Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition 41 (2008) 662–671
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/pr

Fingerprint classification using one-vs-all support vector machines dynamically ordered with naïve Bayes classifiers

Jin-Hyuk Hong, Jun-Ki Min, Ung-Keun Cho, Sung-Bae Cho*

Department of Computer Science, Biometrics Engineering Research Center, Yonsei University, 134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Republic of Korea

Received 7 September 2006; received in revised form 8 June 2007; accepted 4 July 2007

Abstract

Fingerprint classification reduces the number of possible matches in automated fingerprint identification systems by categorizing fingerprints into predefined classes. Support vector machines (SVMs) are widely used in pattern classification and have produced high accuracy when performing fingerprint classification. In order to effectively apply SVMs to multi-class fingerprint classification systems, we propose a novel method in which the SVMs are generated with the one-vs-all (OVA) scheme and dynamically ordered with naïve Bayes classifiers. This is necessary to break the ties that frequently occur when working with multi-class classification systems that use OVA SVMs. More specifically, it uses representative fingerprint features as the FingerCode, singularities and pseudo ridges to train the OVA SVMs and naïve Bayes classifiers. The proposed method has been validated on the NIST-4 database and produced a classification accuracy of 90.8% for five-class classification with the statistical significance. The results show the benefits of integrating different fingerprint features as well as the usefulness of the proposed method in multi-class fingerprint classification.

Keywords: Fingerprint classification; Support vector machine; FingerCode; Naïve Bayes classifier; Singularity; Pseudo ridges; Dynamic classification

1. Introduction

Fingerprint classification is an important part of automatic fingerprint identification systems (AFIS). Classification is necessary to reduce the number of one-to-one comparisons performed when executing a fingerprint query [1,2]. According to the Henry system, fingerprints can be partitioned into several classes including whorl, left loop, right loop, arch and tented arch [3].

Since the Henry system categorizes fingerprints by relative position and number of core and delta points in the print, many researchers have tried to extract singular points in the flow of the ridges [4]. Karu and Jain [5] proposed a heuristic algorithm with singularities, Nyongesa et al. [2] used the relative positions of the cores and deltas while Zhang and Yan [6] used singularities and pseudo ridges to classify fingerprints. Although singularities can lead to vague classification, it is hard to obtain high accuracy when the quality of the fingerprint images is low [6,7].

In order to obtain higher accuracy, various features, such as the FingerCode, ridge distributions and directional images, have also been actively investigated. Jain et al. [8] proposed the FingerCode; a method that uses a Gabor filter to extract directional ridge flow, and Park [9] used an orientation filtered by a fast Fourier transform. Chong et al. [10] employed both a geometric grouping and a global geometric shape analysis of fingerprint ridges, while Cappelli et al. [3] proposed a directional image that models fingerprints with a graph. Nagaty [7] extracted a string of symbols using block directional images of fingerprints, while Chang and Fan [11] proposed a ridge distribution model consisting of a combination of 10 basic ridge patterns with different ridge distribution sequences. Since there is more information than that contained in simple singularities, fingerprint classification is more accurate when using these features.

There have been other attempts to integrate features and methods in order to produce a robust fingerprint classifier

* Corresponding author. Tel.: +82 2 2123 3877; fax: +82 2 365 2579.
E-mail addresses: hjinh@sclab.yonsei.ac.kr (J.-H. Hong), loomlike@sclab.yonsei.ac.kr (J.-K. Min), bearoot@sclab.yonsei.ac.kr (U.-K. Cho), sbcho@cs.yonsei.ac.kr (S.-B. Cho).

0031-3203/$30.00
doi:10.1016/j.patcog.2007.07.004

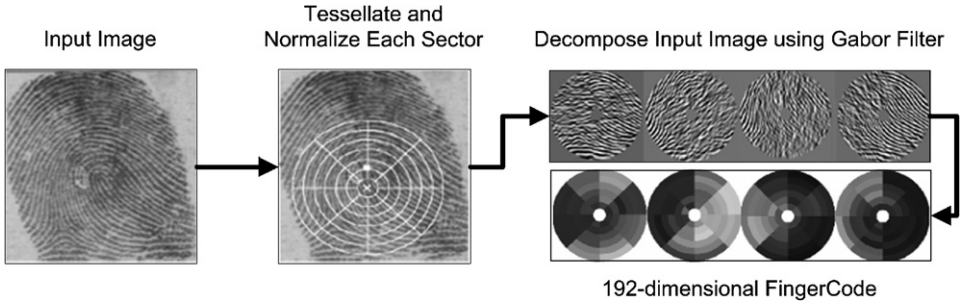

Fig. 1. Flow diagram of the FingerCode feature vector [8].

[4,12]. Senior used hidden Markov models and decision trees to recognize ridge structures of prints [4], while Yao et al. [12] combined flat and structured features using recursive neural networks and support vector machines (SVMs). When several types of features and methods are combined, fingerprint classification appears to be more accurate and reliable, especially when working with fingerprints that contain high levels of noise.

This paper proposes a novel fingerprint classification approach integrating naïve Bayes (NB) classifiers and SVMs that use different fingerprint features. For highly accurate classification, SVMs with FingerCodes are generated based on the OVA scheme that shows good performance for multi-class classification using binary classifiers in the literature [13], while the NBs (with singularities), which provide confident and understandable class probabilities, dynamically organize them according to the probabilities. Since it is hard to determine the class of some ambiguous fingerprints, OVA SVMs are sequentially evaluated by the probability of the classes from the NB.

The proposed method has been validated on the NIST-4 database [14] with three series of experiments. In the first experiment, the NB with singularities achieved an accuracy of 85.4% without rejection. In the second experiment, SVMs with FingerCodes were used with the OVA scheme, and this combination achieved an accuracy of 90.1% with 1.8% rejection. Finally, the proposed method, which integrates NBs and SVMs, produced an accuracy of 90.8% with 1.8% rejection. This result indicates the benefits of integrating different fingerprint features, and it also demonstrates the usefulness of combining NBs and SVMs. The experiments were all performed using discrete, Henry classification, but it is possible to extend the method to continuous classification as well.

