Journal of King Saud University – Computer and Information Sciences 34 (2022) 6292–6302

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

A multi-voter multi-commission nearest neighbor classifier☆

Suyanto Suyanto [a,*], Prasti Eko Yunanto [a], Tenia Wahyuningrum [b], Siti Khomsah [b]

[a] School of Computing, Telkom University, Bandung, Indonesia
[b] Faculty of Informatics, Institut Teknologi Telkom Purwokerto, Indonesia

ARTICLE INFO

Article history:
Received 26 May 2021
Revised 22 December 2021
Accepted 29 January 2022
Available online 23 February 2022

Keywords:
Nearest neighbor classifier
c-Means clustering
Machine learning
Multi-commission
Multi-voter

ABSTRACT

Many improved versions of *k*-nearest neighbor (KNN) have been proposed by minimizing total distances of multi *k* nearest neighbors (multi-voter) in each class instead of the majority voting, such as a local mean-based pseudo nearest neighbor (LMPNN) that give a better decision. In this paper, a new KNN variant called multi-voter multi-commission nearest neighbor (MVMCNN) is proposed to examine its benefits in enhancing the LMPNN. As the name suggests, MVMCNN uses some commissions: each calculates the total distance between the given query point (test pattern) and *k* pseudo nearest neighbors using the LMPNN scheme. The decision class is defined by minimizing those total distances. Hence, the decision in MVMCNN is obtained more locally than LMPNN. Examination based on 10-fold cross-validation shows that the proposed multi-commission scheme can enhance the original (single-commission) LMPNN. Compared with two single-voter models: KNN and Bonferroni Mean Fuzzy *k*-Nearest Neighbors (BM-FKNN), the proposed MVMCNN also gives lower mean error rates as well as higher Precision, Recall, and F1 Score, indicating that the multi-voter model provides a better decision than the single-voter ones.

1. Introduction

In data mining, *k*-nearest neighbor (KNN) is one of the top 10 algorithms (Wu et al., 2008) in terms of popularity, where it is widely used in various applications due to its simplicity to implement. It has been commonly used in research and application in the big data area since it is efficient for classification, regression, and clustering tasks. In fatal and sensitive applications, such as medicine, military, law, and transportation, KNN is favorable due to its explainability. Recently, in Papernot and Mcdaniel (2018) KNN is combined with deep neural networks (DNNs) to create a confident, interpretable, and robust deep learning.

However, KNN has four problematic issues that have been found in the previous literature (Zhang and Member, 2019). Firstly, KNN is generally sensitive to the neighborhood size *k*. The performance of KNN can be made worse by the outliers if *k* is smaller or larger (Gou et al., 2019). It is pretty challenging to set a proper *k* for a given training set since the training samples (instances) usually have different distributions. This problem can be solved by setting various *k* values to different sample subspaces (Tan et al., 2020) or to different test samples (Bulut and Amasyali, 2017, Zhang et al., 2017, Zhong et al., 2017, Zhang et al., 2017, Pan et al., 2017, Zhang et al., 2018, Tan et al., 2020). Nevertheless, both solutions increase the complexity of KNN.

Secondly, KNN is also sensitive to the distance function used to select the *k* nearest neighbors. It is so challenging to identify the most suitable distance formula for all training samples. It means the best distance function is crucial to select the most competent *k* nearest neighbors that effectively make a majority-voting decision across most training samples.

Thirdly, KNN has high complexity due to the nearest neighbor (NN) search. It is challenging since KNN should calculate the distances of all samples to select the *k* nearest neighbors for each given query (test data). A recent method called NearCount can be used to select critical instances based on the cited counts of nearest neighbors (Zhu et al., 2020). An optimized high order product quantization can also be used to search approximate nearest neighbors (Li and Hu, 2020) that achieves high query efficiency and recall. Besides, a novel efficient method based on clustering and adaptive *k* values is probably utilized to find the *k*-nearest

* Corresponding author.
E-mail addresses: suyanto@telkomuniversity.ac.id (S. Suyanto), gppras@telkomuniversity.ac.id (P.E. Yunanto), tenia@ittelkom-pwt.ac.id (T. Wahyuningrum), siti@ittelkom-pwt.ac.id (S. Khomsah).

☆ This research is funded by the Directorate of Research and Community Service, Telkom University, with grant number: 571/PNLT3/PPM/2020.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2022.01.018
1319-1578/

neighbor (Gallego et al., 2022), which significantly reduces the complexity in computation.

Fourthly, KNN gives low effectiveness for imbalanced class datasets. The rule of majority voting used in KNN is generally successful for balanced classes but not for the imbalanced ones (Zhang and Member, 2019). It is challenging to design a proper classification rule to tackle balanced and imbalanced datasets generally.

Many KNN variants have been proposed to tackle the sensitivity of k and the classification rule. The sensitivity of k can be solved by reducing the effects of outliers using the local mean vectors of k nearest neighbors, such as k-harmonic nearest neighbor (KHNN) (Pan et al., 2017; Gou et al., 2019), local mean-based KNN (LMKNN) (Mitani and Hamamoto, 2006), local mean-based pseudo NN (LMPNN) (Gou et al., 2014), multi-local means based NN (MLMNN) (Gou et al., 2017).

The sensitivity of k can also be affected by the classification rule of majority voting since each neighbor has the same weight, which uses the principle of a democratic system: "one man one vote." However, nearest neighbors in the actual datasets usually have different contributions to classify a given query point. Therefore, some KNN variants that use weighted voting have been proposed to address this issue, such as distance-weighted KNN (Dudani, 1976), pseudo nearest neighbor (PNN) (Zeng et al., 2009), collaborative representation-based nearest neighbor (CRNN) (Li et al., 2015), weighted representation-based KNN (WRKNN) (Gou et al., 2019), weighted local mean representation-based KNN (WLMRKNN) (Gou et al., 2019), weighted discriminative collaborative competitive representation (WDCCR) (Gou et al., 2020), double competitive constraints-based collaborative representation for classification (DCCRC) (Gou et al., 2020), fuzzy k-nearest neighbor in every class (FkNNC) (Parande and Suyanto, 2019), and Bonferroni-mean based fuzzy k-nearest neighbor (BM-FKNN) (Mailagaha Kumbure et al., 2020).

However, all those KNN variants apply the local mean or the weighted voting scheme on each class, making them still sensitive to the outliers. Therefore, this paper proposes a new KNN variant named multi-voter multi-commission nearest neighbor (MVMCNN). It is inspired by the decision-making in a parliament with some commissions. The best decision on a particular issue, such as medicine, should be made by the health commission (as the most competent one). In MVMCNN, the data objects in each class are firstly clustered into some clusters (commissions). Next, the LMPNN is applied to each cluster instead of each class. However, this model is an early version of MVMCNN to keep low complexity. Improved versions can be developed using more advanced base models, such as WRKNN, WLMRKNN, WDCCR, DCCRC, or BM-FKNN. The proposed MVMCNN is then comprehensively evaluated based on the 10-fold cross-validation (FCV) using 30 datasets taken from the University of California Irvine (UCI) Machine Learning Repository (Irvine, 2021). Finally, its performance is analyzed and compared with KNN, LMPNN, and BM-FKNN using statistical tests of Friedman and Wilcoxon to see the significance of both accuracy and sensitivity of k.

Next, this paper is organized as follows. Brief reviews of the related works are given in Section 2. The motivation, concept, and detailed algorithm of the proposed MVMCNN are described in Section 3. Next, Section 4 provides a comprehensive evaluation, analysis, and discussion of MVMCNN performance. Finally, Section 5 gives the conclusions.

2. Related work

Based on the k nearest neighbors used in the classification rules, the nearest neighbor-based classifiers can be categorized into two schemes: single k-voters and multi k-voters. In the single-voter scheme, the k nearest neighbors (voters) are selected from all data objects in the training set, as used in KNN, WKNN, and BM-FKNN. In contrast, in the multi-voter scheme, the k nearest neighbors are chosen from each class; thus, they are named pseudo (not actual) nearest neighbors, as used in PNN, LMPNN, FkNNC, CRNN, WRKNN, WLMRKNN, WDCCR, and DCCRC.

In this research, MVMCNN is proposed as a novel nearest-neighbor variant to improve the multi-voter-based methods by introducing a multi-commission scheme. The LMPNN is chosen as the base model of MVMCNN because of two reasons. First, it is simple to implement and proven to outperform PNN (Gou et al., 2014). Second, it can be used as an early model to evaluate the benefits of the proposed multi-commission scheme. Once the multi-commission scheme shows many advantages, MVMCNN can be improved in the future by using any advanced multi-voter KNN variant, such as CRNN, WRKNN, WLMRKNN, WDCCR, and DCCRC.

