Journal of Computer Science

Original Research Paper

Escalating SVM-Efficiency by Sequential QP LP and Slack Variable Analysis

Amit Kumar Kundu, Rezaul Karim and Ali Ahmed Ave

Department of Electrical and Electronic Engineering, Uttara University, Dhaka, Bangladesh

Article history
Received: 16-03-2023
Revised: 29-05-2023
Accepted: 19-06-2023

Corresponding Author:
Rezaul Karim
Department of Electrical and Electronic Engineering, Uttara University, Dhaka, Bangladesh
Email: rezkar@uttarauniversity.edu.bd

Abstract: Support Vector Machine (SVM) is a highly attractive algorithm among many machine learning models due to its generalization power and classification performance based on sound mathematical formulation being convex that offers global minimum. However, despite being sparse, its high classification cost from kernel execution with Support Vectors (SVs) reduces the user's interest when there are hard computational constraints in the application, especially, for large and difficult data. So far in our knowledge, out of many existing works to overcome this problem, some are really interesting and heavy but get less attractive due to improper training difficulties for example, excessive cost-memory requirement, initialization, and parameter selection trouble because of the non-convexity of the problems while the other few that avoid these problems, cannot generate sparsity and complexity simultaneously of the final discriminator upto satisfactory level for very large and tricky data. In this direction, we propose a novel algorithm Efficiency Escalated SVM (EESVM) that solves two convex problems using Quadratic Programming (QP) and Linear Programming (LP) in sequence. This is followed by computational analysis on the remaining smallest set of slack variables that ultimately build two very essential properties of the machine: (i) Highly efficient by being heavily sparse and optimally complex and (ii) Able to handle very large and noise-effected complicated data. Benchmarking shows that this EESVM demands kernel computation as little as 6.8% of the standard QPSVM while posing almost the same classification accuracy on test data and requiring 42.7, 27.7 and 46.6% that of other three implemented state-of-the-art heavy-sparse machines while offering similar classification accuracy. It claims the lowest Machine Accuracy Cost (MAC) value among all of these machines though showing very similar generalization performance that is evaluated numerically using the term Generalization Failure Rate (GFR). Being quite pragmatic for modern technological advancement, it is indispensable for optimum manipulation of the troublesome massive, and difficult data.

Keywords: SVM, Generalization, Sparse, Classification, Non-Convex

Introduction

The size and complexity of data have been intensifying continuously in parallel with the advances in science and technology. Conventional weak classifiers are not fit to classify such complicated data for decision-making. But while the complex non-linear classifiers (for example kernel-based classifiers) are capable of learning these difficult data with optimum generalization and stochastic nature, they demand much computational cost. This increases the uncertainty of successfully performing the quite challenging task of classifying complex data with high speed and accuracy simultaneously.

But while the ultimate goal of the training in a classification learning method is to design the best-generalized mapping from input patterns into their class labels based on the given example patterns and their respective class labels, the size and the selection of kernel operating examples to be applied to form a specific kernel-based stochastic discriminator and preserving its complexity to construct the class-boundaries mainly varies with respect to the size as well as the geometric randomness of the learning data, classification accuracy and essential cost (kernel execution) may vary from one machine to another. In this respect, while the Support Vector Machine (SVM) oriented detectors lead the data

classification area by offering the highest accuracy, an exciting characteristic of it is that because of its sparsity, it is sufficient to use only a subset of the training patterns (known as support vectors) to build the optimum classifier. Thus, the computational load, consequently, the classification cost for non-linear SVM based classifier increases with this number of Support Vectors (SVs) gets higher. Hence, reducing this number of SV is really essential for computational savings and fast classification. Moreover, as the support vectors' number in SVM rises with the increment of the number of training patterns (Steinwart, 2004), this sparsity of support vector set is very significant in the case of large datasets, particularly in the scenario where there exists a hard constraint regarding computational load and time for data classification. Finally, to classify large and complicated datasets efficiently, a highly sparse but high-powered SVM-oriented classifier is strongly demanded. Subsequently, this issue has taken the main focus with a serious concentration in research recently.