2. Using SVMs with FingerCodes for fingerprint classification

2.1. The FingerCode

The FingerCode, proposed by Jain [8], is a representative fingerprint feature extracted from the NIST-4 database. An algorithm sets a registration point in a fingerprint image and tessellates it into 48 sectors. The Gabor filter is then applied in four directions (0°, 45°, 90°, and 135°) so as to accentuate the ridge parallel to each direction. Ridges that are not parallel will appear blurred, as shown in Fig. 1.

Standard deviations are computed on 48 sectors for each of the four filtered images to generate the 192-dimensional feature vector called the FingerCode. Jain et al. [8] achieved accuracy of 90% with 1.8% rejection using the k nearest neighbors and neural networks methods with the FingerCode, while Yao et al. [12] obtained accuracy of 90% with 1.8% rejection with SVMs of the error-correcting code scheme using both the FingerCode and recursive neural networks-extracted features.

Table 1
Kernel functions of SVMs

Linear	Polynomial	Gaussian	Sigmoid
$(x \cdot x_i)$	$(x \cdot x_i + \gamma)^d$	$\exp\left(-\frac{\|x-x_i\|^2}{2\sigma^2}\right)$	$\tanh(x \cdot x_i + \gamma)$

2.2. Using SVMs for fingerprint classification

SVMs, well researched in statistical learning theory, have been actively investigated in pattern classification and regression [15,16]. SVMs map an input sample to a high dimensional feature space and try to find an optimal hyperplane that minimizes the recognition error for the training data using the non-linear transformation function [16].

$$X : x = (x_1, \ldots, x_n) \rightarrow F : \Phi(x) = (\Phi_1(x), \ldots, \Phi_n(x)). \quad (1)$$

Let n be the number of training samples. For the sample x_i with the class-label $c_i \in \{1, -1\}$, the SVM calculates

$$f(x) = \sum_{i=1}^{n} \alpha_i c_i K(x, x_i) + b,$$
$$K(x, x_i) = \Phi(x) \cdot \Phi(x_i). \quad (2)$$

Coefficient α_i in Eq. (2) is non-zero when x_i is a support vector that composes the hyperplane; otherwise it is zero. The kernel function $K(x, x_i)$ can be easily computed by an inner product of the non-linear mapping function. Table 1 shows some representative kernel functions, including the linear, polynomial, Gaussian, and sigmoid functions.

Since SVMs are basically binary classifiers, a decomposition strategy for multi-class classification, such as the one-vs-all, pairwise or complete-code methods, is required [16,17].

(1) *The one-vs-all method*: M (no. of classes) SVMs are trained, where each SVM classifies samples into corresponding classes against all the others. The decision function $f_j(x)$ of the jth SVM replaces c_i of Eq. (2) with t_i as follows:

$$t_i = \begin{cases} +1 & \text{if } c_i = j, \\ -1 & \text{if } c_i \neq j. \end{cases} \tag{3}$$

(2) *The pairwise method*: An SVM manages a pair of classes, where the SVMs for all pairs of classes (${}_M C_2 = M(M-1)/2$) are constructed. This strategy is faster than the others, because it uses only a small portion of the training data. For $j, k = \{1, \ldots, M\}$ and $j \neq k$, the decision function $f_{j,k}(x)$ is defined by replacing c_i of Eq. (2) with t_i as follows:

$$t_i = \begin{cases} +1 & \text{if } c_i = j, \\ -1 & \text{if } c_i = k. \end{cases} \tag{4}$$

(3) *The complete-code method*: The SVMs for all binary combinations of the classes are constructed, while the five-class classification requires both one-vs-all and two-vs-all methods. When whole classes are divided into two groups $\{j\}$ and $\{k\}$, the decision function $f_{j,k}(x)$ is obtained by changing c_i of Eq. (2) by t_i as follows:

$$t_i = \begin{cases} +1 & \text{if } c_i \in \{j\}, \\ -1 & \text{if } c_i \in \{k\}. \end{cases} \tag{5}$$

After constructing the SVMs, a fusion method is required to combine the multiple outputs of the SVMs. Popular methods for combining multiple SVMs include majority voting, winner-takes-all, error-correcting codes (ECCs), behavior knowledge space (BKS) and decision templates.

(1) *The majority voting method*: For a sample, this method simply counts the votes received from the individual classifiers and selects the class with the largest number of votes [18]. An analytic justification may be given by the well-known Condorcet's theorem [19], while a theoretical study can be found in [20,21]. Although it is simple to achieve good performance, this method cannot handle cases where classifiers tie.

(2) *The winner-takes-all method*: In order to resolve problems caused by majority voting, this method classifies a sample into the class that receives the highest value among the L classifiers for the M-class problem. This is often known as maximum where $ind_{i,j}(x)$ is an indicator function with 1 if the label i is the positive class of the jth SVM, -1 if it is the negative class, and 0 if it is otherwise.

$$c = \arg \max_{i=1,\ldots,M} \sum_{j=1}^{L} ind_{i,j}(x) d_j(x). \tag{6}$$

(3) *The ECC method*: This method generates a coding matrix $E \in \{-1, 0, 1\}^{M \times L}$ where M and L are the number of classes and classifiers, respectively [22,23]. $E_{i,j}$ represents an entry in the ith row and jth column of E. ($E_{i,j} = -1$ or 1) indicates that the points in class i are regarded as negative or positive examples when training the jth classifier. If $E_{i,j} = 0$, class i is not used when the jth classifier is trained. A test point is classified into the class whose row in the coding matrix has the minimum distance to the vector of the outputs of the classifiers. Eq. (7) shows the class decision function of ECCs that use the Hamming distance:

$$c = \arg \min_{i=1,\ldots,M} \sum_{j=1}^{L} \frac{1 - sign(E_{i,j} d_j(x))}{2}. \tag{7}$$

(4) *The BKS method*: In this method, possible combinations of the outputs of the classifiers are stored in the BKS-table $T \in \{-1, 1\}^{M^L \times L}$, for the M-class problem with L classifiers. Each entry in the T contains a single class label (most frequently encountered amongst the samples of the training data in this cell) or no label (no sample of the training data has the respective combination of class labels). In tests, a new sample can be classified into the label of entry with the same outputs of the classifiers [24]. It fails to classify when an output pattern is not found in T.

