A Retrieval Strategy for Case-Based Reasoning Using Similarity and Association Knowledge

Yong-Bin Kang, Shonali Krishnaswamy, and Arkady Zaslavsky

***Abstract*—Retrieval is a key phase in case-based reasoning (CBR), since it lays the foundation for the overall effectiveness of CBR systems. Its aim is to retrieve useful cases that can be used to solve the target problem. To perform the retrieval process, CBR systems typically exploit similarity knowledge and is called similarity-based retrieval (SBR). However, SBR tends to rely strongly on similarity knowledge, ignoring other forms of knowledge that can be further leveraged to improve the retrieval performance. This paper argues and motivates that association analysis of stored cases can significantly strengthen SBR. We propose a novel retrieval strategy USIMSCAR that substantially outperforms SBR by leveraging association knowledge, encoded via a certain form of association rules, in conjunction with similarity knowledge. We also propose a novel approach for extracting association knowledge from a given case base using various association rule mining techniques. We evaluate the significance of USIMSCAR in three application domains—medical diagnosis, IT service management, and product recommendation.**

***Index Terms*—Association knowledge (AK), association rule mining (ARM), case-based reasoning (CBR), CBR retrieval.**

I. Introduction

THE FUNDAMENTAL premise of case-based reasoning (CBR) [1] is that experience in the form of past cases can be leveraged to solve new problems. An individual experience is called a case, and its collection is stored in a case base. Typically, each case is described by a problem description and the corresponding solution description. Among the four typical phases in CBR (i.e., retrieval, reuse, revise, and retain), retrieval is a key phase in CBR, since the success of CBR systems is heavily reliant on the performance of retrieval [2]. Its aim is to retrieve useful or relevant cases that can be successfully used to solve a target problem. If the retrieved cases are not useful, CBR systems may not eventually produce a suitable solution to the problem.

Manuscript received August 23, 2011; revised October 03, 2012; accepted March 20, 2013. Date of publication May 14, 2013; date of current version March 13, 2014. This paper was recommended by Associate Editor A. Nuernberger.

Y.-B. Kang is with the Faculty of Information Technology, Monash University, Melbourne, VIC, 3145, Australia (e-mail: yongbin.kang@monash.edu).

S. Krishnaswamy is with the Faculty of Information Technology, Monash University, Melbourne, VIC, 3145, Australia, and also with the Data Analytics Department, Institute for Infocomm Research, A*STAR, Singapore (e-mail: shonali.krishnaswamy@monash.edu).

A. Zaslavsky is with the Information and Communication Technologies Centre, CSIRO ICT Centre, Canberra, Australia (e-mail: arkady.zaslavsky@csiro.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2013.2257746

Typically, retrieval is achieved through a specific strategy leveraging similarity knowledge (SK) and this is referred to as similarity-based retrieval (SBR) [2]. In SBR, SK is used to estimate the usefulness of stored cases with respect to a target problem. SK is usually encoded via similarity measures between the problem and stored cases. By using the measures, SBR finds cases ranked by their similarities to the problem, and then their solutions are used to solve the problem.

However, there are two main problems in SBR. First, SBR is too much dependent on domain experts to define SK in practice [3]. Unfortunately, no clear methodology or general approaches to support the modeling of such measures in an intelligent way have been developed yet. Thus, defining SK is still very complicated, time-consuming, and hard to practice. This often also leads SK to be poor, subjective, and inaccurate. Second, the definition of similarity measures is very often static so that the definition is highly possible to be applied to all target problems consistently. This leads to a problematic situation where a similarity criterion defined in a given domain is useful for some target problems but not useful for some other problems. Thus, depending on target problems, the retrieval performance of SBR is varied even in the same domain, as shown in [4].

In this paper, we propose that association analysis of stored cases can significantly enhance SBR. We propose a new retrieval strategy USIMSCAR[1] that leverages association knowledge (AK) in conjunction with SK. AK represents strongly evident, interesting relationships between known problem features and solutions shared by a large number of cases. We propose that this knowledge is generated by a particular form of association rules (ARs). Our notion of retrieval in this paper is to retrieve a combined set of both cases and rules relevant to a target problem, where the relevance is determined by quantification methods using an integration of SK and AK.

When compared to SK, AK acquisition is carried out through computer-supported association rule mining (ARM). AK is also not subjective in that it is built from observing the general evidence of associations between known problem features and solutions. Furthermore, AK is dynamic in that according to the characteristics of target problems, a best set of rules can be differently chosen and leveraged for the retrieval process. The key strength of USIMSCAR thus is to use AK to complement SK, thereby enhancing the performance of

[1]USIMSCAR is an acronym for a retrieval strategy based on the unified knowledge of similarity and soft-matching class ARs.

TABLE I
PATIENT CASE BASE

Attributes	Weights (w_i)	Patient 1	Patient 2	Patient 3	Patient 4	New Patient (NP)
Pain localiza	0.91	Right flank	Right flank	Epigastrium	Epigastrium	Right flank
Other slight pain	0.78	Vomit	Sickness	Nausea	Nothing	Nausea
Fever	0.60	38.7	37.5	36.8	38.2	37.8
Appetite loss	0.40	Yes	Yes	No	Yes	Yes
Age	0.20	11	35	20	25	14
Diagnosis		Appendicitis	Gastritis	Stitch	Gastritis	?
Similarity with NP		0.623	**0.637**	0.420	0.339	

SBR significantly. Through extensive evaluation, we validate that USIMSCAR significantly outperforms SBR and a well-known retrieval method adopting the concept of similarity and diversity [5] in three CBR applications: medical diagnosis, IT service management, and product recommendation applications.

This paper is organized as follows. Section II discusses our research motivation. Section III reviews the related work. Section IV presents the background of SK and AK. Section V proposes our approach for extracting AK. Section VI presents our novel retrieval strategy USIMSCAR. In Section VII, we evaluate USIMSCAR. Section VIII concludes this paper with future work.

II. MOTIVATING SCENARIO

Our research motivation is explained through a simple medical diagnosis scenario shown in [6], where the weight of each attribute was assigned by a domain expert. Consider that a case base $\mathcal{D}$ includes four patient cases as shown in Table I. For each case, the problem is described by five attributes (i.e., symptoms) A_1, ..., A_5, and the solution denotes the corresponding diagnosis. Our aim is to diagnose the correct disease for a new patient (NP). We note that NP is really suffering from appendicitis as specified in [6]. To predict a diagnosis for NP, in principle, SBR identifies similar cases to NP by finding cases whose attributes are similar to those of NP. The following metric is applied, used in [6], to measure the similarity between NP and each case $C \in \mathcal{D}$

$$sim_g(\text{NP}, C) = \frac{\sum_{i=1}^{n} w_i \cdot sim_l(q_i, c_i)}{\sum_{i=1}^{n} w_i}, \text{ where}$$
$$sim_l(q_i, c_i) = \begin{cases} 1 - \frac{|q_i - c_i|}{A_i^{\max} - A_i^{\min}}, \text{ (if } A_i \text{ is numeric), and} \\ 1, \text{ if } q_i = c_i, \text{ and 0, otherwise (if } A_i \text{ is nominal)} \end{cases} \quad (1)$$

where q_i and c_i are values of an attribute A_i of NP and the case C, respectively; n is the number of attributes of cases; and $A_i^{\max}$ and $A_i^{\min}$ denote the maximum and minimum values, respectively, that A_i takes on in $\mathcal{D}$.

Once similar cases to NP are selected, SBR determines a diagnosis for NP using these cases. Assume that SBR chooses the single most similar case to NP. As shown in Table I, Patient 2 is chosen when applying the above metric, and thus, the diagnosis choice for NP is gastritis. However, it is an incorrect solution, since NP is really suffering from appendicitis, thereby creating a dangerous situation for NP threatening his/her life. Thus, it is clear that SBR has a limitation due to its intrinsic characteristic, that is, its complete reliance on similarity measures.

TABLE II
EXAMPLE OF RULES

ID	Rule	Confidence
r_1:	{right flank} ∈ pain localiza & {vomit} ∈ other slight pain & {38.3, 38.7} ∈ fever & {yes} ∈ appetite loss & {11, 14, 15} ∈ age → appendicitis ∈ diagnosis	0.95
r_2:	{right flank} ∈ pain localiza & {38.5, 38.9} ∈ fever & {yes} ∈ appetite loss & {10, 13} ∈ age → appendicitis ∈ diagnosis	0.60
r_3:	{right flank} ∈ pain localiza & {37.5, 37.9} ∈ fever & {yes} ∈ appetite loss & {23, 25, 35} ∈ age → gastritis ∈ diagnosis	0.90

To overcome this limitation, we build AK from a given case base, and exploit it during retrieval. AK reflects on how known problem features are evidently associated with known solutions, and represented in the form of IF–THEN rules. With the scenario, suppose that we generate the rules shown in Table II from $\mathcal{D}$. Each rule has the form $X \rightarrow y$, where X is the antecedent and y is the consequent. Also, each rule holds with confidence c indicating if a new problem is satisfied with the antecedent, a likelihood for the problem to be satisfied with the consequent is c. We propose that the confidence is formalized using a rule interestingness measure in Section V.

We perform two steps for leveraging such rules during retrieval. First, we quantify the usefulness of each case C for NP by combining the similarity between C and NP, with the confidence of those rules relevant to C. We say that a rule $r : X \rightarrow y$ is relevant to a case, if r is supported by the case (i.e., the case contains a problem and solution that are highly similar to $X \cup y$). Thus, r_1 and r_2 are relevant to Patient 1, while r_3 to Patient 2. To retrieve cases similar to NP, we further leverage the confidence of those rules relevant to the cases, with their similarities to NP. This aims to improve the usefulness of the retrieved similar cases. If we apply multiplication for the combination, the usefulness of Patient 1 for NP is $0.623 \times 0.95 = 0.592$, while that of Patient 2 is $0.637 \times 0.90 = 0.573$. Thus, although the similarity of Patient 1 to NP is lower than that of Patient 2, its usefulness can be higher if its confidence is higher than that of Patient 2.

Second, we directly use discovered rules to identify specific rules that are relevant to NP. The goal is to further identify strongly evident rules that are relevant to NP and then quantify their usefulness with their confidence via a combination scheme, thereby leveraging the usefulness in the retrieval

process. In our approach, the relevance degree between a rule and NP is defined via their similarity (more discussed in Section VI). With the aforementioned scenario, if we adopt multiplication for the combination, the usefulness of r_1 for NP is $0.722 \times 0.95 = 0.686$ and that of r_3 is $0.714 \times 0.90 = 0.646$. As r_1's usefulness is higher than that of r_3, we might conclude that r_1 is more relevant to NP.

Through both steps, although Patient 1 is lower than Patient 2 in terms of the similarity to NP, we may draw the correct solution for NP from Patient 1 and r_1, if we measure and quantify their usefulness with respect to NP. Note that Patient 1 has the solution appendicitis and r_1 is associated with the same one which is the real diagnosis for NP.

