AIMS Mathematics, 11(4): 9228–9259.
DOI:10.3934/math.2026381
Received: 16 January 2026
Revised: 08 March 2026
Accepted: 18 March 2026
Published: 03 April 2026

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

Research article

Robust twin extreme learning machine with adaptive fractional loss

Xiang Jin, Guolin Yu* and Jun Ma

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

* **Correspondence:** Email: yuguolin@nmu.edu.cn.

Abstract: The twin extreme learning machine (TELM), based on the hinge-loss function, demonstrates significant potential for pattern classification tasks. However, the hinge-loss function, which minimizes the shortest distance between sets, results in classifiers that are sensitive to noise and unstable in the presence of overfitting. To enhance TELM's performance, a novel learning framework, termed adaptive fractional loss TELM (AFTELM), is proposed. This framework incorporates an adaptive fractional loss (AF-loss) function, offering improved robustness to noise compared to TELM with hinge loss. A theoretical analysis is provided to examine the noise insensitivity of AFTELM. The concave-convex procedure (CCCP) is employed for efficient optimization. Extensive experiments on benchmark datasets validate the superior performance of AFTELM, demonstrating its robustness to noise and enhanced classification ability.

Keywords: twin extreme learning machine; adaptive fractional loss; robustness; noise insensitivity
Mathematics Subject Classification: 68T10, 91C20

1. Introduction

Classification problems are frequently encountered in machine learning and data mining applications. Pattern recognition requires effective methodologies to discern the differences between different types of data classes. Conventional statistical approaches are restricted by rigorous presumptions concerning data distribution and dimensional limitations. The curse of dimensionality presents a significant challenge in high-dimensional feature spaces. Computational efficiency and generalization capability have become critical requirements for practical classification systems.

Support vector machines (SVMs) [1–4] were developed as a powerful solution to address these classification challenges. The principle of structural risk minimization is implemented in SVMs to achieve optimal generalization performance. The theoretical foundation for SVM optimization is provided by the Vapnik–Chervonenkis theory. The maximum margin criterion is used to construct

distinct hyperplanes among various classes. Optimal decision boundaries are detected through the solution of convex quadratic programming problems. The kernel trick is utilized to transform linearly inseparable problems into high-dimensional feature spaces where linear separation becomes feasible.

Twin support vector machines (TWSVMs) were proposed by Jayadeva et al. [2] to overcome these computational limitations. Two non-parallel hyperplanes are established in TWSVMs instead of a single optimal separating hyperplane. The original optimization problem is decomposed into two smaller quadratic programming problems (QPPs). Each QPP is focused on one class, while the other class is treated as a constraint. Computational complexity is reduced from $O(n^3)$ to $O(n^2)$, where n represents the number of training samples. Training speed is improved by approximately four times compared to standard SVMs. Memory usage is significantly decreased. However, regularization terms are absent in classical TWSVM formulations, potentially leading to overfitting and sensitivity to outliers.

Various improvements have been developed in recent years to address the robustness limitations of TWSVMs. Bounded loss functions are incorporated in robust twin support vector machines (RTSVMs) [5] to reduce outlier influence. Smooth approximations to the ε-insensitive loss with enhanced robustness are provided by the integration of Huber loss functions into twin SVR formulations [6]. Asymmetric noise distributions and imbalanced datasets are effectively handled by generalized pinball loss functions [7, 8]. Sparsity is maintained while quantile-based robustness is provided by the ε-insensitive zone pinball loss [9]. Penalty parameters are dynamically adjusted based on data characteristics in the generalized adaptive Huber loss framework [10]. Significant performance improvements on contaminated datasets are demonstrated by these robust formulations.

Twin bounded support vector machines (TBSVMs) were developed by Shao et al. [11, 12] based on TWSVMs to enhance robustness through structural risk minimization. An L_2-norm regularization term is incorporated in TBSVMs to prevent overfitting. The trade-off between training accuracy and model complexity is controlled by the regularization parameter. Improved generalization performance on noisy datasets is achieved. Enhanced stability is demonstrated when dealing with outliers and mislabeled samples. Recent TBSVM developments include advanced regularization techniques [13], where individual feature influence is bounded through flexible penalty mechanisms. Superior performance in extreme class imbalance scenarios is demonstrated by robust variants for imbalanced data. Theoretical guarantees for generalization bounds and computational stability are ensured through the integration of structural risk minimization principles.

Extreme learning machines (ELMs) [14, 15] were developed as an alternative approach to traditional neural network training. Random weights are assigned to input-hidden layer connections without iterative optimization. Output weights are determined through least squares methods. Fast training speeds are achieved by eliminating backpropagation. However, challenges are faced in network architecture selection. Overfitting problems are frequently encountered in high-dimensional applications. Inconsistent generalization performance is exhibited across different datasets. Robust ELM formulations [16] have been developed using weighted least squares and outlier detection mechanisms to address these limitations.

Twin extreme learning machines (TELMs) were developed by Wan et al. [17] to combine the advantages of TWSVM and ELM approaches. A simplified network structure with minimal parameter requirements is adopted in TELMs. Two non-parallel hyperplanes are constructed to separate different classes. Excellent generalization capabilities are demonstrated across various domains.

Applications span artificial intelligence systems, speech recognition technologies, remote sensing analysis, and financial management tools. The overfitting tendency of traditional ELMs is mitigated by the twin decomposition framework. Faster convergence is achieved while classification accuracy is maintained.

Projection twin extreme learning machines (PTELMs) [18] were subsequently introduced to overcome limitations of traditional projection methods. Concepts from projection twin support vector machines (PTSVMs) [19] are integrated into PTELMs. Two optimal projection directions are identified within the feature space. Within-class variance is minimized for one class while separation is maximized for the other. More discriminative information is preserved compared to conventional dimensionality reduction techniques.

The generalization performance of twin-based methods has been substantially enhanced through robust regularization strategies. Capped norm regularization techniques [20] limit the influence of individual features while maintaining computational tractability. Fisher regularization frameworks [21] leverage class-specific statistical properties to minimize within-class divergence. Correntropy-based robust TELMs [22] effectively mitigate the negative effects of noise and outliers. The applicability of twin-based methods has been further expanded to scenarios with limited labeled data through semi-supervised extensions. The Laplacian TELM [23] integrates manifold regularization to exploit geometric structure information from unlabeled samples. The Lagrangian regularized TELM [24] provides a unified framework for both supervised and semi-supervised classification. Universum-based twin methods [25] leverage non-class samples as prior knowledge to enhance generalization through additional constraints on decision boundaries. Furthermore, machine learning approaches combining feature selection with support vector classifiers [26] have been applied in bioinformatics for bacteriocin prediction, confirming the practical value of twin-based methods in real-world applications.

Despite these advances, existing robust TELM variants still face key challenges. First, they lack adaptability to different noise distributions, as fixed-parameter or unbounded loss designs (e.g., Fisher regularized TELM (FTELM), squared-fractional loss-based robust supervised TELM (SF-RSTELM)) fail to effectively model diverse or unknown noise types. Second, the balance between robustness and accuracy remains inadequate, since adaptive mechanisms developed for TSVMs have not been systematically extended to TELM frameworks. Finally, limited smoothness and differentiability in existing loss functions hinder optimization efficiency and stability. Hard-capped losses risk data omission, while soft-capped alternatives require better trade-offs between robustness, smoothness, and convergence performance under varying noise levels.

To address these limitations, a novel bounded, smooth, and symmetric adaptive fractional loss (AF-loss) is proposed, extending the squared fractional loss (SF-loss) from [20]. It is integrated into the twin extreme learning machine framework to form adaptive fractional loss TELM (AFTELM), a robust supervised learning model. AFTELM features an adaptive dual-parameter mechanism that dynamically adjusts to varying noise distributions, ensuring a balanced trade-off between robustness and accuracy. The AF-loss function effectively limits the influence of outliers and facilitates stable gradient optimization. Experimental results show that the AFTELM outperforms the FTELM, TBSVM, and other baseline methods in classification accuracy and generalization ability across noise conditions from 0% to 30%. The model ensures theoretical convergence and computational efficiency, maintaining TELM's advantages with minimal overhead.

The subsequent structure of this paper is organized as follows:

(1) A new robust loss function called AF-loss is shown. It has some important properties such as being bounded, smooth, symmetric, and noise insensitive. Moreover, the robustness of the AF-loss is analyzed according to the perspective of M estimation theory.

(2) Due to the non-convexity of the optimization model, an efficient algorithm based on CCCP [27] is proposed to solve the problem, and its convergence is proven. The CCCP algorithm for AFTELM models is presented, with convergence established according to the model's characteristics.

(3) Through experiments on artificial datasets, UCI datasets, and image datasets to validate the effectiveness of our proposed algorithm compared to other algorithms, we find that AFTELMs have advantages over several other methods in terms of robustness and feasibility.

In Section 2, we introduce some related models, such as the TELM, TBSVM, FTELM, and PTELM. In Section 3, we integrate the AF-loss function into the model. In Section 4, we solve the convergence. A large number of data experiments are conducted to test the operation of the model in Section 5. The convergence curve analysis is presented in Section 6, and statistical test comparisons are conducted in Section 7. Section 8 is a summary of the full text.

2. Related work

Before introducing the model proposed in this paper, we first provide a brief overview of four classification models: TBSVM, TELM, FTELM, and PTELM.

2.1. Twin bounded support vector machine (TBSVM)

The core idea of TBSVM is to construct two hyperplanes that are respectively proximal to samples of two classes, decomposing the original problem into two smaller-scale QPP problems, thereby significantly improving computational efficiency.

Consider a binary classification sample set where Class +1 contains $X_1 = \{\boldsymbol{x}_i \in \mathbb{R}^d \mid i = 1, 2, \ldots, m_1\}$ with m_1 samples, and Class -1 contains $X_2 = \{\boldsymbol{x}_j \in \mathbb{R}^d \mid j = 1, 2, \ldots, m_2\}$ with m_2 samples. The total number of samples is $m = m_1 + m_2$, and the feature dimension is d.

The objective of TBSVM is to learn two hyperplanes:

$$H_1 : \boldsymbol{w}_1^\top \boldsymbol{x} + b_1 = 0, \quad H_2 : \boldsymbol{w}_2^\top \boldsymbol{x} + b_2 = 0, \tag{2.1}$$

where $\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathbb{R}^d$ are normal vectors, and $b_1, b_2 \in \mathbb{R}$ are bias terms.

The classification rule this: For a new sample $\boldsymbol{x}$, calculate its distances to H_1 and H_2 as

$$d_1 = |\boldsymbol{w}_1^\top \boldsymbol{x} + b_1|/\|\boldsymbol{w}_1\|, \tag{2.2}$$

$$d_2 = |\boldsymbol{w}_2^\top \boldsymbol{x} + b_2|/\|\boldsymbol{w}_2\|. \tag{2.3}$$

If $d_1 < d_2$, the sample is classified as Class +1; otherwise, it is classified as Class -1.

TBSVM determines the hyperplane parameters by solving two independent QPP problems. The design principles are that H_1 should be as close as possible to Class +1 samples while maintaining a margin of at least 1 from Class -1 samples, and H_2 should be as close as possible to Class -1 samples while maintaining a margin of at least 1 from Class +1 samples.

Slack variables $\xi_i \geq 0$ and $\eta_j \geq 0$ are introduced to handle non-linearly separable cases. The optimization models are formulated as follows.

First optimization problem:

$$\begin{aligned} \min_{\boldsymbol{w}_1,b_1,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\boldsymbol{w}_1\|^2 + \frac{c_1}{2}\sum_{i=1}^{m_1}\xi_i^2 \\ \text{s.t.} \quad & \boldsymbol{w}_1^\top \boldsymbol{x}_i + b_1 \geq 1 - \xi_i, \quad \forall \boldsymbol{x}_i \in X_1 \\ & \boldsymbol{w}_1^\top \boldsymbol{x}_j + b_1 \leq -1, \quad \forall \boldsymbol{x}_j \in X_2 \\ & \xi_i \geq 0, \quad i = 1, 2, \ldots, m_1, \end{aligned} \tag{2.4}$$

where $c_1 > 0$ is a penalty parameter controlling the penalty intensity for Class +1 samples that violate the constraints. m_1 is defined as the count of samples from Class +1 that violate the constraint conditions, specifically referring to the total number of samples in this class failing to satisfy the classification constraints, which corresponds to the number of samples with a slack variable $\xi_i > 0$.

Second optimization problem :

$$\begin{aligned} \min_{\boldsymbol{w}_2,b_2,\boldsymbol{\eta}} \quad & \frac{1}{2}\|\boldsymbol{w}_2\|^2 + \frac{c_2}{2}\sum_{j=1}^{m_2}\eta_j^2 \\ \text{s.t.} \quad & \boldsymbol{w}_2^\top \boldsymbol{x}_j + b_2 \geq 1 - \eta_j, \quad \forall \boldsymbol{x}_j \in X_2 \\ & \boldsymbol{w}_2^\top \boldsymbol{x}_i + b_2 \leq -1, \quad \forall \boldsymbol{x}_i \in X_1 \\ & \eta_j \geq 0, \quad j = 1, 2, \ldots, m_2, \end{aligned} \tag{2.5}$$

where $c_2 > 0$ is the penalty parameter for Class -1. m_2 is defined as the count of samples from Class -1 that violate the constraint conditions, specifically referring to the total number of samples in this class failing to satisfy the classification constraints, which corresponds to the number of samples with a slack variable $\eta_j > 0$.

