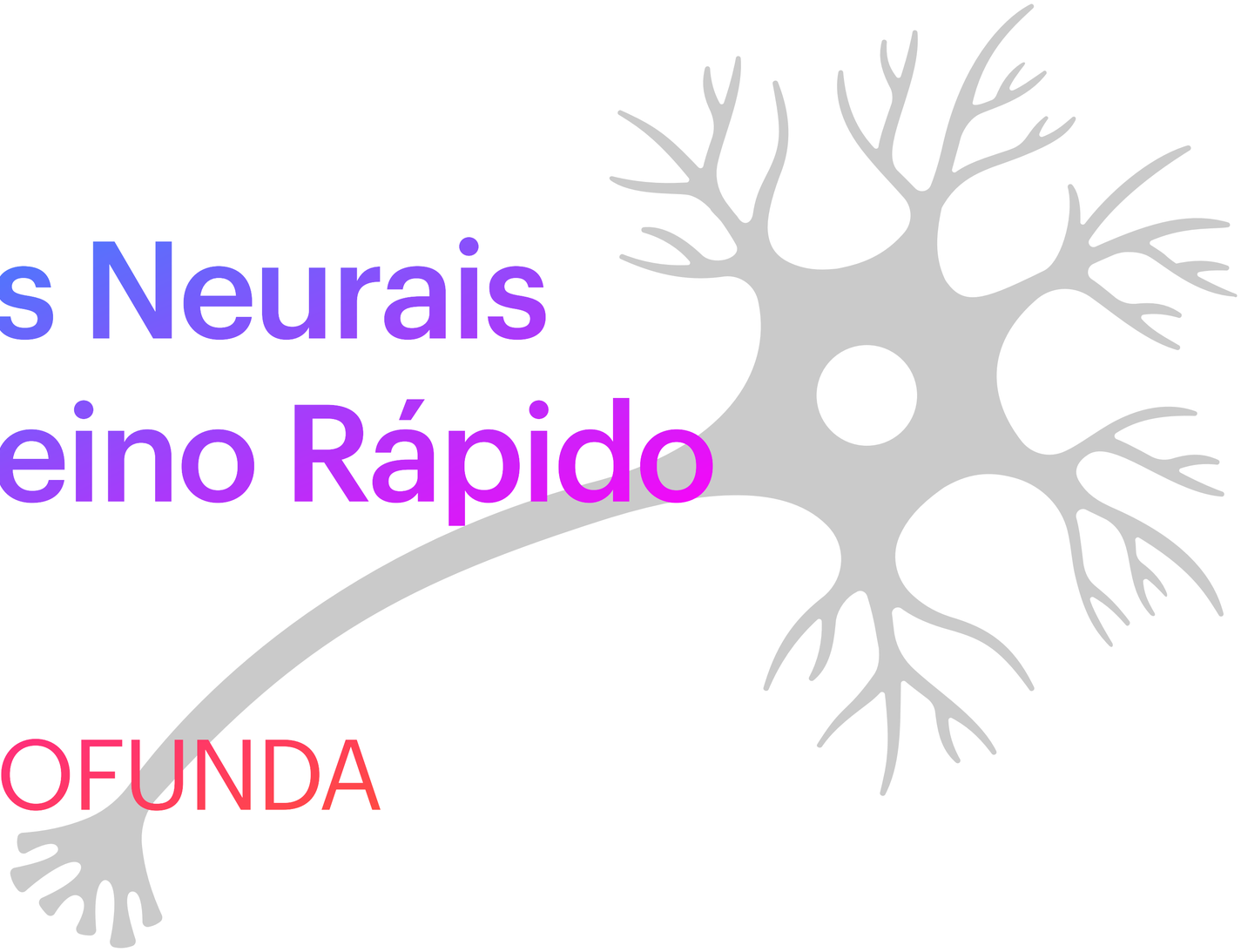
INSTITUTO FEDERAL
Ceará

Programa de Pós-Graduação
em Ciência da Computação

07 Redes Neurais De Treino Rápido



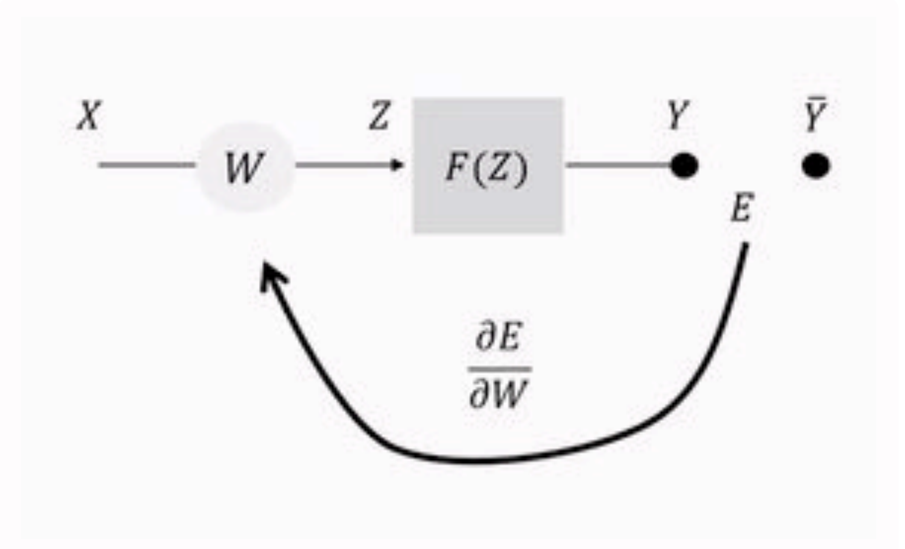
APRENDIZAGEM PROFUNDA

PPGCC – 2023.1

Prof. Saulo Oliveira <saulo.oliveira@ifce.edu.br>

DESVATAGENS DO ALGORITMO DE RETROPROPAGAÇÃO DO ERRO

- Consomem grande tempo de treinamento devido ao ajuste iterativo dos parâmetros;
- Lentidão na convergência quando a taxa de aprendizagem é pequena. Ao passo que pode ocasionar divergência (instabilidade) quando a taxa é de aprendizagem é alta;
- Não há garantias de que o algoritmo pare em um mínimo global;
- Redes Neurais podem ser super-treinadas com o algoritmo BP de maneira que a generalização fique prejudicada (sobeajuste – *overfitting*).



MÉTODO DOS MÍNIMOS QUADRADOS

HISTÓRIA

- Carl Friedrich Gauss, em 1795, aos 18 anos;
- Demonstração de um método para determinar um parâmetro desconhecido de uma equação via minimização da soma dos quadrados dos resíduos;
- Mais tarde, em 1805, Adrien-Marie Legendre chamou este método de mínimos quadrados (*méthode des moindres carrés*);
- Problemas possuem essa "cara": $L(\boldsymbol{\varepsilon}) \equiv \sum_{i=1}^{\infty} (\varepsilon_i)^2.$
- Nas redes, tratamos o problema como um problema de minimização:

