Elsevier Editorial System(tm) for Expert Systems With Applications + OA Mirror
Manuscript Draft

Manuscript Number: ESWA-D-19-01619

Title: A Case-Based Reasoning for Supervised Classification Problems in Mammographic Applications

Article Type: Full length article

Keywords: case-based reasoning (CBR); mammographic mass; cases randomization; supervised classification problem; cases validation

Corresponding Author: Mrs. Miled Basma Bentaiba-Lagrid, Ph.D. student

Corresponding Author's Institution: ESI Algiers

First Author: Miled Basma Bentaiba-Lagrid, Ph.D. student

Order of Authors: Miled Basma Bentaiba-Lagrid, Ph.D. student; Lydia Bouzar-Benlabiod, Doctor; Stuart H Rubin, Professor; Thouraya Bouabana-Tebibel, Professor

Research Data Related to this Submission

Title: Mammographic Mass Data Set
Repository: Mammographic Mass Data Set
http://archive.ics.uci.edu/ml/datasets/mammographic+mass

A Case-Based Reasoning for Supervised Classification Problems in Mammographic Applications

Miled Basma Bentaiba-Lagrid[1](⊠), Lydia Bouzar-Benlabiod[2], Stuart H. Rubin[3], [4]thouraya Bouabana-Tebibel

[1,2,4] Laboratoire LCSI, Ecole nationale Supérieure d'Informatique of Algiers (ESI), BP 68M -16309, Oued Smar, Algiers, Algeria
[1]bm_bentaiba@esi.dz [2]l_bouzar@esi.dz [4]t_tebibel@esi.dz
[3]Naval Information Warfare Center (NIWC), San Diego, CA 92152–6423 USA
[3]stuart.rubin@navy.mil
(⊠) Corresponding Author

Abstract

Case-Based Reasoning (CBR) is the process of solving new problems based on previous experiences. It relies on reuse by bringing previously solved problems into its case base and using them to solve new ones. A static and non-evolutive case base doesn't allow the CBR to be accurate in problem-solving. Also, a massive case base can affect the resolution time. Randomization represents a way to minimize the spatial image of the case base and thus search time as well. However, the cases generated by randomization are not necessarily valid and they need to be validated before use. In this paper, a new randomization technique to amplify the case base is presented, where the case base is segmented in a way that speeds cases retrieval, while supporting cases retention. The generated data is validated through three layers: coherence verification, stochastic validation, and absolute validation. Furthermore, we propose a new way to segment the case base along with new similarity functions based on features' weights to speed CBR retrieval. We carried out experiments to classify the severity of mammographic mass to validate our approach, where the proposed approach is compared to several popular supervised machine learning algorithms and other related works that utilize the same dataset. Experiments have shown that our approach can generate relevant data, which significantly improves the resolution accuracy and makes CBR a good competitor to classification tools.

Keywords: case-based reasoning (CBR); mammographic mass; cases randomization; supervised classification problem; cases validation

1. Introduction

The concept of problem-solving is frequently used to refer to the inference process of finding a solution for a given problem. Humans use their innate abilities to solve various problems, which they confront daily. One of the intuitive ways used for such a purpose is exploiting the experiences that one has been exposed to beforehand. In fact, one tries to build a suitable solution to a newly encountered problem by adapting solutions from similar experiences. For example, if you are a student and you are familiar with a type of math problems, it will be easier for you to solve similar exercises. Another example is that a doctor can better treat a patient who has symptoms that the doctor has previously treated. CBR imitates this human thinking process by storing the previous experiences in its knowledge base and using similar experiences for solving future problems.

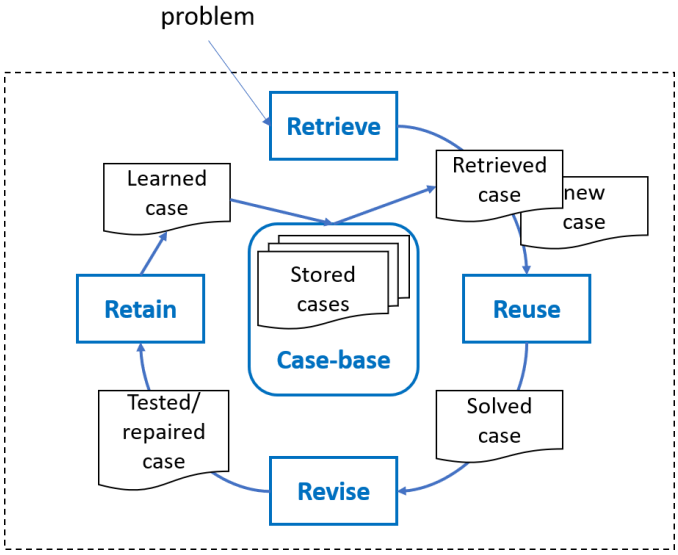


Figure 1: the CBR cycle (Aamodt and Plaza, 1994)

1.1. Case-Based Reasoning

CBR systems incorporate a knowledge-base that contains previously solved problems considered as experiments and referred to as cases. The case base is the heart-engine used to solve newly faced problems, by retrieving the most similar cases. The CBR

process was defined by (Aamodt and Plaza, 1994) as cyclical (Figure 1). This figure contains four processes, called the four REs: (1) REtrieve the most similar case(s); (2) REuse the case(s) to attempt to solve the problem; (3) REvise the proposed solution if necessary; and (4) REtain the new solution as a part of a new case.

CBR is designed to solve problems where experience is needed and there is no possible pattern for it, or it is hardly possible to obtain. In other words, it can solve problems that humans can solve, but it is hard to describe how it is carried out. Thus, the process cannot be written in the algorithmic form and cannot be taught to the machine. Machine Learning (ML) algorithms can do the job too; and, they are designed for this purpose.

Classification, diagnosis and prediction are probably the most common analytic tasks that the CBR can handle (Richer and Weber, 2016). Classification associates an instance to a class in its purest form (Richer and Weber, 2016). Diagnosis is merely a classification problem where the class could be a fault or a disease (Richer and Weber, 2016). While prediction associates an instance with a class representing an event that has not yet occurred (Richer and Weber, 2016). In this paper, we propose an approach to resolve classification problems in general and more specifically, the supervised ones. We are interested in the domains of biology and medicine, which are mainly experience-based. Usually, in such a domain, there are no sure patterns that give the diagnosis, depending on the patient's symptoms. Indeed, we can have two patients having the same symptoms, but not the same disease. This kind of field could be treated either by CBR or by ML algorithms. So why CBR is the most appropriate tool for the medical classification problem?

ML algorithms have generally a training data that is used to learn the pattern (Kotsiantis et al. 2007). This process of learning is applied offline before starting to solve real problems; and, it is not a recurrent one. It is done once, and the resolution of future problems depends on the learning process previously obtained. In contrast, the CBR process for learning is evolutive, continuous and self-referenced, as long as the knowledge is acquired from or outside of the CBR system. CBR problem resolution becomes more and more precise and efficient when it solves problems.

Some ML tools, such as symbolic learning and decision trees (Chen, 1995), suffer from the complexity and the exponential evolution of the need for RAM capacity. Perceptron-based classification (Stephen, 1990), like single layered perceptron and neural networks (Hagan et al., 1996), or the static methods of classification such as the instance-based learning classifiers (De Mantaras and Armengol, 1998), Support Vector Machine (SVM) (Byun and Lee, 2002), Bayesian network (Singh and Valtorta, 1995) … etc., cannot be applied when the knowledge is evolutionary and continuously acquired by the system. Another existing approach for classification is the incremental classification of data streaming (Žliobaitė et al., 2015). In our situation, we don't need to work on the requirements imposed by this classification tool.

Most of the work on CBR, when it is used for medical diagnosis purposes, or breast cancer mainly, are focusing on the case-base maintenance problem. The reason for paying so much interest to the case base is that storing all of the acquired cases in their natural form can certainly highly augment the accuracy of the CBR, but it leads to manage a massive, redundant, and non-optimized case base, which systematically causes deterioration of the CBR resolution time. Medical databases are known to be quite large. Many solutions are proposed to maintain the case base. While the ideas vary in two methods: (i) dividing the case base into smaller case bases containing more homogenous cases. Thus, in the retrieval process, the relevant cases are stored in one small case base with no need to access to the other case bases (Fan et al., 2011; Smiti and Elouedi, 2013), or (ii) shrinking the volume of the case base and reducing it into a smaller size (Smiti and Elouedi, 2018; Yan et al., 2016; Smiti and Elouedi, 2010; Smiti and Elouedi, 2014). In the both ideas, we try as much as possible to keep the competence of the case base, which ensures the accuracy of the CBR.

Another problem that CBR faces is how to retrieve the most useful cases for adaptation to the tackled problem. Note that the most similar cases are not necessarily the most useful ones. Actual tendencies attempt to rank features according to their importance and their impact on the solution (Elter et al. 2007; Huang et al., 2012; Ahn and Kim, 2009; Yan et al., 2016). Other approaches are based on retrieving the most useful cases for a certain problem using optimization methods as genetic algorithms or other intelligent systems (Rezvan et al., 2013; Mazurowski and al., 2008; Quellec et al., 2011).

Although most of the research trends are focusing on case-base maintenance and case retrieval, there is a minority of work that is contributing to (i) retaining the most useful cases without redundancy to keep the case base optimized and reduced (Mazurowski and al., 2008; Yan et al., 2016). It consists of deciding for each captured case, whether it would be stored in the case base or not. (ii) Other works are proposing new attempts to reuse cases to solve active problems (Sharaf-El-Deen et al., 2014; Yan et al., 2016) by adapting them.

1.2. Randomization

Obviously, a rich case base ensures greater efficiency in problem-solving. Hence, the enrichment of the case base implies the improvement of the resolution capacity. As a result, the resolution time may be affected and becomes slower. However, having a static and non-evolving case base limits the system's capacity to solve problems. Thus, we need to find a way to keep the knowledge evolving and reduce space and resolution time and this is the aim of applying randomization in the context of CBR.

Randomization was first defined by G. Chaitin and Klomogorov (Chaitin, 1975) in 1975. The concept underpinning randomization is that information or knowledge can be effectively compressed until the representation of the compressed information is random; or in other words, pattern-less (Rubin, 2007). A non-random information is called symmetric. For example, clearly the sequence "1010101010..." is not random because one can write a simple program to output it. For example, "1010101010..." contains the same information as: “write 01 x times”. When the information is not random, we call it symmetric. There is no proof that can decide whether an arbitrary sequence is random or not. One can only state that it’s more or less random in a relative sense (Rubin, 2007).

Rubin (Rubin, 2007) has proved that effective knowledge acquisition in the large, must be domain-specific and evolutionary, in order to build a system that can learn and acquire new knowledge. Accordingly, the knowledge must be self-referential, which means that the program is able to modify itself. It implies that the system can progressively learn using the stored data. This will be assured by using self-referential transformations, where a transformation could be a logical operation such as generalization, abstraction, explanation, similization, specialization, prediction or dissimilization (Michalski and Myofsciences, 1994), or a randomization operation. Self-referential transformations are transformations applied starting with the data in the system, where the resulting data is fed back to the same system.

The difference between randomization and logical transmutations is that randomization is performed on cases, which are experiences captured from the world. These cases can hang dirty data but can be easily captured. Generating data by randomization allows for an inherent degree of error so as to greatly enlarge the inferential space. The data generated by randomization needs to be validated. Furthermore, logical transmutations are applied to the rules. The latter is always valid, but rarely obtained; and, the output of the logical transmutations is always valid.

According to (Rubin and Bouabana-Tebibel, 2016a), the process of randomization captures existing instances in a more compact form and embodies similar instances, which may or may not be valid. Hence, a step of validation must be performed when the randomization is applied, before manipulating the data for problem-solving. The first-time randomization has been explicitly used in the CBR for knowledge amplification was in (Bouabana-Tebibel et al., 2016a) and many other works followed it (Bouabana-Tebibel et al., 2016b; Chebba et al., 2016; Bouabana-Tebibel et al., 2017; Bentaiba-Lagrid et al., 2018; Bouabana-Tebibel et al., 2018; Bouzar-Benlabiod et al., 2018). The process of knowledge amplification using randomization has been widely used in many pioneering works (Rubin and Bouabana-Tebibel, 2016b; Rubin, 1999; Bouabana-Tebibel et al., 2016a; Rubin et al., 2004; Rubin, 2016; Pedrycz and Rubin, 2010; Rubin and Bouabana-Tebibel, 2016a; Rubin, 1991; Bouabana-Tebibel et al., 2016b; Chebba et al., 2016; Bouabana-Tebibel et al., 2017; Bentaiba-Lagrid et al., 2018; Bouabana-Tebibel et al., 2018; Bouzar-Benlabiod et al., 2018) where a new domain-specific validation after randomization is added.

Although, the fundamental idea behind randomization, which is based on component substitution relative to an appropriate context, remains applicable to many domains. However, its formalization and/or conceptualization may vary from one domain to another. The result depends on the nature of the addressed issues and the form of the provided solutions. The application of randomization has varied from one application domain to another, such as: refrigerator design (Chebba et al., 2016), scheduling systems (Bouabana-Tebibel et al., 2017), traveling robots in (Bouabana-Tebibel et al., 2016a) and (Bouabana-Tebibel et al., 2016b), state-space prediction for computing wartime associates for military strategies in an autonomous way (Rubin and Bouabana-Tebibel, 2016a). The latter was adapted to a cyber context (Rubin and Bouabana-Tebibel, 2016b). Weather prediction (Bichindaritz and Montani, 2011) and mammographic mass (Bentaiba-Lagrid et al., 2018) are application areas for randomization.

1.3. On Randomizing the Case Base

Figure 2 depicts the way in which we amplify a case base through reuse. However, there is another more general way. That is, if the representational formalism for the case base is not fixed, then not only can cases be equated across distinct representations of the same knowledge – enabling the solution of many problems, which otherwise are unsolvable, but a change of case representation often enables the randomization of a case base into one or more coherent rule-base segments. This is how we propose to solve the classic problem of generalization in case bases. That is, the applicability of cases can grow exponentially with insignificant decrease in their validity. An example, borrowed from data science, will serve to illustrate the concept.

Suppose one has radiographic images of Stage II breast cancer and wants to ascertain the degree to which it metastasized, if at all. Given a full-body scan, one may look for co-linear tumor sites. Co-linearity is best-defined along major arteries. Lines running perpendicular to such arteries are far less likely to be indicative of metastasis and in need of biopsy. We would like to automate the scan for co-linear tumors of this type. One can automatically tag the tumor with the name of the body part and superimpose the arterial transport system on top of that. This process allows for a change of case representation. This change serves to highlight possible metastatic progress, while negating data sites, which do not fit the definition of metastasis. Possible metastatic sites have been fused, filtered, and otherwise reorganized to make possible the autonomous probabilistic analysis of each such site involvement. This makes automatic tracking of the cancer spread possible without the need to laboriously go over every square cm of radiographic image by hand. You see, one of the malicious properties of cancer is that when one excises one site, that destroys the introduction of repressor hormones into the bloodstream – enabling the sleeping sites to be activated. Here, we see that the data imaging of tumor cells can be autonomously generalized to probabilistically identify those likely to be a site of

metastatic proliferation – enabling rapid and aggressive localized treatment – especially in patients not able to tolerate another dose of the same or different chemo agent well.

