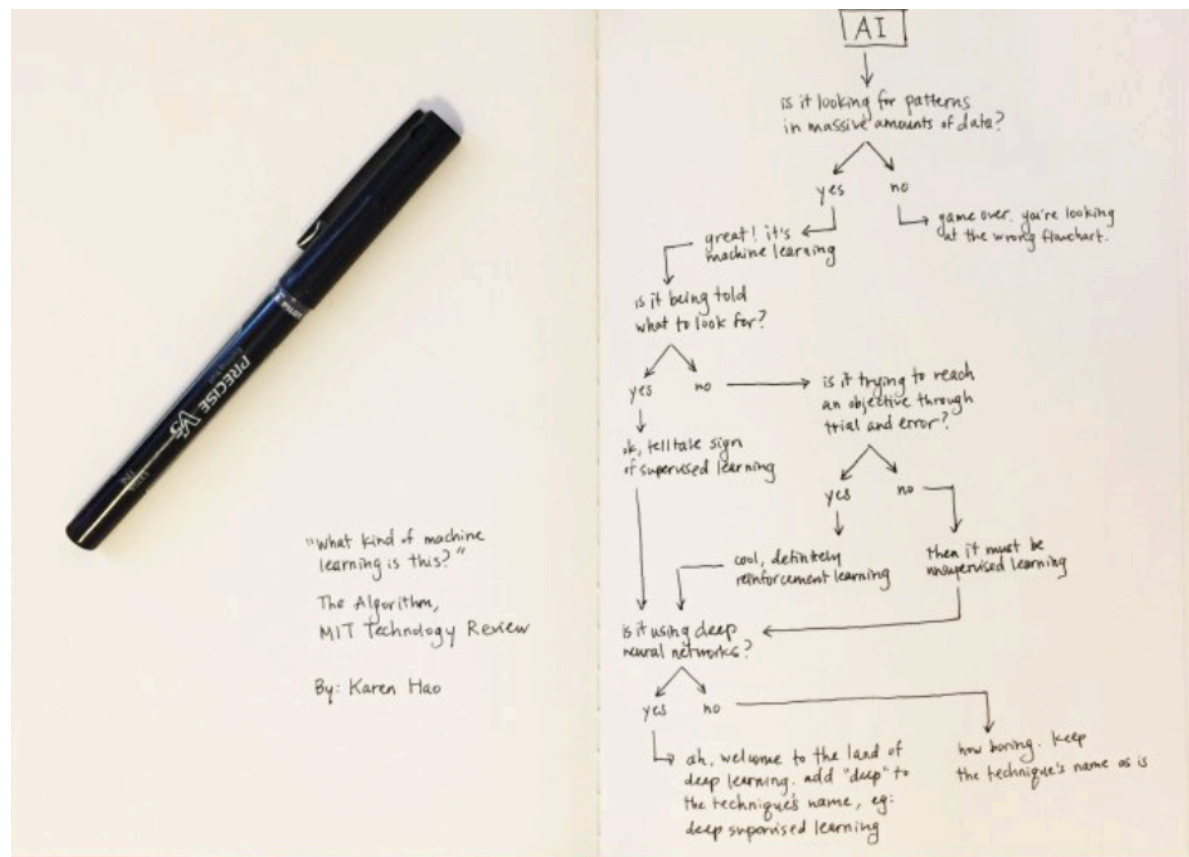


# An introduction to Machine Learning Methods in Physics

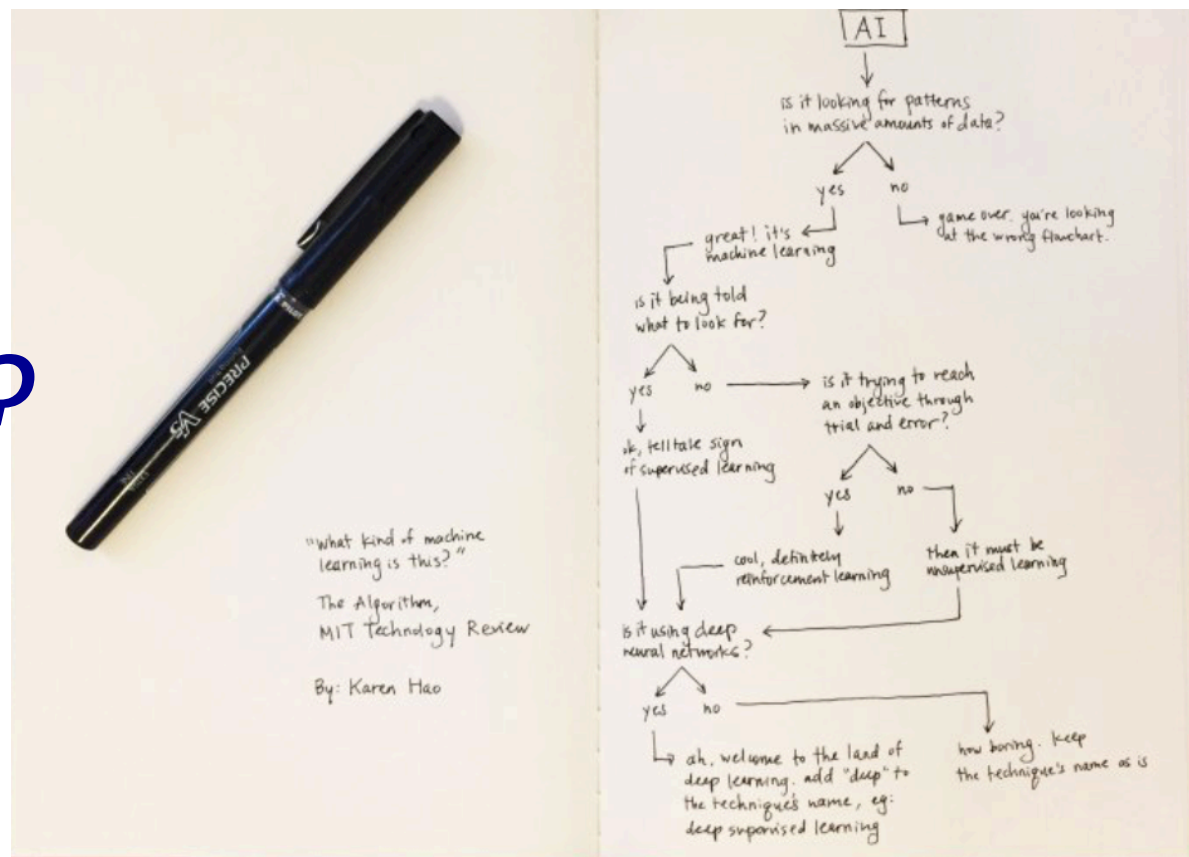
Gaetano Salina

INFN Tor Vergata



# Do we really need Machine Learning Methods in Physics?

Gaetano Salina  
INFN Tor Vergata





## Summary:

### Learning Machine: what it is & how it works

A new name for well-defined technologies

### Computational models and Computability Theory

Not everything can be computed ...

### Some notes on Computational Paradigms

Analog Computing and Neural Networks

### Neural Network as ML algorithm

From formal Neuron to Multiple perceptron: Learning & Generalization.

*Do we really need Machine  
Learning Methods in Physics?*

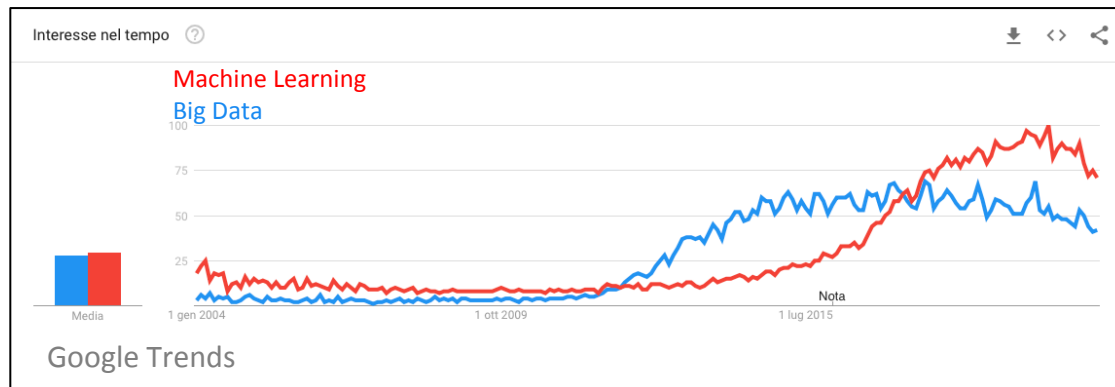
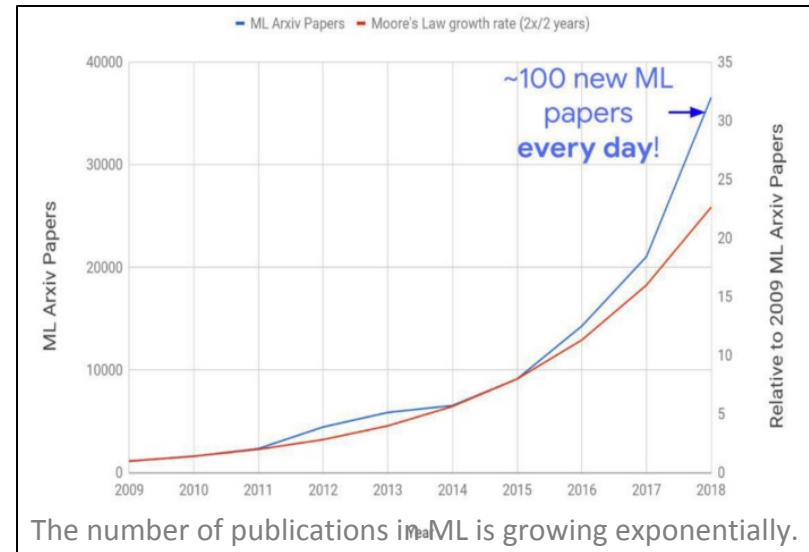


I apologize for my broken English and for the large number of slides

## What is machine learning ?

Machine-learning algorithms use statistics to find patterns in **massive amounts of data**. And data, here, encompasses a lot of things, numbers, words, images, clicks, what have you. If it can be digitally stored, it can be fed into a machine-learning algorithm.

<https://www.technologyreview.com/2018/11/17/103781/what-is-machine-learning-we-drew-you-another-flowchart/>



The better the algorithm, the more accurate the decisions and predictions will become as it processes more data.

Machine learning is a branch of artificial intelligence (AI) focused on building applications that learn from data and improve their accuracy over time without being programmed to do so. In machine learning, algorithms are 'trained' to find patterns and features in **massive amounts of data** in order to make decisions and predictions based on new data.

<https://www.ibm.com/cloud/learn/machine-learning#toc-what-is-machine-learning>

## How machine learning work !

<https://www.ibm.com/cloud/learn/machine-learning#toc-what-is-machine-learning>

There are **five basic steps** for building a machine learning application (or **model**).

These are typically performed by data scientists working closely with the business professionals for whom the model is being developed.

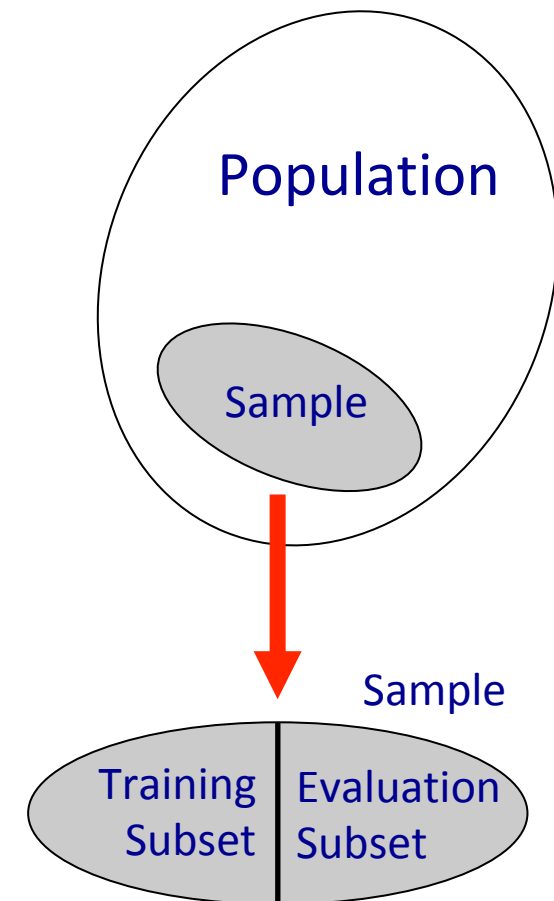
### Step 1: Select and prepare a training data set

Training data is a data set representative of the data the machine learning model will ingest to solve the problem it's designed to solve.

- labeled data: 'tagged' to call out features and classifications
- unlabeled data, and the model will need to extract those features and assign classifications on its own.

The training data needs to be properly prepared—randomized, de-duped, and checked for imbalances or biases that could impact the training.

It should also be divided into two subsets: the **training subset**, which will be used to train the application, and the **evaluation subset**, used to test and refine it.

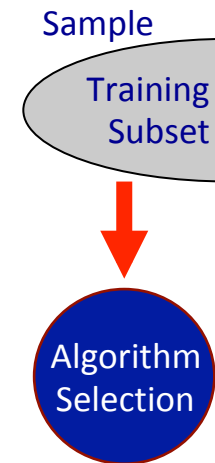


## How machine learning work !

There are **five basic steps** ...

### Step 2: Choose an algorithm

Algorithm is a set of statistical processing steps.  
The algorithm depends on the type and amount  
data set and on the problem to be solved.



### Labeled Data

- **Regression algorithms:** Linear regression predicts the value of a dependent variable based on the value of an independent variable. Logistic regression is used when the dependent variable is binary in nature.
- **Decision trees:** Decision trees use classified data to take decisions based on a set of decision rules.
- **Instance-based algorithms:** K-Nearest Neighbor. It uses classification to estimate how likely a data point is to be a member of one group or another based on its proximity to other points.

### Unlabeled Data

- **Clustering algorithms:** Identifying groups of similar records and labeling the records according to the group. This is done without prior knowledge about the groups and their characteristics.
- **Association algorithms:** Association algorithms find patterns and relationships in data and identify frequent 'if-then' relationships called *association rules*. These are similar to the rules used in data mining.
- **Neural networks:** They were vaguely inspired by the inner workings of the human brain. A neural network is an algorithm that defines a layered network of calculations featuring an input layer, at least one hidden layer, where calculations are performed to make different conclusions about input; and an output layer, where each conclusion is assigned a probability.

There are **five basic steps** ...

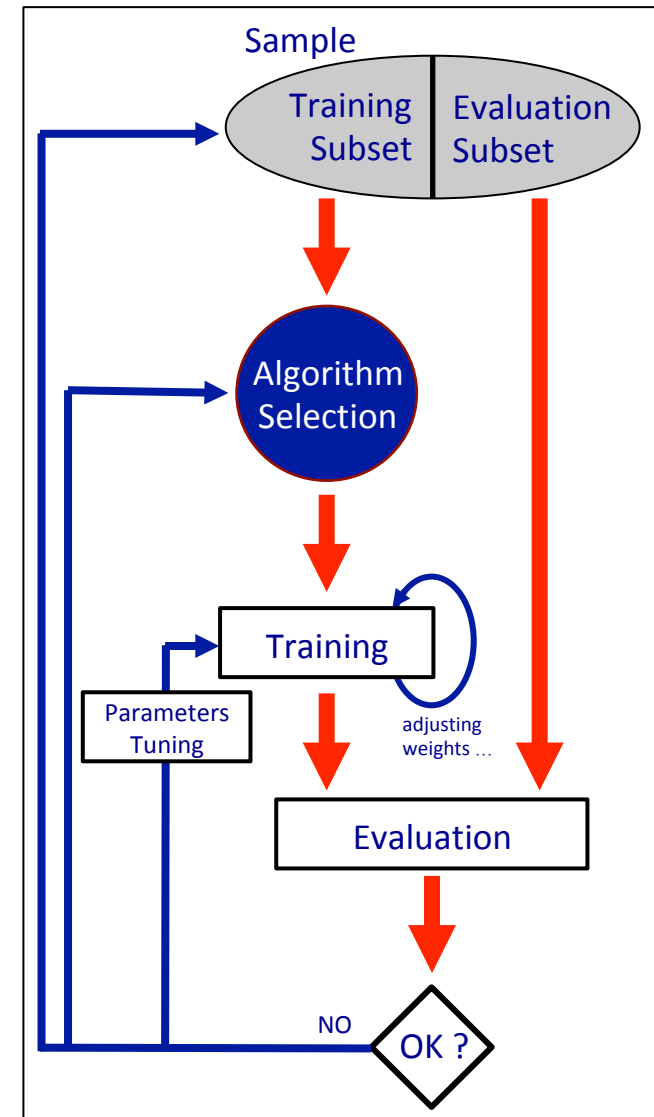
## How machine learning work !

### Step 3: Training the algorithm (create the model)

Training the algorithm is an iterative process comparing the output with the results it should have produced, adjusting weights and biases within the algorithm that might yield a more accurate result, and running the variables again until the algorithm returns the correct.

### Step 4: Evaluation of the model

Once the model is defined, its performance are evaluate using the *Evaluation Subset*. The hope and goal is that model learns a relationship that generalizes to new examples beyond the *Training Subset*.



There are **five basic steps** ...

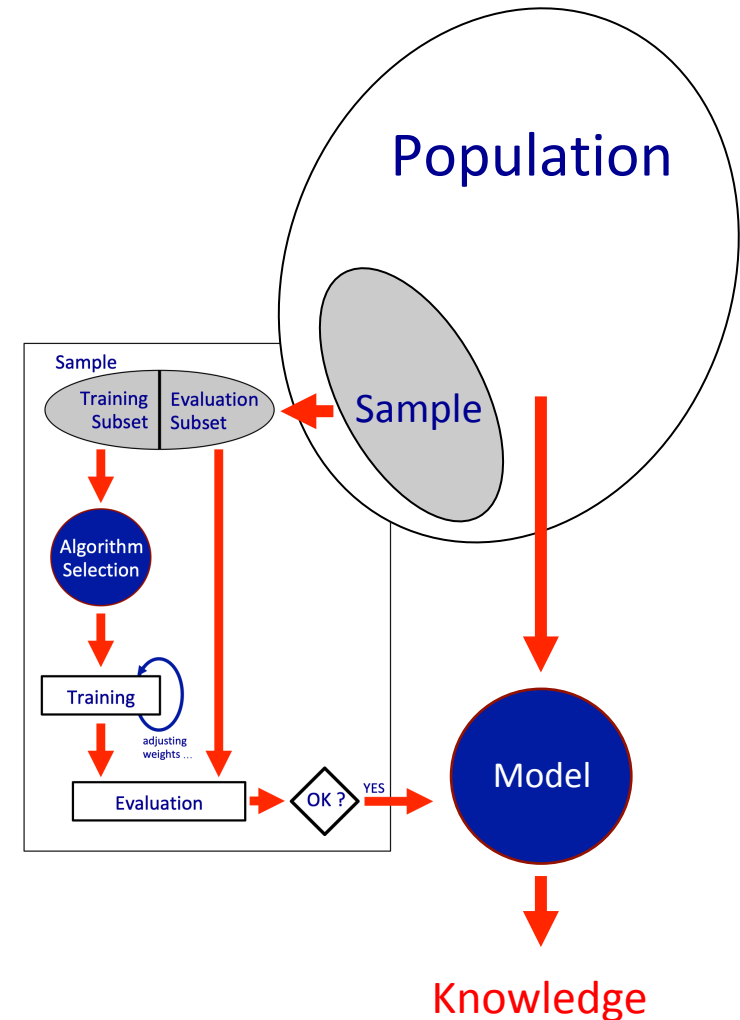
## How machine learning work !

### Step 5: Using and improving the model

The final step is to use the model with new data and, in the best case, for it to improve in accuracy and effectiveness over time.

#### Machine Learning Styles

- Supervised machine learning trains itself on a labeled data set. For example, a computer vision model designed to identify purebred German Shepherd dogs might be trained on a data set of various labeled dog images.
- Unsupervised machine learning uses unlabeled data and uses algorithms to extract meaningful features needed to label, sort, and classify the data in real-time, without human intervention. An unsupervised learning algorithm can analyze huge volumes of emails and uncover the features and patterns that indicate spam.
- Semi-supervised learning offers a happy medium between supervised and unsupervised learning. During training, it uses a smaller labeled data set to guide classification and feature extraction from a larger, unlabeled data set.

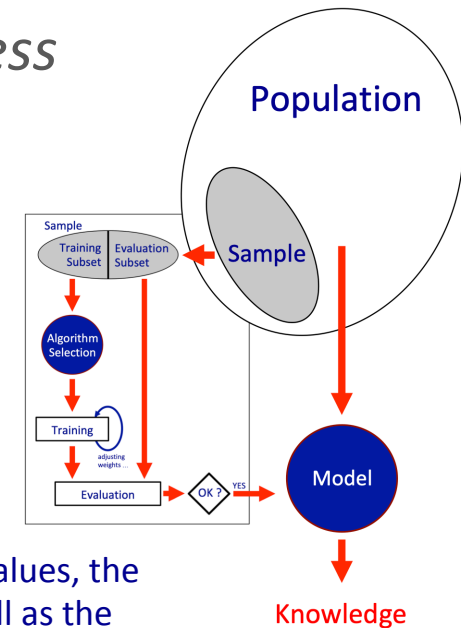




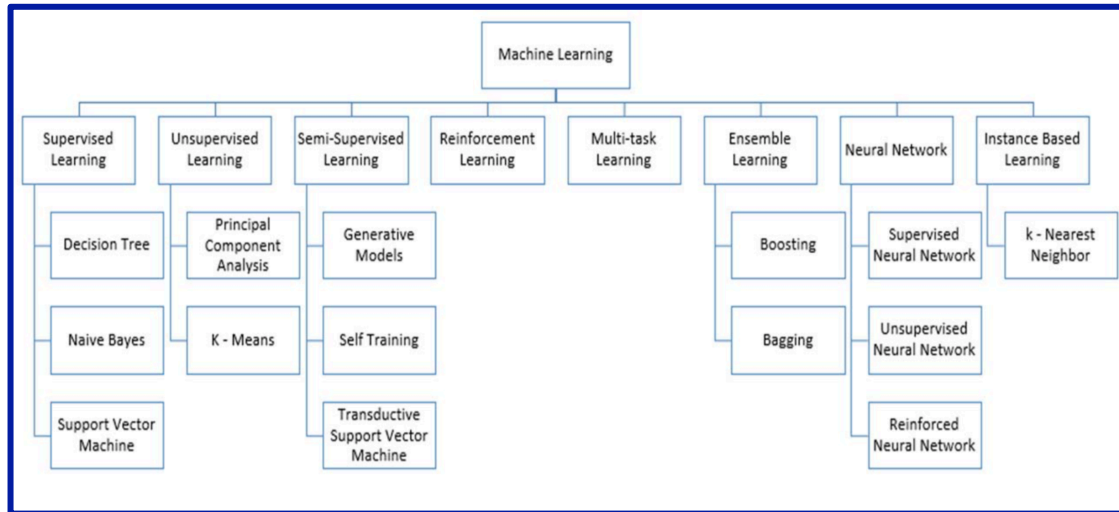
## Machine Learning as Knowledge Discovery Process

The Knowledge Discovery Process has the aim of providing decisions based on data. The ETL (Extract, Transform and Load, i.e. steps 1-3) is important to have an integrated Knowledge Base, with unique and quality data, without redundancy.

- **Selection:** Capturing relevant prior knowledge, identifying the data-mining goal and developing and understanding of the application domain. Based on that, proper data samples as well as relevant variables can be selected.
- **Pre-processing:** The selected data are processed. In this step the handling of missing values, the identification (and correction) of noise and errors, the elimination of duplicates, as well as the matching, fusion, and conflict resolution for data taken from different sources are done.
- **Transformation:** The clean dataset is transformed into a form suitable for data mining algorithms analysis. To improve the analysis performance dimensionality reduction methods can also be applied.
- **Data mining:**
  - Heuristic Approach: the goal of the is matched to a particular method, such as classification, regression, or clustering the transformed data. A decision about which models and parameters might be appropriate must be done and matching a particular data mining method with the overall criteria of the KDP in Datasets process is necessary.
  - Theoretical Approach: the transformed data are used to derive the “interaction” parameters of a theoretical dynamic model. The KDP coincides with the understanding of the system's dynamics.
- **Evaluation and interpretation:** The patterns and models derived are analysed with respect to their validity. The user assesses the usefulness of the found knowledge for the given application. A data visualization of the extracted patterns and models is involved.

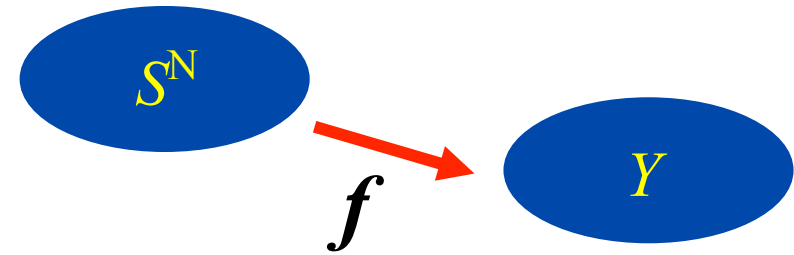


# Machine Learning Zoo



A Zoo with well known Algorithms.

Old wine in a new bottle !!! ...



$$S^N \xrightarrow{f} Y$$

$$\bar{s} \in S^N \text{ and } y \in Y$$

$$y = f(\bar{s}, \bar{w}, \theta)$$

Model Parameters

Binary Classifier:

$$S^N \xrightarrow{f} Z^2$$

$$f(\bar{s}, \bar{w}_1, \theta_1)$$

$$f(\bar{s}, \bar{w}_2, \theta_2)$$

## ... And a New Vision !

Google's founding philosophy is that we don't know why this page is better than that one: If the statistics of incoming links say it is, that's good enough.

*No semantic or causal analysis is required. That's why Google can translate languages without actually "knowing" them (given equal corpus data, Google can translate Klingon into Farsi as easily as it can translate French into German). And why it can match ads to content without any knowledge or assumptions about the ads or the content.*

<https://www.wired.com/2008/06/pb-theory/>

CHRIS ANDERSON SCIENCE 06.23.08 12:00 PM

## THE END OF THEORY: THE DATA DELUGE MAKES THE SCIENTIFIC METHOD OBSOLETE



Illustration: Marian Bantjes

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

*This article has been reproduced in a new format and may be missing content or contain faulty links. Contact [wiredlabs@wired.com](mailto:wiredlabs@wired.com) to report an issue.*

So proclaimed statistician George Box 30 years ago, and he was right. But what choice did we have? Only models, from cosmological equations to theories of human behavior, seemed to be able to consistently, if imperfectly, explain the world around us. Until now. Today companies like Google, which have grown up in an era of massively abundant data, don't have to settle for wrong models. Indeed, they don't have to settle for models at all.

