A New Strategy for Case-Based Reasoning Retrieval Using Classification Based on Association

Ahmed Aljuboori, Farid Meziane and David Parsons

University Of Salford, School of Computing Science and Engineering, M5 4WT, UK
a.s.aljuboori@edu.salford.ac.uk; f.meziane@salford.ac.uk; d.j.parsons@salford.ac.uk

Abstract. This paper proposes a novel strategy, Case-Based Reasoning Using Association Rules (CBRAR) to improve the performance of the Similarity base Retrieval SBR, classed frequent pattern trees FP-CAR algorithm, in order to disambiguate wrongly retrieved cases in Case-Based Reasoning (CBR). CBRAR use class association rules (CARs) to generate an optimum FP-tree which holds a value of each node. The possible advantage offered is that more efficient results can be gained when SBR returns uncertain answers. We compare the CBR Query as a pattern with FP-CAR patterns to identify the longest length of the voted class. If the patterns are matched, the proposed strategy can select not just the most similar case but the correct one. Our experimental evaluation on real data from the UCI repository indicates that the proposed CBRAR is a better approach when compared to the accuracy of the CBR systems used in our experiments.

Keywords: class association rules, frequent pattern trees, case-based reasoning, retrieval, P-trees.

1 Introduction

The basic premise of case-based reasoning (CBR) is that experience in the form of previous cases can be influenced to solve new problems [1]. An individual experience is named a case, and its collection is stored in a case base [2]. Basically, each case is defined by a problem description and its corresponding solution description. Among the four main phases, retrieval is a key stage, with success being heavily reliant on its performance [3]. Its aim is to retrieve similar or useful cases that can be successfully used to solve a target problem. This is of particular importance because if the retrieved cases are not useful, CBR systems may not ultimately produce a suitable solution to the problem [2].

Fundamentally, retrieval is performed through a specific strategy of leveraging similarity knowledge (SK) referred to as 'similarity-based retrieval.' (SBR) [3]. In SBR, SK is utilized to determine the benefit of stored cases with regards to a target problem. SK is typically encoded via similarity measures between the problem and stored cases. In SBR, the measures are used to identify cases ranked by their similarities to the problem. Their solutions are then used to solve the problem.

Association rules mining (ARM) is an important technique in the field of data mining (DM). ARM is used to extract interesting correlations, associations or casual structures among a set of items in a transaction database or other data repositories. It is used in various application areas, such as banking, products relationships and frequent patterns. The class association rule (CAR) technique was first proposed by [4]. It generates classification rules based on association rules (ARs). Other techniques for mining CARs have been suggested in recent years. They include GARC [5], ECR-CARM [6], CBC [7], CAR-Miner [8], CHISC-AC [9] and developed d2O [10]. The methods of classification based on CARs were demonstrated to be more accurate than the classic methods e.g. C4.5 [11]and ILA [12, 13] in their practical results [4].

Frequent pattern mining (FPM) plays a major role in ARM. On its own FPM is concerned with finding frequent patterns (frequently co-occurring sub-sets of attributes) in data. A number of FPM algorithms have been proposed for instance Apriori [14]. With respect to pattern matching the majority of these have been integrated with ARM algorithms. Of these, the best known, and most frequently cited, is the FP-Growth algorithm [15]. FP-growth is constructed on a set enumeration tree structure called the FP-tree. It takes a totally different approach to discovering frequent itemsets. Unlike Apriori, it does not generate and test the paradigm. Instead, FP-growth compacts the data set structure using FP-tree and extracts the frequent pattern directly from this structure [16]. FP-tree is a compressed representation of the input data. It is built by reading the dataset transaction and allocating each transaction to a path in the FP-tree. As various transactions can have many items in common, their paths might overlap. The more the paths overlap with one another, the more can be achieved by using the FP-tree structure. The performance of this process will depend on the amount of memory available on the system being used. If the FP-tree can be held entirely within the available memory, the extraction of frequent itemsets will be faster as it will be possible to avoid repeated passes over stored data.

In this paper, we propose CBRAR a new strategy for enhancing the performance of CBR by using a new more efficient algorithm (FP-CAR) for mining all CARs with FP-tree values for a CBR query *Q*. The proposed algorithm uses an optimum tree derived from the FP-tree and optimized by P-tree concepts to produce a super-pattern that matches the new CBR case. Our initial experimental results show that the CBRAR strategy is able to disambiguate the answers of the retrieval phase compared to those obtained when using Jcolibri [17] and FreeCBR [18] systems for example.

2 Literature Review of CBR and other Types of Knowledge

CBR is a well-studied area in machine learning. In the past decades several researchers have studied CBR methods in real world applications, such as medical diagnosis [19], [20], product recommendation [21] and personal rostering decisions [22]. CBR is a cyclic and integrated process of solving a problem and learning from the experience of experts, which is used to build a knowledge domain which is then recorded to be used to help solve future problems. It can be defined as “to solve a problem, remember a similar problem you have solved in the past and adopt the old solution to

solve the new problem" [23]. CBR methods are composed of four steps: retrieve-find the best matching of previous cases, reuse-find what can be reused from old cases, revise-check if the proposed solution may be correct, and retain-learn from the problem solving experience. This decomposition of CBR phases is based on [1] and illustrated in **Fig**. 1.

