A Weighted Feature C-means Clustering Algorithm for Case Indexing and Retrieval in Cased-Based Reasoning

Chuang-Cheng Chiu and Chieh-Yuan Tsai*

Industrial Engineering and Management Department, Yuan-Ze University, Taiwan, R.O.C.
*cytsai@saturn.yzu.edu.tw

Abstract. A successful Case-Based Reasoning (CBR) system highly depends on how to design an accurate and efficient case retrieval mechanism. In this research we propose a Weighted Feature C-means clustering algorithm (WF-C-means) to group all prior cases in the case base into several clusters. In WF-C-means, the weight of each feature is automatically adjusted based on the importance of the feature to clustering quality. After executing WF-C-means, the dissimilarity definition adopted by K-Nearest Neighbor (KNN) search method to retrieve similar prior cases for a new case becomes refined and objective because the weights of all features adjusted by WF-C-means can be involved in the dissimilarity definition. On the other hand, based on the clustering result of WF-C-means, this research proposes a cluster-based case indexing scheme and its corresponding case retrieval strategy to help KNN retrieving the similar prior cases efficiently. Through our experiments, the efforts of this research are useful for real world CBR systems.

Keywords: Case-based reasoning, K-nearest neighbor search, Case indexing scheme, C-means clustering algorithm, Feature weighting.

1 Introduction

Case-based reasoning (CBR) is a problem-solving methodology commonly seen in artificial intelligence [1]. It has been successfully applied to various industrial applications such as sales operation, quality management, product development, health diagnosis, and so on [2], [3], [4], [5], [6]. Similar to human reasoning, a CBR system uses prior cases to find out suitable solutions for a new case. A CBR system stores prior cases in a data repository called a case-base. Each prior case in the case-base consists of the problem description part and solution part. For a new case, its problem description part is inputted into a CBR system. Prior cases similar to the new case are retrieved by evaluating the dissimilarity between the new case and each prior case in terms of their problem description parts. Then, the solution part of the new case is reasoned from the solution parts of these similar past cases.

A successful CBR system highly depends on how to design an effective and efficient case retrieval mechanism. K-Nearest Neighbor (KNN) search method has been extensively used in the case retrieval phase of CBR [7]. Traditional KNN method adopts an exhaustive search strategy to scan the overall case-base, and then select K prior cases which have the minimum dissimilarities with the new case from the case-base. However, most previous studies related to the KNN method simply assumed all features in the problem description part of a case are equally important when evaluating the dissimilarity between two cases in terms of their problem description parts [7], [8]. This makes the case retrieval result tend to be biased and undesired, especially when the number of features is large. In addition, the number of prior cases stored in the case-base increases progressively as time goes by. How to incorporate a case indexing scheme to improve search efficiency, therefore, becomes a great challenge for the KNN method [9], [10], [11], [12].

The basic prerequisite of performing a CBR system is that two cases with similar problem description parts should reveal similar solution parts. Thus, it is meaningful and reasonable to use the solution parts of similar prior cases to reason the solution part of a new case. It implies that all prior cases in the case-base can be partitioned into several clusters based on their problem description parts. The prior cases in the same cluster are similar to each other and are different from cases in other clusters.

Based on this concept, this research proposes a Weighted Feature C-means clustering algorithm (WF-C-means) and a cluster-based case indexing scheme to conquer ineffective and inefficient problems occurred in KNN. In WF-C-means, the weight of each feature is automatically adjusted based on the importance of the feature to the clustering quality. Then, a clustering procedure similar to C-means [13], a classical clustering algorithm, is conducted using weighted features to partition all prior cases into several clusters. After executing WF-C-means, the dissimilarity definition adopted by KNN to retrieve similar prior cases for a new case becomes refined and objective because the weights of all features adjusted by WF-C-means can be involved in the dissimilarity definition. On the other hand, based on the clustering result of WF-C-means, this research proposes a cluster-based case indexing scheme and its corresponding case retrieval strategy to help KNN retrieving the similar prior cases efficiently.