Downs *et al*. (2001) propose a method that finds the linearly dependent SVs and throws one but all of them where the number of linearly dependent SVs may go down with the rise of data dimension and complexity. Some clever iterative algorithms are devised by Keerthi *et al.* (2006); Cotter *et al*. (2013); Joachims and Yu (2009) for reduced SVMs that give impressive results requiring only a tiny part of kernel computation for classifying a single pattern where (Cotter *et al*., 2013) claimed a memory expire from (Keerthi *et al*., 2006; Joachims and Yu, 2009) considering their applications on bulky dataset and (Cotter *et al*., 2013) has a noticeable amount of feasible deviation with massive parameter choosing from its real stated approach. Approximating the decision function of a full SVM is proposed by Wu *et al.* (2006); Romdhani *et al.* (2004) by realizing a new compact set of vectors for replacing the support vectors altogether whereas Ratsch *et al.* (2005) explores a wavelet estimation for these latter vectors with the aim of evaluating the dot products amidst pattern vectors efficiently for those the kernels are applied. However, these types of computations of new substitutes of support vectors rely upon complicated optimization methods that are finicky to step sizes, initialization, etc. On the other side, Heisele *et al*. (2003) focused on structured SVM-based classification by using a series of SVMs with linear ones in the beginning and a nonlinear one at the final stage and they tuned the thresholds to optimize classification speed and performance where Karim *et al*. (2007) showed that a well-designed cascade of two non-linear SVM can significantly reduce classification cost compared to a standard SVM. Romdhani *et al*. (2004) offer an SVM chain, which follows an optimization by tuning the threshold and by the approximation of a complete nonlinear SVM that is needed to be determined ahead, while a linear SVM-based decision tree is modeled by Arreola *et al.* (2006). Sahbi and Geman (2006) designed a hierarchical form of SVMs following a tree structure that practices threshold selection and reduced set method in Romdhani *et al*. (2004) for optimization and uses various poses to operate on application followed partitioning of the pattern space. Moreover, in some other interesting works, some SVM-associated cheaper classifiers have been developed, which are different from conventional RSVMs. Maji *et al.* (2013) show that SVMs with histogram and additive kernels work at higher speed and perform better than linear SVM where as Ladicky and Torr (2011) offer a novel SVM classifier that they say locally linear having bounded curvature as well as smooth decision boundary while proposing a trade-off the anchor points' number against the articulateness of the classifier with the intention of avoiding overfitting and run-time problem. To compress the trained SVM further, an additional training algorithm on the obtained SVs using a few more parameters is designed by Xu *et al*. (2015). Li and Zhang (2013) developed a fast object detector that is built by cascading comparatively fewer stages where logistic regression is used in the role of weak learner that focuses on efficient training. Raykar *et al.* (2010) designed a cascade in which classifiers refuse patterns based on probability distributions urged by the classifiers at previous stages to try to find a very good balance between accuracy and feature learning cost. Fu *et al.* (2010) model to combine linear SVMs in order to classify complicated data and claim experimental outcomes showing the prediction phase efficiency of LSVMs comparable to non-linear SVMs. Karim and Kundu (2018) find a very compact and strong classifier with higher classification accuracy by solving QP followed by LP and this machine is referred to as second-order SVM by them. However, shortly after that, Karim and Kundu (2019) offered another algorithm to reduce the number of SVs by using computational analysis after solving an LP and provided experimental results on benchmark data validating that their model is significantly more efficient in the prediction stage while providing a classification accuracy very similar to standard non-linear SVMs. Although these classifiers are very efficient, further sparsification is useful for very large and complex datasets.

In our work, we try to obtain a highly sparse machine with a very straightforward method. By analyzing the learning method of the SVM algorithm and using our computational and statistical perceptions, we propose a new learning algorithm to produce an immensely efficient classifier that accelerates the classification task by demanding significantly less expense for classification without compromising the statistical and generalization guarantees.

The paper is organized using the following sequence: First, we explain our approach in the “method” section, then we describe our experimental works with outputs in the “results and discussion" section, and last, we use the "conclusion” section to conclude our idea and findings with possible future extension.

Method

The proposed training algorithm for extracting the reduced support vectors consists of the following steps: A QP based SVM training, an LP based training followed by the 'slack variable' analysis.

QP-Based SVM Learning on Training Data

Vapnik’s Quadratic Programming based SVM (QPSVM) (Vapnik, 1998) is a widely used supervised classifier, which employs the structural risk minimization concept for building the classifier model. Suppose that for a two-class classification problem, a set of training examples with corresponding labels:

$$D = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\};\ x_i \in \mathbb{R}^d,\ y_i \in \{-1, 1\}$$

are known. Utilizing the given training examples, the goal of this *QP*-based training is to find the decision function having the form class $(x) = sgn(w \cdot \varphi(x) + b)$, where w is the weight vector, b is the bias term and $\varphi(x)$ is the Kernel mapping of the example x from input space to a higher dimensional feature space. To obtain the training model parameters w as well as b, SVM looks for a separating hyperplane that is optimal having the maximum margin between two classes and minimum training error depths of the margin-outward-deviated examples (examples that stay outwards from their own class margins), which leads to the following *QP* problem:

$$\min_{w,b,\xi} f_P(w) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \tag{1}$$

$$s.t.\ \ y_i\left(w \cdot \phi(x_i) + b\right) \geq 1 - \xi_i \tag{2}$$

$$\xi_i \geq 0;\ i = 1, 2, ..., N \tag{3}$$

where, the introduced slack variables $\xi_i > 0$ stand for the training-error depths of margin-outward-deviated examples and the $C > 0$ regularization parameter finds the best balance between minimizing the depth of training-errors and maximizing the margin between two classes optimally. The constraints in relation (2) mean that the intended decision function must be capable of classifying the training examples upto some acceptable depth of training error. The problem in (1-3) can be solved efficiently by using a Lagrangian and in dual form. The equivalent dual problem becomes:

$$\max_{\alpha} f_D(\alpha) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \phi(x_i)\cdot\phi(x_j) + \sum_{i=1}^{N}\alpha_i \tag{4}$$

$$s.t\ \ \sum_{i=1}^{N}\alpha_i y_i = 0 \tag{5}$$

$$0 \leq \alpha_i \leq C;\ \ i = 1, 2, ..., N \tag{6}$$

The dual formulation in (4-6) is also a *QP*. Here, $K(x_i, x_j) = \varphi(x_i)\cdot\varphi(x_j)$ is a kernel operation and the Langrangian multiplier $\alpha_i s$ are the optimization variables. The training model parameters w as well as b can be found as $w = \sum\alpha_i y_i \varphi(x_i)$ and $b = y_m - w\cdot\varphi(x_m)$, where m is an index of an example with $0 < \alpha_m < C$. A *KKT* condition for problems in (1) to (3) is, $\alpha_i(y_i(w.\varphi(x_i) + b) - 1 + \xi_i) = 0$ which implies $y_i(w \cdot \phi(x_i) + b) - 1 + \xi_i = 0$ for Support Vector (SV) patterns with $\alpha_i \neq 0$. The SV-set is extracted with corresponding labels $S_1 = \{(x_i, y_i) \in D | \alpha_i > 0\}$. The SV patterns can be classified into two types, firstly, the patterns lying on their own class margins (having $\alpha_i < C$ and $\xi_i = 0$), patterns staying in the opposite direction of their own class margins (having $\alpha_i = C$ and $\xi_i > 0$). It is notable that the constraint in (5) makes sure that QPSVM has SVs from both classes. A simple graphical representation of *QP* based SVM with decision boundary is shown in Fig. 1.

