Missing Information Prediction in Ripple Down Rule Based Clinical Decision Support System

Musarrat Hussain[1], Anees Ul Hassan[1], Muhammad Sadiq[1], Byeong Ho Kang[2], and Sungyoung Lee[1](✉)

[1] Ubiquitous Computing Lab, Department of Computer Science and Engineering, Kyung Hee University, Seoul, South Korea
{musarrat.hussain,anees,sadiq,sylee}@oslab.khu.ac.kr
[2] School of Engineering and ICT, University of Tasmania, Hobart, Australia
byeong.kang@utas.edu.au

Abstract. Clinical Decision Support System (CDSS) plays an indispensable role in decision making and solving complex problems in the medical domain. However, CDSS expects complete information to deliver an appropriate recommendation. In real scenarios, the user may not be able to provide complete information while interacting with CDSS. Therefore, the CDSS may fail to deliver accurate recommendations. The system needs to predict and complete missing information for generating appropriate recommendations. In this research, we extended Ripple Down Rules (RDR) methodology that identifies the missing information in terms of key facts by analyzing similar previous patient cases. Based on identified similar cases, the system requests the user about the existence of missing facts. According to the user's response, the system resumes current case and infers the most appropriate recommendation. Alternatively, the system generates an initial recommendation based on provided partial information.

Keywords: Information prediction · Ripple down rules
Clinical decision support system

1 Introduction

1.1 Background

Clinical Decision Support System (CDSS) are extensively used in healthcare to provide domain-specific decisions about the patient at the required time. It has an imperative role in improving patient care and reducing the chances of error in decision making by assisting physicians in the decision [1]. Domain experts' knowledge is transformed into a computer interpretable format, which is utilized by the system for taking intelligent decisions. Most of existing CDSS [2, 3] model experts' knowledge in if-then-else rules format which is an effective method of knowledge representation.

Existing CDSS generate recommendations based on input symptoms by reasoning over the available rules in the knowledge base [4, 5]. System's decision rely relies on users' provided information and complete information results in the appropriate recommendation. However, users may not be able to provide complete information while

M. Mokhtari et al. (Eds.): ICOST 2018, LNCS 10898, pp. 179–188, 2018.
https://doi.org/10.1007/978-3-319-94523-1_16

interacting with the system. The user may consider some information as irrelevant while that may play an imperative role in decision making. Therefore, the system needs to be smart enough to facilitate users for providing complete information. In order to provide a most appropriate decision, the system should predict missing information and ask the user to provide that information. Alternatively, the system generates an initial recommendation based on provided information.

Apart from the user input, knowledge base also has a pivotal role in recommendation generation. The systems' intelligence usually depends on the knowledge evolution. The knowledge evolves with the passage of time because of new research and findings. The new knowledge needs to be adopted by CDSS to generate right decisions. Most of the existing systems are one-shot systems and uses machine learning approaches for generating knowledge base. The primary drawback of these systems is this that a human expert cannot update the knowledge base. For knowledge evolution, one-shot systems need to rebuild the model with a complete cycle of verification and validation process. Furthermore, it cannot guarantee preservance of model accuracy.

In this study, we used Ripple Down Rules (RDR) for knowledge storing and as a validation methodology, which stores experts' knowledge in a form of tree structure consists of nodes and branches [6, 7]. Each node in RDR tree stores a complete if-then rule and represents a particular classification class. While branches of the tree represent parent-child relations in the RDR tree. The sequence of rules (nodes) matched with a case in RDR tree is known as path of the case. RDR model stores experts' knowledge in comprehensible format (tree-structured), understandable by machine and human experts [7]. Human experts can add their knowledge to the system without the involvement of knowledge engineers. RDR adapts new knowledge by extending existing knowledge (add a new child or new branch to the RDR tree) without affecting the previous one which preserves the previous accuracy. It works as a real-time verification and validation process. All the knowledge is verified at the time of updating the knowledge base. RDR uses forward chaining inference mechanism that matches the current case/input facts with the root node and traverse child nodes of all matched rules at each level until no match is found. Finally, the last matched rule result is presented to the user as a recommendation. RDR has many benefits as compared to other models including real-time knowledge validation and verification. However, it also completely relies on user input. If a user misses some of the key information during interaction with the system, it may generate an inappropriate recommendation.

