A Retrieval Strategy Using the Integrated Knowledge of Similarity and Associations

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

[1] Faculty of Information Technology,
Monash University, Australia
{yongbin.kang,shonali.krishnaswamy}@monash.edu
[2] Department of Computer Science and Electrical Engineering,
Luleå University of Technology, Sweden
arkady.zaslavsky@ltu.se

Abstract. Retrieval is often considered the most important task in Case-Based Reasoning (CBR), since it lays the foundation for overall performance of CBR systems. In CBR, a typical retrieval strategy is realized through similarity knowledge encoded in similarity measures. This strategy is often called similarity-based retrieval (SBR). This paper proposes and validates that association analysis techniques can be used to improve SBR. We propose a retrieval strategy USIMSCAR that performs the retrieval task by integrating similarity and association knowledge. We show its reliability, in comparison with several retrieval methods implementing SBR, using datasets from UCI ML Repository.

1 Introduction

Retrieval is a very important task in CBR, since it lays the foundation for overall performance of CBR systems [1]. The goal of this task is to retrieve *useful* cases that can be successfully reused to solve a new problem. If retrieved cases are not useful, CBR systems will not eventually produce any good solution for the new problem. For the retrieval task, CBR systems are typically reliant on a specific strategy that exploits *similarity knowledge*. This strategy is thus often called *similarity-based retrieval* (SBR) [2]. In SBR, similarity knowledge represents a heuristic for approximating the usefulness of stored cases [3]. This knowledge is usually encoded in *similarity measures*. By using similarity knowledge, SBR retrieves useful cases, ranked by their similarities to a new problem. Then, solutions, of the top ranked cases, are used to solve the new problem.

However, the main limitation of SBR is that it cannot *guarantee* that cases, retrieved through similarity knowledge, are sufficiently useful to solve a new problem. This limitation is rooted in the fact that SBR only considers the *problem space*−a set of problems−in a given case base, when formalizing similarity knowledge. This leads to trouble. The reason is that determining the usefulness of stored cases cannot be completed without considering how known problems are actually *associated* with specific known solutions. SBR thus cannot retrieve

J.X. Yu, M.H. Kim, and R. Unland (Eds.): DASFAA 2011, Part II, LNCS 6588, pp. 16–30, 2011.

those useful cases, whose solutions have stronger associations with the problems of certain cases similar to the new problem.

The goal of this work is to improve SBR by incorporating *association analysis* techniques into retrieval in CBR. For this purpose, we propose a new retrieval strategy USIMSCAR that exploits both similarity knowledge and association knowledge. The aim of *association knowledge* is to represent implicit and potentially useful *associations* (dependencies), between problems and solutions, observed in stored cases. More precisely, this knowledge models a set of highly correlated attribute-value pairs, of problems and solutions, shared by a large number of stored cases. To formalize this knowledge we use *association rule mining* techniques. The key idea of USIMSCAR is the exploitation of association knowledge, ignored in SBR, in conjunction with similarity knowledge, to provide a more complete strategy for retrieval in CBR.

This paper is organized as follows. In Section 2, we provide an overview of SBR, and the main problem of SBR. In Section 3, we introduce a well-known principle that formalizes similarity knowledge, and describe association analysis techniques used for formalizing association knowledge. In Section 4, we present our association knowledge formalism, and the USIMSCAR algorithm. In Section 5, we evaluate USIMSCAR, using 6 datasets found from UCI ML Repository[1], in comparison with 5 retrieval methods implementing SBR. In Section 6, we review related work. In Section 7, we finally conclude this paper with future work.

2 Similarity-Based Retrieval and Its Main Problem

Similarity-based retrieval (SBR) is typically implemented through the technique using a derivative of the *nearest neighbor* algorithm [4,1]. This technique is called *k-nearest neighbor retrieval* or simply k-NN [1]. The idea of k-NN is that, to solve a new problem, useful cases are determined using its k most similar cases (i.e. nearest neighbors). Here, similarity is used to represent a heuristic for estimating such cases. Thus, similarity is the most important aspect in k-NN.

Let a case base $\mathcal{D}$ be a set of cases. These cases (including a new problem q) are described by m (numeric and discrete) attributes $A_1, ..., A_m$. Assume that any numeric attributes have been normalized to the range [0,1]; and each case is labeled with a solution label $y \in Y$. Our aim here is to assign an appropriate solution label to q. For this purpose, k-NN first determines the nearest neighbors (i.e. similar cases) of q by using a distance metric. The standard for this metric is the Euclidean distance [5]. For each case $c \in \mathcal{D}$, the Euclidean distance $DIST(q, c)$, between q and c, is represented as

$$DIST(q,c) = \sqrt{\sum_{i=1}^{m} dist(q_i, c_i)^2}, \quad dist(q_i, c_i) = \begin{cases} |q_i - c_i|, \text{ if } A_i \text{ is numeric,} \\ 0, \text{ if } A_i \text{ is discrete \& } q_i = c_i, \\ 1, \text{ otherwise,} \end{cases} \tag{1}$$

where q_i and c_i denote the values of the A_i of q and c, respectively, and $dist(q_i, c_i)$ represents their distance.

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

The distance metric $DIST(q, c)$ has the merit that it allows knowledge to be brought to bear on the assessment of similarity−the nearer two objects are, the more similar they are. In the rest of this paper, we thus consider similarity only to find the nearest neighbors of q. Once the neighbors are selected, there are various ways for determining a solution of q. The simplest approach is to choose the majority solution among the neighbors, called *majority voting* [6].