Consequently, given a new problem Q, our objective is not only to retrieve useful (or relevant) cases for the problem, but also to identify useful ARs. These cases and rules, in conjunction with their quantified usefulness, are outcomes of our proposed USIMSCAR. Our approach can complement the limitation of SBR, thereby improving the overall retrieval performance. For example, as explained, in a situation where SBR retrieves a wrong case for Q via similarity measures, our approach can complement this situation by additionally leveraging AK. In some cases, we might also hardly find those cases similar to Q in SBR. In such cases, we can also get benefits by additionally approximating relevant cases and rules for Q by combining SK and AK. Also, if there exists a situation where SBR retrieves too many cases very similar to Q, our approach can be used to further choose a smaller number of optimal cases among them through AK.

III. Related Work

SBR has been widely used in various CBR applications, such as medical diagnosis [7], [8], IT service management [9], product recommendation [10], and personal rostering decisions [11], to predict relevant cases having an appropriate solution for a target problem. SBR has been typically implemented through k-nearest neighbor retrieval or simply k-NN [2]. The idea of k-NN is that retrieval is achieved through retrieving the k most similar cases to the target problem. However, a well-known limitation of k-NN lies in allowing irrelevant attributes to influence the similarity computation.

Over the years, approaches integrating data mining (DM) and k-NN have often been applied in the CBR research to improve k-NN through three main schemes. The first is to integrate feature selection (FS) or feature weighting (FW) into k-NN. In this context, FS is used to choose relevant features of cases [8], [11], FW is applied to estimate optimal weights of the original features of cases [12], [13], or their combination is used to leverage their advantages [7]. The second scheme is to combine data clustering with k-NN, where the structure of clustered cases is leveraged to guide more relevant cases [14], [15]. Given a case base, a set of clusters is constructed, where each cluster represents a group of relevant cases. For case retrieval, the similarity between a target problem and each case is combined with the relevance of the clustered group containing the case considered. The third scheme is to apply both SBR and DM techniques together to find relevant cases against the target problem. For example, Chuang [16] shows how to integrate DM with SBR to improve liver diagnosis. Given a target problem, once a DM method (a backpropagation neural network) is applied on the case base, some cases thought to be relevant to the problem are retrieved. These cases are then examined to check whether these are sufficiently similar to the target problem through SBR. Only similar cases are finally used as a retrieval result for the problem. Unlike to these schemes, our approach is based mainly on the use of AK built via ARM. AK identifies interesting associations between case features and solutions in the case base, while FS and FW focus on identifying key case features characterizing the case base. Also, while the third scheme uses DM separately from SBR, we incorporate AK into the retrieval process to enhance SBR.

SBR has also been integrated with statistical learning. For example, it is proposed that k-NN can be enhanced by dynamically finding an optimal number of the nearest cases for a target problem using the distribution of distances between potentially similar cases to the problem [17]. Also, a genetic algorithm is proposed to optimize the number of the nearest neighbors for the target problem [18]. These approaches are based on the observation that k-NN typically uses a fixed number of neighbors, leading to weaken the predictability of desired similar neighbors. However, a problem with these approaches is that the optimal set of the nearest cases are obtainable by using only SK. That is, using similarity measures, the candidates of relevant cases for the target problem are found, and then these are further examined through statistical methods using their similarity scores. Unlikely, our approach leverages two different forms of knowledge [i.e., AK (statistical information drawn from ARs) and SK] to enhance the use SK for retrieval.

The evolution of machine learning has resulted in retrieval approaches that combine SBR and rule-induction (RI) methods to improve SBR. RI systems often learn domain-specific knowledge and represent it as IF–THEN rules. It is proposed that such rules can be used for determining weights of case features in SBR [19]. Also, Huang *et al*. [20] show that decision tree algorithms can be applied to find domain-specific rules from a given case base. From such rules, users choose relevant rules according to the thresholds set up by experts. The extracted rules are then used to guide a target problem to its most similar case group and to estimate the weights of case features. Such knowledge is eventually used to retrieve the most similar case from the case base. A retrieval model in [4] dynamically selects SBR or a RI method (using decision trees) for the target problem, considering similarities and consistencies of cases in a case base. Our approach is different from these approaches in that: 1) AK is not used to measure weights of case features, but to refine the cases retrieved by SBR and guide more specific rules to the target problem; 2) AK is built not manually but automatically through ARM; and 3) unlike model [4], the leveraging of AK is integrated into the retrieval process to strengthen SBR.

Domain knowledge (DK) has also been combined with SBR. DK has been used to refine similarity measures to retrieve more accurate cases for a target problem. DK can

be encoded as the form of accurate training cases where these are chosen from the feedback about the usefulness of some cases previously assessed by domain experts [21]. It is also proposed that DK can be represented in the form of semantic knowledge capturing semantic meanings of cases to enhance the accuracy of SBR [22]. However, DK often does not exist in an explicit form, and thus implicitly available through the support of human domain experts who use informal methods when being asked to describe such knowledge. Unfortunately, it is hard to implement, especially if given domains are poor in domain theory. In contrast, building AK is straightforward as it is acquired via an analysis of cases, a fundamental knowledge source in CBR, without the support of domain experts.

IV. Background of SK and AK

Prior to presenting our proposed USIMSCAR, this section provides the background of SK and AK. First and foremost, we present our case representation scheme. To represent cases, we choose the attribute–value pairs representation, widely used in many CBR systems, 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 $(A_i, a_i)_{i\in[1,m]}$, where A_i is an attribute (or feature[2]) and a_i is a value of A_i. A case C is the form of $C = (X, Y)$ where X is a problem $X = \{(A_1, a_1), ..., (A_{m-1}, a_{m-1})\}$, and Y is the corresponding solution $Y = (A_m, a_m)$. We refer to A_m as a solution attribute. A case base is a set of cases.

A. Background of SK

In a CBR context, SK is referred to as the knowledge encoded via measures computing similarities between a target problem Q and cases. In SBR, SK is used to represent a heuristic for estimating the usefulness of stored cases with respect to Q. The higher the similarity between Q and a case C is, the more useful C is for Q. A formulation of similarity measures suitable for cases represented by attribute–value pairs is often based on a widely used principle. It is the local–global principle that decomposes a similarity measure by local similarities for individual attributes of cases and a global similarity aggregating these similarities [21]. An accurate local similarity function relies on attribute types. A global similarity function can be arbitrarily complex, but simple functions are usually used such as weighted average aggregation [21]. For example, referring to (1), sim_g is a global similarity measure and sim_l is a local similarity measure.

B. Background of AK

Our underlying premise in this paper is that SBR can be qualitatively enhanced by the inclusion of AK. AK aims to represent evidently interesting relationships shared by a large number of relevant stored cases, using a combination of various DM techniques. These are ARM [23], class ARM [24], and soft-matching ARM (SARM) [25].

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

1) *ARM:* ARM aims to mine certain interesting relationships, called associations, in a transaction database [23]. It focuses on discovering a set of highly co-occurred features shared by a large number of transactions in the database. A formal definition of ARs is described as follows [23].

1) Let I be a set of distinct literals called items. A set of items $X \subseteq I$ is called an itemset.
2) 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.
3) Every AR has two parts: an antecedent and a consequent. An AR is an implication of the form $X \rightarrow Y$, where $X \subseteq I$ is an itemset and is the antecedent and $Y \subseteq I$ is an itemset which is the consequent, and $X \cap Y = \emptyset$.
4) 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.

In a CBR context, ARM can be used to discover interesting relationships from a given case base. A transaction and an item can be seen as a case and an attribute–value pair, respectively. Apriori [23] is one of the traditional algorithms for ARM. Interestingness measures are useful to evaluate the quality and rank a large number of ARs extracted [26]. As candidates of the measures, the support and confidence criteria are often used. In general, the problem of ARM is to generate all ARs that have support and confidence not less than a user-specified minimum support minsupp and a user-specified minimum confidence minconf, respectively.

2) *Class ARM:* Class association rules (CARs) [24] are a special subset of ARs whose consequents are restricted to a single target variable. In a CBR context, a CAR is seen as an AR whose consequent holds the item formed as a pair of a solution attribute and a value of it. We call such an item a solution item. A CAR thus has the form $X \rightarrow y$, where $X \subseteq I$ is an itemset and $y \in I$ is a solution item. From Table I, a simple CAR can be expressed as $(\text{fever}, 38.7) \rightarrow$ (diagnosis, appendicitis).

It is noteworthy that to represent AK, we adopt the CAR representation. AK is encoded to reflect how certain problem features are interestingly associated with specific solutions in a given case base. Considering this, note that the form of a CAR $X \rightarrow y$ enables us to represent an association between an itemset X (i.e., a set of problem features) and a solution item y (i.e., the corresponding solution) in a simple way.

3) *SARM:* Consider a rule $X \rightarrow Y$. A limitation of traditional ARM algorithms (e.g., Apriori [23]) 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, in a supermarket sales database, 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 soft-matching criterion was proposed [25], where the antecedents and consequents of ARs are found

by similarity assessment.[3] Using this criterion, the problem of SARM is to find all rules of the form $X \rightarrow Y$, where the soft support and soft confidence of each rule are not less than minsupp and minconf, respectively. The definitions of soft support and soft confidence are generalized by using support and confidence. This generalization is done by allowing items to match, as long as their similarity exceeds a user-specified minimum similarity minsim). 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.

V. AK Representation

This section presents our approach for extracting and representing AK using the techniques outlined in Section IV. The aim of building AK is twofold: 1) representing strongly evident associations between known problem features and solutions from a given case base, and 2) valuably leveraging these associations along with SK in USIMSCAR.

We propose to represent AK via special CARs whose antecedents are determined by using the soft-matching criterion. We refer to these rules as soft-matching class association rules (SCARs). A SCAR has an implication of the form $X \rightarrow y$, where X is a frequent itemset representing problem features that occur frequently and are discovered by the soft-matching criterion from the case base. And y denotes a solution item.

A SCAR $X \rightarrow y$ thus implies that a target problem Q is likely to be associated with the solution contained in y, if Q's problem features are sufficiently similar to X. The likelihood is quantified by the interestingness of the rule. As mentioned, support and confidence are often used for this purpose. On some occasions, their combination is also used. Often, a rationale for doing so is to define a single optimal interestingness (e.g., the Laplace measure [27]). We now present the definition of SCARs.

