Khan *et al. Hum. Cent. Comput. Inf. Sci.* (2019) 9:9
https://doi.org/10.1186/s13673-019-0171-z

Human-centric Computing and Information Sciences

RESEARCH Open Access

Hybrid case-base maintenance approach for modeling large scale case-based reasoning systems

Malik Jahan Khan[1*], Hussain Hayat[1] and Irfan Awan[2]

*Correspondence:
malik.jahan@namal.edu.pk
[1] Department of Computer Science, Namal College, Mianwali, Pakistan
Full list of author information is available at the end of the article

Abstract

Case-based reasoning (CBR) is a nature inspired paradigm of machine learning capable to continuously learn from the past experience. Each newly solved problem and its corresponding solution is retained in its central knowledge repository called case-base. Withρ the regular use of the CBR system, the case-base cardinality keeps on growing. It results into performance bottleneck as the number of comparisons of each new problem with the existing problems also increases with the case-base growth. To address this performance bottleneck, different case-base maintenance (CBM) strategies are used so that the growth of the case-base is controlled without compromising on the utility of knowledge maintained in the case-base. This research work presents a hybrid case-base maintenance approach which equally utilizes the benefits of case addition as well as case deletion strategies to maintain the case-base in online and offline modes respectively. The proposed maintenance method has been evaluated using a simulated model of autonomic forest fire application and its performance has been compared with the existing approaches on a large case-base of the simulated case study.

Keywords: Case-based reasoning, Lazy machine learning, Soft-computing, Case-base maintenance

Introduction

Case-based reasoning (CBR) is one of the widely used lazy machine learning methods inspired by natural learning behavior towards solving a new problem [1, 9, 22, 50]. In lazy machine learning methods, training of the model evolves with time and presentation of more data. As the new data is presented, it continues contributing in improvement of the learning curve of the system. CBR is implemented as a learning layer onto the problem domain. Each new problem instance which needs to be resolved through this learning method is presented as a case. It is matched with the cases present in the knowledge repository called case-base. Set of the nearest neighbors of a new problem is extracted using a selected similarity measure. Solutions of the retrieved nearest neighbors of the case at hand are combined together appropriately to figure out the new solution. It may undergo a fine tuning exercise if the need arises. Finally, the new case comprising of the problem and its relative solution is added into the existing

case-base. This leads to continuous increase in the case-base cardinality which helps to improve its problem solving capability. On the other hand, the time complexity for computing solution of a new problem increases and the overall performance of the decision support system gets compromised [28, 44]. To address this performance bottleneck of a CBR system, the existing case-base needs to be maintained without compromising its problem solving capability [28]. This exercise is known as case-base maintenance (CBM).

The fundamental purpose of CBM is to delete, add, update cases and the associated data and meta-data in the case-base to ensure the robustness of a CBR system [52]. Robustness of a CBR system can be weighed on the following criteria [43]:

- Efficiency: Time to solve a problem.
- Competence: Problem count that can be solved.
- Quality: Accuracy of the proposed solution.

Different approaches have been introduced for maintaining a CBR system. The maintenance of a case-base is triggered depending on the type of maintenance policy. These triggering times can be categorized in three types [21]:

- Continuous timing.
- Conditional timing.
- Ad hoc timing.

Continuous maintenance policies are referred as online processing maintenance policies. For example, a policy that is triggered after each problem solving cycle is referred as continuous timing maintenance policy. Conditional maintenance policies are triggered whenever the case-base cardinality reaches some predefined threshold. Ad hoc maintenance policies are the maintenance tests on the case-base initiated by system administrators to determine whether there is a need for maintenance [21]. There are two main CBM strategies:

- Case addition strategy.
- Case deletion strategy.

There exist conditional, ad hoc as well as continuous timing case addition policies. For instance, Smyth and McKenna proposed a case addition policy in which a completely resolved case is blindly added in the case repository [46]. Then, after some time when there is a need for maintenance, a new case-base is created which consists only those cases that meet some pre-defined criteria. On the other hand, Munoz-Avila presented a retrieval based case addition policy which is continuous in timing and adds a case if the guidance of retrieved cases was useless [35].

Most of the case deletion strategies have conditional timing. For instance, Smyth and Keane proposed a case deletion strategy in which blind addition of new case is done [45]. Moreover, when the case-base cardinality is equal to the swamping limit, harmful and useless cases are no more part of the case repository.

Both techniques have certain limitations:

- In case addition methodologies with continuous or ad-hoc timing, cases can be added only in the case-base. There is no criteria for deleting cases from the case-base as this could lead to retrieval time bottlenecks with the ongoing increase in the case-base cardinality.
- In addition and deletion policies with conditional timing, the whole process of maintenance tends to recur quite often due to the blind addition of completely resolved case. This results in performance bottlenecks.

To address these challenges, a hybrid case-base maintenance model has been proposed in this paper. In this model, we have suggested a continuous timing addition policy as well as conditional timing deletion policy. By using this approach, all the three criteria which can be used to weigh performance of a CBR system are preserved including efficiency, competence and quality. The proposed method has been implemented and evaluated on a sufficiently large synthetic case-base of the simulated model of autonomic forest fire application. The proposed approach has been evaluated and its performance has been compared with the existing benchmarks and its contributions have been discussed. The proposed approach outperforms the existing maintenance strategies. Its key contributions have been highlighted as follows:

- The proposed approach is capable to control the case-base cardinality at a user preferred threshold.
- It acquires knowledge and preserves the case-base competence using online and offline case maintenance mechanisms.
- The resultant case-base from the proposed approach is more compact and equally competent as compared to the the existing benchmarks. Its average problem solving time offers up to 65% reduction as compared to that of the standard CBR cycle without compromising on its performance.
- Over time, it strengthens the existing stronger cases through updating their utility and periodically removes the weaker cases.

The remaining sections of this work have been structured as follows: an overview on how CBR process works has been presented in "Case based reasoning (CBR)" section. This section also highlights the problems that are faced while using a classical CBR model. The need for case-base maintenance (CBM) and its associated methods have also been discussed in this section. In "Related work" section, CBM policies have been discussed along with their limitations. "Proposed hybrid CBM approach" section is comprised of the proposed methodology and its significance in addressing the CBM challenges. In "Case study, experimental settings and initial results" section, the implementation process and experimental settings have been explained. "Case study, experimental settings and initial results" section also consists of initial results of the proposed approach after performing different experiments. Comparison of the results obtained by the implementation of proposed approach and existing benchmarks has been analyzed in "Benchmarking with state-of-the-art approaches" section. The last section derives the conclusion of this research work highlighting some promising future directions.

Case based reasoning (CBR)

CBR cycle

CBR is a nature inspired problem solving methodology. It uses already solved problems to solve a new one. Its working principle is reasoning by remembering. This principle implies that the reasoning used to solve a target problem is remembered. The first working principle of CBR is that similar problems have alike solutions i.e. to solve a new case, the existing cases and their solutions from the case repository are used. The second principle is that the kind of problems which an agent faces tends to repeat. Thus, there is similarity between future and current problems. Therefore, it is worth to remember and reuse [22]. This leads to construction of the case-base which contains completely resolved cases called cases.

For the purpose of solving complex and complicated problems, the applications of CBR are widely distributed directly or indirectly in almost every field of knowledge. It has been effectively used for adoption problem in medical CBR systems [5], e-commerce [39], data mining [25], autonomic systems [19], travel planning [6], software engineering [7, 18] and an enormous range of other applications [34, 48]. Furthermore, CBR can also be used to mine big and sparse data as well [37].

CBR is preferred over other artificial intelligence and rule-based systems because of many reasons. One of these reasons is its minimal learning requirements. Secondly, its pre-processing cost is relatively lower than other techniques. Thirdly, and most importantly it has very flexible adoption and case representation process [30, 40]. A lot of research has also been done on preference based CBR systems in which solution of the new problem is derived based on the preferences which are defined differently in different contexts [2, 3]. CBR cycle presented in Fig. 1 has been explained as follows:

Retrieve phase

In CBR cycle, finding those cases which can perfectly represent a target problem is an essential task along with finding the similarity between two cases [31]. CBRs efficiency

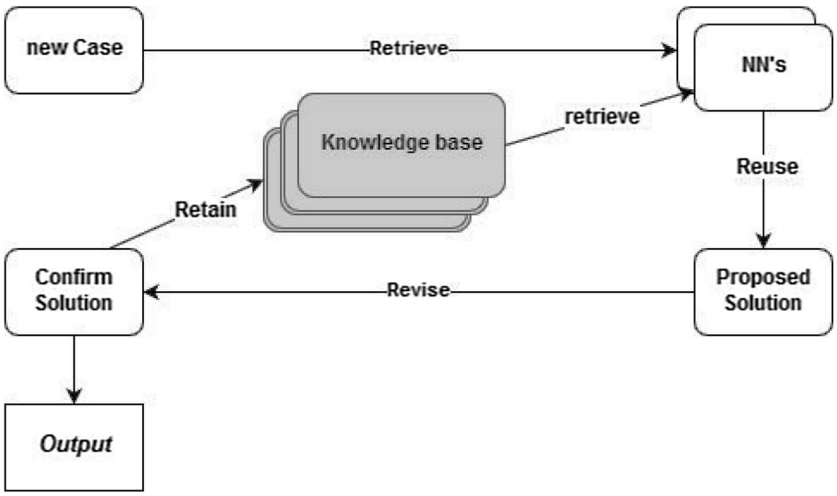

Fig. 1 The CBR cycle

is dependent on improved retrieval performance. The feature vector representation of cases provided in the case description is used to compute the surface distance between two cases. Usually, distance functions are used as a measure of dissimilarity. Inverse dissimilarity is connotated as similarity between two cases, as shown in Table 1. These similarity measures capture surface similarity between two cases. Structural similarity between cases can also be captured using graph theoretic representation and similarity measures [11–13, 47, 49].

Reuse and revise phases

Retrieved nearest cases are used to calculate the solution. In classification problems, the solution of the nearest retrieved case will most likely be returned because the frequency of recurring solution to similar cases is higher. On the other hand, one might have to come up with adoption criterion for devising the solution of problems other than classification [31].

Retain phase

Finally, decision is made if the new completely resolved case should be retained in the existing knowledge. The outcome of possible repair and evaluation triggers the learning from the outcome of the proposed solution. It is important that the case-base maintains a bare minimum size to capture the problem domain clearly. Due to the retain phase, each new success is added in the case-base and the size grows on continuous basis which may result in the performance bottleneck. Case-base needs to be optimized to make sure that the case-base cardinality does not grow beyond an unacceptable limit and at the same time, it does not shrink too much to affect the span of the problem domain. Case-base maintenance takes care of this important factor.

Case-base maintenance (CBM)

Two important considerations play vital role in effectiveness of a CBR system [41]:

1. Case data, i.e., it is more complicated than to just learn from completely resolved case.

Table 1 Sample surface (dis)similarity measures [10, 19, 20, 26, 36]

Distance function	Formula
Euclidian distance	$dis(i,j) = \sqrt{\sum_{k=1}^{m} (w_k(p_{ik} - p_{jk}))^2}$
Manhattan distance	$dis(i,j) = \sum_{k=1}^{m} w_k \lvert p_{ik} - p_{jk} \rvert$
Canberra distance	$dis(i,j) = \sum_{k=1}^{m} w_k \frac{\lvert p_{ik} - p_{jk} \rvert}{\lvert p_{ik} \rvert + \lvert p_{jk} \rvert}$
Squared chord distance	$dis(i,j) = \sum_{k=1}^{m} w_k \left(\sqrt{p_{ik}} - \sqrt{p_{jk}}\right)^2$
Minkowski distance	$dis(i,j) = \left[\sum_{k=1}^{m} w_k \lvert p_{ik} - p_{jk} \rvert^q\right]^{\frac{1}{q}}$

2. Retrieval process time, i.e., the complexity of retrieval process increases in terms of time as size of knowledge base increases.

In a CBR system, the case-base cardinality increases due to the retention of new problem-solving pair. As a result, the overall performance of CBR system degrades. There are three main attributes which affect the performance of a CBR system [43].

Quality

The correctness of the derived solution is one of the major aspects which represents the performance of a CBR system. However, with the increase in size of a typical CBR system, this factor is not affected since more knowledge is always better to produce better results. But on the other hand, while maintaining a case-base by adding or removing cases, this factor may get compromised. Therefore, an acceptable criterion of adding or deleting cases should be deployed to avoid the removal of important cases and to generate a satisfactory solution of desired quality [43].

Competence

The number of successful solutions of target problems proposed by a case-base is termed as its competence. There is a strong relation between case-base competence and its cases. Coverage and reachability are two important factors on which the case-base competence is dependent [43].

Coverage The number of new solvable cases by using an individual case is termed as its coverage. Coverage is defined as [43]:

$$Coverage(a \in A) = \{a' \in A : solves(a, a')\}, \tag{1}$$

where

$$solves(a, a') = matches(a, a') \cap adapts(a, a'), \tag{2}$$

Reachability Reachability of the problem at hand is the set of cases capable to formulate its solution. Reachability is defined as [43].

$$Reachability(a \in A) = \{a' \in A : solves(a', a)\}, \tag{3}$$

Efficiency

The average time taken to solve a new problem is defined as the efficiency of a CBR system. In case of a classical CBR system, the size of knowledge-base tends to increase with the passage of time due to continuous retention of cases. When the case-base cardinality gets large enough, this leads to serious time complexity bottleneck for the case retrieval process of the CBR system. To comprehend this problematic situation, one approach is to decrease the size of CB by using different maintenance techniques. Performance of a CBR system cannot be compromised and thus it raises the need for a maintenance strategy. One way of doing that is by decreasing the size of the knowledge-base by using an appropriate CBM strategy.

Related work

Condensed nearest neighbor method (CNN) is an approach used in instance-based learning and nearest neighbor methods for the purpose of editing training data. An edited set of training examples are generated by CNN method which is consistent

with the unedited original training data [15]. CNN is further extended as reduced edited nearest neighbor method in [8]. The noisy cases are removed, which belong to a class different than the majority of their NN's [8].

Smyth and McKenna introduced an addition policy for creating a compact competent case base [46]. They combined relative coverage (RC) with condensed nearest neighbor (CNN) method in such a way that all cases were arranged in descending order according to their RC value and then applied condensed nearest neighbor algorithm to create a compact competent case-base. The approximate running time of their approach is $O(n)^2$ where n is number of existing cases [46].

Leake and Wilson proposed a case addition procedure which adds cases on the basis of the performance benefit (PB) that they provide by their retention in the case-base. Relative performance (RP) measure and CNN were used for performance guided maintenance. The running time of this approach is $O(n^2)$ [23].

In [35] Munoz-Avila presented a case retention technique which decides whether the retrieved cases could provide any useful guidance. If not, then the new case does not need to be retrained in the case-base. However, if guidance of the cases retrieved is harmful then the new case needs to be stored in the case-base [35].

In [51], a new case-selection policy has been presented which is based on a streaming criteria for addition. A loss function is used to make a decision on usefulness of the case at hand.

Eager case retention policy is inherited from CBR problem solving cycle [1, 24, 35]. It is a permissive policy in which every new completely resolved case is retained in the case-base. Due to a very permissive case retention criteria, this policy leads to a very large case-base quickly and results in performance bottlenecks.

In [16], a case retention policy is proposed where the retrieved cases are extended to provide the solution during the adaptation process and new case is added to the existing case-base. However, if parts of retrieved cases are revised to provide the solution then the new problem does not need to be retained in the case-base.

A case-deletion procedure has been defined to categorize the cases into four different categories on the basis of their coverage and reachability values in [45]. The computational complexity of this approach is $O(n^2)$ for n existing cases [45]. Moreover, it uses a learning heuristic algorithm to update the case categories whenever a new case is learned.

In [42], a deletion policy has been presented to keep a compact case-base. This approach uses mixture of different methods to decide the deletion step. These methods include feature weighting, outlier detection and clustering.

In [32], Markovitch proposed random deletion technique which is completely dependent on the domain knowledge. In this technique, cases are randomly selected and deleted from the case-base once it has reached some predefined limit. This policy works surprisingly well, but it can also be very destructive because important cases can also get deleted which will result in an unrecoverable loss in the case-base competence.

Another deletion policy suggests to delete cases based on their retrieval frequency [33]. This policy calculates the retrieval frequency of all the cases and then deletes those cases which are not accessed frequently.

Ad-hoc timing deletion policy conducts a number of tests on all cases of the case-base for the detection of redundancy and inconsistency [38]. It asks for a user approval for the deletion of detected cases from administrators.

Another policy divides spanning cases into two types i.e. inter-category and intra-category spanning cases [14]. According to this categorization, inter-category spanning cases for a given category are those cases which are partially reachable by an appropriate case of a different category. On the other hand, if a case is being represented in parts by another case which belongs to the same category then that case is an intra-category spanning case. For the suppression of the case-base, these categorized cases are deleted on the basis of their Competence Metric (*CM*) value. *CM* is defined as follows:

$$CM(a) = \frac{|V_c(a)|}{|V_r(a)|}, \tag{4}$$

where V_c is a covering set and V_r is a reachability set for a given case a.

Once the case-base cardinality reaches the swamping limit, cases are treated and deleted. Initially, all the auxiliary cases will be deleted. Then the support cases with relatively lower competence metric will be removed, while retaining the most significant case from each group based on the competence metric value. Afterward, all the intra-category spanning cases will be removed. This process will continue until each case covers only itself among the existing cases in its category.

There also exist some other algorithms like evolutionary algorithms and clustering using random forests which can also be used to maintain a CBR system as mentioned in [4, 17, 27–29].

The literature review reveals that the existing CBM techniques have following drawbacks:

- The existing approaches either focus on case addition or case deletion.
- In case addition policies, only new completely resolved case is added which looks important according to the existing case-base. However, the importance of that new case may degrade with the addition of new cases.
- These addition policies are unable to delete cases amplifying the gradual increase in the case-base cardinality which will eventually degrade the performance of a CBR system.
- On the other hand, the case deletion techniques encourage blind addition of new cases and after reaching a swamping limit, unimportant cases are deleted from the case-base.
- Due to the blind retention of new cases in deletion approaches, the case-base cardinality grows swiftly resulting in frequent need for complex maintenance. It results in degradation of the CBR process.

This work undertakes the above limitations as the research problem and addresses the problem through a hybrid approach which is more robust and efficient in terms of the average problem solving time observed by the CBR cycle and case-base competence to solve new problems.

Proposed hybrid CBM approach

Existing addition and deletion policies may degrade performance of a CBR system as identified and highlighted in the literature review discussed in the previous section. In this paper, a new hybrid CBM technique is proposed to preserve the performance of a CBR system.

The objective is to develop an effective case-base A of n cases. Each case is a collections of different variables representing the problem domain. The case is formulated through the observed system state which provides values for different variables. A set of m variables represented as a case has been shown below:

$$a_i = \{a_{i1}, a_{i2}, a_{i3}, \ldots, a_{im}\}, \tag{5}$$

The complete case-base X is represented as follows:

$$A = \{a_1, a_2, a_3, \ldots, a_n\}, \tag{6}$$

With life time of the system, number of problems observed by the intelligent modeled system increases. It results in the larger case-base cardinality and causes retrieval efficiency bottleneck for CBR system. Each new problem is transformed into a case and its solution is computed using CBR building block. Whenever a new problem is presented and its solution is derived by the CBR system, decision is made whether that completely resolved case should be retained in the case-base. This work proposes an improved hybrid approach to find out a set of good representatives of cases from the given larger set of cases. Whenever a better case is found, this approach suggests to add it into the list of selected cases and at the same time, it may delete the weaker cases having lesser contribution in the decision making capability of the CBR system. A case retention algorithm has been proposed to perform addition and conditional deletion operations in parallel. It exploits benefits of both the strategies and minimizes the bias of selection of either of the two options. The proposed method has been implemented as a maintenance component of the CBR based decision support system. This approach exploits inclusion and exclusion strategies to maintain the case-base. For inclusion purposes, conventional CBR algorithm is applied to find the solution of a test case using the training case-base. Error is computed and compared with the preset threshold. If it does not exceed the threshold then the new case may be retained. After the decision of the inclusion of a new case, conditional exclusion strategy is applied. This work uses a clustering based deletion policy which works as another building block of the proposed algorithm. We use a utility metric to keep track of the usage pattern of each case. It provides an insight about the frequency of utilization of the case under consideration. All cases are classified into pivotal, auxiliary, span and support cases based on the clusters and utility pattern. This process is completed by computing the relative coverage (RC) value of the new problem as shown in Figs. 2, 3 and 4. RC is a measure which is used to calculate the individual case competence [46].

$$RC(a) = \sum_{a' \in V_c(a)} \frac{1}{|V_r(a')|}, \tag{7}$$

The RC value is compared with a predefined threshold. A new case will be added in the case-base, if its RC value is greater or equal to the generated threshold or if that case

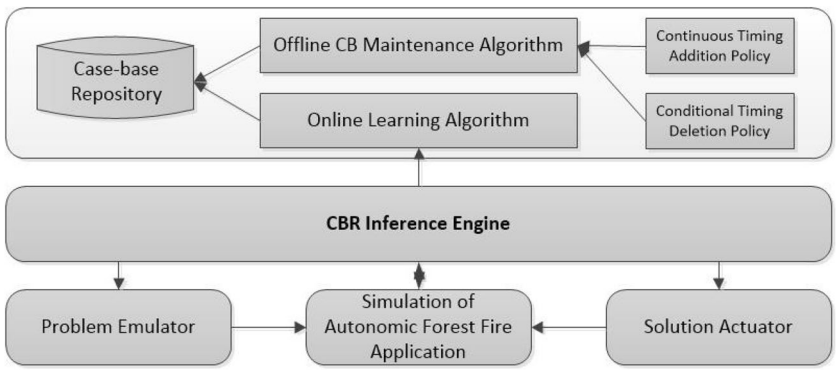

Fig. 2 Architectural model of simulated case-base maintenance approach

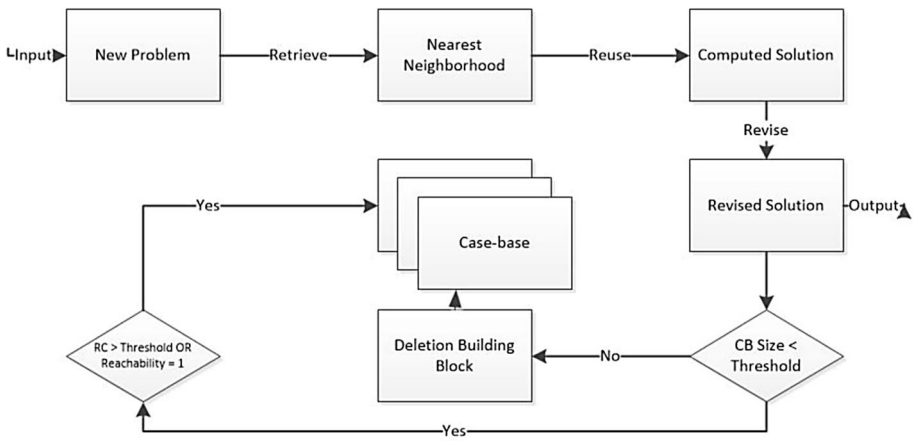

Fig. 3 Proposed hybrid approach of case decision

is only reachable by itself. These criteria have been set so that only the most competent cases according to the live case-base are retained in the case-base. After the addition of new case, threshold will be updated. This online algorithm has to compute *RC* of each new problem for which coverage and reachability of that case needs to be computed. Since, for computing coverage set of new problem, its presence in the nearest neighbors of each case is checked. It results in the computational complexity of $O(n(n))$, where n is the case-base cardinality. The computational complexity of computing *RC* value in the worst case is $O(n)$. So, the overall complexity becomes $O(n + n^2)$ leading to $O(n^2)$. Whenever size of the case-base exceeds the pre-set threshold, all the cases will be divided into four different categories on the basis of their coverage and reachability values. These four categories are described below as defined in [45]:

- *Auxiliary cases* By the removal of auxiliary cases, the competence does not get effected. If the coverage provided by a case is absorbed by one of its reachable cases then that case is an auxiliary case. For instance in Fig. 5, cases *a*, *b* and *c* are auxiliary cases, since the coverage they provide is subsumed by their reachable cases such as case *y*.

Input: Existing case-base, new problem *p*, swampingLimit, threshold
Output: Compact case-base
Basic Idea:
For each new problem *p*
1. If(size(case-base) $\leq$ *swampingLimit*)
   If(RC(*p*) $\geq$ *threshold*)
      case-base = case-base $\cup$ <*p*, sol(*p*)>
   Else
      Exit
2. Else
   a) Compute *RC* of all existing cases
   b) Classify them into pivotal, support, spanning and auxiliary cases
   c) Delete non-pivotal cases with the lowest *RC* in accordance to case categories
   d) Update *RC* threshold for online maintenance

Fig. 4 Proposed algorithm of case selection

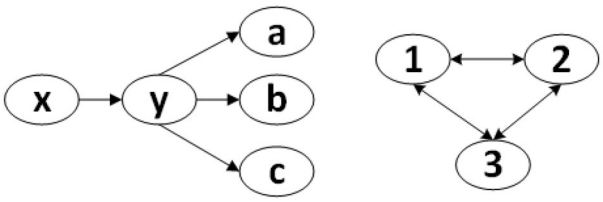

Fig. 5 Case categories

- *Pivotal cases* If a case is only reachable by itself, then that case is a pivotal case. The competence of a system degrades by the deletion of a pivotal case. In Fig. 5, case *x* is a pivotal case since it is reachable by only itself.
- *Spanning cases* The cases with an overlapping coverage space with other cases within the problem space are categorised as spanning cases. In Fig. 5, case *y* is a spanning case as it links together the problem space between case *x* and cases *a*, *b* and *c*.
- *Support cases* The cases which exist in the form of groups and act as spanning cases within the group are known as support cases. In Fig. 5, cases 1, 2 and 3 are support cases and as a whole they represent a support group. The removal of a member of the support category does not harm competence at all. However, deletion of a whole category is as deletion of a pivotal case.

Cases with the lowest *RC* value are selected and deleted according to this pattern. This process will proceed until all non-pivotal cases are no more part of the case-base. Since this offline algorithm also involves the concept of computing coverage of cases, therefore the

computational complexity of this approach is $O(n^2)$. However, it does not harm the overall performance of CBR system because of its non-frequent and off-line nature.

The significance of the proposed approach over existing techniques is its ability to manage the growth and development of the case-base while enhancing the robustness of the CBR system. The growth of case-base is controlled by an online algorithm which adds only the most competent new cases and the case-base cardinality is maintained by an offline deletion algorithm which deletes the least competent cases.

Case study, experimental settings and initial results

The proposed architecture along with two other maintenance techniques, i.e., RC-CNN and footprint deletion have been implemented and evaluated on different training and testing distributions of simulated autonomic forest fire application (AFFA) [19]. Confusion matrix shown in Fig. 6 is computed to evaluate the performance of each method. Performance of the proposed approach has been computed with the selected benchmarks. Accuracy, precision and recall have been used as evaluation metrics and are computed using the following equations:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}, \tag{8}$$

$$Precision = \frac{TP}{TP + FP}, \tag{9}$$

$$Recall = \frac{TP}{TP + FN}, \tag{10}$$

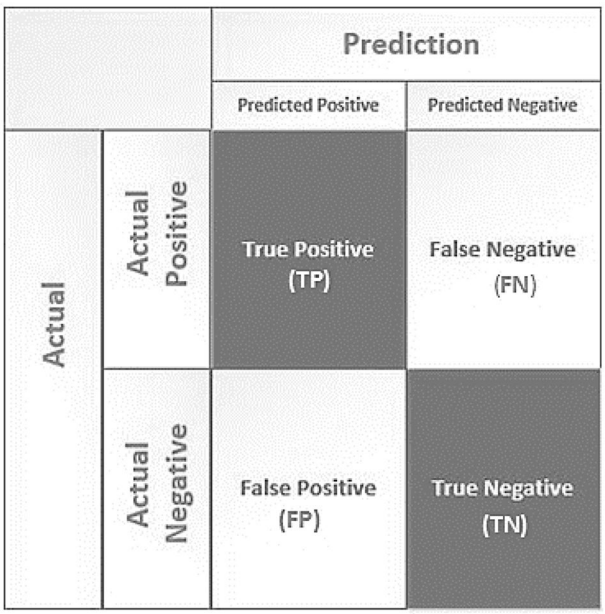

Fig. 6 Confusion matrix

where *TP* is number of positive cases which have been classified correctly, *TN* is number of negative cases which have been classified correctly, *FP* is number of positive cases which have been classified incorrectly and *FN* is number of negative cases which have been classified incorrectly.

Euclidean distance has been computed to compare two cases and inverse distance has been used as a similarity measure:

$$dis(i,j) = \sqrt{\sum_{k=1}^{m}(w_k(a_{ik} - a_{jk}))^2}, \tag{11}$$

where $dis(i, j)$ represents the dissimilarity between ith and jth cases, a_{ik} and a_{jk} are the kth attributes of cases i and j respectively. The weight of the kth attribute is represented by w_k. Similarity between two cases is computed using the inverse distance:

$$sim(i,j) = \begin{cases} \frac{1}{dis(i,j)}, & \text{if } dis(i,j) \neq 0 \\ 1, & \text{otherwise,} \end{cases} \tag{12}$$

Weighted average is used for deriving the solution based on the similarity value of each potential neighbour:

$$sol(j) = \frac{\sum_{i=1}^{n} sim(i,j).sol(i)}{\sum_{i=1}^{n} sim(i,j)}, \tag{13}$$

Case study: Simulated model of autonomic forest fire application

Autonomic forest fire application [19] models spread of fire in a 2D grid. Its environmental variables include wind direction (*WD*), minimum distance from the burning cell (*MD*), wind intensity (*WI*), speed of fire (*FS*) and number of cells exposed to fire (*N*). Configuration script is used as class label. With the help of these parameters, the probability that a certain cell will be caught by fire is predicted. For experimental purposes, this dataset containing 10,000 instances was divided in three different partitions as shown in Table 2. The data was captured through the simulated runs of the application using its environmental variables.

For RC-CNN and footprint deletion maintenance techniques, case-base was allowed to learn 1000 cases for each data partitioning before triggering maintenance. However, in case of hybrid algorithm only the most competent cases were allowed to be learnt. Therefore, this learning limit was reduced for this algorithm for each data partitioning in a way that 500 cases were allowed to be learnt when the case-base cardinality was set 3000 and problem space was 7000. When the case-base cardinality and problem space was 5000 then 400 cases were allowed to be learnt. For the data partitioning of 70:30, 300 cases were allowed to be learnt. The results obtained by the application of each

Table 2 Data partitioning

Distributions	Case-base	Problem space
Distribution 1	3000	7000
Distribution 2	5000	5000
Distribution 3	7000	3000

technique on different distribution of dataset have been presented in different experimental settings.

Optimal nearest neighbors

To compute the solution, the set of the nearest neighbors (*NN*) is retrieved. The size of *NN* is determined by varying the number of the nearest neighbors from 1 to the case-base cardinality and is notated as *k*. The value of *k* resulting in maximum accuracy is obtained as optimal number of the nearest neighbors. It is computed through exhaustive search based on the accuracy. Actual value of *k* does not hinder the performance. This value of *k* is determined for each CBM technique on each data partition.

Optimal value of NN's for simple CBR cycle

The optimal value of *k* nearest neighbors has been derived on each data partition in the same manner as described above. The results obtained have been shown in Figs. 7, 8 and 9. In Figs. 7, 8 and 9, x-axis represents the value of *k* representing size of *NN* and the y-axis represents the percentage of accuracy, precision and recall. It has been observed that after a particular value of *k*, accuracy for each partitioning tends to become consistent or decline. Therefore, that certain value is accounted as optimal value of *k*. Exact results of optimal *k* have been shown in Table 3.

Optimal value of NN's for RC-CNN

The value of *k* required to obtain maximum accuracy using RC-CNN maintenance technique on each data partitions has been presented in Figs. 10, 11 and 12.

Exact performance on optimal *k* is summarized in Table 4.

Optimal value of NN's for footprint deletion

The optimal value of *k* nearest neighbors have been derived for footprint deletion CBM technique and the results have been shown in Figs. 13, 14 and 15.

Exact performance of optimal *k* for each partitioning have been given in Table 5.

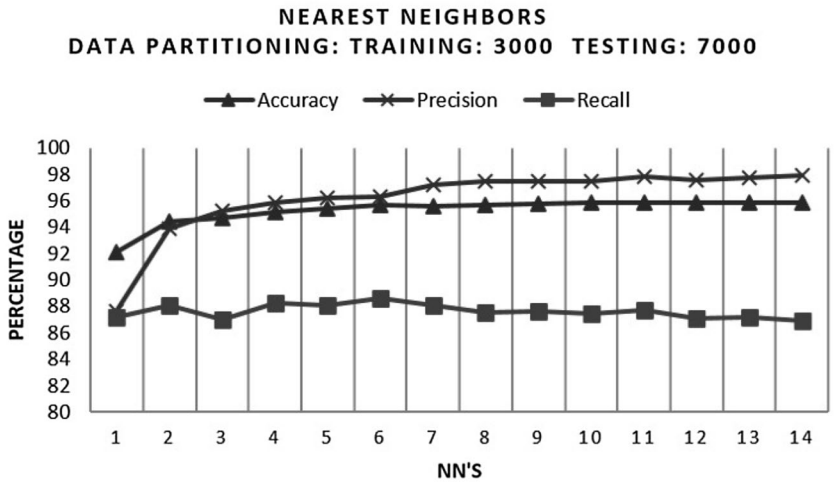

Fig. 7 Optimal NN's for CBR cycle: data distribution 30:70

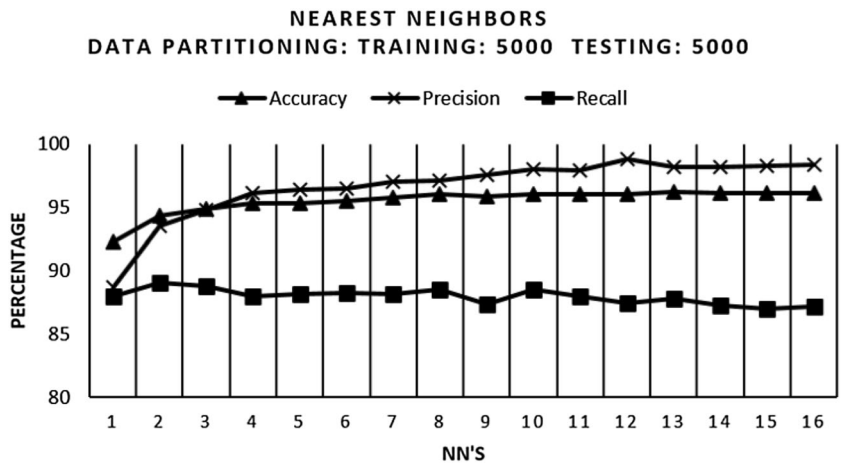

Fig. 8 Optimal NN's for CBR cycle: data distribution 50:50

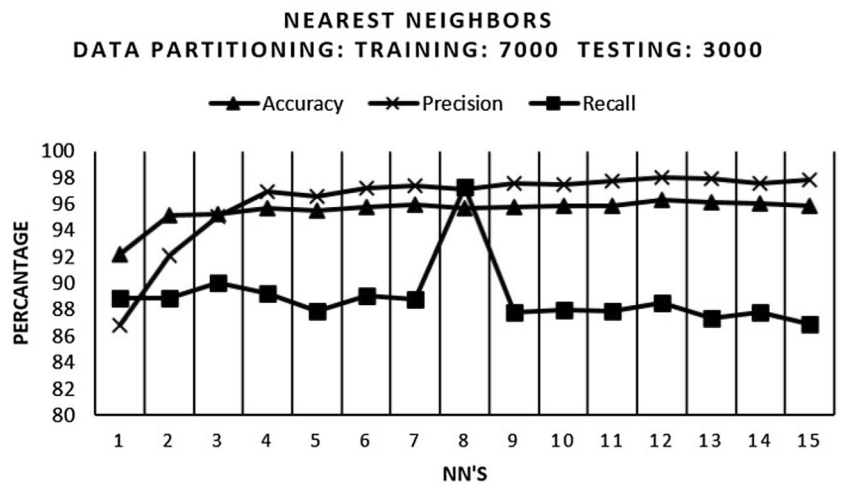

Fig. 9 Optimal NN's for CBR cycle: data distribution 70:30

Table 3 Optimal NN's for CBR cycle

Training	Testing	Optimal *NN* size	Accuracy (%)	Precision (%)	Recall (%)
3000	7000	11	95.9	97.8	87.7
5000	5000	13	96.2	98.2	87.8
7000	3000	12	96.3	98	88.5

Optimal value of NN's for the proposed hybrid approach

The number of nearest neighbors required by the proposed hybrid CBM approach have been determined in the same way as mentioned earlier and shown in Figs. 16, 17 and 18. The optimal nearest numbers for each data partitioning have been presented in Table 6. The highest accuracy achieved in this experimental setting is up to 96%. It has resulted in the selection of optimal set of cases using the hybrid approach. In the next section, its

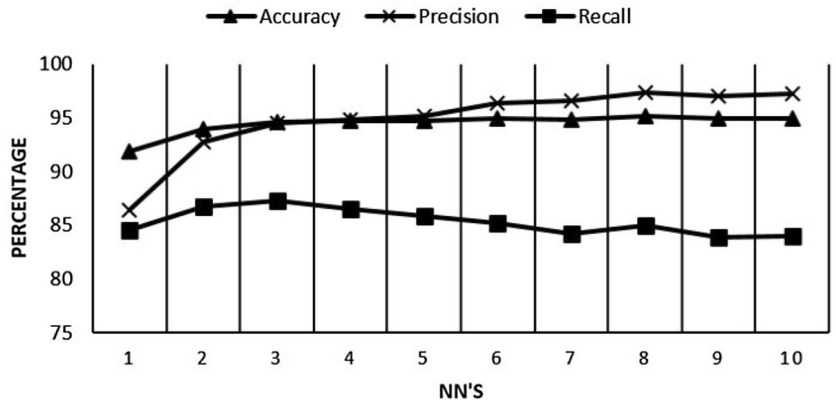

Fig. 10 Optimal NN's for RC-CNN: data distribution 30:70

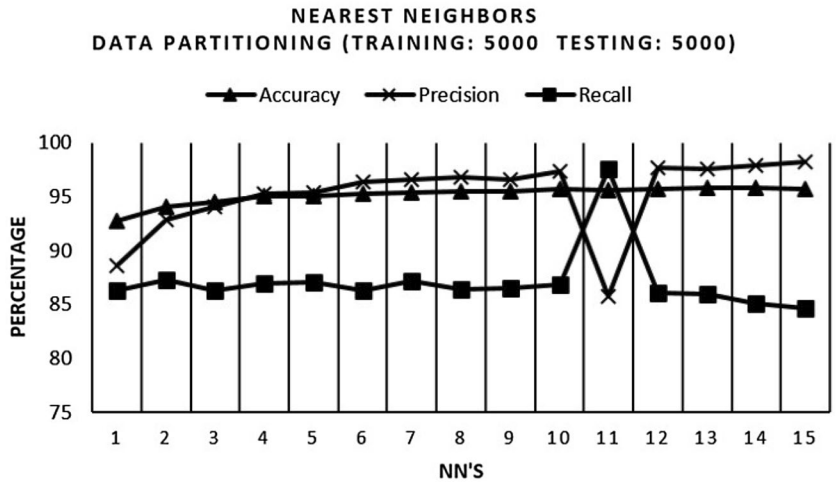

Fig. 11 Optimal NN's for RC-CNN: data distribution 50:50

theoretical complexity, average problem solving time and performance has been compared with the existing benchmarks and benefits have been clearly highlighted.

Selection of the optimal k in all variants of the case-base maintenance activity is done through a brute-force approach. It searches all possible candidates exhaustively and selects the one with the highest accuracy. The actual value of the optimal points for a different problem domain or data may be different and will need to execute this step independently.

Benchmarking with state-of-the-art approaches

In this section, a comparison of proposed approach with existing standard CBM techniques has been presented in terms of theoretical complexity, average problem solving time and performance.

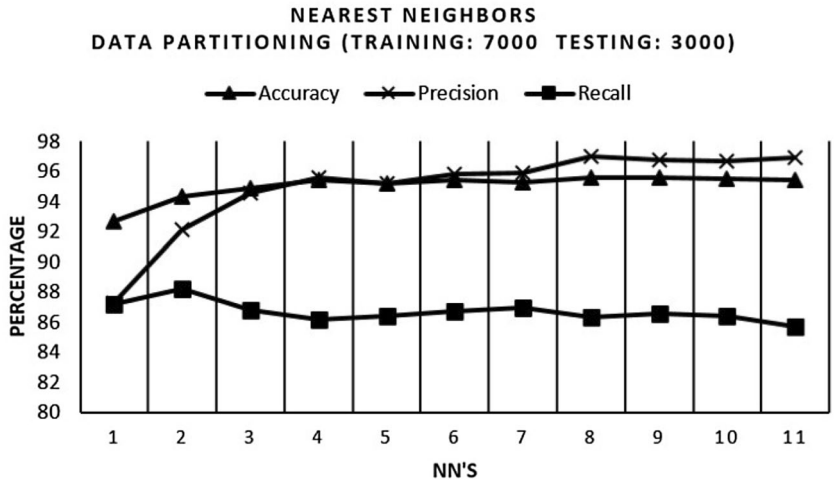

Fig. 12 Optimal NN's for RC-CNN: data distribution 70:30

Table 4 Optimal NN's for RC-CNN

Training	Testing	Optimal *NN*	Accuracy (%)	Precision (%)	Recall (%)
3000	7000	8	95.2	97.3	85
5000	5000	13	95.8	97.6	86
7000	3000	8	95.6	97	86.3

Experiment 1: Comparison of theoretical complexities

A comparison of the theoretical complexities of three different CBM approaches in terms of big *O* has been analyzed. Table 7 shows that the theoretical complexity of the proposed algorithm is relatively higher than the benchmarks. This is because our proposed algorithm contains online as well as offline maintenance. Since offline maintenance is not triggered unless the case-base cardinality has reached the threshold, therefore the complexity of offline algorithm is in acceptable range. The complexity of online algorithm is high since it is executed after each problem solving cycle.

Experiment 2: Comparison of physical time

In this experiment, average problem solving time of typical CBR cycle has been compared with each maintenance technique. The proposed hybrid algorithm is adding only the most competent cases, therefore, the case-base cardinality is not growing swiftly and as a result the retrieval time will reduce which will eventually minimize the time required to solve a new problem. The results obtained have been shown in Figs. 19, 20 and 21 which reflect that the proposed approach consumes lesser amount of time to provide the solution as compared to other techniques, while keeping the execution environment constant.

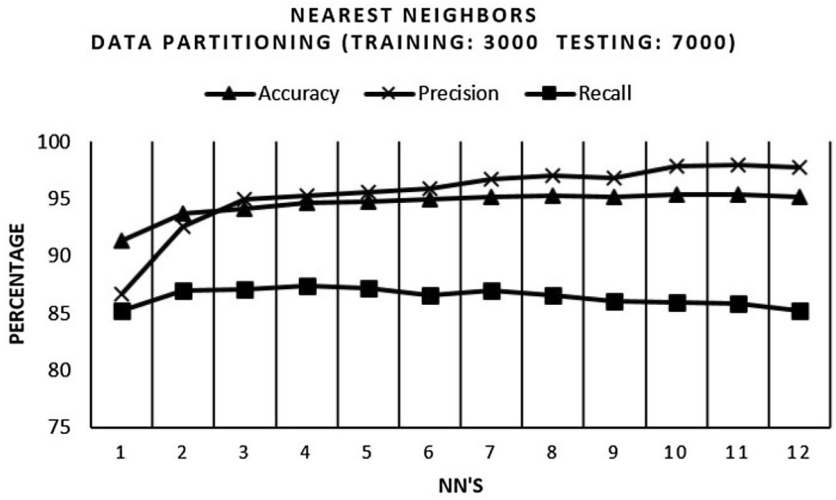

Fig. 13 Optimal NN's for footprint deletion: data distribution 30:70

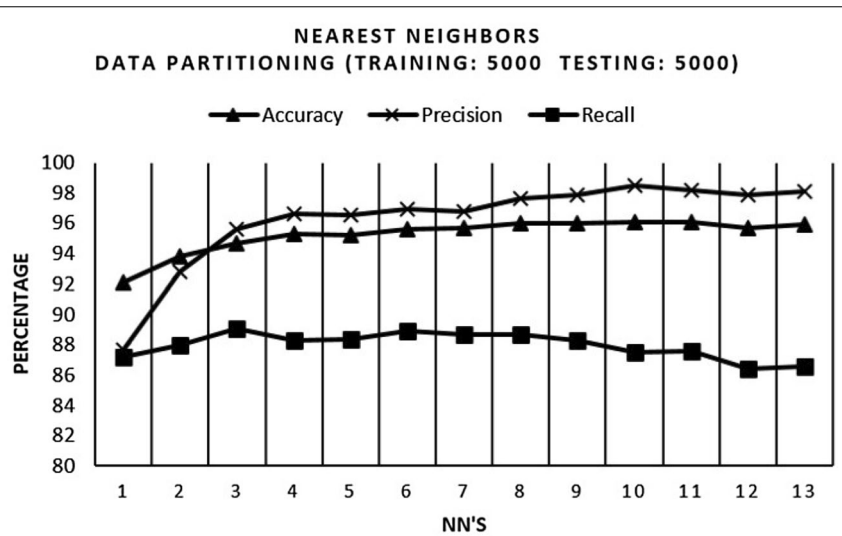

Fig. 14 Optimal NN's for footprint deletion: data distribution 50:50

Experiment 3: Empirical comparison in terms of performance

Accuracy, recall and precision metrics computed for each CBM technique have been compared with classical CBR cycle. The results obtained from each approach are comparable. Each approach provides almost the same accuracy as a CBR cycle. However, each technique reduces the size of a case-base in a way that it will consist of only the most competent cases. Each technique yields optimal accuracy. These results have been shown in Figs. 22, 23 and 24.

Accuracy captures the correct classification rate of the applied algorithm. Precision captures true positive rate out of the total positive predictions. Recall captures the true positive rate out of the total positive actuals. The hybrid approach has clearly shown high performance on the simulated case study across all these three

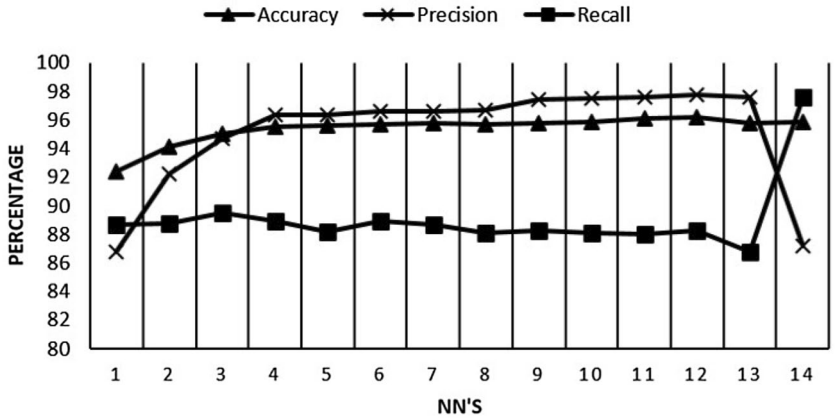

Fig. 15 Optimal NN's for footprint deletion: data distribution 70:30

Table 5 Optimal NN's for footprint deletion

Training	Testing	Optimal NN's	Accuracy (%)	Precision (%)	Recall (%)
3000	7000	10	95.4	97.9	85.9
5000	5000	10	96.1	98.5	87.5
7000	3000	12	96.2	97.8	88.3

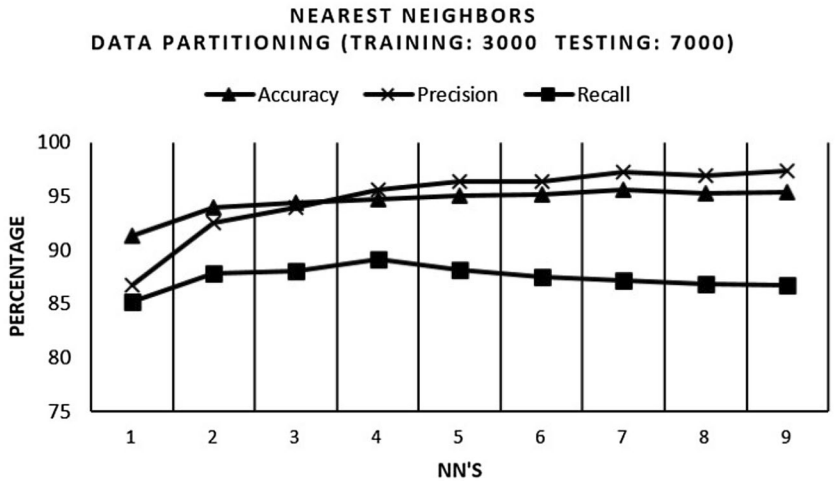

Fig. 16 Optimal NN's for proposed hybrid approach: data distribution 30:70

performance measures. It has shown up to 96% accuracy, 97% precision and 88% recall performance in different data splits. At the same time, it has resulted into the most optimal average problem solving time as compared with the benchmarks. Putting these two performance angles together, it shows that the raise in theoretical

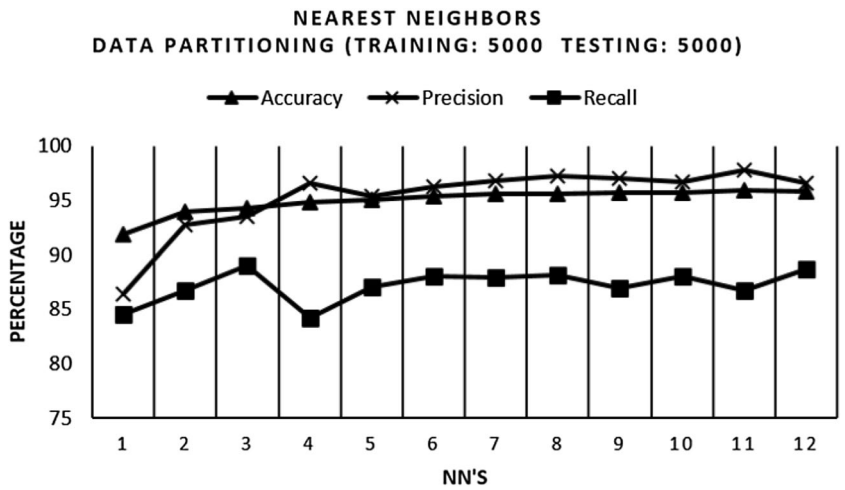

Fig. 17 Optimal NN's for proposed hybrid approach: data distribution 50:50

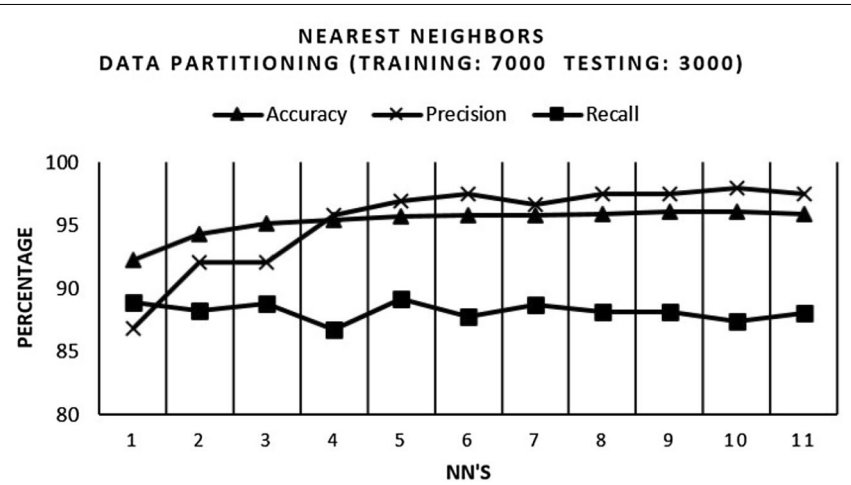

Fig. 18 Optimal NN's for proposed hybrid approach: data distribution 70:30

Table 6 Optimal NN's for proposed hybrid approach

Training	Testing	Optimal NN's	Accuracy (%)	Precision (%)	Recall (%)
3000	7000	7	95.6	97.2	87.2
5000	5000	11	95.8	97.8	86.7
7000	3000	9	96	97.5	88.1

complexity of the proposed algorithm is insignificant because it controls the size of the compact case-base and average problem solving time reduces significantly.

From above experiments, it is clear that the hybrid approach provides optimal performance while improving the efficiency of the CBR system. When the case-base cardinality was set to 3000, the proposed maintenance architecture was able to reduce

Table 7 Comparison of theoretical complexities

Technique	Activating time	Online complexity	Offline complexity
CBR cycle	Continuous	$O(n)$	N/A
RC-CNN	Conditional	N/A	$O(n^2)$
FPD	Conditional	N/A	$O(n^2)$
Hybrid	Continuous + conditional	$O(n^2)$	$O(n^2)$

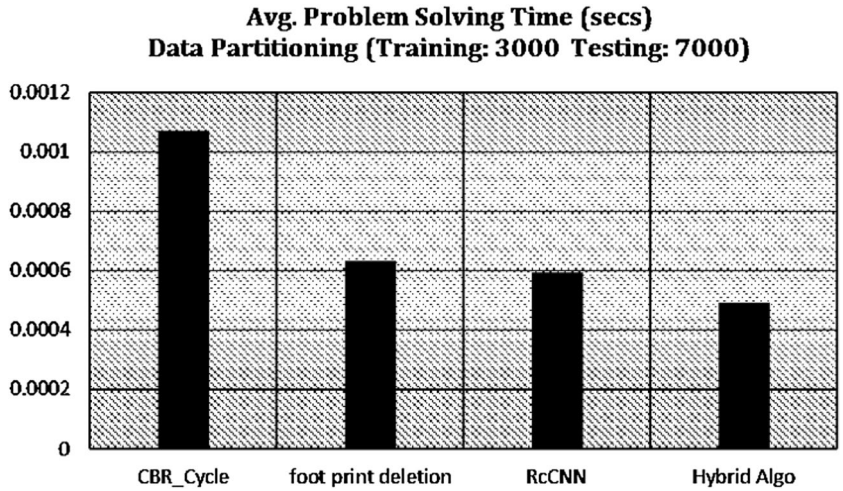

Fig. 19 Average problem solving time: data distribution 30:70

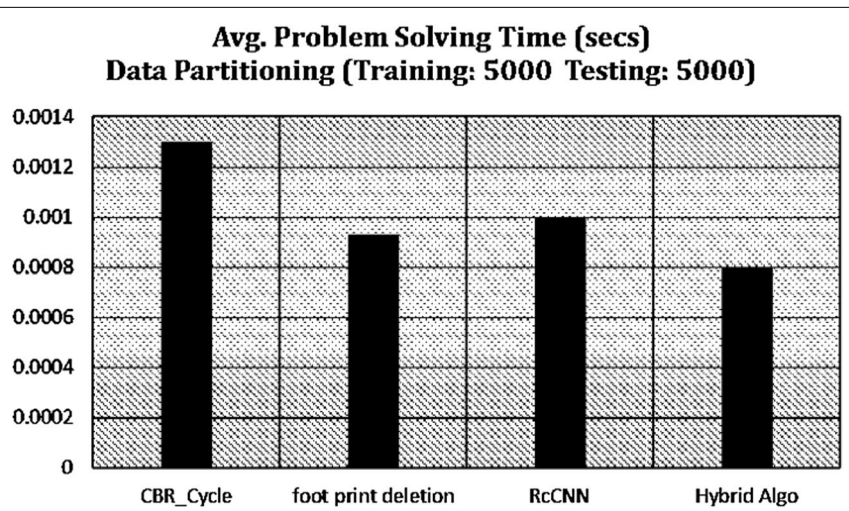

Fig. 20 Average problem solving time: data distribution 50:50

average problem solving time to almost 65% whereas footprint deletion was able to reduce it by 55% and RC-CNN reduced it to 57%. Somewhat of a similar trend was noticed when the case-base cardinality was 5000 and 7000.

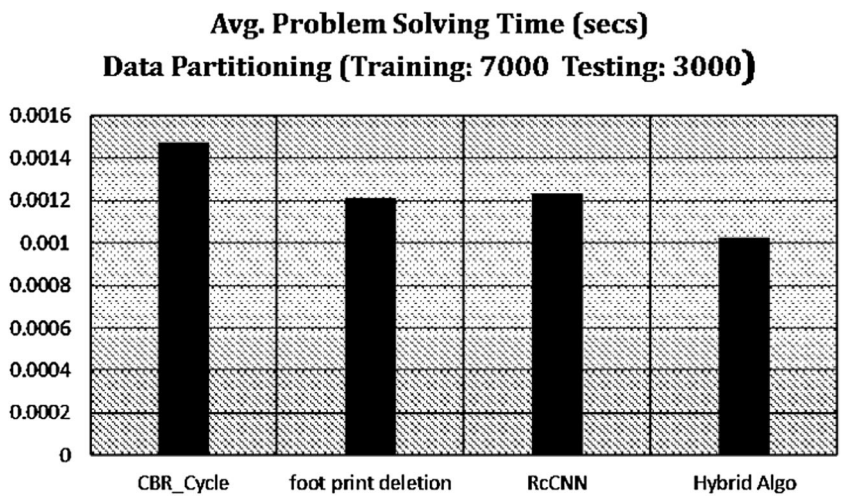

Fig. 21 Average problem solving time: data distribution 70:30

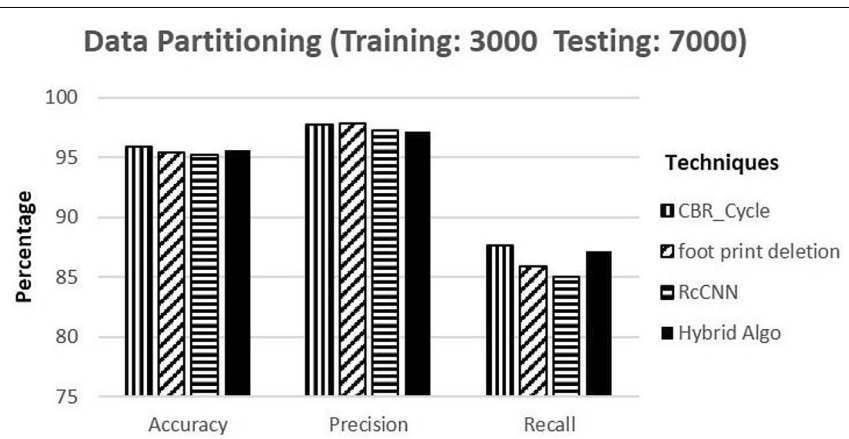

Fig. 22 ARP comparison: data distribution 30:70

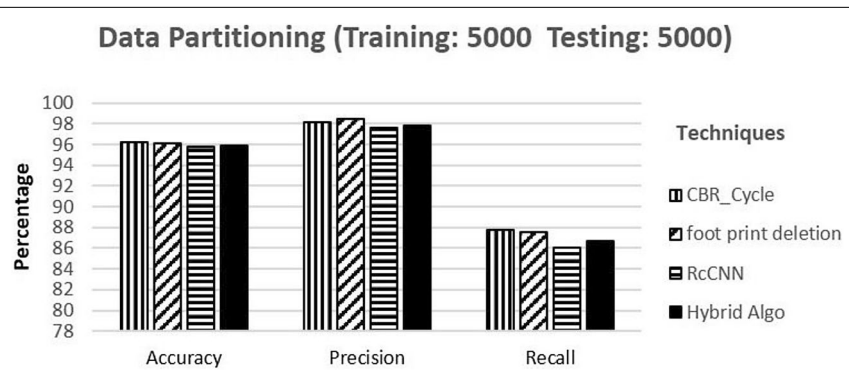

Fig. 23 ARP comparison: data distribution 50:50

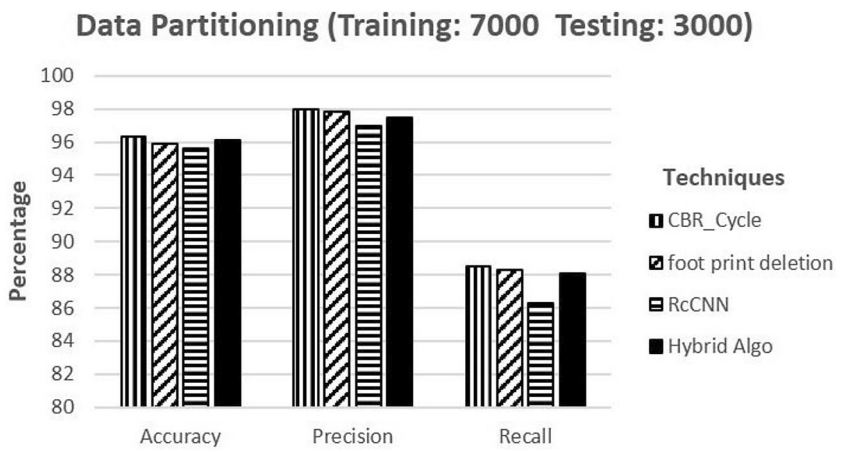

Fig. 24 ARP comparison: data distribution 70:30

Conclusion and future directions

To encounter the retrieval efficiency of existing case-base maintenance techniques, a hybrid CBM approach has been proposed in this work which uses a case addition algorithm on continuous basis in online mode and a deletion algorithm which operates on conditional basis in offline mode. The results obtained suggest that the average problem solving time is much lesser than other standard approaches and the overall competence and the quality of solution is not affected. The computational complexity of the proposed approach for online maintenance is relatively high. However, the bottleneck does not hinder the performance of proposed approach because of its conditional offline and non-frequent nature.

In future, the computational complexity of the online maintenance will be addressed. This could be achieved by exploring different data representation mechanisms including fuzzy representation. Moreover, if we group different parts of data in case-base by deploying appropriate clustering algorithm as an offline option, then the performance and retrieval efficiency are expected to improve further. Another possible future direction is figure out compact and competent case-bases in distributed environments. Synchronisation of distributed components of the case-base and their online maintenance may be addressed in future. Multi-objective criteria may be designed in future to shrink the larger case-bases into set of appropriate representatives like centroids of well-defined clusters. Appropriate representatives may be augmented carefully through various data augmentation techniques so that the quality of case-base is improved and its overall problem solving capability in the respective domain is enhanced.

Authors' contributions

MJ identified the research problem, designed the research study and methodology, formulated proposed algorithms and contributed significantly in writing the manuscript. HH contributed in implementation of algorithms, analysis of results and writing the manuscript. IA contributed in formulating the algorithms, choosing the benchmarks and writing the manuscript. All authors read and approved the final manuscript.

Author details

[1] Department of Computer Science, Namal College, Mianwali, Pakistan. [2] School of Electrical Engineering and Computer Science, University of Bradford, Bradford, UK.

Acknowlegements
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Availability of data and materials
The datasets used and analysed during the current study are available from the corresponding author on reasonable request.

Funding
Authors acknowledge the internal funding support received from Namal College Mianwali to complete the research work.

Publisher's Note

Received: 5 November 2018 Accepted: 25 February 2019
Published online: 13 March 2019

References

1. Aamodt A, Plaza E (1994) Case-based reasoning: foundational issues, methodological variations, and system approaches. In: AI Communications, vol 7:1. IOS Press, New York, pp 39–59
2. Abdel-Aziz A, Cheng W, Strickert M, Hüllermeier E (2013) Preference-based CBR: a search-based problem solving framework. In: International conference on case-based reasoning. Springer, Berlin, pp 1–14
3. Abdel-Aziz A, Hüllermeier E (2015) Case base maintenance in preference-based cbr. In: International conference on case-based reasoning. Springer, Berlin, pp 1–14
4. Aydadenta H, Adiwijaya (2018) A clustering approach for feature selection in microarray data classification using random forest. J Inform Proces Syst 14(5):1167–1175
5. Begum S, Ahmed M U, Funk P, Xiong N, Folke M (2011) Case-based reasoning systems in the health sciences: a survey of recent trends and developments. IEEE Trans Syst Man Cybern C 41(4):421–434
6. Büyüközkan G, Ergün B (2011) Intelligent system applications in electronic tourism. Expert Syst Appl 38(6):6586–6598
7. Costa DM, Teixeira EN, Werner CML (2018) Odyssey-processcase: a case-based software process line approach. In: Proceedings of Brazilian symposium on software quality, pp 170–9
8. Cummins L, Bridge D (2011) On dataset complexity for case base maintenance. In: Case-based reasoning research and development, pp 47–61
9. Cunningham P (1998) CBR: Strengths and weaknesses. In: Proceedings of 11th international conference on industrial and engineering applications of artificial intelligence and expert systems. Springer, Berlin, pp 517–23
10. Cunningham P (2009) A taxonomy of similarity mechanisms for case-based reasoning. IEEE Trans Knowl Data Eng 21(11):1532–1543
11. Fan Z-P, Li Y-H, Wang X, YangLiu (2014) Hybrid similarity measure for case retrieval in cbr and its application to emergency response towards gas explosion. Expert Syst Appl 41(5):2526–2534
12. Feuilltre H, Auffret V, Castro M, Breton HL, Garreau M, Haigron P (2017) Study of similarity measures for case-based reasoning in transcatheter aortic valve implantation. In: Proceedings of international conference on computing in cardiology
13. Hao F, Sim D-S, Park D-S, Seo H-S (2017) Similarity evaluation between graphs: a formal concept analysis approach. J Inform Proces Syst 13(5):1158–1167
14. Haouchine M-K, Chebel-Morello B, Zerhouni N (2008) Competence-preserving case-deletion strategy for case-base maintenance. In: ECCBR'08, vol 1, pp 171–184
15. Hart P (1968) The condensed nearest neighbor rule (corresp.). IEEE Trans Inform Theory 14(3):515–516
16. Ihrig LH, Kambhampati S (1996) Design and implementation of a replay framework based on a partial order planner. In: AAAI/IAAI, vol. 1, Citeseer, pp 849–854
17. Juarez JM, Craw S, Lopez-Delgado JR, Campos M (2018) Maintenance of case bases: current algorithms after fifty years. In: Proceedings of international joint conference on artificial intelligence, pp 5457–5468
18. Khan MJ (2014) Applications of case-based reasoning in software engineering: a systematic mapping study. IET Softw 8(6):258–268
19. Khan MJ, Awais MM, Shamail S, Awan I (2011) An empirical study of modeling self-management capabilities in autonomic systems using case-based reasoning. Simul Model Practice Theory 19(10):2256–2275
20. Khoshgoftaar TM, Seliya N, Sundaresh N (2006) An empirical study of predicting software faults with case-based reasoning. Softw Qual J 14(2):85–111
21. Leake D, Wilson D (1998) Categorizing case-base maintenance: dimensions and directions. In: Advances in case-based reasoning, pp 196–207
22. Leake DB (1996) CBR in context: the present and future. In: Case-based reasoning, experiences, lessons & future directions
23. Leake DB, Wilson DC (2000) Remembering why to remember: performance-guided case-base maintenance. In: European workshop on advances in case-based reasoning, Springer, Berlin, pp 161–172
24. Lenz M, Bartsch-Spörl B, Burkhard H-D, Wess S (2003) Case-based reasoning technology: from foundations to applications, vol 1400. Springer, Berlin

25. Liao S-H, Chu P-H, Hsiao P-Y (2012) Data mining techniques and applications-a decade review from 2000 to 2011. Expert Syst Appl 39(12):11303–11311
26. Liao TW, Zhang Z, Mount CR (1998) Similarity measures for retrieval in case-based reasoning systems. Appl Artif Intell 12(4):267–288
27. Lupiani E, Craw S, Massie S, Juarez JM, Palma JT (2013) A multi-objective evolutionary algorithm fitness function for case-base maintenance. In: International conference on case-based reasoning, Springer, Berlin, pp 218–232
28. Lupiani E, Juarez JM, Palma J (2014) Evaluating case-base maintenance algorithms. Knowl Based Syst 67:180–194
29. Lupiani E, Massie S, Craw S, Juarez JM, Palma J (2016) Case-base maintenance with multi-objective evolutionary algorithms. J Intell Inform Syst 46(2):259–284
30. Mair C, Kadoda G, Lefley M, Phalp K, Schofield C, Shepperd M, Webster S (2000) An investigation of machine learning based prediction systems. J Syst Softw 53(1):23–29
31. Mantaras D, Lopez R, David M, Derek B, Leake D, Smyth B, Craw S, Faltings B, Maher ML, Cox MT, Forbus K et al (2005) Retrieval, reuse, revision and retention in case-based reasoning. Knowl Eng Rev 20(03):215–240
32. Markovitch S, Scott PD (1988) The role of forgetting in learning. In: Machine learning: proceedings of the fifth intl. conf, pp 459–465
33. Minton S (1990) Quantitative results concerning the utility of explanation-based learning. Artif Intell 42(2–3):363–391
34. Montani S, Jain LC et al (2010) Successful case-based reasoning applications, vol 305. Springer, Berlin
35. Muñoz-Avila H (1999) A case retention policy based on detrimental retrieval. In: International conference on case-based reasoning. Springer, Berlin, pp 276–287
36. Myllymäki P, Tirri H (1994) Massively parallel case-based reasoning with probabilistic similarity metrics. In: Topics in case-based reasoning. pp 144–154
37. Perner P (2014) Mining sparse and big data by case-based reasoning. Procedia Computer Sci 35:19–33
38. Racine K, Yang Q (1996) On the consistency management of large case bases: the case for validation. In: To appear in AAAI technical report-verification and validation workshop
39. Segovia J, Szczepaniak PS, Niedzwiedzinski M (2013) E-commerce and intelligent methods, vol 105. Physica
40. Shepperd M (2003) Case-based reasoning and software engineering. In: Managing software engineering knowledge. Springer, Berlin, pp 181–198
41. Smiti A, Elouedi Z (2011) Overview of maintenance for case based reasoning systems. Int J Comput Appl 32(2):49–56
42. Smiti A, Elouedi Z (2014) Wcoid-dg: an approach for case base maintenance based on weighting, clustering, outliers, internal detection and dbscan-gmeans. J Comput Syst Sci 80(1):27–38
43. Smyt B, McKenna E (1999) Footprint-based retrieval. In: International conference on case-based reasoning. Springer, Berlin, pp 343–357
44. Smyth B, Cunningham P (1996) The utility problem analysed. In: European workshop on advances in case-based reasoning, Springer, Berlin, pp 392–399
45. Smyth B, Keane MT (1995) Remembering to forget. In: Proceedings of the 14th international joint conference on artificial intelligence. Citeseer, pp 377–382
46. Smyth B, McKenna E (1999) Building compact competent case-bases. In: ICCBR
47. Sriwanna K, Boongoen T, Iam-On N (2017) Graph clustering-based discretization of splitting and merging methods (graphs and graphm). Hum Centric Comput Inform Sci 21(7):1–39
48. Sun J, Zhu Z, Zhang Y, Zhao Y, Zhai Y (2018) Research on personalized recommendation case base and data source based on case-based reasoning. In: Proceedings of international conference on cloud computing and security. pp 114–123
49. Ullah K, Mahmood T, Jan N (2018) Similarity measures for t-spherical fuzzy sets with applications in pattern recognition. Symmetry 10(6):1–14
50. Watson I, Marir F (1994) Case-based reasoning: a review. Knowl Eng Rev 9(4):327–354
51. Zhang Y, Zhang S, Leake D (2016) Case-base maintenance: a streaming approach. In: Proceedings of international conference on case-based reasoning. Springer, Berlin, pp 222–231
52. Zhu J, Yang Q (1999) Remembering to add: competence-preserving case-addition policies for case-base maintenance. In: IJCAI, vol 99. pp 234–241