Contents lists available at ScienceDirect

Pattern Recognition

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

Choosing the kernel parameters for support vector machines by the inter-cluster distance in the feature space

Kuo-Ping Wu, Sheng-De Wang*

Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, ROC

ARTICLE INFO

Article history:
Received 25 February 2007
Received in revised form 19 August 2008
Accepted 28 August 2008

Keywords:
SVM
Support vector machines
Kernel parameters
Inter-cluster distances

ABSTRACT

Determining the kernel and error penalty parameters for support vector machines (SVMs) is very problem-dependent in practice. A popular method to deciding the kernel parameters is the grid search method. In the training process, classifiers are trained with different kernel parameters, and only one of the classifiers is required for the testing process. This makes the training process time-consuming. In this paper we propose using the inter-cluster distances in the feature spaces to choose the kernel parameters. Calculating such distance costs much less computation time than training the corresponding SVM classifiers; thus the proper kernel parameters can be chosen much faster. Experiment results show that the inter-cluster distance can choose proper kernel parameters with which the testing accuracy of trained SVMs is competitive to the standard ones, and the training time can be significantly shortened.

1. Introduction

The support vector machines (SVMs) [1] have been highly concerned in recent years. The training algorithms of SVMs try to find the optimal separating hyperplane by maximizing the margin between the hyperplane and the data and thus minimizing the upper bound of the generalization error. Delivering promising results makes the SVMs extensively applicable in many information processing tasks, including data classification, pattern recognition and function estimation. SVMs are ordinarily used as binary classifiers that separate the data space into two areas. The separating hyperplane is not explicitly given. It is represented by a small number of data points, called support vectors (SVs). However, the real data are often linearly inseparable in the input space. To overcome this, data are mapped into a high dimensional feature space, in which the data are sparse and possibly more separable. In practice, the mapping is also not explicitly given. Instead, a kernel function is incorporated to simplify the computation of the inner product value of the transformed data in the feature space. That is, choosing a kernel function implies defining the mapping from the input space to the feature space.

As the data distribution may change in different feature spaces, the performance of an SVM depends on the kernel largely. However, there is no theoretical method for determining a kernel function and its parameters. Also, there is no a priori knowledge for setting the kernel parameter C. A straightforward way for model selection is the grid search algorithm [2]. It trains SVMs with all desired combinations of parameters and screens them according to the training accuracy. Ancona et al. use the receiver operating characteristic (ROC) curve for selecting better parameter combinations [3]. The genetic algorithms (GAs) are also applied to SVM parameter search [4–8]. With some possible parameter combinations as the initial population, the GAs update the population by crossover and mutation operations to keep or generate the better combinations according to the performance of the resulted SVMs. The simulated annealing techniques can also be applied to ensure that the global optimum of parameter combinations can be found [9]. These methods select the best parameter combination from the population evolved generation by generation, thus requiring to train many SVMs. However, with a sample size l, the training time complexity of each SVM is experimentally shown to be $O(l)-O(l^{2.2})$ [10,11], or even can be $O(l^4)$ in worst case [12]. Therefore, methods for choosing a parameter combination from a population are still time-consuming. To conquer this, some heuristics are proposed to choose kernel parameters by calculating some kinds of indexes. Debnath and Takahashi create an index for identifying appropriate kernel parameters by analyzing the principle components of data in the feature space [13]. This method requires a non-automatic criteria to choose 'a good set of kernel parameters'. Bi et al. propose using the relationship of boundary points in the feature space to select a suitable σ for Gaussian kernels [14]. Their method operates for several iterations, and the time complexity of each iteration is $O(l^2)$. Such indexes often have computational complexity about the same to the SVM algorithms, resulting in very little

* Corresponding author.
E-mail address: sdwang@ntu.edu.tw (S.-D. Wang).

0031-3203/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2008.08.030

actual computation time saving in comparison to training an SVM for choosing the kernel parameters. Much SVM-related research work treats the kernel parameters and the penalty parameter C in different ways. Changing the kernel parameters is equivalent to selecting the feature spaces, and tuning C is corresponding to weighting the slack variables, the error terms. Therefore, the meanings of tuning the kernel parameters and tuning C are different. In fact, C has no intuitive meaning of the geometry, and most of the researches are concentrated on tuning the kernel parameters.

In this paper we propose using one of the cluster validation measures as the data separation index to predict possible good choices of the SVM parameters. The index shows the degree the data are separated in the feature space defined by the parameters and the corresponding kernel function. The experiment results, presented in Section 4, show that the parameter combinations with higher values of the cluster validation index can have high testing accuracy in most cases. The computational complexity of computing the data separation index is $O(l^2)$, while the actual computation time is much shorter than training many SVMs for selecting one kernel parameter combination. Good parameter combinations can be fast determined by the separation index since calculating such index costs much less computation time than using a grid search method to train an SVM.

2. SVMs and kernel selection

The SVM is designed for binary-classification problems, assuming the data are linearly separable. Given the training data (x_i, y_i), $i = 1, \ldots, l$, $x_i \in R^n$, $y_i \in \{+1, -1\}$, where the R^n is the input space, x_i is the sample vector and y_i is the class label of x_i, the separating hyperplane (ω, b) is a linear discriminating function that solves the optimization problem:

$$\min_{\omega,b} \quad \langle \omega, \omega \rangle$$
$$\text{subject to} \quad y_i(\langle \omega, x_i \rangle + b) \geqslant 1, \quad i = 1, \ldots, l, \tag{1}$$

where $\langle .,. \rangle$ indicates the inner product operation. The minimal distance between the samples and the separating hyperplane, i.e. the margin, is $1/\|\omega\|$.