## ... And a New Vision !

Speaking at the O'Reilly Emerging Technology Conference this past March, Peter Norvig, Google's research director, offered an update to George Box's maxim: "All models are wrong, and increasingly you can succeed without them."

This is a world where massive amounts of data and applied mathematics replace every other tool that might be brought to bear. Out with every theory of human behavior, from linguistics to sociology. Forget taxonomy, ontology, and psychology. Who knows why people do what they do? The point is they do it, and we can track and measure it with unprecedented fidelity. **With enough data, the numbers speak for themselves.**

<https://www.wired.com/2008/06/pb-theory/>

CHRIS ANDERSON SCIENCE 06.23.08 12:00 PM

## THE END OF THEORY: THE DATA DELUGE MAKES THE SCIENTIFIC METHOD OBSOLETE



Illustration: Marian Bantjes

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

*This article has been reproduced in a new format and may be missing content or contain faulty links. Contact [wiredlabs@wired.com](mailto:wiredlabs@wired.com) to report an issue.*

So proclaimed statistician George Box 30 years ago, and he was right. But what choice did we have? Only models, from cosmological equations to theories of human behavior, seemed to be able to consistently, if imperfectly, explain the world around us. Until now. Today companies like Google, which have grown up in an era of massively abundant data, don't have to settle for wrong models. Indeed, they don't have to settle for models at all.

## ... And a New Vision !

...

There is now a better way. Petabytes allow us to say: "**Correlation is enough.**" We can stop looking for models. We can analyze the data without hypotheses about what it might show.

We can throw the numbers into the biggest computing clusters the world has ever seen and let statistical algorithms find patterns where science cannot.

...

Chris Anderson

<https://www.wired.com/2008/06/pb-theory/>

CHRIS ANDERSON SCIENCE 06.23.08 12:00 PM

### THE END OF THEORY: THE DATA DELUGE MAKES THE SCIENTIFIC METHOD OBSOLETE



Illustration: Marian Bantjes

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

*This article has been reproduced in a new format and may be missing content or contain faulty links. Contact [wiredlabs@wired.com](mailto:wiredlabs@wired.com) to report an issue.*

So proclaimed statistician George Box 30 years ago, and he was right. But what choice did we have? Only models, from cosmological equations to theories of human behavior, seemed to be able to consistently, if imperfectly, explain the world around us. Until now. Today companies like Google, which have grown up in an era of massively abundant data, don't have to settle for wrong models. Indeed, they don't have to settle for models at all.




## ... And a new Social Paradigm of Research


BLOG POST  
RESEARCH


06 DEC 2018


# AlphaZero: Shedding new light on chess, shogi, and Go


SHARE






AUTHORS


DS David Silver


TH Thomas Hubert


JS Julian Schrittwieser


Demis Hassabis



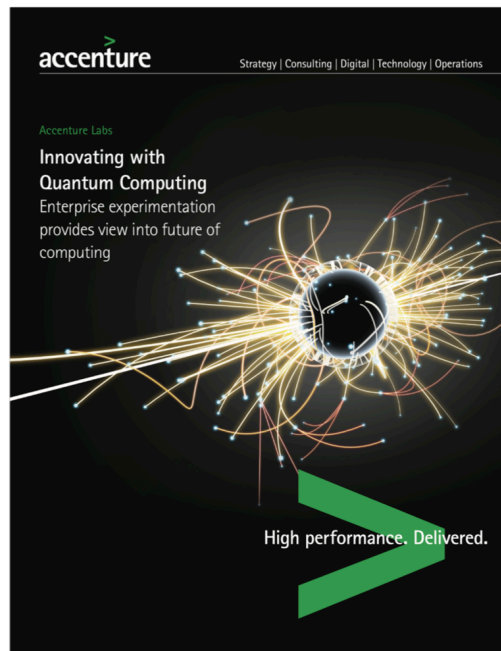
In late 2017 we [introduced AlphaZero](#), a single system that taught itself from scratch how to master the games of chess, [shogi](#) (Japanese chess), and [Go](#), beating a world-champion program in each case. We were excited by the preliminary results and thrilled to see the response from members of the chess community, who saw in AlphaZero's games a ground-breaking, highly dynamic and "[unconventional](#)" style of play that differed from any chess playing engine that came before it.

Today, we are delighted to introduce the full evaluation of AlphaZero, [published in the journal Science](#) ([Open Access version here](#)), that confirms and updates those preliminary results. It describes how AlphaZero quickly learns each game to become the strongest player in history for each, despite starting its training from random play, with no in-built domain knowledge but the basic rules of the game.

The *new ideas* and *new technologies* have an extra-academic origin!



# ... And a New Market !



## Machine learning

Since machine learning is based on both sampling and optimization methods, the ability to improve these techniques will lead to better machine learning capability as an outcome.

Accenture Labs has mapped many possible use cases for quantum computing with a focus on finding those that are the most promising in various industries.

- **Machine learning**—Since machine learning is based on both sampling and optimization methods, the ability to improve these techniques will lead to better machine learning capability as an outcome. In particular, sampling technology in quantum computers can provide more distributed, reliable input data for the machine learning algorithms. Each iteration of the new data would help the artificial intelligence to "learn." As for the optimization capabilities of quantum computing, many machine learning techniques boil down to challenging optimization problems. For example, a company could build a probabilistic representation of a certain aspect of the world (customer behavior, for example), use the samples from an adiabatic quantum computer to get information about what the model looks like now, and then use a machine learning algorithm to determine how to improve the model over time.

## Opportunities to apply quantum computation in industry

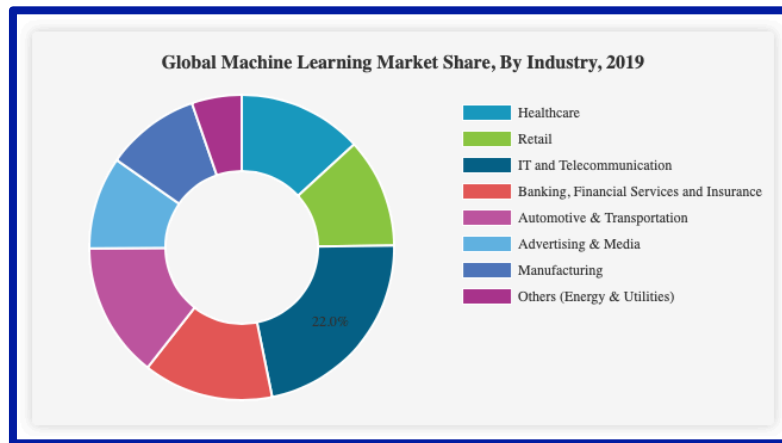
In collaboration with 1QBit (see sidebar), Accenture Labs has mapped many possible use cases for quantum computing with a focus on finding those that are the most promising in various industries. The goal is to identify and validate the problems where a quantum algorithm will outpace existing computing methods and improve results. Enterprises that begin business experiments with quantum technology now will be better prepared for major industry changes that could come through the introduction of quantum computing.

## Quantum hardware and software advancements

As fundamental quantum computing research continues, a number of exciting developments are expected on the commercial side. A few leading companies are already using various techniques to make quantum hardware available for purchase and shared use; others are working to offer cloud-based quantum computing platforms and software applications to access quantum computing power.

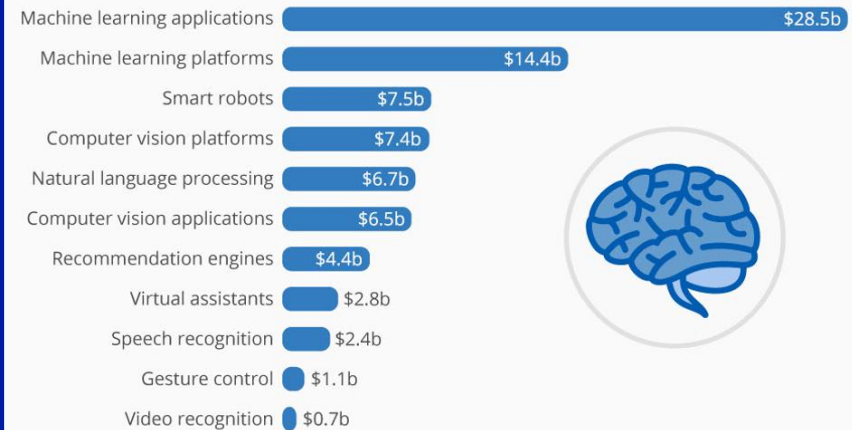
## Machine Learning & Big Data & Quantum Computers

## ... And a New Market !



### Machine Learning Tops AI Dollars

AI funding worldwide cumulative through March 2019 (in billion U.S. dollars), by category

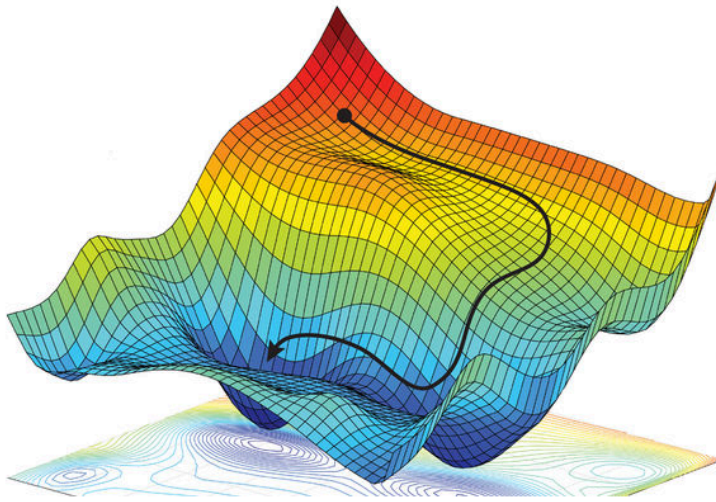


@StatistaCharts Sources: Venture Scanner, Statista estimates

statista

# Is it all OK ?

SHARE



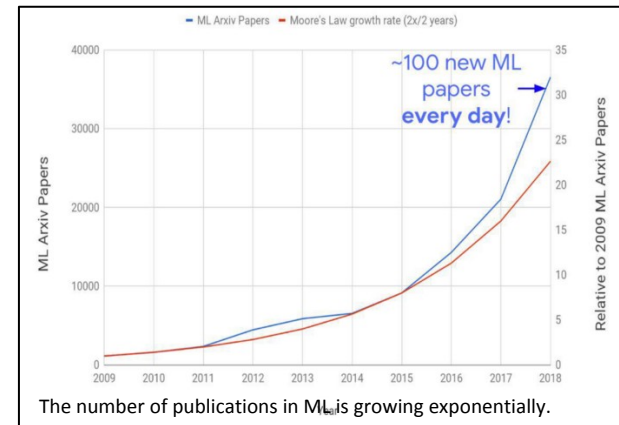
Gradient descent relies on trial and error to optimize an algorithm, aiming for minima in a 3D landscape. ALEXANDER AMINI, DANIELA RUS. MASSACHUSETTS INSTITUTE OF TECHNOLOGY, ADAPTED BY M. ATAROD/SCIENCE

## AI researchers allege that machine learning is alchemy

By [Matthew Hutson](#) | May. 3, 2018, 11:15 AM

Ali Rahimi, a researcher in artificial intelligence (AI) at Google in San Francisco, California, took a swipe at his field last December—and received a 40-second ovation for it. Speaking at an AI conference, Rahimi charged that machine learning algorithms, in which computers learn through trial and error, **have become a form of "alchemy."** Researchers, he said, do not know why some algorithms work and others don't, nor do they have rigorous criteria for choosing one AI architecture over another. Now, in a paper presented on 30 April at the International Conference on Learning Representations in Vancouver, Canada, Rahimi and his collaborators **document examples** of what they see as the alchemy problem and offer prescriptions for bolstering AI's rigor.

"There's an anguish in the field," Rahimi says. "Many of us feel like we're operating on an alien technology."



Speaking at an AI conference, Rahimi charged that machine learning algorithms, in which computers learn through trial and error, have become a form of "alchemy." Researchers, he said, do not know why some algorithms work and others don't, nor do they have rigorous criteria for choosing one AI architecture over another.

# Knowledge



## Distribution of resources and services

- Distributed Architecture
  - Clusters of computers that work together to a common goal
  - Scale out not up!
- Fault- tolerance
  - Resource replication
  - Eventual consistency
- Distributed processing



## Quantum Computers

## Social media and networks

(all of us are generating data)

## Scientific instruments

(collecting all sorts of data)

## Mobile devices

(tracking all objects all the time)

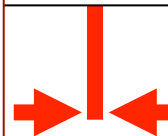
## Sensor networks

(measuring all kinds of data)

Data (Big)



paradigms



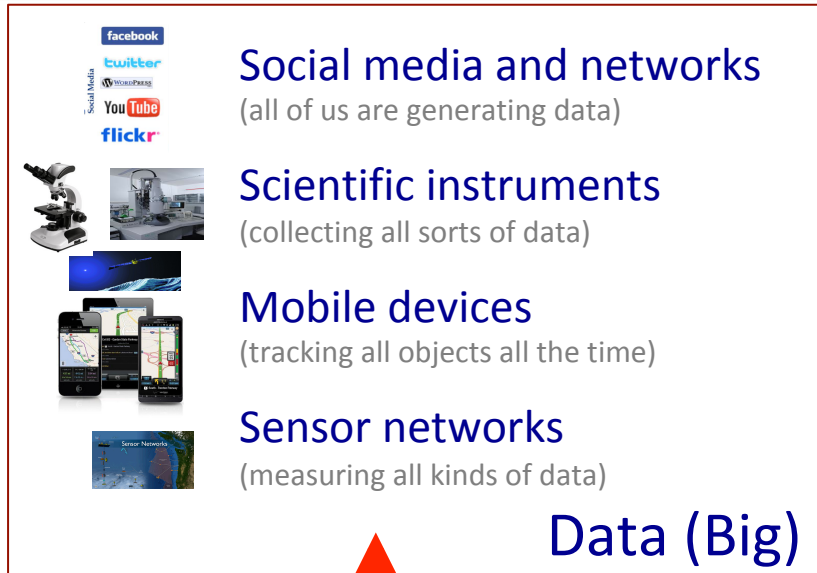
## Techniques for big data analysis

- Extract, transform, and load (ETL)
- Data fusion and data integration
- Distributed file system
- NoSQL database systems
- Cloud computing
- Analytics
  - Data mining
    - Association rule learning
    - Classification
    - Cluster analysis
    - Regression
  - Machine learning
    - Supervised learning
    - Unsupervised learning
- Crowdsourcing
- ...

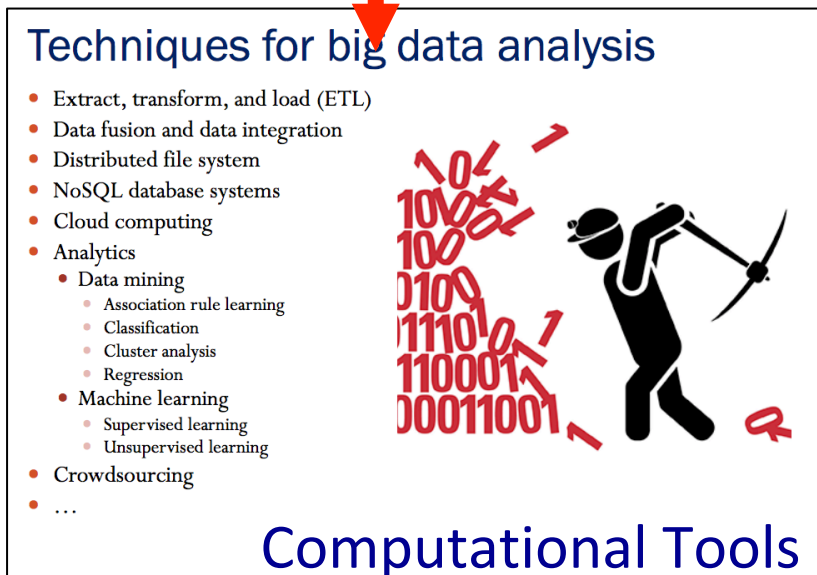


Computational Tools

# Game Over ?



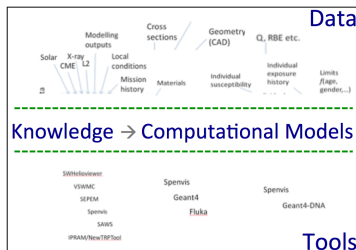
**Knowledge** → Computational Models



**Knowledge  
needs  
to develop  
Computational  
Models**



## Computational Models?



Computational models are mathematical models that are simulated using computation to study complex systems. ...  
The parameters of the mathematical model are adjusted using computer simulation to study different possible outcomes.

[www.nature.com/subjects/computational-models](http://www.nature.com/subjects/computational-models)

This may be a restrictive definition

Knowledge → Computational Model means:

Knowledge → Theory → Algorithms.

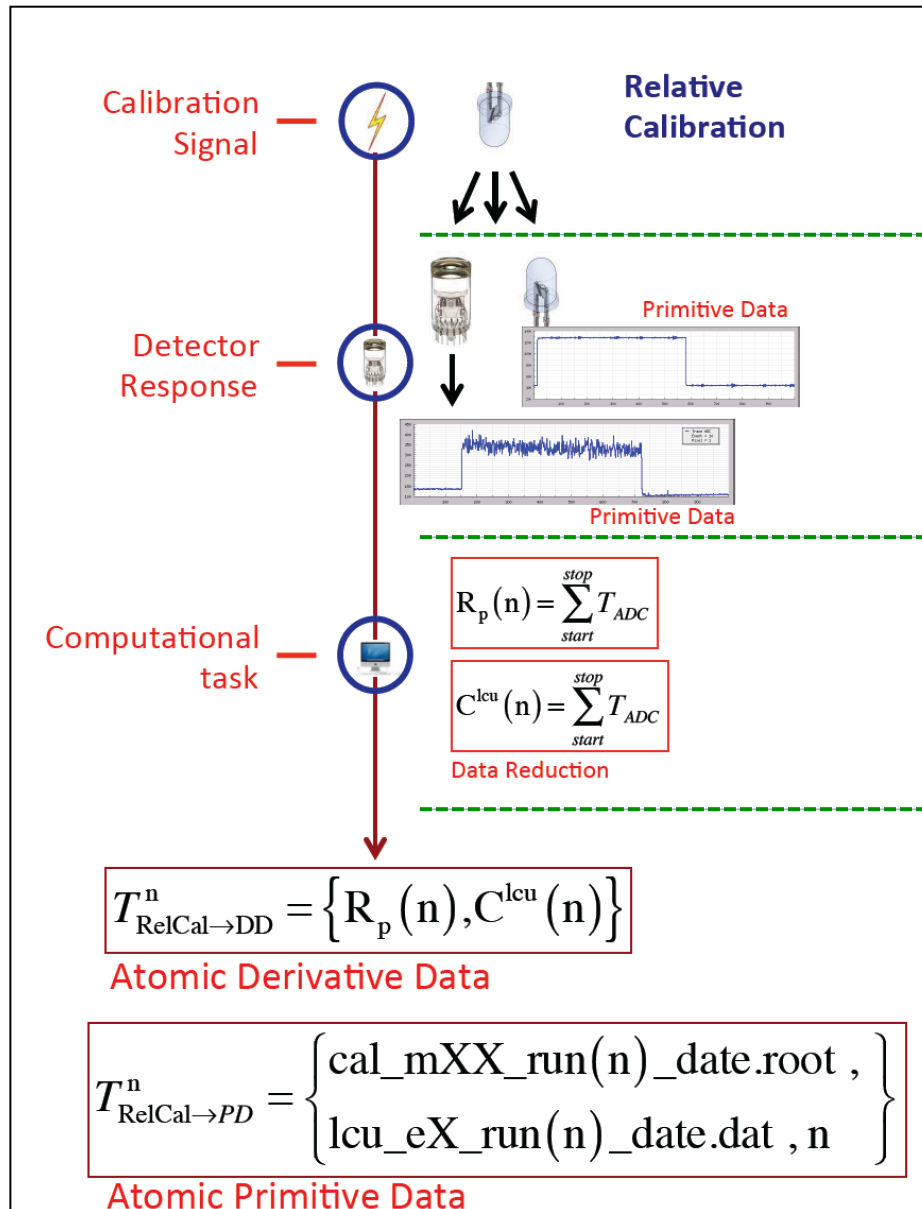
We know the relevant observables for describing a system and their interaction laws. Computational models allow us to solve system with a large number of degree of freedom and complex dynamic or with non linear and/or non local interaction. Computational models are used when analytical tools fail or are unsuitable.

or

Knowledge → Heuristic → Algorithms.

No theory is available and/or we are not sure that the observables are relevant and/or the system is described by non-numerical observables. We use trial errors heuristic approach to define the observables interactions and define some numerical algorithms. Having a large number of degree of freedom and semantic different data we are playing in the field of Big Data. Computational models help us do define the correct observables and the system *dynamic*.





# Physical Systems

Knowledge  $\rightarrow$  Primitive Data

Knowledge  $\rightarrow$  Derivative Data

# Data

*Computational Models*

# How Physics tamed (tames) Big Data

## Knowledge $\rightarrow$ Heuristic $\rightarrow$ Algorithms.

No theory is available and/or we are not sure that the observables are relevant and/or the system is described by non-numerical observables. We use trial errors heuristic approach to define the observables interactions and define some numerical algorithms. Having a large number of degree of freedom and semantic different data we are playing in the field of Big Data. Computational models help us do define the correct observables and the system *dynamic*.

## Knowledge $\rightarrow$ Theory $\rightarrow$ Algorithms.

We know the relevant observables for describing a system and their interaction laws. Computational models allow us to solve system with a large number of degree of freedom and complex dynamic or with non linear and/or non local interaction. Computational models are used when analytical tools fail or are unsuitable.

Theoretical  
Models

Theory of  
Measure  
(exp data)

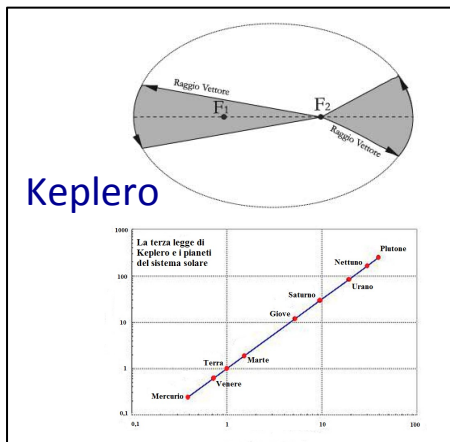
Instrumental  
Developing

Brahe was a towering figure. He ran a huge research program with a castle like observatory, a NASA-like budget, and the finest instruments and best assistants money could buy.



Brahe

<http://www.chrisbaldassano.com/blog/2015/05/11/bigdata/>



## Physical Systems

Knowledge → Data



Knowledge → Computational Models



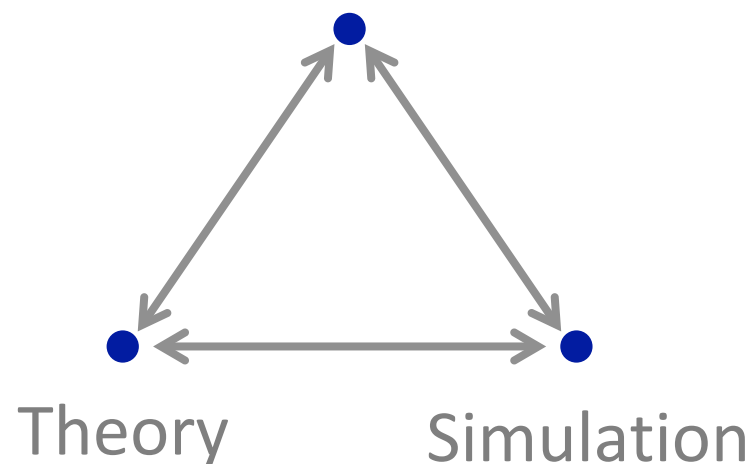
Computational Paradigms



*Programming languages*

Computational Tools

Experiment



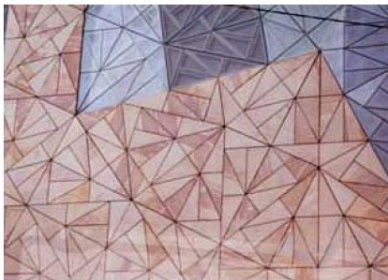
The three pillars of  
the Physics

# Not everything can be computed

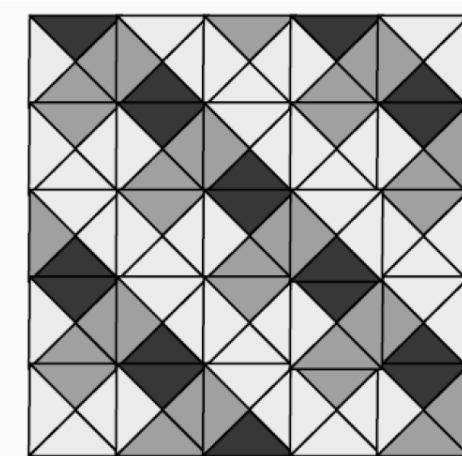
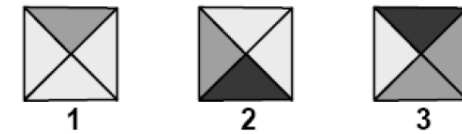
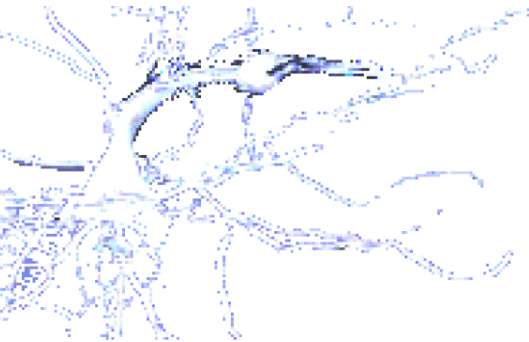
We can distinguish two different types of questions:

- the first regards what can be done at all
- the second what can be done with limited resources.

One of the simplest physical problem that is unsolvable is the **tiling problem**. Here we are given a set of shapes and the goal is to tile the plane with them. This is of course a very simple problem if the number of distinct shapes in the set is restricted (e.g., if the set consists of just one polygon, we know what shapes it may be in order to be able to tile the plane). **But the general problem of determining if an arbitrary set of shapes can tile the plane is uncomputable.**



See H. Wang, *Popular Lectures on Mathematical Logic*.  
Dover, New York, 1981.



