Neurocomputing 176 (2016) 3–13

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A Robust Extreme Learning Machine for pattern classification with outliers

Guilherme A. Barreto [a,*], Ana Luiza B.P. Barros [b]

[a] Federal University of Ceará, Department of Teleinformatics Engineering, Av. Mister Hull, S/N - Center of Technology, Campus of Pici, CP 6005, CEP 60455-760 Fortaleza, Ceará, Brazil

[b] State University of Ceará, Department of Computer Science, Av. Paranjana, 1700, Campus of Itaperi, Fortaleza, Ceará, Brazil

ARTICLE INFO

Article history:
Received 1 April 2014
Received in revised form
19 October 2014
Accepted 21 October 2014
Available online 6 May 2015

Keywords:
Extreme Learning Machine
Ordinary least squares
Least mean squares
Robust pattern classification
Outliers
M-estimation

ABSTRACT

In this paper we introduce a simple and efficient extension of the Extreme Learning Machine (ELM) network (Huang et al., 2006 [19]), which is very robust to label noise, a type of outlier occurring in classification tasks. Such outliers usually result from mistakes during labeling of the data points (e.g. misjudgment of a specialist) or from typing errors during creation of data files (e.g. by striking an incorrect key on a keyboard). The proposed variant of the ELM, henceforth named *Robust ELM* (RELM), is designed using *M*-estimators to compute the output weights instead of the standard ordinary least squares (OLS) method. We evaluate the performance of the RELM using batch and recursive learning rules, and also introduce a model selection strategy based on Particle Swarm Optimization (PSO) to find an optimal architecture for datasets contaminated with non-Gaussian noise and outliers. By means of comprehensive computer simulations using synthetic and real-world datasets, we show that the proposed Robust ELM classifiers consistently outperforms the original version.

1. Introduction

In recent years, a class of supervised feedforward neural network model introduced by Huang and colleagues [17,19], called Extreme Learning Machine (ELM), has attracted a great deal of attention from the machine learning community [30–34,14,45,8]. All this interest in ELM seems to be primarily motivated by the very fast way it is trained, without resorting to a long and tedious learning process, such as that required by the error backpropagation algorithm.

More specifically, an important feature of the ELM is that the input-to-hidden-layer weights are randomly chosen, i.e. the input-to-hidden transformation imposed to the input vectors is in fact a random mapping. This characteristic allows the hidden-to-output-layer weights to be computed via a simple batch learning scheme, such as the standard ordinary least squares (OLS) method [37], although sequential learning schemes, such as the least mean squares (LMS) algorithm [41] or the recursive least squares (RLS) algorithm [29], have also been proposed.

Despite the fact that several authors have been successfully applying the ELM to a number of complex pattern classification and regression problems, it should be noted that these works have not consistently addressed the issue of model performance in the presence of outliers, which are instances considered to be *exogenous* to the input–output data model actually learned for the task of interest. In other words, outliers do not belong to (or are not generated by) the actual process that give rise to the data.

The design of outlier-robust machine learning methods is a complex research topic where sparse contributions in the field of neural networks can be found, including proposals for designing Radial Basis Functions (RBF) networks [25,26], echo-state networks [28], and even one for ELM networks [14]. However, these few works have addressed the issue of robustness to outliers only for regression problems, not for pattern classification.

Outliers in classification problems can be roughly distinguished as attribute (i.e. feature) noise, or class (i.e. label) noise [44]. Attribute noise affects the observed values of the input patterns during the measurement process. Class noise in its turn changes the labels assigned to instances, e.g. by incorrectly setting a positive label on a negative instance in binary classification scenario. It is worth mentioning, however, that despite the growing interest in outlier-robust classification techniques (see [12] for a recent survey of such techniques), outliers often go unnoticed because pattern classification is being more and more automatically executed by computers, without careful inspection or screening. Thus, it urges to implement techniques that can handle them suitably, either by previously elimination from the training data or by reduction of their consequences if training data are unreliable.

* Corresponding author.
E-mail addresses: gbarreto@ufc.br (G.A. Barreto), analuiza@larces.uece.br (A.L.B.P. Barros).

http://dx.doi.org/10.1016/j.neucom.2014.10.095
0925-2312/

While outlier removal tends to improve the quality of the data modeling in both regression [38] and classification tasks [1], it may end up destroying important information in the data that the user are not aware of. For instance, a small cloud of points appearing in a region previously uncovered with data may look like initially as outliers, but as more points populate that region as time goes by, this cloud may acquire informative meaning, since its occurrence may be due to parameter drift phenomenon [2] or to a novel (i.e. previously unmodeled) data cluster/class.

