Subspace based least squares support vector machines for pattern classification

Kitamura, Takuya

Abe, Shigeo

Fukui, Kazuhiro

(Citation)
Neural Networks, 2009. IJCNN 2009. International Joint Conference on:1640-1646

(Issue Date)
2009-06

(Resource Type)
conference paper

(Version)
Version of Record

(URL)
https://hdl.handle.net/20.500.14094/90000931

Subspace Based Least Squares Support Vector Machines for Pattern Classification

Takuya Kitamura, Shigeo Abe, and Kazuhiro Fukui

***Abstract*— In this paper, we discuss subspace based least squares support vector machines (SSLS-SVMs), in which an input vector is classified into the class with the maximum similarity. Namely, we define the similarity measure for each class by the weighted sum of vectors called dictionaries and optimize the weights so that the margin between classes is optimized. Because the similarity measure is defined for each class, the similarity measure associated with a data sample needs to be the largest among all the similarity measures. Introducing slack variables we define these constraints by equality constraints. Then the proposed SSLS-SVMs is similar to LS-SVMs by all-at-once formulation. Because all-at-once formulation is inefficient, we also propose SSLS-SVMs by one-against-all formulation. We demonstrate the effectiveness of the proposed methods with the conventional method for two-class problems.**

I. Introduction

In subspace methods [1], [2] each class region is defined by a set of basis vectors and the similarity of an input vector to a class is measured by the length of projection of the input onto the associated subspace. Similarities of the input vectors serve as discriminant functions.

Various subspace methods, such as class feature compression (CLAFIC) [1] and learning subspace methods [2], have been proposed. In most cases, principal component analysis (PCA) is used to compute basis vectors of subspaces. The basic idea of PCA is to rotate the coordinates so that data samples are non-correlated and delete the axes that do not contribute in representing the data distribution. To extend PCA for nonlinear problems, kernel PCA (KPCA) has been proposed [3], [4]. Recently, using KPCA variants of subspace methods are extended to kernel-based subspace methods, such as kernel mutual subspace methods (KMSMs) [5], [6], kernel constrained mutual subspace methods (KCMSMs) [7], and kernel orthogonal mutual subspace methods (KOSMSs) [6].

In subspace methods using KPCA, the values we assign to the weights in the similarity measure of each class are the eigenvalues or 1. However, because each subspace is defined separately and an overlap of subspaces or the margin between classes is not controlled after the definition of the subspaces, these weights may not be optimal from the standpoint of class separability.

In this paper, we propose subspace based least squares support vector machines (SSLS-SVMs), which optimize the weights in the similarity measure so that the margin between classes is maximized while minimizing the classification error for the training data. This is the same idea as that of support vector machines (SVMs) [8]. We consider the similarity measure as the separating hyperplane that separates the associated class from the remaining classes and formulate the optimization problem under the equality constraints that the similarity measure associated with a data sample has the highest similarity among all the similarity measures. This formulation is the same as all-at-once LS-SVMs, which are considered to be inefficient. However, because kernel evaluations are done when similarity measures are calculated, unlike regular SVMs kernel evaluations are not necessary during optimization. From the formulation, instead of quadratic programming programs, we derive a set of linear simultaneous equations. We call this method subspace based least squares SVMs, SSLS-SVMs for short. To speed up training SSLS-SVMs for large data sets, we propose formulating SSLS-SVMs by one-against-all formulation. By this formulation we can optimize the weights of each class, separately.

This paper is organized as follows. In Section II, we describe kernel-based subspace methods (KSMs) and how to calculate the similarity measures. In Section III, we propose SSLS-SVMs that optimize the weights in similarity measures. In Section IV, we demonstrate the effectiveness of SSLS-SVMs through computer experiments. And we conclude our work in Section V.

II. Subspace Methods

A. Kernel-based Subspace Methods

In kernel-based subspace methods, KPCA is used to represent a subspace of each class. Unlike conventional KPCA, in calculating the covariance matrix, the mean vector is not subtracted from the training data. We consider an n-class classification problem with the m-dimensional input vector $\mathbf{x}$. Let $\mathbf{x}$ be mapped into the l-dimensional feature space by the mapping function $\mathbf{g}(\mathbf{x})$. Thus, r_i dictionaries $\boldsymbol{\varphi}_{ik}$ for the subspace for class i are the eigenvectors of the following eigenvalue problem:

$$\frac{1}{|X_i|}\sum_{i\in X_i}\mathbf{g}(\mathbf{x}_j)\,\mathbf{g}^T(\mathbf{x}_j)\,\boldsymbol{\varphi}_{ik}=\lambda_{ik}\,\boldsymbol{\varphi}_{ik},\qquad(1)$$
$$\text{for}\quad i=1,...,n,\quad k=1,...,r_i,$$

where X_i is the index set for class i training data, $|X_i|$ is the number of elements in X_i, and λ_{ik} is the eigenvalue associated with $\boldsymbol{\varphi}_{ik}$. Here, we assume that we select the first to the r_ith largest eigenvalues in (1).

Takuya Kitamura and Shigeo Abe are with Graduate School of Engineering, Kobe University, Kobe, Japan (email: 084t219t@stu.kobe-u.ac.jp, abe@kobe-u.ac.jp). Kazuhiro Fukui is with Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Japan (email: kfukui@cs.tsukuba.ac.jp).

