Computer Systems Science & Engineering
DOI:10.32604/csse.2022.017221
Article

Tech Science Press

Effective Hybrid Content-Based Collaborative Filtering Approach for Requirements Engineering

Qusai Y. Shambour*, Abdelrahman H. Hussein, Qasem M. Kharma and Mosleh M. Abualhaj

Faculty of Information Technology, Al-Ahliyya Amman University, Amman, 19328, Jordan
*Corresponding Author: Qusai Y. Shambour. Email: q.shambour@ammanu.edu.jo
Received: 24 January 2021; Accepted: 04 March 2021

Abstract: Requirements engineering (RE) is among the most valuable and critical processes in software development. The quality of this process significantly affects the success of a software project. An important step in RE is requirements elicitation, which involves collecting project-related requirements from different sources. Repositories of reusable requirements are typically important sources of an increasing number of reusable software requirements. However, the process of searching such repositories to collect valuable project-related requirements is time-consuming and difficult to perform accurately. Recommender systems have been widely recognized as an effective solution to such problem. Accordingly, this study proposes an effective hybrid content-based collaborative filtering recommendation approach. The proposed approach will support project stakeholders in mitigating the risk of missing requirements during requirements elicitation by identifying related requirements from software requirement repositories. The experimental results on the RALIC dataset demonstrate that the proposed approach considerably outperforms baseline collaborative filtering-based recommendation methods in terms of prediction accuracy and coverage in addition to mitigating the data sparsity and cold-start item problems.

Keywords: Requirements engineering; recommender systems; requirements elicitation; collaborative filtering; content-based filtering

1 Introduction

Requirements engineering (RE) is a critical phase in software development. It is an iterative process of eliciting, analyzing, specifying, validating, and managing the requirements of stakeholders. RE is an error-prone and time-consuming process, and it is considered one of the key reasons for the success of a project. A poor RE process in a software project may 1) cause important deadlines to be missed, 2) increase costs that can surpass the project's budget, or 3) lead to project failure due to missing or incomplete requirements [1,2].

Requirements elicitation is one of the important activities in RE. It involves the discovery and collection of a stakeholder's needs and expectations from different sources. Failure to determine software requirements may potentially lead to missing requirements; the latter is among the major causes of software project failure. Requirements reuse has been recognized as an effective elicitation technique that can enhance quality

specifications for software requirements. Requirements reuse aims to gather requirements that have been specified for previous projects and then use them in new related projects with similar goals and functionalities. The benefits of reusing requirements are as follows: 1) improvement in requirements elicitation efficiency in terms of saving time and cost; 2) reused requirements are easier to understand; 3) faster time to market; 4) reused requirements exhibit higher quality, and they are complete and easy to verify; and 5) lower development cost because it enables the reuse of designs, codes, and tests [3]. Therefore, requirements reuse can increase the success rate of software projects by supporting RE stakeholders in mitigating the risk of missing requirements through the identification of all possible requirements of the software under development.

Consequently, the demand for applying intelligent techniques to support stakeholders in effectively identifying and reusing requirements is high and imperative. In particular, recommender systems (RSs) are considered effective intelligent techniques that can support software stakeholders to work effectively in a number of RE processes, particularly supporting stakeholders in requirements elicitation [2,4–7]. RSs are recognized as efficient knowledge-filtering tools for addressing the information overload issue in various real-world applications, including e-commerce, e-learning, e-health, e-government, e-tourism, e-entertainment, and e-library. RSs assist in retrieving the most suitable items (e.g., requirements, movies, business partners, websites, restaurants, hotels, learning courses, books, and jobs) for a particular user (e.g., stakeholder, buyer, business, tourist, learner, and employer) from a substantial amount of items by utilizing the user's historical records to predict his/her interest in an item [8–14].

Various recommendation methods have been widely used to implement RSs. The most well-known recommendation methods are collaborative filtering (CF)-, content-, knowledge-, and hybrid-based recommendation methods. CF-based recommendation methods utilize user–item interactions to provide recommendations to users on the basis of user–user or item–item similarities. These methods can be divided into memory-based and model-based CF techniques. Memory-based CF techniques can be further divided into user-based and item-based CF techniques. User-based CF provides recommendations to users of items that have been liked by similar users with similar interests. Item-based CF provides recommendations to users of items that are similar to those that they have liked in the past. Different metrics can be applied to calculate similarity among users or items. These metrics include Pearson correlation, constrained Pearson correlation cosine similarity, and adjusted cosine similarity. Content-based recommendation methods analyze the description of items from a user profile to make recommendations. They work by first comparing the attributes of potential items with those of the items listed in the user profile. Then, potential items that are highly similar to the items in the user profile are recommended. Knowledge-based recommendation methods consider functional knowledge about users, items, and their relationships to make recommendations. Hybrid-based recommendation methods integrate two or more recommendation methods to utilize their advantages and eliminate their weaknesses. In practice, many hybrid recommendation methods combine CF- and content-based methods, including item-based and user-based CF with content-based filtering [13,15].

In summary, the major contributions of this research are as follows.

