PERGAMON

Pattern Recognition 36 (2003) 2757–2767

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Constructing support vector machine ensemble

Hyun-Chul Kim, Shaoning Pang, Hong-Mo Je, Daijin Kim*, Sung Yang Bang

Department of Computer Science and Engineering, Pohang University of Science and Technology, San 31, Hyoja-Dong, Nam-Gu, Pohang 690-784, South Korea

Received 13 June 2002; received in revised form 29 April 2003; accepted 29 April 2003

Abstract

Even the support vector machine (SVM) has been proposed to provide a good generalization performance, the classification result of the practically implemented SVM is often far from the theoretically expected level because their implementations are based on the approximated algorithms due to the high complexity of time and space. To improve the limited classification performance of the real SVM, we propose to use the SVM ensemble with bagging (bootstrap aggregating) or boosting. In bagging, each individual SVM is trained independently using the randomly chosen training samples via a bootstrap technique. In boosting, each individual SVM is trained using the training samples chosen according to the sample's probability distribution that is updated in proportional to the errorness of the sample. In both bagging and boosting, the trained individual SVMs are aggregated to make a collective decision in several ways such as the majority voting, least-squares estimation-based weighting, and the double-layer hierarchical combining. Various simulation results for the IRIS data classification and the hand-written digit recognition, and the fraud detection show that the proposed SVM ensemble with bagging or boosting outperforms a single SVM in terms of classification accuracy greatly.

Keywords: SVM; SVM ensemble; Bagging; Boosting; Iris and hand-written digit recognition; Fraud detection

1. Introduction

The support vector machine (SVM) is a new and promising classification and regression technique proposed by Vapnik and his group at AT&T Bell Laboratories [1]. The SVM learns a separating hyperplane to maximize the margin and to produce a good generalization ability [2]. Recent theoretical research work has solved the existing difficulties of using the SVM in practical applications [3,4]. By now, it has been successfully applied in many areas such as the face detection, the hand-writing digital character recognition, the data mining, etc.

However, the SVM has two drawbacks. First, since it is originally a model for the binary-class classification, we should use a combination of SVMs for the multi-class classification. Methods for combining SVMs for the multi-class classification has been proposed [6,7], but the performance does not seem to improve as much as in the binary classification. Second, since learning of the SVM is a very time consuming for a large scale of data, we should use some approximate algorithms [2]. Using the approximate algorithms can reduce the computation time, but degrade the classification performance.

To overcome the above drawbacks, we propose to use the SVM ensemble. We expect that the SVM ensemble can improve the classification performance greatly than using a single SVM by the following fact. Each individual SVM has been trained independently from the randomly chosen training samples and the correctly classified area in the space of data samples of each SVM becomes limited to a certain area. We can imagine that a combination of several SVMs

* Corresponding author. Tel.: +82-562-279-2249; fax: +82-562-279-2299.

E-mail addresses: grass@postech.ac.kr (H.-C. Kim), snpang@postech.ac.kr (Shaoning Pang), invu71@postech.ac.kr (H.-M. Je), dkim@postech.ac.kr (D. Kim), sybang@postech.ac.kr (Sung Yang Bang).

0031-3203/$30.00
doi:10.1016/S0031-3203(03)00175-4

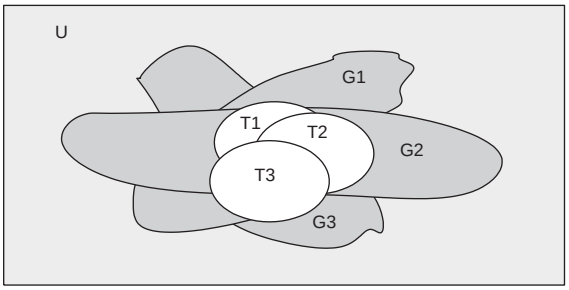

Fig. 1. The area of being correctly classified is expanded by the ensemble.

will expand the correctly classified area incrementally. This implies the improvement of classification performance by using the SVM ensemble. Likewise, we also expect that the SVM ensemble will improve the classification performance in case of the multi-class classification.

Fig. 1 illustrates how a combination of different SVM classifiers expands the region of test sample resulting in the correct classifications. Let U and T_i be a set of all possible examples and a set of training examples for the ith SVM classifier, respectively. Assume that K different SVM classifiers has been training independently using different training samples. Then, each SVM classifiers provide different generalization performance. As we can see in Fig. 1, the region of test examples that results in correct classification by each SVM classifier can be very different each other (see $G_1, G_2, \ldots, G_K$ in Fig. 1) and the region covered by the SVM ensemble is extended further. This expansion implies that the generalization ability will be improved further by the SVM ensemble since an SVM classifier with incorrect classification over some test examples can be compensated by another SVM classifier with correct classification over the same test examples.

The idea of the SVM ensemble has been proposed in Ref. [8]. They used the boosting technique to train each individual SVM and took another SVM for combining several SVMs. In this paper, we propose to use the SVM ensemble based on the bagging and boosting techniques. In bootstrapping (bagging), each individual SVM is trained over the randomly chosen training samples via a bootstrap technique. In boosting, the training samples for each individual SVM is chosen according to updating probability distribution (related to error) for samples. Then, the independently trained several SVMs are aggregated in various ways such as the majority voting, the LSE-based weighting, and the double-layer hierarchical combining.

This paper is organized as follows. Section 2 describes the theoretical background of the SVM. Section 3 describes the SVM ensembles, the bagging and boosting method and three different aggregation methods. Section 4 shows the simulation results when the proposed SVM ensembles are applied to the classification problems such as the IRIS data classification and the hand-written digit recognition, and the fraud detection. Finally, a conclusion is drawn.