2 A Weighted Feature C-means clustering algorithm

The development of the proposed Weighted Feature C-means clustering algorithm (WF-C-means) is derived from the C-means clustering algorithm [13]. WF-C-means adopts three main procedures to partition all objects into *C* clusters based on the objective of minimizing the sum of dissimilarities between all objects and their corresponding cluster centers. The number of clusters, *C*, is generally pre-assigned by users. The first procedure is to assign each object to one of the *C* clusters properly. The second procedure is to update the *C* cluster centers based on the assignments in the first procedure. In the third procedure, the features that are important for the clustering result are given high weights by dimming the weights of trivial features. The three procedures are iterated until all *C* cluster centers remain the same without being changed.

Let a case-base $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_m, \cdots, \mathbf{x}_M\}$ include *M* prior cases and a feature set $\mathbf{F} = \{\mathbf{f}_1, \cdots, \mathbf{f}_n, \cdots, \mathbf{f}_N\}$ comprise *N* features in the problem description part of a case. A prior case $\mathbf{x}_m = (x_{m1}, \cdots, x_{mn}, \cdots, x_{mN})$ is composed of *N* feature values where x_{mn} is the feature value of $\mathbf{f}_n$ in $\mathbf{x}_m$. Let $\mathbf{C} = \{\mathbf{C}_1, \cdots, \mathbf{C}_i, \cdots, \mathbf{C}_C\}$ be a set of the *C* clusters and $\mathbf{A} = \{\mathbf{a}_1, \cdots, \mathbf{a}_i, \cdots, \mathbf{a}_C\}$ be the set of the *C* cluster centers where $\mathbf{a}_i = (a_{i1}, \cdots, a_{in}, \cdots, a_{iN})$ is the cluster center of the *i*th cluster $\mathbf{C}_i$ and a_{in} is the feature value of $\mathbf{f}_n$ in $\mathbf{a}_i$. Accordingly, the dissimilarity between a prior case $\mathbf{x}_m$ and a cluster center $\mathbf{a}_i$, termed as $diss(\mathbf{x}_m, \mathbf{a}_i)$, can be defined as:

$$diss(\mathbf{x}_m, \mathbf{a}_i) = \sum_{n=1}^{N} w_n \times (x_{mn} - a_{in})^2 = \sum_{n=1}^{N} w_n \times d(x_{mn}, a_{in}) \qquad \mathbf{(1)}$$

where $w_n \in \mathbf{w}$ is the feature weight of the feature $\mathbf{f}_n$ and $\mathbf{w} = \{w_1, \cdots, w_n, \cdots, w_N\}$ is the set of the N feature weights, $\sum_{n=1}^{N} w_n = 1$, $0 \leq w_n \leq 1$. In addition, $d(x_{mn}, a_{in})$ is the difference between $\mathbf{x}_m$ and $\mathbf{a}_i$ in terms of feature $\mathbf{f}_n$. The smaller the value of $diss(\mathbf{x}_m, \mathbf{a}_i)$, the higher the probability that $\mathbf{x}_m$ belongs to cluster $\mathbf{C}_i$.

The objective of the WF-C-means algorithm, equivalent to C-means, is to minimize the sum of the dissimilarities between all prior cases to their corresponding cluster centers, which can be expressed as follows:

$$Minimize \;\; S(\mathbf{U}, \mathbf{A}, \mathbf{w}) = \sum_{m=1}^{M} \sum_{i=1}^{C} u_{mi} \times diss(\mathbf{x}_m, \mathbf{a}_i) = \sum_{m=1}^{M} \sum_{i=1}^{C} \sum_{n=1}^{N} u_{mi} \times w_n \times d(x_{mn}, a_{in}) \qquad \mathbf{(2)}$$

subject to

$$\begin{cases} \sum_{i=1}^{C} u_{mi} = 1 \\ u_{mi} \in \{1, 0\} \qquad \text{for } 1 \leq m \leq M,\ 1 \leq i \leq C,\ 1 \leq n \leq N \\ \sum_{n=1}^{N} w_n = 1 \\ \quad w_n \geq 0 \end{cases} \qquad \mathbf{(3)}$$

where $\mathbf{U}$ is a matrix of size $M \times C$ that records the case-cluster memberships and $u_{mi} \in \{1, 0\}$ is an element in $\mathbf{U}$ that represents the membership of $\mathbf{x}_m$ with the ith cluster $\mathbf{C}_i$. If $u_{mi} = 1$, $\mathbf{x}_m$ belongs to $\mathbf{C}_i$. If $u_{mi} = 0$, by contrast, $\mathbf{x}_m$ does not belong to $\mathbf{C}_i$. The WF-C-means algorithm solves the described optimization problem by iteratively solving the following three reduced problems:

1. Problem P_1: Fix $\mathbf{A} = \hat{\mathbf{A}}$ and $\mathbf{w} = \hat{\mathbf{w}}$ to solve the reduced problem $S(\mathbf{U}, \hat{\mathbf{A}}, \hat{\mathbf{w}})$.
2. Problem P_2: Fix $\mathbf{U} = \hat{\mathbf{U}}$ and $\mathbf{w} = \hat{\mathbf{w}}$ to solve the reduced problem $S(\hat{\mathbf{U}}, \mathbf{A}, \hat{\mathbf{w}})$.
3. Problem P_3: Fix $\mathbf{A} = \hat{\mathbf{A}}$ and $\mathbf{U} = \hat{\mathbf{U}}$ to solve the reduced problem $S(\hat{\mathbf{U}}, \hat{\mathbf{A}}, \mathbf{w})$.

The purpose of solving P_1 is to assign a prior case to a cluster whose cluster center is closest to the prior case, defined as Equation (4):

$$\begin{cases} u_{mi} = 1, \text{ if } \sum_{n=1}^{N} w_n \times d(x_{mn}, a_{in}) \leq \sum_{n=1}^{N} w_n \times d(x_{mn}, a_{jn}) \\ u_{mi} = 0, \text{Otherwise} \end{cases} \quad \text{for } 1 \leq i, j \leq C,\ j \neq i \qquad \mathbf{(4)}$$

Accordingly, the procedure for solving P_2 is considered as a cluster-center updating procedure. Equation (5) can be used to calculate the solution of P_2.

$$a_{in} = \sum_{m=1}^{M} u_{mi} \times x_{mn} \Big/ \sum_{m=1}^{M} u_{mi} \qquad \text{for } 1 \leq i \leq C,\ 1 \leq n \leq N \qquad \mathbf{(5)}$$

The difference between WF-C-Means and C-means is that WF-C-Means needs further solving the weight adjusting problem P_3 but C-means does not. The weight of a feature can be evaluated based on how the feature affects the quality of clustering result. In this research, the clustering quality is defined as the degree of minimizing the separations within clusters and maximizing the separations between clusters. Therefore, the problem P_3 is redefined as:

$$Maximize\ V(\hat{\mathbf{U}}, \hat{\mathbf{A}}, \mathbf{w}, \hat{g}) = \frac{S^{'}(\hat{\mathbf{A}}, \mathbf{w}, \hat{g})}{S(\hat{\mathbf{U}}, \hat{\mathbf{A}}, \mathbf{w})} = \frac{\sum_{n=1}^{N}\left[w_n \times \left(\sum_{i=1}^{C} \| \mathbf{C}_i \| \times d(a_{in}, g_n)\right)\right]}{\sum_{n=1}^{N}\left[w_n \times \left(\sum_{m=1}^{M}\sum_{i=1}^{C} u_{mi} \times d(x_{mn}, a_{in})\right)\right]} \quad \textbf{(6)}$$

subject to

$$\begin{cases} \sum_{n=1}^{N} w_n = 1 & \text{for } 1 \le n \le N \\ w_n \ge 0 \end{cases} \quad \textbf{(7)}$$

where $S(\hat{\mathbf{U}}, \hat{\mathbf{A}}, \mathbf{w})$ and $S^{'}(\hat{\mathbf{A}}, \mathbf{w}, g)$ represent the sum of the separations within clusters and the sum of the separations between clusters, respectively. In addition, $g = (g_1, \cdots, g_n, \cdots, g_N)$ is the global center of all M prior cases in the case-base **X**, and g_n is the feature value of g in terms of $\mathbf{f}_n$ which can be obtained using $g_n = \sum_{m=1}^{M} x_{mn} \big/ M$. In addition, $\| \mathbf{C}_i \|$ represents the number of prior cases in the *i*th cluster $\mathbf{C}_i$ such that $\sum_{i=1}^{C} \| \mathbf{C}_i \| = M$.