978-1-4244-3553-1/09/$25.00 ©2009 IEEE

To calculate (1) without using the variables in the feature space, we need to use kernel tricks. But it is time consuming. Thus to speed up calculations we use the concept of the empirical feature space [9], [10]. The empirical feature space is spanned by the mapped training data and gives the same kernel value as that of the feature space.

Let the kernel be $H(\mathbf{x}, \mathbf{x}') = \mathbf{g}^T(\mathbf{x})\,\mathbf{g}(\mathbf{x}')$, and the number of data be M. For the M m-dimensional data $\mathbf{x}_i$ $(i = 1, \ldots, M)$, the $M \times M$ kernel matrix $H = \{H(\mathbf{x}_j, \mathbf{x}_k)\}\,(j, k = 1, ..., M)$ is symmetric and positive semidefinite. Let the rank of H be $N(\leq M)$. Then H is expressed by

$$H = USU^T, \tag{2}$$

where the column vectors of U are eigenvectors of H and S is given by

$$S = \begin{pmatrix} \sigma_1 & & 0 & \\ & \ddots & & 0_{N\times(M-N)} \\ 0 & & \sigma_N & \\ & 0_{(M-N)\times N} & & 0_{(M-N)\times(M-N)} \end{pmatrix}. \tag{3}$$

Here, $\sigma_j (> 0)$ are eigenvalues of H, whose eigenvectors correspond to the jth columns of U, and for instance $0_{(M-N)\times(M-N)}$ is the $(M-N)\times(M-N)$ zero matrix.

Defining the first N vectors of U as the $M \times N$ matrix P and Λ as the $N \times N$ matrix whose diagonal elements are $\sigma_j (j = 1, ...N)$, we can rewrite (2) as follows:

$$H = P\Lambda P^T, \tag{4}$$

where $P^T P = I_{N\times N}$ but $PP^T \neq I_{M\times M}$, and I is the unit matrix.

The mapping function to the N-dimensional empirical feature space is given by

$$\mathbf{h}(\mathbf{x}) = \Lambda^{1/2} P^T \left(H(\mathbf{x}_1, \mathbf{x}), ..., H(\mathbf{x}_M, \mathbf{x})\right)^T \quad \text{for} \quad j = 1, ..., N. \tag{5}$$

It is proved that the empirical feature space gives the same kernel value as that of the feature space. Therefore, without inducing any error, instead of (1), we can carry out KPCA by

$$\frac{1}{|X_i|} \sum_{i \in X_i} \mathbf{h}^T(\mathbf{x}_j)\,\mathbf{h}(\mathbf{x}_j)\boldsymbol{\varphi}_{ik} = \lambda_{ik}\,\boldsymbol{\varphi}_{ik}^T. \tag{6}$$

Although the dimension of the coefficient matrix on the left-hand side of (1) may be infinite for RBF kernels, that of the (6) is finite, i.e., N. Calculations of the eigenvalues and eigenvectors in (5) are not necessary if we obtain the M linearly independent data that span the empirical feature space. This can be done by the Cholesky factorization of the kernel matrix H deleting the linearly dependent data. Then, instead of (5) we use

$$\mathbf{h}(\mathbf{x}) = \left(H(\mathbf{x}_{k_1}, \mathbf{x}), ..., H(\mathbf{x}_{k_N}, \mathbf{x})\right)^T \tag{7}$$

where $\mathbf{x}_{k_j}\,(j = 1, ..., N)$ are linearly independent in the feature space.

For the N eigenvalues obtained by solving (5), we select, for the class i dictionaries, the first r_i largest eigenvalues and the associated eigenvectors according to the accumulation of eigenvalues. A more popular method is that of selecting different dimensions for the various classes based on the cumulative proportion, a, which is defined as follows:

$$a(r_i) = \frac{\sum_{j=1}^{r_i} \lambda_{ij}}{\sum_{j=1}^{N} \lambda_{ij}} \times 100\,(\%). \tag{8}$$

We set a threshold κ and determine r_i so that $a(r_i - 1) < \kappa \leq a(r_i)$.

B. Similarity Measures

We consider an n-class classification problem with the m-dimensional input vector $\mathbf{x}$ by kernel subspace methods in the empirical feature space. Let $\mathbf{x}$ be mapped into the r_i-dimensional subspace for class i in the N-dimensional empirical feature space mapped by $\mathbf{h}(\mathbf{x})$ and the kth dictionary for class i be $\boldsymbol{\varphi}_{ik}\,(k = 1, ..., r_i)$. Then the similarity measure is given by

$$S_i(\mathbf{x}) = \sum_{k=1}^{r_i} \frac{w_{ik}\,(\boldsymbol{\varphi}_{ik}^T\,\mathbf{h}(\mathbf{x}))^2}{\|\boldsymbol{\varphi}_{ik}\|^2\,\|\mathbf{h}(\mathbf{x})\|^2}, \tag{9}$$

where w_{ik} is the weight for the kth dictionary of class i.