1.2 Proposed System Overview

The proposed system extends the RDR mechanism to facilitate users for providing complete information by prediction of missing information in the user's input. Extended RDR identifies and predict missing information by analyzing similar previous patient cases. The system compared facts and attributes of the similar previos patient cases with the current case and find the missing facts in the input case. The system asks questions for identified missing information from the user to get the final decision. However, if the user is unable to provide the system identified missing information the system generates an initial recommendation for provided partial information. The complete detail of the proposed mechanism and its benefit will be described in the upcoming sections.

The rest of the paper is structured as follows. In Sect. 2, we describe the existing work related to Question/Answer based RDR systems. In Sect. 3, we describe the proposed methodology. Section 4 describes a case study. Section 5 describes results and Sect. 6 concludes the study.

2 Related Work

The history of CDSS begins from 1960 and have made colossal progression with the ascent of innovation [8]. It still needs to fulfill day-to-day requirements and expectations. The proposed approach is motivated by conversational systems; therefore, the related work described below is with the domain of general-purpose conversational system rather than CDSS. The commonalities and differences of proposed technique and techniques of existing systems are depicted in Table 1.

E.M. Glina et al. [9] proposed a conversational system, which constructs an arbitrary search tree from parts of the active database and searches node incrementally based on user interaction with the system. The last node returns the result of each step as the conversation response and it logged the complete path followed by the conversation. On users' new request, the system matches current node's children, if no child matched, the system traverses the tree in the backward direction and search in other child nodes. If the system is not able to answer a query, it records that query and later asks domain expert to provide a response. In this way, the system evolves the knowledge base by domain experts' continuous input. The primary drawback of this system is, it totally relies on users' input. The system has no capability to predict the missed information during the user interaction.

Table 1. Comparison of the proposed approach with existing approaches.

Approaches	Interactive	Path tracking	Case preservance	Predict missing information
E.M. Glina	No	Yes	Yes	No
D.Q. Nguyen (KBQAS)	Yes	No	No	No
Q.T. Gia (IRDR)	Yes	No	No	No
Proposed	Yes	Yes	Yes	Yes

D.Q. Nguyen et al. [10] proposed a question answering system named KBQAS having answer retrieval mechanism for asked questions, which are mapped with ontological rules. If the system does not match a concept, then it provides the extracted concept to the user for its clarification. After clarification of concepts, the system maps those concepts with the question format to get an appropriate answer. The limitation of this system is, it processes each question in isolation and extracts concepts from a single question, which may not be beneficial and may lose contextual information. Therefore, the system should consider, all other related questions to extract related concepts for high accuracy in rules mapping to give an appropriate answer.

Q.T. Gia [11] proposed an Interactive RDR (IRDR) that enhanced the RDR capabilities by asking questions for more information and the users each response is considered as a new case. Existing IRDR simulation studies have been done, however, due to the poor results of the technique it has not been published. The main reason for poor results of IRDR is the lack of case status preservation. In each iteration, the system considers the input case as a new case, which increases the computational cost and gives answers with low accuracy.

Mostly the existing systems focus on accurate mapping of current case with the rules of the knowledge bases. These existing systems except IRDR are considered as a one-shot learning and execution systems. One shot system needs to be trained on huge data before using. In execution, the user query from the system and the system responses accordingly. These systems have limited capability to deal with the input queries having incomplete information. If the users miss some information during an interaction, the existing systems are not able to detect the missing information and to generate new questions on the fly to ask the user. Therefore, the existing systems generate results based on provided information, which may cause produced poor results.

3 Proposed Methodology

This work is the part of our current ongoing project so-called Intelligent Medical Platform (IMP)[1]. IMP aims to utilize artificial intelligence and Big Data technologies for providing proactive and predictive healthcare services to improve public health. In this research, we focused on two major issues of the existing CDSS and recommender systems. First, the proposed system predicts missing information by analyzing similar cases and ask users to provide them for delivering most appropriate recommendation. Secondly, it preserves the current status of the user case, and resumes it on new information to generate a recommendation in the appropriate context and reduces the recommendation generation time. To achieve the aforementioned goals, the proposed system is conceived with three major components: *Case Analyzer*, *Question Moderator*, and *Frequent Path Miner* as depicted in Fig. 1. The system has two repositories *RDR Rules Base* and *Frequent Paths* repository. The details of these components are described in following subsections.

[1] http://imprc.cafe24.com/.