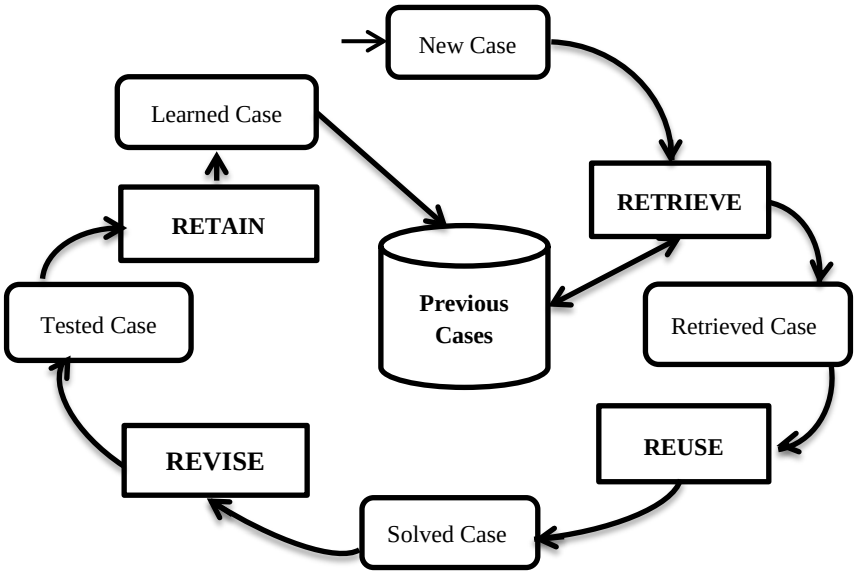


Fig. 1. CBR Cycle [1]

2.1 Machine Learning and Retrieval

The development of machine learning has resulted in retrieval approaches that SBR merges with rule-induction (RI) approaches to enhance SBR. RI systems often learn domain-particular knowledge and represent it as IF–THEN rules. It is suggested that such rules can be utilized for determining the weights of case features in SBR [24]. [25] shows that decision tree algorithms can be used to discover domain-specific rules from a specific case base. From such rules, users select useful rules according to the thresholds set up by experts. The extracted rules are then used to point a target problem to its most similar case set and to calculate the weights of the case features. Such knowledge is finally used to retrieve the most similar case from the case base. A retrieval paradigm in [26] chooses between SBR or a RI method (using decision trees) for the target problem, considering the similarities of cases in a case base.

The CBRAR approach is different from these approaches in that Association Knowledge (AK) is not used to measure the weights of case features, but to refine the cases retrieved by SBR and guide more specific rules to the target problem.

2.2 Data Mining and CBR

Over time, techniques of integrating data mining (DM) and K nearest neighbor (KNN) have often been implemented in CBR research to improve KNN through three main platforms. First, to integrate feature weighting (FW) and feature selection (FS) into KNN. In this framework, FW is used to estimate the optimal weights of the original features of cases [27], [28] , and FS is employed when choosing relevant features of cases [20], [22], or their aggregation is used to leverage their usefulness [19]. Second, to merge data clustering with KNN, where the structure of clustered cases is leveraged to lead to more relevant cases [29], [30]. 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 [31]. Third, to apply both DM and SBR techniques together to discover cases related to the target problem. For instance, [32] displays how to integrate DM with SBR to improve liver diagnosis. Given a target problem, once a DM method (a back-propagation neural network) is applied on the case base, some cases thought to be relevant to the problem are retrieved. These cases are then tested to verify whether these are adequately similar to the target problem through SBR. Similar cases are merely used as a retrieval result for the problem. Unlike this scheme, our approach is based mainly on the use of AK built via CARs.

2.3 Retrieval and CAR

Basically, retrieval is achieved by employing two methods: (AK) and (SK). The retrieval is normally achieved utilizing SBR which is a technique based on SK. In SBR, SK is utilized for estimating the retrieval of similar cases to the target problem. The similarity measure is used between the various cases available and the problem to find those cases that can be selected to solve the target. Nevertheless, defining the SK can be considered as a main disadvantage of SBR because it is reliant on domain experts and is a time consuming process [33]. The similarity standard defined for one domain differs for numerous domains that are helpful for some problems and not for others. Therefore, the performance of SBR varies from problem to problem even within the same domain [26].

Association rules (ARs) aim to find interesting relationships (associations) in a transaction database [14]. The focus is usually on discovering a set of highly co-occurring features shared by a large number of transactions in a database. It is an implication of the form $X \rightarrow Y$, where X and Y are nonintersecting sets of items. For example, {milk, eggs}→{bread} is an association rule that says that when milk and eggs are purchased, bread is likely to be purchased. In the context of CBR, ARs can be employed to determine interesting relationships from a given case base. Furthermore, the transaction of the item can be considered as a case and an attribute as a value pair, respectively. The most traditional algorithm is Apriori [34] which has been used to evaluate and rank a large number of extracted ARs that have support and confidence which is not less than that specified by a user [35]. CAR is a specific subset of ARs whose consequents are restricted to one target class. In the context of CBR, a CAR is considered as an AR whose consequent holds the item formed as a pair of a

solution attributes and its value [36]. In a given case base library, AK is encoded to show how a specific problem's features are associated with a certain solution.