1. An effective hybrid content-based collaborative filtering (HCBCF) recommendation approach is developed to support project stakeholders in mitigating the risk of missing requirements during requirements elicitation.
2. The HCBCF approach is a powerful and intelligent tool for reducing the effect of the information overload problem that is encountered during the process of identifying all potential related requirements of the software under development from repositories with a large number of reusable requirements.

3. The HCBCF approach incorporates two recommendation methods: item-based CF and content-based filtering. It utilizes the content information of items to enhance recommendation performance and reduce the effects of the data sparsity and cold-start item problems when adequate rating data are unobtainable.
4. The experimental results demonstrate that the HCBCF approach considerably outperforms benchmark collaborative filtering-based recommendation approaches in terms of prediction accuracy and coverage in addition to alleviating the data sparsity and cold-start item problems.

The remainder of this paper is structured as follows. Section 2 presents related work in this field. Section 3 describes the proposed HCBCF approach. Section 4 provides the experimental evaluation of the HCBCF approach compared with baseline CF-based recommendation methods. Section 5 presents the research conclusions and plans for future work.

2 Literature Review

Survey studies have been conducted regarding the adoption of RSs to support various RE activities, such as stakeholder identification, requirements elicitation, requirements prioritization, requirements negotiation, and release planning [2,6,7,16–19].

Stakeholder recommendation is a significant activity during the early stage of an RE process because a low degree of user participation leads to project failure. One example is the StakeNet approach proposed by Lim et al. [20] that utilizes social network analysis to produce recommendations for project stakeholders. In StakeNet, an initial set of stakeholders are invited to recommend other stakeholders and identify their roles that can influence or be influenced by a software project. This information is then used to construct a social network in which nodes represent stakeholders and links between nodes represent recommendations expressed by stakeholders. The social network is used to prioritize stakeholders who will be included in the project on the basis of their project influence by using different social network measures. An extension of this work, known as StakeRare, was proposed in Lim et al. [21]; in StakeRare, a CF-based RS is developed to discover and prioritize requirements in large software projects. The identified stakeholders are then asked to select and rate an initial set of requirements. Subsequently, the CF-based approach is used to recommend other relevant requirements. Finally, the recommended requirements are prioritized using the stakeholders' ratings weighted by their project influence in the social network. Hariri et al. [22] highlighted two applications of the implementation of RSs in RE. The first application supports requirements discovery in online discussion forums by recommending relevant discussion topics to stakeholders and expert stakeholders for each specific topic. The second application utilizes product information from publicly available websites to build an RS that can recommend domain-specific features for a specific product.

With regard to requirements elicitation, Portugal et al. [23] presented a content-based RS for GitHub projects to address the time-consuming task of manually examining related projects. The proposed system utilizes natural language processing techniques on the information found in GitHub's README.TXT to recommend related GitHub projects to existing projects under development. AlZu'bi et al. [24] proposed an efficient RS that extracts rules from user requirements based on an *a priori* algorithm to recommend new requirements to users. Similarly, Muhairat et al. [5] presented an intelligent RS to enhance the accuracy and completeness of requirements during requirements elicitation on the basis of association rule analysis, namely, the frequent pattern growth method. The proposed system uses association rule mining to discover requirements that already exist in different systems. Such requirements can be used for similar projects under development. Ramos et al. [25] developed a CF-based RS to assist Scrum practitioners in identifying nonfunctional requirements early in RE. The proposed system uses historical data collected from Scrum-based projects to recommend related nonfunctional requirements to an existing

project under development. Shambour et al. [26] addressed the information overload problem that is intrinsic in requirements elicitation by proposing a hybrid user–item-based CF recommendation algorithm. The proposed recommendation algorithm, which is a hybrid of the improved versions of a user-based CF method and an item-based CF method, supports requirement engineers in identifying related reusable requirements for a current project under development from large-scale requirement repositories. The experimental results demonstrated the effectiveness of the proposed algorithm by outperforming a number of CF-based recommendation approaches in terms of prediction accuracy, recommendation precision, recall, and F1 measures.

The appropriate prioritization and assignment of all the resources and requirements in any software project are the key to achieving a smooth and successful project schedule. This process includes identifying and prioritizing the requirements of a project, appointing appropriate stakeholders for the requirements, and release planning. However, the manual prioritization of a huge number of requirements is a time-consuming and demanding process [2]. To address this issue, Ninaus [27] proposed adopting group recommendation technologies that use the ranks of requirements provided by stakeholders to prioritize requirements. This previous study investigated a number of recommendation heuristics regarding requirements prioritization. Moreover, the author presented a new heuristics for group recommendations with improved prediction quality. Thereafter, Ninaus et al. [28] presented an RE environment based on different recommendation approaches, called IntelliReq, which supports stakeholders in requirement-related tasks, such as requirements definition, requirements reuse, release planning, and quality assurance of requirements. Ahmad et al. [29] presented an RS for requirements negotiation and prioritization. This system has three steps: identification of stakeholders, listing of functional and nonfunctional requirements, and elicitation of decision maker's weight by using inverse function arithmetic and graded mean methods. Samer et al. [30] presented a group recommendation approach that focuses on enhancing requirements prioritization through flexible preferences elicitation and by implementing innovative user interfaces that facilitate knowledge sharing among stakeholders. The evaluation results showed that the proposed approach enhances the quality of requirements prioritization and positively influences the success rate of software projects that were implemented within the scope of the empirical user study conducted in this research. Palomares et al. [19] introduced an RS that is intended to facilitate various RE phases, such as requirements elicitation, requirements analysis, requirements management, and negotiation. Diverse recommendation techniques were used for each RE activity to provide recommendations to stakeholders. In general, software projects have a huge number of requirements that are frequently related to one another. The detection of such dependencies among requirements is a time-consuming and cognitively challenging task that requires the use of intelligent solutions [31]. In this regard, Samer et al. [31] presented two content-based recommendation techniques that automatically detect and recommend requirement dependencies. The first technique identifies possible dependencies among requirements by applying document classification approaches, such as linear support vector machine, random forest, naive Bayes, and *k*-nearest neighbors. The second technique uses a latent semantic analysis approach.

