Cogn Comput (2015) 7:263–278
DOI 10.1007/s12559-015-9333-0

What are Extreme Learning Machines? Filling the Gap Between Frank Rosenblatt's Dream and John von Neumann's Puzzle

Guang-Bin Huang[1]

Received: 9 April 2015 / Accepted: 27 April 2015 / Published online: 15 May 2015

Abstract The emergent machine learning technique—extreme learning machines (ELMs)—has become a hot area of research over the past years, which is attributed to the growing research activities and significant contributions made by numerous researchers around the world. Recently, it has come to our attention that a number of misplaced notions and misunderstandings are being dissipated on the relationships between ELM and some earlier works. This paper wishes to clarify that (1) ELM theories manage to address the open problem which has puzzled the neural networks, machine learning and neuroscience communities for 60 years: whether hidden nodes/neurons need to be tuned in learning, and proved that in contrast to the common knowledge and conventional neural network learning tenets, hidden nodes/neurons do not need to be iteratively tuned in wide types of neural networks and learning models (Fourier series, biological learning, etc.). Unlike ELM theories, none of those earlier works provides theoretical foundations on feedforward neural networks with random hidden nodes; (2) ELM is proposed for both generalized single-hidden-layer feedforward network and multi-hidden-layer feedforward networks (including biological neural networks); (3) homogeneous architecture-based ELM is proposed for feature learning, clustering, regression and (binary/multi-class) classification. (4) Compared to ELM, SVM and LS-SVM tend to provide suboptimal solutions, and SVM and LS-SVM do not consider feature representations in hidden layers of multi-hidden-layer feedforward networks either.

Keywords Extreme learning machine · Random vector functional link · QuickNet · Radial basis function network · Feedforward neural network · Randomness

Introduction

Despite that the relationships and differences between extreme learning machines (ELMs) and those earlier works (e.g., Schmidt et al. [1] and RVFL [2]) have been clarified in [3–8], recently several researchers insist that those earlier works are the "origins" of ELM and ELM essentially the same as those earlier works, and thus, further claimed that it is not necessary to have a new term extreme learning machines (ELMs). This paper wishes to clarify the essential elements of ELMs which may have been overlooked in the past years.

We prefer to avoid the word "origin" in this paper as (1) it may be really difficult to show which is the "true" "origin" in a research area as most works are related to each other and (2) it may cause unnecessary inconvenience or potential "controversy" among those listed as "origins" and other pioneering works which may have been missing in discussions. *The ultimate goal of research is to find the truth of natural phenomena and to move research forward instead of arguing for being listed as "origins."* Otherwise, many earlier works should not have had their own terms, and instead, almost all should simply have been called "feedforward neural networks" or should even simply go back to Frank Rosenblatt's "perceptrons" [9]. Such misunderstandings on ignoring the needs of having new terms would actually discourage researchers' creativeness and their spirit of telling the truth and differences in research. Similarly, there is nothing wrong to have new terms for the variants of ELM (with Fourier series nodes) with

✉ Guang-Bin Huang
egbhuang@ntu.edu.sg

[1] School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore

significant extensions (such as random kitchen sink (RKS) [10], RKS' further extension—FastFood [11] and Convex Network [12]) as well as ELM with LMS referred to as No-Prop algorithm [13].

Generally speaking, as analyzed in Huang et al. [6]: " 'Extreme' here means to move beyond conventional artificial learning techniques and to move toward brain alike learning. ELM aims to break the barriers between the conventional artificial learning techniques and biological learning mechanism. 'Extreme learning machine (ELM)' represents a suite of machine learning techniques in which hidden neurons need not be tuned with the consideration of neural network generalization theory, control theory, matrix theory and linear system theory."

In order to have clearer understanding of ELM, it is better to analyze ELM in the aspects of its philosophy, theories, network architecture, network neuron types and its learning objectives and algorithms.

ELM's Beliefs, Philosophy and Objectives

ELM works start from our intuitive belief on biological learning and neural networks generalization performance theories [14]. Further development of ELM works is also built on top of Frank Rosenblatt's multilayer 'perceptrons" [9], SVM [15], LS-SVM [16], Fourier series, linear systems, numeral methods, matrix theories, etc., but with essential extensions.

Frank Rosenblatt [9] believes that multilayer feedforward networks (perceptrons) can enable computers to "walk, talk, see, write, reproduce itself and be conscious of its existence."[1]. Minsky and Papert [17] do not believe that perceptrons have such learning capabilities by giving a counter example showing that a perceptron without having hidden layers even could not handle the simple XOR problem. Such a counter example made many researchers run away from artificial neural networks and finally resulted in the "Artificial Intelligence (AI) winter" in 1970s. To our understanding, there is an interesting issue between Rosenblatt's dream and Minsky's counter example. Rosenblatt may not be able to give efficient learning algorithms in the very beginning of neural networks research. Rosenblatt's perceptron is a multilayer feedforward network. In many cases, a feedforward network with input and output layers but without hidden layers is considered as a two-layer perceptron, which were actually used in Minsky and Papert [17]. However, a feedforward network with input and output layers but without hidden layers seems like a "brain" which has input layers (eyes, noses, etc.) and output layers (motor sensors, etc.) but without "central neurons." Obviously, such a "brain" is an empty shell and has no "learning and cognition" capabilities at all. However, Rosenblatt and Minsky's controversy also tells the truth that one small step of development in artificial intelligence and machine learning may request one or several generations' great efforts. Their professional controversy[2] may turn out to indirectly inspire the reviving of artificial neural network research in the end.

Thus, there is no doubt that neural networks research revives after hidden layers are emphasized in learning since 1980s. However, an immediate dilemma in neural network research is that since hidden layers are important and necessary conditions of learning, by default expectation and understanding of neural network research community, hidden neurons of all networks need to be tuned. Thus, since 1980s tens of thousands of researchers from almost every corner of the world have been working hard on looking for learning algorithms to train various types of neural networks mainly by tuning hidden layers. Such a kind of "confusing" research situation turned out to force us to seriously ask several questions as early as 1995 [18]:

1. Do we really need to spend so much manpower and great effort on finding learning algorithms and manually tuning parameters for different neural networks and applications? However, obviously, in contrast there is no "pygmy" sitting in biological brains and tuning parameters there.
2. Do we really need to have different learning algorithms for different types of neural networks in order to achieve good learning capabilities of feature learning, clustering, regression and classification?
3. Why are biological brains more "efficient" and "intelligent" than those machines/computers embedded with artificially designed learning algorithms?
4. Are we able to address John von Neumann's puzzle[3] [19, 20] why "an imperfect neural network, containing many random connections, can be made to perform reliably those functions which might be represented by idealized wiring diagrams?"

No solutions to the above-mentioned problems were found until 2003 after many years of efforts spent. Finally, we found that the key "knot" in the above-mentioned open problems is that

1. The counter example given by Minsky and Papert [17] shows that hidden layers are necessary.

[1] http://en.wikipedia.org/wiki/Perceptron.

[2] Professional controversies should be advocated in academic and research environments; however, irresponsible anonymous attack which intends to destroy harmony research environment and does not help maintain healthy controversies should be refused.

[3] John von Neumann was also acknowledged as a "Father of Computers."

2. Earlier neural networks theories on universal approximation capabilities (e.g., [21, 22]) are also built on the assumption that hidden neurons need to be tuned during learning.
3. Thus, naturally and reasonably speaking, hidden neurons need to be tuned in artificial neural networks.

In order to address the above-mentioned open problems, one has to untie the key "knot," that is, for wide types of networks (artificial neural networks or biological neural networks whose network architectures and neuron modeling are even unknown to human being), hidden neurons are important but do not need to be tuned.

Our such beliefs and philosophy in both machine learning and biological learning finally result in the new techniques referred to extreme learning machines (ELMs) and related ELM theories. As emphasized in Huang et al. [6], 'Extreme' means to move beyond conventional artificial learning techniques and to move toward brain alike learning. ELM aims to break the barriers between the conventional artificial learning techniques and biological learning mechanism. 'Extreme learning machine (ELM)' represents a suite of machine learning techniques (including *single-hidden-layer feedforward networks* and *multi-hidden-layer feedforward networks*) in which hidden neurons do not need to be tuned with the consideration of neural network generalization theory, control theory, matrix theory and linear system theory. To randomly generate hidden nodes is one of the typical implementations which ensures that "hidden neurons do not need to be tuned" in ELM; however, there also exist many other implementations such as kernels [6, 23], SVD and local receptive fields [8]. We believe that ELM reflects the truth of some biological learning mechanisms. Its machine-based learning efficiency was confirmed in 2004 [24], and its universal approximation capability (for "generalized SLFNs" in which a hidden node may be a subnetwork of several nodes and/or with almost any nonlinear piecewise continuous neurons (although their exact mathematical modeling/formula/shapes may be unknown to human beings)) was rigorously proved in theory in 2006–2008 [5, 25, 26]. Its concrete biological evidence subsequently appears in 2011–2013 [27–30].

ELM targets at not only "generalized" single-hidden-layer feedforward networks but also "generalized" multi-hidden-layer feedforward networks in which a node may be a subnetwork consisting of other hidden nodes [5, 8, 26]. Single hidden layer of ELM also covers wide types of neural networks including but not limited to sigmoid networks and RBF networks (refer to "'Generalized' Single-Hidden-Layer Feedforward Networks (SLFNs)" section for details).

Compression, feature learning, clustering, regression and classification are fundamental to machine learning and machine intelligence. ELM aims to implement these five fundamental operations/roles of learning in homogeneous ELM architectures (cf. Fig. 1).