3 Related Work

3.1 Soft Matching of ARM (SARM)

A limitation of traditional ARM algorithms for rule $X \rightarrow Y$ e.g. Apriori [34] is that items X and Y are discovered based on the relation of equality. Basically, these algorithms perform poorly when dealing with similar items. For instance, Apriori cannot find rules like 70% of the customers who buy products similar to yogurt (e.g. milk) and products similar to mayonnaise (e.g. egg) also buy baguettes. Soft matching was suggested to address this [37], where the consequents and antecedent of ARs are discovered by similarity valuation. The SARM standard is used to find all rules from $X \rightarrow Y$, where minimum support and minimum confidence of each rule are not less than soft support and soft confidence, respectively. Support and confidence are used to generalize the definition of soft support and soft confidence.

This generalization is performed by allowing elements to match, so long as their similarity exceeds minimum similarity (minsim) as specified by the user. The soft-matching criteria can be employed to model better relationships among features of cases instead of the equality relation, by using the concept of similarity.

3.2 Soft - CAR Algorithm

This algorithm calculates the soft support and finds the frequency of each item soft matching CARs. It also discovers the seed set of rules found in every pass in the corresponding class. For every rule item, the seed set of rules are utilized to generate new rule items known as candidate rule items. The soft support is computed through the set of different cases.

It produces SCARs rules in the last pass after it finds the candidate rule items which are frequent from those frequent items [38]. However, experts are required for calculating and defining the SK domain, making this a time consuming and difficult process.

3.3 USIMCAR Algorithm

This algorithm is an expansion of the retrieval phase to improve the performance of the SBR. It encodes the AK in Soft-CARs together with SK to improve the performance of CBR [38]. USIMCAR is used to enhance the usefulness of cases, retrieved through the SK [36], with regard to a new case Q in addition to including the SCAR, thus meaningfully utilizing the cases with their usefulness [36]. In addition, it leverages the AK by searching and finding those SCARs whose usefulness is greater than others concerning Q, therefore valuably using them with their usefulness. Patel [39] also developed the USIMCAR strategy for hierarchical cases which combines the

support-count bit from multilevel and soft-matching criteria (SC-BF) algorithm for the SCARs. Patel also applied the unified knowledge of the AK and similarity to enhance the performance of the SBR. Both strategies [38] and [39] are a simulation of the retrieval phase by providing a percentage value but do not involve providing a CBR system with feedback inputs as part of the original cycle.

In this paper, we propose the FP-CAR algorithm to generate an optimum tree using CARs and FP-tree. The tree is optimized by utilizing various types of association knowledge i.e. P-trees and an equivalence table of implications. FP-CAR is also a part of the suggested CBRAR technique which is an expansion to the SBR. The novel CBRAR is used to disambiguate the wrong retrieved answers as feedback to the CBR.

4 Proposed Algorithm FP-CAR

The FP-CAR (frequent pattern class association rules algorithm) is based on two steps. First, it generates a FP-tree from a set of CARs [40]. Second, the tree is optimized by utilizing the P-tree [41] concepts and equivalence table of implication. These two steps are combined to gain an optimum tree that can be compared with a new case Q of the CBR as a super-pattern to improve the performance of the SBR. The start of the observation is where the options of CARs have been selected as follows (lower support $\xi = 0.1$ and confidence = 0.9, delta = 0.05, number of rules = Maximum), then the existence of rule $X \rightarrow c$ is a subset should make it necessary to consider it as an antecedent of a superset $X,Y \rightarrow c$. Practically, however, we may still find a rule $Y \rightarrow c$, say, where Y is another subset of the same class, where both X and Y form a Superset-Pattern $X,Y \rightarrow c$. In the first case scenario, logical equivalences concepts are utilized to prove the theory behind gaining the equivalence of $((X \rightarrow c) \vee (Y \rightarrow c)) \equiv (X \wedge Y) \rightarrow c$. In other words if X implies c or Y implies c, it is equivalent to X and Y both implying c.

The second case scenario uses the acute inflammation dataset from UCI (see **Table 1**, **Table 2**, **Fig. 2** and **Fig. 3**). In this case, as in Coenen [42] we take advantage of the P-tree to gain a superset. We consider the partial total accumulated at *ABCD* which makes a contribution for all the subsets of *ABCD*. In other words, the contribution in respect of the subsets of *ABC* is already included in the interim total for *ABCD*, therefore, when considering the superset *ABCD*, we need to examine only those subsets which include the attribute *D* [43].

In this paper we suggest an alternative explanatory method: If we can identify a generic rule $X \rightarrow c$ which meets the required support and confidence thresholds, then it is necessary to look for other rules whose antecedent is a superset combined with *(X^Y)* and whose consequent is *c* which distinguishes our algorithm compared to [40]. The objective of the FP-CAR algorithm is to continue to look for rules that select other classes in order to reduce the risk of overfitting and the number of the considered candidate rules.

FP-CAR uses the concepts of classification based on association and the Total From Partial Classification (TFPC) algorithm [40]. It builds a set-enumeration tree structure of the CARs, where the FP-tree contains an incomplete summation of sup-

port-counts for relevant sets and patterns. Using the FP-tree structure to represent all patterns of the CARs, the T-tree [41] concept is used to build an optimum tree that finally contains all the frequent patterns sets (i.e. those that can be compared to the pattern of the CBR query). The FP-CAR is built level by level, the first level comprising all the subsets that contain a value of the attribute under consideration. It compresses the subsets into a prefix tree, where the root *c* holds all frequent items according to their frequency. In the second pass, the unnecessary subsets are removed, from the tree. Candidate-subsets then form a superset from the remaining sets considering the pattern of the CBR. The process continues, with the voting of a length in each class label, until no more candidate sets can be generated. The patterns of subsets will contain a value of each node which can be compared with a CBR query *Q*.

