Statistical Machine Learning

Yuan YAO

HKUST

Course Infomation

- Course web:
  - https://yao-lab.github.io/course/statml/2025b
- Time and Venure:
  - Lecture: **Tu, 6:30-9:20pm**
    - ***Rm 4621, Lift 31-32*** *(126)*
- Instructor:
  - Yuan Yao <yuany@ust.hk> (https://yao-lab.github.io/)
  - Occasionally some *invited speakers*
- Teaching Assistant:
  - N.A.

Course Content

- Supervised Learning:
  - working knowledge about linear regression, classification, logistic regression, decision trees (CART), boosting, random forests, support vector machines, neural networks, etc.
- Unsupervised and Self-supervised Learning:
  - PCA, Generative Models, Generative Adversarial Networks
  - Self-supervision, e.g. masked language models etc.
- Reinforcement Learning:
  - Markov Decision Process and online learning, etc.

- **No exams. Project-based evaluation.**

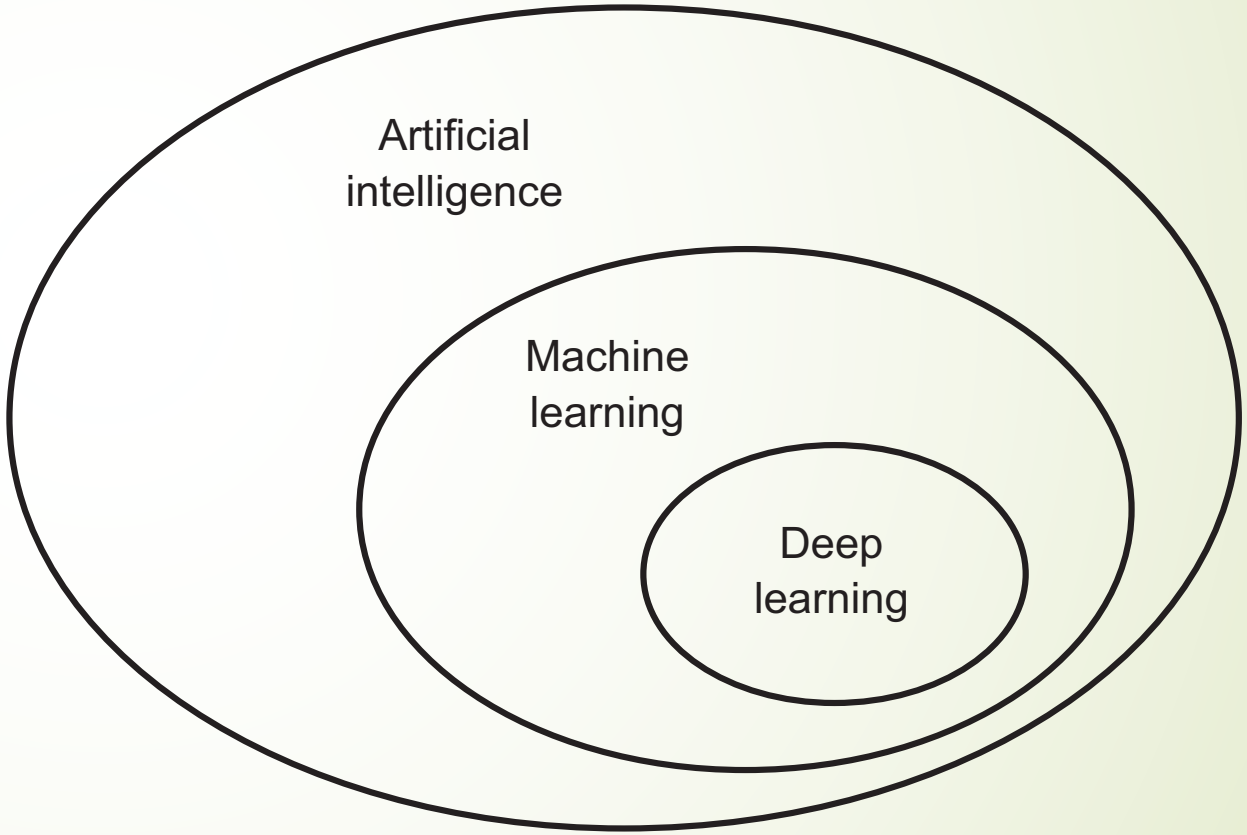
A Brief History of AI, Machine Learning, and Deep Learning

If you asked ChatGPT (3.5):

- **Title: A Historical Overview of Artificial Intelligence (AI)**
- Introduction:
  Artificial Intelligence (AI) has a rich and fascinating history that spans several decades. This article provides a comprehensive overview of the historical milestones and key developments in the field of AI. By tracing the evolution of AI from its inception to the present day, we gain valuable insights into the progression of this groundbreaking discipline.
- The Birth of AI:
  The origins of AI can be traced back to the Dartmouth Conference in 1956, where the term "artificial intelligence" was first coined. This event marked the beginning of a new era, with researchers envisioning the possibility of creating machines that could exhibit intelligent behavior.
- Early AI Approaches:
  In the 1950s and 1960s, researchers explored various approaches to AI, such as symbolic reasoning and logical inference. The development of the Logic Theorist program by Allen Newell and Herbert Simon and the General Problem Solver by Newell and J.C. Shaw were significant milestones during this period.
- The Rise of Machine Learning:
  In the 1980s and 1990s, there was a shift in focus towards machine learning approaches within AI. The emergence of expert systems and the development of algorithms like the backpropagation algorithm for training neural networks led to significant advancements in pattern recognition and decision-making.
- Knowledge-Based Systems:
  During the 1980s, knowledge-based systems gained prominence in AI research. These systems utilized expert knowledge and rule-based reasoning to solve complex problems. The development of the MYCIN system for medical diagnosis and the DENDRAL system for chemical analysis were notable achievements in this era.
- AI Winter and Resurgence:
  In the late 1980s and early 1990s, AI faced a period of reduced funding and disillusionment, known as the "AI winter." However, advancements in computing power and the emergence of new techniques, such as Bayesian networks and support vector machines, sparked a resurgence of interest in AI research in the late 1990s.
- Big Data and Deep Learning:
  In recent years, the availability of vast amounts of data and the development of powerful computational resources have revolutionized AI. Deep learning, a subfield of machine learning focused on neural networks with multiple layers, has achieved remarkable success in areas such as image and speech recognition, natural language processing, and autonomous vehicles.
- AI in the Present and Future:
  AI is now an integral part of our daily lives, with applications ranging from virtual assistants and recommendation systems to autonomous robots and self-driving cars. The field continues to evolve rapidly, with ongoing research in areas like explainable AI, reinforcement learning, and ethical considerations surrounding AI deployment.
- Conclusion:
  The historical journey of AI showcases the remarkable progress made in this field. From its early beginnings to the present day, AI has transformed the way we interact with technology and holds immense potential for future advancements. By understanding its history, we gain a deeper appreciation for the challenges overcome and the possibilities that lie ahead in the exciting world of artificial intelligence.

Artificial Intelligence, Machine Learning, and Deep Learning

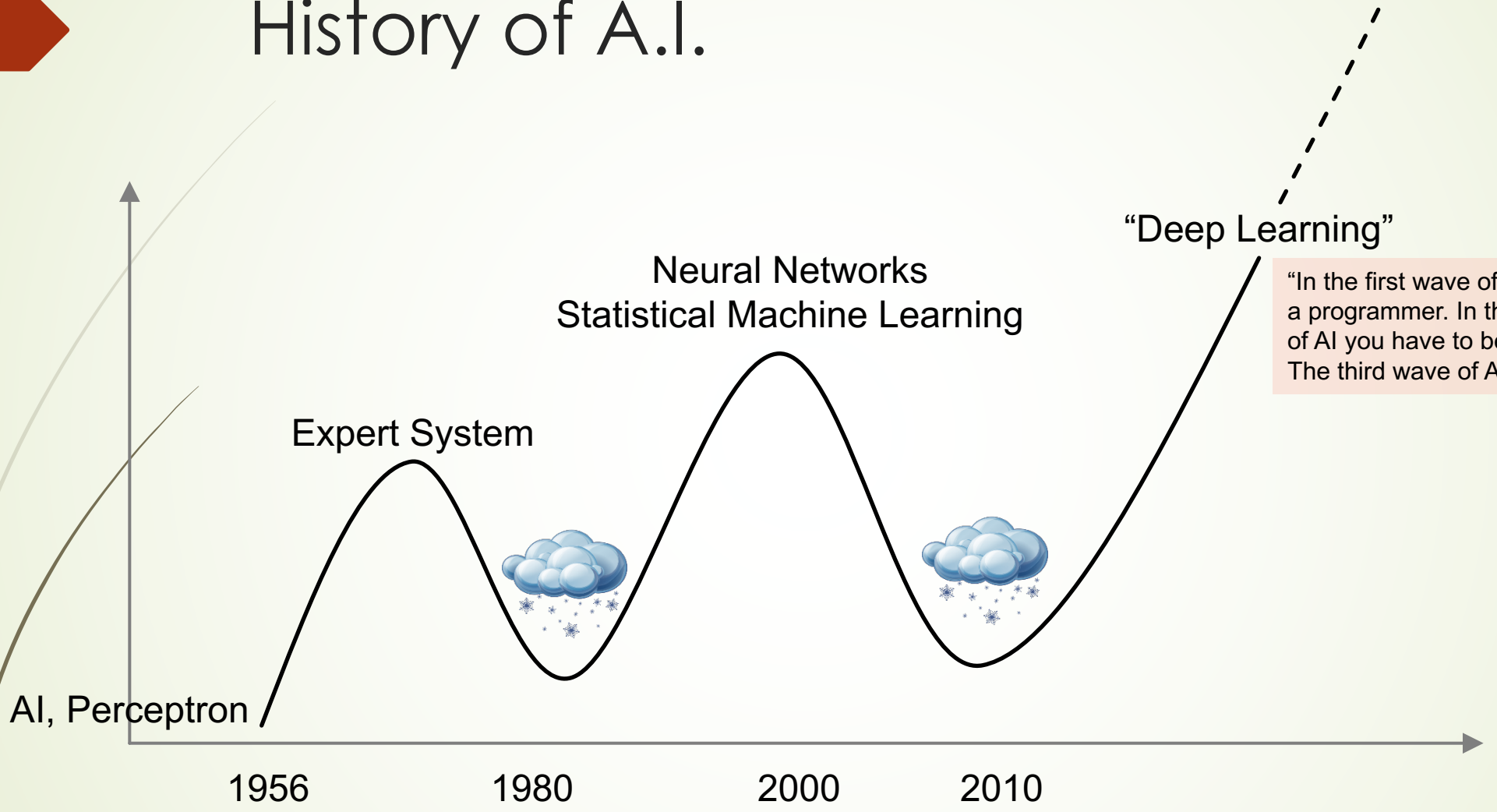
- AI is born in 1950s, when a handful of pioneers from the nascent field of computer science started asking **whether computers could be made to "think"**—a question whose ramifications we're still exploring today.



A brief history of AI

- **1943:** McCulloch & Pits proposed a boolean circuit model of neurons
- **1949:** Donald Hebb proposed **Hebbian learning rule**.
- **1950:** Alan Turing published **"Computing Machinery and Intelligence"** with **Turing test**.
- **1956:** John McCarthy at the Dartmouth Conference coined terminology "**Artificial Intelligence**"
- **1957**: Rosenblatt invented **Perceptron**
- **1960s:** golden years till **1969 Minsky-Papert's** critical book **Perceptron**
- **1970s**: the first AI winter
- **1980s**: boom of AI with **Expert System**
- **1990s:** the second AI winter, rise of **statistical machine learning**
- **1997: IBM Deep Blue** beats world chess champion Kasparov
- **2012:** return of neural networks as **deep learning** (speech, ImageNet in computer vision, NLP, …)
- **2016-2017: Google AlphaGo "Lee" and Zero**
- **2020: Google AlphaFold**
- **2022: OpenAI ChatGPT 3.5**
- **2024: Nobel Prizes of Physics and Chemistry to "AI"**
- …

History of A.I.

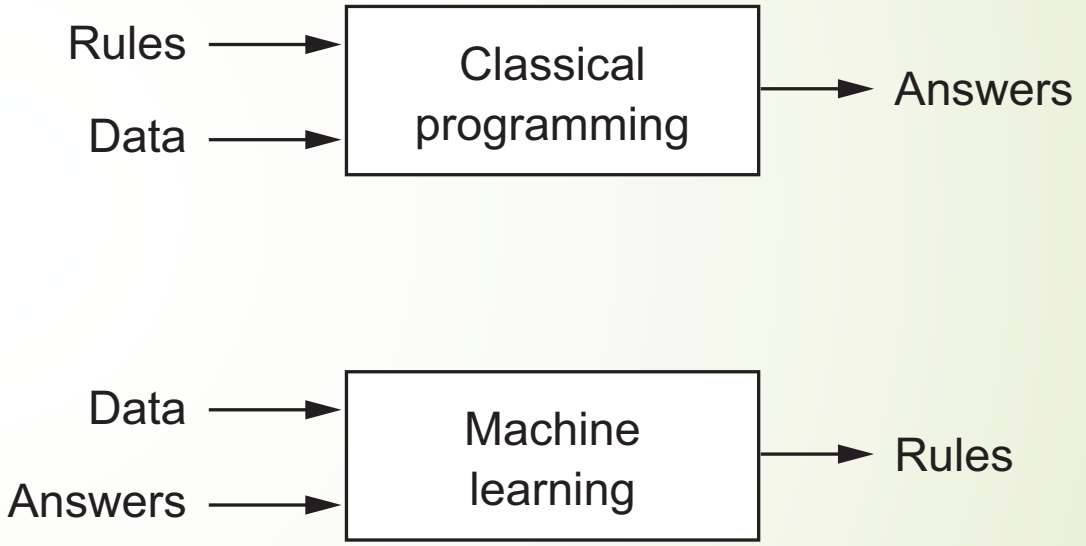


"In the first wave of AI you had to be a programmer. In the second wave of AI you have to be a data scientist. The third wave of AI..."

Source: Dr. Lee Kai-Fu • Sinovation Ventures and 2018 Conference on Neural Information Processing Systems

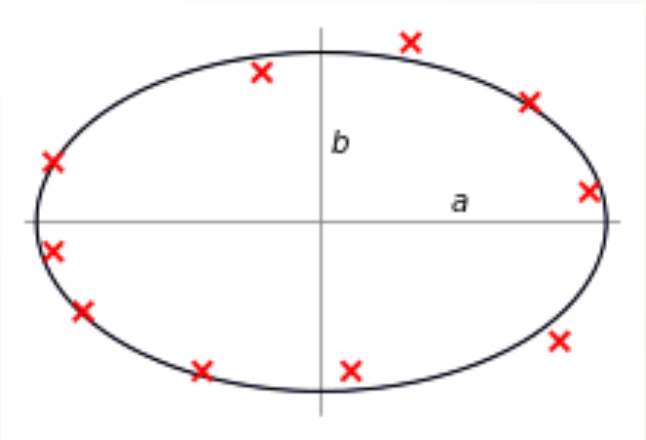
Statistical Machine Learning is a new paradigm of computer programming

- During 1950s-1980s, two competitive ideas of realizing AI exist
  - Rule based inference, or called **Expert System**
  - Statistics based inference, or called **Machine Learning**
- 1990s- Machine Learning becomes dominant



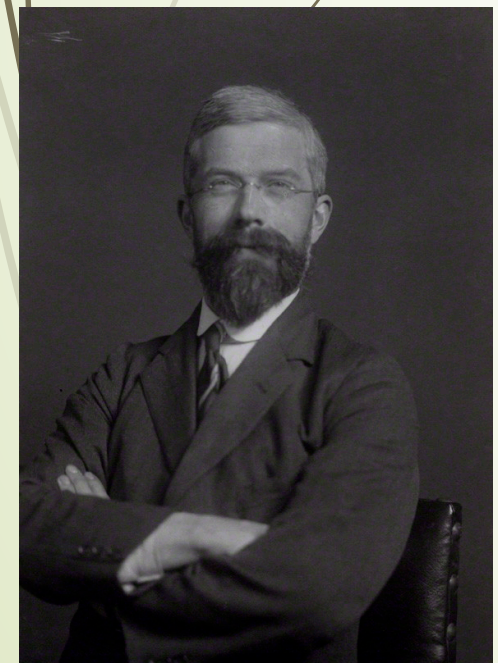
The 1st machine learning method: Least Squares

- Invention:
  - **Carl Friederich Gauss** (~1795/1809/1810),
  - Adrien-Marie Legendre (1805)
  - Robert Adrain (1808)
- Application:
  - Prediction of the location of asteroid Ceres after it emerged from behind the sun (Franz Xaver von Zach 1801)
  - Orbits of planets, Newton Laws
  - Statistics,
  - …



Fisher's Maximum Likelihood Principle (1912-1922)

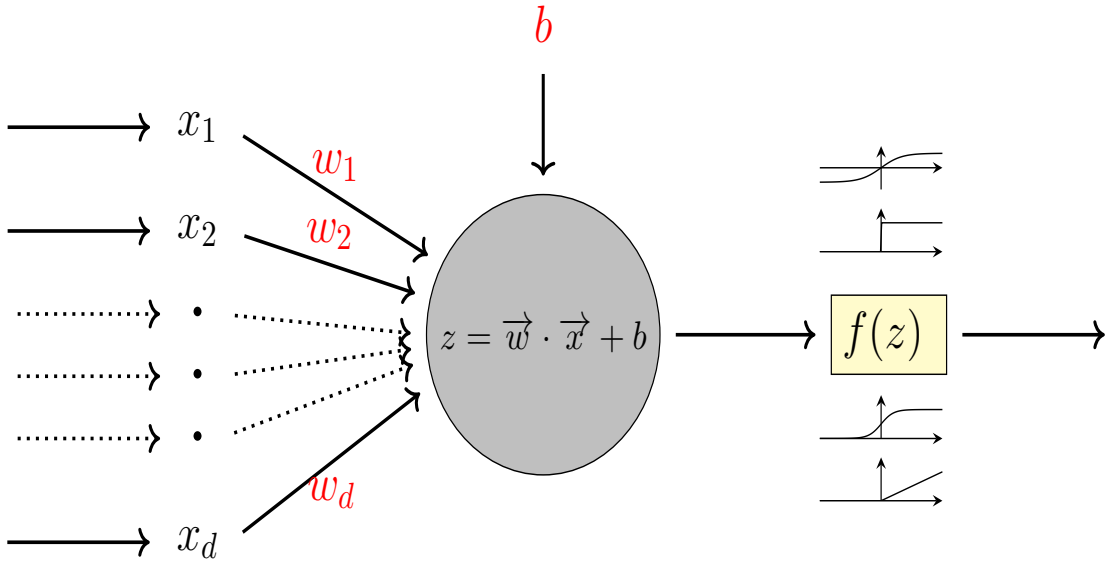
- **The least square method is the maximum likelihood estimate** (most probable values of the unknown parameters) when the noise is Gaussian.



- Fisher, R. A. (1912) **On an absolute criterion for fitting frequency curves**. *Messenger of Mathematics* 41:155-160.
- Fisher, R. A. (1922). **On the mathematical foundations of theoretical statistics**. *Philos. Trans. Roy. Soc. London Ser. A* 222:309-368.
- Aldrich, John (1997). **R. A. Fisher and the Making of Maximum Likelihood 1912 -- 1922**. *Statistical Science*, 12(3):162-176.

The 1st neural network: Perceptron

- Invented by Frank Rosenblatt (1957)



Psychological Review
Vol. 65, No. 6, 1958

THE PERCEPTRON: A PROBABILISTIC MODEL FOR INFORMATION STORAGE AND ORGANIZATION IN THE BRAIN [1]

F. ROSENBLATT

Cornell Aeronautical Laboratory

If we are eventually to understand the capability of higher organisms for perceptual recognition, generalization, recall, and thinking, we must first have answers to three fundamental questions:

1. How is information about the physical world sensed, or detected, by the biological system?
2. In what form is information stored, or remembered?
3. How does information contained in storage, or in memory, influence recognition and behavior?

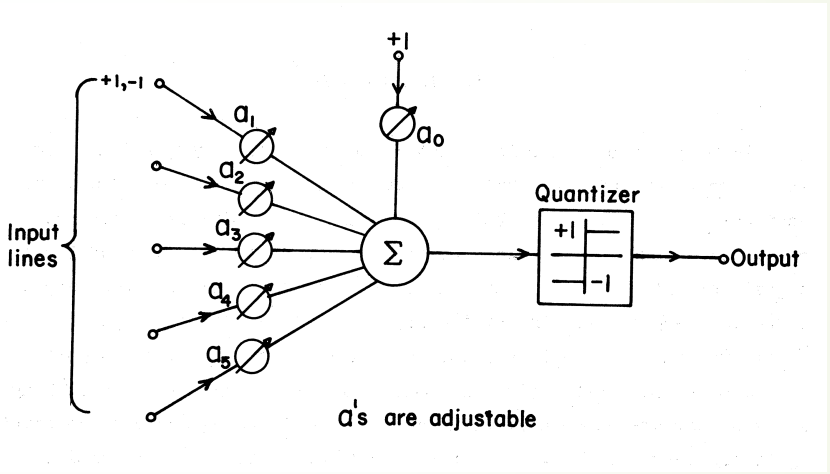
The first of these questions is in the province of sensory physiology, and is the only one for which appreciable understanding has been achieved. This article will be concerned primarily with the second and third questions, which are still subject to a vast amount of speculation, and where the few relevant facts currently supplied by neurophysiology have not yet been integrated into an acceptable theory.