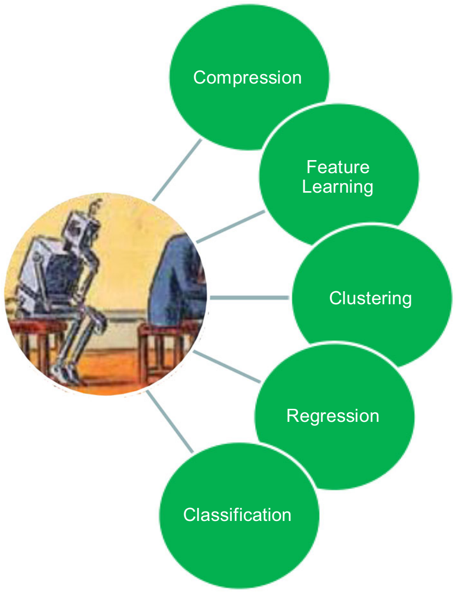


Fig. 1 Fundamental operations/roles of ELM. Courtesy to the anonymous designer who provides the robot icon in Internet

ELM Theories

Although there are few attempts on random sigmoid hidden neurons and/or RBF neurons in 1950s–1990s [1, 2, 9, 31–33], this kind of implementations did not really "take off" except for RVFL [34] due to several reasons:

1. The common understanding and tenet is that hidden neurons of various types of neural networks need to be tuned.
2. There lacks of theoretical analysis except for RVFL.
3. There lacks of strong motivation from biological learning except for Rosenblatt's perceptron.

ELM theories managed to address the challenging issue: "*Whether wide types of neural networks (including biological neural networks) with wide types of hidden nodes/neurons (almost any nonlinear piecewise continuous nodes) can be randomly generated.*" Although ELM aims to deal with both single-hidden-layer feedforward networks (SLFNs) and multi-hidden-layer feedforward networks, its theories have mainly focused on SLFN cases in the past 10 years.

Universal Approximation Capability

Strictly speaking, none of those earlier works (e.g., Baum [31] and Schmidt et al. [1], RVFL [2, 32]) has addressed in theory whether random hidden nodes can be used in their specific sigmoid or RBF networks, let alone the wide type of networks covered by ELM theories. Lowe's [35] RBF network does not use the random impact factor although the centers of their RBF nodes are randomly generated. One has to adjust impact factors based on applications. In other words, semi-random RBF nodes are used in RBF network [35]. Detail analysis has been given in Huang [3].

Both Baum [31] and Schmidt et al. [1] focus on empirical simulations on specific network architectures (a specific case of ELM models).[4] To the best of our knowledge, both earlier works do not have theoretical analysis, let alone the rigorous theoretical proof. Although intuitively speaking, Igelnik and Pao [32] try to prove the universal approximation capability of RVFL, as analyzed in [4, 8], actually Igelnik and Pao [32] only prove RVFL's universal approximation capability when *semi-random sigmoid and RBF hidden nodes* are used, that is, the input weights $\mathbf{a}_i$ are randomly generated, while the hidden node biases b_i are calculated based on the training samples $\mathbf{x}_i$ and the input weights $\mathbf{a}_i$ (refer to Huang et al. [4] for detail analysis).

In contrast, ELM theories have shown that almost any nonlinear piecewise continuous random hidden nodes (including sigmoid and RBF nodes mentioned in those earlier works, but also including wavelet, Fourier series and biological neurons) can be used in ELM, and the resultant networks have universal approximation capabilities [5, 25, 26]. Unlike the semi-random sigmoid and RBF hidden nodes used in the proof of RVFL [32] in which some parameters are not randomly generated, the physical meaning of *random hidden nodes* in ELM theories is that all the parameters of the hidden nodes are randomly generated independently from the training samples, e.g., both random input weights $\mathbf{a}_i$ and biases b_i for additive hidden nodes, or both centers $\mathbf{a}_i$ and impact factor b_i for RBF networks, parameters for Fourier series and wavelets, etc. It is ELM theories first time to show that all the hidden nodes/neurons can be not only independent from training samples but also independent from each other in wide types of neural networks and mathematical series/expansions as well as in biological learning mechanism [5, 6, 25, 26].

Definition 3.1 [5, 25, 26] A hidden layer output mapping $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), \ldots, h_L(\mathbf{x})]$ is said to be an ELM random feature mapping if all its hidden node parameters are randomly generated according to any continuous sampling distribution probability, where $h_i(\mathbf{x}) = G_i(\mathbf{a}_i, b_i, \mathbf{x})$, $i = 1, \ldots, L$ (the number of neurons in the hidden layer)..

Different hidden nodes may have different output functions G_i. In most applications, for the sake of simplicity, same output functions can be chosen for all hidden nodes, that is, $G_i = G_j$ for all $i, j = 1, \ldots, L$.

Theorem 3.1 (Universal approximation capability [5, 25, 26]) *Given any non-constant piecewise continuous function as the activation function, if tuning the parameters of hidden neurons could make SLFNs approximate any target continuous function* $f(\mathbf{x})$, *then the sequence* $\{h_i(\mathbf{x})\}_{i=1}^{L}$ *can be randomly generated according to any continuous distribution probability, and* $\lim_{L\to\infty} \| \sum_{i=1}^{L} \boldsymbol{\beta}_i h_i(\mathbf{x}) - f(\mathbf{x}) \| = 0$ *holds with probability one with appropriate output weights* $\boldsymbol{\beta}$.

Classification Capability

In addition, ELM theories also prove the classification capability of wide types of networks with random hidden neurons, and such theories have not been studied by those earlier works.

Theorem 3.2 (Classification capability [23]) *Given any non-constant piecewise continuous function as the activation function, if tuning the parameters of hidden neurons could make SLFNs approximate any target continuous function* $f(\mathbf{x})$, *then SLFNs with random hidden layer mapping* $\mathbf{h}(\mathbf{x})$ *can separate arbitrary disjoint regions of any shapes.*

Single-Hidden-Layer Feedforward Networks Versus Multi-Hidden-Layer Feedforward Networks

It is difficult to deal with multi-hidden layers of ELM directly without having complete solutions of single hidden layer of ELM. Thus, in the past 10 years, most of ELM works have been focusing on "generalized" single-hidden-layer feedforward networks (SLFNs).

"Generalized" Single-Hidden-Layer Feedforward Networks (SLFNs)

The study by Schmidt et al. [1] focuses on sigmoid networks, and the study by Pao et al. [32] focuses on RVFL (with sigmoid or RBF nodes). Both have strict standard single hidden layers, which are not "generalized" single-hidden-layer feedforward networks (SLFNs) studied in ELM. Similar to SVM [15], the feedforward neural network with random weights proposed in Schmidt et al. [1] requires a bias in the output node in order to absorb the

[4] Schmidt et al. [1] only reported some experimental results on three synthetic toy data as usually done by many researchers in 1980s–1990s; however, it may be difficult for machine learning community to make concrete conclusions based on experimental results on toy data in most cases.

system error as its universal approximation capability with random sigmoid nodes was not proved when proposed:

$$f_L(\mathbf{x}) = \sum_{i=1}^{L} \beta_i g_{\mathrm{sig}}(\mathbf{a}_i \cdot \mathbf{x} + b_i) + b \tag{1}$$

where $g_{\mathrm{sig}}(x) = \frac{1}{1+\exp(-x)}$. Both QuickNet and RVFL have the direct link between the input node and the output node:

$$f_L(\mathbf{x}) = \sum_{i=1}^{L} \beta_i g_{\mathrm{sig\ or\ RBF}}(\mathbf{a}_i, b_i, \mathbf{x}) + \boldsymbol{\alpha} \cdot \mathbf{x} \tag{2}$$

ELM is proposed for "generalized" single-hidden-layer feedforward networks and mathematical series/expansions (which may not be a conventional neural network even, such as wavelet and Fourier series):

$$f_L(\mathbf{x}) = \sum_{i=1}^{L} \beta_i G(\mathbf{a}_i, b_i, \mathbf{x}) \tag{3}$$

The basic ELM is for generalized SLFN, unlike the fully connected networks in those earlier works, there are three levels of randomness in ELM (Fig. 3 for details):

1. Fully connected, hidden node parameters are randomly generated.
2. Connection can be randomly generated, not all input nodes are connected to a particular hidden node. Possibly only some input nodes in some local field are connected to one hidden node.
3. A hidden node itself can be a subnetwork formed by several nodes which naturally forms the local receptive fields and pooling functions, and thus results in learning local features. In this sense, some local parts of a single ELM can contain multi-hidden layers.

Note Unlike Schmidt et al. [1] and Pao et al. [32] in which each node is a sigmoid or RBF node only, each hidden node in ELM can be a subnetwork of other nodes in which feature learning can be implemented efficiently (refer to Huang et al. [8], Figs. 2 and 3 for details).

According to ELM theories[5, 25, 26], ELM SLFNs include but are not limited to:

1. Sigmoid networks
2. RBF networks
3. Threshold networks [36]
4. Trigonometric networks
5. Fuzzy inference systems
6. Fully complex neural networks [37]
7. High-order networks
8. Ridge polynomial networks
9. Wavelet networks
10. Fourier series [5, 6, 25, 26]
11. Biological neurons whose modeling/shapes may be unknown, etc.