Over the years, researchers have widely studied k-NN to improve its performance in terms of accuracy. For example, its sensitiveness to k is overcome by determining a best k through a learning technique such as *cross-validation* [5]. Feature selection [7] is a good technique that determines a subset of relevant features (attributes) among the original features of stored cases. Feature weighting [8] is also a useful technique, in which each feature (attribute) is multiplied by a weight. The weight is usually determined by considering the ability of the feature in distinguishing solution labels. In this work, we categorize k-NN and its extensions, integrated using the above enhancements, into representative techniques (or models) implementing SBR.

Problem Statement. We now present a main limitation of SBR. To illustrate, we choose an extension of k-NN as a representative model implementing SBR. Assume that this model is made by integrating k-NN and feature weighting, and denoted as $\mathcal{Z}$. Consider a medical diagnosis scenario[2], where 5 patient cases are stored in a case base $\mathcal{D}$ (See Table 1a). Each case consists of 5 symptoms (attributes) $A_1, ..., A_5$ and the corresponding diagnosed disease (solution).

Table 1. A patient case base and similarity results

(a) A patient case base

Patients	Local Pain	Other Pain	Fever	Appetite Loss	Age	Diagnosis
p_1	right flank	vomit	38.6	yes	10	appendicitis
p_2	right flank	vomit	38.7	yes	11	appendicitis
p_3	right flank	vomit	38.8	yes	13	appendicitis
p_4	right flank	sickness	37.5	yes	35	gastritis
p_5	epigastrium	nausea	36.8	no	20	stitch
q	right flank	nausea	37.8	yes	14	?
Weight	0.91	0.78	0.60	0.40	0.20	

(b) Similarity results

$SIM(q, p_1) = 0.631$
$SIM(q, p_2) = 0.623$
$SIM(q, p_3) = 0.618$
$SIM(q, p_4) = \mathbf{0.637}$
$SIM(q, p_5) = 0.420$

Our aim is to diagnose the correct disease of a new patient q. $\mathcal{Z}$ achieves this goal by identifying the k most similar cases to q. To find them, it selects cases whose symptoms are similar to q, using a similarity metric. Assume that we use the following metric[3] to measure the similarity between q and each case $p_k \in \mathcal{D}$:

$$SIM(q, p_k) = \frac{\sum_{i=1}^{m} w_i \cdot sim(q_i, p_{ki})}{\sum_{i=1}^{m} w_i}, \quad sim(q_i, p_{ki}) = \begin{cases} 1 - \frac{|q_i - p_{ki}|}{A_i^{\max} - A_i^{\min}}, & \text{if } A_i \text{ is numeric,} \\ 1, & \text{if } A_i \text{ is discrete \& } q_i = p_{ki}, \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

[2] This scenario is a simple modification of the scenario found in the work [9].
[3] This similarity metric is the same one used in the work [9].

where w_i[4] is a weight of A_i; q_i and p_{ki} are the values of the A_i of, q and p_k, respectively; m is 5; $sim(q_i, p_{ki})$ is the similarity between q_i and p_{ki}; and $A_i^{\max}$ and $A_i^{\min}$ are the "max" and "min" values of A_i, respectively, in all the cases.

Using the metric $SIM(q, p_k)$, assume that $\mathcal{Z}$ chooses the single most similar case to q. As seen in Table 1b, we then choose the most similar case to q as p_4. This means that a diagnosis for q is chosen as 'gastritis'. However, this turns out to be wrong, since q is actually identified to suffer from 'appendicitis'[5]. This wrong diagnosis may lead to a serious error for q. If the disease 'appendicitis', that q really has, is not treated correctly, q's health may be endangered.

The scenario clearly shows that SBR has a significant limitation, rooted in the fact that SBR is strongly based on similarity knowledge. To overcome the limitation, a potential idea is to exploit the knowledge of how certain attribute-value pairs of known problems are *associated* with specific known solutions in $\mathcal{D}$. With the above scenario, we may obtain the following knowledge: the attribute-value pairs of p_1, p_2 and p_3 have a strong *association* with 'appendicitis', and those of p_4 with 'gastritis'. The strength of the former association, denoted as A_1, may be higher than that of the latter association, denoted as A_2. Because A_1 is supported by three cases, while A_2 is supported by only one case. If these associations were to be appropriately quantified and integrated with the similarity results in Table 1b, a diagnosis for q may be more accurately determined. This is the fundamental idea underlying USIMSCAR.

3 Similarity Knowledge and Association Analysis

We now present a similarity knowledge formalism, and the association analysis techniques used for building association knowledge. Before this, we first give the case model that is the basis for formalizing these two kinds of knowledge.

To represent cases, CBR systems often adopt well-known knowledge representation formalisms, such as attribute-value pairs or object-oriented representation [1]. We adopt the attribute-value pairs representation, due to its flexibility and popularity [3]. An *attribute-value pair* is represented as the form of (A_i, a_i), where A_i is an attribute (or feature[6]) and a_i is a value of A_i. Let a case be characterized by $m+1$ attributes $A_1, ..., A_m, A_{m+1}$ in a domain T. Let $\mathcal{P}$ be the problem space, a set of potential problems in T, where each problem $x \in \mathcal{P}$ is characterized by $A_1, ..., A_m$. Let $\mathcal{S}$ be the solutions space, a set of potential solutions in T, where each solution $s \in \mathcal{S}$ is characterized by A_{m+1}. We call A_{m+1} *solution-attribute*. A *case* is then defined as a pair (x, s_x), where x is a problem, $x = \{a_1, ..., a_m\} \in \mathcal{P}$, and s_x is a solution of x, $s_x = a_{m+1} \in \mathcal{S}$. For the sake of simplicity, we assume that x is associated with a unique solution s_x. In Section 7, we remark that USIMSCAR can be extended to the case, where a problem is described by more complex structures, and a problem is associated with more than one solution.

