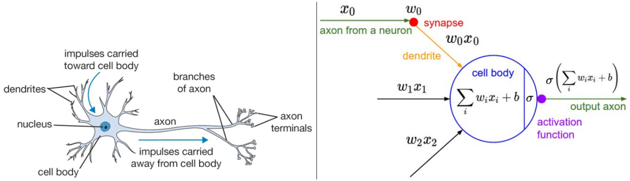
Neural Networks: Old and New

Ju Sun
Computer Science & Engineering
University of Minnesota, Twin Cities

January 22, 2026

Outline

Model of biological neurons



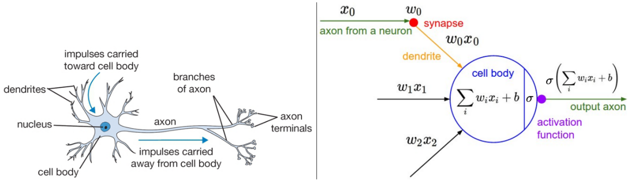
A cartoon drawing of a biological neuron (left) and its mathematical model (right).

Credit: Stanford CS231N

Biologically ...

- Each neuron receives signals from its **dendrites**
- Each neuron outputs signals via its single **axon**
- The axon branches out and connects via **synapese** to dendrites of other neurons

Model of biological neurons



A cartoon drawing of a biological neuron (left) and its mathematical model (right).

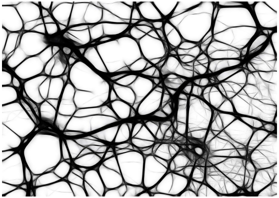
Credit: Stanford CS231N

Mathematically ...

- Each neuron receives x_i's from its **dendrites**
- x_i's weighted by w_i's (synaptic strengths) and summed $\sum_i w_i x_i$
- The neuron fires only when the combined signal is above a certain threshold: $\sum_i w_i x_i + b$
- Fire rate is modeled by an **activation function** σ, i.e., outputting $\sigma\left(\sum_i w_i x_i + b\right)$

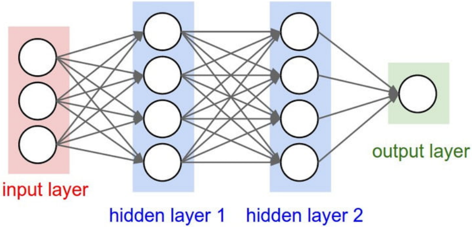
Artificial neural networks

Brain neural networks



~ 86-billion neurons (Credit: Max Pixel)

Artificial neural networks

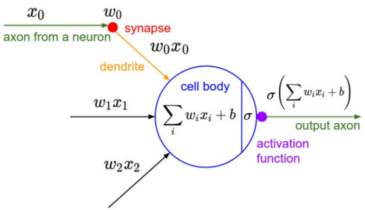


Why called **artificial**?

- (Over-)simplification on neural level
- (Over-)simplification on connection level

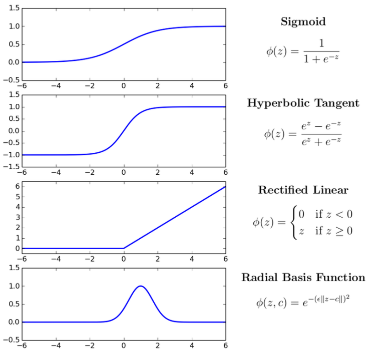
In this course, neural networks are always artificial.

Outline



$$\sigma\left(\sum_i w_i x_i + b\right) = \sigma\left(\boldsymbol{w}^\intercal \boldsymbol{x} + b\right)$$

Examples of activation function σ

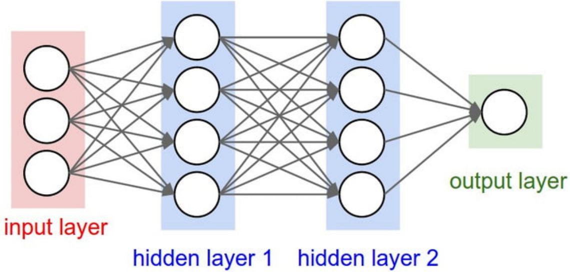


Credit: [Hughes and Correll, 2016]