In order to relax the margin constraints for the non-linearly separable data, the slack variables are introduced into the optimization problem:

$$\min_{\xi,\omega,b} \quad \langle \omega, \omega \rangle + C \sum_{i=1}^{l} \xi_i$$
$$\text{subject to} \quad y_i(\langle \omega, x_i \rangle + b) \geqslant 1 - \xi_i, \quad i = 1, \ldots, l, \quad \xi_i \geqslant 0. \tag{2}$$

This leads to a soft margin SVM that is generally discussed and applied. The resulted classifier is called the 1-norm soft margin SVM, and C is the penalty parameter of error. The decision function of the classifier is

$$\text{sign}\left(\sum_{x_i:SV} y_i \alpha_i \langle x_i, x \rangle + b \right). \tag{3}$$

In practice, since the real data are often not linearly separable in the input space, the data can be mapped into a high dimensional feature space, in which the data are sparse and possibly more separable. The mapping is often not explicitly given. Instead, a kernel function is incorporated to simplify the computation of the inner product value of the transformed data in the feature space.

When using a function $\phi : X \to F$ to map the data into a high dimensional feature space, the decision function of the classifier becomes

$$\text{sign}\left(\sum_{x_i:SV} y_i \alpha_i \langle \phi(x_i), \phi(x) \rangle + b \right). \tag{4}$$

The mapping ϕ is not given explicitly in most cases. Instead, a kernel function $K(x, x') = \langle \phi(x), \phi(x') \rangle$ gives the inner product value of x and x' in the feature space. Choosing a kernel function is therefore choosing a feature space and the decision function becomes

$$\text{sign}\left(\sum_{x_i:SV} y_i \alpha_i K(x_i, x) + b \right). \tag{5}$$

The generally used kernel functions are

- linear: $K(x, x') = \langle x, x' \rangle$;
- polynomial: $K(x, x') = (\gamma \langle x, x' \rangle + r)^d$, $\gamma > 0$;
- radial basis function (RBF): $K(x, x') = e^{-\gamma \|x - x'\|^d}$, $\gamma > 0$;
- sigmoid: $K(x, x') = \tanh(\gamma \langle x, x' \rangle + r)$.

For certain parameters, the linear kernel is a special case of RBF kernels [15]. Also, the sigmoid kernel behaves like the RBF kernel [16]. When the data are linearly inseparable, a non-linear kernel that maps the data into the feature space non-linearly can handle the data better than the linear kernels. As the polynomial kernel requires more parameters to be chosen, the RBF kernel is a reasonable first choice of kernel function [17]. When using the RBF kernel, the parameters $\langle d, \gamma \rangle$ should be set properly. Generally d is set to be 2. Thus the kernel value is related to the Euclidean distance between the two samples. γ is related to the kernel width.

To apply SVM on a multi-class classification problems, the problem can be divided into sub-problems which are binary-classification problems. The often suggested implementations for SVM multi-class classification are the one-against-rest method [18], the one-against-one method [19] and the directed acyclic graph SVM (DAGSVM) [20]. For an n-class classification problem, the one-against-rest method constructs n SVM models with each one separating a single class with all the other classes, while the one-against-one method constructs $n(n-1)/2$ classifiers where each one classifies two classes only. The training phase of DAGSVM is identical to the one-against-one method, while the trained classifiers are organized into a rooted DAG in its testing phase and thus the testing phase can be faster than one-against-rest method. Lin et al. [2] showed that the one-against-one method performs best and can be trained faster than the one-against-rest method.

3. Inter-cluster distances in the feature space

Generalization is the ability that a trained model predicts the target value of an input sample which is not in the training set. Many indexes can be used to assess the generalization ability. The validation accuracy is a straightforward one and easy to be used. For example, the training process of the grid search shown in List 1 uses the validation accuracy to indicate the generalization ability of the classifier to the validation data. When using the grid search method which is shown in List 1 for training an SVM, the training data and each $\langle C, \text{kernel parameter} \rangle$ combination are used for an SVM model. The model is then used to classify the validation data. The parameter combination that results in a model with best validation accuracy is picked as the best choice of the problem. The testing accuracy is acquired by applying the corresponding SVM model on the testing data. The specific C and kernel parameter values form a grid in the

$\langle C$, kernel parameter$\rangle$ parameter space, and the grid search method exhaustively search the grid to find a best combination. Obviously, the wider or the finer the search space is, the more possibility the grid search method find the best parameter combination. In addition, when the validation data are not available, k-fold cross validation can be used to acquire the validation accuracy [17]. Training with k-fold cross validation, the training data are partitioned into k sets. Each set is used as the validation data once, while the rest data are used as the training data. The training process repeats k times for the k validation data sets and the average validation rate is used to choose the parameter combination. In addition to the training and testing samples, the validation samples are reserved for validating the trained model to see whether the model over-fits the training samples. Therefore, it reflects the generalization ability of the model to unknown data.

List 1: Using the grid search method to choose a desired classifier.

```
for each ⟨C, kernel parameter⟩ combination
  train a classifier with the training data
  use the classifier to classify the validation data
pick the classifier with highest validation accuracy for testing
process usage
```

There are some other SVM-related indexes that can estimate the generalization ability. Takahashi [21] used the ratio of the numbers of SVs to the training samples as an index. This suggests a useful index as M_{ij}/SV_{ij} for the classifier of classes i and j, where M_{ij} is the number of training data and SV_{ij} is the number of SVs. Phetkaew [22] proposed using the SVM margin to identify a classifier that causes wrong classifications. As the margin decreases, the distance between the nearest two samples from different classes decreases too, raising the degree of confusion between the two classes. These indexes require the information of the trained classifiers, so they are as time-consuming as the grid search is. Similar to that the margin stands for, our previous work [23] proposed a separation index which indicates the separation of two classes in the feature space. The index is derived from inter-cluster distances (δ_4) which was used by Bezdek for unsupervised data clustering. Bezdek and Pal [24] mentioned several inter-cluster distance measures (δ_i). They are the measurements of the distance between two clusters. A larger inter-cluster distance value implies a pair of more separated clusters. In our application, we consider the two classes as two labeled clusters. The inter-cluster distances in the sample space can be

$$\delta_1(X_+,X_-)=\min_{\substack{x_+\in X_+\\ x_-\in X_-}} d(x_+,x_-), \tag{6}$$

