.53	+0.80	+1.96	+4.48	+3.74	−1.95
−∞	−0.30	−0.21	+0.82	+0.29	+2.91
−∞	−∞	+0.89	+0.67	+2.99	−0.41
−∞	−∞	−∞	+1.31	+1.73	−1.48
−∞	−∞	−∞	−∞	+3.07	+2.94
−∞	−∞	−∞	−∞	−∞	+0.31

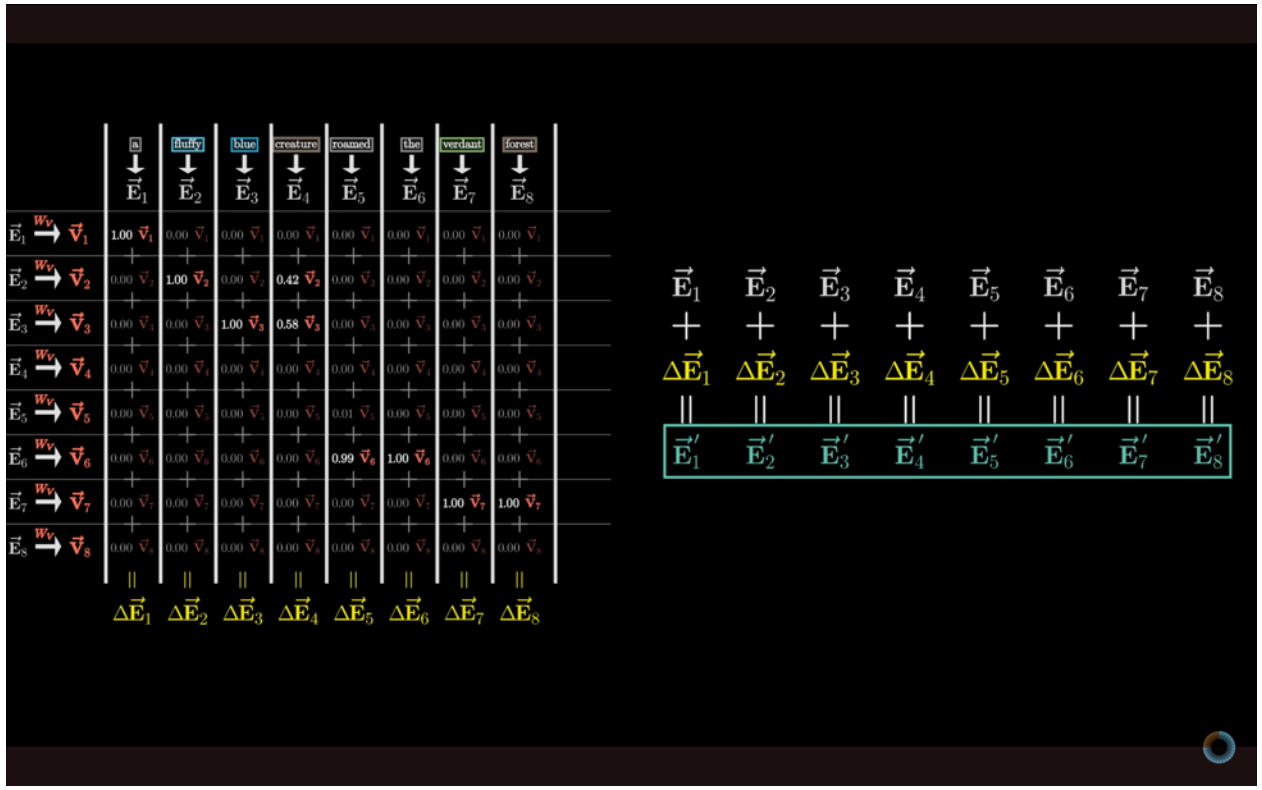
softmax →

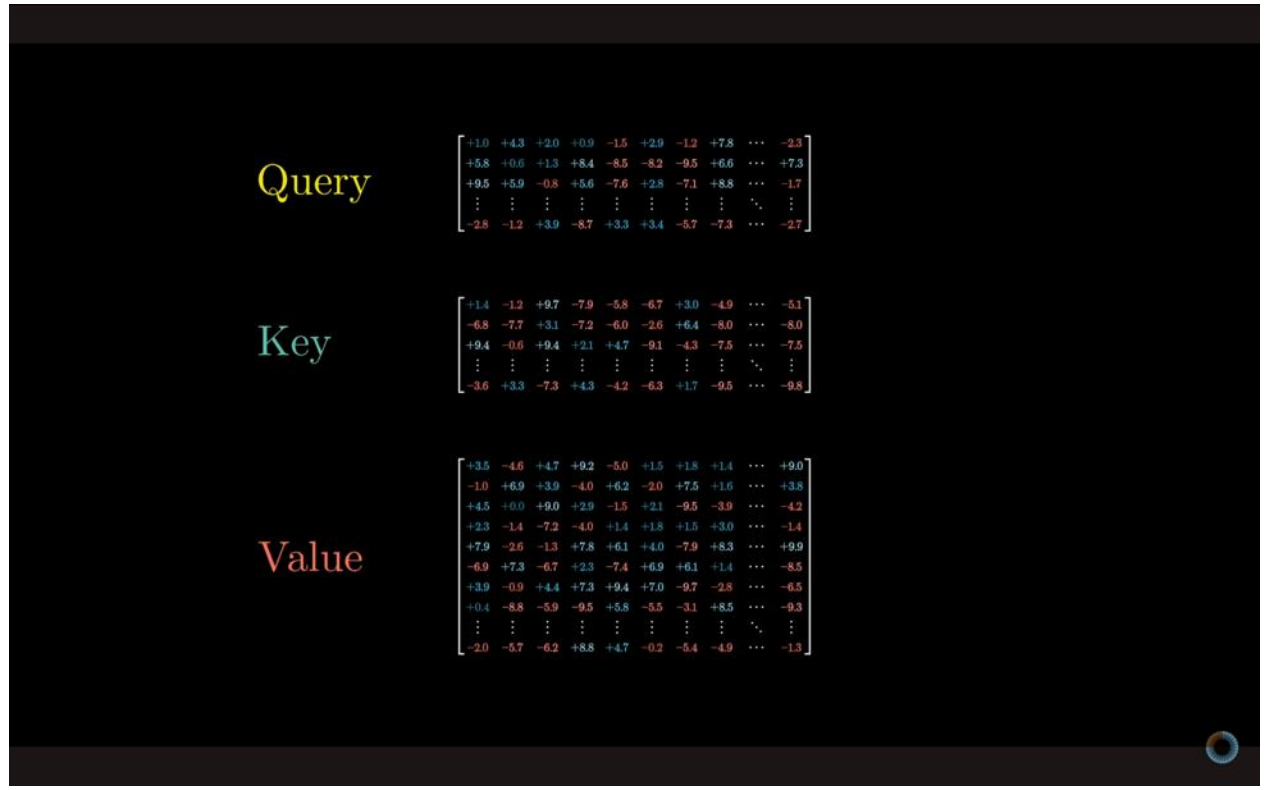
Normalized Attention Pattern

1.00	0.75	0.69	0.92	0.46	0.00
0.00	0.25	0.08	0.02	0.01	0.46
0.00	0.00	0.24	0.02	0.22	0.02
0.00	0.00	0.00	0.04	0.06	0.01
0.00	0.00	0.00	0.00	0.24	0.48
0.00	0.00	0.00	0.00	0.00	0.03

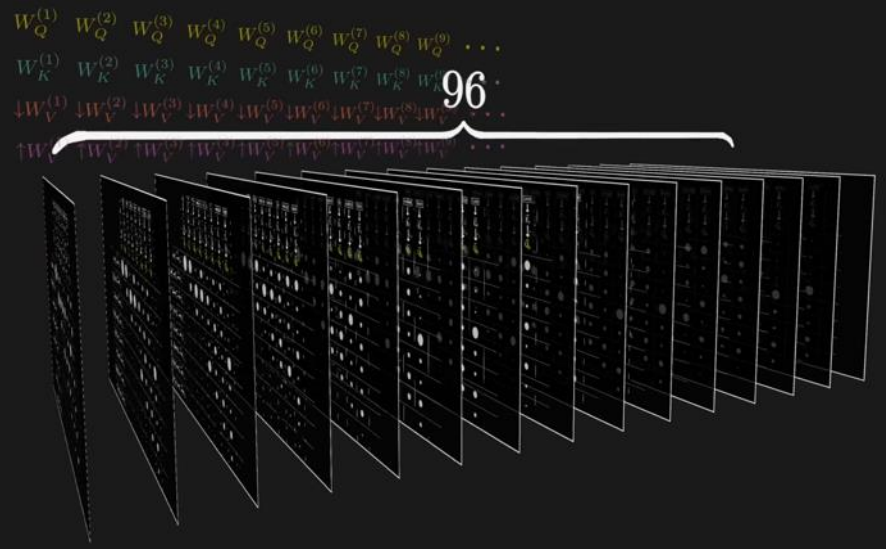

fluffy
creature
W_V
Value
Value

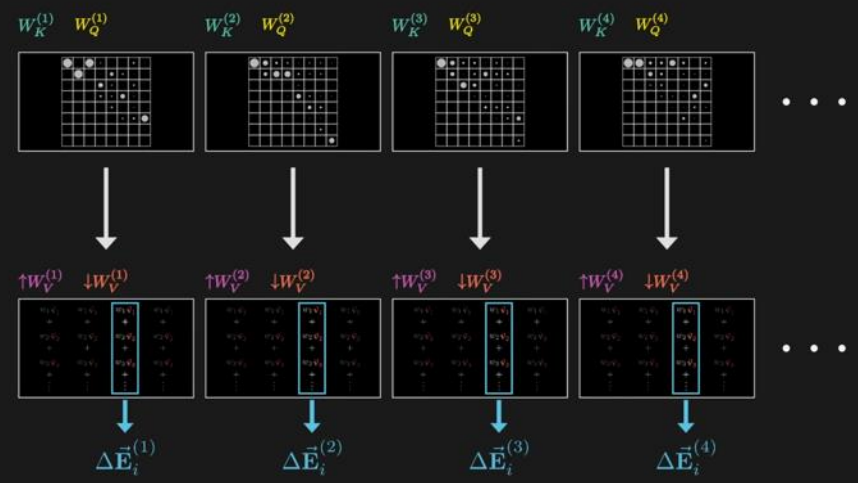

blue
fluffy
creature
W_V


a
fluffy
blue
creature
roamed
the
verdant
forest
$\vec{E}_1$ $\vec{E}_2$ $\vec{E}_3$ $\vec{E}_4$ $\vec{E}_5$ $\vec{E}_6$ $\vec{E}_7$ $\vec{E}_8$
W_V
$\vec{V}_1$ $\vec{V}_2$ $\vec{V}_3$ $\vec{V}_4$ $\vec{V}_5$ $\vec{V}_6$ $\vec{V}_7$ $\vec{V}_8$
1.00 0.42 0.58 0.01 0.99 1.00
$\Delta\vec{E}_1$ $\Delta\vec{E}_2$ $\Delta\vec{E}_3$ $\Delta\vec{E}_4$ $\Delta\vec{E}_5$ $\Delta\vec{E}_6$ $\Delta\vec{E}_7$ $\Delta\vec{E}_8$
$\vec{E}_1'$ $\vec{E}_2'$ $\vec{E}_3'$ $\vec{E}_4'$ $\vec{E}_5'$ $\vec{E}_6'$ $\vec{E}_7'$ $\vec{E}_8'$

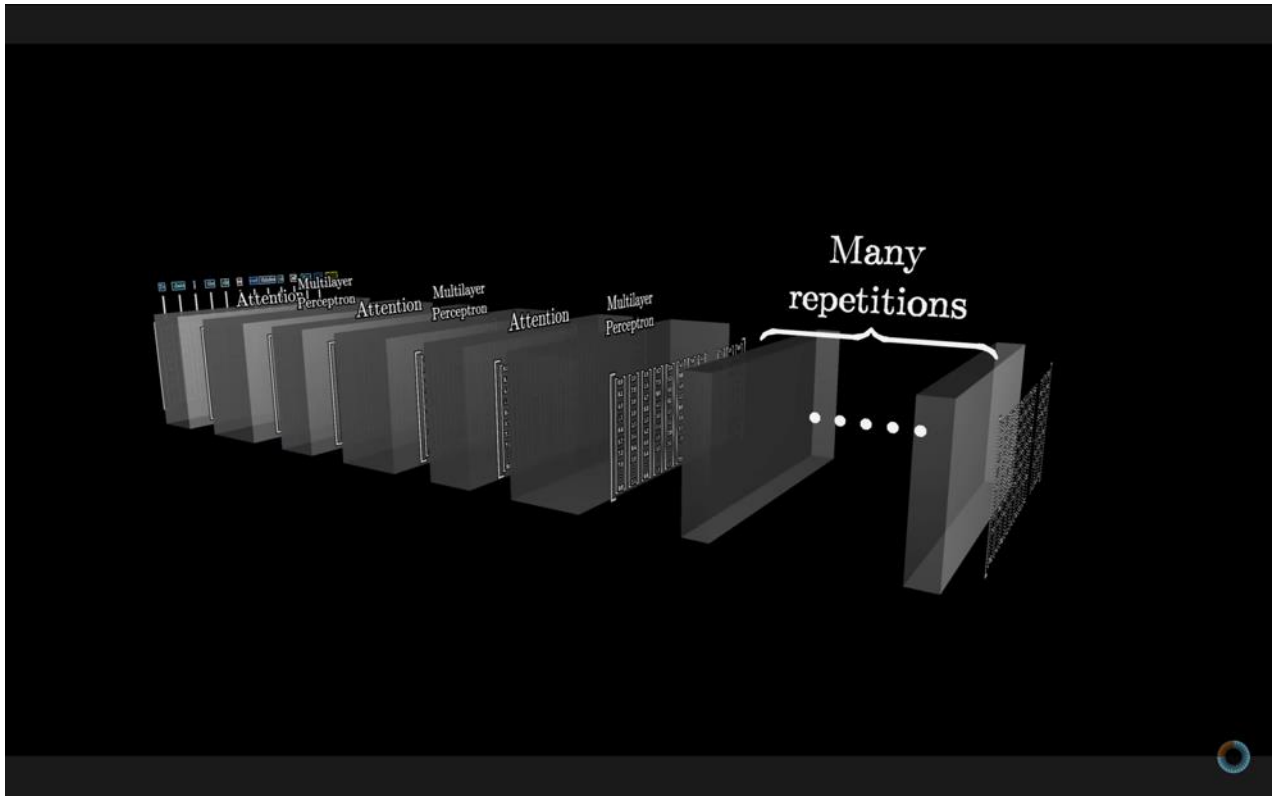

Query
Key
Value

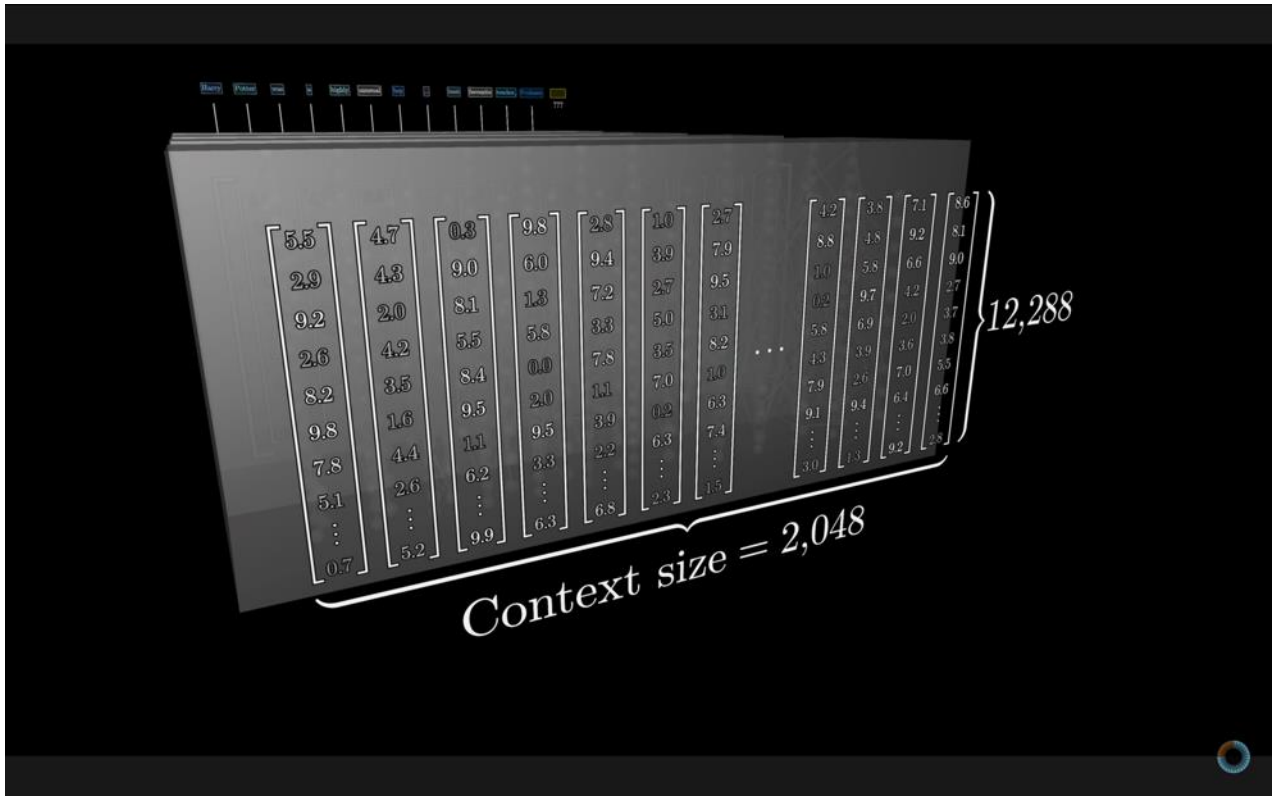
Multi-headed attention





Original embedding $\vec{\mathbf{E}}_i + \Delta\vec{\mathbf{E}}_i^{(1)} + \Delta\vec{\mathbf{E}}_i^{(2)} + \Delta\vec{\mathbf{E}}_i^{(3)} + \Delta\vec{\mathbf{E}}_i^{(4)} + \cdots$


Attention
Multilayer Perceptron
Attention
Multilayer Perceptron
Attention
Multilayer Perceptron
Many repetitions

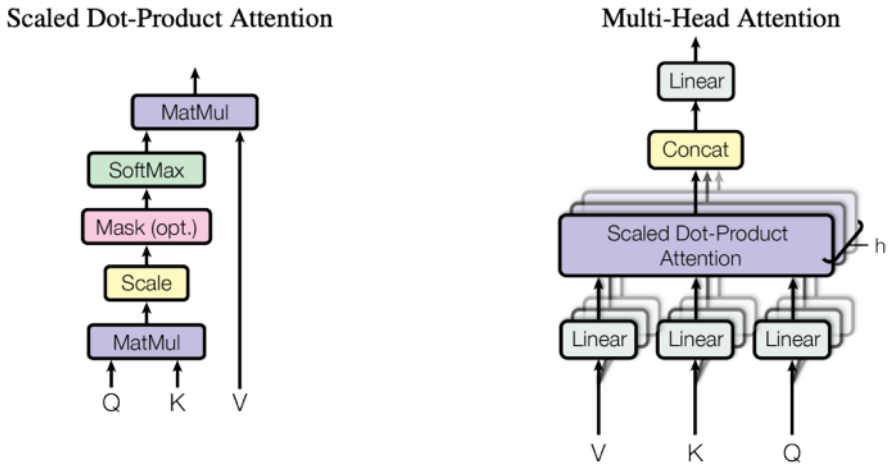

12,288
Context size = 2,048

Transformers

Advantage: we can calculate attention for all the embeddings in a sequence simultaneously. This speeds up computation and allows for massive scalability

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



Vaswani, A. "Attention is all you need." *Advances in Neural Information Processing Systems* (2017).

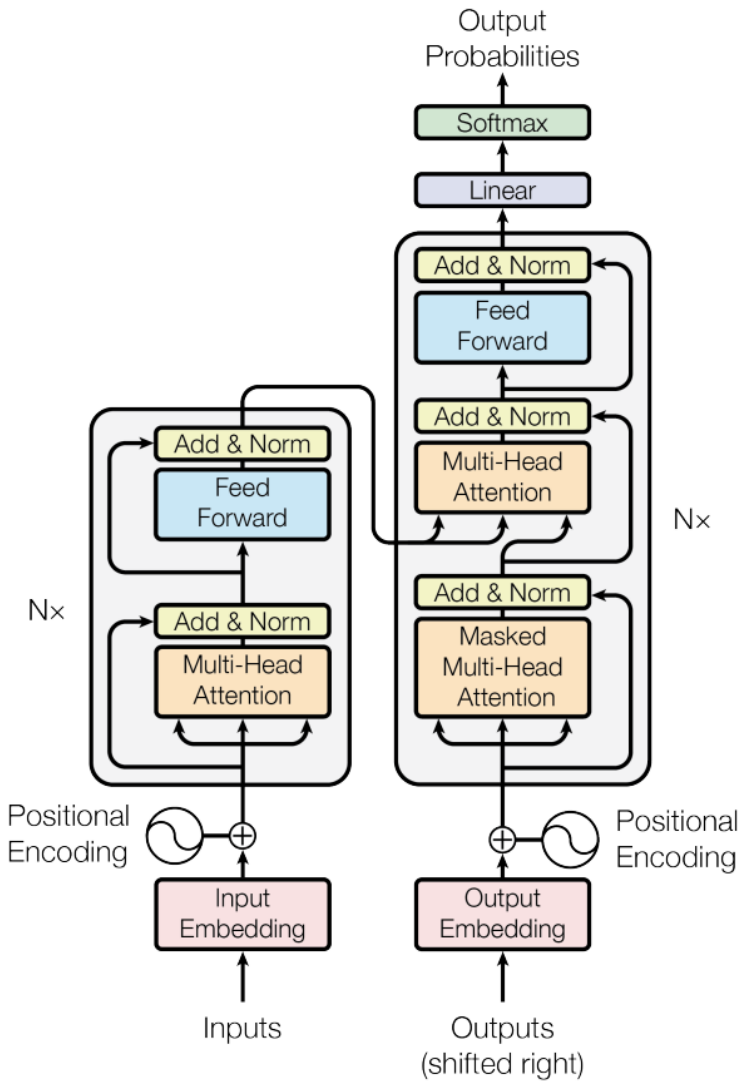


Figure 1: The Transformer - model architecture.