Computational Analysis to Reduce Classification Cost Keeping High Accuracy of the Sparser LPSVM

Rezaul Karim and Amit Kumar Kundu

***Abstract*—Vapnik's quadratic programming (QP)-based support vector machine (SVM) is a state-of-the-art powerful classifier with high accuracy while being sparse. However, moving one step further in the direction of sparsity, Vapnik proposed another SVM using linear programming (LP) for the cost function. This machine, compared with the complex QP-based one, is sparser but poses similar accuracy, which is essential to work on any large dataset. Nevertheless, further sparsity is optimum for computational savings as well as to work with very large and complicated datasets. In this paper, we accelerate the classification speed of Vapnik's sparser LPSVM maintaining optimal complexity and accuracy by applying computational techniques generated from the "unity outward deviation (ζ)"—analysis of kernel computing vectors. Benchmarking shows that the proposed method reduces up to 63% of the classification cost by Vapnik's sparser LPSVM. In spite of this massive classification cost reduction, the proposed algorithm poses classification accuracy quite similar to the state-of-the-art and most powerful machines, for example, Vapnik's QPSVM or Vapnik's LPSVM while being very simple to realize and applicable to any large dataset with high complexity.**

***Index Terms*—Classifier, kernel, LP, QP, sparse, SVM.**

I. Introduction

The main task in a classification problem is to model the best function representing a generalized relation between input patterns and their class labels using the input example patterns with their corresponding class labels. Although the number of kernel operating patterns and their selection to be used to construct a particular kernel-based stochastic classifier and maintain its complexity to design the decision boundary primarily depends on the size and topological randomness of the involved data, classification performance and required cost (kernel computation) may differ from machine to machine. In this regard, while support vector machine(SVM)–based classifiers lead the pattern classification field with top accuracy, one of its key interesting properties is that being sparse, the desired classifier can be designed in terms of only a subset of the training patterns, known as the support vectors (SVs). Hence, computational complexity and classification time for the nonlinear SVM classifier depend on the number of SVs. But although being sparser, Vapnik's linear programming (LP)-based SVM is computationally less costly by one order of magnitude, it still requires kernel evaluations, which is a very expensive task. This stimulates us to reduce further computational cost for classification without significant loss of performance. As a result, this problem has increased research recently.

A. Related Works

Ref. [1] proposed a very sparse and powerful classifier with high accuracy by implementing sequential optimization using quadratic programming (QP) and LP and named this detector as second-order SVM. Although the classifier from their straight forward method is very efficient, effort is needed during training by implementing modified cross-validation to obtain optimum values of two pairs of parameters for its two sub-stages. [2]-[4] proposed smart iterative algorithms for reduced SVMs with impressive results. However, while [3] reported a memory run out from [2], [4] in case of their implementations on large dataset, [3] has a considerable practical variation with heavy parameter selection from its defined approach. [5]-[8] modeled reduced SVMs (RSVMs) that demand less computational costs than the standard one for classifying a pattern. These RSVMs demand only a fraction of kernel operation to classify a pattern either by exploring a sparse subset of the whole SVs of the standard SVM [5] or by finding a novel small set of patterns to replace the whole SVs set [7], [8], which depends on complex optimization problems that are critical to initialization, step sizes, etc. Moreover, extra training effort is required to train the full SVM before approximation. [9] offered a locally linear SVM classifier while proposing a trade-off between the number of anchor points and the expressivity of the classifier to avoid over-fitting and speed problem. [10] designed a post-processing algorithm that compresses the learned SVM by further training on the SVs by adding few extra training parameters. [11] analyzed a problem of combining linear SVMs (LSVMs) to classify nonlinear data. [12] modeled a clustered SVM by weighted combination of LSVMs trained on the clustered subsets of the training set to separate the data locally. [13] proposed a method that finds SV patterns that are linearly dependent on feature space and throw all but one of them. However, it gets more difficult to get such linearly dependent patterns as the dimensions of data increase. On the other hand, another class of approach focuses on structured SVM-based pattern detection. A decision tree with linear SVMs is proposed in [14], while a hierarchy of linear SVMs is studied in [15] having a nonlinear SVM at the end. In [7], a chain of SVMs is optimized by parameter tuning. [16] used two reduced SVMs (RSVMs) in sequence with increasing complexity that offers heavy computational cost reduction compared with a

Manuscript received May 20, 2019; received October 24, 2019.

The authors are with the Department of Electrical and Electronic Engineering, Uttara University, Bangladesh (e-mail: {rezkar, amit31416}@uttarauniversity.edu.bd)

doi: 10.18178/ijmlc.2019.9.6.865

full SVM with very similar accuracy; they also handled asymmetric data successfully. In [17], a hierarchical tree structure using SVMs is developed that is optimized by using a reduction scheme proposed in [7] and threshold selection and operates on application specific pattern-space partitioning. [18] analyzed numerical strategies for optimal cascade and implemented different heuristics on synthetic data using binary SVM at each stage.