Multi-Hidden-Layer Feedforward Networks

However, unlike Schmidt et al. [1] and RVFL [32] which only works for single-hidden-layer feedforward networks, the ultimate tenet of ELM is: Hidden nodes of wide types of multi-hidden-layer networks do not need to be tuned (e.g., [8, 38, 39, 43]) (cf. Fig. 4). Although multilayers of ELM concepts have been given in ELM theories in 2007 [26], it has not been used until recently (e.g., [8, 38, 39, 43]). In essence:

1. Rosenblatt tried to transfer learned behavior from trained rats to naive rats by the injection of brain extracts,[5] which may not consider the fact that different layers of neurons may play different roles. Unlike Rosenblatt's perceptron concept, we think that it is impossible to have all the layers randomly generated. If all layers in a multilayer network are randomly generated, the useful information may not pass through two or more purely random hidden layers. However, each basic ELM can be used in each hidden layer, and hidden neurons do not need to be tuned layer wise, and different layer may have different targets (in terms of ELM's five fundamental operations: compression, feature learning, clustering, regression and classification).
2. The meanings that hidden nodes do not need to be tuned are twofold:
   (a) Hidden nodes may be randomly generated.
   (b) Although hidden nodes do not need to be randomly generated, they need not be tuned either. For example, a hidden node in the next layer can simply be a linear sum or nonlinear transform of some randomly generated nodes in the earlier layer. In this case, some nodes are randomly generated and some are not, but none of them are tuned.[8]
3. Each single ELM can deal with compression, feature learning, clustering, regression or classification. Thus, a homogeneous hierarchical blocks of ELM can be built. For example, one ELM as feature learning, the next ELM works as a classifier. In this case, we have two hidden layers of ELM, overall speaking it is not randomly generated and it is ordered, but hidden nodes in each layer do not need to be tuned (e.g., randomly generated or explicitly given/calculated (Fig. 4a).
4. ELM slices which play feature learning or clustering roles can also be used to link different learning models. Or as an entire networks, some layers are trained by ELM, and some are trained by other models (cf. Fig. 6).

[5] http://en.wikipedia.org/wiki/Frank-Rosenblatt.

Fig. 2 ELM theories [5, 25, 26] show that wide types of hidden nodes can be used in each ELM slice (ELM feature mapping) in which a hidden node can be a subnetwork of several nodes. **a** ELM slice/feature mapping with fully connected random hidden nodes. **b** ELM slice/feature mapping with subnetworks

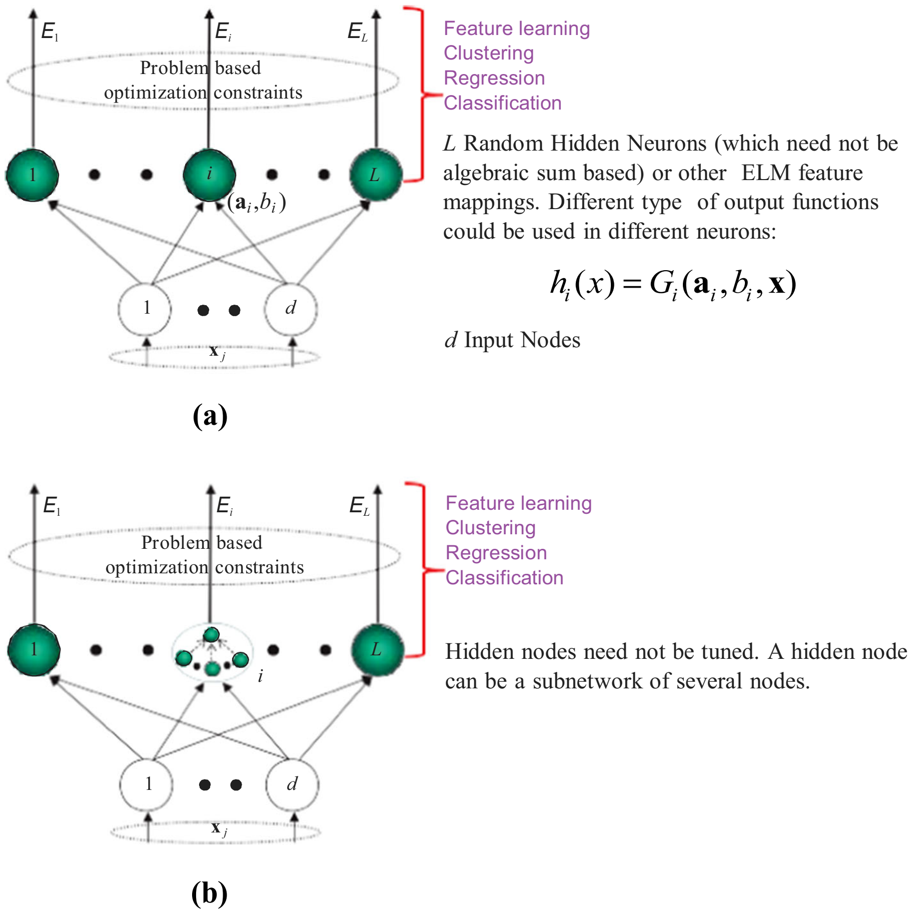
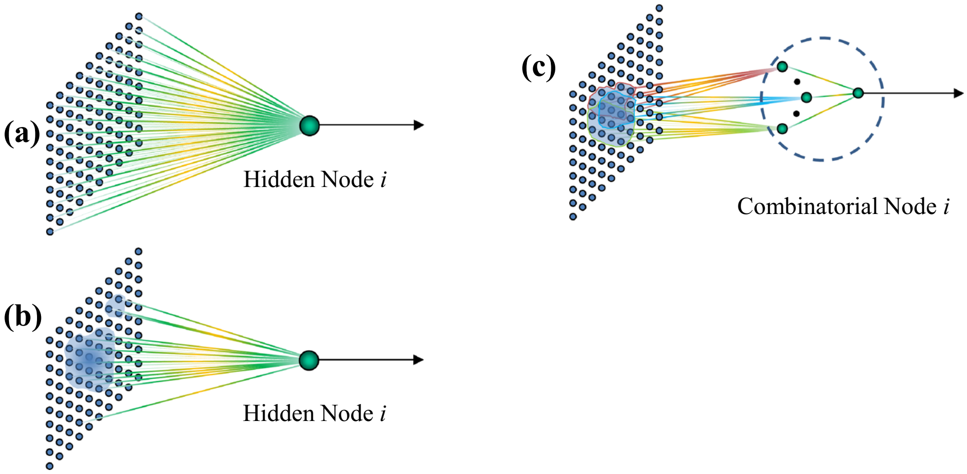

E_1
E_i
E_L
Problem based optimization constraints
Feature learning
Clustering
Regression
Classification
1
i
L
$(\mathbf{a}_i, b_i)$
L Random Hidden Neurons (which need not be algebraic sum based) or other ELM feature mappings. Different type of output functions could be used in different neurons:
$h_i(x) = G_i(\mathbf{a}_i, b_i, \mathbf{x})$
1
d
$\mathbf{x}_j$
d Input Nodes
(a)
E_1
E_i
E_L
Problem based optimization constraints
Feature learning
Clustering
Regression
Classification
1
i
L
Hidden nodes need not be tuned. A hidden node can be a subnetwork of several nodes.
1
d
$\mathbf{x}_j$
(b)

Fig. 3 ELM theories [5, 25, 26] show that wide types of hidden nodes can be used and resultant networks need not be a single-hidden-layer feedforward networks. "Generalized SLFN" referred by ELM means a subnetwork of several nodes. **a** Hidden node in full connection in ELM. **b** Hidden node in local connection/random connection in ELM. **c** Combinatorial node of several nodes in ELM



Relationship and Differences Among ELM, Deep Learning and SVM/LS-SVM

ELM is different from deep Learning in the sense that hidden neurons of the entire ELM do not need to be tuned. Due to ELM's different roles of feature learning and clustering, ELM can be used as the earlier layers in multilayer networks in which the late layers are trained by other methods such as deep learning (cf. Figure 5).

SVM was originally proposed to handle multilayer feedforward networks by Cortes and Vapnik [15] which assumes that when there is no algorithm to train a multilayer network, one can consider the output function of the last hidden layer as $\phi(\mathbf{x})$.

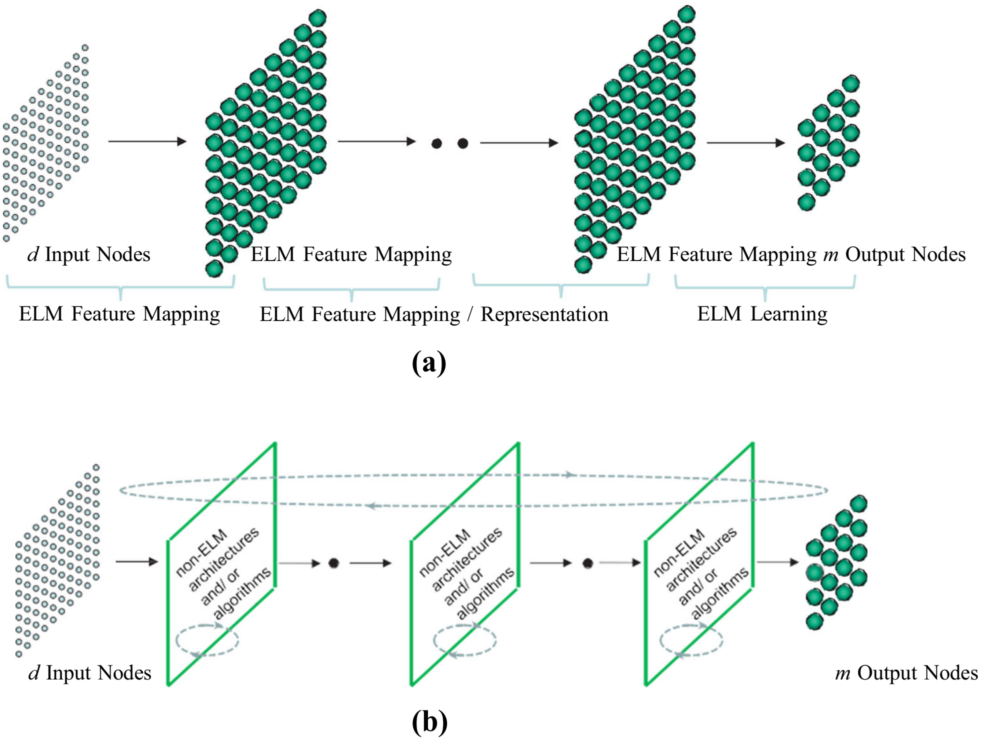

d Input Nodes
ELM Feature Mapping
ELM Feature Mapping
m Output Nodes
ELM Feature Mapping
ELM Feature Mapping / Representation
ELM Learning
(a)
non-ELM architectures and/ or algorithms
non-ELM architectures and/ or algorithms
non-ELM architectures and/ or algorithms
d Input Nodes
m Output Nodes
(b)