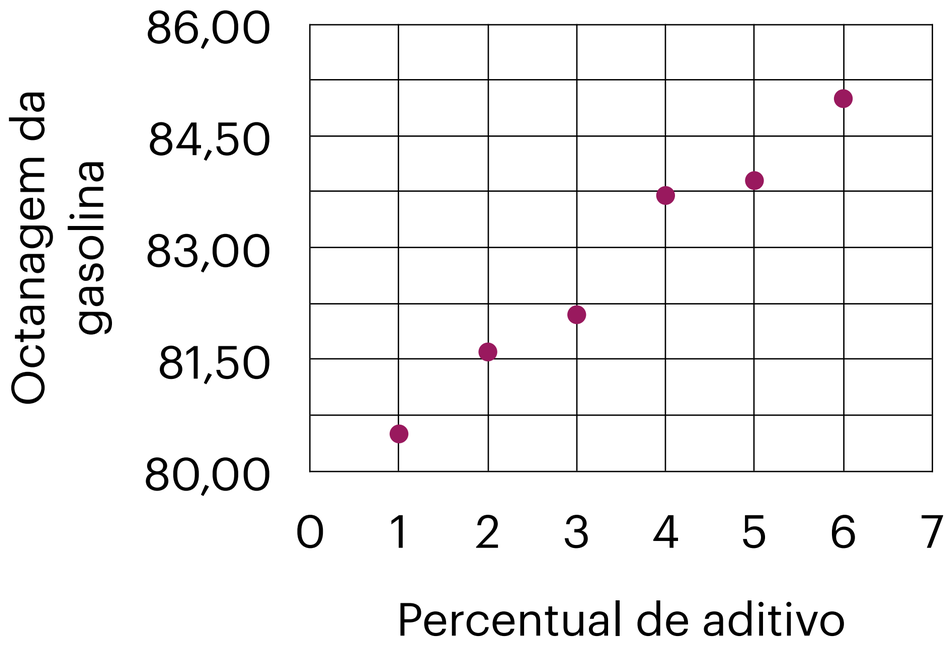
$$\min_{\mathbf{w}} L(\boldsymbol{\varepsilon}) = \sum_{i=1}^{N} \varepsilon_i^2 = \sum_{i=1}^{N} \left(y_i - f(\mathbf{x}_i) \right)^2 = \sum_{i=1}^{N} \left(y_i - \mathbf{w}^\intercal \mathbf{x}_i \right)^2$$

EXPERIMENTO 01

Considere um experimento em que se analisa a octanagem da gasolina em função da adição de um aditivo. Para isto, foram realizados ensaios com os percentuais de 1, 2, 3, 4, 5 e 6% de aditivo. Os resultados dos ensaios seguem na Tabela abaixo. Há como descrever, bem, uma relação linear entre estas duas variáveis?

Percentual do aditivo	Octanagem da gasolina
1	80,5
2	81,6
3	82,1
4	83,7
5	83,9
6	85,0

Índice de correlação linear de Pearson: 0,98.



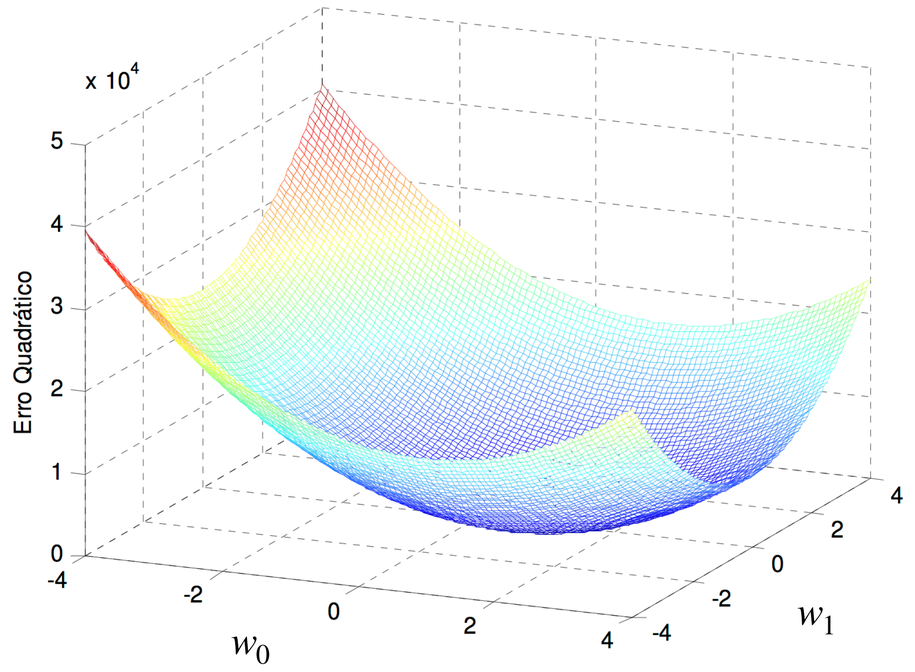
NA PRÁTICA (1)

$$\min_{\mathbf{w}} L(\boldsymbol{\varepsilon}) = \sum_{i=1}^{N} \varepsilon_i^2 = \sum_{i=1}^{N} \left(y_i - f(\mathbf{x}_i) \right)^2 = \sum_{i=1}^{N} \left(y_i - (w_0 + w_1 x_i) \right)^2$$

Como descobrir o $\mathbf{w}$, ou seja, w_0 e w_1 que minimizam $L(\,\cdot\,)$?

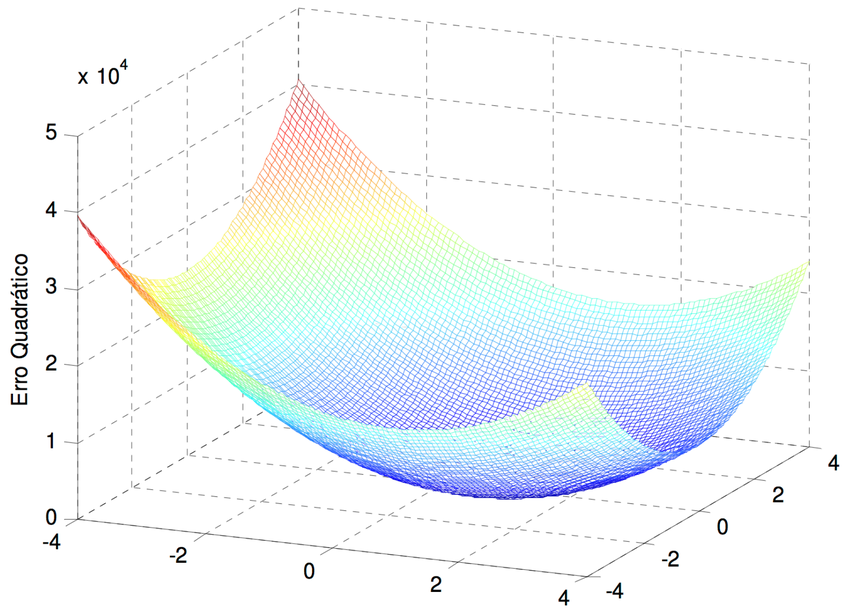
Reposta: igualando as derivadas parciais a zero!

$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial \varepsilon} \cdot \frac{\partial \varepsilon}{\partial f} \cdot \frac{\partial f}{\partial w_i} = 0.$$



NA PRÁTICA (2)

$$\min_{\mathbf{w}} L(\boldsymbol{\varepsilon}) = \sum_{i=1}^{N} \varepsilon_i^2 = \sum_{i=1}^{N} \left(y_i - f(\mathbf{x}_i) \right)^2 = \sum_{i=1}^{N} \left(y_i - (w_0 + w_1 x_i) \right)^2 = \sum_{i=1}^{N} \left(y_i - w_0 - w_1 x_i \right)^2$$



Precisamos calcular isso aqui: $\dfrac{\partial L}{\partial w_i} = \dfrac{\partial L}{\partial \varepsilon} \cdot \dfrac{\partial \varepsilon}{\partial f} \cdot \dfrac{\partial f}{\partial w_i} = 0.$

Colher de chá do cálculo:

- $\dfrac{\partial L}{\partial \varepsilon} = \dfrac{d}{d\varepsilon} \sum \varepsilon^2 = 2 \sum \varepsilon \,.$
- $\dfrac{\partial f}{\partial w_0} = \dfrac{d}{dw_0} (y_i - w_0 - w_1 x_i) = -1.$
- $\dfrac{\partial \varepsilon}{\partial f} = \dfrac{d}{df} y - f(\,\cdot\,) = -1.$
- $\dfrac{\partial f}{\partial w_1} = \dfrac{d}{dw_1} (y_i - w_0 - w_1 x_i) = -x_i \,.$

NA PRÁTICA (3)

