OPT-RNN-DBSVM: OPTimal Recurrent Neural Network and Density-Based Support Vector Machine

El Moutaouakil, K, El Ouissari, A, Olaru, A, Palade, V & Ciorei, M

DOI 10.3390/math11163555
ISSN 2227-7390

Publisher: MDPI

mathematics

MDPI

Article

OPT-RNN-DBSVM: OPTimal Recurrent Neural Network and Density-Based Support Vector Machine

Karim El Moutaouakil [1,*], Abdellatif El Ouissari [1], Adrian Olaru [2,*], Vasile Palade [3,*] and Mihaela Ciorei [3]

[1] Engineering Science Laboratory, Taza Multidisciplinary Faculty, Sidi Mohamed Ben Abdellah University, Fez 30000, Morocco; abdellatif.elouissari@usmba.ac.ma
[2] Centre for Computational Science and Mathematical Modelling, Coventry University, Priory Road, Coventry CV1 5FB, UK
[3] Department of Robotics and Production System, University Politehnica of Bucharest, 060042 Bucharest, Romania; mihaela.ciorei89@gmail.com
* Correspondence: karim.elmoutaouakil@usmba.ac.ma (K.E.M.); adrian.olaru2301@upb.ro (A.O.); vasile.palade@coventry.ac.uk (V.P.)

Abstract: When implementing SVMs, two major problems are encountered: (a) the number of local minima of dual-SVM increases exponentially with the number of samples and (b) the computer storage memory required for a regular quadratic programming solver increases exponentially as the problem size expands. The Kernel-Adatron family of algorithms, gaining attention recently, has allowed us to handle very large classification and regression problems. However, these methods treat different types of samples (i.e., noise, border, and core) in the same manner, which makes these algorithms search in unpromising areas and increases the number of iterations as well. This paper introduces a hybrid method to overcome such shortcomings, called the Optimal Recurrent Neural Network and Density-Based Support Vector Machine (Opt-RNN-DBSVM). This method consists of four steps: (a) the characterization of different samples, (b) the elimination of samples with a low probability of being a support vector, (c) the construction of an appropriate recurrent neural network to solve the dual-DBSVM based on an original energy function, and (d) finding the solution to the system of differential equations that govern the dynamics of the RNN, using the Euler–Cauchy method involving an optimal time step. Density-based preprocessing reduces the number of local minima in the dual-SVM. The RNN's recurring architecture avoids the need to explore recently visited areas. With the optimal time step, the search moves from the current vectors to the best neighboring support vectors. It is demonstrated that RNN-SVM converges to feasible support vectors and Opt-RNN-DBSVM has very low time complexity compared to the RNN-SVM with a constant time step and the Kernel-Adatron algorithm–SVM. Several classification performance measures are used to compare Opt-RNN-DBSVM with different classification methods and the results obtained show the good performance of the proposed method.

Keywords: Recurrent Neural Network (RNN); Density-Based Algorithm; Support Vector Machine (SVM); Kernel-Adatron algorithm (KA); Euler–Cauchy algorithm

MSC: 90C20; 90C29; 90C90; 93E20

Citation: El Moutaouakil, K.; El Ouissari, A.; Olaru, A.; Palade, V.; Ciorei, M. OPT-RNN-DBSVM: OPTimal Recurrent Neural Network and Density-Based Support Vector Machine. *Mathematics* **2023**, *11*, 3555. https://doi.org/10.3390/math11163555

Academic Editor: Leimin Wang

Received: 3 July 2023
Revised: 31 July 2023
Accepted: 14 August 2023
Published: 17 August 2023

1. Introduction

Many classification methods have been proposed in the literature and in a vast array of applications, among them a popular approach called support vector machine based on quadratic programming (QP) [1–3]. The difficulty in the implementation of SVMs on massive datasets lies in the fact that the quantity of storage memory required for a regular QP solver increases by an exponential magnitude as the problem size expands.

This paper introduces a new type of SVM that implements a preprocessing filter and a recurrent neural network, called the Optimal Recurrent Neural Network and Density-Based Support Vector Machine (Opt-RNN-DBSVM).

SVM approaches are based on the existence of a linear separator, which can be obtained by transforming the data in a higher-dimensional space through appropriate kernel functions. Among all possible hyperplanes, the SVM searches for the one with the most confident separation margin for good generalization. This issue takes the form of a nonlinear constrained optimization problem that is usually handled using optimization methods. Thanks to the Kuhen–Tuker conditions [4], all these methods transform the primal mathematical model into the dual version and use optimization methods to find the support vectors on which the optimal margin is built. Unfortunately, the complexity in time and memory grows exponentially with the size of the datasets; in addition, the number of local minima grows too, which influences the location of the separation margin and the quality of the predictions.

A primary area of research in learning from empirical data through support vector machines (SVMs) and addressing classification and regression issues is the development of incremental learning schemes when the size of the training dataset is massive [5]. Out of many possible candidates, avoiding the usage of regular quadratic programming (QP) solvers, the two learning methods gaining attention recently are iterative single-data algorithms (ISDA) and sequential minimal optimization (SMO) [6–9]. ISDAs operate from a single sample at hand (pattern-based learning) towards the best fit solution. The Kernel-Adatron (KA) is the primary ISDA for SVMs, using kernel functions to map data to the high-dimensional character space of SVMs [10] and conducting Adatron [11] processing in the character space. Platt's SMO algorithm is an outlier among the so-called decomposition approaches introduced in [12,13], operating on a two-sample workset of samples at a time. Because the decision for the two-point workset may be determined analytically, the SMO does not require the involvement of standard QP solvers. Due to it being analytically driven, the SMO has been especially popular and is the most commonly utilized, analyzed, and further developed approach. Meanwhile, KA, while yielding somewhat comparable performance (accuracy and computational time) in resolving classification issues, has not gained as much traction. The reason for this is twofold. First, until recently [14], KA appeared to be restricted to classification tasks; second, it lacks the qualities of a robust theoretical framework. KA employs a gradient ascent procedure, and this fact also may have caused some researchers to be suspicious of the challenges posed by gradient ascent techniques in the presence of a perhaps ill-conditioned core array. In [15], lacking bias parameter b, the authors derive and demonstrate the equality of two apparently dissimilar ISDAs, namely a KA approach and an unbiased variant of the SMO training scheme [9], when constructing SVMs possessing positive definite kernels. The equivalence is applicable to both classification and regression tasks and gives additional insights into these apparently dissimilar methods of learning. Despite the richness of the toolbox set up to solve the quadratic programs from SVMs, and with the large amount of data generated by social networks, medical and agricultural fields, etc., the amount of computer memory required for a QP solver from the dual-SVM grows hyper-exponentially, and additional methods implementing different techniques and strategies are more than necessary.

Classical algorithms, namely ISDAs and SMO, do not distinguish between different types of samples (noise, border, and core), which causes searches in unpromising areas. In this work, we introduce a hybrid method to overcome these shortcomings, namely the Optimal Recurrent Neural Network Density-Based Support Vector Machine (Opt-RNN-DBSVM). This method proceeds in four steps: (a) the characterization of different samples based on the density of the datasets (noise, core, and border), (b) the elimination of samples with a low probability of being a support vector, namely core samples that are very far from the borders of different components of different classes, (c) the construction of an appropriate recurrent neural network based on an original energy function, ensuring a balance between the dual-SVM components (constraints and objective function) and

ensuring the feasibility of the network equilibrium points [16,17], and (d) the solution of the system of differential equations, managing the dynamics of the RNN, using the Euler–Cauchy method involving an optimal time step. Density-based preprocessing reduces the number of local minima in the dual-SVM. The RNN's recurrent architecture avoids the need to examine previously visited areas; this behavior is similar to a taboo search, which prohibits certain moves for a few iterations [18]. In addition, the optimal time step of the Euler–Cauchy algorithm speeds up the search for an optimal decision margin. On one hand, two main interesting fundamental results are demonstrated: the convergence of the RNN-SVM to feasible solutions, and the fact that Opt-RNN-DBSVM has very low time complexity compared to Const-RNN-SVM, SMO-SVM, ISDA-SVM, and L1QP-SVM. On the other hand, several experimental studies are conducted based on well-known datasets. Based on several performance measures (accuracy, F1-score, precision, recall), Opt-RNN-DBSVM outperforms recurrent neural network–SVM with a constant time step, the Kernel-Adatron algorithm–SVM family, and well-known non-kernel models. In fact, Opt-RNN-DBSVM improves the accuracy, the F1-score, the precision, and the recall. Moreover, the proposed method requires a very small number of support vectors.

The rest of this paper is organized as follows. Section 2 presents the flowchart of the proposed method. Section 3 gives the outline of our recent SVM version called Density-Based Support Vector Machine. Section 4 presents, in detail, the construction of the recurrent neural network associated with the dual-SVM and the Euler–Cauchy algorithm that implements an optimal time step. Section 5 gives some experimental results. Section 6 presents some conclusions and future extensions of Opt-RNN-DBSVM.

2. The Architecture of the Proposed Method

The Kernel-Adatron (KA) algorithms, namely ISDAs and SMO, treat different types of samples (noise, border, and core) in the same manner (all samples are considered for several iterations and supposed to be a support candidate with uniform probability), which causes searches in unpromising areas and increases the number of iterations. In this work, we introduce an efficient method to overcome these shortcomings, namely Optimal Recurrent Neural Network Density-Based Support Vector Machine (Opt-RNN-DBSVM). This method proceeds in four steps (see Figure 1).

(1) The characterization of different samples based on the density of the datasets (noise, core, and border); to this end, two parameters are introduced: the size of the neighborhood of the current sample and the threshold that permits such categorization.
(2) The elimination of samples with a low probability of being a support vector, namely core samples that are very far from the borders of different components of different classes and the noise samples that contain false information about the phenomenon under study. In our previews work [19], we demonstrated that such suppression does not influence the performance of the classifiers.
(3) The construction of an appropriate recurrent neural network based on an original energy function, allowing a balance between the dual-SVM components (constraints and objective function) and ensuring the feasibility of the network equilibrium points [16,17].
(4) Solving the system of differential equations, managing the dynamics of the RNN, using the Euler–Cauchy method involving an optimal time step. In this regard, the equation of the future state of each neuron, of the proposed RNN, is introduced into the energy function, which leads to a one-dimension quadratic optimization problem whose solution represents the optimal step of the Euler–Cauchy process that ensures the maximum decrease in the energy function [20]. The components of the produced equilibrium point represent the membership degrees of different samples to the support vector dataset.

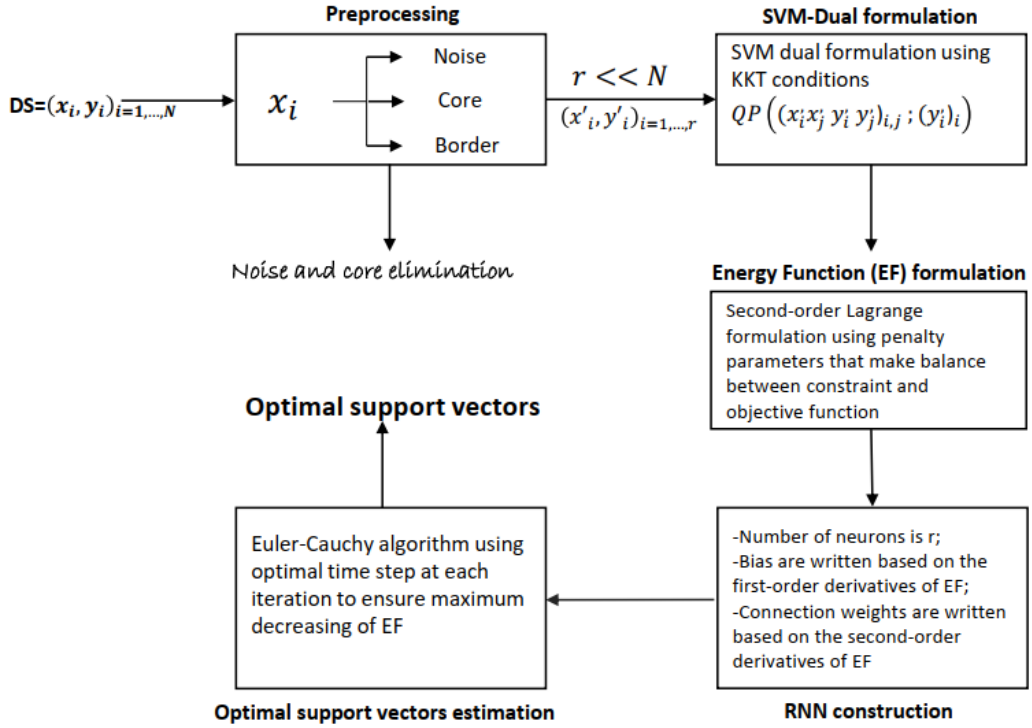


Figure 1. Opt_RNN_DBSVM diagram [21–24].

3. Density-Based Support Vector Machine

In the following, let us denote by BD the set of N samples $x_1, \dots, x_N$ labeled, respectively, by $y_1, \dots, y_N$, distributed via K class $C_1, \dots, C_K$. In our case, K=2 and $y_i \in \{-1, +1\}$.

