# A Case Retrieval Approach Using Similarity and Association Knowledge

Yong-Bin Kang[1], Shonali Krishnaswamy[1], and Arkady Zaslavsky[1,2]

[1] Faculty of IT, Monash University, Australia
{yongbin.kang,shonali.krishnaswamy}@monash.edu
[2] ICT Centre, CSIRO, Australia
arkady.zaslavsky@csiro.au

**Abstract.** Retrieval is often considered the most important phase in Case-Based Reasoning (CBR), since it lays the foundation for overall performance of CBR systems. Retrieval in CBR aims to retrieve relevant cases that can be successfully used for solving a new problem. To realize retrieval, CBR systems typically rely on a strategy that exploits similarity knowledge, and it is called similarity-based retrieval (SBR). In SBR, similarity knowledge approximates the usefulness of cases for solving a new problem. In this paper, we show that *association analysis* of stored cases can be used to strengthen SBR. We present a new approach for extracting and representing *association knowledge* from the cases using association rule mining. We propose a novel retrieval strategy USIMSCAR that qualitatively enhances SBR by leveraging both similarity and association knowledge. We demonstrate the significant advantages of using USIMSCAR over SBR through an experimental evaluation using medical datasets.

## 1 Introduction

Case-Based Reasoning (CBR) [1] is a widely researched technology for problem-solving in many application domains such as medical diagnosis [2], help-desk service [3], product recommendation [4], and classification [5]. The fundamental premise of CBR is that experience in the form of past cases can be leveraged to solve new problems. It is based on the fact that in many application domains, similar problems usually have similar solutions. In CBR, experiences are stored in a database known as a *case base*, and an individual experience is called a *case*.

Typically, there are four well-organized phases adopted in CBR [1]. The first phase is to *retrieve* one or several cases considered useful for solving a given target problem. Once useful cases are retrieved, the second phase is to *reuse* their solution information. The third phase is to *revise* or *adapt* the solution information to better fit the target problem if necessary. The fourth phase is to *retain* the new solution once it has been confirmed or validated.

Retrieval is often considered the most important phase in CBR, since it lays the foundation for overall performance of CBR systems [6]. Its aim is to retrieve *useful* cases that can be successfully used to solve a new problem. If retrieved

R. Meersman, T. Dillon, and P. Herrero (Eds.): OTM 2011, Part I, LNCS 7044, pp. 218–235, 2011.

cases are not useful, CBR systems will not eventually produce any good solution for the new problem.

To accomplish the retrieval process, CBR systems typically rely on a retrieval strategy that exploits *similarity knowledge*. This strategy is often called *similarity-based retrieval* (SBR) [7]. In SBR, similarity knowledge aims to approximate the *usefulness* of stored cases as to solving a new problem [8]. This knowledge is usually encoded in the form of *similarity measures*, which are used to compute similarities between a new problem and the cases. By using similarity measures, SBR retrieves useful cases ranked by their similarities to the new problem. The solutions of these cases are then utilized to solve the problem. A limitation of SBR is that it tends to rely strongly on similarity knowledge only, ignoring other available knowledge that can be additionally leveraged for improving its retrieval performance [7,9,8].

While many kinds of learnt and induced knowledge (e.g. statistical [10], domain [8,11], adaptation [7,12] knowledge) have been utilized to enhance SBR, this paper proposes that association analysis of stored cases can improve traditional SBR. We propose a new retrieval strategy USIMSCAR that leverages *association knowledge* in conjunction with similarity knowledge. Association knowledge is formalized to represent certain interesting relationships, shared by a large number of cases, acquired from stored cases using *association rule mining*. The key idea of USIMSCAR thus lies in its usage of both similarity and association knowledge to deliver an improved retrieval strategy for CBR. We show USIMSCAR improves SBR through an experimental evaluation using medical datasets found in UCI ML Repository.

This paper is organized as follows. In Section 2, we present the motivation of our work. In Section 3, we present a background of similarity knowledge and association knowledge. In Section 4, we present our approach for formalizing association knowledge. In Section 5, we present USIMSCAR that leverages similarity and association knowledge. In Section 6, we evaluate USIMSCAR in comparison with SBR. In Section 7, we review the literature work related to this paper. In Section 8, we present our conclusion and future research directions.

## 2 Motivating Scenario

To illustrate our research motivation, we use a medical diagnosis scenario presented in [13]. Consider a case base $\mathcal{D}$ in Table 1, where each case is represented as a pair of a problem and the corresponding solution. Each problem is described by five attributes (symptoms) $A_1, ..., A_5$, and each solution by an attribute (a diagnosis) $A_6$. Our aim is to diagnose the correct disease 'appendicitis' for a new patient $Q$, since $Q$ really suffered from 'appendicitis' as noted in [13].

To predict a diagnosis for $Q$, in principle, SBR may find similar cases to $Q$ using a similarity metric. Assume that we use the following metric, used in [13], that measures the similarity between $Q$ and each case $P \in \mathcal{D}$,

**Table 1.** A patient case base

| Cases ID | Local Pain ($A_1$) | Other Pain ($A_2$) | Fever ($A_3$) | Appetite Loss ($A_4$) | Age ($A_5$) | Diagnosis ($A_6$) | Similarity to $Q$ |
|---|---|---|---|---|---|---|---|
| $P_1$ | right flank | vomit | 38.6 | yes | 10 | appendicitis | 0.631 |
| $P_2$ | right flank | vomit | 38.7 | yes | 11 | appendicitis | 0.623 |
| $P_3$ | right flank | vomit | 38.8 | yes | 13 | appendicitis | 0.618 |
| $P_4$ | right flank | sickness | 37.5 | yes | 35 | gastritis | **0.637** |
| $P_5$ | epigastrium | nausea | 36.8 | no | 20 | stitch | 0.420 |
| $Q$ | right flank | nausea | 37.8 | yes | 14 | ? | |
| Weight | 0.91 | 0.78 | 0.60 | 0.40 | 0.20 | | |

$$SIM(Q,P) = \frac{\sum_{i=1}^{n} w_i \cdot sim(q_i, p_i)}{\sum_{i=1}^{n} w_i},$$

$$sim(q_i, p_i) = \begin{cases} 1 - \frac{|q_i - p_i|}{A_i^{\max} - A_i^{\min}}, & \text{if } A_i \text{ is numeric,} \\ 1, & \text{if } A_i \text{ is discrete \& } q_i = p_i, \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where $w_i$ is a weight assigned to an attribute $A_i$ of $Q$ and $P$ by domain experts, $q_i$ and $p_i$ are values of $A_i$, $n$ is the number of attributes of $Q$ and $P$ (i.e. $n = 5$), and $sim(q_i, p_i)$ is a function computing the similarity between $q_i$ and $p_i$.

Once cases similar to $Q$ are selected using the metric $SIM$, SBR determines a diagnosis for $Q$. Assume that SBR utilizes the most similar case to $Q$. As seen in Table 1, $P_4$ is thus chosen, since it is the most similar case to $Q$. This means that a diagnosis choice for $Q$ is 'gastritis'. However, it turns out to be wrong, since $Q$ actually suffered from 'appendicitis', as outlined above. This scenario shows that SBR has a limitation rooted in its nature−its ability tends to be strongly determined by the use of only similarity knowledge.

