AIMS Mathematics, 9(9): 25705–25731.
DOI: 10.3934/math.20241256
Received: 02 July 2024
Revised: 18 August 2024
Accepted: 29 August 2024
Published: 04 September 2024

https://www.aimspress.com/journal/Math

Research article

Robust safe semi-supervised learning framework for high-dimensional data classification

Jun Ma* **and Xiaolong Zhu***

School of Mathematics and Information Sciences, North Minzu University, Yinchuan Ningxia 750021, China

* **Correspondence:** Email: jun_ma1990@num.edu.cn, 2020017@nmu.edu.cn.

Abstract: In this study, we introduced an innovative and robust semi-supervised learning strategy tailored for high-dimensional data categorization. This strategy encompasses several pivotal symmetry elements. To begin, we implemented a risk regularization factor to gauge the uncertainty and possible hazards linked to unlabeled samples within semi-supervised learning. Additionally, we defined a unique non-second-order statistical indicator, termed C_p-Loss, within the kernel domain. This C_p-Loss feature is characterized by symmetry and bounded non-negativity, efficiently minimizing the influence of noise points and anomalies on the model's efficacy. Furthermore, we developed a robust safe semi-supervised extreme learning machine (RS3ELM), grounded on this educational framework. We derived the generalization boundary of RS3ELM utilizing Rademacher complexity. The optimization of the output weight matrix in RS3ELM is executed via a fixed point iteration technique, with our theoretical exposition encompassing RS3ELM's convergence and computational complexity. Through empirical analysis on various benchmark datasets, we demonstrated RS3ELM's proficiency and compared it against multiple leading-edge semi-supervised learning models.

Keywords: semi-supervised learning; robustness; risk degree; correntropy; manifold regularization
Mathematics Subject Classification: 68T10, 91C20

1. Introduction

With the exponential growth of computing technology and increased access to diverse data sources, the availability of information has skyrocketed. To effectively extract the relevant data from this vast pool, machine learning has emerged as a crucial tool. It offers a solution to the challenging task of extracting the desired information amidst an abundance of data. In supervised learning, a multitude of data is utilized to train a model, which is then employed to make predictions on unlabeled data. However, if the model fails to generalize adequately and there is an excess of labeled data, it may lead

to overfitting. In numerous practical scenarios, there exists a considerable amount of unlabeled data alongside a subset of labeled data. However, labeling data can be a laborious and costly process. Utilizing the limited labeled data to enhance performance through learning from the vast pool of unlabeled data poses a significant challenge. Over the past decade, semi-supervised learning (SSL) has emerged as a highly effective learning framework, showcasing remarkable success in both theory and practical applications [1, 2]. Obtaining labeled samples is often a challenging and expensive task, while acquiring unlabeled samples tends to be easier and more cost-effective in many real-world problems. Semi-supervised learning (SSL) tackles this issue by leveraging different assumptions to establish connections between labeled and unlabeled samples. Among these assumptions, the manifold assumption [1] has gained significant traction and is widely adopted in practice. This assumption assumes that the data resides on a low-dimensional manifold, facilitating the exploration and utilization of unlabeled data for improved learning outcomes. For instance, Belkin et al. [1] introduced two algorithms, namely Laplacian regularized least squares (Lap-RLS) and support vector machines (Lap-SVM), which utilize manifold regularization to effectively harness information from unlabeled samples. These algorithms have demonstrated promising results. However, recent studies have highlighted a potential concern related to the reliance on unlabeled samples, suggesting that they could be compromised and potentially diminish the performance of SSL [3–5]. This raises limitations on the practical applicability of SSL to some extent [6, 7]. Consequently, there is a need to develop a safe semi-supervised learning (SaSSL) method that guarantees never performing worse than its supervised learning (SL) counterpart utilizing only labeled samples [8–12]. In recent years, numerous outstanding safe semi-supervised learning methods have been proposed [13–16].

Extreme learning machines (ELM) [17–19] have been extensively researched as single hidden layer feedforward networks (SLFNs). ELM has gained traction in the field of machine learning due to its simple architecture, low computational requirements, and wide applicability [20–22]. Additionally, ELM addresses drawbacks associated with conventional neural networks, including issues like local minima, imprecise learning, and slow convergence. Moreover, ELM offers a unified learning framework that caters to various applications such as regression, binary, and multiclass classification [23]. Recently, a semi-supervised ELM algorithm based on manifold regularization has been proposed to effectively leverage both labeled and unlabeled samples [24]. Although the results of ELM have been promising, it is worth mentioning that existing algorithms typically employ the mean square error (MSE) criterion as the cost function, while the impact of sample noise and outliers remain understudied. In the MSE-based criteria, equal penalty is assigned to all samples. However, samples with outliers or non-Gaussian noise tend to exhibit larger errors, leading to higher penalties for these samples. Consequently, the generalization performance suffers in the presence of non-Gaussian noise or outliers. This issue is particularly critical in semi-supervised learning, where misclassifications between labeled data easily affect nearby unlabeled data. Recently, the correlation coefficient has emerged as a novel criterion to address non-Gaussian noise and outliers [25–28]. Correntropy, a local similarity measure in kernel space, has been proposed to complement this approach. It offers an effective mechanism to mitigate the impact of noise and outliers by enhancing robustness through assigning lower weights to data outside the local neighborhood. In the areas of resilient learning and signal processing, such as adaptive filtering [26], state estimation [27], and principal component analysis [28], the correntropy-based criterion has shown promising results [29–32]. The correlation-based criterion has been used in a number of extreme learning machine (ELM)

algorithms, including [33–36]. In order to obtain robust learning performance when addressing outliers, Chen et al. [29] introduced the kernel mean p-power error (KMPE), a non-second-order statistical metric in kernel space. The regularized correntropy criterion (RCC) had been developed by Xing and Wang in order to help the ELM handle noise or outliers in the training data [30]. In parallel, Yang et al. [37] proposed a brand-new semi-supervised ELM technique based on the maximum correntropy criterion (MCC), known as RCC-based SSELM (RC-SSELM). According to experimental findings, RC-SSELM performs impressively in semi-supervised learning scenarios with non-Gaussian noise and outliers. Chen et al. established the Maximum Mixed Correlation Criterion (MMCC) for ELM to achieve robust generalization performance in [32] and presented the mixed correlation approach, which employs a combination of two or more kernel functions, to enhance the performance of ELM. A new semi-supervised ELM algorithm based on the MMCC optimization strategy was proposed by Yang et al. [35], which increases the algorithm's effectiveness and adaptability for managing huge and complicated outliers. For the purpose of demonstrating the efficacy of MC-SSELM, experimental results from several benchmark datasets were collected. The algorithm's superiority in terms of performance and adaptability was also shown through comparisons with a number of cutting-edge semi-supervised learning methods.

It can be observed that none of the existing semi-supervised ELM learning algorithms based on correntropy has focused on the uncertainty and potential risks of the unlabeled samples in the semi-supervised learning process. Inspired by the above research, in this paper, first of all, a robust and safe semi-supervised learning framework is proposed. Then, on the basis of this learning framework, a robust safe semi-supervised extreme machine learning (RS3ELM) framework is proposed. In more specific terms, the main contributions of this paper are as follows:

(1) Targeting uncertainty and potential risk of unlabeled samples in semi-supervised learning, a risk regularization term is constructed, which can better handle the uncertainty and potential risks, thereby improving the security and reliability of the learning algorithm.

(2) Based on the theory of correntropy, a new robust loss function is defined in the kernel space, which is called the p-power C-loss. The influence of noise points and outliers on the performance of the model can be effectively suppressed by this loss function.

(3) A robust and safe semi-supervised learning framework is established based on the risk regularization term and the p-power C-loss.

(4) A robust safe semi-supervised extreme learning machine (RS3ELM) for pattern classification is proposed based on this learning framework.

(5) The generalizing bound of RS3ELM is given by using Rademacher complexity. The fixed-point iteration method is used to solve the RS3ELM, and the computational complexity and the convergence of the algorithm are analyzed from a theoretical point of view.

(6) Experimental results show that RS3ELM performs significantly better than existing semi-supervised ELM learning algorithms on multiple benchmark datasets.

The remaining work is organized as follows. In Section 2, we review related work, including supervised learning, extreme learning machines (ELM), semi-supervised learning, and semi-supervised extreme learning (SS-ELM). In Section 3, we detail our algorithm. Experimental results and analysis are presented in Section 4. Finally, the main conclusions of this work are summarized in Section 5, and we also discuss future developments.

2. Related work

2.1. ELM

Let $\mathcal{T}_l = \{\boldsymbol{x}_i, y_i\}_{i=1}^{l}$ be the training set, where l is the number of training samples, $x_i \in \mathbb{R}^n$, $y_i \in \{-1, +1\}(i = 1, \ldots, l)$. Suppose that $f(\cdot)$ represents the decision function. Let $\mathcal{H}_K$ denote the reproducing kernel Hilbert space (RKHS) associated with a Mercer kernel K. Suppose that the general decision function f, the basic supervised learning framework, can be expressed as

$$\min_{f\in\mathcal{H}_K} \frac{1}{l}\sum_{i\in\mathcal{I}} \rho_{loss}(\boldsymbol{x_i}, y_i, f(\boldsymbol{x_i})) + \gamma_A\|f\|_{\mathcal{H}}^2, \tag{2.1}$$

where $\rho_{loss}(\cdot)$ is any loss function, such as the hinge loss function $\max\{0.1 - y_i f(x_i)\}$; $\|f\|_{\mathcal{H}}^2$ is the RKHS norm penalty and represents the complexity of functions in RKHS $\mathcal{H}_K$.

Under the assumption that L is the number of neurons in the hidden layer, the output function of ELM [17, 18] is

$$\boldsymbol{Y} = \boldsymbol{H\beta}, \tag{2.2}$$

where $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_L]^T$ is the vector of output weights between the hidden layer of L nodes and the output node, $\boldsymbol{Y} = [y_1, y_2, \ldots, y_l]^T$, and $\boldsymbol{H}$ is the output matrix of the hidden layer defined as

$$\boldsymbol{H} = \begin{bmatrix} h_1(x_1) & \cdots & h_L(x_1) \\ \vdots & \vdots & \vdots \\ h_1(x_l) & \cdots & h_L(x_l) \end{bmatrix},$$

where $h_i(x) = G(\mathbf{a_i}, b_i, \mathbf{x}) = \mathbf{a_i} \cdot \mathbf{x} + b_i, i = 1, \ldots, L$ ($\boldsymbol{a}$ and b can be randomly generated according to a continuous probability distribution). The primal regularization ELM framework can be expressed as

$$\min_{\beta} \Psi(\boldsymbol{\beta}) = C\|\boldsymbol{H\beta} - \boldsymbol{Y}\|^2 + \|\boldsymbol{\beta}\|^2, \tag{2.3}$$

where C is a penalty coefficient for the errors in the training process. Then, the output weight vector $\boldsymbol{\beta}$ can obtained by

$$\boldsymbol{\beta}^* = \begin{cases} (\boldsymbol{H}^T\boldsymbol{H} + \frac{\boldsymbol{I}_L}{C})^{-1}\boldsymbol{H}^T\boldsymbol{Y}, & l \geq L, \\ \boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{H}^T + \frac{\boldsymbol{I}_l}{C})^{-1}\boldsymbol{Y}, & l \leq L, \end{cases} \tag{2.4}$$

where $\boldsymbol{I}_L$ is an identity matrix of dimension L and $\boldsymbol{I}_l$ is an identity matrix of dimension l.

