1-Norm random vector functional link networks for classification problems

Barenya Bikash Hazarika^{1,2} · Deepak Gupta¹

Received: 22 June 2021 / Accepted: 28 January 2022 / Published online: 26 February 2022
© The Author(s) 2022

Abstract

This paper presents a novel random vector functional link (RVFL) formulation called the 1-norm RVFL (1N RVFL) networks, for solving the binary classification problems. The solution to the optimization problem of 1N RVFL is obtained by solving its exterior dual penalty problem using a Newton technique. The 1-norm makes the model robust and delivers sparse outputs, which is the fundamental advantage of this model. The sparse output indicates that most of the elements in the output matrix are zero; hence, the decision function can be achieved by incorporating lesser hidden nodes compared to the conventional RVFL model. 1N RVFL produces a classifier that is based on a smaller number of input features. To put it another way, this method will suppress the neurons in the hidden layer. Statistical analyses have been carried out on several real-world benchmark datasets. The proposed 1N RVFL with two activation functions viz., ReLU and sine are used in this work. The classification accuracies of 1N RVFL are compared with the extreme learning machine (ELM), kernel ridge regression (KRR), RVFL, kernel RVFL (K-RVFL) and generalized Lagrangian twin RVFL (GLTRVFL) networks. The experimental results with comparable or better accuracy indicate the effectiveness and usability of 1N RVFL for solving binary classification problems.

Keywords 1 Norm · Single layer feed-forward neural network · Random vector functional link · Sparseness · Classification

Introduction

Neural networks are powerful models for data mining and information engineering which can learn from data to construct feature-based classification models and nonlinear prediction models. Training neural networks (NNs) requires the optimization of a highly non-convex landscape with several local minima and saddle points. Alternative kernel-based methods like support vector machine (SVM) [7, 11, 17] on the other hand, produce well-posed convex problems, which is one of the main reasons for their success over the last few decades. Kernel methods, however, fail to scale effectively to large datasets because one of their main tasks is to compute pairwise kernel values over the complete

dataset. The main benefit of simple NN architecture is that it enables an acceptable level of solution to be achieved in one-hundredth (or even one-millionth) of the time taken by larger, more complex models while maintaining high optimality [37]. A single hidden layer feed-forward NN (SLFN) can handle a function that contains non-linearity with arbitrary precision [26]. They have been widely implemented in various problems associated with classification [19, 22, 24, 33]. Moreover, one of the most popular categories of NN is feed-forward NNs with random weights which were popularized by Pao et al. [32] in their work. They introduced novel random vector functional link networks (RVFL) [5, 8, 30]. In RVFL, it is possible to connect inputs and outputs directly, leading towards an excellent generalization performance. The weights between the input and the hidden layers can also be produced randomly [16, 45]. Zhang and Suganthan [45] further extended the study by performing a comprehensive evaluation of RVFL networks. They performed operations on RVFL using some popular activation functions and discovered that the hardlim and sign activation function significantly degrade the performance of the algorithm. They also suggested that the bias used in RVFL can be a tunable configuration for specific problems. Li et al. [25] pro-

✉ Deepak Gupta
deepakjnu85@gmail.com; deepak@nitap.ac.in
Barenya Bikash Hazarika
barenya1431@gmail.com

¹ Department of Computer Science and Engineering, National Institute of Technology, Jote, Arunachal Pradesh, India

² Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India

posed a novel SVM+ which uses the learning using privileged information (LUPI) [40] procedure. They further suggested a kernelized version of that model for non-linear data processing. They also use the QP solver from MATLAB R2014b as a starting point, to solve the QP problem and proposed MAT-SVM+. Inspired by the work of Li et al. [25], Zhang and Wang [48] embedded the LUPI in the RVFL and proposed a novel RVFL+ model. They further used the kernel trick in the model and proposed a kernel RVFL+ (KRVFL+). Both RVFL+ and KRVFL+ are in analogy to Teacher–Student Interaction [39] in the human learning process. Xu et al. [43] proposed a novel kernel-based RVFL model (K-RVFL) for learning the spatiotemporal dynamic process. Because of the complexity embedded in the kernel, the K-RVFL can handle the complex process better. Kernel ridge regression (KRR) [36] has gained the attention of researchers over the last few decades due to its non-iterative learning approach. It has been widely used to solve a variety of classification [23, 35] and regression [29, 42] problems. KRR is computationally fast since, it adapts equality constraints rather than inequality constraints, and solves a series of linear equations. Several variants of KRR have been proposed to improve its classification performance. For example, Zhang et al. [46], Chang et al. [6], Zhang and Suganthan [47] and more. However, recently the growing popularity of extreme learning machine (ELM) [21, 27] is because of its high generalization performance with low computational cost [2, 14, 15, 18, 20, 38]. Peng et al. [34] proposed a novel discriminative graph regularized extreme learning machine (GELM) model to improve the classification ability of ELM. Due to its closed-form solution, the outputs for GELM can be obtained efficiently. As per the theory, conventional ELM does not guarantee its convergence. The correct convergence results have been shown and proved in Igel'nik and Pao [22], Wang and Wan [41]. The RVFL has been a very efficient and powerful model for tasks related to classification and regression. Despite the high computational efficiency and high generalization ability of RVFL, it was observed that because of the randomly selected weights and hidden layer biases, it requires many nodes to accomplish satisfactory performance. Recently, the 1 norm regularization has gained tremendous popularity among researchers [13, 44] since it results in sparse outputs. The sparse output indicates that most of the elements in the output matrix are zero, hence, the decision surface can be obtained by incorporating lesser hidden nodes compared to the conventional models. Also, these sparse models are easily implementable [1]. An influential contribution in this direction is the 1-norm SVM developed by Mangasarian [28]. The solution of 1-norm SVM is computed by solving its exterior penalty problem as an unconstrained convex minimization problem using the Newton Armijo algorithm. Hence, inspired by the work of Mangasarian [28] and the recent significant pieces of literature on 1-norm regularizations, this paper proposes

Table 1 Pros and cons of a few related models

Model	Advantage	Limitation
ELM	Computationally efficient	Not sparse Noise and outlier sensitive Overfitting problem
KRR	High generalization ability Computationally efficient	Noise and outlier sensitive Not sparse
RVFL	High generalization ability Computationally computational cost Low complexity	Not Sparse Noise and outlier sensitive

a novel 1 norm random vector functional link (1N RVFL) network for binary classification. The key innovations of this work are:

1. A novel 1N RVFL classifier is proposed using two different activation functions.
2. Due to the incorporation of the L1 norm in the proposed model, sparse feature vectors are generated, which is a very useful property for problems related to classification.
3. 1N RVFL produces a classifier that is based on a smaller number of input features. To put it another way, this method will suppress the required number of neurons in the hidden layer.
4. Experiments on real world datasets have been considered to demonstrate the classification ability of 1N RVFL compared to other models.

The advantages and limitations of a few related classifiers are tabulated in Table 1.

The remaining paper is structured as follows; Section “[Mathematical background](#)” gives a brief mathematical description of a few related models, viz., ELM, KRR, GLTRVFL and RVFL. Section “[Proposed 1-norm random vector functional link \(1N RVFL\)](#)” shows the formulation and description of the proposed 1N RVFL model. In Sect. [Simulation and analysis of results](#), the numerical experiments and comparative analysis with ELM, KRR, RVFL, K-RVFL and GLTRVFL are undertaken. In Section “[Conclusion](#)”, we conclude the paper.

