A NEW STRATEGY FOR CASE-BASED REASONING RETRIEVAL USING CLASSIFICATION BASED ON ASSOCIATION

AHMED ALJUBOORI

School of Computing, Science and Engineering University of Salford, UK

Submitted in Partial Fulfilment of the Requirements of the Degree of Doctor of Philosophy, 2017

Table of Content

List of Figures

List of Tables

List of Equations

List of Abbreviations

AK	Association Knowledge
ARM	Association Rules Mining
ARs	Association Rules
CAR	Class Association Rules
CBR	Case-Based Reasoning
CBRAR	Case Based Reasoning using Association Rules Mining
DK	Domain knowledge
DM	Data Mining
FP-CAR	Frequent Pattern Class Association Rules
FPM	Frequent Pattern Mining
FP-tree	Frequent Pattern tree
FS	Feature Selection
FW	Feature Weighting
KDD	Knowledge Discovery in Database
KNN	K- Nearest Neighbor
ML	Machine Learning
NP	New Patient
P-trees	Partial Trees
RACER	Rule-Associated Case-based Reasoning
RI	Rule Induction
SARM	Soft Matching of Association Rules
SBR	Similarity-Based Retrieval
SCAR	Soft Matching of Class Association Rules
SK	Similarity Knowledge
USIMCAR	Retrieval Strategy Based on the Association Knowledge
WPI	Worcester Polytechnic Institute

Acknowledgements

First and above all, I praise God, the almighty for providing me this opportunity and granting me the capability to proceed successfully.

To my father soul, my mother for her prayers and support to take care of two children in Iraq, my children who have been patient without their parents' care throughout this process.

My endless love, respect and appreciation are for my wife who stood by me every single difficult moment. She always provides faith, assistance and patient whoever it was necessary.

I would like to thank Professor Farid Meziane who has lit the way for me through his guidance. His door was open every single time I went to ask why and how. Thanks to him to support me when I was about to give up but because of his knowledge and professional supervision I did not.

I owe my deepest gratitude to Professor David Parson. I was honoured to have his advice and will never forget how he helped me by adding a lot of incomes to my PhD. I also was lucky to have his continuous stream of ideas and endless experience of science and life.

Lastly, without the assistance of those who love and care, this work would have never seen the light. I am grateful from everyone who believed in me while I was working to my goal.

Parts of the current PhD research have resulted in the following publications.

Aljuboori, A. (2016), Enhancing Case-Based Reasoning Retrieval Using Classification Based on Associations, Proceedings of the 6th International Conference on Information Communication and Management (ICICM 2016). IEEE, pp: 52-56, October 29-31, 2016, University of Hertfordshire, UK.

Aljuboori, A., Meziane, F. and Parsons, D. (2016), A new strategy for case-based reasoning retrieval using classification based on association, Proceedings of the 12th International Conference on Machine Learning and Data Mining (MLDM 2016), Springers, pp: 326-340, July 16-21, 2016, New York, USA.

Aljuboori, A., Meziane, F. (2015). Integrating association rules into case-based reasoning. Salford Postgraduate Annual Research Conference (SPARC). University of Salford, UK.

Aljuboori, A., Meziane, F. (2015). Integrating association rules into case-based reasoning. Dean's Annual Research Showcase. University of Salford, UK.

Abstract

Cased Based Reasoning (CBR) is an important area of research in the field of Artificial Intelligence. It aims to solve new problems by adapting solutions, that were used to solve previous similar ones. Among the four typical phases - retrieval, reuse, revise and retain, retrieval is a key phase in CBR approach, as the retrieval of wrong cases can lead to wrong decisions**.** To accomplish the retrieval process, a CBR system exploits Similarity-Based Retrieval (SBR). However, SBR tends to depend strongly on similarity knowledge, ignoring other forms of knowledge, that can further improve retrieval performance.

The aim of this study is to integrate class association rules (CARs) as a special case of association rules (ARs), to discover a set (of rules) that can form an accurate classifier in a database. It is an efficient method when used to build a classifier, where the target is pre-determined.

The proposition for this research is to answer the question of whether CARs can be integrated into a CBR system. A new strategy is proposed that suggests and uses mining class association rules from previous cases, which could strengthen similarity based retrieval (SBR). The proposition question can be answered by adapting the pattern of CARs, to be compared with the end of the Retrieval phase. Previous experiments and their results to date, show a link between CARs and CBR cases. This link has been developed to achieve the aim and objectives.

A novel strategy, Case-Based Reasoning using Association Rules (CBRAR) is proposed to improve the performance of the SBR and to disambiguate wrongly retrieved cases in CBR. CBRAR uses CARs to generate an optimum frequent pattern tree (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.

In addition, CBRAR has been evaluated using two sources of CBR frameworks - Jcolibri and Free CBR. With the experimental evaluation on real datasets indicating that the proposed

CBRAR is a better approach when compared to CBR systems, offering higher accuracy and lower error rate.

Chapter 1: Introduction

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

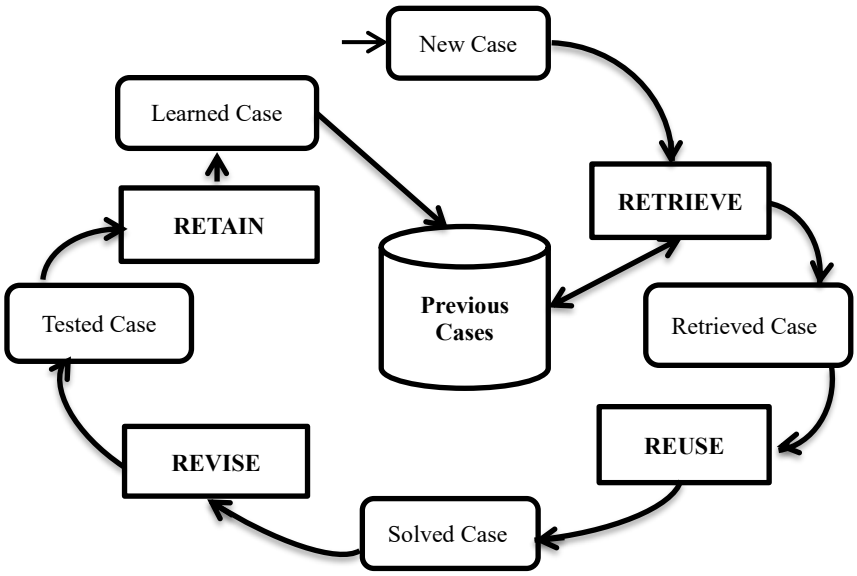


Figure 1 CBR Cycle [1]

Fundamentally, retrieval is performed through a specific strategy of leveraging similarity knowledge (SK) referred to as 'similarity-based retrieval' (SBR) [2]. 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. The solution is basically "associated" to the closest case to enable users to determine the rank of cases [3].

Association rules mining (ARM) is an important technique in the field of data mining (DM). It aims at extracting interesting correlations, frequent patterns, 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 and department stores. [4] Describes an example of an association rule using the following example, "If a customer buys a dozen eggs, they are 70% likely to also purchase milk." meaning that it is possible to determine consumer behaviour and predictions by analysing ARs. Thus, ARM plays a major role in the shopping basket data analysis, product clustering and the design of catalogues and store layouts.

The class association rule technique was first proposed by [5]. It generates classification rules based on association rules as an integration of classification and association. The integrated framework of CARs suggested by [5] is achieved by discovering a special subset of ARs whose right side of the implication equation are confined to the classification class label. Other techniques for mining CARs have been suggested in recent years. They include GARC [6], ECR-CARM [7], CBC [8], CAR-Miner [9], CHISC-AC [10] and d2O [11]. The methods of classification based on CARs were demonstrated to be more accurate than the classic methods e.g. C4.5 [12] and ILA [13], [14] in their experimental results [5]. The concept of classification based on association has been employed in this project to show that patterns of classed rules can be combined to form a similar pattern, to be compared to CBR problem.

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 [15],[16]. 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 [17]. 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 dataset structure using the FP-tree and extracts the frequent pattern directly from this structure [18]. 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 the stored data being accessed. In this project, FP-tree and an implications table are used to produce a compressed tree of CARs in order to find a CBR case problem pattern in the tree.

The work presented in this research concerns the mining of the rules which can disambiguate the retrieved answers of existing case based reasoning systems. Ultimately, the originality and contribution of this work is to highlight that when DM and CBR are combined in a unified way, the cases to be mined will be mined more efficiently. This research will also employ association rules as one of the DM approaches that could be used to improve the performance of the retrieval process. Furthermore, techniques will be developed to allow different rules to be combined in order to produce a correct case not just a similar one.

The research will be validated through extensive experiments using up to date and valid datasets in various areas. These include different applications relating to for example: medicine, the influence of prior knowledge on concept acquisition and a NASA space dataset. The data being used for this research has been published on the UCI website to facilitate the preparation

of artificial intelligence systems algorithms. The datasets are used to evaluate the new strategy CBRAR 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. The experimental results in chapter 4 show that the CBRAR strategy is able to disambiguate the answers of the retrieval phase compared to those obtained when using Jcolibri [19] and FreeCBR [20] systems.

1.1. Research Problem

Basically, the retrieval phase in a CBR system is achieved via a specific strategy described by [2] and known as similarity-based retrieval (SBR) to estimate the benefit of stored cases relating to a target problem. It is ordinarily encoded using similarity measures between the problem and stored cases. Thus, cases ranked by their similarity measures to the solution are then used to resolve new problems.

However, there are two major drawbacks to SBR. The first issue, according to [21], is that in practice SBR is reliant on domain experts to clarify SK. Defining SK is still complex, hard to practice and time consuming as no obvious methodology or any general approaches to support the modelling of measures in an intelligent way have yet to be developed. This often leads SK to being subjective and inaccurate. The second issue is that the definition of similarity measures is often static. So, it is highly possible that it could be continuously applied to all target problems. This leads to a problematic situation where a similarity criterion defined in a given field is beneficial for some target problems but not for others. Therefore, depending on target problems, the retrieval performance of SBR is different even within the same domain [22].

This research, attempts to enhance the performance of CBR system and address the most common problem of the retrieval phase [23]. Thus, it was crucial to address this issue by retrieving not just the most similar case, but also the one with the greatest benefit. In addition, removing the SBR limitation can save life, in some critical domains such as health, time and money. As shown in Table 1, the research problem is explained through a simple medical diagnosis table similar to the case study in [24], where the weight of each attribute was uniform for the sake of simplicity and the CBR retrieved different class labels with the same percentage of similarity. Consider that the case base includes four patient cases.

Table 1 Medical Case Study

Attribute	Weights(*wi*)	Patient 1	Patient 2	Patient 3	Patient 4	New Patient NP
Temperature	1	40.0	40.0	40.1	40.4	40.0
Occurrence of Nausea	1	yes	yes	yes	yes	yes
Lumber Pain	1	yes	yes	yes	yes	yes
Urine Pushing	1	yes	no	yes	yes	yes
Micturition Pain	1	yes	yes	yes	yes	yes
Burning of Urethra	1	yes	no	no	no	no
Diagnosis		yes	no	yes	yes	?
Similarity with NP		0.83	0.83	0.83	0.83	

For each case, the problem is described by six attributes (symptoms) A1 to A6, and the solution denotes the corresponding diagnosis. The aim is to make a correct diagnosis for a new patient (NP). It is pre-determined that the NP is really suffering from acute inflammation of the urinary bladder as specified in the case base as a class label yes. Therefore, to predict a diagnosis for the NP, in principle, SBR identifies similar cases to the NP by finding cases whose attributes are similar. The following metric is applied to measure the similarity between NP and each case $C \in D$, D is a dataset, as case base used by [23].

$$\boldsymbol{sim_g(NP, C)} = \frac{\sum_{i=1}^{n} \boldsymbol{w_i} \cdot \boldsymbol{sim_1(q_i, c_i)}}{\sum_{i=1}^{n} \boldsymbol{w_i}}$$

Equation 1 Similarity Metric Measure

Where

$$sim_1(q_i, c_i) = \begin{cases} 1 - \frac{|\, q_i - c_i \,|}{A_i^{max} - A_i^{min}} \\ 1 \text{ if } q_i = c_i \text{ , and 0, otherwise (if } A_i \text{ is nominal), Attribute's value} \end{cases}$$

Where *n* is the number of attributes of cases, A_i^{max} and A_i^{min} indicate the maximum and minimum values. *Ai* is the attribute of NP and the case *C* contains *qi* and *ci*, respectively that *Ai* takes on in *D*.

Once similar cases to the NP are chosen, SBR determines a diagnosis for the NP using these cases, assuming that, SBR selects the single most similar case to the NP. As shown in Figure 2 and Table 2, all patients are chosen when applying the above metric and recorded the same similarity measure of 0.8333334. The cases retrieved presented contradictory solutions with some cases having the "yes" label and others the "no" label. Hence, from the solution it is not clear which outcome should be associated with the NP. In this specific case, we know in advance that the NP is from an inflammation of the urinary bladder and should be labelled as "yes". The CBR in this case is suggesting an incorrect solution that may affect the outcome of the diagnosis. This case illustrates the limitations of SBR as it relies only on the similarity measure.

In comparison to similarity, AK can be obtained through class association rule mining. A key feature of AK being that it is objective as it is built from general rules of associations between known problem features and solutions.

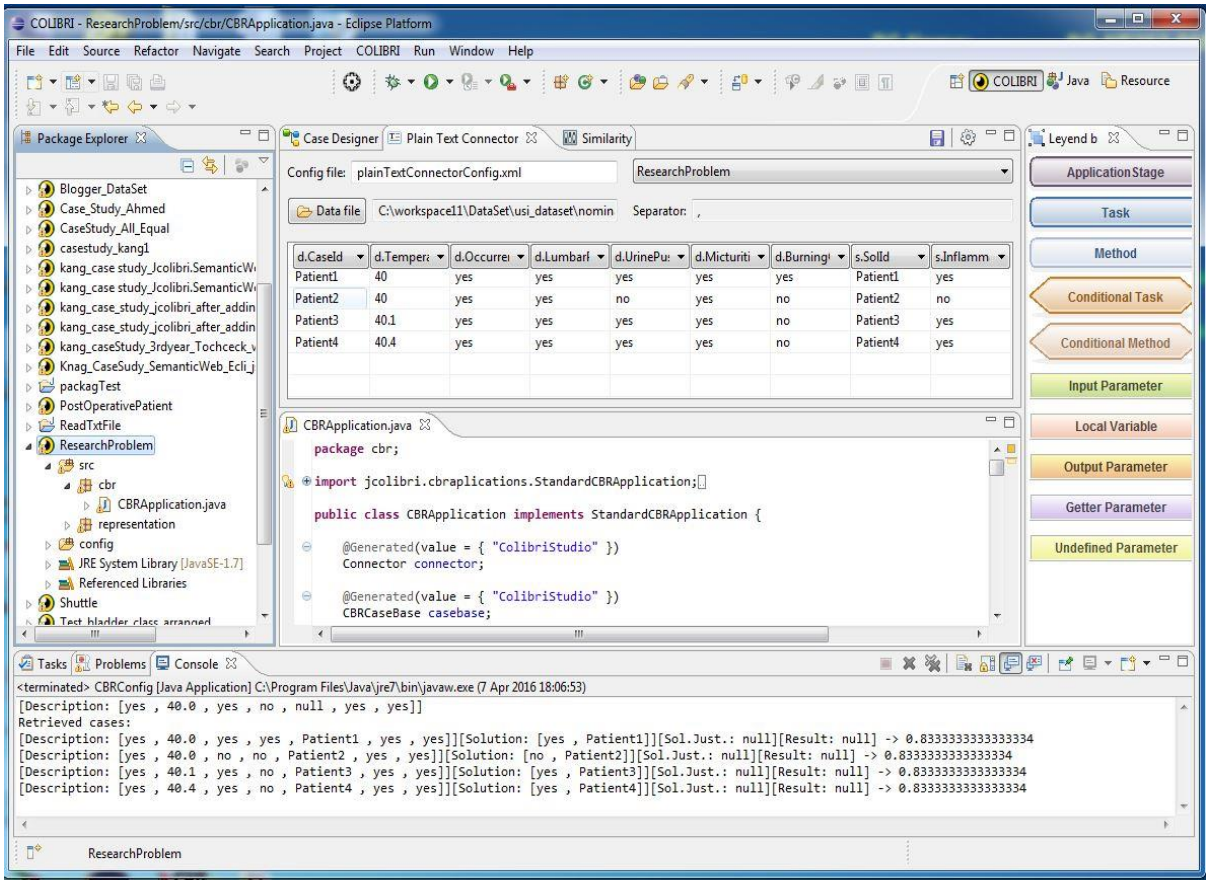


Figure 2 Jcolibri Similarity Results

Table 2 Results of Similarity

Similarity	Percentage	Class
Sim(Patient 1, New Patient)	0.8333334	yes
Sim(Patient 2, New Patient)	0.8333334	no
Sim(Patient 3, New Patient)	0.8333334	yes
Sim(Patient 4, New Patient)	0.8333334	yes

In line with the benefits mentioned above, this research introduced a novel system, which aims to improve the retrieval phase and take into consideration CARs to improve the retrieval phase, which is considered key in retrieving the most similar case. This novel strategy is called CBRAR. Therefore, a key strength of CBRAR is the use of AK to complement SK, thereby significantly enhancing the performance of SBR.

1.2. Motivations

This research was motivated by the now widely acknowledged potential retrieval phase problem in CBR systems. In recent years, researchers have worked to improve a SBR performance but have tended to ignore other types of domain knowledge, where CBR has been more successful. Researchers have also endeavoured to develop improved machine learning algorithms such as Association rules, frequent pattern trees, partial trees and K-Nearest Neighbour. Our research suggests that most CBR methods and ARs algorithms have been developed separately. More recently, some researchers have recognised that CARs as well as FP-trees can be used to improve CBR performance. In addition, the similarity in the case base is poor and subjective, where the CARs algorithm determines the correlation of the objective rules. Therefore, there is a real need to develop a strategy for integrating CARs into CBR in situation where a wrong case is retrieved

This research is concerned with developing and evaluating a new case based reasoning retrieval approach using classification based on association. More specifically, when CBR returns cases with different classes and same accuracy, the proposed strategy mines all CARs using a compressed tree to gain a similar pattern compared to a new CBR case problem. This novel approach is used to disambiguate wrongly retrieved answers in order to facilitate a more correct decision.

1.3. Research Aim and Objectives

Given the above motivation, the aim of this PhD was to construct a new strategy to improve the performance of SBR and to achieve high accuracy in the retrieval phase through leveraging AK in CBR systems. In addition, this system was applied to real problems using different datasets, to retrieve the best similar case from a case base. The specific research objectives were:

1. To carry out an in depth, comprehensive literature review on the existing DM techniques especially ARs, and their application into CBR.
2. To review literature on existing CBR systems, and identify how CBR and ARM can be merged together into this type of study.
3. To develop a CBRAR based on a strategy that is able to retrieve the most similar case by integrating CARs into CBR.
4. To develop an FP-CAR algorithm that is able to mine CARs into a frequent tree to produce a pattern which matches a new CBR case problem.
5. To implement this strategy on real datasets whilst carrying out an empirical evaluation of the proposed system.
6. To conduct an empirical evaluation of the new strategy against existing systems such as Jcolibri [25] and FreeCBR [20] and to measure its accuracy in terms of retrieving the best similar cases.

1.4. Research Methodology

In carrying out this research, several methods have been examined to determine which one was the most suitable. The following research methodologies were investigated:

- **Fundamental versus Applied**

Fundamental studies focus on the establishment of hypotheses or a theory definition. It includes developing a new algorithm or a new mathematical framework. In contrast, applied researches utilises different accumulated theories as an effort to overcome a problem faced by businesses or practical application [26], [27]. Therefore, fundamental research aims to find information that has a broad base of applications to add to the body of scientific knowledge, whereas the applied research is directed to discover a solution to a pressing imperative problem.

- **Descriptive research vs. Analytical research**

Descriptive research is the study that explains the present state without controlling any inputs of the variables. The major purpose of this type of research is that the researcher describes the state of the art which exists at present. The researcher can only explain the facts of the theory and the factors that are affecting this theory with regard to what has occurred or explaining what happened to a specific phenomenon. Therefore, the descriptive research does not consider the study results validity as it does not describe the result causes [28], while in terms of the analytical research, the question is asked as to why we have this result, or, why it is that way. This is carried out through critical assessment for the state of the art by incorporating different inputs to complete a critical evaluation of the results [26], [29].

- **Conceptual vs. Empirical and Scientific methods**

Conceptual studies are conducted to describe a new theory or concept that explains a problem being studied for example, the cause behind a particular disease. This is referred to as a pen and paper approach, where the researcher carries out no experiments but utilises the observation of others, which are then either proven or disproven.

Empirical studies include a number of experiments conducted in an effort to validate an existing theory. It also derives knowledge from the experience that was based on direct and indirect observations. For some researches, a researcher has a complete control over the experiment's design and variables, while adhering to the existed algorithm and his needs [26]. In contrast, a scientific approach is a combination of both conceptual and empirical research, using the formulation of a hypothesis, with experiments then designed with the aim of testing the prediction to support or disprove the hypothesis. Edison for example, used an empirical approach by using trial and error considering the work of some theorists. The current study falls under the method of proving theory during the experimental completion and observations, decreasing the bias on

the outcomes and experiments [30].

Our research method is summarised in the steps listed below and included both conceptual and empirical approaches:

1. The study questions are outlined by emphasising the key motivation that has driven the research. That being, there is a wide acceptance that there is a potential problem in CBR retrieval, where data mining methods are required as a perfect complement to specify the best approach to solving the research problem.
2. Conducting a comprehensive literature review on existing DM methods, particularly CARs to identify possible approaches to address the problem of CBR retrieval.
3. Design and implement a new retrieval strategy CBRAR, that is appropriate for achieving the objective of this research, which involves a new algorithm FP-CAR that has the ability to combine CARs and extract a similar pattern which matches the new case that presented a problem in a CBR system.
4. The proposed strategy has been tested on different benchmarking datasets by performing a number of experiments in order to validate the system. Precision and recall were used to evaluate the efficiency of the proposed retrieval approach. A comparison with two other CBR systems i.e. Jcolibri and FreeCBR to check the reliability of CBRAR is performed. The conclusion sought from the research findings was to ensure that the study objectives were achieved with results that outperformed the retrieval phase of the CBR systems used to compare our results.

1.5. The Contribution of the Research

Although, much research has been previously carried out on both CBR and ARM, very few researchers have examined their integration. In recent years, some researchers have conducted research in ARs as a strategy to improve CBR performance. For instance, [31] suggested the

RACER system, which integrates CBR and association rules mining for supporting General Practitioners by prescribing the appropriate drug or the most appropriate therapy. Furthermore, [32] developed a retrieval strategy by analysing ARs for hierarchal cases and [24] proposed the USIMCAR technique to integrate Apriori algorithms into CBR.

For this research, an attempt was made to create a new retrieval technique by adapting the concepts of the algorithms in [33],[34] and [5], which previously focused on ARs separately and disregarded CBR systems. Consequently, a key initial part of this research contribution was to exploit class association rules instead of using general rules. The research also sought, to adapt an FP-Growth algorithm [33] to construct a frequent tree classified according to its label. The work also adapted [35] to gain an optimum tree in terms of a partial solution. The "combined three" algorithm is called CBRAR, with the objective of improving the performance of SBR. Several potentially different ways of adapting algorithms to use them in CBR were identified. These included: changing FP-tree into FP-CAR frequent pattern class association rules; changing the construction process and even adopting alternative measures in the algorithms that consider the association knowledge. The research explored these alternatives by implementing and evaluating them on various benchmark datasets. It also applied the new strategy to real problems in order to retrieve the most similar SBR cases to address the identified limitations of the CBR Retrieval phase.

1.6. Outline of Thesis

This section describes the outline of the thesis:

Chapter 1: Introduction:

Chapter 1 presents the introduction to the thesis, research problem, motivations, aim and objectives, methodology and contribution.

Chapter 2: Background and literature review:

Chapter 2 describes the background to the research area and includes:

- Case based reasoning approach (structure and phases).
- Association rules mining approaches and techniques covered in the literature (e.g. Apriori and Frequent pattern tree algorithms).
- Data structures for mining association rules (Partial trees).

Chapter 3: New Framework for Retrieval CBRAR Phase and new Algorithm FP-CAR

Chapter 3 shows the new CBR strategy (CBRAR) which includes (FP-CAR) - a newly developed algorithm which will build on the two different approaches that had already been developed and are presented in this thesis. The chapter also includes the new retrieval technique refinement and illustrations of our system.

Chapter 4: Results and Discussion:

Chapter 4 demonstrates the comprehensive empirical experiments of the proposed strategy. It also includes a comparison of the results obtained using CBRAR with two existing CBR systems. This evaluation is based on comparing the accuracy and the error rates of all the systems used for the experiments.

Chapter 5: Conclusions:

Chapter 5 presents the conclusions, including reflections on the extent to which the research objectives have been met. It also identifies any future potential developments which may be possible arising from the research carried out to date.

Chapter 2: Literature Review and Related Work

The fundamental aim of this research is to determine whether data mining association rules can be integrated into the retrieval phase of a CBR system. The purpose being to identify whether, by so doing, it is possible to improve the performance of the CBR retrieval phase in terms of both increased accuracy and reduced errors.

The Chapter therefore, includes literature and state of the art reviews for the following key areas:

- CBR background, applications, tools and techniques.
- Data mining common methods including the classification metrics related to this research.
- Association rules algorithms that utilised in this research in order to produce CARS.
- Frequent pattern and tree structured mining relevant methods that have been used in other researches in order to mine all CARs.
- Related CBR work and other types of Knowledge.

The Chapter concludes with the hypothesis that by integrating data mining methods, in particular association rules, into CBR it is possible to improve both the efficiency and the effectiveness of the CBR retrieval phase.

2.1. CBR Background

This section surveys CBR and its Background, parts of a case, case representation, case bases and retrieval.

2.1.1. Cased-Based Reasoning Background

CBR is a well-studied area in machine learning. In the past decade several researchers have studied CBR methods in real world applications, such as medical diagnosis[36],[37], IT service management [38], product recommendation [39] and personal rostering decisions [40]. CBR is a cyclic and integrated process of solving a problem and learning from the experience of experts, which is used to build 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"[41].

The case-based reasoning term involves of three words and they need to be defined to have an overall understanding of this approach. Firstly, a case is fundamentally the experience of a solving a problem which can be represented in many various ways. A case base is a collection of such cases stored in the CBR system memory. Secondly, the term based indicates the reasoning that was based on cases which are the first source for reasoning. The term reasoning refers to the most characteristic approach of actions. It means, by utilising cases a conclusion can be drawn of the intended approach, given a problem to be solved. CBR basically differs in the way of solving a problem when compared with other main AI techniques. Rather than relying separately on the general knowledge to resolve a problem, or generalizing correlation between problem descriptors and association conclusions, CBR is able to utilize the particular knowledge of formerly experienced, real problem situations (cases). By finding a similar previous case a new problem can be solved, it will then be reused in the new problem situation.

Furthermore, CBR is also a technique to increment the learning knowledge since a new experience is stored each time a problem has been resolved, making it instantly obtainable for future problems. Over the last years a rapid growth has occurred in the CBR field, as can be seen by the increased share of papers at main conferences, in daily applications usage and commercial tools.

2.1.2. **Parts of a Case**

A CBR system utilises cases to solve problems, thus, the experience must involve problem information and its solution as two important parts. A problem part describes the current situation and a solution part explains the response to that problem. Occasionally, CBR limits a solution that has been successful, but that is not necessary sufficient. On the other hand, it is important to highlight that a failed solution is key information which can be used to avoid similar solutions it in the future. The study and understanding of both successful and failed experiences, guide users to positive and negatives experiences (cases). The positive experience is a successful solution made in which to advise the user to reuse the case again. The negative experience is to avoid the failed solution as a leading advice to future solutions [42]. As a result, the occurrence of positive and negative situations can produce C+ positive and C− cases. Negative cases could occur in terms of an advice that has to be considered or when a decision maker has to select from various alternatives. The main types of experience are listed below [43].

- Classification: is the process that assigns an object in a collection to a certain class based on its similarity to former examples. The goal is to precisely predict a class label for each object in the dataset.

- Diagnosis: is the process of identifying a problem by the examination of symptoms e.g. deciding whether what causes a laptop power supply to malfunction is lack of direct current or overuse.
- Prediction: is the task of predicting certain values for given input. For instance, predicting the weather forecasting for tomorrow in order to decide whether to play a football game or not according to a given month's records.
- Planning: plan travel according to a sequence of actions to reach a specific goal.
- Configuration: Select technical features and components to include specific elements for instance.

CBR systems are uniquely designed to tailor each types of experience but to consider one problem at a time. They are also established to achieve one reasoning task in each execution process. In contrast, a human can achieve more than one task as part of the experience in order to recognise the similarity between cases. In addition, reasoning produces additional components of knowledge in cases. This knowledge is basically counted as a case outcome i.e. how often cases are being used or successfully used to record meta-experience.

2.1.3. **Case Representation**

The easiest ways to represent cases are by using feature-value pairs. A value pair is used as a feature to represent a state of an entity, for instance, temperature of an entity, "Ahmed's temperature is normal", where the feature is the temperature of Ahmed and the value is normal, and the entity is Ahmed. In CBR systems, the word attribute is often used instead of the word "feature". Both problem and solution feature should be identified i.e. what problem may cause a headache if someone has specific symptoms as shown in Table 3. In addition, it can be seen that each patient represents a case, with attribute value as symptoms illustrated in Case1 column.

Table 3 Four Diagnosis Cases [42]

Attributes	Case id			
	Case 1	Case 2	Case 3	Case 4
Nausea	Yes	Yes	Yes	No
Fever	Yes	No	No	No
Malaise	Dizzy	Dizzy	Dizzy	Listless
Blood pressure	Normal	To low	High	Normal
Vision changes	No	Yes	No	No
Patient name	Bart	Marge	Lisa	Homer
Diagnosis	Influenza	Migraine	Heart	Stress

In fact, knowledge representation can be listed in three layers as shown in Figure 3. Firstly, at the cognitive layer the knowledge is basically displayed by humans. As soon as the knowledge is formalised in a case based reasoning system, it moved to the representation layer. Once, it is coded using data structures, it reaches the implementation layer. These steps go through knowledge acquisition, design and development phases. For example, if someone wants to buy a car, a cognitive layer is required such low mileage and price descriptions.

In this research we focus on the constraint type case of representation which includes variables in both problem and solution descriptions. Typically, constraint cases are designed for problems under the condition where the objects are formulated and the objects design are found considering the solution descriptions. The solution basically indicates the same object as a query Q. The vocabulary also controls to some extent the usage of representing the data structure of the concepts and notions in two aspects: semantically i.e. the meaning and syntactically for example, the spelling.

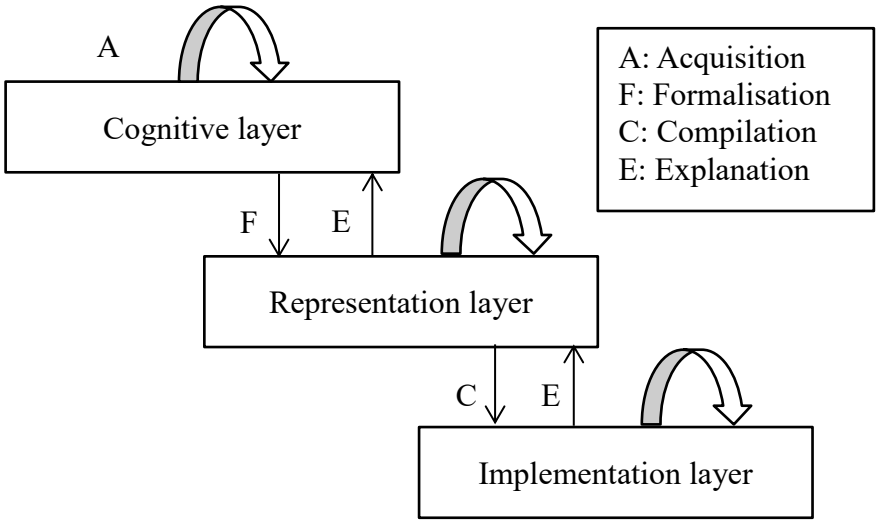


Figure 3 Representation Layers [42]

2.1.4. **Case Bases**

The case base in a CBR system is a memory. It includes a collection of cases that are utilised to perform a reasoning task in the context of the methodology. It represents the data source that is typically finite. In fact, what makes CBR specific is the way in which a case base is used. Case base is a common special term used in CBR requirements. It could also refer to the word "memory" in cognitive science too.

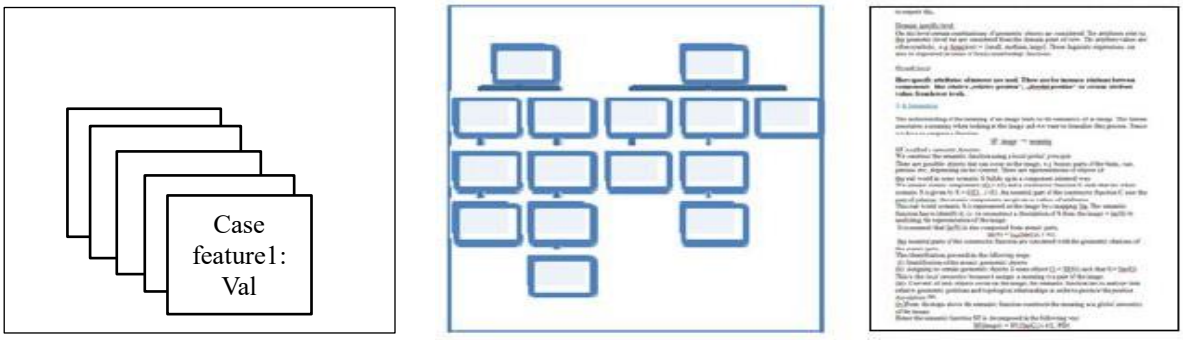


Figure 4 Three types of case organisation: flat, structured, unstructured text [42]

As shown in Figure 4 there are three types of case organisation namely: flat, structured and unstructured for instance images and text. It is noted that the three illustrated types suggest

different paradigms but we will focus on the flat type in this research.

2.1.5. CBR Methods

CBR methods can be divided into 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 is illustrated in Figure 1.

The major tasks of the CBR cycle are summarised as follows:

- Retrieve: is finding the most similar cases from the case base by comparing the value of the attributes of the new case to those of the stored ones. This research focuses on this phase. Wrongly retrieved cases can lead to taking the wrong decision. It will be discussed in more detail in section 2.1.6.
- Reuse: is the process when a case is chosen for its solution to be utilized later. The reuse phase is completed when the new solution is suggested for the next task which is the revision of the case. Basically, the reuse proposes a solution to solve a problem by re-using the knowledge of the retrieved cases. Ultimately, if the new problem exactly matches the retrieved cases the usage of reuse is quite simple; otherwise an adaption is required when they differ.
- Revise: This process begins when a solution is proposed to solve a new problem and ends when it is completed. It aims to assess the applicability of the proposed solution. This type of assessment converts to evaluation if it is tested in the real world. It can also be performed using simulation as it is cheaper but may ignore some important aspects. The evaluation can be achieved in the real world or in a simulation; the latter is cheaper but may ignore an important aspect. This is considered as an old dilemma in artificial

intelligence named the frame problem. It is asserted that all possible facts can never be entirely complete in the real world.

- Retain: This phase starts when revising updates a new case in the case base, so that the new learned case can be used to solve future problems. However, the solution of some systems may not be retained. Systems may learn a new solution through the adapted use, while others receive only actual cases.

2.1.6. **Retrieval**

Case retrieval is the approach of discovering those cases that are the nearest to the current case within a case base. Retrieval plays a major role in the CBR cycle but it is not a standalone procedure as it is basically connected to the similarity measures and case representation - for example, attribute-based, database, Textual and image's representations. The all mentioned forms have various retrieval methods due to different representation. Furthermore, the most common method of the data retrieval process is the one while operates on a fixed database. In the retrieval phase, Searching for an object in the case base is actually presenting a query object to find the nearest neighbour in the group of answers objects, that could lead to the fact that the intended target may not be described precisely [44]. The problem with this is that the user either obtains many answers (noise) or no answer. However, the silence in CBR systems does not exist because they always present answers and the noise can be avoided by controlling the number of nearest neighbour cases.

Retrieving a case means starting with a (partial) new case, and finishing when the best matching cases are retrieved. The sub-functions of the retrieval are designed to: identify features, search for similar case, matching a case, choosing a similar case - usually performed in that order. Basically, a set of relevant problem descriptors are used for the identification task. The target of the matching procedure is to retrieve a group of cases that are adequately similar to the new

case. The selection procedure operates on the same cases and picks the best or the closer match.

Some CBR methods retrieve a former case mainly based on the similarities among problem descriptors [45], [46]. Some approaches focus on deeper features retrieval, [47], [48] and [49], while more recent methods attempt to utilise other knowledge to enhance SBR [50], [51]. This project aims to develop a new technique to improve the retrieval strategy and explores various methods by integrating other knowledge types into CBR. The cases build on similarities and relative significance of features as a large part of the domain knowledge is required to explain the nature of why two cases are matched and how reliable the match is. In addition, the method of matching a case is described as hard or unachievable to obtain because of the poor representation the knowledge. By contrast, combined methods as knowledge intensive are capable of using the meaning of the problem, therefore the description and its meaning make the similarity of matching cases obtainable [1]. In addition, the combined strategies may consist of general knowledge, implicit in their matching strategies. The difference between poor and intensive knowledge is consequently connected to domain knowledge representation. Moreover, it indicates generalised domain knowledge, because cases also consist of explicit knowledge but can be named as specific domain knowledge [52]. The retrieval phase usually involves the following subtasks:

- Identify features – this may simply be to notice the feature values for a case, or can be a more complex evaluation which tries to understand the problem in a context by generating an expectation of other feature values or by asking the user. General features can be used to infer other descriptors that were given as an input, or similar problem features can be retrieved from the case base, using features of those cases as anticipated features. Examining the expectations is possibly achieved within the general knowledge and cases.