2.2. SS-ELM

Let $\mathcal{T} = \mathcal{T}_l \cup \mathcal{T}_u = \{x_i, y_i\}_{i=1}^{l} \cup \{x_i\}_{i=l+1}^{l+u}$ be a semi-supervised learning training dataset, where $x_i \in \mathbb{R}^n$, $y_i \in \{-1, +1\}$, $\mathcal{T}_l$ denotes labeled samples set with l, $\mathcal{T}_u$ denotes unlabeled samples set with u; $n = l + u$. Suppose that the general decision function f, the general semi-supervised learning framework can be expressed as the following optimization problem:

$$\min_{f\in\mathcal{H}_K} \frac{1}{l}\sum_{i\in\mathcal{I}} \rho_{loss}(\boldsymbol{x_i}, y_i, f(\boldsymbol{x_i})) + \gamma_A\|f\|_{\mathcal{H}}^2 + \gamma_I\|f\|_I^2, \tag{2.5}$$

where $\rho_{loss}(\cdot)$ is some loss function, $\|f\|^2_{\mathcal{H}}$ is the RKHS norm penalty and represents the complexity of functions in RKHS $\mathcal{H}_K$, γ_A and γ_I are the nonnegative regularization parameters, and $\|f\|^2_I$ is the manifold regularizer (MR) [1] whose empirical form takes

$$\|f\|^2_I = \frac{1}{(l+u)^2}\sum_{i,j=1}^{l+u} W_{ij}(f(x_i) - f(x_j))^2 = \frac{1}{(l+u)^2}(\boldsymbol{f}^T\boldsymbol{L}\boldsymbol{f}), \tag{2.6}$$

where $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ is the graph Laplacian; $\boldsymbol{D}$ is the diagonal degree matrix of $\boldsymbol{W}$ given by $D_{ii} = \sum_{j=1}^{l+u} W_{ij}$, $D_{ij} = 0$ for $i \neq j$; and the normalizing coefficient $\frac{1}{(l+u)^2}$ is the natural scale factor for the empirical estimate of the Laplace operator. The weight matrix W may be defined by k nearest neighbors or graph kernels as follows

$$W_{ij} = \begin{cases} exp(\frac{-\|x_i - x_j\|^2_2}{2\sigma^2}), & if\ x_i \in N_k(x_j)\ or\ \ x_j \in N_k(x_i), \\ 0, & otherwise, \end{cases} \tag{2.7}$$

where $N_k(x_j)$ denotes the data sets of k nearest neighbors of x_i.

Consequently, the primal problem of the semi-supervised extreme learning machine (SS-ELM) by introducing a manifold regularization term into (2.3), is

$$\min_{\boldsymbol{\beta}} \Psi(\boldsymbol{\beta}) = C\|\boldsymbol{H}_l\boldsymbol{\beta} - \boldsymbol{Y}\|^2 + \|\boldsymbol{\beta}\|^2 + \lambda Tr(\boldsymbol{\beta}^T\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}), \tag{2.8}$$

where C and λ are regularization parameters, $Tr(\cdot)$ denotes the trace of a matrix. Thus, we have

$$\boldsymbol{\beta}^* = \begin{cases} (\boldsymbol{I}_l + \boldsymbol{H}_l^T\boldsymbol{C}\boldsymbol{H}_l + \lambda\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n)^{-1}\boldsymbol{H}_l^T\boldsymbol{C}\boldsymbol{Y}, & n \geq L, \\ \boldsymbol{H}_l^T(\boldsymbol{I}_n + \boldsymbol{C}\boldsymbol{H}_l\boldsymbol{H}_l^T + \lambda\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{H}_n^T)^{-1}\boldsymbol{C}\boldsymbol{Y}, & n \leq L, \end{cases} \tag{2.9}$$

where $\boldsymbol{I}_l$ is an identity matrix of dimension l; and $\boldsymbol{C}$ is a diagonal matrix whose entries are $\boldsymbol{C}_{jj} = \frac{C}{l_{t_j}}$, where l_{t_j} is the number of training samples belonging to class j, $j = 1, 2 \ldots, l$; $\boldsymbol{I}_n$ is an identity matrix of dimension n.

2.3. Correntropy

In this section, the definition of correntropy is derived [25]. Some of its properties are briefly introduced.

Definition 1. *[25] Given two random variables X and Y, the correntropy is defined as:*

$$\begin{aligned} V(X,Y) &= E[\langle\Phi(X), \Phi(Y)\rangle_H] \\ &= \int \langle\Phi(x), \Phi(y)\rangle_{\mathcal{H}} dF_{XY}(x,y) \\ &= E[\kappa(X,Y)], \end{aligned} \tag{2.10}$$

where $E[\cdot]$ is the expectation operator, $F_{XY}(x,y)$ is the joint distribution function, and $\Phi(x) = \kappa(x,\cdot)$ is a nonlinear mapping induced by a Mercer kernel $\kappa(\cdot)$, which transforms x from the original space to a functional Hilbert space (or kernel space) $\mathcal{H}$ equipped with an inner product $\langle\cdot,\cdot\rangle\mathcal{H}$ which satisfies $\langle\Phi(x), \Phi(y)\rangle_{\mathcal{H}} = \kappa(x,y)$. It is obvious that we have $V(X,Y) = E[\kappa(X,Y)]$. Throughout this article, unless otherwise noted, the kernel function is a Gaussian kernel, given by

$$\kappa(x, y) = \kappa_\sigma(x - y) = \exp(-\frac{(x-y)^2}{2\sigma^2}) \tag{2.11}$$

with σ being the kernel bandwidth.

In practice, if the given sample is $\{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$, but the joint probability density function between the data is unknown, the correlation coefficient can be approximated as

$$V_{M,\sigma}(X, Y) = \frac{1}{m}\sum_{i=1}^{m}\kappa_\sigma(x_i - y_i). \tag{2.12}$$

Property 1. *[25] Correntropy is symmetric, expressed as $V_\sigma(X, Y) = V_\sigma(Y, X)$.*

Property 2. *[25] The value of correntropy is positive and bounded, and it is proven that the upper bound is reached if and only if $X = Y$.*

Property 3. *[25] Assume that the probability density function of the sample $\{(x_i, y_i), i = 1, ..., m\}$ is $f_{X,Y}(x, y)$. $E = Y - X$ is defined as the error random variable. $f_{E,\sigma}(e)$ is the Parzen estimate of the error probability density function from the data sample $\{(e_i = x_i - y_i), i = 1, ..., N\}$. Afterwards $V_{m,\sigma}(X, Y)$ is equal to the value of $f_{E,\sigma}(e)$ evaluated at point $e = 0$.*

$$V_{m,\sigma}(X, Y) = f_{E,\sigma}(0). \tag{2.13}$$

Definition 2. *[33, 34] Given two random variables X and Y, the **C-Loss** $C(X, Y)$ is defined by*

$$\begin{aligned} C(X, Y) &= \frac{1}{2}E[\|\Phi(X) - \Phi(Y)\|^2_{\mathcal{H}}] \\ &= \frac{1}{2}E[2\kappa_\sigma(0) - 2\kappa_\sigma(X - Y)] \\ &= E[1 - \kappa_\sigma(X - Y)]. \end{aligned} \tag{2.14}$$

It holds that $C(X, Y) = 1 - V(X, Y)$, so that the minimization of C-Loss is equivalent to the maximization of correlation. The MCC has recently attracted increasing attention due to its robustness to large outliers [33–36].

3. Main contributions

3.1. Motivation

In this section, we systematically describe our approach. Typically, semi-supervised learning leverages a large volume of unlabeled data alongside a sparse amount of labeled data. However, research indicates that the inclusion of unlabeled samples can sometimes degrade the performance of semi-supervised classifiers due to the inherent uncertainties and risks associated with unlabeled data. Moreover, in manifold regularization-based semi-supervised learning frameworks, often only the local manifold geometry of the samples is considered, while the global information is overlooked. Our core methodology involves constructing a risk degree regularization term to assess the uncertainty and potential risk of unlabeled data during the semi-supervised learning process. Intuitively, the risk associated with unlabeled samples is minimal when they contribute positively to semi-supervised

learning; conversely, the risk is high, warranting the use of a supervised classifier to predict their labels. Additionally, we introduce a novel non-second-order statistical indicator, referred to as C_p-Loss, within the kernel domain. The C_p-Loss metric is symmetrical and bounded by non-negativity, which effectively reduces the impact of outliers and noise on the model's performance. Furthermore, we propose a robust, safe, semi-supervised extreme learning machine (RS3ELM) based on this framework. We derive the generalization boundary of RS3ELM using Rademacher complexity, and the optimization of the output weight matrix in RS3ELM is performed through a analysis also covers the convergence and computational complexity of RS3ELM.

3.2. Risk estimation of unlabeled samples

In this section, we will provide a more detailed explanation of our algorithm. Our algorithm is based on two main methods: the SL method and the collaborative representation-based classification (CRC) method. Initially, we utilize the SL method and CRC method to reconstruct the unlabeled samples. This involves using the labeled samples to train a model that can reconstruct the features of the unlabeled samples. By doing this, we aim to capture the underlying structure and patterns in the unlabeled data. Furthermore, to assess the risk of the unlabeled samples, we compare the original and reconstructed versions of the unlabeled samples. By examining how well the reconstructed samples match the original ones, we obtain a measure of the risk associated with the unlabeled data. To incorporate these risk measures into our learning framework, we define risk-based regularization terms. These regularization terms serve as constraints that guide the learning process. By embedding these terms into the semi-supervised learning framework, we ensure that the resulting model strikes a balance between supervised and semi-supervised learning approaches. Therefore, the outputs of our learning framework represent a compromise between the information obtained from the available labeled samples and the reconstruction-based risk assessment of the unlabeled samples. This approach allows us to leverage the benefits of both supervised and semi-supervised learning, resulting in a more robust and effective learning algorithm.

To begin with, we start by training an ELM classifier using the labeled samples $\mathcal{X}_l$. Using this trained classifier, we can then make predictions y_u for the unlabeled samples x_u. These predictions serve as an estimate of the risk associated with each unlabeled sample. Next, we employ the collaborative representation-based classification (CRC) method to reconstruct the unlabeled samples. This involves utilizing the labeled samples $\mathcal{X}_l^{y_u}$ from the same class y_u to reconstruct the unlabeled samples. The objective function of CRC can be defined as follows:

$$\min_{\delta_j}(1 - \exp^{\frac{-\|x_j - \mathcal{X}_l^{y_j}\delta_j\|^2}{2\sigma^2}}) + \lambda\|\delta_j\|^2, \tag{3.1}$$

where λ is a regularization parameter.

Let $L_1(\delta_u) = \frac{\|x_u - \mathcal{X}_l^{y_u}\delta_u\|^2}{2\sigma^2} + \lambda\|\delta_u\|^2$ and $L_2(\delta_j) = \frac{\|x_u - \mathcal{X}_l^{y_j}\delta_j\|^2}{2\sigma^2} - 1 + \exp^{\frac{-\|x_j - \mathcal{X}_l^{y_j}\delta_j\|^2}{2\sigma^2}}$. Thus, the optimization problem (3.1) can be rewritten as

$$\min_{\delta_j} L_1(\delta_j) - L_2(\delta_j). \tag{3.2}$$

Clearly, the functions $L_1(\delta_j)$ and $L_2(\delta_j)$ are both convex. This means that the optimization problem given by Eq (3.2) is a DC (difference of convex) programming problem, where the differentiable convex

component is $L_2(\delta_j)$. By utilizing DC programming techniques, we can find the optimal value of δ_j denoted as δ_j^*. With this optimal value, we can reconstruct the unlabeled samples using the following expression:

$$\bar{x}_j = (\mathcal{X}_l^{y_j})^T \delta_j^*. \tag{3.3}$$

Moreover, we apply the ELM classifier to classify the reconstructed samples $\bar{x}_j$ and obtain the predicted label $\bar{y}_j$.

