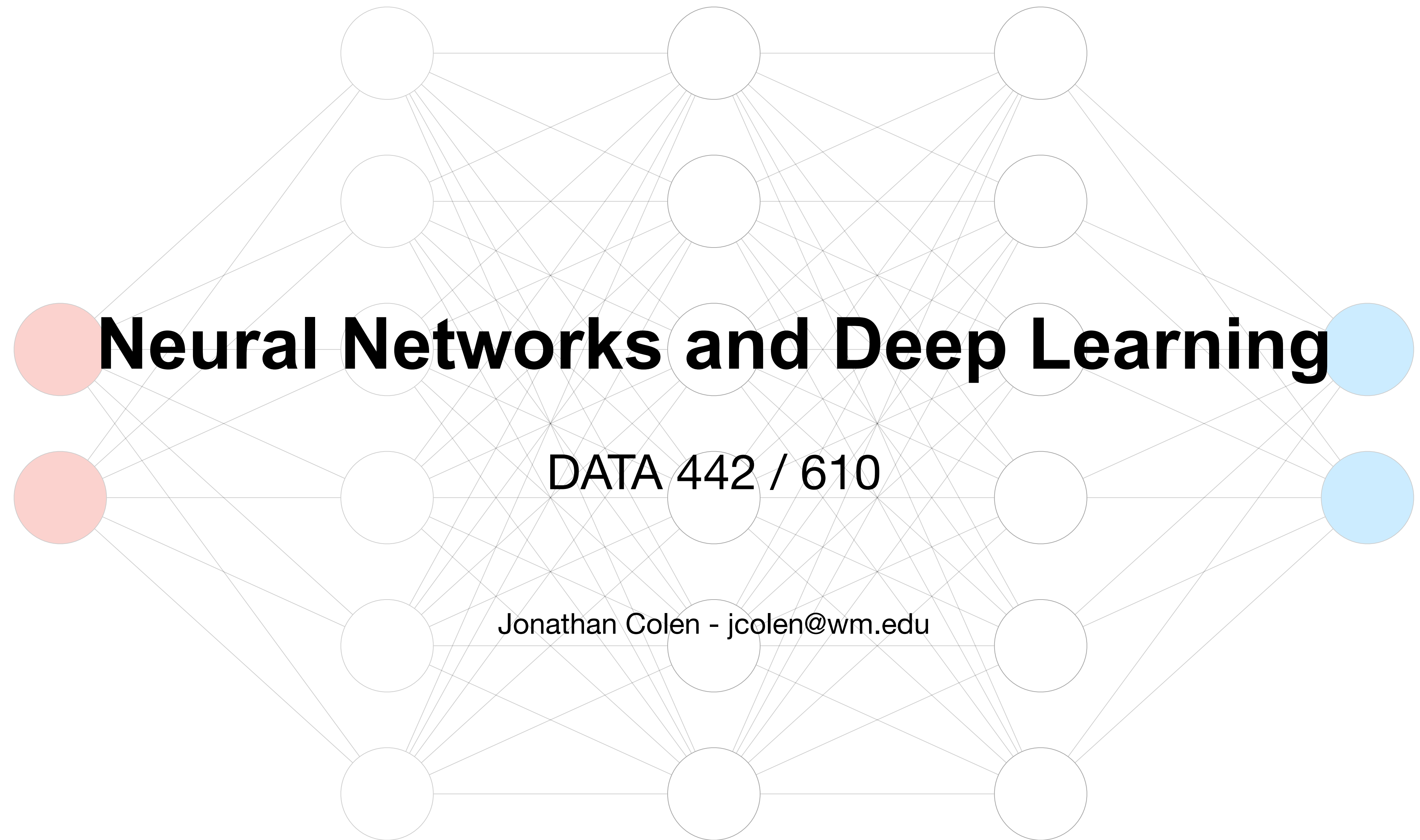




Outline

- Course Introduction
 - Syllabus
 - Structure & grading
 - Schedule
 - Course outline and goals
- History of Deep Learning

Relationships with Other Courses @ W&M DS

DATA 201 - Introduction to Data Science & AI

DATA 301 - Applied Machine Learning

DATA 442 (This Class) - Neural Networks & Deep Learning

Syllabus

- Available on the course website is <https://colen-teaching.github.io/DATA442-Fall-2026>
- All other materials (lecture slides, demos, assignments, etc.) will be posted there, so check frequently

Final Project

Final Report [40%] 3-5 pages

- Introduction/Motivation (including related work)
- Data/Methods
- Results
- Discussion
- Conclusion/Future Work
- References/Appendices (do not count towards page limit)

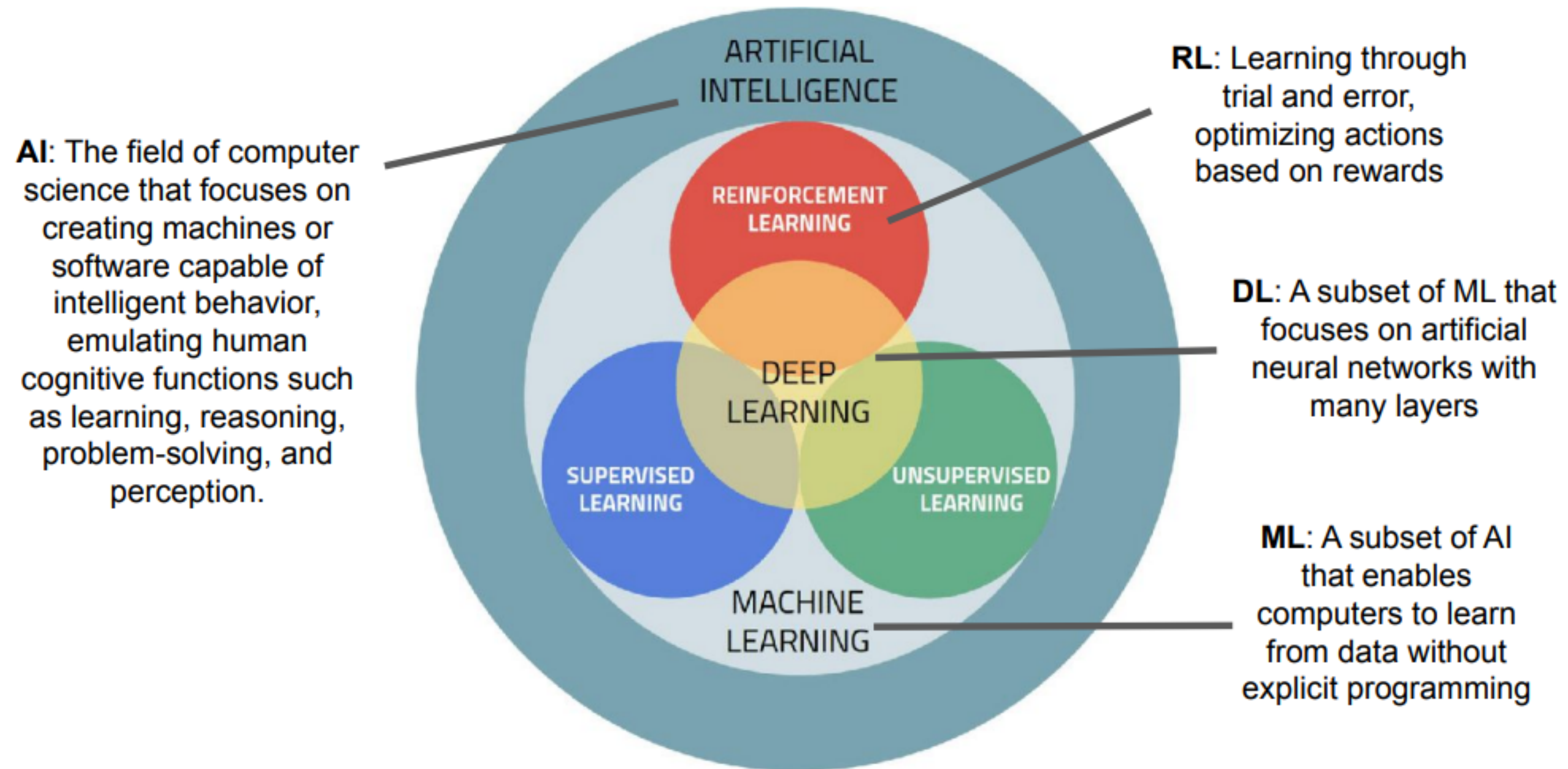
Presentation [60%] ~10 minutes + 5 for questions

Bonus point opportunities

- Originality [5%] is the work novel?
- Participation [5%] are you supporting your classmates?

What is Deep Learning?

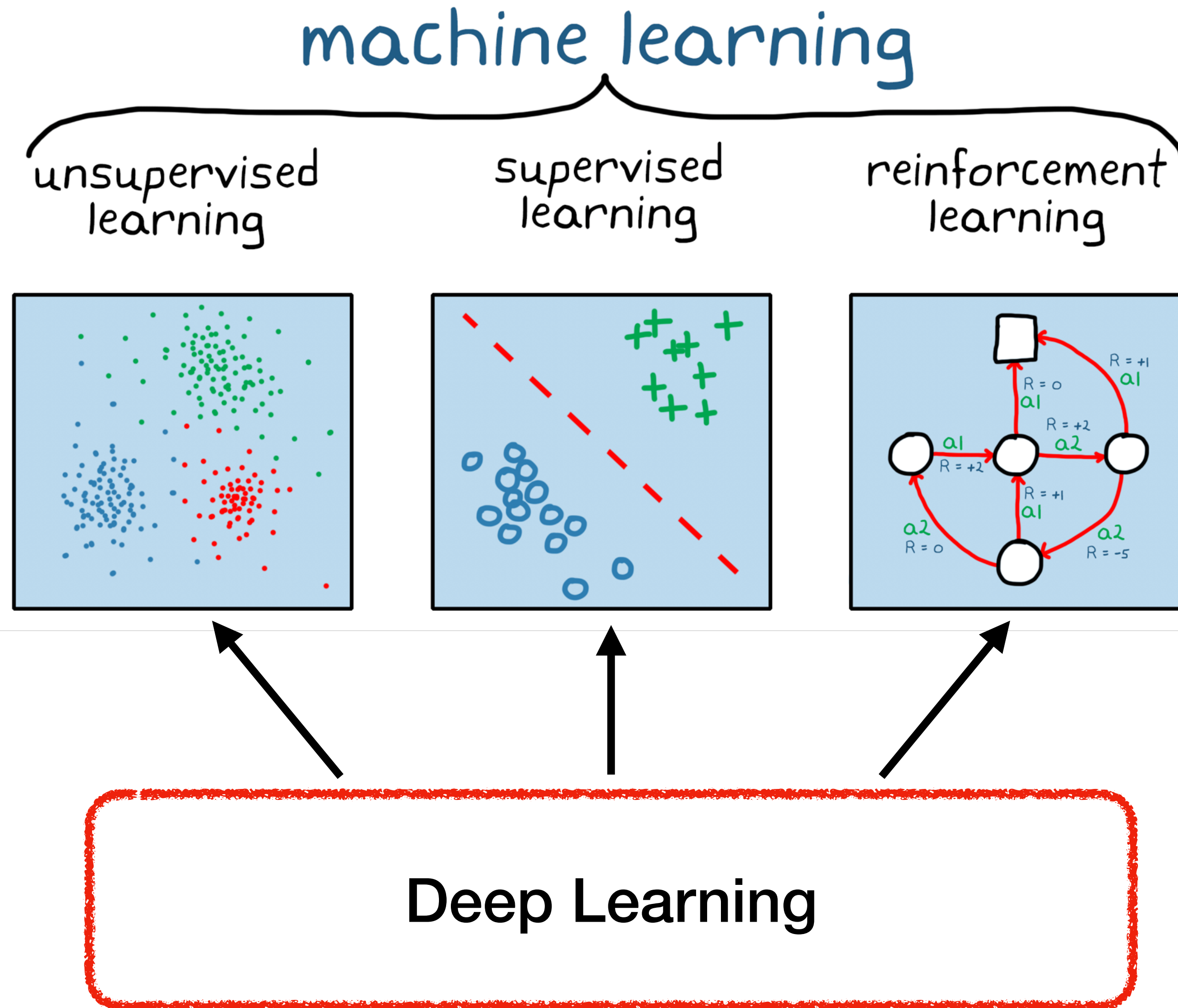
What is Deep Learning?



Courtesy of C. Fanelli

Deep Learning is a subset of Machine Learning

<https://news.sparkfun.com/7369>



Supervised learning maps an input to an output based on a dataset of labeled training examples

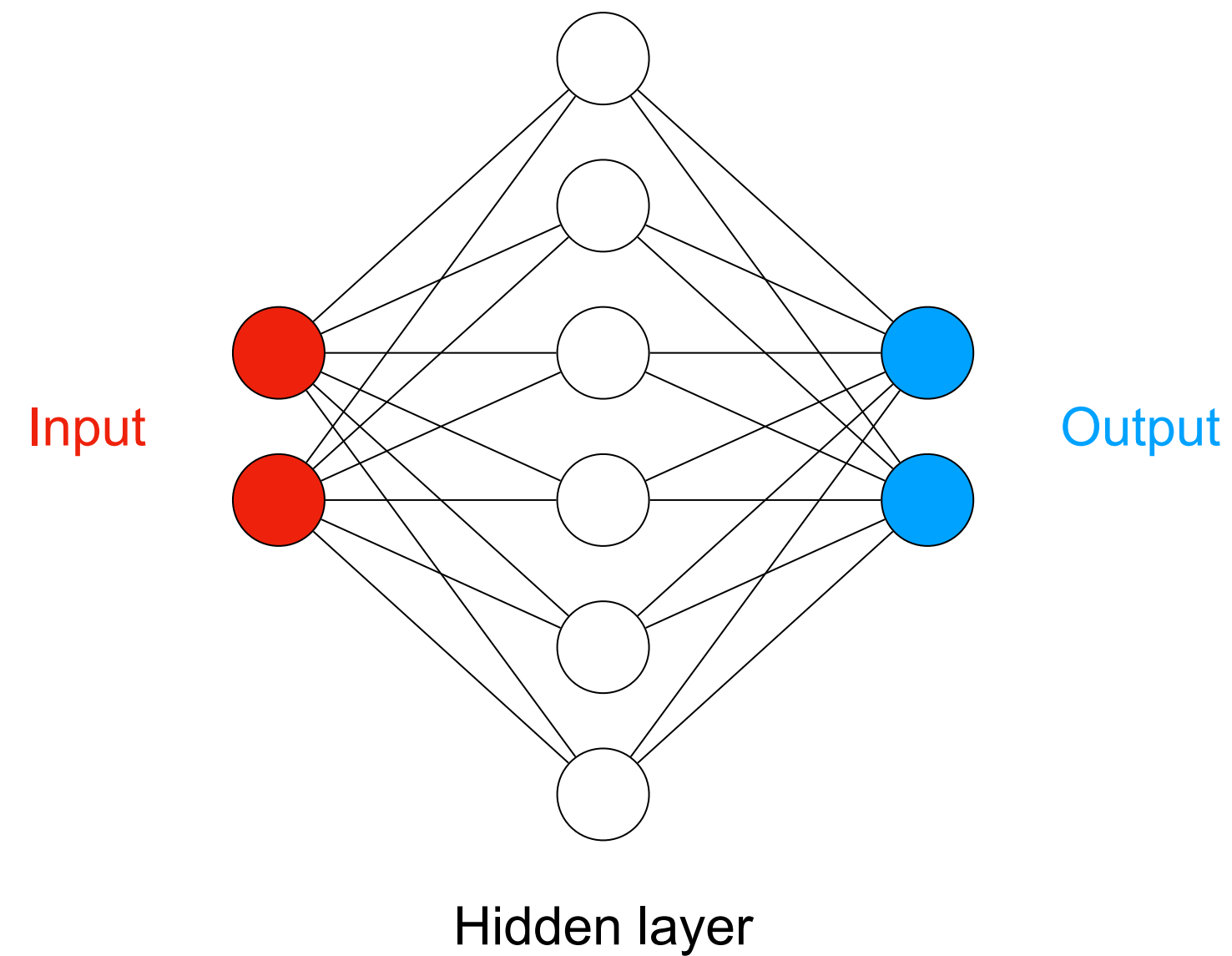
Unsupervised learning learns patterns and structure within training data without prior labels

Reinforcement learning involves agents interacting with an environment to maximize a reward

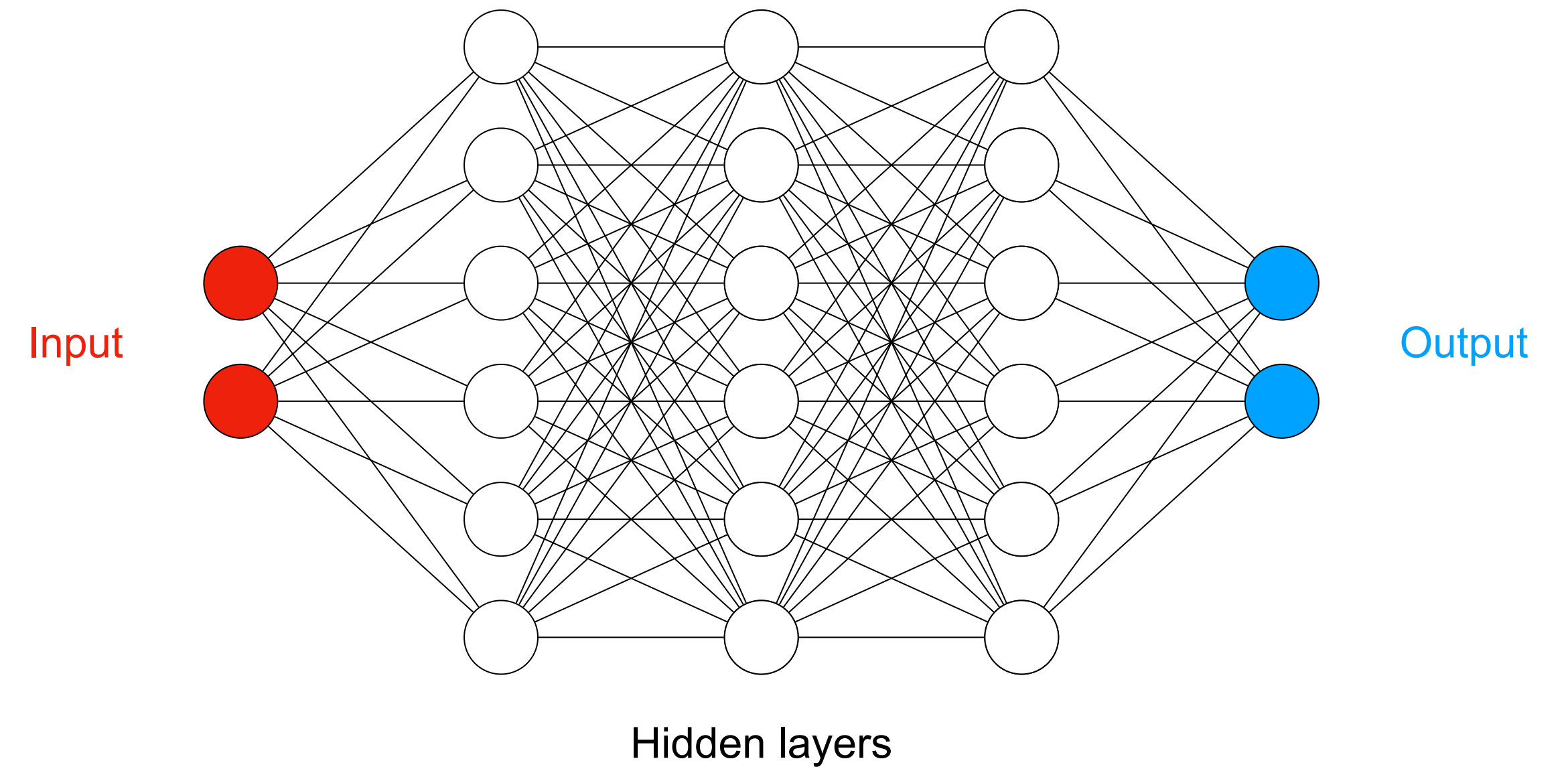
Deep learning applies deep neural networks to solve problems in each of these areas

Deep Neural Networks

This is a neural network



This is a **deep** neural network



Deep neural networks process data through multiple hidden layers of artificial neurons

Why do we care about deep learning and neural networks?