Defining

$$\mathbf{f}_i(\mathbf{x}) = \left(\frac{(\boldsymbol{\varphi}_{i1}^T\,\mathbf{h}(\mathbf{x}))^2}{\|\boldsymbol{\varphi}_{i1}\|^2\,\|\mathbf{h}(\mathbf{x})\|^2}, ..., \frac{(\boldsymbol{\varphi}_{ir_i}^T\,\mathbf{h}(\mathbf{x}))^2}{\|\boldsymbol{\varphi}_{ir_i}\|^2\,\|\mathbf{h}(\mathbf{x})\|^2}\right)^T, \tag{10}$$

(9) becomes

$$S_i(\mathbf{x}) = \mathbf{w}_i^T \mathbf{f}_i(\mathbf{x}), \tag{11}$$

where $\mathbf{w}_i = (w_{i1}, ..., w_{ir_i})^T$.

Input vector $\mathbf{x}$ is classified into class

$$\arg \max_{i=1,...,n} S_i(\mathbf{x}). \tag{12}$$

III. Subspace Based Least Squares Support Vector Machines

A. Idea

In conventional subspace methods using KPCA, we set the eigenvalues or 1 to the weights of the similarity measure of each class. However, these values are not optimal from the standpoint of class separability, because weights are not determined to make class separability as large as possible.

We propose subspace based least squares support vector machines (SSLS-SVMs) to solve this problem. We determine the values using the idea of least squares support vector machines. Because (11) can be viewed as a decision function without a bias term we can borrow the idea of SVMs, namely maximizing margins between classes. But, unlike the decision functions of conventional SVMs, (11) is defined for each class. Thus we need to formulate SSLS-SVMs so that for a data sample the similarity measure for the associated class is the largest. This leads to formulating SSLS-SVMs by all-at-once formulation. To alleviate the computational burden of all-at-once formulation we also formulate SSLS-SVMs by one-against-all formulation.

B. Subspace Based LS-SVMs by All-at-Once Formulation

Equation (11) can be viewed as the separating hyperplane in the class i subspace given by $\mathbf{f}_i(\mathbf{x})$, and it separates class i data from those belonging to other classes. Thus, we can maximize the margin in the subspace by minimizing $\|\mathbf{w}_i\|$. But, unlike the decision functions of conventional SVMs, each class has its decision function. Thus we need to formulate subspace based support vector machines according to all-at-once formulation.

Let for the n class problem M training data pairs be $\{(\mathbf{x}_1, y_1),...,(\mathbf{x}_M, y_M)\}$, where $\mathbf{x}_i$ and y_i are the m-dimensional input vectors and the associated class labels, respectively, and $y_i \in \{1, ..., n\}$. We formulate the SSLS-SVM with all-at once formulation as follows:

$$\begin{aligned} \text{minimize} \quad & Q(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\sum_{i=1}^{n}\|\mathbf{w}_i\|^2 \\ & + \sum_{i=1}^{n}\sum_{\substack{y_j \neq i,\\ j=1}}^{M} \frac{CM}{2n\,|X_{y_j}|}\xi_{ij}^2 \qquad (13) \\ \text{subject to} \quad & \mathbf{w}_{y_j}^T \mathbf{f}_{y_j}(\mathbf{x}_j) - \mathbf{w}_i^T \mathbf{f}_i(\mathbf{x}_j) = 1 - \xi_{ji} \\ & \text{for} \quad i \neq y_j, i = 1, ..., n, j = 1, ..., M, \qquad (14) \end{aligned}$$

where C is the margin parameter that determines the tradeoff between maximizing margins and minimizing misclassifications and ξ_{ij} are non-negative slack variables. The term $C\,M/(n\,|X_{y_j}|)$ in (13) is to avoid biased penalties for the unbalanced class data. In kernel subspace methods, the weights are assumed to be non-negative, but here we do not impose non-negativeness to increase freedom of solutions.

We solve the above optimization problem in the primal form. Substituting (14) into (13), we obtain

$$\begin{aligned} Q(\mathbf{w}, \xi) \quad = \quad & \frac{1}{2}\sum_{i=1}^{n}\|\mathbf{w}_i\|^2 + \sum_{i=1}^{n}\sum_{\substack{y_j \neq i,\\ j=1}}^{M} \frac{CM}{2n\,|X_{y_j}|} \\ & \times (1 - (\mathbf{w}_{y_j}^T \mathbf{f}_{y_j}(\mathbf{x}_j) - \mathbf{w}_i^T \mathbf{f}_i(\mathbf{x}_j)))^2. \qquad (15) \end{aligned}$$