[4] The weight is borrowed from the work [9] and assigned by the domain expert.
[5] This fact is cited from the work [9].
[6] To simplify the presentation, we do not distinguish between terms attribute and feature.

3.1 Similarity Knowledge

Similarity knowledge is referred to as the knowledge encoded in similarity measures. SBR often uses a principle that models the similarity measures, suitable for the attribute-value pairs representation. It is *local-global principle* that decomposes an entire similarity computation by *local similarities* for individual attributes and a *global similarity* that aggregates these local similarities [3]. An accurate definition of local similarities relies on attribute types. An example of a similarity measure, based on this principle, is seen in Equation 2: "SIM" is a global similarity measure, and "sim" includes three local similarity measures.

3.2 Association Analysis

From the CBR viewpoint, *association rule mining* [10] is concerned about mining a set of highly correlated "attribute-value pairs of problems" and "solutions", shared by a large number of stored cases.

To formalize association knowledge, we build named *soft-matching class association rules* (scars) by using association rule mining techniques. A scar is a *class association rule* (car) [11] whose *antecedent* and *consequent* are generated by applying the *soft-matching criterion* [12]. A car is a special form of an *association rule*. Hence, we give an overview of association rules, cars, and the soft-matching criterion, involved in the formalization of scars.

• *association rules* [10]: Let $\mathcal{D}$ be a set of cases. Each case $T \in \mathcal{D}$ is characterized by attributes $A_1, ..., A_m, A_{m+1}$, where the problem is characterized by $A_1, ..., A_m$, and the solution by A_{m+1}. We call a pair $(A_i, a_i)_{1 \leq i \leq m+1}$ an *item*. Let I be a set of items. A set $X \subseteq I$, with $k = |X|$, is called a k-itemset or simply an itemset. We say that a case $T \in \mathcal{D}$ *supports* an itemset $X \subseteq I$, if $X \subseteq T$ holds. An *association rule* has two parts, *antecedent* and *consequent*, and denoted as $X \rightarrow Y$. Here, X is an itemset in the antecedent and Y is an itemset in the consequent, and $X \cap Y = \emptyset$ holds. The fraction of cases that support an itemset X in $\mathcal{D}$ is called the *support* of X, $supp(X) = |\{T \in \mathcal{D} | X \subseteq T\}| / |\mathcal{D}|$. The *support* of $X \rightarrow Y$ is defined as the probability that both X and Y occur together in a case T, $supp(X \rightarrow Y) = supp(X \cup Y)$. The *confidence* of $X \rightarrow Y$ is defined as $conf(X \rightarrow Y) = supp(X \cup Y)/supp(X)$. We say that an itemset X is *frequent*, if $supp(X) \geq$ minsupp (a user-specified minimum support). Apriori [10] is a representative algorithm widely used for association rule mining.

• *class association rules* (cars) [11]: A special subset of association rules, whose consequents are restricted to a single target, is called cars. From the CBR perspective, the solution-attribute (A_{m+1}) can become the target. We call a pair $(A_i, a_i)_{1 \leq i \leq m}$ an *item*, and call a pair (A_{m+1}, a_{m+1}) a *s-item*. Let I be a set of items, and SI be a set of s-items. A car is then an implication of the form $X \rightarrow s$, where X is an itemset $X \subseteq I$ and $s \in SI$ is a s-item.

• *soft-matching criterion* [12]: To discover frequent itemsets F, traditional association rule mining algorithms (e.g. Apriori) consider only itemsets that *exactly* match F. However, when treating attribute values, semantically related to each other, these algorithms may perform poorly. Because they ignore *semantic*

relevance between those values. For example, they cannot find a rule like "80% of the customers, who buy milk-related (e.g. cheese) products and eggs-related (e.g. mayonnaise) products, also buy bread." To address this issue, the SoftApriori algorithm is proposed by [12]. SoftApriori uses *soft-matching* criterion, in which frequent itemsets are found by similarity assessment between itemsets.

4 Modeling Association Knowledge and USIMSCAR

We now propose an approach to formalizing association knowledge used in USIMSCAR. As mentioned above, this knowledge is encoded as scars, generated from stored cases. We then present the algorithm of USIMSCAR.

4.1 Soft-Matching Class Association Rules (SCARS)

A scar is a car whose antecedent and consequent are made by applying the soft-matching criterion. Our aim for building association knowledge is to encode the special knowledge of "how attribute-value pairs of known problems are actually *associated* with specific known solutions." Thus, it needs to be noted that we consider only cars representation, since it is suited to this objective.

A scar r, $X \rightarrow s$, reveals that a problem p is likely to be associated with a solution s, if p's attribute-value pairs are similar to an itemset X. The likelihood is quantified by r's *interestingness*. Interestingness measures are very useful to evaluate the quality of association rules [13]. For the measures, the support and confidence criteria are often 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 the correlations between them. One example is the *Laplace* measure [13]. Below we describe it in detail. We first provide the definition of scars, and scars mining, following the definitions of terms in Section 3.2.