Let $e_n = \sum_{m=1}^{M} \sum_{i=1}^{C} u_{mi} \times d(x_{mn}, a_{in})$ be the sum of separations within clusters in terms of $\mathbf{f}_n$ and $f_n = \sum_{i=1}^{C} \| \mathbf{C}_i \| \times d(a_{in}, g_n)$ be the sum of separations between clusters in terms of $\mathbf{f}_n$. Accordingly, Equation (6) can be simplified as:

$$Maximize\ V(\hat{\mathbf{U}}, \hat{\mathbf{A}}, \mathbf{w}, \hat{g}) = \frac{\sum_{n=1}^{N} w_n \times f_n}{\sum_{n=1}^{N} w_n \times e_n} \quad \textbf{(8)}$$

This research proposes a feature weight adjusting rule to derive **w** from Equation (8). Let $\mathbf{w}^{\text{new}} = \{w_1^{\text{new}}, \cdots, w_n^{\text{new}}, \cdots, w_N^{\text{new}}\}$ be the set of the new weights for original N feature weights in **w** after the adjustment. For each feature $\mathbf{f}_n$, its new weight w_n^{new} can be evaluated by adding an adjustment margin Δw_n to its origin weight w_n. By applying a common decision optimization method in linear programming theory [14], Δw_n can be derived as: $\Delta w_n = (f_n / e_n) \big/ \sum_{n=1}^{N} (f_n / e_n)$. Therefore, the new weight w_n^{new} of $\mathbf{f}_n$ can be calculated using Equation (9).

$$w_n^{\text{new}} = w_n + \Delta w_n = w_n + \frac{f_n / e_n}{\sum_{n=1}^{N} (f_n / e_n)} \quad \text{for } 1 \le n \le N \quad \textbf{(9)}$$

Notes that the new weights derived by Equation (9) might violate the constraint of $\sum_{n=1}^{N} w_n^{\text{new}} = 1$. Therefore, Equation (9) is further modified as Equation (10) through the normalization process.

$$w_n^{\text{new}} = \frac{w_n + \dfrac{f_n/e_n}{\sum_{n=1}^{N}(f_n/e_n)}}{\sum_{n=1}^{N} w_n + \sum_{n=1}^{N}\left(\dfrac{f_n/e_n}{\sum_{n=1}^{N}(f_n/e_n)}\right)} \quad \text{for } 1 \le n \le N \qquad \mathbf{(10)}$$

In WF-C-means, the procedures of prior case assignment, cluster center update, and feature weight adjustment are executed iteratively until all *C* cluster centers remain the same without being changed. The pseudo-code of the WF-C-means algorithm is summarized in Fig. 1. After executing WF-C-means, the dissimilarity definition adopted by KNN for case retrieval, defined as Equation (1), becomes refined and objective because the weights of all features can be involved in the dissimilarity definition. Therefore, it prevents KNN from obtaining the unbiased and undesired retrieval results.

```
Input: a set of $M$ prior cases in which each prior case has $N$ features in its problem description part; the number of clusters, $C$.
Select $C$ cluster centers randomly for these $C$ clusters.
Let the feature weight of each feature be ($1/N$).
Repeat
  Form $C$ clusters by assigning each prior case to its closest cluster center using Equation (4).
  Update the cluster center in each cluster using Equation (5).
  Adjust the feature weight of each feature using Equation (10).
Until all $C$ cluster centers are not changed.
```

Fig. 1. The pseudo-code of the WF-C-means algorithm.

3 Case indexing and retrieval based on WF-C-means

As stated, prior cases in the same cluster have similar problem description part. In other words, in a certain cluster if one prior case is similar to the new case, other prior cases in the same cluster might be similar to the new case under certain degree. Based on this concept, this research proposes a cluster-based case indexing scheme from the clustering result generated by WF-C-means. With this proposed case indexing scheme, KNN can adopt a novel case retrieval strategy to search similar prior cases for a new case instead of its original exhaustive search strategy. Not only the efficiency of KNN can be improved, but also the final retrieval result is the same with the one using the exhaustive search strategy. The proposed cluster-based case indexing scheme is introduced as follows:

1. For each cluster $\mathbf{C}_i$, its cluster center $\mathbf{a}_i$ is considered as a representative case for all prior cases in $\mathbf{C}_i$. Let $y_i = \|\mathbf{C}_i\|$ be that the number of prior cases belongs to $\mathbf{C}_i$ such that $\sum_{i=1}^{C} y_i = M$.