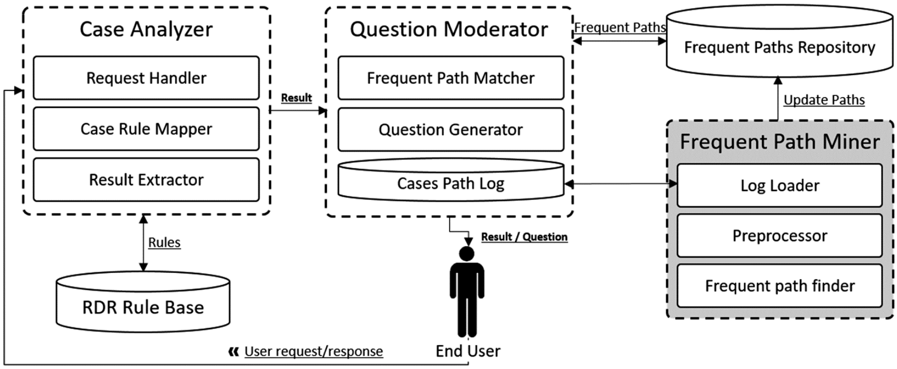

Fig. 1. Proposed system architecture

3.1 Case Analyzer

Case Analyzer is devised for effective manipulation of user request considered as a new case. It consists of three subcomponents. The subcomponent *Request Handler* formulates the case and extracts available facts from the case. *Case Rule Mapper* module starts mapping extracted facts from the root node of RDR tree with the rule condition at each node. If rule condition is matched with the case facts, then it checks the child nodes of that node until no further rule matched. The *Result Extractor* extracts last matched rule result, considered as candidate recommendation. The candidate recommendation of the matched rule along with the case information is sent to *Question Moderator* to find missing information in the current case.

3.2 Question Moderator

Question Moderator is needed to increase the system accuracy by considering similar cases to generate a final recommendation. *Question Moderator* matches the current case path followed in RDR tree with similar cases path known as *Frequent Path*. The objective of this path matching is to find missing information based on a most used path from *Frequent Path Repository* and to generate an appropriate recommendation. If the frequent path is different from the current case path, then *Question Moderator* asks questions for the information required to follow the same path as a frequent path. On the users' response to the question, the current case state is resumed to process the case in the appropriate context and reduced the execution time. Additionally, it tracks the path followed by each case and logs the path information in the *Cases Path Log* repository for finding a frequent path in RDR tree.

3.3 Frequent Path Miner

Different cases follow diverse branches of RDR tree based on associated facts. Therefore, *Frequent Path Miner* module is devised to find path followed pattern in RDR tree and keep the *Frequent Path Repository* up to date. It finds a frequent path for each branch of RDR tree using the path followed by different cases from *Case Path Log*. Frequent path presents the sequence of node in a specific branch of RDR tree that are followed by most of the similar patients. Finding a most used path for each new case arrival is a time consuming and a tedious task, which degrades the performance in terms of execution time. The frequent path may not change very frequently, therefore we performed this task in an offline manner for better performance on the basis of a predefined schedule.

Algorithm 1 shows the required steps for predicting missing information in user input/patient case. The algorithm extracts facts from the patient case at line 1. The extracted facts are matched with RDR rules to find a matched rule. Based on matched rule the algorithm finds a frequent path from the *Frequent Path Repository* (frequentPaths). Using the information of frequent path, the algorithm computes the difference between provided facts and facts from the matched rule and considers as missing information. The missing facts are added to the patient case according to users' acknowledgment, which leads to the most appropriate result. The advantages of our approach are interpreting user query in the given context, considering other similar cases, asking for missing information and preserving the current case status.

Algorithm 1. Missing Information Prediction

Input: *patientCase, FrequentPaths*

Output: *facts #input* **+** *resolved missing facts*

1. *facts ← extractFacts(patientCase);*
2. *mappedRule ← relatedRule(facts);*
3. *frequentPath←frequentPath(mappedRule, frequentPaths);*
4. *predictedRule ← frequentPath.getRule();*
5. *missingFacts ← difference (predictedRule.Facts, facts);*
6. ***for each*** *fact* **in** *missingFacts* ***do***
7.   *acknowledged ← forwardToUser(fact);*
8.   ***if*** *acknowledge, do* ***then***
9.   *facts.add(fact);*
10.   ***end if***
11. ***end***

4 Realization of the Proposed Methodology

To realize the proposed methodology, a diabetes diagnosis scenario is described. The example scenario consists of 7 rules, as shown in Fig. 2. We consider a patient case with facts Hyperursis, Thirst, Unexplained Weight Loss and HbA1c = 6.7. The steps required for the methodology are given as follows.