- $\min_{\mathbf{w}} L(\boldsymbol{\varepsilon}) = \sum_{i=1}^{N} \varepsilon_i^2 = \sum_{i=1}^{N} \left(y_i - f(\mathbf{x}_i) \right)^2 = \sum_{i=1}^{N} \left(y_i - (w_0 + w_1 x_i) \right)^2 = \sum_{i=1}^{N} \left(y_i - w_0 - w_1 x_i \right)^2$
- $\dfrac{\partial L}{\partial w_i} = \dfrac{\partial L}{\partial \varepsilon} \cdot \dfrac{\partial \varepsilon}{\partial f} \cdot \dfrac{\partial f}{\partial w_i} = 0.$
- $\dfrac{\partial L}{\partial w_0} = \left(2 \sum_{i=1}^{N} \varepsilon_i \right)(-1)(-1) = 0 \rightarrow w_0 = \dfrac{1}{N} \sum_{i=1}^{N} y_i - w_1 \dfrac{1}{N} \sum_{i=1}^{N} x_i = \dfrac{\sum y_i - w_1 \sum x_i}{N}.$
- $\dfrac{\partial L}{\partial w_1} = \left(2 \sum_{i=1}^{N} \varepsilon_i \right)(-1)(-x_i) = 0 \rightarrow w_1 = \dfrac{N \left(\sum x_i y_i \right) - \left(\sum x_i \right) \left(\sum y_i \right)}{N \left(\sum x_i^2 \right) - \left(\sum x_i \right)^2}.$

NA PRÁTICA (3)

- $\min_{\mathbf{w}} L(\boldsymbol{\varepsilon}) = \sum_{i=1}^{N} \varepsilon_i^2 = \sum_{i=1}^{N} \left(y_i - f(\mathbf{x}_i) \right)^2 = \sum_{i=1}^{N} \left(y_i$
- $\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial \varepsilon} \cdot \frac{\partial \varepsilon}{\partial f} \cdot \frac{\partial f}{\partial w_i} = 0.$
- $\frac{\partial L}{\partial w_0} = \left(2 \sum_{i=1}^{N} \varepsilon_i \right)(-1)(-1) = 0 \rightarrow w_0 = \frac{1}{N}$
- $\frac{\partial L}{\partial w_1} = \left(2 \sum_{i=1}^{N} \varepsilon_i \right)(-1)(-x_i) = 0 \rightarrow w_1 = \frac{N}{}$

$$\frac{\partial L}{\partial w_0} = 2 \cdot \left(\sum \varepsilon_i \right)(-1)(-1) = 0$$

$$= 2 \left(\sum_i y_i - w_0 - w_1 x_i \right) = 0$$

$$= 2 \cdot \sum_i y_i - 2 \sum w_0 - 2 \sum w_1 \cdot x_i = 0$$

$$= \sum y_i - n \cdot w_0 - w_1 \cdot \sum x_i$$

$$n \cdot w_0 = \sum y_i - w_1 \cdot \sum x_i$$

$$w_0 = \frac{\sum y_i - w_1 \cdot \sum x_i}{n}$$

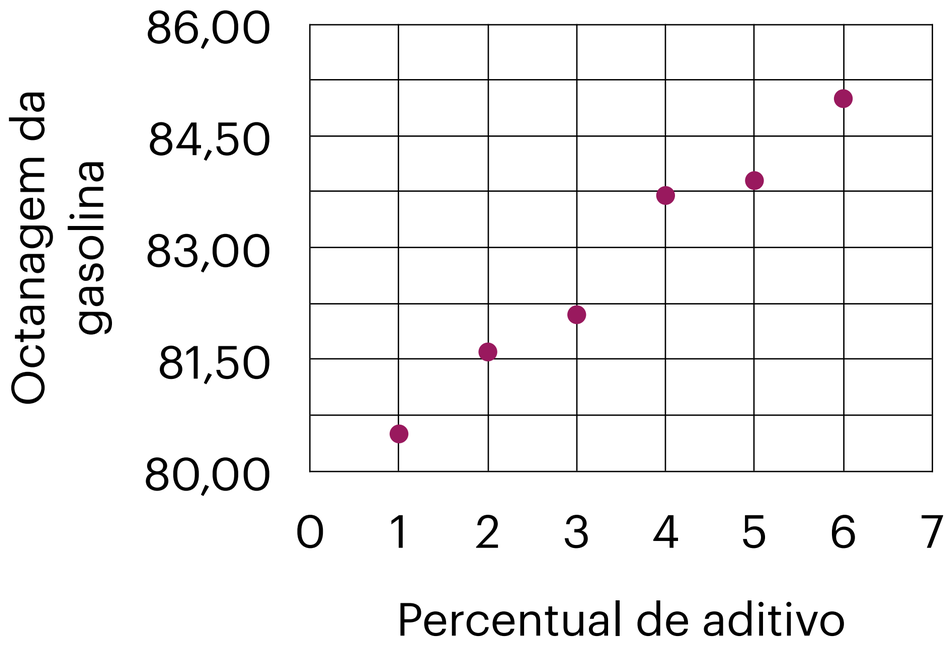
REGRESSÃO SIMPLES

EXPERIMENTO 01

Considere um experimento em que se analisa a octanagem da gasolina em função da adição de um aditivo. Para isto, foram realizados ensaios com os percentuais de 1, 2, 3, 4, 5 e 6% de aditivo. Os resultados dos ensaios seguem na Tabela abaixo. Há como descrever, bem, uma relação linear entre estas duas variáveis?

Percentual do aditivo	Octanagem da gasolina
1	80,5
2	81,6
3	82,1
4	83,7
5	83,9
6	85,0

Índice de correlação linear de Pearson: 0,98.



EXPERIMENTO 01

	Percentual do aditivo	Octanagem da gasolina	x^2	x * y
	1	80,5	1	80,5
	2	81,6	4	163,2
	3	82,1	9	246,3
	4	83,7	16	334,8
	5	83,9	25	419,5
	6	85,0	36	510
Σ	**21**	**496,8**	**91**	**1754,3**

$$w_1 = \frac{n\sum x_i y_i - \left(\sum x_i\right)\left(\sum y_i\right)}{n\left(\sum x_i^2\right) - \left(\sum x_i\right)^2}.$$

$$w_1 = \frac{(6)(1754,3) - (21)(496,8)}{(6)(91) - (21)^2}.$$

$$w_1 = 0,886.$$

$$w_0 = \frac{\sum y_i - w_1 \sum x_i}{n}.$$

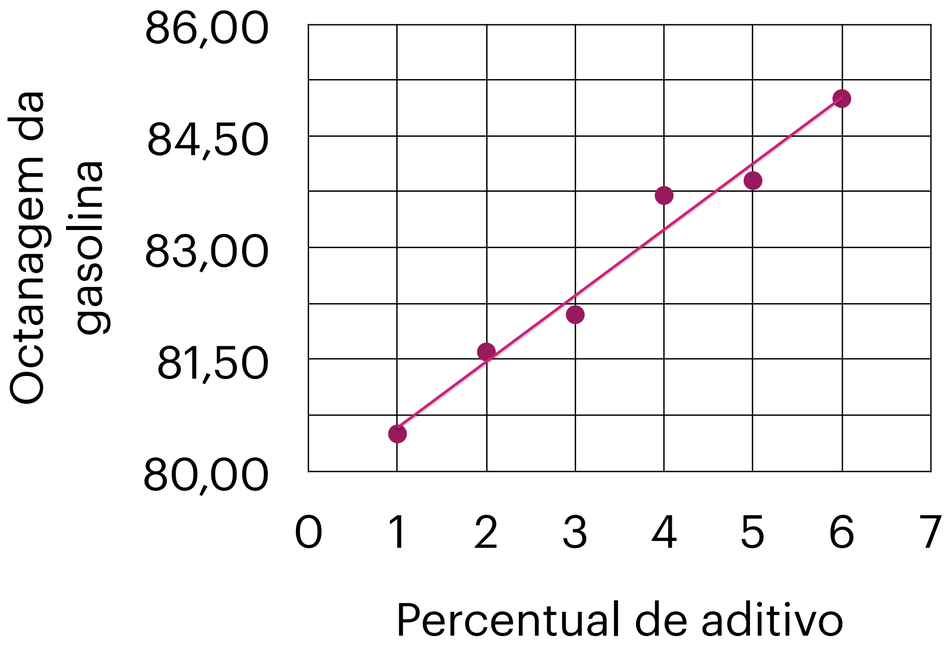
$$w_0 = \frac{496,8 - (0,886)(21)}{6}.$$

$$w_0 = 79,70.$$

EXPERIMENTO 01

Considere um experimento em que se analisa a octanagem da gasolina em função da adição de um aditivo. Para isto, foram realizados ensaios com os percentuais de 1, 2, 3, 4, 5 e 6% de aditivo. Os resultados dos ensaios seguem na Tabela abaixo. Há como descrever, bem, uma relação linear entre estas duas variáveis?