3 HCBCF Recommendation Approach

We address the identification of related reusable requirements from large requirement repositories as a recommendation problem, as presented by Adomavicius et al. [32]. Let S be the set of all stakeholders, and R be the set of all possible related requirements that can be recommended. The utility function T that measures the suitability of a requirement $r \in R$ to a stakeholder $s \in S$ can be defined as $S \times R \rightarrow T$, where T is a nonnegative real number within a definite range. Then, each stakeholder s must be able to estimate the utility function $T(s, r)$ for a requirement r for which $T(s, r)$ is not yet known. A stakeholder can select one requirement or a set of requirements that will maximize his/her utility. Formally,

$$\forall s \in S,\ r = \arg\max_{r \in R}\ T(s,r) \tag{1}$$

Subsequently, we explain the major steps of the proposed HCBCF approach. This approach has three major steps: item-based CF similarity, item-based content similarity, and hybrid prediction. Hereafter, stakeholders are denoted as users and requirements are denoted as items. Assume that each stakeholder will give his/her ratings on a preliminary set of requirements to reflect its relevance to the project under consideration. Then, the HCBCF approach is applied to predict other requirements that a stakeholder may be interested in accordance with the project under investigation.

Step 1: Calculate Item-Based CF Similarity.

In this step, a set of items rated by an active user is first analyzed to identify items that are most similar to them in terms of their ratings. Then, k most similar items for each rated item are selected. On the basis of the user–item rating matrix, the mean square differences (MSD) method is used to quantify the extent of direct implicit similarity between items a and b by evaluating the prediction accuracy of item a as a sole recommender for item b. That is, if an accurate recommendation of item a is obtained using only item b as its neighbor, then items a and b should receive a high implicit similarity score. In this regard, the following Resnick's prediction method [33] is utilized to calculate the predicted rating of an item that uses only one neighbor item:

$$P_{u,a} = \bar{r}_a + (r_{u,a} - \bar{r}_b) \tag{2}$$

where $r_{u,a}$ denotes the rating of item a by user u; and $\bar{r}_a$ and $\bar{r}_b$ are the mean ratings of items a and b, respectively. Thereafter, the MSD method is used to measure the degree of implicit similarity on the basis of the difference between the predicted and actual rating scores of common users who have rated both items $U_{a\cap b}$. The predicted rating $P_{u,a}$ and actual rating $r_{u,a}$ scores are normalized using the max–min normalization approach to ensure that the value of $MSD_{a,b} \in [0,1]$.

$$iSim_{a,b}^{\text{MSD}} = 1 - \left(\frac{\sum_{u=1}^{|U_{a\cap b}|} (P_{u,a} - r_{u,a})^2}{|U_{a\cap b}|} \right) \tag{3}$$

However, the aforementioned MSD metric is a rating-based similarity measure that considers only ratings from co-rated users. That is, it disregards the number of co-rated users, which is an important factor. In a rating-based similarity measure, the similarity among items calculated on the basis of a few users or one user is well thought-out and equally imperative as one calculated on the basis of more co-rated users. Accordingly, using only the MSD measure can misrepresent similarity among items, and thus, produce inaccurate predictions. To solve this issue, a structural similarity measurement [34] that considers the number of co-rated users when calculating similarity among items must be integrated to develop an improved similarity measurement that will increase prediction accuracy. For this purpose, the Ochiai similarity measure, shown in Eq. (4) and which considers the percentage of the number of co-rated users $U_{a\cap b}$ to the number of users who have rated each item U_a and U_b, is selected to be incorporated into the MSD metric as shown in Eq. (5).

$$iSim_{a,b}^{\text{Ochiai}} = \frac{|U_{a\cap b}|}{\sqrt{|U_a| \times |U_b|}} \tag{4}$$

$$iSim_{a,b}^{CF} = iSim_{a,b}^{MSD} \times iSim_{a,b}^{\text{Ochiai}} \tag{5}$$

Step 2: Calculate Item-Based Content Similarity.

In this step, we suppose that all the items are assigned to predefined categories as meta-information. On the basis of the items' categories, when two items belong to the same category, they are implied to be related (similar) to each other. Accordingly, item-based content similarity uses the categorical representations of a set of items rated by an active user to a new set of items that the active user have not investigated until now [7]. Such related items can be potentially recommended to the active user.