Fig. 2 shows the form of a FP-CAR, for the subsets *{{A,B,C,D},{A,E,F},L,*c_1*}}*, *{{A,B,C},L,*c_2*}* where *L* is a length identifier, c_1 and c_2 are class identifiers, each node of a subset holds a value i.e. *A={yes ,no}.* This tree includes all possible related supersets that are not resolved by SBR, except for those including both c_1 and c_2 which we will assume were pruned. The target of FP-CAR is to find a CBR case problem that caused uncertain answers i.e. *{{A=yes, B=yes, C=yes, D=yes, E=no, F=40, L=6}.* FP-CAR nodes include a value of each node for a superset *Q* i.e. *A= {yes, no}.* Practically, an actual FP-tree would contain all those nodes representing the frequent subsets where FP-CAR includes the voting length and values. For instance, if the set *{A,B,C},L}* fails to reach the required support threshold, and length identifier e.g. 4 to conform to the case problem pattern, then the class of the subset {*A,B,C}* would be ignored, and the superset would not be created. All the candidates that contain the class-identifier *c1* with required length can be found in the subtree rooted at *c1* ,starts with *A* node descended by *{B,C,D,E,F}* frequency as shown in **Table 1.** FP-Tree Hash Table Therefore, all the rules that classify to c_1 can be derived from the root *A* (and also for c_2) whereas those subsets which start with other roots will be removed to gain a super-pattern.

Table 1. FP-Tree Hash Table

Item	Frequency	Priority
B	58	2
A	62	1
C	50	3
D	49	4
E	44	5
F	26	6

→

Ordered-Subsets	Length	Class
A,B,C	3	*c1*
A,C	2	*c1*
A,B	2	*c2*
A,B,C	3	*c2*
A	1	*c2*
A,B,C,D	4	*c1*

The algorithm used to build the FP-CAR tree in **Fig. 2** is a modification of the original FP-Tree approach using TFPC concepts. As each pass is concluded, we ignore from the tree all those subsets that fail to meet the target pattern to form a superset. The remaining (frequent) sets that are included within the class-identifier subtrees of (c_1), define a possible partial superset if one matches the voted length and other one is a complement. For example, the set {*ABCD*c_1*}* is a partial one where $X \rightarrow c_1$

of $L=4$ and $AEFc_1$ is a complement that corresponds to rule $Y \rightarrow c_1$. We now build the supersets of all such sets that match the new case $Q = \{yes, yes, yes, yes, no \text{ and } 40.0\}$. If the threshold of L of the subsets is greater than or equal to the voted class c_1, we add the subset to our target set considering the nodes values, and ignore the corresponding subset from the tree that occurs in c_2. The complement of the superset will then be completed from the same cluster of c_1 i.e. $\{X \wedge Y\} \rightarrow c_1 \equiv Q \rightarrow c_1$ as shown in **Fig. 2**. Connecting the tree below to the results in **Table 2**, proves the theory behind the proposed algorithm.

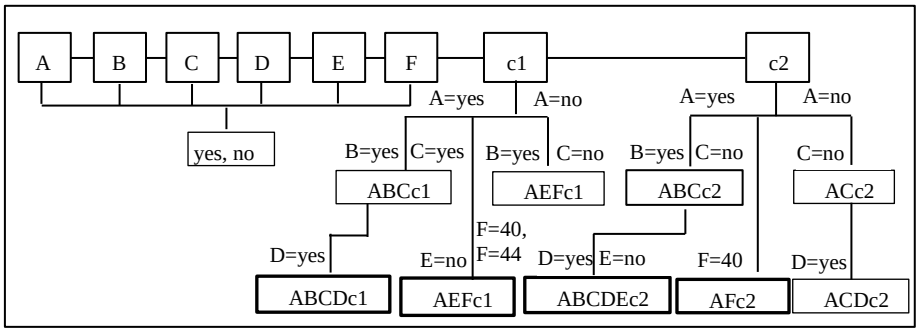


Fig. 2. FP-CAR Algorithm Tree

5 New Strategy CBRAR to Enhance the Performance of SBR

This section presents the proposed new technique CBRAR of integrating CARs into CBR. Basically, there is a possible problem in CBR which is retrieving unrelated cases that cause an incorrect solution. To overcome this problem, CAR is utilized to find the relationship between the case library and a target case. Normally, to achieve the retrieval phase, CBR systems execute similarity SBR. However, SBR tends to depend on similarity knowledge, ignoring other types of knowledge that can benefit and improve retrieval performance. In this research, the challenge is how to retrieve not just the most similar case in CBR but the correct one. Some studies which apply ARs into CBR, for example [38], are much dependent on the experts domain for finding SK. [39] focused on the case representation hierarchically by combining SK and AK depending on the Apriori algorithm when a number of passes are needed to generate new candidates. Both strategies [38], [39] are a simulation of the retrieval phase by providing a percentage value of related cases but do not involve providing a CBR system with feedback, which is part of the original cycle. The new approach CBRAR produces a correct case pattern not just a similar one. It also enables a correct case to be returned back into the retrieval phase to disambiguate any wrong answer produced by CBR.