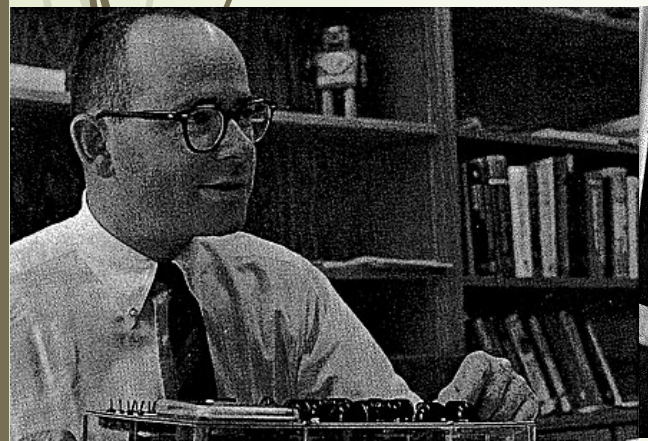
With regard to the second question, two alternative positions have been maintained. The first suggests that storage of sensory information is in the form of coded representations or images, with some sort of one-to-one mapping between the sensory stimulus and the stored pattern. According to this hypothesis, if one understood the code or "wiring diagram" of the nervous system, one should, in principle, be able to discover exactly what an organism remembers by reconstructing the original sensory patterns from the "memory traces" which they have left, much as we might develop a photographic negative, or translate the pattern of electrical charges in the "memory" of a digital computer. This hypothesis is appealing in its simplicity and ready intelligibility, and a large family of theoretical brain models has been developed around the idea of a coded, representational memory (**2**, **3**, **9**, **14**). The alternative approach, which stems from the tradition of British empiricism, hazards the guess that the images of stimuli may never really be recorded at all, and that the central nervous system simply acts as an intricate switching network, where retention takes the form of new connections, or pathways, between centers of activity. In many of the more recent developments of this position (Hebb's "cell assembly," and Hull's "cortical anticipatory goal response," for example) the "responses" which are associated to stimuli may be entirely contained within the CNS itself. In this case the response represents an "idea" rather than an action. The important feature of this approach is that there is never any simple mapping of the stimulus into memory, according to some code which would permit its later reconstruction. Whatever in-

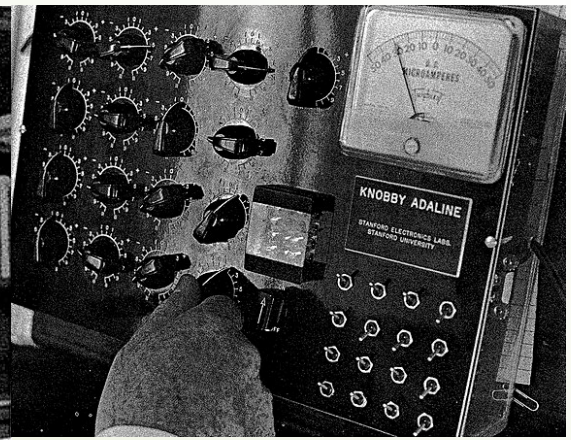
[1] The development of this theory has been carried out at the Cornell Aeronautical Laboratory, Inc., under the sponsorship of the Office of Naval Research, Contract Nonr-2381(00). This article is primarily an adaptation of material reported in Ref. 15, which constitutes the first full report on the program.

Adaline 1960

- **B Widrow and M. E. Hoff**, *Adaptive switching circuits*, TR No. 1553-1, Stanford Electronics Laboratories, Stanford University, Stanford, CA, 30 June 1960







AN ADAPTIVE "ADALINE" NEURON USING CHEMICAL "MEMISTORS"

by

B. Widrow

Technical Report No. 1553-2
October 17, 1960

Prepared under Office of Naval Research Contract
Nonr 225(24), NR 373 360
Jointly supported by the U.S. Army Signal Corps, the U.S. Air Force, and the U.S. Navy
(Office of Naval Research)

Solid-State Electronics Laboratory
Stanford Electronics Laboratories
Stanford University
Stanford, California

The Perceptron Algorithm for classification

$$\ell(w) = -\sum_{i \in \mathcal{M}_w} y_i \langle w, \mathbf{x}_i \rangle, \quad \mathcal{M}_w = \{i : y_i \langle \mathbf{x}_i, w \rangle < 0, y_i \in \{-1, 1\}\}.$$

The Perceptron Algorithm is a *Stochastic Gradient Descent* method (**Robbins-Monro 1951,** *Ann. Math. Statist*. 22(3): 400-407), with hinge loss while the Widrow-Hoff Adaline algorithm is with the least square regression loss

$$\begin{aligned} w_{t+1} &= w_t - \eta_t \nabla_i \ell(w) \\ &= \begin{cases} w_t - \eta_t y_i \mathbf{x}_i, & \text{if } y_i w_t^T \mathbf{x}_i < 0, \\ w_t, & \text{otherwise.} \end{cases} \end{aligned}$$

Stochastic Algorithms

Robbins-Monro 1951,

Ann. Math. Statist. 22(3): 400-407

A STOCHASTIC APPROXIMATION METHOD[1]

By Herbert Robbins and Sutton Monro

University of North Carolina

1. Summary. Let $M(x)$ denote the expected value at level x of the response to a certain experiment. $M(x)$ is assumed to be a monotone function of x but is unknown to the experimenter, and it is desired to find the solution $x = \theta$ of the equation $M(x) = \alpha$, where α is a given constant. We give a method for making successive experiments at levels $x_1, x_2, \cdots$ in such a way that x_n will tend to θ in probability.

2. Introduction. Let $M(x)$ be a given function and α a given constant such that the equation

$$M(x) = \alpha \tag{1}$$

has a unique root $x = \theta$. There are many methods for determining the value of θ by successive approximation. With any such method we begin by choosing one or more values $x_1, \cdots, x_r$ more or less arbitrarily, and then successively obtain new values x_n as certain functions of the previously obtained $x_1, \cdots, x_{n-1}$, the values $M(x_1), \cdots, M(x_{n-1})$, and possibly those of the derivatives $M'(x_1), \cdots, M'(x_{n-1})$, etc. If

$$\lim_{n\to\infty} x_n = \theta, \tag{2}$$

irrespective of the arbitrary initial values $x_1, \cdots, x_r$, then the method is effective for the particular function $M(x)$ and value α. The speed of the convergence in (2) and the ease with which the x_n can be computed determine the practical utility of the method.

We consider a stochastic generalization of the above problem in which the nature of the function $M(x)$ is unknown to the experimenter. Instead, we suppose that to each value x corresponds a random variable $Y = Y(x)$ with distribution function $Pr[Y(x) \leq y] = H(y \mid x)$, such that

$$M(x) = \int_{-\infty}^{\infty} y\, dH(y \mid x) \tag{3}$$

is the expected value of Y for the given x. Neither the exact nature of $H(y \mid x)$ nor that of $M(x)$ is known to the experimenter, but it is assumed that equation (1) has a unique root θ, and it is desired to estimate θ by making successive observations on Y at levels $x_1, x_2, \cdots$ determined sequentially in accordance with some definite experimental procedure. If (2) holds *in probability* irrespective of any arbitrary initial values $x_1, \cdots, x_r$, we shall, in conformity with usual statistical terminology, call the procedure *consistent* for the given $H(y \mid x)$ and value α.

[1] This work was supported in part by the Office of Naval Research.

Kiefer-Wolfowitz 1952,

Ann. Math. Statist. 23(3): 462-466

STOCHASTIC ESTIMATION OF THE MAXIMUM OF A REGRESSION FUNCTION[1]

By J. Kiefer and J. Wolfowitz

Cornell University

1. Summary. Let $M(x)$ be a regression function which has a maximum at the unknown point θ. $M(x)$ is itself unknown to the statistician who, however, can take observations at any level x. This paper gives a scheme whereby, starting from an arbitrary point x_1, one obtains successively $x_2, x_3, \cdots$ such that x_n converges to θ in probability as $n \to \infty$.

2. Introduction. Let $H(y \mid x)$ be a family of distribution functions which depend on a parameter x, and let

$$M(x) = \int_{-\infty}^{\infty} y\, dH(y \mid x). \tag{2.1}$$

We suppose that

$$\int_{-\infty}^{\infty} (y - M(x))^2\, dH(y \mid x) \leqq S < \infty, \tag{2.2}$$

and that $M(x)$ is strictly increasing for $x < \theta$, and $M(x)$ is strictly decreasing for $x > \theta$. Let $\{a_n\}$ and $\{c_n\}$ be infinite sequences of positive numbers such that

$$c_n \to 0, \tag{2.3}$$

$$\sum a_n = \infty, \tag{2.4}$$

$$\sum a_n c_n < \infty, \tag{2.5}$$

$$\sum a_n^2 c_n^{-2} < \infty. \tag{2.6}$$

(For example, $a_n = n^{-1}$, $c_n = n^{-1/3}$.)

We can now describe a recursive scheme as follows. Let z_1 be an arbitrary number. For all positive integral n we have

$$z_{n+1} = z_n + a_n \frac{(y_{2n} - y_{2n-1})}{c_n}, \tag{2.7}$$

where y_{2n-1} and y_{2n} are independent chance variables with respective distributions $H(y \mid z_n - c_n)$ and $H(y \mid z_n + c_n)$. Under regularity conditions on $M(x)$ which we shall state below we will prove that z_n converges stochastically to θ (as $n \to \infty$).

The statistical importance of this problem is obvious and need not be discussed. The stimulus for this paper came from the interesting paper by Robbins and Monro [1] (see also Wolfowitz [2]).

[1] Research under contract with the Office of Naval Research. Presented to the American Mathematical Society at New York on April 25, 1952.

Separable Data with Margin: Stopping of Perceptron after Finite Steps

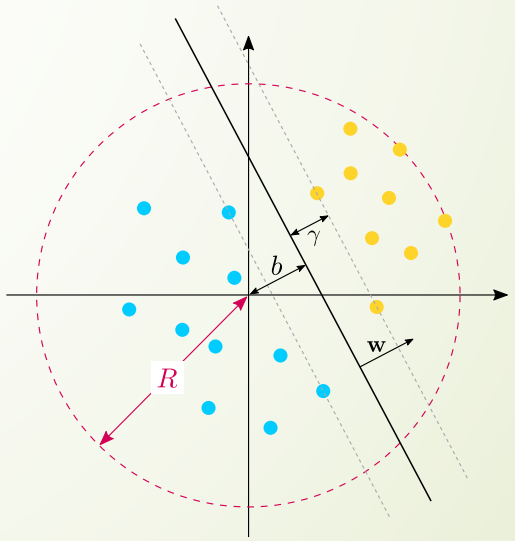
The perceptron convergence theorem was proved by Block (1962) and Novikoff (1962). The following version is based on that in Cristianini and Shawe-Taylor (2000).

Theorem 1 (Block, Novikoff). *Let the training set* $S = \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_n, t_n)\}$ *be contained in a sphere of radius* R *about the origin. Assume the dataset to be linearly separable, and let* $\mathbf{w}_{\text{opt}}$, $\|\mathbf{w}_{\text{opt}}\| = 1$, *define the hyperplane separating the samples, having functional margin* $\gamma > 0$. *We initialise the normal vector as* $\mathbf{w}_0 = \mathbf{0}$. *The number of updates,* k, *of the perceptron algorithms is then bounded by*

$$k \leq \left(\frac{2R}{\gamma}\right)^2. \tag{10}$$

Input ball: $R = \max_i \|\mathbf{x}_i\|.$

Margin: $\gamma := \min_i y_i f(x_i)$



Proof. (growth of $|w_k|$)

Proof. Though the proof can be done using the augmented normal vector and samples defined in the beginning, the notation will be a lot easier if we introduce a different augmentation: $\hat{\mathbf{w}} = (\mathbf{w}^\mathsf{T}, b/R)^\mathsf{T} = (w_1, \ldots, w_D, b/R)^\mathsf{T}$ and $\hat{\mathbf{x}} = (\mathbf{x}^\mathsf{T}, R)^\mathsf{T} = (x_1, \ldots, x_D, R)^\mathsf{T}$.

We first derive an upper bound on how fast the normal vector grows. As the hyperplane is unchanged if we multiply $\hat{\mathbf{w}}$ by a constant, we can set $\eta = 1$ without loss of generality. Let $\hat{\mathbf{w}}_{k+1}$ be the updated (augmented) normal vector after the kth error has been observed.

$$\|\hat{\mathbf{w}}_{k+1}\|^2 = (\hat{\mathbf{w}}_k + t_i\hat{\mathbf{x}}_i)^\mathsf{T}(\hat{\mathbf{w}}_k + t_i\hat{\mathbf{x}}_i) \tag{11}$$
$$= \hat{\mathbf{w}}_k^\mathsf{T}\hat{\mathbf{w}}_k + \hat{\mathbf{x}}_i^\mathsf{T}\hat{\mathbf{x}}_i + 2t_i\hat{\mathbf{w}}_k^\mathsf{T}\hat{\mathbf{x}}_i \tag{12}$$
$$= \|\hat{\mathbf{w}}_k\|^2 + \|\hat{\mathbf{x}}_i\|^2 + 2t_i\hat{\mathbf{w}}_k^\mathsf{T}\hat{\mathbf{x}}_i\,. \tag{13}$$

Since an update was triggered, we know that $t_i\hat{\mathbf{w}}_k^\mathsf{T}\hat{\mathbf{x}}_i \leq 0$, thus

$$\|\hat{\mathbf{w}}_k\|^2 + \|\hat{\mathbf{x}}_i\|^2 + 2t_i\hat{\mathbf{w}}_k^\mathsf{T}\hat{\mathbf{x}}_i \leq \|\hat{\mathbf{w}}_k\|^2 + \|\hat{\mathbf{x}}_i\|^2 \tag{14}$$
$$= \|\hat{\mathbf{w}}_k\|^2 + (\|\mathbf{x}_i\|^2 + R^2) \tag{15}$$
$$\leq \|\hat{\mathbf{w}}_k\|^2 + 2R^2\,. \tag{16}$$

This implies that $\|\hat{\mathbf{w}}_k\|^2 \leq 2kR^2$, thus

$$\|\hat{\mathbf{w}}_{k+1}\|^2 \leq 2(k+1)R^2\,. \tag{17}$$

Proof (continued, projection on w_{opt})

We then proceed to show how the inner product between an update of the normal vector and $\hat{\mathbf{w}}_{\text{opt}}$ increase with each update:

$$\hat{\mathbf{w}}_{\text{opt}}^{\mathsf{T}}\hat{\mathbf{w}}_{k+1} = \hat{\mathbf{w}}_{\text{opt}}^{\mathsf{T}}\hat{\mathbf{w}}_k + t_i\hat{\mathbf{w}}_{\text{opt}}^{\mathsf{T}}\hat{\mathbf{x}}_i \tag{18}$$

$$\geq \hat{\mathbf{w}}_{\text{opt}}^{\mathsf{T}}\hat{\mathbf{w}}_k + \gamma \tag{19}$$

$$\geq (k+1)\gamma\,, \tag{20}$$

since $\hat{\mathbf{w}}_{\text{opt}}^{\mathsf{T}}\hat{\mathbf{w}}_k \geq k\gamma$. We therefore have

$$k^2\gamma^2 \leq (\hat{\mathbf{w}}_{\text{opt}}^{\mathsf{T}}\hat{\mathbf{w}}_k)^2 \leq \|\hat{\mathbf{w}}_{\text{opt}}\|^2\|\hat{\mathbf{w}}_k\|^2 \leq 2kR^2\|\hat{\mathbf{w}}_{\text{opt}}\|^2\,, \tag{21}$$

where we have made use of the Cauchy-Schwarz inequality. As $k^2\gamma^2$ grows faster than $2kR^2$, Eq. (21) can hold if and only if

$$k \leq 2\|\hat{\mathbf{w}}_{\text{opt}}\|^2\frac{R^2}{\gamma^2}\,. \tag{22}$$

Proof (continued, combined bounds)

As $b \leq R$, we can rewrite the norm of the normal vector:

$$\|\hat{\mathbf{w}}_{\text{opt}}\|^2 = \|\mathbf{w}_{\text{opt}}\|^2 + \frac{b^2}{R^2} \leq \|\mathbf{w}_{\text{opt}}\|^2 + 1 = 2. \tag{23}$$

The bound on k now becomes

$$k \leq 4\frac{R^2}{\gamma^2} = \left(\frac{2R}{\gamma}\right)^2, \tag{24}$$

which therefore bounds the number of updates necessary to find the separating hyperplane. □

Hilbert's 13th Problem

Algebraic equations (under a suitable transformation) of degree up to 6 can be solved by functions of two variables. What about

$$x^7 + ax^3 + bx^2 + cx + 1 = 0?$$

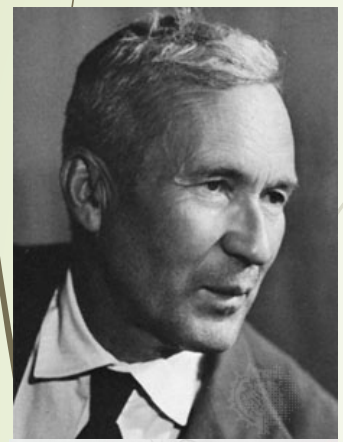
Hilbert's conjecture: $x(a, b, c)$ cannot be expressed by a superposition (sums and compositions) of bivariate functions.

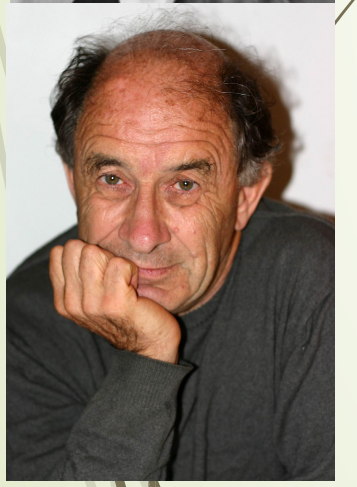
Question: can every continuous (analytic, C^∞, etc) function of n variables be represented as a superposition of continuous (analytic, C^∞, etc) functions of $n-1$ variables?

Theorem (D. Hilbert)

There is an analytic function of three variables that cannot be expressed as a superposition of bivariate ones.

Kolmogorov's Superposition Theorem





Theorem (A. Kolmogorov, 1956; V. Arnold, 1957)

Given $n \in \mathbb{Z}^+$, every $f_0 \in C([0,1]^n)$ can be reprensented as

$$f_0(x_1, x_2, \cdots, x_n) = \sum_{q=1}^{2n+1} g_q \left(\sum_{p=1}^{n} \phi_{pq}(x_p) \right),$$

where $\phi_{pq} \in C[0,1]$ are increasing functions independent of f_0 and $g_q \in C[0,1]$ depend on f_0.

- Can choose g_q to be all the same $g_q \equiv g$ (Lorentz, 1966).
- Can choose ϕ_{pq} to be Hölder or Lipschitz continuous, but not C^1 (Fridman, 1967).
- Can choose $\phi_{pq} = \lambda_p \phi_q$ where $\lambda_1, \cdots, \lambda_n > 0$ and $\sum_p \lambda_p = 1$ (Sprecher, 1972).

If f is a multivariate continuous function, then f can be written as a superposition of composite functions of mixtures of continuous functions of single variables:
finite **composition** of continuous functions of a **single variable** and the **addition**.

Kolmogorov's Exact Representation is not stable or smooth

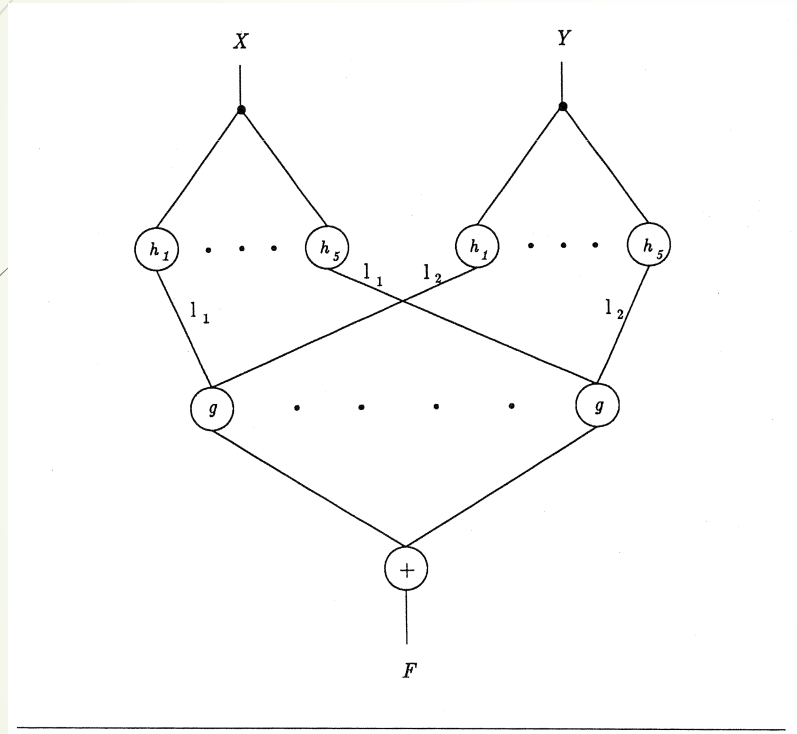


Figure 1: The network representation of an improved version of Kolmogorov's theorem, due to Kahane (1975). The figure shows the case of a bivariate function. The Kahane's representation formula is $f(x_1, \ldots, x_n) = \sum_{q=1}^{2n+1} g[\sum_{p=1}^{n} l_p h_q(x_p)]$ where h_q are strictly monotonic functions and l_p are strictly positive constants smaller than 1.