Furthermore, the preliminary version of MVMCNN, built from a base model of LMPNN, is evaluated and compared to the original LMPNN using 30 datasets from UCI (Irvine, 2021) to see the benefits of the proposed multi-commission scheme. It also compared with two single-voter methods: KNN and BM-FKNN, to examine how significant its performance is. Here, the error rate and stability of k are selected as the performance metrics to give a focus to the detailed experiments. Besides, the statistical tests of Friedman and Wilcoxon are employed to see the significance of both metrics. More detailed performance metrics: Precision, Recall, and F1 Score, are also used here to verify the results, especially for several datasets with high-fatality and imbalanced-class.

2.1. k-nearest neighbor

KNN locally makes a decision using a certain number of closest data objects (nearest neighbors or voters) in the training set (Zhang et al., 2018). In a classification task, it classifies a given query or unlabelled test sample using a majority voting based on the k nearest neighbors from all classes (single-voter scheme), which is selected using a certain metric of distance or dissimilarity depending on the attribute types. For numerical data, dissimilarity is usually calculated using the Euclidean distance as

$$d(X,Y) = \sum_{i=1}^{n}(X_i - Y_i)^2, \tag{1}$$

where X and Y are data objects while n is the dimension (Harrison, 2018).

KNN has two steps that are not efficient in performing a classification task. The first step is the training process, which stores all data objects (training set) in the memory and finds the optimum k that provides the highest accuracy and generalizes unseen data in the future. It does not generate any model like the decision tree, neural network, or support vector machine. Meanwhile, the second step is the classification process. Each time classifying a given query, KNN should examine all the training samples to find k nearest ones.

KNN is easy to implement by setting an optimum k. The decision can be easily tracked so that the classification model can be updated quickly. Besides, it works locally, only considering the amount of data to be suitable for datasets that are grouped locally, where there are several separate data in a class. Nevertheless, it is not robust to the outliers due to the sensitivity of k and the too-simple classification rule of majority voting.

2.2. Pseudo nearest neighbor

The PNN is one of the KNN variants that have been proven to perform better than KNN for many datasets (Zeng et al., 2009). Based on the multi-voter scheme, PNN works by calculating the total distance

of k nearest neighbors in each class and then choosing a class with the lowest total distance as the classification decision (output). Those k neighbors are weighted gradually based on their distance-based rankings. The weight of the jth neighbor is formulated as one divided by j so that the first rank (the closest) neighbor weights 1; the second rank one weights $\frac{1}{2}$, and so on. In other words, the weights gradually decrease for the further neighbors so that the furthest neighbor has the lowest weight. The neighborhood weight W_j^i of the jth neighbor x_j^i from the class ith is formulated as

$$W_j^i = \frac{1}{j}, \tag{2}$$

where $j = 1, 2, \ldots, k$ is the distance-based ranking in ascending order.

Using Eq. (2), PNN gives a better work since it uses k nearest instances in each class to make a decision, which is conceptually better to tackle outliers or anomalies. This concept can be analogous to real life that the voice of the closest person should be more trusted than those who are further away.

2.3. Local mean pseudo nearest neighbor

LMPNN is a KNN variant that combines both LMKNN and PNN (Gou et al., 2014). Similar to PNN, the LMPNN is also developed using the multi-voter scheme and the neighborhood weight. Unlike PNN that selects the pseudo nearest neighbor in each class using the categorical k nearest neighbors, LMPNN finds them based on the categorical k local mean vectors. In LMPNN, the local mean vector $\bar{x}_j^i$ of the first j nearest neighbors to a query point x from the ith class is defined as

$$\bar{x}_j^i = \frac{1}{j}\sum_{l=1}^{j} x_l^i, \tag{3}$$

where $1 \leqslant j \leqslant k$.

Comprehensive experiments on 39 datasets: 32 UCI, 4 artificial, and 3 images show that LMPNN is more effective and robust than LMKNN, PNN, CFKNN, WKNN, and KNN classifiers.

2.4. Bonferroni mean fuzzy k-nearest neighbors

BM-FKNN is an improved version of the Fuzzy KNN (FKNN) that incorporates each class's "degree of truth" into KNN to make a majority voting-based decision. In the FKNN, class membership is assigned to a sample vector. Membership degree of the jth sample in the ith class of the training set is calculated as the inverse of the distances between the query y and k closest neighbors in the ith class formulated as (Keller et al., 1985)

$$u_i = \frac{\sum_{j=1}^{k} u_{ij}\left(1/||y - x_j||^{2/(m-1)}\right)}{\sum_{j=1}^{k}\left(1/||y - x_j||^{2/(m-1)}\right)} \tag{4}$$

where $m \in (1, +\infty)$ is a weight to define the neighbors' contribution to the membership degree. As suggested in Keller et al. (1985), in this research, m is set to 2, which is commonly used.

Different from FKNN, the BM-FKNN (Mailagaha Kumbure et al., 2020) utilizes the Bonferroni mean to calculate the local mean vectors. As described in Bonferroni (1950), Bonferroni mean is formulated as:

$$B^{p,q}(X) = \left(\frac{1}{n}\sum_{i=1}^{n} x_i^p \left(\frac{1}{n-1}\sum_{i\,j=1\,j\neq i}^{n} x_j^q\right)\right)^{\frac{1}{p+q}} \tag{5}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n), x_i \in [0, 1]\ \forall i \in \mathbb{N}$ is a vector with at least one $x_i \neq 0\ \forall i = 1, 2, \ldots, n$ while p and q are two parameters greater than or equal to 0. In this research, they are set to $p = 1$ and $q = 2$, respectively, as recommended for many datasets (Mailagaha Kumbure et al., 2020).

BM-FKNN has similar steps to FKNN. First, it estimates the distances between the given query (or testing data) and the training data. Next, k nearest neighbors are defined and grouped into sub-samples based on their classes. Furthermore, the Bonferroni mean vectors are calculated for the sub-samples representing each class. The distances between the query and those local vectors are then calculated to get the membership degrees based on Eq. 4. By maximizing the membership degrees, the query is finally classified as the highest membership degree class.

Different from KNN that directly compares a given query to k nearest neighbors, BM-FKNN uses the representative vectors for each class. The representative vectors are locally created using local sub-samples represented by the k nearest neighbors and well-positioned to perceive the class information. The local means can tackle the problems with imbalanced-class and similar interclass-samples (Mailagaha Kumbure et al., 2020). However, like KNN that uses a single-voter scheme, the value of k in BM-FKNN can be susceptible. A low k may not be enough to capture a large class-representative sub-samples so that the accurate local Bonferroni mean vectors are failed to create. In contrast, a too–high k can probably make the outliers (considered incompetent voters) affect the decision-making.

3. Proposed MVMCNN

Similar to LMPNN, the proposed MVMCNN is also developed using the multi-voter scheme and the neighborhood weight. However, MVMCNN is developed using a multi-commission scenario. First, the data objects in each class are clustered into some clusters (commissions). The optimum number of clusters is defined by maximizing the silhouette coefficient and the number of samples in the cluster. Next, it classifies a query point by minimizing the LMPNN distances of those commissions, where the total weighted rank-based distances of k instances in each commission is calculated using Eq. (3). Hence, in MVMCNN, the local mean vector $\bar{x}_j^{h,i}$ of the first j nearest neighbors to a query point x from the hth cluster and ith class is defined as

$$\bar{x}_j^{h,i} = \frac{1}{j}\sum_{l=1}^{j} x_l^i, \tag{6}$$

where $1 \leqslant j \leqslant k, 1 \leqslant h \leqslant N_c$, and N_c is the optimum number of clusters created using c-means clustering.

Finally, the best class decision C_{best} with the minimum distance among all the commissions is determined as

$$C_{best} = \arg\min_i \bar{x}_j^{h,i}. \tag{7}$$

Conceptually, the differences between MVMCNN and three previous methods KNN, PNN, and LMPNN, are depicted in Fig. 1. Suppose all methods use $k = 3$ to classify a given query point (purple square) into two classes: Class 1 (blue circle) and Class 2 (red triangle). Class 1 contains three clusters, which can be seen on the top, middle, and bottom, while Class 2 consists of two clusters on the top and bottom. Moreover, the orange circles represent the means of two and three first nearest neighbors with the weights of $\frac{1}{2}$ and $\frac{1}{3}$, respectively, in Class 1 and all the commissions. Meanwhile, the green triangles are the means of two and three first nearest neighbors with the weights of $\frac{1}{2}$ and $\frac{1}{3}$, respectively, in Class 2 and all the commissions.