Fig. 4 Comparisons between hierarchical ELM and deep learning: Each ELM slice forms one hidden layer and hidden node in some hidden layers could be a subnetwork of several neurons. Unlike deep learning concept, ELM (at its both single hidden layer or multi-hidden layers of architectures) emphasizes in learning without tuning hidden neurons. **a** ELM: Entire network as a big single ELM but with ELM working for each layer. Each layer has feature presentation and is trained without hidden neurons tuned. (e.g., [8, 38, 39]). **b** Deep learning: Feature representations are given in hidden layers. Hidden neurons are iteratively tuned in inner models, and iterative tuning is imposed on the entire networks

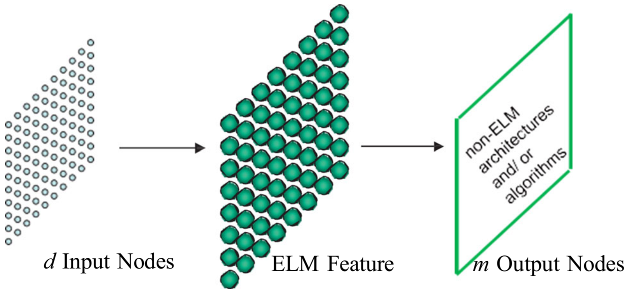


Fig. 5 ELM slice works as the input of other learning models

1. Unlike ELM and deep learning which study feature representations in each layer, SVM and LS-SVM do not consider the feature representation and functioning roles of each inner hidden layer (cf. Figure 7).
2. SVM and LS-SVM can also be considered as single-hidden-layer networks with the hidden layer output function $\phi(\mathbf{x})$. In this case, both ELM and SVM/LS-SVM have single hidden layers. However, ELM has explicit hidden layer mapping $\mathbf{h}(\mathbf{x})$ (convenient for feature representations) and SVM/LS-SVM has unknown hidden layer mapping $\phi(\mathbf{x})$ (inconvenient for feature representations).
3. ELM works for feature learning, clustering regression and classification with ridge regression optimization condition, while SVM/LS-SVM mainly works for binary classification with maximal margin optimization condition. It is difficult for SVM/LS-SVM to have feature representation due to unknown mapping $\phi(\mathbf{x})$ (refer to Table 1 for detail comparisons between ELM and SVM/LS-SVM, and Huang et al. [6, 23] for detail analysis on the reasons why SVM and LS-SVM provide suboptimal solutions in general).

Hidden Neuron Types

Unlike Schmidt et al. [1] and Pao et al. [32] in which each node is a sigmoid or RBF function, ELM is valid for wide types of neural nodes and non-neural nodes. ELM is efficient for kernel learning as well [6, 23].

Real Domain

As ELM has universal approximation capability for a wide type of nonlinear piecewise continuous functions $G(\mathbf{a}, b, \mathbf{x})$, it does not need any bias in the output layer. Some commonly used activation functions covered in ELM theories are:

1. Sigmoid function:

$$G(\mathbf{a}, b, \mathbf{x}) = \frac{1}{1 + \exp(-(\mathbf{a} \cdot \mathbf{x} + b))} \tag{4}$$

2. Fourier function[25, 46]:

$$G(\mathbf{a}, b, \mathbf{x}) = \sin(\mathbf{a} \cdot \mathbf{x} + b) \tag{5}$$

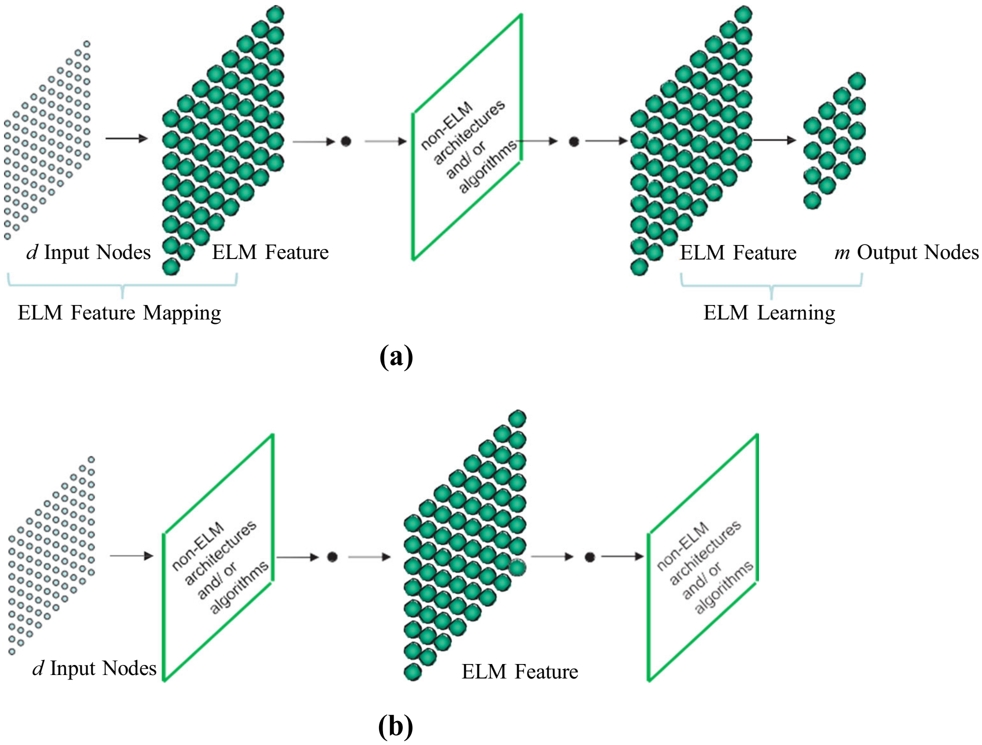

d Input Nodes
ELM Feature
non-ELM architectures and/ or algorithms
ELM Feature
m Output Nodes
ELM Feature Mapping
ELM Learning
(a)
d Input Nodes
non-ELM architectures and/ or algorithms
ELM Feature
non-ELM architectures and/ or algorithms
(b)

Fig. 6 ELM slices work with different learning models: However, each ELM slice as a fundamental learning element can be incorporated into other learning models (e.g., [40–44]). **a** Other learning models work between different ELM slices. **b** ELM slices work between different learning models

3. Hardlimit function [25, 36]:

$$G(\mathbf{a},b,\mathbf{x}) = \begin{cases} 1, & \text{if}\,\mathbf{a}\cdot\mathbf{x} - \mathrm{b} \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

4. Gaussian function [23, 25]:

$$G(\mathbf{a},b,\mathbf{x}) = \exp(-b\|\mathbf{x}-\mathbf{a}\|^2) \tag{7}$$

5. Multi-quadrics function [23, 25]:

$$G(\mathbf{a},b,\mathbf{x}) = (\|\mathbf{x}-\mathbf{a}\|^2 + b^2)^{1/2} \tag{8}$$

6. Wavelet [47, 48]:

$$G(\mathbf{a},b,\mathbf{x}) = \|a\|^{-1/2}\Psi\left(\frac{\mathbf{x}-\mathbf{a}}{b}\right) \tag{9}$$

where Ψ is a single mother wavelet function.

Remark Due to the validity of universal approximation and classification capability on general nonlinear piecewise continuous activation functions, *combinations of different type of hidden neurons can be used in ELM* [49].