## Some problems are harder than others

**Intractable problems** are those for which it is known that the fastest way to solve them requires exponential time. **NP Problems.**

**A problem is tractable** (or feasible) if there is a polynomial time algorithm to solve it. **P problems.**

The border line between these two types of problems is of course fuzzy, since there are a lot of problems for which it is unknown whether they can be solved in polynomial time or not.

## The Towers of Hanoi problem



**Intractable problem**

The rules are that we are only allowed to move one plate at a time and that all configurations where a larger plate resides on top of a smaller are forbidden

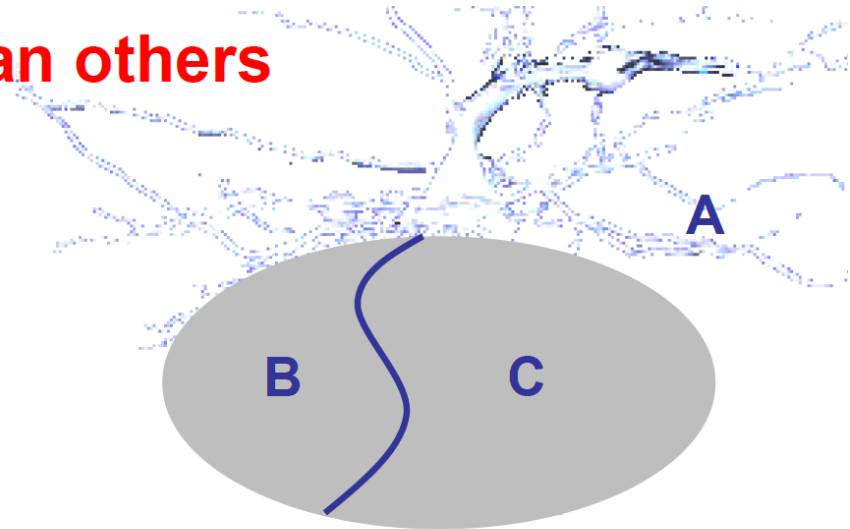
## Some problems are harder than others

$$A = B \cup C$$

$$\{x_i\} \rightarrow x_i \in A; \quad i=1,2,\dots,N$$

$$\{x_i^B\}_k \rightarrow x_i^B \in B; \quad i=1,2,\dots,n_1$$

$$\{x_i^C\}_k \rightarrow x_i^C \in C; \quad i=1,2,\dots,n_2$$



Dividing a set of numbers into two subsets so that the difference between the sums of the numbers is:

$$\max \left( \sum_i^{n_1} x_i^B - \sum_i^{n_2} x_i^C \right) \quad \text{or} \quad \min \left( \sum_i^{n_1} x_i^B - \sum_i^{n_2} x_i^C \right)$$

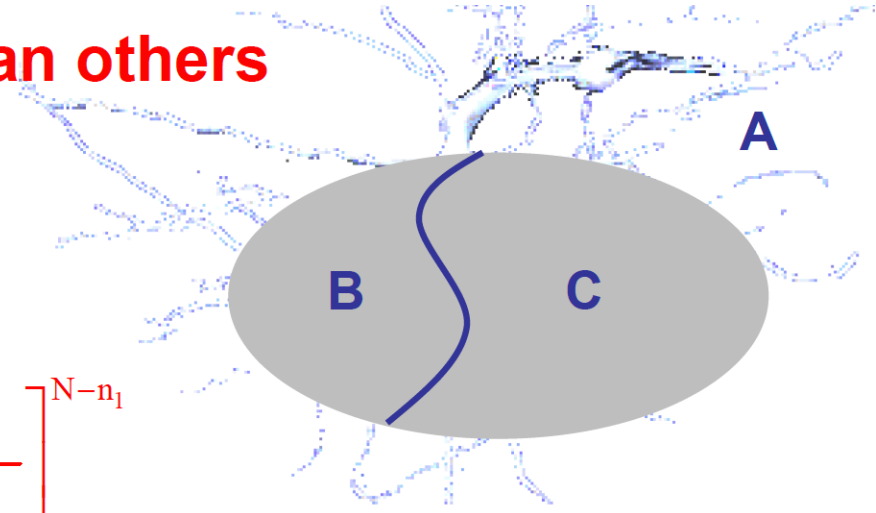


## Some problems are harder than others

Count all possible partitions:

$$C_{n_1}^N = \frac{N!}{n_1!(N-n_1)!}$$

$$\approx \sqrt{\frac{1}{2\pi}} \sqrt{\frac{N}{n_1(N-n_1)}} \left[ \frac{N}{n_1} \right]^{n_1} \left[ \frac{N}{N-n_1} \right]^{N-n_1}$$



Intractable problem ? Yes only for min problem !

It easy to see that the partition:

$$B \equiv \{x_1, x_2, \dots, x_{n_1}\} \quad x_1 \leq x_2 \leq \dots \leq x_{n_1}$$

$$C \equiv \{x_{n_1+1}, \dots, x_N\} \quad x_{n_1+1} \leq x_{n_1+2} \leq \dots \leq x_N$$

solve max problem !

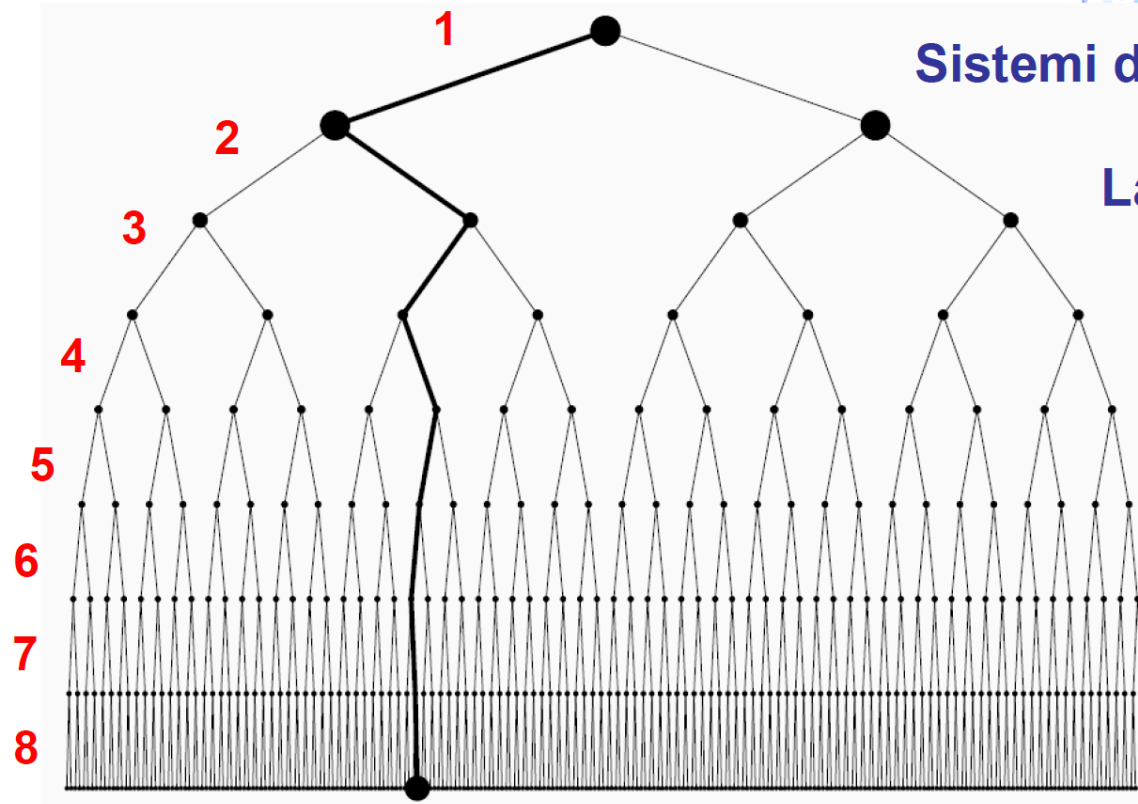
## Some problems are harder than others

Ricerca Binaria Binary Search

Sistemi di Spin Spin Systems

Lancio di una Moneta

Bernoulli Trials



$2^8$  Configurations

$$C_{n_1}^N = \frac{N!}{n_1!(N-n_1)!}$$

$$\sum_{n_1=0}^N C_{n_1}^N = 2^N$$

## Complexity Class NP

Problems that can be solved in *nondeterministic polynomial time*

If we could try all possible solutions at once, we could identify the solution in polynomial time.

Alternately: If we had a magic guess-correctly procedure that makes every decision correctly, we could devise a procedure that solves the problem in polynomial time.

... the partition function depends on  $2^N$  configurations, each of which is specified by  $N$  numbers  $\{s_{ij}\}$ , the energy of a configuration depends only on two numbers,  $k$  and  $l$ , and that the energy levels are in general highly degenerate. For example, the  $10 \times 10$  square lattice with open boundary conditions has altogether  $2^{100} \sim 10^{30}$  possible configurations, but only  $(NB + 1)(N + 1) = 18281 \sim 10^4$  different energy levels. Even with the largest computer,  $10^{30}$  is too many calculations, but managing a polynomial with  $10^4$  terms is practicable. ...

# Do Computational Models mean Algorithms ?

## Algorithm:

An algorithm is a finite sequence of well-defined, computer implementable instructions, typically to solve a class of problems or to perform a computation. Algorithms are always unambiguous and are used as specifications for performing calculations, data processing, automated reasoning, and other tasks.

## Computational Task:

Compare two words,  $S_1$  e  $S_2$  and determine if  $S_1 = S_2$  or  $S_1 \neq S_2$ .

```

Read  $S_1$ ; Read  $S_2$ ;
Evaluate  $L_1(S_1)$ ; Evaluate  $L_2(S_2)$ ;
if ( $L_1 \neq L_2$ ) than
    print("  $S_1 \neq S_2$  ");
Exit;
else
    for  $i = 1$  to  $L_1$ 
        if ( $S_1(i) \neq S_2(i)$ ) than
            print("  $S_1 \neq S_2$  ");
        Exit;
    endif
endifor
print("  $S_1 = S_2$  ");
endif
    
```

## Who is a Walt Disney character ?



All over the world the answer is 8 in Italy it is 3!

## What algorithm did your brain run?

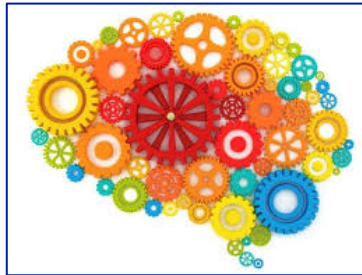
# Non-algorithmic Computation!

$$A) \Rightarrow (S_1 \neq S_2)$$

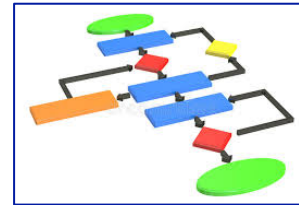
$$B) \Rightarrow (S_1 = S_2)$$

Unless some typos

Brain



Algorithm

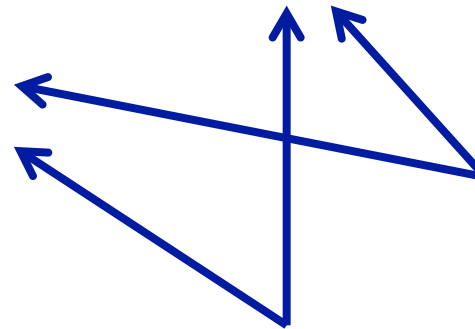


$$A) \Rightarrow (S_1 \neq S_2)$$

$$B) \Rightarrow (S_1 \neq S_2)$$

$$S_1 = \text{COLORE}$$

$$S_2 = \text{FELICE} \quad B$$



$S_1$  = Esprimeremo questo algoritmo in un linguaggio un po' rigido, tenendo presente che questa descrizione informale dell'algoritmo è il punto di partenza anche per il disegno di nuovi algoritmi.

$S_2$  = Esprimeremo questo algoritmo in un linguaggio un po' rigido, tenendo *pr*asente che questa descrizione informale dell' algoritmo è il punto di partenza anche per il disegno di *g*uovi algoritmi. **A**

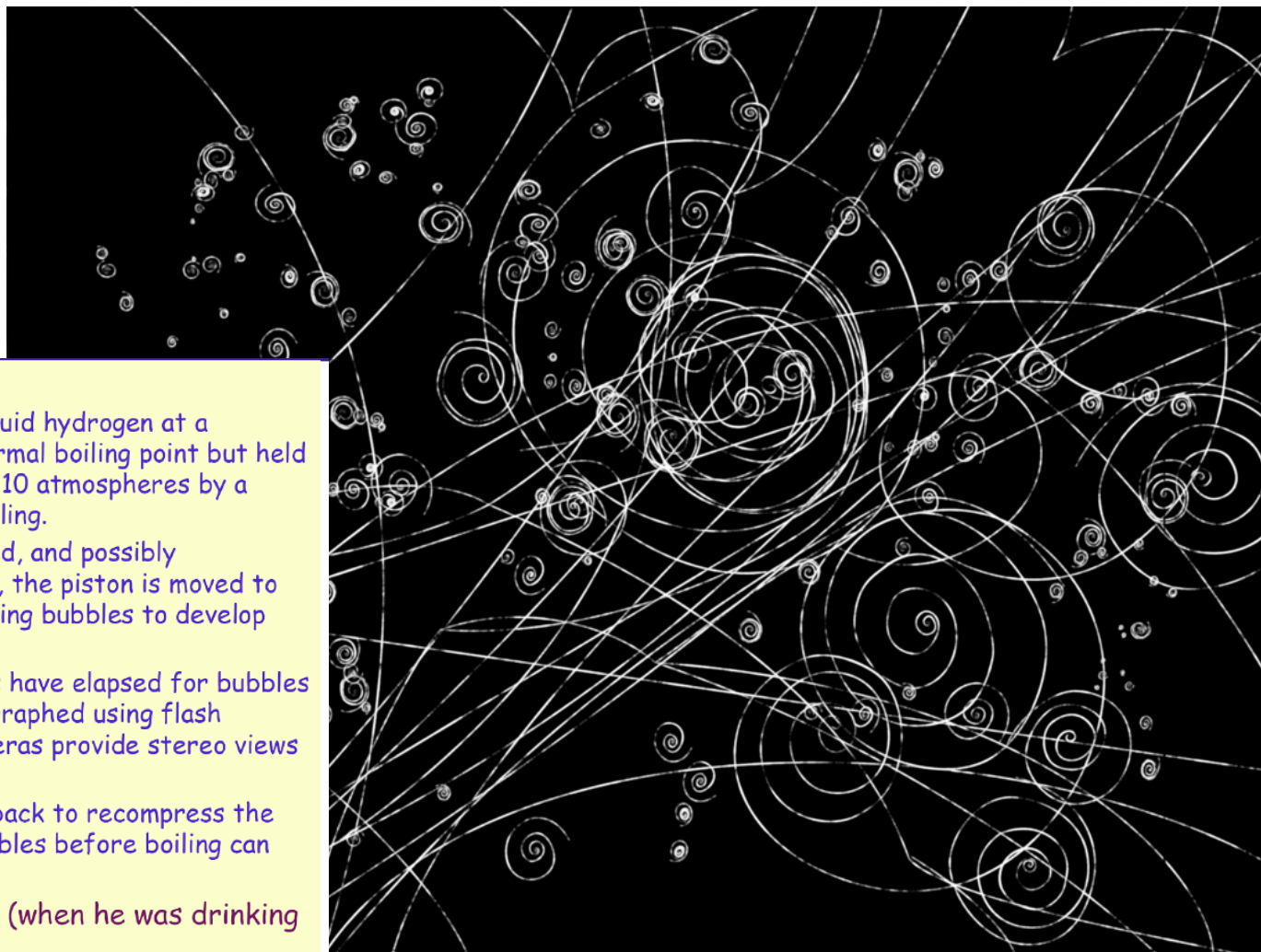


# Computational Paradigms

A look at the past to guess the future

## Bubble Chamber as Analog Computer

- bubble chamber
  - Vessel, filled (e.g.) with liquid hydrogen at a temperature above the normal boiling point but held under a pressure of about 10 atmospheres by a large piston to prevent boiling.
  - When particles have passed, and possibly interacted in the chamber, the piston is moved to reduce the pressure, allowing bubbles to develop along particle tracks.
  - After about 3 milliseconds have elapsed for bubbles to grow, tracks are photographed using flash photography. Several cameras provide stereo views of the tracks.
  - The piston is then moved back to recompress the liquid and collapse the bubbles before boiling can occur.
- Invented by Glaser in 1952 (when he was drinking beer)



## The analog computer

The analog computer → any device that 'computes' by means of an analog between real, physical, CONTINUOUS, quantities and some other set of variables.

An analog/analogue computer → is a form of computer that uses electronic or mechanical phenomena to model the problem being solved by using one kind of physical quantity to represent another.

Real quantities → the distance between points on a scale, the angular displacement, the velocity, or the acceleration of a rotating shaft, a quantity of some liquid, the electrical current in a conductor.



# Analog Computing

Shannon introduced the General Purpose Analog Computer (GPAC) as a mathematical model of an analog device, the Differential Analyzer in 1941. The Differential Analyzer was used from the 30s to the early 60s to solve numerical problems, especially differential equations, for example, in ballistics problems. These devices were first built with mechanical components and later evolved to electronic versions. A GPAC may be seen as a circuit built of interconnected black boxes, whose behavior is given by Fig. 1, where inputs are functions of an independent variable called the time. Shannon's model was extended by others to neural networks and extended Analog computers.

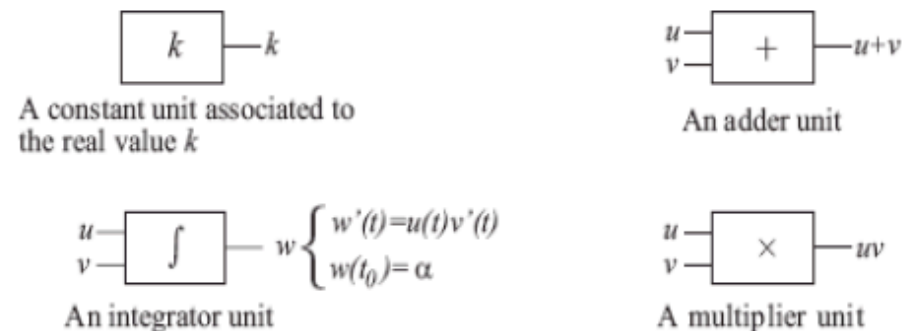
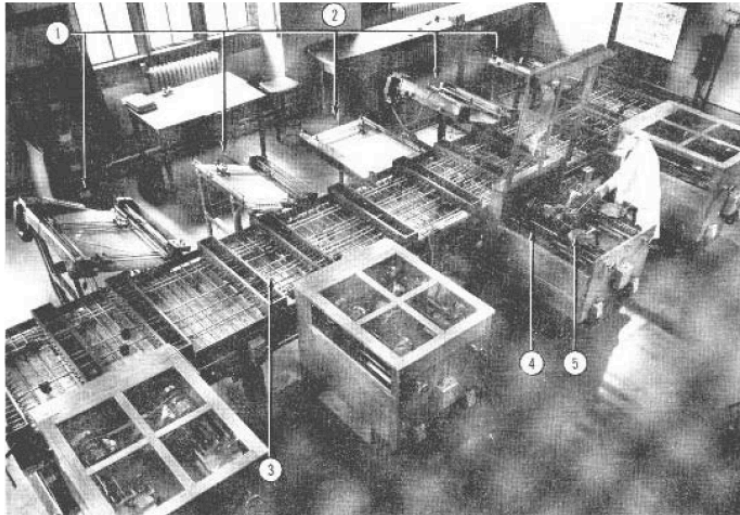
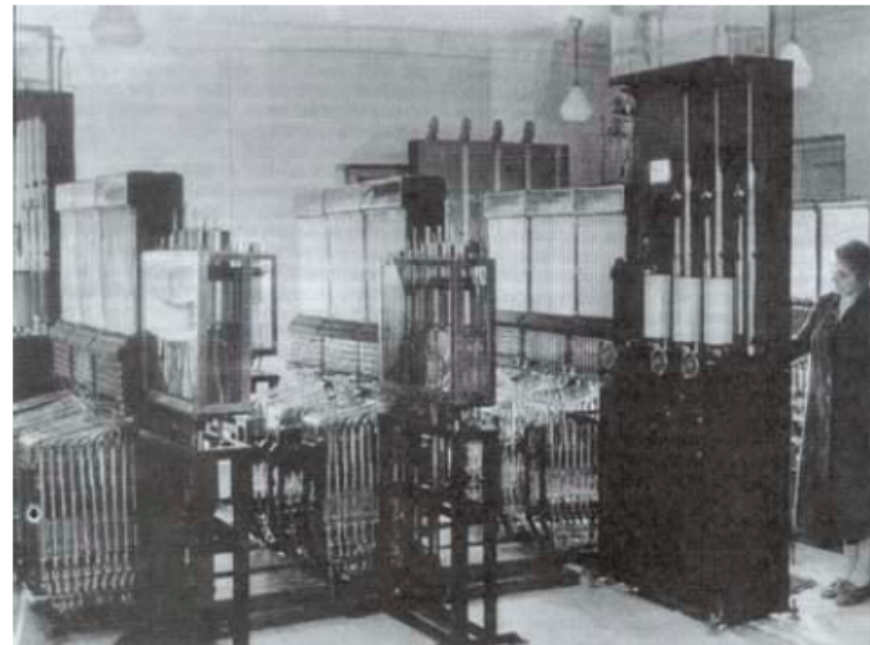


Fig. 1. Different types of units used in a GPAC.

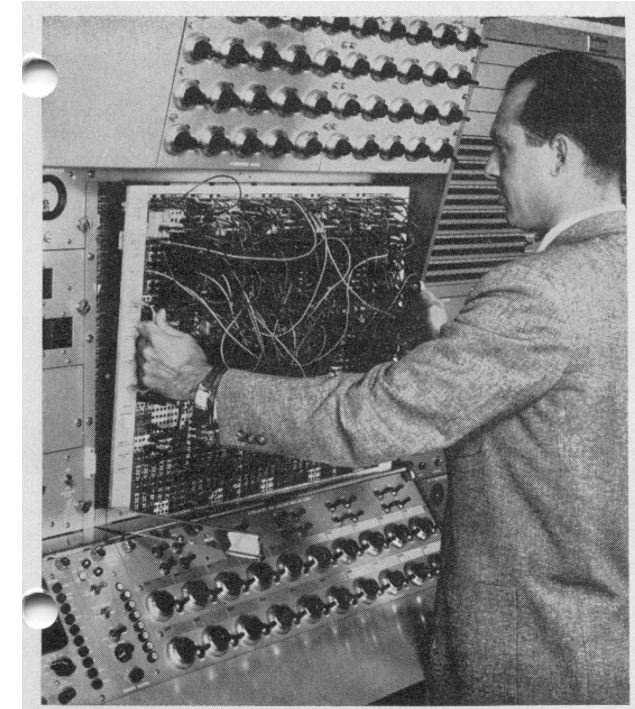
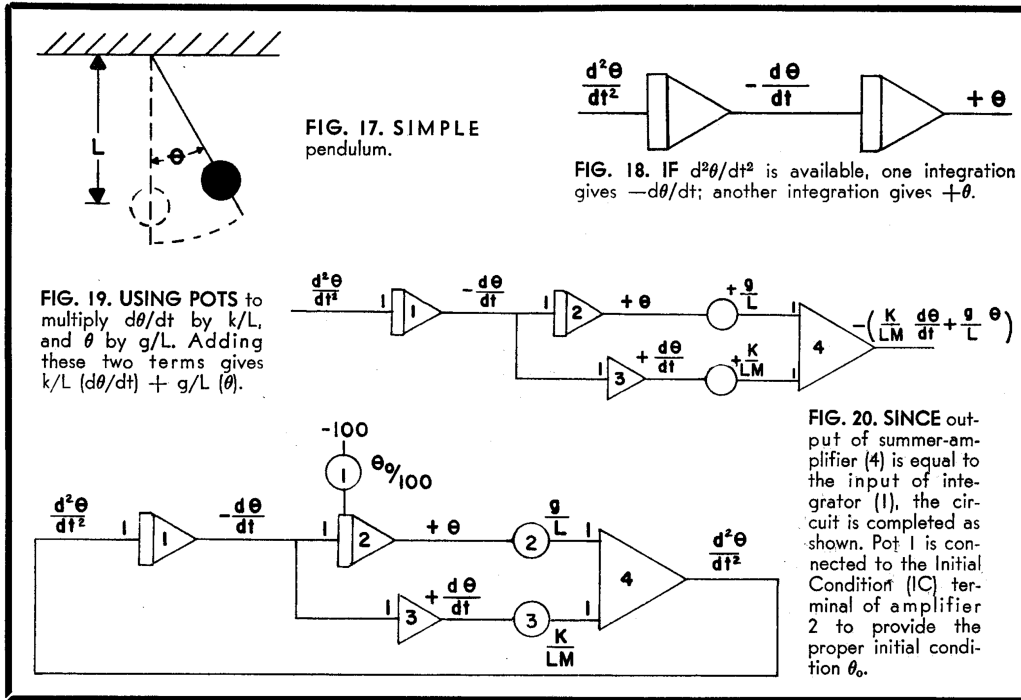


The **differential analyser** is a mechanical analogue computer designed to solve differential equations by integration, using wheel-and-disc mechanisms to perform the integration. It was one of the first advanced computing devices to be used operationally. The original machines could not add, but then it was noticed that if the two wheels of a rear differential are turned, the drive shaft will compute the average of the left and right wheels

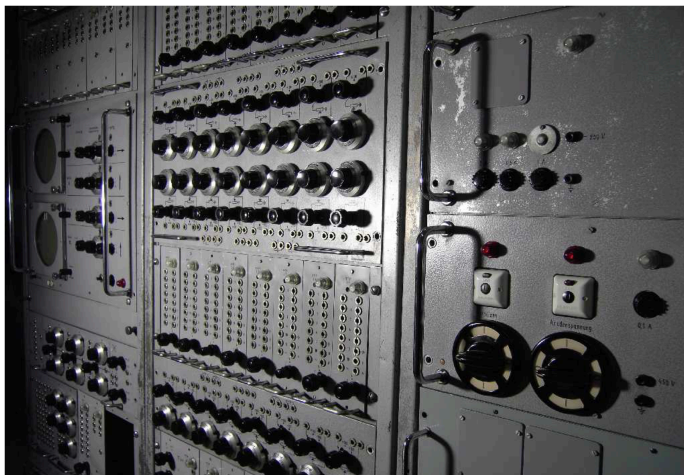
The **Water Integrator** was an early analog computer built in the Soviet Union in 1936 by Vladimir Lukyanov. It functioned by careful manipulation of water through a room full of interconnected pipes and pumps. The water level in various chambers (with precision to fractions of a millimeter) represented stored numbers, and the rate of flow between them represented mathematical operations. This machine was capable of solving inhomogeneous differential equations.