2. For each cluster $\mathbf{C}_i$, all y_i prior cases in $\mathbf{C}_i$ are assigned index numbers based on the ascending order of their dissimilarities with their cluster center $\mathbf{a}_i$. Therefore, $\mathbf{C}_i$ is reorganized as $\mathbf{C}_i = \{\mathbf{c}_1^i, \mathbf{c}_2^i, \cdots, \mathbf{c}_{t-1}^i, \mathbf{c}_t^i, \cdots, \mathbf{c}_{y_i}^i \mid diss(\mathbf{c}_{t-1}^i, \mathbf{a}_i) \le diss(\mathbf{c}_t^i, \mathbf{a}_i)\}$. Meanwhile, The radius of a cluster $\mathbf{C}_i$, termed as r_i, is defined as $r_i = Maximum\ \{diss(\mathbf{c}_t^i, \mathbf{a}_i) \mid \mathbf{c}_t^i \in \mathbf{C}_i\}$.

Based on the proposed cluster-based case indexing scheme, the corresponding case retrieval strategy adopted by KNN is introduced as follows, and its pseudo-code is illustrated in Fig. 2.

1. Let $\mathbf{Sim_Q}$ be the set stores the prior cases similar to the new case **Q** in terms of their problem description part. At the beginning of case retrieval, $\mathbf{Sim_Q}$ is an empty set. During the processing of case retrieval, if the dissimilarity between a prior case and **Q** is less than a user-defined dissimilarity threshold, termed as *r*, the prior case is added into $\mathbf{Sim_Q}$.
2. For each cluster $\mathbf{C}_i$, we first calculate the dissimilarity $diss(\mathbf{a}_i, \mathbf{Q})$ between its cluster center $\mathbf{a}_i$ and the new case **Q**.
3. If $diss(\mathbf{a}_i, \mathbf{Q}) \ge r_i + r$, all prior cases in $\mathbf{C}_i$ are not similar enough to **Q**. Therefore, no further dissimilarity calculation is required for any prior case in $\mathbf{C}_i$ with **Q**. That is, no prior case in $\mathbf{C}_i$ can be added into $\mathbf{Sim_Q}$.
4. If $diss(\mathbf{a}_i, \mathbf{Q}) < r_i + r$, a further dissimilarity checking procedure is then conducted for all prior cases in $\mathbf{C}_i$. Any prior case $\mathbf{c}_t^i$ in $\mathbf{C}_i$ that satisfies the constraint: $diss(\mathbf{a}_i, \mathbf{Q}) - r < diss(\mathbf{a}_i, \mathbf{c}_t^i) < diss(\mathbf{a}_i, \mathbf{Q}) + r$ can be considered as a *candidate* for $\mathbf{Sim_Q}$. The candidates can be found easily using a binary search since all prior cases in $\mathbf{C}_i$ have been indexed based on their dissimilarities from their cluster center $\mathbf{a}_i$ in ascending order in the proposed case indexing scheme.
5. For each candidate case in $\mathbf{C}_i$, we calculate its dissimilarity with **Q**. If the dissimilarity is less than *r*, it becomes a formal member of $\mathbf{Sim_Q}$.
6. After adding similar prior cases of all *C* clusters into $\mathbf{Sim_Q}$, if the number of cases in $\mathbf{Sim_Q}$ is larger than or equal to *K*, the first *K* prior cases with the minimum dissimilarities with **Q** are outputted as the retrieval result. By contrary, if the number of cases in $\mathbf{Sim_Q}$ is less than *K*, the dissimilarity threshold *r* is widened by adding a increasing margin Δr, i.e., $r = r + \Delta r$. With the new threshold, the retrieval proceeding goes back to execute step 3 until all *K* similar prior cases has been retrieved.