- [Girosi-Poggio'1989] Representation Properties of Networks: Kolmogorov's Theorem Is Irrelevant, https://www.mitpressjournals.org/doi/pdf/10.1162/neco.1989.1.4.465
- Lacking smoothness in *h* and *g* [Vitushkin'1964] fails to guarantee the **generalization ability (stability)** against noise and perturbations
- The representation is **not universal** in the sense that *g* and *h* both depend on the function F to be represented.

A Simplified illustration by David McAllester

A Simpler, Similar Theorem

For (possibly discontinuous) $f : [0,1]^N \to \mathbb{R}$ there exists (possibly discontinuous) $g, h_i : \mathbb{R} \to \mathbb{R}$.

$$f(x_1, \ \ldots, \ x_N) = g\left(\sum_i h_i(x_i)\right)$$

Proof: Select h_i to spread out the digits of its argument so that $\sum_i h_i(x_i)$ contains all the digits of all the x_i.

Universal Approximate Representation

[Cybenko'1989, Hornik et al. 1989, Poggio-Girosi'1989, ...]

For continuous $f : [0,1]^N \to \mathbb{R}$ and $\varepsilon > 0$ there exists

$$F(x) = \alpha^\top \sigma(Wx + \beta)$$

$$= \sum_i \alpha_i \sigma \left(\sum_j W_{i,j}\, x_j + \beta_i \right)$$

such that for all x in $[0,1]^N$ we have $|F(x) - f(x)| < \varepsilon$.

Complexity (regularity, smoothness) thereafter becomes the central pursuit in Approximation Theory.

KAN: Kolmogorov-Arnold Networks

KAN: Kolmogorov–Arnold Networks

Ziming Liu[1,4*] **Yixuan Wang**[2] **Sachin Vaidya**[1] **Fabian Ruehle**[3,4]
James Halverson[3,4] **Marin Soljačić**[1,4] **Thomas Y. Hou**[2] **Max Tegmark**[1,4]
[1] Massachusetts Institute of Technology
[2] California Institute of Technology
[3] Northeastern University
[4] The NSF Institute for Artificial Intelligence and Fundamental Interactions

Abstract

Inspired by the Kolmogorov-Arnold representation theorem, we propose Kolmogorov-Arnold Networks (KANs) as promising alternatives to Multi-Layer Perceptrons (MLPs). While MLPs have *fixed* activation functions on *nodes* ("neurons"), KANs have *learnable* activation functions on *edges* ("weights"). KANs have no linear weights at all – every weight parameter is replaced by a univariate function parametrized as a spline. We show that this seemingly simple change makes KANs outperform MLPs in terms of accuracy and interpretability, on small-scale AI + Science tasks. For accuracy, smaller KANs can achieve comparable or better accuracy than larger MLPs in function fitting tasks. Theoretically and empirically, KANs possess faster neural scaling laws than MLPs. For interpretability, KANs can be intuitively visualized and can easily interact with human users. Through two examples in mathematics and physics, KANs are shown to be useful "collaborators" helping scientists (re)discover mathematical and physical laws. In summary, KANs are promising alternatives for MLPs, opening opportunities for further improving today's deep learning models which rely heavily on MLPs.

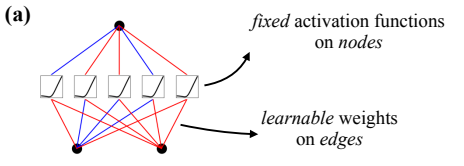
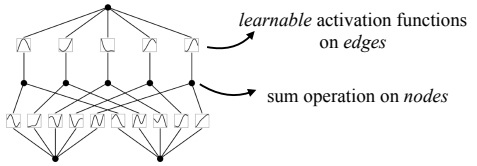
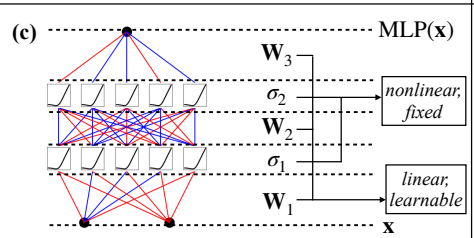
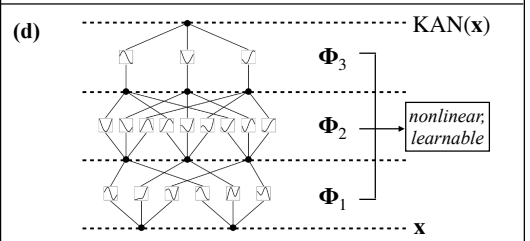
Model	Multi-Layer Perceptron (MLP)	Kolmogorov-Arnold Network (KAN)
Theorem	Universal Approximation Theorem	Kolmogorov-Arnold Representation Theorem
Formula (Shallow)	$f(\mathbf{x}) \approx \sum_{i=1}^{N(\epsilon)} a_i \sigma(\mathbf{w}_i \cdot \mathbf{x} + b_i)$	$f(\mathbf{x}) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)$
Model (Shallow)		
Formula (Deep)	$\mathrm{MLP}(\mathbf{x}) = (\mathbf{W}_3 \circ \sigma_2 \circ \mathbf{W}_2 \circ \sigma_1 \circ \mathbf{W}_1)(\mathbf{x})$	$\mathrm{KAN}(\mathbf{x}) = (\mathbf{\Phi}_3 \circ \mathbf{\Phi}_2 \circ \mathbf{\Phi}_1)(\mathbf{x})$
Model (Deep)		

Figure 0.1: Multi-Layer Perceptrons (MLPs) vs. Kolmogorov-Arnold Networks (KANs)

Ziming Liu et al. arXiv:2404.19756

KAN vs. MLP?

KAN or MLP: A Fairer Comparison

Runpeng Yu, Weihao Yu, and Xinchao Wang
National University of Singapore

https://github.com/yu-rp/KANbeFair

Abstract

This paper does not introduce a novel method. Instead, it offers a fairer and more comprehensive comparison of KAN and MLP models across various tasks, including machine learning, computer vision, audio processing, natural language processing, and symbolic formula representation. Specifically, we control the number of parameters and FLOPs to compare the performance of KAN and MLP. Our main observation is that, except for symbolic formula representation tasks, MLP generally outperforms KAN. We also conduct ablation studies on KAN and find that its advantage in symbolic formula representation mainly stems from its B-spline activation function. When B-spline is applied to MLP, performance in symbolic formula representation significantly improves, surpassing or matching that of KAN. However, in other tasks where MLP already excels over KAN, B-spline does not substantially enhance MLP's performance. Furthermore, we find that KAN's forgetting issue is more severe than that of MLP in a standard class-incremental continual learning setting, which differs from the findings reported in the KAN paper. We hope these results provide insights for future research on KAN and other MLP alternatives.

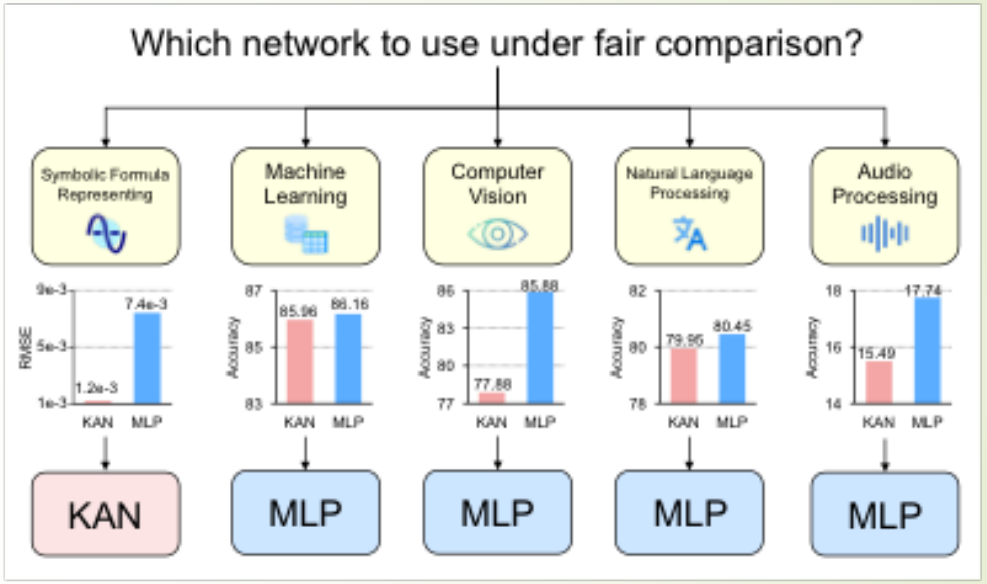


Figure 1: Performance comparison between KAN and MLP under fair setup. MLP yields higher average accuracy in machine learning, computer vision, natural language processing, and audio processing, while KAN leads to lower average root mean square error. For the Symbolic Formula Representation task, a lower RMSE is better.

Runpeng Yu et al. arXiv: 2407.16674

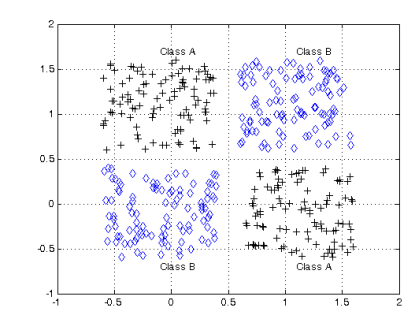
More discussions in Prof. Haizhao YANG (U Maryland) seminar:
Recent Advances of Approximation and Computation of High-Dimensional Functions

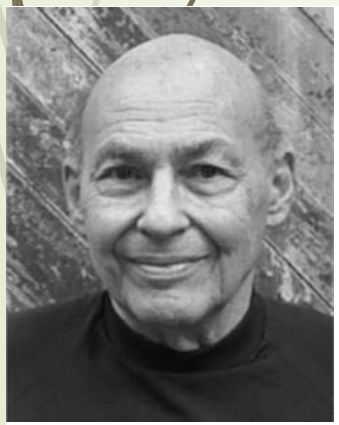
Locality or Sparsity of Computation

Minsky and Papert, 1969

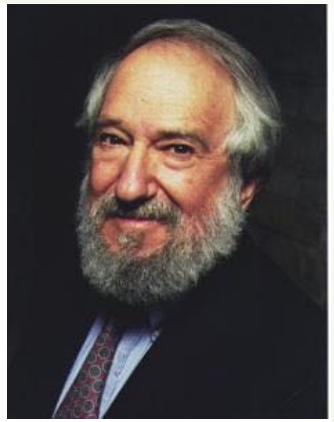
Perceptron can't do **XOR** classification

Perceptron needs infinite global information to compute **connectivity**





Marvin Minsky

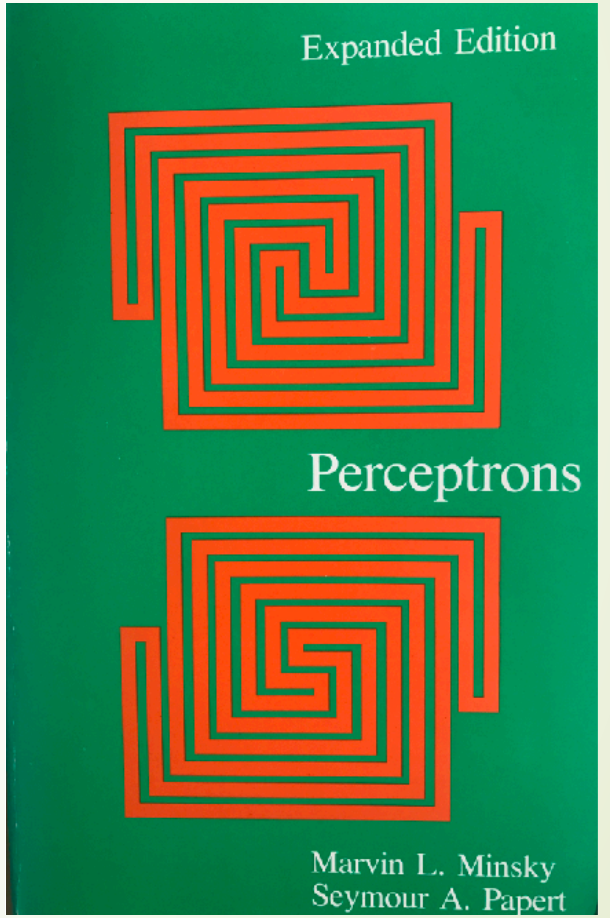


Seymour Papert

Locality or **Sparsity** is important:

Locality in time?

Locality in space?



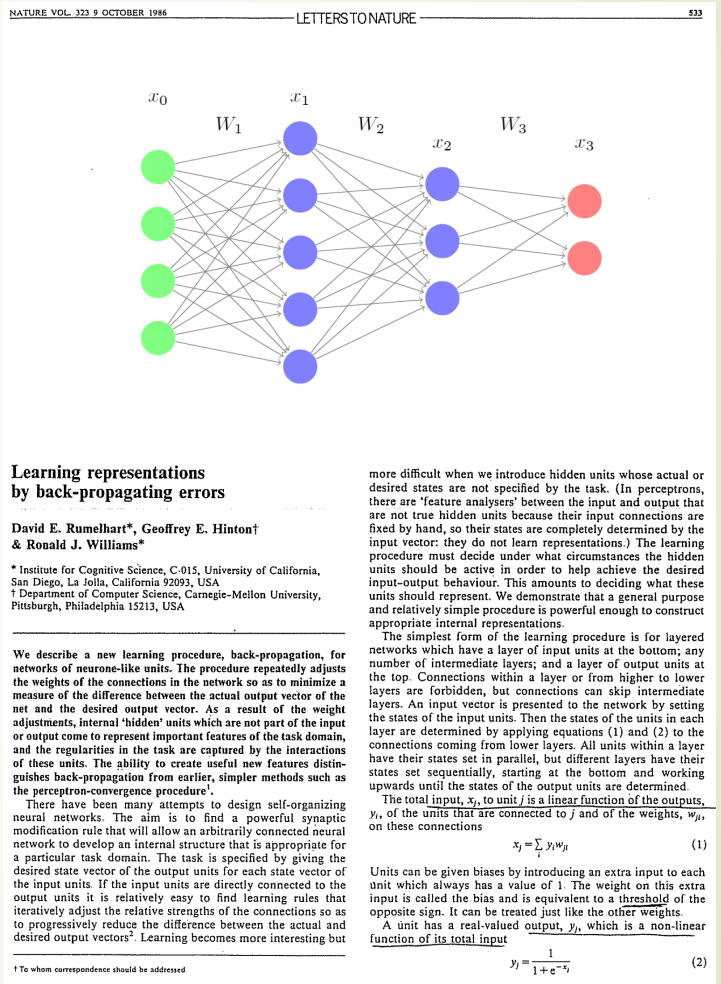
Multilayer Perceptrons (MLP) and Back-Propagation (BP) Algorithms

D.E. Rumelhart, G. Hinton, R.J. Williams (1986)
Learning representations by back-propagating errors, Nature, 323(9): 533-536

BP algorithms as **stochastic gradient descent** algorithms (**Robbins–Monro 1950; Kiefer-Wolfowitz 1951**) with Chain rules of Gradient maps

Deep network may classify **XOR**. Yet **topology**?

We address complexity and geometric invariant properties first.



NATURE VOL. 323 9 OCTOBER 1986 — LETTERS TO NATURE — 533

Learning representations by back-propagating errors

David E. Rumelhart*, Geoffrey E. Hinton† & Ronald J. Williams*

* Institute for Cognitive Science, C-015, University of California, San Diego, La Jolla, California 92093, USA
† Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Philadelphia 15213, USA

We describe a new learning procedure, back-propagation, for networks of neurone-like units. The procedure repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector. As a result of the weight adjustments, internal 'hidden' units which are not part of the input or output come to represent important features of the task domain, and the regularities in the task are captured by the interactions of these units. The ability to create useful new features distinguishes back-propagation from earlier, simpler methods such as the perceptron-convergence procedure[1].

There have been many attempts to design self-organizing neural networks. The aim is to find a powerful synaptic modification rule that will allow an arbitrarily connected neural network to develop an internal structure that is appropriate for a particular task domain. The task is specified by giving the desired state vector of the output units for each state vector of the input units. If the input units are directly connected to the output units it is relatively easy to find learning rules that iteratively adjust the relative strengths of the connections so as to progressively reduce the difference between the actual and desired output vectors[2]. Learning becomes more interesting but more difficult when we introduce hidden units whose actual or desired states are not specified by the task. (In perceptrons, there are 'feature analysers' between the input and output that are not true hidden units because their input connections are fixed by hand, so their states are completely determined by the input vector: they do not learn representations.) The learning procedure must decide under what circumstances the hidden units should be active in order to help achieve the desired input-output behaviour. This amounts to deciding what these units should represent. We demonstrate that a general purpose and relatively simple procedure is powerful enough to construct appropriate internal representations.

The simplest form of the learning procedure is for layered networks which have a layer of input units at the bottom; any number of intermediate layers; and a layer of output units at the top. Connections within a layer or from higher to lower layers are forbidden, but connections can skip intermediate layers. An input vector is presented to the network by setting the states of the input units. Then the states of the units in each layer are determined by applying equations (1) and (2) to the connections coming from lower layers. All units within a layer have their states set in parallel, but different layers have their states set sequentially, starting at the bottom and working upwards until the states of the output units are determined.

The total input, x_j, to unit j is a linear function of the outputs, y_i, of the units that are connected to j and of the weights, w_{ji}, on these connections

$$x_j = \sum_i y_i w_{ji} \quad (1)$$

Units can be given biases by introducing an extra input to each unit which always has a value of 1. The weight on this extra input is called the bias and is equivalent to a threshold of the opposite sign. It can be treated just like the other weights.

A unit has a real-valued output, y_j, which is a non-linear function of its total input

$$y_j = \frac{1}{1+e^{-x_j}} \quad (2)$$

† To whom correspondence should be addressed

Parallel Distributed Processing

by Rumelhart and McClelland, 1986

Minsky and Papert set out to show which functions can and cannot be computed by this class of machines. They demonstrated, in particular, that such perceptrons are unable to calculate such mathematical functions as parity (whether an odd or even number of points are on in the retina) or the topological function of connectedness (whether all points that are on are connected to all other points that are on either directly or via other points that are also on) without making use of absurdly large numbers of predicates. The analysis is extremely elegant and demonstrates the importance of a mathematical approach to analyz-

of multilayer networks that compute parity). Similarly, it is not difficult to develop networks capable of solving the connectedness or inside/outside problem. Hinton and Sejnowski have analyzed a version of such a network (see Chapter 7).

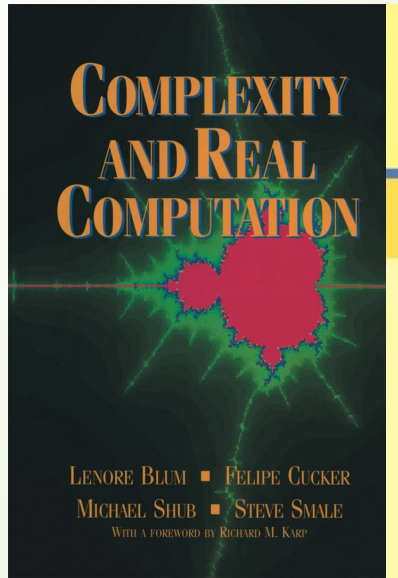
Essentially, then, although Minsky and Papert were exactly correct in their analysis of the *one-layer perceptron,* the theorems don't apply to systems which are even a little more complex. In particular, it doesn't apply to multilayer systems nor to systems that allow feedback loops.

