A Novel AdaBoost Framework With Robust Threshold and Structural Optimization

Peng-Bo Zhang and Zhi-Xin Yang, *Member, IEEE*

***Abstract*—The AdaBoost algorithm is a popular ensemble method that combines several weak learners to boost generalization performance. However, conventional AdaBoost.RT algorithms suffer from the limitation that the threshold value must be manually specified rather than chosen through a self-adaptive mechanism, which cannot guarantee a result in an optimal model for general cases. In this paper, we present a generic AdaBoost framework with robust threshold mechanism and structural optimization on regression problems. The error statistics of each weak learner on one given problem dataset is utilized to automate the choice of the optimal cut-off threshold value. In addition, a special single-layer neural network is employed to provide a second opportunity to further adjust the structure and strength the adaption capability of the AdaBoost regression model. Moreover, to consolidate the theoretical foundation of AdaBoost algorithms, we are the first to conduct a rigorous and comprehensive theoretical analysis on the proposed approach. We prove that the general bound on the empirical error with a fraction of training examples is always within a limited soft margin, which indicates that our novel algorithm can avoid over-fitting. We further analyze the bounds on the generalization error directly under probably approximately correct learning. The extensive experimental verifications on the UCI benchmarks have demonstrated that the performance of the proposed method is superior to other state-of-the-art ensemble and single learning algorithms. Furthermore, a real-world indoor positioning application has also revealed that the proposed method has higher positioning accuracy and faster speed.**

***Index Terms*—AdaBoost algorithm, bounds of empirical error and generalization error, ensemble method, indoor positioning system, robust threshold, special single-layer neural network, structural optimization.**

I. Introduction

As a GENERAL paradigm in machine learning, the ensemble algorithm combines various simple classifiers into a unified and integrated classifier with better stability and accuracy performance than any individual learner. In the pioneering work of Hansen and Salamon [1], a set of neural networks with ensemble consensus mechanism demonstrated superior classification performance than other learning algorithms with one single neural network. Since then, a wide categorizes of ensemble technologies have been explored, such as Bagging [2], Boosting [3], Arc-x4 [4], and random forests [5]. Through analysis of the working mechanism of network ensemble, Zhou *et al.* [6] identified that simply using all available neural networks may not as good as an ensemble of appropriately selected set of them. As one of the most prevailing ensemble methods, the boosting algorithm jointly uses several simple classifiers, called weak learners, to generate a stronger classifier than any single one [7]. The subsequently developed adaptive boosting algorithms (AdaBoost) [8]–[10] further improved on robustness against conventional boosting approaches. Instead of relying on a very large training dataset, AdaBoost algorithms could reuse the identical training dataset in every iteration via randomly replacing subsamples with iteratively updated sample distributions. Moreover, the combination of weak classifiers adopted a weighted majority voting strategy, where the voting weights are adaptive to their corresponding training errors. More investigations on AdaBoost and its massive variations were recently reported in various fields [11]–[14].

On regression problems, a series of AdaBoost methods were were presented and utilized as the ensemble method to enhance the generalization capability in various domains [15]–[17]. AdaBoost.M2 was extended to be AdaBoost.R [3], of which the *ad hoc* modification resulted in AdaBoost.R2 [18]. The special boosting algorithm for regression problems, called AdaBoost.RT [8], [19], employed a constant cut-off threshold. Subsequently, a modified AdaBoost.RT algorithm [20] was introduced for predication of temperature in steel manufacturing. It preset a manually selected value as the initial threshold which was then adjusted self-adaptively for different weak learners. Kummer and Najjaran [21] proposed an AdaBoost.MRT algorithm with a singularity-free and variance-scaled threshold for multivariate estimation. The AdaBoost.MRT algorithm solved the singularity in the misclassification function and achieved excellent performance in the presence of noise. It also extended the original AdaBoost.RT to multivariate output via assigning a specific weight to each output variable of each training sample. However, similar to its predecessor, AdaBoost.MRT also relied on manual selection of the threshold. Recently, Zhang and Yang [22] proposed RAE-ELM to boost the learning performance of extreme learning machine (ELM). In general, most of the existing works regarding AdaBoost.RT have strived to introduce a well

Manuscript received April 5, 2016; revised August 5, 2016 and October 13, 2016; accepted October 24, 2016. Date of publication November 24, 2016; date of current version November 15, 2017. This work was supported by the University of Macau under Grant MYRG2015-00077-FST and Grant MYRG2016-00160-FST. This paper was recommended by Associate Editor Y. Tan. *(Corresponding author: Zhi-Xin Yang.)*

The authors are with the Department of Electromechanical Engineering, Faculty of Science and Technology, University of Macau, Macau 999078, China (e-mail: pengbolong123@126.com; zxyang@umac.mo).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2016.2623900

adaptive threshold into the AdaBoost algorithm to increase the generalization performance. However, a true self-adaptive thresholding mechanism without empirical intervention is still missing. On the other hand, despite the enormous emergence of applications of the AdaBoost.RT algorithm in various regression problems, to date, the theoretical analysis of the AdaBoost.RT algorithm is almost vacuous.

To address these open problems, we propose a novel AdaBoost.RT framework with a robust threshold mechanism and structural optimization, which is a generic framework rather than being configured to any specific weak learner. The proposed approach tackles how to automatically select an optimal threshold according to the characteristics of the dataset without empirical suggestions. Unlike existing AdaBoost.RT algorithms in which the initial threshold value is manually chosen and thus may not be optimal, the proposed approach utilizes error statistics method to define a self-adaptive robust threshold. At each iteration, the robust threshold assigned to each weak learner is defined as the scaled standard deviation of the approximation errors. Thus, the error statistics based self-adaptive threshold mechanism could adjust the cut-off threshold automatically according to its performance on the given dataset. Such mechanism is more robust and accurate than previous threshold determination methods used in existing AdaBoost.RT algorithms. Similar to typical machine learning algorithms on classification problems, the proposed method could also generate two parallel hyper-planes based on the statistical property of the dataset, which are the more optimal boundaries of the feature space where the samples lie. Therefore, the proposed robust AdaBoost.RT algorithm with such optimal boundaries owns superior generalization performance than existing ones.

ELM is an emerging learning algorithm based on single-hidden layer feedforward neural networks with randomly assigned hidden parameters between the input layer and the hidden layer [23]–[26]. The output weights for labeling could be calculated analytically using the pseudo-inverse technology. Many of its variations and the multidomain applications are investigated widely and deeply [27], [28]. Huang *et al.* [26] demonstrated that ELM is able to approximate any continuous target function, and the approximate error trends to zero. ELM without bias b for practical and universal approximation has milder optimization constraints than support vector machine (SVM). Compared with the binary classification initialized SVM [29], ELM was introduced originally for regression problems with random feature mapping. Since ELM fulfills the universal approximation condition for a learning method, it is reasonable to employ ELM as the regression model to predict the target function.

Therefore, we originally propose to further enhance the resulted AdaBoost.RT model with neural networks based structural optimization. ELM is utilized to provide each weak learner with a second round of fine tuning opportunity for performance enhancement. The motivation of this method is inspired by structural optimization methods for scene categorization [30] and data partition learning [31]. The former one proposed an ISABoost algorithm that could adjust the inner structure of weak learners before determining their fusion weights. The model became similar to the standard AdaBoost algorithm after weight assignment. The parameter vectors of inner weight and bias were optimized using genetic algorithms to provide each weak learner additional chance to be strengthened. The latter algorithm was the parent-offspring progressive learning method, which applied multiple ELMs to the separated groups of data points for learning and determining the corresponding point clusters. The elegant method achieved excellent generalization performance in data partition learning. In this paper, we propose a different structure optimization strategy. Unlike the ISABoost, in which the inner structures of weak learners need to be modified prior to the assignment of their fusion weights, our method adopts an ELM network to further approximate the target function upon the trained boosting structure of weak learners. In this paper, ELM could tune outputs of weak learners of robust AdaBoost.RT so as to better match the targeting regression function. Even if the AdaBoost.RT module does not reach the best performance on some exceptional datasets, such downstream structural optimization module could further boost the overall regression performance. The process of structural optimization is detailed in Section II.

Furthermore, the novel robust AdaBoost.RT algorithm has excellent performance in the presence of noise. To theoretically reveal the generalization capability of the proposed method under noise, a soft maximum margin [32] is analyzed. Two slack variables are defined for soft maximum margins, and the robust AdaBoost.RT algorithm can then achieve good generalization performance in highly noisy regimes, such as outliers, overlapping class probability distributions, mislabeled patterns, etc. We further prove that a more general bound on the empirical error of the novel robust AdaBoost.RT algorithm with a fraction of training examples is within a limited soft margin, which demonstrates that this novel method can avoid over-fitting.

Generally speaking, the empirical error in a single hypothesis is considered as an unbiased estimator of the generalization error. However, in a set of hypotheses, the empirical error will underestimate the generalization error because the empirical error is biased. Therefore, following the proof of the empirical error bound, we further analyze bounds on the generalization error of the robust AdaBoost.RT algorithm directly under probably approximately correct (PAC) learning. PAC learning [33] is a popular framework for the theoretical analysis of the generalization error of learning algorithms. The PAC theory introduces a growth function, that can be tightly upper-bounded by the VC-dimension [29], [34], as an appropriate measure of complexity. In this paper, we prove that the robust AdaBoost.RT algorithm can drive the generalization error arbitrarily close to zero using the PAC learning theory.

The merits of this paper are as follows.