- Initially match – usually performed in two steps firstly, an initial matching process that gives a list of potential candidates, which are then further, examined to select the best among these. There are three ways of retrieving a certain case: searching an index; by searching in a model of domain knowledge and the following a direct index pointer from the problem features. [53] uses the first method for its diagnostic reasoning, and the second is used for the test selection. A global similarity metric is applied to evaluate similarity based on surface match on a domain dependent [54]. The second approach uses dynamic memory systems of a general domain which could be employed in combination with a search method. Cases could be retrieved from features deduced from the input, or from input features. The case that matches a specified part of the problem feature (deduced or input) could be retrieved – a case that matches all input feature is no doubt, a significant candidate for matching, nevertheless it depends on the strategy. Global similarity metric is used by [55], with different parameters to analyse a domain. A number of tests for retrieved cases are often carried out. Especially if cases are retrieved on the principal of subset features. A method to evaluate the quality of similarity is required and a similarity metric has been suggested which builds case features and surface similarity.

  Similarity evaluation could be more knowledge based. For instance, by attempting to comprehend the problem more efficiently, and using the targets, from this complicated process to guide the matching [1]. An additional option is to scale the problem descriptors as stated by their significance for distinguishing the problem, throughout the learning phase. In [56], for instance, every feature of the stored cases has a degree of significance assigned to it for the solution of the case. The same technique was adopted by [57], which stores the features that effect the lack number of cases that have no

solution. In addition, discriminatory value of features is stored with reference to the group of cases as a predictive strength.

Matching cases can be found by comparing results with input features. The features can be compared using a similarity measure which is basically normalised, for example to the range [0, 1]. Thus it is easy to compare cases based on several features. The case based reasoning tries to identify the problem, and employ this understanding when comparing cases to a query. Furthermore, it can weigh the input features. A simple relevance test for instance may be to examine if the retrieved solution conforms to the anticipated solution of the new problem.

- Features – from the group of similar cases, the system selects a best match from the cases returned by the initially match. The best matching of cases are basically specified by evaluating the rank of the close cases. This is achieved in an effort to produce a clarification to justify non-resembled features. If the match is inadequate, another attempt to find a better identical selection is performed by using links to other related cases. The selection step can generate results and predictions from each retrieved case, by asking the user and by using an internal model [1].

To conduct a successful case retrieval, there ought to be selection criterion that decides how a case is examined to be significant for retrieval and technique to regulate how the case base is searched. The selection standards are important to decide how close the current case is to the cases stored. The case selection criterion relies relatively on what the case retriever is searching for in the case base. The case retriever is frequently searching for a complete case, the features of which are contrasted to current cases. Nevertheless, there are instances when only a part of a case is being searched. This occurrence may appear for the reason that no full case are found and a solution is being combined by choosing portions of several cases. Similarly, a retrieved case is being amended by utilising another portion of cases in the case base.

The genuine processes in retrieving a case is highly reliant on the memory structures and indexing approaches used. Some retrieval methods utilised by researchers are entirely different, ranging from a simple nearest-neighbour search to the use of intelligent agents. We discuss both the most commonly used and traditional methods in the following sections.

Most of this research concentrates on SBR problems, which focus on Retrieval only regardless of whether the solution is adequate or not. Recently, SBR learning has been an area of extensive research interest and many researchers focus on retrieval problems, where they attempt to predict a correct solution for a target problem. In addition, several researchers have typically implemented SBR through different methods (e.g. K-nearest neighbour retrieval or simply KNN) in [2]. The notion of KNN is that retrieval is performed through retrieving the K most similar cases to the object problem. However, a well-known limitation of KNN remains in allowing irrelevant attributes to impact the similarity computation. Inappropriate decisions are not necessarily monetary but it might be also a waste of time and effort.

The next section will present the literature review of the integration of DM and CBR.

2.2. Data Mining Common Methods

Data mining (DM) is the science of extracting hidden knowledge from databases. It is an influential branch of computer science with great potential to assist researchers focus on the most important information in their data. DM mechanism anticipates future trends and behaviours, allowing users to produce knowledge-driven decisions. DM offers such valuable automated analysis of past events and provides a powerful mechanism of decision support systems. In addition, DM approaches can answer various questions that otherwise would need experts' knowledge or are time consuming to resolve. They refine datasets for hidden patterns, detecting predictive knowledge that users may miss because it could be outside their expectations.

Data mining approaches can be applied rapidly on existing software platforms to enhance the performance of existing resources [58], and can be integrated with systems of different types such as CBR retrieval [24]. When integrated into CBR for instance, the Association Rules (ARs) approach can find the related features which could form ultimately a correct pattern of Rules. This pattern is encoded through certain knowledge in conjunction with similarity knowledge to select a correct answer of the target problem. DM techniques are broadly divided into three categories:

- Classification: is the process of categorising and sorting data into different types, forms or any other specific class. In addition, it enables the segregation of data according to the dataset requirements for different personal or business objectives. For instance, classifying an email into spam or legitimate [59].

- Association Rules: seeks for the correlation between items or variables i.e. the data of a customer purchasing habits might be gathered by a supermarket. Utilising association rules, the market can specify which products are frequently sold together and employ this information for marketing purposes [60].

- Clustering: is the process of partitioning groups or objects into meaningful sub-classes, named clusters. This process helps users to understand the structure of the dataset. Clustering is unsupervised learning because the classes are not predefined [61].

2.2.1. **Classification in Data Mining**

Classification is a procedure that allocates items in a group to target classes or categories. The target of classification is to precisely predict a class for each object in the dataset. For instance, classification model could be used to identify patient as yes or no with regards to inflammation risks.

A classification procedure starts with a dataset in which classes are known. A classification model for example that predicts acute inflammation bladder could be developed based on observed patient over a period of time. Temperature of patient, Urine pushing, occurrence of nausea, lumbar pain, micturition pains and burning of urethra are attributes that constitute a case. Moreover, yes and no labels are the simplest type of the binary classification problem, where the target is to classify two possible values of labels.

There are several classification techniques for predicting an outcome from a dataset such as Naive Bayesian [62] [63], decision trees [64] and K-Nearest Neighbours (KNN) [65]. KNN is used with various distance measures such as Euclidean and Minkowski [66] and is used in CBR as similarity measures. The following, will provide a brief description of the KNN classification algorithm as it is used in this research for classification in CBR.

KNN Algorithm is a common method to classify objects based on the closest training examples in the feature space. It is an approach based on instance learning [67], which is categorised as a lazy learning. Its function is only approximated locally and all computation is postponed until the classification is performed. Amongst the simplest classification algorithms of all machine learning algorithms, the KNN's objects are classified by a majority vote considering its neighbours. The object is assigned to the class most common throughout its k nearest neighbours, if k is a positive integer, basically small). If k equals 1, then the object is merely assigned to the class of its nearest neighbour.

A number of researchers have developed KNN algorithm through years. IBL algorithm for instance was introduced by [67] and is the early developed approach based nearest neighbour algorithm. It assumes that similar instances can have similar classifications. This causes to their local bias for classifying novel instances as per to their greatest similar neighbour's classification.

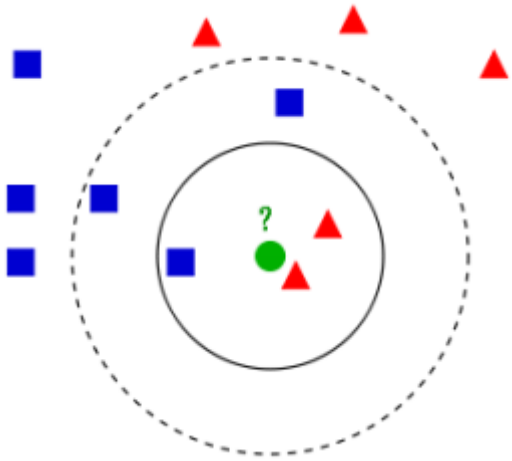


Figure 5 KNN Classification [67]

The example given below is a general form of the KNN algorithm, it measures the difference between *x* and *y* considering *k* number of the nearest classes.

The KNN equation, is given below.

$$Similarity(x, y) = \sqrt{\sum_{i=1}^{k} f(x_i, y_i)}$$

Equation 2 KNN Metric

For numeric values attribute it is: $f(x_i, y_i) = (x_i - y_i)^2$

For Boolean and symbolic values attributes: $f(x_i, y_i) \neq (x_i \neq y_i)$

Some recent work on KNN can be found in [68], [69] and [70].

In CBR, KNN methods are used as similarity measures using Euclidian and Minkowski distance measures. They are applicable to attributes with numerical values and thoroughly linked to numerical distances. If symbolic values exist they need first to be numerically coded [42].

In mathematics, many distance functions have been used for several purposes and metrics are

explained in great detail. An elementary example will be presented using similarities and distances alternatively, an example is given below:

$$dc(x,y) = \sum_{i=1}^{k} (|xi - yi|)$$

Equation 3 Distance Similarity Metric

The name of the metric is derived from driving one street of a city, dc refers to city block. It seems realistic but it should be noticed that it abstracts quite a bit from reality: There may be hilly and one-way, which are essential for the speed of cars and pedestrians. Similarly, weighted Euclidean measures can be defined more realistically. They are shown in the form of a distances [42]:

$$d(x,y) = \sqrt{\sum_{i=1}^{k} w.(x_i - y_i)^2}$$

Equation 4 Euclidean measure Metric

More general is the Minkowski distance where:

$$\boldsymbol{d_{mink}(x,y) = p\sqrt{\sum_{i=1}^{k} w.(x_i - y_i)^p}}$$

Equation 5 Minkowski Metric

In this research, we opt for the use of the Euclidian distance as motivated by [71] and [42]. Minkowski is the most common mathematical distance on which special relativity is formulated, while Euclidian space and time will often differ due to length contraction and time dilation. According to [42], the Euclidian distance, has slightly produced more accurate results. However, diversity metric is used in section 1.1. , Euclidian and Minkowski are the main types of counting similarities.

2.3. Association Rules Algorithms and Association Knowledge

This section elucidates the Association Rules algorithms i.e. Apriori, Predictive Apriori and more general issues associated with CAR that have been used in this research. The attempt was to use these algorithms in order to produce the FP-CAR algorithm. The first experiments we used predictive Apriori to produce the FP- tree. The second experiments we utilized CAR in order to produce an optimum tree which has fulfilled the objectives of this research.

2.3.1. Association Rules

One of the most popular DM approaches is to find frequent item sets from a transaction dataset and derive ARs [72]. The major concept of ARs is to discover the correlation of attributes within data. 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 as well. The process of ARM is formally stated by [73] as follows.

- Let $I = \{i1, i2, \ldots, im\}$ be a set of items.
- Let D is a set of transactions forming n transactions $T = \{T1, T2, T3, T4, \ldots, Tn\}$, where each transaction T is a set of items such that $T \subseteq I$.

The AR of $A \Rightarrow B$, where $\{A, B\}$ are subsets of I and $A \cap B = \emptyset$. The transaction should be read at any time T include A will also possibly include B. The set A is indicated as the antecedent and B as the consequent of association rules. In addition, implication is one direction reserved $A \Rightarrow B$ and does not necessarily equal the implication of $B \Rightarrow A$. The support refers to frequency *(supp)*, the confidence indicates to the accuracy *(conf)*. *supp* in ARM is recognised as the percentage of each record which holds $A \cup B$ that is concerning the total number of records. ARs is considered to be supported or frequent if its support exceeds a user

minimum threshold. *Conf* is identified as the ratio of $A \cup B$ to the support of A.

$$\boldsymbol{Conf(A \Rightarrow B) = \frac{Supp(A \cup B)}{Supp(A)}}$$

Equation 6 Confidence Metric

If the result of confidence equals 1, we would have a very good AR. ARs value are deemed to be valid if the confidence exceeds the user desired confidence value. ARs are thus basically produced by first determining frequent itemsets and then using the standard of support and confidence to find important relationships. Even though the above discussion focuses on the support and confidence of the ARM framework it should be mentioned that this has its critics and that different ARM frameworks have been proposed [74], [75]. The support confidence of ARs framework nevertheless remains the most popular.

ARM given above can be basically described as two procedures, ARM and FPM generation. ARM and FPM are deemed to be computationally expensive because the process generates a large number of possible frequent items. Therefore, much research is done on ARM and FPM, as well as a lot of methods have been derived such as Apriori and FP-Growth which will be discussed later in this chapter.

2.3.2. **Apriori Algorithm**

The Apriori algorithm is the basic algorithm for ARM as suggested by [15]. It functions in an iterative process as a level-wise search [76]. The first pass, support of individual items is calculated and frequent items are determined [77]. In each subsequent pass, a seed set of itemsets found to be frequent in the previous pass is utilised for generating new probable frequent itemsets, called candidate itemsets, and their actual support is counted during the pass over the data. At the final part of the pass, those satisfying minimum support constraints are collected, that is, frequent itemsets are determined, and they become the seed for the next pass. This process

is repeated until no new frequent itemsets are found [78]. The final result is a frequent superset using the threshold of support and confidence specified by the end user [79].

In a CBR context, ARM can be used to find interesting relationships from a given case base. A transaction and an item can be considered as a case and an attribute value pair, respectively. Apriori [80] is an algorithm used to evaluate the quality and rank a large number of ARs extracted for useful Interestingness measures. As candidates of the measures, the support and confidence standard are frequently used. In other words, it can be used for ranking patterns according to their potential interest to the user. In general, the problem of ARM is to produce all ARs that have support and confidence not less than a user-specified minimum support (minsupp) and a user-specified minimum confidence (minconf), sequentially.

2.3.3. **Predictive Apriori**

Another improved type of the Apriori algorithm is the Predictive Apriori algorithm [81], which resolves automatically the problem of balance between two parameters, increasing the probability of producing an accurate prediction for the dataset. In order to accomplish this, a parameter named the exact expected predictive accuracy is explained and calculated using the Bayesian concept, which provides information about the accuracy of the rule found [82]. In this algorithm, confidence & support are combined into one measure named "Accuracy". (Confidence, Support) => Accuracy. This, predictive accuracy is utilised to create the association rules. In WEKA software [83], this algorithm generates *n* best association rules where *n* is number of rules determined by the user. A rule is added if the expected prediction of the rule is among the n best and subsumed by those rules with at least the same expected prediction of accuracy [84]. There is also a confidence based on association rules in ARs ranked are sorted according to predictive accuracy. Therefore, the attempt is to increase the prediction of the

accuracy of the rules rather than confidence in Aporiori [85]. In this PhD research, this algorithm is used in the first experiments in order to produce a frequent tree such rules may reflect a similar pattern of the CBR target problem.

2.3.4. **Class ARM**

Class association rule mining (CAR) is one of the ARs algorithms, which integrates association rule mining (finding all rules existing in the dataset that satisfy some constraints) and classifying rule i.e. discovering a small set of rules in the database that forms an accurate classifier by focusing on mining a special subset of association rules, called class association rules (CARs) [5]. It can be applied not only to linearly separable cases, but also to linearly inseparable cases, or where other linear classification approaches are not applicable [86]. One of the CAR's advantages algorithms over conventional methods, for example support vector machine, is its interpretability. This is because classifiers are generated as a set of simple rules without much sacrifice of accuracy [87]. In addition, when applied to a medical dataset for instance, gene data, the CARs algorithm , which predicts a class label based on specific sets of differentially genes that are actually noticed in training samples, are expected to generate more biologically reasonable classifiers, because it is generally not individual genes but sets of those genes that collectively define phenotypes such as drug responses[88].

It is noticeable that CARs are a special subset of ARs whose consequents are restricted to a single target variable. In a CBR context, a CAR is presented as an AR in which a consequent holds the item built as a pair of a solution attribute and its value. This might be called a solution item. A CAR therefore has the form $X \Rightarrow y$, where $X \subseteq I$ is an itemset and y $\in I$ is a solution item. It is noticeable that to represent AK, CAR representation can be adopted. AK can be encoded to reflect how certain problem features are interestingly associated with specific solutions in a given case base. Considering this, it should be noted that the form of a CAR $X \Rightarrow y$

allows the representation of 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. Also, CAR is considered as an extension of the Apriori algorithm. In other words, the goal of this algorithm is to find all rules of the items built from the form < cond_set, y > where cond_set is a set of items, and $y \in Y$ where Y is the set of class labels. The support count for example of the rule item is the number of instances in the dataset D that include the condset and are labelled with y. Each rule item corresponds to a rule of the form: condset $\Rightarrow y$. The Rule item that has support greater than or equal to minimum support is called a frequent rule item, whereas the others are called infrequent rule items. The rule item with the highest confidence is chosen as the representative of those rule items, for all those that have the same cond_set. The confidence of rule items is calculated to decide if the rule item meets minimum confidence. The set of rules that is determined after calculating the support and confidence is called the (CARs) classification association rules.

2.4. Frequent Pattern Mining

FPM plays a major role in association rules mining. With reference to the CBRAR strategy, a great part of the new algorithm FP-CAR depends on both CAR and FP-Growth algorithms and P-tree is utilised when it is necessary. FP-Growth is well known algorithm that was developed on a set enumeration tree structure. FP-tree is a part of FP-Growth; it is adopted o mine CARs where it holds potential patterns that can match a CBR target problem. FP-Growth is discussed further in sub section 2.4.1. This resulted FP-tree pattern is to be compared with a target problem of CBR in order to disambiguate unrelated cases.

2.4.1. Frequent Pattern Growth Algorithm

As noted previously frequent pattern mining 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. With respect to tabular data the majority of these have been integrated with ARM algorithms. Of these the best known, and most frequently cited, is the Apriori algorithm [89]. Another established FPM algorithm is FP-growth which 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 dataset structure using FP-tree and extracts the frequent pattern directly from this structure [18].

An 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. If the size of the FP-tree is adequate to fit into the main memory, the extraction of frequent itemsets will be possible directly from the structure in memory instead of making repeated passes over the stored data. Figure 6 [90], displays a dataset that contains five items and ten transactions. The structure of the FP-tree is depicted in the diagram after reading the first three transactions. Every node in the tree contains a label of an item accompanied by a counter that displays the number of transactions mapped into a specific path. Basically, the FP-tree includes only the root represented by the null node. The FP-tree is consequently extended in the following ways [91]:

- The dataset is scanned once to define the support count of each item. Infrequent items are ignored. While the frequent items are classified in decreasing support counts. For the dataset presented in Figure 6, *a* is the most frequent item, followed by *b, c, d,* and *e*
- The algorithm starts a second pass over the data to form the FP-tree. After the first transaction is read, *(a, b),* the nodes labelled as *a* and *b* are created. A path is then

constructed from $null \rightarrow a \rightarrow b$ to encode the transaction. Every node along the path has a frequency count of 1.

- After reading the second transaction, $\{b, c, d\}$, a new set of nodes is created for items b, c, and d. A path is then built to represent the transaction by joining the nodes $null \rightarrow b \rightarrow c \rightarrow d$. Every node along this path also has a frequency count equal to one. Although the first two transactions contain an item in common, which is *b*, their paths are disjointed because the transactions do not share a same prefix.
- The third transaction $\{a, c, d, e\}$, includes a common prefix item (which is *a*) with first transaction. The path for the third transaction, is consequently $null \rightarrow a \rightarrow c \rightarrow d \rightarrow e$, and overlaps with the path for the first transaction, $null \rightarrow a \rightarrow b$. Because of their overlapping path, the frequency count for node *a* is increased to two, while the frequency counts for the new nodes, *c*, *d*, and *e*, are equal to one.
- This procedure continues until every transaction has been mapped onto one of the paths stated in the FP-tree. The outcome FP-tree after reading all the transactions is sketched at the bottom of Figure 6.

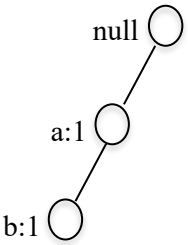
The size of the FP-tree is usually smaller than the size of the uncompressed data because some transactions data often share a few items in common. In the best case scenario, where the transactions contain the same set of items, the FP-tree has only one branch of nodes. The worst case occurs when each transaction has a unique set of items. As none of the transactions include any items in common, the size of the FP-tree is completely the same as the size of the original data. However, the storage requirement of the FP-tree is a little higher because it requires extra space to store pointers between counters and nodes for each item.

Also, the size of the FP-tree depends on how the items are sorted. If the ordering scheme in the preceding instance is reversed, i.e., from lowest to highest support item, the shape of FP-tree

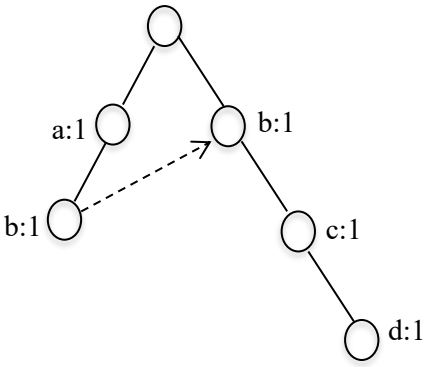
is depicted the Figure 7. In addition, the FP-tree contains a list of pointers connecting between nodes that require the same items. These pointers are represented as dashed lines in Figure 6 and Figure 7 to assist and simplify the rapid access of individual items in the tree.

Transaction Dataset

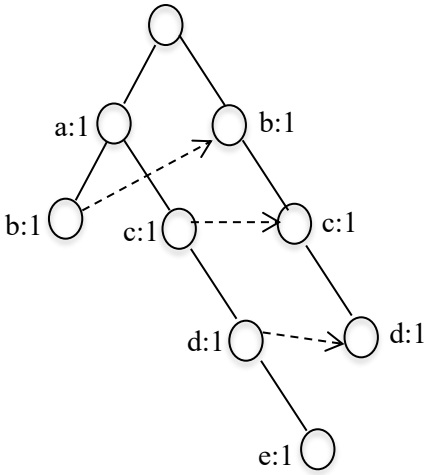
TID	Items
1	{a, b}
2	{b, c, d}
3	{a, c, d, e}
4	{a, d, e}
5	{a, b, c}
6	{a, b, c, d}
7	{a}
8	{a, b, c}
9	{a, b, d}
10	{b, c, e}



(i) After reading TID=1



(ii) After reading TID=2



(iii) After reading TID=3

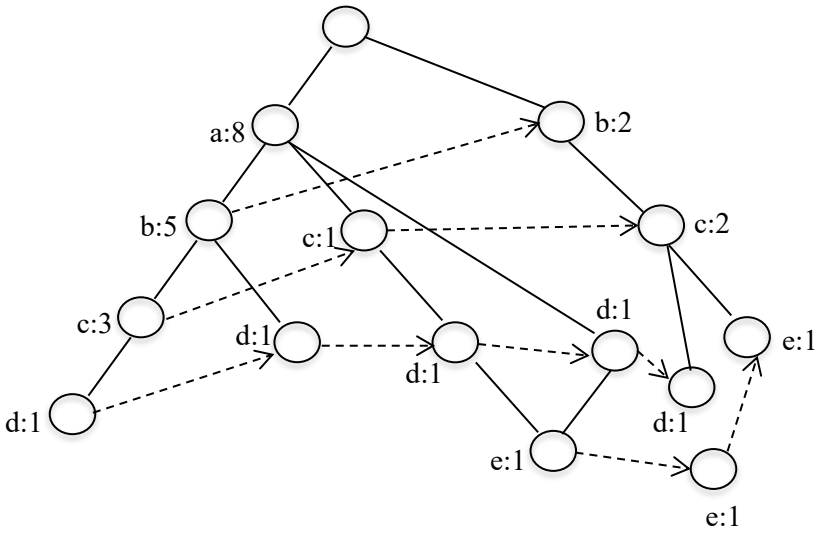


Figure 6 Construction of FP-tree [90]

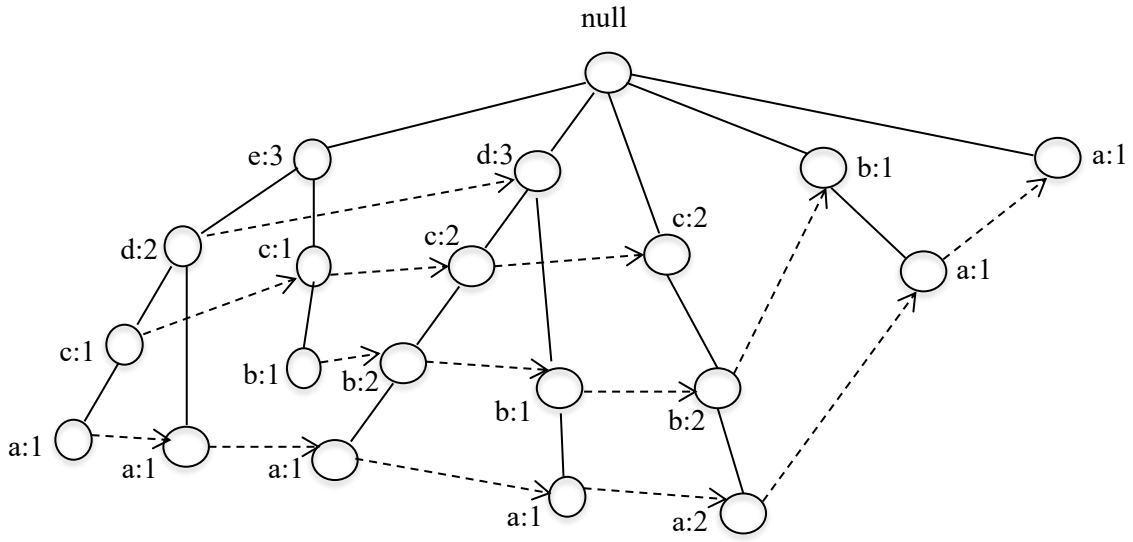


Figure 7 FP-tree representation with different item orders associated with Figure 6 [91]

2.4.2. Tree structures for Mining Association rules

The general problem in deriving association rules is the space complexity and exponential time of the task of computing support counts of all 2n subsets of the attribute set I. Therefore, practicable algorithms attempt to decrease the search space by computing support-counts only for those subsets which are recognised as possibly interesting. The best-known algorithm, "Apriori"[92] and [93], does this by repeated passes of the database, continually computing support-counts for single attributes, pairs and triples. In addition, any set of attributes can be "interesting" only if all its subsets also reach the required support threshold. The candidate set of attributes is pruned on each pass to remove those that do not satisfy this condition. Other algorithms, AIS [94] and SETM [95], have the same general form but differ in the way the candidate sets are derived.

Two aspects of these algorithms in terms of performance are of concern. These are the number

of passes of the database that are demanded, which will generally be one greater than the number of attributes in the largest set, and the size of the candidate sets which may be produced, in particular during the preliminary cycles of the algorithm. The number of passes may be decreased to 2 by techniques which begin by examining subsets of the database[96], or by sampling the database to "guess" the likely candidate set [97]. The disadvantage of these methods is that the candidate set gained is necessarily a superset of the set of interesting sets. Therefore, the search space may become moderately large, particularly with packed database records. Large sizes of candidate-sets create a problem both in the calculation required, as each database record is examined, and in their storage requirement. The process described of the Apriori algorithm saves the candidate set in a hash-tree, which is sought for each record in turn to detect candidates that are subsets of the set of attributes contained in the record being considered.

Dealing with large datasets has led researchers to look for new approaches which seek to identifying *maximal* interesting sets without examining all their subsets.[16] achieved this by dividing the search space into clusters that are associated with attributes. However, these approaches break down if the database is too dense- for many clusters to be apparent [98]. The Max-Miner algorithm also searches for maximal sets, using Rymon's [99] set enumeration framework to organize the search space as a tree. Max-Miner decreases the search domain by pruning the tree to remove both subsets of frequent sets and supersets of infrequent sets. In a development from Max-Miner, the Dense-Miner algorithm [100] implements additional constraints on the rules being required to decrease further the search domain in these cases. Basically, these algorithms perform better with dense datasets than the other algorithms described, but also need multiple database passes. Such databases which can be totally contained in memory also make use of a set enumeration structure. In this case the tree is utilised to store frequent sets that are produced in depth first order via recursive prediction of the database. Nevertheless, because of the combinatorial explosion in the number of candidates which might

be in consideration. Also, because of the cost of repeated access to the database, no existing algorithm deals successfully with large databases of densely-packed records.

2.4.3. **Partial Support Trees P-trees**

The most computationally expensive part of association rules and related algorithms for example (Apriori and FP-Growth) is identifying the subsets of a record that are members of the candidate set being considered, particularly for records that include a large number of attributes [34]. This can be avoided by first counting only sets occurring in the database, without considering subsets [101].

Let *i* be a subset of the set *I* (i.e. *I,* is the set of *n* attributes in the database). *Pi* is defined as the partial support for the set *i*, to be the number of records in which the contents are identical with the set *i*. Also, *Ti*, is the total support for the set *i*. This can be shown as follows:

$$T_i = \sum P_i$$

Equation 7 [34]

For a database of *m* records, the partial supports can, be counted easily in a single database pass, to produce *m*' partial totals, for some $m' \leq m$. Rymon's set enumeration framework [99] can be used to store all counts in a tree; Figure 8 shows this for *I = {A, B, C, D}*. To avoid the possible exponential scale of this, the tree is constructed concurrently as the database is scanned in order to include only those nodes that exemplify sets actually present as records in the database, as well as some additional nodes created to preserve the tree structure when necessary. The cost of construction this tree and its size are linearly related to *m* instead of *2n*.

Advantage can be taken of the structural relationships between sets of attributes from the tree when the construction phase is used to begin the computation of sum supports. While each set is located within the tree during the process of the database pass, it is computationally low-cost

to add to interim support-counts, *Qi* is stored for subsets which precede it in the tree ordering.

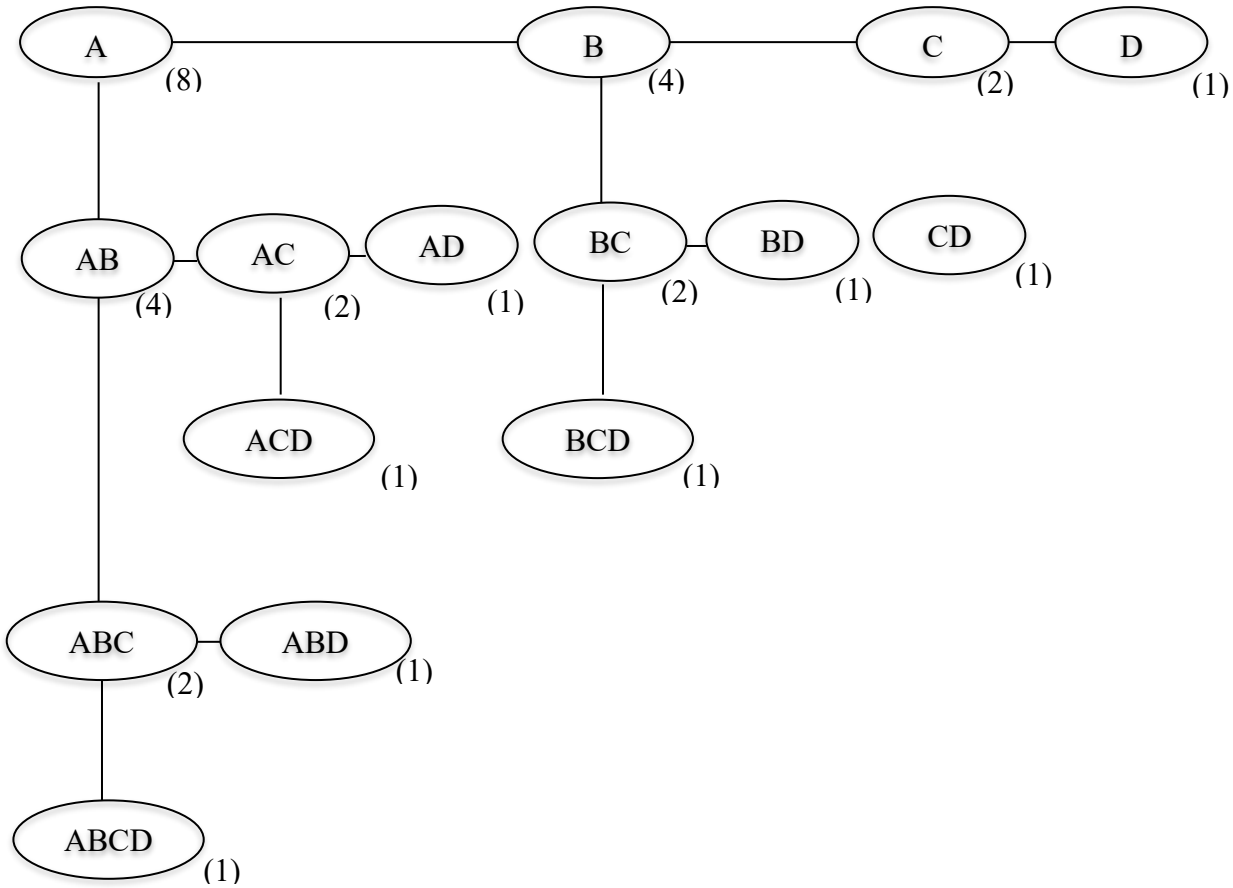


Figure 8 Tree storage of subset of {A, B, C, D} [34]

Therefore, in Figure 8 the number associated with the nodes are the provisional support counts. These can be stored in the tree constructed from the dataset and the records which compose exactly one instance of each possible set. Hence, for instance, *Q (BC)* = 2, is derived from 1 instance of *BC* and 1 of *BCD* as follows:

$$T(BC) = Q(BC) + P(ABC) + P(ABCD) = Q(BC) + Q(ABC)$$

The method described above is named P-tree (partial support tree) and was developed by [34] to indicate this incomplete set- enumeration tree of interim support counts. This algorithm for constructing the P-tree is able to count the interim totals because it contains all the relevant data stored in the original database. Research has shown that this concept can be applied and

utilised to almost any created algorithm to complete the summation of total supports [102] [103]. The use of the P-tree as an alternative for the original database basically offers two possible advantages. First, when n is small ($2n < m$), then traversing the tree to examine every node will be notably quicker than scanning the whole dataset. Secondly, even for great amount of n, if the database includes a high degree of duplication ($m' < m$), utilising the tree will be a significantly faster process compared to a full pass of database, particularly if the duplicated records are densely-populated with attributes. Ultimately, the computation required in each cycle of the algorithm is significantly decreased because of the partial summation already conducted in constructing the tree. For instance, (considering pairs of attributes) in the second pass of Apriori, a record including ***r*** attributes might require the counts for each of its ***r (r - 1) / 2*** subset-pairs to be increased. It is important to consider only those subsets not already covered by a parent node, when examining a node of the P-tree, contrarily, that would be only r – 1subsets, in the best case scenario. To exemplify this, in Figure 8 consider the node ABCD in the tree. The partial total for ABCD has been previously included in the interim total for ABC. In addition, this will be added to the final totals for the subsets of ABC when the second node is examined. This means, in terms of examining the node ABCD, the need is only to consider those subsets not covered by its higher level (parent), namely those including the attribute D. The result obtained from this will be larger in addition to the greater the number of attributes in the set which is being considered. The structure of P-trees is similar to the FP-tree mentioned previously but it has a different form and similar properties. It is noticeable that the FP-tree is built in two database passes. Firstly, to eliminate attributes that fail to reach the support threshold, and then to order the others by frequency of occurrence. The FP-tree also stores each node in a single attribute Therefore each path in the tree represents and counts one or more records in the database. Moreover, it includes more structural information, allowing all the nodes to represent any attribute being related into a list. This structure enables the execution of an FP-

growth algorithm which can generate successively subtrees from the FP-tree similarly to each frequent attribute, to indicate to all sets in which the attribute is associated with its predecessors in the ordering of a tree. The combination of two structures, the FP-tree and P-tree, which have been developed separately, are utilised in the new algorithm, which is discussed in Chapter 3.

2.5. Related Work of CBR and other Types of Knowledge

In CBR applications, similarity based reasoning strategy has been extensively used for example in the detection of retinal abnormalities [104], image diagnosis and therapy [105], model design in automotive interior industry [106] and prediction of soil organic matter concentration [107]. SBR has been usually applied using KNN retrieval method [2]. This method is performed through retrieving the most similar case to the objective problem but a recognised limitation of KNN remains in consenting unrelated attributes to affect the similarity computation. The following section will illustrate some related work of integrating SBR with other knowledge.

2.5.1. Data Mining and CBR

The amount data is increasing over the time and the need to transform this data into useful information is largely demandable. Knowledge discovery in database (KDD) is an important field of the computer science area [108], [109], which uses methods for extracting understandable information from the quickly increasing volumes of data. KDD process involves many phases i.e. data pre-processing, data integration, data transformation, data mining, pattern recognition and knowledge representation.

Data mining is crucial stage in KDD that implements algorithms to find interesting hidden data patterns in the dataset, whereby this data could be saved in databases i.e. information repositories or data warehouses. It merges techniques from machine learning, Artificial intelligence, and statistics to analyse and conclude data into a structured model. The knowledge explored

by data mining strategies can be employed in different applications for instance health care, information technology and market analysis.

Over time, techniques of integrating DM and KNN have often been implemented in CBR research to enhance KNN method through three main platforms. Firstly, is to integrate feature weighting (FW) and feature selection (FS) into KNN. In this framework, FW is achieved by estimating the optimal weights of the original features of cases [110], [111], and FS is employed in choosing relevant features of cases [37], [40], or their aggregation is used to leverage their usefulness[36]. Secondly, is to merge data clustering with KNN, where the structure of clustered cases is leveraged to lead to more relevant cases [112], [113]. Given a case base, a set of clusters is built, where each cluster describes 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 [114]. Thirdly, is to apply both DM and SBR techniques together to discover cases related to the target problem. For instance, [115] displays how to integrate DM with SBR to improve liver diagnosis. Given a target problem, once a DM technique (a back-propagation neural network) is implemented on the case base, some cases thought to be essential 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 ultimately utilized as a retrieval result for the problem. Association knowledge recognises interesting association between solutions and case features in the case base, whereas feature's weighting and selection emphasises on identifying key case features forming the case base.

Unlike this scheme, our approach is based mainly on the use of AK built via CARs.

2.5.2. **SBR and Statistical Learning**

SBR has also been integrated with statistical learning. For instance, it is suggested that KNN

can be improved by dynamically determining an optimal number of the nearest cases for a target problem using the division of distances between possible similar cases to the problem [116]. In addition, a genetic algorithm is applied to optimize the number of the nearest neighbours for the objective problem [117]. These methods are based on the consideration that KNN usually utilises a fixed number of neighbours, which may minimize the ability to predict a desired set of similar neighbours. Nevertheless, a problem with these methods is that the optimal set of the nearest cases are attainable only by using Similarity Knowledge. The candidates of relevant cases for the target problem are established, when using a similarity measures, and then these are additionally examined through statistical approaches using their similarity scores.

The above approaches are unlike the CBRAR retrieval approach which leverages two different forms of knowledge [i.e., AK (statistical information drawn from CARs) and SK] to improve the use of SK for retrieval.

2.5.3. **Machine Learning and Retrieval**

Machine learning ML [118], [119] is involved with computer programs which are able to optimise the performance using training examples and pervious experience. It utilises statistical and computational approaches to build mathematical model that discover and develop patterns in given examples.