2. Support Vector Machines

In theory, the SVM classification can be traced back to the classical structural risk minimization (SRM) approach, which determines the classification decision function by minimizing the empirical risk as

$$R = \frac{1}{l}\sum_{i=1}^{L} |f(\mathbf{x}_i) - y_i|, \tag{1}$$

where L and f are the size of examples and the classification decision function, respectively. For SVM, determining an optimal separating hyperplane that gives low generalization error is the primary concern. Usually, the classification decision function in the linearly separable problem is represented by

$$f_{\mathbf{w},b} = \mathrm{sign}(\mathbf{w}\cdot\mathbf{x} + b). \tag{2}$$

In SVM, the optimal separating hyperplane is determined by giving the largest margin of separation between different classes. This optimal hyperplane bisects the shortest line between the convex hulls of the two classes. The optimal hyperplane is required to satisfy the following constrained minimization as

$$\mathrm{Min}\ \tfrac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w},$$

$$y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geqslant 1. \tag{3}$$

For the linearly non-separable case, the minimization problem needs to be modified to allow the misclassified data points. This modification results in a soft margin classifier that allows but penalizes errors by introducing a new set of variables $\xi_{i=1}^{l}$ as the measurement of violation of the constraints:

$$\mathrm{Min}\ \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\left(\sum_{i=1}^{L}\xi_i\right)^k,$$

$$y_i(\mathbf{w}^{\mathrm{T}}\varphi(\mathbf{x}_i) + b) \geqslant 1 - \xi_i, \tag{4}$$

where C and k are used to weight the penalizing variables ξ_i, $\varphi(\cdot)$ is a nonlinear function which maps the input space into a higher dimensional space. Minimizing the first term in Eq. (4) is corresponding to minimizing the VC-dimension of the learning machine and minimizing the second term in Eq. (4) controls the empirical risk. Therefore, in order to solve problem Eq. (4), we need to construct a set of functions, and implement the classical risk minimization on the set of functions. Here, a Lagrangian method is used to solve the above problem. Then, Eq. (4) can be written as

$$\mathrm{Max}\ F(\Lambda) = \Lambda\cdot\mathbf{1} - \tfrac{1}{2}\Lambda\cdot D\cdot\Lambda,$$

$$\Lambda\cdot y = 0, \quad \Lambda \leqslant C, \quad \Lambda > 0, \tag{5}$$

where $\Lambda=(\lambda_1,\ldots,\lambda_L)$, $D=y_iy_jx_i\cdot x_j$. For the binary classification, the decision function equation (2) can be rewritten as

$$f(\mathbf{x})=\operatorname{sign}\left(\sum_{i=1}^{l}y_i\lambda_i^*(,x\cdot\mathbf{x})+b^*\right). \tag{6}$$

For the multi-class classification, we can extend the SVM in the following two ways. One method is called the "one-against-all" method [5,6], where we have as many SVMs as the number of classes. The ith SVM is trained from the training samples where some examples contained in the ith class have "+1" labels, and other examples contained in the other classes have "−1" labels. Then, Eq. (4) is modified into

$$\begin{aligned}
&\text{Min}: F(\Lambda)=\frac{1}{2}(\mathbf{w}_i)^{\mathrm{T}}\mathbf{w}_i+C\left(\sum_{t=1}^{L}(\xi^i)_t\right)^k,\\
&y_i((\mathbf{w}_i)^{\mathrm{T}}\varphi(\mathbf{x}_t)+b_i)\geqslant(1-(\xi^i)_t \quad \text{if } x_i\in C_i,\\
&y_i((\mathbf{w}_i)^{\mathrm{T}}\varphi(\mathbf{x}_t)+b_i)<(1-(\xi^i)_t \quad \text{if } x_i\in \bar{C}_i,\\
&(\xi^i)_t>0, \quad i=1,\ldots,L
\end{aligned} \tag{7}$$

and the decision function of Eq. (7) becomes

$$f(\mathbf{x})=\operatorname{sign}\left(\operatorname*{Max}_{j\in1,2,\ldots,C}((\mathbf{w}^j)^{\mathrm{T}}\cdot\varphi(\mathbf{x}_j)+b_j)\right), \tag{8}$$

where C is the number of the classes.

Another method is called the one-against-one method [5,7]. When the number of classes is C, this method constructs $C(C-1)/2$ SVM classifiers. The ijth SVM is trained from the training samples where some examples contained in the ith class have "+1" labels and other examples contained in the jth class have "−1" labels. Then, Eq. (4) is modified into

$$\begin{aligned}
&\text{Min}\,F(\wedge)=\frac{1}{2}(\mathbf{w}_{ij})^{\mathrm{T}}\mathbf{w}_{ij}+C\left(\sum_{t=1}^{L}(\xi^{ij})_t\right)^k,\\
&y_t((\mathbf{w}_{ij})^{\mathrm{T}}\varphi(\mathbf{x}_t)+b_{ij})\geqslant(1-(\xi^{ij})_t \quad \text{if } x_t\in C_i,\\
&y_t((\mathbf{w}_{ij})^{\mathrm{T}}\varphi(\mathbf{x}_t)+b_{ij})<(1-(\xi^{ij})_t \quad \text{if } x_t\in C_j,\\
&(\xi^{ij})_t>0, \quad t=1,\ldots,L.
\end{aligned} \tag{9}$$

The class decision in this type of multi-class classifier can be performed in the following two ways. The first decision is based on the "Max Wins" voting strategy, in which $C(C-1)/2$ binary SVM classifiers will vote for each class, and the winner class will be the class having the maximum votes. The second method uses the tournament match, which reduces the classification time to the log scale.

3. SVM ensemble

An ensemble of classifiers is a collection of several classifiers whose individual decisions are combined in some way to classify the test examples [9,10]. It is known that an ensemble often shows much better performance than the individual classifiers that make it up. Hansen et. al. [14] shows why the ensemble shows better performance than individual classifiers as follows. Assume that there are an ensemble of n classifiers: $\{f_1,f_2,\ldots,f_n\}$ and consider a test data $\mathbf{x}$. If all the classifiers are identical, they are wrong at the same data, where an ensemble will show the same performance as individual classifiers. However, if classifiers are different and their errors are uncorrelated, then when $f_i(\mathbf{x})$ is wrong, most of other classifiers except for $f_i(\mathbf{x})$ may be correct. Then, the result of majority voting can be correct. More precisely, if the error of individual classifier is $p<\frac{1}{2}$ and the errors are independent, then the probability p_E that the result of majority voting is incorrect is $\sum_{k=\lceil n/2\rceil}^{n}p^k(1-p)^{(n-k)}$ $(<\sum_{k=\lceil n/2\rceil}^{n}(\frac{1}{2})^k(\frac{1}{2})^{(n-k)}=\sum_{k=\lceil n/2\rceil}^{n}(\frac{1}{2})^n)$. When the size of classifiers n is large, the probability p_E becomes very small.

The SVM has been known to show a good generalization performance and is easy to learn exact parameters for the global optimum [2]. Because of these advantages, their ensemble may not be considered as a method for improving the classification performance greatly. However, since the practical SVM has been implemented using the approximated algorithms in order to reduce the computation complexity of time and space, a single SVM may not learn exact parameters for the global optimum. Sometimes, the support vectors obtained from the learning is not sufficient to classify all unknown test examples completely. So, we cannot guarantee that a single SVM always provides the global optimal classification performance over all test examples.

To overcome this limitation, we propose to use an ensemble of support vector machines. Similar arguments mentioned above about the general ensemble of classifiers can also be applied to the ensemble of support vector machines. Fig. 2 shows a general architecture of the proposed SVM ensemble. During the training phase, each individual SVM is trained independently by its own replicated training data set via a bootstrap method explained in Section 3.1. All constituent SVMs will be aggregated by various combination strategies explained in Section 3.2. During the testing phase, a test example is applied to all SVMs simultaneously and a collective decision is obtained based on the aggregation strategy.

On the other hand, the advantage of using the SVM ensemble over a single SVM can be achieved equally in the case of multi-class classification. Since the SVM is originally a binary classifier, many SVMs should be combined for the multi-class classification as mentioned in 2.2. The SVM classifier for the multi-class classification does not show as good performance as that for the binary-class classification. So, we can also improve the classification performance in the multi-class classification by taking the SVM ensemble where each SVM classifier is designed for the multi-class classification.

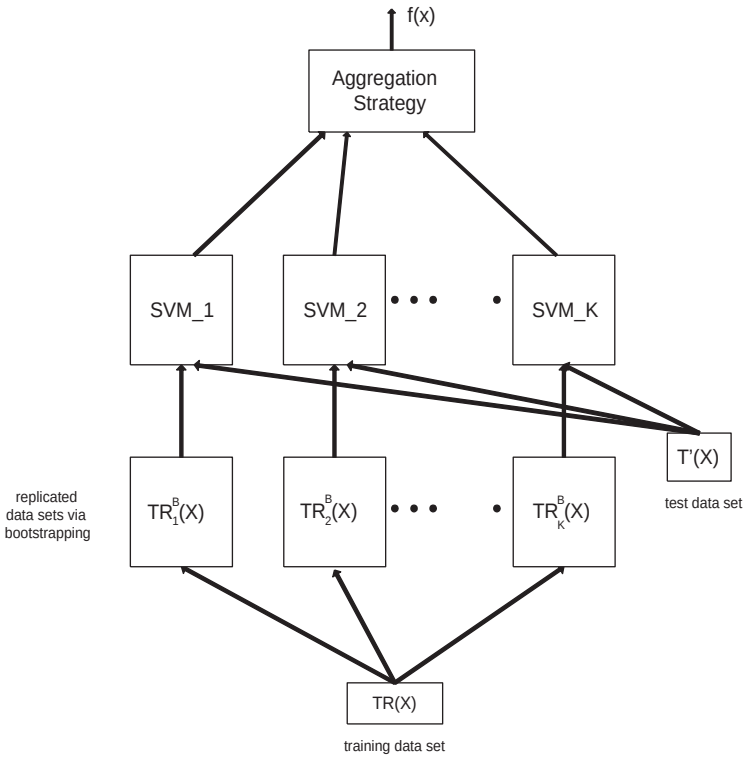

Fig. 2. A general architecture of the SVM ensemble.

3.1. Methods for constructing the SVM ensemble

Many methods for constructing an ensemble of classifiers have been developed. The most important thing in constructing the SVM ensemble is that each individual SVM becomes different with another SVM as much as possible. This requirement can be met by using different training sets for different SVMs. Some methods for selecting the training samples are bagging, boosting, randomization, stacking and dagging [11–13]. Among them, we put focus on the representative methods such as bagging and boosting.

3.1.1. Bagging

First, we explain a bagging technique [15] to construct the SVM ensemble. In bagging, several SVMs are trained independently via a bootstrap method and then they are aggregated via an appropriate combination technique. Usually, we have a single training set $TR = \{(\mathbf{x}_i; y_i)|i = 1, 2, \ldots, l\}$. But we need K training samples sets to construct the SVM ensemble with K independent SVMs. From the statistical fact, we need to make the training sample sets different as much as possible in order to obtain higher improvement of the aggregation result. For doing this, we often use the bootstrap technique as follows.

Bootstrapping builds K replicate training data sets $\{TR_{bootstra\,p_k}|k = 1, 2, \ldots, K\}$ by randomly re-sampling, but with replacement, from the given training data set TR repeatedly. Each example $\mathbf{x}_i$ in the given training set TR may appear repeated times or not at all in any particular replicate training data set. Each replicate training set will be used to train a certain SVM.

3.1.2. Boosting

The representative boosting algorithm is the AdaBoost algorithm [16]. Like bagging, each SVM is also trained using a different training set. But, the selection scheme of training samples in the AdaBoost method is quite different from the bagging method by the following. Initially, we have a training set $TR = \{(\mathbf{x}_i; y_i)|i = 1, 2, \ldots, l\}$ consisting of l whole samples and each sample in the TR is assigned to have the same value of weight $p_0(\mathbf{x}_i) = 1/l$. For training the kth SVM classifier, we build a set of training samples $TR_{boost_k} = \{(\mathbf{x}_i; y_i)|i = 1, 2, \ldots, l'\}$ that is obtained by selecting $l'(< l)$ samples among the whole data set TR according to the weight values $p_{k-1}(\mathbf{x}_i)$ at the $(k-1)$th iteration. This training samples is used for training the kth SVM classifier. Then, we evaluate the classification performance of the kth trained SVM classifier using the whole training sample TR as follows. We obtain the updated weight values $p_k(\mathbf{x}_i)$ of the training samples in TR based on the errorness of the training samples as follows. The weight values of the incorrectly classified samples are increased but the weight values of the