1) To the best of our knowledge, a novel and generic ensemble framework is first proposed for any weak learner. The statistical characteristics of a weak learner's prediction error on a given dataset is used to automate the choice of the optimal cut-off threshold value.

2) After assigning a weight to each weak learner, the structure of the robust AdaBoost.RT algorithm is further adjusted to be a stronger regression model by means of ELM, which is originally utilized for structure optimization.
3) To demonstrate the underlying theory of the proposed algorithm in a high noisy regime, a more general bound for the robust AdaBoost.RT algorithm with fraction of training examples is proved to be within a limited soft margin.
4) Following the proof of the empirical error bound, the bounds on the generalization error of the novel algorithm under PAC learning is further analyzed.
5) The robustness and generalization performance of the novel AdaBoost framework has been verified through experimental studies. It not only surpasses the prevailing ensemble methods and single learning algorithms on various UCI benchmark problems with different data scales and dimensional levels, but also outperforms several state-of-the-art machine learning methods in a real-world indoor positioning application.

The remainder of this paper is organized as follows. The proposed robust AdaBoost.RT algorithm and ELM-based structural optimization method are introduced in Section II. Section III analyzes a general bound on robust AdaBoost.RT. The generalization error of robust AdaBoost.RT is then proved in Section IV. The experiments and comparisons with state-of-the-art methods based on the benchmark UCI dataset and a real-world indoor positioning application are discussed in Section V. Finally, conclusive remarks are provided in the last section.

II. Framework of Robust AdaBoost.RT and Structural Optimization

This paper presents a novel AdaBoost framework of robust threshold with structural optimization. In Fig. 1, the framework is composed of two submodules: 1) the AdaBoost.RT algorithm with robust threshold and 2) the ELM-based structural optimization. Given m samples $(\mathbf{x}_i, y_i)$, where $i = 1, \ldots, m$; $\mathbf{x_i} \times y_i \in R^l \times R$; for each weak learning machine, WL_t, $(t = 1, \ldots, T)$, the proposed AdaBoost.RT algorithm calculates a regression model $f_t(\mathbf{x})$, and assigns a corresponding weight α_t. The submodule outputs a weighted ensemble of these regression functions $f_t(\mathbf{x})$. For the sake of robustness to training dataset, the proposed AdaBoost.RT algorithm could automatically generate initial cut-off threshold and self-adjust its value to optimal one in each iteration. The statistical characteristics of a weak learners prediction error on m samples are used for the self-adaption. Such robust threshold determination mechanism utilizes the prediction error rate and calculates the statically varied threshold as follows:

$$\varepsilon_t = \sum_{i \in P} p_i^t \tag{1}$$

$$P = \left\{ i \middle| |e_t(i) - \bar{e}_t| > \frac{\sigma_t}{2} \right\}, i \in [1, m] \tag{2}$$

where $\bar{e}_t$ is the expected value, σ_t is the standard deviation, and $(\sigma_t/2)$ stands for the *robust threshold*.

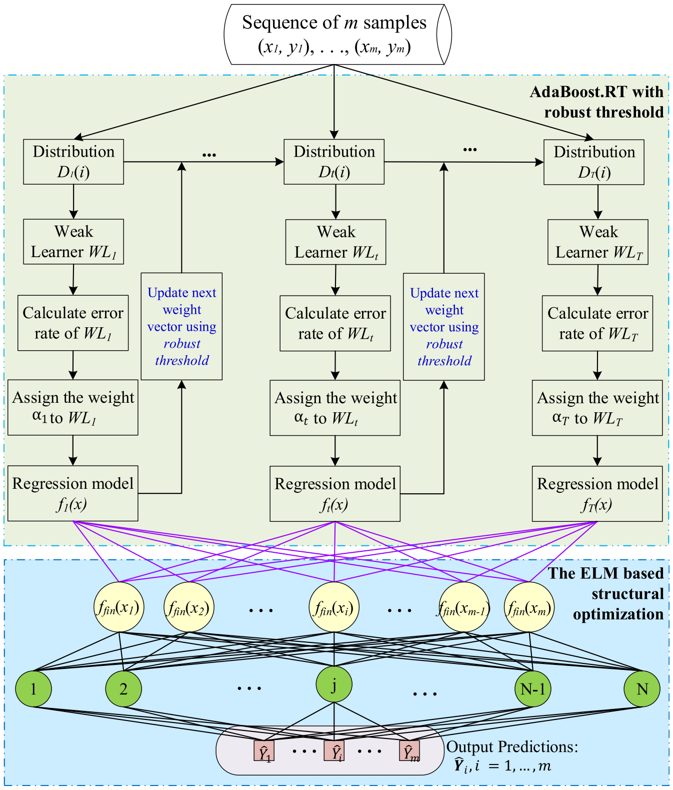
Sequence of m samples
$(x_1, y_1), \ldots, (x_m, y_m)$
AdaBoost.RT with robust threshold
Distribution $D_1(i)$
...
Distribution $D_t(i)$
...
Distribution $D_T(i)$
Weak Learner WL_1
Weak Learner WL_t
Weak Learner WL_T
Update next weight vector using robust threshold
Update next weight vector using robust threshold
Calculate error rate of WL_1
Calculate error rate of WL_t
Calculate error rate of WL_T
Assign the weight α_1 to WL_1
Assign the weight α_t to WL_t
Assign the weight α_T to WL_T
Regression model $f_1(x)$
Regression model $f_t(x)$
Regression model $f_T(x)$
The ELM based structural optimization
$f_{fin}(x_1)$
$f_{fin}(x_2)$
$f_{fin}(x_i)$
$f_{fin}(x_{m-1})$
$f_{fin}(x_m)$
1
2
j
N-1
N
$\hat{Y}_1$
$\hat{Y}_i$
$\hat{Y}_m$
Output Predictions:
$\hat{Y}_i, i = 1, \ldots, m$

Fig. 1. The framework of the novel robust AdaBoost.RT with structural optimization.

The ELM-based structural optimization submodule is subsequent to the robust AdaBoost.RT algorithm, which optimizes its ensemble weighting structure. The special feed-forward neural network gives a few poor weak learners one additional opportunity to be adjusted for a greater contribution in combining a stronger regression model.

A. Robust AdaBoost.RT Method

A scaled standard deviation is defined as the robust threshold for each weak learner. For those training samples with prediction errors exceeding the above defined threshold value, they are considered as "misregressed" and should be rejected. The filtered out samples are to be processed as *ad hoc* cases by the subsequent weak learners. Compared with the existing AdaBoost.RT algorithms, the proposed robust threshold mechanism is completely self-adaptive through calculating the regression distribution of the input dataset, as demonstrated in Fig. 2.

Through analysis of boosting regression estimators [35] and BEM [36], the absolute relative error (ARE) is introduced to compare with the threshold, which could demarcates samples into correct predictions and incorrect ones. The original AdaBoost.RT algorithm defines ARE as the error rate as follows:

$$\varepsilon_t = \text{ARE}_t(i) = \left| \frac{f_t(\mathbf{x}_i) - y_i}{y_i} \right|. \tag{3}$$

Obviously, there is a singularity in (3) when the true value $y_i = 0$. Hence, Kummer and Najjaran [21] revealed that values

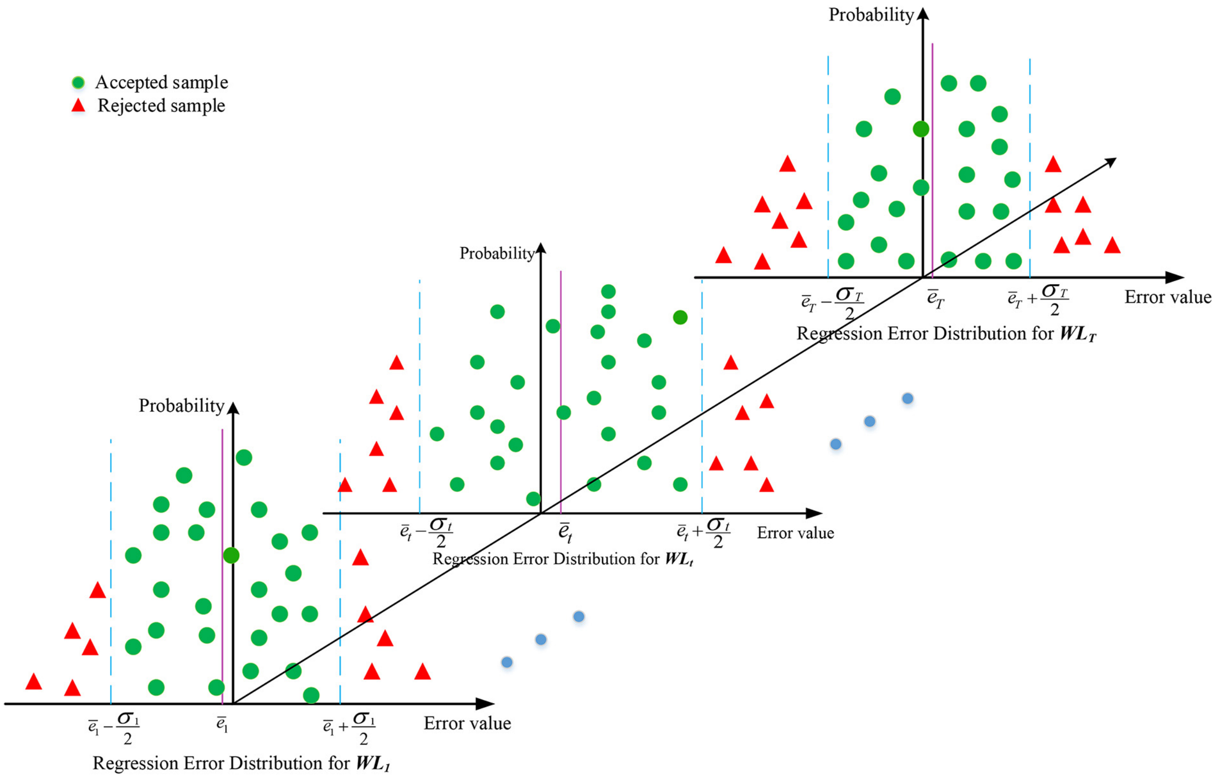
● Accepted sample
▲ Rejected sample
Probability
$\bar{e}_1 - \frac{\sigma_1}{2}$
$\bar{e}_1$
$\bar{e}_1 + \frac{\sigma_1}{2}$
Error value
Regression Error Distribution for WL_1
Probability
$\bar{e}_t - \frac{\sigma_t}{2}$
$\bar{e}_t$
$\bar{e}_t + \frac{\sigma_t}{2}$
Error value
Regression Error Distribution for WL_t
Probability
$\bar{e}_T - \frac{\sigma_T}{2}$
$\bar{e}_T$
$\bar{e}_T + \frac{\sigma_T}{2}$
Error value
Regression Error Distribution for WL_T

Fig. 2. Different regression error distributions for T corresponding weak learners under self-adjusted thresholds.

of approximately zero have high ARE and are determined as rejected samples, resulting in increasing in corresponding sample weights. Consequently, these near-zero values are defined as "hard samples" and are chosen more often for training weak learners in this aggregate method. Hence, when an output variable includes a zero-crossing in the output data, the performance of the AdaBoost.RT is degraded.

Moreover, in order to deal with the shortcoming of the original AdaBoost.RT algorithm, the modified AdaBoost.RT algorithm employed a self-modifiable threshold according to the trend of predication error in each iteration. However, the initial threshold is manually fixed as 0.2. Zhang and Yang [22] analyzed that such modified strategy does not ensure to search the entire threshold domain, and it is also sensitive to the initial threshold value. As a result, such method cannot guarantee to generate a best performed approximation to the target regression function. Furthermore, the AdaBoost.MRT algorithm implements a variance-scaled error rate instead of ARE of the original AdaBoost.RT as follows:

$$\varepsilon_t^r = \frac{\left|f_t(\mathbf{x}_i) - y_{r,i}\right|}{\sigma_t^r}. \tag{4}$$

The adopted variance-scaled error rate is able to allow for zero-crossings in the output data. Such strategy not only frequently classifies a part of predictions as rejected samples at each iteration, but also avoids fitting into outliers. Unfortunately, the threshold of the AdaBoost.MRT algorithm must be selected manually as well, which may cause the ensemble method to confront difficulty reaching a generally optimized learning effect.

In the proposed robust AdaBoost.RT method, the threshold does not require an initial value to be determined in advance manually. Instead, it derives the threshold value automatically via computing the statistics parameters of the approximation errors. Such initialization and subsequently self-adaptive updating mechanism of threshold are purely based on the data samples and weak learners. The performance of the proposed method is rarely influenced by human factors. Therefore, we define the complete self-adaptive threshold as the *robust threshold*.

In addition, at each iteration, if the predication error rate is larger than 0.5, the AdaBoost algorithm cannot ensure to converge and is prone to over-fit. Therefore, as a prerequisite to avoid over-fitting, the parameter β_t is introduced to limit the error rate ε_t to be no more than 0.5 in the proposed algorithm. The β_t is defined as

$$\beta_t = \frac{\varepsilon_t}{1 - \varepsilon_t}. \tag{5}$$

The detailed proof is described in Section III. The comparison of the threshold determination methods in different AdaBoost.RT algorithms is given in Table I.

The final output hypotheses of the AdaBoost.RT algorithm are defined as

$$f_{fin}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t \cdot f_t(\mathbf{x}) \tag{6}$$

where $f_t(\mathbf{x})$ is the tth weak learners output, α_t is the ensemble weight.

TABLE I
COMPARISON OF THE THRESHOLD DETERMINATION METHODS IN DIFFERENT ADABOOST.RT ALGORITHMS

Algorithms	***Initial value***	***Determination method of thershold***
Proposed Method	No need	Self-adjustable: Automatic selection of threshold value according to the statistic characteristics of the dataset.
AdaBoost.MRT	Must specify a constant threshold $\phi \in [0.05, 1.5]$	Fixed, cannot be changed, the error rate is singularity-free, variance-scaled.
Modified AdaBoost.RT	$\phi_0 = 0.2$	Self-adjustable: the value of ϕ increases with increasing error rate at each iteration, and vice versa.
AdaBoost.RT	Must specify a constant threshold $\phi \in [0, 0.4]$	Fixed, cannot be changed, the error rate includes a singularity.

B. ELM-Based Structural Optimization of AdaBoost.RT Model

Given the final output hypotheses and the target label $\{(f_{fin}(\mathbf{x}_i), y_i) | f_{fin}(\mathbf{x}_i) \times y_i \in R \times R, i = 1, \ldots, m\}$, where $f_{fin}(\mathbf{x}_i) = \sum_{t=1}^{T} \alpha_t \cdot f_t(\mathbf{x}_i)$. The mathematical model of ELM is represented by

$$\sum_{j=1}^{N} \beta_j g\big(f_{fin}(\mathbf{x}_i)\big) = \sum_{j=1}^{N} \beta_j g\big(a_j \cdot f_{fin}(\mathbf{x}_i) + b_j\big) = \hat{y}_i \tag{7}$$

where $f_{fin}(\mathbf{x}_i)$ is defined as an input case feature, a_j and b_j are random weights between the input layer and the hidden layer. β_j is the output weight connecting the hidden layer and the output layer, and $\hat{y}_i$ is the output prediction value.

According to the ELM theory [26], ELM is able to approximate m training data with zero error, i.e., $\sum_{i=1}^{m} \|\hat{y}_i - y_i\| = 0$, which implies that there exists a series of β_j such that

$$\sum_{j=1}^{N} \beta_j g\big(a_j \cdot f_{fin}(\mathbf{x}_i) + b_j\big) = y_i. \tag{8}$$

Eq. 8 is rewritten as

$$\boldsymbol{H\beta} = \boldsymbol{Y} \tag{9}$$

where $\boldsymbol{H}$ is defined as the hidden layer output matrix of ELM

$$\begin{aligned} &\boldsymbol{H}\big(a_1, \ldots, a_N, b_1, \ldots, b_N, f_{fin}(\mathbf{x}_1)\big), \ldots, f_{fin}(\mathbf{x}_m)\big) \\ &= \begin{bmatrix} g\big(a_1 \cdot f_{fin}(\mathbf{x}_1) + b_1\big) & \ldots & g\big(a_N \cdot f_{fin}(\mathbf{x}_1) + b_N\big) \\ \vdots & \ddots & \vdots \\ g\big(a_1 \cdot f_{fin}(\mathbf{x}_m) + b_1\big) & \ldots & g\big(a_N \cdot f_{fin}(\mathbf{x}_m) + b_N\big) \end{bmatrix}_{m \times N} \end{aligned} \tag{10}$$

$\boldsymbol{\beta}$ is given by

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_N \end{bmatrix}_{N \times 1} \tag{11}$$

and the output matrix $\mathbf{Y}$ is given

$$\boldsymbol{Y} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}_{m \times 1}. \tag{12}$$

Furthermore, from the perspective of standard optimization theory, ELM is able to be represented in objective function and constraints form

$$\begin{aligned} \text{Minimize: } & L_{\text{PELM}} = \frac{1}{2}\|\beta\|^2 + \frac{C}{2}\sum_{i=1}^{m} \xi_i^2 \\ \text{s.t.: } & \boldsymbol{h}\big(f_{fin}(\mathbf{x}_i)\big)\beta = y_i^T - \xi_i^T, i = 1, \ldots, m \end{aligned} \tag{13}$$

where ξ_i is the training error with respect to $f_{fin}(\mathbf{x}_i)$. The training of ELM can be switched into the following dual optimization problem based on the KKT theorem:

$$L_{DELM} = \frac{1}{2}\|\beta\|^2 + \frac{C}{2}\sum_{i=1}^{m} \xi_i^2 - \sum_{i=1}^{m} \alpha_i \big(\boldsymbol{h}\big(f_{fin}(\mathbf{x}_i)\big)\beta - t_i + \xi_i\big) \tag{14}$$

where α_i is the Lagrange multiplier, and C is a regularization coefficient.

Two solutions with different scales of training samples are obtained through solving the above optimization problem.

1) *Not-Large Training Scale Case:* The solution in this case

$$\boldsymbol{\beta} = \boldsymbol{H}^T\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{HH}^T\right)^{-1}\boldsymbol{Y}. \tag{15}$$

The final output function using ELM for structural optimization in the robust AdaBoost.RT is

$$\hat{\boldsymbol{Y}} = \boldsymbol{h}\big(f_{fin}(\boldsymbol{x})\big)\boldsymbol{\beta} = \boldsymbol{h}\big(f_{fin}(\boldsymbol{x})\big)\boldsymbol{H}^T\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{HH}^T\right)^{-1}\boldsymbol{Y}. \tag{16}$$

2) *Large Training Scale Case:* If the dimensionality of training samples is much smaller than the number of samples, i.e., $m >> N$, the solution in such case is

$$\boldsymbol{\beta} = \left(\frac{\boldsymbol{I}}{C} + \boldsymbol{HH}^T\right)^{-1}\boldsymbol{H}^T\boldsymbol{Y}. \tag{17}$$

The final output function using ELM for structural optimization in robust AdaBoost.RT is

$$\hat{\boldsymbol{Y}} = \boldsymbol{h}\big(f_{fin}(\boldsymbol{x})\big)\boldsymbol{\beta} = \boldsymbol{h}\big(f_{fin}(\boldsymbol{x})\big)\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{HH}^T\right)^{-1}\boldsymbol{H}^T\boldsymbol{Y}. \tag{18}$$

C. Algorithm of the Novel AdaBoost Framework With Robust Threshold and ELM-Based Structural Optimization

To sum up, the pseudocode of the robust AdaBoost.RT algorithm with structural optimization using ELM is shown in Algorithm 1 as follows.

Algorithm 1 Robust AdaBoost.RT Algorithm With Structural Optimization Using ELM

```
Input:
    • Given $m$ samples $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, m$. where $\mathbf{x}_i \times y_i \in R^l \times R$
    • Weak learner algorithm $WL_t$.
    • Distribution $D(i) = 1/m$ for all $i$.
    • $T$ number of iterations (machines).
    • $g(x)$ is the activation function.
    • Number of hidden node $N$.
    • Regularization parameter $C$.
Output:
    Final output prediction function $\hat{\boldsymbol{Y}}$
Algorithm:
    Initialize weight vector $w_i^t = D(i)$ for all $i$.
    Initialize prediction error rate $\varepsilon_1 = 0$.
    for $t = 1$ to $T$ do
        Set $\boldsymbol{p}^t = \frac{\boldsymbol{w}^t}{Z_t}$, where $Z_t$ is a normalization item for $\boldsymbol{p}^t$ to be a distribution.
        Call the $t$-th weak learner $WL_t$ providing it with distribution $WL_t$.
        Construct the regression model: $f_t(x) \longrightarrow y$.
        Compute prediction error rate:$\varepsilon_t = \sum_{i \in P} p_i^t$, where $P = \left\{i||e_t(i) - \bar{e}_t| > \frac{\sigma_t}{2}\right\}, i = 1, \ldots, m$.
        if $\varepsilon_t > 1/2$ then
            break
        end if
        Set $\beta_t = \varepsilon_t/(1 - \varepsilon_t)$
        Assign the weight of the $t$-th weak learner: $\alpha_t = -\log(\beta_t)$
        if $i \in P$ then
            $w_i^{t+1} = w_i^t$
        else
            $w_i^{t+1} = w_i^t \beta_t$
        end if
    end for
    Normalize $\alpha_1, \ldots, \alpha_T$, such that $\sum_{t=1}^{T} \alpha_t = 1$.
    The output final hypotheses: $f_{fin}(\mathbf{x}_i) = \sum_{t=1}^{T} \alpha_t \cdot f_t(\mathbf{x}_i)$
    Input the output final hypotheses $\{(f_{fin}(\mathbf{x}_i), y_i)|f_{fin}(\mathbf{x}_i) \times y_i \in R \times R\}$ for all $i$ into ELM.
    Randomly generate parameters $a_j$ and $b_j, j = 1, \ldots, N$.
    Calculate the hidden layer output matrix $\boldsymbol{H}$, as in Eq.10
    if not-large training scale case then
        Calculate $\boldsymbol{\beta}$ in Eq.15
        Calculate $\hat{\boldsymbol{Y}}$ in Eq.16
    else if large training scale case then
        Calculate $\boldsymbol{\beta}$ in Eq.17
        Calculate $\hat{\boldsymbol{Y}}$ in Eq.18
    end if
    return $\hat{\boldsymbol{Y}}$
```

III. General Bound on Robust AdaBoost.RT Empirical Error

Previous AdaBoost.RT algorithms use a constant or floating value as the regression threshold. However, these algorithms do not care about the characteristics of unseen data themselves. In this novel robust AdaBoost.RT approach, the statistics items of prediction errors are employed for the definition of robust threshold as criterion to select two optimal parallel decision hyper-planes that determine the generalization performance of the algorithm. In some cases, errors slightly beyond the region defined by the two optimal parallel decision hyper-planes are allowed. Based on the soft margin theory [32] introduced in the SVM method [29], we also introduce the soft margin as the measure for the maximum acceptable error in this approach. As a result, only a subset of the training examples is required to build the topology of the model, while other fractions of the training examples that lie beyond the soft margin are discarded. Therefore, the novel AdaBoost.RT algorithm can avoid over-fitting.

Two slack variables τ and τ_* are introduced to define the range of the soft margin, where the errors are tolerated. We prove a more general bound on the empirical error of the proposed algorithm, where a small positive value $\rho = \min(\tau, \tau_*)$ is defined. It is found that the empirical error $E_{ensemble}$ has an upper bound, so the proposed algorithm is convergent.

Theorem 1: The robust AdaBoost.RT generates a series of hypotheses with errors $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_T < 0.5$. The empirical error $E_{ensemble}$ is then bounded by

$$E_{ensemble} \leq 2^T \prod_{t=1}^{T} \sqrt{\varepsilon_t^{1-\rho}(1 - \varepsilon_t)^{1+\rho}}. \tag{19}$$

Proof: The regression problem $X \longrightarrow Y$ is converted into a binary classification case $\{X, Y\} \longrightarrow \{0, 1\}$. A mild assumption is made that the mean of errors is close to zero; i.e., $\bar{e} \approx E(e) = 0$. Thus, mean of errors can be neglected in such self-adaptive thresholds.

Let $I_t = \{i||f_t(\mathbf{x}_i) - y_i| < \sigma_t/2\}$, if $i \notin I_t, I_t^i = 1$; otherwise, $I_t^i = 0$.

The accuracy of each weak learner at each iteration is better than a random binary guess. We introduce a small slack positive value ρ to define the range of the soft margin, where the errors are tolerated. Therefore, the accuracy of each weak learner on any sample i has a lower bound as follows:

$$I_t^i \geq \frac{1}{2} + \frac{\rho}{2}. \tag{20}$$

According to (20), the final hypothesis f does not generates an error estimation on the ith sample data except

$$\prod_{t=1}^{T} \beta_t^{-I_t^i} \geq \left(\prod_{t=1}^{T} \beta_t\right)^{-\left(\frac{1}{2}+\frac{\rho}{2}\right)}. \tag{21}$$

The final weight on any ith sample is

$$w_i^{T+1} = D(i) \prod_{t=1}^{T} \beta_t^{(1-I_t^i)}. \tag{22}$$

Combining (21) and (22), the sum of rejected samples weights always bounds the sum of final ensemble weights

$$\begin{aligned}\sum_{i=1}^{m} w_i^{T+1} &\geq \sum_{i \notin I_{T+1}^{i}} w_i^{T+1} \\ &\geq \left(\sum_{i \notin I_{T+1}^{i}} D(i)\right)\left(\prod_{t=1}^{T} \beta_t\right)^{\frac{1+\rho}{2}} \\ &= E_{ensemble}\left(\prod_{t=1}^{T} \beta_t\right)^{\frac{1+\rho}{2}}. \end{aligned} \tag{23}$$

Using the convex inequality $x^d \leq 1 - (1 - x)d$, where $x \geq 0$ and $0 \leq d \leq 1$

$$\begin{aligned}\sum_{i=1}^{m} w_i^{t+1} &= \sum_{i=1}^{m} w_i^t \beta_t^{1-I_t^i} \leq \sum_{i=1}^{m} w_i^t \big(1 - (1-\beta_t)\big(1 - I_t^i\big)\big) \\ &= \left(\sum_{i=1}^{m} w_i^t\right)(1 - (1-\varepsilon_t)(1-\beta_t)). \end{aligned} \tag{24}$$

Uniting all $t = 1, \ldots, T$, we obtain that

$$\sum_{i=1}^{m} w_i^{T+1} \leq \prod_{t=1}^{T} (1 - (1-\varepsilon_t)(1-\beta_t)). \tag{25}$$

Combining (23) and (25), we obtain that

$$E_{ensemble} \leq \prod_{t=1}^{T} \frac{(1 - (1-\varepsilon_t)(1-\beta_t))}{\beta_t^{\frac{1+\rho}{2}}}. \tag{26}$$

Considering each factor in the multiplication is positive, minimization of every individual factor could be achieved via minimizing the right side of (26). Substituting the coefficient $\beta_t = \varepsilon_t / 1 - \varepsilon_t$ into (26) completes the proof. ■

In a single hypothesis, the empirical error is used as an unbiased estimator to estimate the generalization error. However, in a set of hypotheses, the empirical error will underestimate the generalization error because the empirical error is biased. Therefore, in the next section, we further analyze bounds on the generalization error of the proposed approach directly under PAC learning.

IV. Bounds on the Generalization Error With PAC Theory

In this section, the bounds on the generalization error of the proposed approach are proven directly through analyzing its learning capability. First, we introduce the PAC learning algorithm to prove the generalization error in finite base hypothesis spaces. Similar to the proof in finite base hypothesis spaces, we then prove a more general bound in infinite base hypothesis spaces with its VC-dimension.

Definition 1 [37]: Given a hypothesis space H, and D is a sequence of $m \geq 1$ independent random variables of any target concept c; and a small positive value ε as error tolerance, when $0 \leq \varepsilon \leq 1$, the version space of D (with respect to H) is ε-exhausted (with respect to c) if it does not include any hypothesis that has error more than ε with respect to c

$$\big(\forall h \in VS_{H,D}\big)\text{error}(f) < \varepsilon. \tag{27}$$

Lemma 1 [37] (ε-Exhausted the Version Space): If the hypothesis is finite, its cardinality is denoted by $|H|$, a target concept c, a set of samples D of c; for any $0 \leq \varepsilon \leq 1$, the probability that the version space of D (with respect to H) is not ε-exhausted (with respect to c) is then no more than $|H|e^{-\varepsilon m}$.

Lemma 2 (Hoeffding Bounds): If the hypothesis is finite $|H|$, D independent random variables of any target concept c, and for any $0 \leq \varepsilon \leq 1$, the empirical error $\text{err}(f)$ is measured

$$Pr\big[\text{error}_{true}(f) \geq \text{error}_{training}(f) + \varepsilon\big] \leq e^{-2m\varepsilon^2} \tag{28}$$

where $\text{error}_{true}(f)$ is the generalization error, and $\text{error}_{training}(f)$ is the empirical error.

Theorem 2: Let $\varepsilon = \sqrt{(1/2m)\ln((1/\delta))}$. Then, $\text{error}_{training}(f) \pm \varepsilon$ is the $1 - \delta$ confidence interval for $\text{error}_{true}(f)$.

Proof: This follows from the fact that:

$$\begin{aligned}&Pr\big[(\exists f \in H)\text{error}_{true}(f) \geq \text{error}_{training}(f) + \varepsilon\big] \\ &\quad \leq Pr\big[Max(\exists f \in H)\text{error}_{true}(f) \geq \text{error}_{training}(f) + \varepsilon\big] \\ &\quad \leq e^{-2m\varepsilon^2} = \delta. \end{aligned} \tag{29}$$

■

For any hypothesis in H, we must consider the hypothesis with a high probability to choose the optimal hypothesis. Therefore, the following theorem is obtained.

Theorem 3: If the hypothesis H is finite, which has $|H|$ elements, a target concept c, a sequence of samples D of c, and for any error tolerance ε $(0 \leq \varepsilon \leq 1)$

$$Pr\big[(\exists f \in H)\text{error}_{true}(f) \geq \text{error}_{training}(f) + \varepsilon\big] \leq |H|e^{-2m\varepsilon^2}. \tag{30}$$

Following Theorem 2, let $\varepsilon = \sqrt{1/2m \ln(\frac{|\mathcal{H}|}{\delta})}$, with of probability at least $1 - \delta$, then

$$\text{error}_{true}(f) \leq \text{error}_{training}(f) + \sqrt{\frac{\ln|H| + \ln(1/\delta)}{2m}}. \tag{31}$$

Theorem 3 defines the bounds of the generalization error in finite hypothesis spaces. However, it still suffers two shortcomings. On the one hand, when hypothesis H is large, the confidence interval for $\text{error}_{true}(f)$ is large. The more functions are in H, the more likely we have over-fitting, for which we compensate with a larger confidence interval. On the other hand, the derived Theorem 3 could not be applicable in infinite hypothesis spaces. Therefore, the *growth function* $\Pi_H(m)$ must be introduced into the inequality instead of $|H|$ for analysis in infinite hypothesis spaces.

Lemma 3 (Vapnik and Chervonenkis Theory): If the hypothesis is finite or infinite, a sequence of D independent random variables of any target concept c $(m \geq 1)$, and for any $0 \leq \varepsilon \leq 1$

$$Pr\big[(\exists f \in H)\text{error}_{true}(f) \geq \text{error}_{training}(f) + \varepsilon\big] \leq 8\Pi_H(m)e^{-m\varepsilon^2/32} \quad (32)$$

with of probability at least $1 - \delta$

$$\text{error}_{true}(f) \leq \text{error}_{training}(f) + \sqrt{\frac{32\big[\Pi_H(m) + \ln(8/\delta)\big]}{m}}. \quad (33)$$

A. Finite Base Hypothesis Spaces

All classifiers $f_t(\mathbf{x})$ are chosen from the base classifier space H, and C_T is the space of ensemble classifiers, which is generated by the proposed robust AdaBoost.RT algorithm running for T rounds. $f_{fin}(\mathbf{x})$ is the final hypotheses that compose T classifiers $f_t(\mathbf{x})$ as $f_{fin}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t f_t(\mathbf{x})$. Σ_T is defined as the space of all such linear threshold functions that represent $X \mapsto H_{fin}(x) \in R^T \times R^2$. Hence, $C_T = \{x \mapsto H_{fin}(x) : H_{fin} \in \Sigma_T; h_1, \ldots, h_T \in H\}$.

Lemma 4: The VC-dimension of the space Σ_T of linear threshold functions over R^T is equal to T.

Lemma 5 (Sauer's Lemma): If H is a hypothesis class of VC-dimension $(H) = d$, for $m \geq d \geq 1$, then they hold the bound $\Pi_H(m) \leq (em/d)^d$, where e is the base of the natural logarithm.

Lemma 6: Assume that H is finite. Let $m \geq T \geq 1$. For any set S of m points, the number of dichotomies realizable by C_T is bounded as follows:

$$\Pi_{C_T}(m) \leq \left(\frac{em}{T}\right)^T |H|^T. \quad (34)$$

Theorem 4: In finite base hypothesis spaces, the proposed robust AdaBoost.RT algorithm runs for T rounds on m independent random examples ($m \geq T$) using the base classifiers from a finite space H; then for any $0 \leq \varepsilon \leq 1$

$$Pr\big[(\exists f \in H)\text{error}_{true}(f) \geq \text{error}_{training}(f) + \varepsilon\big] \leq 8\left(\frac{em}{T}\right)^T |H|^T e^{-m\varepsilon^2/32}. \quad (35)$$

Following Theorem 2, let $\varepsilon = \sqrt{(32[T(\ln((em|H|/T))) + \ln(8/\delta)]/m)}$, with probability of at least $1 - \delta$:

$$\text{error}_{true}(f) \leq \text{error}_{training}(f) + \sqrt{\frac{32\Big[T\Big(\ln\Big(\frac{em|H|}{T}\Big)\Big) + \ln(8/\delta)\Big]}{m}}. \quad (36)$$

Proof: By plugging the bound from Lemma 6 into Lemma 3, the bound $\Pi_H(m)$ is represented as a general bound $((em/T))^T|H|^T$ on the generalization error of the proposed algorithm. ■

B. Infinite Base Hypothesis Spaces

In infinite base classifier spaces H with finite VC-dimension $d \geq 1$, $\Pi_{C_T}(m)$ in Lemma 6 has a new upper bound in infinite base classifier spaces. This leads to the following lemma.

Lemma 7: Assume that H has finite VC-dimension $d \geq 1$. Let $m \geq \max\{d, T\}$. For any set S of m points, the number of dichotomies realizable by C_T is bounded as follows:

$$\Pi_{C_T}(m) \leq \left(\frac{em}{T}\right)^T \left(\frac{em}{d}\right)^{dT}. \quad (37)$$

Theorem 5: Assume that H has finite VC-dimension $d \geq 1$ and robust AdaBoost.RT algorithm runs for T rounds on $m \geq \max\{d, T\}$ independent random variables of any target concept c; for any $0 \leq \varepsilon \leq 1$

$$Pr\big[(\exists f \in H)\text{error}_{true}(f) \geq \text{error}_{training}(f) + \varepsilon\big] \leq 8\left(\frac{em}{T}\right)^T \left(\frac{em}{d}\right)^{dT} e^{-m\varepsilon^2/32}. \quad (38)$$

Following Theorem 2, let $\varepsilon = \sqrt{(32[T(\ln((em/T)) + d\ln((em/d))) + \ln(8/\delta)]/m)}$, with probability of at least $1 - \delta$

$$\text{error}_{true}(f) \leq \text{error}_{training}(f) + \sqrt{\frac{32\big[T\big(\ln\big(\frac{em}{T}\big) + d\ln\big(\frac{em}{d}\big)\big) + \ln(8/\delta)\big]}{m}}. \quad (39)$$

Proof: When H has finite VC-dimension $d \geq 1$, by plugging the bound from Lemma 7 into Lemma 3, the bound $\Pi_H(m)$ is represented as a general bound $((em/T))^T((em/d))^{dT}$ on the generalization error of the proposed approach. ■

V. Performance Evaluation

A. Evaluation on UCI Benchmark Problems

In this section, we evaluate the proposed approach with other state-of-the-art algorithms on 15 real-world regression benchmarks, which include various categories from the UCI machine learning repository [38], as demonstrated in Table II. For instance, COIL-2000 is a dataset with a large size and medium dimensions. LVST is a dataset in a small size but large dimensions. *Friedman* #1, #2, and #3 are widely utilized problems for ensemble-based testing. Moreover, in our simulation, in order to demonstrate the effectiveness of the novel robust AdaBoost.RT algorithm based on the robust threshold and the structural optimization, a popular learning algorithm, the backpropagation algorithm [39], is selected as the weak learner of different ensemble methods [30], where the Levenberg–Marquardt algorithm and Log-sigmoid transfer functions are implemented in BP neural networks. The proposed approach is compared with several popular learning algorithms, including seven ensemble methods and three single algorithms, where ensemble methods contain robust AdaBoost.RT without structural optimization, original AdaBoost.RT algorithm [8], modified AdaBoost.RT algorithm [20], AdaBoost.MRT [21], AdaBoost.R2 [18], Bagging [2], and BEM [35], as well as single learning algorithms, include BP neural networks [39], SVR [40], and ELM [26]. All simulations are run using MATLAB in a Windows 7 environment with a 3.20 GHz CPU and 8GB of RAM.

In our simulations, we normalize input and output datasets into the ranges of $[-1, 1]$ and $[0, 1]$, respectively. In addition,

TABLE II
SPECIFICATION OF REAL-WORLD BENCHMARK DATASETS

Problems	Training data	Testing data	Features
Cloud	70	38	9
COIL-2000	6230	3592	85
Housing	340	166	13
Auto-MPG	320	72	7
Computer hardware	110	99	7
Bank	5300	2892	8
Census(house8L)	14000	8784	8
Breast cancer	114	80	32
Balloon	1300	1033	4
Abalone	3050	1127	7
Parkinson disease	3200	2675	21
LVST	80	45	308
Friedman #1	3000	2000	5
Friedman #2	3000	2000	4
Friedman #3	2000	1000	4

each dataset is split into training and testing dataset randomly as shown in Table II. We select 30% of training dataset as the validation dataset for model parameter selection.

For all BP-based ensemble methods, their suitable numbers of hidden nodes are chosen using grid search strategy to avoid over-fitting. The performance of algorithms is verified by RMSE and Dev in testing phases.

For all AdaBoost.RT algorithms, the number of weak learners T must be determined according to a *priori* knowledge, which is a common hyper-parameter in ensemble methods. In general, when the number of weak learners T reaches some degrees, the AdaBoost algorithm tends to over-fit [3]. *Occam's Razor theory says,* "*excessively complex models are affected by statistical noise, whereas simpler models may capture the underlying structure better and may thus achieve better predictive performance* [41]." Therefore, T must not be very large. The parameter T is set to be 5, 10, 15, 20, 25, and 30 in our simulations, where the optimal value is selected as the one that results in the best average RMSE for the COIL-2000 validation dataset.

In addition, we define $\sigma_t/2$ as the robust threshold. To verify that the coefficient $1/2$ is reasonable and reliable, in our simulation trials, the relative coefficient $\lambda \in (0, 1)$ is introduced to evaluate the robust AdaBoost.RT algorithm without structural optimization. We test at 0.1 and increase it at intervals of 0.1, which involve the coefficient $1/2$ in the proposed algorithm. Fig. 3 reveals the average RMSE with different values of T and λ for the COIL-2000 validation dataset.

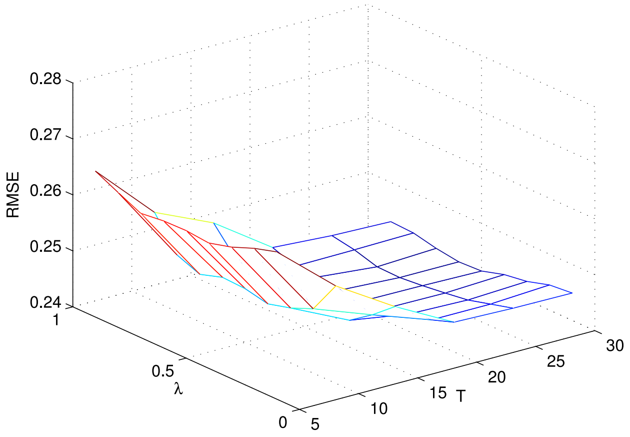

Fig. 3. Average RMSE with different combination T and λ in the proposed approach for COIL-2000 validation dataset.

As seen in Fig. 3, the robust AdaBoost.RT on the COIL-2000 dataset could reach excellent performance when the parameter T is larger than 15. For a given parameter T, the robust AdaBoost.RT is less sensitive to the coefficient λ. Hence, the relative coefficient $\lambda = 0.5$ is with good generalization performance for regression problems and need not be specifically determined by individual users. Even if the performance of the proposed method with the coefficient $\lambda = 0.5$ is poor on few special regression problems, the downstream structural optimization module is able to offer the set of weak learners a second opportunity to become a stronger regression model. Hence, we set the individual parameter $T = 25$ for the robust AdaBoost.RT in the following experiments. For the sake of fair comparison, the number of weak learners also set to be 25 for all ensemble methods.

Regarding to other parameters setting, for the modified AdaBoost.RT, the default initial value of ϕ_0 is set to be 0.2 [20]. For SVR, we utilize Gaussian kernel function and grid search strategy to select the pair of parameters (C, γ). Our simulations for SVR are carried out via libsvm MATLAB packages [42]. ELM employs the regularized version with a sigmoid activation function [26]. We also employ grid search strategy to choose the regularization parameter C from a wide range. Moreover, after training the robust AdaBoost.RT, we utilize ELM to further boost the performance of this proposed aggregate method. The experiment results are demonstrated in Table III.

Table III lists the average results of 50 trials of the proposed algorithm and other ensemble and single methods on 15 regression cases. Obviously, average testing RMSE values obtained by the proposed algorithm on 15 regression cases are always the best among these ensemble methods. The proposed robust AdaBoost.RT algorithm with structural optimization is extremely superior to the AdaBoost.RT algorithm without structural optimization that is purely based on the error statistics of each weak learner on input dataset. It demonstrates that the structural optimization strategy is able to boost the generalization performance of the robust AdaBoost.RT algorithm. Among other methods, AdaBoost.MRT supersedes original AdaBoost.RT on most datasets, especially Friedman #2 case, which includes the addition of noise and/or zero-crossing in the output data. The performance of the original AdaBoost.RT is vulnerable to the selection of the threshold value, which results in different thresholds for different problems. Therefore, the threshold plays a critical role for ensemble algorithms, and the threshold value determined via empirical selection in advance is not optimal. For most datasets, the original AdaBoost.RT algorithm outperforms other three ensemble methods, e.g., AdaBoost.R2, BEM, and Bagging.

TABLE III
COMPARISON OF THE TESTING RMSE OF THE PROPOSED METHOD WITH OTHER ENSEMBLE METHODS AND SINGLE LEARNING ALGORITHMS

Datasets	Proposed Method	Robust Ada.RT	Ada.MRT	Original Ada.RT	Modified Ada.RT	Ada.R2	BEM	Bagging	SVR	ELM	BPNN
Cloud	**0.2575**	0.2731	0.2736	0.2913	0.2859	0.2937	0.2917	0.2895	0.2981	0.2937	0.3048
COIL-2000	**0.2380**	0.2417	0.2617	0.2585	0.2569	0.2629	0.2672	0.2561	0.2638	0.2641	0.2653
Housing	**0.0896**	0.0973	0.0981	0.1089	0.1064	0.1082	0.1069	0.1073	0.1131	0.1046	0.1157
Auto-MPG	**0.0753**	0.0817	0.0808	0.0894	0.0852	0.0939	0.0914	0.0886	0.0903	0.0934	0.0956
Computer hardware	**0.0593**	0.0626	0.0606	0.0751	0.0704	0.0775	0.0762	0.0738	0.0829	0.0795	0.0819
Bank	**0.0557**	0.0581	0.0612	0.0607	0.0596	0.0652	0.0692	0.0629	0.0658	0.0676	0.0693
Census (house8L)	**0.0682**	0.0756	0.0718	0.0838	0.0797	0.0826	0.1062	0.0815	0.0854	0.0839	0.0863
Breast cancer	**0.2231**	0.2572	0.2429	0.2716	0.2653	0.2709	0.2731	0.2695	0.2761	0.2773	0.2792
Balloon	**0.0497**	0.0551	0.0593	0.0634	0.0596	0.0619	0.0648	0.0627	0.0648	0.0631	0.0676
Abalone	**0.0685**	0.0753	0.0712	0.0804	0.0784	0.0813	0.0795	0.0799	0.0872	0.0853	0.0886
Parkinson disease	**0.1936**	0.2387	0.2219	0.2527	0.2483	0.2526	0.2671	0.2481	0.2579	0.2607	0.2695
LVST	**0.2251**	0.2662	0.2596	0.2809	0.2756	0.2875	0.2869	0.2823	0.2865	0.2896	0.2973
Friedman #1	**0.2477**	0.2668	0.2653	0.2895	0.2827	0.3311	0.3245	0.2851	0.3468	0.3373	0.3743
Friedman #2	**8.1839**	9.576	9.027	10.547	9.863	12.352	12.682	11.237	12.0578	11.9645	12.5813
Friedman #3	**0.0916**	0.1091	0.1017	0.1028	0.1169	0.1338	0.1354	0.1091	0.1358	0.1310	0.1372

The modified AdaBoost.RT, in general, performs better than the original AdaBoost.RT. All ensemble methods are superior to the single learning algorithm when BP is chosen as the weak learner in ensemble methods. However, since the BP algorithm has relatively better performance than several simple learning algorithms, e.g., ANN (multilayer perceptron), linear SVM, and M5 model tree. As a result, the performance of a few other ensemble methods are slightly better or worse than that of BP. For instance, on the LVST dataset (large-dimensional training samples), the proposed method supersedes BP neural networks by as much as 24.3%, whereas AdaBoost.R2 outperforms BP neural networks by only 3.3%. In addition, on the Census dataset (large scale of data), the proposed method beats BP neural networks in 21.0% of the trials, whereas the BEM algorithm is worse than BP neural networks since BEM falls into over-fitting.

For the comparison on single learning algorithms, in general, the general performances of ELM and SVR are similar, both of which are slightly superior to BP. However, the proposed method is superior to these three single learning methods. It means that the generalization capability of BP is lower than those of both SVR and ELM in regression problems; however, the proposed ensemble framework can boost the generalization capability of BP through several iterations. Consequently, the proposed ensemble algorithm is capable to boost the performance of its component. On the LVST dataset, the proposed algorithm supersedes SVR and ELM by as much as 21.4% and 22.3%, respectively. On the Census dataset, similarly, the proposed

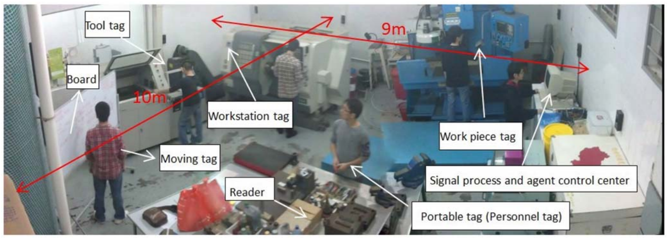
Tool tag
Board
10m
9m
Workstation tag
Work piece tag
Moving tag
Reader
Signal process and agent control center
Portable tag (Personnel tag)

Fig. 4. Test-bed shop floor environment in the IDIM laboratory.

algorithm outperforms SVR and ELM by 20.1% and 18.7%, respectively.

In summary, the experimental results reveal that the proposed method is always the best-performing learner among all candidate learning algorithms on 15 various real-world regression problems, especially the large-dimensional dataset and the larger scale of data. The AdaBoost.R2 algorithm is highly sensitive to outliers and noisy samples. The BEM method is prone to overfitting because of its sensitivity to the selection of BEM value, and it also has very high computation cost compared with other ensemble methods. Both the series of AdaBoost.RT and Bagging are always reliable and have good performances on different regression problems. In contrast, the series of AdaBoost.RT algorithms are more accurate than the Bagging approach. All in all, the proposed method surpasses other state-of-the-art approaches on all 15 real-world benchmarking regression problems covering different data scales and dimensional levels. The reason is that the proposed method benefits from the robust threshold and structural optimization.

B. Evaluation in Real-World Indoor Positioning Application

In this section, a real-world indoor positioning application is deployed to evaluate the proposed algorithm. Recently, machine-learning-based indoor positioning algorithms have been widely investigated [43]–[45] via analysis RFID signals. RFID facilitates next-generation manufacturing with online tracking of manufacturing objects [46]. Therefore, for the sake of fulfilling industrial requirements, it is urged to boost the accuracy and speed of indoor positioning algorithms.

This experimental case study is simulated in IDIM Laboratory of the University of Macau, where the shop-floor test bed (10 m × 9 m) for the experiment as demonstrated in Fig. 4. In this experiment, the shop-floor is set up with nine RFID readers, 20 reference tags, and nine tracking tags mounted on various manufacturing objects as depicted in Fig. 5. In order to increase the challenge of this experiment, a few stochastic disturbances, such as moving workers and work-in-processes in shop floor that simulate real industrial environments, are employed to test the feasibility of the proposed approach.

The indoor positioning belongs to a classical regression problem. Hence, it is reasonable to evaluate the proposed method in the indoor position case. The received signal strength (RSS) values of target and reference tags are acquired by the surrounding readers, and are transformed into coordinate parameters as input vectors for training and testing phase. In addition, noise in the RSS values collected via readers and tags is inevitable, and signal variation and ambient dynamics influence the RSS values as well. Hence, the indoor positioning application can be considered as uncertainties [45], where RSS values are unable to be observed precisely result from various errors in sampling, modeling, and/or measurement processes.

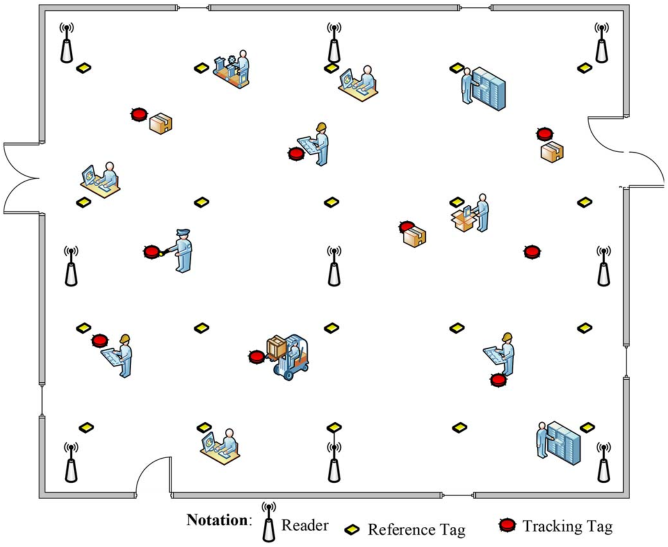

Fig. 5. Test bed shop floor layout for RFID indoor positioning system.

The proposed method is evaluated with state-of-the-art machine learning algorithms, including basic ELM and kernel ELM [26], OPT-ELM [47], SVR [40], LS-SVR [48], BP neural networks [39], as well as CTM-RELM, and SR-RELM [45]. The evaluation process is simulated using MATLAB in a Windows 7 environment with a 3.20 GHz CPU and 8GB of RAM.

We utilize ELM as weak learner in our proposed algorithm. It reveals that our proposed method is a general framework which is suitable for any weak learner rather than only a few specific machine learning methods. ELM is selected since it can fulfill the requirements of such real industrial problem in terms of a good balance of accuracy and efficiency. In the indoor positioning application, we set the number of ELMs in the proposed method to be 15 for sufficient ensemble accuracy. The regularization parameter C of ELM is chosen in the range of $\{2^{-15}, 2^{-14}, \dots, 2^{14}, 2^{15}\}$ using fivefold cross-validation. Since ELM and its variants are insensitive to the number of hidden nodes, and thus the number of hidden nodes in the hidden layer of ELM is fixed at 1000. In addition, other hyper-parameters of CTM-ELM and SR-ELM are selected using the same manner as in [45]. For SVR and LS-SVR, we utilize Gaussian kernel function and grid search strategy to select the pair of parameters. BP neural networks utilize the Levenberg–Marquardt algorithm and log-sigmoid transfer functions.

In the indoor positioning experiment, a total of 5400 RSS observations are collected as training samples, and additional

TABLE IV
COMPARISON OF ACCURACY AND TIME OF THE PROPOSED METHOD AND OTHER INDOOR POSITIONING LEARNING ALGORITHMS

Algorithms	RMSE (m)	Dev (m)	Training Time (s)	Testing Time (s)
Proposed Method	**2.64**	1.65	22.09	3.60
SR-RELM	2.71	**1.58**	17.64	0.49
CTM-RELM	2.83	1.72	113.82	0.56
OPT-ELM	3.48	2.36	2.71	0.43
Basic ELM	3.83	2.55	1.26	0.21
Kernel ELM	4.01	2.83	4.75	2.37
LS-SVR	3.62	2.21	8.39	3.94
SVR	3.98	2.67	138.83	17.35
BP	4.38	3.69	109.56	0.18

4050 RSS samples are utilized as testing samples. Two performance measures are employed: RMSE and Dev. Thirty trials for all compared indoor positioning algorithms have been conducted, and the average results of the 30 trials are listed in Table IV.

Table IV demonstrates the testing experimental results with respect to two performance indicators as well as training and testing time. In general, the proposed method, SR-RELM and CTM-RELM can achieve similar generalization performance, which means that these three algorithms could address the uncertainties of application discussed above. However, the training regression model mechanism of the proposed method is completely different from that of SR-RELM and CTM-RELM. The proposed method first employs multiple ELMs to generate an ensemble model, then further increase its strength using one single ELM network. While SR-RELM and CTM-RELM employ the small-residual constraint and the close-to-mean constraint, respectively, to boost the generalization ability of ELM in noise. The proposed method requires a slightly longer training time than SR-RELM yet has a far shorter training time than CTM-RELM.

VI. CONCLUSION

The selection of threshold is a critical factor influencing the generalization performance of all boosting algorithms. The existing manual prespecification of threshold may be ideal only for a very limited set of cases. In this paper, a novel robust AdaBoost.RT framework with structural optimization is first proposed for generally applicable weak learners on regression problems. The proposed method not only overcomes the shortcomings in the existing AdaBoost.RT algorithm and its variants, in which the threshold value is empirically specified, but also introduces an ELM network-based structure optimization method to further increase the strength of the ensemble model. The novel robust AdaBoost.RT algorithm dynamically computes optimal threshold, based purely on the statistical distribution of the approximation error of weak learners on given dataset. In addition, the resulted robust ensemble model is provided with a second optimization opportunity to own the universal approximation ability by using an ELM network. The ensemble structure is optimized to be further adapted to the source problem, which leverages the robustness of the proposed AdaBoost.RT framework.

Furthermore, to theoretically analyze the effectiveness of the proposed method, we prove the bounds of both empirical error and generalization error. It provides theoretical fundamentals of the AdaBoost.RT algorithm to reveal its advantage for various applications. First, a soft-margin upon the bound of empirical error is introduced to provide a more general condition for the proposed algorithm, which could avoid overfitting. In addition, considering that the empirical error is not unbiased for a set of hypotheses, the adoption of empirical error confronts the problem that it tends to underestimate the generalization error. Subsequently, the bounds of the generalization error of robust AdaBoost.RT are further proved by applying the PAC computational learning theory.

The simulation results on 15 benchmarks from the UCI dataset reveal that the proposed method is superior to other state-of-the-art ensemble methods and single learning algorithms in terms of stability and accuracy. In addition, the real-world indoor positioning experiment also demonstrates that the proposed framework is capable to reach higher positioning accuracy and faster speed.

ACKNOWLEDGMENT

The authors would like to thank Prof. C. L. P. Chen from the University of Macau for great inspiration and valuable discussions of this paper. The authors are grateful to the editor and reviewers for their valuable comments.

REFERENCES

[1] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 10, pp. 993–1001, Oct. 1990.

[2] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[3] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Proc. Eur. Conf. Comput. Learn. Theory*, Barcelona, Spain, 1995, pp. 23–37.

[4] L. Breiman "Arcing classifier (with discussion and a rejoinder by the author)," *Ann. Stat.*, vol. 26, no. 3, pp. 801–849, 1998.

[5] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[6] Z.-H. Zhou, J. Wu, and W. Tang, "Ensembling neural networks: Many could be better than all," *Artif. Intell.*, vol. 137, nos. 1–2, pp. 239–263, 2002.

[7] Y. Freund, R. Schapire, and N. Abe, "A short introduction to boosting," *J. Jpn. Soc. Artif. Intell.*, vol. 14, no. 5, pp. 771–780, 1999.

[8] D. L. Shrestha and D. P. Solomatine, "Experiments with AdaBoost.RT, an improved boosting scheme for regression," *Neural Comput.*, vol. 18, no. 7, pp. 1678–1710, 2006.

[9] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. ICML*, vol. 96. Bari, Italy, 1996, pp. 148–156.

[10] I. Mukherjee, C. Rudin, and R. E. Schapire, "The rate of convergence of AdaBoost," *J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 2315–2347, 2013.

[11] J. C. Duchi and Y. Singer, "Boosting with structural sparsity," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, Montreal, QC, Canada, 2009, pp. 297–304.

[12] V. Kuznetsov, M. Mohri, and U. Syed, "Multi-class deep boosting," in *Proc. Adv. Neural Inf. Process. Syst.*, Montreal, QC, Canada, 2014, pp. 2501–2509.

[13] C. Shen and H. Li, "On the dual formulation of boosting algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 12, pp. 2216–2231, Dec. 2010.

[14] S. Zhai, T. Xia, and S. Wang, "A multi-class boosting method with direct optimization," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, New York, NY, USA, 2014, pp. 273–282.

[15] A. Savran, B. Sankur, and M. T. Bilge, “Regression-based intensity estimation of facial action units,” *Image Vis. Comput.*, vol. 30, no. 10, pp. 774–784, 2012.

[16] N. Cai, F. Jin, Q. Pan, S.-Q. Xu, and F. Li, “Image restoration based on an AdaBoost algorithm,” in *Communications and Information Processing*. Heidelberg, Germany: Springer, 2012, pp. 294–301.

[17] D. Brochero, F. Anctil, and C. Gagné, “Forward greedy ANN input selection in a stacked framework with AdaBoost.RT—A streamflow forecasting case study exploiting radar rainfall estimates,” in *Proc. EGU Gen. Assembly Conf. Abstracts*, vol. 14. Vienna, Austria, 2012, p. 6683.

[18] H. Drucker, “Improving regressors using boosting techniques,” in *Proc. ICML*, vol. 97. Nashville, TN, USA, 1997, pp. 107–115.

[19] D. P. Solomatine and D. L. Shrestha, “AdaBoost.RT: A boosting algorithm for regression problems,” in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 2. Budapest, Hungary, 2004, pp. 1163–1168.

[20] H.-X. Tian and Z.-Z. Mao, “An ensemble ELM based on modified AdaBoost.RT algorithm for predicting the temperature of molten steel in ladle furnace,” *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 1, pp. 73–80, Jan. 2010.

[21] N. Kummer and H. Najjaran, “AdaBoost. MRT: Boosting regression for multivariate estimation,” *Artif. Intell. Res.*, vol. 3, no. 4, p. 64, 2014.

[22] P. Zhang and Z. Yang, “A robust AdaBoost.RT based ensemble extreme learning machine,” *Math. Problems Eng.*, vol. 2015, 2015, Art. no. 260970, doi: 10.1155/2015/260970.

[23] C. L. P. Chen, “A rapid supervised learning neural network for function interpolation and approximation,” *IEEE Trans. Neural Netw.*, vol. 7, no. 5, pp. 1220–1230, Sep. 1996.

[24] C. L. P. Chen and J. Z. Wan, “A rapid learning and dynamic stepwise updating algorithm for flat neural networks and the application to time-series prediction,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 1, pp. 62–72, Feb. 1999.

[25] G.-B. Huang, L. Chen, and C.-K. Siew, “Universal approximation using incremental constructive feedforward networks with random hidden nodes,” *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 879–892, Jul. 2006.

[26] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, “Extreme learning machine for regression and multiclass classification,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 513–529, Apr. 2012.

[27] Y. Yang and Q. M. J. Wu, “Extreme learning machine with subnetwork hidden nodes for regression and classification,” *IEEE Trans. Cybern.*, vol. PP, no. 99, pp. 1–14, Nov. 2015, doi: 10.1109/TCYB.2015.2492468.

[28] Y. Yang and Q. M. J. Wu, “Multilayer extreme learning machine with subnetwork nodes for representation learning,” *IEEE Trans. Cybern.*, vol. 46, no. 11, pp. 2570–2583, Nov. 2016.

[29] C. Cortes and V. Vapnik, “Support-vector networks,” *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[30] X. Qian, Y. Y. Tang, Z. Yan, and K. Hang, “ISAboost: A weak classifier inner structure adjusting based AdaBoost algorithm—ISAboost based application in scene categorization,” *Neurocomputing*, vol. 103, pp. 104–113, Mar. 2013.

[31] Y. Yang *et al.*, “Data partition learning with multiple extreme learning machines,” *IEEE Trans. Cybern.*, vol. 45, no. 8, pp. 1463–1475, Aug. 2015.

[32] K. P. Bennett and O. L. Mangasarian, “Robust linear programming discrimination of two linearly inseparable sets,” *Optim. Methods Softw.*, vol. 1, no. 1, pp. 23–34, 1992.

[33] D. Haussler, “Probably approximately correct learning,” Comput. Res. Lab., Univ. California, Santa Cruz, CA, USA, Tech. Rep. UCSC-CRL-90-16, 1990.

[34] V. N. Vapnik, *Statistical Learning Theory*. vol. 1. New York, NY, USA: Wiley, 1998.

[35] R. Avnimelech and N. Intrator, “Boosting regression estimators,” *Neural Comput.*, vol. 11, no. 2, pp. 499–520, 1999.

[36] R. Feely, “Predicting stock market volatility using neural networks,” B.A. dissertation, Dept. Comput. Sci., Trinity College Dublin, Dublin, Ireland, 2000.

[37] D. Haussler, “Quantifying the inductive bias in concept learning,” in *Proc. AAAI*, Philadelphia, PA, USA, 1986, pp. 485–489.

[38] M. Lichman, “UCI machine learning repository,” School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[39] S.-C. Ng, C.-C. Cheung, and S.-H. Leung, “Magnified gradient function with deterministic weight modification in adaptive learning,” *IEEE Trans. Neural Netw.*, vol. 15, no. 6, pp. 1411–1423, Nov. 2004.

[40] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, “Support vector regression machines,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 9. Denver, CO, USA, 1997, pp. 155–161.

[41] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth, “Occams razor,” *Readings in Machine Learning*. San Mateo, CA, USA: Morgan Kaufmann, 1990, pp. 201–204.

[42] C.-C. Chang and C.-J. Lin, “LibSVM: A library for support vector machines,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.

[43] Q. Yang, S. J. Pan, and V. W. Zheng, “Estimating location using Wi-Fi,” *IEEE Intell. Syst.*, vol. 23, no. 1, pp. 8–13, Jan./Feb. 2008.

[44] Z. Yang, P. Zhang, and L. Chen, “RFID-enabled indoor positioning method for a real-time manufacturing execution system using OS-ELM,” *Neurocomputing*, vol. 174, pp. 121–133, Jan. 2016.

[45] X. Lu, H. Zou, H. Zhou, L. Xie, and G.-B. Huang, “Robust extreme learning machine with its application to indoor positioning,” *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 194–205, Jan. 2016.

[46] X. Zhu, S. K. Mukhopadhyay, and H. Kurata, “A review of RFID technology and its managerial applications in different industries,” *J. Eng. Technol. Manag.*, vol. 29, no. 1, pp. 152–167, 2012.

[47] G.-B. Huang, X. Ding, and H. Zhou, “Optimization method based extreme learning machine for classification,” *Neurocomputing*, vol. 74, nos. 1–3, pp. 155–163, 2010.

[48] J. A. K. Suykens and J. Vandewalle, “Least squares support vector machine classifiers,” *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, 1999.

Peng-Bo Zhang received the B.Eng. degree from the Department of Mechanical Engineering and Automation, Taiyuan University of Science and Technology, Taiyuan, China, in 2012, and the M.Sc. degree with excellent oral defense from the Department of Electromechanical Engineering, University of Macau, Macau, China, in 2015. He is currently pursuing the Ph.D. degree with the Department of Industrial Engineering and Logistics Management, School of Engineering, Hong Kong University of Science and Technology, Hong Kong, China.

He has published several international journal and conference papers. His current research interests include machine learning and deep learning in theory and applications, computer vision, and intelligent systems.

Zhi-Xin Yang (M'14) received the B.Eng. degree in mechanical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 1992, and the Ph.D. degree in industrial engineering and engineering management from the Hong Kong University of Science and Technology, Hong Kong, China, in 2000.

He is currently an Assistant Professor with the Department of Electromechanical Engineering and an Assistant Dean with the Faculty of Science and Technology, University of Macau, Macau, China. His current research interests include innovative design, machine learning theory and applications, fault diagnosis, and intelligent manufacturing.