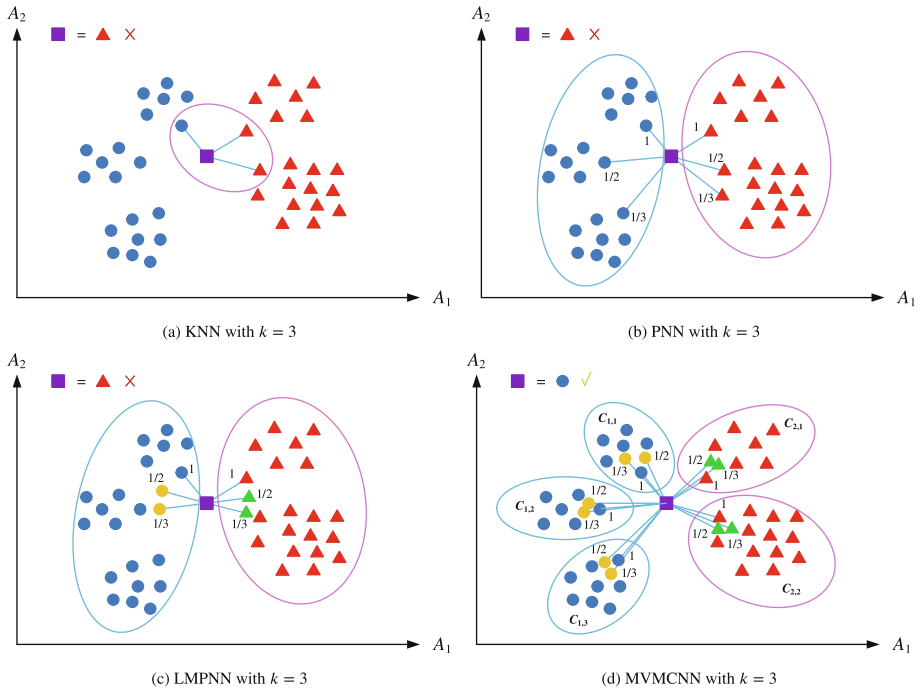

Fig. 1. Difference between KNN, PNN, LMPNN, and MVMCNN.

With $k = 3$, KNN wrongly classifies the query point into Class 2 (red triangle) since 2 out of 3 nearest neighbors are the red triangles. Using $k = 3$, PNN also fails to classify the query since the total distance between the query and the three nearest red triangles is less than that of the three closest blue circles. This failed classification is caused by the closest blue circles come from three different clusters, which make a bias decision. Besides, the three nearest red triangles also come from different clusters (one triangle from the first cluster and the rest from the second cluster). This problem can be analogous to a decision made by some members of the incompetent commissions that produces a bias decision.

Next, although the local means of the first j neighbors in each class gives more precision total distances, LMPNN wrongly classifies the query point. In this case, the total distance between the query point and the three closest red triangles is slightly smaller than the three blue circles. This wrong classification is also caused by the closest neighbors coming from different clusters and, unfortunately, the local means still give a bias decision.

In contrast, MVMCNN successfully classifies the query point into Class 1 (blue circle) since Commission 1 in Class 1 ($C_{1,1}$) gives a smaller total distance than all the other commissions in all classes. This classification is successful because all the closest blue circles and red triangles come from dense clusters, which are most competent to make a decision. There is no nearest neighbor from different clusters that makes a biased decision in calculating the local means. This successful classification can be analogous to the right decision made by several members of the competent commissions without interference by some members of the other incompetent commissions.

The pseudo-code of MVMCNN is depicted in Algorithm 1. It consists of five steps. First, the data objects in each class are clustered (using c-means clustering method) into several optimum clusters by maximizing both silhouette coefficient and cluster members. Here, the number of clusters for each class is limited to an interval [2, 6], which is found by a preliminary observation: each class in the thirty UCI datasets is maximally clustered into six clusters with high silhouette coefficients.

Algorithm 1: MVMCNN

Result: C_{best} as the output class
Cluster data in each class into optimum clusters of N_c by maximizing both silhouette coefficient and cluster members
Name the N_c clusters as $C_{i,m}$ that represent the ith class and the mth commission
for each commission, calculate the distance using Eq. (7)
Find the minimum distance from all the commissions $C_{i,m}$
Return the ith class with the lowest distance as the output class C_{best}

Next, the N_c clusters are named as $C_{i,m}$ that represent the ith class and the mth commission. For each commission, the total distance between the k pseudo nearest neighbors and the given query is determined using Eq. (7). Next, the total distances from all the commissions $C_{i,m}$ are minimized. Finally, the ith class with the lowest total distance is returned as the output class C_{best}.

In MVMCNN, each class can be split into two or more commissions using any clustering method, such as c-means, DBSCAN, fuzzy clustering, or the others. However, since MVMCNN is designed to be general for various datasets, it utilizes the c-means clustering that is simple with only one easy-to-tune parameter: number of clusters c. Meanwhile, DBSCAN requires two should-define-carefully parameters: radius and minimum points.

The clustering method plays an essential role in MVMCNN. It should generate clusters with the highest density possible. Thus, it is designed to maximize the silhouette coefficient and the number of samples in a cluster. Since the maximum k is set to 15, the number of samples in each cluster should be limited to 15. Hence, if the clustering generates a small cluster with less than 15 samples, the cluster will be merged into the closest one. However, if the small cluster cannot be merged because the number of clusters

is only 1 (minimum) or there is difficulty creating more than one cluster, it is kept as one cluster. Hence, for such a case, MVMCNN will perform equal to LMPNN since it does not split them into some clusters. In other words, it uses the original (single-commission) LMPNN. Therefore, it can be expected that MVMCNN guarantees to perform better than (or at least equal to) LMPNN.

Furthermore, since the UCI datasets are commonly small, the c-means clustering is applied with $c = 1$ to 6. Since this method is sensitive to the randomly generated initial points, it is applied to each class for some trials, and then the clusters with maximum silhouette coefficients are selected as the optimum commissions. It increases the computational cost but gives a higher probability of obtaining the maximum density clusters.

4. Results and discussion

The proposed MVMCNN is then comprehensively evaluated based on 10-fold cross-validation using 30 datasets taken from UCI repository (Irvine, 2021). Then, its performance is analyzed and compared with the three competitors using the statistical tests of Friedman and Wilcoxon to see the significance of both accuracy and sensitivity of k. Moreover, both Friedman mean rank and Wilcoxon rank sum tests are used to check the superiority of the MVMCNN. The experiments are carried out using an Intel Core i5-8300H processor and 8 GB of DDR4 with GPU NVidia Geforce GTX 1050Ti.

4.1. UCI datasets

The thirty datasets that have numerical attributes are taken from the UCI repository (Irvine, 2021) to make the experiments simple by exploiting the Euclidean distance. There are fifteen binary-class and fifteen multiclass classification problems. All the datasets are in the original version, without any preprocessing, except the four datasets marked with "*": Glass, Ecoli, Wine Quality Red, and Letter. Several classes containing a few samples in those first three datasets are removed to simplify the classification task. In contrast, the Letter dataset is subsampled to reduce the number of samples. The datasets have various samples: from 146 (Glass) to 10,992 (PenDigits). Besides, they also have varying dimensionality: from 3 (Haberman) to 856 (CNAE).

4.2. Optimum clusters

In MVMCNN, clustering is applied to each dataset to generate some commissions in each class. It is run to maximize the silhouette coefficient and the number of samples in a cluster. Since MVMCNN is implemented using LMPNN with $k = 1$ to 15 (Gou et al., 2014), the number of samples contained in each cluster in MVMCNN is limited to 15. Hence, if the clustering generates a small cluster with less than 15 samples, the cluster will be merged into the closest one. However, if the small cluster cannot be merged because the number of clusters is only 1 (minimum), it is accepted.

Here, the c-means clustering method is applied to each dataset five times with $c = 1$ to 6. It means that the number of clusters for each class is limited to an interval [1, 6] since the UCI datasets are commonly quite small. For instance, the dataset of Glass contains 146 samples: 70 in Class = 1 and 76 in Class = 2. The samples in each class are clustered into two or three clusters depending on the splitting result of 10-FCV. Another example, the dataset of Ecoli consists of 307 samples with four classes: cp = 143, im = 77, imU = 35, and pp = 52. The samples in each class are clustered into two or three clusters depending on the splitting result of 10-FCV, as illustrated in Table 1.