Generally speaking there are various types of machine learning approaches i.e. deductive, inductive and analogy learning [120]. Deductive learning approaches are based on converting the general principals into logically particular examples by analysing the available knowledge to discover the most beneficial information [121]. On the other hand, the inductive methods are constructed on transforming the specific examples in some principal into general principal's

description. This basically is achieved by using statistical and computational methods to extract patterns and rules from database. Analogy or inference learning is the process of displaying the similarity between entities by transferring the information from the source to the target. Inductive and analogy learning methods are often used in the field of machine learning [122].

Basically, machine learning strategies involve three types of learning: supervised, unsupervised and semi-supervised' learning. Supervised learning is used to learn a classifier from training examples annotated by inferring a function from labelled dataset, whilst unsupervised learning method learns from unlabelled training examples as clusters. Semi-supervised learning typically uses small amount of labelled data and large amount of unlabelled data training to learn a classifier.

The development of machine learning has resulted in retrieval approaches that SBR merge 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 [123]. [50] 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 [22] dynamically chooses between SBR or a RI method (using decision trees) for the target problem, considering similarities of cases in a case base.

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

2.5.4. **Retrieval and ARs**

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 [21]. 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 [22].

ARs can be used to analyse patterns in such dataset to calculate a target probability whereas CBR is employed to retrieve similar cases [124]. [50] deployed a case association in order to mine the association rules from the implied correlation among cases to retrieve the most similar one. The literature also revealed that in 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 [24].

Where CAR is a specific subset of ARs whose consequents are restricted to one target class, it can be used in CBR to get the cases which are useful to gain the solution for the given problem. In other words, where a result formed as a pair of a solution attributes and its value [51]. In this research, CAR is encoded to show how a specific problem's features are associated with a certain solution.

2.5.5. **CBR Tools**

In the past twenty years, several CBR shells and software frameworks have been created to

simplify the development the development of CBR application in different problems fields. CBR has also been widely utilised to build number of knowledge based systems such as [125], [126] and [127]. Particularly, Many CBR systems have been developed to assist in decision making, problem solving and health care [128] and [129]. Generally speaking, the development of CBR applications is time consuming and needs technical skills. For example, to build a retrieval application, researchers and developers need to be aware of CBR techniques such as case representation, understanding retrieval algorithms i.e. KNN and knowledge modelling. For those who are new to the retrieval of CBR development, the curve is declined because they do not have the enough skills to build a CBR retrieval as an application [130].

In an attempt to simplify the rapid prototyping of CBR applications and decrease the effort in creating CBR applications, CBR shells and software frameworks in the past two decades has been built by the CBR community. These CBR shells and software frameworks basically offer a set of units and features to assist a developer or user without the required knowledge of CBR algorithms to accomplish a CBR application in a quick and easy way. The author will present a brief information of existed CBR tools i.e. shells and software frameworks to justify the usage of Jcolibri and FreeCBR as an evaluation tools against the proposed CBRAR in this research study.

In fact, the CBR tools are sectioned into two parts: First, CBR shells and second, software frameworks. CBR shell is basically an application generates includes a graphical user interface [131]. Multiple features and modules can be usually offered by CBR shells to build a CBR application for example, case base management or performance monitoring via graphical interface. A non-programmers and users need to learn how to use the shell without having a knowledge of the CBR algorithms and techniques but shells tend to be limited function, inflexible formats and may not represent correctly the complexity of cases [132]. In addition, if the

end user needs further functions which CBR shell does not support, a programmer will be required as a resort solution.

The CBR software frameworks have been established to support users with an open environment for CBR application development and to enable non-expert users to construct CBR applications rapidly, and to expand functions without difficulty [133], [131]. Moreover, a software framework typically offers a number of application programming interfaces (APIs), mechanisms and classes, and provides many advantages such as modularity, code reusability and extensibility. The researcher has used Jcolibri [25] and FreeCBR [20] frameworks for the sake of code reusability and extensibility as an advantage features when compared to the CBR shells.

2.5.5.1 CBR shells

A CBR shell is one of the CBR tools that enables end users with a set of capabilities to create CBR applications. CBR shells and their applications are mainly textual-based and inflexible in modification. As end user computing becomes widely used, user created content or applications must develop more versatile and flexible. Therefore, many CBR shells or tools contain shells we review in this section are no longer under development or maintenance due to complexity and reusability. Below is a brief review of several early influential CBR shells but not exhaustive.

- ReMind: is coded in C++ language includes methods like decision trees, nearest neighbour and knowledge-guided retrieval for similarity valuation. In addition, ReMind supports case adaptation by building adaptation formulas that adjust values based on the difference between the retrieved case and the new case [134]. ReMind Version 2.0 is under development at the Navy Center for Applied Research in Artificial Intelligence in Washington DC [135]. It is not clear when the new version will be released or who

may retail it.

- CASPIAN: is an open source written by university of Salford in C language and runs in a command-line mode. CASPIAN uses its own language (CASL) to define cases including case attributes, and weights. CASPICAN utilises KNN algorithm as the retrieval mechanism and employs rules for case adaptation [136]. This tool does not provide APIs for further development or extension.

- CBR Express: is an example of domain-specific CBR shells. CBR Express was mainly designed for help desk applications. CBR Express has a simple case structure and uses KNN matching to retrieve similar cases. The user interface is built using Asymetrix ToolBook, which is consisted as a type of Windows environment. This tool is appropriate for fields where knowledge can be represented by a set of vectors of attribute-value pairs [134]. CBR Express is able to handle free-form text which is significant for assist in desk applications [137]. This tool is sold by eGain company as a conversational CBR application based on demand.

2.5.5.2 CBR Software Frameworks

The necessity for the development of CBR tools based on the open framework environment is recommended by Abdrabou and Salem [131]. They suggested that CBR software developers to emphasis on the development of CBR frameworks rather than shells order to improve software reusability and extensibility. Some of the CBR software frameworks such as CAT-CBR is no longer in use because of many reasons for instance, deficiency of funding support. Below, is a brief review of some influential CBR software frameworks include the tools used in this study.

- Jcolibri: is used in this research as a popular open source CBR software frameworks.

The current version solves many problems found in its predecessor JCOLIBRI1 and helps most phases of the CBR cycle. Jcolibri is built using the Java language and Java Beans to represent the cases which would suit the proposed CBRAR where procedures can be invoked in the same programming language easily [138]. The architecture of this tool contains two layers respectively for source codes (supporting programmers) and composition tools (supporting designers). Moreover, five different retrieval strategies are offered with seven selection methods, several similarity metrics. Jcolibri includes various case representation i.e. flat, simple case and more complex knowledge intensive structures. It also can be workable to generate complex CBR applications. Yet, Jcolibri has been used to build twenty CBR applications including an application for helping criminal justice [139] and health case [140]. It is also used in this research to test datasets from different fields to show the workability of CBRAR in various domains i.e. Space, health care and psychology. It shows just the top 5 retrieve cases in the result panel.

- FreeCBR: is the second tool used in this research for the evaluation purposes as a free open source and java implementation of a CBR engine [20]. FreeCBR offers a graphical user interface, a command line interface and a Web interface. It employs a set of functions and features to represent each case but lacks strong support for a sophisticated knowledge. The Euclidian distance, Normal Distance algorithm are used to calculate the closest match for case retrieval in this research in both tools Jcolibri and FreeCBR. FreeCBR produces more retrieved cases in the results panel when compared to Jcolibri because the recall is higher.

- myCBR: is another popular open-source software framework for developing CBR applications. myCBR is mainly intended for creating CBR applications that focus on the

similarity-based retrieval phase of the CBR cycle. It is also built using Java and capable of supporting complex knowledge intensive case structures uses a powerful GUI-based workbench to define classes and attributes, model and test similarity measures [141]. myCBR has been successfully used to develop various applications including Web-based and mobile CBR applications [142]. However, compared to Jcolibri and FreeCBR, myCBR is not suitable for applications with large number of attributes. It is more suitable for creating non-complex CBR retrieval systems with a small number of cases [143], and that's why it has not been used in this research.

- IUCBRF: is an open source framework that can be used in CBR applications. JUCBRF is implemented in Java and contains multiple domain independent components and tools to support case representation, retrieval phase. It has better support for flat, simple case structures than complex knowledge intensive case structures. The IUCBRF framework has been used as a pedagogical tool to teach CBR in graduate-level artificial intelligence fields [144]. Nevertheless, the author has not used this tool due to lack of new development of this framework in recent years and no further results would be produced at the top of the used tools i.e. Jcolibri and Free CBR.

Table 4 Features of the used and reviewed CBR software frameworks

CBR Software Frameworks	Jcolibri	FreeCBR	myCBR	IUCBRF
Features Support	Wole CBR Cycle	Wole CBR Cycle	Retrieval Phase	Wole CBR Cycle
Application suitability	Works with large number of Cases	Works with large number of Cases	Works with low number of Cases	Lack of new development
Technology	Use object oriented	Use	Better CUIs for medeling	Used for independent

	framework in jvav	weighted Euclidian distance to retrieve cases	similarity knowledge	domain
Year released and latest version	2005 – Jcolibri2	2006 - FreeCBR 1.1.4	2007 – myCBR 3.0.1	2005 - IUCBRF

2.5.6. **Soft Matching of ARM (SARM)**

A limitation of traditional ARM algorithms for rule $X \rightarrow Y$ e.g. Apriori [72] 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 [145], 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.

2.5.7. **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 [24]. However, experts are required for calculating and defining the SK domain, making this a time consuming and difficult process.

2.5.8. **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 [24]. USIMCAR is used to enhance the usefulness of cases, retrieved through the SK [51], with regard to a new case Q in addition to including the SCAR, thus meaningfully utilizing the cases with their usefulness [51]. 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 [32] 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 [24] and [32] 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 research, 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.

2.5.9. **Domain Knowledge and SBR**

SBR has also been combined with Domain knowledge (DK). The latter has been used to improve on similarity measures in order to retrieve more precise cases for a target problem. DK can be encoded in the form of accurate training cases where these are selected from the feedback about the benefit of some cases formerly assessed by domain experts [146]. 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 [147]. However, there is no explicit form of DK, and therefore implicitly available through the support of human domain experts which use informal methods when being asked to define such knowledge. So far it is believed that the implementation is hard especially if given domains are weak in domain theory. Whereas building AK is straightforward as it is acquired via an analysis of cases, an essential knowledge source in CBR, without the support of domain.

2.5.10. **Similarity Knowledge and Association Knowledge**

This section provides the background of AK and SK. Basically; it presents a case representation scheme prior to presenting our proposed CBRAR approach. To represent cases, the selection of the attribute–value pairs representation is used. This approach is widely utilised in many CBR systems, due to its simplicity, flexibility, and popularity [24]. Let *A1, ...,Am* be attributes defined in a given domain. An attribute–value pair is a pair (*Ai,ai) i*∈[1,*m*], where Ai is an attribute (or feature2) and *ai* is a value of *Ai*. A case *C* is the form of *C = (X, Y)* where *X* is a problem *X = {(A1, a1), ..., (Am −1, am −1)},* and *Y* is the corresponding solution *Y = (Am, am)*. The point of *Am* as a solution attribute, A case base is a set of cases. The combination of AK and SK is called SCAR [24] [32]. It finds the relation between problem features ($X \Rightarrow Y$) and the target problem (Q) through a special measure. It is likely associated with the solution

contained in *y,* if *Q* problem features are sufficiently similar to *X*. Often; a rationale for doing so is to define a single optimal interestingness (e.g. Laplace method [148]).

2.5.11. **Similarity Knowledge**

In a CBR system, SK is indicated as knowledge encoded via measures computing similarities between a target problem *Q* and cases. Normally in SBR, SK is used to represent a heuristic for calculating the usefulness of stored cases with respect to *Q*. The higher the similarity between a case *C* and *Q*, the more useful *C* is for *Q*. A composition of similarity measures suitable for cases represented by attribute – value pairs is often based on a widely used principle. This is the local–global principle that decomposes a similarity measure by local similarities for specific attributes of cases and a global similarity aggregating these similarities [146]. An accurate local similarity function depends on attribute types. A global similarity function can be arbitrarily complex, but simple functions are usually used such as weighted average aggregation [146].

2.6. Summary

In this chapter, an overview of CBR background including case: parts, representation, bases, Methods and Retrieval were given for illustrating their challenges. An explanation of data mining common methods including classification and KNN was presented. The brief survey illustrated and described a number of equations that utilise KNN for measuring distance in CBR field. A through overview of ARs algorithms was presented. The presented overview for ARs also described the AK that has been selected to develop FP-CAR as a part of CBRAR. A survey of frequent pattern mining concepts that are used in this research for optimising the FP-CAR tree was displayed. An overview of the related work and other types of knowledge was presented. The overview illustrated various methods of integrating data mining and CBR, and give

and examples of SBR and statistical learning has integrated. It also shows different methods and techniques that have been used to integrate ARs into CBR to enhance the performance of Retrieval.

In the next chapter, a new retrieval strategy for integrating CARs into CBR will be proposed. A description of the proposed CBRAR is explained in details. Moreover, typical scenario of the FP-CAR algorithm and pseudo code that enhance the performance of retrieval phase will be given.

Chapter 3: New Retrieval Strategy CBRAR and New Algorithm FP-CAR

The previous chapters presented the research's problem, motivations, methodology, and contribution as well as review of CBR, data mining methods and related work. This chapter develops and describes a new strategy and algorithm to enhance the performance of the CBR retrieval phase by using classification based on association.

The Chapter therefore, contains the following key areas:

- New retrieval strategy: presents a new retrieval strategy (CBRAR) that is able to disambiguate the wrongly retrieved answers in order to enhance the performance of SBR.
- Proposed algorithm FP-CAR: proposes a new algorithm FP-CAR tree that mines all CARs and contains potential patterns to be compared with CBR case problem.
- FP_CAR algorithm: displays FP-CAR algorithm pseudo code that refines the retrieval technique to select the correct case not just a similar one.
- Summary: presents a short summery of chapter 3.

3.1. New Retrieval Strategy

This section presents the proposed new technique CBRAR that attempts to integrate CARs and CBR. Basically, there is a known problem in CBR which is retrieving unrelated cases that give incorrect solutions. 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 SBR. However, SBR tends to depend on similarity knowledge, ignoring other types of knowledge that can benefit and improve the 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 [24], are much dependent on the experts domain for finding SK. [32] 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 [24], [32] 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 Figure 9, we start to remove one case from the case based library of the CBR until the system retrieves two different labels 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 in the tree. 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 is processed by CBR, SBR may retrieve unrelated cases from the case library with the same similarity measures as shown in Figure 9 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 the class root.

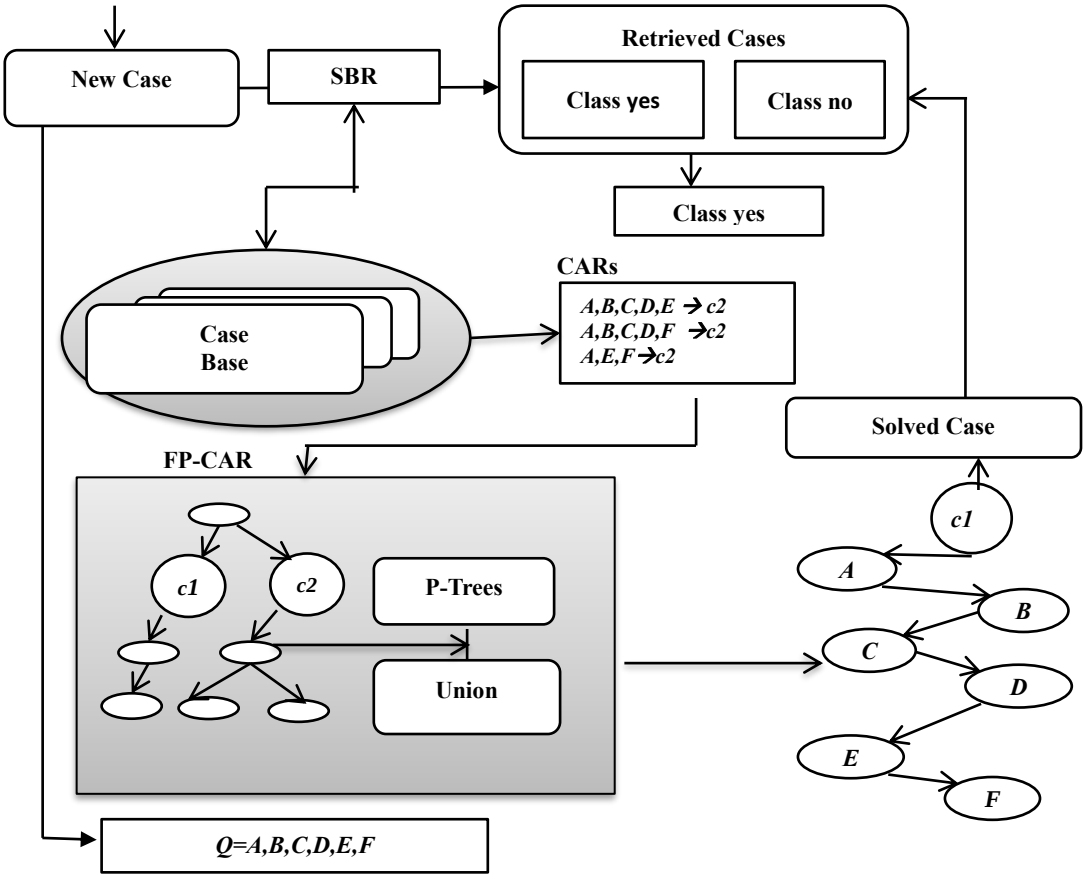


Figure 9 CBRAR Model

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 for finding a partial match.

- P-trees: a P-trees procedure or union are invoked to complete any missing nodes

in the tree if needed to form an equivalent pattern to the CBR query.

In the new CBRAR strategy, the CAR method is adapted to produce class association rules to be mined instead of general association rules. This approach has been used by [15] , where the algorithm drew new class rules from old general rules according to class labels which had been predefined. CARs differ from general ARs mining by presenting constraints to any attribute that is definitely appearing on the generated rules. CARs are a special case of constrained AR which can be utilized to construct a model or classifier [149] [9]. The major advantage is faster execution and lower memory utilization. CAR is theoretically motivated but it has not been used to produce FP-tree. Thus, the new system attempts to use classified rules as an input to the new algorithm to build a FP-tree which has not been used before in the area of integrating AR into CBR.

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 Figure 9. It can be seen that two different labels i.e. classes (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.

3.2. Proposed Algorithm FP-CAR

In this section the modified FP-Growth (FP-CAR) algorithm which is part of the new retrieval strategy is discussed. As explained in the state of the art section, FP-Growth works in a divide and conquer manner. It passes through two scans of the database. Firstly, it computes a list of frequent items stored in descending order (F-List) during the first scan of the database. In the second scan, it compresses the database into the FP-tree. The process of growth then starts to

mine each item with support greater than or equal to ξ (lower support) by building its conditional FP-tree recursively. This tree will be modified into classified labels and to be compared with CBR patterns to find the correct a solution.

The new method tries to change the FP-Growth algorithm in order to consider a class while producing a frequent pattern tree (FP-tree). This is because in mining association rule algorithms, any associated method does not consider class clusters in the process of producing frequent patterns belonging to 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-Growth 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 and new tree.

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 [150]. Second, the tree is optimized by utilizing the P-tree [34] and 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 ξ = 0.1 and confidence = 0.9, delta = 0.05, number of rules = Maximum). Then the existence of the rule $X \rightarrow c$ as 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 addition, 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 first experiments use the acute inflammation dataset from UCI (Table 5, **Table 8**, **Table 9**, **Table 10**, Figure 9 and Figure 10). In this case, as in Coenen [35], 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* [96].

In this research 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 [150]. 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 [150]. It builds a set-enumeration tree structure of the CARs, where the FP-tree contains an incomplete summation of support-counts for relevant sets and patterns. Using the FP-tree structure to represent all patterns of the CARs, the T-tree [34] 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*.

Figure 10 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 5. 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.

We now build the supersets of all such sets that match the new case *Q = {yes, yes, yes, yes, no 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 Figure 10. Connecting the

tree in Figure 10 into the results given in **Table 8**, **Table 9**, **Table 10** and **Figure 24** supports the theory behind the proposed algorithm [151].

Table 5 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*

Furthermore, the union set theory to improve the results is used. Union (U), Set X, Set Y, $XUY=\{x: x \in X \text{ on } x \in Y\}$, it is assumed that X and Y are subsets and the union result is $X \cup Y=\{A,B,C,D,E,F\}$. For example, *{ABCDE}* is subset1 where $X \Rightarrow c2$ of $L=5$ and *{ABCDF}* is subset2 to be united as $Y \Rightarrow c$, the superset will be constructed of the said sets that equal the new case $Q = \{1, 3, 1, 1, 1 \text{ and } 1\}$ i.e. $\{XUY\} \Rightarrow c$ as achieved in Ex 8 which are connected to Table 12 and Figure 25 [152]. The following sections explain how the result of the proposed strategy is evaluated.

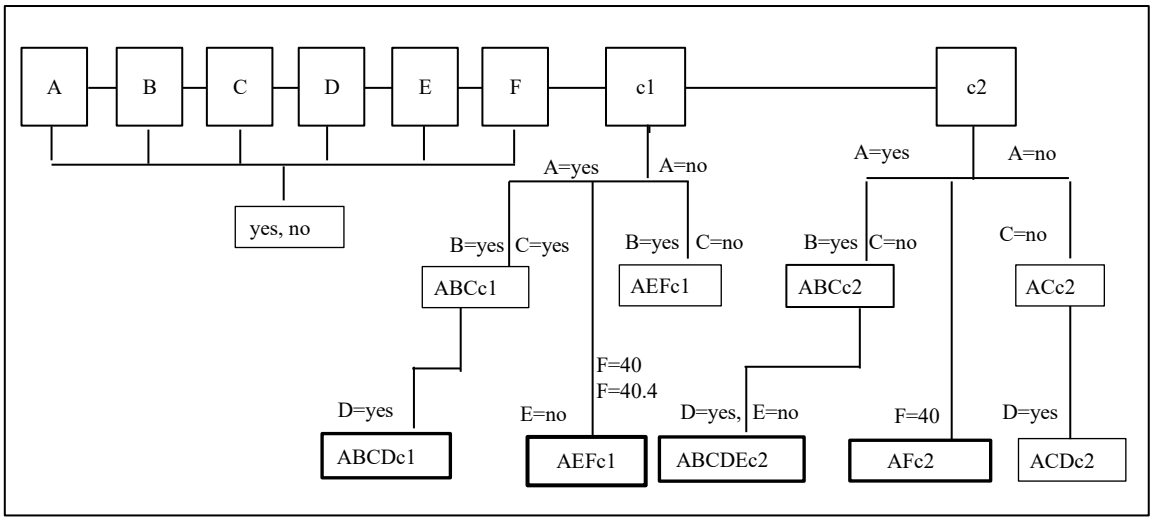


Figure 10 FP-CAR Algorithm Tree - Acute Inflammation Dataset

3.3. FP-CAR Algorithm Pseudo Code

Figure 11 shows the pseudo code for the proposed algorithm. The first step of FP-CAR is to generate a sorted hash table of items, and to remove infrequent items. Table 6 shows that 92 rules are ordered depending on their highest frequency. For example, item I2 always appears first in the ordered items table i.e. R92(I2,I1,I3,I4,I5) because of the frequency of I2=62 followed in a descending order by items I1=58,I3=50,I4=49,I5=26 according to its class c1.

In this version we set $\xi = 1$ to find the maximum routes of FP-CAR patterns. FP-CAR then, compresses these rules into a prefix tree, where the root c1 holds all frequent items according to this order I2, I1, I3, I4, I6 and I5, where item I2 in Table 6 refers to node A in Figure 10 followed sequentially by nodes *B,C,D,E,F*. Each path of the tree also represents a subset of rules that share the same prefix where each node corresponds to one item and is linked to the next node of the same subset. In addition, the list of items is formed to align all rules that possess that item. The FP-tree is a compressed representation of the rules and it also permits quick access to all rules that share a specific item. Once the tree has been built, the comparison of the new case can be conducted. However, a compact representation does not reflect all potential candidates' patterns which are the bottleneck of the original FP-Growth algorithm.

This problem is overcome by using the concepts of P-trees 2.4.3. in order find a potential pattern by combining two frequent subsets to form a superset. The key step in the algorithm is to convert the transactions in a database into rules as an input to FP-CAR so that a new FP-tree is built from different classes during the recursive conditional constructing process. Also, logical equivalences concepts are utilized to prove that $(p \rightarrow r) \vee (q \rightarrow r) \equiv (p \wedge q) \rightarrow r$ [153]. Assuming that, p = subset1, q = subset2 and r = class c1. According to first order logic if sub1 implies c1 or sub2 implies c1, its equivalents sub1 and sub2 both imply c1. The implication concept is employed to support P-tree in case a potential solution is gained by FP-CAR.

```
Algorithm 1: FP-CAR(DB, ξ)
Define and clear F-List : F[];
foreach Transaction Ti in DB do
   foreach Item aj in Ti do
      F[ai] ++;
   end
end
Load Rules-Strings;
Genrate Hash Table H from Rules-Strings;
F[]← Sort (Ti);
Sort F[];
Define and clear the root of FP-tree : r;
Define parent root of each Class Ci;
foreach Rule Ri in DB do
         if Ri ∈ Ci // Splitting
   Make Ri ordered according to F;
   Call ConstructTree( r, Ci);
end
Define NewCase NC;
Subset, Ss = Ri;
MaxLength = new ξ; // Voting
foreach Subset, Ss in I do
      if Ss = NC // Comparing
         Result R = NC;
      Else if Call Ptree( Ss, MaxLength , R );Else Union(Ss1, MaxLength , R), Ss2)
                Go to Voting;
end
```

Figure 11 Algorithm 1 FP-CAR

Table 6 Hash Table with Classes

Rules	List of Items	Frequency	Class
R1	I2, I4	49	C1
R2	I3, I1	41	C2
R3	I1, I4,	41	C2
R4	I3, I1, I4	41	C2
R5	I2	41	C2
etc..			
R92			

Item	Frequency	Priority
I1	58	2
I2	62	1
I3	50	3
I4	49	4
I5	26	6
I6	44	5

Frequent item list or L

Priority of rules i2,i1,i3,i4,i6,i5

Rules	Ordered Items	Class
R1	i2,I4	C1
R2	I1,I3	C2
R3	I1, I4	C2
R4	I1,I3,14	C2
R5	I2	C2
etc..		
R92	I2, I1, I3, I4, I5	C1

Algorithm 2 presents the pseudo code of the last step in terms of a partial solution using the FP-CAR algorithm. In the experiments, the transaction *Ti* is considered as *Ri* to represent all possible rules resulting from the constructed tree. Note that for each set *i* in *T, Ti is* equal to 0 to set a new transaction. Also for each node *j* in the P-tree the procedure begins by subtracting *j* nodes from its parent. Then for each *i* in the transaction, if $i \subseteq j$ and $i \cap k$ is not empty the procedure adds a new node to *Ti*. Notably, this algorithm makes use of a concept introduced in [96] to ensure that any subset, for example the contribution in respect of the {a, b, c} is already included in the interim total for {a, b, c} , so when considering the node {a, b, c, d}, the test will be for only those which include the attribute {d}. This technique is explained in detail in section 2.4.3.

Hence, two subsets can be merged to combine into one superset if they belong to the same root. For instance, sub1 (i2=yes,i1=yes,i3=yes and i4=yes) and sub2(i2=yes, i6=no and i5=40) can

form a superset sub3(i2=yes,i1=yes,i3=yes,i4=yes,i6=no and i5=40) which can be compared to a new case if needed.

```
Algorithm 2 (Inputs P − tree P, candidate set T):
−Returns counts Ti for all sets i in T−
∀ sets i in T do Ti = 0
∀ nodes j in P do
begin k = j − parent (j);
∀ i in T , i ⊆ j, i ∩ k not empty, do
begin add Qj to Ti
  end
end
```

Figure 12 Algorithm 2 P-Tree

As shown in Figure 13, case C73 (sub3) is represented as a tree and compared to the FP-CAR tree, where sub1 and sub2 which belong to Class 1 can be merged to form sub3. CBR retrieves 5 cases, three cases of class c1 (71, 72, 79) and two cases of class c2(76,77). A length voting process is performed on c1 and c2, which identifies that c1 contains 4 edges as the longest length match compared to the new case. The longest length is (sub1), and (sub2) represents the 2 missing nodes to complete the full path of a query (sub3). When Sub3 is compared to the CBR results it will disambiguate the 5 answers of the retrieval phase. Hence, it removes the limitations of SBR when returning wrong cases. Moreover, this new strategy refines the retrieval technique to select the correct case not just a similar one.

3.4. Summary

In this chapter, a typical model to enhance the performance of CBR was presented. A new strategy CBRAR for disambiguating uncertain answers of CBR retrieval was displayed. CBRAR strategy compares the outcome of the solved case with the result of the retrieved cases to remove unrelated answers. CBRAR includes a proposed FP-CAR algorithm which mines all CARs patterns using FP-Growth concepts with a target case to gain the correct answer. A clear example was given using real dataset values to illustrate the objectives of the new model.

In the next chapter, extensive experiments will be demonstrated to validate this proposed strategy as well as the evaluation in order to compare the performance of CBRAR with the existing CBR tool.

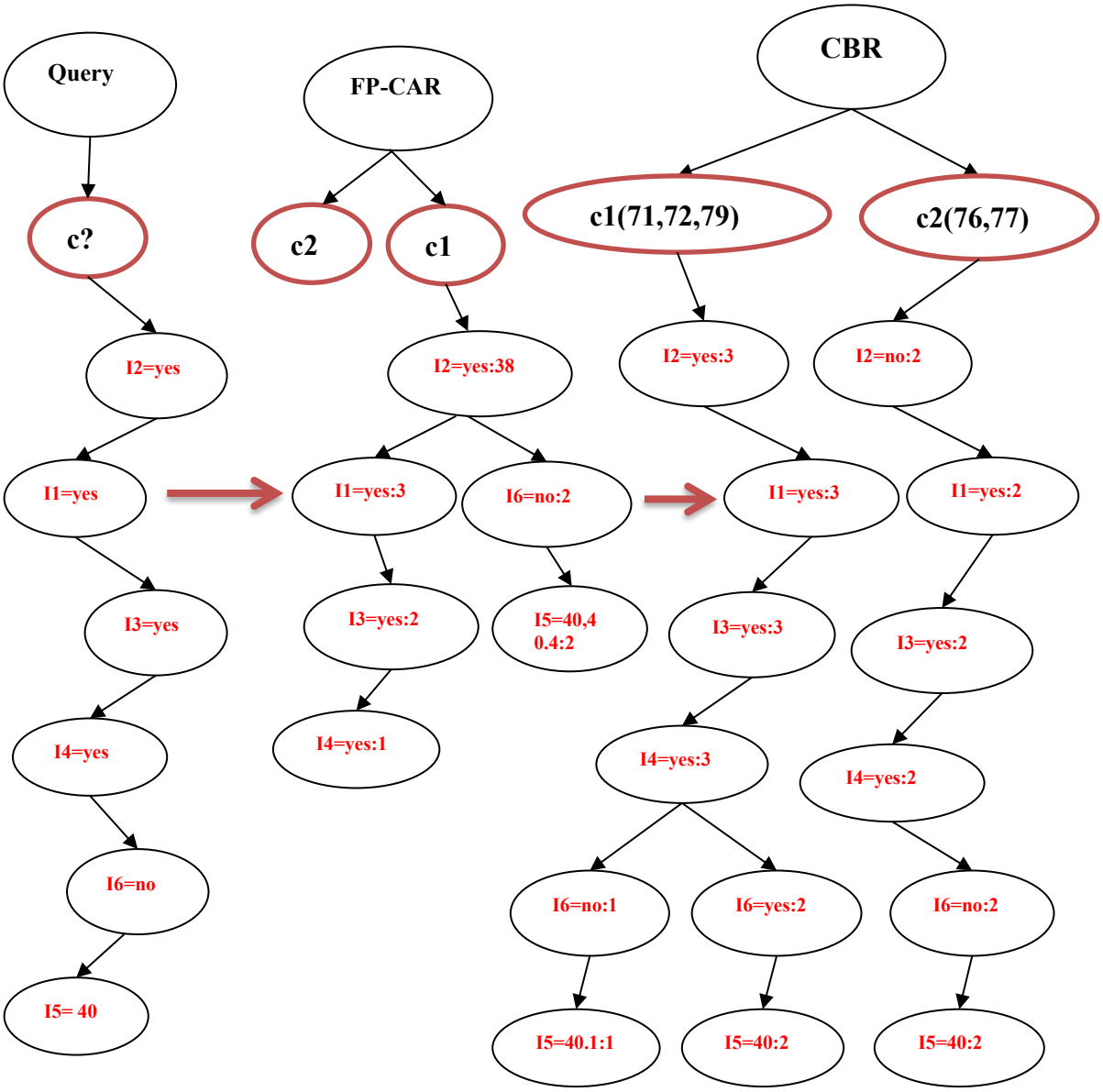


Figure 13 Solved Case Compared to CBR Results

Chapter 4: Experiments and Empirical Evaluation

Chapter 3 describes the new retrieval strategy based on the proposed algorithm for this research, where CAR and FP-trees methods are used to achieve the objectives of integrating CARs into CBR. This chapter presents the experiments and evaluations carried out to date and is organised as follows.

- Section 4.1. describes the experimental process.
- Section 4.2. describes the metrics used to evaluate the performance of the proposed strategy.
- Section 4.3. explains the datasets used in all experiments.
- Section 4.4. provides an overview of the analysis process using a case study.
- Section 4.5. presents a further analysis and experiments of acute inflammation urinary bladder dataset and discussion.
- Section 4.6. illustrates the results of the "space shuttle dataset", experiments, tables, charts and discussion.
- Section 4.7. describes experiments on the "balloon dataset" and illustrates, tables, charts and discussion.
- Section 4.8. describes experiments on the "post-operative dataset" and illustrates tables charts and discussion.
- Section 4.9. describes experiments on the "Lenses dataset" and illustrates tables charts and discussion.
- Section 4.10. gives a summary of this chapter.

All experiments were conducted using an open source CBR framework i.e. Jcolibri [19] and FreeCBR [20]. This software can be used not only for CBR rapid prototyping but also for

developing applications using real scenarios. It has the advantage of using a java platform allowing it to be integrated with other applications. The Waikato Environment for machine learning, WEKA version 3.6, [83] is also used as an open source machine learning environment containing a variety of algorithms. WEKA is becoming a well-used tool in both academia and industry. It has been used to generate the rules i.e. CARs and predictive rules. Eclipse IDE for Java Developers (Version: Mars Release (4.5.0)) open source has been used for building the new FP-CAR algorithm.

Datasets from the UCI website are used to evaluate the performance of SBR and CBRAR. The acute inflammation dataset used and referred to as D1 is used for the initial experiments. The FP-CAR is compared to SBR using the D1 dataset.

4.1. Experimental process

To investigate the accuracy of CBRAR, we conducted experiments using datasets taken from the UCI Machine Learning Repository. The implementation of CBRAR used the Java platform Eclipse (4.5.0), and for comparison purposes we have used the Jcolibri framework [19] and FreeCBR [20] as powerful CBR tools. WEKA 3.6 is used as an open source in order to generate the CARs.

In the conducted experiments, one case is left out from the CBR case until the pre-determined cases register an ambiguity. This ambiguity can be deceiving the decision maker as all retrieved cases have the same percentage of similarity with different labels. Thus, experiments are used for both Jcolibri and FreeCBR.

We used the derived dataset from the UCI repository as the same source to measure the CBR and CBRAR accuracy.

By default, SBR returns the 5 most similar answers when using Jcolibri with a new tested case.

Nonetheless, the pre-determined cases of the tested datasets all registered an uncertainty that misled the end user, due to the fact that all the retrieved cases had an equal percentage of similarity with different labels i.e. ((1, 2) or (yes, no)). Notably, the FreeCBR tool provides further prospective cases in addition to those chosen by Jcolibri. With regards to the tables displayed in each experiment, the findings show that vertically, the first column refers to the new case Q followed by the cases retrieved by the CBR tools. For example, NewCase73 followed by cases (71, 72, 76, 77 and 79, for Jcolibri) and cases (71, 72, 77 until 107 for FreeCBR) as shown in Table 8.

The "Attributes" column begins with the first attribute followed by additional attributes. For instance, the attribute A then by 5 supplementary attributes B, C, D, E and F. The class label column refers to 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 tables of experiments, we use symbols TP, TN, FP and FN to denote True Positive, True Negative, False Positive and False Negative respectively. The assumption is made to indicate the four probabilities on the confusion matrix.

The table for each experiments illustrates that for every novel case tested using CBR, 5 cases with the same similarity measure are recovered by Jcolibri Once the New Case is used, Jcolibri retrieved more (TP, FP) and (TN, FN) cases with the same similarity ratio of accuracy. Some cases were overlooked due to a lower similarity i.e. (0.816) is ignored if Jcolibri retrieved (0.912) of accuracy, (42.264) is ignored when FreeCBR registered (59.175) of accuracy. While, CBRAR recovered 1 TP case from the novel model.

With regard to the second experiment, the NewCase11 was applied to the CBR, and Jcolibri and FreeCBR retrieved 2 TP and 1 FP with an accuracy of 66%. At the same time, CBRAR retrieved 1 TP case from the proposed algorithm.

All the resolved cases can be reworked in the figures of the FP-CAR algorithm trees. One of the most noteworthy results in this research is the greatly improved accuracy rate obtained in CBRAR, when unions of two rules were employed in order to match a target case. With CBRAR offering a resolution to the uncertainty of the FP cases minus the confusion. For instance, case 10, 11, 12, and 15 could be recalculated in Figure 25 in order to verify that CBRAR determines a correct case using the FP-CAR tree.

The bar charts of each experiment depict a comparison of the error rate and accuracy of traditional CBR tools i.e. Jcolibri, FreeCBR with the proposed model in order show the enhanced performance of CBRAR in the overall error rates. In other words, to show how the proposed CBRAR registers lowest error rates and highest accuracy by correctly resolved cases when compared to other CBR tools.

4.2. Performance Evaluation Metrics

As mentioned in the introductory chapter, the work presented in this thesis is experimental in nature. Hence, it is important that all experiments are designed and evaluated in a systematic and reproducible manner. As is the case in many fields, in data mining and machine learning, a key part of any study is how the system is evaluated. This section summarises the methods used in this study to evaluate the performance of the developed strategy CBRAR against existing CBR tools i.e. Jcolibri and FreeCBR.