Artificial Intelligence (AI) can be applied to cases for their generalization. Such AI takes the form of a case base and/or a rule base. For example, the knowledge base can paint a picture from the data, which enables recognition by a neural network. Similarly, it can iteratively transform an image or video into a database hold. It can also perform hyperspectral imagery, which can superimpose radiographic data on top of a sonogram to increase the accuracy of (autonomous) diagnostic applications. Positron Emission Tomography (PET) scans can be fused with Computerized Axial Tomography (CAT) scans. Even the perfectly safe laser mammography can be fused, using different laser frequencies and pulse modulation rates (first photons are not diffused by breast tissue), to yield a technique, which is so safe that daily progression studies can be performed with zero additional patient risk with only a slight reduction in the quality of the image in comparison with that obtained from radiographic studies. Furthermore, the fusion process can include ultrasound studies, which will be of great use in the management of cases that need to be closely-watched for collateral organ involvement.

While case and rule bases can generalize each other over a network configuration and this can increase the quality of available diagnostic information, it can also lead to symmetric extensions of the diagnostic knowledge held in the network of knowledge bases. This is possible where each knowledge base can be applied to each other. Validity of results may be insured by reflecting results off of sets of predefined constraints. Indeed, the application of knowledge to extend knowledge serves as the basis for human creativity and learning through randomization (Chaitin, 1975; Rubin, 2007). The creation of such symmetric knowledge bases (Chaitin, 1975) is beyond the scope of this paper. However, an example will serve to illustrate the utility of the process.

First, the human needs to supply basis knowledge. For example, the strength of an electromagnet may be accurately determined by bouncing a laser off of one of its poles and viewing the reflection through an out of phase polarized filter. That is our case knowledge. It can be generalized by transforming the pole of the magnet and rotating the polarized lens 180 deg. The transformational knowledge is supplied by other rule base segments in the heterogeneous and self-referential network. In this example, the generalization takes form under the union operator. In specific cases, the union set can be replaced by a generalized set, which potentially holds a much denser knowledge base than the segments in the union. For example, suppose this process yields two rules pertaining to a mammographic mass having a density index of 1 or 2. A knowledge base segment contains the knowledge that mammographic mass density is continuous (healthy fibrous tissue vs. cancerous tissue is the subject of other knowledge base segments operating on other aspects of the hyperspectral imagery). Here, the two disjoint rules are replaced by one fuzzy rule, which may be a Type II fuzzy rule. This allows another knowledge-based segment to fit a curve to the fuzzy z-function. This capability, broadly applied, defines the future of diagnostic medicine. This capability, theoretically defined by (Rubin, 2007), is unbounded in its capability to outperform human diagnosticians. We will address this new AI in a subsequent paper. It can utilize deep learning as a data feed or work independently. It is not mathematically trivial as is deep learning. Indeed, this paper suggests that in the future, science will play the major role in improving access to health care for the masses – though that may be a somewhat unfamiliar concept today.

1.4. Objectives

By seeing the example found in (Bagni, 2009), Bombelli's uses for the first time, imaginary numbers that lead to finding a resolution of a cubic equation by applying $i^2 = -1$ in it. Note that there is no real number "i" that gives $i^2 = -1$, which means that "i" is an imaginary number. Employing that imaginary number makes the equation possible to solve. The idea behind the provided example is that allowing errors to flow can lead to getting more data that can be valid or invalid. After its validation, we will possibly have more valid data than if we rejected the not validated data in each iteration. Note that when not validated data is manipulated, it must be treated carefully. Included under this assumption to allow for a certain inherent degree of error is to greatly enlarge the inferential space.

The general purpose of this paper is to propose a new approach to amplify the case base using a novel randomization technique in the context of CBR. We were inspired by the assumption given by Bombelli in (Bagni, 2009) to propose a new technique of randomization. The proposed technique will help materialize the implicit knowledge through newly-generated cases, which allows for better reuse of previous experiences by the CBR to precisely solve newly-faced problems. Hence, the CBR will be more accurate in problem resolution. The proposed approach serves case-base amplification without deteriorating the performance of the CBR. The newly generated cases are not necessarily valid. They will be validated through a new three-layer validation process and only coherent cases are integrated within the case-base (segments).

To ensure that the problem-resolution time of the CBR isn't slowed, a new process of maintaining the case base is proposed. It consists of segmenting the case base into smaller segments that are represented by a delegate and by a solution. Instead of comparing the problem with all the existing cases in the case base, we compare only with the delegates. This leads to propose a new similarity functions for the CBR's Retrieve module.

We propose a new algorithm for rules generation. These rules are used in the third layer of validation to give an opinion about the validity of the cases, and in the Reuse module to give a solution to the user's problem.

In our work, we consider the classification problem in general, based on CBR, as a classification tool. We suppose that the classes are known and defined. We tested our approach in the health science domain, and more specifically to classify mammographic mass as benign or malignant, through experimental results, we prove that our approach of CBR is more efficient and that it can compete with the supervised ML algorithms.

The remainder of this paper is structured as follows: Section 2 presents an overview of our approach. Section 3 presents the preliminaries and the specifications of our approach, including the case format. Section 4 is devoted to describing the algorithm for feature selection and weighting. Section 5 presents the algorithm for rules generation. Section 6 presents the case-base maintenance, including how the case base was segmented and how the knowledge is amplified by randomization. Section 7 present our approach for retrieval and reuse. Revise and Retain are presented in Section 8. Section 9 is dedicated to show the results of experiments using our approach and compares it with related work. A prototype is developed for this purpose.

2. Approach Overview

Our approach is based on the classic CBR where the case base is segmented in a way that speeds and improves the retrieval for similar experiences and the reuse process (Figure 2). The case base is amplified with new cases using randomization. The aim of amplifying data is to improve the ratio of correctly answering a user's problem. When the CBR faces a new user's problem, it retrieves the most similar cases and representatives of the most similar delegates of segments. The representative is composed of a delegate and a class of solution. New solutions are provided to the problem either by copying cases/segments solutions or by generating new solutions and testing them using rules. The Revise module is comprised of three layers of validations composed of coherence verification, where the form of the case is verified. All of the stored cases, in the case base, must be at least coherent. Stochastic validation is a process that gives a probability of validity to the case and absolute validation provides a final decision on the validity of the case – either it is valid, or it is not. The coherent cases are retained either by storing it as part of a new case or incrementing its frequency if it already exists.

The rules generation module is a module that generates rules from *fully valid* cases*. Fully valid* cases are cases captured from the real world or cases that the experts have validated. These rules are used either in the Reuse module to give a solution to the user's problem, or in the Revise module to ensure that the newly solved case is valid or not.

Thus, our major contributions to CBR are summarized in these points:

- A new process of the case-base segmentation into small segments to speed the retrieval process and knowledge amplification using randomization processes;
- A new randomization technique to amplify the case base. The aim is to make the system more able to answer the user's problem;
- in the Retrieve module, we propose new similarity functions based on features' weights to retrieve the most useful information for the user's problem;
- A new three-layer process to validate the data generated by randomization. The same process is used in the CBR's Revise module. The bottom layer is based on rules that are generated from *fully valid cases* and updated periodically.
- A new algorithm for rules generation – these rules are used in the Reuse module as well as in the Revise module;
- Implement some of the most popular supervised ML tools using sklearn library (Buitinck et al., 2013) to compare their accuracy to that of our work.

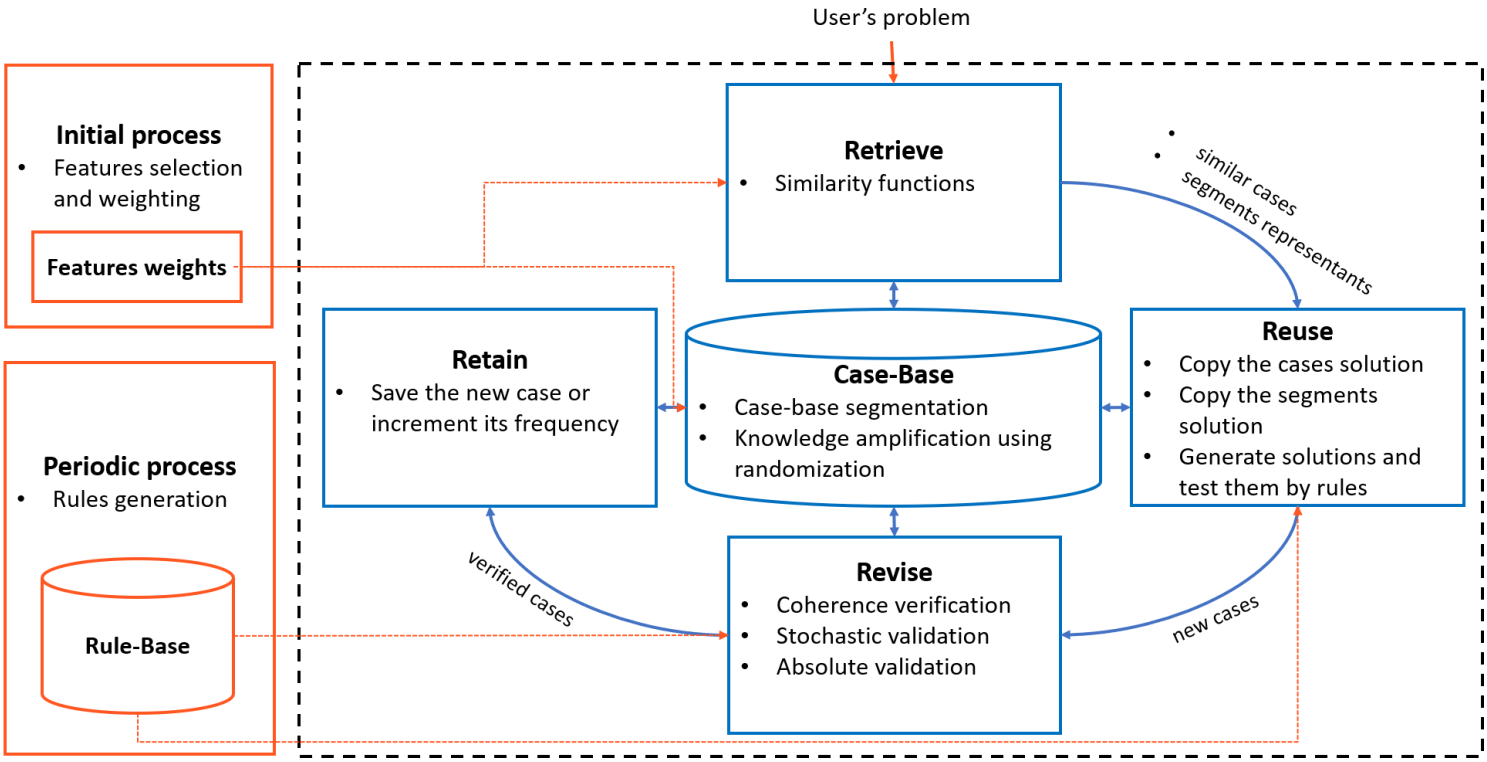


Figure 2: our approach

3. Preliminaries

This section gives background material underlying our approach. We describe the notation used in this paper and form of the case.

3.1. Notation

These are the main notations used to define our approach:

C_j	case j	x	number of classes of solutions
t_i	feature i	Ø	the null value
$v_{j,i}$	the value of the feature t_i in the case C_j	D_f	delegate of the segment f
W_i	t_i feature's weight	abs(x)	the absolute value of x
S_j	solution of the case C_j	$\min^{n}_{i=1}(x_i)$	The minimum value from the sequence $\{x_1, x_2, \ldots, x_n\}$
n	number of features	$\max^{n}_{i=1}(x_i)$	The maximum value from the sequence $\{x_1, x_2, \ldots, x_n\}$
n'	number of relevant features		

3.2. The Case

Each case C_j in the case base consists of a part describing the problem and another part describing the solution. The problem part is a sequence of known and defined features t_n and their values $v_{j,n}$. The features are related to the application domain and could be nominative or quantitative. Cases in the same case base must be related to the same application and have the same features with different values, which may be null. The type of problems we are interested in, are the supervised classification problems. Hence, the solution part may take a value between counted and predefined values called classes. The solution in the case represents a single class. It is possible to have two cases that have the same problem part and a different solution. This is an example of a case:

$$(1)\quad C_j: \{t_1: v_{j,1}, t_2: v_{j,2}, \ldots, t_n: v_{j,n}\} \Rightarrow S_j$$

where $v_{j,1}$, $v_{j,2}$, ..., $v_{j,n}$ are the values of the features t_1, t_2, ..., t_n respectively. S_j is the class of solution of the case C_j.

Example 3.2.1. to define the severity of a mammographic mass, we are using a public dataset of cases taken from patients (Elter et al. 2007). This dataset is provided by *UCI Machine Learning Repository*[1] and contains 961 cases. We have five captured features and two classes of solutions:

- t_1 = BI-RADS: assessment refers to Breast Imaging Reporting And Data System. It's a nominative feature. Its possible values are {0, 1, 2, 3, 4, 5, 6, Ø}. Where 0 means uncomplete assessment, 1 means that the mammogram is definitely benign, until 5 that means that the mammogram is highly suggestive of malignancy.
- t_2 = Age: patient's age in years. A quantitative feature. It could be Ø.
- t_3 = mass shape. A nominative feature. Its possible values are {round, oval, lobular, irregular, Ø}.
- t_4 = mass margin. A nominative feature. Its possible values are {circumscribed, micro-lobulated, obscured, ill-defined, speculated, Ø}.
- t_5 = mass density. A nominative feature. Its possible values are {high, iso, low, fat-containing, Ø}.
- The set of solutions which represent the severity of mammographic mass is {benign, malignant}.

This is a possible case:

$$C_5 = \{t_1: 5, t_2: 67, t_3: lobular, t_4: speculated, t_5: low\} \rightarrow malignant$$

4. Features Selection and Weighting

Features do not have all the same impact on the solution. Some features are more significant than others. In other words, they can change the solution of the case from one class to another after having a small change in their values using the same context. To catch this difference, we propose a novel algorithm that weights the features. This algorithm will be run using a *fully valid case base*.

The weight of each feature t_i is calculated as follows: we compare each two cases (in pairs) and calculate the number of pairs of cases X_i, Y_i, V_i, Z_i that are defined in Table 1. The columns and the rows show the criteria that X_i, Y_i, V_i, Z_i must verify.

When the similarity of values of a feature t_i in a pair of cases is high and the pair has the same solution, this situation could mean that the feature t_i was the reason of having that solution in the cases. Similarly, when having a slight similarity in that feature and the cases have a different solution, could mean that the difference in the values of the feature made the solution changes from a class to another. The quantity $(V_i + Z_i)$ gives the impact of the feature t_i on the solution of the case.

