A Novel Component of Rule Generation in Ubiquitous Computing Environments

Roua Jabla[1,2], Maha Khemaja[3], Félix Buendía[2], Sami Faiz[4]

[1] University of Sousse, ISITCom, 4011, Sousse, Tunisia

jabla.roua@gmail.com

[2] Universitat Politècnica Valencia, 46022, Valencia, Spain
[3] University of Sousse, 4000, Sousse, Tunisia
[4] University of Tunis El Manar, 5020, El Manar, Tunisia

Abstract. The proliferation of ubiquitous computing with mobile devices makes context models and information extremely rich and changing on account of context-aware environment changes over runtime. However, defining rules at design-time may impair their efficiency and the decision-making process. Therefore, it is important to address decision-making problems leveraged by dynamic environments and context models evolution. In this sense, a solution that could emerge is the continuous rule knowledge base evolution at runtime. In this paper, we propose a decision adaptation component to deal with the generation of rules to improve decision-making due to changes occurring around users at runtime. A case study is conducted to illustrate the implementation of the proposed component for the rule knowledge base enrichment and the decision-making improvement.

Keywords: Ubiquitous computing, Context-awareness, Rule generation, Machine Learning.

1 Introduction

Ubiquitous computing is the upward trend in computer technology, in which computation occurs anywhere and everywhere using any device [1]. All current technology may switch to ubiquitous computing environments in the coming years since ubiquitous computing integrates new types of computing such as mobile computing, pervasive computing, context-awareness and adaptation while hiding the technology and making it meld into our daily lives. In the era of ubiquitous computing, a mobile device is the most popular ubiquitous computing device, thanks to the large number of sensors that it could incorporate [2]. In light of moving from a single context to multi-contexts with the use of mobile devices and the exploitation of their advanced sensing capabilities, it is pertinent to capture the dynamic nature of the users' surrounding environment changes and to infer relevant knowledge, i.e., rules.

Due to the rapid development of ubiquitous computing and context-aware technology in mobile devices, ubiquitous computing is leading the advent of mobile

device-oriented context-aware applications because they seek to be everywhere. In existing mobile device-oriented context-aware applications, rules can only behave to changes in environment attributes and context information [3]. On that account, these rules can answer neither dynamic environment nor context model evolution at runtime [3]. This means that rules cannot be static and need to be constantly evolved to remain relevant [4]. Therefore, an important challenge is raised related to the potential of context-aware applications to automatically provide appropriate rules and adaptations to timely react to arisen changes at runtime. To tackle this challenge, supporting middleware of such a context-aware application should carry out dynamic changes in the application behavior at runtime through the evolution of its context model and subsequently its rule knowledge base with new rules to improve decision-making in dynamic and changing environments. Hence, a considerable need for a decision-making process, which aims to continuously enrich a rule knowledge base over time, has emerged to make applications more resilient to dynamic environments as well as to context model evolution at runtime.

To fill this need, we propose a decision adaptation component that augments an existing distributed middleware that enables context-aware applications to support dynamic pervasive computing environments. The main feature of this component is to offer context-aware applications, where their rule knowledge bases are fluid and evolutive. The novelty and contribution of this paper could be drawn from three-fold. First, we provide a component for the purpose of evolving a rule knowledge base following a dynamic ontology-based context model evolution to work in dynamic environments. Second, we propose a novel hybrid learning approach towards effectively generating a complete set of rules, in automated fashion, by adopting two supervised learning algorithms to further enhance the rule generation efficiency. Third, we extend a Genetic Algorithm (GA) [5] with a multi analysis technique targeting the rule optimization. Then, we propose as a case study the application that is intended to assist engineers in the rule generation process at runtime.

The rest of this paper is organized as follows. Section 2 discusses related works about rule generation. In section 3, a detailed presentation of the decision adaptation component is presented. In section 4, a case study and an example of rule generation are described. Finally, Section 5 outlines the conclusions and future work.