Complex Domain

According to Li et al. [4, 37], random hidden nodes used in ELM can be fully complex hidden nodes proposed by Kim and Adali [50], and the resultant ELM in complex domain has the universal approximation capability too. The complex hidden nodes of ELM include but are not limited to:

1. Circular functions:

$$\tan(z) = \frac{e^{iz} - e^{-iz}}{i(e^{iz} + e^{-iz})} \tag{10}$$

$$\sin(z) = \frac{e^{iz} - e^{-iz}}{2i} \tag{11}$$

2. Inverse circular functions:

$$\arctan(z) = \int_0^z \frac{\mathrm{d}t}{1+t^2} \tag{12}$$

$$\arccos(z) = \int_0^z \frac{\mathrm{d}t}{(1-t^2)^{1/2}} \tag{13}$$

3. Hyperbolic functions:

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{14}$$

$$\sinh(z) = \frac{e^z - e^{-z}}{2} \tag{15}$$

4. Inverse hyperbolic functions:

$$\operatorname{arctanh}(z) = \int_0^z \frac{\mathrm{d}t}{1-t^2} \tag{16}$$

$$\operatorname{arcsinh}(z) = \int_0^z \frac{\mathrm{d}t}{(1+t^2)^{1/2}} \tag{17}$$

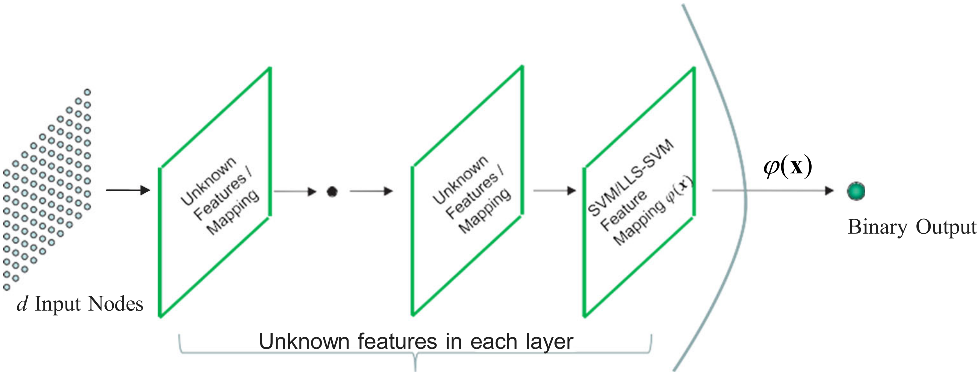

d Input Nodes
Unknown Features / Mapping
Unknown Features / Mapping
SVM/LS-SVM Feature Mapping φ(x)
φ(x)
Binary Output
Unknown features in each layer

Fig. 7 Relationships and differences between ELM, SVM/LS-SVM and deep learning: Unlike ELM and deep leaning, (1) SVM/LS-SVM as multilayers of networks does not emphasize feature representations in hidden layers; and (2) SVM/LS-SVM only handles binary cases directly in their original formula

Regularization Network and Generalization Performance

Similar to most conventional learning algorithms proposed in 1980s–1990s, Schmidt et al. [1] and Pao et al. [32] focus on minimizing training errors only. They are not regularization networks.[6]

However, inspired by neural networks generalization performance theories proposed in 1998 [14], which are published after Schmidt et al. [1] and Pao et al. [32], ELM theory aims to reach the smallest training error but also the smallest norm of output weights [24, 53] (in this sense, generally speaking, ELM is a kind of regularization neural networks but with non-tuned hidden layer mappings (formed by either random hidden nodes, kernels or other implementations)):

$$\text{Minimize: } \|\boldsymbol{\beta}\|_p^{\sigma_1} + C\|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_q^{\sigma_2} \tag{18}$$

where $\sigma_1 > 0, \sigma_2 > 0, p, q = 0, \frac{1}{2}, 1, 2, \ldots, +\infty$. *Different combinations of* $\|\boldsymbol{\beta}\|_p^{\sigma_1}$ and $\|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_q^{\sigma_2}$ *can be used and result in different learning algorithms for feature learning and clustering* [7]. **H** is the ELM hidden layer output matrix (*randomized matrix*):

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}(\mathbf{x}_1) \\ \vdots \\ \mathbf{h}(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} G(\mathbf{a}_1, b_1, \mathbf{x}_1) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_1) \\ \vdots & \vdots & \vdots \\ G(\mathbf{a}_1, b_1, \mathbf{x}_N) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_N) \end{bmatrix} \tag{19}$$

and **T** is the training data target matrix:

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix} = \begin{bmatrix} t_{11} & \cdots & t_{1m} \\ \vdots & \vdots & \vdots \\ t_{N1} & \cdots & t_{Nm} \end{bmatrix} \tag{20}$$

One can linearly apply many ELM solutions (but not all) to the specific sigmoid network with b (Schmidt et al. [1]) and a network with direct link from the input layer to the output network (including but not limited to QuickNet [54] and RVFL [2]); suboptimal solutions will be reached compared to the original ELM. The resultant learning algorithms can be referred to ELM+b and ELM+$\alpha\mathbf{x}$, respectively (refer to Huang et al. [6, 23] for details).

For RVFL, the hidden layer output matrix is:

$$\begin{aligned} &\mathbf{H}_{\text{RVFL}} \\ &= \begin{bmatrix} g_{\text{sig,RBF}}(\mathbf{a}_1, b_1, \mathbf{x}_1) & \cdots & g_{\text{sig,RBF}}(\mathbf{a}_L, b_L, \mathbf{x}_1) & \mathbf{x}_1 \\ \vdots & \vdots & \vdots & \vdots \\ g_{\text{sig,RBF}}(\mathbf{a}_1, b_1, \mathbf{x}_N) & \cdots & g_{\text{sig,RBF}}(\mathbf{a}_L, b_L, \mathbf{x}_N) & \mathbf{x}_N \end{bmatrix} \\ &= \left[\mathbf{H}_{\text{ELM for sigmoid or RBF basis}} \quad \mathbf{X}_{N\times d}\right] \end{aligned} \tag{21}$$

where $\mathbf{H}_{\text{ELM for sigmoid or RBF basis}}$ are two specific ELM hidden layer output matrices (19) with sigmoid or RBF basis, and $\mathbf{X}_{N\times d}$ is a $N \times d$ matrix with i-th input $\mathbf{x}_i$ as the i-th row.

If the output neuron bias is considered as a bias neuron in the hidden layer as done in most conventional neural networks, the hidden layer output matrix for Schmidt et al. [1] will be

$$\begin{aligned} &\mathbf{H}_{\text{Schmidt, et al. (1992)}} \\ &= \begin{bmatrix} g_{\text{sig}}(\mathbf{a}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & g_{\text{sig}}(\mathbf{a}_L \cdot \mathbf{x}_1 + b_L) & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & & & \vdots \\ g_{\text{sig}}(\mathbf{a}_1 \cdot \mathbf{x}_N + b_1) & \cdots & g_{\text{sig}}(\mathbf{a}_L \cdot \mathbf{x}_N + b_L) & 1 & \cdots & 1 \end{bmatrix} \\ &= \left[\mathbf{H}_{\text{ELM for sigmoid basis}} \quad \mathbf{1}_{N\times m}\right] \end{aligned} \tag{22}$$

where $\mathbf{H}_{\text{ELM for sigmoid basis}}$ is a specific ELM hidden layer output matrix (19) with sigmoid basis, and $\mathbf{1}_{N\times m}$ is a $N \times m$

[6] Chen et al. [51, 52] provide some interesting learning algorithms for RVFL networks and suggested that regularization could be used to avoid overfitting. Their works are different from structural risk minimization and maximum margin concept adopted in SVM. ELM's regularization objective moves beyond maximum margin concept, and ELM is able to unify neural network generalization theory, structural risk minimization, control theory, matrix theory and linear system theory in ELM learning models (refer to Huang [6, 23] for detail analysis).

Table 1 Relationship and difference comparison between ELM and SVM/LS-SVM

Properties	ELMs	SVM	LS-SVM
Belief	Unlike conventional learning theories and common understanding, ELM belief: Learning can be made without tuning hidden neurons in wide type of biological learning mechanisms and wide types of neural networks	No such belief (Original assumption [15]: If there is no learning solution for feedforward networks, one only needs to consider the output of the last hidden layer: $\phi(\mathbf{x})$)	No such belief (Original assumption [15]: If there is no learning solution for feedforward networks, one only needs to consider the output of the last hidden layer: $\phi(\mathbf{x})$)
Biological inspired	Yes (Confirmed in rats' olfactory system/visual system)	No	No
Network output functions	$f_L(\mathbf{x}) = \sum_{i=1}^{L} \beta_i G(\mathbf{a}_i, b_i, \mathbf{x})$	$f(\mathbf{x}) = \sum_{s=1}^{N_s} \alpha_s t_s \phi(\mathbf{x}) \cdot \phi(\mathbf{x}_s) + b$	$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i t_i \phi(\mathbf{x}) \cdot \phi(\mathbf{x}_i) + b$
Multi-class classification	Direct solutions	Indirect solutions based on binary ($t_i = 0$ or 1) case	Indirect solutions based on binary ($t_i = 0$ or 1) case
Explicit feature mappings	Yes (Wide types of explicit feature mappings $\mathbf{h}(\mathbf{x})$. Kernels can also be used.)	No (Unknown mapping $\phi(\mathbf{x})$, kernel only.)	No (Unknown mapping $\phi(\mathbf{x})$, kernel only.)
Hidden node types (mathematical model)	Wide types (sigmoid, kernel, Fourier series, etc.)	Kernels	Kernels
Hidden node types (biological neurons)	Yes	No	No
Domain	Both real and complex domains	Real domain (Difficult in handling complex domain directly)	Real domain (Difficult in handling complex domain directly)
SLFNs	"Generalized" SLFN Wide types of SLFNs	No	No
Layer wise feature representation	Yes	No (Feature representations in different layers are ignored)	No (Feature representations in different layers are ignored)
Connectivity	For both fully connected and randomly (partially) connected network	No attention on network connections	No attention on network connections
Hyperplane constraints in dual problem	No (It has no such hyperplane constraints due to lack of bias b in output nodes.)	Yes (It has such hyperplane constraints due to bias b in output nodes.)	Yes (It also provides the model without b but it does still assume binary class. [45])
Universal approximation and classification capability	Proved theoretically for wide types of nonlinear piecewise nodes/neurons	No theoretical proof	No theoretical proof

Table 1 continued

Properties	ELMs	SVM	LS-SVM
Ridge regression theory	Yes (Consistent for feature learning, clustering, regression and binary/multi-class classification.)	No (Maximal margin concept is a specific case of ridge regression theory used in ELM for binary classification.)	No (Maximal margin concept is a specific case of ridge regression theory used in ELM forbinary classification.)
Learning capability	Efficient in feature learning (auto-encoders) and clustering	Difficult in handling auto-encoders	Difficult in handling auto-encoders
Solutions	Closed-form and non-closed-form, online, sequential and incremental	Non-closed-form	Closed-form

matrix with constant element 1. Although bias b in Schmidt et al. [1] seems like a simple parameter, however, it is known that from both mathematical and machine learning point of view, a parameter may result in some significant differences. Its role has drawn researchers' attention [6, 55, 56]. In fact, one of the main reasons why it is difficult to apply SVM and LS-SVM in multi-class applications in the past two decades is mainly due to the output node bias b. Without the output node bias b , SVM and LS-SVM solutions would become much easier [6, 23].

Closed-Form Solutions Versus Non-closed-Form Solutions

In many cases, closed-form solutions of ELM can be given when (18) $\sigma_1 = \sigma_2 = p = q = 2$. However, non-closed-form solutions can also be given if $\sigma_1 = \sigma_2 = p = q = 2$ [5, 13, 25, 26, 57] or if other values are given to σ_1,σ_2, p, and q , especially when ELM is used in the applications of compression, feature learning and clustering [5, 7, 25, 26, 58–60]. Actually the original proof on the universal approximation capability of ELM is based on non-closed-form solutions of ELM [5, 25, 26].

Conclusions

It has been around 60 years since Frank Rosenblatt [9] dreamed that his perceptron could enable computers to "walk, talk, see, write, reproduce itself and be conscious to its existence." It was difficult for many researchers to believe his great dream due to lack of efficient learning algorithms and strong theoretical support in the very beginning of artificial neural network era. On the other hand, John von Neumann was puzzled [19, 20] why "an imperfect neural network, containing many random connections, can be made to perform reliably those functions which might be represented by idealized wiring diagrams"[9]. This paper shows that ELM theories and framework may fill such a gap between Frank Rosenblatt's dream and John von Neumann's puzzle:

1. ELM can be used to train wide type of multi-hidden layer of feedforward networks: Each hidden layer can be trained by one single ELM based on its role as feature learning, clustering, regression or classification. Entire network as a whole can be considered as a single ELM in which hidden neurons need not be tuned (refer to Fig. 8 for the detail summary of ELM).
2. ELM slice can be "inserted" into many local parts of a multi-hidden-layer feedforward network, or work together with other learning architectures/models.

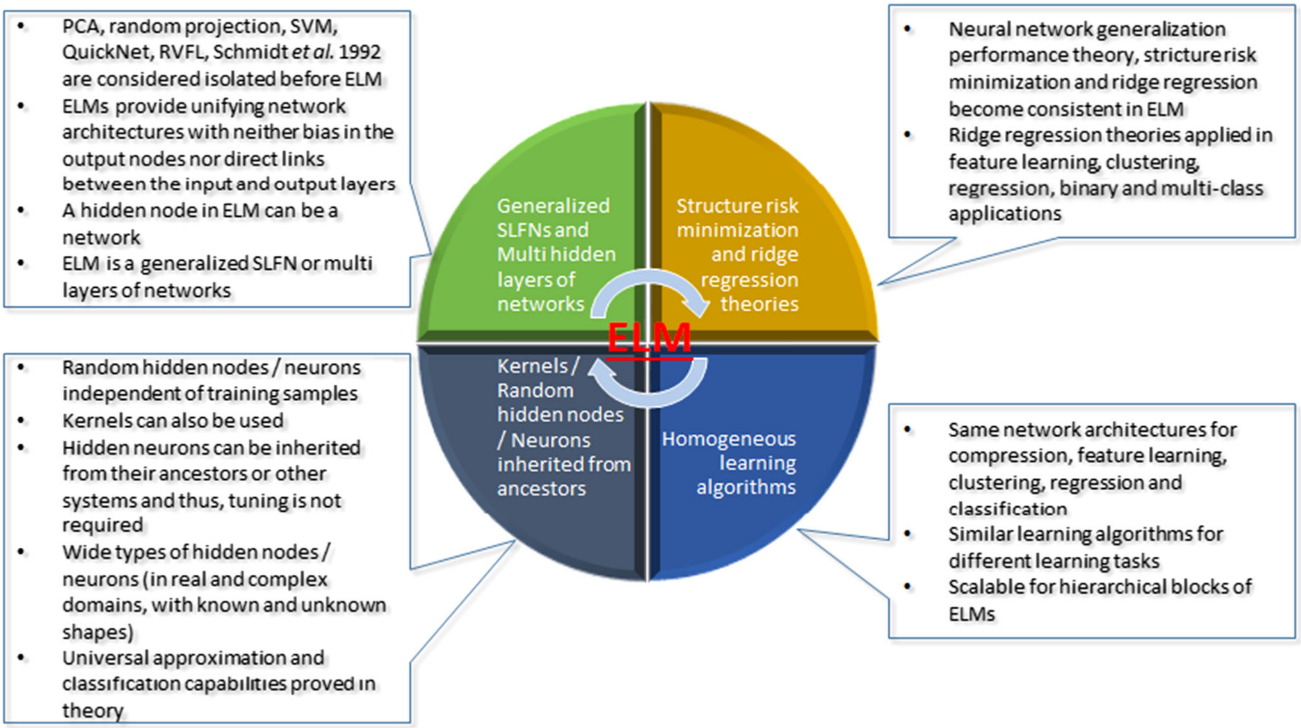

• PCA, random projection, SVM, QuickNet, RVFL, Schmidt *et al.* 1992 are considered isolated before ELM
• ELMs provide unifying network architectures with neither bias in the output nodes nor direct links between the input and output layers
• A hidden node in ELM can be a network
• ELM is a generalized SLFN or multi layers of networks
• Neural network generalization performance theory, stricture risk minimization and ridge regression become consistent in ELM
• Ridge regression theories applied in feature learning, clustering, regression, binary and multi-class applications
Generalized SLFNs and Multi hidden layers of networks
Structure risk minimization and ridge regression theories
ELM
Kernels / Random hidden nodes / Neurons inherited from ancestors
Homogeneous learning algorithms
• Random hidden nodes / neurons independent of training samples
• Kernels can also be used
• Hidden neurons can be inherited from their ancestors or other systems and thus, tuning is not required
• Wide types of hidden nodes / neurons (in real and complex domains, with known and unknown shapes)
• Universal approximation and classification capabilities proved in theory
• Same network architectures for compression, feature learning, clustering, regression and classification
• Similar learning algorithms for different learning tasks
• Scalable for hierarchical blocks of ELMs

Fig. 8 Essential elements of ELM

3. A hidden node in an ELM slice (a "generalized" SLFN) can be a network of several nodes; thus, local receptive fields can be formed.
4. In each hidden layer, input layers to hidden nodes can be fully or partially randomly connected according to different continuous probability distribution function. The performance of the network is stable even if a finite number of hidden neurons and their related connections change.

Thus, overall speaking, from ELM theories point of view, the entire multilayers of networks are structured and ordered, but they may be seemingly "messy" and "unstructured" in a particular layer or neuron slice. "Hard wiring" can be randomly built locally with full connection or partial connections. Coexistence of globally structured architectures and locally random hidden neurons happen to have fundamental learning capabilities of compression, feature learning, clustering, regression and classification. This may have addressed John von Neumann's puzzle. Biological learning mechanisms are sophisticated, and we believe that "learning without tuning hidden neurons" is one of the fundamental biological learning mechanisms in many modules of learning systems. Furthermore, random hidden neurons and "random wiring" are only two specific implementations of such "learning without tuning hidden neurons" learning mechanisms.

Acknowledgments Minsky and Rosenblatt's controversy may have indirectly inspired the reviving of artificial neural network research in 1980s. Their controversy turned out to show that hidden neurons are critical. Although the main stream of research has focused learning algorithms on tuning hidden neurons since 1980s, few pioneers independently studied alternative solutions (e.g., Halbert White for QuickNet, Yao-Han Pao, Boris Igelnik, and C. L. Philip Chen for RVFL, D. S. Broomhead and David Lowe for RBF networks, Corinna Cortes and Vladimir Vapnik for SVM, J. A. K. Suykens and J. Vandewalle for LS-SVM, E. Baum, Wouter F. Schmidt, Martin A. Kraaijveld and Robert P. W. Duin for Random Weights Sigmoid Network) in 1980s–1990s. Although few of those attempts did not take off finally due to various reasons and constrains, they have been playing significant and irreplaceable roles in the relevant research history. We would appreciate their historical contributions. As discussed in Huang [6], without BP and SVM/LS-SVM, the research and applications on neural networks would never have been so intensive and extensive. We would like to thank Bernard Widrow, Stanford University, USA, for sharing his vision in neural networks and precious historical experiences with us in the past years. It is always pleasant to discuss with him on biological learning, neuroscience and Rosenblatt's work. We both feel confident that we have never been so close to natural biological learning. We would like to thank C. L. Philip Chen, University of Macau, China and Boris Igelnik, BMI Research, Inc., USA, for invaluable discussions on RVFL, Johan Suykens, Katholieke Universiteit Leuven, Belgium, for invaluable discussions on LS-SVM and Stefano Fusi, Columbia University, USA, for the discussion on the links between biological learning and ELM, and Wouter F. Schmidt and Robert P. W. Duin for the kind constructive feedback on the discussions between ELM and their 1992 work. We would also like to thank Jose Principe, University of Florida, USA, for nice discussions on neuroscience (especially on neuron layers/slices) and his invaluable suggestions on potential mathematical problems of ELM and M. Brandon Westover, Harvard Medical School, USA, for the constructive comments and suggestions on the potential links between ELM and biological learning in local receptive fields.