(5) *The decision templates method*: Single-DTs generate decision templates of each class by averaging the decision profiles (DPs) for the training samples. For the M-class problem with L classifiers, the $DP(x_i)$ of the ith sample is

$$DP(x_i) = \begin{bmatrix} d_{1,1}(x_i) & \cdots & d_{1,M}(x_i) \\ \vdots & d_{y,z}(x_i) & \vdots \\ d_{L,1}(x_i) & \cdots & d_{L,M}(x_i) \end{bmatrix}, \tag{8}$$

where $d_{y,z}(x_i)$ is the degree of support given by the yth classifier for the sample x_i of the class z. When DPs are generated from the training data, Eq. (9) estimates the decision template DT_c of the class c. $Ind_c(x_i)$ has a value of 1 if x_i's class is c, otherwise it has a value of 0 [25,26].

$$DT_c = \begin{bmatrix} dt_c(1,1) & \cdots & dt_c(1,M) \\ \vdots & dt_c(y,z) & \vdots \\ dt_c(L,1) & \cdots & dt_c(L,M) \end{bmatrix},$$

$$dt_c(y,z) = \frac{\sum_{i=1}^{n} ind_c(x_i) d_{y,z}(x_i)}{\sum_{i=1}^{n} ind_c(x_i)}. \tag{9}$$

In the test stage, the equation computes the distance between the DP of a new sample and the decision templates of each class. The class-label is defined as the class that contains the most similar decision templates. Kuncheva experimented with single-DTs using 11 similarity comparison measures including the Hamming (HM) and Euclidean (EU) distances and achieved a higher classification accuracy than other fusion methods such as the majority voting and BKS methods [25]. Even though there are various methods for combining multiple SVMs, it is

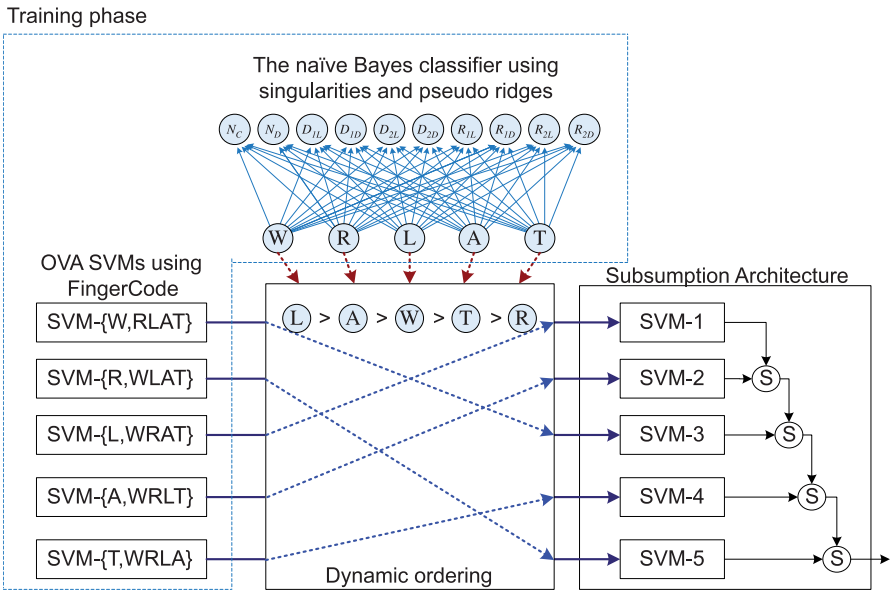

Fig. 2. Procedure of the proposed method.

still hard to manage cases of ties and rejections. Also, conventional fusion methods combine multiple classifiers statically so as to be limited in classifying ambiguous fingerprints.

3. A dynamic fingerprint classifier

Contrary to static classification, in which classifiers are not adaptively changed for an input sample, we propose a dynamic fingerprint classifier that not only uses various fingerprint features (singularity, pseudo ridges and the FingerCode), but also addresses the ambiguity of the OVA SVMs. It is possible that tie cases, which frequently occur when using OVA SVMs for multi-class classification, might decrease classification performance. The proposed method manages this possibility by organizing the OVA SVMs based on the subsumption architecture. The subsumption architecture is a representative method used to select a proper action when there are multiple actions activated [27], while the order of the models is determined by the NB classifier in this paper.

The proposed method consists of the NB and the OVA SVMs as shown in Fig. 2. The NB estimates the posterior probability for the fingerprint classes $prob = \{p_w, p_l, p_r, p_a, p_t\}$ by using singular points and pseudo ridges. The OVA SVMs classify fingerprints by using the FingerCode as explained in Section 2, where the margin of a sample $o\text{-}svm = \{m_W, m_R, m_L, m_A, m_T\}$ is produced. In order to manage ambiguity in cases of ties and rejections, in this paper, the proposed method sequentially selects the OVA SVMs. The evaluation order of the OVA SVMs is determined by the posterior probability of each class that the NB produces. The corresponding OVA SVM of a more probable class takes precedence in the subsumption architecture over the other OVA SVMs.

When a sample is inputted, the method first estimates the probability of finding certain fingerprint classes by using the NB classifier, and then organizes the OVA SVMs as the subsumption architecture according to the probability. Finally, a sample is evaluated sequentially until an OVA SVM is satisfied. When an OVA SVM is satisfied, the sample is classified into the corresponding class of the OVA SVM, while it is classified into the class of the highest probability when no OVA SVMs are satisfied. Fig. 3 shows the pseudo code for the proposed method. Ordering the OVA SVMs properly for an input sample produces dynamic classification.

In order to construct the NB classifier for ordering the OVA SVMs, we adopt two representative features of fingerprints called the singularity and the pseudo ridge. Singular points in fingerprints are known as core and delta points, where the core is the topmost point on the innermost recurving ridge and the delta is the center of a triangular region where three different direction flows meet [5,6]. The number and location of core and delta points are used to classify fingerprints.