Table 1: number of pairs of cases: X_i, Y_i, V_i, Z_i

	and with highly similar values of t_i	and with slightly similar values of t_i
Pairs of cases with the same solution	V_i	Y_i
Pairs of cases with different solution	X_i	Z_i

Conversely, when the similarity of values of a feature t_i of two cases is high and they don't have the same solution, or when the values of that feature are slightly similar, and the cases have the same solution, these situations could mean that the feature t_i has no influence on the solution of the case. The quantity $(X_i + Y_i)$ tells us to what degree the feature could be insignificant to the solution of the case.

The weight W_i' of the feature t_i is calculated as follow:

$$(2)\quad W_i' = (V_i + Z_i) - (X_i + Y_i)$$

The presented algorithm of calculating weights is systematically applied to all of the features to determine their weights. When $(W_i \leq 0)$, the feature t_i doesn't have an influence on the solution, so it will be fired and ignored in the steps of knowledge-amplification and validation. It is called a *non-relevant feature*. We normalize the resulting weights in the interval of *[0, 1]* using Formula (3).

$$(3)\quad W_i = \frac{w_i' - min_{j=1}^{n}(w_j')}{max_{j=1}^{n}(w_j') - min_{j=1}^{n}(w_j')}$$

These steps are summarized in Algorithm 1:

[1] http://archive.ics.uci.edu/ml/machine-learning-databases/mammographic-masses/

Algorithm 1: features weighting

1. For each feature t_i, i goes from *1* to n:
   1.1. For each pair of cases (C_a, C_b). The values of t_i in C_a and C_b respectively are: $v_{a,i}$ and $v_{b,i}$ respectively
   1.2. Count the number V_i of pairs (C_a, C_b) where $v_{a,i}$ and $v_{b,i}$ are highly similar and C_a and C_b have the same solution
   1.3. Count the number Y_i of pairs (C_a, C_b) where $v_{a,i}$ and $v_{b,i}$ are slightly similar and C_a and C_b have the same solution
   1.4. Count the number X_i of pairs (C_a, C_b) where $v_{a,i}$ and $v_{b,i}$ are highly similar and C_a and C_b have different solution
   1.5. Count the number Z_i of pairs (C_a, C_b) where $v_{a,i}$ and $v_{b,i}$ are slightly similar and C_a and C_b have different solution
   1.6. $W_i' = (V_i + Z_i) - (X_i + Y_i)$
   1.7. Normalize the weight W_i' using formula (3). The result W_i is t_i weight

Example 4.1. by applying the weighting algorithm to the dataset given in Example 3.2.1. The weights of the features before normalizing W_i' and after normalizing W_i are detailed in Table 2:

Table 2: features weights before normalizing W_i', and after normalizing W_i

Feature	Label	W_i'	W_i
BI-RADS assessment	t_1	181059	1.0
Age	t_2	31019	0.1754
Mass shape	t_3	88313	0.4903
Mass margin	t_4	91239	0.5063
Mass density	t_5	-887	0 / ignored

According to Table 2, BI-RADS assessment is the most-heavily weighted feature; hence, it's the one that has the greatest impact on the solution, followed by mass margin, then mass shape, and finally, age. Mass density is a *non-relevant feature* and it doesn't have an impact on the solution. It is automatically ignored in the next steps of our approach because it has a weight of zero (0).

5. Rules Generation

Rules are generated to help automate the process of absolute validation in the Revise module of the CBR. In previous works (Bouabana-Tebibel et al., 2016a; Rubin and Bouabana-Tebibel, 2016a; Bouabana-Tebibel et al., 2016b; Bouabana-Tebibel et al., 2017; Bentaiba-Lagrid et al., 2018), the absolute validation was fully based on the review of the expert. He had to verify each case and give an opinion about its validity. Knowing that the number of cases grows fast, and that it's hard for the expert to catch them all in a relatively short time, we are proposing a new process for rules generation using the initial dataset.

The algorithm for rules generation works as follows: starting with a case base that contains only *fully valid cases* and after calculating the features' weights using the algorithm given in Section 4:

- Reorder the n' relevant features and re-label them so that t_1' will be the most heavily-weighted feature, t_2' is the next most heavily-weighted feature, … and t_n', the least heavily-weighted relevant feature. The *non-relevant features* are ignored.
- To prepare the case base for the process of generating rules using trees, we divide the case base into x parts where x is the number of possible classes of solutions. Each part contains cases with the same class of solution. For each section, we order the cases by the values of the features, starting by t_1', then t_2', … until t_n' .
- Create a decision tree in each part based on the cases in that part, where the first level of the tree refers to the existing values of t_1', the second level refers to the existing values of t_2', …, and the leaves refers to the existing values of t_n' . The values of quantitative, continuous features are grouped into intervals.
- By comparing all of the generated trees with each other, rules are created. Thus, when a branch exists in a tree and doesn't exist in any of the other trees, it is reduced to the minimum possibility of features – starting with deleting the leaves. It is then transformed into a rule where the class of the solution of the part is written on the right-side of the rule. Rules are of the form:

*"**IF** $t_1' = v_{1,r}$ and $t_2' = v_{2,r}$ and … **THEN** S = class"*

These steps are summarized in Algorithm 2:

Algorithm 2: rules generation

1. Reorder features from the most to the least heavily weighted, ignore the *non-relevant features*
2. Create *x* parts of cases, each part contains cases having the same solution
3. Generate a decision tree in each part, where the leaves represent values of the least-relevant feature
4. Branches and sub-branches (starting from the top of the tree- that exists in part and doesn't exist in any of the other parts is transformed into a rule having the form:

*"**IF** $t'_1 = v_{1,r}$ and $t'_2 = v_{2,r}$ and ... **THEN** S = class"*

Example 5.1. from Example 4.1., the features are ordered as follows: t'_1: BI-RADS assessment, t'_2: mass margin, t'_3: mass shape and

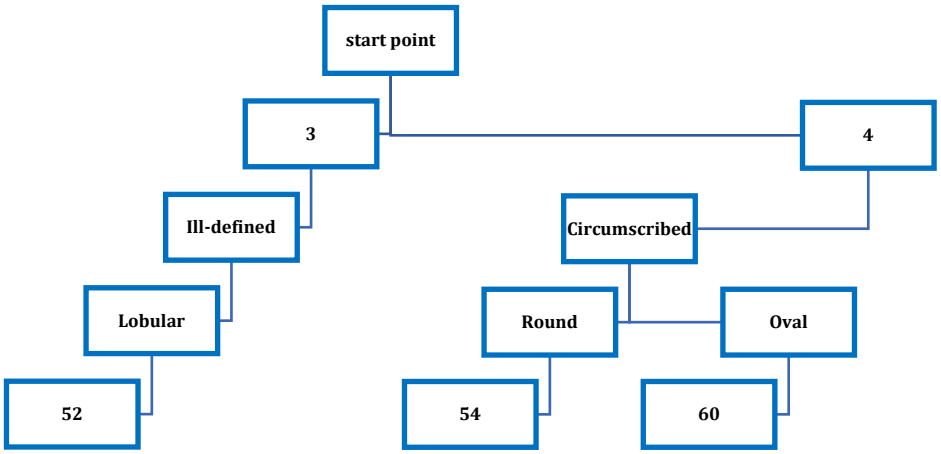


Figure 3: an example of a decision tree created from a sample of cases, which all have benign solutions

t'_4: age respectively. Mass density is ignored.

By using the dataset defined in Example 3.2.1., we will have two parts, a part that contains cases having benign solutions, and another part that contains cases having malignant solutions. In each part, cases are ordered by t'_1, t'_2, t'_3 and t'_4 respectively.

Let's say, we have these cases in the first part:

$C'_1: \{t'_1: 3, t'_2: ill-defined, t'_3: lobular, t'_4: 52\} \Rightarrow benign$

$C'_2: \{t'_1: 4, t'_2: circumscribed, t'_3: round, t'_4: 54\} \Rightarrow benign$

$C'_3: \{t'_1: 4, t'_2: circumscribed, t'_3: oval, t'_4: 60\} \Rightarrow benign$

$C'_4: \{t'_1: 4, t'_2: circumscribed, t'_3: lobular, t'_4: 36\} \Rightarrow benign$

The corresponding decision tree of this part is in Figure 3.

In the decision tree of Figure 3 we have the branch: "start point → 3 → ill-defined →lobular → 52". If this branch doesn't exist in any other decision trees except this one, it could be transformed into the following rule:

*"**IF** t'_1 = 3 and t'_2 = ill-defined and t'_3 = lobular and t'_4 = 52 **THEN** S = benign"*

But first we need to verify whether the rule is reducible or not. To do so, we delete the leaf of the branch and search for this sub-branch: "start point → 3 → ill-defined →lobular" in the other trees. If not found, we repeat the same process until getting the most reduced sub-branch that doesn't exist in any other tree. The rule will be created from it.

The same process will be done for the branch "start point → 4 → circumscribed → round → 54". If this branch can't be reduced to "start point → circumscribed" then all the process will be repeated for the branch "start point → 4 → circumscribed → oval → 60".

The generated rules are updated periodically, or after acquiring a considerable number of *fully valid cases*. The update is done by re-launching the process of rules generation.

6. Case-Base Maintenance

In this section, we present how the case base is segmented and how to do the process of case-base amplification using a new randomization technique.

6.1. Case-Base Segmentation

The case base is prepared to be manipulated by the CBR modules and by the knowledge amplification by randomization. Thus, it is segmented in a matrix form (Figure 4), in order to make the search for cases and their solutions faster. The case base is divided into sectors that are represented by a class of solution. For each class S, we have one sector that contains many segments. Each segment is represented by the class of solution *S* which is the same as the sector's, and by a delegate *D* that gives an overview of the problem part of the cases in the segment. The segment in itself is divided into many levels. These levels depend on the similarity value between the cases and the delegate. The more the problem part of the case is similar to the delegate, the higher is its level, and the higher will be the similarity between the problem parts of the cases at the same level of the segment. This means that the cases in the bottom level of the segment are the least similar to the segment's delegate (Figure 4). All the cases in a segment have the same solution and it is the sectors class. A case can be inserted multiple times in a sector, but only once in a segment of that sector and its frequency is incremented if it already exists anywhere in the case base. The reason for creating a sector is to be able to compare cases having the same solution part, while segments allow us to partition cases depending on their problem parts in an optimized way. To optimize the space, we don't store the full case in the segment, but only its reference.

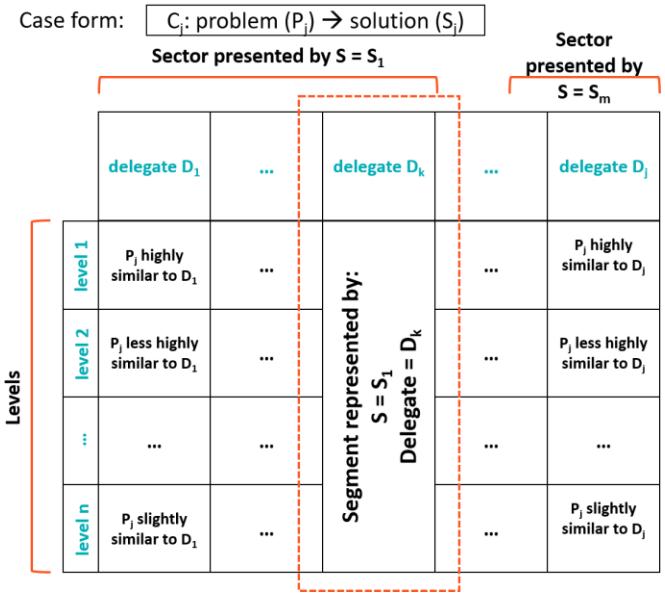


Figure 4: case-base segmentation

The delegate is a summary of the problem parts of the cases in the segment. It is composed of a set of all of the features of the problem part. If the feature t_i is quantitative, its corresponding value in the delegate is the average of all the values of t_i in the cases that are stored in the first level / *m* levels of the segment. On the other hand, If the feature t_k is nominative, for each possible value of t_k symbolized by $v_{n,k}$, we calculate the number of occurrences of each possible value $v_{n,k}$ of t_k using cases stored in the first level / *m* levels of the segment. No need to calculate the occurrences of null values.

The delegate *D* is dynamic, and it is updated each time a new case is stored in the first level / *m* levels of a segment. This will allow him to better describe the cases in the segment. When no segment stores the case in its first level / *m* levels, a new segment will be created thereupon in the appropriate sector, and the case will be saved in its first level.

The levels define the similarity of the case with the delegate of the segment. The possible values of similarities are bounded from 0 to 1. The [0, 1] interval is divided equally between the levels where level 1 refers to the highest similarity and the bottom level refers to the lowest one. We recommend having a number of levels as many as the number of features in the cases or its multiple.

Suppose that we have the case C_j referenced by (1) in Section 3.2 and suppose a segment, which belongs to the sector that is represented by S_j. This segment is represented by a delegate D_f and by the solution S_j. *simDelegate*(C_j, D_f) represents the similarity between the case and the delegate of a segment. The algorithm in Figure 5 represents how *simDelegate* is calculated where *n* is the number of features. We search for the interval that fits *SimDelegate*. The case will be inserted in the corresponding level. We systematically calculate *simDelegate* for C_j and each segment in the sector represented by $(S = S_j)$ to define in which level the case will be stored in each segment.

Each case in the case base must be coherent. Otherwise, it won't be stored in the case base. A case has many metrics to evaluate its validity, naming the frequency metric: which gives the number of times the case was generated by randomization and/or captured from the real world. Other metrics are its stochastic validity and its absolute validity. These metrics are explained in detail in Section 8.

Example 6.1.1. by using the same dataset as in Example 3.2.1. D_1 is a possible delegate for a segment containing 7 cases in its first level.

$$D_1: \langle \begin{array}{c} BI-RADS: \{5: 7 \text{ occurences}, \\ age: 58.85714, \\ mass\ shape: \begin{cases} \text{oval: 1 occurence}, \\ \text{lobular: 2 occurences} \\ \text{irregular: 4 occurences} \end{cases} \\ mass\ margin: \begin{cases} \text{circumscribed: 2 occurences} \\ \text{micro} - \text{lobulated: 1 occurence} \\ \text{obscured: 3 occurences} \\ \text{ill} - \text{defined: 1 occurence} \end{cases} \\ mass\ density: \begin{cases} low: 6\ occurences, \\ fat-containing: 1\ occurence \end{cases} \end{array} \rangle$$

- Mass shape is a qualitative feature: D_1.mass_shape.lobular.occurence = 2
- Number of possible non-null values of mass shape is: 4. The values are: round, oval, lobular, irregular.
- There is no case in that segment where mass shape = round
- Age is a quantitative feature: D_1.age = 58.85714
- Null values don't appear in the delegate and they are not counted

Example 6.1.2. in our dataset, we have five features. It means that we will have five or a multiple of 5 levels. If its 5, the corresponding similarity intervals are defined in Table 3.