To address this issue, our idea is to formalize special knowledge, called *association knowledge*, that indicates how certain known problems are strongly associated with certain known solutions in a case base, and to exploit it during the retrieval process in CBR. For example, in Table 1, we observe that the values of $A_5$ (Age) of three cases $P_1$, $P_2$, and $P_3$ are quite similar, which range from 10 to 13. These values are associated with 'appendicitis'. Whereas the value of $A_5$ of $P_4$ is 35, and it is associated with 'gastritis'. We note that the former association is *supported* by three cases, while the latter by only one case. If we quantify such associations, it may be usefully exploited in conjunction with similarity knowledge for solving the target problem. For example, assume that each of the above associations is quantified as the proportion of the cases that support it. The former association ($as_1$) is then quantified as 0.6 (3/5), and the latter ($as_2$) as 0.2 (1/5). In Table 1, we see that $SIM(Q, P_4)$ is 0.637, and $SIM(Q, P_1)$ is 0.631. We now measure the usefulness of each case with respect to $Q$ by combining its similarity to $Q$ and the quantified value of the association that the case supports. Suppose that the combination is implemented via the arithmetic multiplication operation. Then, we measure the usefulness of $P_4$ as 0.127 by $SIM(Q, P_4) \times as_2$, and that of $P_1$ as 0.379 by $SIM(Q, P_1) \times as_1$. Regarding the

computed usefulness, the higher the better. We thus conclude that $P_1$ is more useful than $P_4$ so that $P_1$'s diagnosis 'appendicitis' can be used as a diagnosis for $Q$. This meets our objective of this scenario. This paper presents how to extract and represent association knowledge as well as exploit this knowledge in conjunction with similarity knowledge in order to qualitatively enhance SBR.

## 3 Background to Research on Similarity and Association Knowledge

Prior to presenting the underpinnings of our proposed retrieval strategy, it is essential to provide a background of similarity and association knowledge. This section provides this background. We first present our case representation scheme, which is the basis for formalizing both similarity and association knowledge.

To represent cases, many CBR systems generally adopt well-known knowledge representation formalisms, such as *attribute-value pairs* and *structural* representations [14]. In our work, we choose the attribute-value pairs representation due to its simplicity, flexibility and popularity. Let $A_1, ..., A_m$ be attributes defined in a given domain. An *attribute-value pair* is a pair $(A_i, a_i)$, where $A_i$ is an attribute (or feature[1]) and $a_i$ is a value of $A_{i\in[1,m]}$. A *case* $C$ is a pair $C = (X, Y)$ where $X$ is a problem, represented as $X = \{(A_1, a_1), ..., (A_{m-1}, a_{m-1})\}$, and $Y$ is the solution of $X$, represented as $Y = (A_m, a_m)$. We call an attribute $A_m$ a *solution-attribute*. A *case base* $\mathcal{D}$ is a collection of cases.

### 3.1 Background to Research on Similarity Knowledge

Similarity knowledge is referred to as knowledge encoded via similarity measures computing the similarities between a new problem $Q$ and stored cases. SBR mainly exploits this knowledge. Similarity knowledge represents a heuristic for estimating the usefulness of stored cases as to solving $Q$. Intuitively, the higher the similarity between $Q$ and the case $C$ is, the more useful $C$ is as to solving $Q$. A formulation of similarity measures suitable for cases represented by attribute-value pairs is often based on a widely used principle. This is the *local-global principle* that decomposes a similarity measure by *local similarities* for individual attributes of the cases and a *global similarity* aggregating the local similarities [8]. An accurate local similarity function relies on attribute types. A global similarity function can be arbitrarily complex, but simple functions are usually used. A widely used form is *weighted average aggregation* [8] (e.g. $SIM$ in Eq. 1).

### 3.2 Background to Research on Association Knowledge

We now present the fundamentals of association knowledge in a *CBR context*. These are *association rule mining* [15], *class association rule mining* [16], and the *soft-matching criterion* [17].

[1] To simplify the presentation, we do not distinguish between terms "attributes" and "features", and use these terms interchangeably.

*Association rule mining* [15] aims to mine certain interesting relationships, called *associations*, in a transaction database. It focuses on discovering a set of highly correlated features shared a large number of transactions in the database. Let $I$ be a set of distinct literals, called *items*. A set of items $X \subseteq I$ is called an itemset. Let $\mathcal{D}$ be a set of transactions. Each transaction $T \in \mathcal{D}$ is a set of items such that $T \subseteq I$. We say that $T$ *contains* an itemset $X$, if $X \subseteq T$ holds. Every *association rule* has two parts: an *antecedent* and a *consequent*. An association rule is an implication of the form $X \rightarrow Y$, where $X \subseteq I$ is an itemset in the antecedent and $Y \subseteq I$ is an itemset in the consequent, and $X \cap Y = \emptyset$. The rule $X \rightarrow Y$ has *support* $s$ in $\mathcal{D}$ if $s\%$ of transactions in $\mathcal{D}$ contain $X \cup Y$. This holds in $\mathcal{D}$ with *confidence* $c$ if $c\%$ of transactions in $\mathcal{D}$ that contain $X$ also contain $Y$. Association rule mining can also be used for discovering interesting relationships among stored cases. In a CBR context, a transaction is seen as a case, and an item is seen as an attribute-value pair. Referring to Table 1, we can mine a rule $r_1 : (A_1, \text{right flank}) \rightarrow (A_2, \text{vomit})$. Let $X$ be an item ($A_1$, right flank). Let $Y$ be an item ($A_2$, vomit). The support of $r_1$ is 0.6, since $X$ and $Y$ occur together in three out of five cases in $\mathcal{D}$. The confidence of $r_1$ is 0.75, since $Y$ occurs in three out of four cases that contain $X$ in $\mathcal{D}$. Apriori [15] is one of the traditional algorithms for association rule mining.

Consider a special subset of association rules whose consequents are restricted to a single target variable. Rules in this subset are called *class association rules* (cars) [16]. In a CBR context, cars can be seen as special association rules whose consequents are restricted to hold special items, formed as pairs of a "solution-attribute" (see earlier in this section) and its values. We call such an item a *solution-item*. Thus, a car has the form $X \rightarrow y$, where $X \subseteq I$ an itemset in the antecedent and $y \in I$ is a solution-item in the consequent. In Table 1, we can mine a car: $r_2 : (A_2, \text{vomit}) \rightarrow (A_6, \text{appendicitis})$. But the above rule $r_1$ is not a car, since $r_1$ does not contain any solution-item in the consequent. We here emphasize that the aim of building association knowledge is to formalize special knowledge encoding how certain attribute-value pairs of known problems are *associated* with known solutions in a case base. For the purpose, we will use the form of cars, since it is suited well for it. Note that the car $X \rightarrow y$ encodes an association between an itemset $X$, holding attribute-value pairs of known problems, and a solution-item $y$ holding an attribute-value pair of a known solution.