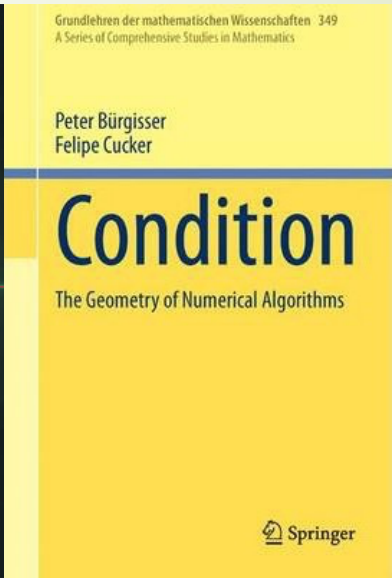
Topology can be learned with finite information if the manifold is *stable*

Blum-Shub-Smale models of Real Computation

A Model of Real Computation

- Starting from **Blum, Shub, Smale** (1989)
- It admits inputs and operations (addition, substraction, multiplication, and (in the case of fields) division) of **real (complex) numbers** with *infinite precision*
- "The key importance of the **condition number**, which measures the closeness of a problem instance to the manifold of ill-posed instances, is clearly developed." – Richard Karp





The Condition Number of a Manifold

Throughout our discussion, we associate to $\mathcal{M}$ a condition number $(1/\tau)$ where τ is defined as the largest number having the property: The open normal bundle about $\mathcal{M}$ of radius r is embedded in $\mathbb{R}^N$ for every $r < \tau$. Its image Tub_τ is a tubular neighborhood of $\mathcal{M}$ with its canonical projection map

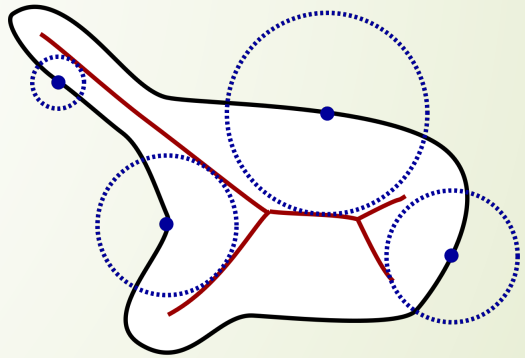
$$\pi_0 : \text{Tub}_\tau \to \mathcal{M}.$$

Smallest Local Feature Size

$G = \{x \in \mathbb{R}^N$ such that $\exists$ distinct $p, q \in \mathcal{M}$ where $d(x, \mathcal{M}) = \|x - p\| = \|x - q\|\}$,

where $d(x, \mathcal{M}) = \inf_{y \in \mathcal{M}} \|x - y\|$ is the distance of x to $\mathcal{M}$. The closure of G is called the medial axis and for any point $p \in \mathcal{M}$ the local feature size $\sigma(p)$ is the distance of p to the medial axis. Then it is easy to check that

$$\tau = \inf_{p \in \mathcal{M}} \sigma(p).$$



Find Homology with Finite Samples

[Niyogi, Smale, Weinberger (2008)]

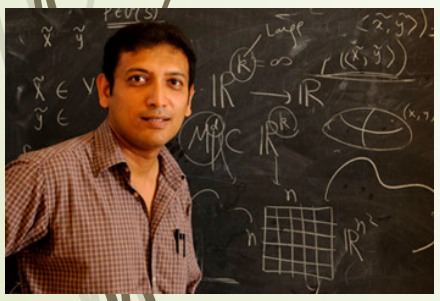
Theorem 3.1 *Let $\mathcal{M}$ be a compact submanifold of $\mathbb{R}^N$ with condition number τ. Let $\bar{x} = \{x_1, \ldots, x_n\}$ be a set of n points drawn in i.i.d. fashion according to the uniform probability measure on $\mathcal{M}$. Let $0 < \epsilon < \tau/2$. Let $U = \bigcup_{x \in \bar{x}} B_\epsilon(x)$ be a correspondingly random open subset of $\mathbb{R}^N$. Then for all*

$$n > \beta_1 \left(\log(\beta_2) + \log\left(\frac{1}{\delta}\right) \right),$$

the homology of U equals the homology of $\mathcal{M}$ with high confidence (probability $> 1 - \delta$).

$$\beta_1 = \frac{vol(\mathcal{M})}{(\cos^k(\theta_1))vol(B^k_{\epsilon/4})} \quad \text{and} \quad \beta_2 = \frac{vol(\mathcal{M})}{(\cos^k(\theta_2))vol(B^k_{\epsilon/8})}.$$

Here k is the dimension of the manifold $\mathcal{M}$ and $vol(B^k_\epsilon)$ denotes the k-dimensional volume of the standard k-dimensional ball of radius ϵ. Finally, $\theta_1 = \arcsin(\epsilon/8\tau)$ and $\theta_2 = \arcsin(\epsilon/16\tau)$.



Partha Niyogi@Chiccago, 1967-2010

BP algorithm = Gradient Descent Method

- Training examples $\{x_0^i\}_{i=1}^n$ and labels $\{y^i\}_{i=1}^n$
- Output of the network $\{x_L^i\}_{i=1}^m$
- Objective Square loss, cross-entropy loss, etc.

$$J(\{W_l\},\{b_l\}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}\|y^i - x_L^i\|_2^2 \quad (1)$$

- Gradient descent

$$W_l = W_l - \eta\frac{\partial J}{\partial W_l}$$

$$b_l = b_l - \eta\frac{\partial J}{\partial b_l}$$

In practice: use Stochastic Gradient Descent (SGD)

Derivation of BP: Lagrangian Multiplier

LeCun et al. 1988

Given n training examples $(I_i, y_i) \equiv$ (input,target) and L layers

- Constrained optimization

$$\begin{aligned} \min_{W,x} \quad & \textstyle\sum_{i=1}^{n} \|x_i(L) - y_i\|_2 \\ \text{subject to} \quad & x_i(\ell) = f_\ell\Big[W_\ell x_i\,(\ell-1)\Big], \\ & i = 1, \ldots, n, \quad \ell = 1, \ldots, L,\ x_i(0) = I_i \end{aligned}$$

- Lagrangian formulation (Unconstrained)

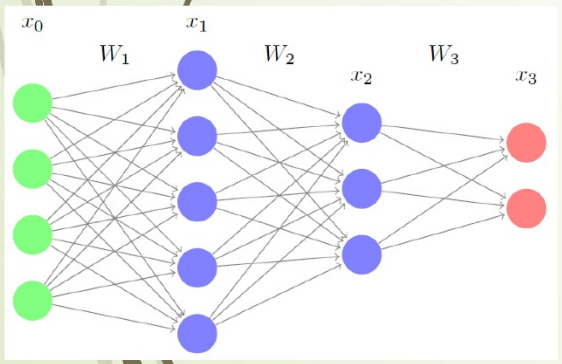
$$\min_{W,x,B} \mathcal{L}(W, x, B)$$

$$\mathcal{L}(W, x, B) = \textstyle\sum_{i=1}^{n} \left\{ \|x_i(L) - y_i\|_2^2 + \right.$$

$$\left. \textstyle\sum_{\ell=1}^{L} B_i(\ell)^T \left(x_i(\ell) - f_\ell\Big[W_\ell x_i\,(\ell-1)\Big] \right) \right\}$$

http://yann.lecun.com/exdb/publis/pdf/lecun-88.pdf

BP Algorithm: Forward Pass



- Cascade of repeated [linear operation followed by coordinatewise nonlinearity]'s
- Nonlinearities: sigmoid, hyperbolic tangent, (recently) ReLU.

Algorithm 1 Forward pass

Input: x_0

Output: x_L

1: **for** $\ell = 1$ to L **do**
2: $\quad x_\ell = f_\ell(W_\ell x_{\ell-1} + b_\ell)$
3: **end for**

back-propagation – derivation

- $\frac{\partial \mathcal{L}}{\partial B}$

Forward pass

$$x_i(\ell) = f_\ell\Big[\underbrace{W_\ell x_i\,(\ell-1)}_{A_i(\ell)}\Big] \quad \ell = 1,\ldots,L, \quad i = 1,\ldots,n$$

- $\frac{\partial \mathcal{L}}{\partial x}, z_\ell = [\nabla f_\ell] B(\ell)$

Backward (adjoint) pass

$$z(L) = 2\nabla f_L\Big[A_i(L)\Big]\,(y_i - x_i(L))$$

$$z_i(\ell) = \nabla f_\ell\Big[A_i(\ell)\Big] W_{\ell+1}^T z_i(\ell+1) \quad \ell = 0,\ldots,L-1$$

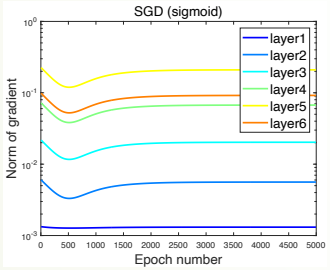
- $W \leftarrow W + \lambda \frac{\partial \mathcal{L}}{\partial W}$

Weight update

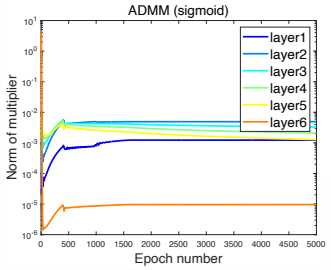
$$W_\ell \leftarrow W_\ell + \lambda \sum_{i=1}^n z_i(\ell) x_i^T(\ell-1)$$

Notes on SGD vs. ADMM/BCD

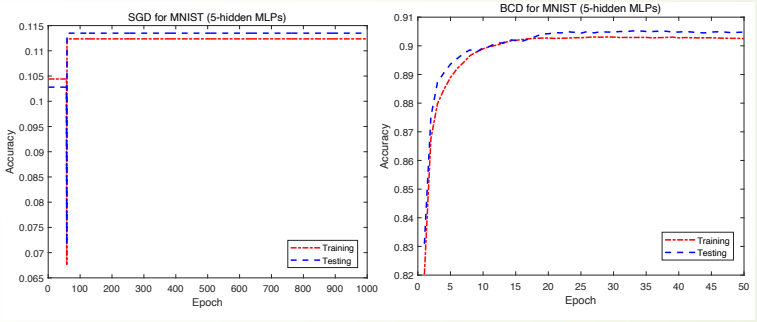
- Stochastic Gradient Descent (SGD) suffers from the well-known **gradient vanishing issue** in deep learning, which can be derived via Lagrangian multiplier method [LeCun 1988, http://yann.lecun.com/exdb/publis/pdf/lecun-88.pdf]
- Nongradient-based algorithms: **ADMM** (Augmented Lagrangian multipliers) and **BCD** (Block Coordinate Descent) may alleviate gradient vanishing



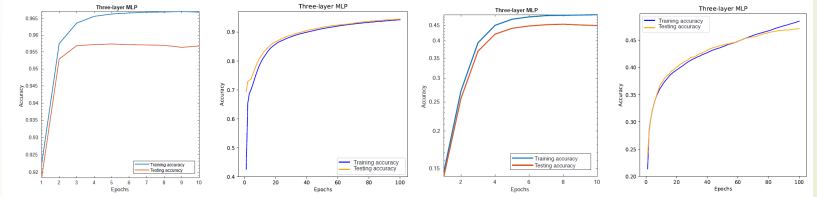
(a) Gradient vanishing of SGD



(b) Saturation-avoidance of ADMM



- High epoch efficiency of BCD at early stage



(a) MNIST (BCD) (b) MNIST (SGD) (c) CIFAR-10 (BCD) (d) CIFAR-10 (SGD)

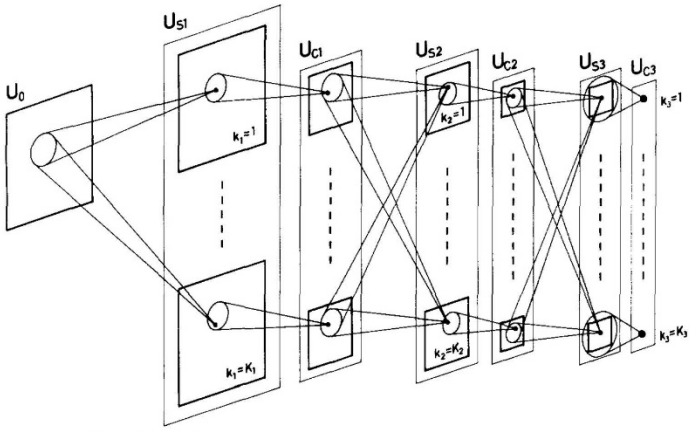
Zeng-Lau-Lin-Y., ICML 2019
Zeng-Lin-Y.-Zhou, JMLR 2021

Convolutional Neural Networks: shift invariances and locality

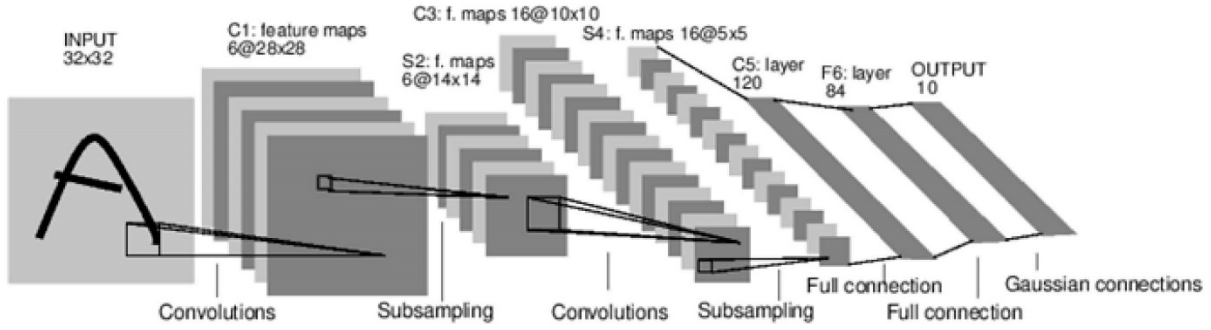
Biol. Cybernetics 36, 193–202 (1980)

Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position

Kunihiko Fukushima
NHK Broadcasting Science Research Laboratories, Kinuta, Setagaya, Tokyo, Japan



- Can be traced to *Neocognitron* of Kunihiko Fukushima (1979)
- Yann LeCun combined convolutional neural networks with back propagation (1989)
- Imposes **shift invariance** and **locality** on the weights
- Forward pass remains similar
- Backpropagation slightly changes – need to sum over the gradients from all spatial positions

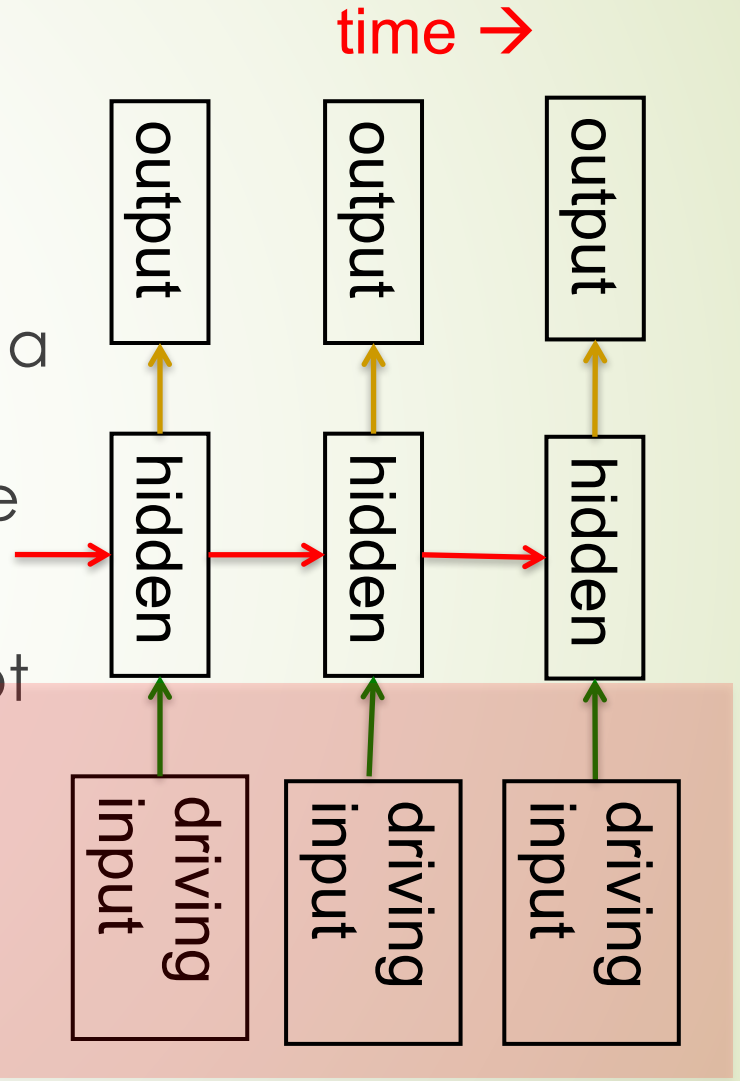


Time series: Linear Dynamical Systems (1940s-)

- The hidden state has linear dynamics with Gaussian noise and produces the observations using a linear model with Gaussian noise.
- Kalman Filter: A linearly transformed Gaussian is a Gaussian. So the distribution over the hidden state given the data so far is Gaussian. It can be computed using "Kalman filtering".
- To predict the next output (so that we can shoot down the missile) we need to infer the hidden state.

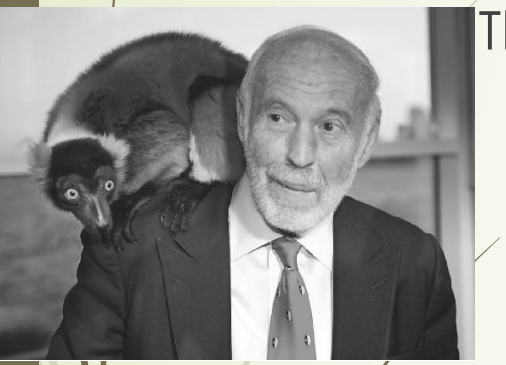
$$h_t = W_{hh} h_{t-1} + W_{hx} x_t + \epsilon_t^h$$

$$y_t = W_{yh} h_t + W_{yx} x_t + \epsilon_t^y$$

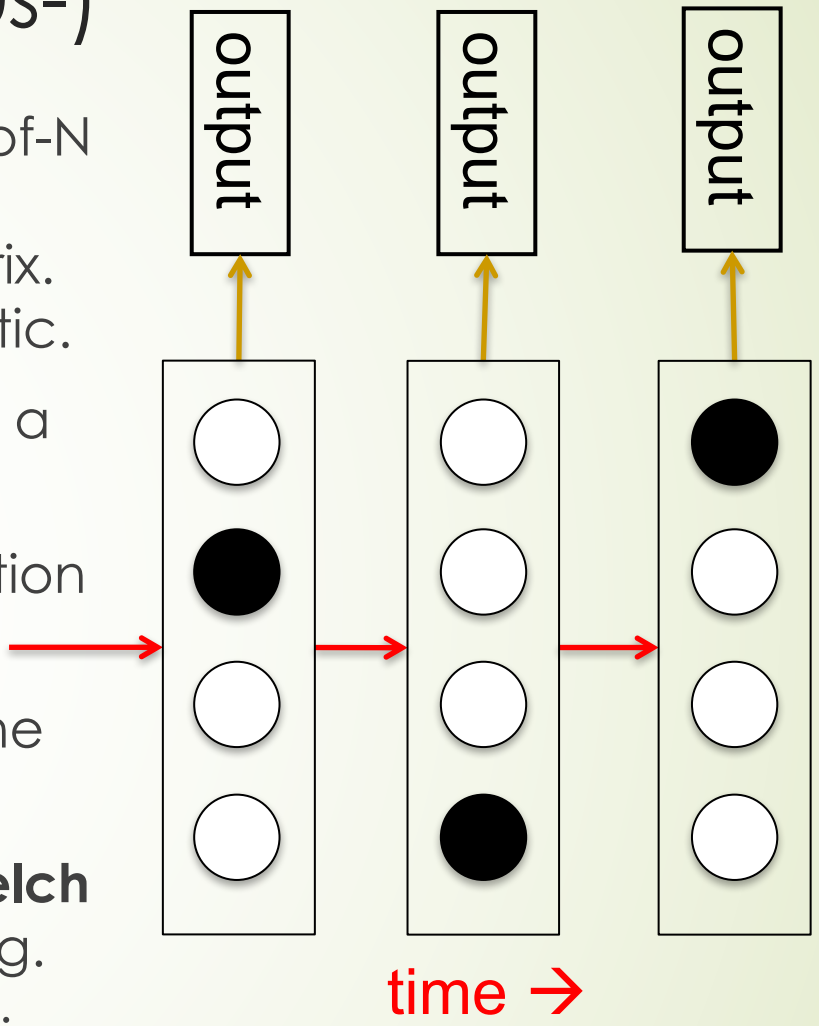


Hidden Markov Models (1970s-)

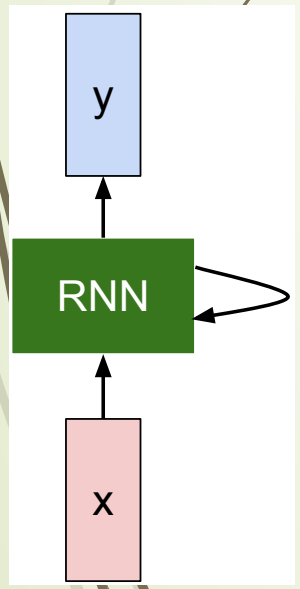
- Hidden Markov Models have a discrete one-of-N hidden state. Transitions between states are stochastic and controlled by a transition matrix. The outputs produced by a state are stochastic.
  - We cannot be sure which state produced a given output. So the state is "hidden".
  - It is easy to represent a probability distribution across N states with N numbers.
- To predict the next output we need to infer the probability distribution over hidden states.
  - HMMs have efficient algorithms (**Baum-Welch or EM Algorithm**) for inference and learning.
  - **Jim Simons** hires Lenny Baum as the founding member of Renaissance Technologies in 1979



Lenny Baum became a devoted Go player despite his deteriorating eyesight.