The confusion matrix displays the contingency table, with two dimensions ("actual" and "predicted") that allows analysing and visualizing the performance of the compared system. It contains instances for the actual and predicted data [154], [155]. **Table 7** illustrates a confusion matrix, where ([156] , [157] :

Table 7 Confusion Matrix

	Predicted		
Actual		+	-
	+	TP	FN
	-	FP	TN

- TP (true positive) signifies the instances that are correctly predicted and marked as positive.
- FP (false positive) refers to instances that are incorrectly classified as positive.
- TN (true negative) indicates instances that are correctly predicted as negative.
- FN (false negative) determines instances that are incorrectly classified as negative.

There are different techniques used to evaluate the performance of systems/classifiers. These techniques are described in the following paragraphs.

The recall and precision rates [156] and [157], which originated from IR, are broadly used in the empirical studies of AI to measure experimentally the effectiveness of machine learning methods.

Precision is a statistical method that measures the probability of retrieving relevant instances which are divided by the total number of the retrieved instances. **Equation 8** denotes the formula of the precision *P* [121].

$$P = \frac{relevant\ instances\ retireved}{retrieved\ instances} = \frac{TP}{TP + FP}$$

Equation 8 Precision Equation

Recall rate is a statistical method that measures the probability of retrieving relevant instances which are divided by the total number of the existing instances that are expected to be retrieved.

Equation 9 displays the formula of the recall *R* [121]:

$$R = \frac{relevant\ instances\ retireved}{total\ instances\ to\ retrieve} = \frac{TP}{TP + FN}$$

Equation 9 Recall Equation

The accuracy is the proportion of instances that are correctly classified. **Equation 10** shows the *accuracy ACC* formula [121]:

$$ACC = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

Equation 10 Accuracy Calculation Equation

Misclassification rate also known as "error rate" for any classifier is the proportion of instances that are incorrectly classified. **Equation 11** displays the error rate *Er* formula.

$$Er = 1 - ACC$$

Equation 11 Error Rate

4.3. The Datasets used in the Experiments

To investigate the accuracy of the CBRAR for a validation purposes, we conducted experiments using datasets taken from the UCI machine learning repository. The five derived datasets are used as the same source to measure the CBR and CBRAR accuracy. The datasets are chosen from different fields to show that the CBRAR is workable in different contexts. For example: health case (acute inflammation, post-operative, Lenses) datasets, cognitive psychology (Balloon) dataset and space (space shuttle). The main characteristics of the used dataset will be stated so any target case can be reworked in the FP-tree figures as follows.

Acute inflammation dataset is used in the first experiments, the attributes values that were used in the hash table and FP-CAR tree are as follows {I2= A=urine pushing {yes, no}}, {I1=B=lumbar pain{yes, no}}, {I3= C=occurrence of nausea{yes, no}}, {I4= D= micturition

pain{yes, no}}, {I6=E=burning of urethra{yes, no}},{ I5=F= temperature of patient{ 35C-42C }}.

The space shuttle dataset is used in the second experiments, the attributes' values and characteristics that were used in the hash table and FP-CAR tree are as follows {A= Magnitude {Low, Medium, Strong, Out Of Range}}, {B= Error { xl, lx, mm, ss}}, {C = Sign{pps, nn}}, {D = Stability{stab, xtab}}, {E= Wind {head, tail}}, {F= Visibility {yes, no}}. The values of attributes appear in the FP-CAR tree as a numeric. In addition, the class characteristic {Class =c{non-auto=1, auto=2}}. The attributes' values between the curled parentheses are converted to numeric range between {1-4} in order to make it easy to draw the nodes of FP-CAR tree as shown in a.

The balloon dataset used in the third series experiments, the attributes' values and characteristics that were used in the hash table and FP-CAR tree are as follows {A= act {Stretch, Dip}}, {B= age {Adult, Child}}, {C = colour {yellow, purple}}, {D=size {Small, Large}}. The values of attributes appear in the FP-CAR tree as a string. In addition, the class characteristic is {Class =c{ True=c1, False=c2}}.

The Post-operative dataset used in the fourth experiments, the attributes' values and characteristics that were used in the hash table and FP-CAR tree are as follows {A= stability of patient's core temperature{stable, mod-stable, unstable}}, {B= stability of patient's surface temperature{stable, mod-stable, unstable }}, {C = patient's perceived comfort at discharge as integers {0 - 20}}, {D= oxygen saturation in %{excellent, good, fair, poor} , {E= patient's internal temperature in Centigrade{high, mid, low} , {F= stability of patient's blood pressure {stable, mod-stable, unstable} , G= patient's surface temperature in Centigrade {high, mid, low} , H = last measurement of blood pressure { high, mid , low}}, class characteristic is { c1= patient sent to general hospital floor, c2= patient prepared to go home , c3= patient sent to Intensive

Care Unit}}.

The Lenses dataset is used in the fifth set of experiments, the attributes' values and characteristics that were used in the hash table are as follows {A= astigmatic {no, yes}}, {B= spectacle prescription{ myope, hypermetrope }}, {C = tear production rate{ reduced, normal}}, {D= age{ young, pre-presbyopic, presbyopic } }, class characteristics are { c1= not fitted, c2= soft contact lenses , c3= hard contact lenses }}.

4.4. An Overview of the Analysis Process using a Case Study

This section provides an overview of the analysis process using three case studies of acute inflammation dataset, which was a heuristic method that led to build the FP-CAR tree includes enough number of rules. The notion is to mine all CARs to create an optimum tree which could be comparable to target case. In Ex1, Predictive Apriori is used but the number of rules were inadequate to produce an optimum tree. In Ex2, Apriori is used, the number of CARs were reasonable to gain an optimum tree but a comparison between a target case and the built tree was un achievable because the tree does not include a value of nodes. Therefore, In Ex3, Ex2 is extended to include a value of each node where a target case can be found within the built tree of the mined CARs so an optimum tree can be achieved to compare a case with a tree. Ex3 covered the drawback in Ex1 and Ex where values of nodes are employed and enough number of rules are generated gain an optimum tree to fulfil the objective of this research. Ex1, Ex2, and Ex3 will be explained in more details in the next sections.

4.4.1. Experiment 1: Using Predictive Apriori to Produce a FP-tree

In the first experiment (Ex1), the dataset of acute inflammation of urinary bladder is used. According to the hash table of the FP-tree, six attributes are ordered as follows (I1, I2, I3, I4, I5, I6; with two labels (c1, c2)).

In this experiment, we will describe the process in a very detailed manner to allow the reader to understand the process. However, in subsequent experiments, we will avoid much of the detail and will highlight only those aspects that are different from other experiments.

In Ex1, the rules are generated using the predictive Apriori algorithm. The results show that there are only seven rules with one class on the right hand side.

These rules are used to produce a FP-tree where each node will represent an item with its frequency, sorted recursively according to the hash table.

In the first experiment, the dataset items are referred to according to the generated rules of the Predictive Apriori algorithm as follows: I1: Lumbar pain=yes, I2: Urine pushing= {yes, no}, I3: Occurrence of nausea=no, I4: Micturition pains={yes ,no}, i5: Temperature='(35 - 41.4]', i6: Burning of urethra=yes, c1: Inflammation of urinary bladder =yes and c2: Inflammation of urinary bladder =no.

Figure 14 shows that Table T2 lists six items of the generated seven rules shown in Table T1. T3 represents a hash table classified according to classes (c1 and c2) where each rule belongs to a class. Furthermore, the outcome of T3 is a prior step to building the FP-tree and items are ordered in a descending order to produce a compressed tree.

In Figure 15, R1of T3 builds the first two nodes I2 and I1 with a frequency 1 i.e. (I2:1, I1:1) of c1. R2 in T3 of Figure 14 will form the next three nodes I2, I3, I4 in Figure 15 with a frequency 1 i.e. I2:1, I3:1, I4:1 of c2, the frequency will be increased every time the node is traversed and so on until all rules belong to c1 or c2.

In Figure 15, for example, R6 of T3, starts from the node I2:1 because it is already built by R1, the frequency must be increased to I2:2 because of the second traversal. A new path of nodes will be built starting from I2:2 i.e. followed by nodes I3, I4 consecutively of frequency of 1 i.e.

(I2:2, I3:1, I4:1), until all rules build the FP-tree.

T1

Rules	List of Items	Frequency	Class	Accuracy
R1	I1,i2	40	c1	0.99494
R2	I3,I2,I4	30	c2	0.99485
R3	I2,I4,I5	30	c2	0.99485
R4	I6,I1	22	c1	0.99465
R5	I6,I5	21	c1	0.99460
R6	I3,I2,I4	21	c1	0.99455
R7	I3,I4	20	c2	0.99455

T2

Item	Frequency1	Frequency2	Priority
I1	2	40,22	5
I2	4	40,30,21	1
I3	3	30,21,20	3
I4	4	30,20	2
I5	2	30,21	4
I6	1	22	6

T3

Rules	Ordered Items	Class	Accuracy
R1	i2,I1	c1	0.99494
R2	I2,I4,I3	c2	0.99485
R3	I2,I4,I5	c2	0.99485
R4	I1,I6	c1	0.99465
R5	I4,I5	c1	0.99460
R6	I2,I4,I3	c1	0.99455
R7	I4,I3	c2	0.99455

Figure 14 Hash Table of Predictive Apriori

The main weakness with this tree is that there are not enough produced rules in order to generate a FP-CAR. In other words, the more rules obtained, the more patterns can be compared to the CBR query. In addition, the label appears in some rules accompanied by another item on the right side of the implication, namely (item1, item2 → item, label) for instance, (I2, I3 → I2, c). Therefore, the compressed tree will not lead to any heuristic method using Predictive Apriori where no clear label appears in the rules. The target case of CBR will not match any pattern in the frequent pattern tree (FP-tree).

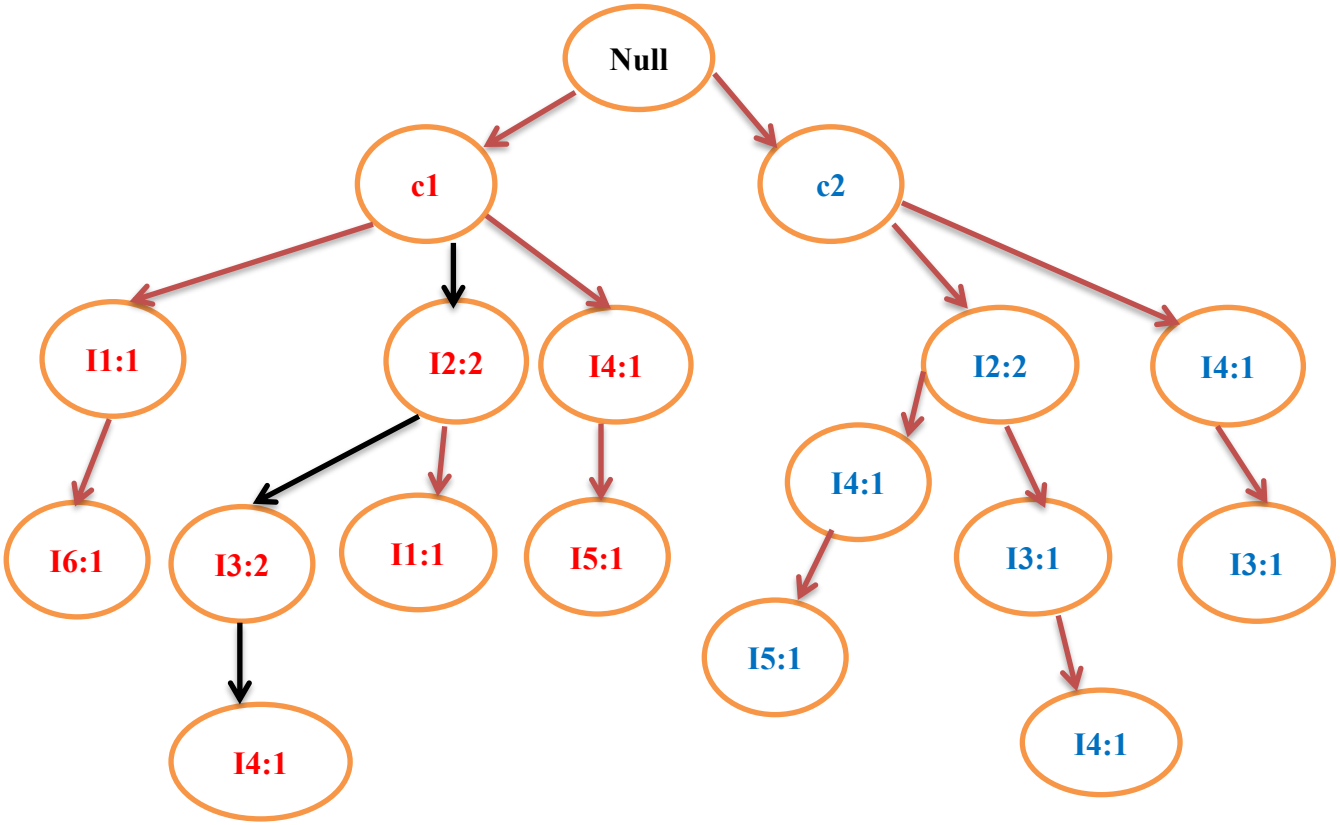
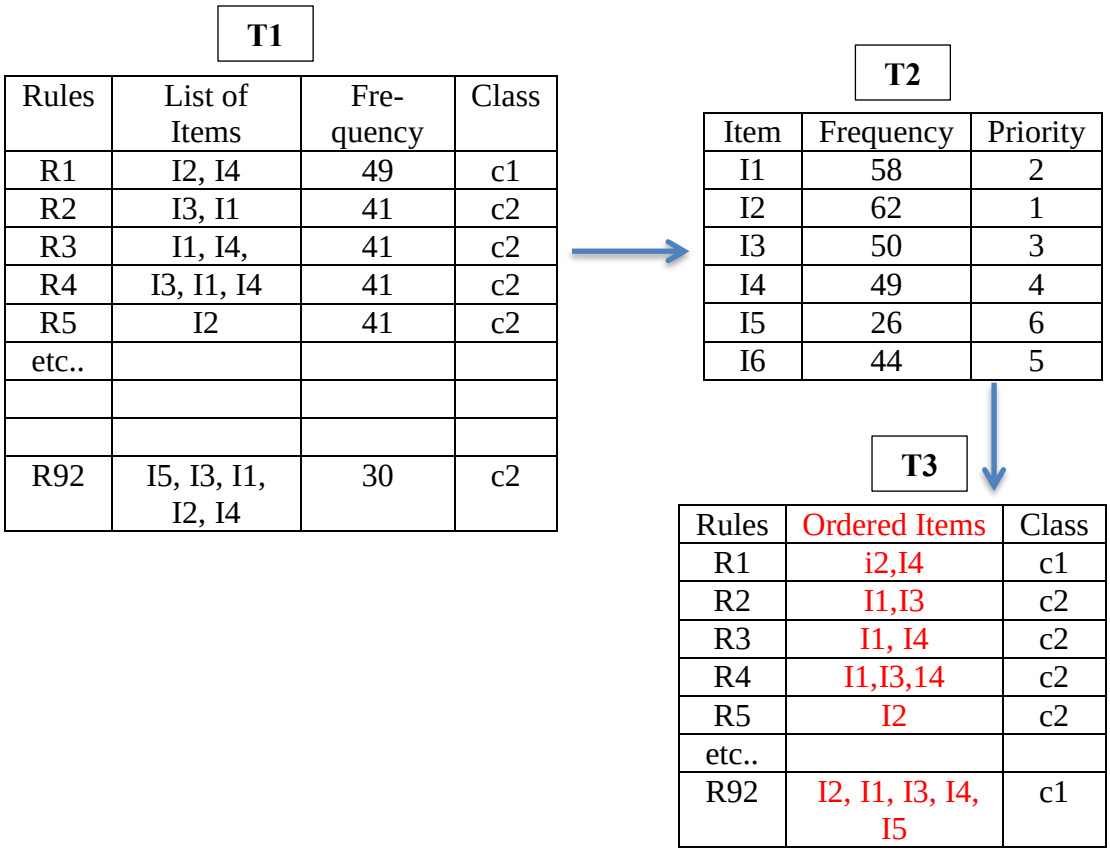


Figure 15 Experiment 1 Tree

4.4.2. Experiment 2 CAR_Rules without Nodes' Values

In the second experiment Ex2, the dataset D1 is considered as mentioned in Ex1 with one label c1: Inflammation of urinary bladder (yes and no). In Ex2, 92 rules (CAR rules) are generated by class. Ex 2 attempted to generate the maximum number of rules to gain a maximum number of patterns classes. These rules were used to construct a FP-tree where each node represented the frequency of items sorted in a descending order according to the hash table. The purpose of this experiment was to find enough rules to build a comparable tree with the pattern of the CBR query. In Figure 16, Table T1 shows that there are 92 rules listed and classified (c1 and c2). Table T2 reflects six genuine dataset attribute items of D1. T3 represents all rules classified according to their labels to construct a tree. Therefore, the new tree will be formed according

to T3 where each route generates a compressed tree pattern.



T1

Rules	List of Items	Fre-quency	Class
R1	I2, I4	49	c1
R2	I3, I1	41	c2
R3	I1, I4,	41	c2
R4	I3, I1, I4	41	c2
R5	I2	41	c2
etc..			
R92	I5, I3, I1, I2, I4	30	c2

T2

Item	Frequency	Priority
I1	58	2
I2	62	1
I3	50	3
I4	49	4
I5	26	6
I6	44	5

T3

Rules	Ordered Items	Class
R1	i2,I4	c1
R2	I1,I3	c2
R3	I1, I4	c2
R4	I1,I3,14	c2
R5	I2	c2
etc..		
R92	I2, I1, I3, I4, I5	c1

Figure 16 Hash Table of CAR_Rules

Basically, in T3, the 92 rules are significantly a better representation of the patterns compared to the 7 rules in Ex1. Using a similar approach to that used in of Ex1, R1 in Ex2 builds two nodes I2 and I1 in the first pass i.e. (I2:1, I1:1). R92 is a final process which constructs the next five nodes with various frequencies as follows I2:8, I1:21, I3:11, I4:4, I5:1of c1 as shown in Figure 17.

This tree does include a reasonable number of patterns which may be suitably comparable to the CBR query. In addition, the rules are classified under the one class to which they belong. However, finding a mutual pattern between a CBR query and a FP-tree is unachievable because the nodes do not contain a value as explained in our proposed algorithm. Ex1 and Ex2 led the researcher to a deductive approach to produce a FP-tree containing a value of

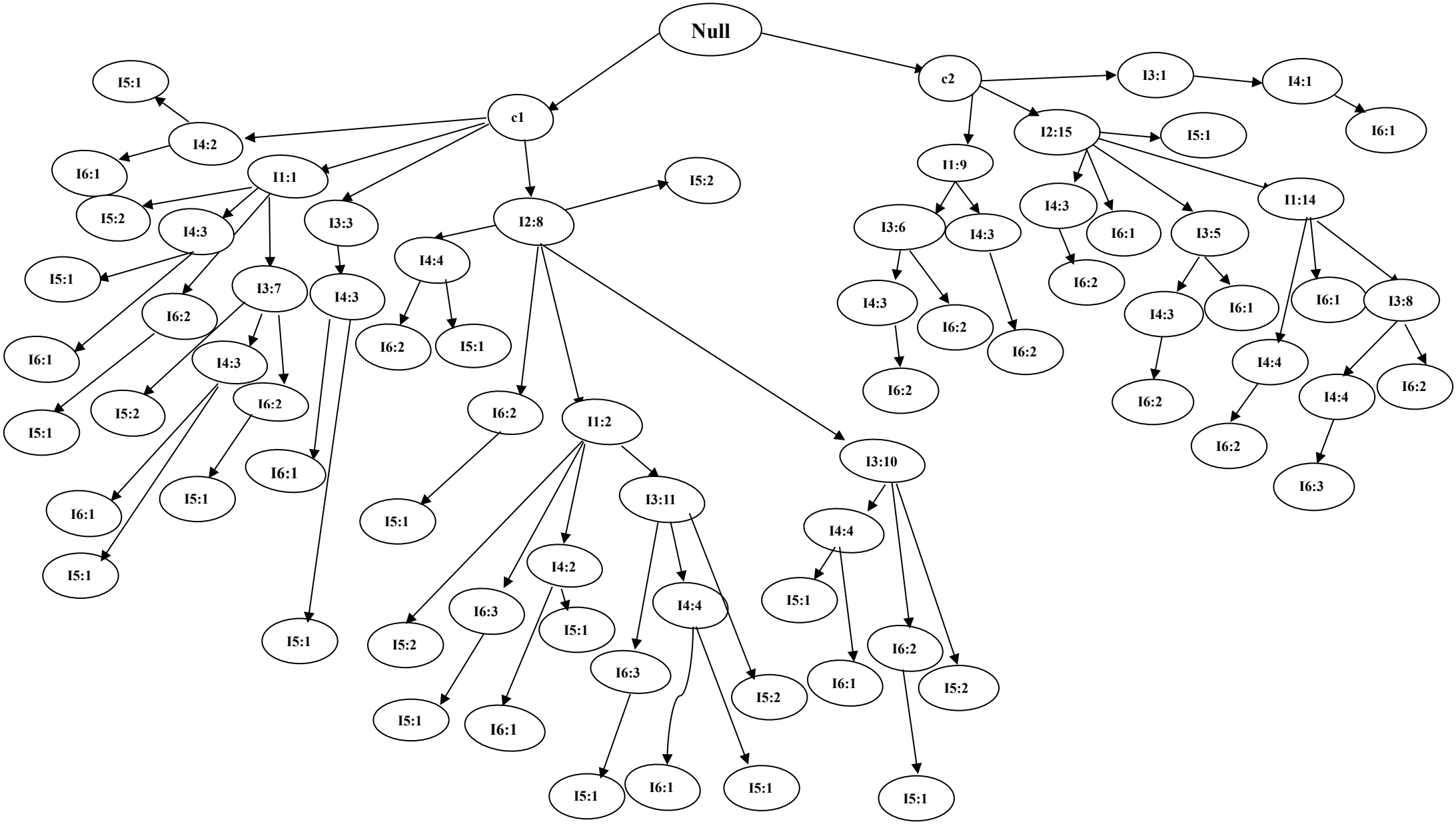


Figure 17 Tree of CAR_Rules without values

each node based on CAR_Rules so as to be practically comparable.

4.4.3. **Experiment 3 CAR_Rules Tree with Nodes' Values**

In the third experiment Ex3 CAR_Rules are generated from the dataset D1. The focus of this experiment is to produce a FP-tree which considers the value of each item where each item represents a node. These nodes form patterns of rules of a FP-tree to be comparable to another pattern. This allowed the comparison between the FP-tree and CBR query to be achieved whereas in Ex2 it was not. As shown in Figure 18 and Figure 19, the 92 rules constructed a descriptive pattern of a tree where values appear in each node. This tree includes an adequate number of patterns which are suitably comparable to another pattern. Furthermore, the rules are classified under the one class to which they belong. Therefore, finding a mutual pattern between a CBR query and a FP-tree is achievable because the nodes contain a value in the proposed algorithm. These values are the key which makes such a tree comparable to the "wrong" SBR answer as shown in Figure 13.

In order to validate our proposition, a pre-determined case was used to check the retrieval phase performance on the same dataset D1. CBR retrieved five cases (Case71: yes, Case72: yes, Case76: no, Case77: no and Case79: yes) with the same similarity (0.912) when a New Case is applied to the SBR phase. This means that the CBR system registered an uncertain answer. The ambiguity can mislead the decision maker as all retrieved cases have the same accuracy. An expert may be required to resolve this dilemma. FP-CAR acts intelligently to disambiguate this issue. It compares a New Case with the FP-CAR tree, if a mutual pattern is found a solution will be returned to the CBR system to support the correct case. Where a partial match is found, it invokes a P-tree to form a correct pattern using the longest length voting of a specific class.

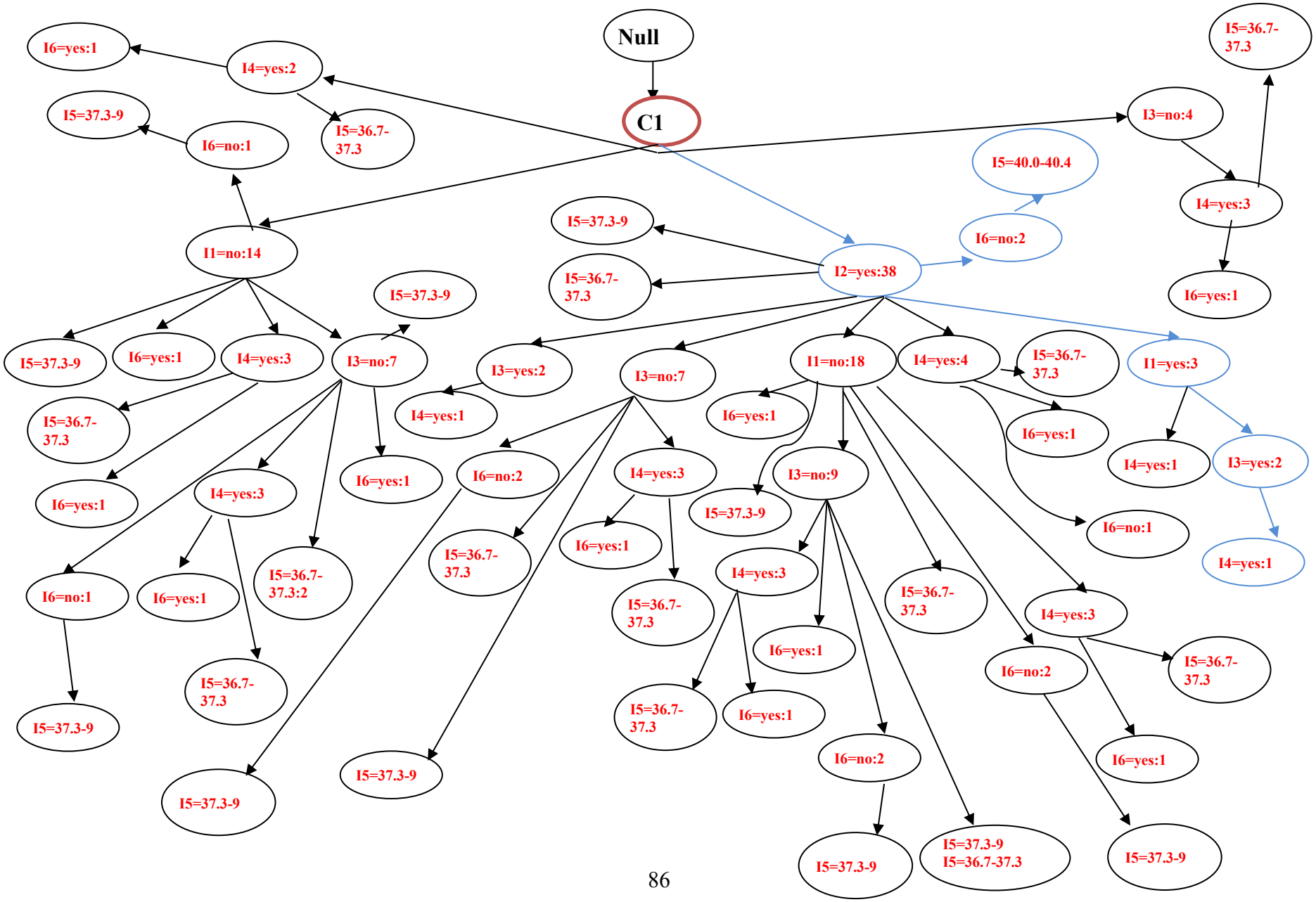


Figure 18 c1_Rules Tree with Values

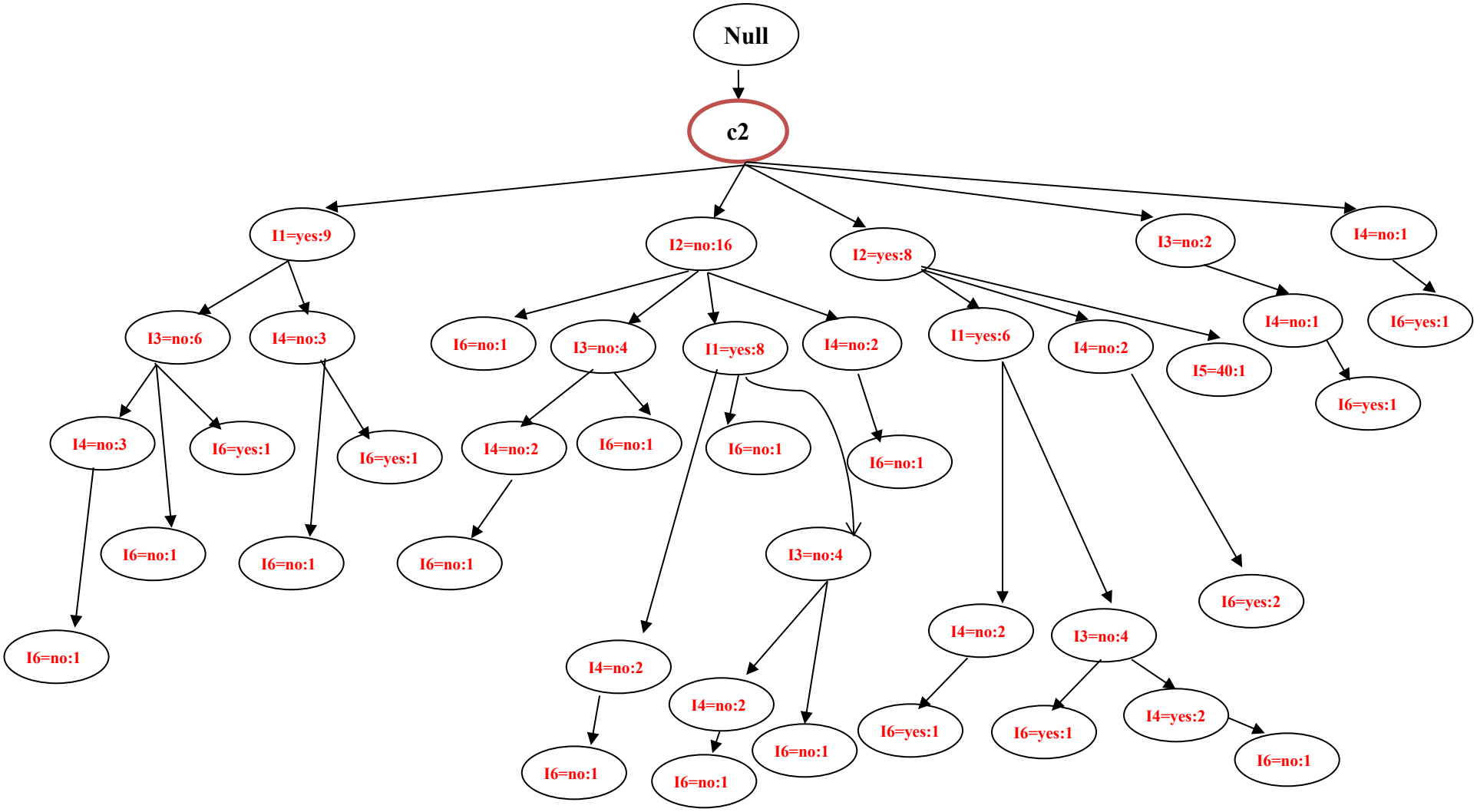


Figure 19 c2_Rules Tree with Values

Experiments Ex1, Ex 2 and Ex3 were a heuristic path that led to conducting more experiments by integrating CARs into CBR [151].

4.5. Further Analysis and Experiments of Acute Inflammation Urinary Bladder Dataset

This section presents the applied cases of the acute inflammation urinary bladder dataset to both CBR tools and CBRAR strategy. The section will include the necessary tables, figures and charts to illustrate the findings in terms of accuracy and error rates as follows.

4.5.1. Experiments 4, 5, 6 and 7: Cases 73, 76, 85 and 88 Using CBRAR Strategy

In Experiments 4, 5, 6 and 7 cases 73, 76, 85 and 85 registered an ambiguity as all retrieved cases have the same 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 8, Table 9, Table 10 and Table 11. Vertically, the first column refers to the new case *Q* followed by the cases retrieved by the CBR tools. For example, New-Case73 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 addition, we use the symbols TP and FP to denote True Positive and False Positive. The assumption is made to indicate the four probabilities on the confusion matrix. Table 8, Table 9, Table 10 and Table 11 show that, for each new case applied to CBR, 5 different cases are retrieved by Jcolibri with the same similarity ratio

of 0.912. In addition, more cases are retrieved by FreeCBR with the same similarity ratio of 59.1751% for cases (73, 76), and (55.278, 55.276) ratio for cases (85, 88).

In the fourth experiment, the NewCase73 applied to CBR, Jcolibri retrieved 3 TP and 2 FP cases with the same similarity ratio, and this achieved 60% accuracy, whereas FreeCBR retrieved 9 TP and 2 FP, equal to 81% accuracy.

In the fifth experiment, when NewCase76 is applied to the CBR, Jcolibri retrieved 4 TN and 1 FN case with the same similarity ratio, and this is equal to 80% accuracy, whereas FreeCBR retrieved 6 TN and 1 FN, and this is equal to 86% accuracy.

In the sixth experiment, when NewCase85 is applied to the CBR, Jcolibri retrieved 4 TP and 1 FP case with the same similarity ratio, achieving 80% accuracy, whereas FreeCBR retrieved 7 TP and 1 FP, achieving 87% accuracy.

CBRAR retrieved 1 TP case of experiments 4, 5 and 6 from the new model which is the correct case hence outperforming the performance of the CBR used tools. This is shown in Figure 20, Figure 21 and Figure 22. Our CBRAR strategy demonstrates advantages over both Jcolibri and FreeCBR by retrieving the correct case with 100% accuracy and no confusion. Cases (73, 76 and 85) in Table 8, Table 9 and Table 10 can be reworked in Figure 10 to prove that CBRAR identifies the correct case using a frequent classed tree.

Table 8 Case 73 Acute Inflammation Dataset - CBR Results

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**	**100**
Case71	40.0	yes	yes	yes	yes	yes	yes	TP	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	
Average								**60%**	**81%**	**100%**

Table 9 Case 76 Acute Inflammation Dataset - CBR Results

Cases	**Attributes**							**Accuracy**		
	F	**A**	**B**	**C**	**D**	**E**	**Class**	**Jcolibri**	**FreeCBR**	**CBRAR**
NewCase76	**40.0**	**yes**	**yes**	**no**	**yes**	**no**	**no**	**0.912**	**59.1751**	**100**
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	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	
Average								**80%**	**86%**	**100%**

Table 10 Case 85 Acute Inflammation Dataset - CBR Results

Cases	**Attributes**							**Accuracy**		
	F	**A**	**B**	**C**	**D**	**E**	**Class**	**Jcolibri**	**FreeCBR**	**CBRAR**
NewCase85	**40.4**	**yes**	**yes**	**yes**	**yes**	**no**	**yes**	**0.912**	**55.278**	**100**
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	TP
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	
Average								**80%**	**88%**	**100%**

The bar charts in Figure 20, Figure 21 and Figure 22 illustrate the error rate and accuracy of

Jcolibri, FreeCBR and CBRAR. From the charts, it is clear that in Cases 73, 76 and 85 CBRAR registered 0 error rate, which is the lowest among the rates (40%, 19%), (20%, 14%) and (20%, 12%) when compared to Jcolibri and FreeCBR.

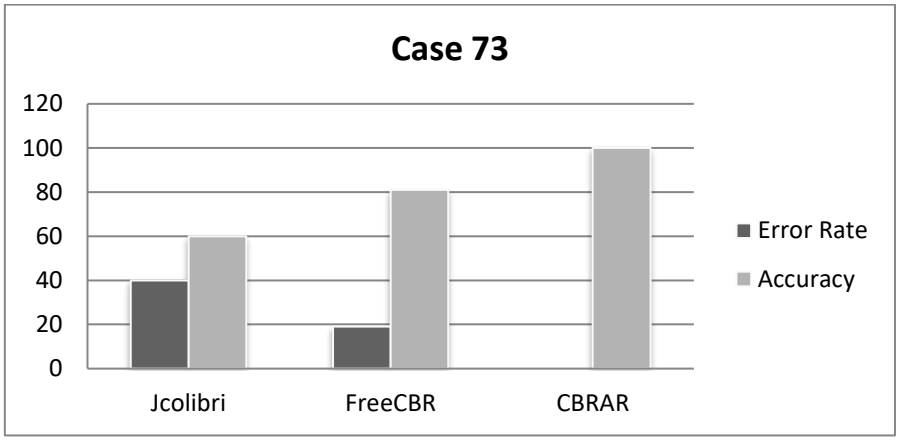


Figure 20 Case 73 Error and Accuracy Rate

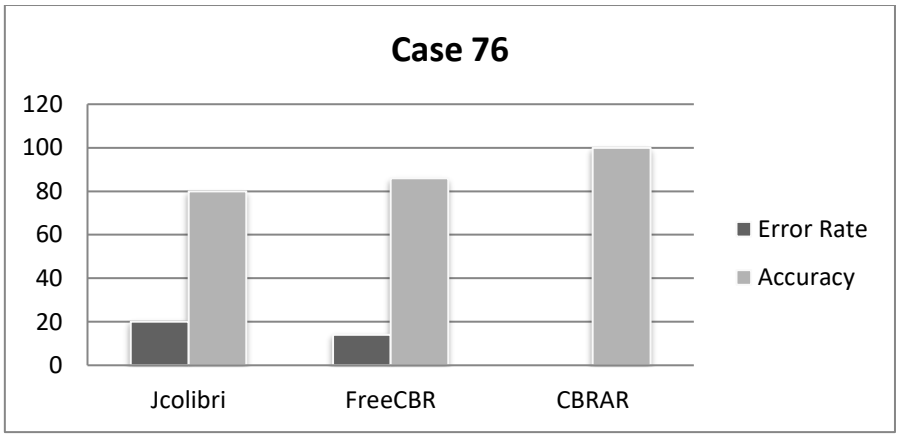


Figure 21 Case 76 Error and Accuracy Rate

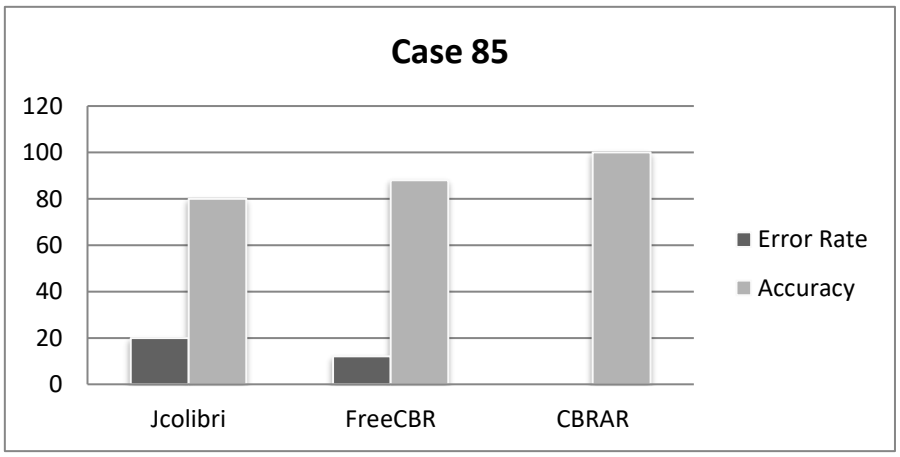


Figure 22 Case 85 Error and Accuracy Rate

CBRAR shows a better performance in overall error rate and correctly resolved the target case.