B. Motivation

The standard SVM algorithm does not focus on the error rate of the discriminator directly; rather, it focuses on error performance indirectly through cost-function penalty and constraints. Therefore, the penalty parameter plays a very crucial role in the algorithm, especially in finding the best trade-off between generalization and over-fitting. However, as there is no exact closed-form expression for this data-dependent regularizer, one has to find it along with the other data-dependent kernel parameter using cross-validation method, which may not give the best values of parameters. Hence, a bit computational manipulation afterward on the first-step solution might be useful. On top of that, it is also observed that to increase a very tiny amount of accuracy on complex dataset, the SVM increases comparatively a large number of SVs (clarified in Fig. 1). Thus, saving a large amount of computation (hence, increasing classification speed by discarding the least significant SVs and thus reducing discriminator complexity) by compromising (if it is needed, in the worst case) with a very tiny amount of accuracy might be a good trade-off. Moreover, accelerating the classification process is continuously desired that stipulates us to work on discarding as many SVs as possible. However, discarding any SV without considering its contribution will affect seriously in the decision making process. On the other hand, in many practical applications, some obtained training data are often affected by noise that is placed far away from the related main cluster (termed as "outliers"), whereas the class-decision boundary from SVM depends on the SVs. But often for the complex datasets, the SVs set contain outliers that may increase the non-smoothness of the decision boundary. Hence, a scheme that reduces the number of SVs after considering the outlier contribution or infliction on the decision function is advantageous in the following directions:

- Reducing the classification cost by requiring less kernel evaluation
- Smoothing the decision boundary that helps to reduce over-fitting

In this direction, we model the approximation of an SVM solution for classification problem by reducing the SV set size. The reduction is achieved by discarding as many insignificant SV patterns as possible based on the analysis from the corresponding coefficient values. Experimental results show that the proposed algorithm reduces the classification cost significantly and poses classification accuracy similar to the standard SVM.

The rest of the paper is organized as follows: Section II presents the related bases, whereas Section III explains the proposed algorithm with proper analysis. In Section IV, the experiment results are shown, and concluding discussion with future work is presented in Section V.

II. Related Basics

SVM is a state-of-the-art machine learning tool for supervised classification. Being very powerful, hence least erroneous in spite of sparsity, it has become popular. At first, it was introduced using QP, termed here as QPSVM. However, while L1 norm is usually more intending to sparser solution compared with L2 norm [19], Vapnik proposed another SVM using LP in the cost function, termed here as VLPSVM. In this section, discussion on SVM with its two main variants (QPSVM and VLPSVM) are presented. Details of them are found in [20].

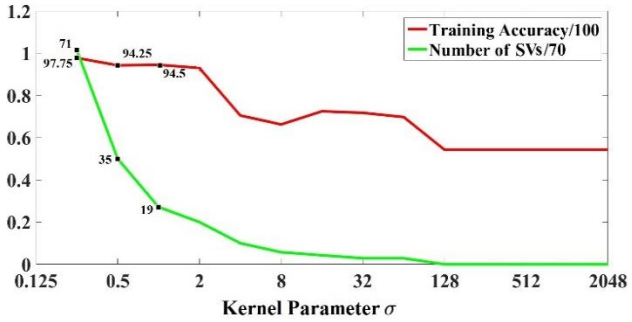


Fig. 1. Fluctuation on the number of SVs with training accuracy. The plots are divided by their corresponding maximum values to scale them between 0 and 1. The fluctuation is well observed from variation in Kernel parameter, σ.

A. QPSVM

QPSVM is based on structural risk minimization principle in statistical learning theory using margin maximization between two classes, which consists in finding the separating hyperplane that is furthest from the closest object. Suppose that a binary classification problem with a set of training patterns $\{(x_i, y_i)\}_{i=1}^{N}; x_i \in \boldsymbol{R^d}$ and $y_i \in \{-1, 1\}$ is given.

To separate these patterns into two classes proficiently, SVM finds a classifier basing on a decision function (for the input pattern x) having the form $f(x) = w \cdot \phi(x)+b$, which leads to $class(x) = sgn(f(x))$, where $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$ is a kernel operation. The primal formulation of QPSVM is as follows:

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

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

$$\zeta_i \geq 0;\ i = 1, 2, \ldots, N \tag{3}$$

where the slack variables $\zeta_i > 0$ stand for the margin outward-deviated patterns (which is the patterns that stay outward from their own class margins) as shown in Fig. 2 and $C > 0$ is a regularizer of the classifier controlling the trade-off between two main intentions of the objective function in (1): One is to minimize the training-error depth, and another is to maximize the margin between two classes optimally. The primal problem in (1)–(3) is solved by introducing the Lagrangian function and after some mathematical procedure, the corresponding dual problem is as follows:

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

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

$$\zeta_i \geq 0;\ i = 1,2,\ldots,N \tag{6}$$

with KKT condition $\alpha_i(y_i(w\cdot\phi(x_i)+b)-1+\zeta_i)=0$. Problem in (4)–(6) is a QP problem as well, and the patterns having $\alpha_i > 0$ are SVs. The optimum values of SV coefficients α are used to find the variables w, b.