As shown in **Fig. 3**, we start to remove one case from the case based library of the CBR until the system retrieves two different classes with the same similarity. The new method adapts the CARs to produce the FP-tree considering a class label, length of subsets and support. This is because in mining association rule algorithms, any

associated method does not consider class clusters and length in the process of producing frequent patterns of a specific class. Thus, in experiments to date an attempt has been made to develop a FP-tree to make the frequent rules more effective to one class by using a parent root of each class label. As a consequence of that, every frequent rule will belong to its class. In the experiments, the first step of the FP-tree algorithm is changed to classify subsets according to its frequency before the rules are produced. Hence, considering the new case as a pattern to be compared with the constructed FP-tree will provide a correct match based on the new case built from the new tree. In other words, if a new case arrives to CBR, SBR may retrieve unrelated cases from the case library with same similarity measures as shown **Fig. 3** in the retrieved cases field. This ambiguous result can make it difficult for the CBR user to take the right decision. Following that, we produce CARs from the same case library in order to gain the FP-CAR tree. The new case will then be compared to the formed tree to find a match which may belong to class root.

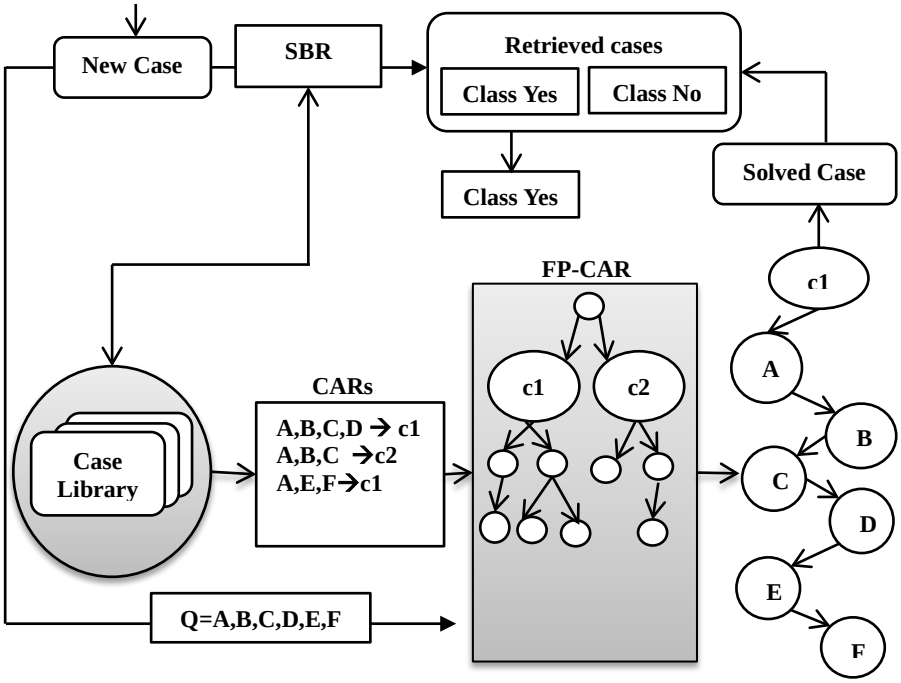


Fig. 3. CBRAR Model

The proposed strategy is compared to existing CBR tools in the following steps:

- Splitting: the new algorithm splits rules into different classes, where each rule represents a subset which belongs to a particular class.
- Comparing: the new algorithm compares a CBR query as a pattern which actually represents a new case; it should match exactly a frequent path FP-tree.

- Voting: the process of voting is performed by considering the longest length of the nodes considering values of the modified FP-tree in terms of finding a partial match.
- P-trees: a P-trees procedure is invoked to complete any missing nodes in the tree if needed to form an equivalent pattern to the CBR query.

In the final step, the result obtained by our new model is compared with the outcomes of the retrieval phase to select a correct answer. We compare the solved case with the result of the retrieved cases to remove unrelated answers as shown in **Fig. 3.** It can be seen that two different labels i.e. class (yes and no) are retrieved by CBR in the retrieved cases field. By returning the solved case into the retrieved cases phase, the ambiguity of the SBR outcomes was removed.

6 Experimental Results

To investigate the accuracy of CBRAR, we conducted experiments using a dataset taken from the UCI Machine Learning Repository. The implementation of CBRAR used a Java platform Eclipse (4.5.0), and for comparison purposes we have used the Jcolibri framework [17] and FreeCBR [18] as powerful CBR tools. WEKA 3.6 is used as an open source in order to generate the CARs. In the set of experiments, we have removed one case from the CBR case library to be considered as a new case in each run of both Jcolibri and FreeCBR. We used the acute inflammations dataset as the same source to measure the CBR and CBRAR accuracy. By default, SBR returns the 5 most similar answers when using Jcolibri when a new case is applied. However, the pre-determined cases 73,76,85,88 have registered an ambiguity that misleads the decision maker as all retrieved cases have the same percentage of similarity with different labels i.e. (yes, no). When using FreeCBR, more potential cases were identified in addition to those found by Jcolibri.

The results are shown in **Table 2**; vertically, the first column refers to the new case *Q* followed by the cases retrieved by the CBR tools i.e. NewCase73 followed by cases (71, 72, 76, 77 and 79, for Jcolibri) and cases (71, 72, 77 until 107 for FreeCBR). The "Attributes" columns start with a temperature attribute *F* followed by 5 additional attributes *A,B,C,D* and *E*. The class label column indicates a diagnosis of Inflammation of the urinary bladder with values (yes and no). The "Accuracy" columns show the comparison between Jcolibri, FreeCBR and CBRAR. In the table, we use symbols TP, TN, FP and FN as follows True Positive, True Negative, False Positive and False Negative. The assumption is made to indicate the four probabilities on the confusion matrix. **Table 2** shows that, for each new case applied to CBR, 5 different cases are retrieved by Jcolibri with the same similarity ratio i.e. 0.912. In the first experiment, a NewCase73 applied to the CBR, Jcolibri retrieved 3 TP and 2 FP cases with the same similarity ratio, and this is equal to 60% of accuracy, whereas FreeCBR retrieved 9 TP and 2 FP, and this is equal to 81% of accuracy. CBRAR retrieved 1 TP case from new model. In the second experiment, a NewCase76 applied to the CBR, Jcolibri retrieved 4 TN and 1 FN with same similarity and this is equal to 80% of accuracy, whereas FreeCBR retrieved 6 TN and 1 FN and this is equal to 86% of accuracy.