1) Let $\mathcal{D}$ be a set of cases, each being described by attributes $A_1, ..., A_m$. A pair $(A_i, a_i)_{i\in[1,m-1]}$ is an item. A pair (A_m, a_m) is a solution item. Let I be a set of items. The k number of items in an itemset is called k-itemset or simply an itemset.
2) Let $sim(x, y)$ be a function computing the similarity between two items $x, y \in I$ in terms of their attribute values. For this, we construct an $m \times m$ similarity matrix, where m is the total number of items in $\mathcal{D}$. Each entry of this matrix represents a similarity score between any two items. We say that x and y are similar, iff $sim(x, y) \geq$ minsim. The similarity is normalized into [0, 1].
3) Let $asim(X, Y)$ be a function that computes the asymmetric similarity of X with respect to Y, where $X, Y \subseteq I$ are two given itemsets and $|X| \leq |Y|$

$$asim(X, Y) = \sum_{x,y} \text{sim}(x, y)/|X| \tag{2}$$

where $x \in X$ and $y \in Y$ are two items whose attributes' labels are the same. We say that X is a soft subset of Y ($X \subseteq_{\text{soft}} Y$) or Y softly contains X, iff $asim(X, Y) \geq$ minsim. We also say that Y contains X, if $X \subseteq Y$ holds.

[3]The detailed information about this criterion is found in [25].

4) The soft-support sum of an itemset X, $\text{softsupp}_{\text{sum}}(X)$, is defined as the sum of the asymmetric similarities of X with respect to cases in $\mathcal{D}$ that softly contain X. The soft support of X, $\text{softsupp}(X)$, is then computed by dividing $\text{softsupp}_{\text{sum}}(X)$ with $|\mathcal{D}|$.
5) The soft-support sum of a rule, $r : X \rightarrow y$, is defined as the sum of the asymmetric similarities of X with respect to cases in $\mathcal{D}$ that softly contain X, and contain y. The soft support of r, $\text{softsupp}(r)$, is then computed by dividing $\text{softsupp}_{\text{sum}}(r)$ with $|\mathcal{D}|$. The soft-confidence sum of r is defined as the sum of the asymmetric similarities of X with respect to cases in $\mathcal{D}$ that softly contain X and also contain y. The soft confidence of r, $\text{softconf}(r)$, is computed by dividing r's soft-confidence sum with $|\mathcal{D}|$.

A rule item is of the form $\langle X, y\rangle$ and basically represents a SCAR $X \rightarrow y$. The key operation for SCAR mining is to find all rule items that have soft support not less than minsupp. We call such rule items frequent rule items. For all the rule items that have the same antecedent, the one with the highest interestingness is chosen as a possible rule (PR). We use the Laplace measure [27] of interestingness that combines soft support and soft confidence such that they are monotonically related (i.e., positively correlated). Given a rule item r, its Laplace measure, $\text{Laplace}(r)$, is given as

$$\frac{|\mathcal{D}|\text{softsupp}(r) + 1}{|\mathcal{D}|\text{softsupp}(r)/\text{softconf}(r) + 2}. \tag{3}$$

Since $|\mathcal{D}|$ is the constant, it is easy to see that this measure is monotone in both soft support and soft confidence. If $\text{Laplace}(r) \geq$ a user-specified minimum level of interesting minint, we say r is accurate. A set of SCARs consists of all the PRs that are frequent and accurate.

The SCAR mining algorithm is presented in Algorithm 1. It first computes the soft support of an individual rule item and decides whether it is frequent. In each subsequent pass, it begins with the seed set of rule items found as frequent in the prior pass. It uses the seed to generate new possibly frequent rule items called candidate rule items. The soft support of these rule items are computed during the pass over $\mathcal{D}$. At the end of this pass, it decides which of the candidate rule items are frequent. From the frequent rule items, it produces SCARs after a rule pruning.

Let k-rule item be a rule item whose antecedent has k items. Let F_k be a set of frequent k-rule items. The detailed algorithm is described as follows.

Step 1: We find a set of frequent 1-rule items F_1 (line 1). Given a 1-rule item X, if $\text{softsupp}(X) \geq$ minsupp, then X is added to F_1. A set of SCARs is then generated by only finding PRs from F_1 (line 2).

Step 2: For each subsequent pass k, we generate a set of candidate rule items CR_k from F_{k-1} found in the $(k-1)$th pass (line 5). This process is similar to apriori-gen() in Apriori [23] except that only those $(k-1)$-rule items with the same solution item are used to generate the candidate k-rule items. We then scan $\mathcal{D}$ and update the soft support of rule items in CR_k (lines 6–15). Finally, we generate a new frequent rule item set F_k by extracting rule items from CR_k whose soft support $\geq$ minsupp.

Algorithm 1 SCAR Mining Algorithm

1: $F_1 = findFrequentRuleitems(\mathcal{D})$;
2: $SCAR_1 = genRules(F_1)$;
3: $k = 2$;
4: **while** $F_{k-1} \neq \emptyset$ **do**
5: $CR_k = generateCandidatesRuleitems(F_{k-1})$
6: **for** each case $C \in \mathcal{D}$ **do**
7: **for** each $r : X \rightarrow y \in CR_k$ **do**
8: **if** $r \subseteq_{soft} C$ **then**
9: $softsupp_{sum}(X)$ += $asim(X, C)$;
10: **if** $y = C$.solution **then**
11: $softsupp_{sum}(r)$ += $asim(X, C)$;
12: **end if**
13: **end if**
14: **end for**
15: **end for**
16: $F_k = \{r \in CR_k \mid softsupp(r) \geq minsupp\}$;
17: $SCAR_k = genRules(F_k)$;
18: k++;
19: **end while**
20: $SCAR_U = \bigcup_{k \geq minitemsize} SCAR_k$;
21: $SCAR_{set} = pruneRules(SCAR_U)$;
22: **Return** $SCAR_{set}$;

A set of rule items is then generated by only choosing PRs from F_k (lines 16 and 17).

Step 3: From $SCAR_1$, ..., $SCAR_k$, we choose only sets whose k is greater than or equal to a user-specified minimum itemset size minitemsize, and store them in a set $SCAR_U$ (line 20). Our intuition is to choose only significant frequent rule items from the large number of resulting frequent rule items. Our premise is that the longer the frequent rule item, the more significant it is. Very often, the significance of frequent itemsets correlates with their length [28]. We then apply a rule pruning on rule items in $SCAR_U$ (line 21). A rule r is pruned, if Laplace(r) $<$ minint. The set of rule items $SCAR_{set}$ after the pruning is finally returned.

For ease of notation of SCARs, we follow the notation of the rules shown in Table II. In the antecedent of each rule, values found to be similar for a given item are shown in an ordered set enclosed by parentheses. The set implies that the value at the leftmost position is similar to the other values in the set. For example, referring to $\{11, 14, 15\} \in$ age in r_1, the value 11 of the age attribute is similar to 14 and 15. The interestingness, measured by the Laplace measure, of each SCAR is marked under confidence in Table II.

VI. USIMSCAR: The Unique Retrieval Strategy

This section presents USIMSCAR that leverages AK encoded via SCARs along with SK to enhance SBR. The main challenge is how to combine these two types of knowledge effectively, thereby enhancing the retrieval performance of SBR. We address it by proposing the USIMSCAR algorithm. We also illustrate how it performs with an example scenario.

A. USIMSCAR Algorithm

Our rationale for leveraging AK in USIMSCAR is twofold: 1) enhancing the usefulness of cases, retrieved via SK, with respect to a new problem Q by additionally including the SCARs, thereby meaningfully exploiting the cases with their usefulness; and 2) directly leveraging AK by finding those SCARs whose usefulness is high with respect to Q, thus valuably exploiting them with their usefulness.

Algorithm 2 USIMSCAR Algorithm

1: SC = $retSimCases(k_{sc}, Q, \mathcal{D})$;
2: SS = $retSimScars(k_{ss}, Q, SC, SCAR_{set})$;
3: **for** each case $C \in$ SC **do**
4: $r_C = findMostRelSCAR(C, SCAR_{set})$;
5: **if** $r_C \neq \emptyset$ **then**
6: $usf(Q, C) = sim_g(Q, C) * Laplace(r_C)$;
7: **else**
8: $usf(Q, C) = sim_g(Q, C) *$ minint;
9: **end if**
10: obj = $createObject(C, usf(Q, C))$;
11: RR = RR $\bigcup$ obj;
12: **end for**
13: **for** each SCAR $r \in$ SS **do**
14: $usf(Q, r) = sim_g(Q, r) * Laplace(r)$;
15: obj = $createObject(r, usf(Q, r))$;
16: RR = RR $\bigcup$ obj;
17: **end for**
18: RR = $enhanceObjects$(RR);
19: **Return** RR;

Given a new problem Q, USIMSCAR generates a retrieval result (**RR**) consisting of potentially useful objects that can be used for solving Q. Such objects can be chosen as in the form of cases from a given case base $\mathcal{D}$ and also as in the form of SCARs from $SCAR_{set}$ generated by Algorithm 1. The USIMSCAR algorithm is described as follows.

Step 1: From $\mathcal{D}$, we find the k_{sc} most similar cases (**SC**) to Q (line 1). We denote $sim_g(Q, C)$ as the similarity between Q and a case $C \in \mathcal{D}$.

Step 2: From $SCAR_{set}$, we find the k_{ss} most similar SCARs (**SS**) to Q (line 2). The aim is to discover specific SCARs that are highly similar to Q and exploit them valuably in Step 4. We observed that there are often specific SCARs whose antecedents are highly similar to Q and are not witnessed in cases in **SC**. The finding of such rules thus enables us to identify additionally useful rules for Q.

But a question raised is how to define a function $sim_g(Q, r)$ computing the similarity between Q and a SCAR $r \in SCAR_{set}$. Our answer to it lies in our choice of the CAR representation for SCAR mining. Note that SCARs have the same structure with cases: the antecedents and consequents correspond to the problems and solutions of cases, respectively. Thus, $sim_g(Q, r)$ can be defined in the same way as $sim_g(Q, C)$ in Step 1. If there is no item with an attribute, we set the similarity for it as 0. Referring to Tables I and II, sim_g(Patient 1,r_2) is computed by aggregating similarities for attributes pain localiza, fever, appetite loss, and age.

Note that when measuring the similarity between values for a case C and an item of a rule r on the same attribute, we use the value of the attribute of C and the value, at the leftmost position, belonging to the attribute of the item in r. Referring to Tables I and II, if we measure the similarity between 38.7 for Patient 1 and a set of the values $\{38.3, 38.7\}$ for r_1 on fever, the similarity between 38.7 and 38.3 is computed. The reason for choosing to use 38.3 lies in the interpretation of the notation of $\{38.3, 38.7\} \in$ fever. As explained in the previous section, the notation implies that 1) 38.3 is an anchor value in a sense that it is compared by all the other values (i.e., 38.7) in

the set, and 2) this anchor value is similar to all the other values in the set. Thus, 38.3 is chosen to measure the similarity.

Note that to generate **SS**, we consider only SCARs in SCAR_{set} such that their antecedents are soft subsets of cases in **SC** rather than all SCARs in SCAR_{set} for efficiency. Assume that each SCAR has the form $X \rightarrow y$. For each SCAR $r \in \text{SCAR}_{\text{set}}$, we can find a case $C \in \mathbf{SC}$ that softly contains r's X. Since C is similar to Q ($C \sim Q$) and $X \subseteq_{\text{soft}} C$, the relation $X \sim Q$ can be derived. By doing so, we can reduce a number of candidates that consist of **SS**.