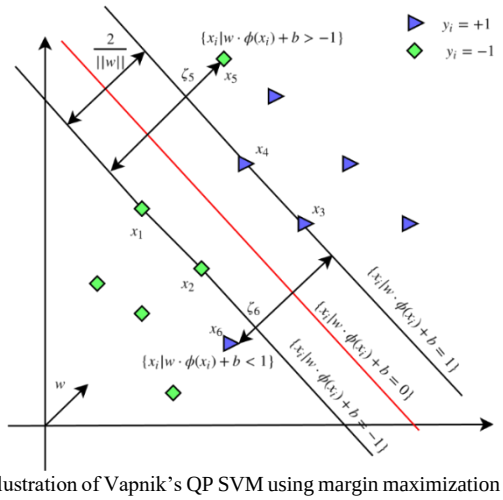


Fig. 2. Illustration of Vapnik's QP SVM using margin maximization concept. The triangles represent positive patterns, and the squares represent negative patterns. The black straight lines indicate the margin of both classes in a hypothetical higher dimensional feature space. Patterns that remain outward from their own class margins are known as margin outward-deviated patterns. The QP SVM tries to maximize the margin while maintaining minimum optimal training-error distances. The training output of QP SVM is the decision boundary represented by the red line.

B. VLPSVM

An LP model is offered by Vapnik to search for a separating hyperplane, which in fact formulates an indirect L_1 norm in the cost function leading to further sparsity with similar accuracy compared with QPSVM that uses the L_2 norm for such. Hence, this machine is similar to the QPSVM one but requires comparatively less computation to classify a pattern. If w be the weight vector of the hyperplane for VLPSVM, its decision function has the form $f(x) = w\cdot\phi(x)+b$ that leads to $class(x) = sgn(f(x))$. The primal formulation of VLPSVM is as follows:

$$\min_{\lambda,\zeta,b} \sum_{i=1}^{N}\lambda_i + C\sum_{i=1}^{N}\zeta_i \tag{7}$$

$$s.t. y_i\left(\sum_{j=1}^{N}\lambda_j y_j \phi(x_j)\cdot\phi(x_i)+b\right) \geq 1-\zeta_i \tag{8}$$

$$\lambda_j \geq 0;\ j = 1, 2,\ldots, N \tag{9}$$

$$\zeta_i \geq 0;\ i = 1, 2,\ldots, N \tag{10}$$

Here, the slack variables $\zeta_i > 0$ stand for unity outward-deviated patterns (that is the training patterns for which $ClassLabel(x_i)\cdot f(x_i) < 1$) and the ζs are optimization variables of the problem in (7)–(10), which are found with the other variable λs. To avoid mystification and clarify sharply, here, the training patterns (x_j) of VLPSVM having coefficients $\lambda_j > 0$ are termed as expansion vectors (EVs) that are similar to the SVs in QPSVM but are not same from both physical and topological point of view. These optimum value of λs are applied to find $w = \Sigma\lambda_j\ y_j\ \phi(x_j)$. The number of EVs in VLPSVM is in general very much smaller with respect to the total training set size proving VLPSVM to be sparser. Our benchmarking and other reports [21] show that VLPSVM has similar performance as QPSVM while being more efficient with less kernel evaluation in classification phase showing VLPSVM to be empirically sound and more productive.

III. Proposed Method with Z-Analysis

This section presents the basic idea before going to the proposed method with analysis.

A. Basic Idea

Sparseness of SVM heavily depends on data complexity, and in case of a very noisy dataset, a good generalized SVM is likely to get more outliers that are included into the SV set. Hence, the motivation is to throw the least contributing SVs from the SV set without hampering generalization capability.

In SVM, the only error-related constraint in (2) and (8) can be rewritten as $y_i f(x_i) \geq 1-\zeta_i \Rightarrow \zeta_i \geq 1-y_i f(x_i)$. Now, for any misclassified training pattern (x_i), if $y_i f(x_i) = -\varepsilon_i$ where $\varepsilon_i > 0$, then $\zeta_i \geq 1+\varepsilon_i$. Hence, for L misclassified training patterns, $\sum_{i=1}^{L}\zeta_i \geq L + \sum_{i=1}^{L}\varepsilon_i$. Thus, suppressing $\sum_i \zeta_i$ by penalizing through regularizer will suppress $L + \sum_{i=1}^{L}\varepsilon_i$, not just L, the number of total training-error. Suppressing $L + \sum_{i=1}^{L}\varepsilon_i$ could happen in the following three ways:

i) Reducing L faster than $\sum_{i=1}^{L}\varepsilon_i$, which will reduce the training-error rate
ii) Reducing L slower than $\sum_{i=1}^{L}\varepsilon_i$, which may bring undesired events on training performance
iii) Reducing L and $\sum_{i=1}^{L}\varepsilon_i$ with same rate and priority, which may not reduce the training-error rate as expected