Taking the partial derivative of (15) with respect to $\mathbf{w}_i$ and setting the resulting equation to $\mathbf{0}$, we obtain

$$\mathbf{w} \quad = \quad \begin{pmatrix} A_1 & B_{12} & \cdots & B_{1n} \\ B_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ B_{n1} & \cdots & \cdots & A_n \end{pmatrix}^{-1} \mathbf{a}, \qquad (16)$$

where $\mathbf{w} = (\mathbf{w}_1, ..., \mathbf{w}_n)^T$, $\mathbf{a} = (\mathbf{a}_1, ..., \mathbf{a}_n)^T$, and

$$\begin{aligned} A_i \quad = \quad & \frac{n}{MC} I_{r_i} + \sum_{j=1, y_j \neq i}^{M} \frac{1}{|X_{y_j}|} \mathbf{f}_i(\mathbf{x}_j)\,\mathbf{f}_i^T(\mathbf{x}_j) \\ & \text{for} \quad i = 1, \ldots, n, \qquad (17) \\ B_{iy_j} \quad = \quad & -\sum_{j=1, y_j \neq i}^{M} \frac{1}{|X_{y_j}|} \mathbf{f}_i(\mathbf{x}_j)\,\mathbf{f}_{y_j}^T(\mathbf{x}_j) \\ & \text{for} \quad i \neq y_j, i = 1, \ldots, n, \qquad (18) \\ \mathbf{a}_i \quad = \quad & (n-1) \sum_{j=1, y_j = i}^{M} \frac{1}{|X_{y_j}|} \mathbf{f}_i(\mathbf{x}_j) \\ & - \sum_{j=1, y_j \neq i}^{M} \frac{1}{|X_{y_j}|} \mathbf{f}_i(\mathbf{x}_j) \quad \text{for } i = 1, \ldots, n. \qquad (19) \end{aligned}$$

The size of the matrix that needs to be solved in (16) is $\sum_i^n r_i$. Thus, as the number of classes or the number or dictionaries increases, training becomes slow.

In our study we use RBF kernels: $\exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$ where γ is the width of the radius. Then, before training SSLS-SVMs, we need to determine the γ value, and the threshold value of the cumulative proportion, κ, and the value of margin parameter C. To compare between the proposed method and conventional kernel subspace methods clear, first we determine γ and κ values for kernel subspace methods by fivefold cross-validation. Then, using these values, we optimize the C value for SSLS-SVMs by fivefold cross-validation. The algorithm of training SSLS-SVMs by all-at-once formulation is as follows:

Algorithm 1

Step 1 Determine the γ and κ values for kernel subspace methods with equal weights or weights equal to eigenvalues by fivefold cross-validation. For the determined subspaces, determine the C value for the SSLS-SVM by fivefold cross-validation.

Step 2 Using the parameter values determined in Step 1, select the linearly independent data from the training data by the Cholesky factorization.

Step 3 Generate the mapping function to the empirical feature space using the linearly independent data obtained in Step 2. Calculate eigenvectors $\boldsymbol{\varphi}_{ik}$ and eigenvalues λ_{ik} for class i $(i = 1, \ldots, n)$ using (6).

Step 4 Determine the dimension of the subspace for class i, r_i, using the κ value determined in Step 1.

Step 5 Calculate $\mathbf{f}_i(\mathbf{x}_j)$ for $i = 1, \ldots, n$, $j = 1, \ldots, M$.

Step 6 Calculate weights $\mathbf{w}$ using (16).

C. Subspace Based LS-SVMs by One-against-All Formulation

We can optimize the weights in the similarity measures by the SSLS-SVM by all-at-once formulation. But if the size of the matrix in (16) is large, it will be difficult to solve the SSLS-SVM. Therefore, to speed up training in

such a situation, we consider formulating SSLS-SVMs in one-against-all formulation, in which we separately optimize the weights of the similarity measure of each class.

To improve the classification ability which may be decreased because of approximation introduced by one-against-all formulation, instead of (11), we use the following decision function which include the class i bias term b_i:

$$S_i(\mathbf{x}) = \mathbf{w}_i \mathbf{f}_i(\mathbf{x}) + b_i. \tag{20}$$

Then for class $i\,(i = 1, \ldots, n)$ we formulate the SSLS-SVM with one-against-all formulation as follows:

$$\text{minimize} \quad Q(\mathbf{w}, \mathbf{b}, \boldsymbol{\xi}) = \frac{1}{2}||\mathbf{w}_i||^2 + \sum_{j=1}^{M} \frac{CM}{2n\,|X_{y_j}|}\xi_{ij}^2 \tag{21}$$

$$\text{subject to} \quad y_j(\mathbf{f}_i(\mathbf{x}_j) + b_i) = 1 - \xi_{ij} \quad \text{for} \;\; j = 1, ..., M. \tag{22}$$

Here, unlike ξ_{ij} in (13), ξ_{ij} in (21) is defined for $j = 1, \ldots, M$ and can be discarded when $\mathbf{w}_i$ is obtained. Thus, it may be possible to drop the subscript i in ξ_{ij}. But since they are different for different classes, we use ξ_{ij}.

We solve the above optimization problem in the primal form. Substituting (22) into (21), we obtain

$$Q(\mathbf{w}, \mathbf{b}) = \frac{1}{2}||\mathbf{w}_i||^2 + \sum_{j=1}^{M} \frac{C\,M}{2n\,|X_{y_j}|}(1 - y_j(\mathbf{f}_i(\mathbf{x}_j) + b_i))^2. \tag{23}$$