3.1. Classical Support Vector Machine

The hyperplane that the SVM searches must satisfy the equation $w.x_i + b = 0$, where w is the weight that defines this SVM separator that satisfies the constraint family given by $\forall i = 1, \dots, N\ y_i(x_i.w + b) \geq 1$. To ensure the maximum margin, we need to maximize $\frac{2}{\|w\|}$. As the patterns are not linearly separable, the kernel function K is introduced (which satisfies the Mercer conditions [25]) to transform the data into an appropriate space.

By introducing the Lagrange relaxation and using the Kuhn–Tuker conditions, we obtain a quadratic optimization problem with a single linear constraint that must be solved to determine the support vectors [26].

To address the problem of saturated constraints, some researchers have added the notion of a soft margin [27]. They employ N supplementary slack variables $\xi_i \geq 0$ at every constraint $y_i(x_i.w + b) \geq 1$. The sum of the relaxed variables is weighted and included in the cost function:

$$\begin{cases} Min \dfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i \\ \quad \texttt{Subject to :} \\ \qquad y_i(\phi(x_i).w + b) \geq 1 - \xi_i \\ \qquad \xi_i \geq 0, \forall i = 1, \dots, N \end{cases}$$

Here, ϕ represents the transformation function derived from the function kernel K. The following dual problem is obtained:

$$\begin{cases} Max \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ \quad \texttt{Subject to :} \\ \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \\ \quad 0 \leq \alpha_i \leq C, \forall i = 1, \ldots, N \end{cases}$$

Several methods can be used to solve this optimization problem: gradient methods, linearization methods, the Frank–Wolf method, the generation column method, the Newton method applied to the Kuhn system, sub-gradient methods, the Dantzig algorithm, the Uzawa algorithm [4], recurrent neural networks [28], hill climbing, simulated annealing, search by calf, A*, genetic algorithms [29], ant colony, and the particle swarm optimization method [30], etc.

Several versions of SVMs are proposed in the literature, e.g., the least squares support vector machine classifiers (LS-SVM) introduced in [21], generalized support vector machine (G-SVM) [22], fuzzy support vector machine [31,32], one-class support vector machine (OC-SVM) [26,33], total support vector machine (T-SVM) [34], weighted support vector machine (W-SVM) [35], granular support vector machine (G-SVM) [36], smooth support vector machine (S-SVM) [37], proximity support vector machine classifiers (P-SVM) [23], multisurface proximal support vector machine classification via generalized eigenvalues (GEP-SVM) [24], and twin support vector machine (T-SVM) [38], etc.

3.2. Density-Based Support Vector Machine (DBSVM)

In this section, a short description of the DBVSM method is given. Let us introduce a real number $r > 0$ and the integer $mp > 0$, called min-points, and three types of samples are defined: noise points, border points, and interior points (or core points). It is possible to show that the interior points do not change their nature even when they are projected into another space by the kernel functions. Furthermore, such points cannot be selected as support vectors [19].

Definition 1. *Let $S \subseteq \mathbb{R}^n$. A point $a \in \mathbb{R}^n$ is said to be an interior point (or core point) of S if there exists an $r > 0$ such that $B(a, r) \subseteq S$. The set of all interior points of S is denoted by $int(S)$ or $\overset{o}{S}$.*

Definition 2. *For a given dataset BD, a non-negative real r, and an integer mp, there exist three types of samples.*

1. *A sample x is called a C_i-noise point (NP_i) if $|C_i \cap B(x, r)| < mp$.*
2. *A sample x is called a C_i-core point (CP_i) if $|C_i \cap B(x, r)| \geq mp$ and $x \in \overbrace{envol(C_i)}^{o}$*
3. *A sample x is called a C_i-border point (BP_i) if $|C_i \cap B(x, r)| < mp$ and there exists a C_i-core point y such as $x \in B(y, r)$.*

Let K be a kernel function allowing us to move from the space $\mathbb{R}^n$ to the space $\mathbb{R}^N$ using the transformation ϕ (here, $n < N$).

Lemma 1 ([19]). *If a is a C_i-core point for a given ϵ and min-points (mp), then $\phi(a)$ is also a C_i-core point with an appropriate ϵ' and the same min-points (mp).*

Theorem 1 ([19]). *A core point is either a noise point or a border point.*

Proposition 1 ([19]). *Let $\epsilon > 0$ be a real number. The core point set corePoints (minPoints) is a decreasing function for the inclusion operator.*

Let $\{\alpha_1, \ldots, \alpha_n\} = BM \cup CM \cup NM$ be the set of the Lagrange multipliers, where BM, CM, and NM are the Lagrange multipliers of the border samples, core samples, and noise samples, respectively.

As the elements of NM and CM cannot be selected to be support vectors, the reduced dual problem is given by

$$(RD)\begin{cases} Max \sum\limits_{\alpha_i \in BM} \alpha_i - \frac{1}{2} \sum\limits_{\alpha_i \in BM} \sum\limits_{\alpha_j \in BM} \alpha_i \alpha_j y_i y_j K(x_i x_j) \\ \quad \texttt{Subject to :} \\ \quad\quad \sum_{\alpha_i \in BM} \alpha_i y_i = 0 \\ \quad\quad 0 \leq \alpha_i \leq C \quad \forall \alpha_i \in BM \;\; \forall i = 1, \ldots, N \end{cases}$$

In this work, as the RD problem is quadratic with linear constraints, in order to solve this, we use a continuous Hopfield network by proposing an original energy function in the following section [39].

4. Recurrent Neural Network to Find Optimal Support Vectors

The continuous Hopfield network consists of interconnected neurons with a smooth sigmoid activation function (usually a hyperbolic tangent). The differential equation that governs the dynamics of the CHN is

$$\frac{du}{dt} = -\frac{u}{\tau} + W.\alpha + I \tag{1}$$

where u, α, W, and I are, respectively, the vectors of neuron states, the outputs, the weight matrix, and the biases. For a CHN of N neurons, the state u_i and output α_i of the neuron i are given by the equation $\alpha_i = tanh(u_i) = f(u_i)$.

For an initial vector state $u^0 \in \mathbb{R}^N$, a vector $u^e \in \mathbb{R}^N$ is called an equilibrium point of the system 1, if and only if $\exists t^e \in \mathbb{R}^+$, such as $\forall t \geq t^e \; u(t) = u^e$. It should be noted that if the energy function (or Layapunov function) exists, the equilibrium point exists as well. Hopfield proved that the symmetry of the matrix of the weight is a sufficient condition for the existence of the Lyapunov function [40].

4.1. Continuous Hopfield Network Based on Original Energy Function

To solve the obtained dual problem via a recurrent neural network [39,41,42], we propose the following energy function:

$$E(\alpha_1, \ldots, \alpha'_N) = \beta_0 \sum_{\alpha_i \in D} \alpha_i - \frac{\beta_0}{2} \sum_{\alpha_i \in BM} \sum_{\alpha_j \in BM} \alpha_i \alpha_j y_i y_j K(x_i x_j) + \beta_1 \sum_{\alpha_i \in BD} \alpha_i y_i + \frac{\beta_2}{2} \Big(\sum_{\alpha_i \in BD} \alpha_i y_i \Big)^2$$

To determine the vector of the neurons' biases, we calculate the partial derivatives of E:

$$\frac{\partial E}{\partial \alpha_i}(\alpha_1, \ldots, \alpha'_N) = \beta_0 - \beta_0 \sum_{\alpha_j \in BM} \alpha_j y_i y_j K(x_i x_j) + \beta_1 y_i + \beta_2 y_i \sum_{\alpha_j \in BD} \alpha_j y_j$$

The components of the bias vector are given by

$$I_i = \frac{\partial E}{\partial \alpha_i}(0) = -\beta_0 - \beta_1 y_i \quad \forall i = 1, \ldots N'$$

To determine the connection weights W between each neuron pair, the second partial derivative of E is calculated: $\frac{\partial^2 E}{\partial \alpha_j \partial \alpha_i}(\alpha_1, \ldots, \alpha_{N'}) = -\beta_0 y_i y_j K(x_i, x_j) + \beta_2 y_i y_j$.

The components of the weight W matrix are given by $W_{i,j} = \frac{\partial^2 E}{\partial \alpha_j \partial \alpha_i}(0) = \beta_0 y_i y_j K(x_i, x_j) - \beta_2 y_i y_j$.

To calculate the equilibrium point of the proposed recurrent neural network, we use the Euler–Cauchy iterative method:

(1) Initialization: $\alpha_1^0, \ldots, \alpha_{N'}^0$ and the step ρ^0 are randomly chosen;
(2) Given $\alpha_1^t, \ldots, \alpha_{N'}^t$ and the step ρ^t, the step ρ^{t+1} is chosen such that E^{t+1} is the maximum and $u_1, \ldots, u_{N'}$ are calculated using $\forall i = 1, \ldots, N', u_i^{t+1} = u_i^t + \rho^{t+1}(\sum_{j=1,\ldots,N'} W_{i,j}\alpha_j^t + I_i)$
Then, $\gamma_1, \ldots, \gamma_{N'}$ are calculated using the activation function f: $\gamma_i = f(u_i^{t+1})$.
Then, the $\alpha_1^{t+1}, \ldots, \alpha_{N'}^{t+1}$ are given by $\alpha^{t+1} = P(\gamma)$, where P is the projection operator on the set $\{\alpha \in \mathbb{R}^{N'} / \sum_{i=1}^{N'} \alpha_i y_i = 0\}$.

(3) Return to (1) until $\|\alpha^{t+1} - \alpha^t\| \leq \epsilon$, where $0 < \epsilon$.

Figure 2 shows the connection weights W between each pair of neurons.

Theorem 2. *If* $i = j, P_{i,j} = 1 - \frac{y_i^2}{N'}$, *else* $P_{i,j} = -\frac{y_i y_j}{N'}$, *where* $S = \sum_{i=1,\ldots,N'} y_i^2$.

Proof of Theorem 2. We have $P = I - A^t.(A.A^t)^{-1}.A$ and $A = [y_1, \ldots, y_{N'}]$.
Then, $(A.A^t)^{-1} = ([y_1, \ldots, y_{N'}].[y_1, \ldots, y_{N'}]^t)^{-1} = \frac{1}{N'}$ because $N' = \sum_{i=1,\ldots,N'} y_i^2$.
Thus $A^t.(A.A^t)^{-1}.A = \frac{1}{N'}.[y_1, \ldots, y_{N'}]^t.[y_1, \ldots, y_{N'}]$.
Finally, for $i = j$, $P_{i,j} = 1 - \frac{y_i^2}{N'}$, and for $i \neq j, P_{i,j} = -\frac{y_i.y_j}{N'}$. □

Concerning the constraint family satisfaction $0 \leq \alpha_i \leq C, \forall i = 1, \ldots, N$, the activation function is used:

$$f(x) = C.tanh(\frac{x}{\tau}) = C.\frac{e^x - e^{-x}}{e^x + e^{-x}},$$

where τ is supposed to be a very large positive real number, which ensures that $\forall x, -C \leq f(x) \leq C$.

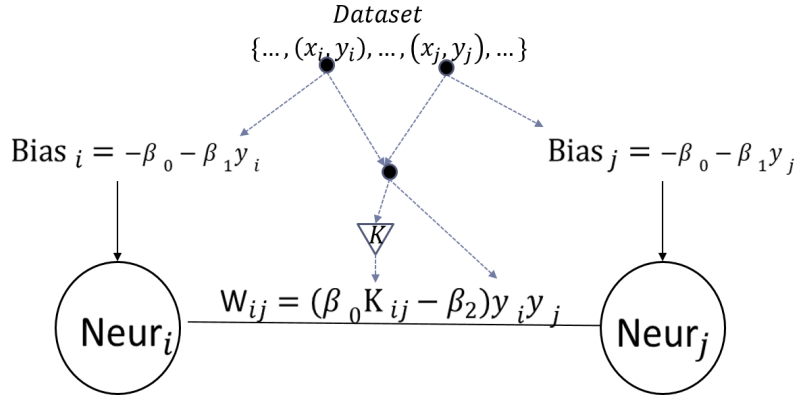


Figure 2. Architecture of the connection weights $W_{i,j}$ between each neuron pair.

Let us consider a kernel function K such that $K(x, x) = C \neq 0$.

Theorem 3. *A continuous Hopfield network has an equilibrium point if* $W_{i,i} = 0$ *and* $W_{i,j} = W_{j,i}$.

Theorem 4. *If* $C = \frac{\beta_2}{\beta_0}$, *then CHN-SVM has an equilibrium point.*

Proof of Theorem 4. We have $W_{i,j} = (\beta_0 K_{i,j} - \beta_2) y_i y_j$ and $\forall i \text{ and } j \; W_{i,j} = W_{j,i}$ because K is symmetric.
On the other hand,

$$W_{i,i} = \beta_0 y_i K(x_i, x_i) - \beta_2 y_i^2 = \beta_0 \times \frac{\beta_2}{\beta_0} - \beta_2 = 0$$

Then, CHN-SVM has an equilibrium point. □

4.2. Continuous Hopfield Network with Optimal Time Step

In this section, we chose, mathematically, the optimal time size in each iteration of the Euler–Cauchy method to solve the dynamical equation of the recurrent neural network proposed in this paper. At the end of the k^{th} iteration, we know α^k and let s_k be the next step size, which permits us to calculate α^{k+1} using the formula

$$\alpha^{k+1} = \alpha^k + s_k \nabla E(\alpha^k)$$

and s_k must be chosen such as $E(\alpha^{k+1}) \leq E(\alpha^k)$ at the maximum.