Step 3: For each case $C \in \mathbf{SC}$, we select the most relevant SCAR $r_C \in \text{SCAR}_{\text{set}}$ (line 4). We say that in SCAR_{set}, a SCAR $r : X \rightarrow y$ is relevant to C, if $X \subseteq_{\text{soft}} C$ and y is equal to C's solution. Among the relevant SCARs in SCAR_{set} to C, we say that a SCAR whose interestingness is highest is the most relevant SCAR to C, denoted by r_C.

We then quantify the usefulness of C with respect to Q, $\text{usf}(Q, C)$, by integrating $sim_g(Q, C)$ (i.e., SK) and $\text{Laplace}(r_C)$ generated from AK via multiplication (line 6). The higher such usefulness is, the more useful the case C is for Q. If there is no candidate for r_C, we use minint as $\text{Laplace}(r_C)$ (line 8). The intuition is that since such r_C is not relevant to Q at all, we treat it to be the least interesting rule with respect to Q by assigning it to the lowest boundary of interestingness, minint. We then cast the case C into a generic object encapsulating any cases and SCARs. This object has two fields: inst stores C and usf stores $\text{usf}(Q, C)$. Then, the object is added to **RR** (lines 10 and 11).

Step 4: For each rule $r \in \mathbf{SS}$, we quantify the usefulness of r with respect to Q by integrating $sim_g(r, Q)$ (i.e., SK) and $\text{Laplace}(r)$ acquired from AK via multiplication (line 14). We then cast r into a generic object, where inst stores r and usf stores $\text{usf}(Q, r)$. This object is added to **RR** (lines 15 and 16).

Step 5: We further enhance the usefulness of each object $o_i \in \mathbf{RR}$, where $1 \leq i \leq |\mathbf{RR}|$ (line 18). This is achieved using the frequency of solution occurrence among objects in **RR**. Our intuition is that if o_i's solution is more frequent in **RR**, o_i is more useful in **RR**. If o_i is cast from a case C, o_i' solution is equal to C's solution. If o_i is cast from a SCAR r, o_i's solution is the solution in r's consequent. Let S be the set of solutions of objects in **RR**. Given an object $o_i \in \mathbf{RR}$, let S_{o_i} be a set of objects in **RR** whose solutions are equal to o_i's solution. Let $\delta(S_{o_i})$ be $|S_{o_i}| / \max(S_{o_k})$ for all $1 \leq k \leq |\mathbf{RR}|$. Finally, we enhance o_i.usf by including $\delta(S_{o_i})$ as o.usf = o.usf $* \ \delta(S_{o_i})$. Eventually, **RR** is returned.

To sum up, USIMSCAR's output is a collection of particular objects consisting of both cases and SCARs that have been evaluated to be relevant to the given target problem. These objects have their own usefulness quantified using both SK and AK. The significance of these objects with respect to improving the retrieval performance is evaluated in Section VII. Before presenting the evaluation results, in the following, we demonstrate how USIMSCAR performs with an example to help to provide a better understanding of its algorithm.

B. Illustration of USIMSCAR With an Example

We illustrate how USIMSCAR produces a retrieval result **RR** from the case base shown in Table I. We also show how we predict the correct solution from **RR** for the patient NP.

TABLE III
THREE SCARS EXTRACTED FROM TABLE I

ID	Rule	Laplace	$\subset_{\text{soft}}$
r_1:	{right flank} ∈ pain localiza & {vomit} ∈ other slight pain & {38.3, 38.7} ∈ fever & {yes} ∈ appetite loss & {11, 14, 15} ∈ age → appendicitis ∈ diagnosis	0.95	Patient 1, ...
r_2:	{right flank} ∈ pain localiza & {38.5, 38.9} ∈ fever & {yes} ∈ appetite loss & {10, 13} ∈ age → appendicitis ∈ diagnosis	0.60	Patient 1, ...
r_3:	{right flank} ∈ pain localiza & {37.5, 37.9} ∈ fever & {yes} ∈ appetite loss & {23, 25, 35} ∈ age → gastritis ∈ diagnosis	0.90	Patient 2, ...

For the sake of simplicity, assume that the SCAR mining algorithm generates the three SCARs shown in Table III from the case base $\mathcal{D}$. This table is the same as Table II, except that the heading confidence is replaced with Laplace. Further, each list under the heading $\subset_{\text{soft}}$ denotes those cases that softly contain the SCAR at the same row (e.g., Patient 1 softly contains r_1; alternatively, r_1 is a soft subset of Patient 1).

Given NP, USIMSCAR performs the following steps using the SCARs in Table III, assuming that the number of the most similar cases and SCARs to NP is set equally as 2, respectively. Also, assume that we only consider four patient cases in Table I to find similar cases to NP for simplicity.

Step 1: USIMSCAR retrieves the two most similar cases to NP: **SC** = {Patient 2, Patient 1}, where sim_g(NP, Patient 2) = 0.637 and sim_g(NP, Patient 1) = 0.623.

Step 2: USIMSCAR retrieves the two most similar SCARs to NP: **SS** = $\{r_1, r_3\}$, where $sim_g(\text{NP}, r_1) = 0.722$ and $sim_g(\text{NP}, r_3) = 0.714$.

Step 3: For each case $C \in \mathbf{SC}$, USIMSCAR selects its most relevant SCAR $r_C \in \text{SCAR}_{\text{set}}$. With this example, $r_{\text{Patient 2}}$ is chosen as r_3, and $r_{\text{Patient 1}}$ as r_1. The usefulness of C with respect to NP is computed as $sim_g(\text{NP}, C) * \text{Laplace}(r_C)$. Thus, usf(NP, Patient 2) = 0.573 and usf(NP, Patient 1) = 0.592. Finally, C is cast as a generic object o, where o.inst = C and o.usf = usf(NP, C), and o is stored in **RR**.

Step 4: For each SCAR $r \in \mathbf{SS}$, USIMSCAR computes the usefulness of r with respect to NP: $\text{usf}(\text{NP}, r_1) = 0.686$ and $\text{usf}(\text{NP}, r_3) = 0.643$. After that, r is also cast as a generic object o, where o.inst = r and o.usf = usf(NP, r), and o is stored in **RR**.

Step 5: Suppose that each object in **RR** has a field s holding its solution. **RR** is then **RR** = $\{o_1, ..., o_4\}$, where

o_1.inst = Patient 1, o_1.usf = 0.592, o_1.s = appendicitis
o_2.inst = Patient 2, o_2.usf = 0.573, o_2.s = gastritis
o_3.inst = r_1, o_3.usf = 0.686, o_3.s = appendicitis
o_4.inst = r_3, o_4.usf = 0.643, o_4.s = gastritis.

Thus, $S_{\text{Patient 1}}$ = {Patient1, r_1} and $S_{\text{Patient 2}}$ = {Patient 2, r_3}. So, $\delta(S_{\text{Patient 1}}) = 1.0$ and $\delta(S_{\text{Patient 2}}) = 1.0$. Finally, the

usefulness of each object in **RR** is enhanced by multiplying $\delta(S_{\text{Patient 1}})$ or $\delta(S_{\text{Patient 2}})$, according to whether its solution is appendicitis or gastritis. With this scenario, the enhancement results are the same as the above. Eventually, if we choose the single most useful object in **RR** for NP, we choose o_3 and its solution appendicitis is provided to NP. This is correct, as NP was identified to suffer from a disease appendicitis as previously mentioned.

As examined, our approach can enhance SBR to find the correct solution for NP via three schemes. First, a set of similar cases to NP retrieved can be further utilized with their relevant SCARs to measure their usefulness with respect to NP. Second, a set of similar SCARs to NP is discovered, and then their interestingness along with their similarities are both leveraged to measure the usefulness of such SCARs. Third, both the retrieved cases and SCARs are further utilized through the use of the frequency of solution occurrence witnessed within themselves to enhance their usefulness.

VII. USIMSCAR Evaluation

We evaluate whether USIMSCAR is an effective retrieval method that performs successfully by applying it to various benchmark and real-life datasets in three application domains: medical diagnosis (MD), IT service management (ITSM), and product recommendation (PR) domains. To show the superiority of USIMSCAR's performance, we compare it with the performance of several retrieval methods adopting SBR. Further, we compare it with one well-known CBR retrieval method that uses a quality metric combining diversity and similarity [5]. This aims to provide a more general spectrum of USIMSCAR's performance by comparing the retrieval method designed to enhance SBR that uses the diversity concept in conjunction with similarity.

A. Evaluation Methodology

For the MD domain, USIMSCAR is applied to estimate the correct diagnosis for illness associated with symptoms of a given patient. For the ITSM domain, USIMSCAR is used to predict the correct IT support group workgroup for a given IT incident problem. A typical IT support organization is structured as a complex network of workgroups, each comprising of a set of skilled technicians dealing with problems presented by customers. Generally, assigning presented problems to appropriate workgroups is largely manual, thereby error-prone and leading to multiple bouncing of the problem within various workgroups [9]. For the PR domain, USIMSCAR is applied to predict the correct rating that a given user will be likely to rate toward a product presented by the user.

1) *Datasets Used:* Table IV shows a summary of all datasets used in our experiments. For the MD domain, we use seven medical datasets, where six are found in UCI ML repository.[4] The other one is a real-life medical dataset (NHSG) obtained from the U.K. National Health Service (Grampian) Health and Safety. In all the datasets, the problem part of each case denotes a set of symptoms or situations of a patient, and the solution part represents the corresponding diagnosis class or care stage. For each dataset, we choose the training and testing data using tenfold cross validation (CV) [29].

For the ITSM domain, we use a real-life incident dataset (IMData) obtained from an installation of HP Service Manager.[5] IMData consists of 6514 incident cases resolved by six workgroups in 2007, where 5211 (80%) cases were chosen as training data and the remaining 1303 (20%) cases as testing data. The problem part of each case is described using three attributes: 1) incident category—the main criterion for classifying a given IT incident problem (e.g., account and application error); 2) incident topic—specifying detailed entries specific to the incident category (e.g., printing and database); and 3) incident description—describing a customer's perception about the problem (e.g., cannot print from Bizflow). The solution part is labeled as one of the six workgroups WG1, ..., WG6.