For example, assume that active stakeholder *a* has already investigated requirements *R1* and *R2*, which have the following assigned categories: user interface and user privileges, respectively (Tab. 1). By contrast, requirements *R3* and *R4*, which have not been investigated yet by stakeholder *a*, can be potentially recommended to stakeholder *a* because they belong to the same categories (similar requirements) as requirements *R1* and *R2*, respectively.

Table 1: Example of content-based filtering

ID	Requirement Description	Requirement Category
R1	A user shall be able to access and print their booking confirmation details.	User Interface
R2	An administrator shall be able to add a new flight to the system.	User Privileges
R3	A user shall be able to search for flights for a trip.	User Interface
R4	An administrator shall be able to view the customers' booking details.	User Privileges

For this purpose, every item is first represented as a vector of binary values [0,1], as depicted in Eq. (6).

$$\overrightarrow{V}_a = (v_{a,1}, v_{a,2}, \ldots.., v_{a,c}), \quad where$$
$$v_{a,c} = \left\{ \begin{array}{ll} 1 & if\ item\ a\ belongs\ to\ category\ C \\ 0 & if\ item\ a\ doesn't\ belong\ to\ category\ C \end{array} \right\} \tag{6}$$

Then, the standard vector-based cosine similarity [35] is utilized as shown in Eq. (7) to calculate item-based content similarity between any pair of items.

$$iSim_{a,b}^{Content} = \frac{\sum_{u=1}^{n} \overrightarrow{V}_a \times \overrightarrow{V}_b}{\sqrt{\sum_{u=1}^{n} (\overrightarrow{V}_a)^2} \times \sqrt{\sum_{u=1}^{n} (\overrightarrow{V}_b)^2}} \tag{7}$$

Step 3: Calculate Hybrid Predicted Ratings.

In this step, the predicted ratings of an unknown item *a* for an active user *u* is divided into two substeps. First, the weighted sum approach [35], which computes the sum of the ratings given by an active user *u* to the most similar items b∈*I* similar to the target unknown item *a*, is used twice. That is, once for the item-based CF approach and once for the item-based content approach to calculate the predicted ratings, as given by Eqs. (8) and (9), respectively.

$$P_{u,a}^{CF} = \frac{\sum_{b \in I} (r_{u,b} \times iSim_{a,b}^{CF})}{\sum_{b \in I} \left| iSim_{a,b}^{CF} \right|} \tag{8}$$

$$P_{u,a}^{Content} = \frac{\sum_{b \in I} (r_{u,b} \times iSim_{a,b}^{Content})}{\sum_{b \in I} \left| iSim_{a,b}^{Content} \right|} \tag{9}$$

Then, the linear weighted hybridization method [36] is used to combine the predicted ratings of the item-based CF approach and the item-based content approach to produce a single final prediction as follows:

$$P_{u,a}^{Hybrid} = \lambda \cdot P_{u,a}^{CF} + (1 - \lambda) \cdot P_{u,a}^{Content} \tag{10}$$

where λ and 1−λ∈ [0,1] denote the relative significance of the item-based CF approach and the item-based content approach, respectively, on the final predicted rating. Selecting an appropriate value for λ is achieved by performing the sensitivity analysis described in Section 4.3.2.

4 Experimental Design and Results

The experimental datasets, validation metrics, benchmark approaches, and experimental results of the proposed HCBCF recommendation approach are presented in this section.

4.1 Datasets and Evaluation Metrics

The RALIC dataset [37], which was collected for a large software project initiated at the University College London, is used for validation. The project involved 60 groups of stakeholders and approximately 30,000 students; its duration was more than 2 years. The RALIC dataset contains 3,113 ratings from 76 stakeholders on 104 specific requirements. The ratings are presented in an integer scale of 1 to 5 and provided in the following format: stakeholder name, requirement, rating. Specific requirements can be categorized into a two-level hierarchical structure, with the project objectives as item categories and specific requirements as item instances. The categories of requirements include the following: user experience, card design, security and access control, improved processes, reduced cost, compatibility with existing systems, data quality, extensibility, delivery activities, and technical constraints.

Mean absolute error (MAE), root-mean-square error (RMSE), and recommendation coverage are used to evaluate recommendation performance in terms of prediction accuracy and coverage. MAE measures prediction accuracy by averaging the absolute differences between the actual and predicted ratings. RMSE measures prediction accuracy by averaging the squared differences between the actual and predicted ratings. Prediction accuracy is higher when the values of MAE and RMSE are lower. Recommendation coverage measures the percentage of items in which a given recommendation approach is able to make a prediction [38].

4.2 Benchmark Approaches

To demonstrate the effectiveness of the HCBCF approach, three benchmark recommendation approaches are selected as follows:

1. the item-based CF approach that applies cosine-based similarity among requirements to make recommendations (ICF-C) [35];
2. the item-based CF approach that applies adjusted cosine-based similarity among requirements to make recommendations (ICF-AC) [35]; and

3. the user-based CF approach proposed by Lim and Finkelstein [21], i.e., StakeRare, which applies Pearson correlation among stakeholders to recommend relevant requirements to them.