However, in the seventh experiment, when NewCase88 is applied to the CBR, Jcolibri retrieved 3 TN and 2 FN case with the same similarity ratio, and this is equal to 60% accuracy, whereas FreeCBR retrieved 7 TN and 2 FN, giving 70% accuracy.

The accuracy of Jcolibri and FreeCBR was better compared to CBRAR. Jcolibri retrieved 2 FN and also FreeCBR retrieved 2 FN cases which would not be considered as an advantage in the total confusion matrix. CBRAR did not retrieve any case from new the model giving 0% accuracy because the target case did not match any pattern in the FP-CAR so no soltution are produced by CBRAR.

Table 11 Case 88 Acute Inflammation Dataset - CBR Results

Cases	Attributes							Accuracy		
	F	A	B	C	D	E	Class	Jcolibri	FreeCBR	CBRAR
NewCase88	**40.4**	**yes**	**yes**	**no**	**yes**	**no**	**no**	**0.912**	**55.276**	**100**
Case76	40.0	yes	yes	no	yes	no	no	TN	TN	FN
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								**60%**	**70%**	**0%**

The bar chart in Figure 23 illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, Case88, CBRAR did not returned any solution and registered 100% error rate, which is the highest among the rates (40%, 30%) when compared to Jcolibri and FreeCBR.

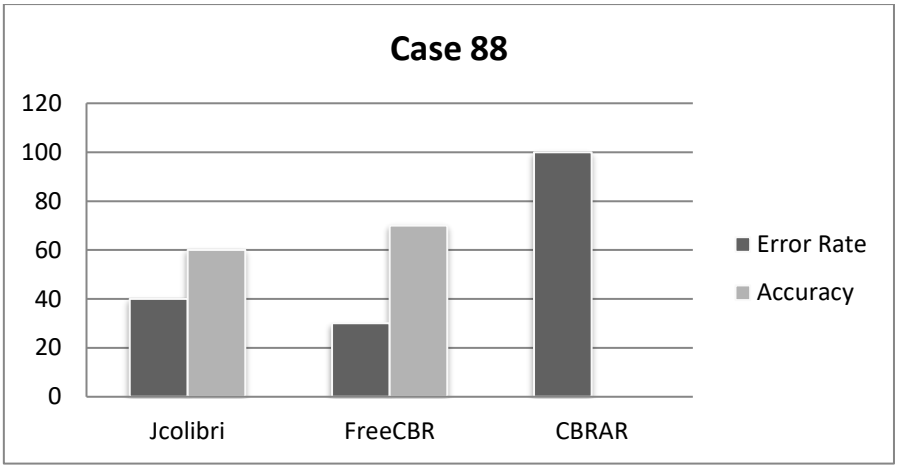


Figure 23 Case 88 Error and Accuracy Rate

4.5.2. **Discussion of Acute Inflammation Urinary Bladder Dataset Results**

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 an advantage 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 8, Table 9 and Table 10 can be reworked in Figure 10 to prove that CBRAR identifies a correct case using a frequent classed tree.

The line chart in Figure 24 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 addition, in case 88 the CBRAR registered the lowest accuracy because in the proposed CBRAR, the FP-CAR algorithm did not return any target case as a solution. As explained in chapter 3, if a full match is found, CBRAR returns cases as a potential solution. If a partial match is found the FP-CAR invokes the P-tree to form a full target case within FP-CAR tree or union set procedures is used to return a solution. No full or initial match of the target case88 are found in the FP-CAR. Therefore, 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.

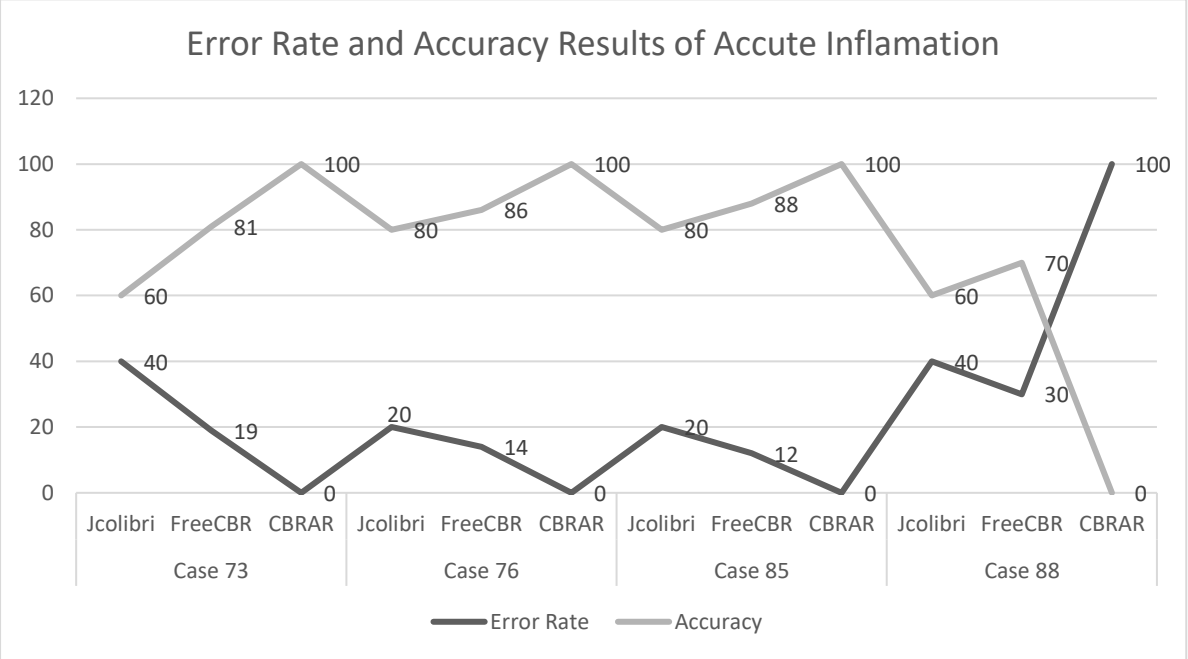


Figure 24 Error Rate and Accuracy Results Assembled of Acute Inflammation Dataset

4.6. Results of the Space Shuttle Dataset

This section displays the retrieved cases of the shuttle space dataset CBR tools which registered ambiguous answers. The section will also contain tables, figures and charts to investigate the findings in terms of accuracy and error rates.

4.6.1. Experiments 8, 9, 10, 11 and 12: Cases 10, 11, 12, 15 and 8 Using CBRAR Strategy

In Experiments 8, 9, 10, 11 and 12, cases 10, 11, 12, 15 and 8 registered as an ambiguous cases as all retrieved cases have the same percentage of similarity with different labels i.e. (1, 2). When using FreeCBR, more potential cases were identified in addition to those found by Jcolibri. The results are shown in Table 12, Table 13, Table 14 ,Table 15 and Table 16, vertically, the first column refers to the new case *Q* followed by the cases retrieved by the CBR tools. For example, NewCase10 followed by cases (11, 12, 14, 1 and 2, for Jcolibri) and cases (11, 12, 14 until 7 for FreeCBR). In Jcolibri, cases 1and 2 have not been counted in the confusion matrix because they returned cases but with a lower similarity of 0.816 when compared to cases 11, 12 and 14 that registered higher similarity of 0.92. A similar approach is used in FreeCBR where cases 1 to 7 have returned a lower similarity 42.264 compared to cases 11, 12 and 14 that registered higher similarity of 59.175.

The "Attributes" columns start with a Magnitude attribute *A* followed by 5 additional attributes *B, C, D* and *F.* The class label column determines the control used for landing non-auto and automatic landing (1 and 2). The "Accuracy" columns show the comparison between Jcolibri, FreeCBR and CBRAR.

In the eighth experiment, a NewCase10 applied to the CBR, Jcolibri retrieved 2 TP and 1 FP cases with the same similarity ratio, and this achieved 66% accuracy, and FreeCBR retrieved 2 TP and 1 FP, equal to 66% accuracy.

In the ninth experiment, when a NewCase11 is applied to the CBR, Jcolibri retrieved 2 TP and 1 FP cases with the same similarity ratio, giving 66% accuracy, whereas FreeCBR retrieved 2 TP and 1 FP, giving 66% accuracy.

In the tenth experiment, when a NewCase12 applied to the CBR, Jcolibri retrieved 1 TP and 4 FP cases with the same similarity ratio, registering 20% of accuracy, wheras FreeCBR retrieved 4 TP and 4 FP, giving to 50% accuracy.

In the eleventh experiment, the NewCase12 applied to the CBR, Jcolibri and FreeCBR retrieved 4 TP and 1 FP cases with the same similarity ratio, and this achieved 80% accuracy.

CBRAR retrieved 1 TP case of experiments 8, 9, 10 and 11 from the new model which is the correct case hence outperforming the performance of the CBR used tools. This is illustrated in Figure 26, Figure 27, Figure 28 and Figure 29. Our CBRAR strategy demonstrates advantages over both Jcolibri and FreeCBR by retrieving the correct case with 100% accuracy and no confusion. Cases 10, 11, 12 and 15 in Table 12, Table 13, Table 14 and Table 15 can be reworked in Figure 25 to prove that CBRAR identifies the correct case using a frequent classed tree.

Table 12 Case 10 Space Shuttle Dataset - CBR Results

Cases	**Attributes**							**Accuracy**		
	A	**B**	**C**	**D**	**E**	**F**	**Class**	**Jcolibri**	**FreeCBR**	**CBRAR**
NewCase10	**1**	**3**	**1**	**1**	**1**	**1**	**2**	**0.912**	**59.175**	**100**
Case11	2	3	1	1	1	1	2	TP	TP	TP
Case12	1	3	1	1	2	1	2	TP	TP	
Case14	3	3	1	1	1	1	1	FP	FP	
Case1	1	3	1	1	2	2	2	0.816	42.264	
Case2	1	3	1	2	1	2	1	0.816	42.264	
Case3	1	2	1	1	1	2	1		42.264	
Case4	1	1	1	1	1	2	1		42.264	
Case5	1	3	2	1	1	2	1		42.264	
Case7	1	4	1	1	2	1	2		42.264	
Average								**66%**	**66%**	**100%**

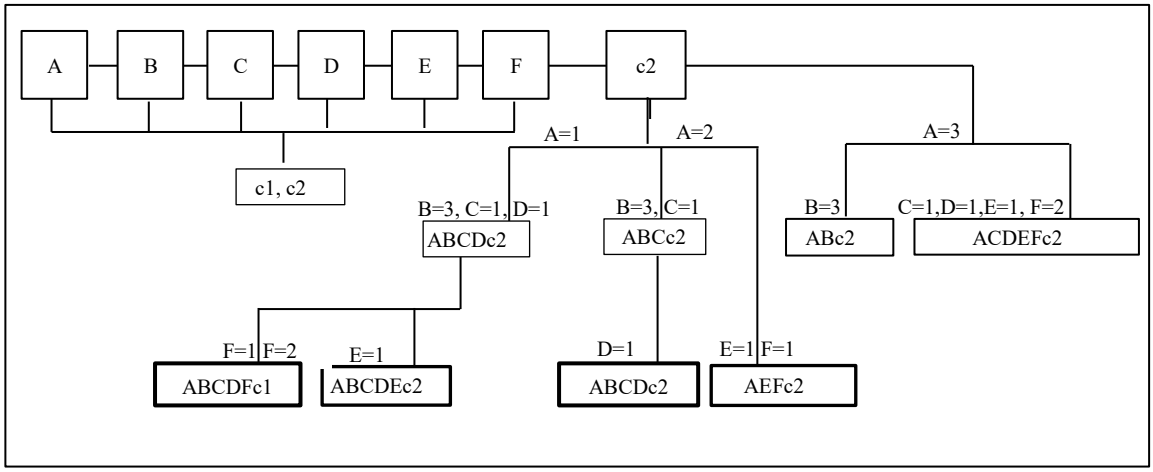


Figure 25 FP-CAR Algorithm Tree – Space Shuttle Dataset

Table 13 Case 11 Shuttle Dataset - CBR Results

Cases	Attributes							Accuracy		
	A	B	C	D	E	F	Class	Jcolibri	FreeCBR	CBRAR
NewCase11	**2**	**3**	**1**	**1**	**1**	**1**	**2**	**0.912**	**59.175**	**100**
Case10	1	3	1	1	1	1	2	TP	TP	TP
Case13	2	3	1	1	1	2	2	TP	TP	
Case14	3	3	1	1	1	1	1	FP	FP	
Case8	2	4	1	1	1	2	2	0.816	42.264	
Case12	1	3	1	1	1	2	2	0.816	42.264	
Case15	3	3	1	1	1	2	2		42.264	
Average								**66%**	**66%**	**100%**

Table 14 Case 12 Shuttle Dataset - CBR Results

Cases	Attributes							Accuracy		
	A	B	C	D	E	F	Class	Jcolibri	FreeCBR	CBRAR
NewCase12	**1**	**3**	**1**	**1**	**1**	**2**	**2**	**0.912**	**59.175**	**100**
Case1	1	3	1	1	2	2	2	TP	TP	TP
Case2	1	3	1	2	1	2	1	FP	FP	
Case3	1	2	1	1	1	2	1	FP	FP	
Case4	1	1	1	1	1	2	1	FP	FP	
Case5	1	3	2	1	1	2	1	FP	FP	

Case10	1	3	1	1	1	1	2		TP	
Case13	2	3	1	1	1	1	2		TP	
Case15	3	3	1	1	1	2	2		TP	
Average								**20%**	**50%**	**100%**

Table 15 Case 15- Shuttle Dataset - CBR Results

Cases	**Attributes**							**Accuracy**		
	A	**B**	**C**	**D**	**E**	**F**	**Class**	**Jcolibri**	**FreeCBR**	**CBRAR**
NewCase15	3	3	1	1	1	2	**2**	**0.912**	**59.175**	**100**
Case9	3	4	1	1	1	2	2	TP	TP	TP
Case12	1	3	1	1	1	2	2	TP	TP	
Case13	2	3	1	1	1	2	2	TP	TP	
Case14	3	3	1	1	1	1	1	FP	FP	
Case1	1	3	1	1	2	2	2	0.816	42.264	
Case2	1	3	1	2	1	2	1		42.264	
Case3	1	2	1	1	1	2	1		42.264	
Case4	1	1	1	1	1	2	1		42.264	
Average								**80%**	**80%**	**100%**

The bar charts in Figure 26, Figure 27, Figure 28 and Figure 29 illustrate the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the charts, it is clear that in Cases (10, 11. 12 and 15) CBRAR registered 0 error rate, which is the lowest among the rates (34%, 34%), (34%, 34%), (80%, 50%) and (20%, 20%) when compared to Jcolibri and FreeCBR.

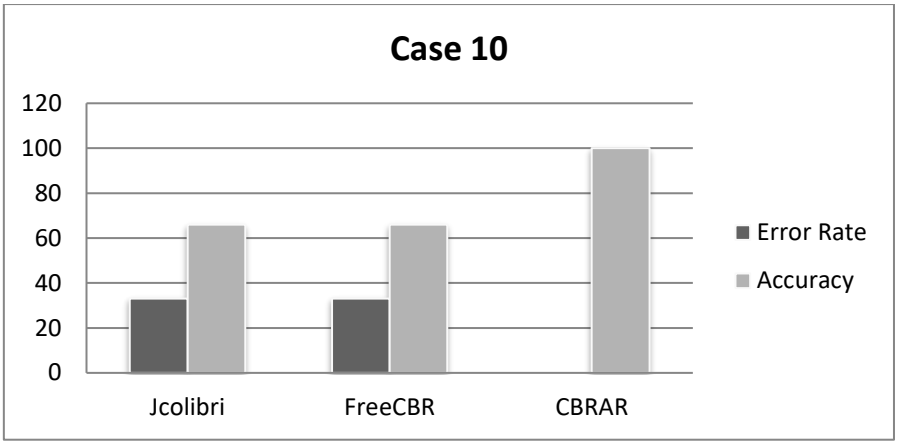


Figure 26 Case 10 Error and Accuracy Rate

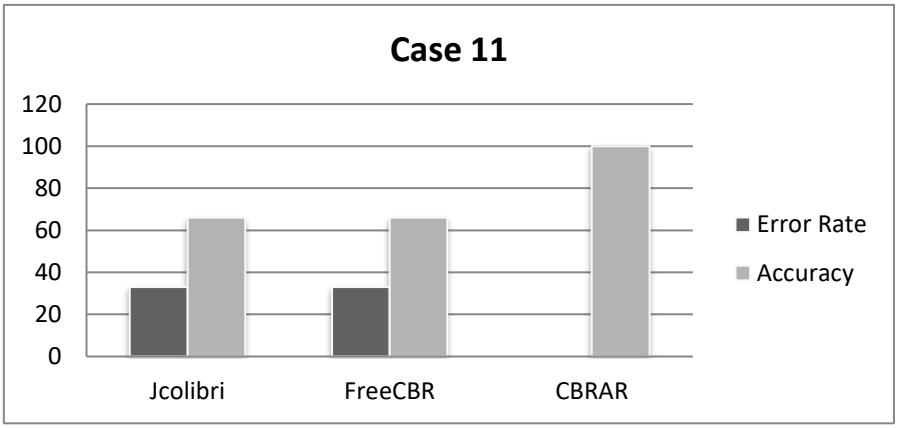


Figure 27 Case 11 Error and Accuracy Rate

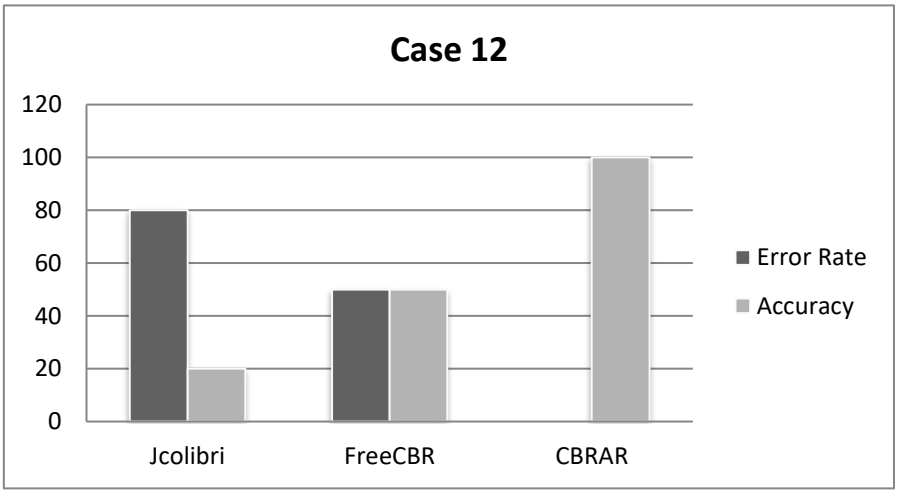


Figure 28 Case 12 Error and Accuracy Rate

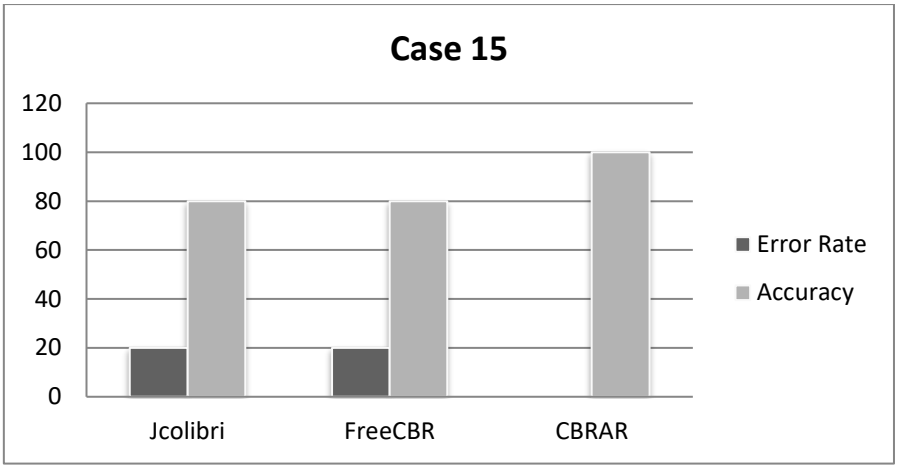


Figure 29 Case 15 Error and Accuracy Rate

CBRAR shows a better performance in overall error rate and also correctly resolved the target case.

However, in the twelfth experiment, a NewCase8 applied to the CBR, Jcolibri and FreeCBR retrieved 2 TP and 1 FP cases with the same similarity ratio, giving 66% accuracy, whereas CBRAR did not retrieve any case from new model, achieving 0%. The accuracy of Jcolibri and FreeCBR was better compared to CBRAR. Jcolibri and FreeCBR retrieved 2 TP and 1 FP with same similarity of 0.912, 56.175 which would not be considered as an advantage in the total confusion matrix. Cases 3 and 4 have been ignored because they recorded a lower similarity ratio of 8.16 for Jcolibri. In FreeCBR, cases 3 to 15 have also been ignored because they recorded a lower similarity ratio of 42.264 compared to those cases which registered 59.175 i.e. 6, 9 and 13.

Table 16 Case 8- Shuttle Dataset - CBR Results

Cases	**Attributes**							**Accuracy**		
	A	**B**	**C**	**D**	**E**	**F**	**Class**	**Jcolibri**	**FreeCBR**	**CBRAR**
NewCase8	2	4	1	1	1	2	**2**	**0.912**	**59.175**	**100**
Case6	4	4	1	1	1	2	1	FP	FP	FP
Case9	3	4	1	1	1	2	2	TP	TP	
Case13	2	3	1	1	1	2	2	TP	TP	

Case3	1	2	1	1	1	2	1	0.816	42.264	
Case4	1	1	1	1	1	2	1	0.816	42.264	
Case11	2	3	1	1	1	1	2		42.264	
Case12	1	3	1	1	1	2	2		42.264	
Case15	2	3	1	1	1	2	2		42.264	
Average								**66%**	**66%**	**0%**

The bar chart in Figure 30 illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, Case8, CBRAR registered 100% error rate, which is the highest among the rates (34%) when compared to Jcolibri and FreeCBR. In case 8, no potential pattern is found in the FP-CAR tree therefore, CBRAR did not retrieve any similar case and no solution is produced by the proposed strategy.

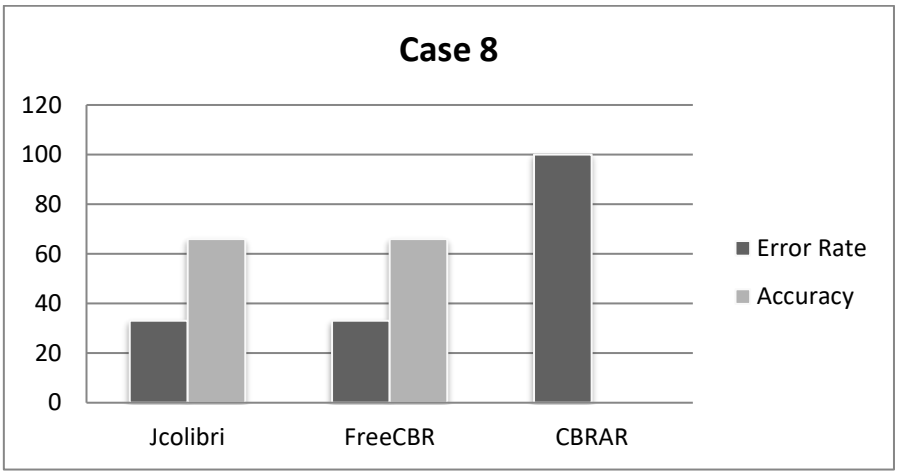


Figure 30 Case 8 Error and Accuracy Rate

4.6.2. **Discussion of Space Shuttle Dataset Results**

The results show that 10 out of the 18 Jcolibri retrieved cases are classified as TP giving 55% accuracy. By comparison, 13 of the 21 cases retrieved by FreeCBR are classified as TP giving 61% accuracy. However, both Jcolibri and FreeCBR deliver "confusing" results. Our CBRAR strategy demonstrates an advantage over both Jcolibri and FreeCBR by resolving 4 out of 5 cases with 80% accuracy and no confusion. The accuracy of CBRAR was better compared to Jcolibri and FreeCBR. CBRAR resolved the ambiguity of the FP cases without confusion. One

of the most noteworthy results in this research is the greatly improved accuracy rate obtained in [151], the union of two rules was not used and the evolution on the acute inflammation dataset returned 3 out of 4 cases with 75% accuracy. With the new approach, it was possible to improve the old system by using the union of two rules and improved the FP-CAR algorithm by returning 4 out of 5 cases of space shuttle dataset instead of 2 out of 5 cases on the space dataset (cases 10 and 12) increasing the accuracy from 40% to 80%. Cases 10, 11, 12 and 15 in Table 12, Table 13, Table 14 and Table 15 can be reworked in Figure 25 to prove that CBRAR identifies a correct case using a frequent classed tree.

The line chart in Figure 31 illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, it is clear that in Case10, CBRAR registered 0 error rate, which is the lowest among the rates (66) when compared to Jcolibri and FreeCBR. The results also show that the error rate of CBRAR is the lowest on Case11, Case12 and Case15 thus giving the highest accuracy, when compared to the other CBR tools used. In Case8, it noticeable that the (34) % error rate of Jcolibri and FreeCBR was considerably lower than CBRAR.

However, whilst CBRAR did not resolve Case8 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. The CBRAR did not resolve case 8 because no full or partial match of the target case is found in the FP-CAR, i.e. no similar nodes of FP-CAR are matched case 8. Therefore, invoking the P-tree to find a full target case within FP-CAR tree or using union set procedures did not produce a solution. 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.

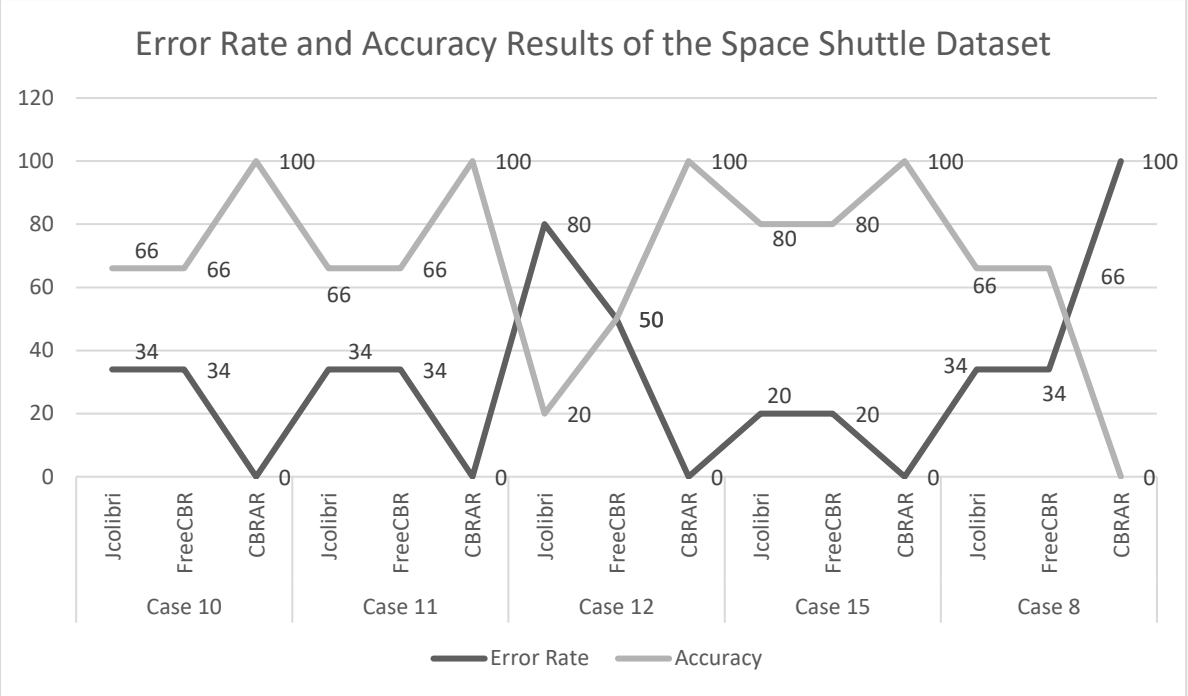


Figure 31 Error Rate and Accuracy Results Assembled of the Space Shuttle Dataset

4.7. Results of the Balloon Dataset

This section displays the retrieved cases of the Balloon dataset CBR tools which recorded ambiguous answers. The section will also contain tables, figures and charts to investigate the findings in terms of accuracy and error rates.

4.7.1. Experiments 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 and 24: Cases (1, 2), 3, 4, 6, 8, 9, (11, 12), 13, 14, (16 ,17), 18 and 19 Using CBRAR Strategy

In Experiments 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 and 24, cases (1, 2), 3, 4, 6, 8, 9, (11, 12), 13, 14, (16 ,17), 18 and 19 were identified as the ambiguous cases as indicated in the results achieved in the tables of each conducted experiments. For example, Table 17 indicates the cases that are identified as similar for both Jcolibri and FreeCBR. The "Attributes" columns start with A followed by 3 additional attributes B, C and D. In addition, the class label column determines the True and False of a balloon inflating (c1 and c2). The "Accuracy" columns show

the comparison between Jcolibri, FreeCBR and CBRAR. Table 17 also shows that, for each new case applied to CBR, 5 different cases are retrieved by Jcolibri with same similarity ratio of 0.866 %, and 6 cases are retrieved by FreeCBR with similarity ratio of 55.0%.

In the thirteenth experiment, NewCases1, 2 applied to the CBR, Jcolibri retrieved 2 TP and 3 FP cases with the same similarity ratio, giving 40% of accuracy, whereas FreeCBR retrieved 2 TP and 4 FP, giving 33% accuracy.

In the fourteenth experiment, NewCase3 applied to the CBR, Jcolibri retrieved 2 TN and 3 FN cases with the same similarity ratio, giving 40% accuracy, and FreeCBR retrieved 2 TN and 3 FN, giving 40% accuracy.

In the fifteenth experiment, NewCase4 applied to the CBR, Jcolibri retrieved 2 TN and 3 FN cases with the same similarity ratio, giving 40% accuracy, Similarly FreeCBR retrieved 2 TN and 3 FN, giving 40% accuracy.

In the sixteenth experiment, NewCase6 applied to the CBR, Jcolibri retrieved 3 TP and 2 FP cases with the same similarity ratio, giving 60% accuracy, whereas FreeCBR retrieved 4 TP and 2 FP, equal to 66% accuracy.

In the seventeenth experiment, NewCase8 is applied to the CBR. Jcolibri retrieved 3 TN and 2 FN cases with the same similarity ratio, giving 40% accuracy, and FreeCBR retrieved 3 TN and 2 FN, giving 40% accuracy.

In the eighteenth experiment, NewCase8 is applied to the CBR. Jcolibri retrieved 3 TN and 2 FN cases with the same similarity ratio, providing 40% accuracy, and FreeCBR retrieved 3 TN and 2 FN, providing 40% accuracy.

In the nineteenth experiment, NewCase11 and 12 are applied to the CBR, Jcolibri retrieved 3

TN and 2 FN cases with the same similarity ratio, giving 60% accuracy, whereas FreeCBR retrieved 4 TP and 2 FP, giving 66% accuracy.

In the twentieth experiment, NewCase13 applied to the CBR, Jcolibri and FreeCBR retrieved 3 TN and 2 FN cases with the same similarity ratio, and this achieved 60% accuracy.

In the twenty first experiment, NewCase14 applied to the CBR, Jcolibri and FreeCBR retrieved 3 TN and 2 FN cases with the same similarity ratio, giving 60% accuracy.

In the twenty second experiment, NewCases16 and 17 applied to the CBR, Jcolibri retrieved 4 TP and 1 FP cases with the same similarity ratio, giving 80% accuracy, and FreeCBR retrieved 4 TP and 2 FP, giving 66% accuracy.

In the twenty third experiment, NewCase18 applied to the CBR, Jcolibri retrieved 3 TN and 2 FN cases with the same similarity ratio, and this achieved 60% accuracy, and FreeCBR retrieved 3 TN and 2 FN, giving 60% accuracy.

In the twenty fourth experiment, NewCase19 applied to the CBR, Jcolibri retrieved 3 TN and 2 FN cases with the same similarity ratio, giving 60% accuracy, and FreeCBR retrieved 3 TN and 2 FN, giving 60% accuracy.

CBRAR retrieved 1 TP case of experiments 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 and 24 from new strategy which is the correct case hence achieving 100% accuracy and outperforming the performance of the CBR tools used. This is illustrated in Figure 33, Figure 35, Figure 36, Figure 37, Figure 38, Figure 39, Figure 40, Figure 41, Figure 42, Figure 43, Figure 44 and Figure 33.