Deep neural networks...

...are really good at learning arbitrary functions
from data

...can learn hierarchical features

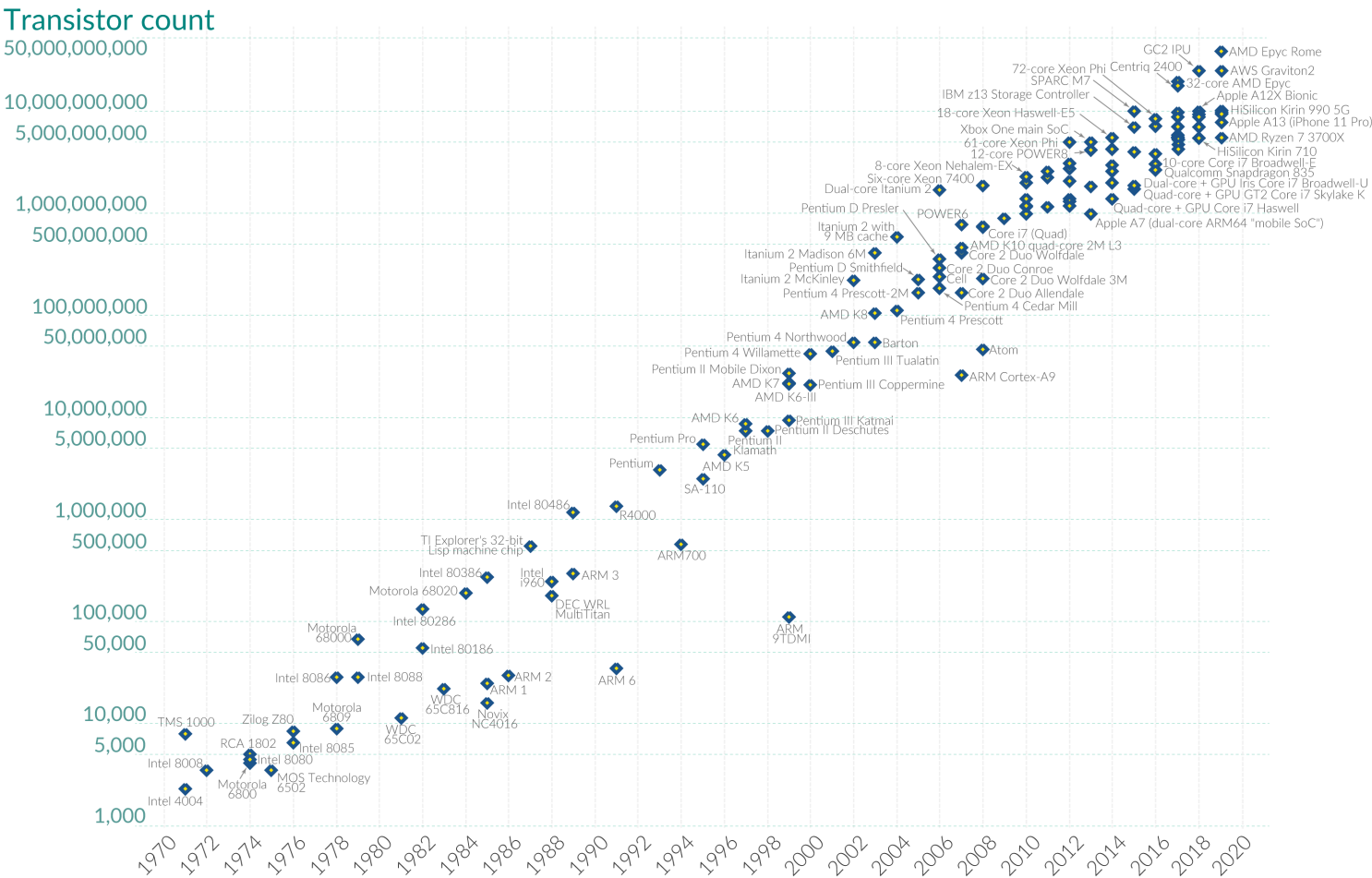
...scale well to large models and
datasets

...are exempt from the classical bias-
variance tradeoff***

Deep learning scales to big data and big problems

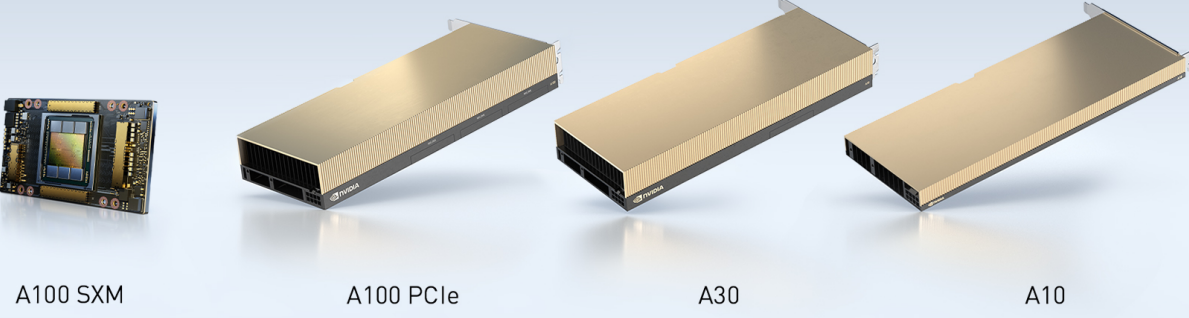
Bigger hardware

Moore's Law: The number of transistors on microchips doubles every two years
Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.

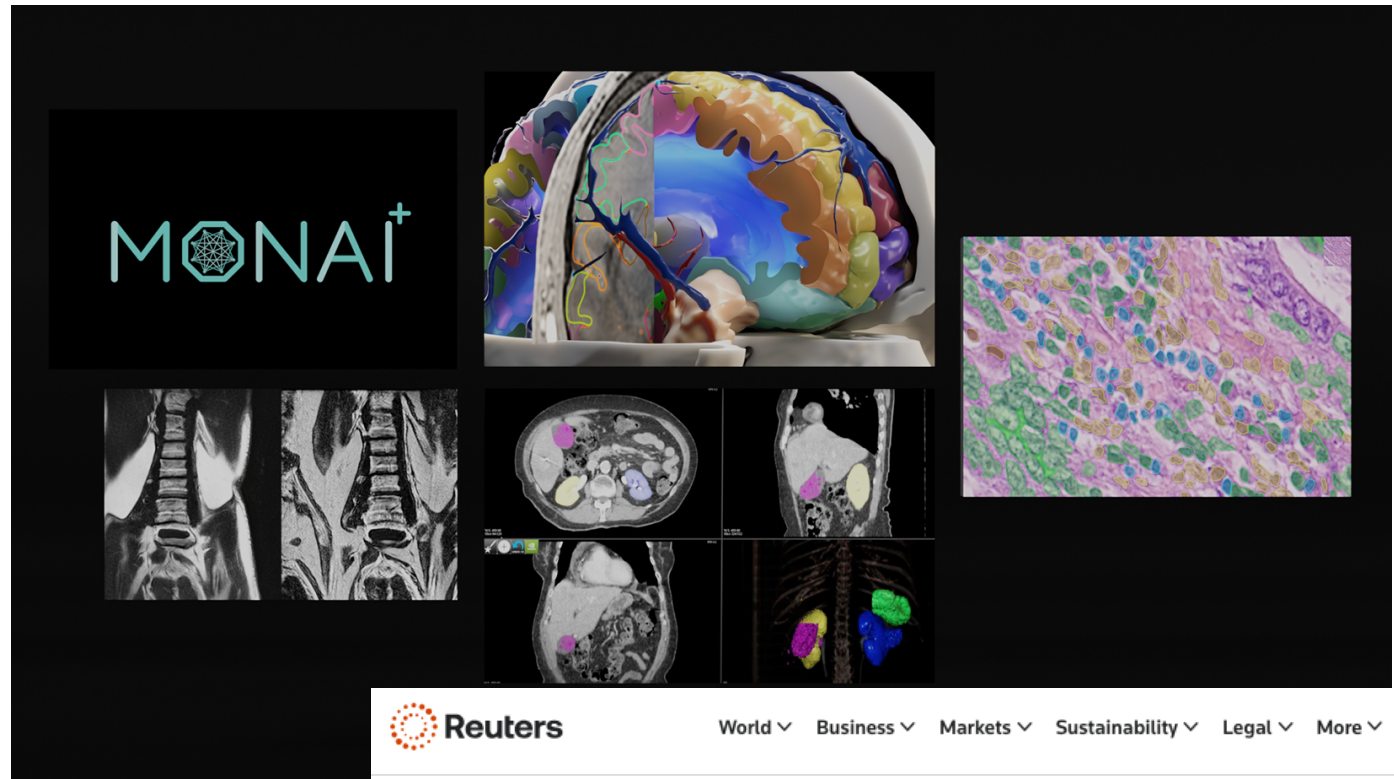


Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
OurWorldInData.org – Research and data to make progress against the world's largest problems.
Licensed under CC-BY by the authors Hannah Ritchie and Max Roser.

NVIDIA Tensor Core GPUs



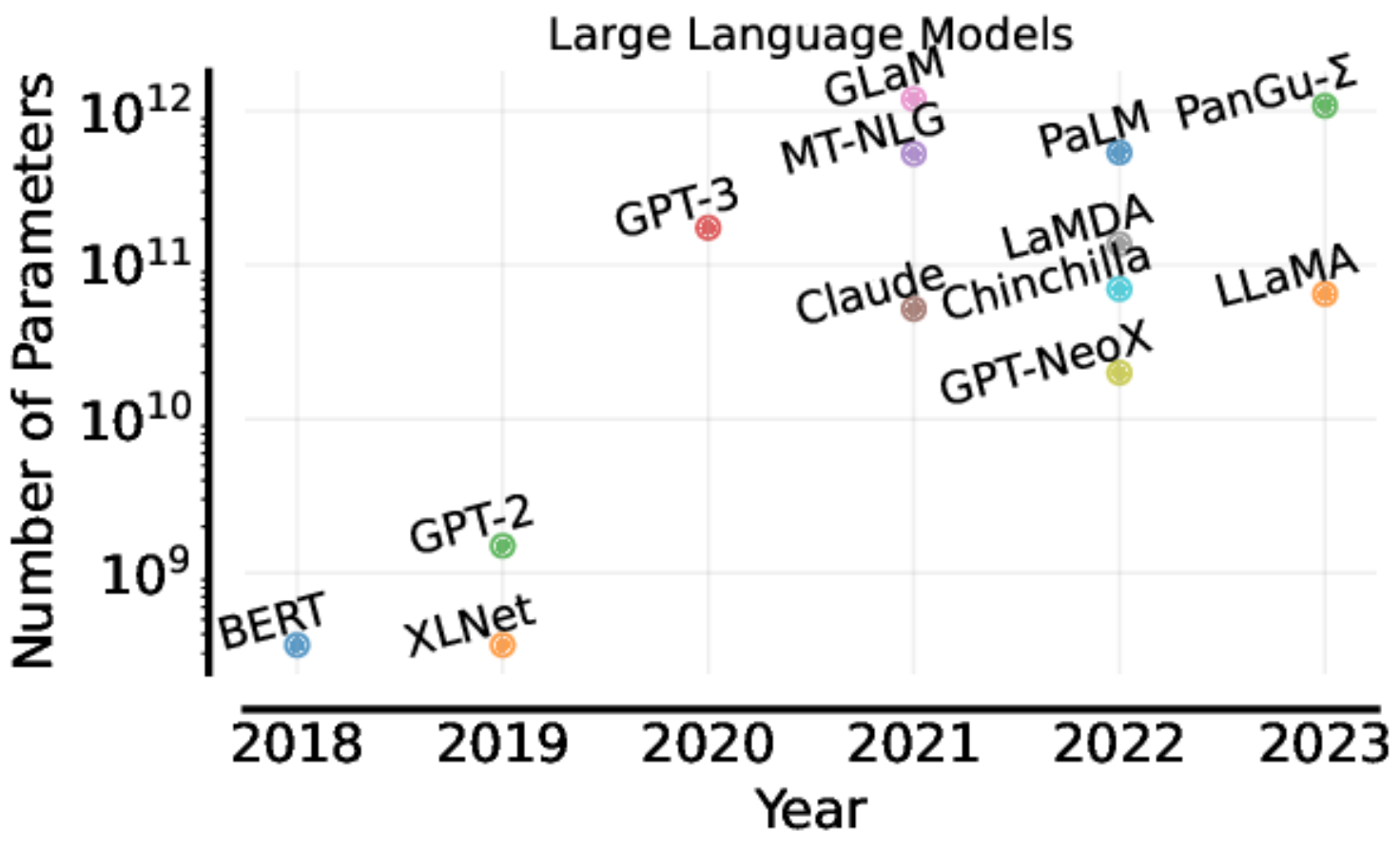
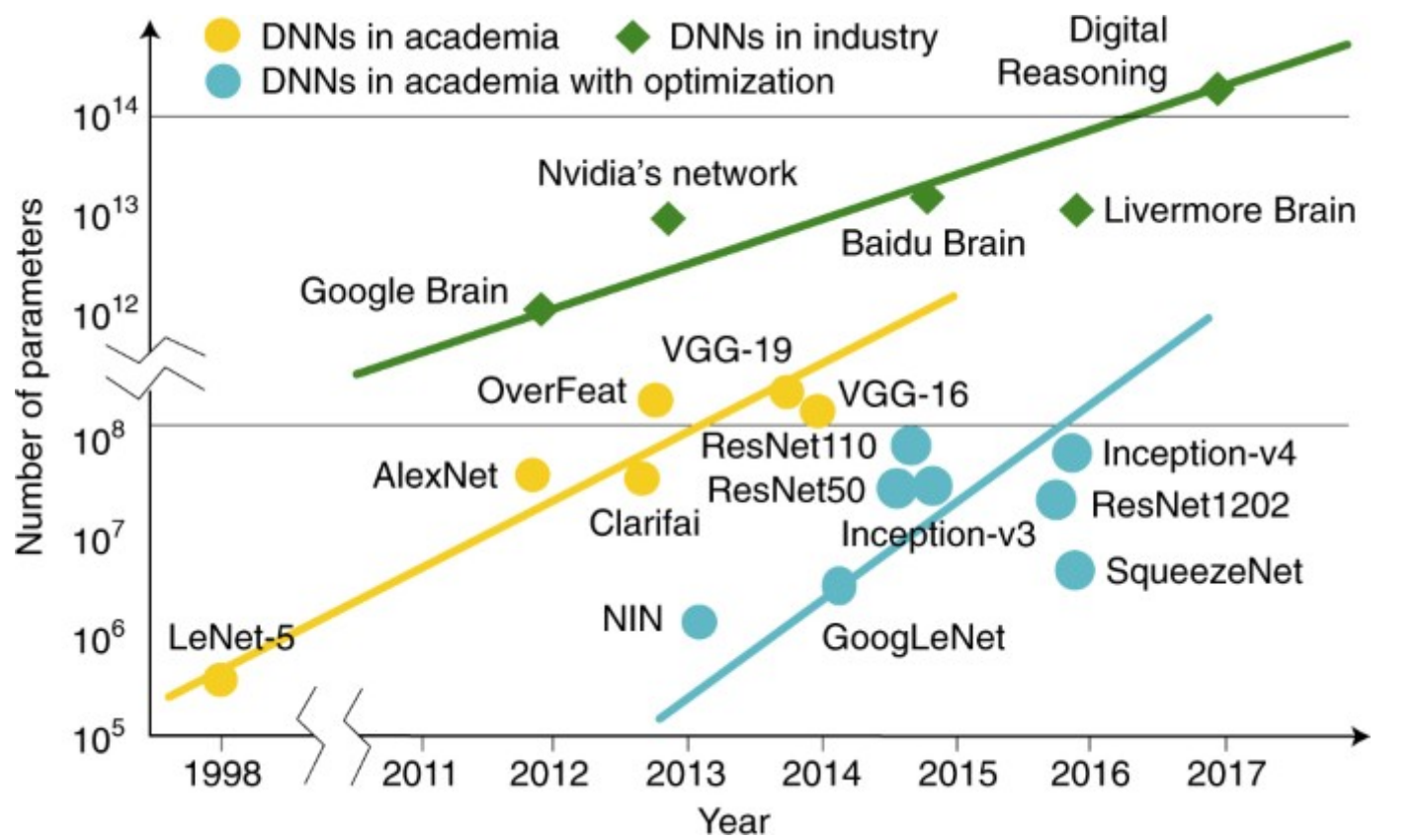
Bigger data