4.3 Experimental Results

4.3.1 Optimal Neighborhood Size

To understand how prediction accuracy varies with neighborhood size and identify the optimal neighborhood size, an experiment is conducted by changing the number of neighbors from 2 to 20 and computing the corresponding MAE. The neighborhood size that corresponds to the minimum MAE value is considered the optimal neighborhood size. Fig. 1 indicates that the optimal neighborhood size that achieves the best performance (i.e., lowest MAE) for the proposed HCBCF approach on the RALIC dataset is four neighbors. Hereafter, a neighborhood size of four is set as the optimal value for achieving the optimum performance of the proposed HCBCF approach.

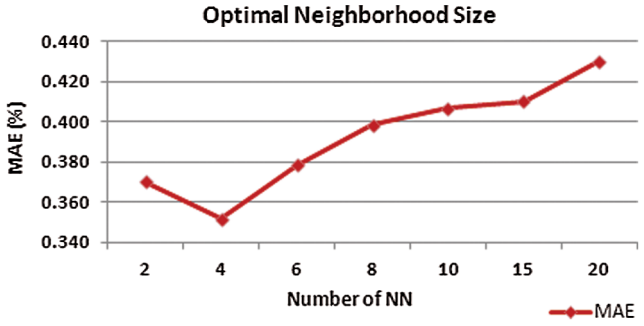

Figure 1: Identification of the optimal neighborhood size

4.3.2 Optimal Value of λ

To understand how prediction accuracy varies with different values of λ and identify the optimal value of the integration parameter λ in the linear weighted formula, an experiment is conducted by varying the values of λ from 0 to 1 and computing the corresponding MAE. The value of λ with the minimum MAE is considered the optimal value of λ. Fig. 2 shows that the optimal value of λ that attains the best performance (i.e., lowest MAE) for the proposed HCBCF approach on the RALIC dataset is 0.5. Henceforth, a value of 0.5 of the integration parameter λ is set as the optimal value for achieving the optimum performance of the proposed HCBCF approach.

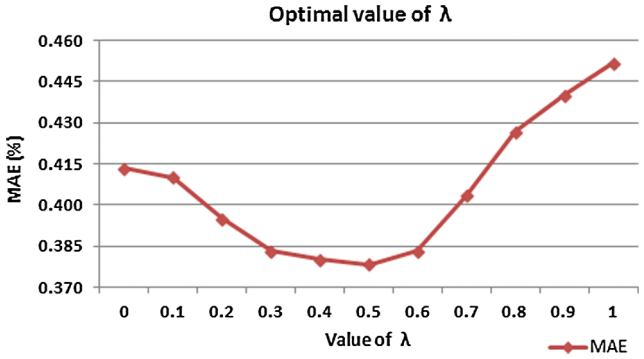

Figure 2: Identification of the optimal value of λ

4.3.3 Comparing the Prediction Accuracy of the Proposed HCBCF with Those of Benchmark Approaches

Two experiments are performed to compare the performance of the proposed HCBCF approach in terms of prediction accuracy with those of benchmark recommendation approaches. In both experiments, MAE and RMSE are computed with respect to varying number of neighbors for all the recommendation approaches. As illustrated in Figs. 3 and 4, the HCBCF approach outperforms the benchmark approaches because it achieves the highest prediction accuracy in all the neighborhood sizes. Fig. 3 shows that the corresponding improvement percentages of the HCBCF approach in terms of MAE are 29.3%, 14.9%, and 26% compared with the ICF-C, ICF-AC, and StakeRare approaches, respectively. Fig. 4 indicates that the corresponding improvement percentages of the HCBCF approach in terms of RMSE are 20.2%, 19.7%, and 20.4% compared with the ICF-C, ICF-AC. and StakeRare approaches, respectively. Accordingly, the HCBCF approach is confirmed as highly effective in terms of prediction accuracy.

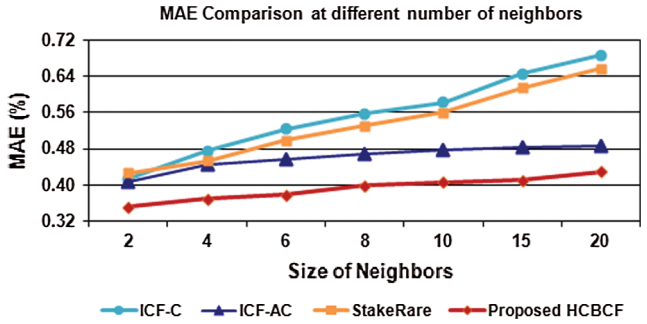

Figure 3: Comparison between the MAE of HCBCF and those of the benchmark methods

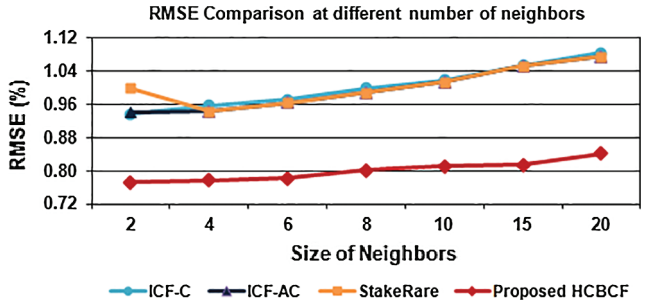

Figure 4: Comparison between the RMSE of HCBCF and those of the benchmark methods