It all began in 1955 with the very first analog computer ever built by Telefunken:

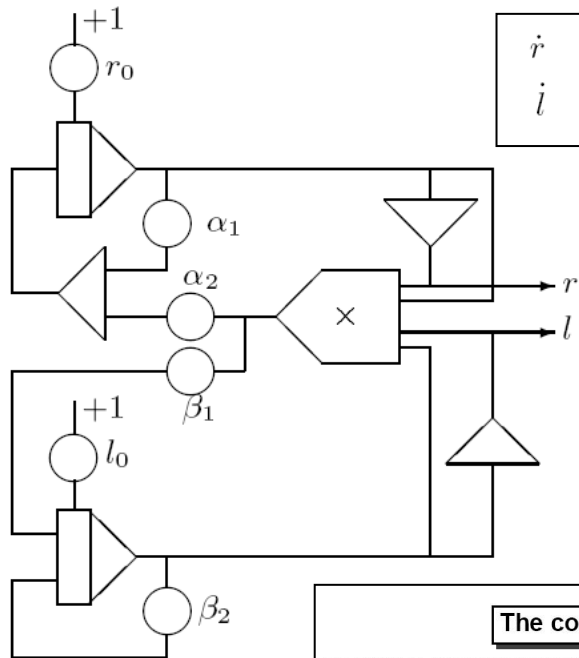


## Analog Computers 1950-1970

The Computational Tasks are defined by the connection topology of the *elementary* computing elements

# Volterra's Differential Equation

## Coupling both differential equations

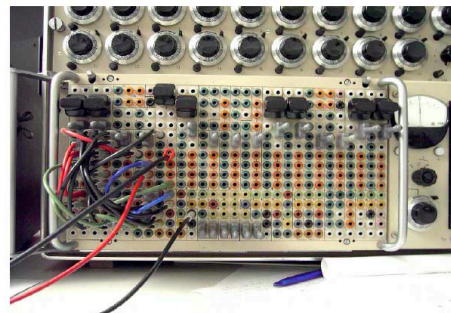


$$\begin{aligned}\dot{r} &= \alpha_1 r - \alpha_2 r l \\ \dot{l} &= -\beta_1 l + \beta_2 r l\end{aligned}$$

The Computational Tasks are defined by the connection topology of the *elementary* computing elements

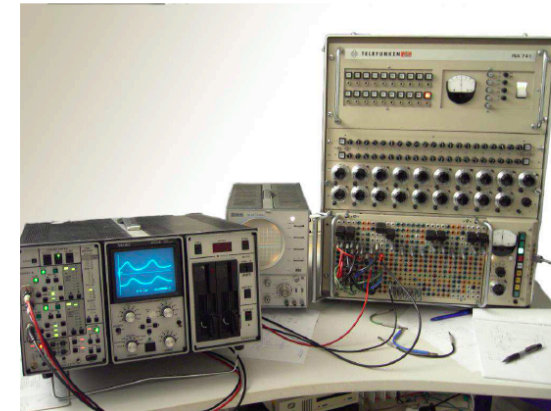
## The completed program

The following picture shows the program as patched for a Telefunken RA741 analog computer:



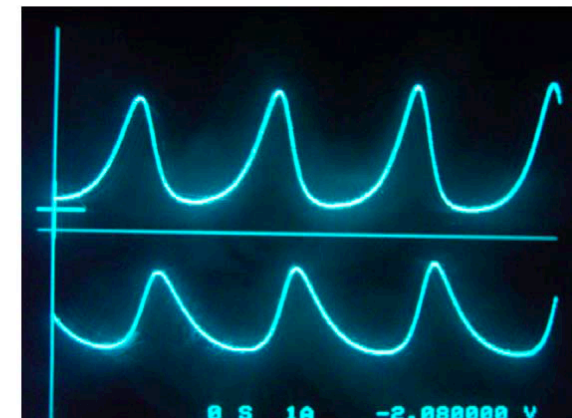
## The overall setup

The next picture shows the overall setup featuring a two channel storage oscilloscope:

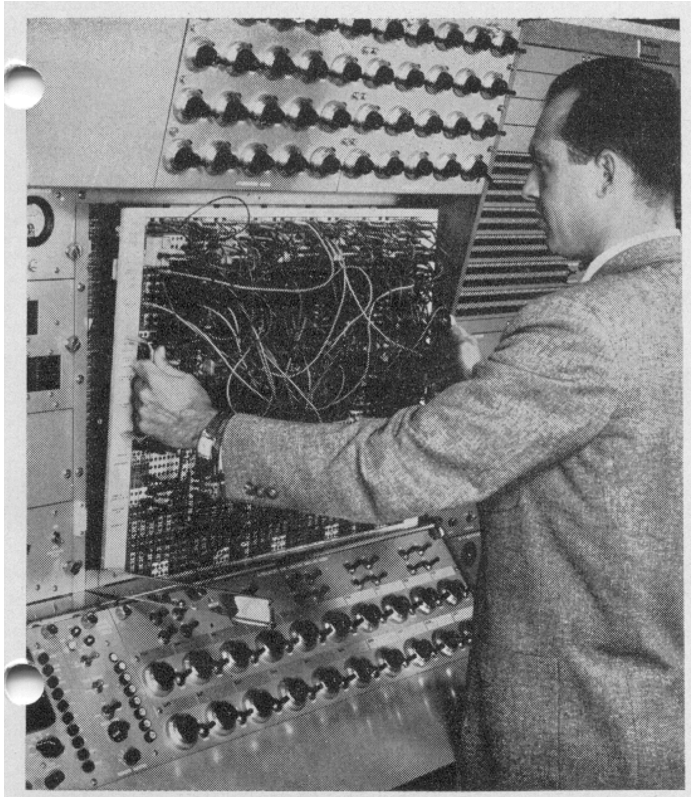


## Running the simulation

The picture below shows the results of the running simulation:







## Analog Computers 1950-1970

The Computational Tasks are defined  
by the connection topology of the  
*elementary* computing elements

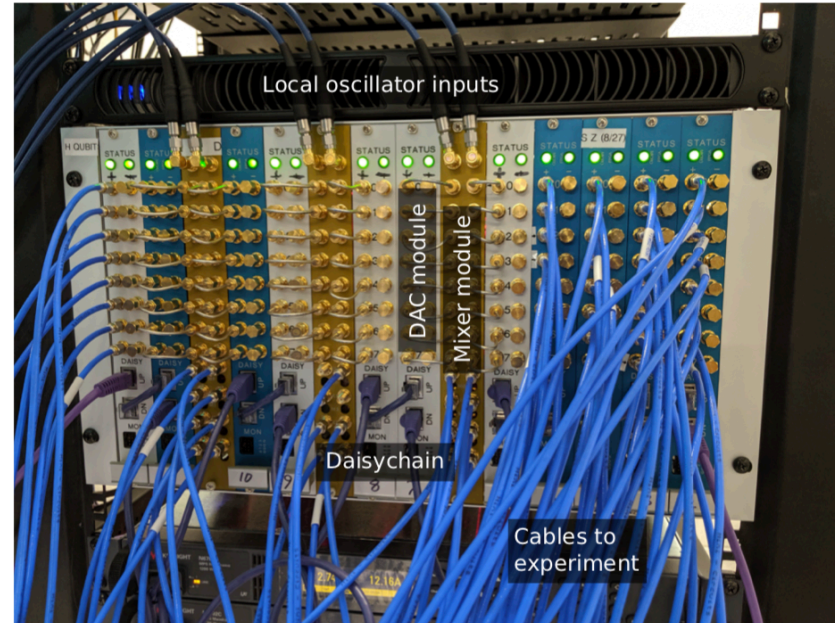


FIG. S4. **Electronics chassis.** Each chassis supports 14 DAC and/or mixer modules. Local oscillators are connected at the top of each mixer module. A set of daisychain cables connects from each ADC module to the next. Control signals exit the chassis through coaxial cables.

## Quantum (Analog ?) Computers 2020



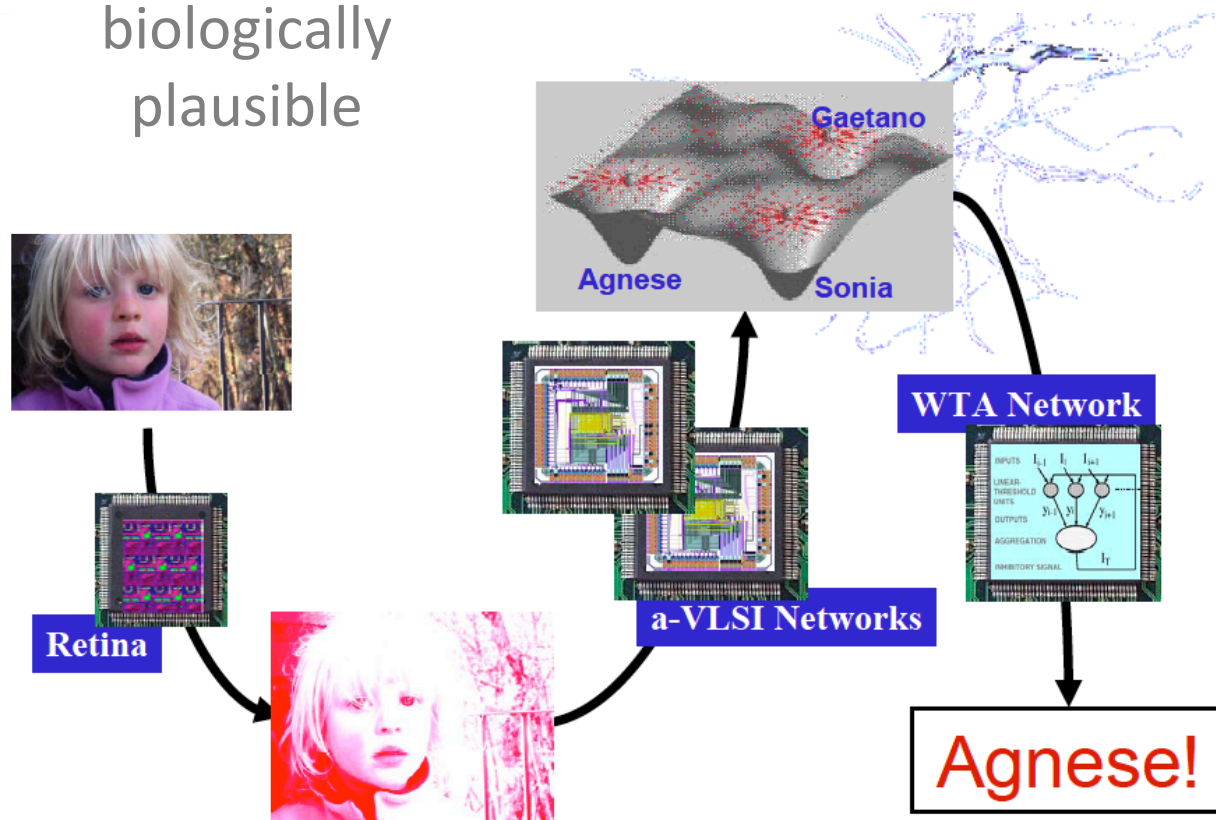
# Integrate and Fire Neurons & Cognitive Models

Real time  
classification  
processes and  
biologically  
plausible

**Neuromorphic Hardware:** It encompasses any electronic device which **mimics** the natural biological structures of our nervous system.

The goal is to impart cognitive abilities to a machine by implementing neurons in silicon.

Due to its much better energy efficiency and parallelism it is being considered as an alternative over conventional computational architectures



The Computational Tasks are defined  
by the connection topology of the  
*elementary* computing elements

# Hopfield Model & Associative Memory



Memory addressable  
by content.

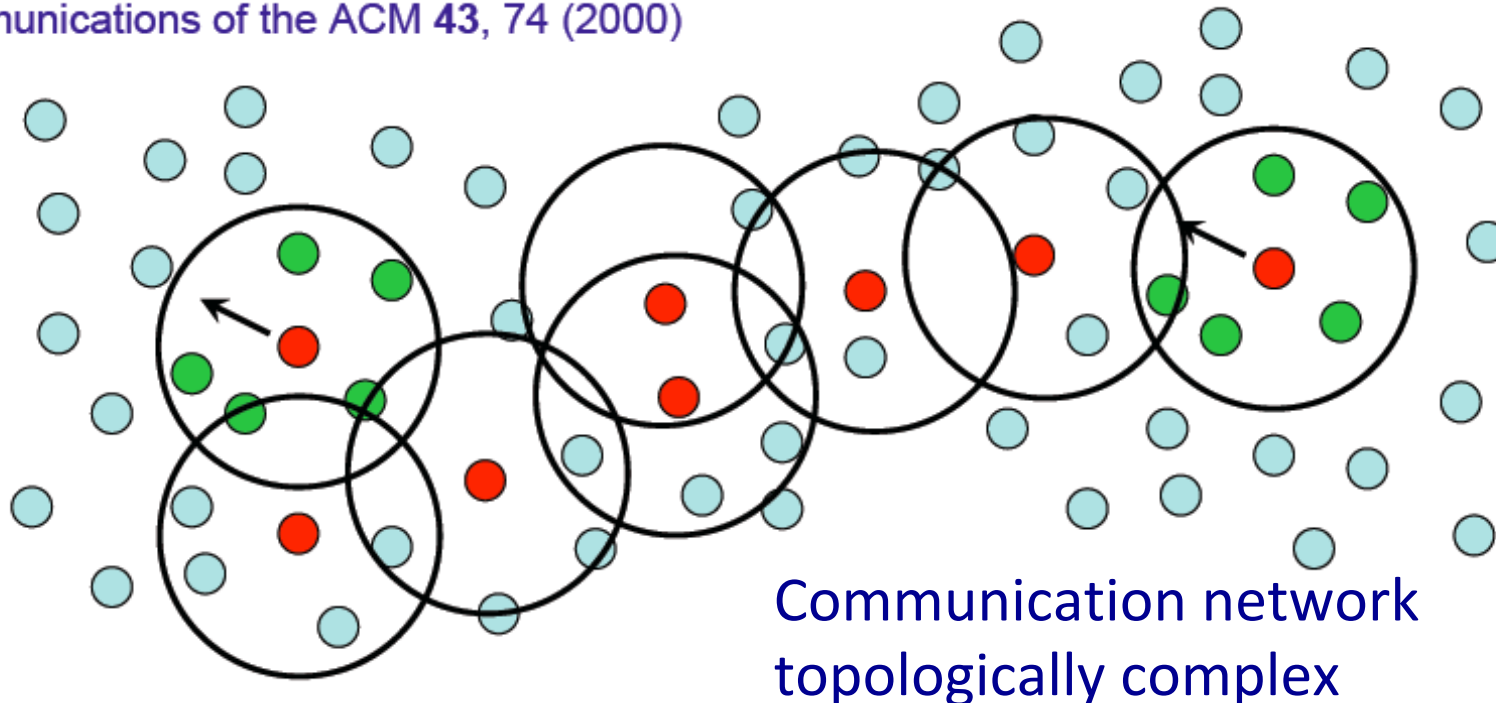
The recall of the  
information is  
triggered by a partial  
knowledge of the  
information



# Amorphous Computing

“How does one **engineer** pre-specified, coherent behavior from the **cooperation** of immense numbers of unreliable parts that are interconnected in unknown, irregular, and time-varying ways?”

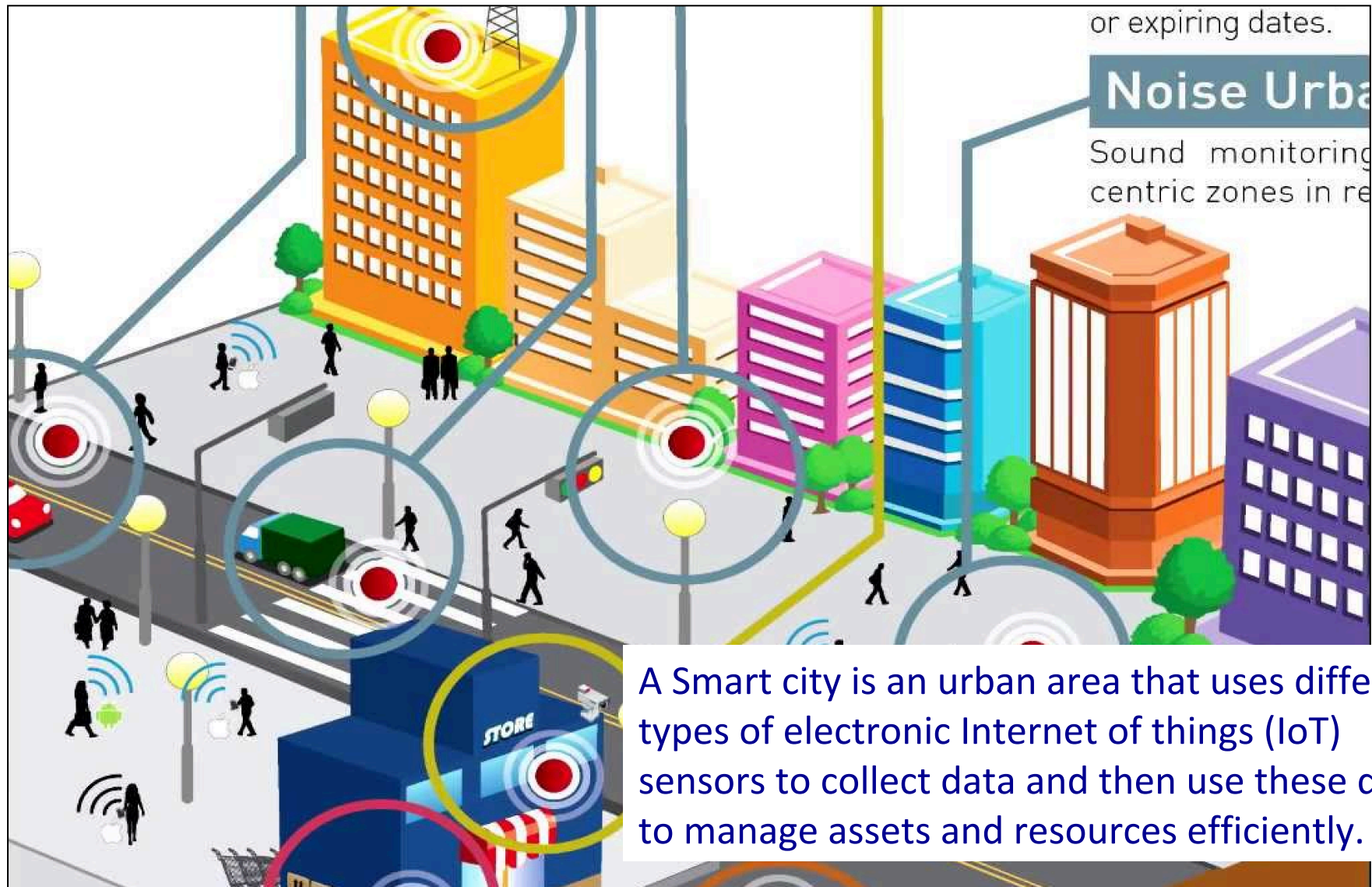
H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. Knight, R. Nagpal, E. Rauch, G. J. Sussman, and R. Weiss, *Amorphous Computing*, Communications of the ACM 43, 74 (2000)



Communication network  
topologically complex

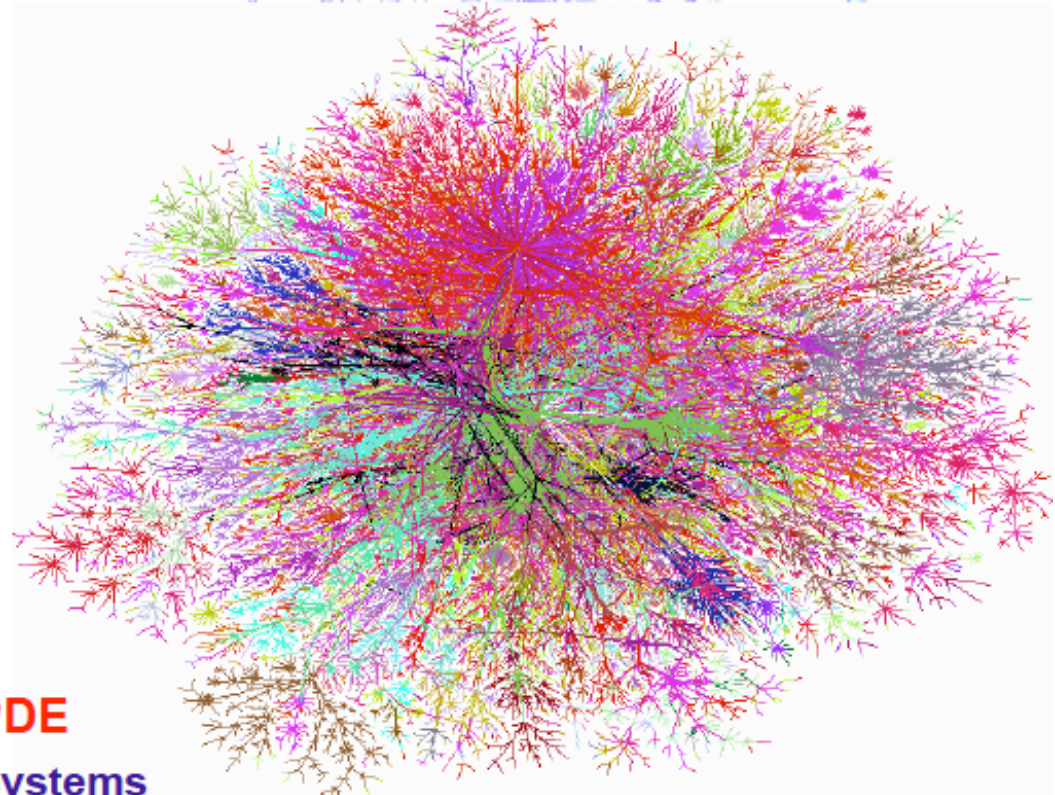


# Smart City as Amorphous System



## Physically-inspired Organizational Paradigms

- **Spatio-temporal Dynamics**  
Waves, Diffusion  
Pattern Formation
- **Complexity**  
Emergent Behavior  
Self-Organization  
*Can we tame complexity?*
- **Conservative systems and PDE**  
Definition of local coordinate systems



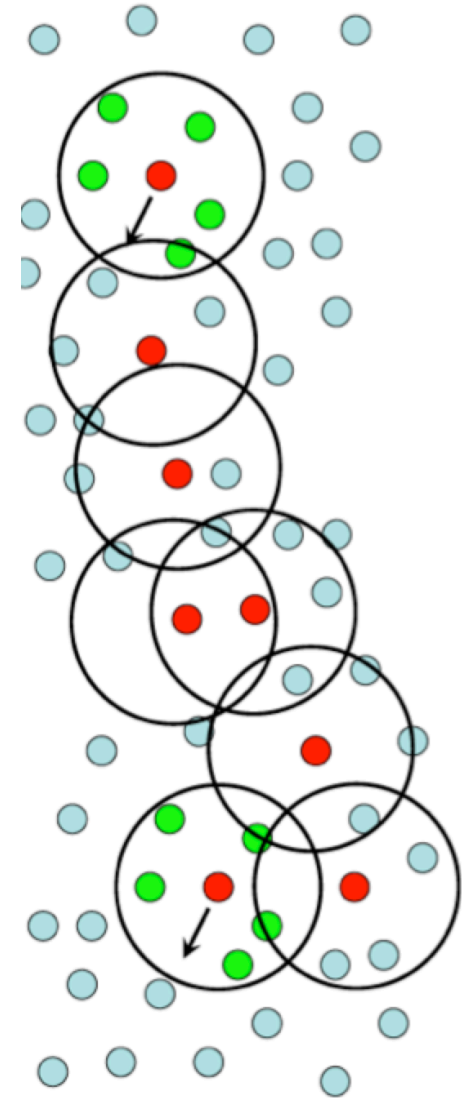
Albert, Reka et al. *Nature* 401, 130-131 (1999).

# Computational paradigms and topologically Complex Systems

Systems with a high number of degrees of freedom: elementary processors are connected by a topologically complex network that mediates their *interaction*.

Collective and naturally parallel behaviour, little influenced by the detailed nature of the temporal evolution of the single processor and the connection network.

Computational paradigms not defined a priori in algorithmic terms. System *code* is in its topological structure.





## Simple Systems

Simple systems are ones in which global properties are inherent in the properties of their component parts. Such systems are additive, and scale with increasing numbers of components.

Can be studied top-down or bottom up by traditional reductional science.

## Complex Systems

They can be defined by what they are not:

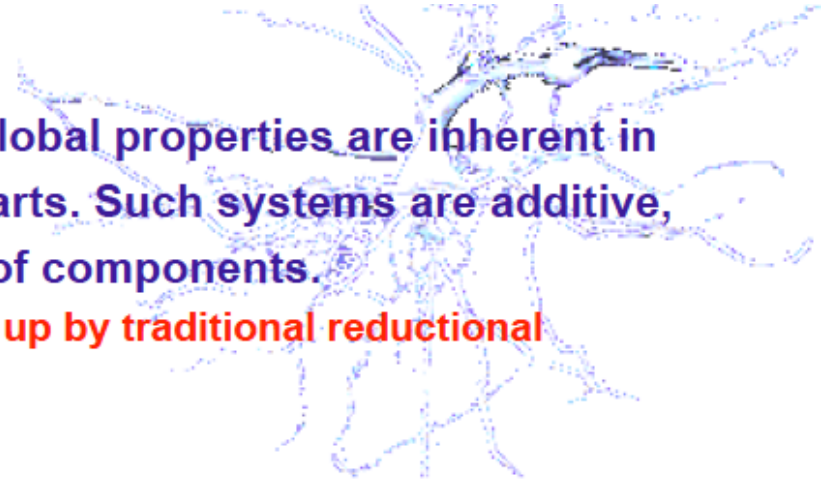
Complex systems are not simple ones.

The fundamental characteristic of a complex system is that it exhibits emergent properties.

Emergent properties are ones that arise due to the interactions in a system, and are not inherent in the individual components

There are almost as many definitions of CxSys as there are CxSys researchers. Many definitions include a notion of “surprise”.

Emergent properties can be surprising, but equating emergence with surprise is a statement about the human observer, not the system





**How machine learning work !**

There are **five basic steps** ...

**Step 2: Choose an algorithm**

Algorithm is a set of statistical processing steps. The algorithm depends on the type and amount data set and on the problem to be solved.

