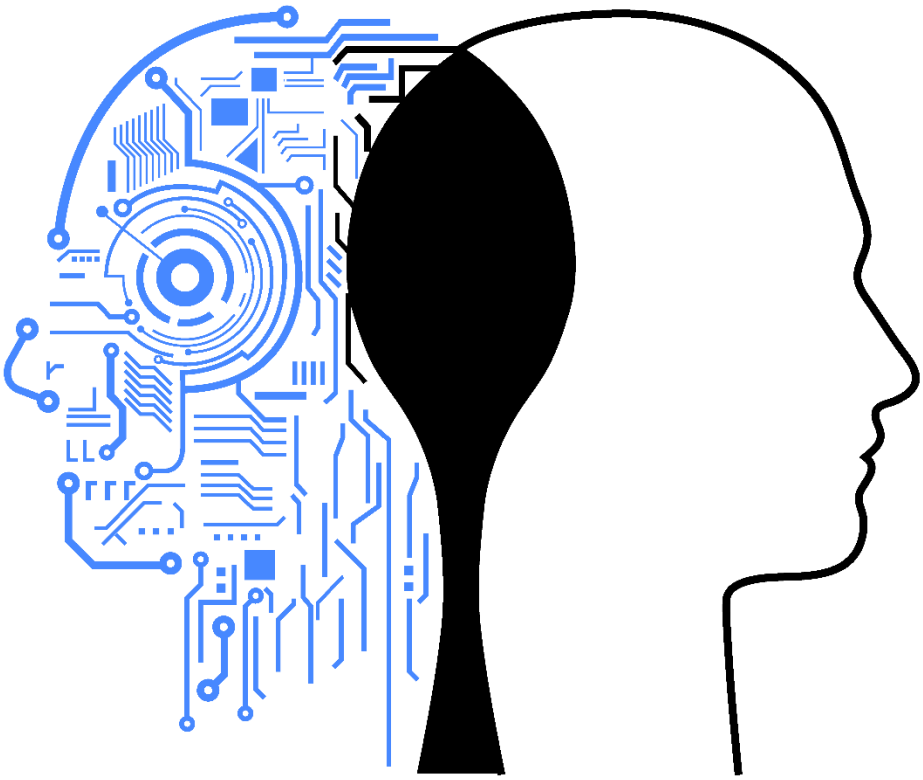
Liceo Scientifico Edouard Bérard, Aosta

Esame di Stato 2017

Hal vs Einstein

Mathematical models of neural networks



by

Matteo Pariset

Gli esseri umani sono portati
a riprodurre per capire e
capire per riprodurre

Ernst Mach

Abstract

Giving birth to computers, the last century fed the human dream of creating an artificial mind. Fifty years of staggering software evolution only brought poor results: we could not even match the abilities of a human beings in their early stages of life. But a different conception of the nature of our intelligence opens whole new horizons.

Contents

1 Introduction

The long walk to build a complete artificial brain takes its origin from the most inaccessible questions about existence. It is shaped by the profound meaning we assign to our freedom, our destiny and our soul. It must go through the contradictions of philosophical models of the mind to reach, at last, the insidious land of working prototypes.

Maybe artificial intelligence always made men curious precisely because it is far more than an engineering problem. It is an opportunity to gain a deeper insight on this world.

The first stop of the ambitious journey of AI is the close observation of nature, to reveal and imitate its alchemies. But, in the attempt of using our minds to understand themselves, the ultimate goal of this search comes to light: the reproduction of the noblest expression of intelligence, self-awareness.

Chapters 2 to 4 briefly present the main concepts in AI, whereas the following ones thoroughly describe Vanilla Networks.

2 The philosophy of intelligence

a. What does intelligent mean?