Percentual do aditivo	Octanagem da gasolina	Predição
1	80,5	80,59
2	81,6	81,47
3	82,1	82,36
4	83,7	83,24
5	83,9	84,13
6	85,0	85,02



REGRESSÃO MÚLTIPLA

- Podemos reescrever as M atributos independentes e suas N observações em forma matricial, o seja, os dados de treino como uma única matriz $\mathbf{X} \in \mathbb{R}^{N \times M}$ e $\mathbf{Y} \in \mathbb{R}^{N}$;

- Neste caso, supõe-se que $y_i = w_0 + w_1 X_{i,1} + w_1 X_{i,2} + \ldots, + w_m X_{i,m} + \varepsilon_i$;

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\mathbf{Y}} = \underbrace{\begin{bmatrix} 1 & X_{1,1} & X_{1,2} & \ldots & X_{1,M} \\ 1 & X_{2,1} & X_{2,2} & \ldots & X_{2,M} \\ & & \cdots & & \\ 1 & X_{N,1} & X_{N,2} & \ldots & X_{N,M} \end{bmatrix}}_{\mathbf{X} \text{ com o bias}} \times \underbrace{\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_M \end{bmatrix}}_{\mathbf{w}} + \underbrace{\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{bmatrix}}_{\boldsymbol{\varepsilon}} \text{ ou } \mathbf{Y} = \mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon}.$$

REGRESSÃO MÚLTIPLA

- Neste caso, supõe-se que $y_i = w_0 + w_1 X_{i,1} + w_1 X_{i,2} + \ldots, + w_m X_{i,m} + \varepsilon_i$;

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\mathbf{Y}} = \underbrace{\begin{bmatrix} 1 & X_{1,1} & X_{1,2} & \cdots & X_{1,M} \\ 1 & X_{2,1} & X_{2,2} & \cdots & X_{2,M} \\ & & \cdots & & \\ 1 & X_{N,1} & X_{N,2} & \cdots & X_{N,M} \end{bmatrix}}_{\mathbf{X} \text{ com o bias}} \times \underbrace{\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_M \end{bmatrix}}_{\mathbf{w}} + \underbrace{\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{bmatrix}}_{\boldsymbol{\varepsilon}} \text{ ou } \mathbf{Y} = \mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon}.$$

- O problema pode ser pode ser reformulada como:

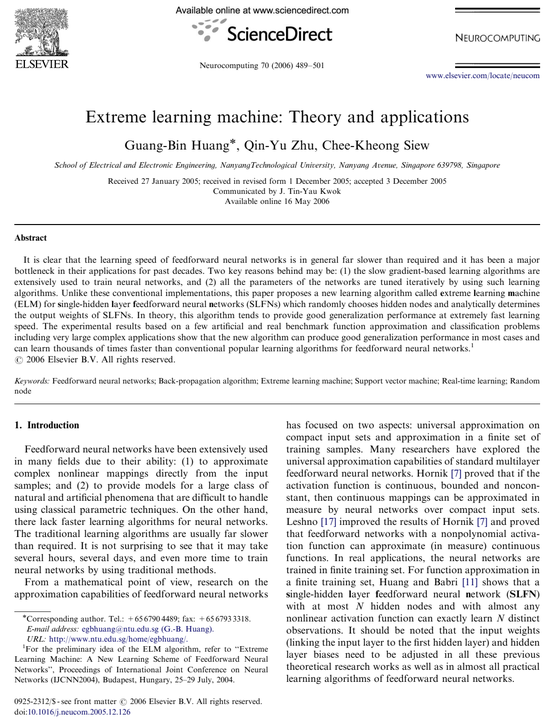
$$\min_{\mathbf{w}} L(\mathbf{w}) = \sum_{i=1}^{N} \varepsilon_i^2 = \boldsymbol{\varepsilon}^\intercal \boldsymbol{\varepsilon} = (\mathbf{Y} - \mathbf{X}\mathbf{w})^\intercal (\mathbf{Y} - \mathbf{X}\mathbf{w}) = \mathbf{Y}^\intercal \mathbf{Y} - \mathbf{Y}^\intercal \mathbf{X}\mathbf{w} - \mathbf{w}^\intercal \mathbf{X}^\intercal \mathbf{Y} + \mathbf{w}^\intercal \mathbf{X}^\intercal \mathbf{X}\mathbf{w}.$$

- Após derivar a função custo acima em função $\mathbf{w}$ e igualar a zero (Matrix Cookbook), tem-se que: $\mathbf{w} = \left(\mathbf{X}^\intercal \mathbf{X}\right)^{-1} \mathbf{X}^\intercal \mathbf{Y}$ ou $\mathbf{w} = \mathbf{X}^\dagger \mathbf{Y}$ (Pseudo-inversa de Moore-Penrose).

MÁQUINA DE APRENDIZAGEM EXTREMA

MÁQUINA DE APRENDIZAGEM EXTREMA

- Esse modelo contorna as desvantagens citadas anteriormente;
- Foi desenvolvido para redes com apenas uma camada oculta (SLFN – *Single Layer Feedforward Network*);
- Os pesos da camada escondida são escolhidos de forma aleatória;
- Os pesos da camada de saída são determinados **analiticamente** (i.e., não há ciclos iterativos para ajuste de parâmetros. É pei e bufo!).



Available online at www.sciencedirect.com
ScienceDirect
Neurocomputing 70 (2006) 489–501
NEUROCOMPUTING
www.elsevier.com/locate/neucom

Extreme learning machine: Theory and applications

Guang-Bin Huang*, Qin-Yu Zhu, Chee-Kheong Siew

School of Electrical and Electronic Engineering, NanyangTechnological University, Nanyang Avenue, Singapore 639798, Singapore

Received 27 January 2005; received in revised form 1 December 2005; accepted 3 December 2005
Communicated by J. Tin-Yau Kwok
Available online 16 May 2006

Abstract

It is clear that the learning speed of feedforward neural networks is in general far slower than required and it has been a major bottleneck in their applications for past decades. Two key reasons behind may be: (1) the slow gradient-based learning algorithms are extensively used to train neural networks, and (2) all the parameters of the networks are tuned iteratively by using such learning algorithms. Unlike these conventional implementations, this paper proposes a new learning algorithm called extreme **l**earning **m**achine (ELM) for **s**ingle-hidden **l**ayer **f**eedforward neural **n**etworks (SLFNs) which randomly chooses hidden nodes and analytically determines the output weights of SLFNs. In theory, this algorithm tends to provide good generalization performance at extremely fast learning speed. The experimental results based on a few artificial and real benchmark function approximation and classification problems including very large complex applications show that the new algorithm can produce good generalization performance in most cases and can learn thousands of times faster than conventional popular learning algorithms for feedforward neural networks.[1]

Keywords: Feedforward neural networks; Back-propagation algorithm; Extreme learning machine; Support vector machine; Real-time learning; Random node

1. Introduction

Feedforward neural networks have been extensively used in many fields due to their ability: (1) to approximate complex nonlinear mappings directly from the input samples; and (2) to provide models for a large class of natural and artificial phenomena that are difficult to handle using classical parametric techniques. On the other hand, there lack faster learning algorithms for neural networks. The traditional learning algorithms are usually far slower than required. It is not surprising to see that it may take several hours, several days, and even more time to train neural networks by using traditional methods.

From a mathematical point of view, research on the approximation capabilities of feedforward neural networks has focused on two aspects: universal approximation on compact input sets and approximation in a finite set of training samples. Many researchers have explored the universal approximation capabilities of standard multilayer feedforward neural networks. Hornik [7] proved that if the activation function is continuous, bounded and nonconstant, then continuous mappings can be approximated in measure by neural networks over compact input sets. Leshno [17] improved the results of Hornik [7] and proved that feedforward networks with a nonpolynomial activation function can approximate (in measure) continuous functions. In real applications, the neural networks are trained in finite training set. For function approximation in a finite training set, Huang and Babri [11] shows that a single-hidden layer **f**eedforward neural **n**etwork (**SLFN**) with at most N hidden nodes and with almost any nonlinear activation function can exactly learn N distinct observations. It should be noted that the input weights (linking the input layer to the first hidden layer) and hidden layer biases need to be adjusted in all these previous theoretical research works as well as in almost all practical learning algorithms of feedforward neural networks.

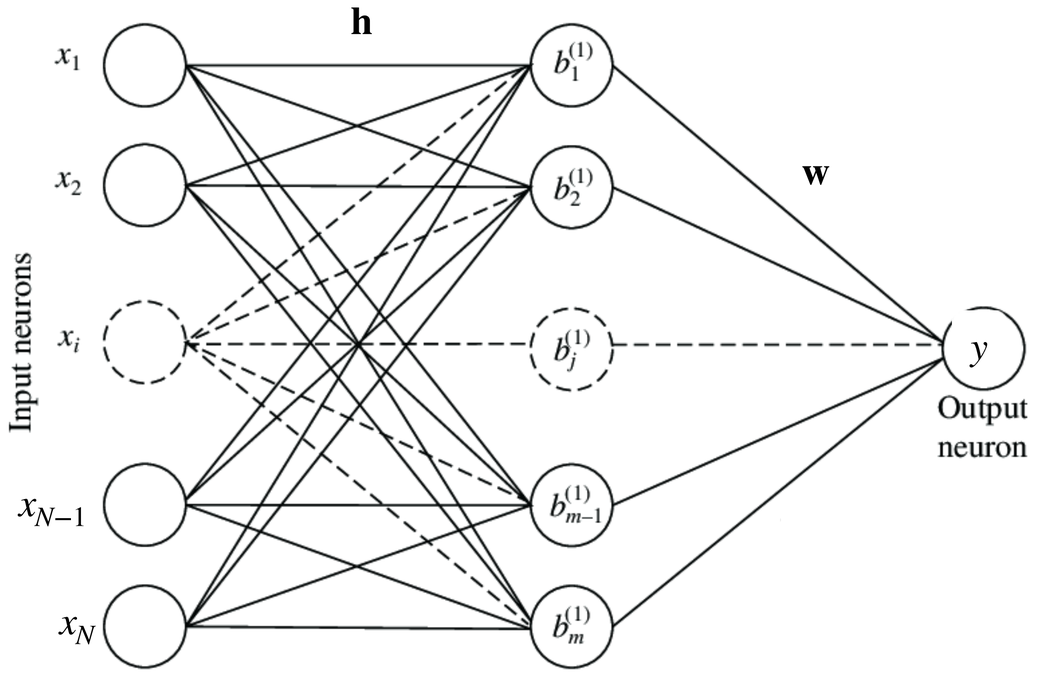
*Corresponding author. Tel.: +65 6790 4489; fax: +65 6793 3318.
E-mail address: egbhuang@ntu.edu.sg (G.-B. Huang).
URL: http://www.ntu.edu.sg/home/egbhuang/.

[1]For the preliminary idea of the ELM algorithm, refer to "Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks", Proceedings of International Joint Conference on Neural Networks (IJCNN2004), Budapest, Hungary, 25–29 July, 2004.

0925-2312/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2005.12.126

G.-B. Huang, Q.-Y. Zhu e C.-K. Siew, Extreme Learning Machine: Theory and Applications, Neurocomputing 70, 489-501 (2006).

MÁQUINA DE APRENDIZAGEM EXTREMA



$$y = \varphi\left(\sum_{j=1}^{M} w_j b_j\right) = \varphi\left(\mathbf{b}^\intercal \mathbf{w}\right). \tag{1}$$

$$b_j = \varphi_j\left(\sum_{i=1}^{N} h_i^{(j)} x_i\right) = \varphi_j\left(\mathbf{x}^\intercal \mathbf{h}^{(j)}\right). \tag{2}$$

$$\mathbf{h}^{(j)} \sim \mathrm{p}(\theta) \quad (\text{e.g.} \quad \mathbf{h}^{(j)} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})) \tag{3}$$

TEORIA SUPORTADA PELA ELM

Teorema 2.1. Dada uma SLFN com N neurônios na camada escondida e função de ativação $\varphi : \mathbb{R} \to \mathbb{R}$ infinitamente diferenciável em qualquer intervalo, para N exemplos de treinamento distintos $(\mathbf{x}_i, \mathbf{y}_i), \mathbf{x}_i \in \mathbb{R}^n$ e $\mathbf{y}_i \in \mathbb{R}^m$, para quaisquer peso ou viés aleatoriamente selecionados dentro de quaisquer internados $\mathbb{R}^n$ e $\mathbb{R}$, respetivamente, por qualquer função de distribuição de probabilidade, então com probabilidade 1 (máxima), a matriz da camada escondida $\mathbf{B}$ da SLFN é inversível e $\| \mathbf{BW} - \mathbf{Y} \| = 0$

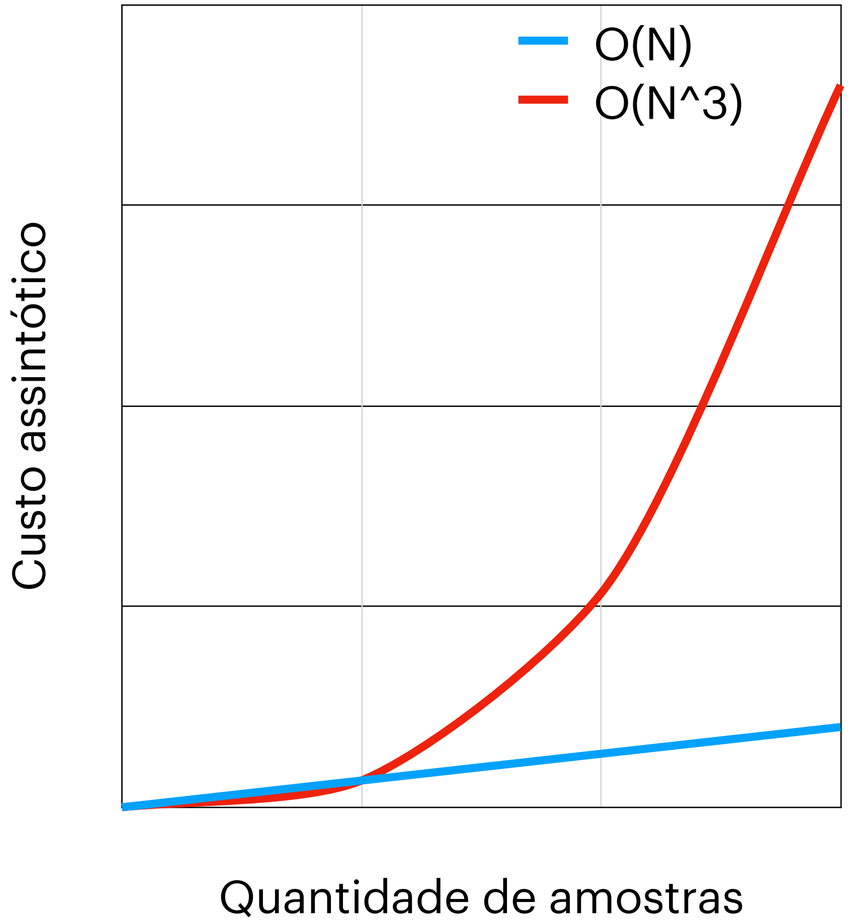
Tradução. Se o número de neurônios $\tilde{N}$ da camada escondida é igual ao número N de exemplos de treinamento, isto é, $N = \tilde{N}$, então a matriz $\mathbf{B}$ é quadrada e inversível quando os pesos e viéses são aleatoriamente escolhidos e, assim, as a SLFNs podem aprender estes exemplos de **treinamento com erro zero**.