Consider the association rule $X \rightarrow Y$. A limitation of traditional association rule mining algorithms (e.g. Apriori [15]) is that itemsets $X$ and $Y$ are discovered based on the equality relation. Unfortunately, when dealing with items similar to each other, these algorithms may perform poorly. For example, consider the sales database of a supermarket. Apriori cannot find rules like "80% of the customers who buy products similar to milk (e.g. cheese) and products similar to eggs (e.g, mayonnaise) also buy bread." To address this issue, the SoftApriori algorithm [17] was proposed. It uses the *soft-matching criterion*, where itemsets in the antecedent and the consequent are found using *similarity assessment*. By employing the concept of similarity, the soft-matching criterion can be used to model richer relationships among features of cases than the equality relation [17].

## 4 Association Knowledge Formalization

Association knowledge is encoded via special rules that are "cars" whose antecedents are determined based on the "soft-matching criterion". We call these rules *soft-matching class association rules* (*scars*). A scar represents a strongly evident correlation, between certain attribute-value pairs of known problems and a known solution shared by a significant number of relevant cases.

Let $\mathcal{D}$ be a set of cases, where each case is characterized by attributes $A_1, ..., A_m$. Based on our case representation scheme, presented in Section 3, we call a pair $(A_i, a_i)_{1 \leq i \leq m-1}$ an *item*. We call a pair $(A_m, a_m)$ a *solution-item*. Let $I$ be a set of items. A set $L \subseteq I$ with $k = |L|$ is called a $k$-itemset or simply an itemset. Let $sim(x, y)$ be a function computing the similarity between two items $x, y \in I$. We say that $x$ and $y$ are similar, iff $sim(x, y) \geq$ *a user-specified minimum similarity* (*minsim*). Let $x$ be an item ($A_2$, 38.6). Let $y$ be an item ($A_2$, 38.7). Assuming a similarity function for an attribute $A_2$ is defined as $sim(x, y) = 1 - \frac{|x-y|}{40}$, $sim(x, y)$ is 0.998. Given two itemsets $X, Y \subseteq I$ ($|X| \leq |Y|$), $softSuppR(X, Y)$ is a function defined as $\sum \frac{sim(x,y)}{|X|}$, where $x \in X$ and $y \in Y$ are two items characterized by the *same* attribute. We say that $X$ is a soft-subset of $Y$ ($X \subseteq_{soft} Y$), iff $softSuppR(X, Y) \geq$ minsim; or $Y$ *softly contains* $X$. The problem described in each case $C \in \mathcal{D}$ is also seen as a $k$-itemset with $k = |m - 1|$, since it is represented as $\{(A_1, a_1), ..., (A_{m-1}, a_{m-1})\}$. The *soft-support-sum* of an itemset $X$ regarding $\mathcal{D}$ is defined as $softSuppSum(X) = \sum_{C \in \mathcal{D}} softSuppR(X, C)$, where $X \subseteq_{soft} C$. The *soft-support* of $X$ is defined as $softSupp(X) = \frac{softSuppSum(X)}{|\mathcal{D}|}$. The *soft-support* for a scar $X \rightarrow y$ is defined as the fraction of cases in $\mathcal{D}$ that softly contain an itemset $X$ and contain a solution-item $y$. The *soft-confidence* of this rule is defined as the fraction of cases in $\mathcal{D}$ that softly contain $X$ also contain $y$. In this paper, a *ruleitem* is of the form $\langle X, y \rangle$ and basically represents a scar $X \rightarrow y$.

The key operation for scars mining is to find all ruleitems that have soft-supports $\geq$ *minsupp* (*a user-specified minimum support*). We call such ruleitems *frequent* ruleitems. For all the ruleitems that have the same itemset in the antecedent, one with the highest *interestingness* is chosen as a *possible rule* (PR). To measure the interestingness of association rules, the support and confidence criteria are typically used. On some occasions, a combination of them is used. Often, a rationale for doing so is to define a single optimal interestingness by leveraging their correlations. We choose an interestingness measure that combines soft-support and soft-confidence such that they are monotonically related. We thus choose the Laplace measure [18]. Given a ruleitem $r : X \rightarrow y$, its Laplace measure $Laplace(r)$ can be denoted as $\frac{N \cdot softSupp(X \rightarrow y)+1}{N \cdot softSupp(X \rightarrow y)/softConf(X \rightarrow y)+2}$, where $N = |\mathcal{D}|$. If $Laplace(r) \geq$ *a user-specified minimum level of interesting* (*min-interesting*), we say $r$ is *accurate*. A candidate set of scars consists of all the PRs that are frequent and accurate.

Let $k$-ruleitem be a ruleitem whose antecedent has $k$ items. Let $F_k$ be a set of frequent $k$-ruleitems. In $F_k$, each ruleitem $r$: $X \rightarrow y$ has two fields: $r.anteSoftSuppSum$ storing soft-support-sum of ruleitems in $\mathcal{D}$ that softly

```
 1: F_1 = findFrequentRuleItems(D);
 2: SCAR_1 = genRules(F_1);
 3: k = 2;
 4: while F_{k-1} ≠ ∅ do
 5:    CR_k = generateCandidatesRuleItems(F_{k-1});
 6:    for each case C ∈ D do
 7:       for each r : X → y ∈ CR_k do
 8:          if r ⊆_soft C then
 9:             r.anteSoftSuppSum += softSuppR(X, C);
10:             if y = C.solution then
11:                r.softSuppSum += softSuppR(X, C);
12:             end if
13:          end if
14:       end for
15:    end for
16:    F_k = {r ∈ CR_k |softSupp(r)| ≥ minsupp};
17:    SCAR_k = genRules(F_k);
18:    k++;
19: end while
20: SCARS = ⋃_{k≥minitemsize} SCAR_k;
21: prSCARS = pruneRules(SCARS);
22: Return prSCARS;
```

**Algorithm 1.** The algorithm for scars mining

contain $X$, and $r.softSuppSum$ storing the soft-support-sum of ruleitems in $\mathcal{D}$ that softly contain $X$ and also contain $y$. Thus, $softSupp(r) = \frac{r.softSuppSum}{|\mathcal{D}|}$ and $softConf(r) = \frac{r.softSuppSum}{r.anteSoftSuppSum}$.

Algorithm 1 is the algorithm for scars mining: (1) For 1-ruleitems $X \subseteq I$, we find $F_1$ as $F_1 = \{\{X\}|softSupp(X) \geq \text{minsupp}\}$. A set $SCAR_1$ is then generated by only choosing PRs from $F_1$ (lines 1 - 2). (2) For each $k$ subsequent pass, we find a set of new possibly frequent ruleitems $CR_k$ using $F_{k-1}$ found in the $(k-1)^{th}$ pass. We then scan $\mathcal{D}$, and updates the $anteSoftSuppSum$ and $softSuppSum$ of ruleitems in $CR_k$. We then generate a new $F_k$ by extracting ruleitems in $CR_k$ whose soft-support $\geq$ minsupp. A set $SCAR_k$ is generated by only choosing PRs from $F_k$ (lines 3 - 19). (3) From $SCAR_1$, ..., $SCAR_k$, we choose only sets whose $i \in [1, k] \geq$ *minitemsize* (*a user-specified minimum ruleitem size*), and store them in a set $SCARS$. Our idea is to choose a small representative subset of frequent ruleitems from the large number of resulting frequent ruleitems. The longer the frequent ruleitem, the more significant it is, driven from the studies [19]. We perform a rule pruning on ruleitems in $SCARS$. A rule $r$ is pruned, if $Laplace(r) <$ min-interesting. The set of ruleitems after the pruning is stored in a set $prSCARS$ and returned (lines 20 - 22).

