Support Vector Number Reduction: Survey and Experimental Evaluations

Ho Gi Jung, *Senior Member, IEEE*, and Gahyun Kim

***Abstract*—Although a support vector machine (SVM) is one of the most frequently used classifiers in the field of intelligent transportation systems and shows competitive performances in various problems, it has the disadvantage of requiring relatively large computations in the testing phase. To make up for this weakness, diverse methods have been researched to reduce the number of support vectors determining the computations in the testing phase. This paper is intended to help engineers using the SVM to easily apply support vector number reduction to their own particular problems by providing a state-of-the-art survey and quantitatively comparing three implementations belonging to postpruning, which exploits the result of a standard SVM. In particular, this paper confirms that the support vector number of a pedestrian classifier using a histogram-of-oriented-gradient-based feature and a radial-basis-function-kernel-based SVM can be reduced by more than 99.5% without any accuracy degradation using iterative preimage addition, which can be downloaded from the Internet.**

***Index Terms*—Reduced-set method, support vector machine (SVM), support vector number reduction (SVNR).**

I. Introduction

A SUPPORT vector machine (SV machine or SVM) is a supervised machine learning method proposed by Vapnik in 1995 [1] and has been reported to show competitive performances in various pattern classification problems. While the SVM shows outstanding performances in the field of recognition systems for intelligent vehicles (IVs) and intelligent transportation systems (ITSs), it has become one of the most popular tools for pedestrian detection [2]–[8], vehicle detection [9], [10], driver monitoring [11]–[13], traffic sign recognition [14], [15], license plate recognition [16], traffic monitoring [17]–[21], and intelligent control [22]–[24].

In spite of its superior accuracy, the SVM has been pointed out as having a significant weakness in that it generally requires more computations than existing competitors such as neural networks in the testing phase (or feedforward phase) and, as a result, requires significantly longer execution time [25]. The computations of an SVM testing phase are proportional to the number of SVs, and Steinwart showed that the SV number of a standard SVM is proportional to the size of the given problem in 2003 [26]. This means that the more learning samples are used to enhance the generalization of a classifier in an actual application, the longer the execution time required in the testing phase. For this reason, since soon after the SVM was published, diverse methods reducing the SV number of a standard SVM without accuracy degradation have been researched and published regularly. In 1996, Burges *et al.* showed that the solution of a standard SVM could be represented by less information and named such an operation the "reduced-set method" [27], [28]. Generally, support vector number reduction (SVNR) methods are divided into either prepruning or postpruning as in [29], according to whether they exploit the results of a standard SVM. This paper categorizes SVNR methods that have been researched for a long period of time into five approaches and summarizes them. This categorization is proposed by incorporating categories of previous works [29]–[32], [86] and analyzing them from the viewpoint of an application engineer.

Although SVNR has been researched for quite some time and has produced useful outcomes, it is little known to the field of recognition systems for IVs and ITSs. This is due to the fact that SVNR has been discussed mainly between experts researching machine learning and pattern recognition theory. Although there were some reports about the application of SVNR in the field of recognition systems for IVs and ITSs, none of them provided any quantitative evaluation for practical-sized problems or any comparison between different approaches. According to [34] and [35], Papageorgiou *et al.* developed a wavelet-feature-based pedestrian detector and reduced the SV number from 331 to 29 using the method shown in [27]. However, the problem was too small compared with recent classifiers developed for practical systems: The dimension of the feature vector was 29, and the number of positive and negative samples was 1848 and 7189, respectively. Recently, according to [36], Natroshvili *et al.* developed a wavelet-feature-based pedestrian detector similar to that in [34] and [35] and produced an execution speed 70 times faster by applying the method in [27]. However, they did not provide any detailed numerical data such as the number of learning samples and SVs. Furthermore, as the previous works did not provide any information about how the accuracy was changing while the SV number was reducing, they were not sufficient to dispel any anxieties regarding the stability and robustness of SVNR.

This paper is intended to help engineers using the SVM easily apply SVNR to their own problems by providing a state-of-the-art survey and quantitatively comparing three

Manuscript received December 9, 2012; revised March 20, 2013, August 5, 2013, and September 3, 2013; accepted September 14, 2013. Date of publication October 7, 2013; date of current version March 28, 2014. This work was supported by the Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology, under Grant 2012-0002467. The Associate Editor for this paper was Q. Ji.

H. G. Jung is with the Department of Automotive Engineering, Hanyang University, Seoul 133-791, Korea (e-mail: hogijung@hanyang.ac.kr).

G. Kim is with Hanyang University, Seoul 133-791, Korea.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2013.2282635

implementations belonging to different postpruning approaches. For an application engineer using the SVM but not developing its theory, postpruning is thought to be more suitable than prepruning, which should deal with complicated SVM theory. The targets of the quantitative evaluation have been selected by considering whether it is easy to use and whether it is expected to show stable performance. By applying these three implementations to three practical-sized classification problems, SVNR performances are compared. The problems are the histogram of oriented gradient (HOG)-based pedestrian classifier [37], the Gabor filter bank (GFB)-based pedestrian classifier [5], and a light-blob classifier for intelligent headlight control (IHC) [38]. Based on the results with these three implementations, iterative preimage addition (IPA) belonging to reduced-set construction is concluded to show the best SVNR performance.

II. Survey of SVNR

This section provides a survey of SVNR, which is categorized by five approaches and three target criteria. It starts with a brief review of the SVM to introduce the basic principles of SVM and related terminologies and then discusses the position of SVNR and proposes a new taxonomy of SVNR. Then, two prepruning and three postpruning approaches are explained sequentially. Where, with respect to the reduced-set method, a brief review including the coefficient estimation method is followed by two SV selection methods, namely, reduced-set selection and reduced-set construction.

A. Brief Review of an SVM

The SVM learning phase estimates a classification function $f : \mathcal{X} \rightarrow \mathcal{Y}$ based on a measurement set $\{x_i, y_i\}$, $1 \leq i \leq s$, which is frequently referred to as the learning sample set [39], [40], where $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^N$ is an input vector, and $y_i \in \mathcal{Y} = \{-1, 1\}$ is the corresponding target. To accept a nonlinear classification function, kernel function (or reproducing kernel) k is introduced. According to Mercer's theorem, k can be represented by a dot product in the feature space as

$$k(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j) \tag{1}$$

where the nonlinear mapping function Φ maps an input vector $\boldsymbol{x}$ in the input space $\mathcal{X}$ onto $\Phi(\boldsymbol{x})$ in the feature space $\mathcal{F}$, that is $\Phi : \mathcal{X} \rightarrow \mathcal{F}$. The feature space is a reproducing kernel Hilbert space, and a linear algorithm in the feature space can be represented by the linear span of mapped input vectors $\Phi(\boldsymbol{x})$. The classification function f is given by

$$f(x) = \boldsymbol{\Psi} \cdot \Phi(x) + \rho \tag{2}$$

where $\boldsymbol{\Psi}$ is the normal vector of a hyperplane in the feature space, and ρ is a scalar offset.

The SVM learning phase is implemented by constrained quadratic programming (CQP) searching for a hyperplane having the maximum margin, which is the cost of the optimization problem. When all input vectors cannot be correctly classified, the cost is increased by the multiplication of penalty parameter C and slack variable ξ_i, which denotes how far the misclassified sample $\boldsymbol{x}_i$ goes beyond the hyperplane. Such a formularization, which is frequently referred to as primal CQP, can be written as

$$\begin{aligned} \min_{\Psi,\rho} \quad & \left(\frac{1}{2} \|\boldsymbol{\Psi}\|^2 + C \sum_{i=1}^{s} \xi_i \right) \\ \text{subject to} \quad & y_i \left(\boldsymbol{\Psi} \cdot \Phi(x) + \rho \right) \geq 1 - \xi_i, \;\; \xi_i \geq 0, \\ & \forall\, i = 1, \ldots, s \end{aligned} \tag{3}$$

where s is the number of learning samples. Based on the Kuhn–Tucker theorem, the primal CQP is transformed into a dual form, which is written as

$$\begin{aligned} \min_{\alpha} \quad & \left(\frac{1}{2} \sum_{i,j=1}^{s} \alpha_i \alpha_j \Phi(x_i) \cdot \Phi(x_j) - \sum_{i=1}^{s} \alpha_i \right) \\ \text{subject to} \quad & 0 \leq \alpha_i y_i \leq C \;\; \forall\, i = 1, \ldots, s, \; \sum_{i=1}^{s} \alpha_i = 0. \end{aligned} \tag{4}$$

The optimization problem is transformed into problem searching for the optimal coefficient α_i of each input vector $\boldsymbol{x}_i$ belonging to the learning sample set. It is well known that input vectors with nonzero coefficients will determine the hyperplane and are called support vectors. The most frequently used kernel functions are linear (5) and Gaussian (6), which is also known as radial basis function (RBF). Thus

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (1 + \boldsymbol{x}_i \cdot \boldsymbol{x}_j)^p \tag{5}$$

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathrm{e}^{-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}. \tag{6}$$

Kernel matrix (or kernel Gram matrix) $\mathbf{K}$ is defined by applying a specific kernel function to every pair of input space vectors such as

$$\mathbf{K}_{ij} := k(\boldsymbol{x}_i, \boldsymbol{x}_j), \quad \text{where} \quad \{\boldsymbol{x}_i, \ldots, \boldsymbol{x}_l\} \subset \mathcal{X}.$$

B. Position and Taxonomy of SVNR

SVM-related research can be roughly divided into either SVM development or SVM application. SVM development is related with SVM theory and efficient implementation, and SVM application is related with feature selection, kernel selection, and hyperparameter optimization for a specific problem. However, the SVNR being discussed in this paper could be thought to be between SVM development and SVM application. SVNR deals with SVM theory and efficient implementation; however, it focuses on the minimum SV number rather than the optimal accuracy unlike general SVM development. Simultaneously, it is not restricted to any specific application. Although efficient implementation of the SVM can reduce the SV number resultantly [30], it is distinguishable as its major and explicit goal is not the minimum SV number. The efficient implementation of the SVM includes "dividing by small problems and solving" such as Platt's sequential minimal optimization [102], approximation of a kernel matrix with smaller matrices [103], and SV selection using various criteria and methods [104], [105]. Meanwhile, as SVNR aims to reduce

TABLE I
TAXONOMY OF SVNR

<table>
<tr><th></th><th colspan="2">Target
Method</th><th>Performance</th><th>Representative</th><th>Boundary</th></tr>
<tr><td rowspan="2">Pre-pruning</td><td colspan="2">Customized Optimization</td><td>- SV number cost [43][44][45]
- SV number constraint [46][47]</td><td></td><td></td></tr>
<tr><td colspan="2">Learning Sample Selection</td><td></td><td>- Cluster inner sample, SVM via clustering [48][49][50][51]
- Random sampling, RSVM [52][53][54]</td><td>- Nearest neighbor-based [55][56][57][58][59][60][61]
- Cluster outer sample [62][63][30]</td></tr>
<tr><td rowspan="3">Post-pruning</td><td rowspan="2">Reduced Set Method</td><td>Reduced Set Selection</td><td>- Approximation error [65][66][32][67][68][69][25][33]
- SV number cost [29]
- Noisy SV [70]</td><td>- Linearly dependent SV: [71][72][73][74]
- Cluster center [75]
- SVR [76]</td><td></td></tr>
<tr><td>Reduced Set Construction</td><td>- Approximation error [27][28][64][39][77][78]</td><td>- Cluster center [79][80][81][82][83]
- KPCA [84]</td><td>- Classification function-based [85][86]</td></tr>
<tr><td colspan="2">Learning Sample Selection</td><td>- Approximation error [89][90]
- Noisy SV [91]</td><td></td><td>- Classification function-based [87][31][88]</td></tr>
</table>

the SV number to reduce the execution time of the testing phase, it could be thought to belong to a hardware-friendly SVM. The hardware-friendly SVM, aiming to implement the SVM on an embedded system, includes hardware-based online learning [40], feedforward SVM without multipliers using Laplacian kernel [41], and SVM with integer parameters [42].