• *scars*: Let SM be an $m \times m$ similarity matrix, where m is the total number of items in I. Let $sim(x, y)$ be a similarity, between two items $x, y \in I$, driven from SM. We say that an item x is *similar* to an item y ($x \sim y$), iff $sim(x, y) \geq$ minsim (the user-specified minimum similarity). Let $SIM(X, Y)$ be a similarity between two itemsets $X, Y \in I$, $SIM(X, Y) = \sum_{x,y} sim(x, y)/|X|$, where $x, y \in X$ are two items characterized by the same attribute. We say that an itemset (or a case) Y *soft-support* of an itemset X ($X \subseteq_{soft} Y$), iff $SIM(X, Y) \geq$ minsim. The *soft-supporting-sum* of X regarding $\mathcal{D}$ is defined as $softSuppSum(X) = \sum_{t \in T} SIM(X, t)$, for each case $T \in \mathcal{D}$ satisfying $X \subseteq_{soft} T$. The *soft-support* of X regarding $\mathcal{D}$ is defined as $softSupp(X) = softSuppSum(X)/|\mathcal{D}|$. The *soft-support* of a scar $X \rightarrow s$ is defined as the fraction of cases, in $\mathcal{D}$, that soft-support X and are described by the s-item s, $softSupp(X \rightarrow s) = softSupp(X \cup s)$. The *soft-confidence* of this rule is defined as $softConf(X \rightarrow s) = softSupp(X \rightarrow s)/softSupp(X)$. A *ruleitem* is of the form $\langle X, s \rangle$ and basically represents a rule $X \rightarrow s$. A scar is an implication of the form $X \rightarrow s$, whose soft-support is greater than minsupp. The definition of our soft-support differs from the one used in SoftApriori. In SoftApriori, the

soft-support of each itemset is calculated by summing the number of *occurrences* of all similar itemsets. For example, the soft-support, of an 1-itemset $x \in I$, is computed as "$softSupp(x) = \sum_{y \in I} sim_{01}(x, y) \cdot supp(y)$", where $sim_{01}(x, y)$ is a *binary* function which is 1, if $x \sim y$, and 0, otherwise; and $supp(y)$ is the support of 1-itemset $y \in I$. Unfortunately, $softSupp(x)$ cannot reflect the different degrees of the similarities between x and all $y \in I$. In contrast, our definition replaces $sim_{01}(x, y)$ by $SIM(x, y)$ so that it can reflect such degrees.

• *scars mining*: The key operation of scars mining is to find all "ruleitems" that have soft-supports greater than minsupp. We call such ruleitems *frequent* ruleitems. For all the ruleitems that have the same set of items in the antecedent, one with the highest interestingness is chosen as *possible rule* (PR). To measure the interestingness, we choose a measure that combines soft-support and soft-confidence such that they are monotonically related (i.e. positively correlated). Thus, we choose the *Laplace* measure [13]. Given a scar r, this measure is defined as $Laplace(r) = \frac{N \cdot softSupp(r)+1}{N \cdot softSupp(r)/softConf(r)+2}$, where N is the total number of cases in $\mathcal{D}$. Since N is a constant, it is easy to see that this measure is monotonically related to soft-support and soft-confidence. If $Laplace(r)$ is greater than a user-specified minimum interestingness, called min-interesting, we say that r is *accurate*. A candidate set of scars consists of all PRs that are both "frequent" and "accurate". To select optimal scars, we finally ignore a scar $X \rightarrow s$ in the set, where $|X|$ is less than a user-specified minimum itemset size, named minitemsize.

4.2 The USIMSCAR Algorithm

USIMSCAR is designed to find potentially useful *objects* that can be used to solve a new problem by exploiting similarity and association knowledge. Each of these objects can be either a case or a scar. Given a new problem q, the goal of USIMSCAR is to generate a *retrieval result set* (RR) that holds those objects.

Let $\mathcal{D}$ be a set of cases; $prSCARS$ be the set of scars to be generated; and SM be the same similarity matrix used in scars mining. We now present the USIMSCAR algorithm $\mathcal{M}$ in the following:

(1) $\mathcal{M}$ retrieves the k most similar cases to q in $\mathcal{D}$, and stores them into a set RC. Assume that $SIM(q, c)$ is the function used for measuring the similarity between q and a case c in $\mathcal{D}$. This function can be defined using the global-local principle. The local similarities, for individual attributes of q and c, are computed using SM.

(2) $\mathcal{M}$ retrieves the k most similar scars to q in $prSCARS$. An important question raised here is how to determine the similarity $SIM(q, r)$ between q and a scar r. Its answer lies in our choice of cars representation for scars mining. This implies that scars have the *identical* structure to all cases in $\mathcal{D}$. To illustrate, assume a case c is simply characterized by attributes A_1 and A_2. So, c is formed as $c = (a_1, a_2)$, where $a_1 \in A_1$ is a problem and $a_2 \in A_2$ is a solution of a_1. Using c, we can generate a scar r, $\{(A_1, a_1)\} \rightarrow (A_2, a_2)$. Note that the values a_1 and a_2, in the antecedent and consequent of r, correspond to the problem a_1 and the solution a_2 of c, respectively. This choice allows $\mathcal{M}$ to compute $SIM(q, r)$ using

the same similarity measure used in the step (1). The retrieved rules are stored into a set RS.

(3) For each case $c \in RC$, $\mathcal{M}$ selects a specific scar $r_c \in prSCARS$ that meets the following condition: a scar $r \in pcSCARS$ is chosen as r_c, if it has the highest interestingness, $Laplace(r)$, among specific scars in $pcSCARS$. These scars must *cover*[7] c and also their solutions are equal to the solution of c. Then, $\mathcal{M}$ computes a *combined score*, for q and c, along with r_c by integrating two factors: $SIM(q, c)$, computed by using similarity knowledge, and $Laplace(r_c)$, calculated by using association knowledge. We denote this score as $cs(q, c)$, defined by $cs(q, c) = SIM(q, c) \cdot Laplace(r_c)$. If r_c is more than one, for example $r_c = \{r_{c1}, ..., r_{cm}\}$, $\mathcal{M}$ uses the average of $Laplace(r_{c1})$, ..., $Laplace(r_{cm})$. If there is no r_c for c, we use the given min-interesting[8], instead of $Laplace(r_c)$. Our combination scheme aims to enhance the significance of $SIM(q, c)$ by weighting the interestingness of r_c. Then, a *universe object*[9] is created. It has two fields. The *instance* field stores c, and the *cs* field holds $cs(q, c)$. This object is added to a set UR.

(4) For each scar $r \in RS$, $\mathcal{M}$ computes a combined score of r regarding q. That is, $cs(q, r) = SIM(q, r) \cdot Laplace(r)$. A universe object is then created, whose *instance* field stores r and *cs* field holds $cs(q, r)$. It is also added to the UR. This combination aims to enhance the significance of r's interestingness by weighting it with r's similarity to q.

(5) $\mathcal{M}$ finally produces the set RR that is a subset of UR through the function $getRR(UR)$. Before we explain this function, we first illustrate how the above four steps (1)-(4) perform using an example, to ease of the readability of $\mathcal{M}$. Then, we give the description of $getRR(UR)$.

An Example. Consider again the patient cases in Table 1a. Using these cases, we can obtain four scars shown in Table 2. To generate the scars in the above table, we used the similarity knowledge encoded in the similarity measure "SIM" in Equation 2. Recall that $\mathcal{Z}$[10] retrieved p_4 as the most useful case to q, and its diagnosis 'gastritis' was determined to be the diagnosis of q. However, this led to trouble, since q suffered from 'appendicitis', not 'gastritis'.

Table 2. The scars generated: A_1 = 'Local Pain', A_2 = 'Other Pain', A_3 = 'Fever', A_4 = 'Appetite Loss', A_5 = 'Age' and A_6 = 'Diagnosis'

ID	Antecedent	Consequent	*Laplace*	Covered by
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 can overcome this trouble. It takes the following steps (assume k=2): (1) It generates the 2 most similar cases to q by using SIM. Thus, RC =

[7] We say that a scar r *covers* a case c, iff r is generated being soft-supported by c.
[8] This is mentioned in "scars mining" in Section 4.1.
[9] We refer to a universe object as a generic object that can encapsulate any case and scar and also have any attributes.
[10] $\mathcal{Z}$ was used as a representative model of SBR in Section 2.

$\{p_4, p_1\}$ (See Table 1b). (2) It generates the 2 most similar scars to q by also using SIM. Thus, $RS = \{r_1, r_4\}$ with $SIM(q, r_1) = 0.640$ and $SIM(q, r_4) = 0.637$. (3) For each case $c \in RC$, the r_c is determined. With this example, for p_4, r_{p_4} is selected as r_4, and, for p_1, r_{p_1} as r_1, r_2 and r_3. Then, the combined scores $cs(q, p_4)$ and $cs(q, p_1)$ are computed: $cs(q, p_4) = 0.494$ and $cs(q, p_1) = 0.582$. Thereafter, p_4 and p_1, with their combined scores, are copied to new universe objects. These object are then added to a set UR. (4) For each scar $r \in RS$, its combined score, regarding q, is computed: $cs(q, r_1) = 0.590$, and $cs(q, r_4) = 0.494$. Then, r_1 and r_4, with their combined scores, are copied to new universe objects. These objects are also added to the UR. The further exploitation of the UR is explained below.

Function *getRR(UR)*. This function aims to retrieve a subset of "universe objects" (simply objects) from the UR. These objects are selected to be potentially useful to solve the query q. To realize this function, we use both the "combined scores" of objects in the UR, and the "number" of objects in the UR that are associated with the same solution. The solution of each object $o \in UR$ is differently interpreted, according to whether o was created from a case c or a scar r. If created from c, its solution corresponds to c's solution, Otherwise, its solution corresponds to the solution in the r's consequent.

Let S_e be a set of solutions of objects in the UR. For each object in the UR, we find a subset $S_{e_k} \in S_e$, where all objects in $(S_{e_k})_{k \leq |S_e|}$ are associated with the same solution. For any $i, j \leq |S_e|$, thus $S_{e_i} \cap S_{e_j} = \emptyset$ holds. Then, for each $S_{e_k} \in S_e$, we compute the average of the combined scores of objects in S_{e_k}, denoted as $avg(S_{e_k})$. This $avg(S_{e_k})$ is further enhanced by multiplying a ratio, denoted as $strength(S_{e_k}) = |S_{e_k}|/|UR|$. The enhanced score is called the *final score* of S_{e_k}, and denoted as $fs(S_{e_k}) = avg(S_{e_k}) \cdot strength(S_{e_k})$. We then retrieve the objects, in the UR, grouped by the n top ranked solutions, by means of their final scores. If $n = 1$, we retrieve the objects grouped by the solution satisfying $\max(fs(S_{e_i}))_{i \leq |S_e|}$. These objects are finally stored into the RR.

To illustrate, consider the UR formed in the above example. Assuming that s_1 = 'gastritis' and s_2 = 'appendicitis', the UR has four objects: $UR = \{o_1, o_2, o_3, o_4\}$[11], where $\{o_1.cs = 0.494, o_1.s = s_1\}$, $\{o_2.cs = 0.582, o_2.s = s_2\}$, $\{o_3.cs = 0.590, o_3.s = s_2\}$ and $\{o_4.cs = 0.494, o_4.s = s_1\}$. With the UR, $fs(s_1) = 0.247$ and $fs(s_2) = 0.293$. If we choose the objects, in the UR, grouped by the solution satisfying $\max(fs(s_i))_{i=1,2}$, USIMSCAR returns the result set $RR = \{o_2, o_3\}$. Then, objects in the RR can be used to determine a solution of q using voting. With this example, since o_2 and o_3 have the solution 'appendicitis'. USIMSCAR thus give a diagnosis for q as 'appendicitis' that q really had.

5 Evaluation

Our evaluation goal is to empirically validates that USIMSCAR can improve similarity-based retrieval (SBR). To achieve it, we need to determine two

[11] Assume that each object has another field s representing "solution".

essential ingredients. The first is a set of representative models, implementing SBR, to be compared with USIMSCAR. The second is an application field, where the models and USIMSCAR can be properly tested. As the representative models, we choose k-NN and its several extensions, since SBR is typically implemented through the technique using a derivative of k-NN, as mentioned in Section 2. Regarding the application field, we choose *classification*, since the case-based approach, using the chosen models, to classification usually requires no sophisticated adaptation methods [3]. For the classification task, the performance of the models relies almost completely on the retrieval task identifying the similar cases to a new problem q [14]. So, we choose several k-NN based classifiers, for our comparison purpose, that will be described in the following section.

From the viewpoint of k-NN based classifiers, classification has two stages. The first is the determination of the nearest neighbors of a new problem q, driven by similarity knowledge. The second is the determination of the class of q using these neighbors. From the viewpoint of USIMSCAR, the first is the determination of the "useful cases and rules" $\mathcal{CR}$ for q, driven by "similarity knowledge and association knowledge". The second is the determination of the class of q using $\mathcal{CR}$. Our work is only focused on the first stage. Hence, to achieve the classification task, we use the simplest approach, majority voting (See Section 2), for the second stage. For the fair comparison, we configure majority voting to be used in the k-NN based classifiers compared. By applying this evaluation scheme, we justify the performance comparison between USIMSCAR and SBR.

5.1 Evaluation Methodology

We compare 5 k-NN based classifiers with USIMSCAR in our evaluation. These are all implemented in Weka [15]: (1) IB1 [16] is the simplest form of k-NN classifiers. To classify a new problem q, its nearest neighbor n_1 is selected by using the Euclidean distance. Then q is classified to be the class of n_1. As the baseline for our comparison purpose, we choose IB1 due to its simplicity. (2) IBkBN extends IB1 by using the best $k-$the number of nearest neighbors. The best k is determined by cross-validation. (3) IBkFS extends IBkBN by using feature selection. We choose the correlation-based feature selector [17] (known as CfsSubsetEval in Weka), to determine the goodness of feature subsets. (4) IBkFW extends IBkBN by using feature weighting. We choose InfoGainAttributeEval evaluator in Weka, due to its popularity. It assigns weights to features individually, based on the information gain with respect to the class. (5) KStar is an implementation of K* [18], where the similarity for finding nearest neighbors is defined by their *entropy*. The entropy is measured as the complexity of transforming one instance into the other. To classify q, KStar uses the probability of q being in class c by summing the probabilities from q to each member of c. It then chooses the class with the highest probability as the classification of q.

As our test datasets, we used 6 datasets found from UCI ML Repository (See Table 3). These were chosen by the criteria of being different in terms of number of instances (cases), number and types of attributes, and number of classes.

Table 3. The test datasets used in the experiments

Dataset	Instance #	Attribute #	Attribute Type			Class #
			Numeric	Binary	Nominal	
Breast Cancer	683	9	9			2
Car	1728	6			6	4
Heart Disease	270	13	6	3	4	2
Clever	297	13	6	3	4	2
Crx	653	15	6	4	5	2
Tae	151	5	1	2	2	3

For the evaluation metric, we used *classification accuracy*, since it is often assumed to be the best performance indicator in classification [19]. It measures the ratio of correctly classified instances over all the instances tested. To test a model on the datasets, we used *10-fold cross-validation*, in which each dataset is divided into 10 subsets. Of the 10 subsets, a subset is retained as *testing data*, and the remaining 9 subsets are used as *training data*. The validation process is then repeated 10 folds (times). Then, the 10 results from the folds were used to measure the classification accuracy of the model.

For USIMSCAR, the similarity knowledge was defined on two attribute types, numeric and categorical (including boolean). It is defined using the global-local principle, actually encoded in the similarity measure in Equation 2. The feature weights, in Equation 2, were equally assigned. Note that this measure is another form of the Euclidean distance that is used in the classifiers compared.

To generate scars, the following parameters were used: minsupp = 0.02 (2%), minsim = 0.98 (98%), min-interesting = 0.7 (70%), minitemsize = 0.8 (80%). To run USIMSCAR, we set k to 6.

5.2 Results and Analysis

We now evaluate the results of USIMSCAR, in comparison with the compared classifiers, in classification accuracy. We first compare the results of USIMSCAR and the baseline IB1. The results are shown in Table 4.

Referring to the above table, for each dataset, the better one in the accuracy is denoted in boldface. Also, the mark "•" indicates that USIMSCAR is determined to attain a statistically significant improvement over the target classifier, while "◦" shows there is no significant improvement found. As observed in the table, USIMSCAR outperforms IB1 on all the datasets in the accuracy. Outstandingly, USIMSCAR achieves 16.425% (maximum difference) higher than IB1 on the Car. Using the Z-test [20] at 95% confidence, for differences in the accuracy, USIMSCAR shows a significant improvement over IB1 on five datasets.

We now compare the results, of USIMSCAR with IBkDN, IBkFS and IBkFW, in classification accuracy. The results are seen in Table 5, where the best one in the accuracy for each dataset is also denoted in boldface. Also, the k, selected from 1 to 30, that gives the best classification accuracy on each dataset for each classifier, is denoted in the parentheses. As observed in the table, USIMSCAR occupies the 2^{th} place on the Breast Cancer, with a small difference (0.44%), compared to the

Table 4. USIMSCAR vs. IB1 in classification accuracy (%)

Dataset	IB1: Baseline	USIMSCAR
Breast Cancer	95.461 ○	**96.193**
Car	80.334 •	**96.759**
Heart Disease	75.556 •	**85.185**
Clever	76.014 •	**84.459**
Crx	81.317 •	**87.136**
Tae	64.238 •	**67.550**

Table 5. USIMSCAR vs. IBkBN, IBkFS and IBkFW in classification accuracy (%).

Dataset	IBkBN	IBkFS	IBkFW	USIMSCAR
Breast Cancer	**96.633** (k=5)	**96.633** (k=5)	95.900 (k=5) ○	96.193
Car	80.334 (k=1) •	80.334 (k=1) •	84.833 (k=1) •	**96.759**
Heart Disease	81.852 (k=8) ○	77.407 (k=6) •	82.222 (k=10) ○	**85.185**
Clever	82.432 (k=7) ○	79.730 (k=10) ○	81.419 (k=7) ○	**84.459**
Crx	86.524 (k=10) ○	85.605 (k=3) ○	86.630 (k=7) ○	**87.136**
Tae	64.238 (k=1) •	46.358 (k=2) •	44.371 (k=1) •	**67.550**

accuracy of IBkBN and IBkFS. However, USIMSCAR outperforms all the compared classifiers on all the remaining datasets. Outstandingly, it achieves 16.425% better than IBkBN on the Car, 21.192% better than IBkFS on the Tae, and 23,179% better than IBkFW on the Tae. Each mark "•" shows that there is a statistically significant difference between USIMSCAR and the target classifier on the considered dataset, using the Z-test at 95% confidence.

We now compare the results of USIMSCAR and KStar in classification accuracy. As observed in Table 6, USIMSCAR outperforms KStar on all the datasets. Outstandingly, USIMSCAR performs 10.370% and 17.463% better than KStar on the Car and Heart Disease, respectively, in the accuracy. Using the Z-test at 95% confidence in these results, we observe that the differences between them are statistically significant on five datasets, as the mark "•" indicates.

Table 6. USIMSCAR vs. KStar in classification accuracy (%)

Dataset	KStar	USIMSCAR
Breast Cancer	94.876 ○	**96.193**
Car	79.296 •	**96.759**
Heart Disease	74.815 •	**85.185**
Clever	75.675 •	**84.459**
Credit Approval	78.560 •	**87.136**
Tae	59.603 •	**67.550**

We finally compare USIMSCAR with the classifiers, in the averages of the results in Tables 4 - 6. The comparison is presented in Fig. 1. As observed, USIMSCAR performs 7.393%, 4.211%, 8.536%, 6.985% and 9.076% better than IB1, IBkBN, IBkFS, IBKFW and KStar, respectively. These differences show

that USIMSCAR achieves statistically significant improvements over the classifiers using the Z-test at 95% confidence. Through the experimental evaluation, we empirically verify that USIMSCAR is an effective retrieval strategy for CBR.

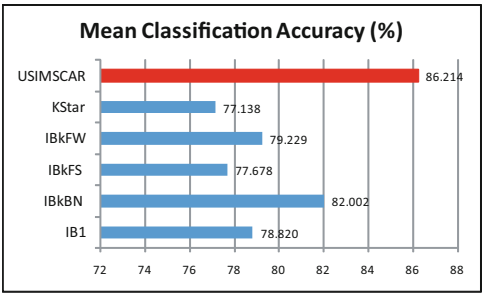

Fig. 1. The mean classification accuracy results

6 Related Work

SBR is typically implemented through the technique using a derivative of k-NN. To improve its accuracy, various extensions have been developed, including the integration of k-NN and the best k, feature selection, or feature weighting. USIMSCAR differs from these techniques. The most distinctive difference is the use of association knowledge for the retrieval task. The last two techniques often focus on finding the features that are highly correlated to specific solutions. But they only consider relationships between individual features and each solution. This leads to trouble, since they ignore complex relationships between multiple features and each solution. For example, two features may be individually correlated with a certain solution, but together they may not, or vice versa. In contrast, USIMSCAR exploits not only the individual relationships, but complex relationships between multiple features (itemsets in scars) and solutions.

Several researchers have attempted to augment SBR with certain factors obtained through statistical learning and adaptation knowledge. For example, Park et al. [21] suggest a new case retrieval technique, called *statistical* CBR (SCBR). The idea of SCBR is that an optimal number of neighbors can be dynamically obtained by considering the distribution of distances between potential similar neighbors for a new problem. SCBR finds the optimal distance threshold θ, and selects similar neighbors satisfying θ. Smyth and Keane [2] propose the adaptation-guided retrieval approach that provides direct link between the retrieval and the adaptation task in CBR. This approach utilizes formulated adaptation knowledge about whether a case can be easily modified to fit a new problem to influence similarity assessment during the retrieval phase. Hoffmann [22] also applies this approach for a dietary consultation evaluation for patients. USIMSCAR also significantly differs from the above approaches. First, it exploits association knowledge derived using data mining techniques and incorporates it into the retrieval task. Second, it does not assume that any kind of adaptation knowledge must be formalized in advance.

7 Extension Schemes and Conclusion

In CBR, cases can also be represented by more complex structures, like object-oriented (OO) or hierarchical (HR) representation [1]. We briefly give possible extension schemes, in which USIMSCAR could support the cases modeled using such structures. The OO representation utilizes the data modeling approach of the OO paradigm, such as "is-a" and inheritance [1]. In the HR representation, a case is characterized through multiple levels of abstraction, and its attribute values can reference nonatomic cases [1]. For USIMSCAR to treat the cases, characterized by those two representations, two issues must be addressed−how to formalize similarity knowledge, and how to generate association knowledge. To address the former, one may use similarity measures, for OO data [23] or HR data [24], widely studied in IR. To address the latter, one may integrate the soft-matching criterion and specific algorithms extending Apriori [10]. A possible choice for such algorithms is OR-FP [25] for OO data and DFMLA [26] for HR data.

USIMSCAR may also be extended to cases, where each case problem is associated with more than one solution. This occasion can be simply generalized into the occasion−each case problem is associated with one solution. The generalization is possibly done by splitting a case $\mathcal{C}$ into k number of sub cases (k: the number of solutions). We then restrict all the sub cases to have the same case identification with $\mathcal{C}$. Then, USIMSCAR may run for the cases, whose solutions are more than one, without any modification. This scheme even may be extended for solutions described in free-text. As long as such solutions are converted to the "bag-of-words" representation [6], the above scheme can be also applied.

In this paper, we proposed a new retrieval strategy USIMSCAR, aimed to improve similarity-based retrieval (SBR), used in many CBR systems. The uniqueness of USIMSCAR is to exploit the specific knowledge, integrating similarity knowledge and association knowledge, into retrieval in CBR. Similarity knowledge is encoded in similarity measures, while association knowledge is derived using association rule mining techniques. The goal of association knowledge is to represent implicit, previously unknown and potentially useful associations between problem features and solutions among stored cases. This knowledge is combined with similarity knowledge to make a more complete retrieval strategy. We empirically evaluated the performance of USIMSCAR, in comparison with several retrieval methods adopting SBR. The evaluation results showed that USIMSCAR is an effective retrieval strategy for CBR.

References

1. Lopez De Mantaras, R., 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)
2. Smyth, B., Keane, M.T.: Adaptation-guided retrieval: questioning the similarity assumption in reasoning. Artif. Intell. 102, 249–293 (1998)

3. Stahl, A.: Learning of knowledge-intensive similarity measures in case-based reasoning. PhD thesis, Technical University of Kaiserslautern (2003)
4. Dudani, S.A.: The Distance-Weighted k-Nearest-Neighbor Rule. IEEE Transactions on Systems, Man and Cybernetics SMC-6, 325–327 (1976)
5. Jiang, L., Cai, Z., Wang, D., Jiang, S.: Survey of Improving K-Nearest-Neighbor for Classification. In: FSKD 2007: Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery, pp. 679–683 (2007)
6. Cunningham, P.: A Taxonomy of Similarity Mechanisms for Case-Based Reasoning. IEEE Trans. on Knowl. and Data Eng. 21, 1532–1543 (2009)
7. Yusta, S.C.: Different metaheuristic strategies to solve the feature selection problem. Pattern Recogn. Lett. 30, 525–534 (2009)
8. Wettschereck, D., Aha, D.W.: Weighting features. In: Proceedings of the First International Conference on CBR Research and Development, pp. 347–358 (1995)
9. Castro, J.L., Navarro, M., Sánchez, J.M., Zurita, J.M.: Loss and gain functions for CBR retrieval. Inf. Sci. 179, 1738–1750 (2009)
10. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: VLDB 1994, pp. 487–499 (1994)
11. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Proceedings of the 4th KDD, pp. 443–447 (1998)
12. Nahm, U.Y., Mooney, R.J.: Mining soft-matching association rules. In: Proceedings of CIKM 2002, pp. 681–683 (2002)
13. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: A survey. ACM Comput. Surv. 38, 9 (2006)
14. Jurisica, I., Glasgow, J.: Case-Based Classification Using Similarity-Based Retrieval. In: Proceedings of ICTAI (1996)
15. Witten, I.H., Frank, E.: Data mining: Practical machine learning tools and techniques with Java implementations. Morgan Kaufmann, San Francisco (2000)
16. Aha, D.W., Kibler, D., Albert, M.K.: Instance-Based Learning Algorithms. Mach. Learn. 6, 37–66 (1991)
17. Hall, M.A.: Correlation-based Feature Subset Selection for Machine Learning. PhD thesis, University of Waikato, Hamilton, New Zealand (1998)
18. Cleary, J.G., Trigg, L.E.: K*: An Instance-based Learner Using an Entropic Distance Measure. In: Proceedings of the 12th ICML, pp. 108–114 (1995)
19. Lim, T.S., Loh, W.Y., Shih, Y.S.: A comparison of prediction Accuracy, complexity, and training time of thirty-three old and new classification algorithms. Machine Learning, 203–229 (2000)
20. Richard, C.S.: Basic Statistical Analysis. Allyn & Bacon, Boston (2003)
21. 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)
22. Hoffmann, A., Khan, A.S.: A new approach for the incremental development of retrieval functions for CBR. Applied Artificial Intelligence 20, 507–542 (2006)
23. 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)
24. Ganesan, P., Garcia-Molina, H., Widom, J.: Exploiting hierarchical domain structure to compute similarity. ACM Trans. on Infor. Sys. 21, 64–93 (2003)
25. Kuba, P., Popelinsky, L.: Mining frequent patterns in object-oriented data (2005)
26. 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)