Advance Access publication on 5 January 2012
doi:10.1093/comjnl/bxr131

A Local Mean-Based *k*-Nearest Centroid Neighbor Classifier

JIANPING GOU[1,*], ZHANG YI[2], LAN DU[3] AND TAISONG XIONG[1]

[1]*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610054, P.R. China*
[2]*College of Computer Science, Sichuan University, Chengdu 610065, P.R. China*
[3]*College of Engineering and Computer Science, The Australian National University, Canberra 2601, Australia*
**Corresponding author: cherish.gjp@gmail.com*

***K*-nearest neighbor (KNN) rule is a simple and effective algorithm in pattern classification. In this article, we propose a local mean-based *k*-nearest centroid neighbor classifier that assigns to each query pattern a class label with nearest local centroid mean vector so as to improve the classification performance. The proposed scheme not only takes into account the proximity and spatial distribution of *k* neighbors, but also utilizes the local mean vector of *k* neighbors from each class in making classification decision. In the proposed classifier, a local mean vector of *k* nearest centroid neighbors from each class for a query pattern is well positioned to sufficiently capture the class distribution information. In order to investigate the classification behavior of the proposed classifier, we conduct extensive experiments on the real and synthetic data sets in terms of the classification error. Experimental results demonstrate that our proposed method performs significantly well, particularly in the small sample size cases, compared with the state-of-the-art KNN-based algorithms.**

Keywords: K-nearest neighbor rule; nearest centroid neighborhood; K-nearest centroid neighbor rule; local mean vector; pattern classification

Received 1 September 2011; revised 11 November 2011
Handling editor: Ethem Alpaydin

1. INTRODUCTION

In pattern classification, the *k*-nearest neighbor (KNN) rule, first introduced by Fix and Hodges [1], is one of the most straightforward nonparametric techniques. The basic rationale for the KNN rule is such that: find KNNs for a query pattern, and then among these nearest neighbors, the most represented class is assigned to the query pattern by a majority voting. The 1-NN rule is a special form of KNN, when $k = 1$. Apart from the conceptual and implementational simplicity of the KNN rule, it has been proved that the KNN rule has asymptotically optimal performance in the Bayes sense. That is to say, the asymptotic classification error rate of the KNN rule (when the number of prototypes $n \to \infty$, the neighborhood size $k \to \infty$, and $k/n \to 0$) approaches the Bayes error rate, and the error rate of 1-NN is bounded by twice the Bayes error rate [2–4]. However, the small training sample size always dramatically degrades the classification accuracy so as to hardly achieve the expected asymptotic performance in the KNN-based algorithms, especially in the case of the outliers [5].

It has been well known that the performance of the nonparametric classifiers is seriously influenced by the existing outliers in the small training sample size settings [5]. In order to overcome this issue, Mitani and Hamamoto have proposed a reliable local mean-based *k*-nearest neighbor algorithm (LMKNN), which employs the local mean vector of each class to classify query patterns [6, 7]. Since the LMKNN is first introduced in [7], the idea of the LMKNN has already successfully applied in the group-based classification [8], discriminant analysis [9] and distance metric learning [10]. Subsequently, to resolve the negative effect of outliers on classification performance, Zeng *et al.* [11] have designed a pseudo nearest neighbor rule utilizing the distance weighted local learning, and another nonparametric classifier based on both local mean vector and class statistics [12]. Although these algorithms that are robust to outliers in [6, 7, 11, 12] do well

in classification performance to some extent, their sensitivity to the small sample size still exists.

As an alternative neighborhood, one of the surrounding neighborhoods, called nearest centroid neighborhood (NCN), is a successful neighborhood [13–15]. Accordingly, the k-Nearest Centroid Neighbor (KNCN) rule, based on the concept of NCN, has been first introduced in KNN-based pattern classification in [14]. The KNCN rule tries to consider both the distance-based proximity and spatial distribution of k neighbors in training set. Even though the theoretical analysis about the classification behavior of the KNCN rule has not been shown in the asymptotic case, many experimental studies have indicated that it works very well, particularly in cases with small sample size [14–20]. The variants of KNCN achieve better performance in prototype-based classification, such as prototype selection and prototype optimization [17–20]. In addition, Sánchez and Marqués gave an intensive review of the KNCN-based methods in pattern classification [21]. Despite the fact that KNCN always outperforms KNN, there are still some problems yet to be resolved in the KNCN-based pattern classification. First, the overestimate of the importance of some neighbors may result in unreliable classification performance [16, 22]. This is because the NCN may contain some prototypes that are not sufficiently close to the query pattern. Second, as in the KNN, the assumption that k centroid neighbors have an identical weight for making classification decision is inappropriate. Finally, due to a majority voting for making a decision in the KNCN, the ties of vote can easily give rise to the mistaken classification results.

In view of the foregoing discussions, we are motivated to propose a local mean-based k-nearest centroid neighbor (LMKNCN) classifier, in which the class label with a nearest local centroid mean vector is assigned to the query pattern. The local centroid mean vector of each class for a query pattern not only takes into account the proximity of k nearest neighbors, but also considers their geometrical distribution. The purpose of the proposed LMKNCN classifier is to mainly handle the problems that existed in the KNCN and LMKNN, so as to improve the classification performance. The proposed scheme, based on local mean vector and NCN, is a very advisable and promising approach, because a local mean vector of KNCNs for each class with respect to a query pattern is well positioned to sufficiently capture the class distribution information. The experimental results on real and synthetic data sets show that our proposed method performs well in terms of the classification error.

This article is organized as follows. In Section 2, we briefly summarize the related work, which motivates our work. In Section 3, we propose a local mean-based KNCN classifier. Section 4 reports in detail the classification results of the experimental comparisons between LMKNCN and KNN, LMKNN, KNCN on both real and artificial data sets. In Section 5, we empirically discuss why the proposed LMKNCN is robust to the outliers. Finally, we conclude this article in Section 6.

2. RELATED WORK

2.1. The LMKNN classifier

The local mean-based KNN algorithm (LMKNN), as a successful extension of the KNN rule, is a simple and robust classifier in the small sample size cases [7]. The goal of LMKNN is to overcome the negative effect of the existing outliers in the training set (TS). The rationale behind this method is that the local mean vector of k nearest neighbors in each class is employed to classify the query pattern in making classification decision.

Let $T = \{x_n \in \mathbb{R}^m\}_{n=1}^{N}$ be a TS of given m-dimensional feature space, where N is the total number of training samples, and $y_n \in \{c_1, c_2, \ldots, c_M\}$ denotes the class label for x_n. $T_i = \{x_{ij} \in \mathbb{R}^m\}_{j=1}^{N_i}$ denotes a subset in T from the class c_i, with the number of the training samples N_i. In the LMKNN rule, a query pattern x is determined by the following steps:

(i) Search the k nearest neighbors from the set T_i of each class c_i for the query pattern x. Let $T_{ik}^{NN}(x) = \{x_{ij}^{NN} \in \mathbb{R}^m\}_{j=1}^{k}$ be the set of KNNs for x in the class c_i using the Euclidean distance metric, i.e. Equation (1). Note that the value of k is $\leq N_i$.

$$d(x, x_{ij}^{\mathrm{NN}}) = \sqrt{(x - x_{ij}^{\mathrm{NN}})^{\mathrm{T}}(x - x_{ij}^{\mathrm{NN}})}. \tag{1}$$

(ii) Calculate the local mean vector u_{ik}^{NN} from the class c_i, using the set $T_{ik}^{\mathrm{NN}}(x)$.

$$u_{ik}^{\mathrm{NN}} = \frac{1}{k}\sum_{j=1}^{k} x_{ij}^{\mathrm{NN}}. \tag{2}$$

(iii) Assign x to the class c if the distance between the local mean vector for c and the query pattern in Euclidean space is minimum.

$$c = \arg\min_{c_i}(x - u_{ik}^{\mathrm{NN}})^{\mathrm{T}}(x - u_{ik}^{\mathrm{NN}}). \tag{3}$$

Note that the LMKNN classifier is equivalent to the 1-NN rule when $k = 1$.

2.2. The NCN

NCN is a good alternative to nearest neighborhood (NN) [13]. The basic idea of NCN is that the neighbors are as close to a query pattern as possible, but also the neighbors are distributed as geometrically around that pattern as possible. Just like the NN, the NCN takes into account the nearness of the neighbors; however, the spatial distribution of the neighbors is not considered in the NN. Thereby, for a query pattern x, NCN

should be subject to two complementary constraints:

(i) The distance criterion: the centroid neighbors should be as close to x as possible.
(ii) The symmetry criterion: the centroid neighbors should be placed as homogeneously around x as possible.