Table 3: Similarity levels and their corresponding intervals ranked from 1 to 5

Level 1	Level 2	Level 3	Level 4	Level 5
]0.8, 1]	]0.6, 0.8]	]0.4, 0.6]	]0.2, 0.4]	]0, 0.2]

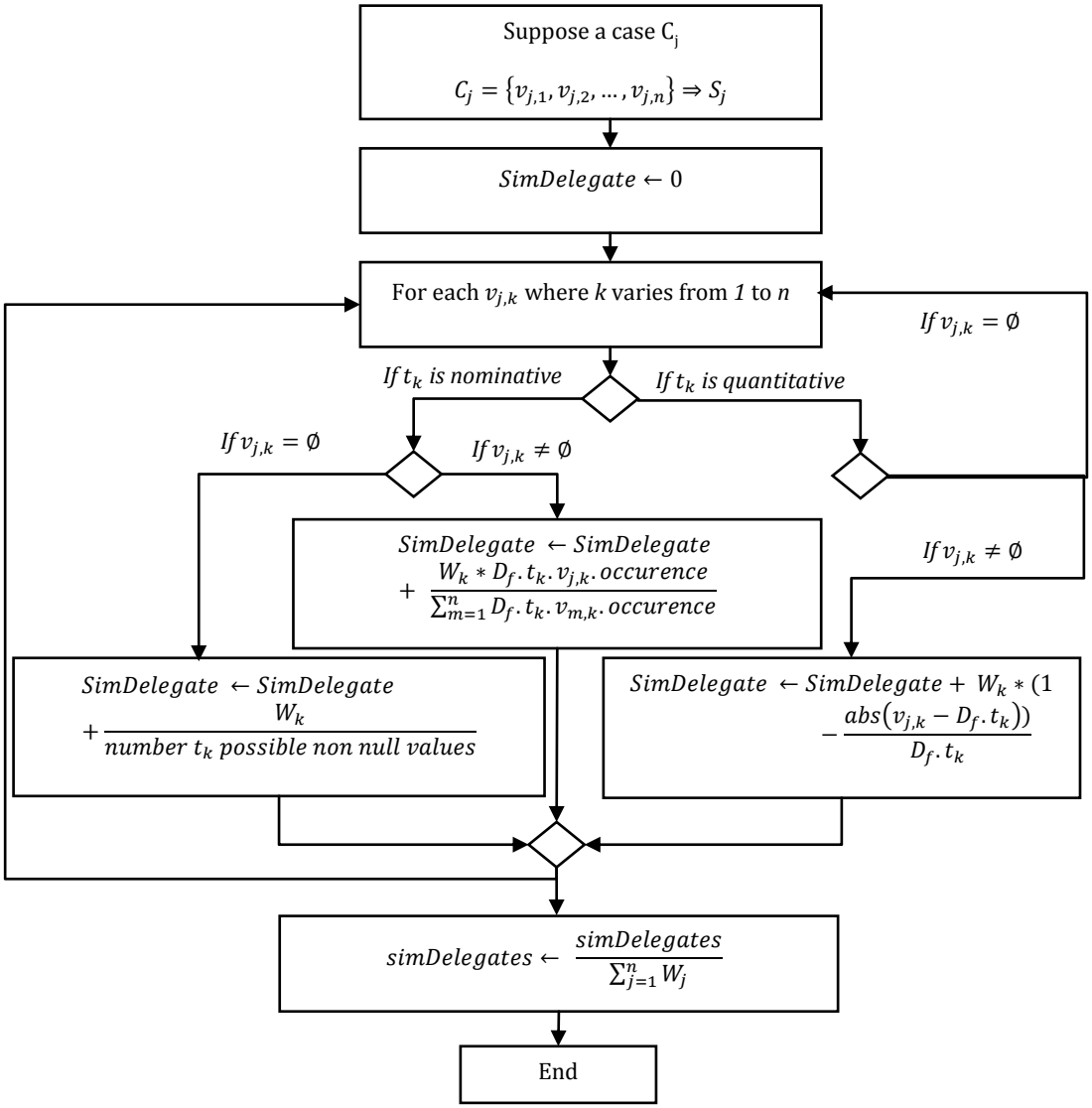


Figure 5: Algorithm of calculation of the similarity between a delegate D_f and a case C_j

6.2. Knowledge Amplification Using Randomization

Each pair of cases belonging to the same level in the same segment, have the same solution. The randomization is fulfilled by interchanging the values of the heavily-weighted features. Then, two new cases are generated. To ensure higher validity, the number of features to substitute will depend on the level where they are stored in the segment. Hence, cases in the highest level have a very similar problem part. Conversely, cases in the lower levels are less similar. At the highest level, we can substitute more features and the new cases will be highly similar to the initial candidate cases. The opposite is true for the lowest level, the substitution of a minimum of features can generate cases totally different from the initial candidate cases. The number of features to substitute *numbF* is determined by the following formula:

(4) $numbF = number\ of\ levels - rank\ of\ the\ level$

The bottom level, which has a rank equal to the number of features is not used for randomization because in this situation, *numbF* is equal to zero, so no feature interchanging is possible in this situation. Thus, it is unnecessary to keep the bottom level stored in the case base.

Let's have C_a and C_b, two cases of the same level of a segment where:

$$C_a: \{t_1: v_{a,1}, t_2: v_{a,2}, \dots, t_j: v_{a,j}, t_{j+1}: v_{a,j+1}, \dots, t_n: v_{a,n}\} \Rightarrow S$$

and

$$C_b: \{t_1: v_{b,1}, t_2: v_{b,2}, \dots, t_j: v_{b,j}, t_{j+1}: v_{b,j+1}, \dots, t_n: v_{b,n}\} \Rightarrow S$$

Suppose that $t_1, t_2, \dots, t_j$ are the *numbF* most weighted features. Then j is equal to *numbF.* By interchanging the values of $\{t_1, t_2, \dots, t_j\}$, two new cases C_c and C_d will be generated:

$$C_c: \{t_1: v_{a,1}, t_2: v_{a,2}, \dots, t_j: v_{a,j}, t_{j+1}: v_{b,j+1}, \dots, t_n: v_{b,n}\} \Rightarrow S$$

and

$$C_d: \{t_1: v_{b,1}, t_2: v_{b,2}, \dots, t_j: v_{b,j}, t_{j+1}: v_{a,j+1}, \dots, t_n: v_{a,n}\} \Rightarrow S$$

If the newly-generated cases are coherent (see Section 8.1), they will be stored in the case base in the appropriate levels of the segments. If the cases already exist in the case base, no need to reintegrate them in the segments, we only increment the cases *frequency*. The *frequency* is initialized to one and represents the number of occurrences of the case. The new generated cases will be used for another iteration of the knowledge amplification by randomization process. The stopping criterion occurs when no new cases (or few new cases) are generated, or after finishing a definite number of iterations. The user can also fix the number of randomization rounds.

Example 6.2.1. from Example 6.1.2., no need to keep the level 5 in the case base, because *numbF* = 0. Thus, when *simDelegate*(C_j, D_f) ≤ *0.2*, C_j won't be stored in that segment.

Example 6.2.2. let's have C_1 and C_2 from the dataset mentioned in Example 3.2.1. where:

$$C_1: \{t_1: 0, t_2: 45, t_3: oval, t_4: \text{ill} - \text{defined}, t_5: \text{low}\} \Rightarrow benign$$

and

$$C_2: \{t_1: 0, t_2: 58, t_3: irregular, t_4: \text{speculated}, t_5: \text{low}\} \Rightarrow benign$$

Suppose that they belong to the same level of a segment ranked by 3, then:

$$numbAttr = 5 - 3 = 2$$

According to Example 4.1., the two most weighted features are BI-RADS and mass margin. The two generated cases are:

$$C_3: \{t_1: 0, t_2: 45, t_3: oval, t_4: \text{speculated}, t_5: \text{low}\} \Rightarrow benign$$

and

$$C_4: \{t_1: 0, t_2: 58, t_3: irregular, t_4: \text{ill} - \text{defined}, t_5: \text{low}\} \Rightarrow benign$$

7. Retrieve and Reuse

We define new functions to calculate the similarity between two cases. These functions are used to retrieve the most useful cases to resolve the current user's problem.

First, we have two types of features in the case: nominative ones and quantitative ones. To compare between two values of a quantitative feature t_j, we propose the following distance function:

$$(5)\ simQuan_j(v_{a,j}, v_{b,j}) = \begin{cases} 1, if(v_{a,j} = v_{b,j}) \\ 0, if\ (v_{a,j} = \emptyset)\ or\ (v_{b,j} = \emptyset) \\ 1 - \frac{abs(v_{a,j} - v_{b,j})}{max_{i=1}(v_{i,j}) - min_{i=1}(v_{i,j})}, if(v_{a,j} \neq v_{b,j}) \end{cases}$$

and to compare between the values of a nominative feature t_j, the expert fills a table of distances between all the possible values of t_j.

Table 4: similarity between possible values of a nominative feature t_j

t_j values	$v_{a,j}$	...	$v_{m,j}$	...	$v_{p,j}$
$v_{a,j}$	$x_{v_{a,j},v_{a,j}} = 1$	...	$x_{v_{a,j},v_{m,j}}$	...	$x_{v_{a,j},v_{p,j}}$
...	...	...	...	...	...
$v_{m,j}$	$x_{v_{m,j},v_{a,j}} = x_{v_{a,j},v_{m,j}}$	...	$x_{v_{m,j},v_{m,j}} = 1$	...	$x_{v_{m,j},v_{p,j}}$
...	...	...	...	...	...
$v_{p,j}$	$x_{v_{a,j},v_{a,j}}$	...	$x_{v_{p,j},v_{m,j}} = x_{v_{m,j},v_{p,j}}$	...	$x_{v_{p,j},v_{p,j}} = 1$

After filling Table 4 by the expert, the similarity between two values of a nominative feature t_j is calculated as follows:

$$(6)\ simNom_j(v_{a,j}, v_{b,j}) = \begin{cases} 1, if(v_{a,j} = v_{b,j}) \\ \frac{1}{number\ of\ possible\ values}, if\ (v_{a,j} = \emptyset)\ or\ (v_{b,j} = \emptyset) \\ x_{v_{a,j},v_{b,j}}, if(v_{a,j} \neq v_{b,j}) \end{cases}$$

Next, the comparison between the values of a feature t_j, no matter its type is summarized in the following function:

$$(7)\ simFeature_j(v_{a,j}, v_{b,j}) = \begin{cases} simNom(v_{a,j}, v_{b,j}), if\ t_j\ is\ nominative \\ simQuan(v_{a,j}, v_{b,j}), if\ t_j\ is\ quantitative \end{cases}$$

Then, to calculate the similarity between two cases, we proposed the following function:

$$(8)\ SimCases(C_a, C_b) = \sum_{j=1}^{n} \frac{W_j * simFeature_j(v_{a,j},\ v_{b,j})}{\sum_{j=1}^{n} W_j}$$

where $v_{a,j}$ and $v_{b,j}$ are values of a feature t_j, and n is the number of features and W_j is its weight.

Suppose that we have a user's problem *Pr*. The Retrieve and Reuse in our CBR are based on three solutions:

i. Retrieve the cases *C* that have *simCases(Pr, C) >= similarity threshold.* For the Reuse we copy the solution of the extracted cases to the user's problem. This will give us new cases having the same problem *Pr* and different solutions.

ii. Fetch for the segments that have *simDelegate(Pr, D) >= similarity threshold*. Where *D* is the delegate of a segment *S*. For the Reuse we copy the classes of solution in the segments' representatives (composed of a delegate and a class of solution). This will give us new cases having the same problem *Pr* and different solutions.

iii. Test the problem *Pr* with different classes of solution *S* using rules. If the new case *Pr* → *S* break one of the rules in the rule-base, the solution *S* is rejected. Otherwise, the new case is kept and revised in the Revise module of the CBR. This step is generally added when no similar cases/delegates to the user's problem are found.

Adding solution iii) after i) or ii) highly augments the ratio of the answered problems.

Note that we are dealing with supervised classification problem that can be mono-class or multi-class problem. Multi-class problem means that we can have many correct solutions that will be provided to one problem.

8. Revise and Retain

We have two sources of knowledge: The first one is the experiences captured from the real word as cases. The system entirely believes in their validity. We call them: *fully-valid cases*. The second one is the knowledge generated using randomization by applying transformations to the previously stored knowledge in the case base. The latter is not necessarily valid. Randomization is a process that can generate a huge amount of knowledge with a high allowance of circling data with unknown validity. It is hard to ensure that all of the cases generated by randomization are valid. In real-world tasks, the error metric cannot be null, but it is bounded (Rubin and Bouabana-Tebibel, 2016a). The Revise module has two functions, the first one is to validate the generated knowledge by randomization and the second one is to revise the solutions given by the Reuse module before providing it to the

user. We introduce a three-layer approach for verification and validation composed of: (1) coherence verification, (2) stochastic validation and (3) absolute validation.

8.1. Coherence Verification

Coherence verification depends on the application domain. Suppose a dataset that contains *M fully valid* cases captured from the real world. Suppose that these cases have *n* features which are $t_1, t_2, \ldots, t_n$ and *x* classes of solutions denoted by $S_1, S_2, \ldots, S_x$. To verify the coherence of the cases generated from this dataset either by randomization or by the Reuse module, we construct the related stochastic context-free grammar. It is a quintuple $G = < N, T, R, D, P >$ where:

- *N* is the set of non-terminal symbols: $N = \{D, T_1, T_2, \ldots, T_n, S\}$
- *T* is the set of terminal symbols: $T = \{\emptyset, all\ possible\ values\ of\ features\}$
- *R* is the set of production rules:

$$R = \begin{cases} D \rightarrow \text{"}\{\text{"}\ T_1 T_2 \ldots T_n\ \text{"}\}\Rightarrow\text{"}\ S \\ T_1 \rightarrow \emptyset / other\ possible\ values\ of\ the\ feature\ t_1\ , \\ T_2 \rightarrow \emptyset / other\ possible\ values\ of\ the\ feature\ t_2\ , \\ \ldots \\ T_n \rightarrow \emptyset / other\ possible\ values\ of\ the\ feature\ t_n, \\ S \rightarrow\ S_1 / S_2 / \ldots / S_x \end{cases}$$

- *D* is the start symbol
- *P* is the set of probabilities on production rules.

$$P = \begin{cases} p(D \rightarrow \text{"}\{\text{"}\ T_1 T_2 \ldots T_n\ \text{"}\}\Rightarrow\text{"}\ S) = 1, \\ p(T_i \rightarrow\ \emptyset) = \dfrac{number\ of\ cases\ having\ (t_i = \emptyset)}{M}, \\ p(T_i \rightarrow\ v_{m,i}) = \dfrac{number\ of\ cases\ having\ (t_i = v_{m,i})}{M}, \\ p(S \rightarrow S_j) = \dfrac{number\ of\ cases\ having\ (S = S_j)}{M}, \\ where\ i\ varies\ from\ 1\ to\ n\ and \\ and\ j\ varies\ from\ 1\ to\ x\ and \\ \{v_{m,i}, m\ varies\ from\ 1\ to\ z_i\}\ are\ all\ the\ possible\ values\ of\ t_i \end{cases}$$

By using the stochastic grammar to verify the coherence of the case we ensure:

- ***the coherence of the problem part***: we ensure that the values of the features belong to the previously defined intervals if it consists of a quantitative feature or set of values if it consists of a nominative feature and are not outside of it. The case found to be containing some corrupted or unexpected values are systematically deleted,
- ***the coherence of the solution part***: as we mentioned above, the possible solutions are defined when the domain of application is defined. Thus, it is not possible to have a newly generated class of solution. The solutions are known and limited to a set of classes,
- ***the coherence of the case***: the solution part will be coherent with the problem part.