Neural networks

One neuron: $\sigma\left(\boldsymbol{w}^\intercal \boldsymbol{x} + b\right)$

Neural networks (NN): **structured** organization of artificial neurons



$\boldsymbol{w}$'s and b's are unknown and need to be learned

Many models in machine learning **are** neural networks

Supervised learning in a nutshell

Supervised Learning

- Gather training data $(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)$
- Choose a family of functions, e.g., $\mathcal{H}$, so that there is $f \in \mathcal{H}$ to ensure $\boldsymbol{y}_i \approx f(\boldsymbol{x}_i)$ for all i
- Set up a loss function ℓ to measure the approximation quality
- Find an $f \in \mathcal{H}$ to minimize the average loss

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(\boldsymbol{y}_i, f(\boldsymbol{x}_i))$$

... known as **empirical risk minimization** (ERM) framework in learning theory

Supervised learning meets NNs

Supervised Learning from NN viewpoint

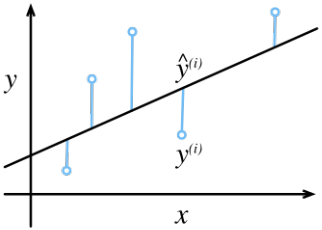
- Gather training data $(\boldsymbol{x}_1, \boldsymbol{y}_1), \dots, (\boldsymbol{x}_n, \boldsymbol{y}_n)$
- Choose a NN with k neurons, so that there is a group of weights, e.g., $(\boldsymbol{w}_1, \dots, \boldsymbol{w}_k, b_1, \dots, b_k)$, to ensure

$$\boldsymbol{y}_i \approx \{\mathsf{NN}(\boldsymbol{w}_1, \dots, \boldsymbol{w}_k, b_1, \dots, b_k)\}(\boldsymbol{x}_i) \quad \forall i$$

- Set up a loss function ℓ to measure the approximation quality
- Find weights $(\boldsymbol{w}_1, \dots, \boldsymbol{w}_k, b_1, \dots, b_k)$ to minimize the average loss

$$\min_{\boldsymbol{w}'s, b's} \frac{1}{n} \sum_{i=1}^{n} \ell\left[\boldsymbol{y}_i, \{\mathsf{NN}(\boldsymbol{w}_1, \dots, \boldsymbol{w}_k, b_1, \dots, b_k)\}(\boldsymbol{x}_i)\right]$$

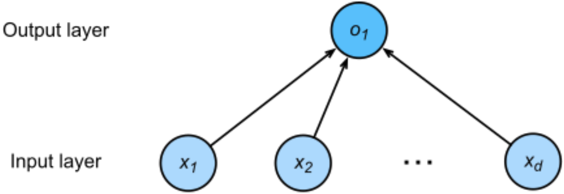
Linear regression



Credit: D2L

- Data: $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$, $\boldsymbol{x}_i \in \mathbb{R}^d$
- Model: $y_i \approx \widehat{y}_i \doteq \boldsymbol{w}^\intercal \boldsymbol{x}_i + b$
- Loss: $\|y_i - \widehat{y}_i\|_2^2$
- Optimization:

$$\min_{\boldsymbol{w}, b} \frac{1}{n} \sum_{i=1}^{n} \|y_i - (\boldsymbol{w}^\intercal \boldsymbol{x}_i + b)\|_2^2$$



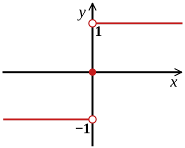
Credit: D2L

σ is the identity function

Frank Rosenblatt

(1928–1971)

- Data: $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$, $\boldsymbol{x}_i \in \mathbb{R}^d$, $y_i \in \{+1, -1\}$
- Model: $y_i \approx \sigma(\boldsymbol{w}^\intercal \boldsymbol{x}_i + b)$, σ sign function

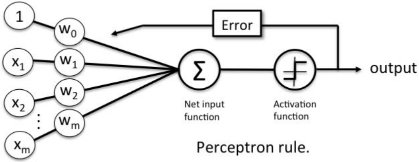


- Loss: $\mathbf{1}\{y \neq \hat{y}\}$
- Optimization:

$$\min_{\boldsymbol{w}, b} \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{y_i \neq \sigma(\boldsymbol{w}^\intercal \boldsymbol{x}_i + b)\}$$