The centroid of a set of points $S = \{s_1, s_2, \ldots, s_q\}$ can be calculated as follows:

$$s_q^c = \frac{1}{q}\sum_{i=1}^{q} s_i. \tag{4}$$

It should be noted that Equation (4) is similar to Equation (2), but they function completely different. Thus, given a TS, $T = \{x_n \in \mathbb{R}^m\}_{n=1}^N$, the nearest centroid neighbors of a query pattern x with both earlier-mentioned conditions are found through an iterative procedure in the following two steps:

(i) The first nearest centroid neighbor of x corresponds to its nearest neighbor, say x_1^{NCN}.
(ii) The ith nearest centroid neighbor, x_i^{NCN} ($i \geq 2$) is such that the ith centroid is defined as the centroid of a new point in T_i plus all previous centroid neighbors, i.e. $x_1^{\mathrm{NCN}}, \ldots, x_{i-1}^{\mathrm{NCN}}$, and the distance between the ith centroid and x is the shortest.

Note that, to compute the ith centroid, all points in T_i that were not considered in computing all the $i-1$ previous centroids must be tested in order to find the centroid to x.

Through the procedure, the selected neighbors in the kind of neighborhood not only have the spatial distribution surrounding a query pattern due to the centroid criterion, but also guarantee the proximity to the query pattern because of the first nearest neighbor (i.e. the first centroid neighbor) to satisfy the incremental nature.

2.3. The KNCN classifier

According to the NCN concept, the KNCN rule is introduced accordingly in pattern classification [14]. It has been empirically found that KNCN is an effective method in the finite sample size situations [14–20]. Compared with the benchmark KNN, the KNCN tries to estimate the class of a query pattern reliably with both the proximity and symmetrical placement of the neighbors by a majority voting in making classification decision.

Let $T = \{x_n \in \mathbb{R}^m\}_{n=1}^N$ be a TS with M classes $c_1, \ldots, c_M$. Given a query pattern x, the KNCN classifier can be carried out as follows:

(i) Search k-nearest centroid neighbors of x from T, say $T_k^{\mathrm{NCN}}(x) = \{x_i^{\mathrm{NCN}} \in \mathbb{R}^m\}_{i=1}^k$.
(ii) Assign x to the class c, which is most frequently represented by the centroid neighbors in the set $T_k^{\mathrm{NCN}}(x)$ (resolve ties randomly).

$$c = \arg\max_{c_j} \sum_{x_i^{\mathrm{NCN}} \in T_k^{\mathrm{NCN}}(x)} \delta(c_j = y_i^{\mathrm{NCN}}), \tag{5}$$

where y_i^{NCN} is the class label for the ith centroid neighbor x_i^{NCN}, and $\delta(c_j = y_i^{\mathrm{NCN}})$, the Kronecker delta function, takes a value of one if $c_j = y_i^{\mathrm{NCN}}$ and zero otherwise.

Similar to the KNN, a set of KNCNs for a query pattern is selected from TS, and then the class label with the greatest number of votes among the neighbors is assigned to the query pattern. However, unlike the KNN, the KNCN finds k neighbors that surround the query pattern.

3. THE PROPOSED LMKNCN CLASSIFIER

In statistical pattern recognition, the classification performance of the KNN-based non-parametric classifiers is seriously influenced by the outliers, especially in the case of small training sample size. As we know, the LMKNN is robust to outliers by using the local mean vector of each class, and the KNCN is effective in the small sample size case due to the NCN concept. Combining the robustness of LMKNN and effectiveness of KNCN, we propose a new approach in regard to the classification performance.

3.1. The local mean-based KNCN classifier

Let $T = \{x_n \in \mathbb{R}^m\}_{n=1}^N$ be a TS with M classes $c_1, \ldots, c_M$, and $T_i = \{x_{ij} \in \mathbb{R}^m\}_{j=1}^{N_i}$ be a class TS of c_i, each of which consists of N_i training samples. In our proposed local mean-based KNCN classifier, given a query pattern x, its class label is predicted through the following steps:

(i) Find the set of KNCNs from the set T_i of each class c_i for the query pattern x, say $T_{ik}^{\mathrm{NCN}}(x) = \{x_{ij}^{\mathrm{NCN}} \in \mathbb{R}^m\}_{j=1}^k$, due to the NCN criterion. Note that the value of k is no more than N_i.
(ii) Compute the local centroid mean vector u_{ik}^{NCN} from each class c_i using the set $T_{ik}^{\mathrm{NCN}}(x)$.

$$u_{ik}^{\mathrm{NCN}} = \frac{1}{k}\sum_{j=1}^{k} x_{ij}^{\mathrm{NCN}}. \tag{6}$$

(iii) Calculate the distance $d(x, u_{ik}^{\mathrm{NCN}})$ between x and the local centroid mean vector u_{ik}^{NCN} according to Equation (1).
(iv) Assign x to the class c, which has the closest distance between its local centroid mean vector and the query pattern x.

$$c = \arg\min_{c_i} d(x, u_{ik}^{\mathrm{NCN}}). \tag{7}$$

It should be noted that the LMKNCN is same as the LMKNN and 1-NN when $k = 1$.

From procedures of these two classifiers, the LMKNCN is quite similar to the LMKNN. However, the local mean vector in the proposed LMKNCN preferably represents the class distribution information by means of taking into account both the proximity and the spatial position of neighbors around the query pattern. This means that the class label that has a nearest local centroid mean vector is allocated to a given query pattern, not only according to nearest neighbors but also considering how patterns are symmetrically placed around the query pattern in the class. Compared with the KNCN, the LMKNCN selects k centroid neighbors of each class, and classifies a query pattern into the class with a closest local centroid mean vector.

3.2. Difference between the LMKNCN and the LMKNN

Being an extension of the LMKNN, the proposed LMKNCN is similar to it. Nevertheless, the LMKNCN employs more hidden information in geometrical distribution among k neighbors to make classification decision. In order to make our proposed idea more intuitive, for instance, the comparison between the LMKNCN and the LMKNN on two-class classification problem is schematically illustrated in Fig. 1 with $k = 5$ in a straight-forward manner. Note that and u_1^{NN} and u_2^{NN}, respectively, indicate the local mean vectors of class 1 and 2 in the LMKNN, u_1^{NCN} and u_2^{NCN} the local centroid mean vectors in the LMKNCN. As can be seen in Fig. 1, the query pattern x can be assigned to its true class (i.e. class 1) by the LMKNCN, rather than the false class (i.e. class 2) by the

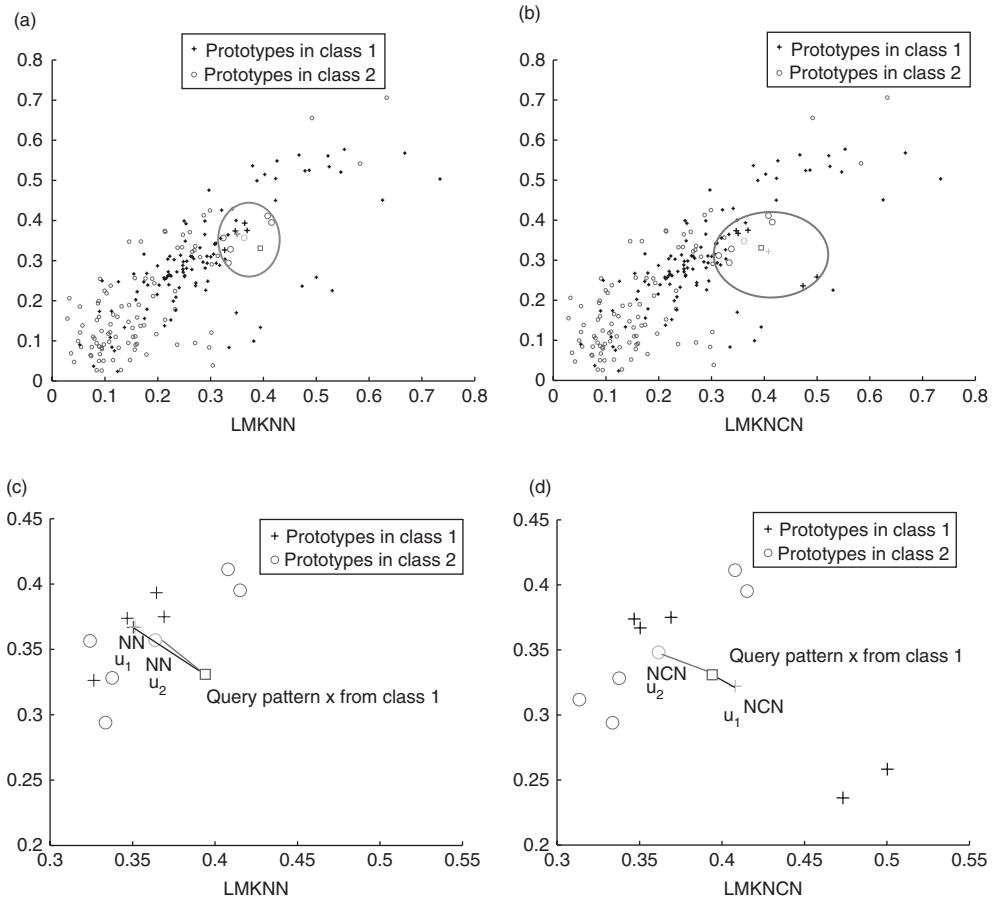

FIGURE 1. A classification comparison between LMKNCN and LMKNN with $k = 5$ on two-class classification problem in two-dimensional feature space. The true class of the given query pattern x (blue square) is class 1. The local mean vectors of class 1 and 2, denoted by green cross and circles respectively, are computed by using KNN in (**a**) and (**c**), and using KNCN in (**b**) and (**d**). The black and red solid lines in (c) and (d) indicate the distances between the local mean vectors of class 1 and 2 to x, respectively. Note that, to clearly understand, (a) and (b) depict the full sample spaces. The two pink ellipses in (a) and (b) are enlarged in (c) and (d), respectively. So the points in (c) are the same as the corresponding bigger ones included in the ellipse in (a), and the points in (d) the same as the corresponding bigger ones in the ellipse in (b).

LMKNN. We can see that the local mean vector of k nearest neighbors is quite different from that of KNCNs in each class, because some nearest neighbors are different from some nearest centroid neighbors as shown in Fig. 1. Moreover, the local centroid mean vector in the LMKNCN can well represent the class distribution information around a given query pattern. The reason is that LMKNCN takes into account the proximity and geometrical distribution of neighbors due to the NCN definition. It can also be observed that the region of the influence of the NCN is often larger than or equal to that of the NN. This graphical representation well clarify the general behavior of the LMKNCN and LMKNN schemes.

Consequently, the proposed LMKNCN scheme, which combines NCN with local mean vector for each class, can be more suitable for making classification decision, because a local mean vector of KNCNs for each class to a query pattern is well positioned to sufficiently capture the class distribution information.

3.3. The LMKNCN algorithm

We summarize the proposed LMKNCN classifier in Algorithm 1 by means of pseudo code. In the LMKNCN algorithm, the Euclidean distance is employed for simplicity, although some other metrics, e.g. the Mahalanobis distance, can be used as well according to different applications.

4. EXPERIMENTAL RESULTS

In pattern classification, the error rate is one of the most effective measures to estimate the performance of algorithms. To study the classification behavior of the proposed classifier in depth, we conduct sets of experiments on both real and artificial data sets in terms of the classification error, and compare it with all the aforementioned state-of-the-art classifiers. The classification performance is investigated by considering: performance with the lowest error rate and the corresponding value of k, performance with varying the neighborhood size k, feature space dimension and the training sample size. Note that in the KNN and KNCN, the neighborhood size k is for all training samples of a query pattern, while is for training samples of each class in the LMKNN and LMKNCN.

4.1. Experimental data sets

The 15 real benchmark data sets in our experiments are sourced from the UCI Repository [23]. For short, among these data sets, the abbreviated names for Ionosphere, Transfusion, Libras Movement, Cardiotocography, Image Segmentation, Robot Navigation and Thyroid Disease are Iono, Trans, Libras, Cardio, Image, Robot and Thyroid, respectively. Their main characteristics: the number of classes, instances and features are described in Table 1. These real data sets only have numeric attributes, which make it straightforward to use Euclidean distance. Among all the real data sets, there are six data sets that belong to two-class classification tasks, while the others are multi-class classification tasks.

TABLE 1. The real data sets used in the experiments.

Dataset	Instances	Features	Classes	Testing set
Iono	351	34	2	151
Pima	768	8	2	268
Wdbc	569	30	2	169
Libras	360	90	15	90
Wine	178	13	3	58
Heart	267	44	2	67
Sonar	208	60	2	58
Trans	748	4	2	248
Vehicle	846	18	4	200
Image	2310	19	7	910
Cardio	2126	21	10	500
Robot	5456	4	4	1086
Thyroid	7200	21	3	2000
Optdigits	5620	64	10	1463
Wave	5000	40	3	2000

Four synthetic Gaussian data sets: I-$\wedge$, I-4I, I-I [5] and Ness [24], are adopted in our experiments. The main reason for utilizing the four artificial data sets is that the number of the available samples can be easily adjusted so that the training and testing sample size can be controlled [7, 11]. These four data sets, each with two classes, are randomly generated by using singular normal density functions and each class has an identical prior probability. In these artificial data sets, u_i is the mean vector and Σ_i is the covariance matrix from class c_i. Specifically, they are generated with the following settings using the p-dimensional Gaussian distribution:

I-$\wedge$ data set: The I-$\wedge$ data set consists of eight-dimensional Gaussian data.

$$u_1 = 0, \quad \Sigma_1 = I_8,$$
$$u_2 = [3.86, 3.10, 0.84, 0.84, 1.64, 1.08, 0.26, 0.01]^{\mathrm{T}},$$
$$\Sigma_2 = \mathrm{diag}[8.41, 12.06, 0.12, 0.22, 1.49, 1.77, 0.35, 2.73],$$

where I_p and diag[$\cdot$] denote the $p \times p$ identity and diagonal matrices, respectively.

I-4I data set: The I-4I data set consists of eight-dimensional Gaussian data.

$$u_1 = u_2 = 0,$$
$$\Sigma_1 = I_8, \quad \Sigma_2 = 4I_8.$$

Algorithm 1 The local mean-based k-nearest centroid neighbor algorithm.

Require:

x: a query pattern, k: the neighborhood size.

$T = \{x_n \in \mathbb{R}^m\}_{n=1}^{N}$: a TS, $T_i = \{x_{ij} \in \mathbb{R}^m\}_{j=1}^{N_i}$: a class TS. M: the number of classes in T, c_1, ..., c_M: M class labels, N_1, ..., N_M: the number of training samples in T_i.

Ensure:

Predict the class label of a query pattern by a nearest local centroid mean vector among classes.

Step 1: Calculate the distances of training samples in each class c_i to x.

for $j = 1$ to N_i **do**

$d(x, x_{ij}) = \sqrt{(x - x_{ij})^T (x - x_{ij})}$

end for

Step 2: Find the first nearest centroid neighbor of x in each class c_i, say x_{i1}^{NCN}.

$[min\_index, min\_dist] = min(d(x, x_{ij}))$

$x_{i1}^{NCN} = x_{min\_index}$

Set $R_i^{NCN}(x) = \{x_{i1}^{NCN} \in \mathbb{R}^m\}$

Step 3: Search k nearest centroid neighbors of x except the first one, $T_{ik}^{NCN}(x) = \{x_{ij}^{NCN} \in \mathbb{R}^m\}_{j=1}^{k}$, in each class c_i.

for $j = 2$ to k **do**

Set $S_i(x) = T_i - R_i^{NCN}(x)$

$S_i(x) = \{x_{il} \in \mathbb{R}^m\}_{l=1}^{L_i(x)}$, $L_i(x) = length(S_i(x))$

Compute the sum of the previous $j - 1$ nearest centroid neighbors.

$sum_i^{NCN}(x) = \sum_{r=1}^{j-1} x_{ir}^{NCN}$

Calculate the centroids in the set S_i for x.

for $l = 1$ to $L_i(x)$ **do**

$x_{il}^c = \frac{1}{j}(x_{il} + sum_i^{NCN}(x))$

$d_{il}^c(x, x_{il}^c) = \sqrt{(x - x_{il}^c)^T (x - x_{il}^c)}$

end for

Find the j^{th} nearest centroid neighbor.

$[min\_index^{NCN}, min\_dist^{NCN}] = min(d_{il}^c(x, x_{il}^c))$

$x_{ij}^{NCN} = x_{min\_index^{NCN}}$

Add x_{ij}^{NCN} to the set R_i^{NCN}.

end for

Set $T_{ik}^{NCN}(x) = R_i^{NCN}(x)$.

Step 4: Calculate the local centroid mean vector u_{ik}^{NCN} in set $T_{ik}^{NCN}(x)$ for each class c_i.

$u_{ik}^{NCN} = \frac{1}{k} \sum_{j=1}^{k} x_{ij}^{NCN}$

Step 5: Assign x to the class c with a nearest local centroid mean vector.

$c = \arg\min_{c_i} d(x, u_{ik}^{NCN})$

I-I data set: The I-I data set consists of p-dimensional Gaussian data, and the dimensionality p can be controlled.

$$u_1 = 0, \quad u_2 = [2.56, 0, \ldots, 0]^{\mathrm{T}},$$
$$\Sigma_1 = \Sigma_2 = I_p.$$

Ness data set: The Ness data set consists of p-dimensional Gaussian data.

$$u_1 = 0, \quad u_2 = [\Delta/2, 0, \ldots, 0, \Delta/2]^{\mathrm{T}},$$
$$\Sigma_1 = I_p, \quad \Sigma_2 = \begin{bmatrix} I_{p/2} & O \\ O & \frac{1}{2} I_{p/2} \end{bmatrix}.$$

It is noted that the values of p and Δ can be controlled, and Δ is set to be 2, 4 and 6 in our experiments.

4.2. Experiments on real data sets

In order to objectively investigate the classification behavior of the proposed LMKNCN method, comparative studies on LMKNCN, LMKNN, KNN and KNCN are first performed on the real data sets. Our experiments are carried out on 15 UCI data sets in Table 1 in terms of the classification error by 10-fold cross validation. As shown in Table 1, training samples are randomly chosen from each real data set, while the remaining samples are used for testing. We run the system 10 times and obtain

TABLE 2. The lowest error rates (%) of each method with the corresponding stds and values of k in the parentheses for the 15 UCI data sets (the best recognition performance is described in bold-face on each data set).

Data set	KNN	KNCN	LMKNN	LMKNCN
Pima	25.34 ± 1.21 (15)	25.37 ± 1.42 (14)	25.86 ± 1.63 (14)	**24.48 ± 1.41 (15)**
Libras	15.00 ± 2.75 (1)	15.00 ± 2.75 (1)	14.00 ± 3.56 (3)	**13.00 ± 3.32 (2)**
Iono	14.83 ± 1.53 (1)	7.88 ± 1.17 (9)	11.99 ± 1.27 (3)	**7.48 ± 1.05 (5)**
Wine	29.14 ± 2.75 (1)	22.93 ± 3.99 (11)	26.90 ± 2.92 (6)	**22.59 ± 4.95 (8)**
Heart	21.34 ± 3.15 (15)	18.21 ± 1.65 (15)	20.60 ± 1.94 (10)	**17.61 ± 1.58 (10)**
Sonar	17.76 ± 3.08 (1)	16.03 ± 4.27 (6)	17.07 ± 4.33 (6)	**12.41 ± 2.84 (3)**
trans	24.35 ± 0.89 (12)	24.23 ± 1.19 (14)	24.76 ± 0.94 (8)	**23.71 ± 0.95 (6)**
vehicle	34.00 ± 1.79 (4)	27.95 ± 1.65 (7)	29.50 ± 1.49 (3)	**26.35 ± 2.09 (3)**
Cardio	27.22 ± 1.43 (1)	25.62 ± 1.14 (13)	27.22 ± 1.43 (1)	**24.90 ± 1.15 (2)**
Wave	16.23 ± 0.53 (15)	16.91 ± 0.53 (15)	16.54 ± 0.68 (15)	**14.61 ± 0.56 (15)**
Wdbc	7.10 ± 1.61 (10)	5.98 ± 1.01 (11)	7.16 ± 0.95 (6)	**5.74 ± 0.77 (8)**
Image	4.74 ± 0.60 (1)	4.47 ± 0.40 (6)	4.45 ± 0.21 (3)	**4.05 ± 0.32 (2)**
Robot	2.62 ± 0.19 (1)	2.44 ± 0.23 (4)	2.62 ± 0.19 (1)	**2.41 ± 0.26 (3)**
Thyroid	2.51 ± 0.08 (10)	2.11 ± 0.12 (9)	2.40 ± 0.15 (15)	**1.80 ± 0.15 (12)**
optdigits	1.29 ± 0.19 (4)	0.92 ± 0.13 (9)	0.94 ± 0.11 (5)	**0.85 ± 0.12 (3)**

10 different training and testing sample sets for performance evaluation. The final performance is achieved by averaging ten classification error rates with 95% confidence. Note that the value of the neighborhood size k in the experiments is preset in the interval ranged from 1 to 15. The optimal or sub-optimal value of k on each real data set that obtains the lowest error rate is chosen within the interval.

The empirical comparisons, shown in Table 2, are evaluated on each real data set by means of the lowest error with the corresponding standard deviations (stds) and values of k. As disclosed by the classification results in Table 2, the proposed LMKNCN classifier performs better than other three methods on all real data sets. It is noticeable that the achieved optimal performance of LMKNCN is significantly better than that of KNN in all classification tasks, making the superiority of the proposed LMKNCN evidently. More interestingly, we can find two facts from Table 2. One is that the lowest error rate of LMKNCN is somewhat similar to that of KNCN on each data set, which is mainly due to the use of NCN. However, the other one is that the optimal performance of LMKNCN in all cases is superior to that of KNCN, which may be contributed by using the local mean vector for each class. Therefore, the promising performance of our proposed LMKNCN classifier is first confirmed on all the real data sets in terms of the lowest classification error with best parameter k.

To further verify the superiority of the proposed LMKNCN, the classification behavior of LMKNCN with varying the neighborhood size k is studied on the real data sets. It is compared with KNN, KNCN and LMKNN, respectively. Their experimental results in terms of the classification error are shown in Figs 2 and 3. As is to be expected, LMKNCN very often outperforms other classifiers with different values of k. It is clear that the performance of LMKNCN is almost superior to that of KNN and LMKNN on each data set while varying the k value. We can observe that the differences between LMKNCN and KNN, LMKNN are very significant in most cases. In the meantime, LMKNCN is always preferable to KNCN with different values of k on all the data sets except for Iono, Image, Libras, optdigits and Cardio. However, LMKNCN obtains the best optimums on these four data sets among these four classifiers at a small value of k. The reason for the classification results on these four data sets may be that the neighborhood size k in KNCN is for all training samples of a query pattern, while is for training samples of each class in LMKNCN. As a consequence, we could draw a conclusion that our proposed LMKNCN method can be an effective algorithm with respect to the neighborhood size k.

As shown in Table 2 and Figs 2 and 3, the proposed LMKNCN works well by reducing the influence of the outliers due to the nearest local centroid mean vector, particularly in the small sample size cases. In order to intuitively investigate the performance of the proposed method, Table 3 illustrates the classification results and the distances between a local mean vector and a given query pattern on Wine data set with each value of k in the LMKNCN and LMKNN classifiers. The given query pattern belongs to class 2. Note that some distances in the LMKNN and LMKNCN at a certain value of k are identical, and this is because the k nearest neighbors may be the same as the k nearest centroid neighbors. It is important to note that the class label, which has a nearest local (centroid) mean vector, is assigned to the query pattern. As shown in Table 3, it is clear that the given query pattern is mistakenly classified by LMKNN when $k = 1, \ldots, 5$ and LMKNCN when $k = 1, 2$, but correctly classified by LMKNCN when $k = 3, 4, 5$, according

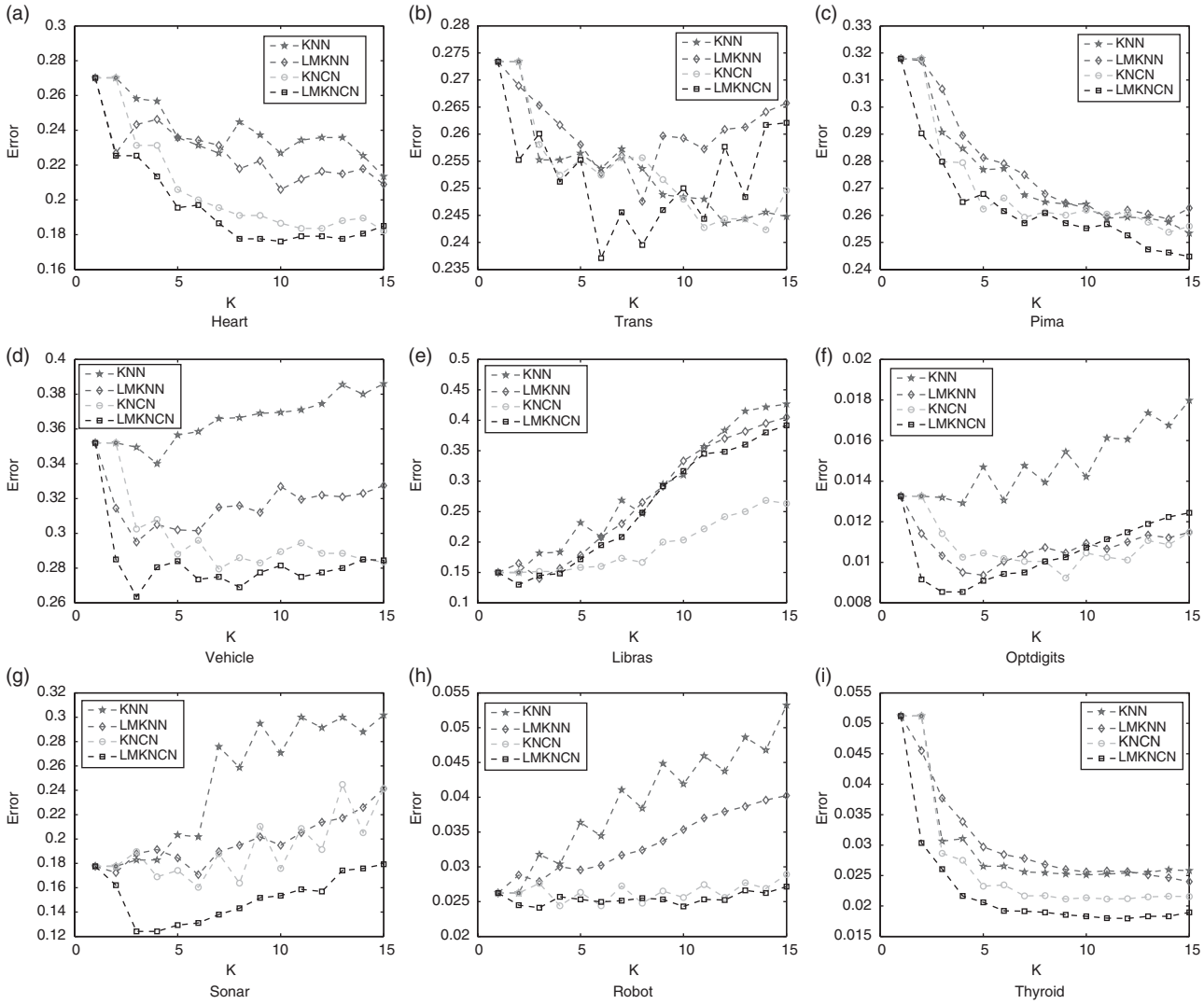

FIGURE 2. The error rates of LMKNCN, LMKNN, KNCN and KNN via the neighborhood size k on the nine real data sets.

to Equations (3) and (7). We can easily see that the query pattern does not belong to the class 1, and the distances of local mean vectors for the class in the LMKNN and LMKNCN to the query pattern grow larger as k increases. We have also observed that the local mean vector of LMKNN for class 2 becomes far away from the query pattern while k increases, but that of LMKNCN grows close to the query pattern. In LMKNN classifier, the local mean vector for class 3 at each k value is always closer to the query pattern than that for class 2, so that the query pattern is wrongly classified to class 3. However, in the LMKNCN, the local mean vectors for class 2 are farther to the query pattern than that for class 3 when $k = 1, 2$ with wrong classification, but when $k = 3, 4, 5$ the local mean vectors for class 2 are nearer to the query pattern than that for class 3 with correct classification. Thereby, the results in Table 3 demonstrate in detail that the local mean vector of k nearest centroid neighbors for each class is more reliable than that of k nearest neighbors with a certain value of k in many practical settings.

4.3. Experiments on synthetic data sets

In order to show the effectiveness of the proposed LMKNCN classifier, we further study the classification performance of LMKNCN on the I-$\wedge$, I-4I, I-I and Ness data sets, compared with KNN, KNCN and LMKNN. The classification performance is evaluated by varying the neighborhood size k, the training sample size N and feature space dimension p. For these synthetic data sets, the performance of the four classifiers is estimated in terms of the classification error. In order to obtain reliable classification, the trials on each model are repeated 10 times. That is to say, training and testing samples are randomly generated by each artificial model 10 times and the

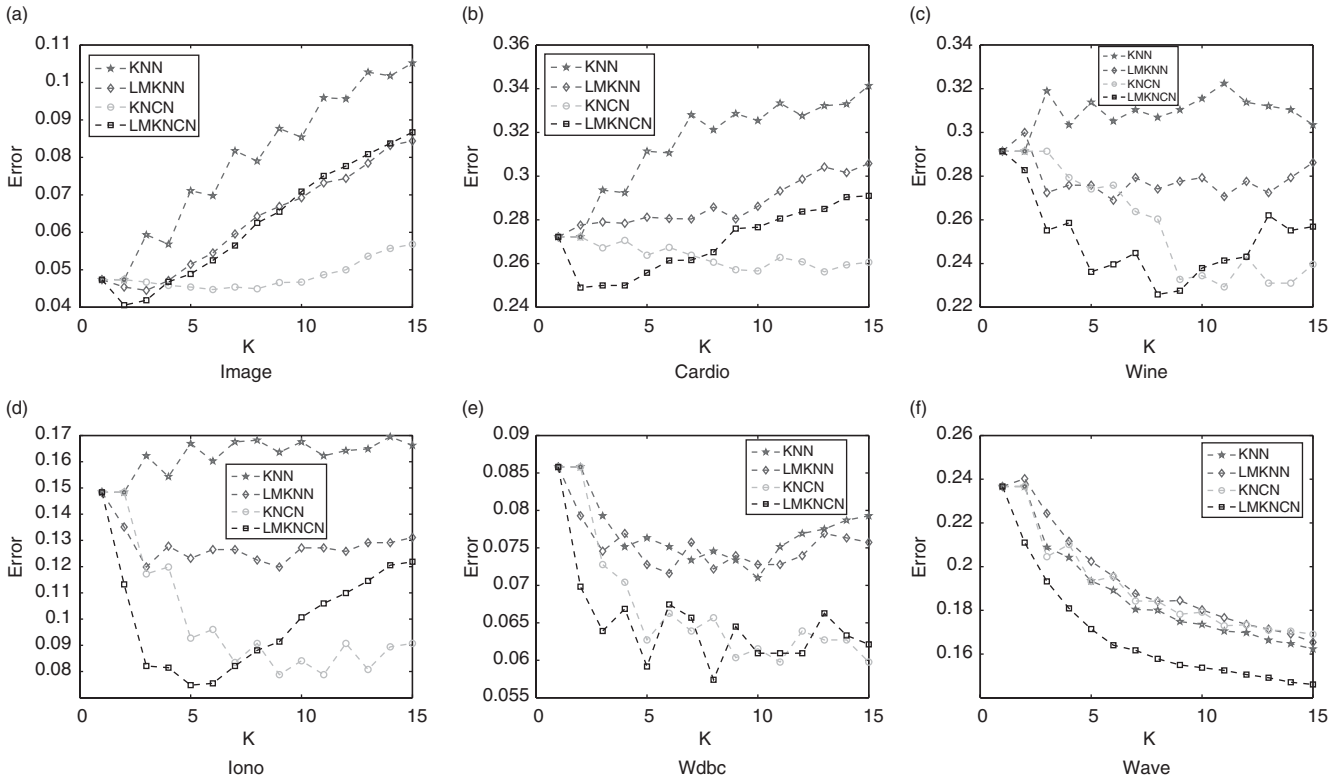

FIGURE 3. The error rates of LMKNCN, LMKNN, KNCN and KNN via the neighborhood size k on the other six real data sets.

TABLE 3. The distances between a query pattern and the local mean vector of each class for values of k, and the classification results on Wine data set (The symbols '+', '◇' and '○' in parentheses denote the labels of the class 1, 2 and 3, respectively. The symbols '×' and '√', respectively, indicate the wrong and right classification. The distances in bold-faces are the smallest distances with a nearest local mean vector for each k value among three classes).

k	The method	Class 1 (+)	Class 2 (◇)	Class 3 (○)	The classification result	
1	LMKNN	25.25	12.88	**7.63**	○	×
	LMKNCN	25.25	12.88	**7.63**	○	×
2	LMKNN	52.64	15.57	**7.12**	○	×
	LMKNCN	52.64	9.60	**7.12**	○	×
3	LMKNN	70.98	16.95	**8.70**	○	×
	LMKNCN	70.98	**3.40**	6.86	◇	√
4	LMKNN	80.23	17.87	**8.94**	○	×
	LMKNCN	80.23	**6.13**	6.92	◇	√
5	LMKNN	89.36	18.72	**8.22**	○	×
	LMKNCN	89.36	**3.89**	7.84	◇	√

final performance is achieved by averaging 10 classification results with 95% confidence. On each data set, we adopt 500 testing samples to verify the classification performance.