All of the incoherent cases are rejected and all the stored cases in the case base by the Retain module, must be coherent. In the Retain module, either the case is stored in an appropriate segment, if it was not stored previously, or its frequency is incremented.

Example 8.1.1. using Example 3.2.1. The related stochastic grammar is the quintuple $G = < N, T, R, D, P >$ where:

- *N* is the set of non-terminal symbols: $N = \{D, T_1, T_2, T_3, T_4, T_5, S\}$
- *T* is the set of terminal symbols.
- *R* is the set of production rules:

$$R = \begin{cases} D \rightarrow \text{"}\{\text{"}\ T_1 T_2 T_3 T_4\ T_5\text{"}\} \Rightarrow\ \text{"}\ S, \\ T_1 \rightarrow\ 0/1/2/3/4/5/6/\emptyset\ , \\ T_2 \rightarrow\ patient's\ age/\emptyset\ , \\ T_3 \rightarrow\ round/oval/lobular/irregular/\varepsilon\ , \\ T_4 \rightarrow\ circumscribed/micro-lobulated/obscured\ , \\ T_4 \rightarrow\ ill-defined/speculated/\varepsilon\ , \\ T_5 \rightarrow\ high/iso/low/fat-containing/\varepsilon\ , \\ S \rightarrow\ benign/malignant \end{cases}$$

- *D* is the start symbol
- *P* is the set of probabilities on production rules.

$$P = \Big\{ p(D \rightarrow \text{"\{" } T_1T_2T_3T_4\ T_5\text{"\}} \Rightarrow \text{" } S) = 1, \quad p(T_1 \rightarrow 0) = \frac{5}{961}, \quad p(T_1 \rightarrow 1) = 0, \quad p(T_1 \rightarrow 2) = \frac{14}{961}, \quad \dots,$$
$$P(S \rightarrow benign) = \frac{516}{961}, \quad P(S \rightarrow malignant) = \frac{445}{961} \Big\}$$

8.2. Stochastic Validation

Stochastic validation consists of calculating a probability of case validity. In the present work, the probability depends on three parameters. The first parameter is the frequency ratio of generating the case. A high-frequency ratio means a high-stochastic validity. Notice that the randomization process starts with a case base that contains only valid cases. These cases are in general captured from the real word. The case that was generated many times with different paths (different segments and/or different candidate cases) is more likely to be valid. The second parameter is related to the distribution of features' values in the initial case base. Thus, we are using the stochastic grammar in section 8.1. to calculate it. The third parameter is how much the generated case is similar to one of the *fully-valid* cases in the case base. If the newly generated case is highly similar to a valid case, then it has a high-validity probability. The stochastic validity is dynamic and may be updated when a new case is inserted in the case base.

The first parameter of the stochastic validity is *frequencyRatio*, which is calculated by dividing the *frequency* of the case by the sum of the *frequency* of all cases that have the exact same problem part regardless their solution:

$$(9)\quad frequencyRatio(C_a: P_a \Rightarrow S_a) = \frac{frequency(C_a: P_a \Rightarrow S_a)}{\sum_i frequency(C_i: P_a \Rightarrow S_i)}$$

The *frequency* of the case is the number of occurrences of the case. In other words, it's the number of times the case was generated.

Experimental results have shown that in more than 66% of the generated cases, *frequencyRatio* is equal or higher than 90%, due to how the randomization is performed. To reduce that probability, we added a new parameter called *randomnessRatio,* which is calculated based on the stochastic grammar given in Section 8.1. using the initial case base:

$$(10)\quad randomnessRatio(C_a: P_a \Rightarrow S_a) = \frac{p(D \overset{G}{\rightarrow} (C_a: P_a \Rightarrow S_a))}{\sum_i p(D \overset{G}{\rightarrow} (C_i: P_a \Rightarrow S_i))}$$

$D \overset{G}{\rightarrow} (C_a: P_a \Rightarrow S_a)$ is the series of productions of the grammar G starting by the starting symbol D until getting the case $C_a: P_a \Rightarrow S_a$, using the leftmost derivation. $p(D \overset{G}{\rightarrow} (C_a: P_a \Rightarrow S_a))$ is its probability and it is calculated as follows:

$$(11)\quad p\left(D \overset{G}{\rightarrow} \text{"}C_a\text{"}\right) = \prod_j p(x_j)$$

where x_i is an elementary production from the set of productions P that leads to the case C_a.

The third parameter for calculating the stochastic validity is the *significance*. It is based on *simCases* values (see Section 7.). For a newly generated case C_a we search in the case base for the most similar case $C_v: P_v \Rightarrow S_v$, from the *fully valid* ones. *significance* will be the value of $SimCases(C_a, C_v)$ or its complement, depending on the solution part of the two cases. *significance* is defined as follows:

$$(12)\quad significance(C_a: P_a \Rightarrow S_a) = \begin{cases} simCases(C_a, C_v), if\ S_a = S_v \\ 1 - simCases(C_a, C_v), if\ S_a \neq S_v \end{cases}$$

Then, the stochastic validity is the average of $frequencyRatio$, $randomnessRatio$ and $significance$. It is calculated as follows:

$$(13)\quad stochasticValidity = (frequencyRatio + randomnessRatio + significance)/3$$

8.3. Absolute Validation

It is recommended to move fast toward absolute validation in the early iterations of the randomization process, to prevent any propagation of errors. After verifying the cases coherence and their stochastic validity, absolute validity is the last step that gives a final decision about its validity. Absolute validation is based on two tools: the first tool has priority over the second and consist of verifying per rules that were previously generated and stored in a rule-base (section 5.). The second tool is verification per expert. The aim is to not overload the expert with repetitive verification tasks. Its presence is not necessary but is preferred.

The verification per rules is fulfilled as follow: cases that break the rules, are set as invalid. Cases that have at least one rule that goes with it, are set valid. There are some cases that can't be verified per rules. Their absolute validity is unknown. In this situation, the expert can give its opinion about it or we only rely on its stochastic validity.

9. Experimental Evaluation

According to (Siegel et al., 2019), breast cancer is the most common cancer in the US. 15.34 percent of persons who had cancer, have had breast cancer (Siegel et al., 2019). The probability of developing breast cancer is 12.5 percent (Siegel et al., 2019).

Mammography is regarded as the most effective tool for breast cancer detection and diagnosis available today (Elter and Horsch, 2009). However, mammogram interpretation is repetitive and thus an error-prone task. This leads to 10 – 30 percent of all cancers being missed by radiologists (Elter and Horsch, 2009). Thus, we need a way to automatically detect suspicious lesions in mammograms, which otherwise might have been missed by radiologists, and serves as a reminder by pointing out their location (Elter and Horsch, 2009). To better highlight our approach, we are working within the mammogram interpretation domain, using the dataset defined in Example 3.2.1. to classify the severity of mammographic-revealed masses. We have performed experiments on four different configurations of the CBR that are defined in Table 5.

Table 5: different configurations of our CBR

Configurations	Case base	Retrieve	Reuse	Revise	Retain
I_NS_CB	Flat case base	Use *simCases* function to retrieve the most similar cases	1. Copy the solution of the similar cases 2. Use rules	Three layers validation	Store the new case if it's coherent in the flat case base
I_S_CB	Segmented case base	Use *simDelegate* algorithm to retrieve the most similar delegates	1. Copy the solution from segments' representatives 2. Use rules	Three layers validation	Store the new case if it's coherent in the segmented case base
R_S_CB	Segmented case base Apply knowledge amplification on it	Use *simDelegate* algorithm to retrieve the most similar delegates	1. Copy the solution from segments' representatives 2. Use rules	Three layers validation	Store the new case if it's coherent in the segmented case base
R_NS_CB	1. One flat case base and 2. One segmented case base Apply knowledge amplification on the segmented case base	Use *simCases* function to retrieve the most similar cases	1. Copy the solution of the similar cases 2. Use rules	Three layers validation	Store the new case if it's coherent in the flat and in the segmented case base

For the coming sections, five metrics will be used to evaluate the CBR performance: resolution capacity, which is a percentage expressed in formula (14), accuracy is also a percentage expressed in formula (15) resolution time in seconds, case base amplification ratio expressed in formulas (16) and (17) and the exactitude of the validation process expressed in Sections 8.1, 8.2 and 8.3.

$$(14) \quad resultion\ capacity = \frac{number\ of\ answered\ problems}{number\ of\ problems}$$

$$(15) \quad accuracy = \frac{number\ of\ answered\ problems\ correctly}{number\ of\ problems}$$

$$(16) \quad percentage\ of\ generated\ cases = \frac{number\ of\ generated\ cases}{number\ of\ initial\ cases} * 100$$

$$(17) \quad percentage\ of\ invalid\ cases = \frac{number\ of\ invalid\ cases\ from\ the\ generated\ ones}{number\ of\ generated\ cases} * 100$$

Then, we will compare our work with the related work in term of resolution capacity and accuracy, naming supervised ML tools and researches on CBR and other intelligent systems that are using the same dataset.

9.1. Resolution Capacity, Accuracy and Resolution Time Evaluation

We applied cross-validation tests on the four different configurations of the CBR shown in Table 5, to get their resolution capacity (formula 14) and their accuracy (formula 15) using different *similarity thresholds*. The results (Figure 6) show that when randomization is performed, the system is able to answer to all of the user's problems (100 percent of the problems are

answered). Even the accuracy is improved with randomization (R_S_CB and R_NS_CB) compared to the CBRs that aren't using randomization (I_NS_CB and I_S_CB).

Segmentation reduces the resolution capacity and the accuracy of the system. The aim of adding a segmentation process is to reduce resolution time, which is shown in the next experiment (Figure 7). Despite this, a randomized segmented case base (R_S_CB) still showing better results than the initial non-segmented case base (I_NS_CB) in term of resolution capacity and accuracy.

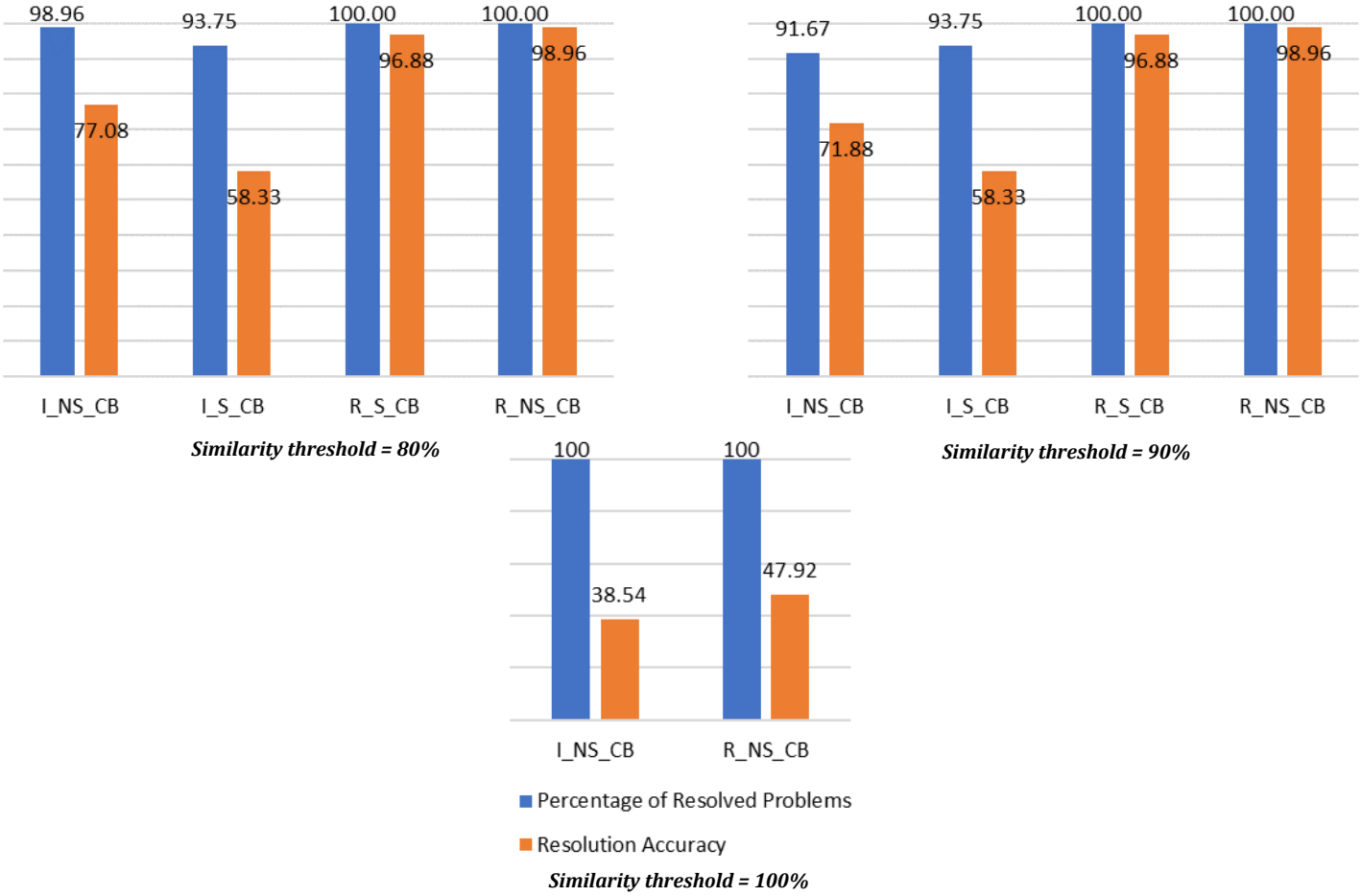


Figure 6: capacity of resolution and accuracy

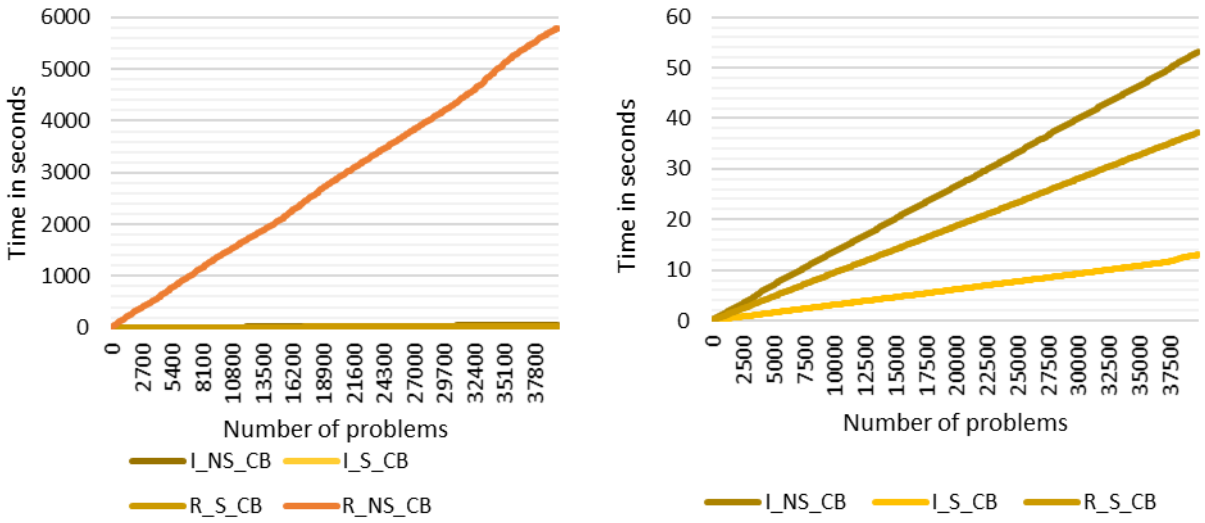
When the *similarity threshold* equals 100 percent, the accuracy of I_NS_CB and I_S_CB is less than 50 percent, which is not satisfying. This is due to the application domain, where some problems can have different solutions. It means that some of the training cases have different solutions than the test cases, having the same problem.