4.3.4 Effectiveness of HCBCF on the Data Sparsity Problem

To validate the effectiveness of HCBCF in alleviating the data sparsity problem in terms of prediction accuracy and coverage, two experiments are performed on six datasets with varied data sparsity levels from the lowest level of 97.0% to the highest level of 99.5% (i.e., 97%, 97.5%, 98.0%, 98.5%, 99.0%, and 99.5%). As shown in Figs. 5 and 6, the HCBCF approach outperforms the benchmark recommendation approaches, mostly on the exceptionally sparse datasets, by attaining better results in terms prediction accuracy and coverage at all data sparsity levels. Fig. 5 indicates that the corresponding improvement percentages of the HCBCF approach in terms of MAE are 48.2%, 52.3%, and 55.6% compared with the ICF-C, ICF-AC, and StakeRare approaches, respectively. Fig. 6 illustrates that the corresponding improvement percentages of the HCBCF approach in terms of prediction coverage are 49.7%, 58% and 64.5% compared with the ICF-C, ICF-AC, and StakeRare approaches, respectively. The HCBCF approach is extremely robust when dealing with exceptionally sparse datasets. Therefore, in terms of

prediction accuracy and coverage, the HCBCF approach is highly effective in reducing the effect of the data sparsity problem.

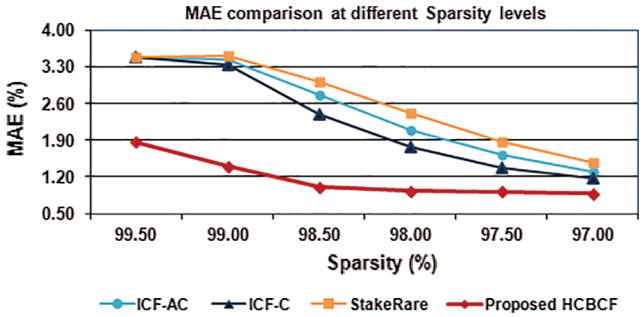

Figure 5: Comparison between the MAE of HCBCF and those of the benchmark methods at different sparsity levels

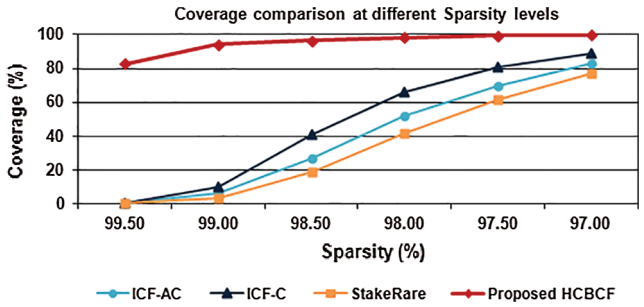

Figure 6: Comparison between the coverage of HCBCF and those of the benchmark methods at different sparsity levels

4.3.5 Effectiveness of HCBCF on the Cold-Start Item Problem

To validate the effectiveness of HCBCF in alleviating the cold-start item problem in terms of prediction accuracy and coverage, two experiments are performed on six datasets with a varied number of ratings for new items from the minimum of 2 ratings to a maximum of 25 ratings (i.e., 2, 5, 10, 15, 20, and 25). As shown in Figs. 7 and 8, the HCBCF approach outperforms the benchmark recommendation approaches by achieving better results in terms of prediction accuracy and coverage at different numbers of ratings for new items. Fig. 7 shows that the corresponding improvement percentages of the HCBCF approach in terms of MAE are 12.2%, 17.2%, and 17.9% compared with the ICF-C, ICF-AC, and StakeRare approaches, respectively. Fig. 8 indicates that the corresponding improvement percentages of the HCBCF approach in terms of prediction coverage are 44.8%, 57.2%, and 58.7% compared with the ICF-C, ICF-AC, and StakeRare approaches. In terms of prediction accuracy and coverage, the HCBCF approach exerts a substantial effect when dealing with the cold-start item problem.

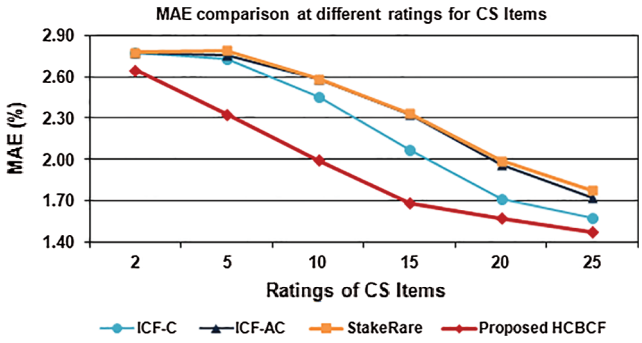

Figure 7: Comparison between the MAE of HCBCF and those of the benchmark methods at different ratings for CS items