This paper categorizes SVNR methods into five approaches by incorporating categories of previous works [29]–[32], [86] and analyzing them from the viewpoint of an application engineer. First, SVNR methods are divided into either prepruning or postpruning as in [29] according to whether they exploit the result of a standard SVM. Prepruning is again divided into either customized optimization or learning sample selection, and postpruning is again divided into three approaches of reduced-set selection, reduced-set construction, and learning sample selection. Notice that although there is a learning sample selection approach in both prepruning and postpruning, they are distinguished by whether a standard SVM is used or not.

Every SVNR method has been commonly based on [27] and [28], which proved that the solution of a standard SVM has room for simplification, but there is great diversity of opinion about what should be the target of selection or elimination. This paper classifies the target selection criteria into three categories: 1) SVM performance: selecting learning samples according to SVM performance or cost; 2) representative sample: based on the idea that duplicate learning samples have no contribution to the SVM solution, duplicate learning samples are replaced with a representative learning sample; and 3) boundary sample: based on the fact that SVM solution is determined only by learning samples adjacent to the boundary between classes, learning samples far from the boundary are eliminated. Such target selection criteria are applied to the learning samples in the case of prepruning and to either learning samples or SVs in the case of postpruning. Table I arranges the main research according to the five approaches and three target selection criteria. Hereafter, each approach will be briefly summarized.

C. *Customized Optimization of Prepruning*

Methods in [43]–[45] add a term about the SV number, i.e., the sum of coefficient α_i multiplied by a constant, to the cost of (4). As a result, a solution with fewer SVs becomes preferable during the optimization procedure. The constant is referred to as a budget parameter, a sparseness weighting parameter, or a penalty parameter.

The methods in [46] and [47] add a constraint that the normal vector $\mathbf{\Psi}$ of the hyperplane is determined by a predefined number of input space vectors to (3). This could be thought of whereby the optimization problem is modified to incorporate the reduced-set method into the standard SVM. The method in [46] initializes the input space vectors with randomly selected learning samples and estimates the input space vectors and their coefficients simultaneously while solving the CQP. The method in [47] iteratively solves the CQP while increasing the number of input space vectors one by one until the performance reaches the required level. In [47], the input space vectors are restricted to be one of the learning samples such as reduced-set selection; however, they are not restricted in [46] in the way reduced-set construction does.

D. *Learning Sample Selection of Prepruning*

Methods belonging to this approach are based on a common assumption that if the number of learning samples is reduced, the problem scale is also reduced, and consequently, the number of SVs is decreased. However, they can be categorized into two groups according to the criteria about which learning samples should be left alone.

The first group removes duplicate learning samples and leaves only representative learning samples. The methods in [48]–[51], which are called "SVM via clustering," detect cluster centers using an unsupervised clustering method such as k-means clustering and leave only the cluster centers [48]–[50] or learning samples near the cluster centers [51]. In the case of [50], it gives weight to each remaining cluster center considering their cluster size to compensate for the information loss that might be caused by the learning sample removal. The methods in [52]–[54], which are called "reduced SVM (RSVM)," assume that randomly selected learning samples can represent all learning samples. Although they apply constraints of (4) to the whole learning sample set, they search for the solution only among the selected subset.

The second group leaves only learning samples near the decision boundary based on the fact that SVs are selected among the learning samples near the decision boundary. The methods in [55]–[61] assume that learning samples whose nearest neighbor belongs to other classes are supposed to be close to the decision boundary. They have investigated various nearest-neighbor detection methods and various criteria to determine whether a learning sample is near the decision boundary. The methods in [30], [62], and [63] detect clusters for each class using an unsupervised clustering method and remove learning samples near the cluster centers. They assume that if clusters are detected from learning samples belonging to the same class, learning samples close to the cluster centers have little chance of being near the decision boundary. Even though the same unsupervised clustering method is used, its application method should be carefully noted as it can be applied to all learning samples [48]–[51] or each class separately [30], [62], [63]. The cluster centers can be left [48]–[51] or removed [30], [62], [63].

E. Brief Review of Reduced-Set Method

The reduced-set method approximates the SVM normal vector $\boldsymbol{\Psi}$ with a reduced set (or reduced SV set), which consists of a smaller number of input space vectors than the original SV set [39]. It could be thought of as a preimage problem of $\boldsymbol{\Psi}$ [27], [28]. Where if the input space vectors are selected among the original SV set, it is referred to as reduced-set selection. Otherwise, it is known as reduced-set construction.

Once the reduced SV set $\{\boldsymbol{z}_i\}$, $1 \leq \boldsymbol{z}_i \leq m$, is determined (the determination will be explained in Sections II-F and G), their coefficients β_i can be calculated from the coefficient α_i of the original SV set $\{\boldsymbol{x}_i\}$, $1 \leq \boldsymbol{x}_i \leq l$ [39]. Let vector $\boldsymbol{\Psi}$ in feature space $\mathcal{F}$ be represented by an expansion of mapped input space vectors $\Phi(\boldsymbol{x}_i)$, $\boldsymbol{x}_i \subset \mathbb{R}^N$. Thus

$$\boldsymbol{\Psi} = \sum_{i=1}^{l} \alpha_i \Phi(\boldsymbol{x}_i) \tag{7}$$

where $\alpha_i \subset \mathbb{R}$. We are looking for a reduced SV set expansion $\boldsymbol{\Psi}'$, given by (8), corresponding to the preimage of $\boldsymbol{\Psi}$. Thus

$$\boldsymbol{\Psi}' = \sum_{i=1}^{m} \beta_i \Phi(\boldsymbol{z}_i) \tag{8}$$

where $m < l$, $\beta_i \subset \mathbb{R}$ and $\boldsymbol{z}_i \subset \mathbb{R}^N$. The problem at hand can be changed into searching for β_i and $\boldsymbol{z}_i$ minimizing the squared error ε defined in (9) between $\boldsymbol{\Psi}$ and $\boldsymbol{\Psi}'$. Hereafter, ε is referred to as the approximation error of $\boldsymbol{\Psi}'$. Although absolute error could be used [27], [28], squared error is used following [39] and [64]. Thus

$$\varepsilon = \|\boldsymbol{\Psi} - \boldsymbol{\Psi}'\|^2. \tag{9}$$

Using (7) and (8), (9) can be rewritten as

$$\begin{aligned}\varepsilon &= \|\boldsymbol{\Psi} - \boldsymbol{\Psi}'\|^2 \\ &= \left\| \sum_{i=1}^{l} \alpha_i \Phi(\boldsymbol{x}_i) - \sum_{i=1}^{m} \beta_i \Phi(\boldsymbol{z}_i) \right\|^2 \\ &= \left(\sum_{i=1}^{l} \alpha_i \Phi(\boldsymbol{x}_i) - \sum_{i=1}^{m} \beta_i \Phi(\boldsymbol{z}_i) \right)^T \\ &\quad \times \left(\sum_{i=1}^{l} \alpha_i \Phi(\boldsymbol{x}_i) - \sum_{i=1}^{m} \beta_i \Phi(\boldsymbol{z}_i) \right) \\ &= \sum_{i,j=1}^{l} \alpha_i \alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sum_{i,j=1}^{m} \beta_i \beta_j k(\boldsymbol{z}_i, \boldsymbol{z}_j) \\ &\quad - 2 \sum_{i=1}^{l} \sum_{j=1}^{m} \alpha_i \beta_j k(\boldsymbol{x}_i, \boldsymbol{z}_j).\end{aligned} \tag{10}$$

By differentiating (10) with respect to β_k, we have

$$\frac{\partial}{\partial \beta_k} \|\boldsymbol{\Psi} - \boldsymbol{\Psi}'\|^2 = -2\Phi(\boldsymbol{z}_k) \left(\boldsymbol{\Psi} - \sum_{i=1}^{m} \beta_i \Phi(\boldsymbol{z}_i) \right).$$

By substituting (7) for $\boldsymbol{\Psi}$ and assuming the result is zero with the optimal $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_m)$, we have

$$\boldsymbol{K}^z \boldsymbol{\beta} = \boldsymbol{K}^{zx} \boldsymbol{\alpha}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_l)$, $\boldsymbol{K}^z_{ij} := k(\Phi(\boldsymbol{z}_i), \Phi(\boldsymbol{z}_j))$, and

$$\boldsymbol{K}^{zx}_{ij} := k\left(\Phi(\boldsymbol{z}_i), \Phi(\boldsymbol{x}_j)\right).$$

Therefore, coefficients of the reduced SV set can be determined as

$$\boldsymbol{\beta} = (\boldsymbol{K}^z)^{-1} \boldsymbol{K}^{zx} \boldsymbol{\alpha}. \tag{11}$$

F. Reduced-Set Selection of Postpruning

Methods belonging to this approach select the input space vectors among the original SV set. They can be categorized into two groups.