Input:
A set TR of l labelled examples: $S = \{(\mathbf{x}_i; y_i), i = 1, 2, \ldots, l\}$,
Labels $y_i \in Y = \{1, \ldots, C\}$.
$p_0(\mathbf{x}_i) := 1/l$.
for $k = 1$ **to** K
Build $TR_{boost_k} = \{(\mathbf{x}_i; y_i) | i = 1, 2, \ldots, l\prime\}$ based on the $p_{k-1}(\mathbf{x}_i)$.
Train the kth SVM h_k using TR_{boost_k}.
$\epsilon_k := \sum_{i=1}^{l} p_l(i) | \{i | h_k(\mathbf{x}_i) \neq y_i\}|$.
$\alpha_k := \frac{1}{2} ln(\frac{\epsilon_k}{1-\epsilon_k})$.
for $i = 1$ **to** l

$$p_{k+1}(\mathbf{x}_i) = \frac{p_k(\mathbf{x}_i)}{Z_k} \times \begin{cases} \exp(-\alpha_k) & \text{if } h_k(\mathbf{x}_i) = y_i, \\ \exp(\alpha_k) & \text{if } h_k(\mathbf{x}_i) \neq y_i. \end{cases}$$

where Z_k is a normalization factor to make $\sum_{i=1}^{l} p_{k+1}(\mathbf{x}_i) = 1$.
end
end

Fig. 3. The AdaBoost algorithm.

correctly classified samples are decreased. This implies that the samples which are hard to classify are selected more frequently. This updated weight values will be used for building the training samples $TR_{boost_{k+1}} = \{(\mathbf{x}_i; y_i) | i = 1, 2, \ldots, l'\}$ of the $(k + 1)$th SVM classifier. This sampling procedure will be repeated until K training samples set has been built for the Kth SVM classifier. Fig. 3 shows the pseudo-code of the used AdaBoost algorithm.

3.2. Methods for aggregating SVMs

After training, we need to aggregate several independently trained SVMs in an appropriate combination manner. We consider two types of combination techniques such as the linear and nonlinear combination method. The linear combination method, as a linear combination of several SVMs, includes the majority voting and the LSE-based weighting. The majority voting and the LSE-based weighting are often used for the bagging and the boosting, respectively. A nonlinear method, as a nonlinear combination of several SVMs, includes the double-layer hierarchical combining that use another upper-layer SVM to combine several lower-layer SVMs.

3.2.1. Majority voting

Majority voting is the simplest method for combining several SVMs. Let $f_k(k = 1, 2, \ldots, K)$ be a decision function of the kth SVM in the SVM ensemble and C_j $(j = 1, 2, \ldots, C)$ denote a label of the jth class. Then, let $N_j = \#\{k | f_k(\mathbf{x}) = C_j\}$, i.e. the number of SVMs whose decisions are known to the jth class. Then, the final decision of the SVM ensemble $f_{mv}(\mathbf{x})$ for a given test vector $\mathbf{x}$ due to the majority voting is determined by

$$f_{mv}(\mathbf{x}) = \arg\max_j N_j. \tag{10}$$

3.2.2. The LSE-based weighting

The LSE-based weighting treats several SVMs in the SVM ensemble with different weights. Often, the weights of several SVMs are determined in proportional to their accuracies of classifications [17]. Here, we propose to learn the weights using the LSE method as follows.

Let $f_k(k = 1, 2, \ldots, K)$ be a decision function of the kth SVM in the SVM ensemble that is trained by a replicate training data set $Thau_k^{\beta} = \{(\mathbf{x}_i'; y_i') | i = 1, 2, \ldots, L\}$. The weight vector $\mathbf{w}$ can be obtained by $\mathbf{w}_E = \mathbf{A}^{-1}\mathbf{y}$, where $\mathbf{A} = (f_i(\mathbf{x}_j))_{K \times L}$, and $\mathbf{y} = (y_j)_{1 \times L}$. Then, the final decision of the SVM ensemble $f_{mv}(\mathbf{x})$ for a given test vector $\mathbf{x}$ due to the LSE-based weighting is determined by

$$f_{LSE}(\mathbf{x}) = \text{sign}(\mathbf{w} \cdot [(f_i(\mathbf{x}))_{K \times 1}]). \tag{11}$$

3.2.3. The double-layer hierarchical combining

We can use another SVM to aggregate the outputs of several SVMs in the SVM ensemble. So, this combination consists of a double layer of SVMs hierarchically where the outputs of several SVMs in the lower layer feed into a super SVM in the upper layer. This type of combination looks similar of the mixture of experts introduced by Jordan et al. [18] in that both use hierarchical structures.

Let $f_k(k = 1, 2, \ldots, K)$ be a decision function of the kth SVM in the SVM ensemble and F be a decision function of the super SVM in the upper layer. Then, the final decision of the SVM ensemble $f_{SVM}(\mathbf{x})$ for a given test vector $\mathbf{x}$ due to the double-layer hierarchical combining is determined by

$$f_{SVM}(\mathbf{x}) = F((f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_K(\mathbf{x}))), \tag{12}$$

where K is the number of SVMs in the SVM ensemble.

3.3. Extension of the SVM ensemble to the multi-class classification

In Section 2, we explained two kinds of extension methods such as one-against-all and one-against-one methods in order to apply the SVM to the multi-class classification problem. We can use these extension methods equally in the case of the SVM ensemble for the multi-class classification. For the C-class classification problem, we can have two kind of extensions according to the insertion level of the SVM ensemble: (1) the binary-class-level SVM ensemble (see Fig. 4) and (2) the multi-class-level SVM ensemble (see Fig. 5).

The binary-class-level SVM ensemble consists of C SVM ensembles in the case of one-against-all method or $C(C - 1)/2$ SVM ensembles in the case of one-against-one method. And, each SVM ensemble consists of K independent SVMs. So, the SVM ensemble is built in the level of binary classifiers. We obtain the final decision from the decision results of many SVM ensembles via either the Max Wins voting strategy or the tournament match.