Table 17 Cases 1, 2- Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase1,2	**STRETCH**	**ADULT**	**YELLOW**	**SMALL**	**c1**	**0.866**	**50.0**	**100**
Case3	STRETCH	CHILD	YELLOW	SMALL	c2	TP	TP	TP
Case4	DIP	ADULT	YELLOW	SMALL	c2	TP	TP	
Case6	STRETCH	ADULT	YELLOW	LARGE	c1	FP	FP	
Case7	STRETCH	ADULT	YELLOW	LARGE	c1	FP	FP	
Case11	STRETCH	ADULT	PURPLE	SMALL	c1	FP	FP	
Case12	STRETCH	ADULT	PURPLE	SMALL	c1		FP	
Average						**40%**	**33%**	**100%**

Table 18 Case 3 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase3	**STRETCH**	**CHILD**	**YELLOW**	**SMALL**	**c2**	**0.866**	**50.0**	**100**
Case1	STRETCH	ADULT	YELLOW	SMALL	c1	TN	TN	TN
Case2	STRETCH	ADULT	YELLOW	SMALL	c1	TN	TN	
Case5	DIP	CHILD	YELLOW	SMALL	c2	FN	FN	
Case8	STRETCH	CHILD	YELLOW	LARGE	c2	FN	FN	
Case13	STRETCH	CHILD	PURPLE	SMALL	c2	FN	FN	
Average						**40%**	**40%**	**100%**

Table 19 Case 4 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase4	**DIP**	**ADULT**	**YELLOW**	**SMALL**	**c2**	**0.866**	**50.0**	**100**
Case1	STRETCH	ADULT	YELLOW	SMALL	c1	TN	TN	TN
Case2	STRETCH	ADULT	YELLOW	SMALL	c1	TN	TN	
Case5	DIP	CHILD	YELLOW	SMALL	c2	FN	FN	
Case9	DIP	ADULT	YELLOW	LARGE	c2	FN	FN	
Case14	DIP	ADULT	PURPLE	SMALL	c2	FN	FN	
Average						**40%**	**40%**	**100%**

Table 20 Case 6 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase6	STRETCH	ADULT	YELLOW	LARGE	**c1**	**0.866**	**50.0**	**100**
Case1	STRETCH	ADULT	YELLOW	SMALL	c1	TP	TP	TP
Case2	STRETCH	ADULT	YELLOW	SMALL	c1	TP	TP	
Case8	STRETCH	CHILD	YELLOW	LARGE	c2	FP	FP	
Case9	DIP	ADULT	YELLOW	LARGE	c2	FP	FP	
Case16	STRETCH	ADULT	PURPLE	LARGE	c1	TP	TP	
Case17	STRETCH	ADULT	PURPLE	LARGE	c1		TP	
Average						**60%**	**66%**	**100%**

Table 21 Case 8 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase8	**STRETCH**	**CHILD**	**YELLOW**	**LARGE**	**c2**	**0.866**	**50.0**	**100**
Case3	STRETCH	CHILD	YELLOW	SMALL	c2	TN	TN	TN
Case6	STRETCH	ADULT	YELLOW	LARGE	c1	FN	FN	
Case7	STRETCH	ADULT	YELLOW	LARGE	c1	FN	FN	
Case10	DIP	CHILD	YELLOW	LARGE	c2	TN	TN	
Case18	STRETCH	CHILD	PURPLE	LARGE	c2	TN	TN	
Average						**60%**	**60%**	**100%**

Table 22 Case 9 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase9	**DIP**	**ADULT**	**YELLOW**	**LARGE**	**c2**	**0.866**	**50.0**	**100**
Case4	DIP	ADULT	YELLOW	SMALL	c2	TN	TN	TN
Case6	STRETCH	ADULT	YELLOW	LARGE	c1	FN	FN	
Case7	STRETCH	ADULT	YELLOW	LARGE	c1	FN	FN	
Case10	DIP	CHILD	YELLOW	LARGE	c2	TN	TN	
Case19	DIP	ADULT	PURPLE	LARGE	c2	TN	TN	
Average						**60%**	**60%**	**100%**

Table 23 Cases 11, 12 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase11,12	**STRETCH**	**ADULT**	**PURPLE**	**SMALL**	**c1**	**0.866**	**50.0**	**100**
Case1	STRETCH	ADULT	YELLOW	SMALL	c1	TP	TP	TP
Case2	STRETCH	ADULT	YELLOW	SMALL	c1	TP	TP	
Case13	STRETCH	CHILD	PURPLE	SMALL	c2	FP	FP	
Case14	DIP	ADULT	PURPLE	SMALL	c2	FP	FP	
Case16	STRETCH	ADULT	PURPLE	LARGE	c1	TP	TP	
Case17	STRETCH	ADULT	PURPLE	LARGE	c1		TP	
Average						**60%**	**66%**	**100%**

Table 24 Case 13 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase13	**STRETCH**	**CHILD**	**PURPLE**	**SMALL**	**c2**	**0.866**	**50.0**	**100**
Case3	STRETCH	CHILD	YELLOW	SMALL	c2	TN	TN	TN
Case11	STRETCH	ADULT	PURPLE	SMALL	c1	FN	FN	
Case12	STRETCH	ADULT	PURPLE	SMALL	c1	FN	FN	
Case15	DIP	CHILD	PURPLE	SMALL	c2	TN	TN	
Case18	STRETCH	CHILD	PURPLE	LARGE	c2	TN	TN	
Average						**60%**	**60%**	**100%**

Table 25 Case 14 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase14	**DIP**	**ADULT**	**PURPLE**	**SMALL**	**c2**	**0.866**	**50.0**	**100**
Case4	DIP	ADULT	YELLOW	SMALL	c2	TN	TN	TN
Case11	STRETCH	ADULT	PURPLE	SMALL	c1	FN	FN	
Case12	STRETCH	ADULT	PURPLE	SMALL	c1	FN	FN	
Case15	DIP	CHILD	PURPLE	SMALL	c2	TN	TN	
Case19	DIP	ADULT	PURPLE	LARGE	c2	TN	TN	
Average						**60%**	**60%**	**100%**

Table 26 Cases 16, 17 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase16,17	**STRETCH**	**ADULT**	**PURPLE**	**LARGE**	**c1**	**0.866**	**50.0**	**100**
Case6	STRETCH	ADULT	YELLOW	LARGE	c1	TP	TP	TP
Case7	STRETCH	ADULT	YELLOW	LARGE	c1	TP	TP	
Case11	STRETCH	ADULT	PURPLE	SMALL	c1	TP	TP	
Case12	STRETCH	ADULT	PURPLE	SMALL	c1	TP	TP	
Case18	STRETCH	CHILD	PURPLE	LARGE	c2	FP	FP	
Case19	DIP	ADULT	PURPLE	LARGE	c2		FP	
Average						**80%**	**66%**	**100%**

Table 27 Case 18 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase18	**STRETCH**	**CHILD**	**PURPLE**	**LARGE**	**c2**	**0.866**	**50.0**	**100**
Case8	STRETCH	CHILD	YELLOW	LARGE	c2	TN	TN	TN
Case13	STRETCH	CHILD	PURPLE	SMALL	c2	TN	TN	
Case16	STRETCH	ADULT	PURPLE	LARGE	c1	FN	FN	
Case17	STRETCH	ADULT	PURPLE	LARGE	c1	FN	FN	
Case20	DIP	CHILD	PURPLE	LARGE	c2	TN	TN	
Average						**60%**	**60%**	**100%**

Table 28 Case 19 - Balloon Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase19	**DIP**	**ADULT**	**PURPLE**	**LARGE**	**c2**	**0.866**	**50.0**	**100**
Case9	DIP	ADULT	YELLOW	LARGE	c2	TN	TN	TN
Case14	DIP	ADULT	PURPLE	SMALL	c2	TN	TN	
Case16	STRETCH	ADULT	PURPLE	LARGE	c1	FN	FN	
Case17	STRETCH	ADULT	PURPLE	LARGE	c1	FN	FN	
Case20	DIP	CHILD	PURPLE	LARGE	c2	TN	TN	
Average						**60%**	**60%**	**100%**

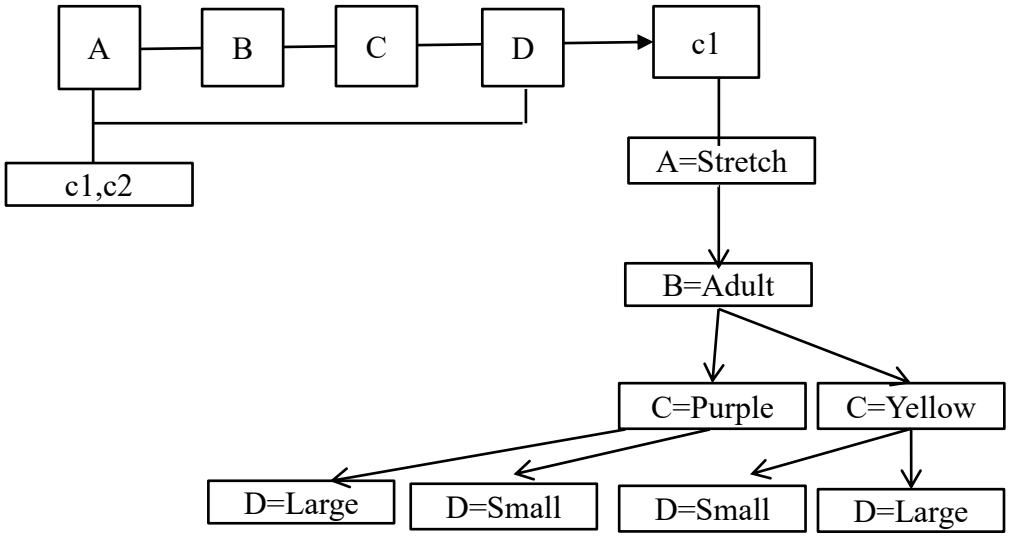


Figure 32 FP-CAR Algorithm Tree - Balloon Dataset – c1

Cases (1, 2), 6, (11, 12), (16, 17) in experiments 13, 16, 19 and 22 have matched a full pattern within FP-CAR algorithm without invoking the P-tree procedure to compensate the missing nodes. The novel strategy (CBRAR) is a significant step in machine learning and the data mining field, where a target case *Q* can be drawn directly from FP-CAR for a further research. In addition, cases that matched a full pattern can be reworked in Figure 32 to prove that CBRAR identifies the correct case using a frequent classed tree. Cases (3, 4, 8, 9, 13, 14, 18 and 19) are resolved by the CBRAR by invoking the P-trees to find a target case match in the FP-CAR tree.

The charts in Figure 33, Figure 35, Figure 36, Figure 37, Figure 38, Figure 39, Figure 40, Figure 41, Figure 42, Figure 43, Figure 44 and Figure 33, illustrate the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the charts, it is clear that in all Cases CBRAR registered 0 error rate, which is the lowest among the rates (40-80%), when compared to Jcolibri and FreeCBR. CBRAR shows a better performance in overall error rate and also correctly resolved the tar-get case.

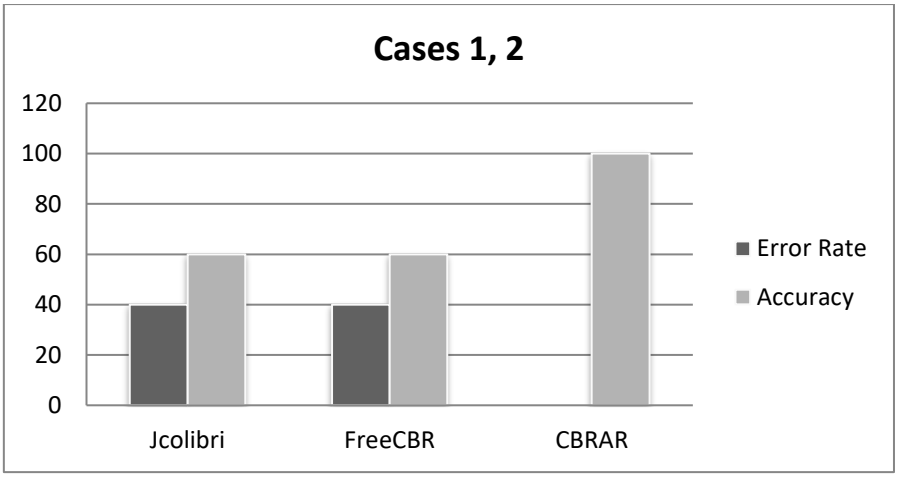


Figure 33 Cases 1, 2 Error and Accuracy Rate

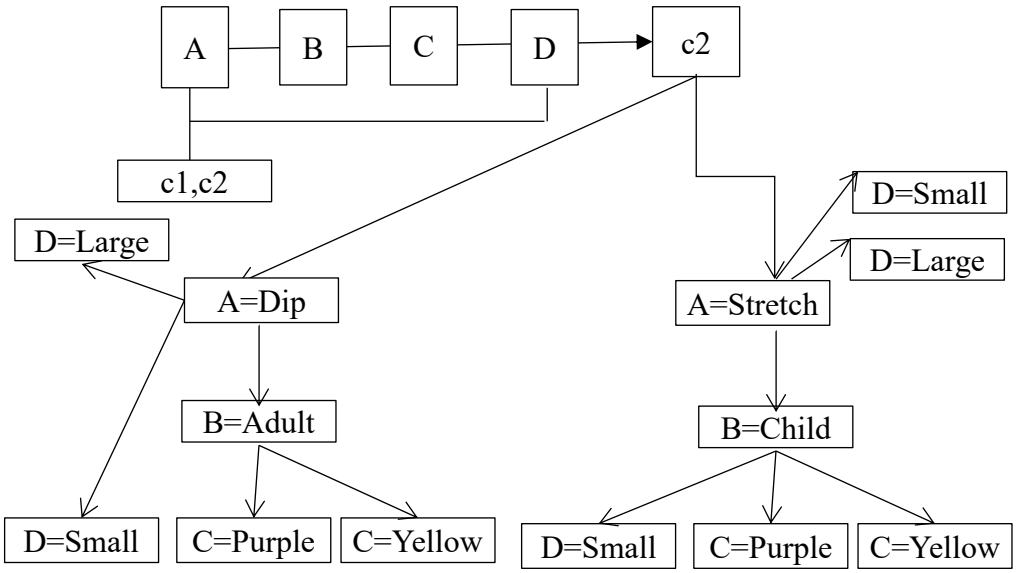


Figure 34 FP-CAR Algorithm Tree - Balloon Dataset – c2

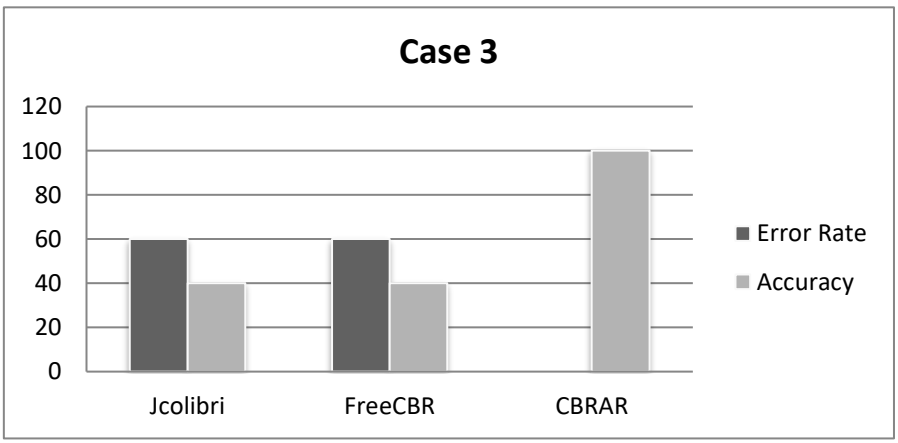


Figure 35 Case 3 Error and Accuracy Rate

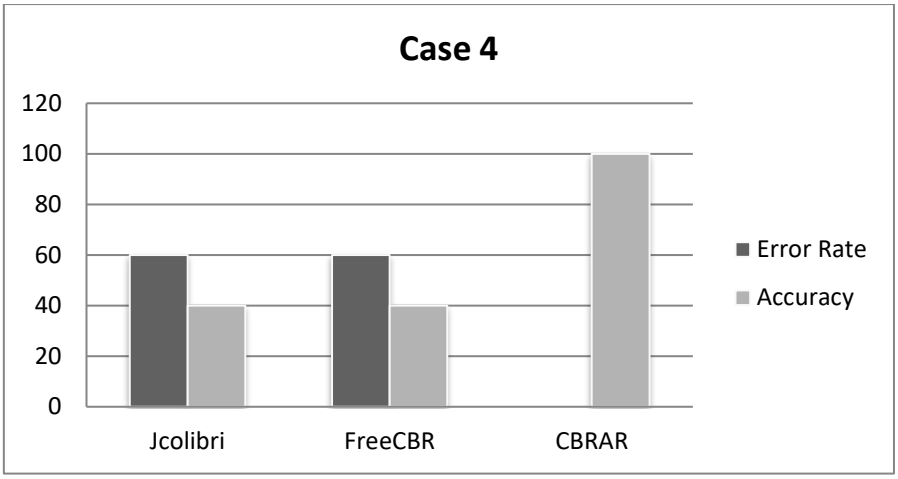


Figure 36 Case 4 Error and Accuracy Rate

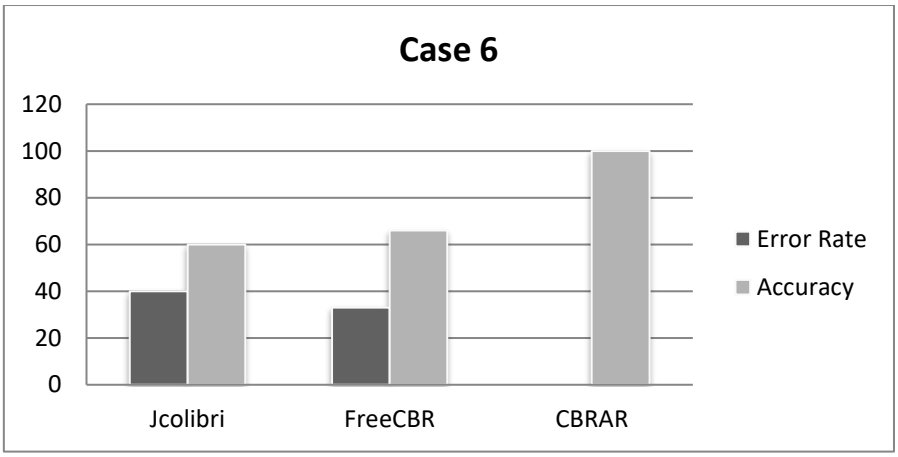


Figure 37 Case 6 Error and Accuracy Rate

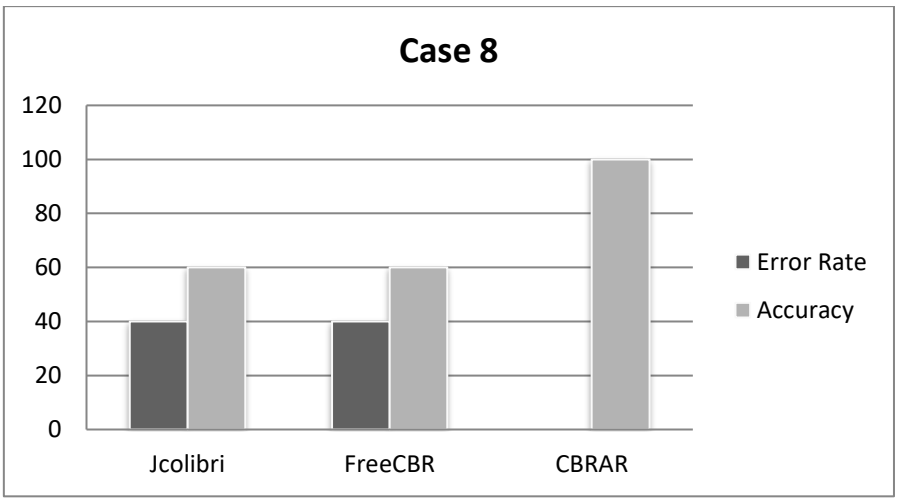


Figure 38 Case 8 Error and Accuracy Rate

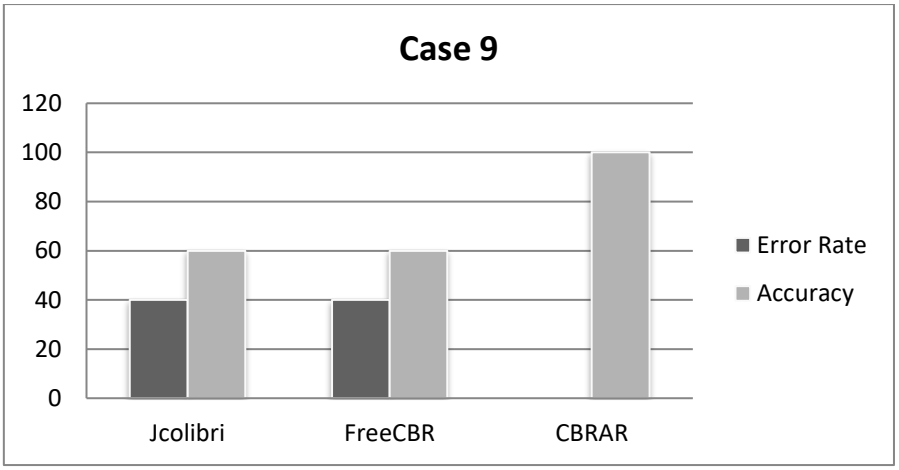


Figure 39 Case 9 Error and Accuracy Rate

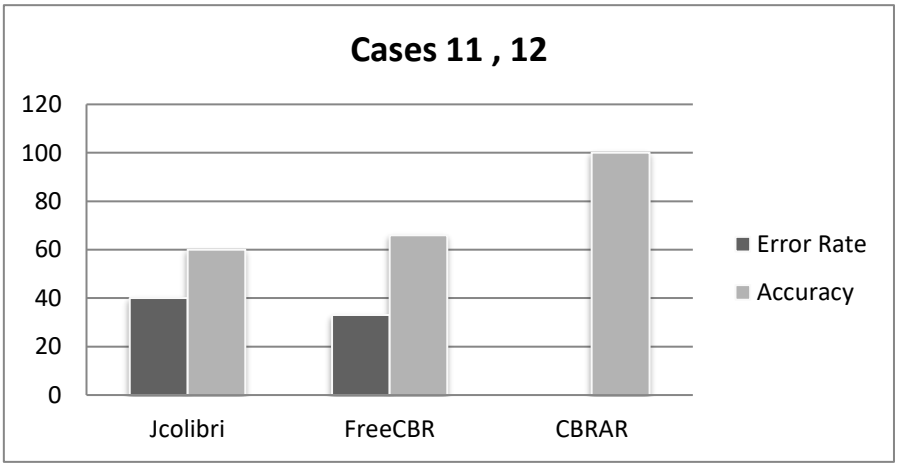


Figure 40 Cases 11, 12 Error and Accuracy Rate

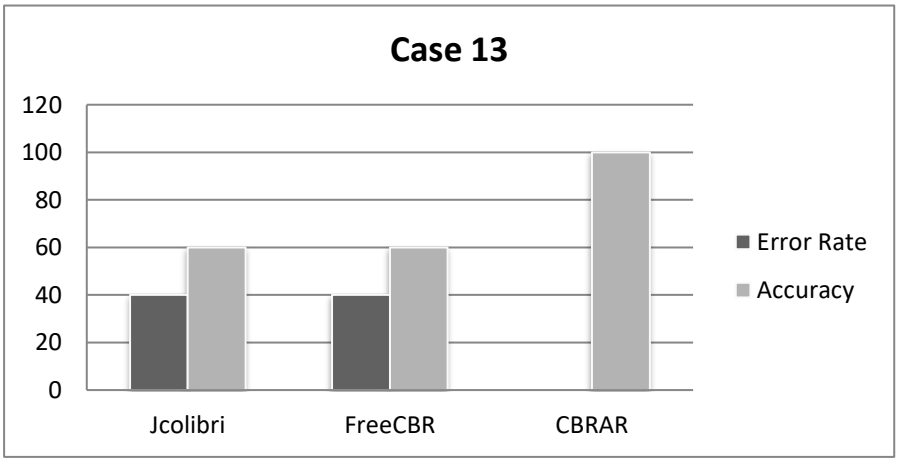


Figure 41 Case 13 Error and Accuracy Rate

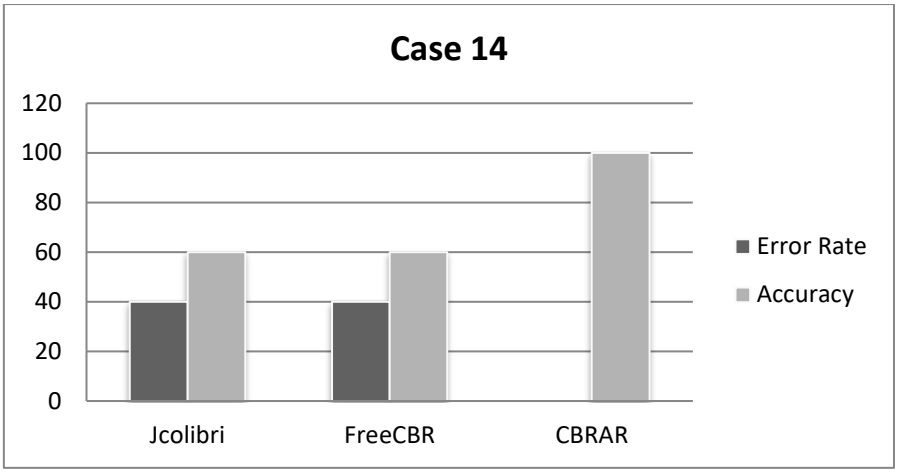


Figure 42 Case 14 Error and Accuracy Rate

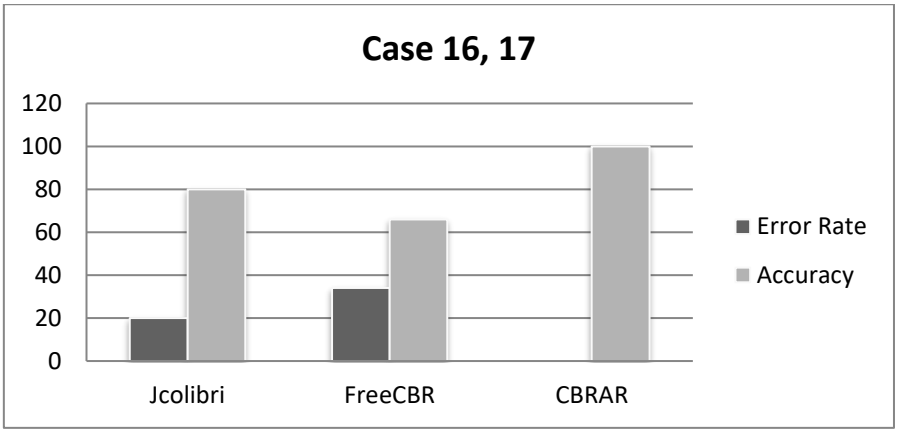


Figure 43 Cases 16, 17 Error and Accuracy Rate

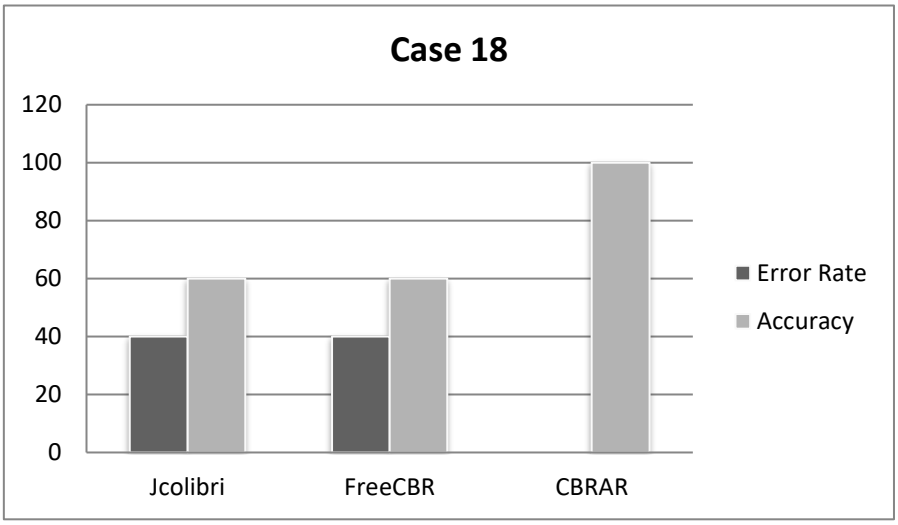


Figure 44 Case 18 Error and Accuracy Rate

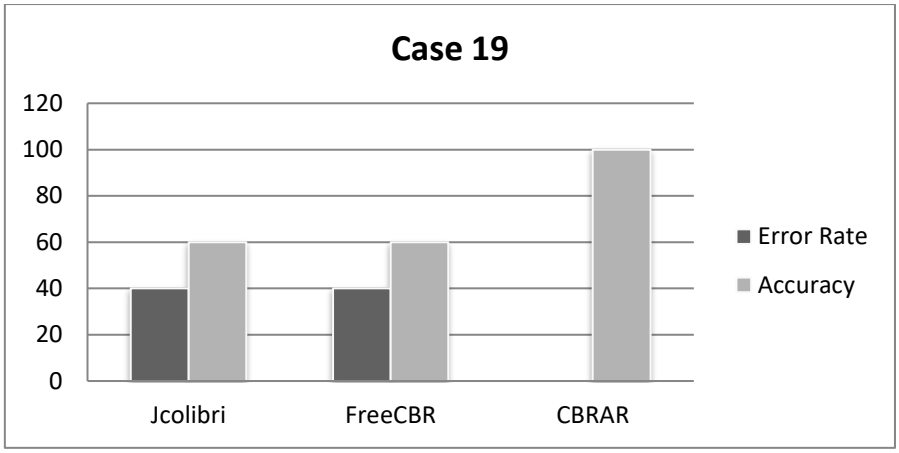


Figure 45 Case 19 Error and Accuracy Rate

4.7.2. **Discussion of Balloon Dataset Results**

The results show that 34 out of the 60 Jcolibri retrieved cases are classified as TP and TN giving 56% accuracy. By comparison, 34 of the 64 cases retrieved by FreeCBR are classified as TP and TN giving 53% accuracy. However, both Jcolibri and FreeCBR deliver "confusing" results. Our CBRAR strategy demonstrates an advantage over both Jcolibri and FreeCBR by resolving 12 out of 12 cases with 100% accuracy and no confusion. The accuracy of CBRAR was better compared to Jcolibri and FreeCBR. CBRAR resolved the ambiguity of the FP cases without confusion. Cases (1,2), 3, 4, 6, 8, 9, (11,12), 13, 14, (16,17), 18 and 19 in Table 17, Table 18, Table 19, Table 20, Table 21, Table 22, Table 23, Table 24, Table 25, Table 26, Table 27 and Table 28 can be reworked in Figure 32 and Figure 34 to prove that CBRAR identifies a correct case using a frequent classed tree.

The line chart in Figure 46 illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, it is clear that in the Balloon Dataset's experiments, CBRAR registered 0 error rate, which is the lowest among the rates when compared to Jcolibri and FreeCBR. CBRAR also correctly resolved 12 out of 12 cases. In Cases 1, 2, 11, 12, 16, 17 it is noticeable that a full match pattern compared to FP-CAR algorithm without invoking the P-tree procedure. The novel strategy (CBRAR) plays a significant role in the CBR field as a second contribution

of this research, where a target case Q can be drawn directly from a classed tree.

CBRAR offered many advantages by resolving 12 cases when compared to using other CBR tools. 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.

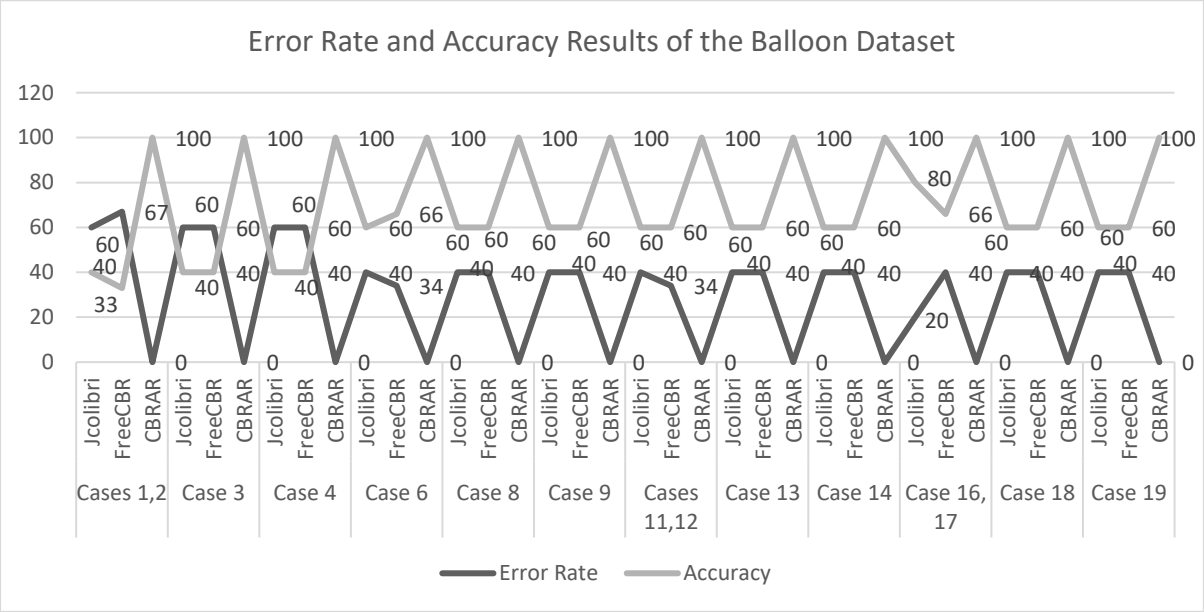


Figure 46 Error Rate and Accuracy Results Assembled of Balloon Dataset

4.8. Results of Post-Operative Patient Dataset

This section illustrates the retrieved cases of the Post-Operative Patient dataset CBR tools which identified ambiguous answers. The section includes tables, figures and charts to show the outcomes in terms of accuracy and error rates.

4.8.1. Experiments 25, 26, 27 and 28: Cases (20, 36, 44), (5, 69, 74) and 14 Using CBRAR Strategy

In Experiment 25, 26, 27 and 28 cases (20, 36, 44), (5, 69, 74) and 14 were identified as the ambiguous cases as indicated in the results achieved in Table 29, Table 30,

Table **31**, Table 32 and Table 33. These tables also indicate the cases that are identified as similar for both Jcolibri and FreeCBR. For instance, in **Table 29** the "Attributes" columns start with A followed by 7 additional attributes B, C, D, E, F, G and H. In addition, the class label column includes the patient class (c1, c2 and c3) as explained in the dataset characteristics. The "Accuracy" columns show the comparison between Jcolibri, FreeCBR and CBRAR. **Table 29** also illustrates that, for each new case applied to CBR, 5 different cases are retrieved by Jcolibri with same similarity ratio of 0.953 %, and 7 cases are retrieved by FreeCBr with similarity ratio of 64.64%.

In the twenty fifth experiment, NewCases20, 36,44 applied to the CBR, Jcolibri retrieved 4 TP and 1 FP cases with the same similarity ratio, giving 80% accuracy, whereas FreeCBR retrieved 5 TP and 2 FP, giving 71% accuracy.

In the twenty sixth experiment, when NewCase5, 69, 74 are applied to the CBR, Jcolibri retrieved 3 TP and 2 FP case with the same similarity ratio, giving 60% accuracy; whereas FreeCBR retrieved 3 TP and 3 FP, giving 50% accuracy.

In the twenty seventh experiment, when NewCase14 is applied to the CBR, Jcolibri retrieved 4 TP and 1 FP case with the same similarity ratio, giving 80% accuracy; whereas FreeCBR retrieved 5 TP and 1 FP, giving 83% accuracy.

CBRAR retrieved 1 TP case of all cases (20, 36, 44), (5, 69, 74) and 14, from new strategy which is the correct case thus giving 100% accuracy and outperforming the performance of the CBR tools used. This is shown in Figure 48, Figure 49 and Figure 51. All resolved cases of post-operative dataset Table 29, Table 30 and Table 31 can be reworked in Figure 47 and Figure 50 to prove that CBRAR identifies the correct case using a frequent classed tree.

Table 29 Cases 20, 36 and 44 - Post-Operative Patient - CBR Results

Cases	Attributes									Accuracy		
	A	B	C	D	E	F	G	H	Class	Jcolibri	FreeCBR	CBRAR
NewCase20, 36 ,44	**sta**	**sta**	**ten**	**good**	**mid**	**sta**	**mid**	**mid**	**c1**	**0.935**	**64.64**	**100**
Case9	sta	sta	ten	good	mid	sta	high	mid	c2	FP	FP	TP
Case11	sta	sta	fiften	good	mid	sta	mid	mid	c1	TP	TP	
Case24	sta	unsta	ten	good	mid	sta	mid	mid	c1	TP	TP	
Case34	sta	sta	ten	good	mid	sta	low	mid	c1	TP	TP	
Case63	sta	sta	ten	good	mid	unsta	mid	mid	c1	TP	TP	
Case66	sta	sta	ten	good	mid	sta	mid	high	c2		FP	
Case80	sta	sta	ten	good	mid	sta	mid	high	c1		TP	
Average										**80%**	**66%**	**100%**

Table 30 Cases 5, 69 and 74 - Post-Operative - CBR Results

Cases	Attributes									Accuracy		
	A	B	C	D	E	F	G	H	Class	Jcolibri	FreeCBR	CBRAR
NewCase5, 69 ,74	**sta**	**sta**	**ten**	**exce**	**mid**	**sta**	**mid**	**high**	**c1**	**0.935**	**64.64**	**100**
Case2	sta	sta	ten	exce	mid	sta	high	high	c2	FP	FP	TP
Case62	sta	sta	ten	exce	mid	sta	low	high	c1	TP	TP	
Case66	sta	sta	ten	good	mid	sta	mid	high	c2	FP	FP	
Case70	sta	sta	ten	exce	mid	sta	mid	low	c1	TP	TP	
Case80	sta	sta	ten	good	mid	sta	mid	high	c1	TP	TP	
Case82	sta	sta	ten	exce	mid	sta	mid	mid	c2		FP	
Average										**60%**	**50%**	**100%**

Table 31 Case 14 - Post-Operative Patient - CBR Results

Cases	Attributes									Accuracy		
	A	B	C	D	E	F	G	H	Class	Jcolibri	FreeCBR	CBRAR
NewCase14	**sta**	**unsta**	**ten**	**good**	**mid**	**mod-sta**	**high**	**mid**	**c1**	**0.935**	**64.64**	**100**
Case20	sta	sta	ten	good	mid	sta	mid	mid	c1	TP	TP	TP
Case26	sta	sta	ten	good	high	mod-sta	high	mid	c1	TP	TP	
Case30	sta	unsta	ten	good	mid	unsta	mid	mid	c2	FP	FP	

Case31	sta	unsta	ten	good	mid	sta	mid	high	c1	TP	TP	
Case36	sta	sta	ten	good	mid	sta	mid	mid	c1	TP	TP	
Case54	sta	unsta	ten	good	mid	mod-sta	mid	mid	c1		TP	
Average										**80%**	**83%**	**100%**

The bar charts in Figure 48, Figure 49 and Figure 51 illustrate the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the charts, it is clear that in Experiments 25, 26 and 27 and 44, CBRAR registered 0 error rate, which is the lowest among the rates (20%, 34%), (40%, 50%) and (20%, 17%), when compared to Jcolibri and FreeCBR. CBRAR shows a better performance in overall error rate and also correctly resolved the target cases.