The first group focuses on SVM performance. The methods in [25], [32], [33], and [65]–[69] search for a subset of the original SV set minimizing the approximation error. The method in [65] starts from an empty set and iteratively adds an SV that is expected to make the largest decrease in the approximation error. Reversely, in [67] and [68], they start from the original SV set and iteratively remove an SV that is expected to make the least increase in the approximation error. In particular, in [68], an efficient method is proposed for the calculation of the approximation error of the reduced SV set, which should be iteratively recalculated while SVs are being removed, by exploiting the kernel matrix of the original SV set. It will be explained in detail in Section III-A. In [32] and [66], the approximation error, which generally requires a large number of calculation, is estimated with the span of SV. In [25], [33], and [69], a solution represents whether each of the original SV set will be included in the reduced SV set or not, and the optimal solution is searched for using either genetic algorithm or particle swarm optimization. The method in [29] exploits genetic algorithms as in [69] but minimizes the sum of the misclassified sample number and SV number instead of the approximation error. Meanwhile, the method in [70] removes SVs with small SV coefficient α_i based on the fact that a significant portion of the original SV set has very little α_i and contributes little to the classification.

The second group focuses on the fact that a reduced SV set is representative of the original SV set. The methods in [71]–[74] remove linearly dependent SVs that can be presented by a linear span of the remaining SVs. In this case, the coefficients of the reduced SV set can be calculated by (11) or a crosswise propagation process [73], [74] in which the coefficients of the remaining SVs are updated using the linear relationship with the removed SVs. Linearly dependent SVs are detected with a reduced row echelon form or row kernel vector projection. The method in [75] assumes that regions with a high SV density contain important information about the classification function and leaves SVs with a high-SV-density region using density-based clustering. In [76], the classification function is approximated with a fewer number of SVs by applying support vector regression to the original SV set.

G. Reduced-Set Construction of Postpruning

Methods belonging to this approach estimate the reduced SV set using various principles. They can be categorized into three groups.

The first group searches for an input vector set minimizing the approximation error. The concept of the reduced-set method was originally proposed in [27] and [28], which show that searching for a reduced SV set is equivalent to estimating the preimage of feature space vector $\boldsymbol{\Psi}$. They simultaneously estimate the whole of the reduced SV set by using unconstrained optimization. The methods in [39], [64], and [77] incrementally add input space vectors one by one. They estimate the preimage with one input space vector and iteratively reduce the problem at hand by subtracting the preimage from the feature space vector $\boldsymbol{\Psi}$. Additionally, in [39] and [64], a fixed-point theorem-like equation corresponding to the search for the reduced SV set is derived, and it is interpreted as a procedure searching for cluster centers. The method in [77] will be explained in detail in Section III-B. A method for speeding up the calculation required for new data by exploiting the previous results is proposed in [78].

The second group uses the cluster center searching concept in [64] to establish the direction of preimage estimation. The methods in [79] and [80] assume that two vectors in the feature space are supposed to be closer to each other when their kernel function outputs a larger value. They replace the closest two SVs with their weighted average. If the kernel is RBF, it can be thought of as two close Gaussian modes being merged into a single mode. In [81], SV cluster centers in the feature space are detected using an unsupervised clustering method, and their preimages are used as the reduced SV set. In [82], SV clusters are detected based on the local quantization error, and the reduced SV set is estimated by applying function approximation to the clusters. In [83], a fixed number of SVs located near each other in the input space are iteratively replaced with their weighted average. Meanwhile, in [84], the principal components are estimated by applying kernel principal component analysis to the original SV set, and the reduced SV set is estimated with their preimage.

The third group focuses on the boundary SVs. The method in [85] is about an online update method of SVM. If new data close to the decision boundary are entered, it removes the SV farthest from the decision boundary or merges two SVs closest to each other to maintain the number of SVs. The method in [86] estimates representative input space vectors near the decision boundary using weighted learning vector quantization (LVQ). As LVQ is generally used to estimate prototypes, it can be thought to be similar to the methods estimating representative input space vectors using detected cluster centers. However, it is different because it weights input space vectors such that they are more preferable when they are nearer the decision boundary. This group is different from prepruning as it exploits the classification function f of the original SVM to evaluate the proximity to the decision boundary.

H. Learning Sample Selection of Postpruning

This approach trains an SVM with all of the learning samples and trains a simplified SVM with a smaller number of learning samples selected by the trained SVM. These two SVMs are referred to as the original SVM and the RSVM, respectively. Methods belonging to this approach can be categorized into two groups.

The first group, which is known as separable case approximation (SCA), notices that the penalty parameter C of (3) introduced to address inseparable cases drastically increases the complexity of the given problem and that the SV number increases unnecessarily [31]. They assume that learning samples misclassified with the original SVM are noisy and remove them while training the RSVM. Basically, they remove learning samples with a classification function output of less than 0, instead of which the smoothed SCA (SSCA) uses a tunable threshold [88]. In [31], the threshold is established adaptively according to the statistics of the classification function output of the original SVs. In [31] and [87], confidently recognized learning samples are additionally removed. It is supposed to leave only learning samples close to the decision boundary. The method in [88] will be explained in detail in Section III-C.

The second group focuses on SVM performance. The methods in [89] and [90] iteratively remove an SV that is expected to make the least increase in the approximation error from the learning sample set and use the span of SV to estimate the approximation error increase, which requires a lot of computations. In [91], SVs having a large generalized curvature when projected on the hyperplane are assumed to be noisy and are removed from the learning sample set.

III. Investigated Implementations of Postpruning Methods

This paper quantitatively compares three implementations belonging to different postpruning approaches. The targets are selected considering whether they are easy for application engineers to use and whether they are expected to show stably competitive performance. In detail, 1) Does it provide a detailed explanation of the implementation method? Additionally, is its implementation open to the public and downloadable by anyone? 2) Does it show that it has superior performance to the previous works by quantitative evaluations? 3) Is it expected to produce similar results with repetitive operations? That is, are random initialization and selection minimized? 4) Does it focus on general learning problems, not just online learning?

A. AEbSE

Approximation-error-based sequential elimination (AEbSE) [68] belonging to reduced-set selection iteratively removes an SV that is expected to make the least increase in the approximation error. The AEbSE name is designated in this paper for the reader's convenience and is evaluated with a code provided by Kobayashi and Otsu in [68]. The SV determined to be removed, i.e., $\boldsymbol{x}_i$, is assumed to be linearly dependent on the other SVs. Thus

$$\Phi(\boldsymbol{x}_i) = \sum_{j \neq i} \gamma_j \Phi(\boldsymbol{x}_j).$$

In this case, $\boldsymbol{\Psi}$ of (7) can be rewritten as

$$\boldsymbol{\Psi} = \sum_{j \neq i} \alpha_j \Phi(\boldsymbol{x}_j) + \alpha_i \sum_{j \neq i} \gamma_j \Phi(\boldsymbol{x}_j) = \sum_{j \neq i} (\alpha_j + \alpha_i \gamma_j) \Phi(\boldsymbol{x}_j).$$

Using this relationship, the approximation error and the SV coefficient of the reduced SV set, which are required to be calculated repeatedly, are efficiently calculated using the kernel matrix and SV coefficients of the original SV set. That is, the approximation error increase ε_i, when the ith SV is removed, is calculated by (13) instead of (10), exploiting the projection of the SV onto the remaining SVs written as (12), where $\mathbf{K}_{(i)}$ is the submatrix of $\mathbf{K}$ excluding the ith row and column, $\boldsymbol{k}_{(i)}$ is the ith column vector of $\mathbf{K}$ excluding the ith row, and λ is introduced as a regularization parameter for avoiding rank reduction of the kernel matrix. Thus

$$\boldsymbol{\gamma}_i = \left(\mathbf{K}_{(i)} + \lambda \mathbf{I}\right)^{-1} \boldsymbol{k}_{(i)} \tag{12}$$

$$\varepsilon_i = k_{ii} - 2\boldsymbol{\gamma}_i^T \boldsymbol{k}_{(i)} + \boldsymbol{\gamma}_i^T \mathbf{K}_{(i)} \boldsymbol{\gamma}_i. \tag{13}$$

After the SV index making the least approximation error increase, i^*, is determined (i.e., $i^* = \arg\min_i \varepsilon_i$), SV coefficient vector $\boldsymbol{\beta}$ is calculated by (14) instead of (11), where $\boldsymbol{\alpha}_{(i^*)}$ is the SV coefficient column vector excluding the i^*th row. Thus

$$\boldsymbol{\beta} = \boldsymbol{\alpha}_{(i*)} + \alpha_{i*} \boldsymbol{\gamma}_{i*}. \tag{14}$$

After the i^*th SV is removed, the reduced SV set is regarded as the new original SV set, and the removal procedure is iterated until the requirements are satisfied: $\boldsymbol{\alpha} = \boldsymbol{\beta}, \mathbf{K} = \mathbf{K}_{(i^*)}$. In [68], the iteration is terminated when Hinge loss [85] becomes larger than a threshold. Additionally, a method solving the least square problem of (12) exploiting the inverse matrix $\mathbf{H} = (\mathbf{K} + \lambda \mathbf{I})^{-1}$ of the original SV set is proposed, and it is reported to drastically reduce the computation time.

B. IPA

IPA [77] belonging to reduced-set construction implements a method iteratively estimating the preimage of $\boldsymbol{\Psi}$ using unconstrained optimization and subtracting its contribution to reduce the given problem. The name, i.e., IPA, is designated in this paper for the reader's convenience. It is open to the public as a function of Statistical Pattern Recognition Toolbox (STPRtool) [77].

Originally, the method iteratively estimating the preimage of $\boldsymbol{\Psi}$ and subtracting it from the continuously shrinking $\boldsymbol{\Psi}$ is proposed by Scholköpf *et al.* in [39]. The acquired preimages are used as the reduced SV set. Let us assume that we can estimate a preimage $\boldsymbol{z}$ of a vector $\boldsymbol{\Psi}$ in the feature space. If the preimage $\boldsymbol{z}_m$ and its coefficient β_m of time step m are determined, feature space vector $\boldsymbol{\Psi}_{m+1}$ of time step $m+1$ can be calculated by subtracting their product from $\boldsymbol{\Psi}_m$ as

$$\boldsymbol{\Psi}_{m+1} := \boldsymbol{\Psi}_m - \beta_m \Phi(\boldsymbol{z}_m). \tag{15}$$

The feature space vector $\boldsymbol{\Psi}_m$ of time step m can be expressed by training samples and acquired preimages by that time step as

$$\boldsymbol{\Psi}_m = \sum_{i=1}^{l} \alpha_i \Phi(\boldsymbol{x}_i) - \sum_{i=1}^{m-1} \beta_i \Phi(\boldsymbol{z}_i).$$

It can be regarded as a feature space vector expressed by $l + m - 1$ input space vectors $\boldsymbol{x}_m = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_{m-1})$ and its coefficients $\boldsymbol{\alpha}_m = (\alpha_1, \ldots, \alpha_l, -\beta_1, \ldots, -\beta_{m-1})$. Therefore, once $\boldsymbol{z}_m$ is determined, β_m can be calculated by (11). Such iteration continues until m reaches the target SV number or approximation error ε becomes smaller than a threshold.

Now, the remaining problem is only how to estimate the preimage of feature space vector $\boldsymbol{\Psi}$. STPRtool [77] provides three implementations, namely, fixed-point theorem-based [39], unconstrained optimization-based, and distance constraints (in the feature space)-based [92]. Among them, as the unconstrained optimization-based shows the best performance in our experiments, it is used for our experiments with IPA. The preimage problem is finding an input space vector $\boldsymbol{z} \in \mathcal{X}$ of which feature space image $\Phi(\boldsymbol{z}) \in \mathcal{F}$ approximates best the feature space vector $\boldsymbol{\Psi} \in \mathcal{F}$ as

$$\begin{aligned}\hat{\boldsymbol{z}} &= \arg\min_{z} \|\Phi(\boldsymbol{z}) - \boldsymbol{\Psi}\|^2 \\ &= \arg\min_{z} \left\| \Phi(\boldsymbol{z}) - \sum_{i=1}^{l} \alpha_i \Phi(\boldsymbol{x}_i) \right\|^2 \\ &= \arg\min_{z} \left(k(\boldsymbol{z},\boldsymbol{z}) - 2\sum_{i=1}^{l} \alpha_i k(\boldsymbol{z},\boldsymbol{x}_i) + \sum_{i,j=1}^{l} \alpha_i \alpha_j k(\boldsymbol{x}_i,\boldsymbol{x}_j) \right).\end{aligned}$$

As the first and third terms are constants, the problem is the same as finding $\boldsymbol{z}$ maximizing the second term. That is

$$\hat{\boldsymbol{z}} = \arg\max_{z} \sum_{i=1}^{l} \alpha_i k(\boldsymbol{z},\boldsymbol{x}_i). \tag{16}$$

Finding $\boldsymbol{z}$ maximizing the cost function

$$f(\boldsymbol{z}) = \sum_{i=1}^{l} \alpha_i k(\boldsymbol{z},\boldsymbol{x}_i)$$

is conducted by gradient descent optimization in three steps. 1) Randomly select 50 input space vectors from the training data and select one having the maximum cost function output as the initial preimage. 2) Determine the optimal step size t in the direction of $\nabla f(\boldsymbol{z})$ using MATLAB function "fminunc." 3) Iteratively update $\boldsymbol{z}$ as $\boldsymbol{z} = \boldsymbol{z} - t \cdot \nabla f(\boldsymbol{z})$. The iteration continues until the changing amount of $\boldsymbol{z}$ becomes smaller than a threshold. Where, as "fminunc" finds the minimum, the minus of cost function output square is temporarily used.

If an RBF kernel is used, (16) is rewritten as

$$\hat{\boldsymbol{z}} = \arg\max_{z} \sum_{i=1}^{l} \alpha_i \exp\left(-\frac{\|\boldsymbol{z}-\boldsymbol{x}_i\|^2}{2\sigma^2}\right).$$

The cost function of steps 2) and 3) above is

$$f(\boldsymbol{z}) = \sum_{i=1}^{l} \alpha_i \exp\left(-\frac{\|\boldsymbol{z}-\boldsymbol{x}_i\|^2}{2\sigma^2}\right)$$

and the gradient $\nabla f(\boldsymbol{z})$ with respect to $\boldsymbol{z}$ is

$$\begin{aligned}\nabla f(\boldsymbol{z}) &= \sum_{i=1}^{l} \alpha_i \exp\left(-\frac{\|\boldsymbol{z}-\boldsymbol{x}_i\|^2}{2\sigma^2}\right)(\boldsymbol{z}-\boldsymbol{x}_i) \\ &= \boldsymbol{z}\sum_{i=1}^{l} \alpha_i \exp\left(-\frac{\|\boldsymbol{z}-\boldsymbol{x}_i\|^2}{2\sigma^2}\right) \\ &\quad - \sum_{i=1}^{l} \alpha_i \exp\left(-\frac{\|\boldsymbol{z}-\boldsymbol{x}_i\|^2}{2\sigma^2}\right)\boldsymbol{x}_i.\end{aligned}$$

Therefore, in the optimization iteration, once $\boldsymbol{z}$ is determined, $\nabla f(\boldsymbol{z})$ is determined, and the quadratic term of the cost function is approximated by a 1-D function in the direction of gradient $\nabla f(\boldsymbol{z})$.

C. SSCA

SSCA [88] belonging to learning sample selection of post-pruning removes learning samples in which the absolute value of the classification function output is smaller than threshold D and retrains a new SVM. SSCA is proposed as an extension of the SCA [31], [87].

Let the label of an input vector $\boldsymbol{x}$ be $y(\boldsymbol{x})$ and the output of the original and RSVM with respect to $\boldsymbol{x}$ be $\hat{y}(\boldsymbol{x})$ and $\hat{y}'(\boldsymbol{x})$, respectively. The fact that an RSVM approximates the original SVM can be thought of as the minimization of the approximation error given by

$$E_1 = P\left(\hat{y}(\boldsymbol{x}) \neq \hat{y}'(\boldsymbol{x})\right).$$

As the output of the RSVM with respect to the input vector that is misclassified by the original SVM does not have a negative influence on the accuracy of the RSVM, those input vectors can be ignored by modifying the approximation error as

$$E_2 = P\left(\hat{y}(\boldsymbol{x}) \neq \hat{y}'(\boldsymbol{x}) | \hat{y}(\boldsymbol{x}) = y(\boldsymbol{x})\right). \tag{17}$$

It can be interpreted that if the outputs of the RSVM with respect to learning samples that is correctly classified by the original SVM are coincidental with those of the original SVM, the approximation error could be minimized. Based on this assumption, SCA proposes that if misclassified learning samples are removed and a new SVM is trained, the problem complexity will be significantly decreased, and the resultant SV number will be decreased without any accuracy degradation. To remove the redundant complexity around the decision boundary, SSCA proposes to additionally remove the learning samples satisfying

$$y(\boldsymbol{x})\left(\boldsymbol{\Psi} \cdot \Phi(\boldsymbol{x}) + \rho\right) < D, \quad \text{with} \quad D > 0. \tag{18}$$

IV. Experimental Results

SVNR performance is evaluated by applying three target implementations to three problems with practical size. The problems are pedestrian classification and light-blob classification. In particular, pedestrian classification is known to be one of the most complicated recent problems in the field of ITS classification [2]–[4], [7], [8]. The problem to solve is to determine whether a given image contains a pedestrian or not. As classifiers using different kinds of features are regarded as different classifiers even though their objectives are common, this paper deals with a HOG-based pedestrian classifier and a GFB-based pedestrian classifier as two different problems for SVNR.

While changing the parameters of each implementation, the SV remaining ratio is measured. The SV remaining ratio is the ratio of the SV number of the RSVM with respect to that of the original SVM. The minimum SV remaining ratio means the smallest SV remaining ratio without accuracy degradation. We investigate whether each implementation shows stable performance and whether they are sensitive to parameter tuning by observing the accuracy graph around the minimum SV remaining ratio.

Fig. 1. Examples of a Daimler pedestrian classification benchmark data set.

LIBSVM [93] generally used in our field is used as the standard SVM required to apply postpruning methods, and only the RBF kernel shown in (6) is dealt with because of practical issues. There are two reasons for the selection of the RBF kernel. First, an RBF kernel is generally known to show competitive and stable performance and requires comparatively fewer numbers of hyperparameters [94]. Second, it is proven that in the case of a homogeneous polynomial kernel, which is one of the polynomial kernels, the reduced SVs should be eigenvectors of the feature space [27]. Therefore, as the exact solution can always be determined, differences between implementations are not supposed to be significant.

A. Application to HOG-Based Pedestrian Classification

The three implementations are applied to HOG-based pedestrian classification. The "Daimler pedestrian classification benchmark data set" is used for training and testing. It is used in [8] and is available to the public and downloadable at [95]. It consists of one training set and one test set, and each consists of three and two data sets, respectively. Each data set contains 4800 pedestrian images and 5000 nonpedestrian images. Consequently, the training set contains $29\,400 (= 4800 \times 3 + 5000 \times 3)$ images, and the test set contains $19\,600 (= 4800 \times 2 + 5000 \times 2)$ images. Fig. 1 shows the examples of the data set. The first row shows examples of pedestrian images, and the second row shows examples of nonpedestrian images. The image resolution is 36×18.

The HOG proposed by Dalad and Triggs in [37] has been reported to show competitive performance in the pedestrian classification problem [2]–[4], [6]. Feature extraction from an input image is implemented by a function that is open at MATLAB CENTRAL [96]. Feature-extraction-related parameters and SVM parameters are set following [6]: bin number $= 18$ (signed gradient), cell size $= 3$, block size $= 2$, description stride $= 2$, L2 norm clipping $= 0.2$, RBF hyperparameter $\gamma = 0.01$, penalty parameter $C = 1$. Consequently, the total feature length is 3960. The accuracy of the original SVM trained using the parameters is 93.55%, and the SV number is 4947.

Fig. 2 shows accuracy graphs with respect to the SV remaining ratio when the three implementations are applied to the HOG-based pedestrian classifier. The red dotted line represents the accuracy of the original SVM. Each implementation is repeatedly applied to the original SVM while changing the dominant parameter for its performance. The parameter is set according to the coarse-to-fine strategy: After the overall range is tested with rough grids, parameters around the point with the largest performance change are tested with finer grids. For AEbSE, 338 Hinge loss thresholds in the range of 0–3.37 are used. For IPA, 339 SV number thresholds in the range of 1–339 are used. For SSCA, 79 threshold values, where D is defined in (18), in the range of 0–4 are used. In the case of IPA, as it uses random initialization when estimating the preimage, a test with a parameter value is repeated ten times, and the final result is acquired by their average. Fig. 2(a) shows that IPA shows the best SVNR performance. To investigate the changing portion of the IPA graph, the area of the SV remaining ratio of 0%–3% is enlarged in Fig. 2(b). In particular, it is noticeable that the RSVM with an SV remaining ratio of 0.42% (the SV number $= 21$) shows the same accuracy as the original SVM.

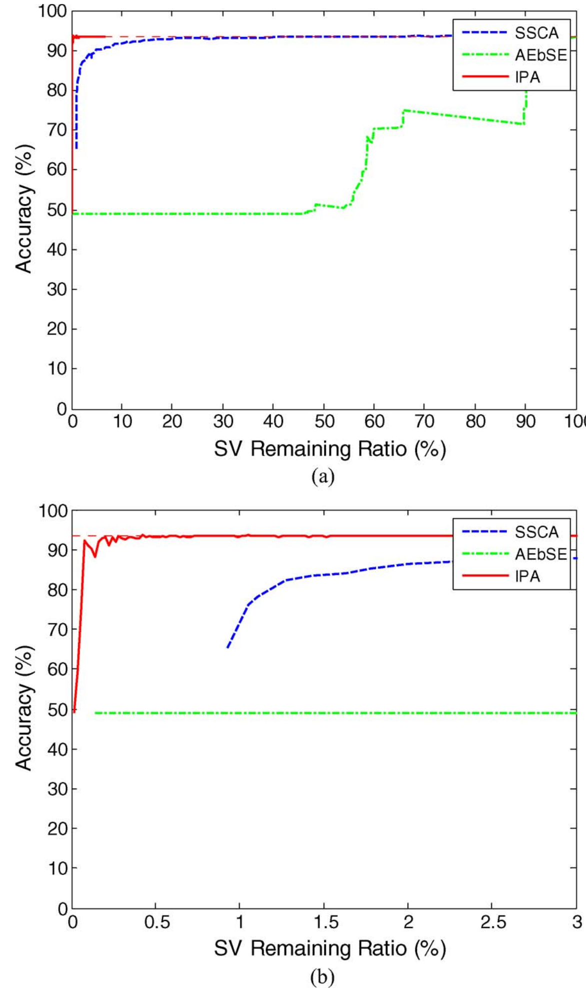


Fig. 2. Experimental results of a HOG-based pedestrian classifier. (a) Accuracy with respect to the SV remaining ratio of 0%–100%. (b) Accuracy with respect to the SV remaining ratio of 0%–3%.

B. Application to GFB-Based Pedestrian Classification

The three implementations are applied to GFB-based pedestrian classification. The "Daimler pedestrian classification benchmark data set" is used for training and testing.

The Gabor filter is defined as a product of a 2-D Gaussian function and a sine wave and is supposed to be able to measure the image power of a specific frequency and direction at a specific location. The GFB is a group of Gabor filters of various shapes in a frequency domain. The GFB-based feature has already shown good performance in various recognition problems

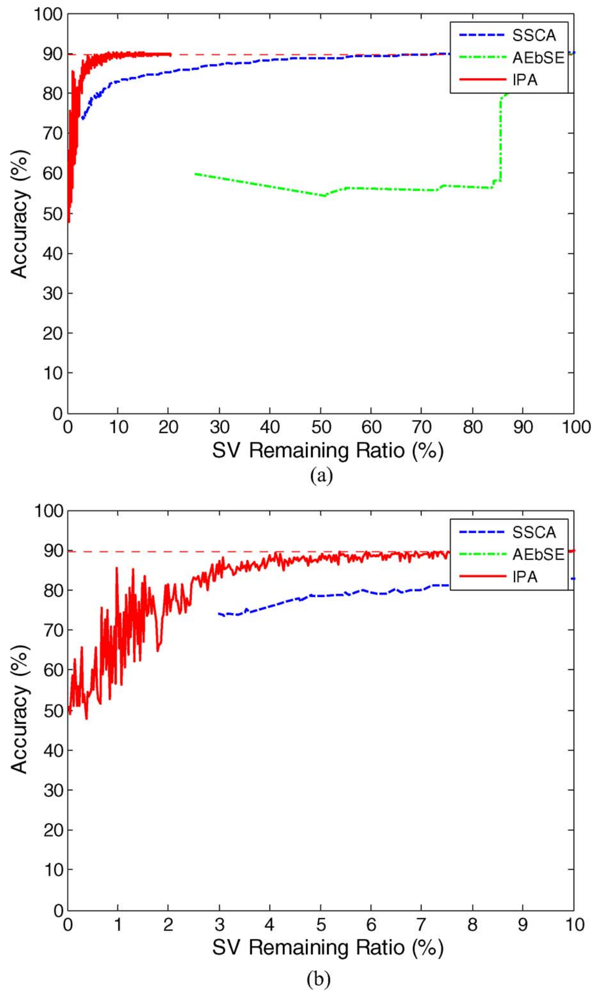


Fig. 3. Experimental results of a GFB-based pedestrian classifier. (a) Accuracy with respect to the SV remaining ratio of 0%–100%. (b) Accuracy with respect to the SV remaining ratio of 0%–10%.

[5], [99]–[101]. Feature-extraction-related parameters and SVM parameters are set following [5]: orientation number $=$ 6, scale number $= 4$, the lowest average frequency $= 0.05$, the highest average frequency $= 0.4$, the filter size $= 16 \times 16$, RBF hyperparameter $\gamma = 0.00095013$, penalty parameter $C =$ 582791681602951.6. For each Gabor filter response, mean, standard deviation, and skewness are extracted and used as features. Consequently, the total feature length is 648. The accuracy of the original SVM trained using the parameters is 89.74%, and the SV number is 3253.

Fig. 3 shows accuracy graphs with respect to the SV remaining ratio when the three implementations are applied to the GFB-based pedestrian classifier. The red dotted line represents the accuracy of the original SVM. For AEbSE, 290 Hinge loss thresholds in the range of 0–2.91 are used. For IPA, 666 SV number thresholds in the range of 1–666 are used. For SSCA, 201 D values in the range of 0–5 are used. Similar to the previous experiments, parameters are tested according to the coarse-to-fine strategy, and results of IPA are the average of ten experiments with a specific parameter. Fig. 3(a) shows that IPA shows the best SVNR performance. To investigate the changing portion of the IPA graph, the area of the SV remaining ratio of 0%–10% is enlarged in Fig. 3(b). In particular, the RSVM with the SV remaining ratio of 7.65% shows the same accuracy as the original SVM.

Fig. 4. Examples of a nighttime forward scene image.

C. Application to Intelligent Headlight Control

To investigate whether the target implementations work well with a multiclass problem and a classifier under development, in other words, when the classifier is not finely tuned and the accuracy is not so high, the three implementations are applied to a light-blob classifier [97] for IHC under development.

IHC enhances a driver's convenience and safety by preventing glare of other vehicles' drivers and improving the usability of the high beam during nighttime driving [38]. IHC detects image areas brighter than the surrounding, which are called light blobs, corresponding to lamps on other vehicles, and classifies them into four classes such as headlamp, tail lamp, street lamp, and nonlamp including reflective traffic signs. The given problem is a multiclass classification with four classes, and LIBSVM decomposes it into six two-class classifications, where LIBSVM uses the 1-against-1 approach, and the detailed information can be found at [98].

The data set used for training and testing is collected by applying the light-blob detection method in [97] to images acquired on a normal suburb roadway at night. Fig. 4 shows the examples of nighttime frontal scene images used for the experiments. The training data set contains 31 263 images, and the test data set contains 16 739 images. The features for the light-blob classifier have 12 dimensions and are extracted by measuring the location, velocity, intensity, area, and color of the detected light blobs. SVM parameters are set by searching with a rough grid: RBF hyperparameter $\gamma = 4$ and penalty parameter $C = 4096$. The accuracy of the original SVM trained using the parameters is 73.33%, and the SV number is 8301.

Fig. 5 shows accuracy graphs with respect to the SV remaining ratio when the three implementations are applied to the light-blob classifier. The red dotted line represents the accuracy of the original SVM. For AEbSE, 129 Hinge loss thresholds in the range of 0–1 are used. For IPA, 63 SV number thresholds in the range of 1–63 are used. For SSCA, 68 D values in the range of 0–4 are used. Similar to the previous experiments, parameters are tested according to the coarse-to-fine strategy,

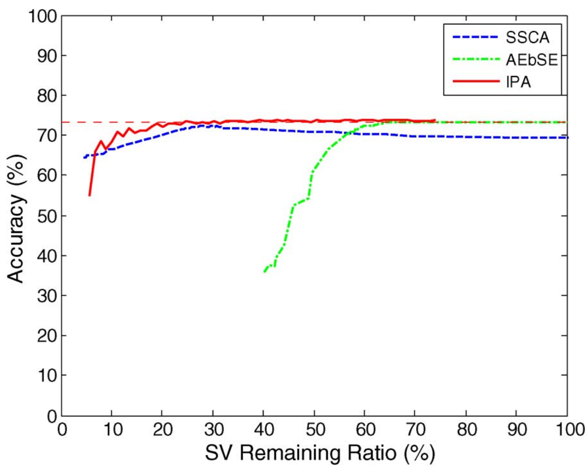


Fig. 5. Experimental results of a light-blob classifier.

and results of IPA are the average of ten experiments with a specific parameter. Fig. 5 shows that IPA shows the best SVNR performance. In particular, the RSVM with an SV remaining ratio of 24.70% shows the same accuracy as the original SVM. This suggests that the target implementations can significantly reduce the SV number even when the accuracy of a classifier is relatively low. In addition, it means that the similar SVNR performance can be achieved with a multiclass classifier.

D. Discussion

By applying three implementations to three problems with practical size, it is confirmed that IPA shows the best SVNR performance. 1) Its minimum SV remaining ratio is the smallest. 2) When the SV remaining ratio is larger than the minimum SV remaining ratio, its accuracy maintains that of the original SVM. 3) As it uses the SV number as the threshold determining whether the iteration will continue or not, users can intuitively set the threshold.

The reasons why AEbSE shows poorer SVNR performance compared with expectations are analyzed as follows. 1) When the dimension of the input vector becomes higher, the possibility that the input vector samples are linearly dependent seems to become lower. 2) The strategy to subtract one-by-one from the original SV set seems unstable and prone to collapse. More than two SVs might be linearly dependent. In such a case, the kernel matrix becomes singular, and the inverse matrix calculation of (12) tends to be unstable. The regulation parameter λ inserted to prevent such a situation makes tuning difficult but seems to fail in terms of clearing the problem. In the experiments explained in the previous sections, various λ values are tested, and the optimal one among them is used. 3) The experimental results of AEbSE coincide with the results in [73], i.e., that a standard SVM generally contains 1%–9% linearly dependent SVs (or dispensable SVs).

The reason why SSCA shows poorer SVNR performance compared with expectations can be explained by the metaphor of the force and torque balance between SVs. If the decision hyperplane is imagined to be a paper sheet,

$$\sum_{i=1}^{S} \alpha_i = 0$$

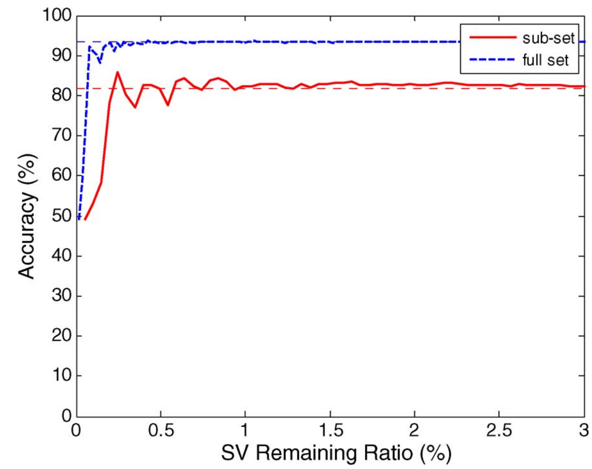


Fig. 6. Comparison between a full-set and subset HOG-based pedestrian classifier.

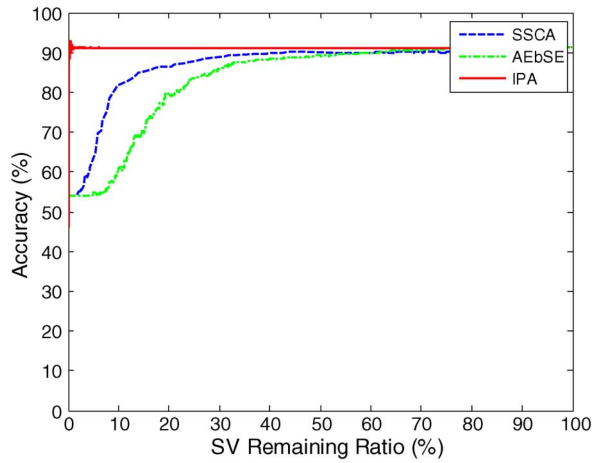
of (4) means the sum of forces that SVs apply on the paper sheet should be 0, and

$$\boldsymbol{\Psi} = \sum_{i=1}^{l} \alpha_i \Phi(\boldsymbol{x}_i)$$

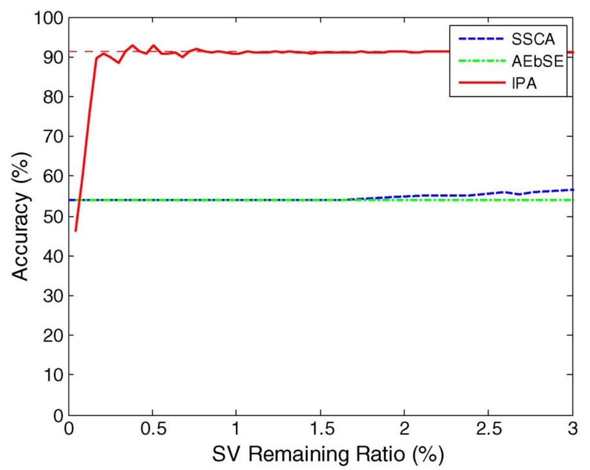
of (7) means the sum of torques that SVs apply on the paper sheet should be also 0 [28], [79]. Therefore, even when an SV is misclassified by the original SVM, it might help the other SVs to be correctly classified by attracting or pushing the decision hyperplane. In other words, the assumption of (17) that misclassified samples could be removed without any accuracy degradation seems to be invalid.

Meanwhile, in the case of IPA, as it estimates the preimage of $\boldsymbol{\Psi}$, which is continuously reduced by subtracting the previous preimages as in (15), the elements of the reduced SV set are supposed to be virtually orthogonal to each other and are less likely to be linearly dependent. As IPA [77] can initialize the reduced SV set exploiting the original SVM, it is expected to show statistically superior performance compared with [46], which belongs to prepruning, and randomly initialize the reduced SV set.

A hypothesis that the SVNR performance is proportional to the accuracy of the original SVM could be derived from the previous experimental results. In other words, can we expect higher SVNR performance if the original SVM shows higher accuracy? To verify the hypothesis, an additional experiment is conducted. By training the HOG-based pedestrian classifier using only one of three data sets, a new original SVM with lower accuracy is constructed. It is noticeable that every condition except the size of the learning data set is the same as the original experiment. The SVM using three data sets is referred to as the full-set classifier, and the SVM using only one data set is referred to as the subset classifier. The accuracy of the subset classifier is 81.84%, and the SV number is 2025. While changing the SV number threshold from 1 to 223, an accuracy graph is acquired. Fig. 6 shows the accuracy graphs of the full-set and subset classifiers with respect to the SV remaining ratio. Although the SV numbers of two original SVMs are different, the SVNR performances are observed to have almost similar

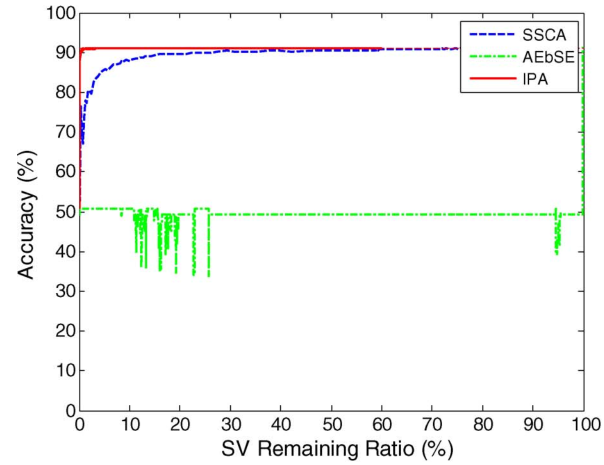


(a)

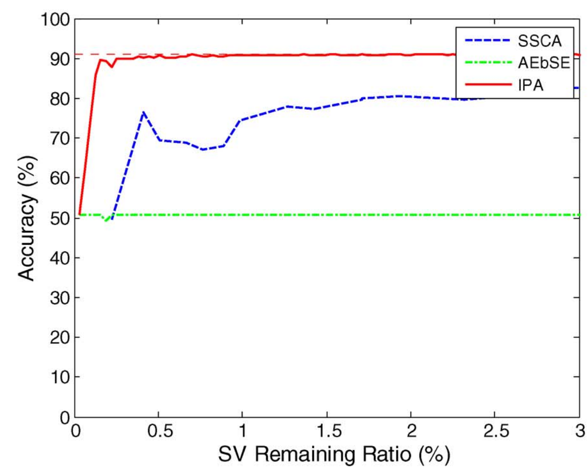


(b)

Fig.7 Experimental results of a bright subset. (a) Accuracy with respect to the SV remaining ratio of 0%–100%. (b) Accuracy with respect to the SV remaining ratio of 0%–3%.



(a)



(b)

Fig. 8. Experimental results of a dark subset. (a) Accuracy with respect to the SV remaining ratio of 0%–100%. (b) Accuracy with respect to the SV remaining ratio of 0%–3%.

tendency. Considering that the SVNR performance of the subset classifier is almost the same as that of the full-set classifier although the accuracy of the subset classifier is definitely lower than that of the GFB-based pedestrian classifier, it can be concluded that the SVNR performance is dependent on the characteristics of the given problem, such as the feature used, rather than the accuracy of the original SVM.

Finally, to compare the robustness of the three implementations, they are applied to two SVMs trained with extremely different illumination conditions. The “Daimler pedestrian classification benchmark data set” is used for training and testing. Based on the average intensity of the training set $(= 122.97)$, the training and testing sets are divided into two subsets. The images with greater average intensity are referred to as the bright subset, and the images with less average intensity are referred to as the dark subset. For each subset, HOG-based classifiers are trained. The bright subset contains 13 233 images for training and 7466 images for testing. The accuracy of the bright subset SVM is 91.24%, and the SV number is 2348. The dark subset contains 16 167 images for training and 12 134 images for testing. The accuracy of the dark subset SVM is 91.13%, and the SV number is 3151. The accuracies are similar to the full-set classifier because the test set is also divided into two subsets. According to the procedures described in Section IV-A, the SVNR performances of the three implementations are compared. Fig. 7 shows the accuracy graphs of the bright subset, and Fig. 8 shows the accuracy graphs of the dark subset. Although the relationship between the SVNR performances of the three implementations does not change, we can conclude that their robustness to constructed SVM is significantly different: The SVNR performances of IPA are almost the same, but those of AEbSE are extremely different.

V. Conclusion

This paper has categorized SVNR reducing the SV number of an SVM into five approaches and quantitatively compared three implementations belonging to different postpruning approaches from each other by applying them to three problems with practical size. Experiments have been conducted with data sets having a similar scale with practical applications, and tuning sensitivity and SVNR performance stability have been evaluated by measuring not only the minimum SV number maintaining the accuracy without degradation but also the accuracy graph with respect to the SV remaining ratio.

Consequently, IPA shows the best SVNR performance in all problems. That is, its minimum SV remaining ratio is the smallest, and its accuracy maintains that of the original SVM when the SV remaining ratio is larger than the minimum SV remaining ratio. Furthermore, IPA is intuitive and easy to use because its iteration is controlled by the SV number threshold. It has been confirmed that SVNR methods can significantly reduce the SV number even when the accuracy of the original SVM is low and the problem is a multiclass classification. It is proven that SVNR performance is dependent on the characteristics of the given problem rather than the accuracy of the original SVM. Therefore, if an application engineer has problems with the execution time of the SVM testing phase or data size, this paper highly recommends that they preferentially apply IPA to their own particular problem using STPRtool open to the public.

Topics requiring further studies are as follows: 1) finding the requirement or common characteristics of features having good SVNR performance; 2) a comparison between various implementations with various problems; and 3) developing or finding efficient preimage estimation methods for various kinds of kernels and problems.

References

[1] C. Cortes and V. N. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[2] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 743–761, Apr. 2012.

[3] D. Gerónimo, A. M. López, A. D. Sappa, and T. Graf, "Survey of pedestrian detection for advanced driver assistance systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 7, pp. 1239–1258, Jul. 2010.

[4] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2179–2195, Dec. 2009.

[5] H. G. Jung and J. Kim, "Constructing a pedestrian recognition system with a public open database, without the necessity of re-training: An experimental study," *Pattern Anal. Appl.*, vol. 13, no. 2, pp. 223–233, May 2010.

[6] S. Paisitkariangkrai, C. Shen, and J. Zhang, "Performance evaluation of local features in human classification and detection," *IET Comput. Vis.*, vol. 2, no. 4, pp. 236–246, Dec. 2008.

[7] T. Gandhi and M. M. Trivedi, "Pedestrian protection systems: Issues, survey, and challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 3, pp. 413–430, Sep. 2007.

[8] S. Munder and D. M. Gavrila, "An experimental study on pedestrian classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 11, pp. 1863–1868, Nov. 2006.

[9] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 5, pp. 694–711, May 2006.

[10] H. T. Niknejad, A. Takeuchi, S. Mita, and D. McAllester, "On-road multivehicle tracking using deformable object model and particle filter with improved likelihood estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 748–758, Jun. 2012.

[11] S. Y. Cheng and M. M. Trivedi, "Vision-based infotainment user determination by hand recognition for driver assistance," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 759–764, Sep. 2010.

[12] T. Ersal, H. J. A. Fuller, O. Tsimhoni, J. L. Stein, and H. K. Fathy, "Model-based analysis and classification of driver distraction under secondary tasks," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 692–701, Sep. 2010.

[13] Y. Ling, M. L. Reyes, and J. D. Lee, "Real-time detection of driver cognitive distraction using support vector machines," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 340–350, Jun. 2007.

[14] H. Gómez-Moreno, S. Maldonado-Bascón, P. Gil-Jiménez, and S. Lafuente-Arroyo, "Goal evaluation of segmentation algorithms for traffic sign recognition," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 4, pp. 917–930, Dec. 2010.

[15] H. Fleyeh and M. Dougherty, "Traffic sign classification using invariant features and support vector machines," in *Proc. IEEE Intell. Veh. Symp.*, Eindhoven, The Netherlands, Jun. 4–6, 2008, pp. 530–535.

[16] Y. Wen, Y. Lu, J. Yan, Z. Zhou, K. M. von Deneen, and P. Shi, "An algorithm for license plate recognition applied to intelligent transportation system," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 830–845, Sep. 2011.

[17] V. Tyagi, S. Kalyanaraman, and R. Krishnapuram, "Vehicular traffic density state estimation based on cumulative road acoustics," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1156–1166, Sep. 2012.

[18] N. Buch, S. A. Velastin, and J. Orwell, "A review of computer vision techniques for the analysis of urban traffic," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 920–939, Sep. 2011.

[19] Y. Ma, M. Chowdhury, M. Jeihani, and R. Fries, "Accelerated incident detection across transportation networks using vehicle kinetics and support vector machine in cooperation with infrastructure agents," *IET Intell. Transp. Syst.*, vol. 4, no. 4, pp. 328–337, Dec. 2010.

[20] Y. Ma, M. Chowdhury, A. Sadek, and M. Jeihani, "Real-time highway traffic condition assessment framework using Vehicle–Infrastructure Integration (VII) with Artificial Intelligence (AI)," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 4, pp. 615–627, Dec. 2009.

[21] J. Zhou, D. Gao, and D. Zhang, "Moving vehicle detection for automatic traffic monitoring," *IEEE Trans. Veh. Technol.*, vol. 56, no. 1, pp. 51–59, Jan. 2007.

[22] B. Pattipati, C. Sankavaram, and K. R. Pattipati, "System identification and estimation framework for pivotal automotive battery management system characteristics," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 4, no. 6, pp. 869–884, Nov. 2011.

[23] X. Huang, Y. Tan, and X. He, "An intelligent multifeature statistical approach for the discrimination of driving conditions of a hybrid electric vehicle," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 453–465, Jun. 2011.

[24] A. Lidozzi, L. Solero, F. Crescimbini, and A. Di Napoli, "SVM PMSM drive with low resolution Hall-effect sensors," *IEEE Trans. Power Electron.*, vol. 22, no. 1, pp. 282–290, Jan. 2007.

[25] H.-J. Lin and J. P. Yeh, "A hybrid optimization strategy for simplifying the solutions of support vector machines," *Pattern Recognit. Lett.*, vol. 31, no. 7, pp. 563–571, May 2010.

[26] I. Steinwart, "Sparseness of support vector machines," *J. Mach. Learn. Res.*, vol. 4, pp. 1071–1105, Dec. 2003.

[27] C. J. C. Burges, "Simplified support vector decision rules," in *Proc. 13th Int. Conf. Mach. Learn.*, 1996, pp. 71–77.

[28] C. J. C. Burges and B. Scholköpf, "Improving the accuracy and speed of support vector learning machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 1997, vol. 9, pp. 375–381.

[29] M. F. A. Hady, W. Herbawi, M. Weber, and F. Schwenker, "A multi-objective genetic algorithm for pruning support vector machines," in *Proc. 23rd IEEE ICTAI*, Boca Raton, FL, USA, Nov. 7–9, 2011, pp. 269–275.

[30] X. Li, N. Wang, and S.-Y. Li, "A fast training algorithm for SVM via clustering technique and Gabriel graph," in *Proc. Commun. Comput. Inf. Sci.*, 2007, vol. 2, pp. 403–412.

[31] G. H. Bakir, J. Weston, and L. Bottou, "Breaking SVM complexity with cross-training," in *Proc. 17th Neural Inf. Process. Syst. Conf.*, 2005, vol. 17, pp. 81–88.

[32] S. Agarwal, V. V. Saradhi, and H. Karnick, "Kernel-based online machine learning and support vector reduction," *Neurocomputing*, vol. 71, no. 7–9, pp. 1230–1237, Mar. 2008.

[33] J. P. Yeh and C. M. Chiang, "Optimal reducing the solutions of support vector machines based on particle swarm optimization," in *Proc. Electron. Signal Process., Lect. Notes Elect. Eng.*, 2011, vol. 97, pp. 203–213.

[34] C. Papageorgiou, T. Evgeniou, and T. Poggio, "A trainable pedestrian detection system," in *Proc. IEEE Int. Conf. Intell. Veh.*, Stuttgart, Germany, Oct. 28–30, 1998, pp. 241–246.

[35] C. Papageorgiou and T. Poggio, "Trainable pedestrian detection," in *Proc. Int. Conf. Image Process.*, Kobe, Japan, Oct. 24–28, 1999, vol. 4, pp. 35–39.

[36] K. Natroshvili, M. Schmid, M. Stephan, A. Stiegler, and T. Schamm, "Real time pedestrian detection by fusing PMD and CMOS cameras," in *Proc. IEEE Intell. Veh. Symp.*, Eindhoven, The Netherlands, Jun. 4–6, 2008, pp. 925–929.

[37] N. Dalad and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Diego, CA, USA, Jun. 25, 2005, vol. 1, pp. 886–893.

[38] Y. Li and S. Pankanti, "A performance study of an intelligent headlight control system," in *Proc. IEEE Workshop Appl. Comput. Vis.*, Kona, HI, USA, Jan. 5/6, 2011, pp. 440–447.

[39] B. Scholköpf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Muller, G. Raetsch, and A. J. Smola, "Input space versus feature space in kernel-based methods," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1000–1017, Sep. 1999.

[40] D. Anguita, A. Boni, and S. Ridella, "A digital architecture for support vector machines: Theory, algorithm, and FPGA implementation," *IEEE Trans. Neural Netw.*, vol. 14, no. 5, pp. 993–1009, Sep. 2003.

[41] D. Anguita, S. Pischiutta, S. Ridella, and D. Sterpi, "Feed-forward support vector machine without multipliers," *IEEE Trans. Neural Netw.*, vol. 17, no. 5, pp. 1328–1331, Sep. 2006.

[42] D. Anguita, A. Ghio, S. Pischiutta, and S. Ridella, "A support vector machine with integer parameters," *Neurocomputing*, vol. 72, no. 1–3, pp. 480–489, Dec. 2008.

[43] P. M. L. Drezet and R. F. Harrison, "A new method for sparsity control in support vector classification and regression," *Pattern Recognit.*, vol. 34, no. 1, pp. 111–125, Jan. 2001.

[44] O. Dekel and Y. Singer, "Support vector machines on a budget," in *Proc. NIPS Found.*, 2006, pp. 1–8.

[45] S.-Y. Chiu, L.-S. Lan, and Y.-C. Hwang, "Two sparsity-controlled schemes for 1-norm support vector classification," in *Proc. IEEE Northeast Workshop Circuits Syst.*, Montreal, QC, Canada, Aug. 5–8, 2007, pp. 337–340.

[46] M. Wu, B. Schölkopf, and G. Bakir, "Building sparse large margin classifiers," in *Proc. 22nd Int. Conf. Mach. Learn.*, 2005, pp. 996–1003.

[47] S. S. Keerthi, O. Chapelle, and D. DeCoste, "Building support vector machines with reduced classifier complexity," *J. Mach. Learn. Res.*, vol. 7, pp. 1493–1515, Jul. 2006.

[48] M. B. de Almeida, A. de Piidua Braga, and J. P. Braga, "SVM-KM: Speeding SVMs learning with a priori cluster selection and k-means," in *Proc. 6th Brazilian Symp. Neural Netw.*, 2000, pp. 162–167.

[49] Q. Tran, Q.-L. Zhang, and X. Li, "Reduce the number of support vectors by using clustering techniques," in *Proc. Int. Conf. Mach. Learn. Cybern.*, Xi'an, China, Nov. 2–5, 2003, pp. 1245–1248.

[50] G. H. Nguyen, S. L. Phung, and A. Bouzerdoum, "Efficient SVM training with reduced weighted samples," in *Proc. IJCNN*, Jul. 18–23, 2010, pp. 1–5.

[51] S.-X. Lu, J. Meng, and G.-E. Cao, "Support vector machine based on a new reduced samples method," in *Proc. 9th Int. Conf. Mach. Learn. Cybern.*, Qingdao, China, Jul. 11–14, 2010, pp. 1510–1514.

[52] Y.-J. Lee and O. L. Mangasarian, "RSVM: Reduced support vector machine," in *Proc. 1st SIAM Int. Conf. Data Mining*, Chicago, IL, USA, Apr. 5–7, 2001, pp. 1–17.

[53] K.-M. Lin and C.-J. Lin, "A study on reduced support vector machines," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1449–1459, Nov. 2003.

[54] Y.-J. Lee and S.-Y. Huang, "Reduced support vector machines: A statistical theory," *IEEE Trans. Neural Netw.*, vol. 18, no. 1, pp. 1–13, Jan. 2007.

[55] N. Panda, E. Y. Chang, and G. Wu, "Concept boundary detection for speeding up SVMs," in *Proc. 23rd Int. Conf. Mach. Learn.*, Pittsburgh, PA, USA, 2006, pp. 581–588.

[56] H. Shin and S. Cho, "Neighborhood property-based pattern selection for support vector machines," *Neural Comput.*, vol. 19, no. 3, pp. 816–855, Mar. 2007.

[57] Z. Sun, Z. Liu, S. Liu, Y. Zhang, and B. Yang, "Active learning with support vector machines in remotely sensed image classification," in *Proc. 2nd Int. CISP*, Oct. 17–19, 2009, pp. 1–6.

[58] A. Lyhyaoui, M. Martıacute;nez, I. Mora, M. Vázquez, J.-L. Sancho, and A. R. Figueiras-Vidal, "Sample selection via clustering to construct support vector-like classifiers," *IEEE Trans. Neural Netw.*, vol. 10, no. 6, pp. 1474–1481, Nov. 1999.

[59] J. Wang, P. Neskovic, and L. N Cooper, "Training data selection for support vector machines," in *Proc. ICNC*, 2005, pp. 554–564.

[60] F. Angiulli, "Fast nearest neighbor condensation for large data sets classification," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 11, pp. 1450–1464, Nov. 2007.

[61] F. Angiulli and A. Astorino, "Scaling up support vector machines using nearest neighbor condensation," *IEEE Trans. Neural Netw.*, vol. 21, no. 2, pp. 351–357, Feb. 2010.

[62] J. Chen and C.-L. Liu, "Fast multi-class sample reduction for speeding up support vector machines," in *Proc. IEEE Int. Workshop Mach. Learn. Signal Process.*, Beijing, China, Sep. 18–21, 2011, pp. 1–6.

[63] R. Koggalage and S. Halgamuge, "Reducing the number of training samples for fast support vector machine classification," *Neural Inf. Process.–Lett. Rev.*, vol. 2, no. 3, pp. 57–65, Mar. 2004.

[64] B. Schölkopf, P. Knirsch, A. Smola, and C. Burges, "Fast approximation of support vector kernel expansions, and an interpretation of clustering as approximation in feature spaces," in *DAGM-Symp., Informatik aktuell Berlin*, 1998, pp. 124–132, Berlin, Germany, Springer-Verlag.

[65] Q. Li, L. Jiao, and Y. Hao, "Adaptive simplification of solution for support vector machine," *Pattern Recognit.*, vol. 40, no. 3, pp. 972–980, Mar. 2007.

[66] V. V. Saradhi and H. Karnick, "Classifier complexity reduction by support vector pruning in kernel matrix learning," in *Proc. 9th IWANN*, 2007, pp. 268–275.

[67] S. Vucetic, V. Coric, and Z. Wang, "Compressed kernel perceptrons," in *Proc. DCC*, Mar. 16–18, 2009, pp. 153–162.

[68] T. Kobayashi and N. Otsu, "Efficient reduction of support vectors in Kernel-based methods," in *Proc. 16th IEEE ICIP*, Nov. 7–10, 2009, pp. 2077–2080.

[69] H.-J. Lin and J. P. Yeh, "Optimal reduction of solutions for support vector machines," *Appl. Math. Comput.*, vol. 214, no. 2, pp. 329–335, Aug. 2009.

[70] J. Manikandan and B. Venkataramani, "Evaluation of multiclass support vector machine classifiers using optimum threshold-based pruning technique," *IET Signal Process.*, vol. 5, no. 5, pp. 506–513, Aug. 2011.

[71] T. Downs, K. E. Gates, and A. Masters, "Exact simplification of support vector solution," *J. Mach. Learn. Res.*, vol. 2, pp. 293–297, Mar. 2001.

[72] S. Wang, "Online modeling based on support vector machine," in *Proc. Chin. Control Decision Conf.*, Jun. 17–19, 2009, pp. 1188–1191.

[73] X. Liang, R.-C. Chen, and X. Guo, "Pruning support vector machines without altering performances," *IEEE Trans. Neural Netw.*, vol. 19, no. 10, pp. 1792–1803, Oct. 2008.

[74] X. Liang, "An effective method of pruning support vector machine classifiers," *IEEE Trans. Neural Netw.*, vol. 21, no. 1, pp. 26–38, Jan. 2010.

[75] H. Xu, W. Song, Z. Hu, C. Chen, X. Zhao, and J. Zhang, "A speedup SVM decision method for online EEG processing in motor imagery BCI," in *Proc. 10th Int. Conf. ISDA*, Nov. 29–Dec. 1, 2010, pp. 149–153.

[76] E. Osuna and F. Girosi, "Reducing the run-time complexity of support vector machines," in *Proc. Int. Conf. Pattern Recognit.*, Brisbane, Australia, Aug. 1998, pp. 1–10.

[77] V. Franc and V. Hlavac, STPRtool: Statistical Pattern Recognition Toolbox. [Online]. Available: http://cmp.felk.cvut.cz/cmp/software/stprtool

[78] B. Chen, H. Liu, and Z. Bao, "Speeding up SVM in test phase: Application to radar HRRP ATR," in *Proc. Lect. Notes Comput. Sci.*, 2006, vol. 4232, pp. 811–818.

[79] D. D. Nguyen and T. B. Ho, "An efficient method for simplifying support vector machines," in *Proc. 22nd Int. Conf. Mach. Learn.*, Bonn, Germany, 2005, pp. 617–624.

[80] D. D. Nguyen and T. B. Ho, "A bottom-up method for simplifying support vector solutions," *IEEE Trans. Neural Netw.*, vol. 17, no. 3, pp. 792–796, May 2006.

[81] Z.-Q. Zeng, J. Gao, and H. Guo, "Simplified support vector machines via kernel-based clustering," in *Proc. 19th Australian Joint Conf. AI LNAI*, 2006, vol. 4304, pp. 1189–1195.

[82] K. Zhang and J. T. Kwok, "Simplifying mixture models through function approximation," *IEEE Trans. Neural Netw.*, vol. 21, no. 4, pp. 644–658, Apr. 2010.

[83] G.-S. Hu, G.-Y. Ren, and J.-J. Jiang, "A support vector reduction method for accelerating calculation," in *Proc. Int. Conf. Wavelet Anal. Pattern Recognit.*, Beijing, China, Nov. 2–4, 2007, pp. 1408–1412.

[84] N. Sundaram, Support vector machine approximation using kernel PCA, Univ. California at Berkeley, Berkeley, CA, USA, Tech Rep. UCB/EECS-2009-94. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-94.pdf

[85] Z. Wang and S. Vucetic, "Online training on a budget of support vector machines using twin prototypes," *Stat. Anal. Data Mining*, vol. 3, no. 3, pp. 149–169, Jun. 2010.

[86] M. Blachnik and M. Kordos, "Simplifying SVM with weighted LVQ algorithm," in *Proc. 12th Int. Conf. IDEAL*, vol. 6936, *LNCS*, 2011, pp. 212–219.

[87] Y. Li, W. Zhang, G. Wang, and Y. Cai, "Simplify decision function of reduced support vector machines," in *Proc. MICAI*, vol. 3789, *LNAI*, 2005, pp. 435–442.

[88] D. Geebelen, J. A. K. Suykens, and J. Vandewalle, "Reducing the number of support vectors of SVM classifiers using the smoothed separable case approximation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 4, pp. 682–688, Apr. 2012.

[89] J. Guo, N. Takahashi, and T. Nishi, "A learning algorithm for improving the classification speed of support vector machines," in *Proc. Eur. Conf. Circuit Theory Des.*, Aug. 28–Sep. 2, 2005, vol. 3, pp. III/381–III/384.

[90] J. Guo, N. Takahashi, and T. Nishi, "An efficient method for simplifying decision functions of support vector machines," *IEICE Trans. Fundam.*, vol. E89–A, no. 10, pp. 2795–2802, Oct. 2006.

[91] Y. Zhan and D. Shen, "Design efficient support vector machine for fast classification," *Pattern Recognit.*, vol. 38, no. 1, pp. 157–161, Jan. 2005.

[92] J. T.-Y. Kwok and I. W.-H. Tsang, "The pre-image problem in kernel methods," *IEEE Trans. Neural Netw.*, vol. 15, no. 6, pp. 1517–1525, Nov. 2004.

[93] C.-C. Chang and C.-J. Lin (2011, Apr.). LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* [Online]. *2(3)*, pp. 27:1–27:27. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[94] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, A Practical Guide to Support Vector Classification. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

[95] Daimler Pedestrian Classification Benchmark Dataset. [Online]. Available: http://www.gavrila.com

[96] X. Leo, Histogram of Oriented Gradients. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/33863-histograms-of-oriente

[97] S. Eum and H. G. Jung, "Enhancing light blob detection for intelligent headlight control using lane detection," *IEEE Intell. Transp. Syst.*, vol. 14, no. 2, pp. 1003–1011, Jun. 2013.

[98] C.-W. Hsu and C.-J. Lin, "How could I generate the primal variable w of linear SVM?" [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm/faq.html

[99] B. S. Manjunath and W.-Y. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 8, pp. 837–842, Aug. 1996.

[100] C. Liu and H. Wechsler, "Independent component analysis of Gabor features for face recognition," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 919–928, Jul. 2003.

[101] Z. Sun, G. N. Bebis, and R. H. Miller, "Monocular precrash vehicle detection: Features and classifiers," *IEEE Trans. Image Process.*, vol. 15, no. 7, pp. 2019–2034, Jul. 2006.

[102] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advanced in Kernel Methods*. Cambridge, MA, USA: MIT Press, 1999, pp. 185–208.

[103] D. Achlioptas, F. Mcsherry, and B. Schölkopf, "Sampling techniques for kernel methods," in *Proc. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 3–8, 2001, pp. 335–342.

[104] G. Valentini and T. G. Dietterich, "Low bias bagged support vector machines," in *Proc. Int. Conf. Mach. Learn.*, Washington, DC, USA, Aug. 21–24, 2003, pp. 752–759.

[105] W. Zhang and I. K. King, "A study of the relationship between support vector machine and Gabriel graph," in *Proc. Int. Joint Conf. Neural Netw.*, Honolulu, HI, USA, May 12–17, 2002, pp. 239–244.

Ho Gi Jung (M'05–SM'10) received the B.E., M.E., and Ph.D. degrees in electronic engineering from Yonsei University, Seoul, Korea, in 1995, 1997, and 2008, respectively.

From 1997 to April 2009, he was with MANDO Corporation Global R&D H.Q., Pyeongtaek, Korea, where he developed environmental recognition systems for intelligent parking assist systems, collision warning and avoidance systems, and active pedestrian protection systems. From May 2009 to February 2011, he was a Full-Time Researcher and Research Professor with Yonsei University, where he researched computer vision applications for intelligent surveillance systems and biometric systems. Since March 2011, he has been an Assistant Professor with Hanyang University, Seoul, where he is engaged in research on recognition systems for intelligent vehicles. His research interests include recognition systems for intelligent vehicles, next-generation vehicles, computer vision applications, and pattern recognition applications.

Gahyun Kim received the B.S. degree in electronics engineering from Ewha University, Seoul, Korea, in 2009 and the M.S. degree in electrical and electronic engineering from Yonsei University, Seoul, in 2011.

She is with MANDO Corporation Global R&D H.Q., Pyeongtaek, Korea, where she is engaged in the development of high-beam assist systems. Her research interests include recognition systems for intelligent vehicles, computer vision, and pattern recognition applications.