

Article

A Sparse $\mathcal{L}_\infty$ -Norm Regularized Least Squares Support Vector Regression

Xiaoyong Liu ^{*}, Dong Li and Chengbin Zeng 

School of Automation Engineering, Moutai Institute, Renhuai 564507, China

* Correspondence: liuxy204@163.com

Abstract

Although Least Squares Support Vector Regression (LSSVR) reduces the hyperparameter space to two, it sacrifices sparsity, causing all training samples to become support vectors and increasing storage costs. In contrast, standard Support Vector Regression (SVR) preserves sparsity but requires tuning three highly coupled hyperparameters, leading to higher computational burden. To address these limitations, this paper proposes a sparse $\mathcal{L}_\infty$ -norm regularized least squares SVR framework that incorporates the infinity norm of approximation errors into both the objective function and inequality constraints. The resulting optimization problem minimizes model complexity while controlling the maximum prediction deviation through a single slack variable, thereby transforming the conventional three-hyperparameter SVR tuning task into a two-parameter problem involving only the regularization coefficient and kernel width. This formulation restores sparsity by enabling a compact support vector set, while preserving the stability and convexity advantages of LSSVR. Experiments on both static and dynamic datasets demonstrate that the proposed method consistently achieves higher predictive accuracy and improved robustness compared with standard SVR and LSSVR. These results indicate that the proposed $\mathcal{L}_\infty$ -norm regularized framework offers a mathematically principled and computationally efficient alternative for sparse, robust, and scalable regression modeling.

Keywords: sparsity; Least Squares Support Vector Regression (LSSVR); $\mathcal{L}_\infty$ -norm regularization; robust regression

1. Introduction

Guaranteeing the sparsity, modeling accuracy, and efficient optimization of a reduced number of hyperparameters in constructing robust regression models is one of the critical unresolved issues in the realm of machine learning. Practitioners in fields ranging from theoretical research to engineering applications are hesitant to implement machine learning solutions, as the process of constructing predictive regression models with a better generalization performance is not transparent. Support Vector Regression (SVR) [1,2] is a well-known type of non-parametric supervised learning model that integrates the control of model structural risk into machine learning. The risk minimization theory allows for a balance between empirical risk and structural risk, and therefore, SVR surmounts a significant shortcoming of other, nonlinear modeling methods, namely over-fitting [3]. Shortly after its introduction, SVR has been extended in numerous applications, ranging from fault detection [4–6], pattern recognition [7,8] and predictive control [9] to brain–computer interfaces [10].



Academic Editor: Frank Werner

Received: 25 January 2026

Revised: 15 February 2026

Accepted: 16 February 2026

Published: 18 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons](https://creativecommons.org/licenses/by/4.0/)[Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

SVR is viewed as one of the most effective and commonly applied techniques in predictive modeling, with improvements in its predictive performance and generalization capacity significantly relying on the adjustment of its hyperparameters. Although it reveals the generalization capability in regression by introducing the ϵ -insensitive margin, providing new insights into controlling the structural complexity of models, the optimization process implementing the selection of three hyperparameters, which includes regularization factor γ , kernel parameter σ , and ϵ , makes the regression modeling method more complex [11–17]. As mentioned in literature [18], the training of SVR is time-consuming on large-scale datasets, especially when it comes to hyperparameter optimization. It is currently reported that finding the optimal hyperparameters in machine learning models efficiently without sacrificing prediction accuracy poses major hurdles [19]. Prevailing techniques like Grid-Search and Random-Search with cross-validation for optimizing the hyperparameters require considerable time in large-scale search spaces, and they do not provide a guarantee of excellent predictive results. Parameters can be categorized into two groups: learnable parameters and hyperparameters. Learnable parameters are obtained from training data using optimization techniques such as quadratic programming (QP), linear programming (LP), or gradient descent. Hyperparameters are not obtained from the data; instead, they must be chosen before starting the learning process, the optimization of which involves identifying the best combination of hyperparameters that either reduces prediction error or enhances prediction accuracy. As a result, hyperparameter optimization plays a vital role in developing powerful and efficient machine learning models.

Various methods exist for hyperparameter optimization in machine learning models, including grid search, probabilistic model-based optimization, random search [20,21], gradient descent algorithms, and nature-inspired optimization methods. Grid search methods incorporating performance metric [22,23] or a weighted error function [19] aim to find the optimal combination of hyperparameters through iterative updates. From the standpoint of Bayesian methods, probabilistic model-based optimization [24,25] employs a Bayesian model to assess the performance of various hyperparameter configurations and chooses the most advantageous options for evaluation. Gradient descent-based approaches are a widely used class of conventional optimization techniques for adjusting real-valued hyperparameters by computing their gradients. Nature-inspired optimization is another method that employs evolution-based techniques, among them genetic algorithms [26] or particle swarm optimization [27], to explore the range of potential hyperparameters. Additionally, the literature [28] proposed an effective warm-start technique as a practical tool for parameter selection in linear SVR, and the approach has been available for public use. The aforementioned optimization methods aim to find the optimal selection to guarantee a better generalization ability by exhaustively exploring all possible combinations in the hyperparameter space. For instance, there are three hyperparameters in SVR, each with 100 options within its respective space, resulting in a total of 10^6 combinations. However, if the number of hyperparameters is reduced to two, the total size of possible combinations will decrease from one million to ten thousand. Thus, the hyperparameter optimization process for SVR will be greatly simplified and improved in terms of both time and spatial scale.

Significant efforts have been dedicated to exploring solutions for the redundancy during the hyperparameter optimization of SVR. In this regard, LSSVR, as a variant of SVR and first introduced in [29,30], has been successfully explored and emerged as a prominent technique in machine learning, particularly for classification and regression problems. In the LSSVR framework—rather than addressing a quadratic programming problem with inequality constraints as in the conventional SVR—equality constraints and an L_2 -loss function are employed. This results in an optimization problem whose dual solution is

found by solving a system of linear equations. More importantly, compared to SVR, LSSVR requires the optimization of only two hyperparameters, namely the kernel width σ and the regularization factor γ , significantly improving the optimization process. As a result, LSSVR has seen widespread application and research in recent years, with over 15,000 citations. For instance, it has been successfully applied in short-term traffic flow prediction [31], optimal control [32], interval-censored data [33] and model identification [34]. Unfortunately, since the emergence of LSSVR, researchers have found that the method lacks model sparsity, meaning that all or most of the samples involved in model training contribute to the regression model, which can easily lead to poor generalization performance. Addressing model sparsity has become a current research hotspot. An early sparse approximation method [35,36] was proposed by Johan Suykens, which can perform pruning based on the physical interpretation of the magnitude of the contribution of ordered support values. Follow-up works have sought to address these challenges by introducing an adaptive pruning algorithm [37,38], Nyström approximation method along with prototype vectors (PVs) [39], gradient descent technique [40] and non-iterative sparse scheme with global representative point selection [41].

However, these methods remain fundamentally limited in achieving true sparsity. Most approaches, such as pruning or global representative point selection, are inherently a posteriori approximations or filtering processes that artificially induce sparsity by discarding samples from the dense solution, inevitably leading to information loss and performance trade-offs. Moreover, variants employing re-weighted or ϵ -insensitive loss functions, while introducing sparsity-inducing mechanisms, still rely on heuristic thresholds or complex regularization without fundamentally altering the underlying structure of the least squares problem, resulting in suboptimal, incomplete, or heavily parameter-dependent sparse solutions. The so-called model sparsity, as described in SVR, refers to the fact that when the value of the Lagrange multiplier is relatively small (e.g., its absolute value is less than 10^{-8}), the corresponding training samples are not support vectors (SVs) and can be ignored. From the perspective of practical applications or current software tools, the core ideas of traditional SVR and LSSVR still dominate the mainstream, including the widely used MATLAB 2025a software. To the best of our knowledge, the following issues remain to be investigated: (1) The optimization of the three hyperparameter combinations in SVR is complex. (2) The introduction of slack variables within SVR results in additional inequality constraints, making the parameter estimation more challenging. (3) The expected sparsity of LSSVR has essentially not been addressed, and despite some proposed improvements, they still come at the cost of model accuracy.

It is worth noting that the use of the ℓ_∞ -norm, min–max formulations, and Chebyshev-type approximation has been extensively studied in the optimization and robust regression literature. Therefore, the contribution of this work does not aim to introduce a fundamentally new norm-based regression theory. Instead, this study focuses on embedding an ℓ_∞ -norm constraint into the SVR framework, with particular attention to kernel-based modeling, sparsity, and computational efficiency.

Unlike classical Chebyshev regression or generic ℓ_∞ -norm minimization approaches, the proposed LI-SVR is derived within the primal–dual structure of SVR and LSSVR. By reformulating the approximation error using a single global variable λ , the resulting optimization problem remains convex and requires only two hyperparameters, namely the regularization parameter and kernel width. Moreover, the number of inequality constraints is reduced to $2n$, which is half of that required by standard ϵ -SVR formulations.

More importantly, the proposed formulation naturally restores model sparsity through the Karush–Kuhn–Tucker (KKT) conditions. Only samples whose prediction errors attain the maximum bound actively constrain the solution and become support vectors, while the

remaining samples are strictly inactive. This sparsity mechanism differs fundamentally from post hoc pruning strategies or explicit sparsity-inducing regularizers, and leads to a compact kernel regression model without sacrificing convexity or introducing additional tuning parameters.

From this perspective, the proposed LI-SVR should be viewed as a structurally simplified and sparsity-enhanced kernel regression framework inspired by ℓ_∞ -norm optimization, rather than as a novel norm definition or a replacement of existing robust regression theories.

Inspired by the core ideas of classical SVR and LSSVR, the main contributions of this work are as follows.

- By integrating the concept of infinite norm into our research, we investigate the problem of lacking sparsity of LSSVR and optimizing hyperparameters of ϵ in SVR, exploring a more efficient and generalizable regression prediction method that ensures both model sparsity and modeling accuracy;
- We propose a novel minimum infinite norm Support Vector Regression (LI-SVR) method, distinct from classical SVR and LSSVR, to explore the optimal balance between the complex hyperparameter optimization process and sparsity, overcoming the limitations of traditional regression modeling methods;
- We demonstrate the sparsity of the proposed method and explain why it does not require optimization of the epsilon-insensitive parameter of ϵ in SVR. Furthermore, we find the maximum approximation error, which is adaptively computed using the proposed method to reflect model accuracy, thereby avoiding the complicated optimization of ϵ .

The rest of this article is structured as follows. Section 2 introduces the theory underlying the algorithms, while the proposed LI-SVR is detailed in Section 3. Section 4 presents the experimental analysis of the results. Finally, the conclusions would be gathered in Section 5.