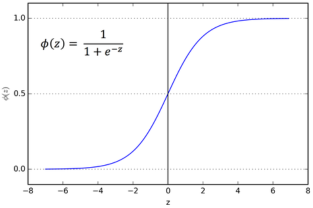
Perceptron

Perceptron is a single artificial neuron for **binary classification**



dominated early AI (50's – 60's)

Logistic regression is similar but with **sigmoid** activation



Softmax regression

- Data: $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$, $\boldsymbol{x}_i \in \mathbb{R}^d$, $y_i \in \{L_1, \ldots, L_p\}$, i.e., multiclass classification problem
- Data preprocessing: labels into vectors via one-hot encoding

$$L_k \Longrightarrow [\underbrace{0, \ldots, 0}_{k-1\ 0's}, 1, \underbrace{0, \ldots, 0}_{p-k\ 0's}]^\intercal$$

So: $y_i \Longrightarrow \boldsymbol{y}_i$

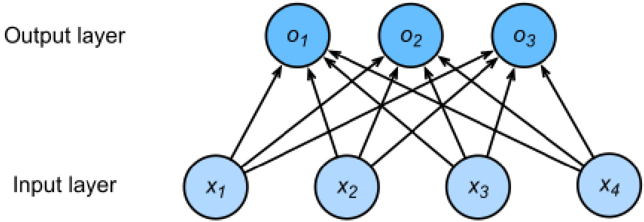
- Model: $\boldsymbol{y}_i \approx \sigma(\boldsymbol{W}^\intercal \boldsymbol{x}_i + \boldsymbol{b})$, here σ is the softmax function (maps vectors to vectors): for $\boldsymbol{z} \in \mathbb{R}^p$,

$$\boldsymbol{z} \mapsto \left[\frac{e^{z_1}}{\sum_j e^{z_j}}, \ldots, \frac{e^{z_p}}{\sum_j e^{z_j}}\right]^\intercal.$$

- Loss: **cross-entropy loss** $-\sum_j y_j \log \hat{y}_j$
- Optimization ...

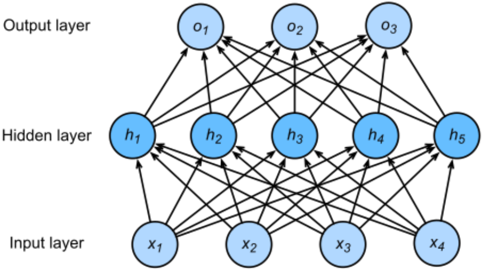
Softmax regression

… for multiclass classification



Credit: D2L

Multilayer perceptrons (MLP)



Credit: D2L

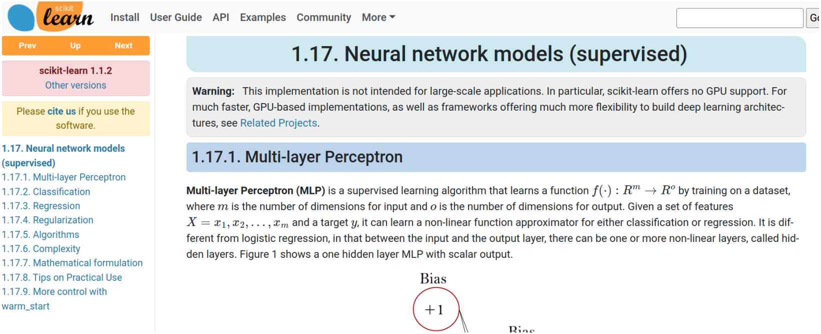
Model: $\boldsymbol{y}_i \approx \sigma_2\left(\boldsymbol{W}_2^\intercal \sigma_1(\boldsymbol{W}_1^\intercal \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2\right)$

Also called **fully-connected networks**

Modern NNs:

- many hidden layers: deep neural networks (DNNs)
- refined/structured connection and/or activations (convolutional/recurrent/graph/... NNs)

MLP in scikit-learn



Install User Guide API Examples Community More

Prev Up Next

scikit-learn 1.1.2
Other versions

Please **cite us** if you use the software.

1.17. Neural network models (supervised)

Warning: This implementation is not intended for large-scale applications. In particular, scikit-learn offers no GPU support. For much faster, GPU-based implementations, as well as frameworks offering much more flexibility to build deep learning architectures, see Related Projects.