**Labeled Data**

- Regression algorithms: Linear regression predicts the value of a dependent variable based on the value of an independent variable. Logistic regression is used when the dependent variable is binary in nature.
- Decision trees: Decision trees use classified data to take decisions based on a set of decision rules.
- Instance-based algorithms: K-Nearest Neighbor. It uses classification to estimate how likely a data point is to be a member of one group or another based on its proximity to other points.

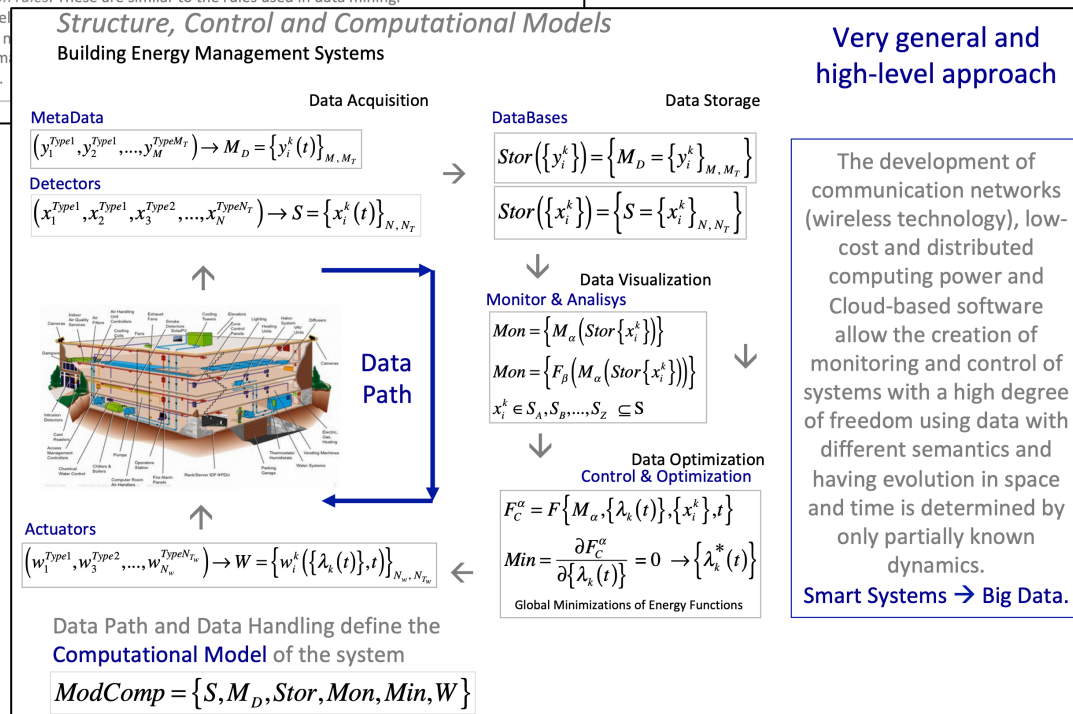
**Unlabeled Data**

- Clustering algorithms: Identifying groups of similar records and labeling the records according to the group. This is done without prior knowledge about the groups and their characteristics.
- Association algorithms: Association algorithms find patterns and relationships in data and identify frequent 'if-then' relationships called *association rules*. These are similar to the rules used in data mining.
- Neural networks: They were vague an algorithm that defines a layered network where calculations are performed and a conclusion is assigned a probability.

Gaetano Salina

Do we really need Machine Learning Methods in Physics?

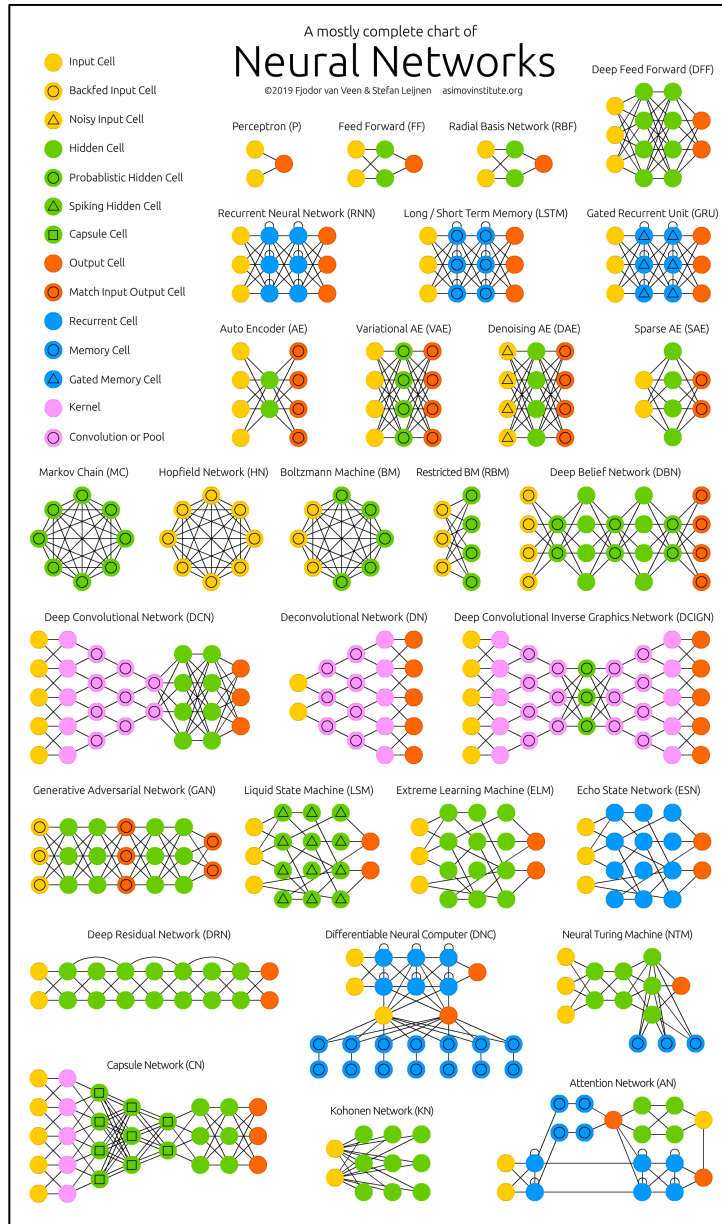
The idea according to which reliable knowledge extraction can be obtained solely on the grounds of our data sets faces insurmountable problems, even in the most idealized and controlled modeling setting.



The role of theoretical modeling cannot be discounted.

To assume that with enough data the numbers speak for themselves is simply false!

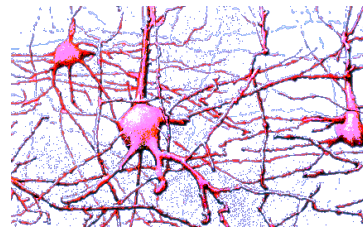
## The Neural Networks Zoo.



Neural networks represent incredibly exciting and powerful techniques used to solve problems.

While human-like deductive reasoning, inference, and decision-making by a computer is still a long time away, there have been remarkable gains in the development of computing paradigms emulating the human **brain**.

### Structurally Complex System



$10^9 - 10^{10}$  Neurons  
 $10^4 - 10^5$  Synapses/Neuron

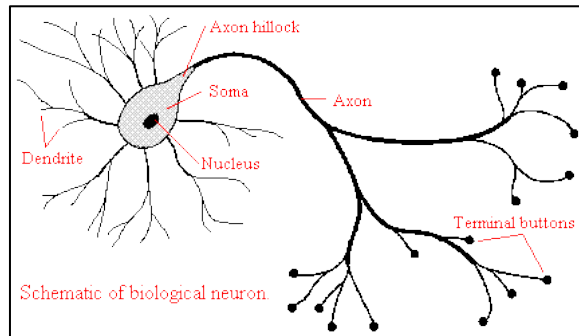
### Slow clock

$$T_N = 10^{-3} \text{ s}$$

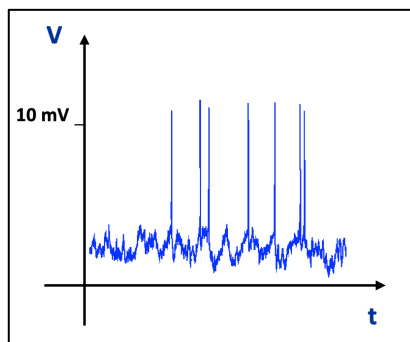
### Input-Compute-Output Structure

Stimulus Sense  $\rightarrow$  Transducer  $\rightarrow$  Processing  $\rightarrow$  Motor  
Neurons  $\rightarrow$  Motor System  $\rightarrow$  Spinal Cord  $\rightarrow$  Muscle

## Neuron



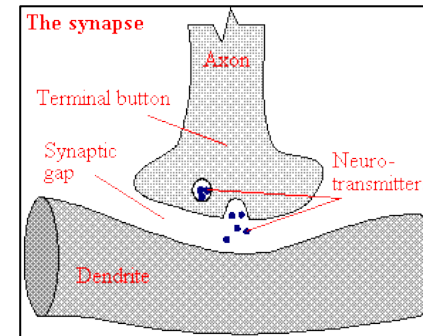
- Soma (Computing Element)
- Axon (Output Connection)
- Dendrites (Input Connections)



A difference in the concentration of sodium and potassium ions creates an electrical potential difference (of about -70 mV) across the cell

membrane of the soma. If the electric potential  $V$ , due to other neurons, is greater than the membrane potential, the neuron emits, for a few milliseconds, a train of impulses along the axon.

## Synapse



- Terminal Buttons

(80 % over Dendrites and 20 % over soma)



Neuro Transmitters  $\begin{cases} \text{Excitatory} \\ \text{Inhibitory} \end{cases}$

The learning process is seen as an alteration of the chemical-physical characteristics of the synapses: variations in the number and / or shape of the terminal buttons and variation in the conductivity of the dendrites

When an axon of a Neuron A is close to excite a Neuron B, and repeatedly and persistently takes part in its simulation, certain processes of growth or metabolic modification take place in both cells, such that the stimulation of Neuron A on Neuron B appears to be increased.

Hebb 1949

# The characteristics !

- Ability to classify and store information
- Information retrieval and error evaluation
- Informations processing

→ Dynamic unsupervised learning!

- Processing and storage as naturally **parallel processes**

→ A man sees a lion and Runs away  
in 0.1 second, i.e. 100 clock cycles

→ You know, that first love you never get over it  
also following a partial destruction of neurons and synapses

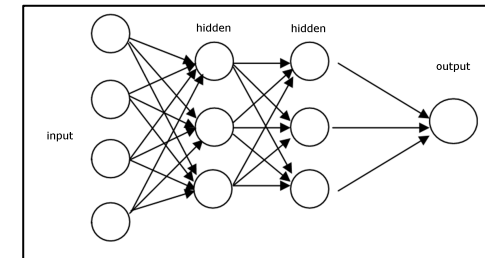
- Extreme plasticity
- Reduced size and weight  
(d < 30 cm; P < 1.5 Kg)
- Basso consumo  
(about 300 g pasta/day)

# The Models: Hearly History!

- **1943: Mc Culloch e Pitts.**  
Formal Neurons (0,1) and Propositional Calculus
- **1949: Hebb.**  
Rule of synaptic dynamics
- **1958: Rosemblat.**  
Extension of the formal Neuron, Perceptron
- **1973: Kohonen e Rouhonen.**  
Multiple Percettron
- **1974: Little, Hopfield, Amit.**  
Attractors Neuronal Networks.  
A new approach using methods of statistical mechanics: Dynamical Complex Systems
- **1990: The origin of NN Zoo.**
- ...

## Two classes of Neuronal Networks

Feed Forward Neural Networks (perceptor like) have a small number of neurons and low connectivity and are used as computational devices (self-organized filters)



Attractor Neural Networks have a high number of neurons and high connectivity and are seen as models of cognitive activity. ANN are dynamical complex systems and the attractors are stable states of them dynamic



## Perceptron & Propositional calculus

$$s_i \in S, i = 1, 2, 3, \dots, N$$

$$w_{i1} \in \mathfrak{R}, i = 1, 2, 3, \dots, N$$

$$y \in Y; \theta \in \mathfrak{R}$$

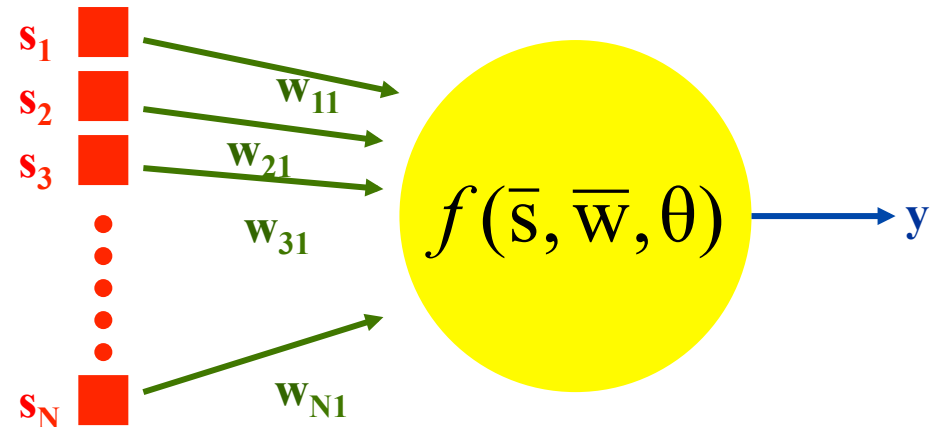
### • Nomenclature:

$s_i$  → pre-synaptic neurons

$y$  → post-synaptic neuron

$w_{i1}$  → synaptic weights

$\theta$  → activation threshold



$$\bar{s} \equiv \{s_i\}$$

$$\bar{w} \equiv \{w_{i1}\}$$

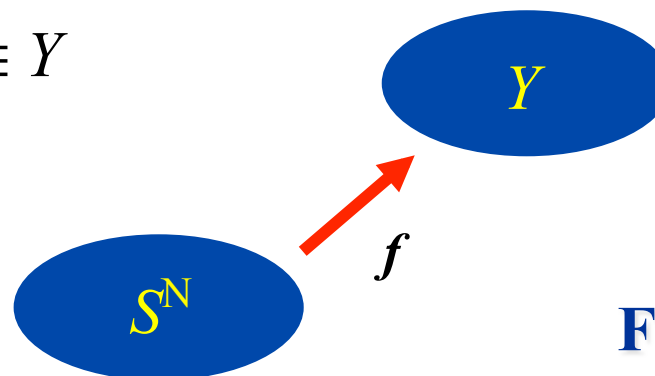
### • The system has a well defined transfer function $f$ . Given

$$\bar{s} \in S^N \quad \text{e} \quad y \in Y$$

**I have**

$$S^N \xrightarrow{f} Y$$

$$y = f(\bar{s}, \bar{w}, \theta)$$



**Formal Neuron**

## Perceptron & Propositional calculus

$$s_i \in Z_2, i = 1, 2, 3, \dots, N$$

$$w_i \in \mathfrak{R}, i = 1, 2, 3, \dots, N$$

$$y \in Z_2; \theta \in \mathfrak{R}$$

$$Z_2 \equiv (0, 1) \quad e \quad S^N \equiv Z_2^N$$

con

$$f(\bar{s}, \bar{w}, \theta) = \Theta\left(\sum_i^N s_i w_i - \theta\right)$$



**$2^N$  configuration states**

$$S^N \downarrow f(\bar{s}, \bar{w}_1, \theta_1)$$

**2 configuration states**



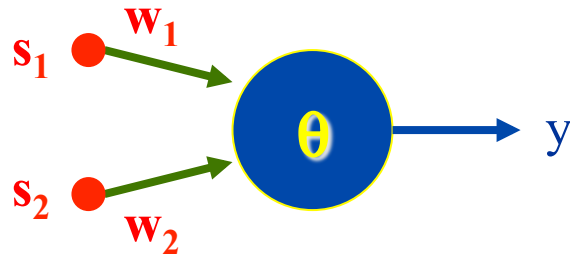
**$2^N$  configuration states**

$$S^N \downarrow f(\bar{s}, \bar{w}_2, \theta_2)$$

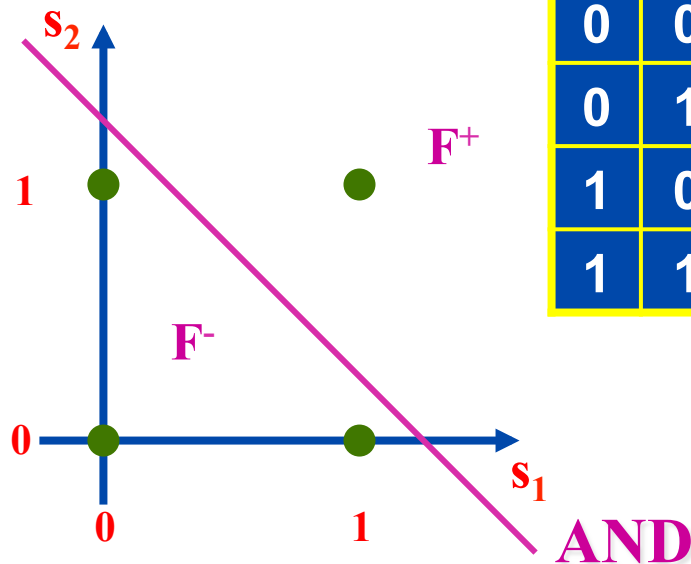
**2 configuration states**



## Perceptron & Propositional calculus: Binary logic AND



$s_1$	$s_2$	$y$
0	0	0
0	1	0
1	0	0
1	1	1



$$s_i \in Z_2, i = 1, 2$$

$$w_i \in \mathfrak{R}, i = 1, 2$$

$$y \in Z_2; \theta \in \mathfrak{R}$$

$$Z_2 \equiv (0, 1) \quad e \quad S^2 \equiv Z_2^2$$

$$f(\bar{s}, \bar{w}, \theta) = \Theta(s_1 w_1 + s_2 w_2 - \theta)$$

$$\rightarrow -\theta \leq 0$$

$$w_2 - \theta \leq 0 \quad \theta > w_1$$

$$w_1 - \theta \leq 0 \quad \rightarrow \quad \theta > w_2$$

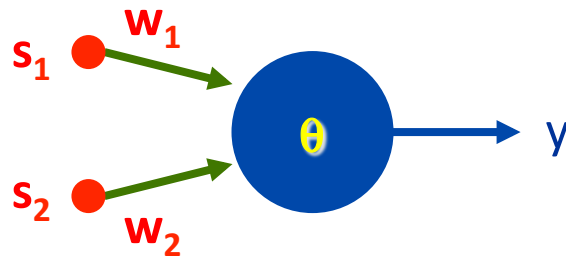
$$w_1 + w_2 - \theta > 0 \quad w_1 + w_2 > \theta \geq 0$$



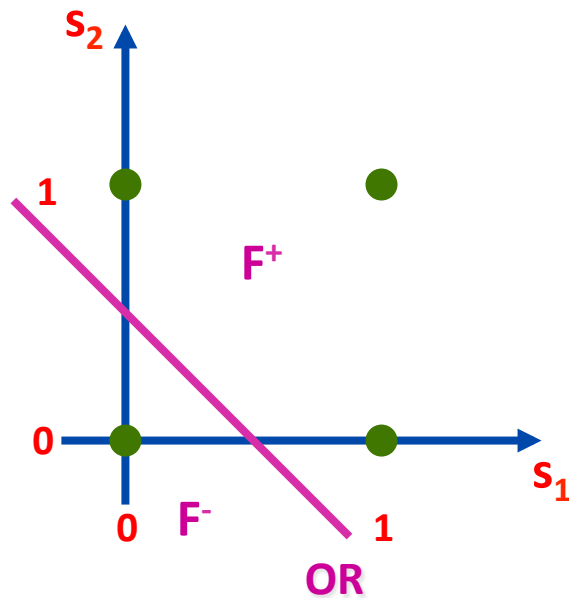
$$w_1^{\text{AND}} = w_2^{\text{AND}} = 1$$

$$\theta^{\text{AND}} = \frac{3}{2}$$

## Perceptron & Propositional calculus: Binary logic OR



$s_1$	$s_2$	$y$
0	0	0
0	1	1
1	0	1
1	1	1



$$s_i \in Z_2, i = 1, 2$$

$$w_i \in \mathbb{R}, i = 1, 2$$

$$y \in Z_2; \theta \in \mathbb{R}$$

$$Z_2 \equiv (0, 1) \quad e \quad S^2 \equiv Z_2^2$$

$$f(\bar{s}, \bar{w}, \theta) = \Theta(s_1 w_1 + s_2 w_2 - \theta)$$

$$\rightarrow -\theta \leq 0$$

$$w_2 - \theta > 0 \quad \theta < w_1$$

$$w_1 - \theta > 0 \quad \rightarrow \quad \theta < w_2$$

$$w_1 + w_2 - \theta > 0 \quad w_1 + w_2 > \theta \geq 0$$

$$w_1^{\text{OR}} = w_2^{\text{OR}} = 1$$

$$\theta^{\text{OR}} = 1/2$$

## Perceptron & Propositional calculus

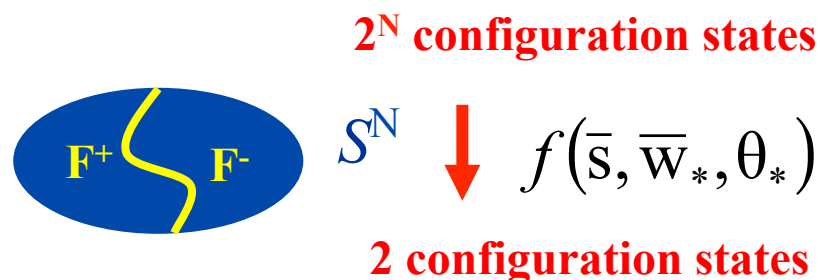
### Boolean functions space

Given  $\mathcal{S}^N \equiv Z_2^N$  as Source Space and  $Y \equiv Z_2$  as Target Space.

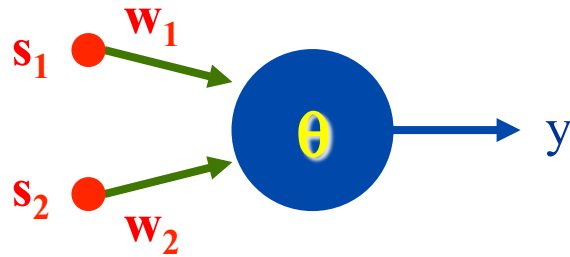
The number of all possible mappings  $\{f_k\}; Z_2^N \xrightarrow{f_k} Z_2$  are  $|\{f_k\}| = 2^{2^N}$

Can we define

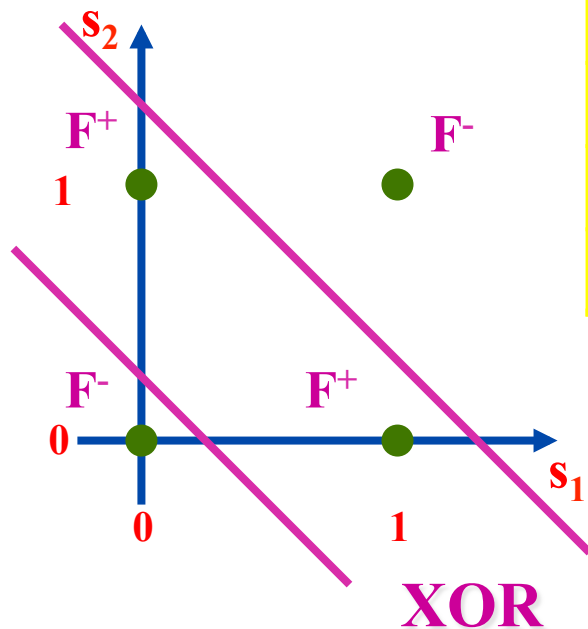
$\forall f_k \in \{f_k\}, \{\bar{w} = \bar{w}_*, \theta = \theta_*\} \rightarrow f_k$  if defined ?