QPSVM is academically richer having a validated theoretical foundation with skillful dual mapping. The training set is represented by the SVs by filtering out the only significant patterns based on theoretically solid and powerful training using QP formulation. Hence, the only representative SV set S_1 is utilized for further sparsification in the next step.

Training of the Representative Examples Using an LP Based SVM

Sparseness of a machine relies on the number of noisy examples as well as the complication of data. For a set of noisy data, a generalized *QP* based SVM gets more outliers and as a result, these examples are included in SV set S_1 along with the patterns lying on their own class margins. Therefore, while being sufficient to represent the discrimination between classes, the number of SVs heavily increases. Thus, another stage of training on the extracted representative examples can be employed to extract a denser set of representative patterns for

representing the discrimination between two classes without losing the generalization capability. Moreover, after applying *QP* in the first stage, the number of training patterns will be reduced heavily because of the filtration and hence the input load and number of variables will be significantly less for the next stage. Additionally, instead of using an L_2 norm in the cost function, another Kernel based classifier is proposed in Vapnik (1998), which utilizes an indirect L_1 norm in the cost function. This L_1 norm leads to a sparser solution compared to an L_2 norm (Boyd and Vandenberghe, 2004), hence, requiring a lower number of kernel computations in the test phase. We select such one for our next stage by modeling an *LP* as follows. If w_v is the weight vector for this machine, the discriminator function is of the form $f_v(x) = w_v \cdot \varphi(x)+b_v$ that leads to class $(x)= sgn(f_v(x))$. A linear cost function is employed to minimize the summation of the SV coefficients associated with SVs along with the unity deviation penalty.

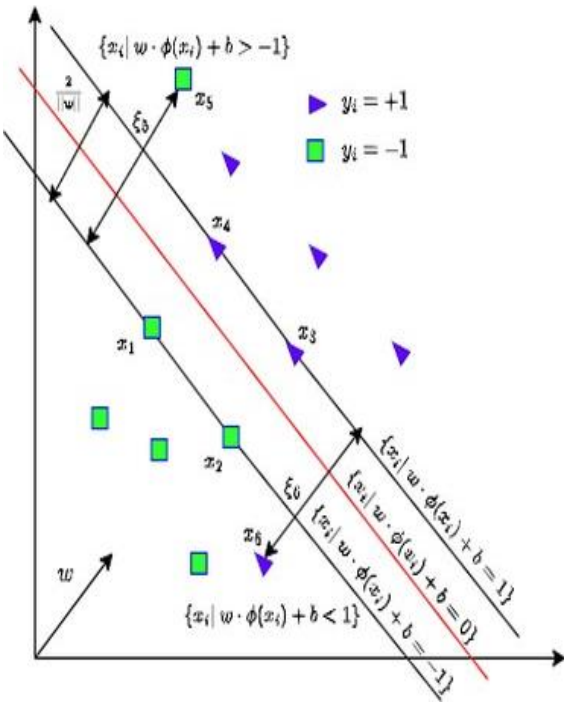


Fig. 1: Sketch of Vapnik's QP SVM (Karim and Kundu, 2019) applying the margin maximization approach: Triangles represent patterns of positive class whereas squares represent patterns of negative class. The dark straight lines represent the edges of the margin from both classes belonging to a hypothetically very higher dimensional feature space. The patterns that stay in the opposite directions of their own class margins are called margin outward deviated patterns and the amount of this deviation is represented by a non-negative variable, ξ. QP SVM maximizes the margin while maintaining the optimal training-error distance. The red line represents QP SVM's decision boundary that is generated by the training output

$$\min_{\lambda,\zeta,b_v} \sum_i \lambda_i + C_v \sum_i \zeta_i \tag{7}$$

$$s.t.\ \ y_i\left(\sum_j \lambda_j y_j \phi(x_j)\cdot\phi(x_i) + b_v\right) \geq 1 - \zeta_i \tag{8}$$

$$\lambda_j \geq 0 \tag{9}$$

$$\zeta_i \geq 0 \tag{10}$$

$$i, j \in \{\mathbb{N} : x_i, x_j \in S_1\} \tag{11}$$

The set of constraints in (8) indicates that the discriminator should classify the representer set of training examples up to an acceptable depth of accuracy error where the variables ζ_i >0 represent the unity outward-deviated examples (examples for which class label $(x_i)\cdot f_v(x_i)$ <1) and a regularizer parameter C_v >0 generates the trade-off between overfitting and learning the data (which is comparatively dense representer of the training data this time). In the case of this machine, the decision function's bias term, b_v is also an optimization variable of the main problem (7-10), which is found with the other variables λ as well as ζ. These optimum values of λ are applied to find the weight vector w_v as $w_v = \sum\lambda_j y_j \varphi(x_j)$. This time, training vectors, x_j having coefficients $\lambda_j > 0$ work as the bases or representer set for the discriminator generated from this problem. For better clarification, we call these patterns "Expansion Vector (EV)" to distinguish them from *SVs* in *QPs* as they work verily in the same way but are produced from different methods. These EVs are generally much less in number relative to the overall size of the training set and at this stage, these vectors with labels are extracted along with the bias b_v to produce an updated discriminator.