Taking the partial derivative of (23) with respect to $\mathbf{w}_i$ and b_i, and setting the resulting equation to $\mathbf{0}$, we obtain

$$\mathbf{w}_i = \Omega_i^{-1}\mathbf{a}_i', \tag{24}$$

$$b_i = \frac{1}{2}\left(\sum_{j=1}^{M} \frac{y_j}{|X_{y_j}|} - \mathbf{w}_i^T \sum_{j=1}^{M} \frac{1}{|X_{y_j}|}\mathbf{f}_i(\mathbf{x}_j)\right), \tag{25}$$

where

$$\Omega_i = \frac{n}{CM} I_{r_i} + \sum_{j=1}^{M} \frac{1}{|X_{y_j}|}\mathbf{f}_i(\mathbf{x}_j)\mathbf{f}_i^T(\mathbf{x}_j) \tag{26}$$

$$-\frac{1}{2}\sum_{j=1}^{M} \frac{1}{|X_{y_j}|}\mathbf{f}_i(\mathbf{x}_j) \sum_{j=1}^{M} \frac{1}{|X_{y_j}|}\mathbf{f}_i^T(\mathbf{x}_j), \tag{27}$$

$$\mathbf{a}_i' = \sum_{j=1}^{M} \frac{1}{|X_{y_j}|} y_j \mathbf{f}_i(\mathbf{x}_j) - \frac{1}{M}\sum_{j=1}^{M} \frac{1}{|X_{y_j}|} y_j \sum_{j=1}^{M} \frac{1}{|X_{y_j}|}\mathbf{f}_i(\mathbf{x}_j). \tag{28}$$

Since we calculate the $r_i \times r_i$ matrixes for $i = 1, \ldots, n$, the number of matrix operations is of the order of $\sum_i^n r_i^3$. While by all-at-once formulation the number of matrix operations is of the order of $(\sum_{i=1}^n r_i)^3$. Thus, by one-against-all formulation, the computational cost will be much cheaper. We call the SSLS-SVM by one-against-all formulation SSLS-SVM (O).

The algorithm of training SSLS-SVM (O) is as follows:

Algorithm 2

Step 1 Determine the γ and κ values for the kernel subspace methods by fivefold cross-validation. And determine the value of the margin parameter for the SSLS-SVM (O) by fivefold cross-validation.

Step 2 Using the parameter values determined in Step 1, select the linearly independent data from the training data by the Cholesky factorization. Set $i = 1$.

Step 3 Calculate eigenvectors $\boldsymbol{\varphi}_{ik}$ and eigenvalues λ_{ik} for $i = 1, \ldots, n$ by (6).

Step 4 Determine the dimension of class i subspace, r_i, using the κ value determined in Step 1.

Step 5 Calculate $\mathbf{f}_i(\mathbf{x}_j)$ for $j = 1, \ldots, M$.

Step 6 Calculate weight vector $\mathbf{w}_i$ and bias term b_i using (24) and (25), respectively.

Step 7 If $i \neq n$, we set $i = i + 1$ and go to Step 3. If $i = n$, terminate the algorithm.

IV. Experimental Results

A. Benchmark Data Sets

We compared the proposed SSLS-SVMs with LS-SVMs and the conventional kernel subspace methods (KSMs) with the weights equal to 1 or the eigenvalues, using the two-class benchmark data sets [11], [12] listed in Table I. The table shows the number of inputs, training data, test data, and training and test data sets. We used RBF kernels and assumed that the diagonal element in the Cholesky factorization is zero if the argument of the square root in the diagonal element is less than or equal to 10^{-5}.

TABLE I
TWO-CLASS BENCHMARK DATA SETS

Data	Inputs	Training	Test	Sets
Banana	2	400	4900	100
B. cancer	9	200	77	100
Diabetes	8	468	300	100
German	20	700	300	100
Heart	13	170	100	100
Image	18	1300	1010	20
Ringnorm	20	400	7000	100
F. solar	9	666	400	100
Splice	60	1000	2175	20
Thyroid	5	140	75	100
Titanic	3	150	2051	100
Twonorm	20	400	7000	100
Waveform	21	400	4600	100

B. Setting of the Parameter Values

To compare SSLS-SVMs with KSMs, we optimized the kernel parameter γ and the threshold of the cumulative proportion κ for KSMs by fivefold cross-validation. Then using the same values of γ and κ, we optimized the value of C for SSLS-SVMs. By this, we can check how optimizing the weights improves the generalization ability of KSMs.

We selected κ from $\kappa = \{80, 85, 90, 95, 99, 99.9\}$, γ from $\gamma = \{0.1, 0.5, 1, 1.5, 3, 5, 10, 15\}$, and C from $C =$

$\{0.1, 0.5, 1, 5, 10, 50, 100, 500, 10^3, 5 \times 10^3, 10^4, 5 \times 10^4, 10^5\}$ by fivefold cross-validation.

Table II shows the parameter values determined by the above procedure. In the table, KSM (1) and KSM (E) denote the conventional KSM with equal weights and with weights set by the eigenvalues, respectively. The γ values are the same for only three problems.

TABLE II
PARAMETER VALUES

Data	KSM (1)		KSM (E)	
	κ (%)	γ	κ (%)	γ
Banana	99.9	15	99.9	15
B. cancer	99.9	0.5	85	3
Diabetes	99.9	3	80	5
German	85	3	85	10
Heart	95	1.5	80	3
Image	99	15	99.9	15
Ringnorm	99.9	0.1	99.9	15
F. solar	80	5	95	10
Splice	99	1	99	15
Thyroid	85	15	80	15
Titanic	80	0.1	95	5
Twonorm	80	0.1	80	3
Waveform	80	1.5	80	5

Table III shows the margin parameter values C_1 and C_2 determined by fivefold cross-validation using the γ and κ values determined for KSM (1) and KSM (E), respectively. In the table, SSLS-SVM and SSLS-SVM (O) denote the SSLS-SVM by all-at-once formulation and the SSLS-SVM by one-against-all formulation.