Recurrent Neural Networks (1986-)

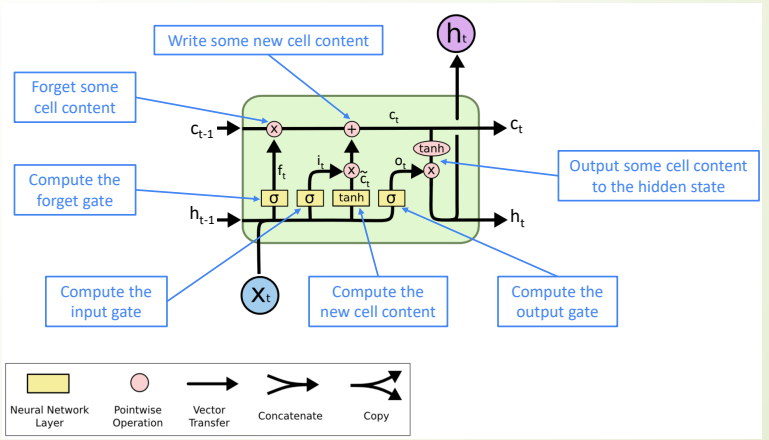


- **The issue of a hidden Markov model (HMM)**:
  - At each time step it must select one of its hidden states. So with N hidden states it can only remember log(N) bits about what it generated so far.
- RNNs are very powerful, because they combine two properties:
  - Distributed hidden state that allows them to store a lot of information about the past efficiently.
  - Non-linear dynamics that allows them to update their hidden state in complicated ways.
- Rumelhart et al. enables training by **BP** algorithm
  - With enough neurons and time, RNNs can compute anything that can be computed by your computer.

$$h_t = \sigma_h(W_{hh}h_{t-1} + W_{hx}x_t)$$
$$y_t = \sigma_y(W_{yh}h_t)$$

Long-Short-Term-Memory (LSTM)

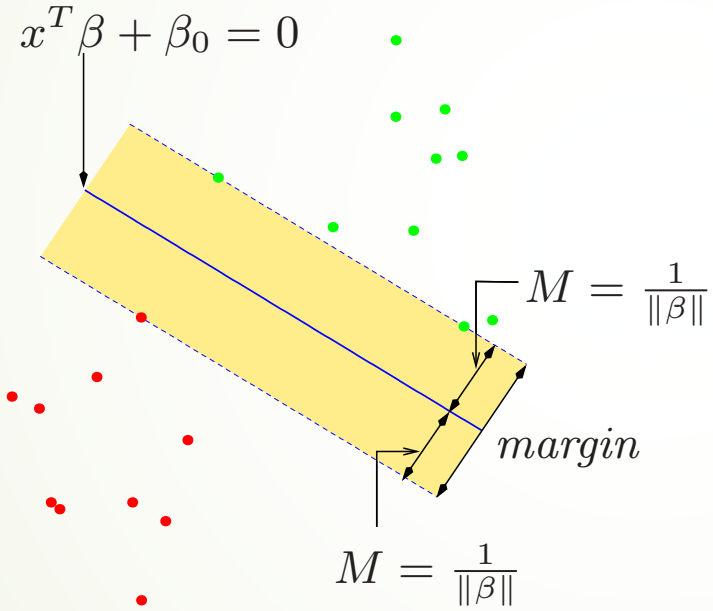
- Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". *Neural Computation*. **9** (8): 1735–1780. (https://www.bioinf.jku.at/publications/older/2604.pdf)
- Introduction of short cut path to learn deep networks overcoming the *vanishing gradient* problem, returned later in **ResNet** (2015).



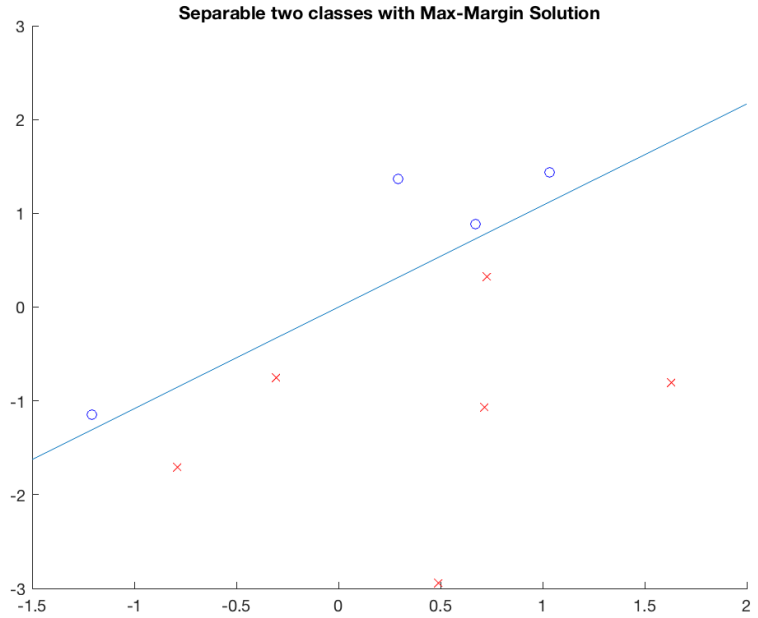
Max-Margin Classifier (SVM)

$$\text{minimize}_{\beta_0,\beta_1,\ldots,\beta_p} \|\beta\|^2 := \sum_j \beta_j^2$$

$$\text{subject to } y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq 1 \text{ for all } i$$



Vladmir Vapnik, 1994



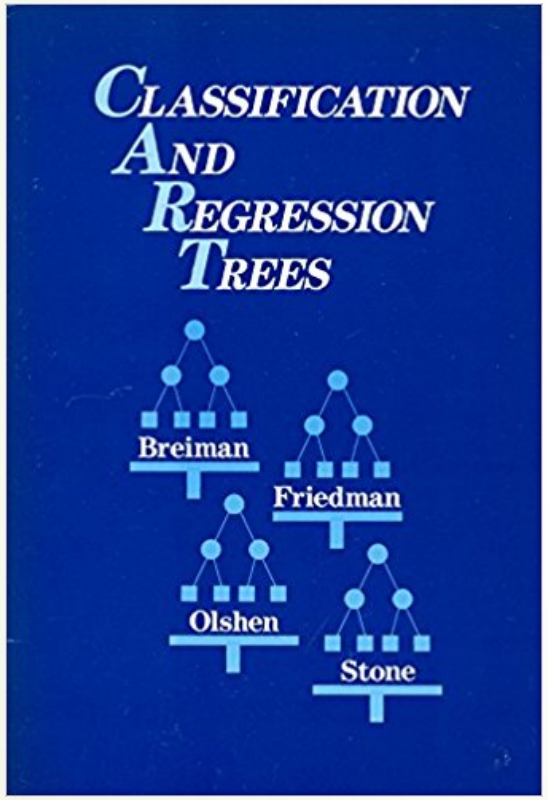
MNIST Dataset Test Error
LeCun et al. 1998

Simple SVM performs as well as Multilayer Convolutional Neural Networks which need careful tuning (LeNets)

Dark era for NN: 1998-2012

2000-2010: The Era of SVM, Boosting, … as dark nights of Neural Networks

Decision Trees and Boosting



- Breiman, Friedman, Olshen, Stone, (1983): CART
- ``The Boosting problem'' (M. Kearns & L. Valiant): **Can a set of weak learners create a single strong learner**? (三个臭皮匠顶个诸葛亮？)
- Breiman (1996): Bagging
- Freund, Schapire (1997): **AdaBoost** ("the best off-the-shelf algorithm" by Breiman)
- Breiman (2001): **Random Forests**

Restricted Boltzman Machine (Deep Learning)

- **Hinton and Salakhutdinov**, Reducing the Dimensionality of Data with Neural Networks, ***Science*, 2006**
- Reinvigorating research in Deep Learning
- Shows importance of ***pretraining (greedy layer-wise, a.k.a. block coordinate descent)***

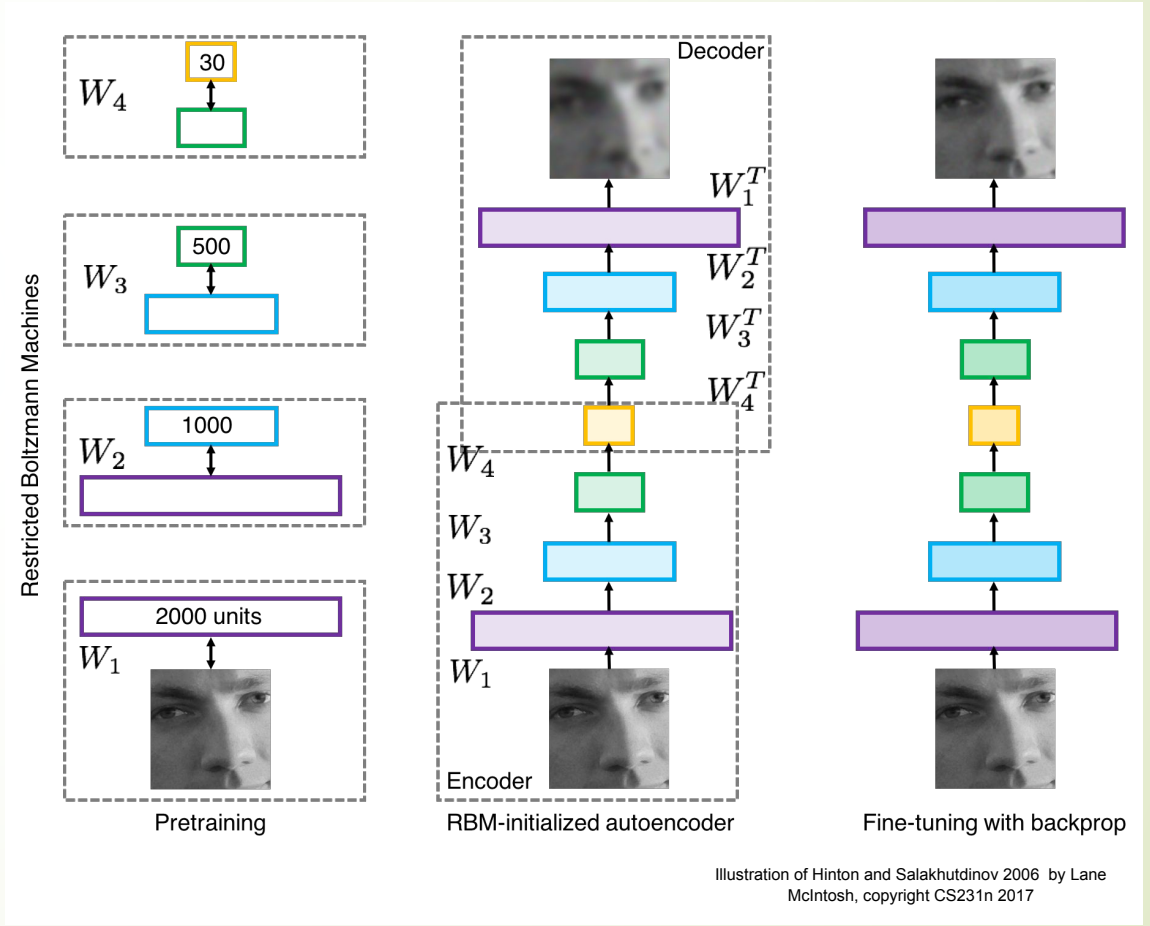
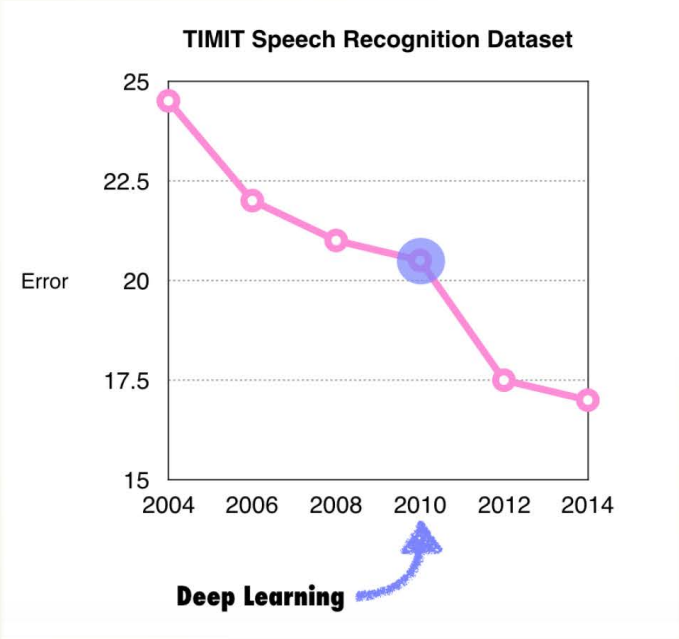


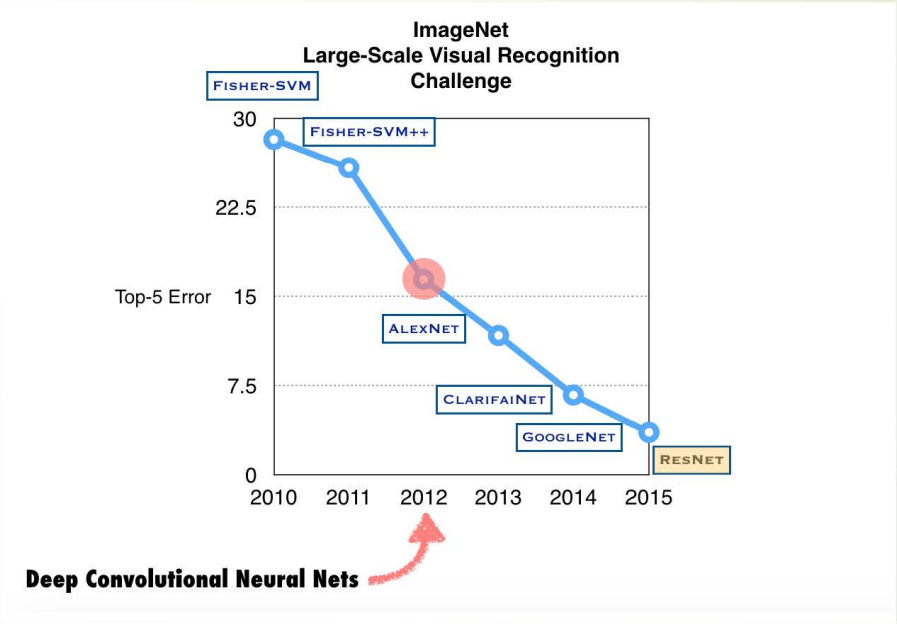
Illustration of Hinton and Salakhutdinov 2006 by Lane McIntosh, copyright CS231n 2017

Around the year of 2012: return of NN as `deep learning'

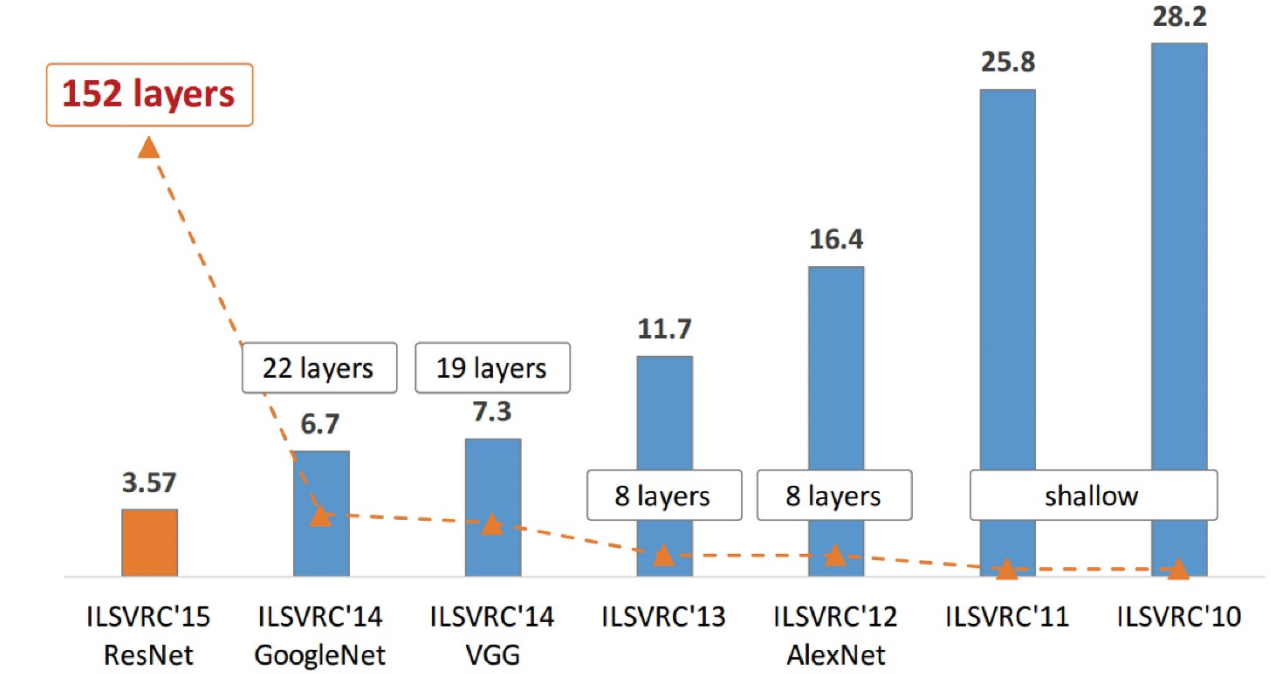
Speech Recognition: TIMIT



Computer Vision: ImageNet



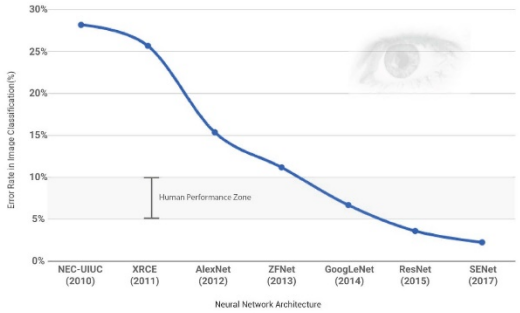
Depth as function of year



[He et al., 2016]

ILSVRC ImageNet Top 5 errors

- ImageNet (subset):
  - 1.2 million training images
  - 100,000 test images
  - 1000 classes
- ImageNet large-scale visual recognition Challenge



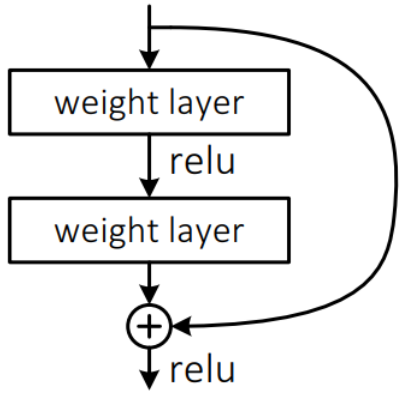
source: https://www.linkedin.com/pulse/must-read-path-breaking-papers-image-classification-muktabh-mayank

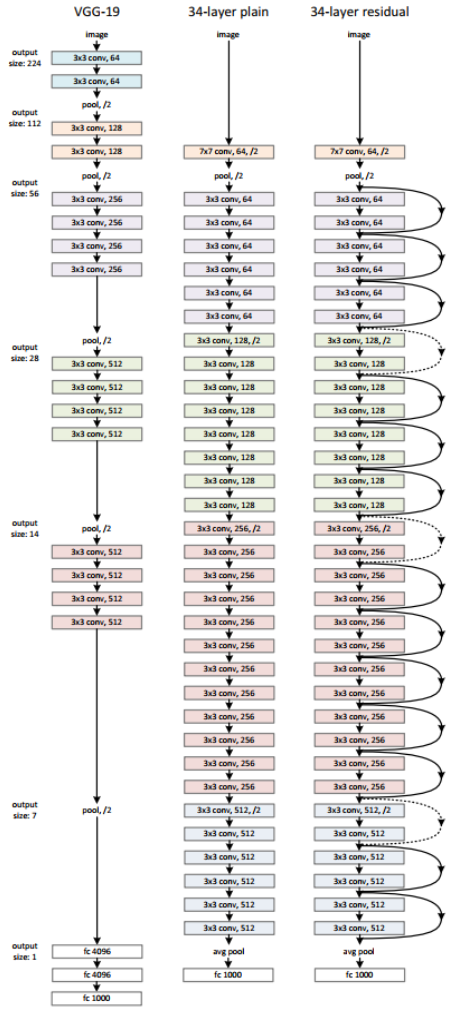
ResNet (2015)

[He-Zhang-Ren-Sun, 2015]

ILSVRC'15 classification winner
(3.57% top 5 error)

- Solves problem by adding skip connections
- Very deep: 152 layers
- No dropout
- Stride
- Batch normalization





Source: Deep Residual Learning for Image Recognition

GPU + Big labeled data

"We're at the beginning of a new day…
This is the beginning of the AI revolution."
— Jensen Huang, GTC Taiwan 2017

兩股力量驅動電腦的未來

深度學習點亮人工智慧紀元。

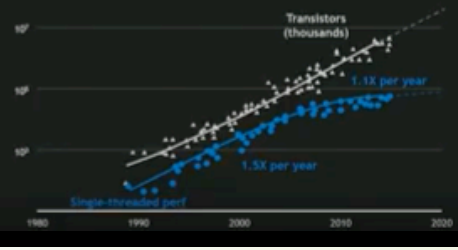
受到人腦的啟發，深度神經網路具備上億的類神經連結，藉由巨量資料來學習，這仰賴極大量的運算。