Therefore, throwing P ($P \leq L < N$) outliers apparently reduces P training errors and also reduces P kernel evaluations in classification phase if these patterns are also members of the final discriminator. However, how this throwing will behave in classifying other patterns is a serious issue to consider. We find it by analyzing the contribution and confusion due to them on correctly classifying other patterns and call it ζ-analysis, which is described in the next parts.

The discriminators both from QPSVM and VLPSVM can be rewritten in the following common form by using a common term "kernel computing vector (KCV)" for SV or EV or such other kernel operating patterns:

$$f(z) = \sum_{l\in KCVset}\lambda_l y_l K(x_l, z) + b$$

where z is any pattern and $K(x_l, z) = \phi(x_l).\phi(z) = e^{-\frac{(x_l - z)^2}{2\sigma^2}}$ is the kernel, which is the source of main computational expenses, and all $\lambda_l > 0$ couple this in the accumulation of time in classification phase. Our main goal is to find a subset of x_l patterns for which $\lambda_l > 0$ by throwing the patterns not having fruitful contribution in decision making for being outlier. This is what we mainly seek in our ζ-analysis discussed next.

B. ζ-Analysis

1) Patterns' position basing on ζ: As discussed earlier, an "outlier" is a pattern that stay outside of its own class boundary. Hence, a pattern is "inlier" if it stays inside its own class boundary. Thus, for any training pattern, x_i with label y_i and the decision function value $f(x_i)$ is an inlier if $y_i = \text{sgn}(f(x_i)) \Rightarrow 1 = y_i\,\text{sgn}(f(x_i)) \Rightarrow y_i f(x_i) > 0$ and outlier if $y_i = -\text{sgn}(f(x_i)) \Rightarrow -1 = y_i\,\text{sgn}(f(x_i)) \Rightarrow y_i f(x_i) < 0$. Re-

writing the error constraints of SVM in (2) or in (8) in a single form using the slack variable ζ as $y_i f(x_i) \geq 1 - \zeta_i \Rightarrow \zeta_i \geq 1 - y_i f(x_i)$, we get for outlier $\zeta_i > 1$ and for inlier $0 \leq \zeta < 1$ considering the following two cases: (i) inlier with $y_i f(x_i) < 1 \Rightarrow 1 - y_i f(x_i) > 0 \Rightarrow \zeta_i > 0$ and (ii) inlier with $y_i f(x_i) > 1 \Rightarrow 1 - y_i f(x_i) < 0 \Rightarrow \zeta_i = 0$ as ζ_i is non-negative.

From the discussion above, it is ensured that patterns having $\zeta > 1$ stay in another class, not in their own classes. Now, we investigate their importance considering their roles in the final discriminator both in QPSVM and VLPSVM:

2) Patterns' role in the final discriminator having $\zeta > 1$:

a) Role in case of QPSVM: From its KKT condition $\alpha_i(y_i(w\cdot\phi(x_i)+b)-1+\zeta_i)=0$, for $\alpha_i \neq 0$, $y_i(w\cdot\phi(x_i)+b)-1+\zeta_i=0 \Rightarrow y_i(w\cdot\phi(x_i)+b)=1-\zeta_i$, which shows that for QPSVM, there are three types of KCVs (patterns having $\alpha_i \neq 0$) basing on ζ-values: (i) $\zeta = 0$, (ii) $0 < \zeta \leq 1$, and (iii) $\zeta > 1$. Using another KKT condition, it is found that for patterns having $\zeta > 0$, have $\alpha = C =$ *penalty parameter*, which is the highest among α values. Hence, throwing $\zeta > 1$ patterns not only violates a KKT condition but also throws members of the decision function with highest coefficient values that may influence the decision making significantly. Moreover, throwing patterns with $\alpha = C$ may severely affect the constraint $\sum_{i=1}^{N}\alpha_i y_i = 0$, which maintains the equilibrium of the SV patterns and coefficients oriented mechanical analogous system for the optimum margin-based final discriminator. Therefore, we do not throw any pattern of this machine.