2 Related work

Rule generation is one of most popular topics in data mining that allows the efficiency of context-aware applications adaptation according to context and environment changes. Recently, this topic has been extensively studied in the literature because of its wide applicability and usefulness. By going through the literature, some approaches have supported rule generation to target ontology enrichment while certain other approaches have targeted the decision-making improvement. In the following, we provide a brief overview of the most recent approaches on this topic.

A couple of approaches for rule generation have been proposed in the scope of ontology enrichment process. In this sense, Paiva et al. [6] introduced a semi-

automatic approach that focuses on enriching an ontology with relations between concepts using association rules. In this approach, they used the FPGrowth algorithm [7] to discover frequent items from unstructured data sources and then generate a set of association rules. After that, they exploited generated association rules for learning useful relations in the ontological model. Another work considered ontology enrichment based on learning association rules is the work of Idoudi et al. [8]. This work described a new approach for ontology content evolution through incorporating knowledge derived from medical records. For this purpose, Apriori algorithm [9] is applied to generate association rules. Next, domain experts are invited to validate generated rules. Then, validated rules are used in the enrichment process of the knowledge base with new association between existing concepts. Later, the authors in [10] presented a work, in which an association rule mining algorithm is employed to find recurrent patterns and discover association rules. The discovered rules can be automatically exploited to enrich an existing ontology with formal definitions of concepts.

Apart from these approaches, novel approaches, in the scope of decision-making improvement, are emerging. For example, Gabroveanu et al. [11] proposed a recommender system in the field of distance learning in higher education. They used data obtained from learning management systems database and Apriori algorithm in order to identify association rules related to courses followed by students. Kaliappan et al. [12] proposed a new modified Apriori algorithm for finding the association rules among large datasets to promote sales and user interaction. They showed that the proposed algorithm improved the efficiency of generating association rules. Davagdorj and Ryu [13] offered an association rule mining method to discover useful patterns, which include medical knowledge, from a medical dataset. They applied the FP-Growth algorithm to extract a set of association rules. Then, the obtained rules are used to support medical decision-making for interpreting diagnosing patient information. Recently, Asadianfam et al. [14] provided a new approach to improve recommendations in recommender systems. One of the challenges considered in this approach is that it could not provide appropriate recommendations to users who have different profiles from the existing users' profiles. To deal with this challenge, authors used Apriori algorithm to generate association rules from users' behaviors and then made appropriate recommendations. They showed that the generated association rules could increase the overall efficiency of the recommender system.

However, the above approaches focus on extracting rules directly from data sources using traditional algorithms, such as FPGrowth and Apriori. This can lead to a huge number of generated rules due to the narrow applicability of these algorithms [15]. Furthermore, there are few approaches that consider rule generation at design-time (i.e., offline). In this case, a context-aware application cannot not stand the dynamic environments at runtime. A further limitation is that a human intervention is needed in certain approaches to validate and manage generated rules. None of the above-mentioned approaches support the rule optimization with a target of getting interesting rules.

Along the similar line, this paper aims, firstly, to derive interpretable rules from training models of supervised machine learning algorithms, secondly, to retrieve interesting rules using an extension of a Genetic Algorithm, which can be applied in a straightforward manner for rule optimization.

3 Decision adaptation component

In this section, a global approach that aims to extend an existing middleware with a new composite component, is presented. The overall objective of the global approach is to support dynamic context evolution and the adaptation of the decision-making process at runtime without developer's intervention or system's disruption. This objective led to the proposal of a composite middleware component based on a four-component as depicted in Fig.1.