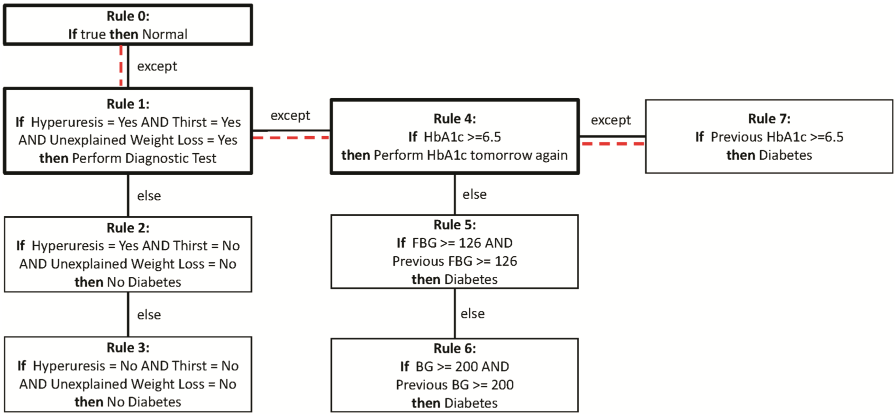

Fig. 2. Partial diabetes knowledge representation using RDR

4.1 Initial Recommendation Generation

The initial recommendation will be generated based on the aforementioned provided input. *Case Analyzer* starts matching the given facts with the rules stored in *RDR Rule Base*. First, it would check the Rule 0 stored at the root node, known as a default rule of the RDR tree. The condition of the rule satisfied therefore, it checks the next level i.e. Rule 1 and continue checking until no further rules' conditions satisfied. For the provided facts, Rule 4 is the last satisfied rule represented by bold boxes in Fig. 2. The system generates "Perform HbA1c tomorrow again" as the initial recommendation. This information is passed to *Question Moderator* for analysis and finding missing information.

4.2 Question Moderation

Question Moderator uses the generated initial recommendation matched rule path and *Frequent Path* to predict missing information. The frequent path for Rule 4 is up to Rule 7 as shown by dotted line in Fig. 2. The *Question Moderator* finds required facts for Rule 7 that are not provided as an initial input. In this example, "Previous HbA1c" is detected as a missing fact. Therefore, the system asks the user about the existence of "Previous HbA1c" and would expect a response from the user. Users' subsequent response would facilitate the system in order to generate a final recommendation.

4.3 Final Recommendation Generation

Starting from the previously fired rule (Rule 4) the users' response is mapped with child rules of Rule 4. As depicted in Fig. 2, most of the patients have "Previous HbA1c" value, but the user was not considering it as important fact but only rely on "Current HbA1c" value. While taking an appropriate decision “Previous HbA1c” value has a key role. If the user provides Pervious HbA1c value greater than 6.5, Rule 7 will be fired and the system will respond “Diabetes” as a final recommendation. Therefore, the system produces a more accurate result by predicting missing information in users’ input, which leads to a better decision for treatment and diagnosis.

5 Result and Discussion

We performed the evaluation of the proposed system based on systems’ usage data for three weeks. The result is based on 300 anonymized real diabetic patients’ data. The results achieved compared with conventional RDR based CDSS system, as depicted in Fig. 3. At the first week, both systems achieved accuracy 86% because the proposed system does not have *Frequent Path* data. At the second week, the conventional system accuracy increased to 86.7% due to the RDR rules modification and maintenance by domain experts. Based on the first week of usage, our system calculates the *Frequent Path*, which increased the proposed system accuracy up to 87%. Similarly, after third week the accuracy of the conventional system increased to 87% due to rules modifications. However, our proposed system accuracy increased to 87.5% because of *Frequent Path* updation which leads to accurate prediction of missing facts and increased overall system accuracy.

