Pseudo Nearest Centroid Neighbor Classification

Hongxing Ma, Xili Wang and Jianping Gou

Abstract In this paper, we propose a new reliable classification approach, called the pseudo nearest centroid neighbor rule, which is based on the pseudo nearest neighbor rule (PNN) and nearest centroid neighborhood (NCN). In the proposed PNCN, the nearest centroid neighbors rather than nearest neighbors per class are first searched by means of NCN. Then, we calculate k categorical local mean vectors corresponding to k nearest centroid neighbors, and assign the weight to each local mean vector. Using the weighted k local mean vectors for each class, PNCN designs the corresponding pseudo nearest centroid neighbor and decides the class label of the query pattern according to the closest pseudo nearest centroid neighbor among all classes. The classification performance of the proposed PNCN is evaluated on real data sets in terms of the classification accuracy. The experimental results demonstrate the effectiveness of PNCN over the competing methods in many practical classification problems.

Keywords K-nearest neighbor rule · Nearest centroid neighborhood · Local mean vector · Pattern classification

---

H. Ma (✉) · X. Wang
College of Computer Science, Shaanxi Normal University, Xi'an, Shaanxi 710062, People's Republic of China

H. Ma
College of Electrical and Information Engineering, Beifang University of Nationalities, Yinchuan, Ningxia 750021, People's Republic of China

J. Gou
School of Computer Science and Telecommunication Engineering, JiangSu University, ZhenJiang, JiangSu 212013, People's Republic of China

J.C. Hung et al. (eds.), *Frontier Computing*, Lecture Notes in Electrical Engineering 375, DOI 10.1007/978-981-10-0539-8_12

1 Introduction

In pattern recognition, *k*-nearest neighbor rule (KNN) is one of the most widely used nonparametric approaches, and is also deemed to be one of the top 10 algorithms in data mining [1], due to its simplicity and effectiveness for classification. It has been theoretically proven that the KNN classifier has asymptotically optimal performance in the Bayes sense [2, 3]. Moreover, the appeal of KNN is that only a single integer parameter *k* is required to adjust and any particular statistical distribution of the training data should not be considered [4]. However, the classification performance of KNN and its variants are still degraded by three main issues: the sparse problem, the imbalance problem and the noise problem [5]. The sparse problem is that there are a small number of training samples in many practical classification tasks. In the small training sample size cases, the nonparametric KNN-based classifiers usually suffer from the existing outliers [6]. The imbalance problem is produced when the data in one class heavily outnumbers the data in another class. In this case, the class boundary can be skewed to the class with few samples. The noise problem is the sensitivity to the outliers or noises that exists in KNN and its variants, as they treat both noisy and normal points equally.

In fact, the choice of the neighborhood size *k* can aggravate the negative influence of the three problems aforementioned on the classification performance of KNN-based methods to some degree [1, 7]. If *k* is too small, the results can be sensitive to the data sparseness and the noisy points. On the other hand, if *k* is too large, then the results can be degraded by the introduction of many outliers from other classes in the neighborhood. Thus, the performance of nonparametric KNN-based classifiers can be severely affected by the existing outliers, particularly in the cases of the sparse, imbalance and noise problems.

To overcome the existing outliers, a reliable KNN-based approaches, called the local mean-based KNN rule (LMKNN), is well designed in [8]. It uses the local mean vector of the categorical *k* nearest neighbors to determine the classes of query patterns. Subsequently, the basic idea of LMKNN has successfully been applied to some approaches, such as the pseudo nearest neighbor rule (PNN) [9], the local mean-based *k*-nearest centroid neighbor rule (LMKNCN) [10] and other methods [11–13]. As an extension of LMKNN, PNN is also robust to the outliers. It utilizes the distance weighted local learning in each class to design the pseudo nearest neighbor of the query pattern, based on the distance weighted *k*-nearest neighbor rule [14] and LMKNN. Then, the query pattern is allocated into the class, which the closest pseudo nearest neighbor belongs to. As we know, *k*-nearest centroid neighbor rule (KNCN), based on NCN [15], is very effective classifier, especially in the small sample size situations [16, 17]. Combined the robustness of LMKNN and effectiveness of KNCN, LMKNCN is introduced in [10]. It employs the local mean vector of *k* nearest centroid neighbor from each class to decide the class of the query pattern.