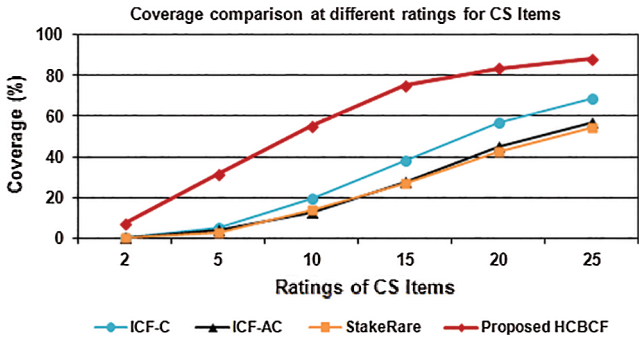

Figure 8: Comparison between the coverage of HCBCF and those of the benchmark methods at different ratings for CS items

5 Conclusions and Future Work

The primary contribution of this study is the development of the HCBCF recommendation approach, which supports requirements elicitation during the development of software projects. This approach assists project stakeholders in mitigating the risk of missing requirements during requirements elicitation by identifying related requirements from software requirement repositories. The HCBCF approach combines content-based filtering and item-based CF methods to utilize their advantages and eliminate their limitations. The item-based CF method considers structural similarity measurement (i.e., number of co-rated users) when computing item–item similarity to enhance prediction performance. The content-based filtering method applies the underlying relationships among items when rating data are inadequate to enhance the quality of recommendations by reducing the effects of the data sparsity and cold-start items problems. The experimental results on the RALIC dataset demonstrate that the proposed HCBCF approach considerably outperforms baseline CF-based recommendation approaches in terms of prediction accuracy and coverage in addition to mitigating the data sparsity and cold-start item problems. In the future, more thorough validation and comparisons will be performed on the HCBCF approach with other recommendation approaches in larger data sets. We also intend to implement the proposed approach in an RS that will be utilized by project stakeholders during requirements elicitation.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflict of interest to report regarding the present study.

References

[1] K. Kaur, P. Singh and P. Kaur, "A review of artificial intelligence techniques for requirement engineering," in *Computational methods and data engineering*, Singapore: Springer Singapore, pp. 259–278, 2021.

[2] R. Samer, M. Stettinger, A. Felfernig, X. Franch and A. Falkner, "Intelligent recommendation & decision technologies for community-driven requirements engineering," in *Proc. of the 9th Int. Conf. on Prestigious Applications of Intelligent Systems (PAIS@ECAI2020)*, Spain, pp. 1–9, 2020.

[3] C. Palomares, C. Quer and X. Franch, "Requirements reuse and requirement patterns: A state of the practice survey," *Empirical Software Engineering*, vol. 22, no. 6, pp. 2719–2762, 2017.

[4] C. Gupta, S. Singhal and A. Kumari, "I-way: A cloud-based recommendation system for software requirement reusability," in *Crowdsourcing and probabilistic decision-making in software engineering: Emerging research and opportunities*, Hershey, PA: IGI Global, pp. 23–34, 2020.

[5] M. Muhairat, S. ALZu'bi, B. Hawashin, M. Elbes and M. Al-Ayyoub, "An intelligent recommender system based on association rule analysis for requirement engineering," *J. UCS*, vol. 26, no. 1, pp. 33–49, 2020.

[6] N. H. Al-walidi, A. Khamis and N. Ramadan, "A systematic literature review of recommender systems for requirements engineering," *International Journal of Computer Applications*, vol. 175, no. 14, pp. 31–41, 2020.

[7] I. Williams and X. Yuan, "Recommender systems for software requirements engineering: Current research & challenges," in *Proc. of the 2019 SoutheastCon*, Huntsville, USA, pp. 1–6, 2019.

[8] J. Lu, D. Wu, M. Mao, W. Wang and G. Zhang, "Recommender system application developments: A survey," *Decision Support Systems*, vol. 74, pp. 12–32, 2015.

[9] Q. Shambour and J. Lu, "A trust-semantic fusion-based recommendation approach for e-business applications," *Decision Support Systems*, vol. 54, no. 1, pp. 768–780, 2012.

[10] Q. Shambour and J. Lu, "An effective recommender system by unifying user and item trust information for b2b applications," *Journal of Computer and System Sciences*, vol. 81, no. 7, pp. 1110–1126, 2015.

[11] S. Fraihat and Q. Shambour, "A framework of semantic recommender system for e-learning," *Journal of Software*, vol. 10, no. 3, pp. 317–330, 2015.

[12] Q. Shambour and J. Lu, "Government-to-business personalized e-services using semantic-enhanced recommender system," in *Electronic government and the information systems perspective*, Heidelberg: Springer, pp. 197–211, 2011.

[13] J. Lu, Q. Zhang and G. Zhang, *Recommender systems: Advanced developments*. Singapore: World Scientific, 2020.

[14] Q. Shambour, "Hybrid recommender systems for personalized government-to-business e-services," Ph.D. dissertation. University of Technology Sydney, Sydney, Australia, 2012.

[15] C. C. Aggarwal, "An introduction to recommender systems," in *Recommender systems: The textbook*. Switzerland: Springer International Publishing, pp. 1–28, 2016.

[16] J. G. Mohebzada, G. Ruhe and A. Eberlein, "Systematic mapping of recommendation systems for requirements engineering," in *Proc. of the Int. Conf. on Software and System Process*, Zurich, Switzerland, pp. 200–209, 2012.

