Expert Systems with Applications 36 (2009) 7535–7543

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Least squares twin support vector machines for pattern classification

M. Arun Kumar *, M. Gopal

Control Group, Department of Electrical Engineering, Indian Institute of Technology Delhi, Hauz Khas, New Delhi 110016, India

ARTICLE INFO

Keywords:
Pattern classification
Support vector machines
Machine learning
Proximal classification
Text categorization

ABSTRACT

In this paper we formulate a least squares version of the recently proposed twin support vector machine (TSVM) for binary classification. This formulation leads to extremely simple and fast algorithm for generating binary classifiers based on two non-parallel hyperplanes. Here we attempt to solve two modified primal problems of TSVM, instead of two dual problems usually solved. We show that the solution of the two modified primal problems reduces to solving just two systems of linear equations as opposed to solving two quadratic programming problems along with two systems of linear equations in TSVM. Classification using nonlinear kernel also leads to systems of linear equations. Our experiments on publicly available datasets indicate that the proposed least squares TSVM has comparable classification accuracy to that of TSVM but with considerably lesser computational time. Since linear least squares TSVM can easily handle large datasets, we further went on to investigate its efficiency for text categorization applications. Computational results demonstrate the effectiveness of the proposed method over linear proximal SVM on all the text corpuses considered.

1. Introduction

Support vector machines (SVMs), being computationally powerful tools for supervised learning, are widely used in classification and regression problems. SVMs have been successfully applied to a variety of real-world problems like particle identification, face recognition, text categorization and bioinformatics (Burges, 1998). The approach is systematic and motivated by statistical learning theory (SLT) and Bayesian arguments. The central idea of SVM is to find the optimal separating hyperplane between the positive and negative examples. The optimal hyperplane is defined as the one giving maximum margin between the training examples that are closest to the hyperplane.

Recently proposed (Mangasarian & Wild, 2006) generalized eigenvalue proximal SVM (GEPSVM), does binary classification by obtaining two non-parallel hyperplanes, one for each class. In this approach, datapoints of each class are clustered around the corresponding hyperplane. The new datapoints are assigned to a class based on its proximity to one of the two hyperplanes. This formulation leads to two generalized eigenvalue problems, whose solutions are obtained as eigenvectors corresponding to the smallest eigenvalues.

Jayadeva, Khemchandani, and Chandra (2007) proposed twin SVM (TSVM) which is similar in spirit to GEPSVM that obtains two non-parallel hyperplanes by solving two novel formulations of quadratic programming problems (QPPs). The idea is to solve two dual QPPs of smaller size rather than solving single dual QPP with large number of parameters in conventional SVM. Experimental results of Jayadeva et al. (2007) show the effectiveness of TSVM over GEPSVM and standard SVM on UCI datasets.

In this paper we have enhanced TSVM to least squares TSVM (LSTSVM) using the idea proposed in Fung and Mangasarian (2001) and Suykens and Vandewalle (1999). We first modified the primal QPPs of TSVM in least squares sense and solved them with equality constraints instead of inequalities of TSVM. As a result the solution of LSTSVM follows directly from solving two systems of linear equations as opposed to solving two QPPs and two systems of linear equations in TSVM. We extended LSTSVM to handle nonlinear kernels whose solution also leads to systems of linear equations. The algorithm can accurately solve large datasets without any external optimizers. Computational comparisons of LSTSVM, TSVM, GEPSVM and proximal SVM (PSVM) (Fung & Mangasarian, 2001) in terms of classification accuracy have been made on 11 UCI datasets and several artificial datasets for both linear and nonlinear kernels. Training time comparison of LSTSVM and TSVM shows that the proposed method is faster in both linear and nonlinear cases.

Given the two facts: (1) LSTSVM surpasses TSVM in speed and gives very comparable classification accuracy; (2) TSVM has better generalization than conventional SVM and GEPSVM, we explored the application of linear LSTSVM to text categorization (TC) problems. TC is the task of assigning a given text document to one or more predefined categories. It is gaining popularity due to the increased availability of documents in digital form and the following need to access them in flexible ways. Automatic indexing of

* Corresponding author. Tel.: +91 11 26596133; fax: +91 11 26581606.
E-mail address: sendtoarun@rediffmail.com (M. Arun Kumar).

0957-4174/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2008.09.066

scientific articles, hierarchical categorization of web pages, filtering of spam e-mails, quick search of interesting topics from large databases and retrieving the information based on user's preferences from information sources, are some examples where automatic TC can play a significant role. The suitability of linear LSTSVM for TC has been verified by conducting experiments on three benchmark text corpuses: reuters-21578, ohsumed and 20 Newsgroups (20NG). We also conducted experiments with PSVM on same datasets for comparison.

The paper is organized as follows. In Section 2, we briefly discuss the SVM problem formulation and its dual problem. Section 3 gives a short summary of TSVM. In Section 4, we extend TSVM to LSTSVM for both linear and nonlinear kernels. Computational comparisons on UCI datasets are done in Section 5. In Section 6 we investigate the performance of linear LSTSVM for TC tasks; and Section 7 gives concluding remarks.

In this paper, all vectors will be column vectors unless transformed to a row vector by a prime $'$. A column vector of ones in real space of arbitrary dimension will be denoted by e. For a matrix $A \in \Re^{l\times n}$, A_i is the ith row of A which is a row vector in $\Re^n$. For a vector $x \in \Re^n$, x^* denotes the vector in $\Re^n$ with components $(x^*)_i = 1$ if $x_i > 0$ and 0 otherwise; $i = 1,\ldots,n$. In other words x^* is the result of applying step function component-wise to x. Identity matrix of arbitrary dimension will be denoted by I.

2. Support vector machines

SVMs represent novel learning techniques that have been introduced in the framework of structural risk minimization (SRM) and in the theory of VC bounds. Compared to state-of-the-art methods, SVMs have showed excellent performance in pattern recognition tasks. In the simplest binary pattern recognition tasks, SVMs use a linear separating hyperplane to create a classifier with maximal margin. Consider the problem of binary classification wherein a linearly inseparable dataset X of l points in real n-dimensional space of features is represented by the matrix $X \in \Re^{l\times n}$. The corresponding target or class of each datapoint X_i; $i = 1,2,\ldots,l$, is represented by a diagonal matrix $D \in \Re^{l\times l}$ with entries D_{ii} as +1 or −1. Given the above problem, SVM's linear softmargin algorithm is to solve the following primal QPP (Burges, 1998):

$$\begin{aligned} \underset{w,b}{\text{Min}} \quad & \frac{1}{2}w'w + Ce'y \\ \text{subject to} \quad & D(Xw + eb) + y \geqslant e,\ y \geqslant 0e. \end{aligned} \tag{1}$$

where C is a penalty parameter and y are the nonnegative slack variables. The optimal separating hyperplane can be expressed as

$$d(x) = w'x + b, \tag{2}$$

where $x \in \Re^n$ is a test datapoint. Since the number of constraints in (1) is large, the dual of (1) is usually solved. The Wolfe dual of (1) is (Mangasarian, 1998):