Before dealing with the evolution of AI, we need some sort of definition of intelligence. Even though the question appears to be easy, no exhaustive definition is available, because of its excessive generality or – on the contrary – its restraining specificity. Here however we only limit ourselves to list three fundamental aspects of it:

1. The ability to interact (acquire information and formulate reactions) with the environment;
2. The ability to carry out with success – or, at least, take steps towards – a pre-set aim;
3. Some degree of self-consciousness or, in other words, the awareness of being an agent and not an effect.

Having in mind the first two features, Alan Turing formulated his famous intelligence test. Turing essentially asserted that if a computer can carry on a conversation with humans without letting them realize its nature, then it can be considered intelligent.

b. Strong AI vs. Weak AI

The conceptual foundations of Turing Test, however, are not universally accepted. In fact, this type of assessment does not consider the third property of our definition of intelligence and can – at least in theory – be passed by an expert system.

Some critics believe that it is doomed simply because of its strong anthropocentrism. They accuse it of establishing an undue correspondence between human intelligence and its artificial counterpart. Explaining this with Bertrand Russell's words, it would be as if aeronautical engineering texts described the goal of their field as *making machines that fly so exactly like pigeons that they can fool other pigeons*. In this perspective, it is not possible to evaluate an intelligent being according to its resemblance to a human being.

A clever argument showing the problem has been proposed by the philosopher John Searle and is known as the Chinese Room Test. If a computer software persuades a Chinese speaker that it is too a Chinese speaker, then its set of instructions can be manually executed by someone that does not speak Chinese. While being able to write well-formed Chinese sentences, he would still be completely stranger to this language.

According to Searle, without intentionality – what we usually call *understanding* – there is no intelligence. This implies that no deterministic procedure can never be as a true human mind – or, using Searle's language, strong AI is no more than a dream.

What could still be possible, however, are forms of weak AI, i.e. extra-human intelligence not resembling humans at all but still useful for some specific tasks.

3 Functionalism and its limits

One of the historically relevant scientific approaches to the AI problem is called functionalism and was proposed by the American mathematician Hilary Putnam.

Mental events are defined as functions, i.e. operational roles. Providing that one reconstructs the same relationships and input/output systems, any physical support – either neurons or Swiss cheese – could be in principle a mind.

Amid all the optimism coming from the construction of the first computers, the brain was thought as no more than a (very powerful) one. Only the logical processes (the "software" running on the

brain) did matter, the hardware (the physical brain) did not.

A well-known success of this line of research is represented by Deep Blue, the software which could defeat the world chess champion Garry Kasparov in 1996.

In spite of some remarkable steps forward, however, many simple human abilities – like voice recognition – could not be imitated. No software could even hold a conversation worthy of a five years old child.

4 Connectionism

The lack of convincing results from functionalism shifted the focus to a different perspective: connectionism. The brain was no more the physical place in which abstract functions could operate but the true essence of intelligence. This huge network of connections between neurons – in which billions of chemical reactions take place every second – is what makes possible the miracle of understanding.

An artificial equivalent, hence, should emulate the biological features of the brain. Neural networks took the stage and their nature will be discussed from now on.

5 An artificial brain

a. First types of neurons

In 1943, McCulloch and Pitts had already elaborated the first model of a binary neuron, a component that could be activated – like human neurons – when its synaptic inputs exceeded a certain threshold.

Many different binary neurons could be interconnected, forming a so-called neural network. The only missing part was a law providing some sort of learning.