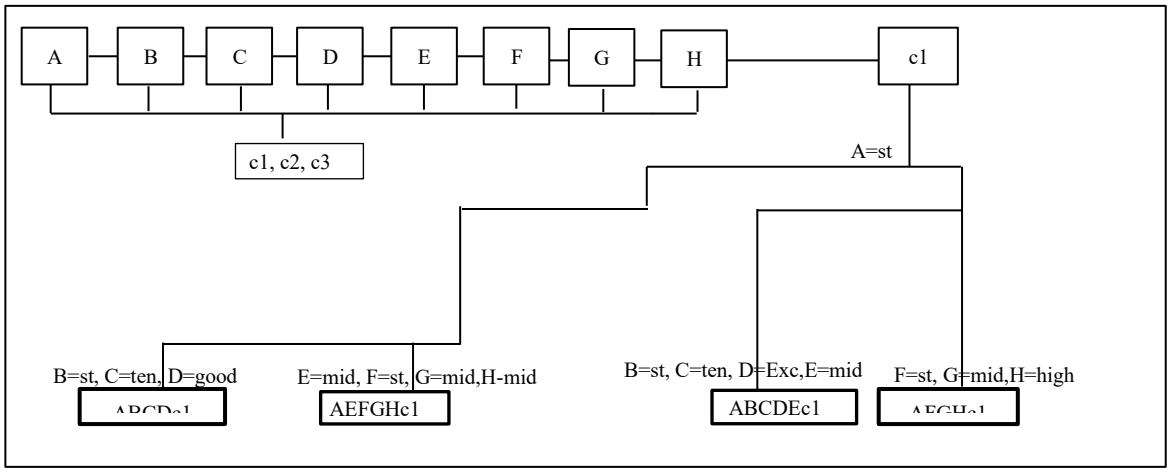


Figure 47 FP-CAR Algorithm Tree - Post-Operative Dataset - 1

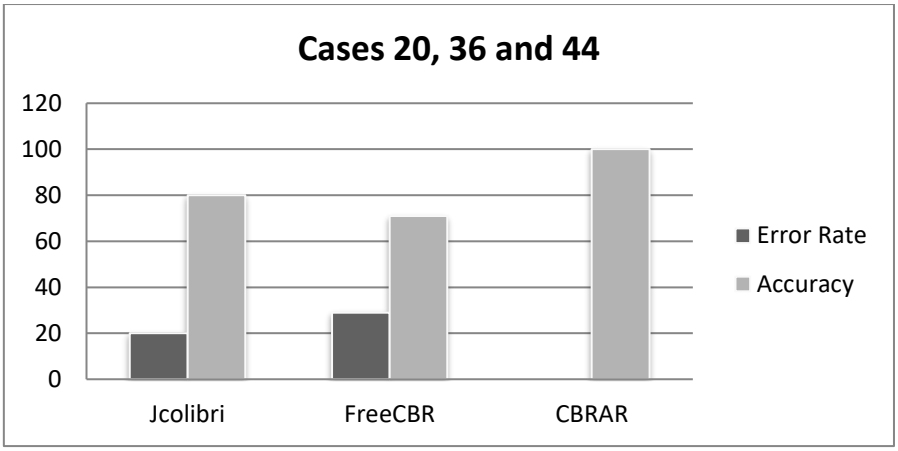


Figure 48 Cases 20, 26 and 44 Error and Accuracy Rate

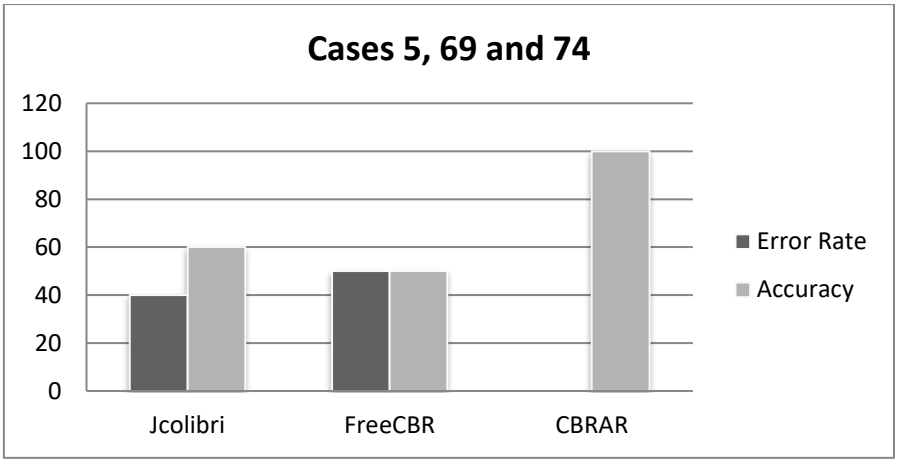


Figure 49 Cases 5, 69 and 74 Error and Accuracy Rate

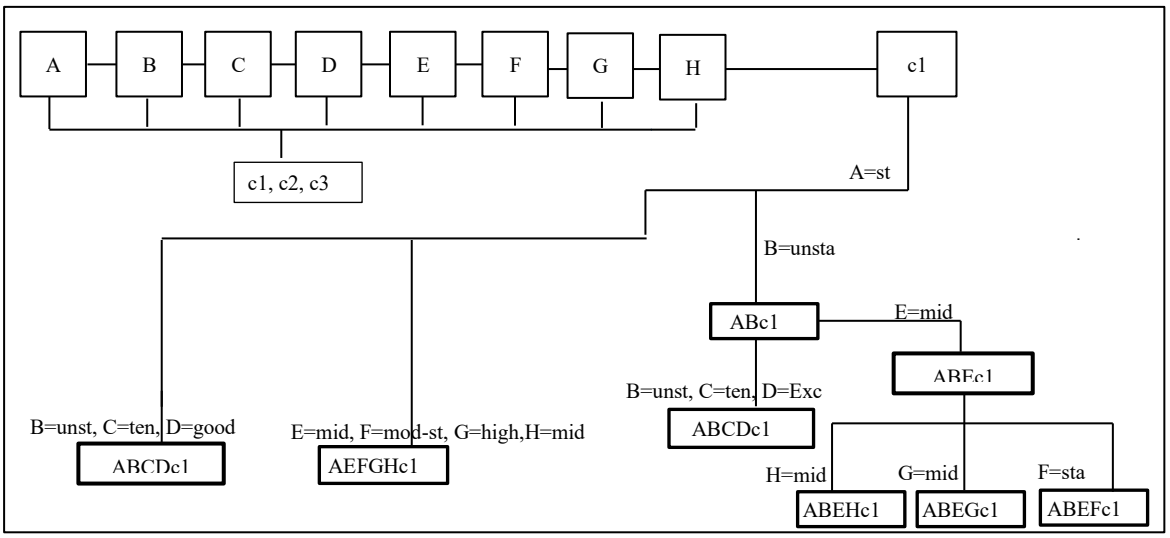


Figure 50 FP-CAR Algorithm Tree - Post-Operative Patient Dataset - 2

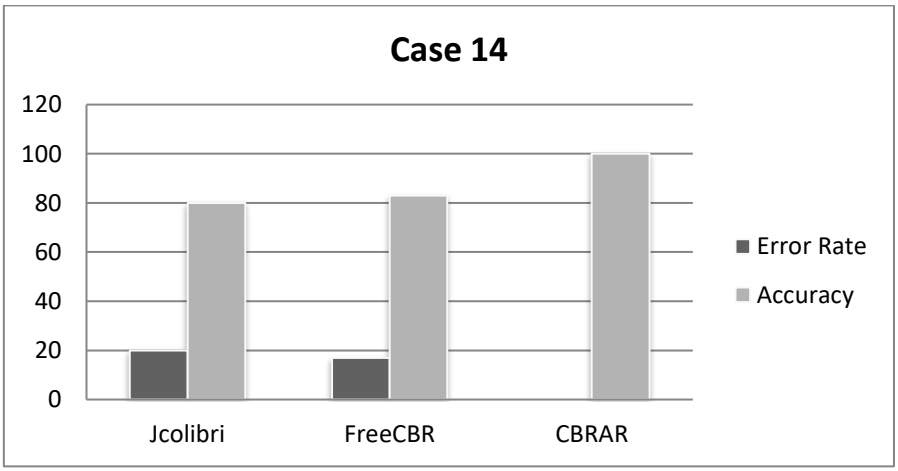


Figure 51 Case 14 Error and Accuracy Rate

4.8.2. **Experiment 28: Cases 48, 83 Using CBRAR Strategy**

In experiment 28, cases 48, 83 were identified as the ambiguous cases as shown in the results given in **Table 32** and **Table 33**.

Table 32 show the cases that are identified as similar for both Jcolibri and FreeCBR where each new case applied to CBR, 5 different cases are retrieved giving the same similarity ratios i.e. 0.935 and 64.64. In other words, Jcolibri and FreeCBR retrieved 2 TP and 3 FP cases, giving 40% accuracy whereas CBRAR did not retrieve any case from new model, giving 0% accuracy as shown in **Figure 52**.

Table 32 Case 48 - Post-Operative - CBR Results

Cases	**Attributes**									**Accuracy**		
	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**Class**	**Jcolibri**	**FreeCBR**	**CBRAR**
NewCase48	**sta**	**unsta**	**ten**	**exce**	**mid**	**sta**	**mid**	**mid**	**c1**	**0.935**	**64.64**	**100**
Case24	sta	unsta	ten	good	mid	sta	mid	mid	c1	TP	TP	FP
Case67	sta	unsta	ten	exce	mid	sta	low	mid	c1	TP	TP	
Case79	unsta	unsta	ten	exce	mid	sta	mid	mid	c2	FP	FP	
Case82	sta	sta	ten	exce	mid	sta	mid	mid	c2	FP	FP	
Case87	sta	unsta	fiften	exce	mid	sta	mid	mid	c2	FP	FP	
Average										**40%**	**40%**	**0%**

A potential solution of case 48 can be found in the FP-tree of Figure 50, where 4 nodes were identified as a partial match of case 48, i.e. (A=sta , B=unsta , C=ten , D=exce), while the remaining 4 nodes (E=mid , F=st ,G=mid ,H=mid) to build case 48 from the FP-CAR can be utilised if the remaining nodes were under the root A rather than node B. Therefore, invoking P-tree or Union have not resolved this case. Figure 52 illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, Case48, CBRAR registered 100 error rate, which is the highest among the rates (60%) when compared to Jcolibri and FreeCBR.

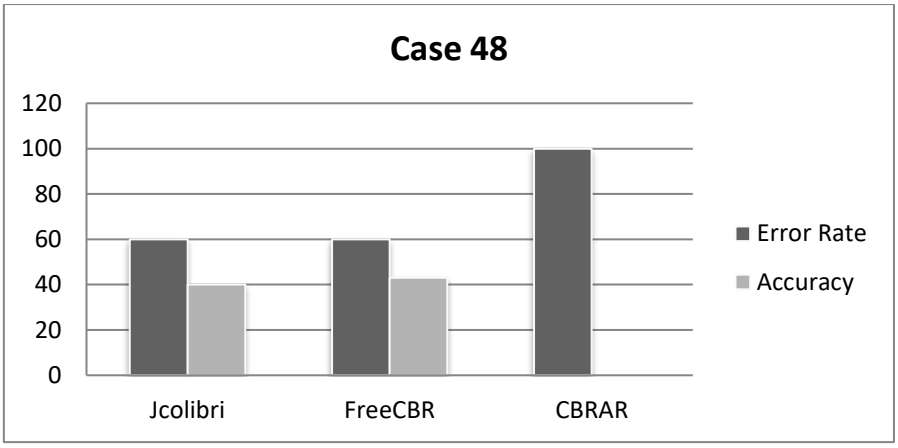


Figure 52 Case 48 Error and Accuracy Rate

For NewCase83, Table 33 show the cases that are identified as similar for both Jcolibri and FreeCBR where for each new case applied to CBR, 3 different cases are retrieved giving the same similarity ratios i.e. 0.935 and 64.64. In other words, Jcolibri and FreeCBR retrieved 3 FN cases, giving 0% accuracy. Similarly, CBRAR did not retrieve any case from new model, giving 0% as shown in Figure 53.

It was found that the proposed system did not resolved case 83, due to the case being under class c3 where just a one record appears. In other words, from rare cases all CBR tools and the enhanced one were not able to find similar cases to compare with. Therefore, CARs do not contain any rules which can belong to c3. Based on that, the FP-CAR algorithm will not produce either a partial solution as was shown in the Acute Inflammation urinary bladder dataset or a full as a full solution which was proved in the Balloon dataset. This dilemma is a special classification case problem where the majority of c1 class and the minority c3 class due to unequal distribution.

Table 33 Case 83 - Post-Operative Patient - CBR Results

Cases	Attributes									Accuracy		
	A	B	C	D	E	F	G	H	Class	Jcolibri	FreeCBR	CBRAR
NewCase83	sta	sta	ten	good	mid	unsta	low	mid	c3	0.935	64.64	100

Case34	sta	sta	ten	good	mid	sta	low	mid	c1	FN	FN	FN
Case63	sta	sta	ten	good	mid	unsta	mid	mid	c1	FN	FN	
Case64	sta	sta	ten	exce	mid	unsta	low	mid	c2	FN	FN	
Case6	sta	sta	fiften	good	high	unsta	low	mid	c2	0.866	42.264	
Case9	sta	sta	ten	good	mid	sta	high	mid	c2	08.66	42.264	
Average										**0%**	**0%**	**0%**

Figure 53 illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR, in Case83, all CBR tools registered 100 error rate when a comparison is performed to show their performance.

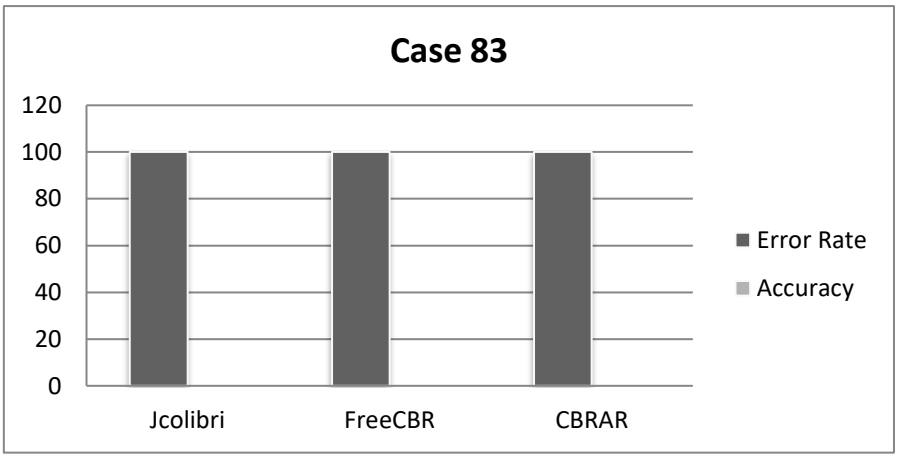


Figure 53 Case 83 Error and Accuracy Rate

4.8.3. **Discussion on Post-Operative Patient Dataset**

The results show that 13 out of the 20 Jcolibri retrieved cases are classified as TP giving 65% accuracy. By comparison, 15 of the 24 cases retrieved by FreeCBR are classified as TP giving 62.5% accuracy. However, both Jcolibri and FreeCBR deliver "confusing" results. Our CBRAR strategy demonstrates an advantage over both Jcolibri and FreeCBR by resolving 3 out of 4 cases with 75% accuracy; Case83 was not counted because of none of the CBR tools has retrieved a correct case. The accuracy of CBRAR was better compared to Jcolibri and FreeCBR. CBRAR resolved the ambiguity of the FP cases without confusion. Cases (20,36,44),

(5,69,74) and 14 in Table 29, Table 30 and Table 31 can be reworked in Figure 47 and Figure 50 to prove that CBRAR identifies a correct case using a FP-CAR tree.

The line chart in Figure 54 illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, it is clear that in Cases 20,36 and 36, CBRAR registered 0 error rate, which is the lowest among the rates (20, 34) when compared to Jcolibri and FreeCBR. The results also show that the error rate of CBRAR is the lowest on Cases (20,36,44) and Case24 thus giving the highest accuracy, when compared to the other CBR tools used. In Case48, it noticeable that the (40) % error rate for Jcolibri and FreeCBR was considerably lower than CBRAR. In addition, neither the CBR tools nor CBRAR has retrieved the target case 83 due to a special problem of classification when unbalanced data are examined.

Therefore, whilst CBRAR did not resolve Case83 neither of the other CBR tools offered any advantage when compared to the new strategy. To sum up, 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.

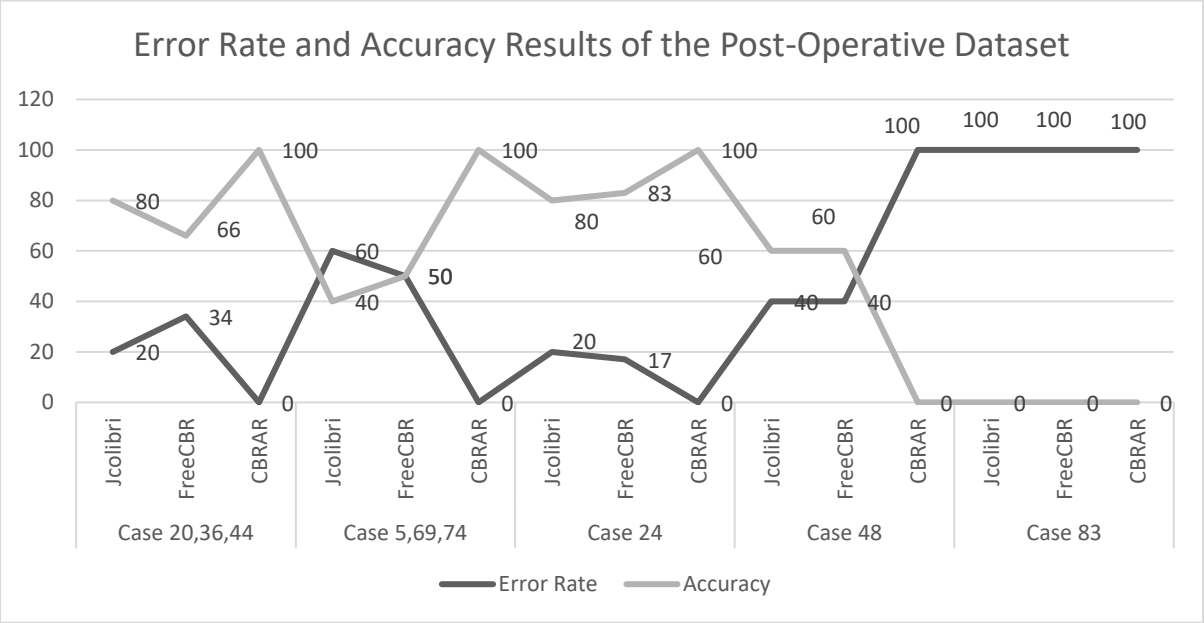


Figure 54 Error Rate and Accuracy Results Assembled of the Post-Operative Dataset

4.9. Results of Lenses Dataset

This section shows the retrieved cases from the Lenses dataset where the CBR tools identified ambiguous answers. The section includes tables, figures and charts to investigate the outcomes of accuracy and error rates.

4.9.1. Experiments 29, 30, 31 and 32: Case 11, 21 and 19 Using CBRAR Strategy

In Experiments 29, 30, 31 and 32, cases 11, 21 and 19 were identified as an ambiguous cases as indicated in the results illustrated in Table 34, Table 35, Table 36, Table 37,

Table **38** and Table 39. These tables also indicate the cases that are identified as similar for both Jcolibri and FreeCBR. For instance, in Table 34 the "Attributes" columns start with A followed by 3 additional attributes B, C and D and the 3 class labels column to determine lenses fitting (c1, c2 and c3). The "Accuracy" columns show the comparison between Jcolibri, FreeCBR and CBRAR. Table 34 also shows that, for each new case applied to CBR, 5 different cases are retrieved by Jcolibri and FreeCBR with the same similarity ratio of 0.866 % and 50.0%.

In the twenty-ninth experiment, NewCase11 applied to the CBR, Jcolibri and FreeCBR retrieved 4 TP and 1 FP cases with the same similarity ratio, giving 80% accuracy.

Table 34 Case 11 - Lenses Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase11	**yes**	**myope**	**redc**	**pre-prspic**	**c1**	**0.866**	**50.0**	**100**
Case3	yes	myope	redc	young	c1	TP	TP	TP
Case9	no	myope	redc	pre-prspic	c1	TP	TP	
Case12	yes	myope	norm	pre-prspic	c3	FP	FP	
Case15	yes	hyprmtr	redc	pre-prspic	c1	TP	TP	

Case19	yes	myope	redc	prspic	c1	TP	TP	
Average						**80%**	**80%**	**100%**

In the thirtieth experiment, NewCase21 applied to the CBR, Jcolibri retrieved 4 TP and 1 FP cases with the same similarity ratio, giving 80% accuracy, Similarlly FreeCBR retrieved 4 TP and 1 FP, giving 80% accuracy.

Table 35 Case 21 - Lenses Dataset - CBR Results

Cases	**Attributes**					**Accuracy**		
	A	**B**	**C**	**D**	**Class**	**Jcolibri**	**FreeCBR**	**CBRAR**
NewCase21	**no**	**hyprmtr**	**redc**	**prspic**	**c1**	**0.866**	**50.0**	**100**
Case5	no	hyprmtr	redc	young	c1	TP	TP	TP
Case13	no	hyprmtr	redc	pre-prspic	c1	TP	TP	
Case17	no	myope	redc	prspic	c1	TP	TP	
Case22	no	hyprmtr	norm	prspic	c2	FP	FP	
Case23	yes	hyprmtr	redc	prspic	c1	TP	TP	
Average						**80%**	**80%**	**100%**

In the thirty-first experiment, NewCase19 applied to the CBR, Jcolibri and FreeCBR retrieved 4 TP and 1 FP cases with the same similarity ratio, giving 80% accuracy.

Table 36 Case 19 - Lenses Dataset - CBR Results

Cases	**Attributes**					**Accuracy**		
	A	**B**	**C**	**D**	**Class**	**Jcolibri**	**FreeCBR**	**CBRAR**
NewCase19	**yes**	**myope**	**redc**	**prspic**	**c1**	**0.866**	**50.0**	**100**
Case3	yes	myope	redc	young	c1	TP	TP	TP
Case11	yes	myope	redc	pre-prspic	c1	TP	TP	
Case17	no	myope	redc	prspic	c1	TP	TP	
Case20	yes	myope	norm	prspic	c3	FP	FP	
Case23	yes	hyprmtr	redc	prspic	c1	TP	TP	
Average						**80%**	**80%**	**100%**

CBRAR retrieved 1 TP case of experiments 29, 30 and 31from new strategy which is the correct

case hence achieving 100% accuracy and outperforming the performance of the CBR tools used. This is illustrated in Figure 56. Cases 11, 21 and 19 in Table 34, Table 35 and Table 36 can be reworked in Figure 55 to prove that CBRAR identifies the correct case using a frequent classed tree.

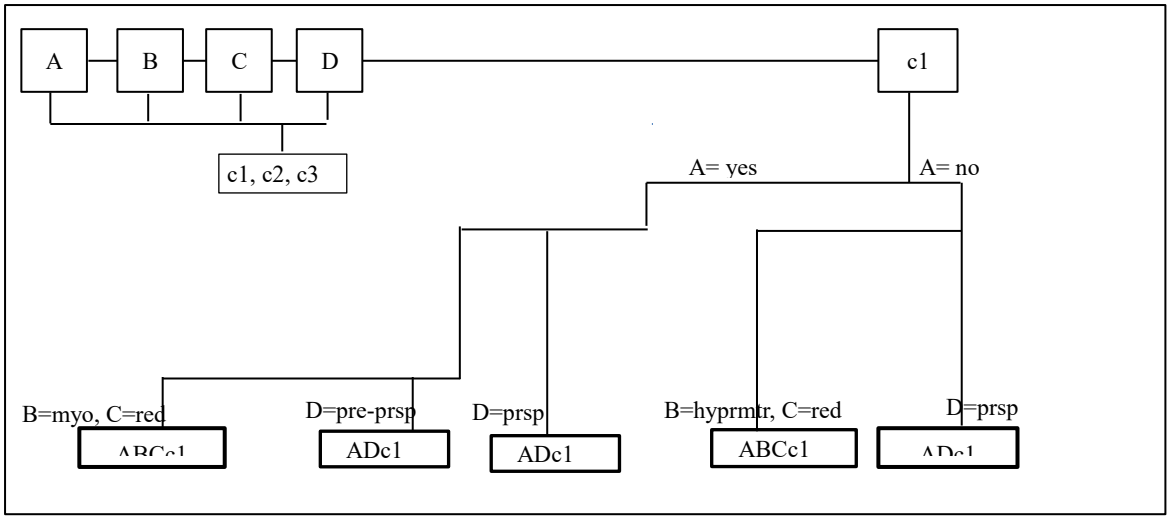


Figure 55 FP-CAR Algorithm Tree - Lenses Dataset – c1

The bar charts in Figure 56, Figure 57 and Figure 58 illustrate the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, it is clear that in experiments 29, 30 and 31, CBRAR registered 0 error rate, which is the lowest among the rates (20%) of all cases when compared to Jcolibri and FreeCBR.

CBRAR shows a better performance in overall error rate and also correctly resolved the target case.

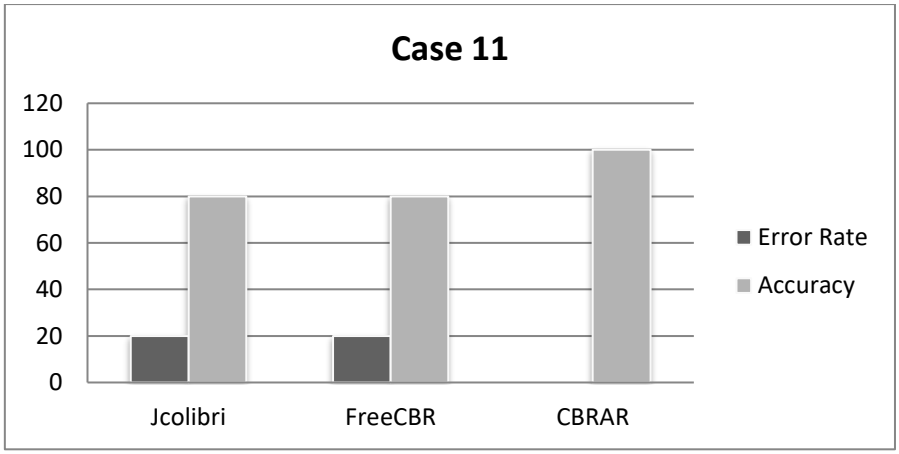


Figure 56 Case 11 Error and Accuracy Rate

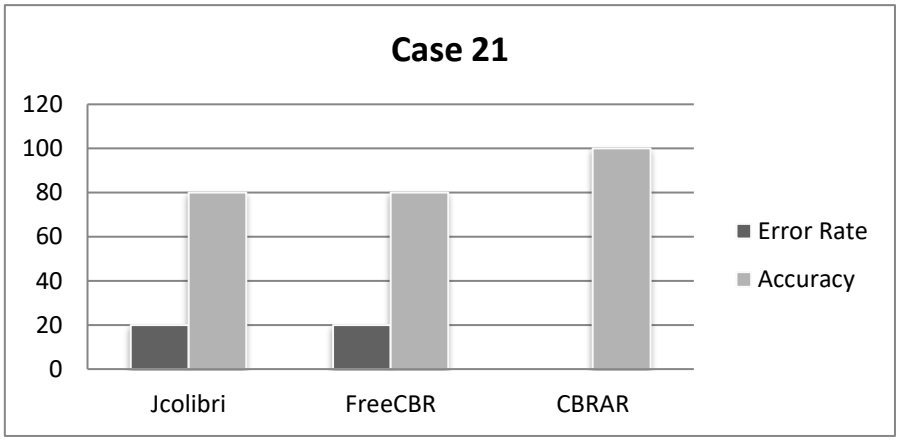


Figure 57 Case 21 Error and Accuracy Rate

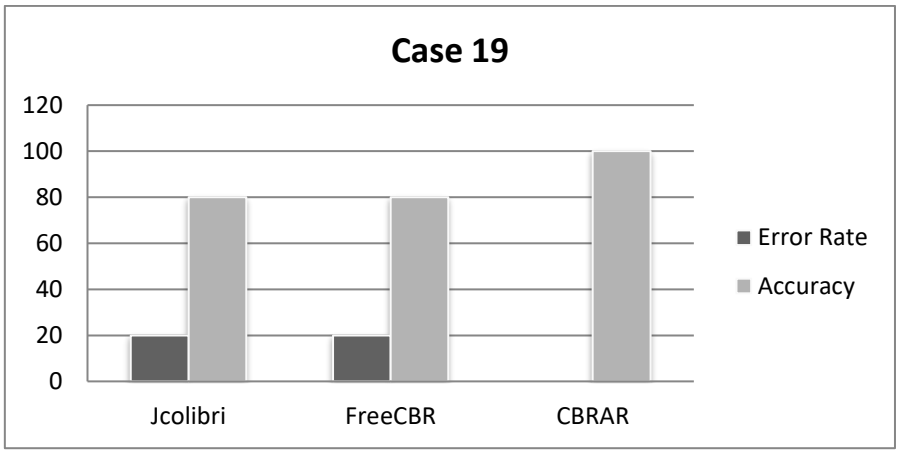


Figure 58 Case 19 Error and Accuracy Rate

4.9.2. Experiment 32: Cases 6, 12 and 14 Using CBRAR Strategy

In experiment 32, cases 6, 12 and 14 were identified as the ambiguous cases as shown in the results given in Table 37,

Table **38** and Table 39. These tables show the cases that are identified as similar for both Jcolibri and FreeCBR where each new case applied to CBR, 5 different cases are retrieved giving the same similarity ratios i.e. 0.866 and 50.0. For case 6, Jcolibri and FreeCBR retrieved 3 TN and 2 FN cases, giving 60% accuracy in Table 37. For case 12, 1 TN and 4 FN cases, giving 25% accuracy in

Table **38**. For case 14, 3 TN and 2 FN, giving 60% accuracy in Table 39. CBRAR did not retrieve any case from the FP-CAR, giving 0% for cases 6, 12 and 14. This is illustrated in **Figure 60**.

Table 37 Case 6 - Lenses Dataset - CBR Results

Cases	Attributes					Accuracy		
	A	B	C	D	Class	Jcolibri	FreeCBR	CBRAR
NewCase6	**no**	**hyprmtr**	**norm**	**young**	**c2**	**0.866**	**50.0**	**100**
Case2	no	myope	norm	young	c2	TN	TN	FN
Case5	no	hyprmtr	redc	young	c1	FN	FN	
Case8	yes	hyprmtr	norm	young	c3	FN	FN	
Case14	no	hyprmtr	norm	pre-prspic	c2	TN	TN	
Case22	no	hyprmtr	norm	prspic	c2	TN	TN	
Average						**60%**	**60%**	**0%**

Table 38 Case 12 - Lenses Dataset - CBR Results

Cases	Attributes					Accuracy
	A	B	C	D	Class	

						Jcolibri	FreeCBR	CBRAR
NewCase12	**yes**	**myope**	**norm**	**pre-prspic**	**c3**	**0.866**	**50.0**	**100**
Case10	no	myope	norm	pre-prspic	c2	FN	FN	FN
Case11	yes	myope	redc	pre-prspic	c1	FN	FN	
Case16	yes	hyprmtr	norm	pre-prspic	c1	FN	FN	
Case20	yes	myope	norm	prspic	c3	TP	TP	
Case2	no	myope	norm	young	c2	0.70	0.43	
Average						**25%**	**25%**	**0%**

Table 39 Case 14 - Lenses Dataset - CBR Results

Cases	**Attributes**					**Accuracy**		
	A	**B**	**C**	**D**	**Class**	**Jcolibri**	**FreeCBR**	**CBRAR**
NewCase14	**no**	**hyprmtr**	**norm**	**pre-prspic**	**c2**	**0.866**	**50.0**	**100**
Case6	no	hyprmtr	norm	young	c2	TN	TN	FN
Case10	no	myope	norm	pre-prspic	c2	TN	TN	
Case13	no	hyprmtr	redc	pre-prspic	c1	FN	FN	
Case16	yes	hyprmtr	norm	pre-prspic	c1	FN	FN	
Case22	no	hyprmtr	norm	prspic	c2	TN	TN	
Average						**60%**	**60%**	**0%**

A possible solution of cases 6, 12 and 14 can be found in the FP-tree of Figure 59 , where 3 nodes were identified as a partial match. Firstly, case 6, i.e. (A=no, B= hyprmtr, C=norm), while the remaining node (D= young) to build case 6 from the FP-CAR cannot be found if the P-tree is invoked. Secondly, Case 12, is another potential solution which can be found where the first three nodes are (A= yes, B= myope, C=norm), while the remaining node (D= pre-prspic) to build this case from FP-CAR is missing. Thirdly, a possible solution in case 14 can be noticed in the first three nodes i.e. (A= no, B= hyprmtr, C=norm), while the remaining node (D= pre-prspic) to build this case from FP-CAR is not produced.

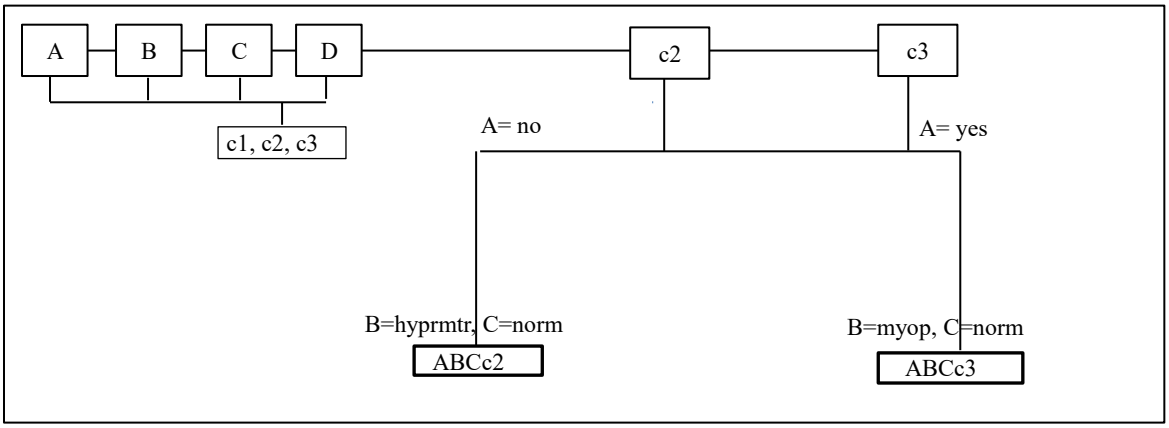


Figure 59 FP-CAR Algorithm Tree - Lenses Dataset – c2 and c3

Figure 60 illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, Cases (6, 12 and 14), CBRAR registered 100% error rate, which is the highest among the rates (40%, 75% and 40%) when compared to Jcolibri and FreeCBR with ambiguous answers.

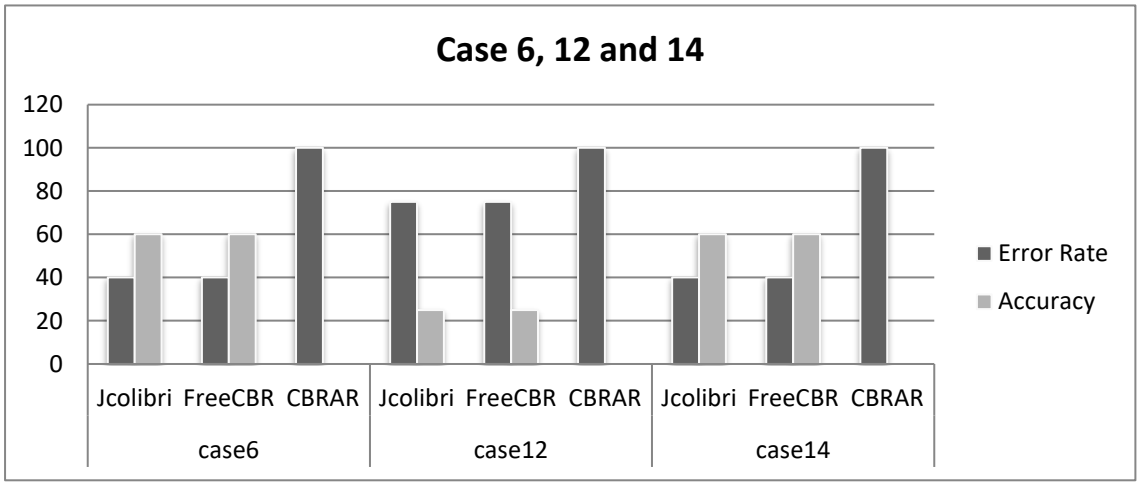


Figure 60 Cases 6, 12 and 14 Error and Accuracy Rate

4.9.3. **Discussion on Lenses Dataset**

The results show that for 12 out of the 15, Jcolibri and FreeCBR retrieved cases are classified as TP giving 80% accuracy. But, both Jcolibri and FreeCBR deliver “confusing” results. Our CBRAR strategy demonstrates an advantage over both Jcolibri and FreeCBR by resolving 3

out of 6 cases with 50% accuracy. The accuracy of CBRAR was better compared to Jcolibri and FreeCBR and resolved the ambiguity of the FP cases without confusion. Cases 11, 21 and 19 in Table 34, Table 35 and Table 36 can be reworked in Figure 55 to prove that CBRAR identifies a correct case using a FP-CAR tree.

The line chart in Figure 61 illustrates the error rate and accuracy of Jcolibri, FreeCBR and CBRAR. From the chart, it is clear that in Case 11, CBRAR registered 0 error rate, which is the lowest the rate (20%) when compared to Jcolibri and FreeCBR. The results also show that the error rate of CBRAR is the lowest on Case 21 thus giving the highest accuracy 100%, when compared to the other CBR tools used. In Case 19, it also noticeable that the 20 % error rate of Jcolibri and FreeCBR was considerably higher than CBRAR. In addition, neither CBR tools nor CBRAR has retrieved the target cases 6, 12 and 14 whereas a partial solution was found in the FP-CAR tree.

Therefore, whilst CBRAR did not resolve Cases 6 12 and 14, neither of the other CBR tools offered any advantage when compared to the new strategy. To conclude, 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.

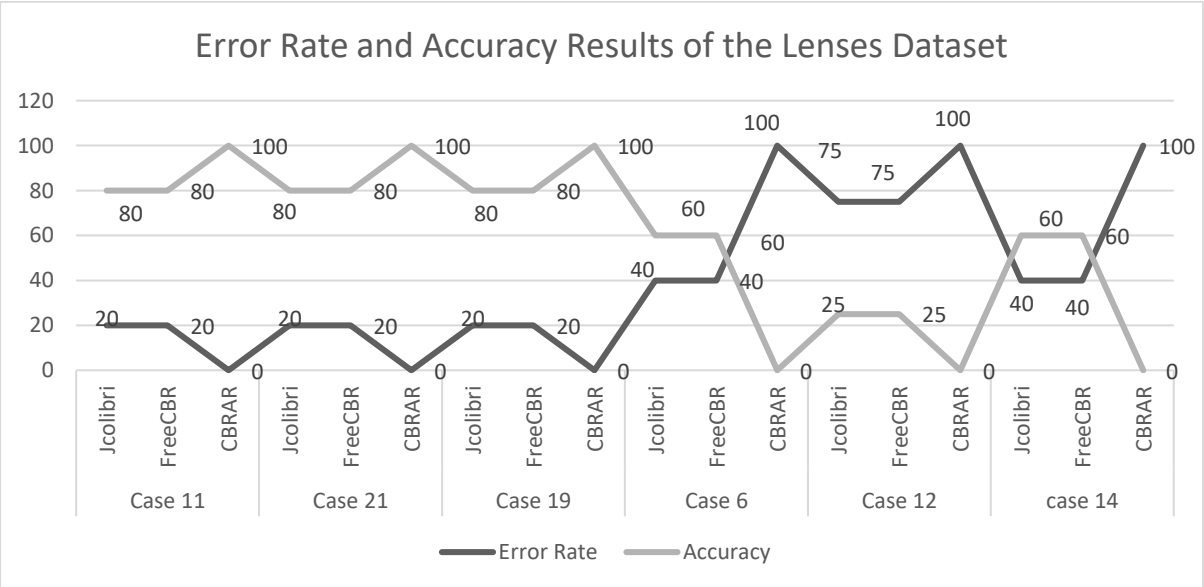


Figure 61 Error Rate and Accuracy Results Assembled of the Lenses Dataset

4.10. Summary

This chapter has presented the results of the experiments and evaluations conducted on various datasets. The objective of this chapter was demonstrated the better performance of the new strategy i.e. CBRAR to assist the decision maker when choosing a correct case. The main part of this corpus was started consecutively with dataset characteristics, experiments and discussions to explain the productivity of this research. In addition, the empirical evaluation was carried out by comparing the performance of CBRAR versus two CBR tools i.e. Jcolibri and FreeCBR. The overall accuracy obtained was 75%, 80%, 100 75% and 50% by finding and disambiguating the wrongly retrieved answers. That comparison has shown the novelty and main contribution of the proposed system by emphasizing its efficiency in achieving a highest accuracy in the implemented experiments.

A conclusion will be presented in the next chapter to summarise the objectives and achievements made in this research containing the results gained through the CBRAR experiments. Given some of the limitations of this research, a summary of possible future work will also be suggested.

Chapter 5: Conclusion and Future Work

CBR is an important part of the artificial intelligence field for making significant decisions computationally. The solution is usually made through four main phases: retrieve, reuse, revise and retain. Among these phases, retrieval is an essential challenge as retrieving a wrong case could lead to taking a wrong decision. Thus, the performance of CBR depends on a strategy that ignores other types of knowledge and enhancing performance and accuracy by employing a powerful technique of association knowledge.

In the scope of the data mining, association rules mining is a popular research area that focuses on finding a correlation between items and their solution. In the state of the art, a number of various methods have been addressed in an effort to find out this correlation, for example approximating the usefulness of ARM with respect to the target problem. Few researchers have addressed their attempts towards integrating a data mining method into CBR which can influence the performance the latter positively.