2. A Concise Overview of Classical SVR and LSSVR

2.1. SVR

As our method in this paper is inspired by SVR and LSSVR, a concise summary of their principles is provided. It can be asserted that SVR developed by Vladimir Vapnik with colleagues [1] offers notable and distinct advantages in small sample learning and is among the most extensively researched models, founded on the statistical learning principles of VC theory. Given a training set with N input–output samples, $\mathbf{x}_k \in \mathbb{R}^d$ and $y_k \in \mathbb{R}$,

$$(\mathbf{x}_k, y_k), k = 1, 2, \dots, N \quad (1)$$

SVR aims to determine a nonlinear regression function,

$$f_{SVR}(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}_k) + b \quad (2)$$

where $f_{SVR}(\mathbf{x})$ can accurately simulate all training data. Since linear regression falls under the umbrella of nonlinear modeling techniques, the following discussion will shift to nonlinear SVR, where its solution commences with the incorporation of high-dimensional feature mapping $\phi(\cdot)$ to formulate the subsequent optimization problem [2],

$$\min : J(\mathbf{w}, b) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + \gamma \sum_{k=1}^N (\xi_k + \xi_k^*) \quad (3)$$

$$s.t. \begin{cases} y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b \leq \epsilon + \xi_k, \\ -y_k + \mathbf{w}^T \phi(\mathbf{x}_k) + b \leq \epsilon + \xi_k^*, \\ \xi_k, \xi_k^* \geq 0, k = 1, 2, \dots, N, \end{cases} \quad (4)$$

where the constant γ regulates the balance between smoothness of f_{SVR} and the extent to which deviations exceeding ϵ are permissible, and the slack variables ζ_k, ζ_k^* are employed as a means to address potentially unattainable constraints within the optimization framework, ensuring feasibility is maintained.

The essence lies in crafting a Lagrange function L , fusing the objective function (3) with its associated constraints (4) through the introduction of a dual variable set. Importantly, this function possesses a saddle point, relative to both primal and dual variables, precisely at the optimal solution, as elaborated below,

$$L = \frac{1}{2} \mathbf{w}^T \mathbf{w} + \gamma \sum_{k=1}^N (\zeta_k + \zeta_k^*) - \sum_{k=1}^N (\eta_k \zeta_k + \eta_k^* \zeta_k^*) + \sum_{k=1}^n \alpha_k (y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b - \epsilon - \zeta_k) + \sum_{k=1}^n \alpha_k^* (-y_k + \mathbf{w}^T \phi(\mathbf{x}_k) + b - \epsilon - \zeta_k^*) \quad (5)$$

where $\alpha_k, \alpha_k^*, \eta_k, \eta_k^*$ are Lagrange multipliers, all of which are greater than or equal to zero. By eliminating the primal variables w, b, ζ_k, ζ_k^* based on the saddle point conditions, we can compute $\mathbf{w}$ as,

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{k=1}^n (\alpha_k - \alpha_k^*) \phi(\mathbf{x}_k) \quad (6)$$

and the corresponding dual optimization problem is obtained,

$$\begin{aligned} \min : \quad & J(\alpha_k, \alpha_k^*) = - \sum_{k=1}^N (\alpha_k - \alpha_k^*) y(k) + \epsilon \sum_{k=1}^N (\alpha_k + \alpha_k^*) \\ & + \frac{1}{2} \sum_{j=1}^N \sum_{k=1}^N (\alpha_j - \alpha_j^*) (\alpha_k - \alpha_k^*) \phi^T(\mathbf{x}_j) \phi(\mathbf{x}_k) \\ \text{s.t.} \quad & \sum_{k=1}^N (\alpha_k - \alpha_k^*) = 0, \quad 0 \leq \alpha_k, \alpha_k^* \leq \gamma \end{aligned} \quad (7)$$

Based on the Karush–Kuhn–Tucker (KKT) conditions [2], it can be inferred that $\alpha_k \alpha_k^* = 0$. Additionally, it has been proven that the Lagrange multiplier is non-zero only when the condition $f(x_k) - y_k \geq \epsilon$ is satisfied, meaning that the k -th sample lies precisely on or outside the ϵ -tube and is referred to as a support vector (SV). To reiterate, the KKT conditions are met solely when the Lagrange multipliers corresponding to all samples that lie within the ϵ -tube (i.e., $f(x_k) - y_k \leq \epsilon$) are exactly zero, and this is why SVR exhibits sparsity.

2.2. LSSVR

LSSVR is based on a standard SVR but substitutes the ϵ -insensitive loss function with a least squares. It transforms the inequality constraints of SVR into an equality one, thereby converting the quadratic programming problem into a linear equation system. Similarly, LSSVR also attempts to find a better regression function $f(\mathbf{x})$ to approximate all training data. By introducing a regularization factor γ to fine-tune the degree of optimality between model complexity and error e_k , it yields the following optimization [29],

$$\min : J(\mathbf{w}, b, \mathbf{e}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + \frac{\gamma}{2} \sum_{k=1}^n e_k^2 \quad (8)$$

$$\text{s.t.} \quad y_k = \mathbf{w}^T \phi(\mathbf{x}_k) + b + e_k. \quad (9)$$

To solve this optimization, the Lagrangian function is constructed using Lagrange multipliers α_k ,

$$\begin{aligned} L(\mathbf{w}, b, \mathbf{e}, \alpha) = & \frac{1}{2} \mathbf{w}^T \mathbf{w} + \frac{\gamma}{2} \sum_{k=1}^n e_k^2 \\ & + \sum_{k=1}^n \alpha_k (y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b - e_k) \end{aligned} \quad (10)$$

Taking partial derivatives of the Lagrangian function with respect to $\mathbf{w}, b, \mathbf{e}, \alpha$ respectively, and setting them to zero, it gets,

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{k=1}^n \alpha_k \phi(\mathbf{x}_k) \quad (11)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{k=1}^n \alpha_k = 0 \quad (12)$$

$$\frac{\partial L}{\partial e_k} = 0 \Rightarrow \alpha_k = \gamma e_k \quad (13)$$

$$\frac{\partial L}{\partial \alpha_k} = 0 \Rightarrow y_k = \mathbf{w}^T \phi(\mathbf{x}_k) + b + e_k \quad (14)$$

Substituting the expression for $\mathbf{w}$ into the constraint (9) and introducing a kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$, Formula (14) can be rewritten as,

$$y_k = \sum_{j=1}^n \alpha_j K(\mathbf{x}_j, \mathbf{x}_k) + b + e_k, \quad k = 1, 2, \dots, n \quad (15)$$

Combining Formulas (13)–(15), a system of linear equations represented in vectors is formed,

$$\begin{bmatrix} 0 & \mathbf{1}_n^T \\ \mathbf{1}_n & \mathbf{K} + \gamma^{-1} \mathbf{I}_n \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{y} \end{bmatrix}, \quad (16)$$

where, the elements of kernel matrix $\mathbf{K}$ are computed by kernel function $K(\mathbf{x}_j, \mathbf{x}_k)$, $j, k = 1, 2, \dots, n$, $\mathbf{y} = (y_1, y_2, \dots, y_n)^T$, $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)^T$, $\mathbf{1}_n = (1, 1, \dots, 1)^T$. After solving Equation (16), the predicted output of $f(\mathbf{x})$ for the LSSVR can be obtained,

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b = \sum_{k=1}^n \alpha_k K(\mathbf{x}, \mathbf{x}_k) + b. \quad (17)$$

The process of constructing the LSSVR has the following advantages: (1) converting quadratic programming problems into linear equations significantly improves computational efficiency; (2) introducing the regularization parameter γ to control model complexity effectively can prevent overfitting and enhance model generalization capability; (3) importantly, the number of selected hyperparameters is only two, while SVR has three with an additional insensitivity margin ϵ . Unfortunately, the use of $\mathcal{L}_2$ -norm of error in LSSVR deprives the model sparsity existing in SVR. The fundamental reason lies in its optimization formulation. The $\mathcal{L}_2$ -norm loss, coupled with equality constraints, forces all training samples to satisfy the Karush–Kuhn–Tucker conditions strictly, resulting in non-zero Lagrange multipliers for every data point. This mechanism lacks the threshold-induced sparsity inherent in the ϵ -insensitive loss of standard SVR, where only samples outside the ϵ -tube contribute support vectors, thereby eliminating the model's capacity for sparse representation.

2.3. Comparison with Related SVR Variants and Robust Regression Methods

To further clarify the position of the proposed LI-SVR, this section provides a conceptual and structural comparison with SVR and LSSVR variants, as shown in Figure 1. From Equations (8) and (9), it can be observed that in the parameter optimization of LSSVR, the training errors of all training samples are considered, which gives rise to a loss of the model's sparsity. Standard ϵ -SVR achieves sparsity by introducing an ϵ -insensitive tube, where samples lying inside the tube do not contribute to the regression model. However,

the performance and sparsity of ϵ -SVR critically depend on the preselection of ϵ , which is typically determined heuristically or through an extensive grid search [3]. Adaptive ϵ strategies have been proposed to alleviate this issue, but they often introduce additional algorithmic complexity, data-dependent heuristics, or iterative update rules. In contrast, the proposed LI-SVR eliminates the explicit ϵ parameter and instead introduces a single global envelope variable λ , which is jointly optimized with the regression function. This formulation avoids manual tuning of ϵ while preserving sparsity through the KKT conditions. ℓ_1 -SVR and least absolute deviation regression promote robustness by penalizing absolute errors, while quantile regression focuses on estimating conditional quantiles rather than central tendencies. Although these methods are effective for handling non-Gaussian noise or asymmetric error distributions, their objectives differ fundamentally from the proposed LI-SVR. Specifically, LI-SVR aims to control the maximum approximation error via an ℓ_∞ -type envelope, rather than minimizing cumulative absolute deviations or targeting specific quantiles. As a result, LI-SVR is particularly suitable for applications requiring bounded worst-case prediction errors. Classical robust regression methods, such as Huber regression and Chebyshev approximation, have long employed ℓ_∞ -norm or hybrid loss functions to enhance robustness against outliers. However, these approaches are typically formulated in linear or non-kernelized settings and do not explicitly exploit the primal–dual structure of support vector machines. Moreover, they do not directly address model sparsity in kernel-based regression. By contrast, the proposed LI-SVR integrates an ℓ_∞ -type constraint into the SVR/LSSVR framework, yielding a sparse kernel regression model with a compact set of support vectors.

Motivated by both ϵ -SVR and LSSVR, we introduce the infinite norm on approximation errors to address the aforementioned issues. In the proposed LI-SVR, sparsity emerges intrinsically from the optimization structure and KKT conditions, without requiring additional sparsity-inducing regularizers or post hoc pruning. This leads to a simpler and more interpretable sparse regression model.

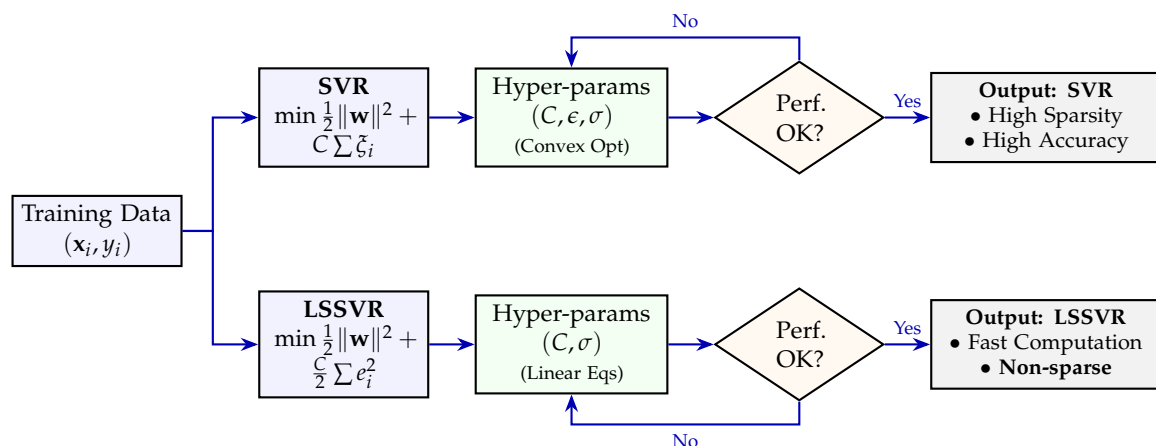


Figure 1. Concise structural diagram for constructing SVR and LSSVR.