S. Tamura e M. Tateishi, Capabilities of a Four-Layered Feedforward Neural Network: Four Layers Versus Three, IEEE Trans. Neural Networks, 251-255 (1997).

G.-B. Huang, Learning Capability and Storage Capacity of Two-Hidden-Layer Feedforward Networks, IEEE Trans. Neural Networks, 274-281 (2003).

TEORIA SUPORTADA PELA ELM

- No entanto, **na maioria dos casos**, é impraticável ter uma matriz tão grande para inverter, pois o custo da inversão é $O\left(N^3\right)$;
- Assim, na maioria dos casos, o número de neurônios da camada escondida é muito menor do que o número de amostras, resultando em uma matriz $\mathbf{B}$ não quadrada;
- Pode-se adotar a solução por mínimos quadrados com a menor norma, isto é, $\mathbf{W} = \mathbf{B}^{\dagger}\mathbf{Y}$, em que $\mathbf{B}^{\dagger}$ é a matriz inversa generalizada de Moore-Penrose da matriz $\mathbf{B}$, i.e., $\mathbf{B}^{\dagger} = (\mathbf{B}^{\intercal}\mathbf{B})^{-1}\mathbf{B}$.
- Utilize as funções do pacote de álgebra linear da biblioteca NumPy, a saber, np.linalg.inv e np.linalg.pinv.



ALGORITMO DA ELM

TREINANDO UMA ELM

1. TREINO
   1.1. Selecionar aleatoriamente valores para os pesos para a camada oculta $\mathbf{H}$;
   1.2. Projetar os dados para gerar a matriz $\mathbf{B} = \varphi\left(\mathbf{XH}\right)$ (atente-se aos viéses);
   1.3. Calcular os pesos ótimos $\mathbf{W} = \mathbf{B}^{\dagger}\mathbf{Y}$.
2. TESTE
   2.1. Para uma amostra simples, faça $y = \varphi_{\mathrm{o}}\left(\mathbf{W}\,\varphi_{\mathrm{h}}\left(\mathbf{x}^{\intercal}\mathbf{H}\right)\right)$.

POLÊMICAS

The Official Homepage on Origins of Extreme Learning Machines (ELM)

https://elmorigin.wixsite.com/originofelm

Possible source of inspiration: Is ELM a follow-up work of the Nature article in 2004 alerting intelligent plagiarim is more harmful than literal (verbatim) plagiarism. In particular, "How should we tackle the increasing problem of researchers rewriting others' results?"

Read these ludicrous comparison papers by G.-B. Huang with our commentaries Cog Comp 2015 and Cog Comp 2014.

Introduction: The objective of launching this homepage is to present evidences regarding the tainted origins of the extreme learning machines (ELM). As we would like all readers to verify the facts within a short period of time (perhaps 10 to 20 minutes), we have uploaded a dozen of PDF files with highlights and annotations clearly showing the following:

1. The kernel (or constrained-optimization-based) version of ELM (ELM-Kernel, PDF: Huang-LS-SVM-2012) is identical to kernel ridge regression (for regression and single-output classification, PDF: Saunders ICML 1998; for multiclass multi-output classification, PDF: An CVPR 2007).

2. ELM-SLFN (the single-layer feedforward network version of the ELM, PDF: Huang IJCNN 2004) is identical to the randomized neural network (RNN, with omission of bias, PDF: Schmidt 1992) and another simultaneous work, i.e., the random vector functional link (RVFL, with omission of direct input-output links, PDF: Pao 1994). According to the recent results, it is apparent that the older and original RVFL is far superior than the ELM for time series forecasting and classification.

3. ELM-RBF (PDF: Huang ICARCV 2004) is identical to the randomized RBF neural network (PDF: Broomhead 1988), with a performance-degrading randomization of RBF radii or impact factors).

4. In all three cases above, Huang got his papers published after excluding a large volume of very closely related literature.

5. Hence, all 3 "ELM variants" have absolutely no technical originality, promote unethical research practices among researchers, and steal citations from original inventors.

NEUCOM-D-17-00546

2 messages

Neurocomputing <eesserver@eesmail.elsevier.com> 17 juin 2017 à 06 h 53
Répondre à : Neurocomputing <neurocomputing@science.ru.nl>
À : saulo.freitas.oliveira@gmail.com

Ref.: Ms. No. NEUCOM-D-17-00546
Two New Corner Detector-based Reference Point Selection Methods for Minimal Learning Machines
Neurocomputing

Dear Mr. Oliveira,

Please find below the referee reports. Based on these and the corresponding recommendations, I have to reach the sad conclusion that your paper

Two New Corner Detector-based Reference Point Selection Methods for Minimal Learning Machines

cannot be accepted for publication in Neurocomputing. I hope that the referees' comments and suggestions are nevertheless useful and can help you improving your scientific work and/or presentation.

Hereby I would like to thank you for submitting your work to Neurocomputing and welcome you to consider us again in the future.

Reviewer #2: This paper presented two proposals for reference point selection based on a corner detector algorithm, namely, the Features from Accelerated Segment Test (FAST), and stated that MLM can be classified as a learning algorithm that has its "nonlinear approximation capability " based on a random projection of the input points. Examples of randomized learning machines include Extreme Learning Machines and Random Vector Functional Link(RVFL).
I think that the authors don't seem to have a good understanding of the nature and history of the neural networks with random weights, and lack some basic background knowledge for random learning of neural networks. The authors need to learn more facts related to random learning techniques, and in particular, the use of papers related to "ELM" in your reference due to a serious academic misconduct and unethical behavior related to the so-called ELM. I suggest the authors to see the following conference publication [1] and review paper [9]. The so-called ELM is a copy of Schmidt et al. works. The authors should also know that Pao et al. and their co-authors in 1992 also proposed the same learning algorithm, and they published a very solid paper in TNN 1995(RVFL). Huang et al copied such a work and published in 2004 IJCNN paper (cited in [2] in your paper) with a term "ELM". This is very unethical and dishonesty indeed. This scandal fact had been disclosed by Wang and Wan [5],
and recently by [6, 7, 8] .
The authors should be to see that the conclusion of Huang's ELM is incorrect. In fact, in so called ELM, the weighs and the threshold are taken in [-1,1] randomly in almost "ELM" papers . This is a big joke, which comes from Huang's the so-called original papers in 2004 and 2006 (the Refs. [2] cited in your manuscript). Indeed, the random assignment of the initial weights and biases has no guarantee at all to modeling exercises. A counter example in the following Ref. [2] can show you this disability.
References
[1] W. Schmidt, M. Kraaijveld, R. Duin, Feedforward neural networks with random weights, in: Proceedings of 11th IAPR International Conference on Pattern Recognition Methodology and Systems, 1992, pp. 1-4.
[2] Ivan Yu. Tyukin and Danil V. Prokhorov, Feasibility of random basis function approximators for modeling and control, IEEE Conference on Control Applications, (CCA) & Intelligent Control, (ISIC), 8-10 July 2009 , On page(s): 1391 - 1396.
[3] Yoh-Han Pao, Gwang-Hoon Park and Dejan J. Sobajic, Learning and generalization characteristics of the random vector Functional-link net, Neurocomputing 6 (1994) 163-180
[4] Bons Igelnik and Yoh-Han Pao, Stochastic Choice of Basis Functions in Adaptive Function Approximation and the