First of all, the experiments with respect to the neighborhood size k on the synthetic data sets are conducted in the small training sample size cases. Our aim is to investigate how the neighborhood size k influences the performance of the proposed classifier. Let k varies from 1 to 20 with an interval of 1. The classification results of all the classifiers with varying k are shown in Fig. 4. Note that the hyper-parameters are represented

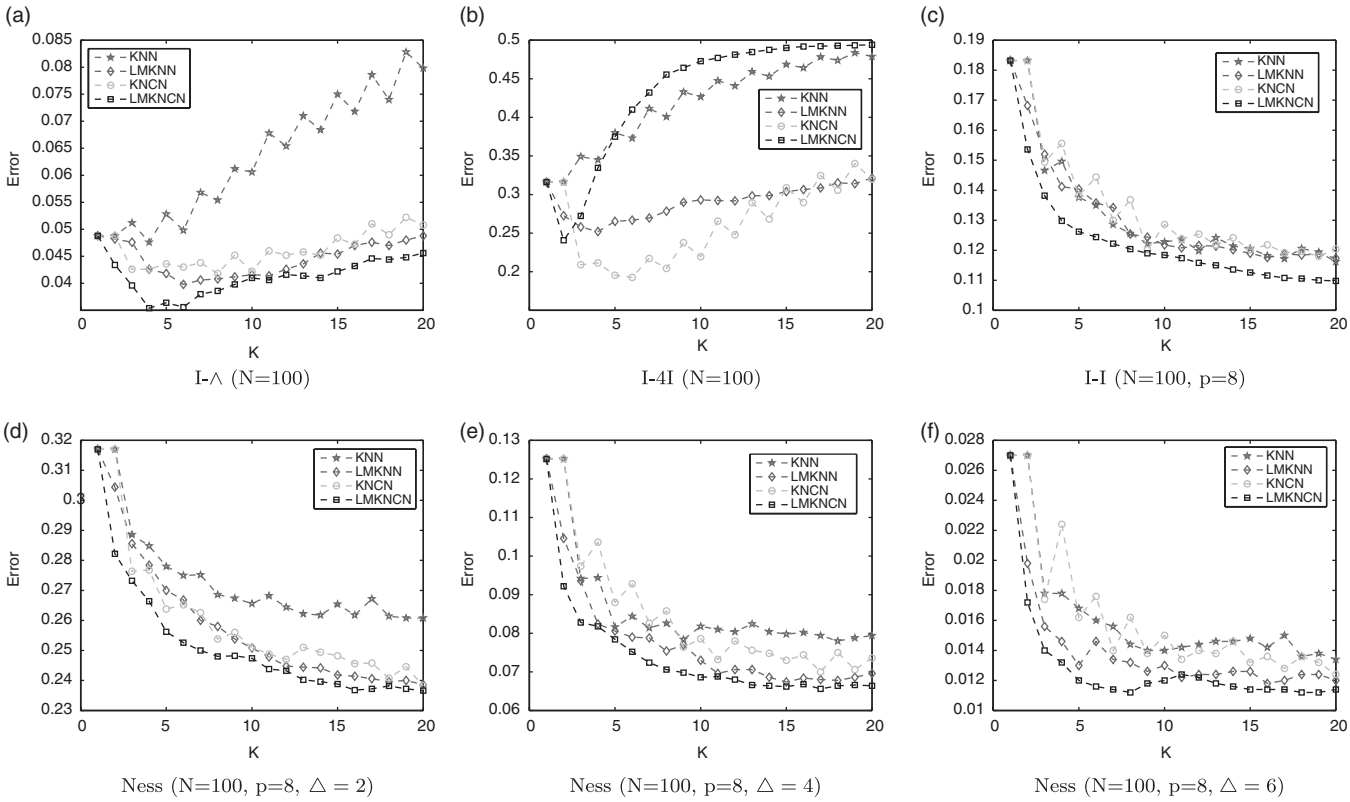

FIGURE 4. The error rates of LMKNCN, LMKNN, KNCN and KNN via the neighborhood size k on each synthetic data set.

in parentheses in each subheading in the figure. As can be seen in Fig. 4, the neighborhood size k strongly influences the classification error, and experimental results imply that the best optimal performance with corresponding k value may exist for the given data sets. Moreover, Fig. 4 shows that the proposed LMKNCN classifier achieves the best classification results on all synthetic data sets except for I-4I over a large range of the neighborhood size k. On I-4I data set, we can see that the performance of LMKNCN is worst among the four classifiers except when $k = 1, 2$. The possible reasons for this inferior performance on I-4I can be that the neighborhood size k in KNN and KNCN is for all training samples of a query pattern while is for training samples of each class in LMKNN and LMKNCN, and the training samples from different classes are seriously overlapped, as discussed in Section 5. As shown in Fig. 4a on I-∧, the classification error rates of LMKNCN, LMKNN and KNCN drop with increase of k at first, and then increase slowly, but those of KNN ascend quickly as values of k increase. It is important to note that the classification performance has a similar pattern on I-I and Ness ($\Delta = 2$, 4 and 6) data sets, and increased values of k lead to better performance. It is clear that the error rates of all methods almost decrease monotonically with increasing k, at first drop steeply and finally approximately keep stable on I-I and Ness. It can also be observed from Figs. 4a and c–f that the classification error rates of LMKNCN are almost lower than KNN, KNCN and LMKNN at each value of k on the I-∧, I-I and Ness data sets. Consequently, experimental results in regard to k demonstrate that LMKNCN is more robust over a large range of k with better classification performance.

Furthermore, the classification behavior of LMKNCN is studied by varying the training sample size N on the I-∧, I-4I, I-I and Ness data sets. Our primary interest is to assess the performance of the proposed classifier in small sample size settings. For all synthetic data sets, their training samples from 10 to 200 with an interval of 10 are generated randomly at each trial. Figure 5 shows the classification performance as a function of N. Note that the selected values of parameters in the experiments are set in parentheses in each subheading in the figure. To do a fair comparison, we fix the identical value of k on all data sets except for I-4I and the same dimensionality p as I-∧ and I-4I. It is worth mentioning that all data sets except for I-4I result in a similar pattern of classification. As can be seen in Fig. 5, the classification error rates of all methods almost drop as the training sample size N increases. In comparing with KNN, LMKNN and KNCN, the proposed LMKNCN classifier almost has best performance with increasing the number of training samples on all data sets. This confirms the robustness of our proposed classifier. On I-4I data set in Fig. 5b, the classification

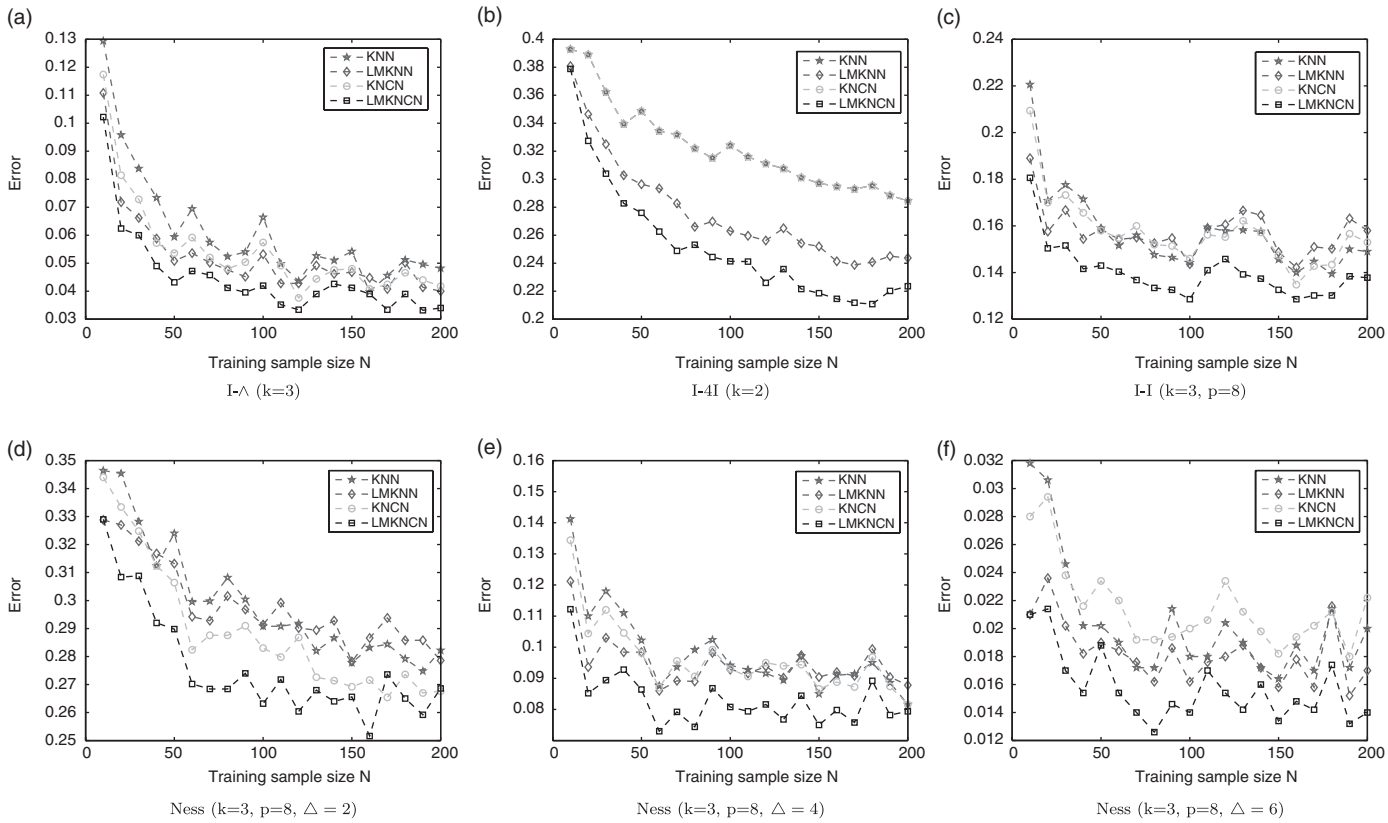

FIGURE 5. The error rates of LMKNCN, LMKNN, KNCN and KNN with varying the training sample size N on each synthetic data set.

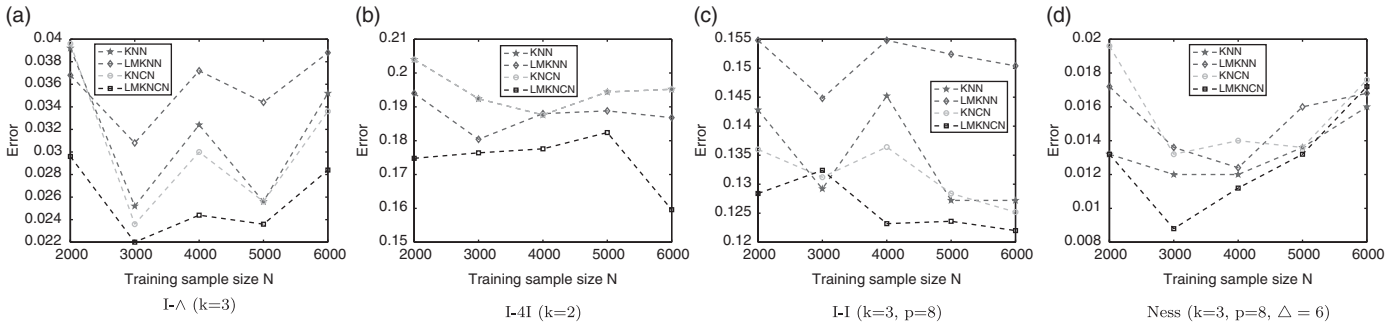

FIGURE 6. The error rates of LMKNCN, LMKNN, KNCN and KNN with varying the large training sample size N on each synthetic data set.

results of LMKNCN are superior to the others only when $k = 2$; however, it obtains unsatisfactory performance like in Fig. 4b. For KNN, KNCN and LMKNN on I-∧, I-I and Ness ($\Delta = 2, 4$ and 6), it is often difficult to clearly distinguish which is better or worse over a large range of training sample size N, but the superiority of the proposed LMKNCN classifier is represented obviously among the four methods. In addition, to verify the scalability of the LMKNCN method to large data, we also conduct experiments on the four data sets with large samples, and generate training samples with number of 2000, 3000, 4000, 5000 and 6000. Note that the classification results are average error rates of five trials with 95% confidence on each model. As shown in Fig. 6, the LMKNCN classifier almost outperforms the other three methods in the large sample size cases. As a consequence, these results in Figs 5 and 6 suggest that the classification performance is improved by increasing the training sample size, and the proposed LMKNCN classifier shows a favorable behavior.

Finally, we investigate the classification performance of the proposed LMKNCN method with varying the number of

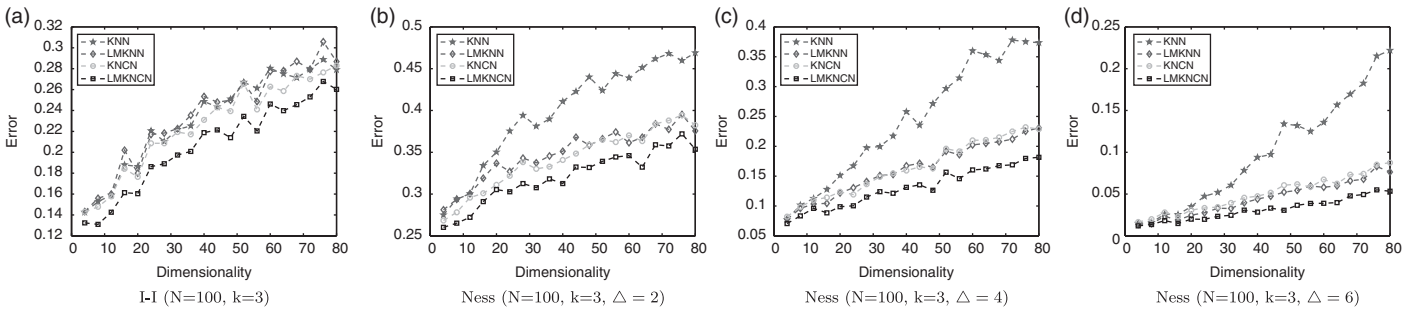

FIGURE 7. The error rates of LMKNCN, LMKNN, KNCN and KNN with increasing the dimensionality on each synthetic data set.

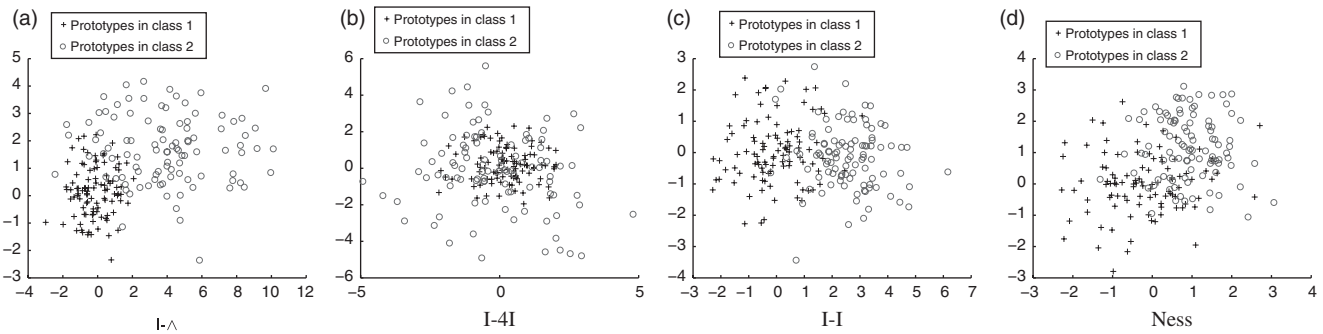

FIGURE 8. The examples of I-∧, I-4I, I-4I and Ness data sets.

dimensionality p in feature space on I-I and Ness ($\Delta = 2, 4$ and 6) data sets, compared with KNN, KNCN and LMKNN. It has been well known that the performance of the nonparametric classifiers suffers from curse dimensionality. Thus, we study the classification behavior of the proposed classifier in high dimensionality. In the experiments, we generate the I-I and Ness data sets with dimensionality from 4 to 80 with an interval of 4. To conduct a fair comparison, we fix the identical training sample size N and neighborhood size k on the data sets, i.e. $N = 100$, $k = 3$, in parentheses in each subheading in Fig. 7. It is interesting to note that the Ness data sets result in a similar pattern of classification with varying the feature space dimension. It is clear that the classification error rates of all classifiers always monotonically increase as the dimensionality p increases, and the better performance of each method is usually obtain at small value of p. As shown in Fig. 7, the classification performance of the proposed LMKNCN is preferable to KNN, KNCN and LMKNN. Therefore, it illustrates from these results that the LMKNCN is less sensitive to dimensionality of the TS.

To sum up, these experiments on the synthetic data sets well demonstrate our that proposed LMKNCN classifier has a satisfactory classification behavior with varying the neighborhood size, the training sample size and feature space dimension, compared with KNN, KNCN and LMKNN. This implies that the combination of the NCN criterion and local mean vector in the proposed LMKNCN classifier has a significant effect on classification error rate in pattern classification, especially in small sample size cases.

5. DISCUSSIONS

In statistical pattern recognition, the classification performance is always influenced by the outliers, especially in the small sample size cases [5]. From the experimental results in Section 4, the proposed LMKNCN method can well overcome the outliers by taking into account of the local mean vector of some nearest centroid neighbors in most cases. To further empirically and simply explain why LMKNCN is robust to outliers, we give the full visualized example spaces of I-∧, I-4I, I-4I and Ness data sets in Fig. 8 to understand the good performance. First, because of eight-dimensional feature spaces of the four data sets used in Section 4, we reduce the dimensionality from 8 to 2 for visualization, using the Fisher criterion in [7]. Given N_j training samples from class c_j in C classes, the Fisher criterion $F(i)$ is defined as

$$F(i) = \frac{\sum_{j=1}^{C-1} \sum_{r=j+1}^{C} P(c_j)P(c_r)(\mu_{ij} - \mu_{ir})^2}{\sum_{j=1}^{C} P(c_j)\sigma_{ij}^2}, \quad (8)$$

where

$$P(c_j) = \frac{N_j}{\sum_{l=1}^{C} N_l}.$$

Note that μ_{ij} and σ_{ij}^2 denote the average and variance on feature i for class c_j, respectively. Then the values of $F(i)$ are computed, the features are ranked in a descent order and two top features are employed.

I-∧, I-4I, I-4I and Ness ($\Delta = 2$), which are randomly generated with 8 dimensionalities and 100 training samples in each class, are visualized in two-dimensional feature spaces in Fig. 8. From Fig. 8a, c and d, although there are many existing outliers for each class, the proposed LMKNCN outperforms the state-of-the-art methods, shown in Figs 4–7. However, in I-4I feature space in Fig. 8b, many samples from two classes are overlapped; so the good performance of LMKNCN is achieved only at a small value of k (i.e. $k = 2$) and it results in inferior performance at a large value of k, shown in Figs 4b and 5b. Hence, we can see that our proposed classifier works well in the case of existing outliers, and the better classification can obtained at a small value of k in training sample situations that samples from different classes are seriously overlapped.

6. CONCLUSIONS

In this article, we propose a local mean-based k-nearest centroid neighbor algorithm that classify one query pattern into the class with nearest local centroid mean vector. This proposed method not only employs the nearness and geometrical distribution of neighbors, but also utilizes the local mean vector of k neighbors from each class. In the proposed LMKNCN classifier, our focus is to mainly address the problems that existed in the KNCN and LMKNN, so as to improve the classification performance. In order to well study the performance of the proposed classifier, our experiments are carried out on the real and synthetic data sets in terms of the classification error, compared with KNN, LMKNN and KNCN. Through the comprehensive comparisons, it suggests that the proposed classifier has the following strengths: (a) it almost obtains the satisfactory classification performance, regardless of the training sample size and feature space dimension. (b) it is more robust to the neighborhood size k with the preferable performance. Consequently, we can draw a sound conclusion that the proposed classifier is a promising algorithm in the field of pattern classification.

ACKNOWLEDGEMENTS

The authors thank the anonymous reviewers for their valuable suggestions.

FUNDING

This work was supported by the National Basic Research Program of China (973 Program) under grant 2011CB302201.

REFERENCES

[1] Technique Report No. 4 (1951) Discriminatory Analysis, Nonparametric Discrimination: Consistency Properties. USAF School of Aviation Medicine, Randolf Field, TX, USA.
[2] Cover, T.M. and Hart, P.E. (1967) Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory*, **13**, 21–27.
[3] Wagner, T. (1971) Covergence of the nearest neighbor rule. *IEEE Trans. Inf. Theory*, **17**, 566–571.
[4] Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statical Learning: Data Mining, Inference, and Prediction*. Springer, New York, NY, USA.
[5] Fukunaga, K. (1990) *Introduction to Statistical Pattern Recognition*. Academic Press, San Diego, CA, USA.
[6] Mitani, Y. and Hamamoto, Y. (2000) Classifier Design Based on the Use of Nearest Neighbor Samples. *Proc. 15th Int. Conf. Pattern Recognition*, Barcelona, Spain, September 3-7, pp. 769–772. IEEE.
[7] Mitani, Y. and Hamamoto, Y. (2006) A local mean-based nonparametric classifier. *Pattern Recogn. Lett.*, **27**, 1151–1159.
[8] Samsudin, N.A. and Bradley, A.P. (2010) Nearest neighbour group-based classification. *Pattern Recogn.*, **43**, 3458–3467.
[9] Yang, J., Zhang, L., Yang, J.Y. and Zhang David (2011) From classifiers to discriminators: a nearest neighbor rule induced discriminant analysis. *Pattern Recogn.*, **44**, 1387–1402.
[10] Chai, J., Liu, H., Chen, B. and Bao, Z. (2010) Large margin nearest local mean classifier. *Signal Process.*, **90**, 236–248.
[11] Zeng, Y., Yang, Y. and Zhao, L. (2009) Pseudo nearest neighbor rule for pattern classification. *Expert Syst. Appl.*, **36**, 3587–3595.
[12] Zeng, Y., Yang, Y. and Zhao, L. (2009) Nonparametric classification based on local mean and class statistics. *Expert Syst. Appl.*, **36**, 8443–8448.
[13] Chaudhuri, B.B. (1996) A new definition of neighbourhood of a point in multi-dimensional space. *Pattern Recogn. Lett.*, **17**, 11–17.
[14] Sánchez, J.S., Pla, F. and Ferri, F.J. (1997) On the use of neighbourhoodbased non-parametric classifiers. *Pattern Recogn. Lett.*, **18**, 1179–1186.
[15] Altinçay, H. (2011) Improving the k-nearest neighbour rule: using geometrical neighbourhoods and manifold-based metrics. *Expert Syst.*, **28**, 391–406.
[16] Sánchez, J.S., Pla, F. and Ferri, F.J. (1998) Improving the k-NCN classification rule through heuristic modifications. *Pattern Recogn. Lett.*, **19**, 1165–1170.
[17] Sánchez, J.S., Pla, F. and Ferri, F.J. (1997) Prototype selection for the nearest neighbour rule through proximity graphs. *Pattern Recogn. Lett.*, **18**, 507–513.
[18] Sánchez, J.S., Barandela, R. and Marqués, A.I. (2003) Analysis of new techniques to obtain quality training sets. *Pattern Recogn. Lett.*, **24**, 1015–1022.
[19] Lozano, M., Sotoca, J.M., Sánchez, J.S., Pla, F., Pękalska, E. and Duin, R.P.W. (2006) Experimental study on prototype

optimisation algorithms for prototype-based classification in vector spaces. *Pattern Recogn.*, **39**, 1827–1838.

[20] Sánchez, J.S. and Marqués, A.I. (2006) An LVQ-based adaptive algorithm for learning from very small codebooks. *Neurocomputing Lett.*, **69**, 922–927.

[21] Sánchez, J.S. and Marqués, A.I. (2002) Enhanced Neighbourhood Specifications for Pattern Classification. In Chen, D. and Cheng, X. (eds), *Pattern Recognition and String Matching*. Kluwer Academic Publishers.

[22] Grabowski, Sz. (2004) Limiting the Set of Neighbors for the K-NCN Decision Rule: Greater Speed with Preserved Classification Accuracy. *Proceedings of the International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science*, Lviv Slavsko, Ukraine, February 24–28, pp. 511–514. IEEE.

[23] Frank, A. and Asuncion, A. (2010) *UCI machine learning repository*. http://archive.ics.uci.edu/ml. University of California, School of Information and Computer Science, Irvine, CA.

[24] Van Ness, J. (1980) On the dominance of non-parametric Bayes rule discriminant algorithms in high dimensions. *Pattern Recogn.*, **12**, 355–368.