b) Role in the case of VLPSVM: Here, the optimal values of EV coefficients along with the slack variables ζ are extracted from the optimization problem solved through primal leading no KKT condition or other constraint-based restriction involving the coefficient of the KCVs. These give us a chance to work on this machine. Experience shows that in VLPSVM, patterns having $\zeta > 1$ gets comparatively very small values for their corresponding coefficients λ in the final decision function, which also makes sense after focusing on the first part and the second part of its cost function $\min_{\lambda,\zeta,b}\sum_{i=1}^{N}\lambda_i + C\sum_{i=1}^{N}\zeta_i$. However, agonizingly, these coefficient values do not vanish completely, which lead the corresponding patterns to be members of the decision function where each of them demands kernel evaluation to classify pattern that may not have significant and desired influence for correct decision. This inspires us to reconsider whether they should be included in the final discriminator concerning computational cost and accuracy. For this, we analyze their contribution and confusion for correct classification as discussed below:

3) Contribution and confusion due to $\zeta > 1$ patterns in VLPSVM: These patterns, always staying inside the opposite class boundaries, give higher kernel values with respect to the patterns of opposite class leading to confusion and lower kernel values with respect to the patterns of their own classes leading to mild contribution in the final decision function. Although some inlier KCVs may also create confusion by attracting patterns from opposite classes, they are not considered here to discuss as they are under a proper system where this confusion is very weak.

The decision function that classifies any arbitrary training pattern x_a having label y_a is $f(x_a)=\sum_{l\in KCVset}\lambda_l y_l K(x_l,x_a)+b$. If we consider m,n: $m \cup n = \{l\}, m \cap n = \{\}, y_m = -y_n$, then x_m, x_n are KCVs. Now as $\lambda_m, \lambda_n > 0$ & $K(x_m,x_a)$, $K(x_n,x_a) > 0$, if $\frac{\lambda_m y_m K(x_m,x_a)}{y_a} < 0 \Rightarrow \frac{y_m}{y_a} < 0 \Rightarrow$, x_a and x_m are from opposite classes and x_m pushes x_a outward to x_a's opposite class, and if $\frac{\lambda_n y_n K(x_n,x_a)}{y_a} > 0 \Rightarrow \frac{y_n}{y_a} > 0 \Rightarrow$, x_a and x_n are from the same class and x_n attracts x_a to x_a's own class. But among these KCVs, those having $\zeta > 1$ stay inside the boundaries of opposite classes give higher kernel values with the patterns of opposite classes that misleads the decision function and less kernel values with patterns of own classes that weakly leads the decision function. Thus aiming at higher classification speed and accuracy, we decide to throw these $\zeta > 1$ KCVs after justifying their depth of confusion (pushing any training pattern outward from the pattern's own class) and contribution (attracting any training pattern toward the pattern's own class) with respect to all training patterns using the following calculation:

Contribution depth of a $\zeta > 1$ KCV, x_O with label y_O on all training patterns, $ContrDepth(x_o)=\frac{1}{Card\{i\}}\sum_{\{i:y_i.y_o=1\}}\lambda_{x_o}K(x_o,x_i)$ and thus contribution of x_o on all training patterns, $Contr(x_o)=\frac{1}{Card\{i\}}\sum_{\{i:y_i.y_o=1\}}K(x_o,x_i)$, whereas confusion depth of that KCV, x_O on all training patterns $ConfDepth(x_o)=\frac{1}{Card\{i\}}\sum_{\{i:y_i.y_o=-1\}}\lambda_{x_o}K(x_o,x_i)$, and thus confusion of x_O on all training patterns, $Conf(x_o)=\frac{1}{Card\{i\}}\sum_{\{i:y_i.y_o=-1\}}K(x_o,x_i)$.

Algorithm 1: Proposed RLPSVM (reduced LPSVM)

1: **Input:** A training set $\{(x_i, y_i)\}_{i=1}^{N}$
2: **Output:** A discriminator $f_{RLPSVM}(\cdot)$
3: Select the best (*penalty parameter*, *kernel parameter*) $\equiv (C,\sigma)$.
4: Run VLPSVM on the training set solving the following problem:

$$\min_{\lambda,\zeta,b}\sum_{i=1}^{N}\lambda_i + C\sum_{i=1}^{N}\zeta_i$$

$$s.t. y_i(\sum_{j=1}^{N}\lambda_j y_j \phi(x_j)\cdot\phi(x_i)+b) \geq 1-\zeta_i$$

$$\lambda_j \geq 0;\ j = 1,2,\ldots,N$$

$$\zeta_i \geq 0;\ i = 1,2,\ldots,N$$

5: Extract KCVs with labels $\{(x_p, y_p)\}_{p=1}^{M}$ and the bias b from the VLPSVM using $\lambda_l > 0$.
6: Find $\{\lambda_p|\zeta_p > 1\}$ and throw the corresponding patterns from the KCV set to create a new reduced KCV (RKCV) set, $\{x_p|\zeta_p \leq 1$ *and* $\lambda_p > 0\}$.
7: Extract RKCVs with labels $\{(x_j, y_j)\}_{j=1}^{R}$.
8: $w_{RLPSVM} \leftarrow \sum_{j=1}^{R}\lambda_j y_j \phi(x_j); b_{RLPSVM} \leftarrow b$
9: Return $f_{RLPSVM}(\cdot) = w_{RLPSVM}\cdot\phi(\cdot) + b_{RLPSVM}$

Numerous experiments on benchmark datasets using the mathematical expressions above show that in almost all cases, the $\zeta > 1$ KCVs offer more confusion than contribution that supports our approach to throw $\zeta > 1$ KCVs from the final discriminator. Fig. 3 is an example of such contribution confusion plot. Algorithm 1 summarizes the steps involved in

the proposed reduced LPSVM, whereas Fig. 4 shows the decision boundaries on benchmark Banana data for QPSVM, VLPSVM, and the proposed RLPSVM with corresponding number of KCVs.

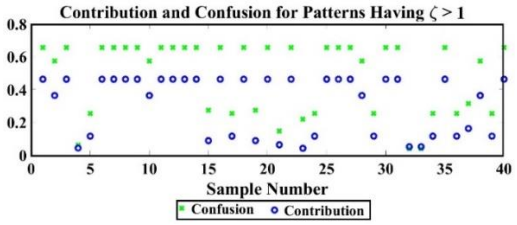
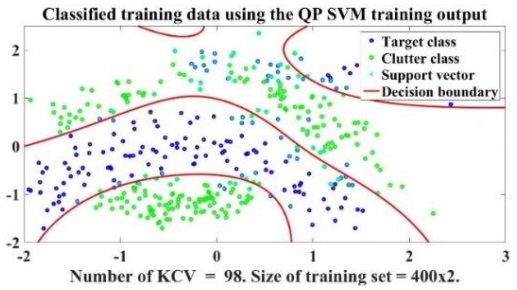
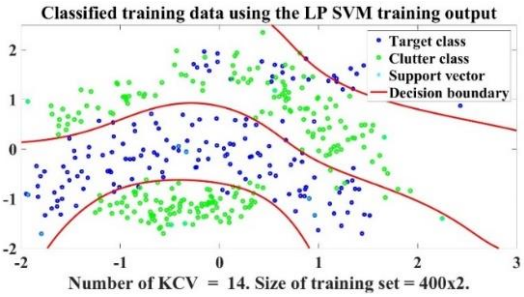


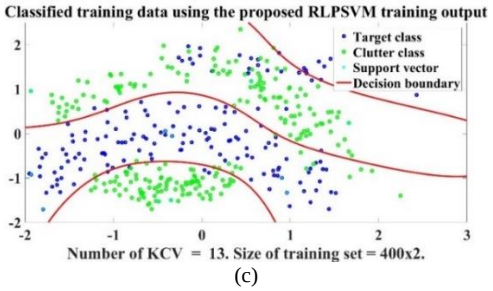
Fig. 3. Contribution and confusion plot of training patterns having $\zeta > 1$ from Titanic dataset.



(a)



(b)



(c)

Fig. 4. Decision boundaries of (a) QPSVM, (b) VLPSVM, and (c) proposed RLPSVM on benchmark Banana data with corresponding number of KCVs.

IV. Experimental Set-up and Results

This section presents the efficiency of the proposed reduced LPSVM algorithm by comparing its experimental results with the results obtained by QPSVM and VLPSVM. The results are presented in two aspects, such as the performance and the classification cost of the algorithms. To evaluate the performance of the methods, nine benchmark datasets, namely, Banana, Breast cancer, Diabetes, Flare-solar, Heart, Thyroid, Titanic, Twonorm, and Waveform, publicly available in [22] are used. All of these data have 100 realizations of training and test set. All the experiments are done using Gaussian kernel. The results are obtained using separate fivefold cross-validation scheme for each method. A varied range of $C \in \{2^{-2},2^{0},2^{2},...,2^{12}\}$ and $\sigma \in \{2^{-2},2^{0},2^{2},...,2^{6}\}$ are used in cross-validation, and the best C and the best σ are extracted for each method. The average number of KCVs and test error rate of QPSVM, VLPSVM, and the proposed method are presented in Table I. It shows that the proposed RLPSVM offers similar accuracy with a significant reduction in the number of KCVs compared with QPSVM and VLPSVM with mean number of KCVs 35.06, which is 16% of QPSVM and 68% of VLPSVM.