TBSVM, through the design of dual hyperplanes and problem decomposition strategies, achieves efficient classification in large-scale data classification tasks while maintaining good generalization performance. This theoretical framework provides a new perspective for the optimization of classification models in statistical learning. Future research directions include enhancing noise robustness and efficiently constructing multi-class models to expand its applicability in practical scenarios.

2.2. Twin extreme learning machine (TELM)

The TELM is an algorithm proposed to improve the performance of traditional ELMs in binary classification tasks. Drawing inspiration from TSVMs, TELM introduces two non-parallel classification hyperplanes to enhance classification capability. This section presents its mathematical model and dual formulation.

Consider a training dataset $T = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ are input samples and $y_i \in \{-1, +1\}$ are sample labels. For binary classification problems, TELM aims to learn two non-parallel hyperplanes.

Let $\boldsymbol{h}(\boldsymbol{x}) = (h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_L(\boldsymbol{x})) \in \mathbb{R}^{1\times L}$ be the hidden layer output vector, where L is the number of hidden nodes. Each node function is defined as:

$$h_i(\boldsymbol{x}) = G\left(\sum_{j=1}^{d} x_j \omega_{ji} + b_i\right), \tag{2.6}$$

where $G(\cdot)$ is an activation function (e.g., sigmoid function $G(x) = \frac{1}{1+e^{-x}}$ or ReLU function $G(x) = \max(0, x)$), ω_{ji} is the input weight connecting the j-th feature to the i-th hidden node, and b_i is the bias term of the i-th hidden node.

TELM determines the parameters of the two hyperplanes by solving the following two optimization problems.

First optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\beta}_1, \boldsymbol{\xi}_1} \quad & \frac{1}{2}\|\boldsymbol{H}_1\boldsymbol{\beta}_1\|_2^2 + C_1\boldsymbol{e}_2^\top\boldsymbol{\xi}_1 \\ \text{s.t.} \quad & -\boldsymbol{H}_2\boldsymbol{\beta}_1 + \boldsymbol{\xi}_1 \geq \boldsymbol{e}_2 \\ & \boldsymbol{\xi}_1 \geq \boldsymbol{0}, \end{aligned} \tag{2.7}$$

where $\boldsymbol{H}_1 = [\boldsymbol{h}(\boldsymbol{x}_1)^\top, \boldsymbol{h}(\boldsymbol{x}_2)^\top, \ldots, \boldsymbol{h}(\boldsymbol{x}_{l_1})^\top]^\top \in \mathbb{R}^{l_1 \times L}$ and $\boldsymbol{H}_2 = [\boldsymbol{h}(\boldsymbol{x}_1)^\top, \boldsymbol{h}(\boldsymbol{x}_2)^\top, \ldots, \boldsymbol{h}(\boldsymbol{x}_{l_2})^\top]^\top \in \mathbb{R}^{l_2 \times L}$ are the hidden layer outputs of positive and negative class samples, respectively, $\boldsymbol{\beta}_1 \in \mathbb{R}^L$ is the output weight vector connecting the hidden layer to the output layer, $C_1 > 0$ is a regularization parameter, $\boldsymbol{e}_1 \in \mathbb{R}^{l_1}$ and $\boldsymbol{e}_2 \in \mathbb{R}^{l_2}$ are vectors with all elements equal to 1, and $\boldsymbol{\xi}_1$ is a slack vector to handle non-linear separability.

The objective is to make positive class samples as close as possible to hyperplane f_1 while keeping negative class samples far from it.

Second optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\beta}_2, \boldsymbol{\xi}_2} \quad & \frac{1}{2}\|\boldsymbol{H}_2\boldsymbol{\beta}_2\|_2^2 + C_3\boldsymbol{e}_1^\top\boldsymbol{\xi}_2 \\ \text{s.t.} \quad & \boldsymbol{H}_1\boldsymbol{\beta}_2 + \boldsymbol{\xi}_2 \geq \boldsymbol{e}_1 \\ & \boldsymbol{\xi}_2 \geq \boldsymbol{0}. \end{aligned} \tag{2.8}$$

Similarly, the objective here is to make negative class samples as close as possible to hyperplane f_2 while keeping positive class samples far from it.

According to the Karush–Kuhn–Tucker (KKT) conditions, the dual problem of (2.7) is:

$$\begin{aligned} \min_{\boldsymbol{\theta}_1} \quad & \frac{1}{2}\boldsymbol{\theta}_1^\top\boldsymbol{Q}_1\boldsymbol{\theta}_1 - \boldsymbol{e}_2^\top\boldsymbol{\theta}_1 \\ \text{s.t.} \quad & \boldsymbol{0} \leq \boldsymbol{\theta}_1 \leq C_1\boldsymbol{e}_2, \end{aligned} \tag{2.9}$$

where $\boldsymbol{Q}_1$ is derived from the KKT conditions.

Similarly, the dual problem of (2.8) is:

$$\begin{aligned} \min_{\boldsymbol{\theta}_2} \quad & \frac{1}{2}\boldsymbol{\theta}_2^\top\boldsymbol{Q}_2\boldsymbol{\theta}_2 - \boldsymbol{e}_1^\top\boldsymbol{\theta}_2 \\ \text{s.t.} \quad & \boldsymbol{0} \leq \boldsymbol{\theta}_2 \leq C_2\boldsymbol{e}_1, \end{aligned} \tag{2.10}$$

where $\boldsymbol{Q}_2$ is derived from the KKT conditions.

2.3. Fisher regularized twin extreme learning machine (FTELM)

The FTELM is an enhanced algorithm that combines the advantages of TELM with Fisher discriminant analysis. By incorporating Fisher regularization into the TELM framework, FTELM not

only constructs two non-parallel classification hyperplanes but also minimizes the within-class scatter of samples, thereby improving classification performance and generalization capability. This section presents its mathematical model and dual formulation.

Consider a training dataset $T = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ are input samples and $y_i \in \{-1, +1\}$ are sample labels. For binary classification problems, FTELM aims to learn two non-parallel hyperplanes with Fisher regularization.

In Fisher discriminant analysis, the within-class scatter matrix in the ELM feature space is defined as:

$$\boldsymbol{S}_w = \boldsymbol{H}_1^\top \left(\boldsymbol{I}_{l_1} - \frac{1}{l_1} \boldsymbol{e}_1 \boldsymbol{e}_1^\top \right) \boldsymbol{H}_1 + \boldsymbol{H}_2^\top \left(\boldsymbol{I}_{l_2} - \frac{1}{l_2} \boldsymbol{e}_2 \boldsymbol{e}_2^\top \right) \boldsymbol{H}_2, \tag{2.11}$$

where $\boldsymbol{H}_1 = [\boldsymbol{h}(\boldsymbol{x}_1)^\top, \boldsymbol{h}(\boldsymbol{x}_2)^\top, \ldots, \boldsymbol{h}(\boldsymbol{x}_{l_1})^\top]^\top \in \mathbb{R}^{l_1 \times L}$ and $\boldsymbol{H}_2 = [\boldsymbol{h}(\boldsymbol{x}_1)^\top, \boldsymbol{h}(\boldsymbol{x}_2)^\top, \ldots, \boldsymbol{h}(\boldsymbol{x}_{l_2})^\top]^\top \in \mathbb{R}^{l_2 \times L}$ are the hidden layer outputs of positive and negative class samples, respectively, $\boldsymbol{I}_{l_1} \in \mathbb{R}^{l_1 \times l_1}$ and $\boldsymbol{I}_{l_2} \in \mathbb{R}^{l_2 \times l_2}$ are identity matrices, $\boldsymbol{e}_1 \in \mathbb{R}^{l_1}$ and $\boldsymbol{e}_2 \in \mathbb{R}^{l_2}$ are vectors with all elements equal to 1, and l_1 and l_2 denote the numbers of positive and negative class samples, respectively.

The Fisher regularization term is then given by:

$$\mathcal{R}_F(\boldsymbol{\beta}) = \frac{1}{2} \boldsymbol{\beta}^\top \boldsymbol{S}_w \boldsymbol{\beta}, \tag{2.12}$$

which penalizes large within-class scatter and encourages the learned model to maintain compact class distributions.

FTELM determines the parameters of the two hyperplanes by solving the following two optimization problems.

First optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\beta}_1, \boldsymbol{\xi}_1} \quad & \frac{1}{2} \|\boldsymbol{H}_1 \boldsymbol{\beta}_1\|_2^2 + C_1 \boldsymbol{e}_2^\top \boldsymbol{\xi}_1 + \frac{\lambda_1}{2} \boldsymbol{\beta}_1^\top \boldsymbol{S}_w \boldsymbol{\beta}_1 \\ \text{s.t.} \quad & -\boldsymbol{H}_2 \boldsymbol{\beta}_1 + \boldsymbol{\xi}_1 \geq \boldsymbol{e}_2 \\ & \boldsymbol{\xi}_1 \geq \boldsymbol{0}, \end{aligned} \tag{2.13}$$

where $\boldsymbol{\beta}_1 \in \mathbb{R}^L$ is the output weight vector connecting the hidden layer to the output layer, $C_1 > 0$ is a regularization parameter controlling the trade-off between training error and margin maximization, $\lambda_1 > 0$ is the Fisher regularization parameter that controls the influence of within-class scatter minimization, and $\boldsymbol{\xi}_1 \in \mathbb{R}^{l_2}$ is a slack vector to handle non-linear separability.

Second optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\beta}_2, \boldsymbol{\xi}_2} \quad & \frac{1}{2} \|\boldsymbol{H}_2 \boldsymbol{\beta}_2\|_2^2 + C_2 \boldsymbol{e}_1^\top \boldsymbol{\xi}_2 + \frac{\lambda_2}{2} \boldsymbol{\beta}_2^\top \boldsymbol{S}_w \boldsymbol{\beta}_2 \\ \text{s.t.} \quad & \boldsymbol{H}_1 \boldsymbol{\beta}_2 + \boldsymbol{\xi}_2 \geq \boldsymbol{e}_1 \\ & \boldsymbol{\xi}_2 \geq \boldsymbol{0}, \end{aligned} \tag{2.14}$$

where $\boldsymbol{\beta}_2 \in \mathbb{R}^L$ is the output weight vector, $C_2 > 0$ is the regularization parameter, $\lambda_2 > 0$ is the Fisher regularization parameter, and $\boldsymbol{\xi}_2 \in \mathbb{R}^{l_1}$ is the slack vector.

Similarly, the objective here is to make negative class samples as close as possible to hyperplane f_2 while keeping positive class samples far from it, and simultaneously minimizing the within-class scatter.

According to the KKT conditions, the dual problem of (2.13) is

$$\begin{aligned} \min_{\boldsymbol{\theta}_1} \quad & \frac{1}{2}\boldsymbol{\theta}_1^\top \boldsymbol{Q}_1 \boldsymbol{\theta}_1 - \boldsymbol{e}_2^\top \boldsymbol{\theta}_1 \\ \text{s.t.} \quad & \mathbf{0} \le \boldsymbol{\theta}_1 \le C_1 \boldsymbol{e}_2, \end{aligned} \tag{2.15}$$

where $\boldsymbol{Q}_1 = \boldsymbol{H}_2(\boldsymbol{H}_1^\top \boldsymbol{H}_1 + \lambda_1 \boldsymbol{S}_w)^{-1}\boldsymbol{H}_2^\top \in \mathbb{R}^{l_2 \times l_2}$.

Similarly, the dual problem of (2.14) is

$$\begin{aligned} \min_{\boldsymbol{\theta}_2} \quad & \frac{1}{2}\boldsymbol{\theta}_2^\top \boldsymbol{Q}_2 \boldsymbol{\theta}_2 - \boldsymbol{e}_1^\top \boldsymbol{\theta}_2 \\ \text{s.t.} \quad & \mathbf{0} \le \boldsymbol{\theta}_2 \le C_2 \boldsymbol{e}_1, \end{aligned} \tag{2.16}$$

where $\boldsymbol{Q}_2 = \boldsymbol{H}_1(\boldsymbol{H}_2^\top \boldsymbol{H}_2 + \lambda_2 \boldsymbol{S}_w)^{-1}\boldsymbol{H}_1^\top \in \mathbb{R}^{l_1 \times l_1}$.

Once the dual problems (2.15) and (2.16) are solved to obtain the optimal solutions $\boldsymbol{\theta}_1^*$ and $\boldsymbol{\theta}_2^*$, the output weight vectors can be computed as

$$\boldsymbol{\beta}_1^* = -(\boldsymbol{H}_1^\top \boldsymbol{H}_1 + \lambda_1 \boldsymbol{S}_w)^{-1}\boldsymbol{H}_2^\top \boldsymbol{\theta}_1^*, \tag{2.17}$$

$$\boldsymbol{\beta}_2^* = (\boldsymbol{H}_2^\top \boldsymbol{H}_2 + \lambda_2 \boldsymbol{S}_w)^{-1}\boldsymbol{H}_1^\top \boldsymbol{\theta}_2^*. \tag{2.18}$$

For a new test sample $\boldsymbol{x}$, its class label is determined by the decision function

$$y(\boldsymbol{x}) = \arg\min_{i=1,2} \frac{|\boldsymbol{h}(\boldsymbol{x})\boldsymbol{\beta}_i^*|}{\|\boldsymbol{\beta}_i^*\|}. \tag{2.19}$$

That is, the sample is assigned to the class corresponding to the hyperplane to which it has the smallest perpendicular distance.