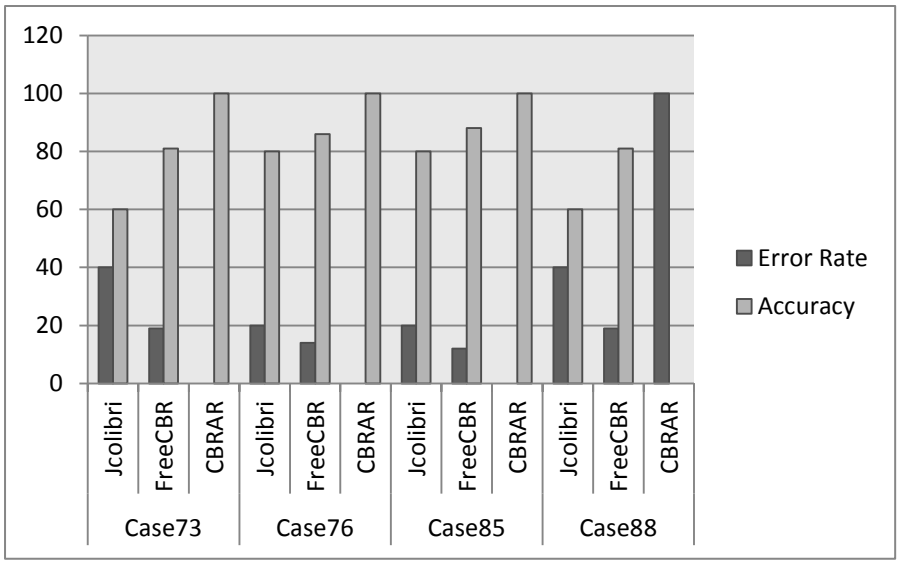
Table 2. Results of Wrong Retrieved Cases

Cases	Attributes							Accuracy		
	F	A	B	C	D	E	Class	Jcolibri	FreeCBR	CBRAR
NewCase73	**40.0**	**yes**	**yes**	**yes**	**yes**	**no**	**yes**	**0.912**	**59.1751**	**TP**
Case71	40.0	yes	yes	yes	yes	yes	yes	TP	TP	
Case72	40.0	yes	yes	yes	yes	yes	yes	TP	TP	
Case76	40.0	yes	yes	no	yes	no	no	FP	FP	
Case77	40.0	yes	yes	no	yes	no	no	FP	FP	
Case79	40.1	yes	yes	yes	yes	no	yes	TP	TP	
Case85	40.4	yes	yes	yes	yes	no	yes		TP	
Case86	40.4	yes	yes	yes	yes	no	yes		TP	
Case89	40.5	yes	yes	yes	yes	no	yes		TP	
Case94	40.7	yes	yes	yes	yes	no	yes		TP	
Case100	40.9	yes	yes	yes	yes	no	yes		TP	
Case107	41.1	yes	yes	yes	yes	no	yes		TP	
NewCase76	**40.0**	**yes**	**yes**	**no**	**yes**	**no**	**no**	**0.912**	**59.1751**	**TN**
Case73	40.0	yes	yes	yes	yes	no	yes	FN	FN	
Case82	40.2	yes	yes	no	yes	no	no	TN	TN	
Case88	40.4	yes	yes	no	yes	no	no	TN	TN	
Case92	40.6	yes	yes	no	yes	no	no	TN	TN	
Case96	40.7	yes	yes	no	yes	no	no	TN	TN	
Case104	41.0	yes	yes	no	yes	no	no		TN	
Case109	41.1	yes	yes	no	yes	no	no		TN	
NewCase85	**40.4**	**yes**	**yes**	**yes**	**yes**	**no**	**yes**	**0.912**	**55.278**	**TP**
Case73	40.0	yes	yes	yes	yes	no	yes	TP	TP	
Case79	40.1	yes	yes	yes	yes	no	yes	TP	TP	
Case84	40.4	yes	yes	yes	yes	yes	yes	TP	TP	
Case88	40.4	yes	yes	no	yes	no	no	FP	FP	
Case89	40.5	yes	yes	yes	yes	no	yes	TP	TP	
Case94	40.7	yes	yes	yes	yes	no	yes		TP	
Case100	40.9	yes	yes	yes	yes	no	yes		TP	
Case107	41.1	yes	yes	yes	yes	no	yes		TP	
NewCase88	**40.4**	**yes**	**yes**	**no**	**yes**	**no**	**no**	**0.912**	**55.276**	**FN**
Case76	40.0	yes	yes	no	yes	no	no	TN	TN	
Case77	40.0	yes	yes	no	yes	no	no	TN	TN	
Case82	40.2	yes	yes	no	yes	no	no	TN	TN	
Case85	40.4	yes	yes	yes	yes	no	yes	FN	FN	
Case86	40.4	yes	yes	yes	yes	no	yes	FN	FN	
Case92	40.6	yes	yes	yes	yes	no	yes		TN	
Case96	40.7	yes	yes	yes	yes	no	yes		TN	
Case104	41.0	yes	yes	yes	yes	no	yes		TN	
Case109	41.1	yes	yes	yes	yes	no	yes		TN	
Average								**70**	**83**	**75**