Mathematical background

Consider, $x_i \in R^n$ for $i = 1, 2, 3, \dots, m$, to be an n -dimensional input vector and D is the $m \times n$ dimensional matrix of training examples. $y_i \in \{+1, -1\}$ for $i = 1, 2, 3, \dots, m$, is the class level of D ; and hence $y = (y_1, \dots, y_m)^T$ is the class labels vectors and the diagonal matrix y is presented by $Y = \text{diag}(y)$. Let A data samples

have +1 and B data samples have -1 class label of $m_1 \times n$ and $m_2 \times n$ dimensions respectively.

The ELM model

Let, $\beta = (\beta_1, \dots, \beta_T)^t$, where $T = L + n$, be the weight vector (WV) to the output neuron with L indicates the hidden layer nodes quantity. $h_l(x_i) = G(a_l, b, x_i)$ for $l = 1, \dots, L$ and $i = 1, 2, 3, \dots, m$ be the output of the activation function $G(., ., .)$ of the l th hidden layer neuron with respect to the i th training sample. $a_l = (a_{l1}, \dots, a_{lm})^t$ indicates the WV and b represents the bias to the hidden layer nodes.

The output equation for ELM [21] can be expressed as:

$$y = H\beta, \quad (1)$$

The Hessian matrix can be formulated as $H = [h_1(D) \dots h_L(D)]$, i.e.,

$$H = \begin{bmatrix} h_1(x_1) & \dots & h_L(x_1) \\ \dots & \dots & \dots \\ h_1(x_m) & \dots & h_L(x_m) \end{bmatrix},$$

β represents the solution in the primal space that can computed as:

$$\beta = H^\dagger y, \quad (2)$$

where, $H^\dagger$ is the Moore–Penrose inverse of H Now, the final classifier of ELM may be expressed as,

$$f(x) = \text{sign}(h(x)^t \beta). \quad (3)$$

The KRR model

The primal problem of KRR [36] may be defined as:

$$\begin{aligned} \min \quad & \frac{C}{2} \|w\|^2 + \frac{1}{2} \|\psi\|^2, \\ \text{s.t.,} \quad & y - \varphi(x)^t w = \psi, \end{aligned} \quad (4)$$

where w is the unknown, e is the one's vector and ψ is the slack variable. y is the output vector and $\varphi(x)$ indicates the feature mapping function of the input x .

The Lagrangian of (4) may be formulated as:

$$\min \quad \frac{C}{2} \|w\|^2 + \frac{1}{2} \|\psi\|^2 - \ell^t \{y - \varphi(x)^t w - \psi\}, \quad (5)$$

where ℓ is the Lagrangian multiplier.

Now equating Eq. (5) to zero and further applying the KKT condition, the dual form may be obtained as:

$$L = -\frac{1}{2C} \ell^t \varphi(x) \varphi(x)^t \ell - \frac{1}{2} \ell^t I \ell - y^t \ell, \quad (6)$$

where I is the identity matrix with the appropriate dimension.

For a new input example, $x \in \mathbb{R}^n$ the KRR classifier may be generated as:

$$f(x) = \text{sign} \left\{ -\frac{1}{C} \varphi(x)^t \alpha \right\}.$$

The RVFL model

RVFL [31] is a type of SLFN that randomly generates the weights to the hidden layer nodes and fixes them without tuning them iteratively.

The regularized version of RVFL can be expressed as

$$\min \|y - \Omega\beta\|^2 + C\|\beta\|^2, \quad (7)$$

where $\Omega = [H \ D]$. Now, by differentiating (7) with respect to β and further equating it to zero we obtain,

$$\beta = (\Omega^t \Omega + CI)^{-1} \Omega^t y. \quad (8)$$

For any new instance, x the classification function for RVFL can be generated as,

$$f(x) = \text{sign}([h(x) \ x]\beta) \quad (9)$$

where, $h(x) = [h_1(x) \dots h_L(x)]$.

The GLTRVFL model

Recently Borah and Gupta [3] proposed a generalized Lagrangian RVFL model called GLTRVFL. The primal problems of GLTRVFL are:

$$\begin{aligned} \min \quad & \frac{1}{2} \left\| \Omega_1 \beta_1 \right\|_2^2 + \frac{C_1}{2} \psi^t \psi, \\ \text{s.t.,} \quad & \Omega_2 \beta_1 \geq e - \psi, \end{aligned} \quad (10)$$

and

$$\begin{aligned} \min \quad & \frac{1}{2} \left\| \Omega_2 \beta_2 \right\|_2^2 + \frac{C_2}{2} \zeta^t \zeta, \\ \text{s.t.,} \quad & \Omega_1 \beta_2 \geq e - \zeta \end{aligned} \quad (11)$$

The dual formulation of (10) and (11) may be expressed in generalized form as:

Now (10) and (11) can be expressed in dual form and after forming their duals we apply the Newton iterative technique, which can be expressed as:

$$\nabla_1(\alpha_1^i) = \left(\left(\Omega_2 (\Omega_1^t \Omega_1)^{-1} \Omega_2^t + \frac{I}{C_1} \right) \alpha_1^i - e \right) - \left(\left(\Omega_2 (\Omega_1^t \Omega_1)^{-1} \Omega_2^t + \frac{I}{C_1} \right) \alpha_1^i - \gamma_1 \alpha_1^i e \right) \quad (12)$$

and

$$\nabla_2(\alpha_2^i) = \left(\left(\Omega_1 (\Omega_2^t \Omega_2)^{-1} \Omega_1^t + \frac{I}{C_2} \right) \alpha_2^i - e \right) - \left(\left(\Omega_1 (\Omega_2^t \Omega_2)^{-1} \Omega_1^t + \frac{I}{C_2} \right) \alpha_2^i - \gamma_2 \alpha_2^i e \right). \quad (13)$$

Proposed 1-norm random vector functional link (1N RVFL)

The 1-norm RVFL with absolute loss is suggested in this section as a standardized classification model resulting in a robust representation of the model. Moreover, motivated by the study of Mangasarian [28], the proposed 1N RVFL model is formulated by using the Newton–Armijo algorithm that considers its dual exterior penalty problem as an unconstrained convex minimization problem. The proposed formulation leads to an iterative solution for the binary classification problem that is simple and rapidly converging.

Consider the regularized formulation of RVFL as:

$$\min_{\beta \in \mathbb{R}^l} \|y - \Omega \beta\|_1 + C \|\beta\|_1, \quad (14)$$

where, $C > 0$ is the trade-off parameter. By using the same procedure as Mangasarian [28] and Balasundaram and Gupta [1], Eq. (14) can be rewritten in linear programming problem form as:

$$\begin{aligned} \beta &= s - t \text{ and } y - \Omega \beta = p - q \\ \text{s.t., } s, t &\geq 0 \in \mathbb{R}^l \text{ and } p, q \geq 0 \in \mathbb{R}^m \end{aligned} \quad (15)$$

Using (15) on (14), the linear programming RVFL in primal form as:

$$\begin{aligned} \min_{r, s, p, q} \quad & d_l^t(s + t) + C d_m^t(p + q) \\ \text{s.t., } \quad & \Omega(s - t) - (p - q) = y, \quad p, q \geq 0, \end{aligned} \quad (16)$$

where d_l and d_m are one's vector of dimensions l and m , respectively. The optimization problem of (16) is easily solvable using the optimization toolbox in MATLAB.

However, it is recommended to determine its dual external penalty problem as an unconstrained minimization problem in m variables, whose solution can be obtained by the Newton–Armijo technique. This is due to an increase in the number of unknowns and constraints and thus an increase in the problem size.

Proposition 1: ([1, 28]) Consider the primal linearly programmable problem:

$$\begin{aligned} \min_{(x, y) \in \mathbb{R}^{n+1}} \quad & e^t x + f^t y \\ \text{s.t., } \quad & Px + Qy \geq b, \quad Sx + Ny = h, \quad x > 0, \end{aligned} \quad (17)$$

is solvable, where $e \in \mathbb{R}^n$, $f \in \mathbb{R}^l$, $P \in \mathbb{R}^{m \times n}$, $Q \in \mathbb{R}^{m \times l}$, $b \in \mathbb{R}^m$, $S \in \mathbb{R}^{k \times n}$, $N \in \mathbb{R}^{k \times l}$ and $h \in \mathbb{R}^k$,

Therefore, the dual penalty optimization problem may be defined as:

$$\begin{aligned} \min_{w, v \in \mathbb{R}^{m+k}} \quad & \phi(-b^t w - h^t v) \\ & + \frac{1}{2} \left(\|P^t w + S^t v - e\|_2^2 + \|Q^t w + N^t v - f\|_2^2 + \|(-u)_+\|_2^2 \right). \end{aligned} \quad (18)$$

Equation (18) is also solvable $\forall \phi > 0$. Furthermore for every $\phi \in (0, \bar{\phi}]$, $\exists \bar{\phi} > 0$ the (w, v) will be a solution of (18) which leads to:

$$x = \frac{1}{\phi} (P^t w + S^t v - e)_+, \quad y = \frac{1}{\phi} (Q^t w + N^t v - f). \quad (19)$$

Now following Proposition 1, the dual penalty optimization problem [1] of (19) may be obtained as:

$$\begin{aligned} \min_{w \in \mathbb{R}^m} \quad & \phi y^t w + \frac{1}{2} \left(\|(\Omega^t w - d_l)_+\|_2^2 + \|(-\Omega^t w - d_l)_+\|_2^2 \right. \\ & \left. + \|(-w - C d_m)_+\|_2^2 + \|(-w - C d_m)_-\|_2^2 + \|(w - C d_m)_-\|_2^2 \right) \end{aligned} \quad (20)$$

where ϕ is the penalty parameter and (20) is solvable for $\phi > 0$. Additionally for any $\bar{\phi} > 0$ there exists:

$$\begin{aligned} s &= \frac{1}{\phi} (\Omega^t w - d_l)_+, \quad t = \frac{1}{\phi} (-\Omega^t w - d_l)_+ \\ s &= \frac{1}{\phi} (\Omega^t w - d_l)_+, \quad t = \frac{1}{\phi} (-\Omega^t w - d_l)_+ \\ p &= \frac{1}{\phi} (-w - C d_m)_+ \text{ and } q = \frac{1}{\phi} (w - C d_m)_+. \end{aligned} \quad (21)$$

The unconstrained minimization problem of (20) can be solved by Newton–Armijo iterative technique.

Algorithm 1: Iterative Newton algorithm with Armijo stepsize

Consider:

```

 $w$  = initial guess such that  $w \in R^m$ 
 $tol$  = level of tolerance
 $iter$  = iterations
 $\max iter$  = maximum iterations
    •  $iter = 0$ 
    • while ( $iter < \max iter$  and  $\|\nabla L(w)\|_2 > tol$ )

 $\nabla^2 L(w) \hat{d} = -\nabla L(w)$ 

/*which is the Armijo step-size and  $\hat{d}$  is the direction vector */
/* Stepsize should be chosen in such a manner that
 $L(w) - L(w + \lambda \hat{d}) \geq -\hat{d} \nabla L(w)' \hat{d}$  and  $\hat{d} \in (0, 1/2)$  */

 $w = w + \lambda \hat{d}$ 
 $iter = iter + 1$ 
end while

```

Here $L(w)$ is the gradient of $L(\cdot)$ expressed by:

$$\nabla L(w) = -\phi y + \Omega (\Omega^t w - d_l)_+ - \Omega (-\Omega^t w - d_l)_+ - (-w - C d_m)_+ + (w - C d_m)_+, \quad (22)$$

which is not differentiable. Hence the second-order derivative of $L(\cdot)$ does not exist. But its “Generalized Hessian” can be formed for $w \in R^m$ as:

$$\begin{aligned} \nabla^2 L(w) &= \nabla^2 L \\ &= (\text{diag} (\Omega^t w - d_l)_* + (-\Omega^t w - d_l)_*) \Omega^t \\ &\quad + \text{diag} (-w - C d_m)_* + (w - C d_m)_* \\ &= \Omega \text{diag} ((|\Omega^t w| - d_l)_*) \Omega^t + \text{diag} (|w| - C d_m)_*, \end{aligned} \quad (23)$$

Equation (23) follows the following form of equality:

$$(\alpha - 1)_* + (-\alpha - 1)_* = (|\alpha| - 1)_* \quad (24)$$

where diag indicates the diagonal matrix. The “Generalized Hessian” can be useful while solving the unconstrained smooth optimization problems and leads to a unique solution.

$$(\nabla^2 L(w^i))(w^{i+1} - w^i) = -\nabla L(w^i). \quad (25)$$

However, $\nabla^2 L(w)$ is positive semi-definite and might get ill-conditioned.

Remark 1 To avoid the ill-conditioning of (25) a very small positive integer $\tau > 0$ is added to (25) and multiplied with an identity matrix I of appropriate dimension. Therefore $\nabla^2 L + \tau I$ is used.

In this work, the optimization problem of (24) is solved using the Newton method without the Armijo step for simplicity. This indicates that w^{i+1} at $(i+1)$ th iteration is obtained by finding the solution of,

$$(\nabla^2 L(w^i) + \tau I)(w^{i+1} - w^i) = -\nabla L(w^i), \quad (26)$$

where $i = 0, 1, \dots$

$$\begin{aligned} &(\Omega \text{diag} ((|\Omega^t w^i| - e_l)_*) \Omega^t + \text{diag} (|w^i| - C e_m)_* + \tau I) \\ &\quad \times (w^{i+1} - w^i) \\ \text{i.e.,} &= -(-\phi y + \Omega (\Omega^t w^i - e_l)_+ - \Omega (-\Omega^t w^i - e_l)_+ \\ &\quad - (-w^i - C e_m)_+ + (w^i - C e_m)_+). \end{aligned} \quad (27)$$

We can determine the value of w by solving the above iterative schemes in (27).

Algorithm 2: 1 norm random vector functional link (1N RVFL)

Consider D is the $m \times n$ dimensional matrix of training examples. $y_i \in \{+1, -1\}$ for $i = 1, \dots, m$, is the class level of D ; and hence $y = (y_1, \dots, y_m)^t$ is the class labels vectors and the diagonal matrix Y is presented by $Y = \text{diag}(y)$. Let A data samples have $+1$ and B data samples have -1 class label. The 1N RVFL can be formulated by the following steps

Step 1. Choose an activation function and define the input parameters.

Step 2. Determine the dual penalty optimization problem expressed in (20).

Step 3. Solve the unconstrained convex minimization problem using the iterative Newton-Armijo method as shown in Algorithm 1.

Step 4. Finally using the solutions as well as using equations (15) and (21) the decision function can be determined.

Step 5. For any new instance $x \in \mathcal{R}^n$ the decision classifier may be expressed as:

$$f(x) = \text{sign}([h(x) \ x]\beta)$$

Simulation and analysis of results

This segment investigates the performance of the 1 norm RVFL model in comparison to ELM, KRR, RVFL, K-RVFL and GLTRVFL for classification problems on some real-world benchmark datasets. All the simulations are performed in MATLAB 2008b environment on a desktop computer of 4 GB of RAM, 64-bit Windows 7 OS, Intel i5 processor with 3.20 GHz speed. No external optimization toolbox was required to solve the optimization problems of the reported models.