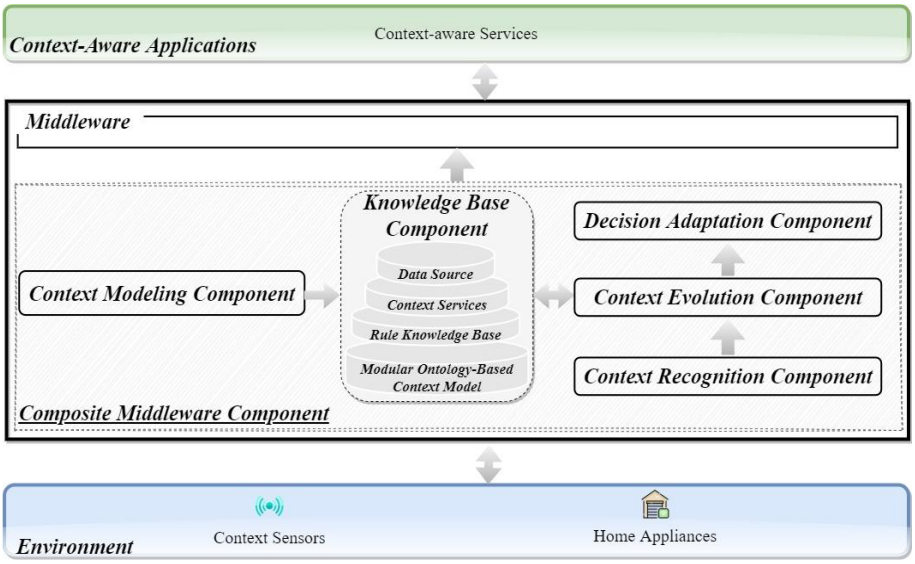

Fig. 1. Global approach architecture overview.

Within this composite middleware component, only the decision adaptation component, which deals with the evolution of a rule knowledge base by generating appropriate rules to go along with dynamic environments and the context model evolution at runtime, is considered in this paper. The proposed component is designed to generate well-performed rules needed to automatically cover dynamic environments that change frequently at runtime. This component runs only once a context model evolution is achieved and a priori rules are deemed not to be relevant anymore for the new context model. The overall architecture of the component is illustrated in Fig. 2.

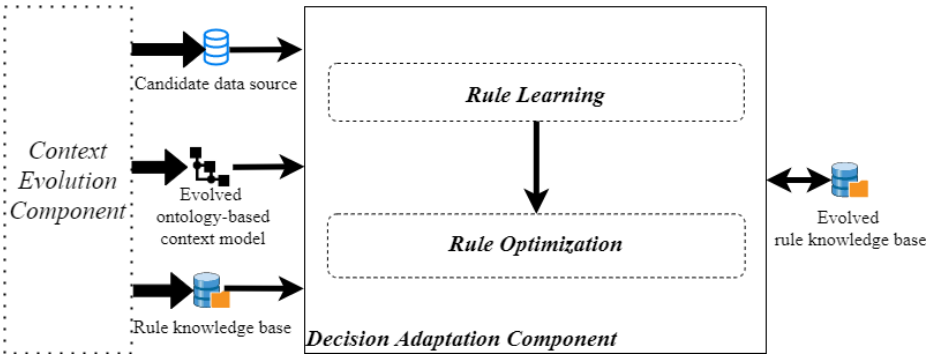

Fig. 2. Decision adaptation component architecture.

As illustrated in Fig. 2, the decision adaptation component includes two key modules, namely "rule learning" and "rule optimization". In the following subsections, we introduce these modules in more detail.

3.1 Rule learning

Rule learning module is responsible for deriving rules of IF-THEN statements from the candidate data source previously considered in the context model evolution process. Fig. 3 highlights three key phases leading, ultimately, to the generation of IF-THEN rules.

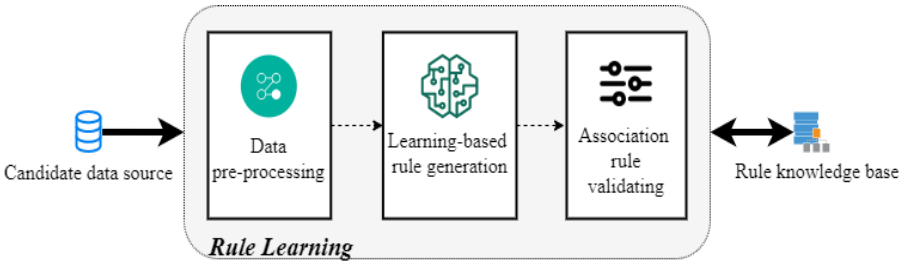

Fig. 3. Rule learning module.

In the first phase, called data pre-processing, the candidate data source is pre-processed with the purpose of improving its quality to ensure the generation of consistent rules. Hence, two major steps are involved:

- Data cleaning that attempts to fill missing values and smooth out noise in the candidate data source. For filling missing values, we replace identified missing values with the average of existing values. And for smoothing noisy data, we carry out techniques, such as clustering, regression and binning, to eliminate noisy data.

- Data reduction that is used to simplify the data source representation without any loss of useful information. To reach it in the easiest manner, we remove redundant and inconsistent data.

In the second phase, called learning-base rule generation, the candidate data source is processed to uncover relationships between seemingly unrelated elements, in a tree structure. This step is carried out through two steps as in Fig. 4. First, the hybrid supervised learning step is applied to train the candidate data source targeting the creation of tree-structured training models. To enhance the learning performance, we explore the idea of hybridizing machine learning algorithms with the use of two supervised machine learning algorithms, Decision Tree [16] and Random Tree [17]. The resulting training models consist of a set of decisions in a tree structure, which could be utilized to generate rules from each leaf node [18]. Then, the rule extraction step is performed to automatically extract rules as IF-THEN statements. Each rule has two parts, an antecedent (IF) and a consequent (THEN) that are corelated to each other. An antecedent can be constituted by a set of atoms while a consequent contains only one atom.

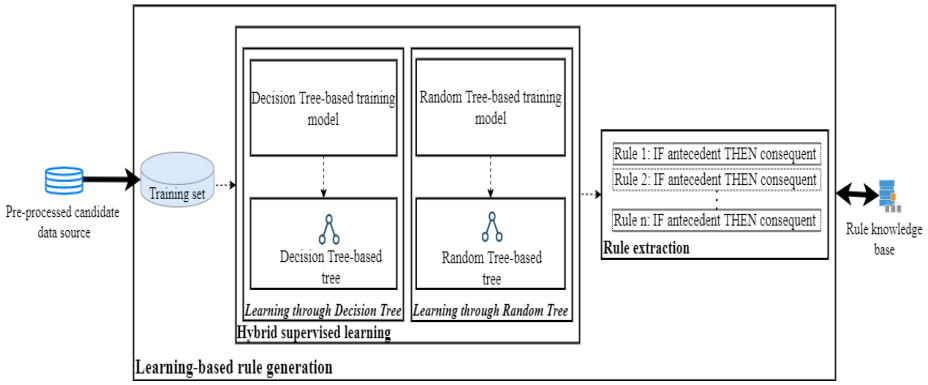

Fig. 4. Learning-based rule generation phase.

In the third phase, called rule validating, the following methods are applied:

- Rule structure verification, which deals with verifying that rules are preserving the inherent IF-THEN structure.
- Rule consistency verification, which has to check the consistency of rules and enumerate all inconsistent rules, where the consequent part does not refer to the antecedent part.

3.2 Rule optimization

The rule optimization module is used to identify the well-performed rules to avoid many relevant and irrelevant rules from the set of earlier validated rules. To this end, GA is extended to support a multi-analysis technique in order to jointly optimize

rules. Fig. 5 illustrates the two main phases in the rule optimization module. The first phase is aimed to select interesting rules and the second phase is used to re-optimize the interesting rules selected in the first phase. The two phases are briefly summarized in Algorithm 1.

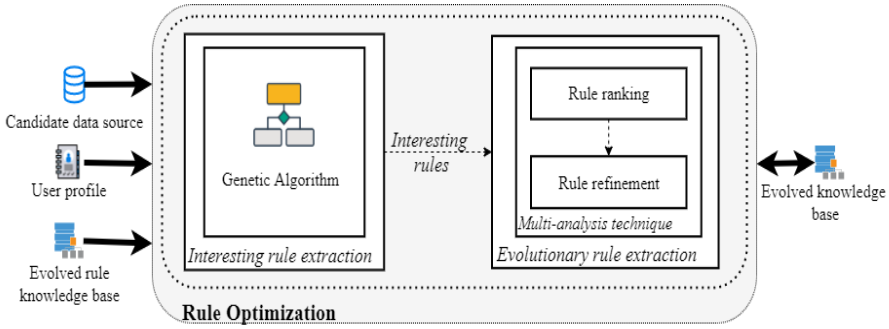

Fig. 5. Rule optimization layer.

According to Fig. 5 and Algorithm 1, the first phase, namely the interesting rule extraction, identifies interesting rules on the basis of the candidate data source using GA. This algorithm starts with the construction of an initial population as a collection of chromosomes and lets them evolve over multiple generations to reach more and more interesting rules. In this case, each chromosome represents a candidate rule. Next, the applied GA measures the performance of each chromosome using the fitness function to find out rules that have a support value above a certain threshold value. Then, it proceeds through three genetic operators, including, selection, crossover and mutation, to evolve a new generation and determine again the fitness function of each chromosome. At the end, it stops when it converges to an optimal set of interesting rules that could satisfy the fitness function.

The second phase, namely the evolutionary rule extraction, proposes a multi-analysis technique to retrieve the well-performed rules from the obtained interesting rules. On the grounds of this, we propose rule ranking and refinement steps as shown in Fig. 5. The multi-analysis technique starts with the rule ranking step that is in charge of automatically ranking rules derived from supervised machine learning algorithms and the interesting rules regarding the frequency of occurrence and the fitness function weight. The rule occurrence frequency is considered as the support degree to classify rules, followed by the fitness function weight. Then, the rule refinement step is performed to retrieve the set of well-performed rules with better accuracy performance on foundation of the ranked rules in order to enhance the decision-making accuracy. This step begins with finding a user who is related to runtime changes that occurred in the dynamic environment and loading the user profile of the corresponding user. The user profile is integrated in the refinement step to infer the well-performed rules.

Algorithm 1. *Rule optimization*

```
    Input: DS pre-processed candidate data source
           FT fitness threshold value
    Output: Well-performed IF-THEN rules
1   Begin
2   Initialize population
3   Repeat
4      Fitness evaluation
5      Selection
6      Crossover
7      Mutation
8   Until fitness of new population > FT
9   For each interesting rule
10     Calculate occurrence frequency
11     If (! existFrequency(frequency))
12       Check fitness function weight
13     Else
14       Calculate rank
15     End if
16  End for
17  For each ranked rule
18     If (! isRelatedToUserContext(ranked rule))
19       Delete rule from the rule knowledge base
20     End if
21  End for
22  End
```

4 Case study

In this section, we present an example of a case study that allows us to illustrate the above-described component through providing the generation of rules from data sources. For this, we consider an application for assisting engineers to evolve a rule knowledge base regarding an ontology-based context model evolution due to arisen changes in the surrounding environment at runtime. The presented application consists of two distinct parts, called frontend and backend. The backend part, which deals with the automatic rule generation, is implemented as REST web services [19] in Java. The frontend is created with Angular to deal with engineers' interactions and interfaces with the backend.

For the present case study, we chose the weather data source [17], containing four condition attributes, such as outlook, temperature, humidity, windy, and one decision attribute 'play'. As noted previously, the candidate data source is already used for the automatic ontology-based context model evolution at runtime. In contrast to previous use, we consider the candidate data source to generate rules that help in making decisions regarding whether a user could go outside for playing or not.

After the data pre-processing step, a hybrid supervised learning is performed on the candidate data source for creating tree-structured training models through the

implementation of Decision Tree and Random Tree algorithms as depicted in Fig. 6 and 7, respectively.

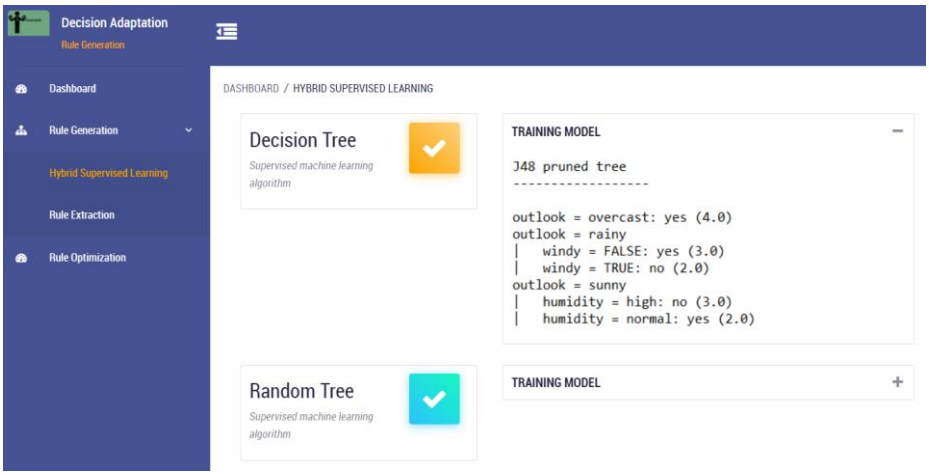

Fig. 6. Tree-structured training model of the Decision Tree algorithm.

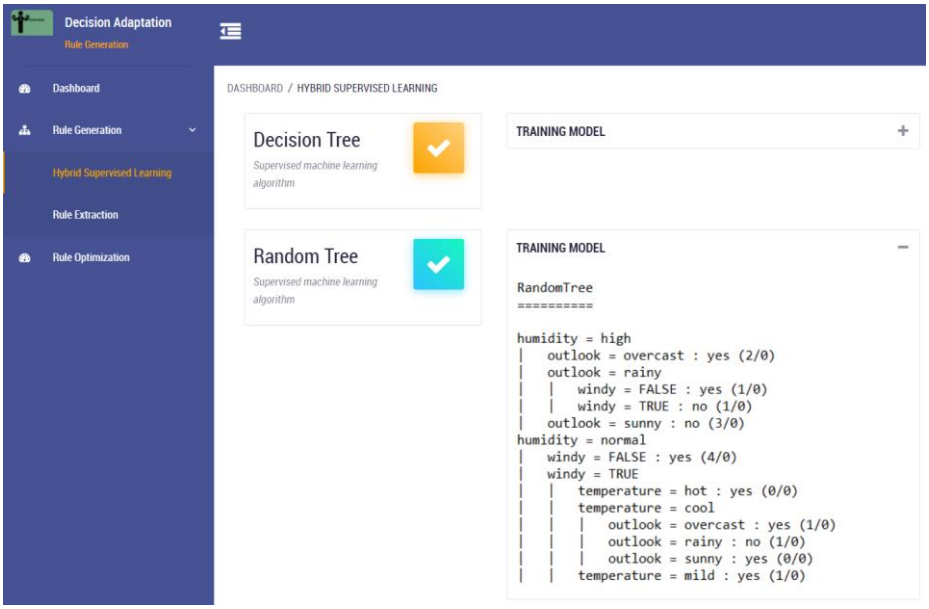

Fig. 7. Tree-structured training model of the Random Tree algorithm.

Next, a rule extraction is applied for automatically analyzing the obtained trees and extracting IF-THEN rules that are thoroughly validated via the different validation

methods. Fig. 8 and 9 present the validated IF-THEN rules from the Decision Tree training model and Random Tree training model, respectively.

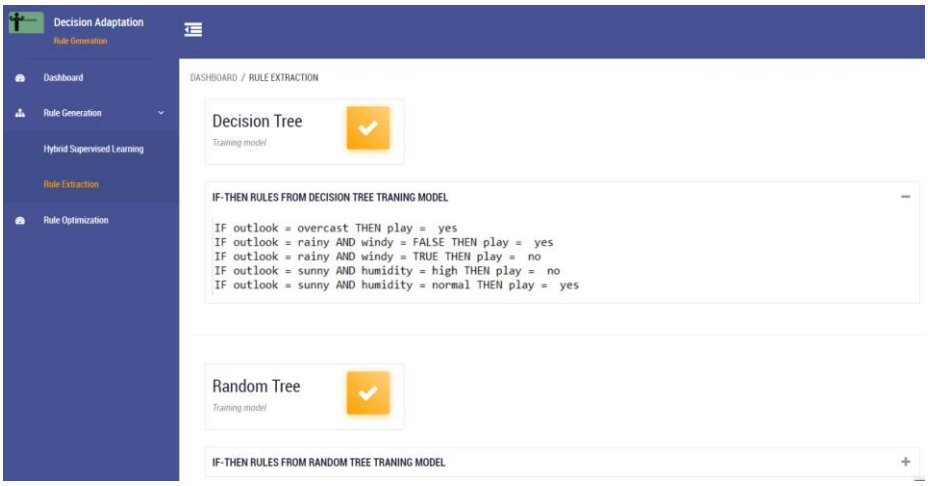

Fig. 8. Validated IF-THEN rules from the Decision Tree-based training model.

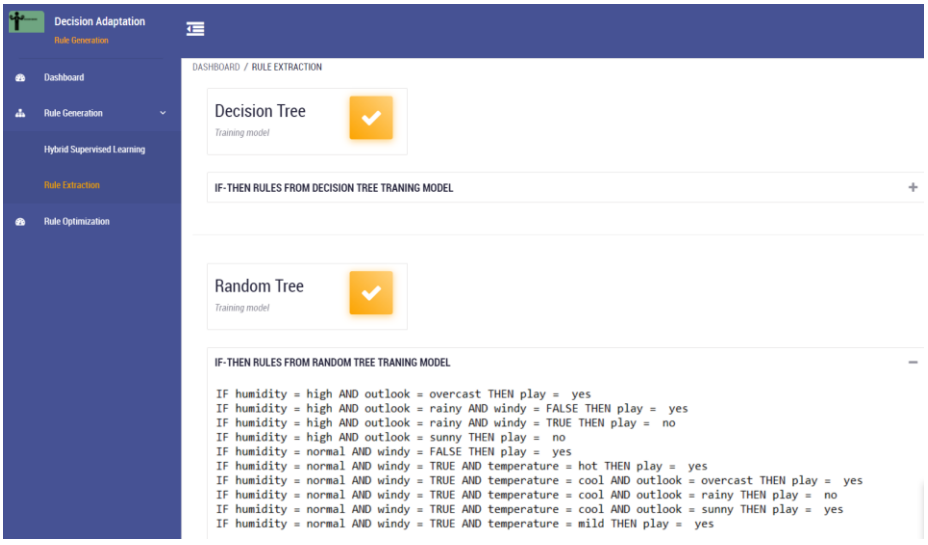

Fig. 9. Validated IF-THEN rules from the Random Tree-based training model.

Then, the GA is employed to find out the interesting rules. Afterward, the multi-analysis technique is performed to retrieve the well-performed rules. Fig. 10 illustrates an excerpt of the well-performed rules.

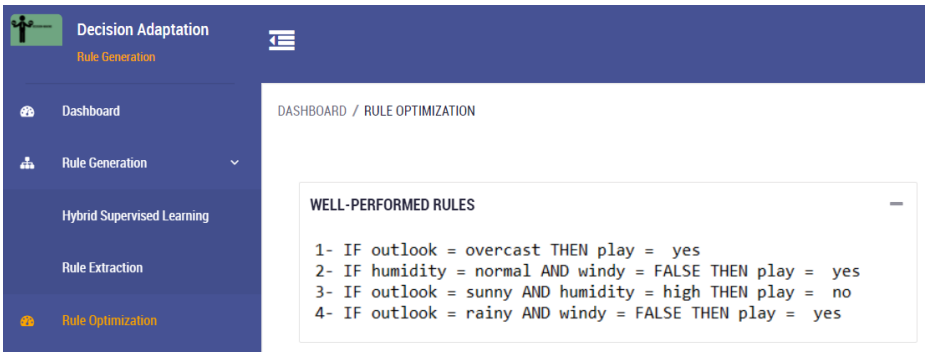

Fig. 10. An excerpt of the well-performed rules.

5 Conclusion

In this paper, we proposed a decision adaptation component that aims to generate rules without engineers' interventions in reaction to dynamic ubiquitous computing environments at runtime. This component first opened with a hybrid supervised learning and then a GA with multi-analysis technique is applied to support rule optimization. Furthermore, we briefly presented a case study-based application for engineers targeting first the automatic IF-THEN rule generation from a data source and then the rule optimization to improve decision-making at runtime. However, it is still a long way to go for using our component in a real-life scenario since there exist some limitations in terms of rule applicability.

For the future, we plan to extend our component with the possibility to automatically transform the IF-THEN rules to rules in the syntax of Jena since we are adopting ontology-based context models that are managed within a Jena based platform equipped with an embedded reasoner. Moreover, we intend to evaluate our component to provide an insight into the potential of our component in improving decision-making in context-aware applications regarding dynamic ubiquitous computing environments and context models evolution.

References

1. Kumar, R., & Paiva, S. (Eds.). (2021). Applications in Ubiquitous Computing. Springer.
2. Garrido, P. C., Ruiz, I. L., & Gómez-Nieto, M. Á. (2014). OBCAS: an agent-based system and ontology for mobile context aware interactions. Journal of Intelligent Information Systems, 43(1), 33-57.
3. Zhao, T. (2016, September). The generation and evolution of adaptation rules in requirements driven self-adaptive systems. In 2016 IEEE 24th International Requirements Engineering Conference (RE) (pp. 456-461). IEEE.

4. Liu, Y., Zhang, W., & Jiao, W. (2016, September). A generative genetic algorithm for evolving adaptation rules of software systems. In Proceedings of the 8th Asia-Pacific Symposium on Internetware (pp. 103-107).
5. Goldberg, D. E. (1989). Genetic algorithms in search. Optimization, and Machine Learning.
6. Paiva, L., Costa, R., Figueiras, P., & Lima, C. (2014, June). Discovering semantic relations from unstructured data for ontology enrichment: Association rules-based approach. In 2014 9th Iberian Conference on Information Systems and Technologies (CISTI) (pp. 1 6). IEEE.
7. Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. ACM sigmod record, 29(2), 1 12.
8. Idoudi, R., Ettabaa, K. S., Solaiman, B., & Mnif, N. (2016). Association rules-based ontology enrichment. International journal of web applications, 8(1), 16-25.
9. Agrawal, R., & Srikant, R. (1994, September). Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB (Vol. 1215, pp. 487-499).
10. Chang, M., D'Aniello, G., Gaeta, M., Orciuoli, F., Sampson, D., & Simonelli, C. (2020). Building ontology-driven tutoring models for intelligent tutoring systems using data mining. IEEE Access, 8, 48151-48162.
11. Gabroveanu, M., & Diaconescu, I. M. (2008, July). Extracting semantic annotations from Moodle data. In Proceedings of the 2nd East European Workshop on Rule-Based Applications (RuleApps 2008) at the 18th European Conference on Artificial Intelligence (ECAI 2008) (pp. 1-5).
12. Kaliappan, J., & Sai, S. M. (2019). Weblog and retail industries analysis using a robust modified Apriori algorithm.
13. Davagdorj, K., & Ryu, K. H. (2018). Association Rule Mining on Head and Neck Squamous Cell Carcinoma Cancer using FP Growth algorithm.
14. Asadianfam, S., Kolivand, H., & Asadianfam, S. (2020). A new approach for web usage mining using case-based reasoning. SN Applied Sciences, 2(7), 1-11.
15. Mahmood, A., Shi, K., Khatoon, S., & Xiao, M. (2013). Data mining techniques for wireless sensor networks: A survey. International Journal of Distributed Sensor Networks, 9(7), 406316.
16. Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1(1), 81-106.
17. Witten, I. H., & Frank, E. (2002). Data mining: practical machine learning tools and techniques with Java implementations. Acm Sigmod Record, 31(1), 76-77.
18. Suresh, A., Udendhran, R., & Balamurgan, M. (2020). Hybridized neural network and decision tree-based classifier for prognostic decision making in breast cancers. Soft Computing, 24(11), 7947-7953.
19. Fielding, R. T., & Taylor, R. N. (2002). Principled design of the modern web architecture. ACM Transactions on Internet Technology (TOIT), 2(2), 115 150.