To further perform classification on the existing outliers well, especially in the sparse, imbalance and noise situations, we propose the pseudo nearest centroid neighbor rule (PNCN), motivated by PNN and NCN. In this method, we design the pseudo nearest centroid neighbor rather than the pseudo nearest neighbor in each class to classify the query pattern. First, k nearest centroid neighbors of one query pattern are found from each class, and then k local mean vectors corresponding to k neighbors are calculated. Second, the weights for the local mean vectors instead of the neighbors are assigned. Third, the pseudo nearest centroid neighbor in each class are decided by using the weighted sum of distances of k local mean vectors. Finally, the query pattern is classified into the class, which the closest pseudo nearest centroid neighbor belongs to. The classification performance of the proposed PNCN is investigated on real data sets, compared to KNN, LMKNN, KNCN, LMKNCN and PNN. Experimental results suggest that PNCN is effective and robust in such practical situations.

2 Pseudo Nearest Centroid Neighbor Classification

In pattern classification, the recognition rates of the KNN-based nonparametric classifiers are easily affected by the outliers in the issues above. Considering the superiorities of both PNN and NCN, we give a new scheme of designing pseudo nearest neighbor and accordingly propose the pseudo nearest centroid neighbor rule (PNCN), in order to improve the classification accuracy rate. In what follows, for the ease of presentation in the general recognition problem, we first suppose that there is a training set $T = \{x_i \in \mathbb{R}^d\}_{i=1}^N$ with M classes in d-dimensional feature space, and the corresponding class labels are $\{y_1, y_2, \ldots y_N\}$, where $y_i \in \{c_1, c_2, \ldots, c_M\}$, a class subset of T from the class c_l is $T_l = \{x_{lj} \in \mathbb{R}^d\}_{j=1}^{N_l}$ with the number of the training samples N_l.

2.1 Nearest Centroid Neighborhood

As we know, the choice of neighborhood plays a critical role in the KNN-based classification [1]. It has been found that Nearest centroid neighborhood (NCN) is a very good alternative to nearest neighborhood [15]. The concept of NCN focuses on the idea that the neighborhood of a query pattern is simultaneously subject to the distance criterion and the symmetry criterion. On the one hand, the neighbors of a query are as close to it as possible by the distance criterion. On the other hand, the neighbors of a query are placed as homogeneously around it as possible with the symmetry criterion. To seek the neighbors according to NCN, the centroid of a set $Z = \{z_1, z_2, \ldots, z_n\}$ should be first defined as

$$\overline{Z} = \frac{1}{n}\sum_{i=1}^{n} z_i. \tag{1}$$

Then, the nearest centroid neighbors of a given query pattern x with both criterions above, are searched through an iterative procedure [15] as follows:

1. Find the first nearest centroid neighbor x_1^{NCN} of x that corresponds to its nearest neighbor.
2. Find the i-th nearest centroid neighbor x_i^{NCN} $(i \geq 2)$, which is imposed by the constraint that the centroid of the query x and all previous centroid neighbors, i.e., $x_1^{NCN}, \ldots, x_{i-1}^{NCN}$, is the closest to x.

Based on the NCN, KNCN and LMKNCN are introduced in the field of pattern classification [10, 16].

2.2 The Proposed PNCN Classifier

Based on NCN and PNN, we introduce the pseudo nearest centroid neighbor rule (PNCN). It first finds the k nearest centroid neighbors per class in terms of the NCN, and then computes each local mean vector of first j categorical neighbors and allocates the weight for each local mean vector. Finally, the pseudo nearest centroid neighbor per class is designed by using the weighted k local mean vectors corresponding to k nearest centroid neighbors. In the process of making classification decision, PNCN assigns the class label, which the closest pseudo nearest centroid neighbor belongs to among all classes, into the unseen pattern.

Given a query pattern x in the pattern classification problem, the PNCN decides the class label of x as follows:

1. Search k nearest centroid neighbors from T_l of each class c_l for the query pattern x in the training set T, say $T_{lk}^{NCN}(x) = \{x_{lj}^{NCN} \in \mathbb{R}^d\}_{j=1}^{k}$.
2. Compute the local mean vector $\overline{u}_{lj}^{NCN}(x)$ of the first j nearest centroid neighbors of a query x from class c_l. Let $\overline{U}_{lk}^{NCN}(x) = \{\overline{u}_{lj}^{NCN}(x) \in \mathbb{R}^d\}_{j=1}^{k}$ denote the set of the k local mean vectors corresponding to k nearest centroid neighbors in the class c_l, and $d(x, \overline{u}_{l1}^{NCN}(x)), d(x, \overline{u}_{l2}^{NCN}(x)), \cdots, d(x, \overline{u}_{lk}^{NCN}(x))$ are their corresponding Euclidean distances to x.

$$\overline{u}_{lj}^{NCN}(x) = \frac{1}{j}\sum_{m=1}^{j} x_{lm}^{NCN}. \tag{2}$$

It should be noted that the local mean vector $\overline{u}_{l1}^{NCN}(x)$ of the first nearest centroid neighbor x_{l1}^{NCN} is the same as the first nearest neighbor.