# Perceptron & Propositional calculus: Binary logic XOR



$s_1$	$s_2$	$y$
0	0	0
0	1	1
1	0	1
1	1	0



$$s_i \in Z_2, i = 1, 2$$

$$w_i \in \mathbb{R}, i = 1, 2$$

$$y \in Z_2; \theta \in \mathbb{R}$$

$$Z_2 \equiv (0, 1) \quad e \quad S^2 \equiv Z_2^2$$

$$f(\bar{s}, \bar{w}, \theta) = \Theta(s_1 w_1 + s_2 w_2 - \theta)$$

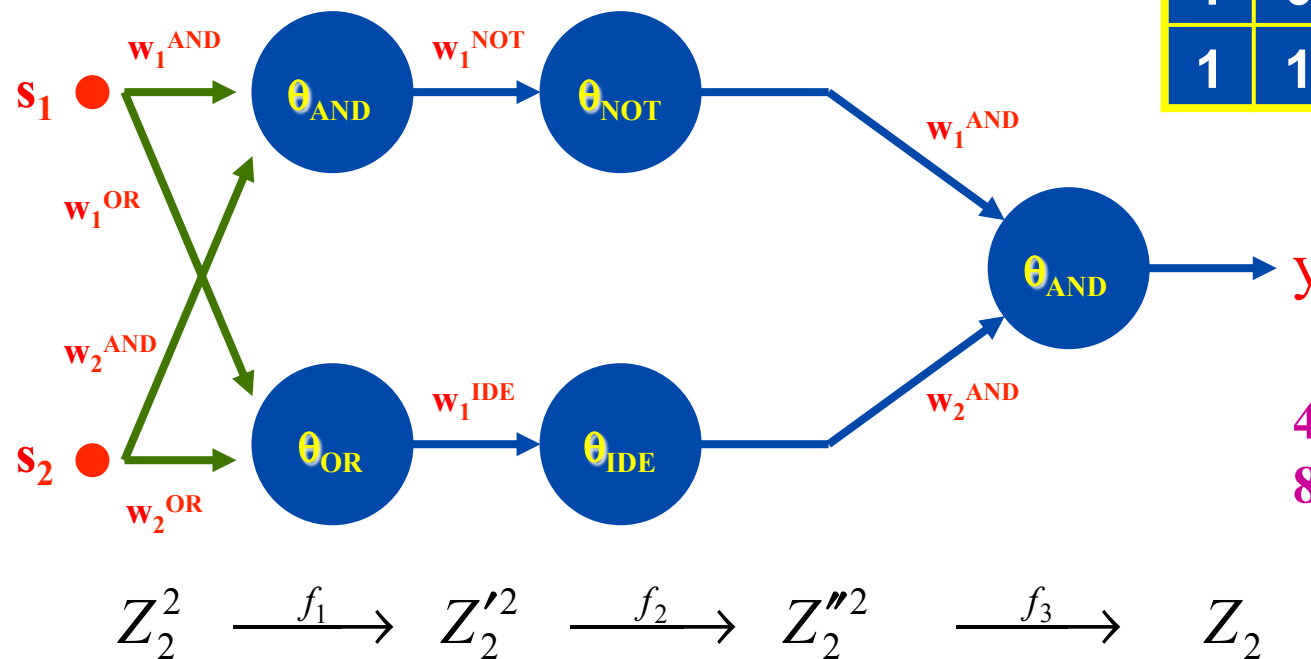
$$\begin{array}{ll} \rightarrow -\theta \leq 0 & \theta > 0 \\ w_2 - \theta > 0 & w_1 > \theta > 0 \\ w_1 - \theta > 0 & w_2 > \theta > 0 \\ w_1 + w_2 - \theta \leq 0 & w_1 + w_2 < \theta \end{array}$$

$$w_1^{\text{XOR}} = ?; w_2^{\text{XOR}} = ?; \theta^{\text{XOR}} = ?$$

## Perceptron & Propositional calculus: Binary logic XOR

$$\text{XOR}(s_1, s_2) = \text{AND}(\text{OR}(s_1, s_2), \text{NOT}(\text{AND}(s_1, s_2)))$$

$s_1$	$s_2$	OR	/AND	AND
0	0	0	1	0
0	1	1	1	1
1	0	1	1	1
1	1	1	0	0

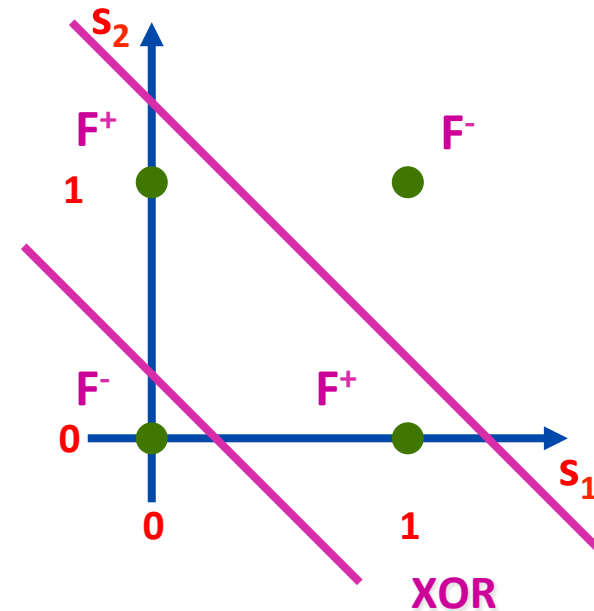
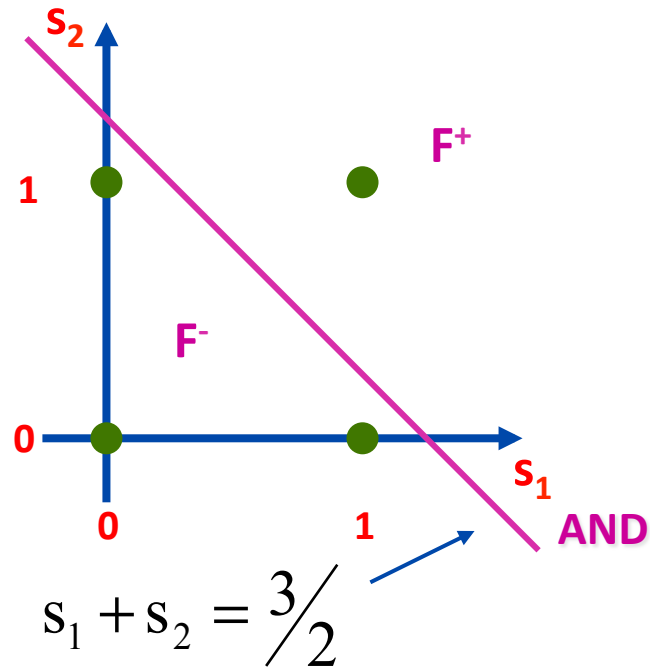


4 neurons layers and  
8 synaptic weights

## Perceptron & Propositional calculus: Binary logic XOR

### Where the problem arises ?

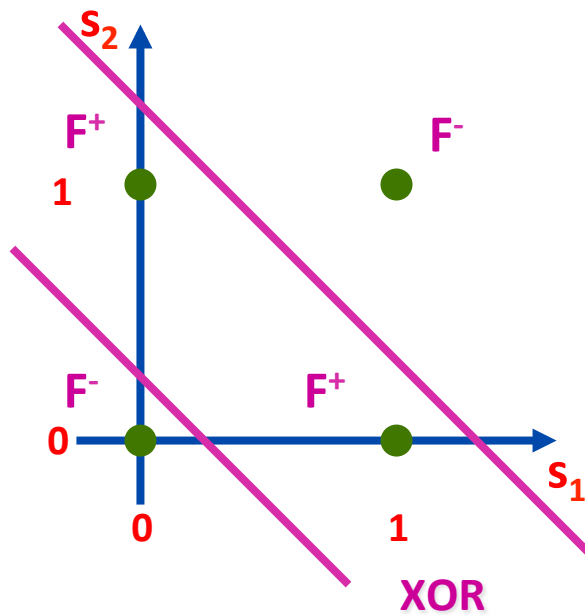
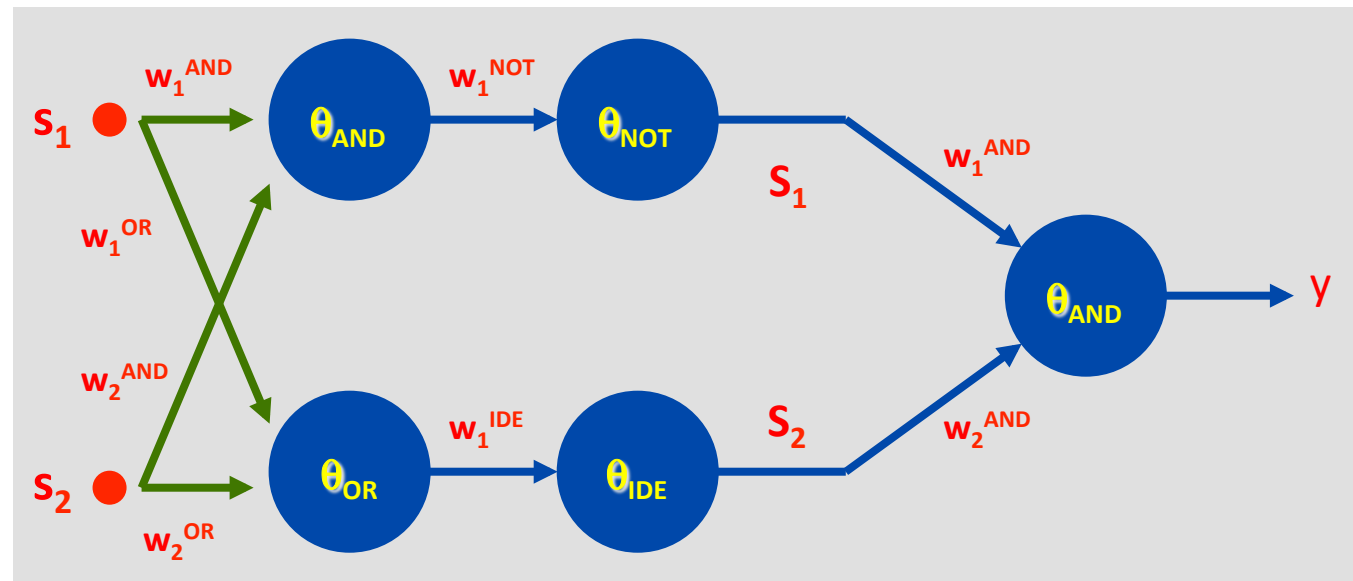
In the **AND** case it is possible to determine a line that separates the input space in two. The same cannot be done for the **XOR** !



Linearly separable space  $\longleftrightarrow f = \Theta\left(\sum_i^N s_i w_i - \theta\right)$

A two-valued function  $f$ , defined in an  $N$ -dimensional space  $S$ , is linearly separable if it is possible to find an  $N-1$  dimensional hyperplane that separates the space  $S$  into two regions ( $F_+$  and  $F_-$ ) where the function takes values 0 or 1.

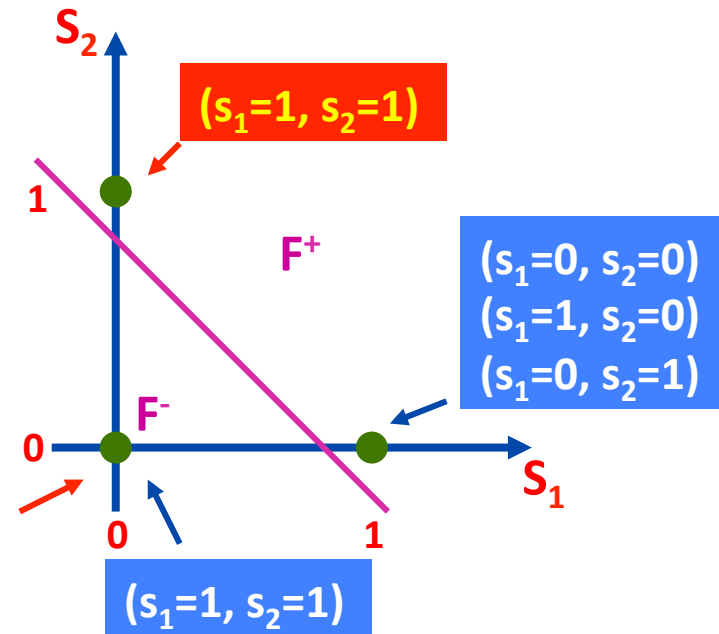
$$\begin{aligned} \text{XOR}(s_1, s_2) &= \\ \text{AND}(\text{OR}(s_1, s_2), & \\ \text{NOT}(\text{AND}(s_1, s_2))) &= \\ \text{AND}(S_1, S_2) \end{aligned}$$



Non Linear  
Trasformation



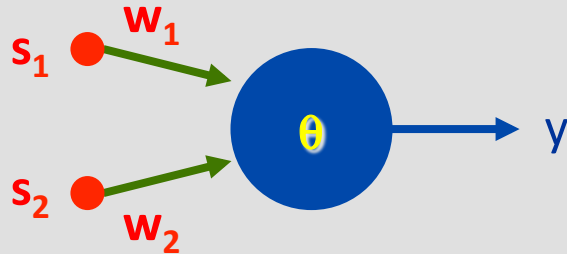
$(s_1=0, s_2=0)$   
 $(s_1=1, s_2=0)$   
 $(s_1=0, s_2=1)$





# From Formal Neuron to Perceptron:

## Formal Neuron



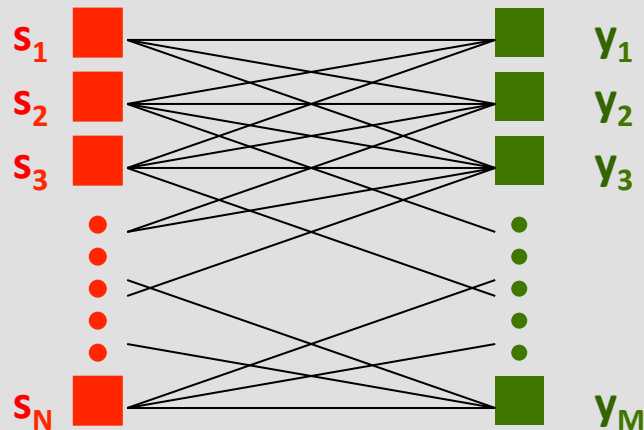
$$s_i \in Z_2, i = 1, 2$$

$$w_i \in \mathbb{R}, i = 1, 2$$

$$y \in Z_2; \theta \in \mathbb{R}$$



Direct Generalization



$$s_i \in S, i = 1, 2, 3, \dots, N$$

$$y_i \in Y, i = 1, 2, 3, \dots, M$$

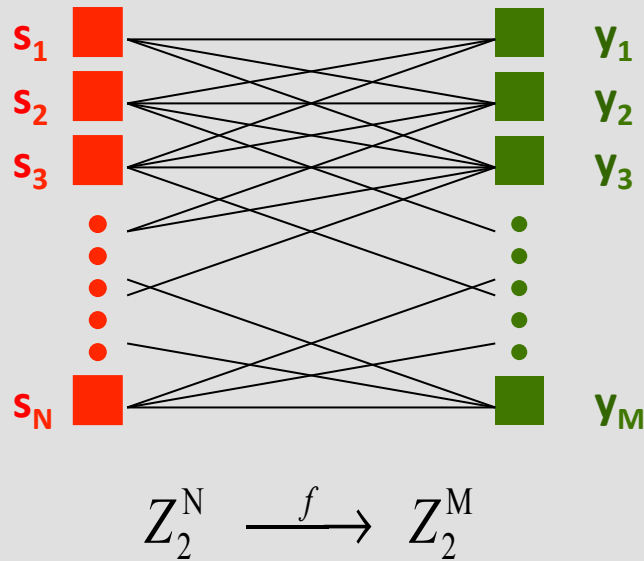
$$w_{ij} \in \mathbb{R}, i = 1, 2, 3, \dots, N; j = 1, 2, 3, \dots, M$$

$$\theta_i \in \mathbb{R}, i = 1, 2, 3, \dots, M$$

$$S^N \xrightarrow{f} Y^M$$

## Perceptron

# Perceptron Learning Rule:



$$s_i \in Z_2, i = 1, 2, 3, \dots, N$$

$$y_i \in Z_2, i = 1, 2, 3, \dots, M$$

$$w_{ij} \in \mathfrak{R}, i = 1, 2, 3, \dots, N; j = 1, 2, 3, \dots, M$$

$$\theta_i \in \mathfrak{R}, i = 1, 2, 3, \dots, M$$

$$\bar{s} \equiv \{s_i\}; \bar{y} \equiv \{y_i\}; \bar{w} \equiv \{w_{ij}\}, \bar{\theta} \equiv \{\theta_i\}$$

**Perceptron**



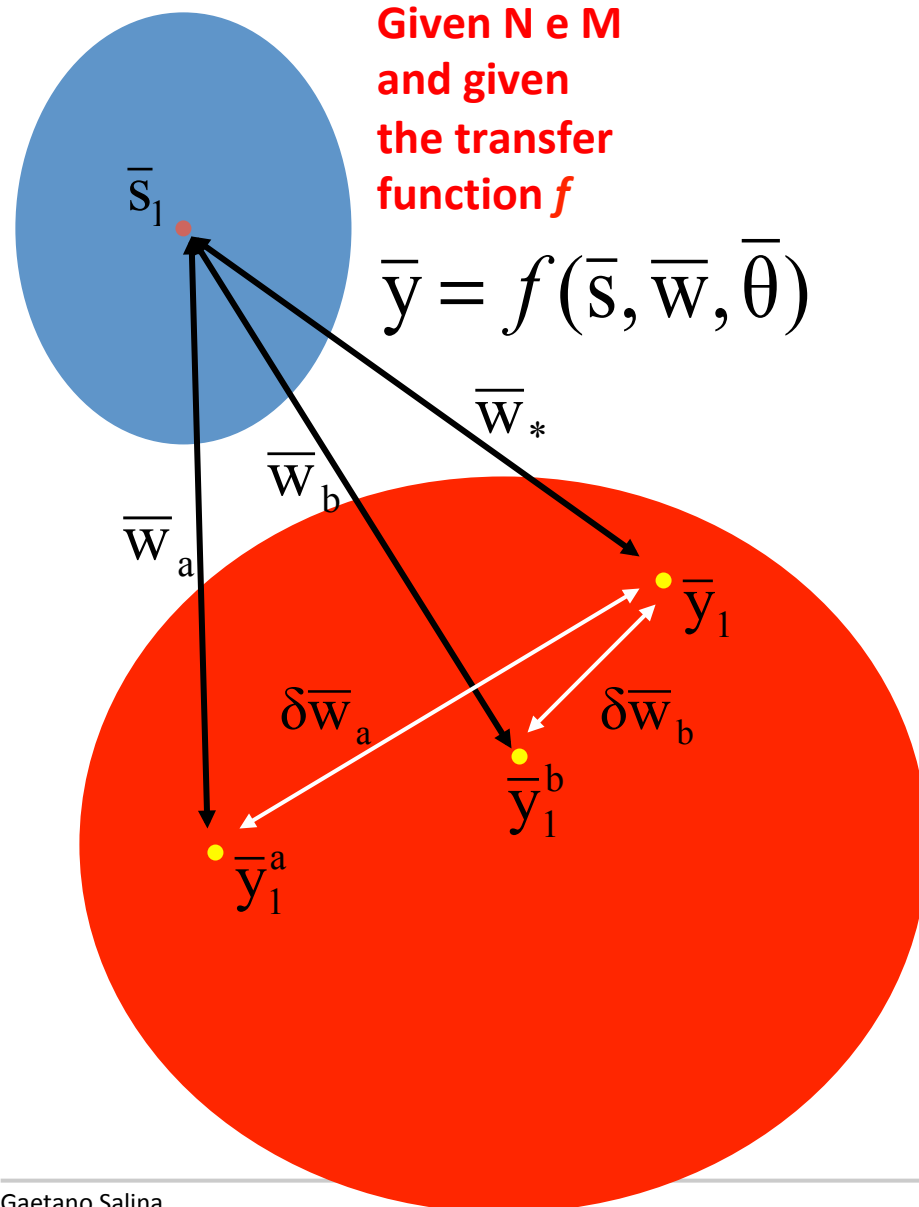
Given  $N$  e  $M$   
and given  
the transfer  
function  $f$

$$\bar{y} = f(\bar{s}, \bar{w}, \bar{\theta})$$



How is it possible to  
determine the synaptic  
weights in order to achieve  
the desired mapping

## Perceptron & Propositional calculus



How is it possible to  
determine the synaptic  
weights in order to achieve  
the desired mapping

### Iterative Algorithm

1) We choose  $\bar{w}_a$  random ( $\theta$  fixed).

$$\bar{y}_1^a = f(\bar{s}_1, \bar{w}_a, \theta)$$

2) and we change the synaptic weights

$$\delta \bar{w}_a = f(\bar{y}_1^a - \bar{y}_1)$$

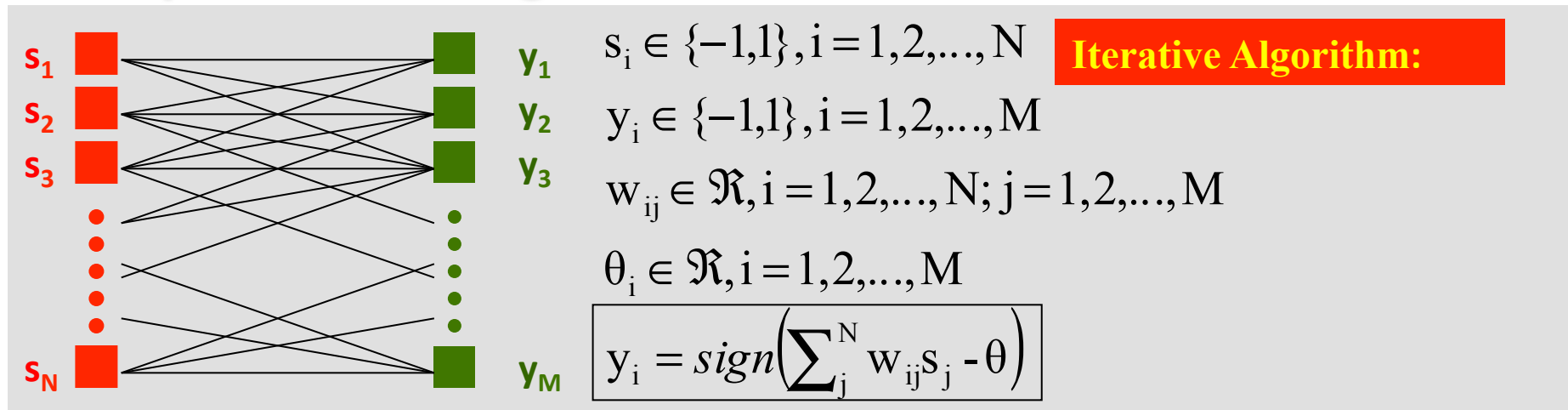
$$\bar{w}_b = \bar{w}_a + \delta \bar{w}_a$$

$$\bar{y}_1^b = f(\bar{s}_1, \bar{w}_b, \theta)$$

...

n)  $\bar{y}_1^n = \bar{y}_1 = f(\bar{s}_1, \bar{w}_*, \theta)$

# Perceptron Learning Rule:



Let:  $Y^\mu \equiv \bar{y}^\mu = \bar{y}(\bar{w}_*, \bar{s}^\mu)$   $\mu=1,2,\dots,2^N$

e, for a given weights set  $\bar{y}^\mu = \bar{y}(\bar{w}_a, \bar{s}^\mu)$   $\mu=1,2,\dots,2^N$

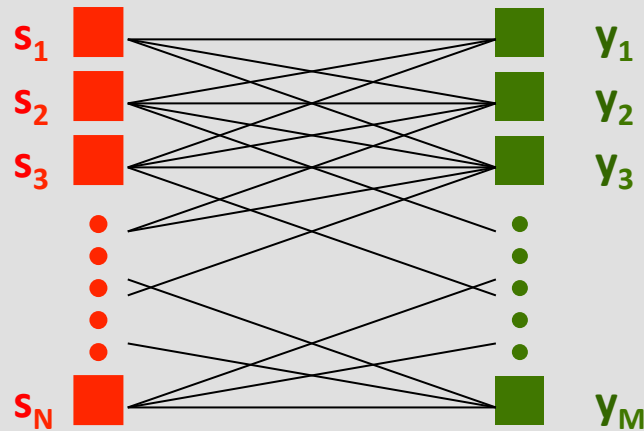
$\downarrow$   
 if  $(Y^\mu \equiv \bar{y}^\mu)$  {  
 $\delta \bar{w} = 0$

} else {  
 $\delta \bar{w} = f(\bar{s}^\mu, Y^\mu, y^\mu)$   
 }

$$\delta w_{i,k} = \varepsilon \sum_{\mu} \left( 1 - \sum_j^M y_j^\mu Y_j^\mu \right) Y_i^\mu s_k^\mu \quad \varepsilon \ll 1$$

$$\delta w_{i,k} = \varepsilon \sum_{\mu} (Y_i^\mu s_k^\mu - y_i^\mu s_k^\mu) \quad \varepsilon \ll 1$$

**Perceptron Learning Rule**



$$s_i \in \{-1, 1\}, i = 1, 2, \dots, N$$

$$y_i \in \{-1, 1\}, i = 1, 2, \dots, M$$

$$w_{ij} \in \mathbb{R}, i = 1, 2, \dots, N; j = 1, 2, \dots, M$$

$$\theta_i \in \mathbb{R}, i = 1, 2, \dots, M$$

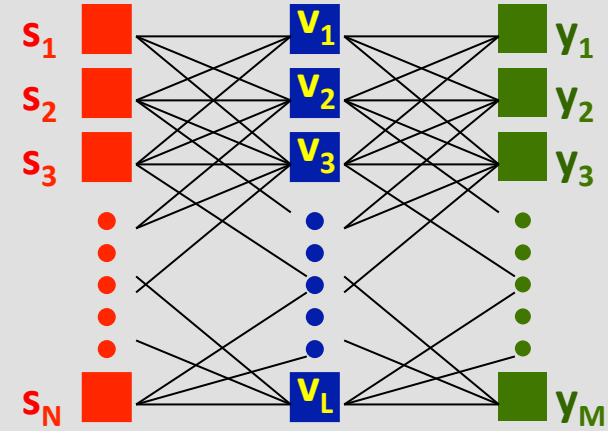
$$y_i = \text{sign}\left(\sum_j^N w_{ij} s_j - \theta_i\right)$$

$$\delta w_{i,k} = \varepsilon \sum_{\mu} (Y_i^{\mu} s_k^{\mu} - y_i^{\mu} s_k^{\mu})$$

Perceptron Learning Rule

## Perceptron

More  
Layers  
↔



$$s_i \in \{-1, 1\}, i = 1, 2, \dots, N$$

$$v_i \in \{-1, 1\}, i = 1, 2, \dots, L$$

$$y_i \in \{-1, 1\}, i = 1, 2, \dots, M$$

$$w_{ij}^1 \in \mathbb{R}, i = 1, 2, \dots, N; j = 1, 2, \dots, L$$

$$w_{ij}^2 \in \mathbb{R}, i = 1, 2, \dots, L; j = 1, 2, \dots, M$$

$$\theta_i^1 \in \mathbb{R}, i = 1, 2, \dots, L$$

$$\theta_i^2 \in \mathbb{R}, i = 1, 2, \dots, M$$

$$v_i = \text{sign}\left(\sum_j^N w_{ij}^1 s_j - \theta_i^1\right)$$

$$y_i = \text{sign}\left(\sum_j^L w_{ij}^2 v_j - \theta_i^2\right)$$

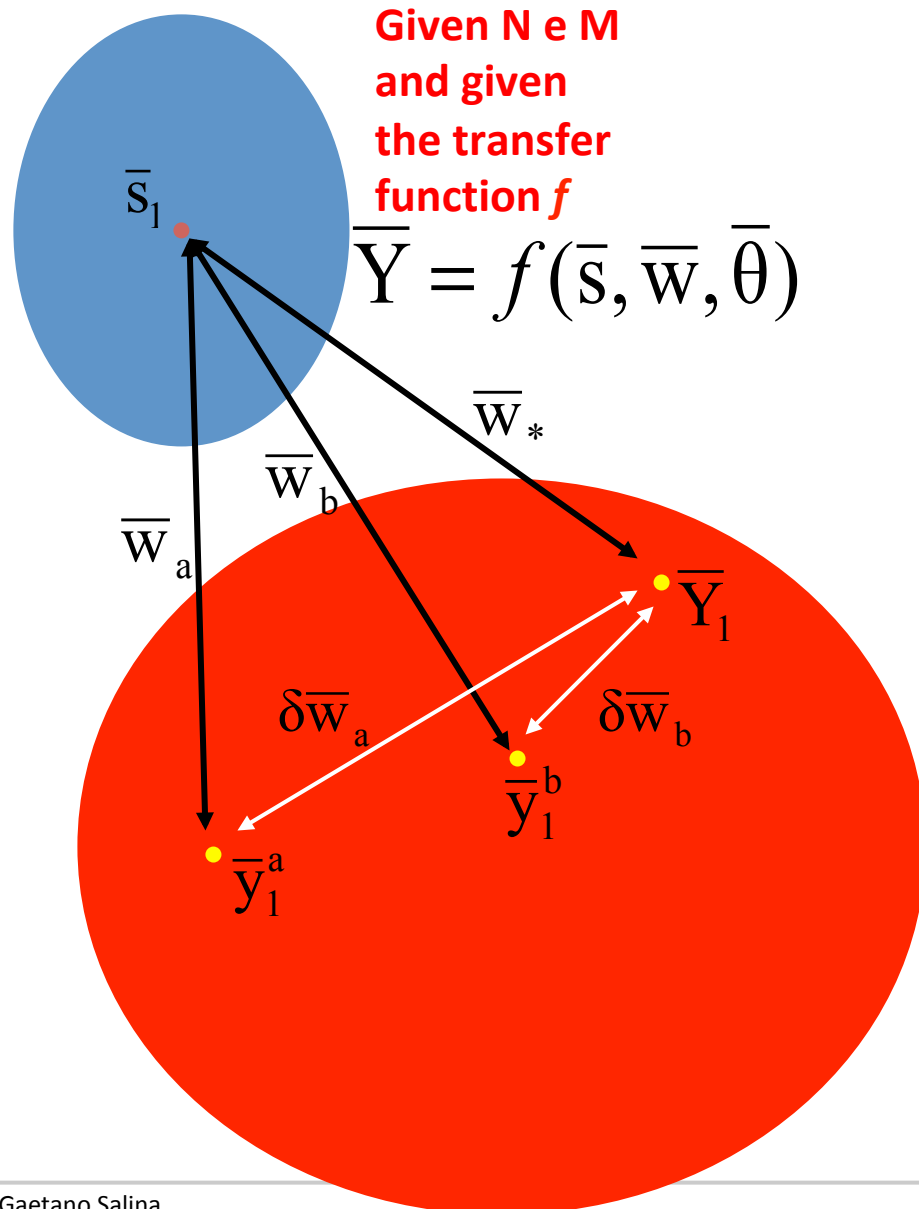
$$D = \frac{1}{2} \sum_{\mu} \sum_i (y_i^{\mu} - Y_i^{\mu})^2$$