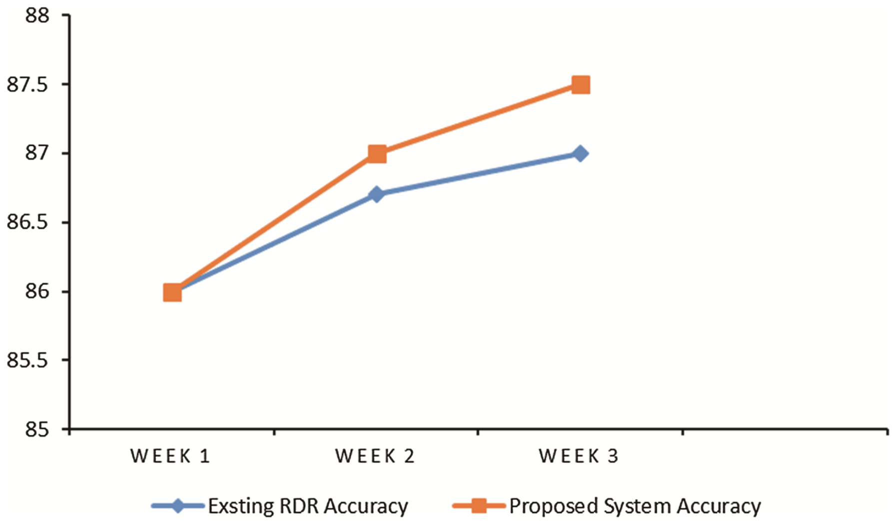

Fig. 3. Result comparison

The preliminary result of three weeks concluded that the proposed system will have greater effects and achievement in improving system accuracy with the systems’ usage and passage of time.

One of the major challenge faced is long waiting time for getting user response. The system waits for the answer of the query until user response. If a user doesn't respond for a long time, the system will remain in waiting state. To overcome this issue, we set 2 min waiting time as a threshold time. When the user is not able to answer in 2 min, the system gives a primitive result based on the initial values.

6 Conclusion and Future Work

The proposed system considers similar cases to give the appropriate recommendation with better performance and accuracy. Additionally, it is capable of generating primitive recommendations based on the initial incomplete information. The similar cases help in finding missing information and the system asks questions for those missing information to reach the appropriate recommendation. The advantages of the proposed technique include accurately predicting missing information by considering similar cases, interacting with the user for missing information, delivering most appropriate recommendations and reducing execution time. In the future, we will evaluate and enhance the proposed system to deploy in different hospitals for timely decision making in diagnosis and treatment.

Acknowledgments. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2011-0030079) and by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2017-0-00655).

References

1. Wright, A., Sittig, D.F., Ash, J.S., Sharma, S., Pang, J.E., Middleton, B.: Clinical decision support capabilities of commercially-available clinical information systems. J. Am. Med. Inform. Assoc. **16**, 637–644 (2009)
2. Caballero-Ruiz, E., García-Sáez, G., Rigla, M., Villaplana, M., Pons, B., Hernando, M.E.: A web-based clinical decision support system for gestational diabetes: automatic diet prescription and detection of insulin needs. Int. J. Med. Inf. **102**, 35–49 (2017)
3. Jung, H., Yang, J., Woo, J.-I., Lee, B.-M., Ouyang, J., Chung, K., Lee, Y.: Evolutionary rule decision using similarity based associative chronic disease patients. Clust. Comput. **18**, 279–291 (2015)
4. Kemppinen, J., Korpela, J., Elfvengren, K., Polkko, J.: Clinical decision support system for opioid substitution therapy. In: 2015 48th Hawaii International Conference on System Sciences (HICSS), pp. 3054–3063. IEEE (2015)
5. Zhou, Q.: A clinical decision support system for metabolism synthesis. In: International Conference on Computational Intelligence and Natural Computing, 2009, CINC 2009, pp. 323–325. IEEE (2009)
6. Gaines, B.: Induction of ripple-down rules. In: Proceedings of the 5th Australian Joint Conference on Artificial Intelligence. World Scientific (1992)
7. Hyeon, J., Oh, K.-J., Kim, Y.J., Chung, H., Kang, B.H., Choi, H.-J.: Constructing an initial knowledge base for medical domain expert system using induct RDR. In: 2016 International Conference on Big Data and Smart Computing (BigComp), pp. 408–410. IEEE (2016)

8. Hussain, M., Khattak, A.M., Khan, W.A., Fatima, I., Amin, M.B., Pervez, Z., Batool, R., Saleem, M.A., Afzal, M., Faheem, M.: Cloud-based Smart CDSS for chronic diseases. Health Technol. **3**, 153–175 (2013)
9. Glina, E.M., Kang, B.H.: Conversation system with state information. University of Tasmania (2009)
10. Nguyen, D.Q., Nguyen, D.Q., Pham, S.B.: Ripple Down Rules for question answering. Semantic Web, pp. 1–22 (2015)
11. Richards, D.: Two decades of ripple down rules research. Knowl. Eng. Rev. **24**, 159–184 (2009)