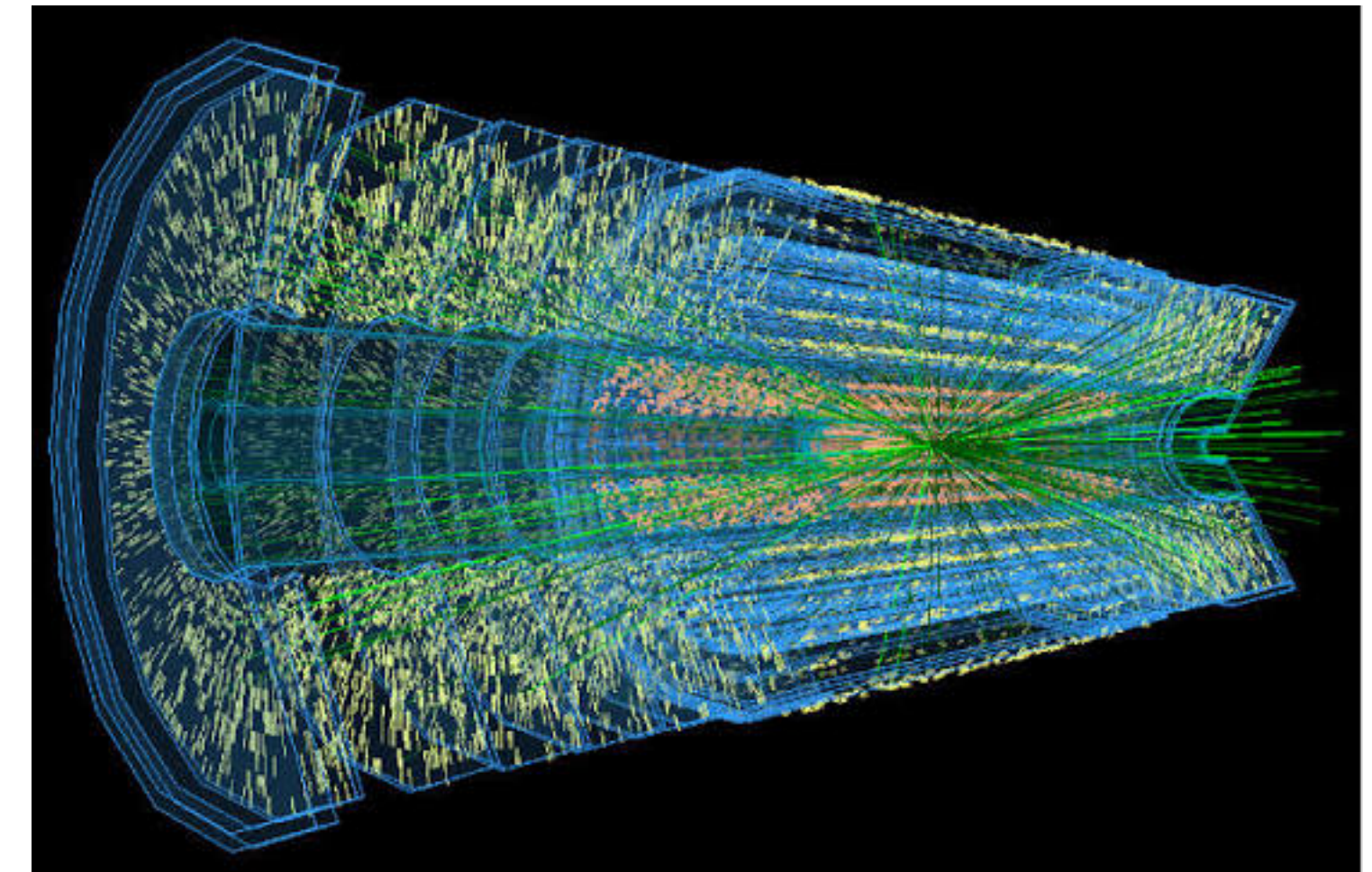
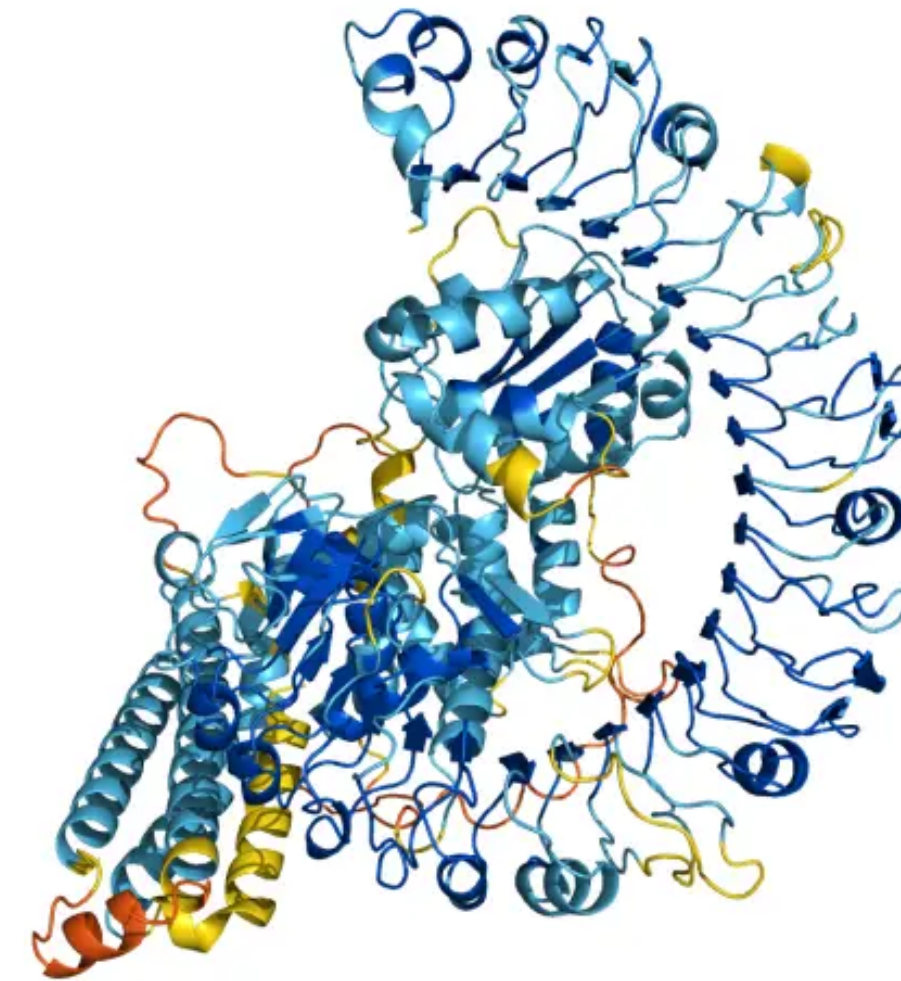
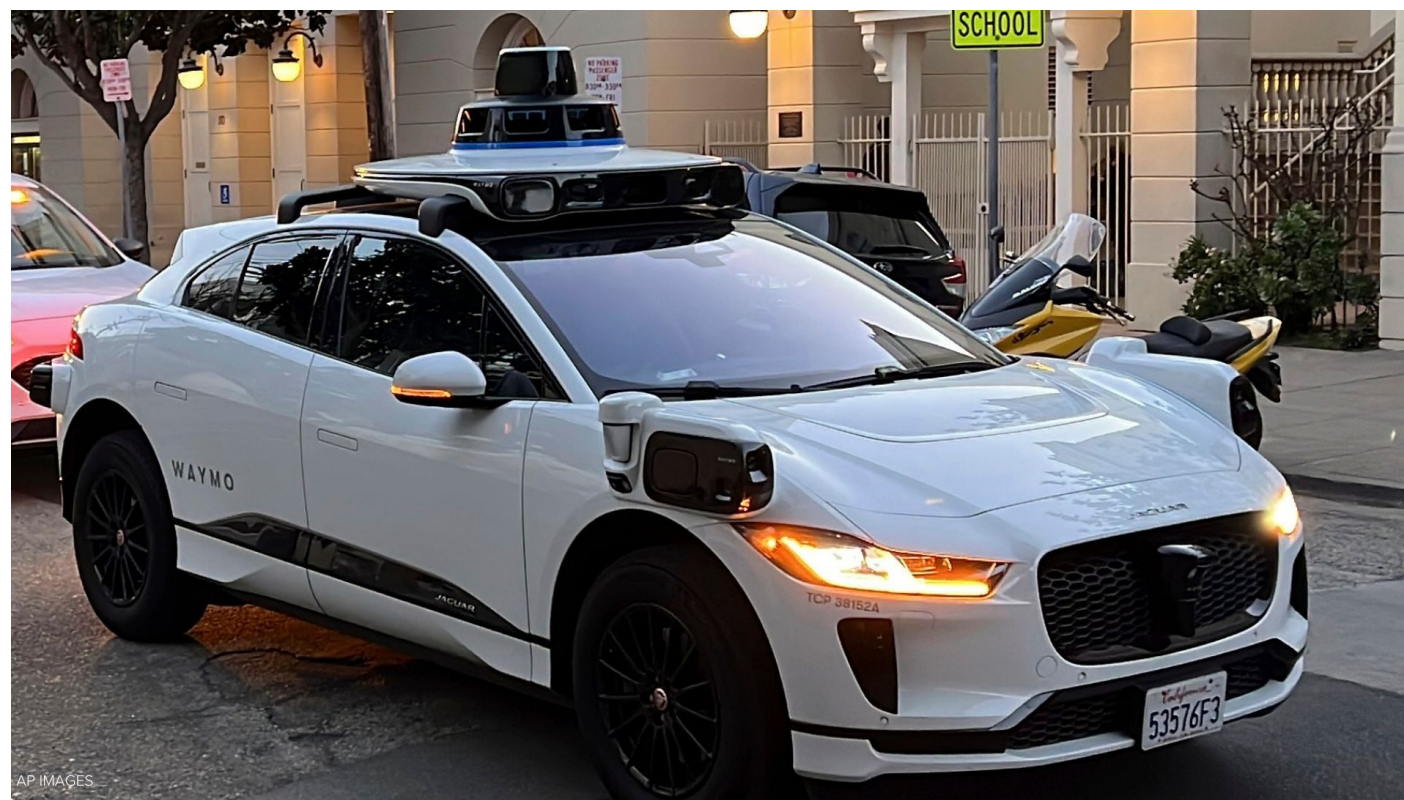
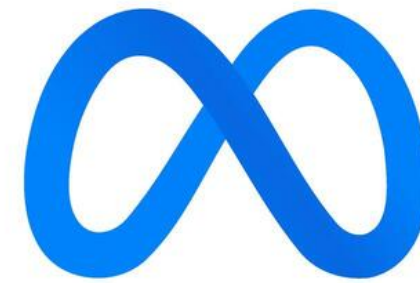
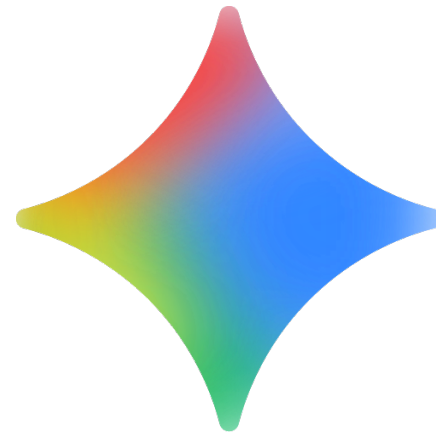
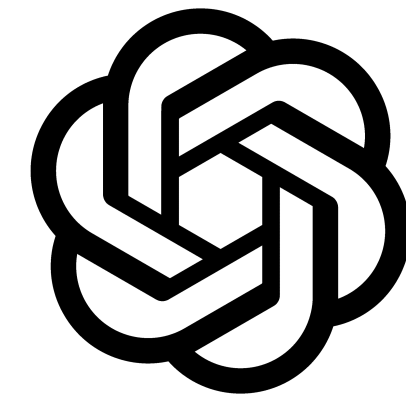
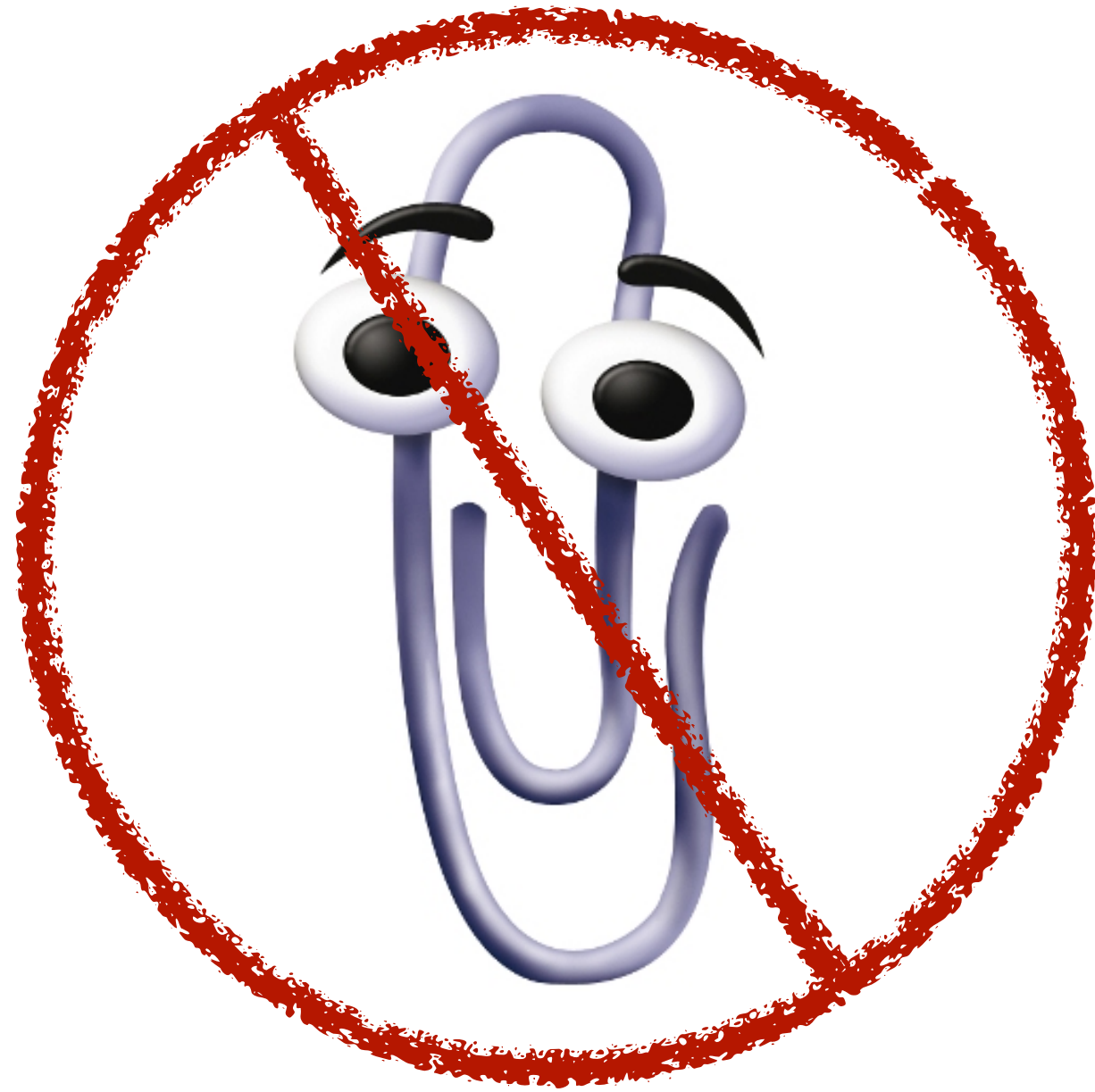
US judge approves Anthropic's \$1.5 billion settlement of copyright lawsuit



Bigger models



Deep neural networks underpin modern AI/ML

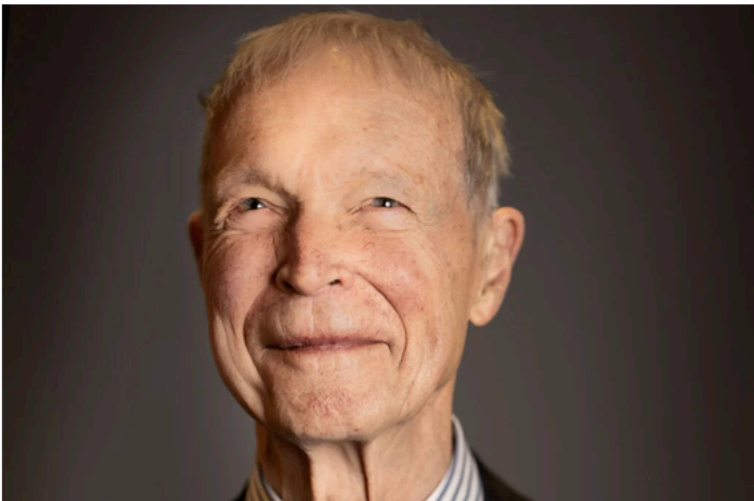


Deep learning wins you Nobel Prizes

The Nobel Prize in Physics 2024

John J. Hopfield

“for foundational discoveries and inventions that enable machine learning with artificial neural networks”



© Nobel Prize Outreach. Photo: Nanaka Adachi

Geoffrey Hinton

“for foundational discoveries and inventions that enable machine learning with artificial neural networks”



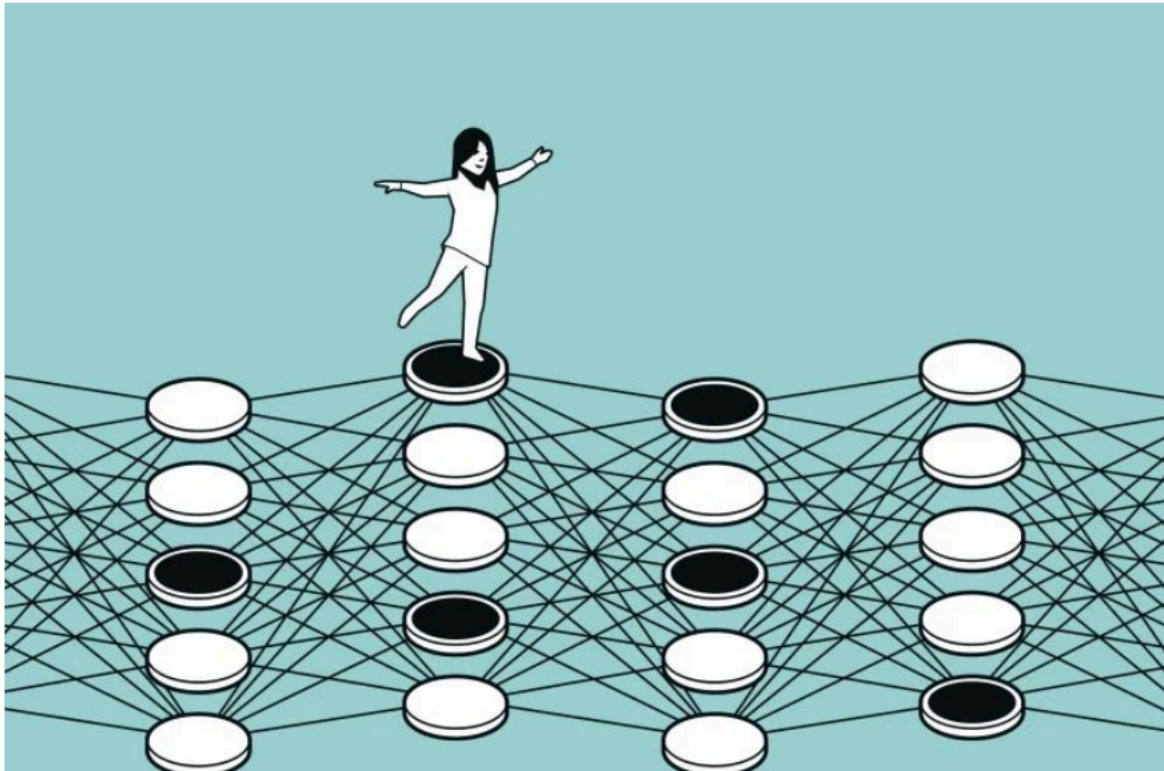
© Nobel Prize Outreach. Photo: Clément Morin

They used physics to find patterns in information

This year’s laureates used tools from physics to construct methods that helped lay the foundation for today’s powerful machine learning. John Hopfield created a structure that can store and reconstruct information. Geoffrey Hinton invented a method that can independently discover properties in data and which has become important for the large artificial neural networks now in use.