Bearing this in mind, we argue that a better approach is to handle outliers automatically by designing robust learning rules. For this purpose, it is important to understand that a common feature shared by the aforementioned learning rules (i.e. OLS, LMS and RLS) is optimality only under the assumption of Gaussianity of the error distribution. However, the presence of attribute and/or label noise in the data causes the error distribution to depart from Gaussianity and hence the classifier performance deteriorates considerably. Thus, we aim at introducing outlier-robust extensions of the ELM capable of handling outliers efficiently. The proposed variants of the ELM classifier, henceforth named *Robust ELM* (RELM) classifiers, will be designed using *M*-estimators to compute the output weights instead of the standard OLS/LMS algorithms.

M-estimation is a broad robust statistics framework [21,37,3,38] widely used for parameter estimation in regression-like problems, such as adaptive filtering, dynamical system identification, time series prediction and function approximation, when the Gaussianity assumption for the prediction errors (i.e. residuals) does not hold. Despite being a classical approach to handle outliers in regression, the possibility of extending the *M*-estimation framework to the context of neural network based robust pattern classification have not been addressed so far. This can be confirmed by analyzing the contributions listed in the recent survey by Frenay and Verleysen [12], who do not list the *M*-estimation framework as one of the robust approaches to handle label noise.

More specifically, we comprehensively evaluate the performances of the resulting RELM classifiers using both batch and recursive learning rules, and also introduce a model selection strategy based on Particle Swarm Optimization (PSO) [22] to find an optimal architecture for datasets contaminated with non-Gaussian attribute or label noise. By means of comprehensive computer simulations using synthetic and real-world datasets, we show that the proposed Robust ELMs consistently outperforms their original versions.

The remainder of the paper is organized as follows. In Section 2, we briefly review the fundamentals of ELM in the context of pattern classification. Then, in Section 3 we describe the basic ideas and concepts behind the *M*-estimation framework. In this section we also introduce two robust variants of the ELM classifier and a PSO-based model selection strategy for inducing a Robust ELM classifier. In Section 4 we present the computer experiments we carried out using synthetic and real-world datasets and also discuss the achieved results. The paper is concluded in Section 5.

2. Fundamentals of the ELM network

Let us assume that N data pairs $\{(\mathbf{x}_n, \mathbf{d}_n)\}_{n=1}^{N}$ are available for building and evaluating the model, where $\mathbf{x}_n \in \mathbb{R}^p$ is the n-th p-dimensional input pattern and $\mathbf{d}_n \in \mathbb{R}^C$ is the corresponding target class label, with C denoting the number of classes. For the labels, we assume an 1-of-C encoding scheme, i.e. for each label vector $\mathbf{d}_n$, the component whose index corresponds to the class of pattern $\mathbf{x}_n$ is set to "$+1$", while the other $C-1$ components are set to "-1".

Then, let us randomly select N_1 ($N_1 < N$) training data pairs from the available data pool and arrange them along the columns of the matrices $\mathbf{D}$ and $\mathbf{X}$ as follows:

$$\mathbf{X} = [\mathbf{x}_1|\mathbf{x}_2|\cdots|\mathbf{x}_{N_1}] \quad \text{and} \quad \mathbf{D} = [\mathbf{d}_1|\mathbf{d}_2|\cdots|\mathbf{d}_{N_1}]. \tag{1}$$

where $\dim(\mathbf{X}) = p \times N_1$ and $\dim(\mathbf{D}) = C \times N_1$.

The ELM is a single-hidden layer feedforward network (SLFN), proposed by Huang et al. [19,16], whose weights from the inputs to the hidden neurons are randomly chosen, while only the weights from the hidden neurons to the output are analytically determined. Consequently, ELM offers significant advantages such as fast learning speed, ease of implementation, and less human intervene when compared to more traditional SLFNs, such as the Multilayer Perceptrons (MLP) and RBF networks. For a network with p input units, q hidden neurons and C outputs, the i-th output for the n-th input pattern is given by

$$y_{in} = \boldsymbol{\beta}_i^T \mathbf{h}_n, \tag{2}$$

where $\boldsymbol{\beta}_i \in \mathbb{R}^q, i = 1, \ldots, C$, is the weight vector connecting the hidden neurons to the i-th output neuron, and $\mathbf{h}_n \in \mathbb{R}^q$ is the vector of hidden neurons' outputs for the n-th input pattern $\mathbf{x}_n \in \mathbb{R}^p$. The vector $\mathbf{h}_n$ itself is defined as

$$\mathbf{h}_n = [f(\mathbf{w}_1^T \mathbf{x}_n + b_1), \ldots, f(\mathbf{w}_q^T \mathbf{x}_n + b_q)]^T, \tag{3}$$

where b_l, $l = 1, \ldots, q$, is the bias of the l-th hidden neuron, $\mathbf{w}_l \in \mathbb{R}^p$ is the weight vector of the l-th hidden neuron and $f(\cdot)$ is a sigmoid activation function. The weight vectors $\mathbf{w}_l$ are randomly sampled from either a uniform or normal distribution.

Let $\mathbf{H} = [\mathbf{h}_1 \ \mathbf{h}_2 \ \cdots \ \mathbf{h}_{N_1}]$ be a $q \times N_1$ matrix whose N_1 columns are the hidden-layer output vectors $\mathbf{h}_n \in \mathbb{R}^q$, $n = 1, \ldots, N_1$, where N_1 is the number of available training input patterns. Then, let $\mathbf{D} = [\mathbf{d}_1 \ \mathbf{d}_2 \ \cdots \ \mathbf{d}_{N_1}]$ be a $C \times N_1$ matrix whose n-th column is the target vector $\mathbf{d}_n \in \mathbb{R}^C$ associated with the input pattern $\mathbf{x}_n$, $n = 1, \ldots, N_1$. Finally, let $\boldsymbol{\beta} = [\boldsymbol{\beta}_1 \ \boldsymbol{\beta}_2 \ \cdots \ \boldsymbol{\beta}_C]$ be a $q \times C$ matrix, whose i-th column is the weight vector $\boldsymbol{\beta}_i \in \mathbb{R}^q$, $i = 1, \ldots, C$.

Thus, these three matrices are related by the following linear mapping:

$$\mathbf{D} = \boldsymbol{\beta}^T \mathbf{H}, \tag{4}$$

where the matrices $\mathbf{D}$ and $\mathbf{H}$ are known, while the weight matrix β is not. The OLS solution of the linear system in Eq. (4) is given by the Moore–Penrose generalized inverse [13] as

$$\boldsymbol{\beta} = \left(\mathbf{H}\mathbf{H}^T\right)^{-1} \mathbf{H}\mathbf{D}^T. \tag{5}$$

Eq. (5) can be split into C individual estimation equations, one for each output neuron i, being written as

$$\boldsymbol{\beta}_i = \left(\mathbf{H}\mathbf{H}^T\right)^{-1} \mathbf{H}\mathbf{D}_i^T, \quad i = 1, \ldots, C, \tag{6}$$

where $\mathbf{D}_i$ denotes the i-th row of matrix $\mathbf{D}$.

In several real-world problems the matrix $\mathbf{H}\mathbf{H}^T$ can be singular, impairing the use of Eq. (5). In fact, a near singular $\mathbf{H}\mathbf{H}^T$ (yet invertible) matrix is also a problem, because it can lead to numerically unstable results. To avoid both problems, a common approach involves the use of the ridge regression method (a.k.a. Tikhonov regularization) [18,6], which is given by

$$\boldsymbol{\beta}_i = (\mathbf{H}\mathbf{H}^T + \lambda \mathbf{I})^{-1} \mathbf{H}\mathbf{D}_i^T, \quad i = 1, \ldots, C, \tag{7}$$

where the constant $\lambda > 0$ is the regularization parameter.

It is worth mentioning that the OLS estimation rules shown in Eqs. (6) and (7) require the storage of all N_1 training input vectors and the corresponding target vectors in order to estimate the output weight vectors $\boldsymbol{\beta}_i$. However, in some applications, such as channel equalization and recursive identification, adaptive (i.e. sequential) learning rules are a better option, where the vector of parameters $\boldsymbol{\beta}_i$ is modified following the arrival of each input pattern [24,29]. The input pattern is then discarded after being used for updating the parameters. Also, a common requirement for