Figure 7 shows resolution time for the different configurations of the CBR. When the *similarity threshold* is less than 100 percent, the resolution time of 40,000 problems in the randomized non-segmented case base (R_NS_CB) is approximately 100 times higher than the resolution time of the other configurations (I_NS_CB, I_S_CB and R_S_CB). This is due to the massive number of cases in the case base. Also, we see that the segmentation highly reduces resolution time comparing to other non-segmented configurations. We get the best resolution time when the required similarity is 100 percent, because here we look for the case that has the exact same problem part rather than extract all of the cases/delegates and calculate their similarities with the user's problem.

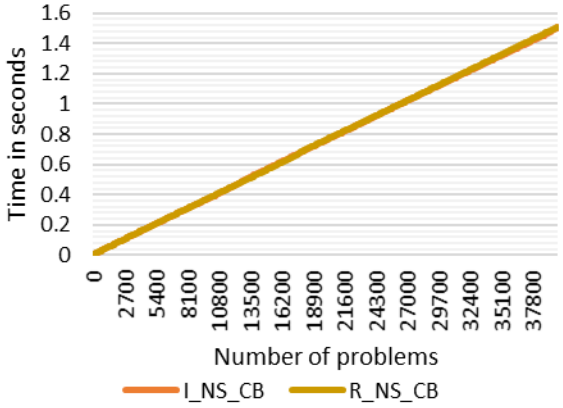
9.2. Case-Base Amplification Analysis

Figure 8 shows the percentage of case base augmentation, using our approach of randomization. In each experiment, we amplify the case base that initially contains 50 random cases. The generated cases are validated using our three-layers validation. The X-axis presents the experiment number, the Y-axis shows the percentage of generated cases. The percentage of generated cases is calculated as follows.

The results show that the percentage of generated cases is random and varies from 138 to 390 percent. Using these cases, the number of invalid ones is negligible, which is a good sign.



Similarity threshold < 100%



Similarity threshold = 100%

Figure 7: resolution time (in seconds)

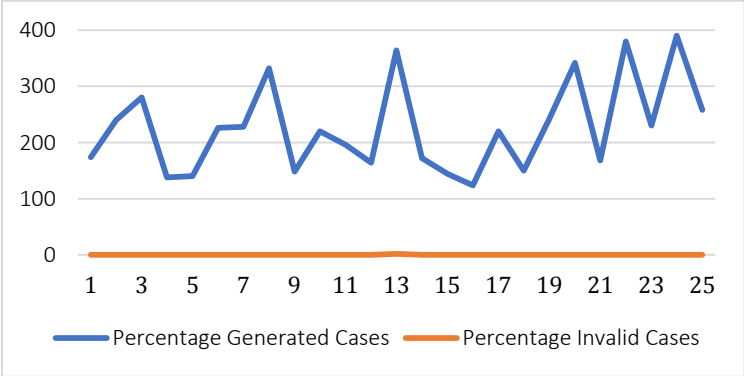


Figure 8: percentage of cases generated by randomization starting with 50 random initial cases; the experiment is repeated 25 times

9.3. Validation Process Evaluation

Using cross-validation, the dataset is divided into a set of training cases and a set of test cases. Both training cases and test cases are *fully valid* because they are captured from real patients. Training cases helps generate rules using the algorithm described in section 5. The aim of this experiment is to show how our three-layers validation verify and validate the cases. The results are presented in Table 8.

- All cases are found to be coherent (using coherence verification) which is true.
- 82.95 percent of training cases have a stochastic validity higher than 0.7 (70 percent). The rest of 17.05 percent is found stochastically invalid. While, in reality all the cases are valid. The reason of having a percentage less than 100 of the

stochastically valid cases is that some of the cases have the same problem and different solutions due to the application domain that is multi-class.

- Systematically, 58.33 percent of the test cases are found stochastically invalid. This is due to the characteristic of the domain.
- None of the training cases were found to be invalid using verification rules. The reason is that rules are generated from the training cases.
- Compared to the test cases: 7.29 percent of cases are found to be invalid using the absolute validation module. Where in reality they are all valid.

Table 6: results of tests on validation process

	Values	Training Cases (%)	Test Cases (%)
Coherence	Cases found as coherent	100	100
	Cases found as uncoherent	0	0
Stochastic Validity	Cases found with stochastic validity < 0.7	17.05	58.33
	Cases found with stochastic validity between 0.7 and 1	12.62	41.67
	Cases found with stochastic validity = 1	70.33	1.04
Absolute Validity (rules)	Number Cases Set as Valid	68.69	38.54
	Number Cases set as Validity Unknown	31.31	54.17
	Number Cases set as Non-Valid	0	7.29

9.4. Comparison Between Our Work and the Related Work

In this set of experiments, we show the resolution accuracy of many supervised ML tools that are dedicated to the classification problems. They are tested using sklearn API (Buitinck et al., 2013), the accuracies are calculated using the same training set, and the same test set of cases for each ML tool. The results in Table 6, show that R_S_CB and R_NS_CB have the highest accuracies. Both of these configurations are using a randomized case base and have the highest accuracies where 100 percent of problems are answered.

Table 7: comparison between accuracies of supervised ML tools

Supervised ML Tool	Accuracy Calculated on the Training Set (%)	Accuracy Calculated on the Test Set (%)
Logistic Regression (Hosmer et al., 2013)	77.54	82.29
Support Vector Machine (Byun and Lee, 2002)	79.34	82.29
K Nearest Neighbors (Fukunage and Narendra, 1975)	75.41	82.29
Gaussian Naive Bayes (Jiang and al., 2007)	76.72	82.29
Perceptron (Stephen, 1990)	71.8	76.04
Linear Support Vector Clustering (Ben-Hur et al., 2001)	78.2	82.29
Stochastic Gradient Descent (Bottou, 2012)	74.75	81.25
Decision Tree (Chen, 1995)	92.46	80.21
Random Forest (Verikas et al., 2011)	92.46	85.42
I_NS_CB	98.96	77.08
I_S_CB	93.75	58.33
R_S_CB	**100**	**96.875**
R_NS_CB	**100**	**98.96**

In the same manner as the latter comparison, we compare the resolution accuracy of many related works. These results are taken from known and ranked journals and conference papers. The authors are clearly showing the accuracies in their papers and they are using the same dataset as we are. The results are listed in Table 7 and show that our system, with an initial segmented

case base (I_S_CB), are less accurate among all of the works, but when randomization is added (R_S_CB), the accuracy grew from 58.33 to 96.875 percent. The highest accuracies belong to the both systems that are using randomization (R_S_CB and R_NS_CB).

Table 8: comparison between accuracies of the related work

Related Work	Accuracy (%)
GDM-GA-CBR (Yan et al., 2016)	75.45
GA-CBR (Yan et al., 2016)	78.13
WE-CBR (Rezvan et al., 2013)	79.47
DSL-CBR (Rezvan et al., 2013)	79.79
NN (Yan et al., 2016)	80
WEH-CBR (Rezvan et al., 2013)	80.62
DUL-CBR (Rezvan et al., 2013)	82.29
(Smiti and Elouedi, 2018)	90.98
(Fan et al., 2011)	92.7
I_NS_CB	77.08
I_S_CB	58.33
R_S_CB	**96.875**
R_NS_CB	**98.96**

10. Conclusion and Discussion

We proposed, in this paper, a new approach based on the CBR and randomization to resolve supervised classification problems accurately without deteriorating the resolution time. The proposed approach is tested on mammographic applications to classify patient's mammogram severity. In our approach, we first introduced a new algorithm for features weighting and selection. Then, we proposed a new process to generate rules that are used in the Reuse module as well as in the Revised modules. Then, we introduced a new case-base segmentation technique to speed up the retrieval for relevant information. After that, we proposed a new randomization technique to amplify the case base. The aim is to extract the implicit knowledge from the explicit ones. We also defined new similarity functions to improve the retrieval for similar cases or segments. Finally, we proposed a new three-layer validation process composed of (1) coherence verification sub-module, (2) stochastic validation sub-module and (3) absolute validation sub-module, to validate the newly-generated cases by randomization or to revise cases generated by the Reuse module before providing it to the user and automating the system, as well as to reduce the expert's intervention. The aim of these contributions is to provide a CBR system that is able to accurately resolve supervised classification problems in relatively minimal time.

The proposed approach is tested on the mammographic application using five metrics that are: resolution capacity, accuracy, resolution time, amplification ratio and the exactitude of the validation process. Thus, we developed a prototype of different configurations of CBR, which are: (i) CBR with an initial non-segmented case base, which is equivalent to a native CBR (I_NS_CB). (ii) CBR with initial segmented case base (I_S_CB). (iii) CBR with randomized segmented case base (R_S_CB), (iv) CBR with randomized non-segmented CBR (R_NS_CB). In configuration (i) and (iv) the case base is stored in a flat format. These four configurations are compared to some related works that are using the same dataset, and to some popular ML tools, in term of resolution capacity and accuracy. The obtained experimental results are promising. They show that the CBR is more efficient in problem solving, and that our approach can concurrently process the supervised ML algorithms. The experiments have shown that the accuracies of the configuration (iii) and (iv) are better than that for the related works, while (iv) has the highest accuracy among all. When we consider both accuracy and resolution time, we recommend using configuration (iii) where we lose 0.02 percent of accuracy, but the answer is provided 100 times faster compared to the configuration (iv).

Note that theoretically, it is possible to apply the proposed approach on other domains, rather than mammographic mass, where the type of the problem is the supervised classification one. In this situation, the approach must be retested with real data to be able to choose the best tool to resolve the problem.

11. Acknowledgment

This research did not receive any specific grant from funding agencies in the public, commercial or not-for-profit sectors. The author Miled wants to thank Abdelhalim Lagrid for his help and encouragement to finish this work.

12. References

Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI communications, 7(1), 39-59.

Ahn, H., & Kim, K. J. (2009). Global optimization of case-based reasoning for breast cytology diagnosis. Expert Systems with Applications, 36(1), 724-734.

Bagni, G. T. (2009). Bombelli's Algebra (1572) and a new mathematical object. For the Learning of Mathematics, 29(2), 29-31.

Ben-Hur, A., Horn, D., Siegelmann, H. T., & Vapnik, V. (2001). Support vector clustering. Journal of machine learning research, 2(Dec), 125-137.

Bentaiba-Lagrid, M. B., Bouzar-Benlabiod, L., Rubin, S. H., Bouabana-Tebibel, T., & Hanini, M. R. (2018, July). Knowledge Amplification Using Randomization in Case-Based Reasoning--Case Study: Severity of Mammography Mass. In 2018 IEEE International Conference on Information Reuse and Integration (IRI) (pp. 155-162). IEEE.

Bichindaritz, I., & Montani, S. (2011). Guest Editorial: Advances in case-based reasoning in the health sciences. Artificial Intelligence in Medicine, 51(2), 75-79.

Bottou, L. (2012). Stochastic gradient descent tricks. In Neural networks: Tricks of the trade (pp. 421-436). Springer, Berlin, Heidelberg.

Bouabana-Tebibel, T., Rubin, S. H., Chebba, A., Bediar, S., & Iskounen, S. (2016a, July). Knowledge induction based on randomization in case-based reasoning. In 2016 IEEE 17th International Conference on Information Reuse and Integration (IRI) (pp. 541-548). IEEE.

Bouabana-Tebibel, T., Rubin, S. H., Hoadjli, Y., & Benaziez, I. (2016b). An Approach Transmutation-Based in Case-Based Reasoning. In Quality Software Through Reuse and Integration (pp. 24-41). Springer, Cham.

Bouabana-Tebibel, T., Rubin, S. H., Bentaiba, M. B., Allaoua, A., & Boumhand, A. (2017, August). Knowledge Amplification through Randomization for Scheduling Systems. In 2017 IEEE International Conference on Information Reuse and Integration (IRI) (pp. 589-598). IEEE.

Bouabana-Tebibel, T., Rubin, S. H., Bouzar-Benlabiod, L., Bentaiba-Lagrid, M. B., & Hanini, M. R. (2018, July). Knowledge-Based Randomization for Amplification. In 2018 IEEE International Conference on Information Reuse and Integration (IRI) (pp. 147-154). IEEE.

Bouzar-Benlabiod, L., Rubin, S. H., & Lila, M. (2018, July). Randomization-Based Knowledge Discovery with Application to Weather Prediction. In 2018 IEEE International Conference on Information Reuse and Integration (IRI) (pp. 163-169). IEEE.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., ... & Layton, R. (2013). API design for machine learning software: experiences from the scikit-learn project. arXiv preprint arXiv:1309.0238.

Byun, H., & Lee, S. W. (2002, August). Applications of support vector machines for pattern recognition: A survey. In International Workshop on Support Vector Machines (pp. 213-236). Springer, Berlin, Heidelberg.

Chaitin, G. J. (1975). Randomness and mathematical proof. Scientific American, 232(5), 47-53.

Chebba, A., Bouabana-Tebibel, T., Rubin, S. H., & Habib, K. (2016). Case Indexing by Component, Context, and Encapsulation for Knowledge Reuse. In Theoretical Information Reuse and Integration (pp. 113-134). Springer, Cham.

Chen, H. (1995). Machine learning for information retrieval: neural networks, symbolic learning, and genetic algorithms. Journal of the American society for Information Science, 46(3), 194-216.

De Mantaras, R. L., & Armengol, E. (1998). Machine learning from examples: Inductive and lazy methods. Data & Knowledge Engineering, 25(1-2), 99-123.