Related articles

[Press release](#)
[Popular information: They used physics to find patterns in information](#)
[Scientific background: “for foundational discoveries and inventions that enable machine learning with artificial neural networks”](#)



© Johan Jarnestad/The Royal Swedish Academy of Sciences

The Nobel Prize in Chemistry 2024

David Baker

“for computational protein design”



© Nobel Prize Outreach. Photo: Clément Morin

Demis Hassabis

“for protein structure prediction”



© Nobel Prize Outreach. Photo: Clément Morin

John Jumper

“for protein structure prediction”



© Nobel Prize Outreach. Photo: Clément Morin

They cracked the code for proteins’ amazing structures

The Nobel Prize in Chemistry 2024 is about proteins, life’s ingenious chemical tools. David Baker has succeeded with the almost impossible feat of building entirely new kinds of proteins. Demis Hassabis and John Jumper have developed an AI model to solve a 50-year-old problem: predicting proteins’ complex structures. These discoveries hold enormous potential.

Related articles

[Press release](#)
[Popular information: They have revealed proteins’ secrets through computing and artificial intelligence](#)
[Scientific background: Computational protein design and protein structure prediction](#)

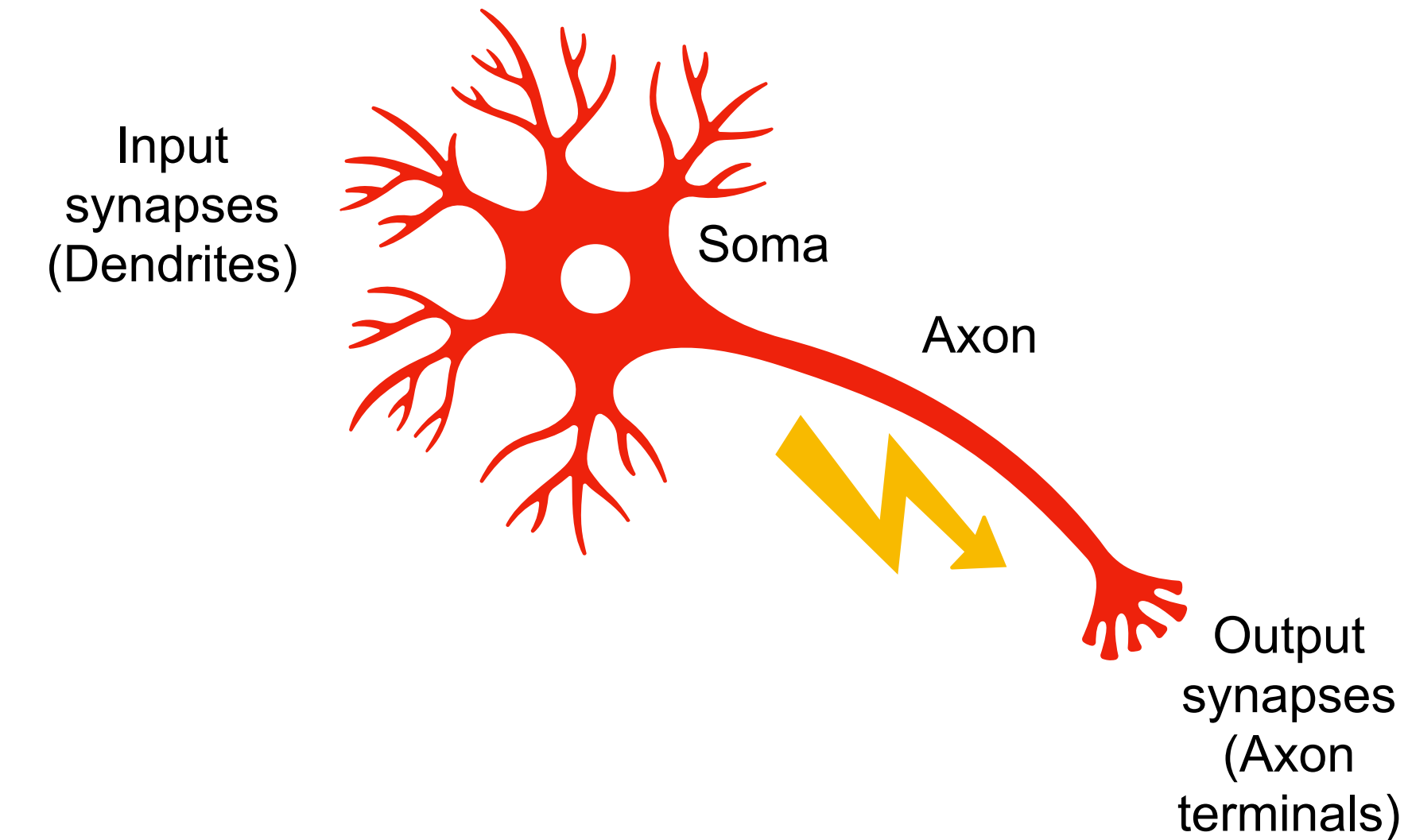


© Johan Jarnestad/The Royal Swedish Academy of Sciences

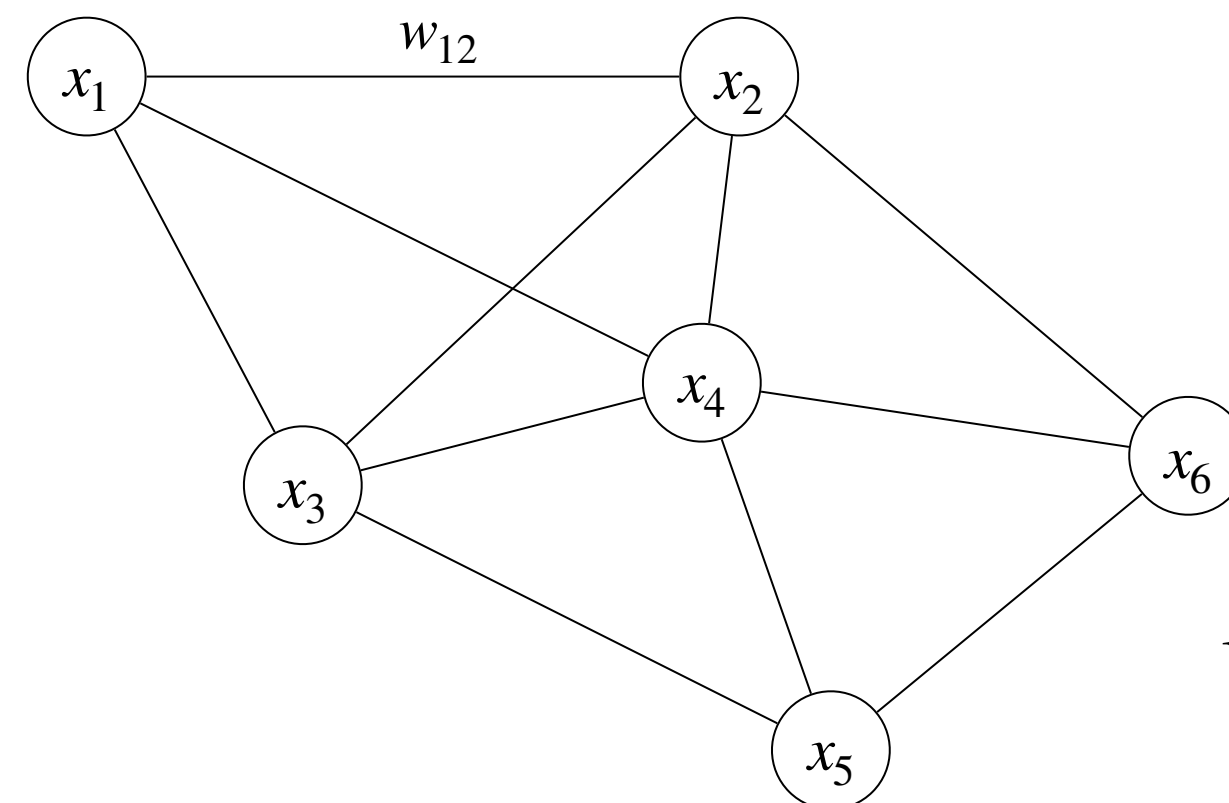
How did we get here?

Pre-history

Biological Neuron



Hebbian Learning (1949)

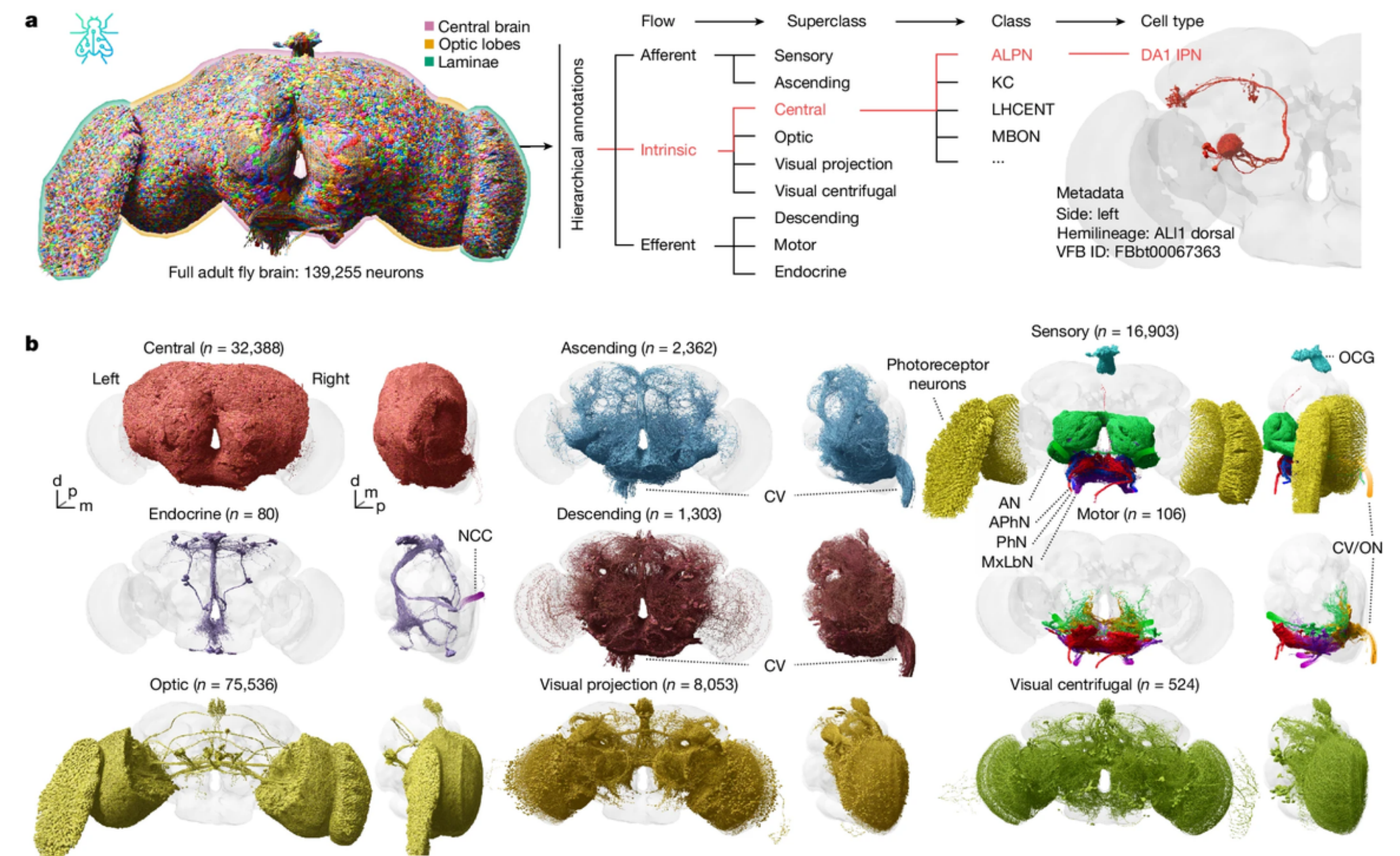


$$w_{ij} \leftarrow \mathbb{E}[x_i \cdot x_j]$$

“Neurons that fire together, wire together”

Fig. 1: Hierarchical annotation schema for a whole-brain connectome.

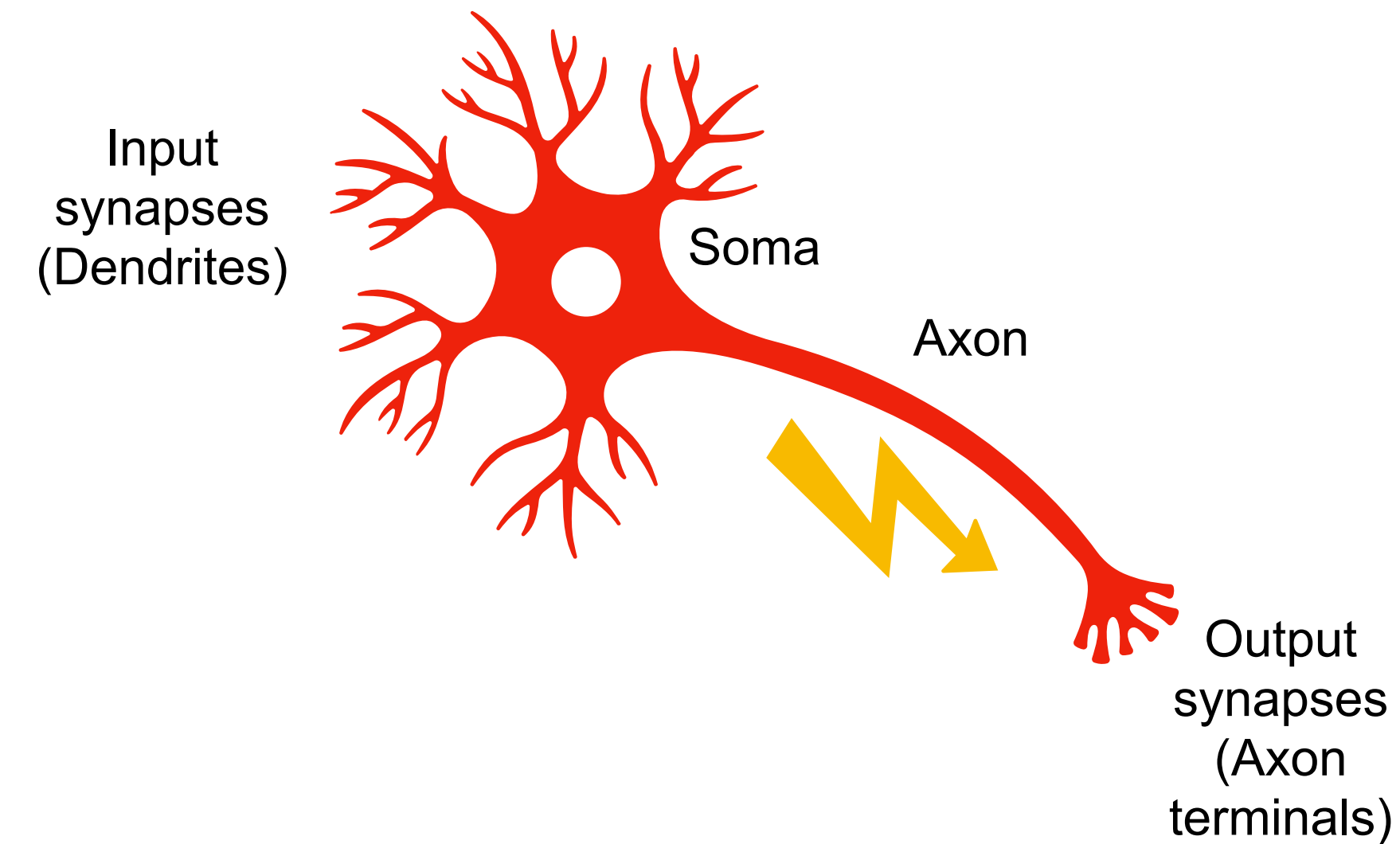
From: [Whole-brain annotation and multi-connectome cell typing of *Drosophila*](#)