CBRAR retrieved 1 TN case from the suggested algorithm. When NewCase85 is applied to the CBR in the third experiment, Jcolibri retrieved 4 TP and 1 FP cases with the same similarity percentage, and this is equal to 80% accuracy whilst FreeCBR retrieved 7 TP and 1 FP and this is equal to 88% of accuracy. CBRAR retrieved 1 TP case from FP-CAR tree. In the fourth experiment, a NewCase88 applied using Jcolibri again retrieved 5 cases with 3 TN and 2 FN with same similarity and this is equal to 60% accuracy. FreeCBR retrieved 9 cases with 7 TN and 2 FN. CBRAR incorrectly retrieved 1 FN as a wrong case.

The results show that 14 out of the 20 Jcolibri retrieved cases are classified as TP and TN giving 70% accuracy. By comparison, 29 of the 35 cases retrieved by FreeCBR are classified as TP and TN giving 83% accuracy. However, both Jcolibri and FreeCBR deliver "confusing" results. Our CBRAR strategy demonstrates advantages over both Jcolibri and FreeCBR by resolving 3 out of 4 cases with 75% accuracy and no confusion. The accuracy of CBRAR was better compared to Jcolibri and FreeCBR. CBRAR resolved the ambiguity of the FP and FN cases without confusion. Cases 73, 76 and 85 in **Table 2** can be reworked in **Fig. 2** to prove that CBRAR identifies a correct case using a frequent classed tree.

Fig. 4. Error Rate and Accuracy



The bar chart in **Fig. 4** illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, it is clear that in Case73, CBRAR registered 0 error rate, which is the lowest among the rates (40, 19) when compared to Jcolibri and FreeCBR. The results also show that the error rate of CBRAR is the lowest on Case76 and Case85 thus giving the highest accuracy, when compared to the other CBR tools used. CBRAR also correctly resolved 3 out of 4 cases. In Case88, it noticeable that the (40, 19) % error rate of Jcolibri and FreeCBR was considerably lower than CBRAR.

However, whilst CBRAR did not resolve Case88 neither of the other CBR tools offered any advantage when compared to the new model. In conclusion, we have shown that the other CBR tools used inherit the same problem of error rates, whereas CBRAR has shown a better performance in overall error rate.

7 Conclusion

This paper has presented a new approach, CBRAR, to improve the performance of SBR. The CBRAR approach includes a new algorithm FP-CAR which produces far fewer frequent classed subsets than would be produced from a generic FP-tree. It uses a new method of length voting compared to the TFPC algorithm where a value of nodes is considered whilst building the tree. Moreover, the subsets left on the tree that meet the support, confidence and longest length of pattern can be used to classify subsets when sorted in a hash table. A superset could be derived; to be compared with other new CBR cases when compared with the CBR tools Jcolibri and FreeCBR, the CBRAR strategy achieves a better accuracy level with the lowest error rate. Moreover, the experimental results have shown the advantages of CBRAR over Jcolibri and FreeCBR in terms of uncertain answers which are retrieved with same similarity. The next phase of our work will extend our experimental results by implementing CBRAR on different datasets and comparing the results with the other CBR tools used for our experiments to date.

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Commun. 7, 39–59 (1994).
2. Perner, P.: Introduction to Case-Based Reasoning for Signals and Images. In Perner, P. (Ed). Case-Based Reasoning on Signals and Images, 1-24 (2008).
3. 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).
4. Ma, B., Liu, W., Hsu, Y.: Integrating classification and association rule mining. In: Proceedings of the 4th Knowledge Discovery and Data Mining (1998).
5. Chen, G., Liu, H., Yu, L., Wei, Q., Zhang, X.: A new approach to classification based on association rule mining. Decis. Support Syst. 42, 674–689 (2006).
6. Vo, B., Le, B.: A novel classification algorithm based on association rules mining. In: Richards, D. and Kang, B.-H. (eds.) Knowledge Acquisition: Approaches, Algorithms and Applications. pp. 61–75. Springer (2009).
7. Deng, H., Runger, G., Tuv, E., Bannister, W.: CBC: An associative classifier with a small number of rules. Decis. Support Syst. 59, 163–170 (2014).