Instead of starting from scratch, the dissimilarity threshold *r* can be initialized using Equation (11). Equation (11) is formulated based on the following concept. For each cluster $\mathbf{C}_i$, if $y_i \ge K$, we can easily identify the *K*th prior case $\mathbf{c}_K^i$ based on its index number. Therefore, the dissimilarity $diss(\mathbf{a}_i, \mathbf{c}_K^i)$ can be considered as a candidate dissimilarity threshold for $\mathbf{C}_i$. However, if $y_i < K$, the radius r_i of $\mathbf{C}_i$ is considered as the candidate dissimilarity threshold for $\mathbf{C}_i$. The maximum among these specific dissimilarity thresholds, defined as Equation (11), is therefore suggested as a reasonable value of r. In addition, the increasing margin Δr, as stated in step 6 of the case retrieval strategy, is also suggested as the value of r for convenient reason.

$$r = Maximum\ \{diss(\mathbf{a}_i, \mathbf{c}^i_K) \mid \mathbf{c}^i_K \in \mathbf{C}_i, 1 \le i \le C\} \qquad \textbf{(11)}$$

```
Input: a new case Q given its problem description part; a user-defined dissimilarity threshold $r$
Output: The first $K$ prior cases which have minimum dissimilarities with Q in a case set $\mathbf{Sim_Q}$
Let the set $\mathbf{Sim_Q}$ be an empty set.
Repeat
  For each cluster $\mathbf{C}_i$ in the cluster-based case indexing scheme {
    Calculate the dissimilarity $diss(\mathbf{a}_i,\mathbf{Q})$ between the cluster center $\mathbf{a}_i$ and the new case Q
    If $diss(\mathbf{a}_i,\mathbf{Q}) < r_i + r$ {    // $r_i$ is the radius of the cluster $\mathbf{C}_i$
      For each prior case $\mathbf{c}_i^l$ in $\mathbf{C}_i$ {
        If $diss(\mathbf{a}_i,\mathbf{Q}) - r < diss(\mathbf{a}_i,\mathbf{c}_i^l) < diss(\mathbf{a}_i,\mathbf{Q}) + r$ {
          Calculate the dissimilarity $diss(\mathbf{c}_i^l,\mathbf{Q})$ between $\mathbf{c}_i^l$ and the new case Q
          If $diss(\mathbf{c}_i^l,\mathbf{Q}) < r$ {
            Add $\mathbf{c}_i^l$ into $\mathbf{Sim_Q}$
          }
        }
      }
    }
  }
  If the number of cases in $\mathbf{Sim_Q} < K$ {
    $r := r +$
  }
Until the number of cases in $\mathbf{Sim_Q}$ is larger than or equal to $K$.
Return the first $K$ prior cases which have minimum dissimilarities with Q in $\mathbf{Sim_Q}$
```

Fig. 2. The pseudo-code of retrieving K prior cases similar to a new case.

4 Experiments

4.1 Clustering performance of WF-C-means

A series of experiments on three popular datasets [15] are made to demonstrate the performance of the proposed WF-C-means algorithm. The properties of the three datasets are shown in Table 1. Because the scales of all features in the wine dataset are different, all features in the wine dataset are standardized. This research adopts the Sum of Square within-cluster Error (SSE), defined as Equation (2), as the measure to judge the clustering accuracy for the traditional C-means and proposed WF-C-means algorithms. The less the SSE value, the more the clustering accuracy is.

The experiments for the same dataset are conducted 100 replicates. In each replicate, the two algorithms use equivalent initial cluster centers that are generated randomly, while the weight values for all N features are initially assigned as $1/N$. The number of clusters, C, is specified as the default class number of each dataset. The clustering performances of the two algorithms using the three datasets are shown in

Table 2. Notes that the results are generated through averaging the experimental results of the 100 replicates.

As shown in Table 2, the clustering accuracy of the proposed WF-C-means algorithm is obviously superior to the accuracy of C-means in terms of SSE measures. For the iris and wine datasets, the number of iterations in WF-C-means is nearly the same with the ones in C-means. When the number of cases increases such as the yeast dataset, WF-C-means needs more iterations to converge the optimal solution. The computational efficiency of WF-C-means, therefore, is slightly inferior to C-means. However, the tradeoff between them is worth based on our study.

Table 1. The properties of the three real world datasets.