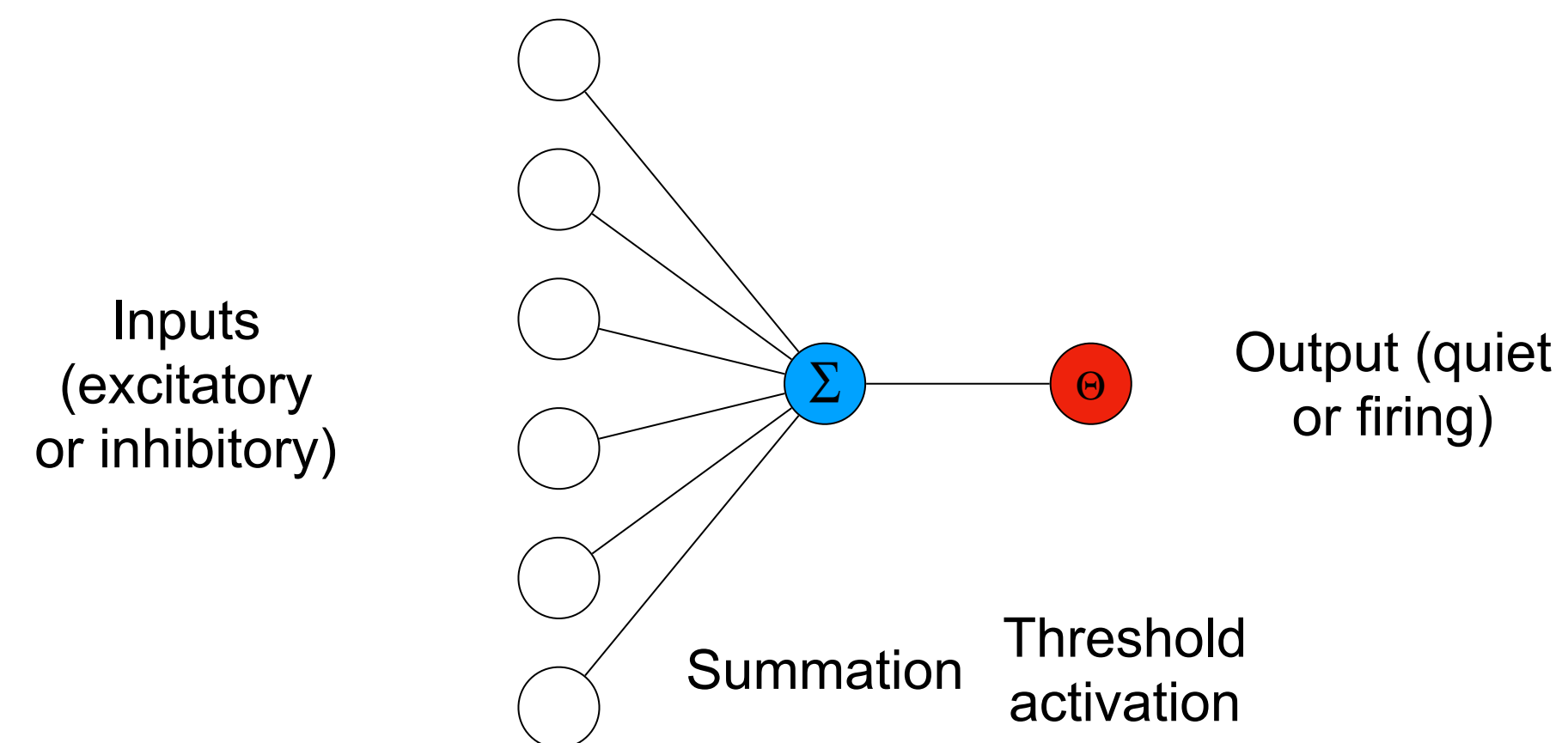
Schlegel et al, *Nature* (2024)

Artificial neurons and the perceptron model

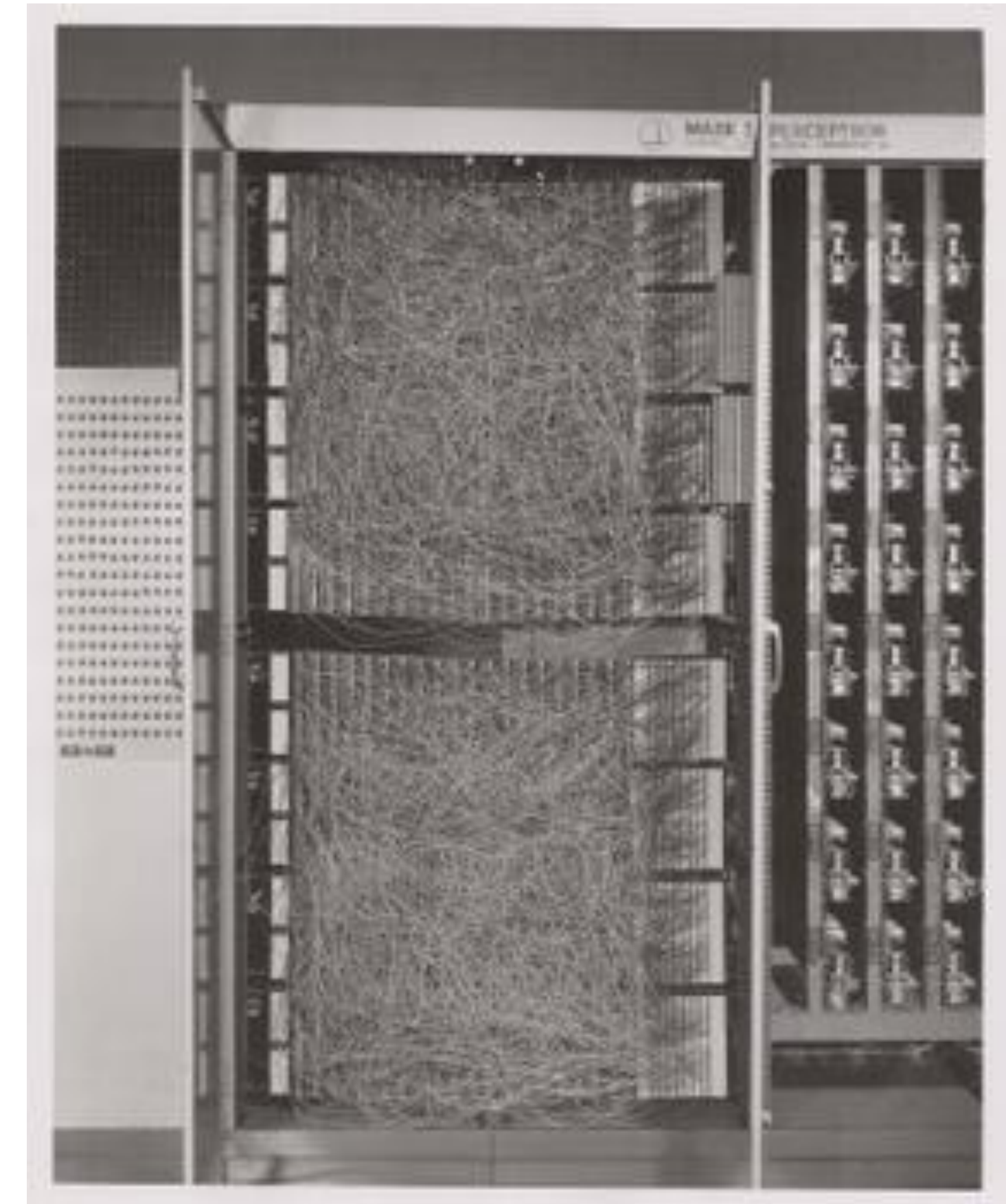
Biological Neuron



Artificial Neuron (McCulloch/Pitts, 1943)



Perceptron (Rosenblatt, 1957-58)

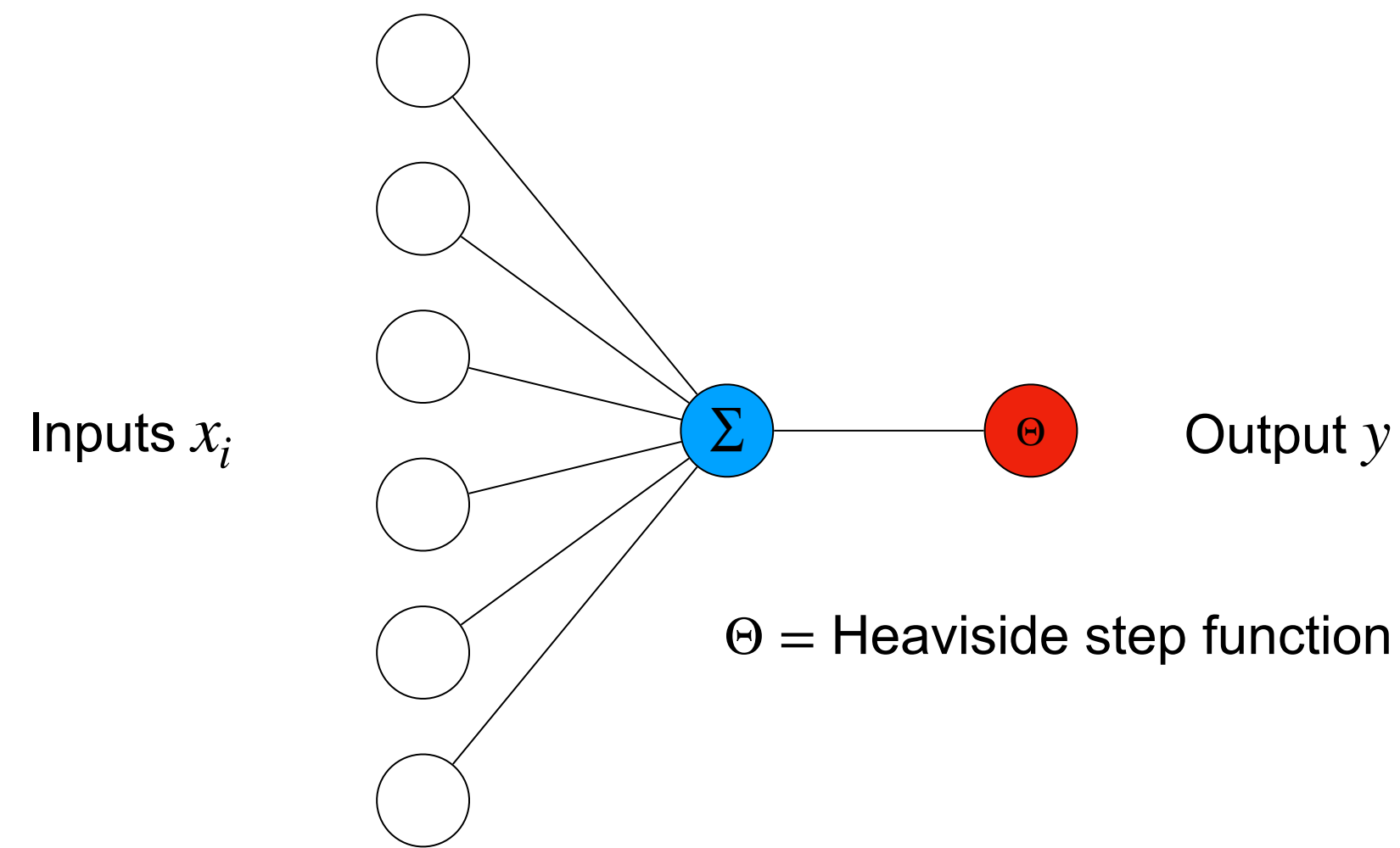


Mark I Perceptron machine (first hardware implementation of learning with an artificial neural network)

An artificial neural network is *not* a brain



Perceptrons for binary classification



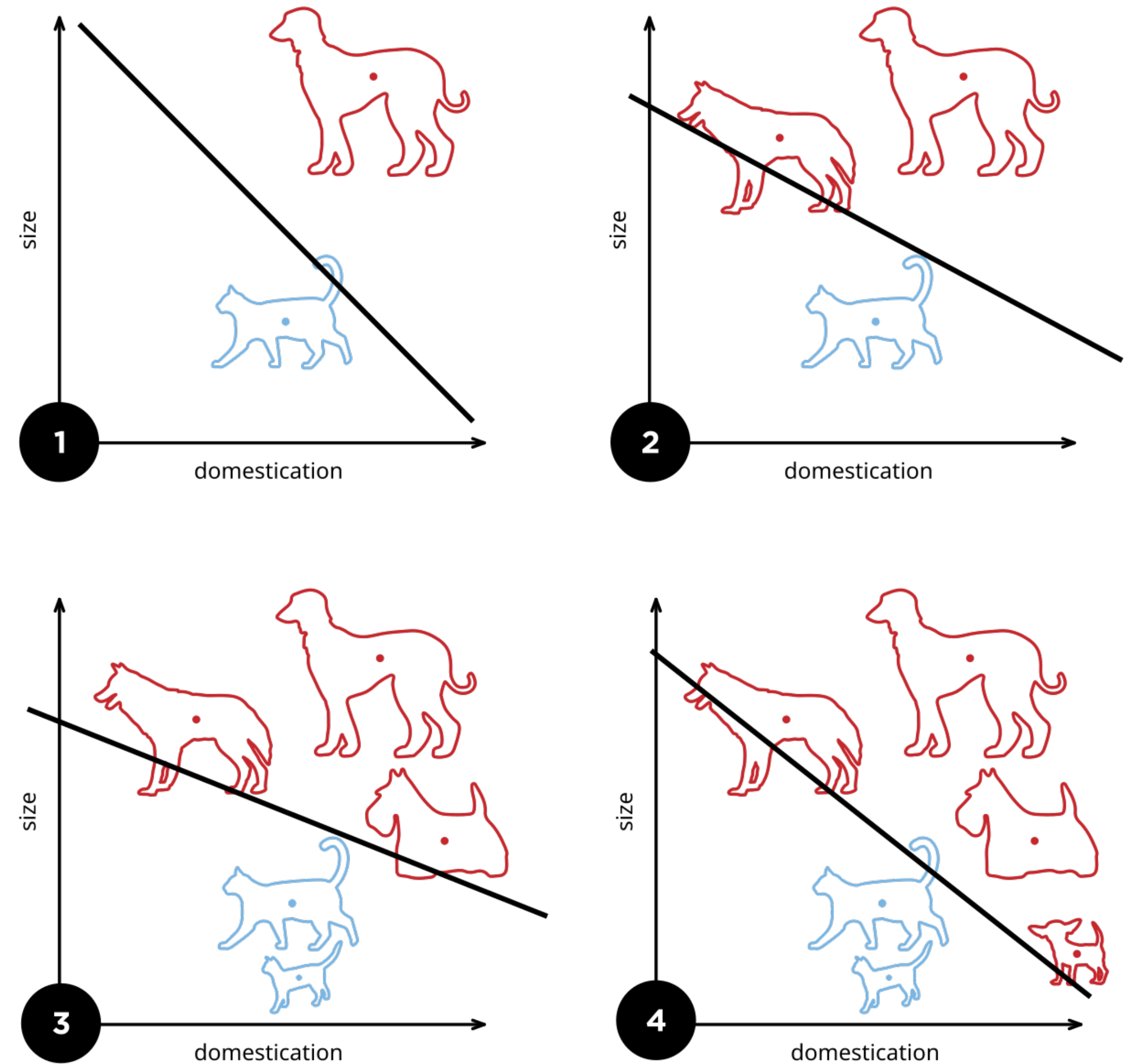
Binary activation

$$y = \begin{cases} 1 & \text{if } \sum_i w_i x_i + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

Update rule

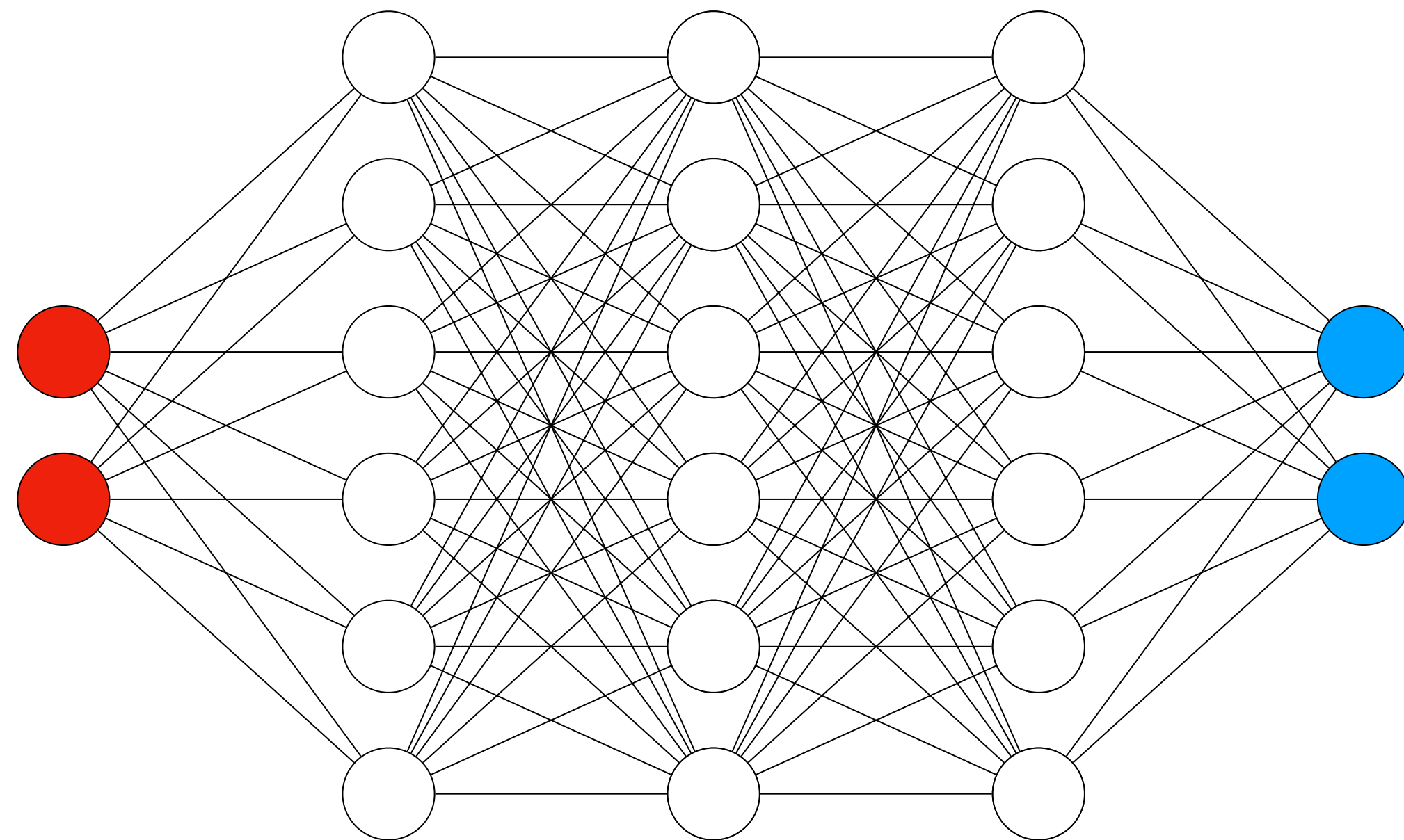
$$w_j \leftarrow w_j + \eta (y^i - \hat{y}^i) x_j^i$$