Zhang and Suganthan [45] suggested that the Hardlim and sign activation function generally degrade the overall performance of the RVFL algorithm. To select the best activation

function, we have performed experiments on a few real-world datasets using different activation functions for the proposed 1N RVFL, viz., Hardlim, multiquad, radial basis function (RBF), triangular bias (Tri-bas), sigmoid, sine and ReLU. The average ranks are shown in Table 2 and based on that we have picked the best and the second-best activation functions i.e., ReLU, and sine have been tested in the experiments for ELM, RVFL, GLTRVFL and the proposed 1 N RVFL. Let us consider x as an input vector. For this purpose, the two activation functions could be defined as-

- (a) ReLU: $\phi(x) = \max(0, x)$
- (b) Sine: $\phi(x) = \sin(x)$

Table 2 Average rank using different activation functions for 1N RVFL (best average rank is bolded)

Dataset	1N RVFL Hardlim	1N RVFL Multiquad	1N RVFL RBF	1N RVFL Tri-Bas	1N RVFL Sigmoid	1N RVFL Sine	1N RVFL ReLU
Average rank	4.2895	4.9211	4.7632	3.7895	3.9474	2.8947	3.3947

Table 3 Information of the datasets

Dataset	<i>S</i>	<i>N</i>	Dataset	<i>S</i>	<i>N</i>	Dataset	<i>S</i>	<i>N</i>	Dataset	<i>S</i>	<i>N</i>
Cleve	297	13	Splice	3175	60	Card	1004	8	Yeast 1	2968	8
Haberman	306	3	Musk	476	167	Hepatitis	129	17	Yeast 2 vs 8	483	8
Transfusion	748	4	Ecoli 0147 vs 56	332	6	Ecoli 2	336	7			
Votes	435	16	Ecoli 0267 vs 35	224	7	Ecoli 3	336	9			
WDBC	569	30	Glass 015 vs 2	172	10	Vehicle 1	846	7			
WPBC	194	33	Glass 016 vs 2	192	9	New thyroid 1	215	5			
Sonar	208	60	Ecoli4	336	7	Yeast 3	1484	8			

Table 4 Classification accuracies obtained by the classifiers on the real-world dataset with optimum parameters

Dataset (Train × Test)	ELM		KRR		RVFL		K-RVFL		GLTRVFL		1 NRVFL	
	ReLU (<i>L</i>)	Sine (<i>L</i>)	Gaussian (<i>C</i> , μ)	ReLU (<i>C</i> , <i>L</i>)	Sine (<i>C</i> , <i>L</i>)	Gaussian (<i>C</i> , μ)	ReLU (<i>C</i> ₁ = <i>C</i> ₂ , <i>L</i>)	Sine (<i>C</i> ₁ = <i>C</i> ₂ , <i>L</i>)	ReLU (<i>C</i> ₁ = <i>C</i> ₂ , <i>L</i>)	Sine (<i>C</i> ₁ = <i>C</i> ₂ , <i>L</i>)	ReLU (<i>C</i> , <i>L</i>)	Sine (<i>C</i> , <i>L</i>)
Card (77 × 179)	74.1786 (1000)	78.0536 (20)	76.5363 (10 ⁻³ , 2 ⁴)	74.8603 (10 ⁻¹ , 50)	80.5587 (10 ⁻¹ , 20)	74.8603 (10 ¹ , 2 ⁴)	77.5419 (10 ⁻³ , 200)	76.5363 (10 ⁻² , 50)	76.8822 (10 ⁻¹ , 20)	78.994 (10 ⁻³ , 2000)		
Cleve (90 × 207)	78.1111 (20)	82.5556 (20)	80.1932 (10 ⁻² , 2 ³)	79.7101 (10 ⁰ , 100)	80.2899 (10 ⁻¹ , 500)	82.1256 (10 ¹ , 2 ¹)	78.744 (10 ⁻³ , 500)	79.6135 (10 ⁻³ , 20)	79.2271 (10 ⁻² , 2000)	82.3188 (10 ⁻¹ , 2000)		
Ecoli 0147vs56 (100 × 232)	95.5 (1000)	98.804 (20)	96.5517 (10 ⁻⁵ , 2 ⁻⁵)	93.9655 (10 ⁵ , 1000)	96.2931 (10 ³ , 500)	97.4138 (10 ⁰ , 2 ⁵)	96.7241 (10 ⁰ , 100)	96.6379 (10 ¹ , 1000)	96.6379 (10 ² , 200)	97.3276 (10 ⁵ , 2000)		
Ecoli 0267vs35 (68 × 156)	96.2619 (1000)	91.5 (20)	94.8718 (10 ⁻⁵ , 2 ⁻⁵)	94.2308 (10 ⁵ , 1000)	95.5128 (10 ⁵ , 20)	94.8718 (10 ⁰ , 2 ³)	95 (10 ⁰ , 50)	95.5128 (10 ⁰ , 1000)	96.6379 (10 ² , 1000)	95.1282 (10 ² , 2000)		
Ecoli 067vs5 (66 × 154)	91.6905 (20)	94.7143 (20)	98.0519 (10 ¹ , 2 ⁻²)	93.1169 (10 ⁻¹ , 200)	97.5325 (10 ⁴ , 20)	98.0519 (10 ⁰ , 2 ²)	97.2727 (10 ⁰ , 20)	97.4026 (10 ⁰ , 100)	96.1039 (10 ⁴ , 20)	97.6623 (10 ⁴ , 20)		
Ecoli2 (101 × 235)	87.9273 (20)	94.3 (20)	91.9149 (10 ⁰ , 2 ⁻³)	92.3404 (10 ² , 20)	92.3404 (10 ² , 500)	92.3404 (10 ⁰ , 2 ⁴)	93.1064 (10 ⁰ , 50)	92 (10 ³ , 200)	91.3191 (10 ² , 100)	96.426 (10 ² , 200)		
Ecoli3 (101 × 235)	89.1818 (20)	88.881 (20)	91.9149 (10 ⁰ , 2 ⁻³)	92.3404 (10 ⁵ , 200)	92.3404 (10 ² , 500)	92.3404 (10 ⁰ , 2 ⁴)	92.8511 (10 ³ , 50)	92.3404 (10 ⁰ , 200)	91.4894 (10 ² , 200)	93.826 (10 ³ , 20)		
Ecoli4 (101 × 235)	95.7 (1000)	94.7143 (20)	96.1702 (10 ⁻⁴ , 2 ⁰)	94.4681 (10 ⁰ , 1000)	96.8511 (10 ² , 1000)	94.8936 (10 ⁰ , 2 ³)	96.3404 (10 ⁰ , 100)	96.8511 (10 ¹ , 1000)	95.9149 (10 ³ , 1000)	96.4255 (10 ⁵ , 100)		
Glass 015 vs 2 (52 × 120)	83.6333 (1000)	88.595 (20)	75 (10 ⁻³ , 2 ⁻¹)	90.8333 (10 ⁻¹ , 100)	90.8333 (10 ⁻¹ , 200)	90.8333 (10 ⁻⁵ , 2 ⁻⁵)	90.8333 (10 ⁻⁴ , 20)	90.8333 (10 ⁻⁴ , 1000)	90.8333 (10 ¹ , 50)	90.8333 (10 ¹ , 2000)		
Glass 016 vs 2 (58 × 134)	87.5333 (20)	76.5 (20)	91.0448 (10 ¹ , 2 ⁻¹)	90.2985 (10 ⁵ , 1000)	90.2985 (10 ² , 20)	91.0448 (10 ⁻⁵ , 2 ⁻⁵)	88.5075 (10 ⁻⁵ , 50)	89.7015 (10 ⁻² , 20)	91.0448 (10 ¹ , 50)	91.045 (10 ² , 20)		
Haberman (92 × 214)	71.2222 (20)	69.9222 (50)	50.9346 (10 ⁻¹ , 2 ¹)	35.9813 (10 ⁵ , 1000)	46.3551 (10 ⁰ , 50)	51.4019 (10 ⁰ , 2 ⁰)	69.3458 (10 ⁻⁵ , 20)	72.8972 (10 ⁻⁴ , 50)	72.4299 (10 ⁻² , 20)	72.4299 (10 ¹ , 100)		
Hepatitis (39 × 90)	87.25 (20)	85.75 (100)	75.5556 (10 ⁻¹ , 2 ⁰)	75.5556 (10 ⁻¹ , 200)	75.5556 (10 ⁻¹ , 200)	75.5556 (10 ⁻⁵ , 2 ⁻⁵)	75.3333 (10 ⁻² , 20)	74.6667 (10 ⁻² , 20)	75.556 (10 ⁰ , 2000)	75.9401 (10 ¹ , 20)		

Table 4 continued

Dataset (Train × Test)	ELM		KRR	RVFL		K-RVFL (C, μ)	GLTRVFL		1 NRVFL	
	ReLU (L)	Sine (L)		ReLU (C, L)	Sine (C, L)		ReLU ($C_1 = C_2, L$)	Sine ($C_1 = C_2, L$)	ReLU (C, L)	Sine (C, L)
Musk (143×333)	95.5762 (1000)	94.9238 (1000)	97.8979 (10^{-2} , 2^1)	97.958 (10^{-2} , 1000)	96.9369 (10^{-2} , 1000)	99.0991 (10^0 , 2^2)	97.4775 (10^{-3} , 500)	97.7778 (10^{-4} , 1000)	96.757 (10^{-4} , 100)	98.078 (10^{-4} , 200)
New thyroid1 (65×150)	97.5952 (500)	99.571 (20)	98.6667 (10^0 , 2^{-2})	94 (10^5 , 1000)	97.0667 (10^3 , 500)	98 (10^{-1} , 2^0)	98.2667 (10^5 , 1000)	97.8667 (10^2 , 1000)	98.9333 (10^4 , 2000)	99.876 (10^5 , 20)
Sonar (63×145)	72.033 (1000)	72 (1000)	79.3103 (10^0 , 2^0)	71.5862 (10^{-2} , 1000)	73.6552 (10^2 , 500)	77.2414 (10^0 , 2^{-1})	74.4828 (10^0 , 1000)	72 (10^1 , 200)	70.3448 (10^3 , 2000)	76.2759 (10^0 , 2000)
Splice (953×2222)	75.236 (200)	78.625 (200)	89.649 (10^{-1} , 2^1)	85.3375 (10^{-1} , 50)	86.5257 (10^{-2} , 1000)	86.7687 (10^0 , 2^5)	84.4734 (10^0 , 20)	82.4392 (10^{-4} , 1000)	85.2565 (10^{-1} , 2000)	87.0387 (10^{-2} , 2000)
Vehicle1 (254×592)	76.3154 (1000)	92.091 (50)	79.3919 (10^{-1} , 2^{-1})	77.8716 (10^{-1} , 500)	83.1419 (10^2 , 500)	81.5878 (10^0 , 2^5)	78.8851 (10^{-2} , 1000)	81.9257 (10^5 , 200)	77.3311 (10^1 , 2000)	96.712 (10^2 , 100)
Votes (131×304)	96.0769 (500)	95.3901 (20)	95.3947 (10^{-1} , 2^1)	94.4079 (10^5 , 1000)	94.7368 (10^0 , 1000)	95.0658 (10^1 , 2^{-2})	95.4605 (10^{-3} , 1000)	94.6711 (10^{-3} , 1000)	94.8026 (10^0 , 500)	94.8684 (10^0 , 1000)
WDBC (171×398)	91.9183 (50)	94.1405 (20)	94.7236 (10^{-4} , 2^5)	94.6734 (10^0 , 50)	96.4824 (10^{-1} , 500)	94.4724 (10^1 , 2^3)	94.1709 (10^0 , 20)	97.0854 (10^{-1} , 100)	96.7337 (10^0 , 2000)	96.4824 (10^0 , 100)
WPBC (59×135)	70.8667 (1000)	76.7 (1000)	76.2963 (10^1 , 2^0)	76.4444 (10^{-2} , 200)	60 (10^3 , 200)	76.2963 (10^{-5} , 2^{-5})	75.5556 (10^{-3} , 20)	74.8148 (10^{-3} , 500)	76.2963 (10^0 , 100)	76.2963 (10^0 , 50)
Yeast 1 (891×2077)	76.3785 (50)	93.752 (20)	77.8045 (10^1 , 2^{-3})	75.5416 (10^{-1} , 200)	77.2557 (10^3 , 1000)	78.5267 (10^{-1} , 2^{-2})	76.2061 (10^{-1} , 200)	76.2446 (10^1 , 1000)	77.4386 (10^1 , 2000)	94.181 (10^4 , 200)
Yeast 2 vs 8 (446×1038)	97.1762 (20)	96.7571 (20)	97.6331 (10^2 , 2^{-2})	97.6331 (10^{-2} , 1000)	95.677 (10^{-1} , 100)	97.6331 (10^0 , 2^{-1})	97.8698 (10^{-1} , 20)	97.929 (10^0 , 20)	97.929 (10^0 , 20)	97.929 (10^1 , 2000)
Yeast 3 (446×1038)	93.1702 (50)	90.5 (20)	94.027 (10^1 , 2^{-3})	94.027 (10^5 , 1000)	94.6821 (10^2 , 500)	94.7013 (10^{-1} , 2^0)	94.2582 (10^0 , 1000)	94.9711 (10^0 , 500)	94.104 (10^5 , 20)	95.467 (10^4 , 2000)
Win/ Tie/ Loss	20/0/3	18/0/5	17/2/4	21/1/1	18/2/3	15/3/5	21/1/1	17/2/4	17/5/1	–

Table 5 Average ranks of ELM, KRR, RVFL, K-RVFL, GLTRVFL and 1N RVFL based on the accuracy

Dataset	ELM		KRR	RVFL		K-RVFL	GLTRVFL		1 N RVFL	
	ReLU	Sine		ReLU	Sine		ReLU	Sine	ReLU	Sine
Card	10	3	6.5	8.5	1	8.5	4	6.5	5	2
Cleve	10	1	5	6	4	3	9	7	8	2
Ecoli-0-1-4-7_vs_5-6	9	1	7	10	8	2	4	5.5	5.5	3
Ecoli-0-2-6-7_vs_3-5	1	10	6.5	9	2.5	6.5	5	2.5	8	4
Ecoli-0-6-7_vs_5	10	8	1.5	9	4	1.5	6	5	7	3
Ecoli2	10	2	8	5	5	5	3	7	9	1
Ecoli3	9	10	7	4.5	4.5	4.5	2	4.5	8	1
Ecoli4	7	9	5	10	1.5	8	4	1.5	6	3
Glass-0-1-5_vs_2	9	8	10	4	4	4	4	4	4	4
Glass-0-1-6_vs_2	9	10	2.5	5.5	5.5	2.5	8	7	2.5	2.5
Haberman	4	5	8	10	9	7	6	1	2.5	2.5
Hepatitis	1	2	6.5	6.5	6.5	6.5	9	10	4	3
Musk	9	10	4	3	7	1	6	5	8	2
New-thyroid1	8	2	4	10	9	6	5	7	3	1
Sonar	6	7.5	1	9	5	2	4	7.5	10	3
Splice	10	9	1	5	4	3	7	8	6	2
Vehicle 1	10	2	6	8	3	5	7	4	9	1
Votes	1	4	3	10	8	5	2	9	7	6
WDPC	10	9	5	6	3.5	7	8	1	2	3.5
WPBC	9	1	4.5	2	10	4.5	7	8	4.5	4.5
Yeast1	7	2	4	10	6	3	9	8	5	1
Yeast2vs8	8	9	6	6	10	6	4	2	2	2
Yeast3	9	10	7.5	7.5	4	3	5	2	6	1
Average rank	7.6522	5.8478	5.1957	7.1522	5.4348	4.5435	5.5652	5.3478	5.7391	2.5217

Fig. 1 Statistical comparison of classifiers against each other based on the Nemenyi test

where $\phi(x)$ represents the output function for the input sample x .

The tests were performed using tenfold cross-validation technique. Here, the sample is split into 10 equal subsamples. One portion is used for testing each of the subsamples and

the other portion is used for training. This method runs 10 times until all components are trained for at least one time [4]. For computational convenience, the input data is split into two parts, where 30% of the data are training data and the other 70% are testing data. To validate the efficacy of

Table 6 Training time (in seconds) for the reported models

Dataset	ELM		KRR		RVFL		K-RVFL		GLTRVFL		INRVFL	
	ReLU	Sine			ReLU	Sine			ReLU	Sine	ReLU	Sine
Card	0.0056	0.0056	0.0013		0.0002	0.00002	0.0006		0.0075	0.0014	0.017	0.3462
Cleve	0.0044	0.0039	0.0011		0.0004	0.00101	0.0008		0.0011	0.0007	0.5555	0.7039
Ecoli-0-1-4-7_vs_5-6	0.1857	0.0044	0.0023		0.0022	0.00104	0.0053		0.0009	0.0009	0.8966	4.3389
Ecoli-0-2-6-7_vs_3-5	0.011	0.0026	0.0019		0.0014	0.0001	0.0031		0.0031	0.0044	0.5332	4.3389
Ecoli-0-6-7_vs_5	0.0031	0.0029	0.0012		0.0004	0.0002	0.0002		0.0005	0.0005	0.0287	0.0267
Ecoli2	0.0054	0.0052	0.0032		0.0002	0.0011	0.0009		0.0007	0.0008	0.0763	0.0982
Ecoli3	0.0055	0.0046	0.0022		0.0007	0.0011	0.0007		0.001	0.0009	0.1232	0.1405
Ecoli4	0.0159	0.0036	0.0018		0.0021	0.0015	0.0007		0.0066	0.0027	1.2145	0.1916
Glass-0-1-5_vs_2	0.0871	0.003	0.0005		0.0003	0.0005	0.0003		0.0005	0.0004	0.0052	0.0053
Glass-0-1-6_vs_2	0.0029	0.0032	0.0006		0.0014	0.0003	0.0003		0.0006	0.0005	0.004	0.0051
Haberman	0.0043	0.0096	0.001		0.0022	0.0004	0.0009		0.0013	0.0006	0.0047	0.0046
Hepatitis	0.0056	0.0056	0.0007		0.0036	0.0002	0.0003		0.0012	0.0007	0.3291	0.0039
Musk	0.0085	0.0079	0.0217		0.0062	0.0005	0.0097		0.0026	0.0026	0.2199	0.1607
New-thyroid1	0.059	0.0035	0.001		0.0013	0.0007	0.0003		0.0007	0.0007	0.9217	3.0986
Sonar	0.0114	0.013	0.0023		0.0019	0.0008	0.0008		0.0011	0.0012	0.8415	0.8258
Splice	0.0367	0.0399	0.2613		0.0033	0.0614	0.0829		0.0333	0.0332	12.487	15.0734
Vehicle 1	0.0442	0.0313	0.0182		0.005	0.0035	0.0037		0.0033	0.0029	0.1001	0.0704
Votes	0.0121	0.0064	0.0006		0.0039	0.0027	0.0018		0.0009	0.0008	0.2059	0.2293
WDPC	0.0205	0.0087	0.0071		0.0007	0.0023	0.0024		0.0011	0.0009	1.4999	1.5293
WPBC	0.0999	0.1048	0.0007		0.0006	0.0005	0.0008		0.0007	0.0007	0.0097	0.0091
Yeast1	0.0556	0.0347	0.0932		0.0094	0.0472	0.0207		0.0343	0.0379	11.1873	16.2482
Yeast2vs8	0.0056	0.0044	0.0036		0.0045	0.0007	0.0016		0.0015	0.0019	0.0154	0.02455
Yeast3	0.0411	0.0196	0.0240		0.025	0.0083	0.0072		0.0145	0.0137	1.001	1.0041

Table 7 Testing time (in seconds) for the reported models

Dataset	ELM		KRR		RVFL		K-RVFL		GLTRVFL		INRVFL	
	ReLU	Sine			ReLU	Sine			ReLU	Sine	ReLU	Sine
Card	0.0004	0.0005	0.001		0.0001	0.0003	0.0014		0.0098	0.0019	0.0201	0.0301
Cleve	0.0003	0.0034	0.0042		0.0004	0.0009	0.0045		0.3686	0.0043	0.5555	0.7038
Ecoli-0-1-4-7_vs_5-6	0.003	0.0037	0.0013		0.0022	0.0014	0.0019		0.0079	0.3132	0.8966	4.3389
Ecoli-0-2-6-7_vs_3-5	0.002	0.0027	0.0009		0.0014	0.0003	0.0009		0.0036	0.0056	0.0329	0.0585
Ecoli-0-6-7_vs_5	0.0002	0.0026	0.0008		0.0004	0.0002	0.0008		0.0018	0.0037	0.0287	0.0267
Ecoli2	0.0004	0.0037	0.0014		0.0002	0.0028	0.0024		0.0111	0.3004	0.0763	0.0982
Ecoli3	0.0003	0.0037	0.0013		0.0007	0.0026	0.0017		0.0147	0.0128	0.1232	0.1405
Ecoli4	0.0028	0.0037	0.0015		0.0024	0.0027	0.0022		0.0074	0.282	0.0308	0.0425
Glass-0-1-5_vs_2	0.0022	0.0022	0.001		0.0003	0.0005	0.0008		0.0013	0.0033	0.0052	0.0053
Glass-0-1-6_vs_2	0.0002	0.0023	0.0006		0.0014	0.0006	0.0008		0.0041	0.0022	0.004	0.0051
Haberman	0.0002	0.0095	0.0016		0.0022	0.0006	0.0024		0.0016	0.002	0.0047	0.0046
Hepatitis	0.0005	0.0146	0.0004		0.0002	0.0005	0.0004		0.0492	0.3273	0.0124	0.0159
Musk	0.0005	0.0006	0.0186		0.0003	0.0006	0.0189		0.1013	0.0644	0.2199	0.1607
New-thyroid1	0.0015	0.0025	0.0008		0.0013	0.0005	0.001		0.2597	0.0743	0.9217	3.0986
Sonar	0.0027	0.1182	0.0029		0.0019	0.0023	0.0026		0.0147	0.0076	0.8415	0.8258
Splice	0.0066	0.3765	0.197		0.0033	0.0998	0.2098		0.1036	0.5657	12.4871	15.0734
Vehicle 1	0.0068	0.0247	0.0077		0.005	0.0012	0.0105		0.2954	0.0233	0.1001	0.0704
Votes	0.0017	0.0057	0.0031		0.0039	0.0023	0.0056		0.2796	0.0561	0.2059	0.2293
WDPC	0.0006	0.0066	0.0055		0.0007	0.0017	0.0068		0.2636	0.0048	1.5	1.5293
WPBC	0.0022	0.1072	0.002		0.0005	0.0019	0.0032		0.0142	0.0024	0.0097	0.0091
Yeast1	0.0014	0.0313	0.0554		0.0094	0.0852	0.0653		0.2265	0.7475	0.1981	0.2716
Yeast2vs8	0.0004	0.0058	0.0036		0.0045	0.0008	0.0041		0.0118	0.0205	0.0154	0.0246
Yeast3	0.0011	0.0168	0.0141		0.025	0.0283	0.0173		0.4989	0.209	1.0097	1.0041

Fig. 2 Parameter insensitivity performance on user-specified parameters, (C, L) of IN RVFL ReLU and sine models for Haberman dataset

the proposed 1 norm RVFL model, the performance of this algorithm was compared with the ELM, KRR, RVFL, K-RVFL and GLTRVFL models on some interesting real-world benchmark datasets.

Since the large value of ELM leads to an increase in computational time [1], hence, for ELM, KRR, RVFL and GLTRVFL the optimum values of the parameter L is considered from a set of $\{20, 50, 100, 200, 500, 1000\}$. For, KRR, RVFL, K-RVFL and GLTRVFL, the optimum parameters of C are obtained from $\{10^{-5}, \dots, 10^5\}$ respectively. The Gaussian kernel is selected while implementing the KRR. The kernel parameter μ is chosen from $\{2^{-5}, \dots, 2^5\}$. In the proposed 1 N RVFL, the optimum values for the two parameters C and L , are chosen from the range of $\{10^{-5}, \dots, 10^5\}$ and $\{20, 50, 100, 200, 500, 1000, 2000\}$, respectively. The statistics of the datasets that are considered during the experiment are tabulated in Table 3, where S indicates the length and N is the total number of attributes.

All the experimental datasets are collected from UCI machine learning databases [12]. The numerical experiments on various datasets are performed after the normalization of the data. The raw data is normalized by considering

$\bar{r}_{ij} = \frac{r_{ij} - r_j^{\min}}{r_j^{\max} - r_j^{\min}}$, where $r_j^{\max} = \max_{i=1, \dots, m}(x_{ij})$ and $r_j^{\min} = \min_{i=1, \dots, m}(x_{ij})$ denotes the maximum and minimum values respectively of the j th attribute over all of the input data r_i . $\bar{r}_{ij}$ represents the normalized value of r_{ij} .

For all the datasets, the attributes, the number of training and test samples and the optimum parameters are obtained using the tenfold cross-validation method. The total number of training and testing samples, the optimum values of the parameters and the classification accuracies of the models are shown in Table 4. Comparable or better performance indicates the efficacy and applicability of the proposed model. Additionally, the ranks based on classification accuracy for each dataset are exhibited in Table 5 for each reported classifier.

Friedman test with Nemenyi statistics for classifier comparison

To compare the performance of the reported algorithms with the proposed algorithm, we perform a non-parametric Friedman test from Table 5 [9] where the average ranks of the models are tabulated. The lowest average rank of our proposed 1N RVFL Sine reflects the efficiency of the model.

Fig. 3 Parameter insensitivity performance of 1N RVFL ReLU and sine models on user-specified parameters, (C, L) for Vehicle1 dataset

The Friedman test for the null hypothesis may be obtained as:

$$\chi_F^2 = \frac{12 \times 23}{10 \times (10 + 1)} \left[\left(7.6522^2 + 5.8478^2 + 5.1957^2 + 7.1522^2 + 5.4348^2 + 4.543^2 + 5.5652^2 + 5.3478^2 + 5.7391^2 + 2.5217^2 \right) - \frac{10 \times (10 + 1)^2}{4} \right] = 43.7819,$$

$$F_F = \frac{(23 - 1) \times 43.7819}{23(10 - 1) - 43.7819} = 5.9013.$$

F_F is distributed according to F -distribution with $((10 - 1), (10 - 1) \times (23 - 1)) = (9, 198)$ degrees of freedom. For the level of significance, $\alpha = 0.10$ the critical value (CV) for $F(9, 198)$ is 1.927 which is less than the F_F . Hence, the null hypothesis can be rejected. Now, let us perform the Nemenyi test to compare the methods. Here, the critical difference (CD) is:

$$CD = 2.92 \times \sqrt{\frac{10 \times (10 + 1)}{6 \times 23}} = 2.607 \text{ for } q_\alpha = 0.10.$$

Figure 1 shows the statistical comparison of classifiers against each other based on the Nemenyi test. Groups of classifiers that are not significantly different (at $\alpha = 0.10$) are connected. One can notice from the figure that the 1N RVFL sine model is significantly better than ELM ReLU, ELM sine, KRR, RVFL ReLU, GLTRVFL ReLU, GLTRVFL sine and 1N RVFL ReLU models. However, despite showing a better average rank than K-RVFL, 1N RVFL sine is not significantly different from K-RVFL.

The training time and testing time (in seconds) of the reported models are shown in Tables 6 and 7 respectively for the models. It can be observed that 1N RVFL is that it is computationally less efficient than RVFL despite showing better generalization performance.

Fig. 4 Parameter insensitivity performance of 1NRVFL ReLU and sine models on user-specified parameters, (C, L) for Yeast3 dataset

Win/Tie/Loss test

The statistical analysis approach win/tie/loss [10] for our best-proposed model, i.e., 1N RVFL sine is used to further validate the efficacy of the proposed 1N RVFL sine model. The outcomes are exhibited in the last row of Table 4. For example, the second column shows the comparison between 1N RVFL sine and ELM ReLU. It can be noted that 1N RVFL wins in 20 cases, tie in no case and loss in 3 cases compared to ELM ReLU. Similarly, the third column shows the comparison between 1N RVFL sine and ELM sine. It can be noticed that 1N RVFL wins in 18 cases, tie in no case and loss in 5 cases compared to ELM ReLU. Similar conclusions can be derived from the other columns. As it can be observed from the last row of Table 3, the proposed method has the best classification accuracy in most situations, indicating that 1N RVFL sine outperforms other algorithms.

The parameter insensitivity plots of the proposed models are presented in Figs. 2, 3 and 4 for Haberman, Vehicle 1 and Yeast3 datasets. One can notice from Figs. 2, 3 and 4

that the proposed models are not very sensitive towards the user-defined parameters C and L .

Moreover, to reveal the sparseness of the proposed model, the average number of “actually” contributing nodes are portrayed in Fig. 5. A lower number of non-zero components indicates that the model is sparse. Ecoli2, New thyroid1 and Haberman datasets are used to see whether the proposed 1NRVFL solution method results in the least number of hidden nodes when determining the decision function. Hence, the degree of sparseness for each pair of (C, L) is determined. It can be observed from Fig. 5 that the 1N RVFL always leads to sparse solutions.

Conclusion

In this work, a novel 1N RVFL has been proposed for solving binary classification problems. The solutions are obtained by solving the dual using Newton-Armijo technique as an unconstrained minimization problem. The basic advantage of the proposed 1N RVFL is that it produces many coefficients

Fig. 5 Number of actually contributing nodes with user-defined parameters for 1N RVFL with ReLU and sine additive nodes for **a** Ecoli2. **b** New thyroid1 and **c** Haberman datasets

(c)

Fig. 5 continued

with zero values, which results in a sparse output. The 1 norm RVFL is a robust model as it only considers the absolute value and can ignore the extreme values. This leads to an increase in the outliers cost exponentially. Extensive experiments on several classification datasets using the reported models portray that the proposed 1 norm RVFL shows better performance compared to ELM, KRR, RVFL, K-RVFL and GLTRVFL. The good generalization ability of the proposed model implies the usability and efficiency of the same. 1N RVFL is useful for classification problems with very high dimensional input spaces. However, the major limitation of 1N RVFL is that it is computationally less efficient than RVFL. Future work of this model will be based on developing this model for solving the multiclass classification problems using the one-versus-rest or the one-versus-one procedure. This can be fruitful for various real-life classification problems such as face classification, character recognition, plant species recognition and others.

Declarations

Conflicts of interest The authors declare that they have no conflicts of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adap-

tation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Balasundaram S, Gupta D (2014) 1-Norm extreme learning machine for regression and multiclass classification using Newton method. *Neurocomputing* 128:4–14
- Balasundaram S, Gupta D (2016) On optimization based extreme learning machine in primal for regression and classification by functional iterative method. *Int J Mach Learn Cybern* 7(5):707–728
- Borah P, Gupta D (2019) Unconstrained convex minimization based implicit Lagrangian twin random vector Functional-link networks for binary classification (ULTRVFLC). *Appl Soft Comput* 81:105534
- Brownlee J (2018) A gentle introduction to k-fold cross-validation. <https://machinelearningmastery.com/k-fold-cross-validation/>. Accessed 22 June 2021
- Cao F, Ye H, Wang D (2015) A probabilistic learning algorithm for robust modeling using neural networks with random weights. *Inf Sci* 313:62–78

6. Chang X, Lin SB, Zhou DX (2017) Distributed semi-supervised learning with kernel ridge regression. *J Mach Learn Res* 18(1):1493–1514
7. Cortes C, Vapnik V (1995) Support-vector networks. *Mach Learn* 20(3):273–297
8. Dai W, Chen Q, Chu F, Ma X, Chai T (2017) Robust regularized random vector functional link network and its industrial application. *IEEE Access* 5:16162–16172
9. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res* 7:1–30
10. Dixon WJ, Mood AM (1946) The statistical sign test. *J Am Stat Assoc* 41(236):557–566
11. Drucker H, Wu D, Vapnik VN (1999) Support vector machines for spam categorization. *IEEE Trans Neural Netw* 10(5):1048–1054
12. Dua D, Graff C (2019) UCI machine learning repository, 2017. <http://archive.ics.uci.edu/ml> 7(1). Accessed 12 July 2019
13. Floyd S, Warmuth M (1995) Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Mach Learn* 21(3):269–304
14. Gupta D, Hazarika BB, Berlin M (2020) Robust regularized extreme learning machine with asymmetric Huber loss function. *Neural Comput Appl*. <https://doi.org/10.1007/s00521-020-04741-w>
15. Gupta U, Gupta D (2021) Regularized based implicit Lagrangian twin extreme learning machine in primal for pattern classification. *Int J Mach Learn Cybern* 12(5):1311–1342
16. Hazarika BB, Gupta D (2020) Modelling and forecasting of COVID-19 spread using wavelet-coupled random vector functional link networks. *Appl Soft Comput* 96:106626
17. Hazarika BB, Gupta D (2021) Density-weighted support vector machines for binary class imbalance learning. *Neural Comput Appl* 33(9):4243–4261
18. Hazarika BB, Gupta D, Berlin M (2020) Modeling suspended sediment load in a river using extreme learning machine and twin support vector regression with wavelet conjunction. *Environ Earth Sci* 79:234
19. Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. *Neural Netw* 2(5):359–366
20. Huang GB, Chen L, Siew CK (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Trans Neural Netw* 17(4):879–892
21. Huang GB, Zhu QY, Siew CK (2005) Extreme learning machine: a new learning scheme of feedforward neural networks. *Neural Netw* 2:985–990
22. Igel'nik B, Pao YH (1995) Stochastic choice of basis functions in adaptive function approximation and the functional-link net. *IEEE Trans Neural Netw* 6(6):1320–1329
23. Katuwal R, Suganthan PN (2019) Stacked autoencoder based deep random vector functional link neural network for classification. *Appl Soft Comput* 85:105854
24. Leshno M, Lin VY, Pinkus A, Schocken S (1993) Multilayer feed-forward networks with a nonpolynomial activation function can approximate any function. *Neural Netw* 6(6):861–867
25. Li W, Dai D, Tan M, Xu D, Van Gool L (2016) Fast algorithms for linear and kernel svm+. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp 2258–2266
26. Li R, Wang X, Lei L, Song Y (2018) L_{21} -norm based loss function and regularization extreme learning machine. *IEEE Access* 7:6575–6586
27. Liu Q, He Q, Shi Z (2008) Extreme support vector machine classifier. In: *Pacific-Asia conference on knowledge discovery and data mining*, vol 1. Springer, Berlin, Heidelberg, pp 222–233
28. Mangasarian OL (2006) Exact 1-norm support vector machines via unconstrained convex differentiable minimization. *J Mach Learn Res* 7:1517–1530
29. Naik J, Satapathy P, Dash PK (2018) Short-term wind speed and wind power prediction using hybrid empirical mode decomposition and kernel ridge regression. *Appl Soft Comput* 70:1167–1188
30. Pao YH, Takefuji Y (1992) Functional-link net computing: theory, system architecture, and functionalities. *Computer* 25(5):76–79
31. Pao YH, Park GH, Sobajic DJ (1994) Learning and generalization characteristics of the random vector functional-link net. *Neurocomputing* 6(2):163–180
32. Pao YH, Phillips SM, Sobajic DJ (1992) Neural-net computing and the intelligent control of systems. *Int J Control* 56(2):263–289
33. Park J, Sandberg IW (1991) Universal approximation using radial-basis-function networks. *Neural Comput* 3(2):246–257
34. Peng Y, Wang S, Long X, Lu BL (2015) Discriminative graph regularized extreme learning machine and its application to face recognition. *Neurocomputing* 149:340–353
35. Rakesh K, Suganthan PN (2017) An ensemble of kernel ridge regression for multi-class classification. *Proc Comput Sci* 108:375–383
36. Saunders C, Gammernan A, Vovk V (1998) Ridge regression learning algorithm in dual variables. In: *Proceedings of the 15th international conference on machine learning*
37. Scardapane S, Wang D (2017) Randomness in neural networks: an overview. *Wiley Interdiscipl Rev Data Min Knowl Discov* 7(2):e1200
38. Sun Y, Li B, Yuan Y, Bi X, Zhao X, Wang G (2019) Big graph classification frameworks based on extreme learning machine. *Neurocomputing* 330:317–327
39. Vapnik V, Izmailov R (2015) Learning using privileged information: similarity control and knowledge transfer. *J Mach Learn Res* 16(1):2023–2049
40. Vapnik V, Vashist A (2009) A new learning paradigm: learning using privileged information. *Neural Netw* 22(5–6):544–557
41. Wang LP, Wan CR (2008) Comments on "The extreme learning machine". *IEEE Trans Neural Netw* 19(8):1494–1495
42. Wu XH, Zhao PW (2020) Predicting nuclear masses with the kernel ridge regression. *Phys Rev C* 101(5):051301
43. Xu KK, Li HX, Yang HD (2017) Kernel-based random vector functional-link network for fast learning of spatiotemporal dynamic processes. *IEEE Trans Syst Man Cybern Syst* 49(5):1016–1026
44. Zhang L, Zhou W (2010) On the sparseness of 1-norm support vector machines. *Neural Netw* 23(3):373–385
45. Zhang L, Suganthan PN (2016) A comprehensive evaluation of random vector functional link networks. *Inf Sci* 367:1094–1105
46. Zhang Y, Duchi J, Wainwright M (2015) Divide and conquer kernel ridge regression: a distributed algorithm with minimax optimal rates. *J Mach Learn Res* 16(1):3299–3340
47. Zhang L, Suganthan PN (2017) Benchmarking ensemble classifiers with novel co-trained kernel ridge regression and random vector functional link ensembles [research frontier]. *IEEE Comput Intell Mag* 12(4):61–72
48. Zhang PB, Yang ZX (2020) A new learning paradigm for random vector functional-link network: RVFL+. *Neural Networks* 122:94–105

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.