b. The Perceptron

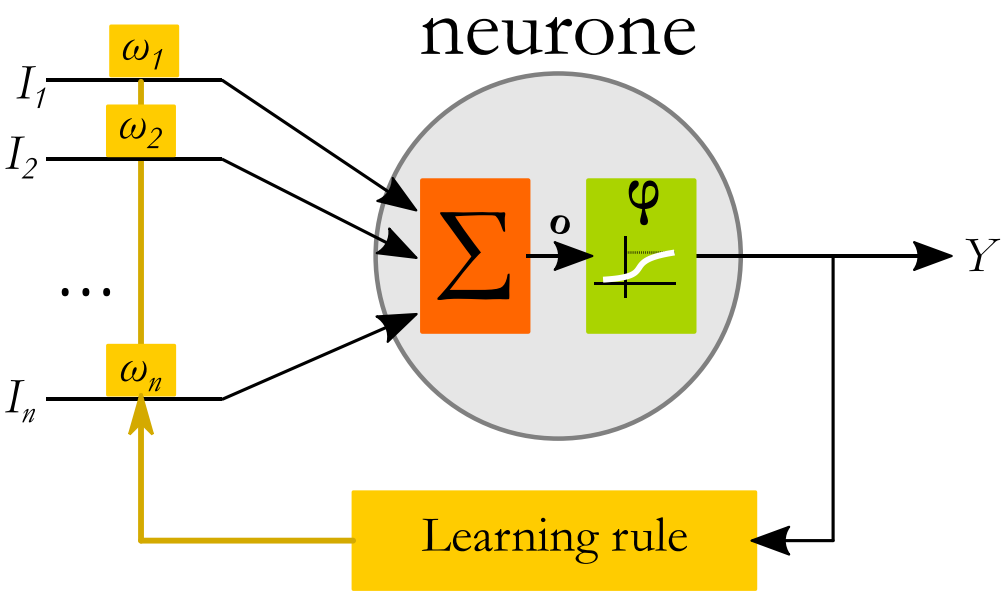
Taking advantage from the intuition of the psychologist Donald Hebb, that learning does not create new links among neurons but only modifies their electrical conductivity, Frank Rosenblatt conceived an evolution algorithm for neurons. The perceptron was born.

The core idea behind it was to provide the neuron with a way to check its own output against the desired one and progressively correct its synaptic weights to reduce the gap.

c. Sigmoid neurons

To increase the performances of binary neurons, models that output real values were created. Here we examine sigmoid neurons even though recent work has shown that they are not the most efficient ones.

A sigmoid neuron assigns to its output a value in the (real) interval [0, 1]. In the picture below, such a value is $\varphi(o)$, where φ is known as activation function and o is the weighted means of inputs (I_1, ..., I_n) according to their synaptic weights ω.