1.17.1. Multi-layer Perceptron

Multi-layer Perceptron (MLP) is a supervised learning algorithm that learns a function $f(\cdot): R^m \rightarrow R^o$ by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output. Given a set of features $X = x_1, x_2, \ldots, x_m$ and a target y, it can learn a non-linear function approximator for either classification or regression. It is different from logistic regression, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers. Figure 1 shows a one hidden layer MLP with scalar output.

https://scikit-learn.org/stable/modules/neural_networks_supervised.html

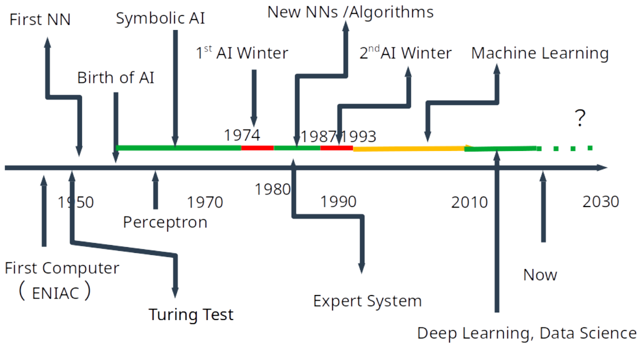
They're all (shallow) NNs

- Linear regression
- Perception and Logistic regression
- Softmax regression
- Multilayer perceptron (feedforward NNs)
- Support vector machines (SVM)
- PCA (autoencoder)
- Matrix factorization

see, e.g., Chapter 2 of [Aggarwal, 2018].

Outline

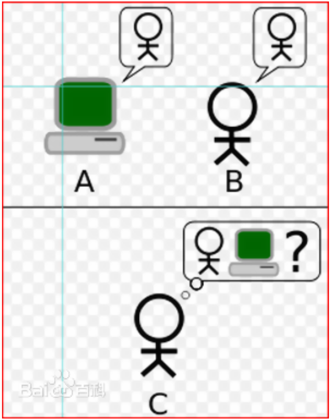
Birth of AI



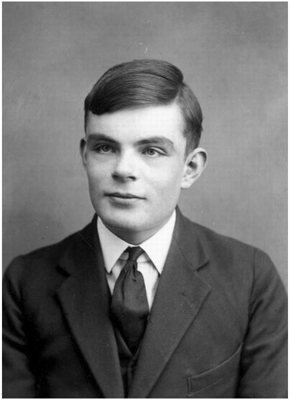
- Crucial precursors: first computer, Turing test
- 1956: Dartmouth Artificial Intelligence Summer Research Project — Birth of AI

What's intelligence?

Turing test

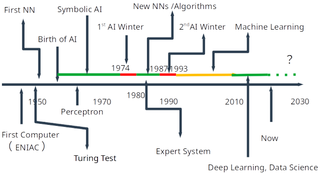


Turing Test

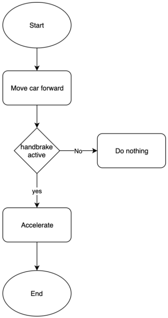


Alan Turing (1912–1954)

First golden age

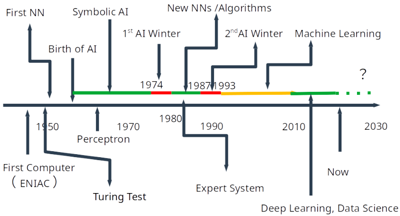


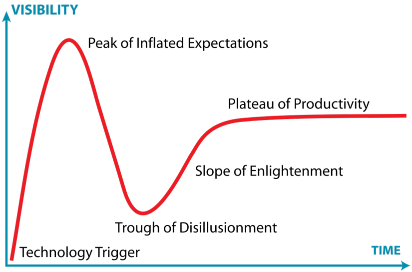
Symbolic AI: modeling general logic and reasoning



rules for recognizing dogs?