3. Assign different weights to k categorical local mean vectors in the same way as the PNN, and the weight $\overline{W}_{lj}^{NCN}$ of the j-th local mean vector $\overline{u}_{lj}^{NCN}(x)$ for the class c_l is determined as:

$$\overline{W}_{lj}^{NCN} = \frac{1}{j} \quad j = 1, \ldots, k. \tag{3}$$

4. Design the pseudo nearest centroid neighbor $\overline{x}_l^{PNCN}(x)$ of the query point x from class c_l, and c_l can be viewed as the class label of $\overline{x}_l^{PNCN}(x)$. The distance $\overline{d}(x, \overline{x}_l^{PNCN}(x))$ between x and $\overline{x}_l^{PNCN}(x)$ can be defined by the weighted sum of distances of k categorical local mean vectors to x as follows:

$$\begin{aligned}\overline{d}\left(x, \overline{x}_l^{PNCN}(x)\right) = \Big(&\overline{W}_{l1}^{NCN} \times d\left(x, \overline{u}_{l1}^{NCN}(x)\right) + \overline{W}_{l2}^{NCN} \times d\left(x, \overline{u}_{l2}^{NCN}(x)\right) \\ &+ \ldots + \overline{W}_{lk}^{NCN} \times d\left(\left(x, \overline{u}_{lk}^{NCN}(x)\right)\right)\Big).\end{aligned} \tag{4}$$

5. Classify the query point x into the class c, which the closest pseudo nearest centroid neighbor belongs to in the light of Eq. (4) among all classes.

$$c = \arg\min_{c_l} \overline{d}\left(x, \overline{x}_l^{PNCN}(x)\right). \tag{5}$$

Note that the proposed PNCN is equivalent to the 1NN, LMKNN, PNN, KNCN and LMKNCN rules only when $k = 1$, and the value of k is no more than N_l.

2.3 The PNCN Algorithm

According to the procedure of the PNCN above, we summarize it in Algorithm 1 by means of the pseudo codes.

Algorithm 1: The pseudo nearest centroid neighbor algorithm
Require:

x: a query pattern, k: number of nearest neighbors, $T = \{x_i \in \mathbb{R}^d\}_{i=1}^{N}$: a training set. $T_l = \{x_{lj} \in \mathbb{R}^d\}_{j=1}^{N_l}$: a training subset from class $c_l, c_1, \ldots, c_M$: M class labels.
M: the number of classes in T, $N_1, \ldots, N_M$: number of training samples for M classes.

Ensure:

Predict the class label of the query pattern x by the closet pseudo nearest centroid neighbor among all classes.

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

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

$$d(x, x_{lj}) = \sqrt{\left(x - x_{lj}\right)^T \left(x - x_{lj}\right)}$$

end for

Step 2: Search the k nearest centroid neighbors of x in each class c_l, say $T_{lk}^{NCN}(x) = \{x_{lj}^{NCN} \in \mathbb{R}^d\}_{j=1}^{k}$.

(i) Find the first nearest centroid neighbor of x in each class c_l, say x_{l1}^{NCN}.

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

set $x_{l1}^{NCN} = x_{min\_index}$, $R_l^{NCN}(x) = \{x_{l1}^{NCN} \in \mathbb{R}^d\}$

(ii) Find k nearest centroid neighbors of x except x_{l1}^{NCN} in each class c_l.

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

Set $S_l(x) = T_l - R_l^{NCN}(x) = \{x_{ln} \in \mathbb{R}^d\}_{n=1}^{L_l(x)}$, $L_l(x) = length(S_l(x))$
Calculate the sum of the previous $j - 1$ nearest centroid neighbors.