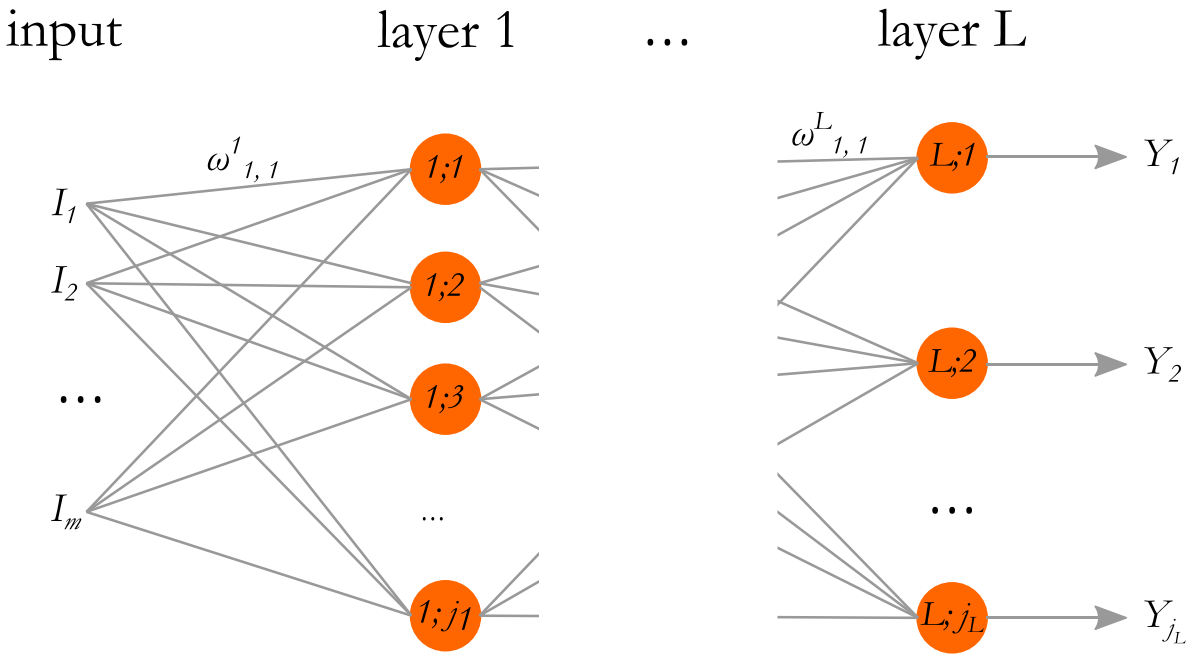
6 Vanilla Networks

Neural networks that make use of sigmoid neurons (or similar) and respect some architectural constraints, for instance providing some notion of layer, are called Multilayer Perceptrons or Vanilla Neural Networks.

Their peculiar structure – especially the connection of each neuron in a layer (uniquely) with every neuron of the following layer – provides the sufficient regularity to design a mathematical method to train the artificial brain to perform complex tasks. Such method is known as Back-Propagation Algorithm and its name, along with its structure, will become clear in the next section.

7 Back-Propagation Algorithm in detail

The following picture shows a Multilayer Perceptron with L layers, with the output of the last layer of neurons acting as answer of the brain. For instance, in an application where the artificial brain must perform face recognition one could establish that each neuron of the output layer corresponds to a different person.



a. Overview

First the network is initialized with random synaptic weights. Then, given a training input, the output of the brain Y is compared to the desired one $\boldsymbol{T}$.

The aim is adjusting the synaptic weights $\boldsymbol{\omega}$ up to minimizing the error made by the network.

b. Analysis

We consider a network taking a vector $\boldsymbol{I}$ of m components as input and returning $\boldsymbol{Y}$ with N^L components. The table below groups the symbols used next.

Symbol	Meaning
L	Number of layers composing the neural network
l	Index of the layer
N^l	Number of neurons in layer l
I_j	j-th component of the input, with $j = \{1, 2, \dots, m\}$
net_i^l	Weighted sum of the input of the i-th neuron in layer l, with $i = \{1, 2, \dots, N^l\}$
O_i^l	Output of the i-th neuron in layer l, with $i = \{1, 2, \dots, N^l\}$
Y_n	n-th component of the network output (layer L), with $n = \{1, 2, \dots, N^L\}$
φ	Activation function of the neuron

ω_{ji}^{l}	Synaptic weight of the link between neuron i in layer $l-1$ and neuron j in layer l
δ_{i}^{l}	Partial derivative of the error with respect to the weighted sum of the inputs of the i-th neuron in layer l
T_j	j-th component of the training vector T, containing the desired output of the network for a specific input I
g_{ji}^{l}	Partial derivative of the error with respect to the weight ω_{ji}^{l}
η	Learning coefficient

As explained above, here we use sigmoid neurons even though many different models exist.

As the activation function φ we choose the standard logistic function because:

1. its first derivative can be easily calculated
2. the value of its derivative changes according to the absolute value of the distance of its argument from $x = 0$;

In particular, we take:

$$\varphi(x) = \frac{1}{1 + e^{-x}}$$

And hence

$$\varphi'(x) = \frac{e^{x}}{(1 + e^{x})^{2}} = \varphi(x)\big(1 - \varphi(x)\big) \qquad [\text{E. } 1]$$

Consequently, the output of the neuron j in layer l will be:

$$O_j^l = \varphi\left(net_j^l\right) = \varphi\left(\sum_{i=0}^{N^{l-1}} \left(O_i^{l-1}\omega_{ji}^l\right)\right)$$

Given that the result is the output of the last layer of the network, then $Y_j = O_j^L$.

We can now define an expression quantifying the error made by the network, for instance:

$$E(T,Y) = \frac{1}{2}\sum_{i=0}^{N^L}(T_i - Y_i)^2$$

Where the only purpose of the coefficient ½ is to avoid multiplicative factors in E'. Clearly, we could have chosen many other expressions as the error.

The aim is to minimize the value of this function.

If Y depended on a single synaptic weight ω, the problem would amount to find the minimum of a parabola. But, since the variables (the weights) are $N^l \cdot N^{l-1} + \cdots + N^1 \cdot N^0$, we should locate an absolute minimum of a surface in $N^l \cdot N^{l-1} + \cdots + N^1 \cdot N^0 + 1$ dimensions. However, by conceding that we might content ourselves with a relative minimum instead that the absolute one(s), we can avoid analytic means and profit from gradient descent.

We observe, nevertheless, that the previous concession adds a certain unpredictability to the outcome of the learning process of our network. The search (by gradient descent) is, in fact, influenced by the starting point – which is chosen at random (see above). In other words, some artificial brains will prove better or worse than others at learning the same tasks (interestingly, in the same way as human brains do).

The value of the correction $\Delta\omega_{ji}^{L}$ of every weight is then given by the negative of the partial derivative of the error with respect to the weight, multiplied by an arbitrary factor η. This factor does not have to be constant and is necessary to adjust the speed of the evolution of the network. Hence:

$$\Delta\omega_{ji}^{L} = -\eta \frac{\partial E}{\partial \omega_{ji}^{L}} \qquad [E.2]$$

We must now evaluate the partial derivate in the formula above. It turns out that, for the last layer of neurons, this is an easy task. In fact, applying twice the chain rule of differentiation we obtain:

$$\frac{\partial E}{\partial \omega_{ji}^{L}} = \frac{\partial E}{\partial O_{j}^{L}} \frac{\partial O_{j}^{L}}{\partial net_{j}^{L}} \frac{\partial net_{j}^{L}}{\partial \omega_{ji}^{L}} \qquad [E.3]$$

Below, we rewrite in useful forms each of the three factors in E.3:

$$\frac{\partial net_j^L}{\partial \omega_{ji}^L} = \frac{\partial}{\partial \omega_{ji}^L}\left(\sum_{k=0}^{N^{L-1}} \left(O_k^{L-1}\omega_{jk}^L\right)\right) = \frac{\partial}{\partial \omega_{ji}^L}\left(O_i^{L-1}\omega_{ji}^L\right) = O_i^{L-1}$$

$$\frac{\partial O_j^L}{\partial net_j^L} = \frac{\partial}{\partial net_j^L}\varphi\left(net_j^L\right) = \varphi\left(net_j^L\right)\left(1 - \varphi\left(net_j^L\right)\right) \quad \text{(by } E.1\text{)}$$

$$\frac{\partial E}{\partial O_j^L} = \frac{\partial E}{\partial Y_j} = \frac{\partial}{\partial Y_j}\left(\frac{1}{2}(T_j - Y_j)^2\right) = \frac{1}{2}2\left(T_j - Y_j\right)(-1) = Y_j - T_j$$

If we call δ_j^L the sensitivity of the output of neuron j with respect to the weighted sum of its inputs, then:

$$\delta_j^L = \frac{\partial E}{\partial O_j^L}\frac{\partial O_j^L}{\partial net_j^L} = \left(Y_j - T_j\right)Y_j\left(1 - Y_j\right)$$

We can rewrite the correction for a weight as:

$$\Delta\omega_{ji}^L = -\eta\frac{\partial E}{\partial \omega_{ji}^L} = -\eta\delta_j^L\frac{\partial net_j^L}{\partial \omega_{ji}^L} = -\eta Y_j\left(1 - Y_j\right)\left(Y_j - T_j\right)O_i^{L-1}$$

The computation of the partial derivative in E.3 for neurons in inner layers is less straight-forward

because we do not know the desired output T_j^l of any neuron. For brevity, we omit the proof.