For the PR domain, we use a large movie dataset, Yahoo! Webscope R4 Movie dataset[6] (simply R4). The problem part of each case is described by the combined information of a user and a movie using 11 complicated attributes, and the solution part denotes a rating assigned to the movie by the user. Each user is described by two attributes: birthyear (e.g., 1981) and gender (e.g., m, f). Each movie is represented by nine attributes—title (text), mpaa rating (e.g., PG) (discrete), genres (set-valued), directors (set-valued), actors (set-valued), the average of the critic reviews of the movie (numeric), rating to the movie from the *Movie Mom* website (numeric), gnpp (the global nonpersonalized popularity of the movie) (numeric), and the average rating of the movie in the training data (numeric). Before our experiments, we preprocessed R4 by removing cases that have any attributes with missing values and eliminating redundant attributes (e.g., actors are represented using both actor id and name, thus including only name). Finally, R4 consists of training data having 2229 cases rated by 697 users for 233 movies and testing data having 1754 instances rated by 620 users for 246 movies. The rating of each case is scaled from 1 (lowest) to 5 (highest). Using R4, our target task is to predict a rating that the user will be likely to rate as liked or disliked using the training data for each instance in the testing data. If a predicted rating is greater or equal to 4, we classify it as liked, and disliked otherwise.

2) *Compared Retrieval Methods:* To identify whether USIMSCAR is an effective retrieval strategy, we choose six retrieval methods for comparison purposes. Among them, five are classified into the SBR strategy, and the other one is a retrieval method using a metric combining the concepts of similarity and diversity.

1) KNN: a k-NN implementation using LinearNNSearch (via the Euclidean distance) available in Weka.[7].
2) KNN-CFS: an extension of KNN with an FS approach CfsSubsetEval available in Weka. CfsSubsetEval assesses the predictive ability of each problem feature

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

[5]http://www.hp.com/software

[6]http://research.yahoo.com

[7]http://www.cs.waikato.ac.nz/ml/weka/

TABLE IV
DATASETS USED IN OUR EXPERIMENTS

Experiment Domain	Dataset	Case No.	Attr No.	Problem Part				Solution Part	Experiment Approach
				Numeric	Discrete	Set-Valued	Text	Solution Class no.	
MD	Breast Cancer (BC)	286	9		9			2	Tenfold CV
	Breast Cancer Wins (BCW)	683	9	9				2	
	Breast Tissue (BT)	106	9	9				6	
	Pima Indian Diabetes (PID)	768	8	8				2	
	StatLog Heart Disease (SHD)	270	13	7	6			2	
	New Thyroid (THY)	215	5	5				3	
	NHSG	1085			12			5	
ITSM	IMData (training data)	5211	3		2		1	6	The holdout method (80%: training, 20%: testing)
	IMData (testing data)	1303	3		2		1	6	
PR	R4 (training data)	2229	11	5	2	3	1	2	Given training/testing data from Yahoo! Research
	R4 (testing data)	1754	11	5	2	3	1	2	

individually, and then chooses a set of features highly correlated with the solution.

3) KNN-LVF: an extension of KNN with an FS approach ConsistencySubsetEval available in Weka. It evaluates problem feature sets by the degree of consistency in solution values when stored cases are projected onto the set.
4) KNN-IG: an extension of KNN with a FW approach InfoGainAttributeEval available in Weka. It measures weighting of problem features by measuring their information gain with respect to the solution.
5) KNN-CS: an extension of KNN with a FW approach ChiSquaredAttributeEval available in Weka. It evaluates problem features by computing the chi-square statistic with respect to the solution.
6) SIM-DIV: a retrieval method using the quality metric combining diversity and similarity. The quality of a case is proportional to the similarity between the case and a target problem, and to the diversity of the relative to those cases selected by the similarity [5]. This method is available in jCOLIBRI.[8]

We note that for the IMData/R4 datasets containing text type attributes, we compare USIMSCAR with the following only: KNN, KNN-IG, KNN-CS, and SIM-DIV. KNN-CFS and KNN-LVF are excluded, since CfsSubsetEval and ConsistencySubsetEval cannot handle text values when finding a subset of relevant attributes. For example, suppose two cases contained in IMData. The first case's problem is formed as {request, print issue, I cannot print in Bizflow.} and the solution as WG1. The second case's problem is represented as {request, print issue, user cannot print from any printer.} and the solution as WG1. In this scenario, it is hard to measure what features in values of the third attribute (i.e., incident description) is correlated (or relevant) to the solution since it is described in free text.

Meanwhile, it is possible for the FW methods to compute weights of text attributes of cases via this procedure. First, using the OpenNLP toolkit,[9] we extracted a set of terms from a value of the given text attribute. The attribute is then seen as a set-valued attribute whose values are the set of such terms. Second, we convert the set into k binary attributes, where k is the distinct number of terms in the set. Each of these attributes has a value of 1 for every occurrence of the corresponding kth value of the discrete attribute, and a value of 0 for all other values. These new synthetic attributes are then treated as discrete attributes in a normal manner. Thus, the weights of the individual binary attributes are obtained by applying FW on them. Finally, as a weight of the text attribute, we average the weights of the whole binary attributes.

3) *Evaluation Metrics:* The effectiveness of USIMSCAR and compared methods is measured using the following criterion: how well they perform the prediction of correct solutions for target problems. A simple measure for assessing this performance is to measure the accuracy of them when performing a validation process. Each case in given testing data is used as input to each retrieval method and the output, the predicted solution, is compared to the known solution of the case. In this context, accuracy can be measured via computing the proportion of correctly classified instances over all the tested instances. However, it does not take into account the cost of making wrong decisions. That is, accuracy can be misleading when the testing data contain a disproportional number of cases with a certain solution class. Thus, we also use F-measure (FM) to overcome this problem. It is defined as the harmonic mean between precision (P) and recall (R): $\frac{2PR}{P+R}$, where P means the proportion of the instances, which truly have a solution s, among all those predicted as a solution s. R indicates the proportion of the instances, predicted as a solution s, among all those instances having s. A high FM value indicates that both P and R are reasonably high.

4) *SK Used in the Retrieval Methods:* The SK used in USIMSCAR and compared methods are all encoded via the global–local principle outlined in Section IV-A. That is, the similarity $sim_g(Q, C)$ between a problem Q and a case C is defined as[10]

$$sim_g(Q, C) = \frac{\sum_{i=1}^{n} w_i \cdot sim_l(q_i, c_i)}{\sum_{i=1}^{n} w_i}. \quad (4)$$

[8]http://gaia.fdi.ucm.es/research/colibri/jcolibri
[9]http://opennlp.apache.org/
[10]The notations used in this equation were explained via (1) in Section II.

TABLE V
USED LOCAL SIMILARITY MEASURES

Type	Local Similarity Measures
Numeric	$1 - \frac{\lvert q_i - c_i \rvert}{A_i^{\max} - A_i^{\min}}$ [see (1)]
Discrete	1, if $q_i = c_i$, and 0, otherwise.
Set-valued	The Jaccard coefficient.
Text	Once extracting two sets of terms from q_i and c_i using the OpenNLP toolkit, we compute the Jaccard coefficient between them.

For both KNN-IG and KNN-CS, the value of w_i is computed using FW, while it is set to 1 equally for the other compared methods and USIMSCAR. Table V shows the used local similarities for numeric, discrete, set-valued, and text types, where A_i denotes an attribute. The similarity sim_g is also used for USIMSCAR to find the most similar SCARs to a target problem (see Step 2 of Algorithm 1).

5) *Configuration for Application Tasks:* As mentioned, our application tasks are centered on estimating the correct diagnosis, workgroups, and movie ratings for given illness symptoms, IT incident problems, and user/movie information in the three domains tested, respectively.

From a context of CBR systems, given a new problem, for a retrieval method, such tasks can be carried out by two phases. The first phase is to retrieve a set of relevant cases to the problem through the retrieval process. The second phase is to adapt the solutions of the retrieved cases to generate the correct solution for the problem. This process is called case adaptation, and its process is well-known highly domain-specific and complex in practice.[11] Thus, as a simple scheme of case adaptation, we use a ranking scheme that ranks the solutions of retrieved cases. Here, the ranking is used to generate a form of adaptation knowledge by learning the knowledge already inside the retrieved cases. A well-known approach for the ranking is voting. Thus, we choose two well-known voting approaches, majority voting (MV) and distance weighted voting (WV), that have been widely used in the CBR community.

In a context of USIMSCAR, to apply MV, a vote of an object o_i in a retrieval result **RR** (see Algorithm 2) receives an equal weight. The vote toward a solution Y is computed as $\sum_{i=1}^{|\mathbf{RR}|} \theta(Y, Y_i)/|\mathbf{RR}|$, where $\theta(Y, Y_i)$ returns 1 if Y and o_i's solution Y_i match, and 0 otherwise. For SBR, MV can also be applied to the most similar cases retrieved to a target problem by the same principle. In WV, more competent objects are assigned more weights. Objects in **RR** get to vote on a solution, with votes weighted by their usefulness with respect to the target problem. The vote of o_i toward a solution Y is given as $(\sum_{i=1}^{|\mathbf{RR}|} \theta(Y, Y_i) * o_i.\text{usf})/\sum_{i=1}^{|\mathbf{RR}|} o_i.\text{usf}$. As observed, the vote toward Y is further multiplied by the usefulness of o_i (i.e., o_i.usf). For SBR, WV can also be applied with the same principle except that the usefulness is measured by the similarity to the target problem.

Since the outcomes of the first phase can be influenced by k_{sc} (the number of the retrieved similar cases), we need to test with different values for k_{sc}. For this, we selected odd values in [1, 15] to avoid tied votes (e.g., 1, 3, ..., 15). The maximum value 15 is chosen, since we observed that increasing values for k_{sc} beyond 15 could not change the results. From this range, the results of each retrieval method for the tasks, with the best choice of a value for k_{sc}, are used for comparison purposes. Also, by the same observation, a value for k_{ss} used in Algorithm 2 is chosen from [1, 15].

[11] As case adaption is beyond the scope of this paper, readers are referred to a related survey [30].

TABLE VI
SETTING PARAMETERS FOR USIMSCAR

Parameters	MD Domain	ITSM Domain	PR Domain
minsupp	0.1 (10%)	0.02 (2%)	0.1 (10%)
minsim	0.95 (95%)	0.55 (55%)	0.66 (66%)
minint	0.7 (70%)	0.55 (55%)	0.65 (65%)
minitemsize	$0.5N$	$1.0N$	$0.5N$

6) *Setting Parameters for USIMSCAR:* For SCAR mining, we need to set values for some parameters. A summary of these values used in the experiments is provided in Table VI.

The values are found by the following criteria. A value for minitemsize is set to $0.5N$, if $N \geq 5$, and 1 otherwise, where N is the number of attributes of the cases being considered. A value for minsim is set to the top quartile of similarity scores between instances in the testing data and cases in the training data. The value 0.95 in the MD domain is made by assigning the average of the top-quartile scores from all the seven datasets in the domain. We set values for minsupp and minint by the following scheme. First, a value for minsupp is initially set to 0.1 with a value 0.5 for minint. Then, SCAR mining is performed to generate a set of SCARs. Second, applying the mining continues by increasing a value for minsupp with 0.01 until it reaches up to 0.2 (the maximum for minsupp we defined). Third, for each value for minsupp, we also continue testing with a value for minint by increasing it with 0.05 until it reaches up to 0.7 (the maximum for minint we defined). Using different sets of the SCARs produced with varied values for minsupp and minint, we measure the performance of USIMSCAR and then choose the values leading to the best performance. Regarding minsupp for IMData, we note that minsupp is set to 0.02. Once minsupp is set to 0.1, we found that very few SCARs are generated (< 50). Thus, we initially set the minsupp as a lower value 0.01 and then apply the aforementioned process.