Elter, M., & Horsch, A. (2009). CADx of mammographic masses and clustered microcalcifications: a review. Medical physics, 36(6Part1), 2052-2068.

Elter, M., Schulz-Wendtland, R., & Wittenberg, T. (2007). The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process. Medical physics, 34(11), 4164-4172.

Fan, C. Y., Chang, P. C., Lin, J. J., & Hsieh, J. C. (2011). A hybrid model combining case-based reasoning and fuzzy decision tree for medical data classification. Applied Soft Computing, 11(1), 632-644.

Fukunage, K., & Narendra, P. M. (1975). A branch and bound algorithm for computing k-nearest neighbors. IEEE transactions on computers, (7), 750-753.

Hagan, M. T., Demuth, H. B., Beale, M. H., & De Jesús, O. (1996). Neural network design (Vol. 20). Boston: Pws Pub.

Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression (Vol. 398). John Wiley & Sons.

Huang, M. L., Hung, Y. H., Lee, W. M., Li, R. K., & Wang, T. H. (2012). Usage of case-based reasoning, neural network and adaptive neuro-fuzzy inference system classification techniques in breast cancer dataset classification diagnosis. Journal of medical systems, 36(2), 407-414.

Jiang, L., Wang, D., Cai, Z., & Yan, X. (2007, August). Survey of improving naive bayes for classification. In International Conference on Advanced Data Mining and Applications (pp. 134-145). Springer, Berlin, Heidelberg.

Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering, 160, 3-24.

Mazurowski, M. A., Zurada, J. M., & Tourassi, G. D. (2008). Selection of examples in case-based computer-aided decision systems. Physics in Medicine & Biology, 53(21), 6079.

Michalski, R. S., & Myofsciences, P. (1994). INFERENTIAL THEORY OF LEARNING: Developing Foundations. Machine Learning: A Multistrategy Approach, 3.

Pedrycz, W., & Rubin, S. H. (2010). Data compactification and computing with words. Engineering Applications of Artificial Intelligence, 23(3), 346-356.

Quellec, G., Lamard, M., Cazuguel, G., Roux, C., & Cochener, B. (2011). Case retrieval in medical databases by fusing heterogeneous information. IEEE Transactions on Medical Imaging, 30(1), 108-118.

Rezvan, M. T., Hamadani, A. Z., & Shalbafzadeh, A. (2013). Case-based reasoning for classification in the mixed data sets employing the compound distance methods. Engineering Applications of Artificial Intelligence, 26(9), 2001-2009.

Richter, M. M., & Weber, R. O. (2016). Case-based reasoning (p. 27). Springer-Verlag Berlin An.

Rubin, S. H. (1991, October). Learning in the large: case-based software systems design. In Conference Proceedings 1991 IEEE International Conference on Systems, Man, and Cybernetics (pp. 1833-1838). IEEE.

Rubin, S. H. (1999). Computing with words. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 29(4), 518-524.

Rubin, S. H. (2007). On randomization and discovery. Information Sciences, 177(1), 170-191.

Rubin, S. H., & Bouabana-Tebibel, T. (2016a). Naval intelligent authentication and support through randomization and transformative search. In New Approaches in Intelligent Control (pp. 73-108). Springer, Cham.

Rubin, S. H., & Bouabana-Tebibel, T. (2016b). NNCS: Randomization and informed search for novel naval cyber strategies. In Recent Advances in Computational Intelligence in Defense and Security (pp. 193-223). Springer, Cham.

Rubin, S. H., Murthy, S. J., Smith, M. H., & Trajkovic, L. (2004). KASER: knowledge amplification by structured expert randomization. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 34(6), 2317-2329.

S.H. Rubin, inventor; The United States Rubin, S. H. (2016). U.S. Patent No. 9,396,441. Washington, DC: U.S. Patent and Trademark Office.

Sharaf-El-Deen, D. A., Moawad, I. F., & Khalifa, M. E. (2014). A new hybrid case-based reasoning approach for medical diagnosis systems. Journal of medical systems, 38(2), 9.

Siegel, R. L., Miller, K. D., & Jemal, A. (2019). Cancer statistics, 2019. CA: a cancer journal for clinicians, 69(1), 7-34.

Singh, M., & Valtorta, M. (1995). Construction of Bayesian network structures from data: a brief survey and an efficient algorithm. International journal of approximate reasoning, 12(2), 111-131.

Smiti, A., & Elouedi, Z. (2010). Coid: Maintaining case method based on clustering, outliers and internal detection. In Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing 2010 (pp. 39-52). Springer, Berlin, Heidelberg.

Smiti, A., & Elouedi, Z. (2013, April). Using clustering for maintaining case based reasoning systems. In 2013 5th International Conference on Modeling, Simulation and Applied Optimization (ICMSAO) (pp. 1-6). IEEE.

Smiti, A., & Elouedi, Z. (2014, June). Maintaining case based reasoning systems based on soft competence model. In International Conference on Hybrid Artificial Intelligence Systems (pp. 666-677). Springer, Cham.

Smiti, A., & Elouedi, Z. (2018). SCBM: soft case base maintenance method based on competence model. Journal of Computational Science, 25, 221-227.

Stephen, I. (1990). Perceptron-based learning algorithms. IEEE Transactions on neural networks, 50(2), 179.

Verikas, A., Gelzinis, A., & Bacauskiene, M. (2011). Mining data with random forests: A survey and results of new tests. Pattern recognition, 44(2), 330-349.

Yan, A., Song, H., & Wang, P. (2016). Case-based reasoning model with genetic algorithms, group decision-making and template reduction. International Journal on Artificial Intelligence Tools, 25(02), 1550032.

Žliobaitė, I., Bifet, A., Read, J., Pfahringer, B., & Holmes, G. (2015). Evaluation methods and decision theory for classification of streaming data with temporal dependence. Machine Learning, 98(3), 455-482.

***Highlights (for review)**

- case-based reasoning for supervised classification problems
- New Retrieve, Reuse, Revise and Retain algorithms in the case-based reasoning
- New randomization technique to amplify the case-base and case-base segmentation
- New algorithm for features selection and weighting
- Apply the full approach to classify patients' mammographic mass

Table 1: number of pairs of cases

	and with highly similar values of t_i	and with slightly similar values of t_i
Pairs of cases with the same solution	V_i	Y_i
Pairs of cases with different solution	X_i	Z_i

Table 2: features weights before and after normalizing

Feature	Label	W_i'	W_i
BI-RADS assessment	t_1	181059	1.0
Age	t_2	31019	0.1754
Mass shape	t_3	88313	0.4903
Mass margin	t_4	91239	0.5063
Mass density	t_5	-887	0 / ignored

Table 3: Similarity levels and their corresponding intervals

Level 1	Level 2	Level 3	Level 4	Level 5
]0.8, 1]	]0.6, 0.8]	]0.4, 0.6]	]0.2, 0.4]	]0, 0.2]

Table 4: similarity between values of a nominative feature

t_j values	$v_{a,j}$	...	$v_{m,j}$	...	$v_{p,j}$
$v_{a,j}$	$x_{v_{a,j},v_{a,j}} = 1$	...	$x_{v_{a,j},v_{m,j}}$	...	$x_{v_{a,j},v_{p,j}}$
...	...	...	...	...	...
$v_{m,j}$	$x_{v_{m,j},v_{a,j}} = x_{v_{a,j},v_{m,j}}$	...	$x_{v_{m,j},v_{m,j}} = 1$	...	$x_{v_{m,j},v_{p,j}}$
...	...	...	...	...	...
$v_{p,j}$	$x_{v_{a,j},v_{a,j}}$	...	$x_{v_{p,j},v_{m,j}} = x_{v_{m,j},v_{p,j}}$	...	$x_{v_{p,j},v_{p,j}} = 1$

Table 5: different configurations of our CBR

Configurations	Case base	Retrieve	Reuse	Revise	Retain
I_NS_CB	Flat case base	Use *simCases* function to retrieve the most similar cases	1. Copy the solution of the similar cases 2. Use rules	Three layers validation	Store the new case if it's coherent in the flat case base
I_S_CB	Segmented case base	Use *simDelegates* algorithm to retrieve the most similar delegates	1. Copy the solution from segments' representatives 2. Use rules	Three layers validation	Store the new case if it's coherent in the segmented case base
R_S_CB	Segmented case base Apply knowledge amplification on it	Use *simDelegates* algorithm to retrieve the most similar delegates	1. Copy the solution from segments' representatives 2. Use rules	Three layers validation	Store the new case if it's coherent in the segmented case base
R_NS_CB	1. One flat case base and 2. One segmented case base Apply knowledge amplification on the segmented case base	Use *simCases* function to retrieve the most similar cases	1. Copy the solution of the similar cases 2. Use rules	Three layers validation	Store the new case if it's coherent in the flat and in the segmented case base

Table 6: results of tests on validation process

	Values	Training Cases (%)	Test Cases (%)
Coherence	Cases found as coherent	100	100
	Cases found as uncoherent	0	0
Stochastic Validity	Cases found with stochastic validity < 0.7	17.05	58.33
	Cases found with stochastic validity between 0.7 and 1	12.62	41.67
	Cases found with stochastic validity = 1	70.33	1.04
Absolute Validity (rules)	Number Cases Set as Valid	68.69	38.54
	Number Cases set as Validity Unknown	31.31	54.17
	Number Cases set as Non-Valid	0	7.29

Table 7: comparison between accuracies of supervised ML tools

Supervised ML Tool	Accuracy Calculated on the Training Set (%)	Accuracy Calculated on the Test Set (%)
Logistic Regression (Hosmer et al., 2013)	77.54	82.29
Support Vector Machine (Byun and Lee, 2002)	79.34	82.29
K Nearest Neighbors (Fukunage and Narendra, 1975)	75.41	82.29
Gaussian Naive Bayes (Jiang and al., 2007)	76.72	82.29
Perceptron (Stephen, 1990)	71.8	76.04
Linear Support Vector Clustering (Ben-Hur et al., 2001)	78.2	82.29
Stochastic Gradient Descent (Bottou, 2012)	74.75	81.25
Decision Tree (Chen, 1995)	92.46	80.21
Random Forest (Verikas et al., 2011)	92.46	85.42
I_NS_CB	98.96	77.08
I_S_CB	93.75	58.33
R_S_CB	**100**	**96.875**
R_NS_CB	**100**	**98.96**

Table 8: comparison between accuracies of the related work

Related Work	Accuracy (%)
GDM-GA-CBR (Yan et al., 2016)	75.45
GA-CBR (Yan et al., 2016)	78.13
WE-CBR (Rezvan et al., 2013)	79.47
DSL-CBR (Rezvan et al., 2013)	79.79
NN (Yan et al., 2016)	80
WEH-CBR (Rezvan et al., 2013)	80.62
DUL-CBR (Rezvan et al., 2013)	82.29
(Smiti and Elouedi, 2018)	90.98
(Fan et al., 2011)	92.7
I_NS_CB	77.08
I_S_CB	58.33
R_S_CB	**96.875**
R_NS_CB	**98.96**

Figure 1: CBR cycle (Aamodt and Plaza 1994)