TABLE III
PARAMETER VALUES

Data	SSLS-SVM		SSLS-SVM (O)	
	C_1	C_2	C_1	C_2
Banana	50	50	1	1
B. cancer	10	0.1	5	1
Diabetes	10^3	1	50	0.5
German	10^4	0.5	100	500
Heart	10	0.1	500	0.5
Image	10^5	10^5	500	10^3
Ringnorm	10^4	500	1	0.1
F. solar	10^4	500	1	10
Splice	10^4	5×10^3	10^3	0.1
Thyroid	500	500	5	5
Titanic	500	0.1	0.5	5
Twonorm	50	10	1	0.1
Waveform	50	5×10^3	1	0.1

C. Comparison with KSM (1)

Table IV shows the average recognition rates and their standard deviations of the validation data sets by KSM (1), SSLS-SVM, and SSLS-SVM (O) for the first five training data sets. For each problem, the best average recognition rate is shown in boldface. For seven data sets, SSLS-SVM performed better than KSM (1). And for five data sets, SSLS-SVM (O) performed better than KSM (1). For the ringnorm problem, by optimizing weights the average recognition rates were significantly improved. But for the splice problem, optimizing weights by SSLS-SVM (O) degraded the average recognition rate. Comparing SSLS-SVM and SSLS-SVM (O), except for the banana problem, the latter performed worse.

TABLE IV
AVERAGE RECOGNITION RATES (%) AND THEIR STANDARD DEVIATIONS OF VALIDATION SETS (EQUAL WEIGHTS)

Data	KSM (1)	SSLS-SVM	SSLS-SVM (O)
Banana	89.8 ± 3.1	89.7 ± 3.1	$\mathbf{90.1 \pm 3.2}$
B. cancer	72.6 ± 4.6	$\mathbf{74.4 \pm 4.3}$	72.8 ± 4.3
Diabetes	73.5 ± 2.6	$\mathbf{73.8 \pm 2.8}$	72.7 ± 3.9
German	$\mathbf{73.9 \pm 3.0}$	72.6 ± 3.3	68.4 ± 3.0
Heart	82.1 ± 3.9	$\mathbf{83.2 \pm 4.4}$	81.7 ± 5.4
Image	$\mathbf{95.7 \pm 0.9}$	94.8 ± 1.0	93.7 ± 1.2
Ringnorm	52.8 ± 1.8	$\mathbf{97.8 \pm 1.7}$	89.9 ± 2.5
F. solar	65.2 ± 2.8	$\mathbf{66.1 \pm 3.5}$	62.6 ± 4.6
Splice	$\mathbf{87.4 \pm 2.5}$	86.3 ± 2.1	57.2 ± 2.2
Thyroid	95.6 ± 2.5	$\mathbf{96.3 \pm 2.4}$	96.2 ± 2.0
Titanic	$\mathbf{79.6 \pm 7.8}$	79.0 ± 7.5	76.0 ± 5.9
Twonorm	97.1 ± 1.5	$\mathbf{97.4 \pm 1.4}$	97.3 ± 1.4
Waveform	$\mathbf{89.6 \pm 3.0}$	82.5 ± 3.2	81.9 ± 3.4

Table V shows the average recognition rates and their standard deviations of test data sets. It also includes the results for regular LS-SVMs. The best results among KSM (1), SSLS-SVM and SSLS-SVM (O) are shown in boldface. For seven problems, SSLS-SVM performed better than KSM (1). And for five problems, SSLS-SVM (O) performed better than KSM (1). Particularly, the average recognition rates of SSLS-SVM and SSLS-SVM (O) for the ringnorm data sets improved extremely. But the average recognition rates of SSLS-SVM (O) for the german and splice data sets were extremely bad. Generally, the average recognition rates of SSLS-SVM (O) were worse than those of SSLS-SVM.

Comparing Tables IV and V, the classifier that performed best for the validation data sets tends to perform best for the test data sets or if not, not so bad. Thus, from the evaluation of the validation data sets we can select the best or near best kernel subspace method in most cases.

Comparing subspace methods with LS-SVMs, the SSLS-SVM showed comparable performance for seven problems and both the KSM (1) and SSLS-SVM (O) did for four problems.

