Master of Aerospace Engineering
Research Project

Knowledge Reuse for Design of a Complex System

S3 Project report

Author(s): Wendi Ding

Due date of report: 25/03/2022
Actual submission date: 25/03/2022

Starting date of project: 29/Jan/2021 Duration: 14 Months

Tutors: Rob Vingerhoeds, Juan José Montero Jimenez, Sebastien Schwartz

Table of Contents

Declaration of Authenticity

This assignment is entirely my own work. Quotations from literature are properly indicated with appropriated references in the text. All literature used in this piece of work is indicated in the bibliography placed at the end. I confirm that no sources have been used other than those stated.

I understand that plagiarism (copy without mentioning the reference) is a serious examinations offence that may result in disciplinary action being taken.

Date 17/03/2022 Signature 丁文迪

Abstract

The project is dedicated to a Predictive Maintenance Case-Based Reasoning (CBR) system built in ISAE-SUPAERO and is aimed at developing a decision support system based on CBR for complex system design. The goal is to improve diversity between retrievals and to optimize the case base performance through exploring Case Base Maintenance (CBM) methods. Existing approaches are studied and applied to the predictive maintenance case base, and a Modified Condensed Nearest Neighbor (CNN) method which integrates separation, generalization and reindexing of descriptions and solutions is proposed and proved to be effective in diversifying the retrievals while keeping the advantages of competence-preserving CBM methods. The Case Base Visualization (CBV) technique of Force-Directed Graph-Drawing is applied to demonstrate case disctribution and retrieval results to the user.

Keywords: Case-Based Reasoning, Case Base Maintenance, Similarity & Diversity, Generalization, Separation & Reindexing, Case Base Visualization

1 Introduction

Case Based Reasoning (CBR) is a branch of symbolic artificial intelligence techniques that focuses on solving a new problem by reusing information and knowledge from a previous similar situation. In the design of complex systems, it is a common performance to reuse designs from the past through recalling previous design experiences and knowledge about existing designs. This reuse of experience allows to reduce time and effort compared with designing a completely new system from scratch and to have a better confidence in the design. In reality, the reuse of design experiences requires high expertise from human beings and cannot easily be shared from one to another. On the contrary, computers can handle information faster than humans and the knowledge of more than one expert can be merged and implemented to the CBR system. Therefore, CBR is a promising paradigm for developing support systems for complex system.

A general CBR cycle can be described by four processes, as depicted in Fig. 1. A new problem is solved by retrieving previously experienced cases that address similar problems from the case base, reusing the solutions developed from those cases, revising the returned solutions based on the reuse and the evaluation of obtained results, and retaining the new knowledge by incorporating the new learned case into the case base [1].

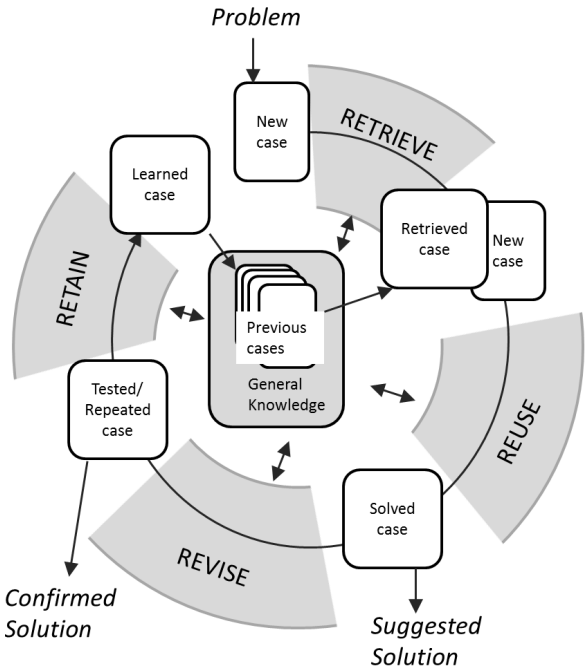


Figure 1: A general CBR cycle [1]

Since the 1960's, as a branch of artificial intelligence, CBR has been raising increasing attention of the scientists. In 2002, R. Bergmann [2] unified terminologies on CBR and donated mathematical definitions to different terminologies. R. Bergmann and J. Kolodner et al. [3] summarized different categories of case representation in CBR. Several CBR applications have come into being since the 1990s. For example, HOMER has been developed for supporting the CAD/CAM help-desk at DaimlerChrysler in Sindelfingen [4]. The READee prototype shows the suitability of knowledge- and similarity-based retrieval methods for catalogs of reusable design components, i.e. digital signal processors (DSPs) [5]. The Protos program was applied to the task in clinical audiology of identifying a patient's hearing disorder from symptoms, test results and history [6]. In the aspect of aerospace,

the DRAMA framework has been developed based on an interactive CBR system with CMaps dealing with the complexity of aerospace design [7].

However, problems such as the utility problem [8] and lack of diversity between retrievals arise with the expansion of the case base, causing long computation time for the retrieval process and redundant retrieval results, which are undesirable for users. Approaches have been explored in order to figure out these problems. To improve the diversity between retrievals, K. Bradley and B. Smyth [9-10] defined the diversity measure of a set of cases as the average dissimilarity between all pairs of cases in the retrieval result set. They proved that the diversity-preserving Bounded Greedy Selection (BGS) algorithm based on the Quality Metrics is an effective strategy to optimise the similarity-diversity trade-off of retrievals.

The diversity-preserving algorithm has been proved to be effective for improving the diversity of retrievals. However, most of the existing diversification methods are integrated into the retrieval process, which could make the retrieval process very heavy and time-consuming, yet the utility problem still exists, i.e. the retrieval time increases linearly with the case base size and the case base is not scalable. Therefore, it is not a good manner to put heavy workload onto the retrieval process, instead, Case Base Maintenance (CBM) methods should be explored to improve the diversity of retrievals.

From a memory standpoint, learning in CBR consists in the creation, update, and organization of the structures and organization in memory. It is often referred to as Case Base Maintenance (CBM) [11]. Several approaches of CBM [12-14] have been proposed and evaluated by previous researchers. M.K. Haouchine et al. have identified 3 performance goals of CBM: Efficiency, Competence and Solution Quality [12], and they proposed a competence-preserving case deletion CBM strategy called CM-CNN (Competence Metrics), which is a derivation of the Condensed Nearest Neighbor (CNN) algorithm [15], to guide the construction of smaller case bases. Other attempts include the case addition RC-CNN (Relative Competence) algorithm [11] proposed by B. Smyth et al., and the case deletion strategy Footprint Deletion (FD) [16] proposed by B. Smyth and M. T. Keane. Apart from competence-preserving CBM strategies, strategies that focus on utility problem and case base adaptation performance are also studied. D.B. Leake and D.C. Wilson [17] proposed the method of RP-CNN (Relative Performance) which combines the performance metrics with competence metrics to guide the construction of the case base, and proved that it is possible to improve the case base performance without impairing the competence of the case base.

Another interesting domain in CBM is the generalization and abstraction of cases to better organize the memory structure. K. Maximini et al. [18] proposed methodologies of similarity measurement for generalized cases, depending on the property of the cases as well as the semantics of sets. R. Schmidt and L. Gierl [19] applied the *Hash-Tree-Retrieval-Algorithm* and proposed to generalize from single cases into prototypical cases and to erase redundant cases so that an indefinite growth of the case base would be avoided. A. Tartakovski et al. [20] also applied generalized cases to the selection of life insurance policies and transformed the retrieval process into an optimization problem, dealing with numerical features that depend on each other.

Different memory structures of the case base are also explored. I. Bichindaritz [21] has made a summary of approaches to case base memory structure organization referring to the theory of the dynamic memory, and categorized learning related to CBR memories into two main types, i.e. mining for memory structures and mining for memory organization. J.H. Gennari et al. [22] propose some incremental conceptual clustering methods for the organization of memory structure from Machine

Learning Community concerning classification problems such as COBWEB [23] that makes use of a heuristic evaluation measure category utility to maximize intra-class similarity and inter-class dissimilarity, while incrementally incorporating instances into a categorization tree. Since the process of incremental conceptual clustering is quite similar to the learning of new cases in CBR and that the COBWEB algorithm aims at maximizing intra-class similarity and inter-class dissimilarity, we regard it as a prospective method for the organization of our case base memory structure.

However, concerning case base memory structure, only few papers discuss the separation of description space and solution space for the case base, while currently existing case base maintenance approaches mostly focus on reducing case base size while preserving case base competence, i.e. to be able to solve the same amount of problems as the original case base with a smaller case base. It is true that competence-preserving case base maintenance methods which remove redundant cases from the case base can reduce retrieval time by maintaining a smaller case base while still providing satisfying retrieval results. Nevertheless, with competence-preserving methods, cases are removed from the case base just according to competence metrics, while the other aspects of case base performance are ignored. One case can be considered as redundant and can be removed from the case base because it is covered by another case with better competence. In this way, we may risk removing cases with diverse and valuable solutions from the maintained ones. Therefore, we consider it necessary to separate the description space and the solution space when we explore case base maintenance methods aiming at diversifying CBR retrieval results.

The aim of this project is to develop a decision support system based on Case Based Reasoning (CBR) for complex system design. The idea is to formalise, to retrieve and to propose a feasible solution for the current situation. Our goal is to improve diversity between retrievals and to optimize the case base performance through exploring CBM methods. The project is dedicated to a Predictive Maintenance CBR system built in ISAE-SUPAERO. Approaches of diversification during the retrieval process and CBM methods are studied and applied to our predictive maintenance case base with 230 cases extracted from reference papers. In the end, a modification on the competence-preserving CNN case addition method which integrates generalization and reindexing of descriptions and solutions is made in order to better adapt the algorithm to our goal of diversification.

In this report, Section 2 describes the context and key issues, work done during Semester 2, and work to be done during Semester 3. Section 3 discusses the investigation methods, including investigation of competence-preserving CBM methods, proposition and elaboration of our modified CNN method, and means of Case Base Visualization (CBV). Section 4 discusses the results and analysis, and Section 5 discusses the conclusion and perspectives.

2 Semester 3 section

2.1 Context and key issues

Predictive Maintenance (PM) focuses on the organization of maintenance actions according to the actual health state of the system, aiming at giving a precise indication of when a maintenance intervention will be necessary [24]. On the way of applying CBR method to predictive maintenance, J.J. Montero Jiminez et al. in ISAE-SUPAERO have built a case base with 230 entries of situation description and solution concerning predictive maintenance diagnostics and prognostics. A.T. Miyagawa [25] and J. Mazouzi [26] have worked on the implementation of the predictive maintenance CBR program based on the software myCBR with the Software Development Kit (SDK). Feature-vector representation has been chosen to represent the cases with 19 attributes (see

Appendix 1). Similarity measurements and aggregation functions have been built for the CBR program, and a GUI interface has been developed by senior students.

The issue is that with the growth of case base size, the quality of retrievals decreases in that the cases with redundant solutions are retrieved. For example, the top five cases retrieved for a query may all propose the same model of neural network. The user may expect five alternative solutions for the query by inputting the desired number of retrievals, but the five retrievals actually offer no alternative solutions, which is undesired by the user. Faced with the redundancy of solutions suggested by the retrieved cases, our goal is to improve diversity between retrievals and to optimize the case base performance through exploring approaches of diversification during the retrieval process and CBM methods.

2.2 Work done during Semester 2

In Semester 2, we consulted papers to obtain a general knowledge of CBR which concerns case representation, retrieve, revise, reuse and retain processes of a CBR cycle [1, 2]. Local similarities for different types of attributes (numerical, symbol, string, etc.) of a case and global aggregation functions are studied (see Appendix 2) [3].

Based on the open-source similarity-based retrieval tool myCBR with the Software Development Kit (SDK), we tested the CBR program with similarity measurements and aggregation functions developed by senior students. In order to improve the diversity of retrievals, we consulted the papers and proposed three methods, the first two methods possible to be integrated into the retrieval process and the third one modifies the memory structure: a) Diversification by Elimination [28], b) the Bounded Greedy Selection (BGS) algorithm based on Quality Metrics (QM) [10], and c) Generalized Cases [29].

The three methods are tested using the Leave-One-Out (L-O-O) approach, through which the attributes of each case are considered as a query. With the 230 queries subtracted from the 230 cases in the case base, we have obtained statistical data of average similarity, average diversity and relative benefit of the retrievals, with respect to different requested numbers of retrievals. The results show that the Diversification by Elimination method emphasizes on diversity and sacrifices more similarity metrics. The BGS algorithm with α=0.5 gives the best trade-off between similarity and diversity.

However, both the first two the methods integrate the computation into the retrieval process, which is extremely time-consuming especially when a large number of queries are to be consulted. We find that CBM methods are necessary and promising to better organize the case base and to improve the scalability and performance of the case base, and that a hierarchical memory structure is better for the scalability of the case base. Therefore, we keep on investigating the method of generalized cases in Semester 3 and keep the testing results of diversification methods applied on the retrieval process as a source of comparison.

2.3 Work to be done during Semester 3

A systematic Case Base Maintenance (CBM) approach is required to ensure the diversity of retrievals and to reduce computation time. In Semester 3, we carry on the research on the existing approaches of CBM [12-14] and apply them to our predictive maintenance case base (see Appendix 3). Knowing that the 3 performance goals of CBM are efficiency, competence and solution quality [12], we focus first on competence-preserving CBM strategies to guide the construction of smaller case bases, and select several existing strategies, including case addition and case deletion strategies, which are likely to fulfill our requirement of improving diversity of retrievals. The case addition strategies include the

Condensed Nearest Neighbour (CNN) algorithm [15] and RC-CNN (Relative Competence) algorithm [11], the Random Addition algorithm is also applied as a source of comparison with the other methods. The selected case deletion strategies include the Footprint Deletion (FD) [16] and RC-FD (Relative Competence). In order to implement the competence models to our case base, we define a case to be soluble to a query if the similarity between them exceeds a threshold of 0.9.

Experiments are carried out to test the effectiveness of CBM methods. Based on the representativeness assumption that the case-base is a representative sample of the target problem space [15], we randomly divide the 230 cases in the case base into 184 cases (80%) and 46 queries (20%). Different CBM methods are applied on the generated case base consisting of 184 cases to generate a new case base, and the 46 queries are used to test the competence of the generated new case bases, as well as the similarity and diversity of retrievals obtained from different generated new case bases. The number of retrievals is set to 5. The test is repeated 5 times with random division of the original case base each time, and statistical data of average case base competence, retrieval similarity and diversity are obtained and analysed.

Statistical testing results show that it is possible to reduce the case base size by up to 65.2%, i.e. to retain 64 out of 184 cases, while maintaining the same level of competence. The average similarity is reduced as a compromise of reduced case base size, but it is controlled by the predefined similarity threshold and is thus maintained to an acceptable value. Interestingly, although it is not the explicit goal of competence-preserving CBM methods, when we reduce the case base size from 184 to 64, the diversity between retrievals also rises from 0.5 to 0.7-0.8, which is a desired phenomenon.

To conclude, we can solve the problem of redundant solutions to some extent by applying competence-preserving CBM methods which selectively retains the cases and reduces the density of cases in the case base. However, a low-resolution case base may reduce the solution quality, e.g. reduced similarity between query and cases in the case base, reduced performance and effectiveness of solutions. Therefore, modifications on memory structures can be developed to further improve retrieval diversity while not compromising the other case base performance metrics.

After consulting the existing memory structure of the case base, such as generalized scenes, prototype learning, incremental conceptual clustering and hierarchical memory organization [21], we propose a new case base memory structure featured by the separation of description space and solution space which is obtained through the generalization and reindexing of descriptions and solutions. We implement this method to our predictive maintenance case base and generate a new case base with hierarchical memory organization. In order to examine the effects on diversification of retrievals, or more precisely, the solutions proposed by retrievals, from the new case base, we integrate this algorithm into the testbench that we built for competence-preserving CBM methods. We integrate the separation, generalization and reindexing of cases into the CNN case addition algorithm which generates a new case base with reduced size by adding cases from the old case base, and we call it Modified CNN Algorithm.

Similarly, the test is carried out on the randomly selected 184 cases to generate a new case base with hierarchical memory structure, and the rest 46 cases form 46 queries to test the competence of the new case base, and the similarity and diversity of retrievals. The number of retrievals is set to 5. The test is repeated 5 times and statistical data of average case base competence, retrieval similarity and diversity are obtained and compared with the previously mentioned competence-preserving CBM methods. The details of the generated case base concerning the number of generalized cases and single cases, the respective number of solutions in two layers are also studied. At last, the generated

case bases using different methods are visualized applying the Case Base Visualization (CBV) algorithm of Force-Directed Graph-Drawing to demonstrate the results.

3 Investigation Methods

3.1 Investigation of Competence-Preserving CBM Methods

After consulting the existing CBM methods, we first choose to apply some competence-preserving methods because of their maturity of investigation by previous researchers and because of their potential on improving diversity of retrievals through reducing the case base size and de-densifying the case base. In order to examine the effects of different CBM methods on diversification of retrievals, we define the diversity metrics which is taken from B. Smyth and P. McClave [10] in the effort of diversifying CBR retrieval results through Bounded Greedy Selection (BGS) algorithm (see Appendix 3).

Based on the representativeness assumption that the case-base is a representative sample of the target problem space [15], we randomly divide the 230 cases in the original case base into 184 cases and 46 queries. Experiments are carried out to test the diversification effects of competence-preserving methods, as well as the other case base performance metrics such as competence. The number of retrievals is set to 5. The test is repeated 5 times with random division of the original case base each time, and statistical data of average case base competence, similarity and diversity of retrievals are obtained and analysed.

After an initial analysis of the retrieval results obtained from competence-preserving CBM methods, we propose a new CBM method called Modified CNN method, which integrates CNN algorithm and hierarchical case base memory structure to further improve diversity between retrievals. We integrate this algorithm into our testbench built for competence-preserving CBM methods to test its effectiveness of diversification on retrievals as well as other performance indices. The testbench is built in Java using the library of myCBR(SDK). The testing results concerning the case base coverage and competence, similarity between retrieved descriptions and query, and diversity between retrieved solutions are compared with competence-preserving CBM methods. The flowchart of testing for CBM methods is shown in Fig. 2 and statistical testing results and analysis are shown in Section 4. A brief explanation of the codes for the CBM testbench can be found in Appendix 6.

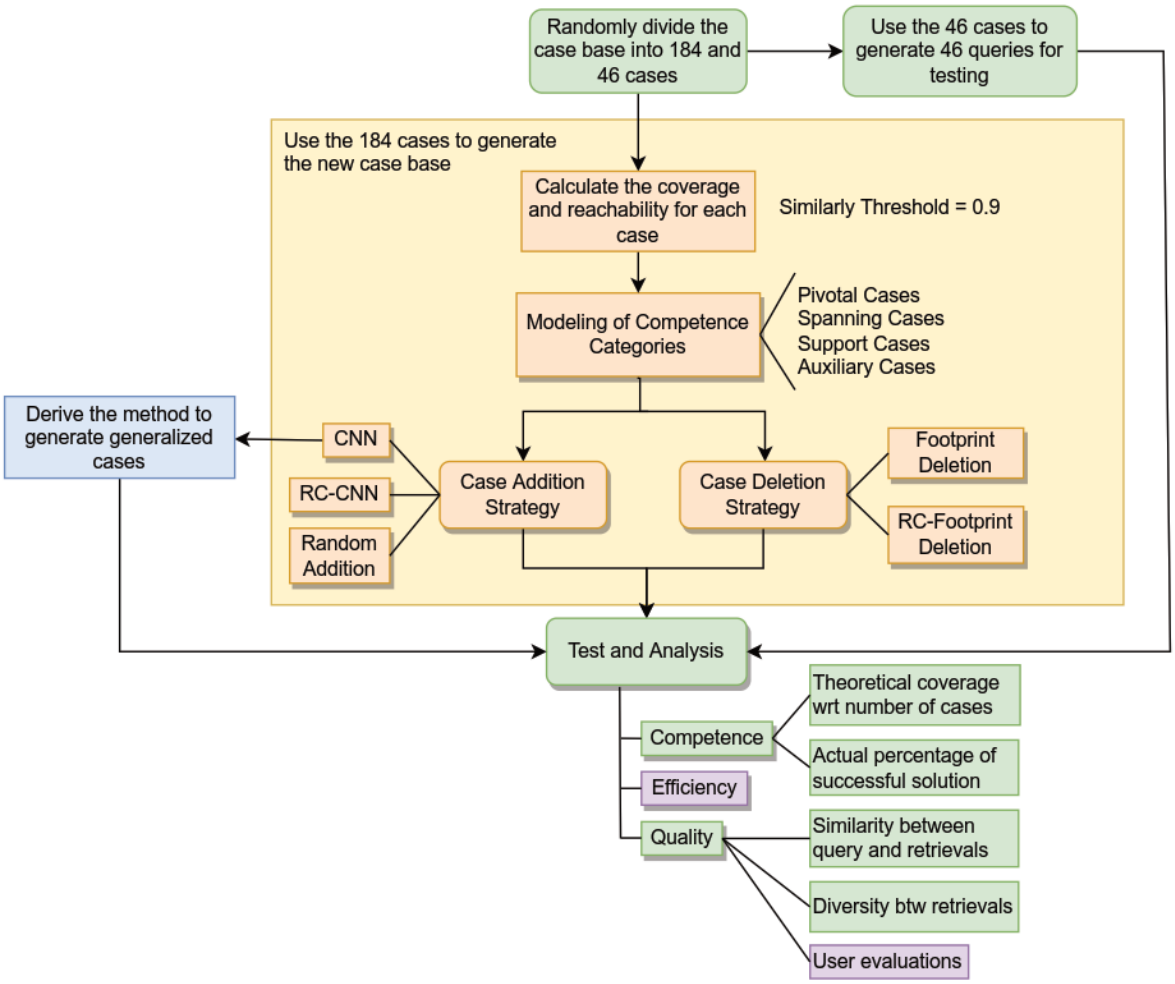


Figure 2: The flowchart of testing of CBM methods

To give an intuitive view of the case distribution in the case base, we applied the technique of Case Base Visualization (CBV) proposed by B. Smyth et al. [30] to visualize the generated case bases. It is a Force-Directed Graph-Drawing Algorithm that maps n-dimensional cases onto a two-dimensional screen while preserving the similarity relationships between pairs of cases as on-screen. Pearson's correlation and Spearman's rank correlation coefficients are used to examine that the relations between cases are correctly restored. We also made a modification to the Force-Directed Graph-Drawing Algorithm in separating the description space and the solution space and connecting the related descriptions and solutions with a spring force. The visualized case bases applying Modified CNN method are shown in Section 4.

3.2 Elaboration of Modified CNN method

To further improve the diversity of retrievals, we propose the idea of Modified CNN method which separates the solution space from the description space. The idea comes from the fact that it is the solution part of the retrieval that the user actually makes use of, and that the diversity of retrievals is actually the diversity of solutions of retrieved cases. Most of the previous CBM methods focus on organizing the case base according to the description part of the cases, and thus they only have an implicit effect on the solution space. The key issue of diversification of solutions is that the previous CBM methods are not directly targeted at the solution space, which leads to the case base performance metrics of competence or efficiency guiding the formation and organization of the case base. Since the competence metrics are directly targeted at the description space, the solutions are just passively organized with their corresponding descriptions. These competence-preserving methods have a

limitation when it comes to the problem of diversification of solutions. Let's consider the following two scenarios (see Fig. 3(a)):

1) Case 1, Case 2 and Case 3 have similar descriptions D1, D2 and D3 with diverse solutions S1, S2 and S3.
2) Case 3 and Case 4 have different descriptions D3 and D4 and similar solutions S1 and S2.

Applying competence-preserving CBM methods, we may remove Case 1 and Case 2 because they have similar descriptions to Case 3 while we preserve Case 3 and Case 4 because their descriptions don't cover one the other. Then during the retrieval process, if Case 3 and Case 4 are the most similar two cases to the query, they are both selected as candidates and thus their solutions S3 and S4 are selected as retrieval results. As a result, the user will find the two retrieval results redundant and very similar to each other, which is not a desired property of the case base. In the meantime, possible alternative solutions which would have been retrieved from the original case base, i.e. S1 and S2, have already been removed from the new case base, and thus we have lost some degree of diversity of solutions.

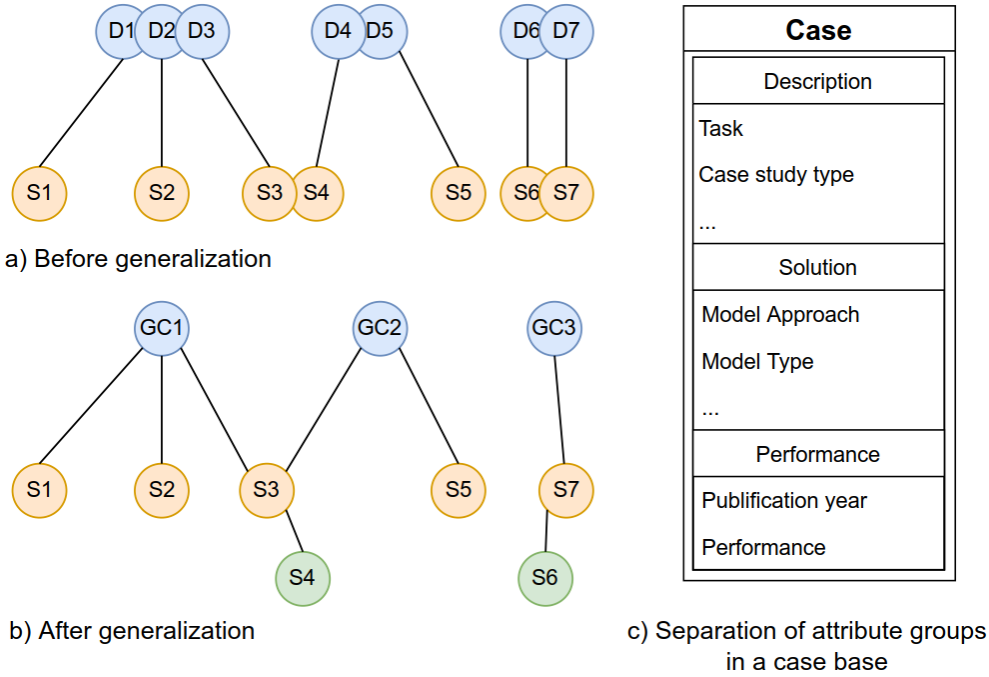


Figure 3: Separation and generalization of solution space and description space

Therefore, it is necessary to separate the description and solution spaces in order to better organize with the solutions in the case base, i.e. to preserve diversity and to reduce redundant solutions. The Modified CNN method takes the advantages of classical CNN method which aims at reducing the case base size while preserving its competence. Three modifications are made on the classical CNN method:

1) Instead of simply removing the redundant cases from the case base, we integrate the similar cases to form a generalized case and store the cases as subcases of the generalized case.
2) Solutions are separated from case descriptions. They are connected to their corresponding descriptions but are not influenced by the manipulation and generalization of descriptions.
3) Manipulation and organization on the solutions are made separately through another generalization process.

In this way, we are not just focusing on preserving case base competence or reducing size of the case base, but also making it possible to diversify the solutions stored in the case base through generalization of redundant solutions.

To apply the Modified CNN method, we first classify the attributes into three groups (see Fig. 3(c)): Description group, Solution group and Performance group. For our predictive maintenance case base, 5 attributes constitute the Description group of a case, i.e. Task, Case study type, Case study, Online/Off-line and Input for the model (see Table 1); 6 attributes constitute the Solution group of a case, i.e. Model Approach, Model Type, Models, Data Pre-processing, Complementary notes and Publication identifier (see Table 2); and 3 attributes constitute the Performance group of a case, i.e. Performance indicator, Performance and Publication year (see Table 3). Then, we separate a case into two parts connected to each other: a Description part and a Solution part, as is shown in Fig. 3(a). The attributes of the Performance group are stored in the Solution part. The description space is constituted with all the Description parts of the cases, while the solution space is constituted with all the Solution parts of the cases. Afterwards, through generalization of descriptions and solutions, a case base memory structure of two layers is applied respectively to the description space and the solution space.

Table 1: Attributes in the description group and their ranges

Attribute	Range
Task	{Feature extraction, Fault detection, Fault identification/isolation, Degradation modelling, Health assessment/degradation analysis, Next state forecasting (one step), Next state forecasting (multiple steps), Remaining useful life calculation}
Case study type	{Rotary machines, Reciprocating machines, Electrical components, Structures, Energy cells and batteries, Production lines, Others}
Case study	String
Online/Off-line	{Off-line, Online}
Input for the model	{Signals, Structure text-based, Text-based maintenance/operation logs, Time series}

Table 2: Attributes in the solution group and their ranges

Attribute	Range
Model Approach	{Multi model, Single model}
Model Type	{Knowledge-based, Data-driven, Physics-based, Multiple-Data-driven}
Models	{LSTM (Long-Short Term Memory Neural Network), FFNN (feed-forward neural network), Kalman Filter, Fuzzy Inference System, ...}
Data Pre-processing	{yes, no}
Complementary notes	String
Publication identifier	String

Table 3: Attributes in the performance group and their ranges

Attribute	Range
Performance indicator	{Error range, Mean accuracy, N/A, Score function, Probability, Reliability, Precision, Convergence, Correlation coefficient, Steady indicator, Standard deviation, Reaction time, Visual indicator, …}
Performance	Double
Publication year	Integer

The generalization process of descriptions and solutions is described in Fig. 4. It consists of the following steps:

1) Generalization of Solutions: Inspired from the CNN case addition method, we randomly add the solutions from original case base into the new case base. We compute the similarity between the solution to be added and the solutions in the case base, and generalize the solution with its nearest neighbour if their similarity is larger than a predefined similarity threshold for solutions. For the generalization of solutions, we choose the solution with higher performance as parent and put it in the second layer, and make the solution with lower performance as its child in the first layer.
2) Reindexing of Descriptions and Solutions: A parent solution is the representative of all its children and the descriptions which are originally connected to children solutions are reconnected to the parent solution. In this way, the children solutions are no longer referred to, instead, the parent solution which represents their group is referred to when the corresponding descriptions are retrieved.
3) Generalization of Descriptions: In the same way, we randomly add the descriptions from original case base into the new case base. We compute the similarity between the current description to be added and the descriptions or Generalized Cases (GCs) in the case base and generalize the description to be added with its nearest neighbour if their similarity is larger than a predefined similarity threshold for descriptions. Descriptions are generalized to form Generalized Cases (GCs) in the second layer, and the original descriptions are stored as subcases of their corresponding GCs in the first layer.
4) Reindexing of GCs and Solutions: All the solutions which are originally connected to the subcases of a GC are reindexed to the GC (see Fig. 3(b)). In this way, we just need to refer to the GC instead of individual descriptions in order to get several diverse solutions belonging to the GC.

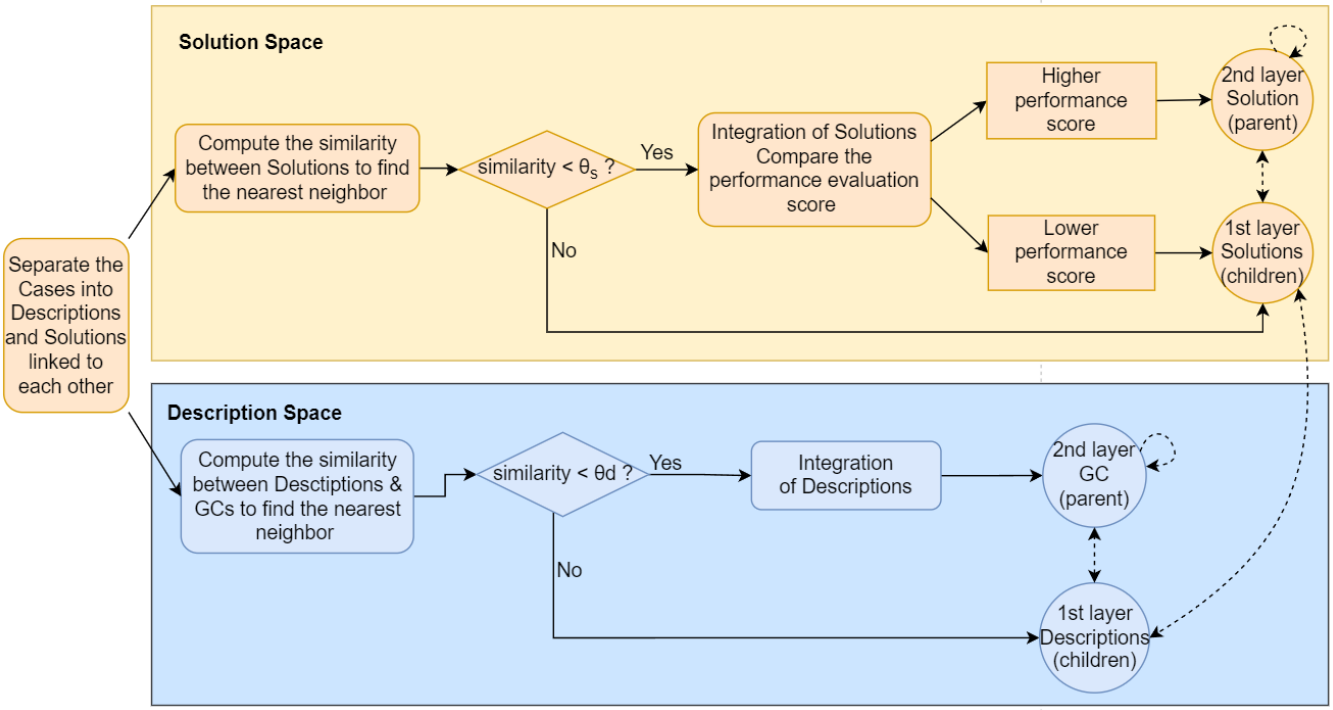


Figure 4: The generalization process of descriptions and solutions

Regarding the performance evaluation of cases, it is only concerned with the Solution part once we have divided the cases into descriptions and solutions, because the performance is to show the effectiveness of the solution, but not the description. Therefore, the attributes of the performance group are allocated to the solution part during the separation of descriptions and solutions.

As is shown in Table 3, since the cases have different performance indicators such as score function, reliability, mean square error, etc, it does not make sense to compare the absolute performance values of the cases with different performance indicators. In order to compare the performance of cases with different performance indicators, we propose to calculate a performance evaluation metrics which is common to all the cases. This performance evaluation integrates the performance values under different performance indicators for each case. Fig. 5 explains how the performance evaluation works. A data base that stores the performance values of different performance indicators that are processed so far by the case base reasoner is added to the CBR system. For each performance indicator, we can compute the normal distribution of its values so that we can predict where the new case to be added ranks among the cases with the same performance indicator in the case base. The performance evaluation of a case is the weighted sum of the ranking percentage of all its performance values under different performance indicators. For example, if a case has a mean accuracy larger than 80% of the other cases whose performance indicator is mean accuracy and an error range smaller than 60% of the other cases whose performance indicator is error range, and both performance indicators have a weight of 1.0, then the performance evaluation of this case is calculated to be 0.7. In this way, we transform the performance values of different performance indicators to a single uniformed performance evaluation metrics, making the solutions with different performance metrics comparable to each other.

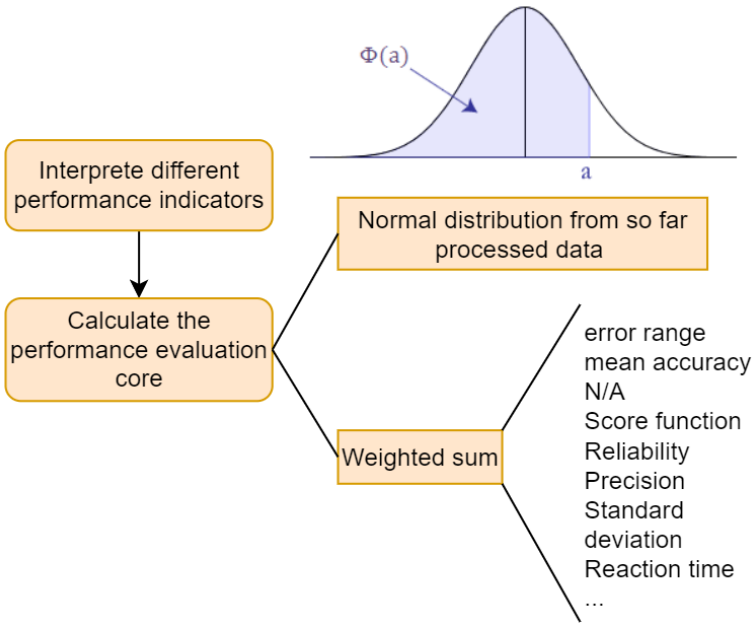


Figure 5: The calculation of performance evaluation metrics

3.3 Case Base Visualization

The Case Base Visualization (CBV) will help the user to (1) perceive case distribution in the case base; (2) to recognise redundant regions with densely clouded similar cases; (3) to recognise regions of poor competence in the case base; (4) to find outliers in the case base [30].

The method that we apply for visualising case-bases is a Force-Directed Graph-Drawing Algorithm. It is aimed to model the similarity relationships between cases as on-screen distances. The key issue is to map the n-dimensional cases onto a two-dimensional screen while preserving the similarity relationships between pairs of cases as on-screen distances.

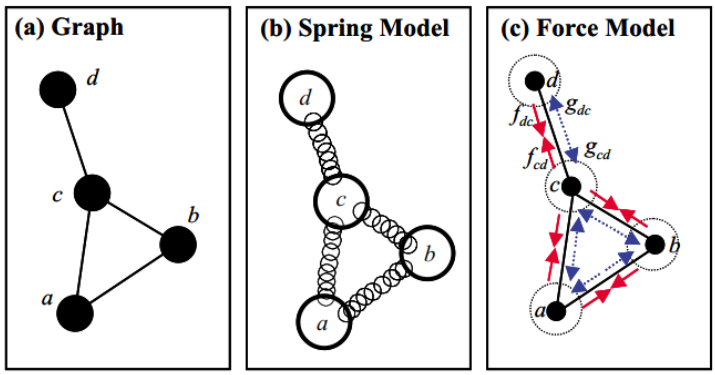


Figure 6: The Force-Directed Graph-Drawing algorithm models a graph as a system of rings and springs under attractive and repulsive forces [30]

The algorithm models a graph as a mechanics system of balls and springs. The balls represent cases, or descriptions and solutions, distributed in the case base, while the springs represent similarity relationships between the cases. The springs exert an attractive force f between connected balls and the balls exert a repulsive force g on each other (see Fig. 6). During graph drawing, the positions of balls are influenced by theses attractive and repulsive forces. The graph-drawing algorithm is an iterative process that begins with a random configuration of balls and proceeds to locate a minimum energy configuration by incrementally adjusting the relative positions of balls to equalise forces.

The attractive force f between balls is a spring force and follows Hooke's law, where the spring length is decided by the reverse of similarity between pairs of cases, i.e. $l_{u_i,u_j} = k_l[1 - sim(u_i, u_j)]$ where $u_i, u_j \in D$ or $l_{v_i,v_j} = k_l[1 - sim(v_i, v_j)]$ where $v_i, v_j \in S$, with D and S representing description space and solution space respectively. The repulsive force follows an inverse square law. Because we are representating the cases in separated description and solution spaces to have a clear idea of the relations between queries, descriptions and solutions, we need to modify the algorithm by adding a new spring force h exerted between the connected descriptions and solutions. The total force exerted on a ball can be represented in Equation (1):

$$\begin{aligned} f(u_i) &= \sum_{u_j \in D} f_{u_j u_i} + \sum_{u_j \in D} g_{u_j u_i} + \sum_{v_j \in sol(u_i)} h_{v_j u_i} \qquad if\ u_i \in D \\ f(v_i) &= \sum_{v_j \in D} f_{v_j v_i} + \sum_{v_j \in D} g_{v_j v_i} + \sum_{u_j \in des(v_i)} h_{u_j v_i} \qquad if\ v_i \in S \end{aligned} \tag{1}$$

The spring force f follows Hooke's law and in x direction we have:

$$\sum_{u_j \in D} f_x\,(u_j u_i) = \sum_{u_j \in D} k_f \frac{(d(u_j,u_i) - l_{u_i,u_j})(x_{u_j} - x_{u_i})}{d(u_j,u_i)} \tag{2}$$

Where $d(u_j, u_i)$ denotes the distance between the two balls representing the two descripions u_j, u_i and l_{u_i,u_j} is the spring length.

The repulsive force follows an inverse square law and in x direction we have:

$$\sum_{u_j \in D} g_x\,(u_j u_i) = \sum_{u_j \in D} k_g \frac{(x_{u_i} - x_{u_j})}{(d(u_j,u_i))^3} \tag{3}$$

The formulars with the solutions in the solution space and with the z direction are similar.

For a certain description, the second spring force h exerted by its connected solutions with zero spring length is as follows:

$$\sum_{v_j \in sol(u_i)} h_{v_j u_i} = \sum_{v_j \in sol(u_i)} k_h (x_{v_j} - x_{u_i}) \tag{4}$$

And verse versa for the the second spring force h exerted on the solutions.

According to Newton's second law and motion law, the relation between total force and displacement is as follows:

$$\Delta_x(u_i) = \frac{a_x(t_{k-1})}{2} \Delta_t^2 \text{ where } a_x(t_{k-1}) = \frac{f(u_i)}{m_i} \tag{5}$$

Δ_t is the time period of one animation step and $a_x(t_{k-1})$ means the accelation of the last time step. Here we set $\Delta_t = 10\text{ms}$. To avoid jumps in ball positions rather than smoothly interpolated transitions, we set the mass of all the balls equal to $m = 100000$ and limit the maximum distance that a ball can move in a single iteration according to Equation (5):

$$\Delta_x(u_i) = \begin{cases} -0.05 & if\ \ \Delta_x(u_i) \le -0.05 \\ \Delta_x(u_i) & if\ -0.05 \le \Delta_x(u_i) \le 0.05 \\ +0.05 & if\ \ \Delta_x(u_i) \ge 0.05 \end{cases} \tag{6}$$

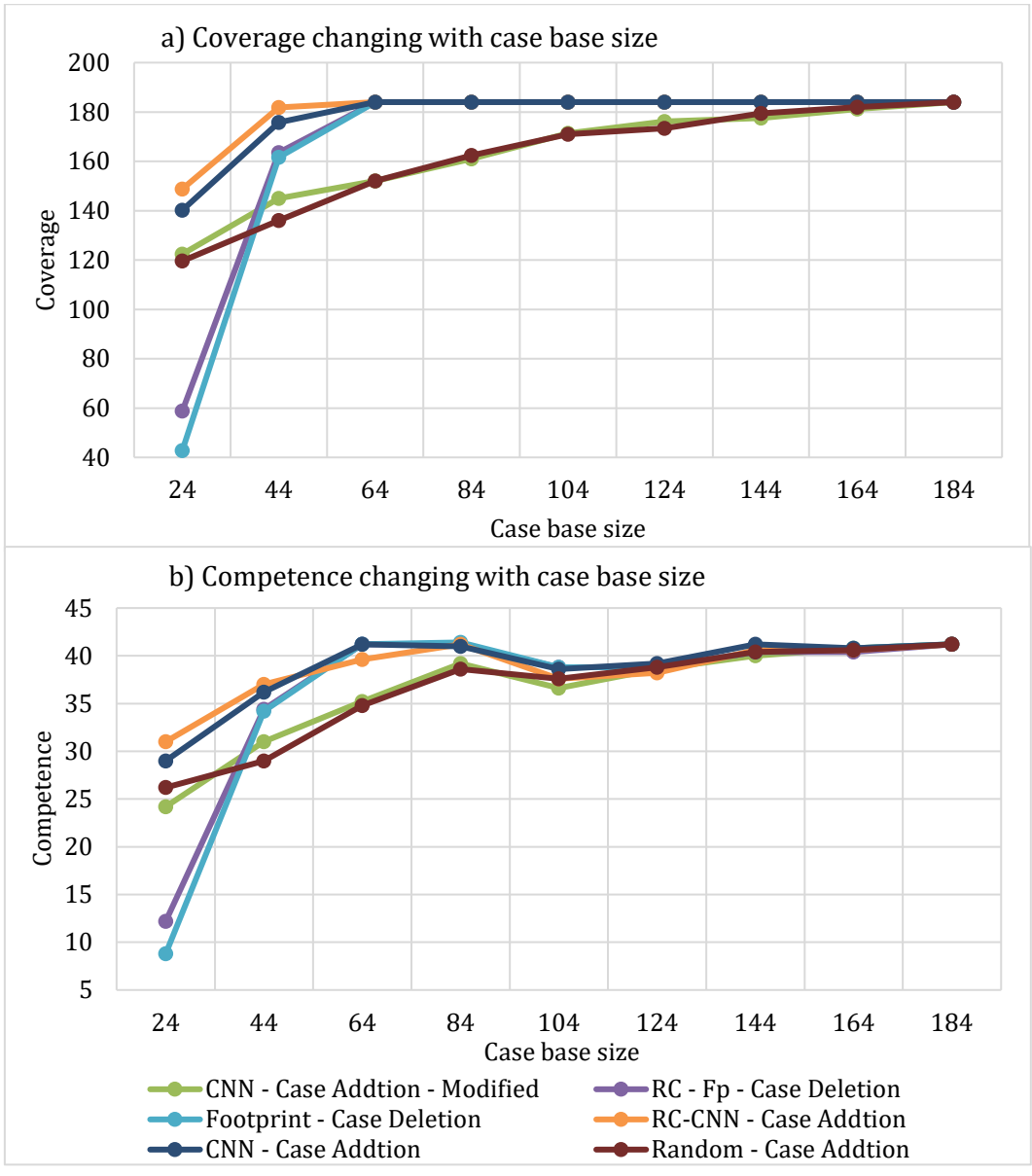
The two main tuning parameters available are the constants used in the force model to weight the contribution of the attractive (k_f) and repulsive (k_g) forces. For the purpose of our experiments, we set these values to respectively $k_f = 10,\ k_g = 0.3,\ k_h = 5,\ k_l = 20$.

To evaluate the quality and restoration of similarity relationships of the CBV algorithm, we use Pearson's correlation coefficient which measures the degree of linear relationship between the similarities and screen distances produced by CBV animations. We also calculate the Spearman's rank correlation coefficient that examines the relative ranking of descriptions and solutions concerning their similarities and screen distances (see Appendix 5).

4 Results and analysis

4.1 General Statistics of the Case Base

Statistical testing results are shown in Fig. 7-8. From Fig.7(a), we can conclude that it is possible to reduce the case base size by up to 65.2% (retain 64 out of 184 cases) while maintaining the same level of competence by applying the competence-preserving CBM methods. The average similarity between query and retrievals is reduced as a compromise of reduced case base size, but it is controlled by the similarity threshold and is maintained to an acceptable value. Interestingly, when we reduce the case base size from 184 to 64 applying different competence-preserving CBM methods, the diversity between retrievals rises from 0.5 to 0.7-0.8, which is a desired phenomenon. Results show that we can solve the problem of redundant solutions to some extent by applying competence-preserving CBM methods which reduces the density of cases in the case base.



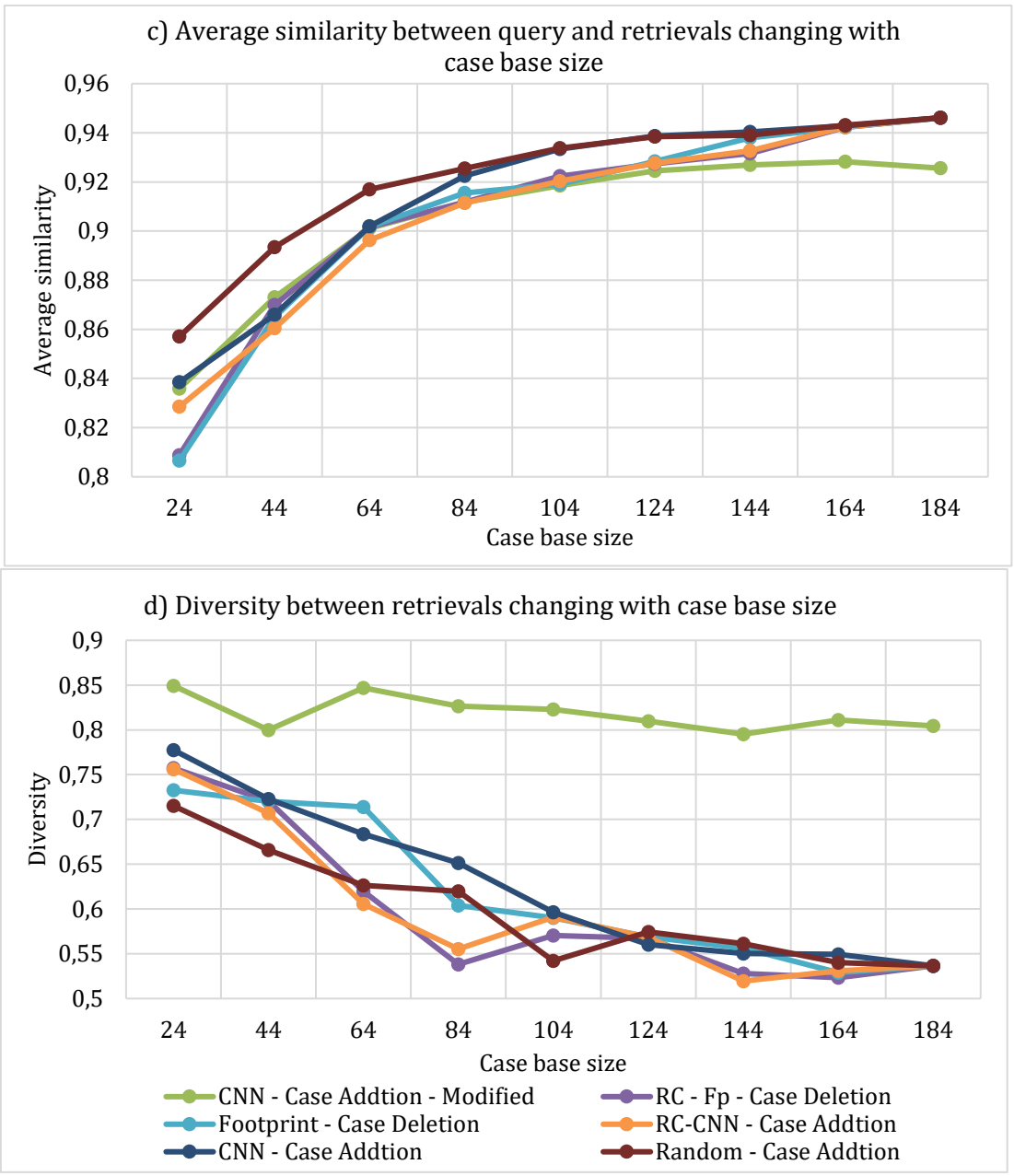
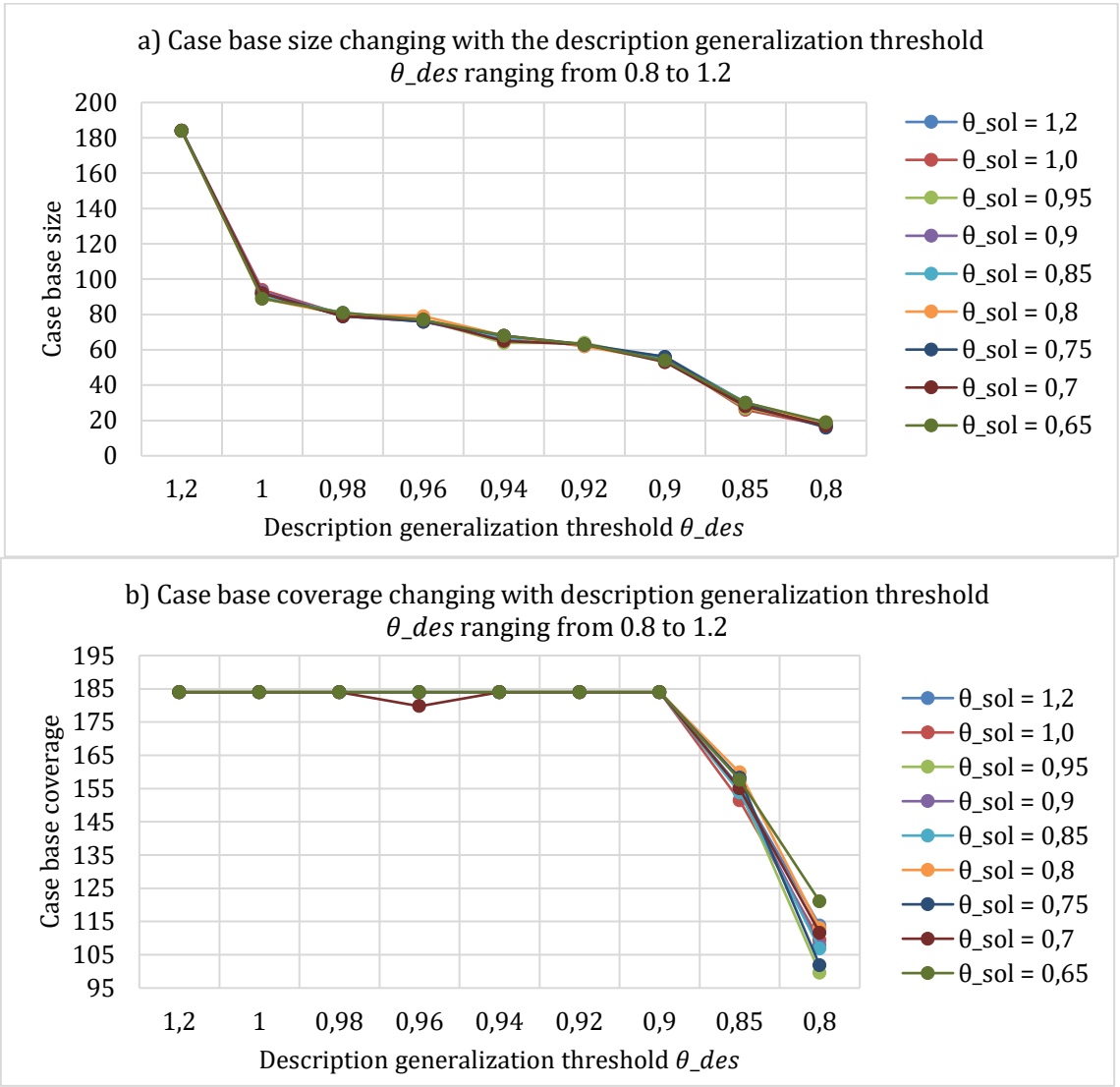


Figure 7: Statistical testing results of different CBM methods: a) Coverage, b) Competence, c) Similarity between query and retrieved descriptions and d) Diversity between retrieved solutions with regard to different predefined reduced case base sizes. The Modified CNN method is tested with a description generalization threshold θ_{des} =1.2 and a solution generalization threshold of θ_{sol} = 0.75.

Applying the Modified CNN method with a description generalization threshold of 1.2 and a solution generalization threshold of 0.75, we obtain the green curve in Fig.7. Since the similarity has a range of [0,1], by setting the description threshold as $\theta_{des} = 1.2$, we mean to avoid the generalization of descriptions to see the effects of purely generalization of solutions. Fig.7(d) shows that the generalization of solutions introduces a significant increase in diversity with a large number of cases in the case base. The method manages to improve diversity by up to 45% percent at the original case base size, making it not necessary to reduce the size of the case base from a diversity point of view, and it maintains the high level of diversity not changing with the reduction of case base size until there are only 24 cases in the case base. Regarding the case base coverage and competence, the curve of the Modified CNN method resembles random addition. It makes sense because θ_{des} corresponds

to the threshold of CNN method. With such a threshold, the cases in the case base are considered to have no coverage, and thus the CNN method reduces to the Random Addition method.

In order to better analyse the effects of the Modified CNN method, a second test is carried out to apply the generalization of descriptions and to use the description generalization threshold to control the size of the new case base. Results are shown in Fig. 8, where the light blue curve with θ_{sol}=1.2 serves as a control group, indicating that no generalization is applied to the solutions. From Fig. 8(a), we can see that the size of the case base reduces with the decrease of description generalization threshold θ_{des}. The relationship is not linear, but depends on the case distribution in the case base. Since most of the pairs of descriptions have a similarity $\theta = 1.0$ to each other, the case base size drops dramatically from 184 to 90 when we reduce the description generalization threshold θ_{des} from 1.2 to 1.0. Fig. 8(c) shows that the generalization of solutions does not have a significant effect on the case base competence, which is mainly decided by the threshold of description generalization. The case base competence is maintained before θ_{des} drops below 0.9. Similarly, Fig. 8(d) shows the influence of θ_{des} on average similarity between the query and retrievals, which drops gradually with the decrease of θ_{des} but is generally maintained before θ_{des} drops below 0.9.



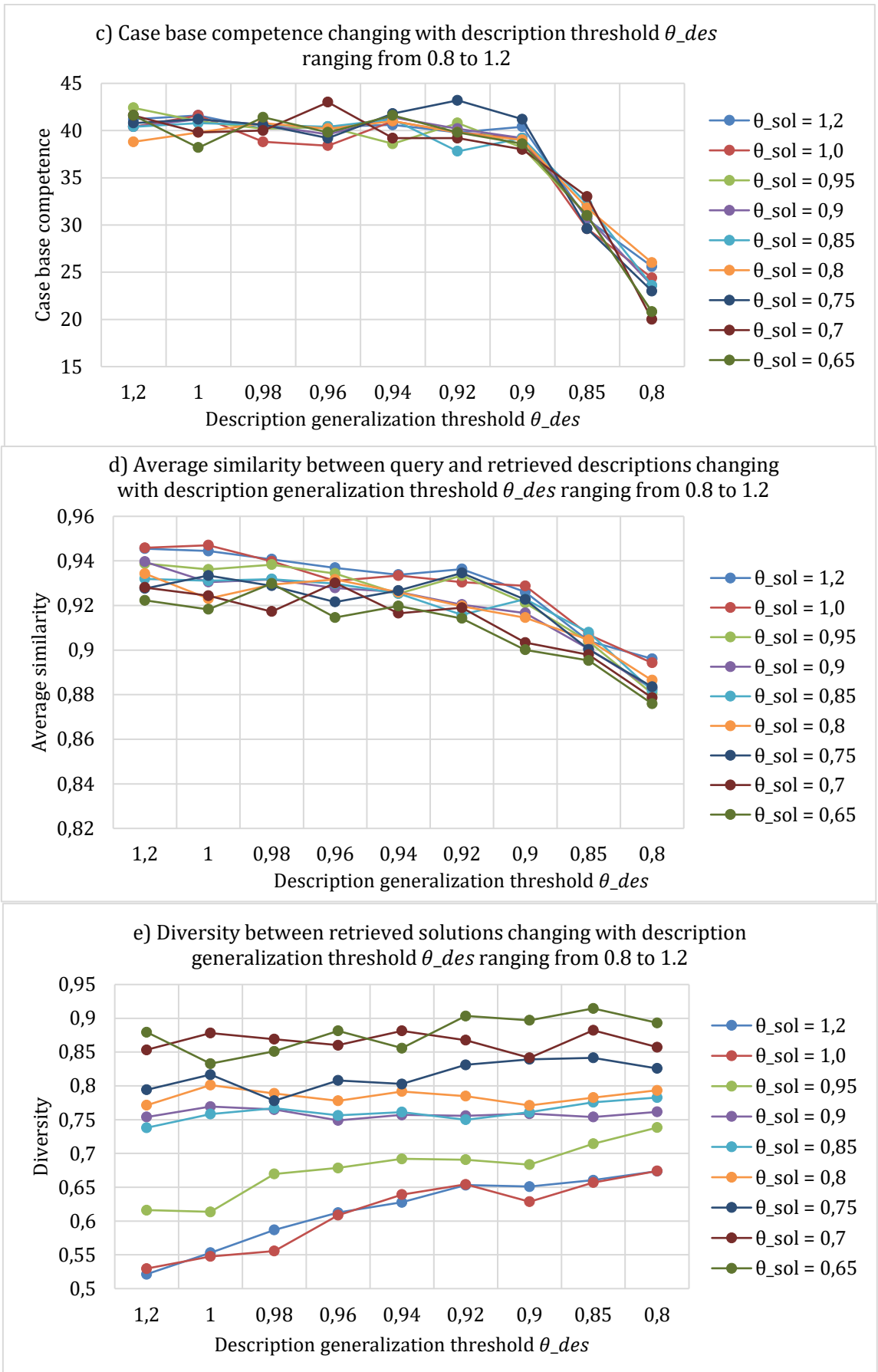


Figure 8: Statistical testing results of the Modified CNN method: a) Case base size, b) Coverage, c) Competence, d) Similarity between query and retrieved descriptions and e) Diversity between

retrieved solutions with regard to different threshold for descriptions θ_{des} ranging from 0.8 to 1.2 (no generalization). The different solution generalization threshold of θ_{sol} =0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 1.0, 1.2 are represented by curves of different colors in the figures.

From Fig. 8(e), we find that the average diversity between retrieved solutions is generally not influenced by θ_{des}, which verifies our assumption that the diversity of retrievals is determined by the distribution of solutions in the solution space, except for the situations of $\theta_{sol} \geq 0.95$, in which case the generalization of solutions occurs so rarely that the solution distribution is mostly indirectly determined by the position and generalization of their corresponding descriptions. The diversity between retrieved solutions is effectively increased with the decrease of the solution generalization threshold θ_{sol}.

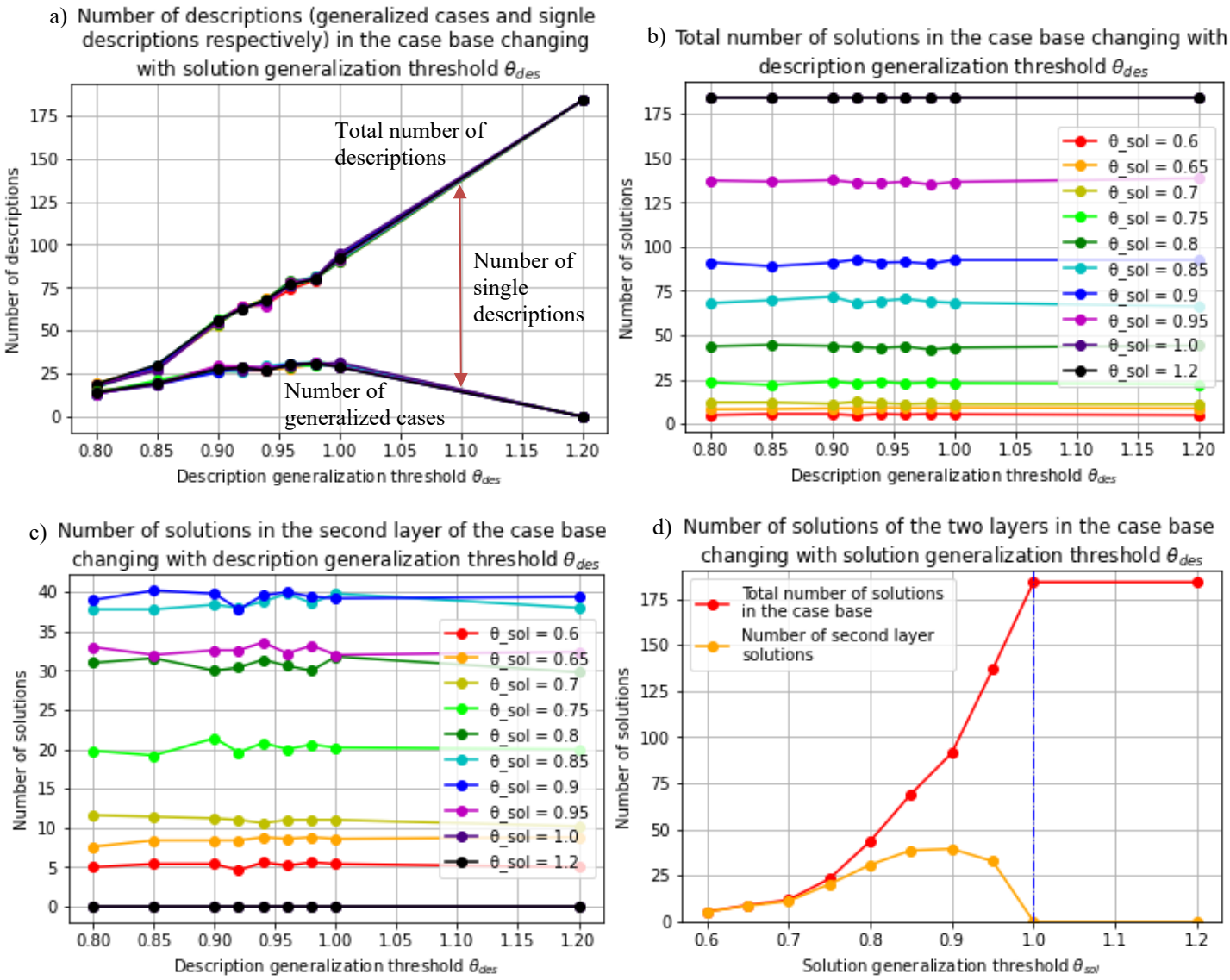


Figure 9: Case base statistics of the Modified CNN method: a) Number of descriptions (generalized cases and single descriptions respectively), b) Total number of solutions in the case base, c) Number of solutions in the second layer of the case base with regard to different threshold for descriptions θ_{des} ranging from 0.8 to 1.2 (no generalization), and d) Number of solutions of the two layers in the case base with regard to different threshold for solutions θ_{sol} ranging from 0.6 to 1.2. For figures a), b) and c), the different solution generalization threshold of θ_{sol} =0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 1.0, 1.2 are represented by curves of different colors in the figures.

To inspect the details of the generated case base applying Modified CNN method, we collect the case base information including number of descriptions (generalized cases and single descriptions), number of solutions in the two layers with regard to the description generalization threshold θ_{des} and solution generalization threshold θ_{sol}, as shown in Fig. 9. From Fig. 9(a), we can find that the generalization of solutions has no influence on the number of descriptions in the case base, neither generalized cases nor single descriptions. Looking at Fig. 9(b) and (c), we can also find that the generalization of descriptions has no influence on the number of solutions in the two layers of the case base, which shows a clear separation of descriptions and solutions. So we change the independent variable of x axis to the solution generalization threshold θ_{sol} and obtain Fig. 9(d) from average number of solutions in the case base with different description generalization thresholds θ_{des}.

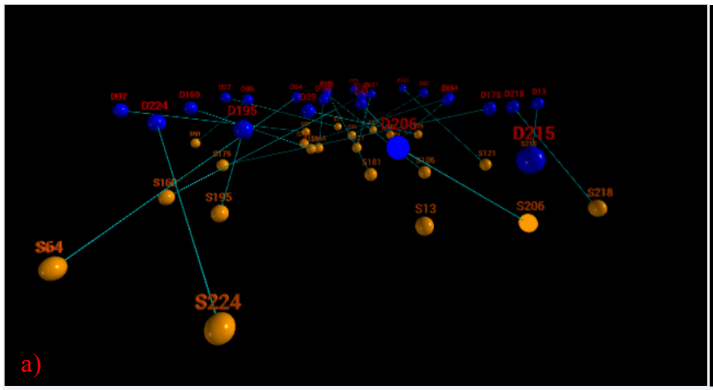
From Fig. 9(a) and (d), we can also see the trend and the process of generalization with the increase of description and solution generalization thresholds. At the beginning when the threshold of generalization is very low, there is only a small number of large clusters of generalized cases in the case base. As the threshold goes up, the clusters of generalized cases start to decompose into smaller clusters or single descriptions. When threshold becomes larger than the biggest similarity between pairs of cases, the generalized cases no longer exist and we only have single descriptions. It is the same with the solution space.

To conclude, we have proved the Modified CNN method to be useful in diversifying the retrievals through separation, generalization and reindexing of the solution space. The diversity of retrievals can be controlled by the solution generalization threshold θ_{sol}. It is also a feasible way to reduce the case base size and to preserve case base competence by adjusting the description generalization threshold θ_{des}.

4.2 Case Base Visualization Results

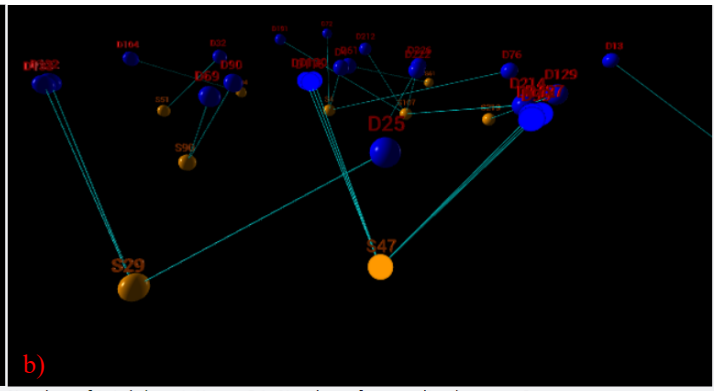
Applying the modified Force-Directed Graph-Drawing algorithm and with the information of objects stored in xml files, we realize the visualization of the generated case bases on a vpython canvas. Taking the parameters $k_f = 10$, $k_g = 0.3$, $k_h = 5$, $k_l = 20$, $m = 100000$, $\Delta_{x,max} = 0.05$ and $dt = 10$, we obtain plausible CBV results with the Pearson's correlation coefficient higher than 0.77 and the Spearman's rank correlation coefficient higher than 0.68. Results show that our implementation of CBV method is capable of producing sufficiently accurate representations of real case-bases.

The process of CBV is to first read and parse the xml files documenting case base information and query retrieval results generated from the CBM testbench in java, and to randomly generate the blue and red balls representing the single descriptions and generalized cases respectively located in a flat surface satisfying $x_{u_i}, z_{u_i} \in (-10, 10), y_{u_i} = 2$. Then the algorithm keeps updating the positions of descriptions. After the system of balls have reached an equilibrium state, the user can press the keyboard "up" to start the process of visualization of the solution space. Then the yellow balls representing the solutions are generated directly under the position of their corresponding descriptions with $x_{v_i} = \sum_{u_i \in des(v_i)} x_{u_i} \,/\, numberOf(des(v_i))$, $z_{v_i} = \sum_{u_i \in des(v_i)} z_{u_i} / numberOf(des(v_i))$, $y_{v_i} = 0$. The iteration of positions starts again and both descriptions and solutions are concerned this time. After reaching another equilibrium state, the user can press the keyboard "up" to continue the process. Upon the keyboard interaction, the green balls representing queries will appear on canvas, randomly distributed in a flat surface $x_{u_i}, z_{u_i} \in (-10, 10), y_{u_i} = 3$. And the iteration of positions starts again to find an appropriate position for each query according to its similarity relationships with the descriptions. This time, the descriptions and solutions are all fixed and their positions will not change any more. After the queries have reached an equilibrium state, the user can press the keyboard "up" to end the iteration process. At this moment, the case base is ready for inspection.



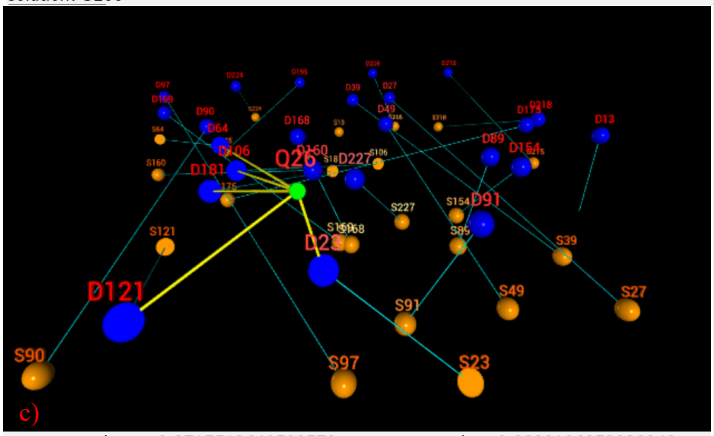
Number of total descriptions: 24, Number of generalized cases: 0
Number of solutions: 24, second layer: 0, first layer: 24

ρ_pearson_des = -0.8631136050873892, ρ_pearson_sol = -0.9067741263248792
ρ_spearman_des = 0.8705991578226991, ρ_spearman_sol = 0.9039615759966187
D206
task: Degradation modelling
caseStudyType: Electrical components
caseStudy: Electrolytic capacitors
onlineOffline: Online
inputForTheModel: Signals
sims: (D1,0.549) (D2,0.513) (D3,0.462) (D4,0.462) (D5,0.462) (D8,0.645) (D9,0.567)
coverage: 0
reachability: 0
solution: S206



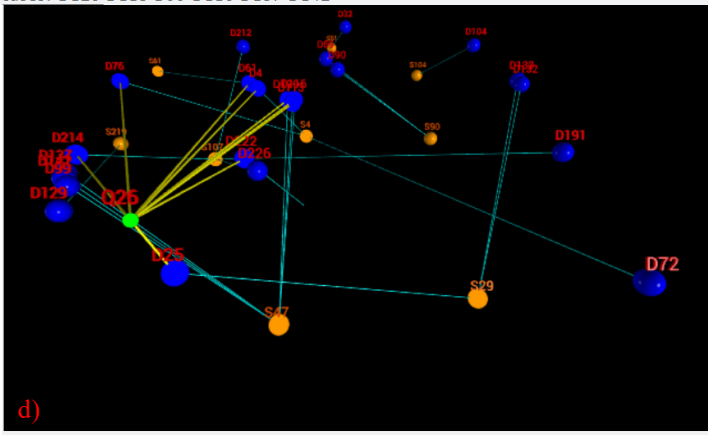
Number of total descriptions: 24, Number of generalized cases: 0
Number of solutions: 10, second layer: 9, first layer: 1

ρ_pearson_des = -0.9271874754062335, ρ_pearson_sol = -0.8642436650163448
ρ_spearman_des = 0.8975050467710165, ρ_spearman_sol = 0.8008017267961763
S47
layer: 2
sims: (S47,0.9999999999999999) (S40,0.923319626485358) (S42,0.92124513678000
modelApproach: Single model
modelType: Data-driven;
models: Deep belief network based hierarchical diagnosis network;
dataPreproc: Yes
complNotes: Data contains normal condition and 3 fault types with fault sizes; 0.18; 0.3
publicationIdentifier: doi.org/10.1109/TSMC.2017.2754287
performanceEvaluation: 1.0
cases: D120 D113 D99 D116 D137 D142



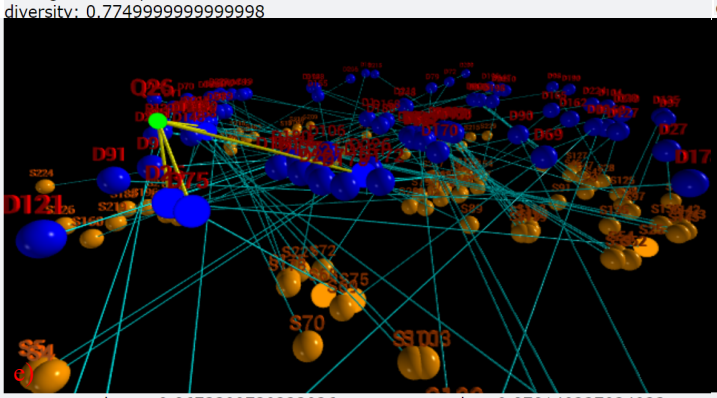
ρ_pearson_des = -0.8715512640599579, ρ_pearson_sol = -0.8829186853020943
ρ_spearman_des = 0.8761195062438154, ρ_spearman_sol = 0.8673575900433723

Query26
Q26
retrieved cases: (D23,1.0) (D181,0.781) (D64,0.779) (D106,0.756) (D121,0.75)
solutions: S23 S181 S64 S106 S121
average similarity: 0.8131999999999999
diversity: 0.7749999999999998



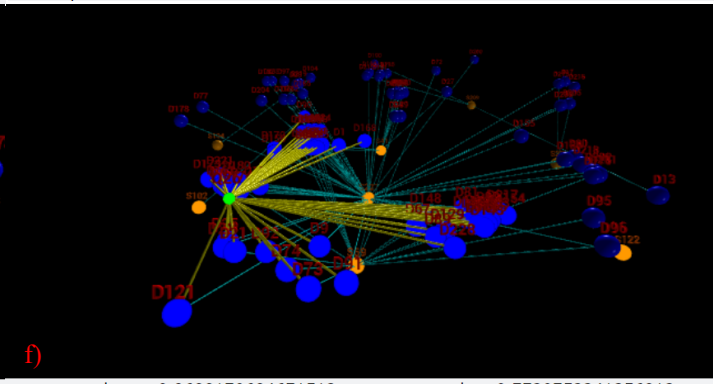
ρ_pearson_des = -0.9024587530736649, ρ_pearson_sol = -0.8450062509163836
ρ_spearman_des = 0.8917100649941839, ρ_spearman_sol = 0.7771199506629664

Query26
Q26
retrieved cases: (D25,1.0) (D76,0.78) (D214,0.78) (D222,0.78) (D4,0.779) (D113,0.7
solutions: S29 S4 S107 S47 S61
average similarity: 0.8038888888888889
diversity: 0.9460357364481695



ρ_pearson_des = -0.8673309730223026, ρ_pearson_sol = -0.878140227924923
ρ_spearman_des = 0.8602163277306505, ρ_spearman_sol = 0.8790879127922265

Query26
Q26
retrieved cases: (D25,1.0) (D68,0.781) (D181,0.781) (D16,0.78) (D75,0.78)
solutions: S25 S68 S181 S16 S75
average similarity: 0.8244000000000001
diversity: 0.6108



ρ_pearson_des = -0.8688170684671512, ρ_pearson_sol = -0.7730753241356813
ρ_spearman_des = 0.8659865928278392, ρ_spearman_sol = 0.68128078817734

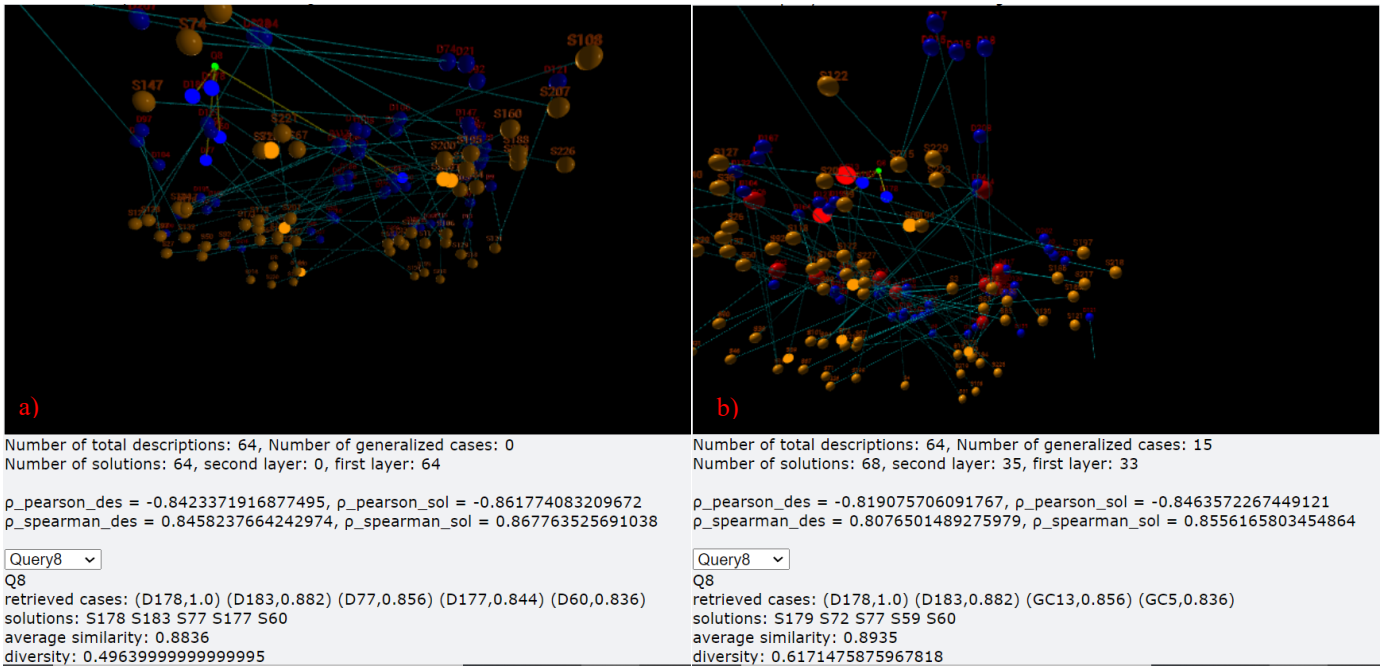
Query26
Q26
retrieved cases: (D21,1.0) (D23,1.0) (D25,1.0) (D67,0.781) (D74,0.78) (D81,0.78) (D1
solutions: S59 S227 S122 S41 S102
average similarity: 0.7708474576271184
diversity: 0.7967265733782769

Figure 10: Case Base Visualization applying the Force-Directed Graph-Drawing algorithm in vpython: a) Inspection of Description 206 in the of case base of size 24 generated by CNN method with $\theta = 0.9$; b) Inspection of Solution 47 in the of case base of size 24 generated by Modified CNN method with $\theta_{des} = 1.2, \theta_{sol} = 0.75$; c) Select Query 26 to inspect the retrieved descriptions and solutions, the average similarity and diversity in the of case base of size 24 generated by CNN method with $\theta = 0.9$; d) Select Query 26 to inspect the retrieved descriptions and solutions, the average similarity and diversity in the of case base of size 24 generated by Modified CNN method with $\theta_{des} = 1.2, \theta_{sol} = 0.75$; e) Select Query 26 to inspect the retrieved descriptions and solutions, the average similarity and diversity in the of case base of size 104 generated by CNN method with $\theta = 0.9$; f) Select Query 26 to inspect the retrieved descriptions and solutions, the average similarity and diversity in the of case base of size 104 generated by Modified CNN method with $\theta_{des} = 1.2, \theta_{sol} = 0.75$;

For inspection of the case base, the user can adjust the view over the canvas. It is also possible to click on the generalized cases (red balls), drag the mouse on single descriptions (blue balls) or solutions (yellow balls) to see the details of a certain object, as shown in Fig.10(a)-(b). Videos of inspection of visualized case bases can be found in the link: https://nextcloud.isae.fr/index.php/s/rgQx4PGdJdTXjEc.

From Fig.10(c)-(d), we can see that compared with original CNN method, the Modified CNN method has effectively improved the diversity of retrieved solution for Query 26 from 0.775 to 0.946. Fig.10(e) demonstrates on canvas two similar solutions retrieved for Query 26 from the case base generated by CNN method, which cause a drop in diversity of retrieved solutions to 0.611, while in Fig.10(f), the Modified CNN method still ensures an even distribution of solutions in the solution space and thus improves diversity of retrieved solutions to 0.797.

When $\theta_{des} \leq 1.0$, generalized cases begin to appear in the description space. The CBV of case bases of size 64 generated by CNN method with $\theta = 0.9$ and Modified CNN method with $\theta_{des} = 1.0, \theta_{sol} = 0.85$ respectively is shown in Fig.11.



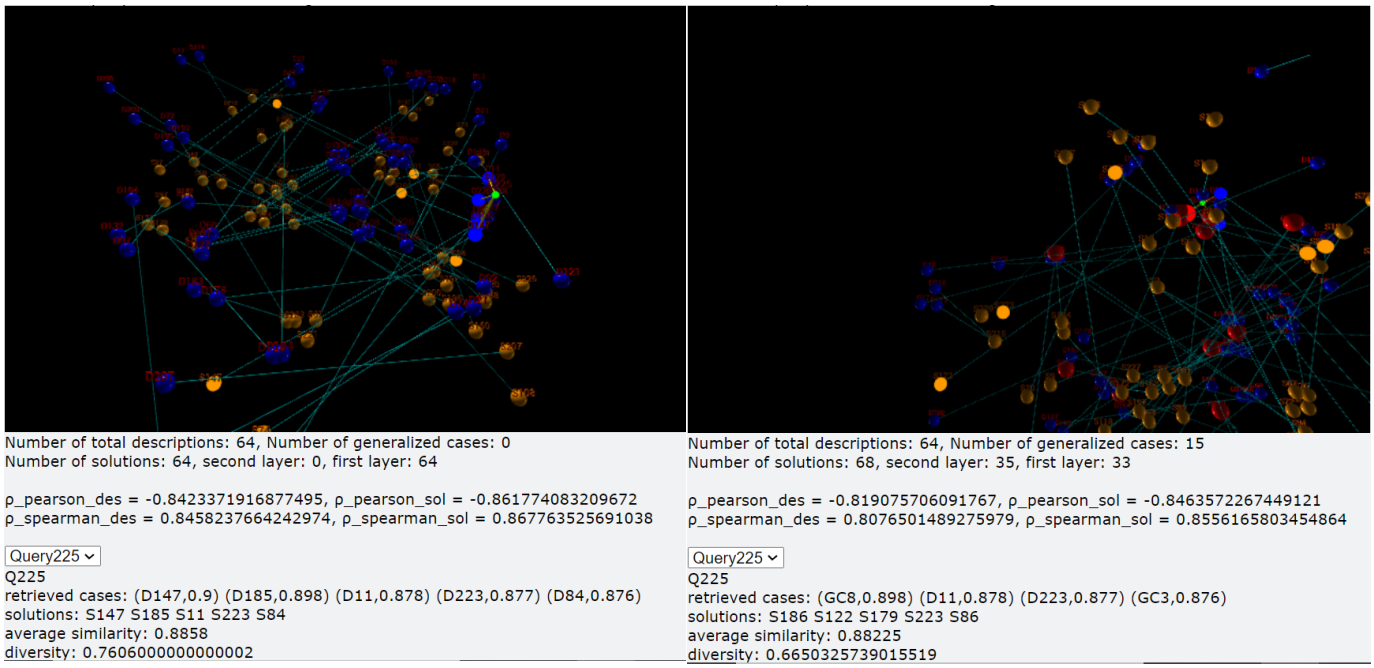


Figure 11: Case Base Visualization applying the Force-Directed Graph-Drawing Algorithm in vpython: To inspect the retrieved descriptions and solutions, the average similarity and diversity in the of case base of size 64, select a) Query 8 in the case base generated by CNN method with $\theta = 0.9$; b) Query 8 in the case base generated by the Modified CNN method with $\theta_{des} = 1.0, \theta_{sol} = 0.85$; c) Query 225 in the case base generated by CNN method with $\theta = 0.9$; and d) Query 225 in the case base generated by Modified CNN method with $\theta_{des} = 1.0, \theta_{sol} = 0.85$.

From Fig. 11(a)-(b), we can see that applying Modified CNN method with $\theta_{des} = 1.0, \theta_{sol} = 0.85$, the average similarity and diversity of retrivals for Query 8 are both improved compared with applying the CNN method with $\theta = 0.9$ on the same original case base. The average similarity increases from 0.884 to 0.894 and the diversity increases from 0.496 to 0.617. However, improvement in both indices is not always the case. Fig. 11(c)-(d) shows an exception where applying Modified CNN method with $\theta_{des} = 1.0, \theta_{sol} = 0.85$, the average similarity and diversity of retrivals for Query 225 both decrease compared with applying the CNN method with $\theta = 0.9$ on the same original case base. This situation can happen because the Modified CNN method is aimed at improving average diversity between solutions in the case base but not improving the diversity of every group of retrievals. Yet for most testing queries, we still witness an improvement in both average similarity and diversity of retrievals because the well chosen description generalization and solution generalization thresholds $\theta_{des} > \theta$ and $\theta_{sol} < \theta$. In this way, the Modified CNN method will generalize less on descriptions and more on solutions than the CNN method, making it possible to improve average similarity and average diversity of retrievals at the same time.

5 Conclusion and perspectives

The degree of novelty lies in the absorption and integration of methods and concepts from different domains, including competence-preserving CBM methods, definition of diversity metrics of retrievals and hierarchical case base memory structure adopted from the Machine Learning society.

Although generalization of cases is not a new term, only few papers discuss the separation of description space and solution space for the case base. However, in this project, we propose an approach, namely Modified CNN, to separate, generalize and reindex the descriptions and the

solutions, which can be integrated to competence-preserving CBM methods to diversify CBR retrieval results while reducing case base size and preserving competence at the same time.

Statistical testing results show that it is possible to reduce the predictive maintenance case base size by up to 65.2% while maintaining the same level of competence by applying competence-preserving CBM methods. In the meantime, diversity between retrievals also rises from 0.5 to 0.7-0.8, which is a desired phenomenon. To further improve diversity between retrievals, we propose the Modified CNN method which separates the generalization processes of descriptions and solutions and controls the case base size and solution diversity by two adjustable variables, i.e. description generalization threshold θ_{des} and solution generalization threshold θ_{sol}. Results show that the θ_{sol} does not have a significant influence on the case base competence, which is mainly decided by θ_{des}. The case base competence is maintained before θ_{des} drops below 0.9. On the other hand, the diversity between retrieved solutions is effectively increased with the decrease of the solution generalization threshold θ_{sol}. The diversity between retrievals is generally not influenced by θ_{des} except for the situations of $\theta_{sol} \geq 0.95$, in which case the generalization of solutions occurs so rarely that the solution distribution is mostly indirectly determined by the position and the generalization of their corresponding descriptions.

The CBV method of Force-Directed Graph-Drawing is applied in vpython to demonstrate intuitively the case distribution in the case base to the user and to show the effects of Modified CNN method on diversification of retrievals compared with original CNN method.

To conclude, we have proved the Modified CNN method to be useful in diversifying the retrievals through separation, generalization and reindexing of the solution space. The diversity of retrievals can be controlled by the solution generalization threshold θ_{sol}. It is also a feasible way to control the case base size and to preserve case base competence with the adjustable description generalization threshold θ_{des}.

Future work may revolve around the following perspectives:

1) In order to maximize the competence of the case base, the modification could be applied to more advanced methods such as RC-CNN method instead of the simple CNN method.
2) The testing results of CBM methods should be compared with the results of diversification methods that focus on the retrieval process such as Bounded Greedy Selection (BGS). The methods of diversification during retrieval process need to be integrated to the testbench.
3) Performance functions should be updated with the evolution of the case base, i.e. to learn new data from the cases with time. At present in our codes, we only use the data extracted from the original case base to calculate performance functions. It would also be interesting to integrate the publication year into the performance evaluation calculation since new researches are always based on old research papers.
4) The constants in the Force-Directed Drawing-Graph Algorithm still need to be studied to find the best combination of them to maximize the Pearson's correlation coefficient and Spearman's rank correlation coefficient.
5) A GUI could be integrated to CBV to show the user the possible generation of new case bases using different CBM methods and with different thresholds θ_{des} and θ_{sol} in order to give constructive suggestions for decision making on which method to take. The GUI should also facilitate the learning process of new cases and enable user manipulation of case base through graphical operations.

6 References

[1] A. Amoldt and E. Plaza, “Case-based reasoning: Foundational issues, methodological variations, and system approaches,” AI Communications, vol. 7, no. 1, pp. 39-59, 1994.

[2] R. Bergmann, “Experience Management: Foundations, Development Methodology, and Internet-Based Applications”, Lecture Notes in Computer Science, 2432. Springer, 2002.

[3] R. Bergmann, J. Kolodner et al., “Representation in case-based reasoning”, The Knowledge Engineering Review, Cambridge, UK: Cambridge University Press 20(3), pp. 209–213, 2005.

[4] P. Oehler, I. Vollrath, P. Conradi et al., “READEE - Decision Support for IP Selection using a Knowledge-Based Approach”, In Proceedings of IP98 Europe, Miller Freeman, 1998.

[5] M. Göker, T. Roth-Berghofer, et al., “The development of HOMER: A case-based CAD/CAM help-desk support tool”. See Smyth and Cunningham, pp. 346 – 357, 1998.

[6] B.W. Porter et al., “Concept learning and heuristic classification in weak-theory domains”, Artificial Intelligence, vol. 45, pp. 229-263, 1990.

[7] D.B. Leake and D.C. Wilson, “Combining CBR with Interactive Knowledge Acquisition, Manipulation and Reuse”, In Proceedings of the Third International Conference on Case-Based Reasoning and Development (ICCBR '99), pp. 203–217. Springer-Verlag, Berlin, Heidelberg, 1999.

[8] B. Smyth and P. Cunningham, “The Utility Problem Analysed: A Case-Based Reasoning Perspective”, Third European Workshop on Case-Based Reasoning, 1996.

[9] B. Keith and B. Smyth. “Improving Recommendation Diversity”, Proceedings of the 12th National Conference in Artificial Intelligence and Cognitive Science, California: AAAI Press, pp. 75-84, 2001.

[10] B. Smyth and P. McClave, “Similarity vs. Diversity”, In Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development, pp. 347–361, Springer-Verlag, Berlin, Heidelberg, 2001

[11] B. Smyth, Barry, and E. McKenna, "Building compact competent case-bases", In International Conference on Case-Based Reasoning, pp. 329-342. Springer, Berlin, Heidelberg, 1999.

[12] M.K. Haouchine, B. Chebel-Morello & N. Zerhouni, “Case Base Maintenance Approach”, International Conference on Industrial Engineering and Systems Management, pp. sur-CD, 2007.

[13] B. Smyth, "Case-base maintenance", In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pp. 507-516. Springer, Berlin, Heidelberg, 1998.

[14] S. Soltani and P. Martin, "Case-based reasoning for diagnosis and solution planning." Queen’s University Technical Report (2013-611), 2013.

[15] P. Hart, "The condensed nearest neighbor rule" IEEE transactions on information theory 14, no. 3, pp. 515-516, 1968.

[16] B. Smyth and M. T. Keane, “Remembering to forget: a competence-preserving case deletion policy for case-based reasoning systems”, In Proceedings of the 14th international joint conference on Artificial intelligence, vol. 1, pp. 377–382, 1995.

[17] D. B. Leake and D. C. Wilson, “Remembering Why to Remember: Performance-Guided Case-Base Maintenance”, In Proceedings of the 5th European Workshop on Advances in Case-Based Reasoning, Springer-Verlag, Berlin, Heidelberg, pp. 161–172, 2000.

[18] K. Maximini et al., “An Investigation of Generalized Cases”, Case-Based Reasoning Research and Development, pp. 261-275, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[19] R. Schmidt and L. Gierl, “Evaluation of Strategies for Generalised Cases within a Case-Based Reasoning Antibiotics Therapy Advice System”, Advances in Case-Based Reasoning, pp. 491-503, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[20] A. Tartakovski et al., “Applying Generalized Cases to Retrieval and Configuration of Life Insurance Policies”, Professional Knowledge Management, pp. 293-303, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[21] I. Bichindaritz, “Memory Structures and Organization in Case-Based Reasoning”, In: Perner, P. (eds) Case-Based Reasoning on Images and Signals, Studies in Computational Intelligence, vol. 73. Springer, Berlin, Heidelberg, 2008.

[22] J.H. Gennari, P. Langley, and D. Fisher, “Models of incremental concept formation”, UC Irvine: Donald Bren School of Information and Computer Sciences, 1988.

[23] D.H. Fisher, “Knowledge Acquisition Via Incremental Conceptual Clustering”, Machine Learning 2, pp. 139-172, 1987.

[24] J.J. Montero Jiminez, S. Schwartz, R. A. Vingerhoeds, B. Grabot, and M. Salaün, “Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics”, Journal of Manufacturing Systems, vol. 56, pp. 539–557, 2020.

[25] A.T. Miyagawa, “A case retrieval algorithm for predictive maintenance systems design”, tech. rep., ISAE-SUPAERO, Toulouse, 2020.

[26] J. Mazouzi, “Evolution of a Case Based Reasoning Program for Predictive Maintenance”, ISAE-SUPAERO, Toulouse, 2020.

[27] J. Kolodner, “Case-based reasoning”, Morgan Kaufmann, 2014.

[28] L. Zhang, F. Coenen & P. Leng, “An Experimental Study of Increasing Diversity for Case-Based Diagnosis”, In Proceedings of the 6th European Conference on Advances in Case-Based Reasoning (ECCBR '02). Springer-Verlag, Berlin, Heidelberg, 448–459, 2002.

[29] M.Y. Gu, “Supporting Generalized Cases in Conversational CBR”, MICAI 2005: Advances in Artificial Intelligence, pp. 544-553, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[30] B. Smyth et al., “Picture perfect - visualisation techniques for case-based reasoning”, In Proceedings of the 14th European Conference on Artificial Intelligence, IOS Press, NLD, pp. 65-69, 2000.

[31] D.B. Leake and D.C. Wilson, “Guiding Case-Base Maintenance: Competence and Performance”, Online Proceedings of the ECAI’2000 Workshop on Flexible Strategies for Maintaining Knowledge Containers, 2000.

[32] B. Smyth and E. McKenna, “Footprint-Based Retrieval”, In Proceedings of the Third International Conference on Case-Based Reasoning, ICCBR '99, Springer Verlag, Berlin, Germany, pp. 27-30, 1999.

[33] D.B. Leake and D.C. Wilson, “Categorizing case-base maintenance: dimensions and directions”, Advances in Case-Based Reasoning, 4th European Workshop on Case-Based Reasoning, EWCBR 98, Proceedings, Springer-Verlag, Berlin, pp. 196-207, 1998.

[34] S. Markovitch and P.D. Scott, “The Role of Forgetting in Learning”, In Proceedings of the Fifth International Conference on Machine Learning, pp. 459-465, 1988.

[35] S. Minton, “Qualitative Results Concerning the Utility of Explanation-Based Learning”, Artificial Intelligence, vol. 42, pp. 363-391, 1990.

[36] B. Smyth and E. McKenna, “Modelling the Competence of Case-Bases”, Advances in case-based reasoning, Lecture notes in computer science, Dublin, vol. 1488, pp. 208-220, 1998.

[37] K. Racine and Q. Yang, “On the consistency Management of Large Case Bases: the Case for Validation”, AAAI Technical Report, Verification and Validation Workshop, 1996.

Appendices

Appendix 1: Attributes of Cases in the Predictive Maintenance Case Base

The goal of our Predictive Maintenance case base is to summarize the main characteristics from successful applications of predictive maintenance models [25]. The selected case representation structure for our case base is a flat attribute-value representation. This representation structure was selected for simplicity and for compatibility with existing knowledge-based model tools.

Nineteen attributes are selected [25]:

1. Reference indicator: This variable defines an unique case Id. It is an integer variable.
2. Publication Year: This variable was selected for scrutiny purposes, a newer study may be privileged among the older ones. It is an integer variable.
3. Task: This variable summarizes the initial functional requirements of the proposed study. It is a structured text-based variable. The possible options for this variable are:
   - Features extraction
   - Fault detection
   - Fault identification/isolation
   - Degradation modelling
   - Health assessment/degradation
   - Next state forecasting
   - Remaining useful life (RUL) calculation
4. Case study: This variable gathers the different machines, equipment, components or technological systems on which the model was applied and validated. It is a structured text-based variable.
5. Case study type: It is a structured text-based variable with a limited number of options. The proposed types for the case studies:
   - Rotary machines
   - Reciprocating machines
   - Electrical components
   - Structures
   - Energy cells and batteries
   - Production lines
6. Input for the model: All predictive maintenance models assess data or information coming from the technical system to perform diagnostics and prognostics. These inputs can have different sources and types:
   - Signals: Direct measurements from the technical system sensors whose reads are signals. These signals can come from different types of sensors, such as: temperature, vibrations, pressure, spinning speed among others.
   - Time series (Discrete measurements or values): A single value is obtained to describe the state of an input from the technical system at a specific work cycle. It can be obtained from sensors, models outputs or other data sources.
   - Structured text-based: This type of variable is similar to time series. The data is recorded in discrete events, but the variables are text-based with a limited number of possible options for the instances.
7. Number of input variables: This is an integer variable that will help to assess the complexity of the case study.

8. Input type: This variable gives the nature of the input data for the model. It is a structured variable with a list of possible options.
9. Data pre-processing: Some deep learning models aim at not performing pre-processing while other models need adjustments on the input data to be trained and to perform their tasks. It is a binary variable with possible values “yes” and “no”.
10. Model approach: It is a binary variable, with two possible values: single-model approach or multi-model approach. This variable represents the number of models used to fulfil one single task in the consulted study.
11. Model type: It is a structured text-based variable linked to the model approach variable. if the model approach is single-model approach, three options are available: knowledge-based, data-driven and physics-based. If it is a multi-model approach, the combinations are possible: [knowledge-based, data-driven], [knowledge-based, physics-based], [data-driven, physics-based], [knowledge-based, data-driven, physics-based], [multiple-knowledge-based], [multiple-data-driven], [multiple-physics-based].
12. Models: This is a structured variable with a list of possible options. More than one model could be used for each task.
13. Online/Off-line: It is a binary variable. Online applications deal with real-time data which is constantly assessed to perform diagnostics or prognostics. Off-line applications gather information from an operative cycle to be assessed later.
14. Number of failure modes: It is an integer variable that will help to assess the complexity of the case study.
15. Performance indicator: Performance indicator allows validating a model for a specific task. For example, fault detection is usually assessed with the percentage of accurate detections, remaining useful life estimation is validated with standard deviation of the results. It is a structured variable with a list of possible options. More than one performance indicator can be included for each task. For example, if one study applies a score function and at the same time uses error percentage to measure the performance of a model, the performance indicator will be [“score function”, “percentage error”].
16. Performance: It is a numeric variable related to the performance indicator. This variable will be used for comparison purposes among the techniques applied for the same tasks. The format of this variable depends on the performance indicator format. There is performance variable for each performance indicator. A coma is used to separate multiple performance indicator values. As it is a numeric value that can have decimals the performance is recorded within round brackets. For the [“score function”, “percentage error”] example, the performance variable will be [(72,2),(1,23)]. Performance can be shown by graphics in the article, showing different precision at the different stages of the lifecycle. For these cases, a range of performance value is recorded. If the article proposes a performance indicator but the performance value is not explicitly shown, the numeric value will be replaced by a “N/A”.
17. Complementary notes: It is an open text variable that will gather complementary information to describe the complexity addressed on each study. Even when this variable is open text, it targets some specific features that are recorded in the same order if available: number of operational modes, number of training instances, number of testing instances to validate the approach, type of data-preprocessing and computational power needed to train or/and to run the models.
18. Study Title: The title of the study gathers concise information that could be used for retrieval purposes. It is an open text variable. Besides, the title will help to avoid the problem of duplicated instances.
19. Publication identifier: This variable is for consistency purposes. It is open text variable that gathers the alpha-numeric code used as publication identifier. It could be DOI, HAL Id, conference Id, among many others. The DOI will be the privileged one if more than one ID is available. This variable is for consistency purposes. If there exist contradiction among the

conclusions obtained from the database, a crossed validation could be done consulting the authors and journals. Reputation of authors and journals in the topic can help choose the right statement when contradictory conclusions are obtained. However, authors and journals are not direct consulting variables in the case base to avoid bias in the conclusions.

In this project, five attributes are selected for case description: Task (3), Case study (4), Case study type (5), Input for the model (6), Online/Off-line (13), and six attributes are selected for the solution part: Data pre-processing (9), Model approach (10), Model type (11), Models (12), Complementary notes (17), Publication identifier(19).

Appendix 2: Basics of Case-Based Reasoning

A case has two components: a characterization part and a lesson part. The space of characterization descriptions is denoted by D and the space of lessons is denoted by L. Thus a case c in the case base can be represented as a pair $c = (d, l) \in D \times L$ [2]. Existing structural case representation approaches include attribute-value representation, object-oriented representation, graph-based representation and predicate logic representation [3]. For the reason of simplicity and better compliance with existing CBR applications, we have chosen attribute-value representation for our predictive maintenance case base.

In attribute-value representation, each case is represented with a set of attributes $A_i, i = 1,2 \dots n$. Local similarity measure for an individual attribute A_i is $sim_{A_i}: T_{i_{range}} \times T_{i_{range}} \rightarrow [0,1]$, where $T_{i_{range}}$ is the range of the type of A_i.

For numerical attributes, similarity measures can be defined by numerical difference. For unordered symbolic attributes, we can use tabular similarity measures, which used in our CBR program. The table of local similarity between the pairs of case study type attributes and task attributes is shown in the Table 4 and 5. Apart from tabular similarity measures, taxonomies are also a possible similarity measure for symbolic variables. A taxonomy is an n-array tree in which the nodes represent symbolic values. It represents an additional relationship between the symbols through their position within the taxonomy tree.

Table 4: The table of local similarity between the pairs of the case study type attributes

Query\case	Rotary machines	Reciprocating machines	Electrical components	Structures	Energy cells and batteries	Production lines	Others
Rotary machines	1	0.75	0.1	0.65	0.1	0.3	0.2
Reciprocating machines	0.75	1	0.1	0.75	0.1	0.3	0.2
Electrical components	0.1	0.1	1	0.1	0.9	0.3	0.2
Structures	0.65	0.75	0.1	1	0.1	0.3	0.2
Energy cells and batteries	0.1	0.1	0.9	0.1	1	0.3	0.2
Production lines	0.3	0.3	0.3	0.3	0.3	1	0.2
Others	0.2	0.2	0.2	0.2	0.2	0.2	1

Table 5: The table of local similarity between the pairs of the task attributes

Query\case	Feature extraction	Fault detection	Fault identification/ isolation	Degradation modelling	Health assessment/ degradation analysis	Next state forecasting (one step)	Next state forecasting (multiplesteps)	Remaining useful life calculation
Feature extraction	1	0.8	0.7	0.3	0.3	0.2	0.2	0.1
Fault detection	0.8	1	0.9	0.5	0.5	0.2	0.2	0.1
Fault identification/ isolation	0.7	0.9	1	0.5	0.4	0.2	0.2	0.1
Degradation modelling	0.3	0.5	0.5	1	0.9	0.8	0.8	0.75
Health assessment/ degradation analysis	0.3	0.5	0.4	0.9	1	0.8	0.8	0.75
Next state forecasting (one step)	0.2	0.2	0.2	0.8	0.8	1	1	0.9
Next state forecasting (multiplesteps)	0.2	0.2	0.2	0.8	0.8	1	1	0.9
Remaining useful life calculation	0.1	0.1	0.1	0.75	0.75	0.9	0.9	1

For the String variables, we can use the Equality, Ngram or Levenshtein functions. Equality is a function that performs the comparison of equality between words. It indicates whether the word is similar or not. Ngram function compares two texts by separating them into N groups. These divisions are then compared to identify the number of groups that are equal between the two texts. Two benefits of this method are simplicity and scalability, since the only parameter that can be modified is N. Levenshtein function considers the minimum number of single character edits required to change one text into another. These edits can be insertions, delections or substitutions of characters. From this, a cost related to the number of editions, which represents the similarity distance, is recorded. With the above-mentioned functions, it is possible to calculate the similarity between two texts.

Global similarity measure [2] $sim_{\Phi}(\bar{x}, \bar{y}) = \Phi(sim_{A_1}(x_1, y_1), \dots, sim_{A_n}(x_n, y_n))$ reflects importance, relevance, and utility aspects of the local similarities and is modelled by the aggregation function $\Phi : [0,1]^n \rightarrow [0,1]$. In the amalgamation function, each variable has a weight and the function calculates the final similarity based on local similarities and weights. There are two options for amalgamation function in myCBR: weighted sum and euclidean distance.

- Weighted sum: The sum of similarities considering the weight of each one (see Equation (7)).

$$sim_{global} = \sum_{i=1}^{n} \omega_i sim_i \quad (7)$$

  Where the values of n, ω_i and sim_i are respectively the number of attributes, the weight and the local similarity of attribute i.
- Euclidean distance: The length of a line segment between the two points in Euclidean space. In the scope of the project, the distance is equal to global similarities (see Equation (8)).

$$sim_{global} = \sqrt{\sum_{i=1}^{n} \omega_i sim_i^2} \quad (8)$$

Where the values of n, ω_i and sim_i are respectively the number of variables, the weight and the local similarity of variable i.

In myCBR, the amalgamation function can be defined by the user together with the choice of query cases. In our project, the amalgamation function is set as Euclidean distance by default.

Contrast to a point case, a generalized case covers not only a point of the case space but a whole subspace of it, i.e. $gc \subseteq D \times L$. Substituting several point-cases by a single generalized case can reduce the size of the case base, but the similarity assessment can be more complex by this way. Fig.12 shows different types of generalized cases in the case representation space.

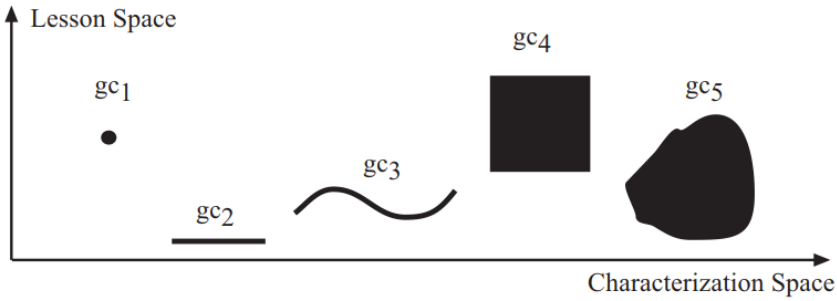


Figure 12: Different types of generalized cases in the case representation space [2].

Cases at higher levels of abstraction which are characterized through a reduced level of detail in the representation can be used as a kind of prototypes, and in turn the prototypes can be used as indexes to a larger set of related, more detailed cases. Such prototypes are abstract cases and they can have hierarchical representations.

Appendix 3: Details of the Competence-Preserving CBM Methods

Case-Base Maintenance (CBM) implements policies for revising the organization or contents (representation, domain content, accounting information, or implementation) of the case base in order to facilitate future reasoning for a particular set of performance objectives [33]. The choice of case-base maintenance strategies is driven by performance goals for the system and by constraints on the system's design and the task environment [31]. In general, there are multiple performance measures for a CBR system, and there is no guarantee that all of them can be maximized simultaneously [12]. Smyth and McKenna define three types of top-level goals for CBR systems [32]:

1. Problem-solving efficiency goals (e.g., average problem-solving time)
2. Competence goals (the range of target problems solved)
3. Solution quality goals (e.g., the error level in solutions)

CBM approaches may be divided into two policies, one concerning optimization and the other partitioning of the case base (See Fig.13). The former approach aims to reduce case research time, following an optimization policy that deletes least relevant cases using case addition or deletion strategies. The latter approach permits the retrieval of distributed case bases using an attribute selection case scheme.

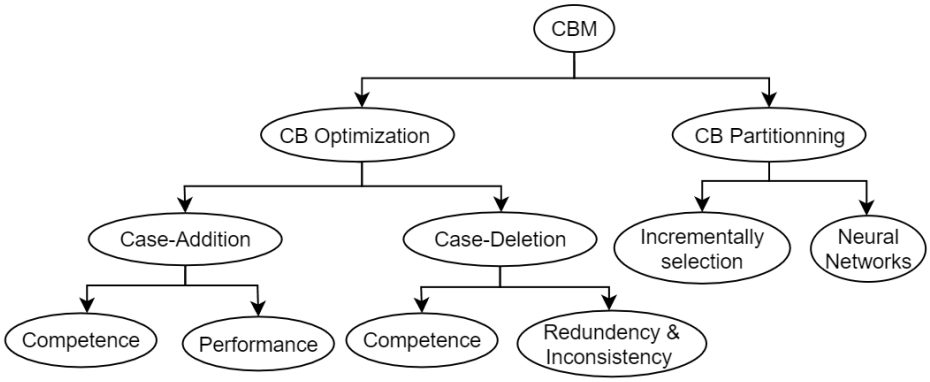


Figure 13: Different strategies and criteria used in CBM [12].

By the successive addition of cases to an original CB, reduced CB will be constructed, thus maximizing the criteria. Existing CBM case-addition strategies focus either on maximizing the competence criterion or on the performance criterion [12].

Smyth and McKenna present a method that uses an explicit case competence model based on notions of coverage and reachability. The relative coverage (RC) metric (see Equation 11) provides a precise measurement of competence contributions for individual cases. The RC metric, associated with the condensed nearest-neighbour (CNN) algorithm (see Fig. 13), permits to successively retain only those cases which are not solved by a case that has already been retained in the case base, in order to obtain a new reduced CB [11]. This method proves to have a good CB recovery.

The key concepts in modelling case competence are coverage and reachability. Given a case-base $C = \{c_1, \dots, c_n\}$ and $c^{©}$ is the set of target cases in the CB, we have:

- $Coverage(c) = \{c^{©} \in C : Adaptable\ (c, c^{©})\},\ for\ c \in C,$ (9)
- $Reachable(c) = \{c^{©} \in C : Adaptable\ (c^{©}, c)\},\ for\ c \in C,$ (10)

Thereby, they defined the RC metric as follows:

$$RC(c) = \sum_{c' \in Coverage(c)} \frac{1}{Reachability(c')} \quad (11)$$

```
O-SET    Original training examples
E-SET    {}
CHANGES    true

While CHANGES Do
          CHANGES    false
          For each case C O-Set Do
            If E-SET cannot solve C Then
              CHANGES    true
              Add C to E-SET
              Remove C from O-Set
            EndIf
          EndFor
EndWhile
```

Figure 13: Condensed Nearest-Neighbour Algorithm [11].

Another metric proposed by M.K. Haouchine et al. [12] is the Competence Metric (CM):

$$CM(c) = \frac{card(Coverage(c))}{card(Reachable(c))} \quad (10)$$

In this project, we adopt RC-CNN and CNN methods instead of CM-CNN method because of our assumption that the coverage and reachable of a case is the same set of cases. In this way we always have a CM=1 which can not guide the CB optimization.

By analogy to the RC metric, Leake and Wilson also developed a Relative Performance (RP) metric aimed at assessing the contribution of a case to the adaptation performance of the system [17]. For each case that might be added to the CB, its contribution to adaptation performance is estimated. The RP value reflects a case's contribution to adaptation performance compared to other cases. This metric can be used to guide case addition to optimize CB adaptation performance.

On the other hand, the case-deletion strategies value cases in the original CB according to some criteria in order to be able to suppress some cases to bring the CB to a specified number of cases. When the CB reaches a certain threshold, it is screened entirely, usually followed by the process of case-deletion. The evaluation criteria such as competence, redundancy and inconsistency have been used in different methods listed below [12]:

- Random Deletion: This is a very simple, inexpensive method and it is completely domain independent. Simply randomly select and delete a case from the CB once the CB size exceeds some predefined limit [34].
- Ironically: It is a slightly more complicated method, it calculates the frequency of each case that is retrieved and it deletes cases which are not frequently accessed in the CB [35].
- Utility Deletion (UD): It is based on Minton's utility metric which chooses a case item for deletion by estimating its performance benefits. The utility problem manifests itself as a trade-off between the solution quality and the efficiency problem of working with a large CB. System efficiency is measured by taking the mean time to solve a target problem. The solution quality is bound to the percentage of good answers provided by the system. Solution quality increases with CB size [36].
- Deletion based on redundancy and inconsistency: After a series of test on each CB case concerning redundancy and inconsistency, specific cases can be removed or kept after the user approval [37].
- Deletion based on CB size and density: This method studies the size, the density of the CB and the distribution of cases in the CB. It tries to homogenize the case density [36].

According to the coverage and reachable sets of each case, the cases can be categorized into 4 categories, as shown in Fig.14:

- Pivotal Cases: $Pivot(c)\ if\ Reachable(c) - \{c\} = \emptyset$, i.e. if a case can not be solved by any other cases in the case base, it is a pivotal case.
- Support Cases: $Support(c)\ if\ \exists c^{©} \in Reachable(c) - \{c\}: Coverage(c^{©}) = Coverage(c)$, i.e. if the coverage of a case is equal to one of its reachable cases and if it is not an auxiliary case, it is then a support case and it forms a support group with the other cases with the same coverage.
- Spanning Cases: $Spanning(c)\ if\ \neg Pivot(c) \wedge Coverage(c) \cap (\cup\ Coverage(c^{©})) \neq \emptyset$, i.e. if the coverage of the case connects to the coverage of the other cases in the case base, it is a spanning case.

- Auxiliary Cases: $Auxiliary\ (c)\ if\ c^{©} \in\ Reachable(c) - \{c\}: Coverage(c)\ \not\subset\ Coverage(c^{©})$, i.e. if the coverage of a case is totally covered by one of its reachable cases, it is an auxiliary case.

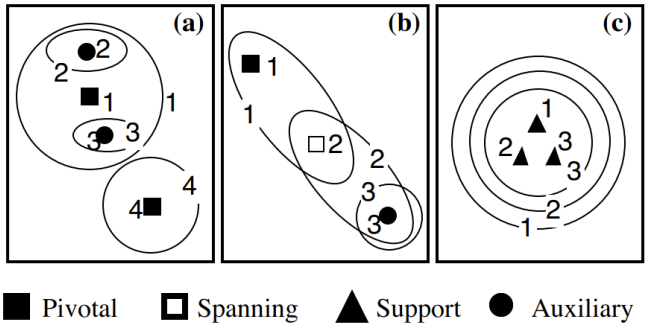


Figure 14: Four Case Categories [16]

Based on the 4 case categories, B. Smyth and M.T. Keane [16] propose the Footprint Deletion (FD) method which removes irrelevant cases to guide the CB towards an optimal configuration that maximises competence while minimising size. The case categories provide a means of ordering cases for deletion in terms of their competence contributions. Auxiliary cases are selected for deletion before support cases, which in turn are chosen before spanning and pivotal cases. The optimal CB can be constructed from all the pivotal cases plus one case from each support group.

In this project, we adopt FD method and RC-FD method to pick the case with lowest RC to delete when there are several candidates. We focus on the competence instead of the utility of each case because the computation time of CBR system is determined by many operational environment facts and is not easy to measure and to compare in a fair manner.

Appendix 4: Details of the Bounded Greedy Selection Algorithm

According to K. Bradley and B. Smyth [9-10], the similarity (Equation 12) of a set of cases is defined as the average of the similarity of the retrieved cases to the query q, while the diversity (Equation 13) of a set of cases is defined as the average dissimilarity between all pairs of cases in the result-set:

$$Similarity(q, c_1, \dots\ c_n) = \frac{\sum_{i=1..n} Similarity(q, c_i)}{n} \tag{12}$$

$$Diversity(c_1, \dots\ c_n) = \frac{\sum_{i=1..n} \sum_{j=1..n} (1 - Similarity(, c_i, c_j))}{\frac{n}{2} * (n-1)} \tag{13}$$

The diversity preserving *Bounded Greedy Selection* algorithm is based on the *Quality Metrics* (Equation 14-15), a measure that explicitly combines both similarity and diversity.

$$Quality(t, c, R) = (1 - \alpha) * Similarity(t, c) + \alpha * RelDiversity(c, R) \tag{14}$$

$$\begin{aligned} RelDiversity(c, R) &= 1\ if\ R = \{\}; \\ &= \frac{\sum_{i=1..m} (1 - Similarity(c, r_i))}{m}, otherwise \end{aligned} \tag{15}$$

where $RelDiversity$ is measured between a candidate case c from the case base and the current retrieval set of cases $R = \{r_1, \dots, r_m\}$.

In our project, 5 features in problem description space, i.e. Task (Symbol attribute), Case study type (Symbol attribute), Case study (String attribute), Online/Off-line (Symbol attribute), Input for the model (Symbol attribute), are considered as features determining average $Similarity(q, c_i)$, and 6 features in solution space, i.e. Model Approach (Symbol attribute), Model Type (Symbol attribute), Models (String attribute), Data Pre-processing (Symbol attribute), Complementary notes (String attribute), Publication identifier (String attribute), are considered as features determining $Similarity(c_i, c_j)$ between the retrievals to compute the diversity.

Appendix 5: Pearson's correlation coefficient and Spearman's rank correlation coefficient

Correlation is a statistical measure that tells about the association between the two variables. It describes how one variable behaves if the other variable changes. If the two variables increase or decrease in parallel, then they have a positive correlation between them. If one variable increases while another one decreases, then they have a negative correlation with each other. If the change of one variable has no effect on another variable, then they have a zero correlation.

Both Pearson's and Spearman's rank correlation coefficients are used to measure the correlation. The difference is that Pearson's correlation coefficient evaluates the linear relationship between two continuous variables, while Spearman's rank correlation evaluates the monotonic relationship. The Spearman correlation coefficient is based on the ranking of the values for each variable rather than the raw data. The formular of Pearson's correlation coefficient is given by Equation (16) and Spearman's rank correlation coefficient is given by Equation (17).

$$\rho_{Pearson} = \frac{cov(X,Y)}{\sigma_x \sigma_y} = \frac{\sum(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum(x_i-\bar{x})^2 \sum(y_i-\bar{y})^2}} \tag{16}$$

Where $cov(X, Y)$ is the covariance between variables X and Y, σ_x is the standard deviation of X, σ_y is the standard deviation of Y, x_i is values of the x variable in a sample, $\bar{x}$ is mean of the values of the x variable, y_i is values of the y variable in a sample, $\bar{y}$ is mean of the values of the y variable.

$$\rho_{Spearman} = \frac{\sum(R(x_i)-\overline{R(x)})(R(y_i)-\overline{R(y)})}{\sqrt{\sum(R(x_i)-\overline{R(x)})^2 \sum(R(x_i)-\overline{R(x)})^2}} \tag{17}$$

Where $R(x_i)$ is rank values of the x variable in a sample, $\overline{R(x)}$ is mean of the rank values of the x variable, $R(y_i)$ is rank values of the y variable in a sample, $\overline{R(y)}$ is mean of the rank values of the y variable.

Appendix 6: Explanation on codes of the CBM testbench

To calculate the diversity between retrievals, it is necessary to compute the similarity between each pair of retrieved cases concerning the attributes in the solution part. Thus, a new method called solveCaseDiv() is added in the Recommender class. This method is similar to the solveOuery() method which solves the similarity between cases and a query, in that it initializes a query, adds local similarity functions for each attribute, adds contents into the query, sets aggregation function, start retrieval and return the retrieval results. The difference lies in the attributes taken to compute local similarities. Instead of taking the 5 attributes from the problem description part, the solveCaseDiv() method takes the 6 attributes from the solution part. Since there are 3 multiple-value attributes out of the 6 attributes from solution part, we have checked the user guidance of myCBR and adapted the codes for similarity calculation of 3 multiple-value attributes. An example of the model type attribute is shown as follows:

```
// Add the multiple-value attribute - model type
String[] modelTypeValues = modelType.split(",");
// define a list that will be used to store the values
LinkedList<Attribute> list = new LinkedList<Attribute>();
for (String value: modelTypeValues)
    list.add(modelTypeDesc.getAttribute(value));
MultipleAttribute<SymbolDesc> modelTypeMultiAttr = new
MultipleAttribute<SymbolDesc>(modelTypeDesc, list);
// add the query attribute to the list
query.addAttribute(modelTypeDesc.getName(), modelTypeMultiAttr);
```

The testbench for competence-preserving CBM methods is built in the CBM class. This class has 6 main methods: divideCB(), generateQueries(), computeSimilarityMatirx(), caseBaseInit(), generateNewCBandRetrieve() and calcuAverage(). In a sequence, they divide the case base into two parts of 184 cases and 46 queries, generate a query file from the queries, generate a query file from the cases calculate similarity between each pair of cases and build the similarity matrix of cases, compute coverage and reachability for each case, categorize the cases and generate pairs of descriptions and solutions from the cases, then generate new case bases from the old case base applying several CBM methods and test the new case bases with 46 queries. At last, after all the iterations, the calcuAverage() method calculates the average values of similarity, diversity, case base coverage and competence of all the iterations.

```
public static void main(String[] args) {
    ArrayList<AllInfo> info = new ArrayList<AllInfo>();
    CBM cbm = new CBM();
    for (int k=0;k<iterations;k++) {
        // divide the case base into two parts: 184 cases and 46 queries
        cbm.divideCB();
        // generate query file, arguments are: query data, query file path to generate the
        // queries, number of retrievals
        cbm.generateQueries(cbm.queryText, data_path_rlts + "Data Queries" + (k + 1) + ".csv",
                            String.valueOf(retrieveNum));
        // generate a query file from the cases to compute the similarity between pairs of cases
        cbm.generateQueries(cbm.caseText, data_path_rlts + "casesSimTestQuery" + (k + 1)
                            +".csv","");
        // calculate similarity between each pair of cases and build the similarity matrix of
        // cases, arguments are: query file path to read the queries, output file name,
        // similarity matrix file name, case id list
        cbm.computeSimilarityMatirx(data_path_rlts + "casesSimTestQuery" + (k + 1) +".csv",
                                    "casesSimMatrix", "similarity matrix" + (k + 1) +".csv",
                                     cbm.caseId);
        // compute coverage and reachability for each case, categorize the cases, generate pairs
        // of descriptions and solutions from the cases
        cbm.caseBaseInit(k);
        // generate new cases base from the old case base applying several CBM methods, test the
        // new case bases with 46 queries, return the case base information (case Id, case base
        // coverage, etc) and testing results (lists of retrieval and similarity pairs), and
        //store all the obtained data stored in the info variable
        info.add(cbm.generateNewCBandRetrieve(cbm, k+1));
    }
    // calculate average values (average similarity, diversity, case base coverage and
    // competence) of all the iterations from the stored data in the info variable
    cbm.calcuAverage(info, info.get(0).getNumOfMethods());
}
```

The generateNewCBandRetrieve() method is composed of 5 steps for each CBM method, as shown in the codes below: First, we generate new case base from the old case base applying a certain CBM method; then we use the new case base to retrieve for the 46 queries and return the result lists; afterwards we analyze the result lists to obtain the average similarity and the diversity of retrievals; then we store the case base information and the testing results into the variables

cbInfoList and annalist; in the end we write the case base information and the testing results into a csv file.

```
int n = 0;  // number of size iteration
for (int finalSize = min; finalSize <= max; finalSize += step) {
    n++;
    // generate new case base from the old case base applying Random Addition CBM Method and
    // store the information of the new case base in the variable cbinfo
    cbInfo = cbm.CBMAddRdm(finalSize);
    // use the new case base to retrieve for the 46 queries and return the result lists
    rltLists = cbm.retrieve(data_path_rlts + "Data Queries" + iteration + ".csv", "addRdm",
                            "addRdmResults.csv", cbInfo.getCaseIdList());
    // analyze the result lists to obtain the average similarity between the queries and
    // retrievals and the diversity of retrievals
    ana = cbm.analyseResults(rltLists, "addRdm",n,iteration);
    // add the case base information and the testing results into the lists cbInfoList and
    // anaList where we store the information of all the CBM methods
    cbInfoList.add(cbInfo);
    anaList.add(ana);
    // write the case base information and the testing results into a csv file
    try{
        File file = new File(data_path_rlts + "testingrlt" + iteration + ".csv");
        FileOutputStream fos = null; //add lines
        fos = new FileOutputStream(file,true);
        OutputStreamWriter osw = new OutputStreamWriter(fos, "UTF-8");
        osw.write("Random - Case Addtion;" + cbInfo.getCaseIdList().size() + ";" +
                  cbInfo.getCoverage() + ";" + cbNum + ";" + ana.getNumSolved() + ";" +
                  cbm.queryId.size() + ";" + String.valueOf(ana.getAverageSim()).replace(".",
                  ",") + ";" + String.valueOf(ana.getAverageDiv()).replace(".", ",") + ";" +
                  cbInfo.getCaseIdList().toString());
        osw.write("\r\n"); osw.close();
    }catch (Exception e) {
        e.printStackTrace();
    }

    // repeat the above-mentioned steps for the other 5 CBM methods
    cbInfo = cbm.CBMAddCNN(finalSize);
    rltLists = cbm.retrieve(data_path_rlts + "Data Queries" + iteration + ".csv", "addCNN",
                            "addCNNResults.csv", cbInfo.getCaseIdList());
    ...
}
```

The private variables of the CBM class are listed below:

```
private static String data_path = "E:\\myCBR_SDK - copie -
copie\\Internship_Project\\data_and_mycbr\\";
private static String data_path_rlts = "E:\\myCBR_SDK - copie -
copie\\Internship_Project\\rlts\\";
private static String data_path_CBs = "E:\\myCBR_SDK - copie -
copie\\Internship_Project\\rlts\\CBs\\";
// Name of the file containing the case base
private static String csv = "Data V05-10-2020-copy.csv";
// Name of the file containing the divided case base with 184 cases
private static String csv1 = "Data CaseBaseCases.csv";
// Name of the file containing the 46 cases to generate queries
private static String csv2 = "Data QueryCases.csv";
// Name of the file containing the 46 queries generated from 46 cases
private static String csv3 = "Data Queries.csv";
private static DecimalFormat df = new DecimalFormat("###0.000");
private ArrayList<String> oriCaseText= new ArrayList<String>();
private ArrayList<String> queryText= new ArrayList<String>();
private ArrayList<String> caseText= new ArrayList<String>();
// predefined number of the cases after the division of the case base
private static int cbNum = 184;
private static int retrieveNum = 5; // number of retrievals required by the user
private static int methodNum = 6; // number of CBM methods to be tested
```

```
private static int iterations = 5; // number of iterations of testing
private static int min = 24; // minimum case base size to be generated using CBM methods
private static int max = 64; // maximum case base size to be generated using CBM methods
private static int step = 20; // step of the case base size between minimum and maximum
// list of case Id after division of the original case base
private ArrayList<Integer> caseId = new ArrayList<Integer>();
// list of query Id after division of the original case base
private ArrayList<Integer> queryId = new ArrayList<Integer>();
// predefined threshold to calculate coverage and reacheability to model the competence of cases
private double threshold = 0.9;
// the matrix stores the similarities between each pair of cases
private ArrayList<List<Entry<Integer, Double>>> rltMatrix = new ArrayList<List<Entry<Integer,
Double>>>();
// predefined threshold for solution generalization for the Modified CNN Method
private double sol_sim_threshold = 0.85;
// predefined threshold for description generalization for the Modified CNN Method
private double des_sim_threshold = 1.0;
private ArrayList<Case> cases = new ArrayList<Case>();
// this list stores all the descriptions separated from solutions of the cases
private ArrayList<Description> descriptions = new ArrayList<Description>();
// this list stores all the solutions separated from descriptions of the cases
private ArrayList<Solution> solutions = new ArrayList<Solution>();
// this list stores support groups for the categorization of cases for FD and RC-FD CBM methods
private ArrayList<ArrayList<Case>> supportGroups = new ArrayList<ArrayList<Case>>();
// the lists below are to store pivotal cases, spanning cases, support cases and auxiliary cases
for FD and RC-FD CBM methods
private ArrayList<Case> pivotalCases = new ArrayList<Case>();
private ArrayList<Case> spanningCases = new ArrayList<Case>();
private ArrayList<Case> supportCases = new ArrayList<Case>();
private ArrayList<Case> auxiliaryCases = new ArrayList<Case>();
// these variables are used to generate new case bases in the CBR Engine to perform customized
retrievals
private DefaultCaseBase testCB = null;
private DefaultCaseBase testCB1 = null;
private Recommender remy = new Recommender();
// collection of instances extracted from the PredictMaint_myCBR.prj project file
private Collection<Instance> ccs;
// list of instances from the PredictMaint_myCBR.prj project file in ascending order of
references
private ArrayList<Instance> lcs = new ArrayList<Instance>();
// this variable stores all the performance functions learned so far from the cases in the case
base
private Hashtable<String,PerformanceFunction> pfs = new Hashtable<String,PerformanceFunction>();
```

The PerformanceFunction class is used to store the information of each performance indicator. It has 4 private variables: perfIndicator, average, standardDeviation and values. This class has three main methods: insertValue(), calcuAverage() and calcuStandardDeviation(). The first method is used to interprete the textual performance values from the cases to Double values and store the values in a list. The other two methods are used to calculate the average and stand deviation of the stored values for a restoration of normal distribuation function.

The Case class has private variables including perfIndicator, performanceValues, performanceScores and performanceEvaluation. It also stores a look-up table for the probability calculation of normal distribution. It has the method calcuPerformanceScore() which calculates a performance sore between [0, 1] for each performance indicator according to the estimated rank of its value compared with the data stored in the PerformanceFunction class. Then the calcuPerformanceEvaluation() method calculates the weighted sum of all the performance indicator scores as a performance evaluation for its corresponding solution.

All the codes for this project can be found through the link:
https://nextcloud.isae.fr/index.php/s/2qdjZpaYWLpc7mS.