$$b \leftarrow b + \eta (y^i - \hat{y}^i)$$



<https://en.wikipedia.org/wiki/perceptron>

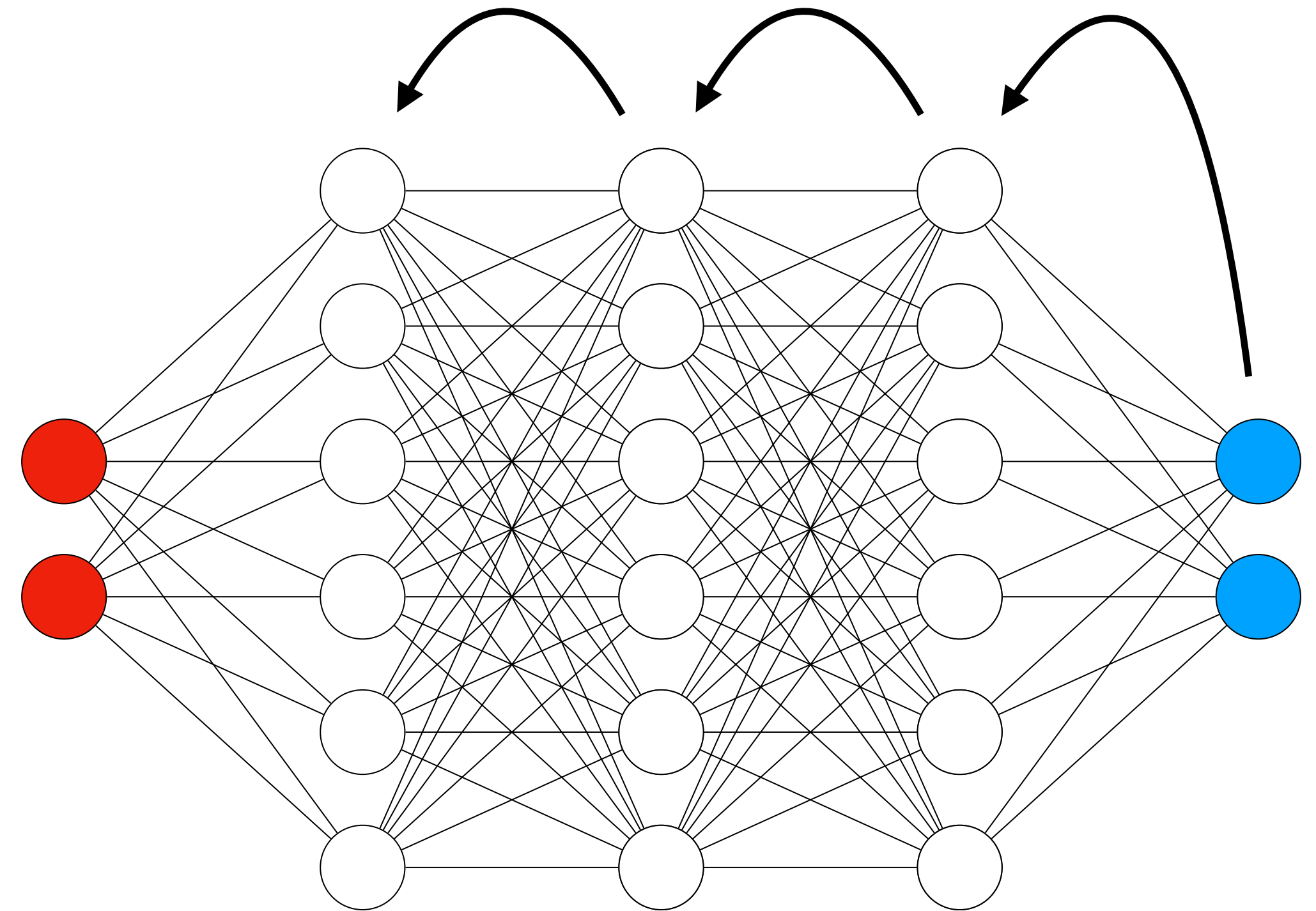
Multi-layer perceptrons



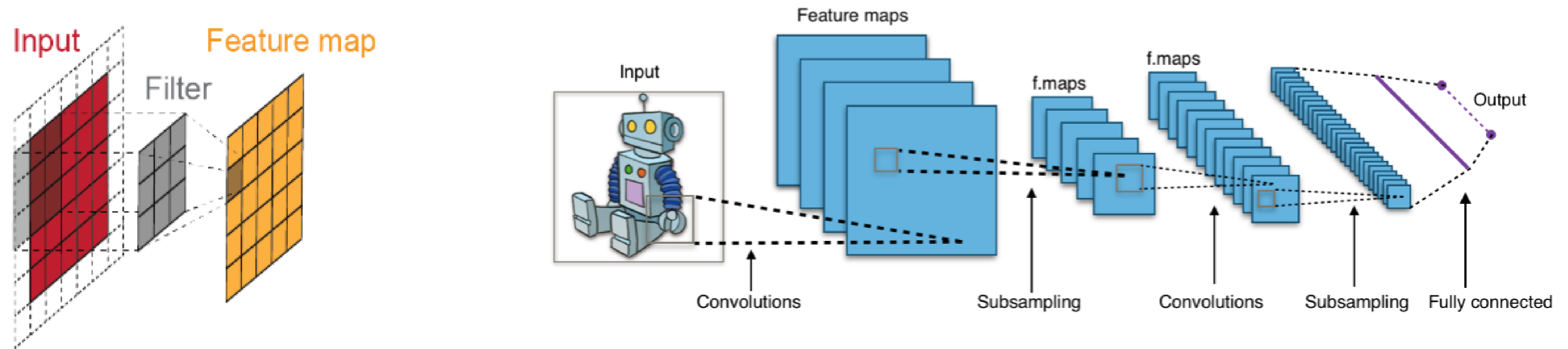
- A multilayer perceptron (MLP) has multiple hidden layers of neurons connected by nonlinear activations
- Early MLPs trained with stochastic gradient descent could perform nonlinear classification
- Hidden layers may learn internal *representations* of input data

Backpropagation

- Backpropagation (Werbos 1982, Rumelhart 1986, others) enabled efficient training of deep multilayer perceptrons
- Rumelhart, Hinton, and Williams (1985) showed that back propagation could lead to neural networks with *distributed* representations
- LeCun gave the first practical application of back propagation for handwritten digit classification (1989)

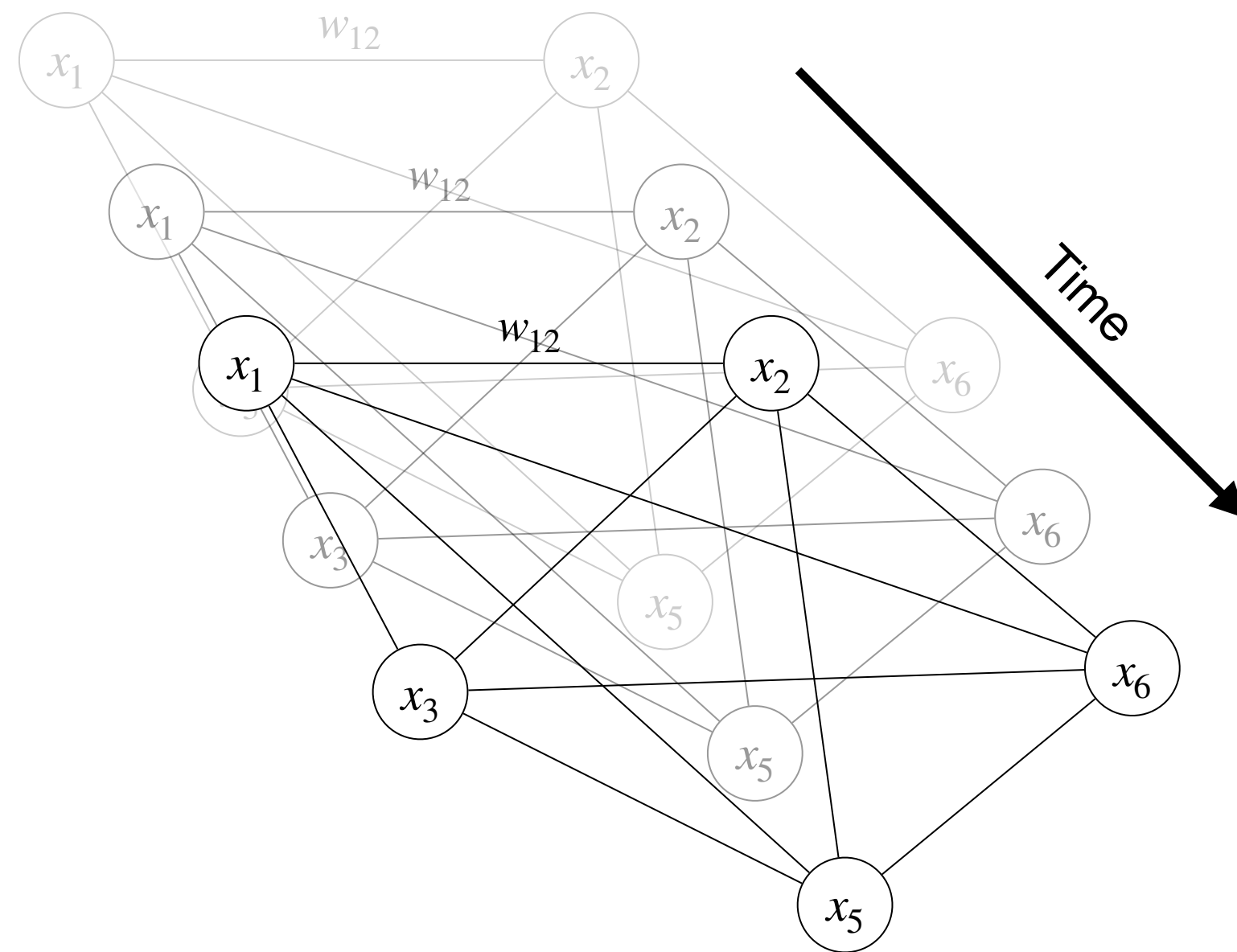


Convolutional neural networks (CNNs)

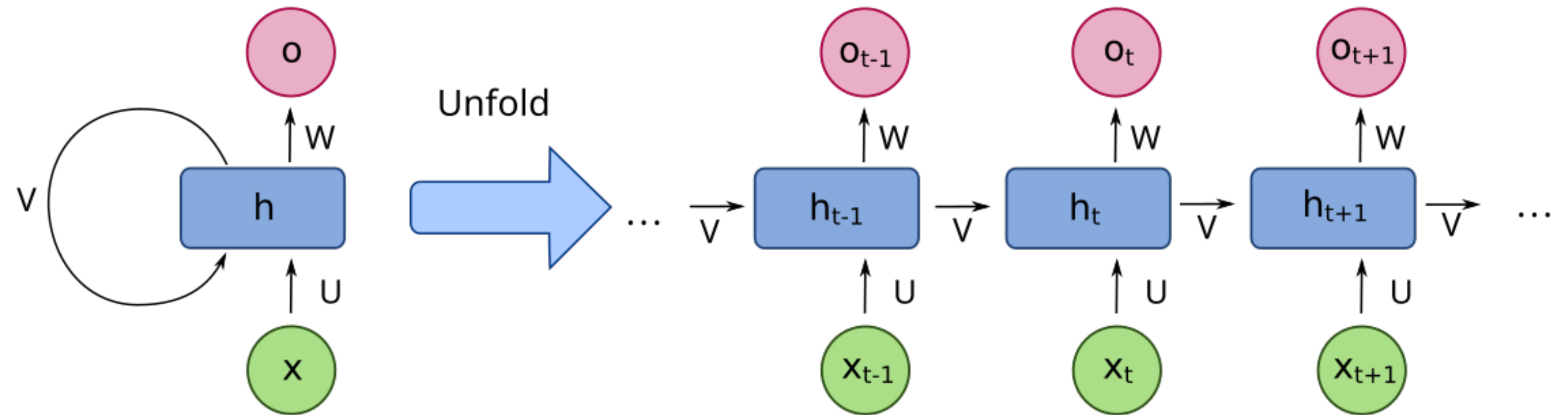


- CNNs were introduced by Fukushima (1979) and aimed to mimic the receptive field of neurons in the visual cortex
- CNNs share weights to learn hierarchical features, i.e. ones that become more complex with network depth

Recurrent neural networks (RNNs)



Hopfield Networks (1982)



- RNNs were introduced in the 1980s (in many forms) and exploited weight sharing in time
- Long short-term memory (LSTM, Schmidhuber 1995) improved on this design by addressing problems like vanishing gradients

ImageNet and the Deep Learning Boom



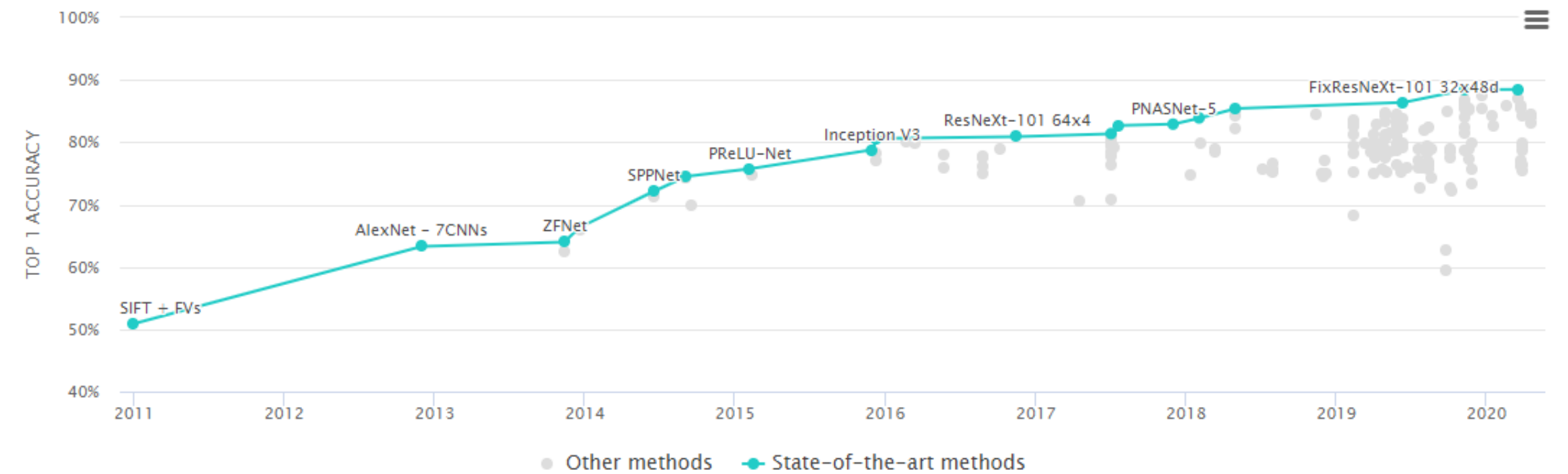
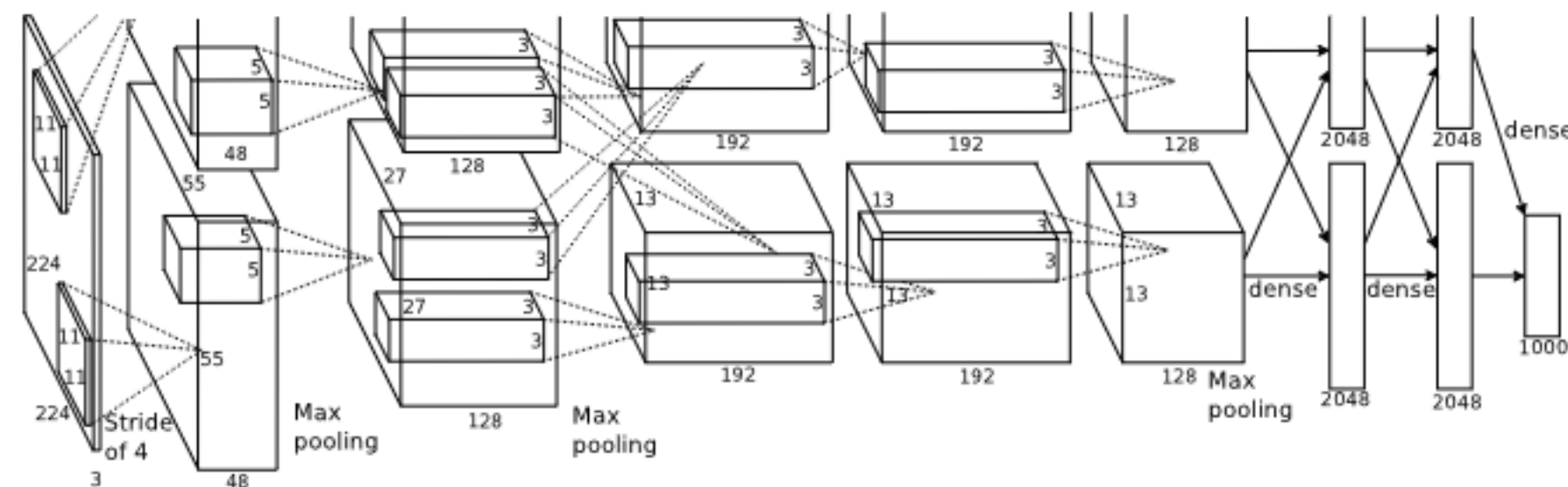
Fei Fei Li introduced the ImageNet database with >14 million images and labels (2009)

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca



- AlexNet (2012) showed the capability of deep neural networks, large data, and GPU training (code here: <https://github.com/computerhistory/AlexNet-Source-Code>)
- Residual networks (2015) enabled greater depth while avoiding vanishing gradients

Beyond labeled data to generative models

Building High-level Features Using Large Scale Unsupervised Learning

Quoc V. Le
Marc'Aurelio Ranzato
Rajat Monga
Matthieu Devin
Kai Chen
Greg S. Corrado
Jeff Dean
Andrew Y. Ng

QUOCLE@CS.STANFORD.EDU
RANZATO@GOOGLE.COM
RAJATMONGA@GOOGLE.COM
MDEVIN@GOOGLE.COM
KAICHEN@GOOGLE.COM
GCCRADO@GOOGLE.COM
JEFF@GOOGLE.COM
ANG@CS.STANFORD.EDU