The multi-class-level SVM ensemble consists of K multi-class classifiers. And each multi-class classifier consists of C binary classifiers in the case of one-against-all method or for $C(C - 1)/2$ binary classifiers in the case of one-against-one method. So, the SVM ensemble is built

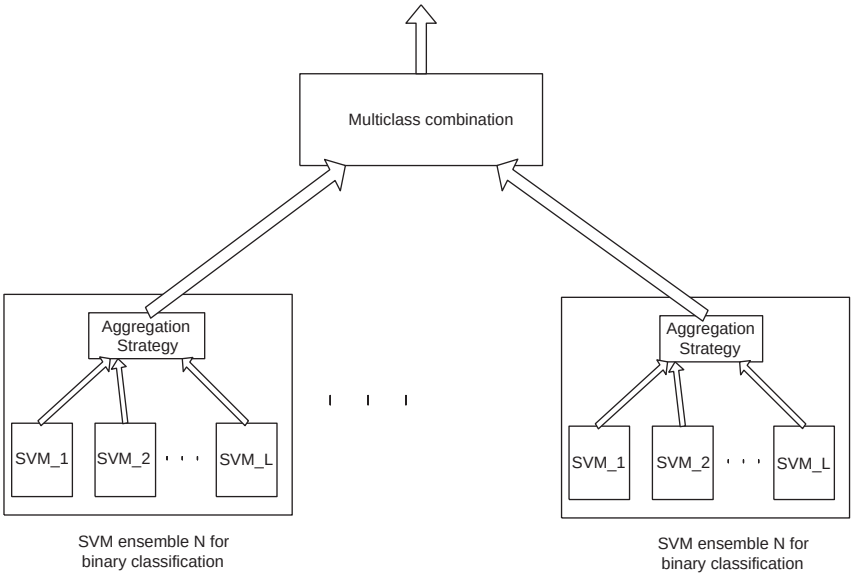

Fig. 4. The binary-class-level SVM ensemble.

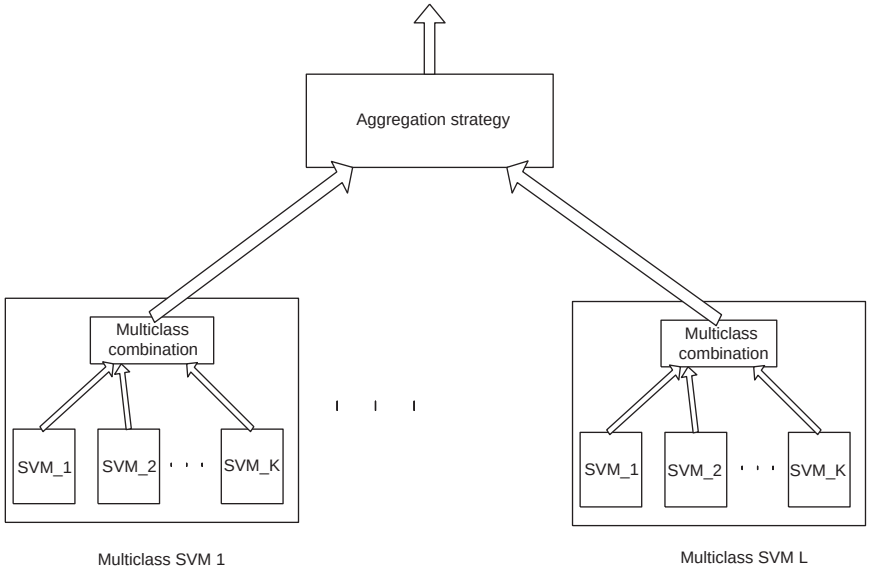

Fig. 5. The multi-class-level SVM ensemble.

in the level of multi-class classifiers. We obtain the final decision from the decision results of many multi-class classifiers via an appropriate aggregating strategy of the SVM ensemble.

4. Simulation results and discussion

To evaluate the efficacy of the proposed SVM ensemble using the bagging or boosting technique, we have performed three different classification problems such as the IRIS data classification, the UCI hand-written digit recognition, and the fraud detection. We used two different training techniques for the SVM ensemble such as bagging or boosting, and four different classification methods such as a single SVM and three different SVM ensembles with different aggregating strategies like the majority voting, the LSE-based weighting, and the double-layer hierarchical combining.

4.1. IRIS data classification

The IRIS data set [19,23] is one of the best known databases to be found in the pattern recognition literature.

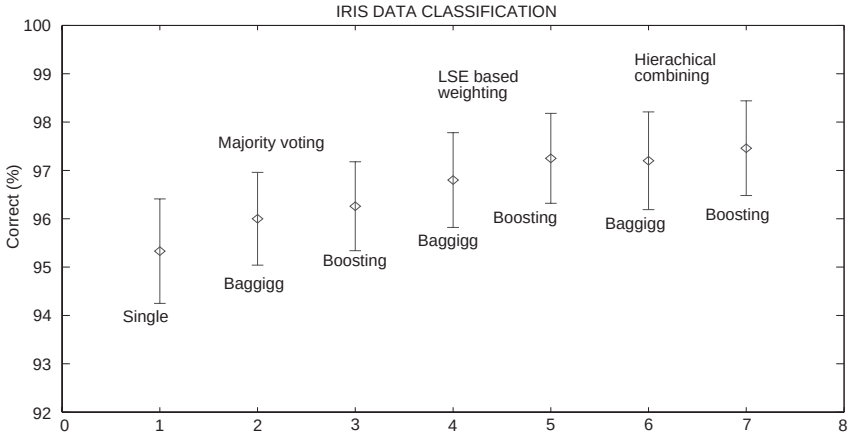

Fig. 6. The correct classification rates for IRIS data classification.

The IRIS data set contains three classes where each class consists of 50 instances. Each class refers to a type of IRIS planet. One class is linearly separable from the other classes but they are not linearly separable from each other.

We used the one-against-one method for the multi-class extension and the multi-class-level method for the SVM ensemble. There are five multi-class SVMs where each multi-class SVM has three SVMs ($SVM_{C1,C1}$, $SVM_{C1,C2}$, $SVM_{C1,C3}$). The decision of each multi-class SVM is obtained from the decision results of three SVMs via the Max Wins voting strategy. The final decision is obtained via the aggregation of five multi-class SVMs. Each SVM used 2-D polynomial kernel function.

We selected 90 data samples randomly for the training set. For bootstrapping, we re-sampled randomly 60 data samples with replacement from the training data set. For boosting, we iteratively re-sampled 60 data samples with replacement according to the updated probability distribution from the training data set. We trained each SVM independently over the replicated training data set and aggregated several trained SVMs via three different combination methods. Fig. 6 shows the classification results of four different classifiers using the test data set consisting of 60 data samples, where each SVM in the SVM ensemble was trained by two different construction methods of training samples. For avoiding the tweak problem, each classifier has been performed 10 independent runs of simulation and the average performance has been reported in the table.

4.2. UCI hand-written digit recognition

There are some literatures on the handwritten digit recognition using SVM [1,20–22]. We used the UCI hand-written digit data [23]. Some digits in the database are shown in Fig. 7. Among the UCI hand-written digits, we chose randomly 3828 digits as a training data set and the remaining 1797 digits as a test data set. The original image of each

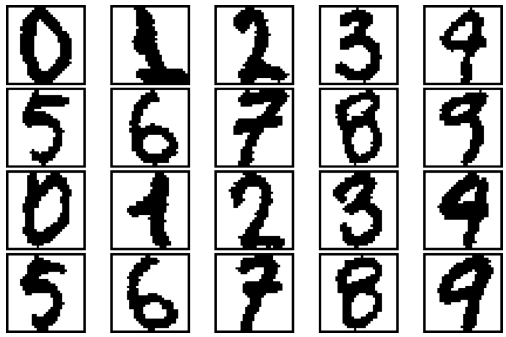

Fig. 7. Examples of hand-written digits in the UCI database.

digit has the size of 32×32 pixels. It is reduced to the size of 8×8 pixels where each pixel is obtained from the average of the block of 4×4 pixels in the original image. So each digit is represented by a feature vector with the size of 64×1.

We used the one-against-one method for the multi-class extension and the multi-class-level method for the SVM ensemble. There are 10 multi-class SVMs where each multi-class SVM has 45 SVMs ($SVM_{C0,C1}$, $SVM_{C0,C2}$, …, $SVM_{C8,C9}$). The decision of each multi-class SVM is obtained from the decision results of 45 SVMs via the Max Wins voting strategy. The final decision is obtained via the aggregation of 10 multi-class SVMs. Each SVM used 2-D polynomial kernel function.

For bagging, we re-sampled randomly 1000 digits samples with replacement from the training data set consisting of 3828 digit samples. For boosting, we iteratively re-sampled 1000 data samples with replacement according to the updated probability distribution from the training data set. We trained each SVM independently over the replicated training data set and aggregated several trained SVMs via three different combination methods. Fig. 8 shows the

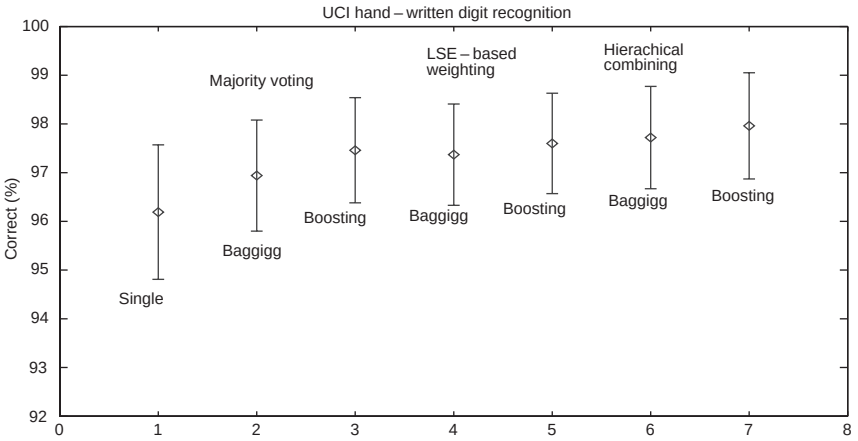

Fig. 8. The correct classification rates of UCI hand-written digit recognition.

classification results of four different classifiers using the test data set consisting of 1797 digits, where each SVM in the SVM ensemble was trained by two different construction methods of training samples. For avoiding the tweak problem, each classifier has been performed 10 independent runs of simulation and the average performance has been reported in the table.

4.3. Fraud detection

We have often met many kind of customer fraud events in our society [24–26]. For examples, the cellular fraud in the telecommunication industry, the credit card fraud in the banking industry, and compensation cheating in the insurance company. Fraud not only costs very much for a company, but also causes inconvenience and much cost for the customer. One method for fraud detection, called user profiling method [24], is checking suspicious changes in user behaviors and determining general patterns of fraud by a massive amount of user action data analysis. There are many different approaches in modelling fraud patterns, such as Bayesian network [26] and HMM [27]. Here, we tackled a mobile telecommunication payment fraud detection using the proposed SVM ensemble.

We used the database obtained from a mobile telecom company. It recorded 1 year's action data of 53,696 customers. The fraud detection method is to analyze their history action records, which is a typical data mining procedure as follows.

Step 1: Data cleaning, to purge redundant data for a certain fraud action analysis.

Step 2: Feature selection and extraction, to discover indicators corresponding to changes in behavioral indicatives of fraud.

Step 3. Modelling, to determine fraud pattern by classifier and predictor.

Step 4: Fraud action monitoring and prediction, to issue the alarm.

For Step 2: we extract the eight salient features to measure the customer payment behaviors using PROC DATA of SAS/BASE and combine the user profile(IBS_USERINFO) and user action (IBS_BILL) to one table (IBS_USERPERFORMANCE) using the SQL sentence given below. Table IBS_USERPERFORMANCE has 53,696 records and nine attribute items.

```
select   a.CreditDegree, d.* ;
from     Fraud.ibs_Credit as a, Fraud.ibs_Bill as b,
         Fraud.ibs_Usrphone as c, Fraud.ibs_Usrinfo as d;
where    a.billphone_id = b.billphone_id and b.account =
         c.account and c.usrinfo_id = d.usrinfo_id;
into     Fraud.ibs_Userperformance.
```

Now, we are focusing on the works of the later two steps: Steps 3 and 4. For determining the fraud patterns, the data set is divided into a training set and a test set, in which the training set contains 70% of 53,696 records, and the test data set is the remaining 30% of 53,696 records. Fraud detection problem can be a binary or a multi-class classification problem. In case that customers are divided into two classes: fraud or non-fraud, fraud detection is a binary classification problem. In case that customers are to be classified as more than two confidence grades, fraud detection is a multi-class classification. We performed both kinds of fraud detection problems.

For binary-class case of fraud detection, the SVMs in the SVM ensemble used the third-order polynomial kernel and the RBF kernel. For multi-class case of fraud detection, we considered four confidence grades labelled 1, 2, 3, and 4, where 1 is the most credible customer and 4 is the most dangerous customer. We used one-against-one method for multi-class extension and the multi-class-level method for the SVM ensemble. There are 11 multi-class SVMs where each multi-class SVM has six SVMs

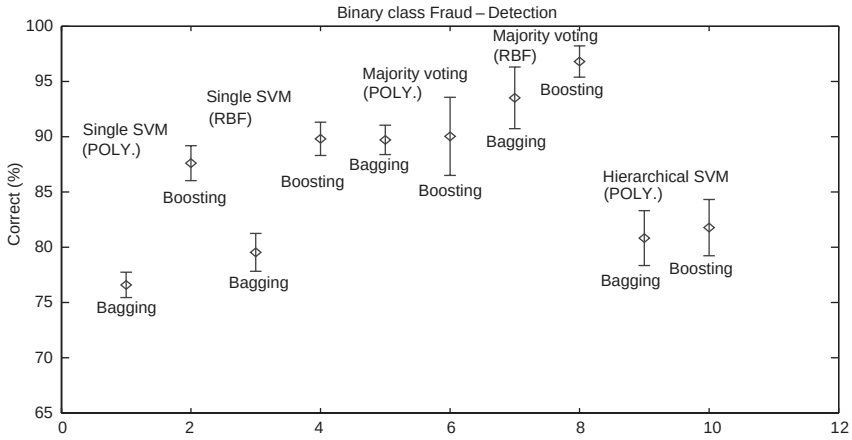

Fig. 9. The correct classification rates of the binary-class fraud detection.

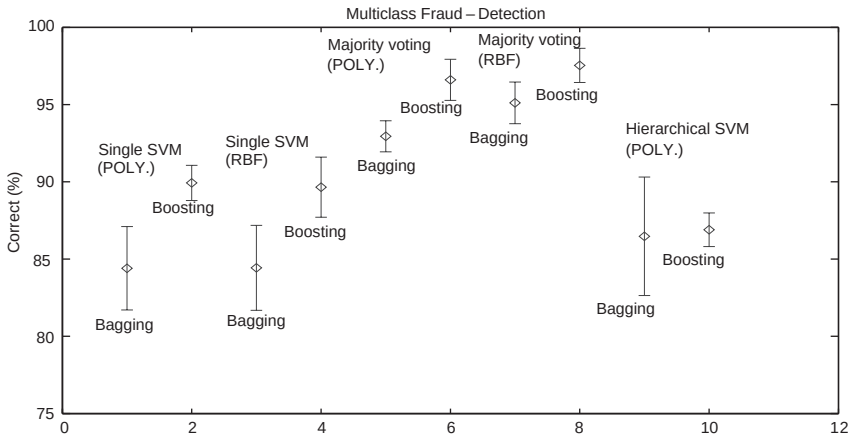

Fig. 10. The correct classification rates of the multi-class fraud detection.

($SVM_{C1,C2}, SVM_{C1,C3}, \ldots, SVM_{C3,C4}$). The decision of each multi-class SVM is obtained from the decision results of six SVMs via the Max Wins voting strategy. The final decision is obtained via the aggregation of 11 multi-class SVMs. While we used the third-order polynomial kernel and the RBF kernel for kernel in the case of majority voting aggregation method, we used only the third-order polynomial kernel in the case of hierarchical combining aggregation method.

For bagging, we re-sampled randomly 30% of the training samples with replacement from the training data set consisting of 53,696 records. For boosting, we re-sampled iteratively 30% of the training samples with replacement according to the updated probability distribution from the training data set. We trained each SVM independently over the replicated training data set and aggregated several trained SVMs via two different combination methods such as the majority voting and the double-layer hierarchical combining method because the LSE-based weighting cannot be applied due to the huge size of training set.

Figs. 9 and 10 show the classification results of five different classifiers using the test data set consisting of 16,109 records in the case of binary-class and multi-class fraud detection, respectively. For avoiding the tweak problem, each classifier has been performed 10 independent runs of simulation and the average performance has been reported in the tables. From these tables, we note that (1) the SVM ensemble improves the correct classification rate greatly compared with that of using a single SVM, (2) the majority voting aggregation using the polynomial kernel provides the best classification performance in the fraud detection, and (3) the multi-class fraud detection provides better correct classification rate than the binary fraud detection.

5. Conclusion

Usually, the practical SVM has been implemented based on the approximation algorithm to reduce the cost of time

and space. So, the obtained classification performance is far from the theoretically expected level of it. To overcome this limitation, we addressed the SVM ensemble that consists of several independently trained SVMs. For training each SVM, we generated many replicated training sample sets via the bootstrapping or boosting technique. Then, all independently trained SVMs over the replicated training sample sets were aggregated by three combination techniques such as the majority voting, the LSE-based weighting, and the double-layer hierarchical combining.

We also extended the SVM ensemble to the multi-class classification problem: the binary-class-level SVM ensemble and the multi-class-level SVM ensemble. The former built the SVM ensemble in the level of binary classifiers and the latter built the SVM ensemble in the level of multi-class classifiers. The former had C SVM ensembles in the case of one-against-all method or $C(C-1)/2$ SVM ensembles in the case of one-against-one method. And, each SVM ensemble consisted of K independent SVMs. The latter consisted of K multi-class classifiers. And each multi-class classifier consists of C binary classifiers in the case of one-against-all method or for $C(C-1)/2$ binary classifiers in the case of one-against-one method.

We evaluated the classification performance of the proposed SVM ensemble over three different multi-class classification problems such as the IRIS data classification, the hand-written digit recognition, and the fraud detection. The SVM ensembles outperform a single SVM for all applications in terms of classification accuracy. For three different aggregation methods, the classification performance is superior in the order of the double-layer hierarchical combining, the LSE-based weighting, and the majority voting.

Acknowledgements

The authors would like to thank the Ministry of Education of Korea for its financial support toward the Electrical and Computer engineering Division at POSTECH through its BK21 program. This research was also partially supported by a grant (R01-1999-000-00224-0) from Korea Science and Engineering Foundation.

References

[1] C. Cortes, V. Vapnik, Support vector network, Mach. Learn. 20 (1995) 273–297.
[2] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Mining Knowledge Discovery 2 (2) (1998) 121–167.
[3] T. Joachims, Making large-scale support vector machine learning practical, Advances in Kernel Methods: Support Vector Machines, MIT Press, Cambridge, MA, 1999.
[4] John Platt, Fast training of support vector machines using sequential minimal optimization, Advances in Kernel Methods: Support Vector Machines, MIT Press, Cambridge, MA, 1999.
[5] J. Weston, C. Watkins, Support vector machines for multi-class pattern recognition, Proceedings of the Seventh European Symposium on Artificial Neural Networks, Bruges, Belgium, 1999.
[6] L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard, V. Vapnik, Proceedings of the 12th IAPR International Conference on Pattern Recognition, Jerusalem, Israel, 1994, pp. 77–87.
[7] S. Knerr, L. Personnaz, G. Dreyfus, Single-layer learning revisited: a stepwise procedure for building and training a neural network, Neurocomputing: Algorithms, Architechtures and Application, Springer, Berlin, 1990.
[8] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1999.
[9] T. Dietterich, Machine learning research: four current directions, Artiff. Intell. Mag. 18 (4) (1998) 97–136.
[10] T. Dietterich, Ensemble methods in machine learning, Multiple Classifier Systems, Lecture Notes in Computer Science, Calgari, Italy, 2000.
[11] K. Ting, I. Witten, Stacking bagged and dagged models, Proceedings of the 14th International Conference on Machine Learning, Sapporo, Japan, 1997.
[12] R. Maclin, D. Opitz, An empirical evaluation of bagging and boosting, Proceedings of the Fourth National Conference on Artificial Intelligence, Austin, Texas, 1997.
[13] T. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization, Mach. Learn. 26 (1998) 1–22.
[14] L. Hansen, P. Salamon, Neural network ensembles, IEEE Trans. Pattern Anal. Mach. Intell. 12 (1990) 993–1001.
[15] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[16] Y. Freund, R. Schapire, A decision theoretic generalization of online learning and an application to boosting, J. Comput. System Sci. 55 (1) (1997) 119–139.
[17] D. Kim, C. Kim, Forecasting time series with genetic fuzzy predictor ensemble, IEEE Trans. Fuzzy Systems 5 (4) (1997) 523–535.
[18] M. Jordan, R. Jacobs, Hierarchical mixtures of experts and the EM algorithm, Neural Comput. 6 (5) (1994) 181–214.
[19] R. Fisher, The use of multiple measurements in taxonomic problems, Ann. Eugenics 7 (2) (1936) 179–188.
[20] B. Schölkopf, C. Burges, V. Vapnik, Extracting support data for a given task, Proceedings of the First International Conference on Knowledge Discovery and Data Mining, Montreal, Canada, 1995.
[21] B. Schölkopf, C. Burges, V. Vapnik, Incorporating invariances in support vector learning machines, Proceedings of IJCNN'96, Washington DC, USA, 1996, pp. 47–52.
[22] C.J.C. Burges, B. Schoölkopf, Improving the accuracy and speed of support vector learning machines, Adv. Neural Inf. Process. Systems 9 (1997) 375–380.
[23] B. Bay, The UCI KDD Archive [http://kdd.ics.uci.edu], Department of Information and Computer Science, University of California, Irvine, CA, 1999.
[24] T. Fawcett, F. Provost, Adaptive fraud detection, Data Mining Knowledge Discovery 1 (1997) 1–28.
[25] J. Hollmen, V. Tresp, Call-based fraud detection in mobile communication networks using a hierarchical regime-switching model, Adv. Neural Inf. Process. Systems 11 (1999) 889–895.

[26] K. Ezawa, S. Norton, Constructing bayesian networks to predict uncollectible telecommunications accounts, IEEE Expert Intell. Systems Appl. 11 (5) (1996) 45–51.

[27] P. Smyth, Hidden Markov models for fault detection in dynamic systems, Pattern Recognition 27 (1) (1994) 149–164.

About the Author—HYUN CHUL KIM received the B.S. degree (1999) in computer science and engineering, the B.S. degree (1999) in mathematics, and the M.S. degree (2001) in computer science and engineering, from Pohang University of Science and Technology, South Korea. He is studying for the Ph.D. degree in computer science and engineering in the same university. His research interests are pattern recognition and machine learning.

About the Author—SHAONING PANG received the B.S. degree in Physics and the M.S. degree in electrical engineering from Xinjiang University, Urumbi, China, in 1994. In 2000, he received the Ph.D. degree from Shanghai Jiao Toing University, Shanghai, China. During 2001–2002, he was a research associate in the Department of Computer Science and Engineering at POSTECH Pohang, South Korea. He is currently a research assistant of school of information and communication, Paisley University, U.K.

About the Author—HONG-MO JE received the B.S. degree in Computer Science from Yeungnam University, Taegu, Korea, in 1999, and M.S. degree in Computer Science and Engineering from Pohang University of Science and Technology (POSTECH), Pohang, Korea, in 2002. During 2002.2–2003.2, he has been working at Information Research Center at Samsung Data System (SDS),. Seoul, South Korea. He is currently a Ph.D. candidate student of Computer Science and Engineering. His research interest are pattern recognition and machine learning.

About the Author—DAIJIN KIM received the B.S. degree in electronic engineering from Yonsei University, Seoul, Korea, in 1981, and the M.S. degree in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Taejon, 1984. In 1991, he received the Ph.D. degree in electrical and computer engineering from Syracuse University, Syracuse, NY. During 1992–1999, he was an Associate Professor in the Department of Computer Engineering at DongA University, Pusan, Korea. He is currently an Associate Professor in the Department of Computer Science and Engineering at POSTECH, Pohang, Korea. His research interests include intelligent systems, biometrics, and bioinformatics.

About the Author—SUNG-YANG BANG received his B.S. in EE from Kyoto University, Japan in 1966 and his M.S. in EE from Seoul National University, South Korea in 1969. He received his Ph.D. in CS from the University of Texas at Austin in 1974. He taught at Wayne State University and worked for NCR and Bell Laboratories. In 1986, he joined Pohang University of Science and Technology, Pohang, South Korea as a professor to establish Department of Computer Science and Engineering, where he currently serves as the department head for the second time. At POSTECH he also is the director of Brain Research Center. He participated in organizing APNNA (Asia Pacific Neural Network Assembly) and served as the organizing chair of the first ICONIP held in Seoul, Korea in 1994. His research interests are pattern recognition and neurocomputing.