4.3. Classification performances

The classification performances of MVMCNN and the competitors: KNN, LMPNN, and BM-FKNN, are evaluated using thirty UCI datasets. The evaluation is performed using 10-FCV to see the error rate. In each dataset, the training samples are randomly divided into ten folds. Next, ten experiments are carried out to produce ten different classification error rates. Each experiment uses fifteen neighborhood sizes, from $k = 1$ to 15. The performance is determined by averaging the ten lowest error rates at the optimum k.

Table 2 illustrates the mean lowest error rates of KNN, LMPNN, BM-FKNN, and MVMCNN with the corresponding standard deviations and neighborhood sizes k in the parentheses. The lowest error rates are given in the bold text. It can be seen that MVMCNN gives the lowest error rates among the competitors for 17 out of 30 datasets: 9 binary-class and 8 multiclass problems. Meanwhile, KNN, LMPNN, and BM-FKNN win only for 4, 5, and 10 datasets, respectively. Besides, MVMCNN also gives low standard deviations that indicate its stability. The statistical test places MVMCNN at the first rank with a Friedman mean rank of 1.66, much smaller than KNN, LMPNN, and BM-FKNN, which produce 3.28, 2.52, and 2.24, respectively.

In general, MVMCNN outperforms all the competitors for the datasets with many attributes, such as CNAE, Musk1, HillValley, and Musk2 that contain 856, 166, 100, and 166 attributes, respectively. These results are achieved since those datasets contain many samples that can be clustered into several commissions. Thus, the multi-commission classification used in MVMCNN effectively reduces the bias decision made by the class-based distance in the LMPNN and majority voting in KNN and BM-FKNN.

Furthermore, the optimum k values giving the lowest error rates in MVMCNN are commonly lower than or the same as in the LMPNN since the decision is made in each cluster instead of each class. These results show that the decision of MVMCNN is made by the smaller number of the most competent voters in each cluster, with no interference from incompetent ones from other clusters.

Nevertheless, the comparison of the classification error rate above is not statistically convincible. Therefore, a statistical test called the Wilcoxon rank sum test with 95% confidence is used to show the significance of MVMCNN compared with both KNN, LMPNN, and BM-FKNN. Wilcoxon rank-sum test is the sum of the ranks for observations from one of the samples, commonly used to test whether two samples are likely to derive from the same population. In other words, it examines if the two populations have the same shape or not. Here, the Wilcoxon rank sum test is applied to the pairwise of the ten error rates (from ten experiments) obtained by KNN, LMPNN, BM-FKNN, and MVMCNN for each dataset.

Table 3 illustrates Wilcoxon's rank sum test of the KNN, LMPNN, and BM-FKNN versus the proposed MVMCNN for the thirty UCI datasets. The symbols "−" or "+" represents that the current result is significantly worse or better than the result of MVMCNN in terms of Wilcoxon's rank sum test at a 0.05 significance level, respectively, while the symbol "≈" denotes a similar (not significant) result. It can be seen that MVMCNN significantly outperforms KNN for 23 datasets, ties for one dataset of Banknote, and loses for 6 datasets. It is also much better than LMPNN for 21 datasets and ties for 9 datasets. Finally, it significantly outperforms BM-KNN for 16 datasets, ties for 2 datasets: Climate Model Simulation Crashes and Wdbc, and loses for 12 datasets.

4.4. Stability of k

The robustness of MVMCNN is finally investigated based on the sensitivity of k in comparison with the competitors. The experi-

Table 1
Number of clusters in each class for the Ecoli dataset.

Fold	Class = cp	Class = im	Class = imU	Class = pp
1	2	2	2	2
2	2	2	2	2
3	3	3	3	3
4	2	3	3	2
5	2	2	2	2
6	2	2	2	2
7	3	2	3	3
8	3	3	3	2
9	2	2	2	2
10	2	2	2	2

Table 2
The lowest mean error rates (%) produced by KNN, LMPNN, BM-FKNN, and MVMCNN as well as the corresponding standard deviations and k in the parentheses for thirty UCI datasets.

Dataset	#Sample	#Attribute	#Class	KNN	LMPNN	BM-FKNN	MVMCNN
Parkinsons	195	22	2	20.63 ± 8.61 (15)	22.24 ± 12.35 (14)	**15.17 ± 7.6** (8)	18.08 ± 11.19 (10)
Glass*	146	9	2	8.7 ± 3.54 (3)	6.63 ± 5.01 (7)	9.04 ± 8.27 (1)	**6.25 ± 3.92** (8)
Climate Model SC	540	18	2	**8.52 ± 0.96** (15)	10.19 ± 2.79 (15)	9.45 ± 1.05 (15)	9.63 ± 1.7 (15)
Musk1	476	166	2	19.48 ± 9.81 (2)	17.81 ± 9.72 (12)	18.3 ± 4.46 (2)	**17.18 ± 9.2** (13)
Transfusion	748	4	2	26.07 ± 8.99 (14)	32.2 ± 9.12 (15)	**23.28 ± 4.67** (4)	27.53 ± 4.29 (15)
Wdbc	569	30	2	6.67 ± 4.38 (10)	**6.39 ± 2.94** (7)	6.39 ± 3.22 (13)	**6.39 ± 2.94** (7)
Plrx	180	12	2	29.1 ± 2.17 (8)	36.78 ± 8.49 (15)	30.79 ± 14.15 (2)	**26.99 ± 5.72** (11)
Haberman	306	3	2	**24.86 ± 10.13** (5)	33.90 ± 4.11 (13)	26.46 ± 7.42 (2)	33.55 ± 16.98 (13)
HillValley	1,212	100	2	41.58 ± 4.87 (1)	35.72 ± 4.01 (15)	37.58 ± 3.94 (4)	**29.62 ± 8.49** (15)
Ionosphere	351	34	2	11.67 ± 5.58 (2)	11.67 ± 7.02 (13)	**10.19 ± 4.14** (2)	11.10 ± 6.64 (15)
Wpbc	198	32	2	21.66 ± 6.98 (2)	21.24 ± 6.34 (13)	23.17 ± 11.23 (2)	**20.24 ± 5.54** (13)
Sonar	208	60	2	39.07 ± 5.48 (2)	38 ± 2.73 (6)	38.68 ± 4.42 (8)	**37.98 ± 2.78** (8)
Banknote	1,372	4	2	**0.00 ± 0.00** (2)	0.07 ± 0.23 (1)	**0.00 ± 0.00** (3)	0.07 ± 0.23 (1)
QSAR biodegradation	1,055	41	2	20.29 ± 5.92 (3)	19.26 ± 6.82 (15)	19.46 ± 2.76 (5)	**19.16 ± 6.75** (15)
Musk2	6,598	166	2	18.26 ± 16.92 (14)	17.69 ± 15.31 (15)	22.83 ± 1.19 (8)	**16.85 ± 16.01** (15)
Wine	178	13	3	25.2 ± 10.05 (1)	24.35 ± 7.69 (13)	24.02 ± 7.41 (7)	**23.99 ± 14.49** (3)
Seed	210	7	3	9.05 ± 9.64 (11)	11.43 ± 10.58 (3)	**7.62 ± 6.02** (12)	10.48 ± 7.38 (12)
Thyroid	7,200	21	3	1.8 ± 0.75 (6)	1.65 ± 1.08 (10)	**1.46 ± 0.52** (5)	1.60 ± 1.10 (10)
Ecoli*	307	7	4	**14.01 ± 4.76** (7)	15.47 ± 4.11 (13)	14.35 ± 8.09 (4)	15.47 ± 4.11 (13)
Robot Navigation	5,456	4	4	2.66 ± 0.87 (1)	**2.52 ± 0.75** (3)	2.66 ± 0.63 (1)	**2.52 ± 0.75** (3)
Wine Quality Red*	1,571	11	4	51.91 ± 3.71 (13)	51.53 ± 4.48 (14)	**44.27 ± 4.42** (8)	50.83 ± 4.56 (14)
Page-Blocks	5,473	10	5	4.06 ± 1.03 (3)	**3.11 ± 0.80** (9)	3.93 ± 0.78 (5)	**3.11 ± 0.80** (9)
LandsatSatellite	6,435	36	6	11.66 ± 2.2 (1)	9.29 ± 1.69 (15)	**7.41 ± 1.03** (11)	9.25 ± 1.78 (15)
CNAE	1,080	856	9	11.58 ± 1.46 (14)	8.98 ± 2.9 (15)	9.91 ± 2.1 (13)	**8.80 ± 3.12** (15)
Letter*	7,648	16	10	4.35 ± 0.48 (3)	2.94 ± 0.44 (15)	3.00 ± 0.28 (15)	**2.08 ± 3.18** (15)
Cardiotocography	2,126	21	10	47.88 ± 4.43 (7)	**44.96 ± 5.17** (13)	46.38 ± 2.97 (15)	**44.96 ± 5.17** (13)
PenDigits	10,992	16	10	0.69 ± 0.22 (1)	**0.25 ± 0.14** (13)	0.45 ± 0.25 (11)	**0.25 ± 0.14** (13)
OptDigits	5,620	64	10	1.55 ± 0.76 (3)	0.96 ± 0.43 (9)	**0.82 ± 0.43** (13)	0.94 ± 0.4 (8)
Vowel	528	10	11	36.26 ± 5.78 (1)	36.06 ± 8.02 (15)	36.26 ± 1.73 (2)	**35.35 ± 5.41** (8)
Libras Movement	360	90	15	16.39 ± 15.01 (1)	15.00 ± 13.04 (5)	**13.89 ± 6.14** (13)	15.00 ± 13.04 (5)
Friedman Mean Rank				3.28	2.52	2.24	**1.66**
Rank				4	3	2	**1**