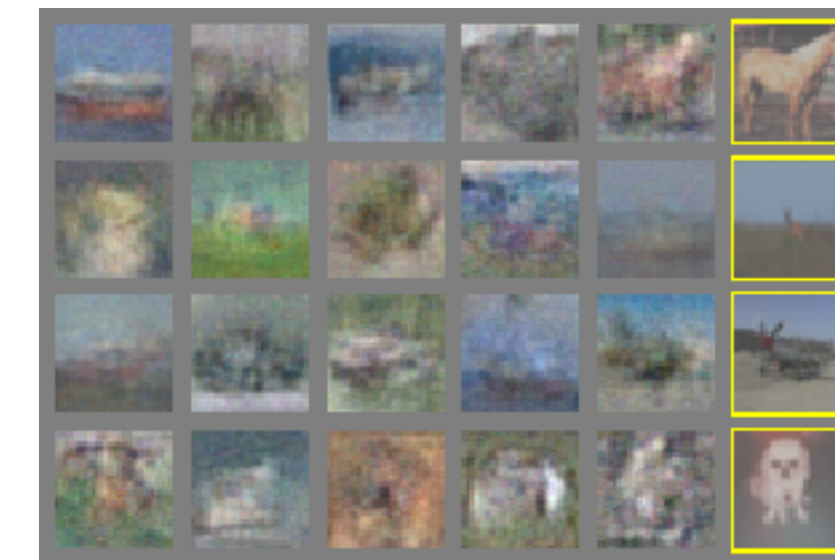
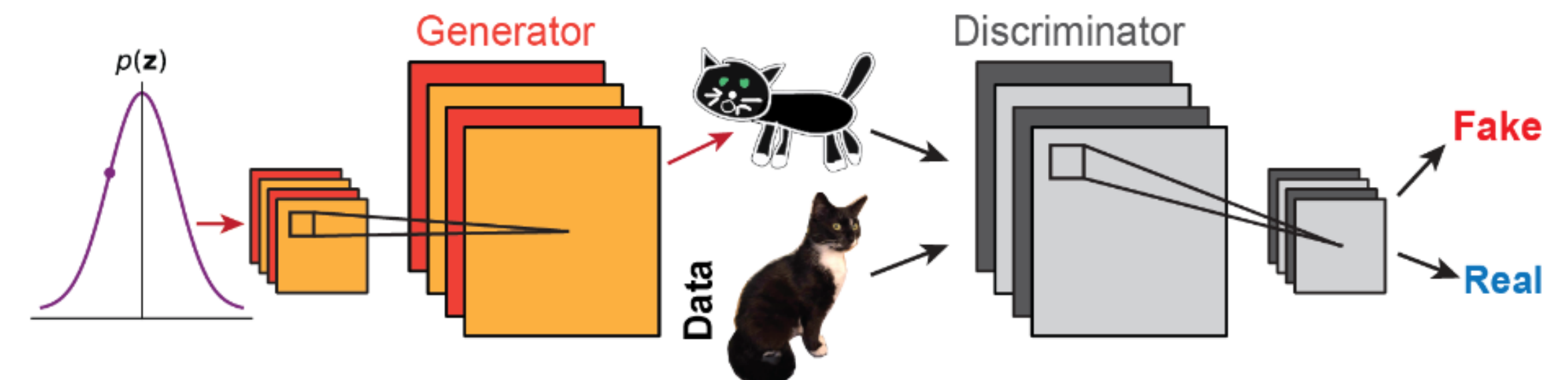
Auto-Encoding Variational Bayes

Diederik P. Kingma
Machine Learning Group
Universiteit van Amsterdam
dpkingma@gmail.com

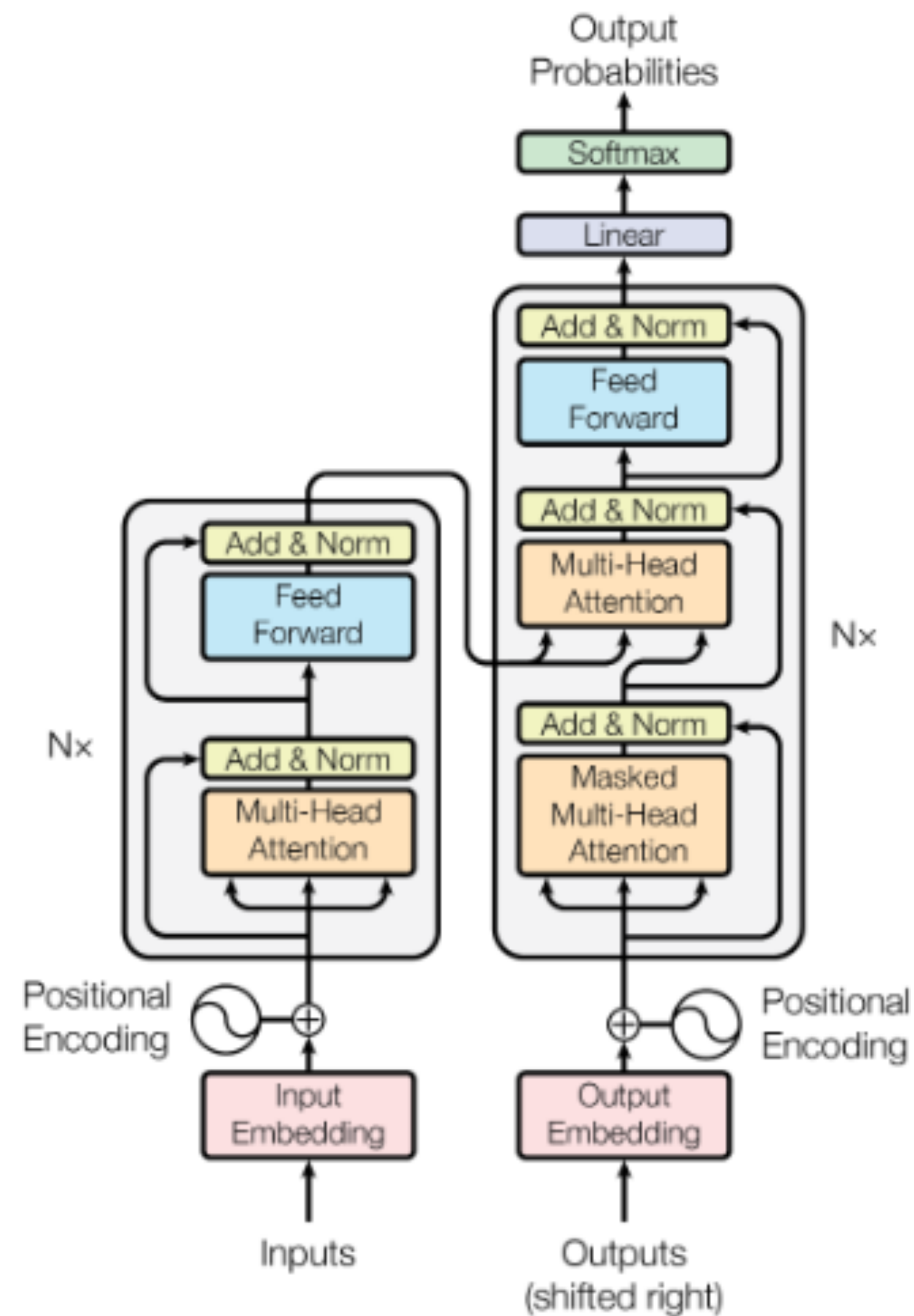
Max Welling
Machine Learning Group
Universiteit van Amsterdam
welling.max@gmail.com

Generative Adversarial Nets

Ian J. Goodfellow, Jean Pouget-Abadie*, Mehdi Mirza, Bing Xu, David Warde-Farley,
Sherjil Ozair†, Aaron Courville, Yoshua Bengio‡
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7



Transformers



The Transformer model (Vaswani et al, 2017) uses attention to efficiently train on sequence data

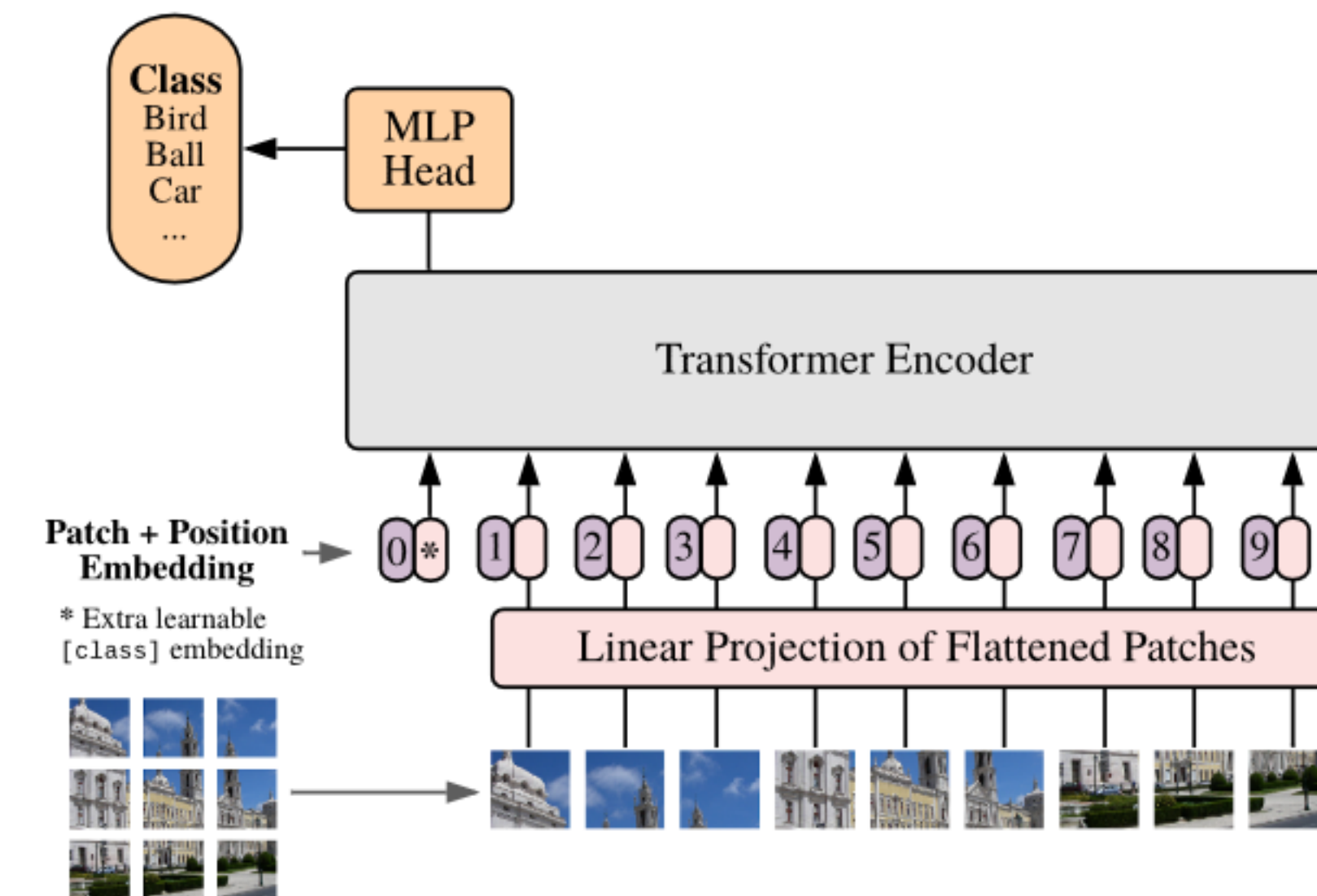
The cat plays with the ball

The cat plays with the ball

The cat plays with the ball

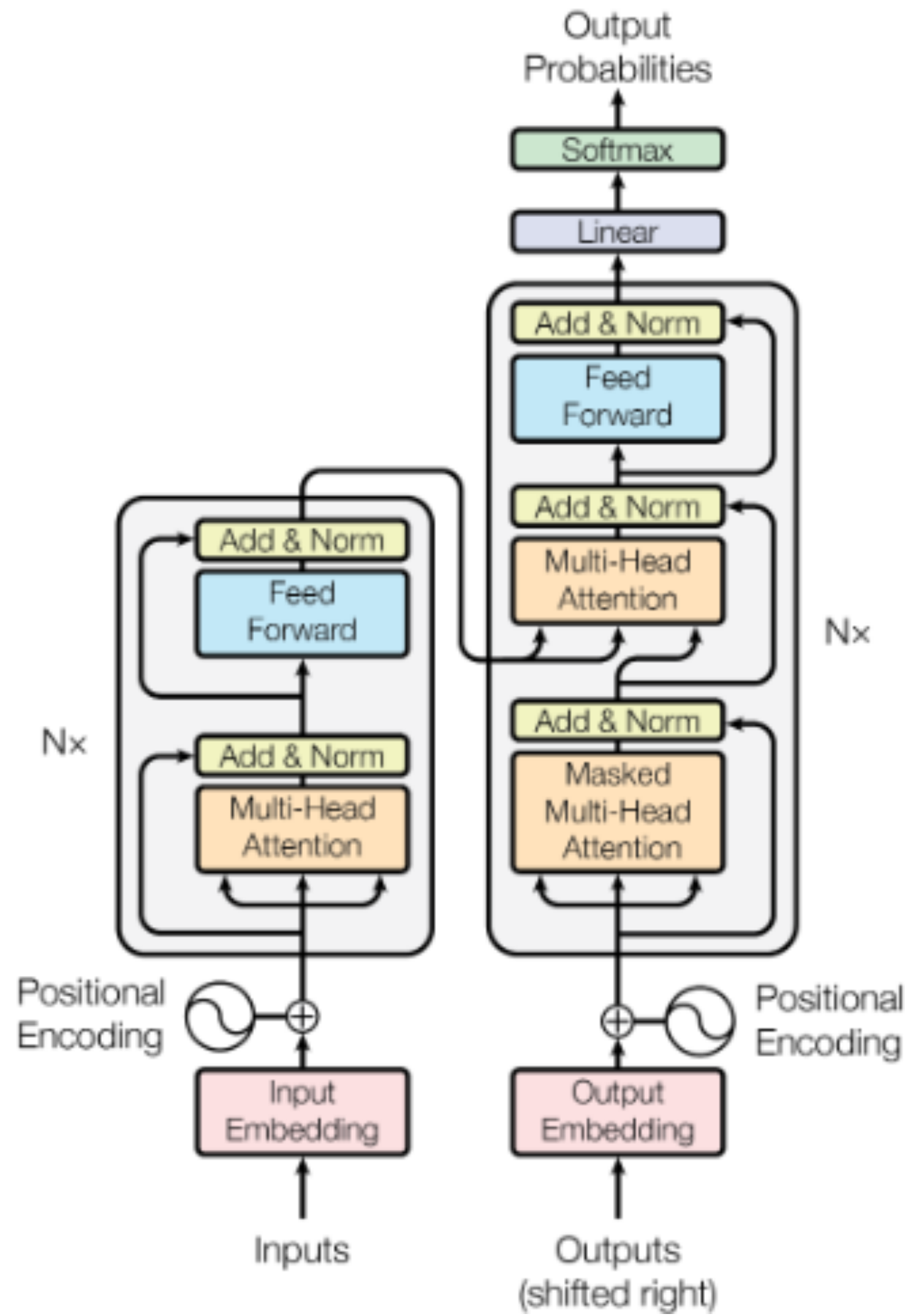
The cat plays with the ball

Attention (Schmidhuber, 1992) identifies alignment between “tokens”



Vision Transformers view an image as a “sentence” of smaller patches

Transformers



+



=

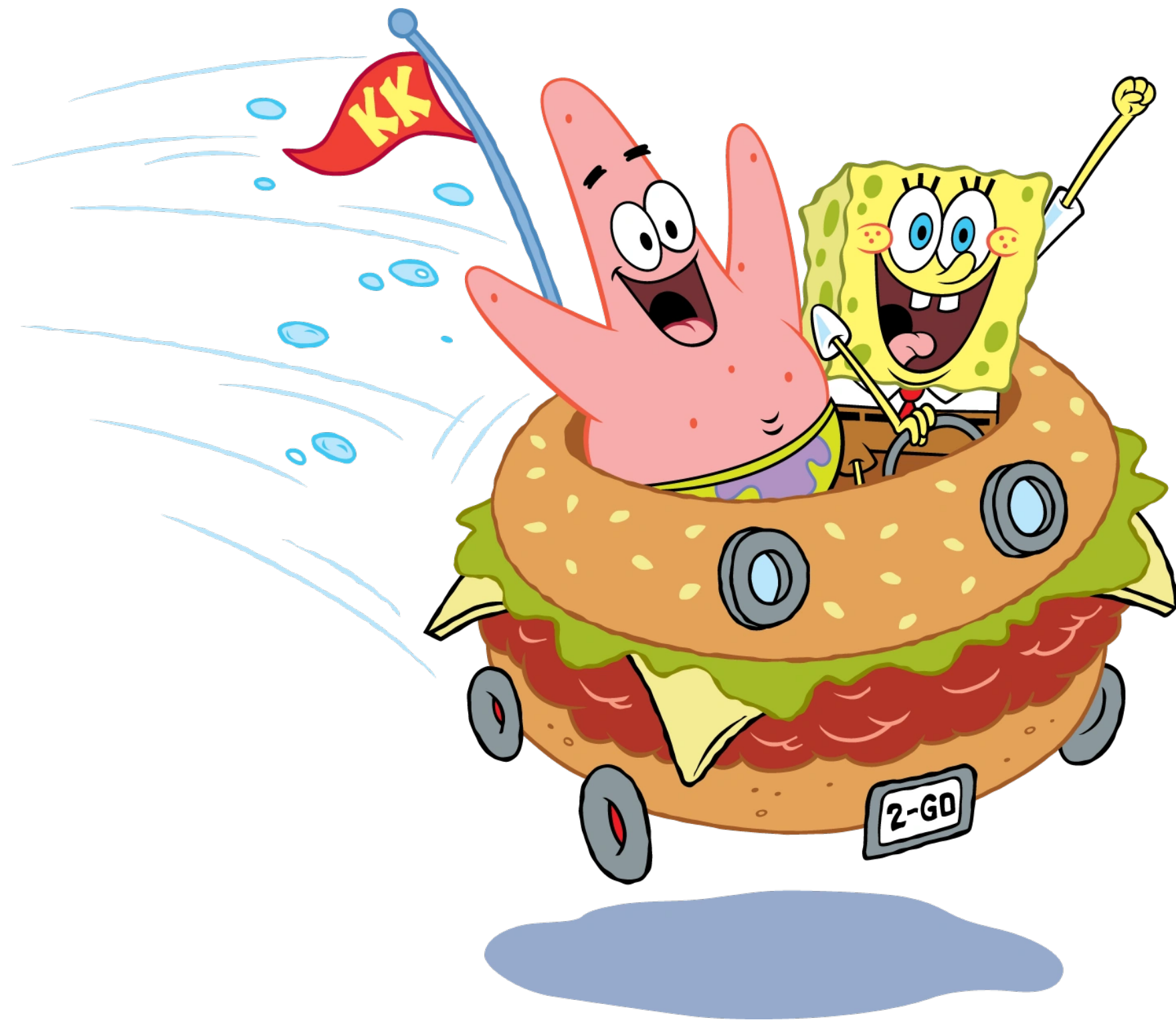


ChatGPT

What will we do in this class?