Slack Variable Analysis

The pattern set in S_2 provides us with a considerably sparser subset of the training data while representing the discriminator function quite well. However, accelerating the classification process is continuously desired, which motivates us to throw as many patterns as possible from this sparser subset (to further reduce the computational load of the classifier) without notable sacrifice in the classification accuracy. Hence, an approximation of the SVM solution is proposed to further reduce the size of this sparser set S_2, which is achieved by discarding the most insignificant patterns in S_2 following the proposed slack variable analysis based on the corresponding coefficient values as described.

We can rewrite the constraint in (8) as $y_i f_v(x_i) \geq 1 - \zeta_i \Rightarrow \zeta_i \geq 1 - y_i f_v(x_i)$. For a misclassified pattern (pattern staying in the region of the opposite class)

$x_i y_i f_v(x_i) < 0 \Rightarrow \zeta_i > 1$. Hence, for L misclassified patterns, $\sum_{i=1}^{L} \zeta_i > L$. Thus suppressing $\sum_i \zeta_i$ by penalizing through regularize will suppress L, the number of training errors. Therefore, throwing outliers from the S_2 set apparently reduces training errors. Moreover, by observing the cost function in the above *LP*, we find that patterns having higher ζ values get comparatively very small (but non-zero) coefficient values λ that leads these patterns to be members of the final discriminator and demand kernel computations during the classification process though create more ambiguity than precision for correct decision by staying in another class. Thus, considering the computational load and classification accuracy, we analyze their contribution to correct decision-making based on misleading and right-leading them.

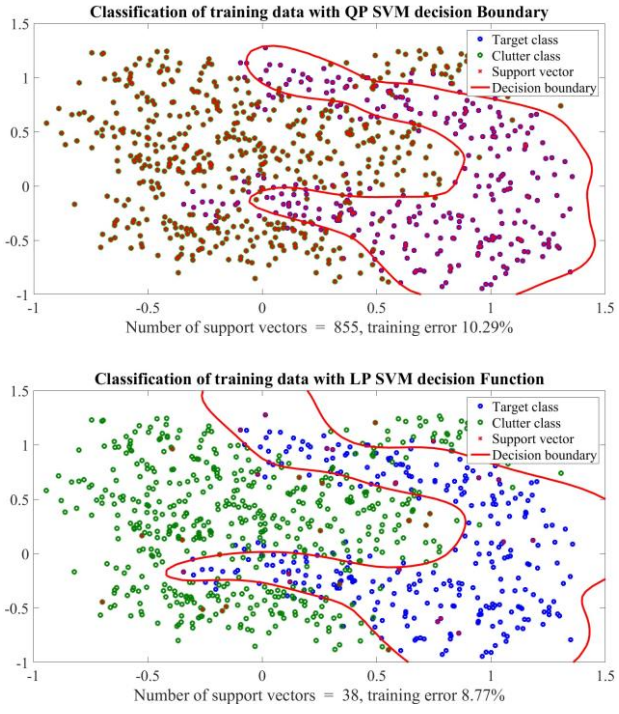
Right-Leading and Misleading in the Discriminator from the Patterns Having $\zeta > 1$

After the sequential *QP* and *LP*, the decision function is modeled in a way that for any arbitrary training pattern x_a, having label y_a, the function value is. $f(x_a) = \sum_{l \in S_2} \lambda_l y_l K(x_l, x_a) + b_v$ Considering $m, n : m \cup n = \{l\}, m \cap n = \{\}, y_m = -y_n$ we get x_m, x_n are all Kernel Computing Vectors (KCVs). Thus as λ_m, $\lambda_n > 0$, and $K(x_m, x_a), K(x_n, x_a) > 0$ if $\frac{\lambda_m y_m K(x_m, x_a)}{y_a} < 0 \implies \frac{y_m}{y_a} < 0 \implies x_a \ \& \ x_m$ are from opposite classes while x_m pushes x_a outwards from x_a*'s* own class, which we call here as wrong force, and in case of $\frac{\lambda_n y_n K(x_n, x_a)}{y_a} > 0 \implies \frac{y_n}{y_a} > 0 \implies x_a \ \& \ x_n$ are from the same class while x_n pulls x_a inwards to x_a*'s* class, which we call here as right force. Now, from these *KCVs*, those who have $\zeta > 1$ stay inside the boundaries of opposite classes, which usually provide lower values for the Euclidean distances between these patterns and the patterns of opposite classes and higher values for the Euclidean distances between these patterns and the patterns of their own classes. As we have used RBF kernel, which is inversely proportional to distance, using these patterns as the bases in the discriminator produces comparatively larger kernel values with respect to the opposite class that boosts the wrong force, which powers Misleading (pushing away from its own class) and smaller kernel values for patterns of the same class that weakens the right force, which weakly supports Right-leading (pulling a training pattern inwards to the pattern's class) for correct decision making. Thus, considering higher classification accuracy and speed, we throw KCVs with $\zeta > 1$ patterns from the S_2 set as part of the active KCVs for our final discriminator after examining their depth of misleading and right-leading with respect to all of the training examples by applying the following analytic survey.