mental results are given in terms of the mean error rates with varying k (from 1 to 15) for each dataset. The thirty datasets are split into two divisions: binary-class and multiclass problems (each has fifteen datasets) to give clear comparisons.

Fig. 2 illustrates the mean error rates with varying k (from 1 to 15) obtained by KNN, LMPNN, BM-FKNN, and MVMCNN for the fifteen binary-class datasets. It shows that KNN produces the lowest stability of k, where the mean error rates fluctuate, for the most (13 out of 15) binary-class datasets. The mean error rates are generally low for odd k but higher for even k since the majority voting produces ties decisions, making KNN works randomly. It gives high stability of k only for two datasets: Glass and Banknote. These results inform that the majority voting scheme is not robust to make a classification decision in the binary-class datasets.

Like KNN, the BM-FKNN is also unstable for 13 out of 15 binary-class datasets and stable only for two datasets: HillValley and QSAR biodegradation. In general, it obtains high mean error rates for $k = 1$, and then stable for small $k = 2$ to 6, but unstable for the higher k (7 or more). These results are affected by the single-voter scheme with majority voting used in BM-FKNN. A low k is not enough to capture a large class-representative sub-samples so that the accurate local Bonferroni mean vectors are failed to create. Contrarily, a too–high k makes the outliers (incompetent voters) give a wrong decision.

Next, LMPNN gives high stable mean error rates for 13 out of 15 binary-class datasets. It produces low stability only for two datasets: Glass and Sonar. These results indicate that the weighted multi-voter scheme (that considers weighted local means of

Table 3
The Wilcoxon's rank sum test of the KNN, LMPNN, and BM-FKNN versus the proposed MVMCNN for the thirty UCI datasets. The symbols − or + represents that the current result is significantly worse or better than the result of MVMCNN in terms of Wilcoxon's rank sum test at a 0.05 significance level, respectively, while the symbol ≈ denotes a similar (not significant) result.

Dataset	# Samples	# Attributes	# Classes	KNN	LMPNN	BM-FKNN
Parkinsons	195	22	2	− **	−	+
Glass*	146	9	2	−	−	−
Climate Model Simulation Crashes	540	18	2	+	−	≈
Musk1	476	166	2	−	−	−
Transfusion	748	4	2	+	−	+
Wdbc	569	30	2	−	≈	≈
Plrx	180	12	2	−	−	−
Haberman	306	3	2	+	−	+
HillValley	1,212	100	2	−	−	−
Ionosphere	351	34	2	−	−	+
Wpbc	198	32	2	+	≈	−
Sonar	208	60	2	−	−	−
Banknote	1,372	4	2	≈	≈	+
QSAR biodegradation	1,055	41	2	−	−	−
Musk2	6,598	166	2	−	−	−
Wine	178	13	3	−	−	−
Seed	210	7	3	+	−	+
Thyroid	7,200	21	3	−	−	+
Ecoli*	307	7	4	+	≈	+
Robot Navigation	5,456	4	4	−	≈	−
Wine Quality Red*	1,571	11	4	−	−	+
Page-Blocks	5,473	10	5	−	≈	−
LandsatSatellite	6,435	36	6	−	−	+
CNAE	1,080	856	9	−	−	−
Letter*	7,648	16	10	−	−	−
Cardiotocography	2,126	21	10	−	≈	−
PenDigits	10,992	16	10	−	≈	−
OptDigits	5,620	64	10	−	−	+
Vowel	528	10	11	−	−	−
LibrasMovement	360	90	15	−	≈	+
Wilcoxon (−)				23	21	16
Wilcoxon (≈)				1	9	2
Wilcoxon (+)				6	0	12

k-nearest neighbors in each class) used in LMPNN can make a better classification than the single-voter decision (with a "one man one vote" majority voting rule) used in KNN.

Meanwhile, the proposed MVMCNN is the most robust compared with both KNN, LMPNN, and BM-FKNN, where the various *k* (from 1 to 15) give stable mean error rates for the most (14 out of 15) binary-class datasets. The MVMCNN produces low robustness only for the Glass dataset, where the mean error rates are quite low only for small *k* (1 to 3), but the bigger the *k* (4 to 15), the higher the error rates.

Furthermore, the sensitivity of *k* given by MVMCNN is similar to that produced by LMPNN in four cases: Climate Model Simulation Crashes, Musk1, Ionosphere, and QSAR biodegradation, but MVMCNN gives lower error rates. It is the same as LMPNN for both Wdbc and Banknote datasets. These results prove that the proposed multi-commission decision used in MVMCNN is capable of enhancing the decision rule in LMPNN for the fifteen binary-class datasets.

Next, Fig. 3 illustrates the mean error rates with varying *k* (from 1 to 15) given by KNN, LMPNN, BM-FKNN, and MVMCNN for the fifteen multiclass datasets. It can be seen that KNN produces the lowest stability of *k*, where the mean error rates are fluctuating, for the most (12 out of 15) multiclass datasets. It is stable only for three datasets: Ecoli, Page-Blocks, and LandsatSatellite. These results also inform that the majority voting scheme is not robust to make a classification decision for the multiclass datasets.

Meanwhile, BM-FKNN obtains higher stabilities than KNN. It stables for the most (9 out of 15) multiclass datasets. Similar to the results in binary-class datasets, it obtains high mean error rates for $k = 1$, and then stable for small $k = 2$ to 6, but unstable for $k = 7$ or more. These results indicate that the BM-FKNN provides an improvement for the single-voter scheme with majority voting since it considers the representative vectors for each class, instead of the direct closest samples.

In contrast, LMPNN gives high stable mean error rates for all the multiclass datasets. These results indicate that the multi-voter scheme with weighted local mean distance used in LMPNN can make a better decision than the single-voter rule used in KNN.

Finally, the proposed MVMCNN gives stable error rates for 13 out of 15 multiclass datasets. It provides low robustness only for both Ecoli and LibrasMovement datasets, where the error rates are low for small *k* (1 to 3), but the bigger the *k* (4 to 15), the higher the error rates. These results are caused by the small size of the datasets, with 307 and 360 samples belonging to 4 and 15 classes. It informs that the proposed multi-commission decision used in MVMCNN can fail to enhance the LMPNN for small multiclass datasets since only a small number of commissions can be generated.