Appendix: Further Clarification of Misunderstandings on the Relationship Between ELM and Some Earlier Works

Recently, it has drawn our attention that several researchers have been making some very negative and unhelpful comments on ELM in neither academic nor professional manner due to various reasons and intentions, which mainly state that ELM does not refer to some earlier works (e.g., Schmidt et al. [1] and RVFL), and ELM is the same as those earlier works. It should be pointed out that these earlier works have actually been referred in our different publications on ELM as early as in 2007. It is worth mentioning that ELM actually provides a unifying learning platform for wide types of neural networks and machine learning techniques by absorbing the common advantages of many seemingly isolated different research efforts made in the past 60 years, and thus, it may not be surprising to see apparent relationships between ELM and different techniques. As analyzed in this paper, the essential differences between ELM and those mentioned related works are subtle but crucial.

It is not rare to meet some cases in which intuitively speaking some techniques superficially seem similar to each other, but actually they are significantly different. ELM theories provide a unifying platform for wide types of neural networks, Fourier series [25], wavelets [47, 48], mathematical series [5, 25, 26], etc. Although the relationship and differences between ELM and those earlier works have clearly been discussed in the main context of this paper, in response to the anonymous malign letter, some more background and discussions need to be highlighted in this appendix further.

Misunderstanding on References

Several researchers thought that ELM community has not referred to those earlier related work, e.g., Schmidt et al. [1], RVFL [2] and Broomhead and Lowe [61]. We wish to draw their serious attention that our earlier work (2008) [4] has explicitly stated: "Several researchers, e.g., Baum [31], Igelnik and Pao [32], Lowe [35], Broomhead and Lowe [61], Ferrari and Stengel [62], have independently found that the input weights or centers $\mathbf{a}_i$ do not need to be tuned" (these works were published in different years, one did not refer to the others. The study by Ferrari and Stengel [62] has been kindly referred in our work although it was even published later than ELM [24]). In addition, in contrast to the misunderstanding that those earlier works were not referred, we have even referred to Baum [31]'s work and White's QuickNet [54] in our 2008 works [4, 5], which we consider much earlier than Schmidt et al. [1] and RVFL [2] in the related research areas. There is also a misunderstanding that Park and Sandberg's RBF theory [21] has not been referred in ELM work. In fact, Park and Sandberg's RBF theory has been referred in the proof of ELM's theories on RBF cases as early as 2006 [25].

Although we did not know Schmidt et al. [1] until 2012, we have referred to it in our work [6] immediately. We spent almost 10 years (back to 1996) on proving ELM theories and may have missed some related works. However, from literature survey point of view, Baum [31] may be the earliest related work we could find so far and has been referred at the first time. Although the study by Schmidt et al. [1] is interesting, the citations of Schmidt et al. [1] were almost zero before 2013 (Google Scholar), and it is not easy for his work to turn up in search engine unless one intentionally flips hundreds of search pages. Such information may not be available in earlier generation of search engine when ELM was proposed. The old search engines available in the beginning of this century were not as powerful as most search engines available nowadays and many publications were not online 10–15 years ago. As stated in our earlier work [4], Baum [31] claimed that (seen from simulations) one may fix the weights of the connections on one level and simply adjust the connections on the other level, and no (significant) gain is possible by using an algorithm able to adjust the weights on both levels simultaneously. Surely, almost every researcher knows that the easiest way is to calculate the output weights by least-square method (closed-form) as done in Schmidt et al. [1] and ELM if the input weights are fixed.

Loss of Feature Learning Capability

The earlier works (Schmidt et al. [1], RVFL [2], Broomhead and Lowe [61]) may lose learning capability in some cases.

As analyzed in our earlier work [3, 4], although Lowe [35]'s RBF network chooses RBF network centers randomly, it uses one value b for all the impact factors in all RBF hidden nodes, and such a network will lose learning capability if the impact factor b is randomly generated. Thus, in RBF network implementation, the single value of impact factors is usually adjusted manually or based on cross-validation. In this sense, Lower's RBF network does not use random RBF hidden neurons, let alone wide types of ELM networks.[7] Furthermore, Chen et al. [64] point out

[7] Differences between Lowe's RBF networks and ELM have been clearly given in our earlier reply [3] in response to another earlier comment letter on ELM [63]. It is not clear why several researchers in their anonymous letter refer to the comment letter on ELM [63] for Lowe [35]'s RBF network and RVFL but do not give readers right and clear information by referring to the response [3].

Table 2 Relationship and difference comparison among different methods: ELMs, Schmidt et al. [1], QuickNet/RVFL

Properties	ELMs	Schmidt et al. [1]	QuickNet/RVFL
Belief	Unlike conventional learning theories and common understanding, ELM belief: Learning can be made without tuning hidden neurons in wide type of biological learning mechanisms and wide types of neural networks	No such belief	No such belief
Network output functions	$f_L(\mathbf{x}) = \sum_{i=1}^{L} \beta_i G(\mathbf{a}_i, b_i, \mathbf{x})$	$f_L(\mathbf{x}) = \sum_{i=1}^{L} \beta_i g_{\mathrm{sig}}(\mathbf{a}_i \cdot \mathbf{x} + b_i) + b$	$f_L(\mathbf{x}) = \sum_{i=1}^{L} \beta_i g_{\mathrm{sig\,or\,RBF}} + \alpha \cdot \mathbf{x}$
SLFNs	"Generalized" SLFN in which a hidden node can be a subnetwork	Standard SLFN only	Standard SLFN plus direct links between input layer to the output layer
Multilayer networks	Yes	No	No
Connectivity	For both fully connected and randomly (partially) connected network	Fully connected	Fully connected
Hidden node types (mathematical model)	Wide types (sigmoid, kernel, Fourier series, etc.)	Sigmoid	Sigmoid and RBF
Hidden node types (biological neurons)	Yes	No	No
Domain	Both real and complex domains	Real domain	Real domain
Hidden layer output matrix	$\mathbf{H}_{\mathrm{ELM}}$	$[\mathbf{H}_{\mathrm{ELM\,for\,sigmoid\,basis}}, \mathbf{1}_{N\times m}]$	$[\mathbf{H}_{\mathrm{ELM\,for\,sigmoid\,or\,RBF\,basis}}, \mathbf{X}_{N\times d}]$
Universal approximation and classification capability	Proved for wide types of random neurons	No theoretical proof	Theoretical proof for semi-random sigmoid or RBF nodes
Structural risk minimization	Minimize $: \|\boldsymbol{\beta}\|_{\mathrm{p}}^{\sigma_1} + \mathrm{C}\|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_{\mathrm{q}}^{\sigma_2}$	Not considered	Not considered
Learning capability	Efficient in feature learning (auto-encoders) and clustering	Difficult in handling auto-encoders	Lose learning capability in auto-encoders
Solutions	Closed-form and non-closed-form, sequential and incremental	Closed-form	Closed-form and non-closed-form for QuickNet, Closed-form for RVFL
Portability	Many (*but not all*) ELM variants can be linearly extended to Schmidt et al. [1] and RVFL/QuickNet instead of vice versa, the resultant algorithms are referred to as "ELMs+b" for Schmidt et al. [1] and "ELM+$\alpha\mathbf{x}$" for QuickNet/RVFL		

that such Lowe [35, 61]'s RBF learning procedure may not be satisfactory and thus they proposed an alternative learning procedure to choose RBF node centers one by one in a rational way which is also different from random hidden nodes used by ELM.

Schmidt et al. [1] at its original form may face difficulty in sparse data applications; however, one can linearly extend sparse ELM solutions to Schmidt et al. [1] (the resultant solution referred to ELM+b).

ELM is efficient for auto-encoder as well [39]. However, when RVFL is used for auto-encoder, the weights of the direct link between its input layer to its output layer will become a constant value one and the weights of the links between its hidden layer to its output layer will become a constant value zero; thus, RVFL will lose learning capability in auto-encoder cases. Schmidt et al. [1] which has the biases in output nodes may face difficulty in auto-encoder cases too.

It may be difficult to implement those earlier works (Schmidt et al. [1], RVFL [2], Broomhead and Lowe [61]) in multilayer networks, while hierarchical ELM with multi-ELM each working in one hidden layer can be considered as a single ELM itself. Table 2 summarizes the relationship and main differences between ELM and those earlier works.

References

1. Schmidt WF, Kraaijveld MA, Duin RPW. Feed forward neural networks with random weights. In: Proceedings of 11th IAPR international conference on pattern recognition methodology and systems, Hague, Netherlands, p. 1–4, 1992.
2. Pao Y-H, Park G-H, Sobajic DJ. Learning and generalization characteristics of the random vector functional-link net. Neurocomputing. 1994;6:163–80.
3. Huang G-B. Reply to comments on 'the extreme learning machine'. IEEE Trans Neural Netw. 2008;19(8):1495–6.
4. Huang G-B, Li M-B, Chen L, Siew C-K. Incremental extreme learning machine with fully complex hidden nodes. Neurocomputing. 2008;71:576–83.
5. Huang G-B, Chen L. Enhanced random search based incremental extreme learning machine. Neurocomputing. 2008;71:3460–8.
6. Huang G-B. An insight into extreme learning machines: random neurons, random features and kernels. Cogn Comput. 2014;6(3):376–90.
7. Huang G, Song S, Gupta JND, Wu C. Semi-supervised and unsupervised extreme learning machines. IEEE Trans Cybern. 2014;44(12):2405–17.
8. Huang G-B, Bai Z, Kasun LLC, Vong CM. Local receptive fields based extreme learning machine. IEEE Comput Intell Mag. 2015;10(2):18–29.
9. Rosenblatt F. The perceptron: a probabilistic model for information storage and organization in the brain. Psychol Rev. 1958;65(6):386–408.
10. Rahimi A, Recht B. Random features for large-scale kernel machines. In: Proceedings of the 2007 neural information processing systems (NIPS2007), p. 1177–1184, 3–6 Dec 2007.
11. Le Q, Sarlós T, Smola A. Fastfood approximating kernel expansions in loglinear time. In: Proceedings of the 30th international conference on machine learning, Atlanta, USA, p. 16–21, June 2013.
12. Huang P-S, Deng L, Hasegawa-Johnson M, He X. Random features for kernel deep convex network. In: Proceedings of the 38th international conference on acoustics, speech, and signal processing (ICASSP 2013), Vancouver, Canada, p. 26–31, May 2013.
13. Widrow B, Greenblatt A, Kim Y, Park D. The no-prop algorithm: a new learning algorithm for multilayer neural networks. Neural Netw. 2013;37:182–8.
14. Bartlett PL. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. IEEE Trans Inform Theory. 1998;44(2):525–36.
15. Cortes C, Vapnik V. Support vector networks. Mach Learn. 1995;20(3):273–97.
16. Suykens JAK, Vandewalle J. Least squares support vector machine classifiers. Neural Process Lett. 1999;9(3):293–300.
17. Minsky M, Papert S. Perceptrons: an introduction to computational geometry. Cambridge: MIT Press; 1969.
18. Huang G-B. Learning capability of neural networks. Ph.D. thesis, Nanyang Technological University, Singapore, 1998.
19. von Neumann J. Probabilistic logics and the synthesis of reliable organisms from unreliable components. In: Shannon CE, McCarthy J, editors. Automata studies. Princeton: Princeton University Press; 1956. p. 43–98.
20. von Neumann J. The general and logical theory of automata. In: Jeffress LA, editor. Cerebral mechanisms in behavior. New York: Wiley; 1951. p. 1–41.
21. Park J, Sandberg IW. Universal approximation using radial-basis-function networks. Neural Comput. 1991;3:246–57.
22. Leshno M, Lin VY, Pinkus A, Schocken S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Netw. 1993;6:861–7.
23. Huang G-B, Zhou H, Ding X, Zhang R. Extreme learning machine for regression and multiclass classification. IEEE Trans Syst Man Cybern B. 2012;42(2):513–29.
24. Huang G-B, Zhu Q-Y, Siew C-K. Extreme learning machine: a new learning scheme of feedforward neural networks. In: Proceedings of international joint conference on neural networks (IJCNN2004), vol. 2, Budapest, Hungary, p. 985–990, 25–29 July 2004.
25. Huang G-B, Chen L, Siew C-K. Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans Neural Netw. 2006;17(4):879–92.
26. Huang G-B, Chen L. Convex incremental extreme learning machine. Neurocomputing. 2007;70:3056–62.
27. Sosulski DL, Bloom ML, Cutforth T, Axel R, Datta SR. Distinct representations of olfactory information in different cortical centres. Nature. 2011;472:213–6.
28. Eliasmith C, Stewart TC, Choo X, Bekolay T, DeWolf T, Tang Y, Rasmussen D. A large-scale model of the functioning brain. Science. 2012;338:1202–5.
29. Barak O, Rigotti M, Fusi S. The sparseness of mixed selectivity neurons controls the generalization–discrimination trade-off. J Neurosci. 2013;33(9):3844–56.
30. Rigotti M, Barak O, Warden MR, Wang X-J, Daw ND, Miller EK, Fusi S. The importance of mixed selectivity in complex cognitive tasks. Nature. 2013;497:585–90.
31. Baum E. On the capabilities of multilayer perceptrons. J Complex. 1988;4:193–215.
32. Igelnik B, Pao Y-H. Stochastic choice of basis functions in adaptive function approximation and the functional-link net. IEEE Trans Neural Netw. 1995;6(6):1320–9.

33. Tamura S, Tateishi M. Capabilities of a four-layered feedforward neural network: four layers versus three. IEEE Trans Neural Netw. 1997;8(2):251–5.
34. Principle J, Chen B. Universal approximation with convex optimization: gimmick or reality? IEEE Comput Intell Mag. 2015;10(2):68–77.
35. Lowe D. Adaptive radial basis function nonlinearities and the problem of generalisation. In: Proceedings of first IEE international conference on artificial neural networks, p. 171–175, 1989.
36. Huang G-B, Zhu Q-Y, Mao KZ, Siew C-K, Saratchandran P, Sundararajan N. Can threshold networks be trained directly? IEEE Trans Circuits Syst II. 2006;53(3):187–91.
37. Li M-B, Huang G-B, Saratchandran P, Sundararajan N. Fully complex extreme learning machine. Neurocomputing. 2005;68:306–14.
38. Tang J, Deng C, Huang G-B. Extreme learning machine for multilayer perceptron. IEEE Trans Neural Netw Learn Syst. 2015;. doi:10.1109/TNNLS.2015.2424995.
39. Kasun LLC, Zhou H, Huang G-B, Vong CM. Representational learning with extreme learning machine for big data. IEEE Intell Syst. 2013;28(6):31–4.
40. Jarrett K, Kavukcuoglu K, Ranzato M, LeCun Y. What is the best multi-stage architecture for object recognition. In: Proceedings of the 2009 IEEE 12th international conference on computer vision, Kyoto, Japan, 29 Sept–2 Oct 2009.
41. Saxe AM, Koh PW, Chen Z, Bhand M, Suresh B, Ng AY. On random weights and unsupervised feature learning. In: Proceedings of the 28th international conference on machine learning, Bellevue, USA, 28 June–2 July 2011.
42. Cox D, Pinto N. Beyond simple features: a large-scale feature search approach to unconstrained face recognition. In: IEEE international conference on automatic face and gesture recognition and workshops. IEEE, p. 8–15, 2011.
43. McDonnell MD, Vladusich T. Enhanced image classification with a fast-learning shallow convolutional neural network. In: Proceedings of international joint conference on neural networks (IJCNN'2015), Killarney, Ireland, 12–17 July 2015.
44. Zeng Y, Xu X, Fang Y, Zhao K. Traffic sign recognition using extreme learning classifier with deep convolutional features. In: The 2015 international conference on intelligence science and big data engineering (IScIDE 2015), Suzhou, China, June 14–16, 2015.
45. Suykens JAK, Gestel TV, Brabanter JD, Moor BD, Vandewalle J. Least squares support vector machines. Singapore: World Scientific; 2002.
46. Rahimi A, Recht B. Uniform approximation of functions with random bases. In: Proceedings of the 2008 46th annual allerton conference on communication, control, and computing, p. 555–561, 23–26 Sept 2008
47. Daubechies I. Orthonormal bases of compactly supported wavelets. Commun Pure Appl Math. 1988;41:909–96.
48. Daubechies I. The wavelet transform, time-frequency localization and signal analysis. IEEE Trans Inform Theory. 1990;36(5):961–1005.
49. Miche Y, Sorjamaa A, Bas P, Simula O, Jutten C, Lendasse A. OP-ELM: optimally pruned extreme learning machine. IEEE Trans Neural Netw. 2010;21(1):158–62.
50. Kim T, Adali T. Approximation by fully complex multilayer perceptrons. Neural Comput. 2003;15:1641–66.
51. Chen CLP. A rapid supervised learning neural network for function interpolation and approximation. IEEE Trans Neural Netw. 1996;7(5):1220–30.
52. Chen CLP, Wan JZ. A rapid learning and dynamic stepwise updating algorithm for flat neural networks and the applications to time-series prediction. IEEE Trans Syst Man Cybern B Cybern. 1999;29(1):62–72.
53. Huang G-B, Zhu Q-Y, Siew C-K. Extreme learning machine: theory and applications. Neurocomputing. 2006;70:489–501.
54. White H. An additional hidden unit test for neglected nonlinearity in multilayer feedforward networks. In: Proceedings of the international conference on neural networks, p. 451–455, 1989.
55. Poggio T, Mukherjee S, Rifkin R, Rakhlin A, Verri A. "*b*", A.I. Memo No. 2001–011, CBCL Memo 198, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 2001.
56. Steinwart I, Hush D, Scovel C. Training SVMs without offset. J Mach Learn Res. 2011;12(1):141–202.
57. Luo J, Vong C-M, Wong P-K. Sparse bayesian extreme learning machine for multi-classification. IEEE Trans Neural Netw Learn Syst. 2014;25(4):836–43.
58. Decherchi S, Gastaldo P, Leoncini A, Zunino R. Efficient digital implementation of extreme learning machines for classification. IEEE Trans Circuits Syst II. 2012;59(8):496–500.
59. Bai Z, Huang G-B, Wang D, Wang H, Westover MB. Sparse extreme learning machine for classification. IEEE Trans Cybern. 2014;44(10):1858–70.
60. Frénay B, van Heeswijk M, Miche Y, Verleysen M, Lendasse A. Feature selection for nonlinear models with extreme learning machines. Neurocomputing. 2013;102:111–24.
61. Broomhead DS, Lowe D. Multivariable functional interpolation and adaptive networks. Complex Syst. 1988;2:321–55.
62. Ferrari S, Stengel RF. Smooth function approximation using neural networks. IEEE Trans Neural Netw. 2005;16(1):24–38.
63. Wang LP, Wan CR. Comments on 'the extreme learning machine'. IEEE Trans Neural Netw. 2008;19(8):1494–5.
64. Chen S, Cowan CFN, Grant PM. Orthogonal least squares learning algorithm for radial basis function networks. IEEE Trans Neural Netw. 1991;2(2):302–9.