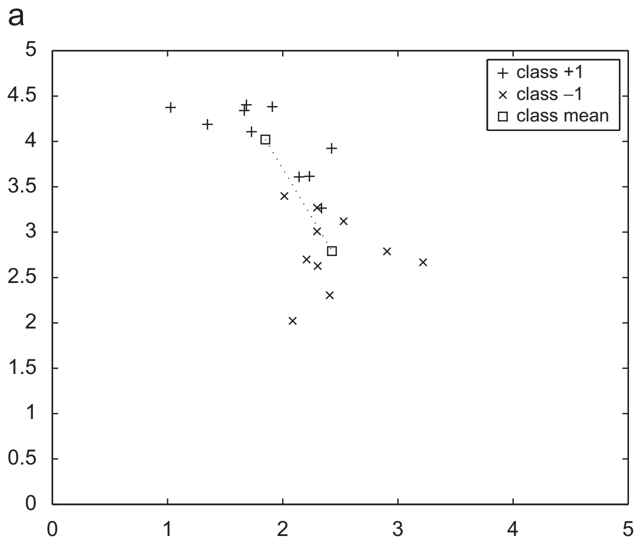
$$\delta_2(X_+,X_-)=\max_{\substack{x_+\in X_+\\ x_-\in X_-}} d(x_+,x_-), \tag{7}$$

$$\delta_3(X_+,X_-)=\frac{1}{l_+l_-}\sum_{\substack{x_+\in X_+\\ x_-\in X_-}} d(x_+,x_-), \tag{8}$$

$$\delta_4(X_+,X_-)=d(\bar{x_+},\bar{x_-})=d\left(\frac{\sum_{x_+\in X_+}x_+}{l_+},\frac{\sum_{x_-\in X_-}x_-}{l_-}\right), \tag{9}$$

$$\delta_5(X_+,X_-)=\frac{1}{l_++l_-}\left(\sum_{x_+\in X_+} d(x_+,\bar{x_-})+\sum_{x_-\in X_-} d(x_-,\bar{x_+})\right), \tag{10}$$

where X_+ and X_- are positive and negative classes, l_+ and l_- are sample sizes of X_+ and X_-, and $\bar{x_+}$ and $\bar{x_-}$ are the class means of X_+ and X_-. δ_1, δ_2 and δ_3 are the shortest, the longest and the



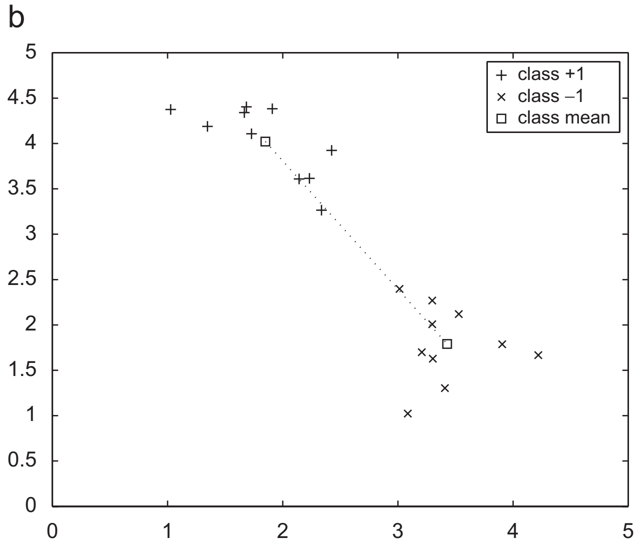


Fig. 1. Artificial data representing the relationship between the class location and the class mean location. The data of +1 class are the same in both figures. However, the data of −1 class in (a) are shifted in (b) by adding the vector (1, −1) to each sample.

average distance between two samples from different classes. δ_5 is a combination of δ_3 and δ_4, and δ_4 is the distance between two class means and is less affected by noises than $\delta_1-\delta_3$ are. Referring to Fig. 1, as the classes more separate away, the distance between class means increases. Thus δ_4 can indicate the class separation robustly.

According to the previous work [23], the distance values can be calculated in the feature space with the L^2-norm and the kernel function incorporated. The kernel function gives the inner product values of vectors in the feature space. Therefore, the distance between two samples in the feature space can be evaluated by applying the L^2-norm on the mapped data as the following equation:

$$\begin{aligned} d(\phi(x),\phi(y)) &= \sqrt{|\phi(x)-\phi(y)|_2^2} \\ &= \sqrt{\phi(x)\cdot\phi(x)-2\phi(x)\cdot\phi(y)+\phi(y)\cdot\phi(y)} \\ &= \sqrt{K(x,x)+K(y,y)-2K(x,y)}. \end{aligned} \tag{11}$$

Eq. (11) can be used to calculate $\delta_1-\delta_3$ in the feature space. We can also calculate the distance value δ_{4F}, which corresponds to the distance measure δ_4 in the feature space:

$$
\begin{aligned}
&\delta_{4F}(X_+,X_-)\\
&\quad = d(\hat{x_+},\hat{x_-})\\
&\quad = \sqrt{|\hat{x_+}-\hat{x_-}|_2^2}\\
&\quad = \sqrt{\left|\frac{\sum_{x_+\in X_+}\phi(x_+)}{l_+}-\frac{\sum_{x_-\in X_-}\phi(x_-)}{l_-}\right|_2^2}\\
&\quad = \sqrt{\frac{\sum_{\substack{x_{+i}\in X_+\\ x_{+j}\in X_+}} K(x_{+i},x_{+j})}{l_+^2}+\frac{\sum_{\substack{x_{-p}\in X_-\\ x_{-q}\in X_-}} K(x_{-p},x_{-q})}{l_-^2}-\frac{2\sum_{\substack{x_{+m}\in X_+\\ x_{-n}\in X_-}} K(x_{+m},x_{-n})}{l_+l_-}},
\end{aligned}
\tag{12}
$$

where $\hat{x_+}$ and $\hat{x_-}$ are the class means of the mapped X_+ and X_- data. Similarly, δ_{5F} can be

$$
\begin{aligned}
&\delta_{5F}(X_+,X_-)\\
&\quad = \frac{1}{l_+ + l_-}\left(\sum_{x_+\in X_+}\sqrt{|\phi(x_+)-\hat{x_-}|_2^2}+\sum_{x_-\in X_-}\sqrt{|\phi(x_-)-\hat{x_+}|_2^2}\right),
\end{aligned}
\tag{13}
$$

where

$$
\begin{aligned}
|\phi(x_+)-\hat{x_-}|_2^2 &= \left|\phi(x_+)-\frac{\sum_{x_-\in X_-}\phi(x_-)}{l_-}\right|_2^2\\
&= K(x_+,x_+)+\frac{\sum_{\substack{x_{-p}\in X_-\\ x_{-q}\in X_-}} K(x_{-p},x_{-q})}{l_-^2}-\frac{2\sum_{x_{-n}\in X_-}K(x_+,x_{-n})}{l_-}.
\end{aligned}
\tag{14}
$$

Calculating these indexes are with $O(l^2)$ computational complexity. Since these indexes are related to inter-cluster distances, they can represent the separation degree of the classes and thus can estimate the classifier generalization ability.