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

Compute the centroids in the set S_l for x.

for $n = 1$ to $L_l(x)$ **do**

$$\overline{x}_{ln} = \frac{1}{j}\left(x_{ln} + sum_l^{NCN}(x)\right), \overline{d}_{ln}(x, \overline{x}_{ln}) = \sqrt{(x - \overline{x}_{ln})^T (x - \overline{x}_{ln})}$$

end for

Find the j-th nearest centroid neighbor.

$$[min\_index^{NCN}, min\_dist^{NCN}] = min(\overline{d}_{ln}(x, \overline{x}_{ln}))$$

Set $x_{lj}^{NCN} = x_{min\_index^{NCN}}$, and add x_{lj}^{NCN} to the set $R_l^{NCN}(x)$.

end for

Set $T_{lk}^{NCN}(x) = R_l^{NCN}(x)$.

Step 3: Compute the local mean vector $\overline{u}_{lj}^{NCN}(x)$ of the first j nearest neighbors of x using $T_{lk}^{NCN}(x)$ and the corresponding distance $d(x, \overline{u}_{lj}^{NCN}(x))$ between $\overline{u}_{lj}^{NCN}(x)$ and x.

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

$$\overline{u}_{lj}^{NCN}(x) = \frac{1}{j}\sum_{m=1}^{j} x_{lm}^{NCN}, d\left(x, \overline{u}_{lj}^{NCN}(x)\right) = \sqrt{\left(x - \overline{u}_{lj}^{NCN}(x)\right)^{T}\left(x - \overline{u}_{lj}^{NCN}(x)\right)}$$

end for

Set $\overline{U}_{lk}^{NCN}(x) = \{\overline{u}_{lj}^{NCN}(x) \in \mathbb{R}^{d}\}_{j=1}^{k}$, $\overline{D}_{lk}^{NCN}(x) = \{d(x, \overline{u}_{l1}^{NCN}(x)), \ldots, d(x, \overline{u}_{lk}^{NCN}(x))\}$.

Step 4: Allocate the weights $\overline{W}_{lj}^{NCN}$ to the j-th the local mean vector $\overline{u}_{lj}^{NCN}(x)$ in the set $\overline{U}_{lk}^{NCN}(x)$.

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

$$\overline{W}_{lj}^{NCN} = \frac{1}{j} \quad j = 1, \ldots, k.$$

end for

Set $\overline{W}_{lk} = \{\overline{W}_{l1}^{NCN}, \ldots, \overline{W}_{lk}^{NCN}\}$.

Step 5: Design pseudo nearest centroid neighbor $\overline{x}_{l}^{PNCN}(x)$ using $\overline{W}_{lk}$ and $\overline{D}_{lk}^{NCN}(x)$.

$$\begin{aligned}\overline{d}\left(x, \overline{x}_{l}^{PNCN}(x)\right) = {} & \overline{W}_{l1}^{NCN} \times d\left(x, \overline{u}_{l1}^{NCN}(x)\right) + \overline{W}_{l2}^{NCN} \times d\left(x, \overline{u}_{l2}^{NCN}(x)\right) \\ & + \ldots + \overline{W}_{lk}^{NCN} \times d\left(x, \overline{u}_{lk}^{NCN}(x)\right)\end{aligned}$$

Step 6: Assign the class c of the closest pseudo nearest centroid neighbor to x.

$$c = \arg\min_{c_l} \overline{d}\left(x, \overline{x}_{l}^{PNCN}(x)\right)$$

3 Experiments

In this section, we conduct the experiments to validate the classification performance of the proposed PNCN on the benchmark real data sets. The PNCN is compared with KNN, LMKNN, PNN, KNCN and LMKNCN in terms of the classification accuracy rate, which is takes as one of the effective measures in pattern recognition [8, 10]. In what follows. we should note that the neighborhood size k is for all training samples of a query pattern in KNN and KNCN, while is for training samples of each class in LMKNN, LMKNCN, PNN and PNCN.

3.1 Data Sets

In the experiments, twelve real data sets taken from the UCI Repository [18] are employed. The information of these UCI data sets including the numbers of samples, attributes, classes, training and testing samples is displayed in Table 1. For short, among these data sets, the abbreviated names for 'Parkinsons', 'Transfusion', 'Libras Movement', 'Cardiotocography', 'LandsatSatellite', 'Page-blocks', 'Image Segmentation' and 'Robot Navigation' are 'Park', 'Trans', 'Libras', 'Cardio', 'Landsat', 'Page', 'Image' and 'Robot', respectively. Note that the Glass data set originally have seven classes, but in our experiments the five classes with very few samples are deleted.

3.2 Experimental Results

In this subsection, to well demonstrate the classification performance of the proposed PNCN, we do the experiments on the real UCI data sets. One of the advantages of using the real data sets is that they are generated without any knowledge of the classification procedures that it will be used to test. The second

Table 1 The real UCI data sets used in the experiments

Data	Size	Attributes	Classes	Testing samples	Training samples
Sonar	208	60	2	132	76
Park	195	22	2	65	130
Seed	210	7	3	105	105
Wine	178	13	3	59	119
Glass	146	9	2	53	93
Trans	748	4	2	248	500
Libras	360	90	15	90	270
Landsat	6435	36	6	2146	4289
Cardio	2126	21	10	710	1416
Image	2310	19	7	1078	1232
Robot	5456	4	4	1818	3638
Page	5473	10	5	1830	3643

advantage is that the sparse, imbalance, noise problems are usually produced in practical classification, and the outliers for one test sample in these selected real data sets always exist. Since the classification performance of each method is verified by using the validation test, each whole data set is randomly split into a training set and test set. To assess the quality of each method, we perform 10 times on each data set, and the average classification accuracy with 95 % confidence over test sets is viewed as the final performance of each method. For each whole data set, the training and test samples are randomly generated, shown in the Table 1. In the experiments, the parameter of the neighborhood size *k* takes the value from 1 to 15 with step 1.

To validate the proposed PNCN method on the performance, we first explore the classification accuracy rates of the competing classifiers with varying the neighborhood size *k* on each real data set. As there is no general way to determine the optimal parameter *k* in KNN-based methods, it is expected that our PNCN can be more robust to the change of *k* when the parameter is common for all compared methods. The experimental comparisons of all the classifiers in terms of the classification accuracy is illustrated in Figs. 1 and 2. We can obviously observe that the proposed PNCN almost surpasses the other methods among the preseted range of the neighborhood size *k* on each data set. Compared to KNN, LMKNN, PNN, KNCN and LMKNCN, the classification performance of PNCN at first increases when the values of *k* is small, and then grows slowly or keeps almost stable as *k* increases on all the data sets except the Seed data set. Moreover, the best performance of the PNCN is usually yielded at the larger value of the neighborhood size, this fact implies that it can use more nearest neighbors to improve the classification. However, KNN, LMKNN, PNN, KNCN and LMKNCN vary drastically