$$\begin{aligned} \underset{\alpha}{\text{Max}} \quad & e'\alpha - \frac{1}{2}\alpha'DXX'D\alpha \\ \text{subject to} \quad & e'D\alpha = 0, \quad 0e \leqslant \alpha \leqslant Ce \end{aligned} \tag{3}$$

where $\alpha \in \Re^l$ are Lagrangian multipliers. The optimal separating hyperplane is same as in (2) whose parameters are given by

$$w = X'D\alpha; \quad b = \frac{1}{N_{sv}}\sum_{i=1}^{N_{sv}} D_{ii} - X_i w, \tag{4}$$

where N_{sv} represents the number of support vectors such that $0 < \alpha_i < C$. The hyperplane described by (2) lies midway between the bounding planes given by

$$w'x + b = 1 \text{ and } w'x + b = -1, \tag{5}$$

and separates the two classes from each other with a margin of $\frac{2}{\|w\|_2}$. A new datapoint is classified as +1 or −1 according to whether the decision function $(w'x + b)^*$ yields 1 or 0 respectively. Fig. 1 shows the geometric interpretation of this formulation for a toy example. An important characteristic of SVM is that it can be extended in a relatively straightforward manner to create nonlinear decision boundaries (Scholkopf & Smola, 2002).

3. Twin support vector machine

TSVM is a binary classifier that does classification using two non-parallel hyperplanes instead of a single hyperplane as in the case of conventional SVMs (Jayadeva et al., 2007). The two non-parallel hyperplanes are obtained by solving two QPPs of smaller size compared to a single large QPP solved by conventional SVMs. Consider a binary classification problem of classifying m_1 datapoints belonging to class +1 and m_2 datapoints belonging to class −1 in the n-dimensional real space $\Re^n$. Let matrix A in $\Re^{m_1\times n}$ represent the datapoints of class +1 and matrix B in $\Re^{m_2\times n}$ represent the datapoints of class −1. Given the above stated binary classification problem, linear TSVM seeks two non-parallel hyperplanes in $\Re^n$:

$$x'w^{(1)} + b^{(1)} = 0 \text{ and } x'w^{(2)} + b^{(2)} = 0, \tag{6}$$

such that each hyperplane is closest to datapoints of one class and farthest from the datapoints of other class. A new datapoint is assigned to class +1 or −1 depending upon its proximity to the two non-parallel hyperplanes. Geometrically the concept of TSVM is depicted in Fig. 2 for a toy example.

The idea in linear TSVM is to solve two QPPs (7) and (8) with objective function corresponding to one class and constraints corresponding to the other class.

$$\begin{aligned} \underset{w^{(1)},b^{(1)}}{\text{Min}} \quad & \frac{1}{2}(Aw^{(1)} + eb^{(1)})'(Aw^{(1)} + eb^{(1)}) + C_1e'y \\ \text{subject to} \quad & -(Bw^{(1)} + eb^{(1)}) + y \geqslant e, \quad y \geqslant 0e. \end{aligned} \tag{7}$$

$$\begin{aligned} \underset{w^{(2)},b^{(2)}}{\text{Min}} \quad & \frac{1}{2}(Bw^{(2)} + eb^{(2)})'(Bw^{(2)} + eb^{(2)}) + C_2e'y \\ \text{subject to} \quad & (Aw^{(2)} + eb^{(2)}) + y \geqslant e, \quad y \geqslant 0e. \end{aligned} \tag{8}$$

The Wolfe dual of QPPs (7) and (8) has been shown in Jayadeva et al. (2007) to be QPPs (9) and (10) in terms of the Lagrangian multipliers $\alpha \in \Re^{m_2}$ and $\beta \in \Re^{m_1}$, respectively.

$$\begin{aligned} \underset{\alpha}{\text{Max}} \quad & e'\alpha - \frac{1}{2}\alpha'G(H'H)^{-1}G'\alpha \\ \text{subject to} \quad & 0e \leqslant \alpha \leqslant C_1e \end{aligned} \tag{9}$$

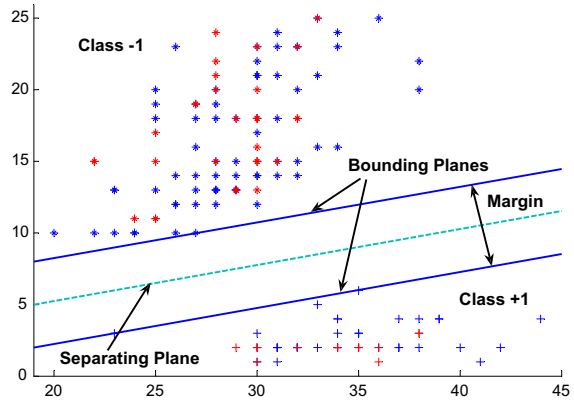

Fig. 1. Geometric interpretation of standard SVM.

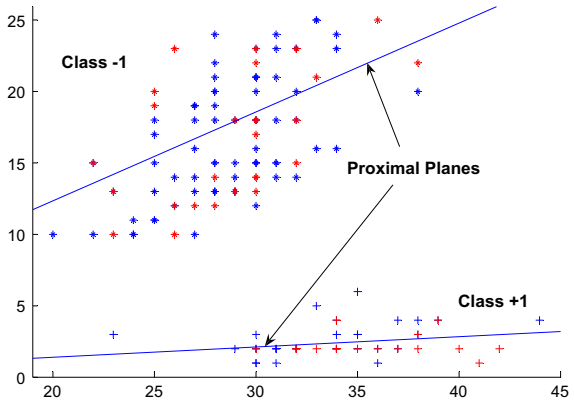

Fig. 2. Geometric interpretation of TSVM.

where $G = [B \quad e]$ and $H = [A \quad e]$

$$\begin{aligned} &\underset{\beta}{\text{Max}} && e'\beta - \frac{1}{2}\beta' P (Q'Q)^{-1} P'\beta \\ &\text{subject to} && 0e \leqslant \beta \leqslant C_2 e \end{aligned} \tag{10}$$

where $P = [A \quad e]$ and $Q = [B \quad e]$.

The non-parallel hyperplanes (6) can be obtained from the solution of QPPs (9) and (10), as given in (11) and (12), respectively.

$$v_1 = -(H'H)^{-1}G'\alpha, \text{ where } v_1 = \begin{bmatrix} w^{(1)} & b^{(1)} \end{bmatrix}' \tag{11}$$

$$v_2 = (Q'Q)^{-1}P'\beta, \text{ where } v_2 = \begin{bmatrix} w^{(2)} & b^{(2)} \end{bmatrix}' \tag{12}$$

Solving two dual QPPs has the advantage of bounded constraints and reduced number of parameters as QPP (9) has only m_2 parameters and QPP (10) has only m_1 parameters, when compared with QPP (3) of SVM which has $l = m_1 + m_2$ parameters. However, it is to be noted that in addition to solving dual QPPs (9) and (10), TSVM also requires inversion of matrix of size $(n+1) \times (n+1)$ twice, where $n \ll l$. The datapoints for which $0 < \alpha_i < C_1 (i = 1,2, \ldots m_2)$ or $0 < \beta_j < C_2$ $(j = 1,2, \ldots m_1)$ are defined as support vectors, as they are significant in determining the hyperplanes (6) and it has been shown in Jayadeva et al. (2007) that, the support vectors will lie on its corresponding hyperplane. Once the non-parallel hyperplanes (6) are obtained, a new datapoint $x \in \Re^n$ is assigned to a class +1 or -1 depending on which of the two hyperplanes lies closest to the point in terms of perpendicular distance.