Minimization

## Multiple Perceptron



## Perceptron & Propositional calculus



Define the Cost Function  $D$

$$D = \frac{1}{2} \sum_{\mu} \sum_i (y_i^{\mu} - Y_i^{\mu})^2 \Rightarrow D(\{w_{ij}\})$$

and let be:

$$D(\{w_{ij}\}) \xrightarrow{\{w_{ij}\} \rightarrow \{w_{ij}^*\}} 0$$

The point

$$\{w_{ij}^*\} \rightarrow D(\{w_{ij}^*\}) = 0$$

is a global minima of the Cost Function  $D$ , and can be determined with the Gradient Rule.

$$\delta w_{ij} = \varepsilon \sum_{\mu} (Y_i^{\mu} - y_i^{\mu}) y_i'^{\mu} s_j^{\mu}$$

Gradient Learning Rule

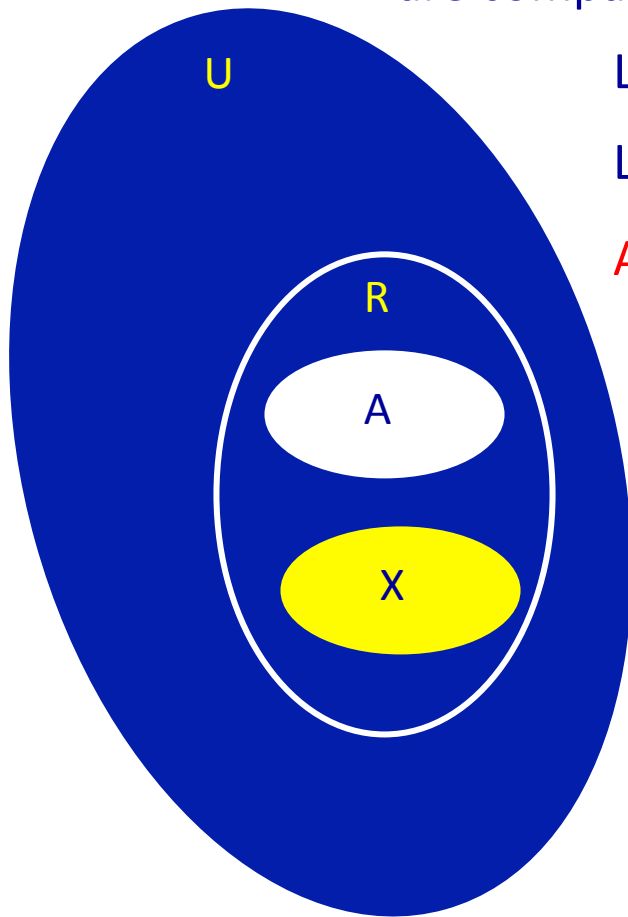
# Multiple Perceptron: Learning & Generalization

Let be  $U$  the set of all possible input-output rules, some of them are compatible with a given rule  $R$

Let be  $A$  the set used for the learning ( $p$  examples)

Let be  $X$  the set used for the model validation.

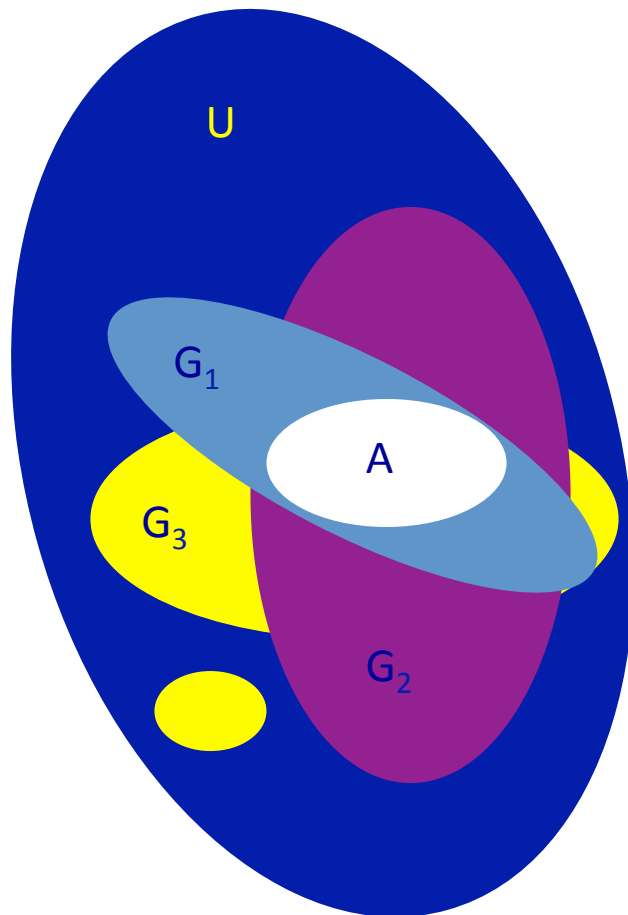
$A$  e  $X$  are random choosen and are representative of  $R$



The Network “known”  $A$  and knows nothing of  $X$  and  $R$

# Multiple Perceptron: Learning & Generalization

All possible generalizations compatible with A are valid !



Count Them?

$N$  Input Neurons



$2^N$  input patterns



$2^{2^N}$  possible functions



$P$  distinct examples in  $A$



$2^{2^N - P}$  different generalizations compatible with  $A$

# Multiple Perceptron: Learning & Generalization

Example:

$s_1 s_2$	$f$
00,01,10,11	0,0,0,0
00,01,10,11	0,0,0,1
00,01,10,11	0,0,1,0
00,01,10,11	0,0,1,1
00,01,10,11	0,1,0,0
00,01,10,11	0,1,0,1
00,01,10,11	0,1,1,0
00,01,10,11	0,1,1,1
00,01,10,11	1,0,0,0
00,01,10,11	1,0,0,1
00,01,10,11	1,0,1,0
00,01,10,11	1,0,1,1
00,01,10,11	1,1,0,0
00,01,10,11	1,1,0,1
00,01,10,11	1,1,1,0
00,01,10,11	1,1,1,1

Diagram illustrating the decomposition of a 4-bit function  $f$  into a sum of products (AND-OR-NOT structure). The input space is partitioned into regions  $A)p_1$  and  $B)p_1$ . The output  $f$  is determined by the combination of these regions and the specific input bits  $s_1, s_2$ .

Logical operations indicated by red arrows:

- AND
- XOR
- OR
- NOR
- NXOR
- NAND

A)

p	$s_1$	$s_2$	y
1	0	0	1
2	1	1	1

$$2^{2^N-p} = 2^{2^2-2} = 4$$

B)

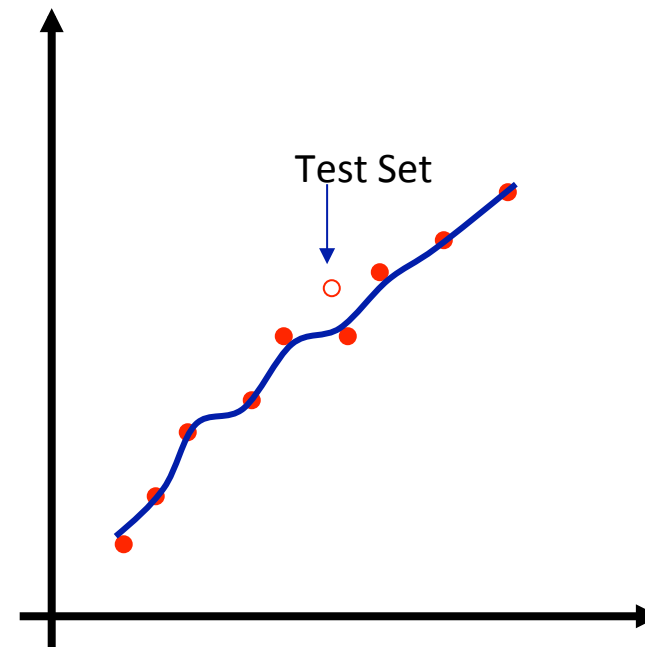
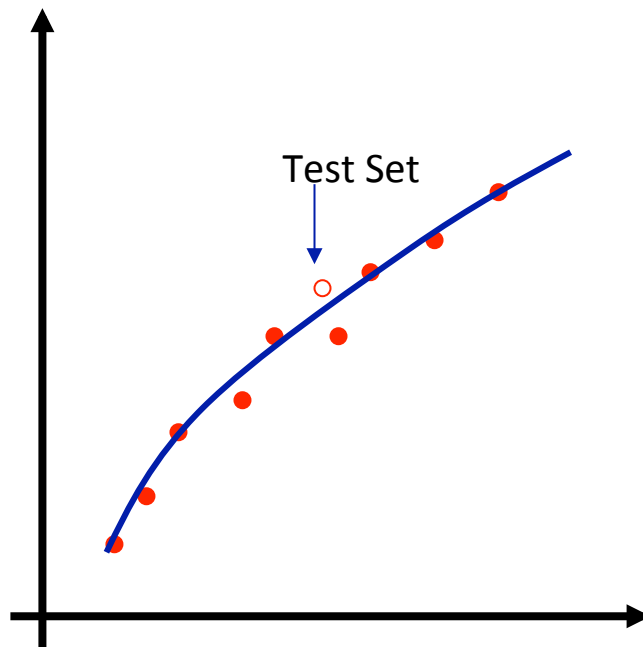
p	$s_1$	$s_2$	y
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	1

$$2^{2^N-p} = 2^{2^2-4} = 1$$

The information needed to uniquely fix a rule on  $N$  bits is  $2^N$  bits, but reasonable rules are specified by no more than  $N^k$  bits, for small  $k$

# Multiple Perceptron: Learning & Generalization

Overfitting:



Too many synaptic weights in a network severely limit the generalization capabilities



# Multiple Perceptron: Learning & Generalization

## Theoretical Framework:

The previous discussion suggests the possibility of a quantitative estimate of what a network can or cannot do according to its architecture and the nature and size of the learning set.

We will focus:

- The average number of alternative generalizations of a given learning set
- The average probability that a trained network generates the correct output for a randomly chosen input
- As above, but in the worst case.

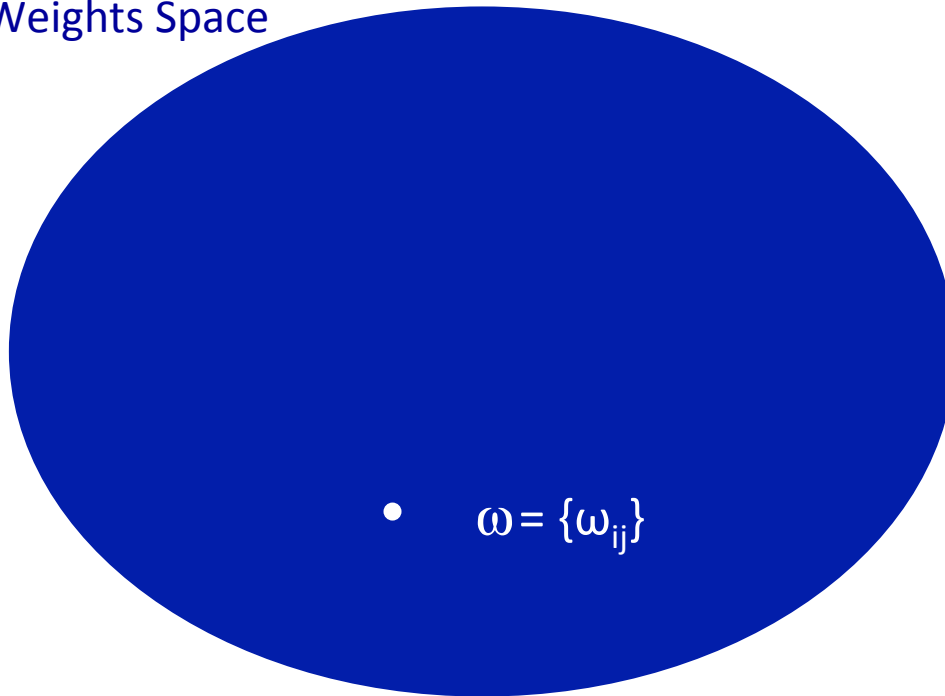
# Multiple Perceptron: Learning & Generalization

## Theoretical Framework:

Suppose we work with a class of networks with a fixed architecture (number of hidden layers, number of neurons for each layer, etc ...)



Weights Space



Average over all possible  
Network

=

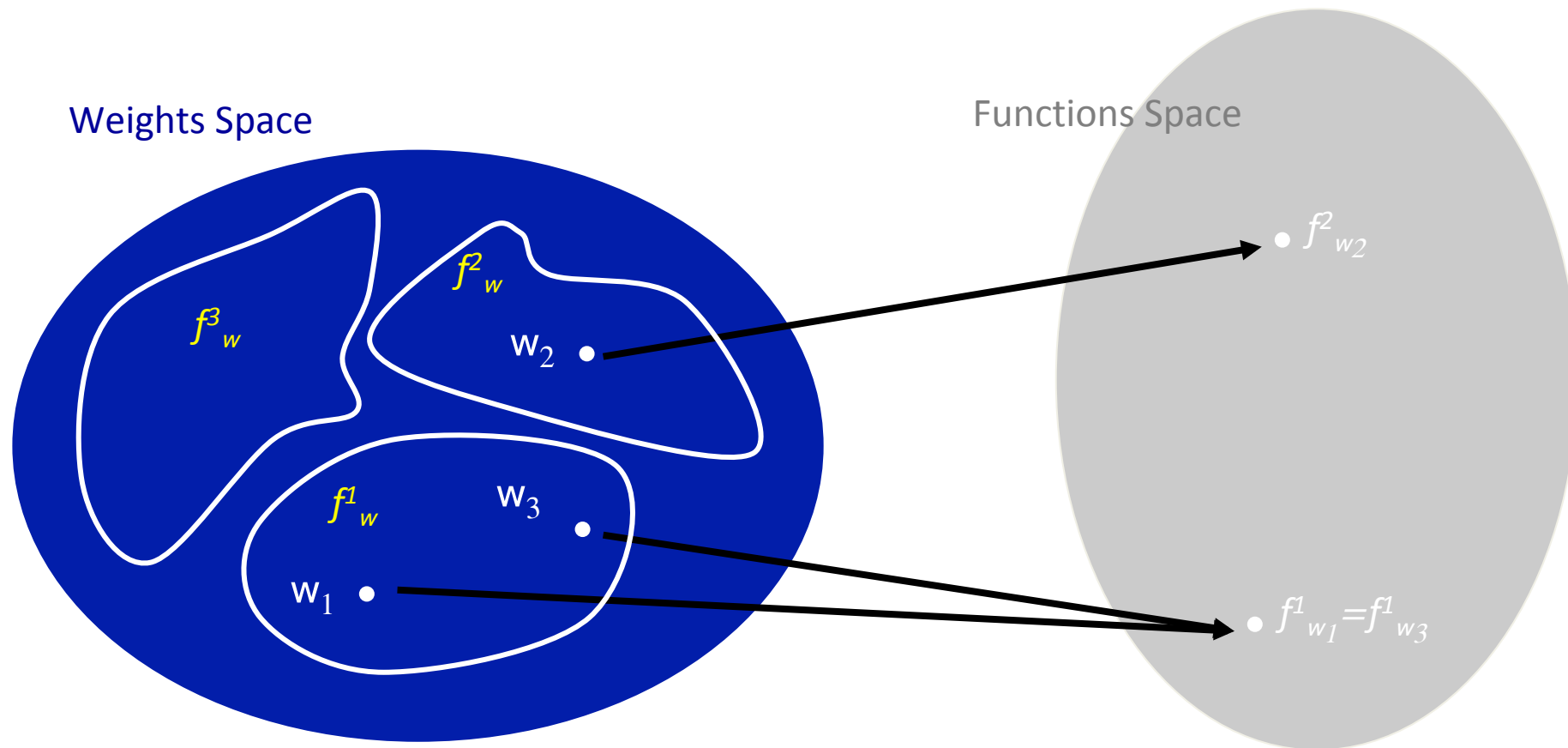
Average over Weights Space  
with a given states density  $\rho(\omega)$

$$V_0 = \int_{\omega} d\omega \rho(\omega)$$

Total Volume

# Multiple Perceptron: Learning & Generalization

## Theoretical Framework:



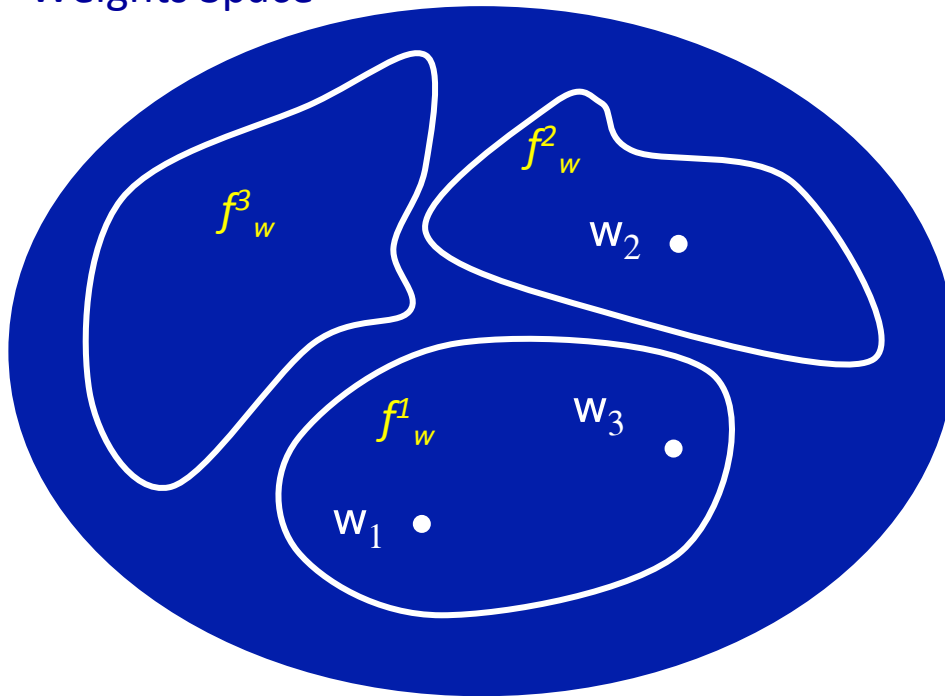
For the sake of simplicity we will use  
binary scalar functions

# Multiple Perceptron: Learning & Generalization

## Theoretical Framework:

The total volume of the region of the space of weights that implements a particular function  $f$  is:

Weights Space



$$V_0(f) = \int_{\omega(f)} d\omega \rho(\omega)$$

$$V_0(f) = \int_{\omega} d\omega \rho(\omega) \theta_f(\omega)$$

where

$$\theta_f(\omega) = \begin{cases} 1 & \text{if } f_{\omega}(\bar{s}) = f(\bar{s}) \quad \forall \bar{s} \\ 0 & \text{otherwise} \end{cases}$$

The ratio  $R_0(f) = \frac{V_0(f)}{V_0}$

is the fraction of the space of the weights that implements the given function, or is it the probability of having the same function if we randomly choose the weights with density  $\rho(\omega)$

# Multiple Perceptron: Learning & Generalization

## Theoretical Framework:

We define the function:

$$S_0 = - \sum_f R_0(f) \ln_2(R_0(f))$$



Bigger is  $S_0$  more information is necessary to determine  $f$

Which measures the functional diversity of architecture.  
Formally the function  $S_0$  it is an entropy associated with the information content of the network.

---

Exemple:  $K$  possible functions with equal volume:

$$R_0(f) = \frac{1}{K} \Rightarrow S_0 = \ln_2 K \Rightarrow K = 2^{S_0}$$

$2^{S_0}$  represents a good estimate of the number of functions even when the volume of individual functions is not the same

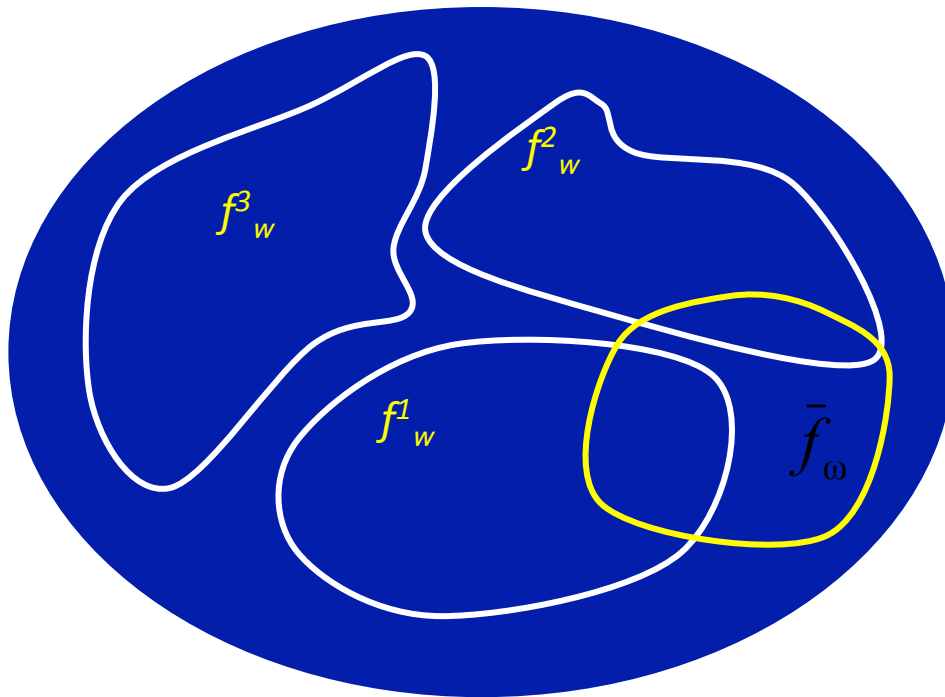


# Multiple Perceptron: Learning & Generalization

Theoretical Framework:

Supervised learning:

Weights Space



$$Y^\mu = \bar{f}(\bar{s}^\mu) \quad \mu = 1, 2, \dots, p$$

Now

$$V_p(\bar{f}) = \int d\omega \rho(\omega) \prod_{\mu=1}^p I(f_\omega, \bar{s}^\mu)$$

where

$$I(f_\omega, \bar{s}^\mu) = \begin{cases} 1 & \text{if } f_\omega(\bar{s}^\mu) = \bar{f}(\bar{s}^\mu) \\ 0 & \text{otherwise} \end{cases}$$

The weights in the region  $V_p$  belong to the function  $\bar{f}$  plus regions corresponding to other functions compatible with  $\bar{f}$  on the learning set.