[17] A. Felfernig, G. Ninaus, H. Grabner, F. Reinfrank, L. Weninger *et al.,* "An overview of recommender systems in requirements engineering," in *Managing requirements knowledge*, Berlin, Heidelberg, Berlin Heidelberg: Springer, pp. 315–332, 2013.

[18] G. Ninaus, F. Reinfrank, M. Stettinger and A. Felfernig, "Content-based recommendation techniques for requirements engineering," in *Proc. of the 2014 IEEE 1st International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)*, Karlskrona, Sweden, pp. 27–34, 2014.

[19] C. Palomares, X. Franch and D. Fucci, "Personal recommendations in requirements engineering: The openreq approach," in *Requirements engineering: Foundation for software quality, 10753*, Cham: Springer International Publishing, pp. 297–304, 2018.

[20] S. L. Lim, D. Quercia and A. Finkelstein, "Stakenet: Using social networks to analyse the stakeholders of large-scale software projects," *Proc. of the 32nd ACM/IEEE Int. Conf. on Software Engineering*, vol. 1, pp. 295–304, 2010.

[21] S. L. Lim and A. Finkelstein, "Stakerare: Using social networks and collaborative filtering for large-scale requirements elicitation," *IEEE Transactions on Software Engineering*, vol. 38, no. 3, pp. 707–735, 2012.

[22] N. Hariri, C. Castro-Herrera, J. Cleland-Huang and B. Mobasher, "Recommendation systems in requirements discovery," in *Recommendation systems in software engineering*, Berlin, Heidelberg: Springer, pp. 455–476, 2014.

[23] R. L. Q. Portugal, M. A. Casanova, T. Li and J. C. S. do Prado Leite, "Gh4re: Repository recommendation on github for requirements elicitation reuse," in *Proc. of the 29th Int. Conf. on Advanced Information Systems Engineering (CAiSE 2017)*, Essen, Germany, pp. 113–120, 2017.

[24] S. AlZu'bi, B. Hawashin, M. EIBes and M. Al-Ayyoub, "A novel recommender system based on apriori algorithm for requirements engineering," in *Proc. of the Fifth Int. Conf. on Social Networks Analysis, Management and Security (SNAMS)*, Valencia, Spain, pp. 323–327, 2018.

[25] F. Ramos, A. Costa, M. Perkusich, H. Almeida and A. Perkusich, "A non-functional requirements recommendation system for scrum-based projects," in *Proc. of the 30th International Conference on Software Engineering & Knowledge Engineering (SEKE)*, San Francisco Bay, USA, pp. 149–155, 2018.

[26] Q. Y. Shambour, M. M. Abu-Alhaj and M. M. Al-Tahrawi, "A hybrid collaborative filtering recommendation algorithm for requirements elicitation," *International Journal of Computer Applications in Technology*, vol. 63, no. 1/2, pp. 135–146, 2020.

[27] G. Ninaus, "Using group recommendation heuristics for the prioritization of requirements," in *Proc. of the sixth ACM conf. on Recommender systems*, Dublin, Ireland, pp. 329–332, 2012.

[28] G. Ninaus, A. Felfernig, M. Stettinger, S. Reiterer, G. Leitner *et al.,* "Intellireq: Intelligent techniques for software requirements engineering," in *Proc. of the 21 European Conf. on Artificial Intelligence (ECAI'14)*, Prague, Czech Republic, pp. 1161–1166, 2014.

[29] S. Ahmad and M. Sadiq, "Recommender systems for software requirements negotiation and prioritization," *International Journal of Computer Applications*, vol. 117, no. 13, pp. 1–7, 2015.

[30] R. Samer, M. Stettinger and A. Felfernig, "Group recommender user interfaces for improving requirements prioritization," in *Proc. of the 28th ACM Conf. on User Modeling, Adaptation and Personalization*, Genoa, Italy, pp. 221–229, 2020.

[31] R. Samer, M. Stettinger, M. Atas, A. Felfernig, G. Ruhe *et al.,* "New approaches to the identification of dependencies between requirements," in *Proc. of the 31st Int. Conf. on Tools with Artificial Intelligence (ICTAI)*, Portland, USA, pp. 1265–1270, 2019.

[32] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[33] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in *Proc. of the 1994 ACM Conf. on Computer Supported Cooperative*, Work, USA, pp. 175–186, 1994.

[34] D. Wang, Y. Yih and M. Ventresca, "Improving neighbor-based collaborative filtering by using a hybrid similarity measurement," *Expert Systems with Applications*, vol. 160, no. 2020, pp. 1–17, 2020.

[35] B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. of the 10th Int. Conf. on World Wide Web*, Hong Kong, pp. 285–295, 2001.

[36] R. Burke, "Hybrid web recommender systems," In: *The adaptive web: Methods and strategies of web personalization*, Berlin, Heidelberg: Springer-Verlag, pp. 377–408, 2007.

[37] S. L. Lim, "Social networks and collaborative filtering for large-scale requirements elicitation," Ph.D. dissertation. University of New South Wales, Sydney, Australia, 2010.

[38] F. O. Isinkaye, Y. Folajimi and B. A. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informatics Journal*, vol. 16, no. 3, pp. 261–273, 2015.