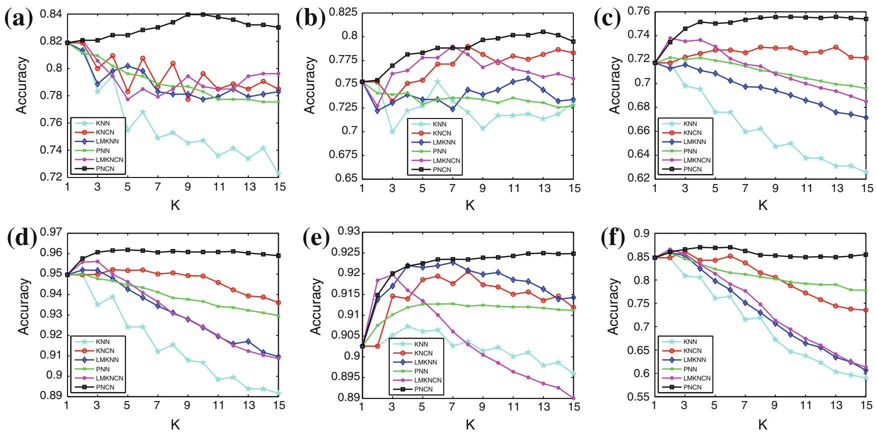

Fig. 1 The accuracy rates of each method via *k* on each real data set

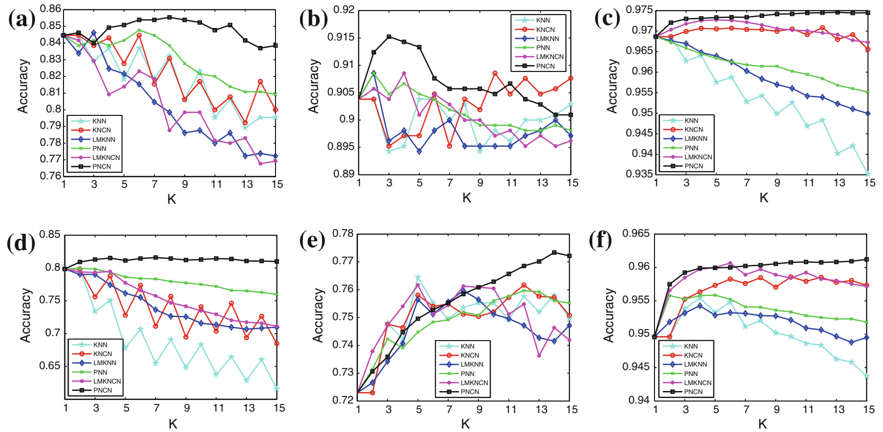

Fig. 2 The accuracy rates of each method via *k* on each real data set

against the parameter *k*. It can also be seen that the differences of the classification accuracy rates between PNCN and the other methods are very significant at a larger *k*. Consequently, we can draw a conclusion that the proposed PNCN is robust to the choice of *k* with satisfactory performance. This means that the selection of *k* for PNCN is easier than that for the other KNN-based classifier.

The empirical comparisons of all the competing classifiers are investigated by the maximal accuracy rates (%) of each method with the corresponding standard deviations (stds) and values of parameter *k* in the parentheses on each real data sets. The classification results of each method on all data sets are given in Table 2. It should be noted that the best classification performance among these methods are indicated in bold-face on each data set. When we look at the best cases in Table 2, the proposed PNCN is found to be very superior to the other methods in most cases. More interestingly, it can be observed that the best classification results of PNCN are nearly yielded at larger values of *k* on all data sets, compared to the other five methods, shown in Table 2, Figs. 1 and 2. In our experiments, there are a finite number of training samples and the training samples are randomly chosen from each whole real data set, so the value of *k* can easily affect the classification performance. Nevertheless, the experiments show that the proposed PNCN can use more nearest neighbors to capture enough information, so as to improve the classification performance.

Table 2 The maximal accuracy rates (%) of each method with the corresponding standard deviations (stds) and values of k in the parentheses on each UCI data sets

Data	KNN	KNCN	LMKNN	LMKNCN	PNN	PNCN
Sonar	79.85 ± 3.30	79.85 ± 3.30	79.85 ± 3.30	79.85 ± 3.30	79.92 ± 3.63	81.59 ± 3.69
	(1)	(1)	(1)	(1)	(2)	(7)
Park	84.46 ± 3.28	84.46 ± 3.28	84.62 ± 4.23	84.46 ± 3.28	84.77 ± 4.00	85.54 ± 3.92
	(1)	(1)	(3)	(1)	(6)	(8)
Seed	90.38 ± 2.03	90.86 ± 1.81	90.86 ± 2.92	90.86 ± 2.02	90.86 ± 2.30	91.52 ± 2.44
	(1)	(10)	(2)	(4)	(2)	(3)
Wine	75.25 ± 4.67	78.98 ± 3.50	75.59 ± 5.19	78.98 ± 4.74	75.25 ± 4.67	80.51 ± 3.12
	(1)	(8)	(12)	(7)	(1)	(13)
Glass	81.89 ± 5.85	81.89 ± 5.85	81.89 ± 5.85	82.08 ± 3.90	81.89 ± 5.85	83.96 ± 3.59
	(1)	(1)	(1)	(2)	(1)	(9)
Trans	76.45 ± 1.49	76.17 ± 1.48	75.97 ± 1.71	76.17 ± 1.89	75.97 ± 1.28	77.34 ± 1.33
	(5)	(12)	(8)	(5)	(12)	(14)
Libras	84.78 ± 3.78	86.11 ± 3.02	85.89 ± 3.78	86.56 ± 3.83	85.78 ± 4.12	87.00 ± 3.27
	(1)	(3)	(2)	(2)	(2)	(4)
Landsat	90.73 ± 0.39	92.05 ± 0.59	92.27 ± 0.71	91.98 ± 0.41	91.28 ± 0.41	92.50 ± 0.46
	(4)	(8)	(7)	(3)	(7)	(13)
Cardio	71.73 ± 1.01	73.03 ± 1.15	71.73 ± 1.01	73.80 ± 1.36	72.17 ± 1.38	75.58 ± 1.36
	(1)	(8)	(1)	(2)	(2)	(13)
Image	94.97 ± 0.55	95.22 ± 0.71	95.20 ± 0.54	95.62 ± 0.68	95.01 ± 0.56	96.19 ± 0.57
	(1)	(4)	(2)	(3)	(2)	(5)
Sensor	96.86 ± 0.38	97.08 ± 0.42	96.86 ± 0.38	97.27 ± 0.33	96.86 ± 0.38	97.46 ± 0.38
	(1)	(12)	(1)	(5)	(1)	(13)
Page	95.54 ± 0.50	95.86 ± 0.34	95.43 ± 0.44	96.07 ± 0.52	95.58 ± 0.45	96.12 ± 0.32
	(4)	(10)	(4)	(6)	(5)	(15)