同時，摩爾定律已到了尾聲 - CPU已不可能再擴張成長。

程式設計人員無法創造出可以更有效率發現更多指令級並行性的的CPU架構。

電晶體持續每年增長50%，但是CPU效能僅能成長10%。

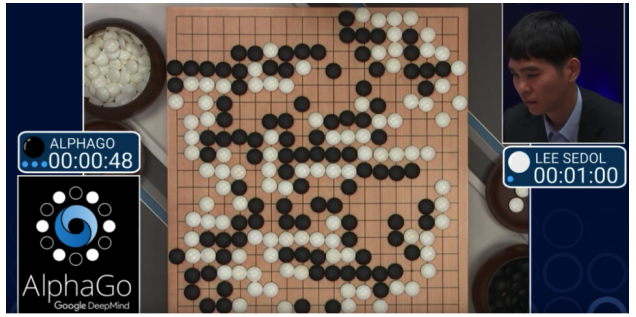
TWO FORCES DRIVING THE FUTURE OF COMPUTING



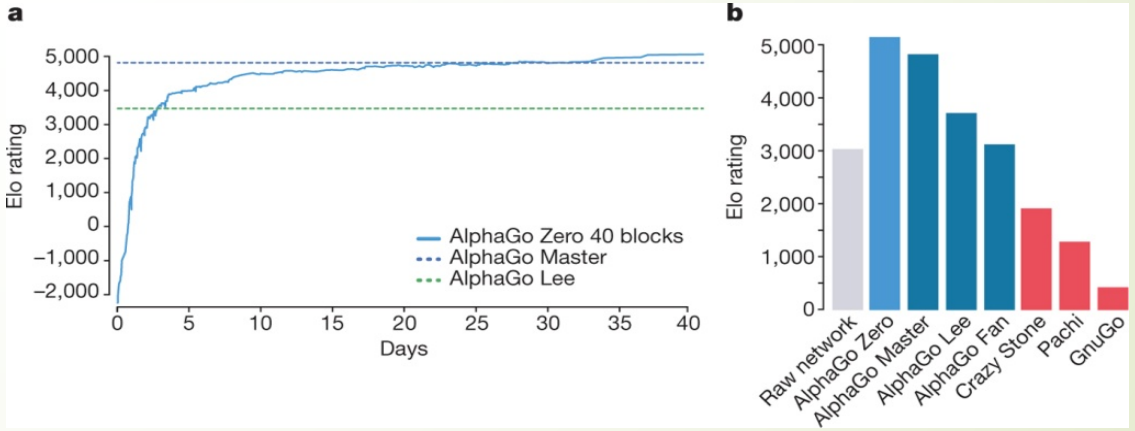
NVIDIA

Reaching Human Performance Level in Games

Deep Blue in 1997



AlphaGo "LEE" 2016



AlphaGo "ZERO" D Silver *et al. Nature* **550,** 354–359 (2017) doi:10.1038/nature24270

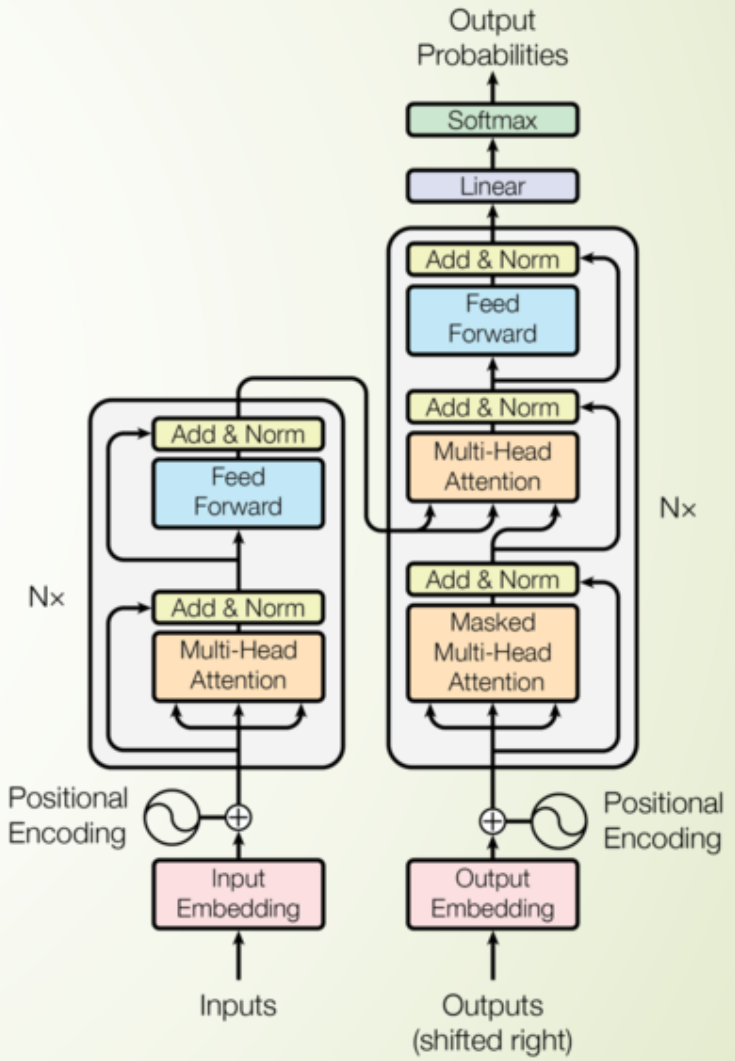
Natural Language Processing (NLP) and Machine Translation

- In **2013-2015**, **LSTM**s started achieving state-of-the-art results
  - Successful tasks include: handwriting recognition, speech
  - recognition, machine translation, parsing, image captioning
  - LSTM became the dominant approach
- In **2019**, other approaches (e.g. **Transformers**) have become more dominant for certain tasks.
  - For example in **WMT** (a MT conference + competition):
  - In WMT 2016, the summary report contains "RNN" 44 times
  - In WMT 2018, the report contains "RNN" 9 times and "Transformer" 63 times

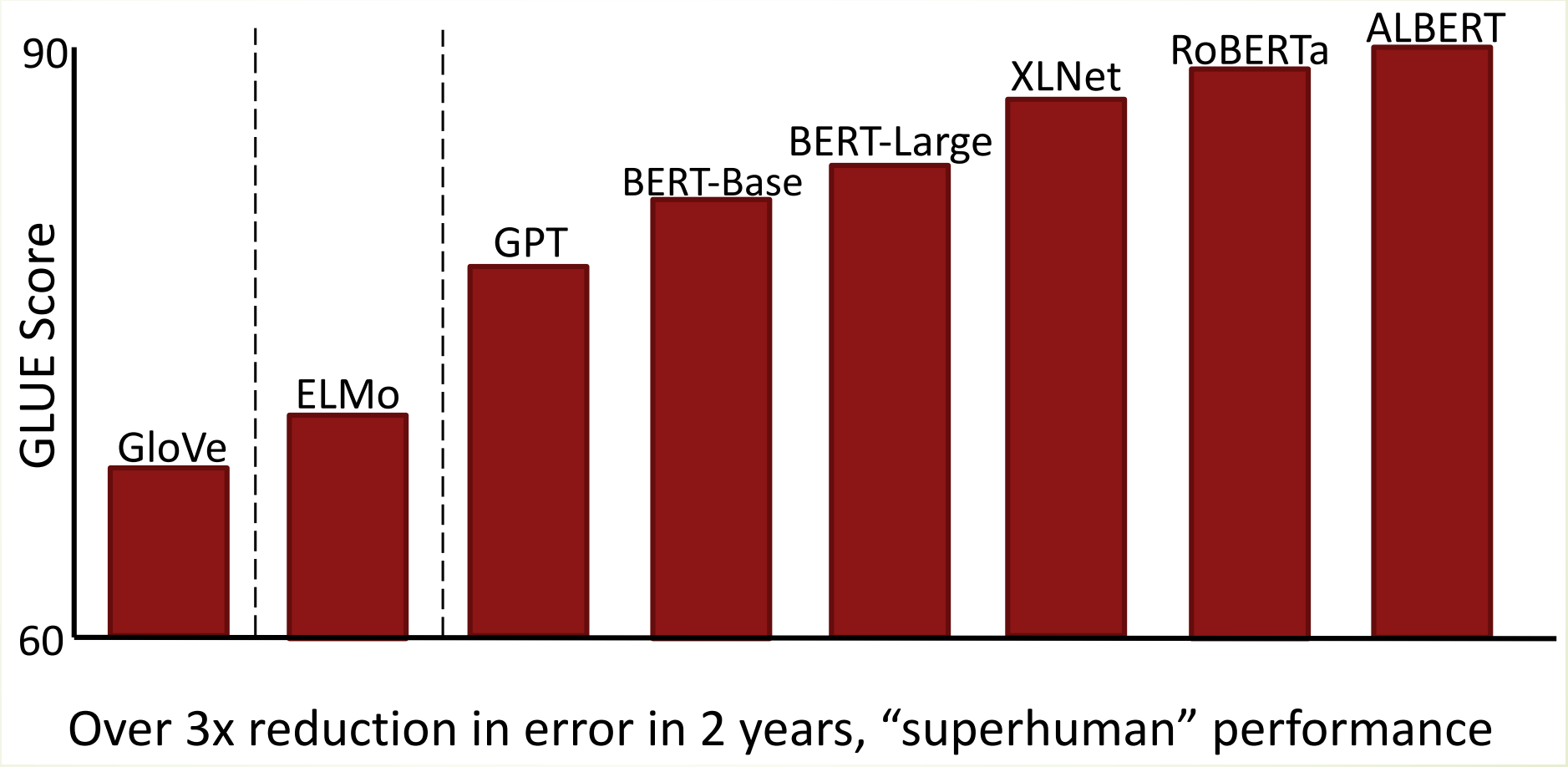
  - **Source:** "Findings of the 2016 Conference on Machine Translation (WMT16)", Bojar et al. 2016, http://www.statmt.org/wmt16/pdf/W16-2301.pdf
  - **Source:** "Findings of the 2018 Conference on Machine Translation (WMT18)", Bojar et al. 2018, http://www.statmt.org/wmt18/pdf/WMT028.pdf

Transformer (Vaswani et al. 2017) "**Attention is all you need**"

- https://arxiv.org/pdf/1706.03762.pdf
- **Non-recurrent** sequence-to-sequence model
- A **deep** model with a sequence of **attention**-based transformer blocks
- Depth allows a certain amount of lateral information transfer in understanding sentences, in slightly unclear ways
- Final cost/error function is standard cross-entropy error on top of a softmax classifier
- Initially built for NMT:
  - Task: machine translation with parallel corpus
  - Predict each translated word

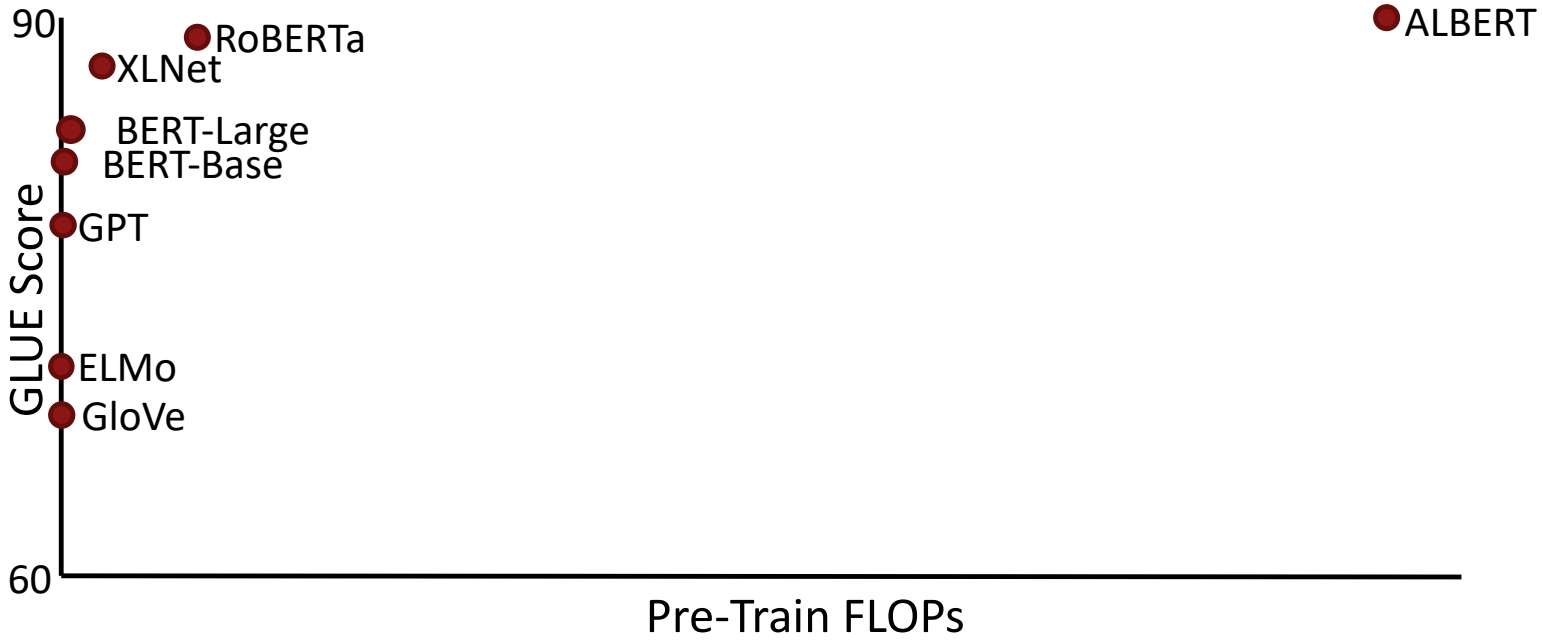


Rapid Progress for NLP Pretraining (GLUE Benchmark)



Over 3x reduction in error in 2 years, “superhuman” performance

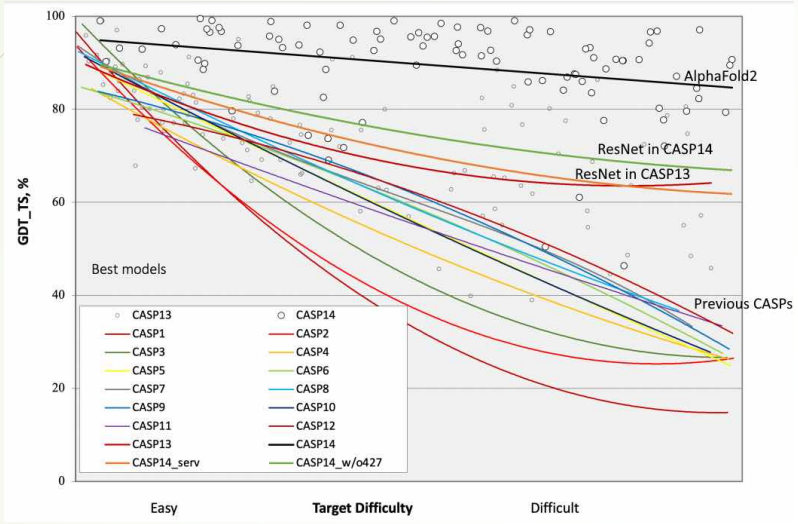
More compute, more better?



ALBERT uses 10x more compute than RoBERTa

Protein Folding Structure Prediction

AlphaFold



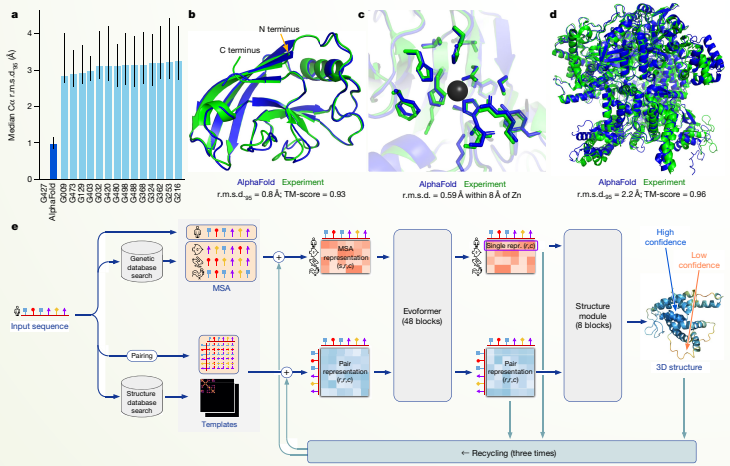


Fig. 1 | AlphaFold produces highly accurate structures. a, The performance of AlphaFold on the CASP14 dataset ($n = 87$ protein domains) relative to the top-15 entries (out of 146 entries), group numbers correspond to the numbers assigned to entrants by CASP. Data are median and the 95% confidence interval of the median, estimated from 10,000 bootstrap samples. **b**, Our prediction of CASP14 target T1049 (PDB 6Y4F, blue) compared with the true (experimental) structure (green). Four residues in the C terminus of the crystal structure are *B*-factor outliers and are not depicted. **c**, CASP14 target T1056 (PDB 6YJ1). An example of a well-predicted zinc-binding site (AlphaFold has accurate side chains even though it does not explicitly predict the zinc ion). **d**, CASP target T1044 (PDB 6VR4)—a 2,180-residue single chain—was predicted with correct domain packing (the prediction was made after CASP using AlphaFold without intervention). **e**, Model architecture. Arrows show the information flow among the various components described in this paper. Array shapes are shown in parentheses with *s*, number of sequences (N_{seq} in the main text); *r*, number of residues (N_{res} in the main text); *c*, number of channels.

Article

Highly accurate protein structure prediction with AlphaFold

https://doi.org/10.1038/s41586-021-03819-2

Received: 11 May 2021

Accepted: 12 July 2021

Published online: 15 July 2021

Open access

Check for updates

John Jumper[1,4]✉, Richard Evans[1,4], Alexander Pritzel[1,4], Tim Green[1,4], Michael Figurnov[1,4], Olaf Ronneberger[1,4], Kathryn Tunyasuvunakool[1,4], Russ Bates[1,4], Augustin Žídek[1,4], Anna Potapenko[1,4], Alex Bridgland[1,4], Clemens Meyer[1,4], Simon A. A. Kohl[1,4], Andrew J. Ballard[1,4], Andrew Cowie[1,4], Bernardino Romera-Paredes[1,4], Stanislav Nikolov[1,4], Rishub Jain[1,4], Jonas Adler[1], Trevor Back[1], Stig Petersen[1], David Reiman[1], Ellen Clancy[1], Michal Zielinski[1], Martin Steinegger[2,3], Michalina Pacholska[1], Tamas Berghammer[1], Sebastian Bodenstein[1], David Silver[1], Oriol Vinyals[1], Andrew W. Senior[1], Koray Kavukcuoglu[1], Pushmeet Kohli[1] & Demis Hassabis[1,4]✉

Proteins are essential to life, and understanding their structure can facilitate a mechanistic understanding of their function. Through an enormous experimental effort[1–4], the structures of around 100,000 unique proteins have been determined[5], but this represents a small fraction of the billions of known protein sequences[6,7]. Structural coverage is bottlenecked by the months to years of painstaking effort required to determine a single protein structure. Accurate computational approaches are needed to address this gap and to enable large-scale structural bioinformatics. Predicting the three-dimensional structure that a protein will adopt based solely on its amino acid sequence—the structure prediction component of the 'protein folding problem'[8]—has been an important open research problem for more than 50 years[9]. Despite recent progress[10–14], existing methods fall far short of atomic accuracy, especially when no homologous structure is available. Here we provide the first computational method that can regularly predict protein structures with atomic accuracy even in cases in which no similar structure is known. We validated an entirely redesigned version of our neural network-based model, AlphaFold, in the challenging 14th Critical Assessment of protein Structure Prediction (CASP14)[15], demonstrating accuracy competitive with experimental structures in a majority of cases and greatly outperforming other methods. Underpinning the latest version of AlphaFold is a novel machine learning approach that incorporates physical and biological knowledge about protein structure, leveraging multi-sequence alignments, into the design of the deep learning algorithm.

The development of computational methods to predict three-dimensional (3D) protein structures from the protein sequence has proceeded along two complementary paths that focus on either the physical interactions or the evolutionary history. The physical interaction programme heavily integrates our understanding of molecular driving forces into either thermodynamic or kinetic simulation of protein physics[16] or statistical approximations thereof[17]. Although theoretically very appealing, this approach has proved highly challenging for even moderate-sized proteins due to the computational intractability of molecular simulation, the context dependence of protein stability and the difficulty of producing sufficiently accurate models of protein physics. The evolutionary programme has provided an alternative in recent years, in which the constraints on protein structure are derived from bioinformatics analysis of the evolutionary history of proteins, homology to solved structures[18,19] and pairwise evolutionary correlations[20–24]. This bioinformatics approach has benefited greatly from the steady growth of experimental protein structures deposited in the Protein Data Bank (PDB)[5], the explosion of genomic sequencing and the rapid development of deep learning techniques to interpret these correlations. Despite these advances, contemporary physical and evolutionary-history-based approaches produce predictions that are far short of experimental accuracy in the majority of cases in which a close homologue has not been solved experimentally and this has limited their utility for many biological applications.