TSVM was also extended in Jayadeva et al. (2007) to handle nonlinear kernels by considering two non-parallel kernel generated surfaces:

$$K(x', C')u^{(1)} + \gamma^{(1)} = 0 \text{ and } K(x', C')u^{(2)} + \gamma^{(2)} = 0, \tag{13}$$

where $C = \begin{bmatrix} A \\ B \end{bmatrix}$ and K is any arbitrary kernel. The primal QPPs of nonlinear TSVM corresponding to the surfaces (13) are given below in (14) and (15), respectively.

$$\begin{aligned} &\underset{u^{(1)},\gamma^{(1)}}{\text{Min}} && \frac{1}{2}\|(K(A, C')u^{(1)} + e\gamma^{(1)})\|^2 + C_1 e'y \\ &\text{subject to} && -(K(B, C')u^{(1)} + e\gamma^{(1)}) + y \geqslant e, \quad y \geqslant 0e \end{aligned} \tag{14}$$

$$\begin{aligned} &\underset{u^{(2)},\gamma^{(2)}}{\text{Min}} && \frac{1}{2}\|(K(B, C')u^{(2)} + e\gamma^{(2)})\|^2 + C_2 e'y \\ &\text{subject to} && (K(A, C')u^{(2)} + e\gamma^{(2)}) + y \geqslant e, \quad y \geqslant 0e \end{aligned} \tag{15}$$

Dual QPPs of (14) and (15) were derived and solved to get the hyperplanes (13) in Jayadeva et al. (2007). However it is worth mentioning that the solution of nonlinear TSVM requires inversion of two matrices of order $(m_1 \times m_1)$ and $(m_2 \times m_2)$ respectively along with two QPPs to be solved.

TSVMs are similar in spirit to GEPSVM proposed by Mangasarian and Wild (2006), as both does binary classification using two non-parallel hyperplanes as opposed to two parallel hyperplanes used by PSVM. However it is to be noted that, TSVMs may fail in some cases when dataset is perfectly symmetric; at least with linear kernel (a classical XOR example). This problem can be alleviated by either slightly shifting a point, so as to disturb the symmetry or by using nonlinear kernels. Experimental results of Jayadeva et al. (2007) show that, generalization performance of TSVM is better than GEPSVM and conventional SVM on UCI machine learning datasets for both linear and nonlinear cases. Comparison tables on training time between conventional SVM and TSVM for linear kernel reveal that TSVM is at least four times faster than conventional SVM.

4. Least squares twin support vector machine

In this section, we solve the primal QPPs of TSVM rather than dual QPPs using PSVM idea proposed in Fung and Mangasarian (2001). PSVM is an extremely fast and simple algorithm that requires only solution of a system of linear equations for generating both linear and nonlinear classifiers. PSVM formulation is obtained from conventional SVM formulation by modifying inequality constraints to equality constraints. Here we modify the primal problem (7) of linear TSVM in least squares sense as (16), with the inequality constraints replaced with equality constraints as follows:

$$\begin{aligned} &\underset{w^{(1)},b^{(1)}}{\text{Min}} && \frac{1}{2}(Aw^{(1)} + eb^{(1)})'(Aw^{(1)} + eb^{(1)}) + \frac{C_1}{2}y'y \\ &\text{subject to} && -(Bw^{(1)} + eb^{(1)}) + y = e, \end{aligned} \tag{16}$$

Also note that QPP (16) uses the square of 2-norm of slack variables y with weight $\frac{C_1}{2}$ instead of 1-norm of y with weight C_1 as used in (7), which makes the constraint $y \geqslant 0e$ redundant (Mangasarian & Musicant, 2001). This very simple modification allows us to write the solution of QPP (16) as a solution of simultaneous system of linear equations. On substituting the equality constraints into the objective function, QPP (16) becomes:

$$\underset{w^{(1)},b^{(1)}}{\text{Min}} \quad \frac{1}{2}\|Aw^{(1)} + eb^{(1)}\|^2 + \frac{C_1}{2}\|Bw^{(1)} + eb^{(1)} + e\|^2 \tag{17}$$

Setting the gradient of (17) with respect to $w^{(1)}$ and $b^{(1)}$ to zero, gives:

$$A'(Aw^{(1)} + eb^{(1)}) + C_1 B'(Bw^{(1)} + eb^{(1)} + e) = 0e, \tag{18}$$

$$e'(Aw^{(1)} + eb^{(1)}) + C_1 e'(Bw^{(1)} + eb^{(1)} + e) = 0. \tag{19}$$

Arranging (18) and (19) in matrix form and solving for $w^{(1)}$ and $b^{(1)}$ gives:

$$\begin{bmatrix} B'B & B'e \\ e'B & m_2 \end{bmatrix}\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix} + \frac{1}{C_1}\begin{bmatrix} A'A & A'e \\ e'A & m_1 \end{bmatrix}\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix} + \begin{bmatrix} B'e \\ m_2 \end{bmatrix} = 0e \tag{20}$$

$$\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix} = \begin{bmatrix} B'B + \frac{1}{C_1}A'A & B'e + \frac{1}{C_1}A'e \\ e'B + \frac{1}{C_1}e'A & m_2 + \frac{1}{C_1}m_1 \end{bmatrix}^{-1}\begin{bmatrix} -B'e \\ -m_2 \end{bmatrix} \tag{21}$$

$$\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix} = \left(\begin{bmatrix} B' \\ e' \end{bmatrix}[B \quad e] + \frac{1}{C_1}\begin{bmatrix} A' \\ e' \end{bmatrix}[A \quad e]\right)^{-1}\begin{bmatrix} -B'e \\ -m_2 \end{bmatrix} \tag{22}$$

Defining $E = [A \quad e]$ and $F = [B \quad e]$, the solution becomes:

$$\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix} = -\left(F'F + \frac{1}{C_1}E'E\right)^{-1}F'e. \tag{23}$$

In an exactly similar way the solution of QPP (24) can be shown to be (25).

$$\underset{w^{(2)},b^{(2)}}{\text{Min}} \quad \frac{1}{2}(Bw^{(2)}+eb^{(2)})'(Bw^{(2)}+eb^{(2)})+\frac{C_2}{2}y'y$$

$$\text{subject to} \quad (Aw^{(2)}+eb^{(2)})+y=e \tag{24}$$

$$\begin{bmatrix} w^{(2)} \\ b^{(2)} \end{bmatrix} = (E'E+\frac{1}{C_2}F'F)^{-1}E'e \tag{25}$$

Thus the linear LSTSVM completely solves the classification problem with just two matrix inverses of much smaller dimensional matrix of order $(n+1)\times(n+1)$ where $n \ll l$.