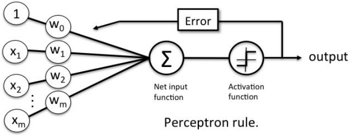
First AI winter



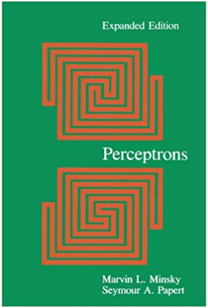


Gartner hype cycle

Perceptron



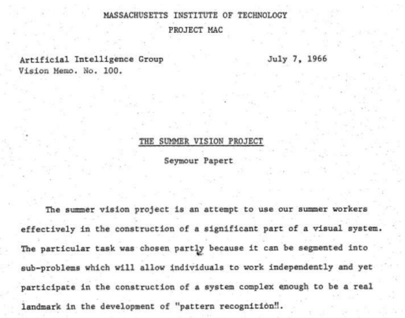
invented 1962



written in 1969, end of Perceptron era

Marvin Minsky (1927–2016)

Birth of computer vision



MASSACHUSETTS INSTITUTE OF TECHNOLOGY
PROJECT MAC

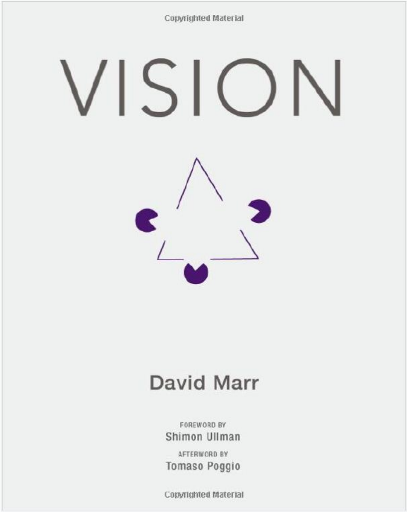
Artificial Intelligence Group July 7, 1966
Vision Memo. No. 100.

THE SUMMER VISION PROJECT

Seymour Papert

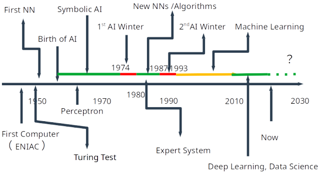
The summer vision project is an attempt to use our summer workers effectively in the construction of a significant part of a visual system. The particular task was chosen partly because it can be segmented into sub-problems which will allow individuals to work independently and yet participate in the construction of a system complex enough to be a real landmark in the development of "pattern recognition".

1966

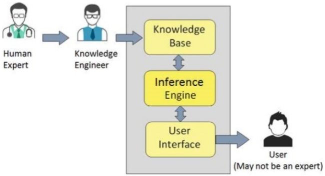


around 1980

Second golden age



expert system—building in domain-specific knowledge



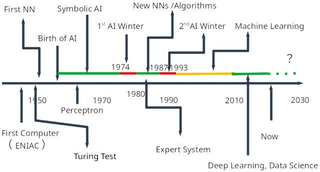
Can we build comprehensive knowledge bases and know all rules?

Big bang in DNNs

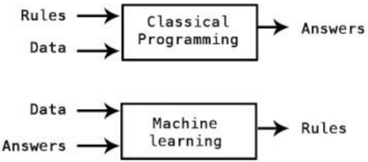
Key ingredients of DL have been in place for 25-30 years:

Landmark	Emblem	Epoch
Neocognitron	Fukushima	1980
CNN	Le Cun	mid 1980s'
Backprop	Hinton	mid 1980's
SGD	Le Cun, Bengio etc	mid 1990's
Various	Schmidhuber	mid 1980's
CTF	*DARPA etc*	*mid 1980's*

After 2nd AI winter



Machine learning takes over ...



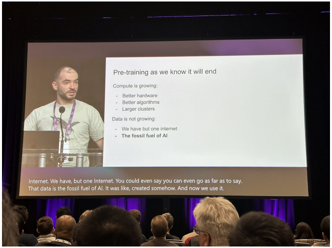
rules learned from data, or **data-driven**

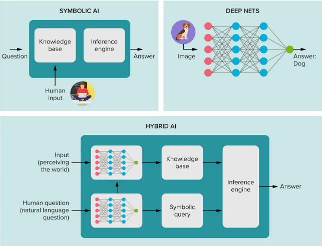
Golden age of Machine learning

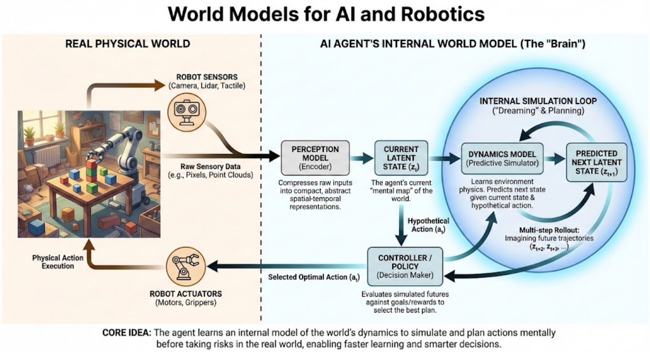
Starting 1990's

- Support vector machines (SVM)
- Adaboost
- Decision trees and random forests
- Deep learning (2010's)
- ...

What’s next?

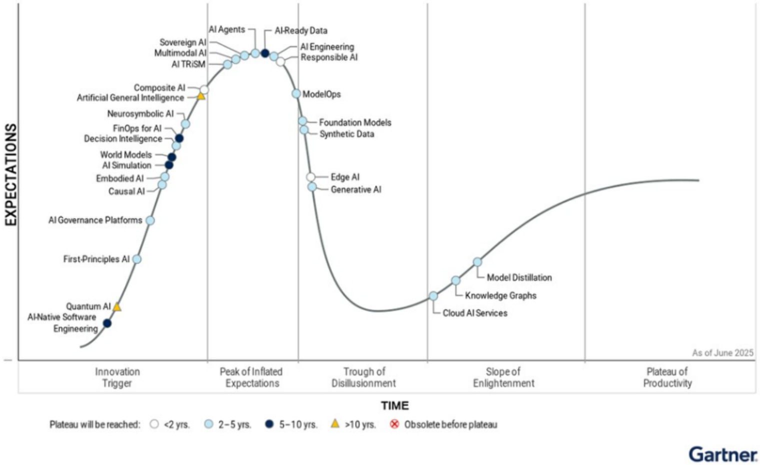

Pre-training as we know it will end
Compute is growing:
- Better hardware
- Better algorithms
- Larger clusters
Data is not growing:
- We have but one internet
- The fossil fuel of AI
Internet. We have, but one Internet. You could even say you can even go as far as to say.
That data is the fossil fuel of AI. It was like, created somehow. And now we use it.


SYMBOLIC AI
Question
Knowledge base
Inference engine
Answer
Human input
DEEP NETS
Image
Answer: Dog
HYBRID AI
Input (perceiving the world)
Human question (natural language question)
Knowledge base
Symbolic query
Inference engine
Answer


World Models for AI and Robotics
REAL PHYSICAL WORLD
AI AGENT'S INTERNAL WORLD MODEL (The "Brain")
ROBOT SENSORS (Camera, Lidar, Tactile)
Raw Sensory Data (e.g., Pixels, Point Clouds)
PERCEPTION MODEL (Encoder)
Compresses raw inputs into compact, abstract spatial-temporal representations.
CURRENT LATENT STATE (z_t)
The agent's current "mental map" of the world.
INTERNAL SIMULATION LOOP ("Dreaming" & Planning)
DYNAMICS MODEL (Predictive Simulator)
Learns environment physics. Predicts next state given current state & hypothetical action.
PREDICTED NEXT LATENT STATE (z_{t+1})
Multi-step Rollout: Imagining future trajectories ($z_{t+2}, z_{t+3}, \dots$)
Hypothetical Action (a_t)
CONTROLLER / POLICY (Decision Maker)
Evaluates simulated futures against goals/rewards to select the best plan.
Selected Optimal Action (a_t)
ROBOT ACTUATORS (Motors, Grippers)
Physical Action Execution
CORE IDEA: The agent learns an internal model of the world's dynamics to simulate and plan actions mentally before taking risks in the real world, enabling faster learning and smarter decisions.

What's next?

Figure 1: Hype Cycle for Artificial Intelligence 2025



Source: Gartner (August 2025)

Outline

Suggested reading

- Chap 2, Neural Networks and Deep Learning.
- Chap 3–4, Dive into Deep Learning.
- Chap 1, Deep Learning with Python.

References i

[Aggarwal, 2018] Aggarwal, C. C. (2018). **Neural Networks and Deep Learning.** Springer International Publishing.

[Hughes and Correll, 2016] Hughes, D. and Correll, N. (2016). **Distributed machine learning in materials that couple sensing, actuation, computation and communication.** *arXiv:1606.03508*.