TABLE I: Benchmarking Number of KCVs and Test Error Rate Using Different Machines

Dataset	Number of training patterns	Number of test patterns	Data Dimension	Mean KCV in QPSVM	Mean Test Error Rate in QPSVM	Mean KCV in VLPSVM	Mean Test Error Rate in VLPSVM	Mean KCV in RLPSVM	Mean Test Error Rate in RLPSVM	$\frac{KCV_{RLPSVM}}{KCV_{QPSVM}}$	$\frac{KCV_{RLPSVM}}{KCV_{VLPSVM}}$
Banana	400	4900	2	102.26	10.61	15.08	10.75	13.47	11.38	0.13	0.89
Breast	200	77	9	200.00	28.53	18.89	26.05	15.46	28.21	**0.07**	0.82
Diabetes	468	300	8	263.22	23.28	12.58	23.40	6.69	23.67	**0.03**	**0.53**
Flare	666	400	9	609.98	32.41	251.66	32.56	140.76	32.44	0.23	**0.56**
Heart	170	100	13	68.23	16.60	21.94	17.44	8.07	15.51	0.12	**0.37**
Thyroid	140	75	5	43.51	5.20	8.97	5.09	8.97	5.09	0.21	1.00
Titanic	150	2051	3	148.50	22.68	83.91	22.91	82.22	22.91	0.55	0.98
Twonorm	400	7000	20	299.18	2.42	32.00	3.71	19.27	3.70	**0.06**	0.60
Waveform	400	4600	21	228.33	13.15	20.81	11.18	20.64	10.44	**0.09**	0.99
Average	–	–	–	218.13	17.21	51.76	17.01	35.06	17.04	**0.16**	**0.68**

Number of KCVs (the patterns that build the machine by using kernel executions) needed by our proposed machine RLPSVM and two state-of-the-art machines QPSVM and VLPSVM with the test error rates due to these machines on different datasets of benchmark data [22]. It can be seen that, in the case of all nine datasets, RLPSVM needs much smaller number of kernel executions compared with the QPSVM, which is sparse. RLPSVM demands as little as 3% of QPSVM while classifying the *Diabetes* data and on an average 16% considering all nine datasets. However, despite this heavy classification cost reduction, classification accuracy of RLPSVM is very similar to the complex and powerful QPSVM posing only 0.39% less in case of *Diabetes* dataset and amazingly 0.17% more on an average considering all nine datasets. Moreover, It can be observed that, in the case of all nine datasets, the proposed RLPSVM requires much smaller or equal (in its worst case) to the number of kernel executions compared with the VLPSVM. RLPSVM demands as little as 37% kernel executions of VLPSVM in the case of predicting the *Heart* dataset and on an average 68% considering all nine datasets. Nevertheless, despite this heavy classification cost reduction, its accuracy is very similar to the powerful VLPSVM posing interestingly 1.93% more in the case of *Heart* dataset and only 0.03% less on average considering all datasets. Notably, regarding prediction accuracy, RLPSVM outperforms other two machines separately on four of these datasets.

V. Conclusion and Future Work

In this paper, we realize a very fast classifier with high accuracy by further sparsifying Vapnik's LP SVM, which is already sparser. The proposed classifier requires kernel computation as small as up to 3% of the sparse QP SVM and 37% of the sparser LP SVM but in average 16% of the QP SVM and 68% the LP SVM though posing very similar classification accuracy while being straight forward demanding least effort to train. One plausible explanation for

its exceptional performance could be that the "ζ-analysis" with contribution and confusion gives two folded benefits: (i) An indirect filtering is designed by taking only those patterns (in the final discriminator) that are useful for classification with high accuracy. (ii) We further sparsify an already sparser classifier by enabling much smaller number of KCVs in the final decision function by throwing misleading KCVs (that have undesired influence on the final discriminator), which helps both in accelerating the classification process and focusing on high accuracy with reducing the risk of over-fitting simultaneously.

Our further manipulation after the solution from VLPSVM fills the very minor gaps that remain from SVM method, which are the following:

i) It does not emphasize on the accuracy of the classifier directly, instead it focuses on the error-depth minimization. (ii) It is difficult and not so certain to get the best values of the continuous-valued regularization parameters through cross-validation, which may lead to offering more patterns to be candidate for KCVs that increases both computational cost and the probability for over-fitting.

ii) SVM problem is based on a full set of training patterns that also include the heavy outlier that is highly deviated and contaminated by noise. Inclusion of such a single pattern may bring a dramatic change in the solution.

iii) Often, a number of SVs may vary heavily for very similar accuracy from the classifier. This could be an indirect effect from the outliers.

Finally, while an efficient and accurate classifier like the proposed reduced LPSVM is very much essential, some future manipulations can be considered with it like to analytically find the number of $\zeta > 1$ patterns to throw and the reduction rate of kernel computation relating classifier's performance.

REFERENCES

[1] R. Karim and A. K. Kundu, "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 (IJACSA)*, vol. 9, no. 2, pp. 311–318, 2018.

[2] S. S. Keerthi, O. Chapelle, and D. DeCoste, "Building support vector machines with reduced classifier complexity," *Journal of Machine Learning Research, (JMLR)*, vol. 7, no. Jul, pp. 1493–1515, 2006.

[3] A. Cotter, S. Shalev-Shwartz, and N. Srebro, "Learning optimally sparse support vector machines," in *Proc. 30th Int'l Conf. on Machine Learning (ICML)*, 2013, pp. 266–274.

[4] T. Joachims and C. N. J. Yu, "Sparse kernel svms via cutting-plane training," in *Proc. European Conf. on Machine Learning and Knowledge Discovery in Databases (ECML)*, 2009, p. 8.

[5] F. Vojtech and V. Hlavac, "Greedy algorithm for a training set reduction in the kernel methods," in *Proc. Int'l Conf. on Computer Analysis of Images and Patterns*. Springer, 2003, pp. 426–433.

[6] M. Ratsch, S. Romdhani, G. Teschke, and T. Vetter, "Over-complete wavelet approximation of a support vector machine for efficient classification," in *Proc. Joint Patt. Recog. Symp.* Springer, 2005, pp. 351–360.

[7] S. Romdhani, P. Torr, B. Scholkopf¨ and A. Blake, "Efficient face detection by a cascaded support–vector machine expansion," in *Proc. Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 460, no. 2051, 2004, pp. 3283–3297.

[8] M. Wu, B. Scholkopf, and G. Bakır, "A direct method for building sparse kernel learning algorithms," *Journal of Machine Learning Research (JMLR)*, vol. 7, no. Apr, pp. 603–624, 2006.

[9] L. Ladicky and P. H. S. Torr, "Locally linear support vector machines," in *Proc. 28th Int'l Conf. on Machine Learning, (ICML)*, 2011, pp. 985–992.

[10] Z. E. Xu, J. R. Gardner, S. Tyree, and K. Q. Weinberger, "Compressed support vector machines," *arXiv preprint arXiv:1501.06478*, 2015.

[11] Z. Fu, A. R. Kelly, and J. Zhou, "Mixing linear SVMs for nonlinear classification," *IEEE Trans. Neural Networks*, vol. 21, no. 12, pp. 1963– 1975, 2010.

[12] Q. Gu and J. Han, "Clustered support vector machines," in *Proc. 16th Int'l Conf. on Artificial Intelligence and Statistics (AISTATS)*, 2013, pp. 307–315.

[13] T. Downs, K. E. Gates, and A. Masters, "Exact simplification of support vector solutions," *Journal of Machine Learning Research*, vol. 2, no. Dec, pp. 293–297, 2001.

[14] K. Z. Arreola, J. Fehr, and H. Burkhardt, "Fast support vector machine classification using linear SVMs," in *Proc. the 18th International Conference on Pattern Recognition (ICPR)*, vol. 3, 2006, pp. 366–369.

[15] B. Heisele, T. Serre, S. Prentice, and T. Poggio, "Hierarchical classification and feature reduction for fast face detection with support vector machines," *Pattern Recognition*, vol. 36, no. 9, pp. 2007–2017, 2003.

[16] R. Karim, M. Bergtholdt, J. H. Kappes, and C. Schnorr, "Greedy-based design of sparse two-stage SVMs for fast classification," in *Proc. the 29th DAGM Symposium on Pattern Recognition*, 2007, pp. 395–404.

[17] H. Sahbi and D. Geman, "A hierarchy of support vector machines for pattern detection," *Journal of Machine Learning Research (JMLR)*, vol. 7, no. Oct, pp. 2087–2123, 2006.

[18] X. Huo and J. Chen, "Building a cascade detector and applications in automatic target detection," *Applied Optics: Information Processing*, vol. 43, no. 2, pp. 1–47, 2003.

[19] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[20] V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[21] A. Nefedov, J. Ye, C. Kulikowski, I. Muchnik, and K. Morgan, "Experimental study of support vector machines based on linear and quadratic optimization criteria," DIMACS Technical Report 2009-18, 2009.

[22] G. Ratsch. IDA benchmark data sets. [Online] Available: http://www.raetschlab.org/ Members/raetsch/benchmark/

Rezaul Karim completed a B.Sc. (Honours) in mathematics from Jahangirnagar University, Bangladesh in 1998 and a B.Sc. in electronic and computer engineering from Engineering College of Copenhagen, Denmark in 2003. He then completed his M.Sc. in telecommunication from the Technical University of Denmark in 2005. He was awarded Marie-Curie scholarship and served as an EU-Guest researcher in Germany from 2005 to 2008. Currently, he is serving as an assistant professor at the Department of Electrical and Electronic Engineering, Uttara University, Bangladesh. He is interested to work in statistical machine learning, optimization.

Amit Kumar Kundu finished a B.Sc. (with honours) in electrical and electronic engineering (EEE) from Bangladesh University of Engineering Technology (BUET) in 2016. He has just defended his M.Sc. thesis to complete his M.Sc. degree in EEE from BUET. Currently, he is serving as a senior lecturer at the Department of Electrical and Electronic Engineering, Uttara University, Bangladesh. He is interested in statistical machine learning, optimization.