The expression we obtain for δ_j^l if $l \in \{1, 2, \dots, L-1\}$ is:

$$\delta_j^l = O_j^l\left(1 - O_j^l\right) \sum_{k=0}^{N^{l+1}} \delta_k^{l+1}\, \omega_{kj}^{l+1}$$

As we can see, the value of δ for a neuron belonging to an inner layer l is calculated from the ones of neurons in layer $l+1$. In other words, the evolution of the network is computed backwards, beginning from the output layer. This feature is the one that determines the name of the algorithm.

If the network is not yet sufficiently precise at the end of the first cycle of learning, another one is performed. In general, this procedure is repeated several thousands of times before obtaining the expected results.

The learning process is considered concluded if

$$\left|g_{ji}^L\right| = \left|\frac{\partial E}{\partial \omega_{ji}^L}\right| \leq g_{MIN}$$

$$for\ \forall j\ \in \{1, 2, \dots, N^L\}\ and\ \forall i\ \in \{1, 2, \dots, N^{L-1}\}$$

or the number of cycles exceeds a chosen threshold.

Bibliography

- Abbagnano, Fornero, *L'ideale e il reale 3*, Paravia, 2013
- Belda, *Menti, macchine e matematica*, RBA, 2011
- Domingos, *L'algoritmo Definitivo*, Bollati Boringhieri, 2016
- Gangemi, Miceli, Sprini, *L'intelligenza, Teorie e modelli*, Editori Laterza, 2003
- Lerda, *Intelligenza umana e intelligenza artificiale, est modus in rebus*, Rubbettino Editore, 2002
- Minsky, *Semantic Information Processing*, MIT Press, 2015
- Nilsson, *Intelligenza artificiale*, Apogeo, 2005
- Searle, *Menti, Cervelli e Programmi*, Clupguide, 1984
- Turing, *La filosofia degli automi. Origini dell'intelligenza artificiale*, Paolo Boringhieri, 1986; trad. it. di *Computing machinery and intelligence*, Mind, 1950

Webography

- https://en.wikipedia.org/wiki/Hopfield_network, 01/05/2017
- https://en.wikipedia.org/wiki/Perceptron, 01/05/2017
- https://en.wikipedia.org/wiki/Multilayer_perceptron, 01/05/2017
- https://en.wikipedia.org/wiki/Action_potential, 05/05/2017
- https://en.wikipedia.org/wiki/Hodgkin%E2%80%93Huxley_model, 06/05/2017
- http://www.cs.stir.ac.uk/courses/ITNP4B/lectures/kms/2-Perceptrons.pdf, 06/05/2017
- https://en.wikipedia.org/wiki/Logistic_function, 06/05/2017
- https://en.wikipedia.org/wiki/Hyperbolic_function, 08/05/2017
- https://en.wikipedia.org/wiki/Deep_learning#Deep_neural_networks, 08/05/2017
- https://en.wikipedia.org/wiki/Backpropagation, 08/05/2017

- http://neuralnetworksanddeeplearning.com/chap2.html, 09/05/2017
- http://wwwold.ece.utep.edu/research/webfuzzy/docs/kk-thesis/kk-thesis-html/node22.html, 15/05/2017
- http://numericinsight.com/uploads/A_Gentle_Introduction_to_Backpropagation.pdf, 15/05/2017
- https://en.wikipedia.org/wiki/Root_mean_square, 15/05/2017
- https://en.wikipedia.org/wiki/Partial_derivative, 15/05/2017
- https://en.wikipedia.org/wiki/Total_derivative, 15/05/2017
- http://www.vetta.org/documents/A-Collection-of-Definitions-of-Intelligence.pdf, 22/05/2017
- https://pdfs.semanticscholar.org/5f2f/031809a4b95dd9de0218648677fcf0999975.pdf, 24/05/2017
- http://didattica.uniroma2.it/assets/uploads/corsi/145317/Turing_Macchine_calcolatrici_ed_intelligenza.pdf, 19/06/2017