List 2: Using the distance index to choose a desired classifier.

```
for each kernel parameter combination
    calculate the preferred distance index with the training data
with the kernel parameter combination which leads to best
separation index
    for each C
        train a classifier with the training data
pick the classifier with highest validation accuracy for test-
ing process usage
```

The SVM algorithms look for the separation hyperplane with which the class data are separated. Since the validation accuracy is an estimation of classifier generalization ability, it can be substituted by using the separation indexes as the heuristics to choose the kernel parameters. Because the kernel functions are involved in calculating the index values, the kernel parameters can be chosen according to the index values. Only the penalty parameter C should be chosen by the validation process, and the time needed for the training process for different kernel parameters can be saved. The corresponding training process is listed in List 2. For each kernel parameter combination, a value of the desired separation index is calculated. The largest separation index value likely implies the most separation of the two classes in the feature space, and the corresponding kernel parameter combination is selected. With the selected kernel parameter combination, for each penalty parameter C value, an SVM model is trained and verified. Then, the C which results in an SVM model with the highest validation accuracy and the kernel parameters selected by the separation index value form the parameter combination for the classification problem. The training time of the proposed method is composed of the calculation time of all separation index values and the SVM models training time for one kernel parameter combination and all C values.

4. Experiments and results

In our experiments, we use the data sets a1a–a5a and w1a–w5a [25] as the binary-class classification problems, and the modified dna, satimage and letter data sets originally from the Statlog collection [26] as the multi-class classification problems. Note that data sets a1a–a5a and w1a–w5a are actually random subsets of Adult and Web databases [25], so there are in fact two real binary-classification problems tested in the experiments. However, we apply the grid search method and the proposed method on each data set separately. Therefore, for each data set, the testing accuracy and the corresponding parameters are acquired as for an independent real problem. Since a series of data sets is formed by different sizes of data from a real problem, the training times of the same series of data can show the time complexity of the training processes. For the multi-class data sets, the original training data [26] are scaled to be in $[-1,1]$ and are partitioned into training and validation data sets [2]. Thus the multi-class data sets contain training, validation and testing data now. The specifications of these data sets are listed in Table 1.

We use LIBSVM [27] to train the SVM models. The sub-sampled binary-class data sets, the scaled and partitioned multi-class data sets and the software are downloaded from the LIBSVM web site. The downloaded binary-class data sets a1a–a5a and w1a–w5a contain only training and testing data. As mentioned in Section 3, k-fold cross validation can be used to acquire the validation accuracy for such data sets for general usage. A fivefold cross validation [4,6] method is used in our experiments when training the SVM models for choosing the kernel parameters for the binary-class data sets. With fivefold cross validation, a training data set is divided into five subsets of equal size. Each subset is used one time for validating the model trained with the rest four subsets. Therefore, for each data set and each parameter combination, five SVM models are trained with the five reduced, overlapped subsets to acquire the corresponding average validation accuracy. Since all the multi-class data sets we download contain training, validation and testing data, it is not necessary to apply the k-fold cross validation when training SVM with these data set. For the multi-class data sets, the general training process and the one-against-one method is used to solve the classification problems. The validation data are used in the training phase to validate the generalization ability of each SVM model which is trained with all the training data, and in the testing phase the testing data are used for evaluating the testing accuracy. When applying the proposed method on a multi-class case, the distance values of all pairs of classes are calculated and added together for each kernel parameter combination.

Table 1
The specifications of these data sets

Data set	# of classes	# of features	# of training data	Testing accuracy (%)[a]
Adult	2	123	32 562	89 [25]
a1a	2	123	1605	–
a2a	2	123	2265	–
a3a	2	123	3185	–
a4a	2	123	4781	–
a5a	2	123	6414	–
Web	2	300	49 749	96 [25]
w1a	2	300	2477	–
w2a	2	300	3407	–
w3a	2	300	4912	–
w4a	2	300	7366	–
w5a	2	300	9888	–
dna	3	180	1400	95.9 [26]
letter	26	16	10 500	93.6 [26]
satimage	6	36	3104	90.6 [26]

[a]The testing accuracy values are the highest ones reported in the source references.

Table 2
Testing accuracy (%) and training times (s) with the grid search method and the proposed indexed method