4.5. Precision, recall, and F1 score

Most of the datasets used in this research are considered balanced-classes. Meanwhile, 10 out of 30 (33.34%) are categorized as imbalanced-classes with the positive labeled data is slightly fewer than the negative and the rest (6 out of 30 or 20%) datasets are imbalanced-classes with much fewer positive labeled data, namely Climate Model Simulation Crashes, Tyroid, Ecoli, Wine Quality Red, Page-Blocks, and Cardiotocography.Table 4 shows the Precision, Recall, and F1 Score given by MVMCNN and the other methods, where the values in parentheses indicate the optimum neighborhood size *k*. It can be seen that all methods give the same

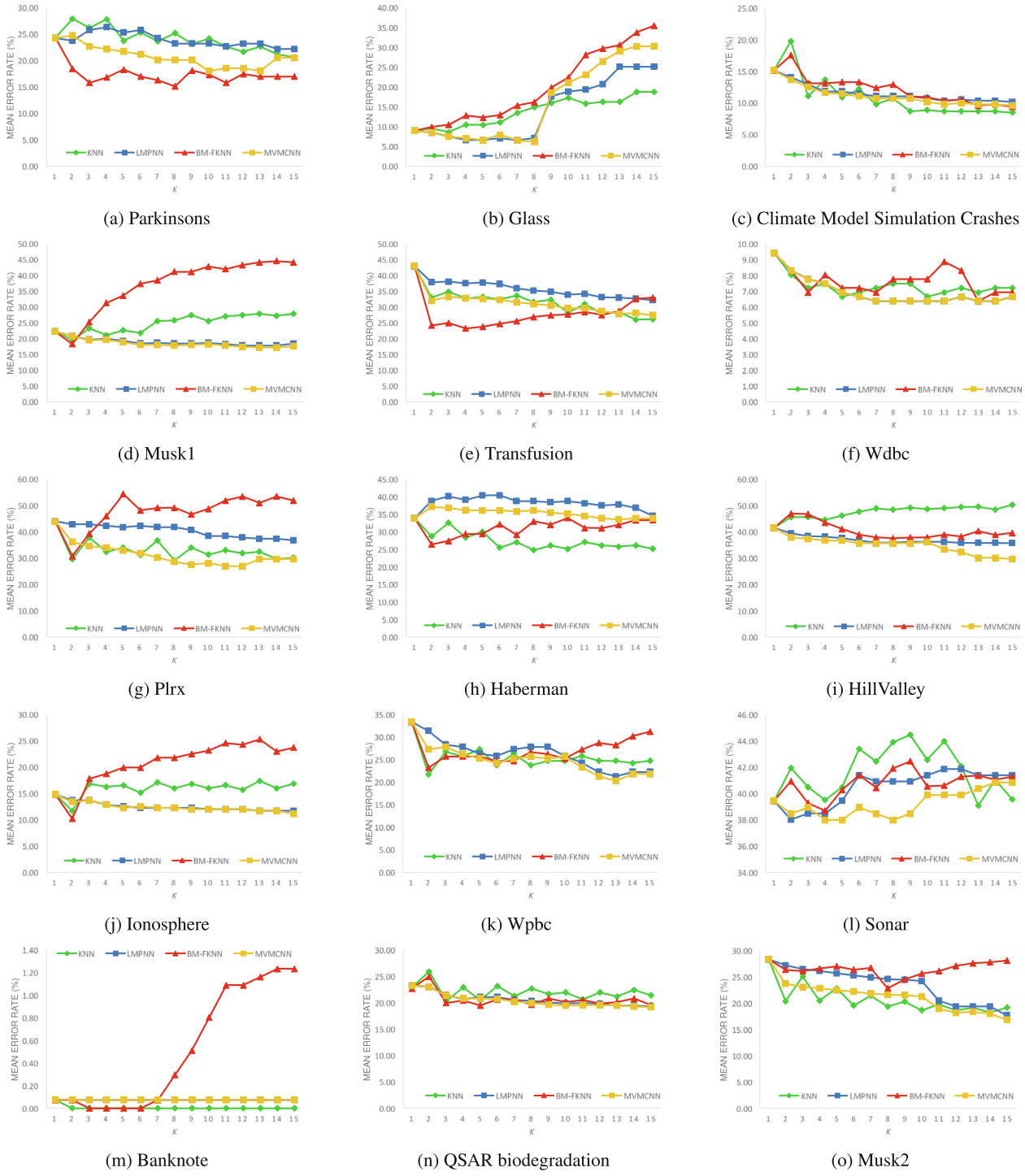

Fig. 2. Mean error rates given by KNN, LMPNN, and MVMCNN using various k for the binary-class datasets.

optimum k as used in the error rate metric for the balanced-class datasets since the formula of the F1 Score can be considered equivalent to Accuracy when the numbers of data in the positive and negative classes are the same (or similar). However, the optimum k are different from those used in the error rate metric for imbalanced-class datasets, especially those with much fewer positive-labeled data.

MVMCNN gives the highest performance in terms of Precision, Recall, and F1 among the competitors. It reaches better (or equal) metrics for 17 out of 30 datasets: 9 binary-class and 8 multiclass problems. Meanwhile, KNN, LMPNN, and BM-FKNN perform better only for 4, 5, and 10 datasets, respectively. The Friedman mean rank places MVMCNN at the first rank with 1.70 while KNN, LMPNN, and BM-FKNN produce 3.17, 2.43, and 2.23, respectively.

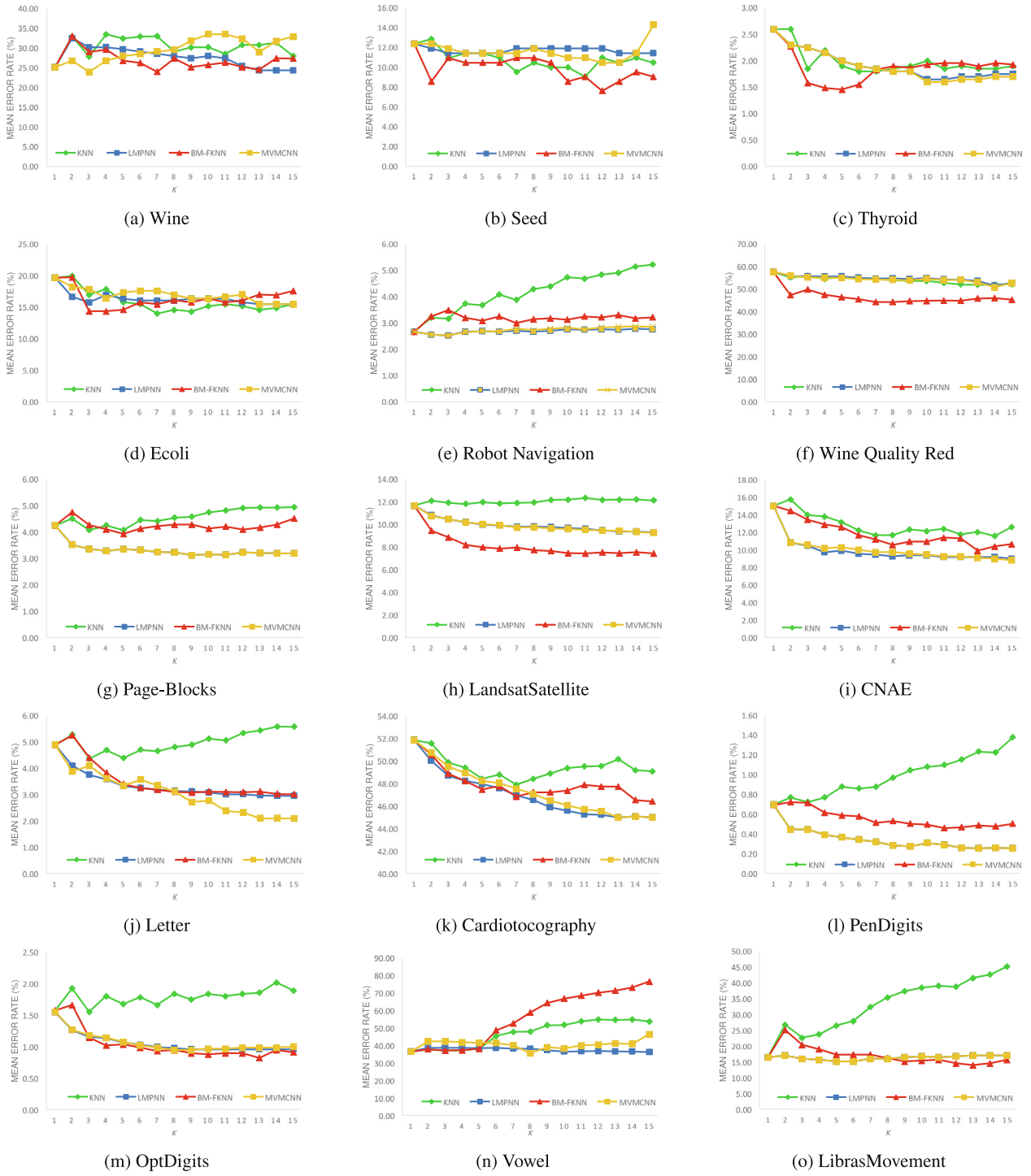

Fig. 3. Mean error rates given by KNN, LMPNN, and MVMCNN using various k for the multiclass datasets.