3. SVR and LSSVR-Inspired Least $\mathcal{L}_\infty$ -Norm Support Vector Regression

Based on the comparative analysis in Section 2, it is evident that SVR achieves sparsity at the cost of increased hyperparameter complexity, whereas LSSVR simplifies optimization but loses sparsity due to its $\mathcal{L}_2$ -norm error formulation. Section 3 builds directly upon this observation by reformulating the LSSVR objective using an $\mathcal{L}_\infty$ -norm error criterion, with the aim of restoring intrinsic sparsity while preserving convexity and limiting the number of tunable hyperparameters to two.

Actually, model sparsity is guaranteed by whether the absolute values of the Lagrange multipliers solved are sufficiently small. Given that LSSVR utilizes the $\mathcal{L}_2$ -norm of the error to assess modeling accuracy, it inherently lacks model sparsity. In contrast, SVR obtains sparsity by introducing ϵ , but it adds the complexity of optimizing an additional hyperparameter. To deal with these issues, we propose the least infinity ($\mathcal{L}_\infty$)-norm Support Vector Regression (LI-SVR) method, as shown in Figure 2.

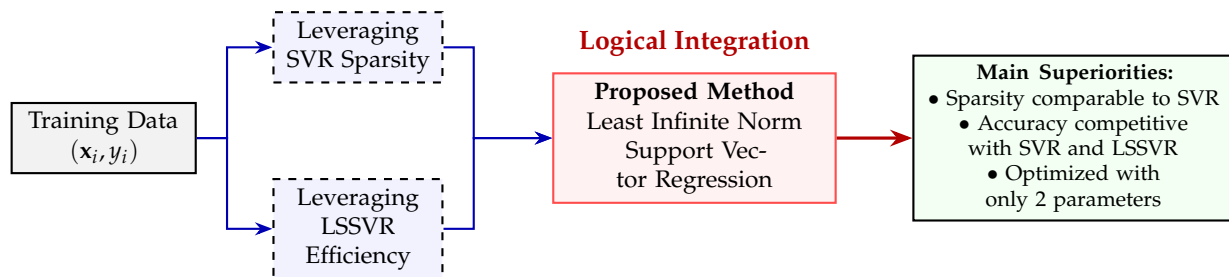


Figure 2. The concise structural framework of the proposed method, illustrating the logical integration of sparsity and parameter efficiency.

3.1. The $\mathcal{L}_\infty$ -Norm on Approximation Error

The $\mathcal{L}_\infty$ -norm (also known as the maximum norm) is defined as the largest element of the absolute value in a vector. In contrast, the $\mathcal{L}_2$ -norm is a square root of the sum of squares in all elements, which calculates the length or distance of the vector in Euclidean space.

Assuming the obtained input–output data and the regression model to be solved are represented as in (1) and (2), respectively, where the measurement of y_k satisfies the following nonlinear system,

$$y_k = g(\mathbf{x}_k), k = 1, 2, \dots, n. \quad (18)$$

According to statistical theory, there exists an arbitrary approximation of the measurement model g by the nonlinear regression model f described in Equation (2). As the approximation precision decreases, fewer SVs are needed in SVR; conversely, more SVs are required. Hence, for any given real continuous function g and $\eta > 0$, a regression model f can be found,

$$\sup_{\mathbf{x}_k \in \mathbf{S}} |f(\mathbf{x}_k) - g(\mathbf{x}_k)| < \eta, \forall k. \quad (19)$$

In traditional LSSVR, the following deviation $e_k, k = 1, 2, \dots, n$ is defined,

$$\begin{aligned} e_1 &= f(\mathbf{x}_1) - y_1 = \mathbf{w}^T \phi(\mathbf{x}_1) + b - y_1 \\ e_2 &= f(\mathbf{x}_2) - y_2 = \mathbf{w}^T \phi(\mathbf{x}_2) + b - y_2 \\ &\vdots \\ e_n &= f(\mathbf{x}_n) - y_n = \mathbf{w}^T \phi(\mathbf{x}_n) + b - y_n. \end{aligned} \quad (20)$$

and its optimization problem employs the $\mathcal{L}_2$ -norm of deviations, implying that all deviations in Formula (20) contribute to the solution of Lagrange multipliers, which is the primary reason for the loss of sparsity.

Now, we introduce $\mathcal{L}_\infty$ -norm on the approximation error to replace the $\mathcal{L}_2$ -norm and solve for the regression model f satisfying condition (19). Initially, we search for the maximum absolute deviation λ in Formula (20), defined as the $\mathcal{L}_\infty$ -norm of the approximation

error. Next, the minimization of λ can be transformed into a min – max optimization problem to get intrinsic sparsity,

$$\begin{aligned}\alpha &= \min \max_{\mathbf{x}_k \in \mathbf{Z}} |y_k - f(\mathbf{x}_k)|, \forall k, \\ &= \arg \min_{\mathbf{w}, b} \max_{\mathbf{x}_k \in \mathbf{Z}} |y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b|. \end{aligned} \quad (21)$$

where $\mathbf{Z}$ denotes the finite training set. For SVR, the interpretation of sparsity implies that contributions of input samples within the pipeline are disregarded by the regression model. Actually, the essentially sparse nature proposed in this paper borrows from the idea of SVR. That is to say, when the deviation e_k satisfies the condition of $-\lambda < e_k < \lambda$, the contribution of the corresponding sample $\mathbf{x}_k$ will be negligible in (21).

Lemma 1. *The min – max optimization problem (21) can be addressed through linear programming,*

$$\begin{aligned} &\min : \lambda, \\ \text{s.t.} &\begin{cases} y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b \leq \lambda, \\ -y_k + \mathbf{w}^T \phi(\mathbf{x}_k) + b \leq \lambda \\ \lambda \geq 0, k = 1, 2, \dots, n, \end{cases} \end{aligned} \quad (22)$$

where $\mathbf{w}$ and b are the parameters to be determined, and λ represents the maximum approximation error.

Proof. Define λ as follows,

$$\lambda = \max |y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b| \quad (23)$$

and the following inequality can be directly derived,

$$|y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b| \leq \lambda \quad (24)$$

By applying the absolute value operation, the system of inequalities (24) can be rewritten in the form of (22). This completes the proof of Lemma 1. $\square$

The sparsity-promoting property of the ℓ_∞ -norm formulation can be intuitively understood from its min–max structure. As shown in (22), the optimization seeks to minimize a single scalar variable λ , which represents the maximum approximation error shared by all training samples. Unlike ℓ_2 -based formulations that accumulate errors over all samples, the objective here is governed solely by the worst-case deviation. As a result, only those samples whose residuals attain the maximum bound λ actively constrain the solution, while samples with smaller errors remain strictly inactive. Consequently, the regression model is effectively determined by a limited subset of critical samples, which naturally leads to a sparse representation. Next, this lemma will be used to establish a sparse regression model in our approach.

3.2. Establishment of Sparse Kernel Regression Model

In fact, early studies [42,43] on the classification of machine learning methods already discussed how the sparsity of models could be enhanced by leveraging the concept of the $\mathcal{L}_\infty$ -norm. A good regression model f should be a comprehensive embodiment of both model sparsity and modeling accuracy, as exemplified by SVR and LSSVR. However, SVR's optimization problem has the following two drawbacks compared to LSSVR: (1) with N training samples, the introduction of slack variables of ξ_k^+, ξ_k^- increases the number of inequality constraints by $2N$; (2) it requires the selection of three hyperparameters, including the kernel parameter σ , regularization factor γ , and insensitivity margin ϵ , while

LSSVR only has two of σ, γ . In light of this [42,43], while maintaining the same number of hyperparameters as LSSVR, we propose the least $\mathcal{L}_\infty$ -norm SVR(LI-SVR). Hence, drawing inspiration from LSSVR, we substitute the $\mathcal{L}_2$ -norm optimization with the min – max problem (22) and yield a new optimization,

$$\min : J(\mathbf{w}, b, \mathbf{e}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + \frac{\gamma}{2} \lambda^2 \quad (25)$$

$$\text{s.t.} \begin{cases} y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b \leq \lambda, \\ -y_k + \mathbf{w}^T \phi(\mathbf{x}_k) + b \leq \lambda \\ \lambda > 0, k = 1, 2, \dots, n, \end{cases} \quad (26)$$

where the first term $\frac{1}{2} \mathbf{w}^T \mathbf{w}$ in (25) controls the model structure, which is minimized to favor smoother model characteristics; the second term λ^2 primarily focuses on improving model accuracy; $\gamma > 0$ is a regularization factor, used to balance the weights between model structure and modeling accuracy. What should be emphasized is that in inequality constraint (26), we only consider $\lambda > 0$. This is because λ represents the maximum one among all absolute deviations. If $\lambda = 0$, then $|\mathbf{w}^T \phi(\mathbf{x}_k) + b - y_k| = 0$, indicating that regression model $f(\mathbf{x})$ can completely fit the actual outputs y , which has a high potential to trigger severe overfitting and result in extreme degradation of the model's generalization performance. It should be emphasized that λ is an optimization variable rather than a hyperparameter. Unlike the ϵ parameter in SVR, which must be manually specified prior to training, λ is automatically determined during optimization as the maximum approximation error, as shown in Equation (29). Therefore, the proposed method involves only two hyperparameters: the regularization coefficient γ and the kernel width σ .

To solve for the weight vector $\mathbf{w}$ and the bias term b , we leverage the Lagrange function with Lagrange multipliers α_k and β_k to convert the constraints into an unconstrained optimization problem,

$$\begin{aligned} \mathcal{L}(\mathbf{w}, b, \lambda, \alpha_k, \beta_k) = & \frac{1}{2} \mathbf{w}^T \mathbf{w} + \frac{\gamma}{2} \lambda^2 + \sum_{k=1}^n \alpha_k (y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b - \lambda) \\ & + \sum_{k=1}^n \beta_k (\mathbf{w}^T \phi(\mathbf{x}_k) + b - y_k - \lambda). \end{aligned} \quad (27)$$

To achieve the optimal solution, partial derivatives of the Lagrange function with regard to $\mathbf{w}, b, \lambda, \alpha_k, \beta_k$ are computed using the Karush–Kuhn–Tucker condition and equated to zero, respectively,

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{w}} = & \mathbf{w} - \sum_{k=1}^n \alpha_k \phi(\mathbf{x}_k) + \sum_{k=1}^n \beta_k \phi(\mathbf{x}_k) = 0 \\ \Rightarrow \mathbf{w} = & \sum_{k=1}^n (\alpha_k - \beta_k) \phi(\mathbf{x}_k), \end{aligned}$$

here, the weight vector $\mathbf{w}$ can be viewed as a linear expansion of coefficients $\alpha_k - \beta_k$ in relation to the nonlinear feature mapping $\phi(\mathbf{x})$.