The Poincare index is a representative algorithm used to detect singular points [1,6]. It computes cores and deltas based on the orientation field of the fingerprint images. Since singular points are usually found in the center of images, some constraints (used in the NIST-4 database) are considered as follows:

- Since the border of a 512×480 image contains a large background region, no singular point is allowed in the 40-pixel wide strips along the border.
- No cores are allowed in the 80-pixel wide strips along the border.
- Deltas are usually found near the bottom of images, so the 160-pixel strip from the top is neglected.
- If a core is less than 8 pixels away from the nearest delta, this core-delta pair is removed.
- The maximum number of cores or deltas is two, and we select the ones that are nearest to the center.

```
prob[5] = {p_W, p_R, p_L, p_A, p_T}          // prob[] is calculated by the naïve Bayes
      classifier
order[5] = {0, 1, 2, 3, 4}
o-svm[5] = {m_W, m_R, m_L, m_A, m_T}         // o-svm[] is obatined by the OVA SVMs

// determine the order of OVA SVMs to evaluate
for (i=0; i<5; i++)
   for (j=i+1; j<5; j++)
      if (prob[i] < prob[j]) {
         int iTemp = prob[i];  prob[i] = prob[j];        prob[j] = iTemp;
         iTemp = order[i];     order[i] = order[j];      order[j] = iTemp;
      }
// classify with OVA SVMs according to the subsumption architecture
if (prob[order[0]] < r_1)                    // r_1 is a rejection threshold
   return reject;
for (i=0; i<5; i++) {
   if (o-svm[order[i]] >= a) {               // a is a threshold
      if(o-svm[order[i]] < r_2)              // r_2 is a rejection threshold
         return reject;
      return order[i];
}}
return order[0];
```

Fig. 3. Pseudo code for probabilistically ordering the OVA SVMs.

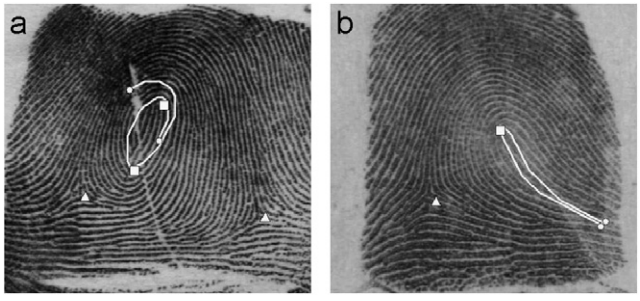

Fig. 4. Singular points and pseudo ridges (□ = core, △ = delta, ◯ = end of the pseudo ridge).

The number of cores and deltas are denoted as *NC* and *ND*, respectively. The nearest core to the center is denoted as *C* (if there is no core, *C* represents the center of the image). Fig. 4(a) shows an example with two cores where the lower core is selected as *C*. *D*1 and *D*2 denote deltas in the order of their locations.

Since this method often fails to correctly extract singular points because of poorly qualified images, many researchers use not only singularity but also additional features. For example, Zhang et al. [6] proposed pseudo ridges to make up for the weak points in singularity-based fingerprint classification. Pseudo ridges consist of a predefined number (100 in this paper) of points. A pseudo ridge is composed of 200 points based on orientation in two opposite directions from the starting point *C*. At first, orientation is estimated using the direction index in the range of (0, 16), as shown in Fig. 5(a).

The *dir* is the index from the current point *p*, *dirb* is the previous index, and *dirn*, the next tracing index, is calculated as follows:

$$dirn = \begin{cases} dir, & \cos^{-1}(\overrightarrow{V_{dirb}} \cdot \overrightarrow{V_{dir}}) \\ & \quad > \cos^{-1}(\overrightarrow{V_{dirb}} \cdot \overrightarrow{V_{(dir+8)\%16}}), \\ (dir+8)\%16 & \text{otherwise.} \end{cases} \tag{10}$$

If the direction is calculated, *p* is linked to a point corresponding to the index shown in Fig. 5(b). Tracing continues from the next point. If d_p satisfies Eq. (12), the pseudo ridge is

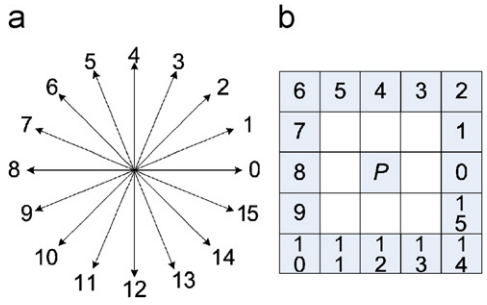

Fig. 5. Direction index for the pseudo ridge.

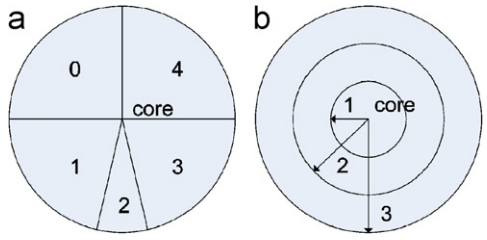

Fig. 6. Relative position and distance between C and P: (a) Location value, (b) distance value.

regarded as turn:

$$d_p = \begin{cases} dir - dirb + 16, & dir - dirb < -8, \\ -dir + dirb + 16, & dir - dirb > 8, \\ dir - dirb & \text{otherwise}, \end{cases} \tag{11}$$

$$\left| \sum_{p=0}^{n} d_p \right| > 15. \tag{12}$$

Tracing the pseudo ridge is terminated if 100 iterations are reached or if it might be regarded as turn. If the pseudo ridge goes to the right-hand side, a fingerprint is classified as 'left loop' (Fig. 5(b)), while it is categorized as 'whorl' if the pseudo ridge is regarded as turn (Fig. 5(a)).