4 Conclusions

In this paper, we propose a new classifier, called the pseudo nearest centroid neighbor rule, with aim of further improving the classification performance. It is motivated by the PNN and NCN. In the new method, we find k nearest centroid neighbors based on the NCN and calculate the k categorical local mean vectors corresponding to the k nearest centroid neighbors. The proposed PNCN designs the pseudo nearest centroid neighbor for each class by using the weighted k local mean vectors, and assigns the class of the closest pseudo nearest centroid neighbor to the query pattern. To investigate the performance of PNCN, we conduct the experiments on real data sets. The experimental results suggest that the proposed PNCN method are promising classifier.

Acknowledgment This work was supported by National Science Foundation of China (Grant Nos. 61162005, 41171338 and 61163002), the Beifang Ethnic University school project (Grant No. 2010Y030), the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant No. 14KJB520007), China Postdoctoral Science Foundation (Grant No. 2015M570411) and Research Foundation for Talented Scholars of JiangSu University (Grant No. 14JDG037).

References

1. Wu X, Kumar V, Quinlan JR, Ghosh J (2008) Top 10 algorithems in data mining. Knowl Inf Syst 14(1):1–37
2. Cover TM, Hart PE (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13 (1):21–27
3. Wagner T (1971) Covergence of the nearest neighbor rule. IEEE Trans Inf Theory 17:566–571
4. Blanzieri E, Melgan F (2008) Nearest neighbor classification of remote sensing images with the maximal margin principle. IEEE Trans Geosci Remote Sens 46(6):1804–1811
5. Guo G, Dyer CR (2005) Learning from examples in the small sample case: face expression recognition. IEEE Trans Syst Man Cybern 35:477–488
6. Fukunaga K (1990) Introduction to statistical pattern recognition. Academic Press, San Diego, CA, USA, pp 219–238
7. Gou J, Du L, Zhang Y, Xiong T (2012) A new distance-weighted k-nearest neighbor classifier. J Inf Comput Sci 9(6):1429–1436
8. Mitani Y, Hamamoto Y (2006) A local mean-based nonparametric Classifier. Pattern Recognition Lett 27(10,15):1151–1159
9. Zeng Y, Yang Y, Zhao L (2009) Pseudo nearest neighbor rule for pattern classification. Expert Syst Appl 36(2):3587–3595
10. Gou J, Du Zhang Yi L, Xiong T (2012) A local mean-based K-nearest centroid neighbor classifier. Comput J 55(9):1058–1071
11. Yang J, Zhang L, Yang JY, David Zhang (2011) From classifiers to discriminators: a nearest neighbor rule induced discriminant analysis. Pattern Recogn 44(7):1387–1402
12. Gou J, Zhan Y, Rao Y, Shen X, Wang X, He W (2014) Improved pseudo nearest neighbor classification. Knowl-Based Syst 70:361–375
13. Yang T, Kecman V (2008) Adaptive local hyperplane classification. Neurocomputing 71(13–15):3001–3004

14. Dudani SA (1976) The distance-weighted k-nearest neighbor Rule. IEEE Trans Syst Man Cybern 6(4):325–327
15. Chaudhuri BB (1996) A new definition of neighbourhood of a point in multi-dimensional space. Pattern Recogn Lett 17(1):11–17
16. Sánchez JS, Pla F, Ferri FJ (1997) On the use of neighbourhoodbased non-parametric classifiers. Pattern Recogn Lett 18(11–13):1179–1186
17. Sánchez JS, Marqués AI (2006) An LVQ-based adaptive algorithm for learning from very small codebooks. Neurocomputing 69(7–9):922–927
18. Bache K, Lichman M (2015) UCI Machine Learning Repository. http://archive.ics.uci.edu/ml. University of California, School of Information and Computer Science, Irvine, CA