Data set	Grid search			Proposed method			
	Chosen $\langle C,\gamma\rangle$	Testing accuracy (%)	Training time (s)	Chosen $\langle C,\gamma\rangle$	Testing accuracy (%)	Training time (s)	Speed-up (times)
a1a	$\langle 2^2,2^{-5}\rangle$	84.40	3315.3	$\langle 2^2,2^{-4}\rangle$	83.89	81.3	40.8
a2a	$\langle 2^5,2^{-6}\rangle$	83.83	6535.3	$\langle 2^1,2^{-4}\rangle$	84.34	161.0	40.6
a3a	$\langle 2^5,2^{-7}\rangle$	84.51	12 717.1	$\langle 2^0,2^{-4}\rangle$	84.31	308.8	41.2
a4a	$\langle 2^5,2^{-8}\rangle$	84.62	28 924.6	$\langle 2^0,2^{-4}\rangle$	84.55	704.1	41.1
a5a	$\langle 2^5,2^{-9}\rangle$	84.45	51 736.8	$\langle 2^1,2^{-4}\rangle$	84.49	1259.0	41.1
w1a	$\langle 2^4,2^{-4}\rangle$	97.83	6458.9	$\langle 2^4,2^{-4}\rangle$	97.83	154.6	41.8
w2a	$\langle 2^4,2^{-5}\rangle$	98.12	12 627.5	$\langle 2^4,2^{-4}\rangle$	98.05	297.3	42.5
w3a	$\langle 2^7,2^{-7}\rangle$	98.30	24 874.2	$\langle 2^3,2^{-4}\rangle$	98.32	570.0	43.6
w4a	$\langle 2^5,2^{-6}\rangle$	98.48	74 027.5	$\langle 2^4,2^{-4}\rangle$	98.38	1234.5	60.0
w5a	$\langle 2^5,2^{-6}\rangle$	98.60	194 603.0	$\langle 2^2,2^{-4}\rangle$	98.53	2179.6	89.3
dna	$\langle 2^3,2^{-6}\rangle$	94.69	4096.3	$\langle 2^3,2^{-6}\rangle$	94.69	288.0	14.2
letter	$\langle 2^3,2^{2}\rangle$	97.00	327 668.6	$\langle 2^6,2^{-1}\rangle$	96.19	6406.5	51.1
satimage	$\langle 2^4,2^{0}\rangle$	90.40	10 039.9	$\langle 2^7,2^{-2}\rangle$	89.10	647.2	15.5

Table 3
Testing accuracy (%) and training times (s) with the grid search method and the proposed indexed method, using only onefold training process

Data set	Grid search			Proposed method			
	Chosen $\langle C,\gamma\rangle$	Testing accuracy (%)	Training time (s)	Chosen $\langle C,\gamma\rangle$	Testing accuracy (%)	Training time (s)	Speed-up (times)
a1a	$\langle 2^7,2^{-11}\rangle$	84.33	660.6	$\langle 2^1,2^{-4}\rangle$	83.73	34.8	19.0
a2a	$\langle 2^0,2^{-3}\rangle$	84.11	1291.3	$\langle 2^1,2^{-4}\rangle$	84.34	68.9	18.8
a3a	$\langle 2^5,2^{-7}\rangle$	84.51	2549.4	$\langle 2^1,2^{-4}\rangle$	84.32	134.3	19.0
a4a	$\langle 2^3,2^{-9}\rangle$	84.29	5762.2	$\langle 2^0,2^{-4}\rangle$	84.55	303.1	19.0
a5a	$\langle 2^7,2^{-7}\rangle$	84.16	10 235.9	$\langle 2^2,2^{-4}\rangle$	84.25	547.0	18.7
w1a	$\langle 2^4,2^{-4}\rangle$	97.83	1274.2	$\langle 2^4,2^{-4}\rangle$	97.83	78.0	16.3
w2a	$\langle 2^4,2^{-4}\rangle$	98.05	2503.5	$\langle 2^4,2^{-4}\rangle$	98.05	153.8	16.3
w3a	$\langle 2^7,2^{-7}\rangle$	98.30	4951.1	$\langle 2^0,2^{-4}\rangle$	97.73	301.9	16.4
w4a	$\langle 2^6,2^{-7}\rangle$	98.49	14 193.2	$\langle 2^2,2^{-4}\rangle$	98.37	674.4	21.0
w5a	$\langle 2^2,2^{-4}\rangle$	98.53	40 426.3	$\langle 2^2,2^{-4}\rangle$	98.53	1192.4	33.9

We use the RBF kernel with $d=2$, $\gamma \in [2^{-20}, 2^{-19}, \ldots, 2^{20}]$. C is set with the values $[2^{-7}, 2^{-6}, \ldots, 2^{7}]$. According to Refs. [2,17,23] and our observation, we assume the chosen values can cover a range of the parameter search space which lead to a high validation accuracy. As mentioned in Section 3, δ_i can be the separation indexes and can be calculated in the feature space. According to our tests, δ_{4F} (δ_4 in the feature space) performs well in most cases. Therefore, we only present the results of the training process in List 2 with δ_{4F} to show the proposed method works.

Table 2 shows the training time and testing accuracy of the SVM with the grid search method and the proposed method with δ_{4F}. For the proposed method, the training time is the sum of the time for calculating the δ_{4F} values for all kernel parameter combinations and the time for training SVMs for all C with the 'best' kernel parameter combination chosen according to the δ_{4F} values. According to the testing results, the SVMs trained by the proposed method with δ_{4F} perform about the same as those models chosen by grid search do when using RBF kernels. Meanwhile, as the kernel parameters are decided by calculating the corresponding distance indexes but not by training all the SVM models, the total training time includes only the computation time of the distance index and the SVM training time for deciding a proper C. Since it is not necessary to train models for the kernel parameters other than the ones chosen by the heuristics, the training time can be significantly shortened. As we use 41 candidates of γ in our experiments, the training process can be sped up from 14 to about 90 times. If a finer grid for searching the parameters is used, more candidates of the kernel parameters are required to be validated and the proposed method would speed-up the training process more.

The training time of the proposed method is composed of the index calculation time for the kernel parameters and the SVM model training time for choosing C. When k-fold cross validation is used, only the SVM model training time for choosing C is affected because δ_{4F} is the distance between two means and does not require a redundant validation and average process. It could be concerned that, for the binary-class data sets, since fivefold cross validation are used for training, more SVM models are trained and more training time are spent, while the separation index is calculated only once and thus the speed-up is enlarged. To see how the training time is affected by changing the fivefold validation to fivefold validation, we conduct the following experiments for the binary-class data sets with only onefold training process is involved. Table 3 lists the results when the grid search method and the proposed method are applied on the binary-class data sets with only one of the fivefold training process is involved. That is, here we assume that 20% of the downloaded training data of the binary-class data sets are validation data, and the rest 80% of data are used for training once. The general training and validation process is applied on each binary-class data set only one time for each parameter combination. It is the same way how we generate the experiment results of the multi-class data sets in Table 2, except that the validation data are randomly partitioned from the original training data by us now. As predicted, the training time of the grid search method is about five-time decreased in comparison with the fivefold cross validation case. The SVM model training time for choosing C of the proposed method is shorten, similarly. However, calculating δ_{4F} for 80% of training data would cost about 64% of time for calculating δ_{4F} for all the training data. As a result, calculating δ_{4F} in the current cases cost relatively more time than in the fivefold cases, and the training time speed-up in the current cases will not as high as in the fivefold cases. However, the proposed method can still speed-up the training process of the binary-class data sets from 16 to about 30 four times. Referring to Table 2, the multi-class cases are sped up from 14 to 50 one times by the proposed method. Without k-fold validation, the proposed method speeds up the binary and multi-class cases similarly. However, the parameters chosen by the grid search method without k-fold validation can be quite different to the parameters chosen with k-fold validation (see the chosen parameters of a1a in Tables 2 and 3). Also, the parameters chosen by the grid search method without k-fold validation of a series of data sets can be quite different (see the chosen parameters of a1a and a2a in Table 3). When there is no validation data available, using k-fold validation can acquire the average performance of k models trained with a parameter combination without over-fitting the training data. In addition, since all the k subsets are used as the validation data, the sub-sampling bias can be avoided. In order to acquire more stable results, the fivefold cross validation process on the binary-class data sets is used in the following experiments.