Once the weights and biases of the two non-parallel separating hyperplanes

$$x'w^{(1)}+b^{(1)}=0 \text{ and } x'w^{(2)}+b^{(2)}=0, \tag{26}$$

are obtained from (23) and (25), a new datapoint $x \in \Re^n$ is assigned to a class +1 or −1 depending on to which of the two hyperplanes, its perpendicular distance is minimum: $|x'\ w^{(1)}+b^{(1)}|$ or $|x'w^{(2)}+b^{(2)}|$. Here $|\cdot|$ denotes the perpendicular distance of a datapoint from the hyperplane. It can be noted that LSTSVM does not change the meaning of support vectors defined in TSVM, however we will not be able to identify them as we are solving primal problems instead of dual problems. For clarity, we explicitly state our linear LSTSVM algorithm.

Algorithm 4.1. Linear least squares twin SVM

Given m_1 datapoints in $\Re^n$ of class +1 represented by matrix A, m_2 datapoints in $\Re^n$ of class −1 represented by matrix B, linear LSTSVM can be obtained using the following steps:

(i) Define $E=[A \quad e]$ and $F=[B \quad e]$.
(ii) Select penalty parameters C_1 and C_2. Usually these parameters are selected based on validation.
(iii) Determine parameters of two non-parallel hyperplanes using (23) and (25).
(iv) Calculate perpendicular distances $|x'\ w^{(1)}+b^{(1)}|$ and $|x'w^{(2)}+b^{(2)}|$ for a new datapoint $x \in \Re^n$.
(v) Assign the datapoint to class +1 or −1 based on which of the distance $|x'w^{(1)}+b^{(1)}|$ or $|\ x'w^{(2)}+b^{(2)}|$ is minimum.

Following the same idea, we extended nonlinear TSVM to nonlinear LSTSVM by considering the following kernel generated surfaces:

$$K(x',C')u^{(1)}+\gamma^{(1)}=0 \text{ and } K(x',C')u^{(2)}+\gamma^{(2)}=0 \tag{27}$$

where $C=\begin{bmatrix} A \\ B \end{bmatrix}$ and K is any arbitrary kernel. The primal QPPs of nonlinear TSVM can be modified in the same way with 2-norm of slack variables and inequality constraints replaced by equality constraints as shown in (28) and (29).

$$\underset{u^{(1)},\gamma^{(1)}}{\text{Min}} \quad \frac{1}{2}\|(K(A,C')u^{(1)}+e\gamma^{(1)})\|^2+\frac{C_1}{2}y'y$$

$$\text{subject to} \quad -(K(B,C')u^{(1)}+e\gamma^{(1)})+y=e. \tag{28}$$

$$\underset{u^{(2)},\gamma^{(2)}}{\text{Min}} \quad \frac{1}{2}\|(K(B,C')u^{(2)}+e\gamma^{(2)})\|^2+\frac{C_2}{2}y'y$$

$$\text{subject to} \quad (K(A,C')u^{(2)}+e\gamma^{(2)})+y=e. \tag{29}$$

By substituting the constraints into objective function, these QPPs become:

$$\underset{u^{(1)},\gamma^{(1)}}{\text{Min}} \quad \frac{1}{2}\|(K(A,C')u^{(1)}+e\gamma^{(1)})\|^2+\frac{C_1}{2}\|K(B,C')u^{(1)}+e\gamma^{(1)}+e\|^2 \tag{30}$$

$$\underset{u^{(2)},\gamma^{(2)}}{\text{Min}} \quad \frac{1}{2}\|(K(B,C')u^{(2)}+e\gamma^{(2)})\|^2+\frac{C_2}{2}\|-K(A,C')u^{(2)}-e\gamma^{(2)}+e\|^2 \tag{31}$$

The solution of QPPs (30) and (31) can be derived to be:

$$\begin{bmatrix} u^{(1)} \\ \gamma^{(1)} \end{bmatrix} = -\left(H'H+\frac{1}{C_1}G'G\right)^{-1}H'e \tag{32}$$

$$\begin{bmatrix} u^{(2)} \\ \gamma^{(2)} \end{bmatrix} = \left(G'G+\frac{1}{C_2}H'H\right)^{-1}G'e \tag{33}$$

where $G=[K(A,C') \quad e]$ and $H=[K(B,C') \quad e]$.

Once the parameters of two hypersurfaces (27): $u^{(1)}$, $\gamma^{(1)}$, $u^{(2)}$ and $\gamma^{(2)}$, were obtained, a new datapoint is classified in the same way as it is done in linear case based on perpendicular distance.

It can be noted that the solution of nonlinear LSTSVM requires inversion of matrix of size $(l+1)\times(l+1)$ twice. However using Sherman–Morrison–Woodbury (SMW) (Golub & Van Loan, 1996) formula, we show that nonlinear LSTSVM can be solved using three inverses of smaller dimension than $(l+1)\times(l+1)$. Below we discuss the solution of nonlinear LSTSVM for two cases.

Case $m_1 < m_2$:

Using SMW formula we can rewrite (32) and (33) as

$$\begin{bmatrix} u^{(1)} \\ \gamma^{(1)} \end{bmatrix} = -(Y-YG'(C_1I+GYG')^{-1}GY)H'e \tag{34}$$

$$\begin{bmatrix} u^{(2)} \\ \gamma^{(2)} \end{bmatrix} = C_2\left(Y-YG'\left(\frac{I}{C_2}+GYG'\right)^{-1}GY\right)G'e \tag{35}$$

where $Y=(H'H)^{-1}$. Following (Jayadeva et al., 2007), we introduce a regularization term $\varepsilon I, \varepsilon > 0$ to Y to take care of problems due to possible ill-conditioning of $H'H$. This allows us to use SMW formula in finding Y as

$$Y=\frac{1}{\varepsilon}(I-H'(\varepsilon I+HH')^{-1}H) \tag{36}$$

Case $m_2 < m_1$:

Using SMW formula we can rewrite (32) and (33) as

$$\begin{bmatrix} u^{(1)} \\ \gamma^{(1)} \end{bmatrix} = -C_1\left(Z-ZH'\left(\frac{I}{C_1}+HZH'\right)^{-1}HZ\right)H'e \tag{37}$$

$$\begin{bmatrix} u^{(2)} \\ \gamma^{(2)} \end{bmatrix} = (Z-ZH'(C_2I+HZH')^{-1}HZ)G'e \tag{38}$$

where $Z=(G'G)^{-1}$ which can be found using SMW formula as

$$Z=\frac{1}{\varepsilon}(I-G'(\varepsilon I+GG')^{-1}G) \tag{39}$$

Thus for the case of $m_1 < m_2$, nonlinear LSTSVM requires two matrix inverses of order $(m_1\times m_1)$ and one matrix inverses of order $(m_2\times m_2)$. For the case of $m_2 < m_1$ nonlinear LSTSVM requires two matrix inverses of order $(m_2\times m_2)$ and one matrix inverse of order $(m_1\times m_1)$. We now give an explicit statement of our nonlinear LSTSVM algorithm.