Dataset	Number of instances (cases)	Number of features	Number of classes
iris	150	4	3
wine	178	13	3
yeast	1484	8	10

Table 2. The clustering performances for the three datasets using the two algorithms.

Algorithm	C-means			WF-C-means		
Dataset	iris	wine	yeast	Iris	wine	yeast
SSE	23.513	98.688	6.213	22.251	66.044	3.372
Number of iterations	7.370	7.367	31.077	7.223	8.943	38.562

Through WF-C-means, the final weights of the four features in the iris dataset are $\{w_1, w_2, w_3, w_4\} = \{0.09, 0.04, 0.53, 0.34\}$. It is clear that the third and fourth features are more important than other two features. When substituting these weights into Equation (1), an objective dissimilarity between two cases $\mathbf{x}_m$ and $\mathbf{x}_p$ is calculated by $0.09(x_{m1}-x_{p1})^2+0.04(x_{m2}-x_{p2})^2+0.53(x_{m3}-x_{p3})^2+0.34(x_{m4}-x_{p4})^2$. Similarly, the top four features with the highest weights in the wine datasets are w_{12}, w_{13}, w_7, and w_6. Their weights are $\{w_{12}, w_{13}, w_7, w_6\} = \{0.24, 0.19, 0.12, 0.10\}$. Similarly, the top four features with the highest weights in the yeast datasets are $\{w_1, w_4, w_2, w_8\}=\{0.60, 0.23, 0.08, 0.05\}$. The important features identified by WF-C-means are almost the same as the result discovered in the paper of [16]. This also verifies that the dissimilarity definition with weighted features is reliable and can contribute accurate case retrieval result.

4.2 Performance of case retrieval based on our case indexing scheme

The traditional KNN method adopts an exhaustive search strategy to evaluate the dissimilarity between each prior case in the case-base and the new case. It requires huge computation time on dissimilarity calculations, especially the number of prior cases in the case-base is considerable. The computational complexity of the exhaustive search strategy is O(M) if the dissimilarity calculation time between two cases is constant.

In opposition, when KNN adopts the proposed case retrieval strategy with the proposed case indexing scheme, its computational complexity equals to O(C+pM) where p is the probability of the event that $diss(\mathbf{a}_i,\mathbf{Q}) < r_i+r$ in the case retrieval strategy and $0 \le p \le 1$. If C is close to 1 (all cases are located within a cluster), the radius of the cluster is large enough to cover all cases. As a result, p tends to be 1 for the cluster. It makes the computational complexity close to O(M). If C is close to M (each case is considered as an individual cluster), p tends to be 0 since the radius of each cluster is close to 0, so the computational complexity is also close to O(M). However, in practice, C<<M so that the computational complexity of the proposed case retrieval strategy can be simplified as O(pM). It is much less than the computational complexity using the exhaustive search strategy.

Let us take the yeast dataset as an example to show the search efficiency using the proposed case retrieval strategy with the proposed case indexing scheme. In this experiment, the leave-one-out validation approach is adopted. That is, each case in the dataset sequentially serves as a new case, while other 1483 cases are considered as the prior cases in the case-base. For each new case, we count the number of dissimilarity calculations required to retrieve K similar prior cases where K is set as 5 in all experiments. Fig. 3 depicts the search efficiency of KNN using the proposed case retrieval strategy. In the figure, the number of required dissimilarity calculations through our method is significantly less than the number using the exhaustive search strategy. The calculation numbers form a concave quadratic shape. When C is between 8 and 128, our method can decrease at least 70% similarity calculations compared to the exhaustive search strategy. From the experiments, we can know that the search efficiency of KNN becomes obviously increased when using the proposed cluster-based case indexing scheme and corresponding case retrieval strategy instead of the exhaustive search strategy.

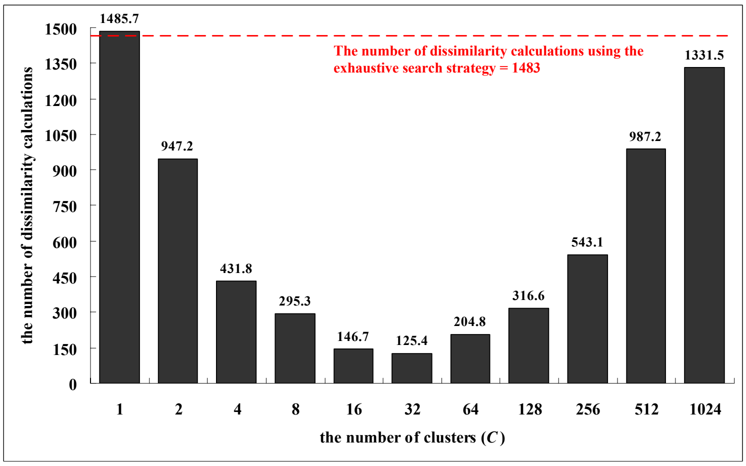

Fig. 3. The efficiency of KNN using the proposed case indexing and retrieval method for the yeast dataset (K=5).

5 Conclusions

This research aims at improving the accuracy and efficiency of the case retrieval phase in a CBR system. A WF-C-means algorithm is proposed to group all prior cases in the case-base into several clusters. Based on the clustering result of WF-C-means, an objective dissimilarity definition can be obtained from the adjusted feature weights. KNN can use the dissimilarity definition to effectively retrieve similar prior cases for a new case. Based on the clustering result, furthermore, a cluster-based case indexing scheme and its corresponding case retrieval strategy are also proposed in order to increase the search efficiency of KNN in the case retrieval phase. Through our experiments, the efforts of this research are useful for real world CBR systems.

In our experiments, we observed that some settings for the cluster number in WF-C-means might increase search efficiency in the case retrieval phase. In the future, we will study how to determine an appropriate number of clusters before performing WF-C-Means so that the final accuracy and efficiency in the case retrieval phase can be further improved.

References

1. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues Methodological Variations, and System Approaches. Artificial Intelligence Communications 7 (1994) 39–59
2. Gardingen, D., Watson, I.: A Web Based CBR System for Heating Ventilation and Air Conditioning System Sales Support. Knowledge Based System 12 (1999) 207–214
3. Suh, M.S., Jhee, W.C., Ko, Y.K., Lee, A.: A Cased-Based Expert System Approach for Quality Design. Expert Systems with Applications 15 (1998) 181–190
4. Belecheanu, R., Pawar, K.S., Barson, R.J., Bredehorst, B., Weber, F.: The Application of Case Based Reasoning to Decision Support in New Product Development. Integrated Manufacturing Systems, 14 (2003) 36–45
5. Li, L.L.X.: Knowledge-Based Problem Solving: An Approach to Health Assessment. Expert Systems with Applications 16 (1999) 33–42
6. Tsai, C.Y., Chiu, C.C., Chen, J.S.: A Cased-Based Reasoning System for PCB Defect Prediction. Expert Systems with Applications 28 (2005) 813–822
7. Watson, I., Marir, F.: Case-Based Reasoning: A Review. The Knowledge Engineering Review 9 (1994) 355–381
8. Kolodner, J. L.: Case-Based Reasoning. Morgan Kaufmann, San Francisco, California (1993)
9. Guttman, A.: R-trees: A Dynamic Index Structure for Spatial Searching, Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data (1984) 47–57
10. Fukunaga, K., Narendra, P.M.: A Branch and Bound Algorithm for Computing K-Nearest Neighbors. IEEE Transaction on Computers 24 (1975) 750–753
11. Gargantini, I.: An Effective Way to Represent Quadtrees. Communications of the ACM 25 (1982) 905–910
12. Yianilos, P.N.: Data Structures and Algorithms for Nearest Neighbor Search in General Metric Spaces. Proceedings of the 4th annual ACM-SIAM symposium on Discrete algorithms (1993) 311–321
13. McQueen, J.: Some Methods for Classification and Analysis of Multivariate Observations. Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability (1967) 281–297
14. Hillier, F.S., Lieberman, G.J.: Introduction to Operation Research. McGraw-Hill, New York (2001)
15. Newman, D.J., Hettich, S., Blake, C.L., Merz C.J.: UCI Repository of Machine Learning Databases. http://www.ics.uci.edu/~mlearn/MLSummary.html (1998)
16. Ahmad, A., Dey, L.: A Feature Selection Technique for Classificatory Analysis. Pattern Recognition Letters 26 (2005) 43–56