TABLE V
AVERAGE RECOGNITION RATES (%) AND THEIR STANDARD DEVIATIONS OF TEST DATA SETS (EQUAL WEIGHTS)

Data	LSSVM	KSM (1)	SSLS-SVM	SSLS-SVM (O)
Banana	89.4 ± 0.5	88.6 ± 0.6	$\mathbf{88.9 \pm 0.6}$	$\mathbf{88.9 \pm 0.6}$
B. cancer	74.0 ± 4.7	$\mathbf{75.0 \pm 4.3}$	73.8 ± 4.6	72.2 ± 4.6
Diabetes	76.9 ± 1.7	$\mathbf{73.5 \pm 1.8}$	72.4 ± 2.2	72.5 ± 1.9
German	76.4 ± 2.2	$\mathbf{75.1 \pm 2.2}$	74.0 ± 2.2	69.7 ± 2.8
Heart	83.8 ± 3.1	80.5 ± 3.3	82.6 ± 3.9	$\mathbf{82.7 \pm 3.5}$
Image	97.5 ± 0.3	$\mathbf{96.3 \pm 0.6}$	95.1 ± 0.6	94.5 ± 0.5
Ringnorm	96.3 ± 0.4	76.6 ± 11.2	$\mathbf{97.6 \pm 0.3}$	91.5 ± 0.8
F. solar	66.7 ± 1.6	65.1 ± 1.8	$\mathbf{66.9 \pm 1.6}$	62.4 ± 2.2
Splice	89.4 ± 0.7	$\mathbf{87.6 \pm 0.8}$	86.2 ± 1.0	57.4 ± 1.8
Thyroid	95.9 ± 2.1	95.6 ± 2.1	$\mathbf{95.9 \pm 2.1}$	95.5 ± 2.2
Titanic	77.3 ± 1.1	76.6 ± 1.2	$\mathbf{77.2 \pm 0.8}$	76.9 ± 0.9
Twonorm	97.4 ± 0.2	97.6 ± 0.1	$\mathbf{97.7 \pm 0.1}$	$\mathbf{97.7 \pm 0.1}$
Waveform	89.9 ± 0.5	$\mathbf{88.5 \pm 0.6}$	81.2 ± 0.9	81.7 ± 0.9

D. Comparison with KSM (E)

Table VI shows the average recognition rates and their standard deviations of the validation data sets generated by the first five training data sets by KSM (E), SSLS-SVM, and SSLS-SVM (O). For each problem, the best average recognition rate is shown in boldface. For seven data sets, both SSLS-SVM and SSLS-SVM(O) performed better than the KSM (E). The average recognition rate of KSM (E) for the image data set was extremely bad. Comparing Table IV and VI, KSM (1) and KSM (E) showed comparable performance, in that KSM (1) performed better than KSM (E) for seven problems. Also SSLS-SVM (O) showed comparable performance for the parameters determined by KSM (1) and KSM (E), but SSLS-SVM performed a little worse for the parameters determined by KSM (E).

TABLE VI
AVERAGE RECOGNITION RATES (%) AND THEIR STANDARD DEVIATIONS OF VALIDATION SETS (EIGENVALUES)

Data	KSM (E)	SSLS-SVM	SSLS-SVM (O)
Banana	88.3 ± 2.9	89.7 ± 3.1	$\mathbf{90.1 \pm 3.2}$
B. cancer	$\mathbf{75.7 \pm 3.5}$	75.1 ± 3.1	71.0 ± 4.6
Diabetes	$\mathbf{73.8 \pm 3.0}$	70.6 ± 3.2	70.8 ± 3.0
German	$\mathbf{71.6 \pm 2.8}$	71.0 ± 2.7	71.0 ± 3.0
Heart	82.9 ± 3.9	$\mathbf{84.3 \pm 4.7}$	82.5 ± 4.3
Image	88.1 ± 1.4	$\mathbf{94.9 \pm 1.2}$	94.1 ± 1.2
Ringnorm	62.5 ± 4.0	$\mathbf{63.2 \pm 3.3}$	$\mathbf{63.4 \pm 3.2}$
F. solar	64.4 ± 3.7	$\mathbf{65.1 \pm 3.8}$	$\mathbf{65.1 \pm 3.7}$
Splice	72.0 ± 3.0	72.1 ± 3.1	$\mathbf{72.4 \pm 2.9}$
Thyroid	$\mathbf{96.0 \pm 2.7}$	95.3 ± 2.5	$\mathbf{96.0 \pm 2.4}$
Titanic	$\mathbf{79.4 \pm 7.0}$	78.4 ± 8.2	78.3 ± 8.1
Twonorm	97.2 ± 1.4	97.2 ± 1.2	$\mathbf{97.3 \pm 1.2}$
Waveform	89.0 ± 2.8	$\mathbf{89.8 \pm 2.2}$	84.2 ± 3.5

Table VII shows the average recognition rates and their standard deviations of test data sets. It also includes the results for regular LS-SVMs. The best results among KSM (E), SSLS-SVM and SSLS-SVM (O) are shown in boldface. For seven problems, SSLS-SVM performed better than KSM (E). And for nine problems, SSLS-SVM (O) performed better than KSM (E). Particularly, the average recognition rates of SSLS-SVM and SSLS-SVM (O) for the image data sets were improved significantly. Comparing Tables VI and VII, the best classifier for the validation data sets tends to be the best or near best for the test data sets.

Comparing the subspace methods with the LS-SVM, SSLS-SVM showed comparable performance for five problems, SSSLS-SVM (O) did four problems, and KSME (E) did three problems.

Comparing the results shown in Tables V and VII, the classifier based on the parameters determined by KSM (1) showed comparable performance with that by KSM (E), but the former showed a slightly better performance. For some problems both classifiers showed very different performance, e.g., SSLS-SVMs for the ringnorm problem.

From the standpoint of computational burden, SSLS-SVM (O) is better than SSLS-SVM but the classification performance of SSLS-SVM (O) was slightly worse.

TABLE VII
AVERAGE RECOGNITION RATES (%) AND THEIR STANDARD DEVIATIONS OF TEST DATA SETS (EIGENVALUES)

Data	LSSVM	KSM (E)	SSLS-SVM	SSLS-SVM (O)
Banana	89.4 ± 0.5	87.8 ± 0.7	$\mathbf{88.9 \pm 0.6}$	$\mathbf{88.9 \pm 0.6}$
B. cancer	74.0 ± 4.7	$\mathbf{75.1 \pm 4.4}$	$\mathbf{75.1 \pm 4.4}$	69.3 ± 4.8
Diabetes	76.9 ± 1.7	$\mathbf{71.7 \pm 2.3}$	71.1 ± 2.3	71.1 ± 2.2
German	76.4 ± 2.2	$\mathbf{73.6 \pm 2.1}$	72.7 ± 2.4	73.1 ± 2.1
Heart	83.8 ± 3.1	82.4 ± 3.6	82.1 ± 3.7	$\mathbf{82.8 \pm 3.7}$
Image	97.5 ± 0.3	88.1 ± 1.0	$\mathbf{95.4 \pm 0.6}$	94.9 ± 0.8
Ringnorm	96.3 ± 0.4	64.1 ± 2.4	64.3 ± 2.3	$\mathbf{64.7 \pm 1.3}$
F. solar	66.7 ± 1.6	63.5 ± 4.0	65.1 ± 1.6	$\mathbf{65.5 \pm 1.8}$
Splice	89.4 ± 0.7	72.0 ± 1.5	72.1 ± 1.6	$\mathbf{72.3 \pm 1.4}$
Thyroid	95.9 ± 2.1	95.1 ± 2.4	$\mathbf{95.7 \pm 2.1}$	95.4 ± 2.3
Titanic	77.3 ± 1.1	77.3 ± 0.6	77.3 ± 0.7	$\mathbf{77.4 \pm 0.7}$
Twonorm	97.4 ± 0.2	97.0 ± 0.5	$\mathbf{97.4 \pm 0.2}$	$\mathbf{97.4 \pm 0.2}$
Waveform	89.9 ± 0.5	88.0 ± 1.1	$\mathbf{88.2 \pm 0.6}$	87.5 ± 0.7

V. CONCLUSIONS

In this paper, we proposed subspace based least squares SVMs (SSLS-SVMs). In SSLS-SVMs, the similarity measure for each class is assumed as the separating hyperplane that separates the associated class with the remaining classes. Then the margin between classes is maximized under the constraints that the similarity measure associated with the class to which a data sample belongs is the largest among all the similarity measures. This leads to a linear all-at-once SVM. Because all-at-once formulation is inefficient, we also formulated SSLS-SVMs by one-against-all formulation.

According to the computer experiments for two-class problems, SSLS-SVMs by all-at-once and one-against-all formulations performed better than the conventional kernel subspace method with equal weights for seven and five problems, respectively, and the SSLS-SVM by all-at-once and one-against-all formulations performed better than those with weights set with eigenvalues for seven and nine problems, respectively.

REFERENCES

[1] S. Watanabe and N. Pakvasa, "Subspace methods of pattern recognition," *Proc. 1st IJCPR*, pp. 283–328, 1973.

[2] E. Oja, *Subspace Methods of Pattern Recognition*, Research Studies Press, 1983.

[3] B. Shöllcopf, S. Mika, C.J.C. Burges, P. Knirsch, K.R. Müller, C. Ratsch, K. Tsuda, and A.J. Smola, "Input space versus feature space in kernel-based methods," *IEEE Trans. Neural Networks*, vol. 10, no. 5, pp. 1000–1016, 1999.

[4] B. Shöllcopf, A.J. Smola, and K.R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, pp. 1299–1319, 1998.

[5] H. Sakano, N. Mukawa, and T. Nakamura, "Kernel mutual subspace method and its application for object recognition," *Electronics and Communication in Japan, Part 2*, vol. 88, no. 6, pp. 45–53, 2005.

[6] K. Fukui and O. Yamaguchi, "The kernel orthogonal mutual subspace method and its application to 3D object recognition," *Proc. ACCV'07*, pp. 467–476, 2007.

[7] K. Fukui, B. Stenger, and O. Yamaguchi, "A framework for 3D object recognition using the kernel constrained mutual subspace method," *Proc. ACCV'06*, pp. 315–324, 2006.

[8] S. Abe, *Support Vector Machines for Pattern Classification*, Springer, 2005.

[9] S. Abe, "Sparse least squares support vector training in the reduced empirical feature space," *Pattern Analysis and Applications*, vol. 10, no. 3, pp. 203–214, 2007.

[10] H. Xiong, M.N.S. Swamy and M.O. Ahmad, "Optimizing the kernel in the empirical feature space," *IEEE Trans. Neural Networks*, vol. 16, no. 2, pp. 460–474, 2005.

[11] G. Rätsch, T. Onda and K.R. Müller, "Soft margins for AdaBoost," *Machine Learning*, vol. 42, no. 3. pp. 287–320, 2001.

[12] http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm.