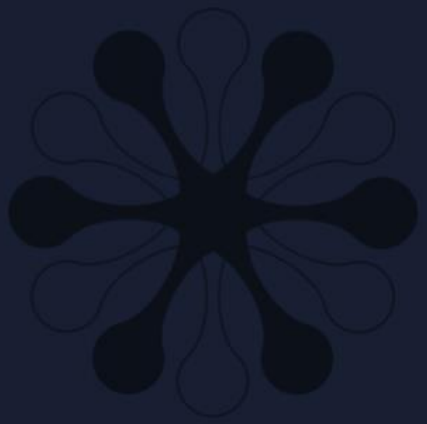


Escuela AI HUB de verano
Sobre IA y Educación
aihub.csic.es

Avances en Aprendizaje Automático

Charla

Martí Sánchez Fibla
IIIA-CSIC

Deep Learning Milestones

Escuela AI HUB Sobre IA y Educación de verano
aihub.csic.es

Learning to see (and hear)

1986 Backpropagation (invented 1960...)
2012 Deep Convolutional Neural Networks

Supervised & Unsupervised Deep Learning

Learning to act

1980 TD learning (Sutton and Barto)
2015 Deep Q Learning solves Atari games

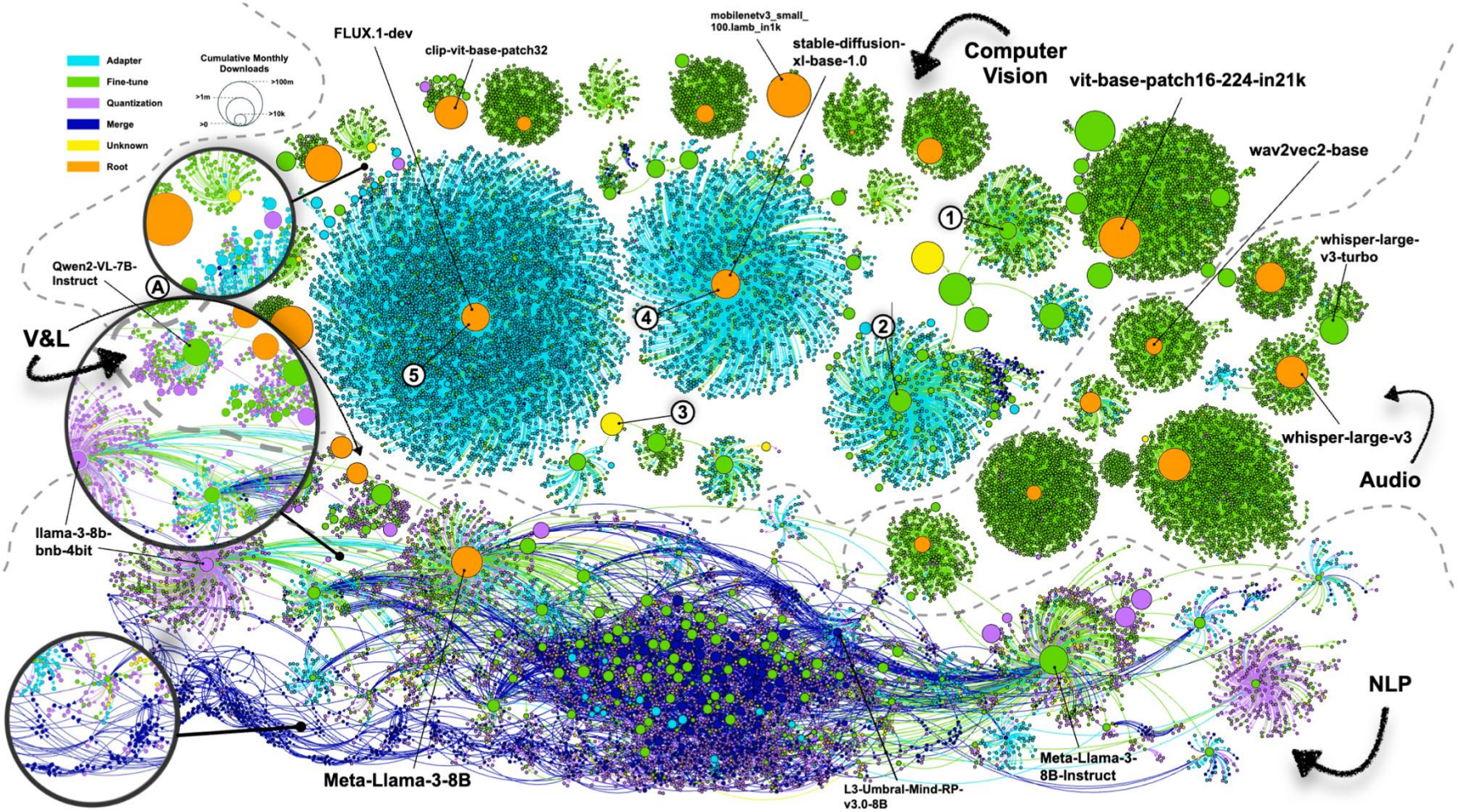
Deep Reinforcement Learning

Learning to ... ??? from text

2013 Embeddings (Word2Vec)
2017 Transformers, 2019 Scaling Works
2022 RL from Human Feedback

Self Supervised Deep Learning

Deep Learning Models Atlas

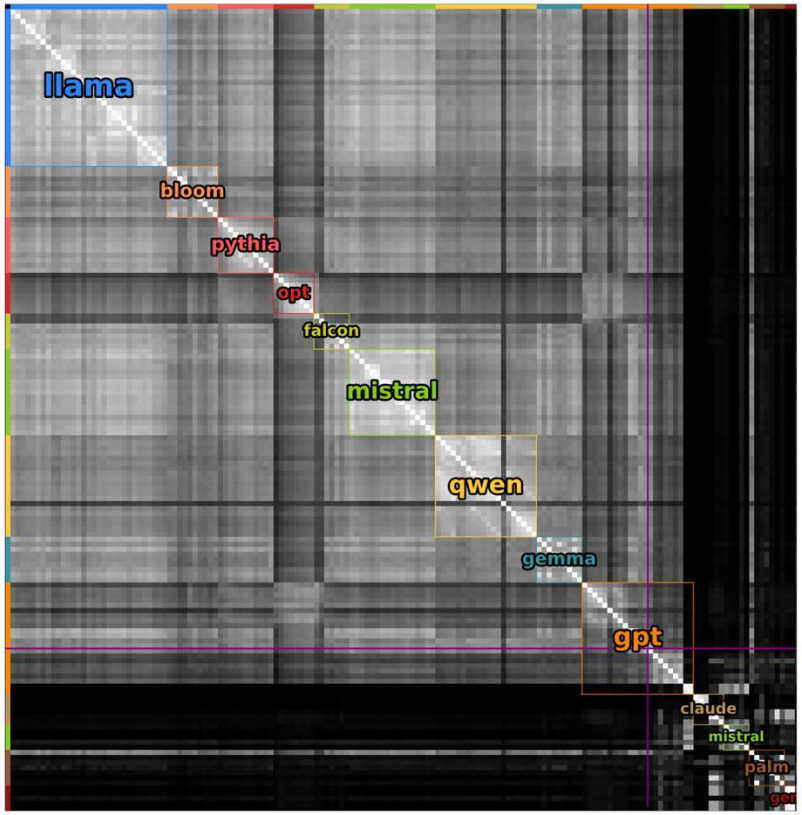


A small selection of 63000 Models HuggingFace Model

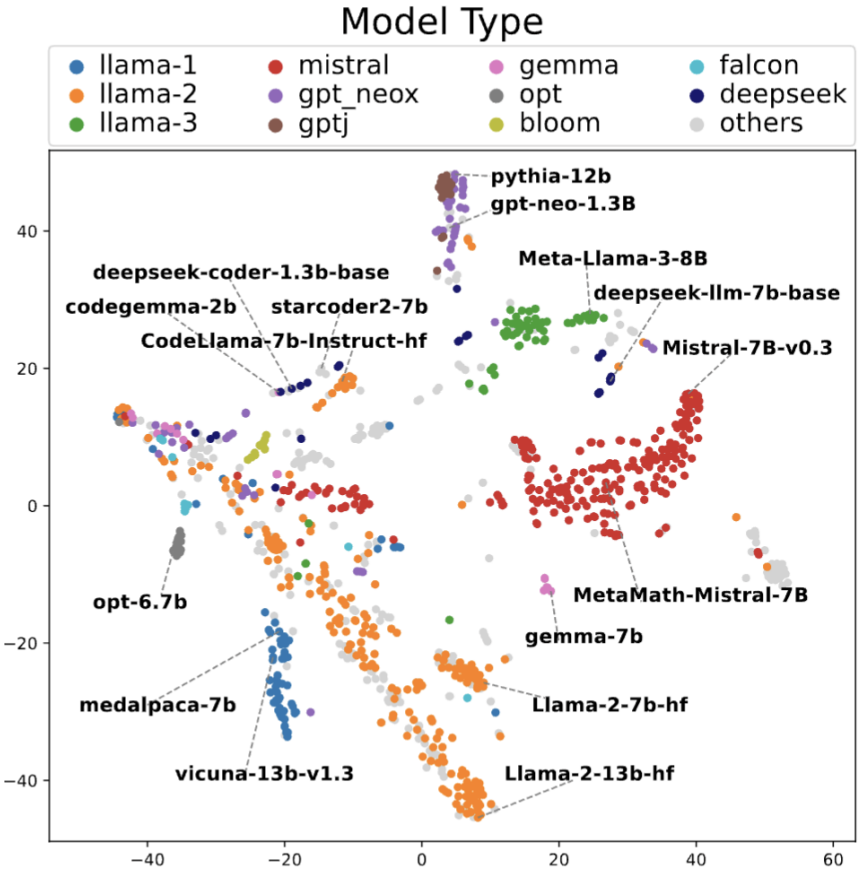
LLM Ecosystem

Escuela AI HUB Sobre IA y Educación de verano
aihub.csic.es

600+ every day

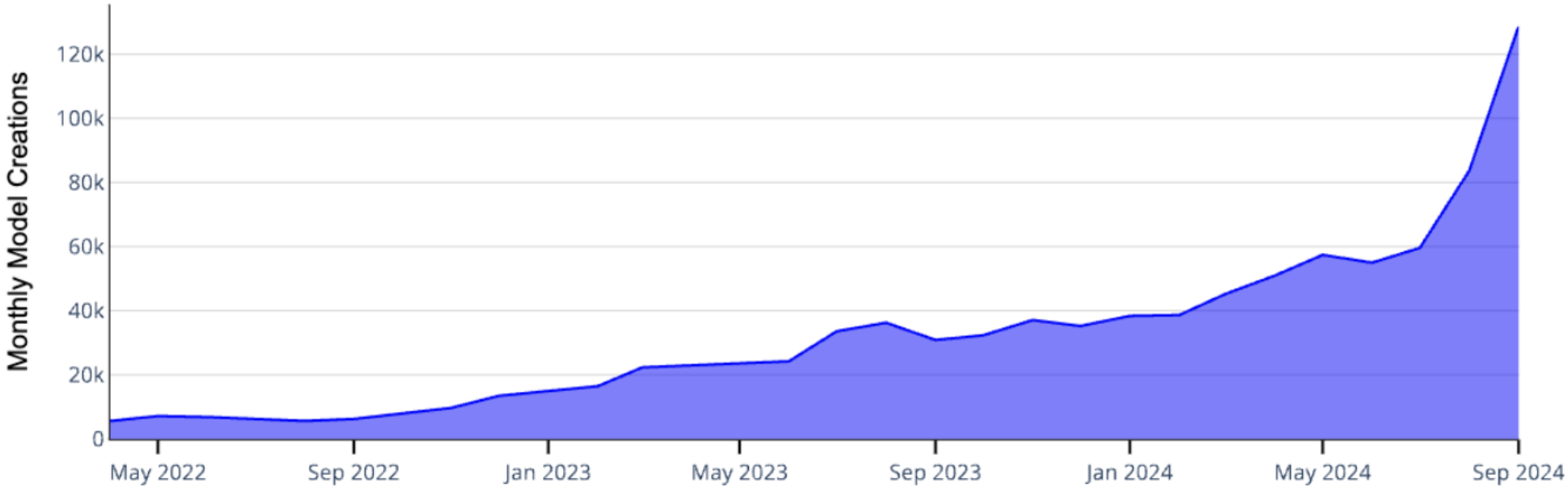


PhyloLM : Inferring the Phylogeny of Large Language Models and Predicting their Performances in Benchmarks Yax et al, 2024



Mapping 1,000+ Language Models via the Log-Likelihood Vector Oyama et al, 2025

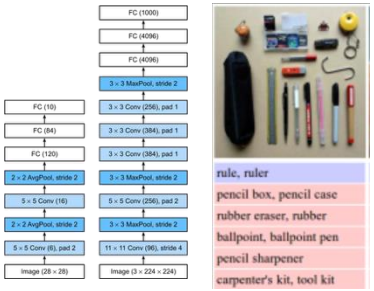
- And the trend is increasing



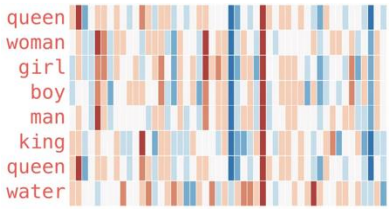
More Deep Learning Milestones

Escuela AI HUB de verano
Sobre IA y Educación
aihub.csic.es

AlexNet: Convolutional



Word2Vec: Embeddings



Transformers

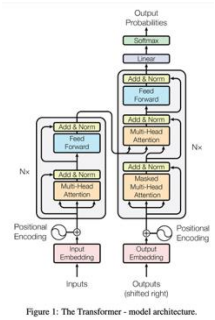


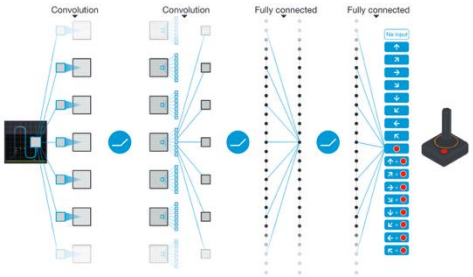
Figure 1: The Transformer - model architecture.

OpenAI ChatGPT

AlphaEvolve
AlphaProof
AlphaGeometry

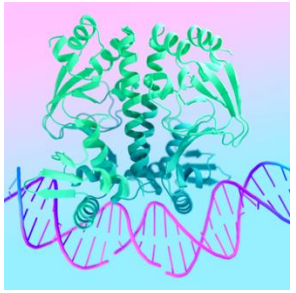
2012 2013 2016 2017 2019 2022 2024 2025

Atari: DQN

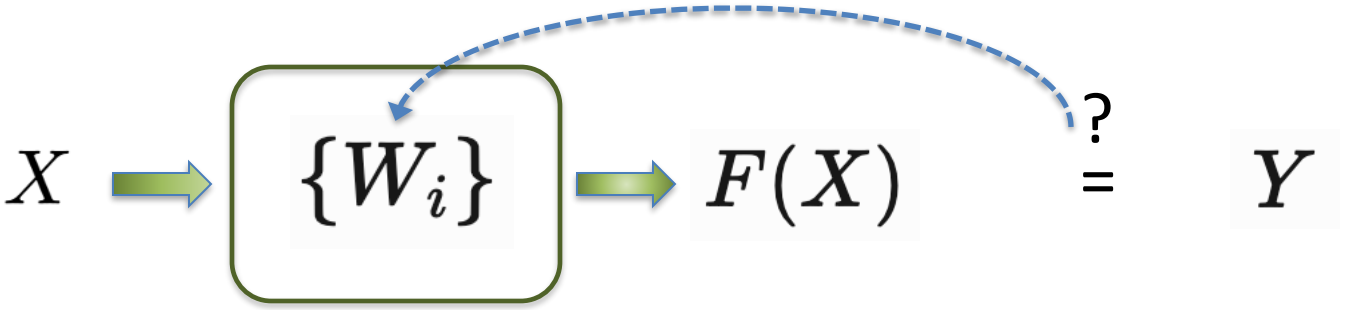


AlphaStar: StarCraft II

AlphaFold 3:
Protein Interactions



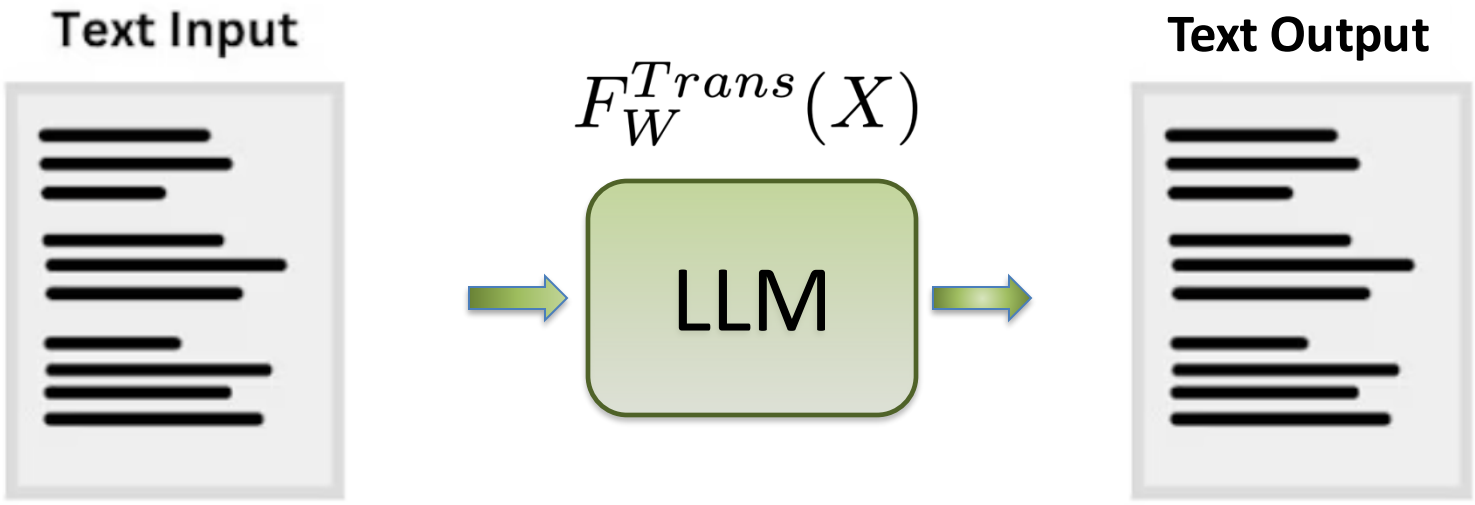
Deep Learning Main Principle



Number 2

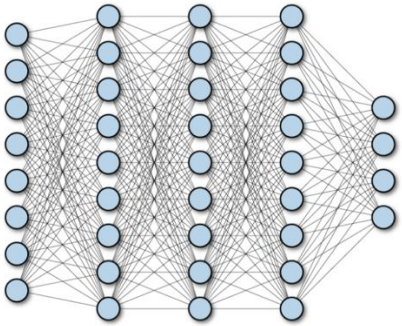
Border Terrier

What is an LLM?



It's a Neural Network very similar to a:

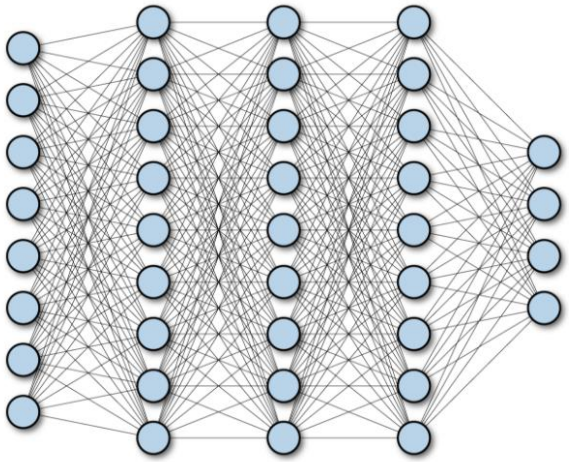
Multi
Layer
Perceptron



What is a Multi Layer Perceptron?

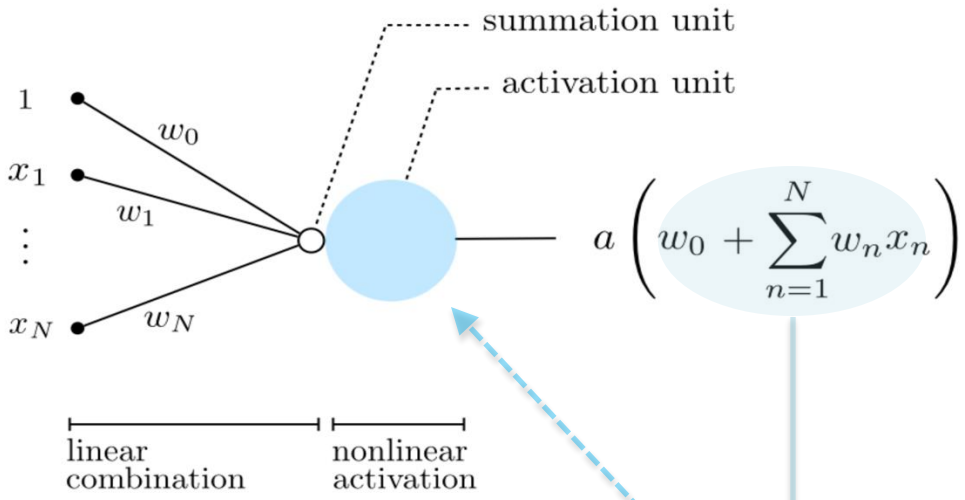
MLP

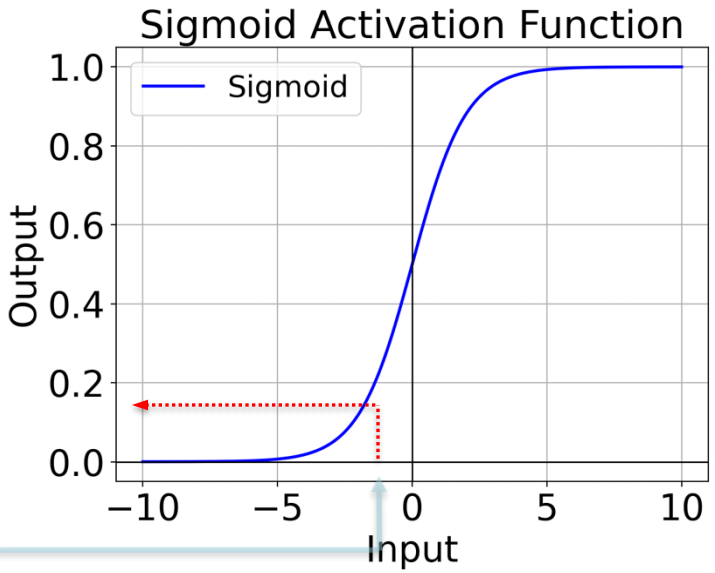
$X \in \mathbb{R}^d$

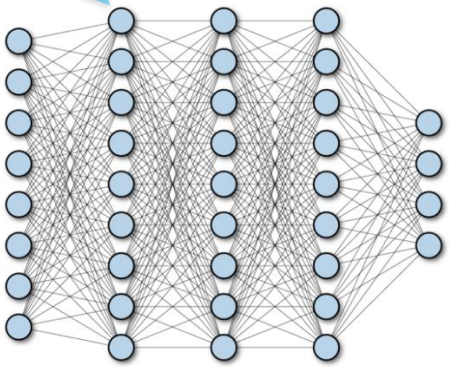


$$F^{MLP}_{\{W_i,b_i\}}(X) = \left(f^{act} \circ f^{L}_{W_L,b_L}\right) \circ \cdots \circ \left(f^{act} \circ f^{L}_{W_1,b_1}\right)(X)$$

What is a Neuron?

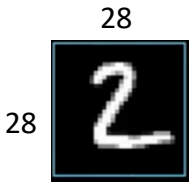


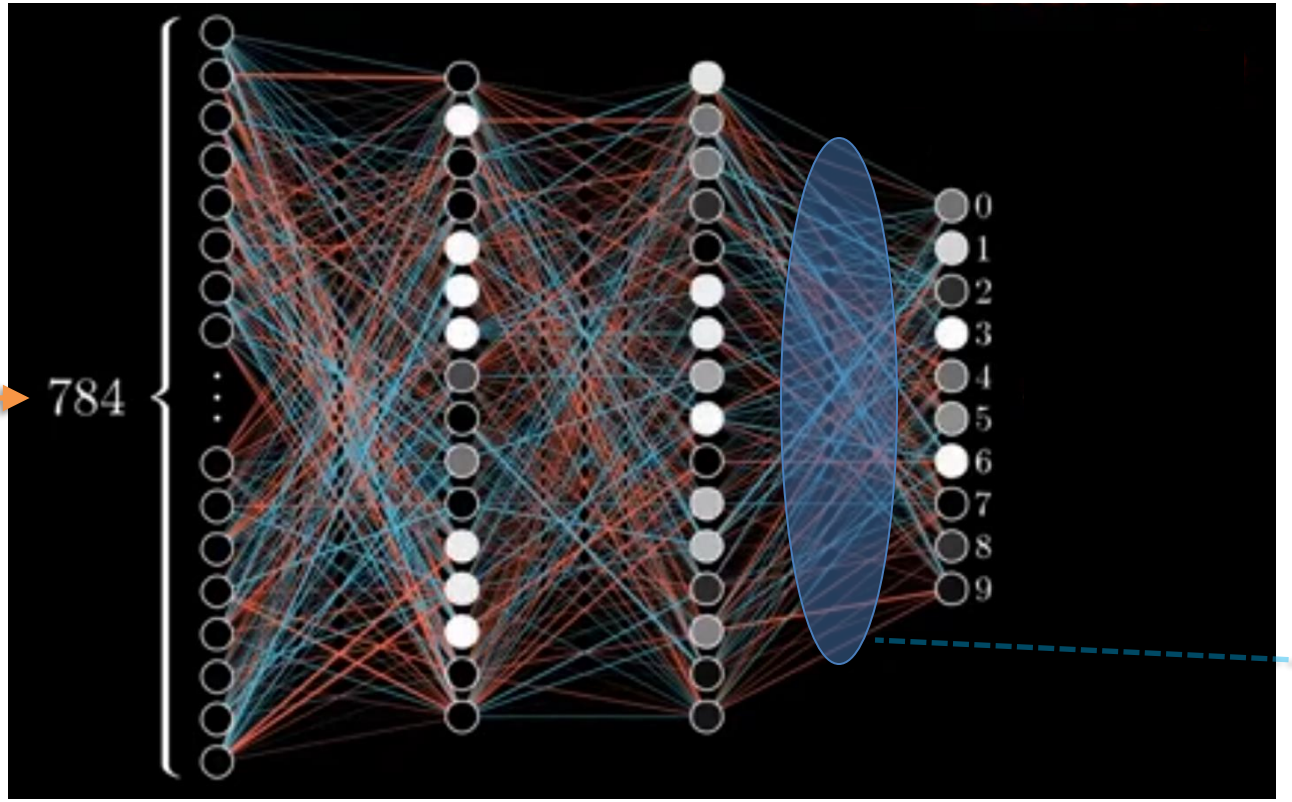




MLP Forward Pass

$X \in \mathbb{R}^d$





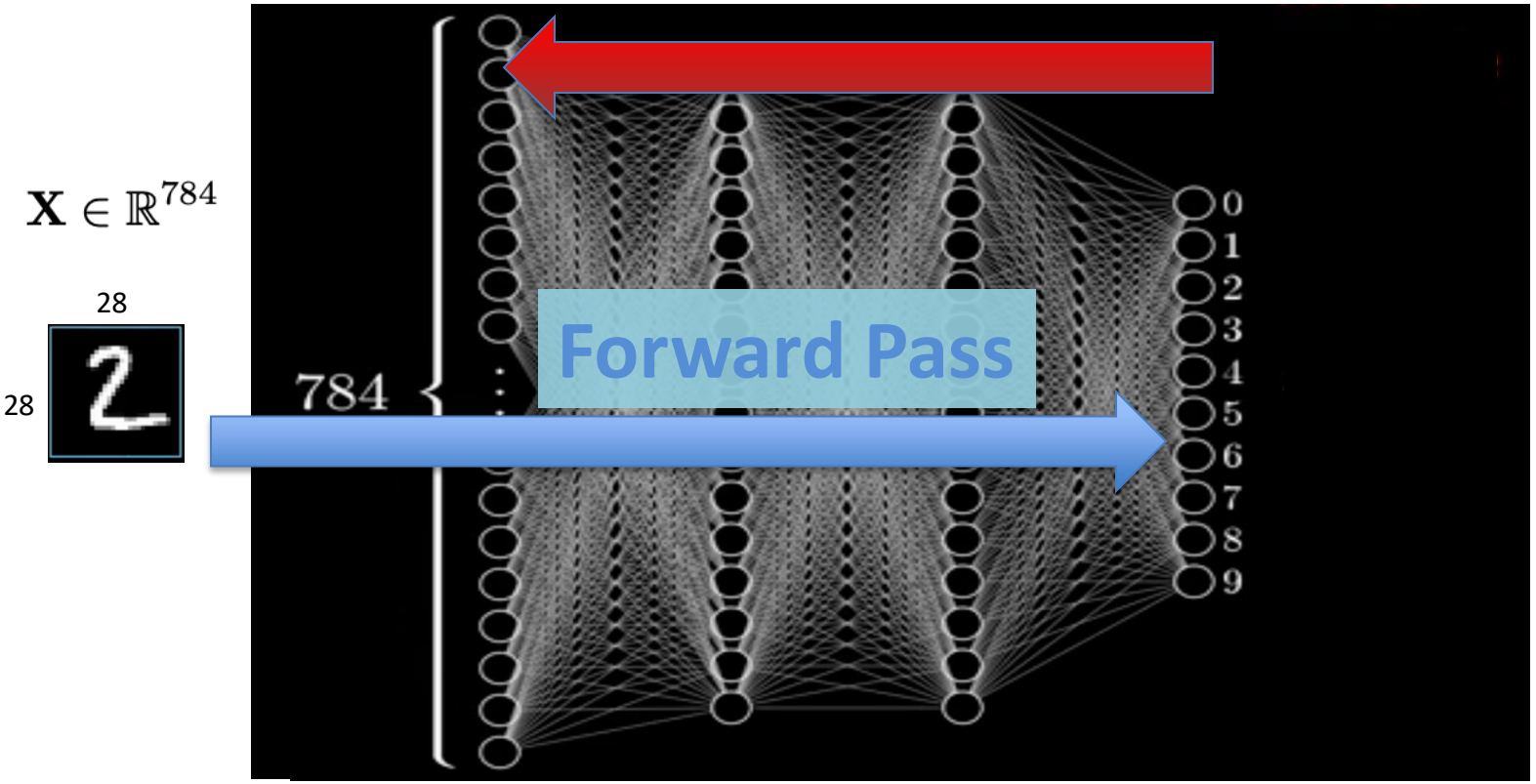
W_1

MLP Training

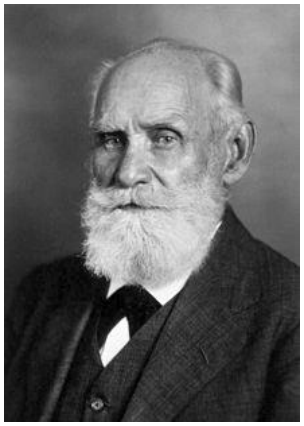
$$F^{MLP}_{\{W_i,b_i\}}(X) = \left(f^{act} \circ f^{L}_{W_L,b_L}\right) \circ \cdots \circ \left(f^{act} \circ f^{L}_{W_1,b_1}\right)(X)$$

$$W_i \leftarrow W_i - \eta \frac{\partial J}{\partial W_i}, \quad b_i \leftarrow b_i - \eta \frac{\partial J}{\partial b_i} \quad \text{for } i = 1, \ldots, L.$$

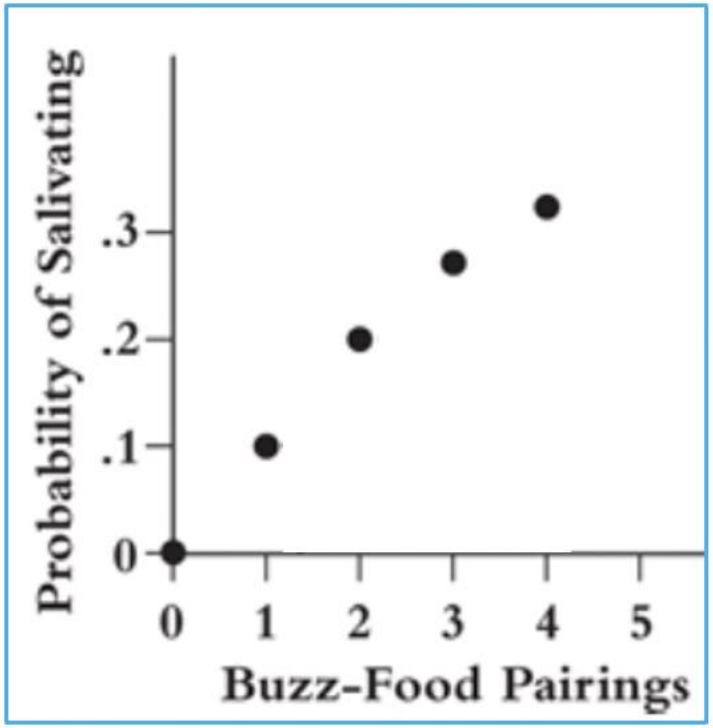
Backward Pass



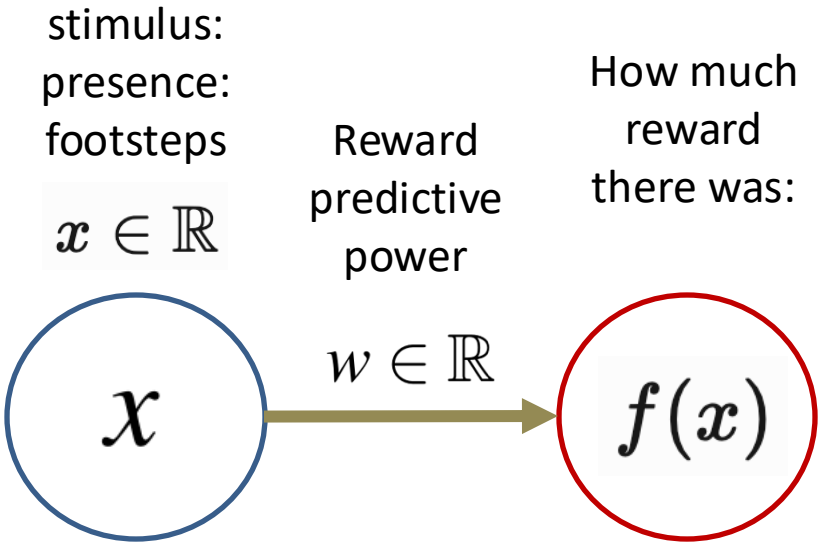
What is Learning?



An experimental curve



How can we model Learning?

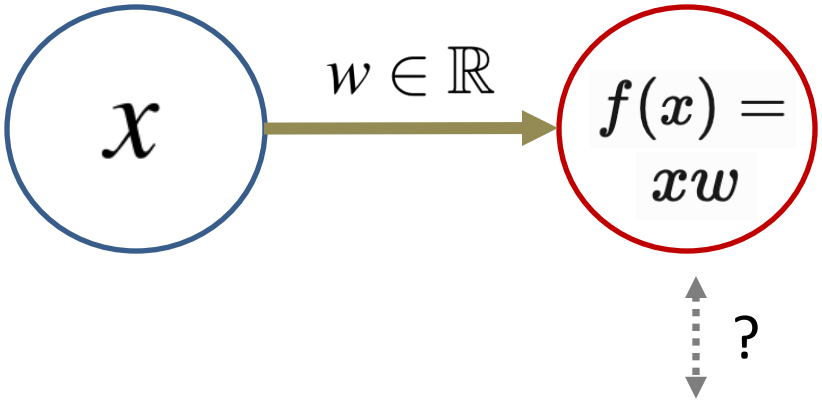


$$f(x) = \text{expression?}$$

$$f(x) = xw$$

How can we model Learning?

$w \leftarrow 0$



We observe a training sample, example

$x = 1$

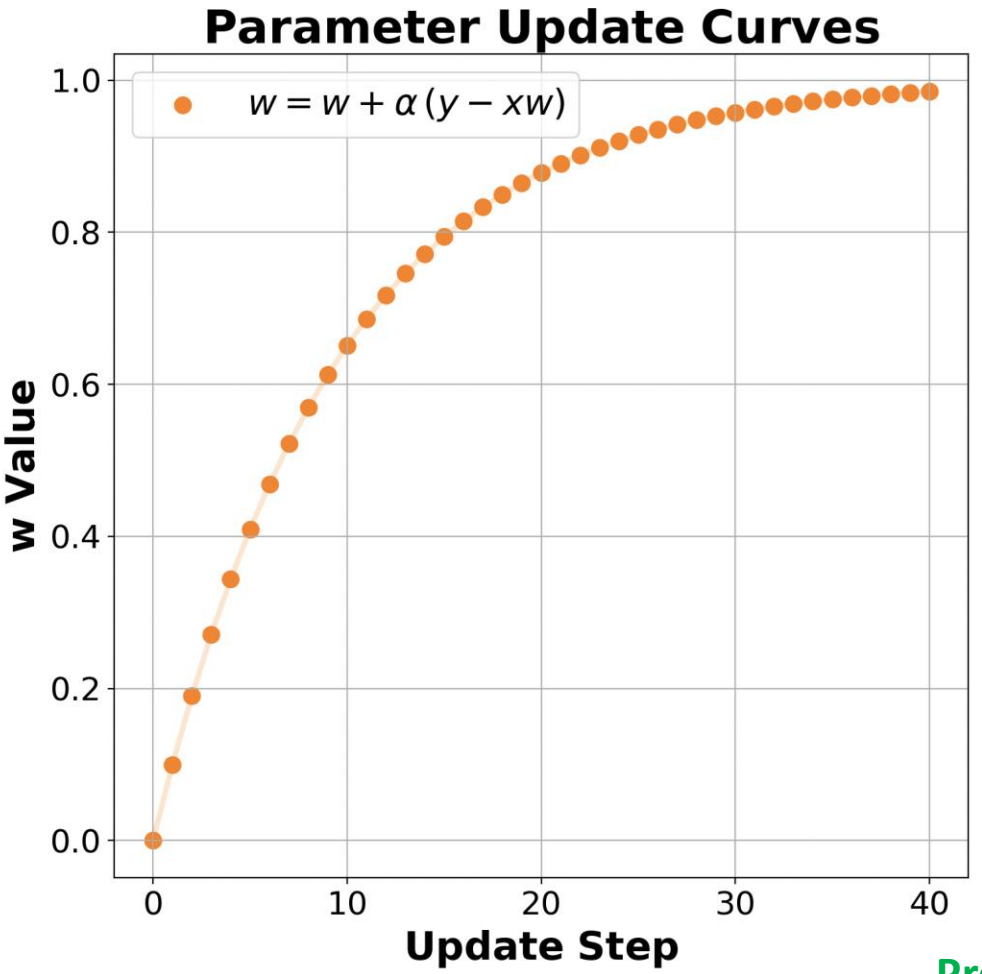
$y = 1$

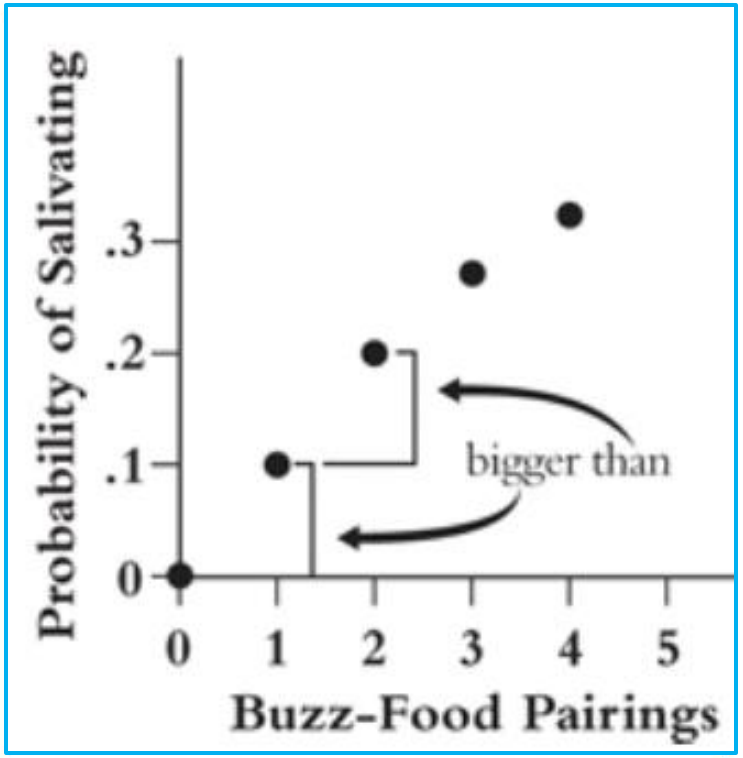
α ?

w update?

$w = w + \alpha y$

What is Learning?

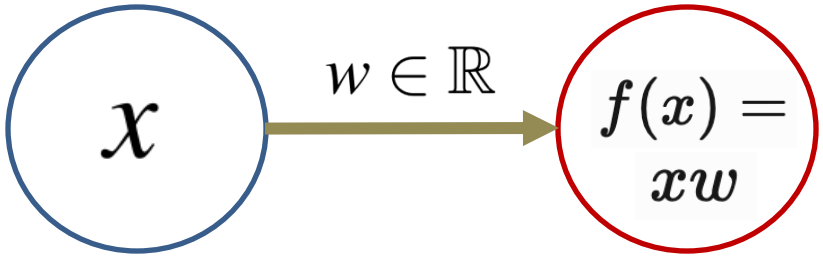




Prediction Error

$$w = w + \alpha(y - xw)$$

Predict Height from Age



age → height

$\alpha \leftarrow 0.01$

$w \leftarrow 0$

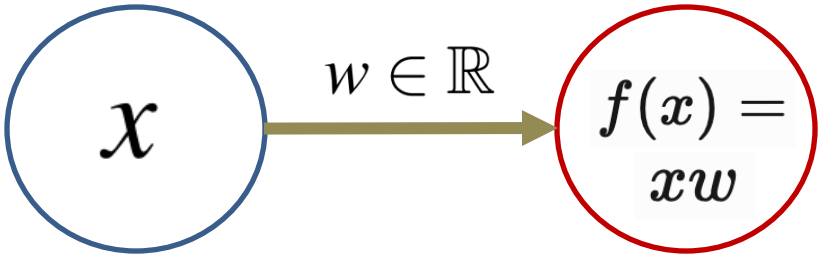
(age=15years, height=100cm) $(x = 15, y = 100)$

$w \leftarrow w + \alpha(100 - 15 \times 0)$

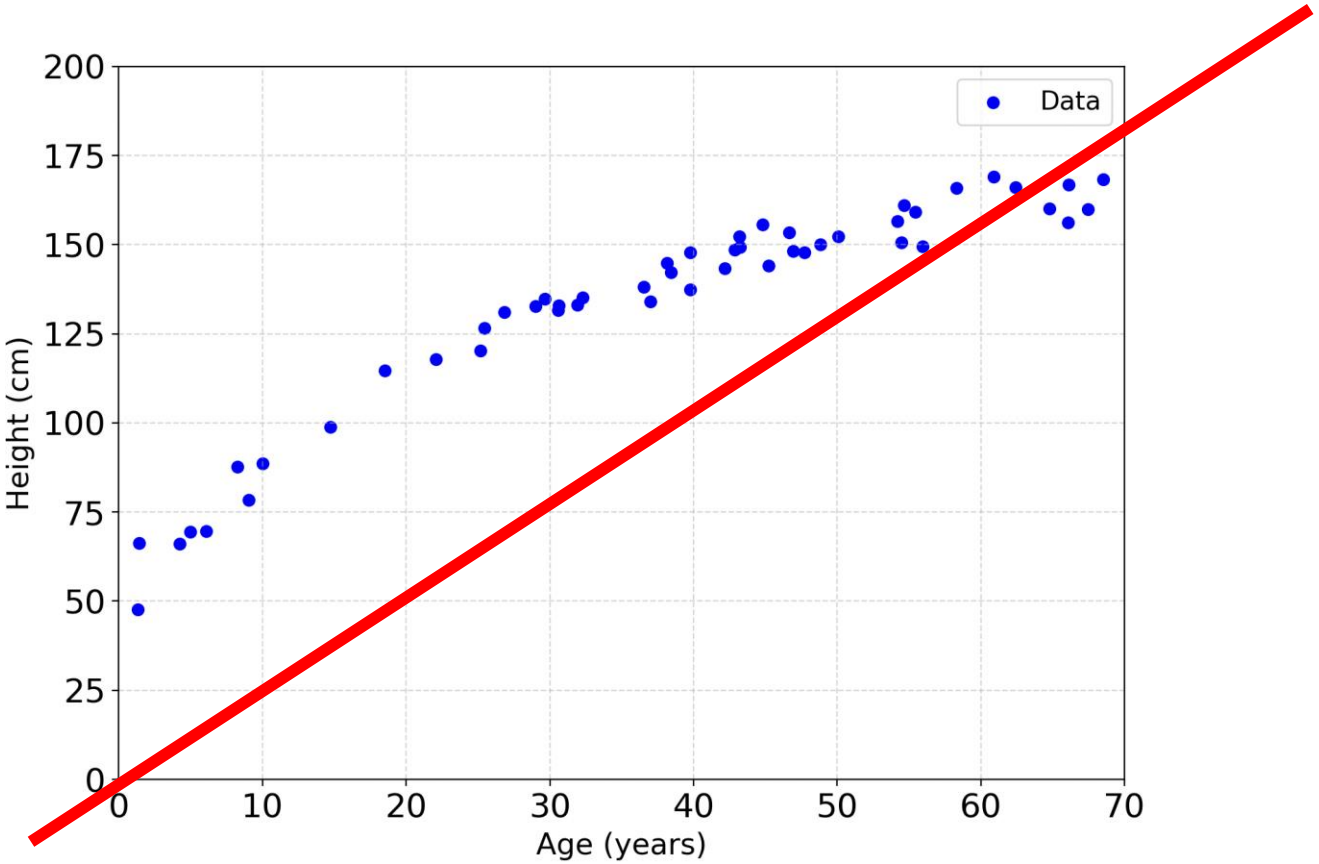
$w \leftarrow w + 0.01 \times 100$

$w \leftarrow w + 1$

Predict Height from Age



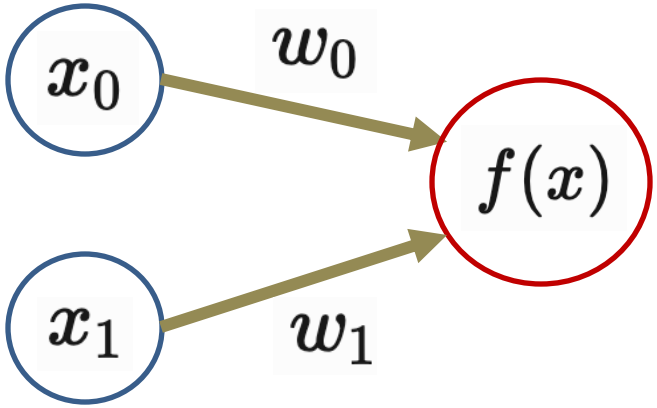
age → height



How can we model Learning?

$$x \in \mathbb{R}^n$$

$$x = [x_0, x_1]$$



$$f(x) = \text{ expression?}$$

$$f(x) = x_0 w_0 + x_1 w_1$$

$$w_0 = w_0 + \alpha(y - f(x))$$

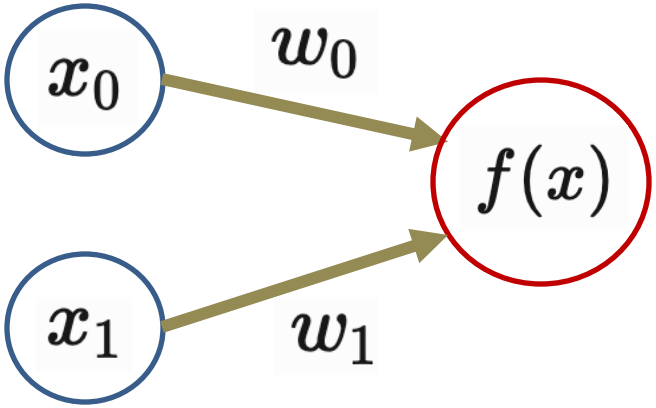
$$w_0 = w_0 + \alpha(y - x_0 w_0 - x_1 w_1)$$

$$w_0 = w_0 + \alpha(y - f(x))x_0$$

$$w_1 = w_1 + \alpha(y - f(x))x_1$$

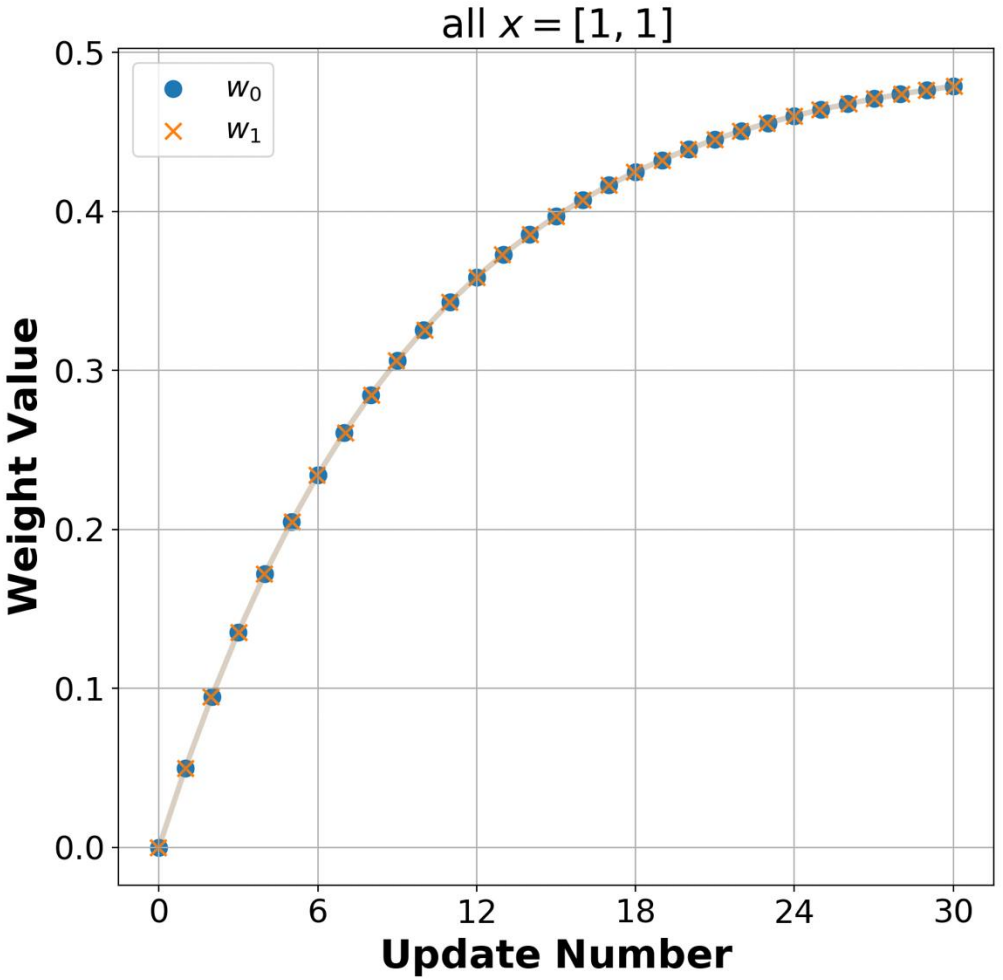
Order of training examples matters

$$x = [x_0, x_1]$$



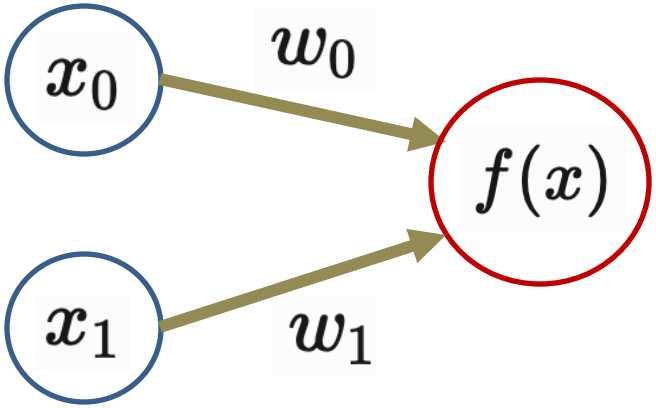
$$D = \{...([1,1],1)...\}$$

x y

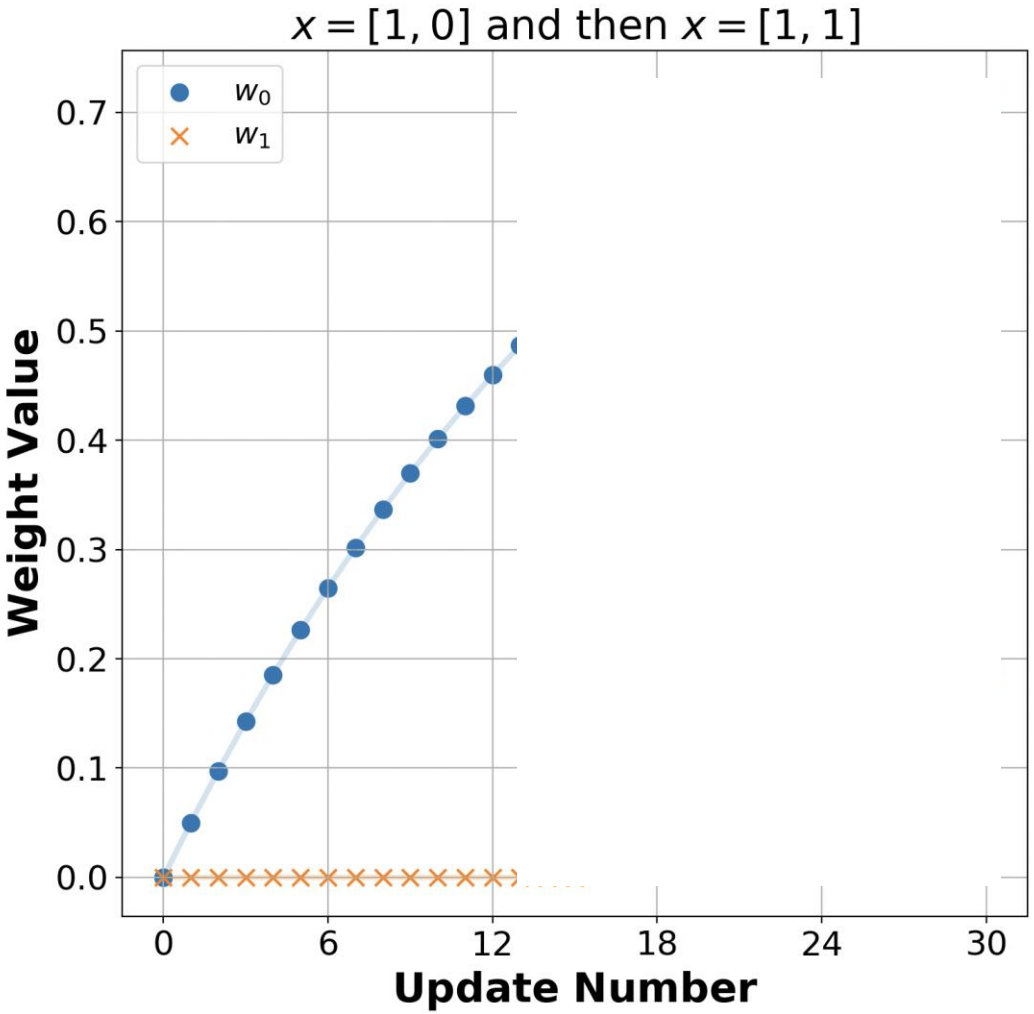


Order of training examples matters

$$x = [x_0, x_1]$$



$$\begin{matrix} x & y & & x & y \end{matrix}$$
$$D = \{([1,0],1)...([0,1],1)\}$$



Forward and Backward pass

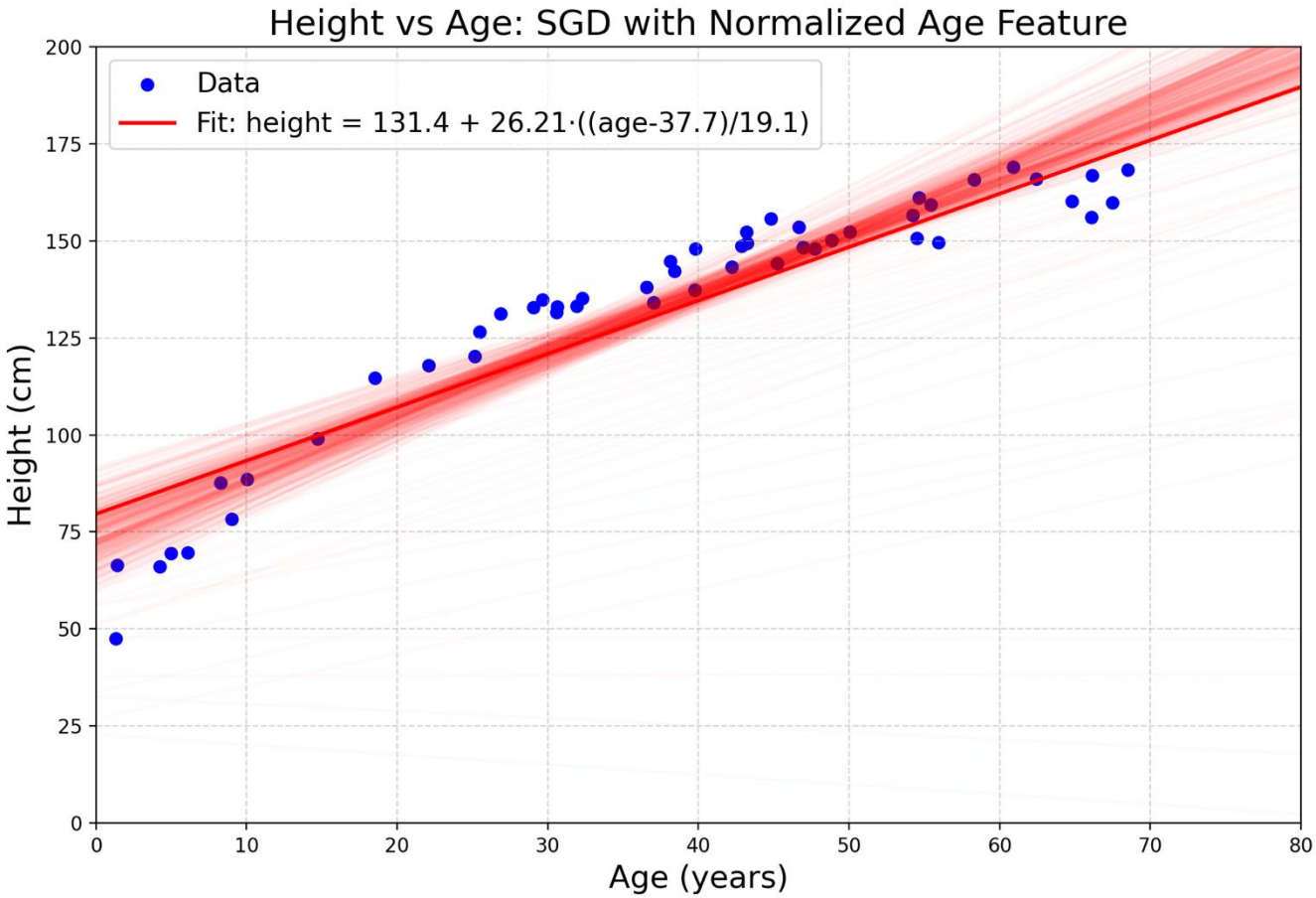
- Lets define two operations:
  - Forward pass: $f(x) = x_0 w_0 + x_1 w_1$
  - Backward update: $w_i = w_i + alpha \cdot (y - f(x)) \cdot x_i$

- Can we deal with any function: $f(x) = y$
  - Think of what happens with $f([0,0])=3$

Can we approximate any function?

- We can introduce the bias term:

$$f(x) = wx + b = w_0 x + b = w_0 x + w_1 1$$

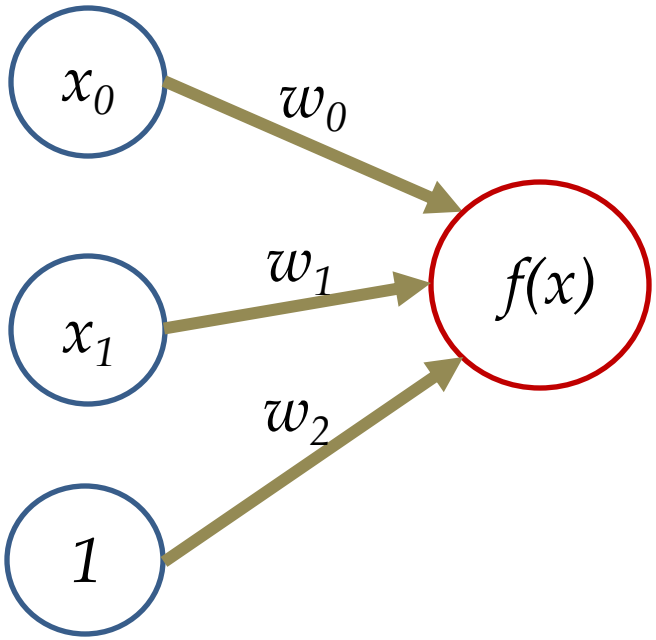


Neural Networks: Adding Bias term

- Add bias term : to cover the case when input is 0

- Prediction: $f(x) = x_0 w_0 + x_1 w_1 + w_2$
- Update: $w_i = w_i + alpha \cdot (y - f(x)) \cdot x_i$

$x = [x_0, x_1, 1]$



Learning Logic Gate functions?

- Lets play with it, can we learn logic gates functions?

Here the training datasets:

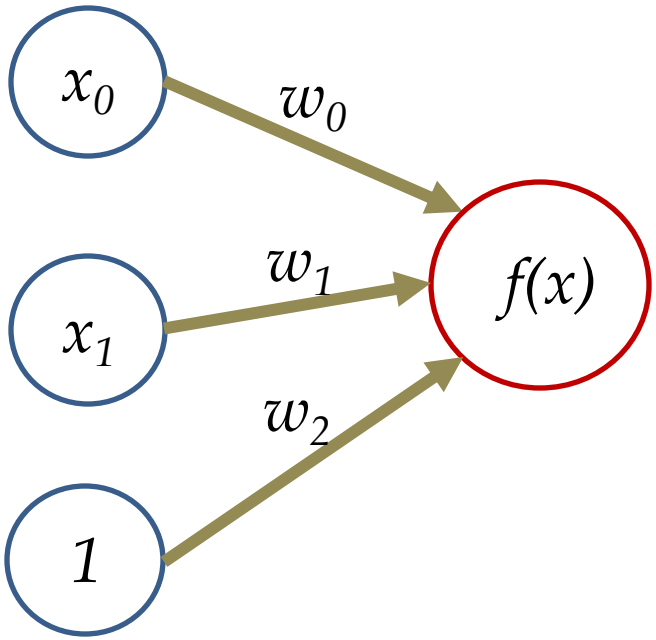
AND

INPUT		OUTPUT
A	B	
0	0	0
1	0	0
0	1	0
1	1	1

OR

INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	1

$x = [x_0, x_1, 1]$



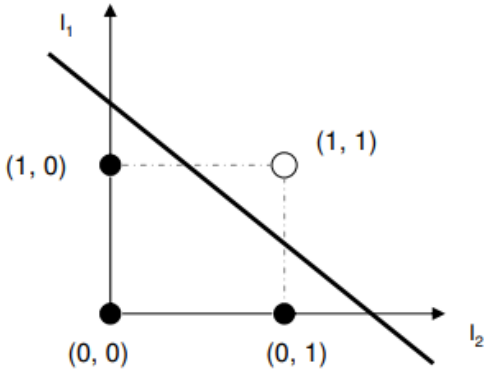
Neural Networks: What about XOR?

- Can we learn logic gates functions?
- What is the interpretation of the learning?
- What happens with XOR gate?

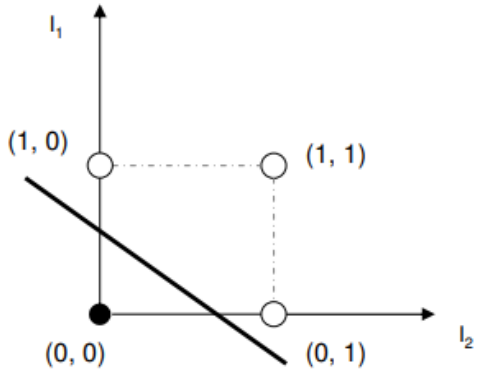
XOR

INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	0

AND		
I_1	I_2	out
0	0	0
0	1	0
1	0	0
1	1	1



OR		
I_1	I_2	out
0	0	0
0	1	1
1	0	1
1	1	1

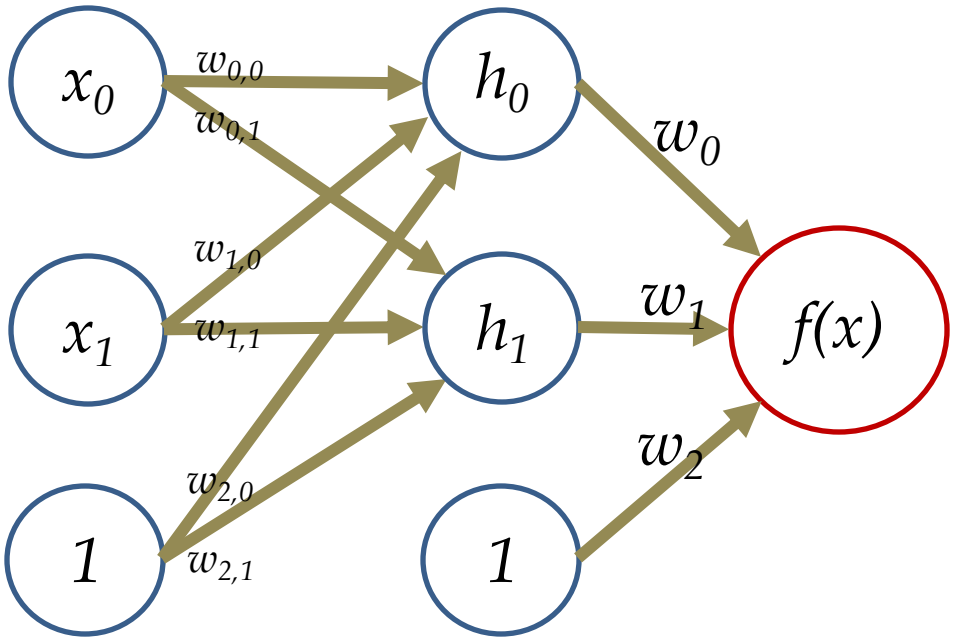


I_2

Neural Networks: Go Multilayer!

- Add layers: Prediction step? And how we do the update rule?

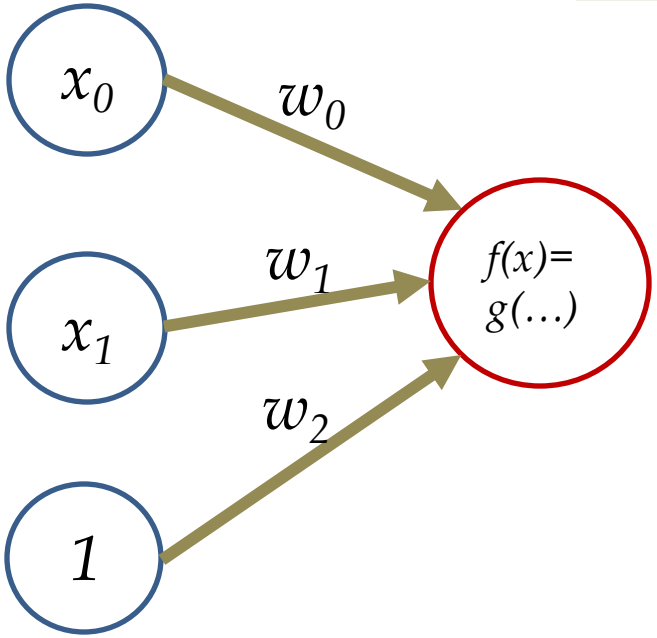
$$x = [x_0, x_1, 1]$$



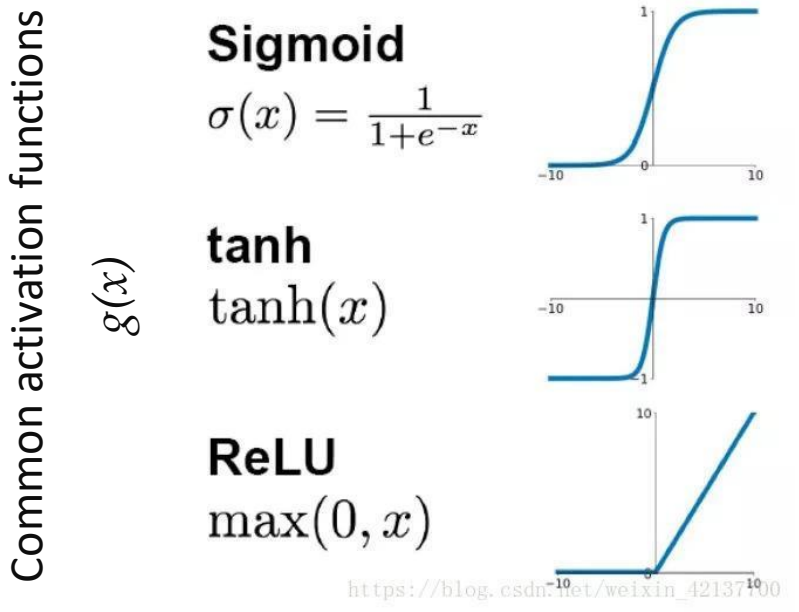
Neural Networks: Add non-linearities

- Any nested combination of linear functions can be re-written as a single linear combination. To make it more powerful let's add non-linearities
- We add an activation function: $g(x)$

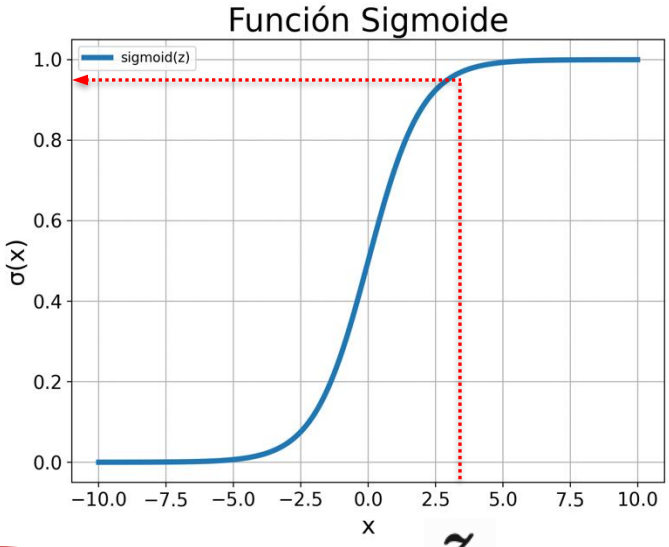
$x = [x_0, x_1, 1]$

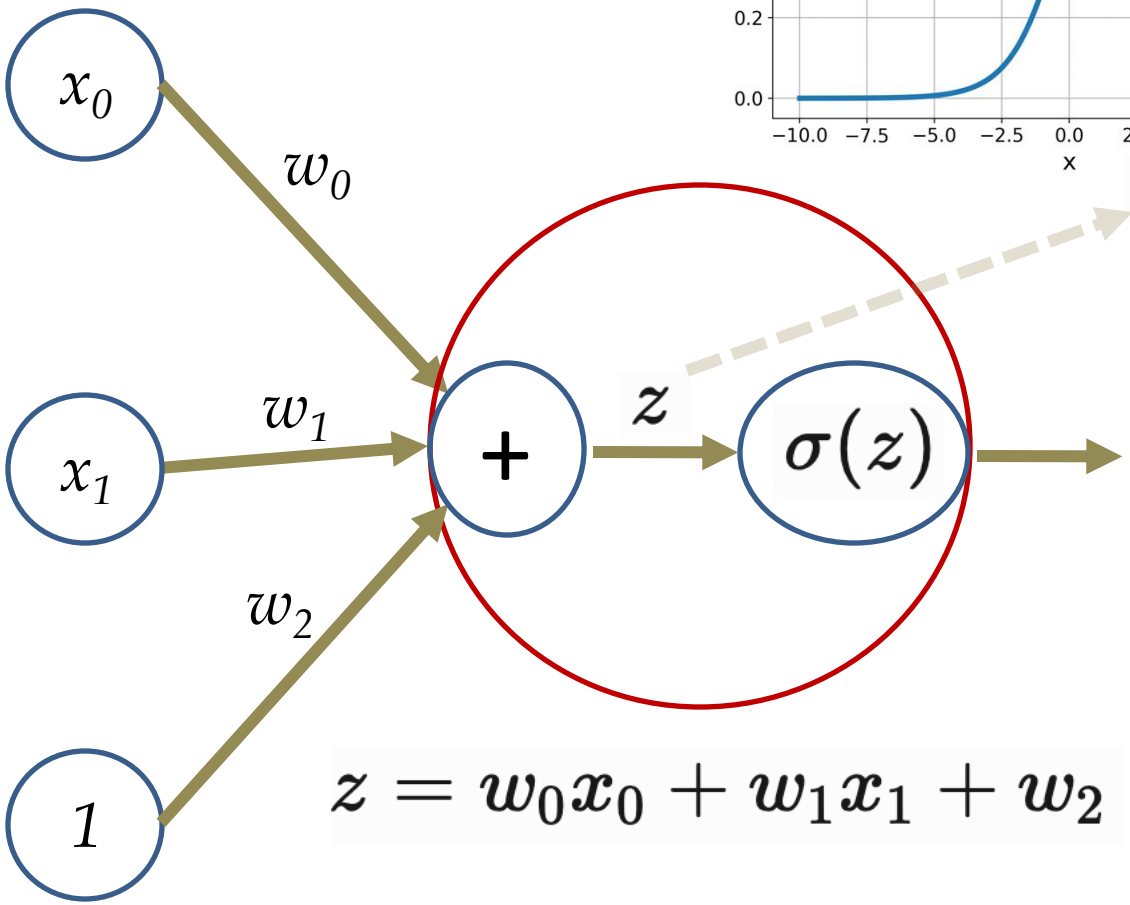


- Prediction: $f(x) = g(x_0w^0 + x_1w_1 + w_2)$
- Update: $w_i = w_i + alpha \cdot (y - f(x)) \cdot x_i$



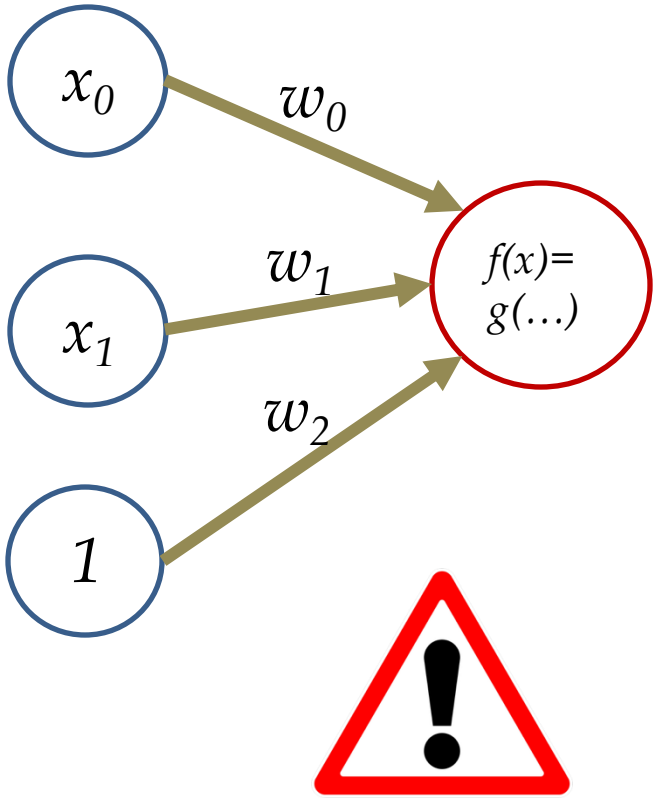
Adding an activation function

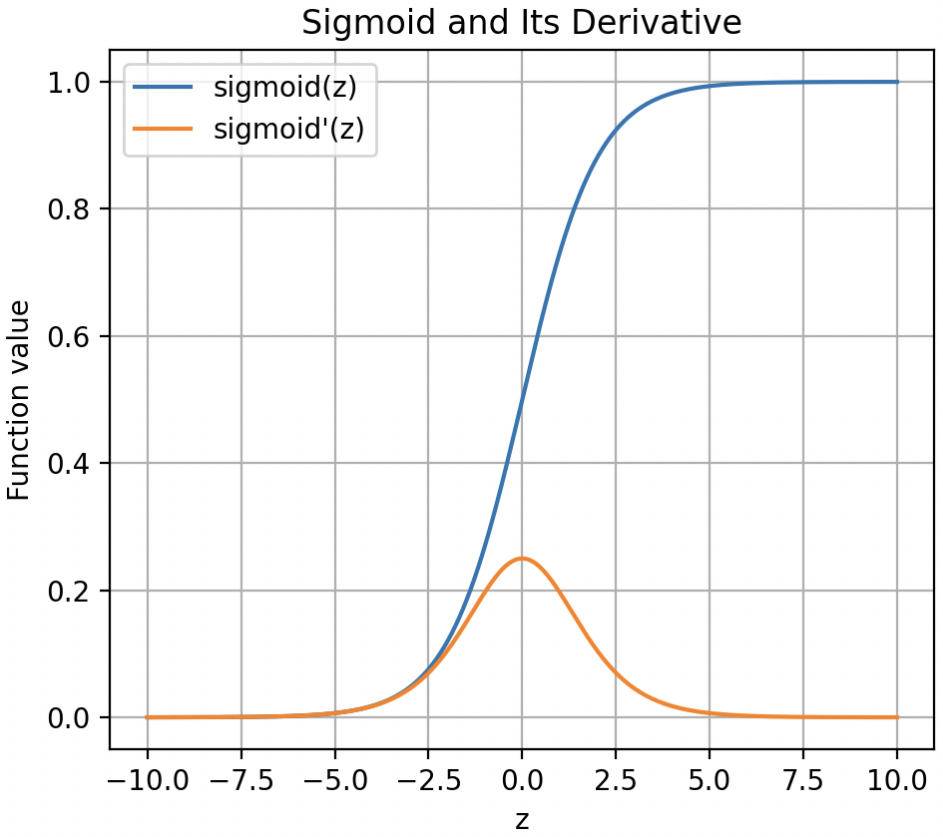




$$z = w_0 x_0 + w_1 x_1 + w_2$$

Neural Networks: Add non-linearities

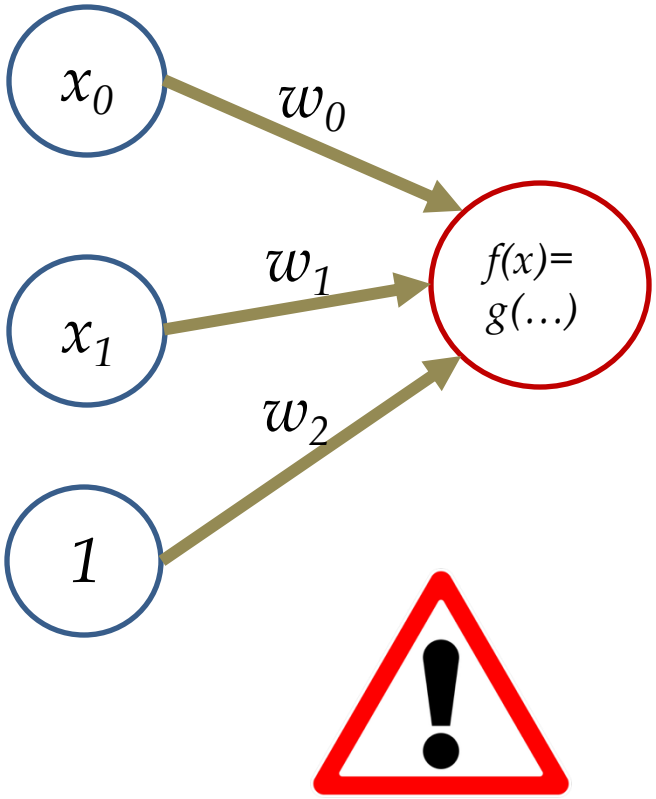


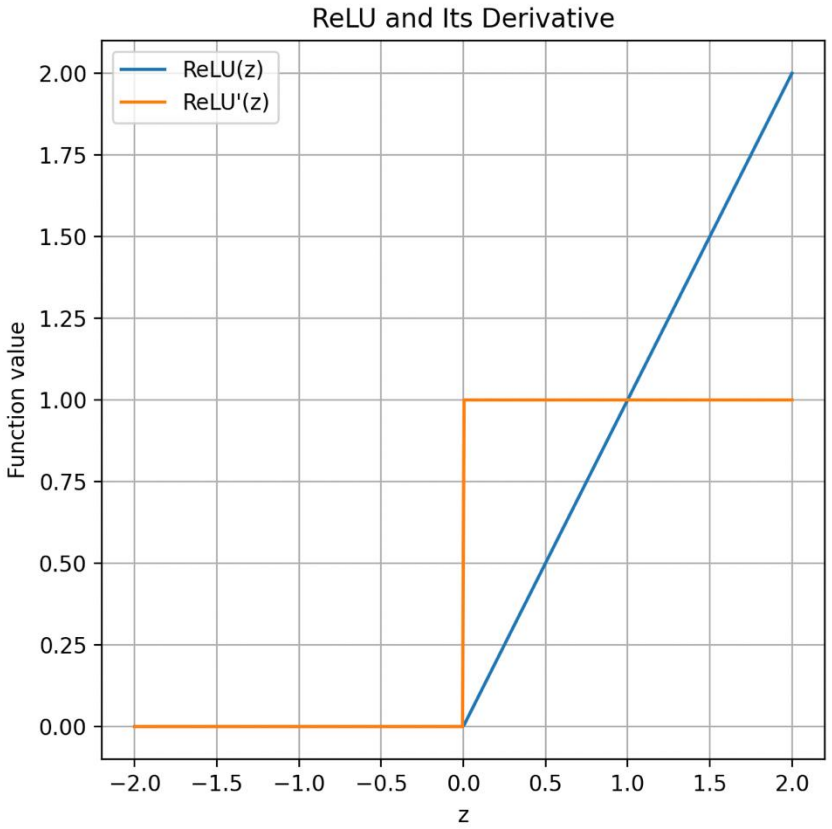


$$f(z) = \sigma(z) = \frac{1}{1+e^{-z}}, \quad f'(z) = \sigma(z)\,(1-\sigma(z)\,) = y(1-y).$$

$$\Delta w_i = \alpha\,(y_{\text{real}} - y)\;y\;(1-y)\;x_i$$

Neural Networks: Add non-linearities



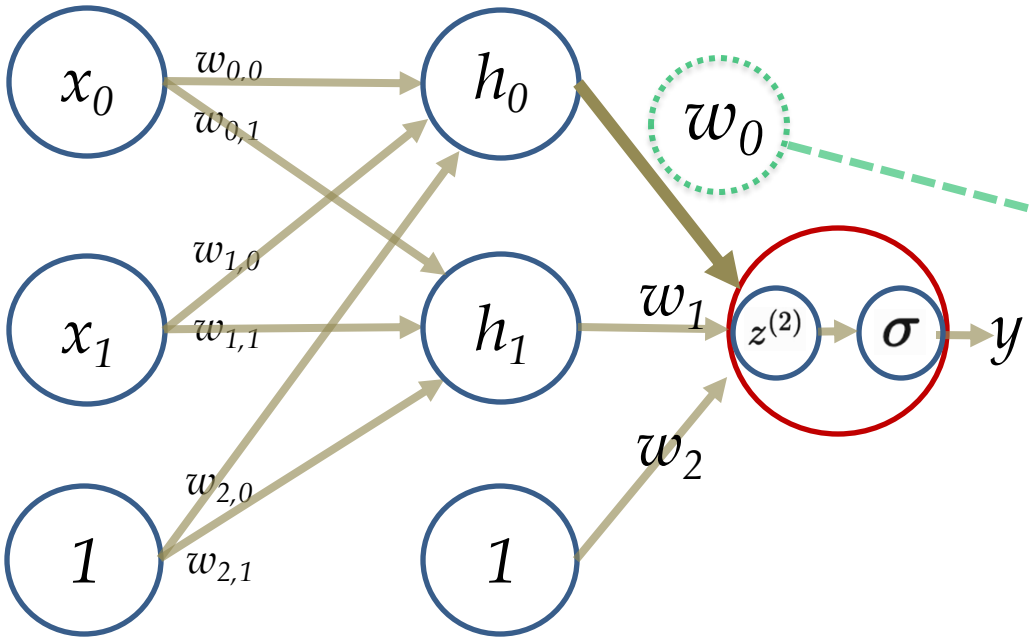


$$f(z) = \max(0, z), \quad f'(z) = \begin{cases} 1, & z > 0 \\ 0, & z \leq 0 \end{cases}$$

$$\Delta w_i = \begin{cases} \alpha \left(y_{\text{real}} - y\right) x_i, & z > 0 \\ 0, & z \leq 0 \end{cases}$$

Neural Networks: Go Multilayer!

- Add layers: Prediction step? And how we do the update rule?



$$L = \frac{1}{2}(y - y_{real})^2$$

$$z^{(2)} = w_0 h_0 + w_1 h_1 + w_2 1$$

$$y = \sigma(z^{(2)})$$

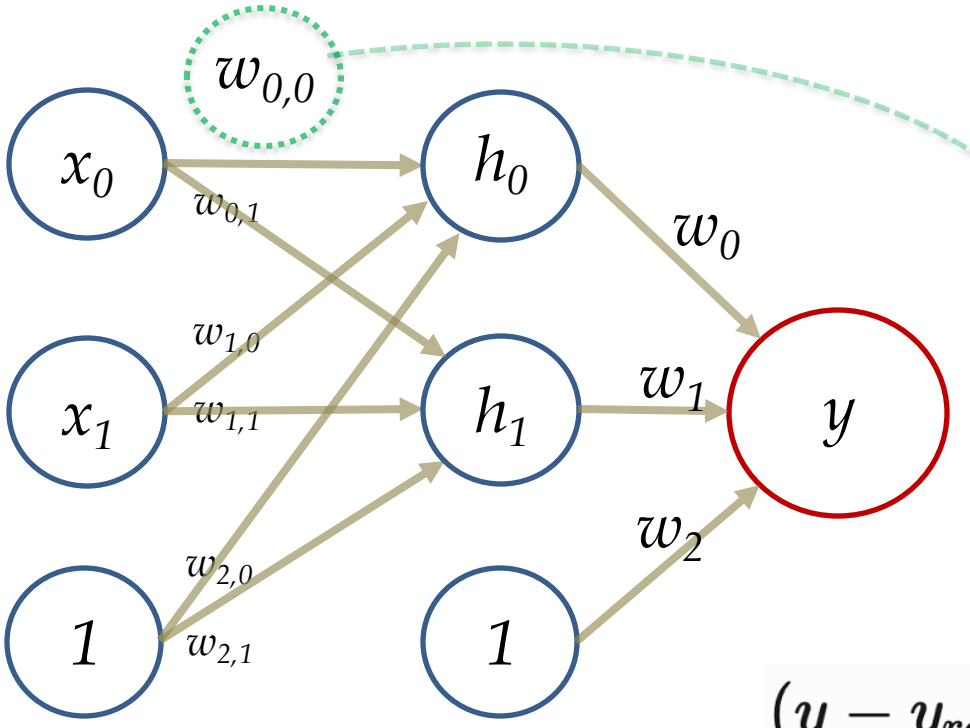
$$\frac{\delta L}{\delta w_0} = \frac{\delta L}{\delta y} \frac{\delta y}{\delta z^{(2)}} \frac{\delta z^{(2)}}{\delta w_0}$$

$$(y - y_{real}) \quad \sigma'(z^{(2)}) \quad h_0$$

$$\Delta w_0 = -\alpha \frac{\delta L}{\delta w_0}$$

Neural Networks: Go Multilayer!

- Add layers: Prediction step? And how we do the update rule?



$$L = \frac{1}{2}(y - y_{real})^2$$

$$\frac{\delta L}{\delta w_0} = \frac{\delta L}{\delta y}\frac{\delta y}{\delta z^{(2)}}\frac{\delta z^{(2)}}{\delta w_0}$$

$$\frac{\delta L}{\delta w_{0,0}} = \frac{\delta L}{\delta y}\frac{\delta y}{\delta z^{(2)}}\frac{\delta z^{(2)}}{\delta h_0}\frac{\delta h_0}{\delta z_0^{(1)}}\frac{\delta z_0^{(1)}}{\delta w_{0,0}}$$

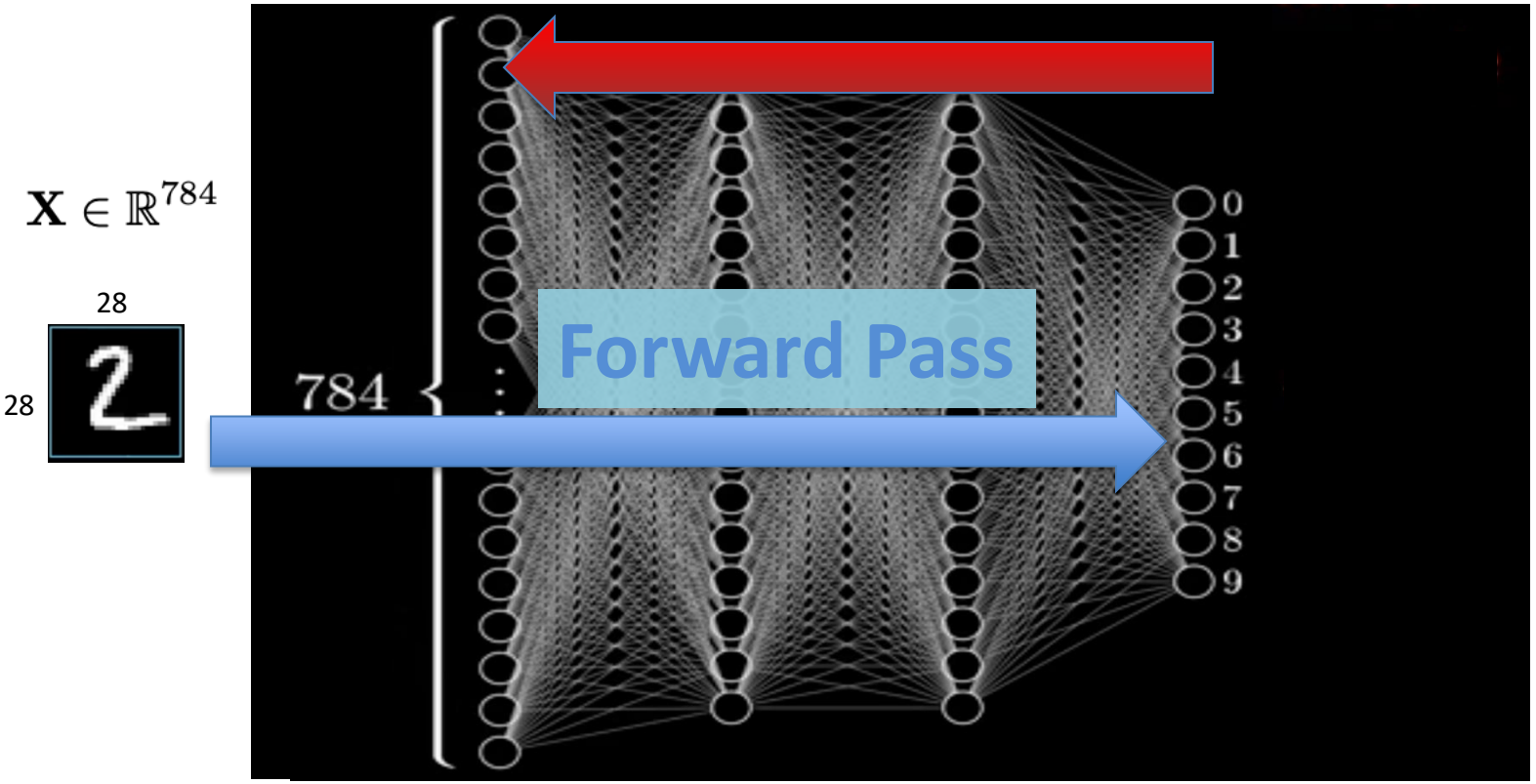
$$(y - y_{real}) \quad \sigma'(z^{(2)}) \quad w_0 \quad \sigma'(z_0^{(1)}) \quad x_0$$

MLP Training

$$F^{MLP}_{\{W_i,b_i\}}(X) = \left(f^{act} \circ f^{L}_{W_L,b_L}\right) \circ \cdots \circ \left(f^{act} \circ f^{L}_{W_1,b_1}\right)(X)$$

$$J(\{W_i,b_i\}_{i=1}^{L}) = \frac{1}{2}\sum_{j=1}^{10}(\hat{y}_j - y_j)^2 \qquad W_i \leftarrow W_i - \eta\frac{\partial J}{\partial W_i}, \quad b_i \leftarrow b_i - \eta\frac{\partial J}{\partial b_i} \quad \text{for } i = 1,\ldots,L.$$

Backward Pass

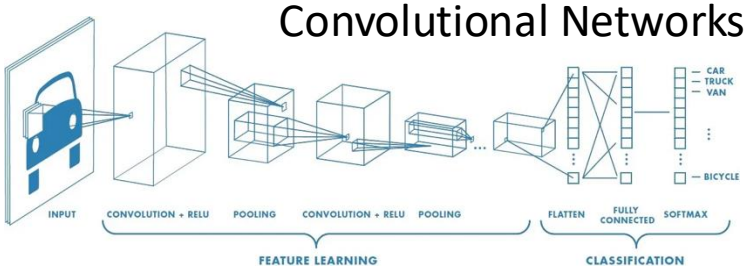


Neural Networks Summary

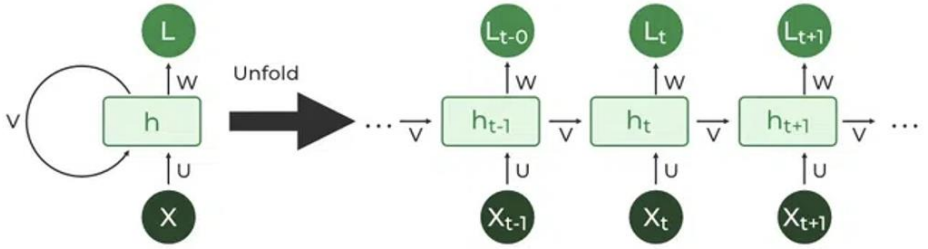
- Activation function why?
- 1 Layer = Perceptron = Linear Projection
- Training: Backpropagation + SGD
  - Chain Rule + Batch Updates
  - Independent Samples Identically Distributed
  - Weight Initialization
- Vanishing Gradient
  - With many layers
  - Depending on activation function
- Let it learn the features

After Multi Layer Perceptrons

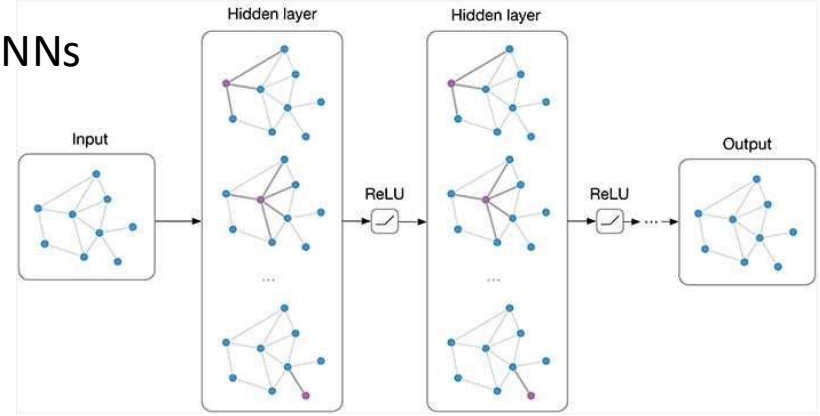
Convolutional Networks



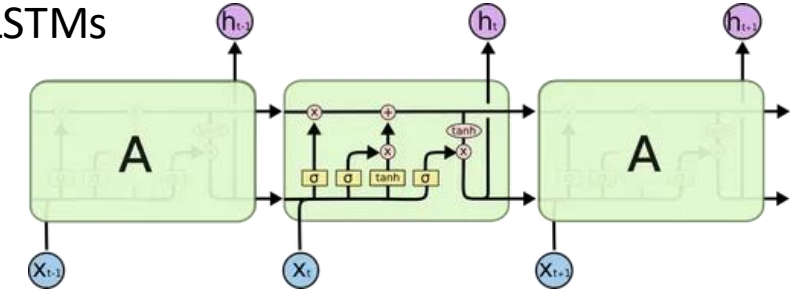
Recurrent Neural Networks



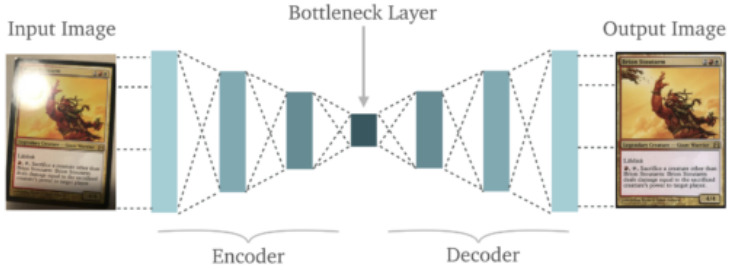
GNNs



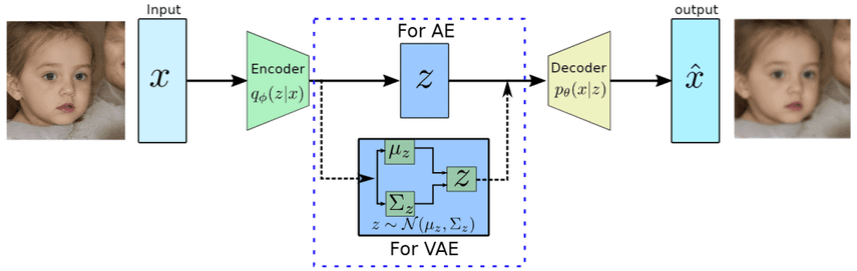
LSTMs



AutoEncoders

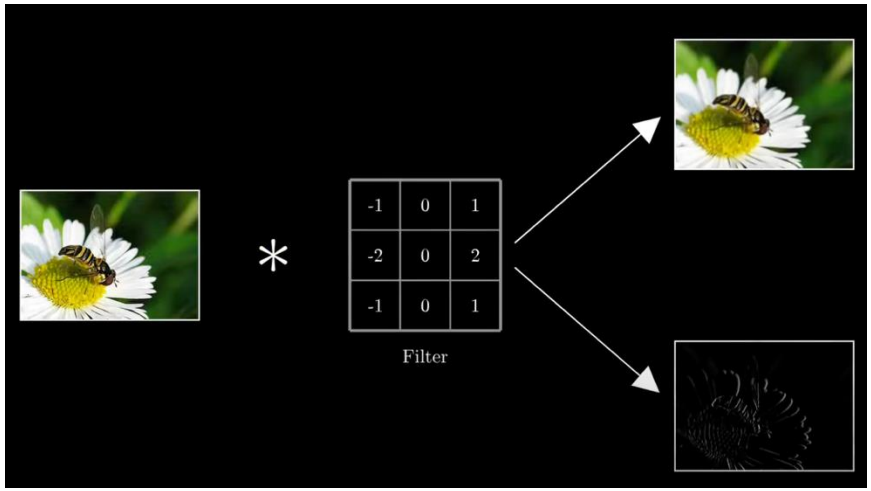


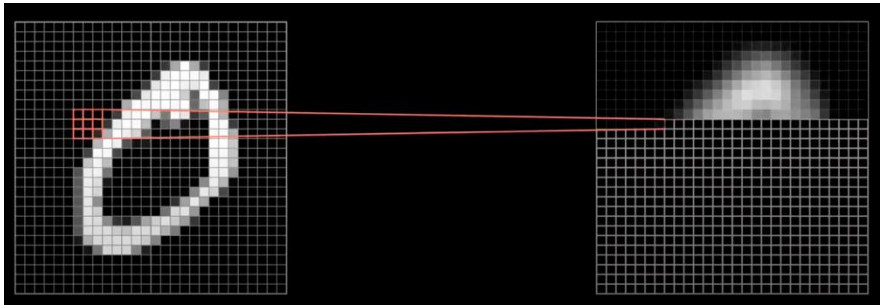
Vaiational AutoEncoders



Convolutional Neural Networks

- A convolutional filter:



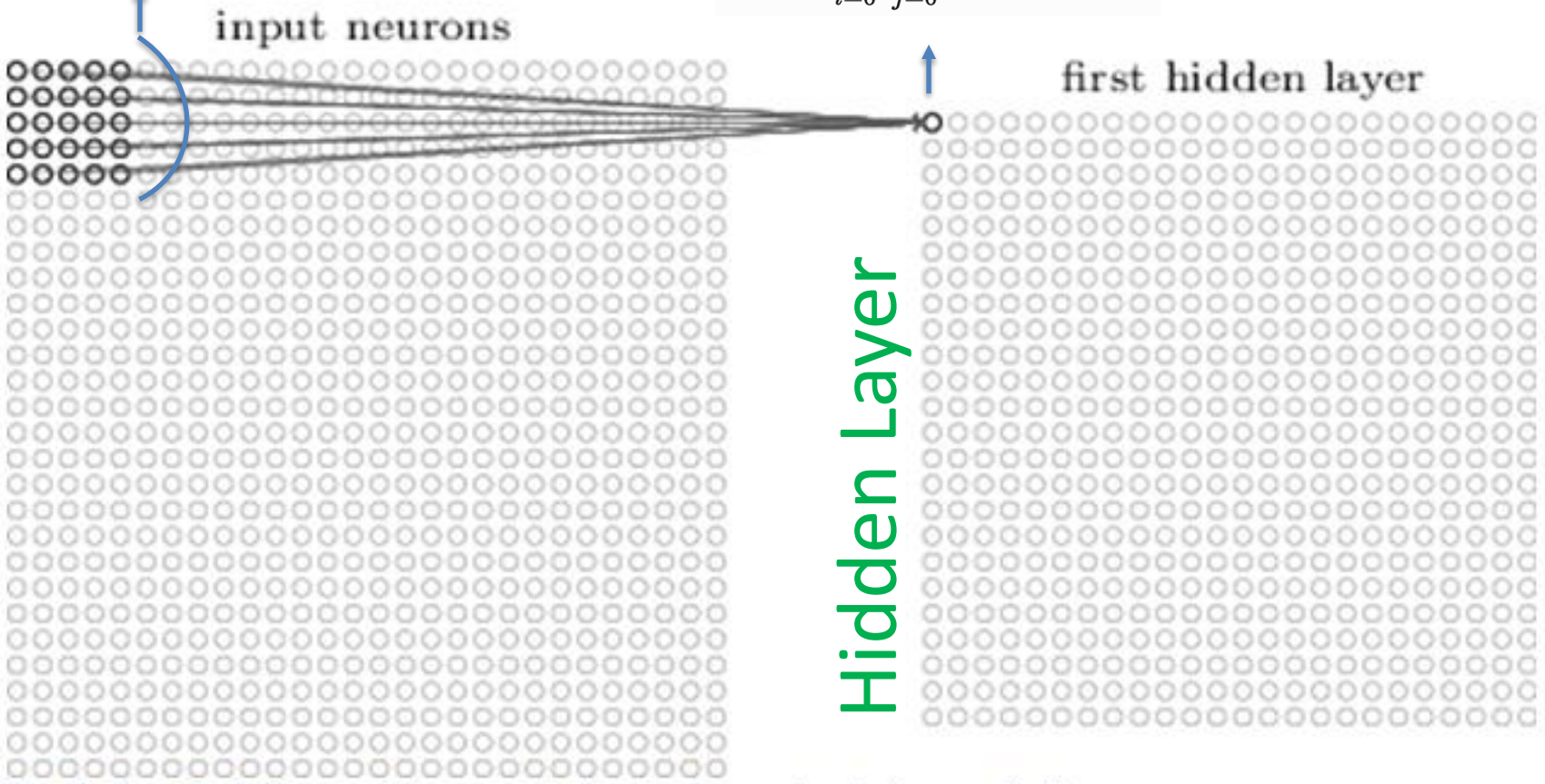


$$(f * k)(i,j) = \sum_{m=0}^{M} \sum_{n=0}^{N} f(i,j) \cdot k(i-m, j-n)$$

Convolutional Neural Networks

$$\begin{pmatrix} w_{00} & w_{01} & w_{02} & w_{03} & w_{04} \\ w_{10} & w_{11} & w_{12} & w_{13} & w_{14} \\ w_{20} & w_{21} & w_{22} & w_{23} & w_{24} \\ w_{30} & w_{31} & w_{32} & w_{33} & w_{34} \\ w_{40} & w_{41} & w_{42} & w_{43} & w_{44} \end{pmatrix}$$

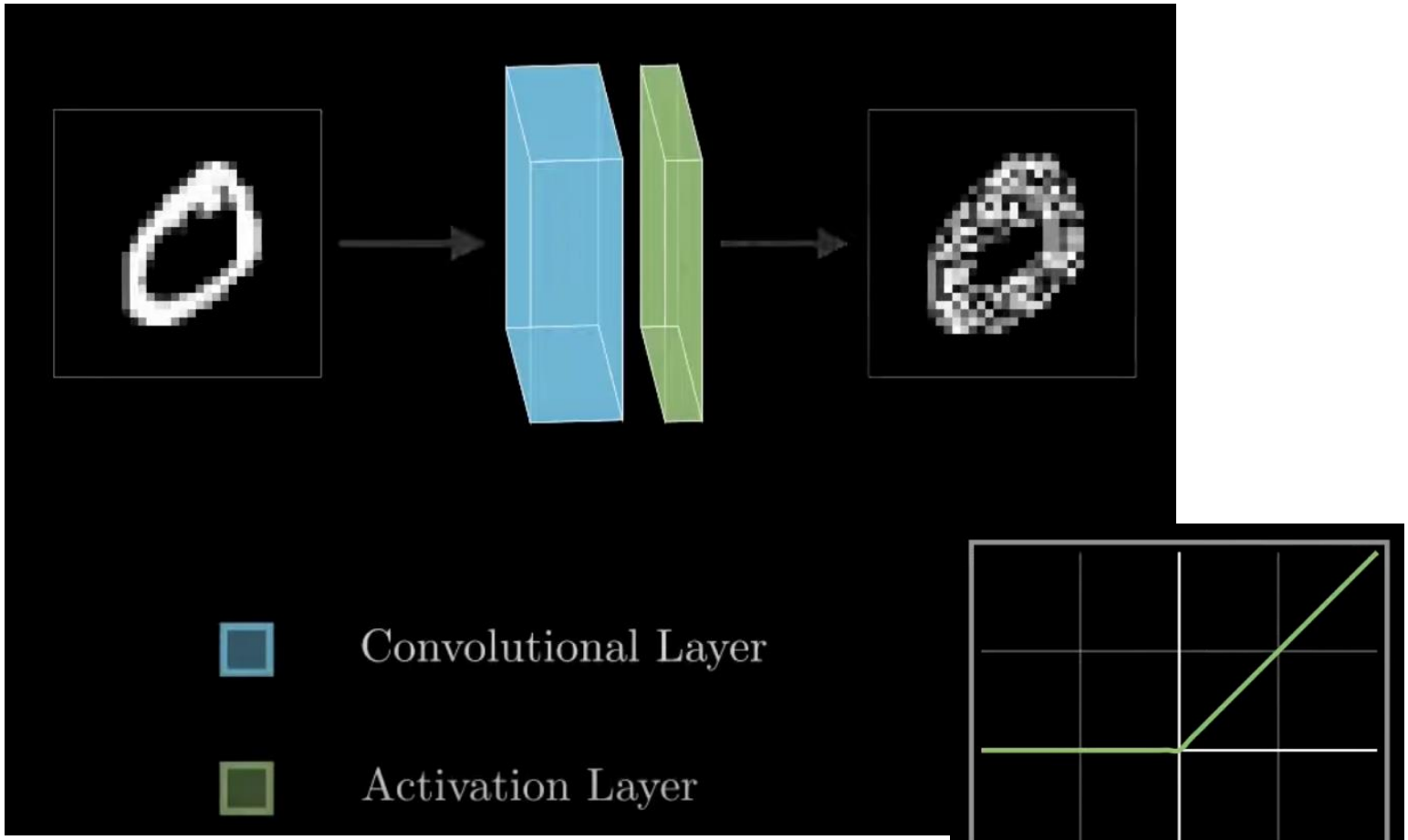
$$h_{0,0} = \sum_{i=0}^{4} \sum_{j=0}^{4} w_{ij} \, X_{i,j} \; + \; b$$



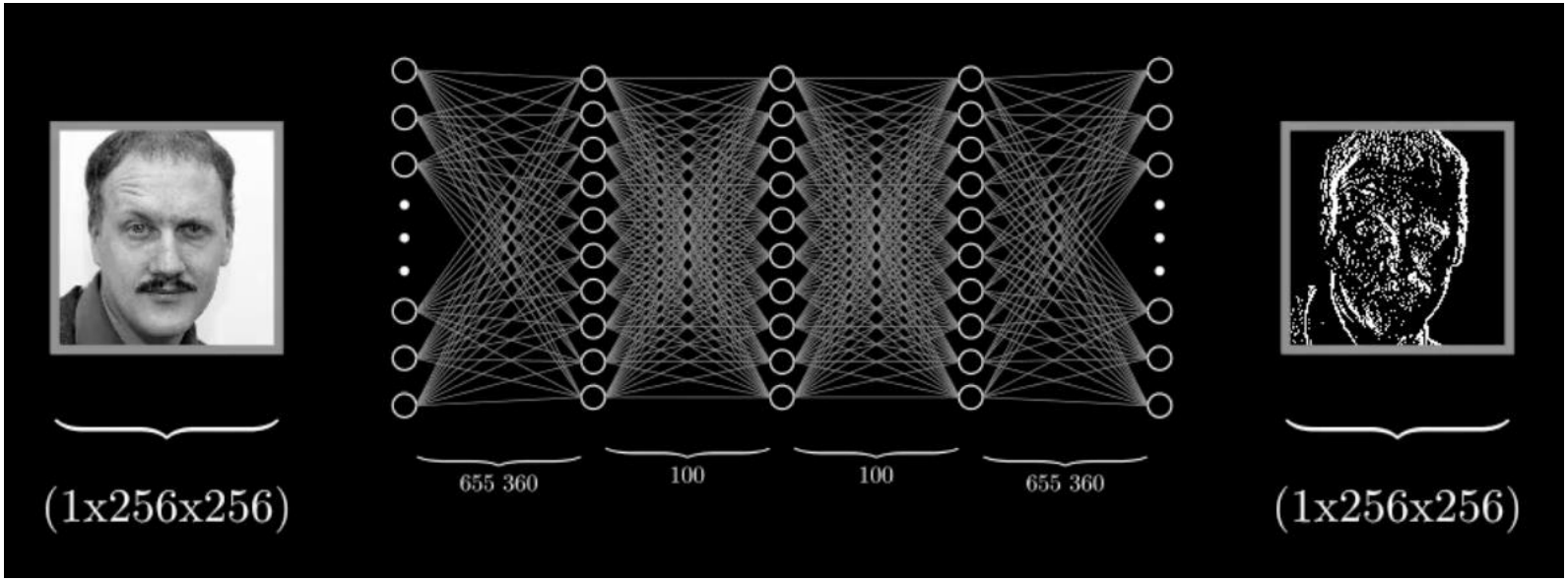
Visualization of 5 x 5 filter convolving around an input volume and producing an activation map

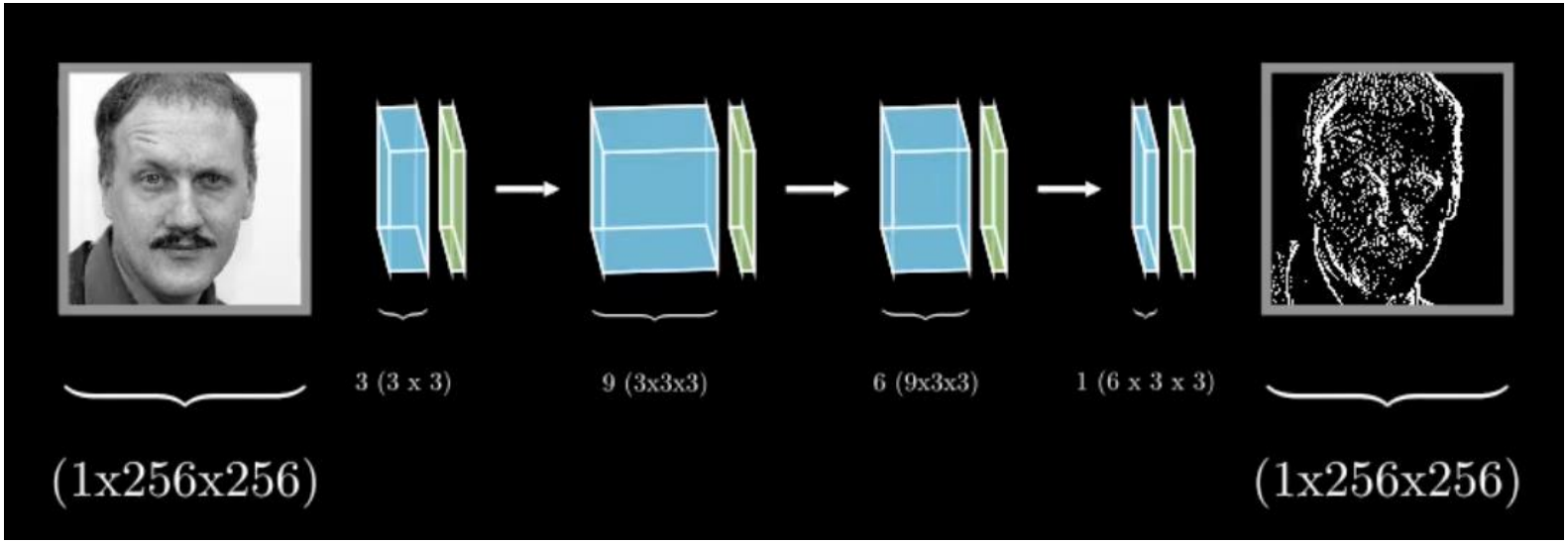
Only 16 weights!!!

Convolutional Neural Networks

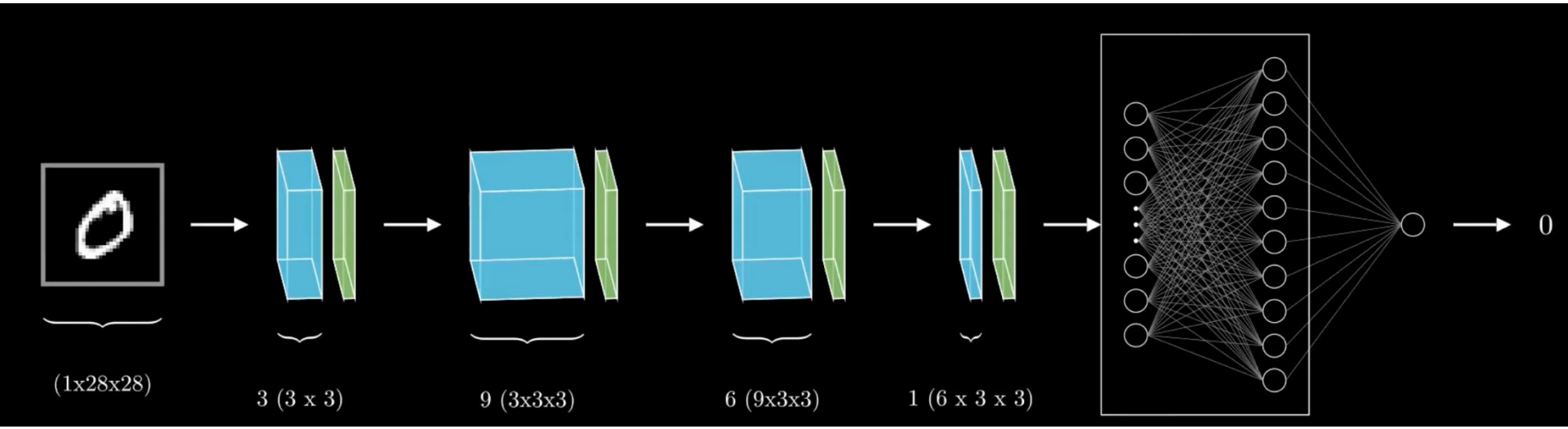


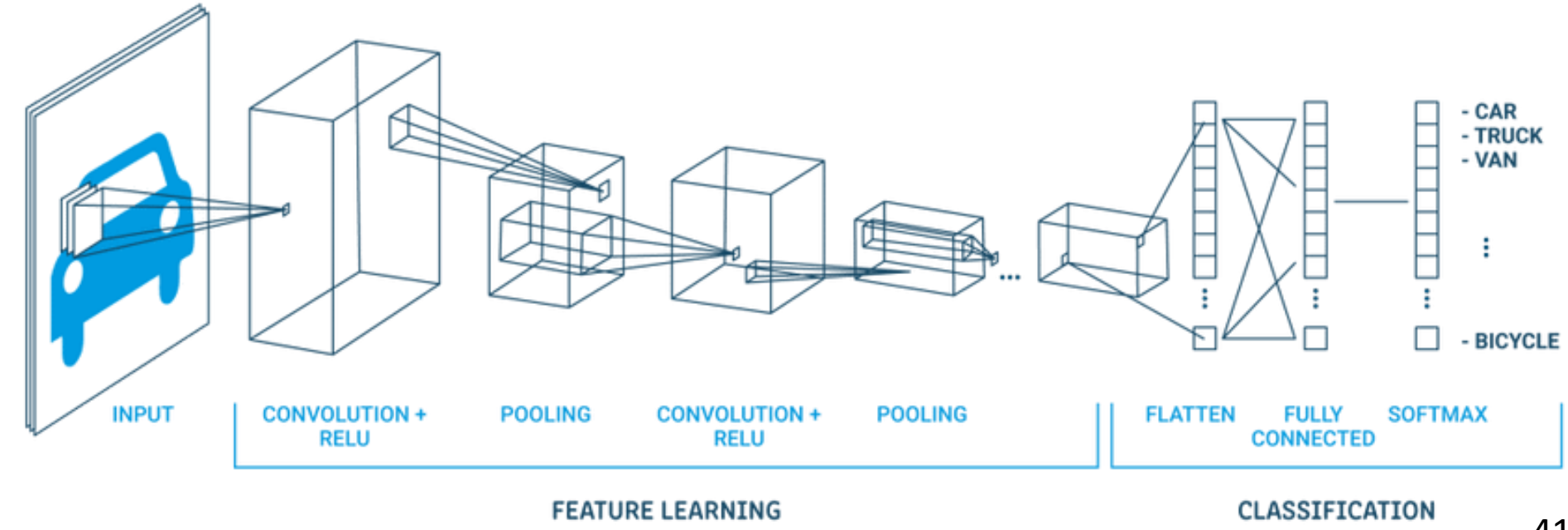
Convolutional Neural Networks





Convolutional Neural Networks





Convolutional Neural Networks

Escuela AI HUB Sobre IA y Educación de verano
aihub.csic.es

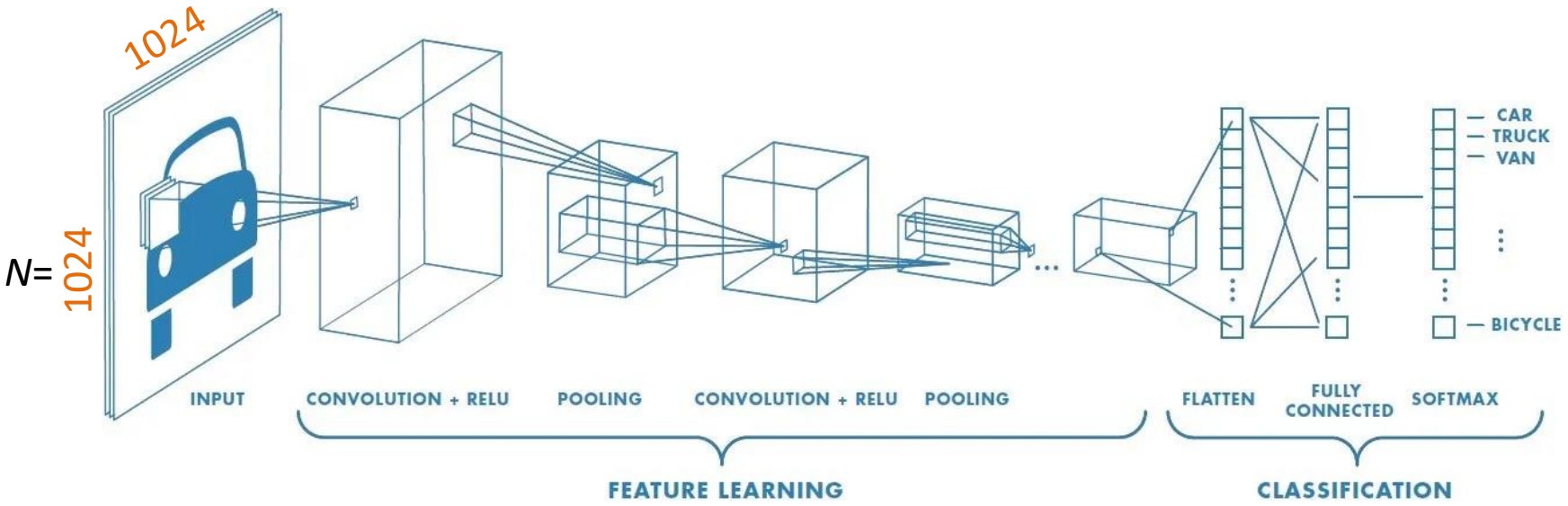
- Convolutional Layer
  - Backprop
    - Each layer has weights:
      - k filters of size k^2

Type	Number of Parameters (approx.)
Convolutional layer	k^3, e.g., for $k = 3$, $3^3 = 27$ parameters
Fully-connected layer	N^4, e.g., $1024^4 \approx 10^{12}$ parameters

Convolutional Network
ConvNet Backpropagation

1979 ConvNet Invented
2012 AlexNet ImageNet solved
Weight Sharing - 2D Input
Inductive Biass: Assumes neighbor pixels are the only relevant

AlexNet 2012 62M ImageNet Accuracy 84.7%
ResNet 2015 60M ImageNet Accuracy 95.5%



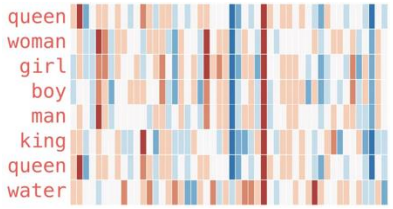
From Convnets to Transformers

Escuela AI HUB Sobre IA y Educación de verano
aihub.csic.es

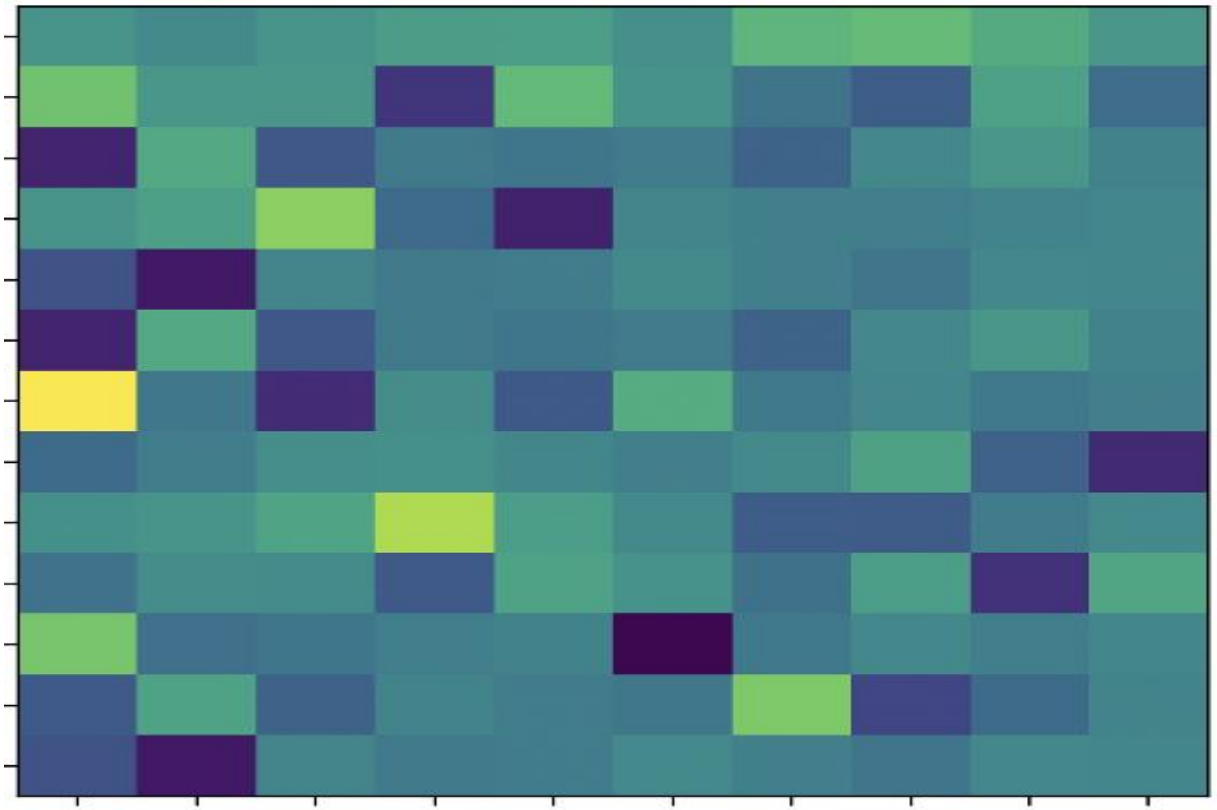
- How we represent words and phrases?

Word2Vec: Embeddings

2013

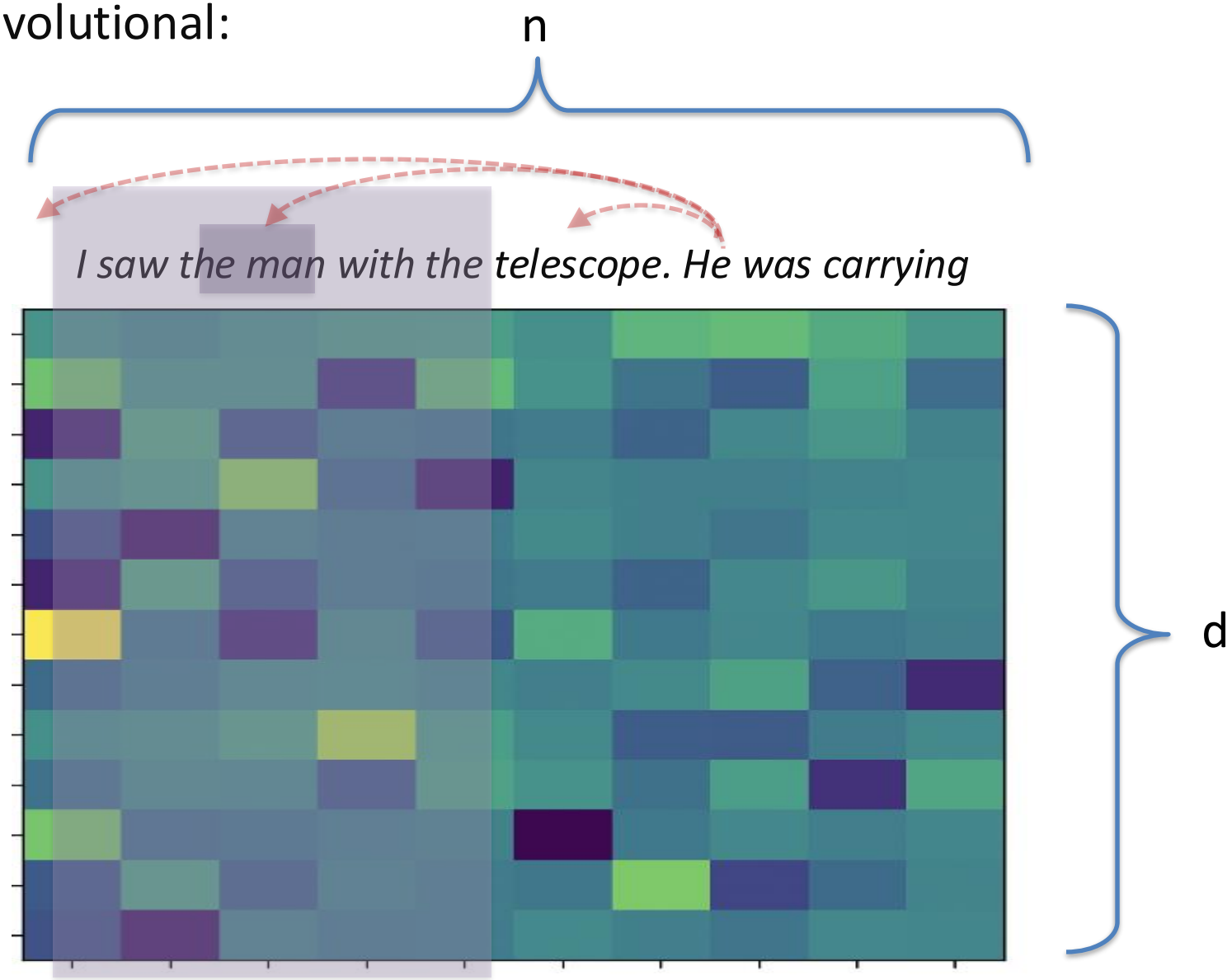


I saw the man with the telescope. He was carrying



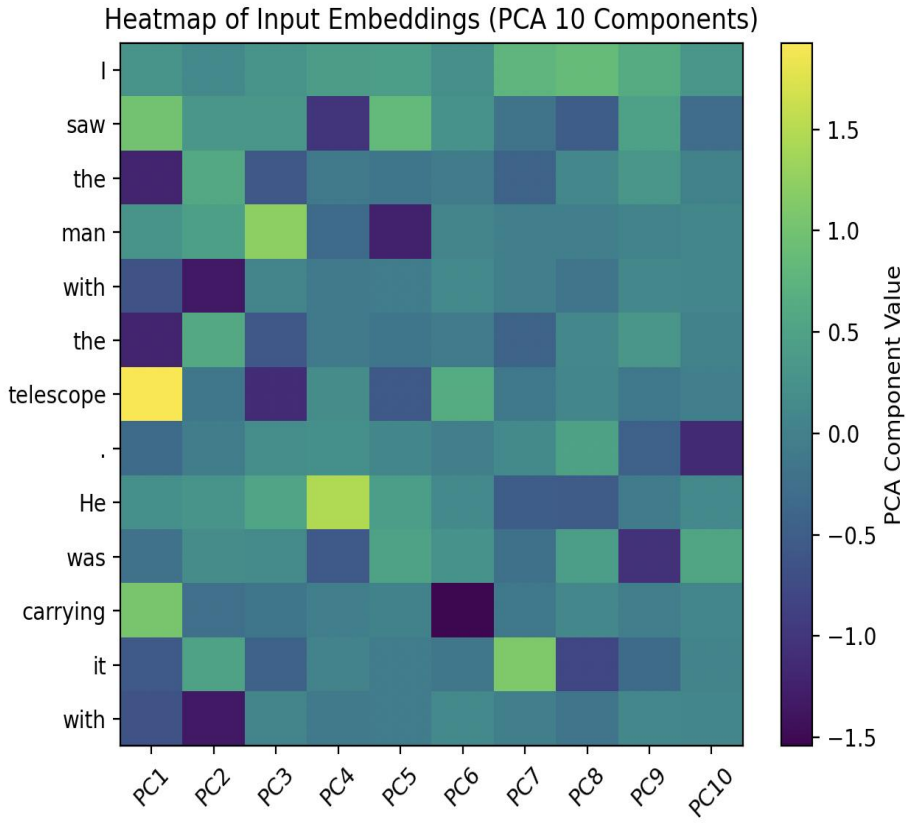
Attention Layer

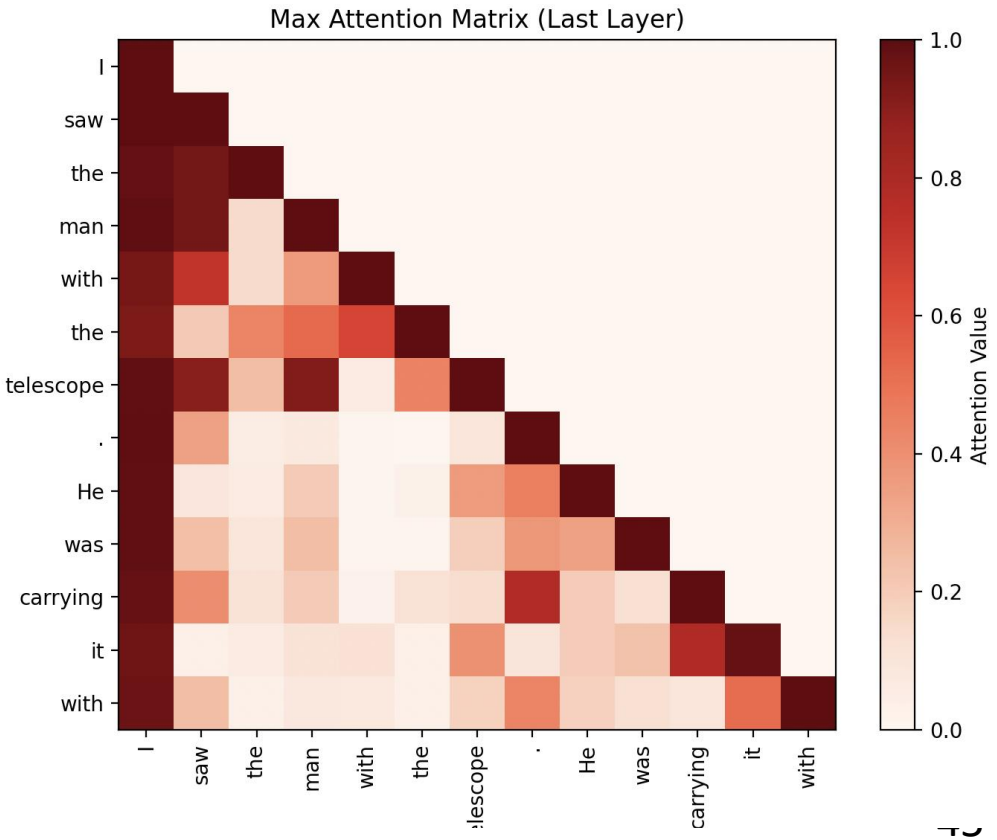
- In Between Fully Connected and Convolutional:



- A **scoring function** of all pairs of input tokens

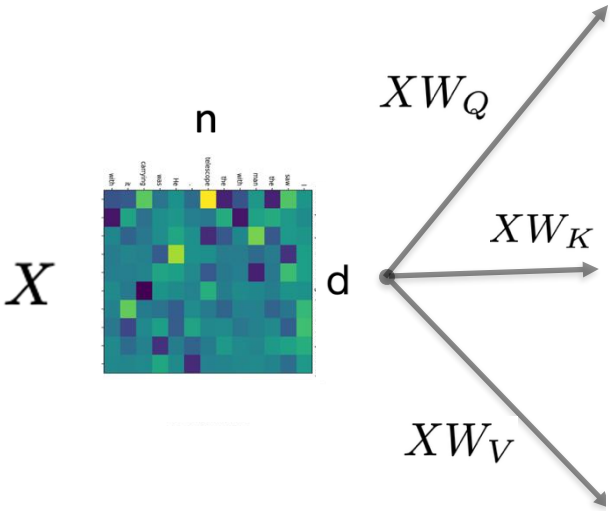
I saw the man with the telescope. He was carrying





Attention Layer

QK^T



$\mathbb{R}^{n \times d}$ → $\mathbb{R}^{n \times d}$

Transformer Architecture

- Which functions it contains?

Normalization Layers

Fully Connected Layer MLP

Softmax
Linear projections,
matrix multiplication

Addition

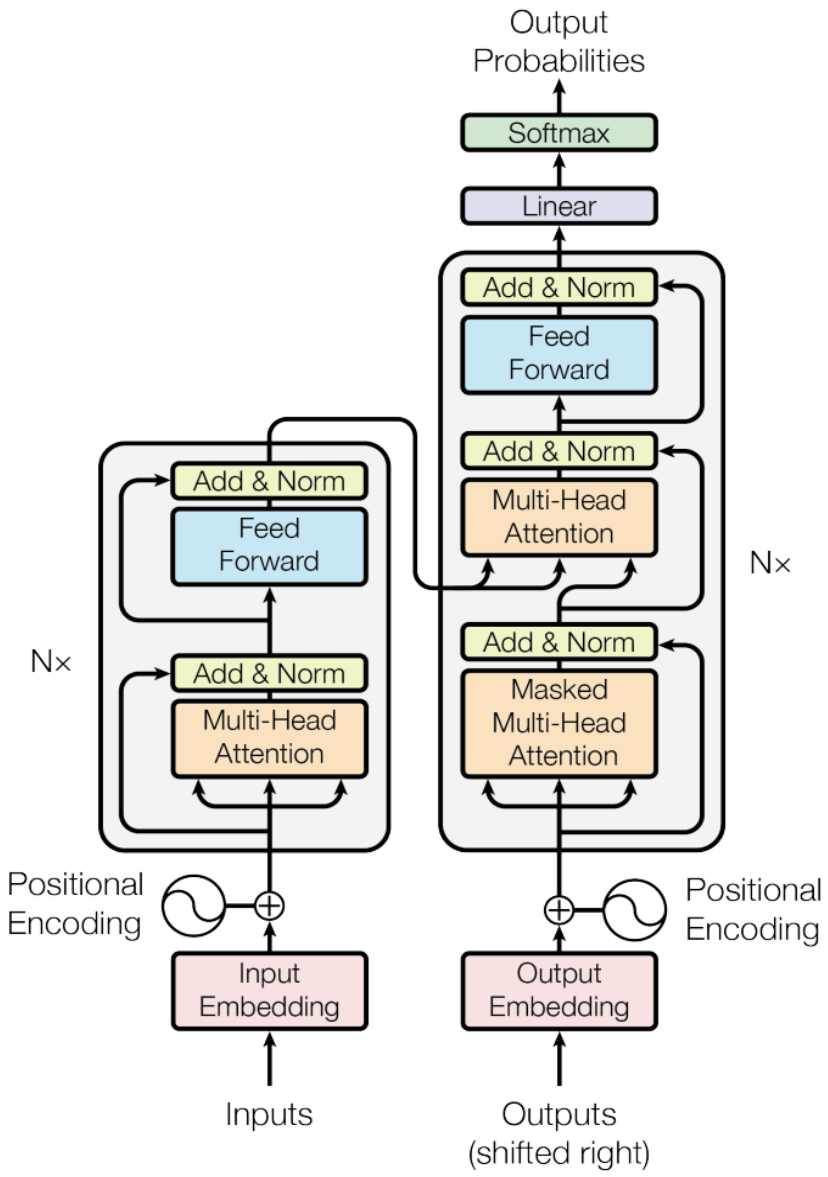
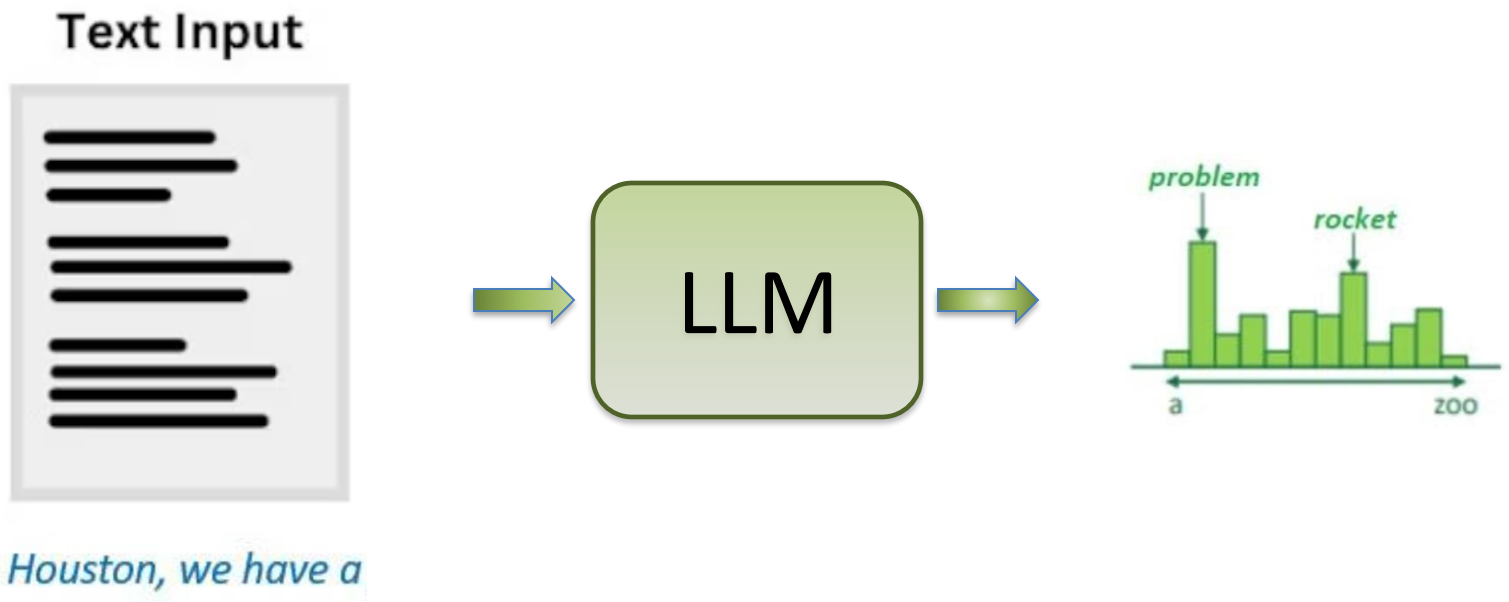
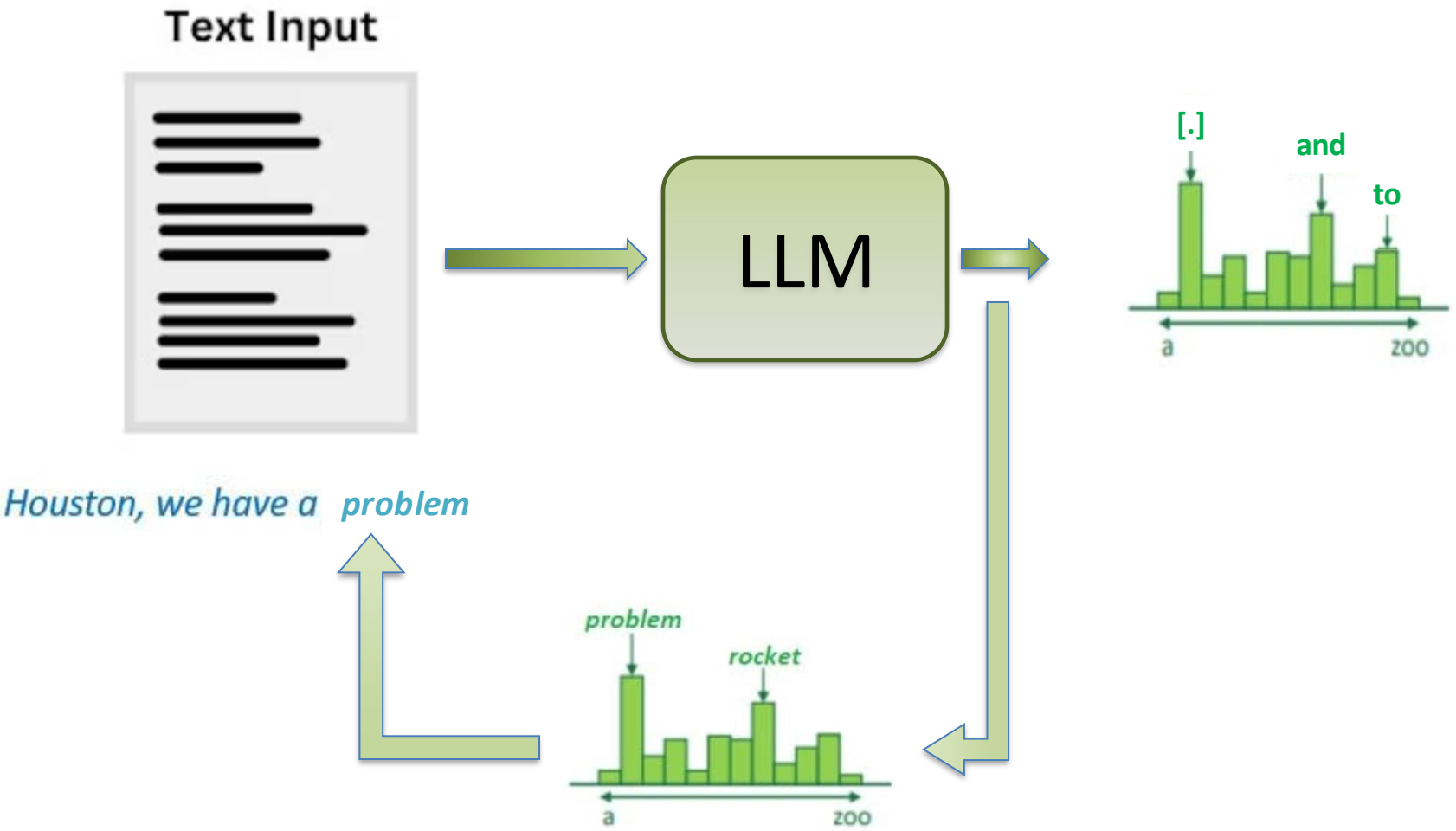


Figure 1: The Transformer - model architecture.

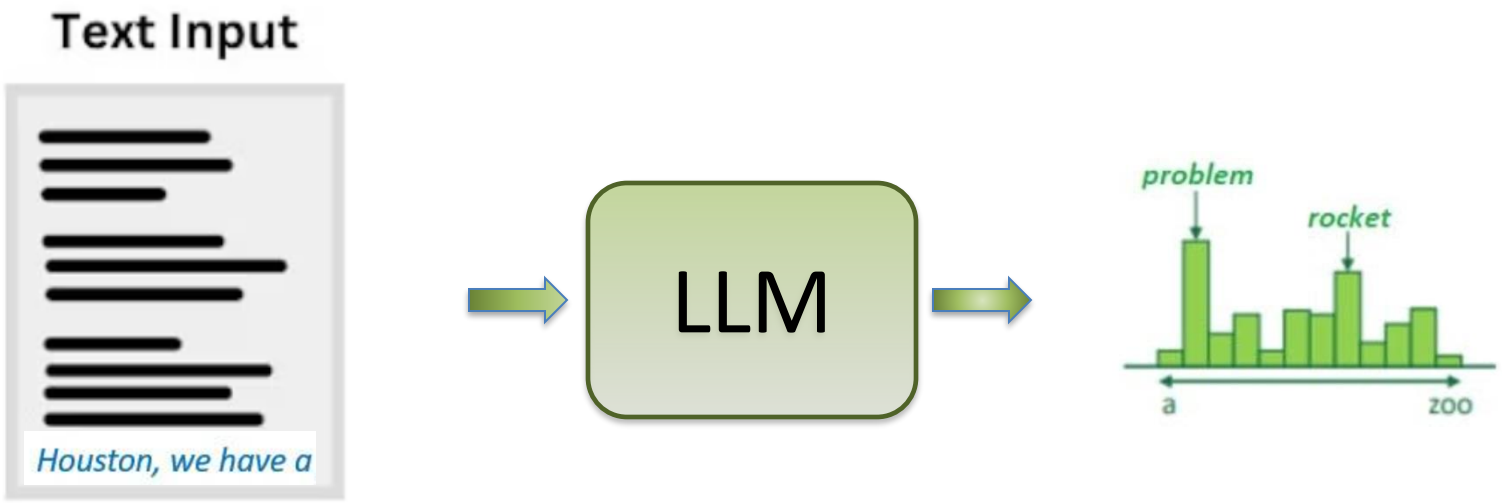
What is an LLM?



LLM autoregresive Inference



LLMs: Tokenization



$$[i_{\text{Houston}},\ i_{\text{we}},\ i_{\text{have}},\ i_{\text{a}}] \in \mathbb{N}^4$$

$$E \in \mathbb{R}^{|V| \times d} \qquad x_j \in \mathbb{R}^d = \begin{cases} E[i_j], & \text{if } 1 \leq j \leq 4, \\ E[\texttt{[PAD]}], & \text{if } 5 \leq j \leq n. \end{cases}$$

$$X = \left[E[i_{\text{Houston}}]; E[i_{\text{we}}]; E[i_{\text{have}}]; E[i_{\text{a}}]; E[i_{\text{pad}}]\right] \in \mathbb{R}^{5 \times d}$$

LLMs : Parallel Processing

$$F_{d,n}^{LLM} : \mathbb{R}^{n \times d} \rightarrow \mathbb{R}^{n \times d}$$

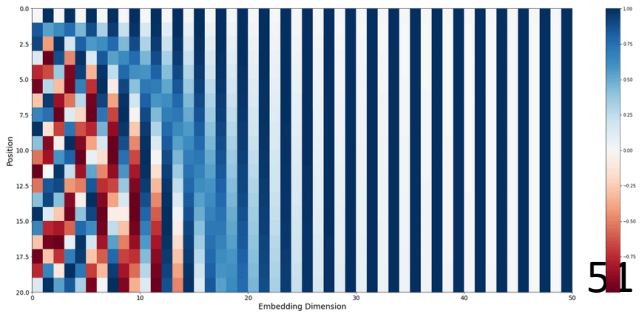
$$X \rightarrow \boxed{\text{LLM}} \rightarrow \mathbf{y} = F_{d,n}^{LLM}(X)$$

$$X = \begin{bmatrix} x_1 + p_1 \\ x_2 + p_2 \\ \vdots \\ x_n + p_n \end{bmatrix} \in \mathbb{R}^{n \times d}$$

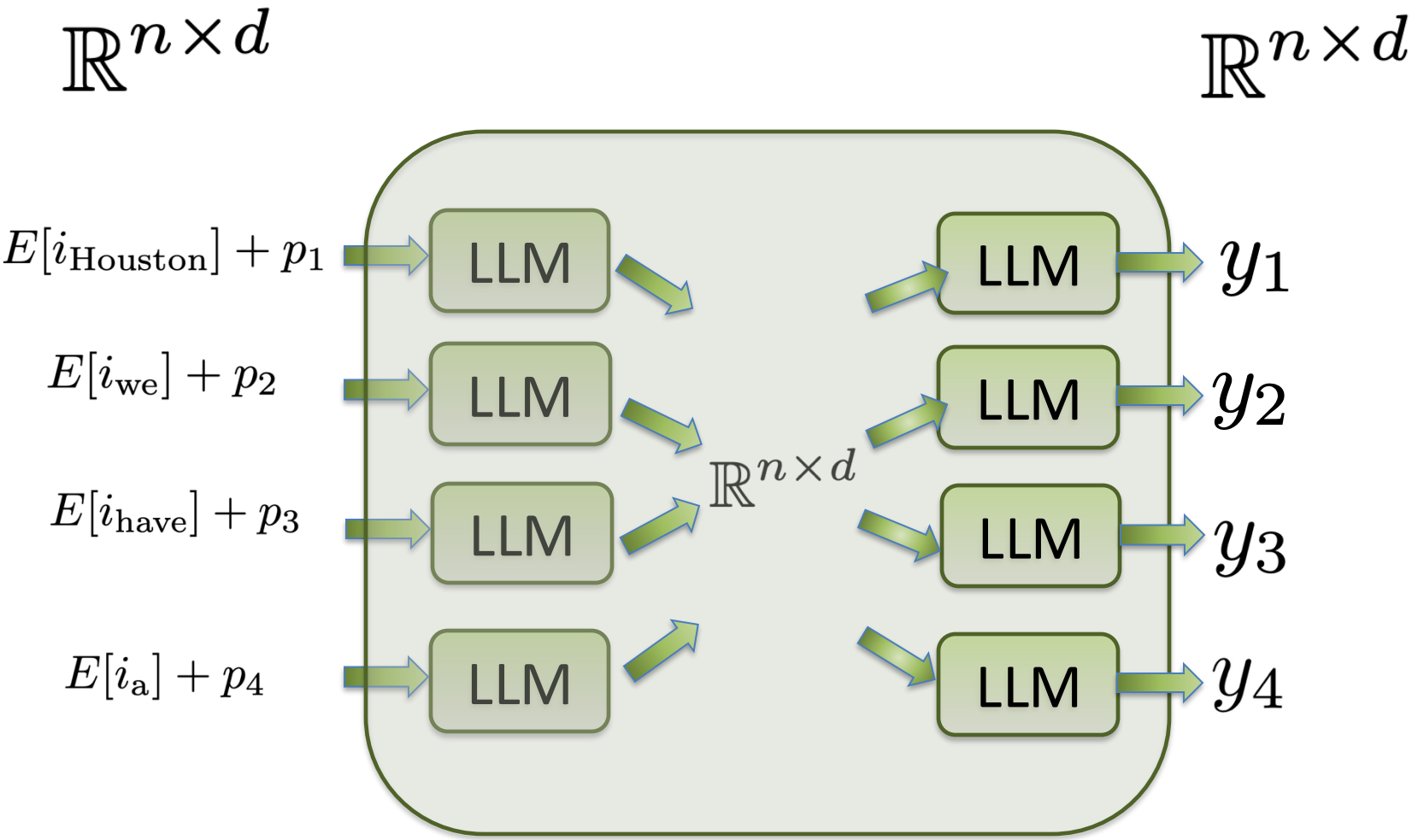
X $\mathbb{R}^{n \times d}$

Houston	.21	.01	.002	0	.1	0	.01	.17	.11	.001
we	0	.01	.17	.11	.001	.21	.01	.002	0	.1
have	0.3	.01	.002	0	.1	.01	.17	.11	.001	.21
a	.01	.02	0	.1	.01	.17	.11	.01	.1	.2

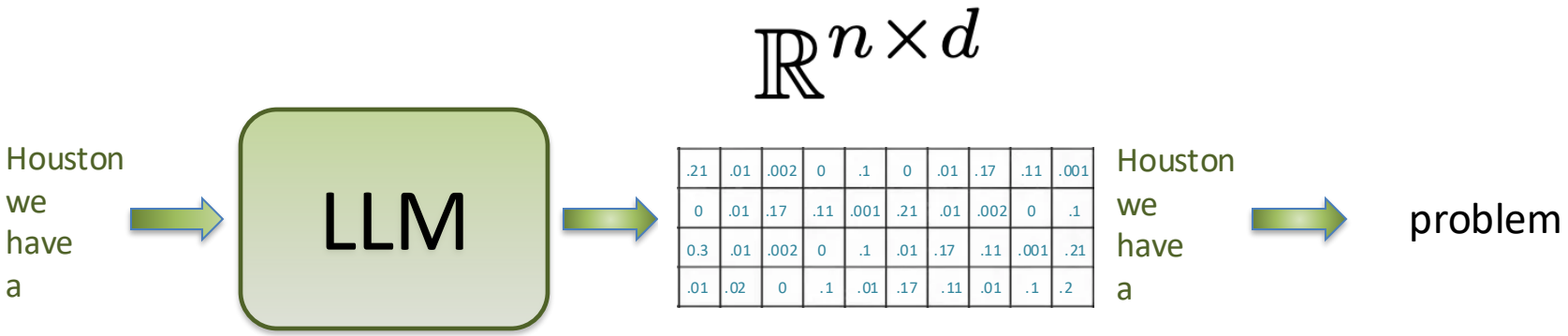
+ $\mathbb{R}^{n \times d}$



Inputs, Outputs & Internal Memory Size



LLM Training is Semi-Supervised



A Training Step

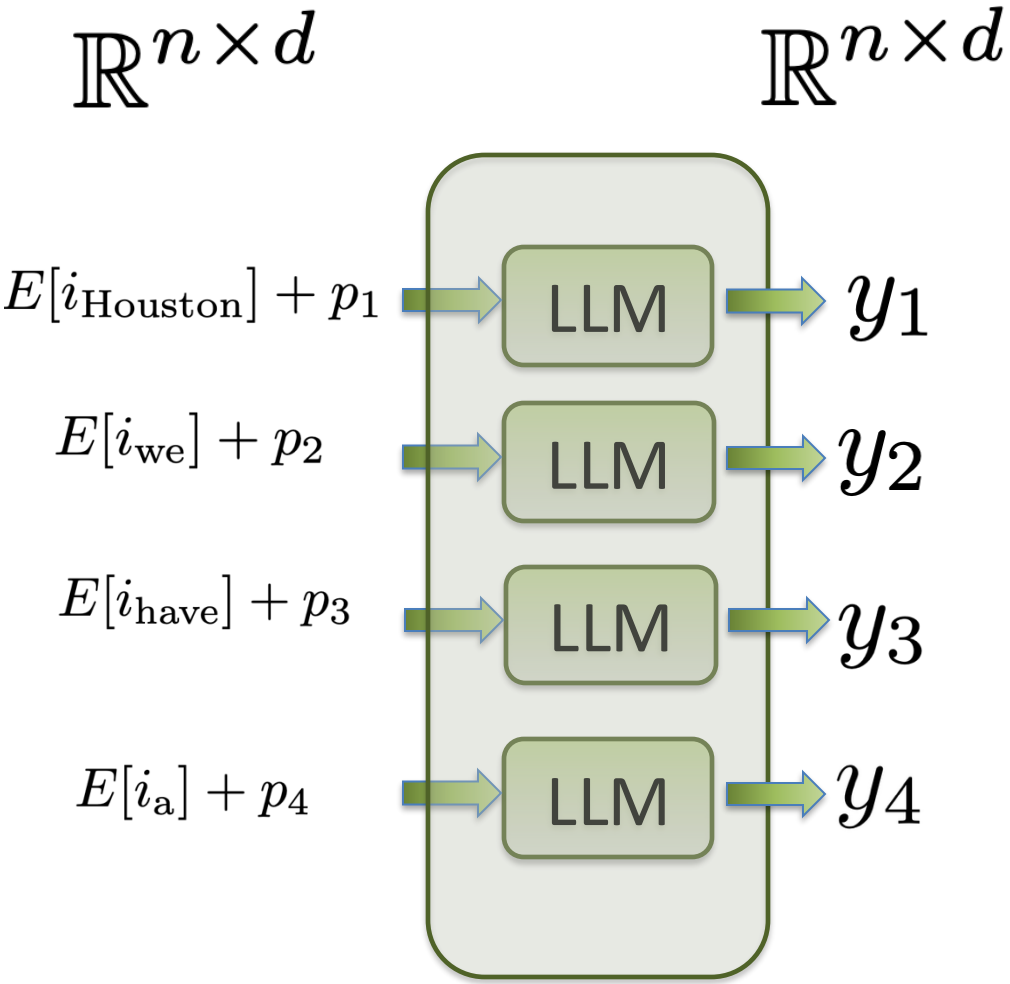
$X =$ "Houston we have a" $\qquad Y =$ "problem".

$$z = F_\theta^{trans}(X) \quad \text{with } \theta = \{W_Q, W_K, W_V, W_O, \dots\}$$

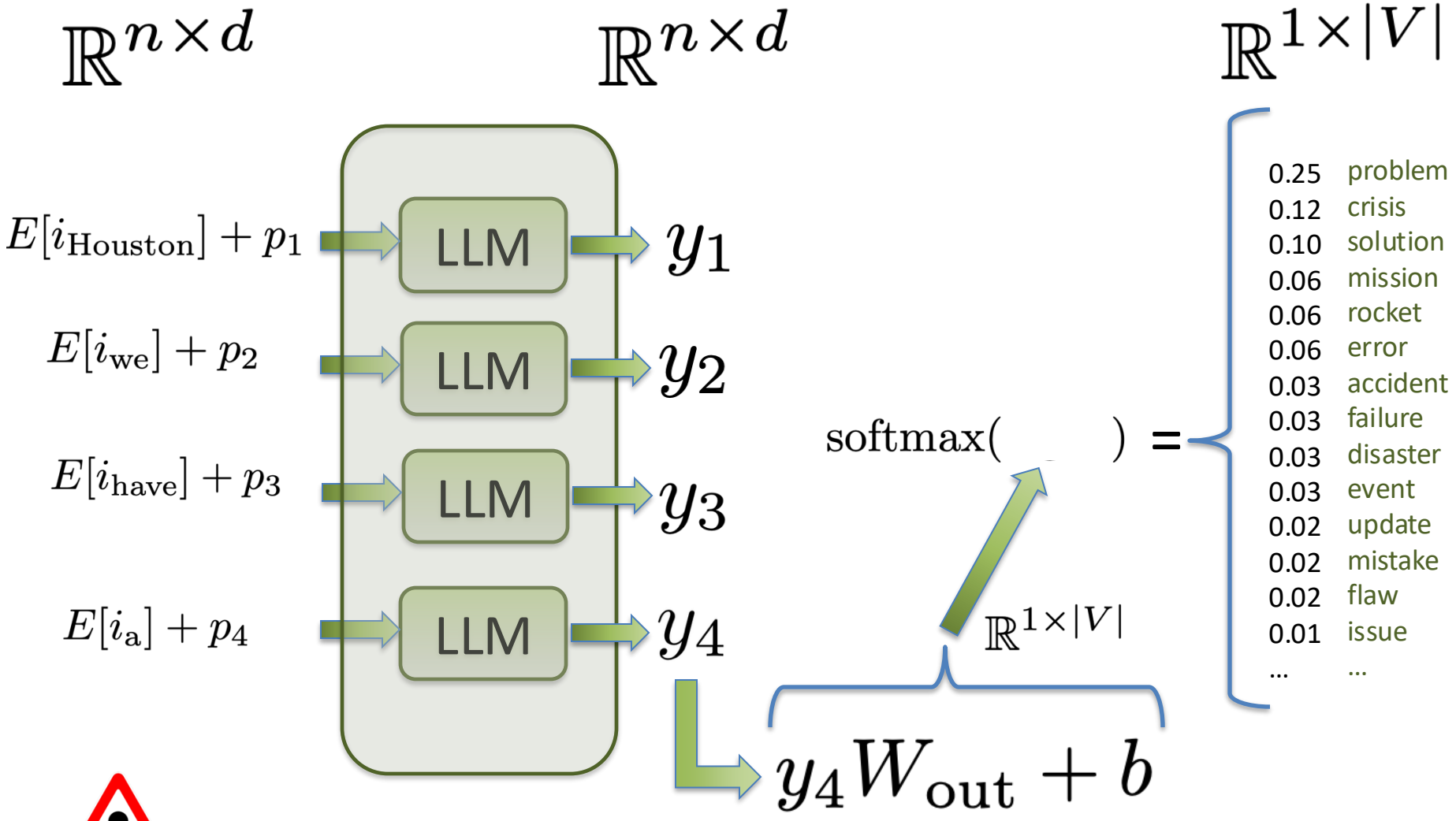
$$J(\theta) = -\log P(\text{"problem"} \mid X) = -\log\left(\text{softmax}(z)_{\text{problem}}\right)$$

$$\theta \leftarrow \theta - \eta \nabla_\theta J\left(F_\theta^{trans}(X)\right)$$

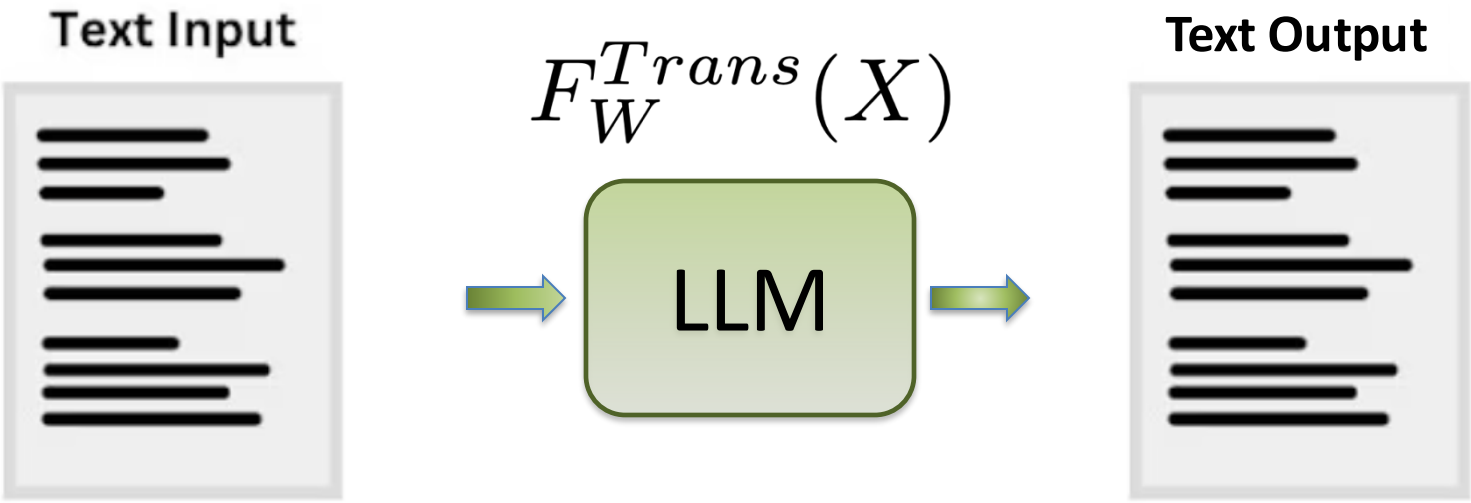
Training is Parallel



Next word generation



What is an LLM?



Which *Input* / *Outputs*?

Can take variable number of Inputs?

Neural Network Transformer defined by?

How is it used to generate text?

How is it trained?

LLM History 2013 - 2024

2013 Word2Vec Embeddings learnt to represent Characters, Parts of Word

2017 Transformers Encoder / Decoder architecture for Language Translation

2019 GPT2 1.5B params Transformer, autoregressive Decoder-Only,
Finetuned: summarization, question-answering, sentiment classification

2020 GPT3 Scaling. 175 billion params.

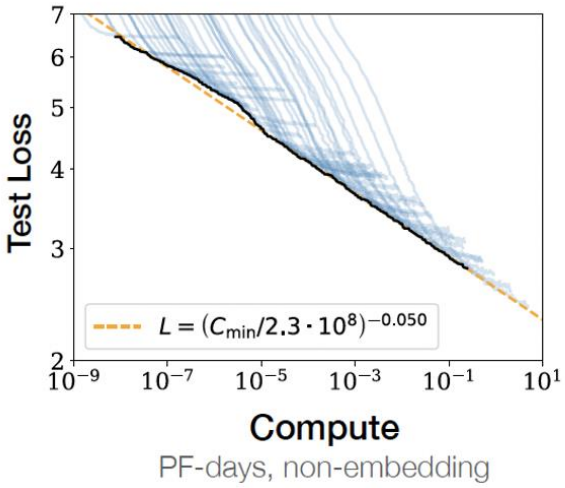
2022 GPT3.5 + RLHF, Instruct models RL from Human Feedback

2024 DeepSeek R1 Chain of Thought, test time compute,

Improving a Model with more…

Escuela AI HUB de verano
Sobre IA y Educación
aihub.csic.es

Compute Datasets Weights



2020

Scaling Laws for Neural Language Models

Jared Kaplan *
Johns Hopkins University, OpenAI
jaredk@jhu.edu

Sam McCandlish*
OpenAI
sam@openai.com

Tom Henighan
OpenAI
henighan@openai.com

Tom B. Brown
OpenAI
tom@openai.com

Benjamin Chess
OpenAI
bchess@openai.com

Rewon Child
OpenAI
rewon@openai.com

Scott Gray
OpenAI
scott@openai.com

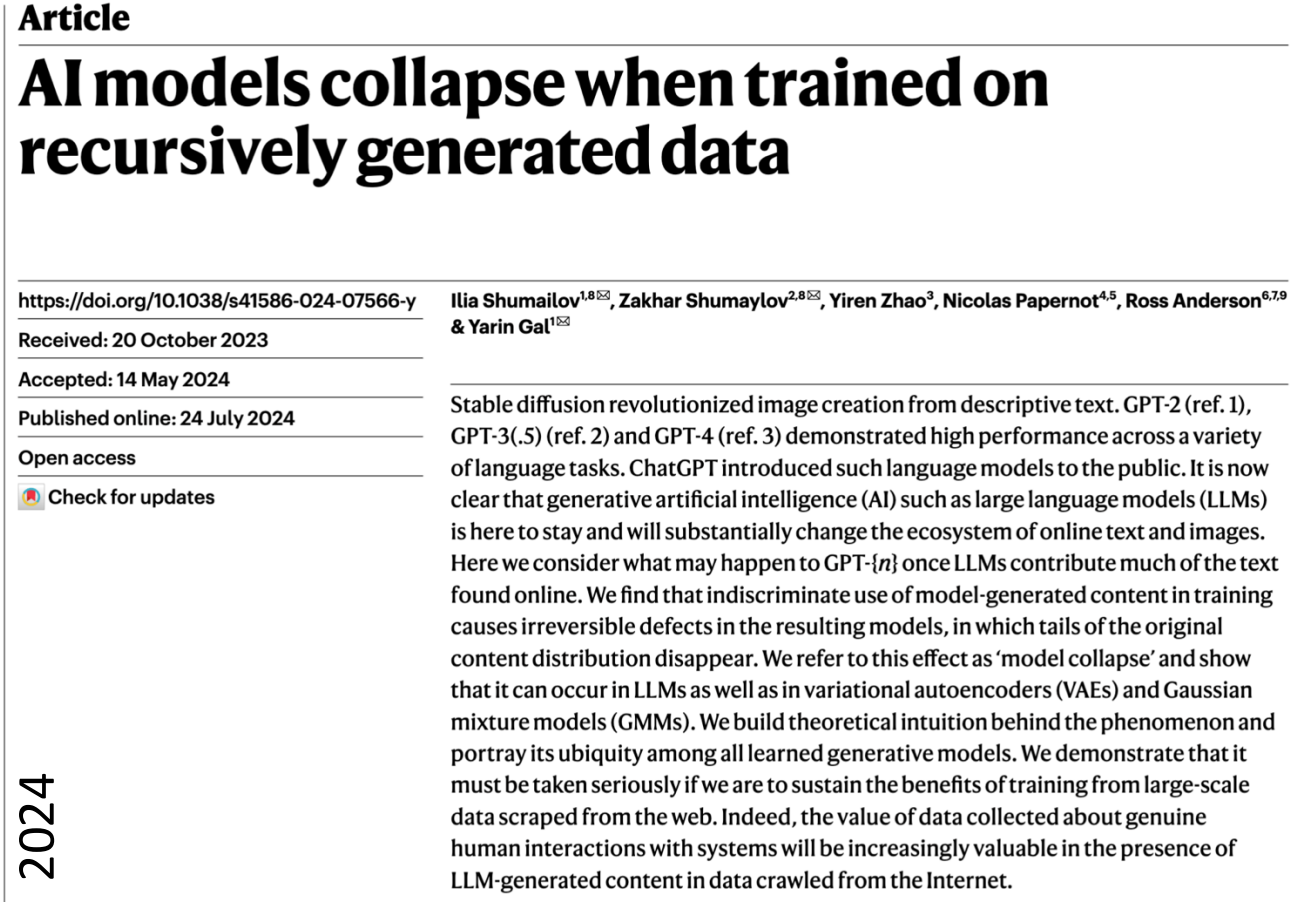
Alec Radford
OpenAI
alec@openai.com

Jeffrey Wu
OpenAI
jeffwu@openai.com

Dario Amodei
OpenAI
damodei@openai.com

More Data

- Use LLMs to generate more high-quality data?
- Model Collapse



Article

AI models collapse when trained on recursively generated data

https://doi.org/10.1038/s41586-024-07566-y

Received: 20 October 2023

Accepted: 14 May 2024

Published online: 24 July 2024

Open access

Check for updates

Ilia Shumailov[1,8]✉, Zakhar Shumaylov[2,8]✉, Yiren Zhao[3], Nicolas Papernot[4,5], Ross Anderson[6,7,9] & Yarin Gal[1]✉

Stable diffusion revolutionized image creation from descriptive text. GPT-2 (ref. 1), GPT-3(.5) (ref. 2) and GPT-4 (ref. 3) demonstrated high performance across a variety of language tasks. ChatGPT introduced such language models to the public. It is now clear that generative artificial intelligence (AI) such as large language models (LLMs) is here to stay and will substantially change the ecosystem of online text and images. Here we consider what may happen to GPT-{*n*} once LLMs contribute much of the text found online. We find that indiscriminate use of model-generated content in training causes irreversible defects in the resulting models, in which tails of the original content distribution disappear. We refer to this effect as 'model collapse' and show that it can occur in LLMs as well as in variational autoencoders (VAEs) and Gaussian mixture models (GMMs). We build theoretical intuition behind the phenomenon and portray its ubiquity among all learned generative models. We demonstrate that it must be taken seriously if we are to sustain the benefits of training from large-scale data scraped from the web. Indeed, the value of data collected about genuine human interactions with systems will be increasingly valuable in the presence of LLM-generated content in data crawled from the Internet.

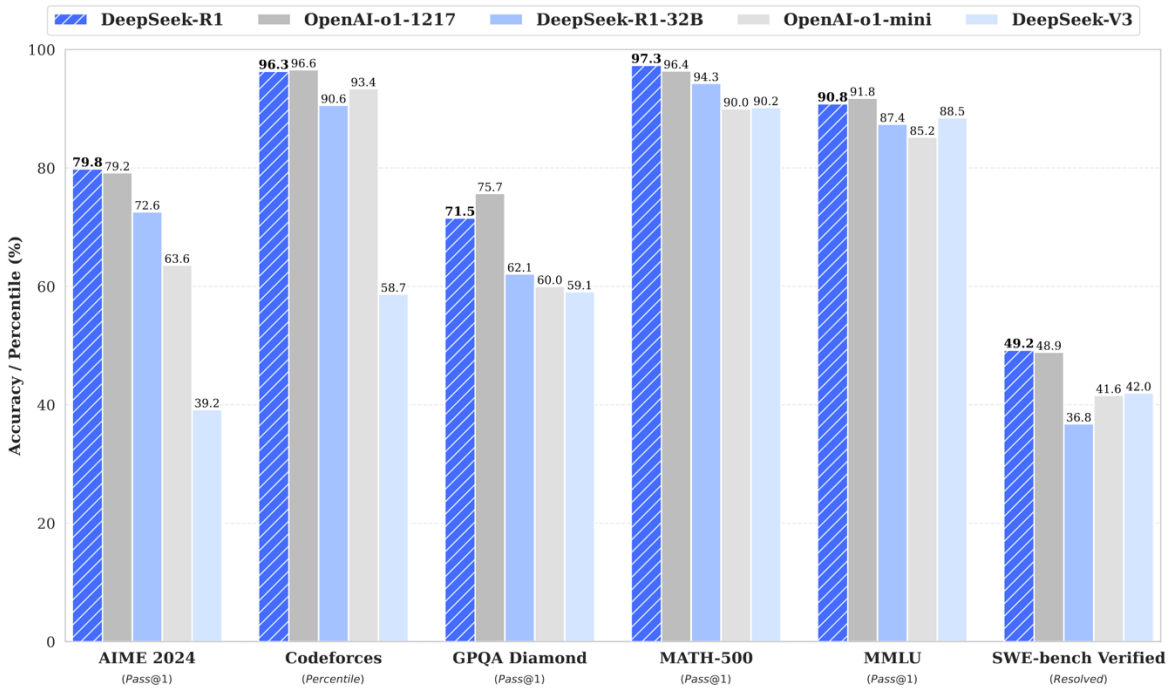
2024

DeepSeek R1

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

Abstract

We introduce our first-generation reasoning models, DeepSeek-R1-Zero and DeepSeek-R1. DeepSeek-R1-Zero, a model trained via large-scale reinforcement learning (RL) without supervised fine-tuning (SFT) as a preliminary step, demonstrates remarkable reasoning capabilities. Through RL, DeepSeek-R1-Zero naturally emerges with numerous powerful and intriguing reasoning behaviors. However, it encounters challenges such as poor readability, and language mixing. To address these issues and further enhance reasoning performance, we introduce DeepSeek-R1, which incorporates multi-stage training and cold-start data before RL. DeepSeek-R1 achieves performance comparable to OpenAI-o1-1217 on reasoning tasks. To support the research community, we open-source DeepSeek-R1-Zero, DeepSeek-R1, and six dense models (1.5B, 7B, 8B, 14B, 32B, 70B) distilled from DeepSeek-R1 based on Qwen and Llama.



Transformer Limitations

Task	Doable?	Reason
Sorting	✅	Pairwise comparisons work well in attention
Finding Maximum/Minimum	✅	Self-attention can highlight extreme values
Classification (Sentiment, Object Recognition)	✅	Pattern recognition fits attention well
Logical AND/OR (Thresholding)	✅	Can be done via aggregation
Parsing & Syntax Trees	✅	Hierarchical representations are learnable
Simple Arithmetic (Small Integers)	✅	Attention can approximate patterns
Pattern Matching (Substring Search)	✅	Self-attention highlights relevant tokens
Learning Some Regular Languages	✅	Some patterns fit attention mechanisms
Copying	❌	Needs exact token recall, which soft attention lacks
Parity Check	❌	Not computable in AC^0 (Transformers belong to AC^0)
Addition/Multiplication (Arbitrary Length)	❌	Requires multi-step, precise memory
Palindrome Detection	❌	Needs mirrored recall beyond context
Reversing a Sequence	❌	Similar to copying, needs exact tracking
Exact Counting Beyond Context	❌	Requires memory persistence
Parentheses Matching (Dyck Language)	❌	Needs recursion, which Transformers lack
Complex Strategy Games (Chess, Sudoku)	❌	Requires long-term planning and memory

Limitations

- Neural Networks are Universal Approximators (Hornik et al., 89)
- But not Turing Complete

- Analog RNNs (using, for example, sigmoidal activation functions and rational weights) have Turing-complete computational power.

- Recurrent neural networks are Turing complete because they can maintain and update hidden states (serving as memory)

Hava T. Siegelmann and Eduardo D. Sontag. 1992. On the computational power of neural nets. In Proceedings of the fifth annual workshop on Computational learning theory (COLT '92). Association for Computing Machinery, New York, NY, USA, 440–449. https://doi.org/10.1145/130385.130432

Transformer Limitations

Bad at:

- Large Continuous Data
- Long Time Series Prediction
- Long discrete DNA sequences
- Bad at Search but Good at Guiding Search

LONG RANGE ARENA: A BENCHMARK FOR EFFICIENT TRANSFORMERS

Yi Tay[1]*, Mostafa Dehghani[1]*, Samira Abnar[1], Yikang Shen[1], Dara Bahri[1], Philip Pham[1]
Jinfeng Rao[1], Liu Yang[1], Sebastian Ruder[2], Donald Metzler[1]
[1]Google Research
[2]Google DeepMind
{yitay, dehghani}@google.com

Beyond Transformers

- Improve Attention: PerFormer (2021), Flash Attention (2022), Infini-Attention (2024)
- Better LLMs: Mixture Of Experts: Switch Transformers (2021)
- Multimodal LLMs
  - Visual Transformers versus CNNs, All-MLPs (2021)
  - CLIP Supervised Contrastive Learning (2021)
- State Space Models Mamba (2023)
- Diffusion LLMs (2022,2023)
- Mixing Graph Neural Networks and Transformers
  - Graph Attention Networks (2017), GraphFormer (2021), Graph Language Models
- Mixing Knowledge Graphs and LLMs

Escuela AI HUB de verano
Sobre IA y Educación
aihub.csic.es

Gracias

Organizan

CSIC Consejo Superior de Investigaciones Científicas | CSIC Hub Inteligencia Artificial

Fundación "la Caixa"

Colaboran

OEIAC | Universitat de Girona

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH
Institut de Ciències de l'Educació

conecta 13 Spin-Off

AI Chair-UPC Siemens Energy
Energy Sustainability for a Decarbonized Society 5.0

ellis | UNIT BARCELONA

Financiado por la Unión Europea NextGenerationEU

Gobierno de España · Ministerio para la Transformación Digital y de la Función Pública

España | digital 2026

Plan de Recuperación, Transformación y Resiliencia

ENIA Estrategia Nacional de Inteligencia Artificial