B. Results and Analysis

We first report the number of SCARs generated via SCAR mining in Table VII. For the MD datasets, we use the averaged number of SCARs generated from the use of tenfold CV. Interestingly, the number of SCARs from BT is the highest as 6010, although the number of its instances is the lowest as 106, as shown in Table IV. The reason is found that some items in BT relatively appear frequently compared to the other datasets. We also find that the numbers of SCARs for two large datasets (IMData and R4) correspond to about 30% of the total number of the cases in the training data.

We now discuss experimental results for each experimental domain. First, using the MD datasets, results using MV are

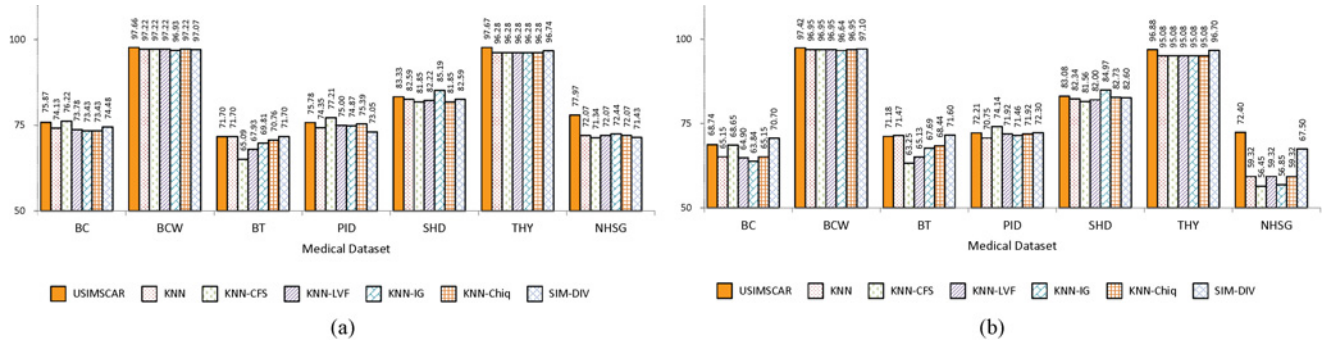

Fig. 1. Results using MV in the MD domain. (a) Accuracy comparison (%). (b) FM comparison (%).

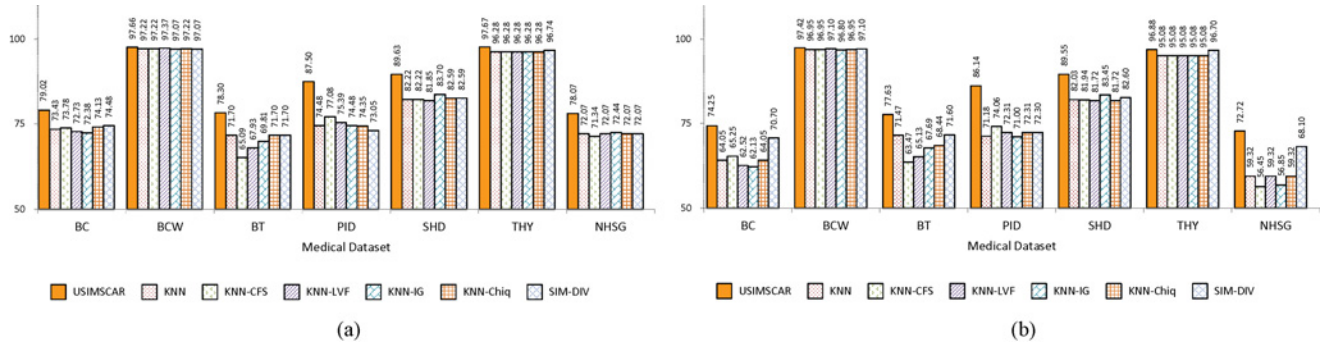

Fig. 2. Results using WV in the MD domain. (a) Accuracy comparison (%). (b) FM comparison (%).

TABLE VII
NUMBER OF GENERATED SCARS

Experimental Domain	Dataset	SCARs No.
MD	BC	228
	BCW	153
	BT	6010
	PID	524
	SHD	676
	THY	1073
	NHSG	673
ITSM	IMData	1568
PR	R4	747

TABLE VIII
STATISTICAL TEST RESULTS: USIMSCAR VERSUS OTHER METHODS

Domain	Dataset	KNN		KNN-CFS		KNN-LVF		KNN-IG		KNN-CS		SIM-DIV	
		MV	WV	MV	WV	MV	WV	MV	WV	MV	WV	MV	WV
MD	BC		○		○		○		○		○		
	BCW												
	BT				●○		○		○				
	PID		●○		●○		●○		●○		●○		●○
	SHD		●○		●○		●○		●○		●○		●○
	THY												
	NHSG	●○	●○	●○	●○	●○	●○	●○	●○	●○	●○	●○	●
ITSM	IMData	●○	●○	–	–	–	–	●○	●○	●○	●○	●○	●○
PR	R4	●	●	–	–	–	–	●	●	●	●	●	●

presented in Fig. 1. Through the figure, we observe that USIMSCAR outperforms the other methods in 39 out of 42 comparisons in terms of both accuracy (ACC) and FM. Only with three comparison occasions for ACC and FM (i.e., KNN-CFS on BC/PID and KNN-CS on SHD in terms of ACC, and KNN on BT, KNN-CFS on PID, and KNN-IG on SHD in terms of FM), USIMSCAR achieves the second best. It shows that USIMSCAR with MV achieves better performance than all compared methods in 92.9% of the comparison cases in terms of the two metrics. The comparison results using WV is found in Fig. 2. Through the results, we now find that USIMSCAR substantially outperforms all compared methods on all the seven datasets in all 42 comparisons in terms of both ACC and FM.

To establish whether the improvements of USIMSCAR using MV and WV against the compared methods are statistically significant, we carried out statistical tests. A common approach for measuring a significant test for a difference between two proportions is the Z-test [31]. We performed statistical tests using the Z-test at 95% confidence. The statistical test results are presented in Table VIII, where ● and ○ indicate that the improvement of USIMSCAR turns out to be statistically significant in terms of ACC and FM, respectively, over the corresponding compared method. Using MV, we find that USIMSCAR's improvements over all the compared methods are significant in 19% comparisons (8 out of 42 comparisons) in terms of both ACC and FM. Using WV, USIMSCAR's improvements are much greater, i.e., 50% and 62% of comparisons are significant in terms of ACC and FM, respectively.

Furthermore, in order to identify whether USIMSCAR's overall performance is significant over the compared methods using all the seven datasets in the MD domain, we performed the paired t-test [31] at 95% confidence. This test is suitable for evaluating the significance of a difference between means

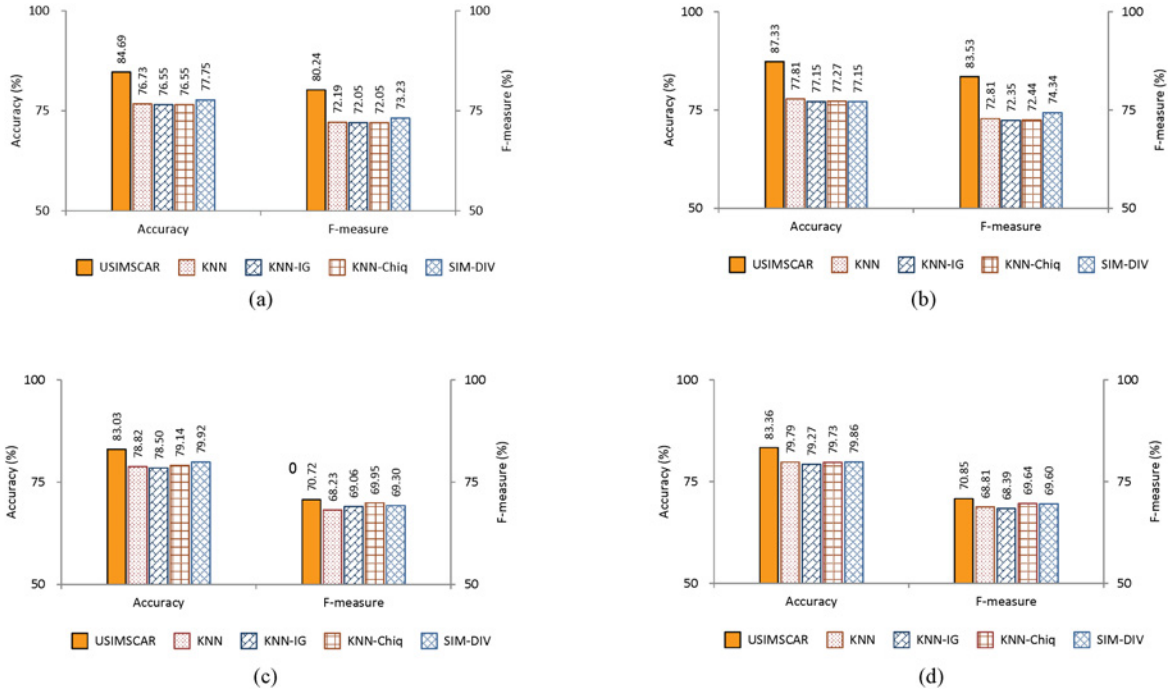

Fig. 3. Results in the ITMS and PR domains. (a) Using MV (with IMData). (b) Using WV (with IMData). (c) Using MV (with R4). (d) Using WV (with R4).

TABLE IX
USIMSCAR'S OVERALL PERFORMANCE IN THE MD DOMAIN

Methods	MV		WV	
	ACC	FM	ACC	FM
USIMSCAR	**82.86**	**80.27**	**86.84**	**84.94**
KNN	81.19	77.29	81.06 •	77.15 ○
KNN-CFS	80.75	76.58	80.43 •	76.17 ○
KNN-LVF	80.64 •	76.47	80.52 •	78.44 ○
KNN-IG	81.28	76.65	80.88 •	76.14 ○
KNN-CS	80.99 •	77.08	81.19 •	76.84 ○
SIM-DIV	81.01	79.79	81.10 •	79.87

of two populations. The test results are shown in Table IX, where each figure denotes the mean of the ACC or FM results of a method against each of the datasets.