## 5 A Unique Retrieval Strategy: USIMSCAR

Given a new problem $Q$, the goal of our novel retrieval strategy USIMSCAR is to generate a retrieval result $RR$. $RR$ consists of potentially useful objects that can be used to solve $Q$ by leveraging both similarity and association knowledge.

Such objects are obtained from both stored cases and scars mined. Let $\mathcal{D}$ be a set of cases. Let $prSCARS$ be the set of scars mined from $\mathcal{D}$. We below present the USIMSCAR algorithm:

```
1: RC = retrieveSimilarCases(Q, D);
2: RS = retrieveSimilarScars(Q, RC, prSCARS);
3: for each case C ∈ RC do
4:    r_C = getBestSCAR(C, prSCARS);
5:    if r_C ≠ ∅ then
6:       USF(C, Q) = SIM(C, Q) · Laplace(r_C);
7:    else
8:       USF(C, Q) = SIM(C, Q) · min-interesting;
9:    end if
10:   object = createObject(C, USF(C, Q));
11:   RR = RR ⋃ object;
12: end for
13: for each scar r ∈ RS do
14:   USF(r, Q) = SIM(r, Q) · Laplace(r);
15:   object = createObject(r, USF(r, Q));
16:   RR = RR ⋃ object;
17: end for
18: RR = enhanceObjects(RR);
19: Return RR;
```

**Algorithm 2.** The USIMSCAR algorithm

(1) In $\mathcal{D}$, we find the $k$ most similar cases $RC$ to $Q$ (line 1). We denote $SIM(C, Q)$ as the similarity between a case $C \in \mathcal{D}$ and $Q$.
(2) In $prSCARS$, we find the $k$ most similar scars $RS$ to $Q$ (line 2). A question raised is how to define a function $SIM(r, Q)$ that computes the similarity between a scar $r$ and $Q$. Our answer to it lies in our choice of the "cars representation" for scars mining. Note that scars have the *identical* structure as cases−the antecedents and consequents of scars correspond to the problem and solution part of cases respectively. Thus, $SIM(r, Q)$ can be defined in the same way as $SIM(C, Q)$, where $C$ is a case in $\mathcal{D}$. To generate $RS$, we only consider the scars in $prSCARS$ such that their itemsets in the antecedents are "soft-subsets" of cases in $RC$, rather than scanning all scars in $prSCARS$ for efficiency. We denote such rules as $RCS$. Note that each case $C \in RC$ is chosen as a similar case to $Q$ ($C \sim Q$). Assuming each scar $r \in RCS$ has the form $r : X \rightarrow y$, $X$ is a soft-subset of $C$ ($X \subseteq_{soft} C$). Since $C \sim Q$ and $X \subseteq_{soft} C$, $X \subseteq_{soft} C \sim Q$ can be derived. It implies that $RCS$ is the collection that is a particular subset (i.e. soft-subset) of cases in $RC$ similar to $Q$.
(3) For each case $C \in RC$, we select a special scar $r_C \in prSCARS$ (line 4). A rule $r \in prSCARS$ is chosen as $r_C$, if it has the highest interestingness (i.e. the Laplace measure) among those scars in $prSCARS$ such that their itemsets in the antecedents are "soft-subsets" of $C$ and their solutions in the consequents are "equal" to the solution of $C$. We then compute the usefulness of $C$ regarding $Q$ ($USF(C, Q)$) by combining $SIM(C, Q)$ and $Laplace(r_C)$ using the multiplication operation. If candidate(s) for $r_C$ is chosen more than one, let

us say $m$, we use the average of the interestingness of these $m$ scars to compute $Laplace(r_C)$. If there is no candidate for $r_C$, we use min-interesting for $Laplace(r_C)$. Note that in SBR, the usefulness of $C$ regarding $Q$ is generally measured by $SIM(C, Q)$. In contrast, our combination schemes aim to enhance such usefulness by leveraging $SIM(C, Q)$ and $Laplace(r_C)$. We then cast $C$ to a *generic object* $O$ that can encapsulate any cases and scars. $O$ has two fields: $O.inst = C$; $O.usf = USF(C, Q)$. $O$ is added to a retrieval result $RR$ (lines 4 - 11);
(4) For each scar $r \in RS$, we compute the usefulness of $r$ regarding $Q$ ($USF(r, Q)$) by combining $SIM(r, Q)$ and $Laplace(r_C)$ using the multiplication operator. This aims to directly leverage each scar in $RS$ whose interestingness is high with respect to $Q$. For each scar $r \in RS$, we cast $r$ to a generic object $O$. $O$ has two fields: $O.inst = r$; $O.usf = USF(r, Q)$. $O$ is added to $RR$ (lines 13 - 17).
(5) We further enhance the usefulness of each object in $RR$ (line 18). This is achieved based on the *solution occurrence* of objects in $RR$. Our premise is that if the solution of an object $O$ is more frequent in $RR$, $O$ is more useful in $RR$. The solution of each object $O \in RR$ is differently interpreted, according to whether $O$ was cast from a case $C$ or a scar $r$. If created from $C$, its solution indicates the solution of $C$; if created from $r$, its solution means the solution in the consequent of $r$. Let $S$ be a set of solutions of objects in $RR$. Let $S_O$ be a set of objects in $RR$ that have the solution equal to the solution of an object $O \in RR$. For each object $O \in RR$, we compute $\delta(S_O)$ as $\delta(S_O) = |S_O|/|RR|$ Finally, we enhance $O.usf$ by multiplying $\delta(S_O)$. Each object $O \in RR$, with its usefulness regarding $Q$ ($O.usf$), will be utilized to induce a solution for $Q$.

We now illustrate how USIMSCAR performs with the case base $D$ shown in Table 1, already used in Section 2, and how a solution is induced from $RR$. From $\mathcal{D}$, assume that we generate the following scars shown in Table 2.

**Table 2.** The scars generated

| Rules | *Laplace* | Soft-subset of |
|---|---|---|
| $r_1$: {($A_1$,right flank),($A_2$,vomit),($A_3$,38.6),($A_4$,yes),($A_5$,13)} → ($A_6$,appendicitis) | 0.922 | $P_1, P_2, P_3$ |
| $r_2$: {($A_1$,right flank),($A_2$,vomit),($A_3$,38.7),($A_4$,yes),($A_5$,10)} → ($A_6$,appendicitis) | 0.922 | $P_1, P_2, P_3$ |
| $r_3$: {($A_1$,right flank),($A_2$,vomit),($A_3$,38.8),($A_4$,yes),($A_5$,10)} → ($A_6$,appendicitis) | 0.922 | $P_1, P_2, P_3$ |
| $r_4$: {($A_1$,right flank),($A_2$,sickness),($A_3$,37.5),($A_4$,yes),($A_5$,35)} → ($A_6$,gastritis) | 0.775 | $P_4$ |

USIMSCAR takes the following steps (assume $k$=2 for steps (1) and (2)): (1) Retrieve the 2 most similar cases to $Q$: $RC = \{P_4, P_1\}$ with $SIM(P_4, Q) = 0.637$, $SIM(P_1, Q) = 0.631$. (2) Retrieve the 2 most similar scars to $Q$: $RS = \{r_1, r_4\}$ with $SIM(r_1, Q) = 0.640$, $SIM(r_4, Q) = 0.637$. (3) For each case $C \in RC$, $r_C$ is determined. For $P_4$, $r_4$ is selected. For $P_1$, $r_1$, $r_2$ and $r_3$ are selected. Thus, $USF(P_4, Q) = 0.494$, $USF(P_1, Q) = 0.581$. Then, $P_4$ with $USF(P_4, Q)$ and $P_1$ with $USF(P_1, Q)$ are cast to new generic objects, and stored in $RR$. (4) For each scar $r \in RS$, its usefulness regarding $Q$ is computed: $USF(r_1, Q) = 0.594$, $USF(r_4, Q) = 0.496$. Then, $r_1$ with $USF(r_1, Q)$ and $r_4$ with $USF(r_4, Q)$ are cast to new generic objects, and stored in $RR$. (5) Assume that each object in

$RR$ has a field $s$ holding its solution. $RR$ has four objects $RR = \{O_1, ..., O_4\}$ (see Table 3). As observed, there are only two sets of objects regarding solutions. For each object $O \in RR$, $O.usf$ is enhanced by weighting $\delta(S_O) = |S_O|/|RR|$. The enhancement results are shown under the column 'final usefulness' in the table. Finally, if we choose the most useful one with respect to $Q$, $O_3$ is chosen. Its solution 'appendicitis', $Q$ really had, is thus used as a diagnosis for $Q$.

**Table 3.** The retrieval result $RR$

| field: inst | field: usf | field: solution | final usefulness |
|---|---|---|---|
| $O_1.inst = P_4$, | $O_1.usf = 0.494$, | $O_1.s$ = gastritis | 0.247 |
| $O_2.inst = P_1$, | $O_2.usf = 0.581$, | $O_2.s$ = appendicitis | 0.291 |
| $O_3.inst = r_1$, | $O_3.usf = 0.594$, | $O_3.s$ = appendicitis | **0.297** |
| $O_4.inst = r_4$, | $O_4.usf = 0.496$, | $O_4.s$ = gastritis | 0.248 |

## 6 Evaluation

Our evaluation goal is to empirically show that USIMSCAR can improve SBR regarding retrieval performance. As a target application task, we choose a task highly dependent on the retrieval performance in CBR. One suitable task is to solve *classification* problems on the basis of the case-based approach. The case-based approach for classification is defined as follows [20]: given a new problem $Q$, its goal is to retrieve a set of similar cases to $Q$ from a case base, and classify $Q$ based on the retrieved cases. Thus, in principle, this approach is strongly dependent on the result obtained through retrieval in CBR. Based on this evaluation approach, we apply USIMSCAR and SBR in predicting an appropriate diagnosis for a new patient through information of patients whose diagnosis is already known with *medical* datasets found in UCI ML Repository[2].

### 6.1 Experimental Setup

SBR is typically realized through the approach using a derivative of the *nearest neighbor* algorithm [6]. This approach is called *k-nearest neighborhood retrieval* or simply $k$-NN. The idea of $k$-NN is that to solve a new problem $Q$, useful cases are determined using the $k$ most similar cases to $Q$. For our comparison purposes, we choose the following $k$-NN based approaches available in Weka [21]: (1) *IBk* is a simple implementation of $k$-NN, and relies on the Euclidean distance to find the $k$ most similar cases to $Q$; (2) *IBkCFS* denotes an approach integrating IBk with an algorithm of *feature selection*, a technique for determining relevant features from the original features of cases. The algorithm chosen is CfsSubsetEval available in Weka. IBk is extended to include feature selection by only considering relevant features when finding the similar cases to $Q$; (3) *IBkLVF* denotes as an approach integrating IBk with the feature selector ConsistencySubsetEval in Weka; (4) *IBkIG* denotes as an approach integrating IBk with an

[2] http://www.ics.uci.edu/~mlearn/MLRepository.html

algorithm of *feature weighting*, a technique for predicting optimal weights of the original features of cases. The chosen algorithm is InfoGainAttributeEval available in Weka. Integrating IBk with feature weighting is straightforward−features of cases (including $Q$) can be weighted by feature weighting, and then such features are used to find the similar cases to $Q$; and (5) *IBkCS* denotes an approach that integrates IBk with the feature weighting evaluator ChiSquaredAttributeEval available in Weka [21]. We also call these $k$-NN based approaches *classifiers*, since these will be used for classification tasks.

In the context of the $k$-NN classifiers, given $Q$, classification is done using two stages: the first is to retrieve a set of similar cases $RR$ to $Q$ using similarity knowledge, and the second is to classify $Q$ using the solutions (classes) in $RR$. In the context of USIMSCAR, these stages can be seen as follows. The first is to retrieve a set of *useful cases and rules* $RR$ regarding $Q$ using similarity and association knowledge, and the second is to classify $Q$ using information driven in $RR$. Our work is focused on the first stage. The second stage is often covered by *voting* [14]. We choose two well-known voting schemes to perform this stage: (1) *majority voting*−the majority solution in $RR$ is chosen as a solution for $Q$, and (2) *distance weighted voting*−each object $O \in RR$ gets to vote on the solution of $Q$ with a vote weighted by its similarity to $Q$ (in the context of the $k$-NN classifiers) or usefulness regarding $Q$ (in the context of USIMSCAR).

Table 4 provides a summary of the medical datasets used in our experiments.

**Table 4.** The medical datasets used in the experiments

| Dataset | No of Cases | No of Attributes | Attr Type | | No of Classes |
|---|---|---|---|---|---|
| | | | Numeric | Nominal | |
| Breast Cancer (BC) | 286 | 9 | | 9 | 2 |
| Breast Cancer Wins (BCW) | 683 | 10 | 10 | | 2 |
| Breast Tissue (BT) | 106 | 9 | 9 | | 6 |
| Pima Indian Diabetes (PID) | 768 | 9 | 9 | | 2 |
| StatLog Heart Disease (SHD) | 270 | 13 | 7 | 6 | 2 |
| New Thyroid (THY) | 215 | 5 | 5 | | 3 |

For our evaluation criteria, we use two metrics widely used for evaluating classifiers. The first is the *classification accuracy* that has been very often assumed to be the best performance indicator for evaluating classifiers. It measures the proportion of correctly classified instances out of all the tested instances. However, this accuracy does not take into account the *cost of making wrong decisions*. To supplement this lack, we choose *F-measure*. F-measure is defined as the harmonic mean between *precision* (P) and *recall* (R), denoted as F-measure $= \frac{2PR}{P+R}$. P represents the proportion of the instances, which truly have a class $X$, among all those classified as a class $X$. R indicates the proportion of the instances, classified as a class $X$, among all those instance having a class $X$. A high F-measure value indicates that both P and R are reasonably high.

USIMSCAR and the five $k$-NN based classifiers compared (simply 5CF) are all tested with six medical datasets by using *10-fold cross-validation*. In this validation practice, each dataset is partitioned into ten subsets. Of the ten subsets,

a single subset is retained as *testing data*, and the remaining nine subsets are used as *training data*. The cross-validation process is then repeated ten times (the folds), with each of the ten subsets used exactly once as the testing data.

The similarity knowledge, used in the experiments, is encoded as a similarity measure using the global-local principle explained in Section 3.1. Given a new problem $Q$ and a case $C$, their similarity is defined as the function $SIM$ in Eq. 1. We configured that this function is equally used in 5CF and USIMSCAR. To perform Algorithm 1, we use the following parameters: minsupp = 0.1 (10%); minsim = 0.95 (95%); min-interesting= 0.7 (70%); and minitemsize = $0.5 * N$, where $N$ is the total number of the attributes of instances in the training data.

### 6.2 Results and Analysis

For each dataset, we first report the mean number of scars generated by performing Algorithm 1, where the mean is attained by applying 10-fold cross-validation: BC:228, BCW: 1513, BT: 6013, PID: 524, SHD: 676, and THY: 1073. We found that the number of the scars generated increases with the increase in the number of the attributes of instances for each dataset. However, interestingly, the number of the scars acquired from BT is the highest as 6010, although the number of instances in the training data is the lowest as 96 (106 * 0.9). This was occurred, since some items in BT relatively appear very frequently.

To test approaches (USIMSCAR and 5CF), we need to find a best value for the top $k$ that indicates the number of the most similar cases to a new problem $Q$. We test 5CF using various values for the $k$, ranging from 1 to 15, since we observed that increasing $k$ beyond 15 hardly changed the results. To run USIMSCAR, we also need to find a value for the top $k$ that indicates the number of the most similar cases and scars to $Q$. We tested USIMSCAR using the same value range for finding it. Our comparison purposes, we finally use the best result obtained from the use of the best choice of the $k$ in terms of classification accuracy (CA) and F-measure (FM).

**Results Using Majority Voting.** We first present the results of USIMSCAR and 5CF using majority voting in terms of CA. Table 5 shows a summary of the results. The best approach is denoted in boldface for each dataset.

As observed in Table 5, USIMSCAR performs best for three dataset (BCW, BT, and THY). For BCW, its improvement over 5CF ranges from 0.44% to 0.73%. For BT, the improvement is up to 6.60%. For THY, the improvement is consistently equal to 1.4%. Whereas USIMSCAR occupies 2nd place for the datasets (BC, PID, and SHD). Thus, it outperforms the other four classifiers for the datasets. For BC, PID, and SHD, it performs better than those classifiers with a range of 1.75% - 2.45%, 0.39% - 1.43%, and 0.74% - 1.48%, respectively.

Table 6 shows a summary of the results of USIMSCAR and 5CF in terms of FM. The best one is also denoted in boldface for each dataset. As can be seen, USIMSCAR outperforms 5CF with three (BCW, BT, and THY) out of the six datasets. For BC, and BCW, its improvement ranges from 0.09% to 4.9%, and 0.47% to 0.78%, respectively. For THY, its improvement is consistently equal

**Table 5.** The best results using majority voting in terms of classification accuracy (%)

| Dataset | USIMSCAR | IBk | IBkCFS | IBkLVF | IBkIG | IBkCS |
|---|---|---|---|---|---|---|
| BC | 75.874 | 74.126 | **76.224** | 73.776 | 73.427 | 73.427 |
| BCW | **97.657** | 97.218 | 97.218 | 97.218 | 96.925 | 97.218 |
| BT | **71.698** | **71.698** | 65.094 | 67.925 | 69.811 | 70.755 |
| PID | 75.781 | 74.349 | **77.214** | 75.000 | 74.870 | 75.391 |
| SHD | 83.333 | 82.593 | 81.852 | 82.222 | **85.185** | 81.852 |
| THY | **97.674** | 96.279 | 96.279 | 96.279 | 96.279 | 96.279 |

**Table 6.** The best results using majority voting in terms of F-measure (%)

| Dataset | USIMSCAR | IBk | IBkCFS | IBkLVF | IBkIG | IBkCS |
|---|---|---|---|---|---|---|
| BC | **68.744** | 65.147 | 68.653 | 64.896 | 63.840 | 65.147 |
| BCW | **97.419** | 96.946 | 96.946 | 96.946 | 96.643 | 96.946 |
| BT | 71.178 | **71.465** | 63.249 | 65.127 | 67.689 | 68.435 |
| PID | 72.212 | 70.751 | **74.137** | 71.923 | 71.463 | 71.923 |
| SHD | 83.078 | 82.339 | 81.561 | 82.000 | **84.966** | 82.725 |
| THY | **96.883** | 95.084 | 95.084 | 95.084 | 95.084 | 95.084 |

to 1.80%. USIMSCAR occupies 2nd place for the other three datasets (BT, PID, and SHD). Thus, it performs better than the other four classifiers for the datasets. For BT, PID, and SHD, it outperforms these classifiers with a range of 2.74% - 7.93%, 0.29% - 1.46%, and 0.35% - 1.52%, respectively.

Through Tables 5 and 6, we find that USIMSCAR outperforms 5CF for the six datasets in 27 out of 30 comparisons in terms of both CA and FM. It indicates that using majority voting USIMSCAR achieves better performance in 90% of the cases than 5CF in terms of the metrics.

**Results Using Weighted Voting.** We now analyze the experimental results of USIMSCAR and 5CF using weighted voting in terms of CA. Table 7 shows a summary of the results. The best one is denoted in boldface for each dataset. As can be observed, USIMSCAR outperforms 5CF for all the six datasets. The improvement of USIMSCAR over them for each dataset is as follows. For BC, BCT, BT, PID, SHD, and THY, it is 4.90% - 6.64%, 0.29% - 0.58%, 6.64% - 13.20%, 10.42% - 13.15%, 5.93% - 7.78%, and 1.40%, respectively.

Table 8 shows a summary of the results of USIMSCAR and 5CF in terms of FM. The best one is also denoted in boldface for each dataset. As observed, USIMSCAR outperforms 5CF for all the six datasets. The improvement of USIMSCAR over them for each dataset is as follows. For BC, BCT, BT, PID, SHD, and THY, it is 10.19% - 12.12%, 0.32% - 0.62%, 6.16% - 14.16%, 13.83% - 15.15%, 6.10% - 7.83%, and 1.80%, respectively.

Through Tables 7 and 8, we find that USIMSCAR outperforms 5CF for the six datasets in all 30 comparisons in both CA and FM. This means that USIMSCAR achieves better performance in 100% of the cases than the classifiers in terms of both metrics.

**Table 7.** The best results using weighted voting in terms of classification accuracy (%)

| Dataset | USIMSCAR | IBk | IBkCFS | IBkLVF | IBkIG | IBkCS |
|---|---|---|---|---|---|---|
| BC | **79.021** | 73.427 • | 73.776 • | 72.727 • | 72.378 • | 74.126 • |
| BCW | **97.657** | 97.218 | 97.218 | 97.365 | 97.072 | 97.218 |
| BT | **78.302** | 71.698 • | 65.094 • | 67.925 • | 69.811 • | 71.698 • |
| PID | **87.500** | 74.479 • | 77.083 • | 75.391 • | 74.479 • | 74.349 • |
| SHD | **89.630** | 82.222 • | 82.222 • | 81.852 • | 83.704 • | 82.593 • |
| THY | **97.674** | 96.279 | 96.279 | 96.279 | 96.279 | 96.279 |

**Table 8.** The best results using weighted voting in terms of F-measure (%)

| Dataset | USIMSCAR | IBk | IBkCFS | IBkLVF | IBkIG | IBkCS |
|---|---|---|---|---|---|---|
| BC | **74.251** | 64.053 • | 65.245 • | 62.517 • | 62.127 • | 64.053 • |
| BCW | **97.421** | 96.946 | 96.946 | 97.104 | 96.798 | 96.946 |
| BT | **77.626** | 71.465 • | 63.466 • | 65.127 • | 67.689 • | 68.435 • |
| PID | **86.140** | 71.177 • | 74.055 • | 72.307 • | 70.995 • | 72.307 • |
| SHD | **89.553** | 82.034 • | 81.942 • | 81.723 • | 83.451 • | 81.723 • |
| THY | **96.883** | 95.084 | 95.084 | 95.084 | 95.084 | 95.084 |

Using the results shown in Tables 5 - 8, to find whether the performance improvement of USIMSCAR in terms of CA and FM is statistically significant over 5CF, we carried out statistical tests. A common approach for measuring a significant test for a difference between two *proportions* is the *Z-test* [22]. We performed statistical tests using the $Z$-test at 90% confidence. In the tables, '•' indicates that USIMSCAR attains a significant improvement over the target classifier at 90% confidence. As observed in Tables 5 and 6, using majority voting, there is no statistical difference between USIMSCAR and 5CF in terms of both CA and FM. However, as seen in Tables 7 and 8, 67% of comparisons (20 out of 30 comparisons) between USIMSCAR and 5CF are statistically significant in terms of both CA and FM. We note that insignificant improvement of USIMSCAR over 5CF does not mean that there is no differences between them, merely that the tests were unable to detect significance differences. As Keen [23] indicated, such improvement still can be important if it occurs repeatedly in many contexts. Thus, the insignificant improvement of USIMSCAR may be still valuable, and this is where a wide range of tests will be additionally carried out.

We emphasize that for all the six datasets, USIMSCAR using weighting voting ($U_{WV}$) outperforms USIMSCAR using majority voting ($U_{MV}$) in terms of both CA and FM. As observed in Table 9, $U_{WV}$ attains better performance than $U_{MV}$ in the 4.63% of the occasions in terms of CA on average. It attains better performance than $U_{MV}$ in the 5.34% of occasions in terms of FM on average. This findings indicate that it is more useful to utilize the quantified "usefulness information" of objects in the retrieval result $RR$ of USIMSCAR, rather than utilizing merely distribution information of the solutions in $RR$. Recall that $U_{WV}$ is configured to perform using the usefulness of objects in $RR$, which is quantified using both similarity and association knowledge. Whereas $U_{MV}$ is configured to perform using distribution information of solutions in $RR$.

**Table 9.** USIMSCAR with two voting schemes

(a) In classification accuracy

| Dataset | $U_{MV}$ | $U_{WV}$ | Diff |
|---|---|---|---|
| BC | 75.874% | 79.021% | 3.147% |
| BCW | 97.657% | 97.657% | 0% |
| BT | 71.698% | 78.302% | 6.604% |
| PID | 75.781% | 87.500% | 11.719% |
| SHD | 83.333% | 89.630% | 6.297% |
| THY | 97.674% | 97.674% | 0% |
| Mean | 83.670% | 88.297% | 4.628% |

(b) In F-measure

| Dataset | $U_{MV}$ | $U_{WV}$ | Diff |
|---|---|---|---|
| BC | 68.744% | 74.251% | 5.507% |
| BCW | 97.419% | 97.421% | 0.002% |
| BT | 71.178% | 77.626% | 6.448% |
| PID | 72.212% | 86.140% | 13.928% |
| SHD | 83.078% | 89.553% | 6.475% |
| THY | 96.883% | 96.883% | 0% |
| Mean | 81.586% | 86.979% | 5.393% |

## 7 Related Work

SBR has been successfully applied in various application domains, such as medical diagnosis [2] and help-desk service [3], to predict useful cases with respect to solving the target problem. As mentioned in Section 6, SBR is typically implemented through $k$-NN.

Over the years, researchers have extensively studied $k$-NN to enhance its accuracy. For example, it is shown that $k$-NN can be integrated with feature selection [24] or feature weighting [25]. As other trends, to enhance SBR, much work focuses on integrating SBR with *machine learning*, *domain knowledge*, and *adaptation knowledge*. The evolution of machine learning has resulted in retrieval approaches that combine SBR with rule-induction (RI) methods to improve SBR [26,9]. RI systems learn domain-specific knowledge, from stored cases, often represented as IT-THEN rules. Once this knowledge is available, SBR is augmented with rule-based reasoning using this knowledge. Several researchers propose a retrieval approach, in which similarity assessment during SBR is integrated with domain knowledge [8]. Aamodt [11] proposes an approach that cases are enriched with domain knowledge. Domain knowledge often represents the knowledge about the environment in which the target system operates, e.g. facts, heuristics. These approaches aim to guide the retrieval of relevant cases using domain knowledge. Some work tries to enhance SBR using adaptation knowledge. For example, the *adaption-guided retrieval* (AGR) approach is proposed [7], in which adaptation knowledge indicates whether a case can be easily modified to fit the new problem. In AGR, matches between the target problem and cases are done, only if there is enough evidence existed in adaptation knowledge that such matches can be catered for during retrieval. USIMSCAR differs from the above approaches in three aspects:

– *Knowledge acquisition*: The acquisition of both domain knowledge and adaptation knowledge is usually known as a very complex and difficult task [8], thus often leads to *knowledge bottleneck phenomenon*. The acquisition is also very often done with the support of domain experts [7,8]. However, the acquisition of association knowledge (AK) is straightforward, since AK is acquired from stored cases. This is also efficient, since acquisition is automatically done using association rule mining.

- *Knowledge formalization*: AK is formalized by capturing interesting associations, between known problem features and known solutions, shared by a large number of cases. In this context, this formalization can be compared to feature selection and feature weighting, since they focus on estimating the relevancy of certain problem features highly correlated to specific known solutions from the CBR viewpoint. However, they are often based on finding only relationships between single individual features and each solution, ignoring interesting relationships between *combinations* of features and each solution. The latter relationships may be curial in a certain circumstance. For example, two individual features may be strongly related to a certain solution but together may not, or vice versa. In contrast, AK can represent both kinds of relationships, thereby providing enriched relationships between the features of problems and solutions. This benefit originally comes from the use of association rule mining, which enables to extract all interesting correlations and frequent patterns derived in a case base.
- *Knowledge exploitation*: The uniqueness of USIMSCAR lies in the use of AK in conjunction with similarity knowledge (SK). This exploitation can be compared to the usage of the rules generated by RI methods from an analysis of cases. We note that this usage is classified into two schemes: one for weighting features [9], and the other for generating a solution for a given problem without using knowledge derived from similarity measurement between a given problem and cases (i.e. SK) [26]. In contrast, we leverage combined information inherent in both SK and AK for the retrieval process in CBR.

