IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2021.Doi Number

Kernel Parameter Optimization for Support Vector Machine Based on Sliding Mode Control

Maryam Yalsavar [1], Paknoosh Karimaghaee [2], Akbar Sheikh-Akbari [3], Mohammad-Hassan Khooban [4], Jamshid Dehmeshki [5] and Salah Al-Majeed [6]

[1]School of Mathematics and Computer Science, University of Waterloo, Waterloo, Canada
[2]School of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran
[3] School of Built Environment, Engineering and Computing, Leeds Beckett University, Leeds, United Kingdom.
[4] Department of Engineering, Aarhus University, Aarhus, Denmark
[5] School of Computer Science and Mathematics Kingston University London, UK
[6] School of Computer Science, University of Lincoln, Lincoln, UK
Corresponding author: Akbar Sheikh-Akbari (e-mail: A.Sheikh-Akbari@leedsbeckett.ac.uk)

ABSTRACT Support Vector Machine (SVM) is a supervised machine learning algorithm, which is used for robust and accurate classification. Despite its advantages, its classification speed deteriorates due to its large number of support vectors when dealing with large scale problems and dependency of its performance on its kernel parameter. This paper presents a kernel parameter optimization algorithm for Support Vector Machine (SVM) based on Sliding Mode Control algorithm in a closed-loop manner. The proposed method defines an error equation and a sliding surface, iteratively updates the Radial Basis Function (RBF) kernel parameter or the 2-degree polynomial kernel parameters, forcing SVM training error to converge below a threshold value. Due to the closed-loop nature of the proposed algorithm, key features such as robustness to uncertainty and fast convergence can be obtained. To assess the performance of the proposed technique, ten standard benchmark databases covering a range of applications were used. The proposed method and the state-of-the-art techniques were then used to classify the data. Experimental results show the proposed method is significantly faster and more accurate than the anchor SVM technique and some of the most recent methods. These achievements are due to the closed-loop nature of the proposed algorithm, which significantly has reduced the data dependency of the proposed method.

INDEX TERMS Support vector machine, Sliding mode control, RBF kernel, 2-degree polynomial kernel, Optimal parameter, Classification speed.

I. INTRODUCTION

Support Vector Machine (SVM) is one of the widely used machine learning classification algorithms, among other classifiers such as: nearest neighbor [1], boosted decision trees [2], regularized logistic regression [3], neural networks [4], and random forests [5]. SVM can be used to achieve robust and accurate classification results, even from non-linearly separable input data, by mapping the data into a higher-dimensional space using kernels [6-7]. SVM is a Quadratic Programming (QP) problem that is aimed at finding a separating hyperplane to achieve maximum margin between classes of data [8-9]. It was first proposed for binary classification by Vapnik in the early 1990s, however, its extensions can be used for multi category problems [10]. Since SVM achieves a unique solution and can learn independently from the dimensionality of feature space, it is robust against overfitting and it is superior to other classifiers [6][10]. SVM has been used in many applications, including text categorization [11] and face detection [12], where it delivers robust and accurate results. SVM has also been used in some control branches, e.g., nonlinear control [13] and optimal control [14], because of the unique and optimal answer that it generates. Despite the advantages and wide range of applications of SVM, it suffers from some limitations such as low classification speed, especially when dealing with large scale problems, due to the large number of support vectors that SVM uses for classification [15-16], dependency of its performance on kernel parameter, kernel selection and its regularization parameter. SVM's test phase time complexity is $O(1) + 4O(n) + 2O(n^3)$, where n is the number of support vectors [10]. This indicates that the SVM classification computation cost increases as its number of

support vectors increases. Various methods have been proposed by the researchers to find optimal kernel for SVM and reducing its number of support vectors as the performance and speed of the algorithm depend on the kernel function and its parameters. These techniques can be classified into two main groups called: closed-loop and open-loop methods, where they either try to find the optimal kernel function and its parameters or dealing with some of the SVM's problems by modifying the training set or its set of support vectors. Closed-loop systems/algorithms have a feedback in their structure so that when a control input (input) changes the output of the system/algorithm, the resulting output is used for correcting and changing the control input (input) for arriving at the desired output. They operate in a self-adjusting mode, while open-loop systems/algorithms need a person to manually review and make the adjustments. Therefore, a close loop system/algorithm converges faster than open loop systems and is more robust to uncertainties and disturbances [17].

The closed loop-based methods for finding optimal kernel function and its parameters mainly use two approaches to achieve this. The group 1 methods first introduce an objective function, which is dependent on SVM and kernel parameters, then use different gradient descent methods to find optimal parameters for the kernel functions [18-23]. The group 2 methods try to find the global optimal solution for the kernel and its regularization parameters [24-31]. Since the goal is arriving at a global solution, they use various optimization algorithm including genetic-, dragonfly- and evolutionary-algorithms with different fitness functions. Genetic Algorithm (GA), Ant Colony Optimization (ACO) algorithm and Particle Swarm Optimization (PSO) algorithm are all Swarm Intelligence (SI) based methods, that one of their main properties is acting in a self-organized mode, and their capability to evolving the components into a good form without any external help. GA is population-based strategy which mainly includes five components: a random number generator, a fitness evaluation unit, a reproduction process, a crossover process, and a mutation operation. It first creates an initial population by random or heuristic, then determines the fitness and performance of each individual, and ranks them using a fitness function. When all individuals are ranked, the resulting low ranked individuals are omitted from the population, and the rest will be used in the reproduction process. GA uses confounded parameter settings, which is one of its main positive points, however, using a random procedure in the crossover and mutation process reduces the GA's convergence speed towards the optimal values, which is considered as its biggest drawback.

ACO is a metaheuristic approach, which has four main components: ant, pheromone, daemon action, and decentralized control. The ACO tries to find the shortest path to the optimal solution in a weighted graph. Hence, in the first step of each iteration every ant constructs its own solution (path) stochastically, then the paths that are built by different ants are compared and in the last step the level of each edge's pheromone is updated. The ACO algorithm can be used in dynamic applications due to its great adaptation to changes such as new distances and suggests a positive feedback results in rapid discovery of good solutions. However, it has slower convergence speed compared with other heuristic-based methods and its theoretical analysis is difficult, research is experimental rather than theoretical and lacks a centralized processor to guide the algorithm towards good solutions. PSO is an optimization technique that is inspired by swarm behavior in birds flocking and fish schooling for searching global optimal solutions. The PSO algorithm first initializes the population, then calculates the fitness value for everyone. After finding all fitness values, it updates the population, the speed, and particles' position. Except its first step, the other steps are repeated till termination condition is satisfied. The PSO has no mutation calculation, and its searching speed is very fast, but it cannot address scattering and non-coordinate system problems, and it is less exact at the regulation of its speed and the direction [32-33]. Although GA, ACO, and PSO algorithms are acting in a self-organized mode. They are not purely and truly closed-loop methods. For clarification, a block diagram representation of a closed-loop and open loop system are shown in Fig. 1.

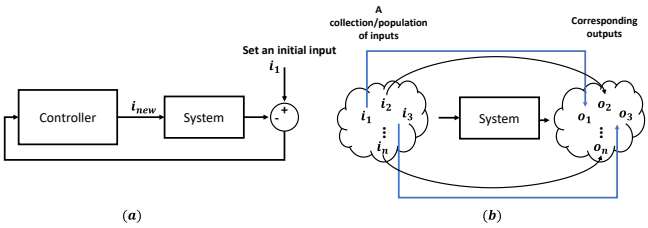

FIGURE 1. A block diagram of a) a closed loop-, b) an open loop-system.

As it can be seen from Fig. 1b, in an open loop system, a collection of inputs (population) is fed to the system and the input (s) that create the best output, will be used for controlling the system or generating the new set of the inputs. This is exactly the procedure in SI based algorithms, while in a closed-loop system (Fig. 1.a) after choosing an initial value as the input, the closed-loop structure will update the input value based on the resulting output. In a closed-loop method, the best input value is not selected by comparing different potential inputs.

Sliding Mode Control (SMC) is a closed-loop method, which benefits from this great property, because acting in a closed-loop manner brings more robustness against disturbances, uncertainties, and un-modeling, and has a superiority to those of SI based algorithms from this aspect. Unlike GA and ACO, the SMC has a vivid, simple, and well-defined theory and mathematics behind itself, which makes it possible to theoretically analysis it. Unlike GA, SMC has no randomness in its structure, and based on the results it just

takes around 8 steps on average to arrive at its best result for different datasets, which is significantly faster than other methods. By defining the SVM algorithm as a closed-loop control system, it provides capability to control and monitor its transient and steady state behavior in detail. In the proposed method, Sliding Mode Control (SMC), which is a powerful tool for robust control of nonlinear systems, is used. Since there are uncertainties in the modeling of real-world systems, it is hard to control such plants with uncertain models and arrive at the desired performance. The SMC is often used to deliver good tracking in systems with uncertain models [34-35]. For achieving this goal an error equation and a sliding surface are first defined and the SMC then tries to drive the state trajectory of the system onto the sliding surface and force the trajectory to maintain on this surface for all subsequent time by using a control input. The state trajectory is driven to the sliding surface by just estimating that it is in which part of the sliding surface. There is no strict region, and it is not important that how far the state is from the sliding surface. Due to this feature of the SMC, it is more robust than other approaches in the control field. Hence, the uncertainties of the system model do not affect its performance and its convergence is guaranteed [36]. Hence, in this research the application of the SMC in improving the performance of the SVM algorithm and reducing its limitation when dealing with large data, which are coming from different fields with no information and knowledge about their dynamic, is investigated.

In this paper, a support vector machine based on sliding mode control RBF kernel parameter optimization is presented. The proposed method does not need a system model to find the optimum value for the RBF kernel parameter and 2-degree polynomial kernel for speeding up the test phase of SVM and improving its prediction accuracy. The proposed method first defines the specification of an error equation and the sliding surface and then it tries to arrive at a good tracking and low training error by updating the parameter(s) of those kernels. This procedure will be repeated until the validation accuracy continues its decreasing trend for a specific number of iterations. The effort of the proposed method to achieve high training accuracy results in prediction accuracy enhancement and finding a smaller set of support vectors, thereby reducing the speed of classification. Experimental results show that the anchor SVM does not necessarily generate the optimal number of support vectors and its kernel parameter selection could affect both accuracy and its resulting number of support vectors, where an optimal and smaller set of support vectors will increase both the speed and accuracy of the classification. Hence, the proposed method is significantly faster and more accurate than the anchor SVM technique. Furthermore, the proposed method generated more accurate results in compared with some of the latest techniques. All of these and its high robustness against uncertainties, which are existed in the data comes from different sources, are due to the closed-loop nature of the SMC algorithm used in conjunction with SVM method.