Reviewer #2: This paper presented two proposals for reference point selection based on a corner detector algorithm, namely, the Features from Accelerated Segment Test (FAST), and stated that MLM can be classified as a learning algorithm that has its "nonlinear approximation capability " based on a random projection of the input points. Examples of randomized learning machines include Extreme Learning Machines and Random Vector Functional Link(RVFL).
I think that the authors don't seem to have a good understanding of the nature and history of the neural networks with random weights, and lack some basic background knowledge for random learning of neural networks. The authors need to learn more facts related to random learning techniques, and in particular, the use of papers related to "ELM" in your reference due to a serious academic misconduct and unethical behavior related to the so-called ELM. I suggest the authors to see the following conference publication [1] and review paper [9]. The so-called ELM is a copy of Schmidt et al. works. The authors should also know that Pao et al. and their co-authors in 1992 also proposed the same learning algorithm, and they published a very solid paper in TNN 1995(RVFL). Huang et al copied such a work and published in 2004 IJCNN paper (cited in [2] in your paper) with a term "ELM". This is very unethical and dishonesty indeed. This scandal fact had been disclosed by Wang and Wan [5],
and recently by [6, 7, 8] .
The authors should be to see that the conclusion of Huang's ELM is incorrect. In fact, in so called ELM, the weighs and the threshold are taken in [-1,1] randomly in almost "ELM" papers . This is a big joke, which comes from Huang's the so-called original papers in 2004 and 2006 (the Refs. [2] cited in your manuscript). Indeed, the random assignment of the initial weights and biases has no guarantee at all to modeling exercises. A counter example in the following Ref. [2] can show you this disability.
References
[1] W. Schmidt, M. Kraaijveld, R. Duin, Feedforward neural networks with random weights, in: Proceedings of 11th IAPR International Conference on Pattern Recognition Methodology and Systems, 1992, pp. 1-4.
[2] Ivan Yu. Tyukin and Danil V. Prokhorov, Feasibility of random basis function approximators for modeling and control, IEEE Conference on Control Applications, (CCA) & Intelligent Control, (ISIC), 8-10 July 2009 , On page(s): 1391 - 1396.
[3] Yoh-Han Pao, Gwang-Hoon Park and Dejan J. Sobajic, Learning and generalization characteristics of the random vector Functional-link net, Neurocomputing 6 (1994) 163-180
[4] Bons Igelnik and Yoh-Han Pao, Stochastic Choice of Basis Functions in Adaptive Function Approximation and the

Saulo Oliveira <saulo.freitas.oliveira@gmail.com> 17 juin 2017 à 07 h 29
À : João Paulo Podeus <jpaulopg@gmail.com>, Amauri Holanda <amauri01@gmail.com>

[Texte des messages précédents masqué]

--

REDES DE BASE RADIAL

PLANO DE FUNDO

- Kohonen (1982, 2001) mostram que há uma solução ótima para a matriz de pesos;
- O método *Optimal Linear Associative Memory* (OLAM) tem como princípio básico o uso de técnicas de inversão de matriz;
  - Kohonen, T., Ruohonen, K. (1973). Kohonen, T., Ruohonen, K. ***Representation of associated data by matrix operations.*** IEEE Trans. on Computers 22 (1973), 701-702.
- 🪦 13 de dezembro de 2021, aos 87 anos!

https://www.aalto.fi/en/news/teuvo-kohonen-in-memoriam

MODELO PARAMÉTRICO VS. NÃO PARAMÉTRICO

PRINCIPAIS DIFERENÇAS

PARAMÉTRICO

ADVANTAGES	DISADVANTAGES
Parametric algorithms are SIMPLE to understand, as they work on a limited number of decided parameters.	But, assuming one functional form, makes them constrained to only that type of form i.e. limited flexibility & scope of work.
They require less computational power, hence are FASTER.	Limited complexity in work i.e. works better for simple & less amounts of data.
Do NOT require perfect or large amount of data for training.	Assumed forms & methods may or MAY NOT match the underlying mapping function.

NÃO-PARAMÉTRICO

ADVANTAGES	DISADVANTAGES
Non-Parametric algorithms **do not make** any **assumptions** on functional forms & work on trying to best fit the data.	But, require **large amounts of training data** for estimating & constructing mapping functions.
Capable of **fitting & learning** a **large number of functional forms** (since no constraints of just one type of form)	They have many parameters to train & work on. Hence, they are **SLOWER** in giving results
They result in **high performance models** for prediction.	Large amounts of training data may result in **OVERFITTING** & **no justifications** available for certain **predictions** made.

https://medium.com/lets-talk-ml/parametric-and-non-parametric-algorithms-in-ml-bc10729ff0e

REGRESSÃO NÃO-PARAMÉTRICA

- Na regressão não paramétrica, não se assume conhecimento a priori sobre a forma da função que se quer estimar;
- A função é estimada usando uma equação contendo parâmetros livres mas numa forma que permite ao modelo representar uma classe muito ampla de funções;
- A rede RBF implementa uma combinação linear de funções de base radiais, elas mesmo não lineares.

REGRESSÃO NÃO-PARAMÉTRICA

- Na regressão não paramétrica, não se assume conhecimento a priori sobre a forma da função que se quer estimar;
- A função é estimada usando uma equação contendo parâmetros livres mas numa forma que permite ao modelo representar uma classe muito ampla de funções;
- A rede RBF implementa uma combinação l
  radiais, elas mesmo não lineares.

Seus pesos não têm um significado particular em relação aos problemas aos quais elas estão sendo aplicadas.

$$f(\mathbf{x}) = \sum_{j=1}^{N} w_j \varphi_j(\mathbf{x})$$

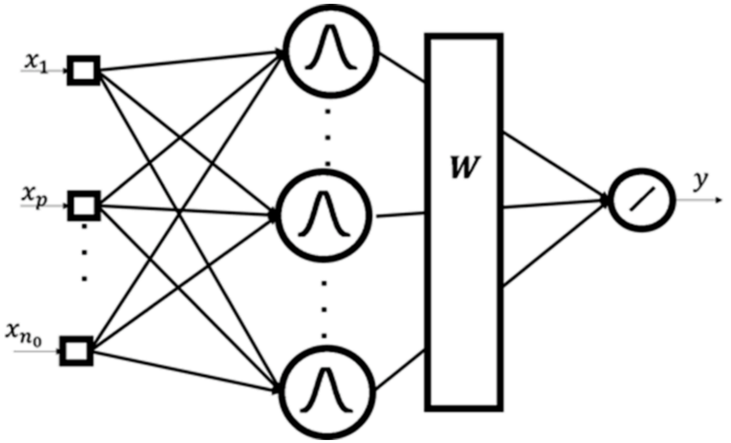
O objetivo principal não é estimar os valores dos parâmetros (pesos) e sim estimar a *função*, ou no mínimo as suas saídas para certos valores desejados de entrada.

Tipicamente a regressão não-paramétrica envolve um grande número de parâmetros sem significado físico em relação ao problema.

REDES NEURAIS PODEM SER NÃO-PARAMÉTRICAS?

VISÃO GERAL DAS REDES RBF

- As redes RBF unificam diversas teorias importantes envolvendo aproximação de funções, regularização, interpolação ruidosa, estimação de densidade, classificação ótima e funções de potencial;

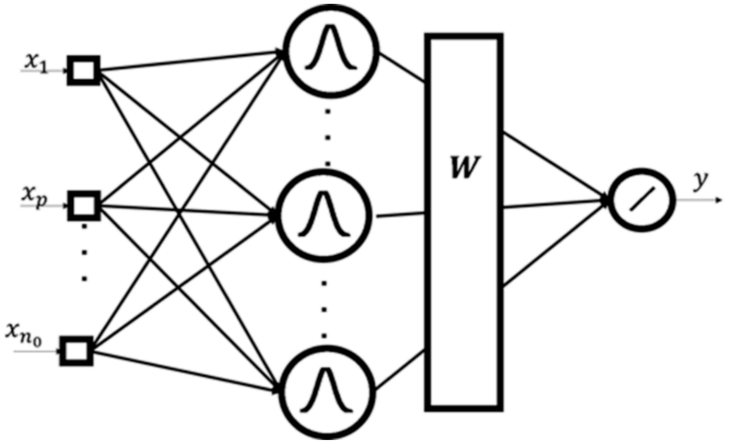


- Essa unificação faz a rede RBF pertencer a uma classe de modelos na qual a ativação de uma unidade oculta é determinada pela distância entre o vetor de entrada e um vetor protótipo (SVM, MLM, dentre outras);

- As unidades ocultas formam representações internas interpretáveis o que leva a um treinamento em dois estágio: (i) são determinados os parâmetros das funções de base (não-supervisionado); (ii) são determinados os pesos da camada de saída (problema linear).

VISÃO GERAL DAS REDES RBF

- Tipicamente, uma rede RBF tem uma camada de entrada, uma camada oculta, composta de nós de funções radiais e uma camada de saída com um nó linear;



- A forma das funções de base é escolhida a priori, de modo que ela tenha um comportamento adequado ao problema;

- Assim, o problema consiste então em localizar os centros e outros parâmetros das funções de base e ajustar os pesos em relação ao treinamento.

FUNÇÕES RADIAIS

FUNÇÕES RADIAIS

- A sua característica principal é que sua resposta diminui (ou aumenta) monotonamente com a distância de um ponto central. Geralmente, usa-se a norma euclidiana para cômputo dessa função.

$\mathbf{c}_j$ é o vetor central da função radial.

$$\varphi_j(\mathbf{x}) = \exp\left(-\frac{\| \mathbf{x} - \mathbf{c}_j \|_2^2}{2\,\sigma_j^2}\right)$$

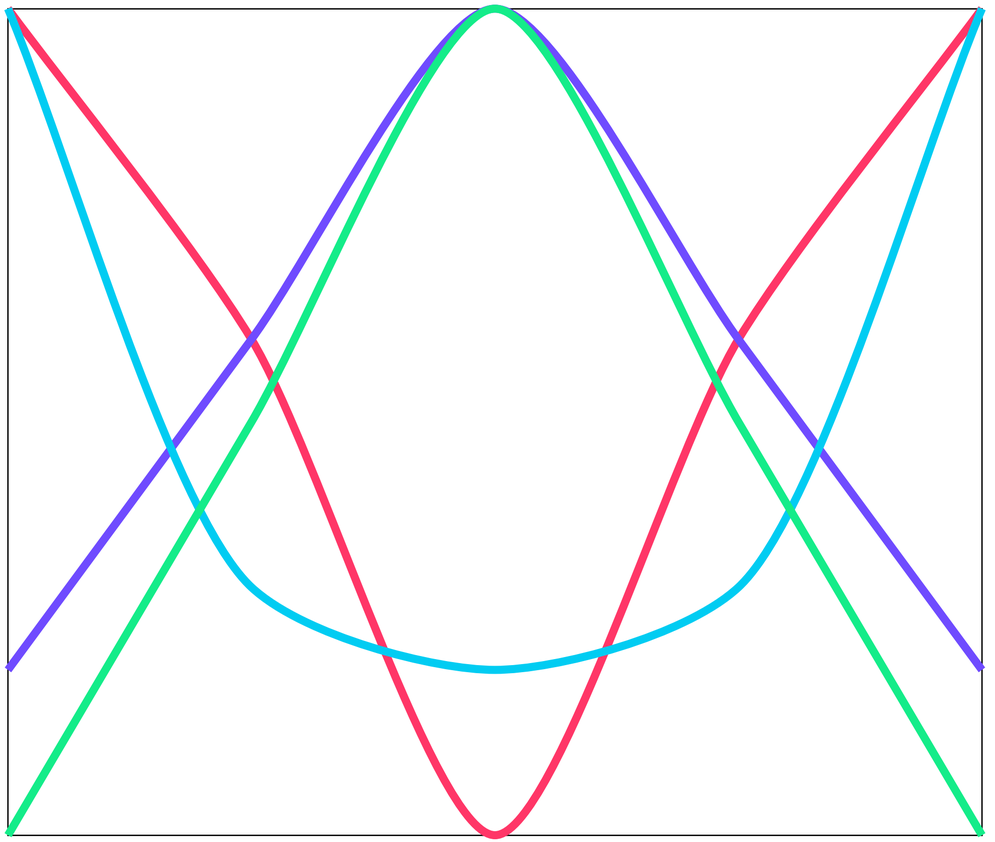
σ_j controla a suavidade da interpolação

$\mathbf{c}_j$ é o vetor central da função radial.

$$\varphi_j(\mathbf{x}) = \exp\left\{-\frac{1}{2}(\mathbf{x} - \mathbf{c}_j)^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x} - \mathbf{c}_j)\right\}$$

$\boldsymbol{\Sigma}_j$ matriz de covariâncias.

FUNÇÕES RADIAIS



- Gaussiana
- Thin-plate splite
- Multiquadrática inversa
- Multiquadrática

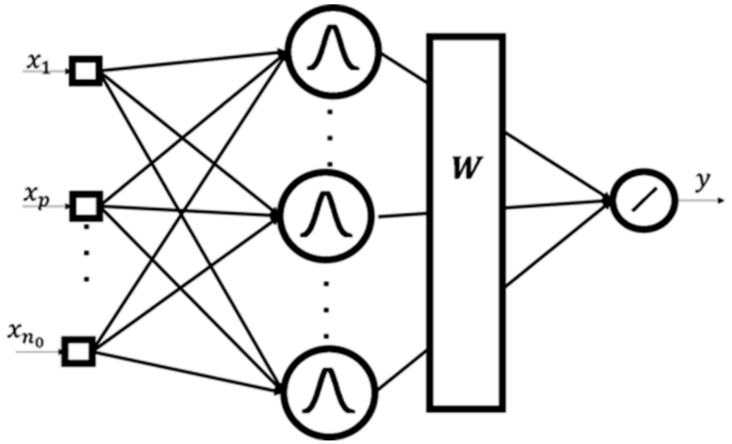
INTERPOLAÇÃO EXATA

TREINANDO UMA RBF

- O problema da interpolação exata requer que cada vetor de entrada seja mapeado exatamente para o seu vetor de saída correspondente, isto é, $f(\mathbf{x}) = \sum_{j=1}^{M} \mathbf{w}_j \varphi(\| \mathbf{x} - \mathbf{x}_j \|)$, resultando assim numa saída do mapeamento que é uma combinação linear das funções de base;

- Em notação matricial, podemos escrever como $\mathbf{y} = \mathbf{\Phi w}$ em que $\Phi_{i,j} = \varphi(\| \mathbf{x}_i - \mathbf{x}_j \|)$;

- Uma vez que possamos inverter $\mathbf{\Phi}$, tem-se que $\mathbf{w} = \mathbf{\Phi}^{-1}\mathbf{y}$.



ALGORITMO DA RBF

TREINANDO UMA RBF

1. Definição dos centros;
    1. Podemos assumir m funções fixas, centradas em pontos $\mathbf{c}_j$ escolhidos aleatoriamente das amostras de treino;
    2. Ou podemos utilizar algum algoritmo de agrupamento (K-médias, por exemplo) definido o número de grupos igual a m.
2. Cômputo dos pesos ótimos;
    1. Inversão da matriz de interpolação;

Referências

- Paulo Martins Engel. **Redes Neurais: A Rede RBF**. Universidade Federal do Rio Grande Sul, 2008. Disponível em: http://www.inf.ufrgs.br/~engel/. Acessado em: 01 de maio de 2022.
- Richard O. Duda, Peter E. Hart, David G. Stork. **Pattern Classification**. John Wiley & Sons, 2012.
- Thiago Henrique Cupertino. **O Algoritmo de Treinamento: Máquina de Aprendizado Extremo (Extreme Learning Machine - ELM)**. Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, 2010. Disponível em: http://wiki.icmc.usp.br/index.php/SCC5809(Roseli). Acessado em: 03 de abril de 2022.