As the activation function of the proposed neural network is the *tanh*, then $\frac{d\alpha_i}{dt} = \frac{2}{\tau}\alpha_i(1-\alpha_i)\nabla E(\alpha)$.

The matrix form of the energy function is

$$E(\alpha) = \beta_0 U^t \alpha - \frac{\beta_0}{2}(\alpha)^t T \alpha + \beta_1 y^t \alpha + \frac{\beta_2}{2}(y^t \alpha)^2$$

where $U = (1, \ldots, 1)^t \in IR^{N'}$, $\alpha = (1, \ldots, 1)^t \in IR^{N'}$, and $T_{i,j} = y_i y_j K(x_i, x_j)$ for all i and j.

At the k^{th} iteration, the state α^k is known, and α^{k+1} is calculated by

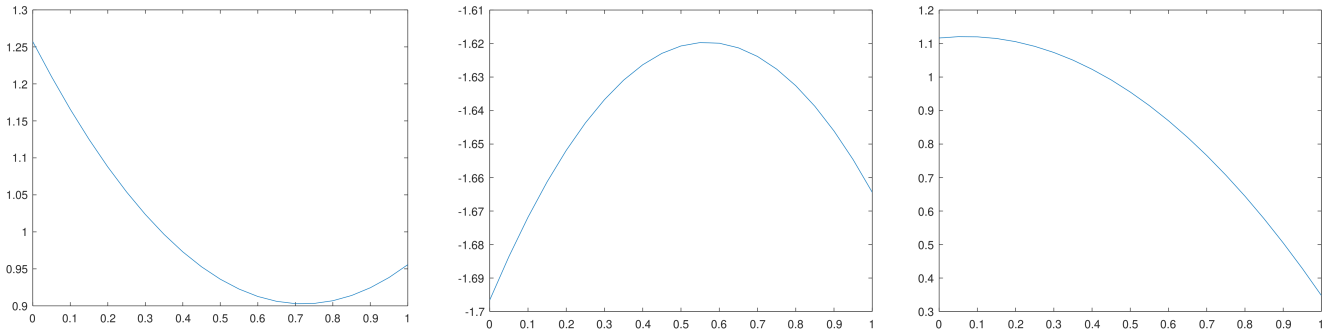
$$\alpha^{k+1} = \alpha^k + s_k \frac{d\alpha^k}{dt}$$

where s_k is the actual time step that must be optimal. To this end, α^{k+1} is substituted by $\alpha^k + s_k \frac{d\alpha^k}{dt}$ in $E(\alpha^{k+1})$:

$$e(s_k) = E(\alpha^{k+1}) = 0.5 A_k s_k^2 + B_k s_k + C_k$$

where $A_k = \beta_2 y^t \frac{d\alpha^k}{dt} - \beta_0 (\frac{d\alpha^k}{dt})^t T \frac{d\alpha^k}{dt}$,
$B_k = \beta_0 U^t \frac{d\alpha^k}{dt} - \beta_0 (\alpha^k)^t T \frac{d\alpha^k}{dt} + \beta_1 y^t \frac{d\alpha^k}{dt} + \beta_2 (y^t \alpha^k)(y^t \frac{d\alpha^k}{dt})$,
$C_k = \beta_0 U^t \alpha^k - 0.5\beta_0 (\alpha^k)^t T \alpha^k + \beta_1 y^t \alpha^k + 0.5\beta_2 (y^t \alpha^k)^2$.

Thus, the best time step is the minimum of $e(s_k)$. Figure 3a–c gives different cases.



(**a**) (**b**) (**c**)

Figure 3. Graphical representation of the variation in the sums and integral terms with respect to s_k. (**a**) RNN-DBSVM 1, (**b**) RNN-DBSVM 2, (**c**) RNN-DBSVM 3.

4.3. Opt-RNN-DBSVM Algorithm

In this section, the procedures described in Sections 3.2, 4.1 and 4.2 are summarized into Algorithm 1.

The inputs of Algorithm 1 are the radius r (the size of the neighborhood of the current sample), the minimum of samples mp into $B(Current\_sample, r)$ (which determines the type of this sample), the three Lagrangian parameters β_0, β_1, and β_2 (which allow a compromise between the dual components), the bound C of the SVM [19], and the number of iterations (which represents artificial convergence).

Algorithm 1 processes in three macro-steps: data preprocessing, RNN-SVM construction, and RNN-SVM equilibrium point estimation. The input of the first phase is the initial dataset with labeled samples. Based on r and mp, the algorithm determines the types of different samples based on the value of the current sample neighborhood's discrete size. The output of this phase is a reduced sub-dataset (the initial dataset minus the core samples). The inputs of the second phase are the reduced dataset, the Lagrangian parameters β_0, β_1, β_2, and the SVM bound C. Based on the energy function built in Section 4.1 and on the first and second derivatives, the architecture of CHN-SVM is constructed; the bias and connection weights, which represent the output of this phase, are calculated. These later represent the input of the third phase and the Euler–Cauchy algorithm is used to calculate the degree of membership of different samples in the set of support vectors; to ensure an optimal decrease in the energy function, at each iteration, an optimal step is determined by solving a quadratic one-dimension optimization problem; see Section 4.2. At convergence, the proposed algorithm produces the support vectors based on which Opt-RNN-DBSVM can predict the class of unseen samples.

Algorithm 1 Opt-RNN-DBSVM

Require: $mp, r, \beta_0, \beta_1, \beta_2, C, ITER$
Ensure: *Optimal support vectors*
% Density based preprocessing:
$CP \leftarrow ;\ \ BP \leftarrow ;\ \ NP \leftarrow ;$
for all s in DS **do**
  if $|B(s,r) \cap DS| > mp$ **then**
    $CP \leftarrow CP \cup \{s\}$
  else if $mp > |B(s,r) \cap DS| > \frac{mp}{2}$ **then**
    $BP \leftarrow BP \cup \{s\}$
  else
    $NP \leftarrow NP \cup \{s\}$
  end if
end for
$RDS \leftarrow DS \setminus \{NP \cup CP\}$
% RNN-Building:
$I \leftarrow \beta_0 + \beta_1 Y$
$W \leftarrow \beta_0 K - \beta_2 Y'^2$
% Optimal Euler–Cauchy to RNN stability:
$step_0 = rand(0,1)$
$\alpha'_0 \leftarrow rand(0,1,|RDS|)$
for k = 1, ..., ITER **do**
  $D_k \leftarrow \frac{d\alpha^k}{dt}$
  $A_k \leftarrow \beta_2 y^t D_k - \beta_0 (D_k)^t T D_k$
  $B_k \leftarrow \beta_0 U^t D_k - \beta_0 (\alpha^k)^t T D_k + \beta_1 y^t D_k + \beta_2 (y^t \alpha^k)(y^t D_k)$
  $C_k \leftarrow \beta_0 U^t \alpha^k - 0.5\beta_0 (\alpha^k)^t T D_k + \beta_1 y^t \alpha^k + 0.5\beta_2 (y^t \alpha^k)^2$
  $s_k^* \leftarrow argmin_{s \in [0,1]} e(s)$
  $\alpha^{k+1} \leftarrow \alpha^k + s_k^* D_k$
end for

Proposition 2. *If N, r, and $ITER$ represent, respectively, the size of a labeled dataset BD, the number of remaining samples (output of the preprocessing phase), and the number of iterations, then the complexity of Algorithm 1 is $O(r^2 * ITER)$.*

Proof. First, in the preprocessing phase, we calculate, for each sample (x^i, x^j), the distance $d(x^i, x^j)$ and execute N comparisons to determine the type of each sample; thus, the complexity of this phase is $O(N^2)$.

Second, during $ITER$ iterations, the activation of each neuron is updated using the activation of all the other neurons to solve the reduced dual-SVM; thus, the third phase has complexity of $O(r^2 \times ITER)$.

Finally, the complexity of Algorithm 1 is $O(N^2) + O(r^2 \times ITER)$. Let us denote Const-RNN-SVM as the SVM version that implements a recurrent neural network based on a constant time step. Following the same reasoning, the complexity of Const-RNN-SVM is $O(N^2 \times ITER)$.

Notes: As the Kernel-Adatron algorithm (KA) is the kernel version of SMO and ISDA, and KA implements two embedded N-loops in each iteration, then the complexity of SMO and ISDA is of [10]. In addition, it is considered that L1QP-SVM implements the numerical linear algebra Gauss–Seidel method [43], which implements two embedded N-loops in each iteration, and thus the complexity of SMO and ISDA is of $O(N^2 \times ITER)$. □

For a very large and high-density labeled dataset, we have $r << N$; thus,

$$ITER_{our} << ITER_{CHN-SVM} \text{ and } ITER_{our} << ITER_{L1QP-SVM}\ ITER_{our} << ITER_{SMO-SVM} \text{ and } ITER_{our} << ITER_{ISDA-SVM}.$$

Thus, $complexity(our) \prec complexity(CHN - SVM)$ and $complexity(our) \prec complexity(L1QP - SVM)$ $complexity(our) \prec complexity(SMO - SVM)$, and $complexity(our) \prec complexity(ISDA - SVM)$.

Firstly, preprocessing the database reduces the number of local minima in the dual-SVM. Secondly, this reduction enables real-time decision making in big data problems. Finally, the optimal time step of the Euler–Cauchy algorithm speeds up the search for an optimal decision margin.

5. Experimentation

In this section, Opt-RNN-DBSVM is compared to several classifiers, Const-RNN-SVM (RNN-SVM using a constant Euler–Cauchy time step), SMO-SVM, ISDA-SVM, L1QP-SVM, and some non-kernel classifiers (Naive Bayes (NB), MLP, KNN, AdaBoostM1 (ABM1), Nearest Center Classifier (NCC), Decision Tree (DT), SGD Classifier (SGDC)). The classifiers were tested on several datasets: IRIS, ABALONE, WINE, ECOLI, BALANCE, LIVER, SPECT, SEED, and PIMA (collected from the University of California at Irvine (UCI) repository [44]). The performance measures used in this study are the accuracy, F1-score, precision, and recall.

5.1. Opt-RNN-DBSVM vs. Const-CHN-SVM

In this subsection, Opt-RNN-DBSVM is compared to Const-RNN-SVM by considering different values of the Euler–Cauchy time step $s \in \{0.1, 0.2, \ldots, 0.9\}$. Tables 1 and 2 show the different values of accuracy, F1-score, precision, and recall on the considered datasets. These results show the superiority of Opt-RNN-DBSVM over Const-CHN-SVM ($step \in STEP = \{0.1, 0.2, \ldots, 0.9\}$). In fact, this superiority is quantified as follows:

$$3.43\% = \max_{(s,d)\in STEP\times DATA} (accuracy(Opt - RNN - DBSVM(s)) - accuracy(const - RNN - SVM))$$

$$2.31\% = \max_{(s,d)\in STEP\times DATA} (F1Score(Opt - RNN - DBSVM(s)) - F1Score(const - RNN - SVM))$$

$$7.52\% = \max_{(s,d)\in STEP\times DATA} (precision(Opt - RNN - DBSVM(s)) - precision(const - RNN - SVM))$$

$$6.5\% = \max_{(s,d)\in STEP\times DATA} (recall(Opt - RNN - DBSVM(s)) - recall(const - RNN - SVM))$$

where $DATA$ is the set of different considered data. These results are not unexpected, because Opt-RNN-SVM ensures an optimal decrease in the CHN energy function at each

step. This superiority is normal, since a single time step of the Euler–Cauchy algorithm does not explore all the regions of the solution space of the dual problem associated with the SVM, and it also causes premature convergence to a poor local solution. On the other hand, the variable optimal time step of this algorithm allowed a higher-order decay in the energy function of the RNN associated with the dual-SVM.

Table 1. Performance Performance of Const-CHN-SVM on different datasets for different values of time step in [0.1, 0.6].

	SVM-CHN $s = 0.1$				**SVM-CHN $s = 0.2$**			
	Accuracy	**F1-Score**	**Precision**	**Recall**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
IRIS	95.98	96.66	90.52	92.00	96.66	95.98	91.62	92.00
ABALONE	80.98	40.38	82.00	27.65	80.66	40.38	81.98	27.65
WINE	79.49	78.26	73.52	74.97	79.49	78.26	73.52	74.97
ECOLI	88.05	96.77	97.83	97.33	88.05	97.77	97.83	97.99
BALANCE	79.70	70.7	55.60	62.70	79.70	70.7	55.60	62.70
LIVER	80.40	77.67	77.90	70.08	80.40	77.67	77.90	70.08
SPECT	92.12	90.86	91.33	90.00	97.36	99.60	97.77	1.00
SEED	85.71	83.43	92.70	75.04	85.71	83.43	92.70	75.04
PIMA	79.22	61.90	84.7	49.6	79.22	61.90	83.97	49.6
	SVM-CHN $s = 0.3$				**SVM-CHN $s = 0.4$**			
	Accuracy	**F1-Score**	**Precision**	Recall	**Accuracy**	**F1-Score**	**Precision**	**Recall**
IRIS	94.53	95.86	89.66	98.23	95.96	93.88	89.33	95.32
ABALONE	77.99	51.68	83.85	30.88	81.98	41.66	83.56	33.33
WINE	80.23	77.66	74.89	74.97	81.33	77.65	73.11	74.43
ECOLI	88.86	95.65	96.88	97.33	86.77	97.66	97.83	97.95
BALANCE	79.75	70.89	55.96	62.32	79.66	70.45	56.1	66.23
LIVER	80.51	78.33	77.9	70.56	80.40	77.67	77.90	70.08
SPECT	97.63	98.99	97.81	98.56	96.40	98.71	96.83	97.79
SEED	85.71	83.43	92.70	75.88	85.71	83.43	92.70	75.61
PIMA	79.22	61.93	84.82	49.86	79.22	61.90	84.98	49.89
	SVM-CHN $s = 0.5$				**SVM-CHN $s = 0.6$**			
	Accuracy	**F1-Score**	**Precision**	Recall	**Accuracy**	**F1-Score**	**Precision**	**Recall**
IRIS	94.53	95.86	89.66	98.23	95.96	93.88	89.33	95.32
ABALONE	78.06	51.83	83.96	40.45	82.1	42.15	83.88	38.26
WINE	80.84	78.26	74.91	75.20	81.39	77.86	73.66	74.47
ECOLI	88.97	95.7	96.91	97.43	86.77	97.66	97.83	97.95
BALANCE	79.89	71.00	56.11	62.72	79.71	70.64	56.33	66.44
LIVER	80.66	78.33	77.9	70.56	80.40	77.67	77.90	70.08
SPECT	91.36	92.60	91.77	84.33	91.36	92.60	91.77	84.33
SEED	84.67	82.96	92.23	74.18	84.11	83.08	92.63	75.48
PIMA	79.12	61.75	84.62	49.86	79.12	61.33	84.68	48.55

Table 2. Performance of Const-CHN-SVM on different datasets for different values of time step in [0.7, 0.9].

	SVM-CHN $s = 0.7$				**SVM-CHN $s = 0.8$**			
	Accuracy	**F1-Score**	**Precision**	**Recall**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
IRIS	94.53	95.86	89.66	98.23	95.96	93.88	89.33	95.32
ABALONE	77.99	51.68	83.85	30.88	81.98	41.66	83.56	33.33
WINE	80.23	77.66	74.89	74.97	81.33	77.65	73.11	74.43
ECOLI	88.86	95.65	96.88	97.33	86.77	97.66	97.83	97.95
BALANCE	79.75	70.89	55.96	62.32	79.66	70.45	56.1	66.23
LIVER	80.51	78.33	77.9	70.56	80.40	77.67	77.90	70.08
SPECT	94.36	84.60	83.77	85.99	94.36	84.60	83.77	85.99
SEED	85.71	83.43	92.70	75.88	85.71	83.43	92.70	75.61
PIMA	79.22	61.93	84.82	49.86	79.22	61.90	84.98	49.89
	SVM-CHN $s = 0.9$				**CHN-DBSVM Optimal Value of s**			
	Accuracy	**F1-Score**	**Precision**	**Recall**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
IRIS	95.96	93.88	89.33	95.32	**97.96**	**96.19**	**95.85**	**98.5**
ABALONE	81.98	41.66	83.56	33.33	**98.38**	**96.07**	**96.18**	**93.19**
WINE	81.33	77.65	73.11	74.43	**96.47**	**95.96**	**96.08**	**96.02**
ECOLI	86.77	88.46	90.77	91.19	**91.82**	**97.66**	**97.83**	**97.95**
BALANCE	79.66	70.45	56.1	66.23	**91.31**	**90.33**	**89.54**	**90.89**
LIVER	80.40	77.67	77.90	70.08	**88.10**	**85.95**	**86.00**	**85.50**
SPECT	94.36	84.60	83.77	85.99	**95.55**	**86.20**	**85.28**	**86.31**
SEED	85.71	83.43	86.18	75.61	**88.90**	**84.31**	**92.70**	**84.40**
PIMA	79.22	61.90	75.90	49.89	**79.87**	**68.04**	**84.98**	**62.05**

Figures 4 and A1–A3 give the series of optimal steps generated by Opt-RNN-DBSVM during iterations for different datasets. It is noted that all the optimal steps are taken from the interval $[0.3; 0.4]$, which explains why a single constant time step of the Euler–Cauchy algorithm can never produce satisfactory support vectors compared to Opt-RNN-DBSVM. However, this simulation provides an optimal domain for those using a CHN based on a constant time step instead of taking a random time step from $[0; 1]$.

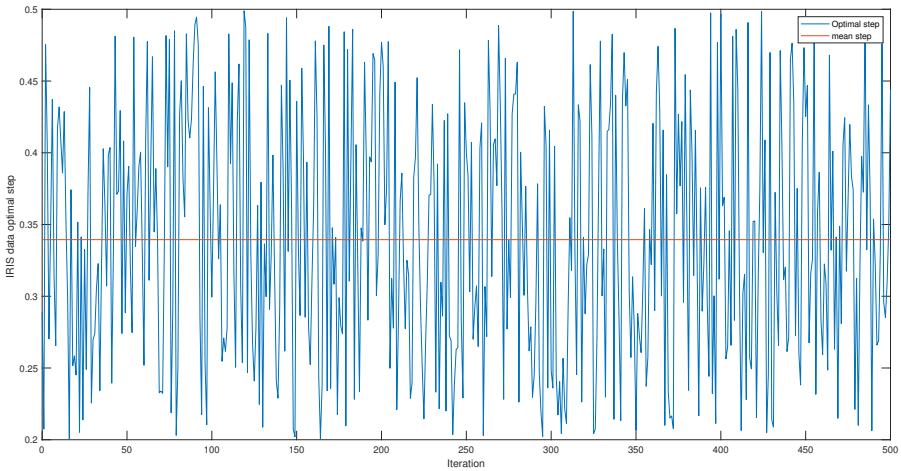


Figure 4. IRIS dataset optimal time steps.

5.2. Opt-RNN-DBSVM vs. Classical Optimizer–SVM

In this section, we give the performance of different Classical Optimizer–SVM models (L1QP-SVM, ISDA-SVM, and SMO-SVM) applied to several datasets and compare the number of support vectors obtained by the different Classical Optimizer–SVM models and Opt-RNN-SVM. Table 3 gives the values of accuracy, F1-score, precision, and recall for Classical Optimizer–SVM on different datasets. The results show the superiority of Opt-RNN-DBSVM. Indeed, when considering each of the performance measures, the proposed method achieves remarkable improvements of 30% for accuracy and F1-score, 50% for precision, and 40% for recall.

Table 3. Performance of Classical Optimizer–SVM on different datasets.

	L1QP-SVM				**ISDA-SVM**			
	Accuracy	**F1-Score**	**Precision**	**Recall**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
IRIS	71.59	62.02	70.80	55.17	82.00	83.64	76.67	92.00
ABALONE	74.15	70.80	71.48	59.30	83.70	68.22	70.00	80.66
WINE	72.90	65.79	75.53	60.11	66.08	65.80	66.00	70.02
ECOLI	66.15	55.89	61.33	41.30	51.60	48.30	33.33	51.39
BALANCE	65.20	53.01	60.51	41.22	50.44	58.36	68.32	60.20
LIVER	64.66	52.06	60.77	40.44	50.00	48.00	62.31	51.22
SPECT	70.66	62.02	67.48	50.11	77.60	71.20	75.33	70.11
SEED	70.51	58.98	67.30	45.30	80.66	81.25	79.80	79.30
PIMA	65.18	53.23	60.88	39.48	49.32	44.33	48.90	50.27
	SMO-SVM				**CHN-DBSVM Optimal Value of *s***			
	Accuracy	**F1-Score**	**Precision**	**Recall**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
IRIS	71.59	62.02	70.80	55.17	**97.96**	**96.19**	**95.85**	**98.5**
ABALONE	74.15	70.80	71.48	59.30	**98.38**	**96.07**	**96.18**	**93.19**
WINE	72.90	65.79	75.53	60.11	**96.47**	**95.96**	**96.08**	**96.02**
ECOLI	66.15	55.89	61.33	41.30	**91.82**	**88.46**	**90.77**	**91.19**
BALANCE	65.20	53.01	60.51	41.22	**91.31**	**90.33**	**89.54**	**90.89**
LIVER	64.66	52.06	60.77	40.44	**88.10**	**85.95**	**86.00**	**85.50**
SPECT	70.66	62.02	67.48	50.11	**95.55**	**86.20**	**85.28**	**86.31**
SEED	70.51	58.98	67.30	45.30	**88.90**	**84.31**	**86.18**	**84.40**
PIMA	65.18	53.23	60.88	39.48	**79.87**	**68.04**	**75.90**	**62.05**

Figures 5–8 illustrate, respectively, the support vectors obtained using L1QP-SVM, L1QP-SVM, SMO-SVM, and Opt-RNN-SVM applied to the IRIS data. We note that (a) ISDA considers more than 96% as support vectors, which is an exaggeration; (b) L1QP and SMO use a reasonable number of samples as support vectors, but most of them are duplicated; and (c) thanks to the preprocessing, Opt-RNN can reduce the number of support vectors by more than 32%, compared to SMO and L1QP, which allows it to overcome the over-learning phenomenon encountered with SMO and L1QP. In this sense, this reasonable number of support vectors used by Opt-RNN-DBSVM will speed up the online predictions of systems that implement these support vectors, especially with regard to to sentiment analysis, which manipulates very long texts.

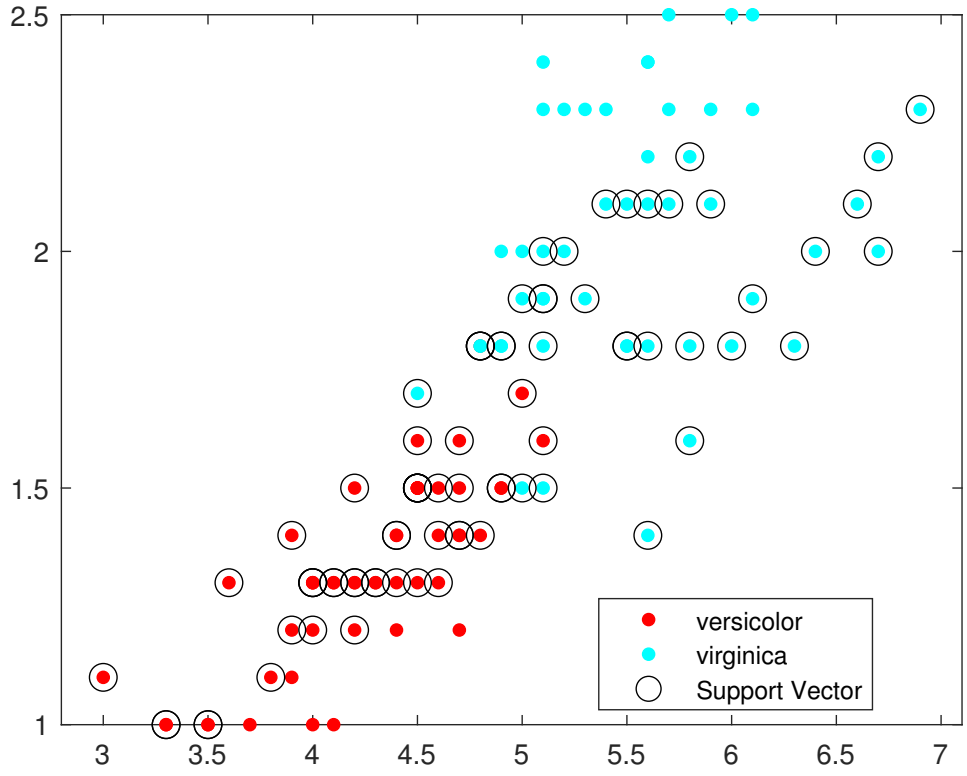


Figure 5. Support vectors obtained by ISDA algorithm.

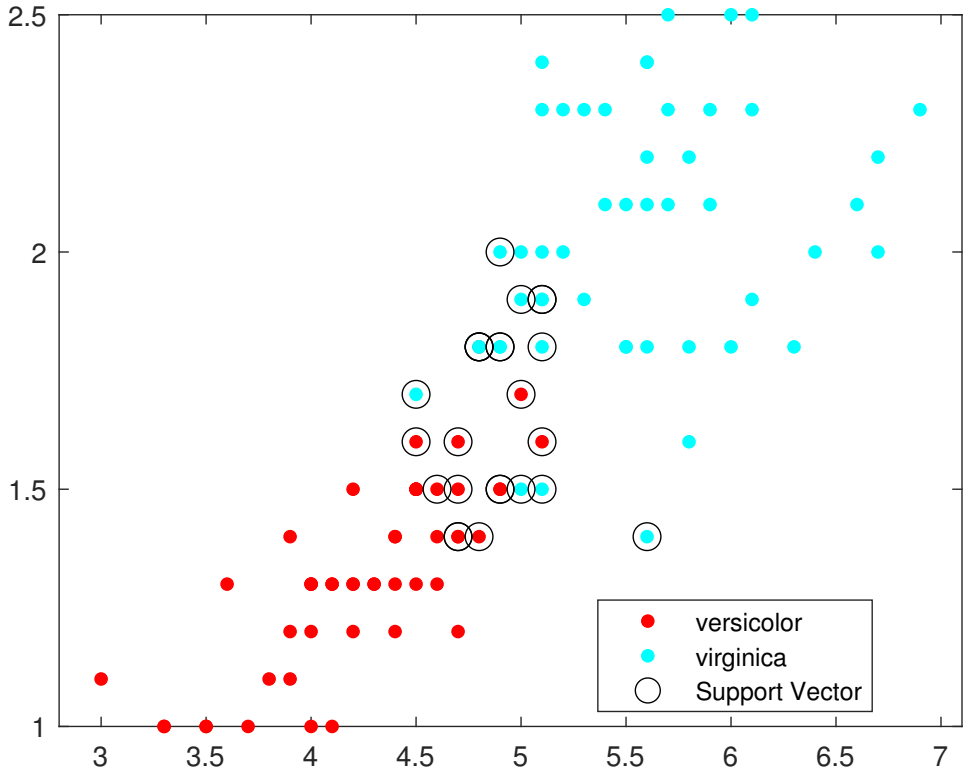


Figure 6. Support vectors obtained by L1QP algorithm.

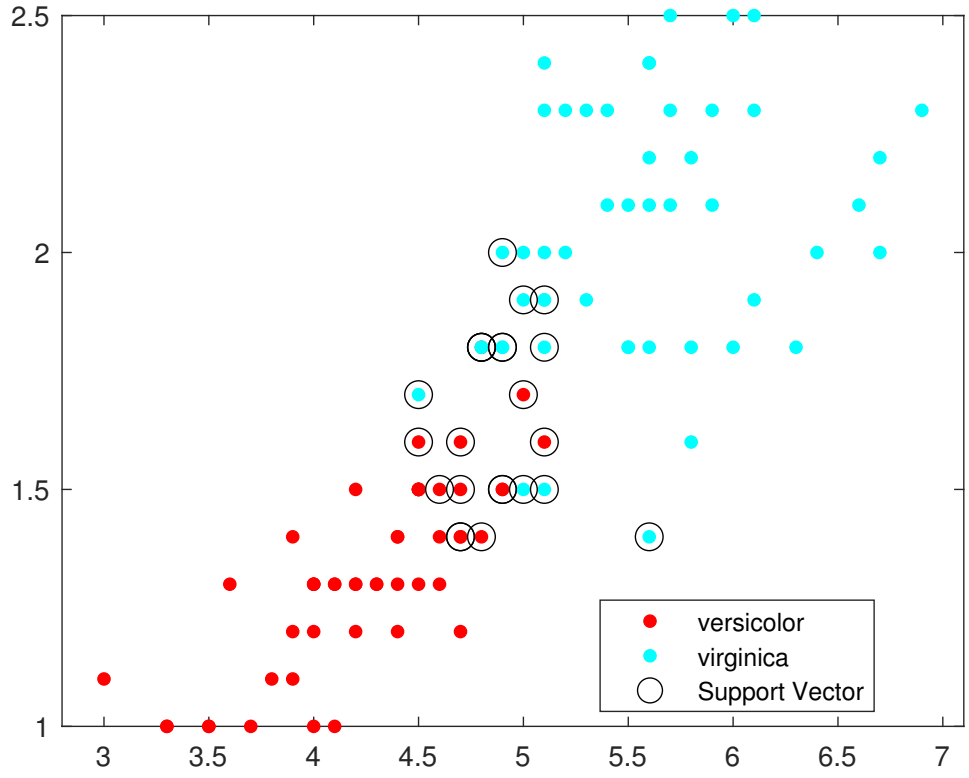


Figure 7. Support vectors obtained by SMO algorithm.

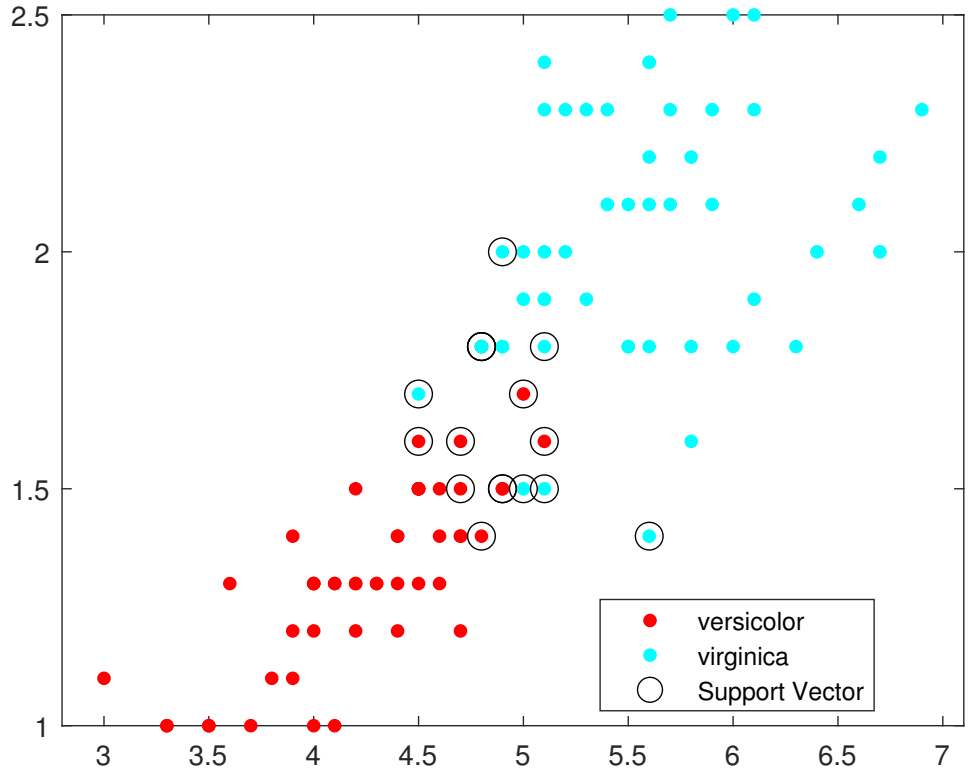


Figure 8. Support vectors obtained by Opt_RNN_SVM algorithm.

To analyze these results further and to evaluate the performance of multiple kernel classifiers, regardless of the data type, we perform the Friedman test to verify the statistical significance of the proposed method compared to other methods with respect to the derived

mean rankings [11]. The null hypothesis is given by H_0^{ker}, i.e., "The kernel classifiers Opt-RNN-DB, SMO, ISDA, and L1QP perform similarly in mean rankings without a significant difference". The considered degree of freedom of the Friedman test is 3 (number of kernel classifiers -1). The significance level is 0.05 and the considered confidence interval is 95%. Three performance measures are considered (accuracy, F1-score, precision).

Considering the accuracy measure, the average rank of the four kernel methods is given in brackets: Opt-RNN-DB(4), SMO(2.06), and L1QP(2.06), and ISDA(1.89). Opt-RNN-DB has the highest ranking, followed by SMO and L1QP. Considering the F1-score, the average rank of the four kernel methods is given in brackets: Opt-RNN-DB(4), ISDA(2.11), SMO(1.94), and L1QP(1.94). Opt-RNN-DB has the highest ranking, followed by ISDA. Considering the precision measure, the average rank of the four kernel methods is given in brackets: Opt-RNN-DB(4), ISDA(2.11), SMO(1.94), and L1QP(1.94). Opt-RNN-DB has the highest ranking, followed by ISDA.

Table 4 gives the results of the Friedman test on the kernel SVM classifiers (SOM, L1QP, ISDA) considering different performance measures. The null hypothesis H_0^{ker} is rejected for all these kernel classifiers at a significance level of $\alpha = 0.05$, indicating that the proposed hybrid classifier outperforms all other kernel classifiers. In this regard, the performance of ISDA-SVM is the closest to that of the Opt-RNN-DBSVM classifier.

Table 4. Results of Friedman test on the kernel SVM classifiers.

Opt-RNN-DB vs.	*p*-Value (Accuracy)	*p*-Value (F1-Score)	*p*-Value (Precision)
SOM	0.008	0.004	0.004
L1QP	0.008	0.004	0.004
ISDA	0.003	0.011	0.011

5.3. Opt-RNN-DBSVM vs. Non-Kernel Classifiers

In this section, we compare Opt-RNN-DBSVM to several non-kernel classifiers, namely Naive Bayes [45], MLP [46], KNN [47], AdaBoostM1 [48], Decision Tree [49], SGD Classifier [50], Nearest Centroid Classifier [50], and Classical SVM [51].

Tables A1–A5 give the values of the measures accuracy, F1-score, precision, and recall for the considered datasets. Considering each of these performance measures, Opt-RNN-DBSVM permits remarkable improvements and clearly outperforms the other methods. The large number of classifiers and datasets makes it difficult to demonstrate this superiority without employing statistical methods.

To analyze these results further and to evaluate the performance of non-kernel classifiers (Naive Bayes (NB), MLP, KNN, AdaBoostM1 (ABM1), Nearest Centroid Classifier (NCC), Decision Tree (DT), SGD Classifier (SGDC)) compared to Opt-RNN DBSVM, regardless of the data type, we perform the Friedman test to verify the statistical significance of the proposed method compared to other methods with respect to the derived mean rankings. Three performance measures are considered (accuracy, F1-score, precision).

The null hypothesis is given by H_0^{nker}, i.e., "The classifiers NB, MLP, KNN, ABM1, NCC, DT, SGDC, Opt-RNN-DBSVM perform similarly in mean rankings without a significant difference".

The considered degree of freedom of the Friedman test is 7 (number of non-kernel classifiers -1). The significance level is 0.05 and the considered confidence interval is 95%.

Considering the accuracy measure, the average rank of the four kernel methods is given in brackets: Opt-RNN-DBSVM(7.80), ABM1(5.2), KNN(5.2), NB(4.4), NCC(3.95), SGDC(3.55), DT(3.5), and MLP(2.4). Opt-RNN-DB has the highest ranking, followed by ABM1(5.2) and KNN(5.2). Considering the F1-score, the average rank of the four kernel methods is given in brackets: Opt-RNN-DBSVM(7.1), ABM1(5.2), KNN(5.05), NB(4.8), SGDC(4.6), NCC(4.05), DT(3.1), and MLP(2.1). Opt-RNN-DB has the highest ranking, followed by ABM1 and KNN. Considering the precision measure, the average rank of the four kernel methods is given in brackets: Opt-RNN-DBSVM(7.1), NCC(5.3), ABM1(5.15),

KNN(4.9), NB(4.25), SGDC(3.95), DT(3.25), and MLP(2.1). Opt-RNN-DB has the highest ranking, followed by NCC and ABM1. Table 5 gives the results of the Friedman test on the non-kernel classifiers (NB, MLP, KNN, ABM1, NCC, DT, and SGDC) considering three performance measures (accuracy, F1-score, and precision). The null hypothesis H_0^{nker} is rejected for all these classifiers at a significance level of $\alpha = 0.05$, indicating that the proposed hybrid classifier outperforms all other non-kernel classifiers. In this regard, the performance of ABM1 is the closest to that of the Opt-RNN-DBSVM classifier.

Table 5. Results of Friedman test on non-kernel classifiers.

Opt-RNN-DB vs.	*p*-Value (Accuracy)	*p*-Value (F1-Score)	*p*-Value (Precision)
Naive Bayes	0.054	0.022	0.026
MLP	0.00	0.00	0.00
KNN	0.018	0.061	0.045
AdaBoostM1	0.018	0.083	0.075
Nearest Centroid Classifier	0.012	0.15	0.1
Decision Tree	0.002	0.007	0.012
SGD Classifier	0.003	0.022	0.004

Additional comparison studies were performed on the PIMA and Germany Diabetes datasets and the ROC curves were used to calculate the AUC for the best performance obtained from each non-kernel classifier. Figures A4 and A6 show the comparison of the ROC curves of the classifiers DT, KNN, MLP, NB, etc., and the Opt-RNN-DBSVM method, evaluated on the PIMA dataset. We point out that Opt-RNN-DBSVM quickly converges to the best results and obtains more true positives and a smaller number of false positives compared to several other classification methods.

More comparisons are given in Appendix B; Figures A5 and A7 show the comparison of the ROC curves of the classical SVM and Opt-RNN-DBSVM methods, evaluated on the Germany Diabetes dataset. More specifically, considering the performance measures "false positive rate" and "true positive rate", predictions based on support vectors produced by Opt-RNN-DBSVM dominate predictions based on support vectors produced by other non-kernel classifiers.

6. Conclusions

The main challenges of SVM implementation are the number of local minima and the amount of computer memory required to solve the dual-SVM, which increase exponentially with respect to the size of the dataset. The Kernel-Adatron family of algorithms, ISDA and SMO, has handled very large classification and regression problems. However, these methods treat noise, boundary, and kernel samples in the same way, resulting in a blind search in unpromising areas. In this paper, we have introduced a hybrid approach to deal with these drawbacks, namely Optimal Recurrent Neural Network and Density-Based Support Vector Machine (Opt-RNN-DBSVM), which performs in six phases: the characterization of different samples, the elimination of samples having a weak probability of being support vectors, building an appropriate recurrent neural network based on an original energy function, and solving the differential equation system governing the RNN dynamics, using the Euler–Cauchy method implementing an optimal time step. Data preprocessing reduces the number of local minima in the dual-SVM; this reduction enables real-time decision making in big data problems. The RNN's recurring architecture avoids the need to explore recently visited areas; this is an implicit tabu search. With the optimal time step, the search moves from the current vectors to the best neighboring support vectors. On one hand, two main, interesting fundamental results were demonstrated: the convergence of RNN-SVM to feasible solutions and the fact that Opt-RNN-DBSVM has

very low time complexity compared to Const-RNN-SVM, SMO-SVM, ISDA-SVM, and L1QP-SVM. On the other hand, several experimental studies were conducted based on well-known datasets (IRIS, ABALONE, WINE, ECOLI, BALANCE, LIVER, SPECT, SEED, PIMA). Based on popular performance measures (accuracy, F1-score, precision, recall), Opt-RNN-DBSVM outperformed Const-RNN-SVM, KA-SVM, and some non-kernel models (cited in Table A1). In fact, Opt-RNN-DBSVM improved the accuracy by up to 3.43%, F1-score by up to 2.31%, precision by up to 7.52%, and recall by up to 6.5%. In addition, compared to SMO-SVM, ISDA-SVM, and L1QP-SVM, Opt-RNN-DBSVM provides a reduction in the number of support vectors by up to 32%, which permits us to save memory for large applications that implement several machine learning models. The main problem encountered in the implementation of Opt-RNN-DBSVM is the determination of the Lagrange parameters involved in the SVM energy function. In this sense, a genetic strategy will be introduced to determine these parameters considering each dataset. In future work, extensions of this method may include combining Opt-RNN-DBSVM with big data technologies to accelerate classification tasks on big data and introducing hybrid versions based on Opt-RNN, deep learning, and fuzzy-SVM.

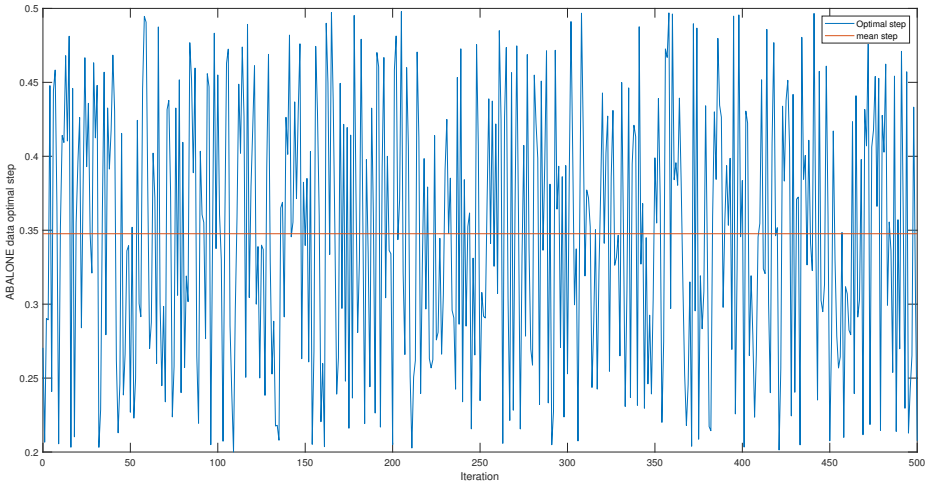
Author Contributions: Conceptualization, K.E.M.; Validation, A.O. and V.P.; Investigation, M.C.; Data curation, M.C.; Writing—original draft, A.E.O.; Visualization, M.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Ministry of National Education, Professional Training, Higher Education and Scientific Research and the Digital Development Agency (DDA) and CNRST of Morocco (No. Alkhawarizmi/2020/23).

Data Availability Statement: Data will only be shared upon request.

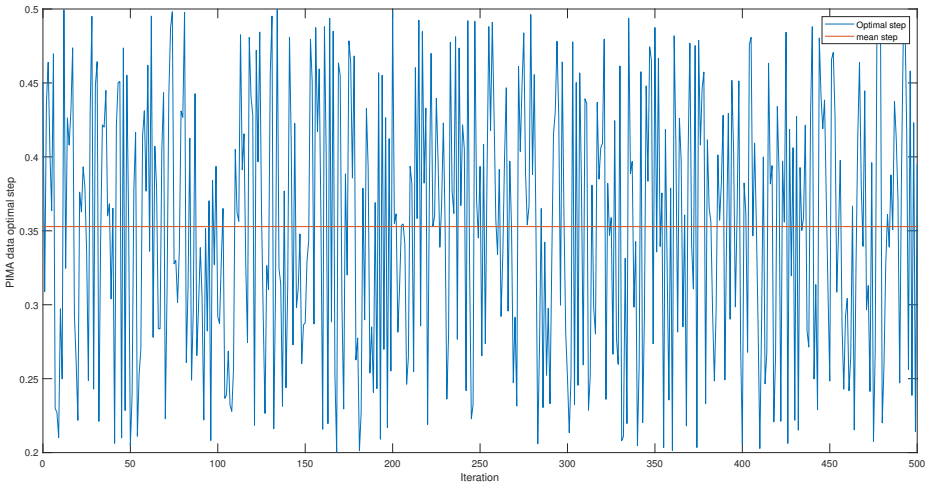
Conflicts of Interest: The authors declare that they have no conflict of interest.

Appendix A. Optimal Steps

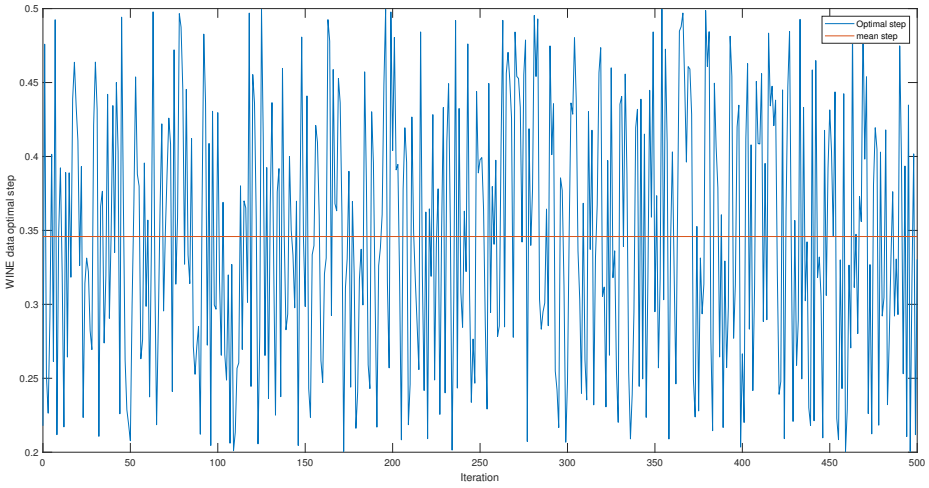


(**a**)

Figure A1. *Cont.*

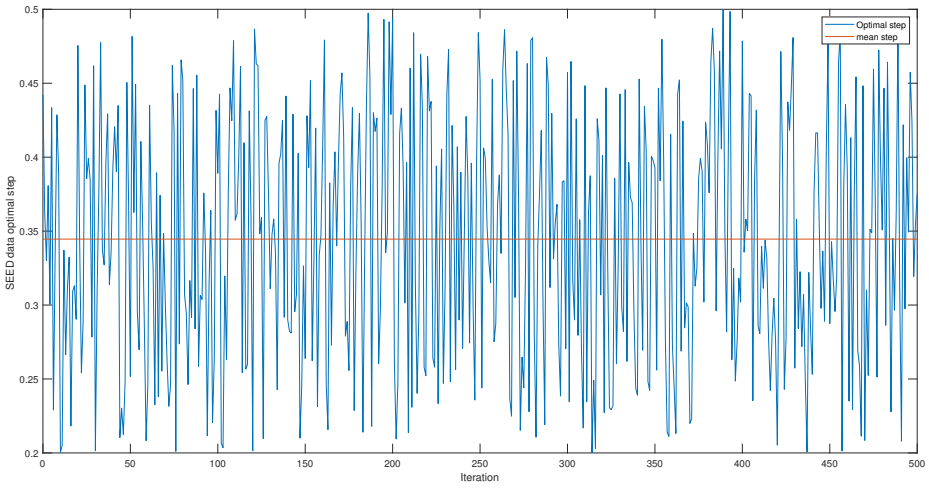


(**b**)



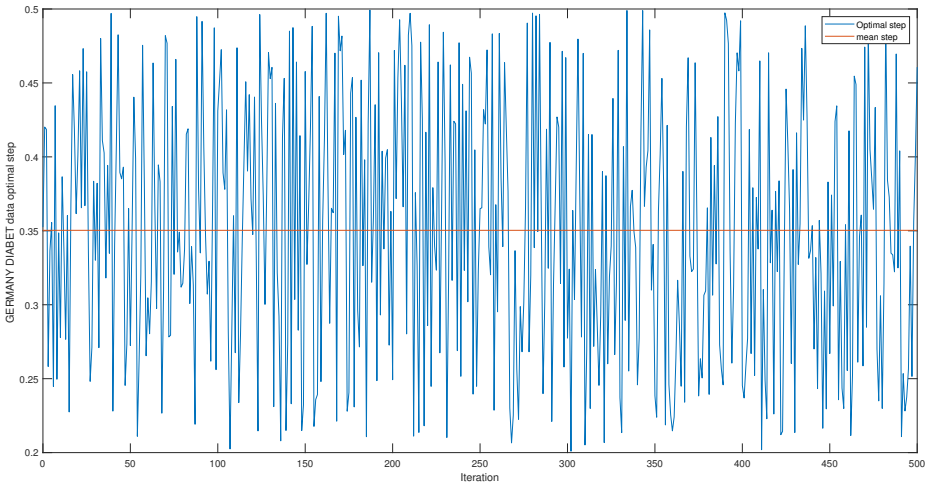
(**c**)

Figure A1. (**a**) ABALONE dataset, (**b**) PIMA dataset, (**c**) WINE dataset.

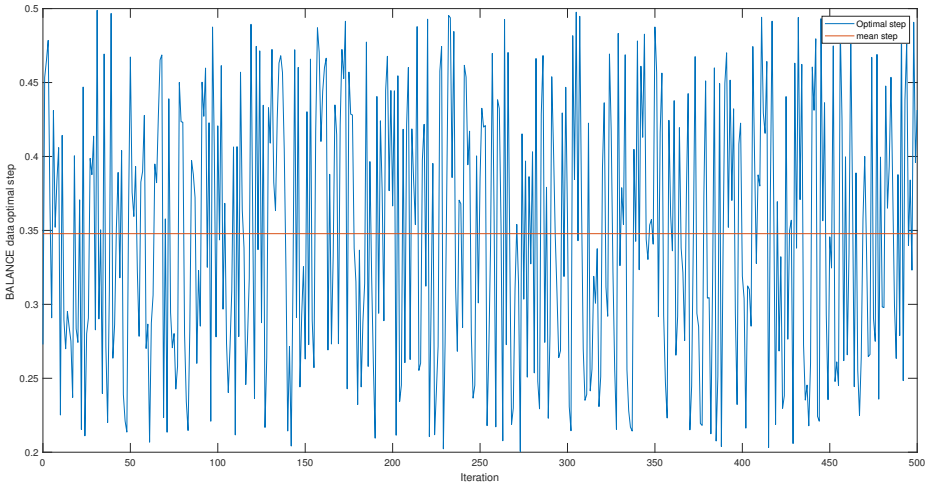


(**a**)

Figure A2. *Cont.*

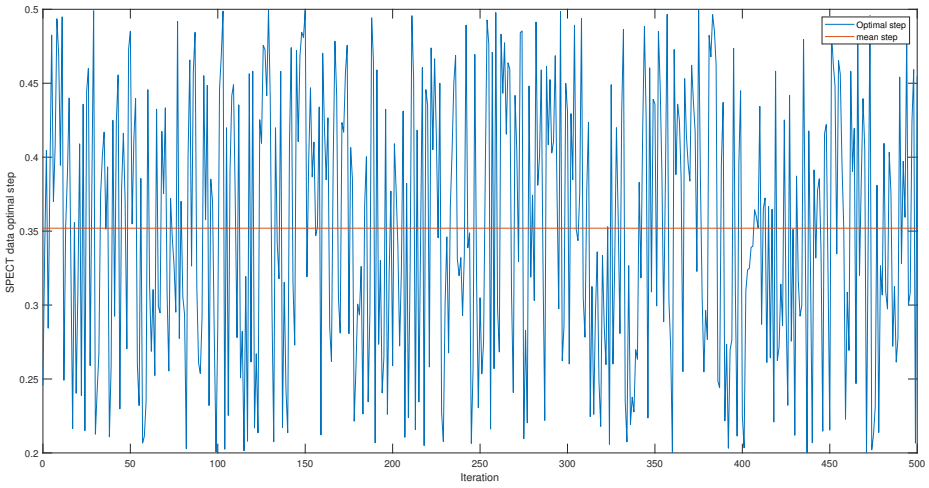


(**b**)



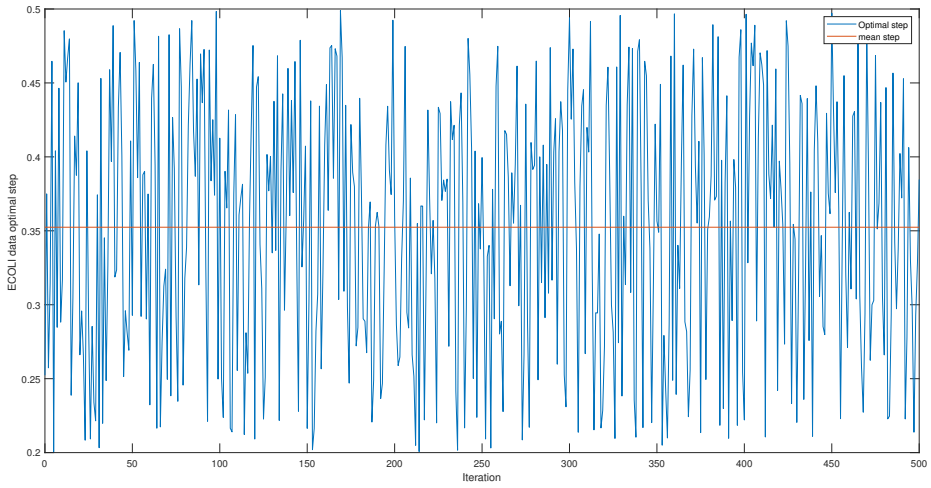
(**c**)

Figure A2. (**a**) SEED dataset, (**b**) Germany Diabetes dataset, (**c**) BALANCE dataset.

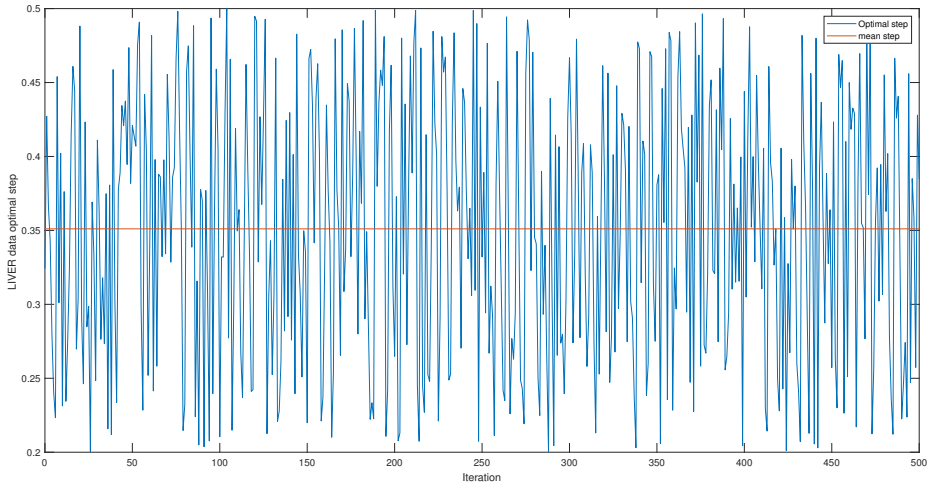


(**a**)

Figure A3. *Cont.*



(**b**)



(**c**)

Figure A3. (**a**) SPECT dataset, (**b**) ECOLI dataset, (**c**) LIVER dataset.

Appendix B. RUC Curves

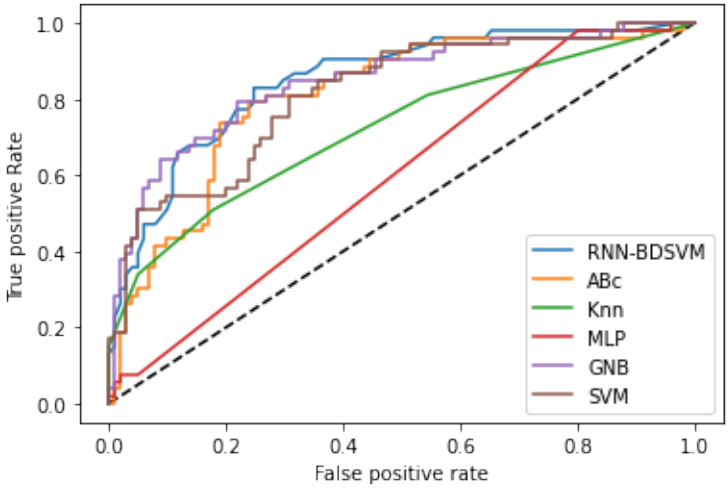


Figure A4. RUC curve for the different classification methods applied to PIMA Diabetes dataset.

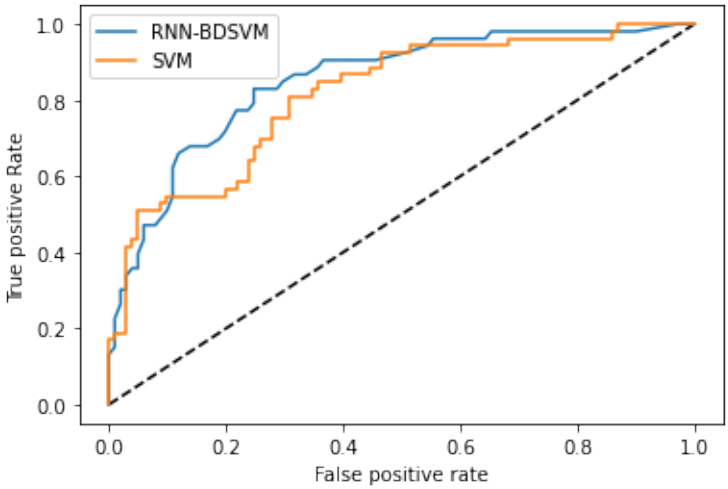


Figure A5. Opt-RNN-DBSVM vs. SVM RUC curve applied to PIMA Diabetes dataset.

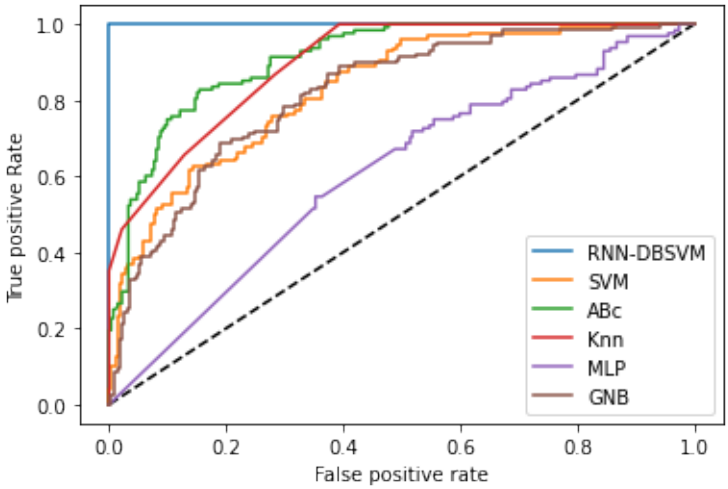


Figure A6. RUC curve for the different classification methods applied to Germany Diabetes dataset.

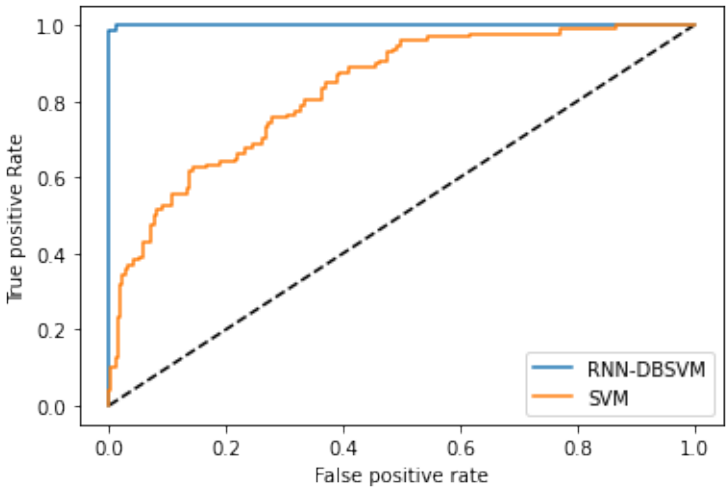


Figure A7. Opt-RNN-DBSVM vs. SVM RUC curve applied to Germany Diabetes dataset.

Appendix C. Opt-RNN-DBSVM and Non-Kernel Classifiers

Table A1. Comparison between Opt-RNN-DBSVM and different classification methods on the IRIS and ABALONE datasets.

IRIS				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	90.00	87.99	77.66	1.00
MLP	26.66	0.00	0.00	0.00
KNN	96.66	95.98	91.62	1.00
AdaBoostM1	86.66	83.66	71.77	1.00
Decision Tree	69.25	76.12	70.01	69.55
SGD Classifier	76.66	46.80	1.00	30.10
Random Forest Classifier	90.00	87.99	77.66	1.00
Nearest Centroid Classifier	96.66	95.98	91.62	1.00
Classical SVM	96.66	95.98	91.62	1.00
Opt-RNN-DBSVM	97.96	92.19	95.85	96.05
ABALONE				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	68.89	51.19	41.37	67.33
MLP	62.91	47.63	36.32	47.63
KNN	81.93	53.74	70.23	43.02
AdaBoostM1	82.29	55.99	70.56	55.06
Decision Tree	76.79	51.33	52.06	49.63
SGD Classifier	80.86	64.74	58.08	70.57
Nearest Centroid Classifier	76.07	64.79	62.60	61.15
Random Forest Classifier	82.28	57.56	71.11	48.34
Classical SVM	80.98	40.38	82.00	27.65
Opt-RNN-DBSVM	98.38	96.07	96.18	93.19

Table A2. Comparison between Opt-RNN-DBSVM and different classification methods on the WINE and ECOLI datasets.

WINE				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	78.48	75.96	74.17	74.89
MLP	67.73	72.67	51.52	43.64
KNN	81.84	79.85	78.35	79.00
AdaBoostM1	88.20	70.60	81.64	76.21
Decision Tree	83.02	81.42	79.36	80.22
SGD Classifier	68.40	83.95	52.28	44.81
Nearest Centroid Classifier	73.44	70.75	72.33	71.23
Classical SVM	79.49	78.26	73.52	74.97
Opt-RNN-DBSVM	96.47	95.96	96.08	96.02

Table A2. *Cont.*

ECOLI				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	82.08	97.11	1.00	95.66
MLP	55.22	71.45	55.00	1.00
KNN	89.55	99.87	1.00	97.02
AdaBoostM1	70.14	87.21	77.73	1.00
Decision Tree	70.14	83.71	82.03	84.19
SGD Classifier	86.56	96.33	1.00	92.01
Random Forest Classifier	85.07	96.11	97.00	95.66
Nearest Centroid Classifier	82.08	97.28	1.00	95.39
Classical SVM	88.05	97.77	97.83	97.99
Opt-RNN-DBSVM	91.82	88.46	90.77	91.19

Table A3. Comparison between Opt-RNN-DBSVM and different classification methods on the BALANCE and LIVER datasets.

BALANCE				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	79.3	67.2	62.50	69.40
MLP	76.6	61.10	57.40	68.30
KNN	80.90	68.40	64.50	66.60
AdaBoostM1	81.10	69.20	70.60	66.50
Decision Tree	79.90	64.80	72.80	68.70
SGD Classifier	69.03	65.62	66.15	65.83
Nearest Centroid Classifier	66.54	65	66.66	64.89
Classical SVM	79.70	70.7	55.60	62.70
Opt-RNN-DBSVM	91.31	90.33	89.54	90.89
LIVER				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	71.66	71.90	71.45	70.80
MLP	61.50	63.15	72.70	68.88
KNN	50.50	71.20	69.81	53.90
AdaBoostM1	88.50	89.37	89.99	79.39
Decision Tree	39.26	45.39	48.38	48.76
SGD Classifier	49.80	60.00	49.49	50.22
Nearest Centroid Classifier	66.50	63.30	60.20	61.87
Classical SVM	80.40	77.67	77.90	70.08
Opt-RNN-DBSVM	88.10	85.95	86.00	85.50

Table A4. Comparison between Opt-RNN-DBSVM and different classification methods on the SPECT and SEED datasets.

SPECT				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	63.15	77.40	96.33	65.94
MLP	97.36	99.60	97.77	1.00
KNN	92.10	96.80	97.80	95.90
AdaBoostM1	94.73	97.99	97.55	97.50
Decision Tree	86.84	93.89	97.55	89.48
SGD Classifier	97.36	99.60	97.77	1.00
Random Forest Classifier	97.36	99.60	97.77	1.00
Nearest Centroid Classifier	57.89	72.20	1.00	72.28
Classical SVM	97.36	99.60	97.77	1.00
Opt-RNN-DBSVM	97.36	99.60	97.77	1.00
SEED				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	85.71	79.76	92.55	69.20
MLP	30.95	0.00	0.00	0.00
KNN	85.71	79.24	73.39	85.40
AdaBoostM1	95.23	93.40	87.44	1.00
Decision Tree	92.85	90.88	1.00	81.00
Random Forest Classifier	92.85	90.88	1.00	81.00
SGD Classifier	85.71	86.76	79.55	94.20
Nearest Centroid Classifier	85.71	79.28	92.70	75.77
Classical SVM	85.71	83.43	92.70	75.04
Opt-RNN-DBSVM	96.88	97.09	96.76	1.00

Table A5. Comparison between Opt-RNN-DBSVM and different classification methods on the PIMA and Germany Diabetes datasets.

PIMA				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	79.3	70.40	74.2	66.50
MLP	66.23	13.33	57.1	8.40
KNN	74.90	66.60	68.4	64.50
AdaBoostM1	72.72	60.37	60.8	60.80
Decision Tree	70.77	52.63	60.2	47.60
SGD Classifier	37.66	52.47	36.98	1.00
Nearest Centroid Classifier	63.63	48.14	47.01	49.55
Classical SVM	79.22	61.90	84.7	49.6
Opt-RNN-DBSVM	79.87	68.04	75.90	62.05

Table A5. *Cont.*

Germany Diabetes Dataset				
Method	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Naive Bayes	81.66	87.19	79.37	77.33
MLP	63.28	54.24	50.39	56.40
KNN	68.08	67.88	76.30	66.00
AdaBoostM1	91.95	90.66	92.56	90.06
Decision Tree	55.66	53.74	59.93	50.02
SGD Classifier	79.96	79.74	70.08	78.57
Nearest Centroid Classifier	86.71	82.63	89.32	85.63
Classical SVM	78.5	59.81	74.10	50.99
Opt-RNN-DBSVM	99.5	99.7	1.00	98.00

References

1. Steyerberg, E.W. *Clinical Prediction Models*; Springer International Publishing: Cham, Switzerland, 2019; pp. 309–328.
2. Law, A.M. How to build valid and credible simulation models. In Proceedings of the 2019 Winter Simulation Conference (WSC), National Harbor, MD, USA, 8–11 December 2019; pp. 1402–1414.
3. Glaeser, E.L.; Kominers, S.D.; Luca, M.; Naik, N. Big data and big cities: The promises and limitations of improved measures of urban life. *Econ. Inq.* **2018**, *56*, 114–137. [CrossRef]
4. Minoux, M. *Mathematical Programming: Theories and Algorithms*; Wiley: Hoboken, NJ, USA, 1983.
5. El Moutaouakil, K.; Roudani, M.; El Ouissari, A. Optimal Entropy Genetic Fuzzy-C-Means SMOTE (OEGFCM-SMOTE). *Knowl.-Based Syst.* **2023**, *262*, 110235. [CrossRef]
6. Huang, T.-M.; Kecman, T.M. Bias Term b in SVMs Again. In Proceedings of the 12th European Symposium on Artificial Neural Networks, Bruges, Belgium, 28–30 April 2004.
7. Kecman, V.; Vogt, T.-M.H. On the Equality of Kernel AdaTron and Sequential Minimal Optimization in Classification and Regression Tasks and Alike Algorithms for Kernel Machines. In Proceedings of the 11th European Symposium on Artificial Neural Networks, ESANN, Bruges, Belgium, 23–25 April 2003; pp. 215–222.
8. Platt, J. *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*; Microsoft Research Technical Report MSR-TR-98-14; MIT Press: Boston, MA, USA, 1998.
9. Vogt, M. *SMO Algorithms for Support Vector Machines without Bias, Institute Report*; Institute of Automatic Control, TU Darmstadt: Darmstadt, Germany, 2002. Available online: http://www.iat.tu-darmstadt.de/vogt (accessed on 1 June 2023).
10. Frieß, T.-T.; Cristianini, N.; Campbell, I.C.G. The Kernel-Adatron: A Fast and Simple Learning Procedure for Support Vector Machines. In Proceedings of the 15th International Conference on Machine Learning, Madison, WI, USA, 24–27 July 1998; Shavlik, J., Ed.; Morgan Kaufmann: San Francisco, CA, USA, 1998; pp. 188–196.
11. Anlauf, J.K.; Biehl, M. The AdaTron—An adaptive perceptron algorithm. *Europhys. Lett.* **1989**, *10*, 687–692. [CrossRef]
12. Joachims, T. Making large-scale svm learning. Practical Advances. In *Kernel Methods-Support Vector Learning*; MIT-Press: Cambridge, MA, USA, 1999.
13. Osuna, E.; Freund, R.; Girosi, F. An Improved Training Algorithm for Support Vector Machines. In Proceedings of the Neural Networks for Signal Processing VII, Proceedings of the 1997 Signal Processing Society Workshop, Amelia Island, FL, USA, 24–26 September 1997; pp. 276–285.
14. Veropoulos, K. Machine Learning Approaches to Medical Decision Making. Ph.D. Thesis, The University of Bristol, Bristol, UK, 2001.
15. Kecman, V.; Huang, T.M.; Vogt, M. Iterative single data algorithm for training kernel machines from huge data sets: Theory and performance. In *Support Vector Machines: Theory and Applications*; Springer: Berlin, Germany, 2005; pp. 255–274.
16. Haddouch, K.; El Moutaouakil, K. New Starting Point of the Continuous Hopfield Network. In Proceedings of the Big Data, Cloud and Applications: Third International Conference, BDCA 2018, Kenitra, Morocco, 4–5 April 2018; pp. 379–389.
17. Hopfield, J.J.; Tank, D.W. Neural computation of decisions in optimization problems. *Biol. Cybern.* **1985**, *52*, 1–25. [CrossRef]
18. Alotaibi, Y. A new meta-heuristics data clustering algorithm based on tabu search and adaptive search memory. *Symmetry* **2022**, *14*, 623. [CrossRef]
19. El Ouissari, A.; El Moutaouakil, K. Density based fuzzy support vector machine: Application to diabetes dataset. *Math. Model. Comput.* **2021**, *8*, 747–760. [CrossRef]
20. Moutaouakil, K.E.; Yahyaouy, A.; Chellak, S.; Baizri, H. An Optimized Gradient Dynamic-Neuro-Weighted-Fuzzy Clustering Method: Application in the Nutrition Field. *Int. J. Fuzzy Syst.* **2022**, *24*, 3731–3744. [CrossRef]

21. Aghbashlo, M.; Peng, W.; Tabatabaei, M.; Kalogirou, S.A.; Soltanian, S.; Hosseinzadeh-Bandbafha, H.; Lam, S.S. Machine learning technology in biodiesel research: A review. *Prog. Energy Combust. Sci.* **2021**, *85*, 100904.
22. Ahmadi, M.; Khashei, M. Generalized support vector machines (GSVMs) model for real-world time series forecasting. *Soft Comput.* **2021**, *25*, 14139–14154. [CrossRef]
23. Xie, X.; Xiong, Y. Generalized multi-view learning based on generalized eigenvalues proximal support vector machines. *Exp. Syst. Appl.* **2022**, *194*, 116491. [CrossRef]
24. Tanveer, M.; Rajani, T.; Rastogi, R.; Shao, Y.H.; Ganaie, M.A. Comprehensive review on twin support vector machines. In *Annals of Operations Research*; Springer: Berlin, Germany, 2022; pp. 1–46.
25. Mercer, J. Functions of positive and negative type, and their connection the theory of integral equations. *Philos. Trans. R. Soc. Lond. Ser. Contain. Pap. Math. Phys. Character* **1909**, *209*, 415–446.
26. Schölkopf, B.; Smola, A.J.; Bach, F. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*; MIT Press: Cambridge, MA, USA, 2002.
27. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
28. Ettaouil, M.; Elmoutaouakil, K.; Ghanou, Y. The Placement of Electronic Circuits Problem: A Neural Network Approach. *Math. Model. Nat. Phenom.* **2010**, *5*, 109–115. [CrossRef]
29. El Moutaouakil, K.; Ahourag, A.; Chakir, S.; Kabbaj, Z.; Chellack, S.; Cheggour, M.; Baizri, H. Hybrid firefly genetic algorithm and integral fuzzy quadratic programming to an optimal Moroccan diet. *Math. Model. Comput.* **2023**, *10*, 338–350. [CrossRef]
30. Abdellatif, E.O.; El Moutaouakil, K.; Hicham, B.; Saliha, C. Intelligent local search for an optimal control of diabetic population dynamics. *Math. Models Comput. Simul.* **2022**, *14*, 1051–1071. [CrossRef]
31. Lin, C.F.; Wang, S.D. Fuzzy support vector machines. *IEEE Trans. Neural Netw.* **2002**, *13*, 464–471.
32. Schölkopf, B.; Smola, A.J.; Williamson, R.C.; Bartlett, P.L. New support vector algorithms. *Neural Comput.* **2000**, *12*, 1207–1245. [CrossRef] [PubMed]
33. Schölkopf, B.; Platt, J.C.; Shawe-Taylor, J.; Smola, A.J.; Williamson, R.C. Estimating the support of a high-dimensional distribution. *Neur. Comp.* **2001**, *13*, 1443–1471. [CrossRef]
34. Bi, J.; Zhang, T. Support vector classification with input data uncertainty. In Proceedings of the Advances in Neural Information Processing Systems 18 Neural Information Processing Systems, NIPS 2005, Vancouver, Canada, 5–8 December 2005; pp. 161–168.
35. Hazarika, B.B.; Gupta, D. Density-weighted support vector machines for binary class imbalance learning. *Neural Comput. Appl.* **2021**, *33*, 4243–4261. [CrossRef]
36. Guo, H.; Wang, W. Granular support vector machine: A review. *Artif. Intell. Rev.* **2019**, *51*, 19–32. [CrossRef]
37. Lee, Y.J.; Mangasarian, O.L. SSVM: A smooth support vector machine for classification. *Comput. Optim. Appl.* **2001**, *20*, 5–22. [CrossRef]
38. Laxmi, S.; Gupta, S.K. Multi-category intuitionistic fuzzy twin support vector machines with an application to plant leaf recognition. *Eng. Appl. Artif. Int.* **2022**, *110*, 104687. [CrossRef]
39. El Moutaouakil, K.; Ettaouil, M. Reduction of the continuous Hopfield architecture. *J. Comput.* **2012**, *4*, 64–70.
40. Hopfield, J.J. Neurons with graded response have collective computational properties like those of two-states neurons. *Proc. Natl. Acad. Sci. USA* **1984**, *81*, 3088–3092. [CrossRef] [PubMed]
41. El Moutaouakil, K.; El Ouissari, A.; Touhafi, A.; Aharrane, N. An Improved Density Based Support Vector Machine (DBSVM). In Proceedings of the 2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech), Marrakesh, Morocco, 24–26 November 2020; pp. 1–7.
42. Moutaouakil, K.E.; Touhafi, A. A New Recurrent Neural Network Fuzzy Mean Square Clustering Method. In Proceedings of the 2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech), Marrakesh, Morocco, 24–26 November 2020; pp. 1–5.
43. Kecman, V. Iterative k data algorithm for solving both the least squares SVM and the system of linear equations. In Proceedings of the SoutheastCon 2015, Fort Lauderdale, FL, USA, 9–12 April 2015; pp. 1–6.
44. Dua, D.; Graff, C. *UCI Machine Learning Repository*; University of California, School of Information and Computer Science: Irvine, CA, USA, 2021. Available online: http://archive.ics.uci.edu/ml (accessed on 1 June 2023).
45. Wickramasinghe, I.; Kalutarage, H. Naive Bayes: Applications, variations and vulnerabilities: A review of literature with code snippets for implementation. *Soft Comput.* **2021**, *25*, 2277–2293. [CrossRef]
46. Tolstikhin, I.O.; Houlsby, N.; Kolesnikov, A.; Beyer, L.; Zhai, X.; Unterthiner, T.; Dosovitskiy, A. Mlp-mixer: An all-mlp architecture for vision. In Proceedings of the Advances in Neural Information Processing Systems 34, 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Online, 6–14 December 2021.
47. Shokrzade, A.; Ramezani, M.; Tab, F.A.; Mohammad, M.A. A novel extreme learning machine based kNN classification method for dealing with big data. *Expert Syst. Appl.* **2021**, *183*, 115293. [CrossRef]
48. Chen, P.; Pan, C. Diabetes classification model based on boosting algorithms. *BMC Bioinform.* **2018**, *19*, 109. [CrossRef]
49. Charbuty, B.; Abdulazeez, A. Classification based on decision tree algorithm for machine learning. *J. Appl. Sci. Technol. Trends* **2021**, *2*, 20–28. [CrossRef]

50. Almustafa, K.M. Classification of epileptic seizure dataset using different machine learning algorithms. *Inform. Med. Unlocked* **2020**, *21*, 100444. [CrossRef]
51. Vapnik, V. *The Nature of Statistical Learning Theory*; Springer Science and Business Media: Berlin, Germany, 1999.