This thesis explores the use of data mining approaches i.e. CARs to improve the accuracy of CBR retrieval performance by developing a new retrieval strategy. The main aim of the enhanced strategy is to retrieve the correct case not just the most similar one by directing the end user to disambiguate the results.

In this chapter, a summary of how the research aim and objectives have been addressed are displayed in section 5.1. with the accomplishments of this research. The limitation of the enhanced strategy and recommendations for future work are given in section 5.2.

5.1. A Revisit of the research objectives

While the aim of the research was to build a new strategy to improve the performance of similarity based reasoning, CBRAR has shown a better performance compared to the CBR standard tools i.e. Jcolibri and FreeCBR. This section displays the research objectives and revises the magnitude to which they have been fulfilled.

1. **To carry out an in depth, comprehensive literature review on the existing DM techniques especially ARs, and their application into CBR.** A literature review of DM methods has been conducted in chapter 2, with classification approach and their usage in the data mining area also surveyed with the study field. A general KNN for the classification and similarity measured distance is addressed in the literature with metrics that are related to CBR. Chapter 2 also explores an in-depth survey of the ARs techniques that examine strategies and their impact on items correlation. The literature review has emphasized that there are a number of Potential ARs methods such as frequent pattern approaches which can be utilised to obtain a super pattern from the Rules. It also contains a study of tree structures such as the partial tree to compensate for missing nodes in a given tree. However, none of these methods focus on integrating CARs into CBR to gain a better performance.

2. **To review literature on existing CBR systems, and identify how CBR and ARM can be merged together into this type of study.** A survey of the CBR background has been carried out, with methods and its component parts also presented. The research has included identifying the problem area of integrating ARM in CBR. It has presented the background work that has been carried out and a survey of related work. The survey showed that although much work has been carried out into integrating DM techniques into CBR, very little has been done on integrating ARM into CBR. Given the success

of CARs as one of ARs approaches, using them for SBR has demonstrated a promising new direction for better performance of CBR systems, and proved that the remaining aims of this research can be fulfilled for this research target.

3. **To develop a CBRAR based on a strategy that is able to retrieve the most similar case by integrating CARs into CBR.** Chapter 3 constructs a new model CBRAR with the ability to propose the most correct case when irrelevant and/or ambiguous cases are retrieved by the CBR. CBRAR has used CARs to overcome this problem by discovering the correlation between a case target and case library. CBRAR was able to disambiguate wrongly retrieved answers by improving the SK using AK, whereas some studies were much reliant on the specialists to discover SK. Some strategies have reproduced a retrieval process by giving an estimated percentage of related cases but do not contain a feedback to the system. By contrast, CBRAR generates a correct pattern to be sent back to the retrieval step to remove uncertain answers.

   Sometimes, CBR system retrieves two different labels with same similarity when one case is removed from the case library. In this situation, CBRAR adopts CARs to generate FP-CAR tree that is able to consider class label and the length of sub-patterns that were produced by this tree. The developed FP-tree was able to make the produced rules more effective to one class by compressing the rules to its root. The CBRAR strategy used FP-CAR in order to classify subsets according to their recurrence before the rules are generated. Therefore, the proposed strategy was able to compare a new case as a pattern within the built tree patterns in order find a correct match. Furthermore, the new strategy has the advantage over the CBR tools by splitting the rules into different classes. This is achieved by comparing a new case problem with the FP-CAR algorithm and considering the value of each node and its longest length to find a partial match and

then invoking the partial tree in order to compensate the missing nodes if it is necessary to construct an equivalent pattern to the CBR query. In the last stage, the proposed model was able to select a correct answer from the outcomes of the wrongly retrieved case. In this way, the returned solved case by the CBRAR was able to remove the ambiguity of SBR outcomes.

4. **To develop an FP-CAR algorithm that is able to mine CARs into frequent tree to produce a pattern which matches a new CBR case problem.** Section 3.2. displays a new and novel powerful algorithm FP-CAR that aims to form an optimum tree that can be compared with a target case. The approach is based on FP-Growth concepts to obtain enough patterns using CARs for each target case and seeking for combining rules as patterns as frequent tree in order to be compared with a new case problem. FP-CAR is developed each frequent CAR more active to one class. The determination of matching a target case with built tree of CARs reduces the number of unrelated cases in considering that each combined rules belong to a specific class.

   The approach adapted in this thesis is a powerful method of classification based on association using total from partial trees and the union of two rules to gain a potential super-pattern to cover a range of target cases. The proposed algorithm in this thesis is novel in that it compresses the CARs into a prefix tree where the root of a class holds the frequent rules as well as pruning the unnecessary compressed rules to provide a target pattern until no candidate can be generated. It has been proved that all target cases can be reworked in FP-CAR in each tested dataset.

5. **To implement this strategy on real datasets whilst carrying out an empirical evaluation of the system.** In the sub-sections of 4.3. , the first 3 experiments were detected

as a heuristic method which led the research into fulfilling this research aim. Acute inflammation of urinary bladder was used as a real dataset to generate such rules that could lead to potential satisfactory outcomes of the FP-tree.

In the first attempt Ex1, the number of rules produced by Predictive Apriori was low when considering a class label in the implication process. In other words, 7 rules were not adequate to produce an optimum tree for traversal purposes. In addition, some rules tend to have a class label accompanied by item so that did not lead to any heuristic methodology and no pattern matching was achieved.

The second attempt of Ex2, the above mentioned dataset using CARs method was used to gain a reasonable number of rules which could lead to better results. 92 rules were a conventional number in order to build a comparable FP-tree for the target problem where each of the compressed rules belongs to a specific class. However, the representation of the classed rules did not seem to be pattern matching because the nodes do not contain a value which could give more accurate results. Therefore, the FP-Tree was unaccomplished according to the hash table.

Ex3 was an extension of Ex2, where advantage was taken from the modification performed on the FP-tree nodes to hold a value. This consideration of the node's values made the comparison possible between the newly built tree and the CBR query i.e. case problem. The same 92 rules used in Ex2 become more descriptive patterns when they formed a FP-CAR tree that contains classified rules considering nodes values. Therefore, this heuristic method has addressed the drawback of Ex2 and gaining mutual patterns becomes accomplishable which motivated further experiments.

6. **To conduct an empirical evaluation of the new strategy against existing systems such as Jcolibri and FreeCBR and to measure accuracy in terms of retrieving the best similar cases.** A system was created in Java to examine the workability of the

model and test functionality by evaluating whether correctly retrieved cases can be chosen from the conducted experiments or not, were demonstrated in Chapter 4. The viability of the strategy was explored and demonstrated on various types of datasets i.e. health diagnosis, space, cognitive psychology and post- operative, where a dataset contains multi-values of attribute as an advantage when compared with FP-tree as it accepts just binary dataset. In section 4.5. , an empirical evaluation of the proposed system was illustrated, where the results acquired are solving 3 out of 4 cases with an overall accuracy of 75% and 25% error rate. The evolved strategy has proved experimentally its ability and effectiveness in assessing whether the uncertainty of the retrieved cases can be removed in the CBR. In section 4.6. , a second empirical evaluation for the space field was carried out and obtained an overall accuracy of 80% and 20% error rate, where CBRAR resolved 4 out of 5 cases. Furthermore, the empirical findings in section 4.7. make a noteworthy contribution when the CBRAR solved all wrongly retrieved cases when a problem case can be derived from the FP-CAR tree without invoking the P-trees. The findings of section 4.8. complement those earlier experiments where the CBRAR solved 3 out of 4 cases giving 75% of accuracy. Lastly, the results shown in section 4.9. shows an advantage of CBRAR when resolved 3 out of 6 cases giving 50% of accuracy. Taken together, these results demonstrate clearly the novelty and contribution of the CBRAR developed in this research over existing CBR tools. In particular, by showing that: (i) CBRAR is able to assist the decision maker to disambiguate wrongly retrieved answers computationally without any need to experts. (ii) Selecting not just the most similar cases but the correct one from the outcomes. (iii) The proposed system significantly outperforms both SBR and a well-known retrieval method of adopting the concept of similarity in different applications of CBR tools.

5.2. Limitations and Future work

The integration of class association rules into case based reasoning for enhancing the performance of similarity based retrieval, is one of the first of its kind. Inevitably, some aspects could be developed in the future, including the following:

1- **Employing another association rules mining approach**

Although several different ARs algorithms are available in the data mining field, the strategy and method recommended in this research has made use of Apriori, which has led to an improvement in the CBR performance. The benefit of this approach is to generate such rules, that could be utilized to construct an optimum FP-tree, to be compared with a case problem. However, negative ARs also consider items the same as positive rules, but these rules which may represent some items are absent from the implication. Therefore, future work could include experiments using different algorithms, encompassing **[Research on Association Rules Mining Based on Positive and Negative Items of FP-tree]** and **[Positive and Negative Association Rule Mining Using a Correlation Threshold and Dual Confidence Approach],** which could resolve the ambiguous cases produced by the CBRAR.

2- **The Number of Class Association Rules (CARs):**

Although the developed strategy was applied successfully on five datasets, there are some limitations. Specifically, there are some datasets in which the proposed strategy has not resolved the problem of unrelated cases. As described in chapter 4, the FP-CAR algorithm resolved 3 out of 6 cases, which is only 50% accuracy. This is due to the small number of CARs, which were inadequate to generate an optimum tree. To address this in future work, more research is required on the other types of CARs algorithms to generate more useful rules. For example, CAR-Miner and GARC can be employed to produce more CARs which can be linked to CBR cases as patterns.

3- **The Domain of the Datasets**:

The research is implemented on various domains of datasets (such as health diagnosis, space and cognitive psychology) because these datasets were available and obtained from the public UCI repository website. However, it would be better to include other real datasets from the field of law and industry. For example, a lawyer may use case based reasoning when deciding or deliberating a case based on legal precedents so this information would be extremely useful. Or in the case of industry, an issue could be solved by looking at the solutions to past problems.

4- **Implementing the proposed strategy CBRAR:**

The data available on the UCI repository website which was used in this research, is not regarded as big data. Examples of industries that use big data, include banking, public health and learning services. Health services for example, use big data to store vital information about a patient's background or medical history and access to this information can be used to determine or decide on a treatment. Using big data in future work could provide a deeper understanding of past problems whilst offering solutions.

In future work, CBRAR may also be used for cases with multifaceted structures such as those that are hierarchical, object-oriented and / or semantic web-based. However, for CBRAR to run with these cases, two issues need to be addressed. The first is how to define similarity measures for the cases, one such method is described in [158]. Secondly, is how to formalize AK from the cases by attempting to leverage the algorithms proposed in [159], [160], and [161]. Moreover, the adaption of CBRAR for cases with more than one solution could be explored. Along with the performance of CBRAR using different measurements such as computation time and memory used.

References:

[1] A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches," *AI Commun.*, vol. 7, no. 1, pp. 39–59, 1994.

[2] R. Lopez De Mantaras *et al.*, "Retrieval, reuse, revision and retention in case-based reasoning," *Knowl. Eng. Rev.*, vol. 20, no. 3, pp. 215–240, 2005.

[3] P. Perner, "Introduction to Case-Based Reasoning for Signals and Images. Case-Based Reasoning on Signals and Images," in *Case-Based Reasoning on Images and Signals*, P. Perner, Ed. Springer Verlag, 2008, pp. 1–24.

[4] D. H. Yang, J. H. Kang, Y. B. Park, Y. J. Park, H. S. Oh, and S. B. Kim, "Association rule mining and network analysis in oriental medicine," *PLoS One*, vol. 8, no. 3, p. e59241, 2013.

[5] B. Ma, W. Liu, and Y. Hsu, "Integrating classification and association rule mining," in *Proceedings of the 4th Knowledge Discovery and Data Mining*, 1998.

[6] G. Chen, H. Liu, L. Yu, Q. Wei, and X. Zhang, "A new approach to classification based on association rule mining," *Decis. Support Syst.*, vol. 42, no. 2, pp. 674–689, 2006.

[7] B. Vo and B. Le, "A novel classification algorithm based on association rules mining," in *Knowledge Acquisition: Approaches, Algorithms and Applications*, D. Richards and B.-H. Kang, Eds. Springer, 2009, pp. 61–75.

[8] H. Deng, G. Runger, E. Tuv, and W. Bannister, "CBC: An associative classifier with a small number of rules," *Decis. Support Syst.*, vol. 59, pp. 163–170, 2014.

[9] L. T. T. Nguyen, B. Vo, T.-P. Hong, and H. C. Thanh, "CAR-Miner: An efficient algorithm for mining class-association rules," *Expert Syst. Appl.*, vol. 40, no. 6, pp. 2305–2311, 2013.

[10] S. P. S. Ibrahim, K. R. Chandran, and C. J. K. Kanthasamy, "CHISC-AC: Compact Highest Subset Confidence-Based Associative Classification[1]," *Data Sci. J.*, vol. 13, pp. 127–137, 2014.

[11] L. T. T. Nguyen and N. T. Nguyen, "An improved algorithm for mining class association rules using the difference of Obidsets," *Expert Syst. Appl.*, vol. 42, no. 9, pp. 4361–4369, 2015.

[12] J. R. Quinlan, *C4.5: programs for machine learning*. San Mateo, Calif: Morgan Kaufmann, 1993.

[13] M. R. Tolun and S. M. Abu-Soud, "ILA: an inductive learning algorithm for rule extraction," *Expert Syst. Appl.*, vol. 14, no. 3, pp. 361–370, 1998.

[14] M. R. Tolun, H. Sever, M. Uludag, and S. M. Abu-Soud, "ILA-2: An inductive learning algorithm for knowledge discovery," *Cybern. Syst.*, vol. 30, no. 7, pp. 609–628, 1999.

[15] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, 1994, vol. 1215, pp. 487–499.

[16] Z. P. Ogihara, M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, "New algorithms for fast discovery of association rules," in *In 3rd Intl. Conf. on Knowledge Discovery and Data Mining*, 1997.

[17] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *ACM SIGMOD Record*, 2000, vol. 29, no. 2, pp. 1–12.

[18] L. Cagliero and P. Garza, “Infrequent weighted itemset mining using frequent pattern growth,” *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 4, pp. 903–915, 2014.

[19] “jCOLIBRI | GAIA – Group of Artificial Intelligence Applications.” [Online]. Available: http://gaia.fdi.ucm.es/research/colibri/jcolibri. [Accessed: 19-Aug-2015].

[20] “FreeCBR.” [Online]. Available: http://freecbr.sourceforge.net/index.shtml. [Accessed: 01-Feb-2016].

[21] 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.

[22] 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.

[23] J. L. Castro, M. Navarro, J. M. Sánchez, and J. M. Zurita, “Loss and gain functions for CBR retrieval,” *Inf. Sci. (Ny).*, vol. 179, no. 11, pp. 1738–1750, 2009.

[24] Y.-B. Kang, S. Krishnaswamy, and A. Zaslavsky, “A Retrieval Strategy for Case-Based Reasoning Using Similarity and Association Knowledge,” *IEEE Trans. Cybern.*, vol. 44, no. 4, pp. 473–487, 2014.

[25] U. C. of Madrid, “JCOLIBRI,” 2010. [Online]. Available: https://gaia.fdi.ucm.es/research/colibri/contact-us.

[26] C. R. Kothari, *Research methodology: Methods and techniques*. New Age International, 2013.

[27] T. Bond and C. M. Fox, *Applying the Rasch model: Fundamental measurement in the human sciences*. Routledge, 2015.

[28] E. Kane, *Doing your own research: Basic descriptive research in the social sciences and humanities*. Marion Boyars, 1983.

[29] D. Silverman, *Doing qualitative research: A practical handbook*. SAGE Publications Limited, 2013.

[30] I. Lakatos, J. Worrall, and G. Currie, *The methodology of scientific research programmes: Volume 1: Philosophical papers*, vol. 1. Cambridge University Press, 1980.

[31] S. L. Ting, W. M. Wang, S. K. Kwok, A. H. C. Tsang, and W. B. Lee, “RACER: Rule-Associated CasE-based Reasoning for supporting General Practitioners in prescription making,” *Expert Syst. Appl.*, vol. 37, no. 12, pp. 8079–8089, 2010.

[32] D. Patel, “A Retrieval Strategy for Case-Based Reasoning using USIMSCAR for Hierarchical Case,” *Int. J. Adv. Eng. Res. Technol.*, vol. 2, no. 2, pp. 65–69, 2014.

[33] J. Han, J. Pei, Y. Yin, and R. Mao, “Mining frequent patterns without candidate generation: A frequent-pattern tree approach,” *Data Min. Knowl. Discov.*, vol. 8, no. 1, pp. 53–87, 2004.

[34] F. Coenen, P. Leng, and S. Ahmed, “Data structure for association rule mining: T-trees and P-trees,” *IEEE Trans. Knowl. Data Eng.*, no. 6, pp. 774–778, 2004.

[35] G. Goulbourne, F. Coenen, and P. Leng, “Algorithms for computing association rules using a partial-support tree,” *Knowledge-Based Syst.*, vol. 13, no. 2, pp. 141–149, 2000.

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

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

[38] Y.-B. Kang, A. Zaslavsky, S. Krishnaswamy, and C. Bartolini, “A Knowledge-rich Similarity Measure for Improving IT Incident Resolution Process,” in *Proceedings of the 2010 ACM Symposium on Applied Computing*, 2010, pp. 1781–1788.

[39] F. Lorenzi and F. Ricci, “Case-based recommender systems: a unifying view,” in *Intelligent Techniques for Web Personalization*, B. Mobasher and S. S. Anand, Eds. Berlin: Springer, 2005, pp. 89–113.

[40] 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.

[41] K.-D. Althof, E. Auriol, R. Barlette, and M. Manago, *A Review of Industrial Case Based Reasoning*. Oxford: AI Intelligence, 1995.

[42] M. M. Richter and R. O. Weber, *Case-based reasoning: A textbook*. Springer, 2013.

[43] H. Muñoz-Avila and F. Ricci, *Case-based reasoning research and development*. Springer, 2005.

[44] M. M. Richter and A. Aamodt, “Case-based reasoning foundations,” *Knowl. Eng. Rev.*, vol. 20, no. 3, pp. 203–207, 2005.

[45] P. Thagard, K. J. Holyoak, G. Nelson, and D. Gochfeld, “Analog retrieval by constraint satisfaction,” *Artif. Intell.*, vol. 46, no. 3, pp. 259–310, 1990.

[46] Y. Orbach, M. E. Lamb, K. J. Sternberg, J. M. G. Williams, and S. Dawud-Noursi, “The effect of being a victim or witness of family violence on the retrieval of autobiographical memories,” *Child Abuse Negl.*, vol. 25, no. 11, pp. 1427–1437, 2001.

[47] J. L. Arcos and E. Plaza, “A reflective architecture for integrated memory-based learning and reasoning,” in *Topics in Case-Based Reasoning*, Springer, 1993, pp. 289–300.

[48] B. Li and H. Johan, “3D model retrieval using hybrid features and class information,” *Multimed. Tools Appl.*, vol. 62, no. 3, pp. 821–846, 2013.

[49] J. Kendall-Morwick and D. Leake, “A Study of Two-Phase Retrieval for Process-Oriented Case-Based Reasoning,” in *Successful Case-based Reasoning Applications-2*, Springer, 2014, pp. 7–27.

[50] 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.

[51] V. / Aparna and M. Ingle, “Enriching Retrieval Process for Case Based Reasoning by using Vertical Association Knowledge with Correlation,” *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 2, no. 12, pp. 4114–4117, 2014.

[52] K. Bo and A. Aamodt, “A new approach to applicable memory-based reasoning,” *J. Exp. Theor. Artif. Intell.*, vol. 11, no. 4, pp. 479–496, 1999.

[53] G. Müller, “Workflow Adaptation in Process-oriented Case-based Reasoning,” in *Proceedings of the ICCBR 2015 Workshops. Frankfurt, Germany*, 2015.

[54] S. De Ridder, “Extending Flexible Querying techniques with Evolutionary Fuzzy Case-Based Reasoning,” Universiteit Gent, 2014.

[55] R. S. Lancho *et al.*, “Intelligent Retrieval of AXISYMMETRIC Solid Parts in Machining Process Planning by Case Based Reasoning,” in *2nd International Conference on Modelling, Identification and Control*, 2015.

[56] S. Dalal, V. Athavale, and K. Jindal, “Case retrieval optimization of Case-based reasoning through Knowledge-intensive Similarity measures,” *Int. J. Comput. Appl.*, vol. 34, no. 3, 2011.

[57] S. V. Shokouhi, P. Skalle, and A. Aamodt, “An overview of case-based reasoning applications in drilling engineering,” *Artif. Intell. Rev.*, vol. 41, no. 3, pp. 317–329, 2014.

[58] L. L. Minku, E. Mendes, and B. Turhan, “Data mining for software engineering and humans in the loop,” *Prog. Artif. Intell.*, pp. 1–8, 2016.

[59] D. T. Larose, *Discovering knowledge in data: an introduction to data mining*. John Wiley & Sons, 2014.

[60] J.-M. Adamo, *Data mining for association rules and sequential patterns: sequential and parallel algorithms*. Springer Science & Business Media, 2012.

[61] M. Bolla, *Spectral Clustering and Biclustering: Learning Large Graphs and Contingency Tables*. John Wiley & Sons, 2013.

[62] J. M. Bernardo and A. F. M. Smith, “Bayesian theory.” IOP Publishing, 2001.

[63] S. L. Ang, H. C. Ong, and H. C. Low, “Classification Using the General Bayesian Network,” *Pertanika J. Sci. Technol.*, vol. 24, no. 1, 2016.

[64] L. Rokach and O. Maimon, *Data mining with decision trees: theory and applications*. World Scientific, 2014.

[65] G. Biau and L. Devroye, “The nearest neighbor distance,” in *Lectures on the Nearest Neighbor Method*, Springer, 2015, pp. 13–23.

[66] M. Parsian, *Data Algorithms: Recipes for Scaling Up with Hadoop and Spark*. O’Reilly Media, 2015.

[67] D. W. Aha, D. Kibler, and M. K. Albert, “Instance-based learning algorithms,” *Mach. Learn.*, vol. 6, no. 1, pp. 37–66, 1991.

[68] R. Agrawal and B. Ram, “A Modified K-Nearest Neighbor Algorithm to Handle Uncertain Data,” in *5th International Conference on IT Convergence and Security (ICITCS),* 2015, pp. 1–4.

[69] Q. Zhang, C. Li, P. He, X. Li, and H. Zou, “Irregular partitioning method based K-nearest neighbor query algorithm using mapreduce,” in *Proceedings of 2015 International Symposium on Computers & Informatics*, 2015.

[70] N. Li, H. Kong, Y. Ma, G. Gong, and W. Huai, “Human performance modeling for manufacturing based on an improved KNN algorithm,” *Int. J. Adv. Manuf. Technol.*, vol. 84, no. 1–4, pp. 473–483, 2016.

[71] S. Walter, “The non-Euclidean style of Minkowskian relativity,” *Symb. universe*, pp. 91–127, 1999.

[72] R. Agrawal, T. Imieliński, and A. Swami, “Mining association rules between sets of items in large databases,” in *ACM SIGMOD Record*, 1993, vol. 22, no. 2, pp. 207–216.

[73] C. Zhang and S. Zhang, *Association rule mining: models and algorithms*. Springer-Verlag, 2002.

[74] E. R. Omiecinski, “Alternative interest measures for mining associations in databases,” *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 1, pp. 57–69, 2003.

[75] J. Rauch and M. Šimůnek, “An alternative approach to mining association rules,” in *Foundations of Data Mining and Knowledge Discovery*, Springer, 2005, pp. 211–231.

[76] J. Dongre, G. L. Prajapati, and S. V Tokekar, “The role of Apriori algorithm for finding the association rules in Data mining,” in *International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*, 2014, pp. 657–660.

[77] A. S. Ashok and S. JoreSandeep, “The Apriori algorithm: Data Mining Approaches is to find frequent item sets from a transaction dataset,” *Int. J. Innov. Res. Sci. Eng. Technol.*, vol. 3, 2014.

[78] Xindong Wu and V. Kumar, *The Top Ten Algorithms in Data Mining*. Chapman & Hall/CRC, 2009.

[79] R. Christopher and M. Shoerio, “Priority Model of Apriori Algorithm,” *Int. J. Adv. Res. Comput. Sci. Manag. Stud.*, vol. 3, no. 6, 2015.

[80] L. Geng and H. J. Hamilton, “Interestingness measures for data mining: A survey,” *ACM Comput. Surv.*, vol. 38, no. 3, p. 9, 2006.

[81] T. Scheffer, “Finding association rules that trade support optimally against confidence,” in *Principles of Data Mining and Knowledge Discovery*, Springer, 2001, pp. 424–435.

[82] E. García, C. Romero, S. Ventura, de C. C. Castro, and T. G. K. Calders, “Association rule mining in learning management systems,” in *Handbook of Educational Data Mining*, Chapman & Hall/CRC, 2011, pp. 93–106.

[83] “Machine Learning Project at the University of Waikato in New Zealand.” [Online]. Available: http://www.cs.waikato.ac.nz/ml/index.html. [Accessed: 19-Aug-2015].

[84] D. Bansal and L. Bhambhu, “Execution of APRIORI Algorithm of Data Mining Directed Towards Tumultuous Crimes Concerning Women,” *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 9, pp. 54–62, 2013.

[85] M. Sharma, J. Choudhary, and G. Sharma, “Evaluating the performance of apriori and predictive apriori algorithm to find new association rules based on the statistical measures of datasets.,” *Int. J. Eng. Res. Technol.*, vol. 1, no. 6, 2012.

[86] D. L. Sampson, T. J. Parker, Z. Upton, and C. P. Hurst, “A comparison of methods for classifying clinical samples based on proteomics data: a case study for statistical and machine learning approaches,” *PLoS One*, vol. 6, no. 9, p. e24973, 2011.

[87] F. P. Pach, A. Gyenesei, and J. Abonyi, “Compact fuzzy association rule-based classifier,” *Expert Syst. Appl.*, vol. 34, no. 4, pp. 2406–2416, 2008.

[88] A. R. Bateman, N. El-Hachem, A. H. Beck, H. J. W. L. Aerts, and B. Haibe-Kains, “Importance of collection in gene set enrichment analysis of drug response in cancer cell lines,” *Sci. Rep.*, vol. 4, 2014.

[89] P. Stanišić and S. Tomović, “Apriori multiple algorithm for mining association rules,” *Inf. Technol. Control*, vol. 37, no. 4, 2015.

[90] P.-N. Tan, *Introduction to Data Mining*. Pearson, 2005.

[91] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Pearson, 2013.

[92] A. Bhandari, A. Gupta, and D. Das, “Improvised Apriori Algorithm using frequent pattern tree for real time applications in data mining,” *Procedia Comput. Sci.*, vol. 46,

pp. 644–651, 2015.

[93] S. O. Fageeri, R. Ahmad, and B. B. Baharudin, “A semi-apriori algorithm for discovering the frequent itemsets,” in *International Conference on Computer and Information Sciences (ICCOINS)*, 2014, pp. 1–5.

[94] Q. Zhao and S. S. Bhowmick, “Association rule mining: A survey,” *Nanyang Technol. Univ. Singapore*, 2003.

[95] A. Savasere, E. R. Omiecinski, and S. B. Navathe, “An efficient algorithm for mining association rules in large databases,” *Coll. Comput. Tech. Reports*, vol. 488, 1995.

[96] F. Coenen, G. Goulbourne, and P. Leng, “Tree structures for mining association rules,” *Data Min. Knowl. Discov.*, vol. 8, no. 1, pp. 25–51, 2004.

[97] H. Toivonen, “Sampling large databases for association rules,” in *VLDB*, 1996, vol. 96, pp. 134–145.

[98] M. H. Mohamed and M. M. Darwieesh, “Efficient mining frequent itemsets algorithms,” *Int. J. Mach. Learn. Cybern.*, vol. 5, no. 6, pp. 823–833, 2014.

[99] R. Rymon, “Search through systematic set enumeration,” *Tech. Reports*, p. 297, 1992.

[100] I. N. M. Shaharanee and J. Jamil, “Evaluation and optimization of frequent association rule based classification,” *Asia-Pacific J. Inf. Technol. Multimed.*, vol. 3, no. 1, pp. 1–13, 2014.

[101] F. Coenen, P. Leng, A. Pagourtzis, W. Rytter, and D. Souliou, “Improved Methods for Extracting Frequent Itemsets from Interim-Support Trees,” in *Research and Development in Intelligent Systems XXII*, Springer, 2006, pp. 263–276.

[102] J. Zhou and K.-M. Yu, “Tidset-based parallel FP-tree algorithm for the frequent pattern mining problem on PC clusters,” in *Advances in Grid and Pervasive Computing*, Springer, 2008, pp. 18–28.

[103] A. Mangalampalli and V. Pudi, “Fuzzy association rule mining algorithm for fast and efficient performance on very large datasets,” in *IEEE International Conference on Fuzzy Systems. FUZZ-IEEE*, 2009, pp. 1163–1168.

[104] S. Banerjee and A. R. Chowdhury, “Case Based Reasoning in the Detection of Retinal Abnormalities Using Decision Trees,” *Procedia Comput. Sci.*, vol. 46, pp. 402–408, 2015.

[105] M. Das Gupta and S. Banerjee, “Similarity Based Retrieval In Case Based Reasoning For Analysis Of Medical Images,” *Proceeding ICIPACV*, 2014.

[106] Z. Li, X. Zhou, W. Liu, Q. Niu, and C. Kong, “A similarity-based reuse system for injection mold design in automotive interior industry,” *Int. J. Adv. Manuf. Technol.*, pp. 1–13, 2016.

[107] F. Liu *et al.*, “A similarity-based method for three-dimensional prediction of soil organic matter concentration,” *Geoderma*, vol. 263, pp. 254–263, 2016.

[108] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, “From data mining to knowledge discovery in databases,” *AI Mag.*, vol. 17, no. 3, p. 37, 1996.

[109] A. A. Freitas, *Data mining and knowledge discovery with evolutionary algorithms*. Springer Science & Business Media, 2013.

[110] K. Bradley and B. Smyth, “Personalized information ordering: a case study in online recruitment,” *Knowledge-Based Syst.*, vol. 16, no. 5, pp. 269–275, 2003.

[111] C. M. Vong, P. K. Wong, and W. F. Ip, “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*, 2010, pp. 17–22.

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

[113] 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.

[114] P. Perner, “Prototype-based classification,” *Appl. Intell.*, vol. 28, no. 3, pp. 238–246, 2008.

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

[116] Y. Park, B. Kim, and S. 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.

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

[118] T. M. Mitchell and T. Michell, “Machine LearningMcGraw-Hill Series in Computer Science.” McGraw-Hill Higher Education, 1997.

[119] E. Alpaydin, *Introduction to machine learning*. MIT Press, 2014.

[120] M. Xue and C. Zhu, “A study and application on machine learning of artificial intellligence,” in *International Joint Conference on Artificial Intelligence. JCAI’09*, 2009, pp. 272–274.

[121] S. Marsland, *Machine learning: an algorithmic perspective*. CRC Press, 2015.

[122] S. Minton, *Machine learning methods for planning*. Morgan Kaufmann, 2014.

[123] 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, 1999.

[124] V. Balakrishnan, M. R. Shakouri, and H. Hoodeh, “Integrating association rules and case-based reasoning to predict retinopathy,” *Maejo Int. J. Sci. Technol.*, vol. 6, no. 3, 2012.

[125] D. B. Leake, “CBR in context: the present and future. Case based reasoning experiences-lessons and future experiences. D. Leake.” Cambridge, MIT Press, 1996.

[126] A. Waheed and H. Adeli, “Case-based reasoning in steel bridge engineering,” *Knowledge-Based Syst.*, vol. 18, no. 1, pp. 37–46, 2005.

[127] V. Dufour-Lussier, F. Le Ber, J. Lieber, and E. Nauer, “Automatic case acquisition from texts for process-oriented case-based reasoning,” *Inf. Syst.*, vol. 40, pp. 153–167, 2014.

[128] I. Jurisica, J. Mylopoulos, J. Glasgow, H. Shapiro, and R. F. Casper, “Case-based reasoning in IVF: Prediction and knowledge mining,” *Artif. Intell. Med.*, vol. 12, no. 1, pp. 1–24, 1998.

[129] S. Chattopadhyay, S. Banerjee, F. A. Rabhi, and U. R. Acharya, “A Case-Based

Reasoning system for complex medical diagnosis," *Expert Syst.*, vol. 30, no. 1, pp. 12–20, 2013.

[130] W. He and F.-K. Wang, "Integrating a case-based reasoning shell and Web 2.0: design recommendations and insights," *World Wide Web*, vol. 19, no. 6, pp. 1231–1249, 2016.

[131] E. Abdrabou and A.-B. Salem, "Case-based reasoning tools from shells to object-oriented frameworks," 2008.

[132] M. A. Mohammed, B. Al-Khateeb, and D. A. Ibrahim, "Case based Reasoning Shell Frameworkas Decision Support Tool," *Indian J. Sci. Technol.*, vol. 9, no. 42, 2016.

[133] J. A. Recio-García, P. A. González-Calero, and B. Díaz-Agudo, "jcolibri2: A framework for building Case-based reasoning systems," *Sci. Comput. Program.*, vol. 79, pp. 126–145, 2014.

[134] K.-D. Altho, R. Barletta, M. Manago, and E. Auriol, "A Review of Industrial Case-Based Reasoning Tools. AI Intelligence." Oxford, 1995.

[135] D. P. Roy and B. Chakraborty, "Case-based reasoning and some typical applications," *Glob. trends Intell. Comput. Res. Dev.*, pp. 229–267, 2013.

[136] I. Pegler, C. J. Price, and I. Watson, "Caspian: A freeware case-based reasoning shell," in *Proceedings of the 2nd UK Workshop on Case-Based Reasoning. Watson, I.(Ed.), Salford University, Salford, UK*, 1996.

[137] I. Watson and F. Marir, "Case-based reasoning: A review," *Knowl. Eng. Rev.*, vol. 9, no. 4, pp. 327–354, 1994.

[138] J. A. Recio-garcía, P. A. González-calero, and B. Díaz-agudo, "j colibri2 : A framework for building Case-based reasoning systems ☆," *Sci. Comput. Program.*, vol. 79, pp. 126–145, 2014.

[139] E. C. Lopes and U. Schiel, "Integrating context into a criminal case-based reasoning model," in *Information, Process, and Knowledge Management, 2010. eKNOW'10. Second International Conference on*, 2010, pp. 37–42.

[140] E. A. M. L. Abdrabou and A.-B. M. Salem, "A breast cancer classifier based on a combination of case-based reasoning and ontology approach," in *Computer Science and Information Technology (IMCSIT), Proceedings of the 2010 International Multiconference on*, 2010, pp. 3–10.

[141] A. Stahl and T. R. Roth-Berghofer, "Rapid prototyping of CBR applications with the open source tool myCBR," in *European Conference on Case-Based Reasoning*, 2008, pp. 615–629.

[142] K. Bach and K.-D. Althoff, "Developing case-based reasoning applications using mycbr 3," in *International Conference on Case-Based Reasoning*, 2012, pp. 17–31.

[143] A. Atanassov and L. Antonov, "Comparative analysis of case based reasoning software frameworks jCOLIBRI and myCBR," *J. Univ. Chem. Technol. Metall.*, vol. 47, no. 1, pp. 83–90, 2012.

[144] S. Bogaerts and D. B. Leake, "Increasing AI Project Effectiveness with Reusable Code Frameworks: A Case Study Using IUCBRF.," in *FLAIRS Conference*, 2005, pp. 2–7.

[145] U. Y. Nahm and R. J. Mooney, "Using soft-matching mined rules to improve information extraction," *Language (Baltim).*, vol. 11, p. 50, 2004.

[146] A. Stahl, "Learning of knowledge-intensive similarity measures in casebased reasoning," PhD Thesis, Univ. Kaiserslautern, 2003.

[147] D. Wang, T. Li, S. Zhu, and Y. Gong, “iHelp: An intelligent online helpdesk system,” *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, vol. 41, no. 1, pp. 173–182, 2011.

[148] R. J. Bayardo Jr and R. Agrawal, “Mining the most interesting rules,” in *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mning*, 1999, pp. 145–154.

[149] L. T. T. Nguyen, B. Vo, T.-P. Hong, and H. C. Thanh, “Classification based on association rules: A lattice-based approach,” *Expert Syst. Appl.*, vol. 39, no. 13, pp. 11357–11366, 2012.

[150] “TFPC Classification Association Rule Mining (CARM) Software.” [Online]. Available: https://cgi.csc.liv.ac.uk/~frans/KDD/Software/Apriori-TFPC/Version2/aprioriTFPC.html. [Accessed: 15-Dec-2015].

[151] A. S. Aljuboori, F. Meziane, and D. J. Parsons, “A new strategy for case-based reasoning retrieval using classification based on association,” in *12th MLDM International Conference*, 2016, p. pp 326-340.

[152] A. Aljuboori, “Enhancing case-based reasoning retrieval using classification based on associations,” *2016 6th International Conference on Information Communication and Management (ICICM)*. pp. 52–56, 2016.

[153] S. Epp, *Discrete mathematics with applications*. Cengage Learning, 2010.

[154] F. Provost and R. Kohavi, “Guest editors’ introduction: On applied research in machine learning,” *Mach. Learn.*, vol. 30, no. 2, pp. 127–132, 1998.

[155] X. Deng, Q. Liu, Y. Deng, and S. Mahadevan, “An improved method to construct basic probability assignment based on the confusion matrix for classification problem,” *Inf. Sci. (Ny).*, 2016.

[156] M. Buckland and F. Gey, “The relationship between recall and precision,” *J. Am. Soc. Inf. Sci.*, vol. 45, no. 1, p. 12, 1994.

[157] M. Sokolova, N. Japkowicz, and S. Szpakowicz, “Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation,” in *AI 2006: Advances in Artificial Intelligence*, Springer, 2006, pp. 1015–1021.

[158] S.-H. Cha, “Comprehensive survey on distance/similarity measures between probability density functions,” *City*, vol. 1, no. 2, p. 1, 2007.

[159] V. Nebot and R. Berlanga, “Mining association rules from semantic web data,” *Trends Appl. Intell. Syst.*, pp. 504–513, 2010.

[160] J. L. Mbuke and L. Songfeng, “Mining Frequent Patterns on Object-Relational Data,” *IOSR Journals (IOSR J. Comput. Eng.*, vol. 1, no. 18, pp. 52–59.

[161] P. Gautam and K. R. Pardasani, “Algorithm for efficient multilevel association rule mining,” *Int. J. Comput. Sci. Eng.*, vol. 2, no. 5, p. 2010, 1700.