The main contribution of this paper can be summarized as follows:

(i) Looking at SVM's problems and concepts from a control field of view and making a connection between these two fields.

(ii) Using a non-model based and close loop method, SMC, for finding the optimal value for RBF kernel parameter.

The rest of this paper is organized as follows. In Section II-III, a brief overview of SVM and SMC methods are presented, respectively. In Section IV, the proposed method for finding the optimum kernel parameters is explained. Experimental results are presented in Section V and Section VI concludes the paper.

II. SUPPORT VECTOR MACHINE

There are many methods that can be used to classify two-class linearly separable data but all of them give infinite answers. To find the best answer, the SVM method could be one of the solutions. The SVM finds the best hyperplane that separates the data using the idea that the best decision boundary is the one that has the maximum distance and margin from both classes of the data. SVM called maximum margin classifier, too. SVM has been shown to produce accurate results that can be explained easily, unlike other methods, e.g., neural networks. If the data known to be linearly separable, hard margin SVM is usually used. Assume that there are n data points in the dataset that their labels are either -1 or 1. The first step is to find its margin and then maximize it. If the equation of hyperplane be $w^T x + b = 0$, where w is an orthogonal vector to the hyperplane and b is the bias then the distance of a point to the hyperplane can be formulated as:

$$d_i(x) = \frac{w^T x_i + b}{\|w\|} \qquad \forall i = 1, \cdots, n. \tag{1}$$

where x_i is the ith data point and $d_i(x)$ is its signed distance. It means if the data is on one side of the hyperplane, its sign will be positive, otherwise its sign is negative. By multiplying the distance of each point by its label, an unsigned distance, $y_i d_i(x)$ is calculated, where y_i is the label of the data. To find the margin, $\min\left\{y_i \frac{w^T x_i + b}{\|w\|}\right\}$ is determined. w and b can be rescaled in a way that distance of all points to the hyperplane become at least one so the margin drives as follow:

$$margin = min\left\{y_i \frac{w^T x_i + b}{\|w\|}\right\} \quad and$$

$$y_i(w^T x_i + b) \geq 1 \xrightarrow{yields} margin = \frac{1}{\|w\|}. \tag{2}$$

SVM is searching for the maximum margin. So, based on eq. 2, the problem can be formulated as following quadratic problem:

$$\min_w \frac{1}{2}\|w\|^2 \tag{3}$$
$$s.t.\ y_i(w^T x_i + b) \geq 1 \quad \forall i = 1, \cdots, n.$$

Quadratic Problem (QP) is a convex problem that results in a global minimum or global maximum solution. By solving this QP problem, both w and the hyperplane are calculated. Classifying nonlinear data with a linear algorithm like SVM can be done by reshaping and increasing the dimension of the data, resulting in a linear dataset. However, increasing the dimensionality of the data, the curse of dimensionality will appear. SVM uses the kernel concept in case of nonlinear data to benefit of dimension enhancement but gets rid of its curse [37]. In the case that SVM is used for classifying nonlinear data, it called soft margin SVM. In this case, the decision boundary is nonlinear because the data is not linearly separable. It means that some points cross the margin or locate in the other side of the hyperplane and cause misclassification like the one that shows in Fig. 2.

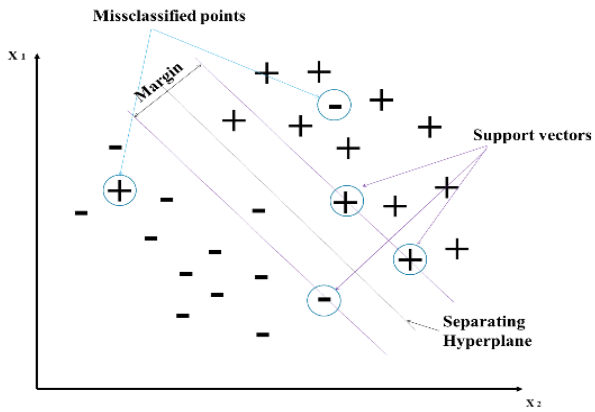

FIGURE 2. **A nonlinearly separable classification problem.**

So, the constraint in hard margin SVM is not valid anymore because some points have $y_i(w^T x_i + b) \leq 1$. The constraint is changed to include these cases or points, too. The nonlinear case problem is formulated as follows [38-39]

$$\min_{w} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \qquad (4)$$
$$s.t.\ y_i(w^T x_i + b) \geq 1 - \xi_i \quad where\ \xi_i \geq 0\ and\ \forall i = 1,\cdots,n.$$

In (4), ξ_i is added to the constraint for the points that violate the constraint. But by changing the constraint in this way all points can violate this. So, the number of points that can violate the margin restricted by adding a penalty or regularization parameter, C. One can solve the dual form of eq. (4) as:

$$\max_{\alpha_i} \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j \qquad (5)$$
$$s.t.\ \sum_{i=1}^{n}\alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \quad \forall i = 1,\cdots,n$$

where α_i is the dual variable that obtains via the QP. The points that their α_i is greater than zero are support vectors and the points that their α_i is equal to C are the ones that violate the constraint in hard margin SVM.

Besides the advantages of SVM, due to the lack of a control perspective on the SVM problem, there are many aspects that are ignored. By studying SVM from a control point of view, the kernel function and its parameters are like the inputs of the SVM algorithm along with data, and that the algorithm finds support vectors as the output of SVM by using them in its training mode. So, both kernel and its parameters are vitally importance in SVM. Their unwise selection will result in poor set of support vectors, which increases the test error and time.

It can be concluded that by using control methods, the inputs of the SVM algorithm can be found in a way that increase both performance and accuracy of the SVM algorithm. As both model and dynamic of the datasets are unknown, model-based methods of control theory are not applicable. Therefore, Sliding Mode Control (SMC), which is not a model-based algorithm and is highly robust to the dynamic of the data and is a closed-loop procedure, seems to be one of the solutions to speed up the algorithm when dealing with large nonlinear data. Moreover, both soft margin and hard margin problems have counterparts in control theory because they both grapple with training error in different ways. In hard margin SVM, a zero-training error is desired, while in soft margin SVM, a limited non-zero value error is acceptable; these two trends are achieved by defining some constraint in the SVM. In control theory, there are many procedures for managing the error, e.g., using integral of absolute/square error or paying attention to the transient behavior of the error besides its steady-state error, while in SVM mainly, steady state error is considered. In addition, there is a vast variety of control methods for dealing with the steady-state errors like methods in classical control, robust control, adaptive control, optimal control, nonlinear control, and intelligent control. In the next sections, after a brief introduction, SMC as a suitable robust control strategy will be used to develop desired kernel functions. Other control algorithms can be applied in the same way.

III. SLIDING MODE CONTROL

Sliding Mode Control (SMC) is a powerful tool for robust control of nonlinear systems [40]. It is based on the idea that controlling 1st-order systems are much easier than controlling n^{th} −order systems, so by defining a notation, an nth order system is reformulated as a 1st-order model [34]. This provides the construction of a sliding surface and drives the states of the system on it in the state space. Once the sliding surface is reached, the SMC keeps the states of the system on the close neighborhood of the sliding surface [40]. SMC consists of two part: the sliding surface, and the off-surface dynamics. The first step to drive this controller is to examine the expression of the error [40]. For the single input dynamic system of form $y = f(x) + b(x)u$, where y is the output, u is the input signal, $f(x)$ and $b(x)$ are system model, which are not exactly specified and have uncertainties. The goal is tracking the desired signal y_d by output, y. So, the error expression can be written as follows:

$$e = y - y_d \tag{6}$$

where y and y_d are the output and desired output, respectively. A time-varying surface $S(y;\ t)$ in the state space R can then be defined by the scaler space $S(y;\ t) = 0$, where:

$$S(y;t) = (\frac{d}{dy} + \lambda)^{n-1} e \tag{7}$$

where λ is a strictly positive constant and for $n = 2$, Eq. (7) can be written as: $S = \dot{e} + \lambda e$. The problem of tracking $y \equiv y_d$ is equivalent to that of remaining on the surface S(t) for all $t > 0$; indeed $S(y,t) \equiv 0$ represents a linear differential equation whose unique solution is $e \equiv 0$, given its initial condition. Thus, the problem of tracking the n-dimensional vector y_d can be reduced to that of keeping the scalar quantity S at zero.

$$S = \dot{e} + \lambda e \equiv 0 \tag{8}$$

when the surface is driven to zero, the error drives to zero too, for $t \to \infty$ [40]. To show that, we work backward by postulating that the off-surface dynamics must be of the form:

$$\dot{S} = -f(S) \tag{9}$$

where $f(S)$ can be any non-decreasing odd function. This shows that the change in S and the 'distance' of the current state of the sliding surface, it is always opposite the sign of the S. The control input should force the states to approach it. So, $\dot{S}$ must be a function of our control input, u. $\dot{S}$ must also be a function of the second derivative of the error, $\ddot{e}$, to just be a function of the input, u, this implies that S should only be a function of error, e, and its first derivative, $\dot{e}$. The simplest form of such a function that guarantees $e \to 0$ as $t \to \infty$ is given in Eq. (8) [40]. Consequently, driven of S to zero, drives the tracking error, e, to zero, too. For Eq. (8) the sliding surface is a line with a slope of $-\lambda$ in the phase plane. By starting from any initial condition, the state trajectory drives to the sliding surface and then it slides along the surface exponentially towards the desired value, y_d, with a time constant of $1/\lambda$ [34]. This procedure is shown in Fig. 3.

SVM has widely used to classify non-linear separable data where there is always some uncertainty in selection of its parameters such as regularization and kernel. This has inspired the author to use the concept of sliding mode control to improve the performance of the SVM algorithm.

I. PROPOSED ALGORITHM

SVM uses the kernel function to increase the dimension of the data and make the data linearly separable in the resulting high dimension space. However, the desired kernel function or its parameters are not specified, as a result, various methods have been introduced to find the best kernel function and its parameters to increase the performance of SVM. There is a variety of kernel functions and some of their well-known

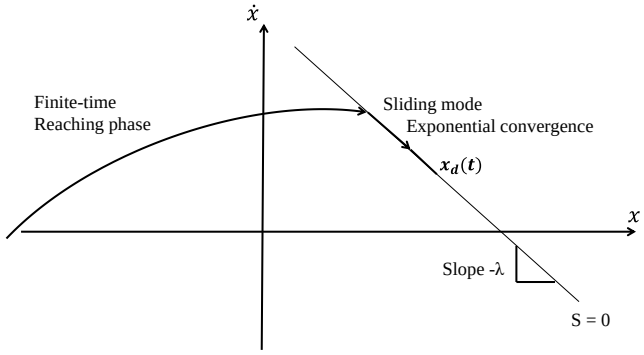

FIGURE 3. The state trajectory approaches to the sliding surface and its slide along the surface towards the desired value, x_d (Graphical configuration of eq. 8) [34].

functions are RBF kernels and polynomial kernels, where different combination of these functions, e.g., linear, and nonlinear, are used to extend SVM capability to deal with non-linear data. All these kernels have some parameters, which need to be chosen in an appropriate way to solve the mentioned problems. In this paper, the Sliding Mode Control (SMC) is used to find optimum parameters of the kernel functions. To prove the effectiveness and performance of the proposed method, without losing its generality, the γ parameter of the RBF kernel as an advanced form and parameters of a 2-degree polynomial kernel as a basic form are calculated using the proposed method. In sub-section A, the application of SMC to determine the optimum γ parameter of the RBF kernel is presented. In sub-section B, the SMC is used to compute the parameters of a polynomial kernel.

A. SLIDING MODE CONTROL BASED SUPPORT VECTOR MACHINE RADIAL BASIS FUNCTION'S KERNEL PARAMETER OPTIMIZATION

Fig. 4 shows a block diagram of the proposed Sliding Mode Control based Support Vector Machine Radial Basis Function's kernel parameter optimization (SMC-SVM-RBF) method. The proposed method takes the input data and splits the data into three subsets, named training, test, and validation subsets. Then SVM trains with training subset and by using the initial value of the mentioned parameters. The 'Train SVM' block takes training subset and initial parameters including Radial Basis Function (RBF) kernel parameter, γ_{new}, regularization parameters, C, λ, d, VE_{old}, which represent the state of the training error to train the SVM, generating some Support Vectors (SVs) and their numbers, N_{SVs}. The resulting classification information, SVs and N_{SVs} , are then used to classify the train and validation data subsets, separately.

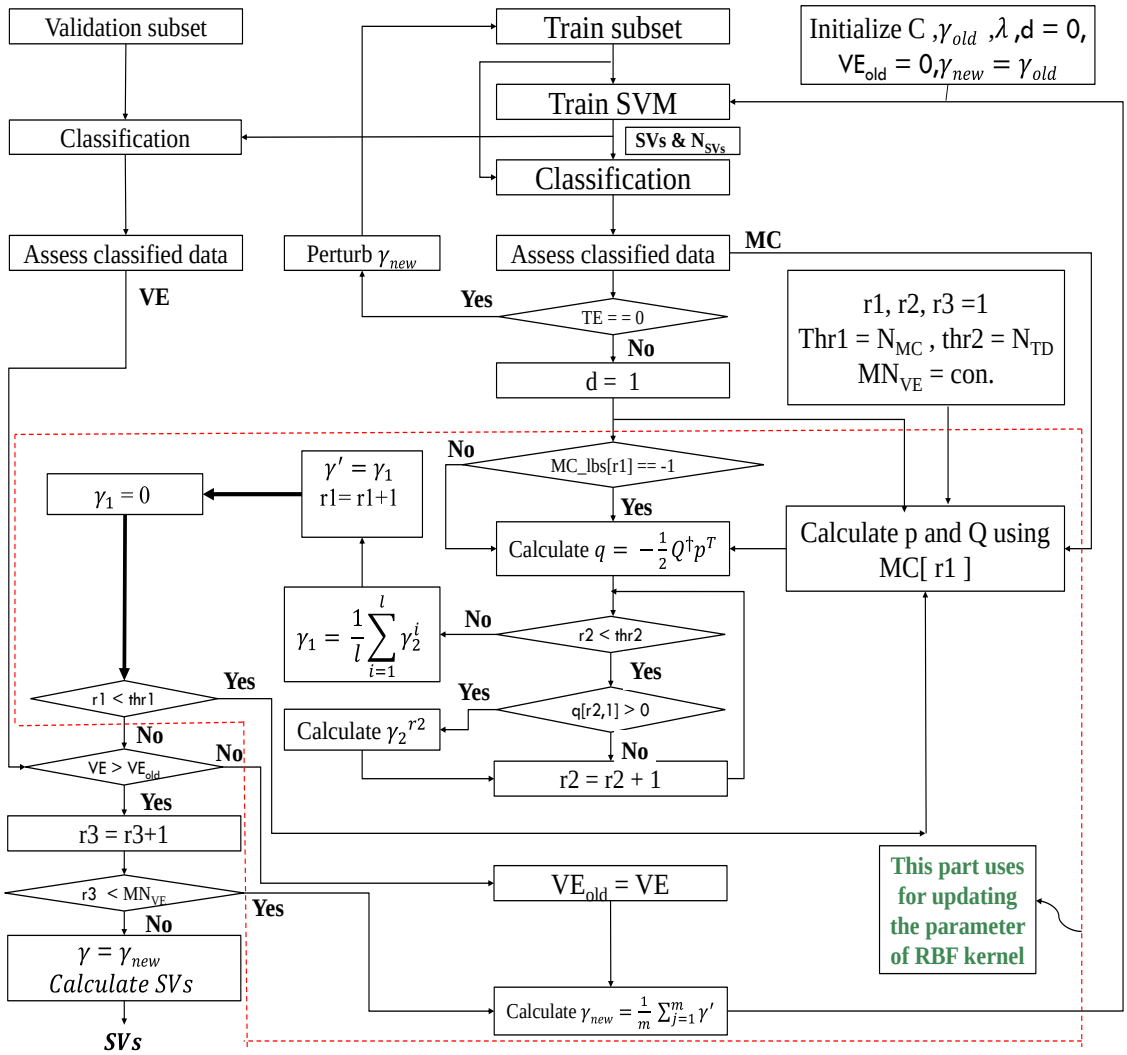

FIGURE 4. **Block diagram of the proposed algorithm.**

The resulting classified train and validation subsets data are the independently assessed and Mis-Classified training data (MC) and their Mis-Classified labels (MC-lbs), the Training Error (TE) of the classified training subset and Validation Error (VE) of the classified validation data are calculated. The calculated MC and MC-lbs parameters are used to update the RBF kernel parameter, TE is used to define the time to perturb the initial value of the RBF kernel parameter and VE is used to terminate the algorithm. For perturbing the value of γ, the algorithm checks the value of the TE. If it is zero, γ_{old} will be perturbed as follows until a non-zero training error is achieved: it checks the value of the RBF kernel parameter, if its value is smaller than a threshold, it perturbs the kernel parameter with a small value, otherwise it will be perturbed with a larger value. In this research, the process is started with a small initial RBF kernel parameter value and when then the training procedure starts updating γ as follows: It first initializes three counters named r1, r2, and r3 with values of 1, $thr1$ with the Number of MisClassified train data (NMC), and $thr2$ with the Number of Training Data (NTD) and the Maximum Number of acceptable iterations to improve the Validation Error ($MNVE$) with a constant value. Then the algorithm goes through each element of Mis-Classified training data using its label, MC-lbs[r1], calculating its p and Q. If MC-lbs[r1] = -1, q will be calculated using $q = -\frac{1}{2}Q^{\dagger}p^{T}$. After that the algorithm goes through elements of q using counter r2 and for each positive element of q , $\gamma_2{}^{r2}$ is calculated, when all elements of $\gamma_2{}^{r2}$ are calculated, it computes $\gamma_1 = \frac{1}{l}\sum_{i=1}^{l}\gamma_2^i$ but if MC-lbs[r1] in not equal to -1, it assigns γ_{new} to γ_1 . The algorithm then assigns γ_1 and 0 to γ' and γ_1 , respectively and increment r1 to point to the next misclassified train data. This procedure is repeated for all misclassified train data. When $\gamma'$is calculated for all misclassified train data, the algorithm will check r3, to see if r3 has reached its maximum number of iterations that are acceptable for improving the validation error (MNVE) threshold value. If not, a new value for γ is calculated as $\gamma_{new} = \sum_{j=1}^{m}\gamma'$ and it backs to 'Train SVM' block and the

procedure is repeated until MNVE reaches its predefined threshold value, otherwise the training is completed and γ_{new} is taken γ and use it to calculate the SVs. The resulting SVs are then used to classify the test subset.

The main aim of the proposed Sliding Mode Control based Support Vector Machine Radial Basis Function's kernel parameter optimization (SMC-SVM-RBF) is to use sliding mode control to find an optimum value for γ parameter of the RBF kernel to improve the SVM's performance in terms of its classification accuracy and speed. Mathematical prove of the proposed Sliding Mode Control based Support Vector Machine Radial Basis Function's kernel parameter optimization (SMC-SVM-RBF) method is detailed as follows: To make a relationship between SVM and SMC in this article, the error expression, considering equation (6), can be assumed as:

$$e^j = \frac{1}{2}\left|y^j - {y_d}^j\right|^2 \tag{10}$$

where e^j is the classification error, y_d^j and y^j are the desired and predicted output values for each training data point related to the j-th misclassified training data, respectively. Based on SVM algorithm y^j can be formulated as follows:

$$y^j(x) = sign\left[\sum_{i=1}^{n} \alpha_i^j y_i^j \, exp(-\gamma^j \|x - x_i\|^2) + \beta^j\right] \tag{11}$$

where α_i^j is a dual variable, y_i^j represent the output of the training data x_i, x is a misclassified training data but the aim is to find its true class label, $y^j(x)$ is the predicted class label for the misclassified training data, x, n is the number of the training data and β^j is a bias related to the jth misclassified training data. After defining the expression for error, sliding surface is defined by equation (8). In equation (8), γ is considered as the input and the aim is finding an optimum value for γ to minimize the training error. By calculating $\dot{S}$ from eq. 8 and replacing $\dot{S}$ with its value in eq. 9, the eq. 9 can be rewritten as:

$$\ddot{e} + \lambda\dot{e} = -f(S) \tag{12}$$

where $\ddot{e}$ and $\dot{e}$ are the second and first derivative of the error, e, respectively, $f(S)$ is time-varying surface in the state space R and λ is a strictly positive constant. From equation (12), it can be seen that the first and second derivatives of y are needed, as e is a function of y and y is a function of γ. Since sign function does not have a derivative, the sign function is replaced with a sigmoid function in different ways to solve and formulate this problem. As the algorithm uses misclassified training points to update the γ parameter, it may result in two types of misclassified data: a) the mis predicted label for the training point is -1 ($y = -1$), where its correct label should be 1. In this paper, $y = sigmoid(x)$ is considered as a function defining the belongness of a data point to the class Sign and sigmoid functions are illustrated in Figure 5. Figure 5 shows that the larger positive x values represent data with labels of 1 and when $x \to +\infty$, the data point is classified to $y = 1$ class. However, if $x \to -\infty$, y becomes zero, this implies that this data point is not belong to class $y = 1$. This misclassification is due to using unoptimized value for RBF parameter, γ, and α_i. SVM uses the sign function to determine

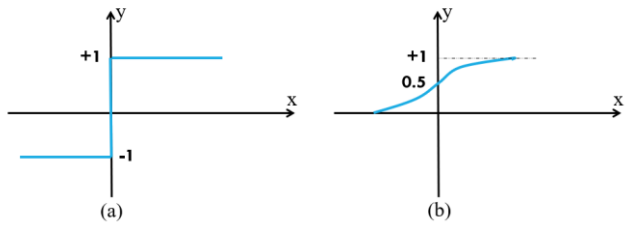

FIGURE 5. **illustration of (a) sign(x) and (b) sigmoid(x) function.**

the class of each data point within the dataset. However, the proposed method uses different functions to find accurate class for identified misclassified data points. For simplicity, in this article $sigmoid(x) = \frac{1}{1 + e^{-x}}$ and $-sigmoid(-x) = \frac{-1}{1 + e^{x}}$ functions, which are reversible functions with known derivative, are used to deal with misclassified data points in class -1 and 1, respectively. These two functions help to tackle the sign function irreversibility problem.

Using the first assumption, eq. 11 can be re-written as:

$$y(x) = sigmoid\left(\sum_{i=1}^{n} \alpha_i y_i \, exp(-\gamma\|x - x_i\|^2) + \beta\right) \tag{13}$$

and the derivative of eq. 13 can be written as:

$$\frac{dy}{dx} = y(x)(1 - y(x)) \tag{14}$$

Thus, by substituting eq. 13 into eq. 10 and removing j, which represent the jth data point in eq. 10 and eq 11, e, $\dot{e}$ and $\ddot{e}$ can be rewritten as:

$$e = \frac{1}{2}|y - y_d|^2$$

$$\dot{e} = |y - y_d|\frac{dy}{d\gamma}$$

$$\dot{e} = [|y - y_d| \sum_{i=1}^{n}(-\alpha_i y_i \|x - x_i\|^2) \, exp(-\gamma\|x - x_i\|^2)]y(1 - y) \tag{15}$$

where $x \in X$ is a misclassified training data point within the set of misclassified training data points, X, α_is are dual variables, y is predicted class label for the misclassified data point, x, y_d is the desirable class label for the misclassified data point, x, y_i is the true label of the training data point, x_i, and n is the total number of the training data points.

$$\ddot{e} = \left[\sum_{i=1}^{n} \alpha_i y_i \|x - x_i\|^2 \, exp(-\gamma\|x - x_i\|^2)\right]^2$$

$$y^2(1 - y)^2 sign(y - y_d)$$

$$+\left[|y - y_d| \sum_{i=1}^{n} \alpha_i y_i \|x - x_i\|^4 \, exp(-\gamma\|x - x_i\|^2)\right]$$

$$y(1 - y)$$

$$+|y-y_d|(1-y)$$
$$\left[\sum_{i=1}^{n}\alpha_i y_i\|x-x_i\|^2\, exp(-\gamma\|x-x_i\|^2)\right]^2$$
$$y(1-y)$$
$$-|y-y_d|y^2(1-y)$$
$$\left[\sum_{i=1}^{n}\alpha_i y_i\|x-x_i\|^2\, exp(-\gamma\|x-x_i\|^2)\right]^2$$
$$=\left[\sum_{i=1}^{n}\alpha_i y_i\|x-x_i\|^2\, exp(-\gamma\|x-x_i\|^2)\right]^2$$
$$(sign(y-y_d)y^2(1-y)^2+|y-y_d|[(1-y)^2y-y^2(1-y)])$$
$$+|y-y_d|(1-y)y$$
$$\sum_{i=1}^{n}\alpha_i y_i\|x-x_i\|^4\, exp(-\gamma\|x-x_i\|^2). \tag{16}$$

For simplification m_i, n_i, and $q_i$are defined as:

$$m_i=\alpha_i y_i\|x-x_i\|^2 \tag{17}$$
$$\frac{dy}{dx}=y(x)(1-y(x)) \tag{18}$$
$$q_i=exp(-\gamma\|x-x_i\|^2) \tag{19}$$

Now by replacing e and the resulting expression for $\dot{e}$ in eq. 8, S can be rewritten as:

$$S=\dot{e}+\lambda e$$
$$S=\left[|y-y_d|\sum_{i=1}^{n}(-\alpha_i y_i\|x-x_i\|^2)\, exp(-\gamma\|x-x_i\|^2)\right]$$
$$y(1-y)+\frac{\lambda}{2}|y-y_d|^2 \tag{20}$$

And by substituting (17), (18) and (19) into (15) and (16) and then substituting (15) and (16) into (12), $\dot{S}$ can be derived as:

$$\dot{S}=(sign(y-y_d)y^2(1-y)^2$$
$$+|y-y_d|[(1-y)^2y-y^2(1-y)])$$
$$[\textstyle\sum_{i=1}^{n}m_iq_i]^2+|y-y_d|(1-y)y$$
$$\textstyle\sum_{i=1}^{n}n_iq_i-\lambda|y-y_d|(1-y)y[\sum_{i=1}^{n}m_iq_i]=-f(S) \tag{21}$$

As these equations are derived for misclassified training data points with $y=-1$, by replacing y and y_d into eq. 21, it results in:

$$-16[\textstyle\sum_{i=1}^{n}m_iq_i]^2-4\lambda[\sum_{i=1}^{n}m_iq_i]-4\sum_{i=1}^{n}n_iq_i=-f(S) \tag{22}$$

To solve eq. 22, this equation is written in matrix form as follows:

$$-16(q^TMM^Tq)-4[N^T+\lambda*M^T]q=-f(S) \tag{23}$$

where

$$\begin{cases} M=[m_1,m_2,\dots,m_i\ ,\dots,m_n\ ]^T \\ N=[n_1,n_2,\dots,n_i,\dots,n_n]^T \\ q=[q_1,q_2,\dots,q_i,\dots,q_n]^T \\ \quad i=1,2,\cdots,n \\ (n\ is\ the\ total\ number\ of\ training\ data) \end{cases},$$

For simplicity, by assuming $Q=-16\,MM^T$ and $p=-4\,[N^T+\lambda*M^T\,]$, eq. 23. Can be rewritten as:

$$q^TQq+pq+f(S)=0 \tag{24}$$

where $Q\in R^{n*n}$, $p\in R^{1*n}$, and $f(S)$ are a matrix, a vector and a constant, respectively, where the value of $f(S)$ is calculated using the previous value of γ. The optimum value of q can be determined by calculating the derivative of eq. 24 with respect to q:

$$\frac{\partial\left(q^TQq+pq+f(S)\right)}{\partial q}=q^TQ+Qq+p^T$$
$$=0\xrightarrow{Q\ is\ a\ symetric\ matrix}2Qq=-p^T \tag{25}$$

As eq. 25 is an underdetermined problem, Q is not a full rank matrix and may have many solutions. In this paper, pseudo-inverse method is used to find an estimation for vector q. Since p^T is not in the column space of Q in general, the calculated q vector is an estimation of q, where the column space of Q, named as $C(Q)$ can be written as: $C(Q)=-16m_1M$ and $p=-4M^T[R^T+\lambda I]$ and R is an $n\ by\ n$ matrix. Consequently, q can be calculated using pseudo-inverse of Q as:

$$q=-\frac{1}{2}Q^{\dagger}p^T \tag{26}$$

where $Q^{\dagger}$ represents pseudo-inverse of Q and q vector can be determined by solving eq. 26. However, only the positive elements of q, which satisfy eq. 19, are acceptable. Using eq. 19, Γ vector can be calculated and written as follows:

$$\Gamma=[\gamma_2{}^1,\gamma_2{}^2,\dots,\gamma_2{}^i,\dots,\gamma_2{}^l]\qquad\forall i=1,\cdots,l. \tag{27}$$

where l is the number of positive elements of q vector and γ_2^i is the corresponding γ value of the ith positive element of q. By calculating the average of all elements of r vector, γ_1 is derived as:

$$\gamma_1=\frac{1}{l}\textstyle\sum_{i=1}^{l}\gamma_2{}^i \tag{28}$$

In the second stage, for the mis-predicted data with $y=1$, $y=-sigmoid(-x)$ function, which is illustrated in Fig. 6, is used to determine the level of belongness of each of these data points to their current class. From Fig. 6, it can be seen that data points with large negative x values are belonging to $y=-1$ class and data points with large positive values ($x\to+\infty$), which have $y=0$, are belonging to other class.

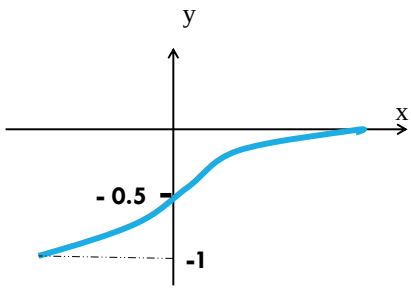

FIGURE 6. **An illustration of $y = -sigmoid(-x)$ function.**

By considering $-sigmoid(-x)$ function for these mis-classified data points, eq. 11 can be re-written as:

$$y(x) = -sigmoid(-\sum_{i=1}^{n} \alpha_i y_i \, exp(-\gamma\|x - x_i\|^2) - \beta) \tag{29}$$

and the derivative of eq. 29 can be written as:

$$\frac{dy}{dx} = -y(x)(1 + y(x)) \tag{30}$$

Thus, by substituting eq. 29 into eq. 10 and removing j, which represent the jth data point in eq. 10 and eq 11, e, $\dot{e}$ and $\ddot{e}$ can be rewritten as:

$$e = \frac{1}{2}|y - y_d|^2$$

$$\dot{e} = |y - y_d|\frac{dy}{d\gamma}$$

$$\dot{e} = -[|y - y_d|\sum_{i=1}^{n}(-\alpha_i y_i\|x - x_i\|^2)\, exp(-\gamma\|x - x_i\|^2)]y(1 + y) \tag{31}$$

$$\ddot{e} = \left[\sum_{i=1}^{n} \alpha_i y_i\|x - x_i\|^2 \, exp(-\gamma\|x - x_i\|^2)\right]^2 y^2(1 + y)^2 sign(y - y_d)$$
$$-\left[|y - y_d|\sum_{i=1}^{n} \alpha_i y_i\|x - x_i\|^4 \, exp(-\gamma\|x - x_i\|^2)\right] y(1 + y)$$
$$+|y - y_d|(1 + y)\left[\sum_{i=1}^{n} \alpha_i y_i\|x - x_i\|^2 \, exp(-\gamma\|x - x_i\|^2)\right]^2 y(1 + y)$$
$$+|y - y_d|y\left[\sum_{i=1}^{n} \alpha_i y_i\|x - x_i\|^2 \, exp(-\gamma\|x - x_i\|^2)\right]^2 y(1 + y)$$
$$= \left[\sum_{i=1}^{n} \alpha_i y_i\|x - x_i\|^2 \, exp(-\gamma\|x - x_i\|^2)\right]^2 (sign(y - y_d)y^2(1 + y)^2$$
$$+|y - y_d|[(1 + y)^2 y + y^2(1 + y)])$$
$$-|y - y_d|(1 + y)y\sum_{i=1}^{n} \alpha_i y_i\|x - x_i\|^4 \, exp(-\gamma\|x - x_i\|^2) \tag{32}$$

And by substituting (17), (18) and (19) into (31) and (32) and then substituting (31) and (32) into (12), and replacing y and y_d with 1 and -1, respectively, $\dot{S}$ can be derived as:

$$16[\sum_{i=1}^{n} m_i q_i]^2 - 4\lambda[\sum_{i=1}^{n} m_i q_i] - 4\sum_{i=1}^{n} n_i q_i = -f(S) \tag{33}$$

To solve eq. 33, assuming $Q = 16\, MM^T$ and $p = -4\,[N^T + \lambda M^T]$, eq. 33 can be written in matrix form as follows:

$$q^T Q q + pq + f(S) = 0 \tag{34}$$

where M, N and q were introduced in eq. 23. The resulting eq. 34 along with the procedures explained in eq. 25 to 28 are then used to calculate γ_1 parameter for this misclassified data point with $y = 1$.

For each mis-classified data point, based on its predicted y, one of the above-mentioned two methods is used to calculate its γ_1 value. The resulting γ_1s for all mis-classified data points are then put together to form the γ' vector, as follows:

$$\gamma' = [\gamma_1{}^1, \gamma_1{}^2, \dots, \gamma_1{}^i, \dots, \gamma_1{}^m] \qquad \forall i = 1, \cdots, m \tag{35}$$

where m is the total number of mis-classified data points and $\gamma_1{}^i$ represents γ_1 for mis-classified data point i.

Finally, a value for RBF kernel parameter, γ, is determined by calculating the average of γ' vector components using eq. 36:

$$\gamma = \frac{1}{m}\sum_{i=1}^{m} \gamma_1^i \tag{36}$$

where m is the total number of misclassified training data points. The resulting γ will be used as the RBF kernel parameter in the next iteration.

γ optimization procedure will be continued until the total number or iteration is reached or the validation error does not change for a pre-defined number of iterations.

B. POLYNOMIAL OPTIMAL KERNEL PARAMETER ESTIMATION USING SVM BASED ON SMC

Without losing the generality of the algorithm, the general form of a 2nd order polynomial kernel is considered as $(ax^T x_j + b)^2$, where a and b are the polynomial kernel parameters, x is a mis-classified training data point and x_j is the jth training data point for $j = 1, \dots, n$ and n is the total number of the training data points. The aim of this algorithm

is to find optimum polynomial parameters. In this paper, for simplicity a 2nd order polynomial was considered. However, a higher-order polynomial can also be used in a similar way. The procedure of the proposed SVM based on SMC algorithm for finding polynomial kernel optimum values is the same as the one that is explained for RBF method, which is illustrated in Fig. 4 with some differences. These differences are detailed as follows:

1. In this algorithm, polynomial kernel parameters, a and b, are first initialized with ones and then updated in each iteration. If using these initial values results in a zero-training error, their values are perturbed in the same way that was explained in Section IV.A for RBF parameter. These initial values were used because Zhang [23] and Zhiliang Liu [30] had also used them in their techniques and the performance of the proposed method in this paper, will be compared with their techniques.
2. Both resulting positive and negative values of q vector are acceptable in polynomial kernel parameter optimization, while only positive values of q vector were acceptable for updating RBF parameter optimization, as explained in Section IV.A.

To find optimum values for the polynomial kernel parameters, a and b, the procedure is started using eq. 8 and 12, where $sigmoid(x)$ and $-sigmoid(-x)$ functions are used for the two types of the mis-classified data points, $y = -1$ and $y = 1$, respectively. Hence, eq. 8 and 12 are derived using eq. 10 and 11 for each type of mis-classified data points, as follows:

1) PROCEDURE FOR FINDING OPTIMUM VALUE FOR a WHEN $y = -1$

By replacing the 2nd order polynomial kernel in eq. 13, $y(x)$ can be written as:

$$y(x) = sigmoid(\textstyle\sum_{i=1}^{n} \alpha_i y_i (ax^T x_i + b) + \beta) \tag{37}$$

And $\dot{e}$, $\ddot{e}$ can be derived by replacing eq 37 into eq. 10 and calculating 1st and 2nd derivative with respect to a.

$$\dot{e} = |y - y_d| \frac{dy}{da}$$

$$\dot{e} = \left[|y - y_d| \sum_{i=1}^{n} 2\alpha_i y_i x^T x_i (ax^T x_i + b) \right] y(1-y) \tag{38}$$

$$\ddot{e} = (sign(y - y_d) y^2 (1-y)^2 + |y - y_d| [(1-y)^2 y - y^2 (1-y)]) \left[\sum_{i=1}^{n} 2\alpha_i y_i x^T x_i (ax^T x_i + b) \right]^2 + |y - y_d| (1-y) y \sum_{i=1}^{n} 2\alpha_i y_i x^T x_i x^T x_i \tag{39}$$

Now by replacing e and the resulting expression for $\dot{e}$ in eq. 8, S can be rewritten as:

$$S = \dot{e} + \lambda e$$
$$S = [|y - y_d| \textstyle\sum_{i=1}^{n} 2\alpha_i y_i x^T x_i (ax^T x_i + b)] y(1-y) + \frac{\lambda}{2} |y - y_d|^2 \tag{40}$$

And by substituting eq. 38 and 39 into eq. 12, $\dot{S}$ can be derived as:

$$\dot{S} = \ddot{e} + \lambda \dot{e} = -f(S)$$
$$\dot{S} = (sign(y - y_d) y^2 (1-y)^2 + |y - y_d| [(1-y)^2 y - y^2 (1-y)]) \left[\sum_{i=1}^{n} 2\alpha_i y_i x^T x_i (ax^T x_i + b) \right]^2 + |y - y_d| (1-y) y \sum_{i=1}^{n} 2\alpha_i y_i x^T x_i x^T x_i + \lambda |y - y_d| (1-y) y$$
$$[\textstyle\sum_{i=1}^{n} 2\alpha_i y_i x^T x_i (ax^T x_i + b)] = -f(S) \tag{41}$$

Now by considering $m_i = 2\alpha_i y_i x^T x_i$, $n_i = 2\alpha_i y_i x^T x_i x^T x_i$ and $q_i = (ax^T x_i + b)$ and replacing $y = -1$ and $y_d = 1$ in eq. 41 and rewriting it in a matrix form, eq. 41 can be rewritten as:

$$q^T Q q + pq + f(S) + N^T r = 0 \tag{42}$$

where $Q = -16MM^T$, $p = -4\lambda M^T$, $N = -4[n_1, n_2, \dots, n_n]^T$, $M = [m_1, m_2, \dots, m_n]^T$, $r = [1,1,..,1]^T$ and n is the total number of the training data points. The optimum value of q can be determined by calculating the derivative of eq. 42 with respect to q:

$$\frac{\partial (q^T Q q + pq + f(S) + N^T r)}{\partial q} = q^T Q + Qq + p^T$$
$$= 0 \xrightarrow{Q \text{ is a symetric matrix}} 2Qq = -p^T$$

Now the q vector can be written as:

$$q = -\frac{1}{2} Q^{\dagger} p^T \tag{43}$$

where $Q^{\dagger}$ represents pseudo-inverse of Q and q vector can be determined by solving eq. 43. By using $q_i = (a_2^i x^T x_i + b)$, Γ vector is obtained as:

$$\Gamma = [a_2{}^1, a_2{}^2, \dots, a_2{}^i, \dots, a_2{}^l] \quad \forall i = 1, \dots, l \tag{44}$$

where l is the total number of q elements and a_2^i is the ith element of Γ and $a_2{}^i = \frac{q_i - b}{x^T x_i}$.

Finally, a_1 is computed by determining the average of all Γ elements:

$$a_1 = \frac{1}{l} \textstyle\sum_{i=1}^{l} a_2{}^i \tag{45}$$

2) PROCEDURE FOR FINDING OPTIMUM VALUE FOR b WHEN $y = -1$

By replacing $y(x)$ from eq. 37 into eq. 10 and calculating 1st and 2nd derivatives of the resulting e with respect to b, $\dot{e}$, and $\ddot{e}$ can be determined. Then by replacing the resulting e, $\dot{e}$, and $\ddot{e}$ into eq. 8 and 9, S and $\dot{S}$, can be derived, as follows:

$$S = \dot{e} + \lambda e$$

$$S = \left[|y - y_d| \sum_{i=1}^{n} 2\alpha_i y_i (ax^T x_i + b)\right] y(1-y) + \frac{\lambda}{2}|y - y_d|^2 \quad (46)$$

$$\dot{S} = \ddot{e} + \lambda \dot{e} = -f(S)$$

$$\dot{S} = (sign(y - y_d) y^2 (1-y)^2 + |y - y_d|[(1-y)^2 y - y^2(1-y)]) \left[\sum_{i=1}^{n} 2\alpha_i y_i (ax^T x_i + b)\right]^2 + |y - y_d|(1-y) y \sum_{i=1}^{n} 2\alpha_i y_i + \lambda |y - y_d|(1-y) y \left[\sum_{i=1}^{n} 2\alpha_i y_i (ax^T x_i + b)\right] = -f(S) \quad (47)$$

Assuming $m_i = 2\alpha_i y_i$ and $q_i = (ax^T x_i + b)$ and replacing $y = -1$ and $y_d = 1$ in eq. 47, and eq, 47 can be written in a matrix form as follows:

$$q^T Q q + pq + f(S) - 4M^T r = 0 \quad (48)$$

where $Q = -16MM^T$, $p = -4\lambda M^T$, $M = [m_1, m_2, \dots, m_n]^T$, $r = [1,1,..,1]^T$ and n is the total number of the training data points. The optimum value of q can be determined by calculating the derivative of eq. 48 with respect to q:

$$\frac{\partial(q^T Q q + pq + f(S) - 4M^T r)}{\partial q} = q^T Q + Qq + p^T$$

$$= 0 \xrightarrow{Q \text{ is a symetric matrix}} 2Qq = -p^T$$

Now the q vector can be written as:

$$q = -\frac{1}{2} Q^{\dagger} p^T \quad (49)$$

where $Q^{\dagger}$ represents pseudo-inverse of Q and q vector can be determined by solving eq. 49. By using $q_i = (x^T x_i + b_2^i)$, r vector is obtained as:

$$\Gamma = \left[\, {b_2}^1, {b_2}^2, \dots, {b_2}^i, \dots, b_2^l \right] \quad \forall i = 1, \dots, l \quad (50)$$

where l is the total number of q elements and b_2^i is the ith element of Γ and $b_2^i = q_i - x^T x_i$. Finally, the average of all Γ elements is defined as b_1, as follows:

$$b_1 = \frac{1}{l} \sum_{i=1}^{l} {b_2}^i \quad (51)$$

In the second stage, to find optimum values for the polynomial kernel parameters, a and b, the procedure is started using eq. 8 and 12, where $-sigmoid(-x)$ function is used for the mis-classified data points with $y = 1$. Hence, eq. 8 and 12 are derived using eq. 10 and 11, as follows:

3) PROCEDURE FOR FINDING OPTIMUM VALUE FOR a WHEN $y = 1$

By replacing the 2nd order polynomial kernel in eq. 13, $y(x)$ can be rewritten as:

$$y(x) = -sigmoid(-\textstyle\sum_{i=1}^{n} \alpha_i y_i (ax^T x_i + b) - \beta) \quad (52)$$

By replacing $y(x)$ from eq. 52 into eq. 10 and calculating 1st and 2nd derivative of the resulting e with respect to a, $\dot{e}$, and $\ddot{e}$ can be determined. Then by replacing the resulting e, $\dot{e}$, and $\ddot{e}$ into eq. 8 and 9, S and $\dot{S}$, can be derived, as follows:

$$S = \dot{e} + \lambda e$$

$$S = \left[-|y - y_d| \sum_{i=1}^{n} 2\alpha_i y_i x^T x_i (ax^T x_i + b)\right] y(1+y) + \frac{\lambda}{2}|y - y_d|^2 \quad (53)$$

$$\dot{S} = \ddot{e} + \lambda \dot{e} = -f(S)$$

$$\dot{S} = (sign(y - y_d) y^2 (1+y)^2 + |y - y_d|[(1+y)^2 y + y^2(1+y)]) \left[\sum_{i=1}^{n} 2\alpha_i y_i x^T x_i (ax^T x_i + b)\right]^2 - |y - y_d|(1 + y) y \sum_{i=1}^{n} 2\alpha_i y_i x^T x_i x^T x_i - \lambda |y - y_d|(1 + y) y \left[\sum_{i=1}^{n} 2\alpha_i y_i x^T x_i (ax^T x_i + b)\right] = -f(S) \quad (54)$$

Now by considering $m_i = 2\alpha_i y_i x^T x_i$, $n_i = 2\alpha_i y_i x^T x_i x^T x_i$ and $q_i = (ax^T x_i + b)$ and replacing $y = 1$ and $y_d = -1$ in eq. 54 and rewriting it in a matrix form, eq. 54 can be rewritten as:

$$q^T Q q + pq + f(S) + N^T r = 0 \quad (55)$$

where $Q = 16MM^T$, $p = -4\lambda M^T$, N= $-4[n_1, n_2, \dots, n_n]^T$, $M = [m_1, m_2, \dots, m_n]^T$, $r = [1,1,..,1]^T$ and n is the total number of the training data points. The optimum value of q can be determined by calculating the derivative of eq. 55 with respect to q:

$$\frac{\partial(q^T Q q + pq + f(S) + N^T r)}{\partial q} = q^T Q + Qq + p^T$$

$$= 0 \xrightarrow{Q \text{ is a symetric matrix}} 2Qq = -p^T$$

Now the q vector can be written as:

$$q = -\frac{1}{2}Q^{\dagger}p^T \tag{56}$$

where $Q^{\dagger}$ represents pseudo-inverse of Q and q vector can be determined by solving eq. 56. By using $q_i = (a_2^i x^T x_i + b)$, r vector is obtained as:

$$\Gamma = [a_2{}^1, a_2{}^2, \dots, a_2{}^i, \dots, a_2{}^l] \quad \forall i = 1, \dots, l \tag{57}$$

where l is the total number of q elements and a_2^i is the ith element of Γ and $a_2{}^i = \frac{q_i - b}{x^T x_i}$.

Finally, a_1 is computed by determining the average of all Γ elements:

$$a_1 = \frac{1}{l}\sum_{i=1}^{l} a_2{}^i \tag{58}$$

4) PROCEDURE FOR FINDING OPTIMUM VALUE FOR b WHEN $y = 1$

By replacing $y(x)$ from eq. 52 into eq. 10 and calculating 1st and 2nd derivatives of the resulting e with respect to b, $\dot{e}$, and $\ddot{e}$ can be determined. Then by replacing the resulting e, $\dot{e}$, and $\ddot{e}$ into eq. 8 and 9, S and $\dot{S}$, can be derived, as follows:

$$S = \dot{e} + \lambda e$$

$$S = \left[-|y - y_d| \sum_{i=1}^{n} 2\alpha_i y_i (ax^T x_i + b)\right] y(1 + y) + \frac{\lambda}{2}|y - y_d|^2 \tag{59}$$

$$\dot{S} = \ddot{e} + \lambda\dot{e} = -f(S)$$

$$\dot{S} = (sign(y - y_d)y^2(1 + y)^2 + |y - y_d|[(1 + y)^2 y + y^2(1 + y)])\left[\sum_{i=1}^{n} 2\alpha_i y_i (ax^T x_i + b)\right]^2 - |y - y_d|(1 + y)y \sum_{i=1}^{n} 2\alpha_i y_i - \lambda|y - y_d|(1 + y)y\left[\sum_{i=1}^{n} 2\alpha_i y_i (ax^T x_i + b)\right] = -f(S) \tag{60}$$

Assuming $m_i = 2\alpha_i y_i$ and $q_i = (ax^T x_i + b)$ and replacing $y = 1$ and $y_d = -1$ in eq. 60, eq. 60 can be rewritten in a matrix form as follows:

$$q^T Qq + pq + f(S) - 4M^T r = 0 \tag{61}$$

where $Q = 16MM^T$, $p = -4\lambda M^T$, $M = [m_1, m_2, \dots, m_n]^T$, $r = [1,1,..,1]^T$ and n is the total number of the training data points. The optimum value of q can be determined by calculating the derivative of eq. 61 with respect to q:

$$\frac{\partial(q^T Qq + pq + f(S) - 4M^T r)}{\partial q} = q^T Q + Qq + p^T$$

$$= 0 \xrightarrow{Q \text{ is a symetric matrix}} 2Qq = -p^T$$

Now the q vector can be written as:

$$q = -\frac{1}{2}Q^{\dagger}p^T \tag{62}$$

where $Q^{\dagger}$ represents pseudo-inverse of Q and q vector can be determined by solving eq. 62. By using $q_i = (x^T x_i + b_2^i)$, r vector is obtained as:

$$\Gamma = \left[b_2{}^1, b_2{}^2, \dots, b_2{}^i, \dots, b_2^l\right] \quad \forall i = 1, \dots, l \tag{63}$$

where l is the total number of q elements and b_2^i is the ith element of Γ and $b_2^i = q_i - x^T x_i$. Finally, the average of all Γ elements is defined as b_1, as follows:

$$b_1 = \frac{1}{l}\sum_{i=1}^{l} b_2{}^i \tag{64}$$

The procedures 1 to 4 will be used to determine a_1 and b_1 for all misclassified training data. Finally, a and b are determined by calculating the average of all resulting a_1s and b_1s, respectively, as follows:

$$a = \frac{1}{m}\sum_{j=1}^{m} a_1 \tag{65}$$

$$b = \frac{1}{m}\sum_{j=1}^{m} b_1 \tag{66}$$

where m is the total number of misclassified training data points. The resulting a and b are used as the 2-degree polynomial kernel parameters for the next iteration.

The above procedure will be continued until the total number of iterations is reached or the validation error does not change for a pre-defined number of iterations.

II. SIMULATION AND EXPERIMENTAL RESULTS

To evaluate the performance of the proposed method, experimental results were generated using ten datasets from UCI machine learning repository [41] called: Letter Recognition (LR) (letters 'A' and 'N' are used for this experiment), Wisconsin Breast Cancer (WBC), Liver Disorder (LD), Heberman, Diabetes, Heart disease dataset, Ionosphere dataset, Parkinson and Sonar dataset. The letter recognition dataset consists of 20000 instances with 17 attributes for each data point (one label and 16 numerical features); Labels consist of 26 English capital alphabets; Wisconsin breast cancer database consists of 699 instances with 11 attributes for each data point, where benign and malignant labels are 2 and 4, respectively; Liver Disorder dataset consists of 345 instances with 7 attributes for each data point; Heberman dataset generated during study on the survival of patients, who had undergone breast cancer surgery; this database consists of 304 instances with 3 attributes for each of its instances. Parkinson dataset is composed of a range of biomedical voice measurements from 31 people, 23 with

Parkinson's Disease (PD). Each column in the table is a particular voice measure, and each row corresponds one of 195 voice recording from these individuals ("name" column). The main aim of the creation of this database was to be used for discriminating healthy people from those with PD. Diabetes database has two classes of data and consists of 804 instances with 8 attributes for each data point. Heart disease database consists of 303 instances with 75 attributes for each data point. Ionosphere database, which is used for binary classification, consists of radar data with 351 instances and 34 attributes for each data point. Sonar database contains 208 instances with 60 attributes for each data point.

TABLE I
DATABASE DESCRIPTION

Database	#Instances	#Dimension
WBC	683	11
Liver disorder	346	7
Heberman	306	3
Diabetes	804	8
Sonar	208	60
Heart	303	75
Ionosphere	351	34
Parkinson	400	22
Letter	1536	17
Iris	150	4

To generate experimental results, all the databases were normalized and then each dataset was randomly divided into three subsets called: train, test, and validation subsets of size 70, 20 and 10 percent, respectively. Training subsets were used for updating kernel parameter, validation subsets were used for terminating the optimization algorithm [20], as mentioned in Section IV and test subset were used for evaluation and comparisons of the performance of the proposed algorithm. The following setting were used to generate results: $f(S) = 50 * \arctan(S/10)$, $\lambda = 0.3$ and regularization parameter, C = 100.1. The resulting number of Support Vectors (SVs) and achieved accuracy for the train and test data of the proposed technique using its RBF kernel parameter optimization algorithm were calculated and compared to those of the anchor SVM and tabulated in Table II. From Table II, the proposed technique generates significantly higher performance in terms of accuracy and the number of SVs than anchor SVM. The proposed method generates significantly lower number of Support Vectors (SVs) in compared to anchor SVM (up to 93.51% reduction), while it gives higher test accuracy. This implies that the proposed method is faster than its anchor SVM in its test phase.

To give the reader a sense of the number of iterations that proposed algorithm needs to determine its optimal kernel parameter, the initial value of γ, the calculated optimal value of γ, number of iterations that algorithm used to determine the optimal value for γ for ten different databases are tabulated in Table III. This table shows that the proposed method arrives at the optimum value of γ using small number of iterations.

The performance of the proposed method using its RBF kernel parameter optimization algorithm were compared to those of Zhang et al.'s [23] and Liu and Xu's [30] methods on five databases (Parkinson, Ionosphere, Sonar, Heberman and Iris databases) are presented in Table IV. From Table IV, it can be seen that the propose method gives either superior or very competitive results to those of Zhang et al.'s and Liu and Xu's methods. The average γ value that used to generate experimental results for the three techniques are also given in Table IV.

The performance of the proposed method using its 2nd-degree polynomial kernel optimization algorithm were also compared to those of Zhang et al.'s [23] and Liu and Xu's [30] techniques on three databases (Iris, Ionosphere, and Heberman databases) are presented in Table V. (In [30], Liu and Xu presented experimental results of the application of a 2nd order polynomial kernel ($(ax^T x_j + b)^2$) for SVM classification, where a and b were set to one, on Iris, Ionosphere and Heberman databases. Therefore, these three databases were used to generate experimental results for the application of the proposed method using its 2nd-degree polynomial kernel optimization algorithm). From Table V, it can be seen that the proposed method outperforms both Zhang et al.'s and Liu and Xu's techniques in terms of accuracy. The average values of a and b, which were used to generate experimental results for the proposed technique are presented in Table V.

III. COUNCLUSIONS

In this paper, a kernel parameter optimization algorithm for support vector machine based on sliding mode control algorithm in a closed-loop manner was presented. The proposed algorithm introduced an error equation and a sliding surface and then iteratively updates the kernel parameter until it reaches maximum number of iterations or the training error stayed unchanged for a predefined number of iterations. Two types of kernels, an RBF or a 2-degree polynomial were considered in this paper. Ten publicly available databases were used to assess and compare the performance of the proposed method with the existing methods. Experimental results show the merit of the proposed method in terms of accuracy, training and testing speed, total number of the support vectors and robustness of the algorithm.

TABLE II
PERFORMANCE OF SVM BASED ON SMC VS. ORIGINAL SVM

Dataset	# Support vectors		Test accuracy		Train accuracy		#Support vectors reduction
	Original SVM	SVM based on SMC	Original SVM	SVM based on SMC	Original SVM	SVM based on SMC	
Liver disorder	171	158	72.46	73.91	98.79	98.79	7.6
Letter	441	286	92.23	99.68	97.17	100	35.14
Wbc	75	41	99.27	99.27	97.35	97.14	45.33
Heberman	183	119	67.74	72.58	88.12	78.53	34.97
Diabetes	552	471	67.53	66.88	64.13	100	14.67
Sonar	149	149	85.71	85.71	75.83	100	0
Heart	217	217	83.60	91.80	80.18	100	0
Ionosphere	251	135	78.87	95.77	85.71	100	53.78
Parkinson	66	58	92.30	94.87	83.57	97.14	12.12
Iris	108	7	96.66	100	98.14	100	93.51

TABLE III
THE OPTIMAL VALUE FOR THE Γ PARAMETER OF RBF KERNEL IN THE FINAL ITERATION FOR EACH DATA SET

Dataset	Initial value of value of γ	Obtained optimal value of γ	#Iteration
Liver disorder	0.5	0.4497829048076992	18
Letter	0.00001	0.19044796894310853	2
Wbc	0.5	0.038161057692307696	1
Heberman	0.5	0.017403710412551898	25
Diabetes	0.00001	16.688341317983557	8
Sonar	0.00001	3.753491664198852	1
Heart	0.00001	0.7599354450040051	8
Ionosphere	0.00001	0.059792421140318186	6
Parkinson	0.00001	0.0006748792225280603	6
Iris	0.00001	0.040178715216006106	5

TABLE IV
EXPERIMENTAL RESULTS FOR THE PROPOSED METHOD USING RBF KERNEL OPTIMIZATION, ZHANG ET AL. [23] AND LIU AND XU'S [30] METHODS

	Zhang et al.		Liu and Xu		Proposed method	
Dataset	Test Accuracy (%)	Optimal γ value	Test Accuracy (%)	Optimal γ value	Test Accuracy (%)	Optimal γ value
Parkinson	93.51	2.59	93.35	3.61	94.87	0.0006748792225280603
Ionosphere	93.80	5.82	95.23	3.99	95.77	0.059792421140318186
Sonar	83.66	18.31	87.31	5.88	85.71	3.753491664198852
Heberman	71.03	1.59	71.37	1.40	72.58	0.017403710412551898
Iris	95.71	1.32	95.84	1.51	100	0.04017871521600610

TABLE V
EXPERIMENTAL RESULTS FOR THE PROPOSED METHOD USING A 2ND-DEGREE POLYNOMIAL KERNEL PARAMETERS OPTIMIZATION, ZHANG ET AL. [23] AND LIU AND XU'S [30] METHODS

	Zhang et al.	Liu and Xu	Proposed method		
Dataset	Test Accuracy (%)	Test Accuracy (%)	Test Accuracy (%)	Optimal value of a	Optimal value of b
Iris	96.31	96.67	100	0.2099999999999993	1.7900000000000007
Ionosphere	92.68	92.74	92.95	0.38999999999999946	1.6100000000000005
Haberman	69.95	71.06	75.80	-985.4958707589385	-0.4921248264528899

REFERENCES

[1] Gou J, Ma H, Ou W, et al. (2019) A generalized mean distance-based k-nearest neighbor classifier. Expert Systems with Applications 115(25): 356–372.
[2] Xia Y, Li CL and Liu N (2017) A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. Expert Systems with Applications 78(1): 225–241.
[3] Shen X and Gu Y (2018) Nonconvex sparse logistic regression with weakly convex regularization. IEEE Transactions on Signal Processing, 66(12): 3199–3211.
[4] Georgevici AI and Terblanche M (2019) Neural networks and deep learning: A brief introduction. Intensive Care Medicine 45(5): 712–714.
[5] Tsouros DC, Smyrlis PN and Tsipouras MG (2018) Random forests with stochastic induction of decision trees. In: IEEE 30th International Conference on Tools with Artificial Intelligence, Volos, 2018, pp. 527–531, Greece. IEEE.
[6] Auria L and Moro RA (2008) Support vector machines (SVM) as a technique for solvency analysis. SSRN Electronic Journal 1(1): 1–18.
[7] Manning CD, Raghavan P and Schutze H (2009) (eds) Support vector machines and machine learning on documents. In: Introduction to Information Retrieval, pp. 293–320, Cambridge University Press.
[8] Demyanov S, Bailey J, Ramamohanarao K and Leckie C (2012) AIC and BIC based approaches for SVM parameter value estimation with RBF kernels. In: Langley P (ed.) Asian Conference on Machine Learning, Journal of machine learning research, JMLR, pp. 97–112.
[9] Famouri M, Taheri M and Azimifar Z (2015) Support vector machines and machine learning on documents. International Journal of Pattern Recognition and Artificial Intelligence 29(8): 155–1013.
[10] Abdiansah A and Wardoyo R (2015) Time complexity analysis of support vector machines (SVM) in LIBSVM. International Journal of Computer Applications 128(3): 0975–8887.
[11] Joachims T (1998) Text categorization with support vector machines: Learning with many relevant features. In: European Conference on Machine Learning pp. 137–142, Chemnitz, Germany. Springer.
[12] Osuna E, Freund R and Girosit F (1997) Training Support Vector Machines: An Application to Face Detection, Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997, pp. 1–8, San Juan, Puerto Rico, USA. IEEE.
[13] Chen ZM, Zhao Q, Wang Z, et al. (2016) Sliding mode control based on SVM for fractional order time-delay system. In: IEEE International Conference on Control and Automation, 2016. IEEE.
[14] Suykens J, Vandewalle J and Moor BD (2001) Optimal control by least squares support vector machines. Neural Networks 14(1): 23–35.
[15] Downs T, Gates KE and Masters A (2001) Exact simplification of support vector solutions. Journal of Machine Learning Research 2(2): 293–297.
[16] Li Y, Zhang W and Lin C (2006) Simplify support vector machines by iterative learning. Neural Information Processing – Letters and Reviews 10(1): 11–17.
[17] Champaigne J (2012) https://www.electronics-inc.com/wpcontent/uploads/BenefitsOfClosedLoop.pdf (accessed 10 August 2019).
[18] Chapelle O, Vapnik V, Bousquet O and Mukherjee S (2002) Choosing multiple parameters for support vector machines. Machine Learning 46(1–3): 131–159.
[19] Chung K, Kao W, Sun T, et al. (2003) Radius margin bounds for support vector machines with the RBF kernel. Neural Computation 15(11): 2643–2681.
[20] Diosan L, Oltean M, Rogozan A and Pecuchet J (2007) Improving SVM performance using a linear combination of kernels. In: International Conference on Adaptive and Natural Computing Algorithms, Berlin, 2007, Springer-Verlag Berlin Heidelberg. Springer.
[21] Keerthi SS (2002) Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms. In: IEEE Transactions on Neural Networks, 13(5): 1225–1229.
[22] Li C, Ho H, Liu Y and Lin C (2012) An automatic method for selecting the parameter of the normalized kernel function to support vector machines. Information Science and Engineering 28(1): 1–15.
[23] Zhang D, Chen S and Zhou Z-H (2006) Learning the kernel parameters in kernel minimum distance classifier. Pattern Recognition 39(1): 133–135.
[24] Imbault F and Lebart K (2004) A stochastic optimization approach for parameter tuning of support vector machines. In: Proceedings of the 17th International Conference on Pattern Recognition Cambridge, UK, pp. 1–4. IEEE.
[25] Liao P, Zhang X and Li K (2016) Parameter optimization for support vector machine based on nested genetic algorithms. Journal of Automation and Control Engineering 4(1): 78–83.
[26] Liu S, Jia C and Ma H (2005) New weighted support vector machine with ga-based parameter selection. In: Proceedings of the 17th International Conference on Pattern Recognition, pp, 4351–4355, Guangzhou, China 2005. IEEE.
[27] Phienthrakul T and Kijsirikul B (2008) Adaptive stabilized multi RBF kernel for support vector regression. In: IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), pp. 3545–3550, Hong Kong, China. IEEE.
[28] Rojas SA and Reyes DF (2005) Adapting multiple kernel parameters for support vector machines using genetic algorithms. In: IEEE Congress on Evolutionary Computation, Edinburgh, 2005, pp. 626–631, Edinburgh, Scotland, UK. IEEE.
[29] Xie T, Yao J and Zhou Z (2019) Da-based parameter optimization of combined kernel support vector machine for cancer diagnosis. Processes 7(5): 263–280.
[30] Liu Z and Xu H (2013) Kernel parameter selection for support vector machine classification. Journal of Algorithms Computational Technology 8(2): 163–177.
[31] Xuefeng L and Fang L (2002) Choosing multiple parameters for SVM based on genetic algorithm. In: 6th International Conference on Signal Processing, pp. 1–34. Beijing, China, 2002. IEEE.
[32] Wahab, M. A., Nefti-Meziani, S., and Atyabi, A. A compre-hensive review of swarm optimization algorithms.PLoSONE, 10(5):e0122827, 2015.
[33] Selvi, V. and R.Umarani. Comparative analysis of ant colonyand particle swarm optimization techniques.Interna-tional Journal of Computer Applications, 5(4):0975 –8887, 2010.
[34] J.-J. E. SLOTINE and W. Li, Applied Nonlinear Control, vol. 199, Massachusetts: Prentice hall Englewood Cliffs, NJ, 1991.
[35] Y. Shtessel, C. Edwards, L. Fridman and A. Levant, "Introduction: Intuitive theory of sliding mode control," in Sliding Mode Control and Observation, -, Springer, 2014, pp. 1-422.
[36] K. Erbatur, M. O. Kaynak and A. Sabanovic, "A study on robustness property of sliding-mode controllers: a novel design and experimental investigations," IEEE Transactions on Industrial Electronics, vol. 46, no. 5, pp. 1012 - 1018, 1999.
[37] L. B. Mohammed and K. Raahemifar, "Improving Support Vector Machine Classification Accuracy Based on Kernel Parameter Optimization," in Proceedings of the Communications and Networking Symposium, Baltimore, 2018.
[38] K. Schittkowski, "Optimal Parameter Selection In Support Vector Machines," journal of industrial and management optimization, vol. 1, no. 4, pp. 465-476, 2005.
[39] Q. Zhang, L. Fang, L. Ma and Y. Zhao, "Research on Parameters Optimization of SVM Based on Improved Fruit Fly Optimization Algorithm," International Journal of Computer Theory and Engineering, vol. 8, no. 6, pp. -, 2016.
[40] B. Gallup, "Sliding Mode Control: A Comparison of Sliding Surface Approach Dynamics," [Online]. Available: http://web.mit.edu/gallup/Public/project.pdf. [Accessed 20 March 2019].
[41] D. Bren, School of Information and Computer Sciences, University of California, Irvine, [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html. [Accessed 12 May 2019].

Maryam Yalsavar received her B.Sc. and M.Sc. (Hons.) degrees in Control Engineering from Shiraz University, Shiraz, Iran. Currently, she is an MSc student at University of Waterloo. She is interested in making connections between different fields of knowledge to improve their performance and reduce their problems specially in fields of machine learning.

DR. Paknoosh Karimaghaei was born in Shiraz, Iran in 1967. He received his B.Sc. (Hons.) in electrical engineering from Shiraz University in 1986. Followed by his M.Sc. (Distinction), and Ph.D. degrees in electronic and electrical engineering, 1995, and 2001, respectively from the Amir Kabir University of Tehran. In 2008, he became an assistant professor in the Shiraz University and then associate Professor in 2014. His research looks at instrumentation, nonlinear control systems, and control theories.

DR. AKBAR SHEIKH-AKBARI is an Associate Professor (Reader) in the School of Built Environment, Engineering and Computing at Leeds Beckett University. He has a BSc (Hons), MSc(distinction) and PhD in Electronic and Electrical Engineering. After completing his PhD at Strathclyde University, he joined Bristol University to work on an EPSRC project in stereo/multi-view video processing. He continued his career in the industry, working on real-time embedded video analytics systems. His main research interests include biometric identification techniques, colour constancy techniques, image source camera identification, hyperspectral image processing, standard and non-standard image/video codecs, e.g., H.264 and HEVC, multiview image/video processing, image resolution enhancement methods, assisted living technologies, compressive sensing, camera tracking using retro-reflective materials, and signal processing.

Dr Mohammad-Hassan Khooban (Senior Member, IEEE) received the Ph.D. degree in power systems and electronics from the Shiraz University of Technology, Shiraz, Iran, in 2017. From 2016 to 2017, he was a Research Assistant with Aalborg University, Aalborg, Denmark, where he conducted research on advanced control of microgrids and marine power systems. From 2017 to 2018, he was a Postdoctoral Associate with Aalborg University. From 2019 to 2020, he was a Postdoctoral Research Assistant with Aarhus University, Aarhus, Denmark, where he is currently an Assistant Professor. He is the Director of the Power Circuits and Systems Laboratory. He has authored or coauthored more than 220 publications on journals and international conferences, three book chapters, and holds one patent. His current research interests include control theory and application, power electronics, and its applications in power systems, industrial electronics, and renewable energy systems.

Prof. Jamshid Dehmeshki received the Ph.D. degree in image processing from the University of Nottingham, Nottingham, U.K., in December 1997. He is a Professor of Medical Image Computing at Kingston University, London, U.K., where he leads the Quantitative Medical Imaging International Institutes. He joined the Nuclear Magnetic Resonance (NMR) Research Unit, Institute of Neurology, University College London, as a Senior Research Fellow and Lecturer in 1999. From 2001 to 2006, as a Chief Technology Officer of the company Medicsight, he established and led research and development on image processing, scientific research, software development, and testing. He was instrumental in the design and development of three families of state-of-the-art medical imaging software products which have received Food and Drug Administration (FDA) approval and European Conformity (CE) marking. He is a holder of 15 international patents and has published more than 150 scientific, technical, and clinical papers on advanced image processing and its applications. His main research interests include developing computer-aided detection/diagnostic and measurement algorithms for quantitative analysis of medical images.

Dr. Salah Al-Majeed is an accomplished academic who has held several key positions at renowned universities with numerous years of experience in; teaching, research, management, and leadership nationally and internationally. Salah has Ph.D. in Electronic Systems Engineering from the University of Essex. He has an extensive portfolio of Industrial and R\&D works, leading the innovation of implementing technologies. His research projects were supported and funded by the UK, EU, and International organisations and companies. Currently, he is leading the IEEE UK and Ireland STEM \& Education Activities, alongside his academic and leadership role as a Deputy Head of School of Computer Science at the University of Lincoln,