In order to use singular points and pseudo ridges in fingerprint classification, locations and distances between C (the core point in Fig. 6) and other points are parameterized. The relative location L between C and a point P is discrete as follows:

(1) $C_y > P_y$:

$$L = \begin{cases} 0, & C_x > P_x, \\ 4, & C_x \leqslant P_x. \end{cases} \tag{13}$$

(2) $C_y \leqslant P_y$:

$$L = \begin{cases} 1, & C_x > P_x \text{ and } |C_y - P_y| < 4.0|C_x - P_x|, \\ 2, & |C_y - P_y| \geqslant 4.0|C_x - P_x|, \\ 3, & C_x \leqslant P_x. \end{cases} \tag{14}$$

Table 2
Features of the naïve Bayes classifier

Feature	Definition	State
N_C, N_D	Number of core points and delta points	0, 1, 2
D_{1L}, D_{2L}	Location of delta point	0, 1, 2, 3, 4, Absent
R_{1L}, R_{2L}	Location of the end point of the pseudo ridge	0, 1, 2, 3, 4, *turn*
D_{1D}, D_{2D}	Distance between C and delta points	1, 2, 3, Absent
R_{1D}, R_{2D}	Distance between C and the end point of the pseudo ridge	1, 2, 3, *turn*

The distance D between them is computed as follows:

$$dis = |(C_x - P_x)^2 + (C_y - P_y)^2|, \tag{15}$$

$$D = \begin{cases} 1, & dis \leqslant 10, \\ 2, & 10 < dis \leqslant 20, \\ 3 & \text{otherwise}. \end{cases} \tag{16}$$

In this paper, the number of cores and deltas (NC, ND), the location and distance between them ($D1L$, $D1D$, $D2L$, $D2D$), and the location and distance between cores and the end points of pseudo ridges ($R1L$, $R1D$, $R2L$, $R2D$) are used for fingerprint classification using the NB classifier. It consists of five mutually exclusive and exhaustive classes (W, R, L, A, T) and 10 features (NC, ND, $D1L$, $D1D$, $D2L$, $D2D$, $R1L$, $R1D$, $R2L$, $R2D$), where each class is linked with all features as shown in Table 2 and Fig. 2. The NB estimates the posterior probability of each class given the observed attribute values for an instance, and the class with the highest posterior probability is finally selected [28].

In order to calculate the posterior probability, the classifier must be defined by the marginal probability distribution of variables and by a set of conditional probability distributions of each attribute A_i given each class c_j [29]. These are estimated from the training set of n_T images. When A_i is the ith state of the attribute A and count (A_i) is the frequency of cases in which the attribute A appears with the ith state, priority probability $P(A_i)$ is estimated as follows:

$$P(A_i) = \frac{count(A_i)}{n_T}. \tag{17}$$

If A has a parent node B, conditional probability $P(A_i|B_j)$ is calculated by the following equation:

$$P(A_i|B_j) = \frac{count(A_i, B_j)}{count(B_j)}. \tag{18}$$

Bayes' theorem yields the posterior probability of each class given 10 features as evidence over a class C and $F_1 \sim F_n$:

$$P(C|F_1, \ldots, F_n) = \frac{P(C)P(F_1, \ldots, F_n|C)}{P(F_1, \ldots, F_n)}. \tag{19}$$

Since only the numerator of the fraction is interested in and the denominator does not affect C, the posterior probability of

Table 3
Comparison of the SVMs classifiers with different methods

Fusion methods	One-vs-all	Pairwise	Complete-code
Winner-takes-all	90.1	87.7	90.0
ECCs (HM)	90.1	88.6	90.0
ECCs (EU)	90.1	88.6	90.0
BKS	88.8	89.4	89.3
DTs (HM)	89.6	87.6	89.6
DTs (EU)	89.8	88.3	89.5

a class might be expressed as follows with the independence assumption among features:

$$P(C)P(F_1,\ldots,F_n|C) = P(C)P(F_1|C)P(F_2|C)\ldots P(F_n|C) = P(C)\prod_{i=1}^{n} P(F_i|C). \quad (20)$$

Finally, the case is assigned to the class with the highest posterior probability as follows:

$$\arg\max_c P(C=c)\prod_{i=1}^{n} P(F_i=f_i|C=c). \quad (21)$$

4. Experimental results

4.1. Experimental environment

The NIST-4 database was used to verify the proposed method [15]. This database consists of 4000 scanned images (at 512 × 512 resolution) obtained from two impressions (F and S) of 2000 fingerprints. Fingerprints were equally distributed into five classes, whorl (W), right loop (R), left loop (L), arch (A) and tented arch (T). Due to the ambiguity in fingerprint images, 350 fingerprints (17.5%) were cross-referenced with two classes. The first label was only considered in training the SVMs and the NB while both labels were used in the test. In the experiment, the fingerprints of the first impression were used as the training set (F0001 ∼ F2000), and the other fingerprints constructed the test set to follow the convention of the previous studies in this area. The FingerCode proposed by Jain [8] was used after normalization from −1 to 1, where some rejected images were included in the training set (1.4%) and the test set (1.85%). The LIBSVM package (available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm) provided the SVMs [30] using the Gaussian kernel with $\sigma^2 = 0.0625$ (determined according to Wu's method [31]).

4.2. SVMs with FingerCodes

Several schemes for generating and combining multiple SVMs were examined. Error-correcting codes and decision templates with HM and EU distances were used and the result is shown in Table 3. Among several fusion methods, the winner-takes-all method resulted in an accuracy of 90.1% with the OVA SVMs. Since it is possible that the FingerCode was rather more reliable than the singularities and pseudo codes, most cases obtained higher performance than the NB. Table 4 shows the confusion matrix of the winner-takes-all method with the OVA SVMs.

Table 4
Confusion matrix of the OVA SVMs of winner-takes-all method

True class	Assigned class				
	W	R	L	A	T
W	382	6	6	0	0
R	7	365	2	5	17
L	10	0	365	13	10
A	3	4	2	356	47
T	2	8	12	40	302

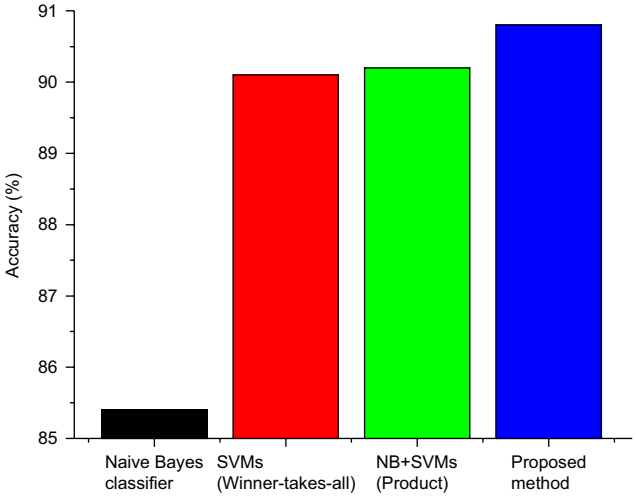

Fig. 7. Comparison with other methods.

Table 5
Confusion matrix of the proposed method

True class	Assigned class				
	W	R	L	A	T
W	373	10	10	0	0
R	4	374	0	6	15
L	5	0	377	8	9
A	0	6	4	365	40
T	1	8	15	39	295

4.3. Dynamically ordered SVMs

The proposed method, which combines NBs and SVMs dynamically, is aimed at classifying fingerprints by handling more subtle discriminations. The major classification task was performed by the SVMs, while the NB worked as an assistant by ordering them (although the NB obtained lower accuracy than the SVMs). It resulted in an accuracy of 90.8%, which is higher than the others as shown in Fig. 7 and Table 3. The confusion matrix is shown in Table 5. The combination

	W	R	L	A	T
NB	0.61	0.80	1.95	75.1	16.05
SVM	-1.29	-1.21	-1.16	0.07	0.34
True label				O	

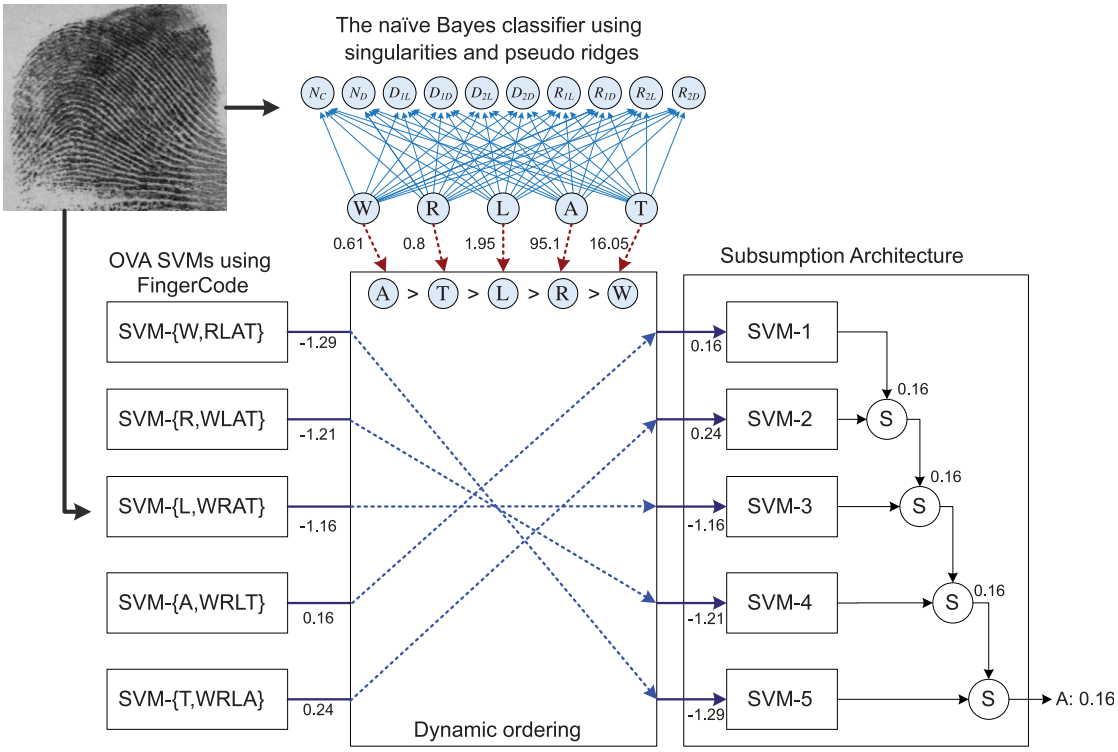

Fig. 8. A misclassified fingerprint by the OVA SVMs with the winner-takes-all method.

Table 6
Statistical significance analysis of the proposed method (five classes)

Fold no.	BN	SVMs	PRO	SUM	Proposed method
1	0.835	0.873	0.817	0.878	0.896
2	0.858	0.937	0.901	0.916	0.944
3	0.822	0.906	0.865	0.865	0.916
4	0.85	0.904	0.838	0.906	0.916
5	0.858	0.916	0.832	0.891	0.919
6	0.83	0.891	0.817	0.883	0.911
7	0.85	0.921	0.865	0.891	0.926
8	0.86	0.924	0.873	0.911	0.939
9	0.863	0.916	0.87	0.903	0.929
10	0.829	0.911	0.875	0.885	0.921
Avg.	0.8455	0.9099	0.8553	0.8929	0.9217
F	31.57040384				
p	1.52765E−12				

Table 7
Statistical significance analysis of the proposed method (four classes)

Fold no.	BN	SVMs	PRO	SUM	Proposed method
1	0.881	0.929	0.871	0.919	0.934
2	0.886	0.97	0.926	0.947	0.972
3	0.873	0.949	0.896	0.914	0.949
4	0.891	0.947	0.883	0.941	0.952
5	0.898	0.964	0.868	0.931	0.957
6	0.883	0.939	0.868	0.934	0.944
7	0.893	0.957	0.896	0.931	0.957
8	0.906	0.957	0.903	0.957	0.962
9	0.901	0.954	0.911	0.936	0.962
10	0.885	0.947	0.911	0.934	0.947
Avg.	0.8897	0.9513	0.8933	0.9344	0.9536
F	52.08063313				
p	2.53936E−16				

classifier of the NB and the SVMs was also examined. The probability and margin, obtained by the NB and the SVMs, respectively, was multiplied to generate a final result. The classifier produced an accuracy rate of 90.2%, which is higher than that of the NB or the SVMs. This means that the combination of different fingerprint features might complement each other and improve classification performance. The proposed method produced a higher improvement, since it managed the ambiguity of multiple SVMs more effectively.

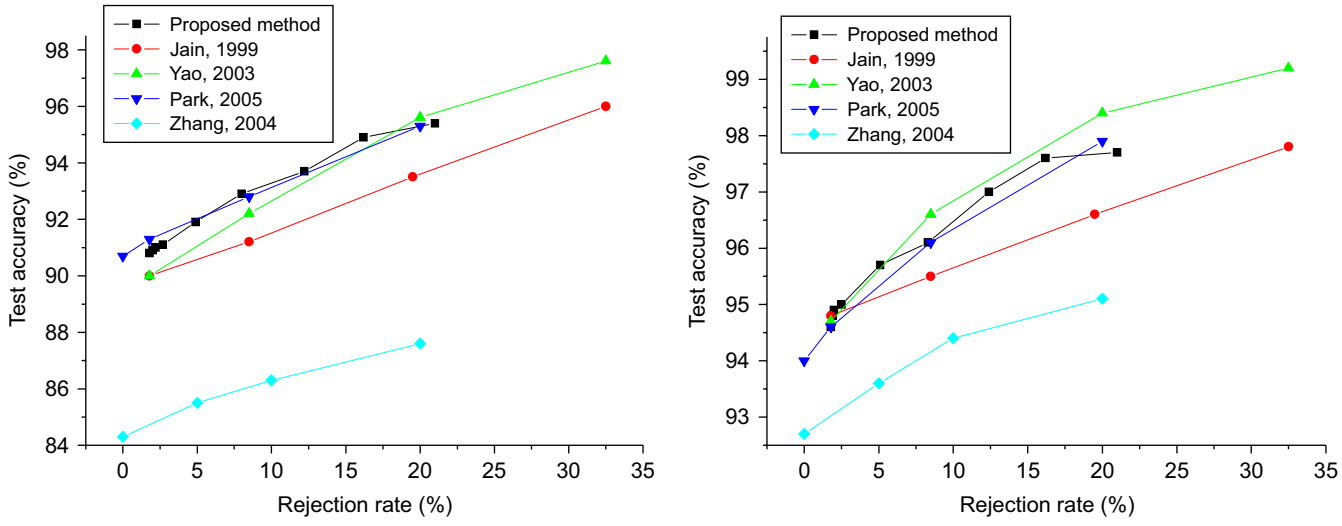

Fig. 9. Five-class (left) and four-class (right) accuracy-rejection plots of the NIST-4 database.

An example in Fig. 8, which often occurs in the test dataset, illustrates the usefulness of the proposed method. When the winner-takes-all method was used to combine the OVA SVMs, the fingerprint was classified as a tented arch. The proposed method, however, adjusted the order of the SVMs dynamically based on the probability obtained by the NB so as to correctly classify it as an arch.

4.4. Significance evaluation

In order to show the statistical significance of the differences between the performances of the methods, a one-way ANOVA test was conducted for the results of 10-fold cross-validation for five-class classification and four-class classification. As shown in Tables 6 (five classes) and 7 (four classes), the proposed method obtained statistically significant higher accuracy than the others including OVA SVMs of the winner-takes-all scheme ($F = 31.57$, $p < 0.05$ and $F = 52.08$, $p < 0.05$, respectively).

4.5. Comparison with related works

We have compared the proposed method with other methods that have previously been published. Several points are plotted in Fig. 9 along with the curve of possible points for the proposed method here and the methods of Jain et al. [8], Yao et al. [12], Park [9], and Zhang and Yan [6]. For each method, the accuracy of the classifier is shown with the corresponding rejection rate. As shown in Fig. 9, the proposed method yields competitive performance against the others in both classification tasks, especially for low rejection rate.

Zhang and Yan's [6] method, based on singular points and pseudo ridges, works fast, but the results are weak when noise is included in the fingerprint images. The proposed method used the FingerCode as well as structural features so as to be robust against noise. Furthermore, comparing the proposed method to Jain's work, singularity information helps improve the classification performance. Singular points and pseudo ridges, which are used in the proposed method, are easier to extract than the relational graph used in Yao's work, even though the classification results are similar to each other [12]. Parks' method uses 11,025-dimensional data that require much computation for highly accurate classification, while the proposed method uses only 202 features extracted from fingerprints [9].

5. Conclusions

In this paper we proposed a novel fingerprint classification method effectively integrating NBs and OVA SVMs, which produced better accuracy than previously reported in the literature contained in the NIST-4 database. Several popular fingerprint features such as singularities, pseudo codes and the FingerCode were used in the proposed method, and the combination of methods described here produced better results (90.8% for the five-class classification problem and 94.9% for the four-class classification problem with 1.8% rejection during the feature extraction phase of the FingerCode) than any of the component classifiers. The proposed method also breaks the tie that often occurs in multi-class classification by ordering the OVA SVMs based on the probability of classes. As future works, we will apply the proposed method of applying binary classifiers to other multi-class classification problems and exploit other methods instead of the NBs which might be more suitable for ordering OVA SVMs.

Acknowledgments

This work was supported by the Korea Science and Engineering Foundation (KOSEF) through the Biometrics Engineering Research Center (BERC) at Yonsei University. Also, the authors would like to thank Prof. A. Jain and Dr. S. Prabhakar for providing the FingerCode data.

References

[1] N. Yager, A. Amin, Fingerprint classification: a review, Pattern Anal. Appl. 7 (1) (2004) 77–93.

[2] H. Nyongesa, S. Al-khayatt, S. Mohamed, M. Mahmoud, Fast robust fingerprint feature extraction and classification, J. Intelligent Robotic Syst. 40 (1) (2004) 103–112.

[3] R. Cappelli, A. Lumini, D. Maio, D. Maltoni, Fingerprint classification by directional image partitioning, IEEE Trans. Pattern Anal. Mach. Intell. 21 (5) (1999) 402–421.

[4] A. Senior, A combination fingerprint classifier, IEEE Trans. Pattern Anal. Mach. Intell. 23 (10) (2001) 1165–1174.

[5] K. Karu, A. Jain, Fingerprint classification, Pattern Recognition 29 (3) (1996) 389–404.

[6] Q. Zhang, H. Yan, Fingerprint classification based on extraction and analysis of singularities and pseudo ridges, Pattern Recognition 37 (11) (2004) 2233–2243.

[7] K. Nagaty, Fingerprints classification using artificial neural networks: a combined structural and statistical approach, Neural Networks 14 (9) (2001) 1293–1305.

[8] A. Jain, S. Prabhakar, L. Hong, A multichannel approach to fingerprint classification, IEEE Trans. Pattern Anal. Mach. Intell. 21 (4) (1999) 348–359.

[9] C. Park, H. Park, Fingerprint classification using fast Fourier transform and nonlinear discriminant analysis, Pattern Recognition 38 (4) (2005) 495–503.

[10] M. Chong, T. Ngee, L. Jun, R. Gay, Geometric framework for fingerprint image classification, Pattern Recognition 30 (9) (1997) 1475–1488.

[11] J. Chang, K. Fan, A new model for fingerprint classification by ridge distribution sequences, Pattern Recognition 35 (6) (2002) 1209–1223.

[12] Y. Yao, G. Marcialis, M. Pontil, P. Frasconi, F. Roli, Combining flat and structured representations for fingerprint classification with recursive neural networks and support vector machines, Pattern Recognition 36 (2) (2003) 397–406.

[13] C. Hsu, C. Lin, A comparison of methods for multiclass support vector machines, IEEE Trans. Neural Networks 13 (2) (2002) 415–425.

[14] C. Watson, C. Wilson, Fingerprint Database, National Institute of Standard and Technology, Special Database 4, FPDB, 1992.

[15] E. Bredensteiner, K. Bennett, Multicategory classification by support vector machines, Comput. Optim. Appl. 12 (1–3) (1999) 53–79.

[16] D. Sebald, J. Bucklew, Support vector machines and the multiple hypothesis test problem, IEEE Trans. Signal Process. 49 (11) (2001) 2865–2872.

[17] L. Xu, A. Krzyzak, C. Suen, Methods of combining multiple classifiers and their applications to handwriting recognition, IEEE Trans. Syst. Man Cybern. 22 (3) (1992) 418–435.

[18] D. Ruta, B. Gabrys, Classifier selection for majority voting, Inf. Fusion 6 (1) (2005) 63–81.

[19] P. Boland, Majority systems and the condorcet jury theorem, Statistician 38 (1998) 181–189.

[20] J. Kittler, M. Hatef, R. Duin, J. Matas, On combining classifiers, IEEE Trans. Pattern Anal. Mach. Intell. 20 (3) (1998) 226–239.

[21] L. Kuncheva, A theoretical study on six classifier fusion strategies, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2) (2002) 281–286.

[22] K. Crammer, Y. Singer, On the learnability and design of output codes for multiclass problems, Mach. Learn. 47 (2–3) (2002) 201–233.

[23] T. Gestel, J. Suykens, G. Lanckriet, A. Lambrechts, B. Moor, J. Vandewalle, Multiclass LS-SVMs: moderated outputs and coding-decoding schemes, Neural Process. Lett. 15 (1) (2002) 45–58.

[24] S. Raudys, F. Roli, The behavior knowledge space fusion method: analysis of generalization error and strategies for performance improvement, Lect. Notes Comput. Sci. 2709 (2003) 55–64.

[25] L. Kuncheva, J. Bezdek, R. Duin, Decision templates for multiple classifier fusion: an experimental comparison, Pattern Recognition 34 (2) (2001) 299–314.

[26] L. Kuncheva, Using measures of similarity and inclusion for multiple classifier fusion by decision templates, Fuzzy Sets Syst. 122 (3) (2001) 401–407.

[27] R. Brooks, A robust layered control system for a mobile robot, IEEE J. Robotics Autom. 2 (1) (1986) 14–23.

[28] J. Liu, B. Li, T. Dillon, An improved naïve Bayesian classifier technique coupled with a novel input solution method, IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 31 (2) (2001) 249–256.

[29] M. Ramoni, P. Sebastiani, Robust Bayes classifiers, Artif. Intell. 125 (1–2) (2001) 209–226.

[30] S. Keerthi, C.-J. Lin, Asymptotic behaviors of support vector machines with Gaussian kernel, Neural Comput. 15 (7) (2003) 1667–1689.

[31] T.-F. Wu, C.-J. Lin, R. Weng, Probability estimates for multi-class classification by pairwise coupling, J. Mach. Learn. Res. 5 (2004) 975–1005.

About the Author—JIN-HYUK HONG received the B.S. and M.S. degrees in Computer Science from Yonsei University, Seoul, Korea, in 2002 and 2004, respectively. Since 2004, he is a P.h.D. student in the Department of Computer Science, Yonsei University. His research interests include evolutionary computation, conversational intelligent agent, and game strategy generation.

About the Author—JUNKI MIN received the B.S. and M.S. degree in Computer Science from Yonsei University, Seoul, Korea, in 2004 and 2006, respectively. Since 2006, he has been a Ph.D. student in the Department of Computer Science, Yonsei University. His research interests include pattern recognition, game strategy generation, image processing, and biometric.

About the Author—UNG-KEUN CHO received the B.S. degree in Computer Science from Yonsei University, Seoul, Korea, in 2005. Since 2005, he has been a M.S. student in the Department of Computer Science, Yonsei University. His research interests include image processing and biometric.

About the Author—SUNG BAE CHO received the B.S. degree in Computer Science from Yonsei University, Seoul, Korea, in 1988 and the M.S. and Ph.D. degrees in Computer Science from KAIST (Korea Advanced Institute of Science and Technology), Taejeon, Korea, in 1990 and 1993, respectively. He has worked as a Member of the Research at the Center for Artificial Intelligence Research at KAIST from 1991 to 1993. He was an Invited Researcher of Human Information Processing Research Laboratories at ATR (Advanced Telecommunications Research) Institute, Kyoto, Japan from 1993 to 1995, and a visiting scholar at University of New South Wales, Canberra, Australia in 1998. Since 1995, he has been an associate professor in the Department of Computer Science, Yonsei University. His research interests include neural networks, pattern recognition, intelligent man–machine interfaces, evolutionary computation, and artificial life. Dr. Cho was awarded outstanding paper prizes from the IEEE Korea Section in 1989 and 1992, and another one from the Korea Information Science Society in 1990. He was also the recipient of the Richard E. Merwin prize from the IEEE Computer Society in 1993. He was listed in Who's Who in Pattern Recognition from the International Association for Pattern Recognition in 1994 and received the best paper awards at International Conference on Soft Computing in 1996 and 1998. Also, he received the best paper award at World Automation Congress in 1998 and listed in Marquis Who's Who in Science and Engineering in 2000 and in Marquis Who's Who in the World in 2001. He is a member of the Korea Information Science Society, INNS, the IEEE Computer Society, and the IEEE Systems, Man, and Cybernetics Society.