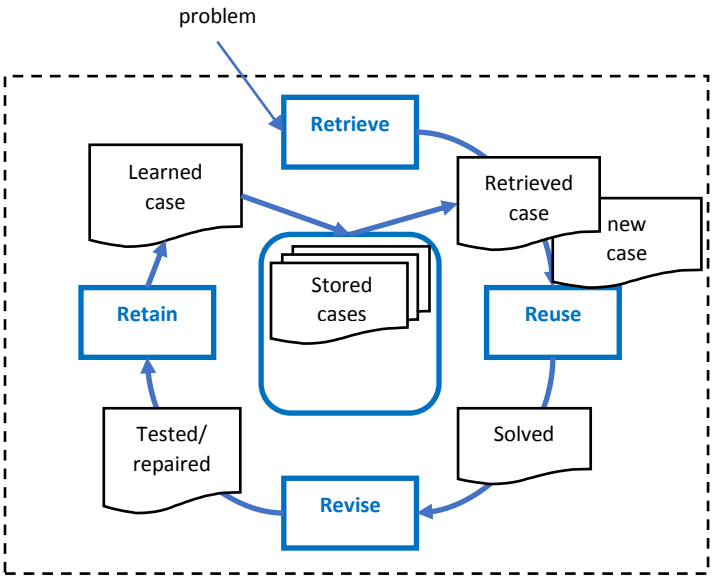


Figure 2: our approach

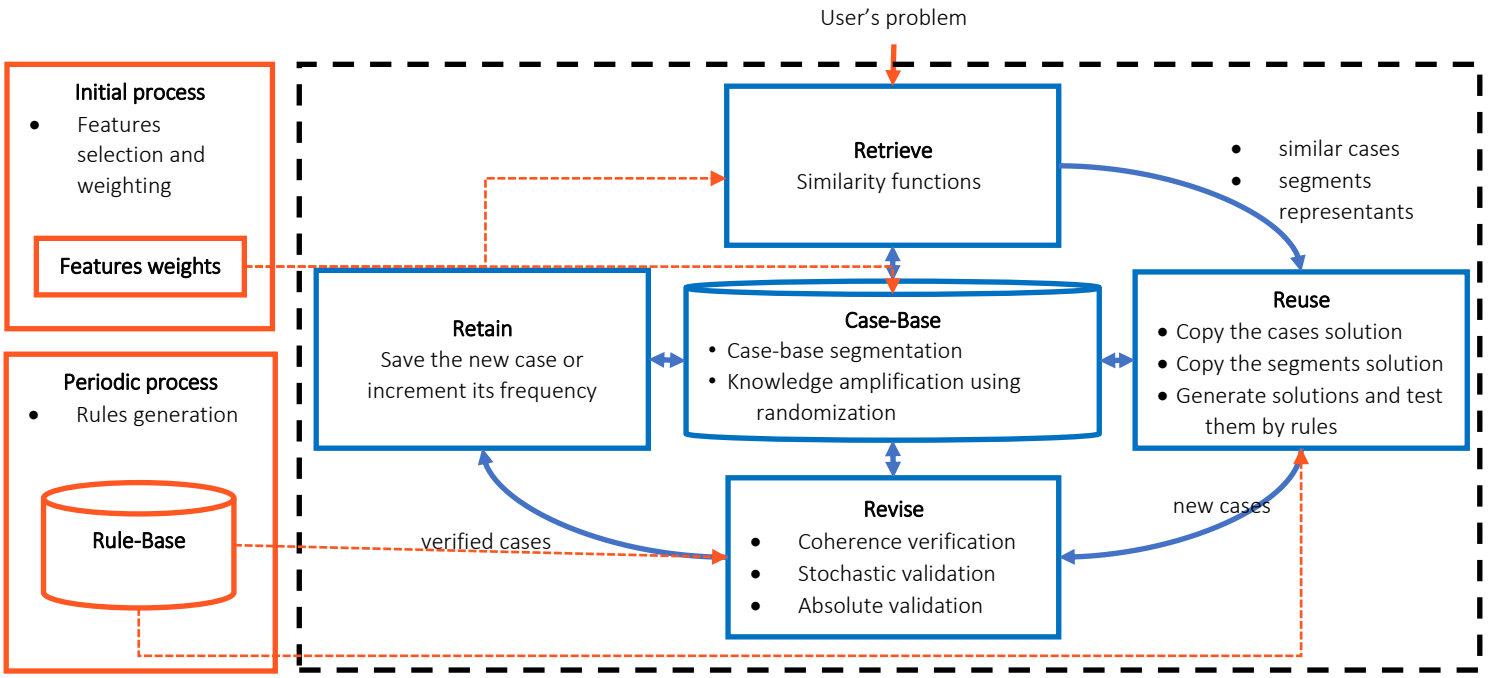


Figure 3: decision tree created from a sample of cases

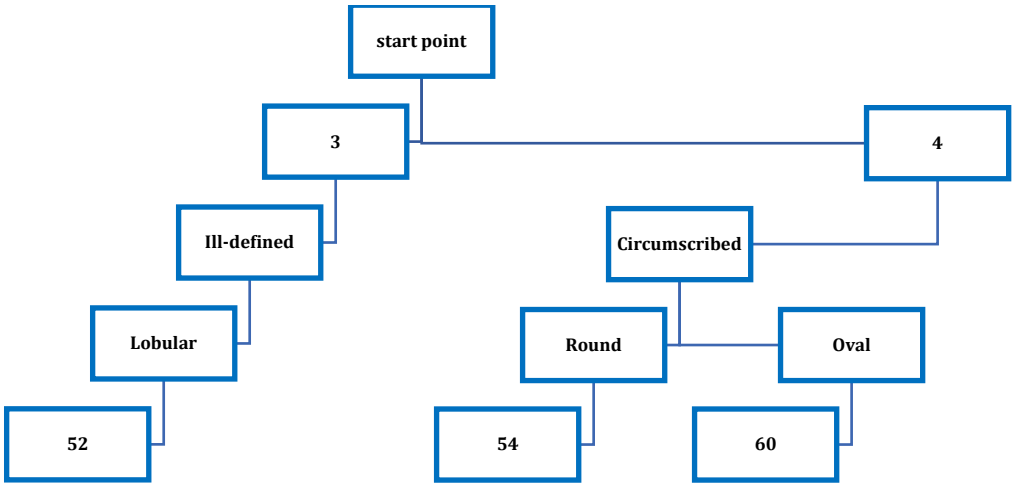


Figure 4: case-base segmentation

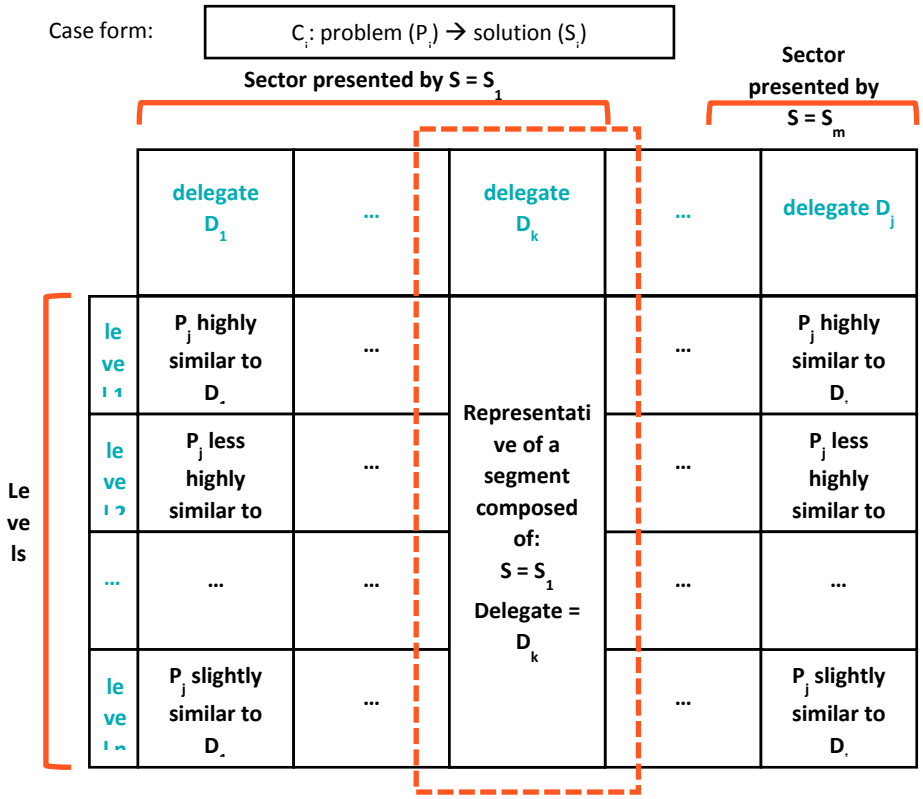


Figure 5: Algorithm of simialrity between a delegate and a case

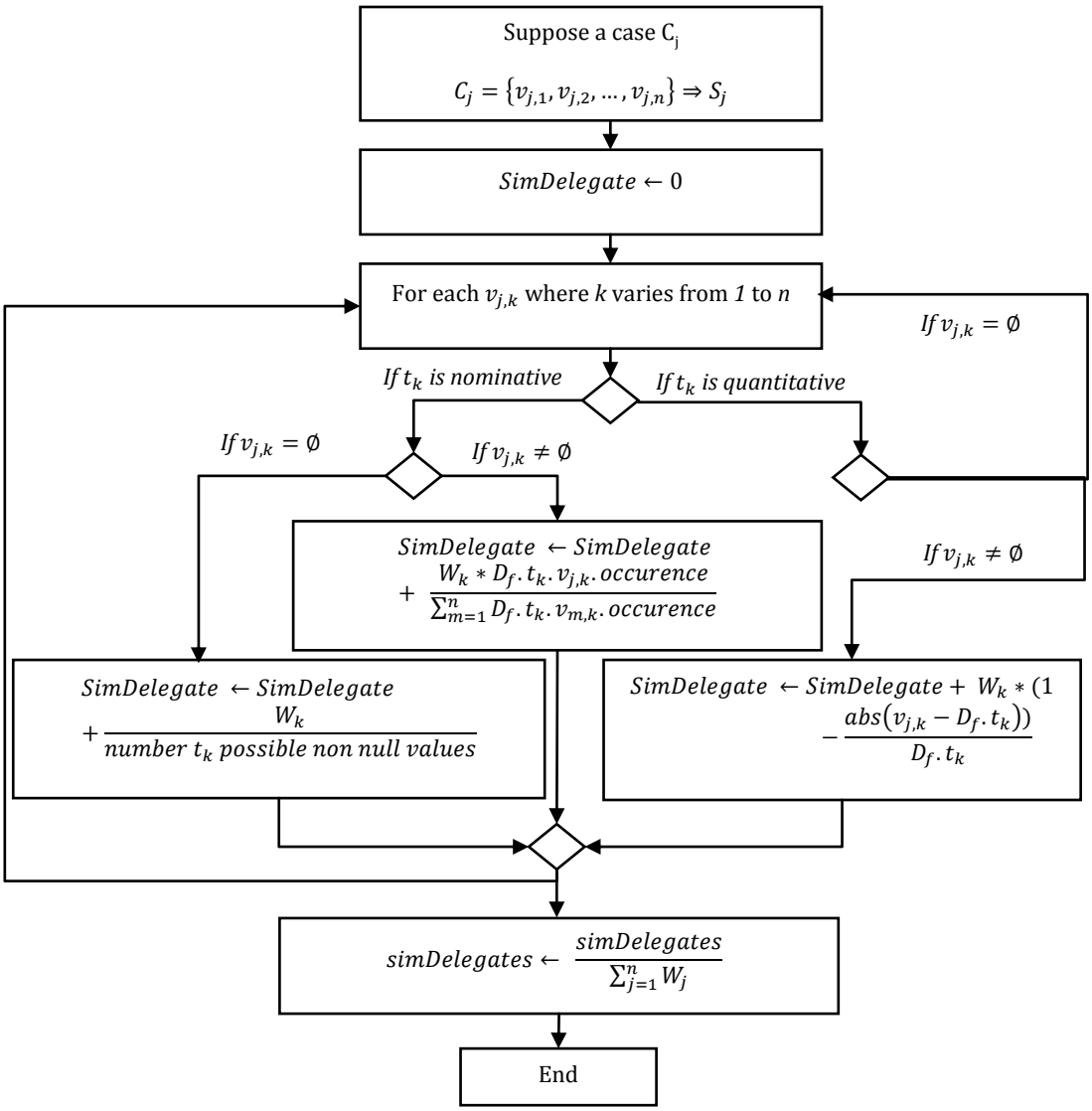
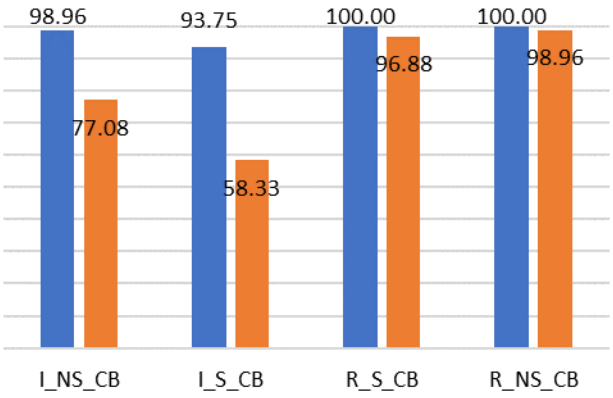
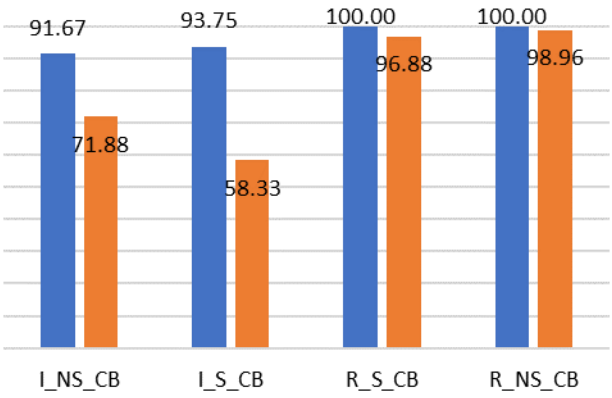


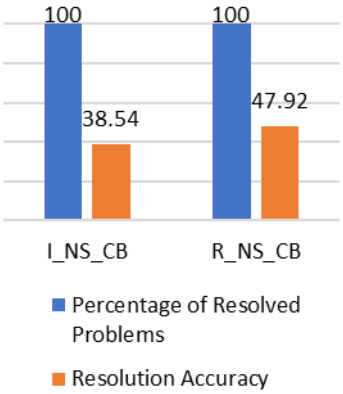
Figure 6: capacity of resolution and accuracy



Similarity threshold = 80%

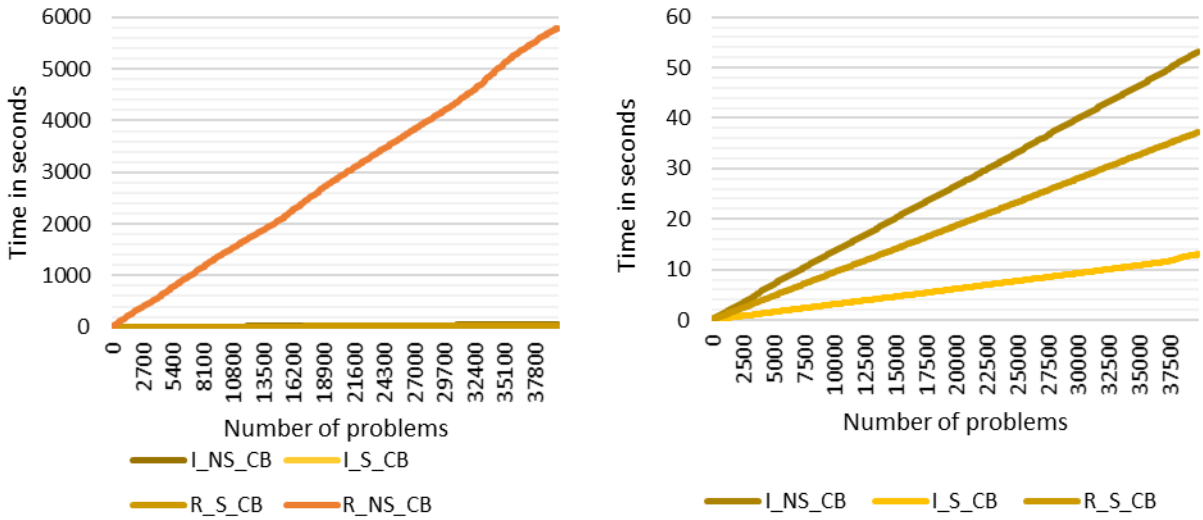


Similarity threshold = 90%

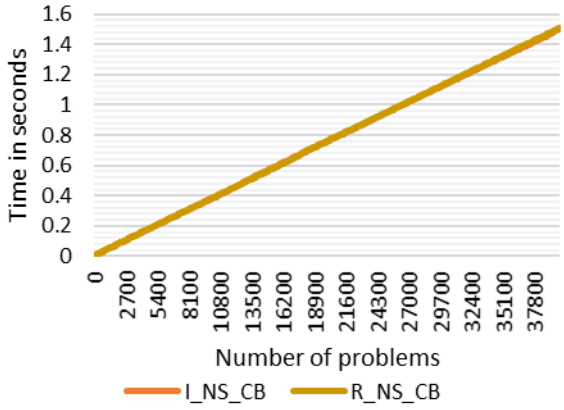


Similarity threshold = 100%

Figure 7: resolution time (in seconds)

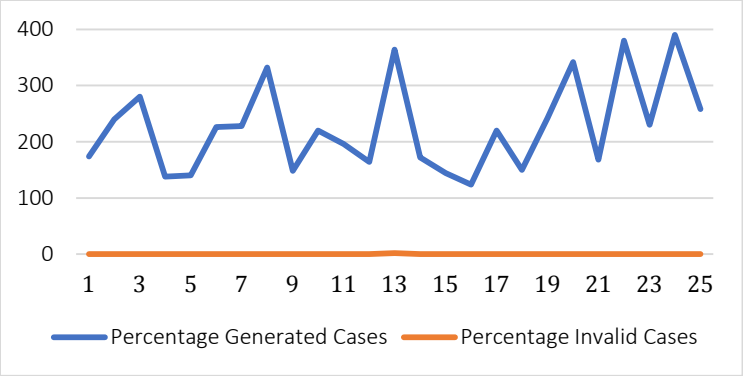


Similarity threshold < 100%



Similarity threshold = 100%

Figure 8: percentage of cases generated by randomization



capacity of resolution and accuracy results

Click here to download Supplementary Material: capacity of resolution and accuracy.xlsx

percentage of cases generation using randomization results

Click here to download Supplementary Material: percentage generation.xlsx

resolution time results

Click here to download Supplementary Material: resolution time.xlsx

declaration of competing interests

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: