Overview of the 4R CBR Cycle Modifications (Extended Version)

Viktor Eisenstadt[1], Klaus-Dieter Althoff[1,2]

[1]University of Hildesheim, Institute of Computer Science
Samelsonplatz 1, 31141 Hildesheim, Germany
viktor.eisenstadt@uni-hildesheim.de

[2]German Research Center for Artificial Intelligence (DFKI)
Trippstadter Strasse 122, 67663 Kaiserslautern, Germany
klaus-dieter.althoff@dfki.de

Abstract The currently most well-known approach to apply case-based reasoning (CBR) to an application or a concept is to use the 4R cycle that consists of the steps Retrieve, Reuse, Revise, and Retain. Being a widely accepted synonym for CBR, the classic setting of the 4R cycle became a dominant basis for CBR-based systems during the last decade. However, since its introduction in 1994, multiple modifications of the original cycle were developed as well, mostly with the purpose to add additional features that might be useful in specific application situations or to provide an alternative CBR solution that might be suitable for an entire domain and related domains. In this paper, we present an overview of the most notable CBR cycle modifications. We provide a description of the main features for each selected modification, and then compare them using a number of specific criteria.
A special emphasis in this work will be put on Explainable AI (XAI) integration as one of the most recent trends in artificial intelligence. This work is an extended version of the 'Related Work' section of our ICCBR 2019 conference paper [5] that presents a flexibility-enhanced version of 4R. The main goals of this paper are to provide a researcher with a comfortable overview of the most relevant and interesting CBR cycle modifications and to discuss the future of the 4R CBR cycle.

Keywords: Case-based reasoning, Survey, CBR cycle, Modification, Retrieval, Explainable AI, Adaptation

1 Introduction

Most applications that use case-based reasoning (CBR) as their underlying structure implement the CBR's most well-known mode of operation in the form of the '4R' cycle that consists of the steps *Retrieve* (search for the most similar

cases), *Reuse* (adaptation of the solution of the most similar case to the current problem), *Revise* (check for correctness of the adaptation), and *Retain* (saving of the accepted solution together with the problem as a separate case in the case memory). The 4R cycle was introduced by Aamodt and Plaza in 1994 [1] as a result of development of the CBR research area that emerged from the idea of dynamic memory by Schank [18]. Since then, because of their versatility and adaptability of use, the 4 R-steps became a widely used approach basis for CBR-based applications and also a synonym for CBR itself. In Figure 1, the original order of execution of the 4 R-steps is shown.

The usage of the 4R cycle, however, differs from application to application: some of them use the cycle only partly (that is, a partial use of a subset of the R-steps is implemented, e.g., only Retrieve+Reuse), whereas other systems make use of all steps to achieve their goals. Furthermore, in some very specific use cases, the 4R cycle underwent structural changes to comply with requirements of its domain of use, i.e., it was structurally modified/edited to contain additional steps or to reorganize or summarize some of the original steps. This paper is intended to provide an overview of the most interesting modifications, presenting the most well-known as well as lesser known ones. Additionally, our goal is to retrace the development and use of the 4R cycle for different domains and tasks.

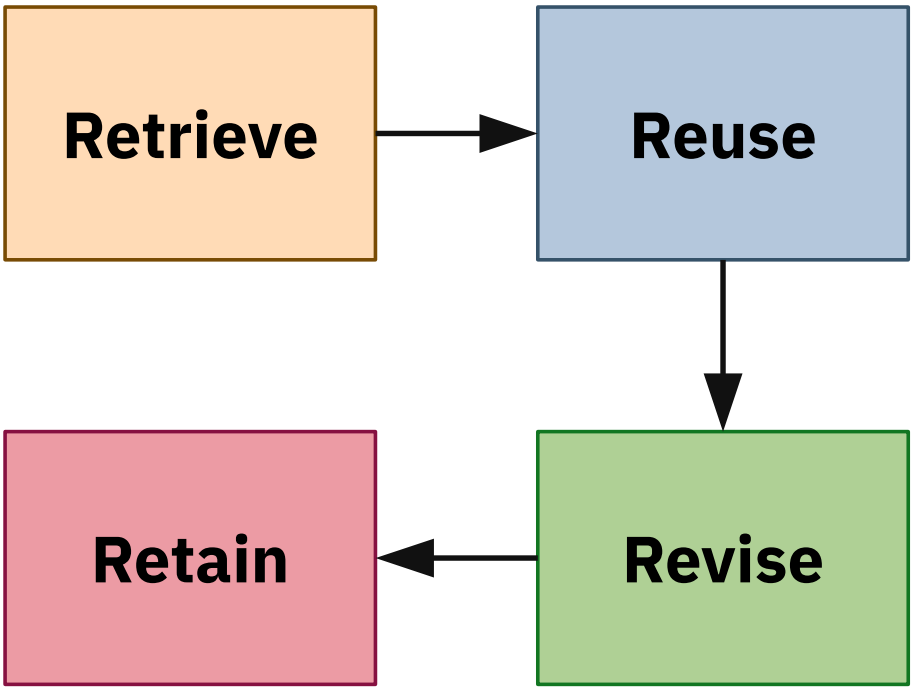

Figure 1. The original 4R CBR cycle as proposed by Aamaodt and Plaza [1].

This paper is structured as follows: first, we give a short explanation of what exactly we consider a CBR cycle modification, providing a difference between a modification and a hybrid approach. In Section 3, we provide a description of main features for each of the selected modifications and rate them with criteria, such as *modification strength*, *versatility of use*, *integration of explainability*, and *future-proveness*. In Section 4, we discuss the questions arising from the results of this work and conclude this paper.

2 What is a Modification?

To define what we will consider a modification of the CBR cycle in this paper, we decided to follow the approach we are best familiar with: the modification of objects or approaches in computer programming. For example, a modification of an algorithm or simply of the source code.

Usually, a modification of an object is defined as *inserting*, *deleting*, or *summarizing* of its properties (in our case: the steps of the 4R cycle), the same is valid for algorithms and/or the source code, where the reasonable reordering during the refactoring of the code can also be considered a modification. Based on these considerations, we defined the following criteria for the examined approaches to be counted as a 4R modification:

1. The modification, as described above, should originate from the 4R CBR cycle and *not* from the alternative CBR cycles or procedures (e.g., by Hunt [11] or by Leak [12]). The alternative cycles or procedures *do not count* as a modification as well. The modification also should not originate from another modification.
2. *The hybrid approaches will not be considered*, in order to focus the survey only on the four R-steps and reduce the scope. The hybrid approaches combine CBR with other AI methods (e.g., artificial neural networks), usually without modification of the cycle itself. However, if the situation occurs in which it will not be possible to definitely determine if the approach is a hybrid or a 4R modification, then the decision will be made in favor of modification.
3. If the approach identifies itself as a 4R modification it will be included, but with a remark that (one of) the two criteria above were not met.

3 The 4R Cycle Modifications

This section contains the selected, most notable, CBR cycle modifications. All of them were already shortly mentioned in a comparison table of modifications in the 'Related Work' section of our work on FLEA-CBR, a flexibility-enhanced extension of the 4R cycle [5]. Besides the criteria already used in that paper, that is, the versatility of use and integration of XAI, for this extended version we added two additional criteria: modification strength (*major*, *middle*, or *minor*) and an estimation of the potential of the approach for use in the future considering the current AI trends.

3.1 CBR-based Recipe Recommender IntelliMeal (Skjold and Øynes 2017)

The most recent modification approach we present in this survey is an integral part of the cooking recipe recommender *IntelliMeal* [20] developed at the Norwegian University of Science and Technology (NTNU) in 2017. This approach was developed in the context of other applications created for the Computer Cooking Contest (CCC, a regular competition of the ICCBR conference). IntelliMeal uses the CBR framework MyCBR [4] to manage the case base and its underlying data model and to conduct retrieval of similar cases, i.e., recipes with ingredients. The main goal of the approach, identically to other CCC approaches [2, 3, 9], is not to find the most similar recipe to the given one, but to create a new one by adapting the instances of the case base to the query recipe.

To achieve this goal, IntelliMeal makes use of all steps of the 4R cycle, but *does not execute them in the original order*, instead, three additional steps are inserted that break the 4R order to enhance the cycle with the following actions (see also Figure 2)[1]:

- Separation of the query into desired and undesired attributes, i.e., ingredients *(inserted before Retrieve)*.
- Setting up of an ephemeral case base after Reuse that contains already adapted cases but excludes cases with high similarity to the undesired attributes.
- Retrieval in the ephemeral case base for the following recipe revision process.

The *middle* modification approach of IntelliMeal cannot be considered versatile or universal as it is clearly bound to the cooking domain, especially the adaptation process is very domain-specific. There is also no explanation facility available. However, the system can be considered future-proof as its structure allows for conversion into, e.g., a GAN-based [7] approach where one artificial neural network can adapt the recipes and other rate them to estimate its success.

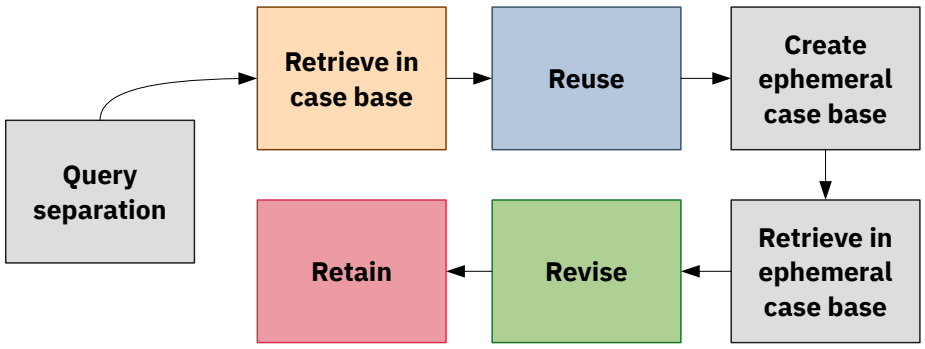

Figure 2. The 4R CBR cycle modification of IntelliMeal [20].

[1] In this and other Figures of this paper, grey boxes indicate non-original cycle steps.

3.2 Neocortex-inspired Decision Support System CHAMPION (Hohimer et al. 2011)

The framework CHAMPION [10], developed at the Pacific Northwest National Laboratory (Washington, USA), aims at providing a human neocortex-inspired *cognitive architercture* for decision support that is based on a hierarchically constructed system of intelligent agents.

The mode of operation of CHAMPION includes a modification of the classic 4R cycle, which is used as the internal mechanism of the agents of the system. The higher ranked group agents (in CHAMPION denoted as AMCs: Auto-associative Memory Columns) reason on a knowledge base unit, denoted by a sub-grah of the entire knowledge graph, in order to draw conclusions from solved problems and insert them into the base graph. ACMs can choose the knowledge unit of interest. Another group of agents works with the raw data and passes the extracted entities to the working memory of the approach from which the ACM base graph is constructed using the ontology rules.

CHAMPION significantly modifies the 4R order as it completely replaces the Retrieve phase with the Reason phase during which no retrieval in the case base takes part, instead, the corresponding agent applies description logic (DL) in order to rate the self-performed classification of the new case: if the classification was correct, the afterwards produced solution is then *retained* in the case base. After that it is analyzed in the *Review* phase, which is a new addition to the cycle as well: during this phase (and the subsequent Revise) a statistical analysis takes place whose result can improve the further classification processes. The modified 4R CBR cycle of CHAMPION is shown in Figure 3.

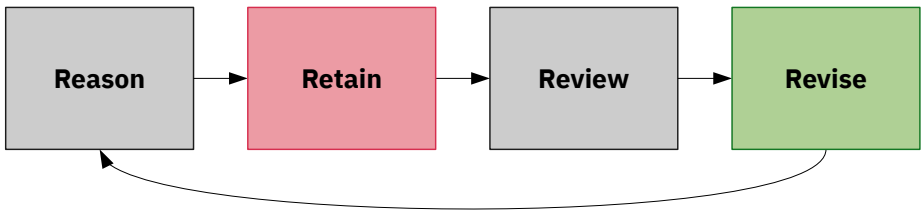

Figure 3. The modified 4R cycle implemented in CHAMPION [10].

CHAMPION is one of the few approaches presented in this work that can be considered versatile or universal. The authors of this *major* modification do not explicitly bind the system to one particular domain, therefore, taking the neocortex-inspired nature of CHAMPION into account, the approach can be considered an alternative to the currently existing cognitive architectures.

An explanation feature is not available in CHAMPION, but for the future, the approach can set a goal to be extended with a component that can provide explainable representation of the graph reasoning.

3.3 coTag: Case-based Retrieval of Similar Code Passages (Roth-Berghofer and Bahls 2008)

The system coTag [16] was developed at the Technical University of Kaiserslautern (Germany) to help software developers tag and retrieve similar source code snippets using the corresponding plugin for the integrated development environment (IDE) Eclipse. A specific property of coTag is that it is able to detect the user's, i.e., the developer's, context too, so that the most relevant code passages can be found and presented.

coTag uses the CBR cycle similarly to the traditional way, however, significant changes were applied to the steps Reuse, Revise, and Retain. The first two were replaced with the phases *Explain* and *Customise*.

Explain is responsible for scrutinizing of the retrieval results, giving the user concrete insights to the similarity assessment. Different scrutinizing types are available, e.g., an explanation of similarity between tag sets or an explanation for the trigram similarity between tags.

The phase Customise allows for customization of user-generated similarity measures, providing a possibility to correct the similarity assessment process for subsequent retrievals. These subsequent search processes are executed as part of Retain, where the corrected similarity measure is saved in the case base and then used for the next retrieval process of this context. The modification of the 4R cycle implemented in coTag is shown in Figure 4.

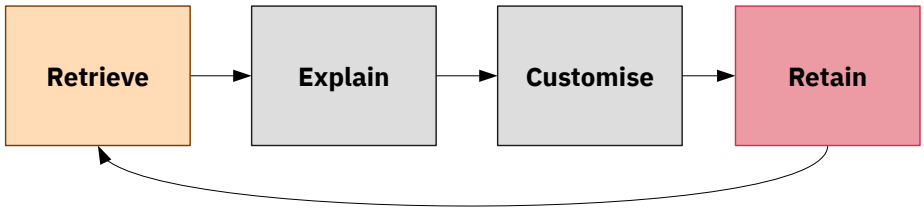

Figure 4. coTag's modification of 4R [16].

Unlike other approaches presented in this paper, coTag contains an explicit explanation feature that is used to explain the similarity assessment of the previous step Retrieve. This makes the *major* modification used in coTag unique, nevertheless, technically bound to the domain of software development.

The approach can be considered future-proof as, for example, the Retrieve and Retain phases can be replaced by a summarizing step where a recurrent neural network (RNN) suggests the next coding step in the current context.

3.4 Reorganization of Case Bases with the R5 Model (Finnie and Sun 2012)

The authors of this work [6] propose an extension of the original 4R cycle with the *Repartition* step that is responsible for the reorganization of the case base of the system in order to separate the entire case base in different, more specific case bases, based on the *similarity relation* between solutions and problems. The model was developed at the Bond University in Australia.

Before presenting their modification, the authors of R5 give a short overview of the (then) available CBR models (i.e. not only the 4R cycle but also its predecessors and contemporary alternatives), providing a foundation for their research on the topic of CBR cycle extension. The author's main claim for modification of the cycle is that the problems and solutions were not examined as separate entities and the similarities between all problems and all solutions were not used to make up a case base.

The R5 approach presented by Finnie and Sun reorganizes the original case base by examining problems and solutions separately, seeking the similarity relations between them and creating a number of separate case bases. This Repartition step is placed between Revise and Retain to save the currently solved and revised case in the correct case base. The authors also claim that this helps to improve Retrieve effectively, as the case base with the most similar cases will likely deliver the most acceptable solution to the new problem.

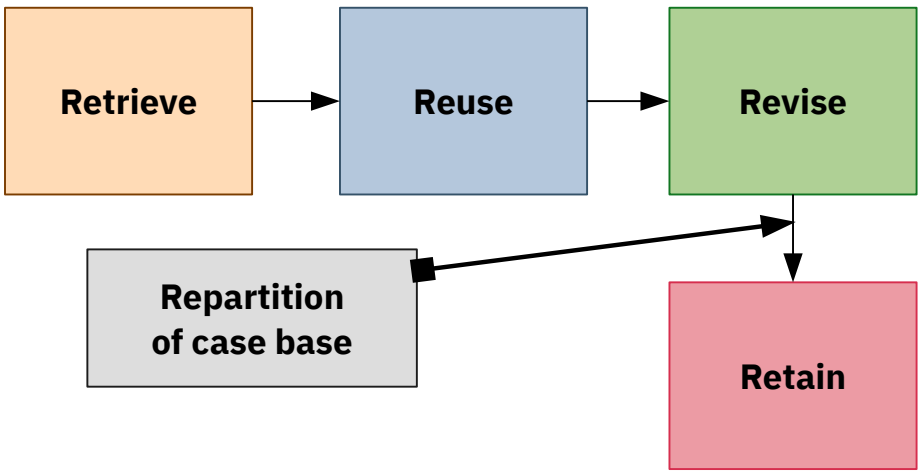

Figure 5. The R5 model for CBR [6].

The *minor* modification R5 presented in [6] does not contain any explanation feature, however, taking into account that this work is focused on the management task of the organization of case bases, it is understandable that none is provided

as the users of the system based on R5 will likely be interested in, e.g., how the similarity was calculated and not how the current database is organized. It is an universal approach as the authors do not provide any domain-specific features in the corresponding paper.

The R5 model can also be seen as future-proof, or, more exactly, independent of current trend constraints, as it provides a methodology for case base reorganization, but does not include any implementation or concrete constraints.

3.5 Q-chef: Generation of Surprising Cooking Recipes with CBR and Deep Learning (Grace et al. 2016)

The system Q-chef [8], developed at the University of North Carolina in Charlotte, employs a double-cycle structure to generate surprising cooking recipes. Similar to IntelliMeal [20] (see also Section 3.1), Q-chef is one of the many applications of CBR in the creative cooking domain. Unlike the other systems presented in this work, the approach of Q-chef is not an entirely pure CBR solution: it makes use of deep learning as one of its collaborative key components as well.

The mode of operation of Q-chef consists of execution of two subsequent cycles whose goal is to create a surprising but plausible recipe for the user. The user can provide the desired features as components and the grade of creativity. Both cycles operate on the same case base.

The grade and the components are then fed to the first cycle, in Q-chef called the *problem framing*, where the deep learning model determines the plausibility and potential of surprising for the given requirements, based on the past recipes from the case base.

In the second cycle, the *problem solving* phase takes place, where the original order of the 4R cycle is executed, including the retrieval for similar recipes in the case base and the adaptation based on the results of the first cycle. The result of the approach execution is a surprising recipe that corresponds to the criteria set by the user.

In the corresponfing work [8], the Q-chef system itself is not described as a 4R modification, and can probably also be seen as a hybrid approach. However, it can definitely count as a modification if the first cycle will be considered an additional or a pre-phase of 4R.

Q-chef's *major* modification does not provide an explanation step or component, being clearly bound to the CBR creativity in cooking approach, it is also not versatile. However, it can be considered future-proof, taking the increasing number of CBR+deep learning combination approaches during the last years. As a possible improvement, the user rating (i.e., if the final recipe was surprising or not) can be included as a feature of the deep learning model.

3.6 Case Base Maintenance Extension with Review and Restore (Roth-Berghofer and Iglezakis 2001, Roth-Berghofer 2003)

The 4R CBR cycle modification that enhances the case base maintenance was developed by Roth-Berghofer and Iglezakis [15] and in Roth-Berghofer's PhD thesis [17]. It was also further developed by Reinartz et al. [14].

Maintenance is an important phase whose task is to keep the case base free of redundant or irrelevant cases so that it contains relevant and unique cases as complete as possible. The approach turns the 4R cycle effectively into a 6R cycle, adding the phases *Review* and *Restore* to the cycle.

These two phases take care not of the case base only, but also of other knowledge containers (i.e., vocabulary, similarity measures, and adaptation rules). The work of Reinartz et al. [14] (the basis of the description of 6R in this paper), however, concentrates only on the case base maintenance. Both new phases are positioned after the Retain step. 6R is one of the older and more well-known modification approaches and had a high influence on the CBR maintenance research area.

During the Review phase the quality measurement of the cases in the case base takes place that produces a specific quality value for each available case. A monitoring service can then be used to enable specific operators that track the quality of the cases and inform the subsequent Restore component that some of them do not correspond to the quality and consistency criteria of the system.

The Restore step, receiving this information, applies a number of specific modification operators to edit the cases and/or remove them in order to keep the quality level of the cae base.

Furthermore, both new steps are part of the new *Maintenance* sub-cycle/phase (together with Retain), other steps build the *Application* sub-cycle. Figure 6 shows both phases and the steps they contain.

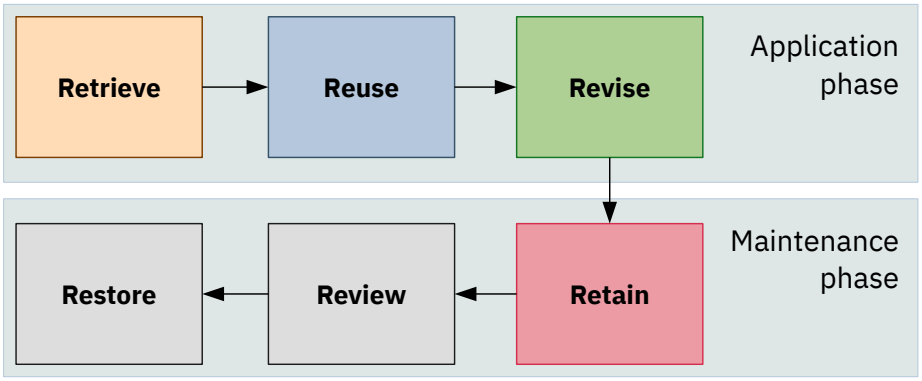

Figure 6. The 6R approach [14,15,17].

Given its nature as an approach that was developed to improve the CBR cycle in order to enhance the contemporary CBR maintenance research, 6R's *middle* modification can be considered fully universal and can be used with any domain. The explanations features, however, are neither available nor planned to be included. The future development of the approach can include the combination with deep learning for the Review phase, where the classification of the case into the quality classes (i.e., the assignment of a quality value) can be improved.

4 Discussion and Conclusion

The closing section of this paper aims at questioning the current development of case-based reasoning applications and concepts and so initiate a discussion on the following questions, taking the 4R cycle modifications described in the previous sections as an inspiration foundation:

4.1 Questions for Discussion

- Is the 4R CBR cycle still the best way to give structure to a CBR application or can a modification of it, e.g., one of the named above or a completely new one, be at least a competitor in the field of CBR resarch and development?
- Being CBR's long-time go-to approach, the 4R cycle has established itself as the basic standard of CBR, but can it hold this position? Or maybe the new AI directions (GANs, deep CNNs [19], transfer learning [13], responsible/explainable AI etc.) require a modified (modern) approach?
- If not a modification, but a completely new restructured alternative is required: which approach or direction of the current AI and/or cognitive science, besides 4R itself, should be considered highly potential to be the main influence?
- Identically to the ideas of Schank: whose or which ideas or research work should play the biggest role? Can one of the older alternatives, e.g., one of the named in Section 2, be relevant again?

4.2 Conclusion

In this paper, we presented an overview of the 4R CBR cycle modifications that can provide a foundation to discuss the questions named above. The presented modifications extend, edit, or reorder the traditional order of execution – Retrieve → Reuse → Revise → Retain – and so provide a tailored solution for a specific task or a domain or can be used universally for any domain. Most probably, the future of modifications depends on the future of CBR itself: the more new CBR approaches and application domains appear, the higher are the chances that new modifications will be developed.

References

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

2. Cordier, A., Dufour-Lussier, V., Lieber, J., Nauer, E., Badra, F., Cojan, J., Gaillard, E., Infante-Blanco, L., Molli, P., Napoli, A., et al.: Taaable: a case-based system for personalized cooking. In: Successful Case-based Reasoning Applications-2. pp. 121–162. Springer (2014)
3. DeMiguel, J., Plaza, L., Díaz-Agudo, B.: Colibricook: A cbr system for ontology-based recipe retrieval and adaptation. In: ECCBR Workshops. pp. 199–208 (2008)
4. DFKI GmbH: myCBR 3 tutorial information for ICCBR 2012 (2012), http://mycbr-project.net/downloads/myCBR_3_tutorial_slides.pdf
5. Eisenstadt, V., Langenhan, C., Althoff, K.D.: Flea-cbr – a flexible alternative to the classic 4r cycle of case-based reasoning. In: International Conference on Case-Based Reasoning (ICCBR-2019) (2019)
6. Finnie, G., Sun, Z.: R5 model for case-based reasoning. Knowledge-Based Systems 16(1), 59–65 (2003)
7. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. pp. 2672–2680 (2014)
8. Grace, K., Maher, M.L., Wilson, D.C., Najjar, N.A.: Combining cbr and deep learning to generate surprising recipe designs. In: International Conference on Case-Based Reasoning. pp. 154–169. Springer (2016)
9. Hanft, A., Ihle, N., Bach, K., Newo, R., Mänz, J.: Realising a cbr-based approach for computer cooking contest with e: Ias. In: ECCBR Workshops. pp. 249–258 (2008)
10. Hohimer, R., Greitzer, F.L., Noonan, C.F., Strasburg, J.D.: Champion: Intelligent hierarchical reasoning agents for enhanced decision support. In: STIDS. pp. 36–43 (2011)
11. Hunt, J.: Evolutionary case based design. In: Watson, I.D. (ed.) Progress in Case-Based Reasoning. Springer Berlin Heidelberg, Berlin, Heidelberg (1995)
12. Leake, D.B.: Case-Based Reasoning: Experiences, lessons and future directions. MIT press (1996)
13. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering 22(10), 1345–1359 (2009)
14. Reinartz, T., Iglezakis, I., Roth-Berghofer, T.: Review and restore for case-base maintenance. Computational Intelligence 17(2), 214–234 (2001)
15. Roth-Berghofer, T., Iglezakis, I.: Six steps in case–based reasoning: Towards a maintenance methodology for case–based reasoning systems. In: Proceedings of the 9th German Workshop on Case-Based Reasoning. pp. 198–208 (2001)
16. Roth-Berghofer, T.R., Bahls, D.: Code tagging and similarity-based retrieval with mycbr. In: International Conference on Innovative Techniques and Applications of Artificial Intelligence. pp. 19–32. Springer (2008)
17. Roth-Berghofer, T.: Knowledge Maintenance of Case-based Reasoning Systems: The SIAM Methodology. Dissertationen zur künstlichen Intelligenz, Aka - Akademische Verlagsgesellschaft (2003)
18. Schank, R.C.: Dynamic memory: A theory of reminding and learning in computers and people, vol. 240. Cambridge University Press Cambridge (1982)
19. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
20. Skjold, K., Øynes, M.S.: Case-Based Reasoning and Computational Creativity in a Recipe Recommender System. Master's thesis, NTNU (2017)