For a $\zeta > 1$ *KCV*, x_O having label y_O, Risght Leading Depth (RLD) on all of the training examples, $RLD(x_O) = \frac{1}{Card\{i\}} \sum_{\{i: y_i \cdot y_O = 1\}} \lambda_{x_O} K(x_O, x_i)$ and hence right-leading of x_O on all of the training examples, $RightLead(x_O) = \frac{1}{Card\{i\}} \sum_{\{i: y_i \cdot y_O = 1\}} K(x_O, x_i)$ whereas MLD (Misleading Depth) of that KCV, x_O on all of the training examples can be found as, $MLD(x_O) = \frac{1}{Card\{i\}} \sum_{\{i: y_i \cdot y_O = -1\}} \lambda_{x_O} K(x_O, x_i)$ and thus Misleading of x_O on all of the training examples, $MisLead(x_O) = \frac{1}{Card\{i\}} \sum_{\{i: y_i \cdot y_O = -1\}} K(x_O, x_i)$.

Exploring the benchmark datasets shows that for almost all cases, the $\zeta > 1$ KCVs Misleads more than Right-leading, which aligns with our proposal of throwing $\zeta > 1$ KCVs from the predictor. The algorithm of our proposed scheme is shown in Algorithm 1.

Figure 2 shows the decision boundary evolution with a number of support vectors and training error rates after the three stages of the proposed algorithm, such as *QP* training, *LP* training, and the slack variable analysis on synthetic data. Fig. 2(a) presents the decision boundary after *QP* SVM training with C = 0.125 and σ = 0.125, Fig. 2(b) presents the decision boundary after LP training on S_1 set with C_{LP} = 16 and σ_{LP} = 0.25 and the decision boundary after the final stage is shown in Fig. 2(c). The escalated efficiency of our method is quite clear here as it demands a number of KCVs only around 4% of standard *QP* SVM while posing similar classification accuracy.



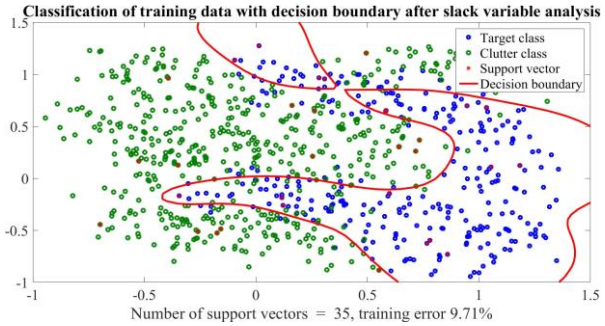


Fig. 2: Illustration of decision evolution of the proposed EESVM

Results and Discussion

This part presents the experimental results to validate our proposed method EESVM along with data description and discussions about the results obtained from experiments. Seven benchmark machine learning datasets available in Diethe (2015) are used for evaluating the performance. The datasets are Banana, German, Heart, Ringworm, Titanic, Twonorm, and Waveform as listed in Tables 1-3. The EESVM is compared with the QPSVM, LP SVM (VLPSVM), SOSVM proposed in Karim and Kundu (2018) and RLPSVM proposed in Karim and Kundu (2019). In all experiments, the Gaussian kernel is implemented. In QPSVM, VLPSVM, and RLPSVM, the corresponding penalty term parameters C and C_V and the kernel computing parameters σ, σ_V are chosen based on the highest cross-validation true prediction rate using 5-fold cross-validation. In implementing these 3 machines, cross-validations are performed with kernel parameters σ and σ_V. To obtain the best parameters, modified fivefold cross-validation is implemented where we randomly choose 4 folds of training examples to train the first stage with specific C and σ. The returned KCVs are used as training sets with the best C_V and σ_V in the later stage. The final KCVs are used to build the predictor and test the remaining training data. C, σ, C_V, and σ_V are chosen from the ranges of $\{2^{-2},2^0,2^2,...,2^{12}\}$,$\{2^{-2},2^0,...,2^6\}$, $C \times \{2^{-2},2^{-1},2^0,...,2^5\}$ and $\sigma \times \{2^{-2},2^{-1},2^0,...,2^5\}$ respectively.

Numbers of KCVs and Test Error Rates of Proposed Method (EESVM) and Different Machines are given in Table 1.

Algorithm 1: Proposed EESVM (Efficiency Escalated SVM)

1: **Input:** A training set $D = \{(x_i, y_i)\}_{i=1}^{N}$
2: **Output:** A discriminator $f_{rom.}(\cdot)$
3: Select the best penalty parameters and Kernel parameters: $\equiv (C,\sigma),(C_V,\sigma_V)$
4: Run the QP-based SVM using the entire training set to solve the following problem:

$$\min_{\alpha} f(\alpha) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \phi(x_i)\cdot\phi(x_j) - \sum_{i=1}^{N}\alpha_i$$

$$s.t \ \sum_{i=1}^{N}\alpha_i y_i = 0$$

$$0 \le \alpha_i \le C; \ \ i = 1,2,...,N$$

5: Extract the *SV*-set of this step with labels as $S^1 = \{(x_l, y_l) \in D | \alpha_l > 0\}$
6: Run LP-based SVM on the extracted SV-set S_1 by solving the following problem:

$$\min_{\lambda,\zeta,b_V} \ \sum_{l\in S_1}\lambda_l + C_v\sum_{l\in S_1}\zeta_l$$

$$s.t. \ \ y_i\Big(\sum_{l\in S_1}\lambda_l y_l \phi(x_l)\cdot\phi(x_i) + b_v\Big) \ \ge \ 1-\zeta_i$$

$$\lambda_l \ \ge \ 0; \ \forall l \in S_1$$

$$\zeta_i \ \ge \ 0; \ \forall i \in S_1$$

7: Extract SVs of this LP step with labels as $S^2 = \{(x_p, y_p) \in S_1 | \lambda_p > 0\}$ and the bias b_v.
8: Find $\{(x_p, y_p) \in S_2 | \zeta_p > 1\}$ and throw the corresponding patterns from S_2 to construct a new compact SV set, $C \in \{2^{-2},2^0,2^2,...,2^{12}\}$ and $\sigma \in \{2^{-2},2^0,2^2,...,2^6\}$. For SOSVM and our proposed method EESVM, there are two penalty parameters in the two stages C and C_V, and two $S_3 = \{(x_p,y_p) \in S_2 | \zeta_p \le 1\}$
9: $w_{Prop.} \leftarrow \sum_{p\in S_3}\lambda_p y_p \phi(x_p)$; $b_{Prop.} \leftarrow b_v$
10: Return $f_{Prop.}(\cdot) = w_{Prop.}\cdot\phi(\cdot) + b_{Prop}$

Goodness of Machines

To measure the goodness of a classifier, two terms, namely Generalization Failure Rate (GFR) and Machine Accuracy Cost (MAC) (Karim and Kundu, 2018) of different machines are analyzed. GFR represents the classifier's deficiency in generalizing the test data, whereas MAC expresses machines' cost per accuracy.

Generalization Failure Rate (GFR)

The primary goal of a training algorithm of a classifier is to obtain a discriminator using the training examples that classifies the novel examples with the highest possible accuracy. The accuracy of test data provides us the information about the classifier's performance but does not tell us about the bridging capability of a machine between the training and test data. Moreover, a classifier should be as sparse as possible for real-time implementation, which means a kernel-based classifier should demand as few kernel evaluations as possible for

classifying a test example. Therefore, the set of KCVs heavily influences the classifier's performance. If the KCV set leads to a relatively simple model, it may fail to properly learn the complexity of the data and thus perform poorly on both the training and test examples from under-fitting. On the other side, if the KCV set leads to a very complicated model, the model learns the unnecessary details and noise from the training examples. Thus, the training error decreases by weakening the general model, which in turn, leads to generalization failure with an increase in the test-error rate, also known as over-fitting. Hence, to evaluate the deficiency of a classifier to generalize, the terms Over-fitting Tendency (OT) and GFR are introduced as follows:

$$OT = \frac{Test\ Error\ Rate - Train\ Error\ Rate}{Train\ Error\ Rate} \quad (12)$$

$$GFR = \frac{OT}{Test\ \ Accuracy} \quad (13)$$

OT increases for a lower value of training error rate and a higher test error rate. *GFR* is defined keeping two points in mind: (i) How much the classifier overfits and (ii) How badly it performs on the test patterns. *GFR* increases with an increase in *OT* or with a decrease in test accuracy. The *GFR* of different machines are given in Table 2.

Machine Accuracy Cost (MAC)

A high-performance machine with less computational cost is always highly desired. In this regard, the objective is to build a predictor demanding a small number of kernel evaluations in the test phase having high accuracy. Hence, to terminate the confusion between the usefulness of an expensive machine with higher accuracy and a cheaper machine with comparable accuracy, the term MAC is defined for kernel-based machines as:

$$MAC = \frac{Number\ \ of\ \ KCVs}{Test\ \ Accuracy} \quad (14)$$

A low MAC value of a machine is always desired, which implies the machine will achieve maximum test accuracy with minimum KCVs. MAC values from different machines are given in Table 3.

Table 1: Numbers of KCVs and Test Error Rates of Proposed Method (EESVM) and different machines

Dataset name (No. of training patterns, No. of testing patterns, dimension)	QPSVM mean (KCVs, TeER)	VLPSVM (Karim and Kundu, 2018) mean (KCVs, TeER)	SOSVM (Karim and Kundu, 2018) mean (KCVs, TeER)	RLPSVM (Karim and Kundu, 2019) mean (KCVs, TeER)	Proposed method EESVM mean (KCVs, TeER)
Banana (400, 4900, 2)	(102.26, 10.61)	(15.08, 10.75)	(15.23, 10.91)	(13.47, 11.38)	(13.62, 11.53)
German (700, 300, 20)	(438.75, 23.70)	(26.47, 24.00)	(178.99, 25.27)	(38.74, 24.30)	(27.61, 24.16)
Heart (170, 100, 13)	(68.23, 16.60)	(21.94, 17.44)	(10.60, 15.55)	(8.07, 15.51)	(7.45, 15.33)
Ringnorm (400, 7000, 20)	(77.00, 2.23)	(15.99, 1.73)	(40.26, 1.81)	(15.99, 1.73)	(15.83, 2.54)
Titanic (150, 2051, 3)	(148.50, 22.69)	(83.91, 22.91)	(48.48, 23.34)	(82.22, 22.91)	(39.19, 23.18)
Twonorm (400, 7000, 20)	(299.18, 2.42)	(32.00, 3.71)	(18.50, 3.38)	(19.27, 3.70)	(19.45, 2.93)
Waveform (400, 4600, 21)	(228.33, 13.15)	(20.81, 11.18)	(21.06, 11.20)	(20.64, 10.44)	(10.67, 12.72)
Average k (–, –, –)	(194.61, 13.06)	(30.89, 13.10)	(47.59, 13.07)	(28.34, 12.85)	(19.11, 13.20)

In Table 1, the number of Kernel Computing Vector (KCVs) of different state-of-the-art machines along the proposed EESVM with the test error rates of different machines on benchmark datasets (Diethe, 2015) is presented. It is observed from the table that for all seven datasets, EESVM requires significantly fewer kernel evaluations compared to the sparse QPSVM. EESVM requires as little as 4.7% of QPSVM for the Waveform dataset and 9.8% averaged over all datasets. Despite huge cost reduction, EESVM performs similarly to the powerful QPSVM posing amazingly 0 .43% more for the Waveform dataset and only 0.14% less on average. Moreover, we observe that the proposed EESVM requires significantly less number of kernel evaluations on average compared to the other three sparser machines. EESVM requires as little as 61.9% kernel computations compared to the VLPSVM, 40.2% kernel executions of SOSVM, and 67.4% kernel executions of RLPSVM while offering very similar classification accuracy

Table 2: Generalization failure rate of different Machines

Dataset	QPSVM	VLPSVM (Karim and Kundu, 2018)	SOSVM (Karim and Kundu, 2018)	RLPSVM (Karim and Kundu, 2019)	EESVM
Banana	0.0031	0.0026	0.0019	0.0016	0.0018
German	0.0049	0.0011	0.0140	0.0015	0.0009
Heart	0.0039	0.0060	0.0012	0.0012	0.0010
Ringnorm	–	0.0103	–	0.0103	0.0162
Titanic	0.0017	0.0015	0.0010	0.0015	0.0010
Twonorm	0.0017	1.7005	0.0246	0.0539	0.0070
Waveform	0.0013	0.0051	0.0051	0.0042	0.0018
Average	0.0028	0.2467	0.0080	0.0106	0.0042

In Table 2, the GFR values of different machines along the proposed classifier are presented. It is observed from the table that in the case of German, Heart, and Titanic datasets, the GFR values of the proposed classifier are less than the corresponding GFR values from the other four classifiers whereas, for the other four datasets, the GFR values of the proposed machines are better in most cases while considering the average value, it is the second best following very closely the most powerful standard QP SVM

Table 3: Machine accuracy cost of different machines

Dataset	QPSVM	VLPSVM (Karim and Kundu, 2018)	SOSVM (Karim and Kundu, 2018)	RLPSVM (Karim and Kundu, 2019)	EESVM
Banana	1.1439	0.1690	0.1710	0.1520	0.1539
German	5.7501	0.3483	2.3952	0.5118	0.3641
Heart	0.8181	0.2657	0.1255	0.0955	0.0880
Ringnorm	0.7876	0.1627	0.4100	0.1627	0.1624
Titanic	1.9208	1.0885	0.6324	1.0666	0.5102
Twonorm	3.0661	0.3323	0.1915	0.2001	0.2004
Waveform	2.6291	0.2343	0.2372	0.2305	0.1222
Average	2.3022	0.3715	0.5947	0.3456	0.2287

In Table 3, the MAC values of different machines along the proposed classifier are presented. It is observed from the table that in the case of Heart, Titanic, and Waveform datasets, the MAC values of the proposed classifier are less than the MAC values of the corresponding other four classifiers, which implies that the proposed machine involves less cost per accuracy than other reported classifiers whereas, for the other four datasets, the MAC values of the proposed machines are better in most cases while considering the average values, it is the best

Conclusion

In this research paper, we propose an SVM-based novel algorithm, EESVM to classify data with a highly powerful discriminator that demands the least computational load. It is built by imitating the Powerful and standard (QP based) Support Vector Machine maintaining optimal complexity and accuracy through filtering its Support Vectors to select their possibly smallest but dense subset that is almost a complete perceptual representer of the data set. This is achieved by modeling objective functions involving margin maximization and direct sparsification that are activated in sequence following the pattern scattering and their topological representation in such a way that different optimizers work on the pattern space using the generalized representer of the training set according to the pattern-spanning in the individual level being interfaced through the constraints along with the objective functions. This solution is further improved by the virtue of some computational analysis and statistical insight on this resultant discriminator involving the slack variables (ζ) that correlate with noise and error to reduce both classification cost and possible overfitting by throwing those kernel computing vectors that require computational cost but may not have countable influence in higher classification accuracy. An important feature of our algorithm is that it is straightforward to build and simple to use having just one stage in the classification phase though there are three stages in training. The key achievement here is that it can learn only a fractional number of Kernel Computing Vectors (KCVs) compared to the standard SVM to form a classifier with high generalization capability producing classification accuracy quite similar to this powerful SVM. It gives the lowest Machine Accuracy Cost (MAC) value. Despite being so sparse, its generalization ability is better than other machines and also very near to QPSVM while for a few datasets, it generalizes even better than that. Thus, our method, based on combined operations of optimizations and computational analysis, is also able to successfully fill up the very little gap that remains from a powerful and parameter-based machine like SVM where there is no ultimate certainty to obtain the best regularization parameters using cross-validation, which may bring a sharp change in the solution by involving a much higher number of KCVs for very similar accuracy from the computationally costly classifier with a significant probability of over-fitting that could even be further triggered by the inclusion of noise effected highly deviated outliers in training set. A classifier with such efficiency, accuracy, and simplicity with noise de-coupling ability is indispensable for real-time classification of the heavy challenging very large, and noise-effected complicated data. In the future, introducing asymmetry and probabilistic bounds on the boundary values of the error rate using an analytic method would be interesting.

Acknowledgment

Thanks to the journal authority for maintaining such a useful platform to accelerate the advances of modern science and give us a chance to work for sharing our findings and hence contributing to the research world.

Funding Information

For this study, the authors are not financially supported by any source.

Author's Contributions

Amit Kumar Kundu: Conception and design, data analysis, interpretation, application, drafted the article, reviewed it critically, final approval of the version to be submitted with the revised version.

Rezaul Karim: Conception and design, data analysis, interpretation, drafted the article, reviewed it critically and updated of the version to be submitted with the revised version.

Ali Ahmed Ave: Data handling, drafted the article, reviewed it and updated of the version to be submitted.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and that no ethical issues are involved.

References

Arreola, K. Z., Fehr, J., & Burkhardt, H. (2006). Fast support vector machine classification using linear SVMs. In *Proc. of the 18th International Conference on Pattern Recognition* (ICPR) (Vol. *3*, pp. 366-369). https://doi.org/10.1109/ICPR.2006.549

Boyd, S., & Vandenberghe, L. (2004). Convex optimization. Cambridge University Press. https://doi.org/10.1017/CBO9780511804441

Cotter, A., Shalev-Shwartz, S., & Srebro, N. (2013). Learning optimally sparse support vector machines. In *Proc. of the 30th International Conference on Machine Learning* (ICML) (pp. 266-274). https://doi.org/10.1109/CVPR.2013.445

Diethe, T. (2015). 13 benchmark datasets derived from the UCI, Delve and Stat log repositories. Retrieved from https://github.com/tdiethe/gunnarraetsch benchmark datasets/

Downs, T., Gates, K. E., & Masters, A. (2001). Exact simplification of support vector solutions. *Journal of Machine Learning Research*, *2*(Dec), 293-297. https://www.researchgate.net/publication/220321087

Fu, Z., Kelly, A. R., & Zhou, J. (2010). Mixing linear svms for nonlinear classification. *IEEE Transactions on Neural Networks*, *21*(12), 1963-1975. https://doi.org/10.1109/TNN.2010.2080319

Heisele, B., Serre, T., Prentice, S., & Poggio, T. (2003). Hierarchical classification and feature reduction for fast face detection with support vector machines. *Pattern Recognition*, *36*(9), 2007-2017. https://doi.org/10.1016/S0031-3203(03)00062-1

Joachims, T., & Yu, C. N. J. (2009). Sparse kernel SVMS via cutting-plane training. *In Proc. European Conf. on Machine Learning and Knowledge Discovery in Databases* (ECML) (p. 8). https://doi.org/10.1007/s10994-009-5126-6

Karim, R., & Kundu, A. K. (2018). Efficiency and performance analysis of a sparse and powerful second order SVM based on LP and QP. *International Journal of Advanced Computer Science and Applications*, *9*(2), 311-318. https://doi.org/10.14569/ijacsa.2018.090244

Karim, R., & Kundu, A. K. (2019). Computational analysis to reduce classification cost keeping high accuracy of the sparser lpsvm. *International Journal of Machine Learning and Computing*, *9*(6), 728-733. https://doi.org/10.18178/ijmlc.2019.9.6.865

Karim, R., Bergtholdt, M., Kappes, J., & Schnörr, C. (2007, September). Greedy-based design of sparse two-stage SVMs for fast classification. In *Joint Pattern Recognition Symposium* (pp. 395-404). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-74936-3_40

Keerthi, S. S., Chapelle, O., & DeCoste, D. (2006). Building support vector machines with reduced classifier complexity. *Journal of Machine Learning Research*, *7*(Jul), 1493-1515. https://doi.org/10.5555/1248547.1248602

Ladicky, L., & Torr, P. H. S. (2011). Locally linear support vector machines. In *Proc. of the 28th International Conference on Machine Learning* (ICML) (pp. 985-992). https://www.researchgate.net/publication/221345963

Li, J., & Zhang, Y. (2013). Learning SURF cascade for fast and accurate object detection. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR) (pp. 3468-3475). https://doi.org/10.1109/CVPR.2013.445

Maji, S., Berg, A. C., & Malik, J. (2013). Efficient classification for additive kernel SVMS. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1), 66-77. https://doi.org/10.1109/TPAMI.2012.62

Ratsch, M. Romdhani, S., Teschke, G., & Vetter, T. (2005). Over-complete wavelet approximation of a support vector machine for efficient classification. *In Proc. of the Joint Pattern Recognition Symposium* (pp. 351-360). https://www.researchgate.net/publication/221114876

Raykar, V. C., Krishnapuram, B., & Yu, S. (2010). Designing efficient cascaded classifiers: Tradeoff between accuracy and cost. In *Proc. of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 853-860). https://doi.org/10.1145/1835804.1835912

Romdhani, S., Torr, P., Scholkopf, B., & Blake, A.¨ (2004). Efficient face detection by a cascaded support-vector machine expansion. In Proc. of the Royal Society of London A: *Mathematical, Physical and Engineering Sciences*, *460*(2051), 3283-3297. https://doi.org/10.1098/rspa.2004.1333

Sahbi, H., & Geman, D. (2006). A hierarchy of support vector machines for pattern detection. *Journal of Machine Learning Research*, *7*(Oct), 2087-2123. http://jmlr.org/papers/v7/sahbi06a.html

Steinwart, I. (2004). Sparseness of support vector machines some asymptotically sharp bounds. *In Advances in Neural Information Processing Systems.* (pp. 1069-1076). https://www.researchgate.net/publication/2602701

Vapnik, V. (1998). Statistical learning theory. Wiley. https://www.scribd.com/doc/191179292/

Wu, M., Scholkopf, B., & Bakır, G. (2006). A direct method¨ for building sparse kernel learning algorithms. *Journal of Machine Learning Research*, *7*(Apr), 603-624. https://www.researchgate.net/publication/220320292

Xu, Z. E., Gardner, J. R., Tyree, S., & Weinberger, K. Q. (2015). Compressed support vector machines. *arXiv preprint arXiv*:1501.06478. https://doi.org/10.48550/arXiv:1501.06478