2.4. Projection twin etreme learning machine (PTELM)

Traditional kernel-based SVMs have limitations in handling nonlinear problems and fail to accommodate samples with different distributions effectively. By introducing the TELM feature space and combining it with the projection concept from projection twin support vector machines (PTSVMs), we construct the PTELM to seek a pair of non-parallel projection directions $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$. We formulate the following optimization problems:

$$\begin{aligned} \min_{\boldsymbol{\beta}_1} \quad & \frac{1}{2}\sum_{i\in I_1} \left(\boldsymbol{h}(\boldsymbol{x}_i)^\top \boldsymbol{\beta}_1 - \boldsymbol{h}(\boldsymbol{m}_1)^\top \boldsymbol{\beta}_1\right)^2 \\ & + C_3 \sum_{j\in I_2} \max\left(0, 1 + \left(\boldsymbol{h}(\boldsymbol{x}_j) - \boldsymbol{h}(\boldsymbol{m}_1)\right)^\top \boldsymbol{\beta}_1\right) + \frac{C_1}{2}\|\boldsymbol{\beta}_1\|^2, \end{aligned} \tag{2.20}$$

$$\begin{aligned} \min_{\boldsymbol{\beta}_2} \quad & \frac{1}{2}\sum_{j\in I_2} \left(\boldsymbol{h}(\boldsymbol{x}_j)^\top \boldsymbol{\beta}_2 - \boldsymbol{h}(\boldsymbol{m}_2)^\top \boldsymbol{\beta}_2\right)^2 \\ & + C_4 \sum_{i\in I_1} \max\left(0, 1 - (\boldsymbol{h}(\boldsymbol{x}_i) - \boldsymbol{h}(\boldsymbol{m}_2))^\top \boldsymbol{\beta}_2\right) + \frac{C_2}{2}\|\boldsymbol{\beta}_2\|^2, \end{aligned} \tag{2.21}$$

where $C_1, C_2, C_3, C_4 > 0$ are trade-off constants, $\boldsymbol{h}(\boldsymbol{x})$ represents the feature vector of sample $\boldsymbol{x}$ in the TELM feature space, and $\boldsymbol{m}_1 = \frac{1}{|I_1|}\sum_{i\in I_1}\boldsymbol{x}_i$ and $\boldsymbol{m}_2 = \frac{1}{|I_2|}\sum_{j\in I_2}\boldsymbol{x}_j$ are the centers of the positive and negative classes, respectively.

For optimization problem (2.20), the objective function consists of three terms. The first term minimizes the variance of projected positive class samples in the TELM feature space, encouraging them to cluster around their mean. The second term employs the hinge loss function $\max\left(0, 1 + (\boldsymbol{h}(\boldsymbol{x}_j) - \boldsymbol{h}(\boldsymbol{m}_1))^\top\boldsymbol{\beta}_1\right)$ to keep negative class instances away from the positive class center in the projection space. The third term is the L_2-norm regularization of projection direction $\boldsymbol{\beta}_1$, which regulates model complexity and prevents overfitting.

The parameters C_1 and C_3 balance these three terms. When C_1 is large, the objective function focuses more on reducing model complexity; when C_3 is large, it focuses more on reducing model loss. Optimization problem (2.21) has a similar interpretation.

For notational simplicity, we define

$$\boldsymbol{A}_k = \begin{bmatrix} (\boldsymbol{h}(\boldsymbol{x}_1) - \boldsymbol{h}(\boldsymbol{m}_k))^\top \\ (\boldsymbol{h}(\boldsymbol{x}_2) - \boldsymbol{h}(\boldsymbol{m}_k))^\top \\ \vdots \\ (\boldsymbol{h}(\boldsymbol{x}_{l_1}) - \boldsymbol{h}(\boldsymbol{m}_k))^\top \end{bmatrix} \in \mathbb{R}^{l_1\times L}, \tag{2.22}$$

$$\boldsymbol{B}_k = \begin{bmatrix} (\boldsymbol{h}(\boldsymbol{x}_1) - \boldsymbol{h}(\boldsymbol{m}_k))^\top \\ (\boldsymbol{h}(\boldsymbol{x}_2) - \boldsymbol{h}(\boldsymbol{m}_k))^\top \\ \vdots \\ (\boldsymbol{h}(\boldsymbol{x}_{l_2}) - \boldsymbol{h}(\boldsymbol{m}_k))^\top \end{bmatrix} \in \mathbb{R}^{l_2\times L}, \tag{2.23}$$

where $k \in \{1, 2\}$.

The matrix formulations of problems (2.20) and (2.21) can be expressed as

$$\begin{aligned} \min_{\boldsymbol{\beta}_1,\boldsymbol{\eta}_1,\boldsymbol{\xi}_1} \quad & \frac{1}{2}\boldsymbol{\eta}_1^\top\boldsymbol{\eta}_1 + C_3\boldsymbol{e}_2^\top\boldsymbol{\xi}_1 + \frac{C_1}{2}\|\boldsymbol{\beta}_1\|^2 \\ \text{s.t.} \quad & \boldsymbol{A}_1\boldsymbol{\beta}_1 = \boldsymbol{\eta}_1 \\ & -\boldsymbol{B}_1\boldsymbol{\beta}_1 \geq \boldsymbol{e}_2 - \boldsymbol{\xi}_1 \\ & \boldsymbol{\xi}_1 \geq \boldsymbol{0}, \end{aligned} \tag{2.24}$$

$$\begin{aligned} \min_{\boldsymbol{\beta}_2,\boldsymbol{\eta}_2,\boldsymbol{\xi}_2} \quad & \frac{1}{2}\boldsymbol{\eta}_2^\top\boldsymbol{\eta}_2 + C_4\boldsymbol{e}_1^\top\boldsymbol{\xi}_2 + \frac{C_2}{2}\|\boldsymbol{\beta}_2\|^2 \\ \text{s.t.} \quad & \boldsymbol{B}_2\boldsymbol{\beta}_2 = \boldsymbol{\eta}_2, \\ & \boldsymbol{A}_2\boldsymbol{\beta}_2 \geq \boldsymbol{e}_1 - \boldsymbol{\xi}_2, \\ & \boldsymbol{\xi}_2 \geq \boldsymbol{0}. \end{aligned} \tag{2.25}$$

The primal problem (2.24) with variables $\boldsymbol{\beta}_1$, $\boldsymbol{\eta}_1$, $\boldsymbol{\xi}_1$ can be converted to the following dual problem with variables $\boldsymbol{u}_1$, $\boldsymbol{\alpha}_1$:

$$\begin{aligned} \min_{\boldsymbol{u}_1,\boldsymbol{\alpha}_1} \quad & \frac{1}{2}\begin{bmatrix}\boldsymbol{u}_1^\top & \boldsymbol{\alpha}_1^\top\end{bmatrix}\begin{bmatrix}\boldsymbol{A}_1\boldsymbol{A}_1^\top + C_1\boldsymbol{I} & -\boldsymbol{A}_1\boldsymbol{B}_1^\top \\ -\boldsymbol{B}_1\boldsymbol{A}_1^\top & \boldsymbol{B}_1\boldsymbol{B}_1^\top\end{bmatrix}\begin{bmatrix}\boldsymbol{u}_1 \\ \boldsymbol{\alpha}_1\end{bmatrix} - \boldsymbol{\alpha}_1^\top\boldsymbol{e}_2 \\ \text{s.t.} \quad & \boldsymbol{0} \leq \boldsymbol{\alpha}_1 \leq C_3\boldsymbol{e}_2. \end{aligned} \tag{2.26}$$

Similarly, the dual problem of (2.25) is

$$\begin{aligned} \min_{\boldsymbol{u}_2,\boldsymbol{\alpha}_2} \quad & \frac{1}{2}\begin{bmatrix}\boldsymbol{u}_2^\top & \boldsymbol{\alpha}_2^\top\end{bmatrix}\begin{bmatrix}\boldsymbol{B}_2\boldsymbol{B}_2^\top + C_2\boldsymbol{I} & \boldsymbol{B}_2\boldsymbol{A}_2^\top \\ \boldsymbol{A}_2\boldsymbol{B}_2^\top & \boldsymbol{A}_2\boldsymbol{A}_2^\top\end{bmatrix}\begin{bmatrix}\boldsymbol{u}_2 \\ \boldsymbol{\alpha}_2\end{bmatrix} - \boldsymbol{\alpha}_2^\top\boldsymbol{e}_1 \\ \text{s.t.} \quad & \boldsymbol{0} \le \boldsymbol{\alpha}_2 \le C_4\boldsymbol{e}_1. \end{aligned} \tag{2.27}$$

The solutions $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ to the primal problems (2.26) and (2.27) can be obtained as

$$\boldsymbol{\beta}_1 = \frac{1}{C_1}\left(\boldsymbol{A}_1^\top\boldsymbol{u}_1 - \boldsymbol{B}_1^\top\boldsymbol{\alpha}_1\right), \tag{2.28}$$

$$\boldsymbol{\beta}_2 = \frac{1}{C_2}\left(\boldsymbol{B}_2^\top\boldsymbol{u}_2 + \boldsymbol{A}_2^\top\boldsymbol{\alpha}_2\right). \tag{2.29}$$

Once the solutions $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ are obtained from Eqs (2.24) and (2.25), a new data point $\boldsymbol{x}$ is assigned to the positive or negative class according to the following decision function

$$y = \arg\min_{k\in\{1,2\}} \left|\left(\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{h}(\boldsymbol{m}_k)\right)^\top \boldsymbol{\beta}_k\right|. \tag{2.30}$$

3. Main contributions

In this section, the proposed AF-loss is first introduced, and its properties are analyzed. The AFTELM model is then constructed by integrating the proposed loss function into the TELM framework, followed by a convergence analysis [27] to examine its stability. To ensure a fair evaluation of the AF-loss, a baseline model, termed squared fractional loss TELM (SFTELM), is also constructed for the ablation study.

3.1. Adaptive fractional loss

The AF-loss function is a bounded, smooth, and non-convex loss function that exhibits strong robustness to noise. It is specifically designed based on the SF-loss [20].

Definition 1. *Given a vector u, the AF-loss is defined as*

$$L_a(u) = \frac{u^2}{bu^2 + k^\gamma}, \tag{3.1}$$

where the parameters $b, k, \gamma \in (0, +\infty)$. Here, k^γ and b are well-defined constant terms.

Importantly, the proposed AF-loss function generalizes the SF-loss introduced in [20]. Specifically, when $k = 1$ and $\gamma = 1$, the AF-loss reduces exactly to the SF-loss:

$$L_r(u) = \frac{u^2}{bu^2 + 1}, \tag{3.2}$$

where the parameter $b \in (0, +\infty)$ controls the upper bound $\frac{1}{b}$ of the loss function. By introducing the additional parameters k and γ, AF-loss provides a more flexible mechanism to adapt the loss shape and upper bound, enhancing its adaptability to diverse data characteristics.

Figure 1 illustrates the AF-loss under various parameter settings. The AF-loss function controls robustness through the scale parameter k and the shape parameter γ. These parameters adjust the

trade-off between sensitivity to nominal data and resilience to outliers. A steep slope near the origin increases sensitivity to small deviations, while a flatter curve at larger values reduces the influence of outliers. The scale parameter k defines the saturation threshold and dynamic range, with smaller values leading to stronger outlier suppression. The shape parameter γ refines the curve in the transition region, enabling finer performance tuning. This design simplifies parameter adjustment: k determines overall robustness, and γ provides secondary refinement. Compared to fixed-parameter loss functions like SF-loss, AF-loss offers greater adaptability, interpretability, and optimization stability, improving performance across classification tasks.

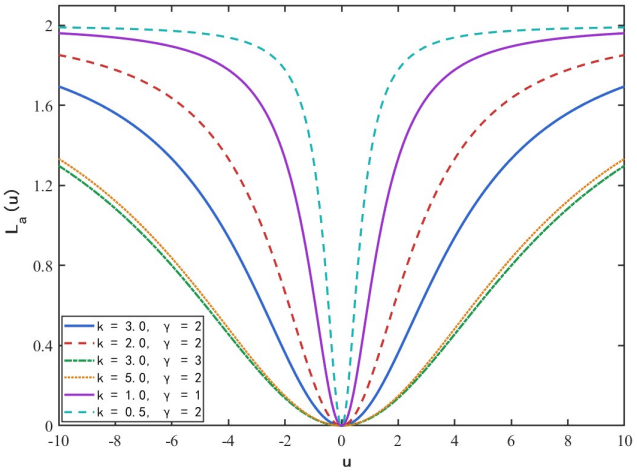


Figure 1. AF-loss with different parameter settings.

3.2. Properties of AF-loss function

Property 1. *$L_a(u)$ is bounded, which can ensure better robustness.*

Proof.

$$\lim_{u\to\infty} L_a(u) = \lim_{u\to\infty} \frac{u^2}{bu^2 + k^\gamma} = \lim_{u\to\infty} \frac{1}{b + \frac{k^\gamma}{u^2}} = \frac{1}{b}, \tag{3.3}$$

where b, k, γ are constants. Therefore, $L_a(u)$ is a bounded function. □

Property 2. *$L_a(u)$ is a differentiable function that can help optimize better.*

Proof.

$$L_a'(u) = \frac{2uk^\gamma}{(bu^2 + k^\gamma)^2}. \tag{3.4}$$

Therefore, $L_a(u)$ is differentiable. □

Property 3. *$L_a(u)$ is a symmetrical function.*

Proof. Calculate $L_a(u)$ and $L_a(-u)$ separately and obtain

$$L_a(u) = \frac{u^2}{bu^2 + k^\gamma}, \tag{3.5}$$

$$L_a(-u) = \frac{(-u)^2}{b(-u)^2 + k^\gamma} = \frac{u^2}{bu^2 + k^\gamma}. \tag{3.6}$$

Since $L_a(u) = L_a(-u)$, $L_a(u)$ is symmetrical. □

Property 4. *$L_a(u)$ is a non-convex function.*

Proof. For computational convenience, let $c = k^\gamma$ and

$$L'_a(u) = \frac{2cu}{(bu^2 + c)^2}, \tag{3.7}$$

$$\begin{aligned} L''_a(u) &= \frac{2c \times (bu^2 + c)^2 - 2cu \times 4bu(bu^2 + c)}{(bu^2 + c)^4} \\ &= \frac{2c(bu^2 + c)\left[(bu^2 + c) - 4bu^2\right]}{(bu^2 + c)^4} \\ &= \frac{2c(c - 3bu^2)}{(bu^2 + c)^3}. \end{aligned} \tag{3.8}$$

When $b = 1$ and $c = 3$, the second-order derivative becomes:

$$L''_a(u) = \frac{2 \times 3 \times (3 - 3 \times 1 \times u^2)}{(1 \times u^2 + 3)^3} = \frac{18(1 - u^2)}{(u^2 + 3)^3}. \tag{3.9}$$

We analyze the concavity and convexity as follows:
When $|u| > 1$, $1 - u^2 < 0$, so $L''_a(u) < 0$. The function is concave in this interval.
When $|u| < 1$, $1 - u^2 > 0$, so $L''_a(u) > 0$. The function is convex in this interval.
So, $L_a(u)$ is a non-convex function. □

3.3. Robustness analysis of AF-loss function

Clearly, the new loss function $L_a(u)$ is bounded. From a robust statistics perspective, the $L_a(u)$ shows noise insensitivity, which ensures superior robustness. The derivative of $L_a(u)$ is expressed as:

$$L'_a(u) = \frac{2uk^\gamma}{(bu^2 + k^\gamma)^2}, \tag{3.10}$$

and we have

$$\lim_{u\to\infty} \frac{2uk^\gamma}{(bu^2 + k^\gamma)^2} = 0. \tag{3.11}$$

This indicates that the influence of samples with large residuals diminishes as the residual increases, thereby suppressing the adverse effects of outliers.

3.4. Comparison with other bounded loss functions

In Figure 2, the performance of the proposed AF-loss function is compared with that of other advanced loss functions, including Welsch loss, truncated least squares loss, capped L_1-norm loss, and SF-loss. With parameters $b = 0.5$, $k = 0.5$, and $\gamma = 2$, AF-loss exhibits clear advantages: It exhibits a faster growth rate near the origin than Welsch loss, which enhances the fitting constraint on

normal samples. Meanwhile, in regions with larger prediction errors, it maintains a smoother growth trend, effectively suppressing the negative impact of outliers. Compared to truncated least squares loss and capped L_1-norm loss, AF-loss stands out for its smoothness: The latter two losses have discontinuities in derivatives and sharp corners at their truncation points, whereas AF-loss remains continuously differentiable, ensuring stable convergence for gradient-based optimization algorithms. Additionally, unlike SF-loss (which has a fixed loss curve shape), AF-loss enables flexible adjustment of its loss profile through parameters k and γ. This adaptability allows AF-loss to balance sensitivity to normal samples and robustness to outliers, depending on dataset characteristics and noise distribution. The combination of smoothness, boundedness, and adjustability improves model generalization while maintaining optimization efficiency and stability.

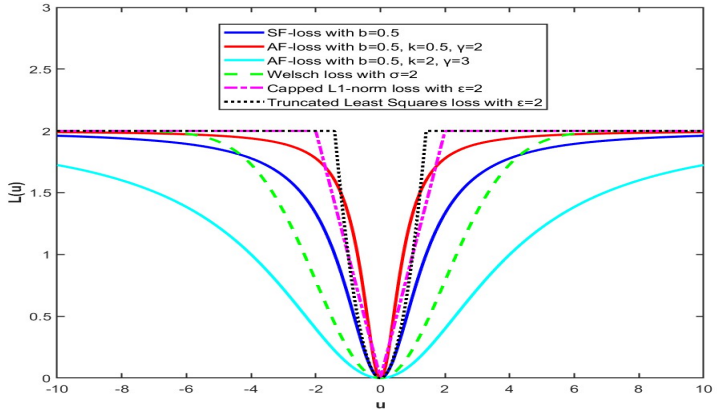


Figure 2. Comparison of AF-loss with capped L_1-norm loss, Welsch loss, truncated least squares loss, and SF-loss.

3.5. Adaptive fractional loss twin extreme learning machine (AFTELM)

To suppress the adverse impact caused by the noise, we propose to incorporate AFTELM to the framework of TELM, which is represented as

$$\min_{\beta_1} \frac{1}{2}\|H_1\beta_1\|_2^2 + C_1 \sum_{j=1}^{m_2} L_a(1 + h(x_j)\beta_1), \tag{3.12}$$

$$\min_{\beta_2} \frac{1}{2}\|H_2\beta_2\|_2^2 + C_2 \sum_{i=1}^{m_1} L_a(1 - h(x_i)\beta_2), \tag{3.13}$$

where $C_1 \geq 0$ and $C_2 \geq 0$.

It has been observed that the optimization problems in (3.12) and (3.13) are non-convex due to the non-convexity of the loss function. To address this, we employ the CCCP method [27]. The loss function can be expressed as the difference between the sum of a convex function and a concave function. The specific expressions for the convex and concave functions are

$$L_{a_1}(u) = \frac{u^2}{k^\gamma}, \tag{3.14}$$

$$L_{a_2}(u) = -\frac{u^2}{k^\gamma} + \frac{u^2}{bu^2 + k^\gamma}. \tag{3.15}$$

Further, the optimization problem (3.12) can be rewritten as

$$\min_{\beta_1} \frac{1}{2}\beta_1^T H_1^T H_1 \beta_1 + \frac{C_1}{2}\sum_{j=1}^{m_2}(L_{a_1}(1 + h(x_j)\beta_1) + \frac{C_1}{2}\sum_{j=1}^{m_2} L_{a_2}(1 + h(x_j)\beta_1)). \tag{3.16}$$

Similarly, the optimization problem (3.13) can be rewritten as

$$\min_{\beta_2} \frac{1}{2}\beta_2^T H_2^T H_2 \beta_2 + \frac{C_2}{2}\sum_{i=1}^{m_1}(L_{a_1}(1 - h(x_i)\beta_2) + \frac{C_2}{2}\sum_{i=1}^{m_1} L_{a_2}(1 - h(x_i)\beta_2)). \tag{3.17}$$

Since (3.16) and (3.17) are similar, we will only show the solution process for (3.16), which can also be expressed as

$$\min_{\beta_1} \underbrace{\frac{1}{2}\beta_1^T H_1^T H_1 \beta_1 + \frac{C_1}{2}\sum_{j=1}^{m_2} L_{a_1}(1 + h(x_j)\beta_1)}_{J_{\text{vex}}(\beta_1)} + \underbrace{\frac{C_1}{2}\sum_{j=1}^{m_2} L_{a_2}(1 + h(x_j)\beta_1)}_{J_{\text{cav}}(\beta_1)}. \tag{3.18}$$

The solution to the above optimization issue can be obtained by addressing the following equation

$$\min_{\beta_1} J(\beta_1) = J_{vex}(\beta_1) + J_{cav}(\beta_1). \tag{3.19}$$

CCCP iteration procedures are

$$\beta_1^{(t+1)} = \arg\min_{\beta_1}\left\{J_{vex}(\beta_1) + \nabla J_{cav}(\beta_1^{(t)})\beta_1\right\}. \tag{3.20}$$

Let $\eta_1^{(t)} = \nabla J_{\text{cav}}(\beta_1^{(t)})$, where $\eta_1^{(t)} = \left[\eta_{1_1}^{(t)}, \eta_{1_2}^{(t)}, \ldots, \eta_{1_j}^{(t)}, \ldots, \eta_{1_{m_2}}^{(t)}\right]^T \in \mathbb{R}^{m_2}$.

$$\eta_{1_j}^{(t)} = \frac{\partial J_{\text{cav}}}{\partial u_j} = \left[-\frac{2(1 + h(x_j)\beta_1)}{k^\gamma} + \frac{2(1 + h(x_j)\beta_1)k^\gamma}{\left(b(1 + h(x_j)\beta_1)^2 + k^\gamma\right)^2}\right]. \tag{3.21}$$

The gradient can express

$$\nabla J_{\text{cav}}(\beta_1^{(t)}) = C_1 H_2^T \eta_1^{(t)}. \tag{3.22}$$

The function $J_1(\beta_1)$ can be decomposed into the sum of the convex function $J_{1,vex}(\beta_1)$ and the concave function $J_{1,cav}(\beta_1)$. The function $J_2(\beta_2)$ can be decomposed into the sum of the convex function $J_{2,vex}(\beta_2)$ and the concave function $J_{2,cav}(\beta_2)$, i.e.,

$$J_1(\beta_1) = \underbrace{\frac{1}{2}\beta_1^T H_1^T H_1 \beta_1 + \frac{C_1}{2k^\gamma}\sum_{j=1}^{m_2} u_j^2}_{J_{1,\text{vex}}(\beta_1)} + \underbrace{\frac{C_1}{2}\sum_{j=1}^{m_2}\left(\frac{u_j^2}{bu_j^2 + k^\gamma} - \frac{u_j^2}{k^\gamma}\right)}_{J_{1,\text{cav}}(\beta_1)}, \tag{3.23}$$

$$J_2(\beta_2) = \underbrace{\frac{1}{2}\beta_2^T H_2^T H_2 \beta_2 + \frac{C_2}{2k^\gamma}\sum_{i=1}^{m_1} u_i^2}_{J_{2,\text{vex}}(\beta_2)} + \underbrace{\frac{C_2}{2}\sum_{i=1}^{m_1}\left(\frac{u_i^2}{bu_i^2 + k^\gamma} - \frac{u_i^2}{k^\gamma}\right)}_{J_{2,\text{cav}}(\beta_2)}, \tag{3.24}$$

where, $u_j = 1 + h(x_j)\beta_1$ and $u_i = 1 - h(x_i)\beta_2$.

Solving (3.20) is equivalent to solving the following subproblem:

$$\begin{aligned} &\min_{\beta_1} \frac{1}{2}\beta_1^T H_1^T H_1 \beta_1 + \frac{C_1}{2k^\gamma}\xi_1^T \xi_1 + \eta_1^{(t)T} H_2 \beta_1 \\ &\text{s.t. } -H_2\beta_1 + \xi_1 = e_2, \end{aligned} \tag{3.25}$$

where $e_2 \in \mathbb{R}^{m_2}$ is the unit vector. Here, we introduce Lagrange multipliers λ_1 for problem (3.25). Next, its Lagrange function is given by

$$L(\beta_1, \xi_1, \lambda_1) = \frac{1}{2}\beta_1^T H_1^T H_1 \beta_1 + \frac{C_1}{2k^\gamma}\xi_1^T \xi_1 + \eta_1^{(t)T} H_2 \beta_1 - \lambda_1^T(-H_2\beta_1 + \xi_1 - e_2). \tag{3.26}$$

Following the KKT conditions, we derive the following constraints:

$$\begin{cases} \frac{\partial L}{\partial \beta_1} = H_1^T H_1 \beta_1 + H_2^T \eta_1^{(t)} + H_2^T \lambda_1 = 0, \\ \frac{\partial L}{\partial \xi_1} = \frac{C_1}{k^\gamma}\xi_1 - \lambda_1 = 0, \\ \lambda_1^T(-H_2\beta_1 + \xi_1 - e_2) = 0. \end{cases} \tag{3.27}$$

From the KKT condition, we can obtain:

$$\begin{cases} \beta_1 = -\left(H_1^T H_1\right)^{-1} H_2^T \left(\eta_1^{(t)} + \lambda_1\right), \\ \frac{C_1}{k^\gamma}\xi_1 = \lambda_1. \end{cases} \tag{3.28}$$

Bring (3.28) into (3.25), and we can obtain the dual problem of the original problem:

$$\min_{\lambda_1} \frac{1}{2}\left(\eta_1^{(t)} + \lambda_1\right)^T H_2 \left(H_1^T H_1\right)^{-1} H_2^T \left(\eta_1^{(t)} + \lambda_1\right) - e_2^T \lambda_1. \tag{3.29}$$

Using a similar approach, we can obtain the dual problem of Eq (3.13)

$$\min_{\lambda_2} \frac{1}{2}\left(\lambda_2 - \eta_2^{(t)}\right)^T H_1 \left(H_2^T H_2\right)^{-1} H_1^T \left(\lambda_2 - \eta_2^{(t)}\right) - e_1^T \lambda_2. \tag{3.30}$$

Dual optimization problems can be written:

$$\min_{\lambda_1} \frac{1}{2}\lambda_1^T Q_1 \lambda_1 + \lambda_1^T q_1 - e_2^T \lambda_1. \tag{3.31}$$

$e_2 \in \mathbb{R}^{m_2}$ is the unit vector, where

$$Q_1 = H_2(H_1^T H_1)^{-1} H_2^T + \frac{k^\gamma}{C_1}, \quad q_1 = C_1 H_2 (H_1^T H_1)^{-1} H_2^T \eta_1^{(t)}.$$

Similarly,

$$\min_{\lambda_2} \frac{1}{2}\lambda_2^T Q_2 \lambda_2 - \lambda_2^T q_2 - e_1^T \lambda_2. \tag{3.32}$$

$e_1 \in \mathbb{R}^{m_1}$ is the unit vector, where

$$Q_2 = H_1(H_2^T H_2)^{-1} H_1^T + \frac{k^\gamma}{C_2}, \quad q_2 = C_2 H_1 (H_2^T H_2)^{-1} H_1^T \eta_2^{(t)}.$$

After solving (3.31) and (3.32) to obtain the optimal solutions λ_1^* and λ_2^*, we can obtain

$$\beta_1 = -(H_1^T H_1)^{-1} H_2^T (\eta_1^{(t)} + \lambda_1), \tag{3.33}$$

$$\beta_2 = (H_2^T H_2)^{-1} H_1^T (\lambda_2 - \eta_2^{(t)}). \tag{3.34}$$

Therefore, the decision function of AFTELM is given as follows:

$$f(x) = \arg\min_{t=1,2} d_t(x) = \arg\min_{t=1,2} \left|\beta_t^T h(x)\right|. \tag{3.35}$$

3.6. Squared fractional loss twin extreme learning machine (SFTELM)

To ensure a fair evaluation of the proposed AF-loss, a dedicated baseline model, termed SFTELM, is constructed for the ablation study. It is important to note that SFTELM differs from the comprehensive SF-RSTELM framework in [20], which integrates SF-loss with the capped $L_{2,p}$-norm metric and Fisher regularization. The SFTELM model is a simplified version, where the standard TELM is equipped only with the SF-loss defined in (3.2). This design allows for the isolation and precise evaluation of the improvement brought by the AF-loss over the direct SF-loss baseline.

SFTELM determines the parameters by solving optimization problems analogous to (3.12) and (3.13), with L_a replaced by L_r.

According to the process of the above operation, we give the pseudo-code of the AFTELM's process as shown in Algorithm 1.

Algorithm 1 Training AFTELM

Input: Training data $\{(x_i, y_i)\}_{i=1}^n$, $y_i \in \{1, 2\}$; Parameters: $C_1, C_2, k, \gamma, \epsilon, T_{\max}$; Activation function $G(x)$; The number of hidden nodes L.

Output: β_1^* and β_2^*;

1: Data Preprocessing: Divide into D_1, D_2, construct H_1, H_2, initialize $\beta_1^{(0)} = 0, \beta_2^{(0)} = 0$.
2: Iterative Optimization:
3: Calculate gradients $\eta_1^{(t)}, \eta_2^{(t)}$;
4: Solve dual problems to get λ_1^*, λ_2^*;
5: Update:

$$\beta_1^{(t+1)} = -(H_1^T H_1)^{-1} H_2^T (\eta_1^{(t)} + \lambda_1^*)$$

$$\beta_2^{(t+1)} = (H_2^T H_2)^{-1} H_1^T (\lambda_2^* - \eta_2^{(t)})$$

6: Check Convergence: If $\|\beta_1^{(t+1)} - \beta_1^{(t)}\| < \epsilon$ and $\|\beta_2^{(t+1)} - \beta_2^{(t)}\| < \epsilon$, stop.
7: Decision Function:

$$f(x) = \arg\min_{t\in\{1,2\}} |\beta_t^T h(x)|$$

4. Convergence analysis

Theorem 1. *Use the CCCP technique to address problem* (3.12) *and the resulting sequence* $\left\{\beta_1^{(t)}\right\}$ *converges.*

Proof. At the iteration point of step $t+1$, the following inequality holds:

$$J_{\text{vex}}\left(\beta_1^{(t)}\right) + \nabla J_{\text{cav}}\left(\beta_1^{(t)}\right)^T \beta_1^{(t)} \geq J_{\text{vex}}\left(\beta_1^{(t+1)}\right) + \nabla J_{\text{cav}}\left(\beta_1^{(t)}\right)^T \beta_1^{(t+1)}. \tag{4.1}$$

It can be written as

$$J_{\text{vex}}\left(\beta_1^{(t)}\right) - J_{\text{vex}}\left(\beta_1^{(t+1)}\right) \geq \nabla J_{\text{cav}}\left(\beta_1^{(t)}\right)^T \left(\beta_1^{(t+1)} - \beta_1^{(t)}\right). \tag{4.2}$$

Due to the concavity of $J_{\text{cav}}(\cdot)$, we have

$$\nabla J_{\text{cav}}\left(\beta_1^{(t)}\right)^T \left(\beta_1^{(t+1)} - \beta_1^{(t)}\right) \geq J_{\text{cav}}\left(\beta_1^{(t+1)}\right) - J_{\text{cav}}\left(\beta_1^{(t)}\right). \tag{4.3}$$

By combining the above inequalities, we have

$$J_{\text{vex}}\left(\beta_1^{(t)}\right) + J_{\text{cav}}\left(\beta_1^{(t)}\right) \geq J_{\text{vex}}\left(\beta_1^{(t+1)}\right) + J_{\text{cav}}\left(\beta_1^{(t+1)}\right). \tag{4.4}$$

Accordingly, the objective value of problem (3.12) decreases monotonically with each iteration and remains non-negative, thereby proving the convergence of the sequence. The proof of $\left\{\beta_2^{(t)}\right\}$ convergence is similar. □

Theorem 2. *Algorithm 1 converges to a local optimum to the problems in* (3.12) *and* (3.13).

Proof. Taking problem (3.12) as an example, the analysis for problem (3.13) follows a similar approach.
A local optimal point $(\hat{\beta}_1, \hat{\xi}_1, \hat{\lambda}_1)$ satisfies the following KKT conditions:

$$-H_2\hat{\beta}_1 + \hat{\xi}_1 = e_2, \quad \hat{\xi}_1 \geq 0, \tag{4.5}$$

$$\hat{\lambda}_1 \geq 0, \tag{4.6}$$

$$(\hat{\lambda}_1)_j \cdot (\hat{\xi}_1)_j = 0, \quad \forall j = 1, \ldots, m_2, \tag{4.7}$$

$$\nabla_{\beta_1}\mathcal{L} = 0, \quad \frac{\partial \mathcal{L}}{\partial \xi_1} = 0. \tag{4.8}$$

Expanding the stationarity conditions (4.8), we get

$$\nabla J_1(\hat{\beta}_1) - H_2^T\hat{\lambda}_1 = 0, \tag{4.9}$$

$$C\mathbf{1} - \hat{\lambda}_1 = 0, \tag{4.10}$$

where C is the penalty parameter and $\mathbf{1}$ is the vector of ones.

The conditions (4.5)–(4.10) follow directly from the standard KKT optimality conditions for the constrained optimization problem with inequality constraints $H_2\beta_1 - \xi_1 \leq e_2$, $\xi_1 \geq 0$, and penalty term $C\mathbf{1}^T\xi_1$ in the objective function. □

Theorem 3. *If $(\hat{\beta}_1, \hat{\xi}_1, \hat{\lambda}_1)$ satisfies the KKT conditions and the projected Hessian is positive definite on the critical cone, then $\hat{\beta}_1$ is a strict local minimizer.*

Proof. Taylor-expand J_1 around $\hat{\beta}_1$:

$$J_1(\hat{\beta}_1 + d) = J_1(\hat{\beta}_1) + \nabla J_1(\hat{\beta}_1)^T d + \frac{1}{2} d^T \nabla^2 J_1(\hat{\beta}_1) d + o(\|d\|^2). \tag{4.11}$$

From the stationarity condition (4.9), we have

$$\nabla J_1(\hat{\beta}_1)^T d = (\hat{\lambda}_1)^T H_2 d. \tag{4.12}$$

For a feasible direction d at $\hat{\beta}_1$, we need $H_2(\hat{\beta}_1 + d) - (\hat{\xi}_1 + \Delta\xi) \leq e_2$, where $\hat{\xi}_1 + \Delta\xi \geq 0$. This gives us the linearized constraint:

$$H_2 d - \Delta\xi \leq 0, \quad \Delta\xi \geq -\hat{\xi}_1. \tag{4.13}$$

Define the critical cone:

$$C = \{d : (H_2 d)_j \leq 0 \text{ for } j \in \mathcal{A}(\hat{\beta}_1)\}, \tag{4.14}$$

where $\mathcal{A}(\hat{\beta}_1) = \{j : (H_2\hat{\beta}_1 - \hat{\xi}_1)_j = (e_2)_j\}$ is the active set. □

By complementary slackness (4.7) and the structure of active constraints, we consider two cases:
Case 1: If $j \in \mathcal{A}(\hat{\beta}_1)$ and $(\hat{\xi}_1)_j = 0$, then $(\hat{\lambda}_1)_j > 0$ and $(H_2 d)_j \leq 0$.
Case 2: If $j \in \mathcal{A}(\hat{\beta}_1)$ and $(\hat{\xi}_1)_j > 0$, then $(\hat{\lambda}_1)_j = 0$ and the constraint is not binding for small perturbations.

Therefore, for $d \in C$:

$$(\hat{\lambda}_1)^T H_2 d \leq 0. \tag{4.15}$$

The Hessian can be decomposed as:

$$\nabla^2 J_1(\hat{\beta}_1) = H_1^T H_1 + \text{regularization terms} + \nabla^2 J_{1,\text{cav}}(\hat{\beta}_1), \tag{4.16}$$

where

- $H_1^\top H_1 > 0$ (assuming H_1 has full column rank),
- the regularization terms are positive semidefinite, and
- $\nabla^2 J_{1,\text{cav}}(\hat{\beta}_1)$ denotes the Hessian of the concave components.

For the projected Hessian to be positive definite on C, we need

$$d^T \nabla^2 J_1(\hat{\beta}_1) d > 0, \quad \forall d \in C, d \neq 0. \tag{4.17}$$

When the regularization is sufficiently strong such that the positive definite components dominate the non-convex terms on the critical cone, condition (4.16) is satisfied.

Combining (4.11), (4.12), (4.15), and (4.17):

$$\begin{aligned} J_1(\hat{\beta}_1 + d) - J_1(\hat{\beta}_1) &= (\hat{\lambda}_1)^T H_2 d + \frac{1}{2} d^T \nabla^2 J_1(\hat{\beta}_1) d + o(\|d\|^2) \\ &\geq \frac{1}{2} d^T \nabla^2 J_1(\hat{\beta}_1) d + o(\|d\|^2) > 0 \end{aligned} \tag{4.18}$$

for sufficiently small $\|d\| \neq 0$ and $d \in C$. Therefore, $\hat{\beta}_1$ is a strict local minimizer.

5. Numerical experiments

5.1. Operating environment

All data used in the experiments are obtained from the normalized dataset. The experiments were implemented in MATLAB R2024b, running on a desktop computer equipped with an Intel(R) Core(TM) i7-11700@(2.5 GHz) processor and 16 GB of memory.

5.2. Benchmark approaches

With this clarification, we compare the proposed AFTELM with the following five algorithms:

- **TBSVM** [11, 12]: Uses hinge loss and squared L_2-norm metric.
- **TELM** [17]: Uses hinge loss and squared L_2-norm metric.
- **FTELM** [21]: Introduces Fisher regularization into TELM.
- **PTELM** [18]: Projection twin extreme learning machine.
- **SFTELM** (This work): The SF-loss-only baseline proposed above, which modifies TELM by adopting the SF-loss (without additional regularization or metric modifications) to serve as a dedicated ablation baseline for AF-loss.

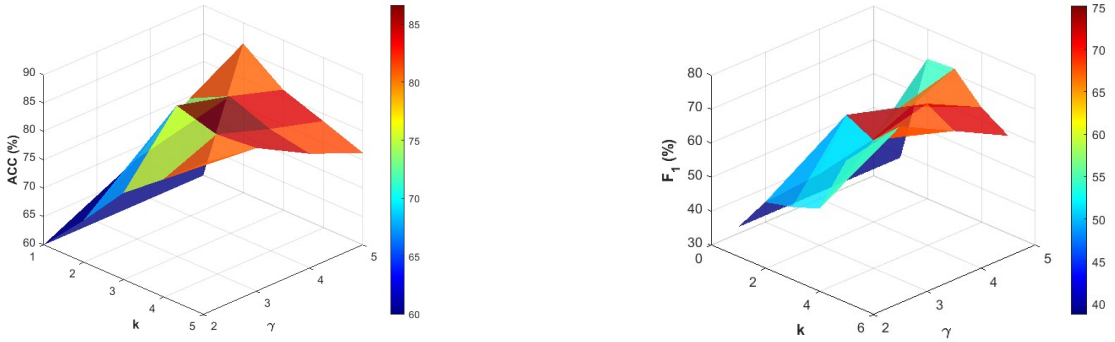
5.3. Parameter selection and evaluation criteria

How a model of algorithm reflects its best performance that depends on the choice of parameters. Therefore, we use traditional accuracy index (ACC) and the F_1-score to measure the performance of these algorithms defined as follows:

$$ACC = \frac{TP + TN}{TP + FN + TN + FP}, \tag{5.1}$$

$$F_1 = \frac{2TP}{2TP + FP + TN}. \tag{5.2}$$

Let TP, TN, FP, and FN represent the true positives, true negatives, false positives, and false negatives, respectively. Higher values of accuracy (ACC) and the F_1-score indicate better model performance. For TBSVM, the kernel parameter σ is selected from $\{10^i \mid i = -4, -3, -2, -1, 0, 1, 2, 3, 4\}$. For AFTELM, k and γ are chosen from $\{1, 2, 3, 4, 5\}$ in the interval $[1, 5]$. Results are presented in Figure 3. The regularization parameters C_1 and C_2 are selected from $\{10^i \mid i = -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5\}$, and the number of nodes is selected from $\{30, 50, 100, 150, 200, 250\}$. A stratified 10-fold cross-validation was employed to ensure statistical validity. The dataset was randomly divided into 10 parts, with 9 parts used for training and 1 part for testing in each iteration. The process was repeated 10 times, and the final result is the average value.

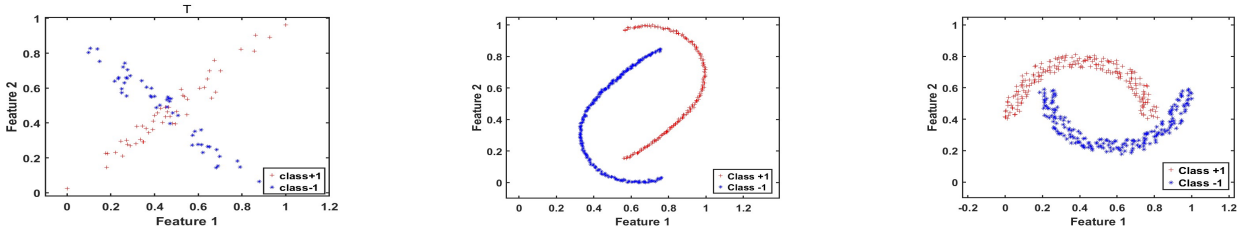


(a) Effects of k and γ on ACC (b) Effects of k and γ on F_1 score

Figure 3. Parameter value comparison.

5.4. Artificial datasets

In this subsection, the Two Moons, XOR, and Banana datasets are given. The Two Moons and Banana datasets each contain 400 samples, while the XOR dataset consists of 100 samples. Figure 4 illustrates the two-dimensional distributions of these three artificial datasets.



(a) (b) (c)

Figure 4. Three artificial datasets.

5.5. UCI datasets

To evaluate the classification performance of AFTELM relative to other related algorithms, experiments were conducted on nine datasets from the UCI repository: Balance, Australian, Vote, Sonar, Spectfsta, Pima, QSAR, WDBC, and Wholesale. These data sets are binary classification data,and the specific information is shown in Table 1.

Table 1. Characteristics of UCI datasets.

Datasets	Samples	Attributes	Datasets	Samples	Attributes
Australia	690	14	QSAR	1055	41
Sonar	208	60	Balance	576	4
Spectfsta	267	44	Pima	768	8
Wholesale	400	7	Vote	435	16
WDBC	569	30			

5.6. Image datasets

The objective of image dataset classification is to enable automated decision-making by algorithmically deriving a structured understanding of visual information. For this purpose, the CMU Facial Expression dataset is employed. To validate the model, photos of individuals with various poses are selected. These images are evaluated under three different conditions: no noise, 10% Gaussian noise, and 30% Gaussian noise. Representative examples of the noisy images are illustrated in Figure 5.

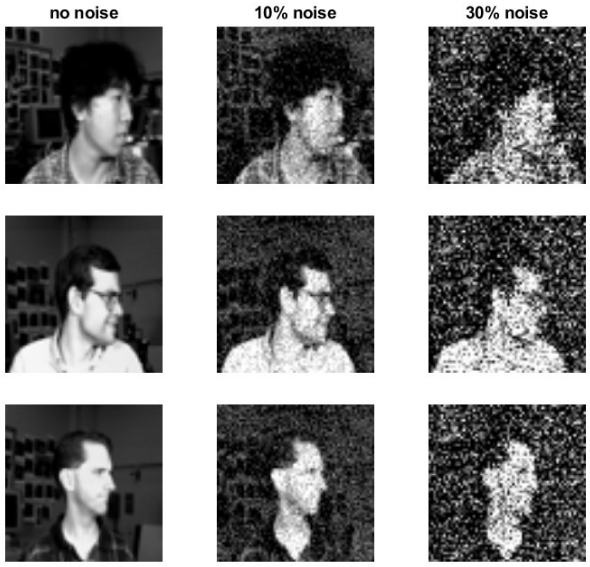


Figure 5. Image datasets.

5.7. Experimental results on the UCI datasets without noise

The accuracy and F_1-score are evaluated by testing with a training set of 9 groups and a corresponding test set. Models tested include TBSVM, TELM, FTELM, PTELM, SFTELM, and AFTELM. Each test is repeated 10 times, with the mean value used to indicate performance. To further assess classification performance, experiments are conducted on the UCI dataset. All results in Table 2 are based on optimal parameters.

Table 2. Experimental results on UCI datasets without noise. The best results are marked in bold.

Datasets	TBSVM	TELM	FTELM	PTELM	SFTELM	AFTELM
	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s) k, γ
Australia	84.75±0.04 83.31±0.07 0.510	84.97±0.05 81.02±0.08 0.599	84.97±0.05 81.16±0.07 0.574	85.92±0.04 **84.84±0.05** **0.125**	83.91±0.05 83.16±0.04 4.231	**85.93±0.05** 84.73±0.05 5.161 $k = 3, \gamma = 2$
Sonar	72.19±0.06 76.50±0.05 0.682	66.95±0.10 62.06±0.12 0.544	62.79±0.15 66.95±0.12 0.464	77.05±0.07 78.20±0.08 **0.421**	75.07±0.09 78.50±0.08 1.444	**78.40±0.08** **81.00±0.06** 1.611 $k = 3, \gamma = 2$
Spectfsta	77.25±0.06 85.94±0.04 1.274	78.44±0.11 87.05±0.07 0.855	78.92±0.09 87.80±0.06 0.983	78.03±0.07 86.54±0.05 **0.085**	77.94±0.05 86.95±0.04 7.680	**79.83±0.05** **87.89±0.04** 8.156 $k = 4, \gamma = 2$
Wholesale	88.42±0.03 91.31±0.03 1.911	88.37±0.05 90.78±0.04 1.841	88.38±0.06 90.84±0.05 1.874	88.45±0.05 91.73±0.03 **1.066**	**89.12±0.04** **92.11±0.03** 4.078	**89.12±0.04** **92.11±0.03** 4.156 $k = 1, \gamma = 1$
WDBC	96.85±0.03 97.47±0.02 0.872	96.27±0.03 97.06±0.02 0.834	96.04±0.03 96.96±0.02 0.743	**97.03±0.03** **97.61±0.02** **0.563**	94.74±0.03 95.80±0.03 4.102	95.45±0.04 96.33±0.03 4.887 $k = 4, \gamma = 2$
QSAR	86.45±0.05 89.62±0.04 0.844	85.90±0.03 89.38±0.02 0.787	85.66±0.04 89.14±0.03 0.613	86.07±0.02 89.31±0.01 **0.644**	**86.92±0.03** **90.38±0.02** 15.561	**86.92±0.03** **90.38±0.02** 17.013 $k = 1, \gamma = 1$
Balance	98.79±0.01 98.78±0.01 0.442	98.35±0.01 98.33±0.01 0.264	98.92±0.01 98.35±0.01 0.387	94.74±0.01 98.96±0.01 **0.105**	98.96±0.11 94.69±0.02 2.561	**98.96±0.01** **98.97±0.01** 3.087 $k = 1, \gamma = 1$
Pima	75.25±0.04 **82.55±0.03** 0.717	**76.10±0.04** 82.52±0.04 0.981	75.77±0.04 82.81±0.03 0.975	75.39±0.04 80.70±0.03 **0.558**	74.87±0.04 81.81±0.03 5.569	74.48±0.04 81.83±0.03 6.083 $k = 2, \gamma = 2$
Vote	94.65±0.03 93.30±0.04 **0.115**	91.31±0.03 90.02±0.04 0.251	94.79±0.02 93.01±0.03 0.274	94.88±0.04 93.51±0.04 0.250	96.05±0.04 94.84±0.05 3.061	**96.07±0.02** **94.98±0.03** 3.164 $k = 3, \gamma = 4$

5.8. Experimental results on the UCI datasets with noise

To further investigate the robustness of AFTELM, the five models are next tested on the UCI datasets incorporating noise. Gaussian noise, with levels of 15% and 25%, are added to the datasets.

Based on the performance of the nine UCI data in Table 2, most of accuracy of AFTELM is superior than TELM, FTELM, PTELM ,TBSVM and SFTELM. According to the experimental results in Table 3 and Table 4, the classification performance of AFTELM is still higher than TBSVM, TELM,

FTELM, PTELM and SFTELM in most cases. It also found that the classification performance of the model is reduced with the continuous noise improvement. In addition, the experimental results of Table 2, Table 3, and Table 4 can find that AFTELM is more robust after adding the AF-loss term.

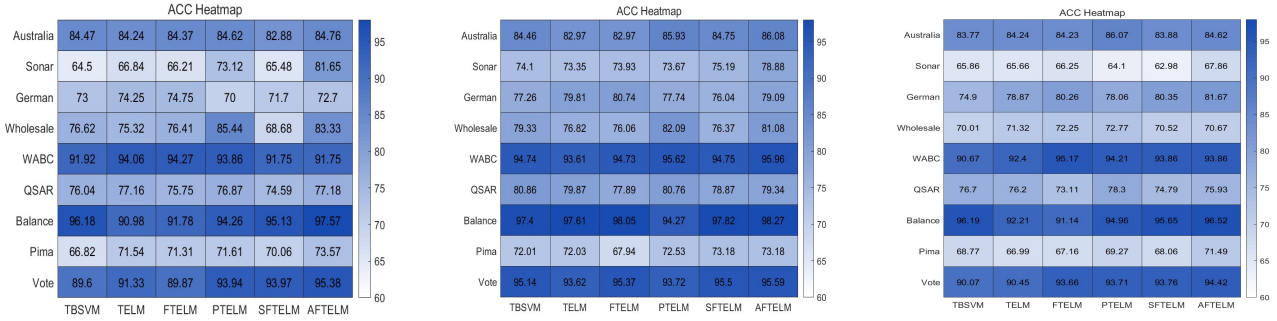
Table 3. Experimental results on UCI datasets with 15% Gaussian noise. The best results are marked in bold.

Datasets	TBSVM	TELM	FTELM	PTELM	SFTELM	AFTELM
	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s) k, γ
Australia	84.46±0.05 84.81±0.06 0.447	82.97±0.03 78.82±0.04 0.582	82.97±0.03 78.82±0.04 0.423	85.93±0.03 84.53±0.04 **0.225**	84.75±0.06 83.38±0.08 4.125	**86.08±0.02** **85.10±0.03** 5.024 $k = 2, \gamma = 4$
Sonar	74.10±0.09 77.54±0.06 0.624	73.35±0.15 76.34±0.14 **0.349**	73.93±0.16 77.53±0.15 0.465	73.67±0.08 75.25±0.09 0.381	75.19±0.12 77.58±0.09 1.451	**78.88±0.10** **81.15±0.10** 1.749 $k = 2, \gamma = 2$
Spectfsta	77.26±0.05 86.91±0.03 1.154	79.81±0.07 88.56±0.04 0.854	**80.74±0.07** **89.04±0.05** 0.563	77.74±0.12 86.37±0.09 **0.185**	76.04±0.06 85.57±0.04 7.648	79.09±0.08 87.34±0.05 8.556 $k = 3, \gamma = 2$
Wholesale	79.33±0.17 82.24±0.21 1.715	76.82±0.15 79.90±0.18 1.824	76.06±0.15 76.94±0.15 1.234	**82.09±0.07** **85.26±0.04** **1.165**	76.37±0.12 81.19±0.13 4.155	81.08±0.10 84.49±0.11 4.468 $k = 2, \gamma = 2$
WDBC	94.74±0.03 95.87±0.02 0.970	93.61±0.04 94.80±0.03 0.854	94.73±0.03 95.93±0.02 0.774	95.62±0.03 96.59±0.02 **0.629**	94.75±0.03 95.80±0.03 4.152	**95.96±0.02** **96.86±0.01** 4.870 $k = 1, \gamma = 1$
QSAR	**80.86±0.07** **85.99±0.04** 0.814	79.87±0.04 85.80±0.03 0.652	77.89±0.04 78.45±0.03 0.644	80.76±0.04 85.15±0.03 **0.542**	78.87±0.04 84.97±0.03 14.491	79.34±0.03 85.17±0.02 18.118 $k = 2, \gamma = 2$
Balance	97.40±0.02 97.40±0.02 0.541	97.61±0.02 97.57±0.02 0.466	98.05±0.02 98.01±0.02 0.458	94.27±0.02 94.37±0.02 **0.305**	97.82±0.04 **98.93±0.03** 2.461	**98.27±0.01** 98.29±0.02 3.053 $k = 3, \gamma = 4$
Pima	72.01±0.05 80.93±0.03 0.814	72.03±0.07 80.82±0.05 0.943	67.94±0.08 74.46±0.07 0.814	72.53±0.05 78.97±0.04 **0.256**	**73.18±0.04** **81.77±0.03** 5.457	**73.18±0.04** **81.77±0.03** 6.543 $k = 1, \gamma = 1$
Vote	95.14±0.03 93.98±0.04 **0.110**	93.62±0.02 91.84±0.03 0.365	95.37±0.03 93.76±0.05 0.431	93.72±0.04 91.51±0.07 0.310	95.50±0.02 93.83±0.03 3.261	**95.59±0.03** **94.36±0.04** 3.521 $k = 2, \gamma = 3$

Table 4. Experimental results on UCI datasets with 25% Gaussian noise. The best results are marked in bold.

Datasets	TBSVM	TELM	FTELM	PTELM	SFTELM	AFTELM
	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s)	ACC±S(%) F_1±S(%) Times (s) k, γ
Australia	83.77±0.06 83.29±0.05 0.540	84.24±0.03 82.65±0.03 0.654	84.23±0.03 81.09±0.05 0.633	86.07±0.04 **84.03±0.04** **0.214**	83.88±0.05 82.48±0.07 4.026	**84.62±0.04** 83.91±0.04 5.138 $k = 3, \gamma = 4$
Sonar	**65.86±0.13** 75.23±0.11 0.531	65.66±0.10 70.68±0.09 0.641	66.25±0.09 70.94±0.09 0.585	64.10±0.06 **75.45±0.08** **0.340**	62.98±0.06 71.20±0.06 1.570	67.86±0.08 73.46±0.07 1.819 $k = 5, \gamma = 2$
Spectfsta	74.90±0.05 85.28±0.04 1.123	78.87±0.07 88.27±0.03 1.073	80.26±0.06 88.57±0.03 1.212	78.06±0.11 86.28±0.07 **0.385**	80.35±0.04 88.91±0.03 7.544	**81.67±0.06** **89.23±0.03** 8.629 $k = 3, \gamma = 4$
Wholesale	70.01±0.19 70.31±0.31 1.863	71.32±0.19 75.10±0.29 1.624	72.25±0.18 75.34±0.27 1.503	**72.77±0.06** **75.63±0.04** **1.061**	70.52±0.15 70.14±0.22 4.275	70.67±0.18 70.99±0.24 4.168 $k = 5, \gamma = 2$
WDBC	90.67±0.07 93.09±0.04 0.988	92.40±0.04 96.41±0.03 0.851	**95.17±0.03** 96.26±0.03 0.733	94.21±0.02 **95.47±0.02** **0.523**	93.86±0.02 95.15±0.02 4.867	93.86±0.02 95.15±0.02 4.710 $k = 1, \gamma = 1$
QSAR	76.70±0.07 81.65±0.08 0.681	76.20±0.07 81.77±0.06 0.570	73.11±0.06 82.63±0.03 0.585	**78.30±0.04** 83.25±0.03 **0.545**	74.79±0.05 83.03±0.03 3.425	75.93±0.05 **83.55±0.03** 7.213 $k = 3, \gamma = 2$
Balance	96.19±0.03 96.16±0.03 0.361	92.21±0.04 92.17±0.04 0.564	91.14±0.04 90.98±0.03 0.654	94.96±0.02 95.05±0.02 **0.305**	95.65±0.04 95.65±0.04 2.421	**96.52±0.02** **96.57±0.02** 3.011 $k = 2, \gamma = 2$
Pima	68.77±0.06 76.83±0.08 0.710	66.99±0.08 69.66±0.14 0.871	67.16±0.08 66.93±0.14 0.644	69.27±0.04 77.04±0.03 **0.325**	68.06±0.02 78.98±0.02 5.658	**71.49±0.06** **79.97±0.04** 5.423 $k = 3, \gamma = 2$
Vote	90.07±0.07 85.44±0.14 0.245	90.45±0.03 88.29±0.04 0.261	93.66±0.03 91.72±0.04 0.282	93.71±0.05 91.49±0.08 **0.224**	93.76±0.02 91.73±0.03 3.341	**94.42±0.04** **92.67±0.05** 3.812 $k = 2, \gamma = 2$

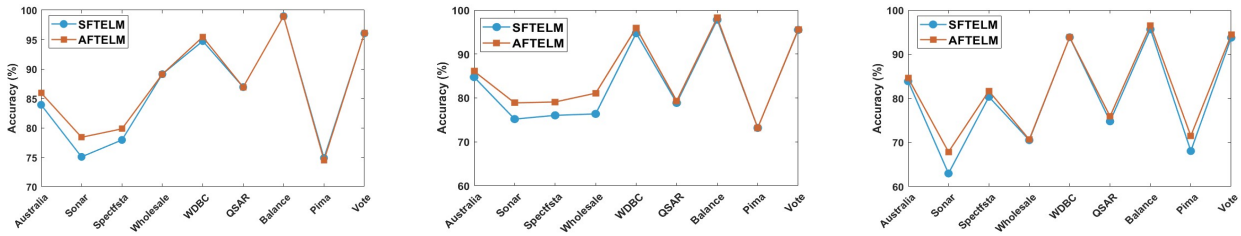
To provide a intuitive understanding of AFTELM's accuracy under varying noise levels, accuracy heatmaps by different methods are generated based on the data presented in Figure 6. From Figure 6, we can find that AFTELM classification performance is mostly better than the other five models.



(a) ACC with 0% Noise (b) ACC with 15% Noise (c) ACC with 25% Noise

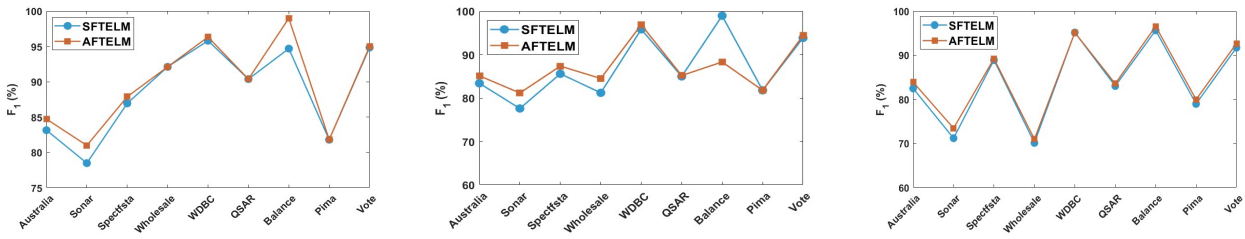
Figure 6. ACC heatmap at 0% noise, 15% noise, and 25% noise.

As shown in Figures 7 and 8, AFTELM consistently outperforms the original SFTELM, achieving improvements of 1%–5% in both accuracy and F_1-score. These results demonstrate a significant performance gain of AFTELM over SFTELM, highlighting its superior classification capability and robustness.



(a) ACC with 0% noise (b) ACC with 15% noise (c) ACC with 25% noise

Figure 7. Performance comparison of accuracy between AFTELM and SFTELM.



(a) F_1 with 0% noise (b) F_1 with 15% noise (c) F_1 with 25% noise

Figure 8. Performance comparison of F_1 between AFTELM and SFTELM.

5.9. Experimental results on artificial dataset

To simulate performance in noisy environments, Gaussian noise is added to the training data. Noise is introduced by selecting samples and perturbing their features with Gaussian noise from a normal distribution $N(0, \sigma^2)$. The noisy training data is represented as $X + \tilde{X}$, where $\tilde{X}$ is the noise matrix. Experiments on nine UCI datasets show improved classification performance and robustness of AFTELM. We also conduct experiments on artificial datasets with 20% Gaussian noise. All results are presented in Figure 9.

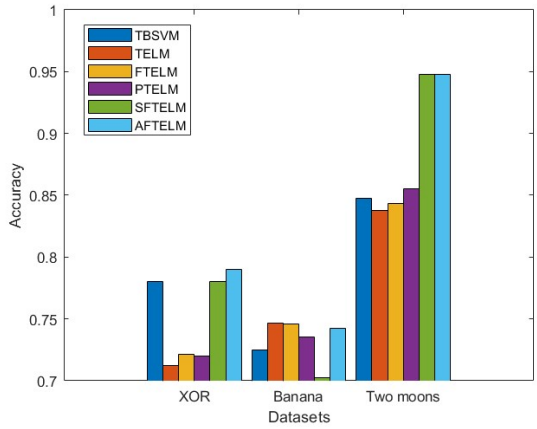


Figure 9. Accuracies of five algorithms on three artificial datasets via 20% noises factors.

As shown in Figure 9, the accuracy of the six models are evaluated on artificial datasets containing noise. For instance, in the XOR dataset, the accuracies of the models are as follows: TBSVM 78.00%, TELM 71.25%, FTELM 72.25%, PTELM 72.00%, SFTELM 78.00%, and AFTELM 79.00%. The experimental results demonstrate that AFTELM maintains high accuracy even in the presence of noise, further highlighting its robustness against the negative effects of AF-loss on noisy data.

5.10. Experimental results on the image datasets

All images are first partitioned and processed such that each image has a pixel size of 64×64. These 64×64 images are then flattened into 4096 dimensional feature vectors. To ensure consistent feature scaling, each feature vector undergoes standardization by subtracting the mean and dividing by the standard deviation. Through this process, all relevant information from the portrait images is extracted via feature representation. The classification performance of various models on this image dataset is presented in Table 5.

Table 5. Experimental results of CMU facial expression datasets. The best results are marked in bold.

Noise content	TBSVM	TELM	FTELM	PTELM	SFTELM	AFTELM
	ACC(%)	ACC(%)	ACC(%)	ACC(%)	ACC(%)	ACC(%)
0	**100**	97.13	98.17	99.17	98.12	99.37
10%	95.42	95.14	97.40	96.96	95.40	**97.93**
30%	60.63	86.34	89.92	86.48	85.64	**92.92**

It can be observed that the AFTELM model consistently achieves higher classification accuracy compared to the other models. As noise levels increase, the accuracy of the other models degrades significantly, whereas AFTELM maintains relatively stable performance. This highlights the superior robustness of the AFTELM model in noisy environments. It is demonstrated by the experimental results that high accuracy is maintained by AFTELM even in the presence of noise, with its robustness against the negative effects of AF-loss on noisy data further highlighted.

6. Convergence curve

In this section, the convergence of AFTELM is experimentally demonstrated. The convergence curve of the algorithm is presented in Figure 10. On this curve, the x-axis denotes the iteration count, and the y-axis indicates the value of the objective function. The curve illustrates the variation of the objective function value in relation to the iteration count.

(a) Vote (b) WDBC

Figure 10. Convergence graph of AFTELM's objective function values increasing with the number of iterations on two datasets.

Figure 10 shows that, in all two classification tasks, the objective function rapidly decreases within the first 10 iterations and stabilizes thereafter, reflecting the efficient optimization performance of the AFTELM model. Moreover, the convergence curves for all classification tasks eventually level off, indicating no significant fluctuation during the model optimization process.

7. Statistical significance analysis

A rigorous non-parametric statistical analysis was conducted to assess the proposed AFTELM model against five benchmark algorithms (TBSVM, TELM, FTELM, PTELM, and SFTELM). The Friedman test was employed as the primary statistical tool. Algorithms were ranked according to their performance metrics on each dataset. Rank 1 was assigned to the best-performing algorithm. The Friedman statistic χ_F^2 was computed as:

$$\chi_F^2 = \frac{12n}{k(k+1)}\left(\sum_{i=1}^{k} R_i^2 - \frac{k(k+1)^2}{4}\right), \tag{7.1}$$

where R_i denotes the average rank of the i-th classifier, n represents the number of datasets, and k denotes the number of classifiers. The F-distributed statistic was utilized to improve test sensitivity:

$$F_F = \frac{(n-1)\chi_F^2}{n(k-1)-\chi_F^2}, \tag{7.2}$$

which follows an F-distribution with $k-1$ and $(k-1)(n-1)$ degrees of freedom. For $k = 6$ classifiers and $n = 9$ datasets, the critical value was $F(5, 40) = 2.449$ at the 0.05 significance level.

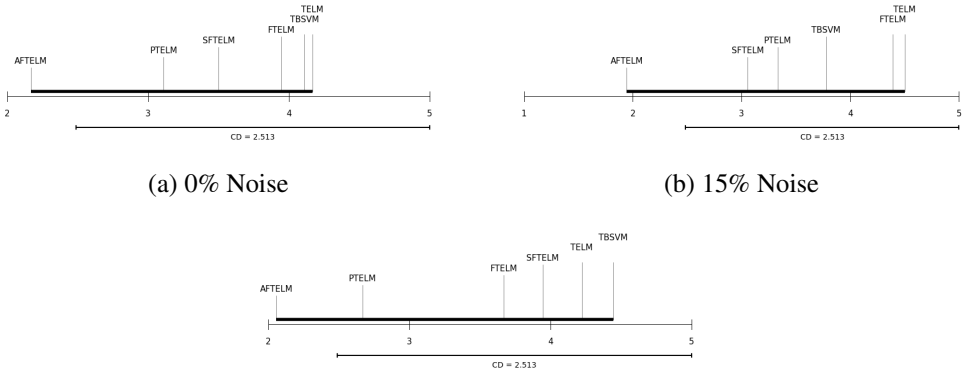
The statistical analysis was conducted across three experimental settings: noise-free conditions, 15% Gaussian noise, and 25% Gaussian noise. For the nine UCI benchmark datasets without noise,

the Friedman test yielded $\chi_F^2 = 9.62$ and $F_F = 2.17$. Under 15% Gaussian noise, $\chi_F^2 = 14.02$ and $F_F = 3.17$ were obtained, with the null hypothesis decisively rejected ($p < 0.05$). Under 25% Gaussian noise, $\chi_F^2 = 12.58$ and $F_F = 2.77$ confirmed significant performance differences among the competing algorithms ($p < 0.05$). Under noise-free conditions, AFTELM achieved the best average rank of 2.17. PTELM followed with an average rank of 3.11, while SFTELM ranked 3.50. FTELM (3.94), TBSVM (4.11), and TELM (4.17) showed weaker performance. Under 15% Gaussian noise, AFTELM maintained excellent performance with an average rank of 1.94. SFTELM ranked second at 3.06, followed by PTELM (3.33) and TBSVM (3.78). FTELM (4.39) and TELM (4.50) showed degraded performance. Under 25% Gaussian noise, AFTELM continued to lead with an average rank of 2.06. PTELM ranked second at 2.67, followed by FTELM (3.67) and SFTELM (3.94). TELM (4.22) and TBSVM (4.44) were substantially outperformed. Across all 27 experimental scenarios (9 datasets × 3 noise levels), AFTELM achieved an overall average rank of 2.06. Consistent top-tier performance was maintained, substantially outperforming PTELM (3.04), SFTELM (3.50), FTELM (4.00), TBSVM (4.11), and TELM (4.30).

Critical difference (CD) diagrams were constructed using the Nemenyi post-hoc test (Fig. 11). These diagrams visually represent the average ranks of all algorithms. Horizontal bars connect algorithms whose performance differences are not statistically significant at $\alpha = 0.05$. The critical difference threshold was calculated as:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6n}}, \tag{7.3}$$

where $q_\alpha = 2.850$ is the critical value from the studentized range distribution for $k = 6$ algorithms. For $n = 9$ datasets, $CD = 2.514$ was obtained. For the overall analysis across 27 scenarios, $CD = 1.451$ was obtained.



(a) 0% Noise

(b) 15% Noise

(c) 25% Noise

Figure 11. Critical Difference diagrams showing Friedman-Nemenyi test results across different noise levels.

As shown in Figure 11(a), under noise-free conditions, AFTELM (rank 2.17) achieved the best performance. PTELM (rank 3.11) and SFTELM (rank 3.50) formed a competitive middle group. FTELM (rank 3.94), TBSVM (rank 4.11), and TELM (rank 4.17) exhibited weaker performance. The thick horizontal bar indicates no significant difference between algorithms within the CD threshold. Figure 11(b) reveals that under 15% noise, AFTELM (rank 1.94) demonstrated clear leadership with

substantial margin over other methods. SFTELM (rank 3.06) and PTELM (rank 3.33) maintained moderate performance, while FTELM (rank 4.39) and TELM (rank 4.50) showed notable degradation. Under 25% noise (Figure 11(c)), AFTELM achieved the best rank (2.06), followed by PTELM (2.67) and FTELM (3.67). SFTELM (3.94), TELM (4.22), and TBSVM (4.44) exhibited considerable performance degradation under high noise conditions. AFTELM maintained its dominant position within the top statistical tier despite increasing noise intensity.

Pairwise comparisons were conducted using the Nemenyi post-hoc test. Under the CD criterion of 2.514, any two algorithms differing by more than this threshold in average rank were considered statistically significantly different. Across all noise conditions, AFTELM consistently achieved the best average rank ranging from 1.94 to 2.17. Under 15% noise, AFTELM's rank difference from the second-best algorithm (SFTELM) was 1.12. Under 25% noise, the rank difference from PTELM was 0.61. In the overall analysis across 27 scenarios with $CD = 1.451$, AFTELM (rank 2.06) demonstrated statistically significant superiority over FTELM (rank 4.00), TBSVM (rank 4.11), and TELM (rank 4.30), as these rank differences exceeded the CD threshold.

The comprehensive statistical evaluation demonstrates that AFTELM maintained consistent top-tier performance with average ranks between 1.94 and 2.17 across all noise conditions. Exceptional resilience to data corruption was exhibited through adaptive fuzzy membership weighting. Statistically significant overall differences were confirmed by the Friedman test under noisy conditions. Strong performance across UCI benchmarks validated AFTELM's robustness and generalization capability.

8. Conclusions and future directions

Based on the binary classification problem, this paper proposes a novel TELM-based model named AFTELM. The proposed AFTELM incorporates a bounded, smooth, and non-convex adaptive fractional loss, which is developed by modifying the fractional loss function from SF-RSTELM [20]. Specifically, by adaptively adjusting two key parameters in the original fractional loss, the model achieves enhanced expressive capability and improved robustness. To efficiently handle this modified loss function, the concave-convex procedure CCCP is employed to iteratively optimize the model variables, thereby improving classification performance. Theoretical analysis demonstrates that the optimization process of AFTELM is convergent. We evaluate the model on both UCI benchmarks and a custom-constructed dataset. Experimental results show that AFTELM outperforms TBSVM, TELM, FTELM, PTELM, and SFTELM in terms of classification accuracy. Furthermore, we extend AFTELM to semi-supervised learning scenarios through additional experiments, aiming to explore its broader applicability.

Author contributions

Jun Ma, Guolin Yu: Writing-original draft, writing-reviewing & editing, methodology, software, data curation, supervision, validation, project administration, funding acquisition; Xiang Jin: Writing-original draft, writing-reviewing & editing, methodology, software, data curation, supervision. All authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

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

Acknowledgments

This research was supported in part by the National Natural Science Foundation of China (No.62366001, No.12361062), in part by the Natural Science Foundation of Ningxia Province (No.2024AAC05055, No.2023AAC02053), in part by the Fundamental Research Funds for the Central Universities of North Minzu University (No.2023ZRLG01), and in part by the Postgraduate Innovation Project of North Minzu University (YCX24261).

Conflict of interest

The authors declare no conflicts of interest.

References

1. Y. Liu, A nonfunctional data transformation approach via kurtosis adjustment and its application to SVM classification, *J. Phys.: Conf. Ser.*, **2294** (2022), 012024. https://doi.org/10.1088/1742-6596/2294/1/012024
2. X. J. Peng, D. Xu, Twin support vector hypersphere (TSVH) classifier for pattern recognition, *Neural Comput. Applic.*, **24** (2014), 1207–1220. https://doi.org/10.1007/s00521-012-1306-6
3. M. A. Chandra, S. S. Bedi, Survey on SVM and their application in image classification, *Int. J. Inf. Technol.*, **13** (2021), 1–11. https://doi.org/10.1007/s41870-017-0080-1
4. P. Zitha, B. A. Thango, On the study of induction motor fault identification using support vector machine algorithms, *2023 31st Southern African Universities Power Engineering Conference (SAUPEC)*, Johannesburg, South Africa, 2023, 1–5. https://doi.org/10.1109/SAUPEC57889.2023.10057836
5. Z. Q. Qi, Y. J. Tian, Y. Shi, Robust twin support vector machine for pattern classification, *Pattern Recogn.*, **46** (2013), 305–316. https://doi.org/10.1016/j.patcog.2012.06.019
6. T. Chakraborty, U. Kumar, Loss function, In: *Encyclopedia of mathematical geosciences*, Cham: Springer, 2022, 1–6. https://doi.org/10.1007/978-3-030-26050-7_187-1
7. W. Panup, W. Ratipapongton, R. Wangkeeree, A novel twin support vector machine with generalized pinball loss function for pattern classification, *Symmetry*, **14** (2022), 289. https://doi.org/10.3390/sym14020289
8. M. Tanveer, A. Sharma, P. N. Suganthan, General twin support vector machine with pinball loss function, *Inform. Sciences*, **494** (2019), 311–327. https://doi.org/10.1016/j.ins.2019.04.032
9. J. M. Shen, J. Ma, Sparse twin extreme learning machine with ε-insensitive zone pinball loss, *IEEE Access*, **7** (2019), 112067–112078. https://doi.org/10.1109/ACCESS.2019.2935008

10. T. Jiang, B. Wei, G. L. Yu, J. Ma, Generalized adaptive huber loss driven robust twin support vector machine learning framework for pattern classification, *Neural Process. Lett.*, **57** (2025), 63. https://doi.org/10.1007/s11063-025-11783-5

11. Y. H. Shao, C. H. Zhang, X. B. Wang, N. Y. Deng, Improvements on twin support vector machines, *IEEE T. Neural Networ.*, **22** (2011), 962–968. https://doi.org/10.1109/TNN.2011.2130540

12. B. Parashjyoti, D. Gupta, Robust twin bounded support vector machines for outliers and imbalanced data, *Appl. Intell.*, **51** (2021), 5314–5343. https://doi.org/10.1007/s10489-020-01847-5

13. H. Y. Wang, G. L. Yu, J. Ma, Capped L2,p-norm metric based on robust twin support vector machine with welsch loss, *Symmetry*, **15** (2023), 1076. https://doi.org/10.3390/sym15051076

14. J. Wang, S. Y. Lu, S. H. Wang, Y. D. Zhang, A review on extreme learning machine, *Multimed. Tools Appl.*, **81** (2022), 41611–41660. https://doi.org/10.1007/s11042-021-11007-7

15. C. W. Deng, G. B. Huang, J. Xu, J. X. Tang, Extreme learning machines: New trends and applications, *Sci. China Inf. Sci.*, **58** (2015), 1–16. https://doi.org/10.1007/s11432-014-5269-3

16. G. A. Barreto, A. L. B. P. Barros, A robust extreme learning machine for pattern classification with outliers, *Neurocomputing*, **176** (2016), 3–13. https://doi.org/10.1016/j.neucom.2014.10.095

17. Y. H. Wan, S. J. Song, G. Huang, S. Li, Twin extreme learning machines for pattern classification, *Neurocomputing*, **260** (2017), 235–244. https://doi.org/10.1016/j.neucom.2017.04.036

18. Y. Yang, Z. X. Xue, J. Ma, X. Chang, Robust projection twin extreme learning machines with capped L1-norm distance metric, *Neurocomputing*, **517** (2023), 229–242. https://doi.org/10.1016/j.neucom.2022.09.156

19. X. B. Chen, J. Yang, Q. L. Ye, J. Liang, Recursive projection twin support vector machine via within-class variance minimization, *Pattern Recogn.*, **44** (2011), 2643–2655. https://doi.org/10.1016/j.patcog.2011.03.001

20. Z. X. Xue, Y. Wang, Y. W. Ren, X. Y. Zhang, The robust supervised learning framework: Harmonious integration of twin extreme learning machine, squared fractional loss, capped L2,p-norm metric, and Fisher regularization, *Symmetry*, **16** (2024), 1230. https://doi.org/10.3390/sym16091230

21. Z. X. Xue, L. C. Cai, Robust Fisher-regularized twin extreme learning machine with capped L1-norm for classification, *Axioms*, **12** (2023), 717. https://doi.org/10.3390/axioms12070717

22. C. Yuan, L. M. 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

23. S. Li, S. J. Song, Y. H. Wan, Laplacian twin extreme learning machine for semi-supervised classification, *Neurocomputing*, **321** (2018), 17–27. https://doi.org/10.1016/j.neucom.2018.08.028

24. J. Ma, G. L. Yu, Lagrangian regularized twin extreme learning machine for supervised and semi-supervised classification, *Symmetry*, **14** (2022), 1186. https://doi.org/10.3390/sym14061186

25. Z. Q. Qi, Y. J. Tian, Y. Shi, Twin support vector machine with universum data, *Neural Networks*, **36** (2012), 112–119. https://doi.org/10.1016/j.neunet.2012.09.004

26. S. Akhter, J. H. Miller, BPAGS: A web application for bacteriocin prediction via feature evaluation using alternating decision tree, genetic algorithm, and linear support vector classifier, *Front. Bioinform.*, **3** (2024), 1284705. https://doi.org/10.3389/fbinf.2023.1284705

27. A. L. Yuille, A. Rangarajan, The concave-convex procedure, *Neural Comput.*, **15** (2003), 915–936. https://doi.org/10.1162/08997660360581958

AIMS Press