Driving a car vs. Working on the engine

Using AI models



This class



A typical problem



We have

- A **dataset** (X, y) : a matrix of inputs X and a vector of outputs y
- A **model** $f_\theta : X \rightarrow y$ that maps inputs to outputs using parameters θ
- A **cost function** $C[f_\theta(X), y]$ that tells us how well the model does what we want

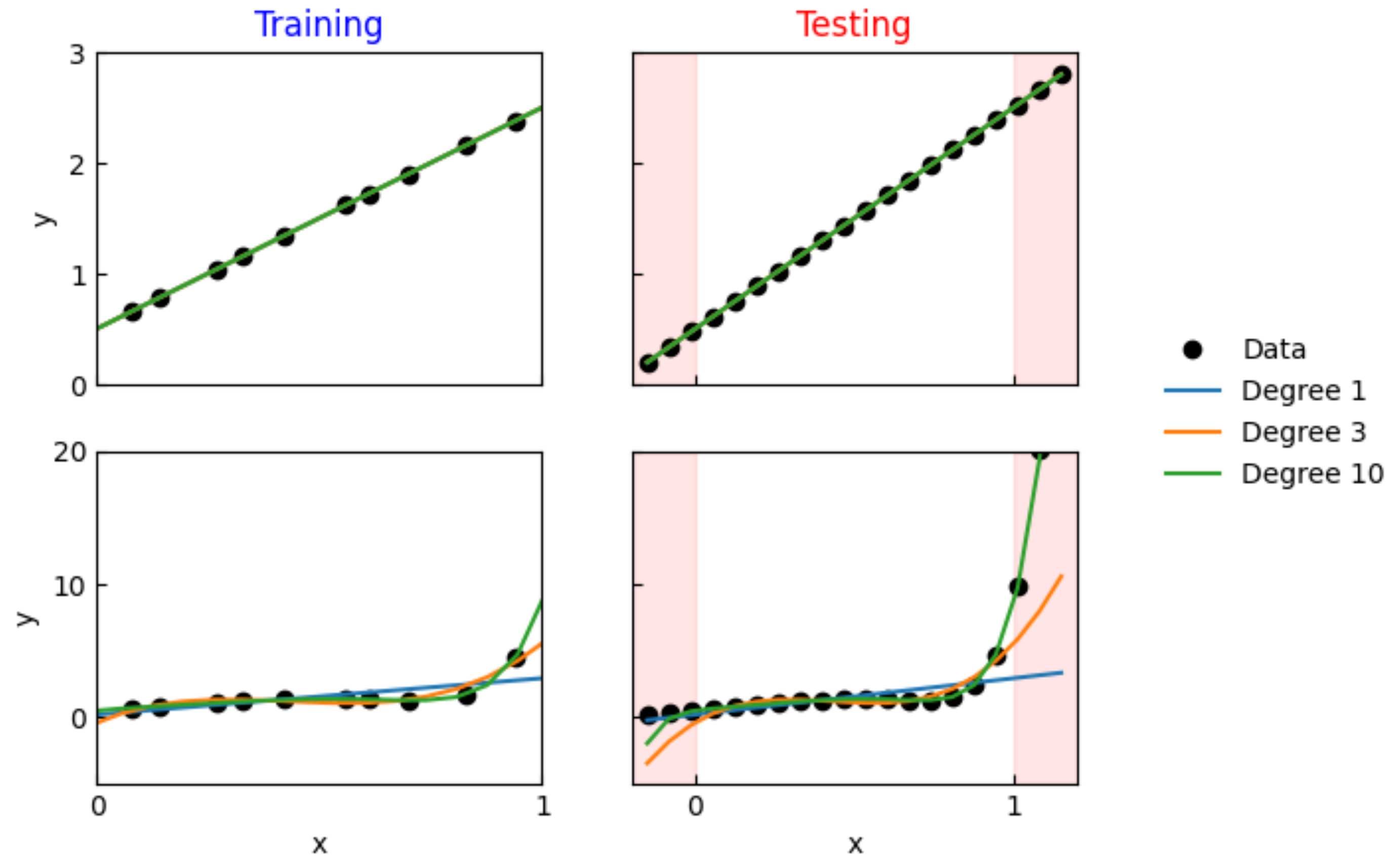
Our goal is to find the parameters θ that minimizes the cost function: $\theta^* = \operatorname{argmin} C[f_\theta(X), y]$

- We will develop the machinery to define good models f_θ , train them using different types of data, and evaluate how well they perform on new examples
- We will cover theory and practice using a common open-source deep learning framework (PyTorch)

✱ It is not always the case that one has either a good dataset or a good cost function...

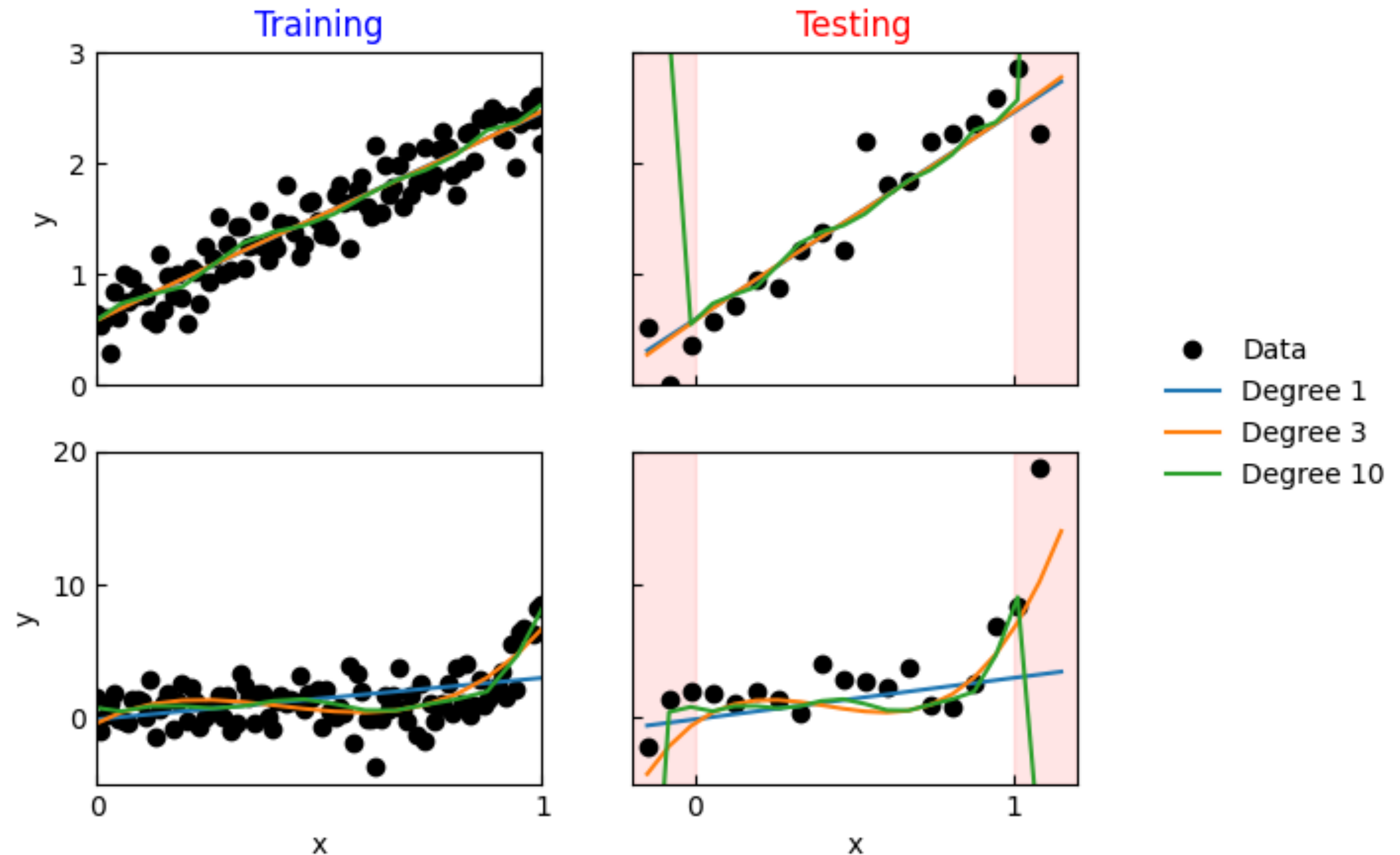
A motivating example - polynomial fitting

- The **best** model is always the one that most resembles the data-generating distribution
- This **fits** the training data and **predicts** the testing data

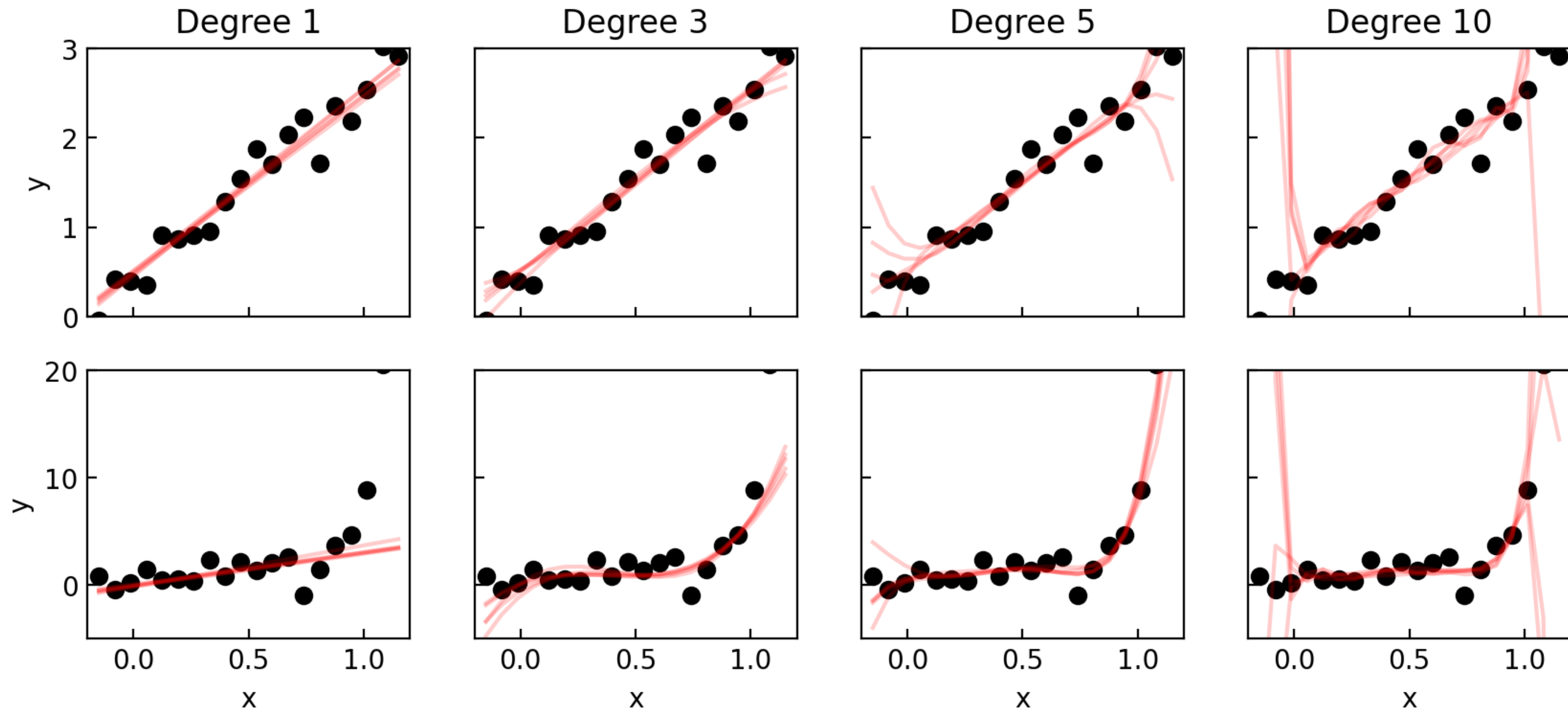


A motivating example - polynomial fitting

- Noise in data may obscure the data distribution
- Expressive models may attempt to capture noise as genuine data patterns
- This can **limit** fit quality as well as test predictions



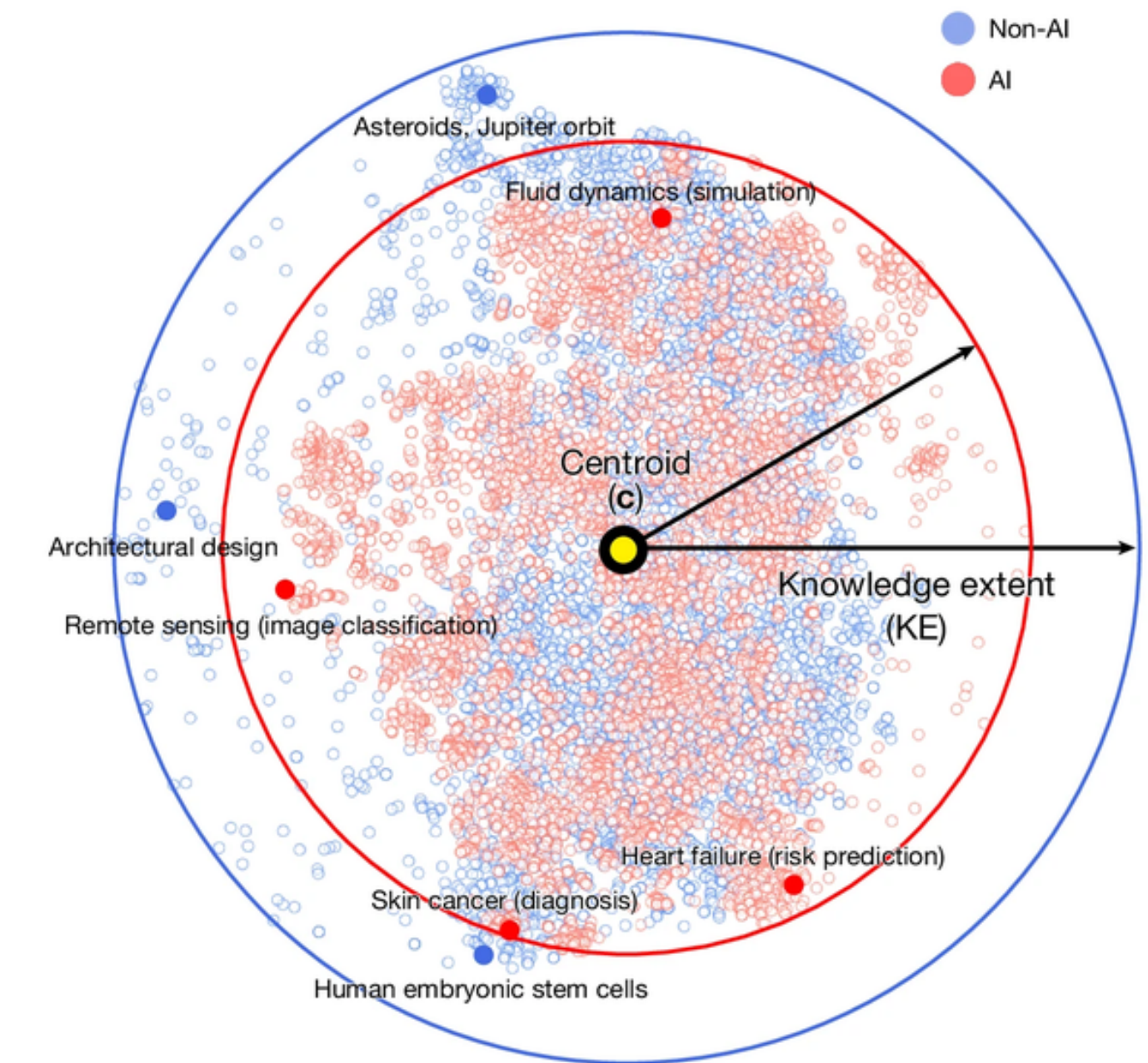
A motivating example - polynomial fitting



- Models with more parameters can show high variation and inconsistent behavior
- Deep NNs have many, many, many (!) parameters

Challenges with deep learning

- Bias-variance tradeoff: neural networks are very expressive (low bias, high variance)
- Not all problems have large well-curated datasets
- Training and inference for SOTA models are expensive!
- We don't *really* know how deep neural networks work yet



Hao, Xu, Li, & Evans. *Nature* (2026)

There Will Be a Scientific Theory of Deep Learning

Jamie Simon* UC Berkeley and Imbue		Daniel Kunin UC Berkeley
Alexander Atanasov Harvard University	Enric Boix-Adserà University of Pennsylvania	Blake Bordelon Harvard University
Jeremy Cohen Flatiron Institute	Nikhil Ghosh Flatiron Institute	Florentin Guth NYU & Flatiron Institute
Arthur Jacot New York University	Mason Kamb Stanford University	Dhruva Karkada UC Berkeley
Eric J. Michaud Astera Institute	Berkan Ottlik University of Pennsylvania	Joseph Turnbull UC Berkeley

Course goals

- Understanding - “What’s going on with these networks?”
 - You should be able to create, debug, train, test, and tweak NN variants including MLP, CNN, RNN, GAN, GNN, etc
- Applications - “How do I use them?”
 - Teach you the advantages and disadvantages of different strategies for fitting networks, software and hardware architectures, and more knowledge required to successfully create your own nets
- Insights - “Where are they going now?”
 - We’ll show some of the newer innovations and give you the tools to experiment

That's all!

- Please make a note of the course website: <https://colen-teaching.github.io/DATA442-Fall-2026>
- Course notes, in class demos, and assignments will be posted periodically on the above website
- If you would like an HPC account, submit a request at <https://hpc.wm.edu/acctreq/>
- Please try to install PyTorch on a machine of your choice! See here for instructions: <https://pytorch.org/get-started/locally/>