8. Nguyen, L.T.T., Vo, B., Hong, T.-P., Thanh, H.C.: CAR-Miner: An efficient algorithm for mining class-association rules. Expert Syst. Appl. 40, 2305–2311 (2013).
9. Ibrahim, S.P.S., Chandran, K.R., Kanthasamy, C.J.K.: CHISC-AC: Compact Highest Subset Confidence-Based Associative Classification[1]. Data Sci. J. 13, 127–137 (2014).
10. Nguyen, L.T.T., Nguyen, N.T.: An improved algorithm for mining class association rules using the difference of Obidsets. Expert Syst. Appl. 42, 4361–4369 (2015).
11. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann, San Mateo, Calif (1993).
12. Tolun, M.R., Abu-Soud, S.M.: ILA: an inductive learning algorithm for rule extraction. Expert Syst. Appl. 14, 361–370 (1998).
13. Tolun, M.R., Sever, H., Uludag, M., Abu-Soud, S.M.: ILA-2: An inductive learning algorithm for knowledge discovery. Cybern. Syst. 30, 609–628 (1999).
14. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. 20th int. conf. very large data bases, VLDB. pp. 487–499 (1994).
15. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: ACM SIGMOD Record. pp. 1–12. ACM (2000).
16. Cagliero, L., Garza, P.: Infrequent weighted itemset mining using frequent pattern growth. Knowl. Data Eng. IEEE Trans. 26, 903–915 (2014).
17. jCOLIBRI | GAIA – Group of Artificial Intelligence Applications, http://gaia.fdi.ucm.es/research/colibri/jcolibri.
18. FreeCBR, http://freecbr.sourceforge.net/index.shtml.
19. Ahn, H., Kim, K.: Global optimization of case-based reasoning for breast cytology diagnosis. Expert Syst. Appl. 36, 724–734 (2009).
20. Pandey, B., Mishra, R.B.: Case-based reasoning and data mining integrated method for the diagnosis of some neuromuscular disease. Int. J. Med. Eng. Inform. 3, 1–15 (2011).
21. Lorenzi, F., Ricci, F.: Case-based recommender systems: a unifying view. In: Mobasher, B. and Anand, S.S. (eds.) Intelligent Techniques for Web Personalization. pp. 89–113. Springer, Berlin (2005).
22. Beddoe, G.R., Petrovic, S.: Selecting and weighting features using a genetic algorithm in a case-based reasoning approach to personnel rostering. Eur. J. Oper. Res. 175, 649–671 (2006).
23. Althof, K.-D., Auriol, E., Barlette, R., Manago, M.: A Review of Industrial Case Based Reasoning. AI Intelligence, Oxford (1995).
24. Cercone, N., An, A., Chan, C.: Rule-induction and case-based reasoning: hybrid architectures appear advantageous. IEEE Trans. Knowl. Data Eng. 11, 166–174 (1999).
25. Huang, M.-J., Chen, M.-Y., Lee, S.-C.: Integrating data mining with case-based reasoning for chronic diseases prognosis and diagnosis. Expert Syst. Appl. 32, 856–867 (2007).
26. Park, Y.-J., Choi, E., Park, S.-H.: Two-step filtering datamining method

integrating case-based reasoning and rule induction. Expert Syst. Appl. 36, 861–871 (2009).
27. Bradley, K., Smyth, B.: Personalized information ordering: a case study in online recruitment. Knowledge-Based Syst. 16, 269–275 (2003).
28. Vong, C.M., Wong, P.K., Ip, W.F.: Case-based classification system with clustering for automotive engine spark ignition diagnosis. In: Computer and Information Science (ICIS), 2010 IEEE/ACIS 9th International Conference on. pp. 17–22. IEEE (2010).
29. Azuaje, F., Dubitzky, W., Black, N., Adamson, K.: Discovering relevance knowledge in data: a growing cell structures approach. Syst. Man, Cybern. Part B Cybern. IEEE Trans. 30, 448–460 (2000).
30. Zhuang, Z.Y., Churilov, L., Burstein, F., Sikaris, K.: Combining data mining and case-based reasoning for intelligent decision support for pathology ordering by general practitioners. Eur. J. Oper. Res. 195, 662–675 (2009).
31. P. Perner, Prototype-based classification, App. Intell., 28(3), pp. 238–246, (2008).
32. Chuang, C.-L.: Case-based reasoning support for liver disease diagnosis. Artif. Intell. Med. 53, 15–23 (2011).
33. Guo, Y., Hu, J., Peng, Y.: Research on CBR system based on data mining. Appl. Soft Comput. 11, 5006–5014 (2011).
34. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: ACM SIGMOD Record. pp. 207–216. ACM (1993).
35. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: A survey. ACM Comput. Surv. 38, 9 (2006).
36. Aparna, V./, Ingle, M.: Enriching Retrieval Process for Case Based Reasoning by using Vertical Association Knowledge with Correlation. Int. J. Recent Innov. Trends Comput. Commun. 2, 4114 – 4117 (2014).
37. Nahm, U.Y., Mooney, R.J.: Using soft-matching mined rules to improve information extraction. Language (Baltim). 11, 50 (2004).
38. Kang, Y.-B., Krishnaswamy, S., Zaslavsky, A.: A Retrieval Strategy for Case-Based Reasoning Using Similarity and Association Knowledge. IEEE Trans. Cybern. 44, 473–487 (2014).
39. Patel, D.: A Retrieval Strategy for Case-Based Reasoning using USIMSCAR for Hierarchical Case. Int. J. Adv. Eng. Res. Technol. 2, 65–69 (2014).
40. TFPC Classification Association Rule Mining (CARM) Software, https://cgi.csc.liv.ac.uk/~frans/KDD/Software/Apriori-TFPC/Version2/apriori TFPC.html.
41. Coenen, F., Leng, P., Ahmed, S.: Data structure for association rule mining: T-trees and P-trees. IEEE Trans. Knowl. Data Eng. 774–778 (2004).
42. Goulbourne, G., Coenen, F., Leng, P.: Algorithms for computing association rules using a partial-support tree. Knowledge-Based Syst. 13, 141–149 (2000).
43. Coenen, F., Goulbourne, G., Leng, P.: Tree structures for mining association rules. Data Min. Knowl. Discov. 8, 25–51 (2004).