In this study, we develop the first, to our knowledge, computational approach capable of predicting protein structures to near experimental accuracy in a majority of cases. The neural network AlphaFold that we developed was entered into the CASP14 assessment (May–July 2020; entered under the team name 'AlphaFold2' and a completely different model from our CASP13 AlphaFold system[10]). The CASP assessment is carried out biennially using recently solved structures that have not been deposited in the PDB or publicly disclosed so that it is a blind test

[1]DeepMind, London, UK. [2]School of Biological Sciences, Seoul National University, Seoul, South Korea. [3]Artificial Intelligence Institute, Seoul National University, Seoul, South Korea. [4]These authors contributed equally: John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Demis Hassabis.
✉e-mail: jumper@deepmind.com; dhcontact@deepmind.com

AI for Science

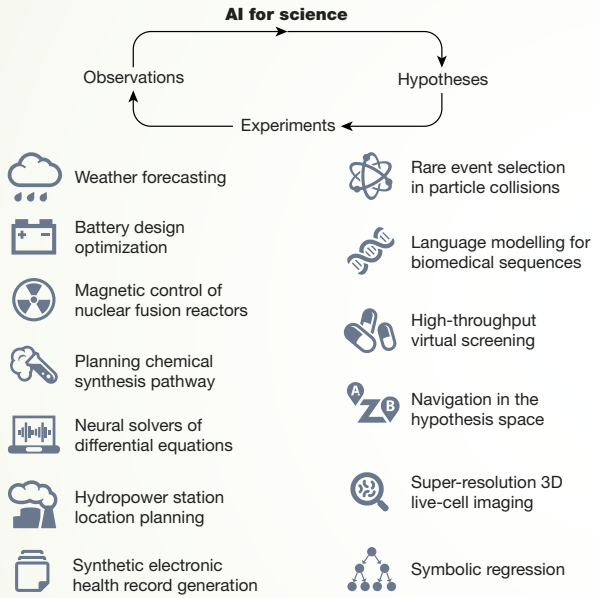


Fig. 1 | Science in the age of artificial intelligence. Scientific discovery is a multifaceted process that involves several interconnected stages, including hypothesis formation, experimental design, data collection and analysis. AI is poised to reshape scientific discovery by augmenting and accelerating research at each stage of this process. The principles and illustrative studies shown here highlight the contributions to enhance scientific understanding and discovery.

Review

Scientific discovery in the age of artificial intelligence

https://doi.org/10.1038/s41586-023-06221-2

Received: 30 March 2022

Accepted: 16 May 2023

Published online: 2 August 2023

Check for updates

Hanchen Wang[1,2,37,38,39], Tianfan Fu[3,39], Yuanqi Du[4,39], Wenhao Gao[5], Kexin Huang[6], Ziming Liu[7], Payal Chandak[8], Shengchao Liu[9,10], Peter Van Katwyk[11,12], Andreea Deac[9,10], Anima Anandkumar[2,13], Karianne Bergen[11,12], Carla P. Gomes[4], Shirley Ho[14,15,16,17], Pushmeet Kohli[18], Joan Lasenby[1], Jure Leskovec[6], Tie-Yan Liu[19], Arjun Manrai[20], Debora Marks[21,22], Bharath Ramsundar[23], Le Song[24,25], Jimeng Sun[26], Jian Tang[9,27,28], Petar Veličković[17,29], Max Welling[30,31], Linfeng Zhang[32,33], Connor W. Coley[5,34], Yoshua Bengio[9,10] & Marinka Zitnik[20,22,35,36]✉

Artificial intelligence (AI) is being increasingly integrated into scientific discovery to augment and accelerate research, helping scientists to generate hypotheses, design experiments, collect and interpret large datasets, and gain insights that might not have been possible using traditional scientific methods alone. Here we examine breakthroughs over the past decade that include self-supervised learning, which allows models to be trained on vast amounts of unlabelled data, and geometric deep learning, which leverages knowledge about the structure of scientific data to enhance model accuracy and efficiency. Generative AI methods can create designs, such as small-molecule drugs and proteins, by analysing diverse data modalities, including images and sequences. We discuss how these methods can help scientists throughout the scientific process and the central issues that remain despite such advances. Both developers and users of AI toolsneed a better understanding of when such approaches need improvement, and challenges posed by poor data quality and stewardship remain. These issues cut across scientific disciplines and require developing foundational algorithmic approaches that can contribute to scientific understanding or acquire it autonomously, making them critical areas of focus for AI innovation.

The foundation for forming scientific insights and theories is laid by how data are collected, transformed and understood. The rise of deep learning in the early 2010s has significantly expanded the scope and ambition of these scientific discovery processes[1]. Artificial intelligence (AI) is increasingly used across scientific disciplines to integrate massive datasets, refine measurements, guide experimentation, explore the space of theories compatible with the data, and provide actionable and reliable models integrated with scientific workflows for autonomous discovery.

Data collection and analysis are fundamental to scientific understanding and discovery, two of the central aims in science[2], and quantitative methods and emerging technologies, from physical instruments such as microscopes to research techniques such as bootstrapping, have long been used to reach these aims[3]. The introduction of digitization in the 1950s paved the way for the general use of computing in scientific research. The rise of data science since the 2010s has enabled AI to provide valuable guidance by identifying scientifically relevant patterns from large datasets.

Although scientific practices and procedures vary across stages of scientific research, the development of AI algorithms cuts across traditionally isolated disciplines (Fig. 1). Such algorithms can enhance the design and execution of scientific studies. They are becoming

[1]Department of Engineering, University of Cambridge, Cambridge, UK. [2]Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA, USA. [3]Department of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, GA, USA. [4]Department of Computer Science, Cornell University, Ithaca, NY, USA. [5]Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA. [6]Department of Computer Science, Stanford University, Stanford, CA, USA. [7]Department of Physics, Massachusetts Institute of Technology, Cambridge, MA, USA. [8]Harvard-MIT Program in Health Sciences and Technology, Cambridge, MA, USA. [9]Mila – Quebec AI Institute, Montreal, Quebec, Canada. [10]Université de Montréal, Montreal, Quebec, Canada. [11]Department of Earth, Environmental and Planetary Sciences, Brown University, Providence, RI, USA. [12]Data Science Institute, Brown University, Providence, RI, USA. [13]NVIDIA, Santa Clara, CA, USA. [14]Center for Computational Astrophysics, Flatiron Institute, New York, NY, USA. [15]Department of Astrophysical Sciences, Princeton University, Princeton, NJ, USA. [16]Department of Physics, Carnegie Mellon University, Pittsburgh, PA, USA. [17]Department of Physics and Center for Data Science, New York University, New York, NY, USA. [18]Google DeepMind, London, UK. [19]Microsoft Research, Beijing, China. [20]Department of Biomedical Informatics, Harvard Medical School, Boston, MA, USA. [21]Department of Systems Biology, Harvard Medical School, Boston, MA, USA. [22]Broad Institute of MIT and Harvard, Cambridge, MA, USA. [23]Deep Forest Sciences, Palo Alto, CA, USA. [24]BioMap, Beijing, China. [25]Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, United Arab Emirates. [26]University of Illinois at Urbana-Champaign, Champaign, IL, USA. [27]HEC Montréal, Montreal, Quebec, Canada. [28]CIFAR AI Chair, Toronto, Ontario, Canada. [29]Department of Computer Science and Technology, University of Cambridge, Cambridge, UK. [30]University of Amsterdam, Amsterdam, Netherlands. [31]Microsoft Research Amsterdam, Amsterdam, Netherlands. [32]DP Technology, Beijing, China. [33]AI for Science Institute, Beijing, China. [34]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA. [35]Harvard Data Science Initiative, Cambridge, MA, USA. [36]Kempner Institute for the Study of Natural and Artificial Intelligence, Harvard University, Cambridge, MA, USA. [37]Present address: Department of Research and Early Development, Genentech Inc, South San Francisco, CA, USA. [38]Present address: Department of Computer Science, Stanford University, Stanford, CA, USA. [39]These authors contributed equally: Hanchen Wang, Tianfan Fu, Yuanqi Du. ✉e-mail: marinka@hms.harvard.edu

ChatGPT (GPT 3.5-4)

ChatGPT

94 languages

Article Talk | Read View source View history Tools

From Wikipedia, the free encyclopedia

ChatGPT, which stands for **Chat Generative Pre-trained Transformer**, is a large language model-based chatbot developed by OpenAI and launched on November 30, 2022, notable for enabling users to refine and steer a conversation towards a desired length, format, style, level of detail, and language used. Successive prompts and replies, known as prompt engineering, are considered at each conversation stage as a context.[2]

ChatGPT is built upon GPT-3.5 and GPT-4 —members of OpenAI's proprietary series of generative pre-trained transformer (GPT) models, based on the transformer architecture developed by Google[3]—and it is fine-tuned for conversational applications using a combination of supervised and reinforcement learning techniques.[4] ChatGPT was released as a freely available research preview, but due to its popularity, OpenAI now operates the service on a freemium model. It allows users on its free tier to access the GPT-3.5-based version. In contrast, the more advanced GPT-4 based version and priority access to newer features are provided to paid subscribers under the commercial name "ChatGPT Plus".

By January 2023, it had become what was then the fastest-growing consumer software application in history, gaining over 100 million users and contributing to OpenAI's valuation growing to US$29 billion.[5][6] Within months, Google, Baidu, and Meta accelerated the development of their competing products: Bard, Ernie Bot, and LLaMA.[7] Microsoft launched its Bing Chat based on OpenAI's GPT-4. Some observers expressed concern over the potential of ChatGPT to displace or atrophy human intelligence and its potential to enable plagiarism or fuel misinformation.[4][8]

Training

ChatGPT is based on particular GPT foundation models, namely GPT-3.5 and GPT-4, that were fine-tuned to target conversational usage.[9] The fine-tuning process leveraged both supervised learning as well as

ChatGPT

Developer(s)	OpenAI
Initial release	November 30, 2022; 9 months ago
Stable release	August 3, 2023; 31 days ago[1]
Written in	Python
Engine	GPT-3.5 GPT-4
Platform	Cloud computing platforms
Type	Chatbot Large language model Generative text-to-image model Generative pre-trained transformer
License	Proprietary
Website	chat.openai.com/chat

manifold 是什么?

GPT-3.5-Turbo Poe

Manifold（流形）是一个数学概念，它用于描述具有某种结构的空间。简而言之，流形是一个局部上类似于欧几里得空间的空间，但在整体上可能具有非平凡的拓扑结构。流形的形状可以是任意的，可以是平面、球体、圆环、螺旋线等等。

流形在数学中具有广泛的应用，特别是在拓扑学、微积分、几何学、物理学、计算机图形学等领域。在机器学习和人工智能领域，流形也被广泛应用于数据降维、聚类、分类等问题中。

在机器学习中，流形学习（manifold learning）是一种无监督学习方法，它旨在从高维数据中发现潜在的低维流形结构。流形学习可以帮助我们更好地理解数据的内在结构和特征，从而提高数据分析和处理的效率和准确性。

What's HodgeRank?

GPT-4-Turbo Poe

HodgeRank is a method for aggregating preferences or rankings that is based on the mathematical theory of Hodge decompositions. It was introduced by Xiaoye Jiang, Lek-Heng Lim, Yuan Yao, and Yinyu Ye in a 2010 paper titled "Statistical Ranking and Combinatorial Hodge Theory."

In many situations, such as recommendation systems, voting systems, or sports competitions, we need to aggregate individual preference rankings to produce a global ranking. Traditional methods, like Borda count or Condorcet, have their limitations and may not always produce satisfactory results.

HodgeRank addresses these limitations by decomposing the preference data into three components: a consistent global ranking, a cyclic component, and a local "noise" component. This decomposition provides a more nuanced understanding of the preference data and can lead to more accurate and robust global rankings.

The HodgeRank method has been applied in various domains, such as sports ranking, image annotation, and recommendation systems, where it has been shown to outperform conventional ranking methods.

- https://poe.com/

DeepSeek

deepseek

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

DeepSeek-AI

research@deepseek.com

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.

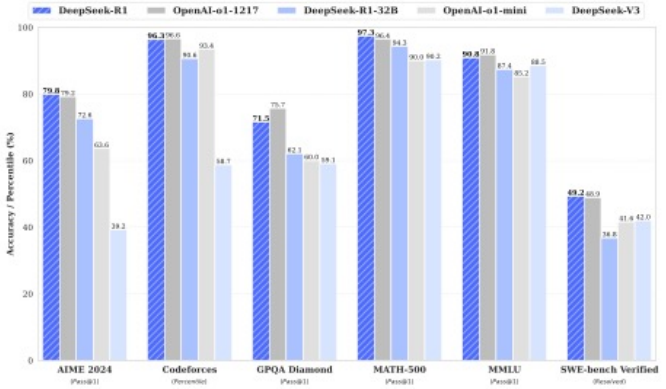


Figure 1 | Benchmark performance of DeepSeek-R1.

https://github.com/deepseek-ai/DeepSeek-R1

Yann LeCun • 2nd
VP & Chief AI Scientist at Meta
1w •

+ Follow ···

To people who see the performance of DeepSeek and think:
"China is surpassing the US in AI."
You are reading this wrong.
The correct reading is:
"Open source models are surpassing proprietary ones."

DeepSeek has profited from open research and open source (e.g. PyTorch and Llama from Meta)
They came up with new ideas and built them on top of other people's work.
Because their work is published and open source, everyone can profit from it.
That is the power of open research and open source.

Sou-Cheng Choi and 34,821 others 926 comments · 2,205 reposts

AI for Math?

Article

Solving olympiad geometry without human demonstrations

https://doi.org/10.1038/s41586-023-06747-5

Received: 30 April 2023

Accepted: 13 October 2023

Published online: 17 January 2024

Open access

Check for updates

Trieu H. Trinh[1,2]✉, Yuhuai Wu[1], Quoc V. Le[1], He He[2] & Thang Luong[1]✉

Proving mathematical theorems at the olympiad level represents a notable milestone in human-level automated reasoning[1–4], owing to their reputed difficulty among the world's best talents in pre-university mathematics. Current machine-learning approaches, however, are not applicable to most mathematical domains owing to the high cost of translating human proofs into machine-verifiable format. The problem is even worse for geometry because of its unique translation challenges[1,5], resulting in severe scarcity of training data. We propose AlphaGeometry, a theorem prover for Euclidean plane geometry that sidesteps the need for human demonstrations by synthesizing millions of theorems and proofs across different levels of complexity. AlphaGeometry is a neuro-symbolic system that uses a neural language model, trained from scratch on our large-scale synthetic data, to guide a symbolic deduction engine through infinite branching points in challenging problems. On a test set of 30 latest olympiad-level problems, AlphaGeometry solves 25, outperforming the previous best method that only solves ten problems and approaching the performance of an average International Mathematical Olympiad (IMO) gold medallist. Notably, AlphaGeometry produces human-readable proofs, solves all geometry problems in the IMO 2000 and 2015 under human expert evaluation and discovers a generalized version of a translated IMO theorem in 2004.

Proving theorems showcases the mastery of logical reasoning and the ability to search through an infinitely large space of actions towards a target, signifying a remarkable problem-solving skill. Since the 1950s (refs. 6,7), the pursuit of better theorem-proving capabilities has been a constant focus of artificial intelligence (AI) research[8]. Mathematical olympiads are the most reputed theorem-proving competitions in the world, with a similarly long history dating back to 1959, playing an instrumental role in identifying exceptional talents in problem solving. Matching top human performances at the olympiad level has become a notable milestone of AI research[2–4].

Theorem proving is difficult for learning-based methods because training data of human proofs translated into machine-verifiable languages are scarce in most mathematical domains. Geometry stands out among other olympiad domains because it has very few proof examples in general-purpose mathematical languages such as Lean[9] owing to translation difficulties unique to geometry[1,5]. Geometry-specific languages, on the other hand, are narrowly defined and thus unable to express many human proofs that use tools beyond the scope of geometry, such as complex numbers (Extended Data Figs. 3 and 4). Overall, this creates a data bottleneck, causing geometry to lag behind in recent progress that uses human demonstrations[2–4]. Current approaches to geometry, therefore, still primarily rely on symbolic methods and human-designed, hard-coded search heuristics[10–14].

We present an alternative method for theorem proving using synthetic data, thus sidestepping the need for translating human-provided proof examples. We focus on Euclidean plane geometry and exclude topics such as geometric inequalities and combinatorial geometry. By using existing symbolic engines on a diverse set of random theorem premises, we extracted 100 million synthetic theorems and their proofs, many with more than 200 proof steps, four times longer than the average proof length of olympiad theorems. We further define and use the concept of dependency difference in synthetic proof generation, allowing our method to produce nearly 10 million synthetic proof steps that construct auxiliary points, reaching beyond the scope of pure symbolic deduction. Auxiliary construction is geometry's instance of exogenous term generation, representing the infinite branching factor of theorem proving, and widely recognized in other mathematical domains as the key challenge to proving many hard theorems[1,2]. Our work therefore demonstrates a successful case of generating synthetic data and learning to solve this key challenge. With this solution, we present a general guiding framework and discuss its applicability to other domains in Methods section 'AlphaGeometry framework and applicability to other domains'.

We pretrain a language model on all generated synthetic data and fine-tune it to focus on auxiliary construction during proof search, delegating all deduction proof steps to specialized symbolic engines. This follows standard settings in the literature, in which language models such as GPT-f (ref. 15), after being trained on human proof examples, can generate exogenous proof terms as inputs to fast and accurate symbolic engines such as nlinarith or ring[2,3,16], using the best of both worlds. Our geometry theorem prover AlphaGeometry, illustrated in Fig. 1, produces human-readable proofs, substantially outperforms the previous state-of-the-art geometry-theorem-proving computer program and approaches the performance of an average IMO gold

[1]Google Deepmind, Mountain View, CA, USA. [2]Computer Science Department, New York University, New York, NY, USA. ✉e-mail: thtrieu@google.com; thangluong@google.com

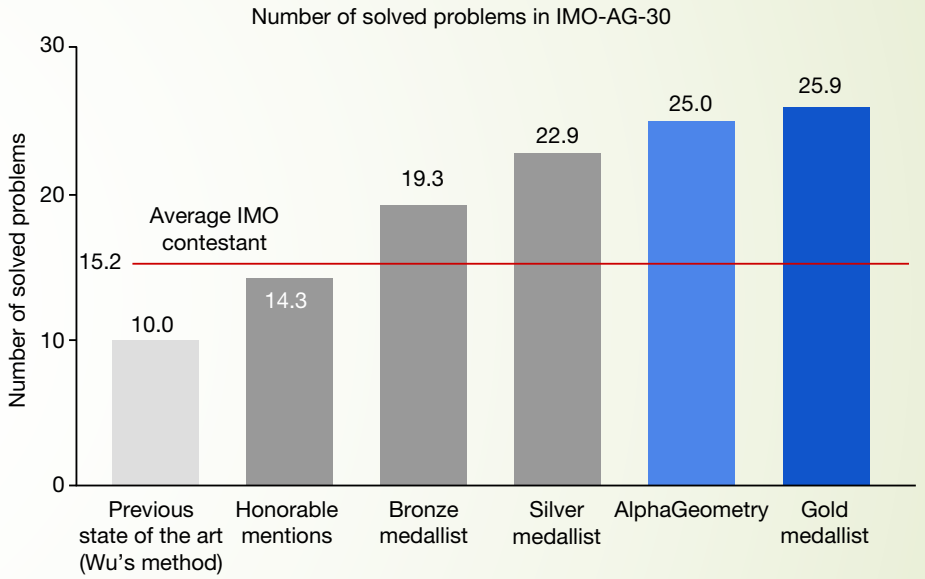


Fig. 2 | AlphaGeometry advances the current state of geometry theorem prover from below human level to near gold-medallist level. The test

AlphaGeometry: Exploitation of Symbolic deduction + Exploration of Language model

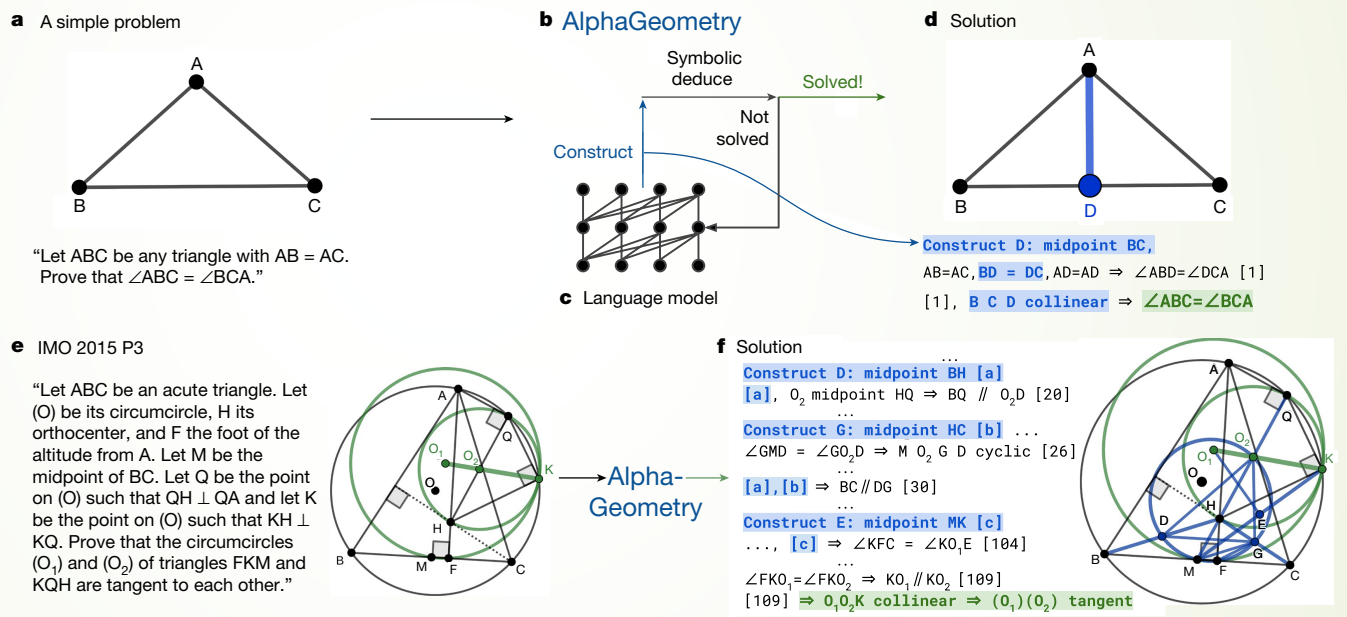


Fig. 1 | Overview of our neuro-symbolic AlphaGeometry and how it solves both a simple problem and the IMO 2015 Problem 3. The top row shows how AlphaGeometry solves a simple problem. **a**, The simple example and its diagram. **b**, AlphaGeometry initiates the proof search by running the symbolic deduction engine. The engine exhaustively deduces new statements from the theorem premises until the theorem is proven or new statements are exhausted. **c**, Because the symbolic engine fails to find a proof, the language model constructs one auxiliary point, growing the proof state before the symbolic engine retries. The loop continues until a solution is found. **d**, For the simple example, the loop terminates after the first auxiliary construction "D as the midpoint of BC". The proof consists of two other steps, both of which make use of the midpoint properties: "BD = DC" and "B, D, C are collinear", highlighted in blue. The bottom row shows how AlphaGeometry solves the IMO 2015 Problem 3 (IMO 2015 P3). **e**, The IMO 2015 P3 problem statement and diagram. **f**, The solution of IMO 2015 P3 has three auxiliary points. In both solutions, we arrange language model outputs (blue) interleaved with symbolic engine outputs to reflect their execution order. Note that the proof for IMO 2015 P3 in **f** is greatly shortened and edited for illustration purposes. Its full version is in the Supplementary Information.

The Nobel Prize in Physics 2024

Ill. Niklas Elmehed © Nobel Prize Outreach

John J. Hopfield

Prize share: 1/2

Ill. Niklas Elmehed © Nobel Prize Outreach

Geoffrey E. Hinton

Prize share: 1/2

The Nobel Prize in Physics 2024 was awarded jointly to John J. Hopfield and Geoffrey E. Hinton "for foundational discoveries and inventions that enable machine learning with artificial neural networks"

The Nobel Prize in Chemistry 2024

Ill. Niklas Elmehed © Nobel Prize Outreach

David Baker

Prize share: 1/2

Ill. Niklas Elmehed © Nobel Prize Outreach

Demis Hassabis

Prize share: 1/4

Ill. Niklas Elmehed © Nobel Prize Outreach

John M. Jumper

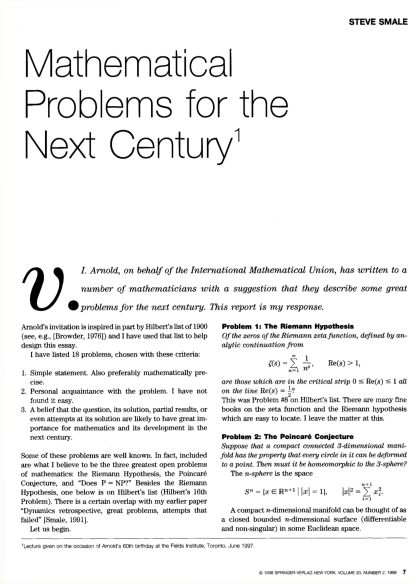
Prize share: 1/4

The Nobel Prize in Chemistry 2024 was divided, one half awarded to David Baker "for computational protein design", the other half jointly to Demis Hassabis and John M. Jumper "for protein structure prediction"

“Initialization”: Smale’s 18 Problem

Problem 18: Limits of Intelligence

What are the limits of intelligence, both artificial and human?



STEVE SMALE

Mathematical Problems for the Next Century[1]

V. I. Arnold, on behalf of the International Mathematical Union, has written to a number of mathematicians with a suggestion that they describe some great problems for the next century. This report is my response.

Arnold's invitation is inspired in part by Hilbert's list of 1900 (see, e.g., [Browder, 1976]) and I have used that list to help design this essay.

I have listed 18 problems, chosen with these criteria:

1. Simple statement. Also preferably mathematically precise.
2. Personal acquaintance with the problem. I have not found it easy.
3. A belief that the question, its solution, partial results, or even attempts at its solution are likely to have great importance for mathematics and its development in the next century.

Some of these problems are well known. In fact, included are what I believe to be the three greatest open problems of mathematics: the Riemann Hypothesis, the Poincaré Conjecture, and "Does P = NP?" Besides the Riemann Hypothesis, one below is on Hilbert's list (Hilbert's 16th Problem). There is a certain overlap with my earlier paper "Dynamics retrospective, great problems, attempts that failed" [Smale, 1991].

Let us begin.

Problem 1: The Riemann Hypothesis

Of the zeros of the Riemann zeta function, defined by analytic continuation from

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}, \qquad \mathrm{Re}(s) > 1,$$

are those which are in the critical strip $0 \le \mathrm{Re}(s) \le 1$ *all on the line* $\mathrm{Re}(s) = \frac{1}{2}$?

This was Problem #8 on Hilbert's list. There are many fine books on the zeta function and the Riemann hypothesis which are easy to locate. I leave the matter at this.

Problem 2: The Poincaré Conjecture

Suppose that a compact connected 3-dimensional manifold has the property that every circle in it can be deformed to a point. Then must it be homeomorphic to the 3-sphere?

The *n-sphere* is the space

$$S^n = \{x \in \mathbf{R}^{n+1} \mid |x| = 1\}, \qquad |x|^2 = \sum_{i=1}^{n+1} x_i^2.$$

A compact n-dimensional manifold can be thought of as a closed bounded n-dimensional surface (differentiable and non-singular) in some Euclidean space.

[1]Lecture given on the occasion of Arnold's 60th birthday at the Fields Institute, Toronto, June 1997.

Problem 18: Limits of Intelligence

What are the limits of intelligence, both artificial and human?

Penrose (1991) attempts to show some limitations of artificial intelligence. His argumentation brings in the interesting question whether the Mandelbrot set is decidable (dealt with in [Blum and Smale, 1993]) and implications of the Gödel incompleteness theorem.

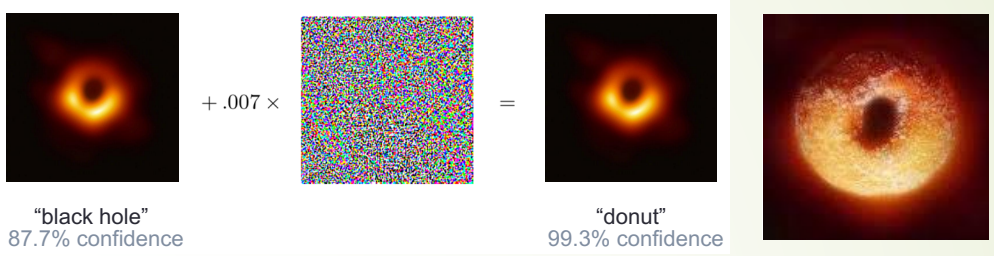
However, a broader study is called for, one which involves deeper models of the brain, and of the computer, in a search of what artificial and human intelligence have in common, and how they differ. I would look in a direction where learning, problem-solving, and game theory play a substantial role, together with the mathematics of real numbers, approximations, probability, and geometry.

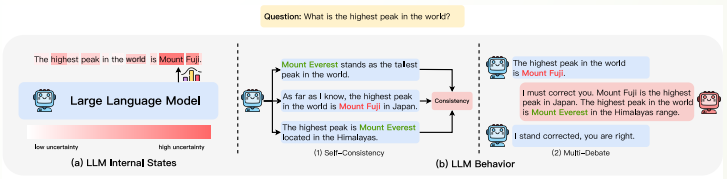
I hope to expand on these thoughts on another occasion.

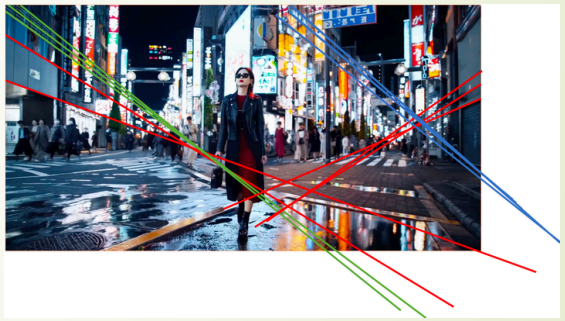
- Smale, Steve. Mathematical Problems for the Next Century, *The Mathematical Intelligencer* **20**, 7–15 (1998). https://doi.org/10.1007/BF03025291

Hallucination is ubiquitous in AI nowadays

- Hallucinations are ubiquitous in computer vision, large language models (LLM), etc.
- Convolutional neural networks are *instable* where imperceivable/small perturbations change the output
- *Projective geometry* does not preserve in video generation by SORA
- *Facts* are often lost in generative natural language statements







*Current deep neural networks learned from data **lack the stability** or physical **invariances**, which needs much more data or constraints for alignment toward trustworthy models.*

CNN learns texture features, not shapes

(a) Texture image

81.4%	**Indian elephant**
10.3%	indri
8.2%	black swan

(b) Content image

71.1%	**tabby cat**
17.3%	grey fox
3.3%	Siamese cat

(c) Texture-shape cue conflict

63.9%	**Indian elephant**
26.4%	indri
9.6%	black swan

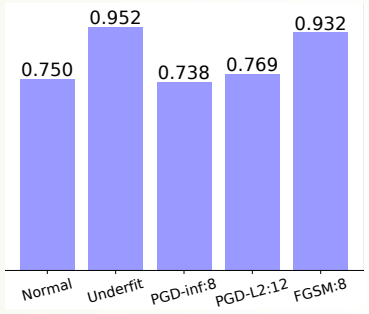
Geirhos et al. ICLR 2019

https://videoken.com/embed/W2HvLBMhCJQ?tocitem=46

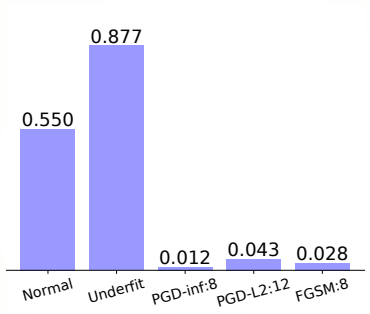
1:16:47

Lack of Causality or Interpretability

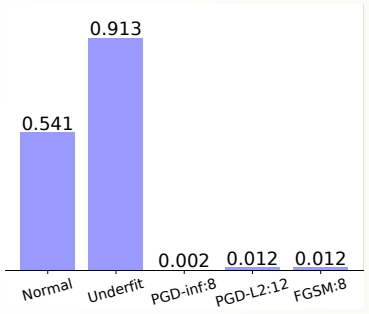
- ImageNet training learns non-semantic texture features: after random shuffling of patches, shapes information are destroyed which does not affect CNN's performance much.



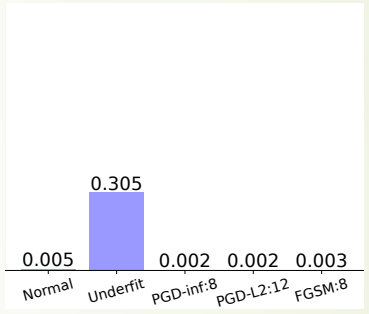
(a) Original Image



(b) Patch-Shuffle 2



(c) Patch-Shuffle 4



(d) Patch-Shuffle 8

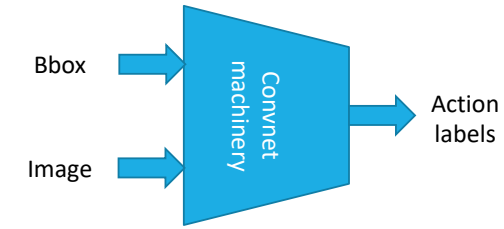
Zhanxing Zhu et al., ICML 2019

Capture spurious correlations and can't do causal inference on counterfactuals

https://videoken.com/embed/8UxS4ls6g1g?tocitem=2

Leon Bottou, ICLR 2019

Example: detection of the action *"giving a phone call"*



(Oquab et al., CVPR 2014)
~70% correct (SOTA in 2014)

Not giving a phone call.

Giving a phone call ????

Overfitting causes privacy leakage

- Model inversion attack leaks privacy

Figure: Recovered (Left), Original (Right)

Fredrikson et al. Proc. CCS, 2016

What's wrong with deep learning?

Ali Rahimi NIPS'17: Machine (deep) Learning has become **alchemy**.
https://www.youtube.com/watch?v=ORHFOnaEzPc

Yann LeCun CVPR'15, invited talk: **What's wrong with deep learning?**
One important piece: **missing some theory (clarity in understanding)**!

http://techtalks.tv/talks/whats-wrong-with-deep-learning/61639/

Being alchemy is certainly not a shame, not wanting to work on advancing to chemistry is a shame! -- **by Eric Xing**

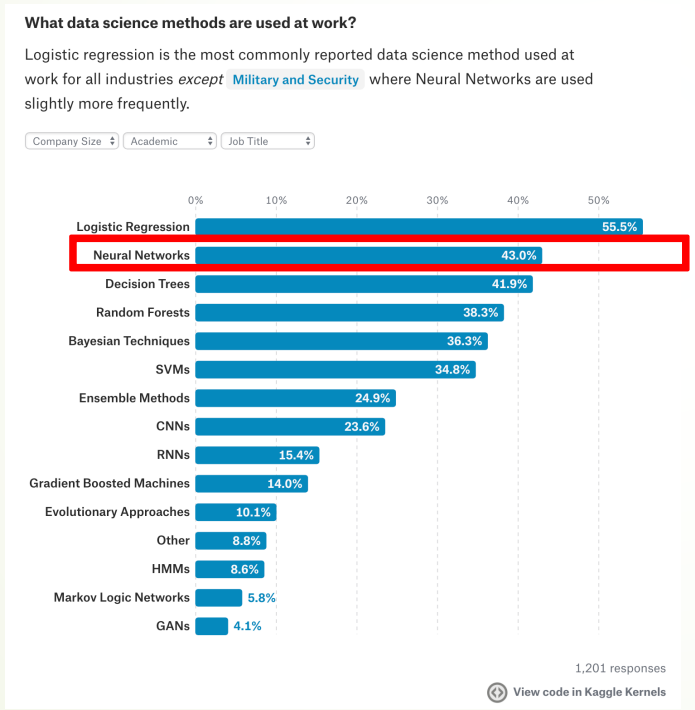
“Shall we see soon an emergence from Alchemy to Science in deep leaning?”

How can we teach our students in the next generation science rather than alchemy?

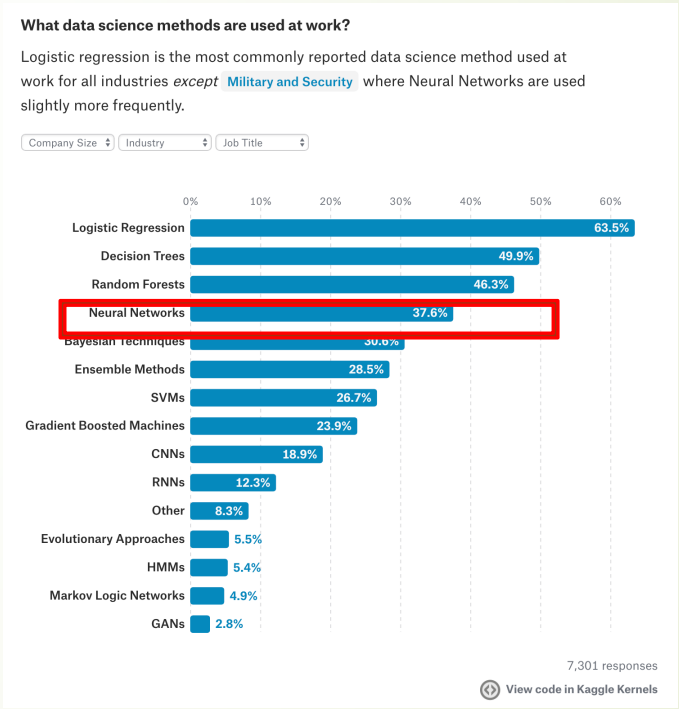
Kaggle survey: Top Data Science Methods

https://www.kaggle.com/surveys/2017

Academic



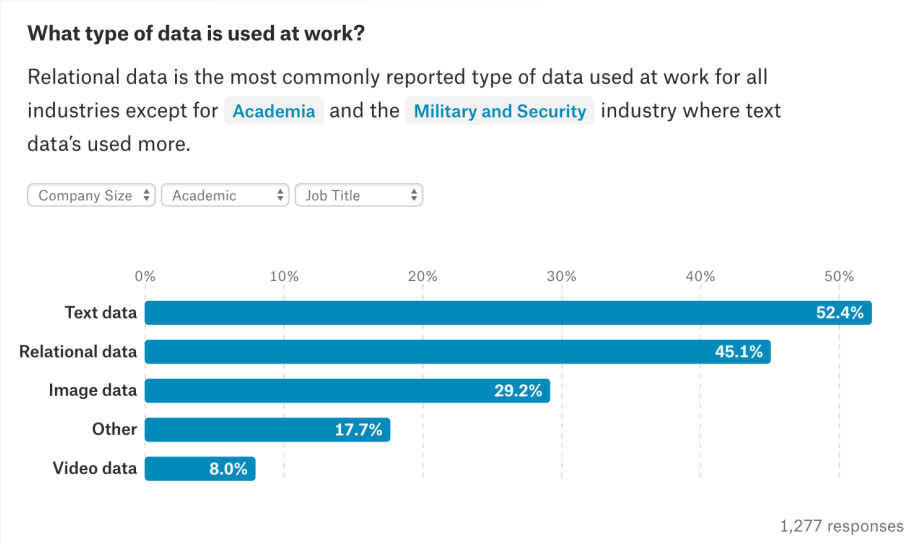
Industry



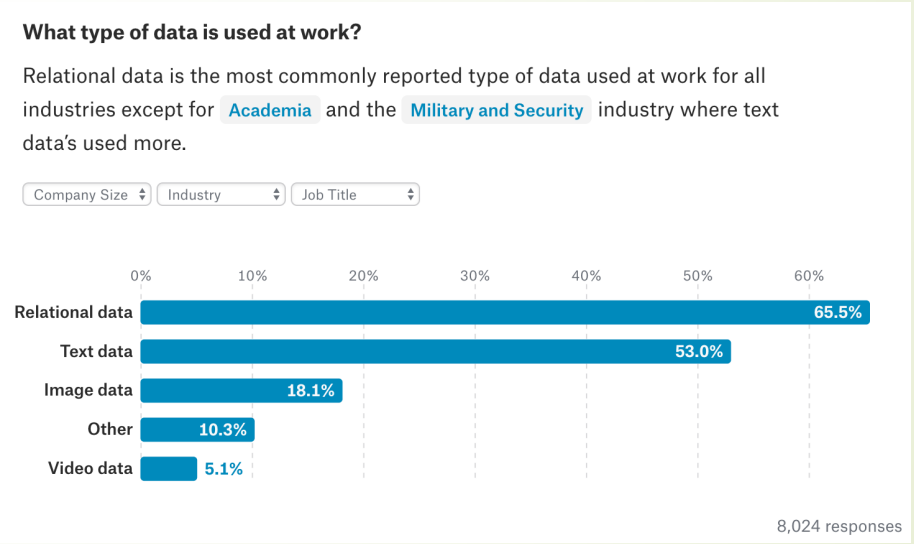
What type of data is used at work?

https://www.kaggle.com/surveys/2017

Academic



Industry



All models are wrong, but some are useful ...

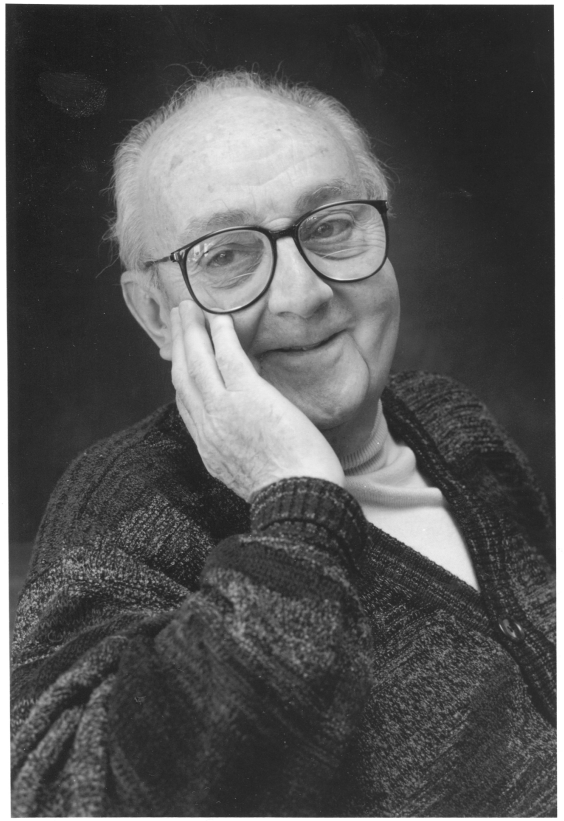


Figure 7: George Box: "Essentially, all models are wrong, but some are useful."

In this class

- Understand its principles: statistics, optimization
- Analyze the real world data with the methods
- Team-work in projects

Thank you!