These results indicate that the proposed multi-commission scheme, which makes a more local decision, can better classify imbalanced-class datasets.

4.6. Usability

Those results show that the proposed multi-commission scheme gives a benefit to the LMPNN as one of the multi-voter KNN variants by reducing the error rate and increasing the k stability. Therefore, MVMCNN can probably be used to improve the recent multi-voter KNN variants, such as FkNNC, CRNN, WRKNN, WLMRKNN, WDCCR, and DCCRC. Hypothetically, the multi-commission scheme also obtains such advantages since it provides more local decision-making. However, since MVMCNN utilizes a c-means clustering method, it can be not only more sensitive to some datasets but also more complex in computation. Hence, two advanced procedures may be applied to solve both issues. First, an advanced clustering method, such as hierarchical clustering, can be utilized to generate more stable clusters. Second, an efficient method based on clustering and adaptive k values to find the k-nearest neighbor as proposed (Gallego et al., 2022), can be exploited to reduce those complexities. By utilizing those advanced

Table 4
The highest Precision, Recall, and F1 Score (%) produced by KNN, LMPNN, BM-FKNN, and MVMCNN as well as the corresponding k in the parentheses for thirty UCI datasets.

Dataset	Precision				Recall				F1 Score			
	KNN	LMPNN	BM-FKNN	MVMCNN	KNN	LMPNN	BM-FKNN	MVMCNN	KNN	LMPNN	BM-FKNN	MVMCNN
Parkinsons	67.05	59.32	**84.47**	80.58	31.88	30.69	**36.25**	35.00	43.21 (14)	40.45 (12)	**50.73 (6)**	48.8 (10)
Glass*	91.00	93.56	90.92	**94.34**	90.29	**92.14**	89.57	**92.14**	90.64 (3)	92.85 (7)	90.24 (1)	**93.23 (8)**
Climate Model SC	**50.00**	36.11	41.25	40.07	**27.17**	25.43	25.87	26.30	**35.21 (9)**	29.85 (10)	31.8 (12)	31.76 (11)
Musk1	79.87	80.96	81.34	**82.10**	73.82	77.20	75.17	**77.34**	76.72 (4)	79.03 (4)	78.13 (2)	**79.65 (4)**
Transfusion	44.67	32.83	**51.28**	41.72	40.00	33.76	**43.88**	39.55	42.21 (12)	33.29 (14)	**47.29 (4)**	40.61 (12)
Wdbc	91.34	**91.63**	**91.63**	**91.63**	90.62	**91.09**	**91.09**	**91.09**	90.98 (7)	**91.36 (6)**	**91.36 (11)**	**91.36 (7)**
Plrx	45.89	29.12	40.94	**52.92**	26.80	22.60	24.40	**29.00**	33.84 (9)	25.45 (10)	30.58 (5)	**37.47 (6)**
Haberman	54.90	39.06	**50.94**	39.68	34.57	30.86	**33.33**	30.86	42.42 (3)	34.48 (6)	**40.3 (5)**	34.72 (7)
HillValley	58.53	65.34	63.49	**73.59**	57.79	60.81	58.45	**63.58**	58.16 (1)	62.99 (15)	60.87 (4)	**68.22 (15)**
Ionosphere	82.18	82.18	**84.96**	82.98	86.19	86.19	**86.98**	86.67	84.14 (2)	84.14 (14)	**85.96 (2)**	84.78 (15)
Wpbc	57.66	58.95	53.43	**61.64**	40.00	40.83	34.42	**43.75**	47.23 (2)	48.25 (14)	41.87 (2)	**51.18 (13)**
Sonar	66.03	67.51	66.86	**67.58**	33.40	35.67	33.81	**35.67**	44.36 (2)	46.68 (6)	44.91 (8)	**46.69 (8)**
Banknote	**100.00**	99.85	**100.00**	99.85	**100.00**	**100.00**	**100.00**	**100.00**	**100.00 (2)**	99.93 (1)	**100.00 (3)**	99.93 (1)
QSAR biodegradation	85.46	88.72	87.96	**88.84**	48.03	49.16	49.04	**49.44**	61.5 (3)	63.26 (15)	62.98 (5)	**63.53 (15)**
Musk2	43.65	44.81	34.09	**46.74**	63.52	63.87	51.52	**66.86**	51.74 (14)	52.67 (15)	41.03 (9)	**55.02 (15)**
Wine	52.54	53.66	54.10	**56.45**	64.58	68.75	68.75	**72.92**	57.94 (1)	60.27 (13)	60.55 (7)	**63.64 (3)**
Seed	83.78	81.69	**85.71**	82.99	88.57	82.86	**94.29**	87.14	86.11 (11)	82.27 (3)	**89.8 (12)**	85.02 (12)
Thyroid	55.50	57.73	**61.46**	58.88	73.51	74.17	**78.15**	76.82	63.25 (7)	64.93 (9)	**68.8 (6)**	66.67 (8)
Ecoli*	**42.00**	37.25	40.59	37.25	**60.00**	54.29	58.57	54.29	**49.41 (6)**	44.19 (12)	47.95 (4)	44.19 (11)
Robot Navigation	79.87	**81.82**	79.87	**81.82**	75.00	**76.83**	75.00	**76.83**	77.36 (1)	**79.25 (3)**	77.36 (1)	**79.25 (3)**
Wine Quality Red*	15.41	15.53	**17.82**	15.74	58.49	58.49	**60.38**	58.49	24.4 (12)	24.54 (14)	**27.52 (7)**	24.81 (14)
Page-Blocks	52.85	**68.53**	54.54	**68.53**	92.86	**96.43**	93.57	**96.43**	67.36 (4)	**80.12 (9)**	68.91 (6)	**80.12 (9)**
LandsatSatellite	45.10	51.26	**56.72**	51.26	92.65	97.44	**98.40**	97.44	60.67 (1)	67.18 (14)	**71.96 (11)**	67.18 (14)
CNAE	48.05	**57.05**	53.75	**57.69**	62.18	74.79	72.27	**75.63**	54.21 (14)	64.73 (15)	61.65 (13)	**65.45 (15)**
Letter*	76.45	86.49	86.10	**89.95**	81.15	83.77	83.51	**89.01**	78.73 (3)	85.11 (15)	84.78 (15)	**89.47 (15)**
Cardiotocography	39.22	46.28	41.34	**46.28**	75.47	**86.79**	79.25	**86.79**	51.61 (7)	**60.37 (13)**	54.33 (15)	**60.37 (13)**
PenDigits	96.93	**98.77**	98.46	**98.77**	95.83	**98.67**	96.77	**98.67**	96.37 (1)	**98.72 (13)**	97.61 (11)	**98.72 (13)**
OptDigits	90.91	92.78	**93.06**	92.78	90.42	97.65	**99.46**	97.65	90.66 (3)	95.15 (9)	**96.15 (13)**	95.15 (8)
Vowel	62.26	63.71	63.06	**65.56**	61.36	61.36	61.36	**61.36**	61.81 (1)	62.51 (15)	62.2 (2)	**63.39 (8)**
Libras Movement	75.00	81.01	**83.90**	81.01	73.04	77.91	**80.35**	77.91	74.01 (1)	79.43 (5)	**82.08 (13)**	79.43 (5)
Friedman Mean Rank	3.20	2.47	2.20	**1.63**	2.93	2.17	2.13	**1.60**	3.17	2.43	2.23	**1.70**
Rank	4	3	2	**1**	4	3	2	**1**	4	3	2	**1**

procedures, MVMCNN can be improved to obtain a low mean error rate and low computational complexity. Moreover, it can be used to develop various real-world applications based on detection, classification, recognition, and others.

5. Conclusion

A novel KNN variant named MVMCNN has been successfully implemented. It is inspired by both the decision-making in a parliament with some commissions. As an early model, it was developed using a base model of LMPNN. However, unlike LMPNN, it makes a decision based on the total distances from multi k nearest neighbors (voters) from each cluster (commission) instead of each class. Evaluation based on a 10-FCV using 30 UCI datasets shows that it outperforms KNN, LMPNN, and BM-FKNN classifiers. Moreover, a detailed investigation informs that it is also more robust, with a lower sensitivity of k, than the three competitors. It can solve some datasets with outliers because the c-means clustering is exploited to split them into small clusters (commissions). However, for some datasets with high-density points in each class, MVMCNN performs the same as the LMPNN since it does not split them into some clusters. In other words, it is enforced to use the original (single-commission) LMPNN for datasets with dense classes. Therefore, it can be concluded that MVMCNN guarantees to perform better than (or at least equal to) LMPNN. Compared to the single-voter models: KNN and BM-FKNN, the proposed MVMCNN also gives lower mean error rates as well as higher Precision, Recall, and F1 Score. This result shows that the multi-voter model provides a better decision than the single-voter ones.

Nevertheless, this study has a limitation since the proposed MVMCNN is evaluated and compared to the competitors only based on the mean error rate, which is a general performance metric. It does not comprehensively give detailed investigations for the specific datasets that are imbalanced-classes, sensitive, and give a high fatality, such as in medical and security applications. For instance, an application of Covid-19 detection needs to be evaluated in more detail based in a confusion matrix since the dataset is imbalance, and false negative or false positive gives a fatal impact.

In the future, it will be enhanced by leveraging advanced base models of nearest neighbors, such as FkNNC, CRNN, WRKNN, WLMRKNN, WDCCR, DCCRC, and BM-FKNN, to obtain much better performance in terms of mean error rate and sensitivity of k. Besides, a comprehensive performance analysis using the confusion matrix will be utilized to provide detailed investigations, especially for imbalanced-class datasets with high fatality.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Bonferroni, C., 1950. Sulle medie multiple di potenze. Bollettino Della Unione Matematica Italiana 5, 267–270.

Bulut, F., Amasyali, M.F., 2017. Locally adaptive k parameter selection for nearest neighbor classifier: one nearest cluster. Pattern Anal. Appl. 20, 415–425. https://doi.org/10.1007/s10044-015-0504-0. https://doi.org/10.1007/s10044-015-0504-0.

Dudani, S.A., 1976. The distance-weighted k-nearest-neighbor rule. IEEE Trans. Syst., Man, Cybern. SMC-6, 325–327. https://doi.org/10.1109/TSMC.1976.5408784.

Gallego, A.J., Rico-Juan, J.R., Valero-Mas, J.J., 2022. Efficient k-nearest neighbor search based on clustering and adaptive k values. Pattern Recogn. 122,. https://doi.org/10.1016/j.patcog.2021.108356. URL: https://www.sciencedirect.com/science/article/pii/S0031320321005367 108356.

Gou, J., Qiu, W., Mao, Q., Zhan, Y., Shen, X., Rao, Y., 2017. A multi-local means based nearest neighbor classifier. In: 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 448–452. https://doi.org/10.1109/ICTAI.2017.00075.

Gou, J., Qiu, W., Yi, Z., Shen, X., Zhan, Y., Ou, W., 2019. Locality constrained representation-based K-nearest neighbor classification. Knowl.-Based Syst. 167, 38–52. https://doi.org/10.1016/j.knosys.2019.01.016.

Gou, J., Wang, L., Yi, Z., Yuan, Y., Ou, W., Mao, Q., 2020. Weighted discriminative collaborative competitive representation for robust image classification. Neural Networks 125, 104–120. https://doi.org/10.1016/j.neunet.2020.01.020. URL: https://www.sciencedirect.com/science/article/pii/S0893608020300228.

Gou, J., Wu, H., Song, H., Du, L., Ou, W., Zeng, S., Ke, J., 2020. Double Competitive Constraints-Based Collaborative Representation for Pattern Classification. Comput. Electr. Eng. 84,. https://doi.org/10.1016/j.compeleceng.2020.106632. URL: https://www.sciencedirect.com/science/article/pii/S0045790620304870 106632.

Gou, J., Zhan, Y., Rao, Y., Shen, X., Wang, X., He, W., 2014. Improved pseudo nearest neighbor classification. Knowl.-Based Syst. 70, 361–375. https://doi.org/10.1016/j.knosys.2014.07.020.

Harrison, O., 2018. Machine learning basics with the k-nearest neighbors algorithm. https://towardsdatascience.com/machine-learning-basics-with-he-k-nearest-neighbors-algorithm-6a6e71d01761.

Irvine, U.o.C., 2021. UCI Machine Learning Repository.https://archive.ics.uci.edu/ml/index.php.

Keller, J.M., Gray, M.R., Givens, J.A., 1985. A fuzzy k-nearest neighbor algorithm. IEEE Trans. Syst., Man, Cybern. SMC-15, 580–585. https://doi.org/10.1109/TSMC.1985.6313426.

Li, L., Hu, Q., 2020. Optimized high order product quantization for approximate nearest neighbors search. Front. Comput. Sci. 14, 259–272. https://doi.org/10.1007/s11704-018-7049-5.

Li, W., Du, Q., Zhang, F., Hu, W., 2015. Collaborative-representation-based nearest neighbor classifier for hyperspectral imagery. IEEE Geosci. Remote Sens. Lett. 12, 389–393. https://doi.org/10.1109/LGRS.2014.2343956.

Mailagaha Kumbure, M., Luukka, P., Collan, M., 2020. A new fuzzy k-nearest neighbor classifier based on the Bonferroni mean. Pattern Recogn. Lett. 140, 172–178. https://doi.org/10.1016/j.patrec.2020.10.005. URL: https://www.sciencedirect.com/science/article/pii/S0167865520303792.

Mitani, Y., Hamamoto, Y., 2006. A local mean-based nonparametric classifier. Pattern Recogn. Lett. 27, 1151–1159. https://doi.org/10.1016/j.patrec.2005.12.016.

Pan, Z., Wang, Y., Ku, W., 2017. A new k-harmonic nearest neighbor classifier based on the multi-local means. Expert Syst. Appl. 67, 115–125. https://doi.org/10.1016/j.eswa.2016.09.031.

Papernot, N., Mcdaniel, P., 2018. Deep k-Nearest Neighbors: Towards Confident, Interpretable and Robust Deep Learning. arXiv arXiv:arXiv:1803.04765v1..

Parande, E.A., Suyanto, S., 2019. Indonesian graphemic syllabification using a nearest neighbour classifier and recovery procedure. Int. J. Speech Technol. 22, 13–20. https://doi.org/10.1007/s10772-018-09569-3. URL: https://link.springer.com/article/10.1007/s10772-018-09569-3.

Tan, M., Zhang, S., Wu, L., 2020. Mutual kNN based spectral clustering. Neural Comput. Appl. 32, 6435–6442. https://doi.org/10.1007/s00521-018-3836-z.

Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.H., Steinbach, M., Hand, D.J., Steinberg, D., 2008. Top 10 algorithms in data mining. vol. 14. https://doi.org/10.1007/s10115-007-0114-2.

Zeng, Y., Yang, Y., Zhao, L., 2009. Pseudo nearest neighbor rule for pattern classification. Expert Syst. Appl. 36, 3587–3595. https://doi.org/10.1016/j.eswa.2008.02.003.

Zhang, S., Li, X., Zong, M., Zhu, X., Cheng, D., 2017. Learning k for KNN Classification. ACM Trans. Intell. Syst. Technol. 8. https://doi.org/10.1145/2990508, DOI: 10.1145/2990508.

Zhang, S., Li, X., Zong, M., Zhu, X., Wang, R., 2018. Efficient kNN Classification With Different Numbers of Nearest Neighbors. IEEE Trans. Neural Networks Learn. Syst. 29, 1774–1785. https://doi.org/10.1109/TNNLS.2017.2673241.

Zhang, S., Li, X., Zong, M., Zhu, X., Wang, R., 2018. Efficient knn classification with different numbers of nearest neighbors, pp. 1774–1785. 10.1109/TNNLS.2017.2673241..

Zhang, S., Member, S., 2019. Challenges in KNN Classification. IEEE Trans. Knowl. Data Eng., 1–13 https://doi.org/10.1109/TKDE.2021.3049250.

Zhang, X., Li, Y., Kotagiri, R., Wu, L., Tari, Z., 2017. KRNN: k Rare-class Nearest Neighbour classification. Pattern Recogn. 62, 33–44. https://doi.org/10.1016/j.patcog.2016.08.023.

Zhong, X.F., Guo, S.Z., Gao, L., Shan, H., Zheng, J.H., 2017. An Improved k-NN Classification with Dynamic k. In: The 9th International Conference on Machine Learning and Computing, pp. 211–216. https://doi.org/10.1145/3055635.3056604.

Zhu, Z., Wang, Z., Li, D., Du, W., 2020. NearCount: Selecting critical instances based on the cited counts of nearest neighbors. Knowl.-Based Syst. 190,. https://doi.org/10.1016/j.knosys.2019.105196. DOI: 10.1016/j.knosys.2019.105196 105196.