Algorithm 4.2. Nonlinear least squares twin SVM

Given m_1 datapoints in $\Re^n$ of class +1 represented by matrix A, and m_2 datapoints in $\Re^n$ of class −1 represented by matrix B nonlinear LSTSVM can be obtained using the following steps:

(i) Choose a kernel function K.
(ii) Define $G=[K(A,C') \quad e]$ and $H=[K(B,C') \quad e]$.
(iii) Select penalty parameters C_1 and C_2. Usually these parameters are selected based on validation.
(iv) *If* $m_1 < m_2$:
determine the parameters of two hypersurfaces using (34)–(36).
Else
determine the parameters of two hypersurfaces using (37)–(39).

(v) Calculate perpendicular distances $|K(x', C')u^{(1)} + \gamma^{(1)}|$ and $|K(x', C')u^{(2)} + \gamma^{(2)}|$ for a new datapoint $x \in \Re^n$.
(vi) Assign the datapoint to class +1 or −1 based on which of the distance $|K(x',C')u^{(1)} + \gamma^{(1)}|$ or $|K(x',C')u^{(2)} + \gamma^{(2)}|$ is minimum.

5. Experimental results on standard datasets

Before introducing experimental results on standard datasets we will present a simple two dimensional "Cross Planes" example, which shows the effectiveness of multi plane/surface classifiers like GEPSVM, TSVM and LSTSVM over PSVM. The "Cross Planes" dataset used in Mangasarian and Wild (2006) is generated by perturbing points lying on two intersecting lines. Fig. 3 shows the dataset and the linear classification results obtained by GEPSVM, TSVM, LSTSVM, and PSVM. While all multi plane classifiers obtained 100% accuracy, PSVM obtained only 83.2% accuracy.

To demonstrate the performance of LSTSVM we conducted experiments on a synthetic dataset and 11 datasets from the UCI Repository (Blake & Merz, 1998): Australian, CMC, heart-statlog, heart-c, hepatitis, ionosphere, liver, pima, sonar, WPBC, and votes. The synthetic dataset is an extension of two dimensional "Cross Planes" to $\Re^7$. We also conducted experiments on TSVM, GEPSVM and PSVM for comparison with LSTSVM on the same datasets. All algorithms were implemented in MATLAB 7.3.0 (R2006b) (http://www.mathworks.com, 2007) environment on a PC with Intel Core2Duo processor (2.13 GHz), 1 GB RAM. The dual QPPs arising in TSVM were solved using mosek optimization toolbox for MATLAB (http://www.mosek.com, 2007), which implements fast interior point based algorithms. Classification accuracy of each algorithm was measured by standard tenfold cross-validation methodology.

Table 1 shows the comparison of classification accuracy for LSTSVM with TSVM, GEPSVM and PSVM for linear kernel on 11 UCI and "Cross Planes" datasets. For each dataset, we estimated the generalized accuracy using best penalty parameter C of PSVM obtained through tuning in the range 2^{-7} to 2^{12}. For TSVM, GEPSVM and LSTSVM, a two-dimensional grid search on penalty parameters C_1 and C_2 was carried out in the same range. Table 2 shows the comparison of classification accuracy for the nonlinear extensions of the LSTSVM, TSVM, GEPSVM and PSVM on 11 UCI and "Cross Planes" datasets. We used Gaussian kernel for all experiments, given by $K(X_i, X_j) = e^{-\mu\|X_i - X_j\|^2}$. The kernel parameter μ along with penalty parameters were tuned for best classification accuracy. The kernel parameter μ was obtained through search from the range 2^{-20} to 2^4. Tables 1 and 2 show that the generalization capability of both LSTSVM and TSVM are better than GEPSVM and PSVM on many of the datasets considered. It also reveals that LSTSVM, whose solution is obtained by solving just two systems of linear equations, performs comparable to TSVM.

We also conducted experiments on large datasets, generated using David Musicant's NDC Data Generator (Musicant, 1998) to get a clear picture of how the computing time of all these algorithms scale with respect to number of datapoints. Table 3 gives a description of NDC datasets. For experiments with NDC datasets, we fixed penalty parameters of all algorithms to be one (i.e, $C = 1, C_1 = 1, C_2 = 1$). We used Gaussian kernel with $\mu = 2^{-17}$ for all experiments with nonlinear kernel. Table 4 shows the comparison of computing time and accuracy for LSTSVM, TSVM, GEPSVM and PSVM with linear kernel. For almost same accuracy, LSTSVM performed several orders of magnitude faster than TSVM on all datasets. It is also worth mentioning that LSTSVM does not require any special optimizers whereas TSVM has been implemented with fast interior point solvers of mosek optimization toolbox for MATLAB. While LSTSVM is fast, it is not as fast as PSVM which is evident from Table 4. This is obvious because in LSTSVM, we require solving two systems of linear equations whereas in PSVM we solve single system of linear equations. Table 5 shows comparison of computing time and accuracy of all four algorithms considered on several NDC datasets with nonlinear kernel. The results demonstrate that LSTSVM performs better than TSVM in terms of generalization. For NDC-2k and NDC-3k datasets, we used rectangular kernel (Fung & Mangasarian, 2001) using 10% of total datapoints. Results on these datasets show that LSTSVM, GEPSVM and PSVM algorithms are much faster than TSVM with reduced kernels. This is because even with reduced kernel of dimension $(l \times \bar{l})$, TSVM still requires solving two QPPs of same size m_1 and m_2.

6. Application to text categorization

Given the impressive performance of LSTSVM with linear kernel, we further went on to investigate its effectiveness for TC applications. Automatic TC is a supervised learning problem which involves training a classifier with some labeled documents and then using the classifier to predict the labels of unlabeled documents. Each document may belong to multiple labels or single label or no label at all. So, a binary classifier learns for each category to form a complete TC system. We performed experiments on 3 well-known datasets in TC research, reuters-21578 (Reuters-21578, 2007), ohsumed (Ohsumed, 2007) and 20 Newsgroups (20NG) (20 Newsgroups, 2004).

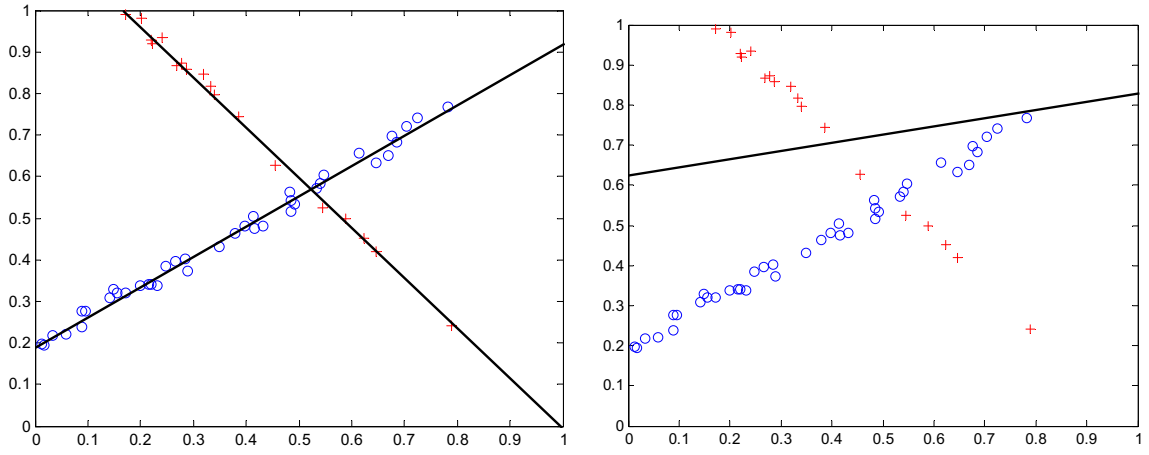

Fig. 3. Classification results of GEPSVM/TSVM/LSTSVM (left) and PSVM (right) for "cross planes" dataset.

Table 1
Classification accuracy for linear kernel

Dataset $l \times n$	LSTSVM	TSVM	GEPSVM	PSVM
Hepatitis (155 × 19)	86.42 ± 9.78	85.71 ± 6.73	85.0 ± 9.19	85.71 ± 5.83
WPBC (198 × 34)	83.88 ± 5.52	83.68 ± 6.24	81.11 ± 7.94	83.3 ± 4.53
Sonar (208 × 60)	80.47 ± 6.7	80.52 ± 4.9	79.47 ± 7.6	78.94 ± 4.43
Heart-statlog (270 × 14)	85.55 ± 4.07	86.66 ± 6.8	85.55 ± 6.1	85.55 ± 7.27
Cross Planes (300 × 7)	98.12 ± 4.67	98.02 ± 3.92	98.2 ± 5.1	60.71 ± 4.19
Heart-c (303 × 14)	85.86 ± 6.17	85.86 ± 6.9	85.51 ± 5.08	85.51 ± 5.08
Bupa Liver (345 × 7)	70.90 ± 6.09	70.5 ± 6.6	66.36 ± 4.39	70.15 ± 8.82
Ionosphere (351 × 34)	89.70 ± 5.58	88.23 ± 3.10	84.11 ± 3.2	89.11 ± 2.79
Votes (435 × 16)	95.23 ± 1.94	95.9 ± 2.2	95.0 ± 2.36	95.0 ± 3.06
Australian (690 × 14)	86.61 ± 4.0	86.91 ± 3.5	80.00 ± 3.99	85.43 ± 3.0
Pima-Indian (768 × 8)	79.4 ± 2.65	78 ± 6.29	76.66 ± 4.62	77.86 ± 3.67
CMC (1473 × 9)	68.84 ± 2.77	68.84 ± 2.39	68.76 ± 2.98	68.98 ± 3.95
Mean accuracy	84.24	84.06	82.14	80.52

Table 2
Classification accuracy for gaussian kernel

Dataset $l \times n$	LSTSVM	TSVM	GEPSVM	PSVM
Hepatitis (155 × 19)	84.28 ± 10.24	83.73 ± 6.25	79.28 ± 5.2	78.57 ± 0.24
WPBC (198 × 34)	81.66 ± 6.95	82.22 ± 6.82	80 ± 5.97	80.55 ± 3.92
Sonar (208 × 60)	90.52 ± 7.36	90.0 ± 7.5	80.0 ± 5.97	90.0 ± 7.21
Heart-statlog (270 × 14)	85.18 ± 5.23	85.84 ± 6.52	86.52 ± 7.36	70.74 ± 6.86
Cross Planes (300 × 7)	98.97 ± 3.7	98.6 ± 2.6	99.02 ± 4.16	83.93 ± 2.8
Heart-c (303 × 14)	83.79 ± 5.87	82.17 ± 5.21	70.37 ± 8.90	70.68 ± 7.66
Bupa Liver (345 × 7)	74.84 ± 6.85	75.15 ± 6.51	68.18 ± 6.2	74.84 ± 9.04
Ionosphere (351 × 34)	96.17 ± 3.68	96.17 ± 3.9	84.41 ± 6.20	95.0 ± 4.17
Votes (435 × 16)	96.19 ± 2.79	95.95 ± 3.37	94.5 ± 3.37	95.95 ± 2.25
Australian (690 × 14)	76.17 ± 5.36	75.8 ± 4.91	69.55 ± 5.37	73.97 ± 6.16
Pima-Indian (768 × 8)	75.33 ± 4.67	75.74 ± 5.2	75.33 ± 4.91	76.8 ± 3.83
CMC (1473 × 9)	74.42 ± 2.55	73.95 ± 3.48	68.62 ± 2.64	73.91 ± 4.12
Mean accuracy	84.79	84.61	79.64	80.42

Table 3
Description of NDC datasets

Dataset	#Training data	#Test data	#Features
NDC-500	500	50	32
NDC-700	700	70	32
NDC-900	900	90	32
NDC-1k	1000	100	32
NDC-2k	2000	200	32
NDC-3k	3000	300	32
NDC-4k	4000	400	32
NDC-5k	5000	500	32
NDC-10k	10,000	1000	32
NDC-1l	100,000	10,000	32
NDC-3l	300,000	30,000	32
NDC-5l	500,000	50,000	32
NDC-1m	1,000,000	100,000	32

Table 4
Comparison on NDC datasets with linear kernel

Dataset	LSTSVM Train % Test % Time (s)	TSVM Train % Test % Time (s)	GEPSVM Train % Test % Time (s)	PSVM Train % Test % Time (s)
NDC-3k	79.4 75.33 0.009	78.73 74.66 27.12	77.6 77.33 0.017	79.4 75.33 0.009
NDC-4k	79.62 73.75 0.011	79.67 73.75 60.86	76.9 71.0 0.021	79.62 73.75 0.01
NDC-5k	78.98 80 0.013	78.78 79.8 116.50	75.0 74.4 0.0224	78.98 80 0.012
NDC-10k	86.26 87 0.022	86.36 87.3 1094.19	84.76 83.9 0.0353	86.27 87 0.02
NDC-1l	86.15 85.94 0.178	[a]	83.90 84.32 0.26	86.15 85.94 0.127
NDC-3l	78.71 78.53 0.51	[a]	75.66 75.34 0.78	78.71 78.53 0.376
NDC-5l	78.67 78.64 0.86	[a]	75.64 75.62 1.32	78.67 78.64 0.627
NDC-1m	86.06 86.24 2.30	[a]	84.13 84.12 3.74	86.06 86.24 1.63

[a] We stopped experiments as computing time was very high.

Table 5
Comparison on NDC datasets with gaussian kernel nonlinear kernel

Dataset	LSTSVM Train % Test % Time (s)	TSVM Train % Test % Time (s)	GEPSVM Train % Test % Time (s)	PSVM Train % Test % Time (s)
NDC-500	100 82 0.106	99 80 0.721	82.0 76.0 8.25	95.4 82 0.073
NDC-700	100 82.85 0.256	99.42 84.28 1.609	80.71 71.42 24.58	96.71 85.71 0.174
NDC-900	100 87.77 0.488	98.88 80 3.11	83.33 66.66 52.12	95.66 81.11 0.329
NDC-1k	100 88 0.636	98.1 83 3.99	79.1 69.0 71.46	96.1 80 0.443
NDC-2k[a]	86.60 81 0.043	88.25 80.5 9.22	76.05 72.5 1.58	84.65 78.5 0.046
NDC-3k[a]	89.06 86.33 0.111	90.76 85 28.56	78.6 74 4.16	85.93 80.33 0.108

[a] A rectangular kernel (Fung & Mangasarian, 2001) $K(A,\overline{A})$ with $\overline{A}$ typically of size 10% of A was used.

6.1. Document representation

Documents which typically are string of characters have to be transformed into a representation suitable for the learning algorithm of classifier. This transformation involves several steps like preprocessing, dimensionality reduction, feature subset selection and term weighting. We used simple '*bag of words*' representation in all our experiments. We removed stop words using a stop word dictionary (Stop words, 2004) consisting of 319 words to reduce the dimension. In addition to stop words, we removed words that occurred in only one training document, uniformly for all three text corpuses considered. We also performed word stemming using Porter stemmer algorithm (Porter, 1980). After dimensionality reduction we did local feature selection using mutual information (MI) measure between a feature f and category c described in Dumais, Platt, Heckerman, and Sahami (1998), given by:

$$MI(f,c) = \sum_{f\in\{0,1\}} \sum_{c\in\{0,1\}} p(f,c) \log \frac{p(f,c)}{p(f)p(c)} \tag{40}$$

The selected features were associated with a weight using log(T-FIDF) (Liao, Alpha, & Dixon, 2007) term weighting scheme described as

$$w_{fd} = \log(tf_{fd} + 0.5)\log\frac{D}{df_f} \tag{41}$$

where w_{fd} is the weight of feature f in document d, tf_{fd} is the occurrence frequency of feature f in document d, D is the total number of documents in the training set and df_f is the number of documents containing the feature f. In all our experiments, we scaled the weights obtained from log(TFIDF) weighting using cosine normalization, given as

$$w^n_{fd} = \frac{w_{fd}}{\sqrt{\sum_{f=1}^{k} w^2_{fd}}}. \tag{42}$$

where k is the number of features selected to represent a document.

6.2. Data collections

6.2.1. Reuters-21578

The reuters-21578 (Reuters-21578, 2007) dataset was compiled by David Lewis and originally collected by the Carnegie group from the reuters newswire in 1987. It contains 21578 news articles each belonging to one or more categories. The frequency of occurrence of documents varies greatly from category to category. We used the mode Apt split which led us to a corpus of 9603 training and 3299 testing documents. Out of the 135 potential categories, only 90 categories have at least one training and one testing document. In our experiments, following other TC projects, we ran a test on the top 10 categories having highest number of documents. After stemming and stop word removal, the training corpus contains 10,789 distinct terms in the global dictionary. We evaluated MI measure for all these 10,789 distinct terms with respect to each category and selected the top 300 words as features for document representation of the corresponding category.

6.2.2. Ohsumed

The ohsumed corpus compiled by William hersh (Ohsumed, 2007) consists of medline documents from the year 1981 to 1991. Following (Joachims, 1998), from the 50,216 documents in 1991, we used first 10,000 for training and second 10,000 for testing. The resulting training set and testing set have more homogeneous distribution across 23 different MeSH "*diseases*" categories. Unlike reuters-21578, it is more difficult to learn a classifier in ohsumed corpus because of the presence of noisy data. We used top 10 out of 23 categories in our experiments. The training corpus of ohsumed contained 12,180 distinct terms after stemming and stop word removal. We evaluated MI measure for all 12,180 terms with respect to each category and selected top 500 words as features for further document representation of the corresponding category.

6.2.3. 20 Newsgroups

The 20NG corpus was first collected as a text corpus by Lang (20 Newsgroups, 2004). It almost contains 20,000 articles taken from the Usenet newsgroups evenly distributed across 20 categories. This corpus is different from the previous corpuses because it includes large vocabulary and words that have more meaning. Unlike other corpuses that we have considered, 20NG does not have standard training and testing sets. So we randomly selected 67% of the total documents in each category for training set and the rest we used for testing. We ran all our experiments on top 10 categories of 20NG. 20NG contained 27,159 distinct terms after stemming and stop word removal, out of which we selected top 500 using MI measure for document representation.

6.3. Evaluation methodology

A number of metrics are being used in TC to measure its effectiveness. In this paper, we used the standard information retrieval metrics, precision and recall, to evaluate each binary classifier. They can be calculated from the confusion matrix as shown in Table 6. A confusion matrix provides counts of different outcomes from an evaluation system. True positive (TP) represents the number of documents the system correctly labeled as positive and true negative (TN) represents the number of documents the system correctly labeled as negative. False positive (FP) and false negative (FN) are the number of documents the system incorrectly labeled as positive or negative respectively. Precision is defined simply as the ratio of correctly assigned category C_j documents to the total number of documents classified as category C_j. Recall is the ratio of correctly assigned category C_j documents to the total number of documents actually in category C_j. They can be obtained from the confusion matrix as

$$\text{Precision} = \frac{TP}{TP + FP} \text{ and Recall} = \frac{TP}{TP + FN} \tag{43}$$

Neither precision nor recall is meaningful in isolation of the other. In practice, combined effectiveness measure namely precision-recall breakeven point (BEP), and F_1 measure are used. F_1 measure is calculated as the harmonic mean of precision (P) and recall (R) given as

$$F_1 = 2PR/(P + R) \tag{44}$$

The precision-recall BEP is the point where precision is equal to recall and is often determined by calculating the arithmetic mean of precision and recall. BEP performance metric is to be computed for each category separately and the overall performance of an approach can be found with the help of microaverage or macroaverage of BEP over all categories. Macroaverage gives equal weight to each category, while microaverage gives equal weight to each document.

6.4. Experimental results

All text categorization steps discussed in Section 6.1 were performed using MATLAB 7.3.0 (R2006b) software (http://www.mathworks.com, 2007). We conducted experiments on the three text corpuses described in Section 6.2 using LSTSVM algorithm with linear kernel implemented in MATLAB. We also conducted experiments on same datasets using linear PSVM for comparison. However we did not conduct experiments using TSVM as its

Table 6
Confusion matrix

Category C_j		Expert judgments	
		YES	NO
Classifier output	YES	TP	FP
	NO	FN	TN

Table 7
BEP performance of 10 largest categories from reuters-21578

	LSTSVM	PSVM
Earn	0.9876	0.9848
Acq	0.9573	0.9515
Money-fx	0.8326	0.7868
Grain	0.9284	0.9386
Crude	0.8464	0.8599
Trade	0.7595	0.7797
Interest	0.7921	0.6908
Ship	0.7672	0.7339
Wheat	0.843	0.846
Corn	0.8477	0.8817
Micro avg. BEP	0.9202	0.9176
Macro avg. BEP	0.8562	0.8454

Table 8
BEP performance of 10 largest categories from ohsumed

	LSTSVM	PSVM
Pathology	0.542	0.5123
Neoplasma	0.8239	0.8085
Cardiovascular	0.8021	0.7913
Nervous	0.6404	0.6206
Environment	0.6736	0.6819
Digestion	0.7288	0.7185
Immunology	0.7413	0.7277
Respiratory	0.6578	0.6751
Urology	0.7724	0.7953
Bacteria	0.6816	0.6747
Micro avg. BEP	0.6939	0.6886
Macro avg. BEP	0.7064	0.7006

Table 9
BEP performance of 10 largest categories from 20 newsgroups

	LSTSVM	PSVM
Hockey	0.9259	0.9124
Christian	0.762	0.7353
Motor cycles	0.9131	0.8958
Baseball	0.913	0.8779
Crypt	0.9412	0.9302
Autos	0.8431	0.8106
Med	0.8556	0.8423
Space	0.8923	0.8822
Windows misc	0.7214	0.6966
Hardware	0.6252	0.6065
Micro avg. BEP	0.823	0.822
Macro avg. BEP	0.8393	0.8190

Table 10
F_1 performance of LSTSVM and PSVM

	Reuters-21578	Ohsumed	20NG
LSTSVM	0.8512	0.7041	0.8320
PSVM	0.8348	0.6810	0.8034

computing time is high. Similar to previous sections the penalty parameters of linear PSVM and LSTSVM were tuned to maximize micro BEP. Tables 7–9 summarize the BEP of top 10 categories obtained by LSTSVM and PSVM on reuters-21578, ohsumed and 20NG corpuses, respectively. It can be observed from these tables that LSTSVM achieves better BEP performance than PSVM on many of the categories over the three text corpuses. Particularly on 20NG corpus, LSTSVM has performed better than PSVM on all the ten categories considered. Table 10 shows the average F_1 performance obtained by both algorithms. Thus LSTSVM achieves improved performance on all the three text corpuses considered in terms of both BEP and F_1 performance measures. However this improved performance is obtained at the cost of more tuning effort involved. This is because LSTSVM requires tuning of two parameters whereas, PSVM has only single parameter.

7. Conclusion and future work

In this paper we have enhanced TSVM to least squares TSVM (LSTSVM). LSTSVM is an extremely simple algorithm for generating linear/nonlinear binary classifiers using two non-parallel hyperplanes/hypersurfaces. In LSTSVM, we solve the two primal problems of TSVM using proximal SVM (PSVM) idea instead of two dual problems usually solved in TSVM. LSTSVM requires just the solution of two systems of linear equations for both linear and nonlinear cases in contrast to TSVM, which requires solving two quadratic programming problems (QPPs) in addition to two systems of linear equations. For a linear classifier, LSTSVM requires inversion of two matrices of the order of input space dimension. For a nonlinear classifier, LSTSVM requires inversion of three matrices of order less than the number of datapoints in the dataset. LSTSVM takes advantage of reduced kernel techniques which leads to much faster training of a nonlinear classifier. This allows LSTSVM to easily classify large datasets for which TSVM requires very high training times. Our computational results on UCI and NDC datasets demonstrate that LSTSVM obtains classification accuracy comparable to that of TSVM, however, at reduced computational effort for both linear and nonlinear kernels. Linear LSTSVM easily classifies a large 1 million datapoint 32-attribute dataset in 2.3 s. Also LSTSVM obviates the need for any external optimizers as required by TSVM for both linear and nonlinear classifiers. On both linear and nonlinear classifiers, LSTSVM has better generalization than PSVM.

We further investigated the application of linear LSTSVM to text categorization using three benchmark text categorization datasets: reuters-21578, ohsumed and 20 Newsgroups (20NG). Comparison of experimental results against linear PSVM shows that linear LSTSVM has better generalization on all the three text corpuses considered. Since both linear LSTSVM and linear PSVM require solution of system of linear equations of the order of input space dimension, the dimensionality reduction and feature selection steps discussed in Section 6.1 are vital. Thus the performance of LSTSVM and PSVM on text categorization can greatly be improved by using it along with advanced feature selection/extraction methods which can bring down the dimension of input document vectors and will be the subject of our future work.

Acknowledgements

The authors are extremely thankful to Prof. S. Chandra for his valuable comments and suggestions. Also, they acknowledge the help of Mr. K.R. Shivaram in text categorization.

References

Blake, C. L. & Merz, C. J. (1998). *UCI repository for machine learning databases*. Irvine: Department of Information and Computer Sciences, University of California. <http://www.ics.uci.edu/~mlearn/MLRepository.html>.

Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery, 2*, 1–43.

Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In *Proceedings of seventh international conference on information and knowledge management*.

Fung, G., & Mangasarian, O. L. (2001). Proximal support vector machine classifiers. In *Proceedings of seventh international conference on knowledge and data discovery* (pp. 77–86).

Golub, G. H., & Van Loan, C. F. (1996). *Matrix computations* (3rd ed.). The John Hopkins University Press.

<http://www.mathworks.com>, 2007.

<http://www.mosek.com>, 2007.

Jayadeva, Khemchandani, R., & Chandra, S. (2007). Twin support vector machines for pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 29*(5), 905–910.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of ECML-98, 10th European conference on machine learning* (pp. 137–142).

Liao, C., Alpha, S., & Dixon, P. (2007). Feature preparation in text categorization, Technical Report, Oracle Corporation. Available at: <http://www.oracle.com/technology/products/text/pdf/feature_preparation.pdf>.

Mangasarian, O. L. (1998). *Nonlinear programming*. SIAM.

Mangasarian, O. L., & Musicant, D. R. (2001). Lagrangian support vector machines. *Journal of Machine Learning Research, 1*, 161–177.

Mangasarian, O. L., & Wild, E. W. (2006). Multisurface proximal support vector classification via generalized eigenvalues. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 28*(1), 69–74.

Musicant, D. R. (1998). NDC: Normally distributed clustered datasets. <http://www.cs.wisc.edu/~musicant/data/ndc>.

20 Newsgroups. (2004). <http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.htm>.

Ohsumed. (2007). <ftp://medir.ohsu.edu/pub/ohsumed>.

Porter, M. (1980). An algorithm for suffix stripping. *Program (Automated Library and Information Systems), 14*(3), 130–137.
Reuters-21578. (2007). <http://www.daviddlewis.com/resources/testcollections/reuters21578/>.
Scholkopf, B., & Smola, A. (2002). *Learning with kernels*. Cambridge, MA: MIT Press.
Stop words. (2004). <http://www.dcs.gla.ac.uk/idom/ir_resources/linuistic_utils/>.
Suykens, J. A. K., & Vandewalle, J. (1999). Least squares support vector machines classifiers. *Neural Processing Letters, 9*(3), 293–300.