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \lambda} = & \lambda \gamma - \sum_{k=1}^n \alpha_k - \sum_{k=1}^n \beta_k = 0 \\ \Rightarrow \lambda = & \frac{1}{\gamma} \sum_{k=1}^n (\alpha_k + \beta_k), \end{aligned} \quad (28)$$

From Equation (28), it can be observed that λ is automatically calculated via α_k and β_k , eliminating the need for prior determination.

$$\frac{\partial \mathcal{L}}{\partial b} = \sum_{k=1}^n (-\alpha_k) + \sum_{k=1}^n \beta_k = 0 \Rightarrow \sum_{k=1}^n \alpha_k = \sum_{k=1}^n \beta_k \quad (29)$$

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \alpha_k} &= y_k - (\mathbf{w}^T \phi(\mathbf{x}) + b) - \lambda = 0 \\ \Rightarrow y_k &= \mathbf{w}^T \phi(\mathbf{x}_k) + b + \lambda \end{aligned} \quad (30)$$

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \beta_k} &= \mathbf{w}^T \phi(\mathbf{x}) + b - y_k - \lambda = 0 \\ \Rightarrow y_k &= \mathbf{w}^T \phi(\mathbf{x}_k) + b - \lambda \\ \beta_k &\geq 0, \quad \alpha_k \geq 0, \quad \lambda > 0, \quad k = 1, 2, \dots, n \end{aligned} \quad (31)$$

where Equations (30) and (31) will be employed for analyzing the sparsity properties and determining whether the k -th sample $\mathbf{x}_k$ qualifies as an SV.

Now, substituting Formula (28) into (27), we obtain,

$$\begin{aligned} \mathcal{L}(\alpha_k, \beta_k) &= -\frac{1}{2\gamma} \left(\sum_{k=1}^n (\alpha_k + \beta_k) \right)^2 + \sum_{k=1}^n (\alpha_k - \beta_k) y_k \\ &\quad - \frac{1}{2} \sum_{j=1}^n \sum_{k=1}^n (\alpha_j - \beta_j) (\alpha_k - \beta_k) \phi^T(\mathbf{x}_j) \phi(\mathbf{x}_k) \end{aligned} \quad (32)$$

where the constraints of $\beta_k \geq 0$ and $\alpha_k \geq 0$ in (32) must be met. Maximizing Formula (27) is equivalent to the maximization of (32). However, the maximization of (32) can be reformulated as its dual problem,

$$\begin{aligned} \min : \quad J(\alpha, \beta) &= \frac{1}{2\gamma} \left(\sum_{k=1}^n (\alpha_k + \beta_k) \right)^2 + \sum_{k=1}^n (-\alpha_k + \beta_k) y_k \\ &\quad + \frac{1}{2} \sum_{j=1}^n \sum_{k=1}^n (\alpha_j - \beta_j) (\alpha_k - \beta_k) \phi^T(\mathbf{x}_j) \phi(\mathbf{x}_k) \\ \text{s.t.} \quad &\sum_{k=1}^n (\alpha_k - \beta_k) = 0, \\ &\alpha_k \geq 0, \quad \beta_k \geq 0, \quad k = 1, 2, \dots, n. \end{aligned} \quad (33)$$

Theorem 1. Optimization problem (33) can be solved through quadratic programming, and the solution of α_k and β_k is globally optimal.

Proof. We only need to transform optimization (33) into vector form, as seen in Equation (34), and the theorem is proven. $\square$

$$\begin{aligned} \min : \quad J(\alpha, \beta) &= \frac{1}{2} \begin{pmatrix} \alpha \\ \beta \end{pmatrix}^T \begin{pmatrix} \mathbf{K} + \frac{1}{\gamma} \mathbf{I} & -\mathbf{K} + \frac{1}{\gamma} \mathbf{I} \\ -\mathbf{K} + \frac{1}{\gamma} \mathbf{I} & \mathbf{K} + \frac{1}{\gamma} \mathbf{I} \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \\ &\quad + \begin{pmatrix} -\mathbf{y} \\ \mathbf{y} \end{pmatrix}^T \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \\ \text{s.t.} \quad &(\Theta^T \quad -\Theta^T) \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = 0, \\ &\beta \geq 0, \quad \alpha \geq 0. \end{aligned} \quad (34)$$

Here,

$$\begin{aligned}\alpha &= (\alpha_1, \alpha_2, \dots, \alpha_n)^T, \quad \beta = (\beta_1, \beta_2, \dots, \beta_n)^T, \\ \mathbf{y} &= (y_1, y_2, \dots, y_n)^T, \quad \mathbf{Z} = (0, 0, \dots, 0)^T \\ K(j, k) &= K(\mathbf{x}_j, \mathbf{x}_k) = \phi^T(\mathbf{x}_j)\phi(\mathbf{x}_k), \\ \mathbf{K} &= \begin{pmatrix} K(1,1) & K(1,2) & \cdots & K(1,n) \\ \vdots & \vdots & & \vdots \\ K(n,1) & K(n,2) & \cdots & K(n,n) \end{pmatrix},\end{aligned}$$

$\mathbf{I}$ is an n -dimensional matrix filled with elements that are all equal to 1, Θ represents a column vector with all elements equal to 1, the entries of kernel matrix $\mathbf{K}$ are calculated using kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \phi^T(\mathbf{x}_j)\phi(\mathbf{x}_i)$, which includes commonly used ones such as polynomial, radial basis, and sigmoid kernels, etc. It has been demonstrated that the radial basis kernel, also known as the Gaussian kernel, exhibits superior performance over other kernel functions in handling nonlinear regression problems. Therefore, the Gaussian kernel function with kernel width σ is adopted in this study,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right). \quad (35)$$

As the optimization (32) is a typical quadratic programming problem, the solution obtained is globally optimal, thus avoiding local optima. After solving the optimization problem of (32), we found that the magnitudes of most Lagrange multipliers of α_k and β_k are sufficiently small, even negligible. The maximum approximation error λ can be computed based on (28). Subsequently, we derive a bias term b from Equation (30),

$$\begin{aligned}b &= y_k - \sum_{j=1}^n (\alpha_j - \beta_j) K(\mathbf{x}_k, \mathbf{x}_j) - \lambda \\ &= \frac{\left(y_k - \sum_{j=1, \mathbf{x}_j \in SV}^{n_{sv1}} \alpha_j K(\mathbf{x}_k, \mathbf{x}_j) - \lambda\right)}{n_{sv1}}\end{aligned} \quad (36)$$

or (31),

$$\begin{aligned}b &= y_k - \sum_{j=1}^n (\alpha_j - \beta_j) K(\mathbf{x}_k, \mathbf{x}_j) + \lambda \\ &= \frac{\left(y_k - \sum_{j=1, \mathbf{x}_j \in SV}^{n_{sv2}} - \beta_j K(\mathbf{x}_k, \mathbf{x}_j) + \lambda\right)}{n_{sv2}}\end{aligned} \quad (37)$$

Here, n_{sv1} and n_{sv2} represent the number of significant differences from zero for components α_k and β_k in α and β , respectively. Assuming that the total number of SVs (support vectors) is n_{sv} , and these SVs are represented as $SV_j, j = 1, 2, \dots, n_{sv}$, we can derive an essentially sparse regression model, namely LI-SVR, comparable to SVR and LSSVR,

$$f(\mathbf{x}) = \sum_{j=1}^{n_{sv}} (\alpha_j - \beta_j) K(\mathbf{x}, SV_j) + b. \quad (38)$$

It is also revealed in Section 3.3 that n_{sv} is substantially smaller than n . In practice, the quadratic programming problem in Theorem 1 is solved using standard interior-point solvers. Numerical stability is ensured by kernel matrix regularization and normalization of input features. All experiments employ identical solver tolerances to ensure fair comparisons.

3.3. Support Vectors Analysis for Sparse LI-SVR

The number of SVs plays a crucial role in determining the model complexity of SVR. Traditionally, LSSVR lacks sparsity, meaning that all samples contribute to the model construction. This not only increases the structural complexity of the model, leading to potential overfitting issues, but also deteriorates its generalization performance. Therefore, the method proposed in this paper can essentially produce a sparse regression model after solving the optimization problem (33). The quality of sparsity depends on the number of SVs. In general, more SVs indicate a poorer sparsity, while fewer SVs imply a better one. Below, we will analyze how the SVs are generated in the proposed sparse LI-SVR.

Firstly, we define a deviation $\tilde{e}_k$ between the actual output y_k and predicted values $\tilde{y}_k$ as follows,

$$\begin{aligned}\tilde{e}_k &= y_k - \tilde{y}_k \\ &= y_k - \sum_{j=1}^n (\alpha_j - \beta_j) K(\mathbf{x}_k, \mathbf{x}_j) - b.\end{aligned}\quad (39)$$

where $k = 1, 2, \dots, n$. Upon completing the solution to the convex quadratic programming problem (33), the maximum approximation error λ can be computed from Equation (28). According to conditions (23) and (30), we further obtain,

$$\begin{cases} \lambda = \max\{\tilde{e}_k\}, \\ k = 1, 2, \dots, n, \lambda > 0. \end{cases}\quad (40)$$

It is worth noting that the difference between $\tilde{e}_k$ and λ is that Equation (39) includes both $\tilde{e}_k < \lambda$ and $\tilde{e}_k = \lambda$, whereas Equation (40) contains only those subject to condition of $\tilde{e}_k = \lambda$. Considering that we derive Equation (40) by computing the partial derivative of the Lagrangian function with respect to α_j , the sparse solution of α_j is dominated by Equation (40). Logically, the k -th sample $\mathbf{x}_k$ is selected as an SV only when $\lambda = \tilde{e}_k$ strictly, and the contribution of the corresponding α_k for sparse LI-SVR should not be overlooked. That is to say, only a small fraction of all deviations of (39) satisfies the condition of $\lambda = \tilde{e}_k$. Similarly, we can conduct an analysis on the sparsity of the Lagrange multiplier β_k . From, Equations (23) and (30), we have,

$$\begin{cases} \lambda = \max\{-\tilde{e}_k, k = 1, 2, \dots, n\} \\ k = 1, 2, \dots, n, \lambda > 0. \end{cases}\quad (41)$$

It should be noted that from Equation (30), λ denotes the maximum value of all negative deviations at this point. Equation (41) allows us to ascertain an index, k , of all SVs in which its negative deviations $-\tilde{e}_k$ equal exactly λ . It is evident that the numbers n_{sv1} and n_{sv2} of the SVs are markedly smaller than the total sample size n . Finally, the total number of SVs used in constructing the sparse LI-SVR is computed as n_{sv} (namely, $n_{sv1} + n_{sv2}$), where the size of coefficient $|\alpha_k - \beta_k|$ corresponding to these SVs is not equal to zero. The essence of the obtained sparse LI-SVR can be reflected through Theorem 2.

Theorem 2. Among the Lagrange multipliers $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$ and $\beta = (\beta_1, \beta_2, \dots, \beta_n)$, at most, one of (α_k, β_k) is non-zero, and one of the following three conditions is always satisfied for $k = 1, 2, \dots, n$ and $\lambda > 0$:

- (a) : $\alpha_k > 0, \quad \beta_k = 0;$
- (b) : $\alpha_k = 0, \quad \beta_k > 0;$
- (c) : $\alpha_k = 0, \quad \beta_k = 0;$

Proof. According to the original constraints (26), it is apparent that any one in the Equations (42)–(44) remains valid,

$$y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b = \lambda, \quad (42)$$

$$-\lambda < y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b < \lambda \quad (43)$$

$$-y_k + \mathbf{w}^T \phi(\mathbf{x}_k) + b = \lambda. \quad (44)$$

From the Karush–Kuhn–Tucker (KKT) conditions, we have,

$$\alpha_k (y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b - \lambda) = 0 \quad (45)$$

$$\beta_k (-y_k + \mathbf{w}^T \phi(\mathbf{x}_k) + b - \lambda) = 0. \quad (46)$$

where $\alpha_k \geq 0$, $\beta_k \geq 0$ and $\lambda > 0$.

Equation (45) is evidently true when $\alpha_k = 0$, whereas Equation (42) must hold when $\alpha_k > 0$. Substituting it into Equation (46) results in $\beta_k \cdot (-2\lambda) = 0$, implying $\beta_k = 0$. Condition (a) subjected to both $\alpha_k > 0$ and $\beta_k = 0$ in Theorem 2 is proven. In addition, Equation (46) also holds true when $\beta_k = 0$. If $\beta_k > 0$, then from Equations (44) and (45), it follows that $\alpha_k \cdot (-2\lambda) = 0$, which leads to $\alpha_k = 0$. This further confirms the condition (b) of Theorem 2. Similarly, when Equation (43) holds, both $\alpha_k = 0$ and $\beta_k = 0$ must be met from Equations (45) and (46). We derive the third condition (c) of Theorem 2 again.

However, it is impossible for both $\alpha_k > 0$ and $\beta_k > 0$ to occur simultaneously. If $\alpha_k > 0$ and $\beta_k > 0$ are both met, then we have,

$$\lambda = y_k - \mathbf{w}^T \phi(\mathbf{x}_k) - b \quad (47)$$

$$\lambda = -y_k + \mathbf{w}^T \phi(\mathbf{x}_k) + b, \quad (48)$$

and it is inferred that $\lambda = 0$ is contradictory to the known fact that $\lambda > 0$. This concludes the proof of Theorem 2. $\square$

Although λ is influenced by the largest residual, the regularization term jointly constrains the model complexity and prevents overreaction to individual samples. This trade-off limits the sensitivity of λ to moderate outliers in practice, as confirmed by the experimental results.

In conclusion, we can clearly see that Theorem 2 further elaborates on how to obtain sparse solutions using the proposed method. Additionally, we also observe that the number of samples satisfying condition (c) in Theorem 2 is much higher than the one corresponding to the other two conditions (a) and (b). Therefore, the proposed sparse LI-SVR method has a fundamentally evolutionary development over the traditional LSSVR presented by Johan Suykens and Joos Vandewalle in enhancing model sparsity and generalization ability.

Further, the proposed LI-SVR formulation explicitly minimizes a regularized upper bound of the prediction error, which inherently limits the influence of noise on the regression function. As further demonstrated in the experimental section, the model exhibits stable performance across different noise realizations and noise distributions, indicating robustness against stochastic disturbances.

3.4. Relationship Between the Global Variable λ and the ϵ -Insensitive Loss

In classical ϵ -SVR, sparsity is induced by the ϵ -insensitive loss, where prediction errors within a predefined tolerance band do not contribute to the objective function. In contrast, the proposed LI-SVR replaces the fixed ϵ -tube with a single global envelope variable λ , which is jointly optimized with the regression function. This subsection provides a formal discussion on the relationship between λ and ϵ .

For a fixed regression function $f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$, the ϵ -SVR constraint enforces

$$|y_k - f(\mathbf{x}_k)| \leq \epsilon + \zeta_k, \quad (49)$$

where ϵ is user-specified and ζ_k are slack variables. In LI-SVR, the corresponding constraint can be written as

$$|y_k - f(\mathbf{x}_k)| \leq \lambda, \quad (50)$$

where λ is an optimization variable shared by all samples.

It can be observed that, for any feasible solution of ϵ -SVR, choosing $\lambda = \max_k |y_k - f(\mathbf{x}_k)|$ yields a feasible solution for LI-SVR. Conversely, for a given LI-SVR solution (f, λ) , setting $\epsilon = \lambda$ leads to an ϵ -SVR feasible set that contains the LI-SVR solution as a special case. Therefore, LI-SVR can be interpreted as optimizing an upper bound of the ϵ -insensitive loss, where the tolerance parameter is no longer fixed a priori but is determined adaptively from the data.

More formally, the LI-SVR objective can be viewed as minimizing a regularized upper bound of the maximum absolute deviation, while ϵ -SVR minimizes the sum of deviations exceeding a fixed threshold. When the optimal solution of ϵ -SVR satisfies that all active errors lie exactly on the boundary of the ϵ -tube, the two formulations coincide with $\lambda = \epsilon$. In general cases, LI-SVR yields a conservative envelope that guarantees bounded worst-case errors, whereas ϵ -SVR allows a trade-off between sparsity and tolerance through a predefined margin.

This analysis clarifies that λ should not be regarded as a direct replacement of ϵ , but rather as an adaptive envelope parameter that implicitly controls the insensitivity region through optimization.

In addition, we emphasize that interpreting λ as an “adaptive ϵ ” is not intended to imply a strict theoretical equivalence. Instead, this interpretation arises naturally from the optimality conditions of the proposed formulation.

From the Karush–Kuhn–Tucker (KKT) conditions of the LI-SVR optimization problem, it follows that only samples satisfying,

$$|y_k - f(\mathbf{x}_k)| = \lambda \quad (51)$$

are associated with nonzero dual variables and thus act as support vectors. All remaining samples strictly satisfy

$$|y_k - f(\mathbf{x}_k)| < \lambda \quad (52)$$

and are inactive at the optimum. This property closely parallels the role of the ϵ -insensitive region in classical ϵ -SVR, where samples inside the ϵ -tube do not contribute to the solution.

However, a key distinction is that λ is not predefined but is jointly optimized together with the regression function. As a result, the effective insensitivity region is determined adaptively by the data distribution and regularization trade-off, rather than by user specification. From an optimization viewpoint, λ represents the smallest envelope that simultaneously satisfies all constraints while minimizing a regularized worst-case error.

4. Experimental Analysis

In comparison with LSSVR and SVR methods, the regression prediction using our approach has the following three significant characteristics: (1) for LSSVR, the proposed method not only achieves inherent sparsity but also outperforms SVR; (2) while SVR requires the optimization of three hyperparameters, the proposed method only optimizes two hyperparameters; (3) by introducing the infinite norm of error, the proposed method adaptively computes the maximum error, which is somewhat similar to the insensitive

zone in SVR. Based on this, the following three metrics are introduced to demonstrate the effectiveness of the proposed method: the percentage of support vectors ($SVs\%$), root mean square error ($RMSE$). The first metric is used to evaluate the complexity of the model structure, while the latter one is used to assess the accuracy of the model. Clearly, these metrics are conflicting, and the evaluation of regression prediction should strike a balance between them. $SVs\%$, $RMSE$ and MAE (maximum absolute error) are defined as, respectively,

$$SVs\% = \frac{N_{SV}}{N} \times 100\% \quad (53)$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{j=1}^N (y_k - \hat{y}_k)^2}, \quad (54)$$

$$(55)$$

here, N_{SV} denotes the total number of SVs, which is calculated by applying the conditions of $|\alpha_k - \beta_k| > 1E-8$, N represents the total number of training samples, y_k and $\hat{y}_k$ are the actual and predicted output, respectively. MAE is reported to reflect the worst-case performance, which is particularly relevant for the proposed $\mathcal{L}_\infty$ -norm formulation. To demonstrate the rationality and superiority of the proposed method, we compare its performance with that of classical SVR and LSSVR on several benchmark datasets, including static nonlinear functions and linear/nonlinear dynamic systems. All experimental simulations were conducted in MATLAB 2022b, with the hyperparameters of SVR and LSSVR optimized using the built-in MATLAB packages and the LS-SVMlab Toolbox, respectively. Hyperparameter selection is not the primary focus of this study. In the experiments, the regularization parameter γ and kernel width σ are selected using grid search with cross-validation to ensure fair and stable performance across different datasets.

4.1. Computational Complexity and Scalability

The proposed LI-SVR is formulated as a convex quadratic programming problem with $2n$ inequality constraints, which is structurally simpler than the standard ϵ -SVR formulation involving $4n$ constraints. In terms of computational complexity, standard SVR requires solving a quadratic programming (QP) problem with $2N$ inequality constraints, resulting in a worst-case training complexity between $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$. LSSVR reduces this burden by solving a linear system of size N , leading to a complexity of $\mathcal{O}(N^3)$. The proposed LI-SVR also involves a convex QP problem, but without slack variables and with only $2N$ non-negativity constraints. In practice, the effective number of active constraints is significantly smaller due to sparsity, yielding training times comparable to or lower than standard SVR. In addition, LI-SVR requires only two hyperparameters, namely the kernel width and the regularization parameter, whereas classical ϵ -SVR introduces an additional ϵ -insensitive parameter. This reduction in hyperparameter dimensionality significantly alleviates the computational burden associated with hyperparameter optimization, especially when grid search or cross-validation is employed.

In terms of convergence behavior, LI-SVR can be efficiently solved using standard quadratic programming solvers and exhibits stable convergence due to its convex objective and reduced constraint set. Since the kernel matrix structure is identical to that of LSSVR and SVR, the memory footprint scales quadratically with the number of training samples, which is common to kernel-based methods.

Regarding scalability, LI-SVR inherits the same limitations and advantages as kernel SVR methods. While the current work focuses on moderate-scale datasets, the proposed

formulation is compatible with existing large-scale extensions, such as decomposition methods or approximate kernel techniques, which can be incorporated in future work.

4.2. Static Nonlinear Functions

Initially, we evaluate the performance of our method by applying it to a simple regression task of the sinc function [44], which is defined as:

$$y = \text{sinc}(x) = \frac{\sin x}{x} + v, \quad x \in U[-4\pi, 4\pi] \quad (56)$$

where U denotes a random variable with a uniform distribution in the interval $[-4\pi, 4\pi]$, and v is Gaussian noise with a mean of 0 and a variance of 0.2^2 . To mitigate potential bias in the regression prediction experiments, the mean value derived from ten independent runs for each dataset is used as the evaluation metric. Then, 500 training samples are generated from Equation (56) to establish the regression prediction model using the LI-SVR method. Likewise, 10 independent test datasets are generated from Equation (56), with each test dataset consisting of 500 samples. The results obtained for a classical SVR, LSSVR and LI-SVR (our approach) are listed in Table 1. For training RMSE, our method performs slightly worse than both SVR and LSSVR, while the testing RMSE is between the two. However, the sparsity of our method significantly outperforms them. The SVs% in our method is 2.60%, indicating that only 13 samples (as shown in Figure 3) out of 500 training samples were used to construct the regression prediction model, compared to 445 and 500 samples required by SVR and LSSVR, respectively. Figure 4 gives the indices of these SVs obtained by our method, along with the corresponding magnitudes. Clearly, while maintaining modeling accuracy, the proposed LI-SVR exhibits superior sparsity compared to the classical SVR and LSSVR in terms of model sparsity.

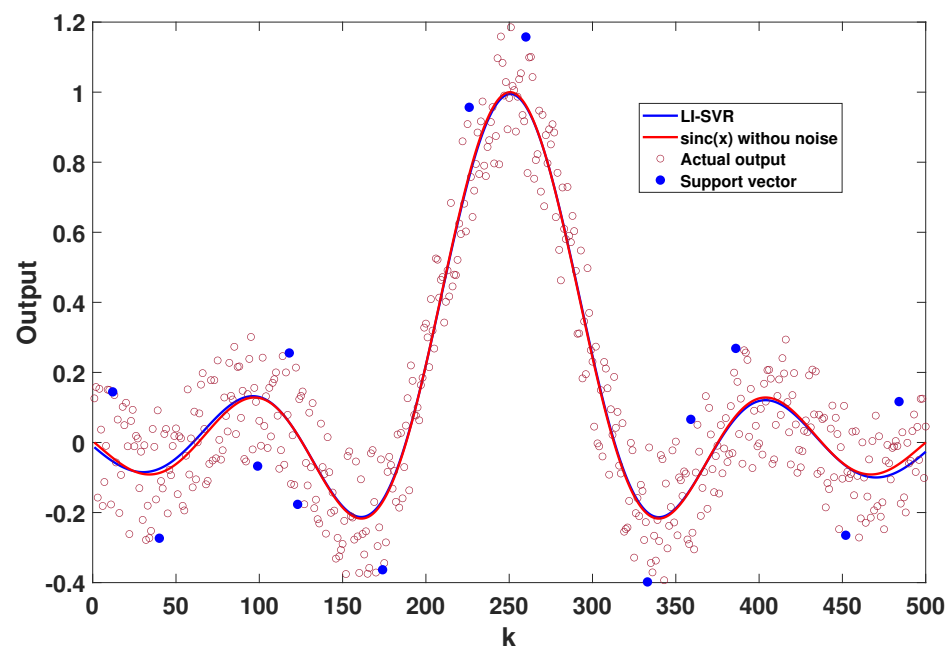


Figure 3. A test output of the proposed LI-SVR.

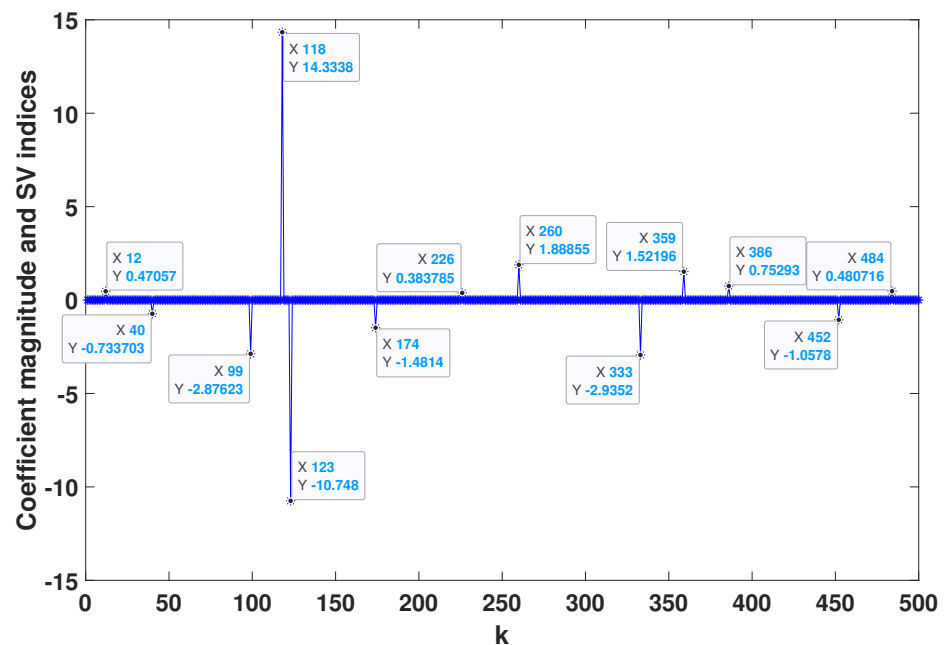


Figure 4. The magnitude of coefficients and the indices of SVs obtained using our approach.

Table 1. The proposed LI-SVR is compared with the classical SVR and LSSVR in terms of training/testing RMSE and SVs%, where the hyperparameter set (σ, γ) in our approach is selected as $(0.8, 100)$.

Model	RMSE (Training)	RMSE (Testing)	SVs%
LSSVR	0.1119	0.1137	100.00%
SVR	0.1127	0.1155	89.00%
LI-SVR	0.1133	0.1143	2.60%

4.3. Linear Dynamic System

Consider a linear dynamic system represented by the following transfer function,

$$G(s) = \frac{C(s)}{R(s)} = \frac{20}{0.05s^2 + 3s + 20} \quad (57)$$

with a sampling time $T_s = 0.01s$, a total of $N = 500$ samples, and input/output delays both set to 2. The input–output pairs of $\{X(k), Y(k)\}$ for constructing LI-SVR are given as follows,

$$\begin{aligned} X(k) &= \{y(k-1), y(k-2), u(k-1), u(k-2)\} \\ Y(k) &= y(k), \quad k = 3, 4, \dots, 500 \end{aligned} \quad (58)$$

here $u(k)$ is a random signal defined as $u(k) = 2.0 \times rand$ in our MATLAB simulation. Figure 5 shows the training output of LI-SVR with hyperparameter set $(\sigma, \gamma) = (0.8, 1000)$, where the blue solid circles represent the SVs. The corresponding magnitude of the LI-SVR model coefficients for these SVs is given in Figure 6, accounting for 10.08% of the total sample size, compared to 66.47% for SVR and 100% for LSSVR. The detailed training/testing RMSEs are presented in Table 2. Although LSSVR achieves the best modeling accuracy, its model structure is significantly more complex, reaching a magnitude of 1×10^7 as indicated by the y-axis in Figure 7. Further, test data were generated based on Equation (58), and LI-SVR achieved favorable predicted outputs, as shown in Figure 8.

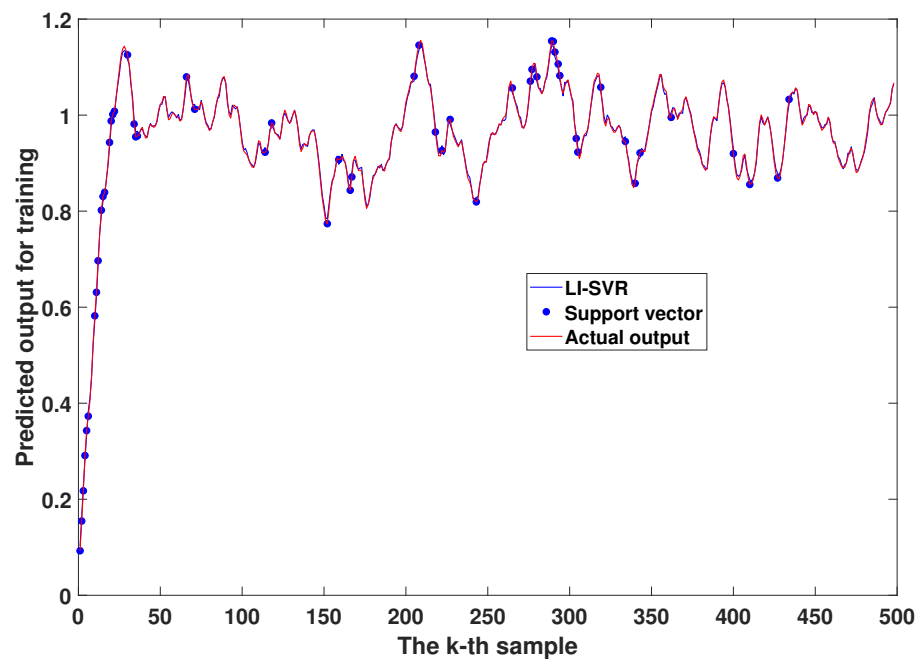


Figure 5. A test output of the proposed LI-SVR.

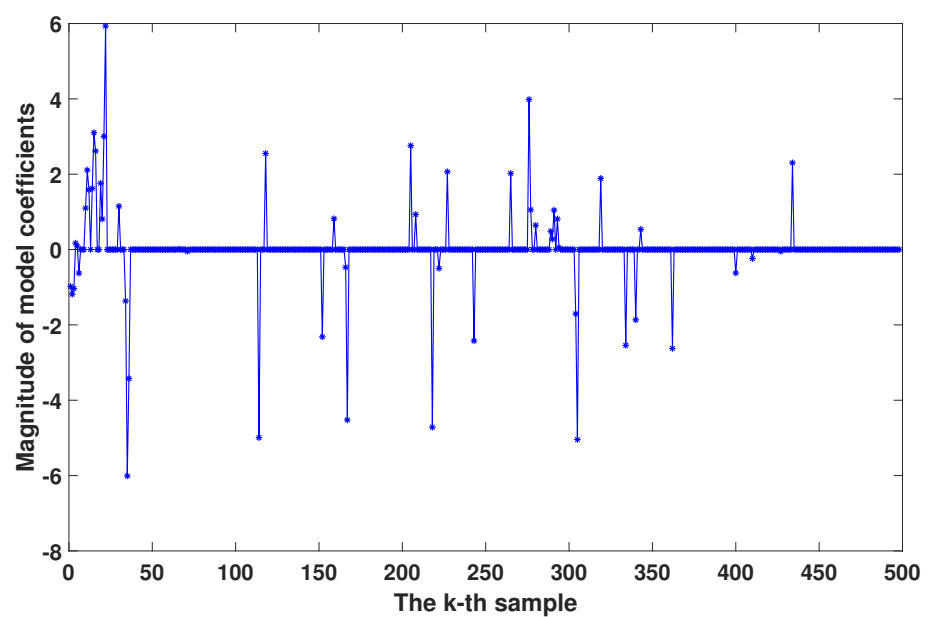


Figure 6. The magnitude of coefficients and the indices of SVs obtained using our approach.

Table 2. The proposed LI-SVR is compared with the classical SVR and LSSVR in terms of training/testing RMSE and SVs%, where the hyperparameter set (σ, γ) in our approach is selected as $(0.8, 1000)$.

Model	RMSE (Training)	RMSE (Testing)	SVs%
LSSVR	6.27×10^{-4}	8.02×10^{-4}	100.00%
SVR	0.0652	0.0833	66.47%
LI-SVR	0.0055	0.0106	10.84%

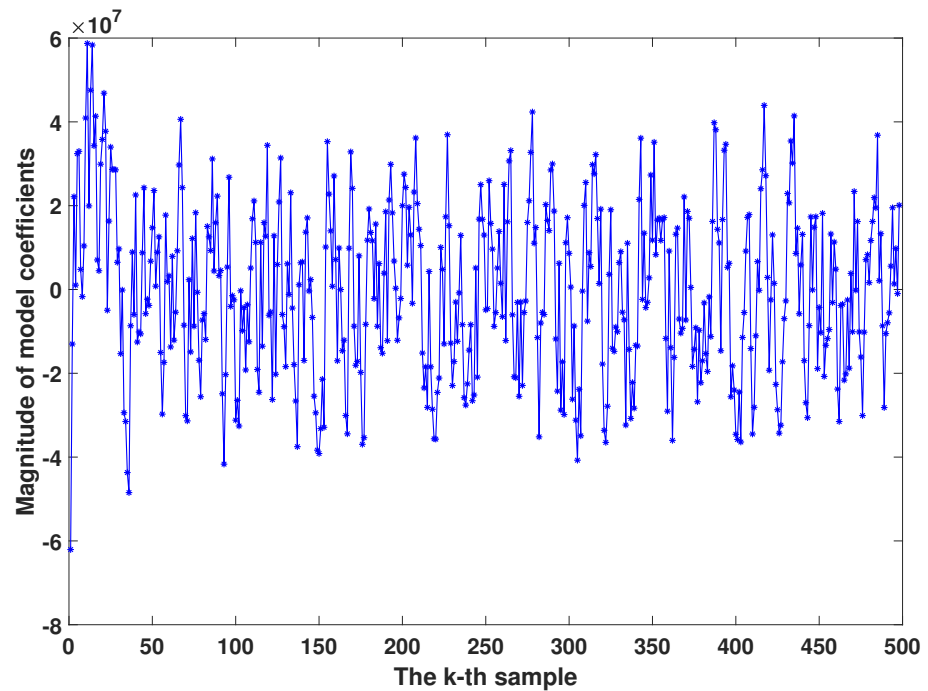


Figure 7. The magnitude of coefficients obtained using LSSVR.

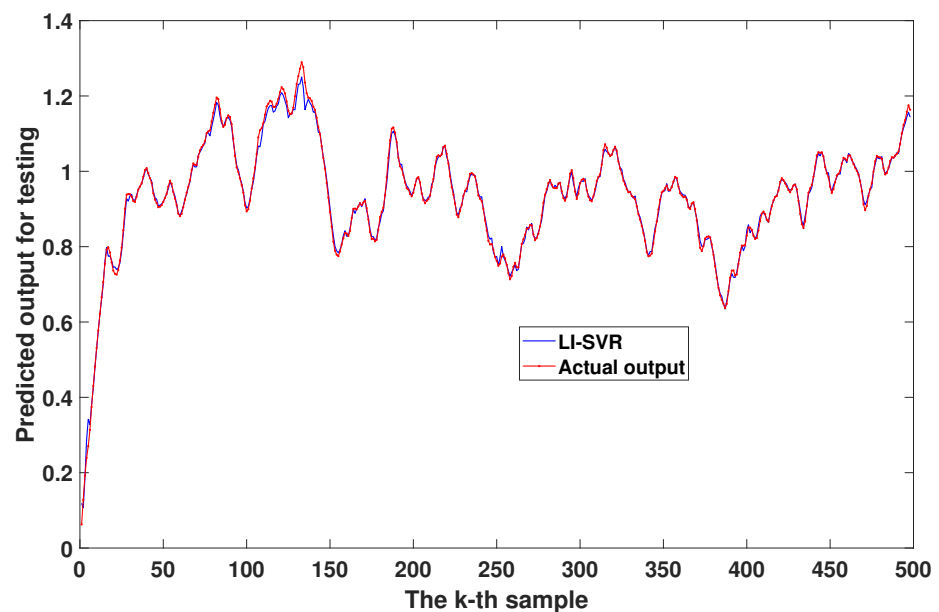


Figure 8. A test output of the proposed LI-SVR.

4.4. Nonlinear Dynamic System

We consider a nonlinear dynamic system [44],

$$y(k+1) = 0.5y(k) + 0.5 \frac{y^2(k-1)}{1+y^2(k-1)} - 0.5y(k-1)u(k) + u(k) \quad (59)$$

with a sampling time $T_s = 0.01s$, a total of $N = 200$ samples. The input–output pairs of $\{X(k), Y(k)\}$ for constructing LI-SVR are the same as (58). When hyperparameter set (σ, γ) are chosen as $(0.2, 1000)$, the predicted output and SV are shown in Figure 9, where the indices associated with the SVs, along with the corresponding model coefficients, are illustrated in Figure 10. The predictive outputs derived from the test data, as governed

by Equation (59), are presented in Figure 11. In Table 3, the modeling precision of the LI-SVR is observed to be intermediate between SVR and LSSVR, yet it demonstrates the most favorable sparsity. In particular, the proposed method constructs a regression model utilizing only 13 SVs out of 200 data points.

Table 3. The proposed LI-SVR is compared with the classical SVR and LSSVR in terms of training/testing RMSE and SVs%, where the hyperparameter set (σ, γ) in our approach is selected as $(0.2, 1000)$.

Model	RMSE (Training)	RMSE (Testing)	SVs%
LSSVR	3.88×10^{-4}	3.56×10^{-3}	100.00%
SVR	0.0170	0.0261	23.74%
LI-SVR	0.0056	0.0067	7.07%

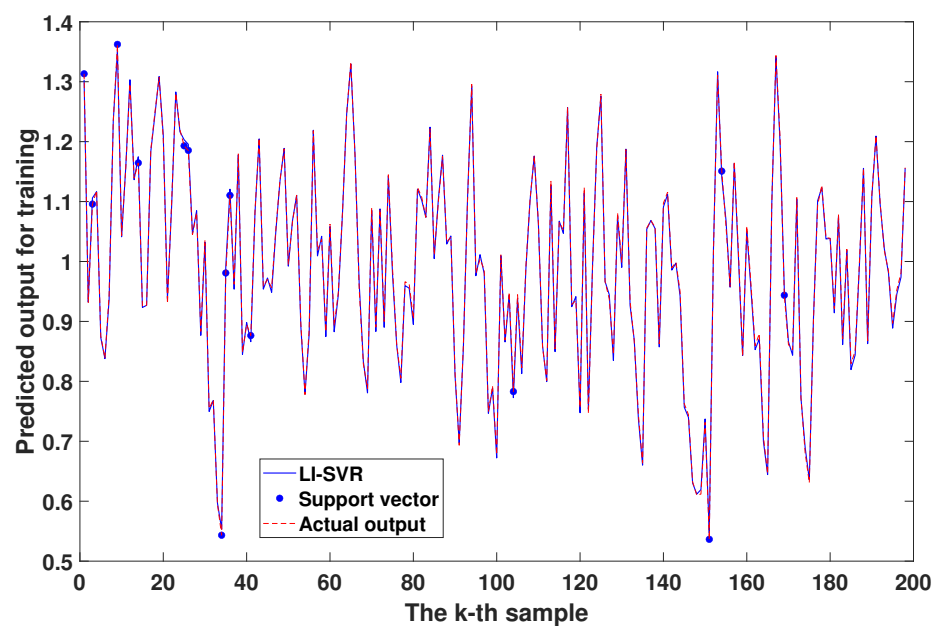


Figure 9. The training output and SV of the proposed LI-SVR.

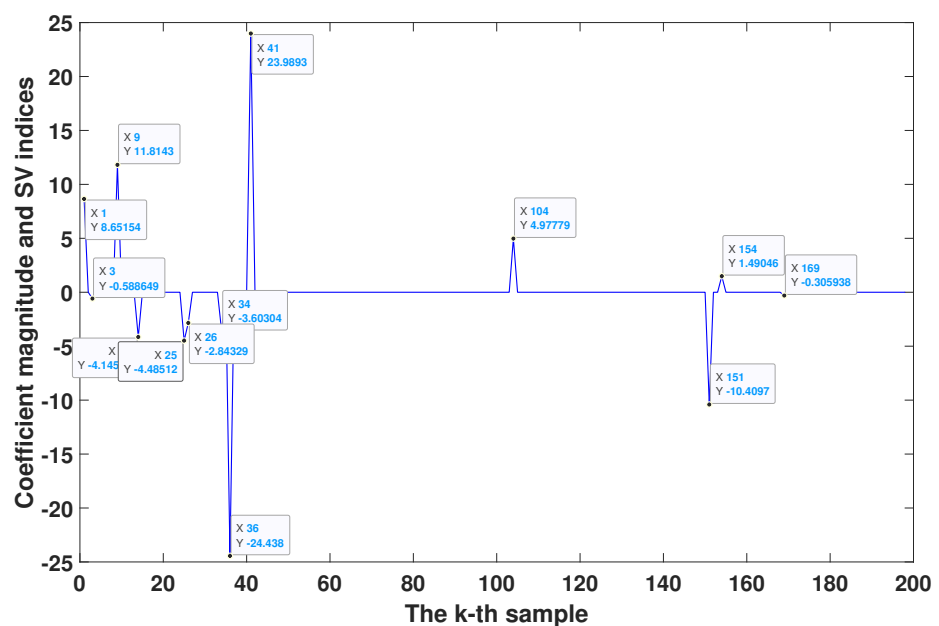


Figure 10. The magnitude of coefficients and the indices of SVs obtained using our approach.

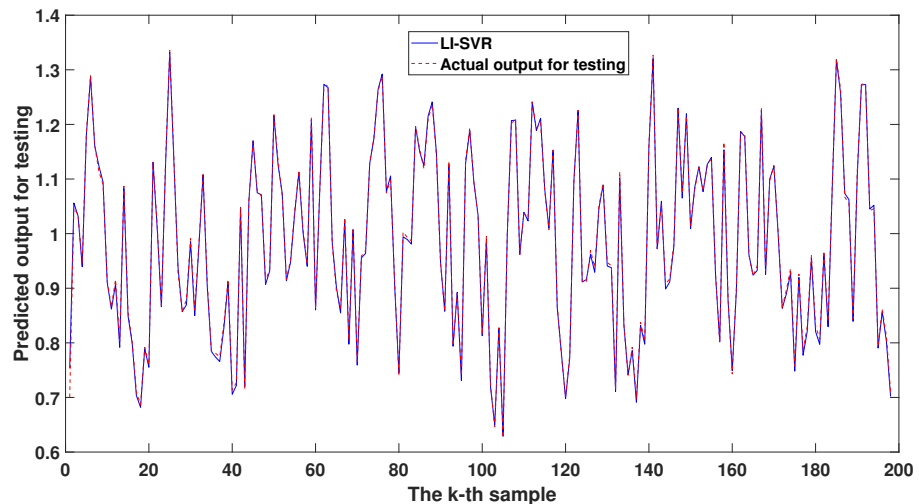


Figure 11. A test output of the proposed LI-SVR.

5. Conclusions

Although LSSVR greatly improves training efficiency by converting inequality into equality constraints, it also forfeits the crucial model sparsity of SVR. To deal with the fundamental flaw of LSSVR, we propose and thoroughly analyze another least infinity-norm ($\mathcal{L}_\infty$ -norm) version for LSSVR and SVR. Essentially, the proposed method exhibits excellent sparsity compared to traditional LSSVR, by adopting the $\mathcal{L}_\infty$ -norm and introducing new constraints to replace the $\mathcal{L}_2$ -norm and its constraints used in the original LSSVR optimization problem. Without the additional insensitivity margin ϵ and slack variables ξ_k seen in SVR, the number of hyperparameters in our approach is the same as LSSVR's. Furthermore, another strength of using the idea of least $\mathcal{L}_\infty$ -norm is that it not only generates truly sparse properties in the regression model, but also automatically obtains the maximum approximation error λ reflecting the model accuracy.

Despite its advantages, the proposed method may face scalability challenges on extremely large datasets due to kernel matrix construction. In addition, its performance depends on kernel selection, and the $\mathcal{L}_\infty$ -norm criterion may be overly conservative in noise-dominated scenarios. Future work will focus on extending the proposed framework using low-rank kernel approximations or online learning strategies to improve scalability in high-dimensional or large-scale settings.

Author Contributions: Conceptualization, X.L., D.L. and C.Z.; methodology, X.L. and D.L.; validation, X.L., D.L. and C.Z.; formal analysis, C.Z.; writing—original draft preparation, X.L. and D.L.; writing—review and editing, X.L. and C.Z.; supervision, C.Z.; funding acquisition, X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (61966006), the Key Laboratory Project of the Guizhou Provincial Department of Education (QJJ[2023]029), and the Zunyi Science and Technology Innovation Team Project (KCTD065), Moutai Institute Joint Science and Technology Research and Development Project (ZSKHHZ[2024] No. 384, ZSKHHZ[2024] No. 385, ZSKHHZ[2023] No. 123), Maotai College Academic New Seed Cultivation and Free Exploration Innovation Special Funding Project (myxm202304) and training program of high-level innovative talents of Moutai institute (mygccrc[2024]011, mygc[2024]012).

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

$\mathbf{x}_k \in \mathbb{R}^d$	Input vector of the k -th training sample
$y_k \in \mathbb{R}$	Output (target) value of the k -th training sample
n	Number of training samples
d	Dimensionality of the input space
$\phi(\cdot)$	Nonlinear feature mapping induced by the kernel function
$\mathcal{D} = \{(\mathbf{x}_k, y_k)\}_{k=1}^n$	Training dataset
$\mathbf{w}$	Weight vector in the feature space
b	Bias term of the regression model
$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$	Regression function
e_k	Approximation error of the k -th sample
λ	Maximum approximation error (optimization variable)
$\gamma > 0$	Regularization coefficient controlling model complexity
$K(\mathbf{x}_i, \mathbf{x}_j)$	Kernel function evaluating inner products in feature space
σ	Kernel width parameter of the RBF kernel
$\ \cdot\ _2$	Euclidean (ℓ_2) norm
$\ \cdot\ _\infty$	Infinity (ℓ_∞) norm, representing the maximum absolute value
α_k	Lagrange multiplier associated with the upper inequality constraint
β_k	Lagrange multiplier associated with the lower inequality constraint
$\boldsymbol{\alpha}$	Vector of Lagrange multipliers $\{\alpha_k\}_{k=1}^n$
$\boldsymbol{\beta}$	Vector of Lagrange multipliers $\{\beta_k\}_{k=1}^n$
$\boldsymbol{\eta}$	Dual variable associated with non-negativity constraints
$J(\cdot)$	Objective function of the optimization problem
SV	Support vector (sample with nonzero dual variables)
RMSE	Root mean square error used for performance evaluation
MAE	Maximum approximation error
LSSVR	Least Squares Support Vector Regression
SVR	Standard Support Vector Regression
LI-SVR	Least ℓ_∞ -norm regularized SVR

References

1. Vapnik, V. *The Nature of Statistical Learning Theory*; Springer: New York, NY, USA, 1995.
2. Smola, A.J.; Schölkopf, B. A tutorial on support vector regression. *Stat. Comput.* **2004**, *14*, 199–222.
3. Burges, C.J.C. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.* **1998**, *2*, 121–167.
4. Zhao, Y.P.; Huang, G.; Hu, Q.K.; Li, B. An improved weighted one class support vector machine for turboshaft engine fault detection. *Eng. Appl. Artif. Intell.* **2020**, *94*, 103796.
5. K.; ukuri, S.T.; Senanyaka, J.S.L.; Huynh, V.K.; Robbersmyr, K.G. A two-stage fault detection and classification scheme for electrical pitch drives in offshore wind farms using support vector machine. *IEEE Trans. Ind. Appl.* **2019**, *55*, 5109–5118.
6. Alamsabi, M.; Tchuindjang, M.; Brohi, S. Embedding-Based Detection of Indirect Prompt Injection Attacks in Large Language Models Using Semantic Context Analysis. *Algorithms* **2026**, *19*, 92.
7. Ahmed, A.; Bosnić, Z. Machine learning for enabling high-data-rate secure random communication: SVM as the optimal choice over others. *Mathematics* **2025**, *13*, 3590.
8. Colgan, M.S.; Baldeck, C.A.; Féret, J.B.; Asner, G.P. Mapping savanna tree species at ecosystem scales using support vector machine classification and BRDF correction on airborne hyperspectral and LiDAR data. *Remote Sens.* **2012**, *4*, 3462–3480.
9. Chakrabarty, A.; Dinh, V.; Corless, M.J.; Rundell, A.E.; Zak, S.H.; Buzzard, G.T. Support vector machine informed explicit nonlinear model predictive control using low-discrepancy sequences. *IEEE Trans. Autom. Control* **2016**, *62*, 135–148.
10. Kirar, J.S.; Agrawal, R.K. Composite kernel support vector machine based performance enhancement of brain computer interface in conjunction with spatial filter. *Biomed. Signal Process. Control* **2017**, *33*, 151–160.
11. Jiang, W.; Siddiqui, S. Hyper-parameter optimization for support vector machines using stochastic gradient descent and dual coordinate descent. *EURO J. Comput. Optim.* **2020**, *8*, 85–101.

12. Azadeh, A.; Saberi, M.; Kazem, A.; Ebrahimipour, V.; Nourmohammadzadeh, A.; Saberi, Z. A flexible algorithm for fault diagnosis in a centrifugal pump with corrupted data and noise based on ANN and support vector machine with hyperparameters optimization. *Appl. Soft Comput.* **2013**, *13*, 1478–1485.
13. Shakya, D.; Deshphe, V.; Safari, M.J.S.; Agarwal, M. Performance evaluation of machine learning algorithms for the prediction of particle Froude number (F_m) using hyper-parameter optimizations techniques. *Expert Syst. Appl.* **2024**, *256*, 124960.
14. Xu, H.; Chen, C.; Zheng, H.; Luo, G.; Yang, L.; Wang, W.; Wu, S.; Ding, J. AGA-SVR-based selection of feature subsets and optimization of parameter in regional soil salinization monitoring. *Int. J. Remote Sens.* **2020**, *41*, 4470–4495.
15. Ramadevi, P.; Das, R. An extensive analysis of machine learning techniques with hyper-parameter tuning by Bayesian optimized SVM kernel for the detection of human lung disease. *IEEE Access* **2024**, *12*, 97752–97770.
16. Ji, Y.; Xu, K.; Zeng, P.; Zhang, W. GA-SVR algorithm for improving forest above ground biomass estimation using SAR data. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 6585–6595.
17. Li, Y.; Liu, G.; Jiao, L.; Marturi, N.; Shang, R. Hyper-parameter optimization using MARS surrogate for machine-learning algorithms. *IEEE Trans. Emerg. Top. Comput. Intell.* **2019**, *3*, 287–297.
18. Zhu, F.; Gao, J.; Xu, C.; Yang, J.; Tao, D. On selecting effective patterns for fast support vector regression training. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *29*, 3610–3622.
19. Zhao, Y.; Zhang, W.; Liu, X. Grid search with a weighted error function: Hyper-parameter optimization for financial time series forecasting. *Appl. Soft Comput.* **2024**, *154*, 111362.
20. Mantovani, R.; Rossi, A.; Vanschoren, J.; Bischl, B.; Carvalho, A. Effectiveness of random search in SVM hyper-parameter tuning. In Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 12–17 July 2015.
21. Nugroho, A.; Suhartanto, H. Hyper-parameter tuning based on random search for DenseNet optimization. In Proceeding of the 2020 7th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), Semarang, Indonesia, 24–25 September 2020.
22. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
23. He, H.; Wang, K.; Jiang, Y.; Pei, H. Quadratic hyper-surface kernel-free large margin distribution machine-based regression and its least-square form. *Mach. Learn. Sci. Technol.* **2024**, *5*, 025024.
24. Snoek, J.; Larochelle, H.; Adams, R.P. Practical bayesian optimization of machine learning algorithms. In Proceedings of Advances in Neural Information Processing Systems 25 (NIPS 2012), Lake Tahoe, NV, USA, 3–6 December 2012.
25. Lahmiri, S.; Tadj, C.; Gargour, C.; Bekiros, S. Optimal tuning of support vector machines and k-NN algorithm by using Bayesian optimization for newborn cry signal diagnosis based on audio signal processing features. *Chaos Solitons Fractals* **2023**, *167*, 112972.
26. Wicaksono, A.S.; Supianto, A.A. Hyper parameter optimization using genetic algorithm on machine learning methods for online news popularity prediction. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *12*, 263–267.
27. Korovkinas, K.; Danenas, P.; Garsva, G. Support vector machine parameter tuning based on particle swarm optimization metaheuristic. *Nonlinear Anal. Model. Control* **2020**, *25*, 266–281.
28. Hsia, J.; Lin, C. Parameter selection for linear support vector regression. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 5639–5644.
29. Suykens, J.A.K.; Vewalle, J. Least squares support vector machine classifiers. *Neural Process. Lett.* **1999**, *9*, 293–300.
30. Suykens, J.A.K.; Lukas, L.; Van Dooren, P.; De Moor, B.; Vewalle, J. Least squares support vector machine classifiers: A large scale algorithm. In Proceedings of the European Conference on Circuit Theory and Design (ECCTD), Stresa, Italy, 29 August–2 September 1999.
31. Yan, H.; Qi, Y.; Ye, Q.L.; Yu, D.Y. Robust least squares twin support vector regression with adaptive FOA and PSO for short-term traffic flow prediction. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 14542–14555.
32. Suykens, J.A.K.; Vewalle, J.; De Moor, B.L.R. Optimal control by least squares support vector machines. *Neural Netw.* **2001**, *14*, 23–35.
33. Liu, X.; Dong, X.G.; Zhang, L.; Chen, J.; Wang, C.J. Least squares support vector regression for complex censored data. *Artif. Intell. Med.* **2023**, *136*, 102497.
34. Goethals, I.; Pelckmans, K.; Suykens, J.A.K.; De Moor, B. Subspace identification of Hammerstein systems using least squares support vector machines. *IEEE Trans. Autom. Control* **2005**, *50*, 1509–1519.
35. Suykens, J.A.K.; De Brabanter, J.; Lukas, L.; Vewalle, J. Weighted least squares support vector machines: Robustness and sparse approximation. *Neurocomputing* **2002**, *48*, 85–105.
36. Suykens, J.A.K.; De Brabanter, J.; Lukas, L.; Vewalle, J. Benchmarking least squares support vector machine classifiers. *Mach. Learn.* **2004**, *54*, 5–32.
37. Hu, L.; Yi, G.X.; He, C. A sparse algorithm for adaptive pruning least square support vector regression machine based on global representative point ranking. *J. Syst. Eng. Electron.* **2021**, *32*, 151–162.
38. Yang, L.X.; Yang, S.Y.; Zhang, R.; Jin, H.H. Sparse least square support vector machine via coupled compressive pruning. *Neurocomputing* **2014**, *131*, 77–86.

39. Hong, X.; Mitchell, R.; Fatta, G.D. Simplex basis function based sparse least squares support vector regression. *Neurocomputing* **2019**, *330*, 394–402.
40. Mall, R.; Suykens, J.A.K. Very sparse LSSVM reductions for large-scale data. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 1086–1097.
41. Ma, Y.; Liang, X.; Sheng, G.; Kwok, J.T.; Wang, M.; Li, G. Noniterative sparse LS-SVM based on globally representative point selection. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 788–798.
42. Chen, D.-R.; Wu, Q.; Ying, Y.; Zhou, D.-X. Support vector machine soft margin classifiers: Error analysis. *J. Mach. Learn. Res.* **2004**, *5*, 1143–1175.
43. Ting, K.; Zhang, L.; Ge, X.; Lv, H.; Li, M. A robust least squares support vector machine based on L^∞ -norm. *Neural Process. Lett.* **2020**, *52*, 2371–2397.
44. Liu, X.; Yan, G.; Zhang, F.; Zeng, C.; Tian, P. Linear Programming-Based Sparse Kernel Regression with L_1 -Norm Minimization for Nonlinear System Modeling. *Processes* **2024**, *12*, 2358.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.