# Multiple Perceptron: Learning & Generalization

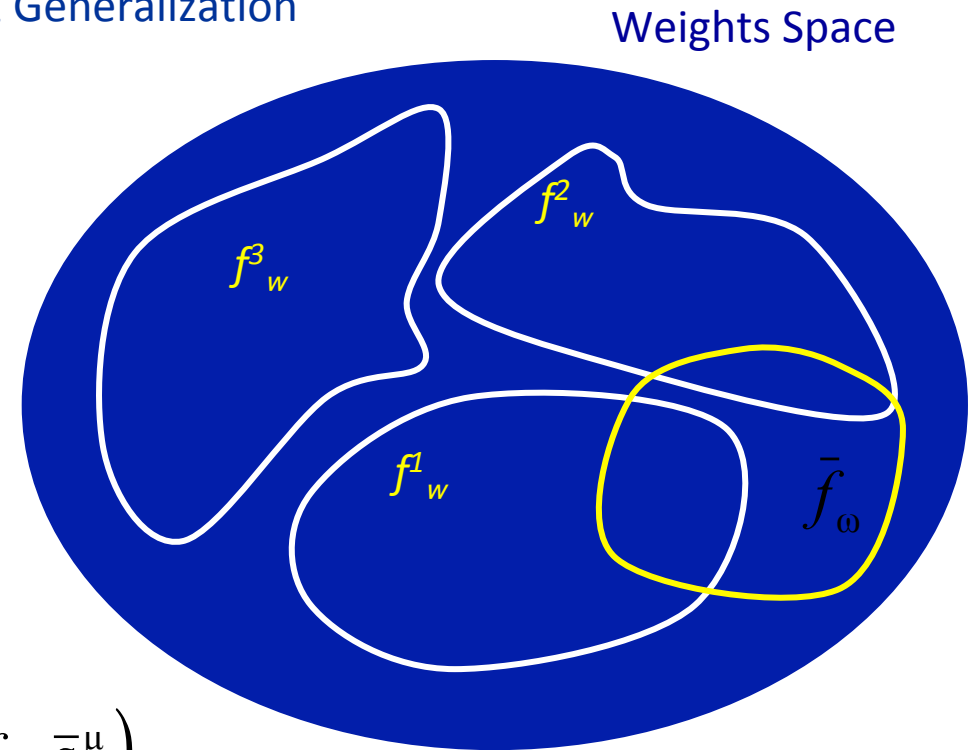
## Theoretical Framework:

Supervised learning: define

$$R_p(f^1) = \frac{V_p(f^1)}{V_p(\bar{f})}$$

where

$$\begin{aligned} V_p(f^1) &= \int d\omega \rho(\omega) \theta_{f^1}(\omega) \prod_{\mu=1}^p I(f_\omega, \bar{s}^\mu) \\ &= V_0(f^1) \prod_{\mu=1}^p I(f^1, \bar{s}^\mu) = \begin{cases} V_0(f^1) & \text{se } f^1(\bar{s}^\mu) = \bar{f}(\bar{s}^\mu) \\ 0 & \text{altrimenti} \end{cases} \end{aligned}$$



# Multiple Perceptron: Learning & Generalization

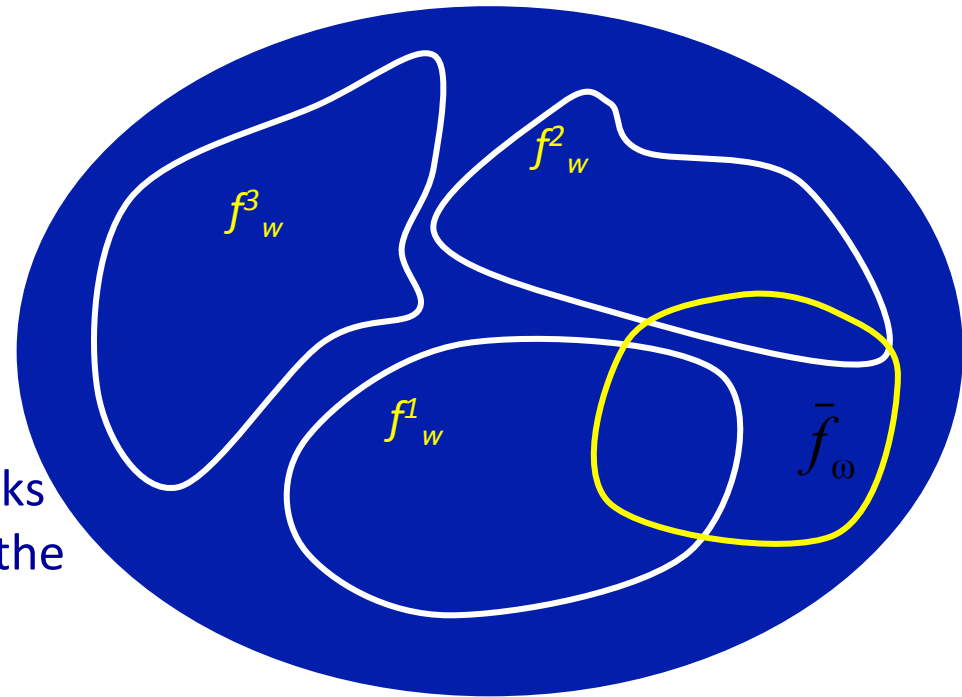
## Theoretical Framework:

Weights Space

The corresponding Entropy:

$$S_p = - \sum_f R_p(f) \ln_2(R_p(f))$$

it is the measure of how much it works implementable are compatible with the learning set.



$$p \Rightarrow 2^N$$

$$S_p \Rightarrow 0$$

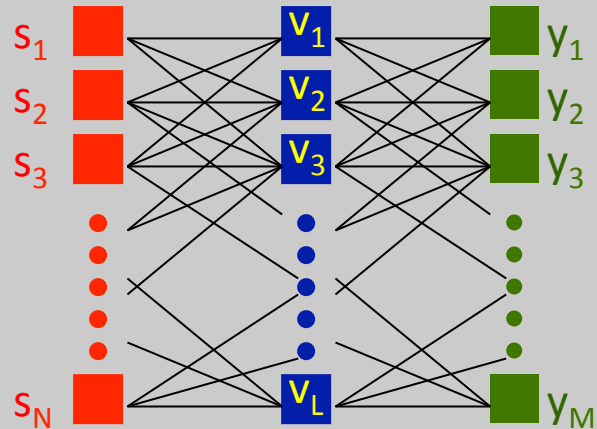
$$\bar{f}_w \Rightarrow f_w^1$$



$$\Delta S_p = S_{p-1} - S_p$$

It represents the information gain by adding an additional pattern to the learning set.

# Multiple Perceptron:



$$s_i \in \{-1, 1\}, i = 1, 2, \dots, N$$

$$v_k \in \{-1, 1\}, k = 1, 2, \dots, L$$

$$y_j \in \{-1, 1\}, j = 1, 2, \dots, M$$

$$w_{ik}^1 \in (-1, 0, 1) \equiv Z_3, i = 1, 2, \dots, N; k = 1, 2, \dots, L$$

$$w_{kj}^2 \in (-1, 1) \equiv Z_2, k = 1, 2, \dots, L; j = 1, 2, \dots, M$$

$$v_k = f\left(\sum_i^N w_{ik}^1 s_i\right)$$

$$y_j = f\left(\sum_k^L w_{jk}^2 v_k\right)$$

Discrete weights  $\rightarrow$  Finite weight space

$$M_{sp} = \frac{3^{NL} 2^{ML}}{L!} \quad \leftarrow \text{Permutation of the neurons of the hidden layer}$$

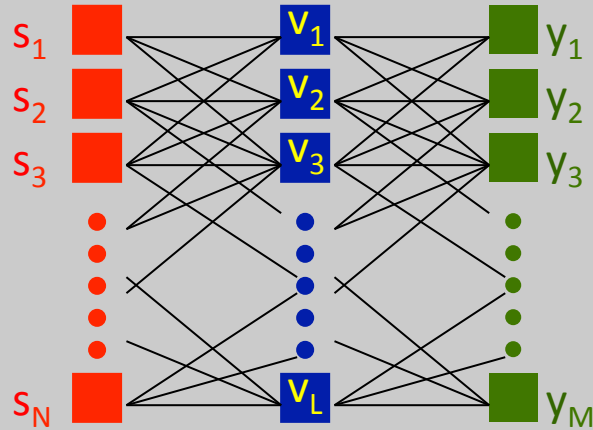
Uniqueness of Mapping

$$\omega_i \Leftrightarrow f_i \quad i = 1, 2, \dots, M_{sp}$$

Density  $\rho(\omega)=1$

$$V_0 = \int_{\omega} d\omega \rho(\omega) = \sum_{i=1}^{M_{sp}} \rho(\omega) = M_{sp}$$

## Multiple Perceptron:



$$s_i \in \{-1, 1\}, i = 1, 2, \dots, N$$

$$v_k \in \{-1, 1\}, k = 1, 2, \dots, L$$

$$y_j \in \{-1, 1\}, j = 1, 2, \dots, M$$

$$w_{ik}^1 \in (-1, 0, 1) \equiv Z_3, i = 1, 2, \dots, N; j = 1, 2, \dots, L$$

$$w_{kj}^2 \in (-1, 1) \equiv Z_2, i = 1, 2, \dots, L; j = 1, 2, \dots, M$$

$$v_k = f\left(\sum_i^N w_{ik}^1 s_i\right)$$

$$y_j = f\left(\sum_k^L w_{jk}^2 v_k\right)$$

## System Entropy

$$R_0(f_i) = \frac{1}{M_{sp}}$$

$$S_0 = \ln_2 M_{sp}$$

$$S_0 = \ln_2 \left( \frac{3^{NL} 2^{ML}}{L!} \right)$$

Steirling:  $x! = (2\pi x)^{-1/2} x^x e^{-x}$

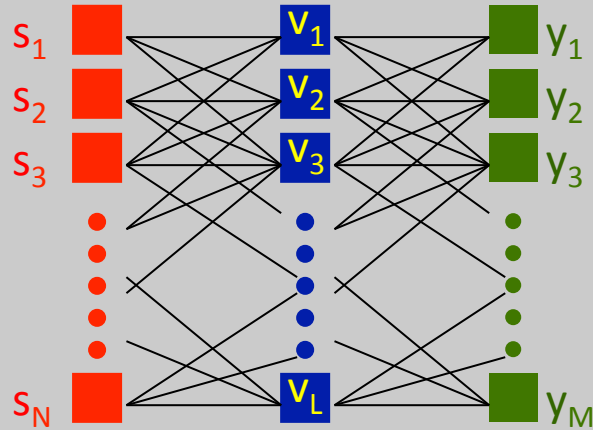
$$S_0 = NL \ln_2 3 + ML - \ln_2 (L!)$$

$$\ln_2 (L!) = \ln_2 \sqrt{2\pi L} + L \ln_2 \left( \frac{L}{e} \right)$$

$$S_0 = NL \ln_2 3 + ML - L \ln_2 \left( \frac{L}{e} \right)$$



## Multiple Perceptron:



$$s_i \in \{-1, 1\}, i = 1, 2, \dots, N$$

$$v_k \in \{-1, 1\}, k = 1, 2, \dots, L$$

$$y_j \in \{-1, 1\}, j = 1, 2, \dots, M$$

$$w_{ik}^1 \in (-1, 0, 1) \equiv Z_3, i = 1, 2, \dots, N; k = 1, 2, \dots, L$$

$$w_{kj}^2 \in (-1, 1) \equiv Z_2, k = 1, 2, \dots, L; j = 1, 2, \dots, M$$

$$v_k = f\left(\sum_i^N w_{ik}^1 s_i\right)$$

$$y_j = f\left(\sum_k^L w_{jk}^2 v_k\right)$$

The amount of information needed to set a rule  $f$  is then:

$$S_0 = L \left( N \ln_2 3 + M - \ln_2 \left( \frac{L}{e} \right) \right)$$

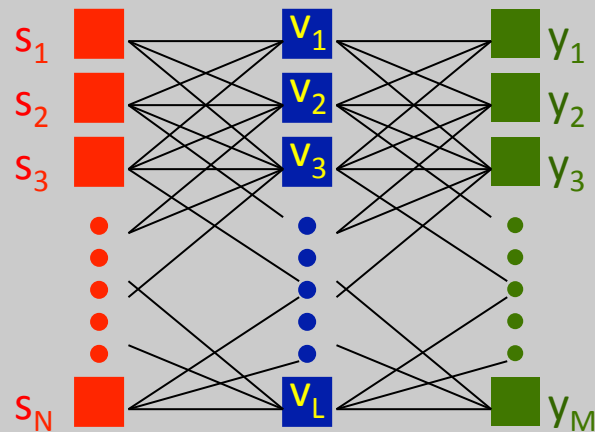
Supervised learning: first pattern

$$\bar{Y}^1 = \bar{f}(\bar{s}^1)$$

$$V_1(\bar{f}) = \int d\omega \rho(\omega) \prod_{\mu=1}^p I(f_\omega, \bar{s}^\mu) = \sum_{i=1}^{M_{sp}} I(f_\omega, \bar{s}^1)$$

$$V_1(f^k) = \sum_{i=1}^{M_{sp}} \theta_{f^k}(\omega) I(f_\omega, \bar{s}^1)$$

# Multiple Perceptron:



$$s_i \in \{-1, 1\}, i = 1, 2, \dots, N$$

$$v_k \in \{-1, 1\}, k = 1, 2, \dots, L$$

$$y_j \in \{-1, 1\}, j = 1, 2, \dots, M$$

$$w_{ik}^1 \in (-1, 0, 1) \equiv Z_3, i = 1, 2, \dots, N; k = 1, 2, \dots, L$$

$$w_{kj}^2 \in (-1, 1) \equiv Z_2, k = 1, 2, \dots, L; j = 1, 2, \dots, M$$

$$v_k = f\left(\sum_i^N w_{ik}^1 s_i\right)$$

$$y_j = f\left(\sum_k^L w_{jk}^2 v_k\right)$$

Supervised learning: first pattern

$$R_1(f^k) = \frac{V_p(f^k)}{V_p(\bar{f})}$$

$$S_1 = -\sum_{f^k} R_1(f^k) \ln_2(R_1(f^k))$$

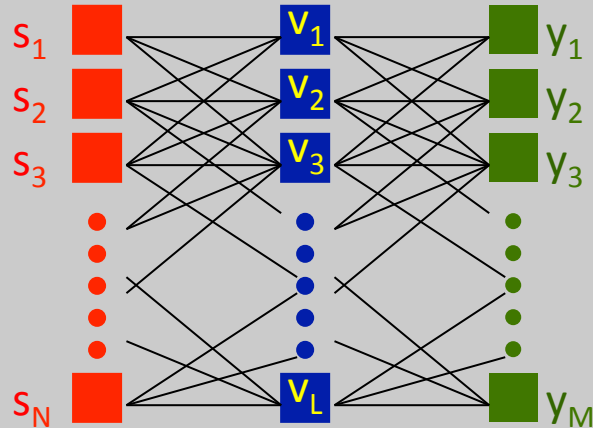
It is hard to determine  $S_1$ ! We must determine the fraction  $f^k$  of functions that have the imposed condition in common.

Reasoning in a probabilistic way !

I have M output neurons and a probability equal to  $\frac{1}{2}$  of a correct answer. The change in entropy is:

$$S_1 = -M \ln_2 \frac{1}{2} = M$$

## Multiple Perceptron:



$$s_i \in \{-1, 1\}, i = 1, 2, \dots, N$$

$$v_k \in \{-1, 1\}, k = 1, 2, \dots, L$$

$$y_j \in \{-1, 1\}, j = 1, 2, \dots, M$$

$$w_{ik}^1 \in (-1, 0, 1) \equiv Z_3, i = 1, 2, \dots, N; k = 1, 2, \dots, L$$

$$w_{kj}^2 \in (-1, 1) \equiv Z_2, k = 1, 2, \dots, L; j = 1, 2, \dots, M$$

$$v_k = f\left(\sum_i^N w_{ik}^1 s_i\right)$$

$$y_j = f\left(\sum_k^L w_{jk}^2 v_k\right)$$

Supervised learning:  $p$  patterns

$$S_p = pM$$

The number of patterns required to set the rule  $f$  is

$$S_{p^*} = S_0$$

$$p^* = \frac{L}{M} \left( N \ln_2 3 + M - \ln_2 \left( \frac{L}{e} \right) \right) < 2^N$$

Purely statistical discourse: it is said that a given set of  $p^*$  stimuli fixes the rule  $f$ . The argument is valid by averaging over a set of  $n$  sets of  $p^*$  stimuli.

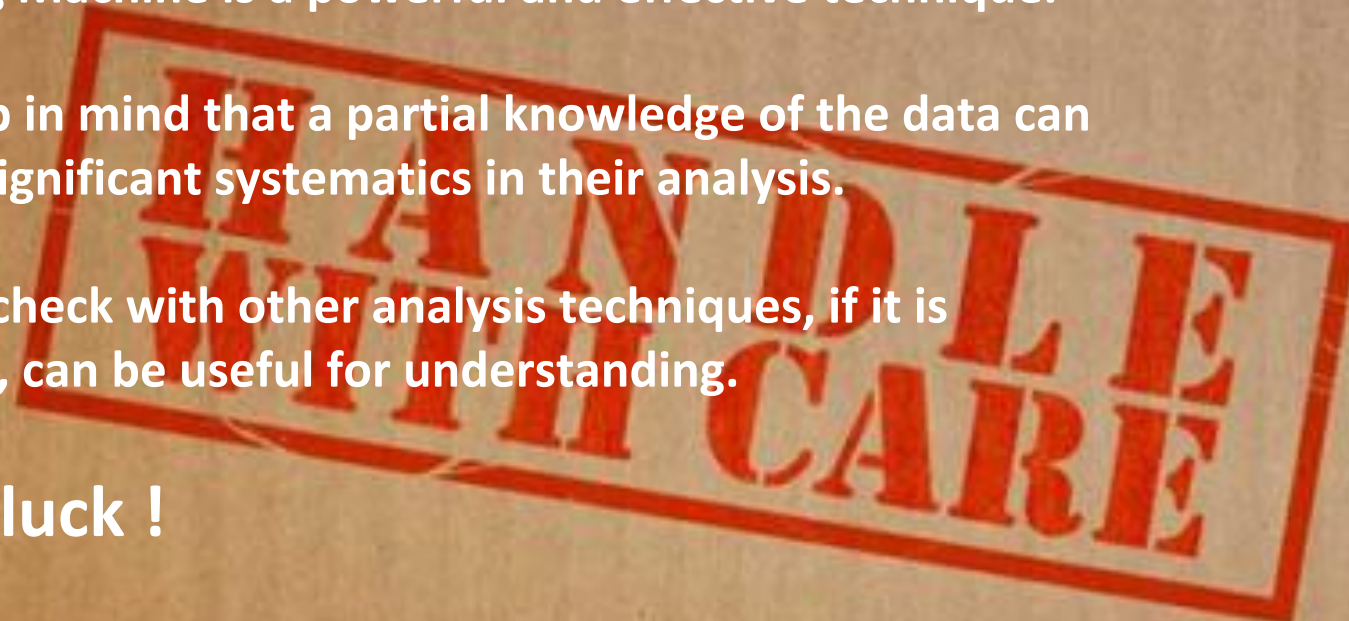
## Some Conclusions ?

Learning Machine is a powerful and effective technique.

But keep in mind that a partial knowledge of the data can induce significant systematics in their analysis.

A cross-check with other analysis techniques, if it is possible, can be useful for understanding.

Good luck !



*More Slides. Forecasting in light of Big Data*

# Big Data:

## Is it really a scientific revolution ?

### FORECASTING IN LIGHT OF BIG DATA

HYKEL HOSNI AND ANGELO VULPIANI

ABSTRACT. Predicting the future state of a system has always been a natural motivation for science and practical applications. Such a topic, beyond its obvious technical and societal relevance, is also interesting from a conceptual point of view. This owes to the fact that forecasting lends itself to two equally radical, yet opposite methodologies. A reductionist one, based on the first principles, and the naïve-inductivist one, based only on data. This latter view has recently gained some attention in response to the availability of unprecedented amounts of data and increasingly sophisticated algorithmic analytic techniques. The purpose of this note is to assess critically the role of *big data* in reshaping the key aspects of forecasting and in particular the claim that *bigger* data leads to *better* predictions. Drawing on the representative example of weather forecasts we argue that this is not generally the case. We conclude by suggesting that a clever and context-dependent compromise between modelling and quantitative analysis stands out as the best forecasting strategy, as anticipated nearly a century ago by Richardson and von Neumann.

*Nothing is more practical than a good theory* (L. Boltzmann)

[arXiv:1705.11186v1](https://arxiv.org/abs/1705.11186v1) [physics.soc-ph] 31 May 2017



## FORECASTING IN LIGHT OF BIG DATA

Big data spans radically diverse domains. This, together with its sodality with machine learning, has recently been fuelling an all-encompassing enthusiasm, which is loosely rooted on a twofold presupposition. First, the idea that **big data will lead to much better forecasts**. Second, it will do so across the board, from scientific discovery to medical, financial, commercial and political applications.

The main lesson can be put as follows: as anticipated nearly a century ago by Richardson and von Neumann, **a clever and context-dependent trade-off between modeling and quantitative analysis** stands out as **the best strategy for meaningful prediction**.

Whilst Anderson's argument fails to stand methodological scrutiny, as the present paper recalls, its key message –big data enthusiasm– has clearly percolated society at large. **This may lead to very serious social and ethical shortcomings**. For the combination of statistical methods and machine learning techniques for predictive analytics is currently finding cavalier application in **a number of very sensitive intelligence and policing activities**.

In particular we ask whether using our knowledge of the past states of a system – and without the use of models for the evolution equation – **meaningful predictions about the future are possible**. Our answer is negative to the extent that rather severe difficulties are immediately found, even in a very abstract and simplified situation. As we shall point out the most difficult challenge to this view is understanding of the **“proper level”** of abstraction of the system. We will see there that the key to understanding the **“proper level”** of abstraction lies **with identifying the “relevant variables” and the effective equations which rule their time evolution**.

# FORECASTING IN LIGHT OF BIG DATA

An extreme inductivist approach to forecasting using Big Data

According to a vaguely defined yet rather commonly held view big data may lead to dispense with theory, modeling or even hypothesizing.

All of this would be encompassed, across domains, by smart enough machine learning algorithms operating on large enough data sets.

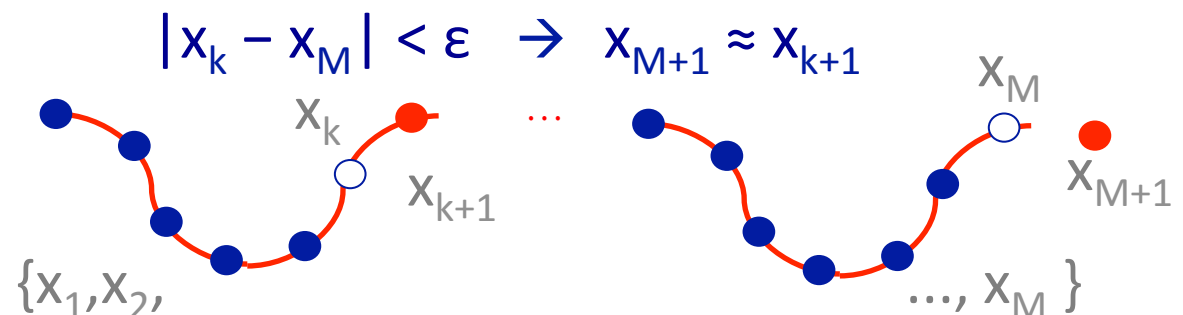
We are interested in forecasts such that future states of a systems are predicted solely on the basis of known past states.

Two hypotheses are needed to give an affirmative answer:

- Similar premises lead to similar conclusions (Analogy);
- Systems which exhibit a certain behavior, will continue doing so (Determinism ).

In more formal terms, given the series  $\{x_1, \dots, x_M\}$ , where  $x_j$  is the vector describing the state at time  $j\Delta t$ , we look in the past for an analogous state, that is a vector  $x_k$  with  $k < M$  “near enough” (i.e.  $|x_k - x_M| < \varepsilon$ , being  $\varepsilon$  the desired degree of accuracy).

Once we find such a vector, we “predict” the future at times  $M + n > M$  by simply assuming for  $x_{M+n}$  the state  $x_{k+n}$ . It all seems quite easy, but it is not at all obvious that an analog can be found.



# FORECASTING IN LIGHT OF BIG DATA

An extreme inductivist approach to forecasting using Big Data

## Poincare' recurrence theorem

After a suitable time, a deterministic system with a bounded phase space returns to a state near to its initial condition.

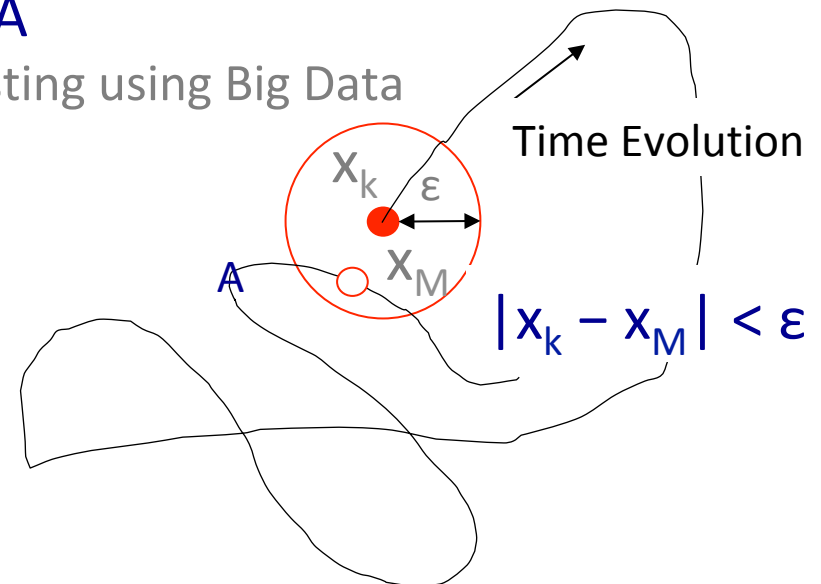
Thus an analog surely exists. How long do we have to go back to find it?

Kac who proved a Lemma to the effect that the average return time in a region A is proportional to the inverse of the probability  $P(A)$  that the system is in A.

$$\langle T_R \rangle \propto \frac{1}{P(A)}$$

Consider a system of dimension D. The probability  $P(A)$  of being in A is

$$P(A) \propto \varepsilon^D \mapsto \langle T_R \rangle = O(\varepsilon^{-D})$$



$$D \approx 10$$

$$\varepsilon \approx 0.01$$

$$\langle T_R \rangle = O(10^{20})$$

The return time is so large that in practice a recurrence is never observed.

So the required analog, whose existence is guaranteed in theory, sometimes cannot be expected to be found in practice, even if complete and precise information about the system.

# FORECASTING IN LIGHT OF BIG DATA

## An extreme inductivist approach to forecasting using Big Data

In many sciences and in engineering, an ever increasing gap between theory and experiment can be observed. This gap tends to widen particularly in the presence of complex features in natural systems science.

In socio-economical systems the gap between data and our scientific ability to actually understanding them is typically enormous. Surely the availability of huge amounts of data, sophisticated methods for its retrieval and unprecedented computational power available for its analysis will undoubtedly help moving science and technology forward.

But in spite of a persistent emphasis on a fourth paradigm (beyond the traditional ones, i.e. experiment, theory and computation) based only on data, there is as yet no evidence data alone can bring about scientifically meaningful advance.

To the contrary, as nicely illustrated by Crutchfield, up to now it seems that the unique way to understand some non trivial scientific or technological problem, is following the traditional approach based on a clever combination of data, theory (and/or computations), intuition and wise use of previous knowledge. Similar conclusions have been reached in the computational biosciences.

P. V. Coveney et al. point out very clearly not only the methodological shortcomings (and ineffectiveness) of relying on data alone, but also unfold the implications of methodologically unwarranted big data enthusiasm for the allocation of research funds to healthcare related projects: “A substantial portion of funding used to gather and process data should be diverted towards efforts to discern the laws of biology”.

# FORECASTING IN LIGHT OF BIG DATA

## An extreme inductivist approach to forecasting using Big Data

Big data undoubtedly constitute a great opportunity for scientific and technological advance, with a potential for considerable socio-economic impact. To make the most of it, however, the ensuing developments at the interface of statistics, machine learning and artificial intelligence, must be coupled with adequate methodological foundations. Not least because of the serious ethical, legal and more generally societal consequence of the possible misuses of this technology.

This note contributed to elucidating the terms of this problem by focusing on the potential for big data to reshape our current understanding of forecasting. To this end we pointed out, in a very elementary setting, some serious problems that the naive inductivist approach to forecast must face: the idea according to which reliable predictions can be obtained solely on the grounds of our knowledge of the past faces insurmountable problems – even in the most idealized and controlled modeling setting.

We therefore conclude that the big data revolution is by all means a welcome one for the new opportunities it opens.

However the role of modeling cannot be discounted.