## 8 Conclusion and Future Work

In this paper, we proposed a new retrieval strategy USIMSCAR that outperforms similarity-based retrieval (SBR). Its uniqueness lies in leveraging association knowledge in conjunction with similarity knowledge during CBR retrieval. Also, we proposed a unique approach for extracting and representing association knowledge using association rule mining techniques. We evaluated USIMSCAR in comparison with SBR using medical datasets found in UCI ML Repository. The experimental results demonstrated that USIMSCAR is an effective retrieval strategy for CBR that qualitatively outperforms SBR.

USIMSCAR can be extended to cases, where each problem is represented by complex structures. In CBR, case problems can be characterized by not only attribute-value pairs, but also more complex structures like object-oriented (OO) or hierarchical (HR) representation [6]. The OO representation utilizes the data modeling approach of OO paradigm such as "is-a". In the HR representation, a case problem is characterized through multiple levels of abstraction, and its attribute values reference nonatomic cases. For USIMSCAR to treat the case problems characterized by such representations, two issues must be addressed−how to formalize similarity knowledge and association knowledge. To address the former, one may use the similarity approaches proposed in [27]. To address the latter,

one may integrate the soft-matching criterion and extended Apriori algorithms [28,29] for association rule mining in OO data and HR data. USIMSCAR can also be extended to cases, where each case problem is associated with more than one solution. This can be simply generalized into the occasion−a case problem is associated with one solution. The generalization is possibly done by splitting a case $C$ into $k$ number of sub cases, according to its $k$ number of solutions. All these cases are forced to have the same case 'id' of $C$. For example, there is a case $C = (X, Y)$ (id $= C$), where $X$ is a problem and $Y$ is a corresponding solution. Assume that $X$ consists of two treatments {2-tylenol, 2-aspirin}. For $C$ to be used in USIMSCAR, we split $C$ into two sub cases, whose ids are the same of $C$. By doing so, we obtain two cases: $C = (X$,2-tylenol) and $C = (X$,2-aspirin). Then, USIMSCAR can run for these cases without modification.

## References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications 7, 39–59 (1994)
2. Ahn, H., Kim, K.-j.: Global optimization of case-based reasoning for breast cytology diagnosis. Expert Syst. Appl. 36, 724–734 (2009)
3. Kang, Y.-B., Zaslavsky, A., Krishnaswamy, S., Bartolini, C.: A knowledge-rich similarity measure for improving it incident resolution process. In: Proceedings of the 2010 ACM SAC, pp. 1781–1788. ACM (2010)
4. Bradley, K., Smyth, B.: Personalized information ordering: a case study in online recruitment. Knowledge-Based Systems 16, 269–275 (2003)
5. Nilsson, M., Funk, P., Sollenborn, M.: Complex Measurement Classification in Medical Applications Using a Case-Based Approach. In: Ashley, K.D., Bridge, D.G. (eds.) ICCBR 2003. LNCS, vol. 2689, pp. 63–73. Springer, Heidelberg (2003)
6. De Mantaras, R.L., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M.L., Cox, M.T., Forbus, K., Keane, M., Aamodt, A., Watson, I.: Retrieval, reuse, revision and retention in case-based reasoning. Knowl. Eng. Rev. 20, 215–240 (2005)
7. Smyth, B., Keane, M.T.: Adaptation-guided retrieval: questioning the similarity assumption in reasoning. Artif. Intell. 102, 249–293 (1998)
8. Stahl, A.: Learning of knowledge-intensive similarity measures in case-based reasoning. PhD thesis. Technical University of Kaiserslautern (2003)
9. Cercone, N., An, A., Chan, C.: Rule-induction and case-based reasoning: hybrid architectures appear advantageous. IEEE Trans. on Know. and Data Eng. 11, 166–174 (1999)
10. Park, Y.J., Kim, B.C., Chun, S.H.: New knowledge extraction technique using probability for case-based reasoning: application to medical diagnosis. Expert Systems 23, 2–20 (2006)
11. Aamodt, A.: Knowledge-intensive case-based reasoning in CREEK. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 1–15. Springer, Heidelberg (2004)
12. Hoffmann, A., Khan, A.S.: A new approach for the incremental development of retrieval functions for CBR. Applied Artificial Intelligence 20, 507–542 (2006)
13. Castro, J.L., Navarro, M., Sánchez, J.M., Zurita, J.M.: Loss and gain functions for CBR retrieval. Inf. Sci. 179, 1738–1750 (2009)

14. Cunningham, P.: A Taxonomy of Similarity Mechanisms for Case-Based Reasoning. IEEE Trans. on Knowl. and Data Eng. 21, 1532–1543 (2009)
15. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: SIGMOD 1993, pp. 207–216. ACM (1993)
16. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Proceedings of the 4th KDD, pp. 443–447 (1998)
17. Nahm, U.Y., Mooney, R.J.: Mining soft-matching association rules. In: Proceedings of CIKM 2002, pp. 681–683 (2002)
18. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: A survey. ACM Comput. Surv. 38, 9 (2006)
19. Hu, T., Sung, S.Y., Xiong, H., Fu, Q.: Discovery of maximum length frequent itemsets. Inf. Sci. 178, 69–87 (2008)
20. Jurisica, I., Glasgow, J.: Case-Based Classification Using Similarity-Based Retrieval. In: International Conference on Tools with Artificial Intelligence, p. 410 (1996)
21. Witten, I.H., Frank, E.: Data mining: Practical machine learning tools and techniques with Java implementations. Morgan Kaufmann, San Francisco (2000)
22. Richard, C.S.: Basic Statistical Analysis. Allyn & Bacon (2003)
23. Keen, E.M.: Presenting results of experimental retrieval comparisons. Inf. Process. Manage. 28, 491–502 (1992)
24. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. 3, 1157–1182 (2003)
25. Park, J.H., Im, K.H., Shin, C.K., Park, S.C.: MBNR: Case-Based Reasoning with Local Feature Weighting by Neural Network: Special Issue: Soft Computing in Case Based Reasoning. Applied Intelligence 21, 265–276 (2004)
26. Auriol, E., Wess, S., Manago, M., Althoff, K.-D., Traphöner, R.: INRECA: A Seamlessly Integrated System Based on Inductive Inference and Case-Based Reasoning. In: Aamodt, A., Veloso, M.M. (eds.) ICCBR 1995. LNCS, vol. 1010, pp. 371–380. Springer, Heidelberg (1995)
27. Bergmann, R., Stahl, A.: Similarity measures for object-oriented case representations. In: Smyth, B., Cunningham, P. (eds.) EWCBR 1998. LNCS (LNAI), vol. 1488, pp. 25–36. Springer, Heidelberg (1998)
28. Kuba, P., Popelinsky, L.: Mining frequent patterns in object-oriented data. Technical Report, Masaryk University, Czech Republic (2005)
29. Pater, S.M., Popescu, D.E.: Market-Basket Problem Solved With Depth First Multi-Level Apriori Mining Algorithm. In: SOFA 2009. 3rd International Workshop on Soft Computing Applications, pp. 133–138 (2009)