Definition 3. *Denote y_j and $\bar{y}_j$ as the predictions of x_j and $\bar{x}_j$, respectively. The degree of risk (RD) of the unlabeled samples can be defined as the following:*

$$r_j = \begin{cases} exp\{-\dfrac{\|x_j - \bar{x}_j\|^2}{\sigma}\}, & if\ y_j = \bar{y}_j, \\ exp\{\dfrac{\|x_j - \bar{x}_j\|^2}{\sigma}\}, & otherwise. \end{cases} \tag{3.4}$$

Based on Eq (3.4), when the predictions are identical ($\hat{y} = \bar{y}$), it implies that the unlabeled samples might be considered safe. In this case, as the error increases, r_j should be reduced to minimize any potential risks. On the other hand, if the predictions are not equal ($\hat{y} \neq \bar{y}$), it indicates that the unlabeled samples could be potentially risky. Therefore, in such instances, r_j should be elevated with increasing error to effectively address these risks.

Definition 4. *Let $g(x)$ and $f(x)$ be the outputs of the supervised classifier and the semi-supervised classifier, respectively. The decision function $\Xi_R(f)$ aims to find a trade-off between $g(x)$ and $f(x)$ by incorporating a risk-based term. It can be expressed as follows:*

$$\Xi_R(f) = \sum_{j=l+1}^{n} r_j \|f(x) - g(x)\|^2. \tag{3.5}$$

Here, r_j represents the safety degree of the unlabeled samples x_j.

Based on the above analysis, it is evident that $\Xi_R(f)$ plays a crucial role in determining the degree of disparity between supervised learning and semi-supervised learning.

3.3. C_p-Loss based on correntropy

Definition 5. *The C_p-Loss between two random variables X and Y can be defined as follows:*

$$\begin{aligned} C_p(X,Y) &= 2^{-p} E\left[\|\Phi(X) - \Phi(Y)\|_{\mathcal{H}}^{2p}\right] \\ &= 2^{-p} E\left[(\|\Phi(X) - \Phi(Y)\|_{\mathcal{H}}^2)^p\right] \\ &= 2^{-p} E\left[(2\kappa_\sigma(0) - 2\kappa_\sigma(X-Y))^p\right] \\ &= E\left[(1 - \kappa_\sigma(X-Y))^p\right]. \end{aligned} \tag{3.6}$$

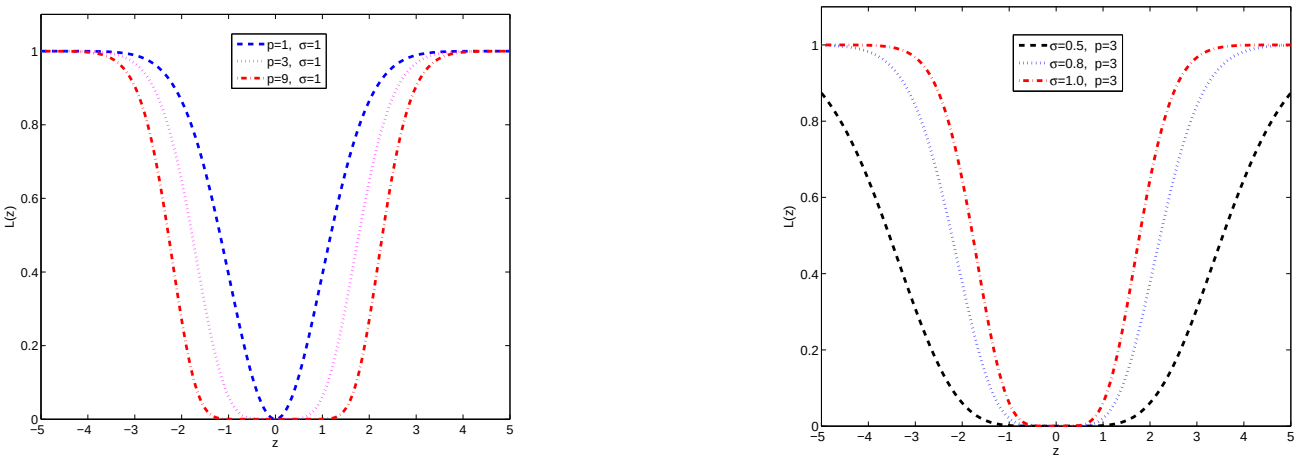
Here, $p > 0$ is the power parameter.

Remark 1. *If $p = 1$, then the C_p-Loss will degenerate into the C-Loss. In other words, C-Loss is a special case of C_p-Loss.*

Remark 2. *Given N samples $\{x_i, y_i\}$, the empirical C_p-Loss can be easily obtained as*

$$\hat{C}_p(X, Y) = \frac{1}{N}\sum_{i=1}^{N}(1 - \kappa_\sigma(x_i - y_i))^p. \tag{3.7}$$

The curves of our proposed loss function under different parameters are shown in Figure 1.



(A) $p = 1, 3, 9$, and $\sigma = 1$.

(B) $\sigma = 0.5, 0.8, 1.0$, and $p = 3$.

Figure 1. Loss function $L(z) = (1 - \kappa_\sigma(z))^p$ with different parameters.

Our $C_p(X, Y)$ loss has the following interesting properties:

Property 4. *Symmetry:* $C_p(X, Y) = C_p(Y, X)$.

Property 5. *Non-negative boundedness:* $0 \leq C_p(X, Y) < 1$. *The equal sign is true if and only if* $X = Y$.

Property 6. *When* $\sigma \to \infty$, *we have*

$$C_p(X, Y) \approx (2\sigma^2)^{-p} E[\|X - Y\|^{2p}]. \tag{3.8}$$

Property 7. *When* $p \to 0$, *we have*

$$C_p(X, Y) \approx 1 + pE[\log(1 - \kappa_\sigma(X - Y))]. \tag{3.9}$$

3.4. Robust safe semi-supervised learning framework

We can define the robust safe semi-supervised learning framework as follows:

$$\min_{f \in \mathcal{H}_k} \left\{ \hat{C}_p(\boldsymbol{x_i}, y_i, f(\boldsymbol{x_i})) + \gamma_A \|f\|^2_{\mathcal{H}} + \gamma_I \|f\|^2_{\mathcal{I}} + \gamma_R \Xi_R(f) \right\}. \tag{3.10}$$

In Eq (3.10), the regularization parameters $\gamma_A > 0$, $\gamma_I > 0$, and $\gamma_R > 0$ are used. The first three terms in Eq (3.10) are responsible for finding the semi-supervised classifier. The last term controls the trade-off between supervised and semi-supervised learning. The objective function in Eq (3.10) possesses the following characteristics:

(1) The empirical risk is defined in the first term of Eq (3.10), which evaluates how well the model fits the trained samples.

(2) The second term of Eq (3.10) represents the structural risk, which ensures the generalization capability of the model and prevents it from overfitting.

(3) The third term of Eq (3.10) introduces a joint regularization term. This term not only utilizes discriminative information more effectively but also explores the local geometric structure of new samples to enhance the classification performance. For this joint regularization, samples that are similar on the manifold should have the same class label if they belong to the same class, or different class labels if they do not.

(4) The fourth term of Eq (3.10) incorporates risk-based regularization. This term controls the trade-off between supervised and semi-supervised learning. The choice of risk degrees determines how the unlabeled samples are utilized.

Building upon our robust safe semi-supervised learning framework, we introduce a novel approach named robust safe semi-supervised extreme learning machine (RS3ELM), which is described as follows:

$$\min \hat{C}_p(\boldsymbol{Y}_l, \boldsymbol{H}_l\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_F^2 + \gamma Tr(\boldsymbol{\beta}^T \boldsymbol{H}_n^T \boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) + \alpha \sum_{j=l+1}^{l+u} r_j \|f(x_j) - g(x_j)\|^2. \tag{3.11}$$

In this framework, we incorporate three regularization parameters: λ, γ, and α. These parameters contribute to the definition of semi-supervised classifiers, while the last term is responsible for ensuring a suitable balance between ELM and SS-ELM.

$$\begin{aligned}
\min_{\boldsymbol{\beta}} \Gamma(\boldsymbol{\beta}) &= \hat{C}_p(\boldsymbol{H}_l\boldsymbol{\beta}, \boldsymbol{T}) + \lambda\|\boldsymbol{\beta}\|_F^2 + \gamma Tr(\boldsymbol{\beta}^T \boldsymbol{H}_n^T \boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) + \alpha \sum_{j=l+1}^{l+u} r_j \|f(x_j) - g(x_j)\|^2 \\
&= \frac{1}{l}\sum_{i=1}^{l}(1 - \kappa_\sigma(e_i))^p + \lambda\|\boldsymbol{\beta}\|_F^2 + \gamma Tr(\boldsymbol{\beta}^T \boldsymbol{H}_n^T \boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) \\
&\quad +\alpha(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM})^T \boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM}) \\
&= \frac{1}{l}\sum_{i=1}^{l}(1 - \exp(\frac{-(e_i^2)}{2\sigma^2}))^p + \lambda\|\boldsymbol{\beta}\|_F^2 + \gamma Tr(\boldsymbol{\beta}^T \boldsymbol{H}_n^T \boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) \\
&\quad +\alpha(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM})^T \boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM}).
\end{aligned} \tag{3.12}$$

In this scenario, we denote $\boldsymbol{\beta}_{ELM}$ as the optimal solution obtained from ELM, $\boldsymbol{H}_u$ as the hidden layer output matrix pertaining to unlabeled samples, and $\boldsymbol{R}$ as a diagonal matrix with the entry $\boldsymbol{R}_{jj} = r_{j+l}$. The derivative of Eq (3.12) with respect to $\boldsymbol{\beta}$ is

$$\begin{aligned}
\frac{\partial\Gamma}{\partial\boldsymbol{\beta}} &= 0 \\
&\Rightarrow \frac{1}{l}\sum_{i=1}^{l}[\frac{-p}{\sigma^2}(1 - \kappa_\sigma(e_i))^{p-1}\kappa_\sigma(e_i)e_i\boldsymbol{h}_i^T] + 2\lambda\boldsymbol{\beta} + 2\gamma(\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) \\
&\quad +2\alpha\boldsymbol{H}_u^T\boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM}) = 0 \\
&\Rightarrow \sum_{i=1}^{l}[-(1 - \kappa_\sigma(e_i))^{p-1}\kappa_\sigma(e_i)e_i\boldsymbol{h}_i^T] + \frac{2\sigma^2 l\lambda}{p}\boldsymbol{\beta} + \frac{2\sigma^2 l\gamma}{p}(\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) \\
&\quad +\frac{2\sigma^2 l\alpha}{p}\boldsymbol{H}_u^T\boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM}) = 0
\end{aligned}$$

$$\begin{aligned}
\Rightarrow \quad & \sum_{i=1}^{l}(\varphi(e_i)\boldsymbol{h}_i^T\boldsymbol{h}_i\boldsymbol{\beta} - \varphi(e_i)t_i\boldsymbol{h}_i^T) + \hat{\lambda}\boldsymbol{\beta} + \hat{\gamma}(\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) \\
& +\hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM}) = 0 \\
\Rightarrow \quad & \sum_{i=1}^{l}(\varphi(e_i)\boldsymbol{h}_i^T\boldsymbol{h}_i\boldsymbol{\beta}) + \hat{\lambda}\boldsymbol{\beta} + \hat{\gamma}(\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta}) \\
& = \sum_{i=1}^{l}(\varphi(e_i)t_i\boldsymbol{h}_i^T) + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}\boldsymbol{H}_u\boldsymbol{\beta}_{ELM}.
\end{aligned} \tag{3.13}$$

Thus, we can get

$$\boldsymbol{\beta} = [\boldsymbol{H}^T\boldsymbol{A}\boldsymbol{H} + \hat{\lambda}\boldsymbol{I} + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}\boldsymbol{H}_u]^{-1}(\boldsymbol{H}^T\boldsymbol{A}\boldsymbol{T} + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}\boldsymbol{H}_u\boldsymbol{\beta}_{ELM}), \tag{3.14}$$

where $\hat{\lambda} = \frac{2\sigma^2 l\lambda}{p}$, $\hat{\gamma} = \frac{2\sigma^2 l\gamma}{p}$, $\hat{\alpha} = \frac{2\sigma^2 l\alpha}{p}$, $\boldsymbol{h}_i$ is the i-th row of $\boldsymbol{H}$, and $\boldsymbol{A}$ is a diagonal matrix with diagonal elements $A_{ii} = \varphi(e_i) = (1 - \kappa_\sigma(e_i))^{p-1}\kappa_\sigma(e_i)$.