Through the tests, we discover that USIMSCAR shows significant improvements over two methods in terms of ACC with MV, while its improvements are much greater with WV, i.e., outperforming all the compared methods in ACC, and five methods in FM. We note that the lack of significant improvements of USIMSCAR does not necessarily mean that there is no difference between them. As Keen [32] indicated, such improvements still can be important if these occur repeatedly in many contexts. Thus, our comparison results may be still valuable, since these provide evidence about a better performance of USIMSCAR over SBR using a number of datasets. Through the aforementioned results, we have shown that USIMSCAR has a greater ability compared to SBR and SIM-DIV in achieving the retrieval process in CBR.

Now, we analyze experimental results of USIMSCAR and the four compared methods for the ITSM domain. Through Fig. 3(a), we find that USIMSCAR substantially outperforms the compared methods in the range of 6.9% to 8.1% in terms of ACC and also the range of 7.0% to 8.2% in terms of FM. As observed in Table VIII, USIMSCAR is determined to attain statistically significant improvements over the compared methods in terms of both metrics. Fig. 3(b) shows the comparison results with the use of WV. As observed, USIMSCAR greatly outperforms the compared methods in the range 9.5% and 10.2% in terms of ACC, and in the range of 9.2% to 11.2% in terms of FM. According to the Z-test, we find that USIMSCAR significantly outperforms the methods in terms of both metrics as shown in Table VIII.

Next, we focus on experimental results using the R4 dataset in the PR domain. The results are given in Fig. 3(c) and (d). As observed in Fig. 3(c), USIMSCAR outperforms all the four methods, ranging from 3.2% to 4.6% in terms of ACC, and 0.77% to 2.5% in terms of FM. These improvements are deemed significant as seen in Table VIII in terms of ACC. Regarding WV, as observed in Fig. 3(d), USIMSCAR significantly outperforms the compared methods, ranging from 3.57% to 4.08% in terms of ACC, and 1.21% to 2.45% in terms of FM. We find that these improvements are also significant in terms of ACC as shown in Table VIII.

A noticeable observation in Fig. 3(c) and (d) is that the ACC results of the methods are relatively higher than their corresponding FM results. The differences are identified as from 9.1% to 12.5%. We find that such differences occurred due to the much smaller number of instances (17.7%) belonging to the solution (disliked) (ratings 1–3), compared to the solution (liked) (ratings 4 and 5). Such lower associations between disliked and the instances entail that the majority of solution prediction tend to be highly to liked, leading to the lower FM results for disliked. Such lower results influence the low mean FM results for both liked and disliked.

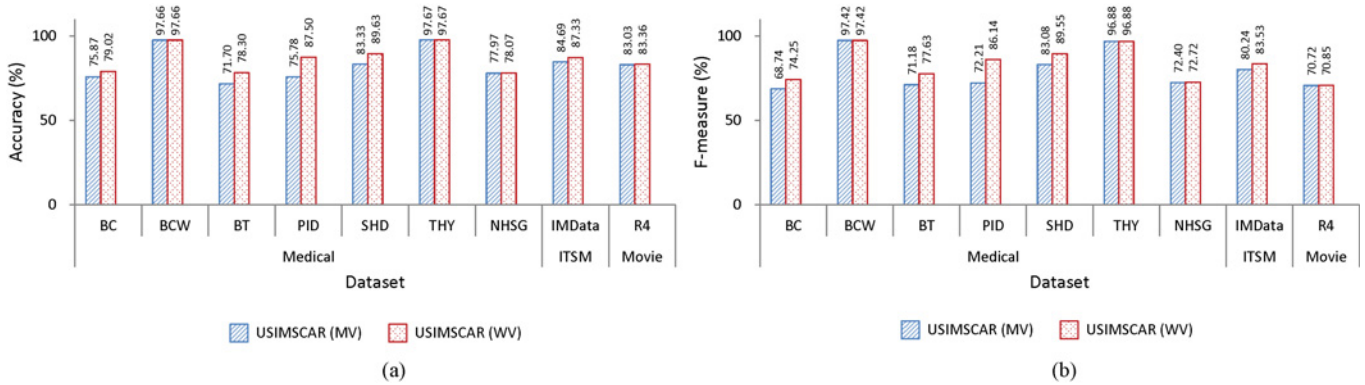

Fig. 4. USIMSCAR (MV) versus USIMSCAR (WV). (a) ACC comparison (%). (b) FM comparison (%).

Through the evaluation results in the ITSM and PR domains, we find that using both MV and WV, USIMSCAR significantly outperforms all the four compared methods for ACC. With results for FM, USIMSCAR's improvements are significant in the ITSM domain. These results reinforce that USIMSCAR has a powerful ability in achieving the retrieval process, in addition to the results in the MD domain.

The following examinations can be conclusively found in our paper.

1) Our all experimental results in the three domains (i.e., MD, ITSM, and PR) show that USIMSCAR significantly outperforms the compared SBR approaches in 36.7% and 38.9% comparisons in terms of ACC and FM, respectively. Against SIM-DIV, USIMSCAR's statistical improvement ratio reaches 44.4% and 22.2% for ACC and FM, respectively. This can provide an evidence that USIMSCAR is a powerful and effective retrieval strategy.
2) In particular, the real strength of our evaluation lies in the fact that USIMSCAR outperforms SBR and SIM-DIV for predicting solutions against different forms of problems described by various types (i.e., numeric, discrete, set-valued, and text) across multiple domains using both real-world (e.g., NHSG, IMData, and R4) and benchmark data.
3) Furthermore, we discovered that USIMSCAR using WV consistently leads to better performance over using MV in terms of both ACC and FM. Their comparison results are shown in Fig. 4. According to the paired t-test at 95% confidence, we find that the improvement of USIMSCAR using WV turns to be significant. This finding implies that it is more significant to utilize the quantified usefulness of objects in the retrieval set of USIMSCAR, rather than merely using distribution information about the solutions associated with such objects.
4) The results shown in Fig. 4 further establishes the validity of the primary motivation of this paper that a combination of SK and AK will lead to enhancing SBR. Conclusively, this finding provides the evidence of the validity and soundness of our proposed USIMSCAR approach for retrieval in CBR.

C. Discussion

The objective of this paper is to design a new retrieval strategy to enhance the typically used retrieval strategy SBR. As shown by the evaluation results, USIMSCAR holds a potential to be used for retrieving relevant cases as well as evidently promising rules in practice and to subsequently improve the retrieval performance. A practical application of USIMSCAR may be to enhance the retrieval strategy implemented in currently available CBR systems that are publicly available (e.g., jCOLIBRI, myCBR[12]).

USIMSCAR may be usefully leveraged in domains (e.g., medical domains) where case retrieval is forming greater part of the overall CBR systems than case adaptation to present solutions to users. For example, in medical applications, it is almost impossible to generate adaptation rules by considering all possible important differences between the current and former similar cases [15]. Thus, case retrieval tends to be more interested in making optimal solutions in such applications. USIMSCAR may be best fitted to needs for such applications.

In practice, USIMSCAR is possibly used for other CBR applications, as long as a dataset in a given application can be represented using attribute–values pairs. USIMSCAR is not affected by the types of each attribute—whether these are primitive (e.g., numeric, discrete), more complex (e.g., set-valued, text) or semantic (e.g., concepts in an ontology), as long as a similarity metric is provided for the attribute. Since many similarity metrics for different attribute types have been actively developed in IR, we may easily integrate them into USIMSCAR depending upon the application context. For example, those similarity metrics for various attribute types provided from the jCOLIBRI and Weka toolkits may be good potentials to be leveraged in USIMSCAR.

We recall that SCAR mining and USIMSCAR work with four user-specified parameters: minitemsize, minsim, minsupp, and minint. In the DM community, choosing optimal values for minsupp and minconf (the similar concept to minint) is still challenging, since these depend on the data. Thus, many DM applications still choose such values in ad hoc manners. Although, in our experiments, we also chose values

[12]http://mycbr-project.net/

for the aforementioned parameters in ad hoc manners, we would suggest one approach based on machine learning as follows. First, from a randomly selected training dataset from a given case base, we could define the lowest and highest boundaries of allowed values for each parameter. Second, we could then start with a low value to higher gradually or vice versa. After that, we can measure the performance of USIMSCAR. Third, a regression model could be learned to identify how each combination of a set of particular values across the parameters is correlated to a certain performance degree of USIMSCAR. The model could be used to catch a good picture on how to optimally combine values for the four parameters. We leave the validation of this idea as future work.

Finally, we point out that AK is different from adaptation knowledge leveraged for the solution transformation of retrieved cases in the following aspects.

1) Goal: while adaptation knowledge generally concerns substituting or transforming specific solutions of retrieved cases to suit new problems, AK concerns finding the general patterns of evident associations between known problems and solutions from the case base.
2) Target Cases: whereas adaptation knowledge is generally required to use for retrieved cases, AK is used to retrieve relevant cases and rules with respect to new problems.
3) Knowledge Characteristic: adaptation knowledge is highly domain specific so that it is difficult to provide a general guideline for formulating this knowledge.

On the other hand, AK can be easily built from the case base and thus is case specific. However, AK may be possibly used to enhance adaptation knowledge or be integrated with it for the better use of adaptation rules.

VIII. Conclusion and Future Work

This paper presented a novel strategy USIMSCAR that can be effectively used for retrieval in CBR. The paper made two main contributions. First, we proposed a unique approach for extracting and representing AK from a given case base. We proposed that this knowledge is encoded via SCARs using ARM. Second, we proposed novel strategies for quantifying the usefulness of cases as well as rules with respect to the target problem by leveraging both SK and AK. This quantification is used in USIMSCAR to determine the most relevant cases and rules with respect to the problem. This idea of leveraging the combined knowledge during CBR retrieval clearly distinguishes USIMSCAR from SBR as well as existing retrieval strategies developed in the CBR research. We showed the significance of USIMSCAR over SBR and a retrieval strategy adopting the similarity and diversity metrics through extensive experiments in various CBR domains.

As future work, USIMSCAR could also be extended for cases with complex structures such as object-oriented, hierarchical, and semantic web-based cases [2], [33]. For USIMSCAR to run with such cases, two issues must be addressed: 1) how to define similarity measures for the cases; and 2) how to formalize AK from the cases. For the former, one may use the similarity approaches remarked in [34] for these cases. For the latter, one could attempt to leverage the algorithms proposed in [33], [35], and [36], according to the given case representations, to integrate the soft-matching criterion for SCAR mining. Also, research on how to adapt USIMSCAR for cases with more than one solution could be further studied. In addition, the performance of USIMSCAR in other metrics (i.e., computation time and memory used) will be investigated further.

References

[1] A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches," *AI Commun.*, vol. 7, pp. 39–59, Mar. 1994.

[2] R. Lopez De Mantaras, D. McSherry, D. Bridge, D. Leake, B. Smyth, S. Craw, B. Faltings, M. L. Maher, M. T. Cox, K. Forbus, M. Keane, A. Aamodt, and I. Watson, "Retrieval, reuse, revision and retention in case-based reasoning," *Knowl. Eng. Rev.*, vol. 20, no. 3, pp. 215–240, 2005.

[3] Y. Guo, J. Hu, and Y. Peng, "Research on CBR system based on data mining," *Appl. Soft Comput.*, vol. 11, no. 8, pp. 5006–5014, 2011.

[4] Y.-J. Park, E. Choi, and S.-H. Park, "Two-step filtering datamining method integrating case-based reasoning and rule induction," *Expert Syst. Appl.*, vol. 36, no. 1, pp. 861–871, 2009.

[5] B. Smyth and P. McClave, "Similarity vs. diversity," in *Case-Based Reasoning Research and Development*. Berlin, Germany: Springer-Verlag, 2001, pp. 347–361.

[6] J. L. Castro, M. Navarro, J. M. Sánchez, and J. M. Zurita, "Loss and gain functions for CBR retrieval," *Inf. Sci.*, vol. 179, no. 11, pp. 1738–1750, 2009.

[7] H. Ahn and K.-J. Kim, "Global optimization of case-based reasoning for breast cytology diagnosis," *Expert Syst. Appl.*, vol. 36, no. 1, pp. 724–734, 2009.

[8] B. Pandey and R. Mishra, "Case-based reasoning and data mining integrated method for the diagnosis of some neuromuscular disease," *Int. J. Med. Eng. Informat.*, vol. 3, no. 1, pp. 1–15, 2011.

[9] Y.-B. Kang, A. Zaslavsky, S. Krishnaswamy, and C. Bartolini, "A knowledge-rich similarity measure for improving IT incident resolution process," in *Proc. ACM Symp. Appl. Comput.*, 2010, pp. 1781–1788.

[10] F. Lorenzi and F. Ricci, "Case-based recommender systems: A unifying view," in *Intelligent Techniques for Web Personalization*, vol. 3169. Berlin, Germany: Springer, 2005, pp. 89–113.

[11] G. R. Beddoe and S. Petrovic, "Selecting and weighting features using a genetic algorithm in a case-based reasoning approach to personnel rostering," *Eur. J. Oper. Res.*, vol. 175, no. 2, pp. 649–671, 2006.

[12] K. Bradley and B. Smyth, "Personalized information ordering: A case study in online recruitment," *Knowl.-Based Syst.*, vol. 16, nos. 5–6, pp. 269–275, 2003.

[13] C. M. Vong, P. K. Wong, and W. F. Ip, "Case-based classification system with clustering for automotive engine spark ignition diagnosis," in *Proc. 9th Int. Conf. Comput. Inf. Sci.*, Aug. 2010, pp. 17–22.

[14] F. Azuaje, W. Dubitzky, N. Black, and K. Adamson, "Discovering relevance knowledge in data: A growing cell structures approach," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 30, no. 3, pp. 448–460, Jun. 2000.

[15] Z. Y. Zhuang, L. Churilov, F. Burstein, and K. Sikaris, "Combining data mining and case-based reasoning for intelligent decision support for pathology ordering by general practitioners," *Eur. J. Oper. Res.*, vol. 195, no. 3, pp. 662–675, 2009.

[16] C.-L. Chuang, "Case-based reasoning support for liver disease diagnosis," *Artif. Intell. Med.*, vol. 53, no. 1, pp. 15–23, 2011.

[17] Y.-J. Park, B.-C. Kim, and S.-H. Chun, "New knowledge extraction technique using probability for case-based reasoning: Application to medical diagnosis," *Expert Syst.*, vol. 23, no. 1, pp. 2–20, 2006.

[18] H. Ahn and K.-j. Kim, "Using genetic algorithms to optimize nearest neighbors for data mining," *Ann. Oper. Res.*, vol. 163, no. 1, pp. 5–18, 2008.

[19] N. Cercone, A. An, and C. Chan, "Rule-induction and case-based reasoning: Hybrid architectures appear advantageous," *IEEE Trans. Knowl. Data Eng.*, vol. 11, no. 1, pp. 166–174, Jan.–Feb. 1999.

[20] M.-J. Huang, M.-Y. Chen, and S.-C. Lee, "Integrating data mining with case-based reasoning for chronic diseases prognosis and diagnosis," *Expert Syst. Appl.*, vol. 32, no. 3, pp. 856–867, 2007.

[21] A. Stahl, "Learning of knowledge-intensive similarity measures in case-based reasoning," Ph.D. dissertation, Artificial Intelligence Knowledge-Based Systems Research Group, Tech. Univ. Kaiserslautern, Kaiserslautern, Germany, 2003.
[22] D. Wang, T. Li, S. Zhu, and Y. Gong, "ihelp: An intelligent online helpdesk system," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 1, pp. 173–182, Feb. 2011.
[23] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *ACM SIGMOD Rec.*, vol. 22, no. 2, pp. 207–216, Jun. 1993.
[24] B. Liu, W. Hsu, and Y. Ma, "Integrating classification and association rule mining," in *Knowledge Discovery and Data Mining*. Berlin, Germany: Springer, 1998, pp. 80–86.
[25] U. Y. Nahm and R. J. Mooney, "Using soft-matching mined rules to improve information extraction," in *Proc. AAAI Workshop Adaptive Text Extract. Mining*, 2004, pp. 27–32.
[26] L. Geng and H. J. Hamilton, "Interestingness measures for data mining: A survey," *ACM Comput. Surv.*, vol. 38, no. 3, Article 9, Sep. 2006.
[27] R. J. Bayardo, Jr., and R. Agrawal, "Mining the most interesting rules," in *Proc. 5th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 1999, pp. 145–154.
[28] T. Hu, S. Y. Sung, H. Xiong, and Q. Fu, "Discovery of maximum length frequent itemsets," *Inf. Sci.*, vol. 178, pp. 69–87, Jan. 2008.
[29] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proc. 14th Int. Joint Conf. Artif. Intell.*, Aug. 1995, pp. 1137–1143.
[30] R. Mitra and J. Basak, "Methods of case adaptation: A survey: Research articles," *Int. J. Intell. Syst.*, vol. 20, no. 6, pp. 627–645, Jun. 2005.
[31] R. C. Sprinthall, *Basic Statistical Analysis*. Boston, MA, USA: Allyn & Bacon, 2003.
[32] E. M. Keen, "Presenting results of experimental retrieval comparisons," *Inf. Process. Manage.*, vol. 28, no. 4, pp. 491–502, 1992.
[33] V. Nebot and R. Berlanga, "Mining association rules from semantic web data," in *Proc. 23rd Int. Conf. Ind. Eng. Appl. Appl. Intell. Syst.*, 2010, pp. 504–513.
[34] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *Int. J. Math. Models Methods Appl. Sci.*, vol. 1, no. 4, pp. 300–307, 2007.
[35] P. Kuba and L. Popelinsky, "Mining frequent patterns in object-oriented data," in *Proc. 2nd Int. Ws Mining Graphs Trees Sequences (ECML/PKDD)*, Pisa, 2004, pp. 15–25.
[36] P. Gautam and K. R. Pardasani, "Algorithm for efficient multilevel association rule mining," *Int. J. Comput. Sci. Eng.*, vol. 2, no. 5, pp. 1700–1704, 2010.

Yong-Bin Kang received the M.Sc. degree in computer science from Pusan National University, Busan, Korea, in 1998, and the Ph.D. degree in information technology from Monash University, Melbourne, Australia, in 2011.

He is currently a Research Fellow with the Faculty of Information Technology, Monash University. He is currently involved in research projects that include developing innovative models for predicting the performance of semantic reasoners inferencing large/hard ontologies, effective models for quantifying research expertise of experts in different domains, and knowledge models for supporting emergency management for mass gatherings. Prior to his Ph.D., he has been for around 10 years at various research positions in IT development companies and premier research organizations in Korea such as Electronics and Telecommunications Research Institute from 1998 and 2007. During this time, he conducted high-quality research, contributing to a number of conference/journal papers in various areas, such as human–computer interface, virtual reality, and location-based tracking, as well as developed a number of real-world systems for different application domains such as traffic management, warehouse management, insurance, document management, and text recognition systems. His current research is focused highly on interdisciplinary research based on machine learning, data mining, semantic web, and bioinformatics.

Shonali Krishnaswamy received the Bachelor of Science degree in computer science from the University of Madras, Chennai, India, in 1996, and the Master of Computing degree and the Ph.D. degree in computer science from Monash University, Australia, in 1998 and 2003, respectively.

She is currently the Head of the Data Analytics Department, Institute for Infocomm Research (I2R), A*STAR, Singapore. She was the Director of the Centre for Distributed Systems and Software Engineering, Faculty of Information Technology, Monash University, Melbourne, Australia, where she is currently an Associate Professor. She has contributed to around 200 research papers. Her current research interests include the areas of mobile, ubiquitous, distributed data mining, and data stream mining. She is increasingly interested in mobile crowd sensing, mobile user analytics, and mobile activity recognition.

Prof. Krishnaswamy was the recipient of the Monash University Vice-Chancellor's Award for Excellence in Research by an Early Career Researcher 2008, the IBM Innovation Award (unstructure information management architecture), the Faculty of Information Technology Early Career Researcher Award, and an Australian Post-Doctoral Fellowship from the Australian Research Council.

Arkady Zaslavsky received the M.Sc. degree in applied mathematics (majoring in computer science) from Tbilisi State University, Tbilisi, Georgia, in 1976, and the Ph.D. degree in computer science from the Moscow Institute for Control Sciences, USSR Academy of Sciences, Moscow, Russia, in 1987.

He is currently the Science Leader of the semantic data management science area in Information Engineering Laboratory, Information and Communication Technologies Centre, Commonwealth Scientific and Industrial Research Organisation (CSIRO), Melbourne, Australia. He is also an Adjunct Professor at Australian National University, a Research Professor at La Trobe University, and an Adjunct Professor at the University of New South Wales. He is currently involved and is leading a number of European and national research projects. Before joining CSIRO in July 2011, he was a Chaired Professor in pervasive and mobile computing at the Lulea University of Technology, Lulea, Sweden, where he was involved in a number of European research projects, collaborative projects with Ericsson Research, Ph.D. supervision, and postgraduate education. Between 1992 and 2008, he was a full-time Academic Staff Member at Monash University, Melbourne, Australia. He made internationally recognized contribution in the area of disconnected transaction management and replication in mobile computing environments, context-awareness, as well as in mobile agents. He made significant internationally recognized contributions in the areas of data stream mining on mobile devices, adaptive mobile computing systems, ad hoc mobile networks, efficiency and reliability of mobile computing systems, mobile agents, and mobile file systems. Before coming to Australia in 1991, he was in various research positions at industrial research and development labs as well as at the Institute for Computational Mathematics, Georgian Academy of Sciences, where he led a systems software research laboratory. He has published more than 300 research publications throughout his professional career and supervised to completion more than 30 Ph.D. students.

Dr. Zaslavsky is a Senior Member of the Association for Computing Machinery, and a member of the IEEE Computer and Communication Societies.