Since the a1a–a5a and w1a–w5a data sets are subsets of the Adult and Web data sets with different sizes, the stable speed-up ratio suggests that both computing δ_{4F} and training SVMs have the same time complexity. Considering the binary case, we only train 21 SVM models for choosing the penalty parameter C. This corresponds to about 5% sampling of the parameter search space. When using GA for searching the parameter combination, a larger initial population or more generations may be helpful to find the better parameter combinations, thus requiring a higher sampling ratio of the parameter search space.

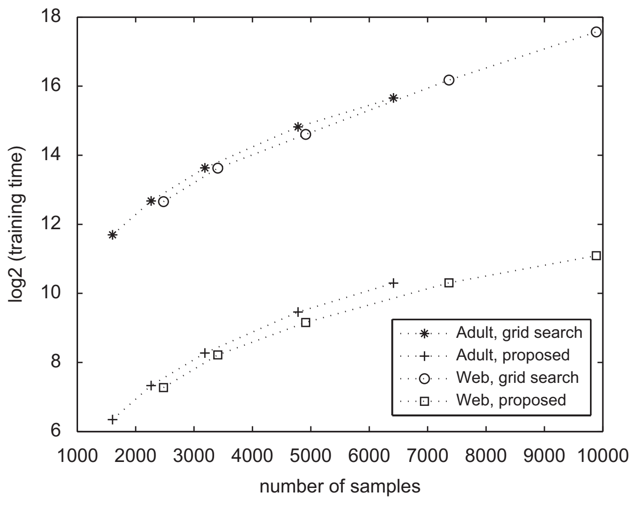


Fig. 2. Training time (in log2()) vs training sample size of the grid search method and the proposed method of the sub-sampled data sets of Adult and Web.

Fig. 2 presents the growing of training time with respect to the training sample size. As mentioned in Sections 1 and 3, training an SVM and calculating δ_{4F} both cost $O(l^2)$. That is, reducing the training sample size can significantly reduce the training time for both methods. However, Fig. 2 shows that the training times of the grid search method and the proposed method grow in a similar way, while the grid search method costs more time. Therefore, both methods can gain benefits on training time from a sub-sampling strategy, and the proposed method can still run faster than the grid search method on a sub-sampled data set. This results in a further speed-up when practically using the proposed method with the sub-sampling technique. As Table 2 shows, the proposed method chooses similar kernel parameters for a series of data from the same problem. That is, applying the proposed method on a properly sub-sampled data set can find the kernel parameters similar to the ones with the original data set. If the data set is sub-sampled with a size $\sqrt{l}$, calculating δ_{4F} will cost $O(\sqrt{l}^2) = O(l)$. This technique is very feasible for calculating the value of noise-resistant distance indexes such as $\delta_{3F} - \delta_{5F}$, because such indexes are the means of distance values or distance between classes means and are less affected by the sample size.

As a reference, we also use the one-against-rest method and the polynomial kernel in the following cases. For the three multi-class problems, we use RBF kernels and the one-against-rest method to see how the proposed method performs with different training strategies with γ and C being the same ones listed above. The results are shown in Table 4. In this case, the SVMs generated by the proposed method performs as good as the models generated by the grid search method in testing accuracy. Although the proposed method still speeds up the training process, the training time speed-up ratio decreases for every data set. This may be due to the 'rest' classes contain overlap data, thus existing many redundant computations.

We also apply polynomial kernels on the a5a, w5a and the three multi-class data sets (using the one-against-one strategy). The parameters are $\gamma = 1$, $r \in [0, 2^0, \ldots, 2^3]$, $d \in [2^0, \ldots, 2^2]$. The results are shown in Table 5. As we use 15 parameter combinations, we expect a 15 time speed-up for these cases. However, except the a5a case (an extra high speed-up with testing accuracy degradation), the speed-up is 5–10 times. The testing accuracy degradation of a5a shows that the proposed method is still a heuristics. With proper kernels and

Table 4
Testing accuracy (%) and training time (s) with the one-against-rest training strategy

Data set	Grid search			Proposed method			
	Chosen $\langle C,\gamma\rangle$	Testing accuracy (%)	Training time (s)	Chosen $\langle C,\gamma\rangle$	Testing accuracy (%)	Training time (s)	Speed-up (times)
dna	$\langle 2^5, 2^{-7}\rangle$	93.0	7080.5	$\langle 2^3, 2^{-6}\rangle$	93.51	604.9	11.7
letter	$\langle 2^4, 2^1\rangle$	94.56	1 497 206.6	$\langle 2^6, 2^{-1}\rangle$	93.36	87567.7	17.1
satimage	$\langle 2^3, 2^0\rangle$	88.75	18 848.1	$\langle 2^5, 2^{-2}\rangle$	88.55	2499.4	7.5

Table 5
Testing accuracy (%) and training time (s) with the polynomial kernel

Data set	Grid search			Proposed method			
	Chosen $\langle C,r,d\rangle$	Testing accuracy (%)	Training time (s)	Chosen $\langle C,r,d\rangle$	Testing accuracy (%)	Training time (s)	Speed-up (times)
a5a	$\langle 2^{-3}, 2^0, 2^0\rangle$	84.48	425 756.7	$\langle 2^{-7}, 2^3, 2^2\rangle$	78.82	1698.3	250.7
w5a	$\langle 2^0, 2^0, 2^0\rangle$	98.51	4954.3	$\langle 2^{-7}, 2^3, 2^2\rangle$	97.8	473.1	10.5
dna	$\langle 2^{-7}, 2^2, 2^1\rangle$	94.35	728.4	$\langle 2^{-7}, 2^3, 2^2\rangle$	92.41	128.6	5.7
letter	$\langle 2^{-2}, 2^0, 2^2\rangle$	95.32	10 070.3	$\langle 2^{-5}, 2^3, 2^2\rangle$	94.82	939.7	10.7
satimage	$\langle 2^{-6}, 2^2, 2^2\rangle$	88.9	452.4	$\langle 2^{-5}, 2^3, 2^2\rangle$	88.65	114.0	4.0

Table 6
Comparing the training time (s) of different training tools in a reduced parameter search space

Data set	LIBSVM			SVMtorch		
	Grid search	Proposed method		Grid search	Proposed method	
	Training time	Training time	Speed-up (times)	Training time	Training time	Speed-up (times)
a1a	750.5	48.7	15.4	924.9	67.5	13.7
a2a	1462.5	97.4	15.0	1694.8	125.9	13.5
a3a	2841.1	182.7	15.6	3067.5	224.2	13.7
a4a	6410.0	421.9	15.2	7241.5	476.7	15.2
a5a	11473.6	741.4	15.5	17963.4	821.1	21.9
w1a	1688.9	71.9	23.5	2441.7	105.7	23.1
w2a	3312.3	133.3	24.8	4477.9	182.0	24.6
w3a	6472.3	241.7	26.8	8300.0	316.2	26.2
w4a	17704.9	496.5	35.7	28457.6	609.4	46.7
w5a	44617.8	851.4	52.4	55850.6	1015.6	55.0
dna (one-against-one)	1154.7	125.2	9.2	615.4	98.6	6.2
dna (one-against-rest)	2028.9	239.4	8.5	1049.7	192.1	5.5
letter (one-against-one)	30446.6	2682.7	11.3	36402.5	3536.1	10.3
letter (one-against-rest)	98956.6	20925.9	4.7	61960.8	21215.7	2.9
satimage (one-against-one)	1192.4	187.9	6.3	1480.5	236.2	6.3
satimage (one-against-rest)	2478.7	645.3	3.8	2113.4	672.1	3.1

distance measures, the proposed method is possible to give a fast and good guess of kernel parameter combinations.

As mentioned in Section 3, the training time of the proposed method is composed of the separation index calculation time and the SVM training time. The proposed method can save the total training time because only one kernel parameter combination is used for training the SVM models. Therefore, if the kernel parameter search space is reduced, the speed-up of the proposed method will decrease. For example, if RBF kernel with $\gamma \in [2^{-5}, 2^{-4}, \dots, 2^{5}]$ are chosen according to some a priori knowledge or pre-processing strategies, the proposed method saves the SVM training time for 10γ but not for 40γ, which implies that the speed-up is possibly to be about 10 times but not 40 times. Table 6 shows the training time and speed-up when only $\gamma \in [2^{-5}, 2^{-4}, \dots, 2^{5}]$ are used. It is the central area of the search space we previously used for RBF kernel, and γ in this area with certain C often leads to an SVM model with a high validation accuracy. We also use another SVM training tool, named SVMtorch, to see whether different training tools lead to significant change of results. SVMtorch is reported to be fast for high input dimension and large sample size problems [28]. It is available at Ref. [29]. According to our observation, SVMtorch runs faster than LIBSVM for some parameter combinations, and is slower for some other parameter combinations in our experiments. Table 6 do not show significant difference between the two tools for the speed-up of the proposed method. This may be due to the experiment design and the data sets. As expected, the values of speed-up decrease when the search space is reduced. However, the results in Table 6 do not take the cost of reducing the search space into account. Actually, the proposed method itself can be considered as a method that reduces the search space into one kernel parameter combination. The selected kernel parameter combination then can be the starting point or the center for searching good kernel parameters. That is, the exhaustive search of parameters can be performed in the priority area indicated by the proposed method first. The results also imply that the proposed method is suitable for a larger kernel search space, such as a finer grid for exhaustive searching for better kernel parameters.

5. Conclusions

We propose calculating the inter-cluster distance in the feature space to help determine the kernel parameters for training the SVM models. The kernel parameters chosen by the proposed method can result in SVM models that perform as good as the models chosen by grid search method in testing accuracy. Calculating the distance is a simple and a non-iterated process. As compared with training an SVM, which generally costs $O(l^2)$ training time with non-deterministic iterations, the proposed index can be calculated much faster than training SVMs. Thus, the training time for the proposed method to obtain SVM models can be significantly shortened. With properly chosen distance indexes, the proposed method performs stable with different sample sizes of the same problem. As a result, the time complexity $O(l^2)$ of calculating the index is possible to be further reduced by the sub-sampling strategy in practical usage, and thus the proposed method can work even the data size is large. Currently the penalty parameter C is not incorporated into the proposed strategies, or else the training time might be further minimized. The proposed indexes can also be used to suggest a starting point or a searching center of the grid search process.

References

[1] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[2] C.-W. Hsu, C.-J. Lin, A comparison of methods for multiclass support vector machines, IEEE Trans. Neural Networks 13 (2) (2002) 415–425.
[3] N. Ancona, G. Cicirelli, E. Stella, A. Distante, Object detection in images: run-time complexity and parameter selection of support vector machines, 16th International Conference on Pattern Recognition, vol. 2, 11–15 August, 2002, pp. 426–429.
[4] S.A. Rojas, D. Fernandez-Reyes, Adapting multiple kernel parameters for support vector machines using genetic algorithms, The 2005 IEEE Congress on Evolutionary Computation, vol. 1, 02–05 September, 2005, pp. 626–631.
[5] X. Liang, F. Liu, Choosing multiple parameters for SVM based on genetic algorithm, 6th International Conference on Signal Processing, vol. 1, 26–30 August, 2002, pp. 117–119.
[6] H.-J. Liu, Y.-N. Wang, X.-F. Lu, A method to choose kernel function and its parameters for support vector machines, in: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, vol. 7, 18–21 August, 2005, pp. 4277–4280.
[7] S. Liu, C.-Y. Jia, H. Ma, A new weighted support vector machine with GA-based parameter selection, in: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, vol. 7, 18–21 August, 2005, pp. 4351–4355.
[8] A.T. Quang, Q.-L. Zhang, X. Li, Evolving support vector machine parameters, in: Proceedings of 2002 International Conference on Machine Learning and Cybernetics, vol 4–5 November, 2002, pp. 548–551.
[9] F. Imbault, K. Lebart, A stochastic optimization approach for parameter tuning of support vector machines, in: Proceedings of 17th International Conference on Pattern Recognition, vol. 4, 23–26 August, 2004, pp. 597–600.
[10] J.C. Platt, Sequential minimal optimization: a fast algorithm for training support vector machines, Microsoft Research, Technical Report MST-TR-98-14, 1998.
[11] K.-I. Maruyama, M. Maruyama, H. Miyao, Y. Nakano, A method to make multiple hypotheses with high cumulative recognition rate using SVMs, Pattern Recognition 37 (2004) 241–251.
[12] D. Hush, C. Scovel, Polynomial-time decomposition algorithms for support vector machines, Mach. Learn. 51 (2003) 51V71.

[13] R. Debnath, H. Takahashi, An efficient method for tuning kernel parameter of the support vector machine, IEEE International Symposium on Communications and Information Technology, vol. 2, 26–29 October, 2004, pp. 1023–1028.
[14] L.-P. Bi, H. Huang, Z.-Y. Zheng, H.-T. Song, New heuristic for determination Gaussian kernel's parameter, in: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, vol. 7, 18–21 August, 2005, pp. 4299–4304.
[15] S.S. Keerthi, C.-J. Lin, Asymptotic behaviors of support vector machines with Gaussian kernel, Neural Comput. 15 (7) (2003) 1667–1689.
[16] H.-T. Lin, C.-J. Lin, A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods, Technical Report, Department of Computer Science and Information Engineering, National Taiwan University.
[17] C.-W. Hsu, C.-C. Chang, C.-J. Lin, A practical guide to support vector classification. [Online] Available from World Wide Web: ⟨http://www.csie.ntu.edu.tw/∼cjlin/libsvm⟩.
[18] L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard, V. Vapnik, Comparison of classifier methods: a case study in handwritten digit recognition, in: Proceedings of International Conference on Pattern Recognition, 1994, pp. 77–87.
[19] U. Kreßel, Pairwise classification and support vector machines, in: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel Methods—Support Vector Learning, MIT Press, Cambridge, MA, 1999.
[20] J.C. Platt, N. Cristianini, J. Show-Tayler, Large margin DAG's for multiclass classification, Advances in Neural Information Processing Systems, vol. 12, MIT Press, Cambridge, MA, 2000, pp. 547–553.
[21] F. Takahashi, S. Abe, Optimizing directed acyclic graph support vector machines, in: Proceedings of the Artificial Neural Networks in Pattern Recognition (ANNPR 2003), September 2003, pp. 166–170.
[22] T. Phetkaew, B. Kijsirikul, W. Rivepiboon, Reordering adaptive directed acyclic graphs: an improved algorithm for multiclass support vector machines, in: Proceedings of the International Joint Conference on Neural Networks (IJCNN 2003), vol. 2, 2003, pp. 1605–1610.
[23] K.-P. Wu, S.-D. Wang, Choosing the kernel parameters of support vector machines according to the inter-cluster distance, in: Proceedings of the International Joint Conference on Neural Networks (IJCNN 2006), 2006.
[24] C. Bezdek, N.R. Pal, Some new indexes of cluster validity, IEEE Trans. Syst. Man Cybern. Part B Cybern. 28 (3) (1998) 301–315.
[25] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, in: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel Methods—Support Vector Learning, MIT Press, Cambridge, MA, 1999.
[26] D. Michie, D.J. Spiegelhalter, C.C. Taylor, Machine learning, neural and statistical classification, 1994. [Online] Available: ⟨ftp.ncc.up.pt/pub/statlog/⟩.
[27] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines. [Online] Available from World Wide Web: ⟨http://www.csie.ntu.edu.tw/∼cjlin/libsvm⟩.
[28] J.X. Dong, A. Krzyzak, C.Y. Suen, Fast SVM training algorithm with decomposition on very large datasets, IEEE Trans. Pattern Anal. Mach. Intell. 27 (4) (2005) 603 –618.
[29] R. Collobert, S. Bengio, J. Mariethoz, Torch. [Online] Available from World Wide Web: ⟨http://www.torch.ch/⟩.

About the Author—SHENG-DE WANG was born in Taiwan in 1957. He received the B.S. degree from National Tsing Hua University, Hsinchu, Taiwan, in 1980, and the M.S. and the Ph.D. degrees in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1982 and 1986, respectively.
Since 1986 he has been on the Faculty of the Department of Electrical Engineering at National Taiwan University, Taipei, Taiwan, where he is currently a professor. From 1995 to 2001, he also served as the Director of Computer Operating Group of Computer and Information Network Center, National Taiwan University. He was a visiting scholar in Department of Electrical Engineering, University of Washington, Seattle during the academic year of 1998–1999. From 2001 to 2003, He has been served as the Department Chair of Department of Electrical Engineering, National Chi Nan University, Puli, Taiwan for the 2-year appointment. His research interests include soft computing, embedded systems, and intelligent systems.
Dr. Wang is a member of the Association for Computing Machinery and IEEE Computer Societies. He is also a member of Phi Tau Phi Honor society.

About the Author—KUO-PING WU is currently a Ph.D. student in Department of Electrical Engineering, National Taiwan University. His research interests include support vector machines, pattern classifications and image processing techniques.