The obtained optimal solution $\boldsymbol{\beta}$ can be expressed as $\boldsymbol{\beta} = [\boldsymbol{H}^T\boldsymbol{A}\boldsymbol{H} + \hat{\lambda}\boldsymbol{I} + \hat{\gamma}\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}\boldsymbol{H}_u]^{-1}(\boldsymbol{H}^T\boldsymbol{A}\boldsymbol{T} + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}\boldsymbol{H}_u\boldsymbol{\beta}_{ELM})$. However, this equation does not have a closed-form solution as the right-hand side matrix depends on the weight vector $\boldsymbol{\beta}$ through the error term $e_i = t_i - \boldsymbol{h}_i\boldsymbol{\beta}$. Therefore, it is essentially a fixed-point equation, and the true optimal solution can be obtained using a fixed-point iterative algorithm.

Given a test set $\boldsymbol{X}_{new}$, we can calculate its corresponding hidden layer output matrix $\boldsymbol{H}_{new}$, and the prediction result can be obtained as follows:

$$\boldsymbol{Y} = \boldsymbol{H}_{new}\boldsymbol{\beta}^*. \tag{3.15}$$

Based on the above discussion, our algorithm will be presented in Algorithm 1.

Algorithm 1 Our algorithm

Input: l labeled examples $\{(x_i, y_i)\}_{i=1}^{l}$; u unlabeled examples $\{(x_i)\}_{i=1}^{u}$; regularization parameters γ_A, γ_D, and γ_R.

Output: The decision function $f^*(x) = \sum_{i=1}^{l+u}\boldsymbol{\beta}_i^* K(x_i, x)$ of RS3ELM.

1: Learn the ELM and predict the output y_u of the unlabeled instance x_u using ELM;
2: calculate $\bar{x}_u$ through (3.3);
3: compute risk degree of unlabeled samples using Eq (3.4);
4: construct data adjacency graph G with $(l+u)$ nodes using k nearest neighbors, and construct weight matrix W_{ij};
5: construct L using W;
6: initiate an ELM network of L hidden neurons with random input weights and biases, and calculate the output matrices of the hidden neurons $\boldsymbol{H}_l$, $\boldsymbol{H}_u$, and $\boldsymbol{H}_n$;
7: compute the output weights $\boldsymbol{\beta}^*$ using Eq (3.13).

3.5. Convergence and computational complexity analysis

3.5.1. Convergence analysis

Theorem 1. *[38] Based on the robust estimation given by Huber, if the loss function* $\Phi(Z)$ *is called a robust loss function, then it needs to satisfy the following:*

- $\Phi(Z) \geq 0$ *and* $\Phi(Z) = 0$*;*
- $\forall Z \in \mathbb{R}$, $\Phi(Z) = \Phi(-Z)$*;*
- $\forall Z \geq 0$, $\Phi'(Z) \geq 0$*;*
- $\Phi(Z)$ *is second-order differentiable in* $\mathbb{R}^+$*, and* $\Phi''(0^+) > 0$*;*
- $\Phi(\sqrt{Z})$ *is concave in* $\mathbb{R}^+$*;*

then there exists a convex function $\Psi(q)$ *such that*

$$\Phi(Z) = \inf_{q>0}\{\frac{1}{2}qZ^2 + \Psi(q)\}, \quad \forall Z. \tag{3.16}$$

When Z is fixed, a minimum solution q^* exists in the right-hand side of (3.16):

$$\inf_{q>0}\{\frac{1}{2}qZ^2 + \Psi(q)\} = \frac{1}{2}q^*Z^2 + \Psi(q^*), \tag{3.17}$$

where $q^* = \begin{cases} \dfrac{\Phi'(Z)}{Z} & Z > 0, \\ \Phi''(0^+) & Z = 0, \\ \dfrac{\Phi'(-Z)}{-Z} & Z < 0. \end{cases}$

Proposition 1. *For the function* $G(z) = \exp\left(\frac{-\|z\|^2}{2\sigma^2}\right)$*, it is possible to find a convex function* $\psi : \mathbb{R} \to \mathbb{R}$ *such that the following equation holds:*

$$G(z) = \sup_{\vartheta \in \mathbb{R}^-}\left(\vartheta\frac{\|z\|^2}{2\sigma^2} - \psi(\vartheta)\right). \tag{3.18}$$

In the above equation, $\mathbb{R}^-$ *represents the set of negative real numbers, and* $\sup(\cdot)$ *denotes the operation of finding the supremum. Additionally, for a fixed* z*, the supremum is attained at* $\vartheta = -G(z)$*.*

To simplify the expression, let us define the objective function of the proposed method as follows:

$$\begin{aligned} \Gamma(\boldsymbol{\beta}) \quad = \quad & \min_{\boldsymbol{\beta}} \frac{1}{l}\sum_{i=1}^{l}\left(1 - \exp(\frac{-(e_i^2)}{2\sigma^2})\right)^p + \lambda\|\boldsymbol{\beta}\|_F^2 + \gamma Tr(\boldsymbol{\beta}^T\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) \\ & +\alpha(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM})^T\boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM}). \end{aligned} \tag{3.19}$$

According to Proposition 1, we can rewrite (3.19) as

$$\hat{\Gamma}(\boldsymbol{\beta}, \vartheta) \quad = \quad \min_{\boldsymbol{\beta},\vartheta} \frac{1}{l}\sum_{i=1}^{l}\left(1 - \vartheta_i\frac{\|t_i^T - h_i\beta\|^2}{2\sigma^2} + \psi(\vartheta_i)\right)^p + \lambda\|\boldsymbol{\beta}\|_F^2 + \gamma Tr(\boldsymbol{\beta}^T\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta})$$

$$+\alpha(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM})^T\boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM}), \tag{3.20}$$

where $\vartheta_i = (\vartheta_1, \vartheta_2, \cdots, \vartheta_l)$.

Thus, given a fixed $\boldsymbol{\beta}$, the local optimal solution of (3.20) is

$$\vartheta_i = (1 - G(e_i))^{p-1}G(e_i), \tag{3.21}$$

where $e_i = t_i^T - h_i\boldsymbol{\beta}$, $i = 1, 2, \cdots, l$.

Then, we have $\Gamma(\boldsymbol{\beta}) = \min_{\vartheta}\Gamma(\boldsymbol{\beta}, \vartheta)$ and $\boldsymbol{\beta}$ can be derived as

$$\begin{aligned}\boldsymbol{\beta} &= \arg\min_{\boldsymbol{\beta}} \frac{1}{l}\sum_{i=1}^{l}\left(1 - \vartheta_i\frac{\|t_i^T - h_i\beta\|^2}{2\sigma^2}\right)^p + \lambda\|\boldsymbol{\beta}\|_F^2 + \gamma Tr(\boldsymbol{\beta}^T\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) \\ &\quad +\alpha(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM})^T\boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta} - \boldsymbol{H}_u\boldsymbol{\beta}_{ELM}) - Const.\end{aligned} \tag{3.22}$$

$Const$ is a constant real number greater than zero, $\sum_{i=1}^{l}(\varphi(e_i)\boldsymbol{h}_i^T\boldsymbol{h}_i\boldsymbol{\beta}) + \hat{\lambda}\boldsymbol{\beta} + \hat{\gamma}(\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n\boldsymbol{\beta}) + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}(\boldsymbol{H}_u\boldsymbol{\beta}) = \sum_{i=1}^{l}(\varphi(e_i)t_i\boldsymbol{h}_i^T) + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}\boldsymbol{H}_u\boldsymbol{\beta}_{ELM}$, where $\varphi(e_i) = (1-\kappa_\sigma(e_i))^{p-1}\kappa_\sigma(e_i)$. Therefore, the output weight vector β can be calculated by the following formula

$$\boldsymbol{\beta} = [\boldsymbol{H}^T\boldsymbol{A}\boldsymbol{H} + \hat{\lambda}\boldsymbol{I} + \hat{\gamma}\boldsymbol{H}_n^T\boldsymbol{L}\boldsymbol{H}_n + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}\boldsymbol{H}_u]^{-1}(\boldsymbol{H}^T\boldsymbol{A}\boldsymbol{T} + \hat{\alpha}\boldsymbol{H}_u^T\boldsymbol{R}\boldsymbol{H}_u\boldsymbol{\beta}_{ELM}),$$

where $\boldsymbol{A}$ is a diagonal matrix and diagonal elements for $A_{ii} = \varphi(e_i) = (1 - \kappa_\sigma(e_i))^{p-1}\kappa_\sigma(e_i)$.

Assume that at iteration k, there is

$$\vartheta_i^k = (1 - G(t_i^T - h_i\boldsymbol{\beta}^{k-1}))^{p-1}G(t_i^T - h_i\boldsymbol{\beta}^{k-1}), i = 1, 2, \cdots, l. \tag{3.23}$$

$$\beta^k = \arg\min_{\boldsymbol{\beta}}\hat{\Gamma}(\boldsymbol{\beta}, \vartheta^k). \tag{3.24}$$

Therefore, we can obtain

$$\Gamma(\beta^k) = \min_{\vartheta}\hat{\Gamma}(\boldsymbol{\beta}^k, \vartheta) = \min_{\vartheta}\hat{\Gamma}(\boldsymbol{\beta}^k, \vartheta^{k+1}). \tag{3.25}$$

For a fixed $\boldsymbol{\beta}^k$, $\hat{\Gamma}(\beta^k, \vartheta^{k+1})$ is the minimum value with respect to ϑ. Therefore,

$$\hat{\Gamma}(\boldsymbol{\beta}^k, \vartheta^{k+1}) \leq \hat{\Gamma}(\boldsymbol{\beta}^k, \vartheta^k), \tag{3.26}$$

$$\hat{\Gamma}(\boldsymbol{\beta}^k, \vartheta^{k+1}) \leq \hat{\Gamma}(\boldsymbol{\beta}^{k-1}, \vartheta^k), \tag{3.27}$$

$$\Gamma(\boldsymbol{\beta}^{k-1}) = \hat{\Gamma}(\boldsymbol{\beta}^{k-1}, \vartheta^k). \tag{3.28}$$

Therefore, we have

$$\Gamma(\boldsymbol{\beta}^k) \leq \Gamma(\boldsymbol{\beta}^{k-1}). \tag{3.29}$$

3.5.2. Computational complexity analysis

In the RS3ELM algorithm, the computational complexity primarily resides in the matrix update ($\boldsymbol{R}$), weight vector update (β), graph Laplacian construction, and the computation of the nearest neighbors (k). The main computational cost of each iteration is determined by the $O(L^3)$ computation required to update β.

Thus, the approximate computational complexity of the RS3ELM algorithm is given by $O(T \cdot (L^3 + (l+u)^2\log(l+u) + 2(l+u)^2)))$. Based on our experimental results, a satisfactory choice for the iteration count T is 10.

3.6. Generalization bound of RS3ELM

In this section, we provide theoretical bounds for the generalization error of the proposed method based on the Rademacher complexity. Let us define the empirical Rademacher complexity of function class $\mathcal{F}$, denoted by $\hat{E}_n(\mathcal{F})$, for a sample $\{x_1, \ldots, x_n\}$ generated by a distribution D, where $\mathcal{F}$ is a real-valued function class with domain $\mathcal{X}$.

$$\hat{E}_n(\mathcal{F}) = \mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\left|\frac{2}{n}\sum_{i=1}^{n}\sigma_i f(x_i)\right|\right]. \tag{3.30}$$

The expectation is taken over $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_n)^T$, where $\sigma_i \in \{-1, +1\}$ are independent uniform random variables. The Rademacher random variables satisfy $\boldsymbol{P}\{\sigma_i = -1\} = \boldsymbol{P}\{\sigma_i = +1\} = \frac{1}{2}$.

Based on the above, we can express the Rademacher complexity of $\mathcal{F}$, denoted by $E_n(\mathcal{F})$, as follows:

$$E_n(\mathcal{F}) = \mathbb{E}_{\boldsymbol{x}}[\hat{E}_n(\mathcal{F})] = \mathbb{E}_{\boldsymbol{x}\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\left|\frac{2}{n}\sum_{i=1}^{n}\sigma_i f(x_i)\right|\right]. \tag{3.31}$$

Here, the expectation is taken over the joint distribution of $\boldsymbol{x}$ and $\boldsymbol{\sigma}$.

Theorem 2. *[37] Let $\mathcal{F} : \mathcal{Z} = \mathcal{X} \times \mathcal{Y} \mapsto [-1, +1]$ be a class of functions. Let n samples be drawn independently from a distribution D. Then, with a probability of at least $1-\theta$, for every $f \in \mathcal{F}$, we have*

$$| Err(f) - \hat{Err}(f) | \leq \hat{E}_n(\mathcal{F}) + 3\sqrt{\frac{\ln(\frac{2}{\theta})}{2n}}, \tag{3.32}$$

where $Err(f)$ and $\hat{Err}(f)$ denote the expected error and the empirical error of f, respectively.

If we can compute the empirical Rademacher complexity $\hat{E}_n(\mathcal{F})$ for the function class $\mathcal{F}$, then this bound is applicable to a wide range of learning algorithms. Additionally, for kernelized algorithms, it is relatively simple to bound the empirical Rademacher complexity using the trace of the kernel matrix.

Theorem 3. *[37] For a kernel $\boldsymbol{K} : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ and a sample $\{x_1, \ldots, x_n\}$ from $\mathcal{X}$, the empirical Rademacher complexity of the class $\mathcal{F}(B)$ satisfies the following condition: If the norm $\|f\|_{\mathcal{H}} \leq B$, then*

$$\hat{E}_n(\mathcal{F}(B)) \leq \frac{2B}{n}\sqrt{\sum_{i=1}^{n}\boldsymbol{K}(x_i, x_i)}. \tag{3.33}$$

If for all $x \in \mathcal{X}$, we have $\boldsymbol{K}(x, x) \leq T^2$ and $\boldsymbol{K}$ is a standard kernel, the inequality above can be rewritten as

$$\hat{E}_n(\mathcal{F}(B)) \leq \frac{2B}{n}\sqrt{\sum_{i=1}^{n}\boldsymbol{K}(x_i, x_i)} \leq 2B\sqrt{\frac{T^2}{n}}. \tag{3.34}$$

Theorem 4. *(Generalization bound of RS3ELM) Suppose $Err(f)$ and $\hat{Err}(f)$ are the expected error and the empirical error of RS3ELM, l and u are the sets of labeled and unlabeled examples, respectively, and $n = l + u$. If the unlabeled sample is risk-free and $\boldsymbol{K}(x, x) \leq T^2$, then for every $\theta \in (0, 1)$, with probability at least $1 - \theta$, the generalization error of RS3ELM is given by*

$$|Err(f) - \hat{Err}(f)| \quad \leq \quad 2T\sqrt{\frac{n}{\lambda n^2 + \gamma\pi_1}\left[1 - \exp(-\frac{1}{2\sigma^2})\right]^p} + 3\sqrt{\frac{\ln(\frac{2}{\theta})}{2n}}. \tag{3.35}$$

Proof. To utilize Theorem 3, we need to determine the value of B in the inequality (3.33), which serves as an upper bound for $\|f\|^2_{\mathcal{H}}$. Assuming the unlabeled samples are safe, the objective function of RS3ELM in the regenerative Hilbert space can be reformulated as

$$\Phi(f) = \sum_{i=1}^{l} Loss(y_i, f(x_i)) + \frac{\lambda}{2}\|f\|^2_{\mathcal{H}} + \frac{\gamma}{2n^2}\boldsymbol{f}^T\boldsymbol{L}\boldsymbol{f}. \tag{3.36}$$

Assuming that $f^* = \arg\min_{f\in\mathcal{H}} \Phi(f)$ is the solution to (3.36), we have $\Phi(f^*) \leq \Phi(\mathbf{0})$. As a result, we can derive additional information

$$\frac{\lambda}{2}\|f^*\|^2_{\mathcal{H}} + \frac{\gamma}{2n^2}\boldsymbol{f}^T\boldsymbol{L}\boldsymbol{f} \leq \Phi(\mathbf{0}). \tag{3.37}$$

Let $\pi_1 < \pi_2 < \cdots < \pi_r$ be the non-zero eigenvalues of the Laplacian matrix $\boldsymbol{L}$, where r is the rank of $\boldsymbol{L}$. In terms of the eigenvalues, we have the inequality $\pi_1 < \pi_2 < \cdots < \pi_r$. This inequality represents the relationship between the minimum eigenvalue π_1 and the maximum eigenvalue π_r of the Laplacian matrix.

$$\pi_1\|f^*\|^2_{\mathcal{H}} \leq \boldsymbol{f}^T\boldsymbol{L}\boldsymbol{f} \leq \pi_r\|f^*\|^2_{\mathcal{H}}, \tag{3.38}$$

and

$$\begin{aligned}\left(\frac{\lambda}{2} + \frac{\gamma\pi_1}{2n^2}\right)\|f^*\|^2_{\mathcal{H}} &\leq \frac{\lambda}{2}\|f^*\|^2_{\mathcal{H}} + \frac{\gamma\pi_1}{2n^2}\boldsymbol{f}^T\boldsymbol{L}\boldsymbol{f} \\ &\leq \left(\frac{\lambda}{2} + \frac{\gamma\pi_r}{2n^2}\right)\|f^*\|^2_{\mathcal{H}}.\end{aligned} \tag{3.39}$$

By combining the inequalities (3.37) and (3.39), we can derive the following resulting inequalities

$$\left(\frac{\lambda}{2} + \frac{\gamma\pi_1}{2n^2}\right)\|f^*\|^2_{\mathcal{H}} \leq \Phi(\mathbf{0}). \tag{3.40}$$

Hence, to restrict the search range of f^*, we can confine it within a radius $R = \sqrt{\frac{\Phi(\mathbf{0})}{\left(\frac{\lambda}{2}+\frac{\gamma\pi_1}{2n^2}\right)}}$ inside the ball. Let $\mathcal{H}_R = \{f \in \mathcal{H} : \|f\|_{\mathcal{H}} \leq R\}$ denote the sphere with radius R in the regenerative Hilbert space $\mathcal{H}$, and the generalized version of RS3ELM can be expressed as:

$$|Err(f) - \hat{Err}(f)| \leq \hat{E}_n(\mathcal{H}_R) + 3\sqrt{\frac{\ln(\frac{2}{\theta})}{2n}}, \tag{3.41}$$

where

$$\hat{E}_n(\mathcal{H}_R) \leq 2R\sqrt{\frac{T^2}{n}}. \tag{3.42}$$

Suppose the output weight vector $\boldsymbol{\beta} = (0, 0, \ldots, 0)^T$ is chosen such that the last two terms in Eq (3.36) become zero. Then

$$\Phi(\mathbf{0}) = \sum_{i=1}^{l} Loss(y_i, f(\mathbf{0})) = \left[1 - \exp(-\frac{1}{2\sigma^2})\right]^p. \tag{3.43}$$

By substituting (3.43) into (3.40), we can get

$$\left(\frac{\lambda}{2}+\frac{\gamma\pi_1}{2n^2}\right)\|f^*\|_{\mathcal{H}}^2 \leq \left[1-\exp(-\frac{1}{2\sigma^2})\right]^p. \tag{3.44}$$

The inequality (3.44) can be equivalently written as

$$\|f^*\|_{\mathcal{H}}^2 \leq \frac{2n^2}{\lambda n^2+\gamma\pi_1}\left[1-\exp(-\frac{1}{2\sigma^2})\right]^p. \tag{3.45}$$

Obviously, the radius R is

$$R=\sqrt{\frac{2n^2}{\lambda n^2+\gamma\pi_1}\left[1-\exp(-\frac{1}{2\sigma^2})\right]^p}. \tag{3.46}$$

By substituting Eq (3.46) into Eq (3.42), we can derive the following inequality.

$$\hat{E}_n(\mathcal{H}_R) \leq 2T\sqrt{\frac{n}{\lambda n^2+\gamma\pi_1}\left[1-\exp(-\frac{1}{2\sigma^2})\right]^p}. \tag{3.47}$$

By substituting Eq (3.47) into Eq (3.41), we can easily obtain an approximation for the generalization error bound of RS3ELM:

$$|Err(f)-\hat{Err}(f)| \quad \leq \quad 2T\sqrt{\frac{n}{\lambda n^2+\gamma\pi_1}\left[1-\exp(-\frac{1}{2\sigma^2})\right]^p}+3\sqrt{\frac{\ln(\frac{2}{\theta})}{2n}}. \tag{3.48}$$

□

4. Experiments

4.1. Experimental setup

In this section, we conducted experiments on nine benchmark datasets to validate the effectiveness of the RS3ELM algorithm. We compared it with the following algorithms: SS-ELM [24], Lap-SVM [1], Lap-RLS [2], RCSSELM [36], and MC-SSELM [35]. For the experiments, we used the following parameters:

- SS-ELM: The number of hidden nodes L was chosen from the set $\{100, 500, 1000, 2000\}$. The regularization parameters C and λ were chosen from the set $\{10^{-5}, 10^{-3}, 10^{-1}, 10^{1}, 10^{3}, 10^{5}\}$.
- RC-SSELM: The regularization parameters C, λ, and the number of hidden nodes L were chosen within the same range as SS-ELM. The Gaussian kernel parameter σ was chosen from the set $\{1, 3, 5, 7, 9\}$.
- Lap-SVM and Lap-RLS: The Gaussian kernel parameter σ was chosen from the set $\{2^{-10}, 2^{-6}, 2^{-2}, 2^{2}, 2^{6}\}$. The regularization parameters γ_I and γ_A were chosen from the set $\{10^{-6}, 10^{-4}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$.
- MC-SSELM: The regularization parameters C, λ, and the number of hidden nodes L were chosen within the same range as SS-ELM. The Gaussian kernel parameters σ_1 and σ_2 were chosen from the set $\{2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, 2^{5}\}$. The maximum number of iterations K was set to 50, and the tolerable error ε was set to 10^{-3}. The variable center values were $\{-3, -1, 1, 3\}$, and the weight parameter α was chosen from the set $\{0.1, 0.2, 0.3, 0.4, 0.5\}$.

- RS3ELM: The maximum number of iterations K was set to 50, and the tolerable error ε was set to 10^{-3}. The Gaussian kernel parameter σ was chosen from the set $\{2^{-10}, 2^{-6}, 2^{-2}, 2^2, 2^6\}$. The regularization parameters γ, λ, and α were chosen from the set $\{10^{-6}, 10^{-4}, 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2\}$.

To ensure a fair comparison, we performed a grid search to find the optimal parameters for all algorithms, and the algorithm with the best performance was selected.

To assess the classification performance of all the algorithms, we used the traditional accuracy index (ACC). The ACC is defined as follows:

$$ACC = \frac{TP + TN}{TP + FN + TN + FP}. \tag{4.1}$$

In this equation, TP represents the number of true positives, TN represents the number of true negatives, FN represents the number of false negatives, and FP represents the number of false positives. A higher ACC value indicates a better model performance. Additionally, to compare the computational time of the algorithms, we recorded their running time, including both training and testing, on all the datasets used.

In the experiment, the main criterion for evaluating the performance of all the algorithms was the average classification accuracy, obtained through Monte Carlo cross-validation (MCCV) [39] and grid search. All the datasets used were approximately balanced, and the following steps were taken:

- Random division: The datasets were randomly divided into training sets and test sets, with a ratio of approximately 7:3.
- Repetition: This process of random division was repeated 10 times.
- Averaging: The results from the 10 repetitions were averaged to obtain a more reliable measure of performance.
- Normalization: To ensure fair comparison, all the datasets were normalized to the interval [0, 1]. Details of the datasets are presented in Table 1.
- Recording: The average classification accuracy, denoted by ACC, was obtained as the average of the 10 test results.

Table 1. Information description of 9 benchmark datasets.

ID	Datasets	Samples	Features	Class
1	Breast Cancer	569	30	2
2	wine	178	13	2
3	COIL20	1440	1024	20
4	Diabetic	1151	19	2
6	Carcinomc	174	9182	11
7	Heart diseasec	270	13	2
8	Lungc	203	3312	5
9	Proteinc	1483	56	10

By following these steps, we aimed to obtain objective experimental results that accurately assessed the performance of the algorithms. We implemented the algorithms used in the experiments using MATLAB scripts. To ensure a fair comparison, we employed the MATLAB toolbox for quadratic programming (QP) to solve all the quadratic programming problems embedded in the algorithms. The implementation was carried out on a PC with the following specifications: Intel(R) Core(TM) i7-8700 processor (3.20 GHz) and 16 GB of RAM. The operating system used was Windows 10, and MATLAB version 2014a was employed for the implementation purpose.

4.2. Experiments on synthetic dataset

In this section, we aim to demonstrate the robustness of our method by conducting experiments on a two-dimensional "XOR" dataset. The dataset is generated by perturbing points from two intersecting lines, Class 1: $y_i = 0.7x_i + \eta$ and Class 2: $y_i = -0.3x_i + \eta$. Each instance is randomly assigned Gaussian noise, following a normal distribution with mean 0 and standard deviation 0.2. Both classes consist of 1000 sample points. We evaluate our method on both the original "XOR" dataset and a contaminated version, which includes a few outliers. The experimental results are presented in Table 2.

The experimental results indicate that all methods perform exceptionally well in classifying the synthetic dataset without outliers. Among the algorithms, Lap-SVM exhibits the highest classification performance. However, the scenario changes when the dataset is contaminated with outliers. In this case, we observe that Lap-SVM, Lap-RSL, and SS-ELM demonstrate significantly poorer performance compared to the other algorithms. The classification accuracies of Lap-SVM, Lap-RSL, SS-ELM, RC-SSELM, MC-SSELM, and RS3ELM is presented in Table 2. It is evident from the experiments that our method, RS3ELM, outperforms the other five classifiers in terms of accuracy when classifying a synthetic dataset with outliers. However, in terms of training time, RS3ELM does not exhibit a notable advantage over the other classifiers. These results effectively highlight the robustness of RS3ELM. RS3ELM are more time-consuming: Despite having similar accuracy, RS3ELM requires more computational time compared to MC-SSELM. This could mean that the internal algorithms or processes involved in RS3ELM are more complex or require more iterations/operations for convergence.

Table 2. The accuracy (%) and learning time(s) of each classifier were evaluated on the "XOR" dataset, both with and without the presence of outliers.

	Lap-SVM	**Lap-RSL**	**SS-ELM**	**RC-SSELM**	**MC-SSELM**	**RS3ELM**
Datasets	ACC±std(%)	ACC±std(%)	ACC±std(%)	ACC±std(%)	ACC±std(%)	ACC±std(%)
	Time (s)	Time (s)	Time (s)	Time (s)	Time (s)	Time (s)
Without outliers	81.78±1.35	80.12±1.75	80.28±1.47	81.29±1.69	81.47±1.56	81.55±1.34
	0.891	0.836	0.465	0.557	0.847	1.062
With outliers	77.35±1.86	78.86±1.26	79.56±1.57	80.13±1.43	80.68±1.54	80.72±1.78
	0.891	0.836	0.465	0.557	0.847	1.062

4.3. Experiment without outliers

In this section, we evaluate the performance of our proposed method in comparison with other algorithms in a noise-free environment. Table 3 presents the experimental results for all algorithms using the optimal parameters. For Lap-SVM, Lap-RSL, SS-ELM, RC-SSELM, and MC-SSELM algorithms, we refer to the literature [35] for their experimental settings and results. Please consult the literature [35] for more detailed information. From Table 3, it is evident that our proposed method achieves comparable performance to the other five algorithms in the noise-free experiment. In general, the ELM-based methods demonstrate better performance than the SVM-based methods in most cases.

Table 3. Learning results of six algorithms in a noiseless environment.

	Lap-SVM	**Lap-RSL**	**SS-ELM**	**RC-SSELM**	**MC-SSELM**	**RS3ELM**
Datasets	ACC±S(%)	ACC±S(%)	ACC±S(%)	ACC± S(%)	ACC± S(%)	ACC± S(%)
1	90.09±1.83	91.08±1.62	91.15±1.19	91.97±1.22	90.41±0.43	90.53±0.67
2	94.87±3.39	95.73±1.96	98.72±0.00	98.72±0.00	100±0.00	97.87±1.33
3	96.15±0.98	94.43±0.20	96.17±0.56	96.31±0.34	96.62±0.40	97.05±0.94
4	70.72±1.61	70.53±1.49	72.35±1.02	72.58±0.60	73.45±0.54	73.21±1.12
5	96.97±0.52	95.15±1.05	96.91±0.50	96.73±1.52	97.64±0.73	97.89±0.84
6	68.60±9.65	65.22± 2.51	75.94 ±4.30	78.26 ± 1.02	78.33±2.34	79.17±1.04
7	81.18±3.28	82.55±1.22	82.47 ± 0.26	83.65 ±0.49	83.78 ±0.43	83.81±0.54
8	90.53±5.70	93.00±1.43	91.36±1.51	91.85±1.41	92.17±1.29	92.22 ±1.26
9	89.95 ±1.20	89.20±2.02	89.59 ±1.51	89.89 ± 0.70	90.23 ± 0.46	90.38± 0.53

4.4. Robustness analysis

To assess the efficacy of our proposed approach, we utilized nine benchmark datasets sourced from the UCI machine learning repository: http://archive.ics.uci.edu/ml/datasets.html. In order to ensure consistency, we standardized these datasets, thereby constraining the feature values to the range of [0, 1]. Subsequently, we conducted two types of experiments on these standardized datasets.

In this section, we initially conducted experiments on datasets containing 10% and 30% outliers to assess the robustness of the proposed method. These outliers were generated by randomly selecting 10% and 30% of the labeled training samples and applying the label inversion technique. (In robustness testing, label inversion is used to test if a model can withstand label noise. For example, flipping a certain percentage of labels randomly in the training data helps to determine how well the model can generalize despite noisy conditions.) The experimental settings and results of Lap-SVM, Lap-RSL, SS-ELM, RC-SSELM, and MC-SSELM algorithms were adopted from the literature [35]. The optimal parameters were used in these experiments, and the results are presented in Tables 4 and 5. Observing the tables, it is evident that the performance of all the algorithms declined as the label noise increased. Moreover, the model utilizing the entropy loss function outperformed other methods, indicating its effectiveness in handling outliers.

Table 4. The performance outcomes of six algorithms were documented on a dataset that contained 10% label noise.

	Lap-SVM	**Lap-RSL**	**SS-ELM**	**RC-SSELM**	**MC-SSELM**	**RS3ELM**
Datasets	ACC±S(%)	ACC±S(%)	ACC±S(%)	ACC± S(%)	ACC± S(%)	ACC± S(%)
1	88.48±0.98	87.86±0.57	89.59±0.53	89.89±1.47	90.41±0.36	90.26±0.43
2	95.30±3.23	93.16±4.12	97.44±1.28	97.69±1.07	99.74±0.51	96.88±1.49
3	93.45±0.20	91.72±1.08	92.07±0.40	93.72±0.26	95.21±0.33	95.56±0.41
4	67.25±1.61	69.76±0.67	69.04±0.43	69.04±0.56	72.00±0.68	72.47±0.64
5	96.67±0.52	95.76±0.52	96.36±1.57	97.09±0.41	96.36±0.57	96.08±1.12
6	67.15 ± 4.66	70.53 ± 3.65	75.94± 2.43	77.97 ± 4.51	78.65 ± 2.23	78.79 ± 2.28
5	96.67±0.52	95.76±0.52	96.36±1.57	97.09±0.41	96.36±0.57	96.08±1.12
6	96.67±0.52	95.76±0.52	96.36±1.57	97.09±0.41	96.36±0.57	96.08±1.12
7	80.78±2.78	81.96±0.90	79.88± 0.77	83.53 ±0.72	83.62 ±0.77	83.78 ±0.52
8	91.77±1.43	91.36±2.47	89.14±2.95	90.12±2.62	90.19±2.55	90.23±2.47
9	86.14±0.65	82.40±2.83	86.70±0.30	86.85±0.48	87.21±0.32	87.52±0.28

Table 5. The performance outcomes of six algorithms were documented on a dataset that contained 30% label noise.

	Lap-SVM	**Lap-RSL**	**SS-ELM**	**RC-SSELM**	**MC-SSELM**	**RS3ELM**
Datasets	ACC±S(%)	ACC±S(%)	ACC±S(%)	ACC± S(%)	ACC± S(%)	ACC± S(%)
1	85.25±3.90	86.74±3.72	89.37±0.86	89.52±1.27	89.59±1.21	89.61±1.17
2	88.89±3.92	90.60±1.48	82.05±4.53	93.33±0.57	95.64±0.63	94.72±0.69
3	85.06±2.65	85.06±2.49	86.28±0.50	86.41±0.61	87.86±0.30	89.73±1.09
4	60.39±1.17	62.22±3.83	64.41±0.63	65.28±0.80	66.96±0.76	66.79±0.52
5	93.94±3.19	94.55±0.91	95.27±1.49	96.36±0.91	97.27±0.57	95.06±1.17
5	96.67±0.52	95.76±0.52	96.36±1.57	97.09±0.41	96.36±0.57	96.08±1.12
6	54.11±1.67	64.25±0.84	68.99±4.18	71.01±5.42	71.54±3.45	71.89±3.32
7	78.43 ±3.78	79.61±1.89	75.53± 0.67	80.00± 0.59	80.41± 1.03	80.47± 0.74
8	88.07± 2.85	86.42±1.23	87.41±5.19	88.15±2.07	88.33±2.11	88.52±2.15
9	78.03± 3.19	81.46 ±1.60	81.46±0.35	83.07±0.71	83.17±0.85	83.03±0.67

In order to further validate the robustness of the proposed method, this study also conducted experiments in a characteristic noise environment. The characteristic noise dataset was generated by randomly selecting 30% of the characteristic values of each sample and assigning them a value of 0, thus creating a new experimental dataset. The experimental settings and results of Lap-SVM, Lap-RSL, SS-ELM, RC-SSELM, and MC-SSELM algorithms were obtained from the literature [35]. The experimental settings and parameter choices were consistent with the previous experiments. The results of this experiment are presented in Table 6 using the optimal parameters. From Table 6, it can

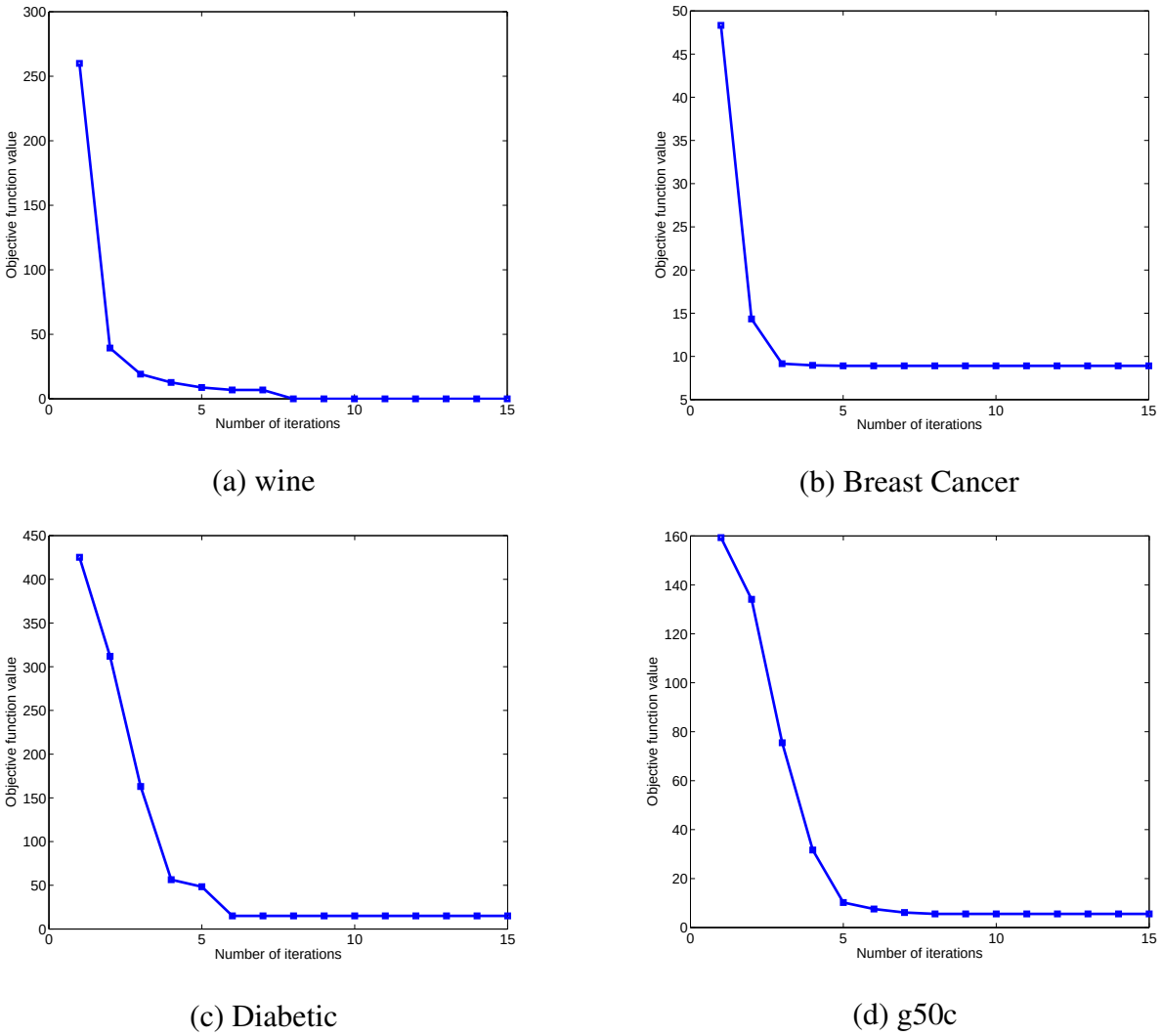
be observed that the proposed method performs comparably to other related algorithms. To summarize, the proposed method in this paper demonstrates effectiveness in terms of robustness and significantly enhances the performance of the model.

Table 6. Learning results of six algorithms on a dataset with characteristic noise.

	Lap-SVM	**Lap-RSL**	**SS-ELM**	**RC-SSELM**	**MC-SSELM**	**RS3ELM**
Datasets	ACC±S(%)	ACC±S(%)	ACC±S(%)	ACC± S(%)	ACC± S(%)	ACC± S(%)
1	85.75±6.11	87.86±0.86	87.66±2.30	88.40±1.45	89.52±1.04	90.11±1.24
2	81.37±2.23	82.35±2.56	80.35±1.22	82.59±1.42	81.76±0.37	82.23±1.19
3	97.01±0.74	94.44±4.12	97.18±1.07	97.95±1.15	98.21±0.63	98.33±1.14
4	59.71±1.53	59.23±0.89	59.36±0.38	61.62±0.76	62.26±0.22	62.05±1.06
5	95.45±1.82	94.55±1.57	96.00±1.04	96.00±0.50	96.36±1.29	96.55±1.26
6	30.92 ±7.15	34.30 ± 10.88	34.49 ± 5.27	40.58±3.40	39.94±3.76	40.71±2.66
7	81.37±2.23	82.35±2.56	80.35±1.22	82.59±1.42	82.74±1.28	82.79±1.24
8	79.01±0.00	78.60 ±0.71	79.01 ±0.00	79.01±0.00	79.01±0.00	79.01±0.00
9	79.21±1.17	80.71±0.19	80.26 ±1.35	81.27±0.88	81.41±0.27	81.45±0.31

4.5. Convergence analysis

To evaluate the convergence of the proposed algorithm, experiments are conducted on multiple datasets including wine, breast cancer, diabetic, and G50C datasets. The optimal parameters are selected, and the experimental procedure is consistent with the noise experiment. The learning outcomes are presented in Figure 2. As observed from Figure 2, it is evident that the objective function value of RS3ELM progressively decreases with each iteration, converging to a stable value in fewer than 10 iterations. Hence, the proposed algorithm RS3ELM demonstrates convergence.



(a) wine (b) Breast Cancer

(c) Diabetic (d) g50c

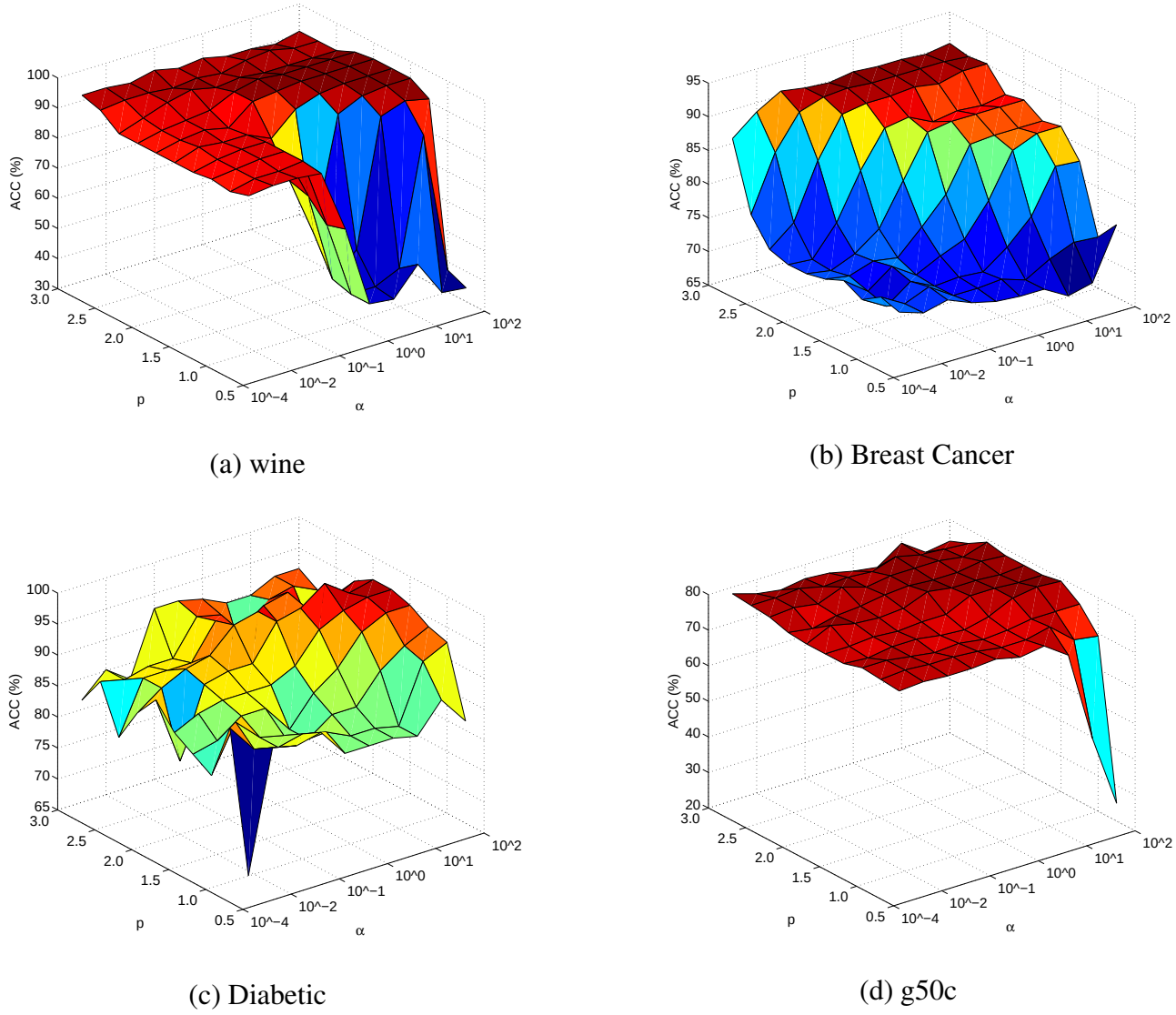
Figure 2. Convergence curves of algorithm RS3ELM in wine, Breast Cancer, Diabetic, and g50c datasets.

4.6. Sensitivity analysis of parameters

The RS3ELM algorithm incorporates several parameters, namely p and σ in the C_p-Loss function, as well as the tradeoff coefficients for three terms in (3.12), which are λ, γ, and α. For simplicity, we assume that γ, λ, and α are equal in value. To assist in parameter selection, we examine the impact of p, σ, λ, γ, and α on the performance of RS3ELM.

In our experimental analysis, we investigated the impact of p and α on the performance of RS3ELM, while keeping $\sigma = 2^{-2}$, $\lambda = 10^{-2}$, and $\gamma = 10^{-1}$. We conducted the experiment by varying α and p within the sets $\{10^{-4}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$ and $\{0.5, 1.0, 1.5, 2.0, 2.5, 3.0\}$, respectively. The accuracy results in relation to the parameters p and α are presented in Figure 3. Although the classification performance of the wine, Breast Cancer, Diabetic, and g50c datasets were minimally influenced by changes in p and α, we can still make certain conclusions about the optimal values of

these parameters based on Figure 3. Generally, outliers tend to affect the determination of hyperplanes. However, RS3ELM can effectively mitigate this influence by adjusting the parameter p, indicating that by controlling p, RS3ELM becomes less sensitive to outliers.



(a) wine

(b) Breast Cancer

(c) Diabetic

(d) g50c

Figure 3. Classification performance of RS3ELM under different parameters p and λ.

5. Conclusions

In this research paper, we propose a robust safe framework for semi-supervised learning that ensures the reliable utilization of unlabeled samples and achieves resistance to noise and outliers. To implement this framework, we introduce a robust safe semi-supervised extreme learning machine (RS3ELM), which is solved using a fixed-point iterative algorithm. We theoretically analyze the computational complexity and convergence of the RS3ELM and provide a generalizing error bound based on the

Rademacher complexity. Our experimental results on multiple datasets validate the robustness and classification accuracy of RS3ELM when compared to similar methods. Consequently, our proposed approach can be effectively applied to robust classification problems. It is important to note that our method focuses solely on binary classification tasks. Investigating the extension of this method to other learning tasks without compromising the model's classification accuracy and robustness is a valuable direction for future research.

Author contributions

Jun Ma and Xiaolong Zhu: Algorithm development, software creation, numerical example preparation, original draft writing, review and editing of the manuscript. All authors have read and approved the final version of the manuscript for publication.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported in part by the Fundamental Research Funds for the Central Universities of North Minzu University (No.2021JCYJ07 and No.2023ZRLG01), in part by the National Natural Science Foundation of China (No.62366001 and No.12361062), in part by the Key Research and Development Program of Ningxia (Introduction of Talents Project) (No. 2022BSB03046), and in part by the Natural Science Foundation of Ningxia Provincial (No. 2023AAC02053).

Conflict of interest

The authors declare that they have no conflict of interest and no relevant financial or non-financial interests to disclose.

References

1. M. Belkin, P. Niyogi, V. Sindhwani, Manifold regularization: A geometric framework for learning from labeled and unlabeled examples, *J. Mach. Learn. Res.*, **7** (2006), 2399–2434. http://dx.doi.org/10.5555/1248547.124863
2. O. Chapelle, B. Scholkopf, A. Zien, Semi-supervised learning, *IEEE T. Neural Networ.*, https://doi.org/10.1109/TNN.2009.2015974
3. T. Yang, C. E. Priebe, The effect of model misspecification on semi-supervised classification, *IEEE T. Pattern Anal.*, **33** (2011), 2093–2103. http://dx.doi.org/10.1109/TPAMI.2011.45
4. Y. F. Li, Z. H. Zhou, Towards making unlabeled data never hurt, *IEEE T. Pattern Anal.*, **37** (2015), 175–188. https://doi.org/10.1109/TPAMI.2014.2299812

5. Y. F. Li, Z. H. Zhou, *Improving semi-supervised support vector machines through unlabeled instances selection*, In AAAI Conference on Artificial Intelligence, **25** (2011), 386–391. https://doi.org/10.1609/aaai.v25i1.7920
6. Y. Wang, S. Chen, Z. H. Zhou, New semi-supervised classification method based on modified cluster assumption, *IEEE T. Neural Networ.*, **23** (2011), 689–702. https://doi.org/10.1609/aaai.v25i1.7920
7. Y. Wang, S. Chen, Safety-aware semi-supervised classification, *IEEE T. Neural Networ.*, **24** (2013), 1763–1772. https://doi.org/10.1109/TNNLS.2013.2263512
8. M. Kawakita, J. Takeuchi, Safe semi-supervised learning based on weighted likelihood, *Neural Networks*, **53** (2014), 146–164. https://doi.org/10.1016/j.neunet.2014.01.016
9. H. Gan, Z. Luo, M. Meng, Y. Ma, Q. She, A risk degree-based safe semi-supervised learning algorithm, *Int. J. Mach. Learn. Cyb.*, **7** (2015), 85–94. https://doi.org/10.1007/s13042-015-0416-8
10. H. Gan, Z. Luo, Y. Sun, X. Xi, N. Sang, R. Huang, Towards designing risk-based safe Laplacian regularized least squares, *Expert Syst. Appl.*, **45** (2016), 1–7. https://doi.org/10.1016/j.eswa.2015.09.017
11. H. Gan, Z. Li, Y. Fan, Z. Luo, Dual learning-based safe semi-supervised learning, *IEEE Access*, **6** (2017), 2615–2621. https://doi.org/10.1109/access.2017.2784406
12. H. Gan, Z. Li, W. Wu, Z. Luo, R. Huang, Safety-aware graph-based semi-supervised learning, *Expert Syst. Appl.*, **107** (2018), 243–254. https://doi.org/10.1016/j.eswa.2018.04.031
13. N. Sang, H. Gan, Y. Fan, W. Wu, Z. Yang, Adaptive safety degree-based safe semi-supervised learning, *Int. J. Mach. Learn. Cyb.*, **10** (2018), 1101–1108. https://doi.org/10.1007/s13042-018-0788-7
14. Y. Y. Wang, Y. Meng, Z. Fu, H. Xue, Towards safe semi-supervised classification: Adjusted cluster assumption via clustering, *Neural Process. Lett.*, **46** (2017), 1031–1042. https://doi.org/10.1007/s11063-017-9607-5
15. H. Gan, G. Li, S. Xia, T. Wang, A hybrid safe semi-supervised learning method, *Expert Syst. Appl.*, **149** (2020), 1–9. https://doi.org/10.1016/j.eswa.2020.113295
16. Y. T. Li, J. T. Kwok, Z. H. Zhou, *Towards safe semi-supervised learning for multivariate performance measures*, In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, **30** (2016), 1816–1822. https://doi.org/10.1609/aaai.v30i1.10282
17. G. B. Huang, Q. Y. Zhu, C. K. Siew, Extreme learning machine: Theory and applications, *Neurocomputing*, **70** (2006), 489–501. https://doi.org/10.1016/j.neucom.2005.12.126
18. Y. Cheng, D. Zhao, Y. Wang, G. Pei, Multi-label learning with kernel extreme learning machine autoencoder, *Knowl.-Based Syst.*, **178** (2019), 1–10. https://doi.org/10.1016/j.knosys.2019.04.002
19. X. Huang, Q. Lei, T. Xie, Y. Zhang, Z. Hu, Q. Zhou, Deep transfer convolutional neural network and extreme learning machine for lung nodule diagnosis on CT images, *Knowl.-Based Syst.*, **204** (2020), 106230. https://doi.org/10.1016/j.knosys.2020.106230

20. J. Ma, L. Yang, Y. Wen, Q. Sun, Twin minimax probability extreme learning machine for pattern recognition, *Knowl.-Based Syst.*, **187** (2020), 104806. https://doi.org/10.1016/j.knosys.2019.06.014

21. C. Yuan, L. Yang, Robust twin extreme learning machines with correntropy-based metric, *Knowl.-Based Syst.*, **214** (2021), 106707. https://doi.org/10.1016/j.knosys.2020.106707

22. Y. Li, Y. Wang, Z. Chen, R. Zou, Bayesian robust multi-extreme learning machine, *Knowl.-Based Syst.*, **210** (2020), 106468. https://doi.org/10.1016/j.knosys.2020.106468

23. H. Pei, K. Wang, Q. Lin, P. Zhong, Robust semi-supervised extreme learning machine, *Knowl.-Based Syst.*, **159** (2018), 203–220. https://doi.org/10.1016/j.knosys.2018.06.029

24. G. Huang, S. Song, J. N. D. Gupta, C. Wu,Semi-supervised and unsupervised extreme learning machines, *IEEE T. Cybernetics*, **44** (2014), 2405. https://doi.org/10.1109/tcyb.2014.2307349

25. W. Liu, P. P. Pokharel, J. C. Principe, Correntropy: Properties and applications in non-Gaussian signal processing, *IEEE T. Signal Proces.*, **55** (2007), 5286–5298. https://doi.org/10.1109/tsp.2007.896065

26. N. Masuyama, C. K. Loo, F. Dawood, Kernel Bayesian ART and ARTMAP, *Neural Networks*, **98** (2018), 76–86. https://doi.org/10.1016/j.neunet.2017.11.003

27. X. Liu, B. Chen, H. Zhao, J. Qin, J. Cao, Maximum correntropy Kalman filter with state constraints, *IEEE Access*, **5** (2017), 25846–25853. https://doi.org/10.1109/access.2017.2769965

28. B. Chen, X. Liu, H. Zhao, J. C. Principe, Maximum correntropy Kalman filter, *Automatica*, **76** (2017), 70–77. https://doi.org/10.1016/j.automatica.2016.10.004

29. B. Chen, X. Lei, W. Xin, Q. Jing, N. Zheng, Robust learning with kernel mean p-power error loss, *IEEE T. Cybernetics*, **48** (2018), 2101–2113. https://doi.org/10.1109/tcyb.2017.2727278

30. H. Xing, X. Wang, Training extreme learning machine via regularized correntropy criterion, *Neural Comput. Appl.*, **23** (2013), 1977–1986. https://doi.org/10.1007/s00521-012-1184-y

31. Z. Yuan, X. Wang, J. Cao, H. Zhao, B. Chen, Robust matching pursuit extreme learning machines, *Sci. Programming*, **1** (2018), 1–10. https://doi.org/10.1155/2018/4563040

32. B. Chen, X. Wang, N. Lu, S. Wang, J. Cao, J. Qin, Mixture correntropy for robust learning, *Pattern Recogn.*, **79** (2018), 318–327. https://doi.org/10.1016/j.patcog.2018.02.010

33. G. Xu, B. G. Hu, J. C. Principe, Robust C-loss kernel classifiers, *IEEE T. Neur. Net. Lear.*, **29** (2018), 510–522. https://doi.org/10.1109/tnnls.2016.2637351

34. A. Singh, R. Pokharel, J. Principe, The C-loss function for pattern classification, *Pattern Recogn.*, **47** (2014), 441–453. https://doi.org/10.1016/j.patcog.2013.07.017

35. J. Yang, J. Cao, A. Xue, Robust maximum mixture correntropy criterion-based semi-supervised ELM with variable center, *IEEE T. Circuits-II*, **67** (2020), 3572–3576. https://doi.org/10.1109/tcsii.2020.2995419

36. J. Yang, J. Cao, T. Wang, A. Xue, B. Chen, Regularized correntropy criterion based semi-supervised ELM, *Neural Networks*, **122** (2020), 117–129. https://doi.org/10.1016/j.neunet.2019.09.030

37. P. L. Bartlett, S. Mendelson, *Rademacher and Gaussian complexities: Risk bounds and structural results*, In: Conference on Computational Learning Theory & European Conference on Computational Learning Theory, Berlin/Heidelberg: Springer, 2001, 224–240. https://doi.org/10.1007/3-540-44581-1-15

38. P. J. Huber, Robust estimation of a location parameter, *Ann. Math. Stat.*, **35** (1964), 73–101. https://doi.org/10.1214/aoms/1177703732

39. Q. S. Xu, Y. Z. Liang, Monte Carlo cross validation, *Chemometr. Intell. Lab.*, **56** (2001), 1–11. https://doi.org/10.1016/s0169-7439(00)00122-2

AIMS Press